

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA DE ENGENHARIA / ENGENHARIA DE CONTROLE E AUTOMAÇÃO
Trabalho Final de Curso II

Matheus de Oliveira e Silva

ROBÔ DELTA: IMPLEMENTAÇÃO DE CINEMÁTICA E EMBARCADOS

Trabalho Final de Curso como parte dos requisitos para obtenção do título de bacharel em Engenharia de Controle de Automação apresentado à Pontifícia Universidade Católica de Goiás.

BANCA EXAMINADORA:

Prof. Dr. Bruno Quirino de Oliveira
Prof. Dr. Antônio Marcos de Melo Medeiros
Prof. Me. Vitor Hugo Martins e Resende

Goiânia, 01 de junho de 2021

Robô Delta: Implementação de Cinemática e Embarcados.

Matheus de Oliveira e Silva, Bruno Quirino de Oliveira, Antônio Marcos de M. Medeiros e Vitor Hugo M. e Resende
Engenharia de Controle e Automação, PUC Goiás

Resumo - Com a modernização das indústrias e o avanço das tecnologias nas mais diversas áreas a precisão, velocidade e a diminuição dos riscos devem ser prioridade. Com esses ideais notamos que a renovação de tecnologias antigas ou aplicação de novas tecnologias vem-se tornando essencial para se manter relevante no mercado. Com essa demanda em vista, foi feito o estudo para implantação de embarcados e novas tecnologias em um robô paralelo tipo delta, para que ele seja capaz de executar tarefas ágeis e precisas de maneira mais confiável e responsiva. Com esses resultados como metas, foram levantadas diversas informações sobre tecnologias novas, como microcontroladores mais potentes e rápidos, implantação de interface homem máquina, tornando mais simples a programação e visualização das informações contidas no robô e o cálculo de cinemática para que seja diminuída a força de inércia em seus movimentos. As propostas podem ser observadas nesta primeira parte do desenvolvimento do projeto.

Palavras-chave: *Robô paralelo, robô delta, embarcado, cinemática.*

Abstract - With the modernization of industries and the advancement of technologies in several areas, precision, speed, and risk reduction must be a priority. With these ideals we noticed that the renewal of old technologies or the application of new ones has become essential to remain relevant in the market. With this demand in mind, a study was done to implement embedded and new technologies in a delta parallel robot, so that it is able to perform agile and precise tasks in a more reliable and responsive way. With these results as goals, several information about new technologies were raised, such as more powerful and faster microcontrollers, implementation of a human machine interface, making it simpler to program and visualize the information contained in the robot, and the calculation of kinematics in order to reduce the inertia force in its movements. The proposals can be observed in this first part of the project's development.

Keywords: *parallel robot, delta robot, embedded, kinematics.*

I. INTRODUÇÃO

Embora atualmente a estrutura predominante dos robôs manipuladores seja do tipo serial, a arquitetura paralela vem ganhando espaço gradativamente. Essa mudança se deve a sua ótima performance nas operações Pegar-e-Posicionar utilizada amplamente em diversos tipos de indústria [2].

O primeiro robô Delta foi patenteado em 1928, por Gwinnett este o fez para servir como uma plataforma para

um teatro. Futuramente em 1947 foi estabelecida os princípios básicos de uma estrutura cinemática em uma estrutura de malha fechada, esses princípios foram estabelecidos por Gough, e está sendo utilizada até os dias de hoje nos simuladores de voo. Outro mecanismo muito utilizado nos robôs paralelos foi o robô delta proposto por Clavel [1].

Com o grande avanço da robótica em todos os campos, e com o grande aumento da popularidade do robô com arquitetura paralela, este trabalho propõe a implementação da cinemática do robô delta, embarcado no *raspberrypi* utilizando a linguagem de programação *python*. Com a finalidade de interação com o robô uma IHM (Interface Homem-Máquina), será implementada.

Para atingir o que se propõe, este trabalho está organizado da seguinte forma: Seção 2 descreve a fundamentação teórica, a Seção 3 apresenta o detalhamento do projeto e a Seção 4 apresenta as considerações finais.

II. FUNDAMENTAÇÃO TEÓRICA

Para melhor desenvolvimento deste trabalho, primeiramente devemos conhecer mais a fundo o termo 'robô', o conceito de robô é bastante antigo e surgem divergências entre os pesquisadores quanto a sua origem, sabe-se que o termo surgiu primeiramente nos livros de ficção científica, sendo aplicado na ciência algum tempo depois, mesmo com todo o tempo passado, não se chegou em um consenso sobre o que é um robô.

Grande parte dos acadêmicos define robô como uma máquina e/ou dispositivos que possa ser utilizado para realizar o trabalho humano, para Ronald Arkin, professor regente do colégio de computação no instituto de tecnologia da Geórgia. "um robô é uma máquina capaz de extrair informações do ambiente e usar conhecimento sobre o mundo de modo a se mover com segurança e com um propósito".

Enquanto para Matt Mason, diretor do Instituto de Robótica da Universidade Carnegie Mellon, a definição de robô não é tão leviana, segundo ele, as pessoas tendem a associar robôs a figuras humanoides ou animais. O pesquisador acredita que o problema consiste na definição por trás de inteligência ou percepção e quão perto ou longe a máquina está de se tornar humano, "Se você tem uma atitude muito liberal, permissiva sobre estas coisas, é difícil excluir qualquer máquina do conceito de robô. Se você tiver uma atitude mais estrita, você pode acreditar que nenhum robô foi produzido ainda", conclui. [3]

A única verdade que podemos afirmar sem contradição é a modernização do trabalho, a substituição da mão de obra humana, pela máquina, se dá em cada vez mais setores, porém o setor industrial é onde predomina a utilização de robôs, dos mais variados tipos, podendo variar de modelos

utilizados para logística ou até mesmo braços robóticos na linha de produção. Conforme apresentado na Figura 1.



Figura 1. Robôs utilizados na logística.

Os mais utilizados, entretanto, são os braços robóticos, este são máquinas programadas para executar trabalhos e tarefas com eficiência, agilidade e extrema precisão, muito aplicados onde se tem a demanda de processos altamente repetitivos ou pesados por longos períodos de tempos.

Um braço robótico industrial, contém uma série de juntas, articulações e manipuladores que trabalham em conjunto juntas, articulações e manipuladores que trabalham coordenadamente para assemelhar o movimento e a funcionalidade de um braço humano. O braço robótico pode ser utilizado como um robô unitário, ou acoplado a um equipamento mais complexo e maior. Tendem a ser construídos com metal resistente e durável, e em sua grande maioria terá entre 4 e 6 juntas articuladas, conforme apresentado na Figura 2.

As juntas de um robô, determinam sua classificação, dependendo dos graus de liberdade que carecem de controle, o sistema de controle necessariamente precisa de um nível igual de autonomia para que execute a tarefa. O número de juntas em um robô determina quantos graus de liberdade ele possui. [4].

Os graus de liberdade determinando se os movimentos do robô podem ser executados em duas ou três dimensões, fazendo este um dos parâmetros mais cruciais no desenvolvimento, cada junta definirá um ou dois graus de liberdade, a quantidade de graus de liberdade será obtida através da soma de todos os outros GdL's (Grau de Liberdade).

Se a junta exercer movimento em um único eixo, ela será classificada com um grau de liberdade, este grau irá aumentar caso a junta possa exercer mais movimentos no eixo. Como pode ser visto na figura 2.

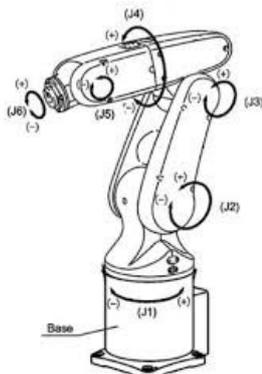


Figura 2. Robô com 6 Graus de Liberdade.

Na classificação dos robôs, também devemos levar em consideração a estrutura cinemática, está podendo ser aberta ou fechada. Os robôs fixos com arquitetura em série são amplamente usados nas mais diversas naturezas. Comumente os robôs de arquitetura serie, são mais utilizados principalmente devido a seu tamanho reduzido e sua fabricação consideravelmente mais simples.

Entretendo para trabalhos que exigem mais velocidade, precisão e uma capacidade de carga elevada, o robô de arquitetura paralela ganha destaque. A Figura 3 apresenta a diferença de sua arquitetura.

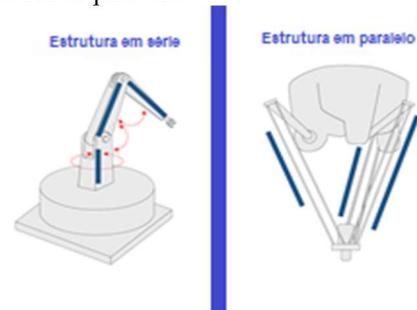


Figura 3. Ligação das juntas em série e paralelo.

A. Robôs fixos de Arquitetura em série

Os robôs fixos em arquitetura série, podem ser definidos como robôs de cinemática aberta, onde diversos elementos são conectados entre si, esta estrutura compõe a maioria dos manipuladores utilizados na indústria podendo ter diferentes estruturas básicas, sendo estas: cartesiano, cilíndrico, esférico, articulado e SCARA. A estrutura de um robô cartesiano pode ser vista na Figura 4.

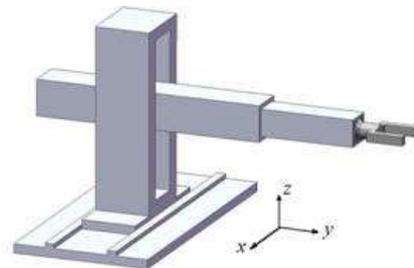


Figura 4. Estrutura de um robô cartesiano.

B. Robôs fixos de Arquitetura Paralela.

Para Merlet [1] um robô manipulador paralelo pode ser definido como um mecanismo de circuito fechado composto por uma ferramenta de trabalho com n graus de liberdade e uma base fixa ligadas entre si, com pelo menos duas cadeias de cinemática independente. Esta arquitetura de robôs possui movimentos angulares perfeitamente sincronizados, possibilitando a movimentação das barras, também chamadas de braços, para executar a tarefa designada

Os robôs fixos em arquitetura paralela, possuem uma cadeia cinemática fechada, isso implica que diversas cadeias abertas são utilizadas paralelamente para suportar uma base ou plataforma. Está estrutura pode ser vista na Figura 5 [5].

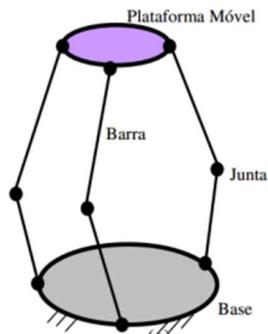


Figura 5. Estrutura paralela.

Nesta estrutura o grau de liberdade normalmente varia entre três a seis GdL. Os graus de liberdade definem as limitações do robô, para que o robô consiga posicionar sua ferramenta livremente no espaço ele precisa de seis GdL, três (três) ângulos de rotação para definir sua orientação e três graus de liberdade para definir sua localização. Robôs que apresentam mais de seis GdL são nomeados redundantes, enquanto os que apresentam menos de seis GdL são classificados como limitados [4].

Neste projeto o foco será no robô fixo de arquitetura paralela tipo Delta (Figura 6), sendo este um robô que possui três GdL (grau de liberdade) devido ao seu arranjo geométrico.



Figura 6. Robô Delta.

Os robôs deltas mais comuns apresentam 3 hastes idênticas (Figura 7), cada uma tendo uma arquitetura que pode variar de juntas rotativas (RRR) ou juntas prismáticas (RPR).

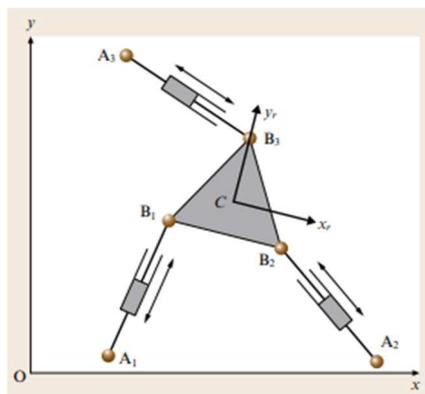


Figura 7. Estrutura com juntas prismáticas RPR.

Os robôs deltas se sobressaem quando a necessidade de uma tarefa que exige velocidade e precisão é solicitada, sendo a aplicação mais comum a de *pick-and-place*, podendo ser utilizado acoplado a outros mecanismos ou usado separadamente, esta aplicação é muito utilizada na indústria em todos os setores [1].

C. Raspberry Pi

Com o objetivo de uma melhor interação com o robô é necessária a implementação de uma IHM (Interface Homem-Máquina), uma aplicação em uma tela com o objetivo de melhorar e tornar mais eficaz a comunicação entre a máquina e o usuário.

Para esse objetivo, foi optado pela utilização do Microcomputador *raspberry Pi*, pela sua melhor capacidade de processamento e de realizar múltiplas tarefas, fazer cálculos mais intensos e controlar robôs mais complexos.

O modelo utilizado para executar a programação deste projeto é o *Raspberry pi 3* (Figura 8), que possui as especificações de processador Quad Core 1.2Ghz, 1 GB de RAM, conexão *Wifi* e *bluetooth*, saída HDMI, slot de cartão de memória MicroSD e a possibilidade de programação em *python*. Fazendo com que ele seja uma excelente opção para implementação de controle e instalação de uma IHM [6]



Figura 8. Microcomputador Raspberry Pi 3.

D. Linguagem de programação Python.

Para a programação do projeto será utilizada a linguagem de programação *python*.

Python é uma linguagem aberta, multiplataforma, de alto nível e com tipagem dinâmica e forte. Foi criada por Guido van Rossum em 1991, com a filosofia de priorizar a importância do esforço do programador sobre o esforço computacional.

O *python* possui uma sintaxe clara e objetiva com uma grande quantidade de bibliotecas padrão, módulos e frameworks desenvolvidos por terceiros, e sua programação é orientada a objetos, como a programação funcional e procedural. Também podendo ser utilizado como uma linguagem de extensão para aplicações que necessitam de uma interface programável (IHM) [7]

E. Cinemática.

Por definição a cinemática descreve e estuda o movimento dos corpos sem se importar com as causas do deslocamento. Para os robôs paralelos os problemas cinemáticos são geralmente resolvidos diretamente, A solução para esses problemas consiste na determinação dos parâmetros das hastes em relação a uma posição determinada na plataforma em um referencial fixo.

No robô delta o resultado da cinemática inversa se dá no cálculo entre os ângulos dos motores $\theta_1, \theta_2, \theta_3$, e a posição final do efetuador (x_0, y_0, z_0) . Criando uma função com as coordenadas desejadas $E_0(x_0, y_0, z_0)$ e que retornem os parâmetros dos ângulos dos motores, logo a função cinemática direta obtém-se: $E_0(x_0, y_0, z_0) = (\theta_1, \theta_2, \theta_3)$, também é necessário fazer a medição dos parâmetros fixos do robô delta, como comprimento das hastes e arestas, distância das juntas, a Figura 9 demonstra os parâmetros necessários para o cálculo da cinemática.

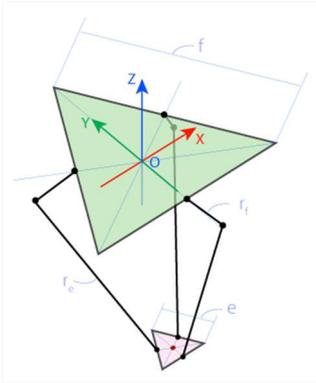


Figura 9. Parâmetros Robô Delta.

Na Figura 9 estão representados os parâmetros físicos do robô, esses valores são definidos no momento de sua construção. O valor da aresta da base é fixa em f , a aresta do efetuador é definida como e , o comprimento da primeira parte da haste até a primeira junta é indicada como rf e o comprimento da segunda parte da haste até o efetuador é definido como re [8].

Após os cálculos da cinemática inversa, saberemos os ângulos $\theta_1, \theta_2, \theta_3$, e com isso poderemos encontrar as coordenadas do efetuador $E_0(x_0, y_0, z_0)$. A geometria do robô permite que as juntas F_1J_1, F_2J_2 e F_3J_3 se movimentem apenas no plano YZ . Então com a posse do valor do ângulo θ , o cálculo das juntas pode ser realizado. Como demonstrado na Figura 10 [8].

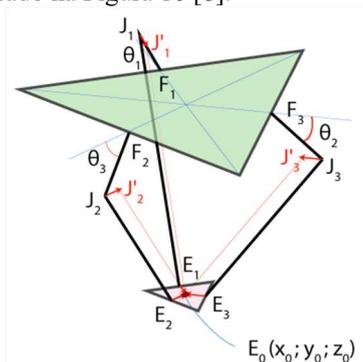


Figura 10. Cinemática Direta Robô Delta.

F. Motores

Os motores são partes fundamentais dos aparelhos modernos, os motores elétricos comumente utilizados em aplicações robóticas, como no caso do robô delta, tem a função de transformar energia elétrica em energia mecânica.

O motor de passo, entretanto, tem a capacidade de converter um sinal de entrada em um ângulo de rotação, com extrema precisão. Pois diferente dos demais motores, o motor de passo possui um número já fixado de polos magnéticos, o que determinam os passos do motor sem precisar de escovas.

Além de sua precisão, os motores de passo apresentam boa estabilidade no transporte de cargas, tanto em repouso quanto fora de seu ponto zero, sem erros significativos. Cada pulso recebido pelo motor de passo, corresponde a um pequeno passo de n graus, variando conforme os tipos de motores. A Figura 11, apresenta os motores utilizados neste projeto.

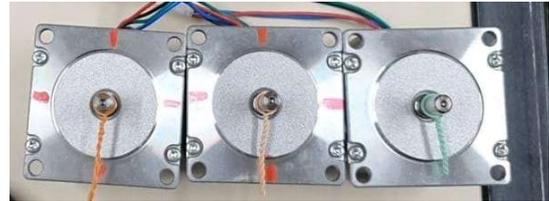


Figura 11. Motores de passo NEMA 23.

G. Sensores

Os sensores estão presentes em nosso cotidiano, e se fazem indispensáveis nos mais variáveis setores da automação. O papel do sensor é simples, porém importantíssimo, o sensor transforma as variáveis físicas como velocidade, temperatura ou campo magnético em variáveis simples de serem observadas e entendidas, dependendo do tipo de sensor escolhido, analógicos ou digitais, esses sinais convertidos podem ser sinais elétricos, sonoros ou sinais discretos, que podem ser interpretados por um controlador.



Figura 12. Sensor Reed Switch Festo.

Neste projeto foi optado pela utilização do sensor de proximidade magnética, o Reed Switch, que se baseiam na detecção de campos magnéticos para fazer a detecção do objeto em questão. Estes sensores, também podem ser chamados de sensor Hall, que é ativado quando um feixe de partículas carregadas, passa pelo campo magnético, fazendo o feixe ser defletido. A Figura 12, apresenta o sensor utilizado no projeto.

III. PROJETO

Com o surgimento de novas tecnologias e a modernização de tecnologias antigas, surge a necessidade de atualização. Tecnologias novas e atuais, tendem a realizar as tarefas propostas de maneira mais assertiva e rápida, e com o surgimento desta demanda, este projeto que visa a implementação de novas tecnologias em um robô delta defasado surgiu.

Como descrito anteriormente, para realizar a implementação da cinemática do robô delta, embarcando o raspberry pi com a linguagem python e a implementação de uma IHM, para que essa meta seja atingida foi iniciado o projeto.

Primeiramente foi realizada uma pesquisa bibliográfica sobre todos os assuntos abordados, cinemática, arquiteturas robóticas, microcontroladores e linguagens de programação. Após as decisões de componentes e estruturas que seriam utilizadas foram realizadas pesquisas mais direcionadas nos conteúdos. A Figura 13 apresenta a implementação do do projeto.

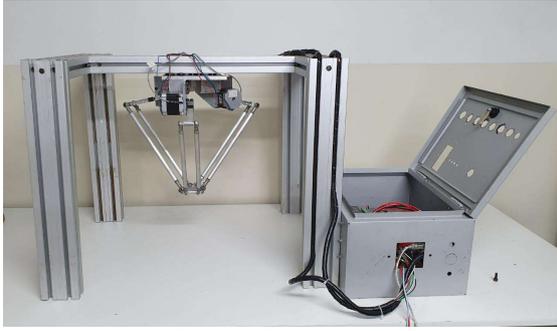


Figura 13. Implementação do robô.

A medição dos parâmetros físicos do robô e os cálculos de cinemática inversa já foram realizados, conforme apresentados abaixo:

$$EE1 = \frac{e}{2} \tan(30)$$

$$EE1 = \frac{e}{2\sqrt{3}} \text{ eq. 1}$$

$$E1 = \left(X0, Y0 - \frac{e}{2\sqrt{3}}, Z0 \right) \text{ eq. 2}$$

$$E1E'1 = X0 \text{ eq. 3}$$

$$E1E'1 = \sqrt{E1J1^2} - E1E1^2$$

$$E'1J1 = \sqrt{re^2} - x0^2$$

$$F1 = \left(0, -\frac{f}{2\sqrt{3}}, 0 \right)$$

$$\begin{cases} (Yj1 - Yf1)^2 + (Zj1 - Zf1)^2 = rf^2 \\ (Yj1 - YE'1)^2 + (Zj1 - ZE'1)^2 = re^2 - x0^2 \end{cases} \text{ eq. 4}$$

$$\begin{cases} \left(Yj1 + \frac{f}{2\sqrt{3}} \right)^2 + Zj1^2 = rf^2 \\ \left(Yj1 - Y0 + \frac{e}{2\sqrt{3}} \right)^2 + (Zj1 - Z0)^2 = re^2 - X0^2 \end{cases}$$

$$\theta1 = \text{Atan} \left(\frac{-Zj1}{YF1 - Yj1} \right)$$

Com o resultado do ângulo $\theta1$, ao girarmos o sistema nos sentidos 120° e -120° , devido a geometria do robô delta, obtendo a seguinte equação:

$$x2 = x0 \cos 120 + y0 \sin 120$$

$$y2 = y0 \cos 120 - x0 \sin 120$$

$$z2 = z0$$

$$x3 = x0 \cos 120 - y0 \sin 120$$

$$y3 = y0 \cos 120 + x0 \sin 120$$

$$z3 = z0$$

Realizados os cálculos manuais, foi dado início ao software que irá gerar os cálculos das cinemáticas do robô, a comunicação e controle dos motores e a leitura dos sensores. Nesta etapa, foi utilizada a plataforma *ThommyIDE*, disponível no *raspberry pi*, para a programação na

linguagem *python*. A Figura 14 apresenta o fluxograma do algoritmo desenvolvido.



Figura 14. Fluxograma do algoritmo.

Os resultados obtidos na programação foram os previstos, os valores dos ângulos e o controle dos motores se mostrou bastante satisfatório, o programa ainda conta com um compensador de velocidade dos motores, fazendo com que eles iniciem e terminem seus movimentos simultaneamente independente da distância dos ângulos que sejam percorridas, diminuindo muito a vibração e melhorando a precisão. Também foi possível perceber que a escolha do *raspberry pi* foi a correta, pois a utilização de *threading*, partições de núcleo, se fez de suma importância para que diversas funções pudessem ser executadas ao mesmo tempo, a programação utilizada neste projeto, está apresentada no Anexo I.

Os sensores *Reed Switch*, foram utilizados para determinar a posição inicial do robô ao ser inicializado, os motores são acionados pelo *raspberry pi*, e só irá desativar ao receber um sinal individual para cada motor, fazendo com que a posição inicial, possa ser variada de acordo com o desejo do usuário, basta manipular os sensores. E então só após finalizar esta rotina, a programação irá pedir para posições de XYZ para a movimentação do robô.

É importante ressaltar, que programação, está realizando movimentos através de coordenadas inseridas pelo usuário, (X, Y, Z), sem uma condição cíclica, fazendo com que o projeto não realize um ciclo de repetição que gera a operação de *Pick-and-Place* (Pega-e-Põe).

Ao finalizar o algoritmo que irá controlar o robô, foi definido que uma interface mais amigável deveria ser elaborada, ou seja, uma IHM. A plataforma utilizada para realização desta etapa é o *PySimpleGUI*, que irá simplificar a inserção de dados e a compreensão do funcionamento do projeto, até o momento da escrita deste projeto, a IHM está em processo de desenvolvimento.

IV. CONCLUSÃO

Com o estudo realizado foi possível compreender melhor as diversas arquiteturas de um robô, também possível analisar as limitações de robô delta e potenciais, sendo possível concluir que este mecanismo pode ser amplamente empregado ao meio industrial, podendo gerar grandes benefícios. As linguagens de programação estão extremamente presentes no nosso meio, o domínio delas no

meio da automação é primordial, o Python foi criado a mais de 20 anos e é usada em grandes empresas como a Nasa e o Google, os estudos realizados nessa plataforma de linguagem mostram como é indispensável o conhecimento dela para que se possa adentrar no mercado da automação de maneira eficaz, essa informação fica evidenciada neste projeto, uma vez que a programação foi a chave de todo o projeto, se mostrando indispensável para que a execução do mesmo se desse de forma satisfatória.

A automação preza por sistemas que sejam assertivos em otimizar o tempo e processo, com objetivo de melhorar todo o ambiente onde ela é implantada, Interface Homem Máquina ou simplesmente IHM, torna a utilização das máquinas muito mais simples, permitindo uma melhor acessibilidade para o usuário final, diminuindo seu tempo de aprendizagem e trazendo agilidade na execução.

Com o surgimento de uma demanda para equipamentos mais ágeis, precisos e versáteis, manifestou-se uma demanda para microcontroladores com as mesmas características, para realizarmos esse objetivo foi feito o estudo sobre diversas famílias e modelos de microcontroladores, visando selecionar o que poderia realizar a maior variadas de programas de leitura e escrita, assim como a comunicação com recursos externos para implementações de IHM.

Com os estudos feitos neste projeto, fica notável a importância deste avanço para qualquer ramo que possa seguir na automação, tendo que o robô delta se aplica perfeitamente em linhas automatizadas para separação e seleção em alta velocidade de operação

Para futuras melhorias no projeto, utilizar motores com maior força de torque, os motores de passo NEMA-23 atendem os requisitos para aprendizagem, porém executam de maneira lenta para aplicações reais, a integração de um efetuator final, como uma garra ou ventosas para aplicações de *pick-and-place*, e a um projeto de visão computacional para separação de objetos.

V. AGRADECIMENTOS

Primeiramente a meus pais, Adilson e Maria da Glória por sempre acreditarem e me incentivarem em todas as etapas da minha vida, e ao meu professor e amigo professor mestre Filipe Fraga Paula Silva, a quem devo a maior parte dos meus conhecimentos em todas as áreas da engenharia, ao meu orientador professor doutor Bruno Quirino de Oliveira a quem sempre esteve disponível para retirar todas as minhas dúvidas durante o desenvolvimento deste trabalho, a professora mestre Fabricia Neres, a quem sempre recorri em diversas etapas deste projeto. Por fim, agradeço, a quem sempre esteve me dando apoio e forcas em todas as etapas do projeto, obrigado Carina Duarte

VI. REFERÊNCIAS BIBLIOGRÁFICAS

[1] Merlet, Jean-Pierre & Gosselin, Clément. (2008). Parallel Mechanisms and Robots. 10.1007/978-3-540-30301-5_13.

[2] FINOTTI, Gilson. Cálculo explícito dos torques dos atuadores de um robô paralelo plano empregando o método de Kane. 2008. 199 f. Dissertação (Mestrado em Engenharia) – Escola Politécnica da Universidade de São

Paulo, 2008. Disponível em :. Acesso em: 12 nov. 2020, 17:21.

[3] OEI. De onde vêm os robôs? Disponível em: https://www.oei.es/historico/divulgacioncientifica/reportajes_183.htm#:~:text=O%20conceito%20de%20rob%C3%B4%2C%20como,do%20dramaturgo%20tcheco%20Karel%20Capek.&text=A%20express%C3%A3o%20rob%C3%B4%20deriva%20da,segundo%20alguns%20autores%2C%20trabalh%20escravo.. Acesso em: 18 nov. 2020.

[4] MUNHOZ, Igor Polezi. Robótica: unidade 1. Londrina — Pr: Educacional S.A., 2017.

[5] SILVA, Filipe Fraga Paula; BRAZ, Guilherme Reis; MAGALHÃES, José Ricardo R. R, LEITE, Rayan Teixeira. de. Robô Paralelo: Projeto e simulação de um robô delta, 2013.82 f. TCC (Graduação)- Curso de Engenharia de Controle e Automação, Pontifícia Universidade Católica de Goiás, Goiânia, 2013.

[6] RASPBERRY. Raspberry. 2020. Disponível em: <https://www.raspberrypi.org/>. Acesso em: 12 nov. 2020.

[7] PYTHON. Python. 2020. Disponível em: <https://www.python.org/>. Acesso em: 16 nov. 2020.

[8] MZAVATSKY. **Delta robot kinematics**. 2020. Disponível em: <http://hypertriangle.com/~alex/delta-robot-tutorial/>. Acesso em: 20 nov. 2020.

APENDICE 1: PROGRAMACAO ROBO PARALELO DELTA.

```

import math as maths
import numpy as np
import threading
import serial
import math
import RPi.GPIO as GPIO
from time import sleep

# Declaração de variáveis para os cálculos cinemáticos
x1=y1=z1=1
pulsos1 = pulsos2 = pulsos3 = 0
thetat1 = thetat2 = thetat3 = 0
theta = theta2= theta3 = 0
t1 = t2 = t3 = 0
x0=y0=z0=0
x02=y02=z02=0
x03=y03=z03=0

#Varivaveis Motor 1
pR1=pR2=0
R1=R2=0
#Variaveis Motor 2
pQ1=pQ2=0
Q1=Q2=0
#Variaveis Motor 3
N1=N2=0
pN1=pN2=0

Pd =0 #Posicao Desejada

vel1=vel2=vel3=0
Vp=0.01

e = 43.30;
f = 173.205;
re = 325.0;
rf = 120.0;

deg30 = maths.radians(30)
deg120 = maths.radians(120)

sqrt3 = math.sqrt(3.0)
pi = 3.141592653
sin120 = 0.866025403784
cos120 = -0.5
tan60 = sqrt3
sin30 = 0.5
tan30 = 1.0 / sqrt3

SENSOR2= 29
SENSOR1=22
SENSOR3=31
# Pino de direção do controlador
DIR = 10
DIR2= 18

DIR3= 15
# Motor de Passo pino do controlador
STEP = 8
STEP2= 16
STEP3= 13
# 0/1 usado para significar no sentido horário ou anti-
horário.
CW = 1
CCW = 0 #Anti

# Configuração do layout do pino no PI
GPIO.setmode(GPIO.BOARD)

# Estabelecer Pins no software
GPIO.setup(DIR, GPIO.OUT)
GPIO.setup(DIR2, GPIO.OUT)
GPIO.setup(DIR3, GPIO.OUT)
GPIO.setup(STEP, GPIO.OUT)
GPIO.setup(STEP2, GPIO.OUT)
GPIO.setup(STEP3, GPIO.OUT)
GPIO.setup(SENSOR1,
GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(SENSOR2,
GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(SENSOR3,
GPIO.IN,pull_up_down=GPIO.PUD_DOWN)

# Defina a primeira direção em que deseja que ele gire
GPIO.output(DIR,CW)
GPIO.output(DIR2,CW)
GPIO.output(DIR3,CW)

#try:

# while GPIO.input(SENSOR1) == False:

# GPIO.output(DIR,CCW)
# GPIO.output(STEP,GPIO.HIGH)
# sleep(.001)
# GPIO.output(STEP,GPIO.LOW)
# sleep(.001)
##### Motor2#####
# GPIO.output(DIR2,CCW)
# GPIO.output(STEP2,GPIO.HIGH)
# sleep(.001)
# GPIO.output(STEP2,GPIO.LOW)
# sleep(.001)
#####MOTOR 3#####
# GPIO.output(DIR3,CCW)
# GPIO.output(STEP3,GPIO.HIGH)
# sleep(.001)
# GPIO.output(STEP3,GPIO.LOW)
# sleep(.001)

#except KeyboardInterrupt:
# pass

def Pzmp1():

```

```

global DIR, STEP, SENSOR1

try:
    while GPIO.input(SENSOR1) == False:
        #print('Entrou no While M1',
GPIO.input(SENSOR1))
        GPIO.output(DIR,CW)
        GPIO.output(STEP,GPIO.HIGH)
        sleep(.01)
        GPIO.output(STEP,GPIO.LOW)
        sleep(.01)
    except KeyboardInterrupt:
        pass

def Pzmp2():
    global DIR2, STEP2, SENSOR2

    try:
        while GPIO.input(SENSOR2) == False:
            #print('Entrou no While M2',GPIO.input(SENSOR2))
            GPIO.output(DIR2,CW)
            GPIO.output(STEP2,GPIO.HIGH)
            sleep(.01)
            GPIO.output(STEP2,GPIO.LOW)
            sleep(.01)
        except KeyboardInterrupt:
            pass

def Pzmp3():
    global DIR3, STEP3, SENSOR3

    try:
        while GPIO.input(SENSOR3) == False:
            #print('Entrou no While M3',
GPIO.input(SENSOR3))
            GPIO.output(DIR3,CW)
            GPIO.output(STEP3,GPIO.HIGH)
            sleep(.01)
            GPIO.output(STEP3,GPIO.LOW)
            sleep(.01)

        except KeyboardInterrupt:
            pass

def DefXYZ():
    global x0,y0,z0,x02,y02,z02,x03,y03,z03

    sleep(0.5)

    try:
        x0 = float(input("Entre com valor de x: "))
    except ValueError:
        x0 = float
        print("x:", x0)

        x02=x0
        x03=x0

    try:
        y0 = float(input("Entre com valor y: "))
    except ValueError:
        y0 = float
        print("y:", y0)

        y02=y0
        y03=y0

    try:
        z0 = float(input("Ente com valor de z: "))
    except ValueError:
        z0 = float
        print("z:", z0)
        print()

def cine_inv():
    # Cinemática inversa
    # Função auxiliar para calcular o ângulo theta1 (Planos
YZ)
    global thetat1, pulsos1, x1, y1, z1,x0 ,y0,z0

    print('Valor de Y0: ', y0)

    #try:
        #x0 = float(input("> x: "))
        #y0 = float(input("> y: "))
        #z0 = float(input("> z: "))
    #except ValueError:
        #print("Erro ao inserir as váriaveis.")
        #return 1

    y1 = -f/2 * tan30 #-f/2*tan30
    y0 = y0 - e/2 * tan30

    a = (x0*x0 + y0*y0 + z0*z0 +rf*rf - re*re -
y1*y1)/(2*z0)
    b = (y1 - y0) / z0

    #Discriminante
    d = -(a+b*y1)(a+b*y1)+rf(b*b*rf+rf)

    if d < 0:
        return 1

    yj = (y1 - a*b - maths.sqrt(d)) / (b**2 + 1)
    zj= a+b*yj
    if yj > y1:
        defasagem = 180
    else:
        defasagem = 0

    theta = 180 * maths.atan(-zj/(y1 - yj))/pi+ defasagem

```

```

print("VALOR DE THETA:", theta)

pulsos1 = int((1600 * theta)/360)
#print("VALOR DE pulso:", pulsos1)

thetat1 = theta

def cine_inv_rotp():

    global thetat2, pulsos2, x1, y1, z1, x0, y0, z0,x02,y02

    #try:
    #x0 = float(input("> x: "))
    #y0 = float(input("> y: "))
    #z0 = float(input("> z: "))
    #except ValueError:
    #print("Erro ao inserir as váriaveis.")
    #return 1

    x2 = (x02 * (-0.5)) + (y02 * (0.866025403784))
    y2 = (y02 * (-0.5)) - (x02 * (0.866025403784))

    #x2 = 5.4903810568
    #y2 = -20.4903810568

    z2 = z0;

    #x0 = x2;
    #y0 = y2;
    #z0 = z2;

    y1 = -f/2 * tan30 #-f/2*tan30
    y2 = y2 - e/2 * tan30

    a = (x2*x2 + y2*y2 + z2*z2 +rf*rf - re*re -
y1*y1)/(2*z2)
    b = (y1-y2)/z2;

    #Discriminante
    d = -(a+b*y1)(a+b*y1)+rf(b*b*rf+rf)

    if d < 0:
        return 1

    yj = (y1 - a*b - maths.sqrt(d)) / (b*b + 1)
    zj= a+b*yj
    if yj > y1:
        defasagem = 180
    else:
        defasagem = 0

    theta2 = 180.0 * maths.atan(-zj/(y1 - yj))/pi+
defasagem
    print("VALOR DE THETA 2:", theta2)

    pulsos2 = int((1600 * theta2) / 360)
    #print("VALOR DE Pulso2:", pulsos2)
    thetat2 = theta2

def cine_inv_rotp2():
    global thetat3, pulsos3, x1, y1, z1,x0,y0,z0
    #try:
    #x0 = float(input("> x: "))
    #y0 = float(input("> y: "))
    #z0 = float(input("> z: "))
    #except ValueError:
    #print("Erro ao inserir as váriaveis.")
    #return 1

    x3 = x02*cos120 - y02*sin120;
    y3 = y02*cos120 + x02*sin120;
    z3 = z0

    x0 = x3;
    y0 = y3;
    z0 = z3;

    y1 = -f/2 * maths.tan(deg30) #-f/2*tan30
    y0 = y0 - e/2 * maths.tan(deg30)

    a = (x0*x0 + y0*y0 + z0*z0 +rf*rf - re*re -
y1*y1)/(2*z0)
    b = (y1 - y0) / z0

    #Discriminante
    d = -(a+b*y1)(a+b*y1)+rf(b*b*rf+rf)

    if d < 0:
        return 1

    yj = (y1 - a*b - maths.sqrt(d)) / (b*b + 1)
    zj= a+b*yj
    if yj > y1:
        defasagem = 180
    else:
        defasagem = 0

    theta3 = 180 * maths.atan(-zj/(y1 - yj))/maths.pi +
defasagem;
    print("VALOR DE THETA 3:", theta3)

    pulsos3 = int((1600*theta3)/360)
    #print("VALOR DE Pulso3:", pulsos3)
    thetat3 = theta3

def sentido():#Definir sentido Motor 1
    global thetat1, pulsos1,R1,R2,t1,pR1,pR2,Pd

    Pd=thetat1 #Pd=Posicao desejada, angulo obtido na
cinematica.

    if Pd > t1: #Sentido Anti-Horario

        R1 = Pd-t1
        t1=thetat1
        #print('Valor de angulo R1 no if: ', R1)
        #print('Valor de angulo t1 no if: ', t1)
        R2=0

```

```

elif Pd < t1: #Sentido Horario

    R2= t1-Pd
    t1=thetat1 #Armazena angulo anterior para comparar
    #print('Valor de angulo R2 no elif: ', R2)
    R1=0

pR1 = int((1600*R1)/360)
#print('Valor de pulso A-horario: ', pR1)

pR2 = int((1600*R2)/360)
#print('Valor de pulso horario: ', pR2)

#####
MOTOR 2
#####

def sentido2(): #Definir sentido Motor 2

    global thetat2,t2,Q1,Q2,pQ1,pQ2,Pd

    Pd=thetat2 #Pd=Posicao desejada, angulo obtido na
    cinematica.

    if Pd > t2: #Sentido Anti-Horario

        Q1 = Pd-t2
        t2=thetat2
        #print('Valor de angulo Q1 no if: ', Q1)
        #print('Valor de angulo t2 no if: ', t2)
        Q2=0

    elif Pd < t2: #Sentido Horario

        Q2= t2-Pd
        t2=thetat2 #Armazena angulo anterior para comparar
        #print('Valor de angulo Q2 no elif: ', Q2)
        Q1=0

    pQ1 = int((1600*Q1)/360)
    #print('Valor de A-horario: ', pQ1)

    pQ2 = int((1600*Q2)/360)
    print('Valor de pulso horario: ', pQ2)

def sentido3():#Definir sentido Motor 1

    global thetat3,t3,N1,N2,pN1,pN2,Pd

    Pd=thetat3 #Pd=Posicao desejada, angulo obtido na
    cinematica.

    if Pd > t3: #Sentido Anti-Horario

        N1 = Pd-t3
        t3=thetat3
        #print('Valor de angulo N1 no if: ', N1)
        N2=0

    elif Pd < t3: #Sentido Horario

        N2= t3-Pd
        t3=thetat3 #Armazena angulo anterior para comparar
        #print('Valor de angulo N2 no elif: ', N2)
        N1=0

    pN1 = int((1600*N1)/360)
    #print('Valor de pulso A-horario: ', pN1)

    pN2 = int((1600*N2)/360)
    #print('Valor de pulso horario: ', pN2)

    def vel_mp():

        global pR1, pR2, pQ1, pQ2,pN2,pN1,Pd, vel1, vel2,
        vel3,t1,t2,t3,Vp

        #pR1 e pR2= Pulsos1
        #pQ1 e pQ2= pulsos2
        #pN1 e pN2= pulsos3

        try:
            ##### SENTIDO
            ANTE HORARIO
            #####
            ####

            if pR1>pQ1 and pR1>pN1 and pQ1>pN1 and
            pR2==0 and pQ2==0 and pN2==0: #Quando M1 for maior
            que todos e M2 maior que M3 :

                vel1=(pN1*Vp)/pR1

                vel2=(pN1*Vp)/pQ1

                vel3=Vp

```

if $pR1 > pQ1$ and $pR1 > pN1$ and $pN1 > pQ1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M1 for maior que todos e M3 maior que M2

$$vel1 = (pQ1 * Vp) / pR1$$

$$vel3 = (pQ1 * Vp) / pN1$$

$$vel2 = Vp$$

if $pQ1 > pR1$ and $pQ1 > pN1$ and $pR1 > pN1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M2 for maior que todos e M1 maior que M3

$$vel2 = (pN1 * Vp) / pQ1$$

$$vel1 = (pN1 * Vp) / pR1$$

$$vel3 = Vp$$

if $pQ1 > pR1$ and $pQ1 > pN1$ and $pN1 > pR1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M2 for maior que todos e M3 maior que M1

$$vel2 = (pR1 * Vp) / pQ1$$

$$vel3 = (pR1 * Vp) / pN1$$

$$vel1 = Vp$$

if $pN1 > pR1$ and $pN1 > pQ1$ and $pR1 > pQ1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M3 for maior que todos e M1 maior que M2

$$vel3 = (pQ1 * Vp) / pN1$$

$$vel1 = (pQ1 * Vp) / pR1$$

$$vel2 = Vp$$

if $pN1 > pR1$ and $pN1 > pQ1$ and $pQ1 > pR1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M3 for maior que todos e M2 maior que M1

$$vel3 = (pR1 * Vp) / pN1$$

$$vel2 = (pR1 * Vp) / pQ1$$

$$vel1 = Vp$$

if $pR1 == pQ1$ and $pR1 > pN1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M1=M2 e (M1 e M2)>M3

$$vel1 = vel2 = (pN1 * Vp) / pR1$$

$$vel3 = Vp$$

if $pR1 == pQ1$ and $pN1 > pQ1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M1=M2 e M3>(M1 e M2)

$$vel3 = (pR1 * Vp) / pN1$$

$$vel1 = vel2 = Vp$$

if $pR1 == pN1$ and $pR1 > pQ1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M1=M3 e (M1 e M3)>M2

$$vel1 = vel3 = (pQ1 * Vp) / pN1$$

$$vel2 = Vp$$

if $pR1 == pN1$ and $pQ1 > pN1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M1=M3 e M2>(M1 e M3)

$$vel2 = (pR1 * Vp) / pQ1$$

$$vel1 = vel3 = Vp$$

if $pQ1 == pN1$ and $pN1 > pR1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M2=M3 e (M2 e M3)>M1

$$vel2 = vel3 = (pR1 * Vp) / pN1$$

$$vel1 = Vp$$

if $pQ1 == pN1$ and $pR1 > pQ1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando M2=M3 e M1>(M2 e M3)

$$vel1 = (pN1 * Vp) / pR1$$

$$vel2 = vel3 = Vp$$

if $pQ1 == pN1$ and $pR1 == pQ1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #Quando todos forem iguais

$$vel1 = vel2 = vel3 = Vp$$

```
#####
#####
#####
SENTIDO                                     HORARIO
#####
```

if $pR2 > pQ2$ and $pR2 > pN2$ and $pQ2 > pN2$ and $pR1 == 0$ and $pQ1 == 0$ and $pN1 == 0$: #Quando M1 for maior que todos e M2 maior que M3 :

$$vel1 = (pN2 * Vp) / pR2$$

$$vel2 = (pN2 * Vp) / pQ2$$

$$vel3 = Vp$$

if $pR2 > pQ2$ and $pR2 > pN2$ and $pN2 > pQ2$ and $pR1 == 0$ and $pQ1 == 0$ and $pN1 == 0$: #Quando M1 for maior que todos e M3 maior que M2

$$vel1 = (pQ2 * Vp) / pR2$$

$$vel3 = (pQ2 * Vp) / pN2$$

$$vel2 = Vp$$

if pQ2>pR2 and pQ2>pN2 and pR2>pN2 and pR1==0 and pQ1==0 and pN1==0:#Quando M2 for maior que todos e M1 maior que M3

$$vel2=(pN2*Vp)/pQ2$$

$$vel1=(pN2*Vp)/pR2$$

$$vel3=Vp$$

if pQ2>pR2 and pQ2>pN2 and pN2>pR2 and pR1==0 and pQ1==0 and pN1==0:#Quando M2 for maior que todos e M3 maior que M1

$$vel2=(pR2*Vp)/pQ2$$

$$vel3=(pR2*Vp)/pN2$$

$$vel1=Vp$$

if pN2>pR2 and pN2>pQ2 and pR2>pQ2 and pR1==0 and pQ1==0 and pN1==0: #Quando M3 for maior que todos e M1 maior que M2

$$vel3=(pQ2*Vp)/pN2$$

$$vel1=(pQ2*Vp)/pR2$$

$$vel2=Vp$$

if pN2>pR2 and pN2>pQ2 and pQ2>pR2 and pR1==0 and pQ1==0 and pN1==0: #Quando M3 for maior que todos e M2 maior que M1

$$vel3=(pR2*Vp)/pN2$$

$$vel2=(pR2*Vp)/pQ2$$

$$vel1=Vp$$

if pR2==pQ2 and pR2>pN2 and pR1==0 and pQ1==0 and pN1==0: #Quando M1=M2 e (M1 e M2)>M3

$$vel1=vel2=(pN2*Vp)/pR2$$

$$vel3=Vp$$

if pR2==pQ2 and pN2>pQ2 and pR1==0 and pQ1==0 and pN1==0: #Quando M1=M2 e M3>(M1 e M2)

$$vel3=(pR2*Vp)/pN2$$

$$vel1=vel2=Vp$$

if pR2==pN2 and pR2>pQ2 and pR1==0 and pQ1==0 and pN1==0:#Quando M1=M3 e (M1 e M3)> M2

$$vel1=vel3=(pQ2*Vp)/pN2$$

$$vel2=Vp$$

if pR2==pN2 and pQ2>pN2 and pR1==0 and pQ1==0 and pN1==0:#Quando M1=M3 e M2>(M1 e M3)

$$vel2=(pR2*Vp)/pQ2$$

$$vel1=vel3=Vp$$

if pQ2==pN2 and pN2>pR2 and pR1==0 and pQ1==0 and pN1==0:#Quando M2=M3 e (M2 e M3)>M1

$$vel2=vel3=(pR2*Vp)/pN2$$

$$vel1=Vp$$

if pQ2==pN2 and pR2>pQ2 and pR1==0 and pQ1==0 and pN1==0:#Quando M2=M3 e M1>(M2 e M3)

$$vel1=(pN2*Vp)/pR2$$

$$vel2=vel3=Vp$$

if pQ2==pN2 and pR2==pQ2 and pR1==0 and pQ1==0 and pN1==0:#Quando todos forem iguais

$$vel1=vel2=vel3=Vp$$

FUNCOES
ASSINCRONAS
#####

M1 >
#####

if pR1>pQ1 and pR1>pN2 and pQ1>pN2 and pR2==0 and pQ2==0 and pN1==0: #Quando M1A> M2A e M3H e M2A>M3H

$$vel1=(pN2*Vp)/pR1$$

$$vel2=(pN2*Vp)/pQ1$$

$$vel3=Vp$$

if pR1>pQ1 and pR1>pN2 and pN2>pQ1 and pR2==0 and pQ2==0 and pN1==0: #Quando M1A>M2A e M3H e M3H>M2A

$$vel1=(pQ1*Vp)/pR1$$

$$vel3=(pQ1*Vp)/pN2$$

$$vel2=Vp$$

if pR1>pQ2 and pR1>pN1 and pQ2>pN1 and pR2==0 and pQ1==0 and pN2==0:#Quando M1A>M2H e M3A e M2H>M3A

$$vel1=(pN1*Vp)/pR1$$

$$vel2=(pN1*Vp)/pQ2$$

$$vel3=Vp$$

if $pR1 > pQ2$ and $pR1 > pN1$ and $pN1 > pQ2$ and $pR2 == 0$ and $pQ1 == 0$ and $pN2 == 0$: #Quando $M1A > M2H$ e $M3A$ e $M3A > M2H$

$$vel1 = (pQ2 * Vp) / pR1$$

$$vel3 = (pQ2 * Vp) / pN1$$

$$vel2 = Vp$$

if $pR1 > pQ2$ and $pR1 > pN2$ and $pQ2 > pN2$ and $pR2 == 0$ and $pQ1 == 0$ and $pN1 == 0$: #Quando $M1A > M2H$ e $M3H$ e $M2H > M3H$

$$vel1 = (pN2 * Vp) / pR1$$

$$vel2 = (pN2 * Vp) / pQ2$$

$$vel3 = Vp$$

if $pR1 > pQ2$ and $pR1 > pN2$ and $pN2 > pQ2$ and $pR2 == 0$ and $pQ1 == 0$ and $pN1 == 0$:

$$vel1 = (pQ2 * Vp) / pR1$$

$$vel3 = (pQ2 * Vp) / pN2$$

$$vel2 = Vp$$

if $pR2 > pQ2$ and $pR2 > pN1$ and $pQ2 > pN1$ and $pR1 == 0$ and $pQ1 == 0$ and $pN2 == 0$:

$$vel1 = (pN1 * Vp) / pR2$$

$$vel2 = (pN1 * Vp) / pQ2$$

$$vel3 = Vp$$

if $pR2 > pQ2$ and $pR2 > pN1$ and $pN1 > pQ2$ and $pR1 == 0$ and $pQ1 == 0$ and $pN2 == 0$: #M1H > M2H e M3A // M3A > M2H

$$vel1 = (pQ2 * Vp) / pR2$$

$$vel3 = (pQ2 * Vp) / pN1$$

$$vel2 = Vp$$

if $pR2 > pQ1$ and $pR2 > pN1$ and $pQ1 > pN1$ and $pR1 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #M1H > M2A > M3A

$$vel1 = (pN1 * Vp) / pR2$$

$$vel2 = (pN1 * Vp) / pQ1$$

$$vel3 = Vp$$

if $pR2 > pQ1$ and $pR2 > pN1$ and $pN1 > pQ1$ and $pR1 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #M1H > M3A > M2A

$$vel1 = (pQ1 * Vp) / pR2$$

$$vel3 = (pQ1 * Vp) / pN1$$

$$vel2 = Vp$$

if $pR2 > pQ1$ and $pR2 > pN2$ and $pQ1 > pN2$ and $pR1 == 0$ and $pQ2 == 0$ and $pN1 == 0$: #M1H > M2A > M3H

$$vel1 = (pN2 * Vp) / pR2$$

$$vel2 = (pN2 * Vp) / pQ1$$

$$vel3 = Vp$$

if $pR2 > pQ1$ and $pR2 > pN2$ and $pN2 > pQ1$ and $pR1 == 0$ and $pQ2 == 0$ and $pN1 == 0$: #M1H > M3A > M2H

$$vel1 = (pQ1 * Vp) / pR2$$

$$vel3 = (pQ1 * Vp) / pN2$$

$$vel2 = Vp$$

M2 >
#####

if $pQ1 > pR1$ and $pQ1 > pN2$ and $pR1 > pN2$ and $pR2 == 0$ and $pQ2 == 0$ and $pN1 == 0$: #M2A > M1A > M3H

$$vel2 = (pN2 * Vp) / pQ1$$

$$vel1 = (pN2 * Vp) / pR1$$

$$vel3 = Vp$$

if $pQ1 > pR1$ and $pQ1 > pN2$ and $pN2 > pR1$ and $pR2 == 0$ and $pQ2 == 0$ and $pN1 == 0$: #M2A > M3H > M1A

$$vel2 = (pR1 * Vp) / pQ1$$

$$vel3 = (pR1 * Vp) / pN2$$

$$vel1 = Vp$$

if $pQ1 > pR2$ and $pQ1 > pN2$ and $pR2 > pN2$ and $pR1 == 0$ and $pQ2 == 0$ and $pN1 == 0$: #M2A > M1H > M3H

$$vel2 = (pN2 * Vp) / pQ1$$

$$vel1 = (pN2 * Vp) / pR2$$

$$vel3 = Vp$$

if $pQ1 > pR2$ and $pQ1 > pN2$ and $pN2 > pR2$ and $pR1 == 0$ and $pQ2 == 0$ and $pN1 == 0$: #M2A > M3H > M1H

$$vel2 = (pR2 * Vp) / pQ1$$

$$vel3 = (pR2 * Vp) / pN2$$

$$vel1 = Vp$$

if $pQ1 > pR2$ and $pQ1 > pN1$ and $pR2 > pN1$ and $pR1 == 0$ and $pQ2 == 0$ and $pN2 == 0$: #M2A > M1H > M3A

$$vel2 = (pN1 * Vp) / pQ1$$

$vel1=(pN1*Vp)/pR2$
 $vel3=Vp$
 if $pQ1>pR2$ and $pQ1>pN1$ and $pN1>pR2$ and $pR1==0$ and $pQ2==0$ and $pN2==0$: #M2A>M3A>M1H

 $vel2=(pR2*Vp)/pQ1$
 $vel3=(pR2*Vp)/pN1$
 $vel1=Vp$
 if $pQ2>pR2$ and $pQ2>pN1$ and $pR2>pN1$ and $pR1==0$ and $pQ1==0$ and $pN2==0$: #M2H>M1H>M3A

 $vel2=(pN1*Vp)/pQ2$
 $vel1=(pN1*Vp)/pR2$
 $vel3=Vp$

 if $pQ2>pR2$ and $pQ2>pN1$ and $pN1>pR2$ and $pR1==0$ and $pQ1==0$ and $pN2==0$: #M2H>M3A>M1H

 $vel2=(pR2*Vp)/pQ2$
 $vel3=(pR2*Vp)/pN1$
 $vel1=Vp$

 if $pQ2>pR1$ and $pQ2>pN1$ and $pR1>pN1$ and $pR2==0$ and $pQ1==0$ and $pN2==0$: #M2H>M1A>M3A

 $vel2=(pN1*Vp)/pQ2$
 $vel1=(pN1*Vp)/pR1$
 $vel3=Vp$

 if $pQ2>pR1$ and $pQ2>pN1$ and $pN1>pR1$ and $pR2==0$ and $pQ1==0$ and $pN2==0$: #M2H>M3A>M1A

 $vel2=(pR1*Vp)/pQ2$
 $vel3=(pR1*Vp)/pN1$
 $vel1=Vp$

 if $pQ2>pR1$ and $pQ2>pN2$ and $pR1>pN2$ and $pR2==0$ and $pQ1==0$ and $pN1==0$: #M2H>M1A>M3H

 $vel2=(pN2*Vp)/pQ2$
 $vel1=(pN2*Vp)/pR1$
 $vel3=Vp$

if $pQ2>pR1$ and $pQ2>pN2$ and $pN2>pR1$ and $pR2==0$ and $pQ1==0$ and $pN1==0$: #M2H>M3H>M1A

 $vel2=(pR1*Vp)/pQ2$
 $vel3=(pR1*Vp)/pN2$
 $vel1=Vp$

 ##### M3
 #####
 if $pN1>pR1$ and $pN1>pQ2$ and $pR1>pQ2$ and $pR2==0$ and $pQ1==0$ and $pN2==0$: #M3A>M1A>M2H

 $vel3=(pQ2*Vp)/pN1$
 $vel1=(pQ2*Vp)/pR1$
 $vel2=Vp$

 if $pN1>pR1$ and $pN1>pQ2$ and $pQ2>pR1$ and $pR2==0$ and $pQ1==0$ and $pN2==0$: #M3A>M2H>M1A

 $vel3=(pR1*Vp)/pN1$
 $vel2=(pR1*Vp)/pQ2$
 $vel1=Vp$

 if $pN1>pR2$ and $pN1>pQ2$ and $pR2>pQ2$ and $pR1==0$ and $pQ1==0$ and $pN2==0$: #M3A>M1H>M2H

 $vel3=(pQ2*Vp)/pN1$
 $vel1=(pQ2*Vp)/pR2$
 $vel2=Vp$

 if $pN1>pR2$ and $pN1>pQ1$ and $pR2>pQ1$ and $pR1==0$ and $pQ2==0$ and $pN2==0$: #M3A>M1H>M2A

 $vel3=(pQ1*Vp)/pN1$
 $vel1=(pQ1*Vp)/pR2$
 $vel2=Vp$

 if $pN1>pR2$ and $pN1>pQ1$ and $pQ1>pR2$ and $pR1==0$ and $pQ2==0$ and $pN2==0$: #M3A>M2A>M1H

 $vel3=(pR2*Vp)/pN1$
 $vel2=(pR2*Vp)/pQ1$

```

    vel1=Vp
    if pN2>pR2 and pN2>pQ1 and pR2>pQ1 and
    pR1==0 and pQ2==0 and pN1==0: #M3H>M1H>M2A
        vel3=(pQ1*Vp)/pN2
        vel1=(pQ1*Vp)/pR2
        vel2=Vp
    if pN2>pR2 and pN2>pQ1 and pQ1>pR2 and
    pR1==0 and pQ2==0 and pN1==0: #M3H>M2A>M1H
        vel3=(pR2*Vp)/pN2
        vel2=(pR2*Vp)/pQ1
        vel1=Vp
    if pN2>pR1 and pN2>pQ1 and pR1>pQ1 and
    pR2==0 and pQ2==0 and pN1==0: #M3H>M1A>M2A
        vel3=(pQ1*Vp)/pN2
        vel1=(pQ1*Vp)/pR1
        vel2=Vp
    if pN2>pR1 and pN2>pQ1 and pQ1>pR1 and
    pR2==0 and pQ2==0 and pN1==0: #M3H>M2A>M1A
        vel3=(pR1*Vp)/pN2
        vel2=(pR1*Vp)/pQ1
        vel1=Vp
    if pN2>pR1 and pN2>pQ2 and pR1>pQ2 and
    pR2==0 and pQ1==0 and pN1==0: #M3A>M1A>M2H
        vel3=(pQ2*Vp)/pN2
        vel1=(pQ2*Vp)/pR1
        vel2=Vp
    if pN2>pR1 and pN2>pQ2 and pQ2>pR1 and
    pR2==0 and pQ1==0 and pN1==0: #M3A>M2H>M1A
        vel3=(pR1*Vp)/pN2
        vel2=(pR1*Vp)/pQ2
        vel1=Vp
#####igualdades#####
#####

    if pR1==pQ1 and pR1>pN2 and pR2==0 and
    pQ2==0 and pN1==0: #M1A=M2A > M3H
        vel3=Vp
        vel1=vel2=(pN2*Vp)/pR1
        if pR1==pQ1 and pN2>pQ1 and pR2==0 and
        pQ2==0 and pN1==0: #M1A=M2A < M3H
            vel3=(pQ2*Vp)/pN2
            vel1=vel2=Vp
        if pR2==pQ2 and pR2>pN1 and pR1==0 and
        pQ1==0 and pN2==0: #M1H=M2H>M3A
            vel1=vel2=(pN1*Vp)/pR2
            vel3=Vp
        if pR2==pQ2 and pN1>pR2 and pR1==0 and
        pQ1==0 and pN2==0: #M1H=M2H<M3A
            vel3=(pR2*Vp)/pN2
            vel1=vel2=Vp
        if pR1==pQ2 and pQ2>pN1 and pR2==0 and
        pQ1==0 and pN2==0: #M1A=M2H <M3A
            vel1=vel2=(pN1*Vp)/pR1
            vel3=Vp
        if pR1==pQ2 and pN1>pR1 and pR2==0 and
        pQ1==0 and pN2==0: #M1A=M2H <M3A
            vel1=vel2= Vp
            vel3=Vp=(pR1*Vp)/pN1
        if pR2==pQ1 and pQ1>pN1 and pR1==0 and
        pQ1==0 and pN2==0: #M1H=M2A > M3A
            vel1=vel2=(pN1*Vp)/pR1
            vel3=Vp
        if pR2==pQ1 and pN1>pQ1 and pR1==0 and
        pQ2==0 and pN2==0: #M1H=M2A < M3A
            print('caiu nessa')
            vel1=vel2= Vp
            vel3=(pR2*Vp)/pN1
        if pR1==pQ2 and pQ2>pN2 and pR2==0 and
        pQ1==0 and pN1==0: #M1A=M2H>M3H
            vel1=vel2=(pN2*Vp)/pR1
            vel3=Vp
        if pR1==pQ2 and pN2>pQ2 and pR2==0 and
        pQ1==0 and pN1==0: #M1A=M2H>M3H

```

```

vel1=vel2=Vp
vel3=(pQ2*Vp)/pN2

if pR2==pQ1 and pR2>pN2 and pR1==0 and
pQ2==0 and pN1==0: #M2A=M1H >M3H

vel1=vel2= (pN2*Vp)/pR2

vel3=Vp

if pR2==pQ1 and pN2>pR2 and pR1==0 and
pQ2==0 and pN1==0: #M2A=M1H <M3H

vel1=vel2= Vp

vel3=(pR2*Vp)/pN2

##### M2=M3
#####

if pQ1==pN1 and pQ1>pR2 and pR1==0 and
pQ2==0 and pN2==0: #M2A=M3A > M1H

vel2=vel3=(pR2*Vp)/pN1

vel1= Vp

if pQ1==pN1 and pR2>pQ1 and pR1==0 and
pQ2==0 and pN2==0: #M2A=M3A < M1H

vel2=vel3=Vp

vel1=(pQ1*Vp)/pR2

if pQ2==pN2 and pQ2>pR1 and pR2==0 and
pQ1==0 and pN1==0: #M2H=M3H > M1A

vel2=vel3=(pR1*Vp)/pQ2

vel1=Vp

if pQ2==pN2 and pR1>pQ2 and pR2==0 and
pQ1==0 and pN1==0: #M2H=M3H < M1A

vel2=vel3=Vp

vel1=(pQ2*Vp)/pR1

if pQ2==pN1 and pQ2>pR1 and pR2==0 and
pQ1==0 and pN2==0: #M2H=M3A > M1A

vel2=vel3=(pR1*Vp)/pQ2

vel1=Vp

if pQ2==pN1 and pR1>pQ2 and pR2==0 and
pQ1==0 and pN2==0: #M2H=M3A < M1A

vel2=vel3= Vp

vel1= (pQ2*VP)/pR1

if pQ2==pN1 and pQ2>pR2 and pR1==0 and
pQ1==0 and pN2==0: #M2H=M3A > M1H

vel2=vel3=(pR2*Vp)/pQ2

vel1=Vp

if pQ2==pN1 and pR2>pQ2 and pR1==0 and
pQ1==0 and pN2==0: #M2H=M3A < M1H

vel2=vel3=Vp

vel1=(pQ2*Vp)/pR2

if pQ1==pN2 and pN2>pR1 and pR2==0 and
pQ2==0 and pN1==0: #M2A=M3H > M1A

vel2=vel3=(pR1*Vp)/pQ1

vel1= Vp

if pQ1==pN2 and pR1>pN2 and pR2==0 and
pQ2==0 and pN1==0: #M2A=M3H > M1A

vel2=vel3= Vp

vel1=(pN2*Vp)/pR1

if pQ1==pN2 and pQ1>pR2 and pR1==0 and
pQ2==0 and pN1==0: #M2A=M3H > M1H

vel2=vel3=(pR2*Vp)/pQ1

vel1=Vp

if pQ1==pN2 and pR2>pQ1 and pR1==0 and
pQ2==0 and pN1==0: #M2A=M3H < M1H

vel2=vel3=(pQ1*Vp)/pR2

vel1= Vp

#####
M3=M1 #####

if pR1==pN1 and pR1>pQ2 and pR2==0 and
pQ1==0 and pN2==0: #M1A=M3A>M2H

vel1=vel3=(pQ2*Vp)/pR1

vel2=Vp

if pR1==pN1 and pR1<pQ2 and pR2==0 and
pQ1==0 and pN2==0: #M1A=M3A<M2H

vel1=vel3=Vp

vel2=(pR1*Vp)/pQ2

```

if pR2==pN2 and pR2>pQ1 and pR1==0 and pQ2==0 and pN1==0: #M1H=M3H>M2A

$$vel1=vel3=(pQ1*Vp)/pR2$$

$$vel2=Vp$$

if pR2==pN2 and pR2<pQ1 and pR1==0 and pQ2==0 and pN1==0: #M1H=M3H<M2A

$$vel1=vel3=Vp$$

$$vel2=(pR2*Vp)/pQ1$$

if pR2==pN1 and pR2>pQ1 and pR1==0 and pQ2==0 and pN2==0: #M1H=M3A>M2A

$$vel1=vel3=(pQ1*Vp)/pR2$$

$$vel2=Vp$$

if pR2==pN1 and pR2<pQ1 and pR1==0 and pQ2==0 and pN2==0: #M1H=M3A<M2A

$$vel1=vel3=Vp$$

$$vel2=(pR2*Vp)/pQ1$$

if pR2==pN1 and pR2>pQ2 and pR1==0 and pQ1==0 and pN2==0: #M1H=M3A>M2H

$$vel1=vel3=(pQ2*Vp)/pR2$$

$$vel2=Vp$$

if pR2==pN1 and pR2<pQ2 and pR1==0 and pQ1==0 and pN2==0: #M1H=M3A<M2H

$$vel1=vel3=Vp$$

$$vel2=(pR2*Vp)/pQ2$$

if pR1==pN2 and pR1>pQ1 and pR2==0 and pQ2==0 and pN1==0: #M1A=M3H>M2A

$$vel1=vel3=(pQ1*Vp)/pR1$$

$$vel2=Vp$$

if pR1==pN2 and pQ1>pR1 and pR2==0 and pQ2==0 and pN1==0: #M1A=M3H<M2A

$$vel1=vel3=Vp$$

$$vel2=(pR1*Vp)/pQ1$$

if pR1==pN2 and pR1>pQ2 and pR2==0 and pQ1==0 and pN1==0: #M1A=M3H>M2H

$$vel1=vel3=(pQ2*Vp)/pR1$$

$$vel2=Vp$$

if pR1==pN2 and pQ2>pR1 and pR2==0 and pQ1==0 and pN1==0: #M1A=M3H<M2H

$$vel1=vel3=Vp$$

$$vel2=(pR1*Vp)/pQ2$$

IGUALDADE TRIPLA #####

if pR1==pQ1 and pQ1==pN2 and pR2==0 and pQ2==0 and pN1==0: #M1A=M2A=M3H

$$vel1=vel2=vel3=Vp$$

if pR1==pQ2 and pQ2==pN2 and pR2==0 and pQ1==0 and pN1==0: #M1A=M2H=M3H

$$vel1=vel2=vel3=Vp$$

if pR1==pQ2 and pQ2==pN1 and pR2==0 and pQ1==0 and pN2==0: #M1A=M2H=M3A

$$vel1=vel2=vel3=Vp$$

if pR2==pQ2 and pQ2==pN1 and pR1==0 and pQ1==0 and pN2==0: #M1H=M2H=M3A

$$vel1=vel2=vel3=Vp$$

if pR2==pQ1 and pQ1==pN1 and pR1==0 and pQ2==0 and pN2==0: #M1H=M2A=M3A

$$vel1=vel2=vel3=Vp$$

if pR2==pQ1 and pQ1==pN2 and pR1==0 and pQ2==0 and pN1==0: #M1H=M2A=M3H

$$vel1=vel2=vel3=Vp$$

if pQ2>pN2 and pN2>pR1 and pR2==0 and pQ1==0 and pN1==0:

$$vel2=(pR1*Vp)/pQ2$$

$$vel3=(pR1*Vp)/pN2$$

$$vel1=Vp$$

else:

$$vel1=vel2=vel3=Vp$$

print('sem condicao definida')

```

except KeyboardInterrupt:
    print("cleanup")
    GPIO.cleanup()

    #print('executou a certa')

def mp():

    global pR1, pR2, vel1

    print('valor de Vel1', vel1)

    # Pino de direção do controlador
    DIR = 10

    # Motor de Passo pino do controlador
    STEP = 8

    # 0/1 usado para significar no sentido horário ou anti-
    horário.
    CW = 1
    CCW = 0 #Anti

    # Configuração do layout do pino no PI
    GPIO.setmode(GPIO.BOARD)

    # Estabelecer Pins no software
    GPIO.setup(DIR, GPIO.OUT)
    GPIO.setup(STEP, GPIO.OUT)
    ####Motor2####

    # Defina a primeira direção em que deseja que ele gire
    GPIO.output(DIR, CW)
    #

    try:
        # Manter Funcionando.
        #while True:

            #""Mudar de direção: Mudar de direção requer
            tempo para mudar. O
            # o tempo é ditado pelo motor de passo e pelo
            controlador. ""
            sleep(0)
            # Estabeleça a direção que deseja seguir
            GPIO.output(DIR,CW)

            # Define o Numero de Passos pro sentido horario
            for x in range(pR2):

                # Defina um enrolamento da bobina para alto
                GPIO.output(STEP,GPIO.HIGH)
                sleep(vel1) # Ajusta a Velocidade do Motor de
                Passo

                # Defina um enrolamento da bobina para baixo
                GPIO.output(STEP,GPIO.LOW)
                sleep(vel1) # Ajusta a Velocidade do Motor de
                Passo

                #""Mudar de direção: Mudar de direção requer
                tempo para mudar. O
                # o tempo é ditado pelo motor de passo e pelo
                controlador. ""
                sleep(0)
                GPIO.output(DIR,CCW)

                for x in range(pR1):
                    GPIO.output(STEP,GPIO.HIGH)
                    sleep(vel1)
                    GPIO.output(STEP,GPIO.LOW)
                    sleep(vel1)

            # Assim que terminar, limpa os IO's
            except KeyboardInterrupt:
                print("cleanup")
                GPIO.cleanup()

def mp2():

    global pQ1, pQ2,vel2

    print('Valor de vel2: ', vel2)

    # Pino de direção do controlador

    DIR2 = 18
    # Motor de Passo pino do controlador

    STEP2 = 16
    # 0/1 usado para significar no sentido horário ou anti-
    horário.
    CW = 1
    CCW = 0 #Anti

    # Configuração do layout do pino no PI
    GPIO.setmode(GPIO.BOARD)

    # Estabelecer Pins no software
    ####Motor2####
    GPIO.setup(DIR2, GPIO.OUT)
    GPIO.setup(STEP2, GPIO.OUT)

    # Defina a primeira direção em que deseja que ele gire

    GPIO.output(DIR2, CW)

```

```

try:
# Manter Funcionando.
#while True:

#""Mudar de direção: Mudar de direção requer
tempo para mudar. O
# o tempo é ditado pelo motor de passo e pelo
controlador. ""
sleep(0)
# Estabeleça a direção que deseja seguir

GPIO.output(DIR2,CW)

# Ajusta a Velocidade do Motor de Passo

#####
#####
# Define o Numero de Passos pro sentido
horario
for x in range(pQ2):

# Defina um enrolamento da bobina para alto
GPIO.output(STEP2,GPIO.HIGH)
sleep(vel2) # Ajusta a Velocidade do Motor de
Passo
# Defina um enrolamento da bobina para baixo
GPIO.output(STEP2,GPIO.LOW)
sleep(vel2) # Ajusta a Velocidade do Motor de
Passo

#""Mudar de direção: Mudar de direção requer
tempo para mudar. O
# o tempo é ditado pelo motor de passo e pelo
controlador. ""
sleep(0)

GPIO.output(DIR2,CCW)

for x in range(pQ1):
GPIO.output(STEP2,GPIO.HIGH)
sleep(vel2)
GPIO.output(STEP2,GPIO.LOW)
sleep(vel2)

# Assim que terminar, limpa os IO's
except KeyboardInterrupt:
print("cleanup")
GPIO.cleanup()

def mp3():

global pN1, pN2,vel3

print('valor de vel3: ',vel3)

# Pino de direção do controlador
DIR3 = 15

```

```

# Motor de Passo pino do controlador
STEP3 = 13

# 0/1 usado para significar no sentido horário ou anti-
horário.
CW = 1
CCW = 0 #Anti

# Configuração do layout do pino no PI
GPIO.setmode(GPIO.BOARD)

# Estabelecer Pins no software
GPIO.setup(DIR3, GPIO.OUT)
GPIO.setup(STEP3, GPIO.OUT)
###Motor2####

# Defina a primeira direção em que deseja que ele gire
GPIO.output(DIR3, CW)

try:
# Manter Funcionando.
#while True:

#""Mudar de direção: Mudar de direção requer
tempo para mudar. O
# o tempo é ditado pelo motor de passo e pelo
controlador. ""
sleep(0)
# Estabeleça a direção que deseja seguir
GPIO.output(DIR3,CW)

# Define o Numero de Passos pro sentido horario
for x in range(pN2):

# Defina um enrolamento da bobina para alto
GPIO.output(STEP3,GPIO.HIGH)
sleep(vel3) # Ajusta a Velocidade do Motor de
Passo
# Defina um enrolamento da bobina para baixo
GPIO.output(STEP3,GPIO.LOW)
sleep(vel3) # Ajusta a Velocidade do Motor de
Passo

#""Mudar de direção: Mudar de direção requer
tempo para mudar. O
# o tempo é ditado pelo motor de passo e pelo
controlador. ""
sleep(0)
GPIO.output(DIR3,CCW)

for x in range(pN1):
GPIO.output(STEP3,GPIO.HIGH)
sleep(vel3)
GPIO.output(STEP3,GPIO.LOW)
sleep(vel3)

# Assim que terminar, limpa os IO's
except KeyboardInterrupt:

```

```
print("cleanup")
GPIO.cleanup()
```

```
def DeltaBot():
```

```
    while True:
```

```
        try:
```

```
            DefXYZ()
            cine_inv()
            cine_inv_rotp()
            cine_inv_rotp2()
            sentido()
            sentido2()
            sentido3()
            vel_mp()
            threading.Thread(target=mp).start()
            threading.Thread(target=mp2).start()
            threading.Thread(target=mp3).start()
```

```
        except KeyboardInterrupt:
```

```
            print("cleanup")
            GPIO.cleanup()
```

```
def Pzmp():
```

```
    threading.Thread(target=Pzmp1).start()
    threading.Thread(target=Pzmp2).start()
    threading.Thread(target=Pzmp3).start()
```

```
Pzmp()
DeltaBot()
```




**PUC
GOIÁS**

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
PRÓ-REITORIA DE DESENVOLVIMENTO
INSTITUCIONAL
Av. Universitária, 1069 | Setor Universitário
Caixa Postal 86 | CEP 74605-010
Goiânia | Goiás | Brasil
Fone: (62) 3946.3081 ou 3089 | Fax: (62) 3946.3080
www.pucgoias.edu.br | prodin@pucgoias.edu.br

RESOLUÇÃO n° 038/2020 – CEPE

ANEXO I

APÊNDICE ao TCC

Termo de autorização de publicação de produção acadêmica

O(A) estudante Mathias de O. e Silva
do Curso de Eng. Controle e Automação matrícula 20220000000,
telefone: 62 992044554 e-mail deoliveira.mth@pucgoias.edu.br, na
qualidade de titular dos direitos autorais, em consonância com a Lei n° 9.610/98 (Lei dos Direitos
do autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o
Trabalho de Conclusão de Curso intitulado
Robô Delta: Aplicações de Cinemática e embu-
cadas,
gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões
do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado
(Texto (PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG,
MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a
título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 23 de junho de 2022.

Assinatura do(s) autor(es): Mathias de O. e Silva

Nome completo do autor: Mathias de Oliveira e Silva

Assinatura do professor-orientador: Beuma Augusto de Oliveira

Nome completo do professor-orientador: Beuma Augusto de Oliveira