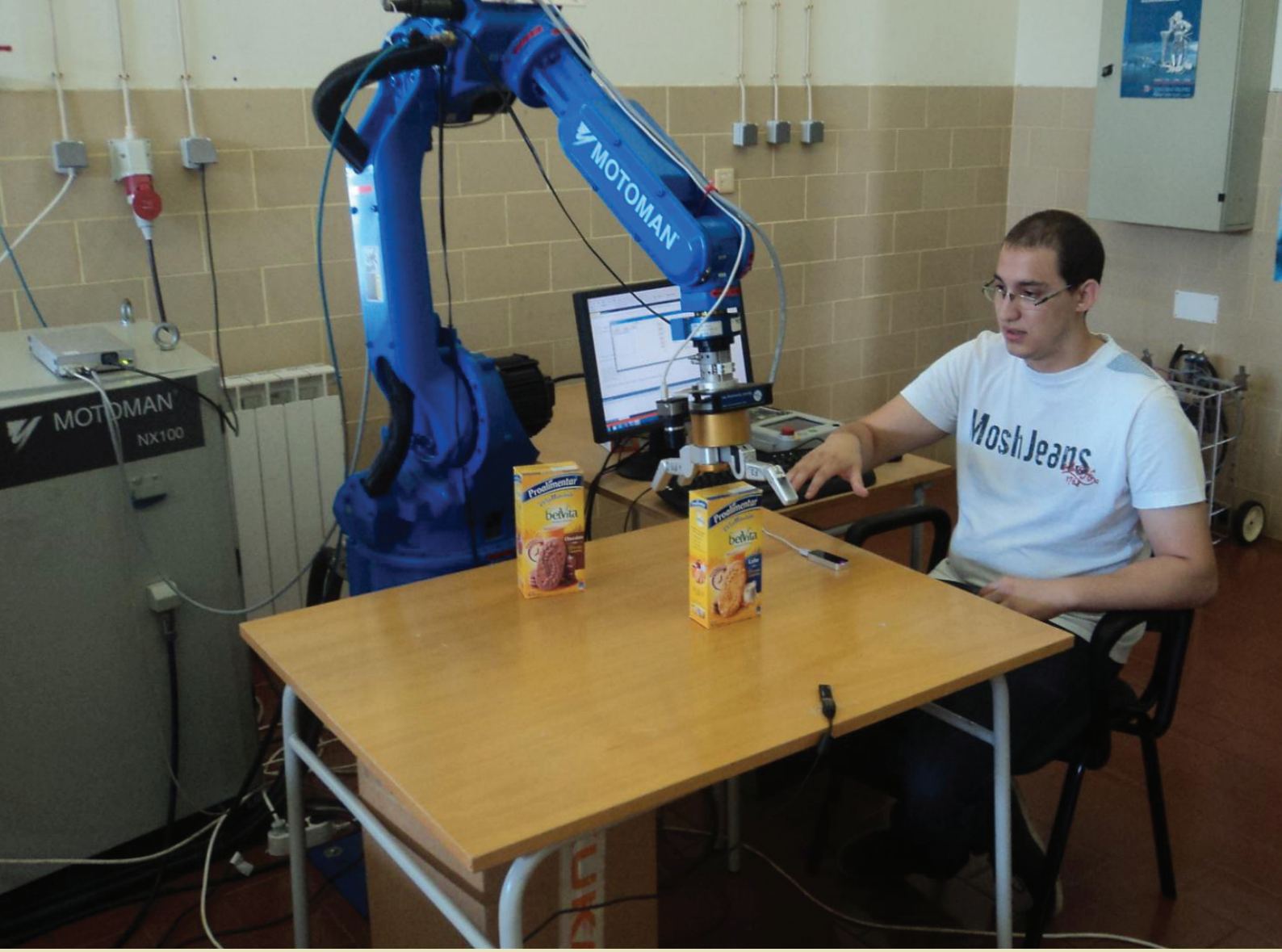


João Pedro Amaro Vitorino

Segmentação e Reconhecimento de Gestos da Mão Humana

Dissertação de Mestrado em Engenharia Mecânica

Setembro / 2014

· U C ·

Universidade de Coimbra

• U C •
FCTUC FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

DEPARTAMENTO DE ENGENHARIA MECÂNICA

Segmentação e Reconhecimento de Gestos da Mão Humana

Dissertação apresentada para a obtenção do grau de Mestre em Engenharia Mecânica

Autor
João Pedro Amaro Vitorino

Orientador
Professor Doutor Pedro Mariano Simões Neto

Júri

Presidente	**Professor Doutor José Luís Ferreira Afonso** **Professor Auxiliar da Universidade de Coimbra**
Vogais	**Mestre Nuno Alberto Marques Mendes** **Universidade de Coimbra**
Orientador	**Professor Doutor Pedro Mariano Simões Neto** **Professor Auxiliar da Universidade de Coimbra**

Coimbra, Setembro, 2014

“Todo o gesto é um acto revolucionário.”

Bernardo Soares, heterônimo de Fernando Pessoa, em Livro do Desassossego, 1914.

Agradecimentos

A dissertação de mestrado é sem dúvida o culminar de uma das etapas mais importantes da minha formação académica, e apesar de ser em alguns momentos um processo solitário, muitos foram aqueles que de forma directa ou indirecta me acarinharam, incentivaram e apoiaram nesta fase.

Quero aqui, neste espaço, expressar a minha homenagem e reconhecimento às pessoas que de certa forma colaboraram ou contribuíram para que a realização desta dissertação fosse possível.

Em primeiro lugar agradeço aos meus pais, João e Graciete, e irmão, João Paulo, pela confiança, apoio e dedicação que tiveram ao longo de toda a minha vida, em especial na fase final do culminar da minha vida académica.

Ao meu orientador, Professor Doutor Pedro Neto pela receptividade e na orientação desta dissertação, pelas explicações, ideias, sugestões e orientação científica.

Ao Mestre Nuno Mendes pela ajuda, ideias e sugestões para passar algumas dificuldades a nível da programação.

À minha família, em especial ao meu tio e padrinho João Pedro, primo Diogo e Nelita, pela preocupação e carinho.

Aos amigos que fiz em Coimbra, pelo privilégio e sorte de poder contar com o seu companheirismo e boa disposição durante todos estes anos de formação. Agradeço aos meus amigos, Norberto Ramos, Ricardo Silva e Cátia Costa que me ajudaram na revisão e *design* de alguns pontos desta dissertação.

Um agradecimento à Liliana Chaves pela paciência, compreensão, apoio e por acreditar sempre em mim, mesmo quando eu não acreditava.

Às amizades de longa data, pelo entusiasmo e encorajamento contínuo que me presenteiam dia após dia.

Por fim, agradeço-te a ti Coimbra, cidade do conhecimento e dos estudantes de capa negra, pela tua magia, mística, história, e simplicidade.

A todos, um caloroso e sincero obrigado!

Resumo

Os robôs industriais têm vindo a ter um papel importante no sector industrial. A sua abrangência às pequenas e médias empresas (PMEs) é importante, mas a complexidade da programação é um grande entrave. Esta necessita de técnicos especializados, e no caso da programação *offline* exige um grande investimento em programas informáticos e um longo período de treino até que se consiga uma utilização eficiente. No sentido de naturalizar a interacção com os robôs tem existido ao longo dos últimos anos um grande esforço relativamente ao desenvolvimento de interfaces Homem-robô (HRI).

Esta dissertação pretende contribuir para essa investigação, sendo essa interacção efectuada recorrendo aos gestos da mão humana. São considerados vários gestos, tanto estáticos como dinâmicos, e quando estes são reconhecidos, é dada uma ordem de operação ao robô. Ou seja, o robô executa a operação desejada com um gesto executado pelo utilizador.

O reconhecimento dos gestos estáticos e dinâmicos é feito recorrendo a dois métodos diferentes. Para o reconhecimento de gestos estáticos recorre-se a uma rede neuronal artificial (ANN) enquanto que para o reconhecimento de gestos dinâmicos utilizam-se modelos ocultos de *Markov* (HMMs). Estes métodos mostram que são adequados neste tipo de aplicações, obtendo-se uma taxa de reconhecimento global de 99,83%, num grupo de 12 gestos estáticos, e 94,20%, num grupo de 10 gestos dinâmicos, efectuados pelo mesmo utilizador que gerou os dados de treino e aprendizagem. Para um utilizador diferente verifica-se uma taxa de reconhecimento global de 99,75% e 93,60%, respectivamente.

Com o sistema proposto torna-se assim possível que um utilizador interaja com um robô industrial de forma mais intuitiva e natural. Assim o utilizador pode operar um robô, através de gestos, fazendo com que este execute movimentos, guarde posições, voltar a posições previamente guardadas, programar ciclos de trabalho, entre outras tarefas relevantes à empresa onde está inserido.

Palavras-chave: Gestos, *Le*ap *motion*, Segmentação e reconhecimento de gestos, Redes Neuronais Artificiais (ANN), Modelo Oculto de *Markov* (HMM), Robô.

Abstract

Industrial robots have been increasingly important in industry. They are especially important in small and medium enterprises (SME) but robot programming is still a difficult task. This requires skilled workers, a relative high investment in offline software and long periods to learn such software. Great efforts have been made to create intuitive human-robot interfaces (HRI).

This thesis is about HRI using gestures. We consider static and dynamic gestures applied to control a robot. Static gestures are recognized using artificial neural networks (ANN) and dynamic gestures recurring to hidden Markov models (HMM). It was obtained a recognition rate of 99,83%, in a library of 12 static gestures, and 94,20% in a library of 10 dynamic gestures, all these gestures performed by the same user that trained the system. When it is tested by a user that do not trained the system, it was obtained a recognition rate of about 99,75% and 93,60%, respectively.

The proposed solution makes possible a user to interact with a robot in an intuitive way. Thus, any user without technical knowledge in robot programming is able to interact with a robot using gestures.

Keywords Gestures, Leap motion, Segmentation and gesture recognition, Artificial Neural Network (ANN), Hidden Markov Models (HMM), Robot.

Índice

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

SIMBOLOGIA E SIGLAS

Simbologia

Redes neuronais:

α – Coeficiente de aprendizagem

β – *Momentum*

φ^i – Função de activação

b – *bias*

E – Função de erro

k – Número do neurónio numa determinada camada

m^n – Número total de neurónios na camada de saída

T – *Output* desejado

W – Peso

y^1 – Neurónios da camada de entrada

y^i – Neurónios da camada escondida

y^n – Neurónios da camada de saída

Modelo oculto de *Markov*:

α – Variável *forward*

β – Variável *backward*

δ – Variável para guardar probabilidade de sequência

λ – Modelo oculto de *Markov*

Ψ – Variável para guardar sequência de estados

ξ – Probabilidade do processo se encontrar num determinado estado num determinado instante e se encontrar noutro estado no instante seguinte

γ – Probabilidade do processo se encontrar num determinado estado num determinado instante

A – Matriz de transição

B – Matriz de emissão

N – Número de estados

M – Número de observações

O – Sequência de observações

Q – Sequência de estados

***Features*:**

ζ – Vector *input* da ANN

ς – Vector *output* da ANN

Ψ – Vector *input* do HMM

Γ – Vector *output* do HMM

Siglas

ANN – Rede Neuronal Artificial (*Artificial Neural Network*)

CPU – Unidade Central de Processamento (*Central Processing Unit*)

DEM – Departamento de Engenharia Mecânica

EMG – Electromiografias

FCTUC – Faculdade de Ciências e Tecnologia da Universidade de Coimbra

GD – Gesto Dinâmico

GE – Gesto Estático

HMM - Modelo Oculto de *Markov* (*Hidden Markov Model*)

HRI – Interacção Humano-Robô (*Human-Robot Interaction*)

LED – Díodo Emissor de Luz (*Light Emitting Doide*)

PMEs – Pequenas e Médias Empresas

RNN – Rede Neuronal Recorrente (*Recorrente Neural Network*)

SDK – *Kit* de Desenvolvimento de *Software* (*Software Development Kit*)

SVM – Máquina de vectores de suporte (*Support Vector Machine*)

USB – *Universal Serial Bus*

1. INTRODUÇÃO

A comunicação é algo natural entre dois ou mais seres humanos. Esta, visa trocar informações por diversas vias, como escrita, oral ou outra. Na expressão oral, para além de a pessoa falar, há também um conjunto de aspectos que o orador, inconscientemente, executa e que o ouvinte os interpreta de forma natural. Aspectos esses que estão relacionados com a forma como o orador fala (tom, compassos de espera, entre outros) e outros que não estão directamente relacionados com a fala (gestos, movimentos corporais, olhares, entre outros). Aos primeiros chamamos "Para-Verbais" e aos segundos "Não-Verbais". Assim, segundo um estudo de *Albert Mehrabian* a comunicação oral e os seus aspectos ficaram conhecidos como a regra dos 7%-38%-55%, como representado na Figura 1.1 (Mehrabian, 1971).

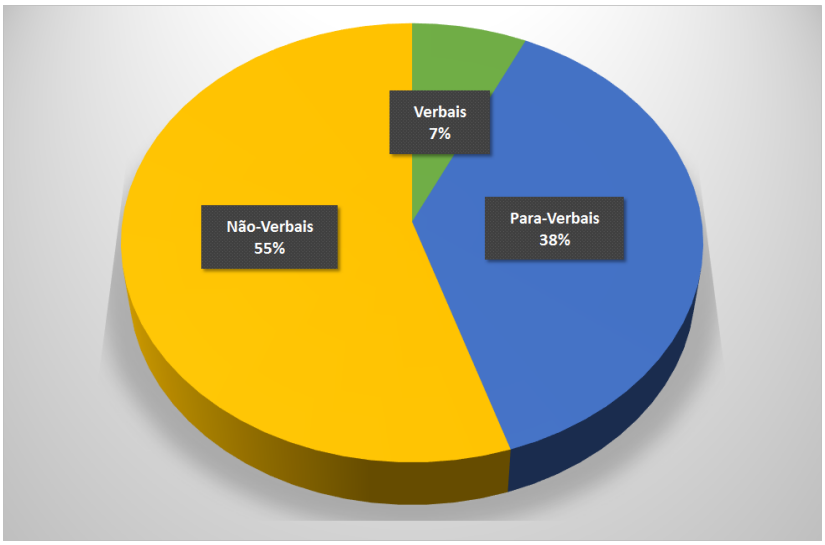


Figura 1.1. Regra dos 7%-38%-55%.

A interacção convencional entre Homem e máquina é algo que nada se assemelha às comunicações entre seres humanos. Assim, o Homem teve de desenvolver dispositivos para que essa comunicação fosse mais natural, surgindo, entre outros, ratos, teclados, comandos de consolas de jogos, *joysticks*. Como grande parte da comunicação humana recai sobre os aspectos "Não-Verbais", nesta dissertação pretende-se recriar a comunicação com robôs industriais, através de gestos.

1.1. Motivação e Objectivos

A programação de robôs industriais, pelo método tradicional, Figura 1.2, requer longos períodos de tempo, necessita de técnicos especializados e é uma tarefa árdua e complexa (Pieskä, et al., 2012).

Figura 1.2. Exemplo de comando usado para a programação por método tradicional, *Teach pedant*.

Um robô industrial é uma máquina desenhada, equipada e programada para desempenhar tarefas específicas. Este não possui qualquer tipo de memória cognitiva, ou seja, não é capaz de interagir de forma intuitiva com as pessoas e vice-versa. A reprogramação de um robô industrial torna-se, deste modo, uma tarefa difícil para um utilizador sem experiência na área.

Uma interacção humano-robô (HRI em inglês *human-robot interaction*) confiável e mais intuitiva é algo que tem vindo a ser alvo de estudo há já algum tempo. Pretende-se que a comunicação com um robô industrial seja o mais semelhante possível com relações interpessoais. Assim, o ideal seria a interacção poder ser feita por qualquer pessoa que pretenda trabalhar com o robô (Neto, et al., 2013). Este facto levaria a um crescente uso de robôs em PMEs, pois estes são considerados elementos fundamentais em sistemas flexíveis de manufactura (Pieskä, et al., 2012).

Posto isto, o principal objectivo deste trabalho é poder contribuir com mais um passo nesta caminhada relacionada com o desenvolvimento da HRI. Deste modo pretende-se recorrer a um sensor óptico, *Leap motion*, com o intuito de substituir a programação tradicional por gestos previamente definidos.

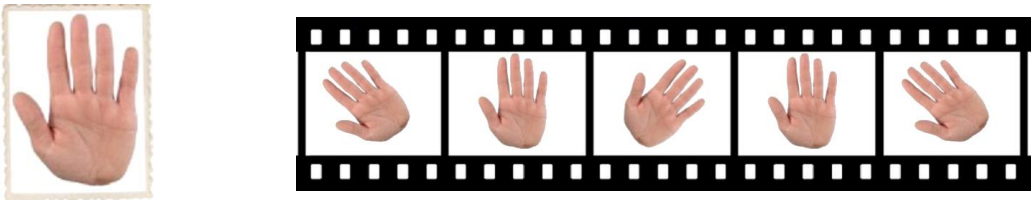
1.2. Problema

Muitos dos métodos de programação de robôs presentes em publicações e estudos não são adequados para PMEs. Exemplo disso é a programação *offline* que tem como vantagem ser efectuada sem interagir directamente com o robô, ou seja, o robô continua a desempenhar a tarefa que lhe tinha sido confinada enquanto o técnico o reprograma. No entanto, tem como desvantagem um grande investimento em programas informáticos específicos, e, longos períodos de treino para uma utilização eficiente, o que inviabiliza o seu uso por grande parte das PMEs (Pieskä, et al., 2012).

Mais recentemente surgiu o reconhecimento de gestos como um método promissor. Este método tenta combater as barreiras artificiais existentes noutros métodos tentando que o utilizador interaja com o robô de forma mais intuitiva (Pieskä, et al., 2012). O problema consiste na escolha do equipamento para o seu reconhecimento, pois factores como o custo, o tipo de aplicação, a fiabilidade e a portabilidade têm de ser tidos em conta (Neto, et al., 2013).

1.3. Abordagem proposta

Existem dois grupos de gestos que se pretendem reconhecer, esses grupos são gestos estáticos e gestos dinâmicos. Gestos estáticos são aqueles cujo movimento da mão ou dedos não é relevante para a sua percepção, como por exemplo o gesto de STOP, Figura 1.3 (a), já nos gestos dinâmicos esse movimento torna-se essencial para o seu entendimento, como por exemplo o gesto de ADEUS, Figura 1.3 (b).



(a) (b)

Figura 1.3. (a) Gesto estático, STOP. (b) Gesto dinâmico, ADEUS.

O que se propõe é reconhecer esses dois grupos de gestos, em simultâneo, através do equipamento *Leap motion*. Para o reconhecimento de gestos estáticos recorre-se a redes

neuronais artificiais (ANNs) e para o reconhecimento de gestos dinâmicos opta-se pelo uso de modelos ocultos de *Markov* (HMMs), como ilustrado na Figura 1.4.

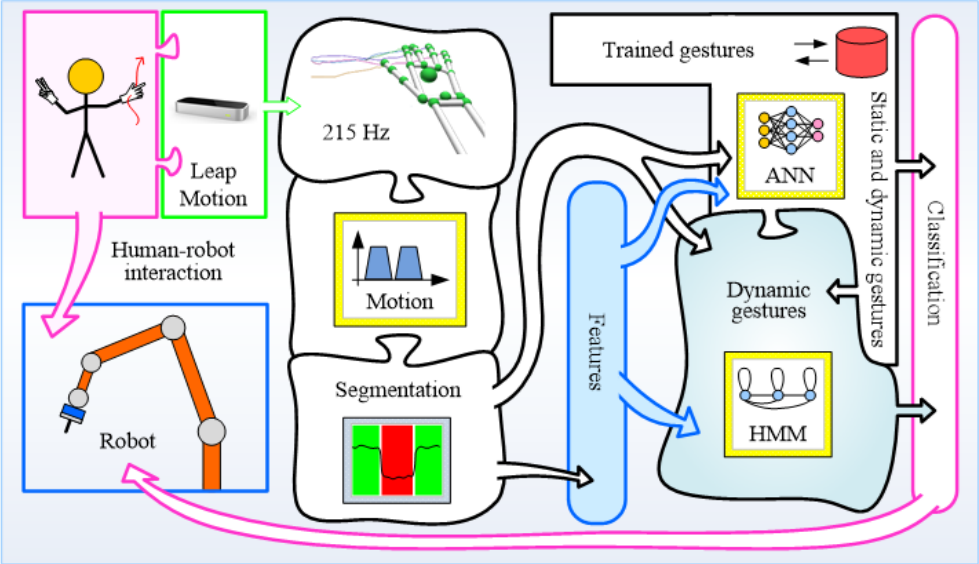


Figura 1.4. Abordagem proposta.

1.4. Organização de conteúdos

O capítulo um é o capítulo introdutório. Contextualiza-se o tema, referem-se a motivação e objectivos, bem como o problema e a forma de abordagem proposta.

O capítulo dois refere-se ao estado da arte. Apresentam-se vários tipos de sensores utilizados no reconhecimento de gestos. É explanado, com algum detalhe, o dispositivo utilizado, *Leap motion*. Finalizando com a forma como se reconhecerão os gestos, tanto os estáticos como os dinâmicos.

O capítulo três é aquele que dá nome a esta dissertação. São apresentados os métodos utilizados para o reconhecimento de gestos estáticos, redes neuronais artificiais, e dinâmicos, modelo oculto de *Markov*. Também são apresentadas as *features* e como se define a distinção entre gestos estáticos e dinâmicos.

O capítulo quatro diz respeito aos testes e resultados. São apresentados os gestos considerados, assim como os parâmetros utilizados. Também são apresentados os resultados, dos diversos testes, de forma a se verificar a fiabilidade deste sistema de controlo.

O capítulo cinco representa as críticas globais sobre este trabalho bem como propostas de trabalhos futuros.

O capítulo seis, e último, encontram-se listadas as referências bibliográficas mais relevantes que foram consultadas ao longo da elaboração desta dissertação.

2. ESTADO DA ARTE

Neste capítulo serão apresentados alguns dispositivos de interacção, em especial o *Leap motion* e como se procede ao reconhecimento de gestos.

2.1. Dispositivos de interacção

Com o avanço tecnológico o Homem começou a desenvolver dispositivos de interacção com o intuito de "naturalizar" a HRI. Neste sentido, muitos destes dispositivos foram integrados, na indústria de videojogos, como controladores. Existem, essencialmente, quatro tipos diferentes de tecnologia, luva de dados, sensores de base magnética com e sem sensores inerciais, visão e híbridos. Este último combina as tecnologias dos anteriores.

A tecnologia luva de dados consiste numa luva com sensores dispersos nesta, que o utilizador veste normalmente. A empresa C*yberGlove Systems LLC* dedica-se ao desenvolvimento e comercialização deste tipo de tecnologia, sendo a Figura 2.1 um dos seus produtos, a *CyberGlove II* (Bianchi, et al., 2012).

Figura 2.1. Exemplo de luva de dados, *CyberGlove II.* **(CyberGlove Systems LLC, 2014)**

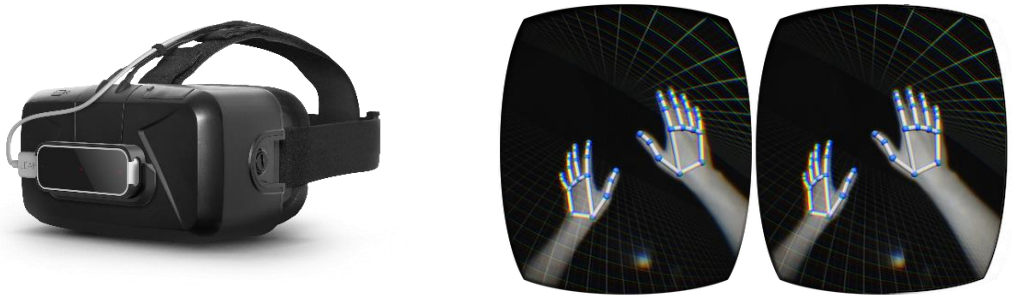
A tecnologia de sensores de base magnética com ou sem sensores inerciais comumente utilizados são os acelerómetros e giroscópios. Um exemplo deste tipo de tecnologia é o comando da consola de videojogos da empresa *Nintendo* (Neto, et al., 2009) (Neto, et al., 2012). Outro exemplo é o uso de electromiografias (EMG) com sensores inerciais (Wolf, et al., 2013).

Figura 2.2. Exemplo de magnéticos com ou sem sensores inerciais, Comando *Wii.* **(Nintendo Co., Lda, 2014)**

A tecnologia de visão pode ser utilizada para o reconhecimento de mãos, face ou corporal. Esta tecnologia não requer que o utilizador agregue a si qualquer tipo de dispositivo, ou seja, o dispositivo encontra-se fixo gerando um campo de visão e o utilizador apenas tem de se encontrar nesse campo para que interaja. Um exemplo de dispositivos que incorporam esta tecnologia é o *Kinect*, da empresa *Microsoft*.

Figura 2.3. Dispositivo *Kinect.* **(Microsoft Corporation, 2014)**

Outro dispositivo equivalente é o *Leap motion*, da empresa com o mesmo nome, que é o dispositivo utilizado nesta dissertação. Recentemente surgiu um novo equipamento que integra o *Leap motion*, *Leap motion VR*, Figura 2.4, recriando realidade virtual e realidade aumentada com capacidade de visão nocturna, visão infravermelha.



(a) (b)

Figura 2.4. (a) Dispositivo *Leap motion VR.* **(Leap Motion, Inc, 2014)**, (b) Realidade virtual **(Leap Motion, Inc, 2014)**

2.1.1. Leap motion

O *Leap motion* (Figura 2.5) foi o primeiro produto da empresa *Leap Motion, Inc.* criada em 2010, por *Michael Buckwald* e *David Holz*. A empresa está sedeada nos Estados Unidos da América, na cidade de São Francisco, estado da Califórnia. Esta dedica-se à produção e venda de sensores de movimento para interacção Homem-computador, através

de dispositivos que transformam posições e movimentos das mãos, dedos ou ferramentas idênticas a estes em *inputs* para o computador.

Figura 2.5. Dispositivo *Leap motion.* **(Leap Motion, Inc, 2014)**

Quanto ao *design*, o *Leap motion*, sofreu uma evolução significativa, como se pode observar na Figura 2.6. Desde Agosto de 2011 que o seu *design* tem vindo a sofrer alterações, passando de um aparelho relativamente grande e pouco apelativo, até ao actual *design*, simples e discreto.

Figura 2.6. Evolução do *Leap motion.* **(Hedlerfog, 2014)**

Foi oficialmente apresentado a 21 de Maio de 2012, mas apenas em Julho de 2013 teve uma grande aceitação por parte do mercado. Este dispositivo foi concebido de modo a erradicar a utilização de ratos e teclados convencionais na modelação 3D.

O *Leap motion* é um pequeno dispositivo de interface *USB* 2.0, no entanto possui um conector *micro-USB* 3.0. A melhoria no desempenho é mínima, contudo a empresa afirma que o facto do dispositivo não incorporar um cabo *USB* 3.0 é devido à inexistência

de um cabo com flexibilidade suficiente para que não o movesse acidentalmente. Este dispositivo é compatível com os sistemas operativos *Linux*, *Mac OS X* e *Windows*. É bastante leve, estrutura concebida em alumínio, borracha, na parte inferior, e um vidro escurecido, na parte superior. Apresenta também um LED de estado (verde para ligado e em estado normal e vermelho para quando o mesmo não está pronto a utilizar). Foi pensado sob o ponto de vista do utilizador, ou seja, não é necessário ter condições perfeitas para a sua utilização como luz e plano de apoio. Utiliza três LEDs infravermelhos e duas câmaras monocromáticas infravermelhas que identificam uma superfície hemisférica a uma distância de cerca de sessenta centímetros. Essa superfície encontra-se esquematizada na Figura 2.7.

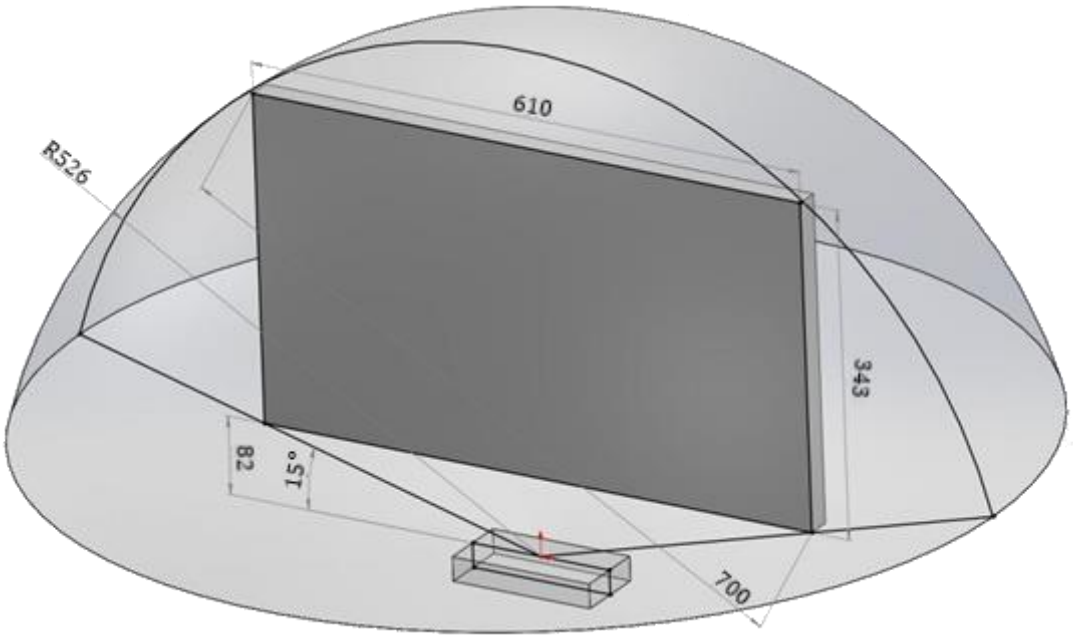


Figura 2.7. Superfície hemisférica. **(Leap Motion, Inc, 2014)**

O conjunto dos três LEDs geram um padrão 3D de pontos de luz infravermelha, cerca de 700 milhões de pontos, e as câmaras geram no máximo de 290 fps (fotogramas por segundo), sendo que em normal funcionamento geram 215 fps de dados reflectidos (Leap Motion, Inc, 2014). Estes são enviados para o computador, onde é analisado pelo *software* do dispositivo de modo a converter os dados 2D gerados pelas câmaras em 3D. O processo é descrito como "matemática complexa" e não é divulgado pela empresa. Dedos ou ferramentas idênticas a estes, como canetas, lápis, entre outros, que obstruam a área de visão

do dispositivo são identificados com uma precisão espacial de cerca de 0,01 mm. A Figura 2.8 apresenta todos os componentes que constituem este dispositivo.

Figura 2.8. Composição do *Leap motion.* **(Limer, 2014)**

Possui um modo próprio com menor sensibilidade para que o utilizador o possa usar em ambientes com muita luz, sendo esse modo denominado por *robust mode*.

As suas principais limitações são a sobreposição de dedos, convertendo estes em apenas um ponteiro de forma irregular e sem sentido. Há aplicações, nomeadamente o *Touchless for Windows*, em que há uma dificuldade imensa no seu uso, ficando a seta hiperactiva e gestos que simplesmente não funcionam (Ziesecke, 2014).

O *Leap motion* funciona mediante uma relação entre a alta precisão e a taxa de fotogramas monitorizados, devolvendo a posição discreta, gestos e movimentos daquilo que interferir no campo de visão. Quando são detectadas mãos, dedos, ferramentas ou gestos, o *software* atribui uma identificação. Identificação essa que permanece constante enquanto a entidade interferir no campo de visão. Se a detecção for perdida e recuperada o *software* atribui uma nova identificação, ou seja, a entidade jamais será conhecida pela mesma identificação.

Estes dados são classificados, primariamente, por o que é identificado, assim a Figura 2.9 representa as diferentes classes que o dispositivo capta.

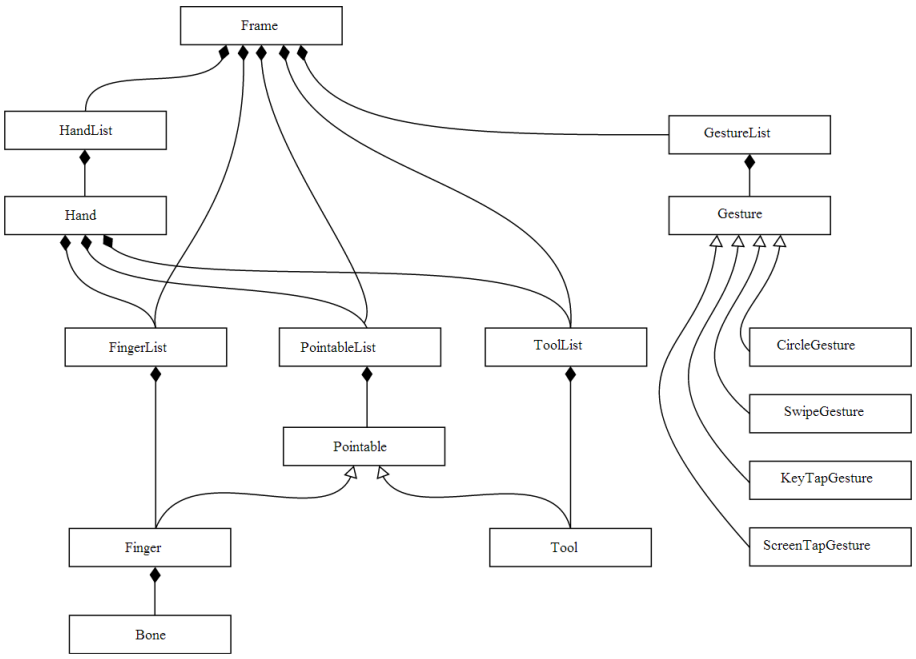


Figura 2.9. Classes identificadas em cada fotograma. **(Leap Motion,Inc, 2014)**

Cada classe subdivide-se em propriedades e métodos, de modo a que o acesso aos dados seja facilitado.

De modo a possibilitar a interpretação dos dados é necessário ter em conta algumas considerações, como o sistema de coordenadas, as unidades e as propriedades mais relevantes, para este trabalho. O sistema de coordenadas é cartesiano e rege-se pela regra da mão direita. A origem está no centro do dispositivo, como ilustrado na Figura 2.10. Este sistema de coordenadas é independente da orientação do dispositivo, ou seja, o indicador luminoso do dispositivo não define nenhum sentido. Os eixos X e Z definem o plano horizontal, onde o eixo X é paralelo à parte mais comprida do dispositivo, e o eixo Z tem valores positivos afastando-se do ecrã do computador. O eixo Y é o eixo vertical e é positivo para cima.

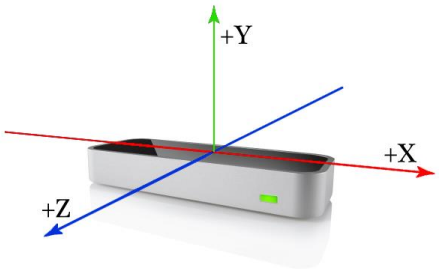


Figura 2.10. Sistema de eixos do *Leap motion.* **(Leap Motion, Inc., 2014)**

As unidades adjacentes a cada tipo de dado encontram-se na Tabela 2.1.

Tabela 2.1. Unidades dos dados do *Leap motion*. **(Leap Motion, Inc., 2014)**

	Unidade	**Símbolo**
Distâncias	milímetros	mm
Tempos	microssegundos	μs
Velocidades	milímetros/segundo	mm/s
Ângulos	radianos	rad

As propriedades e métodos de cada classe são apresentados dividindo-se pelas classes mãos e dedos.

- **Mãos**

A classe mãos contém informações sobre identidade, posição e características da mão detectada, o braço que a antecede, e listas de dedos ou ferramentas associados à mão. O *software* já distingue se a mão detectada é direita ou esquerda ou se está fechada em punho, uma actualização com a versão 2.0 do SDK (*Software Developer Kit*). Esta actualização, usa um modelo computacional da mão humana de modo a que seja possível prever, identificando, partes da mão que não se encontrem visíveis para o dispositivo. Também consegue distinguir mais do que duas mãos, mas é recomendado, para um reconhecimento eficiente, o uso de apenas duas.

O vector normal à palma da mão (*PalmNormal*) aponta perpendicularmente para fora da mão e o vector direcção (*PalmDirection*) aponta para a frente, como ilustrados na Figura 2.11.

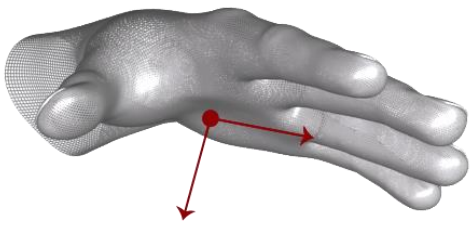


Figura 2.11. Vectores normal e direcção da palma da mão. **(Leap Motion, Inc., 2014)**

O vector centro da esfera (*SphereCenter*) e raio da esfera (S*phereRadius*) pretendem prever a curvatura da mão através de uma esfera virtual, como mostra a Figura 2.12.

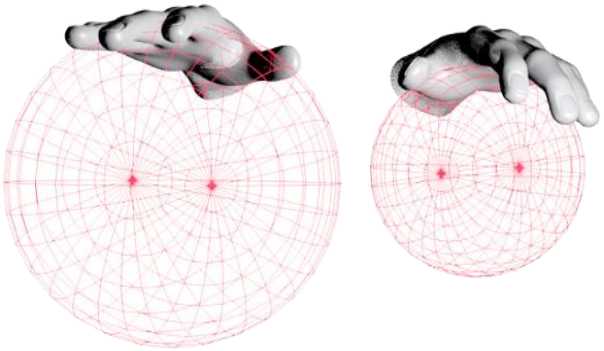


Figura 2.12. Esfera virtual formada para curvatura da mão. **(Leap Motion, Inc., 2014)**

Outras propriedades importantes são o vector velocidade da palma da mão (*PalmVelocity*) e a função agarrar (*GrabStrengh*) que varia entre $[0;1]$, onde zero representa a mão aberta e um a mão fechada.

- **Dedos**

A classe dedos contém informações sobre identidade, posição e características dos dedos detectados, bem como a diferenciação de cada dedo e os seus ossos constituintes. Estes últimos surgiram com a actualização para a versão 2.0 do SDK, assim a cada mão monitorizada são automaticamente identificados cinco dedos.

O vector direcção (*Direction*) de cada dedo aponta para a frente deste, como ilustrado na Figura 2.13.

Figura 2.13. Vectores direcção de cada dedo. **(Leap Motion, Inc., 2014)**

A distância de toque (*TouchDistance*) varia entre $[-1;1]$ e pretende representar a intenção de toque, como o clicar num botão com um rato. Valores negativos significam toque. O valor neutro, zero, apenas indica que o dedo está prestes a interferir na zona de toque. Valores positivos significam existência de dedos a "pairar" (*hovering*). Estas zonas

são delimitadas tendo em conta um plano de toque que se adapta aos movimentos dos dedos e postura da mão.

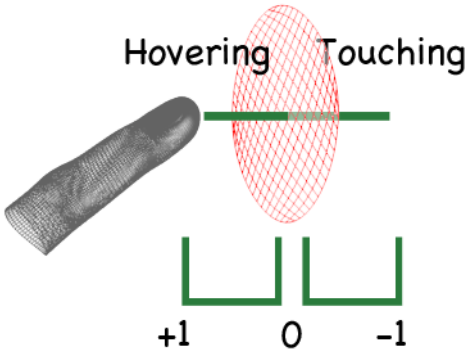


Figura 2.14. Distância de toque. **(Leap Motion, Inc., 2014)**

Outra propriedade importante é o vector velocidade de cada dedo (*TipVelocity*).

2.2. Reconhecimento de gestos

O reconhecimento de gestos pode ser dividido em duas partes, segmentação e o reconhecimento propriamente dito (Mitra & Acharya, 2007). Assim têm surgido diversos trabalhos relacionados com o reconhecimento de gestos usando HMM (Lee & Kim, 1999). Outros autores usam ANN (Nolker & Ritter, 2002), máquina de vectores de suporte (SVM) (Wolf, et al., 2013) e outros recorrem a redes neuronais recorrentes (RNN) para modelar séries temporais (Yamashita & Tani, 2004).

3. SEGMENTAÇÃO E RECONHECIMENTO DE GESTOS

Neste capítulo serão apresentados os métodos utilizados para a segmentação e reconhecimento de gestos, rede neuronal artificial e modelo oculto de *Markov*, assim como as *features*, e a forma como é detectado o movimento, como ilustrado na Figura 1.4.

3.1. Rede neuronal artificial (ANN)

Uma ANN é um modelo computacional inspirado no funcionamento do cérebro humano. É um tipo de inteligência artificial em que a máquina responde a padrões de entrada. O modelo de neurónio artificial foi criado em 1943, pelo fisiologista *Warren McCulloch* e o lógico *Walter Pitts* (McCulloch & Pitts, 1943). Este é constituído por um conjunto de entradas, um conjunto de pesos sinápticos, cada um associado a uma entrada, um elemento de polarização, *bias*, uma unidade de soma e uma unidade de processamento, como representado na Figura 3.1.

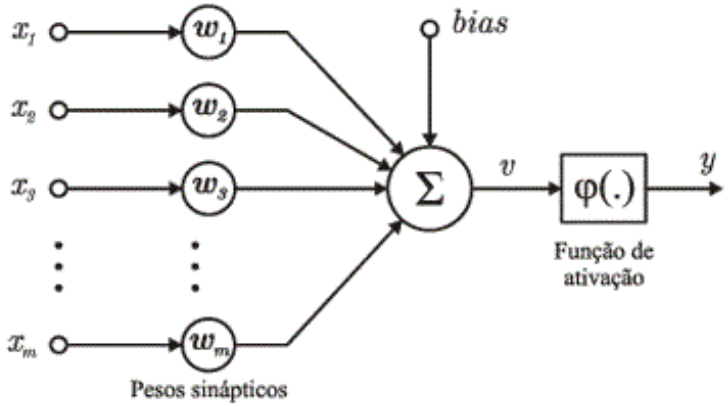


Figura 3.1. Modelo de um neurónio artificial. **(Moreto & Rolim, 2010)**

Assim uma ANN é um sistema que incorpora vários neurónios artificiais ligados entre si, através de pesos sinápticos. Para melhor se classificar uma ANN, agrupam-se os neurónios em camadas. Assim existem dois grupos: as redes de uma única camada e as redes multi-camada. A camada de entrada não é contabilizada, portanto as redes de uma única camada possuem uma camada de neurónios de entrada e uma camada de neurónios de saída. As redes multi-camada, para além das duas referidas atrás, possuem, entre estas, uma ou

mais camadas. Camadas essas que se designam por camadas escondidas. Um complemento a esta classificação é a direcção dos sinais transmitidos entre os neurónios pode definir a arquitectura de uma ANN.

As ANNs *feedforward* são redes onde os neurónios de uma camada, geralmente, não comunicam entre si, e a direcção do sinal é sempre da camada de entrada para a camada de saída, como ilustrado na Figura 3.2.

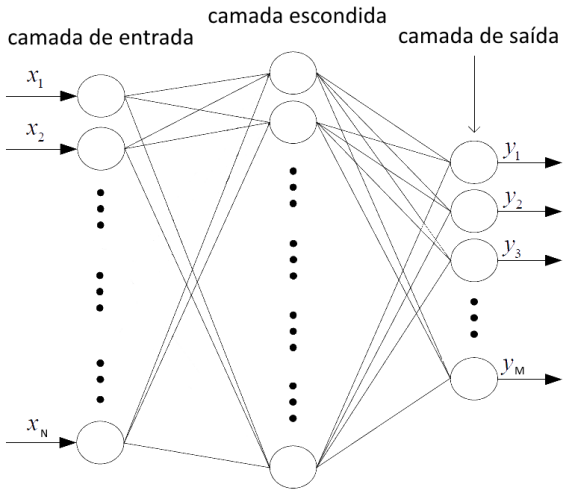


Figura 3.2. Esquema de uma rede neuronal multi-camada *feedforward.* **(Neto, et al., 2013)**

Considere-se agora uma ANN semelhante a esta configuração, onde y^n, y^1 e y^i representam, respectivamente, os neurónios da camada de saída, os neurónios da camada de entrada e os neurónios das camadas escondidas. Existindo *k* neurónios em cada camada e o *output* desejado é *T*. Cada camada tem uma função de activação φ^i. A função de erro é então dada pela equação (3.1) onde m^n representa o número de neurónios da camada *n*, ou seja, camada de saída.

$$E = \frac{1}{2}\sum_{k=1}^{m^n}\left(T_k - y_k^n\right)^2 \tag{3.1}$$

O algoritmo *backpropagation* é utilizado como forma de minimizar esta função de erro. A rede é iniciada com pesos, *W*, aleatórios, contudo este algoritmo ajusta estes valores assim como o valor de *bias*, *b*. O *output* de um neurónio *k* da camada escondida, *i*, pode ser calculado recorrendo à equação (3.2).

$$y_k^i = \varphi^i\left(v_k^i\right) = \varphi^i\left(\sum_{j=1}^{m^{i-1}} W_{k,j}^i y_j^{i-1} + b_k^i\right) \tag{3.2}$$

Os pesos são ajustados pelas equações (3.3) e (3.4), α é o coeficiente de aprendizagem.

$$W_{k,j}^{i}\left(n+1\right)=W_{k,j}^{i}\left(n\right)+\Delta W_{k,j}^{i} \tag{3.3}$$

$$\Delta W_{k,j}^{i}=-\alpha\frac{\partial E}{\partial W_{k,j}^{i}}=\alpha\delta_{k}^{i}y_{k}^{i-1} \tag{3.4}$$

O ajuste do *bias* é dado pela equação (3.5).

$$\Delta b_{k}^{i}=\alpha\delta_{k}^{i} \tag{3.5}$$

Sendo δ dado pela equação (3.6)

$$\delta_{k}^{i}=\begin{cases}\left(T_{k}-y_{k}^{n}\right)\dot{\varphi}^{i}\left(v_{k}^{i}\right) & \text{se } i=n \text{ (camada de saída)}\\ \left(\sum_{j=1}^{m^{i+1}}\left(\delta_{k}^{i+1}W_{k,j}^{i+1}\right)\right)\dot{\varphi}^{i}\left(v_{k}^{i}\right) & \text{se } 2\le i<n \text{ (camadas escondidas)}\end{cases} \tag{3.6}$$

A função de activação é sigmóide assimétrica, equação (3.7), e a sua derivada é dada pela equação

$$\varphi\left(v\right)=\frac{1}{1+e^{-v}} \tag{3.7}$$

$$\dot{\varphi}\left(v\right)=\varphi\left(v\right)\left[1-\varphi\left(v\right)\right] \tag{3.8}$$

O parâmetro *momentum*, β, é introduzido com o intuito de diminuir o tempo da fase de treino da ANN, reduzindo, também, o perigo de saturação dos pesos. O valor típico deste parâmetro é de 0,1 (Haylin, 1999). Assim o valor do peso a cada iteração pode ser obtido através da equação (3.9).

$$W_{p}=W_{p-1}+\Delta W+\beta\left(W_{p-1}-W_{p-2}\right) \tag{3.9}$$

O índice *p* representa uma iteração do processo de treino.

3.2. Modelo oculto de *Markov* (HMM)

Um HMM é um modelo de distribuição de probabilidades, com estados finitos, *N*, onde para cada sequência de observações há uma distribuição de probabilidades de transição, matriz de transição, *A*, que por sua vez é associado a um símbolo para cada observação, *M*, com os quais se forma a distribuição de probabilidades de observação de um

símbolo, matriz de emissão, *B*, e também é definido um vector de estado inicial, π. Assim um HMM é definido pela equação (3.10)

$$\lambda = (N, M, A, B, \pi) \tag{3.10}$$

De forma mais compacta pode-se reescrever a definição de um HMM como demonstrado na equação (3.11).

$$\lambda = (A, B, \pi) \tag{3.11}$$

Esta simplificação acontece pois a matriz *A* tem de dimensão o número de estados por o número de estados, *N*x*N*, e a matriz a *B* têm de dimensão o número de estados por o número de símbolos de observação, *N*x*M*.

Tendo *A*, *B* e π, dada uma sequência de observações $O = O_1\ O_2\ ...\ O_T$ pode ser determinado pelo HMM o que deu origem a essa sequência, através dos seguintes passos:

1) Escolher um estado inicial $q_1 = S_i$, contido em π;
2) Marcar esta observação como instante $t = 1$;
3) Fazer corresponder uma observação do estado S_i a *B*, $b_i(O_t)$;
4) Transitar para um novo estado $q_{i+1} = S_j$, de acordo com *A;*
5) Repetir os passos 3 e 4, somando 1 a t, até ao fim das observações, $t < T$.

De modo a clarificar este conceito, tome-se como exemplo um conjunto de três urnas, cada uma contendo cinco bolas de diferentes cores, como ilustrado na Figura 3.3.

Figura 3.3. Urnas com bolas de cores diferentes.

O procedimento consiste em retirar aleatoriamente uma bola de uma urna, observar a cor e colocá-la na mesma urna. Repete-se o processo para as restantes urnas. Por só interessar a sequência de cores diz-se que a sequência das urnas é oculta.

Assim na Tabela 3.1 é apresentado o HMM representante deste problema.

Tabela 3.1. HMM ilustrativo.

Símbolo	Significado	Aplicação
N	Número de estados	Número de urnas: 3
M	Número de símbolos	Número de cores diferentes: 5
A	Distribuição de probabilidades de transição de estado	Aleatoriedade na escolha das urnas
B	Distribuição de probabilidades de observação de um símbolo	Distribuição de cores em cada urna
π	Distribuição de estado inicial	Probabilidade de ser seleccionada uma urna inicialmente

Existem três problemas no uso de HMM. Esses problemas são:

- Problema 1: Dado uma sequência de observações $O = O_1\ O_2\ ...\ O_T$ e o HMM $\lambda = (A, B, \pi)$, como se pode prever a sequência que lhe deu origem, ou seja, $P(\mathrm{O} \mid \lambda)$?
- Problema 2: Dado uma sequência de observações $O = O_1\ O_2\ ...\ O_T$ e o HMM $\lambda = (A, B, \pi)$, qual a sequência de estados $Q = q_1\ q_2\ ...\ q_T$ que melhor caracterize a sequência de observações?
- Problema 3: Como ajustar o HMM $\lambda = (A, B, \pi)$ de forma a maximizar $P(\mathrm{O} \mid \lambda)$?

Surgem como resposta a estes problemas três algoritmos, um para cada problema. Assim surge para o Problema 1, problema de avaliação, o algoritmo *forward-backward*, para o Problema 2, problema de decodificação, o algoritmo de *Viterbi*, e para o Problema 3, problema de ensinar, o algoritmo de *Baum-Welch*. De notar que para cada acção que se pretenda reconhecer é necessário o uso de um HMM, ou seja, no caso desta dissertação é necessário um HMM para cada gesto dinâmico.

3.2.1. Algoritmo *forward-backward*

Este algoritmo é composto por dois procedimentos independentes mas capazes de resolver eficientemente, por via computacional, o Problema 1. Recorre à programação dinâmica simplificando assim o processo de cálculo. Analisando cada procedimento separadamente, considere-se a variável *forward* definida pela equação (3.12).

$$\alpha_t(i) = P(O_1\ O_2\ ...\ O_t, q_t = S_i \mid \lambda) \tag{3.12}$$

Ou seja, a variável *forward* é definida como a probabilidade de uma sequência de observações parcial $O = O_1\ O_2\ ...\ O_T$ e do estado S_i no instante t (estado de uma sequência de estados fixa $Q = q_1\ q_2\ ...\ q_T$), sabendo o HMM, λ.

1) **Inicialização**

$$P(O_1, q_1 = S_i \mid \lambda) = \alpha_1(i) = \pi_i b_i(O_1) \qquad 1 \le i \le N \tag{3.13}$$

A variável *forward* para $t = 1$ é a probabilidade do estado S_i e a observação inicial, O_1.

2) **Indução**

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i)\, a_{ij}\right] b_j(O_{t+1}) \qquad \begin{matrix} 1 \le t \le T-1 \\ 1 \le j \le N \end{matrix} \tag{3.14}$$

Como ilustrado na Figura 3.4, o estado S_j no instante $t+1$ é obtido recorrendo aos *N* estados possíveis, $S_i, 1 \le i \le N$, do instante anterior, t.

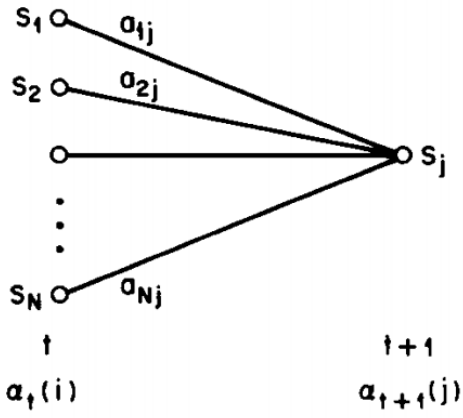


Figura 3.4. Sequência de operações na computação da variável *forward* $\alpha_{t+1}(j)$. **(Rabiner, 1989)**

3) Finalização

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.15}$$

A probabilidade da sequência que lhe deu origem, $P(O|\lambda)$, é a soma de todas as variáveis de *forward* finais, $\alpha_T(i)$.

De modo análogo, apresenta-se o procedimento *backward*. Considere-se agora a variável *backward* definida pela equação (3.16).

$$\beta_t(i) = P(O_{t+1}\ O_{t+2}\ \ldots O_T, q_T = S_i \mid \lambda) \tag{3.16}$$

Ou seja, a variável *backward* é definida como a probabilidade de uma sequência de observações parcial $O = O_{t+1}\ O_{t+2}\ \ldots O_T$, sabendo o estado S_i no instante t (estado de uma sequência de estados fixa $Q = q_1\ q_2\ \ldots q_T$) e o HMM, λ.

1) Inicialização

$$\beta_T(i) = 1 \qquad 1 \le i \le N \tag{3.17}$$

Define-se arbitrariamente que a variável *backward* para todos os estados finais é igual a 1.

2) Indução

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \qquad \begin{matrix} t = T-1,\ T-2,\ \ldots,1 \\ 1 \le i \le N \end{matrix} \tag{3.18}$$

Como ilustrado na Figura 3.5, o estado S_i no instante t é obtido recorrendo aos N estados possíveis, $S_j, 1 \le j \le N$, do instante seguinte, $t+1$, de acordo com as probabilidades de transição a_{ij}, as probabilidades

de observação no estado $b_j\left(O_{t+1}\right)$ e a sequência de observação parcial no estado j, $\beta_{t+1}\left(j\right)$.

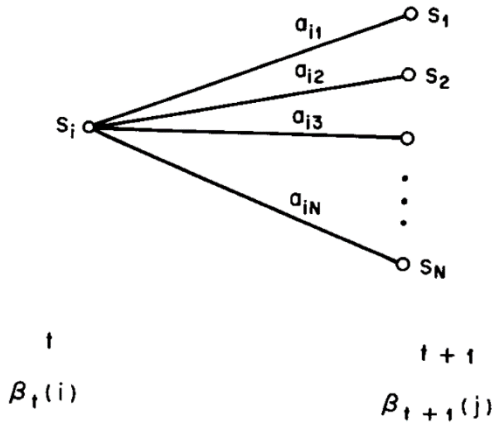


Figura 3.5. Sequência de operações na computação da variável *backward* $\beta_t\left(i\right)$. **(Rabiner, 1989)**

Este algoritmo requer, para ambos os procedimentos, um cálculo computacional de ordem N^2T, em vez de $2TN^T$ como por via do cálculo directo. A Figura 3.6 representa através de um esquema em rede, a parte computacional do algoritmo *forward-backward*.

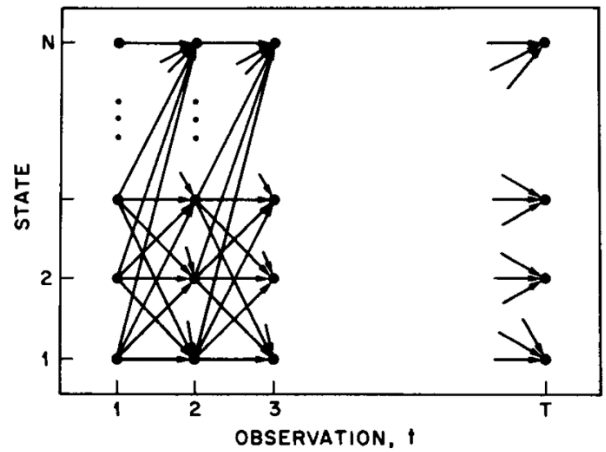


Figura 3.6. Representação das observações *t* em função dos estados *i*. **(Rabiner, 1989)**

3.2.2. Algoritmo *Viterbi*

Este algoritmo é necessário para se verificar qual a sequência de estados $Q = q_1\ q_2\ \ldots q_T$ que melhor caracterize a sequência de observações $O = O_1\ O_2\ \ldots O_T$, ou seja, retirar a sequência de estados que melhor explique a sequência de observações, maximizar $P\left(Q \mid O,\lambda\right)$ e consequentemente maximiza $P\left(Q,O \mid \lambda\right)$, assim surge como resposta ao Problema 2. Assim define-se, pela equação (3.19), a variável que representa a maior

probabilidade ao longo de uma única sequência no instante t, desde as primeiras t observações até ao estado S_i.

$$\delta_t(i) = \max_{q_1, q_2, \ldots, q_{t-1}} P\left[q_1\, q_2 \ldots q_t = i, \mathrm{O}_1\, O_2 \ldots O_t \mid \lambda\right] \tag{3.19}$$

Por indução obtém-se a equação (3.20).

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i) a_{ij}\right] \cdot b_j\left(O_{t+1}\right) \tag{3.20}$$

Para guardar a sequência de estados é necessário recorrer a uma outra variável $\psi_t(j)$, pois a variável $\delta_t(j)$ apenas guarda a probabilidade da sequência de estados.

1) **Inicialização**

$$\begin{aligned} &\delta_1(i) = \pi_i b_i\left(O_1\right) \qquad 1 \le i \le N \\ &\psi_1(i) = 0 \end{aligned} \tag{3.21}$$

Como não se consegue saber qual o melhor estado para o instante $t = 1$, atribui-se $P(q_1, O_1 \mid \lambda)$ à variável $\delta_1(i)$, para se poder iniciar a recursão, e zero a $\psi_1(i)$, pois nenhum estado foi seleccionado como o melhor.

2) **Recursão**

$$\begin{aligned} &\delta_t(i) = \max_{1 \le i \le N}\left[\delta_{t-1}(i) a_{ij}\right] \mathrm{b}_j\left(O_t\right) \qquad 2 \le t \le T \\ &\psi_t(i) = \operatorname*{argmax}_{1 \le i \le N}\left[\delta_{t-1}(i) a_{ij}\right] \qquad 1 \le j \le N \end{aligned} \tag{3.22}$$

Neste passo guarda-se as probabilidades das melhores sequências e as próprias sequências nas variáveis $\delta_t(i)$ e $\psi_t(i)$, respectivamente.

3) **Finalização**

$$\begin{aligned} P^* &= \max_{1 \le i \le N}\left[\delta_T(i)\right] \\ q_T^* &= \operatorname*{argmax}_{1 \le i \le N}\left[\delta_T(i)\right] \end{aligned} \tag{3.23}$$

Este passo finaliza a melhor sequência de estados q_T^* e a sua probabilidade P^*.

4) **"Retrocesso" (Sequência de estados)**

$$q_t^* = \psi_{t+1}\left(q_{t+1}^*\right) \qquad t = T-1,\ T-2,\ \ldots,1 \tag{3.24}$$

Neste passo consegue-se determinar a sequência de estados parcial que melhor caracteriza a observação.

3.2.3. Algoritmo *Baum-Welch*

Este algoritmo surge como resposta ao Problema 3, com o sentido de estimar os três parâmetros de um HMM, *A*, *B* e π. Assim, dando uma sequência de observações pretende-se que o algoritmo ajuste os parâmetros do modelo de forma a maximizar a probabilidade dessa sequência, ou seja, ensinar o HMM.

Não há nenhum método analítico que estime estes parâmetros, contudo determina-se o HMM, λ, tal que $P(\mathrm{O}\,|\,\lambda)$ é maximizada localmente. Este algoritmo é então uma particularização do algoritmo EM (*Expectation-Modification*).

Posto isto, descreve-se o procedimento para a re-estimação (actualização iterativa e melhoramento) dos parâmetros de λ, para tal é definida, na equação (3.25), a variável $\xi_t(i,j)$ como sendo a probabilidade do processo se encontrar no estado S_i no instante t e no estado S_j no instante $t+1$, sabendo a sequência de observações, $O = O_1\ O_2\ \ldots O_T$, e o HMM, λ.

$$\xi_t(i,j) = P\left(q_t = S_i, q_{t+1} = S_j \,|\, \mathrm{O}, \lambda\right) \tag{3.25}$$

Na Figura 3.7 está representada a parte computacional de $\xi_t(i,j)$.

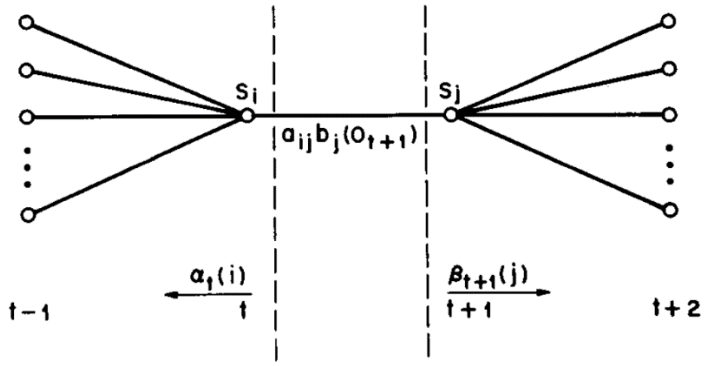


Figura 3.7. Representação da parte computacional de $\xi_t(i,j)$. **(Rabiner, 1989)**

Define-se outra variável $\gamma_t(i)$, equação (3.26), como a probabilidade de estar no estado S_i no instante t, sabendo a sequência de observações e o HMM.

$$\gamma_t(i) = P\left(q_t = S_i \,|\, O, \lambda\right) \tag{3.26}$$

De notar que para um dado instante t a soma desta variável é igual à unidade, equação (3.27).

$$\sum_{i=1}^{N} \gamma_t(i) = 1 \tag{3.27}$$

Estas duas variáveis podem ser relacionadas como se pode verificar pela equação (3.28).

$$\gamma_t(i) = \sum_{i=1}^{N} \xi_t(i, j) \tag{3.28}$$

Somando a variável $\gamma_t(i)$ ao longo de todos os instantes, $1 \le t \le T-1$, equação (3.29), obtém-se o número de vezes que o processo transita para o estado S_i. De igual modo, somando a variável $\xi_t(i, j)$ ao longo de todos os instantes, $1 \le t \le T-1$, equação (3.30), interpreta-se como o número de transições do estado S_i para o estado S_j.

$$\sum_{t=1}^{T-1} \gamma_t(i) \tag{3.29}$$

$$\sum_{t=1}^{T-1} \xi_t(i, j) \tag{3.30}$$

Assim pode-se re-estimar os parâmetros do HMM, sendo dados pelas equações (3.31), (3.32) e (3.33) representando, respectivamente, π, A e B.

$$\bar{\pi} = \text{probabilidade de estar no estado } S_i \text{ no instante } t = 1 = \gamma_t(i) \tag{3.31}$$

$$\bar{a}_{ij} = \frac{\text{número de transições do estado } S_i \text{ para o estado } S_j}{\text{número de transições para o estado } S_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{3.32}$$

$$\bar{b}_j(k) = \frac{\text{número de vezes que foi observado } O_t \text{ no estado } S_j}{\text{número de repetições no estado } S_j} = \frac{\sum_{t=1 \quad O_t}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)} \tag{3.33}$$

Re-estimando os novos parâmetros do HMM, $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$, *Baum* provou que o HMM antigo, $\lambda = (A, B, \pi)$, é o ponto máximo da função de verossimilhança, sendo $\lambda = \bar{\lambda}$, ou o HMM novo, $\bar{\lambda}$, é mais provável que o HMM antigo, λ, ou seja, $P(O|\bar{\lambda}) > P(O|\lambda)$.

Com o intuito de melhorar o reconhecimento de uma dada sequência de observações, repete-se a re-estimação de novos parâmetros utilizando o HMM novo, $\bar{\lambda}$, na vez do HMM antigo, λ, até que seja atingido o critério de finalização. Este critério é normalmente o número de iterações ou uma tolerância. O resultado final do procedimento de re-estimação é conhecido como estimativa de máxima verossimilhança do HMM.

3.3. Features

Uma *feature* é uma propriedade heurística mensurável de algo que se observa, ou seja, é algo que a máquina consegue reconhecer como se fosse um Humano no mundo real (um gesto, uma porta, um computador). Heurística, pois é um método criado com o objectivo de reconhecer algo e mensurável pois só se consegue reconhecer algo se e só se o que é observado é semelhante a uma base de dados previamente construída, ou seja, ao treino do modelo matemático.

Desta forma, as *features* nesta dissertação dividiram-se em dois grupos, pois são utilizados dois métodos distintos, descritos em 3.1 e 3.2, para o reconhecimento de gestos estáticos e dinâmicos, respectivamente.

Apenas são reconhecidos gestos efectuados pela mão direita, e o gesto tem de ser executado a partir do centro do dispositivo com uma margem esférica de cinquenta milímetros de raio de centro em $(x, y, z) = (0,150,0)$. Esta margem foi escolhida com o sentido de o sistema reconhecer o início de um gesto, ou seja, de o sistema ter um referencial de início de gesto. Esta não deve ser muito grande, no entanto a abordagem para a mão esquerda seria semelhante. Podendo-se até usar as duas mãos.

3.3.1. Estáticos

Para que a máquina consiga reconhecer um gesto estático é dado, como *input* à ANN, um vector $\zeta^i = (x_1, x_2, ..., x_{21})^i$. Os dados $(x_1, x_2, ..., x_{21})$ dizem respeito a sete vectores

tridimensionais, gerados pelo dispositivo *Leap motion*. A Tabela 3.2 descreve o que representam esses vinte e um dados presentes no vector.

Tabela 3.2. Descrição dos vinte e um dados requeridos para o reconhecimento de gestos estáticos.

Dados	Descrição
x_1, x_2, x_3	Componentes *x*, *y* e *z*, respectivamente, do vector normal da palma da mão
x_4, x_5, x_6	Componentes *x*, *y* e *z*, respectivamente, do vector direcção da palma da mão
x_7, x_8, x_9	Componentes *x*, *y* e *z*, respectivamente, do vector direcção do dedo pulgar
x_{10}, x_{11}, x_{12}	Componentes *x*, *y* e *z*, respectivamente, do vector direcção do dedo indicador
x_{13}, x_{14}, x_{15}	Componentes *x*, *y* e *z*, respectivamente, do vector direcção do dedo médio
x_{16}, x_{17}, x_{18}	Componentes *x*, *y* e *z*, respectivamente, do vector direcção do dedo anular ou anelar
x_{19}, x_{20}, x_{21}	Componentes *x*, *y* e *z*, respectivamente, do vector direcção do dedo mínimo ou mindinho

De modo a que a ANN reconheça os gestos é necessário efectuar um treino. Esse treino consiste em fornecer um conjunto de vectores ζ e os respectivos vectores ς. O vector $\varsigma^i = (y_1, y_2, ..., y_{12})$ representa o *output* que se pretende reconhecer, ou seja, o número do gesto cujo vector ζ^i diz respeito. De modo a treinar a ANN foram dados quinze vectores ζ e quinze vectores ς, referentes a cada um dos doze gestos previamente treinados.

3.3.2. Dinâmicos

Para se reconhecer um gesto dinâmico é dado, como *input* aos HMMs, um vector $\Psi^i = \begin{Bmatrix} (x_1, x_2, ..., x_5)^1 \\ \vdots \\ (x_1, x_2, ..., x_5)^{18} \end{Bmatrix}$. O vector Ψ é composto por dezoito conjuntos de dados. Estes,

$(x_1, x_2, ..., x_5)^i$, dizem respeito a um vector tridimensional, à propriedade agarrar e a uma componente de um vector, gerados pelo dispositivo *Leap motion*, como descritos na Tabela 3.3.

Tabela 3.3. Descrição dos cinco dados requeridos para o reconhecimento de gestos dinâmicos.

Dados	Descrição
x_1, x_2, x_3	Componentes *x*, *y* e *z*, respectivamente, do vector normal da palma da mão
x_4	Propriedade agarrar
x_5	Componentes *x* do vector direcção do dedo pulgar

Contudo, com o intuito de facilitar o reconhecimento da sequência, esses dados são manipulados de modo a que o vector Ψ tenha uma curta diversidade de dados. Assim estes podem tomar o valor de -1, 0 ou 1. O Algoritmo 1 demonstra como esta manipulação é feita. Isto facilita o reconhecimento e aumenta de certa forma a taxa de sucesso de reconhecimento deste grupo de gestos.

Algoritmo 1 Manipulação dos dados de *input*

Input: velocidade(x); velocidade(y); velocidade(z); agarrar; direcção(polegar).

Output: *inputs* a variar entre -1, 0, 1

```
1:  Begin
2:      maior = x
3:      For i = y, z
4:          If velocidade(|i|) > velocidade(|maior|) Then
5:              maior = i
6:          End If
7:      End For
8:      If velocidade(maior) < 0 Then
9:          velocidade(maior) = -1
10:     Else
11:         velocidade(maior) = 1
12:     End If
13:     If velocidade(j) ≠ velocidade(maior) Then
14:         For j = x, y, z
15:             velocidade(j) = 0
16:         End For
17:     End If
18:     If agarrar variar Then
19:         For k = x, y ,z
20:             velocidade(k) = 0
21:         End For
22:         If agarrar < 0,5 Then
23:             agarrar = -1
24:         Else If agarrar > 0,5 Then
25:             agarrar = 1
26:         End If
27:     Else
28:         agarrar = 0
29:     End If
30:     If direcção(polegar) variar Then
31:         For l = x, y ,z
32:             velocidade(l) = 0
33:         End For
34:         If direcção(polegar) < 0 Then
35:             direcção(polegar) = -1
36:         Else If direcção(polegar) > 0 Then
37:             direcção(polegar) = 1
38:         End If
39:     Else
40:         direcção(polegar) = 0
41:     End If
42: End
```

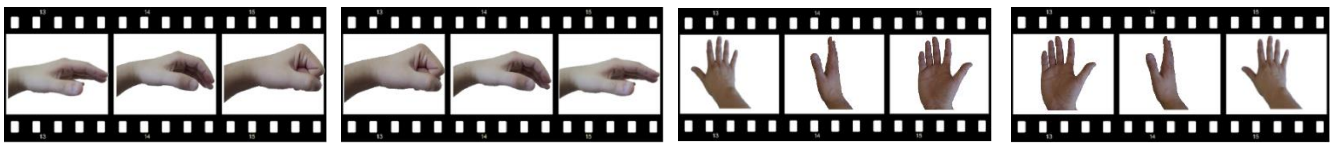
De modo a que os HMMs reconheçam os gestos é necessário efectuar a aprendizagem. Esta consiste em fornecer um conjunto de vectores Ψ^i e os respectivos vectores Γ^i. O vector $\Gamma^i = (y_1, y_2, ..., y_{10})$ representa o *output* que se pretende reconhecer. De modo a ensinar os HMMs foram dados quinze vectores Ψ e quinze vectores Γ, referentes a cada um dos dez gestos previamente guardados.

3.4. Detecção de movimento

A detecção de movimento é feita através da verificação da velocidade de um de dois dedos, polegar e médio, ilustrados na Figura 3.8. Para se verificar movimento, essa velocidade tem de ser superior a 40 mm/s .

Figura 3.8. Nomenclatura dos dedos da mão.

Foram escolhidos estes dois dedos pois são os dedos têm maior influência nos gestos considerados. No gesto de agarrar, Figura 3.9 (a), ou largar, Figura 3.9 (b), o dedo médio apresenta maior variação de velocidade, no gesto rodar sobre o pulso, Figura 3.9 (c) e (d), a velocidade do dedo polegar é a mais relevante, nos restantes gestos é indiferente a velocidade do dedo a considerar, pois estes movem-se com velocidades semelhantes.



(a) (b) (c) (d)

Figura 3.9. Gesto dinâmico (a) agarrar; (b) largar; rodar sobre o pulso (c) sentido horário; (d) sentido anti-horário.

4. TESTES E RESULTADOS

Seleccionaram-se doze gestos estáticos, mostrados na Tabela 4.1, e dez dinâmicos, mostrados na Tabela 4.2. Os gestos foram pensados de forma a tornar mais intuitiva a interacção com o robô, sendo portanto de fácil memorização. GE e GD são apenas referências que visam simplificar a distinção de gestos estáticos (GE) e dinâmicos (GD).

Tabela 4.1. Os doze gestos estáticos considerados.

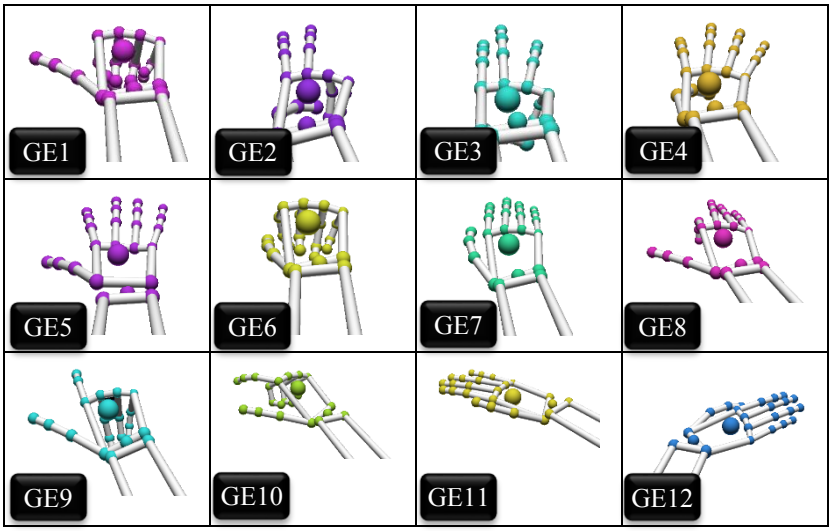
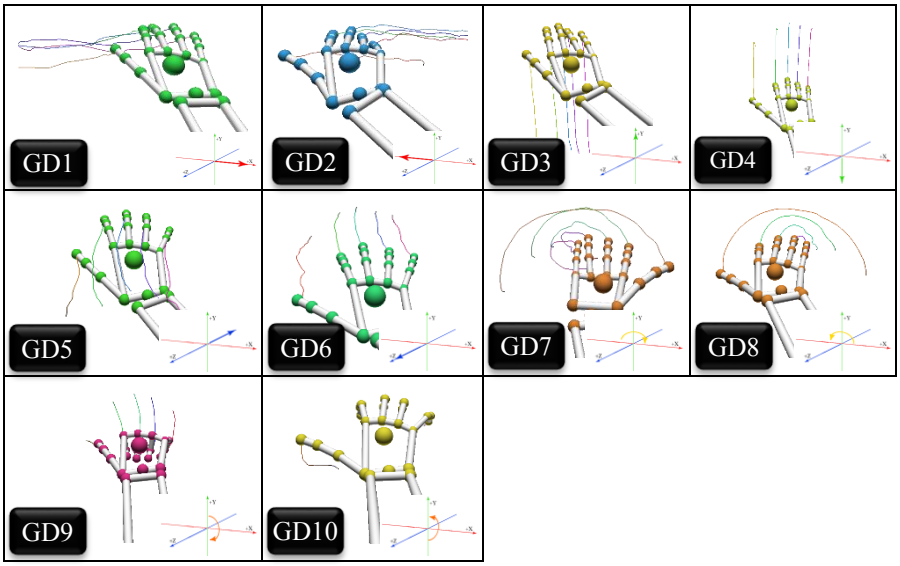


Tabela 4.2. Os dez gestos dinâmicos considerados.



Foram efectuados dois testes para verificar a *performance* deste sistema. O primeiro teste consiste na repetição de cada gesto por parte do utilizador que gerou os dados utilizados para o treino da ANN e para a aprendizagem dos HMMs. O segundo teste é em tudo semelhante ao primeiro, apenas diferindo o utilizador. Os testes são apresentados sob a forma de matriz confusão. Nesta matriz são apresentados o número de vezes que um gesto foi, correctamente ou não, reconhecido pelo sistema. As linhas dizem respeito ao gesto efectuado pelo utilizador e as colunas àquele que o sistema reconheceu.

Para uma melhor HRI o utilizador deve primeiro familiarizar-se com o dispositivo, de modo a que o próprio aprenda a utilizar este sistema de reconhecimento de gestos. Este processo demora entre cinco a dez minutos.

Para o desenvolvimento deste sistema de reconhecimento de gestos, execução do mesmo e realização de testes, recorreu-se a um computador com as especificações técnicas apresentadas na Tabela 4.3.

Tabela 4.3. Especificações técnicas do computador utilizado.

Processador	Intel® Core™ i7-4700HQ CPU @ 2.40 GHz (8CPUs)
Memória	8 GB
Sistema Operativo	Microsoft Windows 8.1 64 bits

4.1. Estáticos

Para o reconhecimento de gestos estáticos foram guardados quinze vectores ζ e respectivos vectores ς de forma a que seja efectuado o treino da ANN para os gestos considerados, Tabela 4.1. Alguns parâmetros tiveram de ser tidos em consideração, parâmetros esse que se encontram na Tabela 4.4.

Tabela 4.4. Parâmetros utilizados para a ANN.

Parâmetro	Valor
Número de neurónios da camada de entrada	21
Número de neurónios da camada escondida	21
Número de neurónios da camada de saída	12
Função de activação na camada escondida	Sigmóide assimétrica, com $\sigma = 1$
Função de activação na camada de saída	Sigmóide assimétrica, com $\sigma = 1$
Coeficiente de aprendizagem	0,25
Momentum	0,1
Número de ciclos de treino	10000
Taxa de actualização [ms]	25
Treino da rede neuronal [min]	1

Tanto para o Teste 1 como para o Teste 2, cada utilizador reproduziu o GE1, GE2, GE3, GE4, GE5, GE6, GE7, GE8, GE9, GE10, GE11 e GE12 cem vezes consecutivas.

4.1.1. Teste 1

O Teste 1, efectuado pelo mesmo utilizador que gerou os dados para o treino da ANN, encontra-se apresentado na Tabela 4.5.

Tabela 4.5. Matriz confusão de gestos estáticos – Teste 1.

	GE1	GE2	GE3	GE4	GE5	GE6	GE7	GE8	GE9	GE10	GE11	GE12
GE1	100	0	0	0	0	0	0	0	0	0	0	0
GE2	0	100	0	0	0	0	0	0	0	0	0	0
GE3	0	0	99	0	0	1	0	0	0	0	0	0
GE4	0	0	0	100	0	0	0	0	0	0	0	0
GE5	0	0	0	0	100	0	0	0	0	0	0	0
GE6	0	0	0	0	0	100	0	0	0	0	0	0
GE7	0	0	0	0	0	0	100	0	0	0	0	0
GE8	0	0	0	0	0	0	0	100	0	0	0	0
GE9	0	1	0	0	0	0	0	0	99	0	0	0
GE10	0	0	0	0	0	0	0	0	0	100	0	0
GE11	0	0	0	0	0	0	0	0	0	0	100	0
GE12	0	0	0	0	0	0	0	0	0	0	0	100

Pode concluir-se que das 1200 repetições 1198 foram correctamente conhecidas, fixando assim a taxa de reconhecimento global nos 99,83%. Esta taxa encontra-se muito próxima dos 100%, não o atingindo, devido ao facto de por vezes o utilizador executar movimentos involuntários podendo estes influenciar no reconhecimento. Outro dos motivos prende-se com a semelhança dos gestos considerados, obrigando o utilizador a replicar o gesto de forma mais exacta.

4.1.2. Teste 2

O Teste 2, efectuado por um utilizador diferente daquele que gerou os dados para o treino da ANN é apresentado na Tabela 4.6.

Tabela 4.6. Matriz confusão de gestos estáticos – Teste 2.

	GE1	GE2	GE3	GE4	GE5	GE6	GE7	GE8	GE9	GE10	GE11	GE12
GE1	100	0	0	0	0	0	0	0	0	0	0	0
GE2	0	100	0	0	0	0	0	0	0	0	0	0
GE3	0	0	99	1	0	0	0	0	0	0	0	0
GE4	0	0	0	100	0	0	0	0	0	0	0	0
GE5	0	0	0	0	99	1	0	0	0	0	0	0
GE6	0	0	0	0	0	100	0	0	0	0	0	0
GE7	0	0	0	0	0	0	100	0	0	0	0	0
GE8	0	0	0	0	0	1	0	99	0	0	0	0
GE9	0	0	0	0	0	0	0	0	100	0	0	0
GE10	0	0	0	0	0	0	0	0	0	100	0	0
GE11	0	0	0	0	0	0	0	0	0	0	100	0
GE12	0	0	0	0	0	0	0	0	0	0	0	100

Conclui-se assim, que das 1200 repetições 1197 foram correctamente conhecidas, fixando assim a taxa de reconhecimento global nos 99,75%. Esta taxa é inferior à taxa do Teste 1. Esta constactação já era espectável pois cada utilizador tem formas diferentes de executar um mesmo gesto. A este facto também se juntam os movimentos involuntários.

4.2. Dinâmicos

Para o reconhecimento de gestos dinâmicos foram guardados quinze vectores Ψ e respectivos vectores Γ de forma a que seja efectuado a aprendizagem dos HMMs para os gestos considerados, Tabela 4.2. Os parâmetros que foram tidos em consideração, encontram-se na Tabela 4.7.

Tabela 4.7. Parâmetros utilizados para o HMM.

Parâmetro	Valor
Número de HMMs	10
Número de estados	20
Número de símbolos	3
Critério de paragem - Tolerância	0,01
Tempo de aprendizagem [min]	<1

Tanto para o Teste 1 como para o Teste 2, cada utilizador reproduziu o GD1, GD2, GD3, GD4, GD5, GD6, GD7, GD8, GD9 e GD10 cem vezes consecutivas.

4.2.1. Teste 1

O Teste 1, efectuado pelo mesmo utilizador que gerou os dados para a aprendizagem dos HMMs, encontra-se apresentado na Tabela 4.8.

Tabela 4.8. Matriz confusão de gestos dinâmicos – Teste 1.

	GD1	GD2	GD3	GD4	GD5	GD6	GD7	GD8	GD9	GD10
GD1	83	0	0	17	0	0	0	0	0	0
GD2	0	93	0	5	2	0	0	0	0	0
GD3	0	0	95	5	0	0	0	0	0	0
GD4	0	0	0	100	0	0	0	0	0	0
GD5	0	0	0	2	98	0	0	0	0	0
GD6	0	0	0	9	2	89	0	0	0	0
GD7	0	0	1	1	0	0	98	0	0	0
GD8	0	0	0	4	0	0	0	96	0	0
GD9	0	0	0	5	3	1	0	0	90	1
GD10	0	0	0	0	0	0	0	0	0	100

A taxa de reconhecimento global é de 94,20%, onde 942 vezes o sistema reconheceu correctamente as 1000 repetições efectuadas pelo utilizador. Em sistemas de reconhecimento de gestos, os gestos dinâmicos têm taxas de reconhecimento notoriamente mais baixas que as de gestos estáticos. Como para o reconhecimento de gestos dinâmicos é necessário um conjunto de 18 dados sequenciais, que posteriormente são manipulados, como descrito em 3.3.2. Dinâmicos, qualquer movimento, pretendido ou involuntário, que o utilizador execute influenciará o reconhecimento. Assim os gestos dinâmicos exigem uma exactidão maior aquando a sua replicação.

4.2.2. Teste 2

O Teste 2, efectuado por um utilizador diferente daquele que gerou os dados para a aprendizagem dos HMMs é apresentado na Tabela 4.9.

Tabela 4.9. Matriz confusão de gestos dinâmicos – Teste 2.

	GD1	GD2	GD3	GD4	GD5	GD6	GD7	GD8	GD9	GD10
GD1	80	0	0	15	5	0	0	0	0	0
GD2	0	90	0	7	3	0	0	0	0	0
GD3	0	0	98	2	0	0	0	0	0	0
GD4	0	0	0	100	0	0	0	0	0	0
GD5	0	0	0	6	94	0	0	0	0	0
GD6	0	0	0	4	2	94	0	0	0	0
GD7	0	0	1	2	1	0	96	0	0	0
GD8	0	0	0	5	0	0	0	95	0	0
GD9	0	0	0	4	3	1	0	0	92	0
GD10	0	0	0	2	1	0	0	0	0	97

Assim se conclui que a taxa de reconhecimento global é de 93,60%, visto que o sistema reconheceu correctamente 936 das 1000 repetições efectuadas pelo utilizador. Esta taxa é inferior à taxa do Teste 1, como era esperado. Cada gesto é efectuado de forma diferente por cada utilizador. No caso de gestos dinâmicos nota-se de forma mais evidente pois é retirado um conjunto de 18 dados. A este facto também se juntam os movimentos involuntários.

4.2.3. Comentário

Em ambos os testes o reconhecimento inequívoco dos gestos GD3, GD4, GD5, GD6 deve-se, essencialmente, ao posicionamento da mão para início de execução do gesto. Por outras palavras, quando o utilizador pretende iniciar um gesto, posiciona a mão dentro da margem esférica. Se o tempo de interacção, até à colocação à posição de início de gesto, for suficiente para que o sistema seja capaz de gerar um vector Ψ, o sistema efectua o reconhecimento do mesmo, dando origem a reconhecimentos inequívocos.

4.3. Interacção com o robô

Como forma de demonstrar que este sistema pode ser considerado um sistema de HRI, foi testado, no laboratório de robótica do DEM/FCTUC, a utilização deste sistema no controlo de um robô industrial com uma garra acoplada. Os comandos e gestos utilizados estão presentes na Tabela 4.10.

Tabela 4.10. Atribuição dos gestos a movimentos do robô.

Gesto	Comandos robô	Gesto	Comandos robô
GE1	Iniciar comunicação com o robô	**GD1**	Mover direita
GE2	Ligar motores	**GD2**	Mover esquerda
GE3	Desligar motores	**GD3**	Mover para cima
GE4	Parar de executar movimento	**GD4**	Mover para baixo
GE5	Pausa – *Hold*	**GD5**	Mover para a frente
GE6	–	**GD6**	Mover para trás
GE7	–	**GD7**	Rodar garra sentido horário
GE8	–	**GD8**	Rodar garra sentido anti-horário
GE9	*Home*	**GD9**	Fechar garra
GE10	–	**GD10**	Abrir garra
GE11	–		
GE12	–		

5. CONCLUSÕES E TRABALHO FUTURO

Ao longo desta dissertação foram abordados todos os passos que foram tomados para se poder chegar ao objectivo final, interagir com um robô através de gestos.

Interagir com um robô industrial através de gestos simplifica, de forma mais intuitiva, a comunicação e dispensa de um utilizador com elevado conhecimento em programação de robôs. Isto permitirá a amplificação do uso de robôs, pois deste modo abrange um maior número de utilizadores. Em contrapartida, limita as operações que o utilizador pode efectuar pois a cada gesto corresponde uma operação.

A utilização de ANN para o reconhecimento de gestos estáticos mostrou ser bastante eficaz obtendo-se uma taxa de reconhecimento, num grupo de 12 gestos, de 99,83%, efectuado pelo mesmo utilizador que gerou os dados de treino, e 99,75%, para um utilizador diferente. Não sendo de 100% pois o utilizador não reproduz o gesto de forma exacta, levando a que a ANN reconheça de forma incorrecta o gesto pretendido.

Recorrendo ao HMM como forma de reconhecimento de gestos dinâmicos, também mostrou ser uma mais-valia pois obtiveram-se taxas de reconhecimento, para um grupo de 10 gestos, de 94,20%, efectuado pelo mesmo utilizador que gerou os dados de treino, e de 93,60%, para um utilizador diferente. Esta, tal como no reconhecimento de gestos estáticos, deve-se ao facto do utilizador não efectuar de maneira exacta o gesto. Qualquer movimento involuntário efectuado pela mão interfere no reconhecimento, como o tremer. Este grupo de gestos suscitou mais problemas, tendo sido ultrapassados recorrendo ao encurtamento na diversidade de dados para apenas três $(-1,0,1)$, como mencionado no subcapítulo 3.3.2. Dinâmicos.

Como trabalhos futuros sugerem-se a optimização do início da execução de um gesto dinâmico, de modo a eliminar o reconhecimento inequívoco de gestos. Também a parte da *interface* do programa, de modo a que um utilizador possa adicionar ou eliminar gestos, tanto estáticos como dinâmicos. E testar o sistema para mais gestos, assim como a optimização da ANN e dos HMMs.

REFERÊNCIAS BIBLIOGRÁFICAS

Nolker, C. & Ritter, H., 2002. Visual recognition of continuous hand postures. *Neural Networks, IEEE Transactions on,* 13(4), pp. 983 - 994.

Bianchi, M., Salaris, P. & Bicchi, A., 2012. Synergy-based optimal design of hand pose sensing. *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on,* pp. 3929 - 3935.

CyberGlove Systems LLC, 2014. *Photos / Video | cyberglovesystems.com.* [Online] Available at: http://www.cyberglovesystems.com/products/cyberglove-II/photos-video

Haylin, S., 1999. *Neural Networks - A Comprehensive Foundation.* 2ª edição ed. Hamilton, Ontario, Canada: Prentice Hall International, Inc. .

Hedlerfog, R., 2014. *Unveiling teardown of Leap motion device.* [Online] Available at: http://touchlessgeneration.com/discover-touchless/teardown-of-leap-motion-device/#.UxizLz-PksO

Leap Motion, Inc., 2014. *API Overview - Leap Motion C# and Unity SDK v2.0 Beta documentation.* [Online] Available at: https://developer.leapmotion.com/documentation/skeletal/csharp/devguide/Leap_Overview.html

Leap Motion, Inc., 2014. *Finger - Leap Motion C# and Unity SDK v2.1.0 Beta documentation.* [Online] Available at: https://developer.leapmotion.com/documentation/skeletal/csharp/api/Leap.Finger.html#csharpclass_leap_1_1_pointable_1a62885a2911b2c21274564c3d9e407c2c

Leap Motion, Inc., 2014. *Hand - Leap Motion C# and Unity SDK v2.0 Beta documentation.* [Online] Available at: https://developer.leapmotion.com/documentation/skeletal/csharp/api/Leap.Hand.html#csharpclass_leap_1_1_hand_1a4ec880bd78819e39e4c90ff013ed3829

Leap Motion, Inc, 2014. *Leap Motion | 3D Motion and Gesture Control for PC & Mac.* [Online] Available at: https://www.leapmotion.com/product/vr

Leap Motion, Inc, 2014. *Leap Motion Developers.* [Online] Available at: https://developer.leapmotion.com/vr

Leap Motion, Inc, 2014. *Technical specifications.* [Online] Available at: https://forums.leapmotion.com/showthread.php?1005-Technical-specifications&p=5795&viewfull=1#post5795

Leap Motion, Inc, 2014. *The Leap Motion Controller.* [Online] Available at: https://store.leapmotion.com/(S(ugyqqoctsoumdvlew3fbvxu5))/Pages/LeapSolution.aspx

Leap Motion,Inc, 2014. *Tracking Data - Leap Motion C# and Unity SDK v2.0 Beta documentation.* [Online]

Available at: https://developer.leapmotion.com/documentation/skeletal/csharp/devguide/Leap_Tracking.html

Lee, H. K. & Kim, J. H., 1999. An hmm-based threshold model approach for gesture recognition. *IEEE PAMI,* 21(10), p. 961– 973.

Limer, E., 2014. *Leap Motion Teardown: Magic Made Simply.* [Online] Available at: http://gizmodo.com/leap-motion-teardown-magic-made-simply-608132248

McCulloch, W. S. & Pitts, W., 1943. *A Logical Calculus of the Ideas Immanent in Nervous Activity.* s.l.:Bulletin of Mathematical Biophysics.

Mehrabian, A., 1971. *Silent messages.* 1ª edição ed. Belmont, California: Cengage Learning, Inc..

Microsoft Corporation, 2014. *Kinect.* [Online] Available at: http://www.xbox.com/pt-BR/Xbox360/Accessories/Kinect/Home

Mitra, S. & Acharya, T., 2007. Gesture Recognition: A Survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on,* 37(3), pp. 311 - 324.

Moreto, M. & Rolim, J. G., 2010. *Sba: Controle & Automação Sociedade Brasileira de Automatica - Automated analysis of digital fault recorder data in power systems.* [Online] Available at: http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-17592010000400003

Neto, P., Pereira, D., Pires, J. N. & Moreira, A. P., 2013. Real-Time and Continuous Hand Gesture Spotting: an Approach Based on Artificial Neural Networks. *2013 IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany, MAy 6-10, 2013,* pp. 178-183.

Neto, P., Pires, J. N. & Moreira, A. P., 2009. Accelerometer-based control of an industrial robotic arm. *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on,* pp. 1192 - 1197.

Neto, P., Pires, J. N. & Moreira, A. P., 2012. High-level programming and control for industrial robotics: using a hand-held accelerometer-based input device for gesture and posture recognition. *Industrial Robot: An International Journal,* 37(2), pp. 137 - 147.

Nintendo Co., Lda, 2014. *Utilização do Comando Remoto Wii | Wii | Assistência | Nintendo.* [Online] Available at: http://www.nintendo.pt/Assistencia/Wii/Utiliza-ccedil-atilde-o/Comando-Remoto-Wii/Utilizacao-do-Comando-Remoto-Wii-/Utilizacao-do-Comando-Remoto-Wii--243981.html

Pieskä, S., Kaarela, J. & Saukko, O., 2012. Towards Easier Human-Robot Interaction to Help Inexperienced Operators in SMEs. *CogInfoCom 2012 • 3rd IEEE International Conference on Cognitive Infocommunications • December 2-5, 2012, Kosice, Slovakia,* pp. 333-338.

Rabiner, L. R., 1989. A Tuturoal on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEE, vol. 77, No. 2 February 1989,* pp. 257-286.

Wolf, M. T. et al., 2013. Gesture-Based Robot Control with Variable Autonomy from the JPL BioSleeve. *2013 IEEE International Conference on Robotics and Automation (ICRA),* pp. 1160-1165.

Yamashita, Y. & Tani, J., 2004. Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment. *PLoS Comput. Biol.,* 4(11), p. 1–17.

Ziesecke, D., 2014. *Review Leap Motion Motion Control Technology.* [Online] Available at: http://www.notebookcheck.net/Review-Leap-Motion-Motion-Control-Technology.98821.0.html