Universidade do Minho
Escola de Engenharia

Miguel João Alves Cunhal

Sistema de Visão para a Interação e Colaboração Humano-Robô: Reconhecimento de Objetos, Gestos e Expressões Faciais

novembro de 2014

Sistema de Visão para a Interação e Colaboração Humano-Robô:
Reconhecimento de Objetos, Gestos e Expressões Faciais
Miguel João Alves Cunhal
UMinho | 2014

Universidade do Minho
Escola de Engenharia

Miguel João Alves Cunhal

Sistema de Visão para a Interação e Colaboração Humano-Robô: Reconhecimento de Objetos, Gestos e Expressões Faciais

Dissertação de Mestrado
Ciclo de Estudos Integrados Conducentes ao Grau de
Mestre em Engenharia Eletrónica Industrial e Computadores

Trabalho efetuado sob a orientação da
Professora Doutora Estela Guerreiro da Silva Bicho Erlhagen

novembro de 2014

DECLARAÇÃO

Nome: Miguel João Alves Cunhal

Endereço eletrónico: a58753@alunos.uminho.pt **Telefone:** 916027765

Número do Bilhete de Identidade: 13929938

Título da dissertação: Sistema de Visão para a Interação e Colaboração Humano-Robô: Reconhecimento de Objetos, Gestos e Expressões Faciais

Orientador: Professora Doutora Estela Guerreiro da Silva Bicho Erlhagen

Ano de conclusão: 2014

Designação do Mestrado: Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores – Automação, Controlo e Robótica

Universidade do Minho, __/ __/ ____

Assinatura: _____

Agradecimentos

A todos os que ao longo deste percurso me acompanharam e contribuíram para o desenvolvimento do meu trabalho, gostaria de expressar o meu reconhecimento e gratidão.

À minha orientadora, Doutora Estela Bicho Erlhagen, pelos conhecimentos transmitidos, disponibilidade demonstrada, aconselhamento e apoio incondicional, fundamentais no desenrolar do trabalho.

Um agradecimento especial ao Rui Silva, pela orientação, paciência e disponibilidade absoluta que evidenciou ao longo do ano, essenciais no desenvolvimento do projeto.

Aos restantes colegas do Laboratório de Robótica Móvel e Antropomórfica do Departamento de Eletrónica Industrial da Universidade do Minho, pela forma como me acolheram e pelo ambiente de entreajuda gerado, tão importante e facilitador na realização do trabalho. Nomeadamente: Luís Louro, Tiago Malheiro, Toni Machado, Emanuel Sousa, Flora Ferreira, Carlos Faria, Gianpaolo Gulleta, Weronika Wojtak, Simão Antunes e Sara Araújo.

À minha família, e em especial aos meus pais, por sempre me apoiarem e se terem esforçado durante toda a minha vida para que eu tivesse a melhor educação possível e trabalhasse numa área que gostasse.

À Sara, pelo apoio incondicional, pela paciência e motivação prestada que me fez suportar e ultrapassar as diferentes etapas deste trabalho de uma maneira muito mais fácil.

A todos os meus amigos e em particular àqueles que me acompanharam de perto neste percurso académico.

A todos os restantes que, de uma maneira direta ou indireta, contribuíram e me apoiaram no desenrolar deste projeto.

Resumo

O objetivo deste projeto de dissertação consistiu no *design*, implementação e validação de um sistema de visão para aplicação no robô antropomórfico ARoS (*Anthropomorphic Robotic System*) no contexto da execução autónoma de tarefas de interação e colaboração com humanos. Foram exploradas três vertentes essenciais numa perspetiva de interação natural e eficiente entre robô e humano: o reconhecimento de objetos, gestos e expressões faciais.

O reconhecimento de objetos, pois o robô deve estar a par do ambiente que o rodeia de modo a poder interagir com o parceiro humano. Foi implementado um sistema de reconhecimento híbrido assente em duas abordagens distintas: caraterísticas globais e caraterísticas locais. Para a abordagem baseada em caraterísticas globais usaram-se os momentos invariantes de *Hu*. Para a abordagem baseada em caraterísticas locais exploraram-se vários métodos de deteção e descrição de caraterísticas locais, selecionando-se o SURF (*Speeded Up Robust Features*) para uma implementação final. O sistema devolve, também, a localização espacial dos objetos, recorrendo a um sistema de visão estereoscópico.

O reconhecimento de gestos, na medida em que estes podem fornecer informação acerca das intenções do humano, podendo o robô agir em concordância após a interpretação dos mesmos. Para a deteção da mão recorreu-se à deteção por cor e para a extração de caraterísticas da mesma recorreu-se aos momentos invariantes de *Hu*. A classificação dos gestos é feita através da verificação dos momentos invariantes de *Hu* complementada por uma análise da segmentação resultante da

Convex Hull.

Por último, o reconhecimento de expressões faciais, visto que estas podem indicar o estado emocional do humano. Tal como em relação aos gestos, o reconhecimento de expressões faciais e consequente aferição do estado emocional permite ao robô agir em concordância, podendo mesmo alterar o rumo da ação que vinha a efetuar. Foi desenvolvido um *software* de análise de robustez para avaliar o sistema previamente criado (*FaceCoder*) e, com base nesses resultados, foram introduzidas algumas alterações relevantes no sistema *FaceCoder*.

Palavras-chave: reconhecimento de objetos, reconhecimento de gestos, reconhecimento de expressões faciais, visão estereoscópica, SURF, momentos invariantes de *Hu*, *Action Units*

Abstract

The objective of this dissertation consisted on the design, implementation and validation of a vision system to be applied on the anthropomorphic robot ARoS (Anthropomorphic Robotic System) in order to allow the autonomous execution of interaction and cooperation tasks with human partners. Three essential aspects for the efficient and natural interaction between robot and human were explored: object, gesture and facial expression recognition.

Object recognition, because the robot should be aware of the environment surrounding it so it can interact with the objects in the scene and with the human partner. An hybrid recognition system was constructed based on two different approaches: global features and local features. For the approach based on global features, Hu's moment invariants were used to classify the object. For the approach based on local features, several methods of local features detection and description were explored and for the final implementation, SURF (Speeded Up Robust Features) was the selected one. The system also returns the object's spatial location through a stereo vision system.

Gesture recognition, because gestures can provide information about the human's intentions, so that the robot can act according to them. For hand's detection, color detection was used, and for feature extraction, Hu's moment invariants were used. Classification is performed through Hu's moment invariants verification alongside with the analysis of the Convex Hull segmentation's result.

Last, facial expression recognition because it can indicate the human's emotional

state. Like for the gestures, facial expression recognition and consequent emotional state classification allows the robot to act accordingly, so it may even change the course of the task it was taking on. It was developed a robustness analysis software to evaluate the previously created system (*FaceCoder*) and, based on the results of the analysis, some relevant changes were added to *FaceCoder*.

Keywords: object recognition, gesture recognition, facial expression recognition, stereo vision, SURF, Hu's moment invariants, Action Units

Siglas e Abreviaturas

ARoS	*Anthropomorphic Robotic System*
ART	*Angular Radial Transform*
AU	*Action Unit*
AUs	*Action Units*
BRIEF	*Binary Robust Independent Elementary Features*
BRISK	*Binary Robust Independent Scalable Keypoints*
C/C++	Linguagem de programação
C#	Linguagem de programação
CK+	*Extended Cohn-Kanade*
EMFACS	*Emotional Facial Action Coding System*
faceAPI	*Face Application Programming Interface*
FACS	*Facial Action Coding System*
FACSAID	*Facial Action Coding System Affect Interpretation Dictionary*
FAST	*Features from Accelerated Segment Test*
fps	*frames* por segundo

FREAK	*Fast Retina Keypoints*
HSV	*Hue, Saturation, Value*
i.e.	isto é
k-NN	*k-Nearest Neighbors*
NASA	*National Aeronautics and Space Administration*
OpenCV	*Open Source Computer Vision Library*
ORB	*Oriented FAST and Rotated BRIEF*
RANSAC	*Random Sample Consensus*
RGB	*Red, Green, Blue*
ROI	*Region of Interest*
SIFT	*Scale Invariant Features Transform*
SURF	*Speeded Up Robust Features*

Conteúdo

Lista de Figuras

Esta página foi intencionalmente deixada em branco!

Lista de Tabelas

Parte I

Enquadramento

Capítulo 1

Introdução

Cada vez mais, nos tempos que correm, se utilizam robôs em tarefas de interação e colaboração com humanos. Exemplos disso são os robôs que auxiliam em tarefas de assistência aos humanos como a série *Care-O-bot*® (atualmente na 3ª geração), ou então, num nível mais sofisticado, o *Robonaut* da NASA, para utilização em trabalhos de construção nas estações espaciais. Porém, estes robôs não passam de automatismos pré-programados para fazer trabalhos específicos ou então são telecomandados. Nestes casos não se pode considerar que o robô é um assistente (socialmente) inteligente visto que o mesmo não tem capacidade de iniciativa nem poder de decisão e não leva em consideração as decisões do parceiro humano com quem está a interagir.

Têm havido, contudo, alguns progressos no sentido de dotar os robôs com ferramentas que permitam um melhoramento nas relações humano-robô. Um exemplo disso é o *Brian 2.1* (McColl et al., 2013). Este robô foi projetado para prestar assistência a pessoas idosas que sofrem de doenças cognitivas. Consegue auxiliar a pessoa em atividades elementares tais como comer e vestir, assim como interage com a pessoa em atividades de estimulação cognitiva e social. O robô determina o comportamento apropriado a adotar com base no estado da atividade e da pessoa com quem está a lidar. Por outro lado, o sistema *GiraffPlus* (Coradeschi

et al., 2013), projetado no intuito de fazer acompanhamento a pessoas idosas, e em particular àquelas que vivem sós, recorre a um conjunto de sensores distribuídos pela casa (desde sensores de movimento, sensores de luminosidade, entre outros), através dos quais o sistema atua autonomamente consoante os diferentes contextos inferidos. O projeto *GiraffPlus* contempla também um robô de tele-presença, equipado com um sistema de videoconferência, que permite uma fácil comunicação com um familiar, amigo ou até mesmo um médico. Em termos de robôs na indústria, referência ao *Baxter* (Fitzgerald, 2013), que se destaca pela facilidade com que "aprende" as tarefas a desempenhar e pela segurança que lhe é imputada, podendo trabalhar com humanos ao contrário da maioria dos robôs industriais. Para "ensinar" o robô basta pegar no manipulador e executar os movimentos pretendidos no modo de aprendizagem (Amadeo, 2014).

Apesar destes avanços no relacionamento entre humanos e robôs em tarefas de interação e cooperação, a extração e processamento de informação sensorial continua a ser um dos maiores desafios neste processo. Isto porque toda a atividade que o robô possa desenvolver está intrinsecamente relacionada com a perceção que o mesmo tem do ambiente que o rodeia.

1.1 Motivação e objetivos

A utilização de robôs como parceiros já é uma realidade tanto na robótica de serviços (razões sociais, como o caso da assistência a idosos) como na robótica industrial (razões económicas). Porém, dadas as potencialidades desta área, vários esforços têm sido feitos em termos de investigação no sentido de melhorar a autonomia dos robôs, possibilitando a maior independência possível de um operador humano. Ora, tal como os humanos recorrem aos sentidos para se orientarem no mundo e poderem realizar as mais variadas tarefas do dia-a-dia, também os robôs necessitam de ser providos de algum tipo de mecanismo que simule esses mesmos sentidos.

E importa referir dois em particular: a audição e a visão. São estes os sentidos fundamentais no contexto da interação social. A simulação da audição efetuada através de técnicas de reconhecimento de voz e a simulação da visão através de técnicas de processamento de imagem (visão por computador).

Motivado pela crescente utilização de robôs como parceiros em tarefas de cooperação humano-robô, este trabalho foca o sistema de visão e, em particular, três temáticas bastante pertinentes no contexto da autonomia de robôs nas referidas tarefas: *i)* o reconhecimento e localização espacial de diferentes objetos, *ii)* o reconhecimento de gestos e *iii)* o reconhecimento de expressões faciais de um parceiro humano, a partir dos quais (*ii* e *iii*) se podem inferir intenções e/ou estados emocionais. Estas capacidades são requisitos essenciais para que a relação entre os intervenientes (i.e. humano e robô) possa ser mais eficiente e minimamente *human-like*. Ou seja, o robô deve estar a par do ambiente que o rodeia e para isso necessita de reconhecer os objetos que irá usar de modo a poder interagir com os mesmos. Deve também estar "atento" ao parceiro humano de modo a poder antecipar as suas intenções (que estão por detrás dos gestos) e interpretar os seus estados emocionais (revelados pelas expressões faciais), alterando o comportamento a adotar no caso de constatar reações do humano que evidenciem esta necessidade.

Como objetivos pretende-se desenvolver um sistema de visão que dote o robô ARoS (Silva, 2008) (ver Figura 1.1) com as capacidades acima referidas. Ou seja, o reconhecimento de objetos e a respetiva localização no espaço em tempo real, permitindo uma interação e colaboração eficiente entre humano e robô. Paralelamente, o robô deve fazer uma análise das expressões faciais e interpretação de gestos de um parceiro humano, para poder, posteriormente, adotar o tipo de comportamentos que vão de encontro às necessidades do humano.

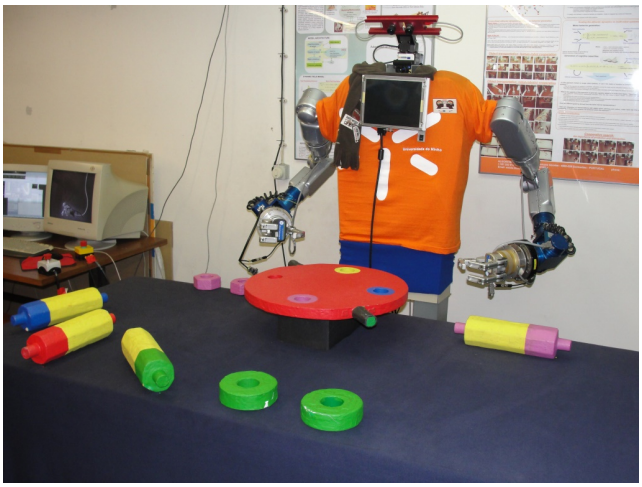


Figura 1.1: Robô antropomórfico ARoS.

1.2 Contribuições da dissertação

A integração de sistemas de visão por computador em robôs de assistência aos humanos não é novidade. A título de exemplo, o *Care-O-bot® 3* (Reiser et al., 2009) utiliza um sistema que realiza o mapeamento 3D através da combinação da informação de cor com a de profundidade. Na fase de treino é efetuada a segmentação e a extração de caraterísticas. São então calculados os descritores que, posteriormente, servirão como referência para a identificação do objeto aprendido. Também o *PR2* (Bohren et al., 2011) utiliza um sistema semelhante, principalmente no algoritmo usado para extração de caraterísticas. Já em relação ao *Brian 2.1*, o sistema de visão implementado é bastante limitado na medida em que o robô recorre a outro tipo de sensores como infravermelhos e células de carga no processo de deteção (McColl and Nejat, 2013). Neste caso, o sistema de visão (recorrendo às imagens provenientes de uma *Kinect*) apenas faz deteção e localização de caraterísticas faciais e determina a orientação da face do humano de modo a poder perceber o estado de atenção do mesmo (por exemplo, se estiver com a cara virada para o lado, i.e., ângulo superior a 45°, sujeito é dado como "distraído"). Num nível mais sofisticado, referência ao robô de companhia *Pepper*, da empresa francesa

Aldebaran (a ser comercializado pela *SoftBank*), que faz a identificação do estado emocional do parceiro humano através da análise de expressões faciais, gestos e tons de voz. O robô apenas possui a componente comunicativa, i.e., não executa tarefas. Age em concordância com o estado emocional identificado (se "sentir" que o humano está triste, põe uma das músicas favoritas deste último a tocar, por exemplo) (Guizzo, 2014).

Com o sistema de visão implementado no ARoS pretende-se reunir algumas das capacidades que os robôs mencionados já possibilitam, integrando as três vertentes - reconhecimento de objetos, gestos e expressões faciais de um utilizador humano - num único robô antropomórfico, permitindo, deste modo, um maior potencial em termos de aplicações nas camadas superiores da arquitetura cognitiva do sistema. Sistema esse que se pretende que seja autónomo, respondendo adequadamente às diferentes necessidades do parceiro humano sem a "ajuda" de um terceiro interveniente (operador).

Figura 1.2: ARoS numa tarefa de interação e colaboração com um humano.

1.3 Organização da dissertação

A dissertação encontra-se dividida em três partes. Na primeira parte é feito um enquadramento do projeto de dissertação através da introdução e da abordagem ao sistema de visão do ARoS. Na segunda parte são apresentados três capítulos que tratam cada uma das três temáticas distintas - objetos, gestos e expressões faciais - em termos do estado da arte, implementação e respetivos resultados. Na terceira e última parte constam alguns dos resultados da integração do sistema de reconhecimento de objetos com o sistema de reconhecimento gestual e as conclusões e trabalho futuro.

Capítulo 2

Sistema de Visão do ARoS

Neste capítulo é feita uma abordagem ao sistema de visão do ARoS. São expostas algumas noções sobre estereoscopia e a sua importância no contexto da localização espacial de objetos, assim como as especificações técnicas do sistema da visão. No que concerne ao *software*, é apresentada a biblioteca *OpenCV* e os *softwares faceAPI* e *FaceCoder* .

2.1 Noções sobre estereoscopia

Tal como os humanos usam os olhos, também os robôs necessitam de ser providos de um sistema que lhes permita orientarem-se no mundo, podendo, deste modo, executar as diferentes tarefas para as quais foram concebidos. Sendo assim, uma câmara seria o suficiente para reconhecer objetos e obter a sua posição bidimensional e orientação. Porém, tendo em conta que o robô poderá manipular o objeto numa fase posterior, importa que esta localização seja feita no espaço tridimensional. Ou seja, o robô deve conseguir "ver" em profundidade, usando, para tal, um sistema de visão estereoscópico.

A estereoscopia consiste no processo de obtenção de informação espacial (3D) tendo por base duas imagens distintas. Isto é, as diferenças entre as imagens do

olho direito e do olho esquerdo (separados, em média, por 65 milímetros) são processadas pelo cérebro criando uma ilusão de profundidade (Siscoutto et al., 2004).

Um sistema estereoscópico artificial baseia-se no mesmo princípio, utilizando duas câmaras separadas por uma determinada distância de modo a criar o efeito de disparidade (geração de duas imagens diferentes).

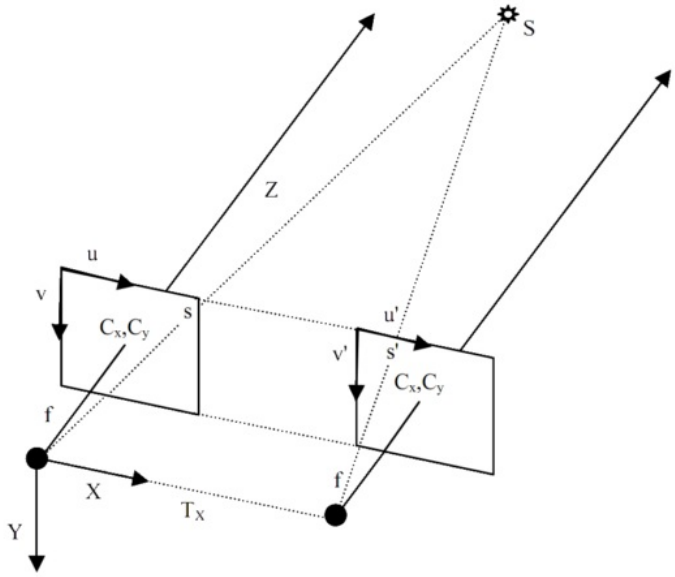
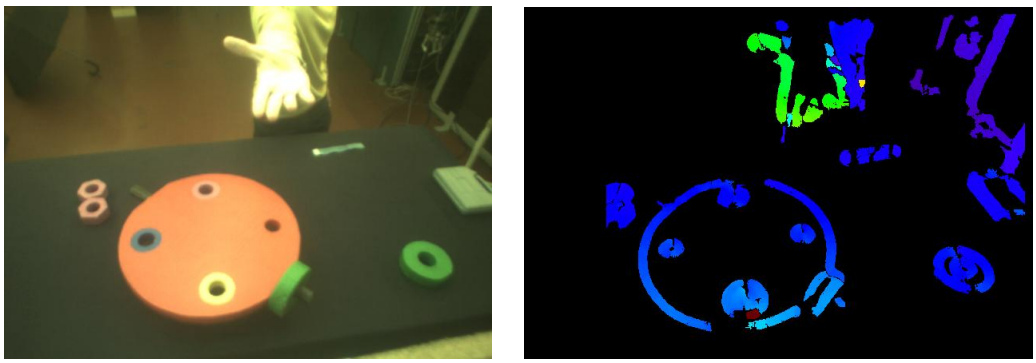


Figura 2.1: Geometria do sistema estereoscópico (imagem retirada de Konolige and Beymer (2007)).

Como se verifica pela Figura 2.1, o ponto S é projetado nas duas imagens intersetando nos pontos s e s' da imagem esquerda e direita, respetivamente. Estes pontos têm o mesmo valor nas coordenadas correspondentes ao eixo vertical (v e v') mas diferem em relação ao eixo horizontal (u e u'). Esta diferença denomina-se de disparidade e resulta do facto do ponto S estar a diferentes distâncias dos pontos de focagem f de cada uma das câmaras.

Para realizar os cálculos associados à geometria explicada, em primeiro lugar é necessário encontrar as correspondências entre a imagem da esquerda (de referência)

e a da direita. Para tal, é utilizado o método de correlação de área, que faz comparação de pequenas áreas das imagens através de correlação: a imagem esquerda é dividida em regiões e é feita uma procura, para cada uma dessas regiões, das respetivas correspondências dentro de uma janela de dimensão variável na imagem direita. Após a correlação é possível gerar a imagem de disparidade (ver Figura 2.2b) que quantifica, para cada ponto, a diferença entre as duas imagens (quanto maior o valor de disparidade de um determinado ponto, mais longe se encontra esse ponto da câmara) (Silva, 2008).



(a) Imagem original (câmara esquerda do robô).

(b) Imagem de disparidade.

Figura 2.2: Imagem de disparidade: cores "frias" (azuis) representam pontos mais afastados do sistema.

2.2 *Hardware* utilizado

O sistema de visão do ARoS é composto por três câmaras (ver Figura 2.3). Duas delas para o processamento estereoscópico. A terceira, utilizada na análise de expressões faciais.

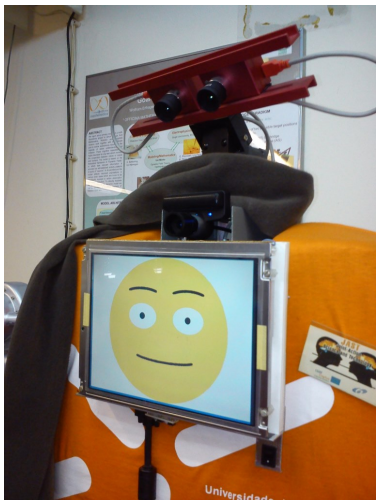


Figura 2.3: Sistema de visão constituído por uma estrutura de alumínio com duas câmaras para o processamento estereoscópico e uma câmara direcionada à face do parceiro humano para a análise de expressões faciais.

2.2.1 Sistema de visão estereoscópico

São apresentadas na Tabela 2.1 algumas das especificações do *hardware* utilizado no sistema de visão estereoscópico (comercializado pela *Videre Design*).

Figura 2.4: Sistema de visão estereoscópico.

Modelo	Resolução	Interface	Distância focal	Estrutura
STH-DCSG-VAR/-C	640x480 *pixéis*	*Firewire*	2,8 *mm*	Alumínio

Tabela 2.1: Especificações técnicas do sistema de visão estereoscópico.

Cálculos para a determinação da distância focal e escolha do tipo de lentes a usar podem ser consultados em Silva (2008).

Câmaras possuem *Global Shutter*, i.e., num determinado intervalo de tempo todos os *pixéis* da imagem capturada são expostos simultaneamente, permitindo uma maior estabilidade e reduzindo o efeito de arrastamento no caso de sequências de *frames* com objetos em movimento. De salientar, também, que estas câmaras se encontram perfeitamente sincronizadas de modo a não afetar os cálculos. Nas etapas de processamento bidimensional (deteção, reconhecimento, entre outras) apenas é usada a câmara da esquerda (Silva, 2008).

2.2.2 Câmara para análise de expressões faciais

Para a análise de expressões faciais é utilizada uma outra câmara (ver Figura 2.5), especificamente direcionada à face do utilizador humano. São apresentadas na Tabela 2.2 algumas das especificações técnicas da câmara. De referir que a lente foi alterada, permitindo uma distância focal variável.

Figura 2.5: Câmara usada no reconhecimento de expressões faciais (imagem retirada de `http://en.wikipedia.org/wiki/PlayStation_Eye` [acedido em 2014-09-16]).

Modelo	Resolução	Interface	Distância focal
PS3 Eye	640x480 *pixéis*	USB	2,8 - 12 *mm*

Tabela 2.2: Especificações técnicas da câmara usada no reconhecimento de expressões faciais.

2.3 *Software* utilizado

Com a vulgarização dos dispositivos eletrónicos e o aumento de capacidade dos mesmos, os programas de visão por computador começaram a fazer parte do dia-a-dia das pessoas, desde as aplicações mais complexas, àquelas mais simplistas e disponíveis gratuitamente para *download*. Este crescimento também se deveu ao desenvolvimento de bibliotecas como o *OpenCV* (*Open Source Computer Vision Library*), que permitem uma abstração das camadas inferiores, possibilitando uma concentração nos problemas de alto nível e, em consequência, a construção de aplicações mais complexas.

2.3.1 *OpenCV*

OpenCV é uma biblioteca de visão por computador de acesso livre. Desenvolvida inicialmente pela *Intel*, surgiu com a necessidade da existência de uma plataforma uniforme para aplicações de visão por computador. Contém mais de 2500 algoritmos otimizados. Estima-se que utilizem esta biblioteca mais de 47 mil pessoas, seja em empresas, grupos de investigação ou mesmo entusiastas da programação. Grandes empresas como a *Google*, *Microsoft*, *Intel*, *IBM*, *Honda*, entre outras, recorrem a esta biblioteca na implementação das suas soluções. Possui interfaces para C/C++, *Python*, *Java* e *MATLAB*. A biblioteca é usada em inúmeras aplicações: sistemas de interface humano-computador ou mesmo humano-robô (através de aplicações de reconhecimento de objetos, faces, gestos, movimentos, entre outros), robótica industrial (através de sistemas de localização de peças, por exemplo), sistemas de realidade aumentada (como o estacionamento visualmente assistido), realidade virtual, entre outros[1].

[1] `http://www.opencv.org` [acedido em 2014-05-07].

2.3.2 *faceAPI*

faceAPI é um *software* desenvolvido pela *Seeing Machines* especializado no processamento de imagem de caraterísticas faciais: permite a localização espacial da face, a deteção e marcação de pontos essenciais (cantos dos lábios, por exemplo), o rastreamento da posição dos lábios e sobrancelhas (úteis na classificação de expressões faciais), a extração de texturas, entre outros. Entre as principais aplicações destacam-se os jogos interativos 3D, os programas de interação humano-robô e humano-computador, como é o caso dos Sistemas Avançados de Assistência ao Condutor (ADAS - *Advanced Driver Assistance Systems*), os *displays* 3D e os sistemas de videoconferência "inteligentes"[2].

2.3.3 *FaceCoder*

FaceCoder é um *software* desenvolvido no Laboratório de Robótica Móvel e Antropomórfica da Universidade do Minho para a análise de expressões faciais em tempo real. O *FaceCoder* usa o *faceAPI* nas fases de deteção e extração de caraterísticas faciais para posterior processamento.

[2] `http://www.faceapi.com` [acedido em 2014-05-20].

Esta página foi intencionalmente deixada em branco!

Parte II

Fundamentos Teóricos, Implementação e Resultados

Capítulo 3

Reconhecimento e Localização Espacial de Objetos

Um requisito elementar para o reconhecimento de algo consiste no conhecimento prévio do mesmo. Isto é, para dotar o robô com a capacidade de identificação de objetos é necessário dar ao mesmo mecanismos que o permitam conhecer de antemão os mesmos. Como tal, o sistema deve possuir uma base de dados com informação relativa aos objetos para posterior identificação/localização. Ora, na fase de reconhecimento propriamente dita, outras questões devem ser tomadas em consideração. Em primeiro, considerando que se trata de um ambiente de trabalho, diferentes objetos poderão estar no plano de trabalho. Alguns dos quais podem até "nem interessar" ao robô, na medida em que não estão na base de dados do sistema e a sua identificação não é relevante. Como isolar os objetos e aferir quais interessam e os que podem ser descartados?

Um dos grandes problemas dos sistemas deste tipo consiste na robustez do algoritmo de reconhecimento. E este é talvez o problema chave visto que o objeto deve ser reconhecido independentemente das transformações a que for sujeito (translações, rotações e escalamentos, as principais). Isto porque criar uma base de dados que contemplasse todas as orientações e escalamentos de um determinado

objeto, tornaria o programa de tal modo pesado que seria impensável utilizá-lo numa aplicação que se pretende que funcione em tempo real. Como tal, devem ser explorados algoritmos de extração de caraterísticas invariantes a transformações da imagem.

3.1 Estado da arte

Dentro dos métodos de reconhecimento (ver Figura 3.1) podemos distingui-los em duas categorias principais[1]: métodos baseados em caraterísticas locais e métodos baseados em caraterísticas globais (métodos holísticos). Em relação aos últimos podemos ainda destacar duas tipologias mais comuns: abordagens baseadas na aparência e abordagens baseadas na forma (abordagens geométricas) (Tuytelaars and Mikolajczyk, 2007; Ruberto and Morgera, 2008; Hsu et al., 2012).

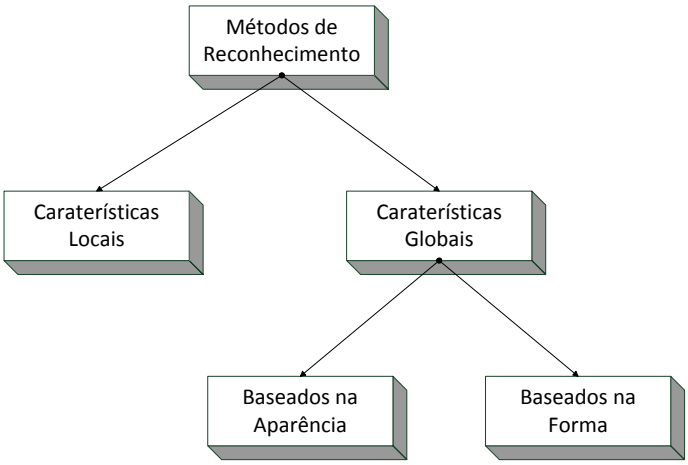


Figura 3.1: Abordagens dos métodos de reconhecimento.

Os métodos baseados em caraterísticas locais procuram saliências da imagem (cantos, beiras, etc). Estas regiões são caraterizadas por descritores que posteriormente são comparados com os descritores do objeto original (aquele que se pretende

[1]Considerados apenas métodos bidimensionais.

identificar) na tentativa de gerar correspondências (Tuytelaars and Mikolajczyk, 2007; Roth and Winter, 2008).

Os métodos baseados na aparência comparam o objeto a identificar com conjuntos de imagens presentes numa base de dados. Cada um desses conjuntos contém imagens de um determinado objeto sujeito a diferentes condições de iluminação e pontos de vista. Usando ferramentas de extração como PCA (*Principal Component Analysis*) (Pearson, 1901; Hotelling, 1933), DCT (*Discrete Cosine Transform*) (Strang, 1999) ou *Euler Vector* (Bishnu et al., 2005), seguidos por um método de classificação como SVM (*Support Vector Machines*) (Cortes and Vapnik, 1995), é possível encontrar o conjunto de imagens que mais se assemelha ao objeto a identificar (Tao, 2006). Por outro lado, os métodos baseados na forma descrevem caraterísticas geométricas como a área e o centroide (Ruberto and Morgera, 2008).

Os métodos baseados na aparência proporcionam uma maior gama de possibilidades em relação aos métodos baseados na forma (Tuytelaars and Mikolajczyk, 2007). Por exemplo, dois objetos diferentes com a mesma forma são classificados da mesma maneira segundo um método baseado numa abordagem geométrica. Já uma abordagem baseada na aparência consegue distinguir dois objetos diferentes com formas semelhantes. Porém, os métodos baseados na aparência exigem maior capacidade de processamento (de uma maneira significativa).

Os métodos holísticos apresentam bastantes limitações. Ou seja, ao considerarem a imagem como um todo, necessitam de uma segmentação precisa visto que não distinguem diferentes planos da imagem (podendo mesmo misturar informação que se encontre em primeiro plano com o fundo da imagem). Por outro lado, a sua eficácia está dependente do grau de visibilidade do objeto em questão. Isto é, caso o mesmo esteja mal segmentado ou então tapado por outro objeto, o sistema deve falhar no processo de identificação (Tuytelaars and Mikolajczyk, 2007). Nesse sentido, cada vez mais se têm concentrado esforços no desenvolvimento de métodos de reconhecimento baseados em caraterísticas locais. Dado que estes

últimos também exigem capacidades de processamento potencialmente limitadoras para uma aplicação de funcionamento em tempo real, a solução ideal consiste na implementação de um sistema híbrido. Isto é, um sistema baseado em duas abordagens, uma mais eficiente que permita distinguir objetos com formas semelhantes e outra que requeira um menor processamento, diminuindo o tempo de resposta. Tendo estes pressupostos em consideração, apresentam-se, de seguida, algumas das abordagens mais usadas em termos de métodos baseados em caraterísticas globais (neste caso apenas são estudados os métodos baseados na forma, visto que são aqueles que se ajustam melhor aos propósitos do projeto) e em caraterísticas locais.

3.1.1 Métodos baseados na forma

Os métodos baseados na forma descrevem a extensão da imagem binária do objeto. As representações de forma podem ser baseadas na região ou no contorno do objeto. São baseadas na região quando se trata de uma representação binária em que *píxeis* "ativos", normalmente brancos, correspondem ao objeto e a restante área é preenchida com *pixéis* pretos (momentos da imagem, *Angular Radial Transform*, entre outros). Já as representações baseadas no contorno apenas consideram a linha que delimita o objeto (descritores de *Fourier*). Estas têm a vantagem de descrever a forma do objeto de uma maneira mais precisa. Por outro lado, é mais fácil aplicar algoritmos de correspondências em representações baseadas na região (Amanatiadis et al., 2011; Sharma and Dhole, 2013).

Apresentam-se, seguidamente, alguns dos métodos de reconhecimento de forma mais usados na identificação de objetos.

3.1.1.1 Descritores de *Fourier*

Os descritores de *Fourier* são bastante usados em aplicações de representação de forma, especialmente no reconhecimento de carateres. São robustos a contaminações de ruído e simples de normalizar (Folkers and Samet, 2002; Amanatiadis

et al., 2011).

Obtêm-se através da aplicação da transformada de *Fourier* num vetor complexo. Esse vetor ($\overline{U}$) resulta da diferença entre os pontos da linha que delimita a forma (x_n, y_n) e o centroide (x_c, y_c) da mesma:

$$\overline{U} = \begin{pmatrix} x_0 - x_c + i(y_0 - y_c) \\ x_1 - x_c + i(y_1 - y_c) \\ \vdots \\ x_n - x_c + i(y_n - y_c) \end{pmatrix}, n = 0, 1, ...N - 1 \tag{3.1}$$

$$x_c = \frac{1}{N} \sum_{n=0}^{N-1} x(n) \tag{3.2}$$

$$y_c = \frac{1}{N} \sum_{n=0}^{N-1} y(n) \tag{3.3}$$

A subtração do centroide permite tornar a representação invariante a translações. Após o cálculo do vetor complexo, é então aplicada a transformada de *Fourier*:

$$\overline{F}_k = FFT[\overline{U}] \tag{3.4}$$

As magnitudes dos coeficientes $\overline{F}_k$ são normalizadas pela magnitude do coeficiente $\overline{F}_0$ de modo a tornar a representação invariante a escalamentos. Os descritores calculados são também invariantes a rotações.

3.1.1.2 ART - *Angular Radial Transform*

Baseada nos momentos da imagem (ver secção 3.1.1.3) e adotada em MPEG-7[2], a ART fornece uma boa descrição da distribuição dos *pixéis* numa região bidimensional

[2] *standard* de descrição de conteúdos multimédia.

(Ricard et al., 2005; Amanatiadis et al., 2011). Os seus coeficientes F_{nm} de ordem n e m são definidos por:

$$F_{nm} = \int_0^{2\pi} \int_0^1 V_{nm}(\rho, \theta) f(\rho, \theta) \rho d\rho d\theta \tag{3.5}$$

$f(\rho, \theta)$ é a função da imagem em coordenadas polares e $V_{nm}(\rho, \theta)$ é a função de base ART. A última divide-se em duas componentes: direção angular ($A_m(\theta)$) e direção radial ($R_n(\rho)$).

$$V_{nm}(\rho, \theta) = A_m(\theta) R_n(\rho) \tag{3.6}$$

$$A_m(\theta) = \frac{1}{2\pi} e^{jm\theta} \tag{3.7}$$

$$R_n(\rho) = \begin{cases} 1 & n = 0 \\ 2\cos(\pi n \rho) & n \neq 0 \end{cases} \tag{3.8}$$

O descritor ART é definido pelo conjunto de magnitudes normalizadas dos coeficientes ART. Através desse conjunto de magnitudes, obtém-se a invariância rotacional. Em MPEG-7, são usadas 12 funções angulares e 3 radiais ($n < 3, m < 12$). Para obter invariância a escalamentos, os coeficientes ART são divididos pela magnitude do coeficiente de ordem $n = 0, m = 0$.

3.1.1.3 Momentos invariantes de *Hu*

O momento ($m_{p,q}$) da imagem (I) é uma caraterística bruta da região, calculado através da integração de todos os *pixéis* da mesma (Bradski and Kaehler, 2008):

$$m_{p,q} = \sum_{i=1}^{n} I(x, y) x^p y^q \tag{3.9}$$

Em (3.9), o somatório é aplicado a todos os *pixéis* (n) da região. De referir que quando p e q são 0, o momento m_{00} vai corresponder à área, em *pixéis*, da região.

O cálculo dos momentos descrito possibilita a obtenção de algumas caraterísticas básicas que podem ser usadas para comparação de duas regiões. Contudo, na maior parte dos casos, os momentos calculados segundo (3.9) não apresentam informação suficiente. Como tal, uma das soluções possíveis consiste na utilização de momentos normalizados, em que objetos da mesma forma mas diferentes tamanhos possuem valores semelhantes (invariância a escalamentos):

$$\mu_{p,q} = \sum_{i=0}^{n} I(x,y)(x - x_{avg})^p (y - y_{avg})^q \tag{3.10}$$

$$x_{avg} = \frac{m_{10}}{m_{00}} \tag{3.11}$$

$$y_{avg} = \frac{m_{01}}{m_{00}} \tag{3.12}$$

(3.10) representa o cálculo do momento central. O momento central é bastante semelhante ao anterior - (3.9) - com a diferença dos valores de x e y surgirem deslocados pelos respetivos valores médios x_{avg} (3.11) e y_{avg} (3.12). A partir do cálculo dos momentos centrais, chega-se aos momentos normalizados:

$$\eta_{p,q} = \frac{\mu_{p,q}}{m_{00}^{(p+q)/2+1}} \tag{3.13}$$

Os momentos invariantes de *Hu* (Hu, 1962) são dos mais usados no reconhecimento visual. Consistem em combinações lineares dos cálculos dos momentos centrais normalizados (ver Figura 3.2). São invariantes a transformações como rotação, translação e escalamento.

$$
\begin{aligned}
h_1 &= \eta_{20} + \eta_{02} \\
h_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
h_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
h_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
h_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2) \\
&\quad + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) \\
h_6 &= (\eta_{20} - \eta_{02})((\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
h_7 &= (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) \\
&\quad - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2)
\end{aligned}
$$

Figura 3.2: Momentos invariantes de *Hu* (imagem retirada de Bradski and Kaehler (2008)).

De salientar que os momentos decrescem à medida que se sobe a ordem da expressão. Isto acontece porque os fatores normalizados (η) são menores do que 1. Matematicamente, facilmente se verifica que as expressões de ordem mais elevada tendem a ser menores.

Em relação aos momentos em si, os primeiros seis são invariantes a translações, escalamentos e rotações enquanto que o sétimo é chamado de invariante oblíquo. Este último serve para distinguir imagens espelhadas.

3.1.2 Métodos baseados em caraterísticas locais

Uma caraterística local é um padrão na imagem que difere substancialmente em relação à vizinhança (Tuytelaars and Mikolajczyk, 2007). Estas diferenças podem manifestar-se através da cor, intensidade e textura (as mais comuns). Em termos práticos, as caraterísticas locais podem ser pontos, cantos, manchas, bordas, entre outros. Os métodos baseados em caraterísticas locais calculam descritores (algumas medidas da região centrada numa caraterística local, também conhecida como ponto de interesse) e, através da aplicação de algoritmos de correspondências, é

possível associar descritores com valores semelhantes.

Estes métodos são invariantes a transformações geométricas da imagem, tais como translação, rotação e escalamento, mas também apresentam bons resultados em relação a alterações fotométricas, como brilho e luminosidade.

3.1.2.1 SIFT - *Scale Invariant Features Transform*

Com o intuito de responder às limitações de outros métodos que privilegiavam o reconhecimento da forma e aparência descurando as caraterísticas locais da imagem, novos métodos começaram a ser explorados. O SIFT (Lowe, 1999) veio revolucionar o campo do reconhecimento visual devido às potencialidades que apresentava.

O processamento pode ser dividido em quatro fases distintas (Lowe, 2004). Em primeiro lugar é feita uma procura de potenciais pontos de interesse por todas as escalas e localizações da imagem usando o algoritmo DoG (*Difference-of-Gaussian*). Através do DoG são criadas duas imagens com diferentes níveis de desfocagem (efeito *blur*) em relação à original. A imagem final é obtida a partir da subtração dos *pixéis* das duas imagens desfocadas e a consequente deteção de passagens por zero, i.e., quando há mudança de sinal. As passagens por zero poderão significar a existência de cantos ou áreas de *pixéis* com variações na vizinhança, ou seja, possíveis pontos de interesse[3]. De seguida, a cada um dos pontos de interesse detetados no passo anterior, é aplicado um modelo para determinar a localização e escala. A seleção final de pontos de interesse baseia-se num critério de estabilidade. Numa terceira fase é calculada a orientação e a mesma é atribuída a cada uma das localizações dos pontos de interesse. Por último, calculam-se os descritores através dos gradientes medidos na respetiva escala, na região em volta de cada ponto de interesse.

Caraterísticas da imagem são invariantes a translações, rotações e escalamentos. São também robustas a mudanças de luminosidade e adição de ruído.

[3] `http://www.roborealm.com/help/DOG.php` [acedido em 2014-01-16].

3.1.2.2 SURF - *Speeded Up Robust Features*

Pode-se considerar o SURF (Bay et al., 2008) como um *upgrade* do SIFT na medida em que produz resultados semelhantes, mas em tempos de processamento mais curtos.

O SURF faz bastante uso das imagens integrais. A utilização de imagens integrais reduz significativamente o tempo de computação (Evans, 2009). A imagem integral corresponde à soma das intensidades do *pixel* numa região retangular. A partir de uma imagem I e um ponto (x, y), a imagem integral $I_{\sum}$ é calculada através do somatório dos valores entre o ponto e a origem:

$$I_{\sum}(x, y) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(x, y) \tag{3.14}$$

Através das imagens integrais, facilmente se consegue calcular a área da região retangular, como se verifica pela Figura 3.3 e em (3.15):

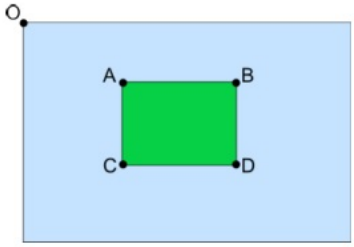


Figura 3.3: Cálculo da área através das imagens integrais (imagem retirada de Evans (2009)).

$$\sum = A + D - (C + B) \tag{3.15}$$

Para o cálculo dos pontos de interesse da imagem utiliza-se a matriz de *Hessian*

- (3.16) - em função do espaço x $= (x, y)$ e escala σ:

$$H(\times, \sigma) = \begin{bmatrix} L_{xx}(\mathrm{x}, \sigma) & L_{xy}(\mathrm{x}, \sigma) \\ L_{xy}(\mathrm{x}, \sigma) & L_{yy}(\mathrm{x}, \sigma) \end{bmatrix} \tag{3.16}$$

$L_{xx}(\mathrm{x}, \sigma)$ representa a convolução da derivada Gaussiana de 2ª ordem ($\frac{\partial^2 g(\sigma)}{\partial x^2}$) com a imagem no ponto x $= (x, y)$ ($\frac{\partial^2 g(\sigma)}{\partial y^2}$ para $L_{yy}(\mathrm{x}, \sigma)$ e $\frac{\partial^2 g(\sigma)}{\partial x \partial y}$ para $L_{xy}(\mathrm{x}, \sigma)$). Calculando o determinante de H e encontrando o máximo, o mesmo pode corresponder a um ponto de interesse.

De seguida, deve-se construir um espaço de escalas de modo a permitir a invariância a escalamentos. A determinação do espaço de escalas consiste na aplicação de uma função contínua utilizada para o cálculo de máximos em todas as escalas possíveis (Witkin, 1983). A abordagem utilizada no SIFT para construir um espaço de escalas consiste em variar o tamanho da imagem e o filtro Gaussiano é aplicado repetidamente, criando um efeito de *smooth* nas diferentes *layers* (ver Figura 3.4 à esquerda). De acordo com a abordagem utilizada no SURF, a imagem original não sofre alterações, apenas se varia o tamanho do filtro (ver Figura 3.4 à direita).

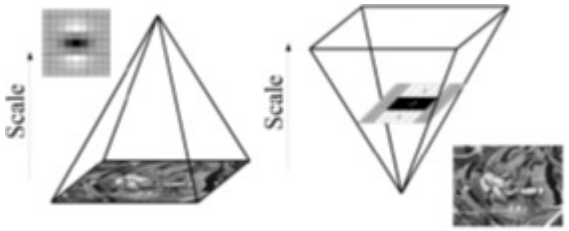


Figura 3.4: Pirâmide de filtragem (imagem retirada de Evans (2009)).

De seguida, de modo a encontrar a localização mais precisa dos pontos de interesse, deve-se aplicar um *threshold* adequado para que apenas os pontos "mais fortes" possam ser considerados como pontos de interesse. Após este passo é aplicado um algoritmo de supressão de não-máximos. Neste algoritmo cada *pixel* é

comparado com os seus 26 vizinhos (os 8 da mesma *layer* mais os 18 das *layers* superior e inferior).

Na Figura 3.5, o *pixel* marcado com um "X" é considerado um máximo no caso de ser maior que os 26 vizinhos. Por último, deve-se fazer a interpolação dos dados para encontrar a localização, tanto no espaço como na escala usada.

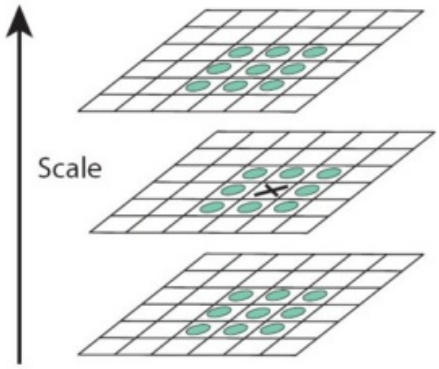


Figura 3.5: Supressão de não-máximos (imagem retirada de Evans (2009)).

Para obter invariância a rotações, cada ponto de interesse deve ser associado a uma orientação. Para determinar a orientação deve-se começar pelo cálculo das respostas da transformada de *Haar* (Haar, 1910) para todos os pontos de interesse ao longo dos eixos coordenados. A aplicação dos filtros correspondentes à transformada de *Haar* (também conhecidos como *Haar wavelets*) permite encontrar os gradientes em ambas as direções dos eixos coordenados x e y. De seguida, define-se uma janela deslizante com um ângulo de $\frac{\pi}{3}$. Calcula-se a soma de todos os valores nessa janela para as direções de ambos os eixos coordenados. Repete-se o mesmo processo para várias janelas. Por fim, seleciona-se a orientação que corresponde à direção da resposta da janela mais forte (repare-se na Figura 3.6 em que o maior vetor, o da direita, indica a orientação dominante).

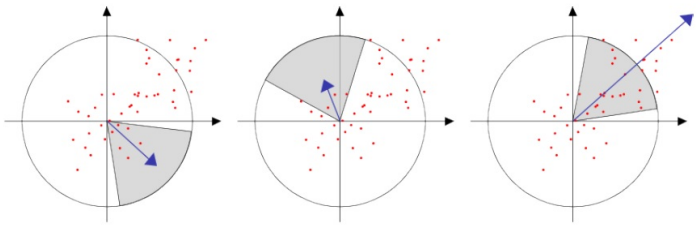


Figura 3.6: Determinação da orientação (imagem retirada de Evans (2009)).

Para calcular os descritores SURF, em primeiro é necessário construir janelas quadradas à volta dos pontos de interesse. Essas janelas vão estar orientadas de acordo com a direção encontrada no passo anterior.

Figura 3.7: Janelas de descritores (imagem retirada de Evans (2009)).

A janela é dividida em 4 x 4 sub-regiões dentro das quais são calculadas as transformadas de *Haar* para 25 pontos de amostragem regularmente distribuídos. Considerando dx e dy como respostas da transformada de *Haar*, obtém-se:

$$v_{subregião} = [\sum dx, \sum dy, \sum |dx|, \sum |dy|] \tag{3.17}$$

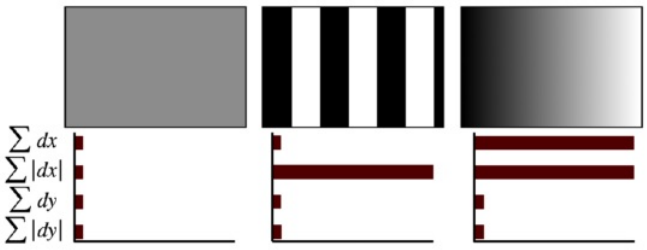


Figura 3.8: Respostas das Transformadas de *Haar*: no lado esquerdo, no caso de uma região homogénea, respostas têm valor baixo; no meio, no caso de haver frequências na direção de x, valor $\sum |dx|$ é alto; se a intensidade for crescente na direção de x, tanto $\sum dx$ como $\sum |dx|$ têm valores altos (imagem retirada de Bay et al. (2008)).

Como tal, cada sub-região contribui com 4 valores para o descritor, sendo que o vetor final apresenta um tamanho de 64 (4 x 4 x 4). O descritor SURF resultante é invariante a rotações, escalamentos, diferenças de brilho e contraste.

Quadrado verde da Figura 3.9 representa uma das 16 sub-regiões. Os pontos azuis são os 25 pontos de amostragem. x e y são calculados em relação à orientação dominante (Evans, 2009).

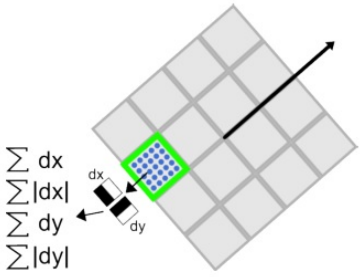


Figura 3.9: Componentes do descritor (imagem retirada de Evans (2009)).

Apesar do SURF apresentar melhorias consideráveis em termos de tempo de resposta em relação ao SIFT, com a banalização dos dispositivos móveis e utilização dos mesmos nas mais diversas aplicações relacionadas com o processamento de imagem, surgiu a necessidade de reduzir, de uma maneira ainda mais significativa, o tempo de processamento. Como tal, novos métodos têm sido estudados e apresentados recentemente.

3.1.2.3 BRIEF - *Binary Robust Independent Elementary Features*

O BRIEF (Calonder et al., 2010) é um descritor de pontos de interesse (não faz a deteção dos mesmos, sendo necessária a utilização de um método complementar para tal). É bastante robusto a transformações geométricas e fotométricas da imagem. Pode ser usado em aplicações de processamento de imagem para dispositivos móveis visto não exigir grande capacidade de processamento.

A abordagem mais comum na descrição de pontos de interesse consiste em fazer os cálculos com *floats* e, só depois, converter para binário. Os autores demonstram (Calonder et al., 2011) a possibilidade de calcular diretamente descritores binários, tornando as fases de descrição e correspondência muito mais rápidas. O processamento faz-se recorrendo a testes de diferença de intensidade.

3.1.2.4 BRISK - *Binary Robust Independent Scalable Keypoints*

O BRISK (Leutenegger et al., 2011) é um detetor e descritor binário cujo reduzido tempo de processamento o torna indicado para aplicações em tempo real.

Baseia-se na deteção de pontos de interesse através de um critério de saliências. Os pontos de interesse são detetados em camadas de oitavas (ver Figura 3.10).

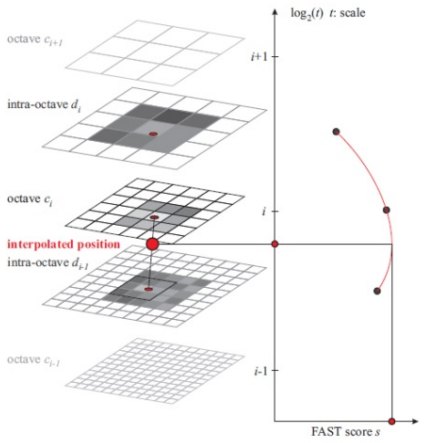


Figura 3.10: Pirâmide de uma imagem dividida em oitavas (imagem retirada de Leutenegger et al. (2011)).

O ponto de interesse é identificado na oitava c_i (Figura 3.10) analisando os 8 *pixéis* vizinhos à procura de saliências e o processo repete-se para a camada superior e inferior. Após ter sido obtida a localização dos pontos nas respetivas oitavas, estes são projetados no eixo de escala, obtendo uma parábola, procedendo-se posteriormente a uma interpolação para encontrar a sua verdadeira localização.

Em relação à descrição dos pontos de interesse, é aplicado na vizinhança de cada um dos últimos um padrão de amostragem composto por círculos concêntricos (ver Figura 3.11). Através da análise dos gradientes da intensidade local de cada um dos círculos, é obtida a direção. Tendo então o padrão orientado, facilmente são obtidas as comparações de brilho que permitem descrever um ponto de interesse.

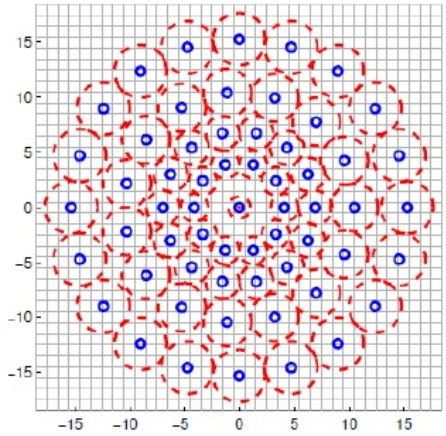


Figura 3.11: Padrão de amostragem BRISK (imagem retirada de Leutenegger et al. (2011)).

3.1.2.5 ORB - *Oriented FAST and Rotated BRIEF*

Método de deteção e descrição de pontos de interesse (Rublee et al., 2011). Usa o detetor FAST (Rosten and Drummond, 2006) (mais rápido mas que não calcula a orientação dos pontos de interesse) conjugado com um algoritmo de determinação

da orientação computacionalmente menos exigente. O cálculo dos descritores é feito através do recurso ao BRIEF.

3.1.2.6 FREAK - *Fast Retina Keypoint*

O FREAK (Alahi et al., 2012) é um dos métodos mais recentes. A descrição de pontos de interesse, segundo o FREAK, é baseada no modelo humano. Utiliza um padrão de amostragem inspirado na retina humana (ver Figura 3.12) que é aplicado a cada um dos pontos de interesse. De referir que o FREAK apenas faz a descrição de pontos de interesse. Ou seja, é necessário utilizar outro método para efetuar a deteção dos mesmos.

Segundos os autores, é um descritor mais rápido e mais robusto do que os anteriores.

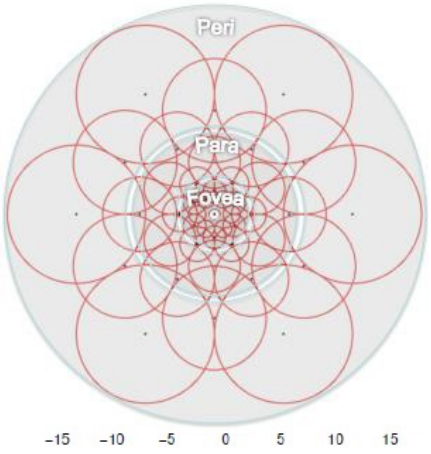


Figura 3.12: Padrão de amostragem FREAK (imagem retirada de Alahi et al. (2012)).

De acordo com a Figura 3.12, verifica-se que o padrão de amostragem utilizado difere substancialmente do padrão BRISK (Figura 3.11), nomeadamente no tamanho dos círculos e na sobreposição dos mesmos. O funcionamento é similar ao BRISK, no entanto este método defende que apenas os primeiros 512 pares fornecem informação descritiva sobre uma imagem, ao contrário do BRISK que usa todos os

pares existentes. Só por este facto facilmente se percebe a maior velocidade do FREAK.

3.1.3 Discussão do estado da arte

Numa perspetiva de criar um sistema que funcione em tempo real importa, antes de mais, a rapidez de processamento, não descurando a eficácia do programa. Em termos de rapidez de processamento, os métodos de reconhecimento baseados na forma seriam, de facto, a abordagem ideal devido ao reduzido processamento que exigem. Contudo, e conforme o explicado na secção 3.1.1, estes métodos trabalham com imagens binárias, o que compromete a eficácia do sistema na medida em que objetos com formas semelhantes vão ser descritos de maneira também semelhante. Porém, objetos "únicos" com formas caraterísticas podem ser reconhecidos através de uma abordagem baseada na forma. Para a distinção de objetos com formas semelhantes deve ser utilizada uma abordagem complementar baseada em caraterísticas locais. Entre os métodos apresentados neste capítulo destacam-se o SURF, o BRISK e o FREAK. O SURF, devido à sua eficiência comprovada, sendo mesmo, dentro dos métodos mais recentes de reconhecimento de caraterísticas invariantes, aquele que tem disponibilizada mais informação na literatura. Por outro lado, o BRISK e o FREAK, mais recentes, "prometem" maior rapidez, e, como tal, não deverão ser descartados. Contudo, e segundo os autores, o aumento da velocidade de processamento associado a cada um dos mesmos é um fator diferenciador em dispositivos móveis (com menor capacidade de processamento), restando saber se, no sistema em causa, existe essa necessidade.

Sintetizando, a intenção passa por desenhar um sistema híbrido que contemple a junção de um método holístico com um método baseado em caraterísticas locais. Como tal, a escolha do método holístico está maioritariamente dependente da sua capacidade em termos de tempo de resposta, ficando o método baseado em caraterísticas locais responsável pela resolução de situações ambíguas (dois objetos

diferentes com formas semelhantes, por exemplo). Sendo assim, e dada a eficiência comprovada dos momentos invariantes de *Hu*, tanto em eficácia como em reduzido tempo de resposta, a solução passa por usar este último complementado por um ou dois métodos baseados em caraterísticas locais (recordando a secção 3.1.2.6, o FREAK apenas faz a descrição de pontos de interesse, sendo necessário usar um outro método para a deteção dos mesmos) que permitam fazer a distinção entre objetos com formas semelhantes.

3.2 Implementação

A implementação do sistema contempla sete etapas fundamentais[4], expostas no diagrama da Figura 3.13.

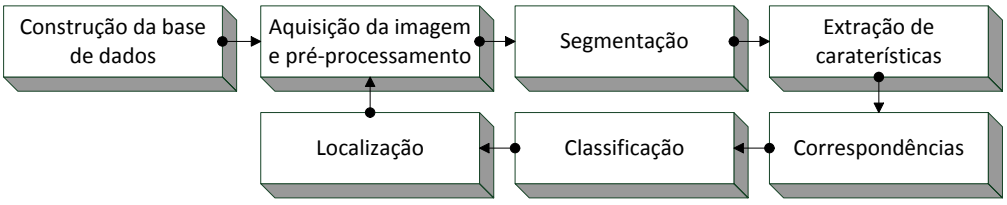


Figura 3.13: Etapas do sistema de reconhecimento e localização de objetos.

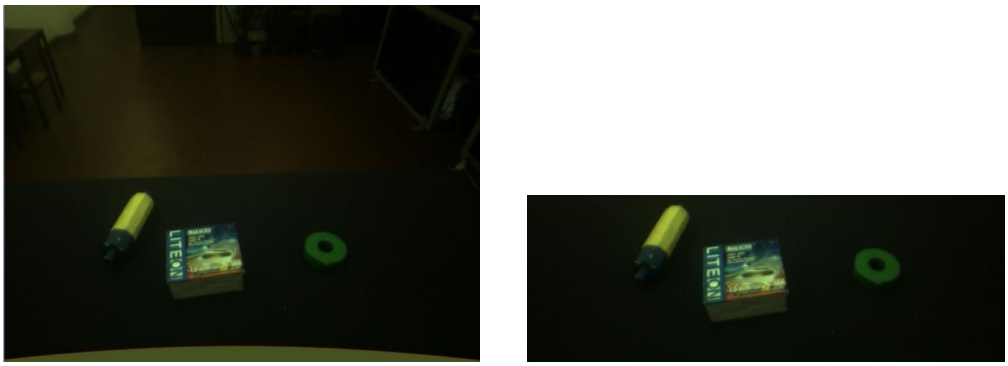
Antes do sistema entrar no ciclo, é necessário construir a base de dados relativa aos objetos a identificar. Isto é, vetores de pontos de interesse e matrizes de descritores dos objetos a identificar segundo método baseado em caraterísticas locais e vetores de momentos dos objetos a identificar segundo método holístico. Este passo apenas necessita de ser executado uma vez, poupando, deste modo, processamento. Após a construção da base de dados o sistema entra no ciclo, ficando apto para funcionamento em tempo real.

[4]Apenas consideradas etapas relativas ao sistema de visão. Ou seja, neste diagrama não é considerada a comunicação com as camadas de alto nível do sistema.

Apresenta-se, seguidamente, a abordagem em termos de implementação ao ciclo exposto no diagrama da Figura 3.13.

3.2.1 Aquisição da imagem e pré-processamento

Após a aquisição da imagem é necessário limitar a informação ao máximo, i.e., redimensionar a imagem (janela de visualização) ao espaço de trabalho do robô. Este é um passo bastante relevante na medida em que, com o avançar da complexidade dos problemas, informação a mais pode gerar situações conflituosas e aumentar o tempo de resposta do programa. Para tal, aplica-se um ROI (*Region of Interest*) (Figura 3.14b) sobre a imagem original (Figura 3.14a). Este foi determinado de forma manual tendo por base ângulos fixos de *pan* (rotação das câmaras segundo o eixo horizontal), em -3º, e de *tilt* (rotação das câmaras segundo o eixo vertical), em -39º[5].



(a) Imagem adquirida. **(b)** Imagem redimensionada.

Figura 3.14: Aquisição e redimensionamento da imagem.

Ainda antes de iniciar a segmentação, é necessário converter a imagem para tons de cinzento (Figura 3.15) de modo a poder executar o processamento posterior. Ao contrário das imagens RGB (*Red*, *Green*, *Blue*), de três canais, as imagens em

[5]Na implementação final o movimento de *pan-tilt* vai variar em tempo-real, com ajuste automático do ROI consoante os ângulos.

tons de cinzento apenas possuem um canal de 8 *bits* por *pixel*, em que o respetivo valor varia entre 0 (preto) e 255 (branco).

Figura 3.15: Transformação da imagem original numa em tons de cinzento.

3.2.2 Segmentação

A segmentação permite detetar todos os objetos (ou mesmo ruído) presentes no plano de trabalho, estejam ou não contemplados na base de dados. Este passo é independente da extração de caraterísticas, podendo mesmo ser executado depois do último. A associação entre os dois passos apenas é efetuada na secção 3.2.5.

Para tornar a segmentação mais precisa e eliminar o ruído, deve-se aplicar um efeito de *blur* à imagem em tons de cinzento. Após este passo, aplica-se o algoritmo *Canny* (Canny, 1986) para a deteção de cantos na imagem (Figura 3.16). O algoritmo *Canny* contempla quatro passos distintos (Moeslund, 2009)[6]:

- Procura de gradientes na imagem: maiores magnitudes poderão corresponder a cantos na imagem (em termos práticos equivale a uma variação na intensidade da imagem em tons de cinzento).
- Supressão de não-máximos: apenas os máximos locais devem ser considerados, tornando os cantos *blurred* (turvos) em cantos *sharp* (afiados).
- Aplicação de *threshold* duplo: *pixéis* dos cantos que ultrapassem o valor do *threshold* superior são considerados como "fortes"; os que são mais baixos

[6]Considerando que a aplicação do efeito de *blur* não faz parte do algoritmo *Canny*.

do que o valor de *threshold* inferior são suprimidos; os que se encontram no intervalo entre *thresholds* são considerados como "fracos".

- Aplicação de histerese: cantos "fortes" são automaticamente validados; cantos "fracos" só são validados no caso de estarem conectados a algum canto "forte".

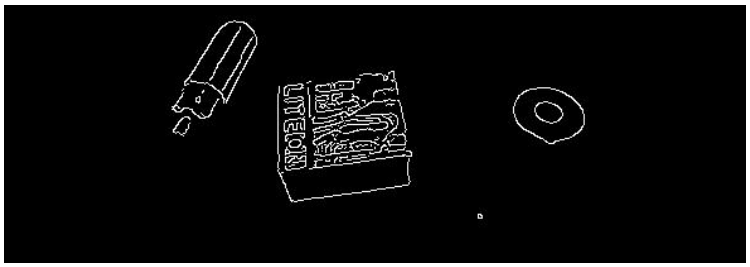


Figura 3.16: Aplicação do algoritmo *Canny* sobre a imagem em tons de cinzento.

De seguida, dilata-se a imagem para unir eventuais cortes gerados pelo algoritmo *Canny* (Figura 3.17).

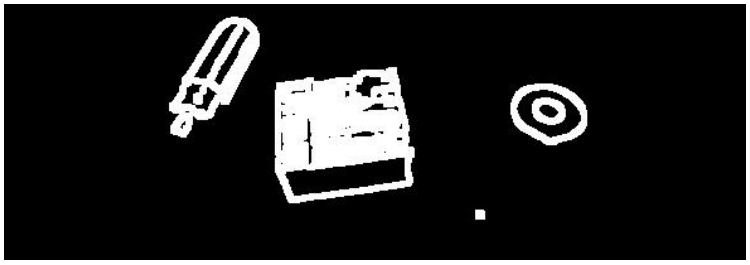


Figura 3.17: Dilatação da imagem.

Por último, guardam-se num vetor todos os conjuntos de pontos, conectados entre si, correspondentes aos contornos exteriores fechados de cada objeto (Figura 3.18).

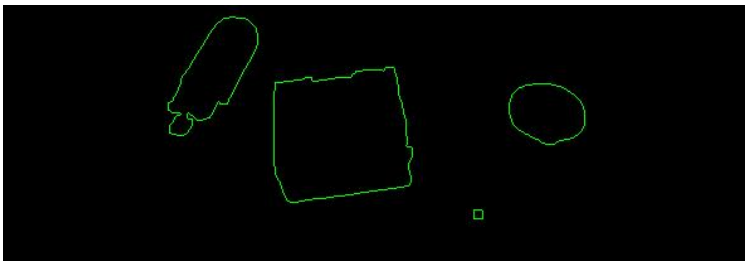


Figura 3.18: Contornos exteriores fechados dos objetos.

3.2.3 Extração de caraterísticas

Nesta fase apenas são extraídas as caraterísticas locais dos objetos no plano de trabalho. Isto é, a extração dos momentos invariantes de *Hu* (método holístico), apenas é efetuada no caso de, na fase de classificação (ver secção 3.2.5), o objeto não for identificado através de uma abordagem baseada em caraterísticas locais.

De acordo com o referido na secção 3.1.3 relativamente aos métodos baseados em caraterísticas locais, foram selecionados três (SURF, BRISK e FREAK) para análise e comparação de modo a escolher qual deles se ajustaria melhor aos propósitos do sistema. Como tal, foi desenhada uma aplicação visual (ver Figura 3.19), em C/C++, para comparação destes métodos com recurso a imagens estáticas.

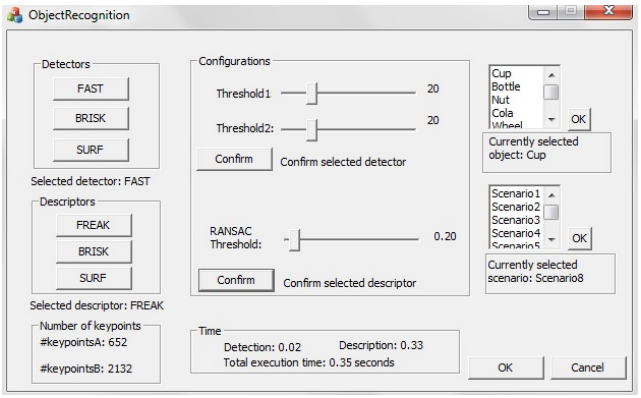


Figura 3.19: Janela principal da aplicação visual.

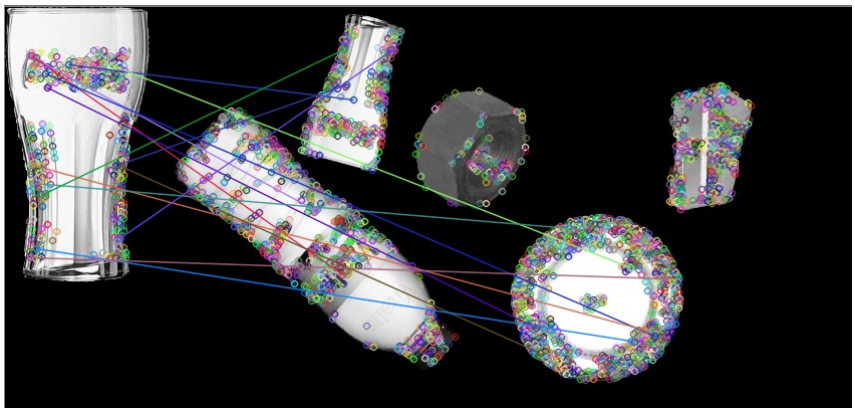
Esta aplicação permite escolher os métodos de deteção e descrição de pontos

de interesse. Permite também a definição dos diferentes *thresholds*. Possui um conjunto de objetos (*cup*, *bottle*, *nut*, *cola*, *wheel*, *tube* e *sharpens*) e diferentes cenários que contemplam esses objetos, para geração de correspondências. A aplicação devolve algumas informações relevantes, como o número de pontos de interesse detetados e os tempos de execução.

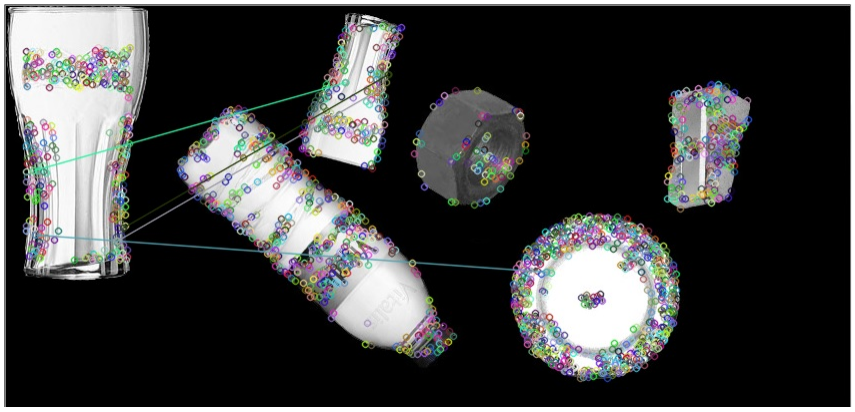
Foi usado o algoritmo de correspondências *Brute-Force matcher* (ver secção 3.2.4) nos três métodos. Visto que tanto o BRISK como o FREAK são métodos binários, usou-se, para estes últimos, a distância de *Hamming* (Hamming, 1950) que calcula as diferenças entre duas *strings* binárias com o mesmo comprimento. Apresentam-se, seguidamente, alguns dos resultados que estiveram na base da escolha do método baseado em caraterísticas locais. Para comparação, utilizou-se um cenário composto pelos objetos *cup*, *bottle*, *nut*, *plate* (objeto "intruso", i.e., não adicionado na base de dados) e *sharpens*. Para além da informação relativa ao número de pontos de interesse detetados e respetivos tempos de deteção e descrição (ver exemplo relativo ao objeto *cup* na Tabela 3.1), a aplicação permite, também, ver a localização dos pontos de interesse e as correspondências geradas, como se verifica pela Figura 3.20, através do objeto *cup*.

	PI objeto	PI cenário	Tempo de deteção (s)	Tempo de descrição (s)	Tempo total (s)
FAST+FREAK	652	2132	0,02	0,23	0,25
BRISK	288	1327	1,27	1,30	2,57
SURF	433	1109	0,41	1,48	1,89

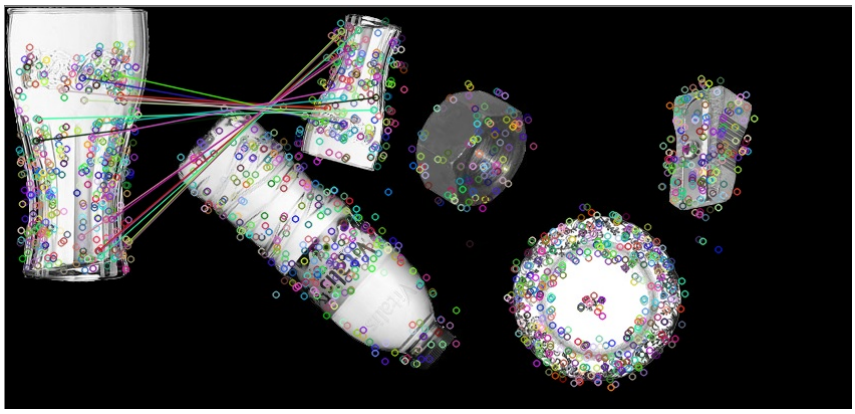
Tabela 3.1: Resultados do número de pontos de interesse detetados e tempos de deteção e descrição para cada um dos métodos de extração de caraterísticas locais (objeto *cup*). Legenda: PI - pontos de interesse.



(a) Deteção FAST e descrição FREAK.



(b) Deteção e descrição BRISK.



(c) Deteção e descrição SURF.

Figura 3.20: Resultados das correspondências obtidas segundo métodos de extração de caraterísticas locais.

Pela Figura 3.20 constata-se que tanto o BRISK como o FREAK fazem uma deteção mais precisa dos pontos de interesse. Porém, a descrição dos mesmos e respetiva análise de correspondências revelou-se mais eficiente segundo o SURF. Isto porque através do último foi possível gerar boas correspondências entre cada um dos objetos adicionados e a imagem do cenário, como se verifica pela Tabela 3.2. A Tabela 3.2 apresenta os objetos que obtiveram, em maioria, boas correspondências segundo cada um dos métodos assim como o tempo médio de execução[7].

	Cup	*Bottle*	*Nut*	*Sharpens*	Tempo (s)
FAST+FREAK		X	X		0,39
BRISK	X	X	X		2,68
SURF	X	X	X	X	2,61

Tabela 3.2: Objetos com predominância de boas correspondências (assinalados com X) e respetivos tempos médios de execução de cada uma das abordagens.

Os tempos elevados devem-se, essencialmente, à alta resolução das imagens utilizadas. Porém, com esta aplicação pretende-se fazer uma análise comparativa das três abordagens, não sendo para o efeito relevante a resolução das imagens mas sim que os objetos e respetivos cenários sejam idênticos. Os algoritmos de otimização e geração de correspondências devem também ser semelhantes, na medida do possível, considerando que dois dos métodos exigem que seja usada a distância de *Hamming*. Ou seja, neste caso importam os tempos relativos.

Com base nos resultados da aplicação visual, a melhor abordagem para extração de caraterísticas locais, i.e., deteção e descrição de pontos de interesse, consiste

[7] Testes efetuados num *Intel® Core™ i5-2410M, 2.30 GHz, 4.0 GB RAM.*

no método SURF. Através deste último seria possível identificar todos os objetos visto que o mesmo devolve uma maioria de boas correspondências e poucos falsos positivos. O FREAK (conjugado com o método de deteção FAST), apesar de ser significativamente mais rápido, apresenta algumas fragilidades, principalmente quando o objeto é rodado. Já o BRISK acaba por apresentar tempos de execução semelhantes ao SURF e uma eficácia inferior em termos de correspondências.

3.2.4 Correspondências

Após a deteção e descrição de pontos de interesse dos objetos no plano de trabalho, associam-se esses mesmos descritores com os descritores presentes na base de dados. Para tal, é necessário recorrer a um algoritmo de correspondências. Utilizou-se o *Brute-Force matcher* programado no modo *k-NN* (*k-Nearest Neighbors*). Ou seja, através deste modo, o algoritmo devolve as *k* melhores correspondências. Neste caso escolheu-se $k = 2$. Isto é, aplica-se o *Brute-Force matcher* entre cada uma das matrizes de descritores relativas aos objetos da base de dados e a matriz de descritores do plano de trabalho. Para cada um dos descritores de cada objeto, são selecionadas as duas melhores correspondências no plano de trabalho. Após este passo é feita uma otimização no sentido de eliminar correspondências em excesso. Ou seja, a primeira correspondência deve apresentar uma distância[8] significativamente menor em relação à segunda correspondência. A melhor correspondência só é aceite no caso da distância se situar abaixo dos 65% em relação à distância da segunda correspondência. Os dados são guardados numa estrutura que contempla o índice do objeto identificado e as respetivas coordenadas (ver Figura 3.21). Por sua vez, a estrutura é guardada num vetor de estruturas, precavendo a eventualidade de existir mais do que um objeto no plano de trabalho.

[8]Quantificação da diferença entre descritor do objeto e do plano de trabalho (distância euclidiana). Quanto menor for a distância, mais "forte" é a correspondência.

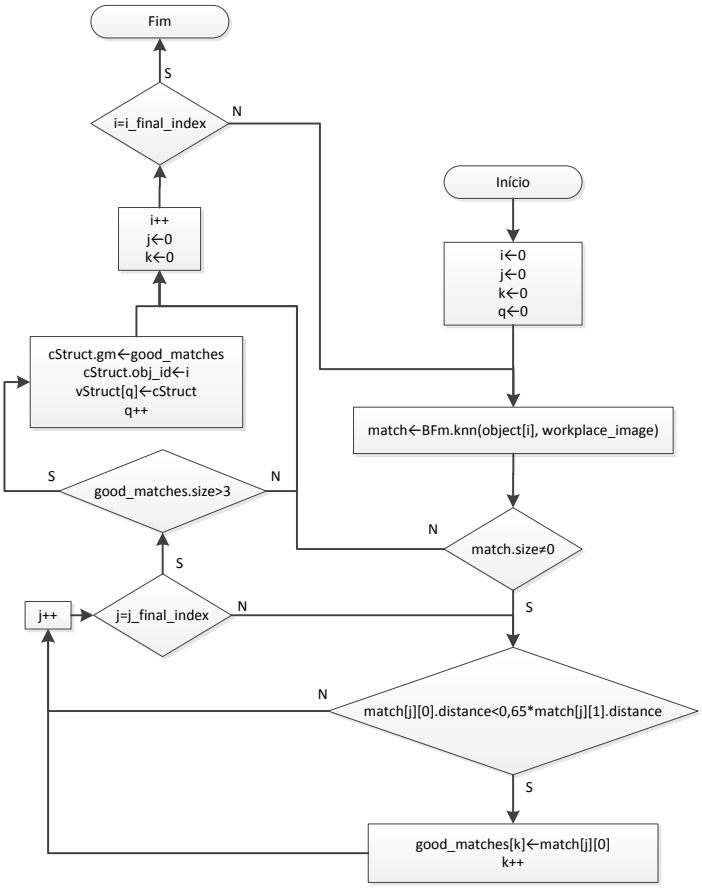


Figura 3.21: Fluxograma do processamento e otimização de correspondências.

Relativamente ao fluxograma da Figura 3.21, *object[i]* representa um vetor (com dimensão correspondente ao número de objetos adicionados) de matrizes de descritores. Por seu turno, *workplace_image* representa a matriz de descritores do plano de trabalho. Tal como referido, o processamento *k-NN* devolve, para cada descritor do objeto a ser analisado, as duas melhores correspondências. Neste caso *match[j][0]* e *match[j][1]*. Caso a correspondência seja aceite, guarda-se, num vetor, a respetiva correspondência. Os dados referentes ao índice do objeto cujo número de correspondências é superior a três e respetivas coordenadas são guardados numa estrutura (*cStruct*). Visto que pode ser identificado mais do que um objeto, é

necessário criar um vetor de estruturas (*vStruct*) para armazenar dados relativos a diferentes objetos.

3.2.5 Classificação

Para o sistema identificar determinado objeto é necessário um mínimo de quatro correspondências. Mas a identificação, por si só, não é suficiente. É necessário obter informação do objeto como um todo e não apenas das suas caraterísticas locais. Uma das abordagens mais comuns para fazer face a este problema consiste em utilizar o algoritmo RANSAC (*Random Sample Consensus*) (Fischler and Bolles, 1981) que estima um modelo matemático a partir de um conjunto de pontos. Nesse conjunto constam *inliers* (pontos "bons") e *outliers* (pontos "intrusos"). Em primeiro, o modelo é estimado a partir de um conjunto de possíveis *inliers*. Os restantes pontos são testados segundo esse modelo e se encaixarem no mesmo são também considerados como possíveis *inliers*. O modelo é considerado apto se tiver obtido um número mínimo de *inliers*. É então re-estimado tendo por base outros conjuntos iniciais de possíveis *inliers*[9] (ver Figura 3.22).

Porém, após alguns testes verificou-se que o algoritmo RANSAC pode apresentar alguma instabilidade em sistemas de tempo real. Ou seja, o algoritmo funciona bem no caso de serem obtidas várias correspondências, o que nem sempre é o caso, principalmente quando se aplicam previamente algoritmos de otimização. Sendo a estabilidade um fator preponderante para o desempenho do sistema, optou-se pela segmentação e respetiva classificação dos contornos obtidos tendo por base os pontos recolhidos na fase de correspondência. A segmentação seria sempre necessária na fase de localização espacial do objeto (ver secção 3.2.6.1), independentemente do uso do algoritmo RANSAC. Como tal, aplicando nesta fase, acaba por servir dois propósitos.

[9] `http://www.pointclouds.org/documentation/tutorials/random_sample_consensus.php` [acedido em 2014-08-27].

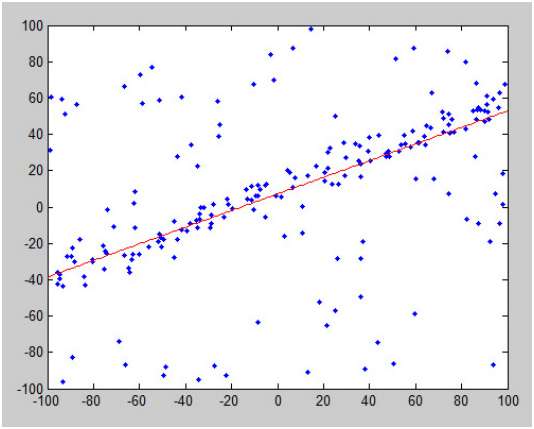


Figura 3.22: Reta $y = mx + b$ (a vermelho), criada pelo algoritmo RANSAC através dos *inliers* (pontos coincidentes com a reta) (imagem retirada de `http://www.mathworks.com/discovery/ransac.html` [acedido em 03-09-2014]).

Para o processo de classificação em si, estando os objetos do plano de trabalho segmentados (ver secção 3.2.2), verifica-se se as coordenadas guardadas (na estrutura) de cada objeto identificado se encontram dentro dos limites de algum dos contornos do plano de trabalho. Para tal, recorre-se ao processamento *Point in Polygon* (Sutherland et al., 1974; Hormann and Agathos, 2001): desenhando uma linha reta entre um ponto fora do contorno (início da janela de visualização, por exemplo) e o ponto que se pretende testar, se a mesma intersetar o contorno num número de vezes ímpar, significa que o ponto está dentro do contorno. Se intersetar num número de vezes par, significa que o ponto está fora do contorno (algoritmo também conhecido como *even-odd rule*). Para cada contorno são testados todos os pontos correspondentes a cada um dos objetos identificados. Um contador faz o somatório de pontos que se encontram dentro do contorno. Se esse somatório representar um máximo comparativamente com os somatórios relativos a outros contornos, índice do contorno é guardado na estrutura, no índice do vetor de estruturas correspondente ao objeto a ser analisado (ver Figura 3.23).

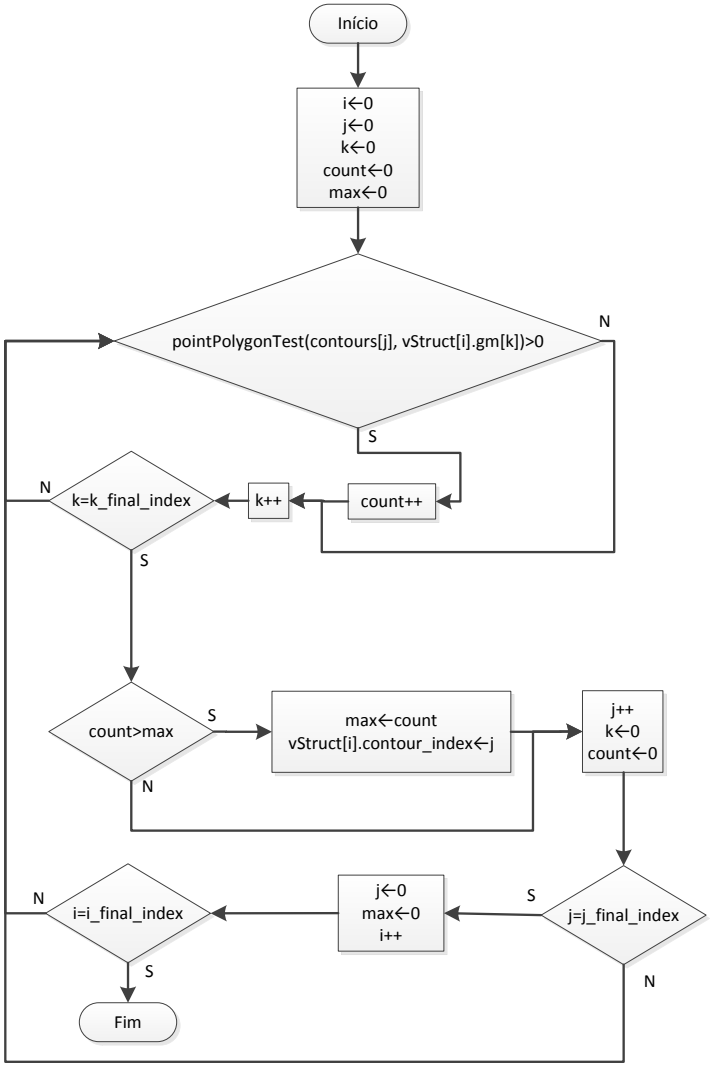


Figura 3.23: Fluxograma do processo de associação de contornos a correspondências.

A função *pointPolygonTest* testa se um determinado ponto (*vStruct[i].gm[k]*) se encontra dentro de um contorno (*contours[j]*). Caso isso se verifique, resultado devolvido é maior que zero. Após encontrar o máximo de pontos dentro de um contorno fechado, índice do contorno é armazenado no vetor de estruturas (*vStruct[i].contour_index*$\leftarrow j$), associando, assim, os pontos resultantes das boas correspondências do objeto ao respetivo contorno.

Sintetizando, a classificação envolve a associação de um conjunto de pontos a determinado contorno fechado. Se esta associação não for feita, independentemente do sistema "saber" que o objeto está presente, o contorno não é classificado. Repare-se na Figura 3.25 em que é feita essa associação com base nas correspondências obtidas entre a imagem inserida no sistema (Figura 3.24) e o plano de trabalho.

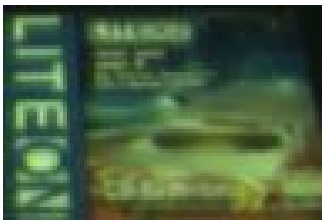


Figura 3.24: Imagem inserida no sistema e a partir da qual se obtém a matriz de descritores relativa a esse objeto.

Figura 3.25: Associação de um conjunto de pontos (circunferências vermelhas), resultantes da análise de correspondências, a um contorno fechado (a verde).

No caso de um contorno não ser associado a um objeto, isso poderá significar duas coisas: ou se trata de ruído, i.e., algum objeto não adicionado, ou a sua classificação deverá ser feita recorrendo aos momentos invariantes de *Hu*. Para fazer essa verificação é necessário obter a representação binária dos objetos em questão (Figura 3.26).

Figura 3.26: Imagem binária dos objetos não reconhecidos segundo método baseado em caraterísticas locais.

Como se verifica pela Figura 3.26, há um pequeno ruído, correspondente a um pionés, próximo do objeto redondo. Através da verificação dos momentos, esta representação binária deverá ser desconsiderada.

Após extração dos momentos das diferentes representações binárias (em primeiro devem ser isoladas), faz-se a comparação destes com os da base de dados, com uma margem de 10% (superior e inferior). Se o conjunto de momentos analisado se inserir dentro do intervalo de um determinado objeto previamente definido na base de dados, esse objeto é dado como identificado.

3.2.6 Localização

Após isolamento e identificação dos objetos relevantes, importa localizar os mesmos. Relativamente à localização, para todos os objetos deve ser determinada a respetiva posição no espaço (através de um sistema de visão estereoscópico), mas para objetos sujeitos a eventual manipulação por parte do robô, poderá também interessar o cálculo da sua orientação.

3.2.6.1 Posição

Através da associação da região (limitada pelo contorno guardado) correspondente a cada objeto com a imagem de disparidade do plano de trabalho (Figura 3.27), obtém-se o ponto médio fazendo uma média pesada de todos os pontos no espaço

tridimensional. Porém, estas coordenadas são relativas aos eixos coordenados do sistema de visão estereoscópico. Ou seja, deste modo os objetos não se encontram aptos para o respetivo manuseamento pois o eixo do robô (*World*) difere do eixo do sistema de visão (*Vision*), como se verifica pela Figura 3.28.

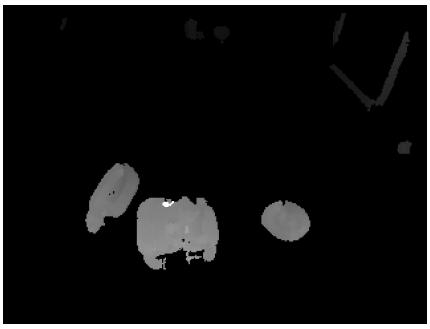


Figura 3.27: Imagem de disparidade do plano de trabalho: partes mais claras correspondem a zonas mais próximas do sistema de visão (sistema apenas processa imagens em tons de cinzento, o que explica a diferença desta imagem em relação à da Figura 2.2b).

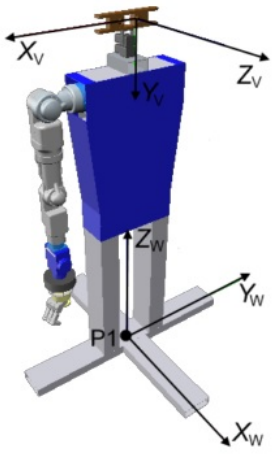


Figura 3.28: Sistemas de eixos coordenados (imagem retirada de Silva (2008)).

Para obter as coordenadas relativas ao eixo do robô, utilizam-se as matrizes de transformação de Denavit and Hartenberg (1955). Para uma análise mais detalhada sobre os cálculos da transformação mencionada, consultar Silva (2008).

3.2.6.2 Orientação

Para os objetos cuja informação sobre a respetiva orientação é relevante para o desempenho do sistema (objetos a serem manipulados pelo robô, por exemplo) deve haver um tratamento específico de modo a ser possível efetuar os cálculos associados. Sendo assim, estando o objeto devidamente reconhecido e a sua posição identificada, é possível calcular a orientação através da deteção por cor (ver explicação da deteção por cor, em detalhe, na secção 4.2.2) e aplicação de cálculos trigonométricos.

Apresenta-se, seguidamente, um exemplo do cálculo da orientação para uma coluna em que as respetivas extremidades possuem cor vermelha (ver Figura 3.29a). Em primeiro, efetua-se a deteção da respetiva cor, como se verifica pela Figura 3.29b, em que se pretende detetar as extremidades. De referir que esta procura está restringida aos limites do ROI que define o objeto.

(a) Imagem original. **(b)** Representação binária.

Figura 3.29: Deteção da cor vermelha.

De seguida, preenchem-se as duas regiões (ver Figura 3.30) com diferentes cores de modo a poder isolá-las e obter as respetivas coordenadas relativas ao eixo do robô.

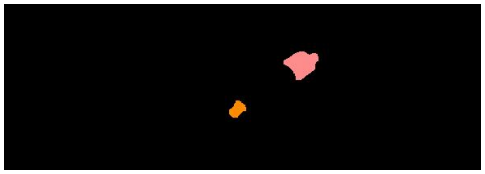


Figura 3.30: Divisão das duas regiões através da atribuição de diferentes cores.

Estando as regiões isoladas e a respetiva posição calculada, entramos num problema de trigonometria.

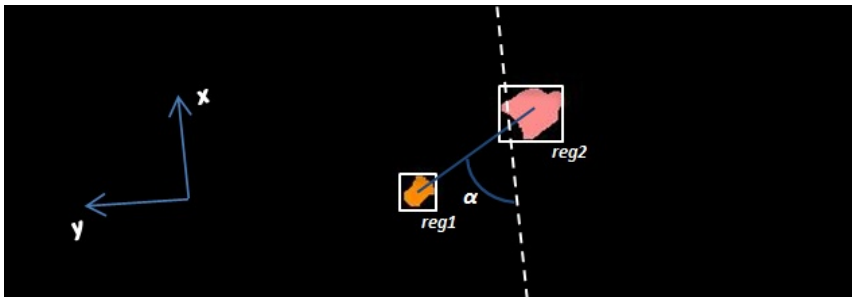


Figura 3.31: Cálculo da orientação (imagem adaptada a partir da Figura 3.30).

$$x = x_{reg2} - x_{reg1} \tag{3.18}$$

$$y = y_{reg2} - y_{reg1} \tag{3.19}$$

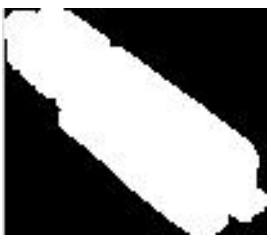
$$\alpha = arctan\frac{y}{x} \tag{3.20}$$

De referir que a ligeira deslocação dos eixos coordenados se deve ao facto de ser a imagem da câmara esquerda a ser processada na visualização em tempo real, com um *pan* de -3º. Note-se que, apesar da imagem da Figura 3.30 ser processada a partir do sistema de visão (sistema de eixos coordenado *Vision*), os cálculos para a orientação são feitos em relação ao sistema de eixos coordenado *World* (perspetiva do robô em relação ao mundo - ver Figura 3.28). Ou seja, os cálculos para a orientação são feitos recorrendo ao ponto médio de cada uma das regiões resultantes do processamento da imagem de disparidade, obtida a partir de ambas as câmaras.

No caso de o ROI de um determinado objeto a ser analisado conter uma outra peça da mesma cor (ver Figura 3.32), isso pode originar um problema de interpretação no sistema.

Figura 3.32: Imagem original em que o ROI da coluna (objeto que se pretende processar) abrange parte da roda verde.

Ao contrário da fase de classificação em que situações deste género não originam problemas de isolamento pois apenas se considera a região delimitada pelo contorno que define o objeto, neste caso o processamento inicial tem de ser feito numa imagem colorida. Visto que as partes coloridas do objeto a analisar estão separadas, se no mesmo ROI estiver outro objeto (ou parte dele) da mesma cor, o sistema vai detetar mais do que os dois contornos necessários para o cálculo da orientação. Para resolver este problema deve-se aplicar a função de lógica binária *AND* entre a imagem binária do objeto isolado resultante da fase de extração (Figura 3.33a) e a imagem binária resultante da procura por cor limitada ao ROI do objeto que se está a analisar (Figura 3.33b).



(a) Imagem binária resultante da extração.

(b) Imagem binária resultante da procura por cor.

(c) Resultado final.

Figura 3.33: Operação lógica *AND* entre duas imagens binárias.

A partir do resultado exposto na Figura 3.33c é possível obter os contornos das extremidades. Obtidos os contornos, o processo é o mesmo em relação ao explicado no início da secção. Ou seja, preenchem-se cada um dos contornos com duas cores distintas e aplicam-se os cálculos trigonométricos especificados para obter a orientação do objeto.

Nem todos os objetos estão devidamente adaptados para funcionarem de acordo com esta metodologia. Para tal, uma solução de adaptação fácil e prática consiste em colocar duas marcas de dimensão suficientemente grande de modo a serem percetíveis pelo sistema de visão em todas as disposições do objeto (não importa a forma), em cada uma das extremidades (ver secção 3.3.2).

Como se referiu, o cálculo da orientação é um processo específico do objeto em questão visto que nem todos os objetos necessitam do mesmo (como é o caso dos objetos redondos ou objetos de disposição vertical, como garrafas). Ou seja, esta é uma das exceções do processamento genérico do sistema na medida em que cada objeto deve ser considerado individualmente. Mesmo no caso da adaptação manual deve-se ter em conta a cor do objeto (a cor das marcas deve sempre diferir da cor do objeto onde são inseridas).

3.3 Resultados

Para teste e validação do sistema híbrido implementado, selecionaram-se três cenários (com quatro objetos cada) relevantes para as tarefas de interação e colaboração humano-robô (ver Figura 3.34): um cenário de construção (habitualmente usado nas experiências realizadas no Laboratório de Robótica Móvel e Antropomórfica), um cenário de refeição e um cenário de leitura. Estes dois últimos surgiram após visita ao *Camélia Hotel & Homes*, do grupo AMI (Assistência Médica Integral®), fruto da parceria entre a Universidade do Minho e esta instituição no âmbito do

projeto europeu NETT[10] (*Neural Engineering Transformative Technologies*).

Ou seja, possíveis tarefas de interação e colaboração a ser executadas entre os utentes da instituição e o robô podem-se inserir em ambiente de refeição, como por exemplo passar um prato, encher o copo de água, entre outras. Por outro lado, constatou-se que um dos espaços mais frequentados - sala comum - pode também ser útil no processo de interação mencionado. Isto é, dados os problemas de visão decorrentes da idade avançada de grande parte dos utentes, pode-se tornar complicado selecionar uma determinada revista ou jornal pretendido, podendo o robô fazer a respetiva seleção. De notar que, com o sistema genérico implementado, adicionar as diferentes revistas/jornais do dia ao sistema é um processo simples que se resume a tirar uma foto ao objeto (ou duas, no caso de se querer inserir, também, a contracapa).

Note-se que a escolha dos métodos para os objetos usados nos testes baseou-se nas caraterísticas dos mesmos. Por exemplo, não faria sentido selecionar uma roda uniforme para ser identificada recorrendo a uma abordagem baseada em caraterísticas locais. Por outro lado, a forma (retangular) de uma revista é pouco caraterística, sendo que a utilização de um método holístico baseado na forma para identificar este objeto pode não ser suficiente, até porque no sistema poderá constar mais do que uma revista. Ou seja, para a abordagem baseada em caraterísticas locais são usados objetos com caraterísticas passíveis de serem extraídas pelo sistema de visão de modo a gerar as respetivas correspondências. Para a abordagem baseada no método holístico são usados objetos com formas caraterísticas ou objetos que não contenham elementos suficientes para extração através do método baseado em caraterísticas locais (ver Tabela 3.3).

[10] `http://www.neural-engineering.eu/` [acedido em 2014-10-10].

(a) Base.

(b) Coluna.

(c) Roda.

(d) Martelo.

(e) Copo.

(f) Garrafa.

(g) Prato.

(h) Chaleira.

(i) Jornal A.

(j) Jornal B.

(k) Revista A.

(l) Revista B.

Figura 3.34: Objetos usados para validação do sistema de reconhecimento implementado (imagens não estão à escala).

3.3.1 Análise de robustez dos algoritmos de reconhecimento

Para fazer a análise de robustez ao sistema implementado, dividiu-se o plano de trabalho em seis partes iguais. Para cada uma dessas seis partes, testaram-se todos os objetos[11] (individualmente) em oito rotações diferentes[12] (separadas por 45°), perfazendo um total de 48 amostras para cada objeto. Na Tabela 3.3 apresentam-se

[11] À exceção da base que, devido à sua dimensão, apenas foi testada para uma posição.

[12] Quando a rotação não é aplicável, o caso da roda por exemplo, apenas são usadas as seis amostras das seis posições do plano.

os resultados obtidos após o teste mencionado.

Objeto	Taxa de identificação	Cenário	Método
Base	87,5%	A	MH
Coluna	87,5%	A	MH
Roda	100%	A	MH
Martelo	95,8%	A	MH
Copo	100%	B	MH
Garrafa	100%	B	MH
Prato	91,7%	B	CL
Chaleira	89,6%	B	MH
Jornal A	89,6%	C	CL
Jornal B	87,5%	C	CL
Revista A	85,4%	C	CL
Revista B	93,7%	C	CL

Tabela 3.3: Resultados da análise de robustez ao sistema de reconhecimento híbrido de objetos. Legenda: Cenário A - construção; Cenário B - refeição; Cenário C - leitura; Método MH - método holístico; Método CL - método baseado em caraterísticas locais.

As falhas na identificação podem ocorrer devido à incapacidade do sistema classificar um determinado objeto ou então confundi-lo, isto é, atribuir-lhe uma classificação errada. Como tal, para avaliar o nível de robustez individual de cada cenário foram construídas matrizes de confusão. Para construir as matrizes usaram-se as mesmas amostras utilizadas na análise de robustez da Tabela 3.3. Desta vez, apenas se consideram as classificações efetivas, i.e., quando o sistema consegue identificar o objeto, independentemente de ser bem ou mal classificado.

	Base	Coluna	Roda	Martelo
Base	**87,5%**	0%	12,5%	0%
Coluna	0%	**100%**	0%	0%
Roda	0%	0%	**100%**	0%
Martelo	0%	0%	0%	**100%**

Tabela 3.4: Matriz de confusão do Cenário A.

Como se verifica pela Tabela 3.4, pode haver confusão entre a roda e a base no Cenário A. O método holístico usado é invariante a escalamentos o que faz com que ambos os objetos sejam semelhantes, em termos de forma, segundo a interpretação do sistema. A única coisa que os distingue são as duas extremidades da base. Porém, quando as mesmas não são segmentadas conjuntamente com o resto do objeto, o sistema classifica como roda em vez de classificar como base.

	Copo	Garrafa	Prato	Chaleira
Copo	**100%**	0%	0%	0%
Garrafa	0%	**100%**	0%	0%
Prato	0%	0%	**100%**	0%
Chaleira	8,3%	0%	0%	**91,7%**

Tabela 3.5: Matriz de confusão do Cenário B.

A chaleira é um objeto que varia bastante de forma consoante a disposição. Isto é, aos "olhos" do robô, a forma do objeto segundo uma disposição horizontal difere da forma segundo uma disposição vertical (ver Figura 3.37). Para compensar este facto devem-se alargar as gamas de momentos para este tipo de objetos. Ao alargar as gamas podem-se gerar conflitos com outros objetos, como acontece, neste caso, com o copo (ver Tabela 3.5).

	Jornal A	Jornal B	Revista A	Revista B
Jornal A	**100%**	0%	0%	0%
Jornal B	0%	**100%**	0%	0%
Revista A	0%	0%	**100%**	0%
Revista B	0%	0%	0%	**100%**

Tabela 3.6: Matriz de confusão do Cenário C.

Dado que todos os objetos do Cenário C recorrem a uma abordagem baseada em caraterísticas locais, a taxa de confusão entre os mesmos é de 0%, como expetável (ver Tabela 3.6).

Para comprovar a validade do sistema implementado, criou-se um vídeo de demonstração (ver Figura 3.35) em que se apresentam os três cenários e diversas mudanças de posição e rotação dos diferentes objetos[13]. Seguidamente, apresentam-se algumas considerações acerca do vídeo:

- No Cenário A, apesar de estarem inseridos quatro objetos, também é feita a distinção entre colunas. Isto é, apesar das colunas terem diferentes cores, todas elas equivalem ao mesmo objeto segundo método de reconhecimento holístico. A sua diferenciação é feita posteriormente, recorrendo à deteção por cor restringida à área de segmentação do objeto detetado e identificado.

- Ainda em relação ao Cenário A, é possível constatar que se a base não estiver corretamente segmentada (como acontece aos 0:42 em que parte da base não é visível pelo sistema), o método holístico vai associar os contornos integrantes da base ao objeto "roda". Isto porque o sistema foi desenhado para considerar apenas os contornos exteriores fechados. Ora, se a base não estiver totalmente visível, contorno exterior vai estar aberto e, como tal, há uma procura dos contornos interiores à base.

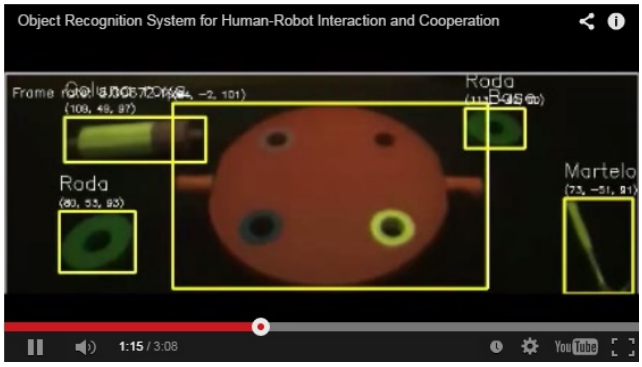
[13] Testes efetuados num *Intel® Core™ 2 Quad Q6600, 2.40 GHz, 2.0 GB RAM.*

- No Cenário B é dada uma especial relevância ao objeto "chaleira". Tal como referido, este é um objeto que pode aparentar diferentes formas. Foram testadas várias posições em diferentes rotações para comprovar a eficácia de reconhecimento.

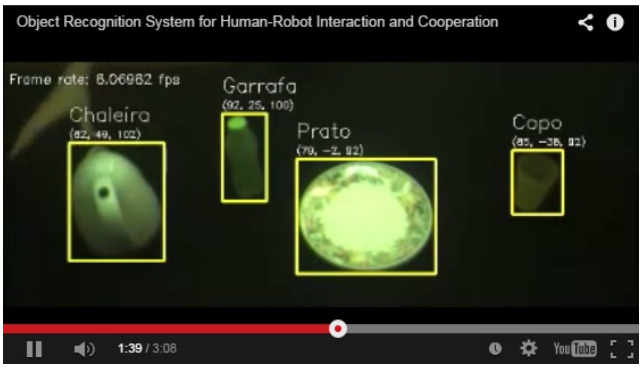
- Também no Cenário B, em relação ao prato, facilmente se constata a influência da luminosidade sobre o mesmo. Porém, dado terem sido adicionadas quatro fotos (ver secção 3.3.3), sistema consegue ultrapassar essa eventual limitação, apresentando uma boa robustez de identificação.

- No Cenário C constata-se uma queda no *frame rate* a cada nova revista/jornal colocada(o) (variando entre 9 fps, com um jornal apenas, e 5 fps, com quatro revistas e jornais). Esta queda é normal pois todos os objetos deste cenário recorrem à abordagem baseada em caraterísticas locais, mais custosa em termos de processamento.

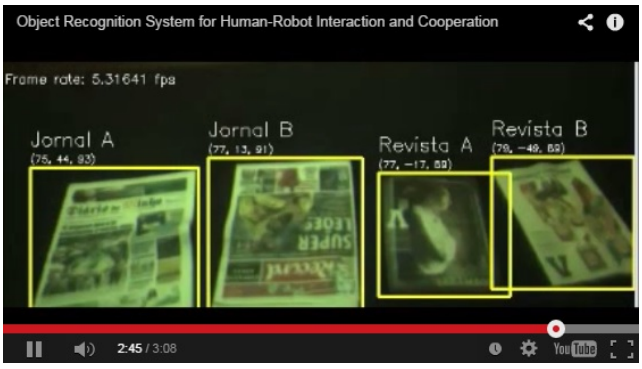
- No fim do vídeo (2:50) foram adicionadas duas outras revistas não incluídas na base de dados. Sistema ignora ambas, como expetável.



(a) Cenário A.



(b) Cenário B.



(c) Cenário C.

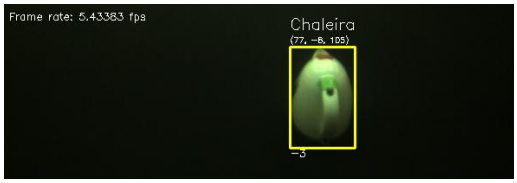
Figura 3.35: Demonstração do sistema de reconhecimento de objetos e respetivos *frame rates* (em fps). Visualizar vídeo em `http://marl.dei.uminho.pt/public/videos/objects.html`

3.3.2 Localização

Para comprovar e validar o sistema de localização espacial colocou-se um objeto, neste caso a chaleira (Figura 3.36), no ponto médio do robô em relação ao eixo horizontal (eixo dos yy, ver Figura 3.28). Um objeto colocado nessa posição deve apresentar valor $y = 0$, como se verifica pela Figura 3.36. Para comprovar o valor x, mediu-se a distância entre o centro do robô e a extremidade do objeto. A distância medida correspondeu a 77,5 centímetros. Em relação ao valor z, mediu-se a distância entre o chão e o ponto superior do objeto - cerca de 105 centímetros. Valor devolvido no eixo dos zz é o que apresenta maior erro, neste caso de 1 centímetro, aproximadamente. Os erros no sistema estereoscópico, e em particular no eixo dos zz, devem-se, essencialmente, ao facto do cálculo estar limitado à superfície visível do objeto. Esta limitação é mais significativa em objetos altos, como é o caso da chaleira.

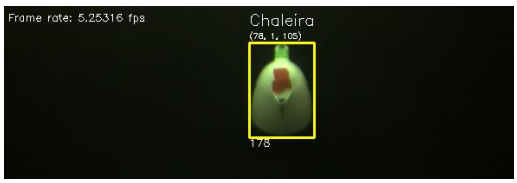
Figura 3.36: Validação do sistema de localização espacial.

Relativamente à orientação, tal como referido na secção 3.2.6.2, devem-se colocar duas marcas em lados opostos nos objetos cujos dados sobre a orientação são relevantes para o desempenho do sistema. Para o caso da chaleira, por exemplo, tendo em conta que o objeto não é simétrico, devem-se usar duas cores diferentes, permitindo indicar ao sistema qual é o lado da pega (ver Figura 3.37). Em objetos simétricos, as duas marcas podem ter a mesma cor.

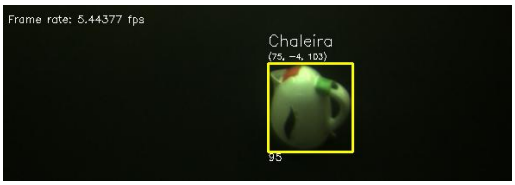


(a) Disposição vertical - pega virada para o robô (aproximadamente 0°).

(b) Disposição horizontal - pega virada para a esquerda (aproximadamente -90°).



(c) Disposição vertical - pega virada para o humano (aproximadamente 180°).



(d) Disposição horizontal - pega virada para a direita (aproximadamente 90°).

Figura 3.37: Cálculo da orientação para objeto não simétrico (resultado demonstrado no canto inferior esquerdo do retângulo que define o objeto).

3.3.3 Discussão dos resultados

Os resultados obtidos vão de encontro ao esperado. Ou seja, o método baseado em caraterísticas locais, mais exigente em termos de processamento, permite uma gama de possibilidades superior em relação ao método holístico, mais limitador em eficácia. Para o sistema híbrido funcionar de forma apropriada, deve ser encontrado um equilíbrio entre as duas metodologias de modo a atingir uma boa relação eficácia/tempo de resposta.

Os objetos identificados segundo método holístico estão sujeitos a uma maior probabilidade de confusão entre os mesmos. Isto porque as gamas de momentos atribuídas a cada um dos objetos por vezes têm de ser alargadas devido a dois fatores: a mudança da forma aos "olhos" do sistema consoante as diferenças de disposição de determinados objetos no plano de trabalho e a própria distorção provocada pelo facto do sistema "ver" o plano de trabalho segundo uma perspetiva oblíqua e não vertical. Estas são limitações decorrentes do facto de este ser um

ambiente controlado e não estruturado. Idealmente, para o reconhecimento de objetos seria necessária, pelo menos, uma outra câmara que incidisse sobre o plano de trabalho na vertical. Porém, este tipo de adaptações desvirtuariam o formato do robô antropomórfico na medida em que se pretende que os diversos constituintes estejam incorporados (*embodied*) no robô.

Os objetos identificados segundo a abordagem baseada em caraterísticas locais devem possuir caraterísticas passíveis de serem extraídas pelo sistema de visão. Ao contrário do método holístico, a forma não é relevante para o reconhecimento. Repare-se no caso das revistas (ou jornais), de igual forma, em que a classificação é independente da última. Porém, para os objetos serem classificados, deve ser gerado um número mínimo de correspondências (neste caso de quatro - ver secção 3.2.4), o que nem sempre acontece. Um dos fatores que contribui para que o sistema não consiga obter as correspondências necessárias consiste no facto de alguns dos objetos sofrerem uma influência maior das condições de luminosidade (é o caso do prato, mais brilhante). Para contornar esta limitação, devem-se adicionar ao sistema outras fotos do objeto sujeito a diferentes condições de luminosidade. No caso do prato, adicionaram-se quatro (quantas mais forem adicionadas, maior é o tempo de resposta). O *threshold* imposto para a classificação do objeto explica a menor taxa média de identificação (cerca de 89,6% face aos 94,3% relativos ao método holístico) associada à abordagem baseada em caraterísticas locais. Por outro lado, esse *threshold* também permite que a abordagem seja mais robusta a confusões, como se verifica pela Tabela 3.6.

Em relação aos objetos iguais[14], há também a possibilidade de colar diferentes padrões nos mesmos (usando o método baseado em caraterísticas locais para diferenciar os objetos), evitando, deste modo, a diferenciação por cor. A abordagem utilizada atualmente consiste em identificar os objetos semelhantes através dos

[14]Caso das colunas do cenário de construção conjunta, em que cada uma tem uma cor diferente para serem inseridas em sítios também diferentes.

momentos e fazer a sua diferenciação por cor. Neste caso, a procura por cor restringe-se ao objeto em si, não estando dependente das condições do plano de trabalho. Porém, estas são situações específicas que devem ser consideradas individualmente consoante as necessidades reveladas. Uma das vantagens do sistema de reconhecimento genérico implementado, consiste, precisamente, no facto de se poderem inserir diferentes objetos no sistema sem que sejam necessárias adaptações de relevo quer no objeto, quer no programa (exceção feita ao cálculo da orientação para os objetos que o requeiram).

Esta página foi intencionalmente deixada em branco!

Capítulo 4

Reconhecimento de Gestos

Grande parte da comunicação entre os humanos é não verbal. Os humanos conseguem facilmente percecionar e até antecipar o objetivo das ações dos outros através da interpretação da combinação de gestos e movimentos com o contexto da tarefa (Sebanz et al., 2006). A preponderância da comunicação não verbal na comunicação entre os humanos é tal que, é possível, e até habitual, executarmos as mais diversas e elementares tarefas do dia-a-dia com parceiros humanos sem trocarmos palavras (Bicho et al., 2010).

Na perspetiva de dotar o ARoS com capacidades cognitivas que lhe permitam executar de uma maneira natural e eficiente as diferentes tarefas em contextos de interação social, é fundamental que o mesmo consiga interpretar gestos de parceiros humanos. A interpretação de gestos permite ao robô fazer uma predição do comportamento do humano e assim decidir qual o melhor comportamento a adotar (Newman-Norlund et al., 2007). A título de exemplo, a maneira como o humano agarra uma garrafa pode fornecer indicações acerca das suas intenções. Isto é, se agarrar de lado, poderá significar que ele mesmo pretende utilizá-la. Ao agarrar por cima, há uma grande probabilidade de a intenção consistir em passá-la ao robô. Neste caso, o robô, prevendo a intenção do humano, pode preparar-se para receber a garrafa, sem que seja necessária qualquer informação verbal no

processo.

4.1 Estado da arte

O processo de interpretação de gestos atravessa três fases distintas: deteção da mão, extração de caraterísticas da mesma e respetiva classificação. Apresentam-se, no decorrer desta secção, algumas das abordagens mais utilizadas.

4.1.1 Deteção da mão

A deteção da mão e consequente segmentação da mesma permite isolar a informação relevante de todo o resto da imagem. Várias abordagens têm sido usadas neste passo, destacando-se aquelas baseadas na deteção por cor, forma, modelos 3D e movimento (Rautaray and Agrawal, 2012).

Relativamente à deteção por cor, utilizam-se espaços de cor como o RGB, o HSV, o YCrCb e o YUV no processo de deteção da cor da pele. Destes, o RGB não é tão usado devido ao facto de não separar a informação cromática da de luminosidade (ver secção 4.2.2). Os métodos baseados nos espaços de cores apresentam algumas limitações derivadas do facto do ser humano ter diferentes tons de pele (mesmo entre pessoas da mesma raça). Para contrariar esta situação, devem ser aplicadas compensações, como em Sigal et al. (2004), em que se propõe uma representação da cor da pele invariante a mudanças de luminosidade. Porém, esta solução é bastante sensível a mudanças bruscas de luminosidade e condições de luminosidade inconstantes (Rautaray and Agrawal, 2012). Existe ainda o problema da deteção ser dificultada pelo plano de fundo, no caso de haver alguma região da imagem com cor semelhante à cor da pele do humano. Para contornar essa limitação, pode-se usar a subtração de fundo (ver algumas das técnicas mais usadas em Piccardi (2004)). Porém, a subtração de fundo exige que o sistema de visão não se mova em relação ao plano de trabalho, o que nem sempre se verifica (Rautaray

and Agrawal, 2012).

Dada a forma caraterística da mão, o uso de métodos baseados na forma é uma das abordagens possíveis. Destacam-se pelo facto de não dependerem da luminosidade, cor da pele ou ponto de vista. Por outro lado, dependem de uma plena visão da mão. Isto é, basta a mão estar ligeiramente tapada ou mal segmentada para a sua deteção ficar comprometida. Como tal, os métodos baseados na forma são normalmente complementados por outras abordagens. Em Belongie et al. (2002) é proposto um método complementado por caraterísticas locais. Este método defende que se duas formas distintas tiverem pontos correspondentes, as mesmas terão uma forma contextual semelhante (Rautaray and Agrawal, 2012). Algumas abordagens (Yin and Xie, 2003; Argyros and Lourakis, 2006) usam informação estereoscópica para detetar a posição espacial das pontas dos dedos e, a partir daí, fazer a reconstrução 3D da mão. Porém, estes métodos falham no caso de as pontas dos dedos estarem tapadas pelo resto da mão. Como possível solução, podem ser utilizadas múltiplas câmaras (Kunii and Lee, 1995).

Os métodos baseados em modelos 3D têm a vantagem de conseguir fazer a deteção de um mesmo gesto sujeito a diferentes pontos de vista (Rautaray and Agrawal, 2012). A deteção é feita através da associação de correspondências entre o modelo criado e a imagem de referência.

Os métodos baseados no movimento são os menos usados. Isto porque exigem um ambiente bastante controlado. Assumem que apenas a mão se pode mover (Cutler and Turk, 1998). Como tal, estes métodos são normalmente acompanhados por outras abordagens como aplicação de filtros para diferenciação da imagem e/ou subtração de fundo (Martin et al., 1998).

4.1.2 Extração de caraterísticas da mão

Algumas das caraterísticas mais usadas nos métodos de reconhecimento consistem em caraterísticas geométricas como pontas dos dedos, direção dos mesmos, con-

tornos das mãos, entre outras. Por outro lado, caraterísticas sob forma de cor ou textura não são viáveis no processo de classificação e, portanto, não se justifica a sua extração. Apresentam-se, de seguida, as três abordagens mais comuns, segundo Murthy and Jadon (2009), no processo de extração de caraterísticas da mão.

4.1.2.1 Abordagens baseadas em modelos

Estas abordagens estimam a posição da palma da mão e respetivos ângulos das juntas dos dedos. Ou seja, é feita uma procura de parâmetros cinemáticos que aproximem uma projeção bidimensional de um modelo tridimensional da mão a uma imagem baseada em contornos (Stenger et al., 2001). Um dos maiores problemas deste tipo de abordagens consiste na obtenção dos contornos internos da mão, difíceis de obter.

De modo a facilitar a extração de caraterísticas, a imagem de fundo deve contrastar significativamente com a mão. Deste modo é mais fácil obter os contornos internos, aplicando um *threshold* mais sensível.

4.1.2.2 Abordagens baseadas na aparência

As abordagens baseadas na aparência modelam a mão a partir de um conjunto de imagens 2D da mesma (diferentes condições de iluminação e pontos de vista).

4.1.2.3 Abordagens baseadas em caraterísticas de baixo nível

Em grande parte das aplicações de reconhecimento gestual é desnecessário fazer uma modelação completa da mão. Sendo assim, é possível fazer apenas a extração de caraterísticas de baixo nível, tornando a aplicação significativamente mais rápida. Destacam-se as caraterísticas geométricas como o centroide (New et al., 2003) e a forma (Otiniano-Rodríguez et al., 2012).

4.1.3 Classificação

Apresentam-se, seguidamente, duas abordagens que se distinguem no processo de classificação gestual (Murthy and Jadon, 2009).

4.1.3.1 Abordagens baseadas em regras

As abordagens baseadas em regras consistem na comparação das caraterísticas obtidas com um conjunto de regras introduzidas manualmente. A identificação é efetuada no caso de determinado gesto se inserir num conjunto de regras relativas a um dos gestos implementados no sistema.

4.1.3.2 Abordagens baseadas em *Machine Learning*

Este tipo de abordagens considera o gesto como *output* de um processo estocástico (Murthy and Jadon, 2009). Uma das técnicas mais usadas na classificação de gestos baseada em *Machine Learning* consiste nos *Hidden Markov Models* (Kim, 1999; Wilson and Bobick, 1999).

4.1.4 Discussão do estado da arte

Tal como em relação aos objetos, o objetivo principal da implementação do sistema de reconhecimento gestual assenta em dois pontos chave: a eficácia do algoritmo de reconhecimento e a rapidez de processamento. São estes os dois fatores essenciais e que não devem ser dissociados sob pena de comprometerem a eficiência de uma comunicação em tempo real entre o humano e o robô. Nesse sentido, em relação à deteção da mão, uma abordagem baseada na cor da pele e de um elemento auxiliar (pulseira, por exemplo) de modo a limitar a região de deteção, é a que melhor se ajusta aos propósitos do sistema, tanto em eficácia como em tempo de resposta. Já em relação à extração de caraterísticas, tendo em conta o conjunto limitado de gestos que se pretende implementar, uma abordagem

baseada em caraterísticas de baixo nível como a forma (através dos momentos invariantes de *Hu*, por exemplo), é suficiente e mesmo aconselhável dada a rapidez de processamento subjacente a esta abordagem. Já em relação à classificação, seguindo a mesma linha de raciocínio, uma abordagem baseada em regras é a que melhor se adequa. Até porque abordagens baseadas em *Machine Learning* ajustam-se no caso da implementação de um sistema com bastantes gestos e com semelhanças entre os mesmos. Para além disso, o uso de classificadores holísticos baseados em *Machine Learning* numa camada de baixo nível é desaconselhado. Por esse mesmo motivo, tal como nos objetos, os métodos de extração de caraterísticas da mão baseados em aparência são também descartados na medida em que estes implicam a aplicação de um classificador.

Sintetizando, pretende-se implementar um sistema que faça a deteção da mão por cor, extraia as caraterísticas sob forma de momentos invariantes e classifique os gestos através de uma abordagem baseada em regras.

4.2 Implementação

Ao contrário dos objetos, neste caso há uma procura específica. Isto é, já se sabe aquilo que se quer detetar. Nesse sentido, optou-se pela utilização de um utensílio auxiliar (uma pulseira) no processo de deteção da mão. O diagrama da Figura 4.1 demonstra os passos da implementação da interpretação de gestos. Foram implementados três gestos para validação (os mais relevantes no processamento das camadas de alto nível do sistema), sendo possível adicionar outros facilmente (desde que difiram), consoante a necessidade.

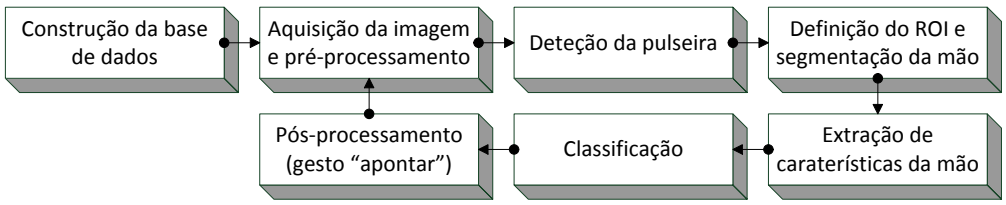


Figura 4.1: Etapas do sistema de reconhecimento de gestos.

Tal como nos objetos, em primeiro lugar, e antes de entrar no ciclo, deve-se construir a base de dados. Neste caso, composta por vetores de momentos invariantes de *Hu*. Nomeadamente três, relativos aos gestos "agarrar por cima", "agarrar de lado" e "apontar". O gesto "apontar" necessita de um pós-processamento, visto que a este gesto está associada uma intenção explícita de localizar ou obter algo. Como tal, o sistema devolve o objeto para o qual o humano aponta.

Segue-se a explicação de cada um dos passos do ciclo do diagrama da Figura 4.1.

4.2.1 Aquisição da imagem e pré-processamento

Tal como na secção 3.2.1, após a aquisição da imagem é necessário fazer um redimensionamento. O redimensionamento não deve ser muito restrito visto que o sistema de visão deve abranger parte do braço do humano (ver Figura 4.2).

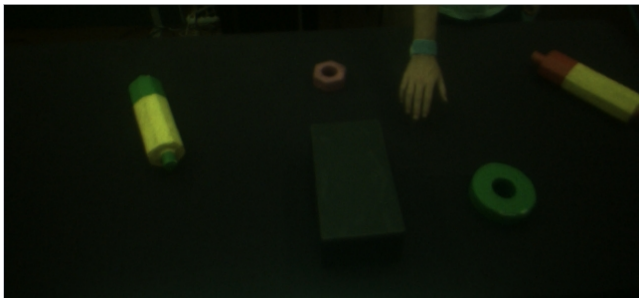


Figura 4.2: Redimensionamento da janela de visualização de modo a abranger a mão e o plano de trabalho.

4.2.2 Deteção da pulseira

A deteção da pulseira é feita por cor. Sendo assim, em primeiro lugar é necessário converter a imagem RGB para o espaço de cores HSV[1] (*Hue*, *Saturation*, *Value*). Começa-se, então, por definir o máximo e o mínimo dos valores RGB expressos entre 0 e 1[2], de modo a calcular a diferença Δ entre os mesmos:

$$máximo = max(R, G, B) \tag{4.1}$$

$$mínimo = min(R, G, B) \tag{4.2}$$

$$\Delta = máximo - mínimo \tag{4.3}$$

Com base nos valores obtidos, calculam-se os valores de H (*Hue*), S (*Saturation*) e V (*Value*):

$$H = \begin{cases} 60 \times \frac{G-B}{\Delta} & máximo = R \text{ e } G \geq B \\ 60 \times \frac{G-B}{\Delta} + 360 & máximo = R \text{ e } G < B \\ 60 \times \frac{B-R}{\Delta} + 120 & máximo = G \\ 60 \times \frac{R-G}{\Delta} + 240 & máximo = B \end{cases} \tag{4.4}$$

$$S = \begin{cases} 0 & \Delta = 0 \\ \frac{\Delta}{máximo} & \Delta \neq 0 \end{cases} \tag{4.5}$$

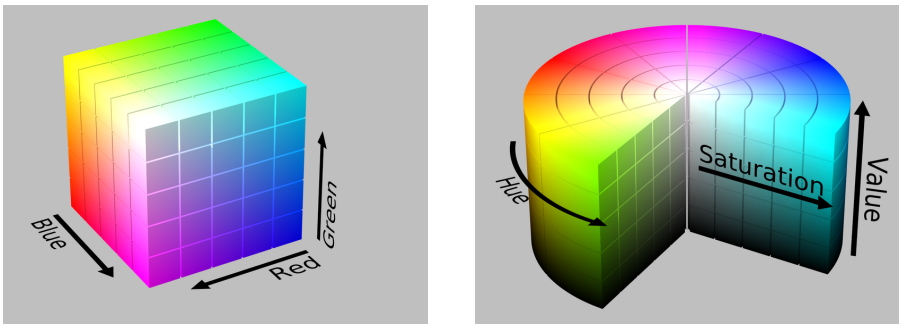
$$V = máximo \tag{4.6}$$

H varia entre 0° e 360°, S e V entre 0 e 1.

[1] Também conhecido por HSB (*Hue*, *Saturation*, *Brightness*).

[2] Deve-se dividir os valores puros RGB por 255.

Esta conversão é necessária porque o modelo RGB (Figura 4.3a) não faz a separação da informação cromática da informação de luminosidade. Em termos práticos, torna-se difícil detetar uma mesma cor com diferentes intensidades de luz. Já o modelo HSV (Figura 4.3b) é mais parecido com o modelo humano, conseguindo identificar uma mesma cor sujeita a diferentes luminosidades (Li et al., 2002). Neste caso, a informação de *Hue* (tonalidade) e *Saturation* (saturação) é independente da informação de *Value* (luminosidade).



(a) Modelo de cores RGB. **(b)** Modelo de cores HSV.

Figura 4.3: Sólidos representativos dos modelos de cores (imagens retiradas de `http://en.wikipedia.org/wiki/HSL_and_HSV` [acedido em 2014-04-10]).

Após a deteção da cor (em tons de ciano), resultado é armazenado numa imagem binária em que *pixéis* brancos correspondem à região da cor pretendida (ver Figura 4.4a). Precavendo eventuais ruídos na imagem, são aplicados os métodos *Erode* e *Dilate*. Ou seja, em primeiro cria-se uma erosão na imagem de modo a limpar todo o ruído que possa existir. De seguida é necessário dilatar a imagem de modo a recuperar a região da pulseira no caso da mesma ter sido deturpada pela erosão (ver Figura 4.4b).

(a) Imagem binária antes da aplicação dos métodos *Erode* e *Dilate*.

(b) Imagem binária depois da aplicação dos métodos *Erode* e *Dilate*.

Figura 4.4: Imagens binárias resultantes da deteção por cor.

Como se pode constatar pela Figura 4.4b, pode acontecer de nem todo o ruído ser "limpo". Neste caso concreto, devido à cor da camisola ser semelhante à da pulseira (ver Figura 4.2), a limpeza fica dificultada. Para salvaguardar estas situações, considera-se apenas a maior região binária em termos de área, que deverá corresponder à da pulseira. Só no caso da cor da camisola ser exatamente igual à da pulseira é que a deteção pode falhar. Nessa eventualidade, deve-se limitar a região de maneira a que o sistema de visão apenas capture a mão e uma pequena parte do antebraço.

4.2.3 Definição do ROI e segmentação da mão

O ROI da mão deve contemplar um espaço abrangente de modo a poder caber qualquer mão, independentemente da dimensão. Sendo assim, a partir da posição da pulseira é definida uma região (ver Figura 4.5) à qual o processamento seguinte se irá restringir.

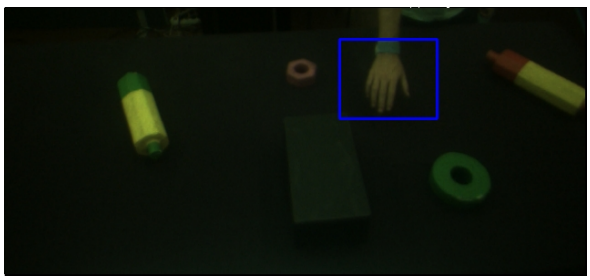


Figura 4.5: ROI da mão (retângulo azul).

Para fazer uma análise das caraterísticas da mão (ver secção 4.2.4), é necessário, em primeiro, segmentá-la, i.e., restringir ainda mais a região. Para tal, utiliza-se, novamente, a deteção por cor. Neste caso pretende-se detetar a cor da pele do humano. Ora, como a cor da pele não é uma caraterística fixa, estando dependente da pessoa que interage com o robô, o sistema permite, de uma maneira *user-friendly*, a definição dos diferentes parâmetros de deteção através de *trackbars* numa janela de configurações (Figura 4.6). De referir que o ajuste dos parâmetros pode ser efetuado em qualquer altura, mesmo durante a execução do programa.

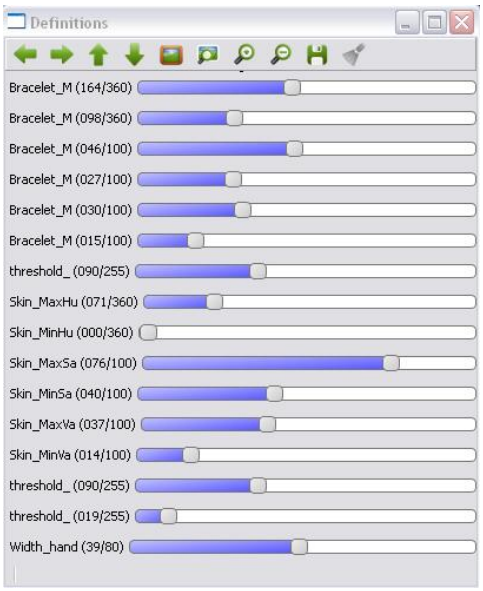


Figura 4.6: Janela de configurações.

4.2.4 Extração de caraterísticas da mão

Após obtenção da imagem binária resultante da deteção da cor da pele (Figura 4.7), obtêm-se os momentos invariantes de *Hu* correspondentes a essa representação.

Figura 4.7: Representação binária da mão.

Após alguns testes preliminares constatou-se que dois dos gestos ("agarrar de lado" e "apontar") podem ter momentos semelhantes nalgumas posições. Para evitar ambiguidades, a largura da mão também é extraída. Para tal, recorre-se à *Convex Hull* (Graham, 1972). A partir da última é possível obter uma representação da mão baseada nas suas extremidades, como se verifica pela Figura 4.8. Como a mão não está, necessariamente, em posição vertical relativamente ao sistema de visão, é preciso criar um retângulo rodado de modo a poder fazer os cálculos associados à largura da mão.

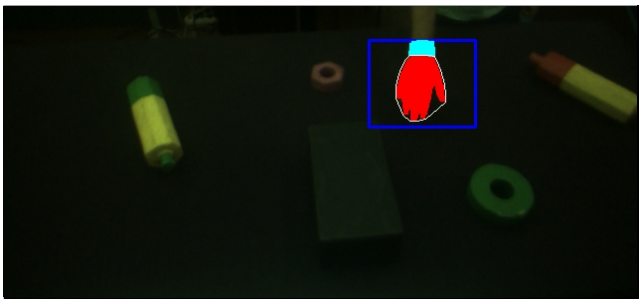


Figura 4.8: Extração da mão (a vermelho) a partir da deteção da região retangular definida pela pulseira (a azul) e respetiva representação através da *Convex Hull* (linha branca).

4.2.5 Classificação

Nesta fase é feita a comparação dos momentos obtidos na secção 4.2.4 com os momentos previamente guardados, correspondentes aos três gestos analisados. Como já foi referido, dois dos gestos apresentam momentos semelhantes em

determinadas posições, derivado da parecença entre os mesmos. Os gestos em causa são o "apontar" e o "agarrar de lado". Como neste último a pessoa tem tendência a alargar a mão, a distinção entre os dois faz-se por aí. Ou seja, caso a largura ultrapasse um determinado *threshold* ajustável (também na janela de configurações, à semelhança dos parâmetros das cores) a diferentes mãos, gesto identificado corresponde ao "agarrar de lado". Caso contrário, gesto identificado corresponde ao "apontar" (ver Figura 4.9).

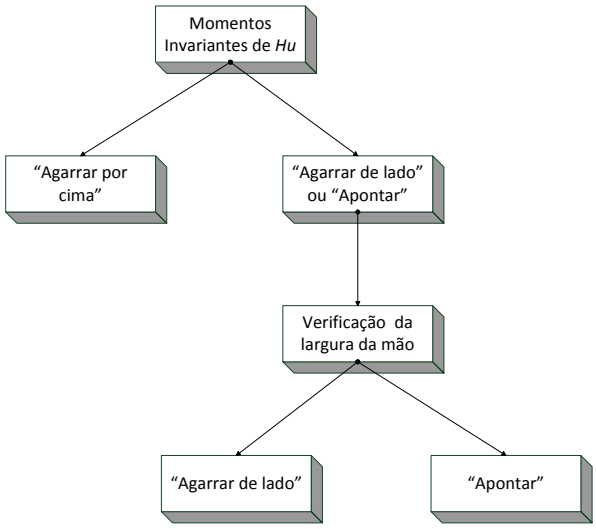


Figura 4.9: Diagrama do processo de classificação gestual.

(a) Agarrar por cima. **(b)** Agarrar de lado. **(c)** Apontar.

Figura 4.10: Classificação dos três gestos implementados.

4.2.6 Pós-processamento (gesto "apontar")

Caso o gesto identificado consista no "apontar", sistema entra num pós-processamento no intuito de "saber" qual é o objeto apontado. Se o gesto identificado for algum dos outros dois, este passo é ignorado, entrando novamente no ciclo através da aquisição de outra *frame*.

Em primeiro lugar, obtêm-se as coordenadas da extremidade do dedo indicador. Para tal, utiliza-se o contorno obtido através da *Convex Hull* na secção 4.2.4. Ou seja, percorre-se todo o contorno e armazena-se o ponto cujo valor y seja superior[3] (Figura 4.11).

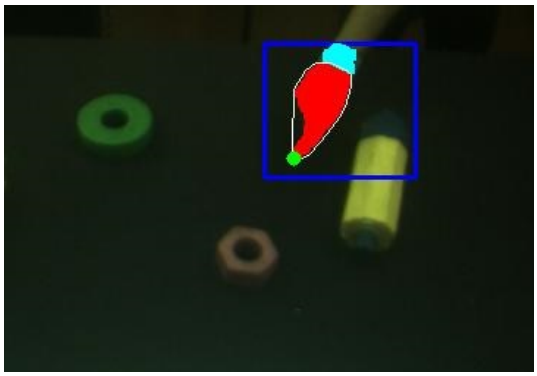


Figura 4.11: Ponto superior assinalado a verde.

Para traçar uma linha que indique qual a direção para a qual o humano aponta, é necessário adquirir um outro ponto. Assim sendo, desenha-se um retângulo que contenha a pulseira e obtém-se o ponto central da mesma dividindo a largura por dois (Figura 4.12).

[3] O eixo dos yy começa na parte superior da janela de visualização (ver Figura 3.28).

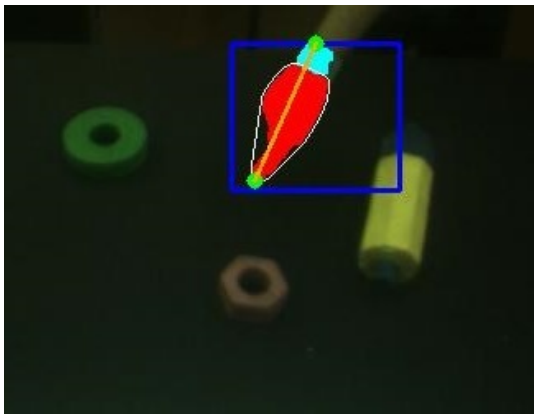


Figura 4.12: Linha entre pulseira e indicador.

Por último, é necessário aumentar a linha de modo a abranger todo o plano de trabalho. Utilizam-se os dois pontos obtidos para encontrar o m e o b da equação da reta $y = mx + b$:

$$m = \frac{y_{indicador} - y_{pulseira}}{x_{indicador} - x_{pulseira}} \tag{4.7}$$

$$b = y_{indicador} - mx_{indicador} \tag{4.8}$$

Tendo a equação da reta definida, para desenhar a linha que contempla todo o plano de trabalho é necessário obter a coordenada x correspondente ao ponto cujo y equivale ao limite da janela de visualização:

$$x_{limite} = \frac{y_{limite} - b}{m} \tag{4.9}$$

Desenhando uma linha entre o ponto correspondente ao indicador e o ponto limite da janela de visualização, obtém-se o resultado da Figura 4.13.

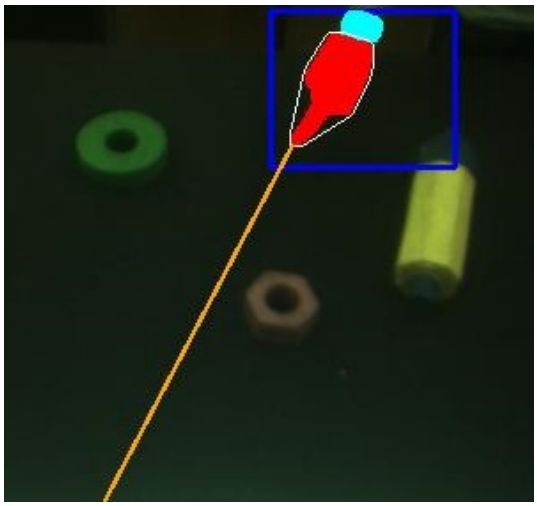


Figura 4.13: Linha final.

Para detetar os objetos do plano de trabalho é necessário fazer uma segmentação dos mesmos, à semelhança do apresentado na secção 3.2.2. Após esta segmentação, calcula-se a menor distância entre o centroide de cada um dos objetos detetados (obtido através do cálculo dos momentos centrais - ver secção 3.1.1.3) e a linha. Essa distância corresponde ao comprimento do segmento de reta, perpendicular à linha, que une o ponto e a linha. Para efetuar este cálculo recorre-se à fórmula (4.10), baseada nas coordenadas cartesianas:

$$distância = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}} \tag{4.10}$$

Fazendo a associação com a equação da reta, obtém-se $a = -m$, $b = 1$ e $c = -b$. x_0 e y_0 correspondem às coordenadas do centroide do objeto. Simplificando (4.10), chega-se a (4.11):

$$distância = \frac{|-mx_0 + y_0 - b|}{\sqrt{m^2 + 1}} \tag{4.11}$$

Aplicando (4.11) a cada um dos centroides dos objetos detetados, menor distância calculada vai corresponder ao objeto pretendido, como se verifica pela Figura 4.14.

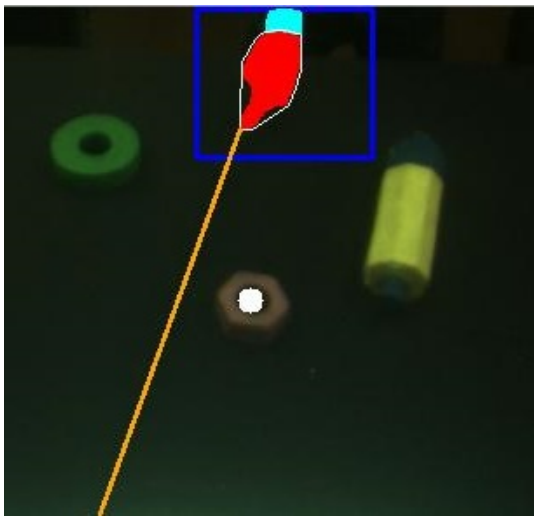


Figura 4.14: Objeto mais próximo da linha assinalado a branco.

4.3 Resultados

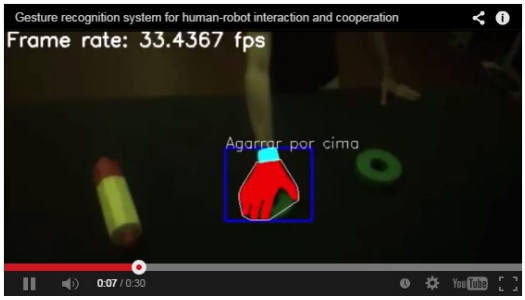
Nesta secção apresentam-se os resultados obtidos após a implementação do reconhecimento dos três gestos: "agarrar por cima", "agarrar de lado" e "apontar". O sistema, além de identificar o gesto, devolve o respetivo *frame rate*. De salientar que o *frame rate* do gesto "apontar", na ordem dos 10-12 fps é significativamente inferior ao dos gestos "agarrar por cima" e "agarrar de lado", com cerca de 33-35 fps[4]. Esta situação explica-se pelo facto do gesto "apontar" incluir um pós-processamento, ao contrário dos outros dois gestos.

Estando os parâmetros bem definidos (deteção de cor e *threshold* da largura da mão), sistema tem uma eficácia de 100%. Ou seja, identifica os três gestos em qualquer uma das posições do plano de trabalho, sem confusão de gestos.

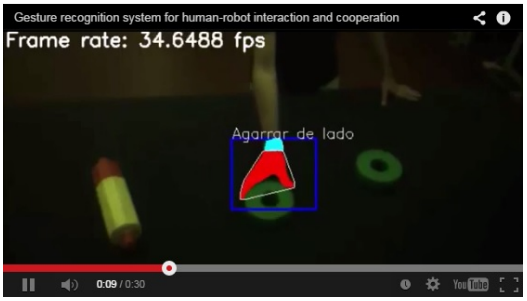
A partir do vídeo apresentado (Figura 4.15) são demonstrados os três gestos implementados e também os resultados do pós-processamento do gesto "apontar" (a linha e o centroide do objeto mais próximo). Foi também incluído um gesto "intruso", neste caso a mão fechada (ver aos 0:18), que não devolve quaisquer

[4]Testes efetuados num *Intel® Core™ 2 Quad Q6600, 2.40 GHz, 2.0 GB RAM*.

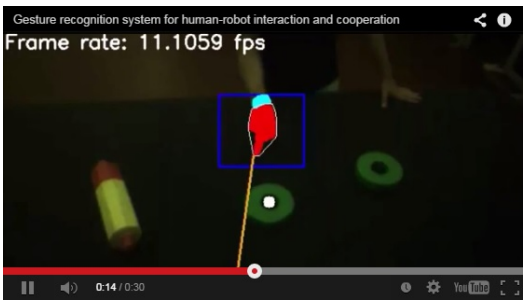
resultados. Na transição de gestos (ver aos 0:09 ou 0:22), pode acontecer de aparecer o gesto "agarrar por cima". Isto porque este gesto também corresponde à disposição normal (descontraída) da mão.



(a) Agarrar por cima.



(b) Agarrar de lado.



(c) Apontar.

Figura 4.15: Classificação dos três gestos implementados e respetivos *frame rates* (em fps). Visualizar vídeo em `http://marl.dei.uminho.pt/public/videos/gestures.html`

O desempenho do sistema de reconhecimento de gestos implementado é

significativamente melhor em relação ao do sistema usado anteriormente (Westphal et al., 2008) no Laboratório de Robótica Móvel e Antropomórfica. Este último apresentava velocidades de processamento de cerca de 0,5 fps (com uma máquina dedicada[5]). Ou seja, o novo sistema, construído de raiz, é cerca de 66 vezes mais rápido para os gestos "agarrar por cima" e "agarrar de lado" e 22 vezes mais rápido para o gesto "apontar".

[5] *Intel® Core™ 2 Duo E6850, 3 GHz, 2.0 GB RAM.*

Esta página foi intencionalmente deixada em branco!

Capítulo 5

Reconhecimento de Expressões Faciais

Em 1978, Paul Ekman e Wallace Friesen publicaram (Ekman and Friesen, 1978) o FACS (*Facial Action Coding System*). O FACS consiste num sistema para categorizar movimentos faciais humanos (alguns exemplos na Tabela 5.1). Decompõe praticamente todas as expressões faciais possíveis em AUs (*Action Units*). A deteção e interpretação de AUs permite associar as expressões faciais a determinada emoção. Por exemplo, a conjugação das AU6 (subir bochecha) e AU12 (subir canto do lábio) pode indicar a presença da emoção "felicidade". Nesta dissertação são apenas consideradas as AUs fundamentais no processo de identificação das seis emoções básicas universais definidas por Ekman et al. (1972), i.e., aquelas que são comuns a todas as culturas: aversão, felicidade, medo, raiva, surpresa e tristeza.

Um dos grandes problemas neste tipo de sistemas consiste na variabilidade inerente às faces de diferentes pessoas. Todos temos traços faciais distintos, o que dificulta o processo de deteção de AUs por um sistema de visão computorizado. Como tal, é necessário efetuar uma análise de robustez através de testes com bases de dados específicas para o efeito e realizando experiências com pessoas.

AU	Descrição
1	Levantar parte interior da sobrancelha
2	Levantar parte exterior da sobrancelha
4	Baixar sobrancelha
5	Levantar pálpebra
6	Subir bochecha
7	Apertar pálpebra
9	Enrugar nariz
12	Levantar canto do lábio
15	Baixar canto do lábio
16	Baixar lábio inferior
20	Esticar lábio
23	Apertar lábio
26	Cair queixo

Tabela 5.1: Principais AUs usadas no processo de identificação das seis emoções básicas definidas por Ekman and Friesen (1978).

Conseguindo uma deteção robusta de AUs, é possível o robô aferir o estado emocional do humano e agir em concordância com o mesmo, podendo, eventualmente, alterar o rumo da ação que vinha a efetuar. Por exemplo, numa tarefa de construção conjunta, a identificação do estado emocional "medo" pode indicar que algo de inesperado aconteceu. Sendo assim, o robô deve agir em função disso, seja através de uma verificação do plano de trabalho ou mesmo através da interrupção da atividade que vinha a executar.

A interpretação de AUs segundo as emoções básicas está contemplada em dois sistemas: o EMFACS (*Emotional Facial Action Coding System*) e o FACSAID (*Facial Action Coding System Affect Interpretation Dictionary*)[1]. Estes dois sistemas

[1] `http://www.face-and-emotion.com/dataface/facs/emfacs.jsp` [acedido em 2014-07-19].

apenas consideram as AUs passíveis de associação a determinada emoção.

Emoção	Combinação de AUs
Aversão	9+15+16
Felicidade	6+12
Medo	1+2+4+5+7+20+26
Raiva	4+5+7+23
Surpresa	1+2+5B+26
Tristeza	1+4+15

Tabela 5.2: Exemplo de combinações prototípicas para associação a determinada emoção (AU5B - B corresponde à intensidade).

Em aplicações informáticas de deteção e interpretação de AUs, a utilização das combinações prototípicas pode tornar-se complicada devido ao facto de haver AUs de difícil deteção. Existem variantes (Ekman et al., 2002; Lucey et al., 2010) introduzidas mais recentemente que defendem algumas alterações em relação às combinações prototípicas. Por exemplo, em vez da combinação de várias AUs, estas variantes propõe que a emoção poderá ser identificada através da deteção de um conjunto mais restrito de AUs, desde que nesse conjunto não constem determinadas AUs incompatíveis. Sendo assim, estes sistemas funcionam como referência, cabendo aos programadores a escolha das AUs mais adequadas e que garantem uma maior taxa de deteção.

5.1 Estado da arte

O reconhecimento de expressões faciais assenta em três passos fundamentais: a deteção da face, a extração de caraterísticas da mesma e a respetiva classificação, identificando as AUs de modo a poder interpretá-las na perspetiva de aferir o estado emocional. Apresentam-se, no decorrer desta secção, alguns dos métodos mais

usados na execução destas etapas.

5.1.1 Deteção da face

Existem vários métodos baseados em diferentes abordagens (Adeshina et al., 2009) (movimento, aparência, cor, textura, cantos e modelos deformáveis) para efetuar a deteção da face.

Nos métodos baseados em textura (Wang and Wang, 2002), aproveita-se o facto de existirem elementos faciais distintos, como é o caso dos lábios e sobrancelhas. A partir da deteção destes elementos, os restantes são extraídos através do conhecimento das caraterísticas da face.

Em relação aos métodos baseados em cantos (Suzuki and Shibata, 2004), os vetores de caraterísticas da face são gerados recorrendo a uma distribuição dos cantos da imagem. Apesar dos cantos da face serem fáceis de detetar, se o fundo da imagem for mais complexo, a robustez do algoritmo não é assegurada.

Os métodos baseados em cor (Liu et al., 2005) fazem a deteção através de cor e profundidade.

Os métodos baseados em movimento (úteis para sequências de imagens ou vídeo) baseiam-se no levantamento das caraterísticas móveis da imagem num determinado intervalo de tempo. Por exemplo, em Espinosa-Duro et al. (2004), o método proposto para a deteção da face usa a reflexão da luz pela face humana quando iluminada com uma fonte de luz fraca.

Os métodos baseados em aparência são os mais usados. São mais robustos mas exigem maior processamento, o que pode ser uma limitação em aplicações de tempo real. Por exemplo, segundo Viola and Jones (2004) o processo baseia-se em três etapas fundamentais: o uso de imagens integrais para acelerar a computação de caraterísticas da imagem, a utilização de um classificador (baseado no algoritmo de aprendizagem *AdaBoost*) para escolha das caraterísticas mais relevantes dentro de uma gama potencial e a combinação dos classificadores em cascata, permitindo

que regiões de fundo possam ser eliminadas e o processamento se restrinja à face unicamente.

Os métodos baseados em modelos deformáveis são usados recorrendo a modelos como o AAM (*Active Appearance Model*) (Edwards et al., 1998) e o ASM (*Active Shape Model*) (Cootes et al., 1995). Este tipo de modelos permite representar as variações na forma e na intensidade da textura de elementos da face. Segundo Cootes et al. (1999), o ASM é mais rápido e robusto mas o AAM representa melhor a textura.

5.1.2 Extração de caraterísticas da face

Existem dois tipos de métodos para extração de caraterísticas da face (Chitra and Balakrishnan, 2012): métodos lineares e não-lineares. Os métodos lineares transformam os dados de um sub-espaço de alto nível dimensional para um de baixo nível (também chamada de redução dimensional) através de mapeamento linear. Os métodos não-lineares atingem o mesmo objetivo mas através de mapeamento não-linear. Em termos práticos, a diferença consiste no facto dos métodos não-lineares serem mais robustos a inclusões de cabelo, mudanças de luminosidade, entre outros.

5.1.2.1 Métodos lineares

- PCA - *Principal Component Analysis* (Pearson, 1901; Hotelling, 1933): trata-se de uma abordagem holística em que toda a região da face é usada no processo. O algoritmo pode ser usado para encontrar sub-espaços cujos vetores de base correspondem às direções de máxima variância no espaço dimensional original.

- LDA - *Linear Discriminant Analysis* (Fisher, 1936): algoritmo que procura o conjunto de vetores que fornece a melhor discriminação entre as classes,

maximizando as diferenças entre classes e minimizando as diferenças dentro das classes. Usa, para tal, matrizes de dispersão.

- SVD - *Singular Value Decomposition* (Eckart and Young, 1936; Golub and Kahan, 1965; Golub and Reinsch, 1970): faz a redução dimensional através da procura de combinações ortogonais lineares das variáveis originais com maior variância (Murthy and Natarajan, 2011). É considerada a melhor técnica baseada na matriz de covariância.

5.1.2.2 Métodos não-lineares

- KPCA - *Kernel Principal Component Analysis* (Schölkopf et al., 1998): reformulação do PCA. O KPCA calcula os principais vetores próprios da matriz *kernel* (*kernel* de uma matriz A é o conjunto de todos os vetores x para os quais $Ax = 0$) em vez dos da matriz de covariância.

- *Isomap* - (Tenenbaum et al., 2000): técnica que preserva a distância curvilínea entre pontos (distância geodésica). Para pontos vizinhos, a distância euclidiana do espaço de entrada fornece uma boa aproximação à distância geodésica. Para pontos distantes, a distância geodésica pode ser calculada fazendo uma sequência de "saltos" entre pontos vizinhos.

- SNE - *Stochastic Neighbor Embedding* (Hinton and Roweis, 2002): abordagem probabilística que mapeia informação de alta dimensão num sub-espaço de baixa dimensão preservando as distâncias relativas entre pontos. É usada uma distribuição Gaussiana centrada num ponto no espaço de alta dimensão para definir a distribuição provável que o ponto "escolhe" para os seus vizinhos.

5.1.3 Classificação

Para esta fase importa referir que um classificador só produz bons resultados se as caraterísticas extraídas forem adequadas. Isto é, o desempenho do classificador está dependente da eficiência do extrator.

Existem dois tipos fundamentais (Chibelushi and Bourel, 2002) de classes no reconhecimento de expressões faciais. As AUs (classificação de baixo nível) e as expressões faciais prototípicas (classificação de alto nível) definidas por Ekman et al. (1972).

Apresentam-se, de seguida, alguns dos métodos mais usados na etapa de classificação:

- Redes Neuronais: emulam o modelo neuronal biológico. Segundo Tivive and Bouzerdoum (2004) são conseguidos altos níveis de precisão. Porém, as redes neuronais podem ser difíceis de treinar no caso de se classificar não só as expressões básicas mas também expressões livres (Fasel and Luettin, 2003). Isto porque foram identificadas cerca de 7000 possíveis combinações de AUs (Ekman, 1982).

- SVM - *Support Vector Machines* (Cortes and Vapnik, 1995): baseiam-se na teoria da aprendizagem estatística (Vapnik, 1995). Consistem em métodos de aprendizagem supervisionados e são usados para classificações e regressões. São eficazes em espaços dimensionais de alto nível; usam um subconjunto de pontos de treino na função de decisão (os vetores de suporte), portanto são eficientes em termos de memória; são versáteis, na medida em que podem ser escolhidas diferentes funções *kernel* na função de decisão. Porém, apresentam duas desvantagens significativas: se o número de caraterísticas extraídas for muito maior que o número de amostras, a eficácia do método pode ficar comprometida; por outro lado não fornecem estimativas probabilísticas diretamente, usando um método um pouco pesado,

em termos de processamento, para tal[2].

- *AdaBoost* (Freund and Schapire, 1997): Classificador bastante preciso que se baseia na combinação de classificadores relativamente mais fracos. O *AdaBoost* é adaptativo no sentido em que cada novo classificador adicionado é alterado consoante eventuais caraterísticas mal classificadas por classificadores anteriores[3]. Trata-se de uma ferramenta poderosa e bastante eficaz mas que apresenta alguns problemas, nomeadamente em termos de sensibilidade a ruído e informação discrepante (Adeshina et al., 2009).

5.2 Implementação

Os objetivos, em termos de implementação, consistiram em fazer uma análise de robustez ao sistema *FaceCoder* e, consoante os resultados, realizar as devidas alterações de modo a melhorar a eficiência do mesmo. O *FaceCoder* utiliza o *software faceAPI* nas fases de deteção da face e extração de caraterísticas. As mesmas são então processadas e codificadas em AUs. A interpretação de AUs e consequente identificação do estado emocional é feita em camadas superiores da arquitetura do sistema.

5.2.1 Aplicação de análise de robustez

Para fazer a análise de robustez ao sistema criou-se uma aplicação gráfica, em C# (ver Figura 5.1), que recebe os dados relativos aos resultados da base de dados CK+[4] (*Extended Cohn-Kanade*). O *FaceCoder* foi adaptado para correr esta base de dados e guardar os resultados num ficheiro *.txt*. Por sua vez, a aplicação gráfica

[2] `http://scikit-learn.org/stable/modules/svm.html` [acedido em 2013-12-12].

[3] `http://www.nickgillian.com/wiki/pmwiki.php?n=GRT.AdaBoost` [acedido em 2013-12-11].

[4] Ver explicação da base de dados na secção 5.3.1.

(doravante designada por *Cohn-Kanade Analysis*) interpreta os dados provenientes do ficheiro *.txt* e faz uma análise genérica em termos de taxas de deteção e de falsos positivos de cada uma das AUs, e individual, permitindo selecionar os sujeitos de modo a ver a respetiva taxa de deteção e fazer uma comparação através das AUs detetadas e das AUs reais. A *Cohn-Kanade Analysis* permite, também, visualizar as fotos de cada um dos sujeitos, tanto na forma neutra como na(s) forma(s) expressiva(s) (a cada sujeito está associada uma ou mais expressões).

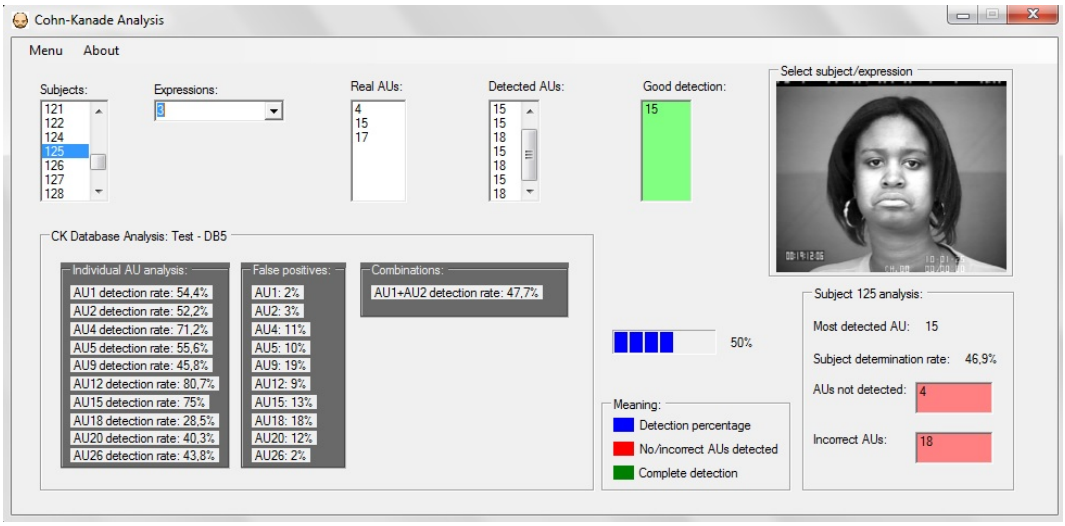


Figura 5.1: Aspeto visual da aplicação de análise de robustez *Cohn-Kanade Analysis*. A imagem do sujeito consta na base de dados CK+ (direitos da foto do sujeito reservados a ©Jeffrey Cohn).

5.2.2 Contributos para o *FaceCoder*

Os resultados preliminares da *Cohn-Kanade Analysis*, juntamente com os testes com pessoas, permitiram verificar algumas vulnerabilidades significativas no *FaceCoder*, principalmente na deteção de AUs associadas à região da boca. Isto porque o *tracking* dos lábios do *faceAPI* revelou-se ser pouco eficiente em movimentos mais

subtis. Para duas AUs em particular, esta fragilidade compromete a deteção das mesmas. São elas a AU15, relativa ao estado "tristeza" e a AU20, relativa ao estado "medo". Por sua vez, a AU12 e a AU26[5] apresentaram bons resultados, derivado do facto de serem AUs mais expressivas. Sendo a AU15 e a AU20 duas AUs fundamentais no processo de identificação dos respetivos estados emocionais, foram construídas implementações alternativas de modo a conseguir obter melhores resultados. Por outro lado, foi também implementada a AU9 relativa ao estado emocional "aversão".

AU	Descrição
1	Levantar parte interior da sobrancelha
2	Levantar parte exterior da sobrancelha
4	Baixar sobrancelha
5	Levantar pálpebra
9	Enrugar nariz
12	Levantar canto do lábio
15	Baixar canto do lábio
18	Franzir lábio
20	Esticar lábio
26	Cair queixo

Tabela 5.3: AUs usadas no *FaceCoder*.

5.2.2.1 AU9

O processo de deteção da AU9 baseia-se na determinação do número de rugas na região do nariz (ver Figura 5.2). Essa determinação é feita na forma neutra da face

[5] A AU26 apresentou melhores resultados através dos testes em tempo real do que na *Cohn-Kanade Analysis* - ver explicação na secção 5.3.2.

e na forma expressiva da face. Se a diferença entre as duas formas ultrapassar um determinado *threshold*, a AU9 é ativada.

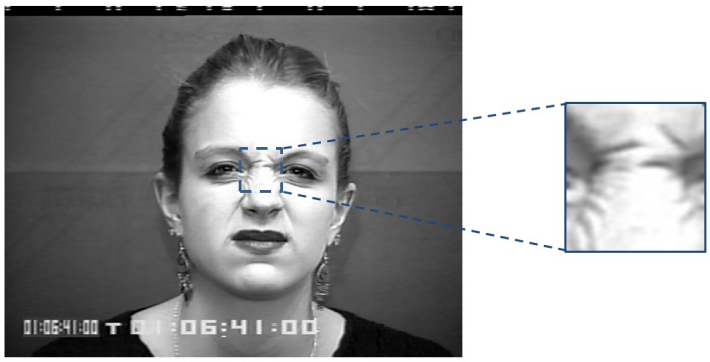
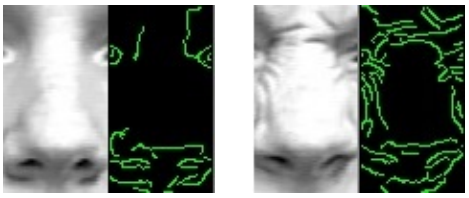


Figura 5.2: Rugas resultantes da ativação da AU9 (©Jeffrey Cohn).

Em primeiro lugar é necessário localizar o nariz do sujeito. Através de funções do *faceAPI* é possível extrair algumas caraterísticas da face (lábios, olhos, nariz e sobrancelhas). Os pontos mais relevantes dessas caraterísticas são codificados em *Face Landmarks*. Para localizar o nariz foram escolhidas *Landmarks* do lábio superior e da parte interior das sobrancelhas.

De seguida, utiliza-se o algoritmo *Canny*, com um valor de *threshold* intermédio[6], para detetar as rugas do nariz. Fazendo um rastreamento da imagem e contando todos os contornos, obtém-se a quantidade de rugas. Por último, calcula-se a diferença de contornos entre a face na forma expressiva e na forma neutra (ver Figura 5.3).

[6] *Threshold* superior atribuído é o dobro em relação ao *threshold* inferior. Relembra-se que o algoritmo *Canny* suporta um *threshold* duplo. Referências à escolha do *threshold* dizem respeito ao de valor inferior.



(a) Forma neutra. **(b)** Forma expressiva.

Figura 5.3: Imagem do nariz e respetivos contornos obtidos pelo algoritmo *Canny*.

Ultrapassando uma diferença de 10 contornos (valor ajustável num ficheiro de configurações), AU9 é ativada.

De referir que a deteção de contornos depende do *threshold* escolhido para o algoritmo *Canny*. Um *threshold* maior implica uma menor deteção de contornos, mesmo na forma expressiva. Um *threshold* menor implica uma maior deteção de contornos, mesmo na forma neutra. É preciso, portanto, encontrar um *threshold* intermédio de modo a não cair em nenhuma das situações extremas.

5.2.2.2 AU15

A deteção da AU15 baseia-se na medição do contorno correspondente à separação dos lábios (contorno mais escuro). Dividindo a boca em três partes semelhantes (ver Figura 5.4), se as partes esquerda e direita representarem um decaimento em relação à parte central, AU15 é ativada.

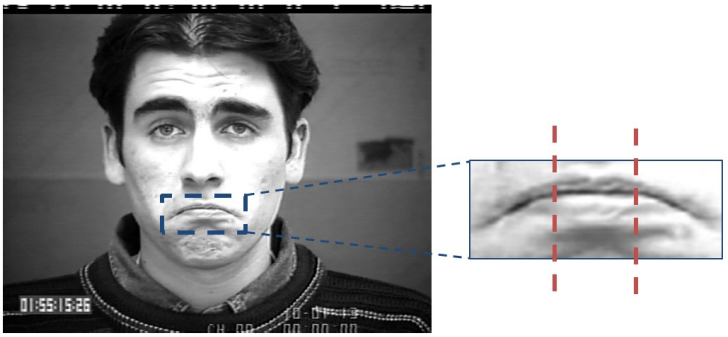
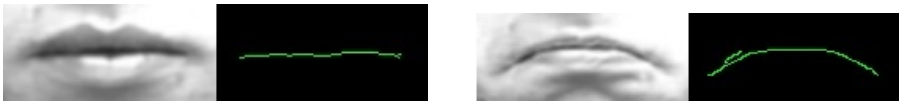


Figura 5.4: Divisão da boca em três partes semelhantes (©Jeffrey Cohn).

Mais uma vez recorre-se às *Landmarks* para localizar a boca. Escolheram-se *Landmarks* das extremidades da boca e adicionou-se uma pequena margem, compensando o facto do *tracking* de lábios do *faceAPI* apresentar algumas limitações. Utiliza-se, novamente, o algoritmo *Canny* para detetar o contorno pretendido. De modo a excluir contornos desnecessários, seleciona-se um valor alto de *threshold*. De todos os contornos devolvidos pelo processamento referido, escolhe-se apenas o maior, que deverá corresponder ao contorno desejado. Desenha-se um retângulo que contenha o contorno e retira-se a largura. A largura é então dividida por três de modo a obter as três partes da boca. Para cada parte calcula-se o valor médio no eixo dos yy. Este processamento é feito para a forma neutra e forma expressiva (ver Figura 5.5).



(a) Forma neutra. **(b)** Forma expressiva.

Figura 5.5: Imagem da boca e respetivo contorno selecionado.

Por último, obtidas as médias de cada uma das partes das duas formas, calcula-se a média entre partes correspondentes. Isto é, a média entre a primeira parte nas formas neutra e expressiva, a média entre a segunda parte nas formas neutra e expressiva e o mesmo para a terceira parte. Caso resultados das partes laterais seja superior[7] à parte central, AU15 é ativada.

5.2.2.3 AU20

A deteção da AU20 (ver Figura 5.6) baseia-se na medição da largura dos contornos dos lábios. Desta vez não é possível usar o contorno correspondente à sombra que separa ambos os lábios devido ao facto de haver a possibilidade da boca do sujeito estar ligeiramente aberta. Sendo assim, são usados os contornos dos lábios e caso a

[7] Mais uma vez se recorda que $y = 0$ corresponde à parte superior da janela de visualização.

diferença entre as larguras destes na forma neutra e na forma expressiva ultrapasse um determinado *threshold*, AU20 é ativada.

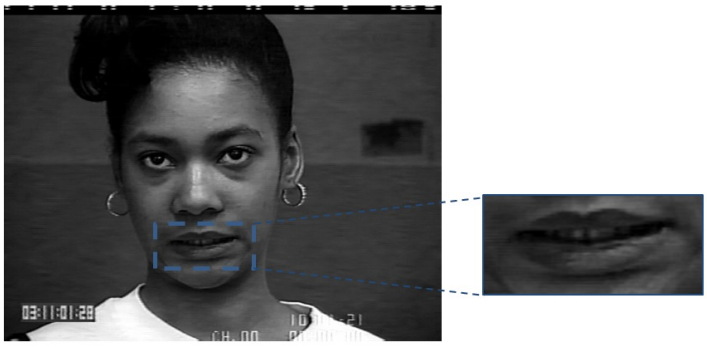
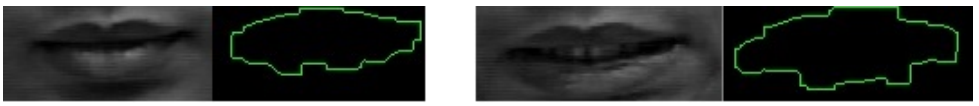


Figura 5.6: Ativação da AU20 (©Jeffrey Cohn).

Em primeiro obtêm-se os pontos correspondentes às *Landmarks* escolhidas (as mesmas da AU15, mas dando uma margem horizontal ainda maior). De seguida, utiliza-se o *Canny* para descobrir os contornos dos lábios. Aplica-se o método *Dilate* para unir eventuais contornos não fechados. Seleciona-se o maior contorno e desenha-se um retângulo que o contenha. Calcula-se, então, a largura desse retângulo. À semelhança das outras AUs, também nesta o processamento é feito para a forma neutra e para a forma expressiva (ver Figura 5.7). Diferença de larguras entre as formas irá determinar se a AU20 é ativada ou não.



(a) Forma neutra. **(b)** Forma expressiva.

Figura 5.7: Imagem da boca e respetivo contorno selecionado.

A ativação da AU20 está, porém, dependente de uma condição. Isto é, a AU12 (equivalente ao sorriso) implica, também, que os lábios estiquem, gerando uma situação de conflito na medida em que estas AUs não são compatíveis. Sendo

assim, caso a AU12 seja ativada, a AU20 é desabilitada.

5.3 Resultados

A análise de resultados baseou-se na interpretação dos dados resultantes da *Cohn-Kanade Analysis* assim como no teste efetivo com pessoas a interagir com o ARoS. Apresentam-se, no decorrer desta secção, algumas das considerações mais relevantes acerca dos resultados obtidos.

5.3.1 *Cohn-Kanade Analysis*

A base de dados CK+ (Kanade and Cohn, 2000; Lucey et al., 2010) é composta por 593 sequências de imagens de 123 sujeitos. As sequências de imagens variam em duração, i.e., de 10 a 60 *frames*. Cada sequência começa na forma neutra e acaba na forma expressiva (ver Figura 5.8). A cada imagem de pico (face na forma expressiva) está associado um ficheiro *.txt* que contempla as AUs reais dessa imagem, identificadas por especialistas no FACS.

Figura 5.8: Exemplo de uma das sequências de imagens (©Jeffrey Cohn).

O *FaceCoder* foi adaptado para analisar todas as 593 sequências: para cada sequência faz a análise da forma neutra através da primeira *frame* e da forma expressiva através das últimas cinco *frames*, num total de 3558 *frames* (593 das formas neutras mais 2965 das formas expressivas).

Os resultados obtidos na fase preliminar revelaram algumas fragilidades na deteção de movimentos associados à região da boca, nomeadamente nas AUs 15 e 20, cujos movimentos são mais subtis. Foram introduzidas implementações alternativas para estas duas AUs, assim como se adicionou a AU9. Na Tabela 5.4 apresentam-se os resultados obtidos após a introdução de alterações.

AU	Taxa de deteção	Taxa de falsos positivos
1	51,2%	3%
2	57,9%	4%
4	73,8%	9%
5	57,6%	9%
9	**53,4%**	**16%**
12	75%	10%
15	**72,3%**	**11%**
18	75%	17%
20	**40,3%**	**12%**
26	45,4%	2%

Tabela 5.4: Resultados de um dos testes da *Cohn-Kanade Analysis* após a introdução de alterações no *FaceCoder*.

Em relação aos resultados obtidos, apresentam-se, seguidamente, algumas das considerações mais relevantes acerca dos mesmos, dando especial relevância às AUs 9, 15 e 20:

- A AU9 não depende diretamente[8] de nenhuma das quatro caraterísticas fundamentais da face (lábios, olhos, nariz e sobrancelhas) extraídas pelo *faceAPI*. Apesar da posição correspondente à ativação da AU9 se situar na parte superior do nariz, esta AU é ativada por uma diferença de textura e

[8]Não depende em termos de deteção mas depende em termos de localização da região.

não pela mudança relativa de posição de uma caraterística facial. Como tal, a AU9 é uma das AUs de deteção mais complicada: em primeiro, porque nem todos os sujeitos conseguem fazer rugas suficientemente salientes de modo a serem identificadas pelo sistema de visão; em segundo, porque é preciso encontrar um *threshold* equilibrado de modo a conseguir bons níveis de eficácia, mas, ao mesmo tempo, uma taxa de falsos positivos baixa. Após vários testes, com diferentes parâmetros, atingiram-se os resultados da Tabela 5.4, satisfatórios tendo em conta as condicionantes associadas a esta AU.

- A anterior implementação da AU15 recorria ao *tracking* dos lábios proporcionado pelo *faceAPI*. Tal como referido, este *tracking* sofre de algumas limitações no caso de movimentos mais subtis. Como tal, usando parâmetros de deteção mais sensíveis, seria possível atingir taxas de deteção semelhantes às da Tabela 5.4[9], mas com taxas de falsos positivos mais elevadas (na ordem dos 15%-20%). Com a implementação alternativa foi possível atingir resultados bastante bons, tanto em taxas de deteção, como em taxas de falsos positivos. De salientar que a taxa de deteção poderia ser ainda mais elevada: isto porque foi criada uma exceção que desabilita a ativação da AU15 no caso da AU12 ter sido ativada, sendo elas incompatíveis. Ou seja, no caso de falsas deteções da AU12 que possam corresponder à AU15, a última é descartada. Sem esta exceção, taxas de deteção da nova implementação da AU15 atingiriam os 85%.

- A AU20 é a que apresenta piores resultados. Mesmo assim, a nova implementação veio aumentar, significativamente, a taxa de deteção, mantendo a taxa de falsos positivos. A anterior implementação apresentava resultados abaixo dos 20%. Os resultados baixos associados a esta AU explicam-se pelo facto de nem sempre, à contração dos lábios, corresponder um alargamento

[9] Segundo resultados da *Cohn-Kanade Analysis*. Nos testes em tempo real, deteção segundo a anterior implementação era bastante limitada.

suficientemente significativo que permita ao sistema de visão ativar a AU20. Por outro lado, quando as caraterísticas dos lábios não são extraídas corretamente, se a margem dada não for suficientemente abrangente para cobrir a extensão dos lábios, o cálculo da largura fica comprometido.

- Em relação às restantes AUs, apresentam-se, de seguida, algumas considerações acerca dos respetivos resultados:
  - As AUs 1, 2 e 4 dizem respeito às sobrancelhas. Para o sistema é mais "fácil" detetar o "baixar das sobrancelhas" (AU4) do que o "subir" (AU1 e AU2, que em grande parte das vezes são ativadas em simultâneo).
  - A AU5 é detetada através do cálculo da diferença da quantidade de cor branca nos olhos na forma expressiva relativamente à verificada na forma neutra. Quanto maior for essa diferença, maior é a probabilidade da pálpebra ter sido levantada. Esta implementação é a que melhor se adequa e os resultados assim o demonstram (isto tendo em conta que é uma das AUs de mais complicada deteção, juntamente com a 9 e a 20).
  - A AU12 não necessita de uma implementação alternativa como o que acontece com a AU15 devido ao facto da expressão de sorriso ser mais saliente do que a expressão de tristeza que ativa a AU15. Como tal, o *tracking* do *faceAPI* é suficiente para ativar a AU12.
  - A AU18 é a que apresenta maior taxa de falsos positivos derivado do facto de ser uma das AUs mais sensíveis. As taxas apresentadas na Tabela 5.4 acabam por representar um bom resultado em termos de relação de equilíbrio entre a taxa de deteção e a taxa de falsos positivos.
  - A AU26 apresenta taxas de deteção na ordem dos 45% pois o *tracking* dos lábios muitas vezes se "perde" com a abertura da boca, fazendo acompanhamento apenas do lábio inferior (resultando num falso positivo da AU12) ou do lábio superior (resultando num falso positivo da AU15).

Porém, neste caso, a introdução de uma implementação alternativa à semelhança da AU15 ou AU20 não resultaria, pois com a perda da localização da boca após abertura, o ROI usado para o processamento de caraterísticas não seria suficientemente grande.

5.3.2 Testes em tempo real

Os testes com pessoas em tempo real revelaram uma boa robustez em curtos tempos de processamento (35-38 fps)[10]. Tirando o caso da AU26, os testes vieram confirmar os resultados da Tabela 5.4. A AU26 apresenta melhores resultados nos testes em tempo real devido ao facto de se poderem processar todas as *frames*. Isto é, nos testes com a base de dados apenas são consideradas as últimas cinco *frames* da forma expressiva, não sendo, por vezes, o suficiente para fazer o acompanhamento do movimento da abertura da boca.

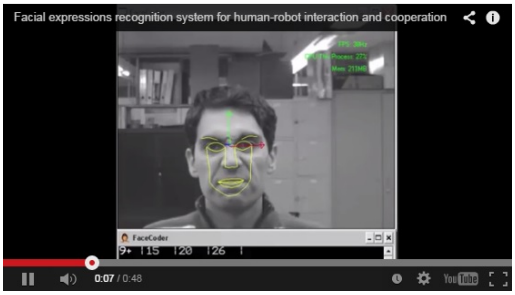
O vídeo ilustrado na Figura 5.9 demonstra as AUs acrescentadas e melhoradas [11] no *FaceCoder*. No vídeo, em primeiro as AUs são ativadas uma de cada vez, por duas vezes seguidas (ver a partir dos 0:06). Por último, ativa-se apenas por uma vez, cada uma delas (ver a partir dos 0:32).

A AU9 apresenta uma boa robustez de deteção, assim como a AU15. Contrariamente, a AU20 apresenta algumas limitações. Tal deve-se, essencialmente, à falha na extração dos lábios (ver a partir dos 0:20). No caso da implementação alternativa da AU15 é possível contornar essa limitação. Repare-se, aos 0:15, em que os traços correspondentes à extração dos lábios não correspondem à posição efetiva dos mesmos. Porém, como o rastreio é feito no eixo vertical, a extração insuficiente dos lábios não influencia o processo de ativação da AU15 (são dadas margens em relação às caraterísticas extraídas). Já em relação à AU20, como o

[10] Testes efetuados num *Intel® Core™ 2 Quad Q6600, 2.40 GHz, 2.0 GB RAM.*

[11] Também demonstra a AU26 para comprovar a fácil ativação da mesma nos testes em tempo real.

rastreio é feito no eixo horizontal, se a extração for curta (como acontece aos 0:20), sistema deve falhar a ativação pois a largura fica condicionada ao espaço limitado pela extração mais a respetiva margem. Por outras palavras, a largura calculada na forma expressiva pode mesmo ser menor do que a largura na forma neutra. Apesar das limitações subjacentes à AU20, a implementação alternativa veio melhorar substancialmente as taxas de deteção da mesma (para o dobro sensivelmente - ver Tabela 5.4 e respetiva explicação). Outra das situações de destaque consiste na combinação das AUs 9 e 15 que é corretamente identificada pelo sistema (0:10).



(a) AU9.

(b) AU15.

(c) AU20.

(d) AU26.

Figura 5.9: Ativação das três AUs implementadas mais demonstração da AU26. Visualizar vídeo em `http://marl.dei.uminho.pt/public/videos/facial_exp.html`

5.3.3 Discussão dos resultados

A deteção de AUs depende, essencialmente, de uma boa extração de caraterísticas faciais. Isto é, com os resultados dos vários testes efetuados, tanto com recurso à base de dados como com pessoas em tempo real, constatou-se que, na maioria das vezes, a falha de deteção de AUs ou mesmo o aparecimento de falsos positivos se devia à incorreta extração de caraterísticas da face (modelo da face criado erradamente, por exemplo). Quando as mesmas são extraídas corretamente, verificam-se elevadas taxas de ativação de AUs certas e, em sentido inverso, poucos falsos positivos.

Em relação às alterações efetuadas, estas vieram melhorar a eficácia do sistema pois incidem sobre AUs fundamentais no processo de identificação dos respetivos estados emocionais: a AU9 (implementação adicionada), relativa ao estado emocional "aversão", a AU15 (implementação alternativa), relativa ao estado emocional "tristeza" e a AU20 (implementação alternativa), relativa ao estado emocional "medo".

Esta página foi intencionalmente deixada em branco!

Parte III

Conclusão

Capítulo 6

Resultados da Integração

Neste capítulo apresentam-se alguns resultados da integração dos sistemas que envolvem as câmaras usadas no processamento estereoscópico. Ou seja, os sistemas de reconhecimento de objetos e reconhecimento de gestos (o reconhecimento de expressões faciais é feito paralelamente, numa outra câmara). A Figura 6.1 ilustra um vídeo que contempla o Cenário B (ver secção 3.3.1)[1]. Através do gesto "apontar", sistema identifica e devolve a localização do objeto apontado. Ou seja, neste vídeo de integração demonstram-se as três vertentes mais dispendiosas em termos de processamento: o gesto "apontar" e respetivo pós-processamento associado, o reconhecimento do objeto apontado e a localização no espaço do mesmo.

Comparativamente com os resultados relativos ao mesmo cenário, expostos no vídeo ilustrado na Figura 3.35, o *frame rate* decresce de 7-9 fps para cerca de 6 fps. Esta diferença curta explica-se pelo facto de o algoritmo de reconhecimento e o processamento estereoscópico serem os principais responsáveis pelo tempo de resposta do programa. Ou seja, a inclusão do sistema de reconhecimento gestual, mesmo com o pós-processamento relativo ao gesto "apontar", não introduz mudanças significativas em termos de tempo de processamento.

[1] Testes efetuados num *Intel® Core™ 2 Quad Q6600, 2.40 GHz, 2.0 GB RAM.*

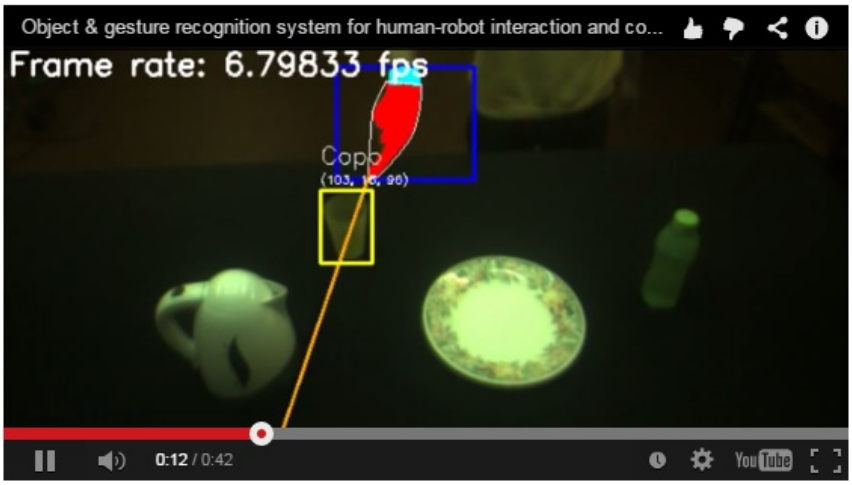


Figura 6.1: Demonstração da integração do reconhecimento do gesto "apontar" e respetivo pós-processamento com o sistema de reconhecimento de objetos. Visualizar vídeo em `http://marl.dei.uminho.pt/public/videos/integration_PointingObjects.html`

Os resultados expostos permitem concluir que a integração dos sistemas de reconhecimento de objetos e de gestos (relembra-se que o reconhecimento de expressões faciais é feito paralelamente) é suficientemente robusta e passível de funcionamento em tarefas de interação e colaboração em tempo real, dada a rapidez de processamento associada.

As três vertentes consistem em componentes de baixo nível para integração num sistema robótico complexo (Bicho et al., 2011). Em relação aos objetos, o Cenário A é o usado atualmente, sendo que os Cenários B e C vêm alargar a gama de possibilidades em termos de tarefas de interação e colaboração humano-robô.

Capítulo 7

Conclusões e Trabalho Futuro

Com este projeto de dissertação implementou-se, no robô antropomórfico ARoS, um sistema de visão dedicado ao reconhecimento de objetos, gestos e expressões faciais. Em relação aos objetos, foi implementado um sistema de reconhecimento híbrido baseado em duas abordagens distintas: caraterísticas globais e caraterísticas locais. Este sistema permite uma fácil integração de novos objetos (inserindo uma foto, por exemplo) sem que seja necessário efetuar mudanças recorrentes no código de modo a processar caraterísticas específicas do objeto (exceção feita aos objetos iguais com cores diferentes), o que representa um avanço em relação ao sistema previamente criado, que se baseava na diferenciação de um conjunto limitado de objetos pela cor.

O sistema de reconhecimento de gestos vem responder à necessidade premente de dotar o ARoS com a capacidade de identificar gestos de modo a poder inferir as intenções do parceiro humano e assim o robô poder adotar um comportamento adequado tendo em conta as expetativas do humano. Também neste caso, a inclusão de novos gestos no sistema desenvolvido é bastante simples, bastando, para tal, definir um conjunto de gamas de momentos para o gesto em questão. O gesto deve, contudo, apresentar diferenças significativas relativamente aos já adicionados, de maneira a não se verificarem situações conflituosas entre diferentes

gestos.

O sistema de reconhecimento de expressões faciais *FaceCoder* (previamente criado) sofreu algumas alterações no intuito de aumentar a robustez do mesmo. A aplicação de análise de robustez construída no âmbito desta dissertação permitiu verificar algumas vulnerabilidades do sistema que estiveram na base das modificações efetuadas. De aqui em diante, esta aplicação será uma ferramenta útil na avaliação de desempenho do sistema, mediante as atualizações que forem sendo realizadas.

As três vertentes referidas foram exploradas e implementadas tendo como pressuposto a criação de um ambiente genérico, de simples adaptação a diferentes contextos. Contudo, algumas limitações do próprio *hardware* podem condicionar as tarefas de reconhecimento. Por exemplo, em relação aos objetos, a resolução limitada do sistema de visão e o facto das câmaras estarem a uma distância considerável do plano de trabalho tornam difícil o reconhecimento de objetos de pequenos tamanhos (talheres, por exemplo), quer por forma, quer por caraterísticas locais.

Como trabalho futuro, pretende-se, por um lado, *i)* desenvolver algumas das capacidades introduzidas com este sistema de visão e, por outro, *ii)* acrescentar uma nova vertente, respeitante aos movimentos corporais.

No que concerne ao desenvolvimento das capacidades introduzidas, o aumento de robustez dos algoritmos e diminuição do tempo de resposta são alguns dos desafios mais relevantes com vista a melhorar a interação e colaboração entre humano e robô em tempo real. Para tal, poderá ser compensatório a utilização de *tracking* nos objetos. Através de *tracking*, o sistema apenas necessita de correr o algoritmo de reconhecimento na primeira *frame*, poupando capacidade de processamento (no caso de um objeto identificado segundo caraterísticas locais). Porém, o *tracking* só deverá ser implementado caso o algoritmo de reconhecimento seja bastante robusto, de modo a não fazer *tracking* de objetos mal identificados. Em relação aos gestos, pretendendo-se implementar um número significativo de

gestos, a probabilidade de existirem ambiguidades na diferenciação de diferentes gestos aumenta e poderá ser necessário recorrer a um classificador (ao contrário dos objetos, no caso dos gestos a forma da mão não é uma caraterística fixa). Também no reconhecimento de expressões faciais poderão ser introduzidas alterações para aumentar os níveis de robustez e até adicionar novas AUs, embora, neste caso, as modificações estejam sempre condicionadas pelo desempenho do extrator de caraterísticas da face (neste caso o *software faceAPI*).

Em relação à implementação da vertente dos movimentos corporais, os mesmos poderão fornecer indicações complementares ao reconhecimento de expressões faciais. Uma aferição mais robusta e abrangente do estado emocional do humano, i.e., que não se restrinja apenas às seis emoções básicas, passa pela interpretação de movimentos corporais. A combinação da informação correspondente às duas vertentes, permite uma gama de possibilidades mais ampla nas camadas de alto nível do sistema.

Esta página foi intencionalmente deixada em branco!

Referências Bibliográficas

Adeshina, A. M., Lau, S.-h., and Loo, C.-k. (2009). Real-time facial expression recognitions: A review. In *2009 Innovative Technologies in Intelligent Systems and Industrial Applications*, number July, pages 375–378. IEEE.

Alahi, A., Ortiz, R., and Vandergheynst, P. (2012). FREAK: Fast Retina Keypoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517. IEEE.

Amadeo, R. (2014). Hands-on with Baxter, the factory robot of the future (online). http://arstechnica.com/gadgets/2014/06/hands-on-with-baxter-the-factory-robot-of-the-future/. Accessed: 2014-07-09.

Amanatiadis, A., Kaburlasos, V., Gasteratos, A., and Papadakis, S. (2011). Evaluation of shape descriptors for shape-based image retrieval. *IET Image Processing*, 5(5):493–499.

Argyros, A. and Lourakis, M. (2006). Binocular Hand Tracking and Reconstruction Based on 2D Shape Matching. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 00, pages 207–210. IEEE.

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359.

Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object

recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522.

Bicho, E., Erlhagen, W., Louro, L., Costa e Silva, E., Silva, R., and Hipólito, N. (2011). A dynamic field approach to goal inference, error detection and anticipatory action selection in human-robot collaboration. In Dautenhahn, K. and Saunders, J., editors, *New Frontiers in Human-Robot Interaction (Advances in Interaction Studies)*, pages 135–164. John Benjamins Publishing Company, 6 edition.

Bicho, E., Louro, L., and Erlhagen, W. (2010). Integrating verbal and nonverbal communication in a dynamic neural field architecture for human-robot interaction. *Frontiers in neurorobotics*, 4(May):1–13.

Bishnu, A., Bhattacharya, B. B., Kundu, M. K., Murthy, C. a., and Acharya, T. (2005). Euler vector for search and retrieval of gray-tone images. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 35(4):801–12.

Bohren, J., Rusu, R. B., Gil Jones, E., Marder-Eppstein, E., Pantofaru, C., Wise, M., Mosenlechner, L., Meeussen, W., and Holzer, S. (2011). Towards autonomous robotic butlers: Lessons learned with the PR2. In *2011 IEEE International Conference on Robotics and Automation*, pages 5568–5575. IEEE.

Bradski, G. and Kaehler, A. (2008). *Learning OpenCV*. O'Reilly Media, Inc., first edition.

Calonder, M., Lepetit, V., Ozuysal, M., Trzcinski, T., Strecha, C., and Fua, P. (2011). BRIEF: Computing a Local Binary Descriptor very Fast. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1281 – 1298.

Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). BRIEF: Binary Robust Independent Elementary Features. In *Proceedings of the 11th European*

Conference on Computer Vision: Part IV, volume 6314, pages 778–792, Berlin, Heidelberg. Springer-Verlag.

Canny, J. (1986). A computational approach to edge detection. *IEEE transactions on pattern analysis and machine intelligence*, 8(6):679–98.

Chibelushi, C. C. and Bourel, F. (2002). Facial Expression Recognition : A Brief Tutorial Overview.

Chitra, S. and Balakrishnan, G. (2012). A Survey of Face Recognition on Feature Extraction Process of Dimensionality Reduction Techniques. *Journal of Theoretical and Applied Information Technology*, 36(1):92–100.

Cootes, T., Edwards, G., and Taylor, C. J. (1999). Comparing Active Shape Models with Active Appearance Models. In *Procedings of the British Machine Vision Conference 1999*, pages 18.1–18.10. British Machine Vision Association.

Cootes, T., Taylor, C., Cooper, D., and Graham, J. (1995). Active Shape Models-Their Training and Application. *Computer Vision and Image Understanding*, 61(1):38–59.

Coradeschi, S., Cesta, A., Cortellessa, G., Coraci, L., Gonzalez, J., Karlsson, L., Furfari, F., Loutfi, A., Orlandini, A., Palumbo, F., Pecora, F., von Rump, S., Stimec, A., Ullberg, J., and Otslund, B. (2013). GiraffPlus: Combining social interaction and long term monitoring for promoting independent living. In *2013 6th International Conference on Human System Interactions (HSI)*, pages 578–585. IEEE.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Cutler, R. and Turk, M. (1998). View-based interpretation of real-time optical flow

for gesture recognition. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 416–421. IEEE Comput. Soc.

Denavit, J. and Hartenberg, R. S. (1955). A kinematic notation for lower-pair mechanisms based on matrices. *Trans. of the ASME. Journal of Applied Mechanics*, 22:215–221.

Eckart, C. and Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218.

Edwards, G., Taylor, C., and Cootes, T. (1998). Interpreting face images using active appearance models. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 300–305. IEEE Comput. Soc.

Ekman, P. (1982). Methods for measuring facial action. In Scherer, K. and Ekman, P., editors, *Handbook of Methods in Nonverbal Behaviour Research*, chapter 2, pages 45–135. Cambridge University Press, New York.

Ekman, P. and Friesen, W. (1978). *Facial Action Coding System: A Technique for the Measurement of Facial Movement.* Consulting Psychologists Press, Palo Alto.

Ekman, P., Friesen, W. F., and Hager, J. (2002). Translating AU Scores Into Emotion Terms. In *FACS Manual 2002, Investigator's Guide*, pages 173–174.

Ekman, P., Friesen, W. V., and Ellsworth, P. (1972). *Emotion in the Human Face*. Oxford University Press.

Espinosa-Duro, V., Faundez-Zanuy, M., and Ortega, J. A. (2004). Face detection from a video camera image sequence. In *38th Annual 2004 International Carnahan Conference on Security Technology, 2004.*, pages 318–320.

Evans, C. (2009). Notes on the OpenSURF Library. Technical Report 1, University of Bristol, Bristol, UK.

Fasel, B. and Luettin, J. (2003). Automatic facial expression analysis: a survey. *Pattern Recognition*, 36(1):259–275.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Fisher, R. A. (1936). The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 7(7):179–188.

Fitzgerald, C. (2013). Developing baxter. In *2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA)*, pages 1–6. IEEE.

Folkers, A. and Samet, H. (2002). Content-based Image Retrieval Using Fourier Descriptors on a Logo Database. In *2002 16th International Conference on Pattern Recognition*, volume III, pages 521–524, Quebec. IEEE.

Freund, Y. and Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

Golub, G. and Kahan, W. (1965). Calculating the Singular Values and Pseudo-Inverse of a Matrix. *Journal of the Society for Industrial and Applied Mathematics Series B Numerical Analysis*, 2(2):205–224.

Golub, G. H. and Reinsch, C. (1970). Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14(5):403–420.

Graham, R. (1972). An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters*, 1(4):132–133.

Guizzo, E. (2014). Meet Pepper, Aldebaran's New Personal Robot With an "Emotion Engine" (online). http://spectrum.ieee.org/automaton/robotics/home-robots/pepper-aldebaran-softbank-personal-robot. Accessed: 2014-07-20.

Haar, A. (1910). Zur Theorie der orthogonalen Funktionensysteme. *Mathematische Annalen*, 69(3):331–371.

Hamming, R. W. (1950). Error Detecting and Error Correcting Codes. *The Bell System Technical Journal*, 29(2):147–160.

Hinton, G. and Roweis, S. (2002). Stochastic Neighbor Embedding. In *Advances in Neural Information Processing Systems 15*, pages 833–840. MIT Press.

Hormann, K. and Agathos, A. (2001). The point in polygon problem for arbitrary polygons. *Computational Geometry*, 20(3):131–144.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441.

Hsu, G.-s., Loc, T. T., and Chung, S.-l. (2012). A Comparison Study on Appearance-Based Object Recognition. In *2012 21st International Conference on Pattern Recognition (ICPR)*, number Icpr, pages 3500–3503, Tsukuba. IEEE.

Hu, M.-K. (1962). Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2):179–187.

Kanade, T. and Cohn, J. (2000). Comprehensive database for facial expression analysis. In *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, pages 46–53. IEEE Comput. Soc.

Kim, J. (1999). An HMM-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):961–973.

Konolige, K. and Beymer, D. (2007). SRI Small Vision System - User's Manual (software version 4.4d). Technical report, SRI International.

Kunii, T. and Lee, J. (1995). Model-based analysis of hand posture. *IEEE Computer Graphics and Applications*, 15(5):77–86.

Leutenegger, S., Chli, M., and Siegwart, R. Y. (2011). BRISK: Binary Robust invariant scalable keypoints. In *2011 International Conference on Computer Vision*, pages 2548–2555. IEEE.

Li, N., Bu, J., and Chen, C. (2002). Real-time video object segmentation using HSV space. In *2002 International Conference on Image Processing*, volume 2, pages 85–88. IEEE.

Liu, Z., Yang, J., and Peng, N. S. (2005). An efficient face segmentation algorithm based on binary partition tree. *Signal Processing: Image Communication*, 20(4):295–314.

Lowe, D. (1999). Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157. IEEE.

Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., and Matthews, I. (2010). The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, number July, pages 94–101. IEEE.

Martin, J., Devin, V., and Crowley, J. (1998). Active hand tracking. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 573–578. IEEE Comput. Soc.

McColl, D., Louie, W.-y. G., and Nejat, G. (2013). Brian 2.1: A socially assistive robot for the elderly and cognitively impaired. *IEEE Robotics & Automation Magazine*, 20(1):74–83.

McColl, D. and Nejat, G. (2013). Meal-Time with a Socially Assistive Robot and Older Adults at a Long-term Care Facility. *Journal of Human-Robot Interaction*, 2(1):152–171.

Moeslund, T. B. (2009). Canny Edge Detection. Technical report.

Murthy, G. R. S. and Jadon, R. S. (2009). A review of vision based hand gestures recognition. *International Journal of Information Technology and Knowledge Management*, 2(2):405–410.

Murthy, K. N. B. and Natarajan, S. (2011). Dimensionality Reduction Techniques for Face Recognition. In Corcoran, P., editor, *Reviews, Refinements and New Ideas in Face Recognition*, chapter 7, pages 141–166. InTech.

New, J. R., Hasanbelliu, E., and Aguilar, M. (2003). Facilitating User Interaction with Complex Systems via Hand Gesture Recognition. In *Proceedings of the 2003 Southeastern ACM Conference*.

Newman-Norlund, R. D., Noordzij, M. L., Meulenbroek, R. G. J., and Bekkering, H. (2007). Exploring the brain basis of joint action: co-ordination of actions, goals and intentions. *Social Neuroscience*, 2(1):48–65.

Otiniano-Rodríguez, K. C., Cámara-Chávez, G., and Menotti, D. (2012). Hu and Zernike Moments for Sign Language Recognition. In *2012 International Conference on Image Processing, Computer Vision, and Pattern Recognition*, pages 1–5.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572.

Piccardi, M. (2004). Background subtraction techniques: a review. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, pages 3099–3104. IEEE.

Rautaray, S. S. and Agrawal, A. (2012). Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review*.

Reiser, U., Connette, C., Fischer, J., Kubacki, J., Bubeck, A., Weisshardt, F., Jacobs, T., Parlitz, C., Hagele, M., and Verl, A. (2009). Care-O-bot® 3 - creating a product vision for service robot applications by integrating design and technology. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1992–1998. IEEE.

Ricard, J., Coeurjolly, D., and Baskurt, A. (2005). Generalizations of Angular Radial Transform for 2D and 3D Shape Retrieval. *Pattern Recognition Letters*, 26(14):2174–2186.

Rosten, E. and Drummond, T. (2006). Machine Learning for High-speed Corner Detection. In *Proceedings of the 9th European Conference on Computer Vision - Volume Part I*, ECCV'06, pages 430–443, Berlin, Heidelberg. Springer-Verlag.

Roth, P. M. and Winter, M. (2008). Survey of Appearance-Based Methods for Object Recognition. Technical report, Institute for Computer Graphics and Vision, Graz University of Technology.

Ruberto, C. D. and Morgera, A. (2008). Moment-Based Techniques for Image Retrieval. In *2008 19th International Conference on Database and Expert Systems Applications*, pages 155–159, Turin. IEEE.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In *2011 International Conference on Computer Vision*, pages 2564–2571. IEEE.

Schölkopf, B., Smola, A., and Müller, K.-R. (1998). Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation*, 10(5):1299–1319.

Sebanz, N., Bekkering, H., and Knoblich, G. (2006). Joint action: bodies and minds moving together. *Trends in cognitive sciences*, 10(2):70–6.

Sharma, S. and Dhole, A. (2013). Content Based Image Retrieval Based on Shape Feature using Accurate Legendre Moments and Support Vector Machines. *International Journal of Computer Science & Engineering Technology*, 3(5):194–199.

Sigal, L., Sclaroff, S., and Athitsos, V. (2004). Skin color-based video segmentation under time-varying illumination. *IEEE transactions on pattern analysis and machine intelligence*, 26(7):862–77.

Silva, R. (2008). Design e Construção de um Robot Antropomórfico. Master's thesis, Universidade do Minho.

Siscoutto, R. A., Szenberg, F., Tori, R., Raposo, A. B., Celes, W., and Gattass, M. (2004). Estereoscopia. In Kirner, C. and Tori, R., editors, *Realidade Virtual: Conceitos e Tendências - Livro do Pré-Simpósio SVR 2004*, chapter 11, pages 179–201. Mania de Livro, São Paulo.

Stenger, B., Mendonça, P. R. S., and R. Cipo (2001). Model-Based 3D Tracking of an Articulated Hand. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 310–315.

Strang, G. (1999). The Discrete Cosine Transform. *SIAM Review*, 41(1):135–147.

Sutherland, E. E., Sproull, R. F., and Schumacker, R. A. (1974). A Characterization of Ten Hidden-Surface Algorithms. *ACM Computing Surveys*, 6(1):1–55.

Suzuki, Y. and Shibata, T. (2004). Multiple-clue face detection algorithm using edge-based feature vectors. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages V–737–40. IEEE.

Tao, H. (2006). Appearance-Based Object Recognition - Subspace Methods. University Lecture - CMPE 264 Image Analysis and Computer Vision. Department of Computer Engineering - University of California at Santa Cruz.

Tenenbaum, J. B., Silva, V., and Langford, J. C. (2000). A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319–2323.

Tivive, F. and Bouzerdoum, A. (2004). A face detection system using shunting inhibitory convolutional neural networks. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, volume 4, pages 2571–2575. IEEE.

Tuytelaars, T. and Mikolajczyk, K. (2007). Local Invariant Feature Detectors: A Survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3):177–280.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*, volume 38. Springer-Verlag New York, Inc., New York, NY, USA.

Viola, P. and Jones, M. J. (2004). Robust Real-Time Face Detection. *International Journal of Computer Vision*, 57(2):137–154.

Wang, H. and Wang, K. (2002). Facial feature extraction and image-based face drawing. In *6th International Conference on Signal Processing, 2002.*, volume 1, pages 699–702. IEEE.

Westphal, G., von Der Malsburg, C., and Würtz, R. P. (2008). Feature-driven emergence of model graphs for object recognition and categorization. In Kandel,

A., Bunke, H., and Last, M., editors, *Applied Pattern Recognition*, pages 155–199. Springer.

Wilson, A. and Bobick, A. (1999). Parametric hidden Markov models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9):884–900.

Witkin, A. P. (1983). Scale-space Filtering. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'83, pages 1019–1022, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Yin, X. and Xie, M. (2003). Estimation of the fundamental matrix from uncalibrated stereo hand images for 3D hand gesture recognition. *Pattern Recognition*, 36(3):567–584.