INSTITUTO POLITÉCNICO DE COIMBRA

INSTITUTO SUPERIOR DE ENGENHARIA DE COIMBRA

Interfaces Gestuais Baseados no Controlador Leap Motion

Trabalho de Projeto para a obtenção do grau de Mestre em

Informática e Sistemas

Autor

João Pedro Pereira Bizarro

Orientação

Prof. Nuno Martins

Prof. Simão Paredes

Junho 2019

Agradecimentos

Quero agradecer ao DEIS por disponibilizar os recursos necessários para a realização deste trabalho. Gostava também de agradecer aos professores Nuno Martins e Simão Paredes por coordenar o trabalho e prestar auxílio nas ocasiões em que problemas surgiram na realização do trabalho.

Abstract

In the present, most of the human-machine interactions are based on the use of peripherals such as keyboard and computer mouse. However, the use of such peripherals can create certain limitations in the way people interact with machines, for this reason, there is a need to create natural interfaces. One of the possible approaches that has been proposed involves performing gestures that are recognized by a sensor and interpreted by the computer. The use of hands on a human-machine interface is justified by the fact that the hands are an important element in nonverbal communications. Due to this, in this project several possible gesture interfaces were analyzed, using the Leap Motion sensor.

The project was based on the development of methods that allowed the recognition of gestures and their association to an action that the computer should perform. Through the analysis of existing studies in the area and the various methods used to allow a program to classify a data set, a gesture classification system was developed. The classification system has tested to verify its accuracy and precision.

Using the knowledge obtained throughout the project, and as proof of concept, an application was developed to demonstrate the usefulness of the classification system in a real situation. This application can recognize a gesture and associate it with a keyboard key, allowing a user to write the message resulting from the gestures he makes.

This project main conclusion was that the gesture classification system trained using SVM can make a good separation of the various gestures and with this classify correctly the gestures. Most of problems that arise during the recognition of a gesture are a consequence of the Leap Motion not being able to track correctly the gesture being made.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

1 INTRODUÇÃO

Este trabalho surge no âmbito de desenvolvimento de interfaces gestuais usando mãos e braços, inserido no contexto do projeto de final de curso do Mestrado em Informática e Sistemas.

Recentemente tem havido grande interesse no desenvolvimento de interfaces, para interação pessoa-máquina, que não necessitam a utilização de periféricos como ratos e teclados. Isto deve-se ao facto de a utilização destes periféricos não ser conveniente para pessoas com certas necessidades especiais. Outro problema é causado pela normalização destes periféricos, o que pode restringir as possíveis maneiras como as interações pessoa-máquina são feitas. Devido a estas razões, é considerado importante analisar outras maneiras de criar interfaces.

Estas foram as principais razões que nos levaram a escolher esta área para a realização deste trabalho. Houve também grande interesse em trabalhar com sensores de movimento e analisar o seu funcionamento. Por último houve um desejo de aprender em maior profundidade como os sistemas de reconhecimento e algoritmos de aprendizagem funcionam.

Foi decidido focar o desenvolvimento da interface gestual das mãos, uma vez que o movimento das mãos em comunicações não-verbais, constitui um dos fatores mais importantes, sendo também importante como suporte em comunicações verbais. Um exemplo da sua importância em comunicações não-verbais é demonstrado pelo facto de vários gestos feitos pelas mãos terem significados associados que não necessitam ser clarificados através de ajuda vocal. A utilização de gestos das mãos para ajudar a interpretar certas ideias expressas verbalmente é um exemplo de um caso em que os gestos das mãos ajudam em comunicações verbais.

1.1 SISTEMAS DE RECONHECIMENTO GESTUAL

Um sistema de reconhecimento de gestos, pode ser distinguido em três etapas principais. Estas etapas são: o método de extração, a extração de parâmetros e a classificação dos gestos. Estes passos e a sua ordem estão representados na figura 1 (Khan & Ibraheem, 2012; Sarkar, Sanyal & Majumder, 2013).

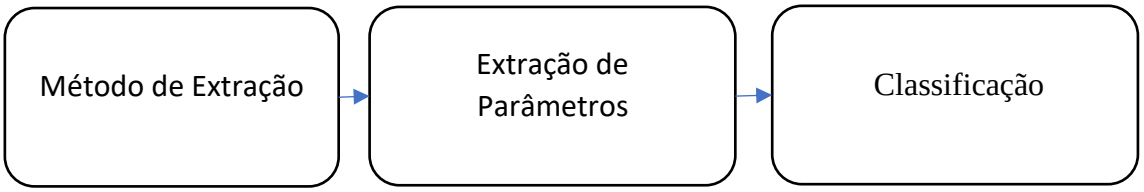


Figura 1. Etapas de um sistema de reconhecimento de gestos.

O método de extração ou modelação do gesto é o método que o sensor utiliza para criar o modelo virtual das mãos reconhecidas pelo sensor. O método começará por segmentar as mãos, esta segmentação servirá para detetar a posição das mãos na

imagem recolhida pelo sensor. O segundo passo será remover o ruído da imagem para aumentar a sua qualidade e é necessário para assegurar uma boa precisão e exatidão do modelo criado. Este ruído é obtido durante a recolha da imagem ou a sua transmissão. O terceiro passo da modelação do modelo é a deteção de contornos, neste passo serão identificadas várias descontinuidades das propriedades da imagem que serão utilizadas para ajustar o modelo de maneira que ele seja mais preciso. O último passo deste processo é a normalização do modelo em que são removidas zonas do modelo que são consideradas de pouca utilidade ou inúteis (Khan & Ibraheem, 2012; Sarkar, Sanyal & Majumder, 2013).

A extração de parâmetros baseia-se em analisar o modelo das mãos criado durante o método de extração e através desse modelo criar certos parâmetros das mãos. Estes parâmetros são utilizados para permitir reconhecer as diferenças entre os vários gestos. Tanto o método de extração, como a extração de parâmetros são passos em que a maioria dos criadores de sistemas de reconhecimento não necessitam fazer nenhum desenvolvimento devido ao facto do software do sensor fazer estes passos automaticamente. Mas existem ocasiões em que programadores podem querer alterar certas partes do software do sensor ou criar métodos independentes do sensor, que podem alterar os dois primeiros passos (Khan & Ibraheem, 2012; Sarkar, Sanyal & Majumder, 2013).

A classificação de gestos é o processo de através da escolha de parâmetros da mão que permitam reconhecer o gesto e de um algoritmo de aprendizagem, permita identificar claramente quais são as diferenças entre os vários gestos. Através da deteção das várias diferenças entre os gestos dados para treinar o sistema, criar uma base de dados que quando dado um novo exemplo permita obter uma previsão que identifique qual é o gesto feito (Khan & Ibraheem, 2012; Sarkar, Sanyal & Majumder, 2013).

Sistemas de reconhecimento de gestos podem ser utilizados em várias áreas como: reconhecimento de linguagens gestuais, controlo de robôs, controlo de editores gráficos, ambientes virtuais, reconhecimento de números, controlo de aparelhos eletrónicos como televisões e modelação 3D (Khan & Ibraheem, 2012).

1.2 OBJETIVOS

Assim, os principais objetivos deste projeto são a análise das várias tecnologias atuais de reconhecimento de gestos das mãos e braços, tendo como objetivo final a criação de um sistema de reconhecimento de gestos das mãos e braços. Este sistema deverá ter uma boa exatidão, mas ao mesmo tempo, não deverá ser demasiado inflexível, sendo capaz de manter uma boa exatidão para cada gesto, mesmo quando trabalhando com gestos muito semelhantes.

Este sistema de reconhecimento de gestos permitirá associar um gesto a uma ação que o computador deve fazer. Esta associação entre gesto e ação permitirá que o sistema tenha várias aplicações práticas. Um exemplo disto é conseguir que um gesto esteja associado a uma tecla especifica do teclado, permitindo com isto a edição de textos usando apenas gestos das mãos. Outro exemplo seria que quando o sistema reconhece

um gesto, o computador iria fazer certas ações, como por exemplo minimizar ou fechar a janela atualmente aberta.

1.2.1 Principais Contributos

Foram identificados três grandes contributos a serem atingidos no âmbito deste trabalho:

1. Avaliação da performance do sensor utilizado no reconhecimento de diversos tipos de gestos.
2. Criar uma aplicação que permita utilizar o sistema de reconhecimento de gestos criado, de maneira a facilitar interação entre o homem e a máquina.
3. Identificar possíveis problemas que podem surgir durante o processo de reconhecimento de gestos e tentar resolver estes problemas.

1.3 TAREFAS PRINCIPAIS

O trabalho desenvolvido dividiu-se em quatro grandes tarefas:

T1 – *Análise e levantamento de requisitos*: Pesquisa do estado da arte sobre o reconhecimento gestual e sobre diferentes formas de interfaces IPM(Interação Pessoa-Máquina);

T2 – *Desenho*: Definição de requisitos de software e hardware para a criação das aplicações de demonstração de conceito;

T3 – *Desenvolvimento e testes*: Desenvolvimento das aplicações definidas na tarefa T2;

T4 – *Relatório de* estágio: Elaboração do relatório de estágio, com os resultados e objetivos atingidos.

A cada uma destas tarefas foi assignada uma meta. Era esperado que quando a data da meta chegasse a tarefa assignada a ela estivesse concluída. As metas foram inicialmente definidas como:

INI – Início dos trabalhos
M1 – Tarefa T1 terminada - (INI + 6 Semanas)
M2 – Tarefa T2 terminada - (INI + 10 Semanas)
M3 – Tarefa T3 terminada - (INI + 14 Semanas)
M4 – Tarefa T4 terminada - (INI + 22 Semanas)

Na figura 2 é apresentado o calendário inicial proposto para o projeto.

	Meses											
Tarefas		N		N+1		N+2		N+3		N+4		N+5
T1												
T2												
T3												
T4												
Metas	INI			M1		M2		M3				M4

Figura 2. Calendarização do Projeto.

1.4 Estrutura do Documento

Este documento encontra-se dividido em cinco capítulos. O capítulo 2, reconhecimento gestual, faz uma introdução teórica das várias formas de classificar gestos da mão. Este capítulo faz também uma revisão dos vários sensores de movimento de onde se destaca o Leap Motion e justifica porque é que foi esse o sensor escolhido para este trabalho. É, também, feita uma análise dos vários classificadores existentes na área de reconhecimento de gestos. Por fim é apresentada uma revisão de aplicações similares à que se vai criar.

No capítulo 3, trabalho desenvolvido, são apresentados os vários requisitos, arquitetura e funcionalidades da aplicação prova de conceito implementada. Este capítulo contém também uma descrição das várias tecnologias utilizadas para criar a aplicação. Por último será feita uma descrição das várias tecnologias utilizadas para criar o sistema de reconhecimento de gestos e a aplicação. Neste capítulo também é explicado como o sistema de reconhecimento de gestos será implementado neste projeto, a descrição e criação de uma aplicação usada para testar em detalhe a exatidão do método utilizado para criar o sistema e os resultados obtidos do teste dessa aplicação.

No capítulo 4, resultados, são descritos os resultados do projeto incluindo o sistema de reconhecimento de gestos, a aplicação criada que utiliza este sistema e testes da robustez do reconhecimento.

No capítulo 5, considerações finais, são tecidas as considerações finais do projeto, nomeadamente as principais conclusões obtidas durante a sua realização. São ainda discutidos os possíveis trabalhos futuros baseados neste projeto, como também possíveis desenvolvimentos futuros de forma a melhorar o trabalho já desenvolvido.

2 RECONHECIMENTO GESTUAL

Neste capítulo serão discutidas as várias áreas necessárias à implementação de um sistema de reconhecimento de gestos, nomeadamente:

- Gestos: Os gestos das mãos são utilizados em comunicação não verbal e como suporte em comunicação verbal. Devido a este facto, a maioria das pessoas têm alguns gestos associados a certas ações e conceitos. Sendo, por esta razão que, uma clara compressão de como os gestos das mãos são interpretados, necessária para permitir a criação de um sistema de reconhecimento de gestos que seja intuitivo e útil para os seus utilizadores.
- Sensores: A utilização de um sensor é necessária para permitir que o sistema de reconhecimento de gestos seja capaz de reconhecer os gestos feitos pelo utilizador no mundo real e transformar esses gestos em parâmetros com que ele consegue trabalhar. Devido a isto, a utilização de um sensor é necessária em qualquer sistema de reconhecimento de gestos.
- Classificadores: Um método de classificação é necessário para o sistema de reconhecimento de gestos. Um classificador através dos dados obtidos do gesto pelos sensores, reconhece o gesto feito baseado em exemplos prévios.

2.1 GESTOS

Gestos são sinais de comunicação não verbal, feitos através da utilização de partes específicas do corpo (braços, mãos, etc), com o objetivo de expressar certas ideias. O gesto pode também ser entendido como uma ação.

Todos os gestos feitos pelas mãos e braços podem ser divididos em dois grandes grupos: gestos estáticos e gestos dinâmicos. Os gestos estáticos são gestos em que são apenas analisados os seus parâmetros num momento fixo no tempo, como se pode ver na figura 3.

Figura 3. Gesto Estático.

Gestos dinâmicos são gestos em que a posição, rotação ou postura alteram-se ao longo do tempo, como demonstrado nas figuras 4,5 e 6.

Figura 4. Gesto dinâmico com alteração na posição da mão.

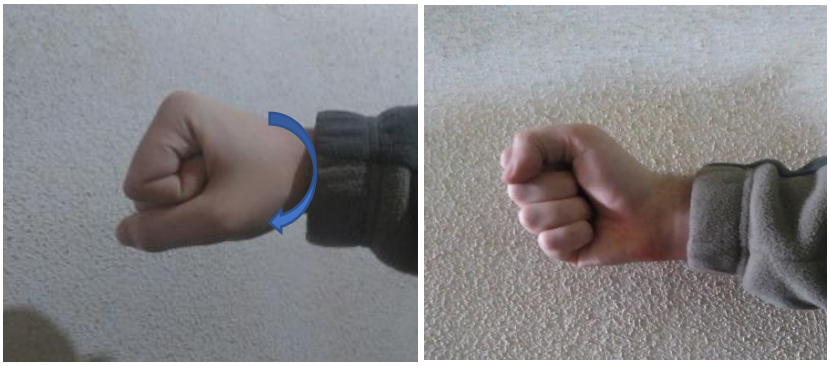


Figura 5. Gesto dinâmico com alterações na rotação da mão.

Figura 6. Gesto dinâmico com alteração da postura da mão.

Existem várias formas de classificar os gestos feitos pelas mãos e braços. Uma das principais formas é feita através de uma análise da taxonomia dos vários gestos feitos. Os estudos feitos por Aigner et al. (2012) e Karam & Schraefel (2005) apresentam os modelos taxonómicos mais apropriado para o reconhecimento de gestos feitos pelas mãos.

No modelo taxonómico proposto no estudo feito por Karam & Schraefel (2005), os gestos são divididos em cinco grupos: deíctico, gesticulação, manipulativo, semafórico e linguístico. Foi considerado que este modelo, apesar de ser útil, apresenta algumas fragilidades. A maior fragilidade é o âmbito dos gestos semafóricos ser muito grande e

abstrato, dando muita importância à influência da fala da pessoa que faz o gesto e como tal menos importância a outros contextos em que o gesto está inserido.

O modelo do estudo feito por Aigner et al. (2012) é uma versão alterada do modelo de Karam & Schraefel (2005). Segundo Aigner et al. (2012) a taxonomia dos gestos das mãos pode ser dividida em cinco tipos de gestos: indicativo, semafórico, pantomímico, icónico e manipulativo. Estas alterações eliminam os problemas do modelo de Karam & Schraefel (2005), mas ao mesmo tempo criam outros problemas. O principal problema surge do método utilizado para fazer redução do âmbito dos gestos semafóricos. Os gestos icónicos foram considerados demasiado semelhantes aos novos gestos semafóricos, resultando numa maior dificuldade a identificar se um gesto é semafórico ou icónico.

Com estes problemas em mente foi criado um novo modelo com os seguintes grupos:

- Indicativo: Este tipo de gesto pode indicar direções ou a posição de objetos. Gestos indicativos podem ser realizados com qualquer parte da mão. Estes gestos são sempre estáticos. Um exemplo deste tipo de gestos é uma pessoa apontar com a mão ou dedo para uma direção quando lhe é perguntado onde está uma pessoa, objeto ou lugar.
- Semafórico: São gestos ilustrativos, em que a postura e posição das mãos tentam exprimir informação acerca de um objeto ou entidade ou exprimir um certo significado. Os gestos deste tipo podem ser estáticos ou dinâmicos. Exemplos deste tipo de gestos é levantar o polegar para exprimir aceitação ou abanar uma mão para exprimir negação.
- Pantomímico: Gestos pantomímicos são usados para ilustrar certas tarefas que envolvem movimentos e posturas específicas das mãos. Gestos deste tipo são geralmente formados por vários gestos simples. Um exemplo deste tipo de gestos é trazer um copo imaginário para a boca para indicar que a pessoa que está a fazer o gesto quer ir beber alguma coisa. Estes gestos são sempre dinâmicos.
- Linguístico: Gestos usados em linguagens gestuais. Distinguidos dos gestos semafóricos pelos gestos terem o seu significado e informação especificados por um grupo externo ao contrário dos gestos semafóricos em que a informação ou significado a ser transmitido é criado pela pessoa que realiza o gesto. Um exemplo disto são os gestos da linguagem gestual portuguesa, em que a mão fechada e o polegar levantado representa a letra “B”.
- Manipulativo: Gestos manipulativos são gestos que são usados para manipular objetos, havendo uma clara interação entre o gesto feito pela mão e o movimento feito pelo objeto, com o gesto feito pelas mãos influenciando a posição, rotação e tamanho do objeto. Estes gestos são sempre dinâmicos. Um exemplo deste tipo de gestos é manipular um cordel de forma que este fique á volta da mão da pessoa que está a fazer o gesto.

Na figura 7 encontra-se representado de uma forma sistemática a classificação dos vários tipos de gestos seguindo este modelo.

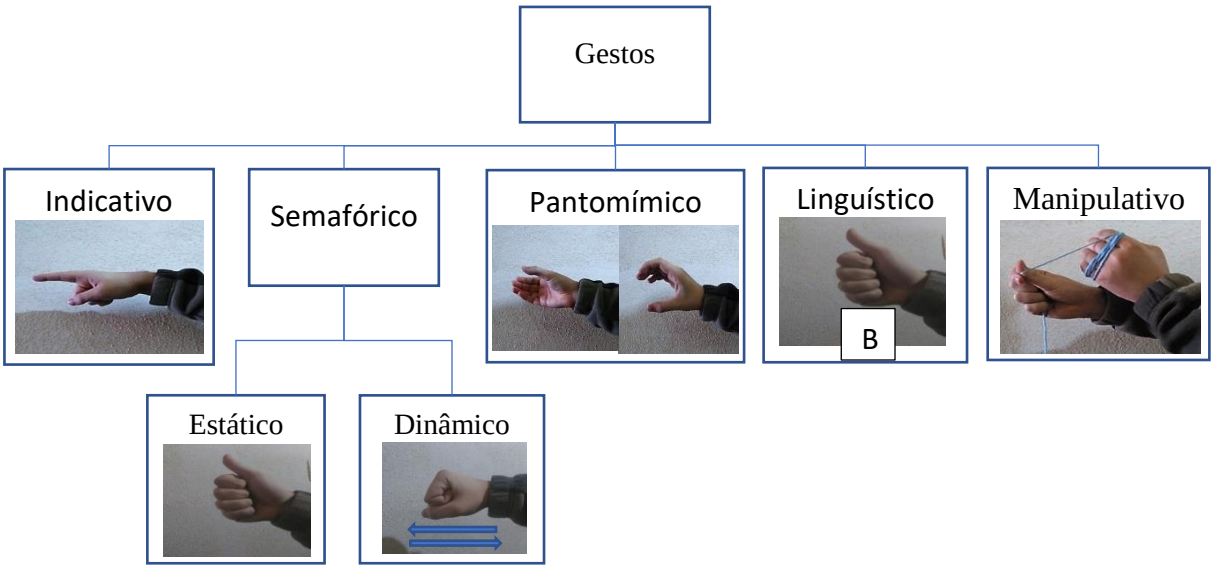


Figura 7. Classificação taxonómica de gestos.

No estudo de Karam & Schraefel (2005) são propostos os gestos de gesticulação. Os gestos deste grupo são gestos em que o seu significado e intenção são afetados pelo contexto da fala do utilizador. Foi considerado que este grupo de gestos não é similar aos restantes grupos propostos por Karam & Schraefel (2005), já que este grupo se concentra mais na relação entre o contexto em que o gesto é feito e como esse contexto vai afetar o significado do gesto, enquanto os outros grupos concentram-se simplesmente na intenção do gesto. Apesar disto, foi decidido que o contexto em que o gesto é feito é um elemento vital para todas as classificações de gestos baseados na taxonomia. Devido a isto criamos uma segunda classificação dos gestos, esta classificação é baseada no contexto em que os gestos estão a ser feitos e não na intenção que o gesto pretende exprimir. Esta nova classificação possui os seguintes grupos:

- Gesticulação: Gestos em que o seu significado e intenção são afetados pelo contexto criado pela fala do utilizador. Um exemplo deste grupo de gestos é uma pessoa apontar para um lugar enquanto ao mesmo tempo descreve esse lugar.
- Corporal: Gestos em que o seu significado e intenção são afetados pelo contexto criado pelas ações de outras partes do corpo do utilizador. Um exemplo é disto são gestos em que o seu significado é determinado pela expressão facial da pessoa.
- Externos: Gestos em que o seu significado e intenção são afetados pelo contexto criado por fatores externos ao utilizador. Estes fatores podem ser outros indivíduos, podem ser objetos que o utilizador interage ou podem ser condições ambientais. Um exemplo é apontar para uma direção quando lhe é perguntado em que direção deve ir para chegar a um certo lugar.

Usar uma análise taxonómica de gestos diretamente como base para a criação de uma biblioteca de reconhecimento de gestos é difícil, isto deve-se à necessidade de

existência de contexto externo para vários dos tipos de gestos, especialmente para os gestos pantomímicos e semafóricos.

Por este motivo a criação de aplicações dedicadas à classificação de gestos implica que seja possível classificar os gestos sem ter necessidade de obter o contexto em que estes estão inseridos ou em alternativa desenvolver métodos que permitam que o classificador tenha sempre acesso ao contexto em que o gesto foi feito. Esta última abordagem exige um aumento dos equipamentos utilizados devido à necessidade de se obter informação acerca do ambiente ao redor do utilizador e um aumento na complexidade do código do classificador. Por conseguinte este tipo de classificador é mais apropriado em contextos de interação pessoa-máquina onde se quer que a máquina consiga interpretar os gestos feitos por uma pessoa quando interage com outra pessoa. Um exemplo disto seria um sistema que fosse capaz de parcialmente inferir o estado emocional de uma pessoa baseado nos seus gestos.

Por outro lado, um método que ignora o contexto em que os gestos são feitos e apenas analisa o gesto é muito mais simples de implementar e o custo de obtenção dos equipamentos necessários é muito menor. Um exemplo deste método de classificação foi proposto por Nowicki et al. (2014).

A classificação proposta por Nowicki et al. (2014) divide gestos em dois grupos:

- Gestos de ação
- Gestos Parametrizados

Os gestos de ação são gestos em que um significado foi atribuído anteriormente fazendo que o programa execute uma ação quando o gesto é reconhecido, ignorando vários parâmetros do gesto ou tendo um grande intervalo de aceitação dos parâmetros a analisar. Gestos deste tipo podem ser estáticos ou dinâmicos.

Os gestos parametrizados são gestos em que para além de um gesto ter um significado associado, os valores dos parâmetros posição, rotação e postura, também, serão considerados durante a classificação do gesto. Apenas gestos dinâmicos podem ser utilizados como gestos parametrizados. Um exemplo de um gesto parametrizado é rodar a mão, sendo que se a rotação parar aos 45^{o} acontece um evento, mas se a rotação parar aos 90^{o} acontece outro evento.

Nowicki et al. (2014) também dividiram gestos dinâmicos em outros três subgrupos: globais, locais e mistos. Os gestos globais são gestos em que apenas a posição e rotação das mãos é alterada durante a realização do gesto. Os gestos locais são gestos em que apenas as posturas das mãos são alteradas. Nos gestos mistos a posição, rotação e postura são alteradas durante a realização do gesto.

Tendo os pontos discutidos nesta subsecção em conta foi decidido que este estudo irá focar-se apenas nos gestos feitos pelas mãos e braços.

O modelo utilizado neste trabalho para fazer a classificação dos gestos é uma versão simplificada do modelo de classificação proposto por Nowicki et al. (2014). Esta

simplificação é feita através da eliminação dos subgrupos referentes aos gestos globais, locais e mistos. Esta escolha foi feita após uma análise de como a divisão original afetava o modelo. A eliminação dos subgrupos foi feita por considerarmos que esta divisão é desnecessária, causando que o modelo fosse mais complicado do que era necessário e que a remoção dos subgrupos simplificava o modelo. Este modelo simplificado é apresentado na figura 8.

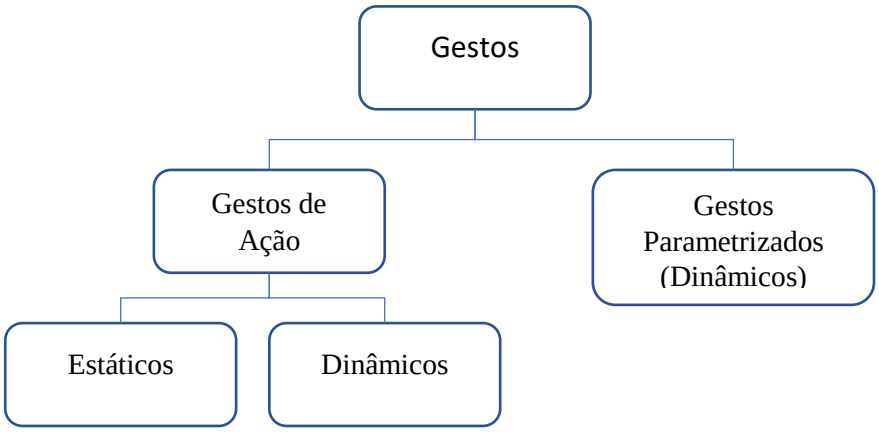


Figura 8. Versão simplificada da classificação de gestos segundo Nowicki et al. (2014).

Em conclusão, a versão simplificada da classificação de gestos segundo Nowicki et al. (2014), apresentada na figura 8, é o modelo de classificação de gestos adotado neste trabalho. Isto deveu-se ao facto de este modelo de classificação de gestos não necessitar que o sistema de reconhecimento de gestos criado, venha acompanhado de uma inteligência artificial avançada para analisar os vários contextos em que o gesto é feito.

2.2 Sensores de Movimento

2.2.1 Análise Comparativa

A decisão de neste trabalho restringir os gestos apenas aos gestos das mãos e braços, tem como consequência uma redução do número de sensores de movimento que é possível selecionar.

Para escolher o melhor sensor de movimento foram analisadas as seguintes alternativas:

1. Sony Move
2. Myo
3. Kinect
4. Leap Motion

2.2.1.1 Sony Move

O Sony Move, conforme se pode ver na figura 9, é um comando com uma luz LED que trabalhando em conjunto com uma câmara vai permitir o sistema detetar os movimentos feitos pela luz LED a que o sistema vai reagir através da execução de um ou mais eventos.

Figura 9. Sony Move[1].

O Sony Move foi escolhido como o representante dos vários sensores de movimento semelhantes a ele, como o Wii remote, o Comando Oculus Touch e Comando HTC Vive, ou seja, sensores em que o utilizador necessita segurar o sensor na sua mão e que utilizam uma câmara para fazer o rastreio do movimento do sensor. Este agrupamento de vários sensores justifica-se pelo facto de todos os sensores mencionados não terem quase nenhuma diferença na forma como funcionam e como interagem com o utilizador. O principal problema do Sony Move é que ele só consegue obter a posição e rotação da mão do utilizador, sendo impossível obter a postura das mãos do utilizador, uma propriedade que é essencial para a análise dos gestos das mãos.

2.2.1.2 Myo

O sensor de movimentos Myo, conforme se pode ver na figura 10, é o único sensor analisado que não é baseado em métodos de reconhecimento visual, não necessitando de nenhuma câmara para funcionar.

Figura 10. Myo[2].

Este sensor irá recolher os movimentos do músculo do antebraço do utilizador e inferir o gesto da mão e braços que o utilizador está a fazer através desse movimento dos músculos. Um dos maiores problemas do dispositivo Myo prende-se com a limitação de apenas reconhecer cinco gestos[3]: deslizar para a esquerda, deslizar para a direita, toque duplo, punho e mão aberta com todos os dedos esticados. Esta limitação no número de gestos disponíveis impõe um limite na complexidade das aplicações que podem utilizar

[1] https://www.amazon.in/Sony-Move-Motion-Controller-PS3/dp/B002I0J51U
[2] https://www.amazon.com/Thalmic-Labs-Gesture-Control-Presentations/dp/B00VHWBH02/ref=cm_cr_pr_sims_t
[3] https://support.getmyo.com/hc/en-us/articles/202647853-What-gestures-does-the-Myo-armband-recognize-

o Myo. O Myo também obteve pouco ou nenhum interesse dos investigadores na área, não tendo sido possível encontrar informação detalhada acerca da precisão do dispositivo.

2.2.1.3 Kinect

O sensor Kinect, conforme se pode ver na figura 11, é um sensor de movimento que através de uma webcam permite os seus utilizadores interagirem com o software sem estes terem necessidade de outros dispositivos complementares (como por exemplo o Sony Move e uma consola Sony).

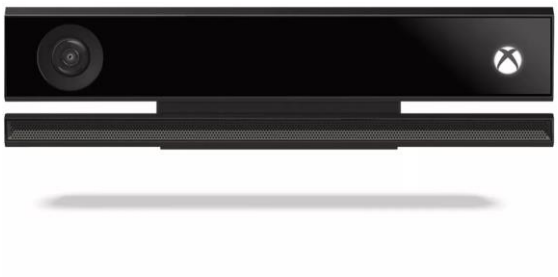


Figura 11. Microsoft Kinect[4].

O Kinect é também capaz de reconhecer comandos vocais. Em termos de performance, vários estudos (Khoshelham & Elberink, 2012; Wasenmüller & Stricker, 2016) concluíram que o Kinect tem uma boa exatidão e precisão, também apresentando bons resultados em estudos em que é usado para rastrear o movimento de pacientes em terapias de reabilitação (Oliver et al., 2016).

2.2.1.4 Leap Motion

Por fim, o último dispositivo estudado foi o Leap Motion, que se pode ver na figura 12.

Figura 12. Dispositivo Leap Motion [5].

Este dispositivo dá aos seus utilizadores a possibilidade de interagir com software através de gestos das suas mãos e braços sem contacto direto com o equipamento. O

[4] https://www.polygon.com/2014/5/13/5714120/no-kinect-xbox-one
[5] https://en.wikipedia.org/wiki/Leap_Motion

sensor utiliza câmaras e infravermelhos para gerar uma imagem das mãos e braços do utilizador. Um modelo das mãos, dedos e braços que o utilizador tiver na área rastreada pelas câmaras do dispositivo é criado. O sensor é ligado ao computador através de um cabo USB. Para desenvolver software com o sensor é necessário instalar um kit de desenvolvimento de software (SDK).

Na análise dos vários sensores verificou-se que o Leap Motion apresenta várias vantagens quando comparado com os sensores anteriores. Ao contrário do Sony Move, o Leap Motion é capaz de detetar mudanças na postura da mão. O Leap Motion, ao contrário do Myo, tem a capacidade de rastrear todos os tipos de gestos de ambos os braços. O Leap Motion como se concentra exclusivamente no rastreamento dos braços e das mãos têm um SDK que permite um desenvolvimento mais fácil de aplicações que utilizam apenas esses gestos, ao contrário do Kinect que para além de rastrear os braços e mãos também rastreia toda a área do corpo apanhada pela câmara, o que faz que a criação de aplicações que reconhecem apenas os gestos das mãos muito mais complexa. O Leap Motion é também mais barato do que o Kinect.

O Kinect tem uma precisão ligeiramente maior do que o Leap Motion, mas esta vantagem é diminuída devido à curva de aprendizagem para desenvolver aplicações que reconhecem gestos das mãos no Kinect ser muito maior do que a do Leap Motion. Este aumento na curva de aprendizagem deve-se ao facto de um programador que utiliza o Kinect não ter nenhuma forma direta para obter os parâmetros dos gestos da mão, enquanto um programador que utiliza o Leap Motion apenas necessita aprender os vários parâmetros do sensor (Marin, Dominio & Zanuttigh, 2014).

Um resumo dos vários pontos analisados pode ser visto na tabela 1.

	Reconhece posturas das mãos	Área do corpo rastreada pelo dispositivo	Curva de Aprendizagem para Reconhecer os Gestos da Mão	Preço do Sensor
Sony Move	Não	Apenas a mão que está a segurar o dispositivo	Baixa	80€/2 unidades
Myo	Limitado	Apenas a mão e braço em que o dispositivo está	Baixa	200€
Kinect	Sim	Toda a área do corpo apanhada pela câmara	Alta	150€ Descontinuado
Leap Motion	Sim	Apenas as mãos e braços	Média	70€

Tabela 1. Resumo dos vários sensores de movimento.

Pelas razões descritas na frase anterior, foi decidido que o Leap Motion seria o sensor de movimento a utilizar no desenvolvimento do sistema de reconhecimento de gestos.

As principais razões para esta escolha foram as capacidades do sensor, o preço e a menor curva de aprendizagem para desenvolver software para o sensor.

2.2.2 Leap Motion

2.2.2.1 Características do Sensor

O Leap Motion usa câmaras e infravermelhos para gerar uma imagem das mãos e braços do utilizador. Esta imagem é processada posteriormente para remover o ruído. Um modelo das mãos, dedos e braços que o utilizador tiver na área rastreada pelas câmaras do dispositivo é criado. O dispositivo Leap Motion é ligado ao computador através de um cabo USB.

Para um programador poder usar o Leap Motion é necessário instalar o SDK que pode ser obtido gratuitamente no site do Leap Motion. Este SDK contém várias API's que permitem o programador facilmente analisar os gestos feitos pelas mãos do utilizador.

O sistema de coordenadas usado pelo Leap Motion é tridimensional (Guna et al., 2013). O eixo:

- x – Indica, em milímetros, o quanto á direita ou á esquerda um ponto a ser rastreado está do centro do dispositivo. Valores positivos desta coordenada indicam que o ponto a ser rastreado está á direita do centro do dispositivo enquanto valores negativos indicam que o ponto está á esquerda do centro.
- y – Indica a altitude, em milímetros, do ponto a ser rastreado. Valores negativos desta coordenada são impossíveis, já que o Leap Motion não consegue rastrear pontos que estão por baixo da sua superfície.
- z – Indica, em milímetros, o quanto á frente ou atrás um ponto a ser rastreado está do centro do dispositivo. Valores positivos desta coordenada indicam que o ponto a ser rastreado está á frente do centro do dispositivo enquanto valores negativos indicam que o ponto está atrás do centro.

Neste sistema de coordenadas, o ponto [0,0,0] está no centro do dispositivo, conforme se pode verificar na figura 13, com o ponto a ser rastreado a tocar a superfície do dispositivo.

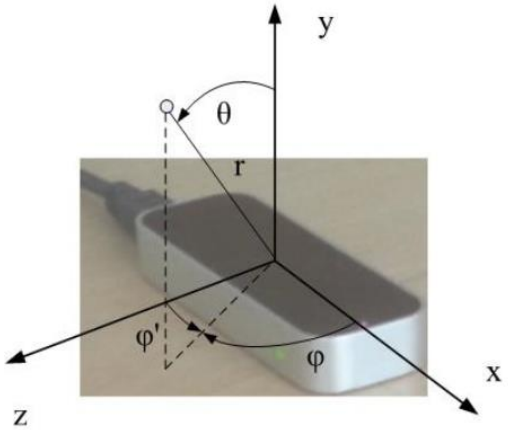


Figura 13. Sistema de coordenadas do Leap Motion (Guna et al., 2013).

Outra informação importante quando se trabalha com o sistema de coordenadas do Leap Motion é a distância entre o centro do Leap Motion e o ponto a ser rastreado, a inclinação e o azimute do ponto a ser rastreado. Esta informação é representada na figura 13 respetivamente por r, θ e ϕ'. Estes parâmetros permitem simplificar a análise dos pontos rastreados pelo Leap Motion, ao permitir que programadores sejam capazes de trabalhar com distâncias e ângulos em vez terem de aprender a utilizar um novo sistema de coordenadas especificamente desenhado para o Leap Motion (Guna et al., 2013).

2.2.2.2 Desempenho

Para analisar o desempenho do sensor Leap Motion é necessário analisar as seguintes características:

1. Exatidão: A exatidão refere-se à conformidade dos valores dos parâmetros recolhidos pelo Leap Motion com os mesmos parâmetros da mão do utilizador no mundo real.
2. Precisão: A precisão refere-se ao grau de variação dos valores dos parâmetros recolhidos pelo Leap Motion quando o utilizador faz várias vezes o mesmo gesto.
3. Consistência: A consistência do Leap Motion é a probabilidade do dispositivo recolher um número constante de amostras por segundo.

As duas primeiras características permitiram analisar se o sensor é realmente capaz de recolher gestos sem alterar muito as características do gesto original. A consistência é importante devido ao facto de um sensor inconsistente tornar difícil a sincronização com outros sistemas. A precisão, exatidão e consistência do funcionamento do Leap Motion pode ser analisado considerando diversos aspetos:

1. Dispersão espacial das medidas ao longo do tempo: Variação da precisão e consistência do dispositivo ao longo do tempo, com um exemplo deste aspeto sendo o dispositivo ter uma queda na sua precisão ou consistência quando é utilizado continuamente por uma certa quantidade de tempo;
2. Distorção espacial da exatidão: A variação da exatidão do dispositivo nas várias zonas rastreadas pelo dispositivo, com um exemplo deste aspeto podendo ser a exatidão do dispositivo ser maior no centro da área rastreada do que nos cantos;
3. Consistência da frequência de amostragem: A probabilidade das amostras recolhidas num segundo pelo dispositivo serem semelhantes à frequência de amostragem (valor médio de amostras recolhidas num segundo pelo dispositivo).

A dispersão espacial das medidas ao longo do tempo do Leap Motion pode em várias ocasiões ter grandes variações, especialmente no que diz respeito á consistência do sensor. À medida que mais amostras são recolhidas pelo sensor, existe uma probabilidade que a consistência da recolha sofra uma grande queda. Esta queda pode ser pequena ou grande, mas existe pouca probabilidade do sensor conseguir recuperar a sua consistência quando esta queda acontece (Guna et al., 2013).

No referente á distorção espacial da precisão do dispositivo, este mostra uma boa exatidão em geral sendo a distorção espacial muito reduzida com o maior valor tomado sendo menor que 0.5mm e nos melhores casos menor que 0.01mm. Ao mesmo tempo o sensor tem grande dificuldades a estabilizar o seu rastreio de objetos caso o valor da coordenada z seja maior que zero, ou seja, caso os objetos estejam à frente do dispositivo (Guna et al., 2013).

Em condições ótimas o Leap Motion é capaz de obter mais de 1000 amostras em 20 segundos. Obter condições ótimas é difícil, já que o dispositivo mostra várias inconsistências com a sua frequência de amostragem e têm um espaço sensorial limitado com grande diminuição do número de amostras. Estes problemas têm uma grande probabilidade de acontecer se os braços estiverem 25cm acima do dispositivo ou se o objeto a analisar estiver a mais de 10 cm de distância do centro do dispositivo num plano bidimensional (Guna et al., 2013).

O Leap Motion tem uma lista de limitações para as várias versões do SDK, as quais são devidamente identificadas no seu site tanto para a versão do SDK mais recente[6] como para as versões mais antigas[7]. Algumas limitações do software do Leap Motion atualmente conhecidas incluem:

- O dispositivo tem dificuldades a rastrear algumas posturas das mãos. Caso isto aconteça, podem ser visualizadas algumas imperfeições no modelo das mãos e braços. Essas imperfeições na visualização incluem a inversão da posição das mãos do modelo, problemas na inicialização e surgimento de posturas erradas nas mãos do modelo, versões mais recentes do SDK resolveram este problema.
- O rastreamento pode não funcionar caso o utilizador esteja a utilizar acessórios nas suas mãos e braços.
- Performance do rastreamento pode ter quedas quando o dispositivo está próximo de superfícies reflexivas, especialmente superfícies que refletem raios infravermelhos.

2.3 CLASSIFICADORES

Os classificadores são algoritmos que irão receber vários exemplos dos valores que os parâmetros de uma classe podem ter. Através desses exemplos o classificador irá criar um modelo que quando recebe um novo exemplo prevê a que categoria da classe os exemplos pertencem através da análise dos seus parâmetros. Estas categorias podem já ser conhecidas pelo classificador ou o classificador irá criar as categorias através da análise dos exemplos dados. Devido a estas características, os classificadores são um fator indispensável para qualquer sistema que queira prever a que categoria, já existente ou criada pelo classificador, um certo exemplo pertence.

[6] https://developer.leapmotion.com/known-issues
[7] https://developer-archive.leapmotion.com/documentation/cpp/supplements/SDK_Release_Notes.html

2.3.1 Análise Comparativa

É necessário efetuar uma análise comparativa dos vários métodos de aprendizagem que podem ser teoricamente utilizados para fazer o reconhecimento de gestos utilizando o Leap Motion. Esta análise permitirá reconhecer que métodos são mais eficazes para permitir uma boa exatidão do sistema de reconhecimento de gestos e que zonas têm fraquezas.

Os métodos de aprendizagem analisados foram os que durante a análise dos estudos feitos na área, foram no mínimo parcialmente utilizados para criar sistemas de reconhecimento.

Support Vector Machines: Os Support Vector Machine (SVM) são modelos de aprendizagem supervisionada que permitem fazer a classificação de duas ou mais classes conhecidas. O funcionamento do modelo é baseado em fornecer ao modelo um grupo de dados de teste, tendo associada uma classe a cada exemplo de teste, e a partir desses dados é criada uma área que separa as várias classes, tentando manter sempre uma certa área vazia entre ambas as classes. A eficiência do modelo é especialmente afetada pela função kernel que o modelo utiliza, com a função utilizada podendo influenciar a exatidão do modelo ou o tempo que o modelo demora a executar. Existem quatro kerneis que podem ser utilizadas: linear, polinomial, sigmóide e função de base radial (RBF). As bibliotecas LIBSVM e LIBLINEAR permitem a utilização de modelos SVMs por aplicações terceiras (Cortes, 1995; Chang & Lin, 2011).

Hidden Markov Models: Os Hidden Markov Models (HMM) são modelos estatísticos em que existem vários estados, e em que cada estado tem várias probabilidades de transitar para um diferente estado. Cada estado terá também uma probabilidade de o sistema começar nele e uma probabilidade de o estado não se alterar. A principal diferença do HMM de outros modelos estatísticos semelhantes, como as cadeias de Markov, é que a sequência de estados que o modelo gera é desconhecido, esta diferença aumenta a complexidade do modelo, mas ao mesmo tempo aumenta a funcionalidade do modelo. O HMM pode ser utilizado como um método de aprendizagem supervisionado quando combinado com certos algoritmos, como o de Baum–Welch, que permitem a criação de um HMM através da análise dos resultados de saída do sistema que produz os dados de treino (Baum & Petrie, 1966; Baum, 1972; Scheffer, Decomain & Wrobel, 2001).

Hidden Conditional Random Field: Os Hidden Conditional Random Field (HCRF) são métodos utilizados para classificar objetos. O modelo HCRF utiliza estruturas ocultas para tentar minimizar os problemas criados por variáveis ocultas que podem prejudicar a exatidão do modelo. Este tipo de modelo é muito eficaz em casos onde o sistema necessita classificar sinais não estacionários e onde existe muita dependência entre os vários estados, exemplos de áreas que tem estas características são o reconhecimento da fala e de gestos dinâmicos. Ao mesmo tempo a utilização de estruturas ocultas aumenta o número de cálculos que o modelo vai necessitar realizar. Na figura 14 está representada a estrutura de um modelo HCRF (Sung et al., 2007; Gunawardana et al., 2005).

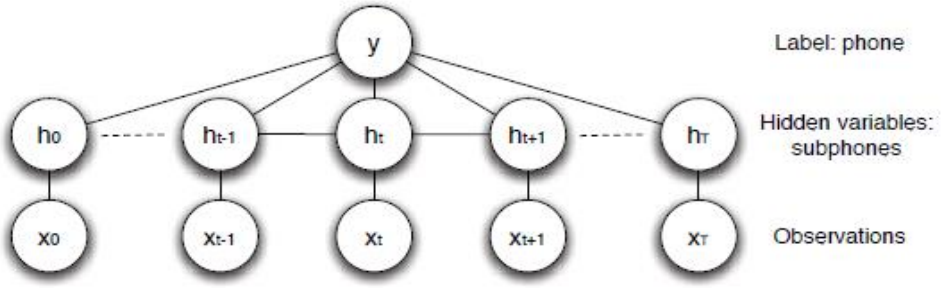


Figura 14. Estrutura de um modelo HCRF (Sung et al., 2007).

Hidden Conditional Neural Field: Os Hidden Conditional Neural Field (HCNF) é um modelo de aprendizagem baseado no HCRF que tem como principal objetivo resolver um dos maiores problemas desse modelo, sendo esse problema a sua incapacidade de resolver problemas com parâmetros não lineares. A resolução deste problema é conseguida ao adicionar funções "gate". Uma função "gate" irá receber vários parâmetros e irá transformá-los de forma que eles possam ser resolvidos usando os métodos do modelo HCRF, resolvendo com isto o problema criado pelo HCRF não conseguir resolver problemas com parâmetros não lineares (Fujii, Yamamoto & Nakagawa, 2011).

Na figura 15 está representada a estrutura de um modelo HCNF. O modelo ao receber vários dados de entrada irá dar esses dados a K funções "gate", com K sendo um número definido pelo programador, onde são processados. Os resultados obtidos pelas funções "gate" são combinados formando um dado de saída (Fujii, Yamamoto & Nakagawa, 2011).

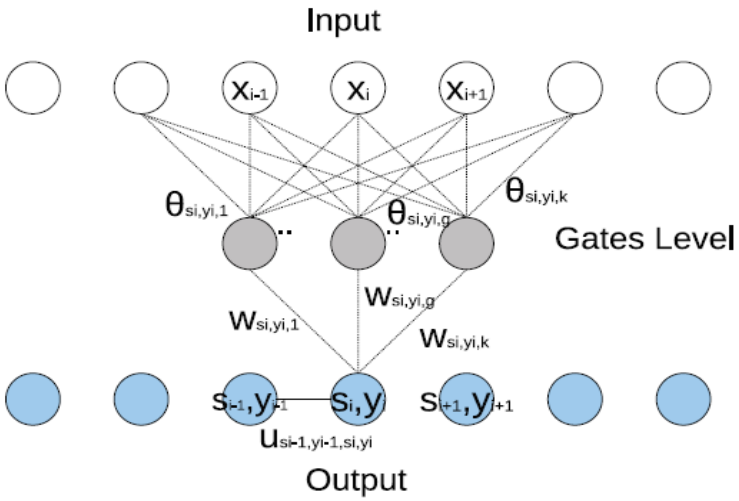


Figura 15. Estrutura de um modelo HCNF (Fujii, Yamamoto & Nakagawa, 2011).

Árvores de Decisão: Árvores de decisão podem ser utilizadas como um método de aprendizagem supervisionada para fazer a classificação de várias classes. O modelo baseia-se em escolher a variável que permite atribuir o maior número de casos de treino à classe a que estes pertencem. Este processo será repetido até todos os casos de treino estarem atribuídos ou o tamanho da árvore atingir uma dimensão máxima. As maiores

vantagens deste modelo são o classificador criado ser fácil de entender e implementar, como também o tempo de execução do modelo ser muito menor quando comparado com outros modelos supervisionados. As maiores fragilidades das árvores de decisão são a baixa exatidão dos resultados criados pelo modelo e o facto de as árvores de decisão muito facilmente fazerem um overfiting dos dados, tendo o classificador bons resultados durante o treino, mas tendo maus resultados durante os testes. (Breiman et al., 1984).

Random Forests: As Random Forests são um método de aprendizagem supervisionada que tenta melhorar os resultados das árvores de decisão. O modelo cria n árvores de decisão, com n sendo um número inteiro maior que zero dado pelo utilizador. A principal diferença de uma árvore de decisão normal sendo que cada árvore de decisão da floresta só terá acesso a uma parte aleatória das variáveis dos dados de treino sendo classificados. Esta limitação fará que todas as árvores de decisão criadas sejam diferentes. A principal vantagem das random forests é que, desde que elas tenham um número de árvores de decisão suficientes, elas irão geralmente gerar resultados bons e permitirão explorar zonas que outros modelos não exploram, especialmente árvores de decisão normais (Ho, 2002).

K-means: O k-means é um modelo de aprendizagem não supervisionada feita através de clustering. Um cluster é um conjunto de vários pontos que o algoritmo que cria os clusters considera semelhantes. Este modelo irá criar k clusters, com k sendo um número inteiro maior que zero obtido do utilizador ou gerado pelo algoritmo. O algoritmo irá gerar k centroides que serão colocados em certos pontos do domínio com os casos de treino que estão a ser utilizados, estes centroides irão ser iterativamente movidos pelo algoritmo até todos eles terem sido posicionados de uma maneira que todos os clusters maximizem os requisitos do algoritmo. Versões tradicionais deste método exigem que o número de clusters que o método cria seja dado pelo utilizador antes de iniciar o processo de clustering. Este modelo tem várias fraquezas, a primeira é o algoritmo que o k-means utiliza criar sempre clusters esféricos, podendo resultar na criação de clusters incorretos quando os dados têm uma forma não esférica. Outra fraqueza do modelo é estar programado para preferir que todos os clusters tenham tamanhos iguais, isto pode criar problemas em situações em que o número de dados de treino para cada classe não é igual (Lloyd, 1982).

Os métodos de aprendizagem HMM, HCRF, HCNF são demasiados complexos para o âmbito deste trabalho, isto deve-se a estes modelos serem mais apropriados para sistemas dinâmicos, onde existem alterações constantes nos parâmetros que o modelo utiliza para fazer a sua previsão. Outro problema é o facto de não existir bibliotecas disponíveis para implementar os métodos HCRF e HCNF. As bibliotecas que existem para o método HMM são demasiado básicas e ineficazes, com apenas as funções mais básicas do modelo sendo implementadas. Estes problemas levam a concluir que estes métodos de aprendizagem não são os mais apropriados para serem utilizadas em sistemas de reconhecimento de gestos.

A utilização de árvores de decisão não é apropriada já que estas árvores não são capazes de lidar eficazmente com sistemas tão complexos, devido a elas serem demasiado rígidas no seu funcionamento. A utilização de uma random forest resolveria o problema das árvores de decisão, mas ao mesmo tempo este método de aprendizagem poderia não demonstrar o seu potencial neste sistema, devido aos parâmetros que o sistema final irá receber serem todos essenciais.

O método de aprendizagem k-means poderia, teoricamente, conseguir fazer a divisão dos gestos de maneira que permitisse o sistema reconhecer um novo gesto. O maior problema surge das várias limitações do k-means referidas no resumo do método de aprendizagem. Estas limitações fazem com que este método não seja apropriado para este trabalho, já que não há maneira de saber se os parâmetros recolhidos irão satisfazer as limitações do modelo.

O método SVM apresenta uma maneira eficaz de fazer o treino do sistema de reconhecimento, tendo este método como um dos seus objetivos principais criar a divisão máxima entre classes. A utilização de kerneis também permite que o método tenha uma grande versatilidade para lidar com vários possíveis casos. Também existem várias bibliotecas, como a LibSVM e LIBLINEAR, que podem ser utilizadas para executar versões otimizadas do método SVM. Apesar de o método de aprendizagem SVM ter algumas limitações, como a sua incapacidade de lidar com situações que mudam ao longo do tempo, dado o âmbito do nosso projeto estas limitações não irão criar nenhum problema.

Devido às conclusões que surgiram da análise dos vários métodos de aprendizagem, foi decidido utilizar o classificador SVM para fazer o treino do sistema de reconhecimento de gestos.

2.3.2 SVM

Com base em vários dos artigos lidos, foi concluído que o melhor método de aprendizagem para treinar modelos que permitam fazer o reconhecimento de gestos estáticos é através da utilização de SVMs (Cortes, 1995; Chang & Lin, 2011; Burges, 1998).

SVMs são métodos de aprendizagem supervisionada que permitem criar e treinar modelos de classificação. Os SVMs são classificadores lineares e como tal eles são particularmente bons a resolver problemas que podem ser resolvidos usando soluções lineares. Mesmo assim existem métodos que permitem a realização de classificação não-linear usando algoritmos SVM (Cortes, 1995; Burges, 1998).

Para treinar um modelo usando um algoritmo SVM, vai ser necessário que ele receba um conjunto de exemplos de resultados do sistema que se pretende classificar. Cada um destes exemplos deve conter uma variável que identifique a que classe esse exemplo pertence e os parâmetros necessários para o SVM poder analisar e conseguir treinar o modelo de classificação.

A tabela 2 contêm um possível conjunto de exemplos que podiam ser utilizados para treinar um modelo SVM, nesse exemplo queremos treinar o modelo de maneira que quando lhe for dado um novo exemplo seja capaz de corretamente identificar se o exemplo pertence à classe A ou à classe B. O parâmetro 1 não permitirá o modelo identificar a classe correta devido ao seu valor nunca variar. Ao mesmo tempo como o valor nunca muda ele não irá prejudicar o resultado final do modelo. O parâmetro 2 é o parâmetro que realmente permite identificar a que classe o exemplo pertence devido a haver uma clara distinção entre os valores para cada classe, nomeadamente com os exemplos da classe A tendo um valor positivo no parâmetro 2 e os exemplos da classe B tendo um valor negativo. O parâmetro 3 não permitirá o modelo identificar a classe correta devido a não haver uma distinção clara dos valores para cada uma das classes. É também importante notar que deixar o parâmetro 3 no modelo durante o treino pode prejudicar a exatidão do modelo devido ao facto de os valores do modelo terem constantes mudanças o que pode confundir o algoritmo, como tal esse parâmetro deve ser removido.

Categoria	Parâmetro 1	Parâmetro 2	Parâmetro 3
A	1	1	1
A	1	2	2
A	1	3	3
A	1	4	2
A	1	5	1
B	1	-1	2
B	1	-2	3
B	1	-3	2
B	1	-4	1
B	1	-5	2

Tabela 2. Exemplo de um possível conjunto de exemplos usados para treinar um modelo SVM.

O número de classes objetivo tem de ser sempre maior do que um, já que caso só seja usado uma não será possível fazer a separação linear e o modelo criado será basicamente inútil. Caso os exemplos obtidos não tenham um parâmetro que serve para indicar a classe a que cada exemplo pertence é necessário utilizar um método de aprendizagem não supervisionada que permita criar clusters. Esses clusters serão compostos por vários exemplos que o método de aprendizagem não supervisionada considera similares, e serão utilizados como classes quando o SVM for executado.

Um dos principais objetivos do SVM quando faz a divisão das classes é maximizar a margem de separação entre classes. Duas classes com uma boa margem de separação terão uma menor probabilidade de encontrar situações em que é difícil classificar um novo caso devido a este estar demasiado perto da linha de separação das classes. Na figura 16 estão exemplos de modelos criados usando SVMs em que a divisão das classes é boa e a margem de separação é grande (16.a), em que a divisão das classes é boa, mas a margem de separação é pequena (16.b) e em que o SVM não conseguiu fazer uma boa separação das classes (16.c) (Cortes, 1995; Burges, 1998).

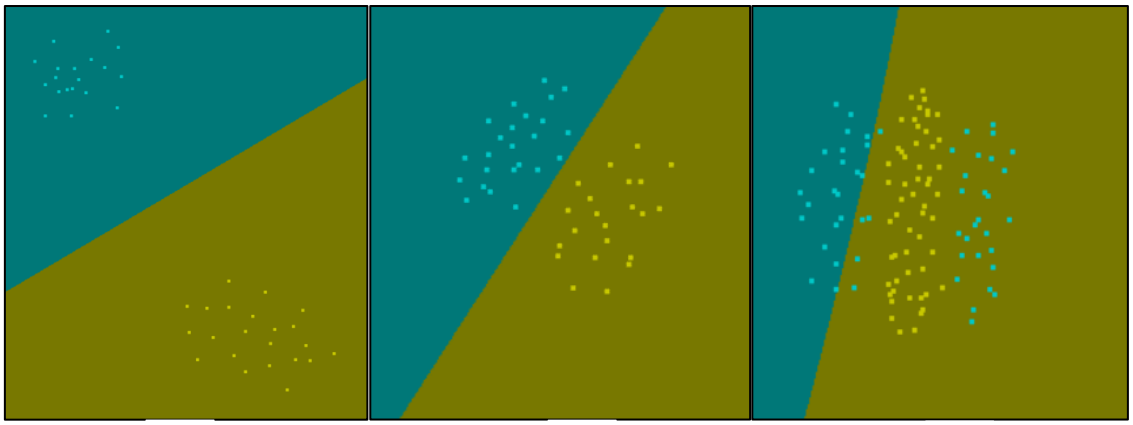


Figura 16. a) Representação de uma boa separação de classes com larga margem, b) Representação uma boa separação de classes com pequena margem, c) Representação de uma má separação de classes.

O treino de um modelo utilizando um modelo de aprendizagem SVM consiste em o algoritmo, escolher o ponto, para cada classe considerada, mais próximo a exemplos de outras classes. O algoritmo criará uma linha reta que conectará os pontos selecionados pelo algoritmo. O hyperplano que dividirá as classes será criado ao criar uma linha reta, perpendicular à original e que intersecta a linha reta original no seu ponto médio. É esperado que essa nova linha reta consiga dividir os exemplos das diferentes classes com a maior margem de separação possível.

A figura 17 ilustra este processo, com a primeira classe sendo os pontos pretos e a segunda os pontos vermelhos, os pontos escolhidos pelo algoritmo têm um contorno azul e estão conectados por uma linha reta com cor azul que é perpendicular à linha preta grossa que é o hyperplano que divide as duas classes (Cortes, 1995; Burges, 1998).

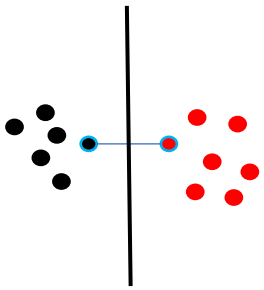
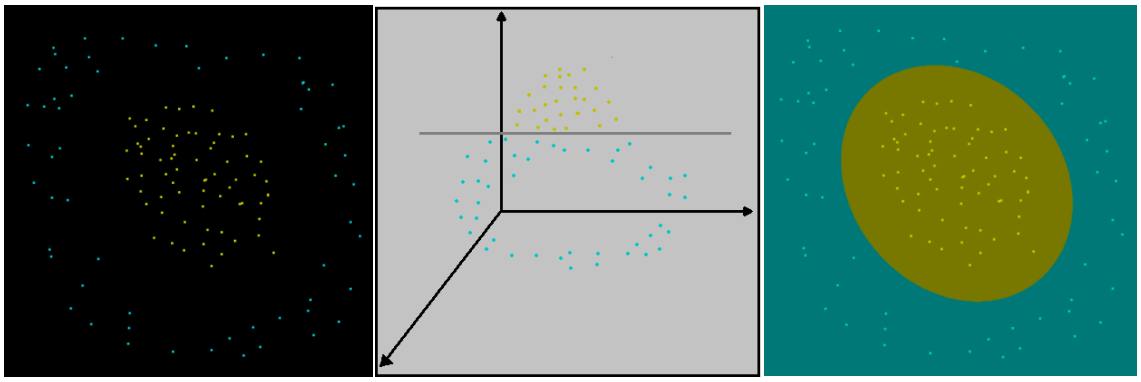


Figura 17. Criação de um hyperplano em SVM.

Vários SVMs também utilizam um conjunto de métodos chamadas kerneis que permitem que o SVM consiga resolver problemas não-lineares. Os kerneis utilizam um tipo de algoritmo que irão alterar o espaço dimensional em que os dados estão inseridos. Esta alteração ao espaço dimensional tem como objetivo permitir ao SVM separar linearmente os dados que anteriormente não conseguia separar. Quando o SVM concluí a separação dos dados o espaço dimensional retorna ao que era originalmente. A figura 18 mostra o funcionamento de um kernel RBF.



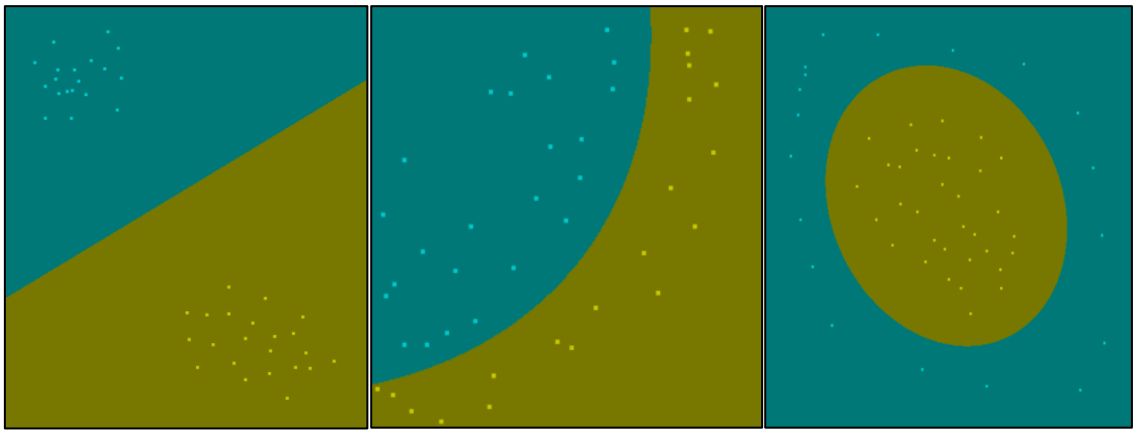
a) b) c)

Figura 18. Funcionamento de um kernel RBF, a) caso em que é impossível fazer separação linear, b) transformação que o kernel RBF faz ao espaço dimensional, c) separação dos pontos após o espaço dimensional retornar ao original.

Existem três tipos de kernel que são usados regularmente, estes são:

- Linear (SVM clássico)
- Polinomial
- RBF (Função de Base Radial)

Na figura 19 estão representados 3 casos em que o método de aprendizagem SVM necessitou utilizar deferentes kerneis para separar as classes. Com o kernel em 19.a sendo linear, o kernel em 19.b sendo polinomial e o kernel em 19.c sendo RBF.



a) b) c)

Figura 19. a) Modelo SVM usando um kernel linear, b) usando um kernel polinomial, c) usando um kernel RBF.

2.4 Aplicações Existentes

Nos anos mais recentes as técnicas para interagir com software usando o Leap Motion têm aumentado.

O estudo feito por Maruyama et al. (2017) mostra que é possível utilizadores autenticarem-se usando apenas o Leap Motion. Isto foi conseguido ao analisar a posição das junções de cada dedo do utilizador, em especial a distância entre junções no mesmo dedo e a distância e ângulo das mesmas junções em diferentes dedos.

Outras áreas em que o Leap Motion tem tido grandes avanços são as áreas da linguagem gestual e escrita no ar. Um dos métodos usados para ambas as áreas é o proposto no estudo de Kumar et al. (2017-2) que usa uma combinação de uma SVM e duas redes neuronais que funcionaram como classificadores, este método demonstrou uma precisão superior a 60% o que indica que têm potencial, mas que ainda necessita ser melhorada.

Na área de reconhecimento de linguagem gestual já existem vários métodos para reconhecer línguas gestuais diferentes, usando o Leap Motion. O estudo feito por Simos & Nikolaidis (2016) consiste em usar um SVM e os dados posicionais tridimensionais obtidos pelo Leap Motion para permitir um maior reconhecimento da linguagem gestual, obtendo uma alta exatidão. O estudo feito por Hisham & Hamouda (2017) melhorou o método ao incluir, para além do SVM, os seguintes algoritmos: K- Nearest Neighbour, redes neuronais artificiais e dynamic time warping, através disto foi possível aumentar o reconhecimento de gestos dinâmicos, o que é uma necessidade para linguagens gestuais que utilizam este tipo de gestos.

Na área de escrita no ar já existe a capacidade se fazer o reconhecimento de textos 3D, mais especificamente, como demonstrado no estudo feito por Kumar et al. (2017-1), é possível desenhar texto no ar através de gestos em três dimensões ao usar a ponta dos dedos dentro do campo de visão do Leap Motion. O sistema tem um reconhecimento superior a 90% e uma capacidade de segmentação superior a 80%. Exemplos disto podem ser vistos na figura 20.

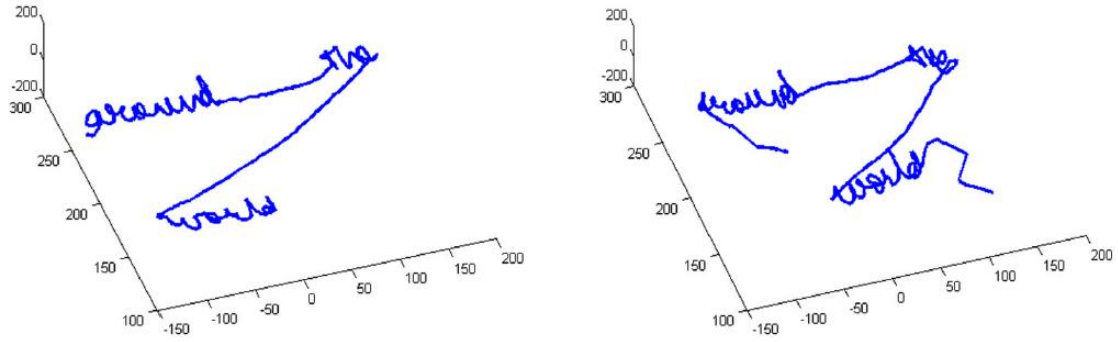


Figura 20. Exemplos de escrita no ar feita por duas pessoas diferentes (Kumar et al., 2017-1).

O Leap Gesture foi proposto por Nowicki et al. (2014). Nowicki et al. propuseram um modelo próprio de classificação de gestos que tornava o contexto exterior ao gesto irrelevante e aumentava as reações que o sistema pudesse ter a um gesto ao dividir os gestos em dois grupos: gestos de ação e gestos parametrizados.

O primeiro grupo são gestos em que quando o modelo usado acaba de reconhecer o gesto, o método utilizado não irá analisar outros parâmetros do gesto e faz uma ação que foi estipulada quando esse gesto é reconhecido. Quando um gesto é reconhecido e pertence ao segundo grupo, o método vai analisar outros parâmetros do gesto e dependo do valor desses parâmetros pode fazer diferentes ações. Por exemplo, se o sistema reconhecer que o utilizador está a acenar com uma mão. Caso esteja definido como um gesto de ação o sistema irá executar imediatamente a ação que está

programada para acontecer caso esse gesto seja reconhecido. Caso seja um gesto parametrizado o sistema irá analisar outros parâmetros como a velocidade da mão e irá fazer ações diferentes dependendo da velocidade com que a mão está a acenar.

O Leap Gesture utiliza vários modelos de aprendizagem supervisionada para permitir o reconhecimento dos gestos. O Leap Gesture utiliza duas abordagens diferentes para classificar os gestos, o fator que decide qual a abordagem que vai ser utilizada é se o gesto é estático ou dinâmico. A necessidade de se usar duas abordagens diferentes deve-se ao facto de um tipo de gestos ser independente do tempo enquanto o outro é dependente.

Nos casos em que o gesto a ser reconhecido é estático, a abordagem utilizada irá usar SVMs para fazer a classificação do gesto feito pelo utilizador. Esta classificação será feita antes do processamento posterior do modelo das mãos e braços criado pelo Leap Motion. Foi escolhido um kernel do tipo RBF, após vários testes, para ser utilizado no SVMs, já que esta função kernel mostrava a maior capacidade de separação num tempo aceitável. A visão geral de como um gesto estático é processado pelo Leap Gesture está ilustrado na figura 21.

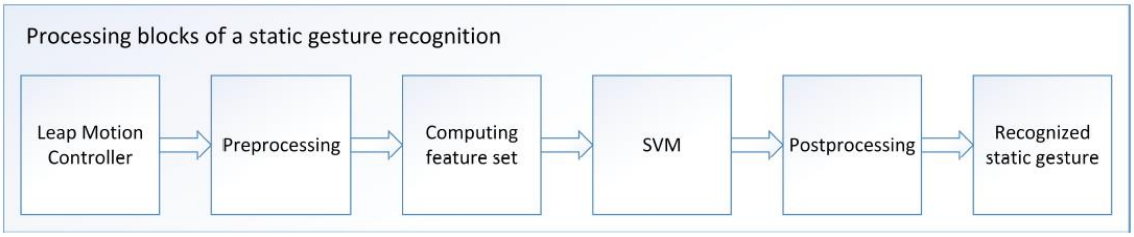


Figura 21. Processamento de um gesto estático em Leap Gesture (Nowicki et al., 2014).

Para o processamento de gestos dinâmicos o Leap Gesture irá utilizar uma combinação de um método de aprendizagem não supervisionado, nomeadamente o k-means, e um método de aprendizagem supervisionado, nomeadamente um modelo de Markov escondido. O k-means é utilizado para criar clusters de observação através da classificação dos dados obtidos pelo Leap Motion. É usado um algoritmo de Baum-Welch em conjunto com os clusters de observação para fazer o treino do HMM. A visão geral de como um gesto dinâmico é processado pelo Leap Gesture está ilustrado na figura 22.

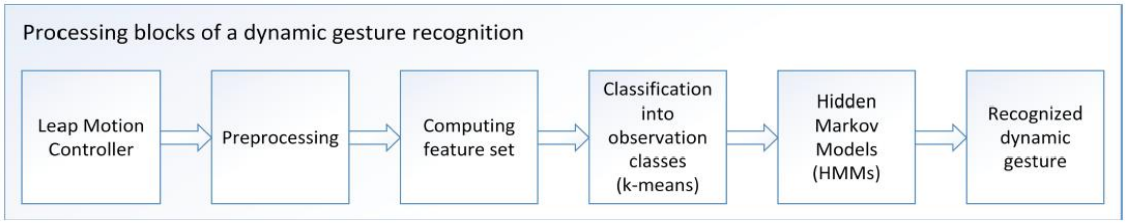


Figura 22. Processamento de um gesto dinâmico em Leap Gesture (Nowicki et al., 2014).

Segundo os autores, os resultados do reconhecimento para os gestos estáticos foram bons com um reconhecimento de 85% no pior dos casos e um reconhecimento de 99% no melhor. O melhor resultado obtido para o reconhecimento de gestos dinâmico foi

apenas 80%. O problema encontra-se num grande nível de ruído nos dados obtidos pelo Leap Motion quando a mão está em movimento, em particular na zona dos dedos.

Du et al. (2017) apresenta uma outra forma de criar um sistema de reconhecimento de gestos. Este sistema apenas se foca reconhecimento de gestos estáticos. Este modelo utiliza suport vector machines com kernel do tipo RBF para fazer a classificação supervisionada dos gestos.

A principal característica distintiva deste estudo é a utilização de imagens do Leap Motion obtidas durante o pré-processamento. Estas imagens serão simplificadas, retendo apenas a área da mão. Estas imagens podem ser vistas na figura 23 em que são mostrados vários grupos de duas imagens, em que a primeira imagem de um grupo é a imagem da mão recolhida pelo Leap Motion e a segunda é a imagem simplificada da mão.

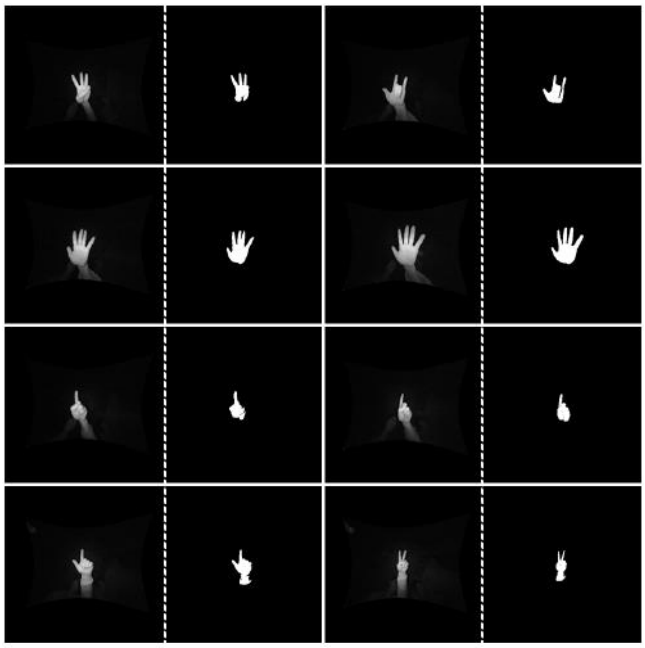


Figura 23. Imagens simplificadas (Du et al., 2017).

Para obter, das imagens simplificadas das mãos, a informação necessária para treinar o modelo de reconhecimento será usado o histograma de gradientes orientados. O histograma de gradientes orientados é um descritor de características visuais que permite detetar e reconhecer objetos em imagens.

Antes do SVM fazer a classificação dos gestos, vários parâmetros do Leap Motion irão ser fundidos e simplificados usando um pré-processamento do tipo “principal component analysis”. Este pré-processamento cria um parâmetro. Era esperado que este parâmetro diminuísse o tempo necessário para fazer a classificação dos gestos e aumentar a precisão da classificação dos gestos.

Os parâmetros usados para criar o parâmetro são: os ângulos das pontas dos dedos, a distância entre as pontas dos dedos, a distância da ponta dos dedos e o histograma de gradientes orientados. Durante a fusão dos vários parâmetros foram feitas várias

alterações ao coeficiente dos pesos do histograma de gradientes orientados. A estrutura deste sistema de reconhecimento de gestos é ilustrada na figura 24.

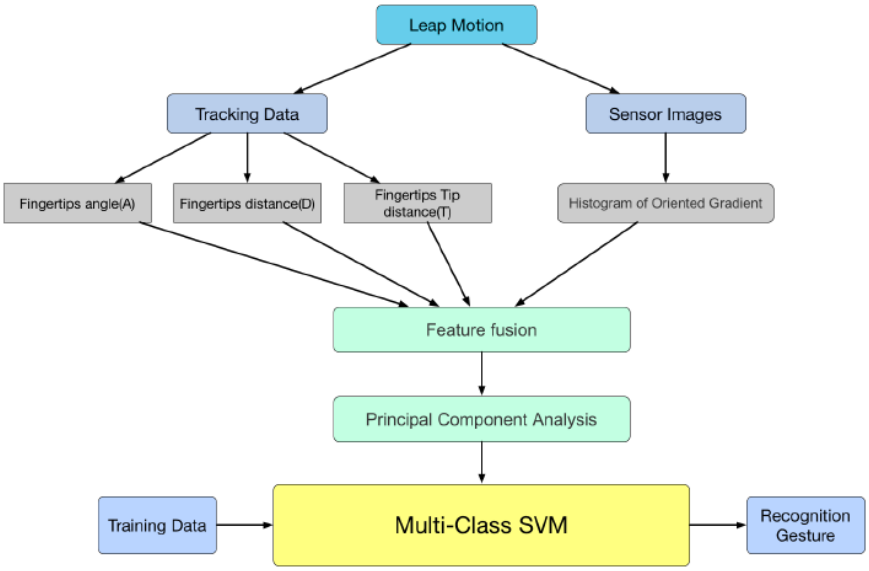


Figura 24. Estrutura do sistema de reconhecimento de gestos de Du et al. (2017).

Este sistema de reconhecimento de gestos mostrou consistentemente bons resultados, quando o peso do coeficiente do histograma de gradientes orientados é 4 ou 5 vezes maior do que o original, com a precisão ao reconhecer os gestos nunca sendo menor do que 95%.

O estudo de Marin, Dominio & Zanuttigh (2016) apresenta um sistema de reconhecimento de gestos que combina o Leap Motion e o Kinect, esperando que essa combinação aumentasse a precisão do reconhecimento de gestos e apresenta-se melhores resultados do que o estudo feito anteriormente pelo mesmo grupo (Marin, Dominio & Zanuttigh, 2014).

No estudo foram utilizados dois modelos de aprendizagem supervisionada para treinar o algoritmo de classificação: o suport vector machines com um kernel RBF e o random forests.

Foram também utilizados três métodos para escolher os parâmetros usados durante o reconhecimento dos gestos, estes foram: F-Score, Forward Sequential Selection e Random Forests.

Os parâmetros do Leap Motion utilizados são a distância, ângulo e elevação da pontas dos dedos em relação à palma da mão e a posição das pontas de dedos no espaço. Os parâmetros da câmara de profundidade são a correlação e curvatura da mão, a distância de cada ponto do contorno da mão do centro da palma da mão e o tamanho e número de componentes conectados do gesto. A estrutura deste sistema de reconhecimento de gestos está representada na figura 25.

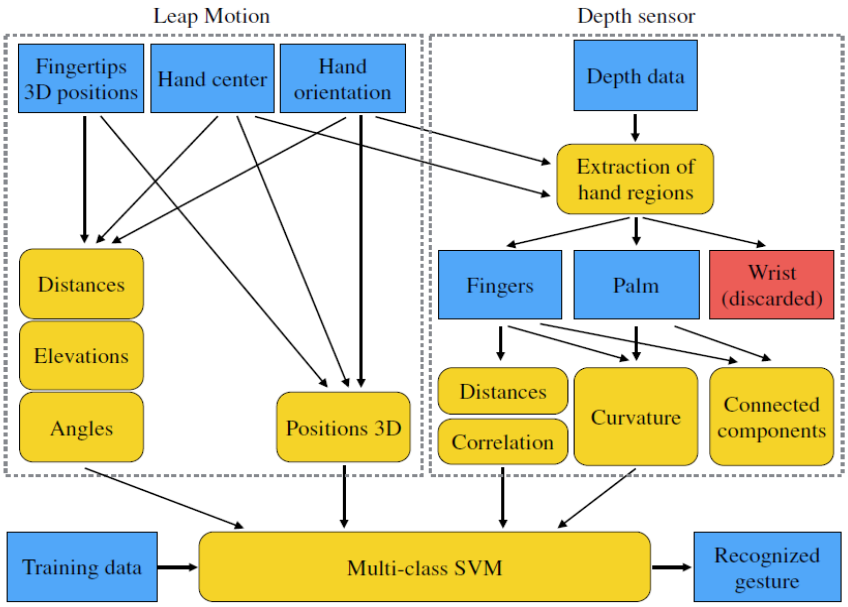


Figura 25. Estrutura do sistema de reconhecimento de gestos de Marin, Dominio & Zanuttigh (2016).

Os resultados do estudo indicam que a utilização de SVM para o treino do algoritmo obtém os melhores resultados em particular quando utiliza os métodos random forests ou Forward Sequential Selection para escolher os parâmetros utilizados, com a exatidão sendo maior do que 95%. A utilização de random forests para fazer o treino do algoritmo também obteve bons resultados, mas estes resultados foram quase sempre 2% menores do que os resultados usando SVM. Os resultados também foram consistentemente superiores aos obtidos em Marin, Dominio & Zanuttigh (2014).

Ambos os estudos referem que eles se concentraram apenas no reconhecimento de gestos estáticos e que um estudo sobre o reconhecimento de gestos dinâmicos seria um objetivo para trabalhos futuros.

O estudo de Lu, Tong & Chu (2016) apresentou um novo método para permitir o reconhecimento de gestos dinâmicos utilizando apenas o Leap Motion, o sistema obtêm os dados de profundidade da janela do Leap Motion e através desses dados irá extrair certas funcionalidades que considera importantes para o reconhecimento dos gestos, estas funcionalidades são: vetor representando a direção da palma da mão, vetor representando o normal da palma da mão, posição das pontas dos dedos e a posição do centro da palma da mão. Após a extração das funcionalidades será feita a classificação dos gestos através da utilização de um classificador HCNF. A estrutura deste sistema está ilustrada na figura 26.

Figura 26. Estrutura do sistema de reconhecimento de gestos de Lu, Tong & Chu (2016).

O estudo utilizou dois grupos de gestos para testar o sistema de reconhecimento implementado com o primeiro grupo tendo 12 gestos e servindo para comparar o sistema criado com outros sistemas semelhantes, o segundo grupo tem 10 gestos e serve para testar o reconhecimento do sistema quando lidando com gestos comuns. A precisão do reconhecimento dos gestos de ambos os grupos utilizando o sistema foi bom com uma precisão de 89,5% para o primeiro grupo e uma precisão de 95% para os gestos do segundo grupo.

Tendo por base a análise efetuada dos vários estudos, é possível concluir que no reconhecimento de gestos estáticos a utilização de SVMs com um kernel do tipo RBF como modelo de aprendizagem supervisionada é o modelo que mostra constantemente uma melhor performance num tempo aceitável.

A segunda conclusão é que a utilização de SVMs para o reconhecimento de gestos dinâmicos não parece ser eficiente. Isto deve-se ao facto de os estudos que trabalharam com o reconhecimento de gestos dinâmicos usam constantemente os métodos de aprendizagem HMM ou HCNF. Sendo também importante notar que os estudos que só trabalham com gestos estáticos dizem nas suas conclusões que os resultados que obtiveram não são aplicáveis para gestos dinâmicos.

A terceira conclusão foi que perante gestos dinâmicos o Leap Motion tem vários problemas envolvendo ruído nos dados obtidos pelo sensor durante a realização do movimento, especialmente na zona dos dedos. Um método para diminuir o ruído deve ser explorado.

A quarta conclusão foi que a exatidão do reconhecimento é a melhor forma de avaliar a classificação dos gestos, com a precisão do reconhecimento sendo também um importante fator. A exatidão permite concluir se o sistema de reconhecimento consegue realmente identificar corretamente os gestos. A precisão permite identificar se a classificação feita pelo sistema é estável. Se a classificação não for estável é muito provável que quando o utilizador estiver a fazer um gesto, a probabilidade de o sistema classificar corretamente o gesto não é alta.

2.5 Considerações Finais

Após a análise feita dos vários pontos apresentados no início deste capítulo, chegámos a várias conclusões.

A primeira foi que após analisar a taxionomia dos gestos feitos pelas mãos e braços, foi decidido que um modelo taxionómico, enquanto importante para perceber como um utilizador irá interpretar um gesto, não é o modelo mais apropriado para desenvolver o nosso sistema de reconhecimento de gestos. Isto deve-se ao facto de modelos taxionómicos exigirem que o gesto feito seja inserido nos vários contextos que o rodeiam, como por exemplo: a fala e outros movimentos corporais da pessoa que está a fazer o gesto e fatores externos ao utilizador, como a pessoa com que a qual está a interagir.

Para ter em conta todos estes fatores seria necessário aumentar o número de sensores utilizados para poder reconhecer outras partes do corpo humano. Seria também necessário implementar uma inteligência artificial extremamente avançada que conseguisse identificar o contexto em que o gesto está a ser feito e como o contexto afeta o gesto. Por esta razão será utilizado um modelo de gestos que ignora o contexto dos gestos.

A segunda conclusão foi que o Leap Motion é o sensor que será utilizado para fazer a recolha de gestos para treinar o sistema. Isto deve-se ao facto de o Leap Motion ser o sensor que mais se adequa ao âmbito deste trabalho e da curva de aprendizagem para desenvolver software para este sensor ser muito menor do que para os outros sensores.

A terceira conclusão foi que o SVM é o melhor classificador para fazer o treino de gestos estáticos do sistema de reconhecimento de gestos. Esta escolha deve-se principalmente à versatilidade do método.

Foi decidido que o reconhecimento de gestos dinâmicos não seria implementado. Isto foi uma decisão dos orientadores do projeto, que consideraram a implementação do reconhecimento dos gestos dinâmicos exigiria demasiado tempo devido aos métodos de aprendizagem necessários para este tipo de reconhecimento serem muito complexos.

3 TRABALHO DESENVOLVIDO

Neste capítulo será feita uma descrição dos vários requisitos funcionais e não funcionais que a aplicação final irá implementar. Também é feita uma descrição geral da arquitetura da aplicação e uma pequena descrição das tecnologias utilizadas.

Por último, este capítulo irá fazer uma descrição em detalhe de como o sistema de reconhecimento de gestos foi criado. Isto incluíra uma explicação do método utilizado para obter uma boa estimativa da exatidão do sistema de reconhecimento e do método utilizado para escolher de entre todo os parâmetros, quais os verdadeiramente necessários para obter uma boa exatidão. Também será analisado como alterações aos parâmetros do método de aprendizagem afetam a exatidão e precisão do sistema treinado e como é que o sistema treinado reage a um novo exemplo.

3.1 REQUISITOS DA APLICAÇÃO

3.1.1 Requisitos Funcionais

Os requisitos funcionais do sistema de reconhecimento de gestos utilizando o Leap Motion serão:

1. **Poder gravar gestos em ficheiros**: Um utilizador poderá indicar que quer gravar um gesto e após o gesto ter sido feito será guardado num ficheiro com o nome que o utilizador escolheu.
2. **Poder visualizar os gestos gravados**: O utilizador poderá indicar que quer visualizar um gesto gravado anteriormente num ficheiro.
3. **Poder atribuir certos gestos a teclas do teclado**: O utilizador poderá indicar que quer associar um gesto que foi gravado anteriormente a uma tecla do teclado.
4. **Poder gravar todas as associações entre teclas do teclado e gestos num ficheiro**: Após o utilizador ter associado todas as teclas que quer usar a um gesto gravado, ele poderá gravar todas as associações num ficheiro que será utilizado para treinar o modelo de reconhecimento.
5. **Identificar o gesto que um utilizador fez**: O sistema, usando um modelo de reconhecimento treinado, deve ser capaz de corretamente identificar o gesto que o utilizador fez.
6. **Treinar um modelo de reconhecimento**: O utilizador poderá treinar um modelo de reconhecimento usando um dos ficheiros que contem associações entre teclas do teclado e gestos. O modelo treinado será guardado num ficheiro á parte.
7. **Carregar um modelo de reconhecimento já treinado**: O utilizador poderá carregar um modelo de reconhecimento que foi treinado anteriormente e guardado num ficheiro á parte.

3.1.2 Requisitos Não Funcionais

Os requisitos não funcionais do sistema de reconhecimento de gestos utilizando o Leap Motion e a aplicação criada utilizando o sistema serão:

1. **Exatidão**: O sistema deverá ser capaz de identificar corretamente 80% dos gestos feitos. Este valor foi escolhido pelo facto de através da análise dos vários artigos que foram lidos é possível inferir que esta é, geralmente, a exatidão necessária para um sistema de reconhecimento de gestos ser considerado robusto e bem implementado.
2. **Precisão**: O sistema deverá ser capaz de identificar o gesto feito com uma certeza de pelo menos 75%. Este valor foi escolhido devido a considerarmos que o modelo de reconhecimento deveria ser consistente nos seus resultados, com 75% sendo o valor em que a previsão do modelo é realmente sólida.
3. **Adaptabilidade**: O tempo entre um gesto ser recolhido pelo sensor e a próxima recolha poderá ser modificado para se adequar ao tempo que um utilizador demora a alterar o gesto que a sua mão está a fazer.

3.2 ARQUITETURA

3.2.1 Módulos

A estrutura da aplicação de reconhecimento e implementação de teclados gestuais utilizando o Leap Motion contém os seguintes módulos:

- Módulo A: Uma aplicação feita em Unity, que irá conectar uma tecla do teclado do computador a um ficheiro que contêm a informação acerca de um gesto. Este processo pode ser replicado para todas as teclas do teclado. Não é necessária que todas as teclas estejam associadas a um gesto. Após os utilizadores da aplicação terem associado os ficheiros a todas as teclas que querem utilizar poderão criar um ficheiro que terá a informação acerca de todas os gestos conectados a letras.
    - Módulo A1: Serve para guardar num ficheiro os parâmetros associados à execução de um gesto. Os ficheiros criados pelo gravador poderão ser atribuídos a uma tecla do teclado normal de um computador.
    - Módulo A2: permite ver graficamente um gesto guardado num ficheiro. Servirá para verificar se um gesto guardado num ficheiro foi bem guardado.
- Módulo B: Uma biblioteca que ao receber um ficheiro com um teclado gestual irá aprender os gestos contidos no ficheiro e manterá as relações dos gestos às teclas que estes gestos estão associados. Através dos métodos públicos disponibilizados pela extensão, quando um utilizador fizer um gesto no Leap Motion que está associado a uma tecla do teclado, a aplicação irá reagir ao gesto da mesma maneira que reagiria se a tecla fosse premida. A linguagens com que esta biblioteca foi feita foram o Java e o C#.
    - Módulo B1: Um conjunto de métodos que ao receber um ficheiro com um teclado gestual irá treinar a aplicação para reconhecer todos os gestos contidos no ficheiro. A aprendizagem dos gestos será feita através do algoritmo de aprendizagem SVM.

3.2.2 Dados

Os dados usados pela aplicação de reconhecimento e implementação de teclados gestuais utilizando o Leap Motion são os seguintes:

- Dados A: Um ficheiro que tem os parâmetros que as mãos e braços têm durante a realização de um gesto. Estes ficheiros são criados pelo módulo A1 e são usados para criar o teclado gestual descrito nos dados B.
- Dados B: Um ficheiro que contem a informação acerca de vários gestos, com cada gesto estando associado a uma tecla do teclado. Estes dados são criados pelo módulo A. Estes dados são usados pela biblioteca descrita no módulo B da aplicação.
- Dados C: Um ficheiro que contem os dados de um modelo de reconhecimento de gestos já treinado. Este modelo de reconhecimento será criado e usado pelo módulo B da aplicação.

Esta estrutura do sistema de reconhecimento e implementação de teclados gestuais usando o Leap Motion está representada na figura 27.

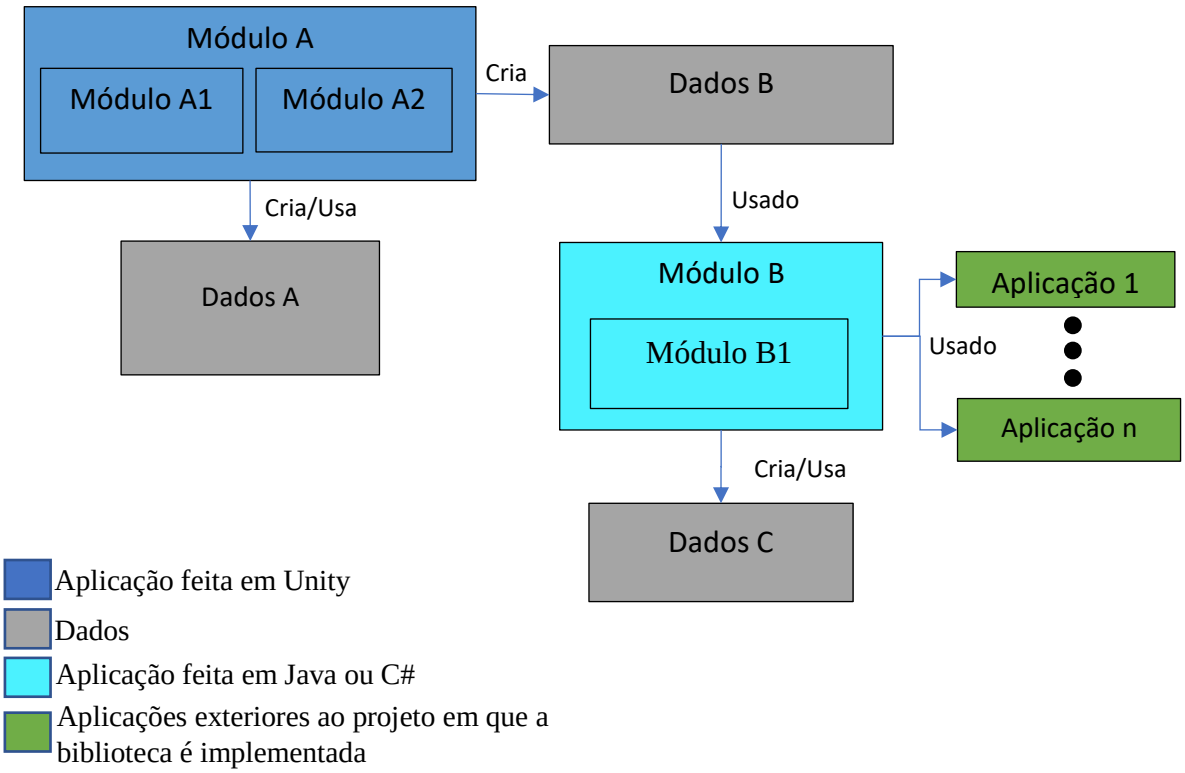


Figura 27. Estrutura do implementador de teclados gestuais.

3.3 Tecnologias e Ferramentas Utilizadas

3.3.1 Unity

O Unity é um game engine, um programa usado para simplificar o desenvolvimento de jogos. Também, apesar de ser uma aplicação para criar jogos, permite a criação rápida e simples de outro tipo de aplicações devido a várias funcionalidades disponibilizadas pela aplicação. O Unity permite programadores serem capazes de desenvolver para

várias plataformas incluindo consolas, dispositivos móveis e computadores. O Unity é também capaz de produzir jogos em duas dimensões e três dimensões. Na figura 28 é apresentado o logotipo atual do Unity.

Figura 28. Logotipo do Unity[8].

O Unity foi principalmente utilizado para criar o módulo A. A razão para o Unity ter sido utilizado foi devido ao facto de o Leap Motion ter disponível software que permite a sua implementação rápida no Unity, incluindo a opção de mostrar o modelo das mãos criadas pelo Leap Motion.

3.3.2 Visual Studio

O Visual Studio é um IDE criado pela Microsoft preparado especialmente para trabalhar com várias linguagens e certos frameworks, especialmente o framework .NET e as linguagens C# e Visual Basic. Na figura 29 é apresentado o logotipo actual do Visual Studio.

Figura 29. Logotipo do Visual Studio[9].

O Visual Studio foi utilizado devido ao facto de o Unity preferir que os scripts utilizados sejam escritos C#, o que faz que o Visual Studio tenha de ser usado já que ele é um IDE especialmente preparado para trabalhar com a linguagem C#. Versões mais recentes do Visual Studio também estão preparadas para trabalhar em conjunto com o Unity. O Visual Studio foi também utilizado para criar uma versão em C# do módulo B.

3.3.3 NetBeans

O NetBeans é um IDE especialmente desenhado para criar aplicações em Java, tendo também extensões que permitem criar aplicações em outras linguagens. A figura 30 mostra o logotipo atual do NetBeans.

[8] https://en.wikipedia.org/wiki/Unity_(game_engine)#/media/File:Unity_Technologies_logo.svg
[9]https://en.wikipedia.org/wiki/Microsoft_Visual_Studio#/media/File:Visual_Studio_2017_logo_and_wordmark.svg

Figura 30. Logotipo do NetBeans[10].

O NetBeans foi usado para criar a versão em Java do sistema de reconhecimento de gestos e o implementador de teclados gestuais. O NetBeans foi escolhido devido a ele ter várias funcionalidades que permitirão simplificar e acelerar o processo de desenvolvimento e ter uma interface mais fácil de trabalhar do que outros IDEs.

3.3.4 LIBSVM

LIBSVM é uma biblioteca open source, desenvolvida pela Universidade Nacional de Taiwan, que disponibiliza vários métodos para executar o treino de um modelo SVM e prever a que classe um novo exemplo irá pertencer utilizando um modelo treinado.

O LIBSVM contém cinco classes:

- svm: Classe que contém todos os métodos necessários para executar o algoritmo do SVM. Os métodos considerados como os mais importantes desta classe são:
  - svm_model svm_train(svm_problem s, svm_parameter s1)
  - double svm_predict(svm_model s, svm_node[] svm_nds)
  - double svm_predict_probability(svm_model s, svm_node[] svm_nds, double[] doubles)
  - void svm_save_model(String string, svm_model s)
  - svm_model svm_load_model(String string)
- svm_model: Usada como variável. Contém o modelo treinado pelo SVM e será utilizado para prever em que classe estará um novo caso.
- svm_node: Usada como variável. Contém os dados de um exemplo do sistema que será classificado.
- svm_problem: Usada como variável. Contém todas as amostras que serão usadas para treinar o modelo. É composto por vários svm_node.
- svm_parameter: Usada como variável. Variável onde serão definidos vários parâmetros do método SVM a ser executado, como tipo de SVM, kernel, custo, etc.

[10] https://en.wikipedia.org/wiki/NetBeans#/media/File:Apache_NetBeans_Logo.svg

3.4 IMPLEMENTAÇÃO

Após ter sido feita uma análise de vários artigos e várias bibliotecas que permitam implementar algoritmos de aprendizagem foi concluído que a utilização de SVMs é o melhor algoritmo de aprendizagem para a criação de um modelo de reconhecimento de gestos estáticos. Isto deve-se ao facto de a maioria dos estudos sobre reconhecimento de gestos estáticos terem usado SVMs e os seus resultados terem sidos bons e pela existência da biblioteca open source LIBSVM que possui métodos que permitem a implementação de algoritmos de aprendizagem baseados em SVM.

Foi decidido que para verificar se um modelo treinado utilizando o algoritmo SVM é realmente capaz de identificar corretamente um gesto, devia ser feita uma análise da exatidão e precisão de um modelo treinado pelo algoritmo SVM. Nessa análise foi feita a recolha de vários gestos que foram utilizados para treinar um modelo SVM. É esperado que esta análise também permita descobrir quais são os parâmetros do Leap Motion que permitirão identificar um gesto feito e quais são os parâmetros do SVM que afetaram a precisão do reconhecimento do modelo e quais são os valores ótimos.

3.4.1 Aquisição e Armazenamento de Informação

3.4.1.1 Gestos de Treino

Para treinar e testar a precisão do sistema de reconhecimento de gestos, vários gestos estáticos serão recolhidos com apoio de vários voluntários. Estes testes foram feitos por 10 pessoas, com cada gesto sendo repetido 30 vezes por cada pessoa. Os gestos foram recolhidos por várias pessoas diferentes para permitir verificar quais os efeitos das características individuais de cada voluntário no modelo de reconhecimento. Foi pedido que os voluntários fizessem os gestos várias vezes para permitir que quando o modelo fosse treinado ele tivesse exemplos de treino suficientes para obter resultados robustos.

Existe um conjunto de 10 gestos estáticos. Os gestos estáticos também terão 3 diferentes posições dos braços, com cada pessoa fazendo o mesmo gesto 10 vezes para cada posição. No final da recolha deveremos ter exatamente 3000 gestos recolhidos, com 300 gestos por pessoa. As posições diferentes dos braços serviram para verificar se o sensor Leap Motion tem dificuldades a reconhecer o mesmo gesto quando ele está num ângulo diferente.

Os gestos estáticos que são analisados durante os testes do sistema são:

1. A mão fechada.
2. A mão aberta com todos os dedos esticados.
3. A mão fechada com polegar esticado.
4. A mão fechada com o indicador esticado.
5. A mão fechada com os dedos do polegar e indicador esticados e o polegar a um ângulo de 90º do indicador.
6. A mão fechada com os dedos do indicador, anelar e mínimo abertos.
7. A mão aberta com a pontas do polegar e indicador em contacto.
8. A mão aberta com a pontas do polegar e médio em contacto.

9. A mão com o indicador em cima do polegar e o médio em cima do indicador e com os restantes dedos fechados.
10. A mão aberta com todos os dedos esticados com elevações diferentes, com o polegar sendo o dedo com menos elevação e o mínimo o dedo com maior, os restantes dedos devem seguir uma ordem crescente do menor para o maior.

Os gestos 1 e 2 são os gestos mais simples que as mãos podem fazer, e é esperado que estes tenham uma enorme probabilidade de ser reconhecidos inicialmente.

Os gestos 3, 4 e 5 servem para testar a precisão do reconhecimento do sistema em casos em que os gestos são simples e um bocado semelhantes. Sendo que caso o sistema seja mal implementado ou não receba bons exemplos, este irá provavelmente confundir o gesto 3 com o gesto 5 ou confundir o gesto 4 com o gesto 5.

O gesto 6 serve para testar a precisão do reconhecimento de gestos em que vários dedos estão levantados, mas em que os dedos que estão levantados não são todos adjacentes.

Os gestos 7 ,8 e 9 servem para testar a precisão do reconhecimento em gestos em que existe contacto entre a pontas dos dedos. Também existe uma grande probabilidade que o sistema confunda o gesto 6 e 8, devido a estes serem extremamente semelhantes.

O gesto 10 testa a precisão do sistema caso os dedos tiverem diferentes elevações. Estes gestos estão ilustrados na figura 31.

Figura 31. Gestos Estáticos.

Todos os gestos estáticos serão também testados com diferentes ângulos do ombro. Isto permitirá analisar se o ângulo da mão afeta a precisão da classificação do sistema, e se sim, que ângulos da mão permitem maximizar a classificação correta dos gestos e que posições causam uma grande diminuição da precisão da classificação. Os ângulos que são analisados nos testes do sistema são:

1. O ângulo do braço a 0^o (palma da mão virada para baixo).
2. O ângulo do braço a 90^o (palma da mão do braço direito virada para a esquerda).
3. O ângulo do braço a 180^o (palma da mão virada para cima).

Estes ângulos do braço são demonstrados na tabela 3.

Ângulos dos Braços		
0º	90º	180º
		

Tabela 3. Ângulos dos braços durante os testes.

3.4.1.2 Setup de Recolha de Amostras

Após os gestos de treino terem sido definidos, foi começada a fazer a sua recolha. Durante a recolha destes gestos, um guião para recolher os gestos foi definido.

Para permitir uma recolha mais eficiente das amostras de treino e teste, foi definido um guião para executar a recolha de amostras. Em primeiro lugar seriam convidados vários indivíduos para ajudar na recolha das amostras. Após um indivíduo ter concordado em ajudar na recolha de amostras seria marcada uma reunião com o individuo onde seria recolhida as amostras de treino. O equipamento utilizado para recolher as amostras foi um computador, com uma aplicação criada para recolher as amostras, e o Leap Motion conectado a esse computador. O individuo estará em frente do Leap Motion e faria vários gestos instruídos pela pessoa a cargo de recolher as amostras, enquanto esta pessoa estaria no computador e irá executar a aplicação para criar as amostras de treino quando o individuo fizer o gesto.

A recolha começou com os gestos estáticos com o ângulo do braço começando a 0º. O processo foi repetido para cada um dos ângulos do braço referidos no subcapítulo acerca dos gestos de treino.

A recolha das amostras foi feita utilizando uma aplicação criada em Unity. Esta aplicação Unity para além de servir para recolher as amostras, também foi desenhada para permitir uma simplificação do processo de organização das amostras de teste e treino recolhidas. Esta simplificação da organização permitiu que as amostras recolhidas fossem facilmente geridas e que fosse fácil saber que voluntário tinha feito cada gesto gravado.

As amostras de treino recolhidas são organizadas em várias subpastas. Isto permitiu que o manuseamento das amostras obtidas fosse mais simples e fácil de entender. A organização baseia-se na existência de uma pasta específica para cada indivíduo e que dentro dessa pasta existam três pastas para cada uma das três possíveis posições da palma da mão que são consideradas neste projeto. O interior da cada uma destas pastas tem dez pastas referentes a cada um dos gestos que considerámos para a análise da eficiência do modelo SVM, e estas pastas possuem as amostras recolhidas de cada gesto. A estrutura desta organização automática das amostras de treino recolhidas é ilustrada na figura 32.

Figura 32. Organização das amostras de treino.

3.4.1.3 Parâmetros Recolhidos

É necessário, para que o algoritmo SVM treine um modelo que permita avaliar e classificar um gesto, que o SVM receba vários exemplos de gestos com vários parâmetros diferentes, que permitam o algoritmo encontrar semelhanças e diferenças entre os vários gestos. Os seguintes parâmetros de um gesto feito pelo Leap Motion são recolhidos e guardados quando feita uma recolha:

- Timestamp do momento em que o gesto é recolhido.
- Número de mãos na janela.
- Indicador de se a mão é esquerda ou direita.
- Posição da palma da mão no espaço.
- Posição estabilizada da palma da mão no espaço.
- Vetor normal da palma da mão (Fig.33).
- Vetor direcional da palma da mão (Fig.33).
- Comprimento e largura dos dedos.
- Posição estabilizada das pontas dos dedos.
- Vetor da direção das pontas dos dedos.

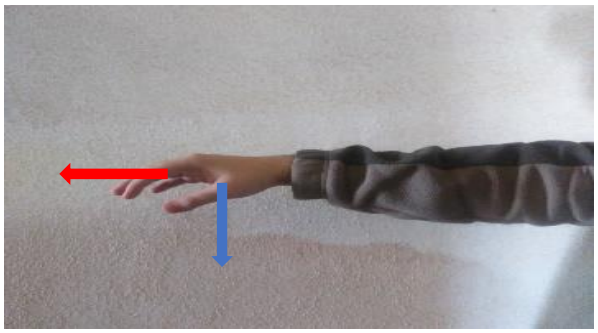


Figura 33. Vetor normal da palma da mão (seta azul) e vetor direcional da palma da mão (seta vermelha).

A seleção inicial de parâmetros teve como principal objetivo maximizar o número de parâmetros escolhidos, com qualquer parâmetro das mãos disponível que nos parecesse útil sendo guardado no ficheiro. Esta maximização da seleção dos parâmetros foi feita para impedir que um parâmetro que possa ser útil para aumentar a exatidão e precisão do modelo de reconhecimento de gestos não fosse ignorado.

Na figura 34 é apresentado o formato com que os valores recolhidos de um gesto são guardados num ficheiro. O valor do timestamp, do número de mãos na janela e a mão que está a fazer o gesto são respetivamente os três primeiros valores que estão escritos em vermelho claro. Os valores da posição e posição estabilizada da palma da mão no espaço são os valores com cor verde. Os valores do vetor normal e direcional da palma da mão são os valores escritos em azul claro.

Após os valores do vetor direcional aparecem os valores dos parâmetros dos vários dedos serão apresentados. Aparecem primeiro todos os parâmetros de um dedo e só depois é que são apresentados os parâmetros do dedo seguinte. Os valores dos dedos são apresentados pela seguinte ordem: polegar, indicador, médio, anelar e mínimo. O comprimento e largura dos dedos são os valores com cor roxa. Os valores da posição estabilizada das pontas dos dedos são os valores escritos a cor de laranja. Por último os valores do vetor da direção das pontas dos dedos são os valores escritos a vermelho escuro.

83750445625;1;L; -37.17561;324.0298;-75.65726;-37.00385;322.1713;-76.40042;0.1207001;-0.8907713;-0.4381301;0.1222436;0.4513351;-0.8839418;99.86689;20.84575;-10.05866;292.8159;-114.3036;-0.1226759;-0.1042021;-0.9869612;78.52512;19.91186;-10.15187;288.2472;-79.56343;-0.09886921;-0.5506722;0.8288456;89.1054;19.5561;-25.90498;284.1508;-74.30215;-0.06014686;-0.4817291;0.8742536;84.86235;18.60887;-39.19163;285.3808;-74.21992;0.1050389;-0.4725963;0.8749969;66.69249;16.52985;-51.71635;291.5535;-77.10708;0.4001298;-0.4160411;0.8165819;

Figura 34. Formato dos valores recolhidos de um gesto.

3.4.2 Avaliação do Reconhecimento

Após a obtenção dos dados dos gestos, definidos no subcapítulo dos gestos de treino, foi necessário processar esses dados para permitir que quando o modelo fosse treinado, seja possível maximizar a exatidão e precisão do modelo.

O treino do modelo para avaliar a exatidão e precisão do sistema de reconhecimento foi dividido em três fases, cada uma destas fases serviu para testar a exatidão e precisão do modelo de aprendizagem SVM em diferentes casos.

A primeira fase consistiu em considerar apenas a mão fechada como um gesto correto e os restantes gestos como incorretos. Isto permitirá testar se o modelo tem uma boa performance ao analisar o caso mais básico possível.

Na segunda fase foi analisado se o modelo consegue prever corretamente a direção da palma da mão. Esta fase é mais complexa do que a primeira e permitiu verificar se o modelo consegue distinguir corretamente certas grandes características dos gestos antes de se começar a analisar características mais subtis.

Na terceira fase foi verificado se o modelo consegue identificar as diferenças entre os gestos de treino apresentados no início deste capítulo e os identificar corretamente.

Para criar o sistema de reconhecimento de gestos, usando o método de aprendizagem SVM, foi necessário a utilização de uma biblioteca que tenha o algoritmo do método de aprendizagem implementado. Neste trabalho foi utilizada a biblioteca open source LIBSVM. Nesta biblioteca está implementada uma versão robusta do método de aprendizagem SVM.

Os resultados dos testes apresentados no resto deste capítulo são demonstrados no capítulo 4.

3.4.2.1 Otimização dos Parâmetros do Gesto

Certos parâmetros poderiam ser removidos se for considerado que esses parâmetros diminuem a exatidão e precisão do reconhecimento dos gestos, devido ao algoritmo dar demasiada importância a esses parâmetros ou por eles não terem nenhuma influência na precisão do modelo.

O primeiro passo do processo de otimização dos parâmetros dos gestos utilizados para treinar o modelo foi uma análise manual dos valores dos parâmetros recolhidos. O objetivo desta análise foi o de encontrar parâmetros que não afetariam o resultado do modelo de reconhecimento devido aos valores desse parâmetro serem demasiado estáticos. Deveriam também ser removidos parâmetros que afetariam negativamente o modelo, devido aos seus valores terem uma variação demasiado grande e que foram identificados como supérfluos para obter uma boa exatidão e precisão do modelo de reconhecimento.

O segundo passo consistiu em dividir os parâmetros dos gestos em diferentes grupos baseados na sua natureza. Após esta divisão dos parâmetros em diferentes grupos, foram treinados vários modelos. Os diferentes modelos foram treinados utilizando respetivamente apenas um dos grupos criados, com exceção de um modelo que foi treinado utilizando os parâmetros de todos os modelos.

Os modelos treinados foram testados, com os exemplos usados para testar sendo iguais para todos os modelos, e foram analisados os resultados em termos da sua exatidão e precisão. Se os resultados de um modelo treinado utilizando apenas um grupo fossem claramente melhores do que os resultados dos testes de todos os outros modelos, os parâmetros desse grupo seriam mantidos e os parâmetros dos outros grupos seriam removidos. Se os resultados dos testes fossem similares para todos os grupos, ou se os resultados do modelo treinado utilizando todos os parâmetros fossem claramente melhores do que o outro, os parâmetros dos grupos seriam iterativamente trocados,

adicionados ou removidos para tentar alterar o resultado do modelo até um grupo ter um resultado claramente superior.

3.4.2.2 Otimização dos Parâmetros do SVM

Outra importante parte do treino do modelo foi a seleção dos parâmetros do modelo SVM que vão influenciar o treino do modelo, nomeadamente o kernel utilizado.

O tipo de kernel utilizado foi alterado para tentar identificar a sua influência no resultado do modelo. Se fosse detetado que a mudança do tipo de kernel do modelo SVM afetava positivamente o resultado do treino do modelo este seria mantido, sendo que caso ele afetasse negativamente o modelo seria removido.

3.4.2.3 Otimização do Número de Dados de Treino

Neste teste foi analisado como o número de exemplos dados para treinar o modelo afetava a exatidão e precisão do reconhecimento do modelo treinado.

Para fazer este teste foi dado ao algoritmo SVM um diferente número de dados de treino baseado no número de gestos associados a cada indivíduo que colaborou na recolha de gestos. Foi testada a exatidão e a precisão dos modelos treinados utilizando os gestos associados a um indivíduo, dois indivíduos, cinco indivíduos e dez indivíduos, ou seja, modelos treinados utilizando respetivamente 300 gestos, 600 gestos, 1500 gestos e 3000 gestos.

Era esperado que á medida que o número de exemplos de treino aumentava, a exatidão e precisão dos resultados do modelo aumentariam. Se o número de exemplos dados para treinar passasse um certo limite, era também esperado que a precisão dos resultados do modelo passasse a ter um crescimento logarítmico.

3.4.3 Utilização dos Gestos

3.4.3.1 Criação Automática de Gestos de Treino

Devido ao facto de não ser conveniente, para um utilizador, criar centenas de exemplos do mesmo gesto, para treinar o modelo SVM, foi necessário desenvolver uma maneira de criar automaticamente vários exemplos de treino utilizando apenas um exemplo.

A maneira mais simples de fazer isso seria fazer *n* cópias do gesto dado e utilizar essas cópias para fazer o treino do modelo. Esse método funcionaria, mas iria fazer que o modelo tivesse um grave caso de overfitting, podendo causar que o modelo não conseguisse reconhecer corretamente o gesto se ele for ligeiramente diferente do gesto dado inicialmente.

Devido a este problema de possível overfitting, foi decidido desenvolver um método que permite, dado um exemplo inicial de um gesto, criar vários gestos semelhantes ao original, mas com várias pequenas diferenças em vários parâmetros, que não irão realmente afetar o gesto que está a ser feito, mas que permitirá evitar o overfitting do modelo.

Na figura 35 é demostrado os efeitos negativos, no modelo, de usar exemplos demasiado semelhantes. No modelo da esquerda, os exemplos de todas as categorias têm uma boa dispersão, isto evitou que os exemplos mais extremos ditassem a área das categorias. No modelo da direita, para a categoria roxa, foram apenas usados exemplos extremamente semelhantes e como tal a área que o modelo roxo ocupa é muito menor do que a área ocupada no modelo da esquerda, sendo o limite entre categorias ditado pelos exemplos mais extremos da categoria azul e amarela.

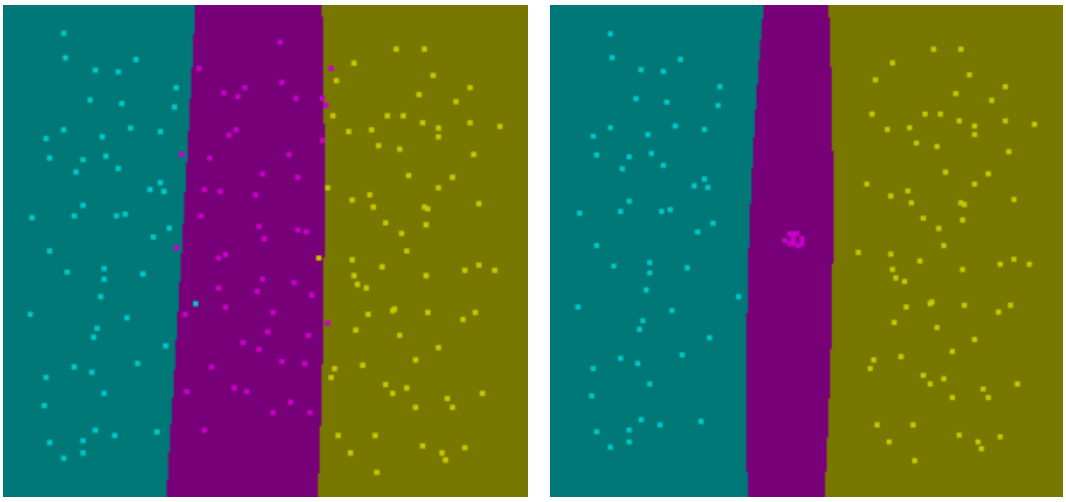


Figura 35. Comparação de um modelo sem overfitting(esquerda) e um modelo com overfitting(direita).

Um dos requisitos mais importantes para criar gestos artificiais foi o de ter o cuidado de alterar certos parâmetros do gesto, sem, ao mesmo tempo, alterar fundamentalmente o gesto que está a ser feito. Para poder fazer isto é necessário entender os parâmetros direcionais do Leap Motion.

Um parâmetro direcional é um vetor que contém três componentes:

- X: Indica se o vetor está direcionado para a esquerda ou para a direita. Sendo que caso o valor deste componente seja -1 o vetor está complemente voltado para a esquerda, caso seja 1 está complemente voltado para a direita. No caso em que o valor é 0, o vetor não está voltado para nenhuma direção.
- Y: Indica se o vetor está direcionado para cima ou para baixo. Sendo que caso o valor deste componente seja -1 o vetor está complemente voltado para baixo, caso seja 1 está complemente voltado para cima. No caso em que o valor é 0, o vetor não está voltado para nenhuma direção.
- Z: Indica se o vetor está direcionado para a frente ou para trás. Sendo que caso o valor deste componente seja -1 o vetor está complemente voltado para a frente, caso seja 1 o vetor está complemente voltado para trás. No caso em que o valor é 0, o vetor não está voltado para nenhuma destas direções.

Nas figuras 36, 37 e 38 encontram-se representados respetivamente os valores que os componentes X e Y dos parâmetros direcionais podem ter baseados na sua posição no plano XY, os valores que os componentes X e Z podem ter no plano XZ e os valores que os parâmetros Y e Z podem ter no plano YZ.

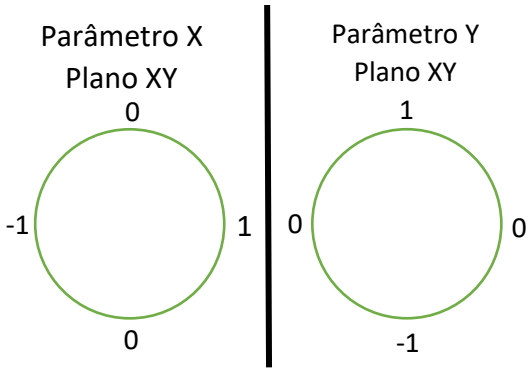


Figura 36. Valores dos componentes X e Y dos parâmetros direcionais no plano XY.

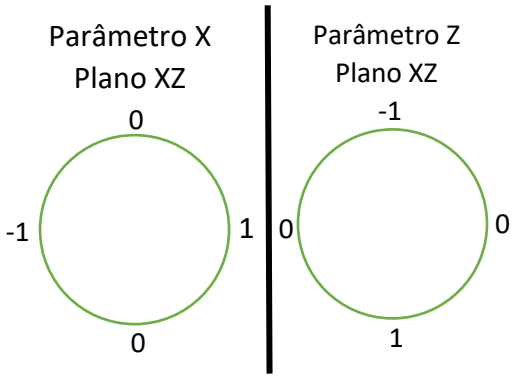


Figura 37. Valores dos componentes X e Z dos parâmetros direcionais no plano XZ.

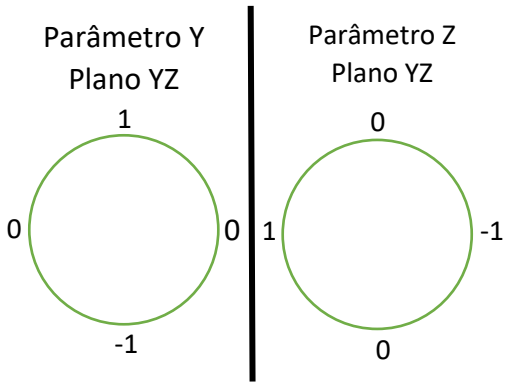


Figura 38. Valores dos componentes Y e Z dos parâmetros direcionais no plano YZ.

Durante a análise dos componentes dos parâmetros direcionais foi detetado que, mesmo quando tentando ter a mão complemente parada, acontecem constantemente pequenas alterações nos componentes dos parâmetros direcionais. Foi feita uma análise das alterações nos componentes dos parâmetros e foi concluído que é possível manter a natureza do gesto, enquanto, ao mesmo tempo, alterando os valores dos componentes dos parâmetros direcionais.

Esta análise baseou-se em treinar um modelo, usando apenas um exemplo para cada gesto com os restantes exemplos desses casos sendo gestos artificiais criados automaticamente pelo programa baseados nos exemplos originais.

Também foram adicionados vários gestos aleatórios diferentes dos gestos principais. Estes gestos aleatórios foram juntados num novo caso chamado "Outros Gestos". Foram depois escolhidos oito gestos com certas parecenças aos gestos principais, mas também com certas diferenças. Era esperado que quando estes gestos fossem classificados pelo modelo, o caso a que os gestos de teste seriam assignados seria os "Outros Gestos". Se o modelo classificasse um gesto de teste como sendo um gesto principal, isto seria considerado um erro de classificação. Na tabela 4 podem ser vistos os resultados desta análise.

	Limites das mudanças dos valores dos gestos originais							
	0	0,01	0,05	0,1	0,2	0,3	0,4	0,5
Exatidão	87,5%	87,5%	87,5%	87,5%	75%	75%	50%	25%

Tabela 4. Resultados da análise da criação de gestos

Foi concluído que o limite máximo de variação, dos valores dos gestos originais, em que o modelo treinado consegue manter a sua exatidão e os gestos preservam a sua natureza é entre -0,1 e 0,1.

Foram também criadas versões do gesto em que ele é rodado 90º, 180º e 270º. Esta rotação servirá o possível propósito de criar gestos que servirão como claros opostos do gesto original, permitindo treinar o modelo usando apenas um gesto. É também esperado que ao utilizar este método a versatilidade do modelo aumente.

O número de exemplos criados usando estas rotações para cada uma das três rotações é o número de exemplos criados do gesto original a dividir por quatro. Isto impedirá que o número de gestos modificados que foram criados seja superior ao número de exemplos semelhantes ao gesto original. Se o número de exemplos de treino dos gestos rodados fosse superior ao número de exemplos do gesto original, existia o risco de o modelo dar demasiada importância aos gestos rodados e diminuir a área em que o modelo reconhece o gesto original.

Na figura 39 está representado uma versão simplificada do processo de criação de gestos artificiais.

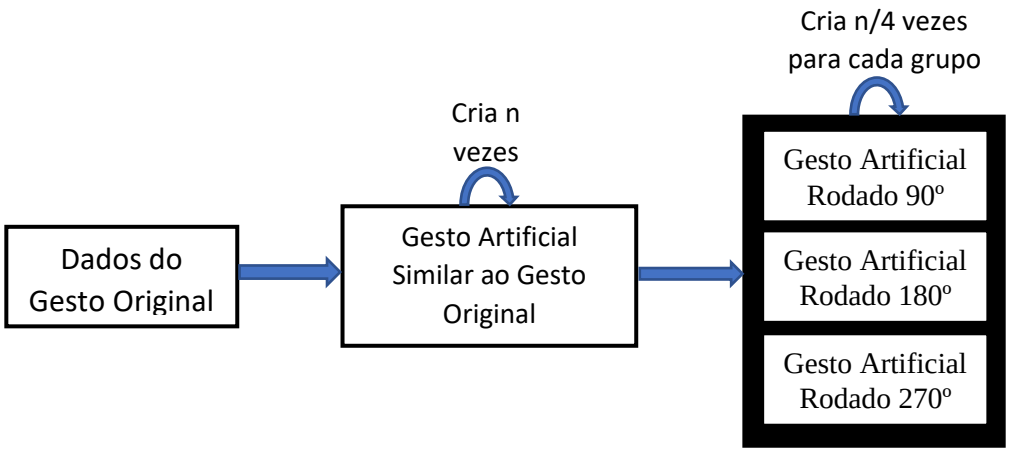


Figura 39. Processo de criação de gestos artificiais.

3.4.3.2 Testes dos Gestos Automáticos

Para testar a exatidão e precisão do modelo treinado, utilizando os gestos automáticos, e para provar que a performance destes modelos é melhor ou igual aos modelos criados utilizando apenas gestos manuais, foi necessário fazer vários tipos de testes.

O gesto que queremos durante todos estes testes é a mão fechada com a palma da mão virada para baixo. Para permitir o modelo fazer a distinção entre o gesto dado e os restantes gestos, também foi dado ao modelo gestos que não eram os que queríamos reconhecer.

Com estes testes também quisemos descobrir o ponto em que o modelo treinado deixa de reconhecer o gesto e se esse ponto coincide com as nossas expetativas. Estas expetativas são que quando qualquer um dos componentes dos vetores da palma da mão tem um valor 0,5 acima ou abaixo do valor original, esse gesto não será reconhecido como o gesto original. O valor 0,5 foi escolhido tendo em conta o tipo de gestos que seriam utilizados para treinar o modelo neste teste. Com base nesses gestos e nas nossas expetativas de como o modelo deveria funcionar este valor foi escolhido.

No primeiro grupo de testes foi analisada a exatidão e precisão do reconhecimento do modelo treinado utilizando gestos automáticos, através do teste deste modelo utilizando gestos recolhidos manualmente.

No segundo grupo de testes também foi analisada a exatidão e precisão do reconhecimento do modelo treinado usando gestos automáticos, mas desta vez usando gestos recolhidos em tempo real pelo Leap Motion para fazer os testes. Nestes testes também foram analisados os pontos em que o gesto deixa de ser reconhecido e se esses pontos coincidem com as nossas expetativas.

O terceiro grupo de testes consistiu em analisar a exatidão e precisão do modelo treinado utilizando gestos automáticos, sendo os exemplos usados para teste outros gestos criados automaticamente. Foram feitos vários testes deste tipo, alterando o número de gestos criados automaticamente e alterando a percentagem de gestos que são utilizados para treinar o modelo e o número de gestos que são utilizados para testar o modelo.

Foi também testado o oposto dos vários treinos apresentados anteriormente. Ou seja, foi feita a análise da exatidão e precisão do reconhecimento do modelo treinado utilizando gestos manuais, através do teste deste modelo utilizando gestos recolhidos automaticamente, através do reconhecimento em tempo real utilizando o Leap Motion e pela utilização de parte dos gestos manuais para testar o modelo.

4 RESULTADOS

Neste capítulo serão apresentados os resultados dos vários treinos feitos para otimizar o modelo de reconhecimento de gestos e para verificar a exatidão e precisão dos modelos criados. Também será apresentada em detalhe as várias partes da aplicação final criada.

4.1 RESULTADOS DO TREINO

Neste subcapítulo são apresentados os resultados dos testes do sistema apresentados no capítulo anterior, com o objetivo de otimizar os parâmetros que o modelo utiliza e recebe, como também testar a eficiência dos gestos criados automaticamente comparado com os gestos criados completamente manualmente. Os principais fatores que interessam para avaliar os resultados dos testes são a exatidão e a precisão dos resultados dos testes. As fórmulas (1), (2) e (3) foram criadas para poder analisar estes fatores.

A exatidão dos testes é a média das exatidões de t testes, em que a exatidão de cada teste é a percentagem de exemplos que foram corretamente classificados pelo modelo. A exatidão dos resultados de um teste será representada pelas fórmulas (1) e (2), em que E representa a exatidão média de todos os testes realizados, L_i representado a exatidão de cada um dos exemplos usados para testar, t representado o número de testes feitos, n_i representando o número de exemplos num teste e com p_i representando o número de exemplos em n_i que foram classificados corretamente.

(1) $E = \frac{\sum_{i=1}^{t} L_i}{t}$

(2) $L_i = \frac{p_i}{n_i}$

A precisão será inferida através da análise da variação das certezas e a média das certezas dos resultados das previsões do modelo. A média das certezas dos resultados de um teste é representada pela fórmula (3), com C representado a média das certezas, n o número de exemplos dados para testar e c_i a certeza do resultado para cada exemplo usado.

(3) $C = \frac{\sum_{i=1}^{n} c_i}{n}$

4.1.1 Resultados da Otimização do Parâmetros dos Gestos

Após uma análise manual mais detalhada dos parâmetros recolhidos, foi decidido que o parâmetro timestamp devia ser imediatamente removido.

Esta remoção deve-se ao facto de o parâmetro timestamp não ter realmente nenhuma relação com o gesto feito e por os valores deste parâmetro serem todos diferentes. A grande diversidade de valores para este parâmetro fez com que o algoritmo SVM dê demasiado atenção a este parâmetro o que prejudica a exatidão e precisão do modelo treinado. Na figura 40 são demonstrados alguns exemplos reais dos valores que os timestamps podem tomar.

83750445625	77233774086	84493282045
83762652939	77245039227	84526407663
83790334918	77255879177	84545225670
83793511707	77266285604	84560103058
83801145084	77276788086	84573009497
83849209603	77311285393	84586043275
83852379891	77323052515	84603676662
83878952747	77333642485	84617433122
83907290177	77346761334	84632256371
83923995912	77357219895	84649682401

Figura 40. Exemplos de Timestamps.

Foram também removidos os parâmetros referentes ao comprimento e largura dos dedos. Isto deveu-se aos valores destes parâmetros serem sempre constantes, o que os torna inúteis para o treino do modelo, já que eles não vão afetar a exatidão e precisão do modelo. Na tabela 5 é mostrado os valores constantes do comprimento e largura dos vários dedos da mão.

	Comprimento	Largura
Polegar	99.86689	20.84575
Indicador	78.52512	19.91186
Médio	89.1054	19.5561
Anelar	84.86235	18.60887
Mínimo	66.69249	16.52985

Tabela 5. Comprimento e largura dos vários dedos segundo o Leap Motion.

Após esta análise manual dos parâmetros recolhidos, foi decidido dividir os restantes parâmetros das mãos em dois grupos baseados nas características desses parâmetros. Estes dois grupos criados foram o grupo dos parâmetros posicionais e o grupo dos parâmetros direcionais. Os parâmetros posicionais continham a posição da palma da mão, a posição estabilizada da palma da mão e as posições dos dedos. Faziam parte dos parâmetros direcionais o vetor normal da palma da mão, o vetor direcional da palma da mão e os vetores direcionais das pontas dos dedos.

Os resultados dos testes usando estes grupos de parâmetros foram extremamente uteis. Apenas o modelo treinado utilizando os parâmetros direcionais conseguiu fazer a separação dos gestos. O modelo que foi treinado utilizando apenas os parâmetros posicionais e o modelo que foi treinado utilizando os dois grupos não conseguiram fazer a divisão dos gestos e devido a isso os modelos colapsaram. Nós dizemos que um modelo colapsou quando este não consegue dividir as diferentes classes e devido a isso quando lhe é dado um novo exemplo para classificar, o modelo dá sempre a mesma previsão. A previsão constante é geralmente a categoria que aparecia mais nos exemplos que foram dados ao modelo para treinar.

O resultado destes testes pode ser visto nas figuras 41, 42 e 43. A tabela na figura 41 mostra os resultados obtidos durante o teste do modelo treinado na fase 1. A tabela na figura 42 mostra os resultados obtidos durante o teste do modelo treinado na fase 2 e

a tabela na figura 43 mostra os resultados obtidos durante o teste do modelo treinado na fase 3.

A primeira coluna das tabelas indica o grupo de parâmetros que foram utilizados para treinar o modelo de reconhecimento que foi usado durante os testes. A segunda coluna, “Gesto;Posição”, indica qual dos dez gestos e três ângulos do braço descritos no capítulo 3 . A terceira coluna, “Resultado Esperado”, indica a categoria a que o gesto pertence e a predição que o modelo de reconhecimento deve dar se este for exato. A quarta coluna, “Predição do modelo”, indica o resultado real que o modelo deu quando o exemplo de teste para classificar foi lhe dado. A quinta coluna, “Correto?”, indica se o modelo conseguiu classificar corretamente o exemplo de teste.

A última coluna, “% de Certeza”, indica a percentagem de certeza que o modelo tem na sua resposta. Se o modelo conseguiu identificar corretamente a categoria do exemplo de teste e se a percentagem de certeza for alta, significa que o modelo consegue facilmente identificar o gesto que foi feito pelo utilizador. Se o modelo identificou incorretamente a categoria do exemplo e a percentagem de certeza for alta, isto indica que existe algo extremamente errado nos parâmetros dados ao modelo para treinar. Se a percentagem de certeza for média ou baixa, indica que o modelo não está muito seguro na sua resposta e pode facilmente mudar a sua previsão. A percentagem de certeza também serve para analisar a precisão do modelo treinado, com modelos em que a percentagem de certeza não varia muito sendo mais precisos.

Parâmetros Usados	Gesto;Posição	Resultado Esperado	Predição do modelo	Correto?	% de Certeza
Direcionais	1;1	1	1	Sim	84,54%
	1;2	2	2	Sim	99,74%
	1;3	2	2	Sim	99,64%
	2;1	2	2	Sim	99,64%
	2;2	2	2	Sim	99,64%
Posicionais	1;1	1	2	Não	96,74%
	1;2	2	2	Sim	96,74%
	1;3	2	2	Sim	96,74%
	2;1	2	2	Sim	96,74%
	2;2	2	2	Sim	96,74%
Todos	1;1	1	2	Não	96,88%
	1;2	2	2	Sim	96,88%
	1;3	2	2	Sim	96,88%
	2;1	2	2	Sim	96,88%
	2;2	2	2	Sim	96,88%

Figura 41. Certeza do reconhecimento utilizando o treino do tipo 1 e diferentes grupos de parâmetros.

Parâmetros Usados	Gesto;Posição	Resultado Esperado	Predição do modelo	Correto?	% de Certeza
Direcionais	1;1	1	1	Sim	98,22%
	1;2	2	2	Sim	98,80%
	1;3	3	3	Sim	98,26%
	2;1	1	1	Sim	96,66%
	2;2	2	2	Sim	98,22%
Posicionais	1;1	1	1	Sim	33,33%
	1;2	2	1	Não	33,33%
	1;3	3	1	Não	33,33%
	2;1	1	1	Sim	33,33%
	2;2	2	1		33,33%
Todos	1;1	1	1	Sim	33,33%
	1;2	2	1	Não	33,33%
	1;3	3	1	Não	33,33%
	2;1	1	1	Sim	33,33%
	2;2	2	1	Não	33,33%

Figura 42. Certeza do reconhecimento utilizando o treino do tipo 2 e diferentes grupos de parâmetros.

Parâmetros Usados	Gesto;Posição	Resultado Esperadd	Predição do modelo	Correto?	% de Certeza
Direcionais	1;1	1	1	Sim	86,10%
	2;2	2	2	Sim	81,54%
	3;3	3	3	Sim	78,09%
	4;1	4	4	Sim	76,50%
	5;2	5	5	Sim	55,17%
	6;3	6	8	Não	80,11%
	7;1	7	7	Sim	84,47%
	8;2	8	6	Não	38,30%
	9;3	9	9	Sim	84,80%
	10;1	10	10	Sim	77,96%
Posicionais	1;1	1	7	Não	10,12%
	2;2	2	7	Não	10,12%
	3;3	3	7	Não	10,12%
	4;1	4	7	Não	10,12%
	5;2	5	7	Não	10,12%
	6;3	6	7	Não	10,12%
	7;1	7	7	Sim	10,12%
	8;2	8	7	Não	10,12%
	9;3	9	7	Não	10,12%
	10;1	10	7	Não	10,12%
Todos	1;1	1	4	Não	10,09%
	2;2	2	4	Não	10,09%
	3;3	3	4	Não	10,09%
	4;1	4	4	Sim	10,09%
	5;2	5	4	Não	10,09%
	6;3	6	4	Não	10,09%
	7;1	7	4	Não	10,09%
	8;2	8	4	Não	10,09%
	9;3	9	4	Não	10,09%
	10;1	10	4	Não	10,09%

Figura 43. Certeza do reconhecimento utilizando o treino do tipo 3 e diferentes grupos de parâmetros.

Na figura 44 encontra-se um exemplo demonstrativo de um caso em que o modelo SVM não conseguiu dividir os exemplos dados para treinar o modelo e como tal colapsou. O modelo colapsado resultante ao receber um novo caso para classificar irá dar sempre a mesma resposta, neste caso, amarelo. Na figura 45 é mostrado um caso em que o SVM consegue separar os exemplos.

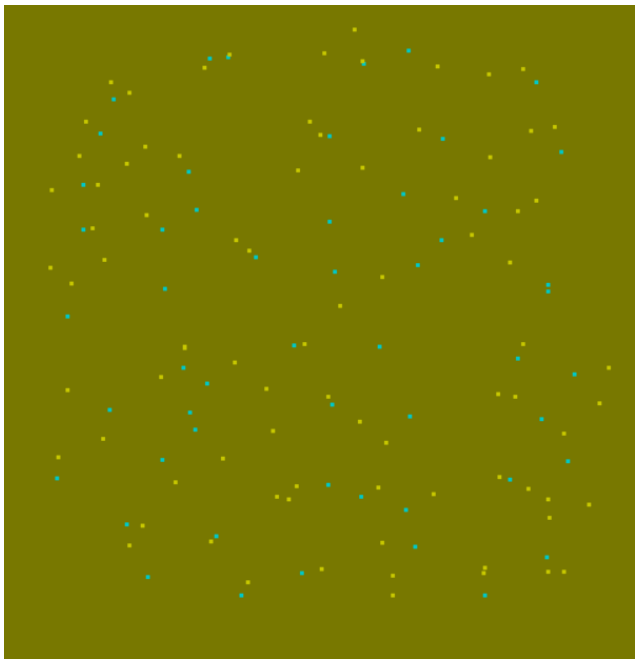


Figura 44. Exemplo gráfico de uma situação em que o algoritmo SVM colapsa.

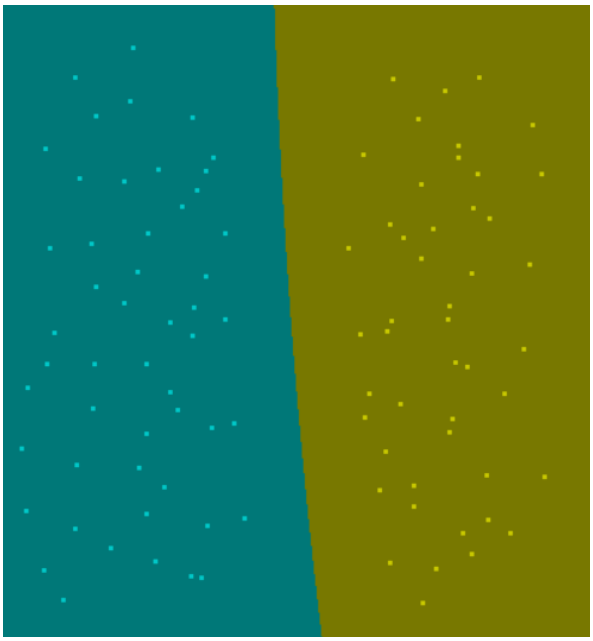


Figura 45. Exemplo gráfico de uma situação em que o algoritmo SVM consegue separar os exemplos.

Após esta análise ter sido concluída, foi inferido que os parâmetros posicionais prejudicam imenso a exatidão do modelo. A razão para os parâmetros posicionais prejudicarem tanto o modelo deve-se ao facto de a posição da mão no espaço do Leap Motion não afetar o gesto que está a ser feito pelo utilizador. O utilizador pode ter uma mão fechada no lado esquerdo ou no lado direito da área de reconhecimento do Leap Motion. Não importa em que lado da área de reconhecimento a mão está, não deixa de ser o mesmo gesto.

Os parâmetros posicionais seriam de maior utilidade num sistema de reconhecimento de gestos dinâmicos, onde conhecer a posição da mão no espaço seria útil para inferir os movimentos feitos pela mão entre dois gestos. Foi decidido que o treino do modelo do sistema de reconhecimento de gestos devia ser feito utilizando apenas os parâmetros direcionais.

4.1.2 Resultados da Otimização dos Parâmetros do SVM

Foi testado o efeito da utilização de diferentes kerneis no treino do modelo de reconhecimento. Os testes mostraram que os kerneis linear e RBF são os mais apropriados para o treino do modelo de reconhecimento. Isto deve-se a eles terem conseguido fazer o reconhecimento perfeito dos gestos quando treinado durante a primeira e segunda fase e o reconhecimento durante a terceira fase foi bom, com os problemas encontrados sendo similares para ambos os kerneis. Os resultados da utilização do kernel polinomial no treino dos modelos foram claramente muito maus, com os modelos treinados colapsando e tornando-se inúteis, como se pode ver nas figuras 46, 47 e 48 em que o modelo treinado com o kernel polinomial dá sempre a mesma predição.

Os resultados do treino do modelo durante a várias fases utilizando diferentes kerneis estão ilustrados nas figuras 46, 47 e 48. Os resultados mostram que os melhores kerneis

para fazer o treino do modelo são o linear e o RBF, com o kernel polinomial fazendo o modelo SVM colapsar em todas as fases.

Kernel Usado	Gesto;Posição	Resultado Esperado	Predição do modelo	Correto?	% de Certeza
Linear	1;1	1	1	Sim	95,86%
	1;2	2	2	Sim	95,33%
	1;3	2	2	Sim	99,99%
	2;1	2	2	Sim	99,99%
	2;2	2	2	Sim	99,99%
Polinomial	1;1	1	2	Não	96,61%
	1;2	2	2	Sim	96,61%
	1;3	2	2	Sim	96,61%
	2;1	2	2	Sim	96,61%
	2;2	2	2	Sim	96,61%
RBF	1;1	1	1	Sim	84,54%
	1;2	2	2	Sim	99,74%
	1;3	2	2	Sim	99,64%
	2;1	2	2	Sim	99,64%
	2;2	2	2	Sim	99,64%

Figura 46. Certeza do reconhecimento do modelo usando o treino do tipo 1 com diferentes kerneis.

Kernel Usado	Gesto;Posição	Resultado Esperado	Predição do modelo	Correto?	% de Certeza
Linear	1;1	1	1	Sim	97,06%
	1;2	2	2	Sim	99,06%
	1;3	3	3	Sim	99,61%
	2;1	1	1	Sim	94,68%
	2;2	2	2	Sim	96,95%
Polinomial	1;1	1	2	Não	33,95%
	1;2	2	2	Sim	33,95%
	1;3	3	2	Sim	33,95%
	2;1	1	2	Sim	33,95%
	2;2	2	2	Sim	33,95%
RBF	1;1	1	1	Sim	98,22%
	1;2	2	2	Sim	98,80%
	1;3	3	3	Sim	98,26%
	2;1	1	1	Sim	96,66%
	2;2	2	2	Sim	98,22%

Figura 47. Certeza do reconhecimento do modelo usando o treino do tipo 2 com diferentes kerneis.

Kernel Usado	Gesto;Posição	Resultado Esperado	Predição do modelo	Correto?	% de Certeza
Linear	1;1	1	1	Sim	92,59%
	2;2	2	2	Sim	88,46%
	3;3	3	3	Sim	73,20%
	4;1	4	4	Sim	69,55%
	5;2	5	5	Sim	89,78%
	6;3	6	8	Não	48,39%
	7;1	7	7	Sim	62,92%
	8;2	8	8	Sim	47,05%
	9;3	9	9	Sim	92,52%
	10;1	10	10	Sim	80,10%
Polinomial	1;1	1	1	Sim	10,60%
	2;2	2	1	Não	9,95%
	3;3	3	1	Não	10,01%
	4;1	4	1	Não	9,89%
	5;2	5	1	Não	9,49%
	6;3	6	1	Não	10,16%
	7;1	7	1	Não	10,06%
	8;2	8	1	Não	10,09%
	9;3	9	1	Não	9,78%
	10;1	10	1	Não	9,93%
RBF	1;1	1	1	Sim	86,10%
	2;2	2	2	Sim	81,54%
	3;3	3	3	Sim	78,09%
	4;1	4	4	Sim	76,50%
	5;2	5	5	Sim	55,17%
	6;3	6	8	Não	80,11%
	7;1	7	7	Sim	84,47%
	8;2	8	6	Não	38,30%
	9;3	9	9	Sim	84,80%
	10;1	10	10	Sim	77,96%

Figura 48. Certeza do reconhecimento do modelo usando o treino do tipo 3 com diferentes kerneis.

Como os kerneis, linear e RBF, tiveram resultados semelhantes com apenas algumas pequenas diferenças, nomeadamente nos gestos em que a previsão do modelo tem uma

percentagem de certeza média ou baixa, foi decidido escolher o kernel RBF devido a ele conseguir adaptar-se melhor a modelos em que apenas os exemplos de um caso estão espalhados. Esta situação vai provavelmente acontecer muitas vezes no sistema de reconhecimento final devido à introdução de gestos artificiais.

4.1.3 Resultados da Otimização dos Número de Dados de Treino

Começou-se por utilizar-se apenas os gestos recolhidos de apenas uma pessoa em todas as fases, ou seja, 300 gestos. Este teste inicial mostrou que o modelo treinado usando este número de indivíduos conseguiu fazer uma boa divisão dos gestos claramente diferentes, mas teve dificuldades no modelo da terceira fase a separar gestos similares, como o gesto 6 e 8, que foram confundidos pelo modelo de reconhecimento. Estes resultados são mostrados nas linhas referentes a um indivíduo das figuras 49, 50 e 51.

Após a análise dos resultados utilizando apenas um gesto foi feito o treino utilizando os gestos recolhidos de duas pessoas, ou seja, 600 gestos. Os resultados deste teste foram melhorares do que o anterior, com o modelo sendo mais exato. Ainda se verificam alguns problemas a nível da exatidão e certeza das previsões no modelo da terceira fase, havendo um gesto que o modelo não conseguiu reconhecer corretamente. Esperamos que estes problemas desapareçam quando adicionarmos mais gestos. Estes resultados são mostrados nas linhas referentes a dois indivíduos das figuras 49, 50 e 51.

A seguir foi feita a análise dos resultados utilizando os gestos recolhidos de cinco indivíduos, ou seja, 1500 gestos. A resultados do reconhecimento usando este número de indivíduos foram muito melhores do que o anterior, com uma exatidão perfeita para todos os testes e um grande aumento na certeza. Como era esperado o aumento do número de exemplos de treino tornou o modelo mais robusto e permitiu o modelo ter mais uma maior certeza das características que formam o gesto. Estes resultados são mostrados nas linhas referentes a cinco indivíduos das figuras 49, 50 e 51.

Foi também feita uma análise do reconhecimento utilizando todos os 3000 gestos recolhidos. Os resultados deste teste foram muito semelhantes aos obtidos quando fazendo o treino utilizando apenas os gestos de 5 indivíduos, com uma exatidão perfeita e alta certeza para o reconhecimento de todos os gestos. Estes resultados são mostrados nas linhas referentes a dez indivíduos das figuras 49, 50 e 51.

Nº de Individuos	Gesto;Posição	Resultado Esperado	Predição	Correcto?	% de Certeza
1	1;1	1	1	Sim	95,25%
	2;2	2	2	Sim	99,59%
	3;3	2	2	Sim	99,69%
	4;1	2	2	Sim	99,64%
	5;2	2	2	Sim	99,74%
	6;3	2	2	Sim	99,59%
	7;1	2	2	Sim	99,60%
	8;2	2	2	Sim	99,60%
	9;3	2	2	Sim	99,59%
	10;1	2	2	Sim	99,60%
2	1;1	1	1	Sim	94,82%
	2;2	2	2	Sim	99,82%
	3;3	2	2	Sim	99,86%
	4;1	2	2	Sim	99,82%
	5;2	2	2	Sim	99,89%
	6;3	2	2	Sim	99,82%
	7;1	2	2	Sim	99,82%
	8;2	2	2	Sim	99,82%
	9;3	2	2	Sim	99,82%
	10;1	2	2	Sim	99,82%

Nº de Individuos	Gesto;Posição	Resultado Esperado	Predição	Correcto?	% de Certeza
5	1;1	1	1	Sim	99,06%
	2;2	2	2	Sim	99,92%
	3;3	2	2	Sim	99,95%
	4;1	2	2	Sim	99,92%
	5;2	2	2	Sim	99,95%
	6;3	2	2	Sim	99,92%
	7;1	2	2	Sim	99,92%
	8;2	2	2	Sim	99,92%
	9;3	2	2	Sim	99,92%
	10;1	2	2	Sim	99,92%
10	1;1	1	1	Sim	99,98%
	2;2	2	2	Sim	99,84%
	3;3	2	2	Sim	99,97%
	4;1	2	2	Sim	99,88%
	5;2	2	2	Sim	99,99%
	6;3	2	2	Sim	99,93%
	7;1	2	2	Sim	99,87%
	8;2	2	2	Sim	99,88%
	9;3	2	2	Sim	99,93%
	10;1	2	2	Sim	99,85%

Figura 49. Comparação da precisão do reconhecimento do treino 1.

Nº de Individuos	Gesto;Posição	Resultado Esperado	Predição	Correcto?	% de Certeza
1	1;1	1	1	Sim	98,51%
	2;2	2	2	Sim	98,46%
	3;3	3	3	Sim	98,98%
	4;1	1	1	Sim	95,65%
	5;2	2	2	Sim	98,56%
	6;3	3	3	Sim	98,59%
	7;1	1	1	Sim	98,18%
	8;2	2	2	Sim	98,35%
	9;3	3	3	Sim	98,60%
	10;1	1	1	Sim	96,98%
2	1;1	1	1	Sim	99,36%
	2;2	2	2	Sim	99,20%
	3;3	3	3	Sim	99,56%
	4;1	1	1	Sim	98,71%
	5;2	2	2	Sim	99,35%
	6;3	3	3	Sim	99,40%
	7;1	1	1	Sim	98,77%
	8;2	2	2	Sim	99,13%
	9;3	3	3	Sim	99,27%
	10;1	1	1	Sim	99,06%

Nº de Individuos	Gesto;Posição	Resultado Esperado	Predição	Correcto?	% de Certeza
5	1;1	1	1	Sim	99,70%
	2;2	2	2	Sim	99,29%
	3;3	3	3	Sim	99,47%
	4;1	1	1	Sim	99,47%
	5;2	2	2	Sim	99,63%
	6;3	3	3	Sim	99,78%
	7;1	1	1	Sim	99,86%
	8;2	2	2	Sim	99,87%
	9;3	3	3	Sim	99,61%
	10;1	1	1	Sim	99,68%
10	1;1	1	1	Sim	99,26%
	2;2	2	2	Sim	99,69%
	3;3	3	3	Sim	100,00%
	4;1	1	1	Sim	97,95%
	5;2	2	2	Sim	99,86%
	6;3	3	3	Sim	100,00%
	7;1	1	1	Sim	99,57%
	8;2	2	2	Sim	99,89%
	9;3	3	3	Sim	100,00%
	10;1	1	1	Sim	99,10%

Figura 50. Comparação da precisão do reconhecimento do treino 2.

Nº de Individuos	Gesto;Posição	Resultado Esperado	Predição	Correcto?	% de Certeza
1	1;1	1	1	Sim	85,77%
	2;2	2	2	Sim	81,14%
	3;3	3	3	Sim	75,72%
	4;1	4	4	Sim	81,12%
	5;2	5	5	Sim	53,00%
	6;3	6	8	Não	85,11%
	7;1	7	7	Sim	79,81%
	8;2	8	6	Não	34,77%
	9;3	9	9	Sim	89,96%
	10;1	10	10	Sim	78,35%
2	1;1	1	1	Sim	94,84%
	2;2	2	2	Sim	89,36%
	3;3	3	3	Sim	84,03%
	4;1	4	4	Sim	89,57%
	5;2	5	5	Sim	93,17%
	6;3	6	8	Não	56,66%
	7;1	7	7	Sim	93,44%
	8;2	8	8	Sim	60,61%
	9;3	9	9	Sim	94,32%
	10;1	10	10	Sim	80,08%

Nº de Individuos	Gesto;Posição	Resultado Esperado	Predição	Correcto?	% de Certeza
5	1;1	1	1	Sim	97,93%
	2;2	2	2	Sim	94,37%
	3;3	3	3	Sim	95,23%
	4;1	4	4	Sim	94,63%
	5;2	5	5	Sim	98,84%
	6;3	6	6	Sim	79,69%
	7;1	7	7	Sim	96,52%
	8;2	8	8	Sim	77,46%
	9;3	9	9	Sim	95,93%
	10;1	10	10	Sim	97,49%
10	1;1	1	1	Sim	99,12%
	2;2	2	2	Sim	97,88%
	3;3	3	3	Sim	98,51%
	4;1	4	4	Sim	96,77%
	5;2	5	5	Sim	99,56%
	6;3	6	6	Sim	65,54%
	7;1	7	7	Sim	97,27%
	8;2	8	8	Sim	85,39%
	9;3	9	9	Sim	98,48%
	10;1	10	10	Sim	99,38%

Figura 51. Comparação da precisão do reconhecimento do treino 3.

Com base nos resultados obtidos durante o treino da terceira fase, que são mostrados na figura 51, foi concluído que os gestos 1, 2, 3, 4, 7, 9 e 10 foram os gestos que o modelo de reconhecimento conseguiu reconhecer mais facilmente com cada um destes gestos tendo uma percentagem de certeza superior a 75% nos testes em que o número de exemplos de treino era apenas 300 e aumentou ainda mais quando o número de exemplos aumentou.

O modelo teve um bocado de dificuldade a reconhecer o gesto 5 quando o modelo foi treinado usando apenas 300 gestos, mas esta dificuldade rapidamente desapareceu quando o número de exemplos de treino aumentou. Isto provavelmente deveu-se ao modelo ter tido dificuldades a fazer uma clara distinção entre este gesto e os gestos 3 e 4. A adição de mais exemplos de treino ajudou o modelo a perceber as diferenças entre o gesto 5 e os restantes gestos.

O gesto 6 e 8 foram os gestos que o modelo teve mais dificuldade a reconhecer, especialmente o gesto 6. Isto era esperado já que ambos os gestos são muito semelhantes.

4.1.4 Resultados dos Testes dos Gestos Automáticos

Começou-se por treinar o modelo usando gestos criados automaticamente e testando-o usando gestos criados manualmente. Foram executadas trinta iterações deste teste, sendo obtido no final a média da exatidão de todas as iterações e a variação da precisão do reconhecimento. Os resultados deste teste mostraram que o modelo conseguiu reconhecer todos os gestos manuais dados para ele reconhecer e a precisão do reconhecimento para cada iteração variou entre 85,65% e 86,15%.

Após o modelo treinado usando gestos automáticos ter sido testado usando gestos manuais, foi feito o teste do modelo utilizando gestos recolhidos pelo Leap Motion em tempo real. Este teste mostrou que o modelo tem uma exatidão perfeita nunca falhando o reconhecimento de um gesto e a precisão do reconhecimento do gesto varia entre 70% a 92%. Este teste também confirmou as nossas expetativas acerca dos limites em que o gesto é reconhecido, com um gesto deixando de ser reconhecido quando um dos componentes dos vetores da palma da mão tem uma alteração superior a 0,5 ou -0,5.

O teste que se seguiu testou o modelo treinado usando gestos automáticos ao remover parte dos gestos criados automaticamente para treinar o modelo, usando esses gestos removidos para testar o modelo. Os testes foram feitos alterando o número de gestos criados para treinar o modelo, sendo testados modelos que foram treinados utilizando 10 gestos, 50 gestos e 100 gestos. Foi também sendo alterados a percentagem de gestos criados que eram utilizados para treinar o modelo, com os restantes gestos sendo usados para testar o reconhecimento do modelo. Foram treinados modelos em que a percentagem de gestos criados utilizados para treinar o modelo era 90%, 70% e 50%. Foram executadas trinta iterações deste teste para cada configuração possível.

Na tabela 5 são apresentados os resultados destes testes. Na tabela é mostrando a clara influência do número de exemplos dados para treinar o modelo na exatidão e precisão do modelo. Como se pode ver, existe um claro aumento na exatidão à medida que o número de exemplos de treino aumentava até ser alcançada uma exatidão média perfeita. Também foi notado um aumento da precisão do reconhecimento e uma diminuição da variação da precisão à medida que o número de exemplos de teste ia aumentando, conforme se pode verificar nas três últimas colunas da tabela 6.

Número De Gestos Criados	Percentagem Treino/Teste	Exatidão Média	Variação na Precisão	Precisão Média
10	90-10	96,67%	86-92%	89,1%
	70-30	85,55%	82-97%	88,6%
	50-50	68,67%	75-100%	86%
50	90-10	100%	94-97%	96,1%
	70-30	100%	93-96%	94,9%
	50-50	100%	87-97%	92,4%
100	90-10	100%	97-98%	97,9%
	70-30	100%	96-98%	97,1%
	50-50	100%	95-97%	96,3%

Tabela 6. Resultados dos testes do modelo usando gestos automáticos tanto para o treino como para o teste.

Depois foi feito o treino do modelo usando gestos criados manualmente e testando-o usando gestos criados automaticamente. Os resultados deste teste foram semelhantes aos do seu oposto, o treino do modelo com gestos automáticos e teste com manuais, obtendo também uma exatidão perfeita com apenas uma pequena subida de 1% na precisão do reconhecimento.

O teste seguinte foi o do modelo treinado usando gestos manuais e testado utilizando gestos recolhidos pelo Leap Motion em tempo real. Os resultados deste teste foram similares aos obtidos quando treinado o modelo usando gestos automáticos, com a exatidão do reconhecimento sendo perfeita, mas com a precisão do reconhecimento aumentando para uma variação de 70% a 95%.

O último teste consistiu em treinar o modelo usando gestos manuais, mas removeu parte dos gestos manuais para usar no teste do modelo. A percentagem de gestos que eram utilizados para treinar o modelo foi sendo alterado ao longo dos testes, sendo treinados modelos em que a percentagem de gestos criados utilizados para treinar o modelo era 90%, 70% e 50%. Foram executadas trinta iterações deste teste para cada uma destas percentagens.

A tabela 7 apresenta os resultados dos testes do modelo, sendo notada uma clara diminuição da precisão do modelo e um aumento na variação da precisão que coincide com a diminuição do número de exemplos dados para treinar o modelo.

Percentagem Treino/Teste	Exatidão Média	Variação na Precisão	Precisão Média
90-10	100%	91-96%	93,9%
70-30	100%	90-95%	93,4%
50-50	100%	80-93%	90,8%

Tabela 7. Resultados dos testes do modelo usando gestos manuais tanto para o treino como para o teste.

A conclusão alcançada com estes treinos foi que apesar de o modelo treinado utilizando gestos obtidos manualmente ter uma precisão ligeiramente superior ao modelo treinado usando gestos criados automaticamente, os dois modelos são efetivamente iguais a nível de reconhecimento, com os gestos automáticos mostrando-se superiores aos gestos manuais a nível do esforço necessário para os recolher.

4.1.5 Problemas Encontrados que Afetam o Sistema

Ao longo do projeto foram encontrados vários problemas, principalmente no que diz respeito ao sensor, que afetam a exatidão e precisão do sistema de reconhecimento. Estes problemas foram:

1. Quando um utilizador faz um gesto em que a palma da mão não está virada para cima ou para baixo, existe uma grande probabilidade que o Leap Motion tenha dificuldade a reconhecer a posição e direção do polegar, não conseguindo reconhecer se o polegar está ou não levantado. Este problema foi corrigido ao fazer que quando o modelo reconhece o gesto como uma mão fechada ou como apenas o polegar levantado com a palma da mão que faz o gesto virada na

direção da outra mão. Caso o sistema detete um destes dois casos o programa irá pedir que o utilizador mantenha o gesto e rode a mão de forma que a palma da mão fique virada para baixo, isto permite o sistema reconhecer o gesto que está a ser feito.

2. O Leap Motion também mostrou grandes dificuldades a reconhecer gestos em que o dedo médio e o dedo anelar tomam posições diferentes aos dedos adjacentes. Este parece ser um problema relacionado ao Leap Motion e não encontrámos maneira de solucionar o problema.
3. Notou-se que vários gestos são interpretados pelo Leap Motion de maneiras diferentes dependendo da mão que está a ser utilizada para fazer o gesto. Um exemplo disto ocorre quando um gesto é feito em as pontas dos dedos do polegar e o médio entram em contacto, quando fazendo utilizando a mão esquerda o Leap Motion consegue interpretar o gesto corretamente, mas se o gesto for feito utilizando a mão direita o Leap Motion não consegue interpretar o gesto corretamente e também fecha o anelar. Este problema está associado ao método que o Leap Motion utiliza para recolher dados e não encontrámos maneira de solucionar o problema, sendo a única solução ter precaução quando recolhendo gestos. Na tabela 8 encontra-se demonstrado um exemplo de como o Leap Motion interpreta o mesmo gesto de uma maneira diferente.

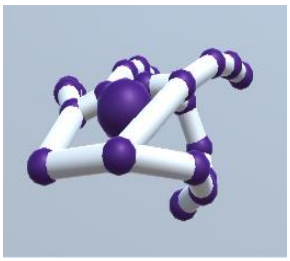
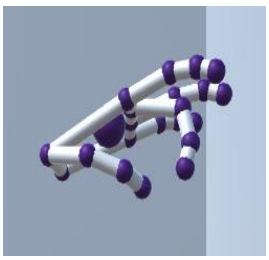
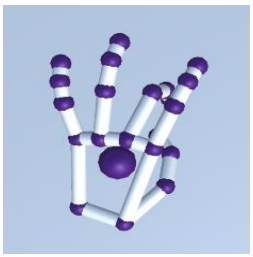
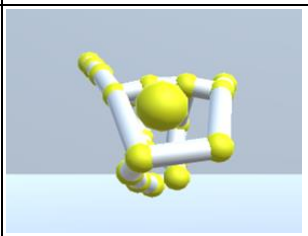
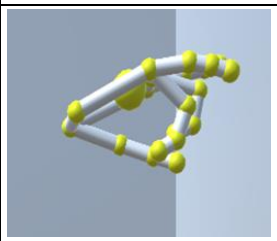
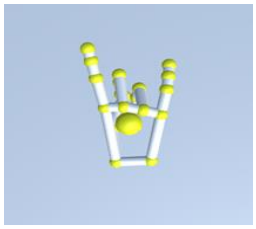
	Vista da Frente	Vista Lateral	Vista de Cima
Mão Esquerda			
Mão Direita			

Tabela 8. Comparação de como o Leap Motion interpreta de maneira diferente o mesmo gesto feito por mãos diferentes.

Foi encontrado outro problema na versão feita em C# do sistema de reconhecimento de gestos. Esta versão do sistema tem vários problemas a reconhecer gestos recebidos em tempo real, especialmente quando comparado com a versão do sistema feita em java. Estes problemas incluem:

- Quedas na certeza da previsão do modelo de reconhecimento quando comparada com a versão feita em java. A certeza da previsão na versão em C# quando reconhecendo gestos é entre 15% a 20% pior do que a certeza da previsão na versão em java.
- O modelo de reconhecimento, treinado pela versão do algoritmo de aprendizagem feito em C#, também sofre regularmente de graves casos de overfitting quando os exemplos de treino de uma classe estão mais espalhados do que os exemplos de outras classes. Em casos como este os gestos das classes em que os exemplos de treino não estão muito espalhados só são reconhecidas se eles forem exatamente iguais aos dos exemplos de treino. Isto não acontece com a versão feita em java.

A razão para este overfitting do modelo e queda na certeza da previsão não foi apurada devido a falta de tempo. Uma possível razão para isto pode ser os métodos usados pelo algoritmo da versão do LIBSVM em C# serem menos eficientes do que os métodos usados pela versão em java do LIBSVM.

4.2 Aplicação Criada

A aplicação criada no final do trabalho está dividida em cinco módulos, que são, por ordem:

- Módulo de Recolher Gestos: Este módulo permite gravar os dez gestos que serão usados para testar as capacidades de reconhecimento do modelo treinado. Também permite combinar todos os gestos recolhidos num único ficheiro que irá ser usado para treinar o modelo.
- Módulo de Reconhecimento de Gestos: Permite testar o reconhecimento do modelo treinado usando os gestos recolhidos no módulo de recolher gestos. Estes testes do reconhecimento são feitos em tempo real através do Leap Motion.
- Módulo de Criar Teclados: Permite o utilizador recolher e visualizar gestos criados por ele. Também permite associar esses gestos a uma letra do teclado e criar um ficheiro com o teclado gestual baseado nessas conexões.
- Módulo de Reconhecimento de Teclados: Este módulo da aplicação permite o utilizador utilizar os teclados gestuais criados no módulo de criar teclados. O utilizador pode fazer um dos gestos definidos no teclado e quando o sensor Leap Motion reconhecer esse gesto a letra que o gesto está associado será escrita.
- Módulo de Comparação de Modelos Automáticos e Modelos Manuais: Permite testar a exatidão e precisão dos modelos treinados utilizando ou gestos automáticos ou gestos manuais, usando vários tipos de teste.

Todas estas partes estão conectadas através de um menu inicial. A figura 52 mostra a estrutura do menu inicial da aplicação.

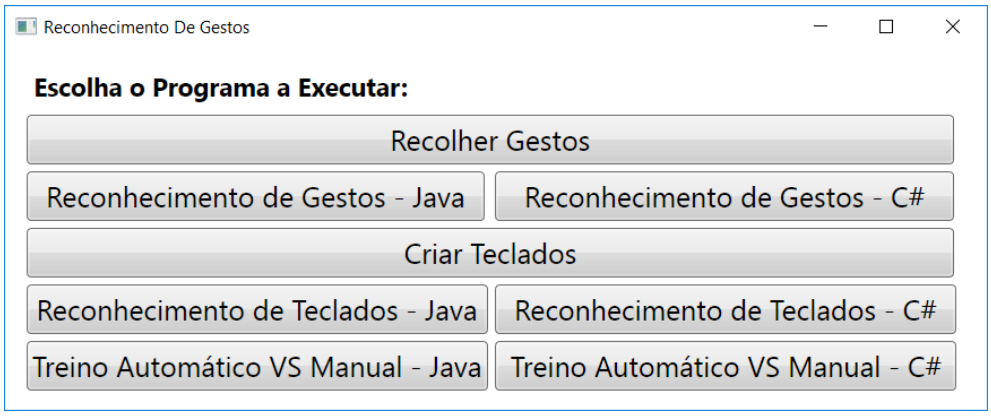


Figura 52. Menu inicial da aplicação.

4.2.1 Módulo de Recolher Gestos

O primeiro módulo da aplicação foi desenvolvido com o Unity e permite a recolha dos dez gestos, descritos no capítulo anterior, em qualquer uma das posições da palma da mão, também descritas no capítulo anterior. Todos os gestos recolhidos são combinados em apenas um ficheiro que é utilizado no módulo de reconhecimento de gestos para fazer o treino do modelo de reconhecimento.

O menu inicial desta parte da aplicação consiste na escolha do utilizador a que os gestos gravados irão estar associados, com a opção de adicionar um novo utilizador. Também existe a opção que permite começar o processo de gravar um gesto, a opção que permite começar o processo de transferir todos os gestos recolhidos para apenas um ficheiro e a opção que permite sair deste módulo da aplicação. Na figura 53 é apresentado o menu inicial do módulo de recolher gestos.

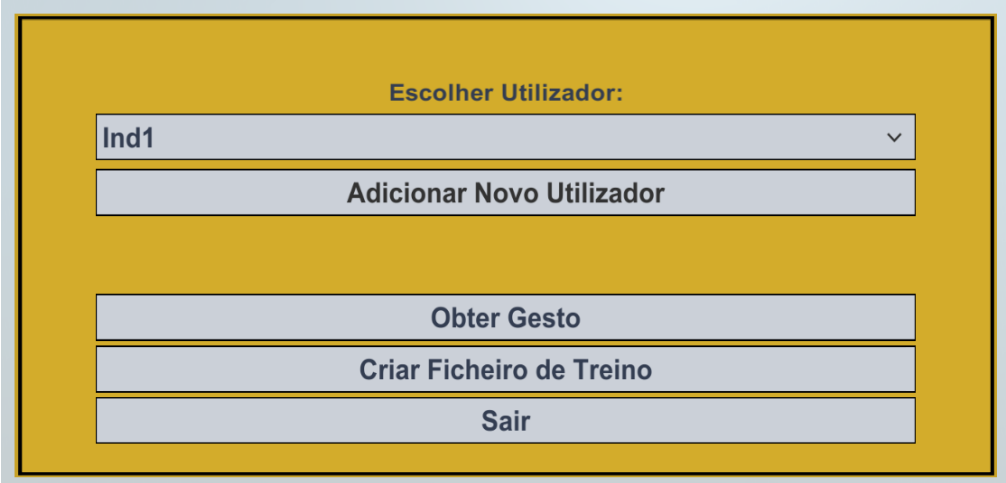


Figura 53. Menu inicial da aplicação de recolha de gestos.

Quando um utilizador indica que quer adicionar um novo utilizador, ele necessita escrever o nome do nove utilizador na caixa de texto do novo menu mostrado. Após o nome ter sido escrito, o utilizador deve selecionar a opção para adicionar o novo indivíduo. Se o nome tiver apenas três letras ou menos e se o nome existe o indivíduo não será adicionado e o nome tem de ser alterado. Quando um nome é aceite e o novo indivíduo é adicionado a aplicação mantem-se no menu para adicionar novos indivíduos,

só voltando ao menu inicial ao selecionar quando essa opção é selecionada. Na figura 54 é apresentado o menu utilizado para adicionar um novo utilizador.

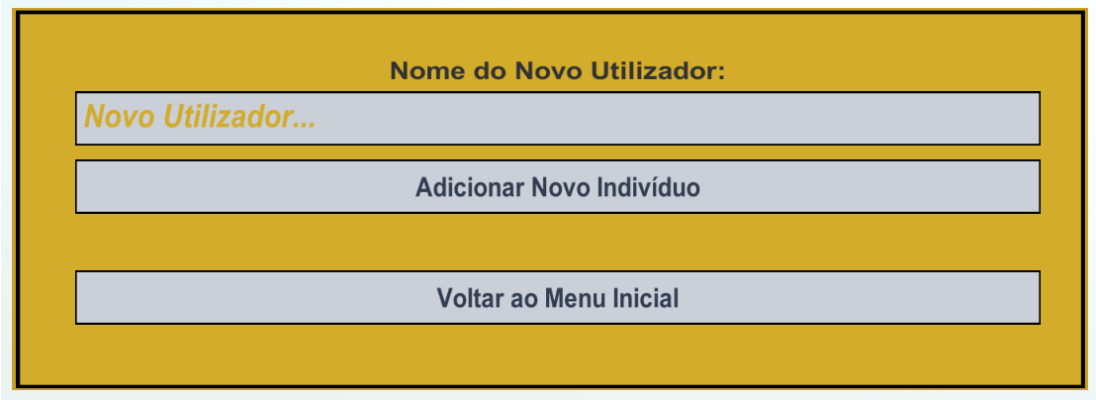


Figura 54. Menu para adicionar novo utilizador.

Ao escolher obter um novo gesto para o utilizador selecionado, o utilizador tem de escolher o gesto que quer gravar e a direção da palma da mão que o gesto deve ter. Os gestos e as posições da palma da mão são os que foram definidos no capítulo três, mas também é mostrada uma descrição deles na aplicação quando estes são selecionados. Após ambos terem sido definidos o utilizador avançará para a gravação do gesto. A figura 55 mostra o menu em que o utilizador escolhe o gesto e direção da palma da mão com uma descrição de cada opção mostrada para ambas as opções.

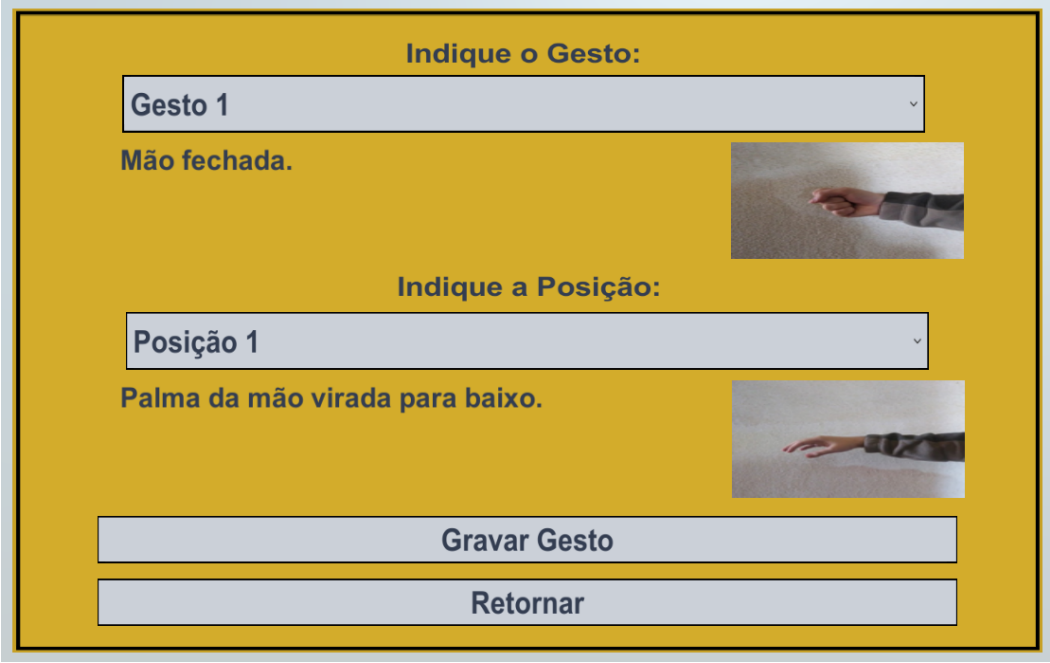


Figura 55. Escolha do gesto a ser gravado e da posição da mão.

Após o utilizador ter escolhido o gesto e posição que que gravar, será mostrado um aviso acerca do tempo que ele tem para fazer o gesto que gravar. É também dado ao utilizador uma última oportunidade para mudar o gesto ou posição da mão que quer gravar ao permitir o utilizador voltar à janela anterior, esta janela é mostrada na figura 56.

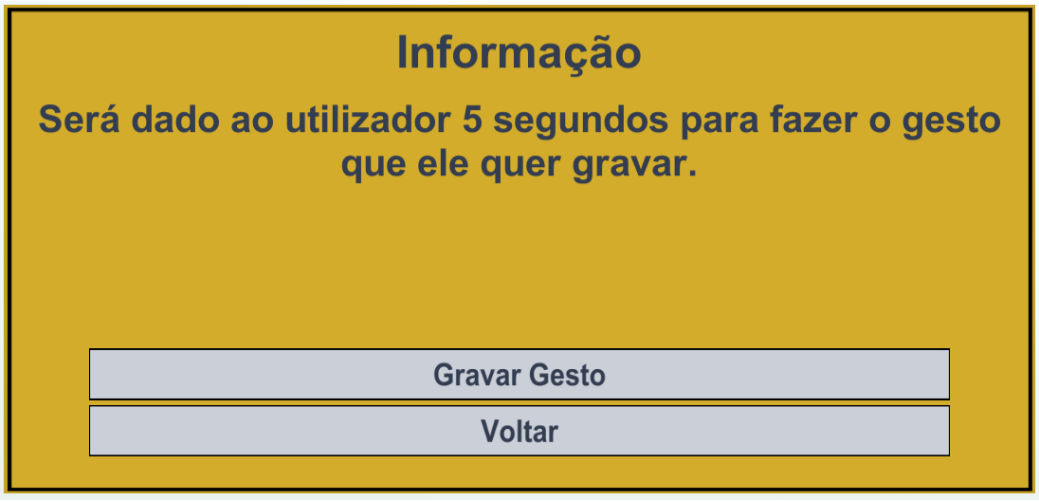


Figura 56. Aviso acerca do processo de gravação do gesto.

Após o utilizador ter confirmado que quer continuar, a janela mudará e mostrará uma área onde o Leap Motion está ativado e o utilizador pode fazer gestos, os gestos feitos serão mostrados na janela. O utilizador tem cinco segundos para mover a mão para o gesto que ele quer fazer, antes da aplicação gravar o gesto. Após o gesto ter sido gravado, a aplicação avisa o utilizador que o gesto foi gravado com sucesso e sai da janela em que o gesto foi gravado três segundos após feita a gravação. A figura 57 mostra a área da aplicação em que o gesto é gravado.

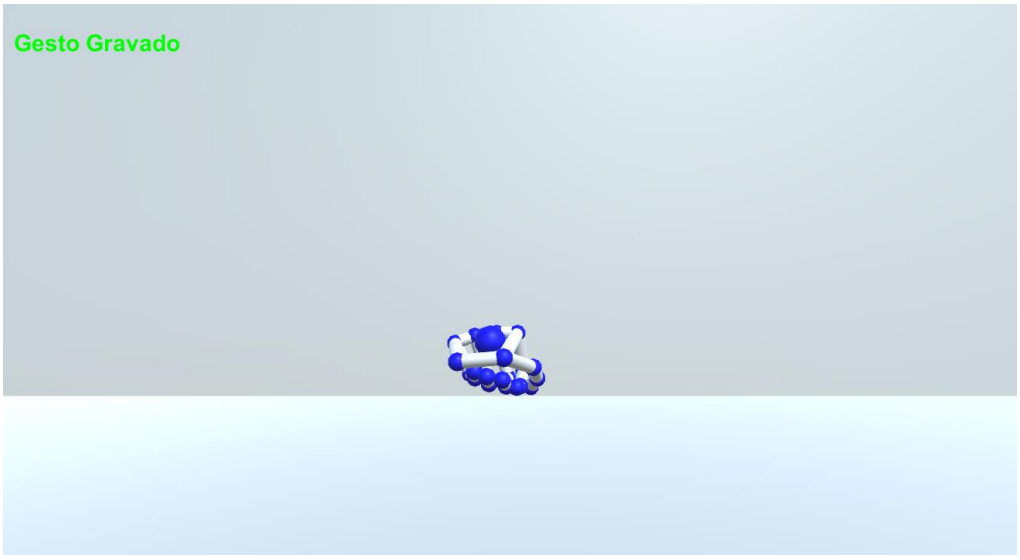


Figura 57. Gravação do gesto especificado.

Após o gesto ter sido gravado, é mostrado ao utilizador três imagens do gesto gravado: uma vista da frente, uma vista lateralmente e uma vista de cima. O utilizador escolherá se quer manter ou remover o gesto, dependendo se ele considerar se o gesto está bem feito ou não. A figura 58 mostra um exemplo disto, onde o utilizador escolhe se quer manter o gesto.

Figura 58. Aceitar ou rejeitar o gesto gravado.

Após aceitar ou rejeitar o gesto o utilizador volta à janela para selecionar o gesto e a posição da mão que quer gravar (Figura 55).

Quando no menu inicial é pressionado o botão para "Criar Ficheiro de Treino", é mostrado ao utilizador o menu para criar o ficheiro de treino. O processo de criar o ficheiro de treino que contêm todos os gestos recolhidos pelos vários utilizadores consiste em escrever o nome do ficheiro de treino e carregar no botão "Criar Ficheiro de Treino", o nome do ficheiro deve ter mais de três letras para ser aceite, a figura 59 mostra o menu que o utilizador utiliza para criar o ficheiro de treino com todos os gestos gravados.

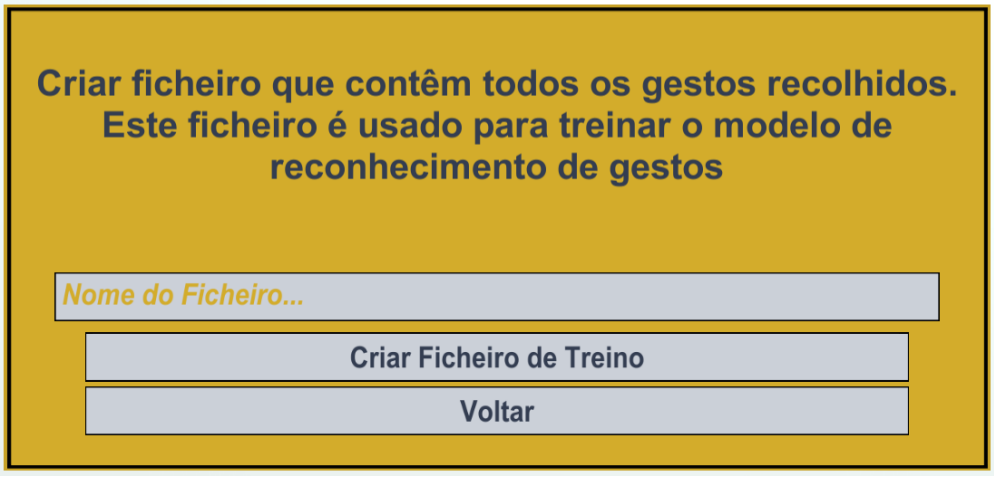


Figura 59. Menu para criar ficheiros de treino.

4.2.2 Reconhecimento de Gestos

O módulo foi implementado em java e em C# e testa o reconhecimento do modelo treinado utilizando os gestos obtidos na parte anterior.

O módulo obtém o ficheiro com todos os gestos recolhidos e irá treinar o modelo usando esses gestos, permitindo após o treino testar se o modelo tem um bom reconhecimento, usando o Leap Motion. A aplicação também permite três tipos diferentes de reconhecimento dos gestos, estes são: reconhecer apenas se a mão está fechada, identificar a direção da palma da mão e reconhecer que gesto está a ser feito.

O utilizador também pode modificar o número de frames que o modelo de reconhecimento de gestos faz o reconhecimento. Isto é feito ao dividir o reconhecimento do modelo em duas categorias: contínuo e descontínuo.

O reconhecimento contínuo reconhece todas as frames que o modelo recebe do Leap Motion. No reconhecimento descontínuo o modelo recebe apenas uma frame e depois não reconhece as várias frames seguintes. O número de frames entre reconhecimentos é definido pelo utilizador. O número de frames entre reconhecimentos no reconhecimento descontínuo deve ser maior ou igual a 10. Isto deve-se ao facto de se o número for muito pequeno, não se notará nenhuma diferença entre o reconhecimento contínuo e o reconhecimento descontínuo.

A figura 60 demonstra a interface desta parte da aplicação, com o ficheiro com os gestos recolhidos e o tipo de treino selecionados e com um modelo já treinado fazendo o reconhecimento do gesto que está sendo feito nesse momento pelo Leap Motion utilizando um reconhecimento contínuo.

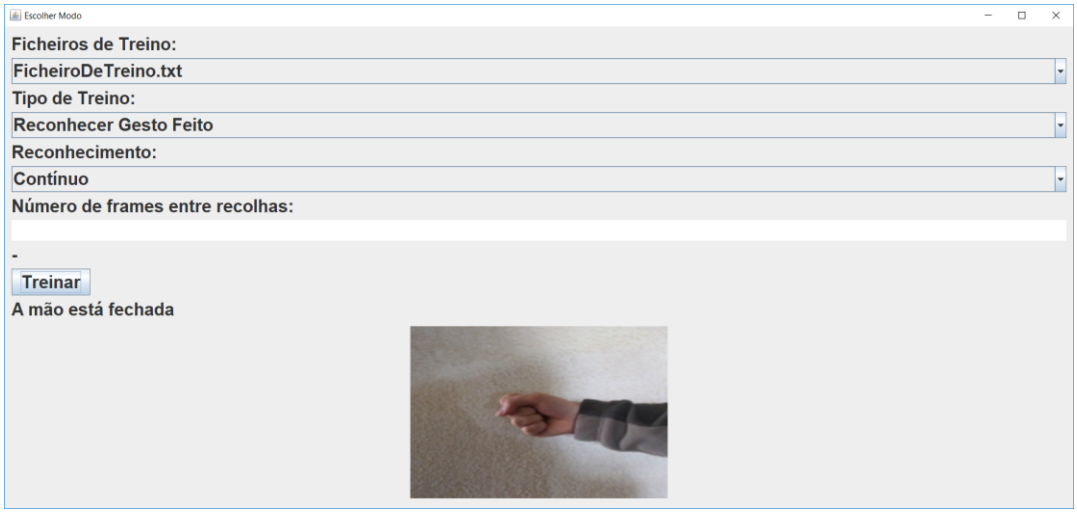


Figura 60. Escolha dos parâmetros para treinar o modelo.

4.2.3 Criação de Teclados

O módulo de criação de teclados foi implementado em Unity e permite a recolha de gestos pessoais do utilizador, sem limitações nos gestos que o utilizador pode fazer impostas pelos programadores. A conexão desses gestos a teclas do teclado e a utilização dessas conexões para a criação de um teclado gestual que será guardado num ficheiro para outras aplicações utilizarem.

Este módulo da aplicação começa por apresentar um menu com várias opções. Estas opções são:

1. Gravador de Gestos: Grava um gesto feito pelo utilizador usando o Leap Motion.
2. Visualizador de Gestos: Permite visualizar os gestos que foram gravados.
3. Alterar Teclado: Permitirá criar conexões entre os gestos guardados e teclas do teclado gerando um teclado gestual. Também permite guardar esse teclado gestual num ficheiro que pode ser utilizado por outras aplicações.
4. Sair: Fecha este módulo da aplicação.

Na figura 61 é apresentado o menu inicial desta parte da aplicação.

Figura 61. Menu inicial da aplicação para criar teclados gestuais.

Ao escolher gravar um gesto, primeiro será pedido o nome do gesto que vai ser gravado. Após o nome ter sido inserido, começará o processo de gravação do gesto, este processo será igual ao processo usado para recolher gestos no módulo de recolher gestos, com o sistema dando algum tempo para o utilizador por a mão na posição que quer antes de gravar o gesto e com a possibilidade aceitar ou rejeitar o gesto feito. O gesto será gravado num ficheiro com o nome escolhido pelo utilizador. Um ficheiro só poderá gravar um gesto. Na figura 62 é apresentado o menu onde o nome do ficheiro é inserido antes de começar a gravação do gesto.

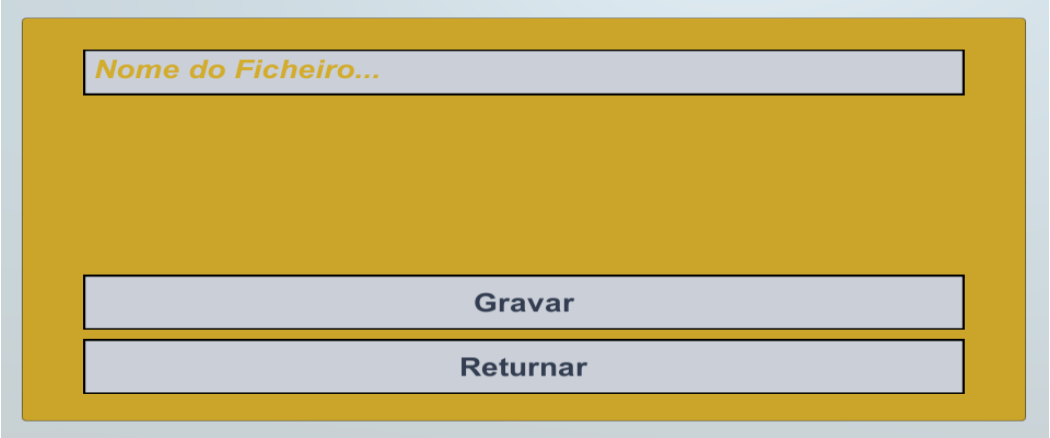


Figura 62. Indicação do nome do gesto a ser guardado.

A opção de visualizar um gesto já guardado permite o utilizador escolher um de todos os gestos que foram guardados anteriormente e visualizar esse gesto em três perspetivas: frente, lado e cima. A figura 63 mostra o menu usado para escolher qual dos gestos gravados o utilizador quer visualizar.

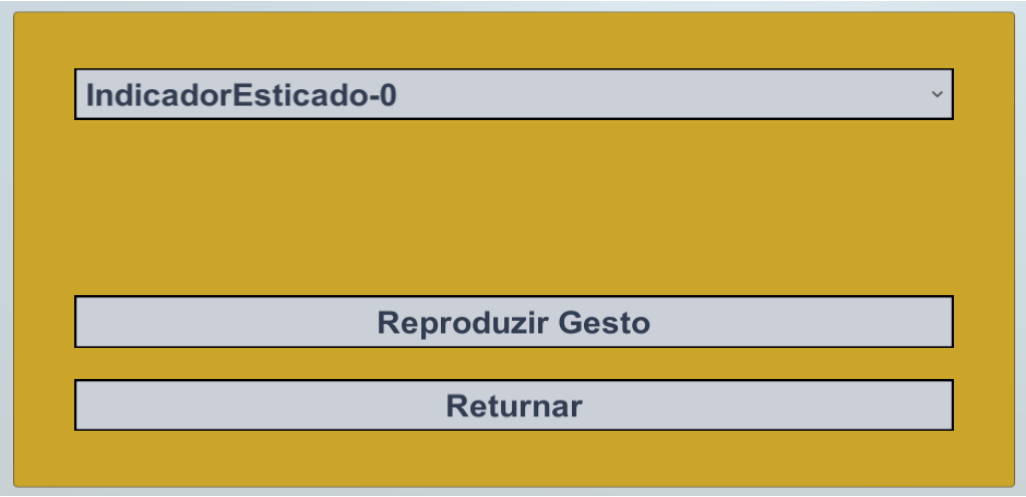


Figura 63. Escolha do gesto gravado para visualizar.

A figura 64 apresenta uma janela onde é visualizado um gesto que foi gravado anteriormente e que o utilizador quer ver, após selecionar a opção "Reproduzir Gesto" da figura 63.

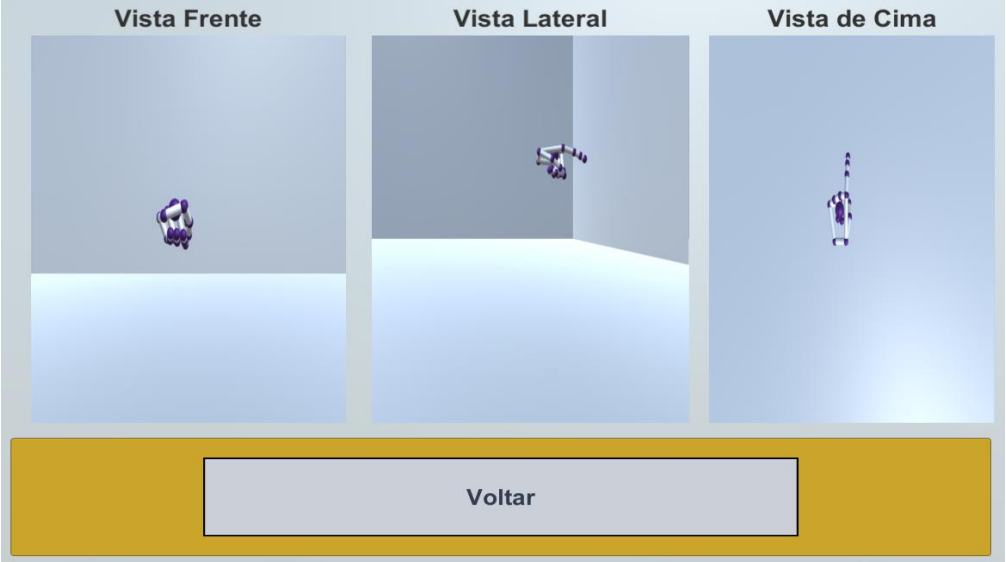


Figura 64. Visualização do gesto guardado escolhido.

A opção para definir e alterar os teclados gestuais irá mostrar uma janela onde ao premir a opção para adicionar uma tecla. O utilizador poderá carregar em qualquer tecla do teclado e esta será guardada pelo sistema. Ao premir a opção de concluir a criação de uma tecla o sistema conectará a tecla selecionada com o gesto atualmente selecionado.

É possível alterar a associação entre uma tecla ou gesto que já foram associados. Para fazer isto é necessário apenas associar o gesto ou tecla já usado ao novo elemento que o utilizador quer usar e premir a opção para concluir a criação de uma tecla. Ao fazer isto a associação antiga, que usava o gesto ou tecla, vai ser eliminada e vai ser criada a nova associação entre o elemento antigo e o novo elemento.

O utilizador pode repetir este processo até todas as teclas que ele quer utilizar estarem associadas a um gesto. Quando isso acontecer, o utilizador pode criar um ficheiro com o teclado gestual criado, ao indicar o nome do teclado e premindo a opção “Criar Teclado”. A figura 65 apresenta a estrutura da janela que permite criar teclados gestuais.

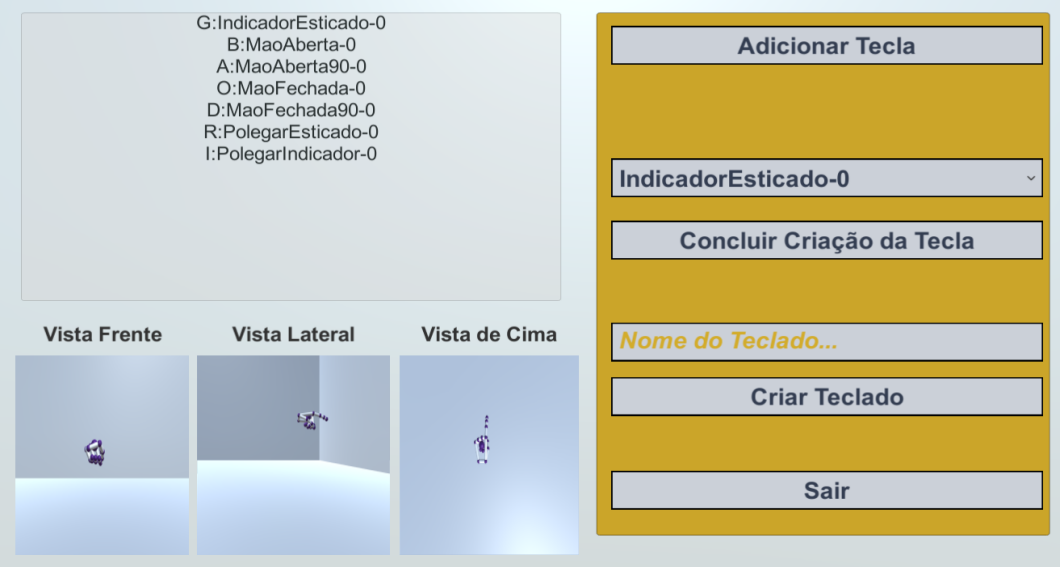


Figura 65. Menu para criação de um teclado gestual.

A figura 66 mostra a estrutura de um ficheiro com associações entre teclas do teclado e gestos. Uma associação entre uma tecla e um gesto começa por referir qual é a tecla que está a ser associada a um gesto, no caso do exemplo da figura 66 a teclas que estão a ser usadas são a “A”, “B” e “C”. Após a tecla aparecem os caracteres “-E”, estes caracteres servem para mostrar que a tecla acabou e que a seguir são os parâmetros do gesto a que a tecla está associada. Os parâmetros do gesto guardados são apenas o número de mãos, se uma mão é esquerda ou direita, os vetores normal e direcional da palma da mão e os vetores direcionais das pontas dos dedos. Quando os parâmetros do gesto acabam de ser lidos, aparece o caracter “-” para mostrar que a associação acabou e que a seguir bem uma nova associação.

```
A-E
1;L;-0.1430199;-0.8961034;-0.4201715;0.05985305;0.4159278;-0.
9074259;-0.1407949;-0.06175148;-0.9881111;-0.2768931;-0.66382
81;0.6947391;-0.06416489;-0.5631163;0.8238828;0.05583323;-0.5
721267;0.8182626;0.2682139;-0.6297659;0.7290104;
-
B-E
1;L;0.09492815;-0.9930748;0.06921709;0.1019241;-0.05946984;-0
.993013;0.8724015;0.1586988;-0.4623097;0.3427801;0.01690317;-
0.9392636;0.06752007;-0.006005748;-0.9976998;-0.2353701;0.025
76317;-0.9715643;-0.5560252;-0.02526304;-0.8307813;
-
```

Figura 66. Exemplo de um ficheiro de associações entre gestos e teclas.

4.2.4 Reconhecimento de Teclados

O módulo de reconhecimento de teclados foi implementado em java e em C# e permite testar o reconhecimento dos gestos que pertencem a um teclado gestual e a conexão destes gestos com as teclas que lhes foram atribuídos. Para executar este teste o utilizador necessitará indicar o teclado gestual guardado que ele quer utilizar e o nome do modelo de reconhecimento que será criado quando o módulo acabar de treinar o modelo, usando o teclado gestual selecionado.

Quando o teclado gestual for selecionado o utilizador necessitará escolher o tipo de treino que quer utilizar: específico ou genérico. O treino específico treina o modelo de maneira que mesmo que o gesto feito pela mão seja o mesmo, se a direção da palma da mão não for semelhante o gesto será considerado inválido. Se o treino escolhido for o genérico o modelo irá apenas reconhecer o gesto feito ignorando a direção da palma da mão.

O treino genérico tem um grande problema que o torna inferior ao treino específico. Este problema é a grande redução dos gestos possíveis resultante de ignorar o efeito da direção da palma da mão. Isto resulta em que alguns teclados gestuais que funcionam bem com o treino específico deixem de funcionar com o treino genérico, divido a eles terem gestos em que a única coisa que os distingue é a direção da palma da mão. O treino genérico é apenas mantido para permitir comparar o seu reconhecimento de gestos, sendo preferido que o treino do modelo de reconhecimento seja feito usando o método específico.

O modelo de reconhecimento depois de treinado é guardado num ficheiro, com um nome escolhido pelo utilizador. Este ficheiro pode ser usado a qualquer momento pelo módulo para fazer o reconhecimento de gestos, sem necessitar ter de treinar de novo o modelo.O utilizador também pode escolher se quer que o modelo reconheça todas as frames ou se quer que exista um intervalo no reconhecimento das frames. Este processo neste módulo é igual ao mesmo processo no segundo módulo.

Na figura 67 é apresentado o menu do módulo de reconhecimento de gestos. O menu apresenta primeiro uma caixa de opções, onde o utilizador indica o teclado gestual, de todos os disponíveis, que quer usar para treinar o modelo. A seguir é pedido é pedido que o utilizador escreva o nome do ficheiro em que o modelo treinado será guardado. Após o nome ter sido inserido o utilizador deverá indicar se quer treinar o modelo usando o método genérico ou o específico. O utilizador também pode escolher um dos modelos de reconhecimento já treinado e fazer o reconhecimento dos gestos usando esse modelo.

O utilizador também necessita indicar se o algoritmo de aprendizagem processa todas a frames que recebe do Leap Motion ou se deve ignorar um certo número de frames. Se o algoritmo de aprendizagem não processar todas as frames que recebe do Leap Motion, o utilizador tem de indicar o número de frames, que o Leap Motion recolhe, que o algoritmo deve ignorar.

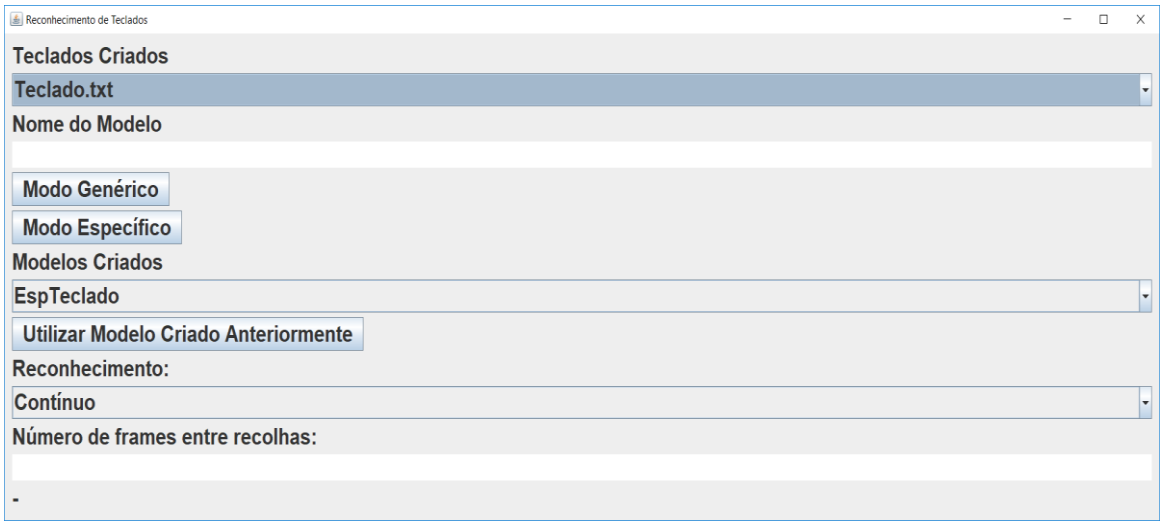


Figura 67. Menu do módulo de reconhecimento de teclados.

Após o modelo ter sido treinado ou um modelo já criado ter sido carregado, será aberta uma nova janela. Nesta nova janela o utilizador poderá fazer um gesto usando o Leap Motion. Se esse gesto estiver no teclado usado para treinar o modelo, a letra a que este gesto está conectado será escrita na caixa de texto. O utilizador poderá voltar ao menu anterior ao carregar no botão "Voltar". Na figura 68 é mostrada a janela de reconhecimento de gestos.

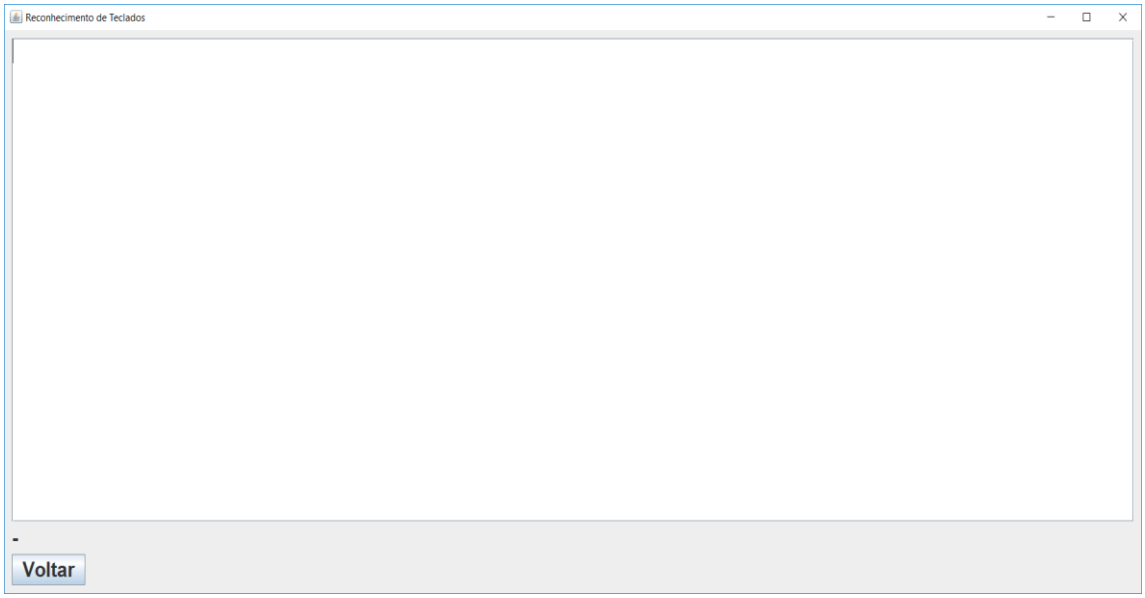


Figura 68. Área de reconhecimento de gestos.

4.2.5 Comparação de Modelos Automáticos e Modelos Manuais

A implementação do módulo de comparação de modelos automáticos e modelos manuais foi feita em java e em C#. O módulo apresenta vários métodos que permitem testar a exatidão e precisão dos modelos treinados, usando o algoritmo de aprendizagem SVM. Os gestos usados para fazer o treino podem ser criados automaticamente ou recolhidos manualmente.

O menu inicial desta parte da aplicação pergunta se o utilizador quer testar um modelo treinado usando gestos criados automaticamente pelo sistema através de um único gesto, usando o método de criação descrito no capítulo 3, ou treinar o modelo usando apenas gestos recolhidos manualmente. A figura 69 apresenta o menu inicial desta parte da aplicação.

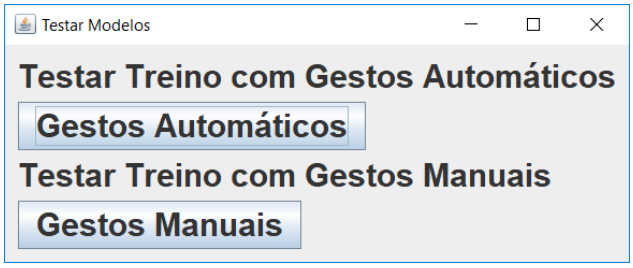


Figura 69. Escolha do tipo de modelo a ser testado.

Caso o utilizador indique que quer fazer testes ao modelo treinado usando gestos criados automaticamente, várias opções para testar o modelo vão aparecer. O utilizador pode testar o modelo usando gestos recolhidos manualmente, em que o modelo é treinado usando gestos criados automaticamente e é depois testado usando um grupo de gestos recolhidos manualmente. Para realizar este tipo de testes é apenas necessário premir a opção referente a esse tipo de testes.

O utilizador pode também testar o modelo de reconhecimento, treinado usando os gestos automáticos, através do Leap Motion. Ao escolher esta opção, o módulo começa a receber os dados dos gestos obtidos pelo Leap Motion. O módulo processa os dados, que são depois recebidos pelo modelo de reconhecimento que faz a classificação do gesto. A classificação do gesto é mostrada pelo módulo, sendo também mostrada a percentagem de certeza da classificação.

Para testar o modelo usando gestos automáticos é necessário indicar que percentagem dos gestos criados automaticamente vai ser utilizada para treinar o modelo e a percentagem que vai ser utilizada para testar o modelo. As percentagens disponíveis são:

1. 90% para treinar e 10% para testar.
2. 70% para treinar e 30% para testar.
3. 50% para treinar e 50% para testar.

Para este último tipo de testes também é necessário indicar o número de gestos que vão ser criados automaticamente. O número de gestos que serão criados deve ser igual ou maior que dez.

O gesto original usado para criar os gestos automáticos é escolhido pelo utilizador de uma lista de vários gestos recolhidos manualmente pelo utilizador. A parte da aplicação usada para criar estes gestos manuais será descrita no final deste capítulo. A figura 70 apresenta o menu dos testes para o modelo treinado usando gestos automáticos.

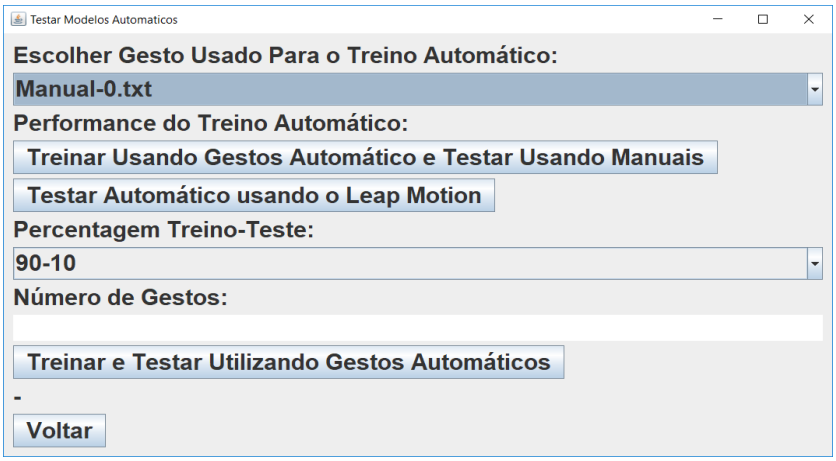


Figura 70. Testes do modelo treinado com gestos automáticos.

Caso seja indicado que o utilizador quer fazer testes ao modelo treinado com gestos criados manualmente, existem três opções para testar o modelo:

1. Testar usando gestos criados automaticamente.
2. Testar usando o Leap Motion.
3. Testar usando gestos manuais.

O processo de testar o modelo de reconhecimento, treinado usando gestos manuais, utilizando gestos criados automaticamente, é igual ao processo de testar o modelo treinado, usando gestos criados automaticamente, através de gestos recolhidos manualmente. O mesmo aplicasse ao teste do modelo usando gestos de teste obtidos, em tempo real, pelo Leap Motion.

O processo de testar o modelo, treinado usando gestos manuais, usando uma percentagem dos manuais é igual ao processo de testar o modelo, treinado usando gestos criados automaticamente, usando uma percentagem dos automáticos. A única diferença é que não é necessário indicar o número de gestos que o utilizador quer criar.

Também existe a opção de gerir os gestos manuais que são utilizados no treino do modelo. Na figura 71 é apresentado o menu com os testes do modelo treinado utilizando gestos manuais.

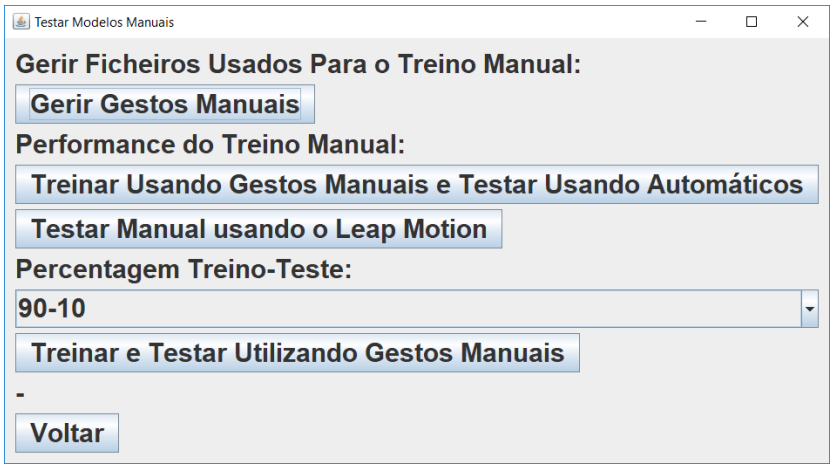


Figura 71. Testes do modelo treinado com gestos manuais.

Caso o utilizador queira adicionar ou remover um gesto recolhido manualmente, uma nova janela será aberta. Nessa janela o utilizador poderá apagar um dos gestos manuais guardados ou gravar um novo gesto manual ao indicar o nome do gesto e se é correto ou incorreto. Um gesto correto é o gesto que o utilizador quer reconhecer e um gesto incorreto é um gesto diferente do correto que permite o SVM fazer a distinção entre o gesto que queremos reconhecer e os restantes gestos. O gesto é gravado usando o Leap Motion, cinco segundos depois da ordem para guardar o gesto ser dada. Os resultados destas operações aparecem no fundo da janela. A figura 72 apresenta o menu que é usado para a gestão dos gestos recolhidos manualmente.

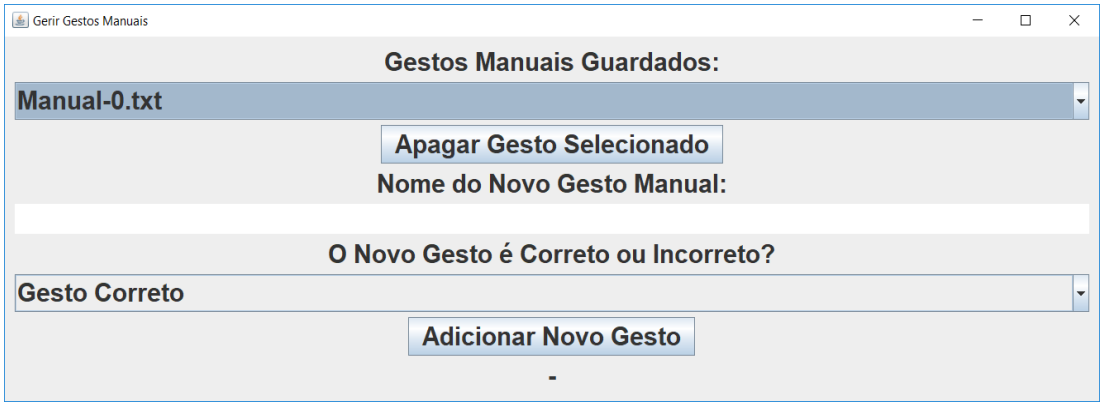


Figura 72. Menu de gestão dos gestos manuais.

5 CONSIDERAÇÕES FINAIS

5.1 CONCLUSÕES

Neste trabalho foi proposto a criação de um sistema de reconhecimento de gestos que seria capaz de reconhecer um gesto e que quando o modelo reconhecesse esse gesto o sistema iria realizar uma ação previamente definida. Este objetivo principal foi alcançado com sucesso, com os modelos treinados sendo capazes de reconhecer um gesto feito pelo utilizador, usando o sensor Leap Motion, e escrever a letra do teclado, a que esse gesto está associado, no ecrã.

A principal conclusão deste trabalho foi que o algoritmo SVM consegue criar um sistema de reconhecimento de gestos capaz de fazer uma boa separação entre gestos diferentes. Com o modelo sendo capaz de detetar perfeitamente a direção da palma da mão e conseguindo detetar o gesto sendo feito, com apenas algumas dificuldades em gestos semelhantes. A maioria dos problemas encontrados foram a nível do processamento das mãos pelo Leap Motion.

Outra importante conclusão foi que os parâmetros direcionais da palma da mão e dedos são os componentes principais para fazer o reconhecimento de um gesto. Sendo que os parâmetros posicionais não afetam positivamente os resultados das previsões do modelo, mas também podem causar que o modelo colapse e deixe de ser capaz de prever o gesto que está a ser feito.

A utilização de gestos criados automaticamente pelo sistema, com base em apenas um gesto original, para treinar o modelo de reconhecimento, também mostrou bons resultados. Os modelos de reconhecimento treinados usando os gestos automáticos tiveram resultados, em termos de exatidão e precisão, extremamente semelhantes aos obtidos usando modelos treinados com gestos manuais. E dado os gestos automáticos serem muito mais fáceis de criar do que os gestos manuais, eles são claramente superiores, em termos de utilidade, para a criação de sistemas de reconhecimento de gestos.

Foi também concluído que podemos minimizar o efeito que a direção e rotação da mão tem no gesto e o modelo continuará a reconhecer o gesto sendo feito. Mas ignorar a direção e rotação também diminui imenso o número de gestos que podem ser feitos. Isto deve-se ao facto de o mesmo gesto poder ser dividido em vários gestos, se estes dois parâmetros da mão forem considerados. Um exemplo disto é o simples gesto em que a mão está fechada poder ser divido em vários gestos, como por exemplo: mão fechada com a palma da mão virada para baixo, mão fechada com a palma da mão virada para a direita, a palma da mão virada para cima, etc. Como tal é recomendado ter sempre em conta a direção e rotação da mão quando fazendo o reconhecimento um gesto.

Os problemas que surgem no reconhecimento de gestos são resultantes de problemas com o sensor Leap Motion. O Leap Motion tem por exemplo dificuldade a reconhecer a

posição dos dedos em casos em que o dedo médio está dobrado, mas o anelar está esticado. O Leap Motion também tem a tendência de não reconhecer a mão da mesma maneira, dependo se a mão é a esquerda ou a direita.

Outro problema foi que o Leap Motion tem dificuldades a reconhecer certos gestos quando o braço tem certos ângulos, nomeadamente, gestos em que a mão ou dedos impedem o processo de localização de um dedo devido a eles estarem à sua frente. Um destes problemas é o gesto em que a mão está fechada, o polegar está esticado e a palma da mão está virada para um lado de maneira que faz a mão estar à frente do polegar, tornando impossível para o sensor saber se o polegar está levantado.

Isto deve-se ao facto de o Leap Motion ser um sensor que deteta a mão de baixo para cima, o que torna certas posições dos dedos difíceis de detetar. O Leap Motion partilha este problema com todos os sensores que utilizam apenas uma câmara para fazer o reconhecimento de gestos. Uma possível solução para isto é adicionar novos sensores que iriam detetar a posição da mão e dedos de posições diferentes da do Leap Motion.

5.2 Trabalhos Futuros

O possível trabalho futuro mais óbvio, é a implementação do reconhecimento de gestos dinâmicos. Foi notado durante a pesquisa inicial para o trabalho que a maioria das bibliotecas existentes tem vários problemas, que diminuem a sua usabilidade num sistema de reconhecimento de gestos dinâmicos. Uma possível solução para isto seria criar manualmente um método de aprendizagem para situações dinâmicas ou alterar o sistema de reconhecimento criado usando SVM de maneira que fosse capaz de reconhecer gestos dinâmicos.

Outro possível trabalho para o futuro, é adicionar um novo sensor de gestos que iria reconhecer a mão de uma posição diferente da do Leap Motion. Esta adição iria aliviar ou remover o problema de sensores que tem apenas uma câmara terem dificuldades a reconhecer gestos em que a deteção de um dedo é obstruída pela mão ou outros dedos.

Também é possível fazer um estudo acerca dos efeitos de alterar o sensor ou método de aprendizagem utilizados para criar o sistema de reconhecimento de gestos. Isto permitiria comparar a performance do Leap Motion e do SVM com outros sensores e métodos de aprendizagem. Esta comparação também permitiria descobrir se existem áreas em que outras opções de sensor ou método de aprendizagem iriam ajudar a melhorar o resultado do sistema já existente.

É também possível aprofundar a distinção entre o processamento contínuo e descontínuo das frames que o modelo de reconhecimento recebe do sensor de movimento. Isto consistiria em modificar o sistema de reconhecimento de maneira que o sistema reconhecesse alguns gestos de maneira contínua e outros gestos de maneira descontínua.

Isto seria útil, já que, em vários casos reais existem ocasiões em que é necessário diferenciar entre uma ação contínua e uma ação momentânea. Um exemplo disto é a diferença entre pressionar uma tecla e soltá-la imediatamente e pressionar uma tecla e

continuar a fazer pressão sobre ela por um certo tempo. Para o primeiro caso usar o reconhecimento contínuo não seria conveniente, já que o modelo processaria mais frames do que o utilizador quer. No segundo caso o reconhecimento descontínuo teria problemas, devido ao caso exigir que a aplicação obtenha informação continuamente, o que o reconhecimento descontínuo não consegue fazer.

6 REFERÊNCIAS

Aigner, R., Wigdor, D., Benko, H., Haller, M., Lindbauer, D., Ion, A., Zhao, S., & Koh, J.T.K.V. (2012). Understanding mid-air hand gestures: A study of human preferences in usage of gesture types for hci. Technical Report MSR-TR-2012-111.

Baum, L. E., & Petrie, T. (1966). Statistical inference for probabilistic functions of finite state markov chains. The Annals of Mathematical Statistics, 37(6), 1554–1563.

Baum, L. E. (1972). An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of a Markov Process. Inequalities, 3, 1–8.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and regression trees. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery. 2(2), 121-167.

Chang, C., & Lin, C. (2011). LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2(3).

Cortes, C. (1995). Support-vector networks. Machine Learning, 20(3), 273–297.

Du, Y., Liu, S., Feng, L., Chen, M., & Wu, J. (2017). Hand Gesture Recognition with Leap Motion.

Fujii, Y., Yamamoto, K., & Nakagawa, S. (2011). Automatic speech recognition using hidden conditional neural fields. Paper presented at the IEEE International Conference on Acoustics, Speech and Signal Processing (pp. 5036–5039), Prague, Czech Republic.

Guna, J., Jakus, G., Pogacnik, M., Tomazic S., & Sodnik, J. (2013). An Analysis of the Precision and Reliability of the Leap Motion Sensor and Its Suitability for Static and Dynamic Tracking. Sensors, 13(5), 3702-3720.

Gunawardana, A., Mahajan, M., Acero, A., & Platt, J. C. (2005). Hidden conditional random fields for phone classification. Paper presented at the Eurospeech, 9th European Conference on Speech Communication and Technology (pp. 1117-1120), Lisbon, Portugal.

Hisham, B., & Hamouda, A. (2017). Arabic Static and Dynamic Gestures Recognition Using Leap Motion. Multimedia Tools and Applications, 13(8), 337-354.

Ho, T. K. (2002). A Data Complexity Analysis of Comparative Advantages of Decision Forest Constructors. Pattern Analysis and Applications, 5(2), 102–112.

Lloyd, S. P. (1982). Least Squares Quantization in PCM. IEEE Transactions on Information Theory, 28(2), 129-137.

Khoshelham, K., & Elberink, S.O. (2012). Accuracy and resolution of Kinect depth data for indoor mapping applications. Sensors, 12(2), 1437-1454.

Karam, M., & Schraefel, M.C. (2005). A Taxonomy of Gestures in Human Computer Interactions. Technical Report ECSTR-IAM05-009. Electronics and Computer Science, University of Southampton.

Khan, R. Z., & Ibraheem, N. A. (2012). Hand Gesture Recognition: A Literature. International Journal of Artificial Intelligence & Applications, 3(4), 161-174.

Kumar, P., Saini, R., Behera, S. K., D. Dogra, D., & P. Roy, P. (2017). 3D text segmentation and recognition using leap motion. Multimedia Tools and Applications, 76(15), 16491-16510.

Kumar, P., Saini, R., Behera, S. K., D. Dogra, D., & P. Roy, P. (2017). Real-time recognition of sign language gestures and air-writing using leap motion. Paper presented at the Fifteenth IAPR International Conference on Machine Vision Applications (pp. 157-160), Nagoya, Japan.

Lu, W., Tong, Z., & Chu, J. (2016). Dynamic Hand Gesture Recognition With Leap Motion Controller. IEEE Signal Processing Letters, 23(9), 1188-1192.

Marin, G., Dominio, F., & Zanuttigh, P. (2014). Hand gesture recognition with leap motion and kinect devices. Paper presented at the 2014 IEEE International Conference on Image Processing (pp. 1565-1569), Paris, France.

Marin, G., Dominio, F., & Zanuttigh, P. (2016). Hand gesture recognition with jointly calibrated Leap Motion and depth sensor. Multimedia Tools and Applications, 75(22), 14991-15015.

Maruyama, K., Shin, J., Kim C., & Chen, C. (2017). User Authentication using Leap Motion. Paper presented at the International Conference on Research in Adaptive and Convergent Systems (pp. 213-216), Krakow, Poland.

Nowicki, M., Pilarczyk, O., Wasikowski, J., & Zjawin, K. (2014). Gesture recognition library for leap motion controller. Poznan University of Technology. Poznan, Poland.

Oliver, M., Simarro, F. M., Molina, J. P., González, P., & Fernández-Caballero, A. (2016). Multi-camera systems for rehabilitation therapies: a study of the precision of Microsoft Kinect sensors. Frontiers of Information Technology & Electronic Engineering, 17(4), 348-364.

Sarkar, A. R., Sanyal, G., & Majumder, S. (2013). Hand Gesture Recognition Systems: A Survey. International Journal of Computer Applications, 71(15), 26.37.

Scheffer T., Decomain C., & Wrobel S. (2001). Active Hidden Markov Models for Information Extraction. Paper presented at the Advances in Intelligent Data Analysis, 4th International Conference, IDA 2001, Cascais, Portugal. Berlin, Heidelberg: Springer.

Simos, M., & Nikolaidis, N. (2016). Greek sign language alphabet recognition using the leap motion device. Paper presented at the 9th Hellenic Conference on Artificial Intelligence (pp. 34:1-34:4), Thessaloniki, Greece.

Sung, Y.-H., Boulis, C., Manning, C. D., & Jurafsky, D. (2007). Regularization, adaptation, and non-independent features improve hidden conditional random fields for phone classification. Paper presented at the IEEE Workshop on Automatic Speech Recognition & Understanding (pp. 347-352), Kyoto, Japan.

Wasenmüller, O., & Stricker, D. (2016). Comparison of Kinect V1 and V2 Depth Images in Terms of Accuracy and Precision. Paper presented at the ACCV Workshops 2016 (pp. 34-45), Taipei, Taiwan.