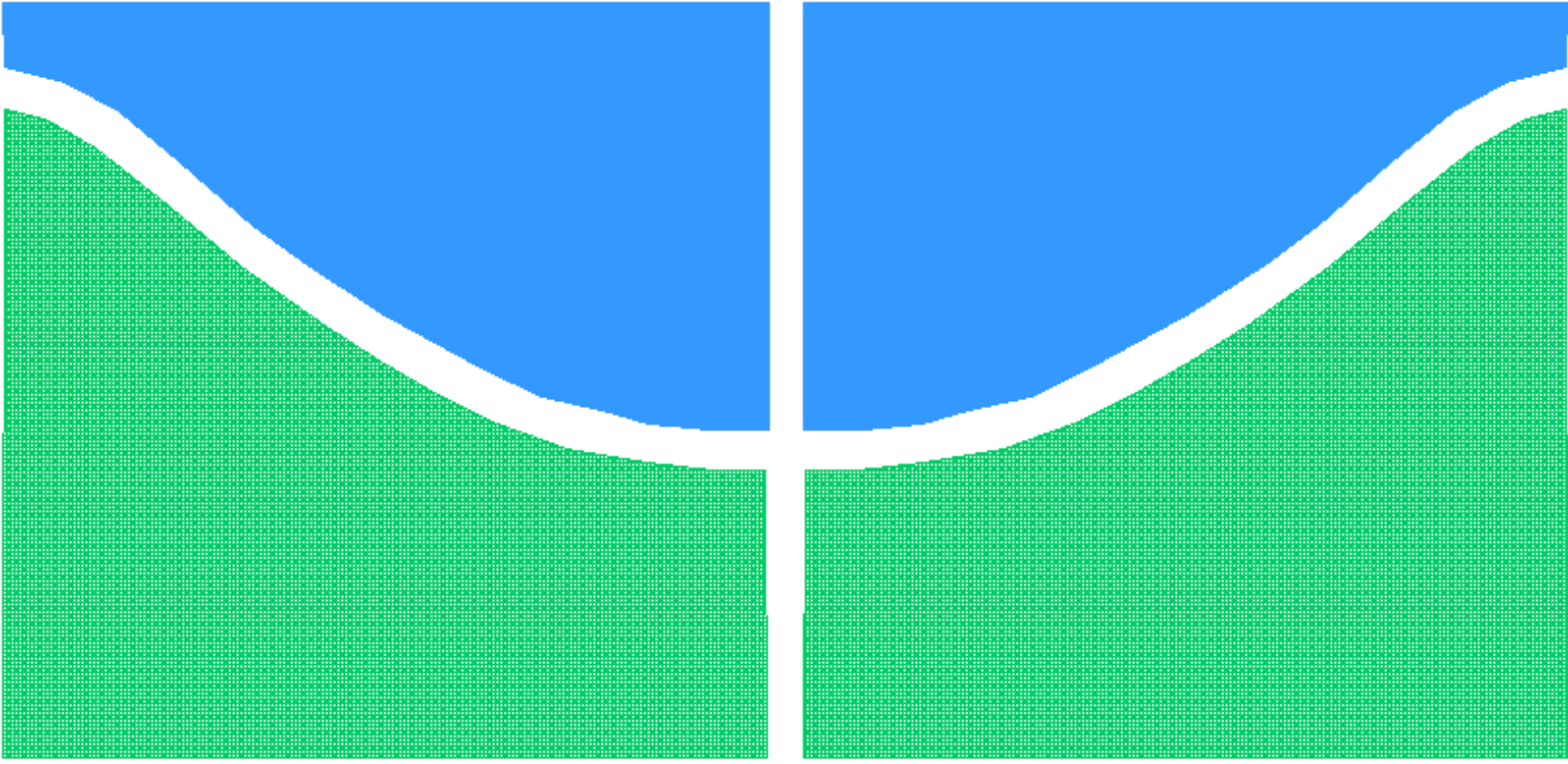


TRABALHO DE GRADUAÇÃO

REALIZAÇÃO DE UMA PLATAFORMA PARA ESTUDO DE ROBÓTICA COMPORTAMENTAL BASEADA EM QUADRÚPEDES

Gustavo Henrique Cotta
Laurindo Raulino Neto

Brasília, dezembro de 2006

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

REALIZAÇÃO DE UMA PLATAFORMA PARA ESTUDO DE ROBÓTICA COMPORTAMENTAL BASEADA EM QUADRÚPEDES

Gustavo Henrique Cotta

Laurindo Raulino Neto

Relatório submetido como requisito parcial para obtenção
do grau de Engenheiro de Mecatrônica

Banca Examinadora

Prof. Geovany Araújo Borges, UnB/ ENE (Orientador) _____

Prof. Adolfo Bauchspiess, UnB/ ENE _____

Prof. Lélio Ribeiro Soares Jr., UnB/ ENE _____

Dedicatórias

Dedico este trabalho à todos os que tornaram possível essa grande conquista, em especial aos meus pais, que demonstraram apoio incondicional.

Laurindo Raulino Neto

Não apenas esse projeto, mas todas as minhas conquistas profissionais são dedicadas aos meus pais, Geralda e Luiz, que em nenhum momento deixaram de me apoiar e de me incentivar, e ao meu irmão, Leandro, por ter me servido de referência como pessoa e profissional.

Gustavo Henrique Cotta

Agradecimentos

Primeiramente aos meus pais, Geralda e Luiz, e irmãos, Leandro e Ana, que sempre me ajudaram e me apoiaram durante ao longo do curso.
Ao meu amigo e companheiro de projeto, Laurindo, por ter dividido comigo todas as dificuldades encontradas no desenvolvimento deste trabalho e ao longo do curso.
Aos meus amigos de curso que tornaram muito mais fácil o caminho até a graduação, em especial o próprio Laurindo, o Maurílio, o Alexandre e o Rodolfo, por estarem sempre presentes.
Ao professor-orientador Geovany, por todo o conhecimento a nós transmitido, sempre com muita paciência e atenção.
Aos colegas de laboratório por toda a ajuda a mim prestadas, em especial o Eng. Alexandre Martins, sempre disposto a ajudar a solucionar os mais diversos problemas encontrados ao longo do projeto.
Aos técnicos da oficina mecânica, SG-09, em especial o Pereira, pela grande ajuda prestada no desenvolvimento do projeto.
E por fim, à todos aqueles que de certa forma influenciaram, participaram e colaboraram com este projeto.

Gustavo Henrique Cotta

Primeiro agradeço a meus familiares pela paciencia e apoio necessário para superar dificuldades e seguir em frente.
Não poderia deixar de agradecer todos os professores que contribuiram de alguma forma para minha formação, em especial ao professor Geovany por todo apoio e orientação dada a este trabalho.
Agradeço a meus amigos que estiveram sempre comigo dividindo dificuldades ao longo do curso, em especial ao Gustavo Cotta.
E, por fim, agradeço aos funcionários da oficina mecânica do SG-9 que sem eles não teriamos concluido esse projeto a tempo.

Laurindo Raulino Neto

RESUMO

O presente trabalho apresenta a realização de uma plataforma para o estudo da robótica terrestre e comportamental baseada em quadrúpedes. O projeto se baseou em um sistema constituído por servo motores, os mesmos utilizados em aeromodelismo, módulos de acionamento micro-controlados com reguladores de alta potência e um módulo de controle utilizando o PC via comunicação serial RS-485. Esse sistema foi o que se mostrou mais prático de se realizar dentre as opções anteriormente sugeridas. Para executar o projeto foi realizada uma simulação da cinemática direta e inversa do quadrúpede com auxílio do MatLab, sendo feita uma animação dos movimentos para analisar os resultados da simulação. Foram realizados testes de cinemática inversa com diferentes métodos para que pudéssemos chegar ao que melhor atendesse a proposta.

Palavras-chaves: quadrúpede, Denavit-Hartenberg, servo motor, cinemática direta, cinemática inversa, comunicação RS-485.

ABSTRACT

This manuscript describes the development of a platform for a study on Robotic Behaviour based on quadrupeds. The project was based on a system that was built with aero models servo motors, micro-controlled drivers with voltage power regulators, and a control module using PC through RS-485 serial communication. This system was the one that had shown the best results compared to the others suggested before. For this project, it was done a simulation of the quadruped direct and inverse kinematics assisted with the MatLab software, in which an animation was designed to help in the analyzis of the simulation results. Some tests with different methods of inverse kinematics have been made to choose the one that fits the project goals.

Key-words: quadruped, Denavit-Hartenberg, Servo RC, direct kinematics, inverse kinematics, RS-485 serial communication.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIMBOLOS

Símbolos Latinos

O	Origem dos sistemas de coordenadas	
T	Torque	[N· m]

Símbolos Gregos

Δ	Variação da grandeza
∂	Derivada parcial
λ	Constante de *damping*
$\Re$	Conjunto dos números Reais

Subscritos

I/O	Entrada e saída
o	Saída
in	Entrada

Sobrescritos

$\dot{\cdot}$	Variação temporal
$\overline{\quad}$	Lógica de negação
$\dagger$	Pseudoinversa da matriz
-1	Inversa da matriz
r	Posto ou *rank* da matriz
T	Transposta da matriz

Siglas

D-H	Denavit-Hartenberg
LARA	Laboratório de Robótica e Automação
PWM	Pulse-width Modulation
RC	Rádio controlado
SVD	Singular Value Decomposition
ULA	Unidade Lógica Aritimética

1 INTRODUÇÃO

Nos últimos anos o estudo de robôs com pernas vêm aumentando significativamente e ganhando importância dentro da robótica terrestre. Nessa linha de pesquisa o Laboratório de Robótica e Automação (LARA) do departamento de Engenharia Elétrica da Universidade de Brasília também iniciou trabalhos.

Outra vertente a ser explorada com o desenvolvimento deste trabalho é o auxilio ao estudo da robótica comportamental, uma vez que o quadrúpede desenvolvido servirá de base para o estudo comportamental de *pet robots*, ou robôs afetivos, capazes de interagir com o ser humano simulando estados emocionais e comportamentais de um determinado animal de estimação. O projeto iniciou com a proposta da concepção de um robô-cachorro e então foi substituída para o desenvolvimento de um robô-tamanduá.

Neste trabalho foi realizada a concepção/construção de um robô quadrúpede. Para isto foi desenvolvido o modelo cinemático do robô que nos permitiu, a elaboração de uma simulação em matlab capaz de realizar o posicionamento das pernas do robô nas posições desejadas e então sua devida implementação no protótipo construído.

Também foi objeto do trabalho tornar o robô capaz de atuar de acordo com comandos emitidos por outros dispositivos conectados ao seu barramento de comunicação, possibilitando uma integração entre o trabalho aqui realizado com o trabalho realizado por outras equipes, como por exemplo a equipe responsável pelo estudo da robótica comportamental.

1.1 CONTEXTUALIZAÇÃO

A robótica móvel é um campo que está sob grande desenvolvimento e vem sendo largamente estudada nos últimos anos. Seus estudos tiveram inicio com robôs que se locomoviam sobre rodas e foi se diversificando cada vez mais, passando a atuar em robótica aérea, com o uso de modelos de helicópteros e aviões, e na própria robótica terrestre com a utilização de veículos que se locomovem com pernas, sendo este último objeto do nosso estudo.

A grande motivação para tal diversificação foi devido à limitação que os veículos que se utilizam de rodas para se locomover possuem, uma vez que, só são capazes de se locomover em terrenos com um certo grau de regularidade. Para tais veículos existe uma certa dificuldade para transpor obstáculos, como descer e subir escadas, entre outros.

Robôs que se utilizam de pernas para locomoção, em relação a robôs com rodas, é que o último deve estar em contato diretamente com o plano/superfície sobre a qual ele irá se locomover, já robôs dotados de pernas não sofrem com esta restrição, uma vez que são capazes de "escolher"onde vão posicionar os pés. Robôs com rodas não são capazes de se isolar das irregularidades dos terrenos, apesar do grande avanço dos sistemas de suspensão. E já robôs com pernas são capazes de isolar o caminho percorrido pela perna do caminho percorrido pelo robô.

1.1.1 Breve histórico

Apresenta-se abaixo um breve histórico da evolução do estudo de robôs com pernas pela robótica móvel:

- Sistema Articulado, desenvolvido por Chebychev[2]

  Criado em 1870 surge o primeiro modelo de uma estruturas com pernas utilizando o sistema artic-

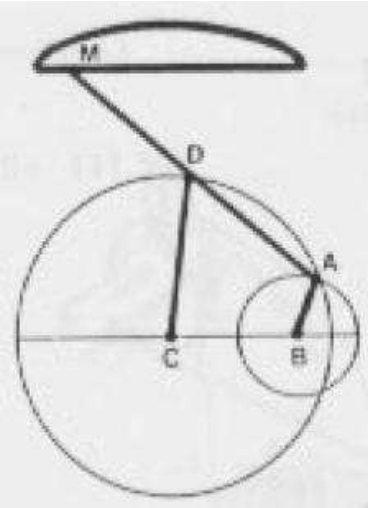


Figura 1.1: Sistema Articulado de Chebychev

ulado desenvolvido por Chebychev, mostrado na figura 1.1, permitia o corpo andar apenas em linha reta. Em meados dos anos 50, já no século XX, concluiu-se que veículos com esse tipo de articulação não seriam alternativas aos veículos que utilizavam rodas. Desta forma chegaram a conclusão de que tais veículos deveriam ser controlados, não tendo mais a necessidade de se seguirem movimentos criados apenas pelas articulações.

- Caminhão da General Eletric

  A primeira abordagem que se tem notícia veio com a construção de um caminhão com quatro pernas, criado por Ralph Moster, da General Eletric, baseado em um caminhão desta empresa, figura 1.2. O controle era realizado por alavancas e pedais que eram manipulados por humanos. Após algum tempo Ralph Moster foi capaz de conduzir o veículo com uma certa destreza.

  Porém o operador humano deveria ser bastante especializado, e somente operar o veículo em terrenos simples pois ele não contava com o auxílio de computadores. Foi com a chegada de computadores, no fim dos anos sessenta, que esse controle passou a ser possível.

- *Phoney Poney*

  O primeiro veículo a ser completamente controlado por computador surge em 1966, um quadrúpede chamado de "Phoney Poney", também baseado na arquitetura do caminhão da General Eletric. Construído por McGhee e Frank na Universidade do Sul da Califórnia.

  Um ano mais tarde surge na Rússia o primeiro robô com seis pernas, construído por Okhotsimski. Este número de pernas proporciona uma ótima relação entre estabilidade e complexibilidade, uma vez que o problema com o equilíbrio do robô pode ser facilmente garantido mantendo-se quatro patas no chão enquanto outras duas realizam o movimento.

- *OSU Hexapod*

  No mesmo ano do desenvolvimento do primeiro Hexapod, McGhee lança o OSU Hexapod, figura 1.3, similar ao robô russo, este também era controlado por um computador que realiza o cálculo das equações cinemáticas para controle dos desoito motores elétricos nele presente.

  Uma outra abordagem foi introduzida por Hirose, que aliou o desenho de articulações com o poder de cálculo dos computadores, o que melhorou muito a eficiência de tais sistemas.

  Raibert e Sutherland construíram, em 1983, o primeiro robô com seis pernas com um micro-controlador embarcado. Este foi o primeiro robô completamente independente, uma vez que ainda era dotado de um motor a gasolina, e era capaz de levar a bordo um homem.

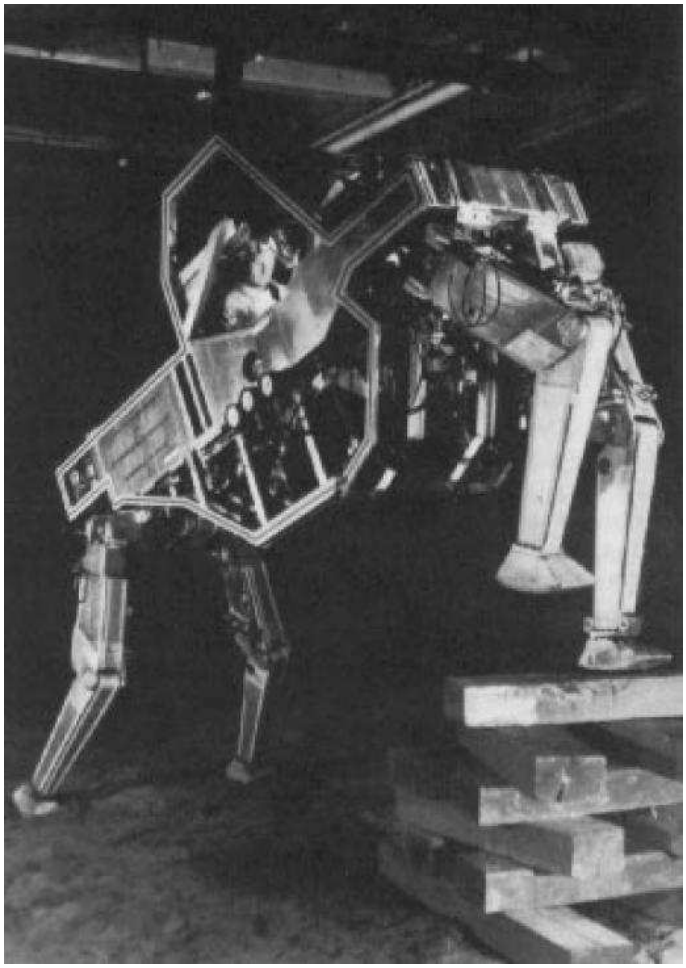


Figura 1.2: Caminhão da General Eletric

- *Plustech*

  Atualmente já existem alguns exemplares de robôs com pernas, o *Plustech*, figura 1.4, por exemplo é utilizado para deslocamento em áreas de floresta nas quais a utilização de tratores destroi o ambiente na medida que vai se locomovendo.

- *ROBOT III*

  A sinergia entre a robótica e a biologia deu origem ao ROBOT III, um robô criado com o intuito de imitar os movimentos de uma barata. Trata-se de um robô com seis pernas cujos estudos cinemáticos foram baseados na barata *blaberus discoidalis*, figura 1.5.

- *Mechant*

  Outro desenvolvimento atual, realizado na Universidade de Helsink, na Finlândia, é o Mechant (*Mechanical Ant*), figura 1.6.

  Já na área de desenvolvimentos de *pet-robots* que seriam robôs de "estimação", como robôs cachorros dentre outros, temos alguns representantes que serviram de incentivo para a concepção da plataforma que aqui será apresentada levando em consideração a atuação na área da robótica comportamental.

- *JoinMax*

  O JoinMax da MCII Robot é um pequeno robô dotado de servo motores que é capaz de realizar movimentos pré-configuráveis pelo usuário. Sua estrutura física, e limitações nos ângulos das juntas inspiraram a construção do quadrúpede desenvolvido neste projeto. A figura 1.7 mostra tal robô.

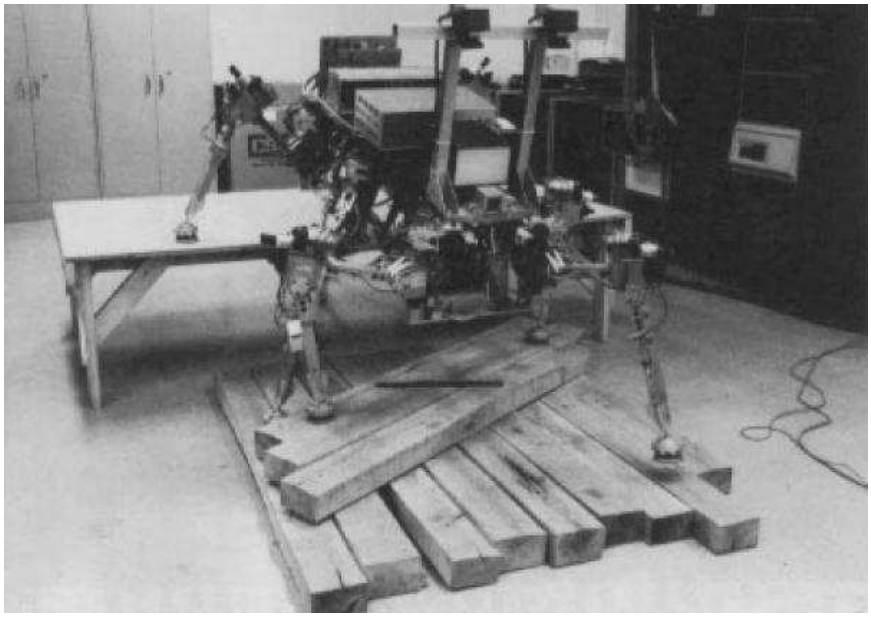


Figura 1.3: OSU Hexapod

- *Aibo*

  O Aibo da Sony é um belo exemplar de atuação da robótica comportamental na concepção de *pet-robots*, ele também tem como base um cachorro e o mesmo interage com o usuário de diversas maneiras, alterando seu estado emocional. A figura 1.8 mostra o Aibo modelo ERS-7M3 da Sony.

Diversos outros projetos atuam no campo de desenvolvimento de *pet-robots*, e este trabalho também visa o início do desenvolvimento de projetos nessa linha de pesquisa.

1.2 DEFINIÇÃO DO PROBLEMA

Conceber uma plataforma de estudos na área da robótica terrestre na área de robôs dotados de pernas bem como apoiar projetos de robótica comportamental. Para isso é necessário a obtenção da cinemática de um quadrúpede, concepção mecânica da plataforma para o robô e prover a capacidade de intercomunicação entre as duas áreas da robótica em estudo.

1.3 OBJETIVOS DO PROJETO

O trabalho proposto tem como objetivo a iniciação do LARA no estudo em robótica móvel terrestre utilizando robôs com pernas, bem como servir de apoio ao estudo em robótica comportamental, oferecendo uma plataforma capaz de mimetizar a estrutura física de animais quadrúpedes em geral. Com este fim está em desenvolvimento a plataforma de um quadrúpede, dotado de servo-motores que são responsáveis por realizar o movimento das juntas das pernas do robô. Sendo assim fez-se necessário a obtenção de um modelo cinemático para o quadrúpede, possibilitando obter o posicionamento do *end-efector*, extremidade das pernas, dado uma certa configuração de ângulo para as juntas das pernas e também obter a configuração de ângulos necessária para posicionar a extremidade da pata em uma posição específica. Para validação do modelo cinemático foi feita uma simulação em matlab do quadrúpede e posterior concepção mecânica da plataforma do robô.

Figura 1.4: Plustech

1.4 APRESENTAÇÃO DO MANUSCRITO

No capítulo 2 é feita uma revisão bibliográfica sobre o tema de estudo, abrangendo a obtenção da cinemática direta e inversa de um robô quadrúpede, uma caracterização dos servos-motores RC, o tipo de sinal utilizado para o acionamento dos servos e uma seção sobre a comunicação utilizada. Em seguida, o capítulo 3 descreve a metodologia empregada para o desenvolvimento do projeto, incluindo as etapas da concepção mecânica e todo o desenvolvimento da eletrônica do robô. Resultados experimentais, simulação e práticos utilizando a plaforma e os motores, com o controle do posicionamento dos mesmos são discutidos no capítulo 4, seguido das conclusões sobre o resultado alcançado e propostas para continuidade do projeto no capítulo 5.

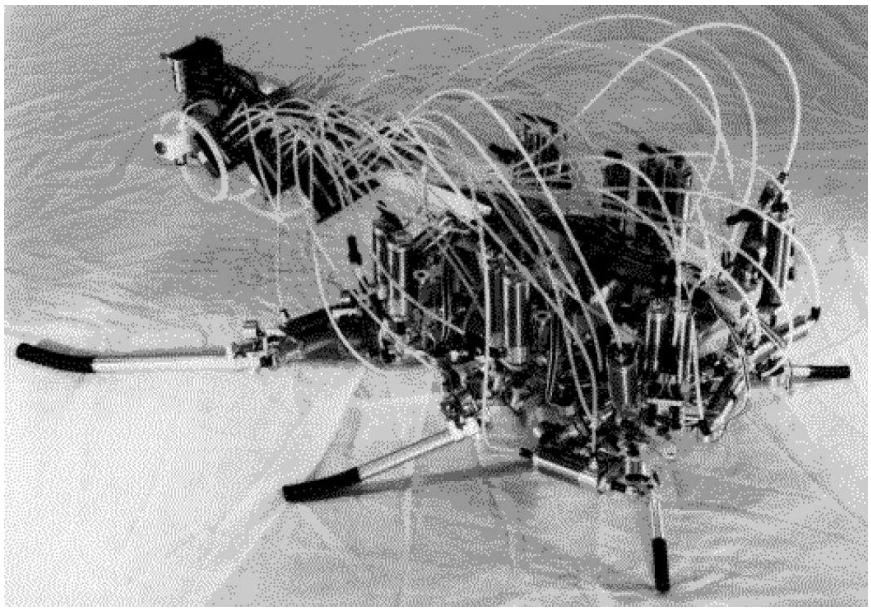


Figura 1.5: ROBOT III

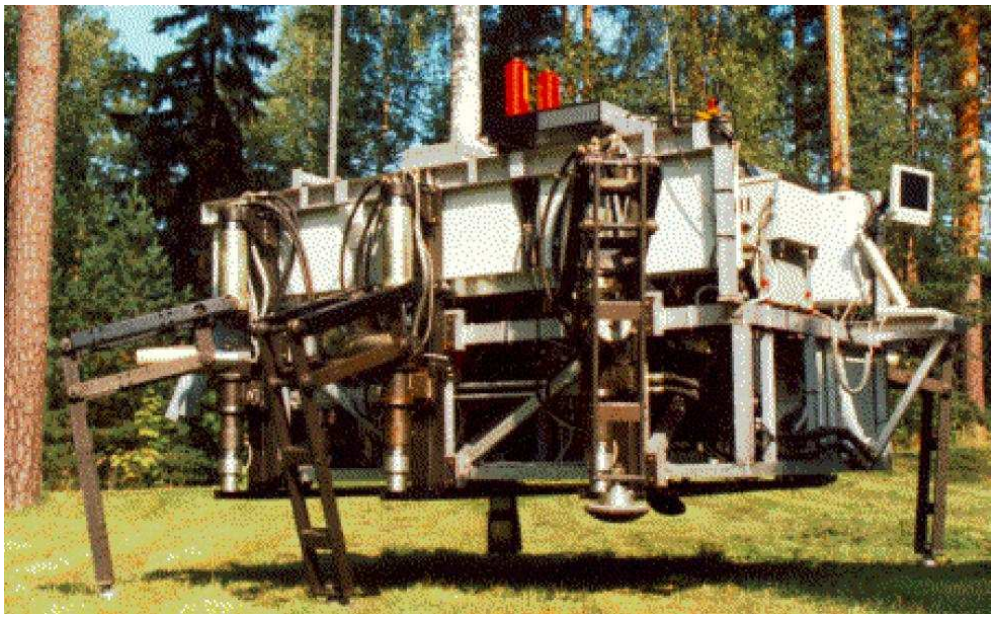


Figura 1.6: Mechanial Ant - Mechant

Figura 1.7: JoinMax da MCII Robot

Figura 1.8: Aibo da Sony

2 FUNDAMENTOS TEÓRICOS

2.1 INTRODUÇÃO

Nesta seção são apresentados os principais conceitos, em sua grande maioria, extra-curriculares para o desenvolvimento deste projeto.

Aqui apresenta-se a metodologia seguida para se obter as cinemáticas, direta e inversa, adotadas para a concepção da geometria do robô, seguida de uma introdução sobre as características e especificações para a utilização dos servos RC, também é feita uma breve introdução sobre os micro-controladores, em especial o ATmega8 da Atmel, que foi o micro-controlador utilizado para realizar tanto o controle do posicionamento dos servos quanto fornecer a possibilidade de comunicação com o PC, deixando uma seção apenas para se comentar sobre a saída PWM necessária para o controle do posicionamento dos servos e por fim uma revisão sobre a comunicação utilizada no projeto.

2.2 CINEMÁTICA DE ROBÔS MANIPULADORES

A cinemática é o ramo da mecânica que faz o estudo analítico da geometria do movimento em relação a um dado referencial, usualmente um sistema de eixos cartesianos.

No caso em questão pretende-se obter um modelo matemático com o qual seja possível descrever seu movimento, não só do centro do corpo, mas também de suas partes constituintes, os seus deslocamentos e rotações relativas.

Primeiramente será considerado o centro do robô e então analisado o movimento das pernas. Cada perna do quadrúpede possui 3 graus de liberdade. Naturalmente serão consideradas como variáveis independentes os ângulos associados a cada um desses graus de liberdade e os ângulos de rotação e a posição do centro do robô.

A cinemática pode ser dividida em duas questões, por um lado, dados os ângulos das juntas, qual a posição do ponto de apoio da perna com relação a um referencial base? Questão esta posta à cinemática direta. Por outro lado, dado um ponto onde queremos colocar o ponto de apoio, qual deve ser o conjunto de rotações de cada junta? Este é o problema da cinemática inversa.

Visto que grande parte da literatura descreve manipuladores, enquanto as referências sobre robôs com pernas é muito escassa. Considera-se uma perna como um manipulador simples com 3 graus de liberdade e com isso aproveita-se muitos dos conceitos dos manipuladores clássicos. Convenções tais como a de Denavit-Hartenberg (D-H) que descreve as relações espaciais dos manipuladores com o uso da álgebra de matrizes.

2.2.1 Cinemática direta

A teoria aqui apresentada baseia-se nas seguintes publicações: [3], [4] e [5]. Sendo que foram utilizadas complementarmente.

Para se resolver o problema da cinemática direta, partiu-se da semelhança da perna com um manipulador com 3 graus de liberdade. Para isso, será utilizado o método de D-H.

Após obter o modelo da perna se seguiu à integração dos quatro membros com o centro do robô de forma que possamos ter uma análise completa da cinemática direta do robô.

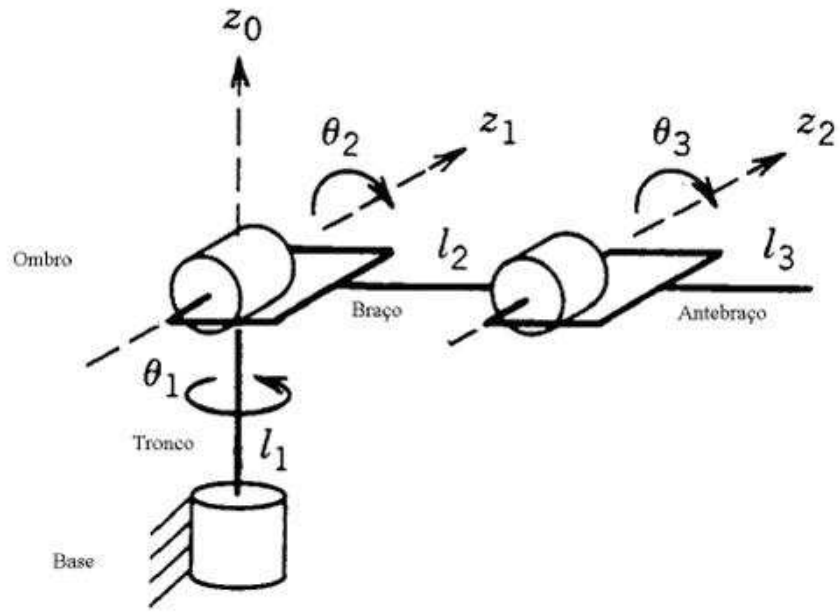


Figura 2.1: Robô articulado ou antropomórfico

2.2.1.1 Conceitos básicos

Manipuladores robóticos são constituídos por membros conectados por juntas em uma cadeia cinemática aberta. Tais juntas podem ser rotativas, que permitem apenas rotação relativa entre os membros, ou prismáticas, que permitem translação linear entre dois membros. Aqui será tratado apenas o caso de juntas do tipo rotativa.

Cada junta interconecta dois membros l_i e l_{i+1}. O eixo de rotação de uma junta é sempre denotado como eixo da junta z_i, se a junta interconectar os membros i e $i+1$. As variáveis das juntas são denotadas por q_i ou θ_i. O número das juntas determina os graus de liberdade do manipulador.

Dentre as configurações dos robôs industriais a perna do robô se encaixa na configuração de robô articulado ou antropomórfico, conforme mostra a figura 2.1.

2.2.1.2 Movimentos de Corpo Rígido

Para que se possa desenvolver as equações cinemáticas, há necessidade de se estabelecer vários sistemas de coordenadas para representar as posições e orientações de corpos rígidos. É necessário também conhecer as relações entre estes sistemas, de modo que vetores representativos de posições de um determinado sistema de coordenadas possam ser representados em outros sistemas de coordenadas.

2.2.1.3 Rotações

Dado um ponto genérico no espaço no sistema de coordenadas $Ox_1y_1z_1$ móvel, deseja-se representar este mesmo ponto no sistema $Ox_0y_0z_0$ fixo.

Sejam $\mathbf{i}_0$, $\mathbf{j}_0$ e $\mathbf{k}_0$ os vetores unitários do sistema $Ox_0y_0z_0$ e $\mathbf{i}_1$, $\mathbf{j}_1$ e $\mathbf{k}_1$ os vetores unitários do sistema $Ox_1y_1z_1$. Então é possível representar $\mathbf{P}$ nos dois sistemas:

$$\text{Sistema } Ox_0y_0z_0: \qquad \mathbf{p}_0 = p_{0x} \cdot \mathbf{i}_0 + p_{0y} \cdot \mathbf{j}_0 + p_{0z} \cdot \mathbf{k}_0 \tag{2.1}$$

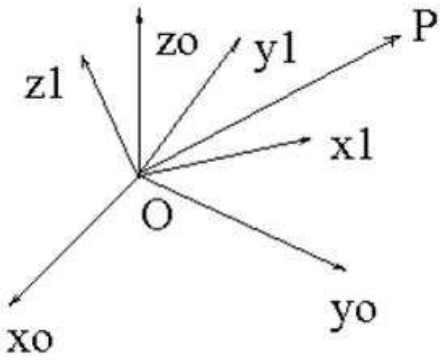


Figura 2.2: Relação entre sistemas de coordenadas

$$\text{Sistema } Ox_1y_1z_1: \qquad \mathbf{p}_1 = p_{1x} \cdot \mathbf{i}_1 + p_{1y} \cdot \mathbf{j}_1 + p_{1z} \cdot \mathbf{k}_1 \tag{2.2}$$

Como $\mathbf{p}_0$ e $\mathbf{p}_1$ representam o mesmo ponto pode-se escrever:

$$p_{0x} = \mathbf{p}_0 \times \mathbf{i}_0 = \mathbf{p}_1 \times \mathbf{i}_1 \tag{2.3}$$

$$p_{0y} = \mathbf{p}_0 \times \mathbf{j}_0 = \mathbf{p}_1 \times \mathbf{j}_1 \tag{2.4}$$

$$p_{0z} = \mathbf{p}_0 \times \mathbf{k}_0 = \mathbf{p}_1 \times \mathbf{k}_1 \tag{2.5}$$

Levando em conta as equações 2.2, 2.3, 2.4 e 2.5 te-se:

$$p_{0x} = p_{1x} \cdot \mathbf{i}_1 \times \mathbf{i}_0 + p_{1y} \cdot \mathbf{j}_1 \times \mathbf{i}_0 + p_{1z} \cdot \mathbf{k}_1 \times \mathbf{i}_0 \tag{2.6}$$

$$p_{0y} = p_{1x} \cdot \mathbf{i}_1 \times \mathbf{j}_0 + p_{1y} \cdot \mathbf{j}_1 \times \mathbf{j}_0 + p_{1z} \cdot \mathbf{k}_1 \times \mathbf{j}_0 \tag{2.7}$$

$$p_{0z} = p_{1x} \cdot \mathbf{i}_1 \times \mathbf{k}_0 + p_{1y} \cdot \mathbf{j}_1 \times \mathbf{k}_0 + p_{1z} \cdot \mathbf{k}_1 \times \mathbf{k}_0 \tag{2.8}$$

Que de forma compacta pode ser escrita como:

$$\mathbf{p}_0 = \mathbf{R}_0^1 \cdot \mathbf{p}_1 \tag{2.9}$$

Onde a matriz 3x3,

$$\mathbf{R}_0^1 = \begin{pmatrix} \mathbf{i}_1 \times \mathbf{i}_0 & \mathbf{j}_1 \times \mathbf{i}_0 & \mathbf{k}_1 \times \mathbf{i}_0 \\ \mathbf{i}_1 \times \mathbf{j}_0 & \mathbf{j}_1 \times \mathbf{j}_0 & \mathbf{k}_1 \times \mathbf{j}_0 \\ \mathbf{i}_1 \times \mathbf{k}_0 & \mathbf{j}_1 \times \mathbf{k}_0 & \mathbf{k}_1 \times \mathbf{k}_0 \end{pmatrix} \tag{2.10}$$

transforma o vetor $\mathbf{p}_1$ do sistema $Ox_1y_1z_1$ no vetor $\mathbf{p}_0$ do sistema $Ox_0y_0z_0$.

Pré-multiplicando a equação 2.1 por $(R_0^1)^{-1}$, tem-se:

$$\mathbf{p}_1 = (R_0^1)^{-1} \cdot \mathbf{p}_0 \tag{2.11}$$

Seguindo o mesmo raciocínio anterior pode-se mostrar que:

$$\mathbf{p}_1 = \mathbf{R}_1^0 \cdot \mathbf{p}_0 \tag{2.12}$$

Onde

$$\mathbf{R}_1^0 = \begin{pmatrix} \mathbf{i}_0 \times \mathbf{i}_1 & \mathbf{j}_0 \times \mathbf{i}_1 & \mathbf{k}_0 \times \mathbf{i}_1 \\ \mathbf{i}_0 \times \mathbf{j}_1 & \mathbf{j}_0 \times \mathbf{j}_1 & \mathbf{k}_0 \times \mathbf{j}_1 \\ \mathbf{i}_0 \times \mathbf{k}_1 & \mathbf{j}_0 \times \mathbf{k}_1 & \mathbf{k}_0 \times \mathbf{k}_1 \end{pmatrix} \tag{2.13}$$

que transforma o vetor $\mathbf{p}_0$ do sistema fixo $Ox_0y_0z_0$ no vetor $\mathbf{p}_1$ do sistema móvel $Ox_1y_1z_1$.

Comparando as equações de 2.12 a 2.13 verifica-se facilmente que:

$$\mathbf{R}_1^0 = (\mathbf{R}_1^0)^{-1} = (\mathbf{R}_1^0)^T \tag{2.14}$$

Uma vez que a inversa é igual a transposta, implica que a matriz de rotação é ortogonal.

Supondo um outro sistema móvel $Ox_2y_2z_2$ pode-se fazer composição de rotações da seguinte maneira:

$$\mathbf{p}_0 = \mathbf{R}_0^1 \cdot \mathbf{p}_1 \tag{2.15}$$

$$\mathbf{p}_0 = \mathbf{R}_0^2 \cdot \mathbf{p}_2 \tag{2.16}$$

$$\mathbf{p}_1 = \mathbf{R}_1^2 \cdot \mathbf{p}_2 \tag{2.17}$$

Substituindo a equação 2.17 na equação 2.15 tem-se:

$$\mathbf{p}_0 = \mathbf{R}_0^1 \cdot \mathbf{R}_1^2 \cdot \mathbf{p}_2 \tag{2.18}$$

Que pode-se generalizar:

$$\mathbf{R}_0^n = \mathbf{R}_0^1 \cdot \mathbf{R}_1^2 \cdots \mathbf{R}_{n-1}^n \tag{2.19}$$

Uma representação que é utilizada aqui para orientação do centro do robô é por ângulos de navegação *Roll-Pitch-Yaw*, que é uma maneira de se representar rotações através de três rotações sucessivas em torno dos eixos de um mesmo sistema fixo na seguinte seqüência:

1. Rotação ψ (Yaw = Guinagem): em torno de x.
2. Rotação θ (Pitch = Arfagem): em torno de y.
3. Rotação ϕ (Roll = Rolagem): em torno de z.

Assim tem-se:

$$\mathbf{R}_0^1 = \mathbf{R}_{z,\phi} \cdot \mathbf{R}_{y,\theta} \cdot \mathbf{R}_{x,\psi} \tag{2.20}$$

2.2.1.4 Transformações Homogêneas

Tem-se um sistema móvel $Ox_1y_1z_1$ obtido por translação pura a partir de $Ox_0y_0z_0$, como mostra a figura 2.3.

A origem foi deslocada de um vetor $\mathbf{d}_0^1$ que fornece a posição do novo sistema e nada diz a respeito de sua rotação que é dado por $\mathbf{R}_0^1$. Agora, tomando uma translação com rotação e querendo representar o vetor $\mathbf{p}_1$ de $Ox_1y_1z_1$ no sistema $Ox_0y_0z_0$, tem-se:

$$\mathbf{p}_0 = \mathbf{R}_0^1 \cdot \mathbf{p}_1 + \mathbf{d}_0^1 \tag{2.21}$$

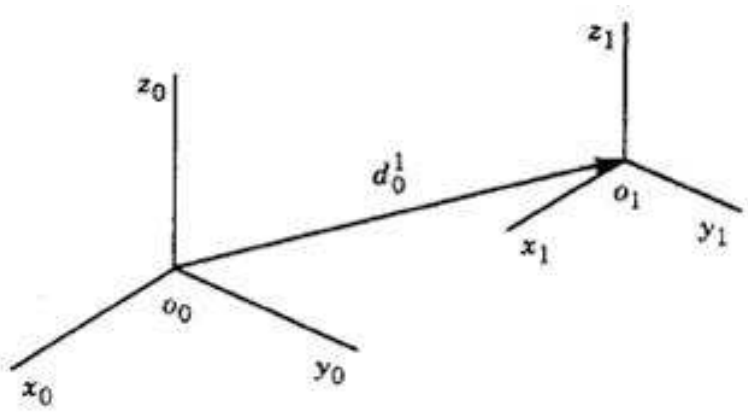


Figura 2.3: Translação pura

Que pode ser representado de forma conveniente como uma matriz quadrada 4x4:

$$\begin{bmatrix} \mathbf{p}_{0x} \\ \mathbf{p}_{0y} \\ \mathbf{p}_{0z} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{11} & \mathbf{r}_{12} & \mathbf{r}_{13} & \mathbf{d}_{0x}^1 \\ \mathbf{r}_{21} & \mathbf{r}_{22} & \mathbf{r}_{23} & \mathbf{d}_{0y}^1 \\ \mathbf{r}_{31} & \mathbf{r}_{32} & \mathbf{r}_{33} & \mathbf{d}_{0z}^1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_{1x} \\ \mathbf{p}_{1y} \\ \mathbf{p}_{1z} \\ 1 \end{bmatrix} \tag{2.22}$$

2.2.1.5 Notação de Denavit-Hartenberg

A obtenção das equações que resolvem a cinemática direta, com base em conhecimentos de geometria e trigonometria, é relativamente fácil caso se queira modelar robôs que atuam apenas em um plano, contudo para robôs espaciais essa formulação se tornam bastante complexas, assim optou-se pela representação de Denavit-Hartenberg, que é um notação consagrada em mecanismos e robótica.

Um robô de cadeia aberta com $n+1$ membros, incluída a base que é o membro 0, conectados por n juntas. Seus corpos são numerados de 0 a n a partir da base, as juntas de 1 a n, sendo que a i-ésima junta conecta o membro i ao membro $i-1$. A variável da i-ésima junta será denotada aqui por q_i. Um sistema de coordenadas será associado a cada membro, onde o subscrito desse sistema denota qual membro o mesmo representa.

Seja uma matriz $\mathbf{A}_{i-1}^{i}$ que representa a transformação do membro i para o sistema $i-1$ e como cada matriz tem apenas um grau de liberdade, essa matriz é função apenas da variável q_i.

Para se chegar à orientação e posição do órgão terminal deve-se percorrer toda a cadeia cinemática desde a base até o órgão terminal:

$$\mathbf{H}_0^n = \mathbf{A}_0^1 \cdot \mathbf{A}_1^2 \cdots \mathbf{A}_{n-1}^n \tag{2.23}$$

onde cada transformação homogênea é dada por:

$$\mathbf{A}_{i-1}^{i} = \begin{bmatrix} \mathbf{R}_{i-1}^{i} & \mathbf{d}_{i-1}^{i} \\ 0 & 1 \end{bmatrix} \tag{2.24}$$

Desse modo, a resolução da cinemática direta resume-se a encontrar a solução da equação 2.23, obtendo nove equações, onde 3 determinam a posição e 9 determinam a orientação do órgão terminal.

Utilizando DH consegue-se uma simplificação para chegarmos a esta solução. Nessa representação cada matriz $\mathbf{A}_{i-1}^{i}$ é representada pelo produto de quatro transformações básicas:

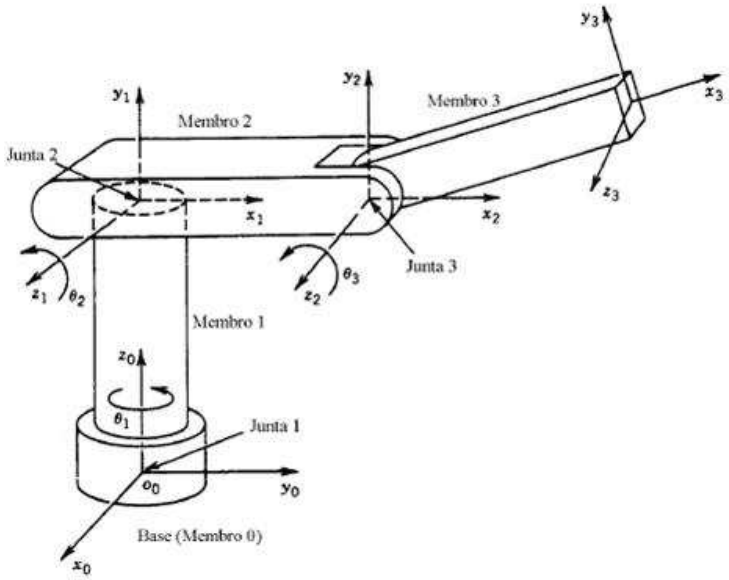


Figura 2.4: Convenções adotadas de membros, juntas e sistemas de coordenadas

$$\mathbf{A}_{i-1}^{i} = \mathbf{R}_{z,\theta} \cdot \mathbf{T}_{z,d} \cdot \mathbf{T}_{x,a} \cdot \mathbf{R}_{x,\alpha} \tag{2.25}$$

onde:

- $\mathbf{R}_{z,\theta}$ representa uma rotação de θ em torno do eixo z, com sentido dado pela regra da mão direita.
- $\mathbf{T}_{z,d}$ representa uma translação de d ao longo do eixo z, com o sinal dado pelo sentido do eixo.
- $\mathbf{T}_{x,a}$ representa translação de a ao longo do eixo x, com o sinal dado pelo sentido do eixo.
- $\mathbf{R}_{x,\alpha}$ representa rotação de α em torno do eixo x, com sentido dado pela regra da mão direita.

Os parâmetros θ, d, a e α são chamados de parâmetros do membro ou parâmetros DH. Suas denominações são as seguintes:

- θ = ângulo
- d = excentricidade
- a = comprimento
- α = torção

Desenvolvendo agora a equação 2.25 obtemos:

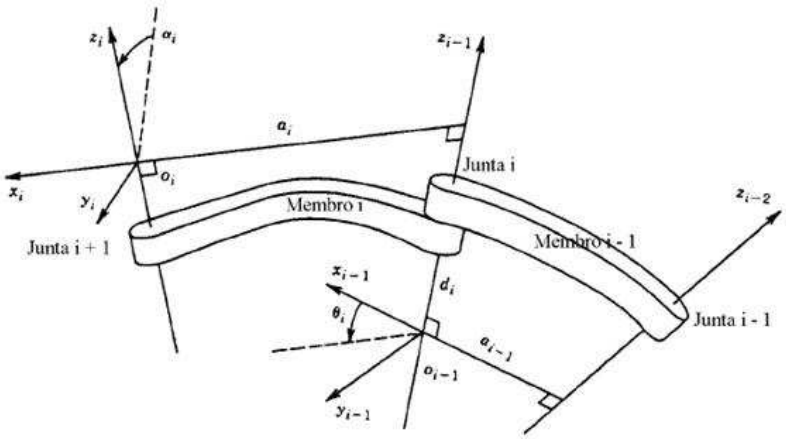


Figura 2.5: Representação de DH

$$\mathbf{A}_{i-1}^{i} = \begin{bmatrix} cos(\theta_i) & -sen(\theta_i) & 0 & 0 \\ sen(\theta_i) & cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha_i) & -sen(\alpha_i) & 0 \\ 0 & sen(\alpha_i) & cos(\alpha_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.26}$$

$$= \begin{bmatrix} cos(\theta_i) & -sen(\theta_i) \cdot cos(\alpha_i) & sen(\theta_i) \cdot sen(\alpha_i) & a_i \cdot cos(\theta_i) \\ sen(\theta_i) & cos(\theta_i) \cdot cos(\alpha_i) & -cos(\theta_i) \cdot sen(\alpha_i) & a_i \cdot sen(\theta_i) \\ 0 & sen(\alpha_i) & cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.27}$$

Os parâmetros d, a e α são constantes para um dado manipulador, eles dependem da geometria do manipulador, e o parâmetro θ é a única variável, que é a própria variável da junta, ou o q, como aqui utilizado.

Para poder se utilizar D-H deve-se seguir as seguintes condições:

- O eixo z_{i-1} é o eixo da junta i.

- O eixo x_i é perpendicular ao eixo z_{i-1}, apontando para fora deste.

Com a figura 2.5 pode-se extrair os parâmetros de DH para os manipuladores e concluíndo-se a cinemática direta. Para o projeto aqui apresentado é importante apenas a posição do órgão terminal, então será tratada apenas a quarta coluna da matriz.

2.2.2 Cinemática inversa

A teoria aqui apresentada foi baseada nas seguintes publicações: [6] e [7]. Sendo que foram utilizadas complementarmente.

O problema da cinemática inversa tem sido muito estudado na robótica móvel, diversos métodos têm sido descritos na literatura para resolver esse problema. Aqui serão citados os três principais métodos que vêm sendo utilizados: Transposta do Jacobiano, Pseudoinversa e *Damped Least Squares*.

2.2.2.1 Jacobiano

O problema da cinemática inversa é encontrar um conjunto de ângulos das juntas que satisfaça a posição final desejada do ponto de apoio. Infelizmente podem haver mais do que uma solução.

$$\mathbf{t}_i = \mathbf{s}_i(\theta) \tag{2.28}$$

onde $\mathbf{t}$ é o vetor alvo, onde $\mathbf{t}_i$ é a posição alvo do i-ésimo ponto de apoio e $\mathbf{s}$ é a função que relaciona o vetor alvo com os ângulos das juntas.

Pode-se usar métodos iterativos para aproximar uma boa solução. Para isso as funções de posição encontradas pela cinemática direta (quarta coluna da matriz de DH) são linearmente aproximadas pela Matriz Jacobiano, que é definida pela equação 2.29.

$$\mathbf{J}(\theta) = \left(\frac{\partial \mathbf{s}_i}{\partial \theta_i}\right)_{i,j} \tag{2.29}$$

A equação básica que descreve as velocidades dos pontos de apoio pode ser escrita como:

$$\dot{\mathbf{s}} = \mathbf{J}(\theta) \cdot \dot{\theta} \tag{2.30}$$

O uso do jacobiano leva a um método iterativo para se resolver o problema da cinemática inversa. Supondo que se possui os valores atuais para θ, $\mathbf{s}$ e $\mathbf{t}$. Com estes valores pode-se encontrar o Jacobiano. Então procura-se por um valor de atualização $\Delta\theta$ com o propósito de se-atualizar as juntas por esse incremento:

$$\theta = \theta + \Delta\theta \tag{2.31}$$

Pela equação 2.30 pode-se estimar as mudanças nos pontos de apoio causadas por esse incremento:

$$\Delta\mathbf{s} \approx \mathbf{J}\Delta\theta \tag{2.32}$$

Desse modo os valores de θ são alterados iterativamente até que um valor de s seja suficientemente próximo da solução. Será tratado agora como escolher $\Delta\theta$ de modo a atualizar os ângulos da junta. À luz da equação 2.32, uma solução é resolver a equação 2.33.

$$\mathbf{e} = \mathbf{J}\Delta\theta \tag{2.33}$$

onde $\mathbf{e}$ é a mudança desejada na posição do ponto de apoio:

$$\mathbf{e} = \mathbf{t} - \mathbf{s} \tag{2.34}$$

Na maioria das vezes a equação 2.33 não pode ser resolvida de maneira única, o Jacobiano pode não ser quadrado ou não inversível. Mesmo que seja, o usado de sua inversa ($\Delta\theta = \mathbf{J}^{-1}\mathbf{e}$) não funcionará de maneira adequada próximo a uma singularidade.

2.2.2.2 *Singular Value Decomposition*

Aqui provém-se uma maneira adequada de análise dos métodos da pseudoinversa e do *damped least squares*.

A decomposição de $\mathbf{J}$ consiste em:

$$\mathbf{J} = \mathbf{U}\mathbf{D}\mathbf{V}^T = \sum_{i}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \tag{2.35}$$

onde $\mathbf{U}$ e $\mathbf{V}$ são matrizes ortogonais, $\mathbf{D}$ é uma matriz diagonal e r é o posto (*rank*) de $\mathbf{J}$. Os únicos valores não nulos da matriz $\mathbf{D}$ são $\sigma_i = d_{i,i}$ ao longo da diagonal. Na equação 2.35, $\mathbf{v_i}$ é a i-ésima coluna da matriz $\mathbf{V}$. O mesmo vale para $\mathbf{u_i}$ com relação a $\mathbf{U}$.

2.2.2.3 A transposta do jacobiano

A idéia do método da Transposta do Jacobiano é muito simples. Será utilizada apenas a transposta do Jacobiano ao invés de sua inversa[6]. Assim:

$$\Delta\theta = \alpha \mathbf{J}^T \mathbf{e} \tag{2.36}$$

Utilizando 2.32 tem-se que a mudança no ponto de apoio é dada aproximadamente por:

$$\Delta\mathbf{s} \approx \alpha \mathbf{J}\mathbf{J}^T \mathbf{e} \tag{2.37}$$

Cabe agora decidir como escolher o escalar α. Uma maneira razoável de se fazer isso é assumindo que a mudança na posição do ponto de apoio será exatamente a $\alpha\mathbf{J}\mathbf{J}^T\mathbf{e}$, e escolher α de forma a fazer esse valor ir o mais próximo possível de $\mathbf{e}$. Levando a:

$$\alpha = \frac{\langle \mathbf{e}, \mathbf{J}\mathbf{J}^T\mathbf{e} \rangle}{\langle \mathbf{J}\mathbf{J}^T\mathbf{e}, \mathbf{J}\mathbf{J}^T\mathbf{e} \rangle} \tag{2.38}$$

2.2.2.4 A pseudoinversa

O método da pseudoinversa escolhe os valores de atualização das juntas como:

$$\Delta\theta = \mathbf{J}^{\dagger}\mathbf{e} \tag{2.39}$$

onde a matriz $\mathbf{J}^{\dagger}$ é a pseudoinversa (solução de mínima energia) do Jacobiano, também chamada de inversa de *Moore-Penrose*. Essa inversa é definida para todas as matrizes, inclusive as não quadradas.

A pseudoinversa tem problemas de estabilidade próximo a singularidades. Em uma singularidade a matriz $\mathbf{J}$ perde seu posto, correspondendo ao fato de que há uma direção de movimento do ponto de apoio que não é alcançável. Se estiver exatamente em uma singularidade, a pseudoinversa não tentará mover em uma direção impossível. Esse método tem sido discutido muito na literatura, mas freqüentemente tem uma performance ruim por causa da instabilidade próximo às singularidades. O método a seguir tem performance muito superior.

Utilizando SVD pode-se reescrever a pseudoinversa como:

$$\mathbf{J}^{\dagger} = \mathbf{V}\mathbf{D}^{\dagger}\mathbf{U}^{T} = \sum_{i}^{r} \sigma_i^{-1}\mathbf{v}_i\mathbf{u}_i^{T} \tag{2.40}$$

A pseudoinversa diverge quando σ_i se aproxima de zero. Na realidade, quando se aproxima de uma singularidade σ_i se aproxima de zero.

2.2.2.5 *Damped Least Squares*

Esse método pode evitar muitos dos problemas da pseudoinversa com singularidades e leva a um método numérico estável ao selecionar $\Delta\theta$. Aqui ao invés de encontrar um vetor mínimo de $\Delta\theta$ que gera a melhor solução da equação 2.33, procuramos o valor de $\Delta\theta$ que minimiza a equação:

$$\|\mathbf{J}\Delta\theta - \mathbf{e}\|^2 + \lambda^2\|\Delta\theta\|^2 \tag{2.41}$$

onde λ é uma constante de *damping* não-nula $\in \Re$.

Desse modo, a solução para esse método é:

$$\Delta\theta = \mathbf{J}^T\left(\mathbf{J}\mathbf{J}^T + \lambda^2\mathbf{I}\right)^{-1}\mathbf{e} \tag{2.42}$$

A constante de textitdamping depende de detalhe do corpo e das posições do alvo e deve ser escolhido de maneira cuidadosa para que a equação 2.42 se torne numericamente estável. A constante deve grande suficiente de modo que as soluções para $\Delta\theta$ sejam estáveis próximos às singularidades, mas se for escolhido muito grande a taxa de convergência é muito lenta.

Utilizando SVD pode-se reescrever a pseudoinversa como:

$$\mathbf{J}^T\left(\mathbf{J}\mathbf{J}^T + \lambda^2\mathbf{I}\right)^{-1} = \sum_{i}^{r} \frac{\sigma_i}{\sigma_i^2 + \lambda^2}\mathbf{v}_i\mathbf{u}_i^T \tag{2.43}$$

Para valores em que σ_i é grande se comparado a λ , esse método não é muito diferente da pseudoinversa, mas quando σ_i é da mesma magnitude de λ, os valores σ_i^{-1} e $\frac{\sigma_i}{\sigma_i^2+\lambda^2}$ divergem. Desse modo, esse método é muito semelhante ao da pseudoinversa fora das singularidades, mas suaviza a performance da pseudoinversa na vizinhança das singularidades.

2.3 SERVOS R/C

Estão sendo utilizados ao todo 15 servos, 3 servos por perna, totalizando 12 servos para as pernas, 2 servos para a cabeça e apenas um servo para o rabo.

Para a cabeça e o rabo são utilizados servos da futaba, modelo s3003, figura 2.6, com as seguintes especificações técnicas mostradas nas tabelas 2.1 e 2.2 (retiradas do site[1] do próprio fabricante).

Para o posicionamento das pernas são utilizados servos da HITEC, modelo HS-755HB, figura 2.7. Tais servos possuem uma resistência em suas engrenagens muito superior aos servos da futaba utilizados além de um torque muito mais alto, como mostram suas especificações nas tabelas 2.3 e 2.4 (dados retirados do site[2] do fornecedor).

[1] http://www.futaba-rc.com/

[2] http://www.imagesco.com/catalog/motors/HS-755HB.html

Tabela 2.1: Especificações Servo Futaba S3003

Tensao DC (V)	**Torque (N.m)**	**Tempo (em segundos) para percorrer** $60\,^{\circ}$
4,8	0,31	0,23
6,0	0,40	0,16

Tabela 2.2: Dimensões Servo Futaba S3003

Massa (g)	**Dimensoes(mmxmmxmm)**
40	40x20x38

Os servo-motores utilizados são os mesmos amplamente utilizados em carros, barcos, aviões e helicópteros RC (Controlados remotamente). Os servo motores utilizados possuem 3 fios, vermelho e preto para alimentação (+VDC e terra) e um fio branco para o controle de posição do servo. O sinal de controle é um PWM de período entre 17 e 21 ms, e ciclo de trabalho variando entre 1 e 2 ms que controla a posição do rotor.

Um pulso de 1ms provoca uma rotação no sentido anti-horário e posiciona o servo em um limite e um pulso de 2ms provoca uma rotação do rotor no sentido horário, posicionando o servo no outro extremo. A posição central do servo é obtida com um pulso de 1,5ms. Tais pulsos são mantidos a uma freqüência de aproximadamente 50Hz. A figura 2.8 mostra o posicionamento do rotor do motor relacionando-o com a largura do pulso emitida.

Deve-se tomar um cuidado especial ao se tratar com servos-motores utilizando os valores especificados acima, uma vez que diferentes servos operam a diferentes faixas de operação e valores muito acima, ou muito abaixo dessa faixa de operação podem vir a danificar os servos. Por isso foram realizados diversos testes para identificarmos as posições mínimas e máximas que cada servo em particular iria operar.

2.4 MICRO-CONTROLADOR AVR ATMEGA8 DA ATMEL

Os micro-controladores permitem a execução de diversas tarefas com lógica digital, uma vez que sua estrutura funcional permite tais realizações.

Características como conversão A/D, comunicação síncrona e assíncrona, *watchdog* programável, comparador analógico, oscilador interno calibrável e capacidade de re-programação são algumas das muitas funcionalidades que permitem integração, confiabilidade e funcionalidade dos micro-controladores com outros sistemas digitais ou analógicos.

Além de que, em geral, possuem um atrativo especial o seu baixo custo e em alguns casos uma grande facilidade de operação.

Tabela 2.3: Especificações Servo HITEC HS755HB

Tensao DC (V)	**Torque (N.m)**	**Tempo (em segundos) para percorrer** $60\,^{\circ}$
4,8	1,08	0,28
6,0	1,29	0,23

Figura 2.6: Servo motor Futaba modelo S3003

Tabela 2.4: Dimensões Servo HITEC HS755HB

Massa (g)	Dimensoes(mmxmmxmm)
110	59x29x50

Todas as informações nesta seção foram retiradas do datasheet do próprio componente [8], porém aqui são apresentadas com uma maior clareza e altamente direcionada aos nossos propósitos de utilização do micro-controlador.

O ATmega8 é um micro-controlador de 8-bits CMOS cujo conjunto de instruções é baseado na arquitetura AVR RISC. Uma característica interessante deste dispositivo é o poder computacional que se aproxima de uma instrução por ciclo. Esta vazão (*throughput*) é possível em virtude da conexão direta de seus 32 registradores, de propósito geral, com a ULA (Unidade Lógica Aritmética). Isso permite que dois registros independentes sejam alcançados em uma única instrução, que no caso do AVR é executada em um único ciclo de clock.

Dentre várias funcionalidades do AVR ATmega8 as abaixo relacionadas são utilizadas neste projeto:

1. Memória programável de 8 Kbytes
2. EEprom de 512 Bytes;
3. 1 Kbyte de SRam;
4. 23 pinos de E/S (Entrada e saída);
5. 32 registradores de uso geral;
6. 3 Timer/counters flexíveis com módulo de comparação e saída PWM;
7. Interrupção interna e externa;
8. USART serial programável.

A arquitetura do AVR pode ser vista na 2.9.

Com o objetivo de maximizar a performance e o paralelismo, a filosofia adotada na implementação do AVR é a arquitetura Harvard em que os barramentos associados às memórias de dados e do programa são distintos. Diferentemente da forma clássica proposta por Von Neumann, em que os programas e os dados são alocados na mesma memória e o barramento a ser utilizado conseqüentemente acaba sendo o mesmo.

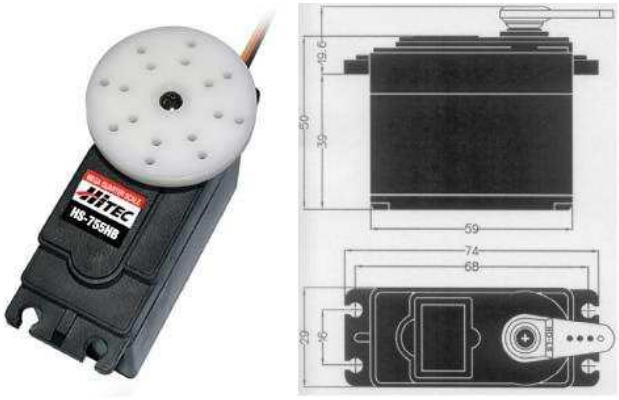


Figura 2.7: Servo motor HITEC modelo HS755HB

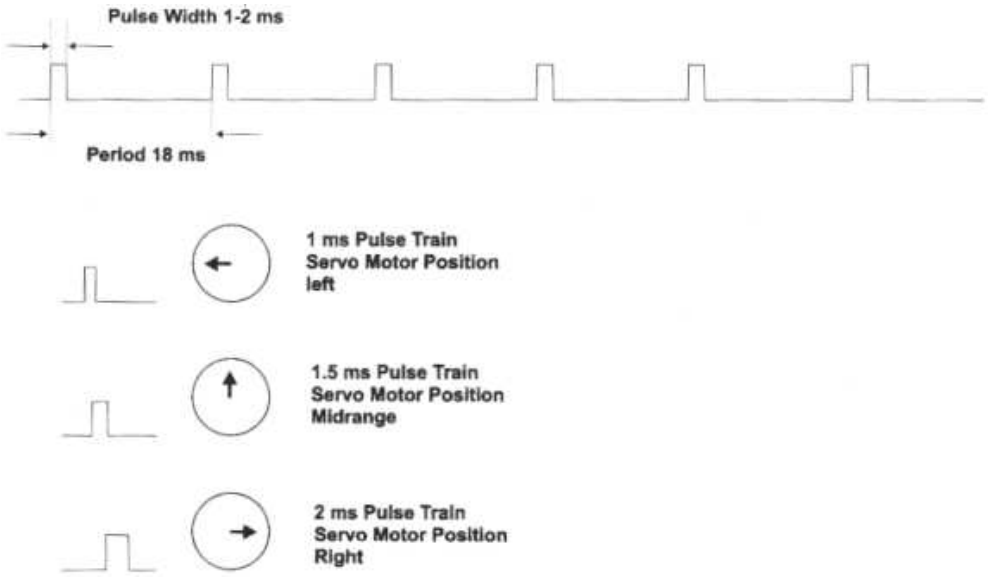


Figura 2.8: Sinal de posicionamento dos servos

O método pipeline também é utilizado na arquitetura do AVR. Neste método, enquanto uma instrução começa a ser executada, uma outra já é buscada da memória de programa para que a mesma possa ser iniciada no próximo ciclo de clock. Um dispositivo de acesso rápido é constituído de 32 registradores de 8 bits de propósito geral, conectados diretamente a ULA (Unidade Lógica Aritmética). Esta configuração permite a execução de 1 operação típica da ULA em um único ciclo de *clock*.

Seis dos 32 registradores podem ser utilizados como 3 ponteiros indiretos de 16 bits, essa característica permite cálculos de endereços mais eficientes.

Um elemento muito importante do diagrama, que também é interessante comentar, é a ULA (Unidade Lógica Aritmética). A ULA do AVR possui uma conexão direta com todos os 32 registradores. As operações são divididas em três categorias principais; lógica, aritmética e funções bit a bit. A arquitetura também permite algumas multiplicações com ou sem sinal e em formato fracionário.

2.4.1 *"Clock"*

A Figura 2.10 mostra o diagrama de geração *clock* do AVR, bem como sua distribuição.

Utilizando-se de um multiplexador, a fonte dos pulsos de *clock* é selecionada e distribuída aos sistemas

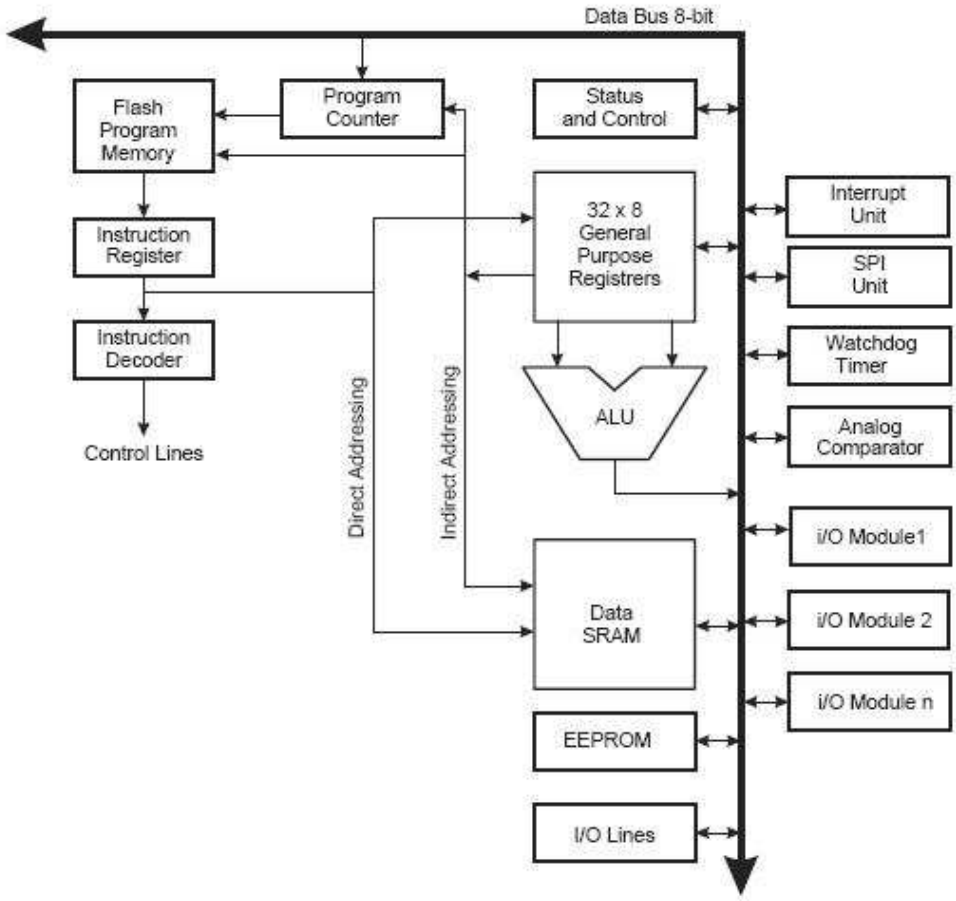


Figura 2.9: Diagrama de Blocos da Arquitetura do ATmega8

secundários por meio do CCU (*Clock Control Unit*), possibilitando assim a execução de praticamente todas as operações do AVR. Os módulos de segurança, como por exemplo o *Watchdog Oscillator* é conectado diretamente ao multiplexador para que não haja interferência do CCU no sistema de proteção, o que aumenta a confiabilidade das aplicações.

Diversas configurações e tipos de geradores de pulsos são configuráveis para o AVR. Optou-se discutir e mostrar apenas o *Calibrated Internal RC Oscillator*, pois as outras configurações não serão utilizadas neste projeto.

O Oscilador RC interno calibrável pode ser configurado para trabalhar com freqüências fixas de 1, 2, 4 ou 8 megahertz. Os valores nominais destas freqüências são obtidos quando as condições de trabalho são: alimentação de 5V e temperatura $25\,^{\circ}\mathrm{C}$. Neste projeto optou-se por utilizar, em um primeiro momento, a utilização do oscilador RC interno operando na freqüência de 1 megahertz.

Caso se deseje uma configuração de *clock* mais precisa, existe a possibilidade da utilização de um cristal externo, esta configuração pode ser vista na figura 2.11. A utilização deste recurso requer uma pequena alteração nas configurações de gravação do arquivo *makefile* (diretivas de gravação do AVR GCC).

2.4.2 *"Timers"'*

O micro-controlador possui 3 *Timers*, sendo 2 de 8 bits e 1 de 16 bits. Entre os *Timers*, 2 possuem um módulo comparador, ou seja, é possível determinar a mudança de estado dos contadores pela ocorrência de um evento externo, não utilizando, conseqüentemente, recurso computacional devido a ocorrência do

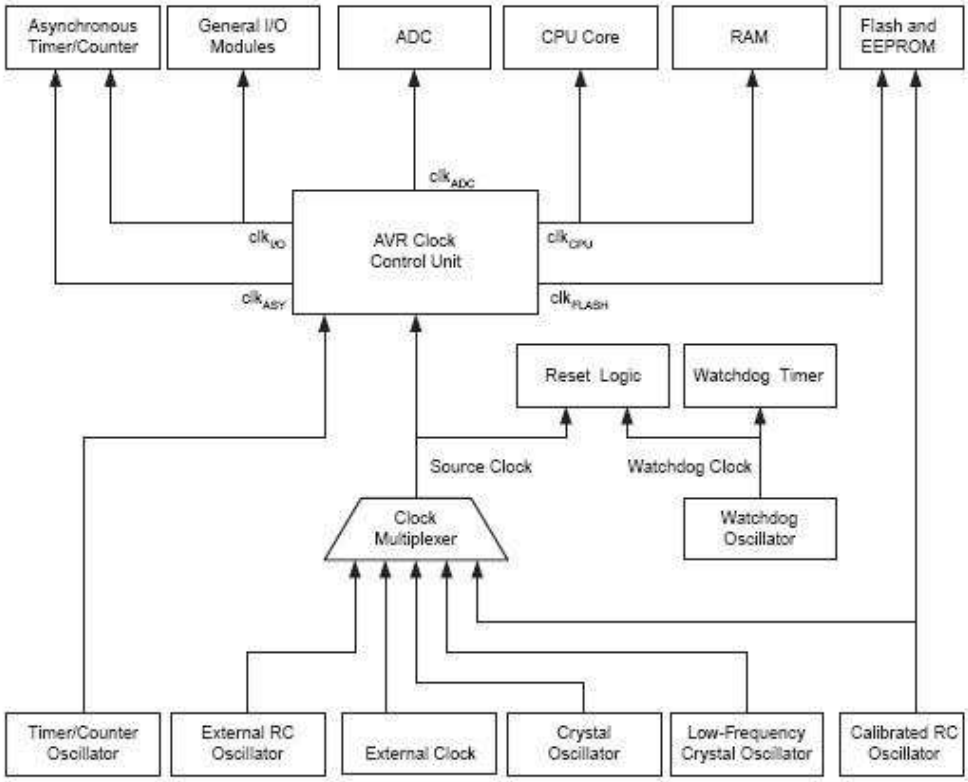


Figura 2.10: Diagrama de geração do *Clock*

evento.

É possível configurar a referencia dos *Timers* tanto para o oscilador interno, quanto para o externo. Cada um dos *Timers* possuem uma possível configuração de *prescalers* independente, sendo que a configuração padrão é sempre desligado.

Para configuração de cada um dos *prescalers* é necessário setar alguns bits dos registradores específicos de cada *Timer*.

O registrador TCNT[0..2] são os contadores de interrupções, estes são registradores tanto de leitura quanto de escrita.

Um ferramental extremamente útil no projeto é a possibilidade de gerar interrupções por meio do clock, as interrupções geradas podem chamar rotinas programadas para executar algum conjunto de código. Para gerar as interrupções existem alguns registradores que necessitam ser configurados.

Com exceção do *Timer0* os outros *Timers* trazem consigo a funcionalidade da configuração do PWM (*Pulse Width Modulator*).

2.4.3 Interrupções

Um aspecto muito importante na utilização de um micro-controlador é o tratamento das interrupções, que podem ser tanto internas, quanto externas. Pode-se descrever interrupções como sendo desvios condicionais efetuados pelo programa em função da ocorrência de um fenômeno prioritário em um determinado instante. As interrupções são muito importantes, pois elas executam as respostas programadas para cada interação com o meio.

De acordo com [8], como medida de segurança, existe para cada interrupção um vetor de registradores único. Essa configuração impede que na execução de uma interrupção outra interrupção seja executada,

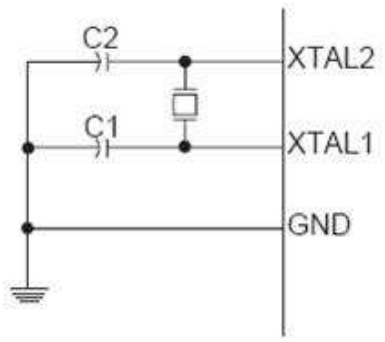
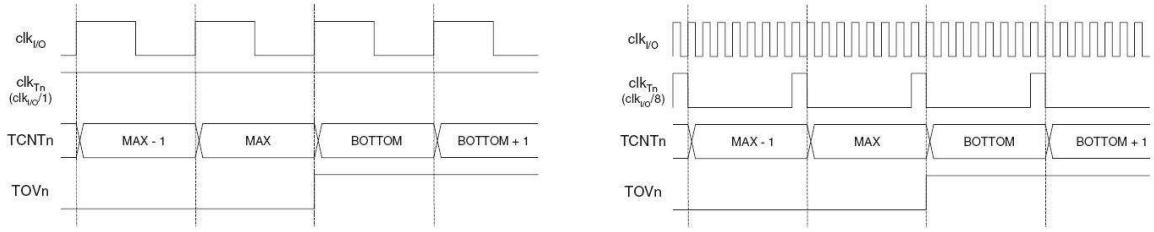


Figura 2.11: Esquemático para utilização do Cristal externo



(a) Sem *prescaler* (b) Com *prescaler* ($f_{clk\_I/O}$/8)

Figura 2.12: *Timer/Counter*, diagramas de tempo

anulando a anterior ou que o programa perca o ponteiro do PC (*Program Couter*) e com isso não retorne a execução normal do programa. Um outro fato é a existência de prioridades na execução das interrupções, ou seja, caso a prioridade de uma interrupção seja maior que a da outra, o seu endereço é colocada no vetor geral de interrupção como prioritária e conseqüentemente ocorrerá primeiro.

2.4.4 Saída PWM

Os micro-controladores possuem *timers* que funcionam como contadores programados para incrementar um determinado registrador em intervalos de tempo constantes definidos pelo *clock* de operação e por um divisor da escala de tempo, os *prescalers*. O *clock* do microcontrolador gera pulsos em intervalos constantes, a cada pulso o registrador incrementa seu valor, caso o *timer* esteja configurado com algum *prescaler* este incremento só é realizado caso o número de pulsos seja igual ao valor definido pelo *prescaler*, como mostra a figura2.12.

O registrador dos contadores de pulsos dos *timers* são de 16 e 8 bits, no caso do atmega8. Sendo assim o valor máximo que os contadores dos *timers* de 16 bits podem chegar é de 0xFFFF (65535) enquanto os de 8 bits chegam até 0xFF (255). Ao chegar ao fim da contagem o contador inicia do zero novamente ou começa a decrementar o registrador do contador. Para ambos os *timers*, foi configurado que os contadores iniciariam em 0 e contariam até um valor máximo que determinaria a freqüência do sinal do PWM.

2.5 COMUNICAÇÃO SERIAL

Devido à necessidade de utilizarmos diversos micro-controladores é necessário estabelecermos comunicação entre tais dispositivos e com o PC. Tal comunicação poderia ser paralela ou serial, como os micro-controladores utilizados são dotados internamente de uma USART (*Universal Synchronous-Asynchronous Receiver Transmiter*), que é um controlador de comunicação serial, este tipo de comunicação foi a escol-

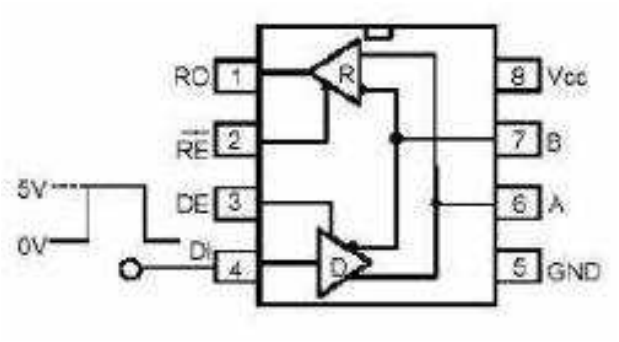


Figura 2.13: Diagrama dos transceptores DS485 e ST485

hida para ser implementada no projeto. A nota técnica [1] foi a base da elaboração desta seção.

Dos padrões mais conhecidos para o estabelecimento da comunicação serial estão o RS-232 e o RS-485. O padrão RS-232 é comumente utilizado para realizar a comunicação entre apenas dois dispositivos, a uma distância relativamente curta e a baixas velocidades de transmissão, devido sua baixa imunidade à ruídos.

Já o padrão RS-485, por ser diferencial, apresenta grande imunidade a ruídos e permite a transmissão a maiores distâncias. Uma desvantagem deste padrão em relação ao RS-232 é o fato de que o padrão RS-485 é *half-duplex* enquanto o RS-232 é *full-duplex,* ou seja, no padrão RS-232 os dispositivos são capazes de receber e enviar dados simultâneamente, já no padrão RS-485 ou o dispositivo está no modo de recepção ou está no modo de transmissão. Outra peculiaridade a ser considerada, em relação ao padrão RS-485, é que caso 2 dispositivos diferentes tentem acessar o barramento no mesmo instante a mensagem provavelmente será comprometida, desta forma faz-se necessário a concepção de um controle de acesso ao barramento.

A Figura 2.13 mostra o diagrama de dois tipos de transceptores do padrão RS-485, o DS485 e o ST485.

As letras D e R, indicam os drivers de transmissão e recepção, respectivamente, de cada dispositivo. O pino $\overline{RE}$ habilita o driver de recepção R, sendo ativo com $\overline{RE}$ no nível 0. O pino DE habilita o driver de transmissão D (ativo em 1), a fim de garantir que o dispositivo irá se encontrar apenas em um dos dois modos, transmissão ou recepção, os pinos de habilitação dos drivers são, geralmente, curto-circuitados, desta forma, garantimos que o transceptor esteja apenas recebendo ou transmitindo. Os pinos RO e DI representam saída da recepção e driver de entrada, respectivamente, e trabalham com níveis lógicos TTL (0 a 5V). Os pinos A e B são pinos de saída para o barramento e operam com tensão diferencial entre seus terminais.

Para realizar-se a transmissão devemos fazer com que DE assuma o nível lógico 1 (5V), habilitando o driver de transmissão e fazendo com que $\overline{RE}$ também assuma o nível lógico 1, desabilitando o driver de recepção. Ao final da transmissão o pino DE deve ser desabilitado, deixando o transceptor no modo de recepção. O controle dos valores para os pinos DE e $\overline{RE}$ é o controle de acesso ao barramento, previamente citado, que pode ser realizado via software.

Deve-se ter atenção para certas considerações elétricas importantes a respeito da utilização do barramento para comunicação no padrão RS-485. Devido ao fato de que os cabos possuem uma impedância característica, em transmissões a altas freqüências podem ocorrer reflexões do sinal em sua extremidade, provocando inconsistência dos dados. Para minimizar esses efeitos devemos adicionar resistores de terminação, de valores iguais à impedância característica do cabo para que ele se comporte como um cabo infinito, no dispositivo mais afastado da fonte principal de sinal, no nosso caso esta fone é o PC. Valores típicos usualmente utilizados são resistores de 120 Ohms. Outra consideração elétrica a ser assumida é que quando todos os dispositivos estão em modo de recepção, o nível lógico do barrramento pode ficar indefinido. Para garantir que o barramente fique sempre em nível lógico 1, devemos adicionar um resistor de *pull-up* ao pino A e um resistor de pull-down ao pino B. Tais resistores devem possuir o mesmo valor

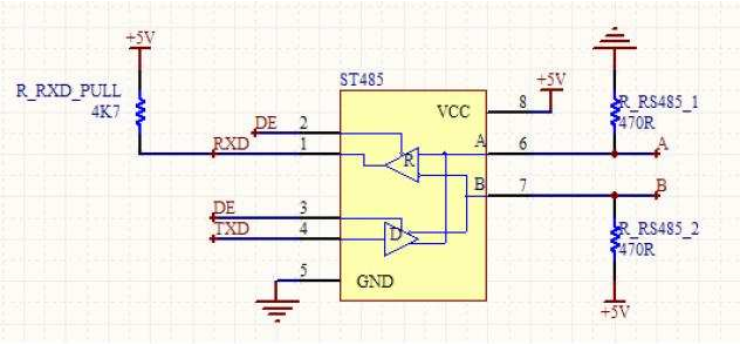


Figura 2.14: Conexões entre um AVR atmega8 e transceptores DS485 com controle pelo pino PD2

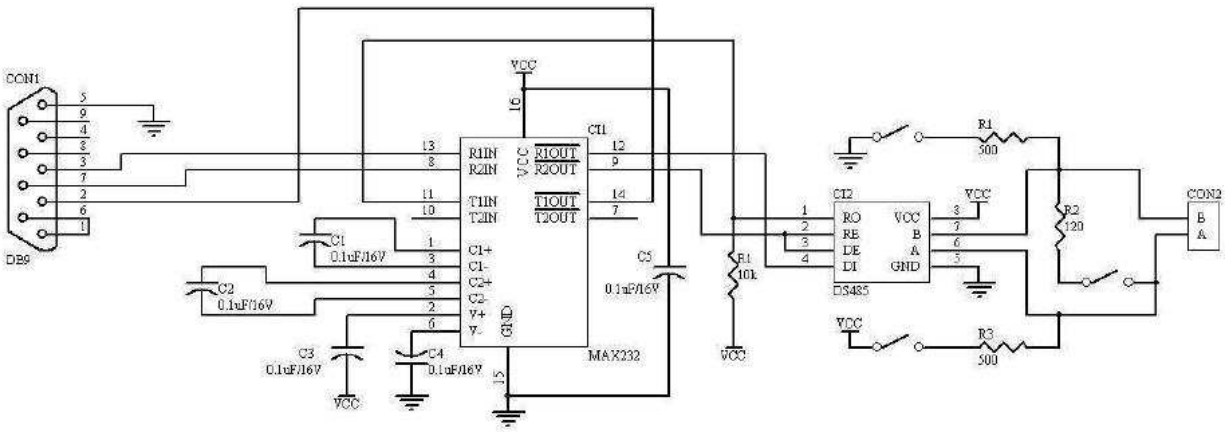


Figura 2.15: Conversor RS-232/RS-485 com controle pelo pino RTS[1]

para não alterarmos o balanceamento da linha de transmissão. Outro resistor de *pull-up* deve ser ligado ao pino RO a fim de evitar possíveis falsos recebimento de dados por parte do dispositivo conectado ao transceptor.

O circuito da figura 2.14 mostra as ligações entre transceptores RS-485 com o micro-controlador, AVR atmega8, onde o controle do acesso ao barramento é realizado pelo pino PD2 do micro-controlador. Já o circuito da figura 2.15 é o esquemático de um simples conversor RS-232/RS-485, necessário para realizar a conexão do PC com o barramento RS-485. Neste caso o pino que realiza o controle de acesso ao barramento é o pino RTS da porta serial do PC.

3 DESENVOLVIMENTO

3.1 INTRODUÇÃO

Para se realizar o desenvolvimento deste projeto partiu-se inicialmente da concepção da estrutura mecânica, usinando-se as peças a serem utilizadas. Com a estrutura mecânica definida, partiu-se para a modelagem geométrica concomitantemente com o desenvolvimento da parte eletrônica do projeto para que, por fim, fossem realizadas as simulações em MatLab.

3.2 CONCEPÇÃO MECÂNICA

Para a criação do robô tomou-se como base de inspiração alguns robôs já existentes, como o JoinMax e o Aibo da Sony. Dessas plataformas existentes tirou-se os graus de liberdade de suas patas e a amplitude de movimento de cada junta foi retirada das especificações do JoinMax da *MCII Robot*, para isso utilizou-se o digrama disponibilizado no site[1] do fabricante, figura 3.1.

Com auxilio da ferramenta de CAD, SolidWorks 2006, esboçou-se o desenho mecânico do robô, levando em conta algumas intenções de projeto, como suas dimensões uma vez que devia-se levar em conta um tamanho razoável para que toda a eletrônica pudesse ser embarcada, limitações mecânicas pois pensou-se em fixações dos motores nas quais as juntas tivessem as amplitudes desejadas para se movimentarem e os motores disponíveis. Tais esboços foram fundamentais para chegar ao projeto final, pois pôde-se verificar com maior clareza a liberdade de movimento e acertar nas dimensões de cada peça.

Após um consenso sobre a estrutura do robô, pensou-se em que material utilizar para cada parte. Como o robô deveria ser muito leve, optou-se por materiais como nylon e alumínio, tanto por suas baixas densidades como facilidade de usinagem. Assim, pernas e eixos foram feitos de nylon, já que se consegue com facilidade tarugos cilíndricos deste material e as outras partes deveriam ser usinadas em alumínio.

Com a definição do material, buscou-se no mercado os tarugos de acordo com a disponibilidade e com o preço. Após grande procura nas lojas de Brasília que vendem esses materiais encontrou-se os tarugos ideais para o projeto, pois conciliavam a idéia que se tinha e, facilitavam a usinagem e ainda tinham baixo custo.

Dado um concenso fez-se os desenhos técnicos de cada peça do robô, conforme anexo, que foram levadas ao SG-9 para analisar a possibilidade de serem feitas com o maquinário existente. Depois de aprovado pelo responsável da oficina, foi dado início à fabricação das peças, que embora tivesse simplificado o projeto para facilitar usinagem, ainda levou-se muito tempo para finalizar todas as peças.

Com o robô já montado foram feitos testes com os servos, afim de garantir se o torque seria suficiente para movimentar o robô, pois não se dispunha as especificações do fabricante. O resultado não foi satisfatório, os servos eram fracos e suas engrenagens muito frágeis. Foi necessário uma reformulação do projeto, e então decidiu-se pela compra de novos servos com o torque necessário para a movimentação do robô. Estimou-se o torque necessário para os novos motores.

Com o desenho em mãos, pôde-se estimar a massa do cachorro com o software CAD utilizado, chegando-se a uma massa de aproximadamente: 2.517 g, isso já utilizando a massa dos novos servos futaba s3003 e a massa do provável motor que seria comprado, o HS-755HB da Hitec. O braço de alavanca do máximo torque é o comprimento total da perna (183,97mm). Assim calculou-se o torque máximo que

[1] http://www.mciirobot.com/product/detail/DOG001.htm

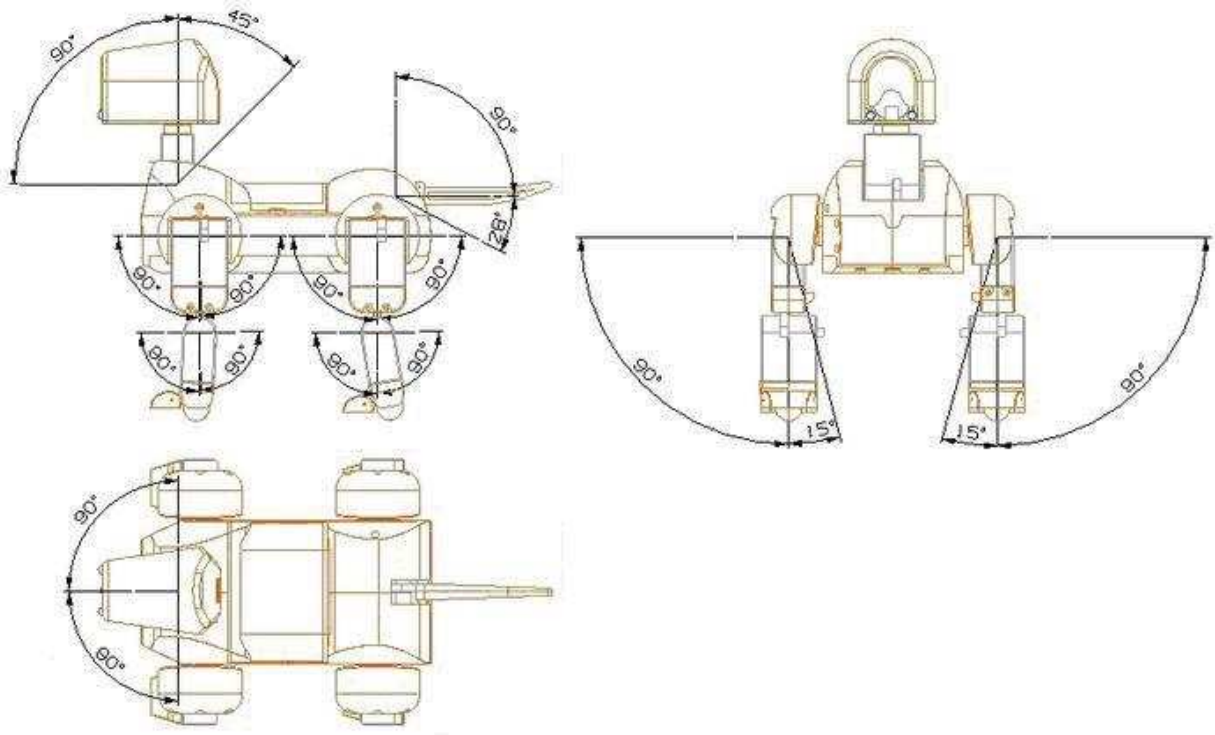


Figura 3.1: Amplitude do ângulo de cada junta

seria exigido do motor utilizando a equação 3.1.

$$T = F \cdot d = \frac{2.517,51}{4} \cdot 18,397 \approx 11,6kg \cdot cm. \tag{3.1}$$

Onde T é o valor calculado para o torque, considerando-se que a força exercida em cada uma das patas seria de igual valor, sendo que este valor é o peso total do robô divido pelo número de pernas, no caso 4, F é a força que atua na extremidade do braço de alavanca e d é o comprimento do braço de alavanca em centímetros.

Com este torque, encontrou-se que o melhor motor seria o HS-755HB da Hitec, que possui um torque adequado para o projeto (13,2kg· cm) e possui engrenagens de carbonite que são mais resistentes.

Com o novo motor escolhido pôde-se chegar a uma configuração final para o robô onde se redesenhou os eixos para se poder acoplar o novo motor e foram feitos adaptadores para fixação dos motores à estrutura antiga do robô. Chegando ao desenho final apresentado na figura 3.2.

Os eixos foram então refeitos para serem adaptados à nova estrutura.

Até a presente data, os servos que foram encomendados ainda não haviam chegado, dessa forma só apenas a imagem do robô como era anteriormente está disponível, com os servos de torque baixo, como mostra a figura 3.3

3.3 ELETRÔNICA

A alimentação do quadrúpede será provida por uma, ou mais, baterias de 12V. Os circuitos de alimentação dos componentes lógicos e de potência operam em tensões de 5V e 6V respectivamente, desta forma é necessário a utilização reguladores de tensão para alimentá-los. Até o presente momento a alimen-

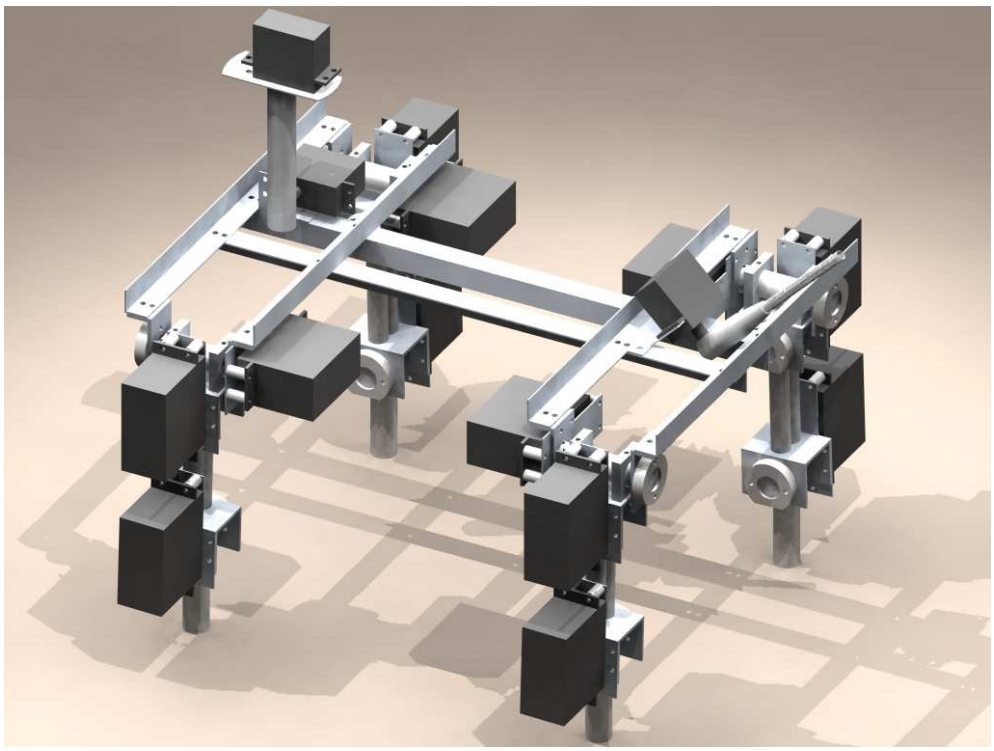


Figura 3.2: Estrutura do quadrúpede desenvolvido, desenho em solidworks

tação do quadrúpede está sendo fornecida por uma fonte de computador, da qual utiliza-se a saída de 12V (potência máxima de 400W) para alimentarmos os circuitos de regulação das tensões.

A necessidade do desenvolvimento de circuitos de alimentação independentes deve-se principalmente ao fato de que ao se submeter o circuito de potência a cargas elevadas o mesmo pode sofrer oscilações na sua tensão de saída e caso tal oscilação atingisse um valor abaixo do valor mínimo de operação dos micro-controladores estes poderiam vir a serem resetados, prejudicando o comportamento do quadrúpede. Outro fato que também motivou a criação de circuitos independentes foi a escolha da tensão de alimentação dos servos em 6V, enquanto os circuitos lógicos operam em 5V.

3.3.1 Circuitos de Potência

São chamados de circuitos de potência os circuitos desenvolvidos para alimentação dos servo motores. São de potência uma vez que são capazes de fornecer, sem sofrer queda significativa na tensão de saída, correntes de até 5A.

Tais circuitos possuem reguladores de tensão ajustável configurados de forma a fornecerem uma tensão de saída de até 6V. O circuito implementado foi retirado de [9], *datasheet* do próprio regulador utilizado, e é exibido na figura 3.4. O circuito de regulação consiste na escolha de resistores que fornecem a tensão de saída seguindo a seguinte fórmula:

$$V_o = 1,25(1 + \frac{R_2}{R_1}) \tag{3.2}$$

Lembrando que $\mathbf{V_o}$, é, necessariamente, menor do que $\mathbf{V_i}$ (tensão de entrada, no caso, 12V).

Analisando o circuito da figura 3.4, percebe-se a existência de capacitores, que são utilizados para filtrar o sinal e também para reduzir *ripples*, indesejáveis, pois estes podem vir a ser amplificados na saída, e também percebe-se a presença de diodos de proteção, coibindo a possível descarga dos capacitores nos

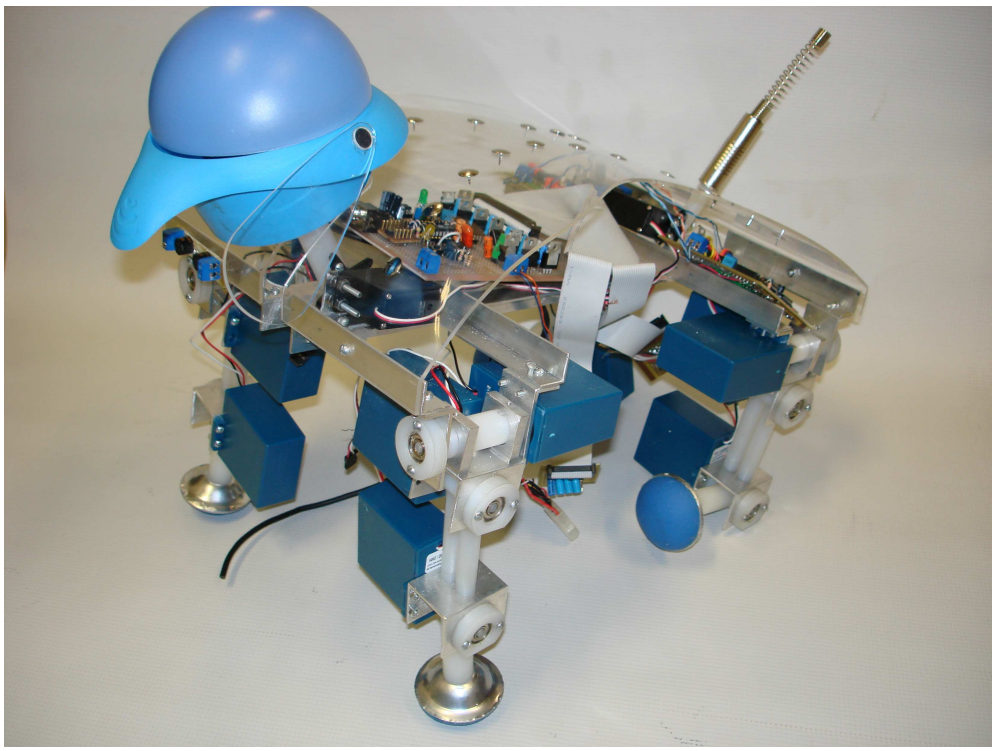


Figura 3.3: Foto da primeira versão desenvolvida, com servos de baixo torque

reguladores evitando que se danifiquem os componentes.

Foram utilizados dois tipos de reguladores de tensão ajustável, o LM338 e o LM350, ambos regulados com saída em 6V, capazes de fornecer até 5A e 3A respectivamente. O primeiro foi utilizado para alimentação dos servos das patas (1 regulador de tensão por pata) devido à maior carga que atua sobre tais servos e o LM350 foi utilizado para alimentar os servos da cabeça e rabo.

3.3.2 Circuitos de sinal, lógicos

Os componentes lógicos utilizados foram micro-controladores da Atmel, AVR modelo ATMega8 e transceptores RS-485 e RS-232. Estes operam com uma tensão de alimentação de 5V e são de baixo consumo, desta forma foram utilizados reguladores de tensão capazes de fornecer até 1A de corrente (caso tenham um bom sistema para dissipação de calor) e manter a tensão próxima a 5V, o regulador utilizado foi o LM7805. Como os componentes alimentados por estes reguladores são de baixo consumo não se fez necessário à utilização de dissipadores de calor e um regulador por placa foi o suficiente para fornecer a corrente necessária. A fim de se garantir uma alimentação de qualidade para o micro-controlador e demais componentes, utilizou-se capacitores como filtros nas alimentações.

3.3.3 Saídas PWM

Cada micro-controlador foi programado para gerar 3 sinais PWM diferentes para posicionamento dos servo motores. Para tal utilizou-se o *timer1* para gerar 2 PWMs e o *timer2* para gerar a terceira saída PWM. Cada PWM deve variar seu ciclo de trabalho entre, aproximadamente, 3% e 12%, no intuito de se alcançar os extremos do posicionamento dos servos.

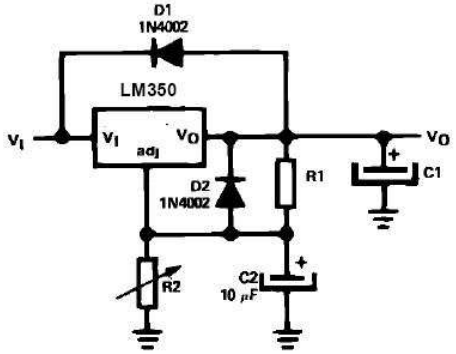


Figura 3.4: Circuito regulador de tensão variável

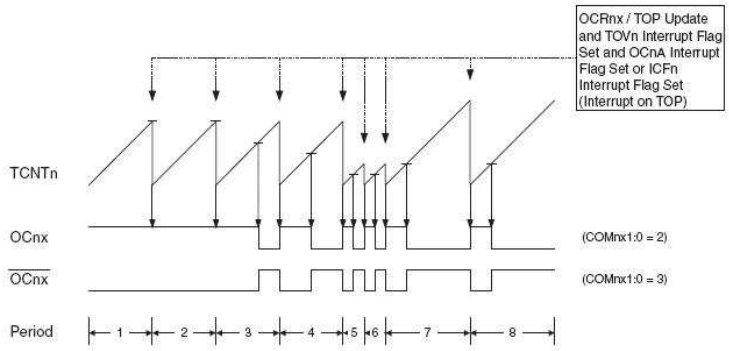


Figura 3.5: Modo *FAST-PWM* diagrama de tempo

3.3.3.1 Configuração do *timer1*

O timer1 foi configurado para operar no modo *FAST-PWM* com seu valor máximo em um registrador que pode ser alterado seu valor pelo usuário. O cálculo da freqüência de operação do mesmo é dado pela equação[2] 3.3.

$$freq = \frac{f_{clk\_I/O}}{N(TOP+1)} \tag{3.3}$$

Onde $\mathbf{N}$ é o valor do *prescaler* e $\mathbf{TOP}$ é o valor máximo para o contador definido no registrador associado.

Este *timer* possui 2 registradores (OCR1A e OCR1B) que são utilizados para comparação com o contador e foi configurado no modo que ao se igualar o valor do contador com o valor desses registradores o pino de saída correspondente a cada registrador é resetado sendo que tais pinos de saída são setados em 1 quando ocorre o estouro do contador, desta forma utilizando apenas um *timer* é possível gerar 2 sinais de PWMs com valores de ciclo de trabalho entre 0 e 100% de forma independente, apenas com a freqüência em comum como mostra a figura 3.5.

3.3.3.2 Configuração do *timer2*

Devido à baixa resolução do *timer2* e como só seria utilizado aproximadamente uma variação de 10% do ciclo de trabalho do PWM gerado na saída teve que se elaborar um lógica que nos permitisse ter uma resolução de pelo menos $1\,^{\circ}$, correspondente a 0,56%. Caso tivesse sido utilizado a mesma lógica que a

[2]Equação retirada do *datasheet* do micro-controlador [8]

utilizada no *timer1* se teria aproximadamente 26 valores diferentes para posicionar os servos, dando uma resolução de quase 7 °, o que deixaria o sistema com um movimento pouco preciso e com aspecto de não continuidade.

Sendo assim utilizou-se uma variável que conta o número de estouros do contador do *timer2* e definimos um número máximo permitido para esta variável retornando-a a zero ao atingir tal valor. Desta forma esta variável estaria funcionando como um *prescaler* para o *timer2*. Durante o período de cada estouro do contador do *timer* utilizou-se o registro de comparação juntamente com uma lógica para resetar-se o pino de saída associado ao *timer2*. Este pino de saída é setado em 1 assim que a variável contadora dos estouros do *timer2* retorna para o valor zero.

3.3.4 Comunicação entre módulos e PC (RS-485)

Como todo o processamento necessário para o cálculo da cinemática do quadrúpede é realizado no PC, faz-se necessário a comunicação entre o PC e os módulos (placas das pernas e da base). O PC é o responsável por enviar aos micro-controladores os valores de referência dos ângulos que cada servo de cada perna deve estar configurado, bem como os servos da cabeça e do rabo.

A comunicação com o PC também é utilizada pelo grupo responsável pela parte da robótica comportamental que divide a mesma plataforma construída e apresentada nesse projeto.

Inicialmente a comunicação utilizada usava o padrão RS-232, e era realizada entre o PC e apenas 1 micro-controlador, mono-ponto. Ao se necessitar de comunicação entre diversos dispositivos e com o PC tivemos que mudar para o padrão RS-485, que permite a comunicação multi-ponto.

O PC envia dois tipos de mensagens, uma de requisição de dados, chamada de mensagem de leitura e outra que é um comando, mensagem de escrita. As mensagens são enviadas a apenas um escravo por vez. Como a mensagem é lançada no barramento todos os dispositivos a ele conectados tem acesso a ela, porém, dado o protocolo utilizado, apenas será interpretada pelo escravo a mensagem que contiver seu endereço. Caso o escravo identifique seu endereço na mensagem e esta for uma mensagem de escrita ele irá executar as ações pré-estabelecidas e pré-acordadas entre mestre-escravo. Caso a mensagem seja de leitura o escravo deverá responder com os valores mais atuais da leitura dos sensores.

Uma observação importante é que para este trabalho não houve a necessidade de se enviar resposta de leitura dos valores de sensores uma vez que não existem tais sensores. Os sensores são utilizados pelo grupo da robótica comportamental.

3.3.4.1 Protocolo de comunicação

O protocolo utilizado é bastante simples. O PC envia uma mensagem com caracteres específicos que definem o destino e a ação que deve ser tomada pelo dispositivo/módulo receptor. Formato das mensagens[3]:

- Escrita: Wxx:nn:dd...dd
- Leitura: Rxx?

Onde:

- xx é o endereço do dispositivo de destino. 8 bits de endereços.
- nn é o número de *words* que é formado o dado.
- dd é o dado transmitido. Cada dado possui 8 bits.

[3]O formato dos dados é ascii-hexadecimal, a junção dos 2 caracteres definem o valor do byte em hexadecimal

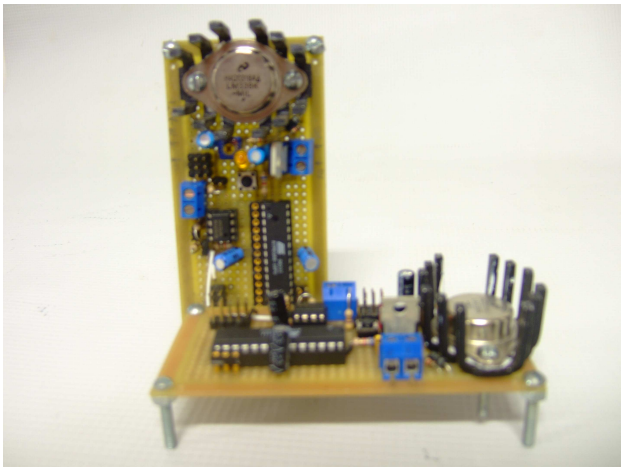


Figura 3.6: Placa de alimentação e acionamento dos servos da perna

Os demais caracteres, W,R,? e : são caracteres que compõe o protocolo e são utilizados como delimitadores dos campos e também são utilizados para validação da mensagem enviada.

3.3.5 Confecção das placas: base e pernas

Foram confeccionadas 5 placas, 1 para cada perna e uma placa base. Todas as placas foram construídas utilizando placas de matrizes perfuradas com fundo de cobre.

As placas das pernas devem fazer o interfaceamento entre o micro-controlador e o barramento de comunicação, gerar o sinal de posicionamento dos servos da perna em questão, além de alimentar os componentes (CIs) e os servos. Desta forma elas possuem apenas o circuito de potência para a alimentação dos servos das pernas, o circuito de baixa potência para a alimentação dos componentes lógicos, o circuito da comunicação serial do micro-controlador com o barramento RS-485 e o circuito da programadora. Os micro-controladores destas placas geram o sinal de acionamento para 3 servos independentemente e os valores do posicionamento dos servos são oriundos do PC. A figura 3.6 mostra a foto de uma das placas.

Já a placa de controle dos servos da cabeça e do rabo, além de todas as funcionalidades apresentada pelas placas das pernas, possui ainda o circuito que realiza o interfaceamento entre o PC e o barramento 485, ou seja, realiza a conversão RS-232/RS485, também possui leds RGB que são acionados por comandos oriundos do PC que identificam o estado emocional do quadrúpede. A figura 3.7 mostra a foto desta placa.

3.4 PROGRAMAÇÃO *IN CIRCUIT*

Todas as placas construídas (figura 3.8) possuem o circuito que permite que o micro-controlador seja programado *in-circuit*, ou seja, possui o circuito de gravação na própria placa. A gravação do micro-controlador nada mais é que a transferência do programa desenvolvido no PC, por sua porta paralela, para a memória memória *flash* do micro-controlador.

Os pinos de gravação são disponibilizados juntamente com um pino de alimentação, necessário para alimentar o *buffer* da gravadora. O circuito utilizado foi o da gravadora BSD [10] e seu esquemático é mostrado na figura 3.9.

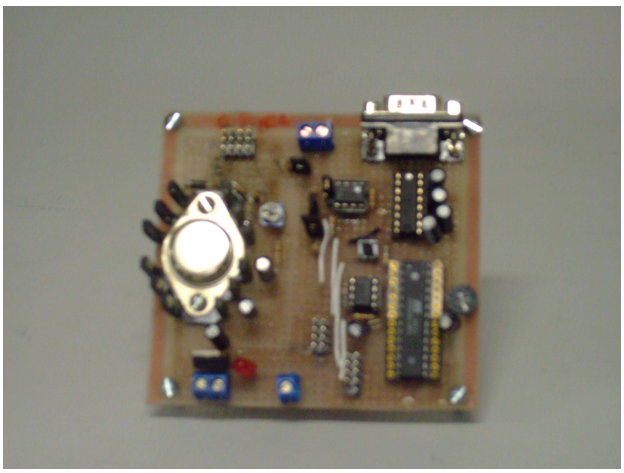


Figura 3.7: Placa de alimentação e acionamento dos servos da cabeça e do rabo

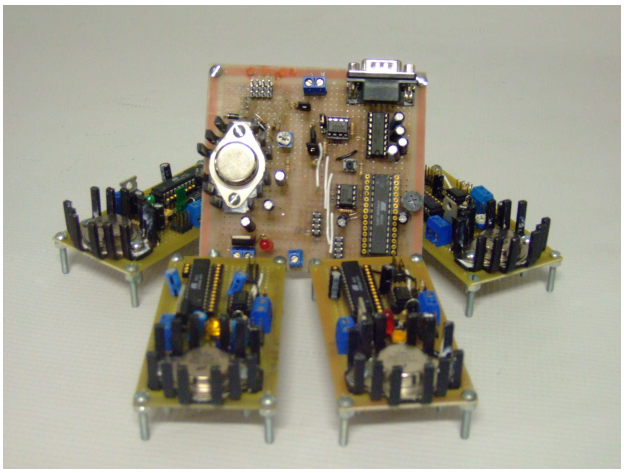


Figura 3.8: Placas desenvolvidas: base e pernas

3.5 SIMULAÇÕES EM MATLAB

Para simular os movimentos do robô, calcular sua cinemática direta e inversa foi utilizada uma ferramenta já consagrada na engenharia que é o MatLab. Utilizamos como referência [11] para auxílio na programação em MatLab.

Com este software é possível conceber uma interface amigável onde se pode escolher o posicionamento das pernas usando a posição do ponto de apoio (cinemática inversa) ou informando os ângulos das juntas (cinemática direta).

Nesta seção explicaremos de uma maneira geral como foi feito o programa, qual a matemática utilizada e como usar a interface do mesmo. A programação dos movimentos do robô ficou separada em módulos, cada um responsável por uma parte do programa.

O arquivo que contém a modelagem de Denavit-Hartenberg está em $matrizes.m$ (Anexo I), neste arquivo implementaremos todas as matrizes para representar o ponto de apoio com relação a um sistema de coordenadas do $mundo$.

Partimos inicialmente da seguinte configuração para os eixos das pernas, mostrada na figura 3.10 ,onde $a3$ e $a2$ são os comprimentos do braço e antebraço, respectivamente. Para o caso em questão o comprimento $a1$ é zero, uma vez que os eixos das juntas do ombro e da perna têm origem no mesmo ponto.

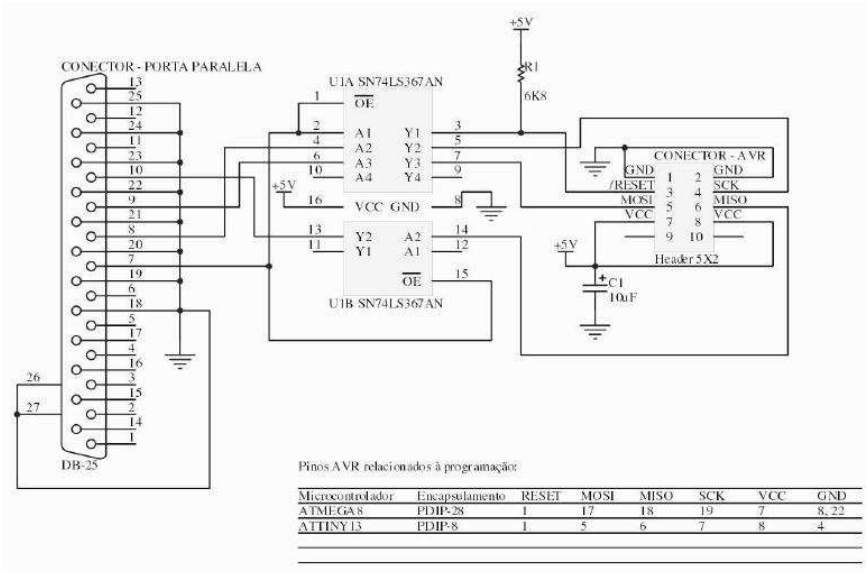


Figura 3.9: Circuito da gravadora BSD do AVR Atmega8

Com essa configuração foram montandas as matrizes de DH, conforme a equação 2.27, seguindo a figura 2.5.

Temos então as seguintes matrizes:

$$\mathbf{A}_2^3 = \begin{bmatrix} cos(q3) & -sin(q3) & 0 & -a3 \cdot cos(q3) \\ sin(q3) & cos(q3) & 0 & -a3 \cdot sin(q3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

que relaciona o eixo de coordenadas do *end-effector* ao eixo de coordenadas da junta $q2$.

$$\mathbf{A}_1^2 = \begin{bmatrix} cos(q2) & -sin(q2) & 0 & -a2 \cdot cos(q2) \\ sin(q2) & cos(q2) & 0 & -a2 \cdot sin(q2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

que relaciona o eixo de coordenadas da junta $q2$ ao eixo de coordenadas da junta $q1$.

$$\mathbf{A}_0^1 = \begin{bmatrix} cos(q1+\frac{\pi}{2}) & 0 & -sin(q1+\frac{\pi}{2}) & 0 \\ sin(q1+\frac{\pi}{2}) & 0 & cos(q1+\frac{\pi}{2}) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -sin(q1) & 0 & -cos(q1) & 0 \\ cos(q1) & 0 & -sin(q1) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

que relaciona o eixo de coordenadas da junta $q1$ ao eixo de coordenadas da base da perna. A matriz em 3.6, embora pareça em formato diferente, foi obtida a partir da relação de DH apresentada na matriz 2.27. O que aconteceu foi que para se ter os eixos de coordenadas equivalentes, $q1$ deveria ser escrito na forma $q1 + \frac{\pi}{2}$, pois esse ângulo de $\frac{\pi}{2}$ radianos é necessário para que as coordenadas dos dois sistemas sejam equivalentes. Isso pode ser verificado realizando a torção de $\frac{-\pi}{2}$ no eixo x da junta de $q2$ e então verificamos que falta uma rotação de $\frac{\pi}{2}$ radianos em z para que os dois sistemas sejam equivalentes.

Com a multiplicação destas 3 matrizes consegue-se representar um ponto do *end-effector* no sistema de coordenadas de $q1$. Com essa representação partiu-se para uma transformação que leve da junta $q1$ de cada pata até o centro do robô e para isso, partiu-se da figura 3.11.

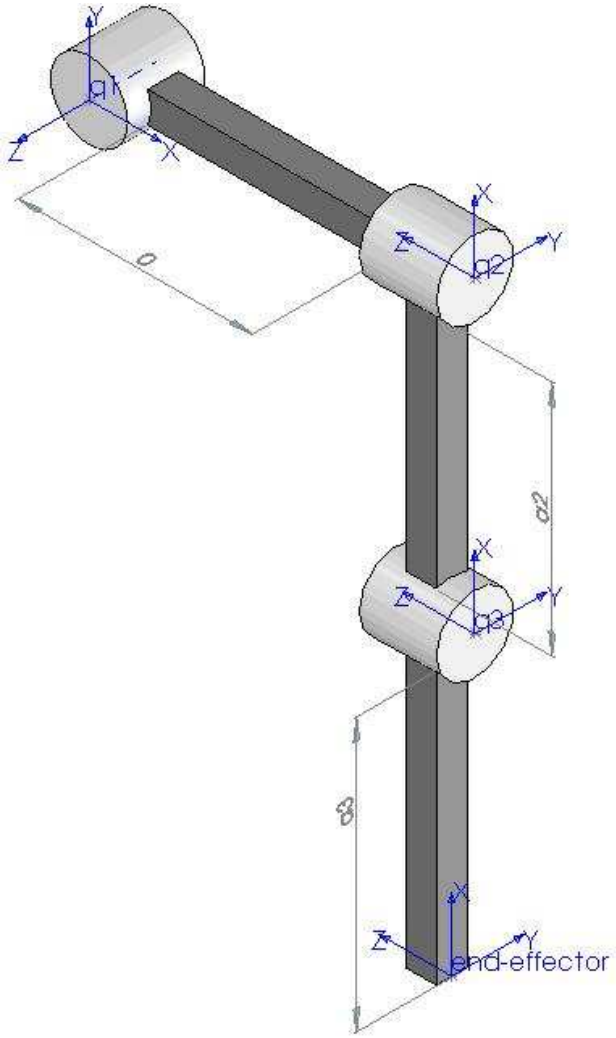


Figura 3.10: Eixos coordenadas das juntas das pernas

Com essa distribuição dos eixos chega-se a uma matriz geral para transformar as coordenadas das pernas para as coordenadas do centro do robô, bastando transladar cada perna nos eixos x e z:

$$\mathbf{A}_0^c = \begin{bmatrix} 1 & 0 & 0 & (-1)^n \cdot d_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & (-1)^k \cdot d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

onde n e k são valores que dependem de qual perna que está-se transformando.

Agora deve-se realizar a translação do centro do robô para um sistema de coordenadas fixo do $mundo$.

Com a figura 3.12 chega-se à seguinte matriz:

$$\mathbf{T}_w^c = \begin{bmatrix} 1 & 0 & 0 & Px \\ 0 & 1 & 0 & Pz \\ 0 & 0 & 1 & -Py \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.8}$$

que são as translações que devem ser feitas do centro do robô para o sistema fixo.

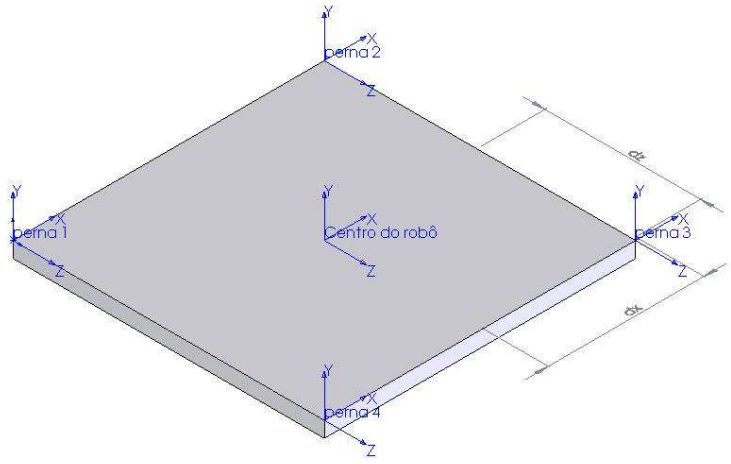


Figura 3.11: Eixos da junta de q1 e centro do robô

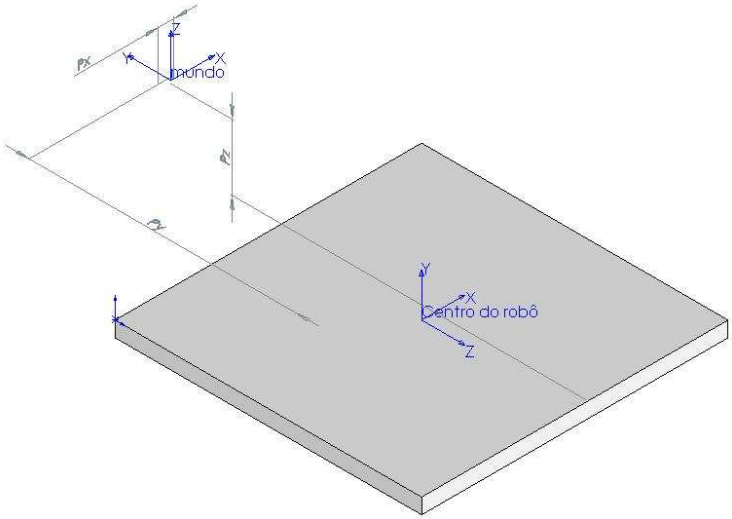


Figura 3.12: Eixos do centro do robô com relação a um sistema fixo

$$\mathbf{R}_w^c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\frac{\pi}{2}) & -sin(\frac{\pi}{2}) & 0 \\ 0 & sin(\frac{\pi}{2}) & cos(\frac{\pi}{2}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

que consiste em uma rotação de $\frac{\pi}{2}$ do eixo fixo para o eixo do centro do robô.

Por fim, utilizou-se uma matriz de orientação *(Yaw-Pitch-Roll)*, equação 3.10, para a orientação do centro do robô, onde $q6$ é o ângulo de rotação em torno do eixo z, $q5$ é em torno de y e $q4$ é em torno de x em relação ao sistema fixo.

$$\mathbf{R_{YPR}} = \begin{bmatrix} cos(q6) & -sen(q6) & 0 & 0 \\ sen(q6) & cos(q6) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} cos(q5) & 0 & sin(q5) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(q5) & 0 & cos(q5) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(q4) & -sin(q4) & 0 \\ 0 & sin(q4) & cos(q4) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

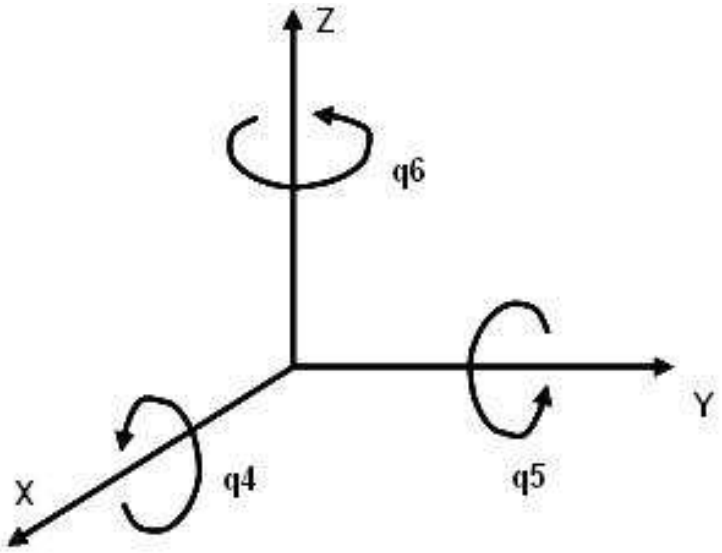


Figura 3.13: Sistema de orientação *Yaw-Pitch-Roll*

Se forem multiplicadas todas as matrizes, iniciando pela 3.10, até a 3.4. Tem-se a cinemática direta concluída, e a última coluna da matriz resultante representa a posição do *end-effector* em relação ao sistema de coordenadas fixo do $mundo$.

Para o cálculo da cinemática inversa de posição do robô criou-se um arquivo chamado $calcularangulos.m$ (Anexo II), neste arquivo faz-se o uso de métodos iterativos para poder calcular a posição do ponto de apoio. Aqui utiliza-se três métodos para poder-se decidir qual método melhor que se adapta ao modelo em questão, o método da Transposta, da Pseudoinversa e do *Damped Least Squares*. Para mensurar qual desses métodos é o melhor para a aplicação, analisou-se a velocidade de convergência e qual método mostra menor oscilação e *jerks*. Para isso utilizou-se uma função que calcula o tempo do inicio do algoritmo até o ponto de convergência do mesmo e ainda plotou-se um gráfico de como se comporta a variação dos ângulos pelo o algoritmo, pois grandes oscilações implicam em uma convergência ruim.

Logo no início do algoritmo é calculada a matriz Jacobiano para aquela configuração da pata, conforme equação 2.29, sendo o vetor $\mathbf{s}$ tomado como a quarta coluna da matriz que é obtida na cinemática direta:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \mathbf{Sx}}{\partial \mathbf{q1}} & \frac{\partial \mathbf{Sx}}{\partial \mathbf{q2}} & \frac{\partial \mathbf{Sx}}{\partial \mathbf{q3}} \\ \frac{\partial \mathbf{Sy}}{\partial \mathbf{q1}} & \frac{\partial \mathbf{Sy}}{\partial \mathbf{q2}} & \frac{\partial \mathbf{Sy}}{\partial \mathbf{q3}} \\ \frac{\partial \mathbf{Sz}}{\partial \mathbf{q1}} & \frac{\partial \mathbf{Sz}}{\partial \mathbf{q2}} & \frac{\partial \mathbf{Sz}}{\partial \mathbf{q3}} \end{bmatrix} \tag{3.11}$$

onde $\mathbf{Sx}$ é o primeiro elemento da quarta coluna da matriz que é resultado da cinemática direta, $\mathbf{Sy}$ é o segundo elemento e $\mathbf{Sz}$ é o terceiro.

Em seguida calcula-se o vetor deslocamento que é a diferença entre o ponto objetivo e o ponto atual:

$$\mathbf{D} = (\mathbf{P}_{desejado} - \mathbf{P}_{atual}) \tag{3.12}$$

Esse vetor deslocamento é normalizado para uma distância possível de ser alcançada pela perna. Pois se assim mesmo o ponto estiver distante do ponto de apoio o algoritmo buscará a melhor solução para esse caso.

Depois disso, é feita uma estimativa das variações de cada ângulo para aquele vetor deslocamento utilizando um dos métodos citados. Ao utilizar o método da transposta, ter-se-á que a atualização dos valores dos ângulos das juntas será dada por 2.36. Com o método da pseudoinversa utiliza-se a equação 2.39 e para o *Damped Least Squares* usa-se a equação 2.40.

Com esse resultado são atualizados os valores dos ângulos com essas variações e calcula-se a nova posição da perna. É feito um cálculo de erro quadrático entre a nova posição e a desejada:

$$erro = \| (\mathbf{P}_{desejado} - \mathbf{P}_{atual}) \| \tag{3.13}$$

Se o erro for maior que uma determinada tolerância esse algoritmo é repetido, recalculando-se o novo Jacobiano para a nova configuração de ângulos, novo vetor deslocamento com a nova posição atual, as novas variações dos ângulos das juntas por um dos métodos escolhidos, atualizam-se os ângulos e calcula-se novamente o erro, repetindo todo o algoritmo até que esse erro seja menor que a tolerância ou estoure o número máximo de iterações.

Esse arquivo recebe como entradas um fator, que é utilizado no método do *Damped Least Squares*, a posição desejada do centro do robô, qual perna será posicionada, sua posição desejada, ângulos das juntas da perna naquele momento e os ângulos desejados da orientação do centro do robô. Desse modo, consegue-se fechar a cinemática inversa de posição tendo como saída a nova configuração dos ângulos das juntas da perna para a posição desejada.

o algoritmo acima descrito pode ser entendido da seguinte forma:

1. Calcular erro de posição. Se erro < tolerância sai do algoritmo, senão continua o algoritmo.
2. Calcular Jacobiano
3. Calcular $\mathbf{\Delta E}$ = $\mathbf{E_{objetivo}}$ - $\mathbf{E_{atual}}$
4. Calcular o Δq com um dos métodos (Transposta, Pseudoinversa ou *Damped Least Squares*)
5. Aplicar Δq a todo o sistema
6. Calcular a nova posição das pernas
7. Calcular o novo erro dessa posição
8. Se erro > tolerância repetir a partir de 2 até erro < tolerância ou atingir o número máximo de iterações.

Uma outra função interessante é a $calcularangulosdocentro.m$ (Anexo III), essa função tem como objetivo fazer uma inversa para o centro do robô, ou seja, escolhe-se um plano de apoio para as pernas do robô e dá-se uma configuração de ângulos para as suas patas, com isso, deve-se encontrar a posição e a orientação do centro do robô com relação ao $mundo$. O Jacobiano dessa função é então diferente da situação que se tinha anteriormente.

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \mathbf{Sz}}{\partial \mathbf{q4}} & \frac{\partial \mathbf{Sz}}{\partial \mathbf{q5}} & \frac{\partial \mathbf{Sz}}{\partial \mathbf{q6}} & \frac{\partial \mathbf{Sz}}{\partial \mathbf{Px}} & \frac{\partial \mathbf{Sz}}{\partial \mathbf{Py}} & \frac{\partial \mathbf{Sz}}{\partial \mathbf{Pz}} \end{bmatrix} \tag{3.14}$$

O Jacobiano é calculado levando em conta apenas as coordenadas de z, pois só é interessante que o ponto de apoio fique em contato com um plano em z.

Dessa forma, o vetor deslocamento é calculado também levando em conta apenas a coordenada z da perna.

$$d = \| (\mathbf{z}_{desejado} - \mathbf{z}_{atual}) \| \tag{3.15}$$

Com isso, encontrar-se-á os valores de $q4$, $q5$, $q6$, Px, Py e Pz que atendam à configuração de ângulos das juntas das pernas e o plano de apoio das pernas.

O calculo da atualização desses valores se dá de maneira diferente aqui. Ainda utiliza-se um dos métodos propostos (Transposta, Pseudoinversa ou *Damped Least Squares*) para o cálculo das novas atualizações em cada perna, contudo, como a posição e orientação dependem de todas as pernas, o valor da variação total será dado aqui pela soma ponderada das variações causadas por cada perna, onde o fator de ponderação é dado pela razão entre o erro de posicionamento de cada perna e a soma dos erros de todas as pernas. Dessa forma, a perna que tiver com pior posicionamento terá um peso maior favorecendo corrigir sua posição.

$$\begin{aligned} \Delta_{total} = \\ = \frac{erro_{perna1}}{erro_{soma}} \cdot \Delta_{perna1} + \frac{erro_{perna2}}{erro_{soma}} \cdot \Delta_{perna2} + \cdots \\ \cdots + \frac{erro_{perna3}}{erro_{soma}} \cdot \Delta_{perna3} + \frac{erro_{perna4}}{erro_{soma}} \cdot \Delta_{perna4} \end{aligned} \tag{3.16}$$

Essa função foi criada devido a uma necessidade de representar as emoções do robô, pois para cada emoção tínhamos uma configuração de ângulos para as patas, cabeça e rabo e um plano das patas, devíamos então encontrar qual seria o novo centro do robô.

O arquivo $jacobiano.m$ (Anexo IV) é o responsável para calcular o Jacobiano para a atual configuração de ângulos. Para deixar o algoritmo mais veloz, esta função fica separada em duas seções, uma que calcula o Jacobiano específico para posicionamento das patas e outra que calcula o Jacobiano para encontrar o centro do robô na função $calcularangulosdocentro.m$.

A função $circulo.m$ (Anexo V) descreve uma trajetória circular em torno da origem, é uma função muito simples, nela apenas é calculada as coordenadas novas para o centro do robô em uma circunferência com centro em (0;0;-0,2).

$$\mathbf{Px^2 + Py^2 = R^2} \tag{3.17}$$

Incrementando-se o valor de Py a cada passagem dentro do intervalo de $[-R, R]$ e com 3.14 encontra-se o novo valor para Px.

Assim, calcula-se o conjunto de ângulos das juntas para os novos centros com a função $calcularangulos.m$ e a cada mudança o robô é redesenhado.

A função $andar\_frente.m$ (Anexo VI) é uma função em que o robô executa um passo de comprimento fixo na direção do eixo Y do $mundo$. A geração desse passo foi feita empiricamente, sem grande desenvolvimento, utilizando apenas a função $calcularangulos.m$ para poder-se encontrar a nova configuração das juntas a cada posição do passo. Para gerar o passo de cada perna pega-se apenas 2 pontos, um dos pontos se localiza na metade do comprimento do passo e a uma altura predefinida e o outro é o ponto final do passo, que se localiza no mesmo plano em que estava no inicio mas a uma distância do ponto inicial que é equivalente ao comprimento do passo.

Outras funções utilizadas nesse programa são para controle da interface gráfica, como ler as variáveis que o usuário coloca na tela, desenhar o robô na tela ou mostrar variáveis de saída para cada caso, como, por exemplo, se for escolhida a opção de posição (cinemática inversa de posição) tem-se uma atualização dos valores no campo dos ângulos, mostrando os ângulos para aquela nova posição.

A figura 3.14 é um exemplo da interface que foi feita para se poder modelar o robô. Com esta interface tem-se a opção de entrar com valores de posição das pernas do robô utilizando a cinemática inversa de posição para se encontrar a configuração das juntas, ou com valores de ângulos utilizando a cinemática direta de posição. Ainda é possível mudar o centro do robô, tem-se ainda posições pré-definidas que representam alguns comportamentos (estados emocionais) do cachorro. Sempre que se escolher uma dessas opções os resultados aparecem nos outros campos. Nessa situação acima, por exemplo, escolhe-se a posição 3 do

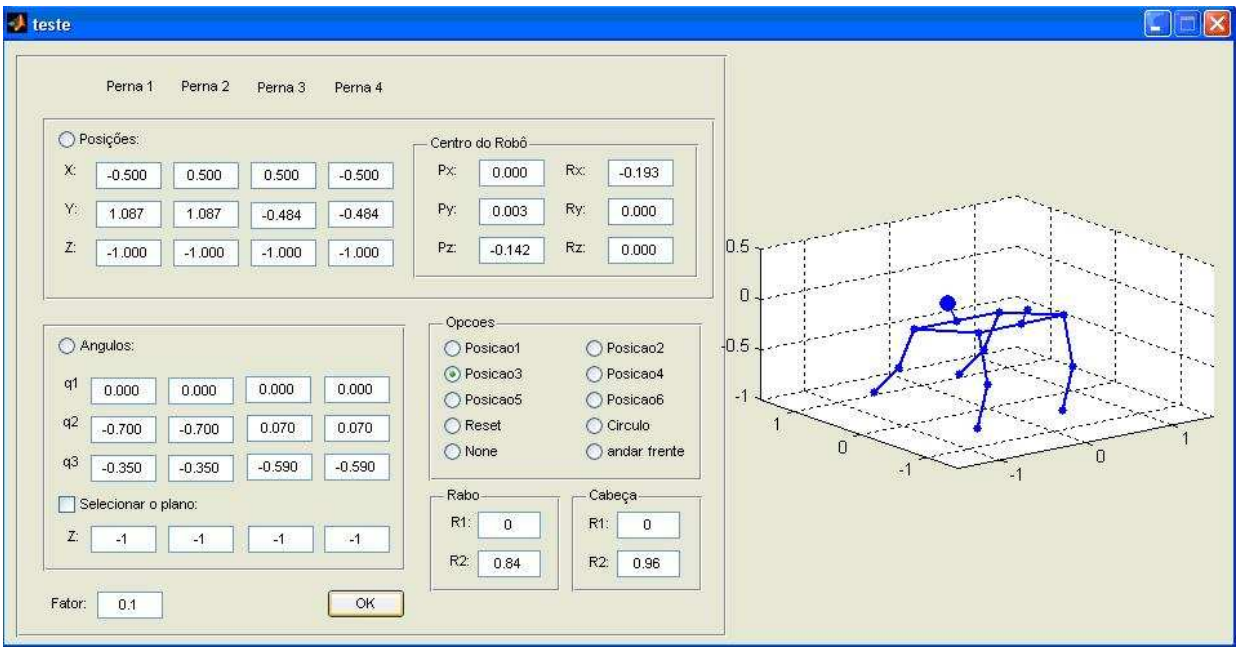


Figura 3.14: Simulador feito em MatLab

robô e nos campos de ângulos e posição aparecem os valores da configuração dos ângulos das juntas e as coordenadas das pernas do robô.

4 RESULTADOS EXPERIMENTAIS

4.1 INTRODUÇÃO

A parte experimental serviu para que fosse possível avaliar o desenvolvimento do projeto.

A comunicação serial foi testada de modo a validarmos o protocolo utilizado, verificando se todos os módulos estão recebendo corretamente as informações enviadas pelo programa principal de maneira correta e também verificar se os mesmos estão interpretando de forma certa tais mensagens.

A calibração dos servos motores foi realizada a fim de se definir os valores do *dutycicle* dos PWMs que representam os limites máximo e mínimo do posicionamento dos servos, além do posicionamento do seu ponto central. Testes de verificação quanto à força dos servos também foram avaliados de modo a termos ciência que os mesmos serão capazes de suportar a carga do robô. Além de testes de velocidade do posicionamento dos servos.

4.2 CALIBRAÇÃO DOS SERVOS

A calibração aqui referida foi realizada alterando o sinal de controle, variando o PWM, dos servos motores a fim de se descobrir os valores extremos e central do posicionamento dos mesmos. Para tal utilizou-se o programa de controle e interface entre o PC e o robô, fazendo com que o mesmo enviasse comandos de posicionamento dos servos até encontrarmos valores que satisfizessem a questão da posição dos servos.

A tabela 4.1 mostra os valores, em hexadecimal, para os limites[1] e posicionamento central dos servos da cabeça e rabo, uma vez que os servos das pernas não estão disponíveis até o momento.

4.3 SIMULAÇÕES REALIZADAS EM MATLAB

Nesta seção foram realizadas diversas simulações que nos forneceram informações a respeito do comportamento da implementação dos algoritmos para a cinemática direta e inversa de posição em diversas situações.

Algumas definições de termos aqui utilizados:

- q1: ângulo, em radianos, da junta do ombro de movimento transversal em relação à base.

[1]Os resultados obtidos para os valores do posicionamento dos servos são definidos por limitações mecânicas, exceto o seu valor central, obtido por interpolação entre os extremos

Tabela 4.1: Valores calibrados para o posicionamento dos servos

	Extremidade da esquerda	**Posição central**	**Extremidade da direita**
Servo da Cabeça	0x3D	0x77	0xC3
Servo do Pescoço	0x1D	0x4E	0x80
Servo do Rabo	0x4B	0x77	0xA3

Tabela 4.2: Valores de convergência com o método da Transposta do Jacobiano

	Perna 1	**Perna 2**	**Perna 3**	**Perna 4**	**Média**
Tempo de Execução (s)	0,482032	0,077651	0,010310	0,010311	0,145076
No. de Iterações	93	93	93	93	93
Erro de posicionamento (u.c.)	9,8179e-004	9,8179e-004	9,8179e-004	9,8179e-004	9,8179e-004

Tabela 4.3: Valores de convergência com o método da Pseudo-inversa

	Perna 1	**Perna 2**	**Perna 3**	**Perna 4**	**Média**
Tempo de Execução (s)	0,505299	0,074210	0,007547	0,007601	0,14866425
No. de Iterações	14	14	14	14	14
Erro de posicionamento (u.c.)	2,4662e-004	2,4662e-004	2,4662e-004	2,4662e-004	2,4662e-004

- q2: ângulo, em radianos, da junta do ombro de movimento longitudinal em relação à base.
- q3: ângulo, em radianos, da junta do joelho, movimento longitudinal em relação à base.
- q4: ângulo, em radianos, da rotação em torno x, coordenadas $mundo$, do plano da base.
- q5: ângulo, em radianos, da rotação em torno y, coordenadas $mundo$, do plano da base.
- q6: ângulo, em radianos, da rotação em torno z, coordenadas $mundo$, do plano da base.
- Px: Posição no eixo x, coordenadas $mundo$, do centro do robô.
- Py: Posição no eixo y, coordenadas $mundo$, do centro do robô.
- Pz: Posição no eixo z, coordenadas $mundo$, do centro do robô.

4.3.1 Simulação da Cinemática Inversa - Caso 1

Realizou-se simulações utilizando a inversa com o MatLab para se verificar qual método é o mais eficaz. Nos três métodos utilizados realizou-se a mesma simulação, alterando apenas a posição do centro robô através do eixo z, fazendo com que o mesmo movimentasse seu centro na direção vertical (para cima e para baixo) sem inclinar o plano de sua base.

Pela análise das tabelas 4.2, 4.3, 4.4, 4.5 e figuras 4.1, 4.2, 4.3 e 4.4, pode-se tirar conclusões quanto aos métodos utilizados e o mais adequado ao modelo do cachorro. Quanto ao erro de posicionamento todos

Tabela 4.4: Valores de convergência com o método *Damped Least Squares* fator 0,1

	Perna 1	**Perna 2**	**Perna 3**	**Perna 4**	**Média**
Tempo de Execução (s)	0,393085	0,066829	0,005779	0,005241	0,1177335
No. de Iterações	14	14	14	14	14
Erro de posicionamento (u.c.)	3,6196e-004	3,6196e-004	3,6196e-004	3,6196e-004	3,6196e-004

Tabela 4.5: Valores de convergência com o método *Damped Least Squares* fator 0,5

	Perna 1	**Perna 2**	**Perna 3**	**Perna 4**	**Média**
Tempo de Execução (s)	0,402396	0,070705	0,009142	0,008944	0,12279675
No. de Iterações	30	30	30	30	30
Erro de posicionamento (u.c.)	8,3800e-004	8,3800e-004	8,3800e-004	8,3800e-004	8,3800e-004

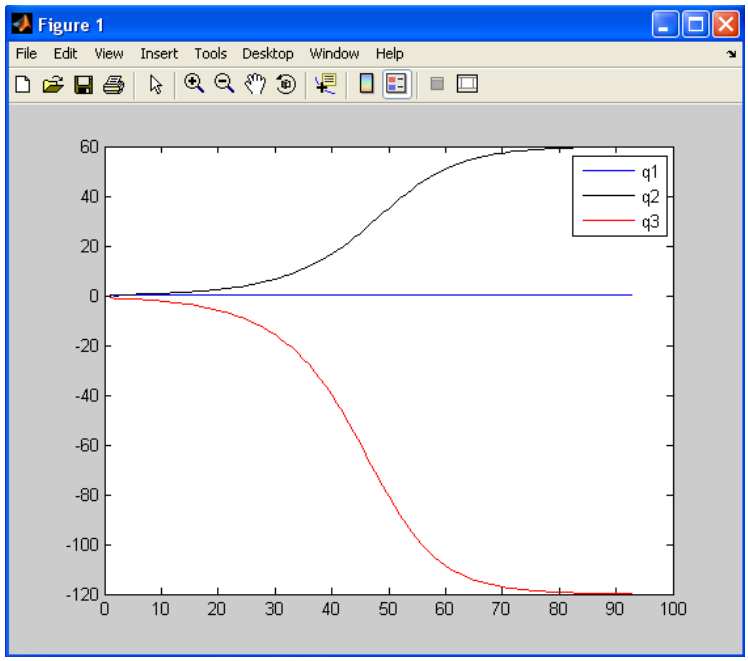


Figura 4.1: Gráfico do cálculo dos ângulos das juntas - método Transposta do Jacobiano

os três testes foram muito satisfatórios, não há diferença significante entre eles. Quando comparamos o tempo de convergência é interessante notar a relação com o número de iterações, porque da relação entre esses dados podemos retirar qual método é mais econômico computacionalmente.

O método da pseudoinversa se mostrou como o mais lento, contudo com número de iterações igual ao do *Damped Least Squares* com fator de 0,1, que se mostrou mais rápido dos três. Com isso, pode-se concluir que o método da pseudoinversa é o que consome mais máquina por iteração (0,0106 segundo/iteração). Além disso, pode-se concluir pela análise do gráfico que com a pseudoinversa tem-se um sobre-passo maior na convergência do resultado, que se deve à instabilidade do método próximo a regiões de singularidades.

O método da Transposta se mostrou muito interessante para o modelo em questão, possui uma boa convergência, pois não se vê um sobre-passo em seu gráfico, figura 4.1, ao contrário do observado nos gráficos, figuras 4.3 e 4.4, do Damped Least Squares e no gráfico 4.2 da pseudoinversa. Além disso, possui uma relação muito boa entre o tempo decorrido e o número de iterações (0,0016 segundo/iteração), o que representa menor gasto computacional por iteração com esse método, menor até mesmo que com o *Damped Least Squares* com o fator de 0,5 (0,0041 segundo/iteração). Apesar disso, esse método ainda se mostrou como o mais lento para convergir, precisando de um número de iterações grande para que pudesse convergir.

O método do *Damped Least Squares* se mostrou mais satisfatório para nosso caso, tem o menor tempo de convergência e o menor número de iterações para chegar ao resultado, além disso, pode-se controlar essa convergência alterando o valor do fator, que para menores valores temos uma convergência mais rápida,

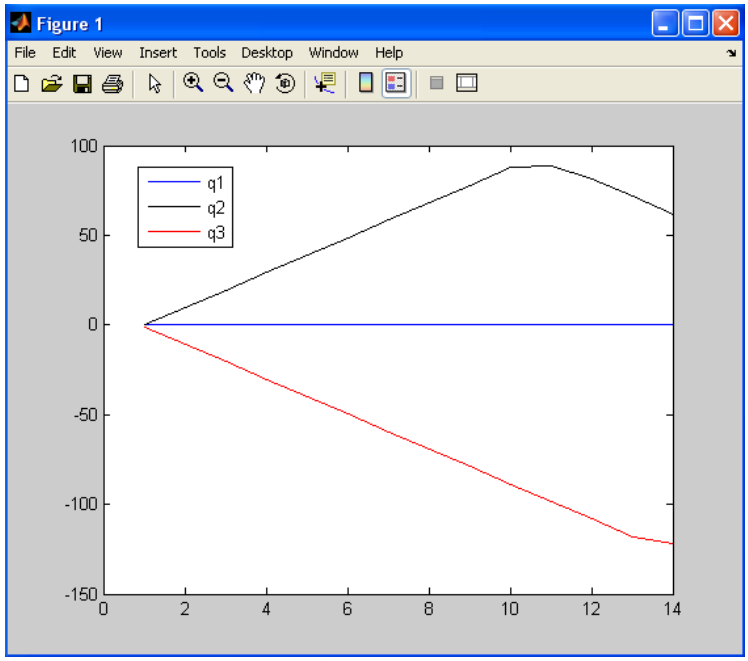


Figura 4.2: Gráfico do cálculo dos ângulos das juntas - método da Pseudo-inversa

assemelhando a uma pseudoinversa e, com valores mais altos, um método que se comporta melhor próximo às singularidades, convergindo mais suavemente para o resultado conforme podemos ver com o gráfico da figura 4.4, com fator de 0,5. Dessa forma, optou-se pelo *Damped Least Squares* como melhor método para nosso modelo, por possuir maior velocidade de convergência, ser eficaz computacionalmente e, de acordo com a escolha do fator, convergir mais suave ao resultado.

Pela figura 4.5 se vê a interface do programa destacando o resultado da cinemática inversa de posição ao variarmos a altura do plano do centro do robô de z=0 para z=-0,5.

As simulações para a inversa quanto ao centro do robô funcionaram muito bem, ao se colocar uma das configurações relacionadas ao comportamento do robô o cálculo da nova posição do centro do robô foi bem satisfatória. O método *Damped Least Squares* mostrou-se o mais satisfatório, uma vez que o cálculo da inversa foi rápido e teve pouca oscilação ao convergir.

4.3.2 Simulação da Cinemática Inversa - Caso 2

Dado uma configuração para os ângulos das juntas e fixando-se o plano de apoio das pernas deseja-se obter a posição e orientação do centro do robô.

Neste caso foi feita uma simulação utilizando a configuração de ângulos obtida na parte anterior onde q3 = -2.095 radianos, q2 = 1.048 radianos e q1 = 0. Utilizou-se ainda como plano de apoio para as patas o plano z = -1. Desse modo será utilizada a função $calcularangulosdocentro.m$ que calculará as novas coordenadas do centro do robô para essa configuração de ângulos e plano de apoio das patas.

Como pode-se verificar pelos dados acima a convergência foi rápida, calculada em 0,053951 segundo e obteve-se pequenas oscilações em $q4$, da ordem de $\mathbf{10^{-3}}$. A convergência de Pz já foi muito suave sem oscilar até chegar na posição -0,5. Os resultados para este caso, apresentados no campo de Posições, coincidem para o caso da inversa do caso anterior.

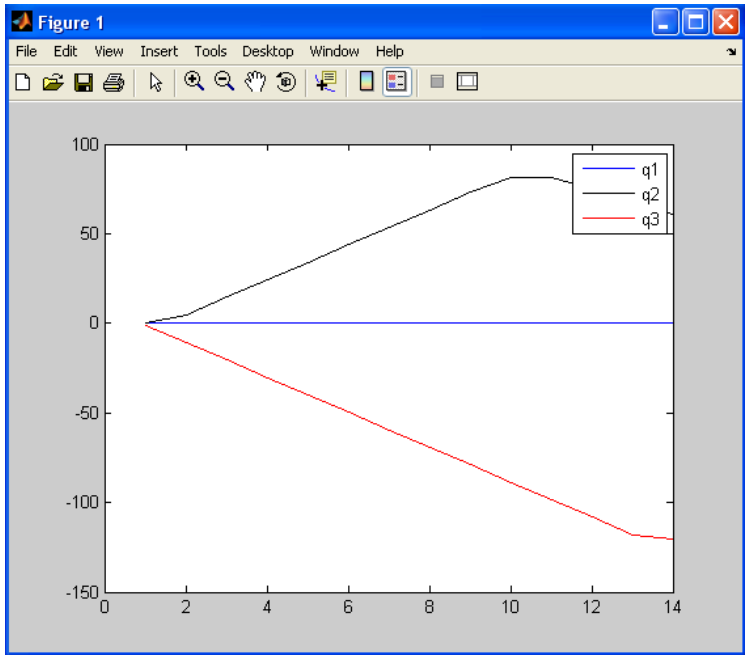


Figura 4.3: Gráfico do cálculo dos ângulos das juntas - método *Damped Least Squares* fator 0,1

4.3.3 Simulação da Cinemática inversa - Caso 3

Os dados a seguir são para posicionar o robô em uma das formas pré-definidas da robótica comportamental, indicando um estado do comportamento do robô que não cabe a nós aqui discuti-lo, posição esta mostrada na figura 4.11.

Aqui nos tem-se uma situação mais complexa que a anterior, uma vez que o posicionamento do centro varia em q4, Py e Pz, mas mesmo assim ainda é uma convergência muito boa e em tempo aceitável, 0,504641 segundo. Os resultados da configuração de ângulos da junta e das posições da perna e centro do robô aparecem nos campos mostrados na figura 4.11.

4.3.4 Simulação da Cinemática inversa - Caso 4

A figura 4.12 é referente à simulação do robô executando uma circunferência em torno da origem das coordenadas do $mundo$.

Como foi possível verificar, através da simulação, cada nova posição definida para o robô tem-se que os resultados da cinemática direta e inversa da posição vão aparecendo na tela. Verificou-se que as posições das pernas mudam apenas na terceira casa decimal, enquanto os ângulos das juntas e a posição do centro variam a toda alteração do centro do robô. Essa pequena alteração no posicionamento das patas pode ser evitado alterando-se a tolerância da convergência, que para este caso estava em 0,001 mostrando que o resultado apresentado está dentro do tolerável.

4.3.5 Simulação da Cinemática inversa - Caso 5

Para esta simulação temos o robô dando um passo à frente. Aqui foi feito um algoritmo simples em que utilizou-se valores empíricos para posicionar as pernas do robô de forma seqüêncial de forma a se obter um passo completo. O resultado se mostrou satisfatório em MatLab. A figura 4.13 mostra um instante deste movimento.

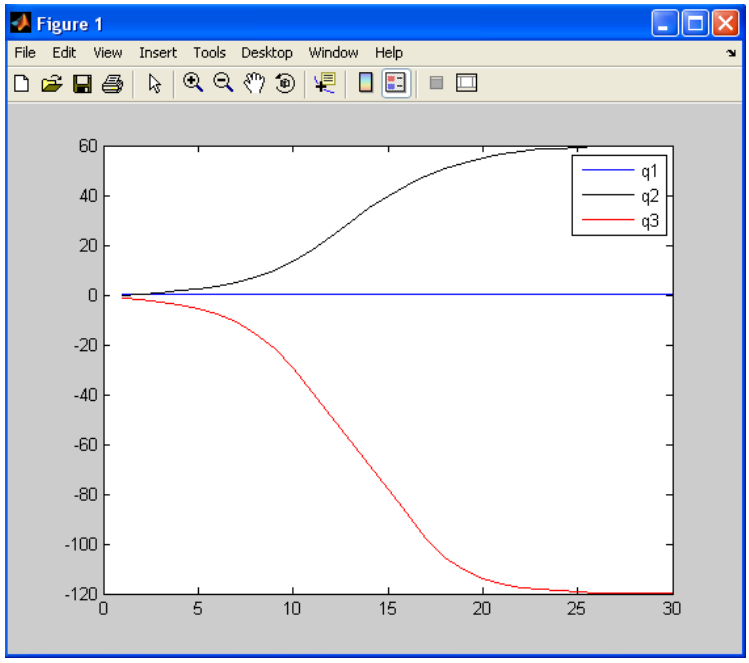


Figura 4.4: Gráfico do cálculo dos ângulos das juntas - método *Damped Least Squares* fator 0,5

4.3.6 Simulação da Cinemática Direta de Posição

Agora é exibido o resultado da simulação refere à cinemática direta de posição. Como pode-se verificar pela figura 4.14 a simulação funcionou muito bem quando utilizada a cinemática direta de posição, o que permitiu avaliar o modelo geométrico utilizado. Neste exemplo, colocou-se uma configuração de ângulos para as pernas tal como a obtida quando transladamos o centro do robô da posição z = 0 para a posição z = -0,5. Contudo, como é uma cinemática direta de posição, posicionaremos as patas com o centro do robô fixo, assim o plano de apoio das patas passa a ser z = -0.5 e não z = -1 como no caso em que o centro do robô é alterado. Isto pode ser verificado pelos resultados dos valores apresentados no campo Posições da figura 4.14.

No caso aqui apresentado as configurações para os ângulos das juntas são definidas pelo usuário e o posicionamento da extremidade da perna (*end-effort*) é dado apenas em função dos parâmetros dos ângulos da perna em questão, uma vez que não se determina o plano de apoio das patas e o centro é mantido fixo. No caso real o centro do robô seria deslocado para a posição z = -5, uma vez que o plano de apoio das pernas é fixo, o chão ou a superfície de uma mesa, por exemplo.

4.4 TESTES COM O ROBÔ

Devido ao não recebimento dos motores das pernas os únicos testes efetuados com o robô foram para o posicionamento dos servos da cabeça e do rabo.

Nestes testes foi possível validar toda a implementação do hardware e da programação nos micro-controladores, além de se testar o programa, ainda em desenvolvimento, que será utilizado para realizar todo o controle do robô.

A estrutura mecânica pôde ser previamente testada numa fase anterior do projeto na qual utilizávamos servos com torque inferior aos atuais e que não conseguiram suportar a carga do robô. A mesma não pode ser mais apresentada uma vez que o acoplamento dos servos antigos nas juntas do robô é bastante diferente.

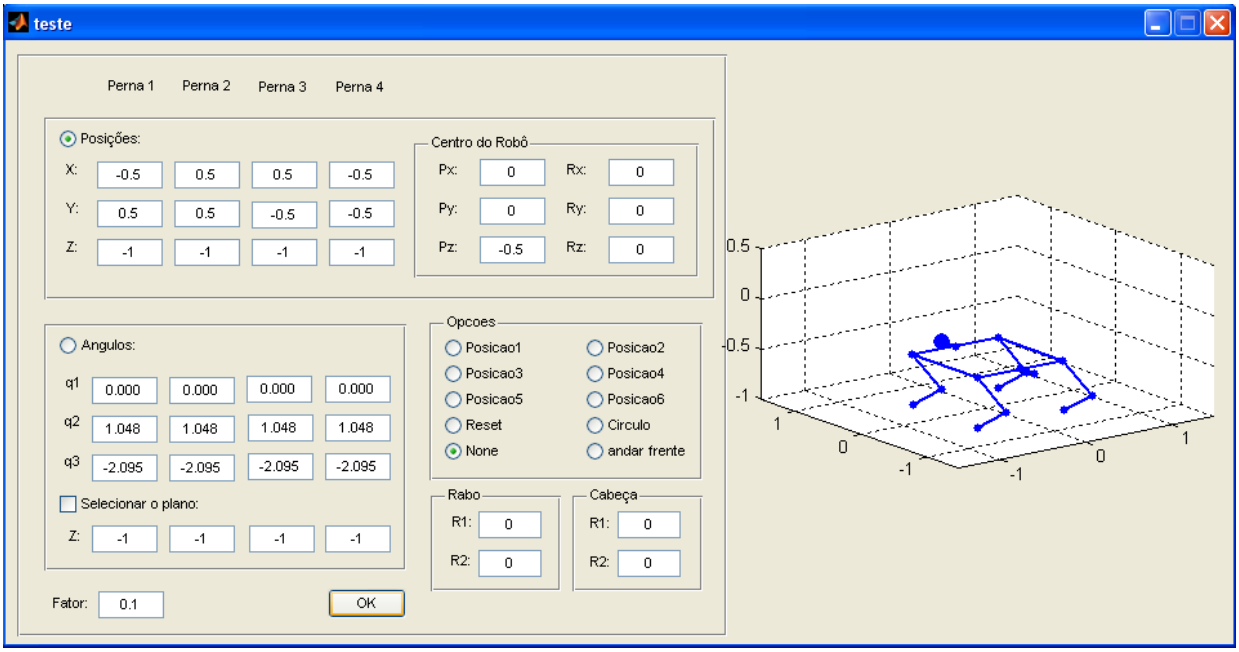


Figura 4.5: Interface do programa mostrando o resultado da cinemática inversa de posição

Robô-tamanduá - Primeiros testes

A estrutura mecânica, assim como os hardwares serviram de base para o desenvolvimento do projeto de um robô-tamanduá (Figura 4.15). Devido a impossibilidade de se obter toda a mecânica funcional, controle das pernas, cabeça e rabo, estando apenas os dois últimos já implementados, o projeto foi apresentado apenas com o controle do posicionamento da cabeça (2 graus de liberdade, dado por 2 servos) e com o controle do posicionamento do rabo.

Este robô-tamanduá, como foi base do estudo da robótica comportamental, tem algumas possíveis configurações para o seu estado emocional, e de acordo com o estado emocional o servo do rabo deve realizar movimentos períodicos. Para tal, desenvolveu-se um *driver* que realizasse tal interfaceamento, entre o sinal para o acionamento dos leds e a lógica comportamental, assim como para o acionamento do servos do rabo e da cabeça.

O posicionamento de todos os servos, bem como o acionamento dos LEDs se mostraram em perfeito funcionamento. Assim pode-se esperar que para realizar-se o posicionamento dos servos das pernas não existirão maiores dificuldades.

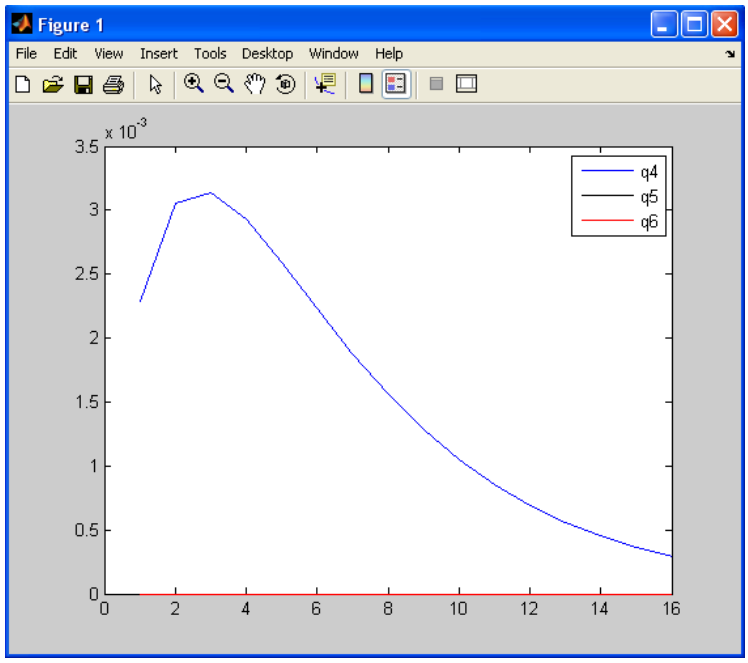


Figura 4.6: Simulação da cinemática inversa de posição para o centro do robô usando o método do *Damped Least Squares* com fator de 0,5 para o caso em que utilizou-se o resultado da figura 4.5 (o eixo vertical é de ângulos em graus e o horizontal é das iterações)

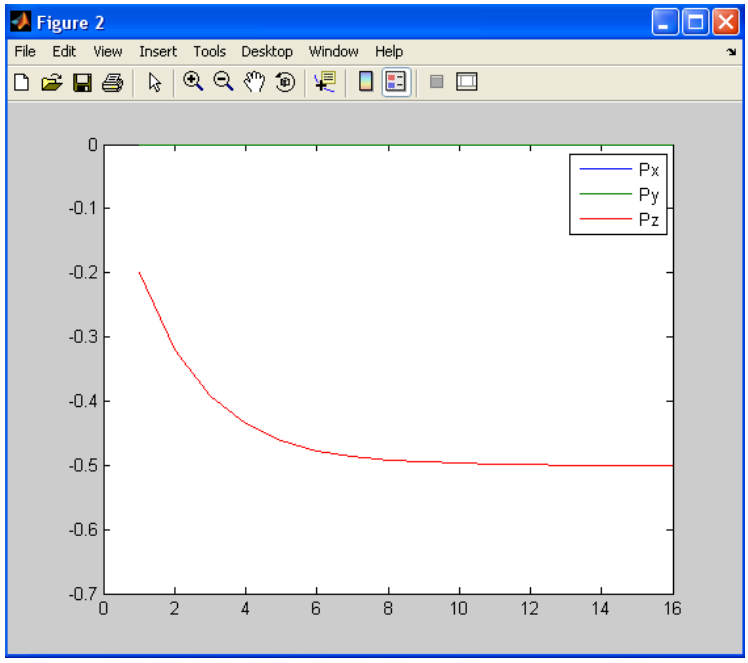


Figura 4.7: Simulação da cinemática inversa de posição para o centro do robô usando o método do *Damped Least Squares* com fator de 0,5 para o caso em que utilizou-se o resultado da figura 4.5 (o eixo vertical é de ângulos em graus e o horizontal é das iterações)

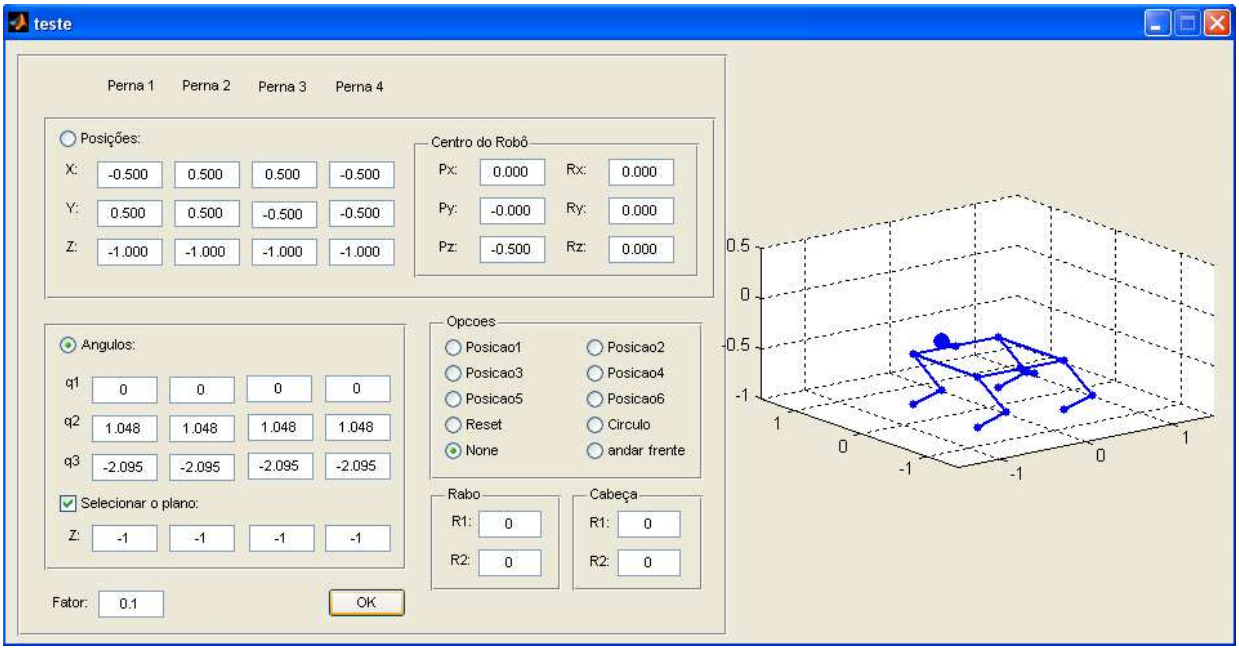


Figura 4.8: Interface do programa mostrando o resultado da cinemática inversa de posição do centro do robô quando usamos os resultados da figura 4.5

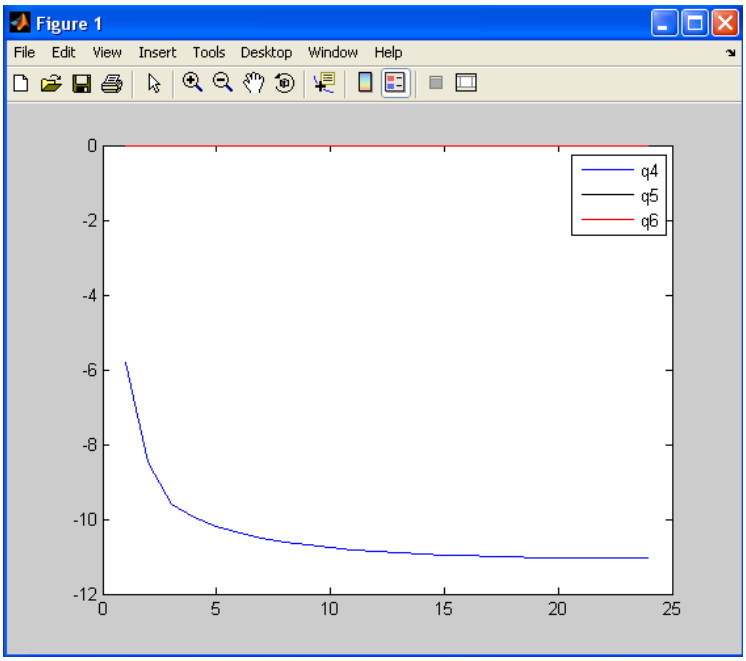


Figura 4.9: Simulação da cinemática inversa de posição para o centro do robô usando o método do *Damped Least Squares* com fator de 0,5 para o caso em que utilizaou-se a posição 3 de comportamento do robô (o eixo vertical é de ângulos em graus e o horizontal é das iterações

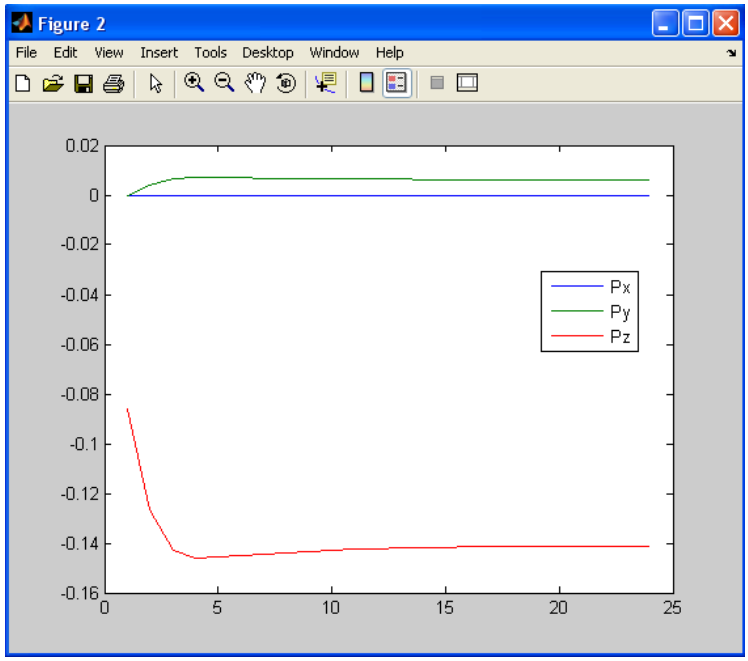


Figura 4.10: Simulação da cinemática inversa de posição para o centro do robô usando o método do *Damped Least Squares* com fator de 0,5 para o caso em que utilizou-se a posição 3 de comportamento do robô (o eixo vertical está em unidades de comprimento e o horizontal é das iterações

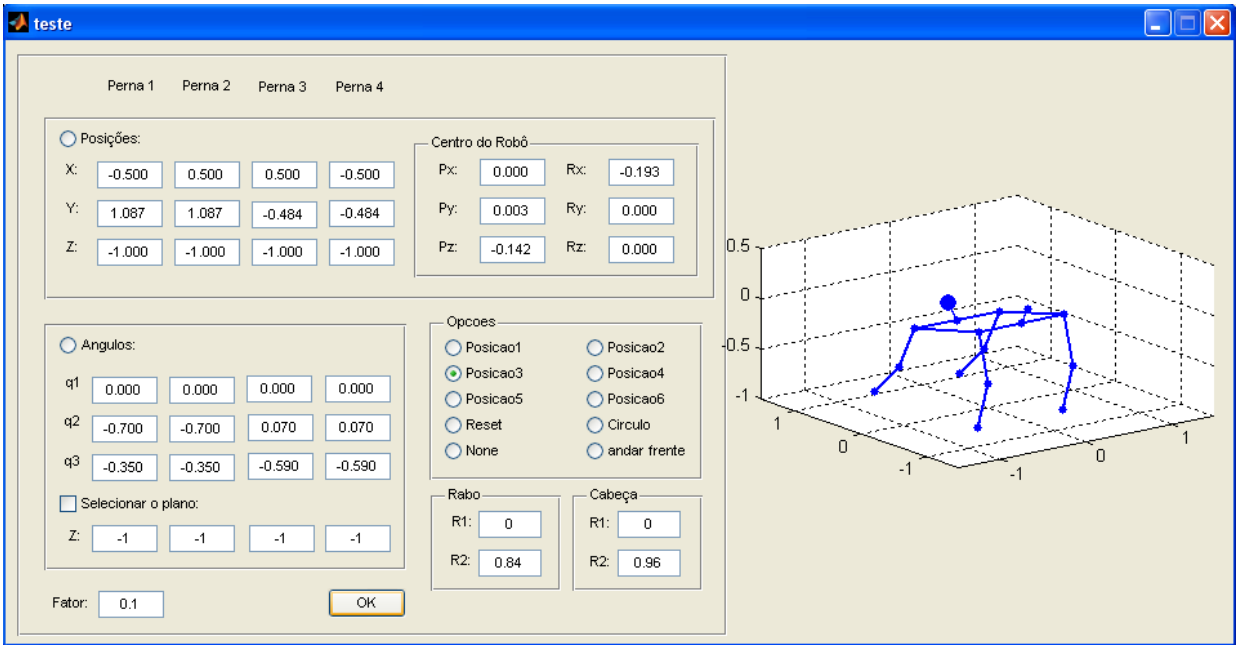


Figura 4.11: Interface do programa mostrando o resultado da cinemática inversa de posição do centro do robô

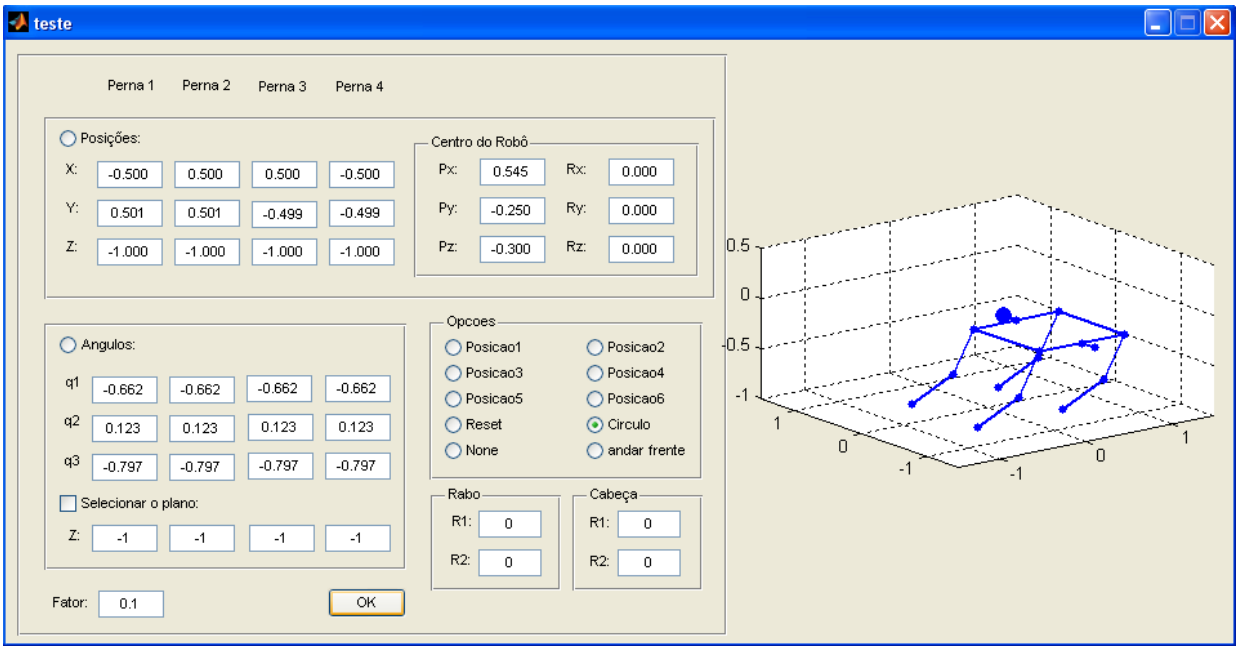


Figura 4.12: Interface do programa mostrando o resultado da cinemática inversa de posição quando o centro do robô realiza um círculo em torno da origem

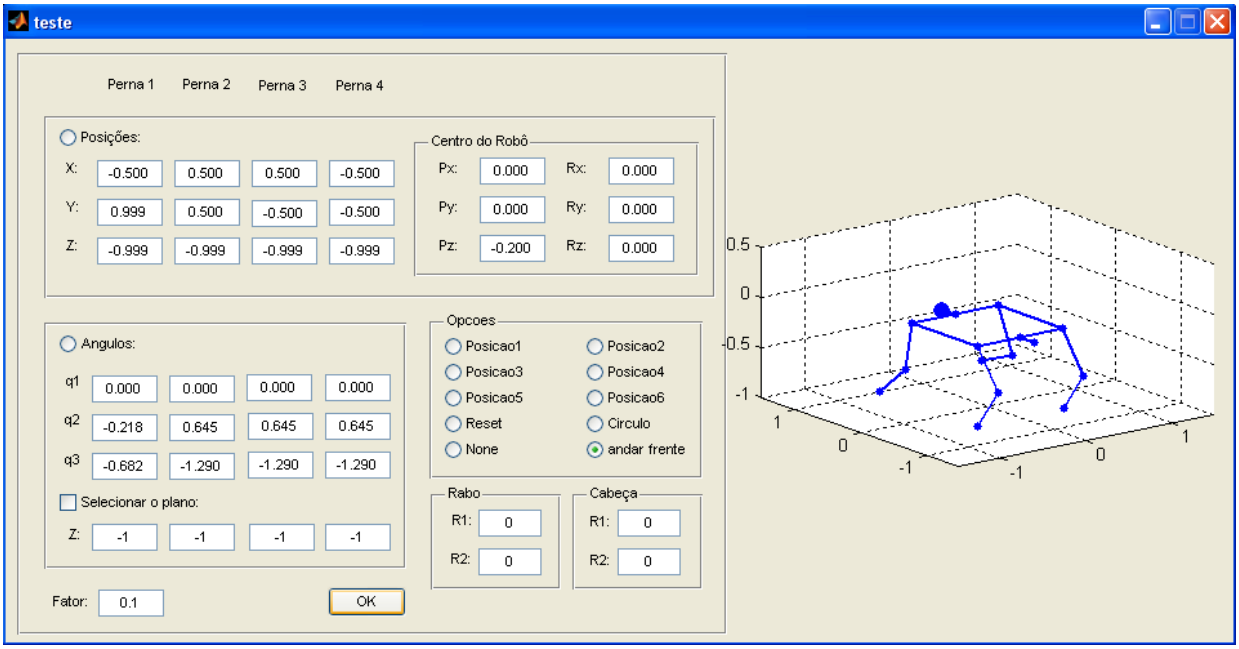


Figura 4.13: Interface do programa mostrando o resultado da cinemática inversa de posição quando o robô realiza um passo à frente

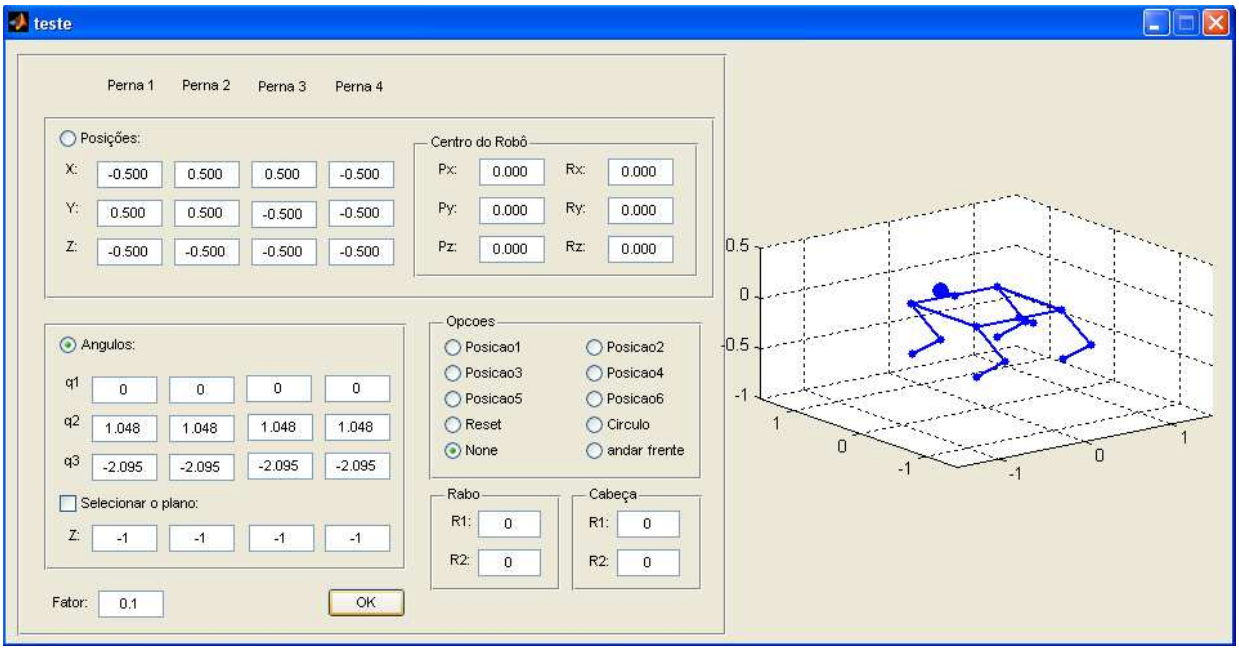


Figura 4.14: Interface do programa mostrando o resultado da cinemática direta de posição

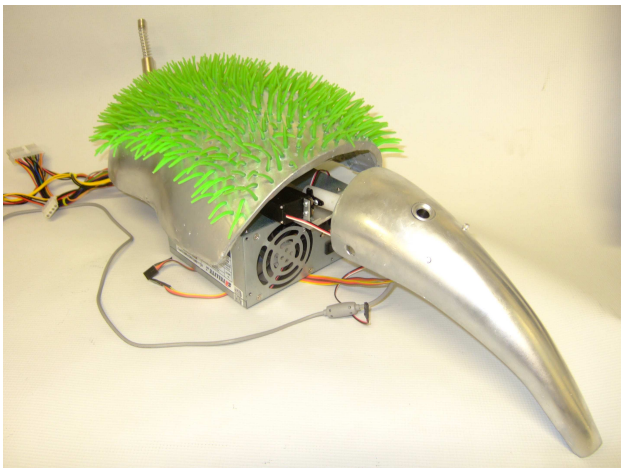


Figura 4.15: Foto do estágio atual do Robô-tamanduá

5 CONCLUSÕES

Neste trabalho apresentou-se o estado atual de desenvolvimento de um quadrúpede que deve ser utilizado em robótica móvel e também para estudo da robótica comportamental. Discutiu-se aspectos relativos a concepção mecãnica, onde a maior dificuldade encontrada foi a adequação de peças existentes no mercado e acessíveis econômicamente para o projeto, porém uma vez adaptada e ajustada tais peças, após a montagem a estrutura mecânica mostrou-se adequada ao ideal proposto devido a sua alta resistência associada ao baixo peso da armação. Outro fato bastante importante foi o dimensionamento dos servos motores, em uma etapa inicial do projeto utilizava-se servos, devido a sua disponibilidade, cujo o torque não foi suficiente para, nem sequer, manter o robô sobre suas próprias pernas. A alimentação dos servos também não estava adequada, então partiu-se para a concepção de drivers de potência e substituição dos servos por servos de alto torque. Apesar do não recebimento dos servos verificou-se a robustez dos drivers de potência submetendo os servos a cargas elevadas enquanto observava-se a corrente fornecida, o que não foi o limitador do uso do servo.

Uma vez que não foi concluída a parte física do projeto a simulação tornou-se uma grande aliada para se verificar e se validar os modelos geométricos utilizados, bem como as cinemáticas direta e inversa de posição. O matlab mostrou-se uma ferramenta extramamente útil para realizar tal simulação, onde pudemos visualizar a execução de movimentos simples de posicionamento do robô, além de ser capaz de realizar os cálculos envolvidos no processo.

A modularidade e flexibilidade são pontos fortes desse projeto, uma vez que de posse do protocolo de comunicação em um barramento de comunicação RS-485, as placas das pernas são capazes de controlar o posicionamento de qualquer servo RC que seja alimentado de 4 a 6V e tenha como sinal de acionamento PWM de períodos próximos a 50Hz e necessitem de pulsos entre 1,0 e 2,0 ms para varrer os extremos das posições.

Para a continuação desse projeto tem-se algumas proposições, sendo que algumas delas já estão sendo desenvolvidas, como a implementação do algoritmo da cinemática inversa e direta de posição, além da cinemática inversa e direta de velocidade (quando concluída) em C/C++ utilizando o linux, o que daria uma maior flexibilidade ao robô, além de interligar os aspectos da robótica comportamental com os da robótica móvel para o seu posicionamento e deslocamento. Um próximo passo a ser desenvolvido, pensando em robótica terrestre, é a obtenção da dinâmica do quadrúpede e então a geração de passos para o deslocamento do quadrúpede de forma não empírica, como a aqui realizada.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] MARTINS, A. S.; BORGES, G. A. *Introducao ao padrao fisico RS485 para comunicacao serial*. [S.l.], June 2006.

[2] MACHADO, J. T.; SILVA, M. F. An overview of legged robots.

[3] SHILING, R. J. *Fundamentals of Robotics: Analysis and Control*. [S.l.]: Prentice Hall, 1990.

[4] CRAIG, J. J. *Introduction to Robotics*. [S.l.]: Addison-Wesley, 1990.

[5] BRUYNINCKX, H.; SCHUTTER, J. D. *Introduction to Intelligent Robotics*. [S.l.], 2001.

[6] BUSS, S. R. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. 2004.

[7] XIAO, J. Kinematics of robot manipulator. 2005.

[8] ATMEL. *8-bit AVR with 8K Bytes In-System Programmable Flash*. [S.l.], October 2006.

[9] MICROELECTRONICS, S.-T. *THREE TERMINAL 3A ADJUSTABLE VOLTAGE REGULATORS*. [S.l.], March 1993.

[10] BORGES, G. A. et al. *Desenvolvimento com microcontroladores Atmel AVR*. [S.l.], June 2006.

[11] MATSUMOTO, l. Y. *MatLab7: Fundamentos*. [S.l.]: São Paulo, 2004.

ANEXOS