Técnicas de segmentação de cor para seguimento de alvos – aplicação à aprendizagem de tarefas em Robots humanóides.

(LEIC)

Ano Lectivo 2004/2005

Departamento de Engenharia Informática

Título: Técnicas de segmentação de cor para seguimento de alvos – aplicação à aprendizagem de tarefas em Robots humanóides.

Professor Orientador:

Alexandre Bernardino _____

Professor Acompanhante:

José Santos-Victor _____

Alunos:

Júlio Jesus León Pérez _____

Agradecimentos.

Ao professor Alexandre pela sua constante orientação do trabalho e os objectivos. Pela sua inestimável ajuda na compreensão dos conceitos teóricos para a resolução do trabalho e a grande quantidade de tempo dedicado a colaboração na construção do trabalho.

Ao professor José por o seu espírito crítico e pela sua insistência na construção dum trabalho de qualidade.

A Manuel, Pedro, Plínio e Rodrigo pela sua companhia no Laboratório do Vislab e pela quantidade de sugestões e dúvidas que foram capazes de resolver.

A todo o equipo do Vislab por permitir-me trabalhar com eles.

A todos. Muito Obrigado.

Obrigado também a Baltazar.

Resumo.

No presente trabalho é abordado o problema da segmentação da cor para o tratamento de cenas completas, e para o seguimento de alvos, orientado a integrar a segmentação num sistema robótico humanóide.

Implementa-se um sistema que é capaz de treinar objectos e depois fazer a pesquisa de esses objectos em cenas novas.

Índice

1 Introdução

Os Robots humanóides têm vindo a ser cada vez mais utilizados na Robótica de serviços onde existe uma forte componente de interacção com humanos, por exemplo em hospitais, lares, museus. A forma humanóide torna os robots mais atractivos e aceitáveis em ambientes com utilizadores não especializados. No entanto, a sua utilização ainda é limitada devido à forma dinâmica e não estruturada destes ambientes, o que dificulta a percepção e programação das tarefas. Uma das formas que tem vindo ser explorada para abordar este problema consiste na aprendizagem por imitação, onde através da demonstração de execução de tarefas, o robot irá adquirir as capacidades necessárias num ambiente específico. Uma das capacidades perceptivas fundamentais consiste na modelação, detecção e seguimento dos objectos envolvidos nas tarefas robóticas.

O Laboratório de Visão do ISR tem vindo a construir uma plataforma robótica humanóide composta por uma cabeça robótica com visão binocular e um torso com um braço e uma mão. O sistema permite efectuar tarefas de manipulação em objectos simples coloridos. As técnicas de segmentação de cor são, por isso, importantes na funcionalidade global do sistema, uma vez que permitem detectar e seguir os objectos de interesse.

Neste trabalho é implementado um sistema que se baseia na informação de cor existente nas imagens e que permite segmentar a cena observada em diversas regiões, identificar regiões correspondendo a objectos conhecidos e efectuar o seguimento desses objectos ao longo do tempo. É também implementada uma forma, supervisionada pelo utilizador, de aprender os modelos de cor em objectos de interesse.

O presente documento mostra os objectivos do trabalho (Capítulo 2) assim como os conceitos teóricos sobre os quais se realiza (Capítulo 3). Faz-se uma descrição das tarefas levadas a termo (Capítulo 4) e mostram-se os resultados atingidos (Capítulo 5). Por último são expressas as conclusões que se obtêm da realização do trabalho (Capítulo 6).

2 Objectivos.

2.1 Objectivo.

O objectivo do trabalho realizado foi o estudo de diferentes técnicas de segmentação de cor aplicável ao seguimento de alvos e aprendizagem de tarefas em robots humanóides.

O objectivo fundamental é o desenvolvimento e implementação de algoritmos de segmentação de cor para detectar objectos simples colocados no ambiente do robot. Apesar de simples, os objectos podem ter forma variada e cores que podem alterar-se no tempo devido a diferenças de iluminação e a mudanças de pose. Por isso as técnicas estudadas deverão ser capazes de considerar estes problemas.

Pretende-se efectuar o estudo e implementação de dois módulos base para a funcionalidade do sistema: (i) segmentação inicial da cena em zonas distintamente coloridas representando potenciais objectos e, (ii) seguimento dos objectos detectados. No primeiro módulo pretende-se o estudo de técnicas de segmentação não supervisionada para conseguir dividir as imagens observadas num conjunto de regiões de cor aproximadamente uniformes. As métricas para avaliar a similaridade de cor e o espaço de cor utilizados deverão ser cuidadosamente analisados.

No segundo módulo pretende-se efectuar o seguimento de um objecto em movimento, modelando devidamente a distribuição de cor do objecto e efectuando a sua segmentação ao longo do tempo. A descrição da cor objecto poderá ser adaptada ao longo do tempo.

2.2 Integração do estudo.

Pretendendo integrar as técnicas de segmentação no seguimento de alvos e aprendizagem de tarefas em robots humanóides, o objectivo do trabalho abarca também a construção dum pequeno sistema que permite treinar objectos, segmentar uma imagem e extrair objectos de interesse, para depois poder fazer tracking de esse objecto ou objectos.

A criação dum espaço do trabalho também constitui um objectivo do presente trabalho. Dito espaço de trabalho deve permitir trabalhar com diferentes representações de cor. Também tem

que ser possível comparar os diferentes resultados dos algoritmos de segmentação de cor. Deve também integrar o processo global de treino e seguimento de objectos.

3 Conceitos, Técnicas e Metodologias.

3.1 Metodologia.

O trabalho foi concebido modularmente. A modularização permite a reutilização dos algoritmos que depois neste mesmo capítulo serão explicados.

Em coerência com os objectivos foram desenvolvidos módulos para a segmentação inicial duma cena assim e também módulos de segmentação de objectos simples. Outro módulo foi construído para a interface gráfica.

Para o controlo do seguimento dos objectos foi construído mais um módulo, e para o treino de objectos simples foi construído mais outro.

Um último módulo de busca e seguimento automático de objectos também foi desenhado.

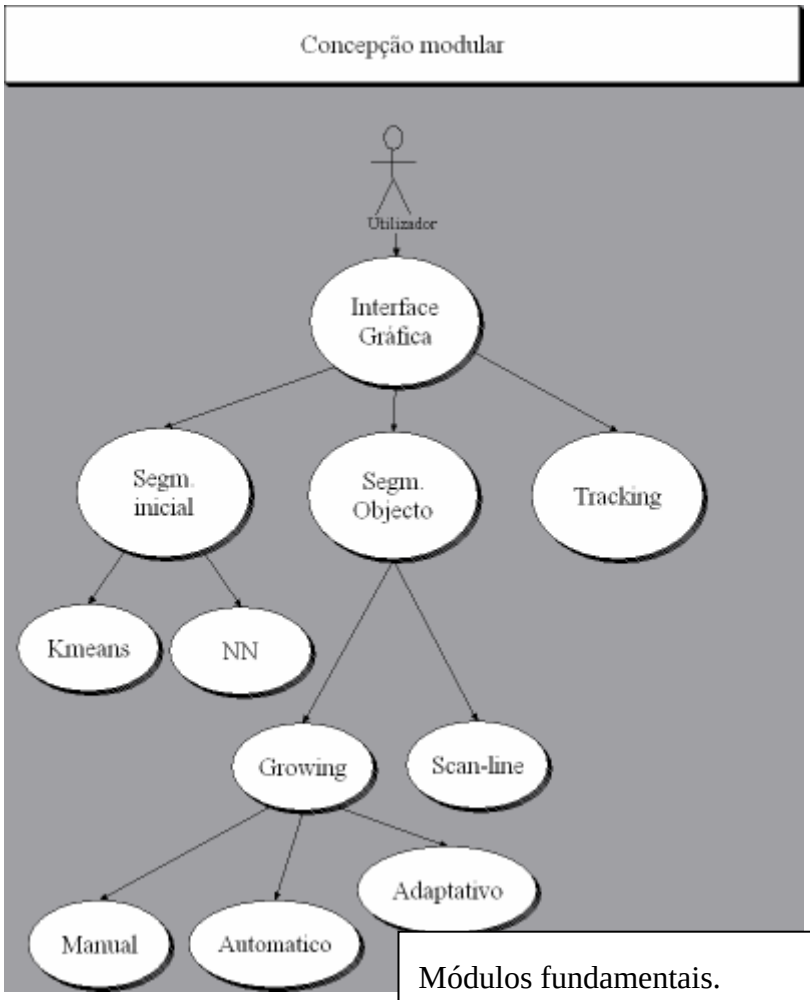

Figura 3.1

3.2 Coordenação.

Pelas características do grupo de trabalho, constituído por uma só pessoa, o fluxo de informação foi basicamente entre o estudante e o professor orientador. Também resultou fundamental a periodicidade semanal (quase diária) nas consultas e comentários do trabalho. Isto permitiu a correcção quase imediata do trabalho com uma concepção errónea.

3.3 Ferramentas.

As ferramentas para a resolução do problema são as seguintes.

3.3.1 Ferramentas Software.

MATLAB.

- MATLAB® é uma linguagem de programação de altas prestações para programação técnica. MATLAB Integra cálculo, visualização e programação num ambiente fácil de usar, onde os problemas e soluções são expressados em notação matemática.
- GUIDE. *Graphical User Interface development environment* permite a construção de GUIs *Graphical user interfaces* mediante uma série de ferramentas gráficas. Estas ferramentas facilitam o processo de desenho e criação de GUIs.
- Mex Files interface. Permite interactuar com MATLAB desde programas em C para a passagem de parâmetros, controlar o uso de memória e outras funções.
- Versão. 7.0.1.24704(R14) Service Pack 1.

Visual C++ 2005.

- Foi utilizado como editor para os programas em C.
- Versão. Microsoft Visual Studio 2005 Beta 1 Versão 8.0.40607.16.

IPP.

- Intel® Integrated Performance Primitives v4.0 *for Windows* on Intel® Pentium® and Itanium® processors.*
- IPP é uma API para o uso de funções em C que trabalham com imagens dum jeito muito rápido.

3.3.2 Hardware.

Computador.

- Fujitsu Siemens Computers Mobile Inter®.
- Pentium® 4 CPU 3.20GHz.
- 1GB de RAM.

- ATI RADEON 9700.

Descrição da câmara.

- Sensor. CMOS.
- Pixels. 352 (H) x 288 (V).
- Still image resolution. 640 (H) x 480 (V).
- Illumination.< 10 lux.
- Integrated lens. F2.0.

3.4 Conceitos

3.4.1 Segmentação da cor.

Este termo refere-se a segmentação duma imagem em relação à cor dos seus pixels. Isto é, uma imagem é dividida num conjunto de regiões que a cobrem. As regiões são definidas em relação a uma ou varias características como cor, contornos, sombra e outras.

A segmentação tradicionalmente tem dois objectivos:

- Decompor a imagem em partes para analises posteriores.
- Realizar uma transformação na representação. Os pixels são classificados em unidades de mais alto nível que têm maior significado ou são mas eficientes para a análise posterior.

3.4.2 Representações de cor.

Uma representação da cor pode ter umas propriedades que proporcionam vantagens ou desvantagens em relação a outras representações. Vantagens como uma maior diferenciação entre cores ou uma maior facilidade de conversão para outras representações da cor.

As representações de estudo são RGB, por ser uma representação baseada nas três cores básicas e também por ser o standard na representação de cor nos monitores, L*a*b* e HSV por ser representações nas quais o cromatismo é representado por duas variáveis.

3.4.2.1 RGB.

Esta representação da cor está directamente relacionada com a forma em que as pessoas percebem a cor do mundo. A representação corresponde às três cores fundamentais, vermelho, verde e azul.

3.4.2.2 HSI ou HSV.

HSV (Hue Saturation Intensity ou Hue Saturation Value) usa dois valores para codificar o cromatismo de cor, separando estes dum terceiro valor de Intensidade (I ou V).

- H define a cor.
- S define a pureza da cor (H).
- I representa a intensidade.

HSV tem uma boa aplicação em algoritmos informáticos porque as duas variáveis HS têm mais que ver com as propriedades da superfície que com a luz que está a incidir no objecto.

3.4.2.3 CIE 1976 (L* a* b*) colour space.

Lab, do mesmo jeito que os outros dois espaços de cor definidos, permite a especificação da percepção de cor num espaço tridimensional.

O eixo L* é conhecido como a luminosidade e varia de 0 (cor negra) a 100 (cor branca). As outras duas coordenadas a* e b* representam vermelho-verde e amarelo-azul respectivamente. Então têm-se que a*b* representam a cromatismo do mesmo jeito que HS no espaço HSV.

3.4.3 Métricas de distância.

Resulta interessante estudar como a utilização de diferentes distâncias pode fazer mudar o jeito em que uma imagem é segmentada. A distância de um ponto à média pode definir se o ponto é incluído na região ou não.

O espaço de trabalho vai ser 2-dimensional porque os espaços de trabalho das imagens HS e a*b* são 2-dimensional. Portanto e importante analisar a forma em que essas duas variáveis estão relacionadas na definição de distância a utilizar.

Quando se fala de distância não nos referimos a distância espacial, mas a distância de cromatismo.

- **a** é a primeira dimensão.
- **b** é a segunda dimensão.
- $\bar{x}$ é a média da distribuição.

3.4.3.1 Distância Euclidiana duas dimensões.

$$d = \sqrt{(\bar{x}_a - a)^2 + (\bar{x}_b - b)^2} \quad (3.1)$$

Esta distância não tem em conta a variância da distribuição conjunta entre as duas variáveis.

3.4.3.2 Distância de Mahalanobis duas dimensões.

$$d^2 = (x - \mu)^T \Sigma^{-1} (x - \mu) \quad (3.2)$$

$(x - \mu)$: Média menos o novo ponto.

Σ: matriz de covariância.

Tem em conta a distribuição conjunta entre as duas variáveis.

3.4.4 Modelo de objectos.

Os objectos de trabalho são objectos homogéneos numa representação da cor bidimensional.

Os objectos estão caracterizados por a média de cada uma das duas dimensões, e pela matriz de covariância dos pontos do objecto.

O objecto assim pode ser modelado como uma elipse não alinhada com os eixos da representação da cor.

Sirva a seguinte figura como exemplo.

Figura 3.2.

3.5 Armazenagem de dados.

Neste trabalho a forma de guardar os dados referentes aos objectos treinados realiza-se por meio de um simples ficheiro onde se guardarão as características mais importantes dos objectos e estudo.

3.6 Interface gráfica.

A interface gráfica tem a função de fazer de nexo entre os diferentes módulos da aplicação. A aplicação tem com ponto de partida para qualquer fluxo de trabalho a interface gráfica.

O seguinte esquema mostra esta concepção.

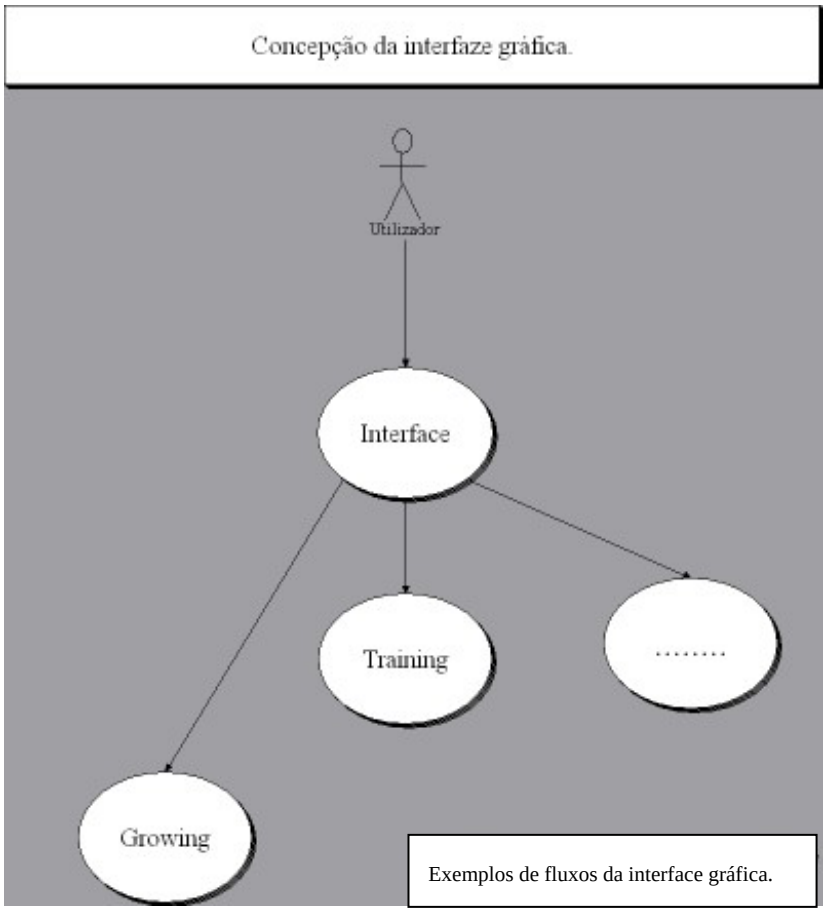

Figura 3.3

Como se pode ver no esquema da Figura 3.3 o utilizador tem uma grande importância, pelo que a interface gráfica está orientada a facilitar o uso da aplicação.

3.7 Esquema do desenho dos fluxos de treinamento e seguimento.

Nesta seção estão comentados os fluxos das operações de treinar e de seguimento automático de um objecto uma vez reconhecido como interessante.

Para o fluxo “Treinar” as operações efectuadas são:

- Extracção do objecto de treino mediante k-means, Region-Growing ou Scan-line.
- Guardar características do objecto se o utilizador assim o quer.

Para o fluxo “Olhar ou seguimento automático” as operações efectuadas são:

- Decomposição da cena mediante k-means.
- Extracção dos objectos da decomposição previa.
- Buscar objectos similares os segmentados no ficheiro de treinamento do sistema.
- Centrar a atenção no objecto mais grande.
- Tracking do objecto no que o sistema centrou a sua atenção.

Estes fluxos estão representados na figura 3.3.

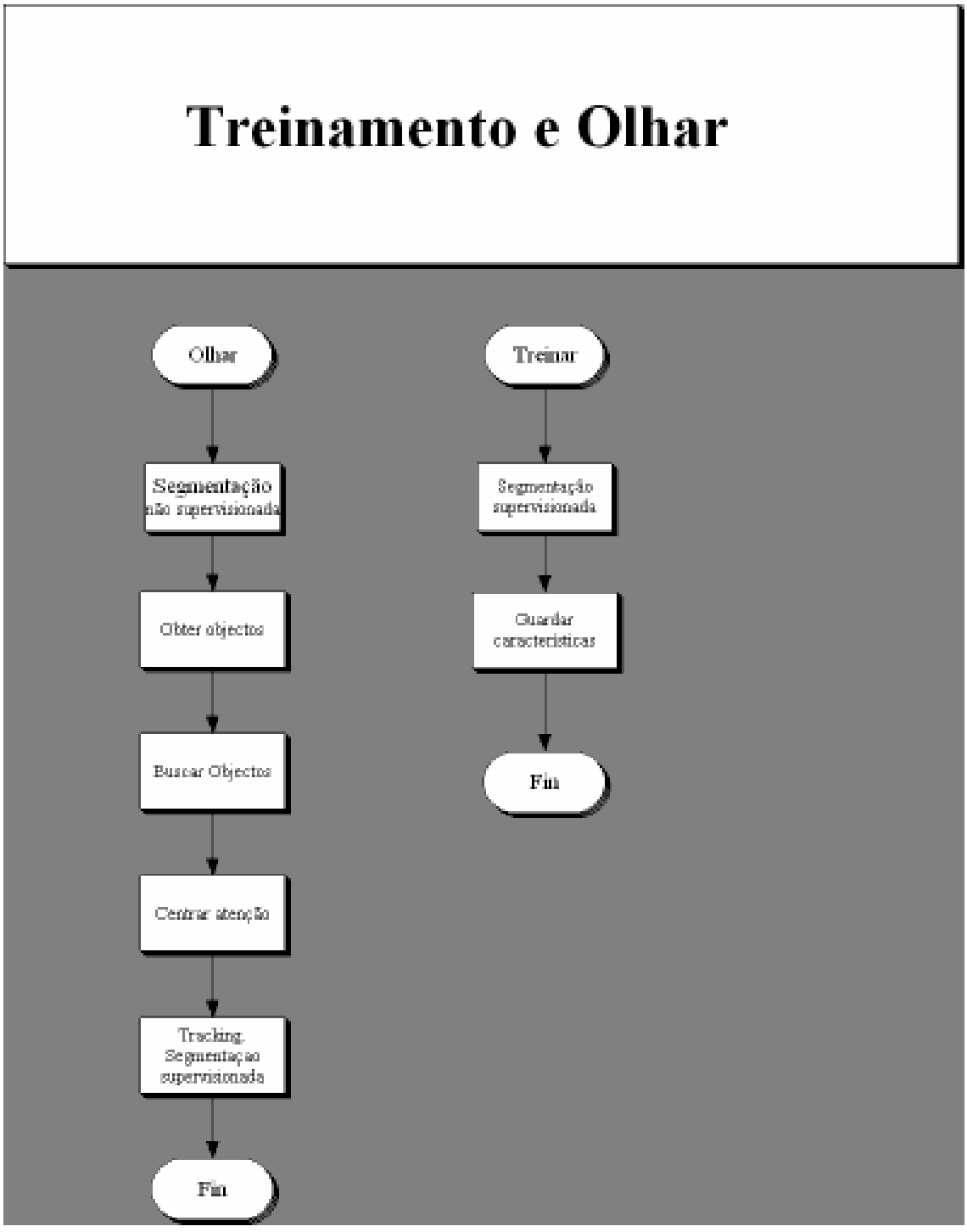
Treinamento e Olhar
Olhar
Segmentação
não supervisionada
Obter objectos
Buscar Objectos
Centrar atenção
Tracking.
Segmentação
supervisionada
Fim
Treinar
Segmentação
supervisionada
Guardar
características
Fim

Figura 3.3

4 Descrição do Trabalho.

O trabalho principal feito foi a construção dos módulos descrito na figura 3.1. Os principais módulos construídos foram:

- Segmentação K-means.
- Segmentação com Growing.
- Segmentação com Scan-line.
- Tracking.
- Treino de objectos.
- Busca automática de objectos.

4.1 Descrição da interface gráfica.

Para a integração dos diferentes algoritmos, técnicas, e para facilitar o uso de todos os componentes da aplicação construiu-se um espaço de trabalho que tem o aspecto seguinte.

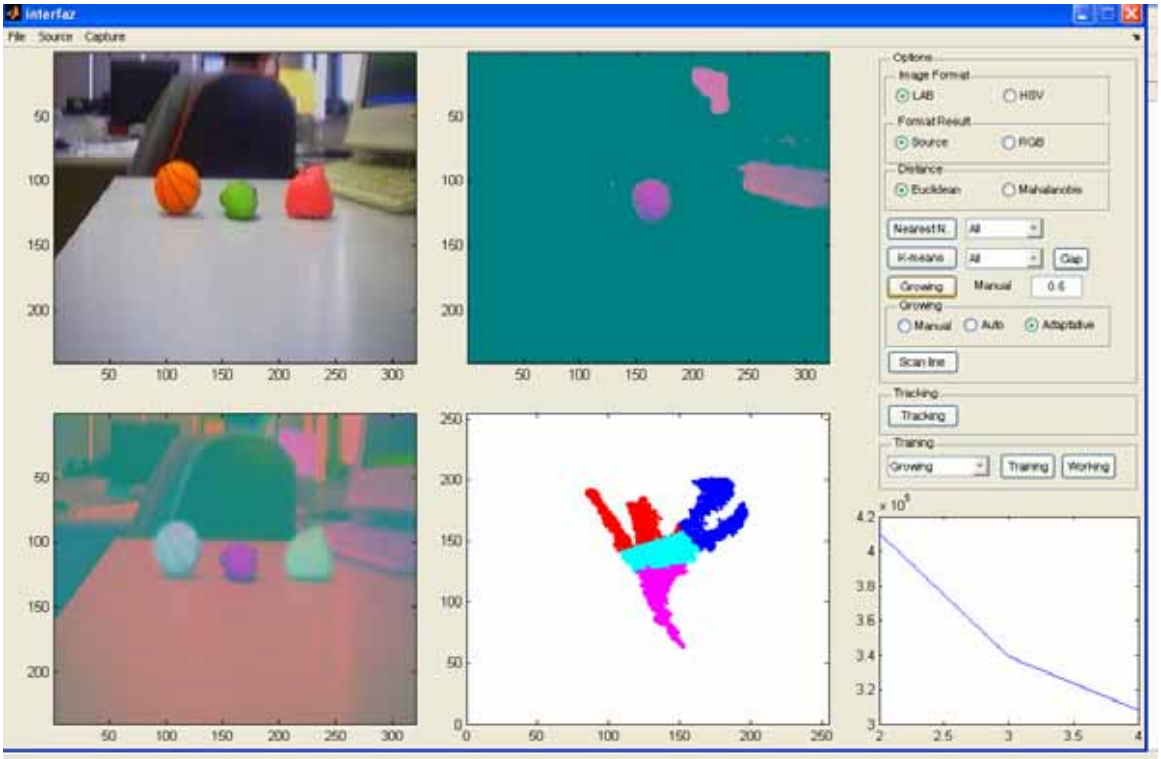

Figura 4.1

O espaço de trabalho divide-se nas seguintes partes.

4.1.1 Menu Principal

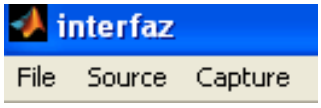

Figura 4.2

As acções que se podem realizar são.

File→Stop: permite parar o tracking ou parar a busca automática de objectos.

File→Exit: permite sair da aplicação.

Source→Camera: Estabelece a câmara como fonte das imagens.

Source→Video: Estabelece o vídeo seleccionado como fonte das imagens.

Capture→Camera: Realiza uma captura da câmara.

Capture→Video: Inicia o vídeo para o primeiro frame.

4.1.2 Imagens

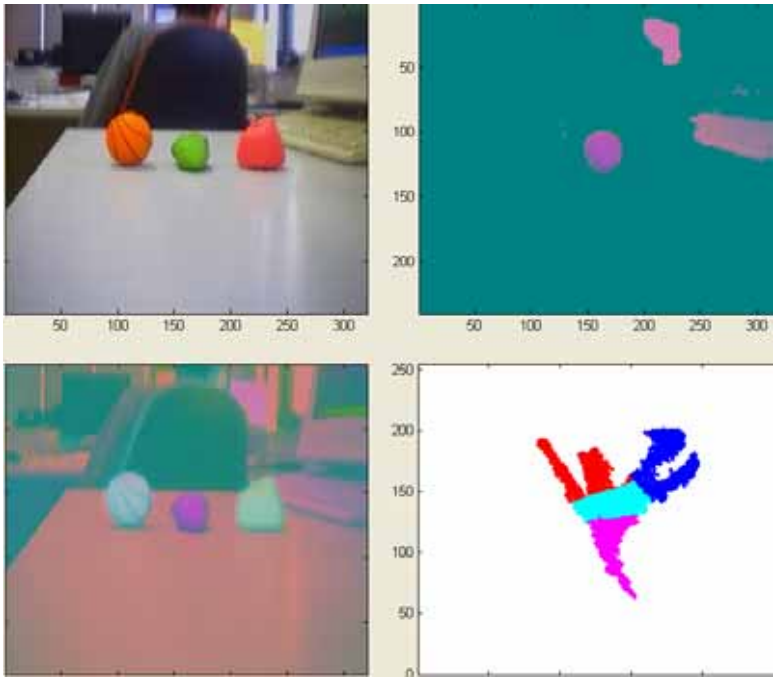

Figura 4.3

O espaço das imagens permite.

Superior Esquerda: permite ver a imagem original em formato RGB.

Inferior Esquerda: permite ver a imagem original em formato RGB ou HSV.

Superior Direita: permite ver a imagem processada por algum dos algoritmos.

Inferior Esquerda: permite ver os pontos dos objectos processados na representação da cor eleita.

4.1.3 Opções.

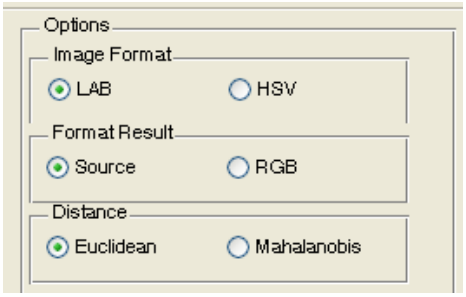

Figura 4.4

As opções permitem a escolha entre:

Image Format: A imagem de trabalho será processada em HSV ou LAB.

Format Result: A imagem depois de ser processada é mostrada no mesmo formato que indica **Image Format** ou RGB.

Distance: A distância de trabalho para alguns algoritmos pode ser determinada aqui: Mahalanobis ou Euclidiana.

4.1.4 Algoritmos.

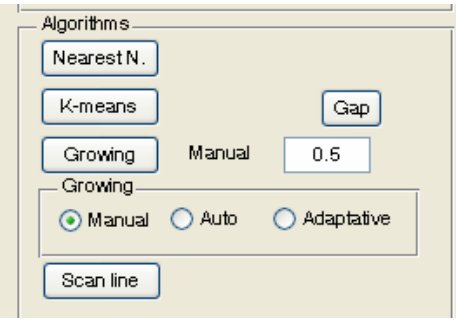

Figura 4.5

O painel Algorithms permite o seguinte.

Nearest N.: O algoritmo Nearest Neighbour para 2 clusters.

K-means: O algoritmo k-means que decide automaticamente o número de clusters.

Manual: Permite especificar o nível do limiar para os algoritmos Growing e Scan-line.

Growing: O algoritmo Growing Regions.

Growing Options:

- **Manual.** O threshold é calculado pelo utilizador.
- **Auto.** O threshold é calculado de jeito automático pelo algoritmo.
- **Adaptative.** O threshold é calculado automaticamente e vai mudando.

Scan-line: O algoritmo Scan-line.

Gap: O algoritmo Gap.

4.1.5 Tracking.

Figura 4.6

O painel tracking permite o seguinte.

Tracking: O algoritmo de tracking baseado em Scan-line. O conteúdo de **Manual** é tido em conta.

4.1.6 Training.

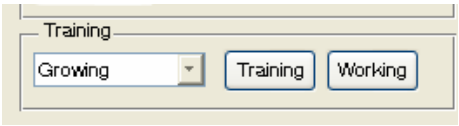

Figura 4.7

O painel training permite o seguinte.

Training: O algoritmo de training.

- **Growing**: Treino com Growing.
- **Scan-line**: Treino com Scan-line.
- **K-means**: Treino com k-means (múltiplos objectos).

Quando um objecto é treinado o seguinte painel é visível:

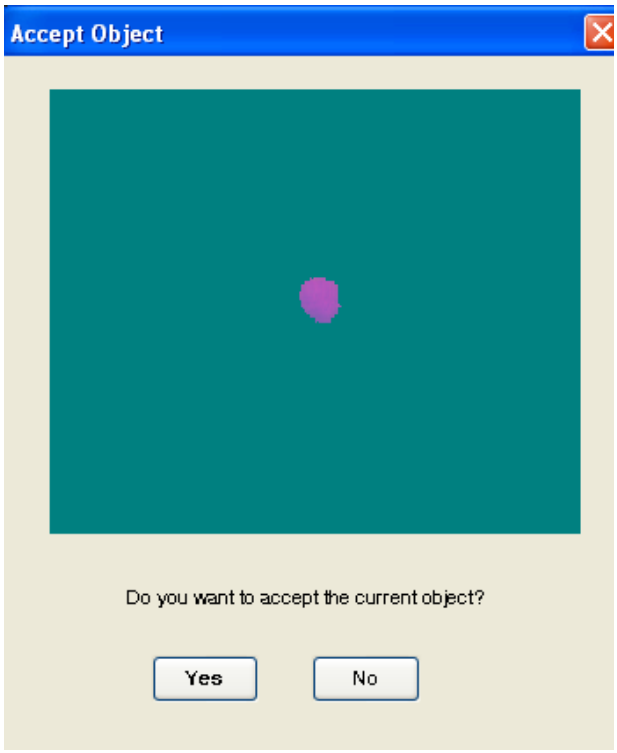

Figura 4.8

Este painel permite escolher entre aceitar ou recusar o objecto.

4.1.7 Information Zone.

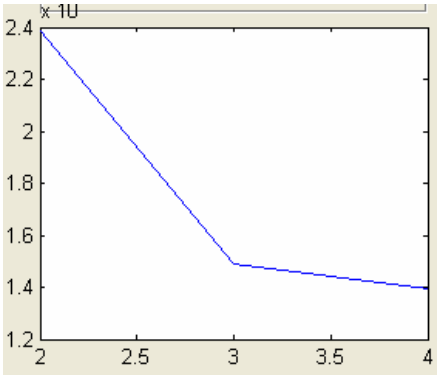

Figura 4.9

A zona de informação mostra informação adicional do processo ou actividade realizada.

4.2 Algoritmos.

4.2.1 Algoritmos de segmentação inicial.

Para a segmentação inicial os algoritmos estudados foram:

4.2.1.1 Nearest-Neighbour.

O Nearest-Neighbour (NN) segmenta a imagem em K clusters definidos com uma inicialização. Calcula a distância de todos os pontos à média de cada cluster e decide qual é o cluster que está mais perto do ponto no espaço da cor que está a ser utilizada.

O NN foi implementado e testado com formatos de imagem HSV e L*a*b*. Também foi testado para poder trabalhar com distância Euclidiana e como distância de Mahalanobis. O algoritmo resolveu-se como pouco efectivo ante backgrounds não uniformes. Resulta de interesse para olhar as diferenças entre a distância Euclidiana e a distância de Mahalanobis aplicadas a um algoritmo simples.

4.2.1.2 K-means clustering.

K-means clustering segmenta a imagem em k clusters como também faz o Nearest-Neighbour. Não precisa de iniciar nenhuma região porque é o próprio algoritmo quem inicia as regiões. Classifica cada ponto numa região dependendo da distância à média de cada cluster. Recalcula as médias de cada cluster e volta a fazer a classificação. Repete o processo até não fazer modificações.

Descrição formal:

- Iniciar ic (iteration count) a 1
- Fazer uma escolha aleatoriamente de um conjunto de K medias $m_1(1), m_2(1), ...m_k(1)$
- Para cada vector x_i calcular $D(x_i, m_k(ic))$ para cada $k = 1, \ldots, K$ e atribuir x_i ao cluster C_j com a media mais próxima.
- Incrementar ic por 1 e actualizar as medias para obter um novo conjunto $m_1(ic), m_2(ic), ...m_k(ic)$.
- Repetir passo 3 e 4 hasta $C_k(ic) = C_k(ic+1)$ para todo k.

No K-means clustering, tem-se feita uma implementação do algoritmo com formatos de imagens HSV e L*a*b, neste caso a distância usada no algoritmo é a distância Euclidiana. Na implementação resolveu-se achar o número de clusters nos que dividir a imagem dum jeito automático. O algoritmo vai incrementando o número de clusters K até que a diferença do erro entre duas segmentações diferentes não aumenta.

Na imagem seguinte pode-se ver onde que o algoritmo resolve parar.

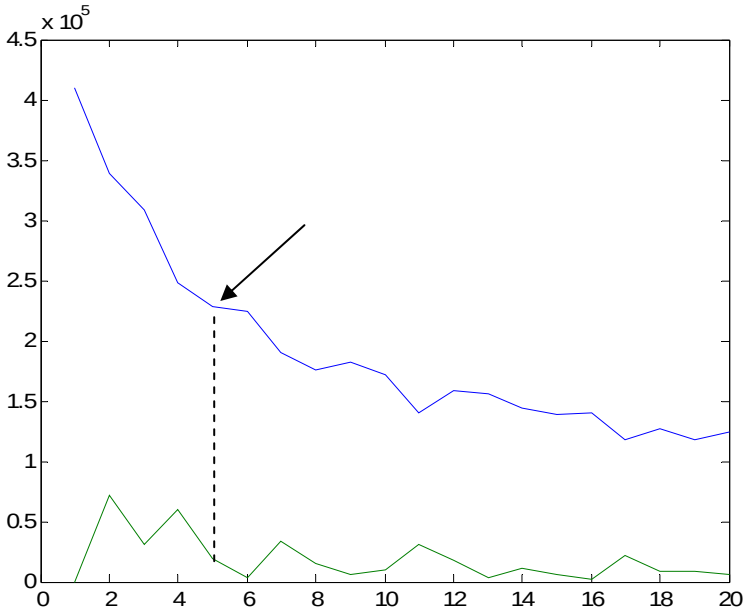

Figura 4.10

A série superior mostra o erro do algoritmo k-means que é calculado como a distância dos pontos pertencentes a cada cluster ao centroide do mesmo cluster.

A série inferior mostra a diferença do erro entre unha segmentação com n-1 clusters e unha segmentação com n clusters.

4.2.2 Algoritmos de seguimento.

Os algoritmos para fazer o seguimento dum objecto precisam de inicialização. É necessária uma região inicial e também certas propriedades da cor do objecto, como a media e a variância.

4.2.2.1 Growing.

Os algoritmos de growing avaliam se os pontos da vizinhança pertencem à região. Se pertencem, os pontos são incluídos e novos pontos da vizinhança são avaliados. O critério para avaliar se o ponto pertence ou não a região é a distância cromática.

A medida que os novos pontos entram na região esta vai crescendo e novos pontos da vizinhança são avaliados como pertencentes à região. Isto permite que no final do growing o objecto obtido esteja conectado.

No caso do growing diferentes alternativas do algoritmo foram avaliadas em referencia ao threshold que determina se um ponto e admitido ou não.

- Growing com threshold manual e fixo.
- Growing com threshold automático e fixo.
- Growing com threshold adaptativo.

Como generalidade os algoritmos de growing foram implementados do jeito seguinte:

A partir de uma região inicial, analisa-se a vizinhança de cada um dos pixels pertencentes à região. Se estes pixels são incluídos na região, uma nova vizinhança é analisada.

Os algoritmos de Growing implementados e estudados foram os seguintes.

- Threshold manual e fixo, com distância Euclidiana.
- Threshold manual e fixo, com distância de Mahalanobis.
- Threshold automático fixo com distância Euclidiana.
- Threshold automático fixo com distância de Mahalanobis.
- Threshold automático adaptativo com distância Euclidiana.
- Threshold automático adaptativo com distância de Mahalanobis.

Outros algoritmos estudados previamente foram

- Growing com threshold fixo com distância Euclidiana mas tratado só na zona de vizinhança do pixel candidato.

4.2.2.2 Scan-line.

O algoritmo Scan-line determina, de uma forma sequencial e mais eficiente que o growing, se os pontos de uma janela definida ao redor da região de estudo pertencem a essa região.

O algoritmo Scan-line foi implementado com distância de Mahalanobis dos pontos incluídos na janela, de tamanho variável, ao redor da distribuição da região inicial.

4.2.3 Algoritmo de tracking.

Foi implementado um algoritmo simples de seguimento de objectos baseado no algoritmo de Scan-line com uma janela variável.

O modelo para fazer o tracking é o Modelo de Velocidade Constante. Neste modelo, considera-se que os objectos têm uma velocidade constante.

Para fazer uma previsão do movimento utilizam-se as equações de predições do movimento.

As equações de actualização de velocidade são depois utilizadas para corrigir a predição em função das medições efectuadas.

$\boxed{\dot{x}(t+1|t) = \dot{x}(t|t)}$ (4.1) é a equação de predição da velocidade. A velocidade no instante futuro é a mesma que no instante actual.

$\boxed{x(t+1|t) = x(t|t) + T \cdot \dot{x}(t+1|t)}$ (4.2) é a equação de predição da posição. A posição no instante futuro é definida em função da posição actual e a velocidade predita.

$\boxed{\dot{x}(t|t) = \dot{x}(t|t-1) + \frac{h}{T}(y(t) - x(t|t-1))}$ (4.3) é a equação de actualização da velocidade no que $y(t)$ é a posição real do objecto.

$\boxed{x(t|t) = x(t|t-1) + g(y(t) - x(t|t-1))}$ (4.4) é a equação de actualização do espaço no que $y(t)$ é a posição real do objecto.

$y(t)$ é calculado como o centroide do objecto que é segmentado por Scan-line para cada instante t.

Se há oclusão ou o objecto está fora da imagem então $y(t)$ é indeterminado e faz-se:

$\boxed{\dot{x}(t|t) = \dot{x}(t|t-1)}$ (4.5)

$\boxed{x(t|t) = x(t|t-1)}$ (4.6)

Os parâmetros h e g são ajustados para obter um bom regime transitório. Fazendo $h = 1-\theta^2$ e $g = (1-\theta)^2$ obtém-se um transitório suave. O parâmetro θ é ajustado para obter tempos de resposta adequados. Tipicamente faz-se $0.25 < \theta < 0.75$.

4.2.4 Gap statistic.

O GAP é um sistema para detectar cal é o melhor k a utilizar para a segmentação com o k-means.

O critério GAP statistic compara a curva $\log W_k$, donde W_k é o erro do k-means para uma divisão duma imagem em k clusters, com a curva obtida dos dados uniformemente distribuídos num rectângulo que contem os dados origem.

As curvas são do modo seguinte:

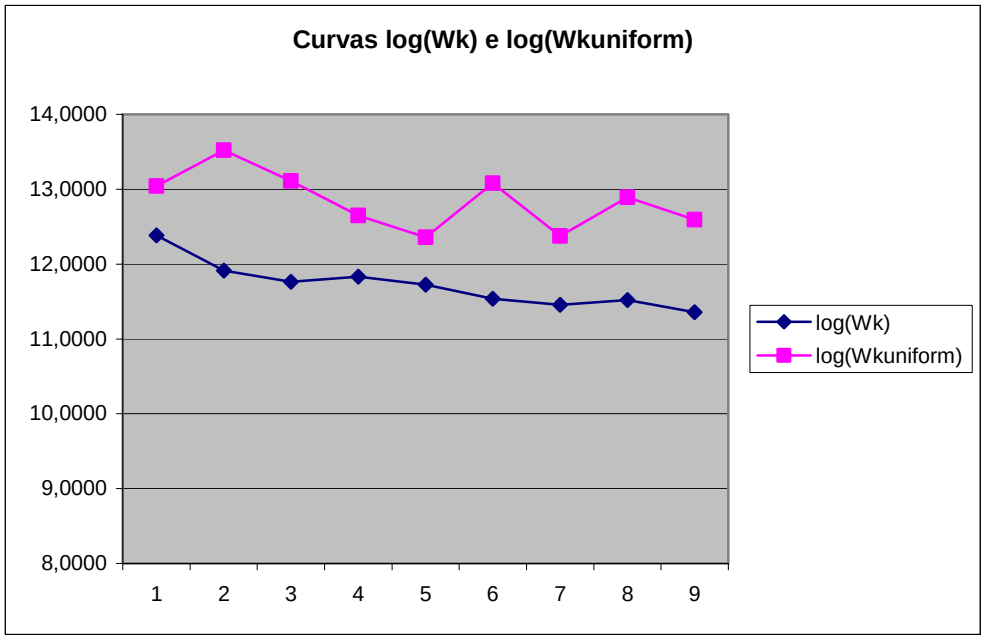

Figura 4.11

O GAP statistic faz a subtracção das duas curvas: log(Wkuniform)-log(Wk) e escolhe o k que maximiza essa diferença.

O problema é que o algoritmo precisa fazer a segmentação para um máximo de clusters que não tem porque ser conhecido. No caso da figura 4.11 o máximo foi definido como 10.

4.3 Treino de Objectos.

O treino consiste em obter a média e covariância da cor numa região identificada como objecto de interesse e guardar esses valores.

Para que sistema tenha a capacidade de aprender foi implementado um sistema de treino. O sistema de treino pode decompor a cena em diversos objectos ou a partir de uma selecção duma região pode construir o objecto.

No caso de decompor uma cena o algoritmo que utiliza o sistema é o k-means. Cada um dos clusters que é retornado por o k-means é decomposto em objectos independentes. Cada um de esses objectos é avaliado pelo utilizador como um objecto interessante ou não.

No caso de utilizar uma região inicial para construir o objecto, os algoritmos utilizados são Growing e Scan-line. Neste caso o utilizador também tem a oportunidade de escolher se o objecto é válido ou não é.

O growing permite definir bem os pixels que pertencem ao objecto. No scan-line tem que haver um processo posterior para garantir a conectividade dos objectos.

Uma vez se decide que o objecto é valido, são guardadas as suas imagens em formato LAB, HSV e RGB, depois as características de cor do objecto são guardadas na armazenagem de dados. As características guardadas são a media nas duas variáveis de cor, e a matriz de covariância.

Os objectos válidos são os que o sistema pode buscar numa cena de jeito automático.

Os objectos descartados também são guardados para futuro uso.

4.4 Buscando objectos.

O sistema pode buscar objectos previamente treinados numa cena nova. Isto permite o robot buscar objectos que já conhece numa habitação ou noutro tipo de situações.

Para fazer isto, o sistema decompõe com o algoritmo k-means a cena, cada um dos clusters e dividido em diferentes objectos, cada um desses objectos é comparado com os objectos que o sistema já tem treinado.

O critério para a escolha do objecto a seguir pelo sistema e muito simples, o sistema segue o objecto maior, que o sistema reconheceu como parecido com aqueles que já reconheceu previamente.

Depois de que o objecto seja escolhido, o sistema segue o objecto por meio do tracking.

Desta busca de objectos pode resultar que nenhum objecto seja reconhecido como treinado previamente. O sistema neste caso não segue nenhum objecto.

5 Resultados.

Neste capítulo são apresentados os resultados mais importantes do trabalho realizado. Os resultados estão relacionados com a análise comparativo da métrica de distância utilizada (Capítulo 5.1), com a comparação entre as representações da cor HSV e LAB (Capítulo 5.2), também são discutidas as diferenças entre os diferentes algoritmos Growing/Scan-line implementados (Capítulo 5.3). A seguir a estes estudos mostram-se alguns resultados do algoritmo k-means (Capítulo 5.4) e também resultados com o algoritmo Scan-line (Capítulo 5.5). Posteriormente é analisado o tracking e a sua robustez (Capítulo 5.6) e comenta-se o modelo de objectos criados pelo treino (Capítulo 5.7). Um exemplo da busca automática de objectos pode-se ver na seguinte seção (Capítulo 5.8). Por último mostra-se uma discussão do critério GAP e a sua relação com o algoritmo k-means na escolha do k.

5.1 Diferenças Euclidiana-Mahalanobis.

As seguintes experiências mostram como a distância de Mahalanobis aporta maior flexibilidade para a construção dos modelos dos objectos.

Utilizando a distância de Mahalanobis o objecto pode ser modelado como uma elipse não alinhada com os eixos da representação bidimensional da cor escolhida. Isto não é possível com a utilização da distância euclidiana.

Para mostrar isto podemos ver como são representados os diferentes clusters no algoritmo Nearest-Neighbour.

Com distância euclidiana a criação dos clusters e da forma seguinte:

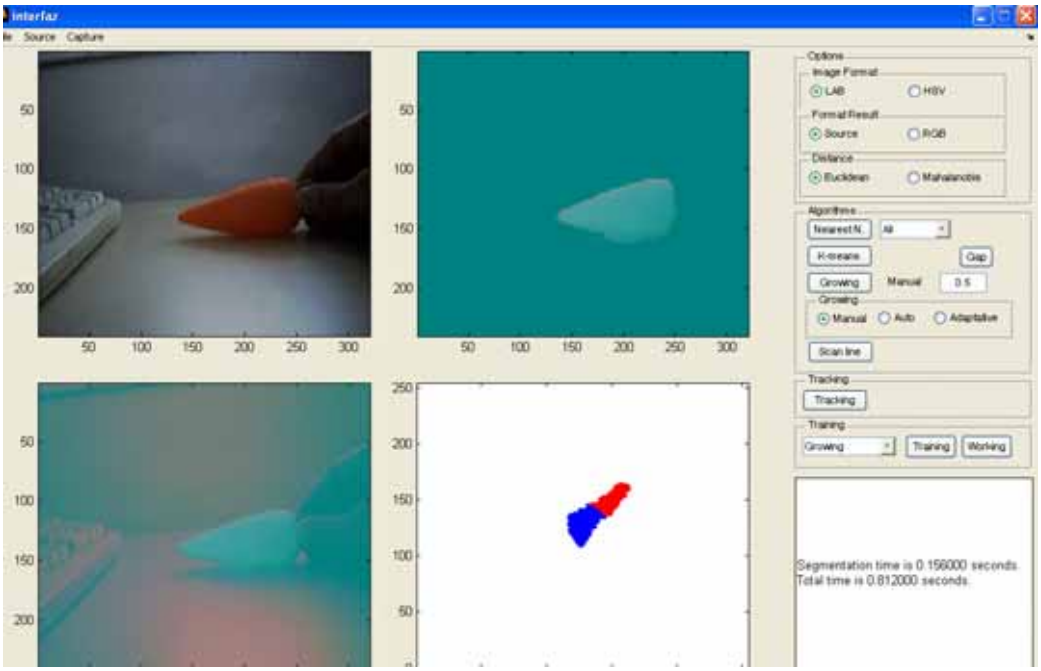
interfaz
File Source Capture
Options
Image Format
LAB
HSV
Format Result
Source
RGB
Distance
Euclidean
Mahalanobis
Algorithms
Nearest N.
K-means
Gap
Growing
Manual
0.5
Growing
Manual
Auto
Adaptative
Scan line
Tracking
Tracking
Training
Growing
Training
Working
Segmentation time is 0.156000 seconds.
Total time is 0.812000 seconds.

Figura 5.1

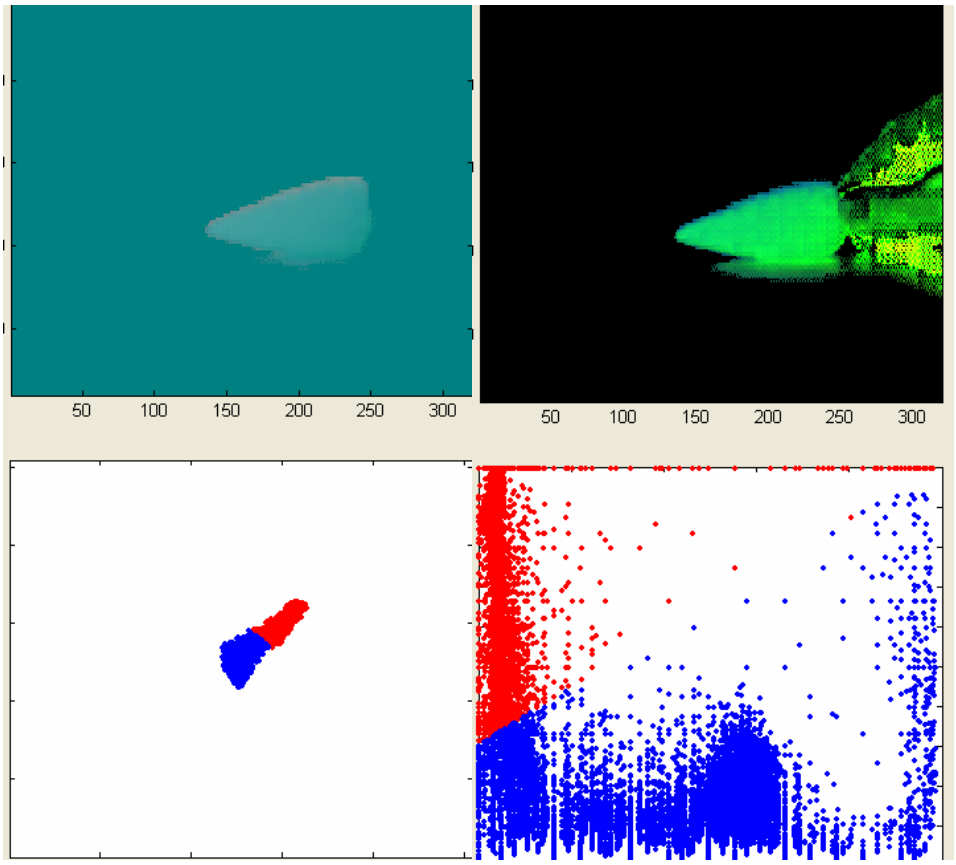

Figura 5.2

Os clusters são divididos por uma línea recta, fruto da utilização da distância euclidiana. Na figura 5.2 pode-se ver a divisão de clusters para a representação da cor LAB e HSV respectivamente.

Utilizando a distância de Mahalanobis para a mesma situação.

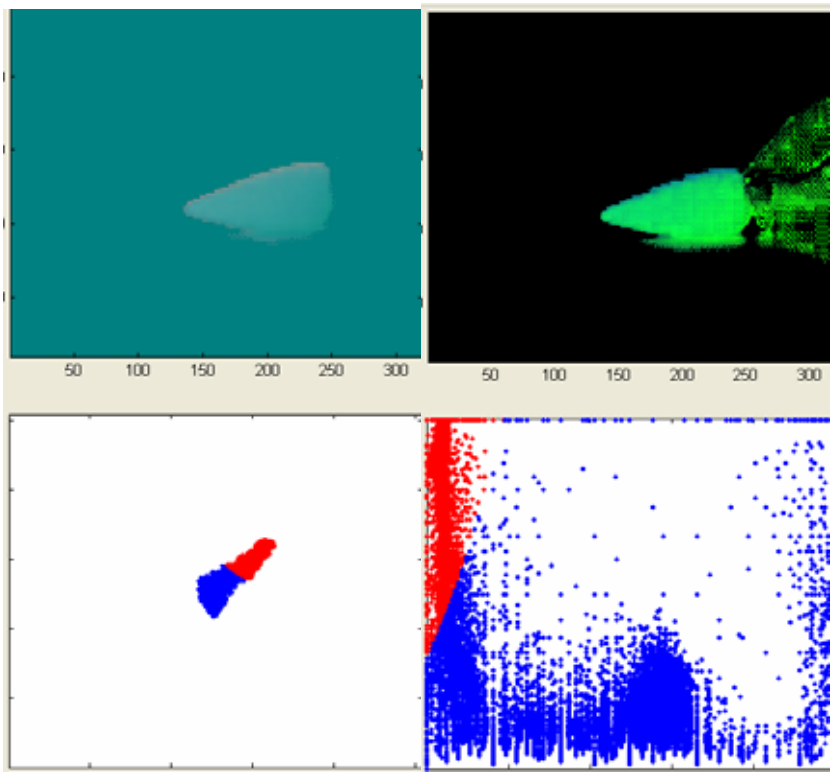

Figura 5.3

Na figura 5.3 pode-se ver a segmentação com Nearest-Neighbour utilizando a distância de Mahalanobis.

Na representação HSV é mais claro como os clusters são formados com fronteiras não rectas no caso da distância de Mahalanobis e sim no caso da distância euclidiana.

5.2 Comparações do HSV-LAB.

Os seguintes resultados mostram como a representação da cor LAB e melhor em termos de representação de objectos.

Realizou-se tracking de diferentes sequências de vídeo para diferentes objectos. Para cada frame calculou-se o número de pontos pertencentes ao objecto no que se está a fazer tracking. Calculou-se a diferença entre o número de pontos, e essa diferença e a que é mostrada nas seguintes figuras para diferentes sequências de tracking.

As diferenças de pontos entre frames também foram normalizadas pelo número de pontos pertencentes ao objecto.

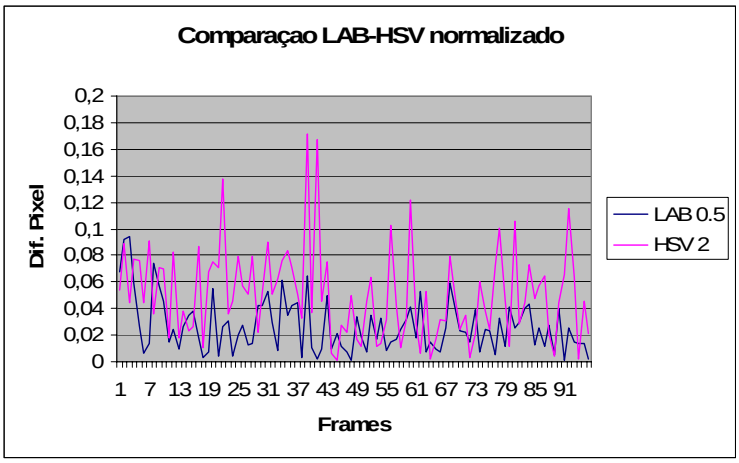

Figura 5.4

Na figura 5.4 o objecto do tracking está sob umas condições de luminosidade variáveis. Mostra-se no eixo x os frames da sequência de estudo, e no eixo y a diferença de pixels entre frames normalizada pelo número de pixels do objecto. A representação LAB parece mais robusta pela menor variabilidade no número de pontos do objecto ao largo dos frames.

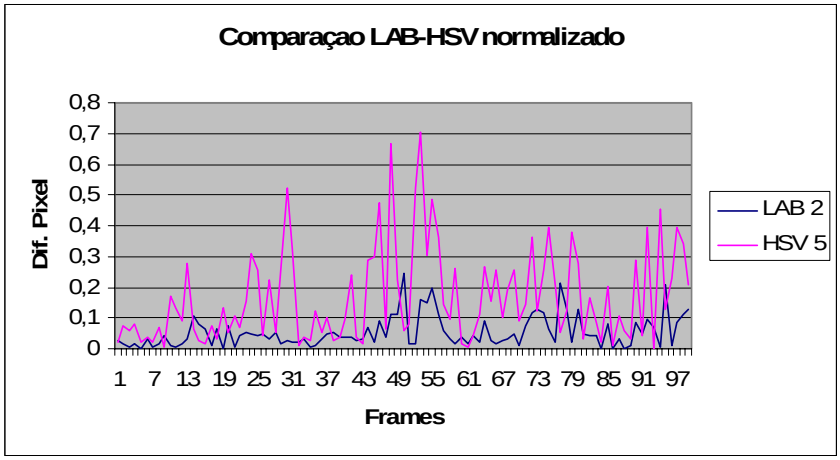

Figura 5.5

Do mesmo jeito que para a Figura 5.4 o objecto está baixo umas condições de luminosidade não óptimas. A representação LAB é de novo melhor.

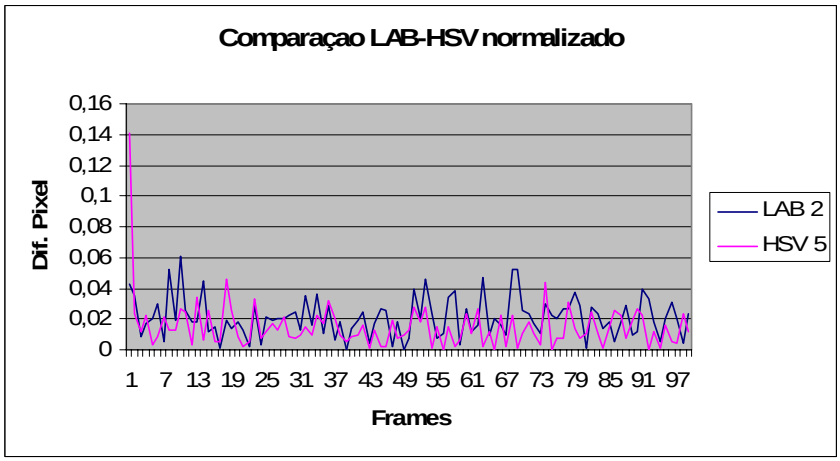

Figura 5.6

Para a figura 5.6 as condições da sequência de vídeo testada são melhores em quanto a iluminação. Neste caso a representação HSV pode competir com a representação LAB.

Como conclusão a representação LAB é mais robusta que a representação HSV, e discrimina melhor os pontos que pertencem ao objecto.

5.3 Comparações dos diferentes Growing/Scan-line.

Os objectos obtidos com as diferentes segmentações estudadas são muito semelhantes uns a outros. A escolha na utilização do Scan-line como algoritmo de segmentação com inicialização é motivada pelo seu melhor desempenho em questões temporais.

Nas seguintes figuras pode-se observar a comparação dos diferentes tempos de execução dos algoritmos.

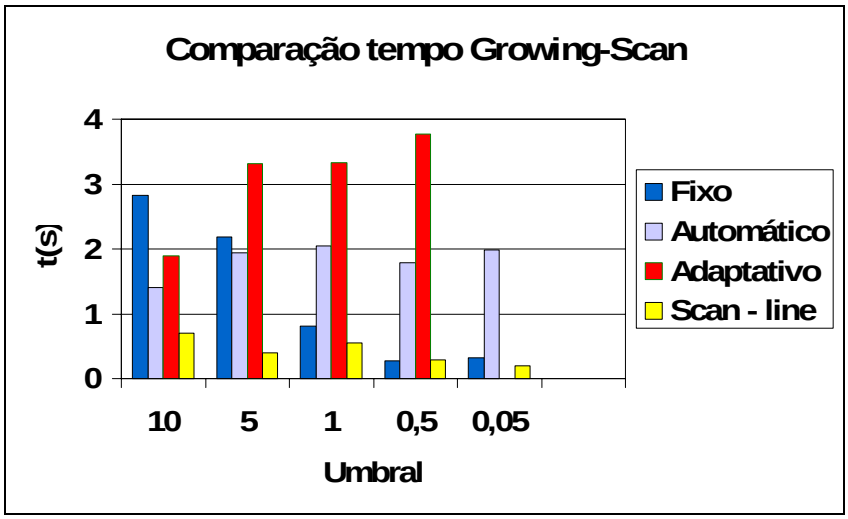

Figura 5.7

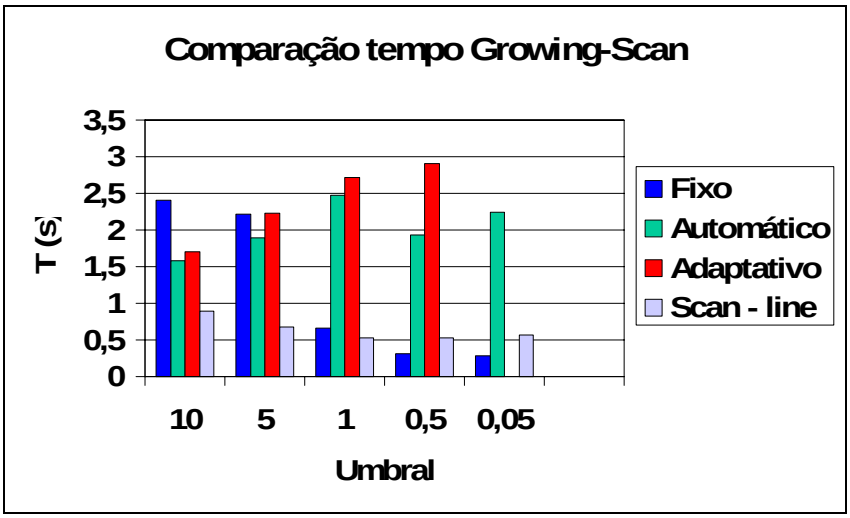

Figura 5.8

As lendas Fixo, Automático e Adaptativo correspondem com Growing com Threshold fixo, Growing com Threshold automático e Growing com Threshold Adaptativo.

Na figura 5.7 os tempos mostrados são para a representação da cor LAB. Os tempos mais baixos são os correspondentes ao algoritmo Scan-line. Os dados para threshold fixo que são mais baixos, não são representativos porque os objectos obtidos não representam ao objecto real. Isto mesmo é observável na figura 5.8 para os tempos obtidos para a representação da cor HSV.

Pelo tanto o algoritmo que melhores e mais rápidos resultados obtém é o Scan-line.

O algoritmo de Growing com threshold fixo é o único no que o utilizador tem que decidir o limiar que decide que pontos pertencem ao objecto ou não o que o converte no algoritmo mais rígido dos algoritmos implementados.

Por outra parte também resulta interessante ver como o algoritmo que melhor adaptação tem a mudanças na iluminação é o Growing com Threshold Adaptativo. Os algoritmos Growing com Threshold automático e Scan-line têm o mesmo desempenho neste aspecto. O Growing com threshold fixo é que descreve um pior comportamento.

5.4 Experiências com k-means.

O algoritmo k-means foi testado para diferentes cenas. O resultado foi bom dado que objectos homogéneos foram divididos em clusters diferentes do background. O feito de que o número de clusters seja decido dum modo automático permite que o sistema obtenha eventualmente boas segmentações que diferenciam objectos de interesse de elementos do background.

Nas figuras seguintes pode-se apreciar a segmentação em diferentes clusters duma determinada cena.

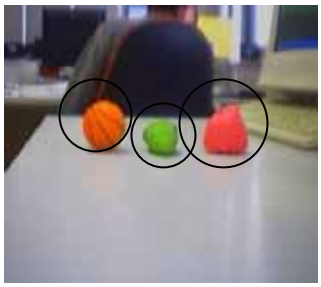

Figura 5.9

A figura 5.9 mostra a cena a ser segmentada pelo k-means e os objectos marcados como de interesse.

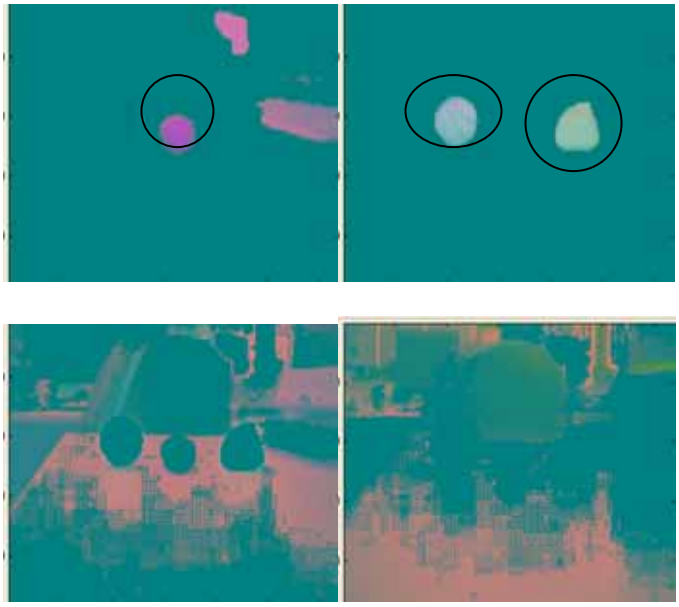

Figura 5.10

Na figura 5.10 podemos ver a segmentação que o algoritmo k-means efectua na cena original. Estão marcados os objectos de interesse.

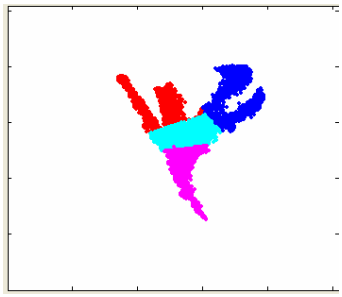

Figura 5.11

A figura 5.11 mostra como na representação da cor LAB é divida a cena inicial em quatro clusters.

5.5 Experiências com Scan-line.

Mostram-se na seguinte seção como o algoritmo Scan-line segmenta objectos com uma região inicial seleccionada.

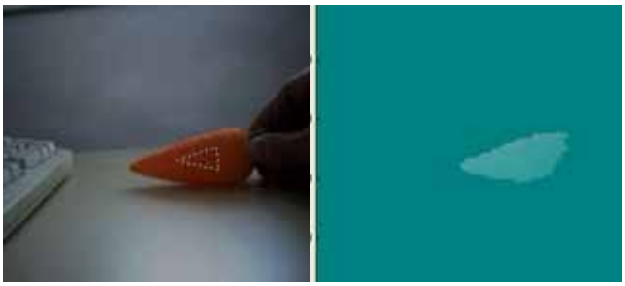

Figura 5.12

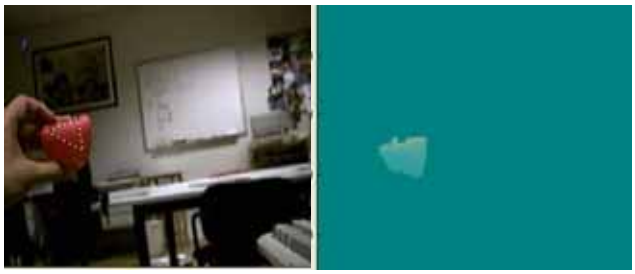

Figura 5.13

Figura 5.14

5.6 Tracking.

O modelo do tracking criado foi testado para muitas sequências de vídeo, entre as características mais importantes dos resultados obtidos pode-se destacar que o tracking e robusto frente câmbios da iluminação. O tracking também tem um bom comportamento frente a rotação dos objectos a seguir.

O modelo não foi preparado para ser muito robusto frente as oclusões e as saídas dos objectos da cena que o sistema está a olhar.

No modelo teórico do tracking fala-se (capítulo 3.9) da existência de um parâmetro θ que deve cumprir que $0.25 < \theta < 0.75$. Para estimar o valor correcto do θ realizou-se a seguinte experiência.

Foi testado o tracking para valores pequenos (0.30) e grandes (0.80) de θ. A continuação mostra-se umas figuras que mostram a posição do centroide real e estimado para o eixo X e o eixo Y para uma sequência de tracking linear e para uma sequência de tracking circular.

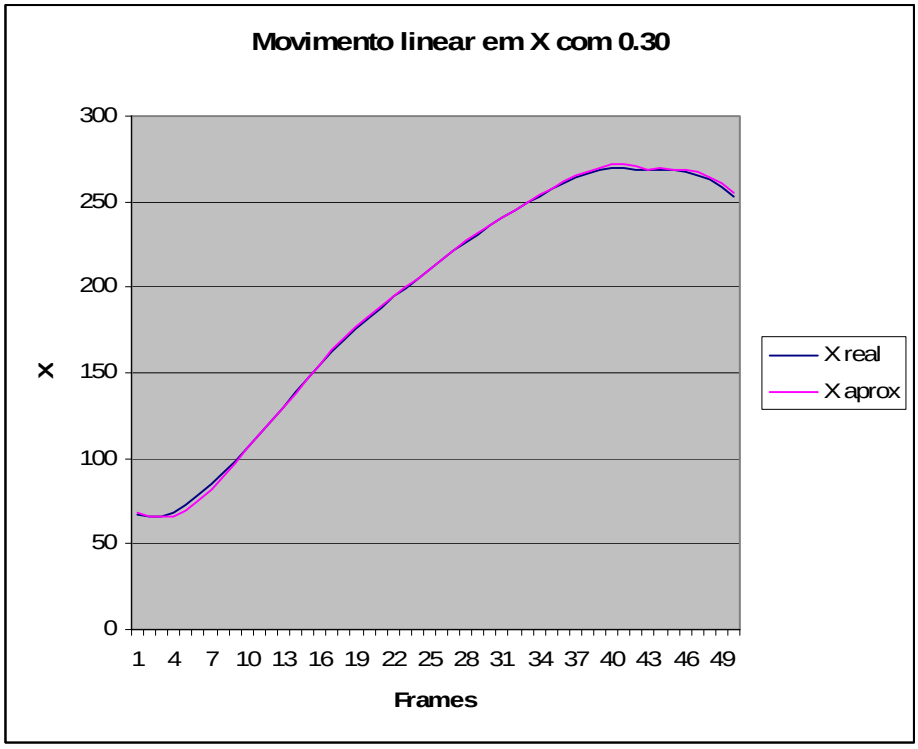

Figura 5.15

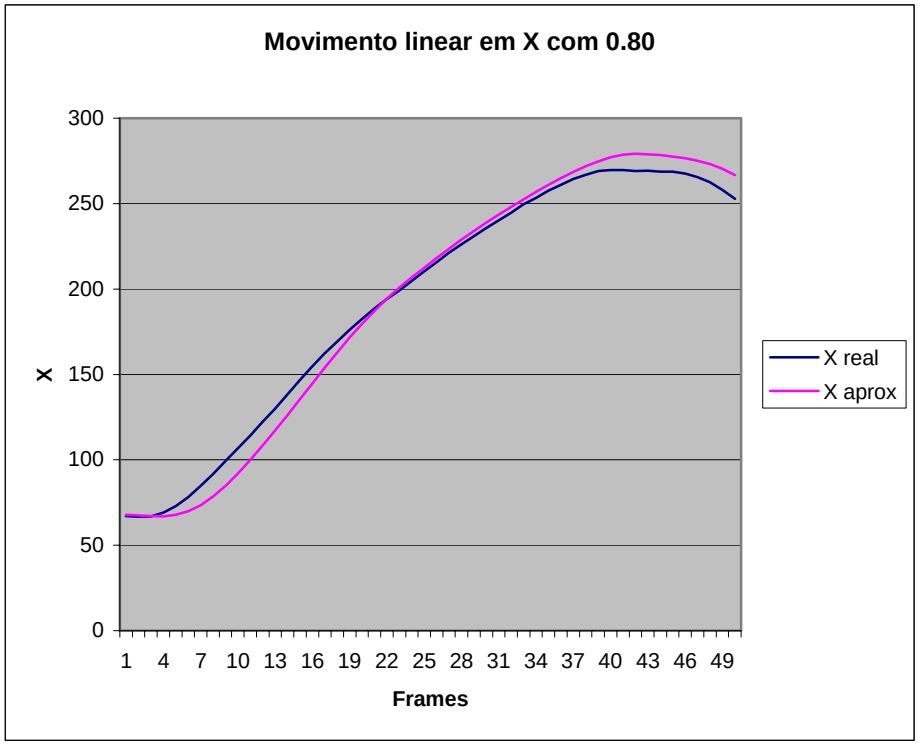

Figura 5.16

A comparação entre as figuras 5.15 e 5.16 mostra como o valor mais pequeno de θ e melhor para o seguimento do objecto num tracking dum objecto com movimento linear.

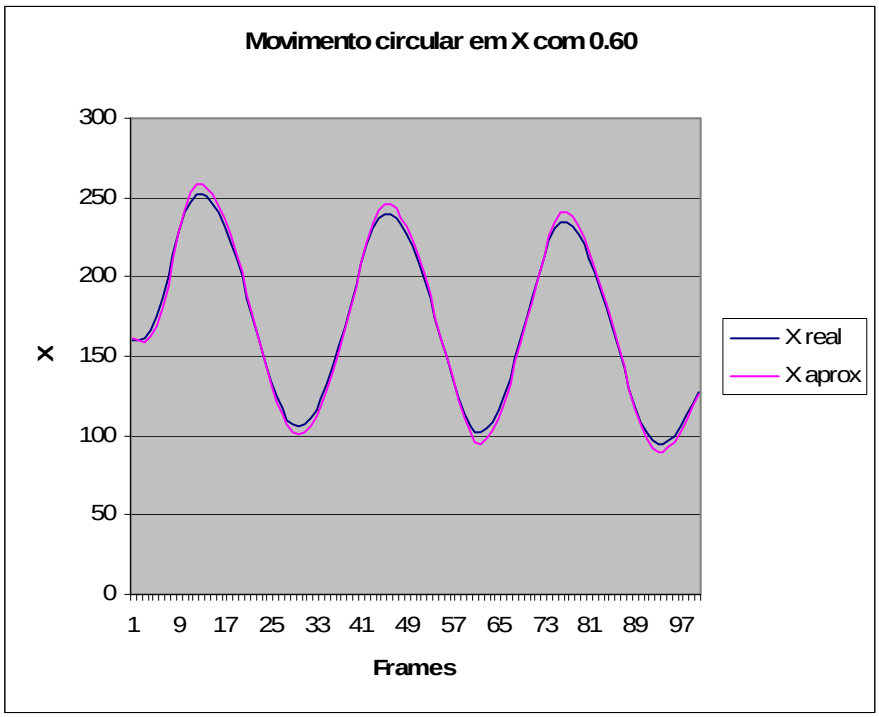

Figura 5.17

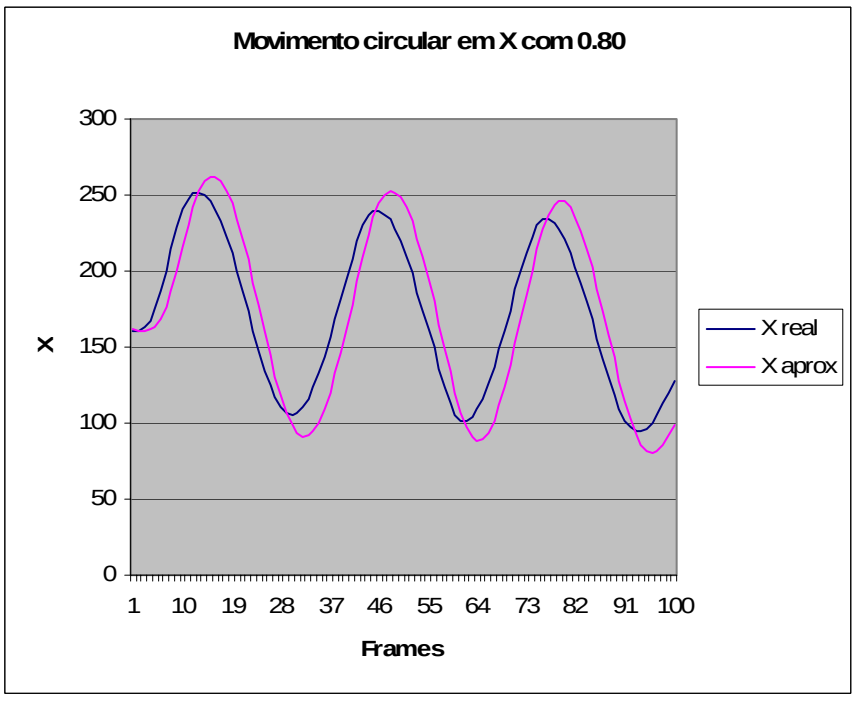

Figura 5.18

A comparação entre as figuras 5.17 e 5.18 mostra como o valor mais pequeno de θ e melhor para o seguimento do objecto num tracking dum objecto com movimento circular.

Como conclusão um valor pequeno (0.30) de θ obtém melhores resultados na predição do tracking.

5.7 Modelos criados para diferentes objectos (Treino).

Para realizar o treino de objectos o sistema guarda a media das duas variáveis de cor, assim como a matriz de covariância que permite o sistema o posterior reconhecimento do objecto.

Pode-se ver a continuação o modelo criado para um determinado objecto.

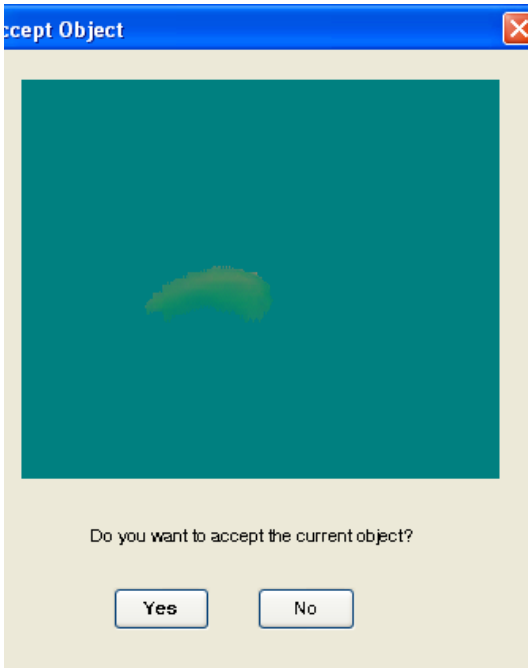

Figura 5.19

Para o objecto que se mostra na figura 5.19 o modelo guardado na representação HSV é o seguinte:

Id	Mean a	Mean b	Cov 11	Cov 21	Cov 12	Cov 22
65	0.662269	0.638537	0.000180	-0.000339	-0.000339	0.021838

Tabela 5.1

Para a representação LAB e guardada uma tabela igual a Tabela 5.1.

O “Id” da tabela 5.1 representa um identificador interno do sistema para o objecto treinado.

A “Mean a” e a “Mean b” representam as médias na primeira e segunda variável da representação da cor, HSV ou LAB.

“Cov” representa a matriz de covariância da representação da cor do objecto.

5.8 Reconhecimento de objectos treinados e tracking automático.

O reconhecimento automático de objectos treinados e o seu posterior seguimento foi efectuado para integrar a segmentação de cenas completas e a segmentação de objectos individuais.

Os resultados obtidos para o reconhecimento de objectos não são especialmente bons, mas servem como uma prova satisfatória do objectivo de integração dos dois módulos de segmentação da cor. Segmentação com inicialização e segmentação sem inicialização.

Na figura seguinte mostra-se como o objecto treinado é automaticamente reconhecido, e qual é o ratio que o objecto obteve para ser reconhecido. O objecto é posteriormente seguido mediante tracking.

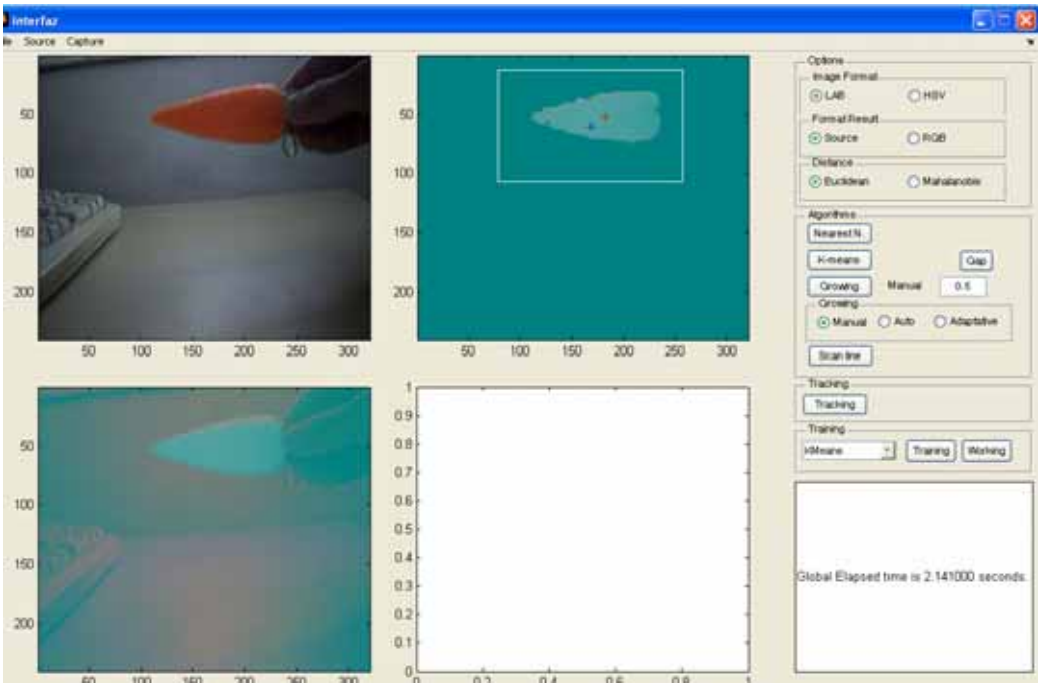

Figura 5.20.

Os ratios de coincidência dos objectos segmentados por k-means são os seguintes:

Matching: 0.000000

Matching: 0.981009

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.012963

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000

Matching: 0.000000
Matching: 0.000000
Matching: 0.000000
Matching: 0.000000
Matching: 0.000000
Matching: 0.000000
Matching: 0.000000

O valor de 0.981009 foi o valor seleccionado como mais provável elemento a seguir, e foi o correspondente à cenoura que aparece na figura 5.20.

O ratio é calculado pela percentagem de pontos do objecto candidato que estão perto do objecto treinado. A distância de Mahalanobis é a que determina este facto.

No exemplo anterior indica que o 98% pontos do objecto candidato estão muito perto dum objecto treinado.

5.9 Critério GAP na determinação do K no k-means.

Os seguintes resultados mostram como o k-means com selecção faz uma divisão automática da imagem e compara o resultado obtido com o k obtido por meio do Gap statistic.

Figura 5.22

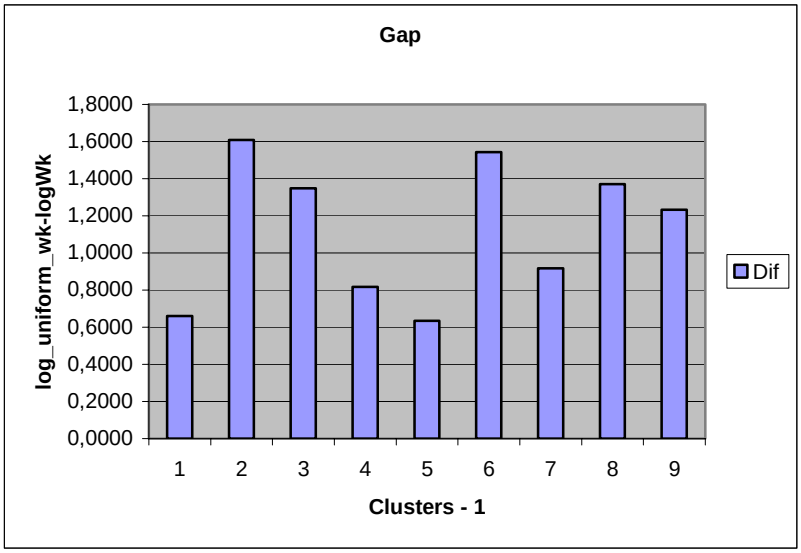

Figura 5.23

Para a imagem da figura 5.22 o resultado do k para o k-means automático é 3 que é o mesmo resultado que resulta do GAP statistic tal como se observa na figura 5.19 (coluna 2 e maior, que corresponde com divisão em 3 clusters).

Figura 5.24

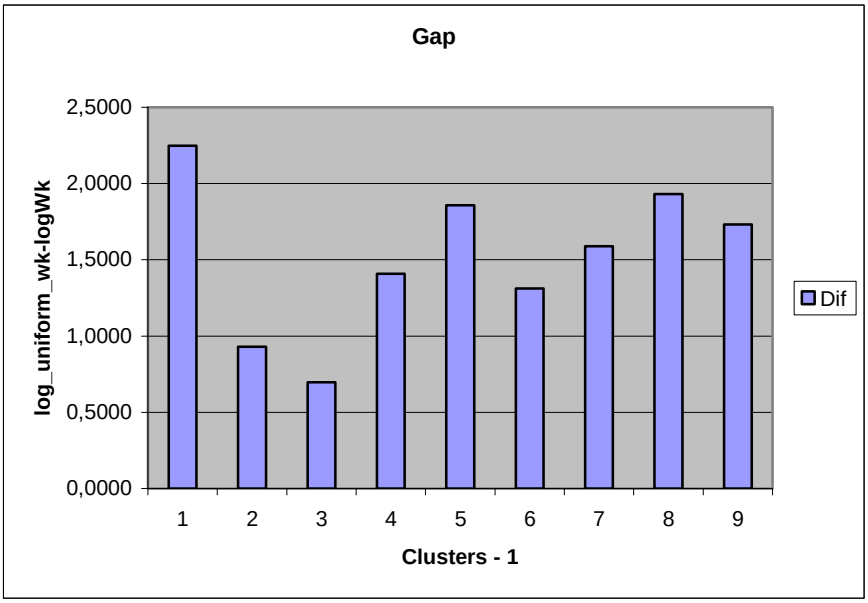

Figura 5.25

Para a imagem da figura 5.24 o resultado do k para o k-means automático é 3 que é diferente do resultado do GAP statistic que devolve 2 como número de clusters.

Figura 5.26

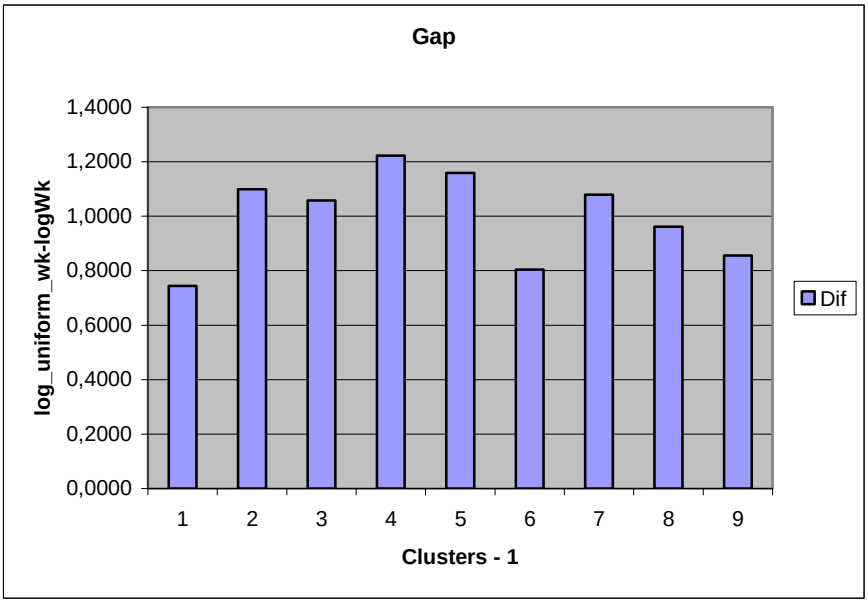

Figura 5.27

Para a imagem da figura 5.26 o resultado do k para o k-means automático é 4 que é diferente do resultado do GAP statistic que devolve 5 como número de clusters.

Estudando as diferencias entre a escolha automática do k no k-mean e a escolha do GAP statistic observa-se que o número de clusters não é muito diferente. Uma vantagem do k-means de escolha automática é que não tem que fazer a segmentação ate N clusters e depois fazer a escolha. Pelo tanto o custo é menor que aplicar o GAP statistic que apresenta também o problema de ter que definir um máximo de clusters para fazer a segmentação.

Experimentalmente os resultados como o k-means foram bons como se pode ver em 5.4.

6 Conclusões.

Foram estudadas diferentes técnicas de segmentação de cor aplicável ao seguimento de alvos e aprendizagem de tarefas em robots humanóides.

Foram implementados algoritmos de segmentação de cor para detectar objectos simples colocados no ambiente do robot. Os algoritmos implementados são robustos em consideração as condições de iluminação e a mudanças da pose.

A segmentação inicial da cena em zonas distintamente coloridas representando potenciais objectos foi implementada com o algoritmo k-means com o cálculo automático do número de clusters. O cálculo automático resolveu-se como válido na prática ainda que não coincide exactamente com o critério GAP.

O seguimento dos objectos detectados resolveu-se fazer com Scan-line que resultou no mais rápidos dos algoritmos testados.

A representação da cor que resultou mais útil foi a representação LAB.

A construção dum processo de treinamento e seguimento automático, assim como a criação da interface gráfica, integrou as técnicas estudadas e facilitou o cumprimento dos objectivos do presente trabalho.

Como trabalho a fazer:

- Melhorar a caracterização dos objectos treinados.
- Melhorar a velocidade nos tempos de reconhecimento de objectos.
- Melhorar a velocidade no seguimento de objectos.

Como conclusão final os objectivos principais do presente trabalho foram conseguidos.

Apêndice A. Descrição dos módulos principais.

Color_segmentation_nearest_neighbor.m

Implementado em Matlab.

Descrição: A imagem de entrada e dividida em duas imagens. O algoritmo utilizado e o Nearest-Neighbour.

O módulo pode trabalhar com as distâncias de Mahalanobis e Euclidiana.

Trabalha com formatos HSV e LAB.

Color_segmentation_k_means_clustering.m

Implementado em Matlab.

Descrição: A imagem de entrada e dividida em k imagens. O algoritmo utilizado é k-means, em que K é calculado automaticamente. O modo de calcular k é descrito no capítulo (4.2.1.2.). Três clusters são sempre gerados.

O módulo pode trabalhar com distância Euclidiana.

Trabalha com formatos HSV e LAB.

Region_growing_procedure.m

Implementado em Matlab.

Descrição: Da imagem de entrada é extraído um objecto. O algoritmo utilizado é Growing Region. Precisa como argumento de entrada uma região inicial do objecto a extrair. Outro argumento é o modo de trabalho. Manual, Automático ou Adaptativo. Um terceiro parâmetro é o limiar de distância que permite decidir se um pixel pertence a uma região ou não.

O módulo pode trabalhar com distância Mahalanobis.

Trabalha em três modos.

- Manual. O módulo calcula a distância de Mahalanobis dos pontos da vizinhança à distribuição de pontos da região inicial. Essa distância é

comparada com um limiar definido pelo utilizador para decidir se o ponto é incluído ou não na região.

- Automático. Existe um limiar que é relativo a distribuição de cor do objecto. Com esse limiar é calculada a distância para que os pontos de entrada na região sejam aceites ou não. Os novos pontos que entram na região **não contam** para o cálculo da distância de Mahalanobis para próximos pontos.
- Adaptativo. O limiar indica uma percentagem. Com esse limiar é calculada a distância para que os pontos de entrada na região sejam aceites ou não. Os novos pontos que entram na região **contam** para o cálculo da distância de Mahalanobis.

Para os modos automático e adaptativo o limiar é calculado por meio da função de MATLAB *norminv* que calcula a inversa da distribuição normal acumulativa.

Trabalha com formatos HSV e LAB.

Scan_line_procedure.m

Implementado em Matlab.

Descrição: Da imagem de entrada é extraído um objecto. O algoritmo utilizado é Scan-line. Precisa como argumento de uma região inicial pertencente ao objecto. Outro parâmetro é o limiar de distância que permite decidir se um pixel pertence a região ou não. Este limiar é calculado pela função de MATLAB *norminv* do mesmo modo que no caso do Growing Automático e o Growing Adaptativo.

Trabalha com formatos HSV e LAB.

Tracker_final.m

Implementado em Matlab.

Descrição: É o módulo encarregado de fazer tracking. Duma imagem de entrada e com os conhecimentos do instante anterior da velocidade e centro do objecto, devolve o objecto na imagem de entrada, o novo centróide e a nova velocidade. Precisa como parâmetro de entrada

também um limiar percentual para o cálculo da distância dos pontos da imagem de entrada à distribuição definida pelas características do objecto.

Apêndice B. Descrição dos módulos secundários.

Convert_image.m

Implementado em Matlab.

Descrição: Encapsula todas as conversões entre os diferentes tipos de imagens nos que trabalha o sistema. Os tipos são RGB, HSV e LAB.

Save_objects.m

Implementado em Matlab.

Descrição: Guarda as imagens e as características no armazenamento de dados dos objectos que são passados como parâmetro.

Read_accepted_objects.m

Implementado em Matlab.

Descrição: Lê os dados dos objectos treinados da armazenagem de dados.

Get_statistics.m

Implementado em Matlab.

Descrição: Calcula as estatísticas dos objectos à entrada, medias e covariância.

Write_statistics.m

Implementado em Matlab.

Descrição: Escreve as características obtidas por get_statistisc no arquivo “statistics.doc”.

Center_attention.m

Implementado em Matlab.

Descrição: Faz a escolha do objecto para fazer tracking.

Decide_objects.m

Implementado em Matlab.

Descrição: Faz a escolha dos objectos candidatos para fazer tracking. Os objectos muito pequenos ou que não são parecidos com nenhum objecto treinado, são descartados.

Get_biggest_object.m

Implementado em Matlab.

Descrição: Escolhe o objecto maior duma segmentação binária.

Get_objects_from_images.m

Implementado em Matlab.

Descrição: Interface para obter os diferentes objectos duma sequência de imagens.

Selecting_objects.m

Implementado em Matlab.

Descrição: Interface para obter os diferentes objectos duma imagem.

Get_region_from_image.m

Implementado em Matlab.

Descrição: Interface para obter a zona da imagem indicada por uma planilha.

Apêndice C. Melhora de velocidade em C.

Com o objectivo de incrementa a velocidade da aplicação, foram implementados alguns dos módulos do sistema em C. A continuação uma breve descrição do que foi implementado em C.

k_means_euclidean_c.c

Implementado em C.

Descrição: Realiza o clustering em k clusters com distância euclidiana. Incrementa a velocidade da segmentação com k-means.

Update_speed _c.c

Implementado em C.

Descrição: Faz a actualização da velocidade e do centróide do objecto durante o tracking ou o reconhecimento automático de objectos.

Tracker_c.c

Implementado em C.

Descrição: Implementação do algoritmo Scan-line.

Rgb2hsv_c.c

Implementado em C.

Descrição: Conversão entre representação da cor em RGB e HSV.

Get_region_characteristics_c.c

Implementado em C.

Descrição: Obtém as médias e a matriz de covariância inversa, também obtém a janela do objecto.

Referências.

[1] L.G.Shapiro e G. C. Stockman, *Computer Vision*, New Jersey: Prentice Hall, 2001.

[2] S. T,heodoridis e K. Koutroumbas, *Pattern Recognition*, Academic Press, 2003.

[3] L.T. Magalhães , *Álgebra linear como introdução a matemática aplicada,* Texto Editora, 1992.

[4] T. Hastie, R. Tibshirani e J. Friedman, *The elements of Statistical Learning: Data Mining, Inference and Prediction*, Springer, 2001.

[5] A. K. Jain, *Fundamentals of digital image processing*, Prentice Hall International Editions, 1989.