UNIVERSIDADE DA BEIRA INTERIOR
Covilhã | Portugal

Realidade Aumentada via Browser

Luís Filipe Aleixo Silva

Submetida à Universidade da Beira Interior para obtenção do grau de
Mestre em Engenharia Informática

Orientador: Prof. Doutor Frutuoso Silva

Departamento de Informática
Universidade da Beira Interior
Covilhã, Portugal
http://www.di.ubi.pt

Agradecimentos

Esta investigação nunca poderia ter sido realizada sem o apoio incondicional dos meus pais e irmã, que fizeram inúmeros sacrifícios para o bem da minha educação. Para tal, quero lhes agradecer de forma especial todo este apoio e carinho, que foram essenciais para a elaboração da minha dissertação de mestrado.

Quero agradecer também ao meu orientador científico, o Professor Doutor Frutuoso Silva, que me apoiou neste projecto desde o início e que sempre teve a disponibilidade para me aconselhar e direccionar da melhor forma no meu trabalho. Sem a sua valiosa ajuda, conhecimento, força, motivação e entusiasmo esta dissertação de mestrado não seria a mesma!

Também agradeço ao Professor Doutor Abel Gomes pelo seu apoiou e motivação ao longo desta minha fase académica, assim como ao IT (Instituto de Telecomunicações) da Universidade da Beira Interior pela bolsa e projecto de investigação apresentado nesta tese, e também as instalações e equipamento cedido para a elaboração do mesmo.

Agradeço de forma especial à minha noiva Lia Lourenço, que me acompanhou mais de perto no meu trabalho e que sempre esteve pronta para me ajudar naquilo que eu precisava. Sem a tua força, encorajamento e compreensão este trabalho não teria sido possível!

Quero também agradecer aos meus amigos e família que me ajudaram muito e foram para mim um suporte emocional muito valioso. Ordem alfabética: Avelino Pimentel, Carlinda Rio, Carlos Rocha, Débora Lourenço, Elisa Costa, Fábio Beirão, Fátima Rangel, João Cardoso, João Vila Cardoso, Luís Simões, Miguel Senos, Ruben Costa, Rui Franco, Teresa Anes, entre muitos outros.

À minha Mãe, não existem palavras que lhe consigam agradecer.

"Nada há como começar para ver como é árduo concluir."[Victor Marie Hugo]

Resumo

As tecnologias têm a capacidade de nos ajudar e informar ou então entreter e deslumbrar.

A Realidade Aumentada é uma tecnologia que permite enriquecer o ambiente real com objectos virtuais. Ela pode ser utilizada em diversas áreas e para diversos fins. Actualmente assiste-se a uma crescente utilização desta tecnologia, por exemplo na Internet e também em dispositivos móveis. Mas a sua massificação obriga a que esta se torne cada vez mais simples para que a generalidade das pessoas a possa usar.

Nesta tese apresenta-se o estudo feito sobre Realidade Aumentada e sua implementação via browser, onde se identificam as fases e os conceitos essenciais para a construção de um sistema de Realidade Aumentada. Assim foram desenvolvidas algumas aplicações de teste para a Web, as quais foram desenvolvidas usando as ferramentas PaperVision3D e FLARToolKit. Foi também criada uma aplicação de Realidade Aumentada que não recorre às tradicionais marcas fiduciais para manipular os modelos virtuais. Para isso foi necessário implementar outros métodos para posicionar e manipular os modelos virtuais, tais como a detecção da cara do utilizador e a detecção de movimentos.

Com esta nova abordagem obtiveram-se resultados muito satisfatórios, tendo sido desenvolvido um jogo simples que ilustra as potencialidades desta nova abordagem. Foi ainda testada a utilização de modelos com vários níveis de detalhe neste tipo de aplicações com vista ao aumento do seu desempenho.

Palavras-chave

Realidade Aumentada, Augmented Reality, Realidade Virtual, Virtual Reality, Interactividade, Web Browser, ARToolKit, FLARToolKit, Papervision3D, Detecção de Movimentos, Detecção da Cara, Interface Gestual, Níveis de Detalhe, LOD's, Level Of Details, Marcas, Fiducial Markers, Webcam.

Conteúdo

Lista de Figuras

Acrónimos

RA Realidade Aumentada

RV Realidade Virtual

HMD Head-Mounted Displays

PDA Personal Digital Assistant

GPS Geo-Posicionamento por Satélite

ISO International Organization for Standardization

GPS Geo-Posicionamento por Satélite

LOD Level Of Detail

COLLADA COLLAborative Design Activity

FPS Frames Por Segundo

SWF ShockWave Flash

XML eXtensible Markup Language

API Application Programming Interface

Capítulo 1

INTRODUÇÃO

A área da Realidade Virtual, mais propriamente da RA, foi desenvolvida com o intuito de representar e acrescentar algo imaginário ao mundo real. Pois desde sempre o ser humano tenta expressar de diversas formas aquilo que imagina. A evolução computacional permite então dar vida a desenhos, figuras, pinturas e todos os elementos possíveis de uma representação multimédia. Assim, numa sociedade que vive marcada pelo desenvolvimento de novos meios tecnológicos, as limitações existentes na representação têm vindo a ser ultrapassadas. Pois representar conceitos e ideias transcendentes às simples verbalizações, imagens ou maquetes torna-se mais fácil recorrendo à realidade virtual e/ou aumentada.

Uma pergunta que entre muitas pessoas se torna imperativo sobre esta área é: "O que é realmente virtual?". A resposta é encontrada na percepção ou captação dos sentidos humanos, sobre algo que é criado computacionalmente. Essa criação pode ser cruzada com o real ou então ser independente num ambiente virtual, daí advém a designação de realidade virtual. Mas será possível fazer a separação do real e do virtual? A resposta até ao momento é não, pois é necessária a existência de um observador (pessoa), para a vivência dessa realidade virtual e este observador estará sempre na parte real do ambiente criado. Mesmo que o sistema tente ser "totalmente" imersivo com headsets, óculos, luvas, sensores de pressão (input e output), passadeiras interactivas, etc, nunca será totalmente imersivo pois o observador não é integralmente transportado para esse mundo virtual. Então verifica-se que a realidade virtual é uma relação paradoxal com o real.

O conceito de realidade virtual dá a possibilidade de desenvolvimento de enu-

meras funcionalidades e permite criar elementos que no mundo real seriam inviáveis. A programação desses elementos, permite um cenário virtual onde o utilizador faz uso dos seus sentidos para o manipular/experimentar. A ligação estabelecida entre o utilizador e um cenário virtual, permite também simular situações reais ou mesmo impossíveis de acontecerem no ambiente real.

Consequência da evolução tecnológica por parte dos sistemas virtuais, levou a que estes sejam utilizados para demonstrar informação e conteúdos. A Internet é uma das áreas onde estes sistemas têm um grande crescimento, com o objectivo de cativar o público alvo. Pois são utilizados por empresas, comércio, entretenimento e outros ramos, através da inovação e criatividade desses mesmo sistemas.

A Realidade Aumentada pode ser vista como um dos sistemas utilizados na divulgação, promoção e demonstração de conteúdos. Mais que um sistema virtual este é um sistema tridimensional, o que eleva ainda mais as capacidades representativas. Aliada à Internet, a RA tornou-se uma tecnologia aliciante, de potencial evolutivo, que desperta a curiosidade dos seus utilizadores.

1.1 Motivação

A RA é uma tecnologia em crescimento, quer na investigação, quer na utilização em diversas áreas. Por esse motivo ela apresenta desafios a nível de desenvolvimento e de uma utilização simples. Daí o meu interesse em conhecer esta tecnologia e o desenvolvimento de aplicações de RA para a Internet e as suas forma de interacção. Ficando assim a par dos principais problemas subjacentes.

Para além da interacção com o sistema, existe a necessidade de facilitar o acesso a esta tecnologia, tornando-a cada vez mais acessível aos utilizadores.

1.2 Objectivos

Os objectivos propostos foram:

- Analisar e descrever conceitos, bibliotecas, ferramentas e tecnologias inerentes à RA. Assim como os mecanismos e procedimentos utilizados para a criação de aplicações de RA.
- Criação de aplicações de RA para executar via browser através da Internet.
- Utilização de modelos com vários níveis de detalhe em aplicações de RA via browser.
- Desenvolver novos métodos de interacção com os objectos, sem recorrer às convencionais marcas, que sejam intuitivos e de fácil utilização.

1.3 Estrutura da Tese

Neste capítulo faz-se uma introdução ao tema, apresentam-se os objectivos do trabalho e a estrutura da tese.

No capítulo 2 apresenta-se a Realidade Aumentada e as formas de interacção usadas neste tipo de aplicações. Além disso faz uma breve descrição do trabalho relacionado e das ferramentas normalmente utilizadas para o desenvolvimento da RA.

No capítulo 3 faz-se a descrição do trabalho desenvolvido nesta tese, nomeadamente a interacção com marcas e sem marcas e o uso de modelos com vários níveis de detalhe.

No capítulo 4 são apresentados os apresentados os principais resultados das aplicações de RA desenvolvidas.

No capítulo 5 são apresentadas algumas conclusões sobre o trabalho desenvolvido e perspectivado o trabalho futuro.

Capítulo 2

ESTADO DA ARTE

2.1 Realidade Aumentada

A Realidade Aumentada é proveniente da realidade virtual. A realidade virtual não é uma tecnologia recente, pois em 1968 na Universidade de Harvard, Ivan Sutherland criou uma das primeiras aplicações de realidade virtual. Com essa criação, surgiram várias sub-áreas de investigação, entre as quais, a RA. Foi também Ivan Sutherland, que desenvolveu o primeiro hardware de visualização imersivo intitulado "Sword Damocles" [5], um equipamento muito pesado suspenso no tecto (ver a figura 2.2a). A sua evolução levou à criação dos actuais head-mounted display (capacetes de realidade virtual).

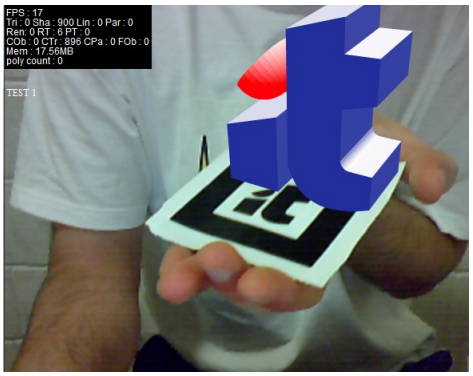
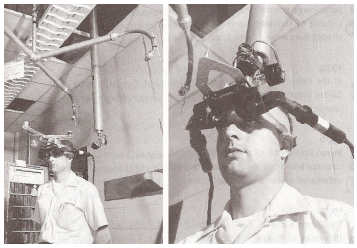


Figura 2.1: Um exemplo de realidade aumentada.

Uma definição simples da RA, dada por Ronald Azuma *et al.* [6] é: Ao ambiente real é adicionado algo criado computacionalmente (virtual). Ou seja, a junção do virtual com o real (exemplo figura 2.1). Também foram criadas três propriedades



(a) 1ºHMD, criado por Ivan Sutherland

(b) Ecrãs de projecção [1].

(c) HMD com câmara para RA[2].

Figura 2.2: Imagens de Head mounted displays (equipamentos imersivos).

para sustentar essa definição:

- Os objectos virtuais são colocados no ambiente real (nunca o contrário);
- A interacção e visualização são feitas em tempo real;
- Tem de existir um posicionamento e alinhamento dos objectos virtuais, de acordo com o ambiente real.

O artigo de Ronald Azuma *et al.* [6], é um dos principais pontos de referência para o tema da RA. Esta publicação apresentou uma recolha bibliográfica detalhada, problemas que existiam e ainda existem na RA, o que levou a novas soluções para a RA.

A tecnologia de RA, permite colocar objectos virtuais no mundo real, mas a colocação só é possível se existir uma forma alternativa de visualização. Existem algumas formas de visualização da RA. Em seguida e através da enumeração feita por Bowman [7], são mostradas as diferentes categorias para a visualização da RA:

- Monitor/ecrã, onde é feita a projecção da RA com base na captura de uma câmara (ex: webcam).
- Head-Mounted Displays (HMD) ou capacete/óculos oclusivos, são menos utilizados que os monitores, devido ao seu custo elevado. Estes capacetes

[1]Óculos *EzVision*® Video Eyewear da empresa ezGear.
[2]HMD *VR Pro AR*® da empresa Virtual Realities.

são uma adaptação dos originais HMD utilizados na realidade virtual, que apenas mostravam o mundo virtual. Os HMD para a RA contêm (na grande parte deles) uma webcam adaptada na frente do equipamento, para capturar o ambiente onde o utilizador se encontra (ver figura 2.2c). A visualização das imagens geradas computacionalmente é feita em pequenos ecrãs no interior do equipamento (ver figura 2.2b). Baseada nessa tecnologia existe a tecnologia de projecção em lentes transparentes, o que permite ao utilizador ver através do equipamento (*See-Through*), ou seja, é um dispositivo não oclusivo.

- Existem também outros dispositivos imersivos da realidade virtual, que são utilizados pela RA. As *Workbenches* [8] são um exemplo disso, compostas por uma superfície de grandes dimensões, onde são projectados os objectos virtuais. Juntamente com o HMD, estas mesas permitem uma interacção e abordagem ao ambiente virtual de forma mais natural. Além disso, a projecção deixa de ser feita próxima dos olhos, o que permite resoluções mais elevadas.

Muitas das aplicações iniciais da RA foram desenvolvidas para HMD's, contudo é crescente o desenvolvimento para sistemas como: Personal Digital Assistant (PDA)s, projectores/monitores, telemóveis e outros que contêm dispositivos de visualização e um modo de captura. O objectivo é implementar a tecnologia de RA nos sistemas já utilizados pelas pessoas, pois é cada vez mais comum os utilizadores com as novas tecnologias (electrónicas e informáticas) terem ao seu dispor, uma webcam e um ecrã. O que leva os sistemas com HMD's a serem apenas utilizados em casos específicos. Este facto deve-se ao custo de aquisição, manutenção e serem rapidamente ultrapassados tecnologicamente.

Figura 2.3: Esquema criado por Milgram intitulado "Virtuality Continuum".

Existe a necessidade de se fazer uma clara distinção entre as realidades: virtual, aumentada e real. Para uma melhor síntese desta problemática, Milgram [9] fez um esquema classificativo com o nome de "Virtuality Continuum" (figura 2.3), onde

colocou o ambiente virtual e real em cada extremo. A separar esses dois pólos está a "Mix Reality", ou seja, a mistura dos dois ambientes. De forma clara, a RA tem apenas uma pequena parte de virtual em relação ao real. O inverso acontece para a "virtualidade aumentada"(Augmented Virtuality), onde a parte real é menor.

O mais importante para a distinção, são as principais diferenças entre a realidade aumentada e virtual. Uma das diferenças é a imersividade da aplicação, pois a RA tem apenas a função de "acrescentar" algo ao ambiente real. Já a Realidade Virtual (RV), tem de criar todo o ambiente para o utilizador, o que torna a imersividade mais importante. O aspecto visual é outra das diferenças, porque ao substituir o mundo real, a qualidade e o realismo das cenas virtuais são uma das prioridades da RV. Para a RA, o aspecto visual não é algo prioritário, mas sim o posicionamento e alinhamento dos objectos virtuais adicionados no ambiente real. Já na RV o posicionamento e alinhamento apenas tem de ter em conta a cena virtual. Por fim, na grande maioria dos casos existem diferenças entre o input da RA e da RV. No caso dos sistemas de RV, na sua maioria, apenas necessitam de comandos de entrada para a sua execução, como por exemplo, deslocar-se no ambiente e interagir com ele, através do teclado ou rato. Pelo contrário, na RA é necessário para além de possíveis comandos, também a recepção de imagens, sensores, Geo-Posicionamento por Satélite (GPS), entre outras entradas adaptadas ao sistema. Ou seja, implica uma grande quantidade de dados de input a serem processados. Esse processamento tem de ser rápido, para que a interacção e o novo alinhamento, sejam feitos em tempo real. Resumindo, a RA contém muitos dados de input, que têm de ser processados em tempo real, basta pensar no input do vídeo do ambiente real.

2.1.1 Ambiente Virtual

Na implementação do ambiente virtual nas aplicações desenvolvidas, foi utilizado o motor gráfico Papervision3D [3]. Sendo este um motor Open Source em evolução, existem muitas funcionalidades que necessitam de ser implementadas, em relação a outros motores gráficos.

Para uma melhor percepção do ambiente virtual e com base em [10], é possível explicar de forma sucinta, como funciona e é processado o espaço/ambiente tridimensional no Papervision3D. Este espaço tem o nome de cena (*Scene*), por outras

[3] http://code.google.com/p/papervision3d

palavras, é o ambiente virtual. Os objectos virtuais criados, apenas são visíveis quando colocados na cena, segundo um sistema de coordenadas cartesianas.

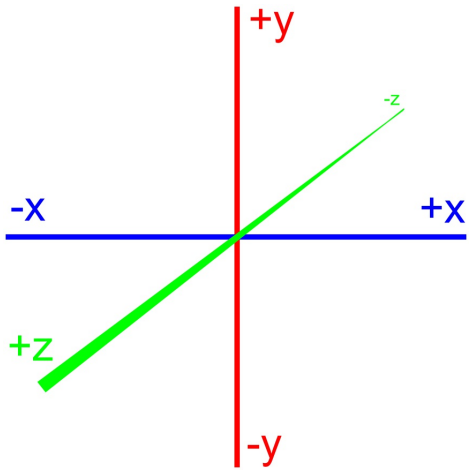


Figura 2.4: Sistema de coordenadas cartesianas, utilizadas pelo Papervision3D.

O Papervision3D utiliza o sistema de coordenadas cartesianas com a regra da "Mão direita" para a sua orientação. Consequência dessa utilização tridimensional (x, y, z), os cálculos espaciais têm como ponto de origem o centro $(0, 0, 0)$. Ou seja, a coordenada x fica encarregue do deslocamento para a esquerda se negativo e direita se positivo, o y para cima se positivo e baixo se negativo. O mesmo acontece com a coordenada z onde esta controla a profundidade, mais perto se negativa e mais distante se positiva. Tudo isto, se o sistema for visto com a mesma perspectiva que a figura 2.4.

As transformações de rotação, translação e de variação de escala, são possíveis de executar na cena através duma matriz de 4 por 4. Cada objecto contem também uma matriz individual, para as suas transformações. Por esse motivo tem o nome de matriz transformada.

Para a cena criada ser visível é necessário ainda a existência de pelo menos uma câmara. O Papervion3D contém por defeito as câmaras *target* (funciona direccionada num ponto da cena), free (contrariamente à *target* não tem direcção fixa), *debug* (permite a sua movimentação pela cena) e *spring* (segue um objecto, dando um efeito de 1ª , 2ª e 3ª pessoa), cada uma delas tem funcionalidades de zoom, foco e campo de visão (*FOV*) entre outras. É necessária, também, pelos menos uma luz que ilumine a cena, para que o seu conteúdo seja visível. No Papervion3D só existe um tipo de luz para a iluminação da cena (*PointLight3D*).

Por fim, a visualização da cena é feita através do *viewport*. Basicamente, será a lente da câmara ou então uma janela para a cena 3D, onde as suas dimensões vão definir a altura e a largura da visão sobre a cena.

2.2 Interacção com marcas

2.2.1 Funcionamento

Figura 2.5: Exemplo de uma marca utilizada em RA.

A RA é definida pela adição de objectos virtuais ao ambiente real, onde os objectos são posicionados segundo uma marca de referência [11]. Essa marca (ex: figura 2.5) vai ser procurada através da análise/segmentação à imagem capturada. Geralmente essa marca tem características específicas para uma melhor percepção/contraste do resto da imagem. É por esta razão que se utilizam mais frequentemente marcas a preto e branco, de forma quadrada e com símbolos no seu interior. Essas características ajudam a uma melhor diferenciação e identificação da marca. O tipo de marcas não é restrito apenas ao preto e branco, mas também ao uso de marcas com cores e diferentes formatos, para tentarem passar despercebidas e misturarem-se com o ambiente circundante ao observador.

Todas estas marcas referidas, têm um nome específico de *Fiducial markers* [12] e também são colocadas na classe das marcas passivas, pois não requerem uma fonte de energia para a sua utilização. Este conceito já se encontra fortemente implementado e é utilizado em estabelecimentos de venda, armazenamento, entre outros, através dos códigos de barras. Ou seja, os códigos de barras também são marcadores passivos *fiducials*, reconhecidos por um software, que os identifica pelo

seu "aspecto visual", daí advir o mesmo conceito para a RA. Porém, o sistema de código de barras apenas identifica o marcador, deixando de parte o posicionamento e orientação do mesmo. Já a RA baseada em marcas retém toda essa informação.

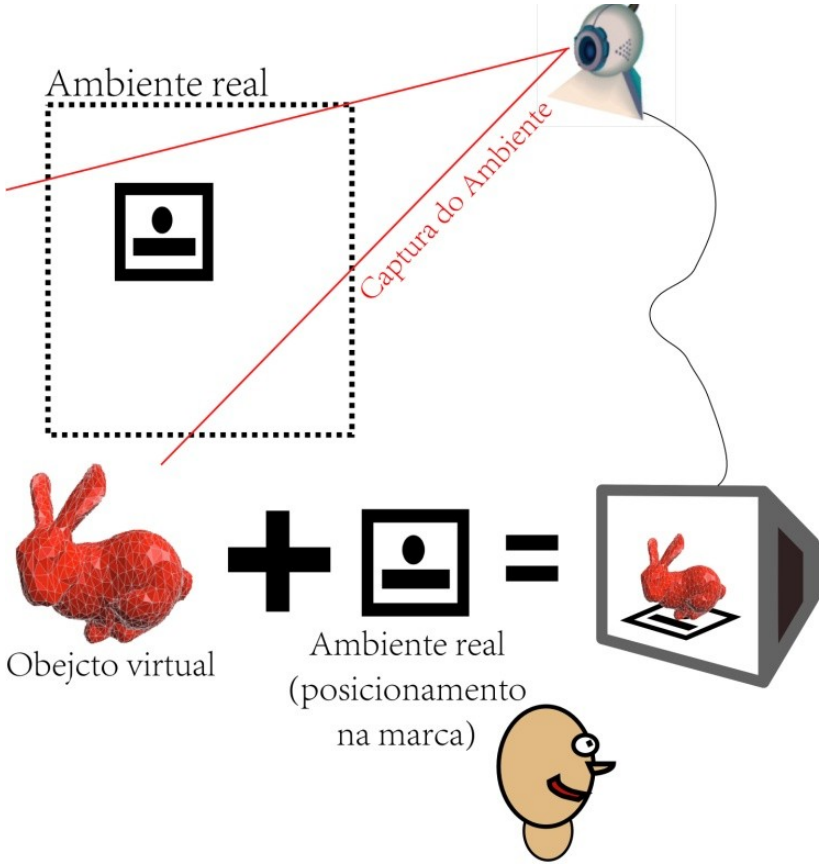


Figura 2.6: Esquema básico do processo de criação da RA.

De forma simples e em traços gerais, o processo de funcionamento de uma aplicação de RA é ilustrado na figura 2.6. Nesta figura existe um ambiente real onde está presente uma marca, este ambiente é capturado por uma câmara. É feita uma análise ao ambiente capturado para detectar a existência de marcas. Se for encontrada alguma marca equivalente ao *Template/Pattern* da aplicação, é carregado e colocado um modelo virtual (neste caso a malha dum coelho) alinhado com a marca. O resultado deste processo é perceptível ao utilizador, através dum dispositivo de visualização (ex: um monitor).

2.2.2 Posicionamento e orientação

Os objectos virtuais vão posicionar-se e orientar-se de acordo com as coordenadas (x, y, z) da marca. O cálculo das coordenadas x, y, é feito com os valores da "grelha de pixeis" da imagem capturada, com base na posição da marca na imagem. Estas coordenadas controlam o deslocamento no plano XY do objecto. Para o posicionamento em profundidade do objecto, o calculo da coordenada z vai basear-se na distância entre a marca e a fonte de captura (i.e. a câmara). Ou seja, a maneira mais fácil para calcular essa distância, é com base no tamanho da marca (maior tamanho mais perto e vice-versa). Na orientação do objecto é utilizada a matriz transformada já referida anteriormente. A matriz é calculada em relação ao ponto de captura(exemplo câmara), através do algoritmo [13] com base nas coordenadas (x, y, z) anteriores e na análise à imagem da marca capturada, como será explicado na secção 2.2.4. Com toda esta informação é possível fazer um mapeamento do ambiente 3D e recriá-lo sobre a imagem capturada.

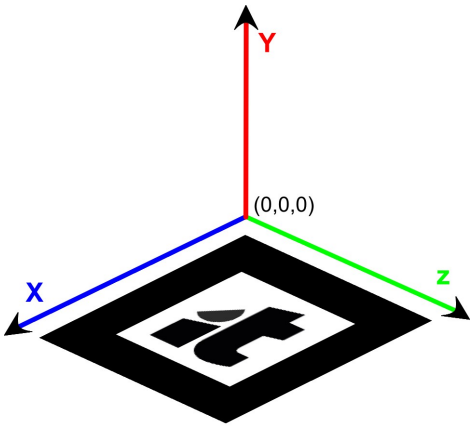


Figura 2.7: Representação possível de um sistema de coordenadas associado a uma marca.

Até agora falou-se das marcas como ponto de referência de um objecto(s) virtual. No entanto, estas servem também como modo de interacção entre o utilizador e o ambiente virtual, onde o utilizador ao mexer a marca, consegue deslocar o ponto de referência do objecto, o que leva por sua vez ao cálculo de novas coordenadas no ambiente virtual, Mas existem outros modos de interacção possíveis de associar à marca, por exemplo emular botões se numa sequência de marcas uma for tapada por algum tempo. Pode ser apenas identificadora ou um *trigger*, para quando é detectada na captura ser feita uma acção. Assim sendo, não é necessário associar sempre um objecto virtual à marca, para todos os tipos de interacção.

2.2.3 Tipos de marca

Como já foi referido na secção anterior, a RA necessita de pontos de referência para que seja possível a junção dos objectos virtuais com o ambiente real. Esses pontos de referência são na grande maioria marcas. Também, como referido anteriormente, estas podem ser separadas em marcas activas, que requerem fonte de energia e, em marcas passivas, que não requerem fonte de energia para a sua utilização. As utilizadas como método de estudo foram as passivas, onde podem existir marcas ilustradas e não ilustradas.

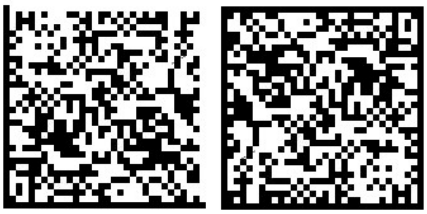


Figura 2.8: Marca DataMatrix à esquerda e VScode à direita.

A *DataMatrix* [14] é uma das marcas utilizadas na RA, ela está em grande crescimento em diversas áreas, para a substituição dos tradicionais códigos de barras. Estas marcas são símbolos standards da ISO para os novos códigos de barras 2D. Assim sendo, elas são aplicadas a vários tipos de produtos para identificação, tal como números de série. A marca *DataMatrix* é uma matriz binária, em que a representação dos zeros e uns, são feitos através de células de cores preto e branco (ver figura 2.8). A sua estrutura é quadrada ou rectangular, limitada por duas linhas solidadas a preto e duas linhas tracejadas com branco e preto, onde a sua capacidade máxima é de 3116 dígitos, 2335 caracteres ou 1556 bytes. Isto permite ter um grande conjunto de marcas de identificação na RA, devido ao número de combinações possíveis entre os elementos da matriz (254x254, 64516 identificadores). Utiliza também parte da matriz, para informação adicional. Por fim, é possível implementar níveis de segurança, através da codificação. Outra marca muito semelhante é a *VScode* [4], que difere no rebordo, com quatro linhas pretas e o armazenamento de 4151 bytes.

[4]http://www.veritecinc.com/vericode.html

Figura 2.9: Marca QRCode.

Também, no campo das *2D Barcodes* existe a *Quick Response Code* (*QRCode* [15] uma matriz bi-dimensional de células pretas e brancas. Esta foi criada em 1994 e apresenta uma rápida detecção e orientação da marca. Também tem a capacidade de armazenar caracteres alfanuméricos (4296), numéricos (7089) e Kanji japoneses (1817), com a totalidade de 2953 bytes armazenados. A sua estrutura é divida em 16 áreas e também apresenta 3 células distintas das normais, que permitem a orientação (ver figura 2.9).

Figura 2.10: Marca BCH ID.

O facto de muitas das aplicações desenvolvidas até ao momento serem para investigação, leva ao uso de marcas mais robustas e de fácil segmentação. São então utilizadas marcas como a *BCH ID* (*ID*), esta é a marca usada pela ferramenta ARToolKit[5] para desenvolvimento de aplicações de RA. Esta marca assemelha-se à DataMatrix, também constituída por uma estrutura de blocos pretos e brancos. A marca *ID* na sua última versão, tornar possível a representação de 4096 marcas [16] e a implementação do *algoritmo* CRC (*Cyclic redundancy check*) [17]. Este algoritmo faz a recuperação de informação danificada ou errada referente à captura da marca. O problema das marcas *DataMatrix* e *ID* é a qualidade da captura, pois se a imagem capturada tiver pouca resolução ou muito ruído, a segmentação da estrutura de dados fornecida pela marca não é eficaz.

[5]http://www.hitl.washington.edu/research/shared$_s$*pace/download/*

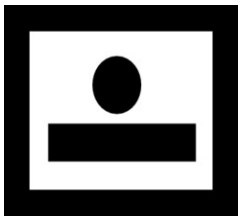


Figura 2.11: Marca Template.

As marcas *Template*, que são muito utilizadas pelas ferramentas de desenvolvimento. Estas possuem um rebordo a preto e uma figura no seu interior. As figuras mais utilizadas são: figuras geométricas ou outros tipos de formas simples (ver figura 2.11). Isso permite uma segmentação e comparação robusta, por parte do sistema e também um efeito visual mais agradável. Uma desvantagem na utilização de uma *Template*, é limitar a segmentação dum número reduzido de marcas, pois os símbolos/figuras utilizados no interior das marcas, não são estruturadas como as marcas anteriormente referidas.

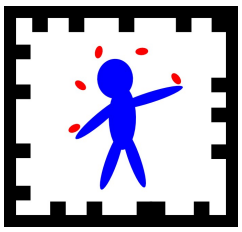


Figura 2.12: Marca Frame.

A nível de marcas ilustradas, existem as marcas *Frame* (figura 2.12) são marcas híbridas, pois possibilitam que no interior da marca existam imagens/ilustrações independentes à estrutura identificadora da marca em si. Esse interior pode ser utilizado para logotipos, imagens, fotos, etc. . . Assim a detecção da marca é feita com base no seu rebordo, através do contraste entre os pontos pretos e as imagens ou pontos brancos.

As marcas *Split* foram criadas e utilizadas pela Sony® para o jogo Eye of Judgment (figura 2.26). Estas utilizam basicamente o mesmo princípio que as marcas *Frame*, pois não utilizam o interior da marca para a segmentação, mas sim as suas margens. As *Split* utilizam apenas duas margens da marca, mas requerem um maior tamanho para a sua estrutura identificadora, em relação à marca *Frame*. Este aumento de tamanho serve para manter a qualidade da segmentação. Contudo as marcas *Spit*

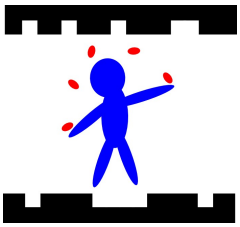


Figura 2.13: Marca Split.

têm uma vantagem no seu manuseamento, pois deixam dois lados nas marcas que podem ser tapados sem interferir na segmentação (ver figura 2.13).

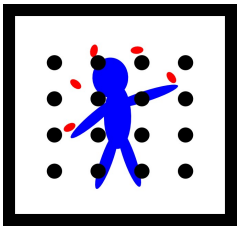


Figura 2.14: Marca Dots.

As marcas por pontos (*Dots*), não são muito utilizadas. A razão deve-se ao aparecimento das duas marcas referidas anteriormente. Ou seja, a marca *Dots* é implementada através da sobreposição de pontos circulares pretos em forma de grelha, numa superfície (ver figura 2.14). Essa superfície é independente dos pontos da marca, onde mais uma vez é possível utilizar imagens e texturas que possam atenuar e disfarçar a utilização das marcas. Mas o inconveniente desta técnica é ocultar parcialmente a imagem com a sobreposição dos pontos, ao contrário das duas marcas anteriores. Mesmo assim, esta solução tem utilização em mapas interactivos, onde esses pontos podem ser atenuados [18].

Para concluir, apesar da aparência mais ilustrativa e menos identificadora, em comparação com as marcas não ilustradas. Os resultados obtidos pelas marcas ilustradas, na segmentação/codificação da estrutura identificadora e detecção de erros, mostram que este tipo de marcas podem ser utilizadas com um bom desempenho.

2.2.4 Detecção das marcas

A utilização da marca torna a procura mais fácil na análise da imagem. Para isso, tem de existir o template da marca no sistema de análise, para servir de método comparativo para essa procura. É necessário que existam algumas condições para uma boa detecção da marca, tais como iluminação moderada, a não oclusão da marca, uma resolução da captura adequada à distancia pretendida para o uso da marca, entre outras. Quando são utilizadas grandes resoluções, apesar de tornarem a aplicação mais fiável e perceptível, o processamento das frames por segundo é mais elevado e pode influenciar a latência da apresentação das frames por segundo.

É possível ter mais do que uma marca no ambiente real, pois cada uma delas fica associada a um objecto virtual. Nos sistemas para a criação de aplicações de RA, as marcas são normalmente guardadas de forma individual como *Patterns* em ficheiros específicos. Nesse ficheiro é guardada a informação da marca, para ser incluída na aplicação e servir de comparação com as marcas capturadas. Essa informação na generalidade dos casos, encontra-se estruturada em forma de matrizes numéricas. São guardadas 4 orientações da marca(0^o, 90^o, 180^oe 270^o), para cada orientação existe 3 matrizes, correspondentes às cores vermelho, verde e azul (RGB).

O método de utilização de marcas teve uma abordagem mais séria por Rekimoto e Katashi [19], que mesmo sendo uma apresentação simples, era também eficaz. Alguns anos depois, os mesmos autores [20] mostram um dos algoritmos de reconhecimento de marcas em imagens, feito em 5 passos:

1. O primeiro passo do algoritmo é a binarização [21]. Este procedimento vai transformar as cores da imagem em diferentes níveis de cinzento. Seguidamente é definido um *threshold* (zona de separação), que permite a separação em duas zonas através da análise do histograma dos níveis de cinzento. Ou seja, a binarização converte os valores acima do *threshold* em píxeis brancos e os valores abaixo em píxeis pretos. Para que sejam obtidos resultados fiáveis deve-se fazer uma minimização da variância na separação entre os dois grupos. No caso das marcas utilizadas serem semelhantes à figura 2.5, o *threshold* vai ser mais fácil de definir, visto que o seu contraste é grande.

2. Depois de ser aplicada a binarização, o segundo passo tem como objectivo escolher as regiões a preto da nova imagem, que representam possíveis el-

ementos/marcas do ambiente capturado (ver figura 2.15). Para cada região escolhida, apenas as que apresentam a forma de polígono são seleccionadas. A selecção é feita com base nos ângulos entre os vértices do contorno da região. Para a optimização da procura deste processo, é feita a diminuição da consistência da imagem. Ou seja, a passagem pelos píxeis da imagem não é feita de forma contínua, mas com um intervalo de píxeis ignorados.

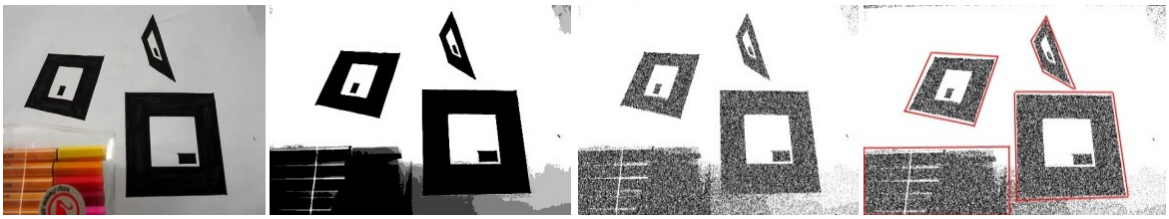


Figura 2.15: Processo de binarização e escolha das regiões para analisar.

3. O passo seguinte do algoritmo é analisar cada um dos polígonos seleccionados. No entanto, os polígonos não se encontram alinhados com o plano da captura. Pois as marcas encontram-se num ambiente 3D, com a possibilidade de diferentes perspectivas de colocação, em relação ao plano de captura 2D. É então necessário fazer a transformação dos polígonos para o plano da captura, recorrendo ao algoritmo de Homografia [22]. Este algoritmo vai fazer a correspondência de 4 pontos dum polígono seleccionado (plano da marca), com os 4 pontos dum polígono quadrado (plano da captura). A correspondência é feita com uma matriz homogénea não-singular e o plano do polígono ($C_i = H * P_i$ ver equação 2.1). x, y representam as coordenadas do plano da imagem capturada (C_i). x', y' representam as coordenadas do plano do polígono (P_i). h_{ij} são os valores da matriz homogénea(H).

$$C_i = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} * P_i \tag{2.1}$$

Ao fazer a correspondência, segundo a equação 2.1 e assumindo que $h_{33} = 1$, é possível construir as duas equações seguintes:

$$x'_i = (h_{11}x_i + h_{12}y_i + h_{13})/(h_{31}x_i + h_{32}y_i + 1) \Leftrightarrow x'(h_{31}x_i + h_{32}y_i + 1) = h_{11}x_i + h_{12}y_i + h_{13}; \tag{2.2}$$

$$y'_i = (h_{21}x_i + h_{22}y_i + h_{23})(h_{31}x_i + h_{32}y_i + 1) \Leftrightarrow y'(h_{31}x_i + h_{32}y_i + 1) = h_{21}x_i + h_{22}y_i + h_{23}; \tag{2.3}$$

Do resultado das quatro correspondências (equação 2.2 e 2.3) vão derivar oito equações, que permitem elaborar um sistema matricial apresentado na equação 2.4. Por fim resolve-se o sistema linear, onde se encontram os valores x, y do plano da captura, pois estes são valores fixos que eliminam o máximo de termos do sistema.

$$\begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \\ x'_4 \\ y'_4 \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 \\ x_3 & x_3 & 1 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y'_2x_2 & -y'_3y_3 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -y'_4x_4 & -y'_4y_4 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} \tag{2.4}$$

4. Encontrar uma possível marca nos polígonos convertidos para o plano da captura. Ao ter como resultado um quadrado equivalente ao existente na aplicação, com base na *Pattern/Template* da marca, é feita uma comparação entre os valores da matriz de píxeis do polígono e os valores correspondentes na matriz da marca. Tendo em conta que os valores do polígono já se encontram binarizados, a comparação é entre zeros e uns. A marca é validada se existir uma taxa de erro aceitável na comparação. Essa taxa de erro é calculada com o método *CRC-error* [23], devido à perda de informação quer por parte da conversão da homografia, quer por ruído da captura.

5. O último passo é calcular a orientação e o posicionamento da marca capturada. O resultado deste cálculo é obtido através do algoritmo [24]. Posteriormente, os valores são convertidos para o sistema tridimensional e transmitidos através

de uma matriz transformada para um motor gráfico, para este conseguir alinhar os objectos virtuais com a marca (se for esse o caso).

2.3 Trabalho Relacionado

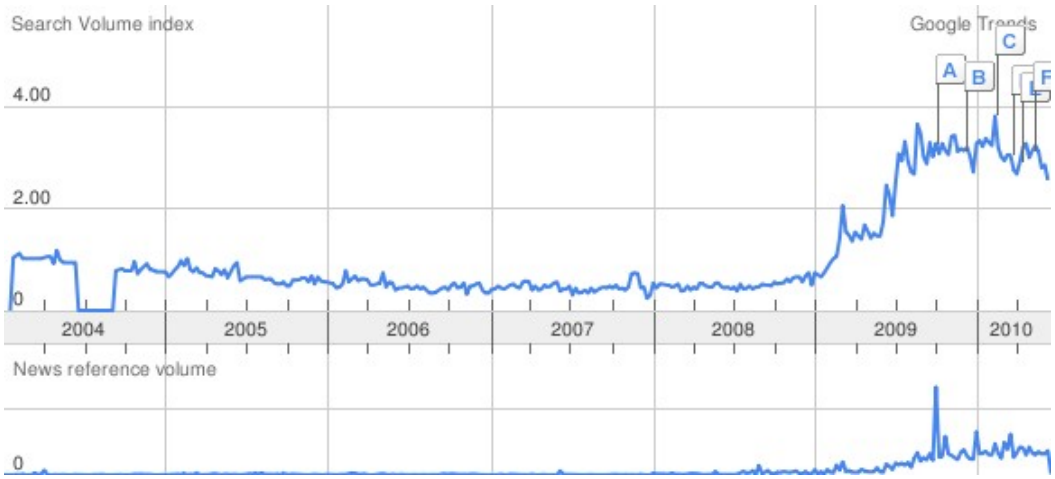


Figura 2.16: Gráfico do Google Trends mostra o crescimento das pesquisas sobre Realidade Aumentada.

Devido ao facto da RA ser uma tecnologia pouco conhecida entre a generalidade das pessoas, leva-as a ficarem em muitos casos curiosas e atraídas ao utilizarem aplicações de RA. Contudo é visível uma grande afluência à tecnologia nos últmos tempos (ver figura 2.16). Estes motivos levam a uma adesão crescente de diferentes áreas aos sistemas de RA. O desenvolvimento de sistemas, muitas vezes específicos para cada uma destas áreas, é uma mais valia na representação/visualização e interacção dos conteúdos a transmitir. Em seguida são mencionadas algumas áreas, onde é possível encontrar muitas das aplicações/tecnologias de RA existentes actualmente.

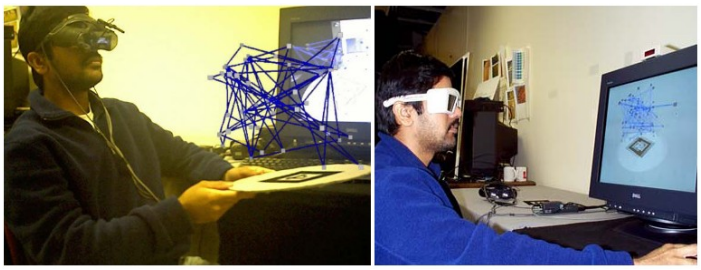


Figura 2.17: Imagens das aplicações criadas por Daniel Belcher *et al.* [1], para visualizar grafos.

A RA na área da educação torna as aplicações interactivas e apelativas ajudando a uma boa compreensão e eficiência no suporte dos vários temas educacionais [25]. Entre vários exemplos, a aplicação desenvolvida para a matemática por Daniel Belcher *et al.* [1], apresenta um método imersivo de RA. Para esta aplicação, os autores construíram várias *interfaces* de interacção, para o utilizador visualizar grafos tridimensionais de forma mais perceptível (ver figura 2.17). Muito semelhante ao exemplo dos grafos, é a aplicação para a geometria da Universidade de Tecnologia de Vienna. Onde Hannes Kaufmann [26] desenvolveu uma aplicação de RA colaborativa entre vários utilizadores, que integra a visualização e construção de sistemas geométricos e matemáticos, utilizados nos programas escolares e universitários.

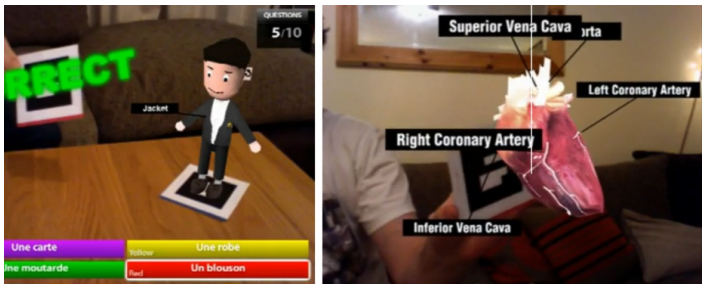


Figura 2.18: Dois exemplos retirados da aplicação *LearnAR*, na áreas de matemática e línguas.

Outro projecto vocacionado para a educação tem o nome *LearnAR* [6]. Este projecto foi criado pela *Specialist Schools and Academies Trust* (uma organização que tem como membros, mais de 5000 escolas secundárias) com o objectivo de proporcionar aos alunos e professores recursos didácticos sobre vários temas. A aplicação criada, tem alojamento na Web e uma simples utilização através duma webcam e marcas. As

[6] www.learnar.org

actividades de aprendizagem dividem-se pela matemática, física, línguas, química e biologia, que podem ser exploradas através de jogos de perguntas. O seus utilizadores usam, muitas das vezes, as marcas para interagir e visualizar os problemas/questionários impostos pela aplicação (ver figura 2.18).

Figura 2.19: Aplicação médica de RA, que adiciona a uma parte do corpo capturado, o modelo virtual do seu interior[7].

Na área da saúde existe também investigação em RA, tal como a criação de aplicações para a formação e simulação de procedimentos médicos. Isto é visível no trabalho desenvolvido por *David Sickinger* [27], onde este, através da junção de diferentes ferramentas de desenvolvimento para RA, criou uma base de dados biomdica para a sobreposição do modelo virtual do crânio do utente na captura feita pela aplicação. Também como suporte à visualização, foi criado um projecto inovador pelo físico *Hebert Zeman* [3]. O projecto de seu nome VienViewer® é capaz de projectar as veias de paciente na sua própria pele (ver figura 2.20). O processo é feito com a medição da temperatura corporal através de raios infravermelhos, com os resultados, são feitos os cálculos para encontrar o posicionamento das veias na captura e a projecção das mesmas sobre a pele analisada.

Figura 2.20: Projecção das veias sobre a pele com o VienViewer® [3].

[7]Imagens retiradas do artigo [2].

Já na simulação é possível encontrar um simulador de treino [28] para equipamentos dispendiosos, onde são feitas montagens pelos utilizadores de agulhas para biópsias, mas recorrendo a um método virtual. Existem muitos outros simuladores para procedimentos médicos [29] [30].

Figura 2.21: Imagens do projecto *Mobile AR Tour Guide* [4].

A ideia de implementação da RA, para suporte às pessoas que visitam lugares históricos e culturais, surgiu já alguns anos [31]. Juntando a tecnologia da RA com a ajuda da georreferenciação (GPS), é feita a criação dum guia virtual com apresentação de trajectórias (ver figura 2.21), como por exemplo o projecto *Mobile AR Tour Guide* criado por Reitmayr e Schmalstieg [4] na Universidade de Tecnologia de Viena, Áustria. O *Mobile AR*, leva o utilizador a uma visita guiada por uma das praças de Viena, com diversos pontos turísticos e caminhos relativamente difíceis nos seus arredores. Outro projecto semelhante é o *LifeClipper2*[8], mas o grande problema destes sistemas é todo o equipamento necessário para a sua utilização. Ou seja, para além de dispendioso é pesado, porque o utilizador necessita de um capacete ou óculos para ver a aplicação, uma mochila para levar o hardware, bateria, entre outras coisas.

O motivo da mobilidade levou o conceito a migrar e expandir-se para os *smartphones*, onde o utilizador usa o telemóvel e a webcam para visualizar conteúdos quer turísticos, como informativos em RA (farmácias, ATMs, empresas, etc. . .). Exemplo disso, são projectos como *Layar*®, *Wikitude*®, *Yell AR*® ou *ARgo*®, que através da tecnologia GPS, 3G e análise comparativa de imagens, fazem apresentação de conteúdos utilizando a RA. Utilizando esta tecnologia, advêm outros tipos de aplicações, como por exemplo as mensagens enviadas no *Twitter*, segundo a direcção em que o telemóvel se encontra (ver figura 2.23).

[8]www.lifeclipper.net

(a) Sistema Yell AR® **(b)** Sistema Layar® **(c)** Sistema AR Go®

Figura 2.22: Alguns exemplos de *Web-Browsers* para RA[9].

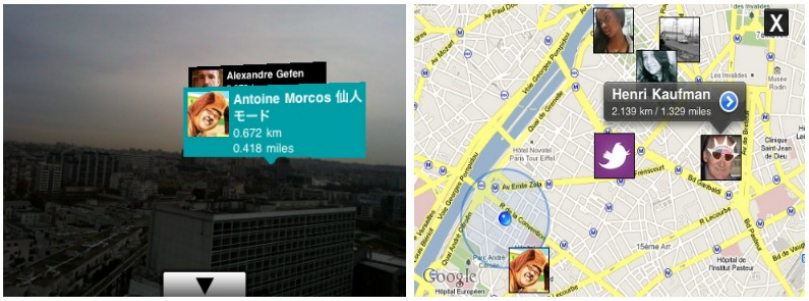


Figura 2.23: Aplicação Twitter360® [10] que mostra os *twittes* em redor do utilizador.

Uma das grandes e crescentes apostas na tecnologia de RA tem sido na promoção de produtos e conceitos na área do marketing e comércio. Isto acontece principalmente na utilização da RA através da Internet, onde pequenas aplicações são alojadas para serem utilizadas em grande escala pelo público-alvo. Assim com recurso a uma tecnologia nova e fascinante para grande parte das pessoas, encontram-se muitas dezenas de aplicações em diferentes meios. Por exemplo na indústria:

- Cinematográfica para a promoção de filmes, como Star Trek® [11], Avatar®, GI Joe Rise of Cobra®, Night at the Museum 2® ou a recente aplicação criada para o filme IronMan 2®[12] que apenas utiliza a cara do utilizador e os movimentos;

[9] Imagens da Apple® Store.
[10] http://www.twitter-360.com
[11] www.experience-the-enterprise.com
[12] www.iamironman2.com

- Automóvel, onde marcas com Citroën® (C3 Picasso), Nissan® (Cube), Mini® (Cooper), entre outras marcas;

- Da confecção de roupa como a Springfield®, com um provador on-line de roupa através de modelos virtuais.

Figura 2.24: Campanha publicitária num revista de automóveis ao carro Mini Cooper®.

Na ajuda ou entretenimento sobre os produtos adquiridos, em que a empresa Lego®, faz-se acompanhar por uma marca no exterior de algumas caixas, onde a aplicação recria e constrói o produto virtualmente (sobre a caixa). Outros projectos são também encontrados em Portugal, onde a banda Blasted Mechanism [13] utiliza a capa de um dos seus álbuns, para fazer a projecção dos elementos da banda em formato digital (a tocar uma música). Existem muitas outras áreas de negócio, que recorrem à RA para novas ideias de marketing.

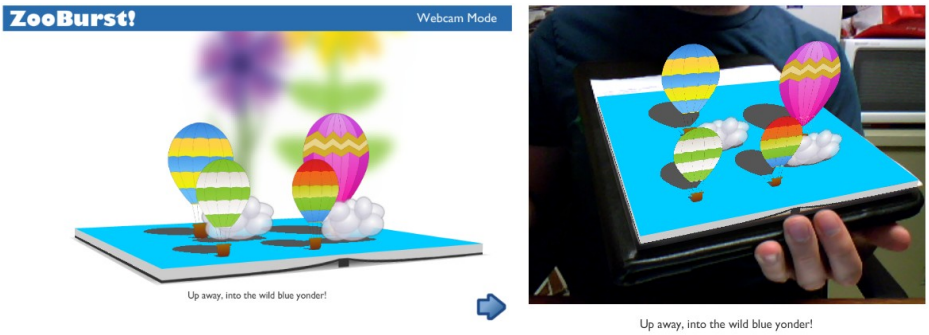


Figura 2.25: Imagens do blog *ZooBurst*, da sua aplicação para criar e ver livros *Pop-Up* através da RA.

[13] http://ar.blastedmechanism.com/pt/inicio.aspx

Ao ser utilizada no entretenimento, a tecnologia RA consegue mais recursos e investimentos, para investigação e desenvolvimento. Uma das investigações levada-as a cabo durante os últimos anos, foi para aplicações como puzzles, livros educacionais para as crianças [32] Eva09 ou de histórias contadas sobre a forma de RA [33]. Para este tipo de livros foi criado um repositório (*ZooBurst*) [14] que permite aos utilizadores criarem a suas próprias historias e colocá-las num livro virtual com paginas *Pop-Up*, visíveis via browser.

Figura 2.26: Jogo Eye of Judgment Playstion®.

A grande indústria dos jogos é uma das áreas do entretenimento que proporciona evolução da RA. Exemplo disso foi a criação do jogo para a consola Playstion3® intitulado *Eye of Judgment*® [15]. O jogo consiste num tabuleiro onde se colocam cartas, cada uma contém uma personagem virtual de combate. Através de uma Webcam direccionada para o tabuleiro e ligada à consola, é possível ver as personagens virtuais sobre as cartas (ver figura 2.26). O modo de jogo é por turnos e pode ser jogado por dois oponentes, que utilizam as suas personagens para combater. Esta tecnologia promoveu a interacção entre marcas (quando aproximadas umas das outras), uma melhor qualidade gráfica dos modelos e suas animações. Mais tarde a consola portátil PSP lançou o jogo *Invizimals*, que utiliza o mesmo conceito que o jogo *Eye of Judgment*. Outro avanço feito pela Playstion3® foi o *Eye Pet*®, onde os jogadores para além da marca podem interagir com o modelo virtual através de movimentos.

[14] http://alpha.zooburst.com/index.php
[15] www.eyeofjudgment.com

Mas uma das grandes novidades na RA é o projecto *Kinect*™ [16] da Xbox360® a ser lançado em 2011. Este tecnologia vai ter virtualidade aumentada e RA, através dos movimentos capturados por uma câmara e sensores que detectam movimentos e profundidade dos objectos.

Tal como a Sony®, empresas como a Total Immersion [17] ou a portuguesa YDreams [18], apostam e desenvolvem tecnologias e aplicações para RA. É de salientar um projecto da YDreams relacionado com o objectivo deste trabalho. Esse projecto é uma *Natural User Interface*, que foi desenvolvida com tecnologia de captura em profundidade da empresa Canesta e o objectivo é interagir com objectos virtuais, apenas com o corpo (semelhante ao projecto *Kinect* anteriormente referido).

Para finalizar o trabalho relacionado sobre a RA, é de mencionar a tecnologia criada por Taehee Lee e Tobias Höllerer. Os autores criaram uma aplicação [34], que posiciona e orienta objectos virtuais com base na mão do utilizador. Para os objectos virtuais serem colocados sobre a mão, é feita a segmentação da mão, depois é efectuado o cálculo e classificação das pontas dos dedos, para saber a orientação e posição da mão.

Existem ainda outras aplicações e tecnologias de RA em diversas áreas, tais como na arquitectura [35], química [36], cartões de visita[19], etc. . .

2.4 Ferramentas de desenvolvimento

Os ambientes de desenvolvimento de aplicações de RA, basicamente vão receber a captura de um espaço real, fazer uma análise a essa captura (dependendo do software) e a ela sobrepor objectos virtuais com um efeito "presencial" (aumentado). Por fim transmitem o resultado num *display* de saída. Uma das características/funções mais importantes destes ambientes, deve ser a interactividade do utilizador com o sistema.

Existem algumas escolhas para os ambientes que permitem desenvolver aplicações de RA. Começando pela ferramenta ARToolKit [20] desenvolvida pelo Dr.

[16] http://www.xbox.com/kinect
[17] http://www.t-immersion.com/
[18] http://www.ydreams.com
[19] http://www.burtonposey.com/thecard
[20] http://www.hitl.washington.edu/artoolkit/

Hirokazu Kato. Esta foi uma das ferramentas pioneiras para a criação de aplicações de RA com marcas. É um conjunto de bibliotecas desenvolvidas em C e C++ e dá liberdade aos utilizadores para um melhor desenvolvimento quer das aplicações, quer do próprio sistema. Por fim, esta ferramenta contém todas as funcionalidades para o *tracking* das marcas, tal como a criação e posicionamento do ambiente gráfico. Na actualidade encontra-se na Universidade de Washington onde continua a pesquisa e desenvolvimento do mesmo. O ARToolKit é gratuito para aplicações não comerciais e distribuído como open-source sobre a Licença GPL [21].

Devido ao grande avanço feito pelo ARToolKit, a sua estabilidade e boa estruturação, muitas outras ferramentas e bibliotecas foram produzidas com base no ARToolKit. Entre elas as mais conhecidas são a ARToolKitPlus [37] criada na Universidade Tecnológica de *Graz*, onde através de várias optimizações à biblioteca base (ARToolKit), foi criado um software mais leve com vista à integração em dispositivos móveis. Para além das optimizações também apresenta inovações, tais como uma técnica de adaptação do *threshold*, onde o sistema tenta descobrir os melhores valores para cada frame capturado, de acordo com a iluminação. Esta inovação e a detecção de marcas, foram baseadas na ferramenta ARTag.

A OSGART foi outra biblioteca criada a partir do ARToolKit, no laboratório HIT-LabNZ [22] da Universidade de Canterbury. Esta utiliza as funções já existentes (*raster* e detecção de marcas) e acrescenta funções de construção para ambientes virtuais em OpenSceneGraph. Ou seja, este tem a capacidade de executar as aplicações de RA, recorrendo a um render de melhor qualidade para os objectos virtuais. Contudo, existem outras mais valias nesta biblioteca, tais como a importação e exportação de formatos do programas Maya e 3D Studio MAX. Tem também a possibilidade de utilização de sombras, oclusão, videos e multi-marcas. Como já foi dito, a OSGART usa o ARToolKit, mas também pode juntar outras ferramentas como ARToolKitPlus ou a ARTag, entre outras.

O mencionado ARTag [12] é mais um sistema baseado na ARToolKit, onde a sua criação teve como objectivo principal a resolução dos problemas inerentes das marcas na RA. Tais como: os falsos positivos/negativos e as falsas associações. Isto é, quando na captura é detectada a forma duma marca, que na realidade não é uma marca. O contrário também pode acontecer, quando uma marca não é detectada,

[21] http://www.gnu.org/licenses/gpl.html
[22] http://www.hitlabnz.org

apesar de se encontrar na captura. Quanto às falsas associações, deve-se ao facto do sistema confundir a marca detectada, com outra marca que se encontra no sistema(exemplo Multi-marcas). Este sistema tem uma boa precisão e detecção das marcas, mesmo que estas se encontrem parcialmente ocultadas e também com a iluminação menos indicada para a passagem do *raster*.

Até ao momento foram apenas mencionadas bibliotecas e sistemas baseados no ARToolKit, onde as suas linguagens de programação são o C++ e o OpenGL. Isso impossibilita a criação de uma aplicação capaz de ser executada na Internet, mais precisamente via Browser. Como um dos objectivos deste trabalho foi elaborar uma aplicação acessível de forma genérica a todos os utilizadores, existiu a necessidade de procurar sistemas, em que o resultado da compilação obtivesse suporte no Browser. Foi então estudada a ferramenta FLARToolKit (Flash Augmented Reality Toolkit), que através do plugin Flash Player da Adobe® permite a visualização no Browser da aplicação desenvolvida nesta ferramenta. O FLARToolKit é uma importação da biblioteca NyARToolKit, onde esta, por sua vez, é mais uma das ferramentas baseadas no ARToolKit.

O NyARToolKit foi uma adaptação da RA para a linguagem Java, feita por Nyatla [23]. As bibliotecas do NyARToolKit contêm todas as funcionalidades que o ARToolKit tem, desde o render das cenas virtuais, funções para captura e análise da mesma, até ao output da RA. Mesmo sendo a sua criação feita em Java, também pode ser usada pelo C♯ e pelo o sistema Android.

Assim com base na estrutura do NyARToolKit, Tomohiko Koyama criou o FLARToolKit no Spark Project [24]. Ao contrario do NyARToolKit, o FLARToolKit não exportou todas as funcionalidades, onde apenas fez a implementação das funções para o cálculo e pesquisa das marcas. Estas funções utilizam um algoritmo idêntico ao algoritmo descrito na secção 2.2.4. Depois de feitos os cálculos para encontrar a matriz(es) transformada(s) da(s) marca(s), é necessário aplicar esses valores ao ambiente virtual. Como a ferramenta FLARToolKit não contém funcionalidades para uma representação gráfica, tem de ser utilizado um motor gráfico externo (Papervision3D, Away3D, Sandy, Alternativa3D), para fazer o render do ambiente virtual.

[23] http://nyatla.jp/nyartoolkit/wiki/index.php?FrontPage.en
[24] http://www.libspark.org/wiki/saqoosha/FLARToolKit/en

Capítulo 3

Trabalho Desenvolvido

3.1 Interacção com marcas

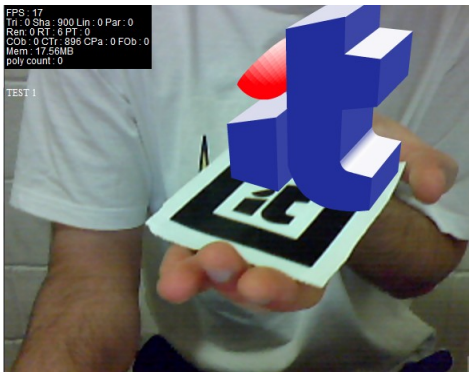


Figura 3.1: Aplicação de RA desenvolvida com marcas.

Com base nas tecnologias anteriormente descritas foi desenvolvida uma aplicação de RA. Esta aplicação recorre à utilização de marcas para posicionar objectos virtuais no ambiente capturado. Esta aplicação foi criada utilizando o Papervision3D como motor gráfico e o FlarToolkit para processamento das marcas. Também, foi feita uma separação das funcionalidades, de modo a que o projecto possa servir de base a outras aplicações de RA.

A separação foi feita em três partes (ver figura 3.2), onde a primeira cria o cenário de projecção executável em Flash, assim como a inicialização da câmara de captura do ambiente real, a *Pattern* (template) da marca e os seus métodos de detecção. A segunda parte é a associação do motor gráfico e a criação de todos os seus elementos (câmara virtual, viewport, cena, entre outros) e ainda os métodos para os cálculos

da matriz transformada e manipulação dos objectos virtuais. A terceira e última parte serve para iniciar as partes anteriores e inicializar os objectos virtuais.

É então possível modificar este projecto base e colocar novos objectos, marcas e mesmo novas funcionalidades na aplicação. Este facto pode ser comprovado no capítulo dos Resultados, onde são apresentadas outras aplicações desenvolvidas com base neste projecto.

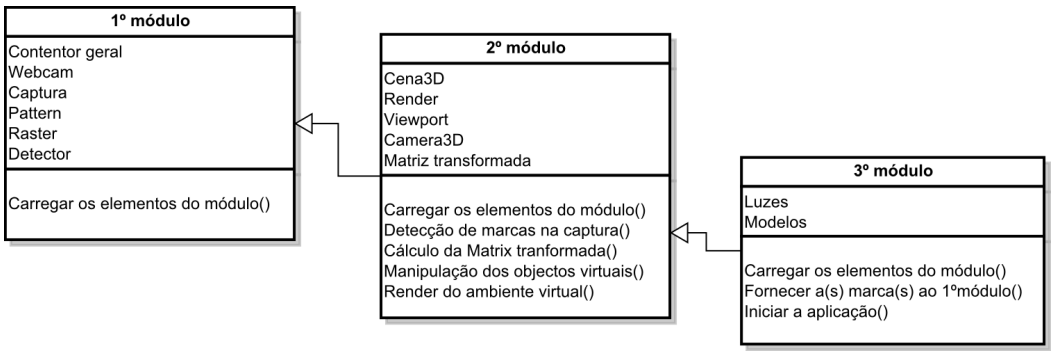


Figura 3.2: Diagrama resumido do funcionamento do sistema de RA com marcas.

3.2 Interacção sem marcas

O desenvolvimento de uma aplicação de RA sem o uso de marcas, teve como objectivo simplificar e facilitar a interacção do utilizador. Esta nova aplicação tem de transmitir ao utilizador o mesmo efeito de uma Realidade Aumentada, que as aplicações com marcas. Para isso têm de ser alinhados/posicionados de acordo com as características do ambiente capturado tendo em conta possíveis mudanças que lá aconteçam. Assim foi necessário detectar e utilizar outros pontos de referência existentes no ambiente capturado, como forma de colocar os objectos virtuais.

A segmentação e a procura de pontos de referência na captura têm que se basear em informação/dados específicos. Isto leva a que haja uma distinção entre os pontos de referência e o resto do ambiente. Por esse motivo, surgiu a ideia de usar o utilizador como ponto de referência para a substituir as marcas, visto que o utilizador ao executar a aplicação vai ser um elemento presente na captura. Essa presença deixa padrões/características específicos nas imagens capturadas, que podem ser utilizadas como pontos de referência. No entanto, considera-se a cara e os movimentos do utilizador, na segmentação e procura.

O funcionamento desta técnica fica então dependente da presença de um utilizador. Este passa a ser o responsável pelas alterações do ambiente virtual. Tendo como base as aplicações de RA com marcas, tentou-se fazer a criação de um mesmo ambiente "aumentado" sobre captura. Assim sendo, tal como as marcas são utilizadas para o posicionamento dos objectos virtuais, foi utilizada a cara do utilizador para fazer esse posicionamento. No entanto, desta forma não é possível orientar os objectos virtuais tal como o é com as marcas. Isto acontece porque a orientação dos objectos tem como base a própria orientação da marca. Neste caso foi necessário criar um método que faz a rotação dos objectos com base em movimentos do utilizador. Resumindo, o sistema emula um ambiente de RA usando a posição da cara e os movimentos do utilizador.

Contudo, ao abandonar a interacção com marcas, todas as funcionalidades suportadas pelos sistemas como o ARToolkit e FlarToolkit deixam de ter utilidade, pois estas estão dependentes da informação fornecida pelas marcas. Mas a aplicação desenvolvida utiliza os mesmos conceitos na criação do ambiente virtual. A diferença está na forma como as coordenadas e ângulos são calculados e transmitidos ao motor gráfico para colocar os objectos virtuais. Esta forma é explicada nas seguintes secções.

3.2.1 Posicionamento

O ambiente virtual é centrado na imagem capturada, ou seja, coloca-se a origem das coordenadas virtuais $(0,0,0)$ no centro da imagem capturada $(largura/2, altura/2)$. É definida uma distância para a câmara virtual em relação à origem $(0,0,0)$, que é também o alvo de captura da câmara. Desta forma os objectos virtuais são visíveis, pois inicialmente são colocados nas coordenadas de origem.

Como já foi referido o posicionamento dos objectos virtuais é feito com base na cara do utilizador. Ou seja, quando o utilizador move a cara, o(s)objecto(s) virtual(ais) acompanham esse movimento. Existe então uma procura na matriz de píxeis da imagem capturada (i.e. a mesma técnica das marcas) para detectar a cara do utilizador. O método para a detecção da cara é explicado em detalhe mais adiante. Se a cara for detectada, os seus valores na matriz de píxeis são guardados. Mas esses valores encontram-se num plano de coordenadas bidimensionais, pois são retirados da imagem capturada. Visto que para fazer o (re)posicionamento dos objectos

virtuais, as coordenadas fornecidas ao motor gráfico têm de ser tridimensionais, é necessário um método de conversão de 2D para 3D.

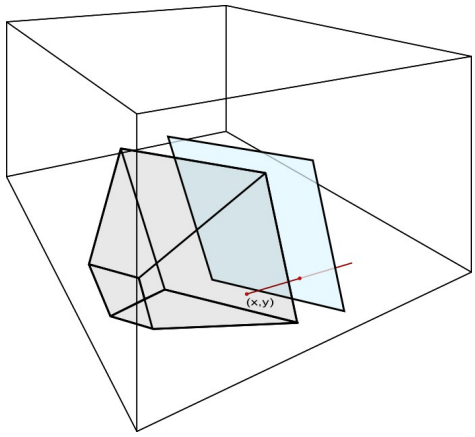


Figura 3.3: Cálculo das coordenadas correspondentes no ambiente virtual.

O método de conversão cria para cada coordenada (x, y) da imagem capturada, uma coordenada (x, y, z) correspondente no ambiente virtual. De certa forma é fazer o "mapeamento" da captura para a cena 3D. Mas apenas é utilizado para a conversão o ponto médio do rectângulo resultante da segmentação da cara detectada. Esta conversão (ver figura 3.3) é feita através dos passos seguintes:

1. As coordenadas (x, y) do ponto médio do rectângulo da cara são projectadas no plano da câmara virtual (P1);

2. É criado um raio com origem nas coordenadas resultantes da projecção anterior. Este raio é perpendicular ao plano P1;

3. Em seguida o raio vai intersectar um plano P2, que foi criado paralelamente ao plano P1 e passa na origem.

Assim as coordenadas resultantes da intersecção entre o raio e o plano P2, são as coordenadas tridimensionais correspondentes ao ponto médio da cara na captura.

Para calcular a profundidade dos objectos é obrigatório também modificar a coordenada z dos mesmos, pois apenas se converteram as coordenadas x, y da captura. Este posicionamento em profundidade é calculado em relação ao tamanho da cara do utilizador, onde a escala de conversão é feita entre o tamanho (máximo e mínimo) da cara e a distância da câmara virtual à origem do ambiente virtual.

Como os objectos acompanham o movimento da cara do utilizador, então, quando ele se aproxima ou afasta da fonte de captura, os objectos também o fazem.

O posicionamento dos objectos virtuais é assim feito de forma interactiva pelo o utilizador, através da detecção da cara. Dos algoritmos analisados [38] [39] [40] [41] para a detecção da cara foi escolhido aquele que apresentava melhores condições de implementação, i.e. rapidez e simplicidade. Assim, a escolha recaiu no algoritmo criado por Viola and Jones [38], devido à rápida detecção de objectos, o qual será explicado no decorrer desta secção.

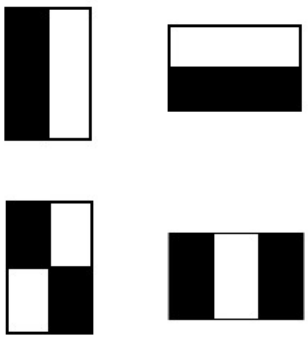


Figura 3.4: Representação simples das *Haar-like Features*.

O algoritmo tem como base a utilização de *Features*. Papageorgiou *et al.* [42] apresentou as *Features* como sendo representações binárias de uma(s) área(s) com características geométricas (figura 3.4). Estas *Features* podem ser utilizadas em diferentes partes de uma imagem, para comparação e consequentemente classificação dessas mesmas partes. As *Features* apresentadas por Papageorgiou *et al.* são compostas por dois rectângulos adjacentes. Devido à utilização apenas de simples rectângulos é possível fazer uma análise escalável (vários tamanhos). A análise vai procurar partes/áreas da imagem com características semelhantes às das *Features*, recorrendo à proximidade do contraste e tonalidade. Os autores inicialmente utilizaram o mesmo número de *Features* que o método original. Posteriormente, adicionaram mais rectângulos adjacentes ao utilizar as *Features* no seu algoritmo, para conseguir uma detecção mais abrangente das características da cara.

O algoritmo foi criado em três etapas, onde começa por fazer a divisão analítica da imagem em pequenas áreas (sub-imagens). Nestas áreas intituladas de *Intergal Images*, é aplicado o cálculo das *summed area tables* (ex: figura 3.5) criado por Crow [43]. Baseando-se apenas em quatro pontos da *Intergal Images*, este cálculo

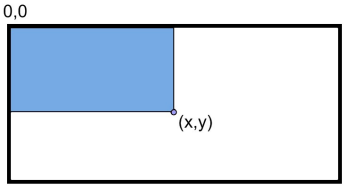


Figura 3.5: Summed Area Tables.

permite a soma dos elementos contidos num rectângulo. A fórmula para o cálculo da *Intergal Image* é:

$$I(x,y) = \sum_{x' \leq x, y' \leq y} p(x', y') \tag{3.1}$$

O seu resultado é um array I utilizado na fase seguinte do algoritmo. Os valores I correspondem às intensidades de todos os pixéis $p(x', y')$ à esquerda e acima do pixel (x, y) da *Integral Image* em análise. Com este cálculo é possível fazer um método de treino para os dados da imagem sobre as *Features*, através da diferença entre a soma dos pixeis da região branca, com os da região preta.

Como foi referido, existem versões mais recentes do algoritmo, que implementam ainda mais *Features*. E apesar do cálculo das *Integral Image* utilizar o método criado por Lienhart e Maydt [44], que permite uma análise de 45º às *Integral Images*, para aumentar a rapidez do cálculo, o método de Lienhart e Maydt foi retirado. Contudo ao diminuir os cálculos feitos nas *Integral Images*, a detecção da cara não vai ser tão eficaz.

A segunda parte do algoritmo consiste em optimizar/filtrar as comparações entre as *Features* e os cálculos feitos anteriormente para as *Integral Images*. Este processo é feito para o conjunto de treino criado para n amostras de dimensão 24x24, que podem ou não ser caras e que vão ultrapassar 180.000 *Features*. Para resolver o problema desta quantidade gerada por cada imagem, a solução passa por filtrar aquelas que têm mais possibilidade de corresponderem à figura geométrica. Só então é feita a análise concreta das imagens seleccionadas. Por isso, o *AdaBoost* de Viola and Jones é uma ferramenta com boa performance na classificação das *Integral Imagens*. A função do *Adaboost* AB_j utilizada para a classificação é:

$$AB_j(x,y) = \begin{cases} 1 & if \quad F_j(x) < \sigma_j \theta_j \\ 0 & \quad caso\ contrario \end{cases} \tag{3.2}$$

Onde o F_j representa as *Features* e o θ_j o *threshold* definido que permite ou não a classificação das *Features*. Para definir a orientação do sinal de comparação na condição if é utilizado o σ_j. Segundo os autores, durante a passagem de j vezes pelo classificador *Adaboost*, o erro de classificação é minimizado e eliminadas cerca de 50% das *Integral Images*.

A última parte do algoritmo vai optimizar a utilização do *AdaBoost*, onde é utilizado de forma individual e sequencial. Ou seja, é transformado numa máquina de estados [45] com o nome de *Cascade of Classifiers*. Ela vai estabelecer em cada estado, qual a complexidade da comparação através do *threshold* visto anteriormente. O *threshold* estabelece a passagem reduzida de *Integral Images* (com valores positivos), que podem corresponder às *Features*. As que passarem ao estado seguinte vão sofrer a mesma análise, mas com valores de cálculo mais refinados e assim sucessivamente. Então é possível fazer uma análise das *Integral Images* menos pesada, mas também menos precisa e vice-versa, alterando a complexidade dos estados. Resumindo, esta implementação em árvore permite rejeitar nos estados iniciais, grande parte das *Integral Images* sem interesse para comparações mais complexas nos estados finais. É também na máquina de estados, que é feita a detecção de vários tamanhos e posições da cara, através da distribuição pelos diferentes estados de *Features* escaláveis (ver figura 3.6).

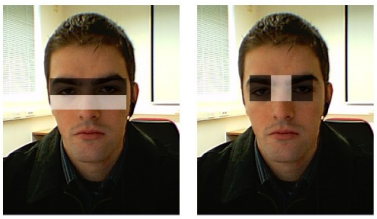


Figura 3.6: Possível representação de *Features* numa imagem.

Para mais detalhes sobre o algoritmo consultar [38]. Tudo o que diz respeito aos procedimentos para a detecção da cara do utilizador foram implementados tendo como base e orientação, a importação das funcionalidades já desenvolvidas pela biblioteca OpenCV[1].

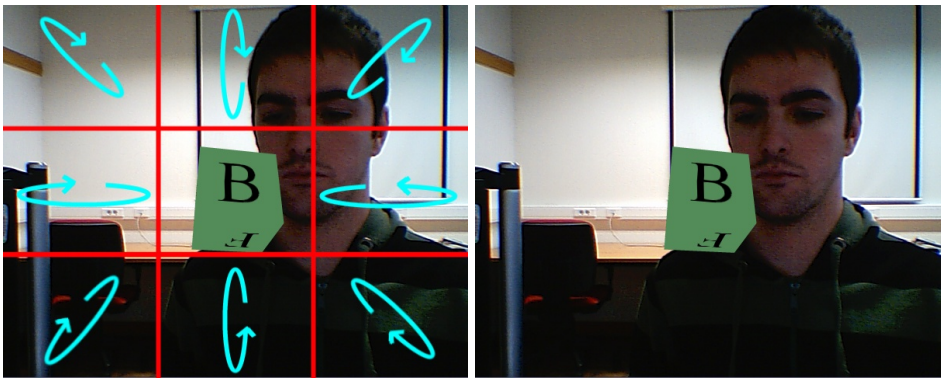


Figura 3.7: Representação das áreas para a rotação, associadas na detecção de movimentos.

3.2.2 Orientação

A forma encontrada para manipular os objectos, i.e. modificar a orientação dos objectos, passa pela rotação com base em movimentos efectuados pelo utilizador na área de captura. Para isso, a área de captura foi dividida em oito sub-áreas iguais (ver figura 3.7). Estas áreas funcionam como zonas de interacção com o ambiente virtual e permitem manipular o(s) objecto(s) virtual(ais). A activação de uma sub-área é feita através dos movimentos do utilizador e provoca uma rotação no(s) objecto(s) virtual(ais). Assim definiu-se um valor comum para todos os ângulos de rotação, onde cada uma das sub-áreas contém um tipo diferente de rotação (ângulo positivo ou negativo). A rotação é feita segundo o eixo X ou/e Y, manipulando assim a orientação do(s) objecto(s) no ambiente virtual. Para executar uma rotação no eixo X, o utilizador tem de fazer movimentos nas sub-áreas superiores (com um ângulo positivo) ou inferiores (para um ângulo negativo). Para eixo Y, o utilizador tem de fazer movimentos nas sub-áreas à direita para uma rotação com um ângulo positivo e à esquerda para um ângulo negativo. Nos quatro cantos da imagem capturada, encontram-se definidas as áreas que geram a rotação em ambos os eixos X e Y. A rotação é feita em simultâneo, mas com os mesmos critérios anteriormente referidos para cada um dos eixos. A figura 3.7 ilustra os tipos de rotações possíveis.

Um dos problemas de ambientes de RA é a falta de percepção do que está acontecer na aplicação. O utilizador ao movimentar a cara nota que o objecto virtual de certa forma segue esse mesmo movimento. Mas existe o problema do utilizador não compreender a razão do objecto virtual rodar e como isso é feito.

[1]http://sourceforge.net/projects/opencvlibrary/

Figura 3.8: Setas geradas para *feedback* da rotação.

Para tornar esta interacção mais intuitiva foi criado um *feedback* associado a cada ordem de rotação. Este efeito visual é criado com o lançamento de partículas no local do movimento. As partículas são setas orientadas de acordo com a rotação correspondente (ver figura 3.8). Ou seja, se o utilizador executar uma movimentação na zona inferior da captura, as partículas vão ser criadas e direccionadas para baixo, o mesmo acontece de forma análoga para as outras áreas. Assim, este *feedback* ajuda o utilizador a compreender os movimentos que executa e as acções que lhe estão associadas.

A orientação dos objectos virtuais na aplicação é feita através da rotação dos mesmos, com base nos movimentos presentes na captura. A detecção de movimentos tem como objectivo ser rápida e de baixo custo computacional, pois vai ser utilizada em conjunto com todos os métodos já referidos anteriormente. Assim, depois de analisadas algumas escolhas [46, 47, 48, 49] para a detecção de movimentos, o algoritmo escolhido foi a subtracção de frames. A implementação do algoritmo depende de dois factores: tempo e posição. Este verifica se houve variação ao longo do tempo, para uma determinada posição na imagem capturada. Isso é feito através da diferença entre os pixeis, em duas imagens consecutivas capturadas. Ao transformar este método em formula, temos a seguinte equação:

$$R = |L_2(x_2, y_2, t_2) - L_1(x_1, y_1, t_1)| \tag{3.3}$$

Para a equação 3.3, R é o módulo da subtracção entre a luminosidade $L2$ e $L1$ de cada frame capturada para uma dada coordenada (x, y). Esta subtracção faz-se segundo o instante de tempo t_1 e t_2. O cálculo é aplicado a todos os pixeis da imagem capturada e da imagem anterior. Depois, os resultados próximos do valor zero são aqueles que apresentam pouca ou nenhuma luminosidade, equivalente

à não existência de movimentos nessas zonas. Já os valores distantes de zero apresentam uma diferença entre pixeis, o que equivale à existência de movimentos. Na implementação é feito então o cálculo das diferenças entre os pixéis das imagens. É também definido um *threshold* para o valor das diferenças. Os valores superiores a esse *threshold* são transformados em pixeis de cor branca, caso contrário cor preta. Por fim, são criadas áreas rectangulares, para os aglomerados de píxeis resultantes de cor branca (i.e. a área dos movimentos).

Também, para um melhor funcionamento da aplicação, no que diz respeito à detecção de movimentos é aplicado um filtro aos movimentos. Esta filtragem permite ignorar grandes movimentos efectuados na captura. Então os movimentos relevantes para activação da rotação, têm de ter tamanhos inferiores a 80% duma sub-área pré-definida (ver figura 3.7). Assim grandes movimentações presentes na captura são ignoradas, devido a grande parte delas não serem ordens para rotação. O que leva ao processamento apenas de instruções de rotação (por exemplo) de um simples movimento dos dedos nas sub-áreas pré-definidas.

3.2.3 Implementação das transformações

Para a implementação das transformações no ambiente virtual, é necessário utilizar os valores do posicionamento e orientação calculados anteriormente. Estes valores têm de ser transmitido ao motor gráfico (ex: Papervison3D) através das matrizes transformadas. Como foi mencionado na secção 2.1.1, cada elemento do ambiente virtual contém uma matriz transformada. O cálculo desta matriz é utilizado nos sistemas baseados em marcas. Não sendo utilizadas essas funcionalidades, é criado um método que recebe os valores, processa-os e devolve a matriz resultante. Então os passos seguidos pelo método são:

1. Recebe os novos ângulos de rotação e as novas posições;

2. Posiciona os objectos virtuais na origem (0,0,0) da cena, pois o motor gráfico utilizado não permite a rotação global e esta é feita em relação à origem;

3. Cria uma matriz transformada para cada ângulo recebido;

4. Multiplica as matrizes criadas;

5. A matriz resultante da operação anterior, é multiplicada pela matriz de cada objecto;

6. À matriz resultante da operação anterior, faz-se a alteração para as novas posições;

7. A matriz resultante é atribuída ao objecto correspondente.

Para o posicionamento, apenas são enviados para o método os ângulos com valores nulos. É de salientar que o método também tenta emular o deslocamento feito pelo motor gráfico, para os vários valores recebidos. Por exemplo, quando uma marca roda, os novos cálculos do objecto virtual são feitos para alguns valores intermédios da rotação. Ou seja, existe um acompanhamento e suavização da rotação do(s) objecto(s). Pois, este não se limita a receber e executar apenas o resultado final da rotação.

Na aplicação desenvolvida, a rotação é feita através das áreas de interacção, onde cada uma delas fornece um ou dois ângulo(s) específico(s) a este método criado. Para que a rotação seja suave, cada novo ângulo de rotação é dividido em vários valores intermédios e distribuídos sequencialmente por várias frames. Isto por exemplo, vai efectuar não uma rotação com ângulo de $\sigma = 0.5$ de forma instantânea, mas sim várias rotações por frame com $\sigma = 0.1$ até atingir o valor 0.5. O mesmo processo acontece para o posicionamento. Assim é criado um efeito de deslocamento continuo, semelhante ao conseguido com as marcas.

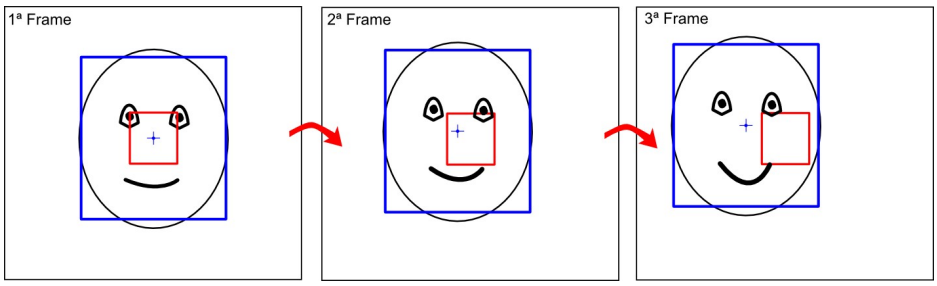


Figura 3.9: Threshold (caixa pequena vermelha) que ao ser ultrapassado valida a nova detecção da cara (caixa grande azul).

O método criado executa modificações tanto no posicionamento como na orientação. Mas quando são detectados na mesma sequência de frames, existe um conflito entre eles. Isso deve-se ao facto da detecção de movimentos ser accionada

com a movimentação da cara do utilizador e a orientação com base na detecção de movimento. Logo, neste caso, o comando dado pelo utilizador que prevalece é o posicionamento e não a orientação. Mediante esta disputa das transformações, foi definida uma regra de prioridade que tenta estabelecer uma hierarquia funcional. A regra obedece à seguinte ordem:

1. Os dados da captura são encaminhados para a detecção da cara;
2. É feita a análise para a detecção de uma nova cara.
3. Se a análise for positiva, não é feita a passagem dos dados da captura para a detecção de movimentos;
4. Se a análise for negativa, é feita a passagem dos dados para a detecção dos movimentos.

Mesmo definindo a detecção da cara como prioritária, coloca-se ainda o problema da cara do utilizador estar "sempre" presente em todos frames capturados. Como consequência disso é feita sempre a detecção consecutiva da cara, nunca permitindo (segundo a regra imposta) a análise da detecção de movimentos. Para resolver esta problemática estabeleceu-se um *threshold*, aplicado aos novos valores para a detecção da cara.

Na detecção da cara é estabelecida uma forma rectangular com base no tamanho da cara. Esta forma permite criar uma nova forma com um *threshold* (ver figura 3.9). A detecção é validada se o novo posicionamento da cara ultrapassar o *threshold*. Caso não seja efectuada nenhuma validação durante um número de frames pré-determinado, o *threshold* é desactivado até nova detecção. Para o posicionamento no eixo do z (profundidade), o *threshold* é estabelecido segundo o tamanho da cara do utilizador.

Esta implementação torna possível a detecção de movimentos nas frames da captura, no intervalo entre a detecção da cara e a nova validação. A detecção da cara do utilizador, quando aplicada a todas a frames capturadas (25 por segundo), torna-se um método com um custo computacional elevado (para a tecnologia utilizada). Logo, o método de detecção é feito apenas em intervalos de 0.5 segundos, onde em média são analisados duas frames por segundo. Esta funcionalidade deixa mais espaço para a detecção de movimentos, como também uma maior rapidez por parte da aplicação.

Assim, ao fazer-se a ligação entre todas as técnicas e os métodos anteriores, é possível estabelecer um diagrama (ver figura 3.10) que mostre de forma resumida o funcionamento global do sistema de RA sem marcas proposto nesta secção.

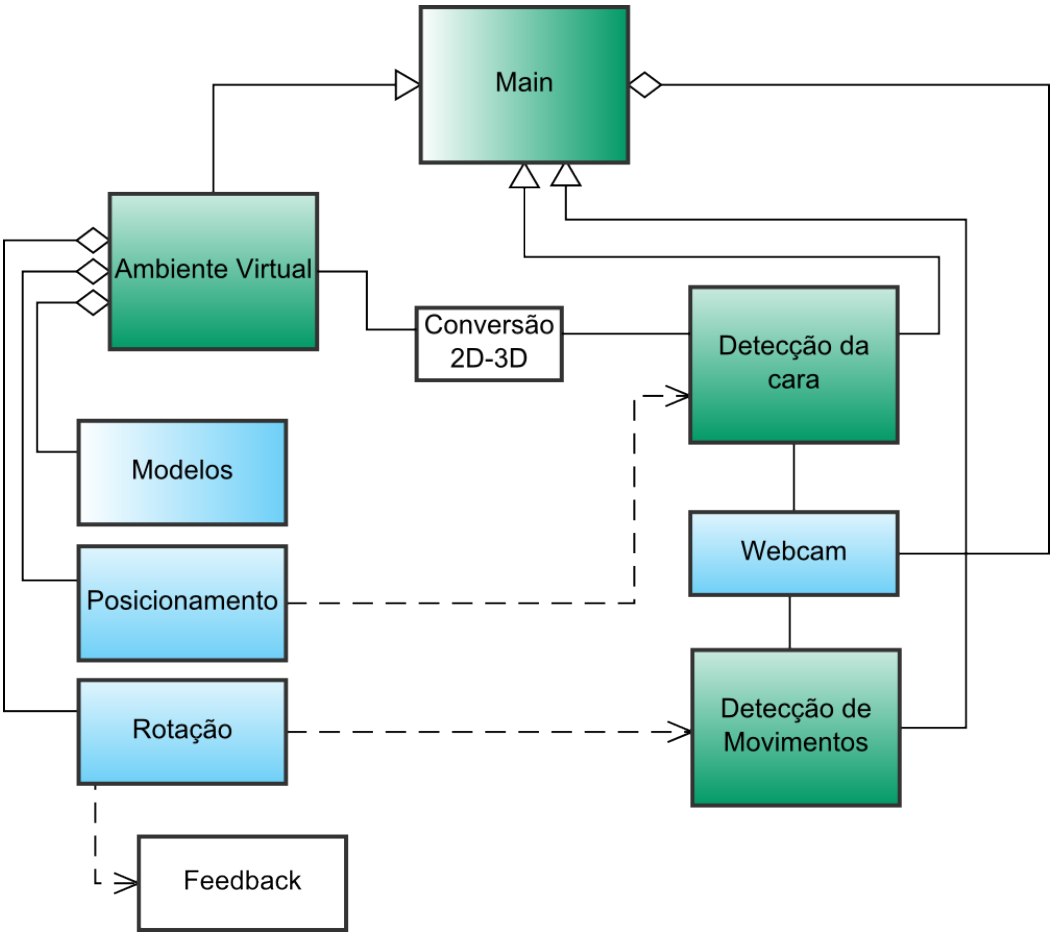


Figura 3.10: Diagrama resumido do funcionamento do sistema de RA sem marcas.

3.3 Modelos com diferente níveis de detalhe

Muitas das aplicações de RA desenvolvidas em Flash e ambiente Web, apresentam uma baixa taxa de frames por segundo. Isso deve-se às limitações que estas tecnologias têm no processamento de toda a informação de Input em tempo útil. Ou seja, os frames deveriam ser analisados em tempo útil, para uma representação em tempo real. Este problema de *performance* é agravado quando há a necessidade em fazer a pesquisa duma marca e também a apresentação de cenas virtuais de grande complexidade. Isto por exemplo é visível quando um objecto contém um

grande número de triângulos na sua malha, pois este mesmo objecto torna o motor gráfico incapaz de fazer o render em tempo útil.

Para tentar melhorar esta problemática foi implementada a técnica de LOD's [50]. Onde foi criada uma aplicação para optimizar a visualização de objectos com um grande número de triângulos, usando modelos com diferentes níveis de detalhe.

Este tipo de implementação pode ser uma mais valia, em casos pontuais com modelos de grande complexidade, por exemplo, em aplicações como a apresentada na figura 2.24 para publicitar automóveis. Onde os modelos criados do interior do carro (acentos, volante, rádio, etc...) podem ser mais complexos quando nos aproximamos. Depois, quando estes modelos se encontram mais distantes, podem ser usadas representações com menos detalhe, e assim pode ser melhorada a *performance* da aplicação. Um exemplo que também utiliza os conceitos dos LOD's é a aplicação criada por Stephen DiVerdi *et al.* [51], em que os modelos virtuais perdem ou ganham funcionalidades conforme a distância.

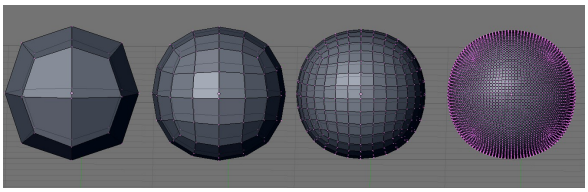


Figura 3.11: Modelo representativo dum esfera com vários níveis de detalhe.

Como o próprio nome indica, são utilizados vários níveis de detalhe para um objecto (várias representações com diferentes detalhes para o mesmo modelo). A metodologia vai tirar partido da percepção do utilizador, pois basicamente os objectos serão substituídos entre eles (ver figura 3.11), de acordo com o cálculo da distância do objecto virtual à câmara virtual. Sendo evidente que as representações com menos detalhe são utilizadas em posições mais distantes e as representações com mais detalhe, nas posições mais próximas da câmara. O utilizador (na maioria dos casos) não consegue distinguir a diferença e a troca entre as representações com menor e maior detalhe. Os modelos com diferentes níveis de detalhe são implementados usando um array de representações geométricas para cada modelo, onde o índice do array é actualizado conforme o cálculo das distâncias segundo o eixo Z. Uma desvantagem na utilização dos LOD's é ao carregar a aplicação, onde esta se torna mais pesada computacionalmente, pois requer mais memória para todas as representações, embora este não seja um aspecto crítico.

Capítulo 4

RESULTADOS

As aplicações desenvolvidas encontram-se no formato SWF, para poderem ser visualizadas pelo Flash Player. Para efeitos de testes foi criada uma página Web [1], onde estão alojadas as aplicações desenvolvidas. O objectivo é verificar o funcionamento e possíveis erros das aplicações, quando executadas na Internet. Todas as aplicações foram testadas usando uma webcam Creative® VF-0080, utilizando uma resolução de 640 por 480 píxeis.

4.1 Interacção com Marcas

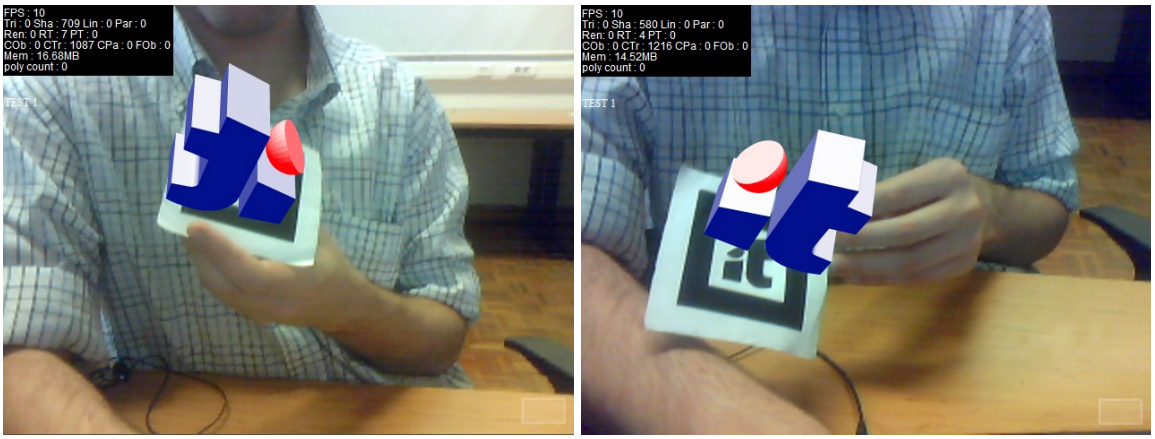


Figura 4.1: Aplicação de RA utilizando marcas, com um modelo virtual.

exportado em formato COLLADA3. Este é um formato eXtensible Markup

[1] http://webar.it.ubi.pt

Language (XML) standard de licença livre que permite a importação e exportação de modelos tridimensionais.

A aplicação desenvolvida usando marcas, faz o posicionamento e orientação do modelo na cena virtual, com base na marca capturada como mostra a figura 4.1. O modelo foi criado no Blender[2] e exportado em formato COLLADA[3]. Este é um formato XML standard de licença livre que permite a importação e exportação de modelos tridimensionais.

A marca utilizada na aplicação, foi criada com uma ferramenta de edição de imagem. Depois com base na imagem resultante foi criado um ficheiro *Pattern*, para ser utilizado pela aplicação como *template* de comparação. O utilizador apenas tem de imprimir a marca, que está também na página da aplicação. Para executar a aplicação o utilizador tem de aceitar que a aplicação aceda à sua webcam. Depois, é só colocar a marca em frente da webcam e o modelo virtual será apresentado como mostra a figura 4.2.

Figura 4.2: Aplicação de RA utilizando marcas, com dois modelos virtuais.

[2]http://www.blender.org
[3]http://www.Collada.org

Com base na aplicação anterior, criou-se mais dois modelos virtuais, que substituíram o já existente. Assim a aplicação tem de representar mais de 3000 triângulos (aproximadamente) e requerer o dobro da memória para a sua utilização. Apesar deste aumento, os FPS apresentados mantêm-se semelhantes à aplicação anterior (ver figura 4.2). Mas caso os modelos a representar sejam compostos por mais de 10000 triângulos (mesmo não sendo normal nestas aplicações actualmente), a média dos FPS desce para 3 (ver figura 4.3).

Figura 4.3: Aplicação de RA para demonstrar a baixa taxa de FPS, para modelos com grande número de triângulos.

A aplicação desenvolvida para suportar modelos com diferentes níveis de detalhe, foi na fase inicial criada com uma primitiva do motor gráfico. Então, utilizando como modelo virtual quatro esferas (primitivas) com diferentes níveis de detalhe, é feita uma representação destes modelos. O modelo virtual visível na aplicação vai ser uma das quatro esferas escolhidas segundo a distância em relação à câmara. É também visível a alteração dos FPS quando é feita a troca dos objectos virtuais (ver figura 4.4).

Posteriormente foi utilizado o modelo do coelho da Universidade de Standford[4], como modelo de teste para a implementação dos LOD's. Também se utilizaram quatro modelos de diferente detalhe, na representação do coelho. Nesta aplicação é possível constatar que a diferença entre os modelos com maior e menor detalhe, não tem relevância na percepção do utilizador. Pois o utilizador encontra-se longe do monitor quando é colocado o objecto com menor detalhe (ver figura 4.4).

[4]http://www-graphics.stanford.edu/data/3Dscanrep/

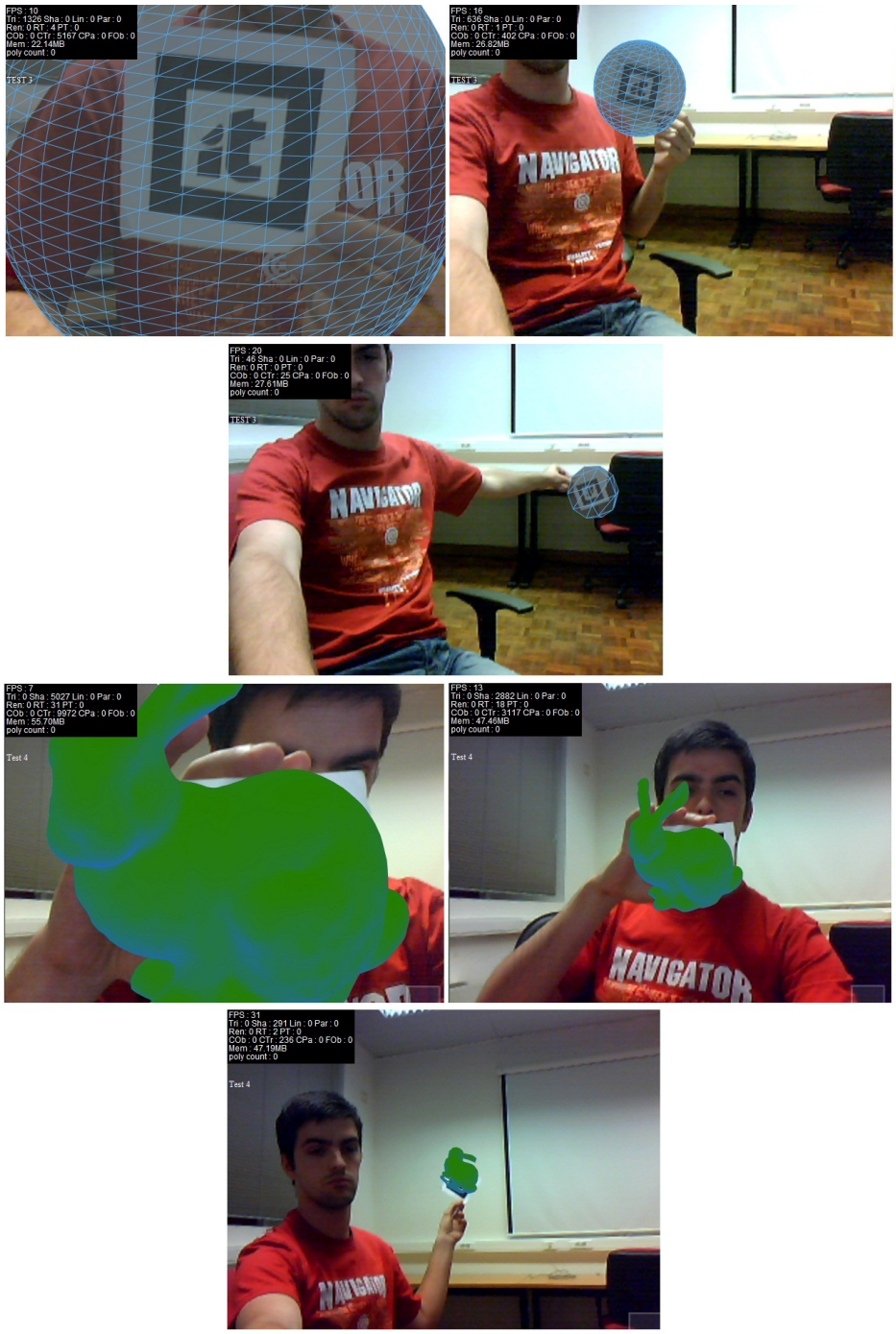


Figura 4.4: Duas aplicações com LOD's, em cima com uma esfera e em baixo com o modelo do coelho.

4.2 Interacção sem Marcas

As primeiras funcionalidades desenvolvidas na aplicação de RA sem marcas, foram a detecção da cara do utilizador e a detecção de movimentos. Na detecção da cara foi utilizado o método de Viola and Jonas. Esta funcionalidade vai então detectar a cara e devolver a sua área de ocupação na imagem capturada, como mostra a figura 4.5.

Figura 4.5: Detecção da cara assinalada por um rectângulo.

Mesmo que na captura esteja presente mais do que uma cara, apenas é considerada última cara detectada através do método. Mesmo sendo uma aplicação vocacionada para um utilizador, um dos melhoramentos a serem feitos, seria corrigir o problema da detecção de múltiplas caras na imagem.

Figura 4.6: Três imagens de detecção de movimentos, no caso da imagem esquerda a detecção não é valida.

Na detecção de movimentos foi criada também uma sub-funcionalidade de subtracção de frames. Existem algumas limitações para este método, pois se a luminosidade do ambiente capturado for muito intensa ou fraca, a detecção não é feita para todos os casos de movimentos. Acontece a mesma situação se os contrastes

da imagem forem pouco acentuados, por exemplo se o local de captura for castanho e o utilizador estiver vestido de castanho. Tal como nas múltiplas detecções de caras, aqui existe o problema de outros movimentos para além dos efectuados pelo utilizador.

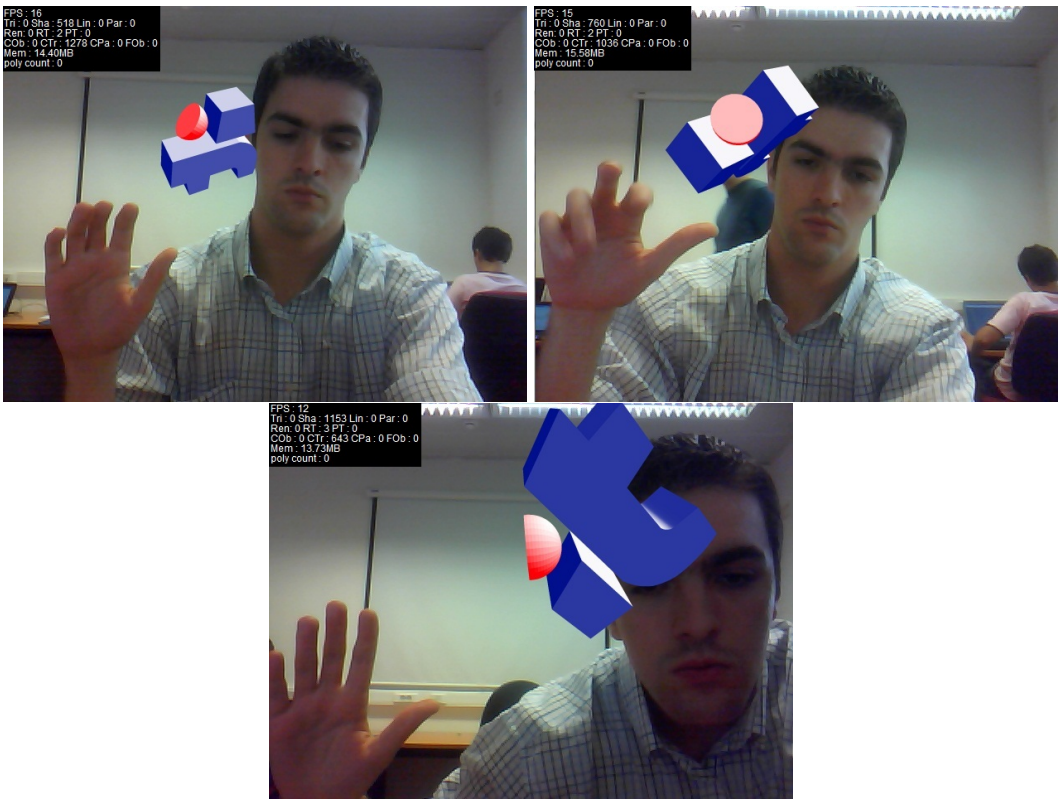


Figura 4.7: Protótipo de RA sem o uso de marcas.

O resultado da junção de todos os métodos criados permitiu criar um primeiro protótipo de RA sem o uso de marcas, capaz de fazer o posicionamento do objecto virtual, tal como a sua rotação (ver figura 4.7). Assim, esta aplicação pode ser comparada com a sua equivalente com marcas na secção 4.1. O utilizador tem a capacidade de deslocar o objecto sempre que a detecção da cara for activada e o mesmo acontece com a detecção de movimentos para a rotação. O desempenho desta aplicação permite obter resultados em tempo real, com uma média de 15 FPS.

Na aplicação actual, quando o utilizador se encontra próximo da câmara de captura, o objecto desaparece ou é cortado (*clipping*). Se a aplicação não tiver este efeito como objectivo, será necessário fazer alterações a nível de posicionamento. Por exemplo a criação de uma *boundingbox* no objecto virtual, onde esta não ultrapasse a área de visão da câmara virtual (*view frustum*).

4.3 Jogo de Realidade Aumentada

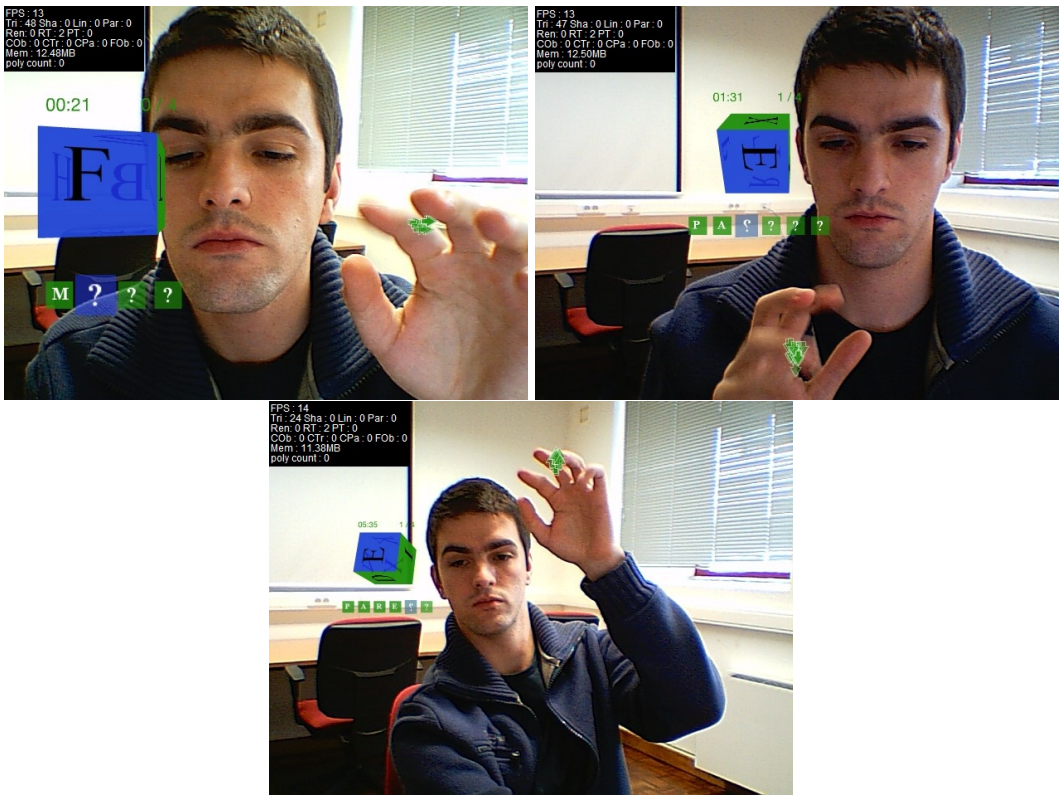


Figura 4.8: Jogo de RA demonstrativo da interacção sem marcas, para descobrir a letra em cada quadrado através do cubo.

Utilizando as técnicas desenvolvidas, foi criado um simples jogo de RA sem marcas, para mostrar as funcionalidades do trabalho desenvolvido. Esta aplicação é de fácil utilização e intuitiva, pois apenas necessita de uma webcam, e do utilizador para interferir e pode ser executada via browser. A aplicação coloca um modelo virtual (um cubo) que o utilizador consegue posicionar esse mesmo modelo com a sua cara e executar rotações com movimentos nas sub-áreas pré-definidas como ilustra a figura 4.8.

O objectivo do jogo é encontrar as letras para completar todas as palavras existentes no jogo. Inicialmente é criada uma sequência de seis letras, onde uma delas será a primeira letra da palavra a descobrir. Cada uma das letras da sequência é distribuída pelas diferentes faces do cubo. O utilizador tem de manipular o cubo até encontrar a letra correcta. Ou seja, até que a face do cubo que contém a letra da palavra fique virada para o utilizador . Depois de encontrada, é criada uma

nova sequência e repetido o processo (ver o exemplo na figura 4.8) até chegar ao fim da palavra e passar para a seguinte. Para dar ao utilizador mais *feedback* da sua interacção, foram criados planos 3D no ambiente virtual correspondentes a cada uma das letras da palavra activa em cada momento. Quando o utilizador descobre uma das letras correctas, esta é colocada no plano correspondente à sua posição, substituindo o ponto de interrogação (colocado por defeito inicialmente em todos os planos). Também se criou um vector de caracteres, que indicam o tempo decorrido e o número de palavras descobertas relativamente ao total de palavras do jogo. Estes objectos tridimensionais vão sempre acompanhar o cubo no ambiente virtual, para que não se perca o efeito de Realidade Aumentada.

Os movimentos efectuados vão permitir a rotação do cubo nos eixos X e Y. A aplicação apenas utiliza 4 áreas de interacção (figura 3.7). Cada ordem de rotação é executada com um valor de 90 graus e só quando esta termina, uma nova ordem pode ser aceite. Isto deixa sempre uma das faces viradas de frente para o utilizador. As zonas superior e inferior da captura, fazem rodar o cubo em direcção à face de cima ou de baixo. As zonas laterais rodam o cubo para a face da esquerda ou da direita.

Para facilitar a visualização, o modelo do cubo tem as faces transparentes para ser visível quais as letras das outras faces. Assim, o utilizador pode aproximar-se da webcam e como o cubo "segue" a sua cara, ele consegue ver as outras faces. Se o utilizador sair da captura, o ambiente virtual fica posicionado na última posição da cara (figura 4.9). As palavras do jogo encontra-se num ficheiro XML e podem ser alteradas e adicionadas novas palavras. Este jogo mostra que as técnicas desenvolvidas funcionam e podem ser usadas em outras aplicações e executadas em ambiente Web.

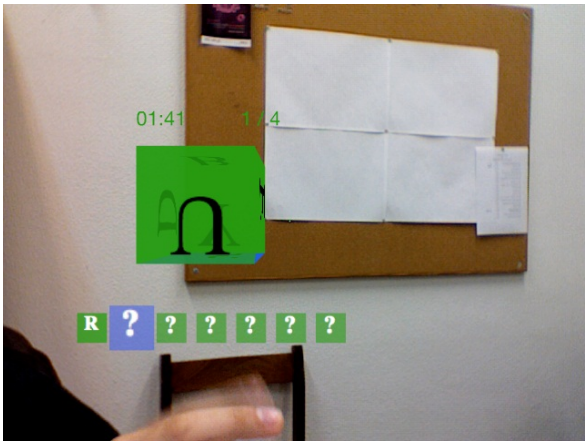


Figura 4.9: Jogo de RA demonstrativo da interacção sem marcas, proposta nesta tese.

Capítulo 5

CONCLUSÕES E TRABALHO FUTURO

5.1 Conclusões

A implementação de aplicações de RA através da linguagem ActionScript revelou resultados satisfatórios. Pois, apesar de não ser a linguagem mais indicada para este tipo de aplicações, esta permite uma compilação para Adobe® Flash. Como um dos objectivos do trabalho era a criação de aplicações para correr no browser, a utilização do ActionScript revelou-se uma escolha apropriada.

Algumas das aplicações de RA desenvolvidas para a Web ficam dependentes da transferência e instalação dum *plugin* ou programa específico para serem executadas. O mesmo não acontece com as aplicações desenvolvidas nesta tese, pois o utilizador apenas necessita do Flash Player genérico da Adobe® que já e suportado por a grande maioria dos browsers e de uma câmara para a captura do ambiente real (ex. webcam).

Para além do fácil acesso à aplicação e dos poucos recursos necessários, foi ainda possível implementar uma aplicação que suporta modelos com vários níveis de detalhe. Contudo, nestas aplicações, a implementação dos LOD's torna-se desnecessária para modelos até 6000 faces (triângulos). Só são notados ganhos significativos na representação de mais de 6000 triângulos. Pois, nestes casos é justificável a utilização de modelos com mais/menos detalhe, para diferentes distâncias. Esta técnica pode ser utilizada quando temos modelos provenientes de

scanners 3D, isto porque neste caso os modelos ficam com um grande número de faces, devido à grande resolução dos scanners.

É visível uma crescente evolução de tecnologias para RA, tal como o aparecimento de várias aplicações da mesma. Desenvolveu-se pois um sistema que explora algumas dessas técnicas criadas e houve a necessidade de implementar outras. Por exemplo, na aplicação sem marcas teve de se implementar uma nova forma de interacção com o sistema de RA. Considerando que a maior parte das aplicações está dependente de marcas para interacção, então o nosso objectivo foi alcançado ao criar um sistema sem marcas. Este sistema continua a ser de Realidade Aumentada, onde os objectos virtuais são adicionados no ambiente real com base na cara do utilizador. A interacção dos objectos virtuais é conseguida com base nos movimentos com sub-áreas pré-definidas. Neste sentido foi concebida uma aplicação que proporciona ao utilizador uma maior liberdade na interacção e não necessita de marcas para funcionar.

Resumindo, o jogo desenvolvido serviu para ilustrar as potencialidades de um sistema de RA, que utiliza uma nova forma de interacção sem marcas e que funciona via Web-Browser.

5.2 Trabalho Futuro

Os pontos de maior importância para trabalho futuro é melhorar alguns aspectos do sistema de interacção sem marcas, mais propriamente a implementação de métodos que permitem executar animações que os modelos virtuais possam conter. Ainda a implementação de um *Loader*, para permitir ao utilizador carregar modelos nos formatos compatíveis para o sistema. Relativamente ao ambiente virtual há a necessidade da criação de uma *bounding box*, para restringir o deslocamento dos objectos virtuais, de modo a que não fiquem demasiado próximos da câmara virtual.

Como já foi referido, outro dos melhoramentos é corrigir o problema da detecção de múltiplas caras na captura. Esta funcionalidade pode passar pela criação de um método, qu guarda a posição da última cara detectada, para na próxima detecção existir uma prioridade sobre a cara das imediações da última.

É evidente que existem outros métodos mais robustos para a detecção de movimentos, mas têm de se avaliar a sua *performance* para poderem ser usados neste

tipo de aplicações em tempo real. Por exemplo, tentar utilizar um algoritmo de *optical flow* [52] para fazer o posicionamento e as rotações dos objectos. Com os dados obtidos pelo método *optical flow*, possivelmente consegue-se uma interacção mais natural, embora a sua implementação provavelmente vá aumentar o custo computacional da aplicação. Outros estudos a serem feitos passam pela adição de algoritmos para o cálculo dos contornos [53], algoritmos de cor [54] e remoção do fundo [48].

A nível da análise de imagem (para a segmentação) o trabalho futuro passa pela criação de valores de ajustamento automáticos, tentando corrigir o problema da variação das condições de iluminação, o que por vezes provoca ruído. O Ruído é muitas vezes introduzido por erros ou deficiência na captura. No entanto tem de se ter em conta que esta aplicação é vocacionada para qualquer tipo de câmara de vídeo. Logo, quanto melhor a qualidade (i.e resolução) tiver a câmara usada, menos ruído vamos ter.

Outro trabalho futuro é analisar a possibilidade de utilizar as tecnologias de RA, com base noutras Application Programming Interfaces (APIs). Por exemplo, APIs dedicadas à criação de gráficos tridimensionais para browser, como a O3D[1] da Google ou WebGL[2] do grupo Khronos. E também experimentar a mais recente ferramenta SLARToolkit[3] lançada para o SilverLight 4 da Microsoft®.

[1] http://code.google.com/apis/o3d
[2] http://www.khronos.org/webgl
[3] http://slartoolkit.codeplex.com/

Referências

[1] Daniel Belcher, Mark Billinghurst, SE Hayes, and Randy Stiles. Using augmented reality for visualizing complex graphs in three dimensions. In *Mixed and Augmented Reality*.

[2] O. Kutter, A. Aichert, C. Bichlmeier, Christoph Bichlmeier, S. M. Heining, B. Ockert, E. Euler, and N. Navab. Real-time Volume Rendering for High Quality Visualization in Augmented Reality. In *International Workshop on Augmented environments for Medical Imaging including Augmented Reality in Computer-aided Surgery (AMI-ARCS 2008)*, New York, USA, Sept. 2008. MICCAI Society.

[3] Roberto Kasuo Miyake, Herbert David Zeman, Flavio Henrique Duarte, Rodrigo Kikuchi, Eduardo Ramacciotti, Gunnar Lovhoiden, and Carlos Vrancken.

[4] Information Browsing Gerhard, Gerhard Reitmayr, and Dieter Schmalstieg. Collaborative augmented reality for outdoor navigation and. In *In Proceedings of the Symposium on Location Based Services and TeleCartography*, pages 31–41. Wiley, 2004.

[5] Ivan Sutherland. A head-mounted three-dimensional display. In *Fall Joint Computer Conf.*

[6] Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *Computers and Graphics IEEE*, 2001.

[7] *3D User Interfaces: Theory and Practice*. Addison-Wesley Professional, 2004.

[8] Bernd Fröhlich, Berthold Kirsch, Wolfgang Krüger, and Gerold Wesche. Further development of the responsive workbench. In *VE '95: Selected papers of the Eurographics workshops on Virtual environments '95*, pages 237–246, London, UK, 1995. Springer-Verlag.

[9] Paul Milgram and Fumio Kishino. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems*, 1994.

[10] Paul Tondeur and Jeff Winder. *Papervision3D Essentials*. Packt Publishing, 2009.

[11] Fotis Liarokapis, Martin White, and Paul Lister. Augmented reality interface toolkit. In *IEEE Proceedings of the Eighth International Conference on Information Visualisation*.

[12] Mark Fiala. Artag, a fiducial marker system using digital techniques. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, pages 590–596, Washington, DC, USA, 2005. IEEE Computer Society.

[13] ADAM. L. Janin, DAVID. W. Mizell, and THOMAS. P. Caudell. Calibration of head-mounted displays for augmented reality applications. In *IEEE Virtual Reality Annual International Symposium*, pages 246–255, 1993.

[14] Cognex. Implementing direct part mark identification: 10 important considerations. white paper of company Cognex,.

[15] José Rouillard. Contextual qr codes. *Computing in the Global Information Technology, International Multi-Conference on*, 0:50–55, 2008.

[16] Daniel Wagner and Dieter Schmalstieg. Artoolkitplus for pose tracking on mobile devices. In *Proceedings of 12th Computer Vision Winter Workshop (CVWW'07)*, 2007.

[17] H. Michael Ji and Earl Killian. Fast parallel crc algorithm and implementation on a configurable processor. In *ICC 2002 IEEE International Conference*.

[18] Daniel Wagner, Dieter Schmalstieg, and Horst Bischof. Multiple target detection and tracking with guaranteed framerates on mobile phones. In *ISMAR '09: Proceedings of the 2009 8th IEEE International Symposium on Mixed and*

Augmented Reality, pages 57–64, Washington, DC, USA, 2009. IEEE Computer Society.

[19] Jun Rekimoto and Katashi Nagao. The world through the computer: computer augmented interaction with real world environments. In *UIST '95: Proceedings of the 8th annual ACM symposium on User interface and software technology*, pages 29–36, New York, NY, USA, 1995. ACM.

[20] J. Rekimoto. Matrix: A realtime object identification and registration method for augmented reality. In *APCHI '98: Proceedings of the Third Asian Pacific Computer and Human Interaction*, page 63, Washington, DC, USA, 1998. IEEE Computer Society.

[21] Pierre Wellner. Interacting with paper on the digitaldesk. *Commun. ACM*, 36(7):87–96, 1993.

[22] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2003.

[23] Terry Ritter. The great crc mystery. *Dr. Dobb's J.*, 11(2):26–34, 1986.

[24] Akira Takahashi, Ikuo Ishii, Hideo Makino, and Makoto Nakashizuk. A high accuracy realtime 3d measuring method of marker fo vr interface by monocular vision. In *In*, pages 167–172. 3D Image Conference 96, 1996.

[25] Barry Kort, Rob Reilly, and Rosalind Picard. An affective model of interplay between emotions and learning: Reengineering educational pedagogy-building a learning companion. In *In*, pages 43–48. IEEE Computer Society, 2001.

[26] Hannes Kaufmann. Collaborative augmented reality in education. *Proc. Imagina 2003 Conf*, 2003.

[27] David Sickinger and Supervisor Burkhard Wuensche. Augmented reality: Combining the artoolkit, itk and vtk for use in a biomedical application, 2004.

[28] Susanna Nilsson and Björn Johansson. Fun and usable: augmented reality instructions in a hospital setting. In *OZCHI '07: Proceedings of the 19th Australasian conference on Computer-Human Interaction*, pages 123–130, New York, NY, USA, 2007. ACM.

[29] Jorge Solis, Nobuki Oshima, Hiroyuki Ishii, Noriyuki Matsuoka, Atsuo Takanishi, and Kazuyuki Hatake. Quantitative assessment of the surgical training methods with the suture/ligature training system wks-2rii. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, ICRA'09, pages 853–858, Piscataway, NJ, USA, 2009. IEEE Press.

[30] Tobias Sielhorst, Tobias Obst, Rainer Burgkart, Robert Riener, and Nassir Navab. An augmented reality delivery simulator for medical training. In *In International Workshop on Augmented Environments for Medical Imaging - MICCAI Satellite Workshop. 141*, 2004.

[31] Hollerer T. Exploring mars: developing indoor and outdoor user interfaces to a mobile augmented reality system. *Computers and Graphics*, 23:779–785(7), December 1999.

[32] Claudio Kirner, Ezequiel R. Zorzal, and Tereza G. Kirner. Case studies on the development of games using augmented reality. In *Systems, Man and Cybernetics*.

[33] Mark Billinghurst, Hirokazu Kato, and Ivan Poupyrev. The magicbook - moving seamlessly between reality and virtuality. *Computer Graphics and Applications IEEE*, 2001.

[34] *Handy AR: Markerless Inspection of Augmented Reality Objects Using Fingertip Tracking*, 2007.

[35] Yan Guo, Qingyun Du, Yi Luo, Weiwei Zhang, and Lu Xu. Application of augmented reality gis in architecture. In *Remote Sensing and Spatial Information Sciences*.

[36] Yu-Chien Chen. A study of comparing the use of augmented reality and physical models in chemistry education. In *VRCIA '06: Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, pages 369–372, New York, NY, USA, 2006. ACM.

[37] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and Dieter Schmalstieg. Pose tracking from natural features on mobile phones. In *ISMAR '08: Proceedings of the 7th IEEE/ACM International Symposium on*

Mixed and Augmented Reality, pages 125–134, Washington, DC, USA, 2008. IEEE Computer Society.

[38] Paul Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Computer Vision and Pattern Recognition*, pages –, 2001.

[39] Ming-Hsuan Yang, David J. Kriegman, and Narendra Ahuja. Detecting faces in images: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(1):34–58, 2002.

[40] Henry A. Rowley, Student Member, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *IEEE Transactions On Pattern Analysis and Machine intelligence*, 20:23–38, 1998.

[41] Henry Schneiderman and Takeo Kanade. A statistical method for 3d object detection applied to faces and cars. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:1746, 2000.

[42] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *International Conference on Computer Vision*, pages 555–562, 1998.

[43] Franklin C. Crow. Summed-area tables for texture mapping. In *International Conference on Computer Graphics and Interactive Techniques*, pages 207–212, 1984.

[44] R. Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. In *IEEE ICIP 2002*, pages 900–903, 2002.

[45] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory: Eurocolt 95*, pages 23–37, 1995.

[46] Ahmed Elgammal, David Harwood, and Larry Davis. Non-parametric model for background subtraction. In *6th European Conference on Computer Vision*, pages 751–767, 2000.

[47] Zhifeng Wang, Yurong Xu, J. Ford, F. S. Makedon, Zhenwu Zheng, Ling Gao, and J. D. Pearlman. An adaptive approach for image subtraction. In *26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1818–1820, 2004.

[48] Qi Zang and R. Klette. Robust background subtraction and maintenance. In *17th International Conference on Pattern Recognition*, pages 90–93, 2004.

[49] Zhen Yu and Yanping Chen. A real-time motion detection algorithm for traffic monitoring systems based on consecutive temporal difference. In *Proceedings of the 7th Asian Control Conference*, pages 1594–1599, 2009.

[50] Tan Kim Heok and D. Daman. A review on level of detail. In *International Conference on Computer Graphics and Imaging and Visualization (CGIV 2004)*, pages 70–75, 2004.

[51] Stephen DiVerdi, Tobias Hollerer, and Richard Schreyer. Level of detail interfaces. In *Third IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2004)*, pages 300–301, 2004.

[52] Ulrich Neumann and Suya You. Natural feature tracking for augmented reality. *IEEE Transactions on Multimedia*, 1(1):53–64, 1999.

[53] Zhan Chaohui, Duan Xiaohui, Xu Shuoyu, Song Zheng, and Luo Min. An improved moving object detection algorithm based on frame difference and edge detection. In *Fourth International Conference on Image and Graphics (ICIG 2007)*, pages 519–523, 2007.

[54] S. J. McKenna, Y. Raja, and S. Gong. Tracking colour objects using adaptive mixture models. *Image and Vision Computing*, 17(3):225–231, 1999.