Alex Pisciotta

CONTROLE DE UM ROBÔ EQUILIBRISTA DE DUAS RODAS UTILIZANDO LÓGICA FUZZY

Taubaté, 2020

Alex Pisciotta

CONTROLE DE UM ROBÔ EQUILIBRISTA DE DUAS RODAS UTILIZANDO LÓGICA FUZZY

Dissertação apresentada para obtenção do Título de Mestre pelo curso de Mestrado Acadêmico em Engenharia Mecânica do Departamento de Engenharia Mecânica da Universidade de Taubaté.

Área de Concentração: Projeto Mecânico

Orientador: Prof. Dr. Álvaro Manoel S. Soares

Taubaté – SP

2020

SIBi – Sistema Integrado de Bibliotecas / UNITAU

P676c Pisciotta, Alex
Controle de um robô equilibrista de duas rodas utilizando lógica Fuzzy / Alex Pisciotta. -- 2020.
73 f. : il.

Dissertação (mestrado) – Universidade de Taubaté, Departamento de Engenharia Mecânica e Elétrica, 2020.
Orientação: Prof. Dr. Álvaro Manoel de Souza Soares, Instituto Básico de Ciências Exatas.

1. Robô Equilibrista. 2. Lógica Nebulosa. 3. Lógica Fuzzy. 4. Giroscópio. 5. Acelerômetro. I. Mestrado em Engenharia Mecânica. II. Titulo.

CDD – 670.4278

Ficha catalográfica elaborada **por Shirlei Righeti – CRB-8/6995**

ALEX PISCIOTTA

CONTROLE DE UM ROBÔ EQUILIBRISTA DE DUAS RODAS UTILIZANDO LÓGICA FUZZY

Dissertação apresentada para obtenção do Título de Mestre pelo curso de Mestrado Acadêmico em Engenharia Mecânica do Departamento de Engenharia Mecânica da Universidade de Taubaté, Área de Concentração: Projeto Mecânico.

Data: __22/08/2020____

Resultado: __Sem Restrições__

BANCA EXAMINADORA

Prof. Dr. _Álvaro Manoel de Souza Soares__ Universidade de Taubaté

Assinatura _____

Prof. Dr. _Luís Fernando de Almeida____ Universidade de Taubaté

Assinatura _____

Prof. Dr. _Sebastião Cardoso__ Instituto Tecnológico de Aeronáutica

Assinatura _____

Dedico este trabalho a André e Elice, filho e esposa, que apoiaram minhas horas de dedicação para que esse trabalho se tornasse realidade.

A Paulo Pisciotta, meu querido pai, a quem eu devo tudo o que sou, e que deixou este mundo tão repentinamente.

AGRADECIMENTOS

A Deus, por me despertar a vontade e a força necessárias, e me conservar saudável para continuar trabalhando e aprendendo.

Ao Professor Dr. Pedro Paulo Leite Prado, meu padrinho acadêmico e profissional, dando seu voto de confiança na indicação para o referido curso e para meu primeiro emprego como Engenheiro.

Ao Professor João Bosco Gonçalves por me acolher como primeiro orientador até seu desligamento da instituição, e me apresentou a lógica *Fuzzy* e me deu os primeiros indicativos na busca da solução deste desafio.

Ao Professor Dr. Álvaro Manoel S. Soares que gentilmente aceitou meu convite para orientar após a saída do professor João Bosco, e o fez com firmeza até a conclusão deste trabalho.

À Universidade de Taubaté, que me concedeu bolsa parcial durante o tempo regular deste curso e muito me ajudou.

A todos os familiares e colegas que me incentivaram a alcançar essa meta: Muito Obrigado!

RESUMO

Este trabalho apresenta o projeto e a construção de um robô equilibrista de duas rodas e discute o estudo de diferentes métodos de controle empregados, mostrando resultados práticos obtidos com a aplicação das estratégias de controle à um protótipo especialmente desenvolvido para esse fim. O protótipo utiliza uma unidade de medição inercial MPU-6050 que combina um acelerômetro de três eixos e um giroscópio de três eixos como elementos sensores do ângulo do robô, motores de corrente contínua como atuadores, acionados por um circuito integrado L293D para compatibilidade elétrica e um micro controlador, Arduino UNO R3, de 8 *bits* como unidade de processamento programado através de sua interface de desenvolvimento de software. Todos os circuitos do protótipo são alimentados por uma bateria de *Lithium-Polymerum* de 11,1V. As características eletrônicas de cada componente foram estudadas para permitir o desenvolvimento do circuito, e então o corpo mecânico foi construído. São apresentados dois tipos de filtro por software para tratamento dos sinais dos sensores, e adotado o Filtro de Kalman para combinar os dados do giroscópio e do acelerômetro. Com esse procedimento foram obtidas leituras mais estáveis de ambos os sensores. O sistema de controle abordado neste trabalho foi desenvolvido utilizando Lógica *Fuzzy*, também conhecida como Lógica Difusa ou Lógica Nebulosa, para se atingir o equilíbrio do robô em sua posição vertical, naturalmente instável. Após análise dos resultados, pode se notar uma ótima resposta do sistema após a aplicação da estratégia de controle.

Palavras-Chave: Robô Equilibrista. Lógica Nebulosa. Lógica Fuzzy. Giroscópio. Acelerômetro.

ABSTRACT

This work presents main characteristics regarding the construction of a two wheeled self-balancing robot and the different methods to its control, showing practical results obtained from the attempted control method in a prototype model specially designed for this work, using a MPU-6050 IMU (*Inertial Measurement Unit*) as sensor, which combines a 3 axis accelerometer and a 3 axis gyroscope, DC (*direct current*) motors as actuators, control driver L293D for electrical compatibility and an AVR 8 bit microcontroller as the processing unit with Arduino Integrated Development Environment (IDE) for programming it, with all the hardware being supplied by a 3S Lithium Polymeric battery. Electronics characteristics from each component are studied to allow hardware design, and so the mechanics body is assembled. Kalman Filter is studied to fuse data from gyroscope and accelerometer in order get stable measurements, once each sensor has peculiar weaknesses. Fuzzy Logic control method is studied and then implemented to keep the robot automatically in the vertical position, so the results are registered and commented.

Keywords: Self-balancing robot. Fuzzy Logic. Gyroscope. Accelerometer.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Um robô equilibrista de duas rodas é um sistema naturalmente instável (ESMAEILI, ALFI e KHOSRAVI, 2017), uma variação do sistema “carro com pêndulo invertido” (MARTÍNEZ, 2015) no qual deseja-se controlar sua estabilidade vertical com o centro de massa para cima enquanto o atuador corrige a posição da base para que o sistema fique na posição vertical de equilíbrio (BLOMSTEDT, HARALDSSON e NORDIN, 2016).

Este trabalho analisa um sistema mecânico naturalmente instável, com objetivo de se aplicar um método de controle através de Lógica Nebulosa. Os desafios do trabalho compreenderam os desenvolvimentos eletrônico, mecânico-estrutural, computacional e também a lógica de controle *Fuzzy*. Portanto as primeiras atividades consistiram na realização de pesquisas para compreensão dos conceitos envolvidos na solução deste tipo de problema, as possibilidades tanto em relação às características elétricas e mecânicas quanto aos métodos de controle estudados em outros trabalhos, assim como estudos em lógica nebulosa.

1.1 Objetivos do Trabalho

Os objetivos do trabalho são divididos em gerais e específicos, como segue.

1.1.1 Objetivos Gerais

Este trabalho visa explicar os conceitos envolvidos para o controle desse tipo de dispositivo enfatizando o método de controle por Lógica Nebulosa (também conhecida como Lógica Difusa, ou *Fuzzy Logic* em inglês) aplicada a um protótipo construído especialmente para esse propósito. São explorados os componentes mínimos necessários, como atuadores, unidade de controle e sensores, e então apresentados métodos para acondicionamento dos dados do sensor para sua correta interpretação pelo sistema de controle escolhido.

1.1.2 Objetivos Específicos

O objetivo específico deste trabalho foi estudar e implantar um sistema de controle nebuloso em um robô equilibrista especialmente projetado para este fim. Portanto, embora não seja o foco deste trabalho, inicialmente foi implantado um sistema de controle PID (Proporcional, integral e derivativo) para validar o sistema eletromecânico, para que, depois de atingido o equilíbrio por esse sistema de controle, o conjunto eletromecânico fosse validado e então submetido ao controlador por Lógica Nebulosa. Considerando-se o primeiro contato com os conceitos de Lógica Nebulosa, validar o sistema eletromecânico pela aplicação de outro método de controle conhecido demonstrou-se uma boa estratégia de mitigação de riscos.

Descrevem-se o processo de desenvolvimento e montagem do protótipo, a codificação do firmware, a execução de testes e as melhorias implantadas até se atingir o equilíbrio dinâmico por Lógica *Fuzzy*.

1.2 Delimitação do Trabalho

Este trabalho delimita-se ao desenvolvimento da plataforma de testes que se deseja equilibrar, compreendendo o sistema mecânico e eletrônico, bem como o desenvolvimento do firmware para obtenção do sistema de controle de equilíbrio na posição vertical. Este trabalho não propõe o controle de trajetória ou de posição absoluta do robô, ou qualquer forma de controle externo.

1.3 Estrutura do Trabalho

Este trabalho está estruturado em seis capítulos, sendo o primeiro capítulo dedicado à introdução do trabalho. O capítulo dois apresenta as características do sistema dinâmico de um robô equilibrista, e discorre sobre os principais componentes empregados. O terceiro capítulo apresenta os conceitos da lógica nebulosa, e o desenvolvimento do protótipo junto dos resultados são tratados no quarto capítulo. O quinto capítulo expõe as conclusões, seguido do capítulo contendo as referências utilizadas durante a execução da pesquisa.

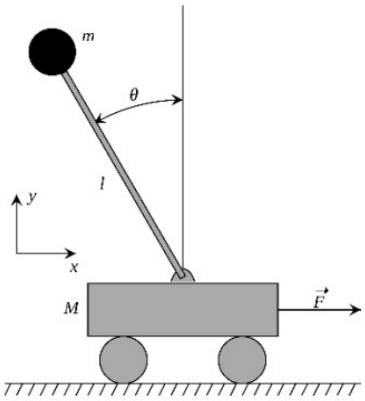
2 O ROBÔ EQUILIBRISTA

A principal característica de um robô equilibrista de duas rodas é sua condição naturalmente instável, com dinâmica altamente não-linear, que atrai a atenção de muitos pesquisadores desafiados em determinar as características físicas e modelos de controle para permitir seu equilíbrio próximo à sua posição instável vertical (MARTÍNEZ, 2015).

2.1 Modelo Dinâmico

A maioria dos estudos a respeito de sistemas equilibristas consideram sistemas lineares ou linearizados (ESMAEILI, ALFI e KHOSRAVI, 2017), como equações dinâmicas de Newton-Euler, ou Lagrange (GONZALEZ, ALVARADO e MUÑOZ DE LA PEÑA, 2017), para que as equações de controle possam ser determinadas e então inseridas em um sistema de controle que mantém uma rotina de leitura das medidas de um sensor para ativar motores e outros atuadores com o intuito de manter o sistema equilibrado através de um controle. Segundo Bhatti *et al.* (2015), esse sistema de robô equilibrista imita o comportamento de um pêndulo invertido, e de fato trabalha com o mesmo princípio como a teoria do Carro e Mastro (do inglês *cart and pole*) representado pela FIG. 1, que consiste em uma base móvel na qual é fixada um mastro com um grau de liberdade e uma massa na extremidade superior.

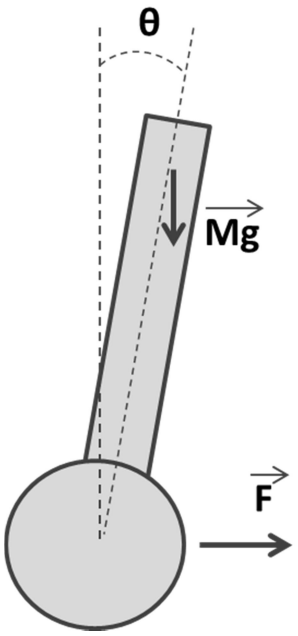
Figura 1 - Representação do modelo Carro e Mastro



Fonte: Bhatti *et al.*, 2015.

A condição naturalmente de equilíbrio estável desse sistema ocorre para a haste na horizontal com a massa próximo ao solo devido à força gravitacional. Quando a haste encontra-se na vertical, uma condição de equilíbrio instável é verificada. Para manter esse equilíbrio, com a haste na vertical, é necessário que o carro realize movimento na direção em que o mastro se inclina (BLOMSTEDT, HARALDSSON e NORDIN, 2016).

Figura 2 - Representação do modelo físico do robô equilibrista



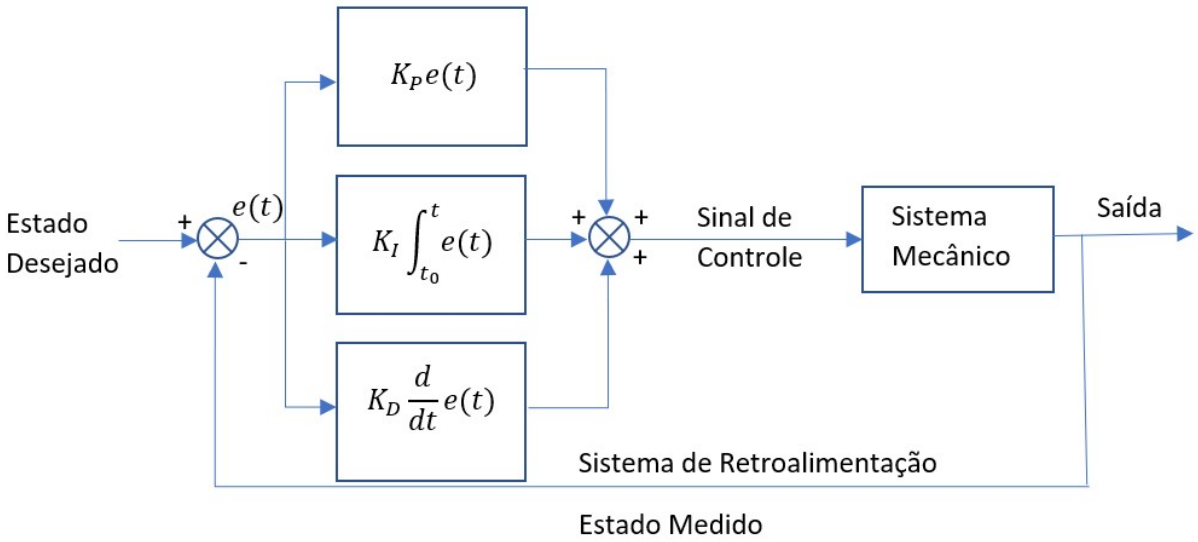
Fonte: o autor.

Um sistema de controle capaz de manter o robô equilibrado deve monitorar o ângulo de inclinação da estrutura em relação à vertical e comandar atuadores que desloquem o conjunto no sentido da inclinação, como apresentado na FIG. 2, com força controlada para que insuficiências ou excessos não coloquem o robô novamente em condição de instabilidade. Para isso, um sistema de controle de malha fechada deve ser aplicado.

Além do controle PID apresentado na FIG. 3, podem ser empregados outros métodos de controle, como o aplicado por Dan (2015) que desenvolveu um sistema de controle baseado em Controle Robusto H∞ obtendo resultados satisfatórios, ou Tomašić (2012) que implantou um controle LQR (Regulador Linear Quadrático) em seu robô equilibrista. Siqueira *et al.* (2013) considera o método não-linear no

Estados de Espaço para obter o modelo do sistema de controle de um pêndulo invertido.

Figura 3 - Sistema de controle clássico: PID



Fonte: O autor.

Blomstedt (2016) afirma que quanto mais alto é o robô, mais próximos de zero estão localizados os polos da função de transferência, e mais fácil é a estabilização do sistema. Os robôs equilibristas pertencem à classe dos sistemas mecânicos não-holonômicos (KEDZIERSKI e TCHON, 2017). Para a determinação dos parâmetros de controle por métodos clássicos, como o PID, há a necessidade de se determinar o modelo dinâmico do sistema físico, determinar a função de transferência e determinar as constantes do sistema de controle de modo a obter-se a melhor resposta dinâmica do sistema.

Considerando-se o controlador por Lógica Nebulosa, no entanto, não existe a necessidade de modelamento dinâmico do sistema, uma vez que as variáveis utilizadas remetem a conceitos incertos, aproximados, condizentes ao pensamento humano e, portanto, não houve a preocupação neste trabalho de se definirem as equações do sistema dinâmico.

2.2 Unidade de Controle

A unidade de controle é o dispositivo responsável por realizar os cálculos baseados nas medições realizadas através dos sensores e, através do sistema de

controle empregado, enviar a correta informação para controle dos atuadores a fim de estabilizar o sistema na posição vertical. Existem muitas opções de controladores para utilizar em um robô equilibrista. Chinnadurai (2015) escolheu utilizar uma plataforma de desenvolvimento da *Texas Instruments* chamada CC3200, a qual contém um micro controlador ARM de 32 bits e outros circuitos dedicados à comunicação sem fio. Eriksson (2016) usou uma ferramenta de PLC chamada CODESYS em uma placa *Raspberry Pi* para controlar seu robô. Outros autores, como Tomašić (2012) e Ghani (2011) consideram suficientes para a tarefa os micro controladores de 8 bits como a família ATmega utilizada na plataforma Arduino. Utilizou-se neste trabalho a placa microcontrolada Arduino Pro Mini, que se mostrou bastante adequada para realizar o controle proposto e utiliza o micro controlador ATmega328P.

2.2.1 O Micro controlador ATmega328P

O ATmega328P é um micro controlador da família AVR criado pela ATMEL *Corporation*, empresa fundada em 1984 e adquirida por sua antiga concorrente Microchip no início de 2016 (MICROCHIP TECHNOLOGY INC., 2016; MICROCHIP TECHNOLOGY INC., 2018), com características que suprem uma ampla gama de aplicações, com tensão lógica de funcionamento entre 1,8V e 5,5V e frequência de operação crescente de acordo com a tensão de alimentação, conforme o *datasheet* (folha de dados) do componente (MICROCHIP TECHNOLOGY INC., 2018). A tabela 1 apresenta a máxima frequência possível em cada faixa de alimentação.

Tabela 1 - Relação entre tensão de alimentação e frequência de operação do ATmega328P

Tensão de Alimentação	**Máxima Frequência de Trabalho**
1,8V – 5,5V	4MHz
2,7V – 5,5V	10MHz
4,5V – 5,5V	20MHz

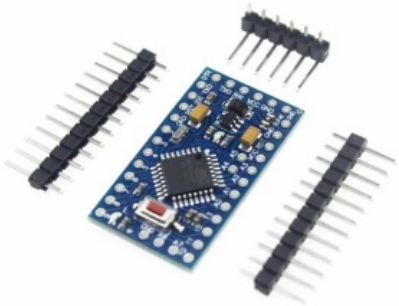
Fonte: Microchip Technology Inc., 2018.

O componente possui quatorze pinos digitais configuráveis como entradas ou saídas, sendo que destes, seis podem ser configurados como saídas PWM. Possui ainda seis entradas analógicas com 10 bits de precisão, portas de comunicação serial UART (*Universal Asynchronous Receiver & Transmitter*), I^2C (*Inter Integrated Circuit*) e SPI (*Serial Peripheral Interface*), com memória RAM de 2kB, memória flash de 32kB e EEPROM de 1kB.

2.2.2 O Arduino Pro Mini

O Arduino é uma plataforma de prototipagem rápida que se difundiu rapidamente pelo mundo devido à facilidade de uso e baixo custo. Nasceu na Faculdade de Artes de Ivrea, na Itália, baseado no trabalho de conclusão de curso do Colombiano Hernando Barragán entregue em 2004, que propôs *Wiring*: uma plataforma micro controlada aliada à ferramenta de desenvolvimento de software chamada *Processing* criada por Ben Fry e Casey Reas, ambos professores do MIT nos Estados Unidos. O objetivo era facilitar o desenvolvimento de obras de arte interativas (BARRAGÁN, 2018). Seu professor Massimo Banzi e mais quatro colegas aperfeiçoaram a ideia e lançaram o Arduino em 2005 como um hardware básico com um IDE (*Integrated Development Environment* – Ambiente de Desenvolvimento Integrado) baseado no *Processing* e que podia receber inúmeras bibliotecas para facilitar seu uso por pessoas não especializadas em programação ou em hardware (ARDUINO CC, 2019).

Figura 4 - Arduino Pro Mini

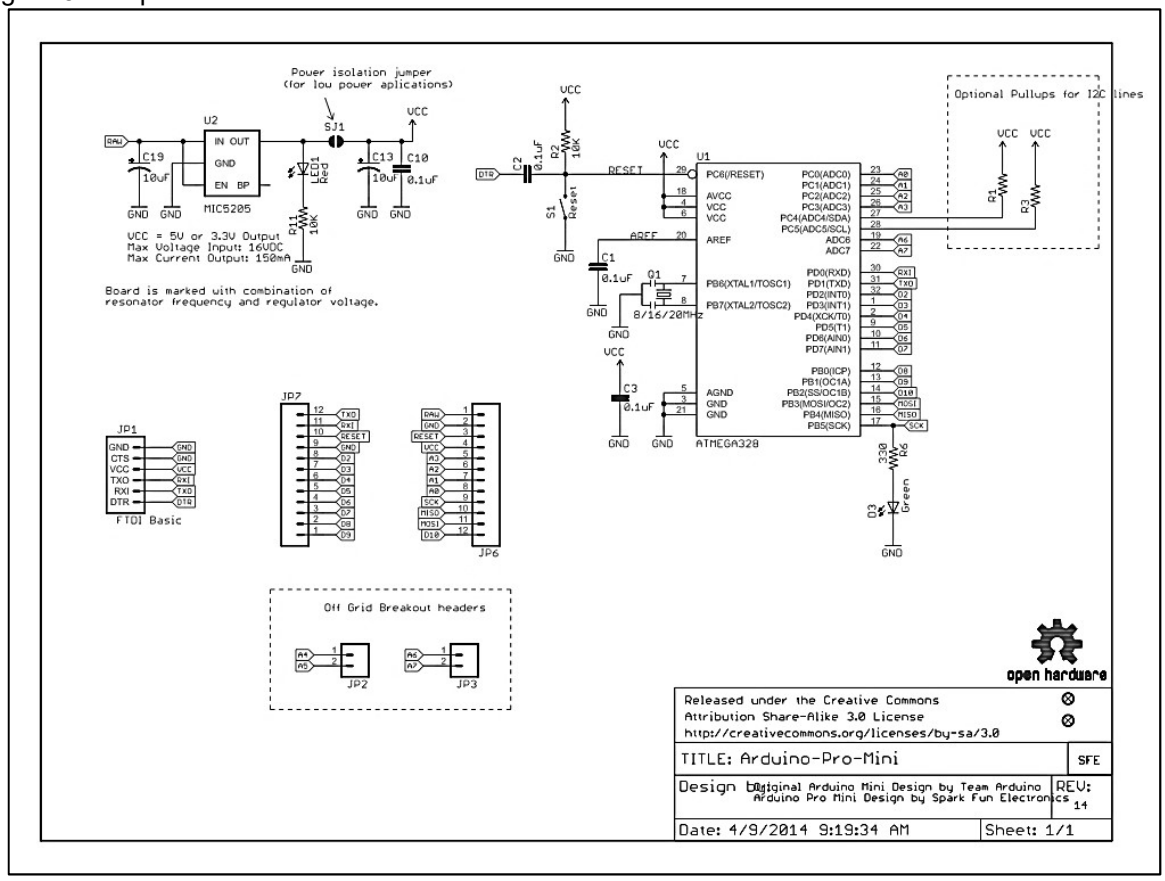


Fonte: Arduino CC, 2019.

A placa Arduino Pro Mini, apresentada na FIG. 4, é uma versão menor e simplificada da placa Arduino Uno, utilizando o mesmo tipo de micro controlador em versão SMD, mantendo-se somente os componentes essenciais ao funcionamento (SPARKFUN ELECTRONICS, 2019).

Consultando-se o esquema elétrico do Arduino Pro Mini, apresentado na FIG. 5, observa-se que essa versão foi desenvolvida pela *Spark Fun Electronics* baseada na versão Arduino Mini desenvolvida pela equipe Arduino.

Figura 5 - Esquema elétrico do Arduino Pro Mini



Fonte: *SparkFun Electronics*.

Essa placa tem um micro controlador ATMEGA328 SMD, e possui dimensões reduzidas, ideal para sistemas embarcados. Suas principais diferenças em relação ao Arduino UNO são a ausência de um conversor UART/USB, o que impede ligar a placa diretamente à porta USB do computador para sua programação. Para isso utiliza-se um conversor Serial/USB externo ajustando-se a tensão desse conversor para a tensão da placa. Outra diferença importante é que a função *auto-reset* para a

gravação dos programas só pode ser obtida se o conversor USB/serial utilizado possuir o pino DTR. No conversor utilizado na execução deste trabalho não há o pino DTR, exigindo que se pressione o botão de *reset* quando o IDE do Arduino estiver pronto para descarregar o programa na placa. Caso o *reset* não seja pressionado, a transferência do programa não ocorre e o IDE apresenta uma mensagem de erro.

Observa-se pela FIG. 5 que os pinos A4 e A5 compartilham a função de porta de comunicação I²C, e são localizados em barramento de pinos separado, JP2, e que existem dois resistores R1 e R3 opcionais que, se existirem na placa, possuem a função de elevar o potencial do barramento I²C para a tensão V_{CC} da placa. Deve-se atentar para a tensão de operação desse barramento para não ocasionar danos aos circuitos interligados devido à incompatibilidade de tensão. Deve ser estudada a melhor forma de compatibilização com eventual remoção dos resistores R1 e R3 do circuito.

Existem duas versões principais de placa Arduino Pro Mini, e a diferença está no regulador de tensão (que pode ser de 5V ou de 3,3V) e no cristal do oscilador, que pode ser 8 ou 16MHz, conforme explanado anteriormente através da tabela 1. Geralmente a placa possui uma marcação indicando as características da mesma, mas a placa utilizada neste trabalho não possui indicação alguma.

Para descobrir as características da placa utilizada, foram aplicados 9V ao pino *RAW* (pino 1) e então realizada a medição da tensão no pino VCC (pino 4), obtendo-se a tensão de 3,3V. Para verificar a frequência de operação, foi utilizada uma lente de aumento para ler a inscrição sobre o ressonador (peça cromada), 80f, indicando 8MHz. Para confirmar, foi gravada no Arduino Pro Mini uma rotina chamada "Blink" para piscar o LED do pino 13 a cada 10 segundos, e então verificar se o *clock* estava correto, verificando-se que a frequência resultante na saída foi a metade do esperado. Com isso concluiu-se que a velocidade de processamento da placa disponível era a metade do Arduino Uno convencional, e que essa característica poderia comprometer o tempo de resposta necessário ao controle do robô em desenvolvimento. Decidiu-se continuar os testes com o Arduino Mini e posteriormente desenvolver-se uma placa eletrônica específica utilizando o micro

controlador ATMEGA328P com as características necessárias ao bom desempenho do sistema controlador.

2.3 Métodos de Controle

Similarmente às unidades de controle discutidas anteriormente, existem também diversos métodos de controle que podem ser empregados para se atingir a estabilidade de um robô equilibrista. Este trabalho adota o Controlador *Fuzzy* como principal objetivo, e esse assunto é abordado adiante com maior profundidade. O trabalho de Fang (*A Posture Control System Design for a Two-wheeled and Self-balancing Robot,* 2015) publicado em 2015 aborda esse assunto, e outro trabalho publicado pelo mesmo autor no ano anterior (*The Research on the Application of Fuzzy Immune PD Algoorithm in the Two-Wheeled and Self-balancing Robot System, 2014)* mostra que ele também utilizou outros métodos de controle como o LQR, obtendo resultados mais rapidamente e com menor erro que os métodos experimentados anteriormente. Um Regulador Linear Quadrático é um método para definição a posição ideal dos polos de uma função de transferência de modo que o sistema se torne estável e rápido suficiente para reagir a perturbações, baseado na teoria do controle ótimo de um sistema dinâmico (NGUYEN e PHUONG, 2016). Tomašić (2012) realizou simulações com controladores PID, *Fuzzy* e LQR, e embora tenha obtido melhores resultados simulados com a aplicação da Lógica *Fuzzy*, decidiu por utilizar LQR devido à dificuldade de aplicar a lógica *Fuzzy* em seu sistema físico com as ferramentas disponíveis para a unidade de controle adotada. Outra técnica para definir os parâmetros corretos de um controle PID é apresentado por Ram (2017), que aplica o algoritmo PSO (Optimização por Enxame de Partículas, do inglês *Particle Swarm Optimization*), ramo da inteligência artificial que otimiza um problema por iterações a fimd e melhorar a solução candidata medindo-se a qualidade. Velazquez (2016) aplicou a técnica de Controle PID em cascata e obteve resultados com variação de inclinação menor que um grau quando não existem perturbações aplicadas.

2.4 Atuadores

Motores CC (corrente contínua) possibilitam movimentos de rotação e permitem ao robô se mover para buscar sua estabilidade, e são usados como o subsistema atuador do robô (FRANKOVSKY, DOMINIK, *et al.*, 2017). A entrada desse subsistema é a tensão elétrica enquanto a saída pode ser representada por velocidade angular e torque. Motores CC convertem uma tensão elétrica em movimento rotativo através de forças eletromagnéticas geradas no rotor, em oposição ao campo permanente existente no estator.

Figura 6 - Motor CC com caixa de redução mecânica

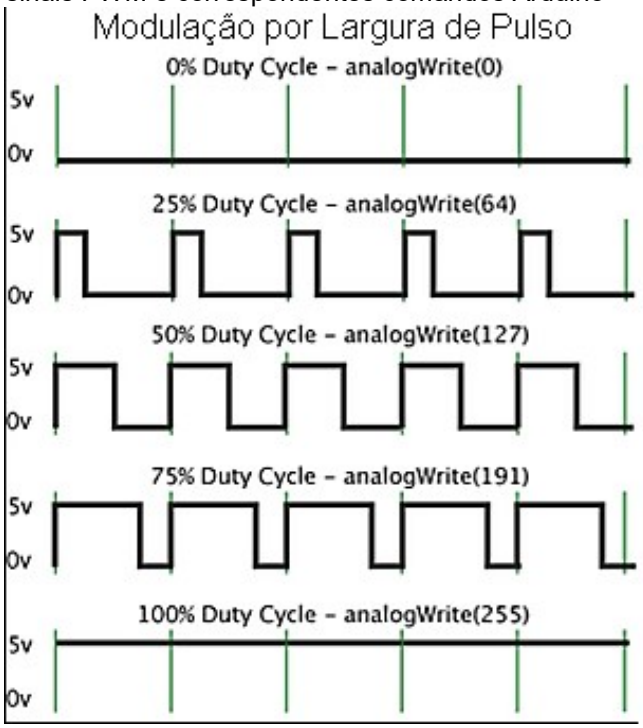
Fonte: www.filipeflop.com, acessado em 10/03/2019.

Caixas de redução mecânica podem ser utilizadas para multiplicar o torque do motor, disponibilizando-o no eixo de saída, reduzindo-se, por consequência, a velocidade de rotação desse eixo. Caixas de redução podem conter conjuntos de engrenagens metálicas ou plásticas, dependendo dos torques envolvidos. A FIG. 6 apresenta um conjunto motor CC e caixa de redução plástica. São considerados componentes de baixo custo em comparação a outras opções.

O controle da velocidade dos motores CC deve ser realizado pela variação de tensão nos motores, ou o controle PWM (Modulação por Largura de Pulso, do inglês *Pulse Width Modulation*), após os sinais PWM serem amplificados por um circuito dedicado (GONZALEZ, ALVARADO e MUÑOZ DE LA PEÑA, 2017). A modulação por largura de pulso consiste em enviar ondas quadradas ao motor e variar o *duty-cycle* da onda (razão entre o tempo em nível lógico alto pelo período da onda),

resultando em um valor médio de tensão variável no motor, de forma que 0% represente o motor parado e 100% represente a velocidade máxima do dispositivo eletromecânico, como pode ser observado na FIG. 7. O sentido de rotação pode ser alterado invertendo-se o sentido da corrente pelo motor, o que pode ser conseguido pelo uso de circuitos chamados Pontes H.

Figura 7 - Exemplos de sinais PWM e correspondentes comandos Arduino



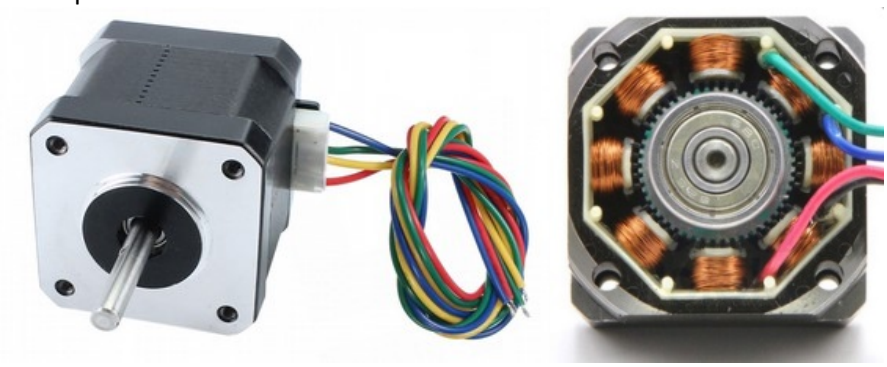
Fonte: Adaptado de http://wiki.sainsmart.com/index.php/Chapter_4_PWM, acesso em 12/03/2019.

Servo motores também podem ser controlados por PWM, e possuem como principal vantagem o seu alto torque, mas possuem limite de rotação física, geralmente, menor que de 180º em servomotores de baixo custo (LANGLEY, 2016).

Motores de passo, representados pela FIG. 8, não possuem restrição quanto ao ângulo de rotação, e podem fornecer alto torque mesmo em baixas velocidades, mas em contrapartida possuem custos elevados (na ordem de seis vezes maiores que motores CC) e requerem controladores mais complexos.

São motores síncronos que possuem de 4 a 6 terminais ligados a conjunto de bobinas dispostas no estator que, de acordo com a sequência de alimentação, geram forças eletromagnéticas que, combinadas, permitem a rotação do eixo de forma controlada em determinado sentido, em passos definidos tanto por suas características construtivas quanto pelo modo de acionamento.

Figura 8 - Motor de passo



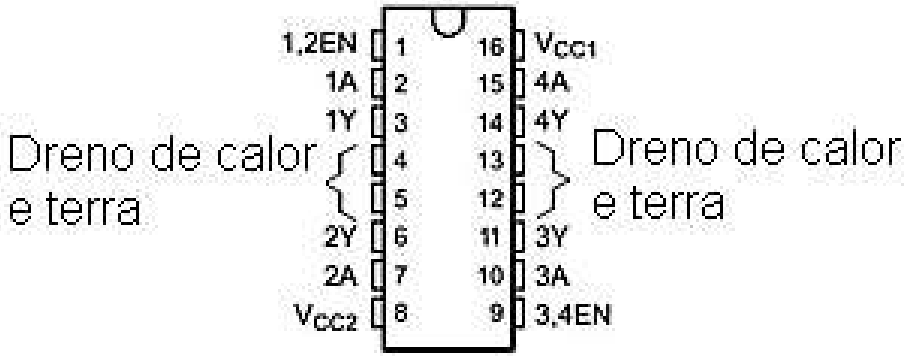
Fonte: Composição do autor, a partir de, à esquerda, https://www.usinainfo.com.br/motor-de-passo/motor-de-passo-nema-17-12v-45kgfcm-42hbd40bj4-tf0-cabo-3038.html, acesso em 15/03/2019, e a direita https://www.pololu.com/product/2267, acesso 15/03/2019.

2.5 *Drivers* dos Motores

Considerando o uso de motores de corrente contínua, Phan (2017) sugere o uso de um componente L298P para amplificar a corrente dos sinais de PWM recebidos do micro controlador antes de aplicar aos motores. A estrutura do componente deve possuir o conceito de ponte-H para permitir que os motores girem em ambos os sentidos (CHUN-HONG e BIN, 2018).

Neste trabalho utilizou-se o *driver* L293D, que permite ligar dois motores em ponte-H. O componente L293 é um circuito integrado que disponibiliza quatro meias-pontes, ideal para controle de cargas indutivas de até 600mA no caso do L293D, ou até 1A (1,2A de pico) no L293 sem sufixo (TEXAS INSTRUMENTS, 2016). Sua pinagem é exibida na FIG. 9, onde se observa que as entradas são representadas pela letra A e as saídas são representadas pela letra Y, e o prefixo numérico indica a porta relacionada. Possuem um pino de habilitação (*Enable*) para cada par de "meia-ponte", e podem trabalhar chaveando pulsos de até 5kHz, permitindo o controle de velocidade através de PWM.

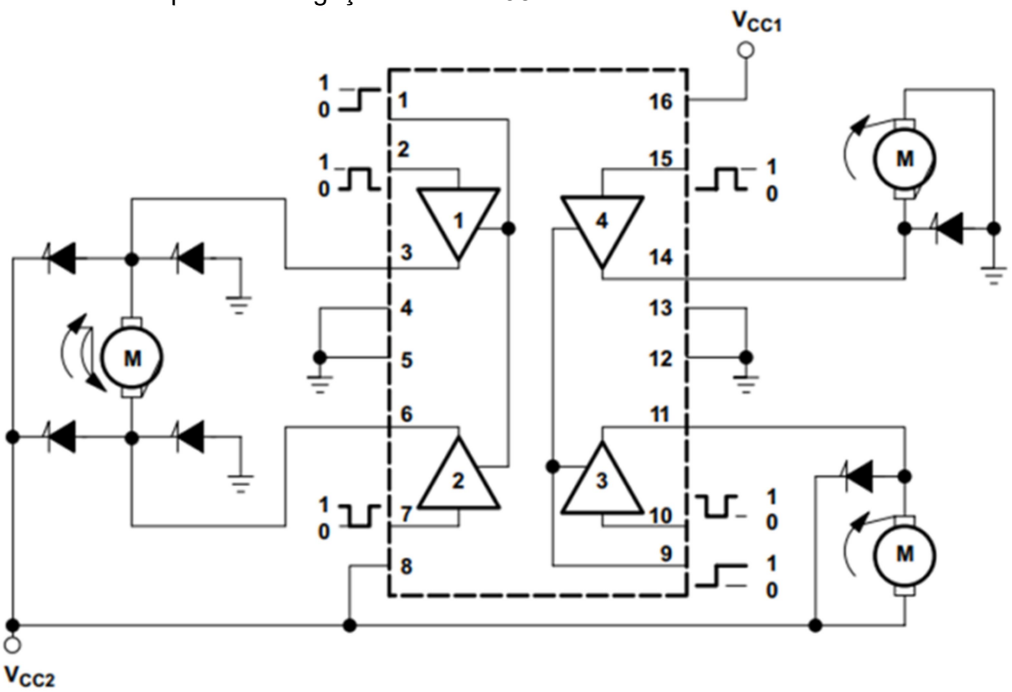
Figura 9 - Fragmento da folha de dados do componente L293D mostrando a lógica de controle



Fonte: Adaptado de *Texas Instruments*.

A FIG. 10 apresenta os possíveis modos de operação do componente para controle de motores CC. A versão L293D possui diodos de proteção internos contra tensão reversa e contra descargas provenientes de estática, simplificando o circuito.

Figura 10 - Possíveis Esquemas de ligação com o L293D



Fonte: Texas Instruments.

O circuito integrado possui dois pinos distintos para alimentação, sendo um para alimentação lógica e outro para alimentação dos motores. A alimentação lógica VCC1 deve compreender a faixa de 2,3V a 7V, recomendando-se a aplicação da tensão padrão TTL: 5V. A tensão de alimentação da carga pode ser entre 4,5V e 36V. Os pinos de aterramento (GND) possuem a função secundária de dissipar o

calor do componente e, por isso, devem todos ser ligados à placa de circuito impresso para ajudar a refrigerá-lo. Caso a temperatura do componente ultrapasse o valor limite, as saídas são colocadas em estado de alta impedância independentemente do estado lógico das entradas até que a temperatura volte a patamares seguros.

Tabela 2 - Tabela verdade do L293

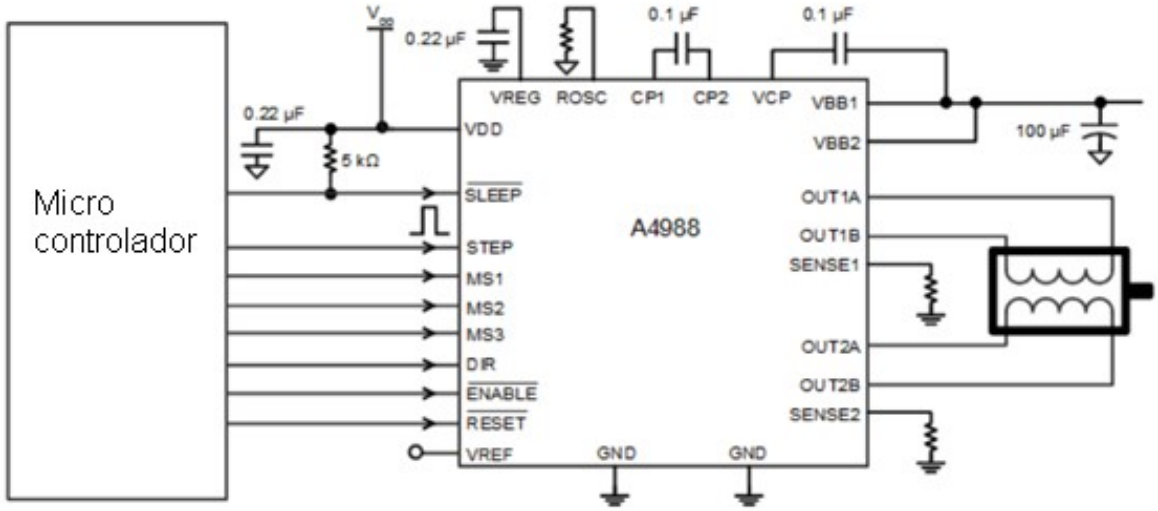
Entradas		**Saídas (Y)**
A	**EN**	
Nível Lógico Alto	Nível Lógico Alto	VCC2
Nível Lógico Baixo	Nível Lógico Alto	GND
Indiferente	Nível Lógico Baixo	Alta Impedância

Fonte: O autor, adaptado de Texas Instruments.

Seu funcionamento é representado pela tabela 2: quando o *Enable* se encontra em nível lógico baixo, as saídas Y correspondentes são colocadas em alta impedância (ou seja, se comportam como se estivessem desconectadas do circuito). Quando *Enable* está em nível lógico alto, as saídas Y assumem o estado lógico das entradas A correspondentes, fornecendo a tensão VCC2 para a carga em caso de nível lógico alto da entrada, ou GND em caso de nível lógico baixo.

Considerando-se motores de passo, A4988 é um *driver* tradutor da fabricante Allegro Microsystems que controla motores de passo de forma simplificada, permitindo configurações *full-step*, ½ *step*, ¼ *step*, 1/8 *step* ou 1/16 *step*, com tensões de operação de até 35V e correntes de até 2A (ALLEGRO MICROSYSTEMS INC., 2012). O módulo disponível no mercado com o circuito integrado A4988 possui regulador de corrente ajustável para limitar a corrente no motor, sendo necessário um módulo para cada motor de passo. A FIG. 11 exibe o circuito recomendado para seu funcionamento. As entradas MS1, MS2 e MS3 controlam o modo de passo (inteiro, meio, etc), DIR controla o sentido da rotação do motor a cada pulso enviado à entrada STEP. A entrada SLEEP coloca o componente em estado de baixo consumo quando recebe nível lógico baixo, e o terminal ENABLE habilita o funcionamento do motor quando recebe nível lógico baixo, e RESET reinicia o componente.

Figura 11 - Diagrama típico de aplicação do componente A4988



Fonte: Adaptado de Allegro Microsystems.

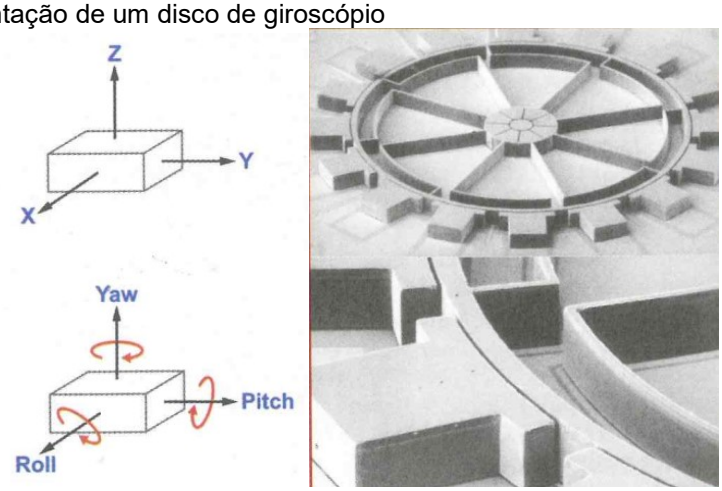
2.6 Sensoriamento

Uma Unidade de Medição Inercial (*IMU - Inertial Measurement Unit*) é um componente baseado em mais de um tipo de sensor capaz de medir variações de posição, que podem conter um acelerômetro e um giroscópio e, às vezes, um magnetômetro (conhecido como bússola eletrônica), e que pode, por exemplo, ser utilizado para medir o ângulo de inclinação do robô para então permitir ao controlador definir as ações a serem tomadas pelos atuadores a fim de manter-se a estabilidade do conjunto.

> A combinação de um acelerômetro linear de três eixos e um giroscópio de três eixos (três graus de liberdade lineares e três graus de liberdade angulares) forma um sensor de movimento de 6-eixos, ideal para a medição precisa de todos os movimentos de um sistema (TAUSCHECK, 2007).

Um giroscópio é um dispositivo que mede a variação do ângulo, traduzido em quão rápido o objeto está girando em graus por segundo Torige (2013). Um giroscópio é capaz de medir a velocidade angular aplicando uma vibração entre 10kHz e 20kHz em um anel dentro do MEMS (sigla em inglês para Sistema Micro Eletro-Mecânico) medindo-se alterações no eixo de vibração causadas pelo efeito Coriolis quando o sensor é rotacionado (TAUSCHECK, 2007) conforme representado pela FIG. 12.

Figura 12 - Representação de um disco de giroscópio



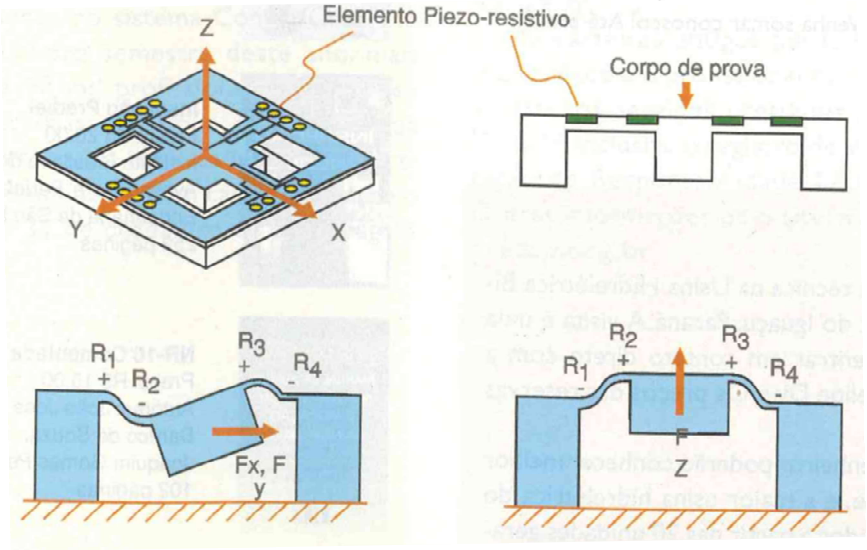
Fonte: Tauscheck, 2017.

Devido às suas características construtivas, as medidas de um giroscópio são consideradas mais confiáveis do que as medidas de acelerômetros, contudo o giroscópio acumula um erro incremental em suas medidas conforme o tempo passa (HAN, HAN e JO, 2014) devido à influência da rotação terrestre, que deve ser considerada no tratamento da informação coletada.

O acelerômetro mede acelerações inerciais através do conceito de uma massa de prova conhecida na qual forças externas provocam a deformação de componentes internos aos quais a massa está conectada, cujos efeitos podem ser medidos por diferentes técnicas. Com o acelerômetro é possível medir a aceleração da gravidade e, medindo-se a composição nos três eixos da força da gravidade é possível calcular o ângulo de inclinação do sistema (TORIGE, 2013). Contudo, um acelerômetro é sensível a perturbações mecânicas, e pode resultar em medições imprecisas em ambientes ruidosos ou com vibração.

Segundo Cunha (2007), os seguintes métodos podem ser utilizados para medir a aceleração dentro de sistemas MEMS: capacitivo, resistivo, de fibra ótica, piezoelétrico e piezo-resistivo, este último representado pela FIG. 13.

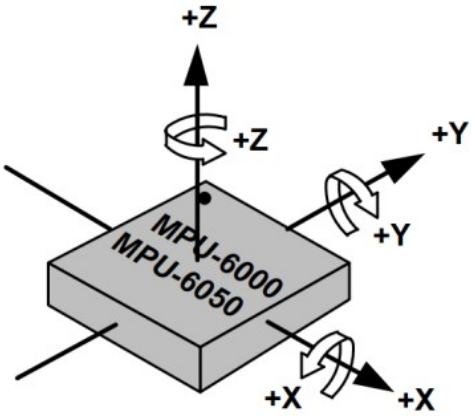
Figura 13 - Representação de um acelerômetro piezo-resistivo



Fonte: (CUNHA, 2007).

Hoje existem muitas opções para aquisição de IMU's no mercado, mas há quinze anos no Brasil a realidade era bem diferente, como afirmado por Oliveira *et al.* (2008) que precisou utilizar uma placa de uma antiga câmera de vídeo para desenvolver um dispositivo de estabilização de uma plataforma horizontal. De acordo com a folha de dados do fabricante InvenSense (2012), o IMU MPU-6050 possui um giroscópio de três eixos e um acelerômetro de três eixos, e a comunicação com o micro controlador pode ser realizada por protocolo I²C (*Inter-Integrated Circuit*) de forma a obter os dados já convertidos na representação digital. A FIG. 14 apresenta os eixos de medição do acelerômetro e giroscópio do componente MPU-6050.

Figura 14 - Representação dos eixos de acelerômetro e giroscópio do MPU-6050

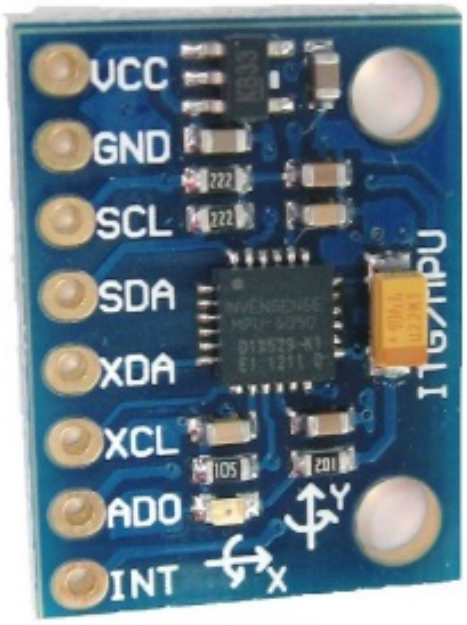


Fonte: (INVENSENSE INC;, 2012)

Os dados de ambos os sensores podem ser combinados para obter resultados de posição mais confiáveis. Combinando-se a informação de um giroscópio e um acelerômetro é possível alimentar o sistema de controle para obter-se uma boa estabilização. Um giroscópio com um sensor de posição pode prover informação ainda mais valiosa ao usuário do que um giroscópio sozinho (LAUBLI, GARABEDIAN, *et al.*, 2015).

O módulo GY-251 é uma placa de dimensões reduzidas que contém um MPU-6050 e seus periféricos essenciais, conforme pode ser observado na FIG. 15, e seu esquema elétrico é apresentado na FIG. 16.

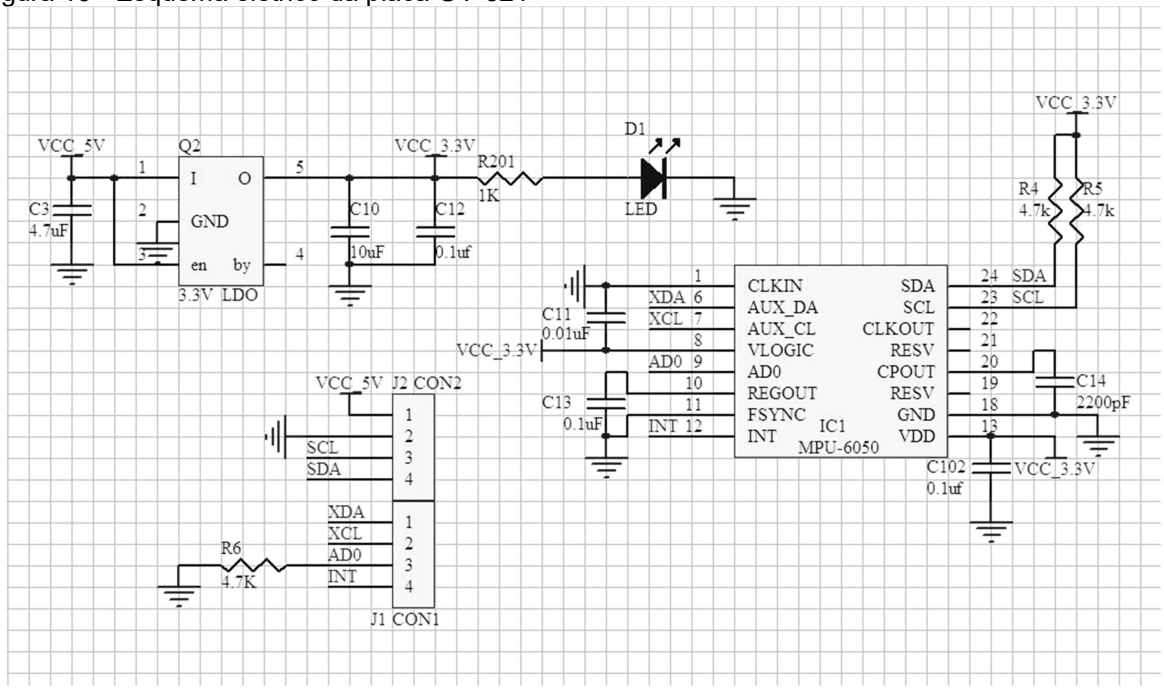
Figura 15 - Placa GY-521



Fonte: O autor.

Através do esquema elétrico do módulo GY-521, apresentado na FIG. 16, pode-se observar que existem resistores (R4 e R5) no barramento de comunicação serial I²C, e que tais resistores tornam possível a comunicação com o micro controlador sem a necessidade de inclusão de resistores de *Pull-Up* (elevador de tensão) nos demais dispositivos e que, ao contrário, a presença de mais resistores poderia comprometer a correta comunicação entre os dispositivos.

Figura 16 - Esquema elétrico da placa GY-521



Fonte: http://dami.azw.pt/gy-521-mpu6050-acelerometro-e-giroscopio/, acessado em 20/05/2019.

2.7 Filtros de Sinais

Leituras de giroscópios e acelerômetros são, por sua natureza, contaminadas por ruído e pelos próprios erros que se acumulam com o passar do tempo (TORIGE, 2013). Para eliminar reações indesejadas devido a ruído são aplicados algoritmos de fusão sensorial ou filtros digitais nos dados lidos. Os filtros mais comuns utilizados em robôs equilibristas são o Filtro de Kalman e o Filtro Complementar. O Filtro Complementar não atua nos sinais, mas no ruído, e estima o ângulo de múltiplas fontes que podem conter dados incorretos (KRISHNA e RAO, 2016). O filtro complementar é um algoritmo que combina dois filtros: um filtro passa-altas aplicado ao giroscópio e um filtro passa-baixas aplicado ao acelerômetro, cada qual multiplicado a um índice matematicamente complementar, de forma que o resultado da soma desses índices seja unitário. Essa técnica entrega bons resultados práticos e exige menor poder computacional para ser executado, porém pode apresentar um erro crescente com o passar do tempo (BUENO e ROMANO, 2014).

Lekshmy (2015) afirma que o filtro de Kalman é utilizado para fundir os dados de dois sensores, e consiste em um conjunto de equações matemáticas que provêm

meios computacionais recursivos para estimar o estado de um processo de uma forma que minimiza a média do erro quadrático. Afirma ainda que esse filtro é muito poderoso uma vez que pode estimar estados do passado, presente e até futuro, mesmo que a natureza do sistema modelado seja desconhecida.

Desenvolvido por Rudolf E. Kalman em 1960 e também conhecido como estimativa linear quadrática (LQE – *linear quadratic estimation*) é um algoritmo que usa uma série de medidas observadas em um período de tempo que inclui ruído não previsível e produz estimativas de variáveis desconhecidas que tendem ser mais acuradas que aquelas baseadas somente em medições (LEKSHMY, GEORGE e ATHIRA, 2015).

Yao (2015) apresenta um teste com um potenciômetro e um filtro de Kalman, enquanto Prasetio (2018) apresenta um sistema de controle que inclui sensores de proximidade em que o sistema prevê perturbações antes que as mesmas aconteçam. Por esses motivos, foi considerado empregar o filtro de Kalman durante o desenvolvimento deste trabalho.

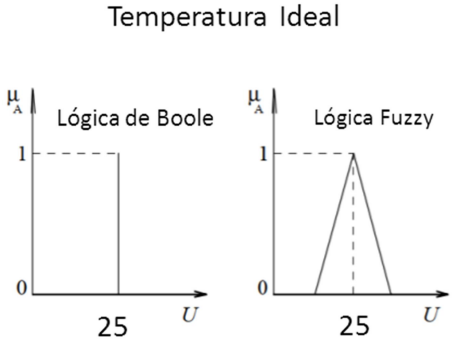
3 A TÉCNICA DE CONTROLE *FUZZY*

3.1 Introdução à Lógica Nebulosa

Lógica Nebulosa, ou Lógica Difusa, do inglês *Fuzzy Logic* é um método de tomada de decisões baseado na incerteza. É utilizada para solucionar problemas em que a imprecisão e a incerteza são variáveis complexas que dificultam o modelamento matemático desses problemas e a implementação de controladores convencionais, baseando-se no conhecimento de especialistas para definir os modos de controle (PRADO e MASSELLI, 2017).

Muitos autores consideram Lotfi Aliasker-Zadeh como o criador da Lógica Nebulosa, mas outros autores afirmam que Zadeh formulou em 1965 a teoria da lógica nebulosa baseado no trabalho de Jan Lukasiewicz que, em 1920, apresentou conjuntos com grau de pertinência 0, ½ e 1, e mais tarde passou a considerar o intervalo de infinitos valores entre 0 e 1. Zadeh percebeu que muitos problemas do cotidiano não eram bem representados por conjuntos Booleanos que só permitem 0 ou 1, verdadeiro ou falso (RIGNEL, CHENCI e LUCAS, 2011) afirmando que o mundo é nebuloso.

Figura 17 - Comparação entre Lógica Booleana e *Fuzzy Logic*



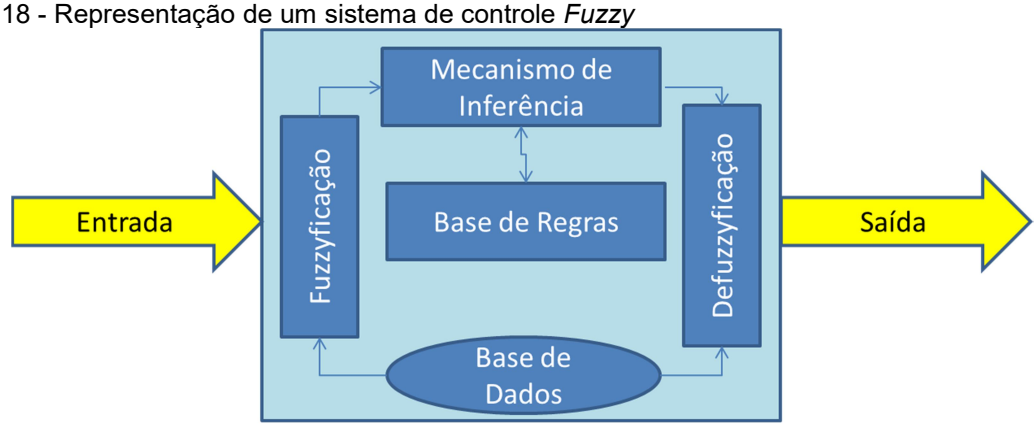
Fonte: O autor, adaptado de (GOMIDE e GUDWIN, 1994).

Um exemplo comumente apresentado é acerca de temperatura ideal em laboratórios, considerado 25°C, e pode ser observado na FIG. 17. Pela Lógica de Boole, só é admitido situação verdadeira quando a temperatura é exatamente 25°C, e qualquer valor diferente disso representa a situação Falsa. Já a Lógica Nebulosa considera que existe uma margem de aceitação com diferentes graus de pertinência

ao redor do valor 25°C, em que o valor central representa 100% de certeza, e conforme se afasta desse valor, o grau de pertinência diminui.

Em 1974, Ebrahim Mamdani foi o primeiro a aplicar os conceitos de Lógica Nebulosa em um sistema de controle real, abrindo campo para tratar com problemas complexos nos quais o modelo matemático é desconhecido ou difícil de ser obtido ou também em sistemas não-lineares (MARTÍNEZ, 2015). A Lógica Nebulosa utiliza conjuntos em um domínio que podem significar uma incerteza em resposta a uma afirmação. A Lógica Nebulosa utiliza variáveis linguísticas para definir possibilidades e graus de pertinência dessa variável no conjunto de elementos. Os elementos podem ser combinados de acordo com regras "se, então" formando sua base de regras que, através do mecanismo de inferência, resultam em saídas linguísticas que são transformadas em resultados numéricos durante o processo de "defuzzyficação", como é representado pela FIG. 18.

Figura 18 - Representação de um sistema de controle *Fuzzy*



Fonte: O autor.

Um sistema de Controle *Fuzzy* pode ser dividido em 4 etapas básicas:

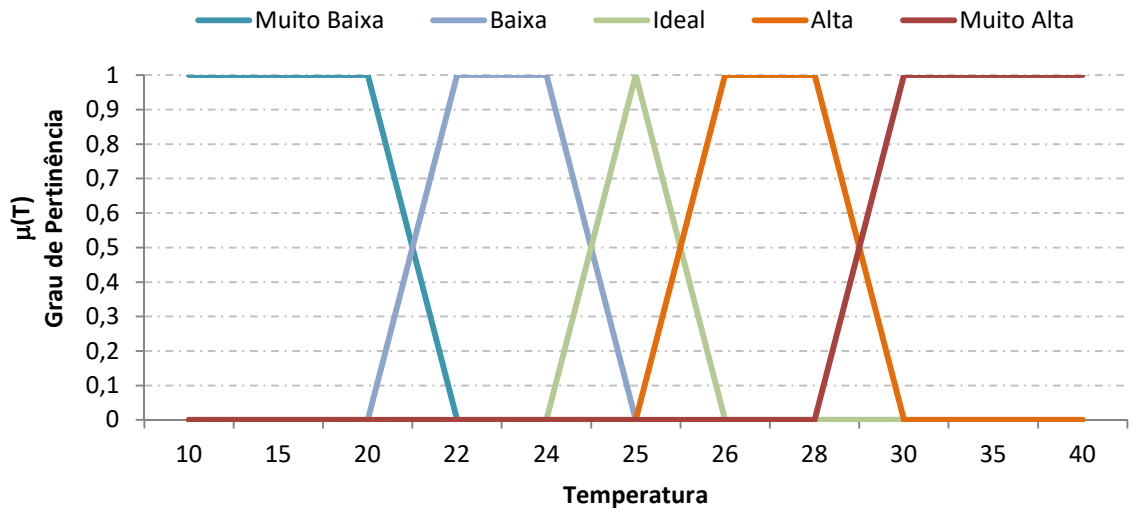
- "Fuzzyficação" (variáveis e termos linguísticos, funções de pertinência);
- Regras *Fuzzy* construídas através da base de conhecimento;
- Motor ou Mecanismo de Inferência;
- "Defuzzyficação".

Para desenvolver um sistema de controle *Fuzzy* é necessário definir-se as entradas e saídas, suas correspondentes variáveis linguísticas e seus termos. Definir as funções de pertinência e definir um conjunto de regras que serão responsáveis por calcular a saída do controlador baseado no mecanismo de inferências. A "*defuzzyficação"* é o processo pelo qual as saídas são calculadas e apresentam os valores numéricos para uso no sistema atuador.

Aprofundando um pouco no assunto, após definirem-se as entradas, é necessário determinar se as variáveis e valores linguísticos de cada uma. As variáveis linguísticas representam características conhecidas, às quais se definem termos ou valores linguísticos que referem-se a estados conhecidos designados por nomenclaturas de fácil compreensão humana e denotam valores inexatos. No caso da temperatura de uma sala, poder-se-ia definir a variável linguística Temperatura e seus possíveis valores linguísticos como Muito Baixa, Baixa, Ideal, Alta e Muito Alta.

As Funções de Pertinência relacionam as variáveis linguísticas a valores numéricos no domínio considerado em diferentes intensidades chamadas de grau de pertinência, através de funções matemáticas que podem ser representadas graficamente, como observado na FIG. 19. As funções de pertinência podem ser triangulares, trapezoidais, sigmoidais, gaussianas ou ainda *singleton*.

Figura 19 - Exemplo de variável linguística, seus termos e funções de pertinência



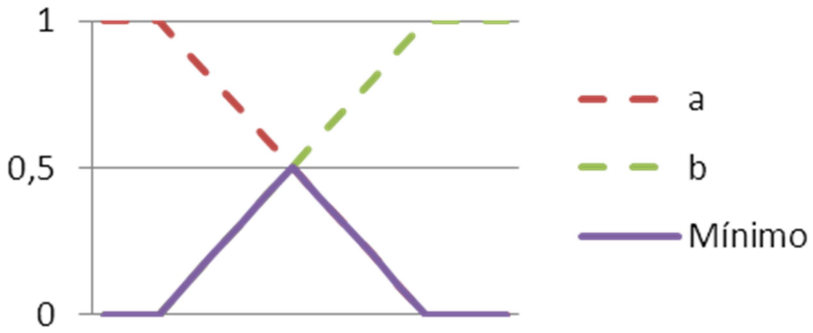
Fonte: O autor.

As Regras Linguísticas *Fuzzy* compreendem um conjunto de regras Se – Então relacionando uma condição e uma conclusão, e são determinadas por um especialista no processo que se deseja controlar (PRADO e MASSELLI, 2017). São utilizados operadores em *Fuzzy* de acordo com a teoria de conjuntos, como a seguir, muito bem explicados por Martínez (2015):

1. Intersecção: A **e** B, também representado por $A \cap B$ representa a parte comum ao conjunto A e B ao mesmo tempo. Os métodos para "fuzzyficação" são:

   a. Implicação Mínima, em que se seleciona o menor dos valores das entradas (fig. 20);

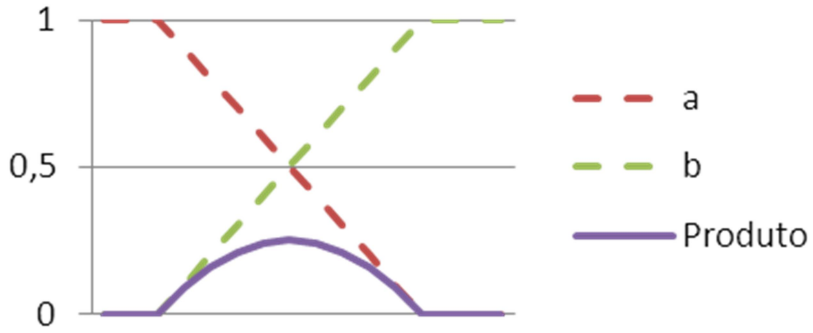
Figura 20 - Método da Implicação Mínima



Fonte: O autor.

   b. Implicação de Produto, aonde se multiplicam os valores das entradas, que é considerado mais fácil na prática (fig. 21);

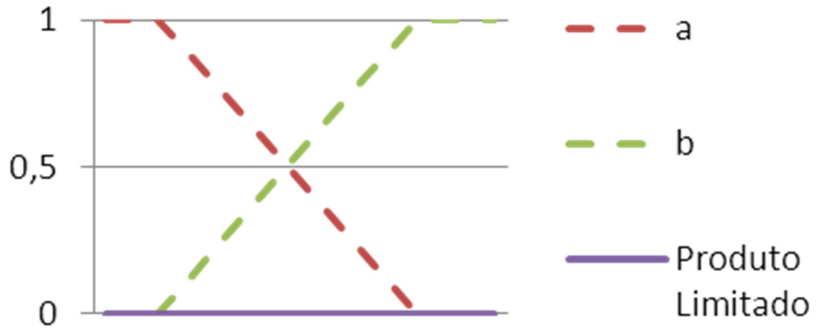
Figura 21 - Método do Produto Algébrico



Fonte: O autor.

   c. E ainda o método do Produto Limitado que considera o máximo entre 0 e a operação a + b – 1, como pode ser observado na FIG. 22.

Figura 22 - Método do Produto Limitado

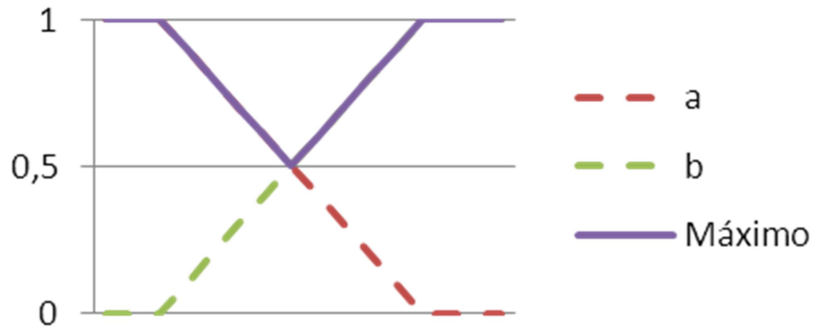


Fonte: O autor.

2. União: A **ou** B, também representado por A∪B que representa a totalidade dos conjuntos A + B, em que se podem aplicar três métodos para “fuzzyficação”:

a. Implicação Máxima, em que se adota o máximo valor das entradas, conforme FIG. 23;

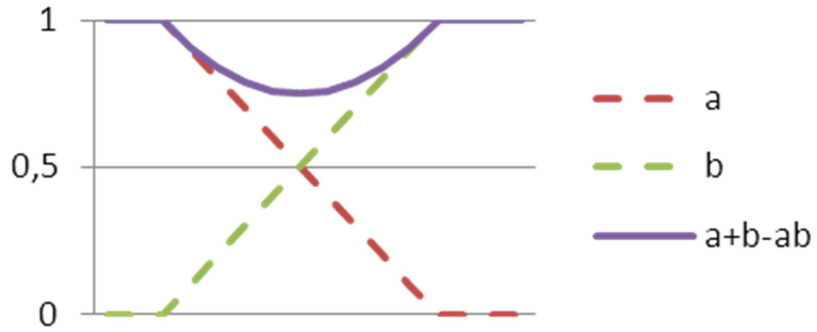
Figura 23 - Método do Máximo valor



Fonte: O autor.

b. Implicação da Soma Menos o Produto, ou Soma Algébrica, dado por A+B – A.B, conforme FIG. 24;

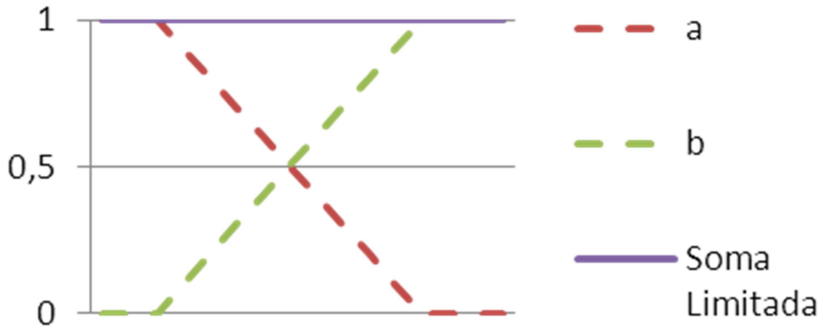
Figura 24 - Método da Soma Algébrica, ou Soma menos Produto



Fonte: O autor.

c. E ainda a Soma Limitada, dado pelo mínimo entre 1 e a soma da pertinência de a e b, ou min(1, $\mu_A(x)$ + $\mu_A(x)$) conforme pode ser observado na FIG. 25.

Figura 25 - Método da Soma Limitada



Fonte: O autor.

3. Negação: NOT A, ou !A, onde A é a exceção.

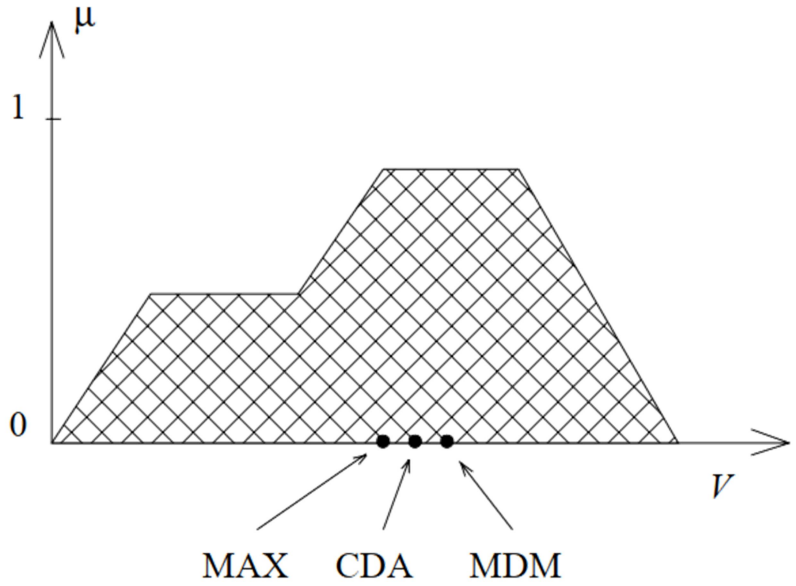
A Inferência *Fuzzy* é o processo em que os resultados *Fuzzy* são calculados, através de operadores como o Complemento *Fuzzy*, a Intersecção *Fuzzy* (similar ao E Booleano) e a União *Fuzzy* (similar ao OU Booleano), como apresentado anteriormente. Esse processo verifica todas as regras em que se satisfaçam as condições, obtendo-se as saídas *Fuzzy* correspondentes a cada regra com o respectivo grau de pertinência, que são combinadas através dos operadores Fuzzy AND e OR aplicados por diferentes métodos possíveis (MARTÍNEZ, 2015), como:

- Método de Inferência max-min (Mamdani), que utiliza o mínimo para AND e máximo para OR;
- Método de Inferência max-produto, que também utiliza o máximo, com a diferença de utilizar o produto em lugar do mínimo no cálculo das conclusões “ENTÃO”;
- Método de Inferência soma-produto, que utiliza o produto para AND e soma para OR.

A “defuzzyficação” é o processo de transformação dos valores *Fuzzy* encontrados em valores de saída numéricos, e possui dois principais métodos distintos para a “defuzzyficação”: Mamdani e Takagi-Sugeno (TS). Mamdani defende que os resultados das inferências *Fuzzy* devem ser constantes, enquanto Takagi-

Sugeno defende que a saída deve ser uma função da entrada (MACHADO, 2003). A “defuzzyficação” por Mamdani pode utilizar diferentes formas de cálculo, como o Método do Centro de Área (ou Centro de Gravidade) da figura geométrica resultante da função inferida (CDA), o Critério do Máximo (MAX) que escolhe o ponto onde a função inferida tem o seu máximo, ou a Média dos Máximos (MDM) que representa o valor médio dentre todos os pontos de máximo quando existe mais de um máximo (GOMIDE e GUDWIN, 1994). A FIG. 26 apresenta um exemplo de uma figura resultante da combinação das regras aplicadas pelo método de inferência e os possíveis resultados “defuzzyficados” pelos três diferentes métodos.

Figura 26 - Métodos de “Defuzzyficação”

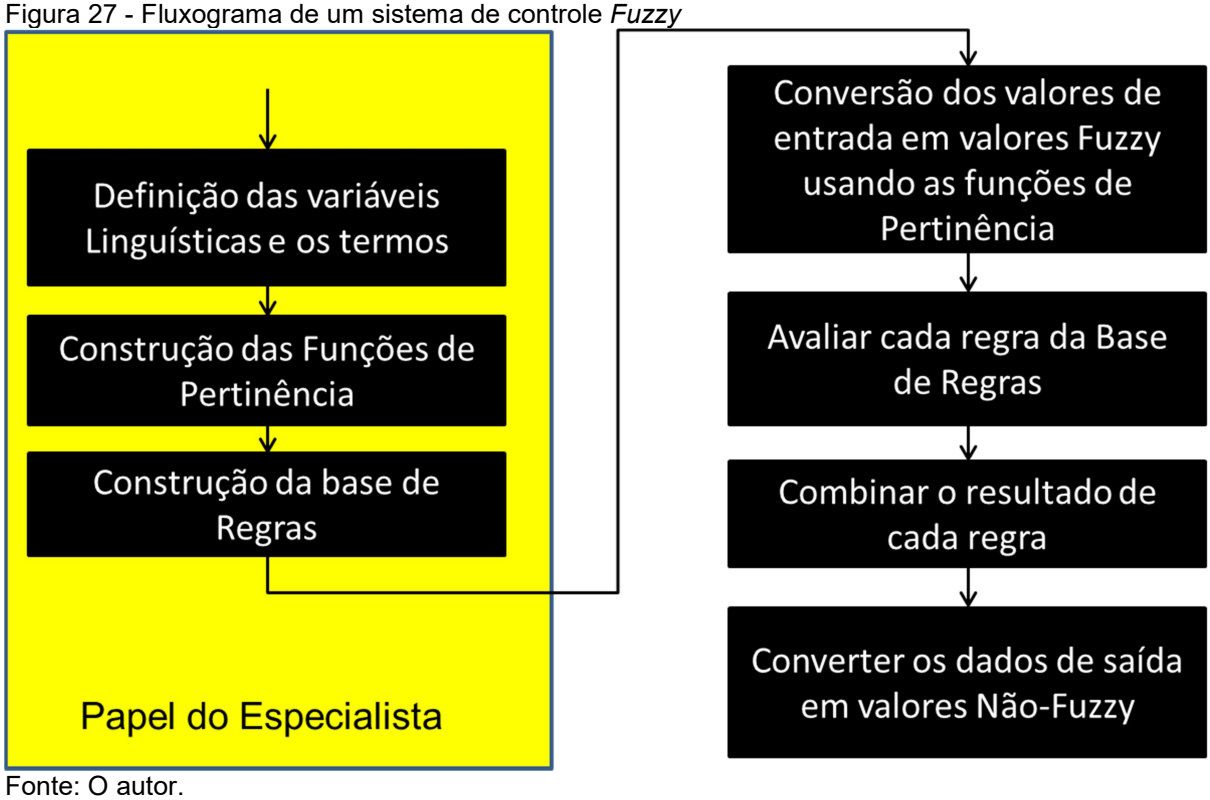


Fonte: (GOMIDE e GUDWIN, 1994).

O diferencial deste trabalho é a aplicação da Lógica *Fuzzy* para a solução do problema: manter o robô em equilíbrio. Em um sistema equilibrista clássico, a solução seria realizada pela aplicação de um controle realimentado PID (Proporcional, Integral, Derivativo) para o qual se necessita estabelecer uma função de transferência em malha fechada. Para isso, deve-se conhecer o modelo matemático que representa o problema. A Lógica *Fuzzy* não exige a Função de Transferência, mas apenas o raciocínio de um especialista para montar o conjunto de regras.

O fluxograma apresentado na FIG. 27 sumariza como um sistema *Fuzzy* deve ser desenvolvido.

Figura 27 - Fluxograma de um sistema de controle *Fuzzy*



Fonte: O autor.

3.2 Biblioteca eFLL (*Embedded Fuzzy Logic Library*)

A biblioteca para sistemas embarcados eFLL foi desenvolvida pelo *Robotic Research Group* (RRG – Grupo de Pesquisa Robótica) da Universidade Estadual do Piauí em 2011 e pode ser utilizada em qualquer plataforma que aceite instruções em linguagem C++, incluindo a plataforma Arduino. A limitação do uso da biblioteca encontra-se na capacidade de memória do dispositivo para armazenar todas as variáveis necessárias aos cálculos por esse método (KRIDI, ALVES, *et al.*, 2012).

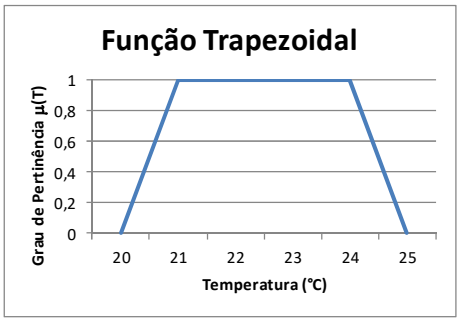
A biblioteca utiliza o método de inferência Máximo-e-Mínimo, o método de composição Mínimo de Mamdani e o método de “Defuzzyficação” por cálculo do centro de área (PRADO e MASSELLI, 2017).

No trabalho de Kride (2012) a biblioteca é descrita como uma composição de 4 classes que contém métodos específicos para a configuração e execução do processamento. Fuzzy.cpp é a classe principal onde se determina a quantidade de

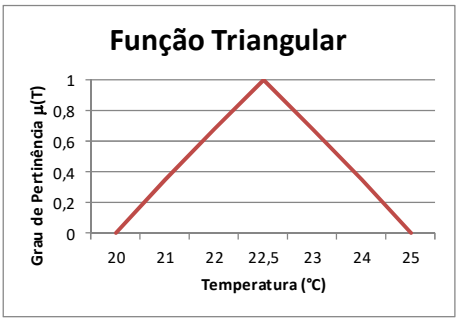
entradas do sistema, determinam-se as regras, executa-se o processo de “fuzzyficação” e “defuzzyficação”, além de se determinar o ponto em que a “fuzzyficação” atinge os conjuntos das variáveis. Contudo, no trabalho de Prado (2017) observa-se que a biblioteca recebeu aprimoramentos para permitir a adoção de infinitas entradas e saídas, além de simplificar seu uso.

O objeto “Fuzzy” engloba o sistema *Fuzzy* para manipular os conjuntos de entradas e saídas e as regras linguísticas. O objeto “FuzzyInput” agrupa os conjuntos de entrada pertencentes ao mesmo domínio, e da mesma forma trabalha o objeto “FuzzyOutput”. O objeto “FuzzySet” realiza o modelamento do sistema, permitindo a construção das funções de pertinência através de quatro pontos, que podem resultar em funções trapezoidais, triangulares ou *singleton*, como pode ser observado pela FIG. 28. O primeiro ponto é o ponto de mínimo, seguido pelo ponto de máximo 1, depois pelo ponto de máximo 2, e por fim o limite superior.

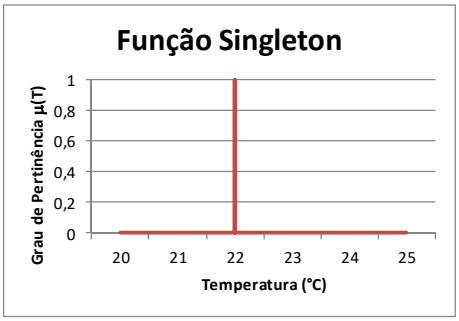
Figura 28 - Possíveis funções através do objeto *FuzzySet*



FuzzySet* fs= FuzzySet(20, 21, 24, 25)



FuzzySet* fs= FuzzySet(20, 22.5, 22.5, 25)



FuzzySet* fs= FuzzySet(22, 22, 22, 22)

Fonte: O autor.

O objeto “FuzzyRule” constrói o conjunto de regras que forma a base de conhecimento do sistema *Fuzzy*. Cada regra é formada por uma condição

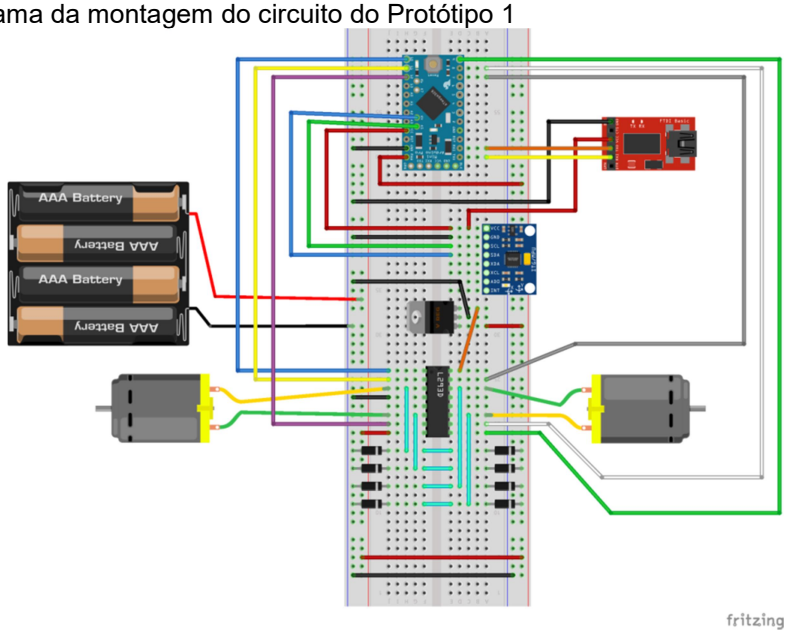
representada pelo objeto “FuzzyRuleAntecedent” e uma consequência representada pelo objeto “FuzzyRuleConsequent”. As condições podem ser combinadas através de operadores *AND* e *OR*, utilizando-se respectivamente as funções “joinWithAND” e “joinWithOR”, obtendo-se regras mais complexas. A biblioteca aplica a o método do mínimo para o operador *AND* e o máximo para o operador *OR*.

4 RESULTADOS OBTIDOS

4.1 Desenvolvimento Eletrônico e Mecânico – Protótipo 1

Foi definido para este trabalho o desenvolvimento de um robô equilibrista em duas rodas adotando motores elétricos de corrente contínua acoplados em caixas de redução mecânica para aumento de torque e redução de velocidade, localizados na base de uma estrutura vertical plástica. Os motores são controlados por um circuito integrado L293D capaz de amplificar os sinais PWM enviados pelo micro controlador e inverter a polaridade para troca do sentido de rotação. O sensor utilizado é o MPU-6050 que se comunica com o micro controlador através de um barramento I²C, e o conjunto todo é alimentado por um conjunto de baterias recarregáveis de 1,2V totalizando 4,8V de tensão. O controlador é uma placa Arduino Pro Mini versão 3,3V e 8MHz de frequência de *clock*. Foi utilizado um adaptador USB/Serial para permitir comunicação com o computador para a programação do Arduino e para leitura dos dados provenientes dos sensores do IMU.

Figura 29 - Diagrama da montagem do circuito do Protótipo 1



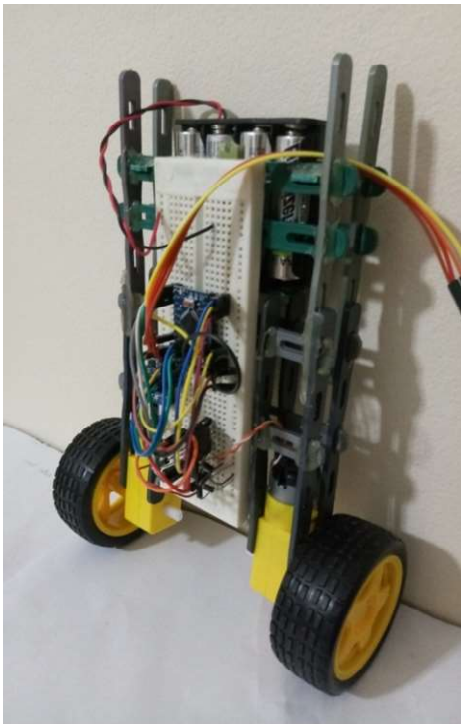
Fonte: O autor.

O primeiro protótipo foi construído com o intuito de conhecer as características básicas dos componentes antes de se desenvolver placas de circuito e dispositivos mecânicos mais robustos e complexos, pois ainda não era certo que os componentes utilizados seriam suficientes para o cumprimento do objetivo. O

diagrama de montagem do circuito do primeiro protótipo é mostrado na FIG. 29, ilustrado através do programa *Fritzing*.

O robô é apresentado na FIG. 30, construído a partir de hastes plásticas encaixadas e unidas por cola quente. A estrutura abriga uma matriz de contatos na qual foi montado o primeiro circuito de controle. O conjunto de baterias foi acomodado na parte superior do robô, em uma espécie de gaveta, de modo que a maior massa do robô se concentrasse longe do solo conforme orientações dos trabalhos de referência consultados. Na matriz de contatos, o circuito de acionamento dos motores foi posicionado na parte inferior, enquanto a parte superior abrigava o controlador Arduino Pro Mini, com o módulo IMU posicionado entre ambos.

Figura 30 - Protótipo 1 montado



Fonte: O autor.

O primeiro protótipo foi fundamental para a compreensão do funcionamento dos componentes essenciais à solução do problema, como o módulo GY-521, que contém o componente MPU-6050, e o componente L293D. Foi realizada a ligação dos motores, sendo o motor esquerdo ligado aos pinos 1Y e 2Y (saídas do L293D), e o motor direito aos pinos 3Y e 4Y. As entradas correspondentes foram ligadas ao Arduino Pro Mini, além dos pinos de *Enable* e as alimentações do CI. Desse modo,

para ligar o motor esquerdo em um sentido, primeiro deve-se enviar nível lógico alto para o pino 1A e nível lógico baixo ao pino 2A e então nível lógico alto ao pino EN1 para habilitar as saídas. Para inverter o sentido de rotação do motor, deve-se inverter o nível lógico das entradas 1A e 2A, e então habilitar as saídas através de EN1 novamente. Para ativar o motor da direita deve-se realizar o procedimento análogo. Se forem enviados Alto e Alto ou Baixo e Baixo nos pinos 1A e 2A não ocorre curto-circuito, mas os motores não são energizados com diferença de potencial que possa provocar circulação de corrente, permanecendo em repouso.

Assim, os pinos de controle PWM foram conectados aos pinos de ENABLE do L293, e os pinos de controle de orientação ligados a pinos digitais sem a função PWM do Arduino. O circuito foi projetado antes da chegada do material e, por não ter certeza se seria entregue um L293 ou um L293D, foram inseridos os diodos de proteção na montagem, mas como foi recebido o L293D que já possui proteção interna, esses diodos foram removidos do projeto.

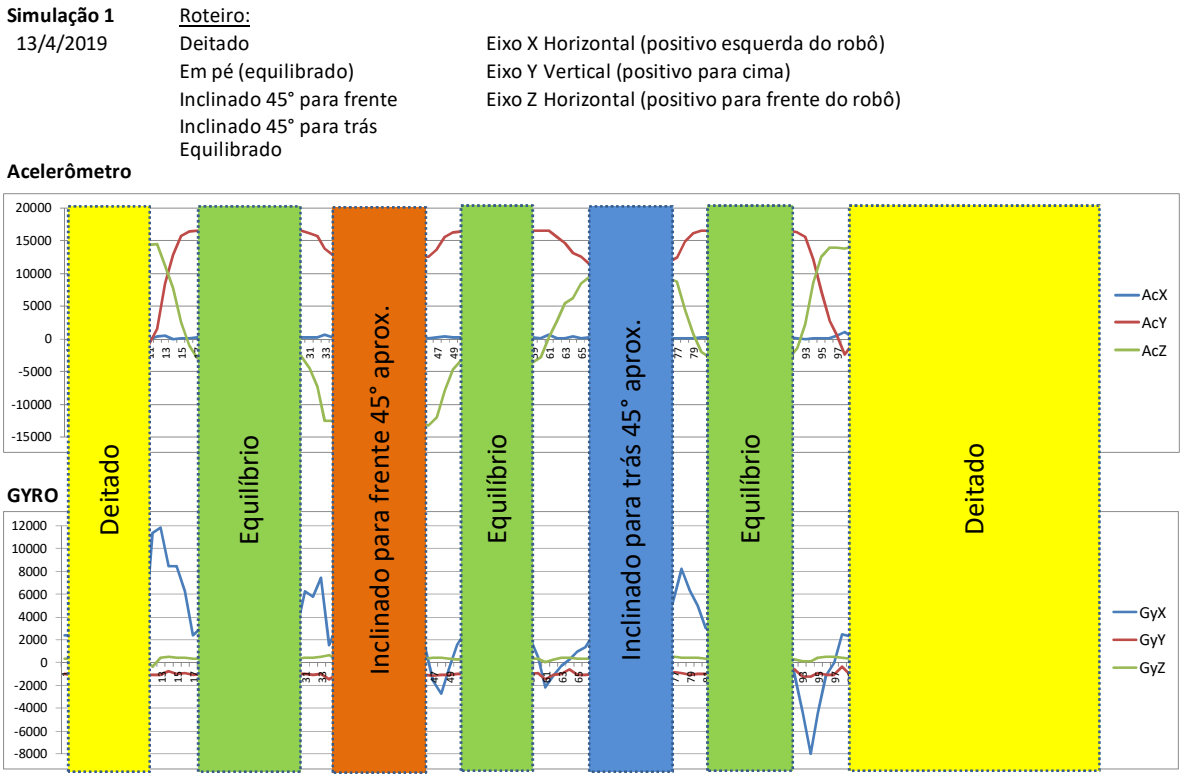
Nesta etapa do projeto foi elaborado um software simples para ativação dos motores através de comandos enviados pela porta serial, controlando-se a velocidade e a sentido de rotação dos motores, construindo o conhecimento necessário ao controle de movimento para as próximas etapas.

Foram realizados testes no IMU MPU-6050 para compreender seu funcionamento, realizando as leituras dos três eixos do acelerômetro e dos três eixos do giroscópio, plotando os dados em gráficos enquanto se alterava o ângulo da estrutura seguindo um roteiro previamente definido, conforme a FIG. 31, mostrando que o cálculo do ângulo de inclinação deve ser realizado pelos valores dos eixos X e Z do Acelerômetro, considerando-se a orientação do sensor nesse primeiro protótipo.

Observou-se que as medições do giroscópio em torno do eixo X apresentou um valor constante pouco superior a 2000 mesmo quando o robô permanecia na posição de repouso (considera-se deitado o robô com a parte traseira apoiada em superfície horizontal), e que os valores eram positivos quando a estrutura girava em torno do eixo das rodas no sentido da frente do robô, e valores subtraíam do valor

de repouso quando o giro era no sentido das costas do robô e, quanto mais rápido esse movimento acontecia, maior a amplitude do sinal. Observa-se pela FIG. 31 que os picos do giroscópio acontecem justamente nas transições das posições definidas no roteiro.

Figura 31 - Primeiro teste de leitura dos dados do MPU-6050

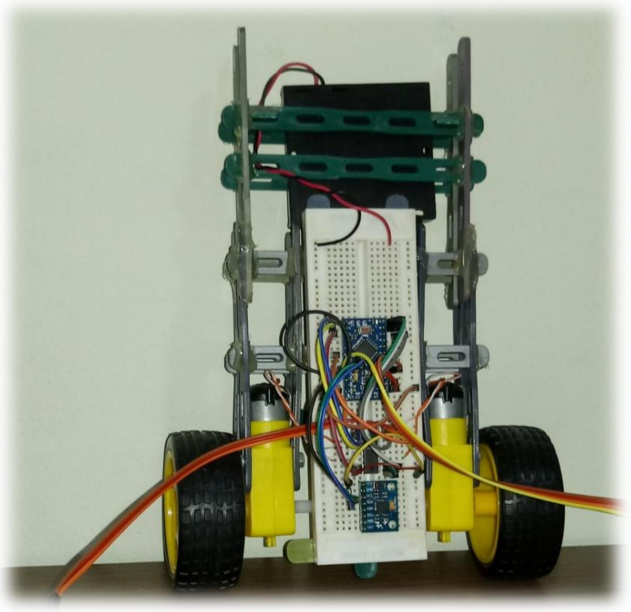


Fonte: O autor.

Para estudar a interferência da posição do sensor nas medições, o MPU-6050 foi realocado para uma posição inferior, entre os motores, conforme a FIG. 32, e então os dados foram lidos novamente, mas não foram observadas diferenças significativas conforme é apresentado na FIG. 33.

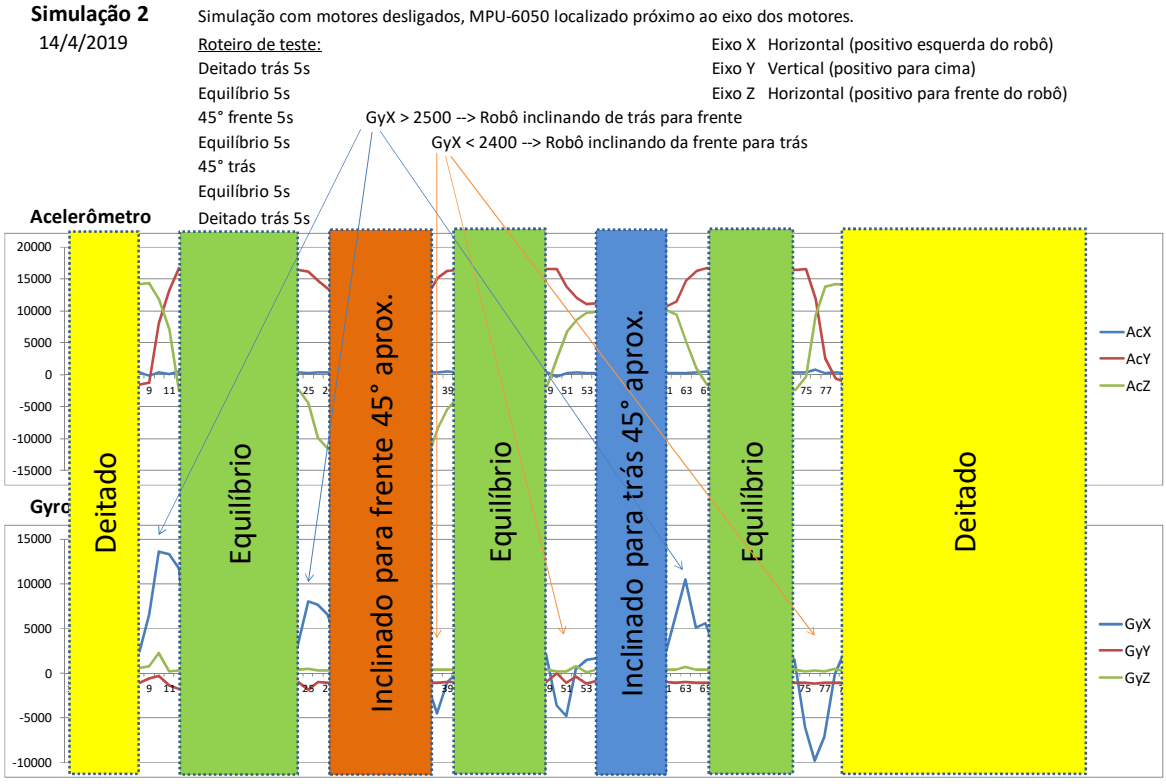
Foi dada ênfase à observação do comportamento do giroscópio e confirmado que o mesmo possui um valor razoavelmente constante mesmo em repouso, e seu acréscimo ou decréscimo depende do sentido do giro e da amplitude da velocidade angular em torno do eixo analisado. Foi identificado também que o cálculo do ângulo de inclinação do robô não pode considerar apenas um dos eixos do sensor, mas sim a amplitude do vetor do eixo y em relação ao plano formado pelos eixos x e z, considerando-se a posição relativa do módulo sensor no protótipo.

Figura 32 - Protótipo com sensor reposicionado entre os eixos das rodas



Fonte: O autor.

Figura 33 - Releitura dos dados do MPU-6050



Fonte: O autor.

Através desse experimento foi possível compreender o funcionamento dos sensores e verificar como o acelerômetro é sensível a vibrações, e que o uso dos

dados brutos dificulta a inserção em um controlador, sendo necessário o cálculo do ângulo através de funções trigonométricas e, segundo a literatura consultada, aplicação de tratamento dos sinais para fusão dos dados e posterior filtragem para eliminação de ruídos, adotando-se o filtro de Kalman para essa dupla função, obtendo-se resultados muito satisfatórios.

Foram realizados testes com esse protótipo anexando-se apoios na estrutura de modo que o robô permanecesse inclinado a 15° da posição vertical desejada, e aplicando-se comando para a maior velocidade nos motores para verificar sua capacidade de recuperar o ângulo de inclinação para 0° e posterior inclinação para o sentido oposto. Os testes mostraram que a velocidade dos motores era insuficiente com a alimentação de 4,8V. Como medida corretiva, foi adicionado um segundo conjunto de 4 baterias recarregáveis para totalizar 9,6V aplicados aos motores, o que aumentou a velocidade dos mesmos, mas também ocasionou o aumento da massa do conjunto. Contudo, novos testes comprovaram a eficácia da ação limitando-se o ângulo de inclinação para o intervalo de ±10°.

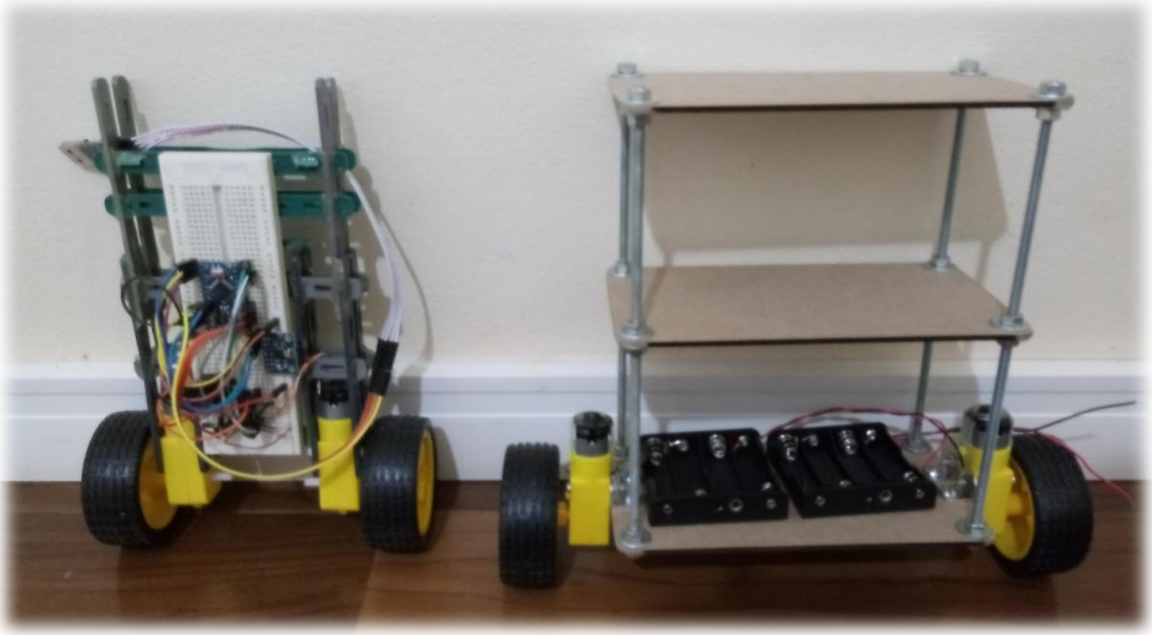
Iniciados os testes para o desenvolvimento do controlador PID (seguindo a estratégia para validar o sistema eletromecânico para então adquirir conhecimento em relação ao controlador nebuloso), constatou-se que as medições de ângulo eram muito impactadas pelo movimento relativo de componentes da própria estrutura do robô, como os suportes de baterias e a matriz de contatos na qual o circuito eletrônico estava montado. Foram realizados trabalhos na tentativa de eliminar essas folgas, mas logo se observou a necessidade de melhorar o projeto mecânico do robô, utilizando-se das anotações de pontos fortes e pontos fracos registrados durante os testes do primeiro protótipo.

4.2 Desenvolvimento do Protótipo 2

Para o desenvolvimento do protótipo 2 foram consideradas as observações realizadas no protótipo 1. Inicialmente foi considerado a montagem de uma estrutura em três andares de placas de MDF de 3mm, como pode ser observado pela FIG. 34, mas logo que a estrutura ficou pronta foi observado que o sistema físico seria muito

pesado para a capacidade de torque dos motores empregados e essa ideia foi descartada.

Figura 34 - À esquerda, protótipo 1; à direita, estrutura do protótipo 2 - descartada



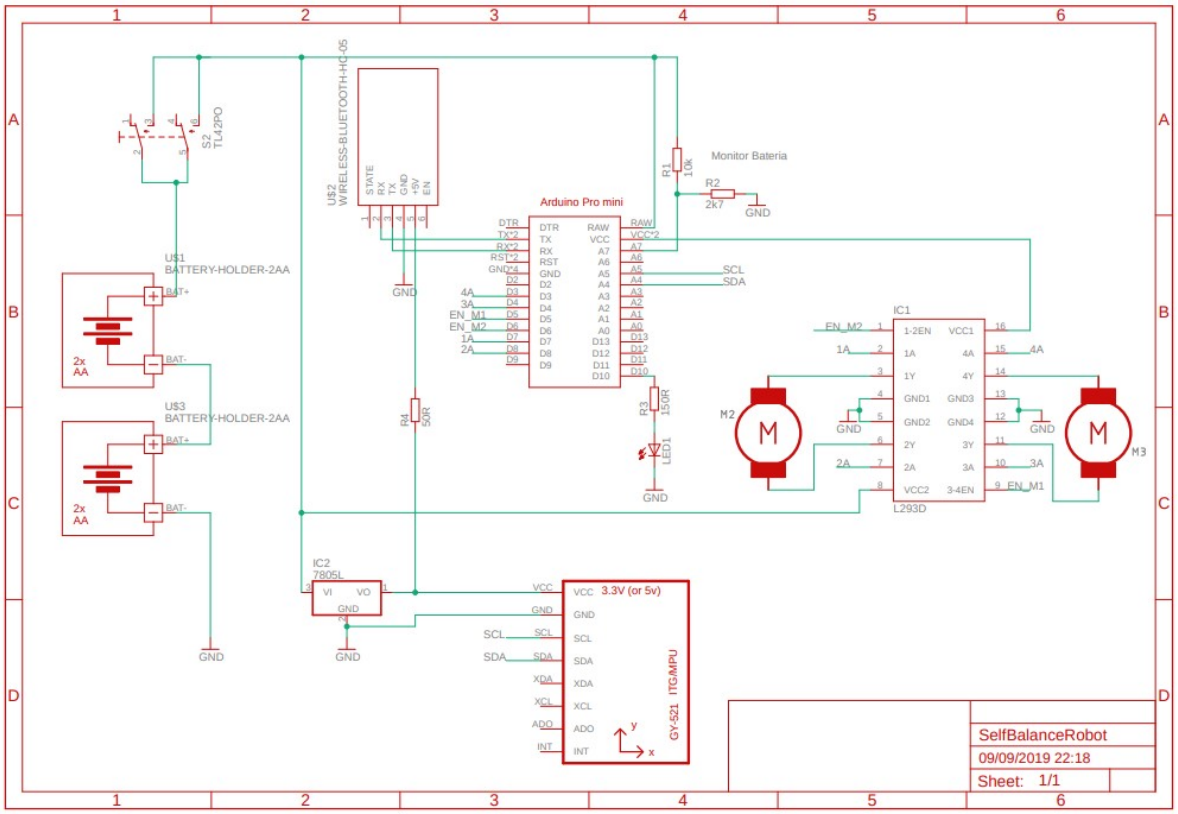
Fonte: O autor.

Decidiu-se, então, utilizar a mesma forma do protótipo 1, utilizando apenas uma placa de MDF de 3mm na vertical com os componentes fixados a ela. Com essa ideia, foram desenhados os diagramas do circuito eletrônico para controlar o robô, adicionando-se um módulo de comunicação Bluetooth para permitir interação com dispositivos externos, por exemplo, um celular Android para controle e aquisição de dados.

A FIG. 35 mostra o primeiro desses circuitos ainda utilizando o Arduino Pro Mini como unidade de processamento e um regulador de tensão LM7805L para alimentar os módulos GY-521 e o módulo *Bluetooth*.

Foi adicionado um circuito divisor de tensão a uma das entradas analógicas do Arduino para permitir a leitura da tensão da bateria e, assim, permitir compensação da descarga da bateria. Através de uma rotina *software* pode-se monitorar a tensão da bateria e, dessa forma, compensar os valores PWM enviados aos motores e o robô poderia manter o mesmo desempenho quando a bateria estiver totalmente carregada ou estiver, por exemplo, a 70% de sua capacidade.

Figura 35 - Primeira versão do circuito desenhado para o segundo protótipo



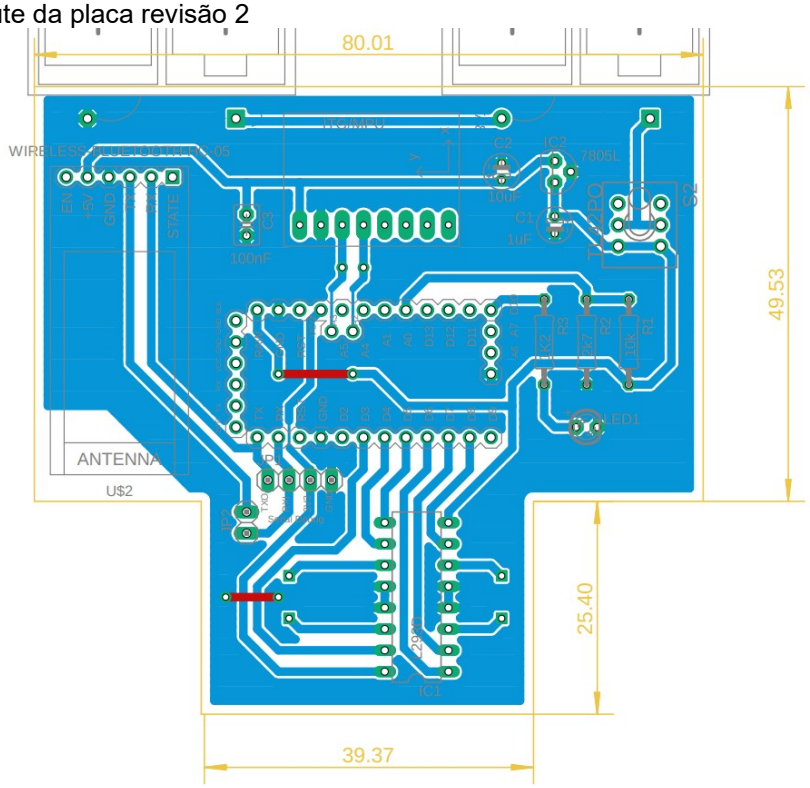
Fonte: O autor.

Utilizando a versão gratuita do software Eagle da Autodesk, foram desenhadas diversas versões de leiautes para a placa de circuito impresso enquanto os testes eram realizados em matriz de contatos, buscando-se um circuito estável para a aplicação. Foram incluídos filtros capacitivos e resistores de proteção como pode ser observado no leiaute da FIG. 36.

Os testes em matriz de contatos mostraram que o regulador de tensão LM7805L era insuficiente para alimentar ao mesmo tempo o Arduino Pro Mini, o módulo sensor GY-521 e o módulo *Bluetooth*, e então foi necessário substituir o regulador de tensão por um LM7805 com capacidade de fornecimento de corrente de 1A.

O *jumper* JP2 foi incluído para conectar/desconectar o pino de TX do módulo *Bluetooth* ao restante do circuito para não interferir nas comunicações durante a programação do Arduino. Sem essa modificação, seria necessário remover-se o módulo toda vez que uma nova versão do firmware fosse descarregada na placa de controle.

Figura 36 - Leiaute da placa revisão 2



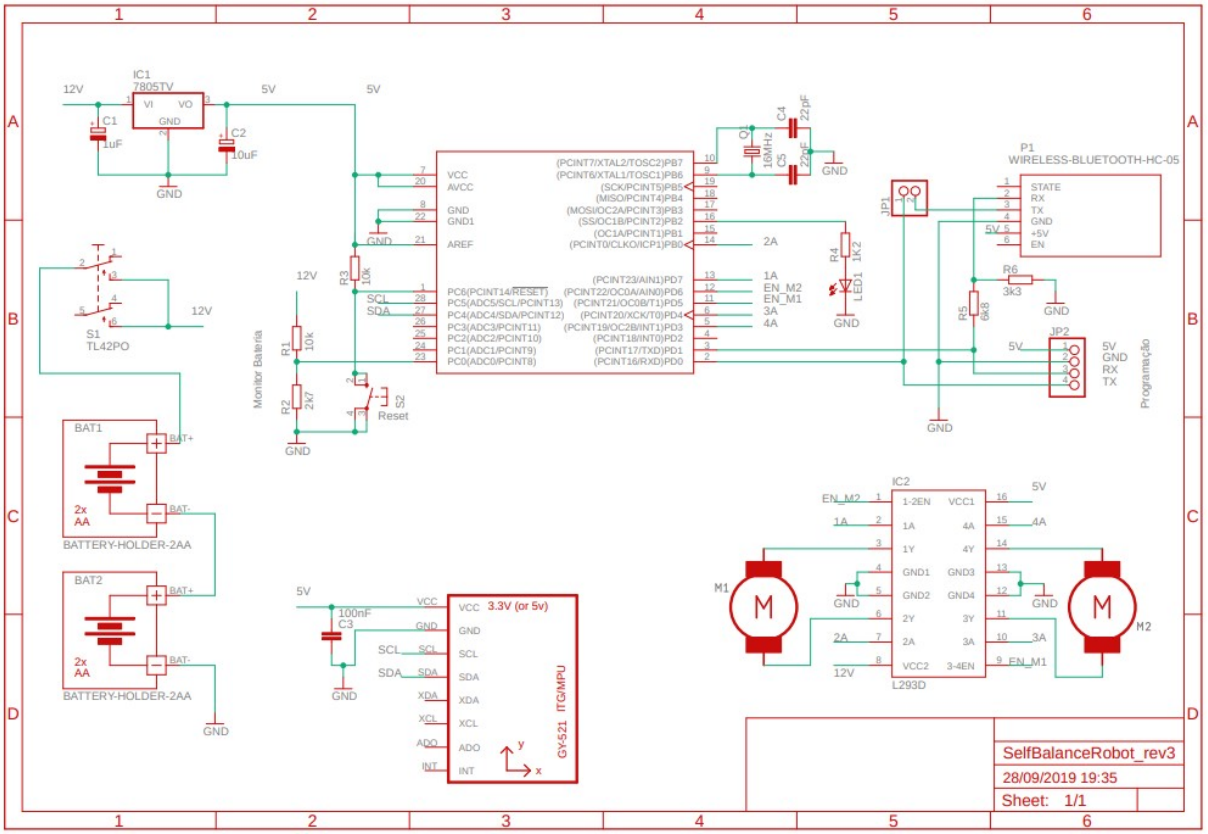
Fonte: O autor.

Após mais alguns testes foi constatado que o uso de uma placa Arduino Pro Mini de 3,3V e 8MHz não seria suficiente para o controle do robô, uma vez que o processamento dos sinais deve ser realizado de forma que a resposta ao sistema físico seja suficientemente rápida. Por isso foi desenvolvida uma terceira versão do circuito utilizando somente o micro controlador ATmega328P alimentado em 5V e com frequência de trabalho de 16MHz (o mesmo padrão da placa Arduino Uno), como pode ser observado na FIG. 37.

Os valores dos resistores do circuito divisor de tensão para monitoramento da bateria foram revisados considerando-se a nova tensão de operação do micro controlador, e foi adicionado um circuito divisor de tensão ao pino de recepção do módulo *Bluetooth* para compatibilizar à sua tensão de operação. Foi necessário trocar a função de alguns pinos digitais anteriormente definidos para adequação à nova realidade, facilitando, assim, o desenho da placa de circuito impresso. Essa troca de pinos exigiu também a adequação do software em desenvolvimento para correspondência aos novos pinos. Essa versão de placa incluiu um barramento de

pinos específicos para a programação do micro controlador, de modo a não necessitar a remoção do módulo Bluetooth para executar-se essa tarefa.

Figura 37 - Circuito eletrônico versão 3



Fonte: O autor.

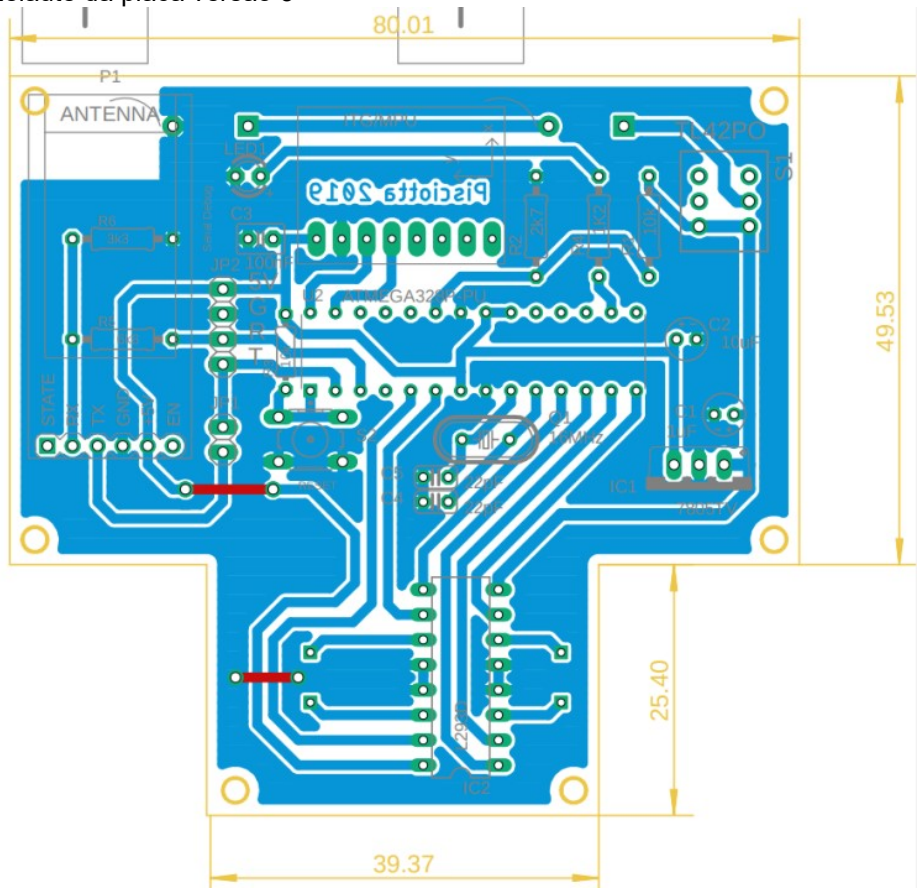
A versão três do circuito possui um capacitor de 100nF próximo aos terminais de alimentação do módulo GY-521 que abriga o IMU MPU-6050, uma vez que fóruns da internet registram relatos de problemas com ruído nas leituras do sensor devido a um erro de montagem do fabricante em um determinado lote. O circuito possui uma chave geral que conecta a alimentação das baterias ao regulador de tensão IC1 e ao controlador dos motores IC2. O regulador IC1 regula a tensão para 5V alimentando o micro controlador, o módulo Bluetooth e o módulo GY-521.

Além dos componentes essenciais, como o ressonador de 16MHz Q1, os capacitores de acoplamento C4 e C5 que fazem parte do circuito de *clock* e o resistor R3 com o botão de pressão que faz parte do circuito de *reset* do micro controlador, foi adicionado um LED com o resistor R4 para limitar a corrente para indicar o funcionamento do sistema através de emissão intermitente. Caso o micro controlador trave a execução do software, o LED cessará de piscar, indicando o mau

funcionamento do sistema e facilitando a investigação de problemas durante o desenvolvimento do projeto.

O conector JP1 conecta o pino de RX do módulo Bluetooth ao pino TX do micro controlador durante o uso normal e, ao ser removido, permite que o sistema receba programas do IDE do Arduino sem a necessidade de remover o módulo da placa principal. O circuito divisor de tensão formado por R1 e R2 compatibiliza a tensão da bateria para a tensão máxima de trabalho do micro controlador para que o mesmo possa medir a tensão da bateria sem ser danificado.

Figura 38 - Leiaute da placa versão 3

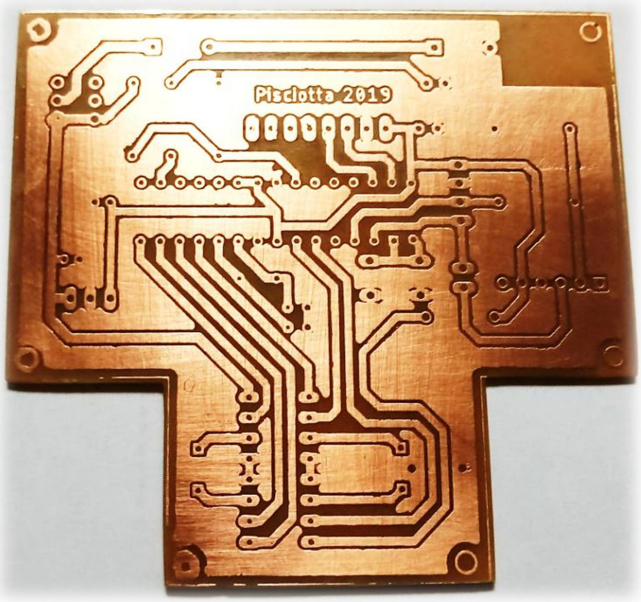


Fonte: O autor.

A FIG. 38 mostra o leiaute da placa de circuito impresso na versão 3, com o micro controlador e seus periféricos essenciais.

Foi realizada a confecção da placa de circuito impresso em placa cobreada através do método de transferência térmica e corrosão do cobre utilizando percloreto de ferro, e o resultado pode ser observado na FIG. 39.

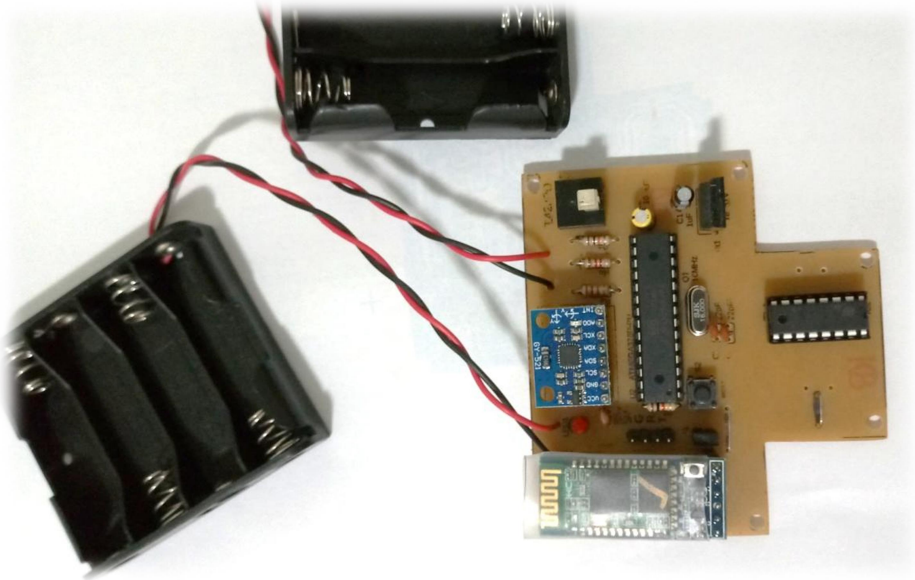
Figura 39 - Placa de circuito impresso confeccionada



Fonte: O autor.

A placa foi protegida por verniz incolor e os componentes foram soldados, como mostrado na FIG. 40.

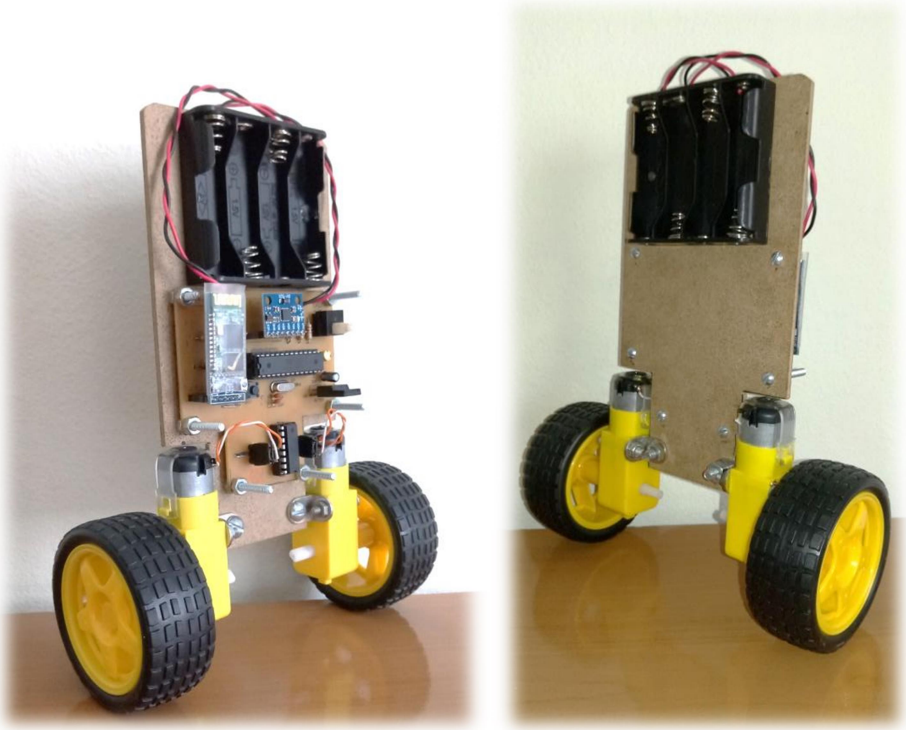
Figura 40 - Placa eletrônica finalizada para aplicação no protótipo 2



Fonte: O autor.

A estrutura do protótipo 2 foi montada e todos os componentes fixados por parafusos, conforme FIG. 41, para permitir a realização dos primeiros testes.

Figura 41 - Protótipo 2 montado (frente à esquerda e traseira à direita), pronto para testes



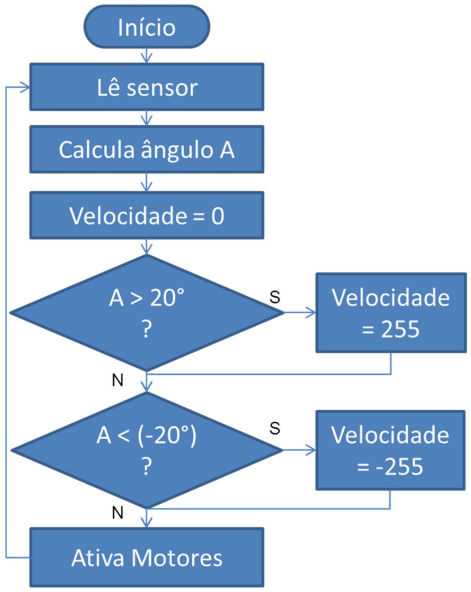
Fonte: O autor.

Para a realização do primeiro teste funcional do hardware foi elaborado um software simples para piscar o LED de indicação de estado à frequência de 1Hz, efetuar a leitura do ângulo de inclinação através do acelerômetro aplicando-se trigonometria nas leituras dos três eixos e acionar os motores com velocidade máxima no sentido da inclinação caso o ângulo calculado seja maior que 20° em qualquer dos dois sentidos, conforme o fluxograma da FIG. 42. Para inclinações cujo módulo seja inferior a 20°, os motores são colocados em estado de repouso.

Contudo, durante o teste de validação de hardware, foi observado que um dos motores girava em apenas um dos sentidos. Analisando o circuito foi encontrado um curto-circuito no cobre da placa em um dos pinos do circuito integrado L293D que impedia o recebimento do sinal do controlador, deixando inoperante uma parte do circuito de potência desse motor.

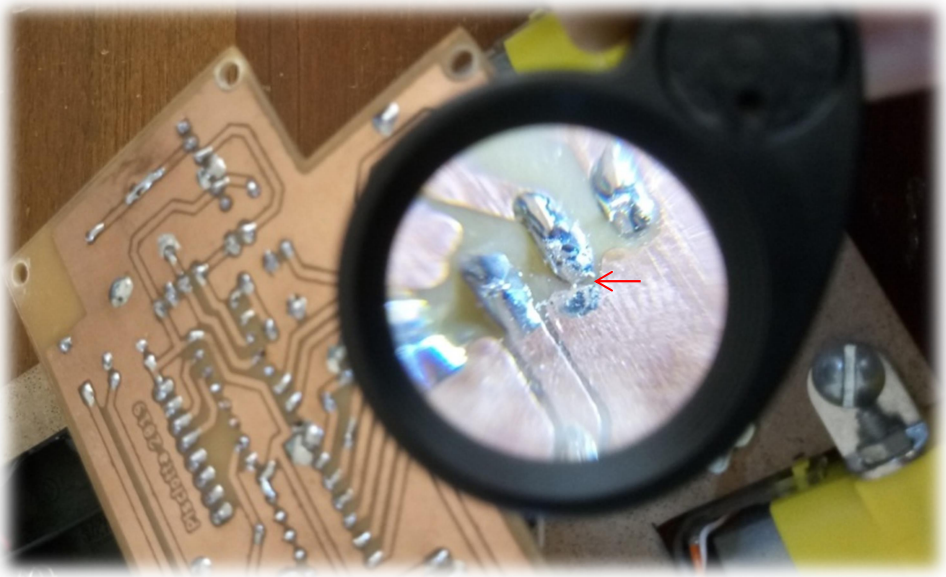
A FIG. 43 evidencia esse curto-circuito quase imperceptível a olho nu. Foi realizado o reparo nessa ilha da placa através da remoção da camada de cobre e o teste foi repetido com sucesso.

Figura 42 - Fluxograma do firmware de teste de hardware



Fonte: O autor.

Figura 43 - Curto-circuito no terminal do L293D

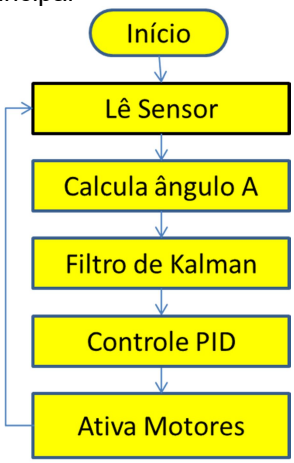


Fonte: O autor.

Com a placa reparada, foi dado início ao desenvolvimento do software de leitura do ângulo de inclinação do robô pelo acelerômetro e giroscópio combinados através do filtro de Kalman, e acionamento dos motores inicialmente por controle PID. A estratégia foi atingir a estabilidade do robô pelo controle PID para assim validar o conjunto eletromecânico, para depois iniciar o desenvolvimento do controlador por Lógica Nebulosa.

Essa estratégia foi adotada porque, tratando-se de um projeto experimental, houve a preocupação de que um erro na definição de características físicas comprometeria o sucesso do controlador *Fuzzy* e não estaria claro se o motivo seria o controlador *Fuzzy* ou qualquer outra característica.

Figura 44 - Fluxograma do firmware principal



Fonte: O autor.

A primeira etapa do firmware desenvolvido, como observado no fluxograma da FIG. 44, é a leitura do ângulo dos sensores, acessando os registradores do MPU-6050 através da comunicação I^2C. Isso é feito enviando-se um comando com o endereço do sensor, seguido do endereço do primeiro registrador desejado e a quantidade de endereços consecutivos que se deseja receber. Os valores lidos de cada eixo do giroscópio, do acelerômetro e do termômetro interno são armazenados em dois registradores de 8 bits, que devem ser lidos individualmente e combinados para formar uma palavra de 16 bits para cada informação desejada. Dessa forma, são lidos 14 registradores que são combinados dois a dois para obtenção de sete informações correspondentes aos três eixos do giroscópio, três eixos do acelerômetro e temperatura interna, mesmo que esta última não seja utilizada no projeto.

Os três eixos do acelerômetro são utilizados para o cálculo do ângulo de inclinação utilizando trigonometria, calculando-se o arco-tangente do ângulo formado entre o eixo X e o plano formado entre os eixos Y e Z. A velocidade angular obtida pelo giroscópio considerada é apenas aquela ao redor do eixo Y. Esses dados são combinados pelo filtro de Kalman para obter-se a medida estável do ângulo de inclinação do robô. O valor do ângulo é aplicado ao controlador PID para

determinação dos valores de saída que representam os valores de PWM enviados aos motores.

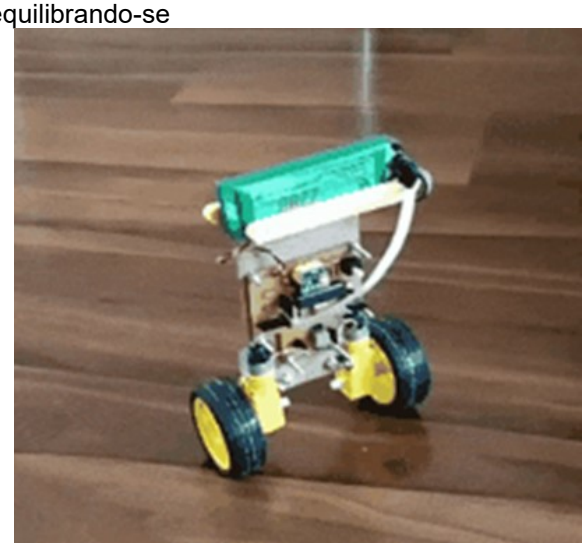
Os primeiros testes de equilíbrio mostraram que a resposta do controlador estava abaixo do necessário para corrigir o ângulo, impossibilitando o equilíbrio. Os valores das constantes multiplicadoras kp, ki e kd foram ajustados até se obter o comportamento de equilíbrio, que pode ser observado na FIG. 45.

Figura 45 - Robô equilibrando pela primeira vez

Fonte: O autor.

Embora o equilíbrio tenha sido atingido, o robô apresentou oscilação em torno da posição vertical, e ficou muito sensível a perturbações externas, de modo que qualquer “esbarrão” fosse suficiente para que o robô perdesse a capacidade de retornar à estabilidade. Além disso, mesmo sem perturbações externas, o robô perdia a capacidade de equilíbrio após 3 minutos de funcionamento. Foi constatado que o circuito integrado L293D se aquecia, ativando o sistema de proteção interno que desliga os motores até a temperatura retornar a valores seguros. Por isso foi adicionado um dissipador de calor ao componente, e também foi substituído o conjunto de pilhas recarregáveis por uma bateria de Lítio-Polímero de três células em série, totalizando 11,1V de tensão e capacidade de corrente de 2,2A. Foram removidos os suportes das oito baterias e confeccionado um novo suporte de material PET para fixar a nova bateria no topo do robô, como pode ser observado na FIG. 46, e o comportamento obtido foi muito superior. Obviamente os parâmetros do controlador PID foram reajustados para a nova realidade.

Figura 46 - Protótipo 2.1 equilibrando-se



Fonte: O autor.

O equilíbrio dessa versão do protótipo ainda foi insatisfatório, pois apresentou sensibilidade a perturbações externas, além de grande oscilação na tentativa de encontrar o ponto de equilíbrio. Chegou-se à conclusão que essa oscilação se dava pela limitação dos motores em atingir acelerações suficientes para a correção da posição da base no tempo necessário. Por isso foi reduzida a altura do centro de gravidade do robô alterando-se a posição da bateria, conforme pode ser observado na FIG. 47, e o resultado foi muito melhor, com menor oscilação e melhor recuperação a perturbações externas.

Figura 47 - Protótipo versão 2.2 - robusto contra perturbações externas

Fonte: O autor.

A continuidade desses testes resultou em um acidente em que o robô sofreu uma queda de três metros, danificando sua estrutura conforme FIG. 48.

Figura 48 - Danos estruturais ao protótipo após queda

Fonte: O autor.

Foi realizado o reparo da estrutura para a continuidade dos testes, mas logo foi decidido desenvolver a terceira versão do protótipo, com uma nova estrutura e todos os componentes centralizados para se obter o máximo equilíbrio.

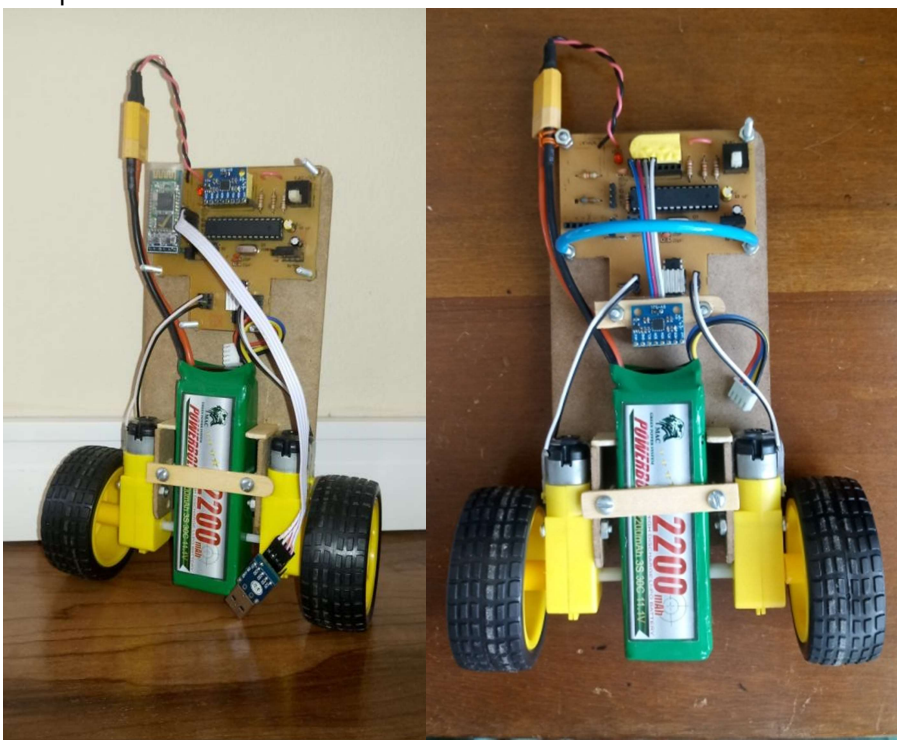
4.3 Desenvolvimento do Protótipo número 3

O protótipo número 3 foi desenvolvido considerando-se a bateria entre os motores de corrente contínua, a placa de circuito impresso foi levada para a parte superior e o sensor foi posicionado no centro de massa do robô, de forma a não sofrer movimentações desnecessárias. Houve preocupação em manter todos os componentes centralizados nos eixos Y e Z, de forma que as oscilações naturais do sistema representem movimentos puramente angulares no MPU-6050, minimizando medidas imprecisas nos sensores, como pode ser observado na FIG. 49.

Foram instaladas proteções de borracha na parte frontal e traseira do robô para evitar batidas secas contra o piso em caso de perda de equilíbrio, e a bateria foi fixada através de parafusos em uma haste de madeira que a prende contra camadas de borracha para mantê-la aderente na posição desejada. O suporte do protótipo

anterior possuía folga que causava a vibração da bateria, criando perturbações nas medições do sensor.

Figura 49 - Protótipo 3 montado com bateria mais abaixo e opção do sensor ao centro de massa do robô ou em seu topo



Fonte: O autor.

O robô apresentou melhores resultados com o sensor localizado ao centro de massa do robô do que com o sensor ao topo porque neste último o sensor fica sujeito a acelerações lineares que interferem na medida do ângulo instantâneo obtido pelo acelerômetro. Já no centro de massa, os movimentos resultantes são majoritariamente de inclinação, exercendo maior influência nas medições obtidas pelo giroscópio do que no acelerômetro.

Através de pesquisas a fóruns da internet foi constatado que a resposta do sistema, quando colocado sobre um carpete, era muito superior, apresentando menor oscilação. Testes sobre um carpete comprovaram essa informação, por isso foram confeccionados aros macios utilizando mangueiras de silicone e instalados na superfície das rodas do robô, obtendo-se uma resposta do sistema com menor oscilação, a ponto de o sistema físico ser considerado validado.

Figura 50 - Sistema físico validado

Fonte: O autor.

Com isso, foi possível prosseguir os trabalhos para o desenvolvimento do controlador *Fuzzy*. A FIG. 50 apresenta a versão do protótipo com os aros macios citados e o equilíbrio possível, com baixa oscilação e robusto contra perturbações externas.

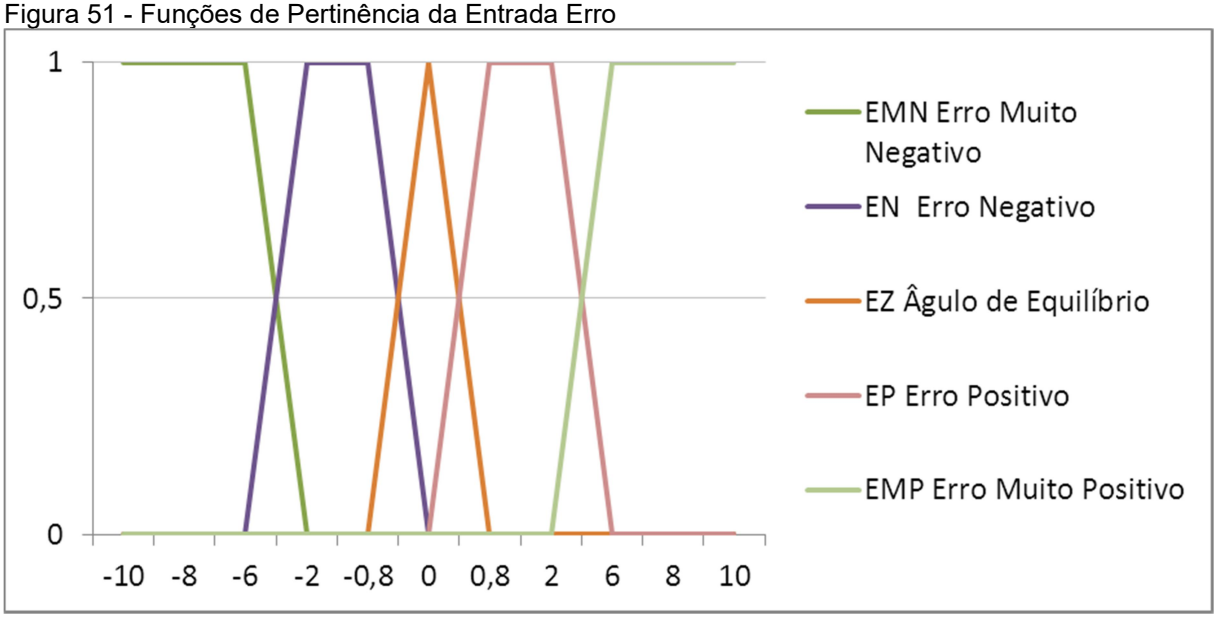
4.4 Desenvolvimento do Controlador *Fuzzy*

Foram definidas as variáveis linguísticas de entrada e de saída e seus respectivos graus de pertinência, conforme apresentado nas FIG. 51, 52 e 53.

A variável chamada **Erro** representa a diferença entre o ângulo atual medido pelos sensores e o ângulo desejado, indicando a inclinação do robô em relação à referência desejada.

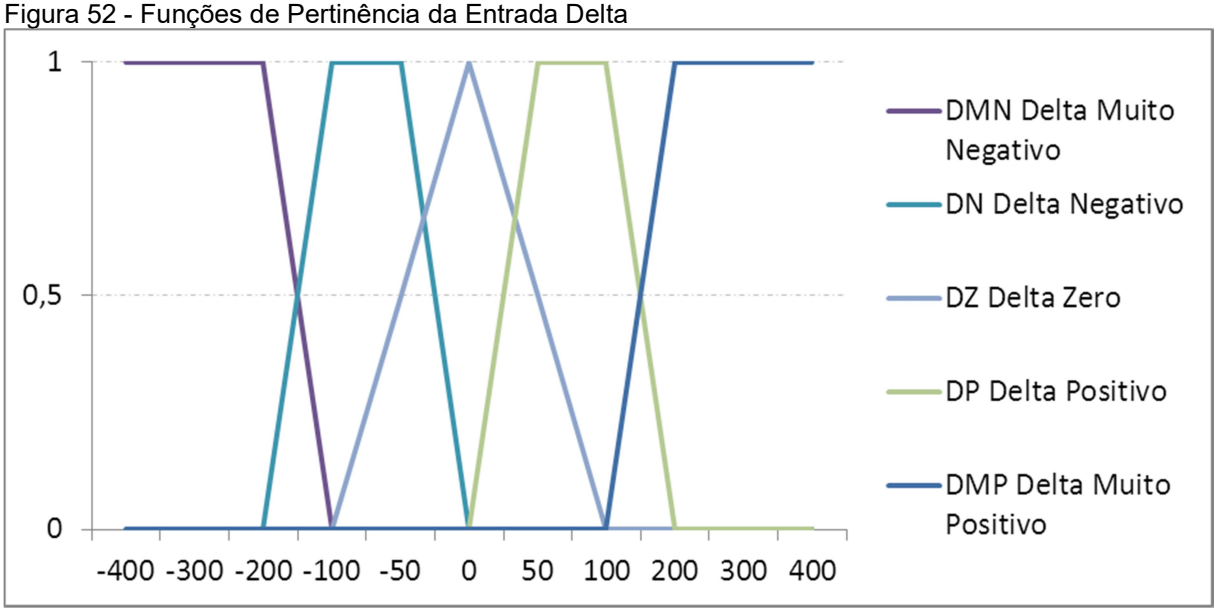
Delta Erro representa a diferença entre o erro atual e o erro do ciclo anterior, provendo a informação de quão rápido a variação do erro acontece, de forma que a resposta seja condizente à necessidade.

Figura 51 - Funções de Pertinência da Entrada Erro



Fonte: O autor.

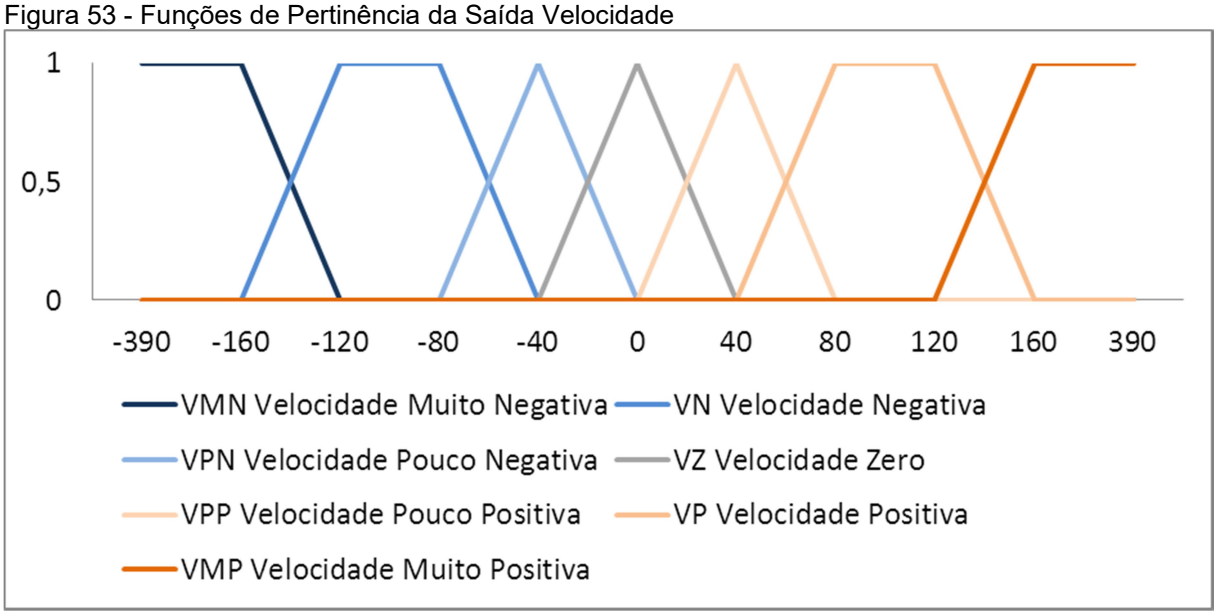
Figura 52 - Funções de Pertinência da Entrada Delta



Fonte: O autor.

A saída velocidade corresponde ao valor PWM a ser enviado aos motores, que compreende valores entre 0 e 255. Os limites inferior e superior da variável linguística Velocidade (VMN e VMP) passam do limite de 255 porque o método de "defuzzyficação" da biblioteca eFLL é o centro de gravidade da figura resultante, ou seja, apresentará como saída máxima o valor correspondente ao centro de gravidade do trapézio de VMN ou VMP, que deve ser -255 e 255 respectivamente.

Figura 53 - Funções de Pertinência da Saída Velocidade



Fonte: O autor.

Após a definição das variáveis linguísticas, seus termos e as funções de pertinência, foi definido o conjunto de regras do sistema de controle, baseado em observações do sistema físico durante os testes anteriores e na intuição do autor, resultando na tabela 3. Nela se observa, por exemplo, “Se Delta Muito Negativo (DMN) E Erro Muito Negativo (EMN) ENTÃO Velocidade Muito Positiva (VMP)”.

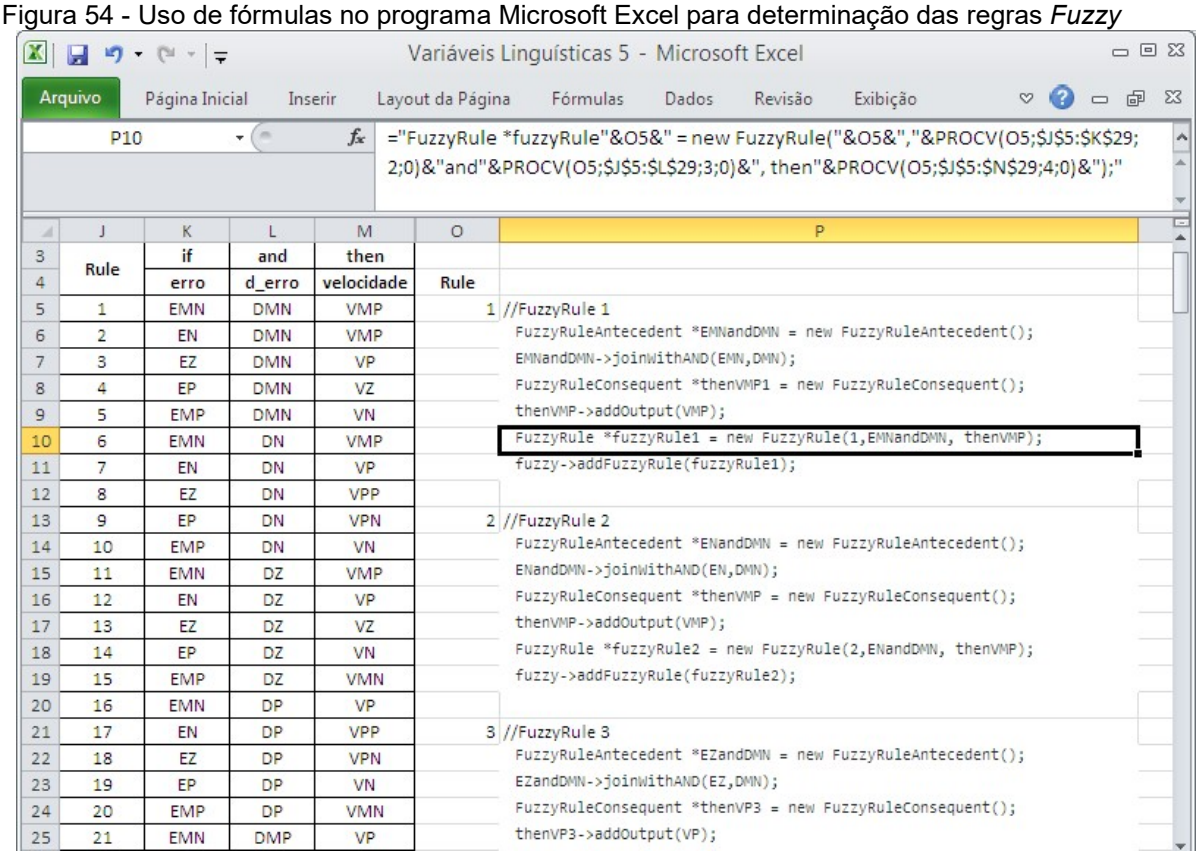
Tabela 3 - Conjunto de Regras Fuzzy

Delta / Erro	EMN	EN	EZ	EP	EMP
DMN	VMP	VMP	VP	VZ	VN
DN	VMP	VP	VPP	VPN	VN
DZ	VMP	VP	VZ	VN	VMN
DP	VP	VPP	VPN	VN	VMN
DMP	VP	VZ	VN	VMN	VMN

Fonte: O autor.

Com a definição do conjunto de regras “se – então” obteve-se um conjunto de 25 combinações possíveis entre as entradas, com sete diferentes resultados possíveis na saída. Para codificar as vinte e cinco regras no formato exigido pela biblioteca eFLL foram utilizadas fórmulas no programa Microsoft Excel para facilitar a escrita do código referente às declarações da biblioteca, como pode ser observado na FIG. 54.

Figura 54 - Uso de fórmulas no programa Microsoft Excel para determinação das regras *Fuzzy*

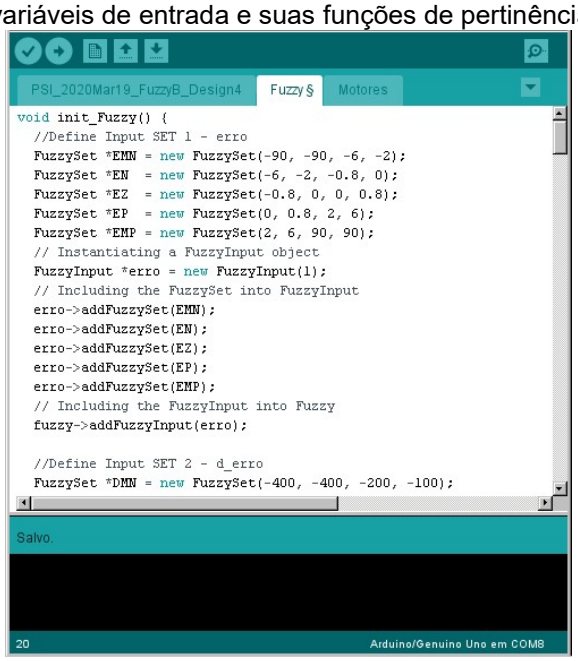


	J	K	L	M	O	P
3	Rule	if	and	then		
4		erro	d_erro	velocidade	Rule	
5	1	EMN	DMN	VMP	1	//FuzzyRule 1
6	2	EN	DMN	VMP		FuzzyRuleAntecedent *EMNandDMN = new FuzzyRuleAntecedent();
7	3	EZ	DMN	VP		EMNandDMN->joinWithAND(EMN,DMN);
8	4	EP	DMN	VZ		FuzzyRuleConsequent *thenVMP1 = new FuzzyRuleConsequent();
9	5	EMP	DMN	VN		thenVMP->addOutput(VMP);
10	6	EMN	DN	VMP		FuzzyRule *fuzzyRule1 = new FuzzyRule(1,EMNandDMN, thenVMP);
11	7	EN	DN	VP		fuzzy->addFuzzyRule(fuzzyRule1);
12	8	EZ	DN	VPP		
13	9	EP	DN	VPN	2	//FuzzyRule 2
14	10	EMP	DN	VN		FuzzyRuleAntecedent *ENandDMN = new FuzzyRuleAntecedent();
15	11	EMN	DZ	VMP		ENandDMN->joinWithAND(EN,DMN);
16	12	EN	DZ	VP		FuzzyRuleConsequent *thenVMP = new FuzzyRuleConsequent();
17	13	EZ	DZ	VZ		thenVMP->addOutput(VMP);
18	14	EP	DZ	VN		FuzzyRule *fuzzyRule2 = new FuzzyRule(2,ENandDMN, thenVMP);
19	15	EMP	DZ	VMN		fuzzy->addFuzzyRule(fuzzyRule2);
20	16	EMN	DP	VP		
21	17	EN	DP	VPP	3	//FuzzyRule 3
22	18	EZ	DP	VPN		FuzzyRuleAntecedent *EZandDMN = new FuzzyRuleAntecedent();
23	19	EP	DP	VN		EZandDMN->joinWithAND(EZ,DMN);
24	20	EMP	DP	VMN		FuzzyRuleConsequent *thenVP3 = new FuzzyRuleConsequent();
25	21	EMN	DMP	VP		thenVP3->addOutput(VP);

Fonte: O autor.

Após a determinação das regras, foram escritas as instruções da lógica Nebulosa no micro controlador utilizando o IDE Arduino, substituindo-se o controlador PID desenvolvido anteriormente.

Figura 55 - Definição das variáveis de entrada e suas funções de pertinência

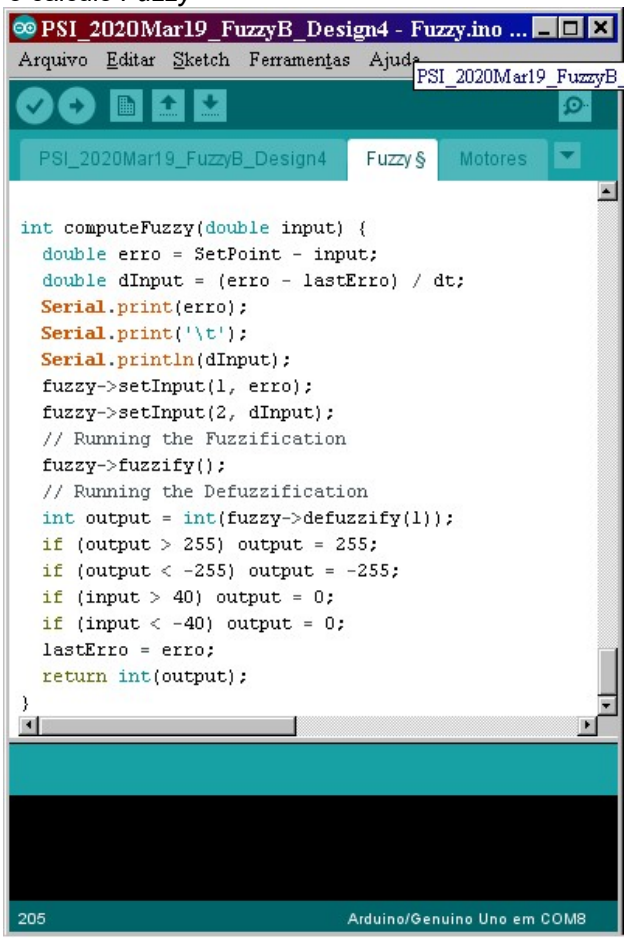


Fonte: O autor.

Foram declaradas as funções de pertinência da entrada erro, como pode ser observado pela FIG. 55 e, de forma semelhante, foram definidas as funções de pertinência da entrada “d_erro” e da saída velocidade. Em seguida foram definidas as regras através dos objetos da biblioteca eFLL *FuzzyRuleAntecedent*, que forma as condições, *FuzzyRuleConsequent* que forma os resultados decorrentes das condições e *addFuzzyRule* que relaciona as condições aos resultados.

Após as declarações de construção do sistema *Fuzzy* o programa chama a função *computeFuzzy* (FIG. 56) para execução dos cálculos, retornando o valor aos motores. A saída é limitada entre -255 e +255 para evitar ultrapassar os limites do PWM, e caso o valor do ângulo lido seja acima de 40° para qualquer das direções, os motores são desligados, pois esses ângulos representam impossibilidade de recuperação de equilíbrio, e o robô apenas continuaria se deslocando em velocidade máxima na horizontal, podendo ser facilmente danificado.

Figura 56 – Função para o cálculo *Fuzzy*



Fonte: O autor.

A primeira tentativa de utilização da biblioteca eFLL para construir um controlador *Fuzzy* no micro controlador através da IDE Arduino considerando vinte e cinco regras distintas não apresentou erro de compilação, contudo os motores mantiveram-se inoperantes. Foram revistos todos os códigos em busca de erros de digitação, mas nenhum erro foi encontrado.

Durante a investigação das causas foram comentadas no programa a maioria das declarações de regras, deixando apenas três regras com objetos de consequência distintos, e quando o programa foi regravado, os motores começaram a responder de acordo com as regras atuantes. Houve suspeita de que a causa seria a existência de diferentes regras para um mesmo resultado de saída. Por exemplo, quatro diferentes combinações das entradas resultavam na saída VMP. Dessa forma foram utilizados operadores *AND* e *OR* para combinar as regras de forma a obter-se apenas uma regra para cada possível saída, totalizando sete, como pode ser observado na tabela 4.

Tabela 4 - Novo conjunto de regras Fuzzy

Se	**Então**
EPandDMNorEZandDZorENandDMP	VZ
EZandDNorENandDP	VPP
EPandDNorEZandDP	VPN
EZandDMNorENandDNorENandDZorEMNandDPorEMNandDMP	VP
EMPandDMNorEMPandDNorEPandDZorEPandDPorEZandDMP	VN
EMNandDMNorENandDMNorEMNandDNorEMNandDZ	VMP
EMPandDZorEMPandDPorEPandDMPorEMPandDMP	VMN

Fonte: O autor.

Os novos testes resultaram ainda em ausência de movimento e, durante nova análise, foi desativada uma das regras (a maior delas) e então o robô começou a responder com velocidades variantes em seus motores de acordo com a inclinação do robô. Levantou-se a hipótese de que a regra desativada possuía algum erro de escrita, mas logo foi observado que reativando-se essa regra e desativando-se qualquer outra aleatoriamente, o sistema também respondia.

Então sete regras ainda resultavam em inatividade dos motores, mas se qualquer regra fosse aleatoriamente comentada, os motores funcionavam. Concluiu-se então que o limitante era a quantidade de memória necessária para a criação de

todas as regras, insuficiente no ATmega328P. Então foi decidido por desativar a regra da saída referente a Velocidade Zero (VZ), permanecendo apenas seis regras em operação. Foi escolhido VZ porque essa é a regra que mantém os motores desligados quando o robô está em equilíbrio.

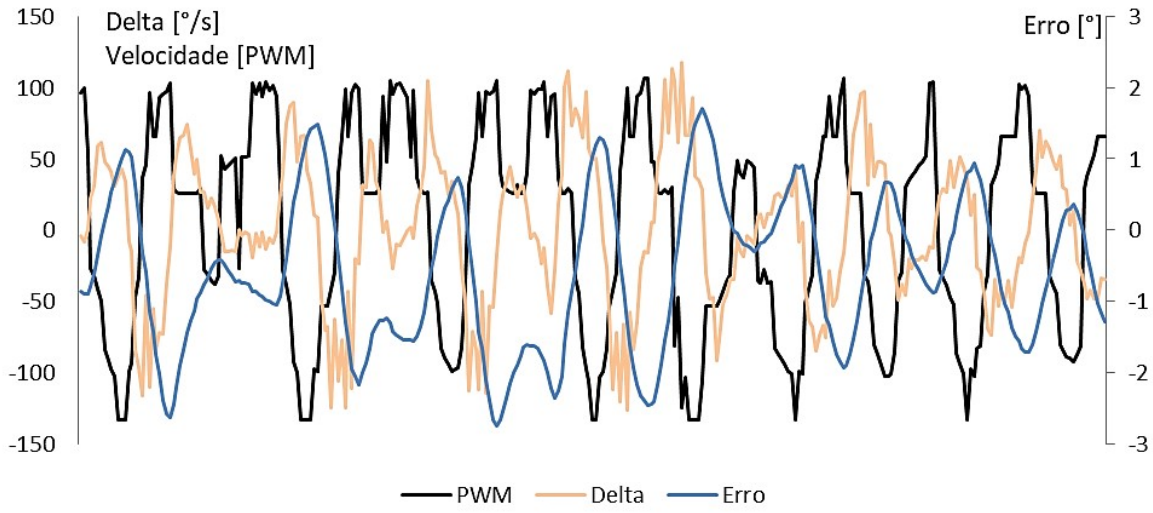
Depois que o número de regras foi reduzido para seis e o programa foi recompilado, o sistema respondeu com variações na saída e depois de ajustado o ângulo de equilíbrio (*offset*) o robô foi capaz de se equilibrar, atuando os motores na direção em que o robô se inclina com velocidade proporcional ao ângulo e à variação do mesmo, de forma a buscar o equilíbrio na posição vertical, como pode ser observado pela FIG. 57.

Figura 57 - Robô equilibrando-se através da Lógica *Fuzzy*

Fonte: O autor.

O sistema busca o equilíbrio dinâmico em torno do ângulo de equilíbrio instável, e parte da vibração é causada principalmente por folgas no conjunto de engrenagens e pela imprecisão natural dos motores de corrente contínua. A Fig. 58 apresenta o gráfico que relaciona as variáveis de entrada Erro em graus e Delta em graus por segundo e a saída Velocidade (adimensional) do controlador *Fuzzy* enquanto o robô está em equilíbrio dinâmico.

Figura 58 - Variáveis de entrada antes da “fuzzyficação” e variável de saída após “defuzzyficação” durante equilíbrio dinâmico



Fonte: o autor.

5 CONCLUSÃO

Como resultado deste trabalho construiu-se um sistema eletromecânico e computacional para um robô equilibrista, constatando-se sua elevada complexidade, exigindo-se muita pesquisa e repetitiva execução de testes para compreensão do comportamento dinâmico do sistema, com a vantagem de um controlador *Fuzzy* não exigir o conhecimento do modelo matemático do sistema físico. O robô apresentado neste trabalho é capaz de se equilibrar utilizando-se lógica nebulosa, conforme objetivo principal, proporcionando a compreensão dos conceitos envolvidos no desenvolvimento de um controlador baseado na lógica nebulosa, além do sensoriamento por acelerômetros e giroscópios e os métodos para combinação de dados e filtragem para eliminação de ruídos ou mesmo obtenção de estimativas. A necessidade de se combinar e reduzir o número de regras *Fuzzy* para aplicações em que haja limite de memória foi certamente de grande valia na solução de problemas futuros.

Foi observada na prática a principal vantagem da Lógica Nebulosa em relação ao controlador PID: a dispensa de determinação do modelo matemático do dispositivo, podendo utilizar o conhecimento de um especialista para definir as variáveis linguísticas, as funções de pertinência e o conjunto de regras utilizadas na solução do problema. Comparando-se o processo de testes entre os dois métodos de controle utilizados no desenvolvimento deste trabalho, foi surpreendente o quão rápido obtiveram-se resultados satisfatórios de equilíbrio com o controlador por lógica nebulosa.

Concluiu-se que a utilização do giroscópio e acelerômetro combinados através do filtro de Kalman proporciona uma medição eficaz e segura para se atingir a estabilidade do conjunto, e afirmo que todos os componentes devem ser firmemente fixados na estrutura para não gerar vibrações prejudiciais às medições pelo acelerômetro. O uso do L293D não é recomendado para esse tipo de projeto, uma vez que a alteração brusca de direção da rotação dos motores faz a corrente se elevar, e o componente em questão limita-se a 600mA, comprometendo o resultado das manobras quando aquece-se.

5.1 Sugestões para continuação da pesquisa

É possível com baixo investimento substituir as rodas por outras de menor diâmetro, que resultariam em menor deslocamento livre (sem controle) ocasionado pelas folgas existentes nas caixas de engrenagem, e o uso de material mais macio para a superfície das rodas conferiria ao robô um comportamento dinâmico mais suave.

Com algum aumento de custos, a troca dos atuadores por motores de passo e seus respectivos controladores eliminaria completamente todas as folgas do sistema e proporcionaria controle total na atuação das rodas, resultando em um sistema de equilíbrio suave e preciso. E mesmo que se opte por continuar a utilizar motores de corrente contínua, sugiro que sejam escolhidos conjuntos de maior qualidade, com menores folgas e maior torque disponível nos eixos de saída, além do uso de um *driver* de maior capacidade de potência.

É possível implantar no protótipo desenvolvido a funcionalidade de comunicação remota para controlar seu deslocamento utilizando-se um dispositivo *Bluetooth* através de pequenas alterações no firmware, uma vez que o circuito eletrônico já foi preparado para essa função.

Cita-se ainda como boa opção para continuidade deste projeto a alteração de uma das variáveis linguísticas da entrada do sistema *Fuzzy*, substituindo-se a variável Delta pela variável Gyro, que representa a velocidade angular quando o robô realmente está em mudança de ângulo.

Como sugestão indica-se emprego de micro controlador de maior capacidade de memória e maior velocidade de processamento também pode proporcionar melhores resultados ao sistema dinâmico, principalmente se permitir a inserção de todas as regras definidas pelo especialista da planta.

O sistema pode ser utilizado como base para implantação de modelos de movimento que permitam o controle do robô em trajetórias determinadas, adicionando-se sensores e associando códigos de *software* apropriados.

6 REFERÊNCIAS

ALLEGRO MICROSYSTEMS INC. **A4988**. [S.l.]. 2012.

ARDUINO CC. About Us. **Arduino CC**, 2019. Disponivel em: <https://www.arduino.cc/en/Main/AboutUs>. Acesso em: 21 abr. 2019.

BARRAGÁN, H. Copyright \ Wiring. **Wiring**, 2018. Disponivel em: <http://wiring.org.co/copyright.html>. Acesso em: 12 ago. 2018.

BHATTI, O. S.; MEHMOOD-UL-HASAN, K.; IMTIAZ, M. A. **Attitude Control and Stabilization of a Two-Wheeled Self-Balancing Robot**. CEAI, 17, 2015. 98-104.

BLOMSTEDT, J.; HARALDSSON, J.; NORDIN, J. **Expressive Arduino Controlled Self-Balancing Robot.** UPPSALA Universitet, 2016.

BONAFILIA, B. et al. **Self-Balancing Two-Wheeled Robot.** Chalmers University of Technology, 2013.

BUENO, A. G.; ROMANO, R. A. **Filtro Complementar Aplicado a Medida de Inclinação de Plataformas Móveis.** Escola de Engenharia Mauá (EEM/CEUN-IMT), 2014.

CHINNADURAI, G.; RANGANATHAN, H. **IOT Controlled Two Wheel Self Supporting Robot Without External Sensor**. Middle-East Journal of Scientific Research 23 (Sensing, Signal Processing and Security), 2015. 286-290.

CHUN-HONG, H.; BIN, R. **Design of Two Wheel Self-balancing Car**. IOP Conference Series: Earth and Environmental Science 113. [S.l.]: [s.n.]. 2018.

CUNHA, A. **A vez dos Acelerômetros**, São Paulo, n. 419, 2007.

DAN, N.; WANG, J. **Two Wheeled Self Balancing Control Research**. Indonesian Journal of Electrical Engineering and Computer Science, 2, June 2015. 617-624.

ERIKSSON, E. **Self-Balancing Robot Control System in CODESYS for Raspberry Pi.** Umeå University, 2016.

ESMAEILI, N.; ALFI, A.; KHOSRAVI, H. **Balancing and Trajectory Tracking of Two-Wheeled Mobile Robot Using Backstepping Sliding Mode Control: Design and Experiments.** Springer Science + Business Media Dordrecht, 2017.

FANG, J. **The Research on the Application of Fuzzy Immune PD Algoorithm in the Two-Wheeled and Self-balancing Robot System**. International Journal of Control and Automation Vol. 7, No. 10, 2014. 109-118.

FANG, J. **A Posture Control System Design for a Two-wheeled and Self-balancing Robot**. International Journal of Computer, Consumer and Control (IJ3C), 4, 2015.

FRANKOVSKY, P. et al. **Modeling of Two-wheeled Self-balancing Robot Driven by DC Gearmotors**. International Journal of Applied Mechanics and Engineering, Vol. 22, No. 3, 2017. 739-747.

GHANI, N. M. A.; NAIM, F.; YON, T. P. **Two Wheels Balancing Robot with Line Following Capability**. International Journal of Mechanical and Mechatronics Engineering Vol. 5 No 7, 2011.

GOMIDE, F. A. C.; GUDWIN, R. R. **Modelagem, Controle, Sistemas e Lógica Fuzzy**. SBA Controle & Automação, Campinas, 4, 1994.

GONZALEZ, C.; ALVARADO, I.; MUÑOZ DE LA PEÑA, D. **Low Cost Two-Wheels Self-Balancing Robot for Control Education**. Sevilla. 2017.

HAN, H. Y.; HAN, T. Y.; JO, H. S. **Development of Omnidirectional Self-Balancing Robot**. ResearchGate, Faculty of Engineering, Computing and Science, Swinburne University of Technology Sarawak Campus, Kuching, Malaysia, 2014.

INVENSENSE INC. **MPU-6000 and MPU-6050 Register Map and Descriptions Revision 4.2**. [S.l.]. 2013.

INVENSENSE INC. **MPU-6000 and MPU-6050 Product Specification Revision 3.3**. [S.l.]. 2012.

KEDZIERSKI, J.; TCHON, K. **Feedback Control of a Balancing Robot**. Wroctaw University of Technology, ul. Janiszewkiego 11/17, 2017. 50-317.

KRIDI, D. S. et al. **Desenvolvimento de Uma Biblioteca Fuzzy para o Controle Autônomo de um Robô Móvel em Ambiente Desconhecido**. Universidade Estadual do Piauí, Teresina, 2012.

KRISHNA, B. S. B. V.; RAO, P. M. **Implementation of Two Wheeled Self-balancing Platform**. International Research Journal of Engineering and Technology (IRJET), Vol. 3, 2016.

LANGLEY, R. **Development of a Self-balancing Robot Utilizing FPGA**. Murdoch University, Perth, Australia, 2016.

LAUBLI, D. et al. **Self-balancing Two Wheeled Robot**. [S.l.]. 2015.

LEKSHMY, S.; GEORGE, A.; ATHIRA, C. V. **Self Balancing Robot**. International Journal of Computer Engineering in Reasearch Trends, Vol. 2, Issue 12, 2015. 1091-1095.

MACHADO, E. R. M. D. **Modelagem e Controle de Sistemas Fuzzy Takagi-Sugeno**. UNESP - Universidade Estadual Paulista, Ilha Solteira, 2003.

MARTÍNEZ, J. D. **Balancing Robot: Fuzzy Control of the Tilt Angle and Communications with Android Devices**. Hong Kong. 2015.

MICROCHIP TECHNOLOGY INC. **Atmel Supplier Letter**. [S.l.]. 2016.

MICROCHIP TECHNOLOGY INC. **ATmega48A/PA/88A/PA/168A/PA/328/P Datasheet**. [S.l.]. 2018.

NGUYEN, D.; PHUONG, K. **SB-Bot A Self-balancing Robot**. KTH Industrial Engineering and Management, Stockholm, 2016.

OLIVEIRA, E. J. L.; SILVA, F. R.; BARBOSA, L. F. W. **Inertial Device Based in Gyroscope for Robotics Application**. II INIC Jr Latin America Meeting of Scientific Research. [S.l.]: [s.n.]. 2008.

PHAN, H. N.; NGUYEN, C. X. **Building Embedded Quasi-time-optimal Controller for Two-wheeled Self-balancing Robot**. Le Quy Don Technical University, Hanoi, Vietnam, 2017.

PRADO, L. A.; MASSELLI, Y. M. C. **Fuzzy Logic and Arduino Application on Process Control**. SAISEE - Industrial Automation and Electric-Electronics Systems Seminar, INATEL, 2017.

PRASETIO, B. H.; KURNIAWAN, W. **Disturbance Rejection Using Feed-forward Control System on Self Balancing Robot**. MATEC Web of Conferences 154, University of Brawijaya, Indonesia, 2018.

RAM, S. S.; KUMAR, D. D. **Designing of Optimization Techniques Based PID Controller for Self-balancing Bicycle**. International Journal of Advances in Computer and Electronics Engineering, Vol. 2, Issue 6, 2017. 21-25.

RIGNEL, D. G. S.; CHENCI, G. P.; LUCAS, C. A. **Uma Introdução à Lógica Fuzzy**. Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica. Vol. 1, No. 1, 2011.

SIQUEIRA, A. L. L.; CASTRO, E. B. **Modern Control in States Space of an Inverted Pendulum with Nonlinear Modeling**. Campo dos Goyatacazes - RJ. 2013.

SPARKFUN ELECTRONICS. **Arduino Pro Mini 328 - 3.3V / 8MHz**. Sparkfun, 2019. Disponivel em: <https://www.sparkfun.com/products/11114>. Acesso em: 21 abr. 2019.

TAUSCHECK, S. **Giroscópios de Silício Micromecânicos**. Revista Elektor eletrônica e microinformática n. 73, 2007.

TEXAS INSTRUMENTS. **L293x Quadruple Half-H Drivers Datasheet**. [S.l.]. 2016.

TOMAšIć, T.; DEMETLIKA, A.; CRNKOVIC, M. **Self-balancing Mobile Robot Tilter**. Transactions of Famena XXXVI-3, 2012.

TORIGE, G. L. **Construction of a Self-balancing Vehicle Prototype**. Revista da Graduação, Pontifícia Universidade Católica do Rio Grande do Sul, 2013.

VELAZQUEZ, M. et al. **Velocity and Motion Control of a Self-balancing Vehicle Based on a Cascade Control Strategy**. International Journal of Advanced Robotic Systems, 2016.

YAO, J. **The Study on Design and Algorithm of Robot Control System**. International Conference on Information Sciences, Machinery, Materials and Energy (ICISMME), 2015.