

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
DEPARTAMENTO DE ENGENHARIA DE CONTROLE E
AUTOMAÇÃO**

Controle de Manipuladores Robóticos

Thiago Tavares Pimenta

Orientador: Marco Antônio Meggiolaro

Coordenador do curso: Mauro Speranza Neto

Rio de Janeiro

Dezembro de 2009

Dedico meu trabalho e minha graduação aos meus pais
Gutemberg de Souza Pimenta e Maria Fernanda Tavares Pimenta;
à minha irmã Daniela Tavares Pimenta presente em todos os momentos da
minha vida e;
à minha namorada Camila Borsotto Monteiro Machado pela compreensão nos
momentos de dificuldades.

Agradecimento

Ao meu orientador Marco Antonio Meggiolaro e co-orientador Dado Sutter pela oportunidade de inserção em um projeto de excelência;

Ao coordenador do curso de engenharia de controle e automação, Mauro Speranza Neto, por sua disponibilidade para contribuir na formação dos alunos do curso

À Lablua, pelo espaço de trabalho, apoio e interesse pelo projeto desde o início.

Aos alunos do departamento por estarem presentes e disponíveis a ajudar em todos momentos.

Resumo

A proposta deste trabalho é o estudo de um manipulador robótico, o MA2000, englobando sua modelagem, cinemática e dinâmica. Este braço mecânico tem 6 graus de liberdade, através de 6 motores DC mais uma garra em sua extremidade. Sua área de trabalho é referente aos deslocamentos e rotações nos eixos x , y e z . Ao invés de usar o *Hardware* e o *Software* de fábrica do produto, o projeto tem como objetivos a implementação de um controle do tipo *PID* para o deslocamento, desenvolvido em linguagem *LUA*, e o desenvolvimento da eletrônica de potência para os motores. Será apresentado os conceitos para modelagens de robôs, o projeto do manipulador, descrição de suas partes, implementação da lógica e os resultados experimentais e teóricos obtidos.

ABSTRACT

The proposal of this work is the study of a robotic manipulator, the MA200, including its modeling, cinematic and dynamic. This mechanical arm has 6 degree of freedom, provided by 6 DC motors and one more gripper in its extremity. Its work area is regarding the displacements and rotations in the axis x,y and z. Instead of using the *Hardware* and the *product factory Software*, the project has as objective the implementation of a kind Control PID for the displacement, which is developed in Lua Language, and the power electronics development for the motors. It will be presented the concepts for robot's modeling, the manipulator's project, description of their parts, logic implementation and the experimental and obtained theoretical results

SUMÁRIO

INTRODUÇÃO	09
1.1 Objetivo.....	09
1.2 Organização.....	10
ROBÔS.....	11
2.1 Os Manipuladores e suas Classificações.....	12
2.2 Anatomia dos Braços Robóticos.....	16
2.2.1 Juntas (<i>Joints</i>).....	16
2.2.2 Elos (<i>Links</i>).....	17
2.2.3 Garras (<i>Grippe</i>).....	18
2.3 Graus de Liberdade (DOF – Degrees of Freedom).....	19
2.3.1 Manipulador Serial em cadeia Aberta.....	19
2.4 Cinemática de um Manipulador.....	20
2.4.1 Cinemática Espacial.....	20
2.4.2 Transformação de Coordenadas.....	21
2.4.3 Transformação Homogênea.....	22
2.4.4 Notação de Denavit-Hartenberg.....	23
2.4.5 Cinemática direta (Foward Kinemetics)	24
2.4.6 Cinemática Inversa.....	26
2.5 Análise de Movimento Diferencial.....	26
2.6 Singularidade.....	29
2.7 Redundância.....	30
2.8 Estática.....	31
2.9 Dualidade Estática e Cinemática.....	32
2.10 Servo Rigidez	32
2.11 Dinâmica.....	33
2.11.1 Newton-Euler	33
2.11.2 Formulação de Lagrange.....	34
2.11.3 Equações de movimento da Dinâmica Direta.....	35
2.11.4 Energia Cinética.....	35

2.11.5 Energia Potencial.....	36
2.11.6 Dinâmica Direta (Foward Dinamycs)	36
2.11.7 Dinâmica Inversa.....	37
2.11.7.1 Controle de Torque Computado.....	37
CONTROLE PID.....	39
MECÂNICA	42
4.1 Transmissão.....	42
4.1.1 Engrenagens.....	42
4.1.2 Caixa de redução	42
4.1.3 Correias e Polias.....	42
4.2 Atuadores	43
4.2.1 Cilindros.....	44
4.2.2 Músculo Artificial Pneumático.....	45
4.2.3 Motor.....	45
4.2.3.1 Motor de Corrente Contínua (DC- <i>Direct Current</i>)	46
4.2.3.2 Servo Motores.....	47
4.3.3.3 Motor de passo.....	48
4.3 Sensores.....	49
4.3.1 Encoders.....	49
4.3.2 Potenciômetros.....	50
CAPÍTULO 5.....	52
5.1 Lua.....	52
5.2 eLua.....	55
5.3 ARM.....	56
5.4 Protocolo	57
5.4.1 Protocolo USB.....	57
MA2000.....	59
6.1 Punho.....	60
6.1.2 Sensor.....	60
6.1.3 Controle	61
6.1.4 Eletrônica.....	62

6.1.5 Resultados	62
6.2 Braço.....	62
6.2.1 Modelagem.....	63
6.2.2 Resultados.....	67
6.3 Trabalhos Futuros.....	67
Bibliografia	68
ANEXOS	

INTRODUÇÃO

O estudo da *Robótica*¹ é um ramo da tecnologia que engloba áreas de Mecânica, Eletrônica e Computação, com graus de teoria de controle, microeletrônica, inteligência artificial, fatores humanos e de produção. Sua aplicação está associada a projetos, fabricações e desenvolvimento dos Robôs, sendo destacado na área de processos industriais.

Estudos atuais, revelam que empresas investem cada vez mais em automatização, visando otimizar seus processos. Elas contam com um argumento muito utilizado desde os tempos da revolução industrial, para que usar do esforço humano ou animal para fazer um trabalho árduo, perigoso e demorado, se um mecanismo robótico pode fazer melhor e sem oferecer esses problemas. Devido a isso, a robótica encontrou uma ampla área nas indústrias, que cada vez mais, encontra-se mais automatizada devido ao uso de robôs e manipuladores industriais.

Os braços mecânicos são os manipuladores mais encontrados no mercado, são equipamentos mecânicos pré-programados pelo homem com o objetivo de realizar determinada tarefa no espaço operacional, desde um simples movimento de um objeto até uma complexa montagem de uma peça.

1.1 Objetivo

O objetivo e estudo deste trabalho é a implementação do braço robótico *MA2000*² através do desenvolvimento do *hardware* e *software* embarcados. O sistema proposto deve ser capaz de se posicionar em para qualquer ponto do seu espaço de trabalho, entretanto, os

¹ Termo usado pela primeira vez pelo Checo Karel Capek (1890-1938) na peça (teatro) - R.U.R. (Rossum's Universal Robots) estreada em Prada-Itália(1911).

² Equipamento desenvolvido pela TecQuipment, DC Servo Controlled 6 Axis Robot (Product Code:MA2000)

testes práticos não foram realizados por completo, apenas com os elos do punho. Os elos do braço foram simulados pelo *software MatLab*³ usando a biblioteca “Robot”

1.2 Organização

O trabalho está dividido da seguinte maneira, a primeira parte falando sobre classificações, características dos robôs; a segunda parte falando modelagem, cinemática e dinâmica dos manipuladores; a terceira sobre o controle PID; a quarta fala a respeito da Mecânica; a quinta parte sobre as tecnologias usadas; e por ultimo falando sobre a parte prática do projeto.

³ Software de alta performance voltado para o cálculo numérico

2. ROBÔS

O termo *Robô* tem origem na palavra tcheca *Robota*, que significa "trabalho forçado". Robôs seriam dispositivos, ou grupo de dispositivos, eletromecânicos ou biomecânicos capazes realizar trabalhos de maneira autônoma, pré-programada podendo ser reprogramada, ou através de controle humano. Eles interagem com o meio físico, tem suas decisões baseadas em informações adquiridas, e acima de tudo respeitam as Leis de *Asimov*⁴:

- 1ª lei: Um robô não pode ferir um ser humano ou, por omissão, permitir que um ser humano sofra algum mal.
- 2ª lei: Um robô deve obedecer às ordens que lhe sejam dadas por seres humanos, exceto nos casos em que tais ordens contrariem a Primeira Lei.
- 3ª lei: Um robô deve proteger sua própria existência desde que tal proteção não entre em conflito com a Primeira e Segunda Lei.

Os robôs são divididos em duas categorias:

- Robôs móveis: possuem a capacidade de se mover ao redor do ambiente, não estão fixados na sua própria base, também conhecido por veículo robótico ou *rover*. Podem ser classificados de acordo com o ambiente que se movem, tais como ambientes terrestres, aquáticos (AUVs), aéreos (UAVs) e híbridos.



Figura 1 - AUV

⁴Escritor de Ficção Científica Isaac Asimov (1920-1992) elaborada em seu livro de ficção *I, Robot* ("*Eu, Robô*") o comportamento dos robôs (Leis da Robótica).



Figura 2 - UAV

- Robô Manipulador: “Sistema multifuncional reprogramável capaz de mover, posicionar e manipular peças, ferramentas ou dispositivos especiais” (RIA-Robotics Institute of America)⁵[3]. Um Manipulador é semelhante a um “braço mecânico” com 2 ou mais graus de liberdade. Suas definições e exemplos serão citados no próximo tópico.

Existem distintas definições para Robô. Porém todos englobam conceitos semelhantes, e não engloba no projeto tal abordagem.

2.1 Os Manipuladores e suas Classificações

- Paralelo: formado tipicamente por duas plataformas, sendo uma fixa (denominada base) e outra móvel (denominada plataforma móvel), estão ligadas entre si por duas ou mais cadeias cinemáticas abertas e independentes. Suas principais características são:
 - Boa capacidade de posicionamento;
 - Elevada capacidade de carga;
 - Grande rigidez estrutural;
 - Baixa inércia;
 - Pode atingir altas velocidades
 - Reduzido espaço de trabalho.

⁵Conceito definido pela RIA “Robotics Institute of America” para robôs. Porém para fins de explicação e compreensão, foi usado para definição de Robô Manipulador.

Entre as mais conhecidas estão a Plataforma de *Stewart*⁶ e o “*Delta Robot*”⁷. São usadas para Simulações de voo, tecnologia de máquinas e ferramentas, manipulação de veículos no espaço (NASA), entre outros.

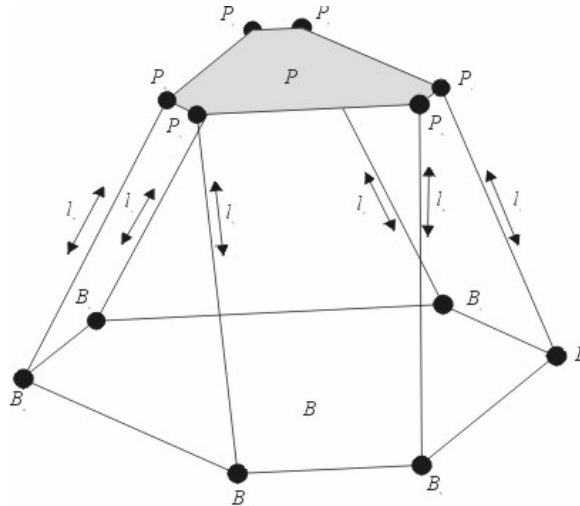


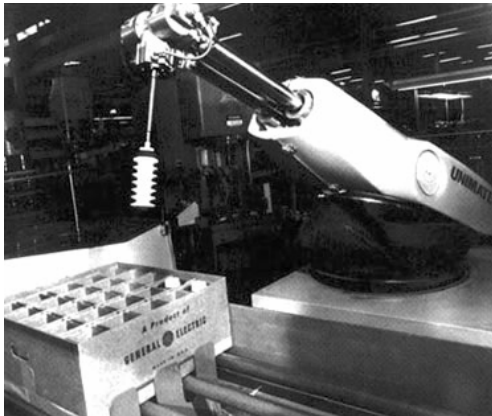
Figura 3 – Plataforma de Stewart

- Serial: Podem ser formadas por n estruturas rígidas ligadas linearmente e dependentes (A posição inicial de uma depende da final da anterior). Assemelha-se com o braço humano, onde temos um membro ligado ao próximo por articulações. Tal conceito de Juntas e Elos será apresentado mais adiante. Suas vantagens e desvantagem são:
 - Amplo espaço de trabalho;
 - Tem uma baixa rigidez;
 - Os erros são acumulados e amplificados de elo para elo.
 - Na maioria das vezes, o fato de terem que transportar e mover grande peso referente a seus atuadores (Motores, Pistões, entre outros);
 - Relativamente podem manipular baixa carga efetiva.

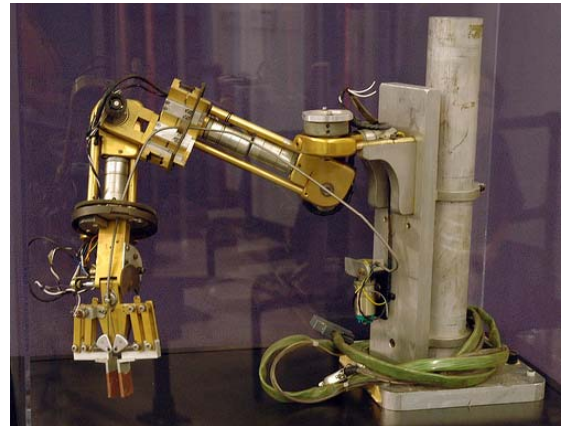
⁶Plataforma de Stewart (Gough) é um manipulador paralelo com 6 graus de liberdade – 1965 by D.Stewart

⁷Delta Robot é um manipulador paralelo de 4 graus de liberdade - 1980 by Reymond Clavel at the École Polytechnique Fédérale de Lausanne (EPFL, Switzerland).

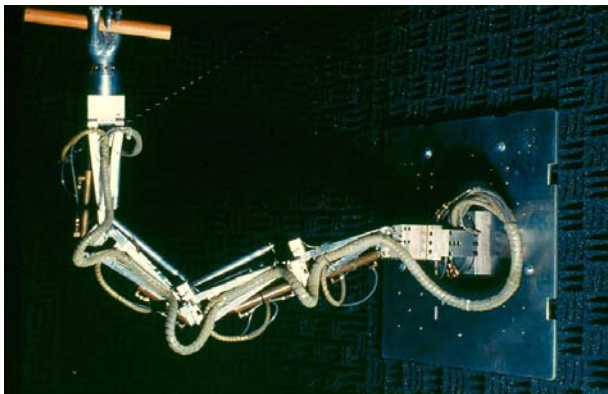
Abaixo serão apresentados uns dos principais manipuladores seriais.



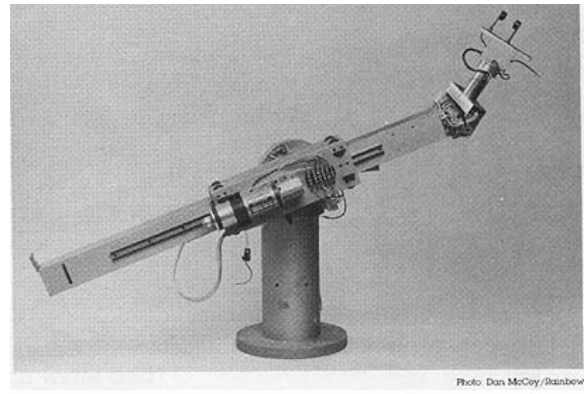
“Ultimate“ primeiro robô industrial usado na linha de montagem pela *General Motors* (1961). O protótipo do “Ultimate” foi o MH-1, desenvolvido por Heinrich Ernst no MIT(1961)



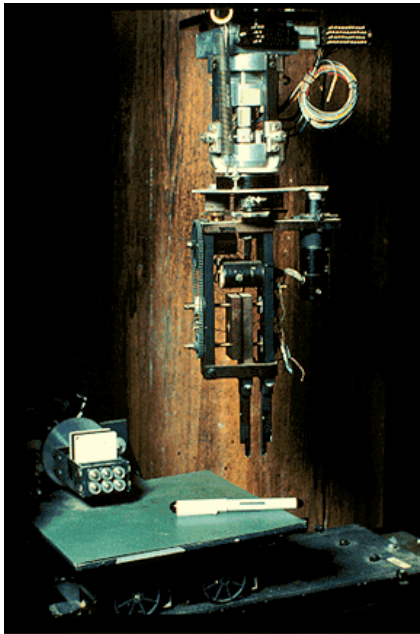
“Rancho Arm” com 6 graus de liberdade dando uma flexibilidade semelhante a de um braço humano. Primeiro robô artificial controlado por computador. (1963 – Stanford University



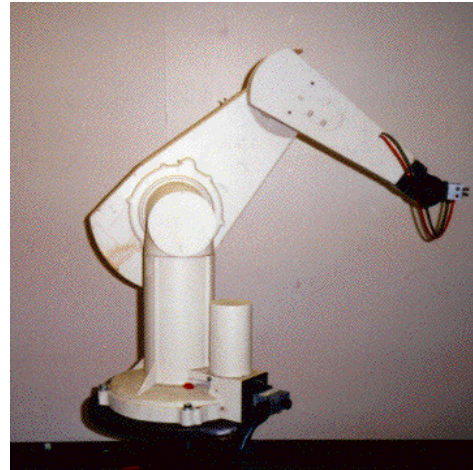
“Tentacle Arm” capaz de levantar o peso de um ser humano, através das suas 12 juntas com atuadores de fluidos hidráulicos. (1968 by Marvin Minsky)



“Stanford Arm” Primeiro robô elétrico com sucesso, que influência até hoje o desenvolvimento de novos manipuladores elétricos. (1969-Victor Scheinman-SAIL) Stanford Artificial Intelligence Lab



“Silver Arm” capaz de montar pequenas partes usando com realimentação sinal dos sensores de pressão (1974 - Victor Scheinman)



“Puma Arm”Primeiro robô industrial elétrico, foi baseado no manipulador de Stanford. (1978- Desenvolvido pela Unimation para a General Motors). Programmable Universal Machine for assembly



“SCARA Arm” (1979 - Sankyo and IBM), Selective Compliance Assembly Robot Arm



“Salisbury Hand” (1982).

2.2 Anatomia dos Braços Robóticos

Basicamente o braço robótico é composto por uma base, por braços e por um punho. A base geralmente é dita como o referencial do sistema, ou seja, o referencial fixo da base. O braço consiste em elementos denominados elos, que são conectados entre eles por juntas que realizam os movimentos relativos, que por sua vez estão acoplados aos atuadores responsáveis pelos movimentos, dotados de uma capacidade sensorial e instruídos por um sistema de controle. Na extremidade do braço encontra-se o punho, que consiste em conexões de juntas próximas entre si, permitindo a orientação da extremidade do manipulador no espaço. No último elo, na extremidade, temos o “*end effector*” que está diretamente ligado a algum tipo de garra ou ferramenta, responsáveis por realizar uma determinada tarefa exigida pelo sistema.

2.2.1 Juntas (*Joints*)

Como dito anteriormente, as juntas conectam os braços e são responsáveis por permitir determinados tipos de movimentos. Abaixo serão citados os tipos de juntas e suas funcionalidades.

- Prismática ou Linear (P - *Prismatic*) – Movem-se em linha reta, compostas de duas hastes que deslizam entre si;
- Rotacional (R - *Revolute*) - Giram em torno de uma linha imaginária estacionária denominada eixo de rotação. Considerando como um exemplo, os eixos x, y e z, podemos dizer que uma junta é rotacional no eixo z, ou seja, ela terá a liberdade de realizar o movimento de rotação em torno do eixo z;
- Esférica (S - *Spherical*) - Funcionam com a combinação de três juntas de rotação, realizando a rotação em torno de três eixos;
- Parafuso (H - *Helical*) - Constituídas de um parafuso que contém uma porca ao qual executam movimento semelhante ao da junta prismática, porém, com movimentos no eixo central (movimento do parafuso).

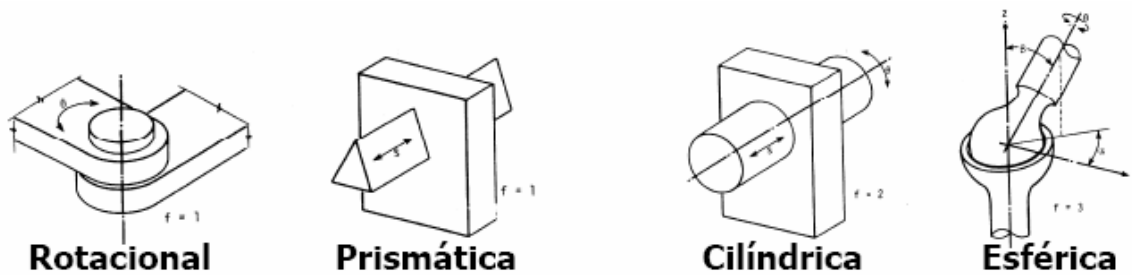


Figura 4

2.2.2 Elos (*Links*)

Um elo é um corpo rígido, idealmente sem deformação pela ação de forças aplicadas sobre estes, que define uma relação entre duas juntas adjacentes de um manipulador. Eles são enumerados de forma crescente, desde a base imóvel, sendo o elo 0, passando para o primeiro em movimento, o elo 1, e assim sucessivamente. Podem se apresentar de diversos tamanhos e formatos.

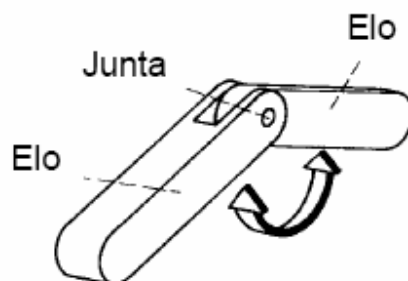


Figura 5

2.2.3 Garras (Gripper)

Conectados ao ultimo elo, as garras ou ferramentas são responsáveis por realizar determinado tipo de tarefa, como por exemplo: aparafusar um determinado equipamento, soldar uma determinada peça, encaixar um pino em uma determinada superfície, pegar algum tipo de instrumento através de uma pinça. Logo chamamos de “*end effector*” o ponto da extremidade que iremos definir como o ponto final do sistema, ou seja, um ponto

localizado a uma distância $Pe = \begin{pmatrix} x_e \\ y_e \\ z_e \end{pmatrix}$ do referencial da base(origem 0 do sistema).

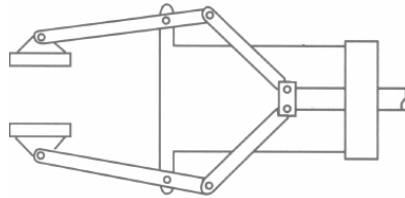


Figura 6

Os manipuladores ainda podem se encontrar em dois tipos diferentes de cadeias cinemáticas, as cadeias abertas, onde a trajetória entre dois corpos é única (número de corpos moveis é igual ao número de juntas) e as cadeias fechadas, encontram em formas de loop, onde $n_l = n_g - n_b$;

- n_l - número de Loops;
- n_g - número de Juntas;
- n_b - número de corpos em movimento.

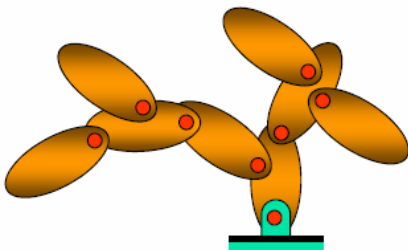


Figura 6 – Cadeia Aberta

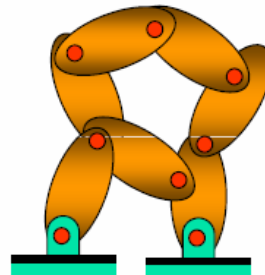


Figura 7 – Cadeia Fechada

2.3 Graus de Liberdade (DOF – Degrees of Freedom)

“O número de graus de liberdade se refere à liberdade de movimento no espaço cartesiano” (Wikipédia, 22 agosto 2009). Com isso temos que o DOF determinam os movimentos do manipulador no espaço bidimensional ou tridimensional

2.3.1 Manipulador Serial em cadeia Aberta

Cada junta define um ou mais de dois graus de liberdade, com isso, o numero de graus de liberdade de um manipulador é igual ao somatório do grau de liberdade de suas n juntas.

$$DOF = \sum_{i=1}^n f_i ;$$

- n - numero de juntas do manipulador;
- f_i - numero de graus de liberdade da junta.

Por exemplo, quando o movimento relativo ocorre em um único eixo, a junta tem grau de liberdade 1, caso o movimento se dê em mais de um eixo, a junta tem 2 graus de liberdade.

Como apresentado anteriormente, temos quatro tipos de juntas mais importantes, as juntas Prismáticas com 1 grau de liberdade, Rotacional com 1, Helicoidal com 1 e a Esférica com 3 graus de liberdade.

Os demais manipuladores seguem o critério de Grueber⁸, onde o número de graus de liberdade segue a seguinte formula:

$$DOF = \lambda \times (n_b - n) + \sum_{i=1}^n f_i ;$$

- λ - para Mecanismos Espaciais $\lambda = 6$, para Mecanismos Planos $\lambda = 3$;
- n_b - número de corpos em movimento;
- n - numero de juntas do manipulador;
- f_i - numero de graus de liberdade da junta.

⁸ Equação de Kutzbach-Gruebler usada para calculo do número de graus de liberdade

O número de graus de liberdade de um manipulador está associado ao número de variáveis posicionais independentes que permitem definir a posição de todas as partes de forma unívoca. Com isso, observa-se que quanto maior a quantidade de graus de liberdade, mais complicadas é a cinemática, a dinâmica e o controle do manipulador.

2.4 Cinemática de um Manipulador

A cinemática de um manipulador é o estudo do conjunto de relações entre posição e deslocamentos dos seus elos. Nos próximos tópicos serão analisados alguns conceitos importantes para modelagem da cinemática direta e a cinemática inversa.

Para que se possa fazer o controle correto de um manipulador, primeiramente, é preciso conhecer e calcular alguns conceitos corretamente. Os próximos tópicos serão meramente teóricos, e os resultados do Manipulador MA2000 serão apresentados em outro instante.

2.4.1 Cinemática Espacial

Conceito fundamental para conhecer a posição de um dado ponto no espaço, em um determinado referencial.

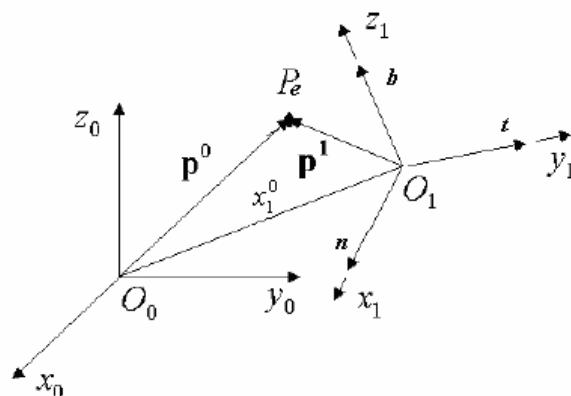


Figura 8

No sistema anterior, observa-se duas maneiras distintas de se representar o ponto da extremidade $P_e = \begin{pmatrix} x_e \\ y_e \\ z_e \end{pmatrix}$, uma em relação à origem O_0 , representada no sistema de coordenadas x_0, y_0, z_0 pelo vetor p^0 , e outra em relação à origem O_1 , representada no sistema x_1, y_1, z_1 pelo vetor p^1 . Ambos têm como finalidade representar o ponto P_e , porém estão sendo representados em sistemas e valores diferentes.

Para que haja cálculos corretos e sem erros, é necessário tomar muito cuidado com a escolha do referencial. Portanto, daqui em diante, sempre tomaremos como referencial principal o sistema da base, a origem O_0 .

Os vetores \mathbf{n} , \mathbf{t} , \mathbf{b} acima estão representando as de coordenadas x_1, y_1, z_1 no sistema de coordenadas de x_0, y_0, z_0 . São os vetores de módulo unitários nas das direções dos eixos x_1, y_1, z_1 respectivamente, e ortogonais entre si. Com esses três vetores, temos a Matriz de Orientação (\mathbf{R}), designada por:

$$\mathbf{R} = \begin{bmatrix} n_i & t_i & b_i \end{bmatrix}_{3 \times 3} \text{ para } i = 1, 2, \dots, n;$$

Referente à representação do sistema de coordenadas de $[x_i \ y_i \ z_i]$ em relação ao sistema $[x_{i-1} \ y_{i-1} \ z_{i-1}]$, tal que $\begin{cases} |n| = |t| = |b| = 1 \\ n^T t = n^T b = t^T b = 0 \end{cases}$.

O vetor x_1^0 representa a distância entre a origem O_1 e a origem O_0 no referencial da origem O_0 . De uma forma análoga, temos que o vetor x_i^{i-1} representa a distância entre a origem O_i e a origem O_{i-1} no sistema de coordenadas de O_{i-1} .

2.4.2 Transformação de Coordenadas

A partir dos conceitos citados acima, é possível localizar qualquer ponto localizado no espaço em relação ao seu sistema principal através das transformações de coordenadas, sendo:

$$x_{i-1} = x_i^{i-1} + R \cdot x_i$$

$$x_{i-1} = x_i^{i-1} + \begin{bmatrix} n_i & t_i & b_i \end{bmatrix} \cdot \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \text{ para } i = 1, 2, \dots, n;$$

→ x_{i-1} a distância de um ponto qualquer do espaço em relação à origem principal, ou

$$\text{seja, a origem } O_{i-1}, \text{ tal que } x_{i-1} = P_e = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix};$$

→ x_i^{i-1} a distância do deslocamento da origem O_{i-1} para a origem O_i ;

→ R Matriz Rotação;

→ x_i a distância do ponto P_e em relação à origem O_i , tal que $x_i = P_e = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix}$.

2.4.3 Transformação Homogênea-

A Transformação Homogênea nos trás algumas informações importantes, tais como a Matriz Rotação, que nos mostra a rotação do sistema de O_i em relação ao sistema O_{i-1} e a Matriz 3x1 que informa a componente de translação, isto é, o valor do deslocamento entre os sistemas O_i e O_{i-1} .

$$\underbrace{\begin{pmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \\ 1 \end{pmatrix}}_{\mathbf{X}_{i-1}} = \underbrace{\begin{pmatrix} \underbrace{\begin{bmatrix} n_i & t_i & b_i \end{bmatrix}}_{\substack{\text{Matriz de Rotação} \\ 3 \times 1}} & \underbrace{\begin{pmatrix} x_i^{i-1} \end{pmatrix}}_{\substack{\text{Deslocamento da origem } O_{i-1} \text{ para a} \\ \text{origem } O_i}} \\ \hline 0 & 0 & 0 & 1 \end{pmatrix}}_{\mathbf{A}_i^{i-1}} \cdot \underbrace{\begin{pmatrix} x_i \\ y_i \\ z_i \\ 1 \end{pmatrix}}_{\mathbf{X}_i}$$

Considerando um ponto desejado no sistema sendo a extremidade do robô, e n origens no espaço, obtém-se a seguinte relação:

$$X_0 = \underbrace{A_1^0 A_2^1 A_3^2 \cdots A_n^{n-1}}_{A_n^0} \cdot X_n;$$

sendo $A_n^0 = \begin{bmatrix} n_e & t_e & b_e & x_e^0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, onde e – extremidade do robô.

2.4.4 Notação de Denavit-Hartenberg

Os parâmetros de Denavit-Hartenberg permitem obter o conjunto de equações que descreve a cinemática de uma junta com relação à junta seguinte e vice-versa. São 4 os parâmetros: o ângulo de rotação da junta θ , o ângulo de torção da junta α , o comprimento do elo a e o deslocamento da junta d , que serão mostrados na figura a seguir.

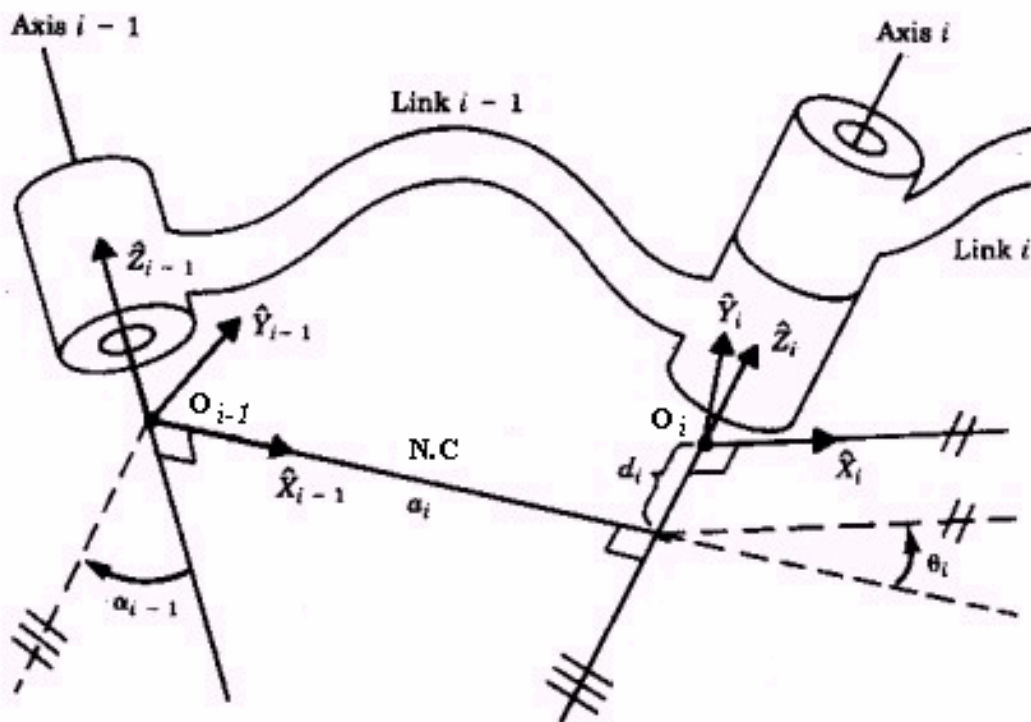


Figura 9

- O_i - (Interseção) \cap da Normal Comum (N.C) e a $Junta_{i+1}(link_{i+1})$;
- Z_i - Alinhado com a $Junta_{i+1}(link_{i+1})$;
- X_i - Prolongamento da N.C ou Perpendicular a Z_i e Z_{i-1} ;
- Y_i - Regra da Mão Direita⁹.

Parâmetros de Denavit-Hartenberg

- a_i - Comprimento da N.C;
- d_i - Distância entre O_{i-1} e N.C;
- θ_i - Ângulo entre X_{i-1} e X_i (na direção de Z_{i-1});
- α_i - Ângulo entre Z_{i-1} e Z_i (na direção de X_i).

Para Juntas Rotativas temos d_i constante e para Juntas Prismáticas O_i constantes.

Definição da Matriz A_i^{i-1}

$$A_i^{i-1} = \begin{bmatrix} \cos \theta_i & -\text{sen} \theta_i \cos \alpha_i & \text{sen} \theta_i \text{sen} \alpha_i & a_i \cos \theta_i \\ \text{sen} \theta_i & \cos \theta_i \cos \alpha_i & -\cos \theta_i \text{sen} \alpha_i & a_i \text{sen} \theta_i \\ 0 & \text{sen} \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.4.5 Cinemática direta (Foward Kinematics)

Temos como entrada do sistema na cinemática direta as coordenadas das juntas, e na saída às coordenadas finais do Ponto da extremidade P_e (“*End Effector*”). Portanto com o uso da notação de Denavit-Hartenberg é possível expressar a posição e a orientação da extremidade P_e (“*End Effector*”), como também analisar o comportamento do sistema variando as variáveis de entrada q_i , que podem ser:

- $q_i = \theta_i$ para juntas Rotacionais (R);
- $q_i = d_i$ para juntas Prismáticas (P).

⁹ Regra e recurso mnemônico que determina a orientação do resultado do produto vetorial

A posição e orientação do elo i relativo ao elo $i-1$ é descrito sendo uma Matriz 4x4 em função de q_i , representada por $A_i^{i-1}(q_i)$.

O primeiro passo para se determinar a Cinemática direta de um manipulador será descrever a posição e orientação do último elo em relação à base, assim como a função de deslocamento das juntas, de q_1 à q_n .

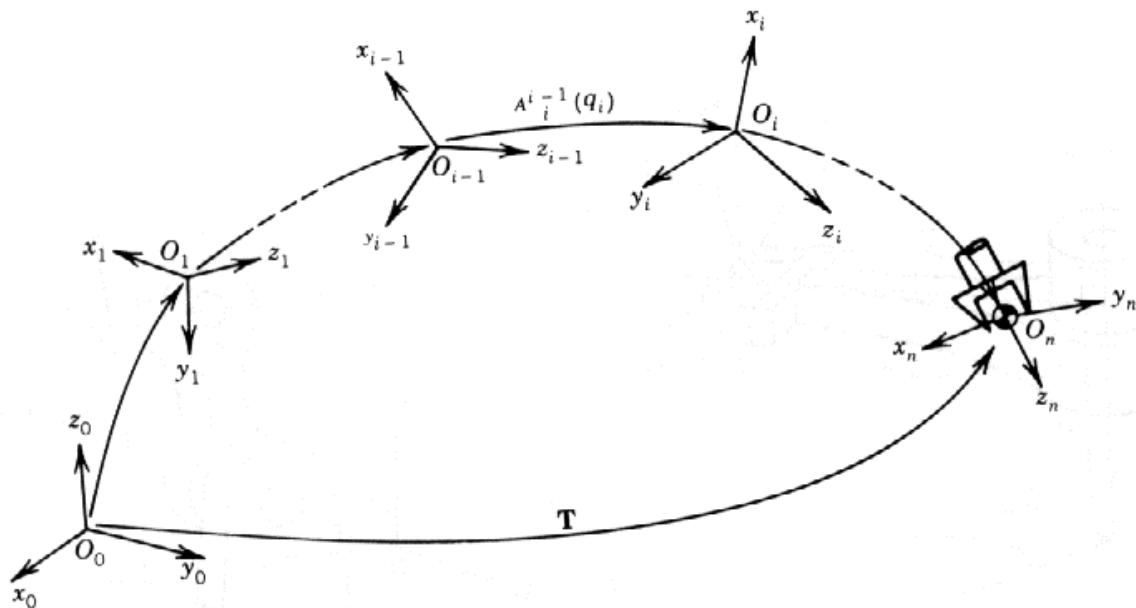


Figura 10

Considerando as n transformações de coordenadas consecutivas ao longo do manipulador serial, podemos encontrar uma relação final da extremidade com o seu referencial da base, a origem O_0 .

$$T = A_1^0(q_1) \cdot A_2^1(q_2) \cdot A_3^2(q_3) \cdot \dots \cdot A_n^{n-1}(q_n);$$

onde T é uma matriz 4x4 que representa a posição da extremidade referente ao sistema

de coordenadas da origem O_0 , ou seja, $T^T = \begin{matrix} & & & \\ & & & \\ & & & \\ x_0 y_0 z_0 & x_e & y_e & z_e & 1 \end{matrix}$.

2.4.6 Cinemática Inversa

Diferente da Cinemática direta, a Cinemática inversa tem como parâmetros de entrada as coordenadas do Ponto da extremidade P_e (“*End Effector*”) para encontrar achar na saída possíveis ângulos ou deslocamentos de q_i , ou seja, procura determinar o conjunto de valores das juntas que se adequam a uma dada configuração no espaço operacional.

No entanto, nem sempre é possível achar soluções analíticas, pois pode haver mais de uma resposta para o mesmo problema, como exemplo funções não lineares ou funções trigonométricas, por vez nem mesmo é possível, achar soluções, como casos onde os parâmetros de entrada fogem dos limites do manipulador. Quando ocorre o fato de encontrar mais de uma solução, dizemos haver uma redundância no sistema, tópico que será estudando adiante.

De uma forma geral temos que $A_n^{0^{-1}} \cdot T = q(n)$, onde $q(n)$ serão os deslocamentos das juntas em relação ao eixo x_{x-1} .

A Cinemática inversa é essencial para o controle robótico, no entanto, seus cálculos são realizados por algum software numérico. No caso de manipuladores paralelos a dinâmica inversa é mais simples do que comparado com manipuladores em série.

2.5 Análise de Movimento Diferencial

Os estudos de cinemática direta e inversa abordam-se as relações entre coordenadas no espaço operacional e relações entre coordenadas das juntas de um dado sistema robótico. No entanto, essas relações não dão qualquer informação a respeito das características do movimento entre duas ou mais configurações quaisquer, ou seja, não trazem informações a respeito das evoluções temporais das coordenadas no espaço, tanto a velocidade angular quanto a velocidade linear.

Para uma primeira análise, iremos considerar um manipulador planar de apenas 2 graus de liberdade, ou seja, se movimentando restrito apenas aos eixos x e y .

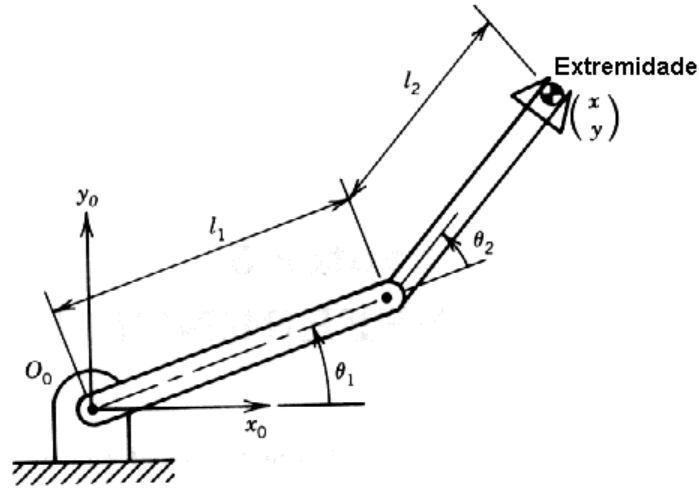


Figura 11

As equações da cinemática que relaciona o ponto $P_e = \begin{pmatrix} x_e \\ y_e \end{pmatrix}$ com os deslocamentos angulares de $q = \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix}$ são representadas por:

$$\begin{aligned} x(\theta_1, \theta_2) &= l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \\ y(\theta_1, \theta_2) &= l_1 \text{sen}(\theta_1) + l_2 \text{sen}(\theta_1 + \theta_2) \end{aligned}$$

Como estamos interessados nas velocidades das posições e configurações em relação ao tempo, passaremos a trabalhar com as derivadas das variáveis no tempo. As variáveis $p = \begin{pmatrix} x \\ y \end{pmatrix}$ e $q = \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix}$ derivadas no tempo ficam $\dot{p} = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix}$ e $\dot{q} = \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{pmatrix}$, que por sua se correlacionam através da fórmula $\dot{p} = J \cdot \dot{q}$, onde J representa a matriz Jacobiana, que relaciona as velocidades no espaço operacional em função das velocidades das juntas.

$$J = \begin{bmatrix} \frac{\partial F_1}{\partial q_1} & \frac{\partial F_1}{\partial q_2} & \dots & \frac{\partial F_1}{\partial q_n} \\ \frac{\partial F_2}{\partial q_1} & \frac{\partial F_2}{\partial q_2} & \dots & \frac{\partial F_2}{\partial q_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial F_m}{\partial q_1} & \frac{\partial F_m}{\partial q_2} & \dots & \frac{\partial F_m}{\partial q_n} \\ \frac{\partial q_1}{\partial q_1} & \frac{\partial q_2}{\partial q_2} & \dots & \frac{\partial q_n}{\partial q_n} \end{bmatrix}$$

Voltando ao exemplo temos $J = \begin{bmatrix} -l_1 \text{sen}(\theta_1) - l_2 \text{sen}(\theta_1 + \theta_2) & -l_2 \text{sen}(\theta_1 + \theta_2) \\ l_1 \cos(\theta_1) l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$,

sendo $J_1 = \begin{bmatrix} -l_1 \text{sen}(\theta_1) - l_2 \text{sen}(\theta_1 + \theta_2) \\ l_1 \cos(\theta_1) l_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$ referente às derivadas em relação ao $q_1 = \theta_1$ e

$J_2 = \begin{bmatrix} -l_2 \text{sen}(\theta_1 + \theta_2) \\ l_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$ referente às derivadas em relação ao $q_2 = \theta_2$.

Podemos representar a velocidade linear como $v_e = \dot{p}_e = J_1 \dot{\theta}_1 + J_2 \dot{\theta}_2$. Caso a junta referente ao elo 2 seja fixa ($\dot{\theta}_1 \neq 0$ e $\dot{\theta}_2 = 0$), ou seja, não tendo o movimento de rotação em seu eixo (eixo z), teremos a velocidade da extremidade sendo $v_e = J_1 \dot{\theta}_1$. Diferente quando a junta 1 está fixa na base ($\dot{\theta}_1 = 0$ e $\dot{\theta}_2 \neq 0$), sendo assim temos a velocidade linear da extremidade $v_e = J_2 \dot{\theta}_2$.

Para um manipulador com n graus de liberdade, teremos:

$$v_e = \dot{p}_e = J_1 \dot{\theta}_1 + J_2 \dot{\theta}_2 + \dots + J_n \dot{\theta}_n.$$

Para representar as velocidades da extremidade no sistema, a velocidade linear e angular, teremos uma Matriz $6 \times m$ denominada J_i , onde m é o número de variáveis do sistema.

A matriz J_i será representada por duas matrizes, a J_L ($3 \times m$) referente à velocidade

Linear, e a J_A ($3 \times m$) referente à angular, na qual $\dot{p} = \begin{bmatrix} J_L \\ J_A \end{bmatrix}_{6 \times 1} \cdot \begin{pmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{pmatrix}$.

$$J_i = \begin{bmatrix} J_L \\ J_A \end{bmatrix} = \begin{cases} \begin{bmatrix} J_L \\ b_{i-1} \end{bmatrix} \rightarrow \text{Junta Rotativa} \\ \begin{bmatrix} J_L \\ 0 \end{bmatrix} \rightarrow \text{Junta Prismática} \end{cases}$$

\ Non contribui com a velocidade angular

- J_L - será o Jacobiano calculado anteriormente $\dot{p} = J \cdot \dot{q}$, onde $J_L = J$;
- b_{i-1} - representa a matriz J_A , com os valores de b que representam os deslocamentos das juntas no eixo z, encontrados na matriz A_n^0 na transformação Homogênea. Na junta Prismática com não temos rotação no eixo z sua matriz

$$b = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}_{3 \times m}, \text{ onde } m \text{ irá representar o numero de juntas prismáticas no sistema.}$$

2.6 Singularidade

A Singularidade se diz respeito às configurações do manipulador que impedem a extremidade de gerar velocidades não nulas ($v \neq 0$) em alguma direção, ou então, a configurações que o manipulador perca graus de liberdade.

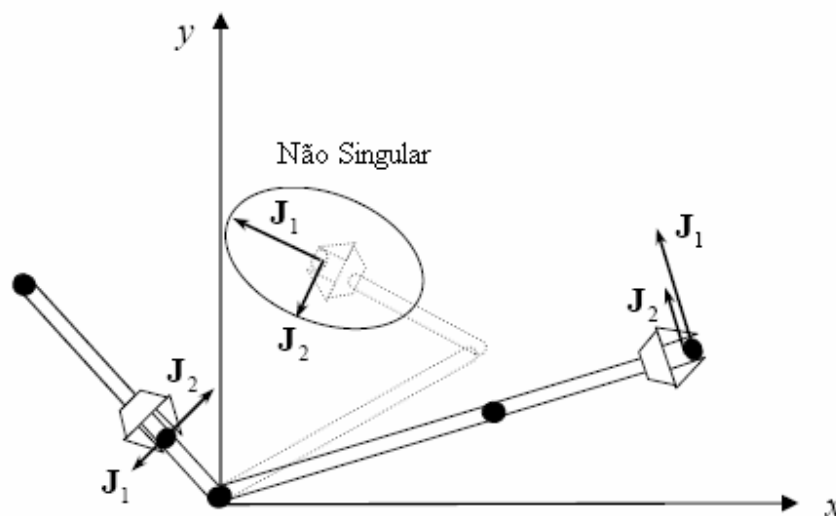


Figura 12

Para o calculo de singularidade basta achar valores que satisfaça a equação a seguir:

$$\det J = 0$$

Por definição, temos que nas relações de cinemática direta o Jacobiano é sempre definido e singular. Já o Jacobiano inverso é não definido, portanto a singularidade é válida apenas para:

$$\dot{p} = J \cdot \dot{q}$$

e não é válido para o caso:

~~$$\dot{q} = J^{-1} \dot{p}$$~~

2.7 Redundância

Um sistema é Redundante quando há mais de grau de mobilidade do que os necessários. Considerando $\dot{p} = J \cdot \dot{q}$ e $p \in R^m, q \in R^n / n > m$, conclui-se que há mais de um valor de q que satisfaça a equação. Portanto, são configurações distintas de q que resultam no mesmo ponto no espaço operacional.

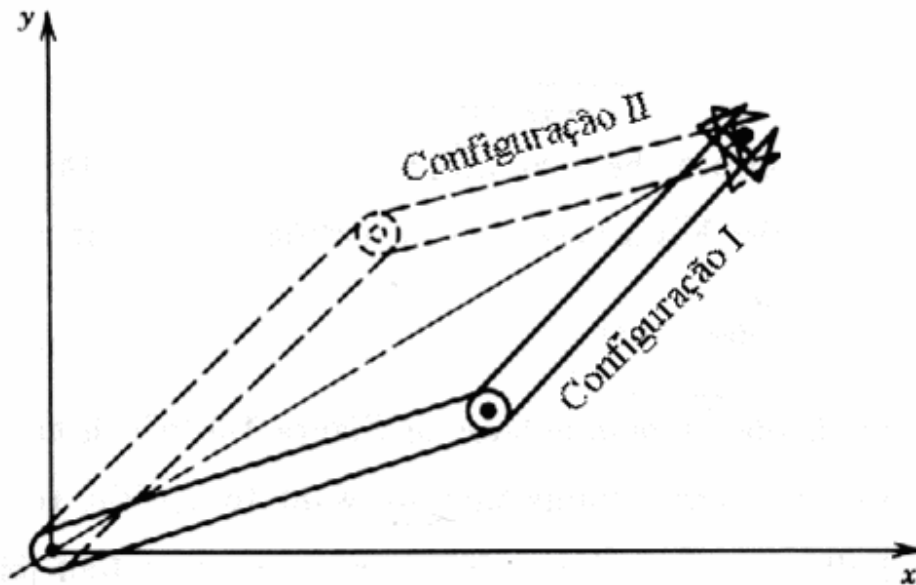


Figura 13

2.8 Estática

Os manipuladores têm como sua principal função exercer determinado tipo de trabalho no espaço operacional, no entanto, essa interação com o meio, pode exigir algum contato mecânico com outros equipamentos. Considerando-se um braço robótico em repouso, ao ser acoplado uma ferramenta de massa m em sua extremidade, aparecerá uma força F (na direção da aceleração da gravidade) em sua extremidade. Para o manipulador continuar em repouso, ele terá que exercer uma contrária, uma força $-F$ (na direção contrária de F), além do mais, o surgimento dessa força F , causará um determinado momento ($N=F*d$) nas juntas robô, e novamente para o manipulador continuar parado, as juntas terão de exercer um torque τ contrário a esse momento.

Por isso é fundamental o conceito de estática para determinar tais forças e torques. Um exemplo dado em sala de aula:

“Imagina-se, uma braço robótico que tem como função escrever com giz em um quadro negro. O manipulador terá que controlar a força com que o giz irá fazer contra o quadro, pois, caso a força seja grande o giz poderá quebrar, ou caso a força seja pequena ele não irá escrever no quadro”. (Marco Antonio Meggiolaro, Introdução à Robótica-2009, PUC-Rio). O controle dessas forças será feito através do controle dos torques das Juntas.

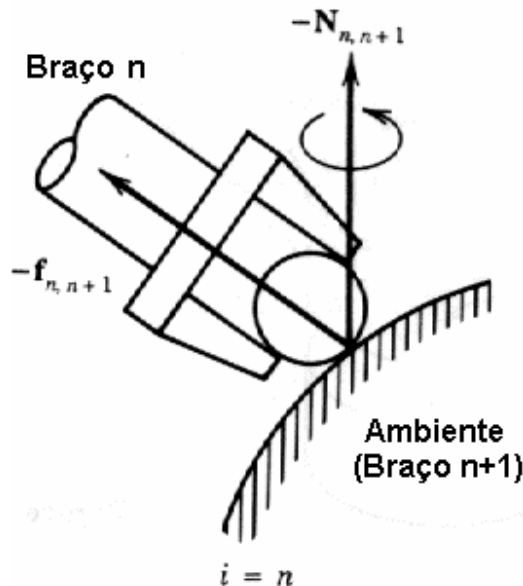


Figura 14

2.9 Dualidade Estática e Cinemática

Através da dualidade entre as duas é possível descobrir as forças e torques desejados, onde $\tau = J^T F$.

$$\tau = \begin{pmatrix} \tau_1 \\ \vdots \\ \tau_n \end{pmatrix} = [J]^T \begin{pmatrix} F_x \\ F_y \\ F_z \end{pmatrix}$$

2.10 Servo Rigidez

Quando uma força é aplicada na extremidade do manipulador, essa extremidade tende a se defletir em relação a seu ponto inicial, dependendo da sua rigidez e da força aplicada. Portanto, a rigidez do manipulador determina o quanto ele é flexível e o quanto a extremidade irá se deslocar ao ser aplicado a força.

Temos que:

$$\begin{aligned} \rightarrow \tau_i &= k_i \Delta q_i ; \\ \rightarrow \tau &= K \Delta q = J^T F ; \\ \rightarrow K &= \begin{bmatrix} k_1 & & 0 \\ & \ddots & \\ 0 & & k_n \end{bmatrix} \end{aligned}$$

$$\tau = \begin{pmatrix} \tau_1 \\ \vdots \\ \tau_n \end{pmatrix} = \begin{bmatrix} k_1 & & 0 \\ & \ddots & \\ 0 & & k_n \end{bmatrix} \cdot \begin{pmatrix} \Delta q_1 \\ \vdots \\ \Delta q_n \end{pmatrix}$$

ERRO - deslocamento devido a Força aplicada

Matriz de Ganho das Juntas
Matriz de rigidez das Juntas

Considerando um Δp pequeno, então $\Delta p \cong J \Delta q \rightarrow \Delta q = K^{-1} \tau$, substituindo novamente, $\Delta p = JK^{-1} \tau = JK^{-1} J^T F$. A partir desse ponto, pode-se deduzir a matriz de flexibilidade da extremidade $C_e = JK^{-1} J^T$.

Para achar a matriz de rigidez da extremidade basta fazer o calculo de C_e invertido, ou seja, como $\Delta p = C_e F \rightarrow F = K_e \Delta p$, $C_e = K_e^{-1}$.

Assumindo uma força unitária de módulo igual a 1, é possível através da matriz de flexibilidade analisar as direções em que o manipulador é mais rígido ou mais flexível. Os deslocamentos da extremidade serão representados pelos os autovalores de C_e , e as direções pelo autovetores de C_e . Quanto maior seu autovalor maior será sua flexibilidade, quanto menor maior será sua rigidez.

2.11 Dinâmica

Ramo da ciência que trata da ação da força em corpos físicos, em movimento ou em repouso, considerando a cinética, cinemática e estática, tratando-as coletivamente [Webster Dictionary].

O comportamento dinâmico é descrito em função das taxas de variação de deslocamento das configurações do manipulador e em função dos torques que os atuadores exercem sobre as juntas, expressadas pelas equações de movimento.

Podem ser usados dois métodos para descrever as equações de movimento, sendo Formulação de Newton-Euler e outra Formulação Lagrangeana.

2.11.1 Newton-Euler

As equações de Newton-Euler descrevem a dinâmica do sistema em termos de Força e Momento. A formulação incorpora todas as forças e momentos que atua em cada elo individualmente, incluindo forças e momentos referentes ao acoplamento.

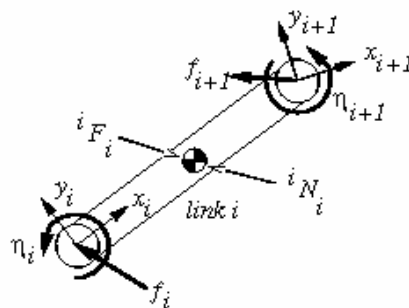


Figura 15

As equações de Newton-Euler não serão desenvolvidas nem analisadas nesse projeto. Para análise do comportamento da dinâmica será usada a Formulação Lagrangeana, no próximo tópico. Para melhores informações sobre a Formulação de Newton-Euler consultar as referências usadas nesse capítulo.

2.11.2 Formulação de Lagrange

Como dito anteriormente o modelo dinâmico de um robô descreve as relações existentes entre os torques aplicados nas juntas e o movimento dos elos robóticos. As equações de Lagrange dependem das coordenadas generalizadas, que são o conjunto de variáveis independentes que localizam unicamente um sistema.

A Formulação de Lagrange é uma alternativa mais simples quando comparada com as Equações de Newton-Euler, que não são expressas em função das coordenadas generalizadas e não incluem os torques exercidos nas juntas explicitamente. A mecânica Lagrangeana descreve o comportamento dinâmico em termos de Trabalho e Energia do sistema ao invés de Momentos e Forças aplicadas em cada junta individualmente.

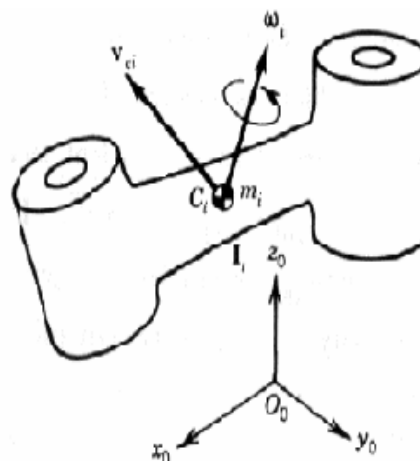


Figura 16

2.11.3 Equações de movimento da Dinâmica Direta

Na dinâmica direta teremos como entrada do sistema os Torques e as Forças aplicadas pelo sistema, e teremos como saída às velocidades e acelerações das juntas do manipulador.

O sistema mecânico é definido por: $L(q_i, \dot{q}_i) = T(q_i, \dot{q}_i) - U(q_i)$, onde T é a energia Cinética e U a energia Potencial armazenada do sistema. A energia Potencial está em função das coordenadas generalizadas (q_i), enquanto que a Cinética em função das velocidades (\dot{q}_i) assim como as coordenadas generalizadas (q_i). A partir do Lagrangeano

obtem-se a equação de Lagrange dada por: $\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}_i} \right) - \frac{\partial L}{\partial q_i} = Q_i$ $i = 1, \dots, n$. Onde Q_i são as forças generalizadas, que poderão ser as forças ou os torques gerado pelos manipuladores, ou demais sistemas.

2.11.4 Energia Cinética

$$T = \sum_{i=1}^n T_i = \sum_{i=1}^n \frac{1}{2} \left(m_i |Vc_i|^2 + \omega c_i^T I_i \omega c_i \right)$$

- $Vc_i = J_L^i \dot{q}_i$ - Velocidade linear no centro de massa do elo i ;
- $\omega c_i = J_A^i \dot{q}_i$ - Velocidade angular no centro de massa do elo i ;
- $|Vc|^2 = Vc^T Vc$.

Substituindo as velocidades chegamos à seguinte equação:

$$T = \frac{1}{2} \sum_{i=1}^n \left(m_i q_i^T J_L^{i^T} J_L^i \dot{q}_i + q_i^T J_A^{i^T} I_i J_A^i \dot{q}_i \right)$$

- J_L - Matriz Linear;
- J_A - Matriz Angular;
- I_i - Matriz de Tensor de Inércia relacionada ao elo i .

Onde a Matriz Inércia do Manipulador H é definida como:

$$H = \left(m_i J_L^{iT} J_L^i + J_A^{iT} I_i J_A^i \right)$$

H é uma matriz simétrica, positiva e definida (autovalores > 0) e seus valores dependem das configurações dos ângulos.

2.11.5 Energia Potencial

$$U = -\sum_i^n m_i g^T J_{L_i}^{(i)}$$

→ $J_{L_i}^{(i)}$ - Primeira coluna da Matriz Linear de i

2.11.6 Dinâmica Direta (Forward Dynamics)

Com o resultando da energia do sistema L, podemos substituir na equação de Lagrange chegando ao resultado igual:

$$\tau_i = \sum_{j=1}^n H_{ij} \ddot{q}_j + \sum_j^n \sum_k^n h_{ijk} \dot{q}_j \dot{q}_k + G_i = Q_i$$

- τ_i - Torque associado à junta i ;
- $H_{jk} \ddot{q}_j$ - Representa quando $j=i$ o momento de inércia da junta i quando todas as outras juntas encontram-se bloqueadas e quando $j \neq i$ leva em conta o efeito da aceleração da junta j sobre a junta i ;
- $h_{ijk} \dot{q}_j \dot{q}_k$ - Representa o efeito Centrífugo e de Coriolis na junta i , provocado pela velocidade da junta j e k ;
- G_i - Representa o Torque gerado no eixo i devido ao efeito da Gravidade.

De um modo geral, quando temos um motor preso no próprio elo atuando diretamente na junta então:

$$\delta W = \tau_1 \delta \theta_1 + \dots + \tau_n \delta \theta_n = Q_1 + \dots + Q_n.$$

Porém, quando os motores estão presos à base do manipulador, temos:

$$\delta W = \underbrace{(\tau_1 + \tau_n)}_{Q_1} \delta \theta_1 + \dots + \underbrace{\tau_n}_{Q_n} \delta \theta_n.$$

Para maiores informações a respeito do desenvolvimento das contas, pesquisar nas referências citadas para esse capítulo.

2.11.7 Dinâmica Inversa

Na Dinâmica inversa temos como entrada as velocidades e as acelerações das juntas do manipulador, e como saída teremos os torques aplicados nelas. Essencial para o controle de um braço mecânico. Usando um Controle tipo PID, o torque seria a realimentação para o sistema, e a diferença das acelerações e velocidades desejadas com as velocidades e acelerações atuais das juntas, seria o ERRO. O controle tem como função tentar eliminar esse erro, e para isso enviará um sinal de realimentação para o sistema.

2.11.7.1 Controle de Torque Computado

Resumidamente, este tipo de controle irá substituir o termo da aceleração pelo sinal de controle do sistema. Fazendo uma breve análise, temos:

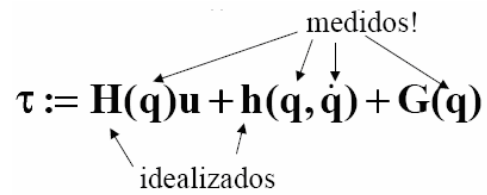
$$\tau_i = \sum_{j=1}^n H_{ij} \ddot{q}_j + \sum_j \sum_k h_{ijk} \dot{q}_j \dot{q}_k + G_i ;$$

Considerando os termos de acoplamento entre juntas h e os termos da gravidade G , a equação se resume:

$$\tau = H\ddot{q} + h(q, \dot{q}) + G(q);$$

Supondo uma modelagem correta, ou aproximadamente correta, teremos os termos de H e h da equação, e os termos de q e \dot{q} poderão ser medidos por sensores do sistema. Portanto, ao substituir o termo da aceleração \ddot{q} pelo sinal de controle do sistema u_i , teremos o torque τ_i desejado na saída. A nova Lei de Controle ficaria:

$$\tau := \mathbf{H}(\mathbf{q})\mathbf{u} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q})$$



3. CONTROLE PID

Um Controlador PID (Proporcional, Integral e Derivativo) é um genérico Controle Loop Mecanismo, que tem como principal função eliminar o erro de um determinado sistema. O erro é a diferença entre o valor do processo decorrente e seu valor desejado, seu *SetPoint*. O controle PID agirá em cima desse erro de forma corretiva no sistema, tentando ajustar o processo de forma rápida e eficiente.

Existem três parâmetros distintos para cálculos do controle, o parâmetro Proporcional de ganho K_p ; o parâmetro Integral de ganho K_i e o parâmetro Derivativo de ganho K_d .

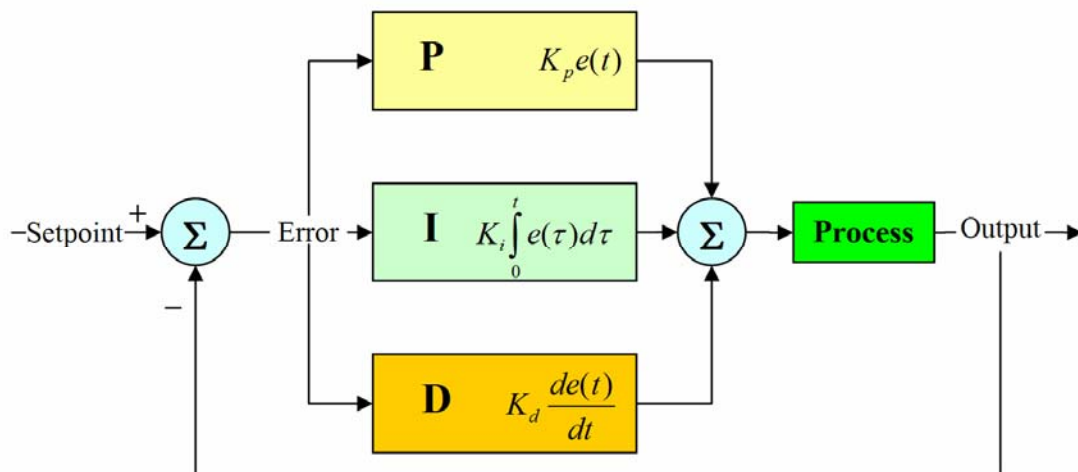


Figura 17

Temos que a saída do processo (*Output*) é dada pelo somatório dos termos proporcional, integral e derivativo.

$$u(t) = MV(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt}(t)$$

A função de transferência do sistema é dado por:

$$G_{pid} = \frac{u(s)}{r(s)} = K \left(1 + \frac{1}{T_i s} + \frac{spT_d}{s+p} \right) = \frac{K(s^2 + \frac{1+T_d T_i}{T_i} s + \frac{p+T_i p}{T_i})}{s(s+p)}$$

A resposta do sistema ao termo proporcional será o erro atual multiplicado por um ganho K_p determinado, ou seja:

$$P_{out} = K_p e(t)$$

A ação proporcional pode acelerar a resposta de um processo de controle, ou seja, sua variação tem influência direta no tempo de subida ou descida na curva de resposta do sistema, podendo ser mais rápida ou mais lenta. No entanto quando calculados ou usados incorretamente seu tempo de estabilização pode ficar bastante demorado, ou até mesmo, produzir um “over-shoot” (diferença da curva do sistema com o valor desejado) indesejável. Geralmente usando apenas os termos proporcionais, o processo se apresenta com um caráter oscilatório, para eliminar tais oscilações é usado o termo derivativo.

O termo derivativo determina a reação do sistema com base na taxa em que o erro for variando no tempo, a resposta pode ser ajustada multiplicando a variação do erro num determinado tempo a um ganho K_d (ganho derivativo), dado por:

$$D_{out} = K_d \frac{de}{dt}(t)$$

A ação derivativa está ligada diretamente com o amortecimento do sistema, que de certa forma dissipa a energia, fazendo com que o sistema convirja para o estado desejável, ou seja, tende a acelerar e estabilizar a malha. A ação deste termo ajuda na redução do “overshoot”, porém não elimina o erro por completo, para isso é usado o termo integrativo.

O termo integral determina a reação baseada na soma dos últimos erros, tendo uma variação na saída proporcional tanto a magnitude do erro quanto a duração do erro.

A resposta pode ser ajustada multiplicando a integral do erro em relação ao tempo a um ganho K_i (ganho integral), dado por:

$$I_{out} = K_i \int_0^t e(\tau) d\tau$$

A ação integral responde ao passado do erro enquanto este for diferente de zero, ele tende a eliminar o erro para se alcançar ao valor desejável, reduz o valor de subida, no entanto pode gerar um pouco de instabilidade no sistema. Esse termo integral produz respostas lentas e oscilatórias, porém tende a estabilizar a malha.

Abaixo, uma tabela com uma pequena simplificação correspondente a respostas aos termos Proporcional, Integral e Derivativo.

Resposta	Tempo de Subida	Oscilação	Tempo de Estabelecimento	Erro Estacionário
Proporcional	Diminuição	Aumenta	Sem Alteração	Diminuição
Integral	Diminuição	Aumenta	Aumenta	Elimina
Derivativo	Sem Alteração	Diminuição	Diminuição	Sem Alteração

Tabela 1

A seleção do controlador deve depender das condições operacionais do sistema e de especificações desejadas, tais como, o erro estacionário máximo, oscilações máximas, tempo de estabilização, entre outras. Por exemplo, se o erro estacionário não é tolerado, então, o controle exigirá a ação do termo integral, que agirá no processo de maneira a eliminar o erro. Outro caso, seria se fosse desejado a eliminação das oscilações no sistema, para isso, usaríamos o termo derivativo que influencia e suas eliminações.

4. MECÂNICA

4.1 Transmissão

Responsável por transmitir ao sistema os torques e forças geradas pelos atuadores. Apresentam-se de diversas formas, sendo algumas descritas abaixo.

4.1.1 Engrenagens

Engrenagens são elementos rígidos utilizados na transmissão de movimentos rotativos entre eixos. Sua principal característica é que não haja qualquer diferença de velocidade entre pontos em contato. A razão entre o número de dentes na engrenagem é diretamente proporcional à razão de torque e inversamente proporcional à razão das velocidades de rotação.

4.1.2 Caixa de redução

Caixas de redução são sistemas mecânicos acoplados nos motores, compostos por diferentes tipos e combinações de engrenagens. Tais combinações são denominadas de estágio, sendo o último estágio ligado diretamente ao eixo de saída. As caixas de reduções têm como finalidade reduzir as velocidades de rotação do eixo do motor e aumentar o torque resultante no eixo de saída.

4.1.3 Correias e Polias

É um dos meios mais antigos de transmissão de movimentos. Tem como principais características o fato de ser simples, baixo custo, durabilidade boa quando dimensionadas

adequadamente, reduzem significativamente a propagação de choques e vibrações, operam silenciosamente e limitam sobrecargas pela ação do deslizamento.

As correias e polias podem ser de diversos formatos, tais como: plana, trapezoidal (em 'V'), redonda ou dentada.

Alem de correias, as transmissões podem ser feitas por cabos, ou tendões presos a polia.

4.2 Atuadores

Para que haja a movimentação linear ou rotacional das juntas do manipulador no espaço, é necessário o acoplamento de certos mecanismos denominado atuadores. Podem estar conectados as próprias juntas ou não. Em alguns casos podem-se encontrar juntas em movimento mesmo sem nenhum atuador ligado a ela, logo, observa-se a influência de outra junta sobre esta. Os atuadores podem ser de tipos diferentes: rotacionais, lineares, pneumáticos, hidráulicos, cabos flexíveis.

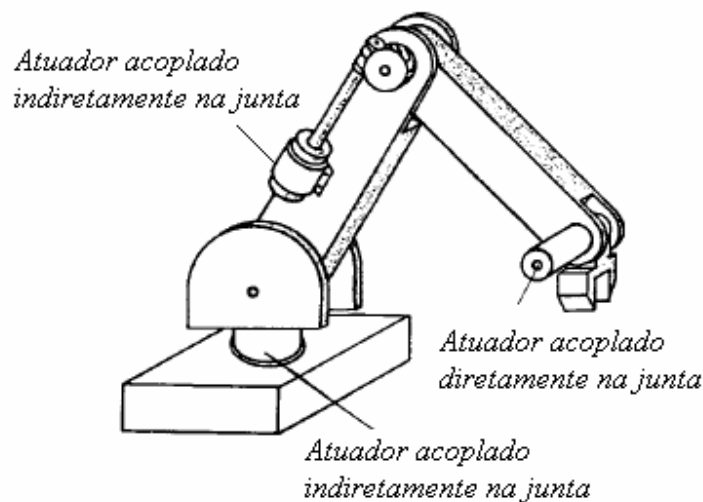


Figura 18

➤ Acionamento Indireto

O acionamento indireto se caracteriza por ter o atuador afastado da junta que ele controla, ou seja, ele não está acoplado diretamente a junta. Geralmente ocorre esse acionamento quando o se deseja diminuir a carga que a junta terá que mover. Porém para funcionar, terá que existir alguma forma de transmissão de potência entre o atuador e a junta, entretanto esse método sofre efeitos indesejados no desempenho do robô, devido à folga nas engrenagens, escorregamento da polia, flexão dos vínculos do manipulador.

➤ **Acionamento Direto**

O atuador está diretamente conectado a junta, esse acionamento permite uma melhor precisão e rendimento de potencia em relação ao acionamento indireto.

4.2.1 Cilindros

Conhecido também pelo o nome de pistão, os cilindros são acionados e controlados pelas válvulas. O fluido de escoamento responsável por sua movimentação pode ser o ar, a água, ou qualquer outro tipo de fluido. Lembrando que estão restritos apenas por deslocamentos lineares nos seus extremos. Para que haja um movimento de rotação é preciso uma relação de transmissão responsável por isso.



Figura 19 -Cilindros da Festo (Empresa Alemã de Automação Industrial)

4.2.2 Músculo Artificial Pneumático

Trata-se de um sistema de contração de membrana, ou seja, o tubo quando submetido a uma pressão, aumenta a extensão de seu diâmetro, criando uma força de tensão e um movimento de contração na direção longitudinal de até 25% do seu comprimento inicial sem carga.

Quando distendido, o músculo desenvolve até dez vezes mais força que um cilindro pneumático convencional de mesmo diâmetro e consome apenas 40% da energia para uma força idêntica.

O músculo pneumático realiza trabalho semelhante ao músculo humano, pode ser utilizado em ambientes agressivos carregados de pó, cerragem e até mesmo mergulhado em água, pois tem a característica de ser impermeável.



Figura 20 – Músculos Artificiais Pneumáticos da Festo

4.2.3 Motor

Por definição, motor é um dispositivo responsável por converter uma determinada forma de energia em energia mecânica, com a finalidade de impelir o movimento a um determinado objeto. Podem gerar movimentos rotacionais quanto lineares as juntas dependendo da transmissão feita.

4.2.3.1 Motor de Corrente Contínua (DC- *Direct Current*)

É um motor elétrico que tem seu acionamento por corrente contínua. São conhecidos por seu controle preciso de velocidade e por seu ajuste fino. Tendo em conta uma análise elétrica, podemos modelar o motor DC sendo um resistor (resistência de armadura), um indutor (indutância de armadura) e uma fonte de tensão, todos em série, onde o valor da tensão é diretamente proporcional a velocidade do motor. Segue abaixo alguns parâmetros para a escolha de um motor adequado.

- V_{input} - a tensão aplicada aos terminais (medida em V);
- K_t - a constante de torque do motor, que é a razão entre o torque gerado pelo motor e a corrente elétrica nele aplicada (medida em Nm/A);
- R_{motor} - resistência elétrica entre os terminais do motor (medida em Ω), quanto menor o seu valor, maiores são as correntes que o motor consegue puxar, e maior seu torque;
- I_{no_load} - corrente elétrica requerida para o motor girar sem nenhuma carga no seu eixo (medida em A), quanto menor o seu valor menos atrito existe nos rolamentos/buchas do motor.

As equações para um motor DC são:

$$\tau = K_t \times (I_{input} - I_{no_load})$$
$$\omega = K_v \times (V_{input} - R_{motor} \times I_{input})$$

onde:

- τ - torque aplicado pelo motor em um dado instante (em Nm);
- ω - velocidade angular do motor (em rad/s, para RPM multiplicar por 9.55);
- I_{input} - corrente elétrica que atua no motor (em A);
- K_v - a constante de velocidade do motor, é a razão entre a velocidade do motor e a tensão nele aplicada (medida em (rad/s)/V) é calculada por $K_v = 1/K_t$;

Apesar de desprezarem a indutância do motor, as equações acima são boas aproximações se a corrente não variar bruscamente.

Temos que a potência elétrica consumida vale $P_{input} = V_{input} \times I_{input}$, e a potência mecânica gerada pelo motor é $P_{output} = \tau \times \omega$.

Em um motor ideal, não há nenhuma perda por atrito, logo $I_{no_load} = 0$, e a sua resistência elétrica é zero, $R_{motor} = 0$, e nesse caso sua eficiência é de 100%, ou seja, $\eta = 1$. Como não existe um motor ideal, os motores reais possuem $0 \leq \eta < 1$ (eficiência entre 0 e 100%).

4.2.3.2 Servo Motores

Trata-se de um motor, em geral de corrente contínua, com um sensor de posição ou de velocidade acoplado a ele, que permite ao controlador conhecer essas grandezas físicas e assim controlá-las. Muito usado na área da robótica devido sua precisão.

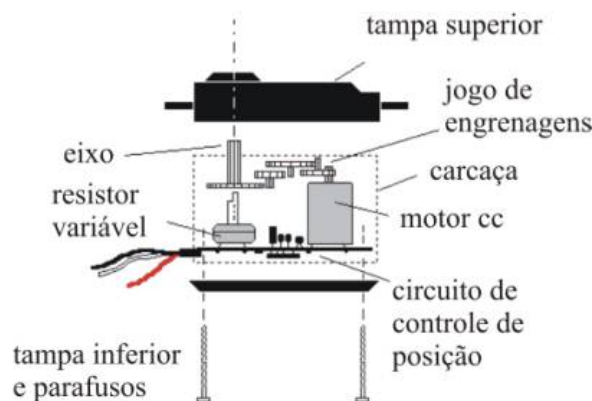


Figura 21

Os servos-motores de posição exigem como entrada de referência um sinal elétrico PPM (Pulse Position Modulation), onde a largura do pulso é proporcional à posição desejada. O controle de posição é feito através de uma eletrônica interna, que compara a tensão no potenciômetro acoplado ao eixo de saída da caixa de redução do motor com a tensão gerada pelo o circuito de medição da duração do pulso quadrado. Assim sendo, o motor DC irá girar até que o potenciômetro atinja a posição desejada.

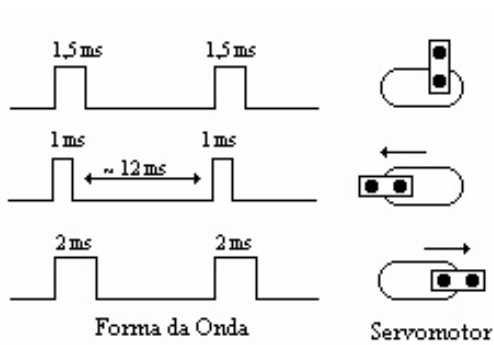


Figura 22

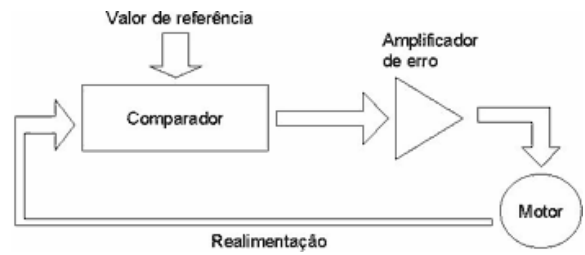


Figura 23

4.2.3.3 Motor de passo

Os motores de passo se movimentam em pequenos passos (movimentos incrementais que podem chegar a frações de grau) com o objetivo de conseguir um posicionamento muito preciso, ou fazer a rotação do eixo em um ângulo exato. Estes motores se caracterizam pelo alto torque em baixas velocidades, e principalmente pela sua capacidade de manter o eixo estático em determinada posição, com a energização das suas bobinas.

Seu controle é feito eletricamente, pois requer acionamento sincronizado e seqüencial das suas bobinas.

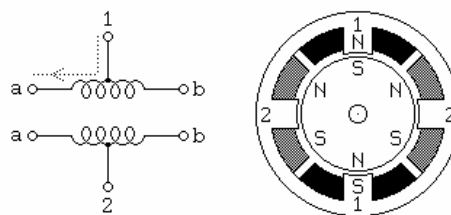


Figura 24

Por terem uma precisão boa, os motores de passo são utilizados em sistemas de malha aberta, ou seja, sem realimentação.

4.3 Sensores

Basicamente um sensor é um elemento que recebe e responde a um estímulo ou a um sinal. Os sensores são responsáveis por fazer a leitura do meio que estão trabalhando. Muitas vezes o sensor é composto de um transdutor e um aparte que converge à energia resultante em um sinal elétrico. Transdutor por sua vez, é um dispositivo que transforma uma determinada forma de energia em outra adequada para fins de medida, não necessariamente em um sinal elétrico.

Em sistemas de malha fechada, é normal o uso dos sensores para ajudar na realimentação do sistema, ou seja, sabendo a leitura correta dos sensores e possível saber o quanto o sistema terá que ser realimentado para chegar ao seu estado desejado.

Existem diversos tipos de sensores, tais como: Sensores de posição, temperatura, velocidade, som, aceleração, presença, resistência e muitos outros.

4.3.1 Encoders

São transdutores de movimento capazes de converter movimentos lineares ou angulares em informações elétricas, que podem ser transformadas em informações binárias e trabalhadas por um programa que converta as informações passadas. Ou seja, é uma unidade de realimentação que informa sobre posições atuais do sistema que possam ser comparadas com posições desejadas e seus movimentos sejam planejados.

Internamente, os encoders possuem um ou mais discos (mascara) perfurado, que permite, ou não, a passagem de um feixe de luz infravermelha gerado por um emissor que se encontra de um lado do disco e captado por um receptor do outro lado do disco. Este, com o apoio de um circuito eletrônico gera um pulso, e dessa forma a velocidade ou posicionamento é registrada contando-se o número de pulsos gerados.

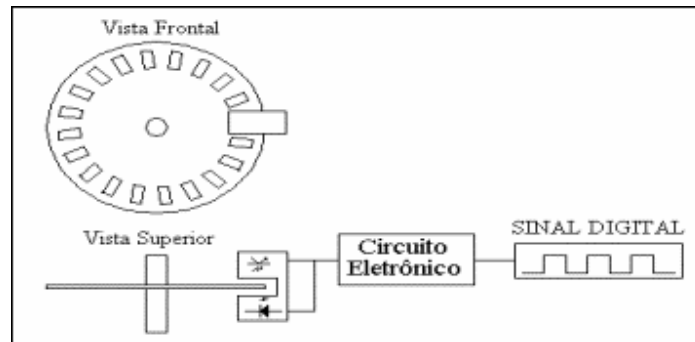


Figura 25

4.3.2 Potenciômetros

É um componente eletrônico que possui resistência elétrica ajustável, geralmente é um resistor de três terminais onde a conexão central é deslizante e manipulável. Se todos os três terminais são usados, ele atua como um divisor de tensão.

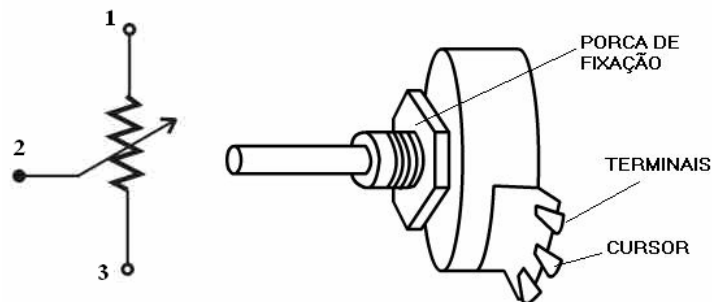


Figura 26

No modelo acima, ligaríamos o terminal 1 no VCC e o 3 no Terra (GND), restando o terminal 2 como cursor, ou seja, com sua resistência variando. O valor da tensão no terminal 2 seria:

$$V_{out} = \frac{R_3}{R_1 + R_3} \times V_{in}$$

- V_{in} - Tensão de entrada no terminal 1;
- V_{out} - Tensão de saída no terminal 2;
- R_1 - Resistência do terminal 1 ao 2;
- R_3 - Resistência do terminal 2 ao 3;

Dessa maneira o potenciômetro funcionará de forma linear, portanto, é possível traçar uma reta que representando a tensão em função da resistência variável.

5. TECNOLOGIA

Neste capítulo serão abordados alguns conceitos da linguagem Lua e suas funcionalidades, assim como os conceitos de eLua, ARM e o tipo de protocolo usado para a comunicação.

5.1 Lua

Lua¹⁰ é uma linguagem de programação dinâmica, desenvolvida na Pontifícia Universidade Católica do Rio de Janeiro, PUC-Rio, desde 1993. Atualmente desenvolvida no laboratório É uma linguagem simples e poderosa, dado sua integração natural com C e sua flexibilidade. Lua tem um papel fundamental na história das linguagens de programação e por isto foi incluída na terceira edição do History of Programming Language¹¹ juntamente com outras linguagens populares como C++ e Haskell. Para ilustrar as vantagens de seu uso segue uma lista das características que mais chamam a atenção:

➤ Robusta

Lua tem sido usada em muitas aplicações na indústria, como o Adobe LightRoom¹². Com ênfase em jogos, especialmente no maior Game Multiplayer online atual World of Warcraft, em sistemas embarcados como o no robô "Crazy Ivan"¹³ e até no tradicional LEGO¹⁴, que em sua versão moderna inclui um microcontrolador ARM. Lua tem um manual de referência sólido e livros de alta qualidade escritos a seu respeito.

¹⁰www.lua.org

¹¹<http://research.ihost.com/hopl/HOPL.html>

¹²<http://since1968.com/article/190/mark-hamburg-interview-adobe-photoshop-lightroompart-2-of-2>

¹³www.hougaard.com/robot

¹⁴www.hougaard.com/robot

➤ **Rápida**

"Benchmarks" independentes mostram que Lua está entre uma das mais rápidas linguagens interpretadas existentes.

O código fonte pode ser encontrado na Wiki dos usuários de Lua¹⁵. Este Benchmark mostra como Lua é mais rápido do que outras linguagens populares de script.

Seu interpretador ainda pode ser ajustado para máxima performance em aplicações muito específicas. Além disso, sua integração natural com a linguagem C garante grande velocidade nos mecanismos de chamadas externas.

➤ **Portátil**

Escrita em ANSI C, pode ser compilada para qualquer ambiente que tenha suporte aos padrões da ANSI¹⁶, o que significa virtualmente qualquer ambiente embarcado de hoje.

Uma vez que o interpretador tenha sido compilado para determinado ambiente, todos os programas Lua que utilizem bibliotecas padronizadas poderão ser executados sem nenhuma alteração no código fonte.

➤ **Controle de memória**

Lua possui seu próprio sistema de gerenciamento de memória, que cuida da alocação e liberação de espaço em RAM. Gerência de memória é um problema complexo em sistemas muito grandes sem esta funcionalidade. Em sistemas embarcados, onde memória é um recurso escasso, o sistema de "garbage collection" pode evitar desperdícios de memória ou até mesmo erros como "memory leak"¹⁷.

¹⁵<http://lua-users.org/lists/lua-l/1999-09/msg00036.html>

¹⁶American National Standards Institute: <http://www.ansi.org/>

¹⁷ Termo para definir a geração recorrente de áreas da memória que são alocadas e tem a referência perdida sem liberação

➤ **Pequena**

O interpretador Lua pode incluindo as bibliotecas básicas. ocupar menos do que 100 kilobytes de código, Este é um fator muito importante em dispositivos com limitações de memória não-volátil, em particular, memória Flash. É importante ressaltar que é praticamente impossível reduzir o custo em memória das máquinas virtuais de outras linguagens dinâmicas sem denegrir funcionalidades importantes. Assim sendo, o tamanho do interpretador Lua é imbatível, com todas as funcionalidades disponíveis.

Além disso, o uso mínimo de memória dinâmica é muito importante para a performance do sistema, até em dispositivos com maior capacidade, pois a velocidade de acesso a memória é menor do que nos desktops ou servidores.

➤ **Simples**

Lua tem uma sintaxe simples, que pode ser descrita em uma página¹⁸. A semântica é consistente e intuitiva. Aprender e ensinar Lua é relativamente fácil, além de existirem livros de alta qualidade no mercado. Não é necessário nenhum ambiente especial para o desenvolvimento ou bibliotecas complexas para sua utilização.

Lua tem funcionalidades (como closures, corrotinas e meta-mecanismos) que a torna atraente para projetos profissionais, porém nenhum destes mecanismos é de uso obrigatório, não dificultando o aprendizado de iniciantes.

➤ **Código Aberto**

Todo o código fonte de Lua está disponível no site oficial da linguagem, além de diferentes projetos que fornecem o interpretador compilado, pronto para ser usado. Lua é distribuída sob uma licença de software liberal¹⁹, onde não há restrições para uso acadêmico ou industrial. Esta característica também é muito importante por favorecer a participação de muitos desenvolvedores no projeto, gerando uma comunidade qualificada e altamente prestativa.

¹⁸www.lua.org/manual/5.1/manual.html#2.1

¹⁹ A licença MIT, suas diretrizes podem ser encontradas no seguinte endereço:
<http://www.opensource.org/licenses/mit-license.html>

5.2 eLua

O termo eLua vem de "Embedded Lua", Lua "embarcado" em português. eLua não é apenas uma versão adaptada de Lua para microcontroladores, pois adiciona bibliotecas para acesso a funcionalidades de hardware, permitindo controle dos periféricos internos dos microcontroladores, que em geral só são acessados pela linguagem nativa do microcontrolador.

O projeto é uma junção de dois desenvolvimentos anteriores:

- ReVaLuaTe²⁰, um terminal de vídeo com Lua embarcado, desenvolvido por Bogdan Marinescu, um engenheiro de software romeno e premiado em um concurso da Renesas.
- O Projeto Volta²¹ desenvolvido no Laboratório de Monitoramento Ambiental Remoto da PUC-Rio (LabMAR), pelo pesquisador Dado Sutter.

Seguindo a filosofia de Lua, eLua também é um projeto livre e de código aberto. Por ser uma linguagem interpretada, Lua permite que trechos de código possam ser carregados dinamicamente, em tempo de execução. Por esta característica, os sensores que executam eLua "herdam" esse dinamismo, permitindo soluções modernas e importantes, como a alteração do firmware online, sem a necessidade de procedimentos locais de atualização de software. Pode-se assim adicionar funcionalidades de software remotamente a dispositivos microcontrolados. Outra vantagem é a facilidade de expansão, permitindo a criação rápida e organizada de módulos complementares para novos periféricos.

Outro ponto importante são as corrotinas de Lua. Normalmente as implementações de multi-tarefa em microcontroladores são baseadas em agendadores de tarefas, que são particularmente difíceis de se implementar com a qualidade desejada e consomem recursos importantes de processamento e memória. Com Lua sendo executado há a vantagem das corrotinas, que são funcionalidades da linguagem que permitem que processos concorrentes sejam executados de modo simples, coerente e coordenado. Esta característica de Lua facilita imensamente as tarefas de depuração de código concorrente.

²⁰ www.circuitcellar.com/renesas2005m16c/winners/1685.htm

²¹ www.eluaproject.net/en/Authors

Para o início das implementações de eLua, foram escolhidos os microcontroladores da família ARM, dado sua popularidade, disponibilidade e custo baixo. Mas isto não é um pré-requisito para a execução de eLua, é necessários apenas que a arquitetura seja 32-bits (por enquanto), que exista um compilador ANSI C para a plataforma-alvo e que o dispositivo cumpra os requisitos mínimos de memória que atualmente são: 128 kilobytes de memória flash e 32 kilobytes de memória RAM, embora o recomendável seja 256 kilobytes de flash e 64 kilobytes ou mais de RAM.

5.3 ARM

O termo ARM se refere a uma arquitetura de processadores RISC de 32-bits, introduzida em 1983 pela inglesa ARM Holdings (na época conhecida como Acorn computers). Atualmente esta arquitetura é muito utilizada em microcontroladores e está ganhando espaço nas aplicações de sensoriamento[16,17]. Por esta grande aceitação eLua foi desenvolvido inicialmente para processadores ARM.

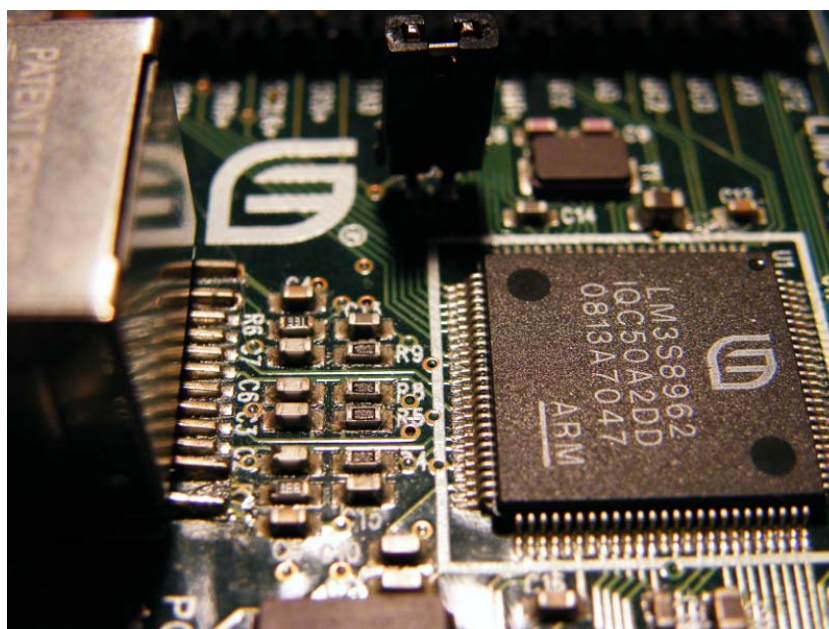


Figura 27 - Controlador ARM Cortex M3 da Luminary Micro

As implementações iniciais do "VHeWS", por terem sido feitas em eLua, foram também baseadas em ARMs. Em especial, no modelo Cortex M3, ilustrado acima que contém 256 kilobytes de memória flash e 64 kilobytes de memória RAM.

A versão de eLua utilizada nestes processadores inclui o interpretador Lua completo e foi compilada utilizando o GNU C Compiler - GCC - e as ferramentas de link, assembly e programação para processadores ARM.

5.4 Protocolo

Temos como definição de Protocolo um conjunto de informações ou dados que passam por um preparo para serem repassados a outros programas, isto é, para haver a comunicação, conexão ou transferência de dados entre dois sistemas computacionais (ex: comunicação entre dois computadores, transferência de dados de um microprocessador para um computador, sensores, etc) é preciso que estes tenham a mesma convenção ou padrão, tenham o mesmo tipo de protocolo. Portanto, é o conjunto de regras que especifica o formato, a sincronização, o seqüenciamento e a verificação de erros, na comunicação entre computadores. Os protocolos podem ser implementados pelo Hardware, Software ou por uma combinação dos dois.

5..4.1 Protocolo USB

O protocolo USB (Universal Serial Bus)²² foi introduzido em 1996 com o principal objetivo de acelerar a comunicação serial entre o computador e os periféricos. Devida a ausência de padronização de conectores de periféricos, necessidade de uma única interface de simples conexão, facilitando a conexão de dispositivos (periféricos) ao computador, em 1995 um conjunto de empresas, entre elas Microsoft, Intel, NEC, IBM, Apple, formaram um consórcio dando origem ao padrão, surgindo então o USB Implementers Forum.

²² www.usb.org

O padrão USB é um tipo de conexão Plug and Play (“ligar e usar”), ou seja, o computador reconhece e configura automaticamente qualquer dispositivo que seja instalado (plugado), facilitando a expansão segura dos computadores e eliminando a configuração manual. O padrão permite que sejam conectados até 127 equipamentos em cada micro com velocidades de transmissão de 1,5 ou 12 Mbps (até 480 Mbps na versão 2.0)

O protocolo USB é o modo de comunicação entre a máquina e a placa usada no projeto, ou seja, é através desse protocolo que a troca de sinais e informações é feita.

6. MA2000

O manipulador MA2000 constitui se em um braço mecânico com 6 graus de liberdade e em sua extremidade acoplado uma pinça com acionamento pneumático. Junto com o equipamento existe uma central de controle, onde toda sua manipulação poderia ser feita. No entanto, o projeto foi desenvolvido sem nenhum contato com tal equipamento e nenhum material de consulta ou pesquisa.

Devido a essa dificuldade, o manipulador ficou dividido em duas partes, a 1º parte sendo o punho (com 3 motores) e a 2º parte (com 3 motores) correspondente aos braços.

O projeto foi dividido em 3 etapas divididas da seguinte maneira, aquisição de informações, desenvolvimento e conclusão da 1º parte, e desenvolvimento da 2º parte.

Abaixo será descrito as etapas e seus respectivos anexos

➤ 1º Etapa:

- Verificação de contatos dos fios do manipulador;
- “Pinagem”[1] completa dos atuadores e potenciômetros de todo o manipulador, planilha em anexo;
- Verificação de funcionamento dos potenciômetros, definição das curvas de funcionamento;
- Desacoplamento dos atuadores e aquisição dos dados usados a partir de suas especificações[2];
- Não foi encontrada nenhuma especificação a respeito dos motores do punho, parâmetros que seriam fundamentais para os cálculos da dinâmica.

➤ 2º Etapa:

- Foi decidido trabalhar com o manipulador como dois sistemas, um correspondente ao Punho com 3 graus de liberdade e o outro referente aos Braços com 3 graus de liberdade e considerando toda a modelagem cinemática e dinâmica;
- Feito o software de controle do Punho, programado em Lua[3].;
- Feita a eletrônica e seu escopo para fabricação[4];
- Simulações e testes.

➤ 3º Etapa:

- CAD feito em SolidWork[5] do manipulador com suas respectivas aproximações de medidas (mm) e massa(g)[6];
- Modelagem do manipulador;
- Estudo teórico da modelagem cinemática e dinâmica com os parâmetros encontrados pelo SolidWorks;
- Simulação em Matlab.

6.1 Punho

O punho é responsável pela rotação dos 3 eixos finais, os eixos mais próximos da pinça. Eles fazem o movimento semelhante ao punho humano. É composto por três juntas muito próximas uma da outra, seus atuadores são três motores elétricos, que são alimentados²³ com uma tensão de até 5 Volts.

6.1.2 Sensor

O manipulador possui potenciômetros acoplados nos seus eixos, portanto, à medida que o eixo roda, a resistência no terminal 2 varia. Considerando o terminal 1 ligado em 3.3V e o terminal 3 no Terra, então temos que a tensão no terminal 2 varia de 0 a 3.3V.

Foi feita uma reta com os valores das tensões em função dos ângulos para cada motor. Ou seja, foram anotados os valores da tensão de cada ponto ao variar gradativamente de 0 grau a 180 graus, e depois foi encontrada a equação da tensão em função do ângulo.

²³ Por não possuir nenhuma especificação, foi estipulada uma tensão comum em Servo-Motores (4V à 6V).

6.1.3 Controle

Foi escolhido um controle do tipo PID, onde o “*set-point*” é o valor do ângulo desejado, que substituído na função descrita acima, é transformada em um valor de tensão de 0V a 3.3V. O canal A/D da placa lê o valor da tensão do terminal 2 e envia para o programa que o interpretava como o valor atual. O erro nada mais é que o valor do “*set-point*” menos o erro atual, que será tratado pelos Ganhos K_p , K_i e K_d que foram escolhidos na base de testes e melhor performance.

Depois de que o erro é tratado pelos ganhos o sinal é enviado para os motores em forma de PWM, através da função da biblioteca de Lua que gera o PWM²⁴ através de ciclos ativos. A função está pré-definida na biblioteca de Lua.

Para não ter problemas com valores absurdos na saída, são criados alguns mecanismos para limitar a tensão no motor, nesse caso, a tensão está limitada entre -5V até 5V, nunca ultrapassando essa faixa.

Para limitar que o erro não seja multiplicado por valores altos, devido aos termos da integral e derivada que lidam com o acumulo de erros passados, são criados filtros para essas situações.

- *Wind-Up* – Filtro para o termo da integral, ou seja, a integral só irá contribuir quando o valor acumulado do erro for menor que um valor limite. Impede que o termo da integral contribua com valores altíssimos quando o erro acumulado em um determinado tempo seja grande, por exemplo: quando um manipulador é impedido de fazer uma trajetória por um determinado motivo, enquanto não se move, ele está acumulando erros na integral, a partir do momento que puder se mexer, ele estará com um erro acumulado altíssimo, que resultaria em uma contribuição grande na saída.
- *Anti-Chattering* – Filtro para o termo da Integral, a derivada só irá contribuir quando a diferença dos seus erros for superior a $0.6 \cdot \text{Resolução do A/D}$. O filtro impede mudanças ocorridas por pequenas vibrações, ou modificações muito pequenas.

²⁴ $\text{Crtduty} = \text{math.floor}((\text{output}/5) * 99)$, gera um ciclo de 0 a 99%

6.1.4 Eletrônica

A eletrônica de potência foi feita com 5 microcontroladores, 2 Pontes H (L293E), 2 AND(7404), e 1 inversor(7408). Seu esquemático[4] está em anexo.

A entrada da eletrônica é composta pelo o sinal de PWM, por um *Bit* de direção que indica para que lado o motor irá girar, pelos pinos de Terra(GND), pelos pinos de alimentação do PIC (5V) e pelo pino de alimentação do motor. Esse pino, é responsável pelo valor de saída do PWM.

6.1.5 Resultados

Os resultados foram positivos, foi alcançado o objetivo de se alcançar os ângulos desejados na entrada, sem apresentar erros. Foi encontrado apenas um único problema, por ser um material antigo e sem manutenção, com algumas folgas nas caixas de redução, em certos momentos o equipamento travava, só voltando ao normal após reajustar as engrenagens.

A eletrônica foi suficiente para ativar os motores, que mesmo quando travados, não puxavam nenhuma corrente que queimasse os microcontroladores.

O programa em Lua foi programado desde o início para atender o manipulador com um todo, porém não foi feito o teste devido a modelagem do braço ter requerido um pouco mais de tempo.

6.2 Braço

O braço é responsável pelos movimentos semelhantes à rotação do tronco, ombro e antebraço. Para análise desses termos foi levado em conta sua estrutura, peso, inércia, sua modelagem por completo.

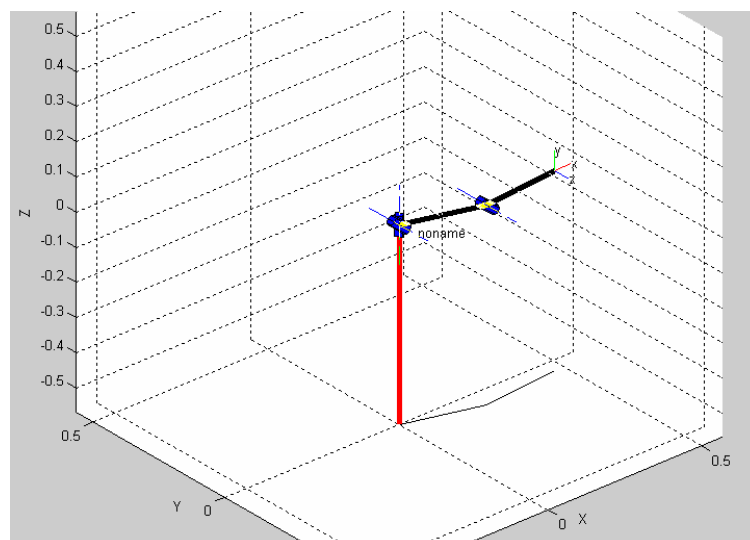
6.2.1 Modelagem

A modelagem começou a partir dos desenhos em SolidWorks²⁵ com os valores aproximados de medidas e massa. A partir dessas aproximações foi possível descobrir os momentos de inércia do manipulador, que serão usados para cálculo da dinâmica.

Primeiramente, a intenção era fazer as comparações dos cálculos feitos na mão com os resultados finais do Matlab, no entanto, isso iria gerar muitas contas e não seria o foco principal para ser mostrado nesse trabalho. Por isso, apenas será ilustrada a parte da modelagem, deixando os cálculos da cinemática e da dinâmica para o software.

Ao passar os parâmetros de Denavit-Hartenberg para o Matlab, ele gerou um desenho ilustrativo, que representa o MA2000. Através dos parâmetros, também é possível montar as matrizes A_i^{i-1} .

	alpha	A	theta	D
$DH =$	$pi/2$	0	θ_1	11
	0	13	θ_2	12
	0	15	θ_3	-14



²⁵ Programa de CAD(computer-aided design) de desenhos mecânicos

Figura 28

$$A_{01} := \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 0 \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{12} := \begin{bmatrix} \cos(\theta_2) & -\sin(\theta_2) & 0 & l_3 \cos(\theta_2) \\ \sin(\theta_2) & \cos(\theta_2) & 0 & l_3 \sin(\theta_2) \\ 0 & 0 & 1 & l_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{23} := \begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & l_5 \cos(\theta_3) \\ \sin(\theta_3) & \cos(\theta_3) & 0 & l_5 \sin(\theta_3) \\ 0 & 1 & 0 & -l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Para descobrirmos a Matriz da extremidade em relação a origem 1, basta fazer:

$$A_{03} = A_{01} * A_{12} * A_{23};$$

Fazendo a próxima multiplicação, poderemos verificar o valor do *end-effector*.

$$P = \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix}; \quad P_e = A_{03} * \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix};$$

$$P_e = \begin{bmatrix} l_3 * \cos(\theta_2) * \cos(\theta_1) + l_5 * \cos(\theta_2 + \theta_3) * \cos(\theta_1) + (l_2 - l_4) * \sin(\theta_1) \\ l_3 * \cos(\theta_2) * \sin(\theta_1) + l_5 * \cos(\theta_2 + \theta_3) * \sin(\theta_1) + (l_4 - l_2) * \cos(\theta_1) \\ l_1 + l_3 * \sin(\theta_2) + l_5 \sin(\theta_2 + \theta_3) \\ 1 \end{bmatrix}$$

→ *Pe* – *End-effector*, posição da extremidade em relação a origem 0.

Na cinemática direta, são passados os ângulos das juntas e é retornado o ponto no espaço que se encontra a extremidade. Ou seja, bastaria substituir os valores dos ângulos θ que a cinemática direta retornará o *End-effector*.

Através do Matlab é possível passar os ângulos e retornar a configuração final.

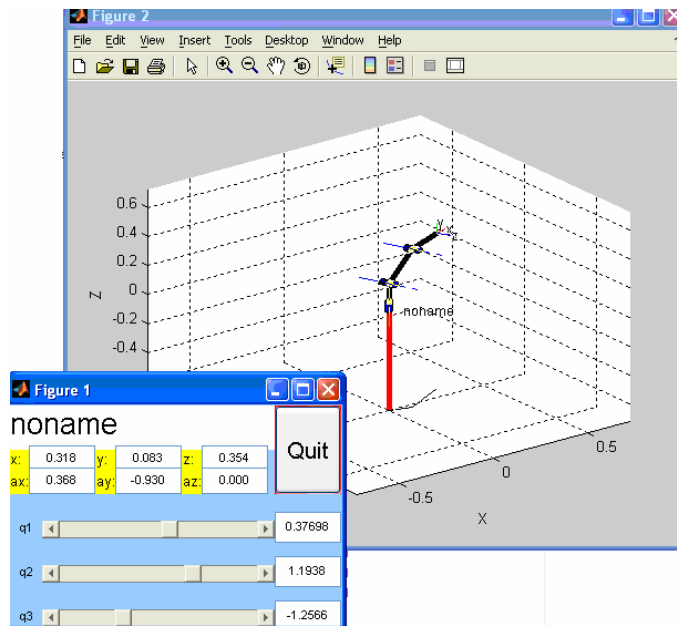


Figura 29

Para a parte da dinâmica foi passado todos os parâmetros de casa elo.

```

Link 1-----
alpha = 1.5708
A      = 0
theta  = 0
D      = 0.1555
sigma  = 0
mdh    = 0
offset = 0
m      = 4.0999
r      = 0.01556    0.03977    0.12414
I      = 0.018507  0.00017712  0.0019753
        0.00017712  0.014487    0.0016372
        0.0019753  0.0016372   0.013877
Jm     = 2e-010
G      = 1
B      = 1.48e-012
Tc     = 0  0
qlim   =
    
```

Tabela 2

```

Link 2-----
alpha = 0
A      = 0.23
theta = 0
D      = 0.068
sigma = 0
mdh   = 0
offset = 0
m     = 0.46623
r     = 0.10553   -0.00149       0
I     = 0.00047758 2.9178e-005   0
      2.9178e-005 0.0045122   -3e-011
      0          -3e-011   0.0041368
Jm    = 2e-010
G     = 1
B     = 8.17e-013
Tc    = 0 0
qlim  =

```

Tabela 3

```

Link 3-----
alpha = 0
A      = 0.2415
theta = 0
D      = -0.028
sigma = 0
mdh   = 0
offset = 0
m     = 1.0368
r     = 0.00041   0.04606   -3e-005
I     = 0.0014556 -0.0014347 -6.2415e-006
      -0.0014347 0.0073046 1.0713e-006
      -6.2415e-006 1.0713e-006 0.0084017
Jm    = 2e-010
G     = 1
B     = 1.38e-012
Tc    = 0 0
qlim  =

```

Tabela 4

A partir desses parâmetros foi possível calcular a dinâmica direta e a inversa.

6.2.2 Resultados

Infelizmente não foi possível implementar esses resultados na prática, devido ao tempo. No entanto, o material estará disponível para qualquer trabalho futuro em cima do Manipulador MA2000.

Contudo, o estudo da modelagem foi bastante interessante, por se tratar de uma matéria com certo grau de dificuldade e que não é muito explorada ainda nos cursos de graduação.

6.3 Trabalhos Futuros

Por se tratar de um tópico da robótica bastante interessante e de extrema importância para desenvolvimento de robôs, pretende-se implementar o manipulador com os dois sistemas no futuro. O próximo passo seria a programação de uma rotina de trabalho em um espaço operacional. Com êxito nesses passos a última idéia seria fazer o controle através de um controle de vídeo game do Nintendo Wii.

BIBLIOGRAFIA

Meggiolaro, MA. Controle de Sistemas Robóticos. **Departamento de Engenharia Mecânica**. Pontifícia Universidade Católica – Rio de Janeiro.

Asada, HH. Rotic. 2.12. Lecture Notes. **Mechanical Engineering**. (2005).

Ogata, K. **Engenharia de Controle Moderno**. Universidade de Minnesota.

Angeles, J. **Fundamentals of Mechanical Systems**. (2 edn)

Graig, J. J. **Introduction to Robotics: Mechanics and Control**, (2 edn), Mac Graw Hill, (1989).

Rosário, J. M. **Princípios de Mecatrônica**, (1 edn,) Prentice Hall, 2005.

Zwirtes, R. Cinemática Inversa para Controle da Abordagem de Órgãos Terminais de Robôs Manipuladores. **Ciência da Computação**. Universidade do estado de Santa Catarina (2004)

Ormiga, AGB. Controle de Robô Usando Técnicas Inteligentes. **Departamento de Engenharia de Eletrônica e de Telecomunicações**. Universidade do Estado do Rio de Janeiro (Julho 2008)

Acesso em 24 de outubro a revista eletrônica Mecatrônica Fácil, disponível em www.mecatronicafacil.com

Acesso em 22 de outubro a revista eletrônica Mecatrônica Atual, disponível em www.mecatronicaatual.com

eLua Project. <<http://eluaproject.net>>

Ierusalimschy, Roberto. Celes, Waldemar. Figueiredo, L. Henrique. **Lua 5.1 Reference Manual**. 1.^a edição. Rio de Janeiro 2006.

<<http://www.lua.org>>

WoW Wiki - Lua. <<http://www.wowwiki.com/Lua>>

Bittencourt. P. VHeWs: Uma arquitetura para redes de sensores via “Web”. **Engenharia de Controle e Automação**. Pontifícia Universidade Católica-Rio de Janeiro (Jul ho-2009)

The internet Society. **Hypertext Transfer Protocol**. <<http://www.w3.org/Protocols>>

Meggiolaro. Marco. **RioBotz Combat Tutorial**. (Version 2.0 – March 2009)
<<http://www.riobotz.com.br>>

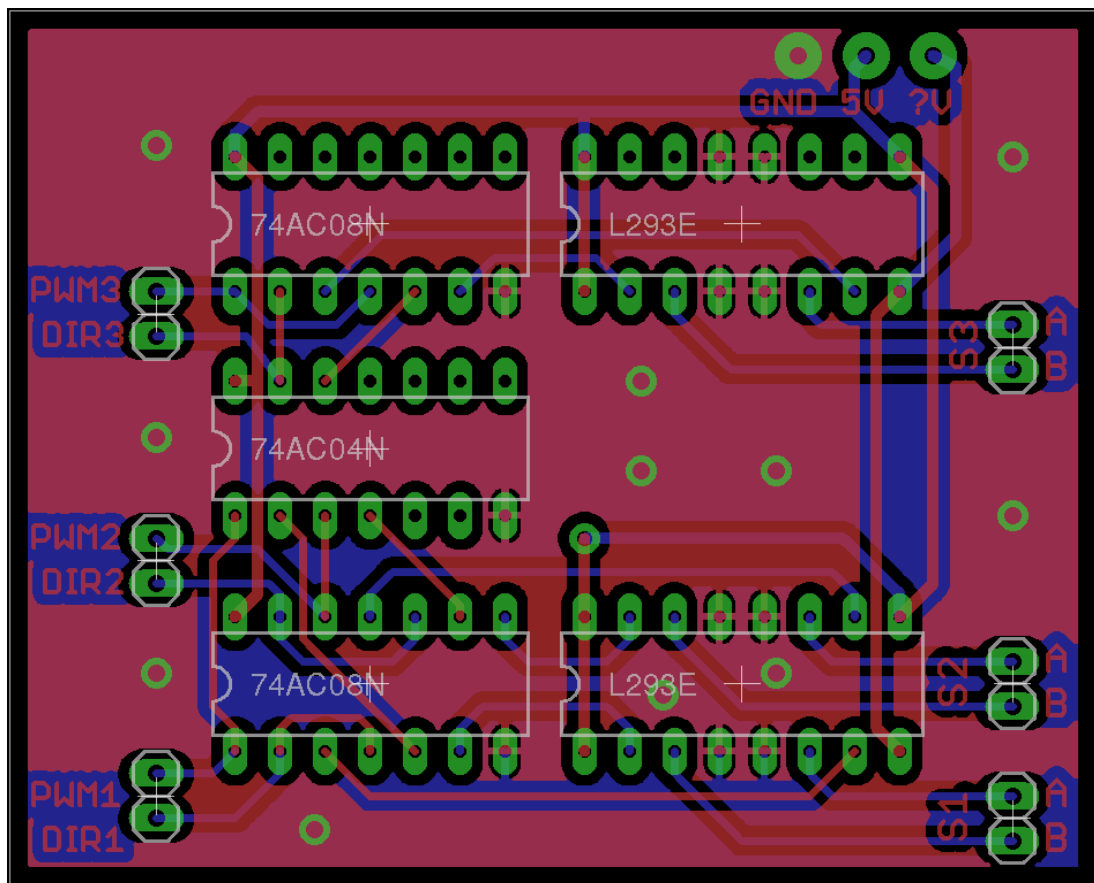
Nakamura,M. Goto,S. Kyua,N –**Mechatronic Servo Sysyems Control**

Anexos

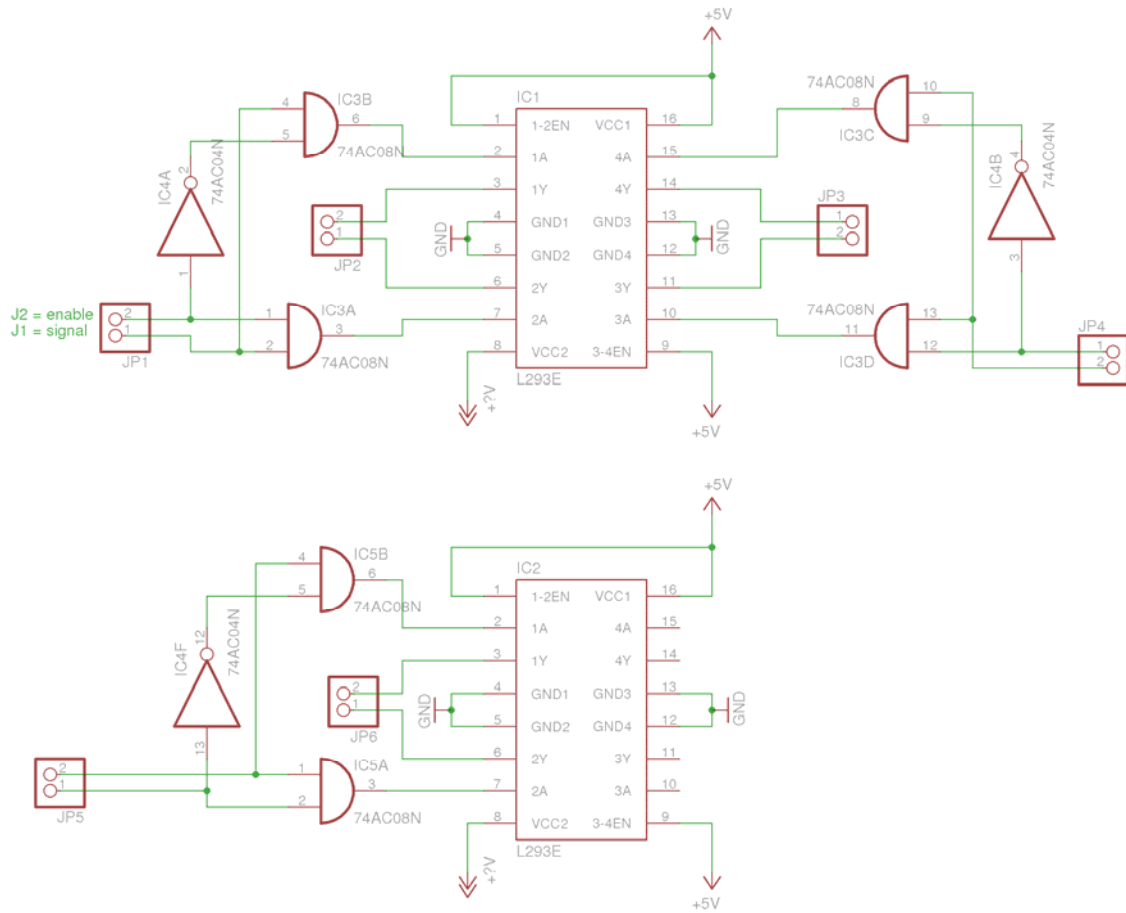
[1] - “Pinagem”

	Cor do fio	Pinagem/Posição(D-E)	Conector	Especificação
Motor 1 (base)	(+) Branco	3	A	Maxon
	(-) Preto e Vermelho	2	A	
Encoder 1	Laranja e Vermelho	9	B	
	Preto e Branco	1/1 Vermelho	B	
	Preto e Laranja	7	B	
Motor 2	(+) Preto e Rosa	7	A	Escop
	(-) Verde e Vermelho	8	A	
Encoder 2	Laranja e Preto	7	B	
	Amarelo e Vermelho	2/2 Azul	B	
	Laranja e Vermelho	9	B	
Motor 3	(+) Verde	12	A	Maxon
	(-) Roxo	11	A	
Encoder 3	Azul	8	B	
	Branco e Vermelho	3/3 Verde	B	
	Laranja	10	B	
Servo 1 (Pinça)	(+) Cinza	24	A	
	(-) Cinza	23	A	
Potenciômetro1	Azul	8/8 Roxo	B	1 k ohm
	Roxo	6/6 Preto	B	
	Laranja	10/10 Rosa	B	
Servo 2	(+) Verde	15	A	
	(-) Amarelo	16	A	
Potenciômetro2	laranja	10	B	
	Azul	8	B	
	Amarelo	4	B	
Servo 3	(+) Verde e Vermelho	19	A	
	(-) vermehlo	20	A	
Potenciômetro3	Laranja	10	B	
	Azul	8	B	
	Marron	5	B	

[2] – Eletrônica de Potência da Pinça



[3] – Esquemático da Eletrônica de Potência da Pinça



[4] – Programa em Lua

```
-----  
-- Trabalho final do Aluno Thiago Tavares Pimenta  
-- Curso de Engenharia de Controle e Automação  
  
-- Controle do Manipulador Robótico MA2000  
-- VERSÃO 2.2                               Agosto 2009 @ LED Lab PUC-Rio  
  
-- Colaboradores : Téo Benjamin, Ives Negreiros, Pedro Bittencourt,  
-- Lablua  
  
-- Controle PID para o manipulador de 6 graus de liberdade.  
-- Início da implementação genérica para vários motores:  
-- Teste com pinça - OK!  
-- Teste com braço - Não testado!!  
-----  
  
-- 0 no pino de direção anda para frente e 1 anda para trás.  
  
local motor = {}  
  
motor[1] = {}  
motor[2] = {}  
motor[3] = {}  
  
--[  
motor[4] = {}  
motor[5] = {}  
motor[6] = {}  
--]  
  
motor[ 1 ].angle = false  
motor[ 1 ].Kp = 4  
motor[ 1 ].Ki = 0.008  
motor[ 1 ].Kd = 150  
motor[ 1 ].integral = 0  
motor[ 1 ].previous_error = 0  
motor[ 1 ].pwmid = 0  
motor[ 1 ].adcid = 0  
motor[ 1 ].dirpin = pio.PC_5  
motor[ 1 ].pwm_threshold = 25  
motor[ 1 ].min_angle = 0  
motor[ 1 ].max_angle = 180  
motor[ 1 ].integral_threshold = 1  
motor[ 1 ].Chattering = ( 0.6* Resolução A/D)  
motor[ 1 ].div_tensao = 3.3 / 5.27  
-- divisor de tensao, Vin/Resistencia (R1+R2)  
motor[ 1 ].Res1 = ( 0.0202 * motor[ motorid ].angle + 0.9382 )  
-- equação da resistencia do potenciometro no pino 2
```

```

motor[ 2 ].angle = false
motor[ 2 ].Kp = 4
motor[ 2 ].Ki = 0.005
motor[ 2 ].Kd = 200
motor[ 2 ].integral = 0
motor[ 2 ].previous_error = 0
motor[ 2 ].pwmid = 4
motor[ 2 ].adcid = 1
motor[ 2 ].dirpin = pio.PC_7
motor[ 2 ].pwm_threshold = 30
motor[ 2 ].min_angle = 30
motor[ 2 ].max_angle = 150
motor[ 2 ].Chattering = ( 0.6* Resolução A/D)
motor[ 2 ].integral_threshold = 1
motor[ 2 ].div_tensao = 3.3 / 5.27
-- divisor de tensao, Vin/Resistencia (R1+R2)
motor[ 2 ].Res1 = ( 0.0202 * motor[ motorid ].angle + 0.9382 )
-- equação da resistencia do potenciometro no pino 2

motor[ 3 ].angle = false
motor[ 3 ].Kp = 4
motor[ 3 ].Ki = 0.008
motor[ 3 ].Kd = 150
motor[ 3 ].integral = 0
motor[ 3 ].previous_error = 0
motor[ 3 ].pwmid = 2
motor[ 3 ].adcid = 2 -- verificar canal
motor[ 3 ].dirpin = pio.PF_1 -- IDX1
motor[ 3 ].pwm_threshold = 25
motor[ 3 ].min_angle = 0
motor[ 3 ].max_angle = 180
motor[ 3 ].integral_threshold = 1
motor[ 3 ].Chattering = ( 0.6* Resolução A/D)
motor[ 3 ].div_tensao = 3.3 / 5.27
-- divisor de tensao, Vin/Resistencia (R1+R2)
motor[ 3 ].Res1 = ( 0.0202 * motor[ motorid ].angle + 0.9382 )
-- equação da resistencia do potenciometro no pino 2

--[
motor[ 4 ].angle = false
motor[ 4 ].Kp =
motor[ 4 ].Ki =
motor[ 4 ].Kd =
motor[ 4 ].integral = 0
motor[ 4 ].previous_error = 0
motor[ 4 ].pwmid =
motor[ 4 ].adcid =
motor[ 4 ].dirpin =
motor[ 4 ].pwm_threshold =
motor[ 4 ].min_angle =
motor[ 4 ].max_angle =
motor[ 4 ].integral_threshold =
motor[ 4 ].div_tensao = 3.3 / 2.00
-- divisor de tensao, Vin/Resistencia (R1+R2)
motor[ 4 ].Res1 = ( 0.0073 * motor[ motorid ].angle + 0.3803 )
-- equação da resistencia do potenciometro no pino 2

```

```

motor[ 5 ].angle = false
motor[ 5 ].Kp =
motor[ 5 ].Ki =
motor[ 5 ].Kd =
motor[ 5 ].integral = 0
motor[ 5 ].previous_error = 0
motor[ 5 ].pwmid =
motor[ 5 ].adcid =
motor[ 5 ].dirpin =
motor[ 5 ].pwm_threshold =
motor[ 5 ].min_angle =
motor[ 5 ].max_angle =
motor[ 5 ].integral_threshold =

motor[ 6 ].angle = false
motor[ 6 ].Kp =
motor[ 6 ].Ki =
motor[ 6 ].Kd =
motor[ 6 ].integral = 0
motor[ 6 ].previous_error = 0
motor[ 6 ].pwmid =
motor[ 6 ].adcid =
motor[ 6 ].dirpin = pi*PC_
motor[ 6 ].pwm_threshold =
motor[ 6 ].min_angle =
motor[ 6 ].max_angle =
motor[ 6 ].integral_threshold =
--]]

```

```

local tmrid = 0

```

```

function PID ( angle, motorid )

```

```

    if not angle then
        return 0
    end

```

```

    if angle < motor[ motorid ].min_angle then
        angle = motor[ motorid ].min_angle
    elseif angle > motor[ motorid ].max_angle then
        angle = motor[ motorid ].max_angle
    end

```

```

local setpoint = ( 0.0073 * angle + 0.3803 ) * motor[ motorid ].div_tensao
-- Entrada do angulo e conversao em tensao para entrada, k = Vin/(R1+R2)
--print( "Setpoint = ", setpoint )

```

```

        if adc.isdone( motor[motorid].adcid ) == 1 then
            adc.sample( motor[motorid].adcid, 32 )
        end

local voltage_value = adc.getsample( motor[motorid].adcid ) or 0
voltage_value = ( voltage_value/adc.maxval( motor[motorid].adcid ))*3.3
--print("Vin = ", Voltage_Value )
--print("Angulo = ", ((Voltage_Value / k) - 0.9382)/0.0202)
--print( "vin = ", voltage_value )

local error = setpoint - voltage_value
-- diferença entre setpoint e da saída em Volts
--print( "Erro ".motorid, error )

motor[ motorid ].integral = motor[ motorid ].integral + error
-- Somatorio dos erros
    if
        motor[ motorid ].integral > motor[ motorid ].integral_threshold then
            motor[ motorid ].integral = motor[ motorid ].integral_threshold

        elseif
            motor[ motorid ].integral < -motor[ motorid ].integral_threshold then
                motor[ motorid ].integral = -motor[ motorid ].integral_threshold
    end
-- WIND-UP, filtro para o termo da integral

local derivative = error - motor[ motorid ].previous_error
    if
        motor[ motorid ].previous_error > motor[ motorid ].chattering then
            motor[ motorid ].previous_error = motor[ motorid ].chattering

        elseif
            motor[ motorid ].previous_error < -motor[ motorid ].chattering then
                motor[ motorid ].previous_error = -motor[ motorid ].chattering
        else
            motor[ motorid ].previous_error = error
    end
-- ANTI-CHATTERING, filtro para o termo da derivada

--print("P, I, D = ", ( (Kp*Error) + (Ki*integral) + (Kd*derivative) ))
return ( ( motor[ motorid ].Kp * error
          ( motor[ motorid ].Ki * motor[ motorid ].integral ) +
          ( motor[ motorid ].Kd * derivative )
        )
    )
end
-- Somatorio dos termos

function generatePWM( output, motorid )
--print("Output = ", output)
    if output > 5 then
        output = 5
    elseif output < -5 then

```

```

        output = -5
end

local crtduty = math.floor( ( output / 5 ) * 99 )
--print( "Ciclo Real: ", crtduty, dir )

    if crtduty < -4 then
    if crtduty > -motor[ motorid ].pwm_threshold then
        crtduty = -motor[ motorid ].pwm_threshold
    end

pio.pin.setval( 1, motor[ motorid ].dirpin )
crtduty = -crtduty
    elseif crtduty > 4 then
    if crtduty < motor[ motorid ].pwm_threshold then
        crtduty = motor[ motorid ].pwm_threshold
    end

pio.pin.setval( 0, motor[ motorid ].dirpin )
    else
crtduty = 0
    end

--print( "crtduty "..motorid, crtduty )
--print( "dir "..pio.pin.getval( motor[ motorid ].dirpin ) )
pwm.setup( motor[ motorid ].pwmid, 25000, crtduty )
end

function Init()
    for motorid, _ in pairs( motor ) do
        print(motorid)
        pwm.setclock( motor[ motorid ].pwmid, 25000000 )
        pwm.setup( motor[ motorid ].pwmid, 25000, 0 )
        pwm.start( motor[ motorid ].pwmid )
        pio.pin.setdir( pio.OUTPUT, motor[ motorid ].dirpin )
--- Configurar o A/D
        adc.setclock( motor[ motorid ].adcid, 0, tmrid )
        adc.setblocking( motor[ motorid ].adcid, 1 )
        adc.setsmoothing( motor[ motorid ].adcid, 32 )
    end
end

Init()

while true do
    for motorid, _ in pairs( motor ) do
        generatePWM( PID( false ), motorid )
        motor[ motorid ].integral = 0
        motor[ motorid ].previous_error = 0
    end
end

--[ print"Kp = "
    Kp = tonumber( io.read() )
    print"Ki = "
    Ki = tonumber( io.read() )
    print"Kd = "
    Kd = tonumber( io.read() )
--]]

    for motorid, _ in pairs( motor ) do

```

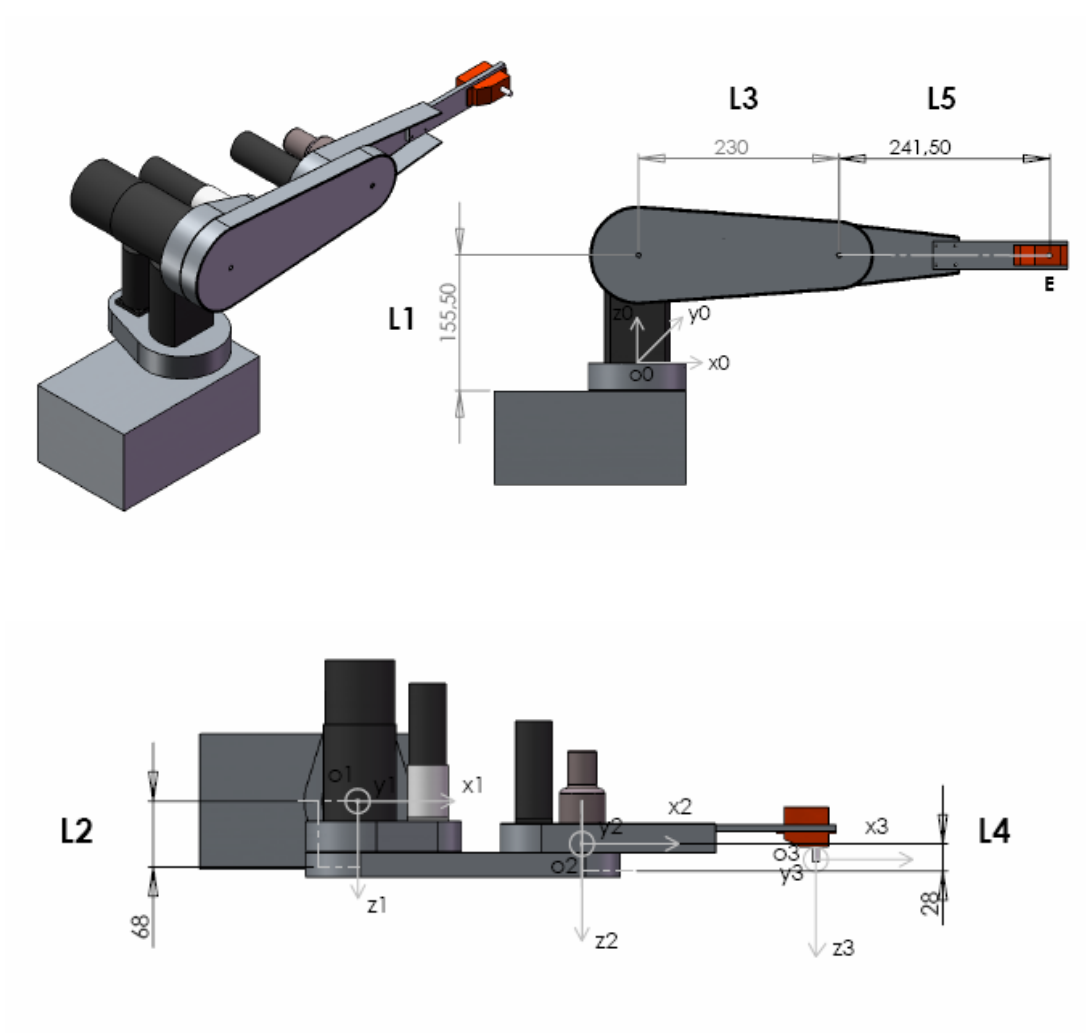
```

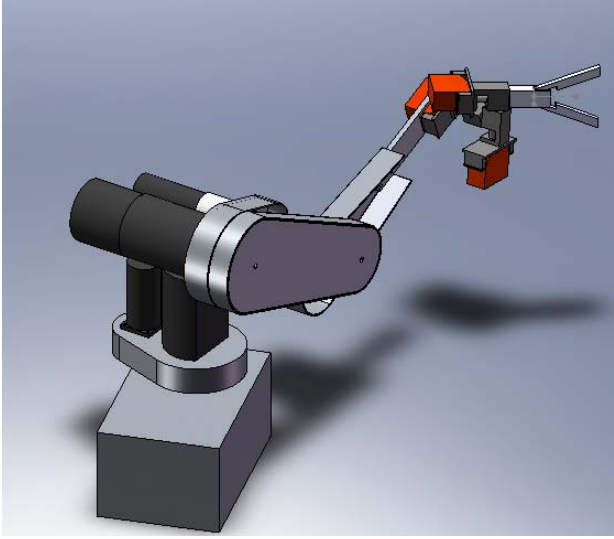
        print( "Angle for motor " .. motorid .. ":" )
        motor[ motorid ].angle = tonumber( io.read() )
    end

    while uart.getchar( 0, uart.NO_TIMEOUT ) == "" do
-- Loop para esperar comando do teclado
        for motorid, _ in pairs( motor ) do
            generatePWM( PID( motor[ motorid ].angle, motorid ), motorid ) --
Gera os sinais de PWM
        end
    end
end
end

```

[5] – Modelagem





[6] – Medidas

	MA2000 (real)_ (g)	SolidWorks MA2000_ (g)	
Elos 2,3,4,5,6	1984	1805.81	
Braço_Tampa	224	225.91	
Braço	243	240.32	
Anti-braço+Servo2	1165	1036.81	
Elos 4,5,6	352	302.76	
ERRO	8,90%		
massa concentrada no elo 3	352		
Centro de Massa (mm)	X	Y	Z
Anti-Braço (origem2)	0.41	46.06	(-)0.03
Braço (origem1)	105.53	(-)1.49	0
Base (origem 0)	18.04	45.09	130.91