UNIVERSIDADE DO ALGARVE

INTERFACE NATURAL BASEADO NO MOVIMENTO DA CABEÇA E OLHOS

João Miguel da Silva Martins

Dissertação para obtenção do grau de
Mestre em Engenharia Elétrica e Eletrónica
Área de Especialização: Sistemas de Energia Elétrica e Controlo

Trabalho realizado sob a orientação de:
Professor Doutor João Miguel Fernandes Rodrigues

2015

INTERFACE NATURAL BASEADO NO MOVIMENTO DA CABEÇA E OLHOS

Declaro ser o autor deste trabalho, que é original e inédito. Os autores e trabalhos consultados estão devidamente citados no texto e constam na listagem de referências bibliográficas incluída.

(João Miguel da Silva Martins)

Copyright

A Universidade do Algarve tem o direito, perpétuo e sem limites geográficos, de arquivar e publicitar este trabalho através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, de o divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objetivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

Faro, 15 de Janeiro de 2015

Agradecimentos

Agradeço à minha família todo o apoio que me deram ao longo desta dissertação, e de toda a minha vida académica. Agradeço toda a atenção, apoio e paciência prestados pelo Professor João Rodrigues ao longo desta dissertação.

Muito Obrigado.

NOME: João Miguel da Silva Martins
INSTITUIÇÃO: Instituto Superior de Engenharia
ORIENTADOR: Professor Doutor João Miguel Fernandes Rodrigues
DATA: Janeiro de 2015
TÍTULO: Interface natural baseado no movimento da cabeça e olhos

Resumo

Muitas vezes as pessoas procuram a sua liberdade através do meio virtual. No entanto, nem todos têm a possibilidade de interagir com o computador da mesma forma. As pessoas com maiores dificuldades motoras acabam por não ter a mesma liberdade que as pessoas totalmente capacitadas têm para mexer no rato, no teclado, no *touchscreen*, etc. Dado que os equipamentos eletrónicos estão espalhados por todos os ramos profissionais, a necessidade de interagir com um computador é uma capacidade imprescindível, mas com todos os problemas económicos que as famílias e as empresas suportam, e com os cortes nos apoios governamentais aos portadores de deficiências motoras, mostra-se imperativo o desenvolvimento de interfaces de baixo custo que permitam efetuar essa interação homem-máquina. Por outro lado o reaproveitamento de sensores, tais como o Microsoft Kinect de consolas de jogos desatualizadas, podem ser utilizados para fazer estes interfaces e trazer alguma liberdade e independência às pessoas de mobilidade reduzida. Esta dissertação foca-se essencialmente no desenvolvimento de uma interface que permita o controlo do computador sem a necessidade de um teclado, rato, ou *touchscreen*, fácil de utilizar e de código aberto, para que aqueles que tiverem possibilidade e o pretendam, a possam melhorar e adaptar a seu gosto. O interface é baseado no sensor Kinect, e permite o controlo do computador apenas pelo movimento da cabeça/olhar e abertura/fecho da boca, complementarmente está também disponível comandos por voz. A solução apresentada nesta dissertação traduz assim uma ferramenta de baixo custo, capaz de ser utilizada no quotidiano, que proporciona novos caminhos para quem sofre mobilidade reduzida. De notar que existem, no entanto, soluções comercialmente disponíveis para o efeito mas o seu preço torna-se um verdadeiro obstáculo em tempos de crise, por outro lado as soluções gratuitas acabam por pecar na liberdade de movimento, apesar do preço ser reduzido. Este sistema pretende colmatar estas desvantagens.

Palavras-chave: Acessibilidade; Tecnologias Assistivas; *e-Inclusion*; Seguimento da cara; Interação Homem-Máquina; Microsoft Kinect.

TITLE: Natural interface based on movement of head and eyes

Abstract

Sometimes people look for freedom in the virtual world. However, not all have the possibility to interact with the computer in the same way. In addition, nowadays, almost every job requires interaction with computerized systems. People with physical impairments do not have the same freedom to control the mouse, keyboard or touchscreen. In the last years, some of the government programs to help people with reduced mobility suffered a lot with world economic crisis. Some of these programs have been shut down to reduce costs. This dissertation focuses on the development of a human-machine interface, which allows a touchless computer's control, avoiding the use of keyboard, mouse or touchscreen. By reusing Microsoft Kinect sensors from old videogames, a reduced cost, easy to use, and open-source (to allow further development) interface was developed, which allows controlling the computer by only moving the head/eyes and mouse, complementary sound commands can also be used. Nevertheless, there are several similar commercial solutions, but they are so expensive that their price tends to be a real obstacle in their buying, in the other hand, the free solutions usually do not offer the freedom that people with reduced mobility needs. The present solution try to address these drawbacks.

Keywords: Accessibility; Assistive Technologies; e-Inclusion; Facetracking; Human-Computer Interaction; Kinect sensor.

Índice

1 Introdução

A Vida do ser humano na sociedade de hoje está totalmente dependente de equipamentos eletrónicos. Onde quer que uma pessoa se dirija encontrará, basicamente, um computador com o qual terá de interagir.

Com o aparecimento de serviços prestados através da Internet, os utilizadores têm vindo a tornar-se dependentes da informática. Atualmente, considera-se a falta de conhecimento em operar com um computador, como sendo uma incapacidade técnica da pessoa. Isso acontece com pessoas que devido à sua vida, ou não quiseram ou não tiveram possibilidade de aprender. No entanto, para as pessoas com deficiências motoras a informática que deveria ser uma ajuda, começou por revelar-se muitas vezes um obstáculo, mas ao mesmo tempo uma solução para outros problemas.

Por exemplo, as pessoas que sofrem de Esclerose Lateral Amiotrófica (*Amyotrophic Lateral Sclerosis* - ALS) perdem o controlo do seu corpo, ficando incapacitados de se movimentarem, e totalmente dependentes de terceiros. Mas não perdem as suas funções cerebrais. Do ponto vista mental, são pessoas completamente conscientes, presas no seu próprio corpo imóvel. É uma doença que não escolhe idade, raça ou género, e até ao momento, sem causa ou cura conhecida. O músico português Zeca Afonso foi diagnosticado em 1982, falecendo em 1987. Um dos mais consagrados físicos teóricos da atualidade, Stephen Hawking, equiparado a Isaac Newton, sofre desde dos seus 21 anos com esta doença (atualmente tem 72 anos). Mas, consegue mexer os olhos, e é esta a forma como ele comunica com o mundo exterior. Utiliza um sistema que lhe permite controlar um computador através do seu olhar.

1.1 Enquadramento e objetivos

Pretende-se nesta dissertação criar uma interface homem-máquina robusto, de baixo custo e *open source*, que permita o controlo do computador (todas as suas funcionalidades) apenas com a combinação de movimentos da cabeça (direita, esquerda, cima e baixo), movimentos do olhos (pálpebras) e boca, para utilizadores com mobilidade condicionada, mas no entanto consigam mover a cabeça, incluindo utilizadores seniores que tenham limitações no uso do tradicional rato e teclado. O

interface deve ser o menos intrusivo possível.

Mais especificamente pretende-se atingir os seguintes objetivos:

(a) Controlar o rato usando o movimento da cabeça, i.e., emular a utilização de um dispositivo apontador como o rato nas suas funcionalidades;

(b) Controlar um teclado virtual usando movimentos da cabeça;

(c) Aplicar o interface desenvolvido a situações reais e a utilizadores reais.

1.2 Contribuição

Desenvolvimento de uma interface que permita o controlo do computador sem a necessidade de um teclado, rato, ou *touchscreen*, fácil de utilizar e de código aberto, para que aqueles que tiverem possibilidade e o pretendam, o possam melhorar e adaptar a seu gosto. O interface é baseado no sensor Kinect, e permite o controlo do computador apenas pelo movimento da cabeça olhar/pálpebra e abertura/fecho da boca. A solução apresentada foi testada com utilizadores reais e traduz-se numa ferramenta de baixo custo, capaz de ser utilizada no quotidiano, que proporciona novas oportunidades para quem sofre mobilidade reduzida.

O interface e o código-fonte estão disponíveis para download em https://github.com/MigMart/Usertracking.

Submissão de uma publicação a conferência: Martins, J.M.S., Rodrigues, J.M.F. (2015) *An natural human-cumputer interface based on movement of head and eyes*. Submitted to the 6th Int. Conf. on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2015), June 10-12, 2015, Fraunhofer FIT, Sankt Augustin, Germany.

1.3 Vista geral

No presente Capítulo foi introduzido o tema da dissertação bem como os objetivos pretendidos e principais contribuições. No Capítulo 2, apresenta o estado de arte, serão apresentadas algumas das soluções comerciais já existentes, comparando as suas características funcionais e preços de custo. Serão também abordadas algumas técnicas utilizadas em localização facial, posicionamento tridimensional (3D), extração de características faciais e seguimento da face. No Capítulo 3 está descrito a constituição e o desenvolvimento do interface, tanto em termos de hardware como de software. Será também descrito pormenorizadamente os métodos e estratégias implementados para

seguimento da face, eliminação de ruídos dos sensores e calibração do sistema. No Capítulo 4, será descrito o funcionamento do protótipo desenvolvido, i.e., será descrito os passos necessários para a montagem e utilização da solução.

No Capítulo 5 estão os resultados dos ensaios realizados e as opiniões dadas pelos voluntários. Para finalizar, o Capítulo 6, contempla as considerações finais e propostas a melhorias futuras ao sistema desenvolvido.

2 Estado da arte

Neste capítulo será descrito em resumo, discutido e analisado algumas das técnicas estudadas e necessárias ao desenvolvimento da dissertação. Nomeadamente técnicas de: (a) deteção de face, de (b) extração de caraterísticas faciais, de (c) deteção da pose da face em 3D e (d) medição da profundidade numa cena, bem como ainda são apresentados e analisados (e) produtos similares existentes no mercado.

2.1 Deteção de face

A deteção da face dos utilizadores é utilizada em várias áreas, tais como biométrica, Existem vários algoritmos que fazem deteção da face do utilizador. A titulo de exemplo, o algoritmo Viola-Jones [1] é um algoritmo disponível em várias *frameworks* de computação visual, tais como OpenCV [2] e Processing [3]. Pode portanto ser utilizada em várias linguagens de programação: C, C++, C#, Python, Java, Javascript [4]. Também existem algumas implementações em Action Script e Matlab [4]. É um algoritmo que deteta faces em imagens, assim como outro tipo de objetos, desde que treinado para isso. O treino é um processo altamente consumidor de recursos computacionais, e demora horas ou dias, dependendo da margem de erro que se pretenda ter. O resultado é guardado em ficheiro .xml, e utilizado em qualquer aplicação. Essencialmente aplica Haar Wavelets a várias zonas da imagem. Vai progredido à medida de que estes se encaixem no resultado previsto. A sua execução é rápida, e pode ser aplicada quase em tempo real. Melhores resultados podem ser obtidos se recorrer à versão AdaBoost da Microsoft, denominada como FloatBoost [5].

A análise estatística Schneiderman-Kanade [6] utiliza histogramas de imagens contendo faces e outras sem faces. As imagens que contêm faces são separadas consoante a perspetiva da face. Aplica a transformada Wavelet discreta e associa o resultado a cada uma das imagens. Este método tem que ser treinado. O resultado do treino pode depois ser aplicado, mas segundo os tempos de resposta descritos em [6], o algoritmo não consegue dar uma resposta em tempo real. Em [7], o autor mostra como este algoritmo pode ser otimizado para diminuir a margem de erro, e dá a indicação de que também consegue detetar outro tipo de características, tais como boca e olhos. A distância de Hausdorff [8, 9, 10] tem sido utilizada para fazer deteção de caras e objetos [11] em imagens unicamente pela forma. Na deteção de caras, determina a semelhança

de uma forma geométrica elíptica às arestas detetadas em imagens binárias, calculando a distância entre cada ponto da elipse e da aresta detetada. Existem muitos mais técnicas e algoritmos para efetuar a deteção da cara, ver por exemplo [12, 13].

2.2 Deteção de características faciais

A extração de caraterísticas faciais, tais como o canto dos olhos, os cantos da boca é um passo importante a dar no sentido de redução do tempo de resposta. Deste modo, consegue-se reduzir da área da imagem a analisar, especialmente quando se pretende detetar o estado da boca ou dos olhos – abertos ou fechados. Existem várias técnicas, e.g., técnicas baseadas na geometria, por segmentação da cor e por modelos probabilísticos.

Nas técnicas baseadas na geometria, aplicam-se detetores de arestas como o Canny [14, 15]. A utilização combinada de métodos Adaboost, *Local Binary Pattern* (LBP), redes neuronais e transformadas Wavelet tal como em [16, 17, 18, 19], são algumas das soluções que se baseiam na geometria para fazer deteção das características. As técnicas por segmentação da cor procuram isolar as regiões da cara que não apresentem cor de pele [20]. É nessas regiões que se procuram as características. A deteção da pele pode ser realizada mediante uma certa escala de cor [21]. Algumas destas técnicas são altamente sensíveis à variação da iluminação. Portanto, a escolha do modelo de cor (RGB, HSL, YcbCr) é muito importante. Ainda assim procura-se aplicar métodos de classificação um pouco mais complexos como *Skin Probability Maps* (SPM) [22] e *Support Vector Machine* (SVM) [23, 24] para tornar a procura mais dinâmica e menos suscetível à iluminação.

Os modelos probabilísticos como Modelo de Forma Ativa (*Active Shape Model*-ASM) [25] baseiam-se na comparação de uma forma média (parametrizada) da característica, que é ajustada (por transformação Euclidiana) iterativamente, ao longo da região da cara. Desse ajuste resulta uma classificação (diferença entre o modelo ajustado e a imagem). A deteção da característica é obtida consoante a análise da classificação obtida. A probabilidade da classificação obtida estar correta é realizada pelo método de Análise Discriminante Linear de Fisher (*Fisher Linear Discriminant Analysis* - FLDA) [26, 27]. Mais uma vez existem técnicas e algoritmos recentes que efetuam e usam a deteção de características faciais, como por exemplo, [28, 29].

2.3 Deteção da pose da face

Alguns dos algoritmos que têm sido investigados para a deteção da pose da face são o algoritmo de Otimização por Enxame de Partículas (*Particle Swarm Optimization*-PSO) é aplicado em diversas áreas científicas e sociais. Este algoritmo foi apresentado por Kenneth e Eberhart [30] para fazer a otimização de funções não-lineares e treino de redes neuronais. Foi deduzido a partir do modelo computacional que visa a simulação de comportamentos de indivíduos (pessoas, animais, partículas) que trabalham em grupo para satisfazer um objetivo.

A aplicação deste algoritmo na estimativa da pose facial [31], visa o aumento da precisão e da tolerância às oclusões parciais da face. Neste caso, faz com que todas as partículas se desloquem até atingirem a velocidade nula, isto é, até atingirem a posição que melhor se apresenta como solução (a melhor estimativa de pose). A informação proveniente das partículas vizinhas é importante na determinação da velocidade e orientação durante o movimento de cada partícula. Todas as partículas têm velocidade e posição iniciais aleatórias, e ao longo do movimento também são influenciadas por fatores aleatórios, nomeadamente os componentes cognitivos e sociais. O algoritmo apresenta-se como tolerante à variação do tamanho e distância da face, às expressões faciais e a poses oblíquas.

A triangulação de marcas faciais pode também ser realizada por meio de sensores de profundidade ou comparação estéreo [32, 33, 34]. Estas marcas são as pupilas, cantos dos olhos, cantos da boca e ponta do nariz. Liang *et al.* [35] apresentaram um algoritmo baseado numa transformação afim. Consiste na aproximação entre a forma circular e a posição frontal da cara, e a forma elíptica com a posição oblíqua da cara. A circunferência pode ser transformada numa elipse, e através dos parâmetros dessa transformação (afim) obtidos iterativamente, consegue-se determinar a pose da cara. Mas, a aproximação da face a um plano é uma aproximação grosseira, como tal, o resultado é refinado e otimizado [36, 37].

Um Modelo de Aparência Ativa (*Active Apearance Model* - AAM) [38, 39] consiste num conjunto de equações definidas pela combinação de um modelo de forma ativa (*Active Shape Model* - ASM) [40] com um modelo de textura, isto é, o AAM dá importância à forma e à textura do objeto a modelar. É necessário criar um modelo a partir de uma base de dados com imagens bidimensionais (2D) das poses do objeto. Estas imagens devem estar classificadas (com a orientação 3D) de forma que o

algoritmo de treino correlacione os componentes principais da imagem de forma correta. Outros exemplos de algoritmos de deteção da pose da face são [41, 42].

2.4 Medição da profundidade numa cena

A medida da profundidade de uma cena 3D ou objeto particular pode ser realizada aplicando algumas das técnicas, tais como: cálculo do tempo de voo (*Time-of-Flight* – ToF), deslocamento de fase (*Phase-Shift*), distorção de luz estruturada e visão estéreo.

Na literatura encontram-se soluções de modelação 3D de um objeto particular utilizando sensores baseados no ToF [43, 44]. O ToF é o intervalo de tempo entre a emissão de um sinal (pulso) e a receção do mesmo. Sabendo a velocidade de propagação desse sinal, e o intervalo de tempo, é possível estimar a distância percorrida pelo sinal. Este tipo de sensores podem ser utilizados para varrer uma superfície e fazer *scan* 3D da mesma, ou simplesmente medir uma distância. Um LIDAR é um equipamento capaz de emitir um feixe laser, e captar a sua reflexão numa superfície remota, determinado a distância através do ToF [45]. Gueuning *et al.* [46] apresentaram uma combinação do método de deslocamento de fase com o método do tempo de voo. Angrisani *et al.* [47] propõe a medição de profundidade com um feixe ultrassónico. Determinam o ToF, e filtram o ruído recorrendo a um Filtro de Kalman discreto.

Um outro método alternativo ao ToF é o da medição da distorção de um padrão (estruturado) de luz (infravermelho) refletido [48]. O sensor largamente conhecido sensor Kinect da Microsoft utiliza este método para medir a profundidade da cena [49].

O método de visão estéreo [50, 51] consiste em capturar duas imagens de ângulos (conhecidos) diferentes. Essas duas imagens terão o mesmo ponto (um objeto) presente, mas estará desfasado de uma imagem para a outra. Esse desfasamento é proporcional à distância entre as câmaras e a distância focal das câmaras utilizadas. Usualmente utilizam-se duas câmaras iguais (para terem a mesma distância focal), e ficam sobre o mesmo eixo horizontal e à "mesma distância" do ponto de referência. Juntando a informação de profundidade, forma e textura é possível detetar um, ou vários utilizadores, e acompanhar os seus movimentos (postura, reconhecimento de movimentos específicos, etc.). Esta prática tem sido aplicada recentemente em interfaces de acessibilidade inovadoras, como por exemplo para leitura labial [52].

Existem no mercado produtos que utilizam as técnicas referidas nas secções anteriores para implementar o controlo do computador, na secção seguinte vamos

analisar resumidamente algumas das soluções existentes no mercado.

2.5 Produtos similares existentes no mercado

Atualmente podem encontrar-se soluções para controlo de computador tanto grátis (pressupondo de que o utilizador já possua computador com webcam) como pagas (comercializadas). Nesta secção, apresentam-se alguns exemplos tentando mostrar as variantes das várias soluções. Inicialmente, abordam-se alguns dos sistemas grátis e finalizam-se com alguns exemplos comercializados atualmente.

Chen *et al.* [17] desenvolveu um sistema que permitia ao utilizador controlar o rato do computador com o movimento da cabeça. Era composto por um sensor de posição fixado na cabeça do utilizador, ligado a um módulo que fazia o processamento do sinal e transmitia-o ao controlador principal (microprocessador Intel 8951). Este sistema destinava-se a ser utilizado por pessoas portadoras de deficiência, e tinha por objetivo ajudá-las a controlar um computador em termos profissionais, de modo a conseguirem uma vida independente.

Rajaee *et al.* [53] desenvolveu em 2004 um projeto que permitiu a pessoas com deficiência motora acederem à Internet usando um interruptor num computador ou um *Personal Digital Assistant* (PDA). Através de Software específico, a pessoa conseguia navegar e enviar emails utilizando interruptores. O projeto era composto por vários elementos: um módulo de interruptores, um braço mecânico que segura o PDA, o próprio PDA, e o software que permitia a navegação pela Internet.

Connor *et al.* [54] descrevem o seu sistema – *Camera Mouse* – como um substituto ao rato convencional, destinado especialmente a utilizadores com dificuldades motoras. É um Software que funciona com qualquer câmara convencional ligada ao computador. Baseia-se no seguimento de um ponto facial. Em caso de oclusão desse ponto, consegue recuperar automaticamente sem necessidade de re-calibração. Este software é gratuito e está disponível para download na web, consultar também [55].

O produto comercial Dynavox EyeMax System [56] é um sistema criado pela Dyanvox Meyer-Johnson especialmente para as pessoas com mobilidade reduzida, ou nula, e que apenas controlam conscientemente o movimento dos seus olhos. Foi possível obter algumas características a partir do web site do fabricante [56], assim como o preço no web site de um fornecedor [57]. O sistema é constituído por um mini computador (*panel PC*) com um ecrã táctil de 12.1" um CPU Intel Atom 1,6GHz e 2GB

de RAM, e um sensor composto por uma câmara a cores, e outra de profundidade. Tem tamanho de 250 mm x 320 mm x 76 mm (milímetros) e pesa cerca de 2.86 kg (quilogramas). Possui bateria interna carregável de ion-lítio (inclui carregador), com um software especialmente adaptado para utilizadores deficientes. O Software permite a interação utilizando a estratégia que melhor se adapta ao utilizador (símbolos, fotografias, palavras e/ou letras). Possui várias vozes sintetizadas de alta qualidade. Possui Bluetooth e Wifi integrados para facilitar a ligação à Internet, email e troca de mensagens de texto. Para além disso, também possui emissor de infravermelhos configurável. Sendo devidamente configurado, o último permite o controlo básico de televisão, rádio, ar-condicionado ou qualquer outro aparelho com telecomando por infravermelhos. Vem de origem com teclado e rato USB, mala, caneta *styles*, manuais,entre outros acessórios. O preço praticado pela Toby Churchill Ltd ronda os £9,990.00 (cerca de 12.287€ e ainda acresce IVA e portes de envio).

Uma outra solução comercial para quem tem dificuldade em controlar o rato do computador é o Quha Zono Mouse [58]. É uma solução *plug'n'play*, sem fios, que permite através dos movimentos do utilizador, mover o cursor no ecrã, selecionar, clicar e arrastar itens no computador. O dispositivo pode ficar agarrado à cabeça (v.d. [58]), pulso, pé ou qualquer outra parte do corpo. Baseia-se no mesmo principio de um joystick. Inicialmente é definido um ponto de referência (*Home*) e à medida que o sensor se afasta desse ponto, o equipamento dá indicação ao cursor do rato para se mover proporcionalmente. Vem acompanhado de carregador e recetor USB para conectar no computador, adaptador para ligar dois interruptores extra, entre outros acessórios. O preço do produto mais completo ronda os 899$ (cerca de 708€), excluindo IVA e portes de envio.

A Visual Interaction GmbH (sediada na Alemanha) disponibiliza a solução myGaze Assistive Eye Tracker como uma alternativa à utilização convencional do rato para controlo do computador. Este sistema está preparado para ser utilizado em conjunto com ecrãs de 10” a 22”. O sensor é colocado na frente do monitor [59], e liga ao computador por cabo USB. O dispositivo que vem acompanhado de um software para instalar em MS Windows, e através deste, é realizada a calibração e configuração de todo o sistema. O preço definido no web site do fabricante [59] só para o sensor é de 499€. O fabricante também disponibiliza um pacote de desenvolvimento constituído por uma API (*Application Programming Interface*) e um sensor. Com a API este sensor

pode ser integrado noutros sistemas. Os preços desse pacote de desenvolvimento variam entre os 790€ e os 5.890€ (excluindo impostos e despesas de transporte).

A Xcessity Software Solutions desenvolveu um software – KinesicMouse [60] - que utiliza o sensor Kinect para simular a utilização do rato convencional do computador. Permite simular os cliques simples e duplos com determinados movimentos faciais. Esta solução não requer qualquer apetrecho no corpo do utilizador, o que lhe dá toda a liberdade de movimento desde que se encontre no campo de visão do sensor. Basta o utilizador sentar-se à frente do sensor, e ao fim de apenas alguns segundos faz a deteção do utilizador. E a partir desse momento basta simplesmente mover a cabeça e fazer alguns movimentos faciais para controlar o rato do computador em pleno. Este software está disponível para download em [60] e a licença de utilização tem um preço de 349€. Oferece 15 dias de utilização gratuita e para isso requer uma ligação à Internet. A licença de utilização não inclui o custo de aquisição do sensor, pelo que este terá que ser adquirido à parte, por um preço que ronda os 200€ (com IVA incluído). No total, esta solução fica em cerca de 549€.

Em comparação de preço com as outras soluções comerciais, esta acaba por ser a mais barata, e em termos de controlo do rato do computador é suficiente. Fica no entanto uma solução mais cara do que as soluções *freeware* baseadas em webcam, mas permite uma maior liberdade de movimento, graças ao componente de medição de profundidade utilizado na Kinect. Este sistema baseado na Kinect, graças ao seu preço e à tecnologia que disponibiliza deve ser tomada em consideração para o interface que se pretende desenvolver.

De notar que apesar do preço de referencia da Kinect rondar os 200€, uma Kinect em segunda mão pode ser encontrada facilmente a preços que rondam os 50€, sendo que se pretende desenvolver o Software e disponibiliza-lo gratuitamente, o protótipo a implementar (e a ser usado pelo utilizador final) usando esta tecnologia rondaria os 50-100€, que será o preço de uma Kinect em 2 mão, envio da mesma e mais algum equipamento necessário como cablagem e suporte [61] para a Kinect.

3 Desenvolvimento do interface

O interface deverá simular a utilização de um dispositivo apontador como o rato. Dessa forma, um utilizador pode aceder a todas as funcionalidades de software existentes num computador pessoal que tenha o Microsoft Windows instalado. O sistema é composto por duas componentes, uma em Software, que consiste no desenvolvimento de uma aplicação e outra em Hardware, que essencialmente consiste na escolha e ligação do sensor a ser utilizado, as duas componentes em conjunto formam o protótipo do sistema. A escolha do Hardware é apresentada na secção seguinte, tendo as restantes dedicadas a explicação dos pontos fundamentais no desenvolvimento do Software.

3.1 Seleção do sensor

A escolha do Hardware para este tipo de sistema é definida em função das funcionalidades necessárias, do seu custo de aquisição e na simplicidade de montagem e utilização. Podem considerar-se algumas hipóteses tais como:

a) uma única câmara RGB[1] (designação usual dada a uma câmara que adquire imagens a cores) junto ao monitor;
b) uma única câmara RGB junto à cara do utilizador;
c) as hipóteses a) e b) em simultâneo;
d) duas câmaras RGB junto ao monitor e nenhuma junto da cara do utilizador;
e) uma único sensor RGB-D[2], por exemplo, Microsoft Kinect junto ao monitor.

Em todos os casos, câmara RGB, ou sensor RGB-D existem inúmeras variantes de equipamentos afins. Estando esta dissertação vocacionada para o desenvolvimento de um sistema de baixo custo, todas as características que vão ser analisadas de seguida, são de equipamentos genéricos de baixo custo. Para o caso de câmaras RGB, vão ser analisadas as câmaras usualmente designadas por webcam. Relativamente ao sensor RGB-D, também existem no mercado vários, os mais conhecidos serão o Microsoft Kinect [62] e o Asus Xtion-Pro [63], embora a Kinect seja aquele que se consegue

1 RGB é a abreviatura do sistema de cores aditivas formado por Vermelho (*Red*), Verde (*Green*) e Azul (*Blue*). O propósito principal do sistema RGB é a reprodução de cores em dispositivos eletrónicos como monitores de TV e computador, *datashows*, câmaras digitais, etc.

2 RGB-D complementa a informação disponível através do RGB com mais a informação da profundidade (*Depth*).

encontrar a mais baixo preço (principalmente em segunda mão), por ser o mais utilizado universalmente.

A comparação entre as várias hipóteses a) a e) pode ser realizada mediante as funcionalidades e características necessárias: (1) boa qualidade de imagem em vários ambientes de luz, (2) máximo *frame rate* disponível, (3) consumo de CPU em termos de processamento, (4) largura do plano de vista (*field view*) e (4) permitir medir distâncias da câmara/sensor aos objetos. Foi realizado um resumo comparativo entre várias hipóteses e foi criada a Tab. A.1 no Anexo A.

Da comparação, chegou-se à conclusão que a melhor solução é a hipótese e) que consiste numa único sensor Kinect posicionada junto ao monitor. O facto de um único equipamento reunir uma câmara RGB com uma qualidade de imagem suficiente para o que se pretende desenvolver, um sensor que permite obter a distância (profundidade) a que cada ponto da imagem RGB se encontra, um SDK (*Software Development Kit*) que está a ser atualizado constantemente pela Microsoft, e a um preço bastante reduzido (comparativamente a outros equipamentos de gama profissional), faz desta a escolha mais viável para o sistema proposto.

Para além disso, também estão disponíveis quatro microfones internos que em conjunto com a SDK permite o reconhecimento de voz. A Fig. 1 mostra um esquema da Kinect e seus componentes.

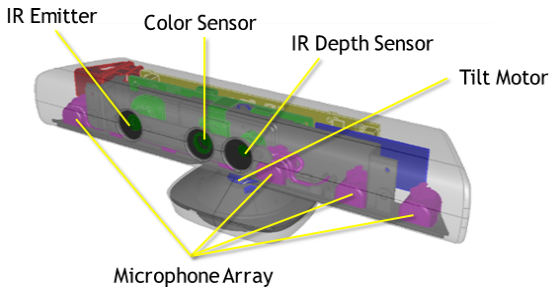


Figura 1: Componentes da Kinect, adaptado de [62].

A Kinect, tanto na versão para Windows, tal como na versão XBOX 360, apresenta algumas limitações ao nível do Hardware, nomeadamente os ângulos de vista (*field view*) da câmara e a gama de distância (profundidade) que o sensor de infravermelho (IR) consegue medir. Segundo a página de especificações da Kinect [61], a câmara (ambas as versões) possui um ângulo de visão de 53º (graus) na vista horizontal e 43º na

vertical. Para além disso, o motor acoplado à base permite regular a inclinação vertical entre -27º e +27º. O que diferencia as duas versões da Kinect é o modo de operação, pré-definido na inicialização do equipamento em modo "próximo" ou "normal". Só a Kinect for Windows aceita operar no modo "próximo" (na documentação é referenciado como *near mode)*. A versão XBOX 360 só aceita operar no modo "normal" (pré-definido). O modo "próximo" é especialmente útil nas aplicações de *face tracking*, em que o utilizador se encontra sentado a uma distância próxima da Kinect (o que permite ativar a opção *seated mode* no reconhecimento do esqueleto). Em modo "normal", a gama da distância medida varia entre 0.8 m (metros) e 4.0 m. E em modo "sentado", a distância mínima passa a ser 0.4 m, e a máxima fica em 3.0 m. A Fig. 2 ilustra as propriedades da Kinect acima referidas (distância à câmara e ângulos de visão) .

Na Fig. 3 tem um esquema do seguimento do utilizador pela forma do seu corpo (esqueleto) - *Skeleton Tracking* [63], que permite compreender melhor a relação entre o campo de visão vertical e de que forma limita o reconhecimento do utilizador, consoante a distância a que se encontra.

A Kinect consegue detetar o esqueleto do utilizador (na vertical) com 1.8 m de altura a cerca de 3.0 m. Apesar de não estar explicitamente indicado na documentação fornecida pela Microsoft, verificou-se durante os ensaios com o equipamento - versão XBOX 360 – é capaz de detetar o utilizador sentado a cerca de 1.0 m, bastando para isso ativar o modo "sentado" (*seated mode*) na opção de *TrackingMode* (mesmo estando em modo "normal"). . Apesar desta funcionalidade não apresentar qualquer vantagem para os utilizadores mais ativos, acaba por favorecer utilizadores com maior dificuldade de movimento. Isto deve-se ao facto de que o seguimento das várias partes do corpo requer sempre algum movimento das mesmas, o que pode apresentar-se como um obstáculo para utilizadores com dificuldade de movimento nos membros inferiores (pernas). Quando a opção de "seated mode" está ativa, só é realizado o seguimento dos membros superiores.

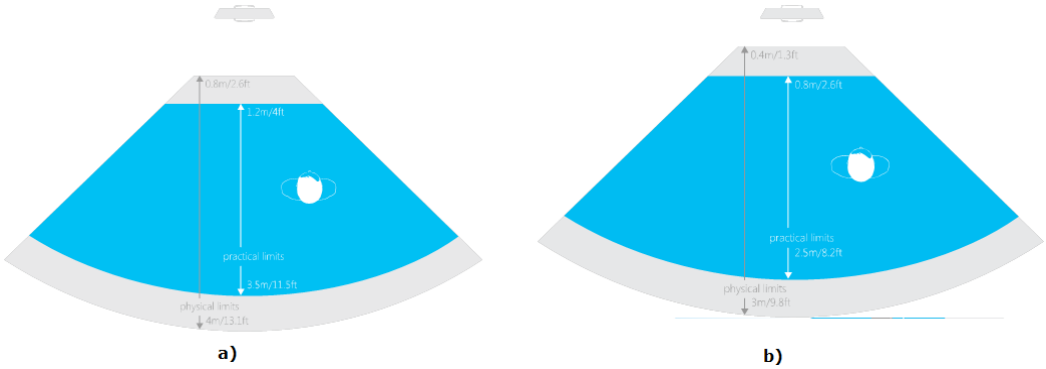


Figura 2: Distância medida em a) modo normal e b) modo sentado, adaptado de [62].

Resumindo, a Kinect dispõe internamente de um sensor de profundidade que utiliza infravermelhos (IR), uma câmara RGB e quatro microfones, bem como o facto de possuir uma API sem custo para além da aquisição do equipamento. E que por si só, e pela tecnologia que disponibiliza, apresenta um custo muito competitivo comparativamente com as restantes soluções comercializadas, torna-se uma boa escolha para o cumprimento dos objetivos da dissertação.

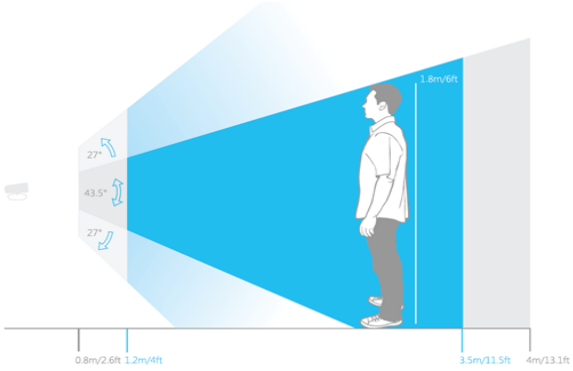


Figura 3: Limitação do campo de visão vertical da Kinect (adaptado de [64]).

3.2 Sistema proposto

Como referido na secção anterior o Hardware é composto unicamente por uma Kinect que se liga diretamente ao computador. Em termos de Software, a aplicação deverá permitir ao utilizador controlar o cursor do rato e simular os cliques, permitirá também a parametrização do sistema, através de menus e janelas. Esta aplicação,

tratando-se de uma funcionalidade de interface com o computador, pode ser expandida e adaptada a várias aplicações, tais como projetos como ecrãs de navegação *touchless* [64]. A aplicação desenvolvida foi designada por *UserTracking*.

O sistema pode ser descrito através do diagrama apresentado na Fig. 4. A aquisição da informação de áudio, imagem RGB e profundidade é feito por um sensor Kinect que é colocado por cima do monitor, centrado com este (ver também Fig. 18 e detalhes no Capitulo 4). A partir da medida da profundidade e da imagem RGB é feita a deteção do utilizador mais próximo, bem como é feita a estimativa da pose da face. São ainda extraídas as *Animation Units* [62], da sobrancelha e boca, que por sua vez vão servir como "botões" do rato. Em paralelo é retirado o som, que também pode servir (complementar) como comandos para os "botões" do rato. Tendo os botões do rato coberto, é necessário saber a posição no monitor do mesmo, isso é conseguido pela estimativa da pose da face, isto é, se a face está a olhar para a esquerda/direita ou para cima/baixo, o rato move-se em consonância. De seguida vamos apresentar em detalhe cada um dos referidos módulos.

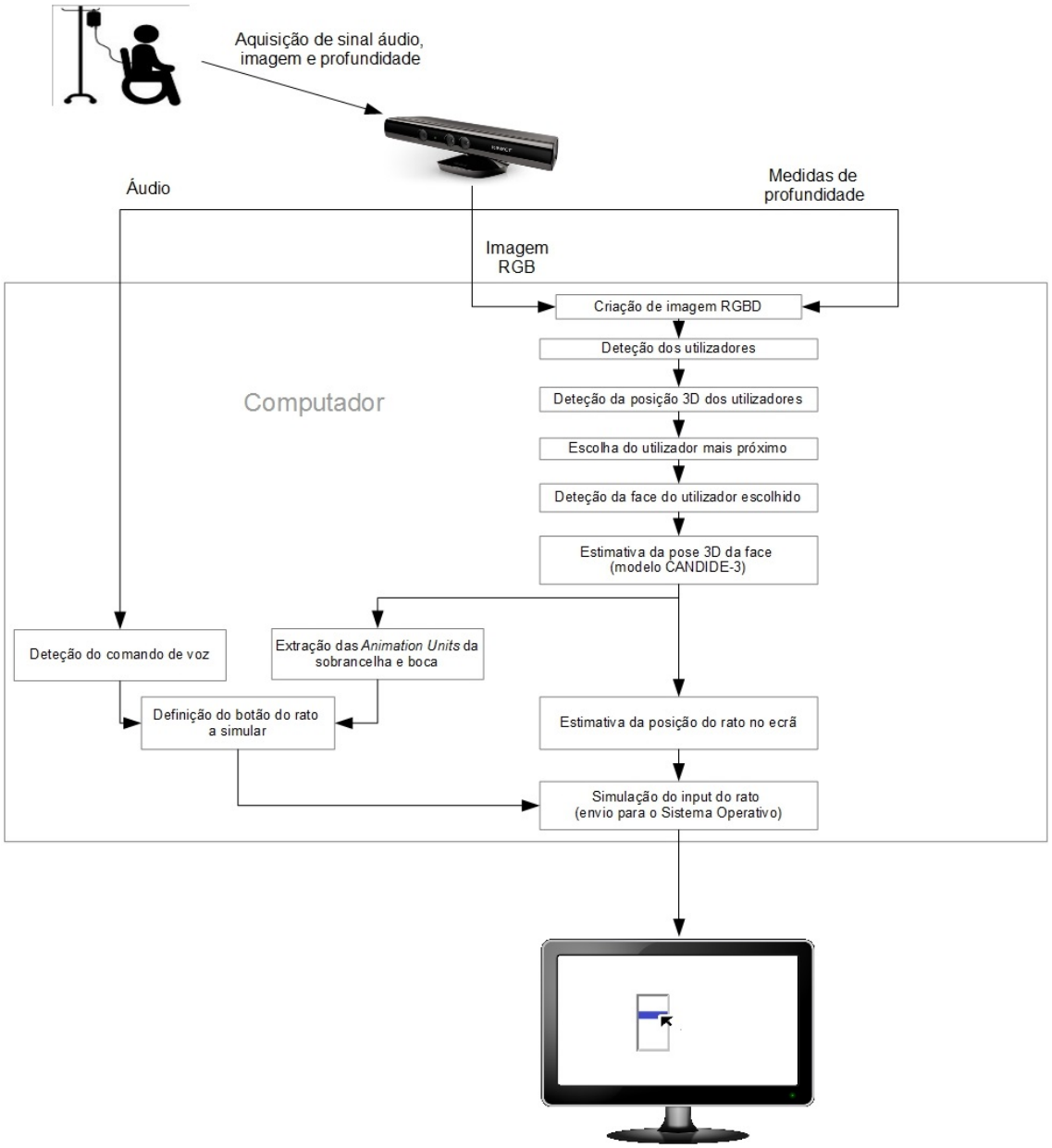


Figura 4: Diagrama do sistema proposto.

O protótipo desenvolvido utiliza um sensor Kinect XBOX360 e um computador com MS Windows 7, 64bits instalado. Foi também testada com a Kinect for Windows. A diferença entre elas resume-se à gama de distâncias suportada, e à licença de comercialização. No caso da Kinect XBOX360, a Microsoft autoriza o desenvolvimento de software, mas não autoriza a distribuição e comercialização. Enquanto que na versão para Windows estão autorizadas ambas as operações. Mas nada impede da aplicação ser testada durante o desenvolvimento com a versão XBOX360 e depois ser

comercializada/utilizada com a versão "*for Windows"*.

Com a câmara RGB é possível obter a imagem para deteção da face. Apesar de suportar várias resoluções, considera-se que a resolução 640×480 px (pixeis) é suficiente para o objetivo pretendido, permitindo assim também maior número de aquisições (*frames*) por segundo, 30 fps (*frames per second*) no máximo. A câmara de profundidade utiliza a técnica *Structured Light Imaging* e produz uma imagem em escala de cinza com uma resolução máxima de 640×480 px.

A comunicação entre o computador e a Kinect é realizada por um cabo USB da própria Kinect. A ligação em alguns casos pode necessitar de uma extensão, isto se a distância da Kinect ao computador for muito grande. Em termos de controlo da Kinect e aquisição de dados, é possível fazê-lo através de: (a) recorrendo aos drivers e software de desenvolvimento (Kinect SDK) disponibilizado pela Microsoft; (b) recorrendo aos drivers e software de desenvolvimento OpenNI [66] disponibilizados pela Primesense; (c) através da biblioteca *libfreenect* [67] da comunidade OpenKinect.

Neste projeto, a comunicação é realizada com recurso aos drivers e SDK da Microsoft. A linguagem de programação utilizada é C# e o ambiente de desenvolvimento integrado (IDE) é Visual Studio 2010 Express [68]. Optou-se pela linguagem pela sua simplicidade, e pelo IDE por estar otimizado para desenvolvimento rápido de aplicações (RAD) e pela sua licença gratuita (até mesmo para uso comercial).

3.3 Controlo da posição do rato no ecrã através do movimento da face

A deteção do utilizador é realizada pela API desenvolvida pela Microsoft especificamente para *FaceTracking* [69] com a Kinect. É possível descarregar esta API a partir do site da Microsoft. Toda a documentação, exemplos, e a própria API podem ser obtidos em [69]. No desenvolvimento deste projeto, foi utilizada Kinect for Windows SDK v1.8.

A *FaceTracking* SDK v1.8 requer a utilização simultânea das imagens obtidas pela câmara RGB e IR (profundidade), para fazer a deteção da cara. Para além dessas imagens, requer também informação do esqueleto de maneira que consiga reduzir o tamanho da imagem, isto é, a região de interesse onde são aplicados os algoritmos de deteção e seguimento. O próximo passo depois da cara detetada é o cálculo do ponto de foco.

3.3.1 Cálculo do ponto de foco

Nesta secção pretende-se calcular as coordenadas (ponto) no ecrã para onde a face do utilizador está orientada. A essa posição designou-se por ponto de foco. Em primeiro lugar, é necessário saber onde o utilizador se encontra, mais especificamente onde está a sua cara relativamente à Kinect. Sabendo a localização espacial da cara relativamente à Kinect, e sabendo a localização do ecrã (ponto superior esquerdo) relativamente à Kinect, é possível determinar o ponto de foco.

No sistema espacial da Kinect, todos os pontos que representam a cara obedecem ao conjunto de eixos cartesianos representados na Fig. 5. A origem de referencial encontra-se fisicamente no ponto médio entre as lentes da câmara RGB e do recetor de IR. A face é detetada e de seguida é aplicando o algoritmo *Active Appearance Model (AAM)* [70, 71, 72], este aplica o modelo facial CANDIDE-3 descrito detalhadamente em [73]. Douglas Gantenbein [74], referindo o investigador da Microsoft Jian Sun, declara que o algoritmo implementado na Kinect foi treinado e aprimorado durante cerca de 3 anos.

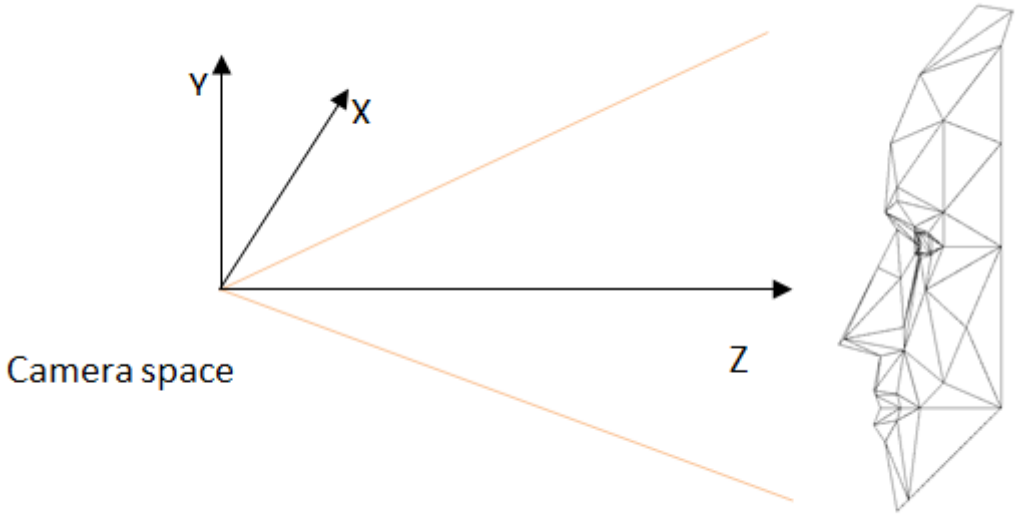


Figura 5: Representação da mesh relativamente à Kinect, adaptado de [78].

Na Fig. 6 apresenta-se um exemplo de um utilizador com os pontos da cara representados em forma de máscara (definida pelo modelo CANDIDE-3; ver também Fig. 9). A seta vermelha representa a sua orientação da face relativamente à Kinect. Para efeitos de cálculo definiu-se que esta seta tem origem no ponto médio entre os olhos do

utilizador.

Passando da Fig. 6 para a representação esquemática na Fig. 7, podemos arbitrar matematicamente que o vetor de orientação é representado como $\vec{V}$ (corresponde à seta de orientação na Fig. 6) e o ponto de origem de referencial é o ponto *O*. O ponto *P* corresponde ao ponto médio entre os olhos do utilizador e representa a origem do vetor $\vec{V}$. O ponto *Q* representa o ponto de foco, ou seja, representa o ponto de intersecção entre o vetor $\vec{V}$ e o plano *Π* formado pelo ecrã.

Como a Kinect fica colocada em cima do monitor, com os sensores no mesmo plano do ecrã, então podemos definir que para efeitos de cálculo, todos os pontos *Q* com coordenadas cartesianas (*x, y, z*) em metros incluídos neste plano *Π* têm a componente de profundidade nula, i.e., $z = 0$ m. A Fig. 7 pode ainda ser simplificada e ficar tal como está na Fig. 8.

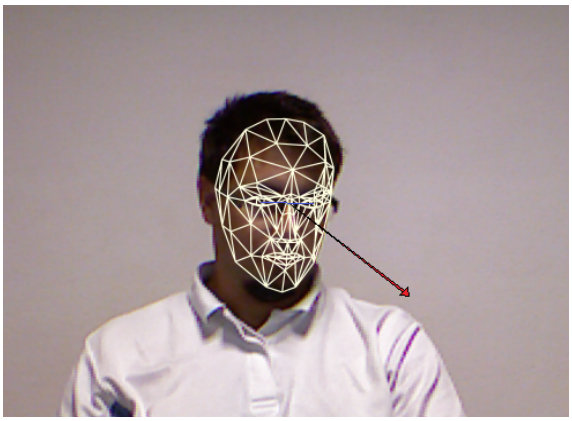


Figura 6: Representação facial detetada a partir do modelo CANDIDE-3 e respetivo vetor de orientação a vermelho.

Numa primeira fase, as coordenadas do ponto de foco são determinadas espacialmente relativamente à Kinect, e só depois é que são mapeadas para coordenadas de imagem no ecrã (em pixeis). Para este cálculo foi aplicada a técnica de *ray tracing* utilizada em computação gráfica para detetar o ponto de intersecção entre um raio e um plano [75]. Em que na presente situação, o raio corresponde ao nosso vetor de orientação da face. E o plano é paralelo ao ecrã do monitor para onde o utilizador está a olhar. Neste trabalho apenas foi considerada a utilização de um único monitor, o que não invalida que em trabalhos futuros se aumente a complexidade do sistema (uso de vários monitores em simultâneo).

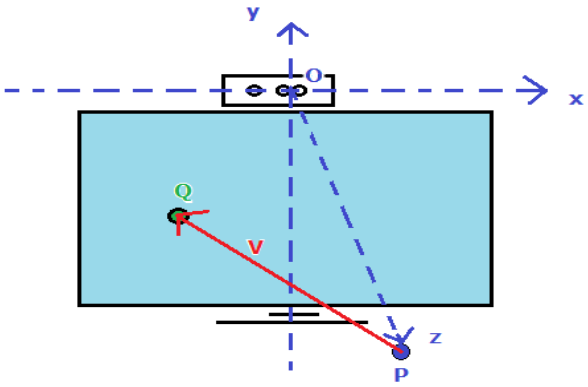


Figura 7: Representação das coordenadas da cara do utilizador pela posição do centróide e vetor de orientação.

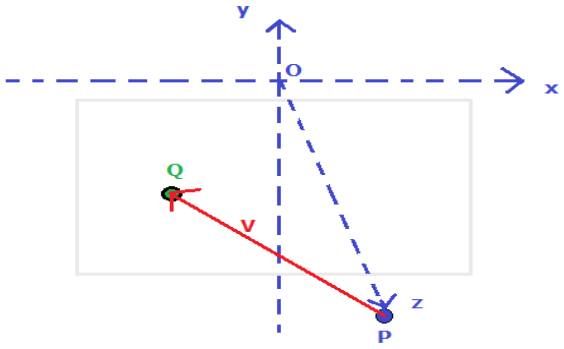


Figura 8: Representação simplificada do centróide e vetor de orientação da face do utilizador.

Das equações apresentadas em [75], a que devolve as coordenadas do ponto Q é $\vec{R}_i$ que é dado por

$$\vec{R}_i=(x_o+x_d t_i, y_o+y_d t_i, z_0+z_d t_i), \quad (1)$$

onde $(x_{0,} y_{0,} z_0)$ são as coordenadas do ponto P, (x_d, y_d, z_d) são as coordenadas do vetor $\vec{V}$, (x_i, y_i, z_i) são as coordenadas do ponto Q. E t_i resulta da equação

$$t_i=-\frac{\vec{N}\cdot\vec{R}_0+D}{\vec{N}\cdot\vec{R}_d}, \quad (2)$$

onde: $\vec{N}$ é o nosso vetor normal ao plano Π, que neste caso é (0, 0, 1), $\vec{R}_0$ é o vetor entre a origem e o ponto P, portanto $\vec{R}_0=\vec{OP}=P-O$, i.e, $\vec{R}_0=(x_0-0)\hat{e}_x+(y_0-0)\hat{e}_y+(z_0-0)\hat{e}_z=x_0\hat{e}_x+y_0\hat{e}_y+z_0\hat{e}_z$, D é o fator presente na equação geral de um plano $Ax+By+Cz+D=0$, que neste caso assume valor nulo

(explicação no próximo parágrafo) e $\vec{R}_d$ é o nosso vetor $\vec{V}$.

O fator *D* assume valor nulo porque pela definição em geometria analítica, é o produto interno entre o vetor $\vec{N}$ e um ponto conhecido do plano *Π*. Portanto, como o nosso plano contêm o ponto de origem de referencial *O* de coordenadas (0, 0, 0) então $D=\vec{N}\cdot O=(0,0,1)\cdot(0,0,0)=0\times0+0\times0+1\times0=0.$

Simplificando a equação (2), pela substituição $\vec{N}$ com (0, 0, 1), i.e, $\vec{N}=\hat{e}_z$, $\vec{R}_0=x_0\hat{e}_x+y_0\hat{e}_y+z_0\hat{e}_z$, $\vec{R}_d=x_0\hat{e}_x+y_0\hat{e}_y+z_0\hat{e}_z$ e $D=0$ fica

$$t_i=-\frac{\vec{N}\cdot\vec{R}_0+D}{\vec{N}\cdot\vec{R}_d}=-\frac{(\hat{e}_z)\cdot(x_0\hat{e}_x+y_0\hat{e}_y+z_0\hat{e}_z)+0}{(\hat{e}_z)\cdot(x_d\hat{e}_x+y_d\hat{e}_y+z_d\hat{e}_z)}=-\frac{z_0}{z_d}. \quad (3)$$

Substituindo a equação (3) em z_i , fica

$$z_i=z_0+t_i z_d=z_0-\frac{z_0}{z_d}z_d=0, \quad (4)$$

como seria de esperar. Portanto, a equação (1) pode ser reescrita para o ponto *Q* de coordenadas (x_i, y_i, z_i) como

$$(x_i, y_i, z_i)=\left(x_0-\frac{z_0}{z_d}x_d, y_0-\frac{z_0}{z_d}y_d, 0\right). \quad (5)$$

É de salientar que quando o utilizador tem a sua cara orientada ortogonalmente ao ecrã, vamos ter $z_d=0.$ Quando o utilizador está de costas viradas para o ecrã, temos $z_d>0,$ e neste caso específico, também não tem qualquer interesse realizar o cálculo. Como a API só faz *facetracking* quando o ângulo de desvio (*yaw angle*) está compreendido entre [-45º, 45º], e quando o ângulo de elevação (*pitch angle*) entre [-20º, 20º], então teremos sempre $z_d<0.$

Apesar da API fornecer o centróide e os ângulos de desvio, elevação e enrolamento da cara, não fornece o vetor de orientação $\vec{V},$ pelo que é necessário recalculá-lo também. O que é bastante simples recorrendo à aplicação de funções trigonométricas. Mas durante a implementação verificou-se que estas funções consomem demasiados recursos computacionais, e por isso, foi necessário adotar outra estratégia que envolvesse operações matemáticas mais simples.

Para o cálculo do vetor de orientação foi necessário recorrer aos pontos da máscara facial retornados pela API da Kinect. Como já foi referido anteriormente, a API utiliza o modelo facial CANDIDE-3 [73], o qual está representado na Fig. 9.

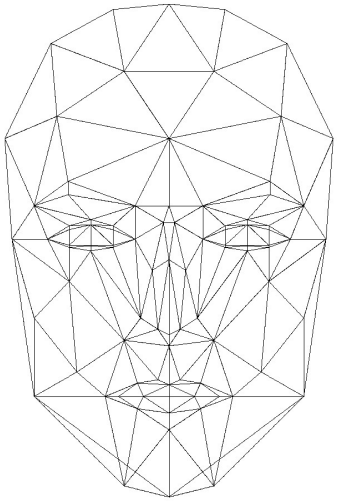


Figura 9: Modelo facial CANDIDE-3, adaptado de [73].

O ponto de origem *P* do vetor de orientação foi determinado a partir dos olhos (v.d. Fig. 10). Para cada olho, foram escolhidos três pontos que formam um triângulo. No caso do olho direito da face, deu origem ao triângulo *ABC*. No caso do olho esquerdo, foi formado o triângulo *DEF*.

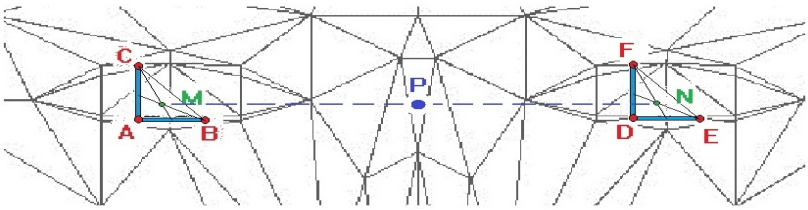


Figura 10: Cálculo do ponto de origem P do vetor de orientação.

Para efeitos de cálculo podemos definir os pontos e as suas respetivas coordenadas, A com (A_x, A_y, A_z), B com (B_x, B_y, B_z), C com (C_x, C_y, C_z), D com (D_x, D_y, D_z), E com (E_x, E_y, E_z), e F com (F_x, F_y, F_z). Os baricentros (vide Fig. 10) dos triângulos *ABC* e *DEF* estão representados pelos pontos M com (M_x, M_y, M_z) e N com (N_x, N_y, N_z), respetivamente, e podem ser calculados como:

$$M_x=\frac{A_x+B_x+C_x}{3};M_y=\frac{A_y+B_y+C_y}{3};M_z=\frac{A_z+B_z+C_z}{3};$$

$$N_x=\frac{D_x+E_x+F_x}{3};N_y=\frac{D_y+E_y+F_y}{3};N_z=\frac{D_z+E_z+F_z}{3}. \quad \text{Figura 11(6)}$$

O ponto $P\ (x_{0,}y_{0,}z_0)$ é o ponto médio entre os baricentros dos dois triângulos e pode ser determinado como

$$x_0=\frac{M_x+N_x}{2};y_0=\frac{M_y+N_y}{2};z_0=\frac{M_z+N_z}{2}. \quad (7)$$

Para determinar o vetor de orientação $\vec{V}$, foi necessário determinar primeiramente dois vetores $\vec{U}$ e $\vec{W}$, cada um ortogonal aos olhos direito e esquerdo, respetivamente. Foi utilizado o mesmo conjunto de pontos {*A, B, C*} e {*D, E, F*} que determinou ponto *P*. A Figura 11 exemplifica o cálculo do vetor $\vec{U}$ ortogonal ao olho direito.

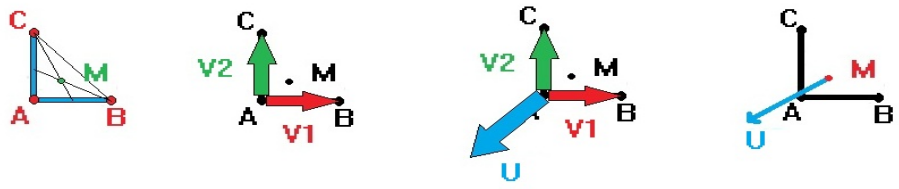


Figura 11: Cálculo do vetor ortogonal ao olho direito.

Em termos algébricos, podemos calcular $\vec{V}_1$ como sendo

$$\vec{V}_1=V_{1x}\hat{e}_x+V_{1y}\hat{e}_y+V_{1z}\hat{e}_z=\vec{AB}=(B_x-A_x)\hat{e}_x+(B_y-A_y)\hat{e}_y+(B_z-A_z)\hat{e}_z, \quad (8)$$

com $V_{1x}=B_x-A_x, V_{1y}=B_y-A_y, V_{1z}=B_z-A_z$ e $\vec{V}_2$ como sendo

$$\vec{V}_2=V_{2x}\hat{e}_x+V_{2y}\hat{e}_y+V_{2z}\hat{e}_z=\vec{AC}=(C_x-A_x)\hat{e}_x+(C_y-A_y)\hat{e}_y+(C_z-A_z)\hat{e}_z, \quad (9)$$

com $V_{2x}=C_x-A_x, V_{2y}=C_y-A_y, V_{2z}=C_z-A_z.$

Como $\vec{U}=U_x\hat{e}_x+U_y\hat{e}_y+U_z\hat{e}_z=\vec{V}_1\times\vec{V}_2$ então

$$\vec{U}=(V_{1y}\cdot V_{2z}-V_{1z}\cdot V_{2y})\hat{e}_x-(V_{1x}\cdot V_{2z}-V_{1z}\cdot V_{2x})\hat{e}_y+(V_{1x}\cdot V_{2y}-V_{1y}\cdot V_{2x})\hat{e}_z, \quad (10)$$

onde

$$U_x=(V_{1y}\cdot V_{2z}-V_{1z}\cdot V_{2y});U_y=(V_{1z}\cdot V_{2x}-V_{1x}\cdot V_{2z});U_z=(V_{1x}\cdot V_{2y}-V_{1y}\cdot V_{2x}). \quad (11)$$

Aplicando o mesmo raciocínio, pode-se determinar que o vetor $\vec{W}$ ortogonal ao olho esquerdo, e resulta em

$$\vec{W} = W_x \hat{e}_x + W_y \hat{e}_y + W_z \hat{e}_z = \vec{V}_3 \times \vec{V}_4 ; W_x = (V_{3y} \cdot V_{4z} - V_{3z} \cdot V_{4y}); \\ W_y = (V_{3z} \cdot V_{4x} - V_{3x} \cdot V_{4z}); W_z = (V_{3x} \cdot V_{4y} - V_{3y} \cdot V_{4x}), \quad (12)$$

onde

$$\vec{V}_3 = V_{3x} \hat{e}_x + V_{3y} \hat{e}_y + V_{3z} \hat{e}_z = \vec{DE} = (E_x - D_x) \hat{e}_x + (E_y - D_y) \hat{e}_y + (E_z - D_z) \hat{e}_z, \quad (13)$$

com $V_{3x} = E_x - D_x, V_{3y} = E_y - D_y, V_{3z} = E_z - D_z,$ e $V_{4x} = F_x - D_x, V_{4y} = F_y - D_y,$ $V_{4z} = F_z - D_z,$ assim,

$$\vec{V}_4 = V_{4x} \hat{e}_x + V_{4y} \hat{e}_y + V_{4z} \hat{e}_z = \vec{DF} = (F_x - D_x) \hat{F}_x + (F_y - D_y) \hat{F}_y + (F_z - D_z) \hat{F}_z \quad (14)$$

Finalmente, pode-se calcular o vetor de orientação $\vec{V}$. Este vetor é na verdade o vetor diretor unitário do segmento de reta $\bar{PQ}$, definido como $\vec{V} = x_d \hat{e}_x + y_d \hat{e}_y + z_d \hat{e}_z = \hat{S}$ onde $\vec{S} = (U_x + W_x) \hat{e}_x + (U_y + W_y) \hat{e}_y + (U_z + W_z) \hat{e}_z,$ i.e., $\vec{S} = \vec{U} + \vec{W},$

$$|\vec{S}| = \sqrt{(U_x + W_x)^2 + (U_y + W_y)^2 + (U_z + W_z)^2}, \hat{S} = \frac{\vec{S}}{|\vec{S}|}. \quad (15)$$

O vetor de orientação $\vec{V}$ fica localizado no ponto *P*, tal como se mostra em pormenor na Fig. 12.

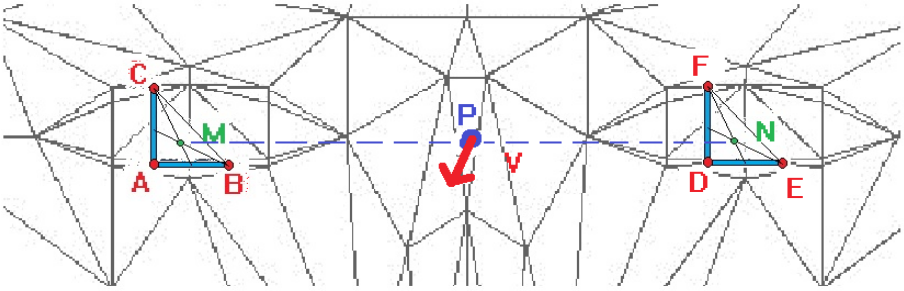


Figura 12: Posição do vetor de orientação na máscara facial.

Até agora todos os cálculos foram realizados em coordenadas espaciais (3D). Mas, o que interessa é saber as coordenadas na imagem do ecrã (em pixeis) para onde se vai enviar o ponteiro do rato. Como tal, são necessários mais alguns cálculos, para obter as coordenadas do ponto *Q* em unidades de imagem (pixeis). Como a Kinect, fica paralela ao ecrã vamos assumir que se encontra no mesmo plano $z = 0$ m. E como tal, os cálculos podem agora ser realizados em 2D, olhando apenas às componentes Q_x e Q_y.

É solicitado na inicialização da aplicação (apenas 1 vez) que o utilizador (pessoa que vai configurar inicialmente a aplicação) insira manualmente a posição da Kinect (em metros) relativamente ao canto superior esquerdo e o tamanho do ecrã (ver também

Capítulo 4). Com esta informação temos então todos os dados necessários à conversão.

Definimos o ponto G como sendo o ponto Q convertido para o sistema de coordenadas da imagem (px). Deve-se tomar em atenção que os dois sistemas têm o eixo vertical em sentidos opostos, tal como indicado na Fig. 13

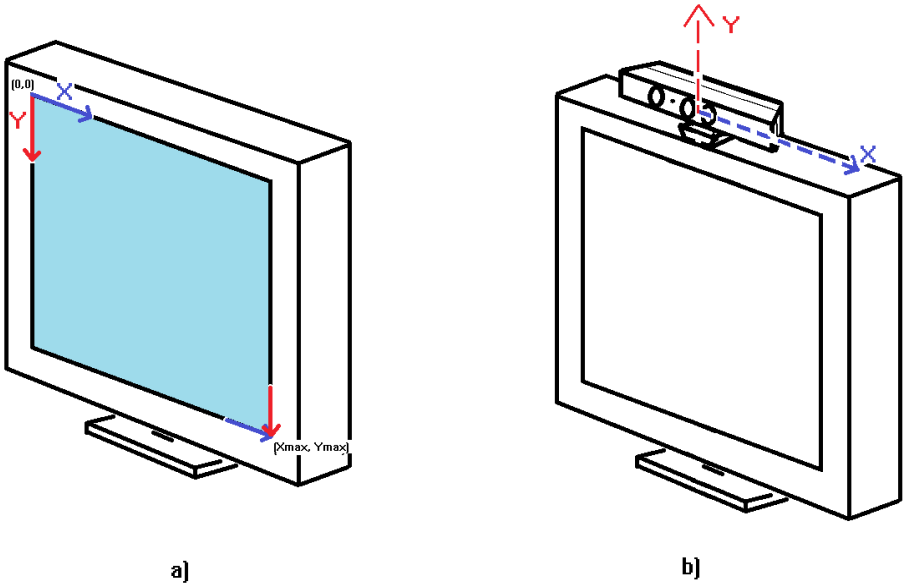


Figura 13: a) Representação dos sistemas de coordenadas de imagem no ecrã. b) Representação do sistema de coordenadas espaciais da Kinect.

Com as medidas introduzidas pelo utilizador (por quem vai configurar a aplicação) no software, vide Fig. 21, podemos então fazer a conversão, em que: w é a distância da Kinect ao canto superior esquerdo do ecrã, h é a distância da base da Kinect ao canto superior esquerdo do ecrã, m_w é a largura do ecrã, m_h é a altura do ecrã, r_w é a resolução horizontal da imagem, r_h é a resolução vertical da imagem. Assim, tendo o ponto Q de coordenadas (x_i, y_i) (em metros) pode-se determinar G de coordenadas (G_x, G_y) (em pixeis) como

$$G_x = \frac{(x_i + w)\cdot r_w}{m_w}; G_y = \frac{(y_i + h + 0.05)\cdot r_h}{m_h}. \quad (16)$$

Como a Kinect está posicionada ao centro, então qualquer ponto de imagem ao centro do ecrã medirá horizontalmente $x_i + w$ m, e verticalmente $y_i + h + 0{,}05$ m relativamente ao canto superior esquerdo do ecrã. O valor constante de 0,05 m = 5 cm (centímetros) deve-se à distância entre o ponto inferior da base e o ponto de origem das coordenadas espaciais da Kinect (vide Fig. 14).

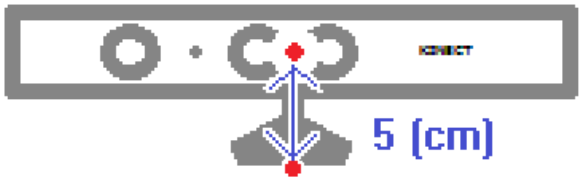


Figura 14: Distância entre o referencial espacial e a base da Kinect.

3.3.2 Estabilização e correção de ruído

A aquisição de dados pela Kinect®, tal como acontece em todos os equipamentos eletrónicos, está sujeita a ruído. Neste caso o ruído manifesta-se como a posição do rato não ficar exatamente no mesmo ponto enquanto o utilizador mantenha a cabeça completamente imóvel. Tentaram-se algumas estratégias para redução desse ruído, tais como: (a) a média dos últimos *n* pontos, (b) a média ponderada dos últimos *n* pontos, (c) a mediana dos últimos *n* pontos (excluindo o primeiro e último pontos do buffer), (d) a aplicação de filtro de Kalman (recorrendo às funções da biblioteca OpenCV [76]) e (e) o desenvolvimento de uma zona inerte, onde o rato não sofre alterações (nas coordenadas) em conjunto com um atraso do movimento do rato.

Nenhuma das estratégias a) a d) deu resultados suficientemente estáveis. Apesar do filtro preditivo de Kalman [77] possuir um vasto leque de aplicações, não trouxe a estabilização desejada ao deslocamento do ponteiro do rato. Talvez se deva por este sofrer mudanças bruscas de direção, sentido e aceleração.

A estratégia e) está dividida em duas componentes: (e.1) a aplicação de uma zona inerte, isto é, uma zona circular com um raio 3 px a volta da última posição do rato, onde as coordenadas deste não são alteradas, esta zona mantém-se inalterada durante 3 s (segundos). Foram testados vários valores, tendo sido verificado que estes valores são os que melhor se adequam, valores maiores o sistema (coordenadas do rato no ecrã) fica mais estável, no entanto depois perde precisão quando se pretende aceder a ícones pequenos. Em conjunto com a estratégia anterior foi desenvolvido (e.2) um atraso na deslocação/velocidade do rato no ecrã, esse atraso é proporcional à taxa de aquisição das imagens RGBD. Este atraso será mais uma vez configurado no início da aplicação, servirá também como um "regulador" de conforto para o utilizador. Diferentes

utilizadores preferem (sentem-se mais, ou menos, confortáveis) com velocidades e acelerações diferentes na movimentação do rato. Utilizadores com mobilidade reduzida, é de esperar que usualmente prefiram que o rato se desloque mais lentamente que os outros utilizadores ditos "normais".

A estratégia e) veio revelar-se uma estratégia simples e eficiente, sem o consumo de recursos de CPU que por exemplo, o filtro de Kalman apresentava.

3.3.3 Calibração automática

O método de calibração automática é realizado na primeira utilização do software *UserTracking*. Durante este processo, o utilizador deve apontar a sua face (para facilitar é dito para "apontar o nariz") para os nove pontos de fixação (são números) que vão aparecendo sequencialmente no ecrã, tal como mostra a Fig. 15.

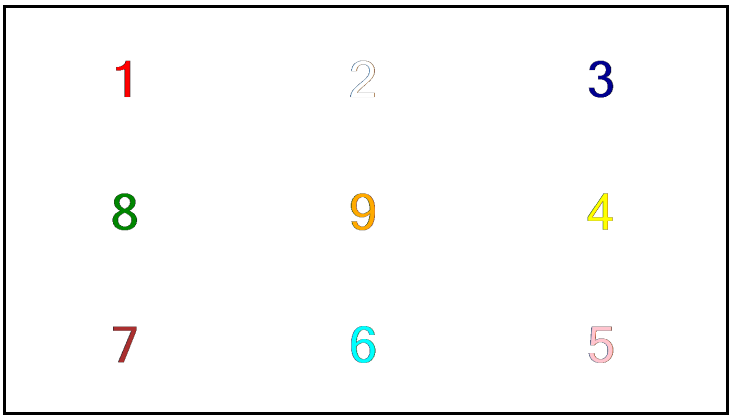


Figura 15: Ecrã de calibração automática.

O utilizador deve fixar cada ponto durante cerca de três segundos. Findo esse tempo, aparece outro ponto de fixação. E assim sucessivamente até fazer os nove pontos. Caso, o utilizador deixe de ser detetado, o processo de calibração fica em pausa, exibindo a mensagem de aviso "*User not detected*!" (utilizador não detetado). Logo que o utilizador volte a ser detetado, o software prossegue com o resto do processo de calibração. No entanto, esta falha de deteção vai dar origem a um resultado final deficiente, uma vez que durante a perda e a retoma do utilizador são enviados dados de movimento anormais (apresentam grande desvio relativamente ao suposto ponto de fixação). Por essa razão, sempre que existam falhas de deteção de utilizador durante o processo de calibração, é recomendado a repetição deste processo. Para repetir o

processo de calibração, o utilizador deve ir ao menu de configuração (ver próximo Capítulo) e clicar na opção de calibração.

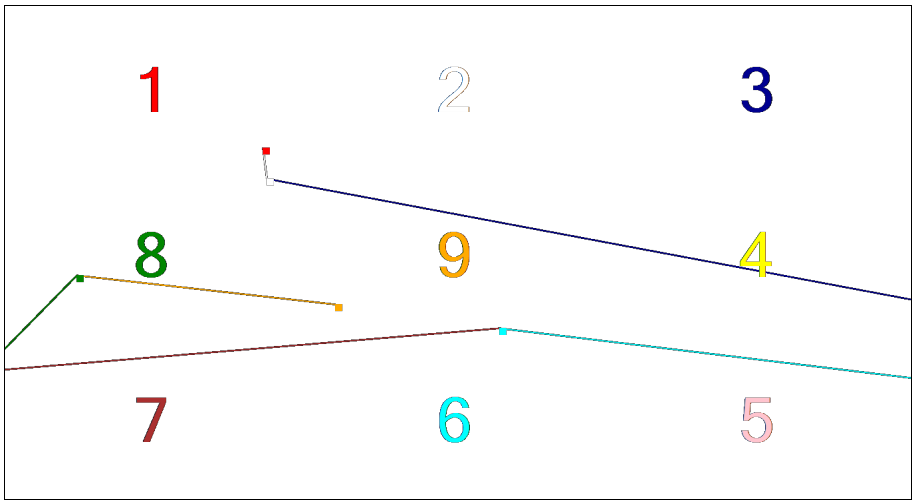


Figura 16: Exemplo do processo de calibração onde o utilizador olhou pouco para o ecrã.

Na Fig. 16, apresenta-se um exemplo de um utilizador que aponta, e desvia, várias vezes para o ecrã durante o processo de calibração. O resultado desde exemplo resulta numa má calibração, o que dificulta a utilização do sistema. Na Fig. 17, o exemplo apresentado, já demonstra uma tentativa de calibração mais ativa. O resultado desse processo já permite ao utilizador ter um maior controlo do ponteiro do rato comparativamente ao exemplo da Fig. 16.

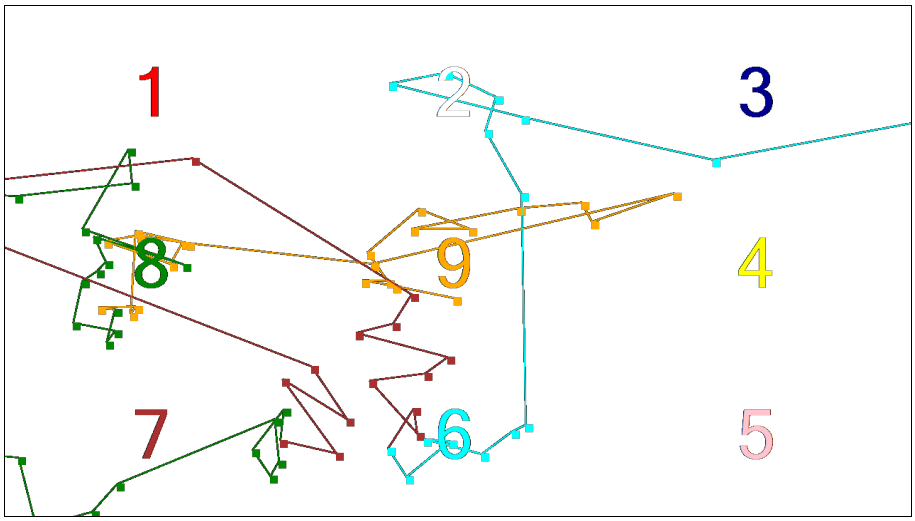


Figura 17: Exemplo de um processo de calibração onde o utilizador fixou alguns dos pontos de calibração.

3.4 Controlo dos botões do rato através da face

Os botões do rato podem ser simulados através da boca e da sobrancelha. No caso de serem detetadas várias pessoas em frente ao sensor, será escolhida como utilizador, aquela que estiver mais próxima do sensor. É apresentado um *snapshot* da cara do utilizador na barra de menu da aplicação *Usertracking*.

A partir do utilizador são extraídas as Unidades de Animação (*Animation Units* - AU) [78] da sobrancelha e da boca. Os valores destas AU variam entre 0 e 1. E é com base nestes valores, que são despoletadas as ações de controlo, simulando os botões do rato. Quando o utilizador movimenta a sobrancelha para cima, e a sua AU atinge um determinado valor limite, o *Usertracking* envia um comando para o sistema operativo, simulando um botão pressionado do rato. E quando o valor da AU da sobrancelha vem abaixo do valor limite, o *Usertracking* envia um comando para o sistema operativo, simulando a libertação do botão do rato [79]. E o mesmo procedimento é realizado relativamente à abertura da boca (que também tem a sua própria AU). Este procedimento está resumido no algoritmo na Tabela 1.

```
1   Parâmetros de entrada: ValorAberturaBoca, ValorAberturaSobrancelha,
    ValorLimiteBoca, ValorLimiteSobrancelha, BotãoASimular
2
3   Início
4
5      Se  ValorAberturaBoca >=  ValorLimiteBoca ou
6          ValorAberturaSobrancelha>= ValorLimiteSobrancelha então
7         PressionaBotãoDoRato(BotãoASimular)
8      Senão
9         LibertaBotãoDoRato(BotãoASimular)
10     Fim Se
11  Fim
```

Tabela 1: Algoritmo de simulação dos botões do rato através face.

Os valores limites são definidos manualmente pelo utilizador no software *Usertracking*. Habitualmente, os valores mais baixos são mais propensos a falsos positivos, e à medida que se aumentam os valores, torna-se cada vez mais difícil criar verdadeiros positivos. Os valores mais comuns, e que costumam apresentar uma utilização confortável, são 0.2, 0.3 e 0.4 (obtidos empiricamente ao longo da experimentação).

As funções PressionaBotãoDoRato() e LibertaBotãoDoRato(), chamadas no algoritmo da Tabela , podem ser implementados com os algoritmos apresentados nas Tabelas 8 e 9, respetivamente, dispostas na secção 3.6 Ligação das funcionalidades implementadas às do rato.

Para detalhes complementares de implementação deste controlo, será necessário consultar [80].

3.5 Comandos usando a voz

O desenvolvimento dos comandos por voz é bastante trivial, uma vez que basta usar o SDK da Kinect, que se integra com Microsoft Speech API (requer instalação à parte). Em primeiro lugar, deve ser definida uma lista de frases, ou palavras, que serão os comandos reconhecidos acusticamente através das funcionalidades da Microsoft Speech API por meio dos microfones da Kinect®. Esta lista de comandos é chamada de *gramática*, uma vez que vai conter as palavras e regras necessárias para que a API possa fazer o reconhecimento corretamente. A estrutura da gramática obedece a um formato XML estilizado, e deve ser encontrado na documentação online [81]. A cada comando

de voz é associada programaticamente uma ação, isto é, traduz o comando de voz e simula o mesmo efeito do clique físico no rato. Até ao momento só foram desenvolvidos os seguintes comandos em Inglês: *"Left button", "Right Button", "Click","Left Click", "Right Click", "Double Click", "Click Down", "Click Up", "Drag" e "Drop"*. As funcionalidades destes comandos estão explicadas na secção 4.2, e encontram-se resumidos na Tab. 10.

Durante a instanciação das classes da API, é necessário definir a língua utilizada nos comandos de voz. Neste projeto só se encontra implementada a língua Inglesa Americana (abreviada como *en-US*). Existem muitas outras línguas e variantes disponíveis para utilizar, e apresentam-se na Tab. 2.

Língua-País	Língua-Código do País
Catalan - Spain	ca-ES
Danish - Denmark	da-DK
German - Germany	de-DE
English - Australia	en-AU
English - Canada	en-CA
English - Great Britain	en-GB
English - Indian	en-IN
English - United States	en-US
Spanish - Spain	es-ES
Spanish - Mexico	es-MX
Finnish - Finland	fi-FI
French - Canada	fr-CA
French - France	fr-FR
Italian - Italy	it-IT
Japanese - Japan	ja-JP
Korean - Korea	ko-KR
Norwegian - Norway	nb-NO
Dutch - Netherlands	nl-NL
Polish - Poland	pl-PL
Portuguese - Brazil	pt-BR
Portuguese - Portugal	pt-PT
Russian - Russia	ru-RU
Swedish - Sweden	sv-SE
Chinese - China	zh-CN
Chinese - Hong Kong	zh-HK
Chinese - Taiwan	zh-TW

Tabela 2: Línguas reconhecidas pela Microsoft Speech API, adaptado de [82].

Um exemplo de um ficheiro de gramática é o que se apresenta na Tab.3.

```
<?xml version="1.0" encoding="utf-8" ?>
<grammar version="1.0" xml:lang="en-US" tag-format="semantics/1.0-literals"
xmlns="http://www.w3.org/2001/06/grammar">
  <rule id="MouseCommands" scope="public">
    <one-of>
        <item>double click</item>
        <item>click</item>
        <item>click down</item>
        <item>click up</item>
        <item>left button</item>
        <item>left click</item>
        <item>right button</item>
        <item>right click</item>
        <item>drag</item>
        <item>drop</item>
    </one-of>
  </rule>
</grammar>
```

Tabela 3: Exemplo de um ficheiro de gramática utilizado no reconhecimento dos comandos de voz.

Depois de instanciadas e inicializadas, as classes da API, têm que despoletar as ações necessárias. Esta relação é definida numa estrutura condicional de múltipla escolha, e deve ser processada cada vez que for reconhecido um comando de voz. O pseudocódigo deste evento deve ser semelhante ao descrito na Tabela 4. A função Simula_Evento_Do_Rato() pode ser implementada com o algoritmo em pseudocódigo apresentado na Tabela 6, e informação complementar na Tabela 7.

Para detalhes complementares de implementação consultar [83].

```
1   Parâmetros de entrada: ComandoVoz
2
3   Início
4      Escolha ComandoVoz
5         Caso "double click":
6            Botão_Selecionado ← Botão_Esquerdo
7            Simula_Evento_Do_Rato(Esquerdo_Em_Baixo)
8            Simula_Evento_Do_Rato(Esquerdo_Em_Cima)
9            Simula_Evento_Do_Rato(Esquerdo_Em_Baixo)
10           Simula_Evento_Do_Rato(Esquerdo_Em_Cima)
11        Caso "click":
12           Se Botão_Selecionado = Botão_Esquerdo Então
13              Simula_Evento_Do_Rato(Esquerdo_Em_Baixo)
14              Simula_Evento_Do_Rato(Esquerdo_Em_Cima)
15           Senão
16              Simula_Evento_Do_Rato(Direito_Em_Baixo)
17              Simula_Evento_Do_Rato(Direito_Em_Cima)
18           Fim Se
19        Caso "drag":
20           Botão_Selecionado ← Botão_Esquerdo
21           Simula_Evento_Do_Rato(Esquerdo_Em_Baixo)
22        Caso "drop", "click up":
23           Se Botão_Selecionado = Botão_Esquerdo Então
24              Simula_Evento_Do_Rato(Esquerdo_Em_Cima)
25           Senão
26              Simula_Evento_Do_Rato(Direito_Em_Cima)
27           Fim Se
28        Caso "right button":
29           Botão_Selecionado ← Botão_Direito
30        Caso "right click":
31           Botão_Selecionado ← Botão_Direito
32           Simula_Evento_Do_Rato(Direito_Em_Baixo)
33           Simula_Evento_Do_Rato(Direito_Em_Cima)
34        Caso "left button":
35           Botão_Selecionado ← Botão_Esquerdo
36        Caso "left click":
37           Botão_Selecionado ← Botão_Esquerdo
38           Simula_Evento_Do_Rato(Esquerdo_Em_Baixo)
39           Simula_Evento_Do_Rato(Esquerdo_Em_Cima)
40     Fim Escolha
41  Fim
```

Tabela 4: Algoritmo que associa o comando de voz às ações do rato.

3.6 Ligação das funcionalidades implementadas às do rato

Após termos a emulação dos movimentos do rato pelo movimento da face, bem como os comandos de voz, é agora necessário associá-los aos movimentos do rato. Os cliques do rato são simulados através da abertura boca, movimento das sobrancelhas e por comandos de voz. Ao abrir a boca, ou levantar as sobrancelhas, é simulado o pressionar do botão, e quando fecha a boca, ou baixa as sobrancelhas, simula o despressionar do botão, relembra-se que esta informação é extraída a partir das *Animation Units* [61]. A associação destes movimentos podem ser feitos através das funcionalidades da linguagem C# e das funções **mouse_event()** [84], **SetCursorPos()** [85] e **GetCursorPos()** [86]**,** disponíveis na livraria de sistema "user32.dll".

Sempre que o utilizador move a sua face, é calculada a suposta posição do rato no ecrã (o cálculo é realizado com as equações expostas na secção **3.3.1 Cálculo do ponto de foco**) e realiza-se uma chamada à função **SetCursorPos().** Veja o pseudocódigo que implementa o deslocamento do ponteiro do rato na Tab.5.

```
1   Parâmetros de entrada: Posicao_Foco_X, Posicao_Foco_Y
2
3   Início
4      SetCursorPos(Posicao_Foco_X, Posicao_Foco_Y)
5   Fim
```

Tabela 5: Algoritmo em pseudocódigo para implementação de deslocação do ponteiro do rato.

Sempre que o utilizador move a sobrancelha, ou move a boca, ou transmite um comando de voz reconhecido, deverá ser enviado o respetivo comando para o sistema operativo com a função **mouse_event()**. Para utilizar esta função é necessário enviar também a posição do rato, tal como indicado na documentação em [84]. Portanto, pode-se utilizar a função **GetCursorPos()** para obter a posição atual do rato.

```
1   Parâmetros de entrada: Ordem_Rato
2   Declare variáveis Posicao_Rato_X, Posicao_Rato_Y como inteiros
3   Início
4      GetCursorPos(Posicao_Rato_X, Posicao_Rato_Y)
5      mouse_event(Ordem_Rato, Posicao_Rato_X, Posicao_Rato_Y, 0, 0)
6   Fim
```

Tabela 6: Algoritmo Simula_Evento_Do_Rato() em pseudocódigo para implementação dos cliques.

No algoritmo apresentado na Tabela 6, para implementação dos cliques, a variável "Ordem_Rato" representa o valor correspondente à ordem que deverá ser transmitida ao sistema operativo. Sendo portanto simultaneamente a indicação do botão, e respetivo estado, pretendidos. No âmbito deste projeto, essa variável assumirá, consoante a situação, um dos valores da Tabela 7.

Valor	Constante	Significado
0x0002	Esquerdo_Em_Baixo	botão esquerdo em baixo (pressionado)
0x0004	Esquerdo_Em_Cima	botão esquerdo em cima (livre)
0x0008	Direito_Em_Baixo	botão direito em baixo (pressionado)
0x0010	Direito_Em_Cima	botão direito em cima (livre)

Tabela 7: Argumentos a utilizar com a função mouse_event() consultados de [84].

As funções chamadas no algoritmo da Tabela 1, PressionaBotãoDoRato() e LibertaBotãoDoRato() podem recorrer à função Simula_Evento_Do_Rato() (Tabela 7) bastando para isso implementar os algoritmos apresentados nas Tabelas 8 e 9, respetivamente. Para detalhes complementares de implementação consultar [80].

```
1   Parâmetros de entrada: BotãoASimular
2
3   Início
4
5      Escolhe  BotãoASimular
6         Caso Botão_Esquerdo: Simula_Evento_Do_Rato(Esquerdo_Em_Baixo)
7         Caso Botão_Direito: Simula_Evento_Do_Rato(Direito_Em_Baixo)
8      Fim Escolhe
9
10  Fim
```

Tabela 8: Algoritmo PressionaBotãoDoRato() em pseudocódigo.

```
1   Parâmetros de entrada: BotãoASimular
2
3   Início
4
5      Escolhe  BotãoASimular
6         Caso Botão_Esquerdo: Simula_Evento_Do_Rato(Esquerdo_Em_Cima)
7         Caso Botão_Direito: Simula_Evento_Do_Rato(Direito_Em_Cima)
8      Fim Escolhe
9
10  Fim
```

Tabela 9: Algoritmo LibertaBotãoDoRato() em pseudocódigo.

4 Protótipo desenvolvido

Nas secções seguintes é apresentado o protótipo desenvolvido, e respetivo software *UserTracking,* bem como os passos necessários à configuração do mesmo.

4.1 Preparação do sistema

Antes de conectar a Kinect ao computador, o utilizador deve descarregar os drivers (só foi testada a versão 1.8.0) diretamente do site da Microsoft e instalá-los. Depois de instalados os drivers, é que o utilizador deve conectar a Kinect. Aconselha-se a instalação do software *UserTracking* (desenvolvido nesta dissertação) apenas depois de conectar a Kinect. O *UserTracking* tem um pacote de instalação que instala para o executável e todas as livrarias necessárias (DLL), e inicia automaticamente.

A Kinect deverá ficar montada no topo do monitor, de preferência ao centro, e a parte frontal deverá ficar à face do monitor, tal como indicado na Fig. 18. Se o monitor não tiver espaço suficiente para assegurar a estabilidade da Kinect, então deverá ficar em cima de um apoio, como por exemplo, semelhante ao que está na Fig. 19. Pode-se encontrar este apoio à venda na Internet por preços muito baixos.

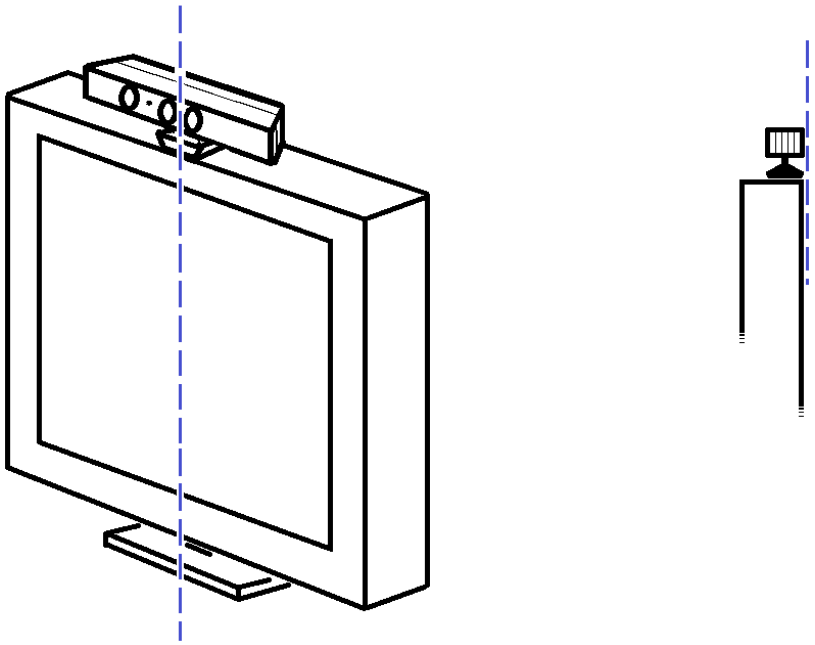


Figura 18: Kinect colocada no topo de um monitor, vista em perspetiva à esquerda, e vista lateral à direita.

Figura 19: Suporte para Kinect, adaptado de [61].

Depois de montar a Kinect é necessário medir a distância, nas suas componentes horizontal e vertical, desde a base da Kinect (ponto central baixo da base) até ao ponto superior esquerdo do ecrã, tal como mostrado na Fig. 20. As medidas deverão ser tiradas em centímetros, em que *w* (*L* na Fig. 20) representa distância horizontal, e *h* (*A* na Fig. 20) representa a distância vertical entre o ecrã.

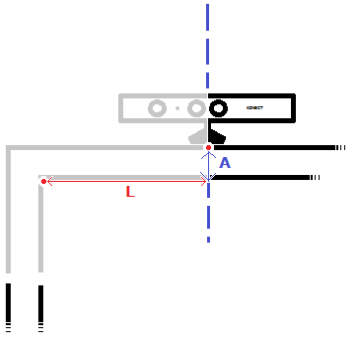


Figura 20: Medidas necessárias para configuração do sistema.

Essas medidas deverão ser introduzidas em centímetros na janela de configuração (separador Kinect) do software *UserTracking* (vide Fig. 21).

Na Fig. 22 temos um cenário em que a Kinect está montada no topo do monitor de um computador portátil. Utiliza um suporte semelhante ao que foi apresentado na Fig. 19. Logo que o software de *UserTracking* tenha sido inicializado, e o utilizador detetado (processo que demora alguns segundos), este passa a ter o controlo do ponteiro do rato no ecrã. Mesmo que tente controlar com a mão no rato, o *UserTracking* vai sempre sobrepor o seu controlo. Na Fig. 23 é apresentado um utilizador que está a modificar os parâmetros de configuração do software, movimentando unicamente a sua cabeça.

É recomendado que haja sempre outra pessoa a auxiliar na configuração, uma vez que o utilizador pode perder o controlo do ponteiro do rato, devido a um parâmetro eventualmente mal definido.

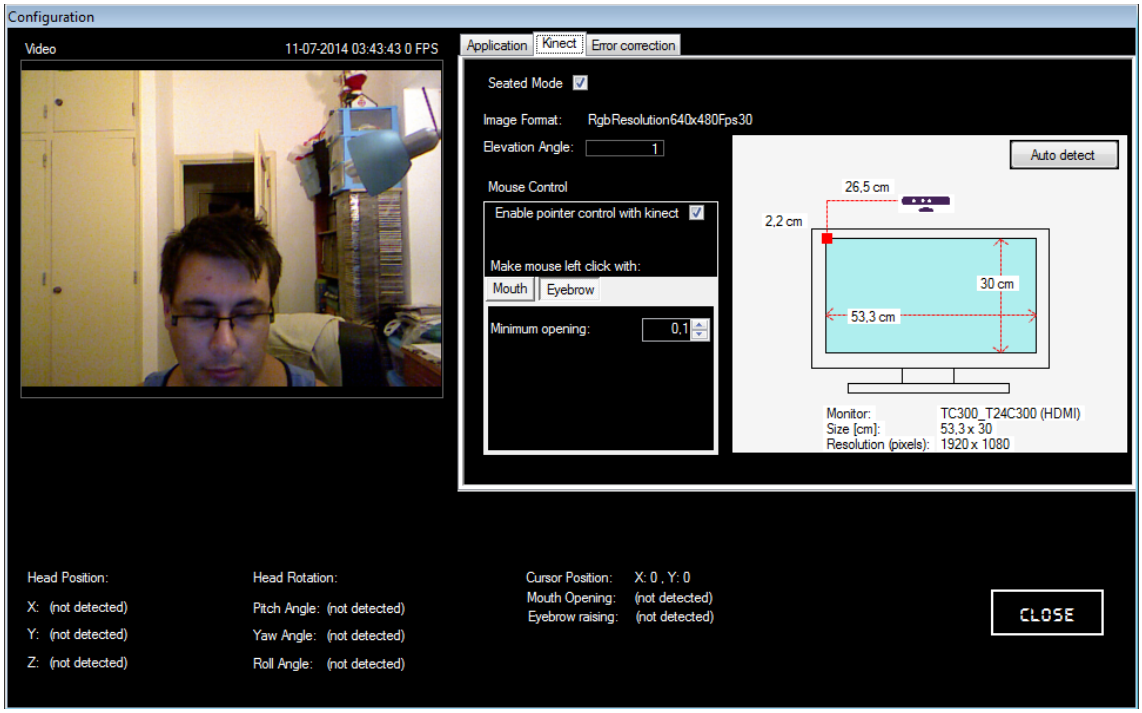


Figura 21: Janela de configuração do software de UserTracking.

Figura 22: Cenário em que mostra como posicionar a Kinect por cima do monitor de um computador portátil.

Figura 23: Utilizador sentado em frente ao computador a controlar o rato apenas com o movimento da sua própria cabeça.

4.2 Utilização do sistema

O *UserTracking* atua no ponteiro do rato simulando as ordens do rato enviadas para o Sistema Operativo (SO). O ponteiro do rato é deslocado para a posição para onde o utilizador aponta a sua cara. Costuma-se utilizar o nariz como melhor referência do apontador. Quer isto dizer que o utilizador deve imaginar que possui uma seta no seu nariz, que aponta para o ponto onde quer posicionar o rato. Chama-se a este tipo de controlo de deslocamento absoluto do rato.

Os cliques do rato são simulados através da abertura boca, movimento das sobrancelhas e por comandos de voz. Ao abrir a boca, ou levantar as sobrancelhas, é simulado o pressionar do botão, e quando fecha a boca, ou baixa as sobrancelhas, simula o despressionar do botão. Portanto, tal como acontece com o rato, quando o botão está pressionado, é possível arrastar um item no ecrã do computador, e depois largar (*drag'n'drop*). Os comandos de voz e a respetiva descrição encontra-se na Tab. 10.

No caso de se estar a utilizar a boca ou as sobrancelhas, a escolha do botão é realizada na barra disponibilizada pelo programa (ver Fig. 24). Basta clicar para alternar

entre *Right Click* e *Left Click*. A opção *Parking Mode* faz com que os movimentos do utilizador, e a voz, não sejam reconhecidos como ordens, ou seja, desativa os cliques. Para voltar a habilitar o reconhecimento das ordens, basta desabilitar a opção de *Parking Mode*.

Comando	Descrição
Left button	Seleciona o botão esquerdo.
Right Button	Seleciona o botão direito.
Click	Faz clique com o botão selecionado (previamente).
Left Click	Seleciona o botão esquerdo e faz clique.
Right Click	Seleciona o botão direito e faz clique.
Double Click	Seleciona o botão esquerdo e faz duplo clique.
Click Down	Coloca botão selecionado no estado pressionado. Útil para arrastar um item no ecrã do computador.
Click Up	Coloca botão selecionado no estado de-pressionado. Útil para largar um item no ecrã do computador, na sequência de um arrastamento.
Drag	Seleciona o botão esquerdo e mantêm o botão pressionado.
Drop	Solta o botão pressionado. Idêntico ao "Click Up".

Tabela 10: Comandos de voz aceites pelo software UserTracking.

4.3 Deteção de utilizador

A deteção do utilizador é realizada pela API desenvolvida pela Microsoft Kinect for Windows SDK. Na realidade é feita a deteção do esqueleto dos utilizadores que estiverem no campo de visão. Nesta versão permitidos até cinco utilizadores, mas apenas um deles, o que está mais próximo, é escolhido para controlar o sistema. Quando o utilizador não é detetado, o sistema dá a informação de "*No user detected*!", sob a forma de pessoa desconhecida, tal como se mostra na Fig. 24.

Para o utilizador ser detetado deverá colocar-se no campo de visão do sensor. Quando o utilizador é detetado, o sistema apresenta a foto da face do utilizador em vez de pessoa desconhecida. É apresentado um exemplo de como fica a imagem do utilizador na Fig. 25. A barra disponibilizada pelo programa, Fig. 24, mantém-se sempre visível. Sendo possível configurá-la para desaparecer (fica escondida) após ter perdido o foco durante 10 segundos, e voltando a aparecer sempre que o utilizador leva o rato para os limites (direito, esquerdo, topo ou fundo) do ecrã.

Figura 24: Mensagem apresentada quando o sistema não consegue detetar o utilizador.

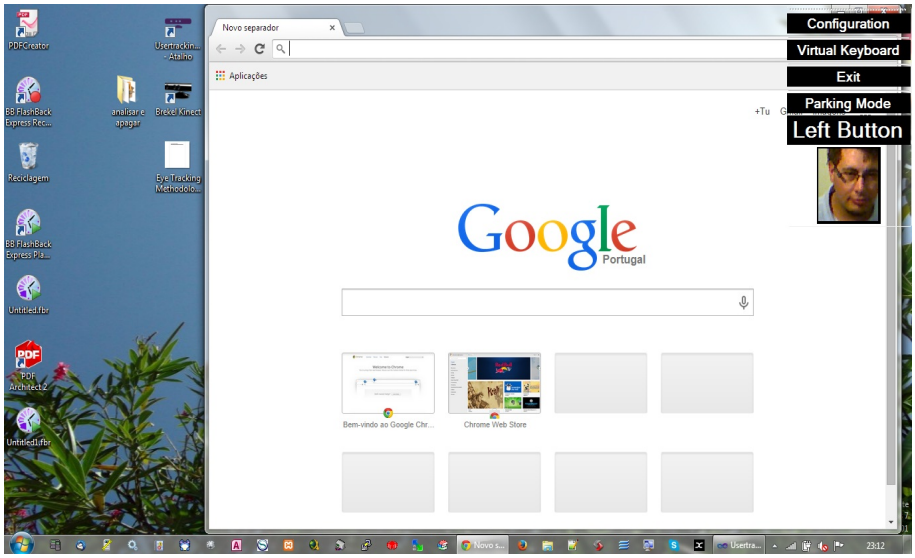


Figura 25: Utilizador detetado, imagem facial inicial detetada e mostrada na barra do software de controlo.

5 Ensaio e análise de resultados

O sistema desenvolvido foi submetido a ensaio com vários utilizadores. Pretendeu-se avaliar a possibilidade do sistema apresentado se traduzir numa alternativa viável às interfaces tradicionais para interação com o computador.

5.1 Ensaio

No ensaio pretendeu-se avaliar a capacidade do sistema em detetar utilizadores com diferentes características (forma da cara, cor de pele, cor de olhos, óculos, corte de cabelo, etc.) em ambientes (apenas interiores) com caraterísticas luminotécnicas diferentes. Em cada sessão de ensaio foi disponibilizado um inquérito/formulário com uma lista de procedimentos a realizar, existe disponível uma cópia do inquérito no Anexo B.

Antes de cada sessão foi realizado um pequeno treino para o utilizador se adaptar, calibrar e conhecer o sistema proposto. Durante a sessão, o utilizador foi filmado em simultâneo com as ações despoletadas no computador, para assim, se avaliar a maneabilidade do sistema. Esta gravação foi igualmente necessária para se aferir o tempo útil que o utilizador demorou a realizar cada procedimento, e por forma a termos uma base de comparação. No final da sessão pretendeu-se saber o nível de conforto sentido pelo utilizador, a sua opinião e sugestões de melhoria.

Através do questionário acima referido, foram também recolhidas algumas características dos utilizadores de maneira a facilitar a posterior análise. As características recolhidas foram o género, idade e ascendência apenas para classificação estatística. Foram recolhidas mais algumas características que poderiam afetar a capacidade de deteção de utilizadores, tais como, a cor de pele, a cor dos olhos, o corte de cabelo, a presença de barba, óculos e outros acessórios faciais. Os acessórios poderiam tornar-se num obstáculo à deteção da presença de uma cara. Pretendeu-se também apurar a experiência prévia do utilizador com o computador, para assim ajudar a compreender melhor o tempo que cada utilizador demorou em cada procedimento.

Relativamente aos procedimentos, ou tarefas, realizadas durante o ensaio, o utilizador teve acesso a dois teclados virtuais. Para além do teclado nativo no software *UserTracking*, também esteve disponível como alternativa, o teclado nativo do Microsoft Windows 7. Este teclado costuma acompanhar todas as versões do Microsoft

Windows e pertence à categoria das ferramentas de acessibilidade.

Todos os procedimentos foram cronometrados para assim se apurar a sua duração. A duração está no formato hh:mm:ss. Os procedimentos 3.1-7, são considerados os procedimentos principais, ou objetivos a atingir. Os procedimentos 3.1-7.# são as instruções a executar para atingir o objetivo descrito no procedimento principal 3.1-7. Sendo que não é crítico, se o utilizador não cumprir exatamente as instruções para atingir o objetivo principal. Se o utilizador assim o entender, deve seguir um outro caminho mais rápido para atingir o objetivo. No ensaio, o que realmente importa é atingir o objetivo principal.

O primeiro procedimento instruí o utilizador a criar uma pasta no ambiente de trabalho com um nome específico "Ensaio nº#" onde "#" corresponde ao número sequencial do ensaio. Este procedimento procedimento é identificado pela numero 3.1. De seguida, pediu-se ao utilizador para abrir um *browser* (procedimento 3.2), aceder ao website do Youtube, e deixar uma música a tocar. Estes são os procedimentos 3.3.1 a 3.3.3. A pesquisa da música seria pela palavra "saxophone" (ver Anexo B). Com estes procedimentos, pretendeu-se testar a escrita com recurso a um teclado virtual, e em simultâneo, verificar a resposta do sistema a comandos de voz na presença de música.

O utilizador teve que descarregar um ficheiro da Internet, neste caso do site da UAlg, e gravá-lo na pasta criada no ambiente de trabalho. O ficheiro em causa é um ficheiro .zip com as marcas gráficas do Instituto Superior de Engenharia (ISE). Depois de gravado localmente, o ficheiro deve ser extraído através dos procedimentos 3.5.1 a 3.5.3. Os procedimentos 3.6.1 a 3.6.4 são instruções para o utilizador criar um desenho simples – uma cara – junto ao logótipo do ISE, e gravá-lo num ficheiro .jpg.

Os últimos procedimentos 3.7.1 a 3.7.7 são instruções para criar um documento de texto com 3 frases com formatação específica. Estes procedimentos servem para testar se o sistema desenvolvido possibilita ao utilizador a redação de um documento formatado utilizando apenas os movimentos da sua cabeça e um teclado virtual. Na próxima secção, vamos avaliar os resultados dos ensaios.

5.2 Análise de resultados

Todos os dados recolhidos foram resumidos em tabelas e gráficos para melhor comparação e apresentação. Foram recrutados oito voluntários, dos quais três desistiram durante a fase de calibração e treino para o ensaio. As razões que levaram à desistência foram apuradas e ficou-se a saber que se devia ao difícil controlo do ponteiro do rato (era demasiado sensível para os movimentos realizados com a cabeça). E noutros casos, queixaram-se de um grande desconforto ao utilizar o sistema, nomeadamente de dores no pescoço. Um dos voluntários (que desistiu) possuía lentes bifocais e o facto de alternar o campo de visão entre a parte da lente para ver de perto e a outra parte para ver de longe, teve uma grande dificuldade a fazer a calibração (quando via uma parte do ecrã, não via bem a parte oposta). Sendo que durante a fase de treino para o ensaio, começou a queixar-se de dores de cabeça, o que levou à sua desistência.

Na Tab. 11 estão representadas as características dos voluntários que realizaram o ensaio. Observando a referida tabela conclui-se que os utilizadores foram maioritariamente do género masculino, de olhos castanhos, cor de pele clara, e possuíam óculos ou lentes de contacto. Alguns dos participantes (60%) apresentaram uma barba leve, e essa característica não foi objeto de qualquer impedimento para a deteção pelo sistema.

Alguns dos participantes com pele escura tiveram alguma dificuldade na deteção, nomeadamente na presença de um fundo também de cor mais escura (móvel castanho escuro). Nesses casos, os participantes deslocaram-se um pouco para o lado, de maneira que a sua cabeça recaísse no fundo claro (parede branca). Não houve nenhum participante com acessórios faciais (*piercings*). E o facto de se usar óculos, ou lentes de contacto, também não impediu a utilização do sistema.

O corte de cabelo predominante foi o corte curto. Mesmo os poucos voluntários que tinham cabelo longo ou encaracolado, não tiveram dificuldade em ser detetados pelo sistema. Por coincidência, o único voluntário que era careca também era o único que tinha óculos de lentes bifocais (acessório que contribuiu fortemente para a desistência), portanto, a ausência de cabelo não se revelou um obstáculo à deteção do utilizador.

Todos os utilizadores que realizaram o ensaio, possuíam já experiência com o Software necessário para as tarefas solicitadas, i.e., costumavam navegar na Internet, descarregar ficheiros e assistir vídeos (ou ouvir música) no YouTube. Já estavam

bastante familiarizados com o editor de desenho Microsoft Paint e o editor de texto OpenOffice Writer, portanto, os tempos que os utilizadores levaram a realizar o ensaio não foram, na sua maioria, afetados pela falta de experiência com o Software disponibilizado.

Total de voluntários	8					
Total de desistências	3					
Total de ensaios	**5**					
1.1 Género				**1.9 Corte de cabelo**		
Masculino	1	20,00%		Curto liso	4	80,00%
Feminino	4	80,00%		Liso preso atrás	1	20,00%
1.2 Idade				Careca	0	0,00%
0 – 10	0	0,00%		Encaracolado volumoso	0	0,00%
10 – 20	2	40,00%		Encaracolado preso atrás	0	0,00%
20- 30	0	0,00%		**1.10 Utiliza acessórios faciais**		
30-40	3	60,00%		Sim	0	0,00%
40-50	0	0,00%		Não	5	100,00%
50-60	0	0,00%		**2.1 Frequência de utilização de PC**		
mais de 60	0	0,00%		Diária	3	60,00%
1.3 Ascendência				Esporádica	2	40,00%
Portuguesa	3	60,00%		Nenhuma	0	0,00%
Indiana	2	40,00%		**2.2 Costuma utiliza o MS Windows**		
1.4 Cor de pele				Sim	5	100,00%
Clara	3	60,00%		Não	0	0,00%
Escura	2	40,00%		**2.3 Costuma navegar na Internet**		
1.5 Barba				Sim	5	100,00%
Sim	3	60,00%		Não	0	0,00%
Não	2	40,00%		**2.4 Costuma aceder ao Youtube**		
1.6 Cor dos olhos				Sim	5	100,00%
castanhos	5	100,00%		Não	0	0,00%
verdes	0	0,00%		**2.5 Costuma descarregar ficheiros da Internet**		
azuis	0	0,00%		Sim	5	100,00%
1.7 Utiliza óculos				Não	0	0,00%
Sim	3	60,00%		**2.6 Já utilizou o Paint para desenhar**		
Não	2	40,00%		Sim	5	100,00%
1.8 Utiliza lentes de contacto				Não	0	0,00%
Sim	1	20,00%		**2.7 Costuma redigir documentos**		
Não	4	80,00%		Sim	5	100,00%
				Não	0	0,00%

Tabela 11: Características dos voluntários que realizaram o ensaio.

A realização de cada procedimento de cada ensaio foi cronometrada, e o resultado dos tempos está na Tab. 12. Esta tabela também apresenta a duração média, e desvio padrão, nas duas colunas da direita. A negrito estão evidenciadas procedimentos principais ou objetivos a atingir.

Procedimento	Ensaio 1	Ensaio 2	Ensaio 3	Ensaio 4	Ensaio 5	Duração média	Desvio padrão
3.1	**00:11:20**	**00:06:31**	**00:00:00**	**00:17:45**	**00:18:11**	**00:13:27**	**00:05:35**
3.2	**00:00:17**	**00:03:07**	**00:00:00**	**00:00:33**	**00:01:26**	**00:01:21**	**00:01:17**
3.3	**00:06:52**	**00:04:46**	**00:00:00**	**00:03:36**	**00:19:41**	**00:08:44**	**00:07:26**
3.3.1	00:05:15	00:00:00	00:00:00	00:01:04	00:14:42	00:07:00	00:06:15
3.3.2	00:01:09	00:00:00	00:00:00	00:02:20	00:04:39	00:02:43	00:01:57
3.3.3	00:00:28	00:00:00	00:00:00	00:00:12	00:00:20	00:00:20	00:00:12
3.4	**00:16:34**	**00:02:14**	**00:00:00**	**00:10:09**	**00:05:27**	**00:08:36**	**00:06:14**
3.4.1	00:00:05	00:00:00	00:00:00	00:00:26	00:00:16	00:00:16	00:00:11
3.4.2	00:09:12	00:00:00	00:00:00	00:00:56	00:05:11	00:05:06	00:04:03
3.4.3	00:01:07	00:00:00	00:00:00	00:00:28	00:00:00	00:00:48	00:00:29
3.4.4	00:00:26	00:00:00	00:00:00	00:00:36	00:00:00	00:00:31	00:00:17
3.4.5	00:02:03	00:00:44	00:00:00	00:01:08	00:00:00	00:01:18	00:00:52
3.4.6	00:02:01	00:00:21	00:00:00	00:02:43	00:00:00	00:01:42	00:01:16
3.4.7	00:01:40	00:01:09	00:00:00	00:03:52	00:00:00	00:02:14	00:01:35
3.5	**00:02:06**	**00:02:16**	**00:00:00**	**00:03:43**	**00:00:00**	**00:02:45**	**00:00:53**
3.5.1	00:00:00	00:00:07	00:00:00	00:00:00	00:00:00	00:00:07	00:00:03
3.5.2	00:00:00	00:00:18	00:00:00	00:01:36	00:00:00	00:00:57	00:00:42
3.5.3	00:00:00	00:01:51	00:00:00	00:02:07	00:00:00	00:01:59	00:01:05
3.6	**00:09:42**	**00:00:00**	**00:27:39**	**00:16:46**	**00:00:00**	**00:18:02**	**00:09:03**
3.6.1	00:01:41	00:00:08	00:00:00	00:00:21	00:00:00	00:00:43	00:00:43
3.6.2	00:00:47	00:00:29	00:00:00	00:00:41	00:00:00	00:00:39	00:00:22
3.6.3	00:03:31	00:00:00	00:22:01	00:06:50	00:00:00	00:10:47	00:09:08
3.6.4	00:03:43	00:00:00	00:05:38	00:08:54	00:00:00	00:06:05	00:03:49
3.7	**00:13:07**	**00:00:00**	**00:15:08**	**00:25:15**	**00:00:00**	**00:17:50**	**00:06:30**
3.7.1	00:00:34	00:00:00	00:00:25	00:00:03	00:00:00	00:00:21	00:00:16
3.7.2	00:07:43	00:00:00	00:00:00	00:08:00	00:00:00	00:07:52	00:04:18
3.7.3	00:00:00	00:00:00	00:00:00	00:00:00	00:00:00	00:00:00	00:00:00
3.7.4	00:00:00	00:00:00	00:00:00	00:00:00	00:00:00	00:00:00	00:00:00
3.7.5	00:03:37	00:00:00	00:14:04	00:11:05	00:00:00	00:09:35	00:06:29
3.7.6	00:01:04	00:00:00	00:00:32	00:06:07	00:00:00	00:02:34	00:02:36
3.7.7	00:00:09	00:00:00	00:00:07	00:00:00	00:00:00	00:00:08	00:00:04
3.8	**00:00:00**	**00:00:00**	**00:00:00**	**00:00:00**	**00:00:00**	**00:00:00**	**00:00:00**

Tabela 12: Duração de cada procedimento em cada ensaio. Média e desvio padrão da duração por procedimento apresentado nas duas colunas da direita.

Analisando a Tab. 12, verifica-se que existem muitos valores a 00:00:00. Estes valores indicam que o utilizador não realizou o procedimento indicado. Este valor está excluído do cálculo nas médias e desvio padrão. Só interessa obter a média dos tempos de quem tentou realizar o procedimento. No entanto, houve algumas situações em que o utilizador cumpriu o objetivo principal (procedimentos 3.#), sem seguir à letra as instruções (procedimentos 3.#.#). E por isso, poderão haver casos em que a duração apenas se encontra registada no objetivo principal, e não nas instruções. Vide por exemplo, o procedimento 3.3 do ensaio 2 apresenta uma duração, mas as instruções 3.3.1, 3.3.2 e 3.3.3 estão a 00:00:00.

Por esta razão, a Tab. 13 e 14 foram criadas como uma simplificação da Tab. 12. A Tab. 13 apresenta o tempo que cada objetivo demorou a ser atingido em cada ensaio. No entanto, existe sempre algum tempo que não é contabilizado entre os ensaios, porque o utilizador está focado noutra atividade, como por exemplo, a configurar o Software *UserTracking*, ou organizar pastas, ficheiros, janelas no ecrã, etc. Dos ensaios realizados, verificou-se que um ensaio completo demoraria entre 01:30:00 e 02:00:00, dependendo das dificuldades que o utilizador sentisse ao longo do ensaio.

Procedimento	Ensaio 1	Ensaio 2	Ensaio 3	Ensaio 4	Ensaio 5
3.1	00:11:20	00:06:31	00:00:00	00:17:45	00:18:11
3.2	00:00:17	00:03:07	00:00:00	00:00:33	00:01:26
3.3	00:06:52	00:04:46	00:00:00	00:03:36	00:19:41
3.4	00:16:34	00:02:14	00:00:00	00:10:09	00:05:27
3.5	00:02:06	00:02:16	00:00:00	00:03:43	00:00:00
3.6	00:09:42	00:00:00	00:27:39	00:16:46	00:00:00
3.7	00:13:07	00:00:00	00:15:08	00:25:15	00:00:00

Tabela 13: Simplificação da Tab. 12. Tempo que cada objetivo demorou a ser atingido em cada ensaio.

A Tab. 14 reúne a duração média, e o desvio padrão, que cada um dos procedimentos demorou a ser realizado. Para facilitar a comparação dos tempos da Tab. 13 foi criado o gráfico apresentado na Fig. 26. Por observação do gráfico facilmente chegamos à conclusão de os procedimentos que envolvem a utilização do teclado virtual, são os mais demorados. Ainda assim, podem-se comprar os objetivos 3.1 e o 3.7. No objetivo 3.1 é normal o utilizador demorar muito tempo, uma vez que é a primeira utilização do teclado durante o ensaio. Mas se o compararmos com o objetivo 3.7, a quantidade de

instruções a realizar é inferior, e o objetivo 3.1 é claramente mais simples do que o 3.7. Neste último é exigida a escrita de mais caracteres, e tem que lhes aplicar uma formatação específica, o que em termos proporcionais levaria mais do triplo do tempo. Mas o que acontece, é que demora apenas, em média, mais 50% (ver Tab. 14). Esta poupança de tempo, levando em consideração a complexidade de ambos os objetivos, pode dever-se ao facto do utilizador estar a ganhar experiência no controlo do sistema, uma vez que normalmente estes procedimentos ocorrem com cerca de uma hora de intervalo.

Procedimento	Duração média	Desvio Padrão
3.1	00:13:27	00:05:35
3.2	00:01:21	00:01:17
3.3	00:08:44	00:07:26
3.4	00:08:36	00:06:14
3.5	00:02:42	00:00:53
3.6	00:18:02	00:09:03
3.7	00:17:50	00:06:30

Tabela 14: Duração média, e desvio padrão, de cada procedimento.

Os procedimentos 3.3 e 3.4 costumam demorar bastante tempo por duas razões. A primeira razão deve-se ao facto do teclado compacto tornar-se pouco prático para escrever, razão pela qual os utilizadores durante estes procedimental optam por usar o teclado nativo do Microsoft Windows. A outra razão está relacionada com a utilização das barras verticais que aparecem no *browser* durante a navegação na web.

O procedimento 3.6 é um verdadeiro teste ao controlo do ponteiro do rato. Neste procedimento, o utilizador tem que desenhar uma cara com círculos e linhas no Microsoft Paint. Obriga a coordenar o movimento de deslocamento com o controlo de clique. Neste procedimento, verificou-se que o pior tempo se deveu ao cariz artístico do utilizador. Para além disso, este utilizador optou por iniciar o ensaio a partir deste ponto, ignorando todos os objetivos anteriores, alegando falta de disponibilidade para continuar com o ensaio.

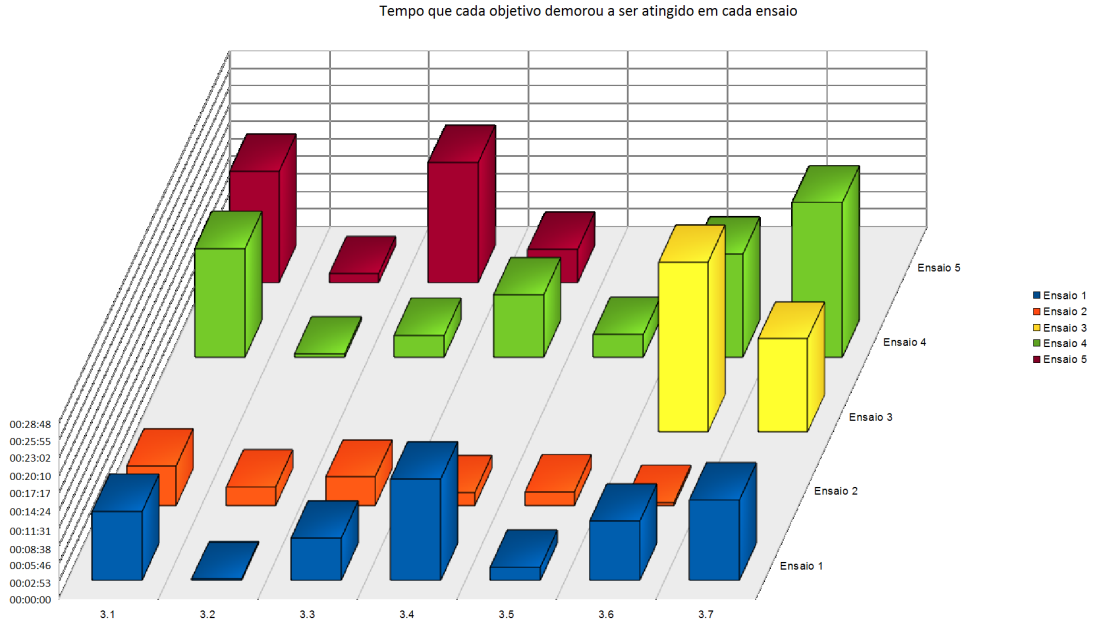


Figura 26: Tempo que cada objetivo demorou a ser atingido em cada ensaio, representação gráfica dos dados da Tab. 13.

Existe algum interesse em comparar os tempos entre os utilizadores de óculos e os que não utilizam óculos. Foi criada a Tab. 15 e o gráfico da Fig. 27, com esses tempos. Este interesse vem do facto de se demonstrar que para o sistema, a utilização de óculos é pouco relevante. Facilmente se comprova pela observação do gráfico da Fig. 27, onde os utilizadores de óculos estão praticamente ao mesmo nível dos restantes em termos de tempos demorados a completar os objetivos propostos. Pelo gráfico observa-se que existiu alguma dificuldade no ponto 3.3 para os utilizadores de óculos. Mas na realidade ao confirmar com a Tab. 13, esse valor é influenciado pelo tempo excessivo demorado no Ensaio 5. A resposta para esse tempo foi a difícil adaptação do utilizador ao sistema por falta de paciência, até porque o utilizador só completou metade do ensaio.

Uma outra característica que também interessava avaliar era a da tonalidade da pele. Este tipo de soluções tem a fama de ter dificuldades com a deteção de caras com tonalidade mais escura. Os algoritmos que se baseavam apenas na deteção de olhos em pixeis mais escuros, têm uma grande margem de erro, especialmente quando se trata de pele escura. Mas os algoritmos mais modernos já implementam estratégias mais avançadas, tais como redes neuronais para detetar a face humana.

Procedimentos	Duração média nos ensaios com óculos	Duração média nos ensaios sem óculos
3.1	00:14:46	00:12:08
3.2	00:00:52	00:01:50
3.3	00:13:17	00:04:11
3.4	00:11:01	00:06:12
3.5	00:02:06	00:03:00
3.6	00:18:41	00:08:42
3.7	00:14:08	00:25:15

Tabela 15: Duração média obtida nos ensaios dos utilizadores com óculos e nos ensaios dos utilizadores sem óculos.

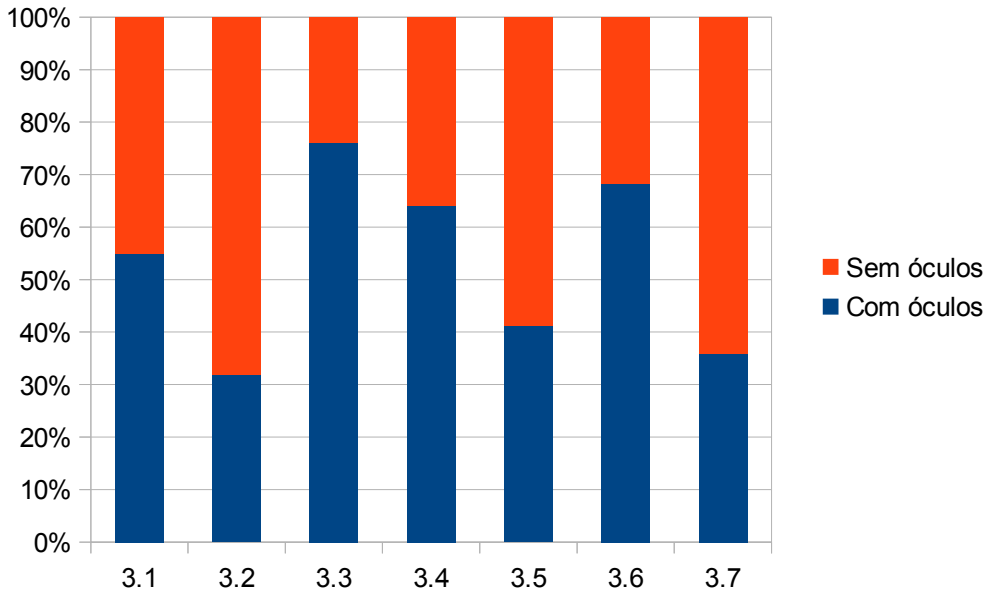


Figura 27: Comparação dos tempos médios que os utilizadores de óculos demoraram a realizar os procedimentos versus os que não possuíam óculos.

Determinou-se a duração média dos procedimentos realizados pelos utilizadores de pele mais clara, e pelos de pele mais escura. E esses valores estão disponíveis para consulta na Tab. 16 e no gráfico da Fig. 28 (os tempos foram convertidos em percentagem para melhor comparação).

Procedimento	Tempo médio dos utilizadores de pele clara	Tempo médio dos utilizadores de pele escura
3.1	00:08:56	00:17:58
3.2	00:01:42	00:01:00
3.3	00:05:49	00:11:39
3.4	00:09:24	00:07:48
3.5	00:02:11	00:03:43
3.6	00:12:39	00:16:46
3.7	00:14:08	00:25:15

Tabela 16: Duração média dos procedimentos realizados pelos utilizadores de pele clara e de pele escura.

Os valores da Tab. 16 dão a indicação que na maioria das vezes, os utilizadores de pele escura demoram mais tempo a realizar os mesmos procedimentos do que os utilizadores de pele branca. Mas por avaliação visual dos vídeos dos ensaios constatou-se que os tempos mais demorados não se deveram à cor da pele, mas sim a alguma falta de prática a trabalhar com o computador. Para além disso, utilizavam recorrentemente comandos de voz que não existiam, ou que não eram reconhecidos. Esta última observação também afeta os utilizadores de pele branca.

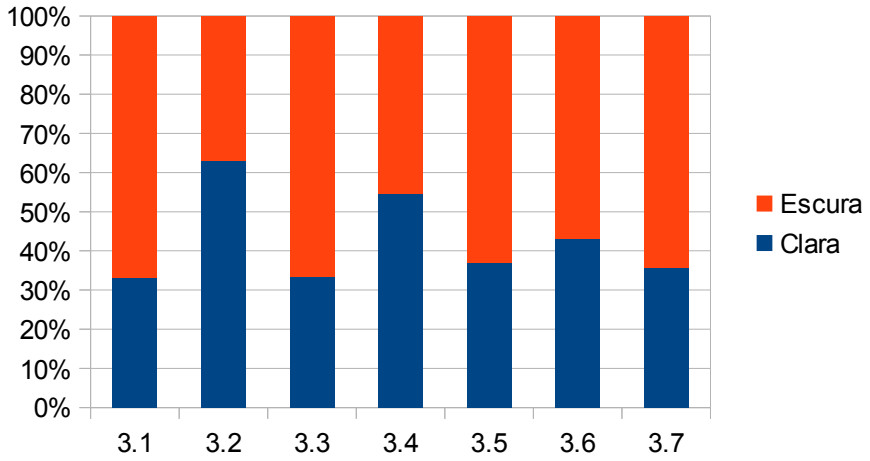


Figura 28: Comparação dos tempos médios de resposta entre os utilizadores de pele clara e os utilizadores de pele escura.

Durante os ensaios constatou-se que a música tocada no computador interferia

bastante com os comandos por voz. Sempre que os utilizadores sentiram necessidade de utilizá-los foi necessário baixar, ou parar por completo a música.

5.3 Opinião e sugestões dos intervenientes

Na opinião de alguns utilizadores o sistema torna-se simples de controlar depois de alguma prática. Algumas vezes o sistema não deteta a posição da cabeça corretamente, e isso deveria ser corrigido. Alguns utilizadores acharam que o ponteiro do rato se movia demasiado rápido para o movimento que eles exerciam com a cabeça. Por vezes, o sistema altera automaticamente e inexplicavelmente para o clique direito, o que dificulta o controlo do sistema. Os utilizadores queixaram-se de falsos cliques sem ordem deles, o que causou algum stress.

Os comandos de voz e os movimentos da boca nem sempre são reconhecidos. O sistema provoca muito cansaço principalmente para que usa óculos com lentes progressivas. Cansa muito os olhos. E torna-se stressante, porque há zonas no ecrã em que os utilizadores têm dificuldade em ver o ponteiro do rato no ecrã (porque a posição das lentes não corrige a visão corretamente). Também houve queixas de dor nas costas. Surgiram algumas sugestões para melhorar o teclado, como a criação de uma tecla "www." que enviasse logo a expressão completa para o ecrã. Alguns utilizadores acharam que o ambiente de trabalho deveria estar dividido em duas partes. Em que uma parte seria o espaço para as janelas e documentos abrirem, e na outra parte deveria ter o teclado sempre visível. Alguns utilizadores sugeriram a eliminação do teclado compacto, e ficar só com o teclado do Microsoft Windows. A interação do sistema com as barras de deslocamento pode ser melhorada, talvez com comandos de voz.

6 Conclusões e trabalhos futuros

Verificou-se que a biblioteca matemática fornecida na *framework* .net da Microsoft é pouco eficiente no que toca ao desempenho. E que portanto, a sua utilização deve ser substituída por operações matemáticas, ou instruções mais simples (adição, multiplicação, subtração, divisão, etc.) sempre que tal seja possível.

O inquérito (Anexo B) teve por objetivo ajudar a perceber em que tarefas, ou movimentos, onde os utilizadores tiveram maior dificuldade em realizar. Para além disso, permitiu também saber qual o nível de conforto sentido durante a utilização desta interface. A opinião dos utilizadores é fundamental, e por isso o inquérito tem algumas perguntas de resposta aberta, dando liberdade a sugestões de melhoria do sistema.

Quando se utiliza este tipo de sistemas de controlo com a cabeça, é aconselhável modificar o tema visual do computador. Pelo facto do pescoço não estar preparado para movimentos tão finos, como se consegue por exemplo com as mãos, o utilizador esforça e provoca uma grande tensão nos músculos que o constituem afim de fixar pontos que se encontrem muito próximos entre si. Este esforço gera um grande desconforto no utilizador, dando origem a dores no pescoço, e à necessidade constante de se afastar do computador para exercer movimentos de alívio da tensão acumulada. Na tentativa de se reduzir o desconforto, o computador deve ser configurado com um tema que apresente ícones bastante grandes e espaçados para obrigar o utilizador a fazer movimentos mais longos.

Os ensaios revelaram que o sistema pode ser utilizado por pessoas de pele clara ou escura. A existência dos óculos também não se revelou um obstáculo na deteção da cara e da sua pose. Mas, constatou-se que alguns tipos de óculos devem ser evitados na utilização deste sistema, devido ao desconforto sentido pelo utilizador. Os utilizadores com óculos com campo de visão verticalmente reduzido, e óculos bifocais, tiveram mais dificuldade em utilizar o sistema. No caso especial dos óculos com lentes bifocais, que utilizam lentes com correção simultânea para ver ao perto e ao longe, provocaram um desconforto tão grande durante a fase de pré-ensaio que impossibilitaram o utilizador de realizar o ensaio padrão.

O sistema de controlo com a cabeça requer algum tempo de prática. Recomenda-se portanto que o utilizador não abuse na duração que passa em frente ao computador. É preferível passar pequenos intervalos de tempo, até começar a ter maior controlo. Na

fase de adaptação, o utilizador precisa ter alguma paciência, caso contrário, vai acabar por desmotivar e colocar o sistema de parte. Ao fim de algumas sessões, verifica-se que o utilizador começa a controlar o cursor calmamente com o movimento da cabeça.

O teclado compacto disponibilizado pelo software *UserTracking* deve realmente ser substituído por uma versão mais longa, em que as letras, e números, tenham cada uma a sua própria tecla. A criação de vários caracteres na mesma tecla só apresentou vantagem pelo facto de ocupar menos espaço no ecrã. Tirando isso, os utilizadores não se sentiram confortáveis em utilizá-lo, pelo que preferiram a utilização do teclado virtual presente nas ferramentas de acessibilidade do Microsoft Windows 7, apesar deste ocupar uma grande parte do espaço no ecrã. Em versões futuras, recomenda-se a utilização do teclado do Microsoft Windows 7 para escrita de caracteres e alternativa ao teclado real.

Ficou como sugestão dos utilizadores a criação de um novo teclado virtual com teclas personalizáveis. Por exemplo, o novo teclado poderá ter teclas que são configuradas pelo utilizador, para reproduzir mensagens de voz, utilizando para isso, ficheiros de áudio associados ou vozes sintetizadas *text-to-speech*. A funcionalidade de conclusão automática de palavras também deverá ser considerada nas próximas versões.

Os comandos de voz não foram contemplados inicialmente. Surgiram como um extra ao longo do desenvolvimento, mas ao longo do ensaio com os vários utilizadores veio a provar ser uma parte funcional de grande importância, e deverá permanecer em futuros trabalhos. Nas próximas versões, a lista de comandos deverá estender-se, e facilitar por exemplo, o deslocamento vertical e horizontal de páginas *web*, e outras aplicações que assim o necessitem, tais como editores e visualizadores de extensos conteúdos. A escolha da língua também poderá ser interessante, e deverá ser considerada.

Conclui-se que este projeto atingiu o seu principal objetivo, que é o de criar uma alternativa de baixo custo aos meios tradicionais de controlo do computador, recorrendo aos movimentos da cara do utilizador. O projeto está disponível em https://github.com/MigMart/Usertracking.

Bibliografia

[1] Viola, P., Jones, M., "Robust real-time face detection," *In Proc. 8th IEEE Int. Conf. on Computer Vision*, vol.2, pp.747-747, 2001

[2] OpenCV http://opencv.org/, consultado em 23 de dezembro 2014.

[3 Processing https://processing.org/, consultado em 23 de dezembro 2014.

[4] Top 10 Programming Languages, Spectrum's 2014 Ranking http://spectrum.ieee.org/computing/software/top-10-programming-languages, consultado em 23 de dezembro 2014.

[5] Li, S.Z., Zhang, Z., "FloatBoost learning and statistical face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol.26, no.9, pp.1112-1123, 2004

[6] Schneiderman, H., Kanade, T., "A statistical method for 3D object detection applied to faces and cars," *In Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, vol.1, pp.746-751, 2000

[7] Schneiderman, H., "Learning a restricted Bayesian network for object detection," *In Proc. of the 2004 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition,* vol.2, pp.639-646, 27June-2 July, 2004

[8] Guang, C., Wen-wei, W., Qiu-ping, Z., "A Face Detector Based on Hausdorff Distance," *In Proc. 5th Int. Conf. on Wireless Communications, Networking and Mobile Computing,* pp.1-4, 24-26 Sept., 2009

[9] Srisuk, S., Kuratach, W., "New robust Hausdorff distance-based face detection," *In Proc. Int. Conf. on Image Processing,* vol.1, pp.1022-1025, 2001

[10] Alajel, K.M., Xiang, W., Leis, J., "Face detection based on skin color modeling and modified Hausdorff distance," *In Proc. IEEE Conf. Consumer Communications and Networking* , pp.399-404, 9-12 Jan., 2011

[11] Sim, D. G., Kwon, O. K., Park, R. H., "Object matching algorithms using robust Hausdorff distance measures," *IEEE Transactions on Image Processing,* vol.8, no.3, pp.425-429, 1999

[12] Nanni, L., Lumini, A., Dominio, F., Zanuttigh, P. "Effective and precise face detection based on color and depth data". *Applied Computing and Informatics,* 2014

[13] Wang, Y.X., Lo, L.Y., Hu, M.C. “Eat as much as you can: a kinect-based facial rehabilitation game based on mouth and tongue movements”. *In Proc. of the ACM Int. Conf. on Multimedia,* pp. 743-744, 2014

[14] Lee, K. K., Cham, W. K., Chen, Q. R., “Chin contour estimation using modified Canny edge detector,” *In Proc. 7th Int. Conf. on Control, Automation, Robotics and Vision,* vol.2, pp.770-775, 2-5 Dec. 2002

[15] Chen, Q. R., Cham, W. K., Lee, K. K., “Chin contour estimation for face recognition,” *In Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, vol.6, pp.761-764, 6-10 April, 2003

[16] Do, T.T., Doan, K.N., Le, T.H., Le, B.H. “Boosted of Haar-like features and local binary pattern based face detection,” *In Proc. Int. Conf. on Computing and Communication Technologies,* pp.1-8, 13-17 July, 2009

[17] Chen, Y. L., Kuo, T. S., Chang, W. H., Lai, J. S., “A novel position sensors-controlled computer mouse for the disabled,” *In Proc. of the 22nd Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society,* vol.3, pp.2263-2266, 2000.

[18] Nikisins, O., Greitans, M., “Local binary patterns and neural network based technique for robust face detection and localization,” *In Proc. of the Int. Conf. of the Biometrics Special Interest Group*, pp.1-6, 6-7 Sept., 2012

[19] Lim, J., Kim, Y., Paik, J. “Comparative analysis of Wavelet- based scale-invariant feature extraction using different Wavelet bases,” *Int. Journal of Signal Processing, Image Processing and Pattern Recognition,* vol. 2, no. 4, 2009

[20] Vezhnevets, V., Sazonov, V., Andreeva, A., “A Survey on pixel-based skin color detection techniques”, *In Proc. Graphicon, v*ol. 3, pp. 85-92, 2003

[21] Kovac, J., Peer, P., Solina, F., “Human skin color clustering for face detection," In Proc. *IEEE EUROCON 2003,* vol.2, pp.144-148, 22-24 Sept., 2003

[22] Brand, J., Mason, J.S., “A comparative assessment of three approaches to pixel-level human skin-detection,” *In Proc. 15th Int. Conf. on Pattern Recognition,* vol.1, pp.1056-1059, 2000

[23] Li, Y., Gong, S., Liddell, H., “Support vector regression and classification based multi-view face detection and recognition," *In Proc. 4th IEEE Int. Conf. on Automatic Face and Gesture Recognition,* pp.300-305, 2000

[24] Wang, P., Ji, Q., “Multi-view face detection under complex scene based on combined SVMs,” *In Proc. of the 17th Int. Conf. on Pattern Recognition,* vol.4, pp.179-182, 23-26 Aug., 2004

[25] Mahoor, M.H., Abdel-Mottaleb, M., “Facial features extraction in color images using enhanced active shape model,” *In Proc. 7th Int. Conf. on Automatic Face and Gesture Recognition,* pp.148-153, 2-6 April, 2006

[26] Lee, C.J., Chen, T.Y., Sun, D.G., Yang, T.N., Chang, A.Y., “Combining Gradientfaces, principal component analysis, and Fisher linear discriminant for face recognition,” *In Proc. 6th Int. Conf. on Networked Computing and Advanced Information Management*, pp.622-625, 16-18 Aug., 2010

[27] El Aroussi, M., Ghouzali, S., Rziza, M., Aboutajdine, D., El Hassouni, M., “Face recognition using enhanced fisher linear discriminant,” *In Proc. 5th Int. Conf. on Signal-Image Technology & Internet-Based Systems*, pp.48-53, Nov.29-Dec.4, 2009

[28] Xiong, X., Cai, Q., Liu, Z., Zhang, Z. “Eye gaze tracking using an RGBD camera: a comparison with a RGB solution.” *In Proc. of the 2014 ACM Int. Joint Conf. on Pervasive and Ubiquitous Computing: Adjunct Publication*, pp. 1113-1121, 2014

[29] Malawski, F., Kwolek, B., Sako, S. “Using Kinect for Facial Expression Recognition under Varying Poses and Illumination.” *Active Media Technology, Springer International Publishing*, pp. 395-406, 2014

[30] Kennedy, J., Eberhart, R., “Particle swarm optimization,” *In Proc. IEEE Int. Conf. on Neural Networks*, vol.4, pp.1942-1948, 1995

[31] Padeleris, P., Zabulis, X., Argyros, A.A., “Head pose estimation on depth data based on Particle Swarm Optimization,” *In Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, pp.42-49, 16-21 June, 2012

[32] Xu, M., Akatsuka, T., “Detecting head pose from stereo image sequence for active face recognition,” *In Proc. 3rd IEEE Int. Conf. on Automatic Face and Gesture Recognition,* pp.82-87, 14-16 Apr, 1998

[33] Perakis, P., Passalis, G., Theoharis, T., Kakadiaris, I.A., “3D Facial landmark detection under large yaw and expression variations," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol.35, no.7, pp.1552-1564, 2013

[34] Matsumoto, Y., Zelinsky, A., “An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement,” *In Proc. 4th IEEE Int. Conf. on Automatic Face and Gesture Recognition,* pp.499-504, 2000

[35] Liang, G., Zha, H., Liu, H. “Affine correspondence based head pose estimation for a sequence of images by using a 3D model,” *In Proc. 6th IEEE Int. Conf. on Automatic Face and Gesture Recognition,* pp.632-637, 17-19 May, 2004

[36] Idan, M., “Estimation of Rodrigues parameters from vector observations," *IEEE Transactions on Aerospace and Electronic Systems,* vol.32, no.2, pp.578-586, 1996

[37] Sorgi, L., “Two-view geometry estimation using the Rodrigues rotation formula," *In Proc. 18th IEEE Int. Conf. on Image Processing*, pp.1009-1012, 11-14 Sept., 2011

[38] Zhou, M., Liang, L., Sun, J., Wang, Y., “AAM based face tracking with temporal matching and face segmentation,” *In Proc. IEEE Conf. on Computer Vision and Pattern Recognition,* pp.701-708, 13-18 June, 2010

[39] Cootes, T.F., Taylor, C.J., Lanitis, A., Cooper, D.H., Graham, J., “Building and using flexible models incorporating grey-level information,” *In Proc. 4th Int. Conf. on Computer Vision*, pp.242-246, 11-14 May, 1993

[40] Zhu, J., Liu, Y., Zhang, L., “3D face reconstruction based on geometric transformation,” *In Proc. Int. Conf. on Virtual Reality and Visualization*, pp.46-49, 14-15 Sept., 2012

[41] Han, B., Lee, S., Yang, H. S., “Head pose estimation using image abstraction and local directional quaternary patterns for multiclass classification.” *Pattern Recognition Letters*, vol. 45, pp. 145-153, 2014

[42] Kondori, F. A., Yousefi, S., Li, H. “Direct three-dimensional head pose estimation from Kinect-type sensors.” *Electronics Letters*, vol. 50, no. 4, pp. 268-270, 2014

[43] Cui, Y., Schuon, S., Chan, D., Thrun, S., Theobalt, C., “3D shape scanning with a time-of-flight camera,” *In Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp.1173-1180, 13-18 June, 2010

[44] Cui, Y., Schuon, S., Thrun, S., Stricker, D., Theobalt, C., “Algorithms for 3D shape scanning with a depth camera,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol.35, no.5, pp.1039-1050, 2013

[45] Horwood, J.M.K., Thurley, R.W.F., Belmont, M.R., Baker, J., “Shallow angle LIDAR for wave measurement,” *In Proc. Oceans 2005 - Europe*, vol.2, pp.1151-1154, 20-23 June, 2005

[46] Gueuning, F.E., Varlan, M., Eugne, C.E., Dupuis, P., “Accurate distance measurement by an autonomous ultrasonic system combining time-of-flight and phase-shift methods,” *IEEE Transactions on Instrumentation and Measurement,* vol.46, no.6, pp.1236-1240, 1997

[47] Angrisani, L., Baccigalupi, A., Schiano Lo Moriello, R., “Ultrasonic time-of-flight estimation through unscented Kalman filter,” *IEEE Transactions on Instrumentation and Measurement,* vol.55, no.4, pp.1077-1084, 2006

[48] Kawasaki, H., Horita, Y., Morinaga, H., Matugano, Y., Ono, S., Kimura, M., Takane, Y., “Structured light with coded aperture for wide range 3D measurement," *In Proc. 19th IEEE Int. Conf. on Image Processing,* pp.2777-2780, Sept. 30-Oct. 3, 2012

[49] Wang, J., Zhang, C., Zhu, W., Zhang, Z., Xiong, Z., Chou, P.A., "3D scene reconstruction by multiple structured-light based commodity depth cameras," *In Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp.5429-5432, 25-30 March, 2012

[50] Lu, K., Wang, X., Wang, Z., Wang, L., “Binocular stereo vision based on OpenCV," *In Proc. IET Int. Conf. on Smart and Sustainable City*, pp.1-4, 6-8 July, 2011

[51] Guangjun, L., Yueming H., Zhifu, L., Xuedong, W. “Human-computer interaction system based on binocular stereo vision,” *In Proc. 31st Chinese Control Conference*, pp.3640-3644, 25-27 July, 2012

[52] Yargic, A.; Dogan, M., "A lip reading application on MS Kinect camera," *Innovations in Intelligent Systems and Applications (INISTA), 2013 IEEE International Symposium on* , vol., no., pp.1,5, 19-21 June 2013

[53] Rajaee, S., “Web browser software with single switch interface for PDAs or computers,” *In Proc. of the IEEE 30th Annual Northeast Bioengineering Conf.*, pp. 253- 254, 17-18 April, 2004

[54] Connor, C., Yu, E., Magee, J., Cansizoglu, E., Epstein, S., Betke, M., “Movement and recovery analysis of a Mouse-Replacement interface for users with severe disabilities”, *In Proc. 5th Int. Conf. UAHCI 2009 (held as part of HCI Int. 2009)*, pp.493-502, 19-24 July, 2009

[55] CameraMouse, http://www.cameramouse.org, consultado em 10 de março 2014.

[56] Dynavox EyeMax System, http://www.dynavoxtech.com/products/eyemax/, consultado em 10 de março 2014.

[57] Toby-Churchill, Preço de Dynavox EyeMax System, http://www.toby-churchill.com/products/dynavox-eyemax-system/, consultado em 10, março 2014.

[58] Quha Zono Mouse, http://www.quha.com/products-2/zono/, consultado em 10 de março 2014.

[59] myGaze Eye Tracker, http://www.mygaze.com/products/mygaze-eye-tracker/, consultado em 10, março 2014.

[60] KinesicMouse, http://kinesicmouse.com/, consultado em 10 de março 2014.

[61] Suporte para Kinect, http://www.bigben-interactive.co.uk/produit/produit/id/4379, consultado em abril 2014

[62] Kinect for Windows Sensor, Components and Specifications, http://msdn.microsoft.com/en-us/library/jj131033.aspx, consultado em 10 de março 2014.

[63] ASUS Xtion PRO http://www.asus.com/pt/Multimedia/Xtion_PRO/, consultado em 23 de dezembro 2014.

[64] Skeletal Tracking, http://msdn.microsoft.com/en-us/library/hh973074.aspx, consultado em 10 de março 2014

[65] Dias, P., Sousa, T., Parracho, J., Cardoso, I., Monteiro, A., Sousa Santos, B., "Student projects Involving Novel Interaction with Large Displays," IEEE *Computer Graphics and Applications,* vol.34, no.2, pp.80-86, 2014

[66] OpenNI, https://github.com/OpenNI/OpenNI, , consultado em 26 de dezembro 2014

[67] OpenKinect, libfreenect, https://github.com/OpenKinect/libfreenect, consultado em 26 de dezembro 2014

[68] Visual Studio Express, http://www.visualstudio.com/en-us/products/visual-studio-express-vs.aspx, consultado em 26 de dezembro 2014

[69] Technical documentation and tools, http://www.microsoft.com/en-us/kinectforwindows/develop/downloads-docs.aspx, consultado em 15 abril 2014

[70] Zhou, M., Liang, L., Sun, J., Wang, Y., "AAM based face tracking with temporal matching and face segmentation," *In Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp.701-708, 13-18 Jun., 2010

[71] Cao, Z., Yin, Q., Tang, X., Sun, J., "Face recognition with learning-based descriptor," *In Proc. IEEE Conf. on Computer Vision and Pattern Recognition,* pp.2707-2714, 13-18 Jun., 2010

[72] Yin, Q., Tang, X., Sun, J., “An associate-predict model for face recognition,” In Proc. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp.497-504, 20-25 10 de Jun., 2011

[73] CANDIDE, parametrized face, http://www.icg.isy.liu.se/candide/, consultado em 10 de jul. 2014

[74] Helping Kinect, Recognize Faces, http://research.microsoft.com/en-us/news/features/Kinect®facereco-103111.aspx, , consultado em 10 de jul. 2014

[75] Lopes, João, "Ray Tracing", Instituto Superior Técnico, pp. 10-11, Abril 2013

[76] Motion Analysis and Object Tracking, http://docs.opencv.org/modules/video/doc/motion_analysis_and_object_tracking.html, consultado em 20 de abril 2014

[77] Kalman, R.E., “A New Approach to Linear Filtering and Prediction Problems”, *Journal of Basic Engineering Transactions of the ASME*, vol.82, no.series D, pp.35-45, 1960

[78] Microsoft Developer Network, Face Tracking, http://msdn.microsoft.com/en-us/library/jj130970.aspx, consultado em 20 mar. 2014

[79] How to: Simulate Mouse and Keyboard Events in Code, http://msdn.microsoft.com/en-us/library/ms171548(v=vs.110).aspx, consultado em 29 de Dezembro 2014

[80] Troelsen, A. Pro C# 5.0 and the. NET 4.5 Framework. Apress, 2012

[81] Microsoft Speech Platform SDK 11 Documentation, http://msdn.microsoft.com/en-us/library/dd266409(v=office.14).aspx, consultado em 29 de Dezembro 2014

[82] Runtime Languages for use with the Speech Platform SDK 11, http://msdn.microsoft.com/en-us/library/hh378476(v=office.14).aspx, consultado em 29 de Dezembro 2014

[83] Microsoft Developer Network, Speech, http://msdn.microsoft.com/en-us/library/jj131034.aspx, consultado em 29 de dezembro 2014

[84] Windows Dev Center, Mouse input functions: mouse_event function, http://msdn.microsoft.com/en-us/library/windows/desktop/ms646260(v=vs.85).aspx, consultado em 30 de Dezembro 2014

[85] Windows Dev Center, Cursor functions: SetCursorPos function, http://msdn.microsoft.com/en-us/library/windows/desktop/ms648394(v=vs.85).aspx, consultado em 30 de Dezembro 2014

[86] Windows Dev Center, Cursor functions: GetCursorPos function, http://msdn.microsoft.com/en-us/library/windows/desktop/ms648390(v=vs.85).aspx, consultado em 30 de Dezembro 2014

Anexo A

A Tab. A.1 seguinte ilustra a comparação das possíveis soluções de Hardware para desenvolver os sistema. A classificação de 1 a 3 corresponde respetivamente a “Mau”, “médio” e "Bom”, e “C.F.” a classificação final, corresponde a soma das classificações de cada um dos três pontos analisados: “Funcionalidades disponíveis”, Simplicidade de montagem e utilização” e “Custo de aquisição”.

Hipótese	**Funcionalidades disponíveis**		**Simp. de mont. e util.**		**Custo de aquisição**		**C.F.**
a) Câmara única junto ao monitor (webcam de baixo custo < 200€)	Boa qualidade de imagem obtida só com uma boa luz. Consegue-se um bom *frame rate,* acima de 20 fps. Como não possui a deteção de distância, obriga a implementar algoritmos de deteção de distância [44] (implementação e calibração demorada). Obriga a implementar algoritmo de deteção da posição e pose do utilizador.	1	Simples de montar. Poderá ser necessário um apoio, consoante o modelo do monitor. A calibração do sistema poderá tornar-se o verdadeiro obstáculo para um utilizador comum.	1	A câmara deverá rondar os 100,00 €. Dado que nem todas as câmaras têm forma de afixar flexível e adaptável aos diversos estilos de monitor que existem no mercado, deverá também ser considerado um suporte para afixar câmara.	2	4
b) Câmara única junto à cara do utilizador (webcam de baixo custo < 200€)	Boa qualidade de imagem obtida só com uma boa luz. Consegue-se um bom *frame rate,* acima de 20 fps. Como não possui a deteção de distância, e obriga a implementar algoritmos de deteção de distância (implementação e calibração demorada). Obriga a implementar algoritmo de deteção da posição e pose do utilizador.	1	É simples de montar, mas o peso das câmaras e cabos poderá trazer incómodo para o utilizador. Também obriga o utilizador a colocar, e a retirar o dispositivo, sempre que começa ou termina a utilização. Também poderá ser necessário estar constantemente a ser calibrado. A câmara terá que ser adaptada para ficar presa à cabeça da utilizador.	1	A câmara deverá rondar os 100,00 €. Deverá também ser considerado o suporte e todos os ajustes necessários para afixar a câmara.	1	3
c) Combinação das condições de (a) e (b), (webcam de baixo custo < 200€).	Para além das funcionalidades indicadas nas opções A e B, tem que ser levar em consideração que o consumo de CPU será muito superior, visto que terão que ser processadas duas fontes de imagem independentes. Esta carga de processamento poderá levar a um atraso nos resultados, tornando o sistema inutilizável.	1	Requer processos de calibração mais demorados e complexos. Terão que ser montados vários equipamentos independentes.	1	O preço duplica.	1	3

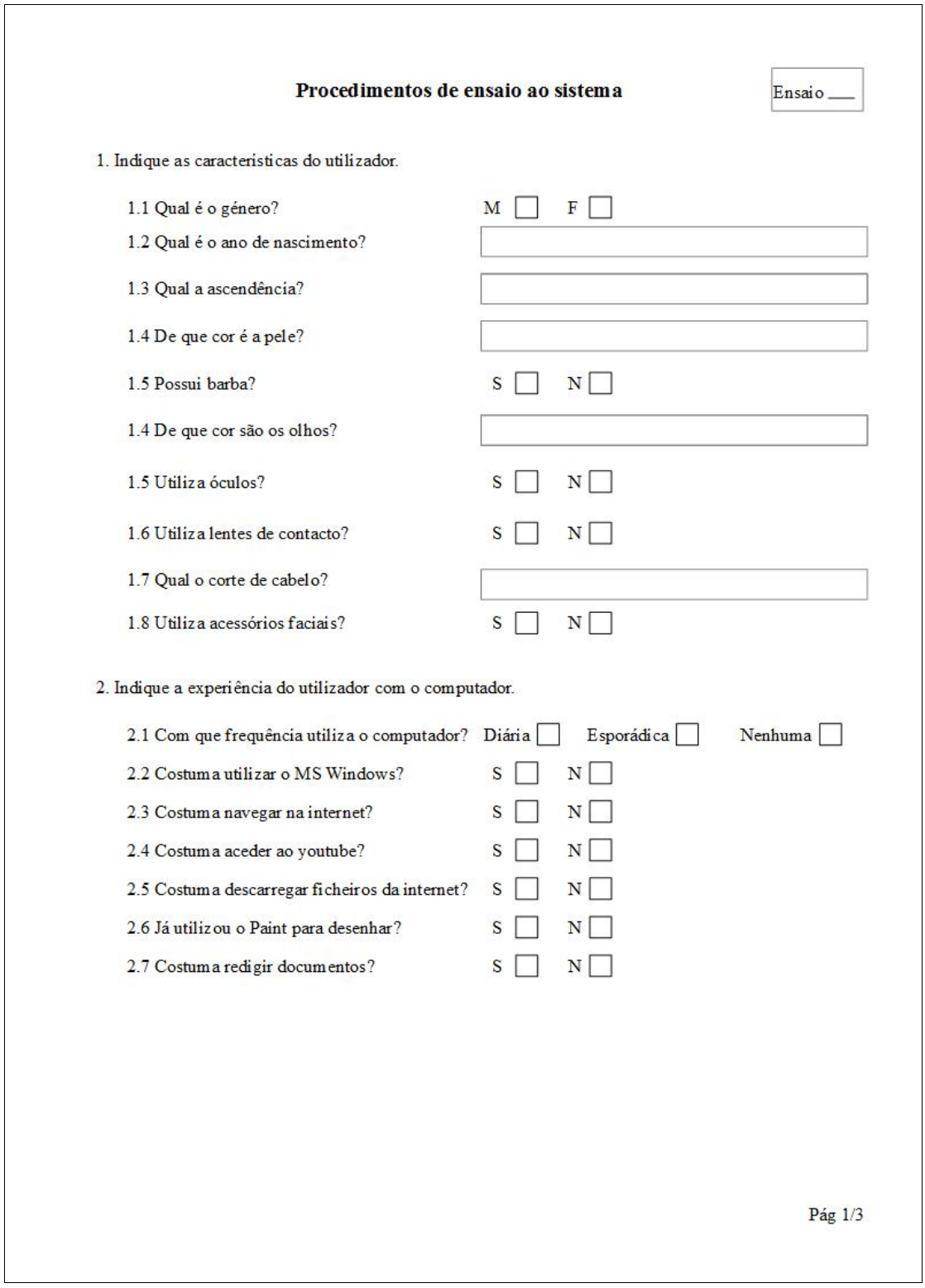
Tabela A.1 (cont.): Comparação das possíveis soluções de hardware. Classificação de 1 a 3 correspondendo de "Mau" a "Bom", respetivamente.

Hipótese	Funcionalidades disponíveis		Simp. de mont. e util.		Custo de aquisição		C.F.
d) Duas câmaras RGB junto ao monitor e nenhuma junto da cara do utilizador (webcam de baixo custo < 200€).	Idêntico à hipótese C.	1	Apesar de serem duas câmaras, estas podem ser acopladas como se tratasse de uma única estrutura, conforme descrito por Mrovlje-Vrančić [45]. Dado que requer uma estrutura própria, poderá tornar-se de difícil adaptação aos monitores e aos espaços disponíveis.	2	São necessárias duas câmaras logo o preço de aquisição deverá rondar os 200€. Será necessário criar uma estrutura que suporte as duas câmaras, e que se consiga colocar atrás de um monitor.	2	5
e) Um único sensor RGB-D, e.g. Microsoft Kinect junto ao monitor. (Kinect < 200€).	Boa qualidade de imagem obtida só com uma boa luz. Consegue-se um bom *frame rate* acima de 20 fps. Tem no entanto sensores incorporados para medir distâncias. Consegue-se uma imagem por infravermelhos. Possui API própria que disponibiliza rotinas para deteção de pessoas. Consome demasiado CPU.	3	Reúne todos os componentes numa única peça. Requer um apoio que se adapte aos vários modelos de monitor. É leve e simples de manipular.	3	O preço de aquisição ronda os 200€. O apoio ronda os 10€.	2	8

Tabela A.1 (cont.): Comparação das possíveis soluções de hardware. Classificação de 1 a 3 correspondendo de "Mau" a "Bom", respetivamente.

Anexo B

A Fig. B.1 ilustra o inquérito apresentado aos utilizadores que testaram o protótipo desenvolvido.



Procedimentos de ensaio ao sistema

Ensaio ___

1. Indique as caracteristicas do utilizador.

1.1 Qual é o género? M ☐ F ☐

1.2 Qual é o ano de nascimento? ☐

1.3 Qual a ascendência? ☐

1.4 De que cor é a pele? ☐

1.5 Possui barba? S ☐ N ☐

1.4 De que cor são os olhos? ☐

1.5 Utiliza óculos? S ☐ N ☐

1.6 Utiliza lentes de contacto? S ☐ N ☐

1.7 Qual o corte de cabelo? ☐

1.8 Utiliza acessórios faciais? S ☐ N ☐

2. Indique a experiência do utilizador com o computador.

2.1 Com que frequência utiliza o computador? Diária ☐ Esporádica ☐ Nenhuma ☐

2.2 Costuma utilizar o MS Windows? S ☐ N ☐

2.3 Costuma navegar na internet? S ☐ N ☐

2.4 Costuma aceder ao youtube? S ☐ N ☐

2.5 Costuma descarregar ficheiros da internet? S ☐ N ☐

2.6 Já utilizou o Paint para desenhar? S ☐ N ☐

2.7 Costuma redigir documentos? S ☐ N ☐

Pág 1/3

Figura B.1: Inquerito apresentado aos utilizadores do UserTracking.

Procedimentos de ensaio ao sistema Ensaio ___

	Duração
3.1. Crie uma pasta no ambiente de trabalho com o nome "Ensaio XXX", onde XXX é o número do ensaio.	
3.2. Abra um browser.	
3.3. Ouça música no youtube seguindo as instruções das próximas alíneas.	
3.3.1. Aceda ao endereço "www.youtube.com".	
3.3.2. Escreva "saxophone" na caixa de procura.	
3.3.3. Clique numa das opções e deixe tocar.	
3.4. Descarregue um ficheiro da internet seguindo as instruções das próximas alíneas.	
3.4.1. Abra um novo separador.	
3.4.2. Aceda ao endereço "www.ualg.pt".	
3.4.3. Clique na opção "INSTITUCIONAL".	
3.4.4. Clique na opção "IDENTIDADE VISUAL".	
3.4.5. Clique na opção "Marcas Gráficas".	
3.4.6. Clique na opção "INSTITUTO SUPERIOR DE ENGENHARIA".	
3.4.7. Grave o ficheiro na pasta "Ensaio XXX".	
3.5. Extraia o ficheiro gravado seguindo as próximas instruções.	
3.5.1. Coloque o ponteiro do rato por cima do ficheiro ise.zip e faça clique direito.	
3.5.2. Selecione a opção "Extrair...".	
3.5.3. Clique novamente na opção "Extrair".	
3.6. Siga as próximas instruções para fazer um desenho.	
3.6.1. Aceda à pasta ISE (duplo clique esquerdo).	
3.6.2. Clique direito em cima do ficheiro "logo_cor.jpg" e "Editar".	
3.6.3. Redimensione a área de desenho para o triplo do tamanho e desenhe uma cara (2 círculos para os olhos e um linha que representa uma boca, e outro circulo em torno dos olhos e boca).	
3.6.4. Grave na pasta "Ensaio XXX" com o nome "desenho", e feche o programa.	

Figura B.1 (cont.): Inquerito apresentado aos utilizadores do UserTracking.

Procedimentos de ensaio ao sistema Ensaio ___

	Duração
3.7. Proceda agora à criação de um documento de texto seguindo as próximas instruções.	
3.7.1. Aceda ao editor de texto (Word ou Writer).	
3.7.2. Escreva a frase: "Esta foi a primeira linha.".	
3.7.3. Mude de parágrafo, e escreva a frase: "Esta foi a segunda linha.".	
3.7.4 Mude de parágrafo, e escreva a frase: "Esta foi a terceira linha.".	
3.7.5 Seleccione todo o texto e altere a fonte para Arial, tamanho 16, e cor azul.	
3.7.6 Grave o ficheiro na pasta "Ensaio XXX" com o nome "texto".	
3.7.7 Feche o programa.	
3.8. Feche o browser.	

4. Expresse a satisfação e opinião relativamente ao ensaio realizado.

4.1. O que achou do sistema?

4.2. Na sua opinião o que deve ser melhorado?

Obrigado pela participação.

Figura B.1 (cont.): Inquerito apresentado aos utilizadores do UserTracking.