



Universidade de Aveiro

Sistema de Visão em Tempo-Real

Sistemas de Tempo-Real 07/08

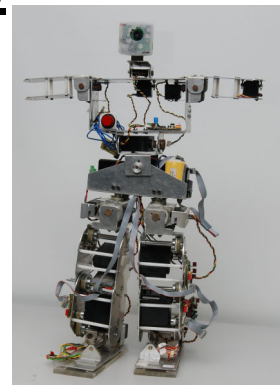
Docentes:

Luís Almeida

Paulo Pedreiras

Realizado por:

28256 – Mauro Rodrigues





Introdução

Este trabalho surge no âmbito do desenvolvimento de um robô humanóide na Universidade de Aveiro, um projecto de colaboração entre o Departamento de Mecânica e o Instituto de Engenharia Electrónica e Telemática de Aveiro, e da disciplina de Sistemas de Tempo-Real. O humanóide tem como meta, a participação na *Robocup*, categoria *Penalty Kick*. É objectivo deste documento relatar a implementação do Sistema de Visão em Tempo-Real do humanóide.

Para a implementação do sistema é necessária a utilização das funcionalidades de tempo-real uma vez que as tarefas associadas a este sistema têm de ser executadas dentro de um intervalo de tempo finito, não interessa processar imagens capturadas há muito tempo pois o mundo real já se alterou entretanto.

Objectivos

Criação de um sistema de Tempo-Real para um Robô Humanóide:

1. Enquadramento do problema.
2. Definição e modularização do processo.
3. Escalonamento Tempo-Real.
4. Discussão de eventuais conflitos com outras tarefas do humanóide.

Enquadramento

A plataforma é um Robô Humanóide desenvolvido na Universidade de Aveiro. Este possui como características principais: 22 graus de liberdade, 60 cm de altura, uma massa total de 6 Kg, arquitectura distribuída de controlo constituída por três tipo de unidades – *CPU*, *Master* e *Slaves* –, formando uma rede de controladores interligada por um barramento CAN em configuração *Master/Multi-Slave*. É ainda importante referir que todo o é do tipo *embedded*, sendo o *CPU* um processador *standard* PC/104+ com a seguinte configuração – *AMD Geode LX800@500MHz*, *512Mb RAM*, *SSD 1Gb*.

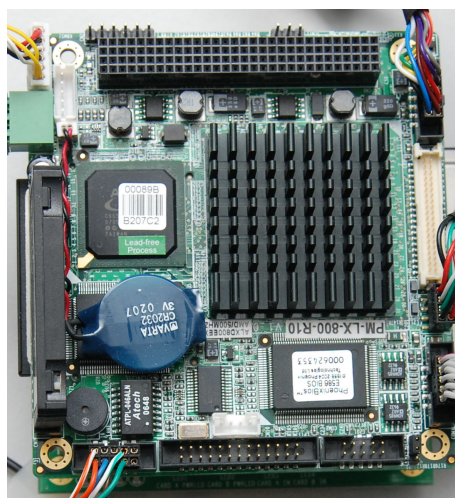


Figura 1. PC/104+



A captura do sinal de vídeo é feita através de uma câmara *UniBrain Fire-i @ 30fps (640x480)* ligada a uma placa *PCMCIA FireWire*.

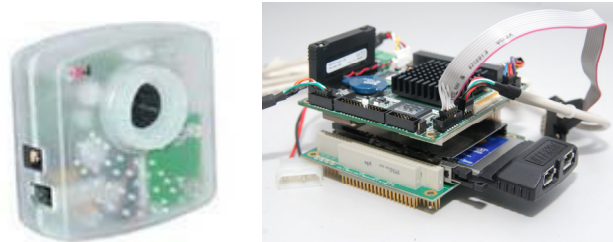


Figura 2. Câmara Unibrain Firewire e Módulo PCMCIA

O sistema operativo é o GNU/Linux *Debian 40r0 i386 Net Install*. Para efeitos de programação está a ser usada a linguagem C/C++ e as bibliotecas OpenCV (processamento imagem/vídeo) e PMan (funcionalidades tempo-real).

Definição e modularização do processo

A arquitectura definida para o sistema é a que está representada na figura abaixo (Fig. 3). Esta consiste em quatro tarefas, *Image Acquisition*, *Image Processing*, *Object Tracking* e *Control*. A primeira delas é a tarefa que faz o *trigger* do sistema, ou seja, após a aquisição da imagem, as tarefas seguintes são despertadas e executam a sua função.

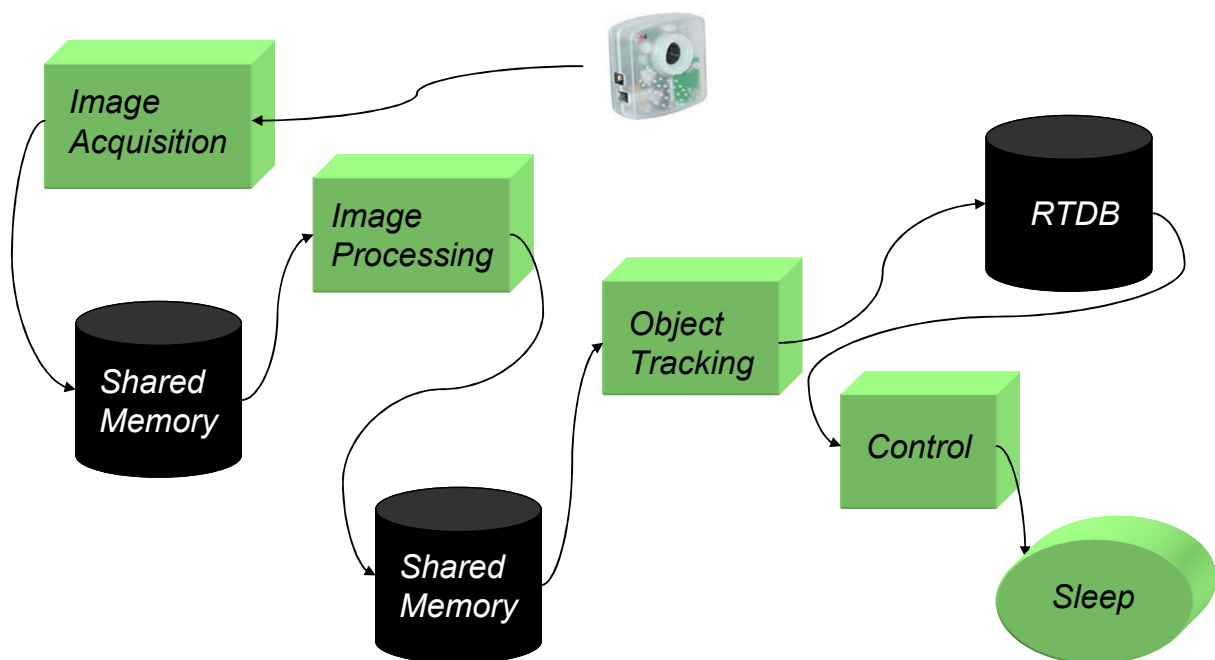


Figura 3. Arquitectura do Sistema de Visão

Image Acquisition

A tarefa *Image Acquisition* é, como já foi referido, o *tick* do sistema, activa os processos seguintes quando uma nova imagem está disponível. A imagem adquirida é armazenada numa *shared memory* (SHM_ATOP) para posterior utilização por parte de outros processos.

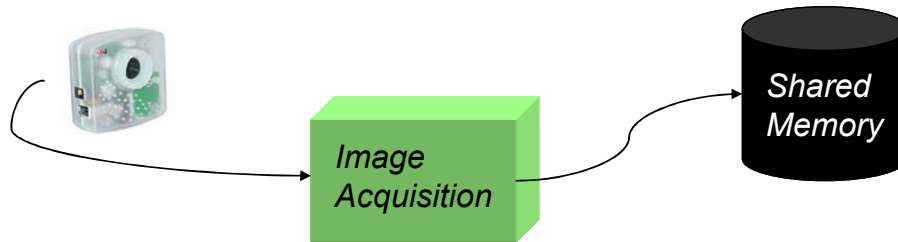


Figura 4. Tarefa de aquisição de imagem

Image Processing

Image Processing recolhe da *shared memory* referida anteriormente (SHM_ATOP) a imagem armazenada e efectua o respectivo processamento de cor, a imagem em RGB é convertida para HSV e posteriormente dividida nas suas componentes H, S e V. Os parâmetros da máscara, previamente ajustados com a tarefa auxiliar *Color Calibration*, são então aplicados a cada componente e estas são de seguida combinadas formando a mesma. Esta máscara é então armazenada numa outra *shared memory* (SHM_PTOT).

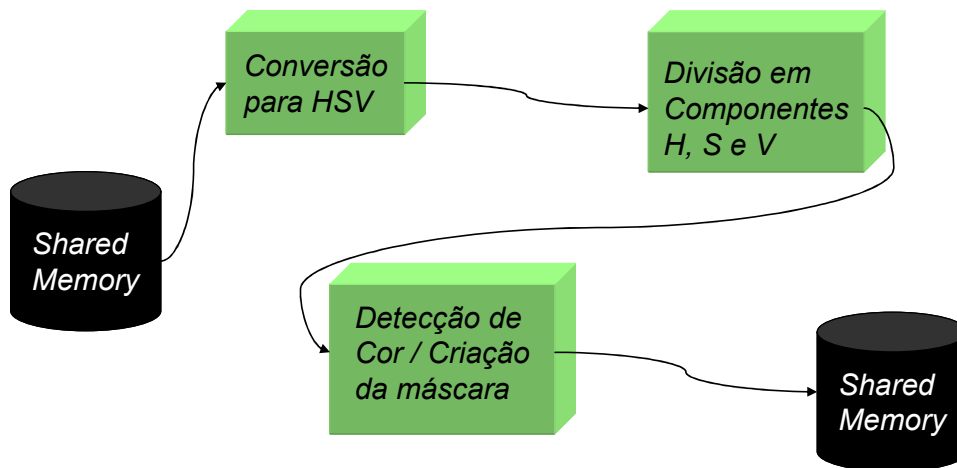


Figura 5. Tarefa de processamento de imagem

Object Tracking

Object Tracking faz a detecção da bola na imagem e respectiva validação através da detecção de círculos na imagem filtrada. É calculado o centro de massa do objecto detectado e determinada a sua localização relativa na imagem, estes parâmetros são então armazenados na base de dados tempo-real (SHM_RTDB).

A sequência dos eventos do processamento de imagem está representada na Figura 7.

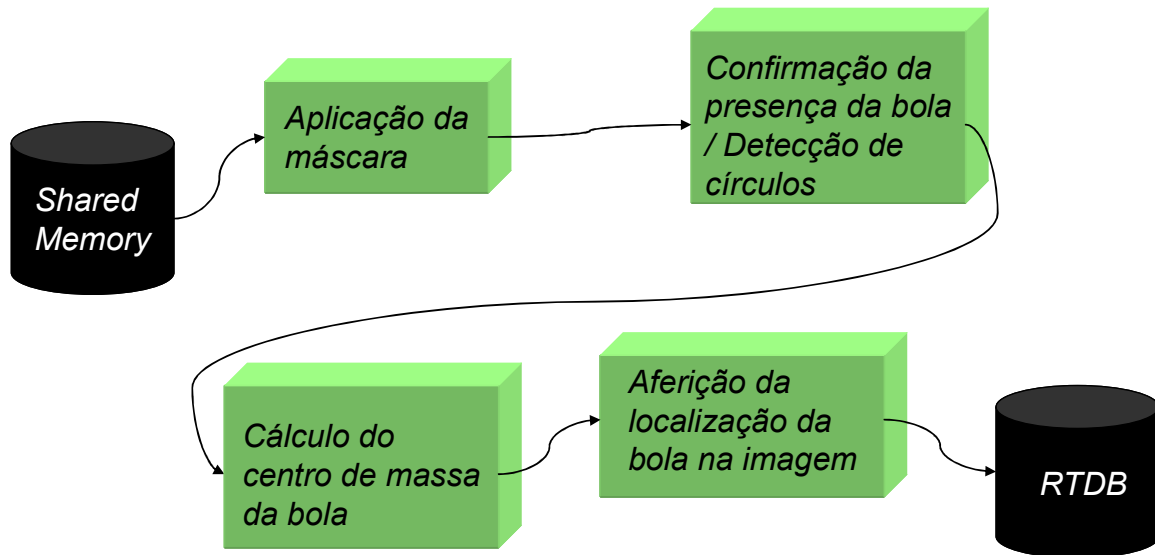


Figura 6. Tarefa de localização da bola

A figura seguinte ilustra os passos desde a aquisição da imagem até à determinação dos parâmetros do objecto, passando criação da máscara através das componentes H, S e V da imagem.

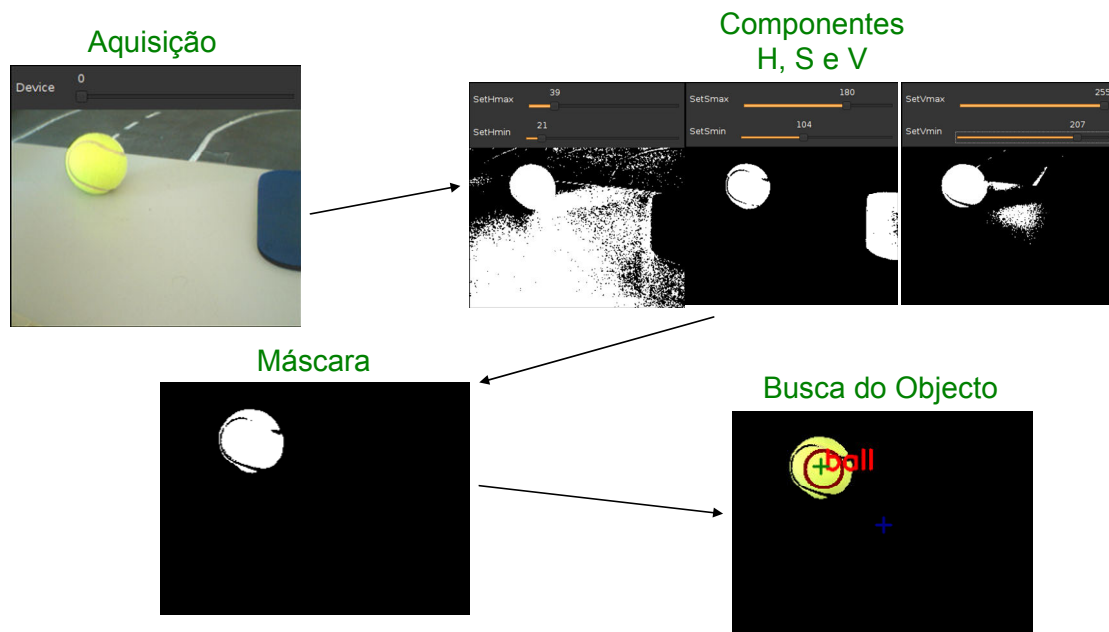


Figura 7. Sequência de processamento de imagem

Control

A tarefa *Control* recolhe da RTDB, os parâmetros calculados pela tarefa *Object Tracking* e faz uma validação dos valores. De seguida, são calculados os parâmetros do movimento a efectuar, por ex., uma deslocação da bola no plano da imagem, corresponde a uma combinação de movimentos das juntas do pescoço necessários para que o humanóide possa seguir a bola com a cabeça (câmara), esta transformação tem o nome Cinemática Inversa.

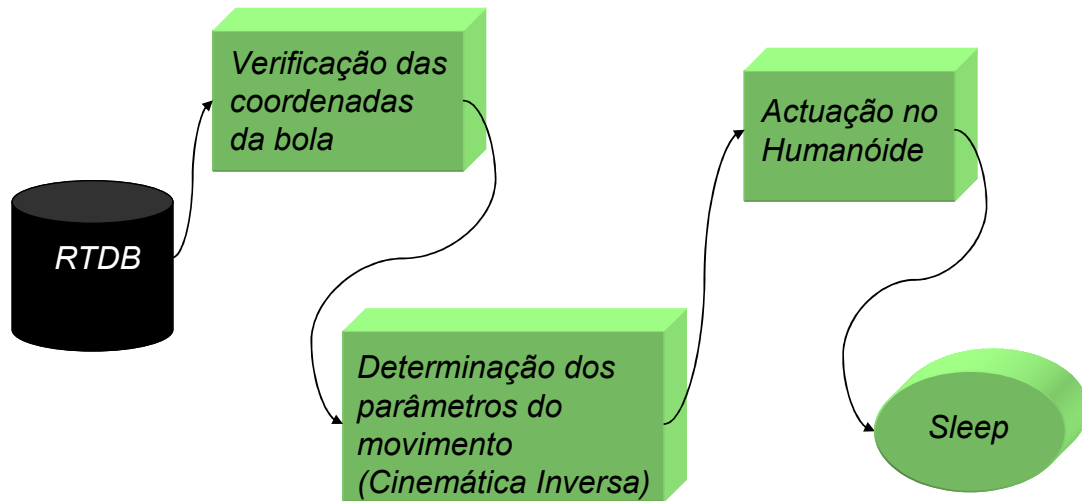


Figura 8. Tarefa de actuação e controlo do Humanóide

Color Calibration

Color Calibration é uma tarefa auxiliar independente que permite calibrar os parâmetros que serão posteriormente utilizados no processamento de imagem. Este processo é efectuado com recurso a janelas com barra de deslocamento que fazem o ajuste “on-line” de cada componente (Fig. 9). No final da execução os parâmetros são salvos para um ficheiro de texto chamado *hsv_param.txt*.

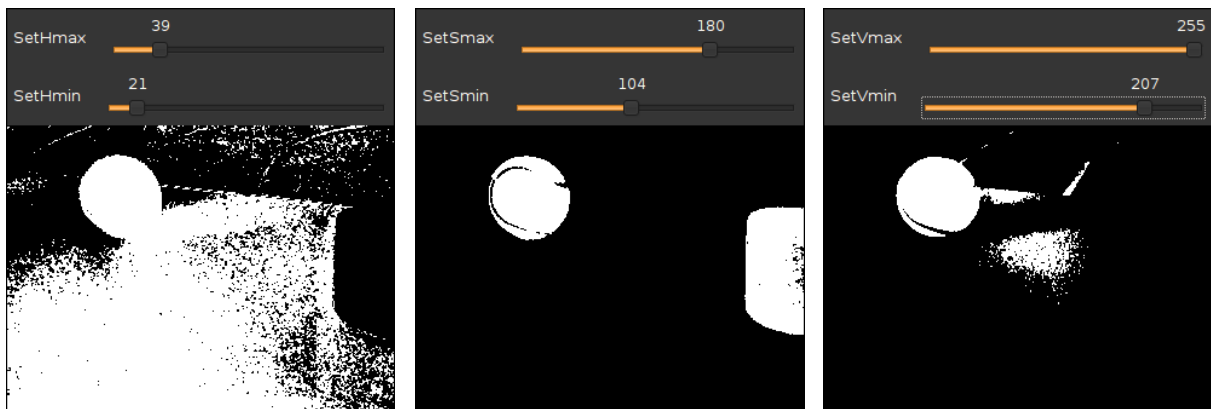


Figura 9. Janelas de calibração

Escalonamento

Precedências

As precedências foram definidas como está ilustrado na figura abaixo. A sequência das tarefas é lógica, captura da imagem, processamento da mesma, extracção dos parâmetros do objecto alvo e finalmente actuação no humanóide concordante com a informação recolhida e tarefa a executar, no caso específico, seguir uma bola.

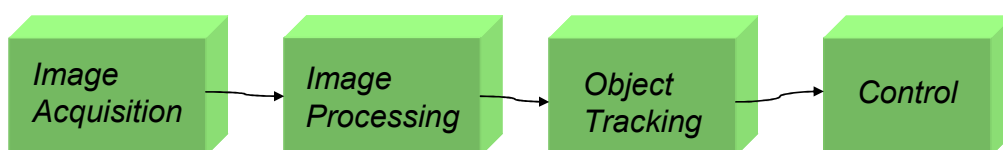


Figura 10. Esquema de precedências



de um refinamento do algoritmo para uma melhor validação da bola, uma vez que esta ainda não é muito precisa.

As restantes funções aparentemente possuem um comportamento normal, o que ainda não foi possível verificar devido ao obstáculo já referido. Na tarefa de controlo, o respectivo algoritmo ainda não se encontra implementado em *software*, estando no entanto determinado teoricamente (controlo das juntas do pescoço! - Anexo I). Este poderá vir a introduzir algum atraso no processamento global pois recorre a funções trigonométricas, mais complexas de tratar.

É de referir que os testes foram efectuados no computador portátil do autor pois devido à limitada capacidade de processamento da plataforma final, foi decisão do mesmo efectuar os testes preliminares naquele. Por exemplo, fazer o *debug* do processamento de imagem com recurso a janelas de visualização é uma tarefa que se torna pesada, sendo preferível fazê-lo externamente e posteriormente correr lá uma versão mais madura sem interface gráfica, o que reduz consideravelmente a capacidade de processamento exigida.

Relativamente a tarefas *não-visão* do Humanóide, este aspecto ainda não foi considerado mas será tido em conta futuramente incluindo rotinas de verificação de estado para posterior controlo, por ex., em casos de desequilíbrio.

Para finalizar é importante recordar quais os passos a efectuar no futuro próximo:

- *Debug* da tarefa de processamento de imagem;
- Verificação do funcionamento das restantes tarefas;
- Passagem para *software* do algoritmo de controlo das juntas do pescoço;
- Execução de testes para determinar o impacto temporal de todas as tarefas na execução;
- Inclusão das rotinas de verificação de estado do Humanóide e respectivo controlo;

Nota: devido à extensão do código criado, este não será introduzido em anexo, seguindo apenas no ficheiro zip que contém todos os elementos do trabalho.



Bibliografia

- Pedreiras, Paulo e Almeida, Luís; “*Task Management for Soft Real-Time Applications Based on General Purpose Operating Systems*”; 2007
- Pedreiras, Paulo et. al.; “*Enhancing the reactivity of the vision subsystem in autonomous mobile robots using real-time techniques*”; 2005
- Silva, Filipe M., “*Capítulo 4 – Fundamentos de Cinemática (Robótica)*”, 2002-2003



Anexo I – Cinemática Inversa do Humanóide

A cinemática inversa obtém-se invertendo a matriz Jacobiana da cinemática directa. A cinemática directa relaciona os movimentos nas juntas com a deslocação no plano de imagem.

$$\begin{bmatrix} \dot{x}_3 \\ \dot{y}_3 \end{bmatrix} = J \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

Equação 1. Equação da Cinemática Directa

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = k.J^{-1} \begin{bmatrix} \dot{x}_3 \\ \dot{y}_3 \end{bmatrix}$$

Equação 2. Cinemática Inversa

em que J é a matriz Jacobiana

$$J = \begin{bmatrix} \frac{\partial x_3}{\partial \theta_1} & \frac{\partial x_3}{\partial \theta_2} \\ \frac{\partial y_3}{\partial \theta_1} & \frac{\partial y_3}{\partial \theta_2} \end{bmatrix}$$

Equação 3. Jacobiano

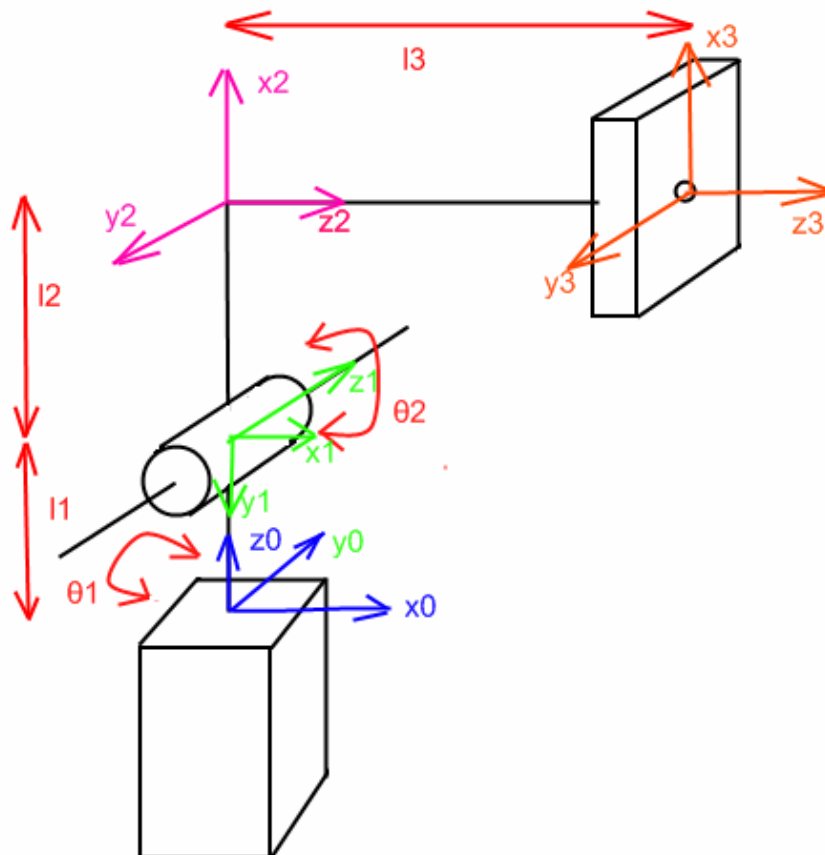


Ilustração 1. Esquema do pescoço do Humanóide



Cinemática Directa – Algoritmo de Denavit-Hartenberg

Transformações entre sistemas de eixos:

i	θ_i Rot(Zi)	l_i T(xi+1)	d_i T(Zi)	α_i R(xi)
1	θ_1	0	l_1	-90°
2	$\theta_2 - 90^\circ$	l_2	0	-90°
3	0	0	l_3	0

Nota: Nos passos seguintes Ci refere-se a $\cos(\theta_i)$ e Si a $\sin(\theta_i)$.

$$A_i = Rot_z(\theta_i) \cdot T(l_i, 0, 0) \cdot T(0, 0, d_i) \cdot Rot_x(\alpha_i)$$

$$A_1 = \begin{bmatrix} C1 & 0 & -S1 & 0 \\ S1 & 0 & C1 & 0 \\ 0 & -1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, A_2 = \begin{bmatrix} S2 & 0 & C2 & l_2 \times S2 \\ -C2 & 0 & S2 & -l_2 \times C2 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2 \cdot A_3 = \begin{bmatrix} S2 & 0 & C2 & C2 \times l_3 + l_2 \times S2 \\ -C2 & 0 & S2 & S2 \times l_3 - l_2 \times C2 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_1 \cdot A_2 \cdot A_3 = \begin{bmatrix} \dots & \dots & \dots & C1(C2 \times l_3 + l_2 \times S2) \\ \dots & \dots & \dots & S1(S2 \times l_3 - l_2 \times C2) \\ \dots & \dots & \dots & -(S2 \times l_3 - l_2 \times C2) + l_1 \\ \dots & \dots & \dots & \dots \end{bmatrix}$$

$$\begin{cases} x_3 = C1(C2 \times l_3 + l_2 \times S2) \\ y_3 = S1(S2 \times l_3 - l_2 \times C2) \\ z_3 = -(S2 \times l_3 - l_2 \times C2) + l_1 \end{cases}$$

$$J = \begin{bmatrix} \frac{\partial x_3}{\partial \theta_1} & \frac{\partial x_3}{\partial \theta_2} \\ \frac{\partial y_3}{\partial \theta_1} & \frac{\partial y_3}{\partial \theta_2} \end{bmatrix} = \begin{bmatrix} -S1(l_3 \times C2 + l_2 \times S2) & C1(l_3 \times S2 + l_2 \times C2) \\ C1(l_3 \times C2 + l_2 \times S2) & S1(l_3 \times S2 + l_2 \times C2) \end{bmatrix}$$

$$J^{-1} = \begin{bmatrix} \frac{-S1}{l_2 \times S2 + l_3 \times C2} & \frac{C1}{l_2 \times S2 + l_3 \times C2} \\ \frac{C1}{l_2 \times C2 - l_3 \times S2} & \frac{S1}{l_2 \times C2 - l_3 \times S2} \end{bmatrix}$$