Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Ciência da Computação

Arcabouço para o Desenvolvimento de Aplicações de Realidade Aumentada para Dispositivos Portáteis com o uso de Múltiplos Sensores

Maurílio da Silva

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Engenharia de Software

Orientadores:
Hyggo Oliveira de Almeida
Angelo Perkusich

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

S586a Silva, Maurílio da.

Arcabouço para o desenvolvimento de aplicações de realidade aumentada para dispositivos portáteis com o uso de múltiplos sensores / Maurílio da Silva. -- 2013.

51 f. : il. color.

Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.

"Orientação: Prof. Dr. Hyggo Almeida de Oliveira, Prof. Dr. Angelo Perkusich".

Referências.

1. Arcabouço de Software. 2. Realidade Aumentada. 3. Dispositivos Portáteis. I. Oliveira, Hyggo de Almeida. II. Perkusich, Angelo. III. Título.

CDU 004.415.2.01(043)

"ARCABOUÇO PARA O DESENVOLVIMENTO DE APLICAÇÕES DE REALIDADE AUMENTADA PARA DISPOSITIVOS PORTÁTEIS COM O USO DE MÚLTIPLOS SENSORES"

MAURILIO DA SILVA

DISSERTAÇÃO APROVADA EM 28/08/2013

HYGGO OLIVEIRA DE ALMEIDA, D.Sc, UFCG
Orientador(a)

ANGELO PERKUSICH, D.Sc, UFCG
Orientador(a)

KYLLER COSTA GORGÔNIO, Dr., UFCG
Examinador(a)

MARCOS RICARDO ALCÂNTARA MORAIS, D.Sc, UFCG
Examinador(a)

CAMPINA GRANDE - PB

Resumo

Ao longo dos anos, pesquisadores e desenvolvedores têm encontrado muitas áreas que podem se beneficiar da Realidade Aumentada (RA). Os primeiros sistemas foram focados em aplicações militares, industriais e médicas, mas sistemas voltados para uso comercial e de entretenimento apareceram logo em seguida. As aplicações de RA em dispositivos portáteis aos poucos estão se tornando uma realidade e à medida que novos recursos, sensores, maior poder de processamento, redes de dados mais velozes vão surgindo e são adicionados a estes dispositivos, esse tipo de aplicação só tende a crescer. Ainda é complexo desenvolver esse tipo de aplicações para dispositivos portáteis e uma possível causa seria a alta demanda de conhecimento necessário para sua implementação. Apesar destas dificuldades, os componentes necessários têm se mantido os mesmos desde os primeiros trabalhos, datados da década de 60. Dessa forma, não são encontrados arcabouços ou formas viáveis de desenvolvimento de sistemas que possam fazer o uso de múltiplos sensores em conjunto (como câmera, GPS, acelerômetro, magnetômetro e giroscópio). Neste trabalho apresenta-se um arcabouço para o desenvolvimento de aplicações de realidade aumentada para dispositivos portáteis utilizando múltiplos sensores. Este arcabouço deve esconder do desenvolvedor a complexidade que envolve o desenvolvimento de aplicações deste tipo e dessa forma permitir que o mesmo foque-se na aplicação e não nos detalhes que envolvem a parte de RA.

Palavras-chave: Arcabouço de Software, Realidade Aumentada, Dispositivos Portáteis

Abstract

Over the years, researchers and developers have found many areas that can take benefit from Augmented Reality (AR). The first systems were focused on military, industrial and medical applications, but commercial and entertainment systems appeared soon after. The AR applications on handheld devices are slowly becoming a reality as new features, sensors, increased processing power and faster data networks are emerging and being added to these devices. However, it is still complex to develop such kind of applications for portable devices and a possible cause is the high demand of knowledge required for their implementation. Despite these difficulties, the necessary components have remained the same since the first works, dating from the 60s. But, it is difficult to find frameworks or viable ways of developing systems that can make use of multiple sensors together (like camera, GPS, accelerometer, magnetometer and gyroscope). In this work, it is presented a framework for the development of augmented reality applications for handheld devices using multiple sensors. The framework hides the complexity from the developer of this kind of applications and thus allows to focus on the application and not on the details involving the AR.

Keywords: Software Framework, Augmented Reality, Handhelds

Agradecimentos

À minha família, em especial Maria Aparecida (Dona Cida, minha mãe) e Amauri (Seu Bida, meu pai), pelo empenho em minha educação, por todo o apoio prestado, pela compreensão e pelo amor incondicional.

Aos meus irmãos Amaury e Marisa, pela força, carinho e estímulo.

À minha noiva, Vivianny Duarte, pela paciência, cumplicidade e apoio incondicionais durante todo este tempo em que estamos juntos. Obrigado por me entender nos momentos em que mais necessito e por estar sempre ao meu lado. Muito obrigado por me fazer uma pessoa mais feliz e realizar um dos meus sonhos: ser pai.

Aos meus amigos Paulo José, Rodrigo Lins, James Aguiar, Valderi Medeiros, Ingridy Eunice e Berg Dantas, pela força, estímulo e boas lembranças.

À Romeryto Lira e Lorena Maia, pelos bons momentos e boas lembranças, mesmo durante os feriados, no laboratório e pela motivação e energia sempre presente.

Aos colegas de curso: Antônio Junior, Adriano Santos e Sergio Espíndola, pelas conversas, pelos momentos de descontração e companheirismo.

Aos alunos do laboratório: Frederico Bublitz, Ivo Calado, Thiago Sales e Marcos Fábio, pelas colaborações.

Ao professor Robson Pequeno por seu apoio, estímulo, atenção e, em especial, por suas aulas de Computação Gráfica que foram essenciais para o correto entendimento e execução da parte 3D deste trabalho.

Aos professores Tiago Massoni e Leandro Dias, pelo estímulo e atenção.

Aos meus orientadores Hyggo Almeida e Angelo Perkusich, pela credibilidade, pelas oportunidades oferecidas, pela paciência e pelos ensinamentos.

Às secretárias da COPIN, Vera Oliveira e Rebeka Lemos, pela presteza e atenção.

À CAPES pelo apoio financeiro durante o período da pesquisa.

Enfim, a todos que contribuíram de alguma forma para a realização deste trabalho.

Conteúdo

Lista de Símbolos

3D - *Three-Dimensional* - Três Dimensões

6DOF - *Six Degrees of Freedom* - Seis Graus de Liberdade

AOSP - *Android Open Source Project* - Projeto de Código Aberto Android

AR - *Augmented Reality*

GPS - *Global Positioning System* - Sistema de Posicionamento Global

JSON - *JavaScript Object Notation* - Notação de Objeto JavaScript

HTTP - *Hypertext Transfer Protocol* - Protocolo de Transferência de Hipertexto

HMD - *Head-Mounted Display*

HUD - *Head-Up Display*

iOS - *iPhone OS*

mTKar - *Mobile ToolKit for Augmented Reality* - Kit de Ferramentas Portatíl para Realidade Aumentada

PC - *Personal Computer* - Computador Pessoal

PDA - *Personal Digital Assistant* - Assistente Pessoal Digital

POI - *Point of Interest* - Ponto de Interesse

RA - *Realidade Aumentada*

RV - *Realidade Virtual*

VR - *Virtual Reality*

Lista de Figuras

Lista de Tabelas

Lista de Códigos Fonte

Capítulo 1

Introdução

A Realidade Aumentada (RA) é reconhecida como uma tecnologia emergente desde 2007 [48] e hoje, com os *smartphones*, os navegadores de RA estão começando a abranger este mais novo tipo de interação homem-máquina. Ao longo dos anos, pesquisadores e desenvolvedores têm encontrado muitas áreas que podem se beneficiar da RA. Os primeiros sistemas foram focados em aplicações militares, industriais e médicas, mas sistemas voltados para uso comercial e de entretenimento apareceram logo em seguida [48].

As principais características dos Sistemas de RA são: a capacidade de combinar objetos reais e virtuais em um ambiente real; registrar (alinhar) estes objetos uns com os outros de modo a permitir a ilusão de completude da cena; serem executados de forma interativa, fazendo o uso de imagens tridimensionais e em tempo real [2, 48]. Devido a essas características, as demandas tecnológicas para o desenvolvimento destes sistemas são muito maiores do que as que são utilizadas para o desenvolvimento de ambientes virtuais (*Virtual Environments* ou Realidade Virtual/RV) [48].

No entanto, as aplicações de RA em dispositivos portáteis aos poucos estão se tornando uma realidade e a medida que novos recursos, sensores, maior poder de processamento e o surgimento de redes de dados mais velozes vão surgindo e sendo adicionados a estes dispositivos, esse tipo de aplicação só tende a crescer. Entretenimento [8, 25], Publicidade [28, 36, 43], Aplicações Médicas [12, 35, 41] e Educacionais [10, 27, 29, 53] são apenas algumas das áreas que podem se beneficiar de um uso massivo dos sistemas de RA.

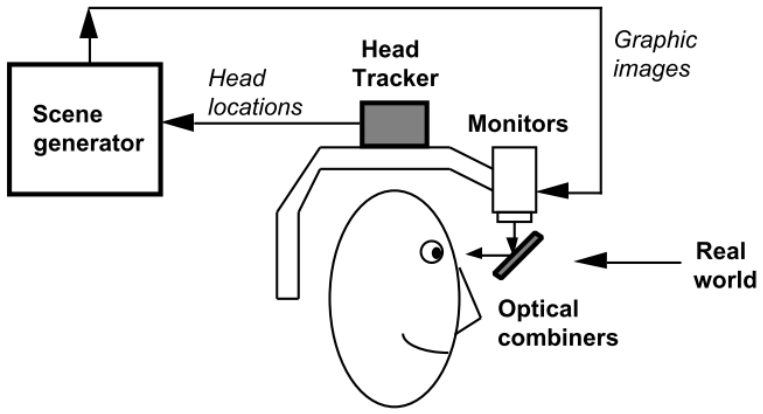
Porém, ainda é complexo desenvolver aplicações de RA para dispositivos portáteis. Uma possível causa seria a alta demanda de conhecimento necessário para sua implementação.

Em muitos casos se faz necessário um aprofundado conhecimento de processamento de imagens e técnicas de visão computacional, que acabam envolvendo alguns domínios da Inteligência Artificial, sem falar na necessidade de se tratar individualmente cada um dos sensores envolvidos. Estes "pré-requisitos" acabam dificultando o desenvolvimento e a massificação de aplicações de RA disponíveis [6].

1.1 Problemática

Apesar das dificuldades que envolvem o desenvolvimento na área de RA, os componentes necessários para o desenvolvimento de sistemas/aplicações têm se mantido os mesmos desde os primeiros trabalhos, datados da década de 60. *Displays*, rastreadores/sensores e softwares de computação gráfica continuam essenciais para a experiência com sistemas de RA [48].

Existem basicamente três técnicas para se apresentar visualmente um sistema de RA: *optical see-through* [3] (Figura 1.1), *video see-through* [31] (Figura 1.2), e projeção [2,54] (Figura 1.3). A técnica que mais se aproxima dos sistemas de RV é o vídeo *"see-through"*, no qual os ambientes virtuais são substituídos por uma fonte de vídeo do ambiente real e os elementos da RA sobrepõem as imagens capturadas do vídeo.

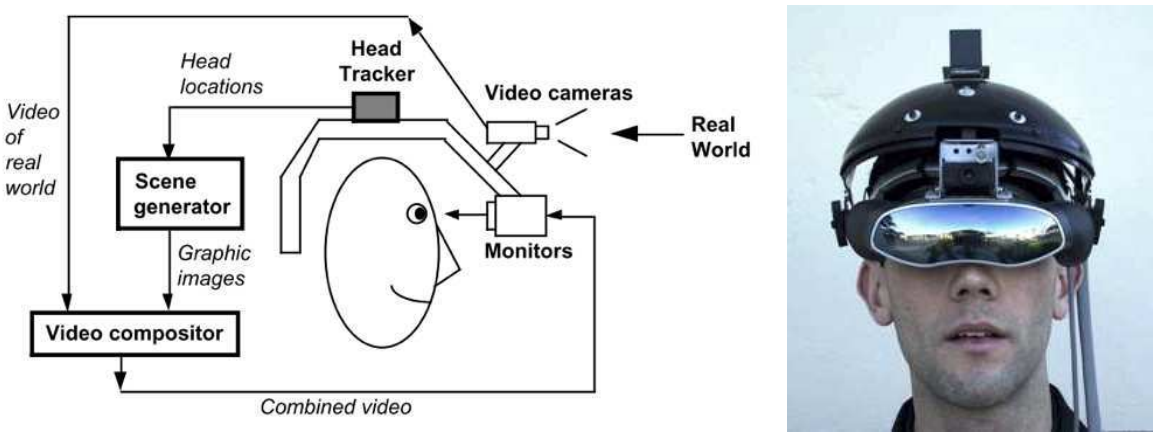


(a) Diagrama [4]

(b) *HMD System* [37]

Figura 1.1: *Optical see-through* HMDs, diagrama conceitual e seu uso

Além de ser a forma mais fácil de se implementar, esta técnica oferece algumas vantagens. Como o ambiente real é digitalizado, torna-se fácil adicionar ou remover objetos, isso inclui substituir marcadores por objetos virtuais. As desvantagens da técnica de ví-



(a) Diagrama [4]

(b) Sony Glasstron HMD [38]

Figura 1.2: *Video see-through* HMD, diagrama conceitual e seu uso

Figura 1.3: Realidade Aumentada Baseada em Projeção [1]

deo *"see-through"* incluem a baixa resolução da "realidade", o limitado campo de visão fornecido e a desorientação causada no usuário devido ao deslocamento dos planos (efeito paralaxe [39, 44]). Uma alternativa é o *"optical see-through"*, utilizada no *Google Glass*[1] (Figura 1.4) que tem como vantagem ser segura para o usuário, pois o mesmo pode continuar vendo mesmo quando há falha na alimentação energética do sistema, fazendo desta técnica

[1] http://www.google.com/glass/

a ideal para propósitos médicos, militares e entretenimento.

Figura 1.4: Google *Glass* [17]

Com a popularização, os dispositivos portáteis, *smartphones* e *tablets*, têm sido classificados como a melhor opção para introduzir a RA ao mercado de massa, devido aos baixos custos de produção e a facilidade de uso, além de serem os mais versáteis ao permitir que os sistemas desenvolvidos possam fazer uso de *"video/optical see-through"* e até projetores portáteis. Segundo [34], sistemas de RA portáteis que utilizam vídeo *"see-through"* para simular uma lente de aumento podem ser baseados em produtos de consumo existentes como os telefones celulares.

No entanto, antes que um sistema de RA possa exibir objetos virtuais em ambientes reais, através da tela de um *smartphone* ou *tablet*, este precisa identificar o ambiente e acompanhar o movimento relativo ao usuário. Porém, ainda hoje, determinar a orientação do usuário continua um problema complexo sem nenhuma solução padronizada. Quando comparados com os sistemas de RV, os rastreadores empregados nos sistemas de RA precisam ter maior precisão dos dados, maior variedade de entradas, maior largura de banda e intervalos mais longos [2, 4].

O rastreamento é, na maioria dos casos, fácil quando os sistemas de RA são projetados para serem utilizados em ambientes fechados (*indoor*), ao invés de ambientes abertos

(*outdoor*). Atualmente, ambientes que não foram preparados para o uso do sistema de RA, utilizando marcadores por exemplo, continuam problemáticos na hora de determinar a orientação e posicionamentos do usuário [48].

Além do rastreamento, registro e interação, Höllerer e Feiner [19] mencionam mais três requisitos para um sistema de RA voltado para sistemas móveis: arcabouço computacional, rede sem fio e armazenamento de dados. Quando combinados estes três itens tendem a diminuir os gastos envolvidos no desenvolvimento de aplicações em geral. No caso de desenvolvimento de aplicações voltadas ao uso de RA em dispositivos portáteis não são encontrados arcabouços de software que possam ser utilizados com este propósito, muito menos formas viáveis de desenvolvimento de sistemas que possam fazer o uso de múltiplos sensores em conjunto (como câmera, GPS, acelerômetro e giroscópio). Existem alguns arcabouços baseados apenas em marcadores, como é o caso do *Android Augmented Reality (AndAR)* [2] e do *Qualcomm Vuforia* [3], mas que não contemplam o uso de múltiplos sensores.

1.2 Objetivo

Neste trabalho tem-se como objetivo desenvolver um arcabouço para o desenvolvimento de aplicações de realidade aumentada para dispositivos portáteis utilizando múltiplos sensores. O arcabouço deve esconder do desenvolvedor a complexidade que envolve o desenvolvimento de aplicações deste tipo e dessa forma permitir que o mesmo foque-se na aplicação e não nos detalhes que envolvem a parte de RA. Assim, reduz o *time-to-market* no desenvolvimento de tais aplicações.

O arcabouço apresentado, intitulado mTKar (*Mobile ToolKit for Augmented Reality*), deve permitir ao desenvolvedor: criar aplicações fazendo uso do sensor de imagem (câmera), dos sensores inerciais (acelerômetro, magnetômetro e giroscópio) e do sensor de posicionamento (GPS). Esses sensores são utilizados com o objetivo de: registrar o posicionamento do usuário, alinhar a visão do mesmo, utilizar estas aplicações em ambientes preparados (*indoor*) e/ou não preparados (*outdoor*), visualizar o resultado do mundo gerado pela aplicação através de vídeo *"see-through"*[4] e economizar tempo, uma vez que o arcabouço esconde a

[2] http://code.google.com/p/andar/

[3] https://www.vuforia.com/platform

[4] Técnica que permite "ver através" da tela do *smartphone/tablet*

complexidade do desenvolvimento de aplicações de RA tornando-se simples de ser utilizado.

O arcabouço proposto disponibiliza três modelos de cenas que o desenvolvedor poderá escolher como base no desenvolvimento de sua aplicação, cada modelo abrange uma determinada classe de sensores e serve para propósitos distintos:

- Modelo 1: **Navegador de RA** - Utiliza o GPS, câmera e sensores inerciais;
- Modelo 2: **Visualizador de Objetos** - Câmera com o uso de marcadores para a orientação e posicionamento dos objetos;
- Modelo 3: **Navegador de Interno de Ambientes** - Utiliza os sensores inerciais.

Com o objetivo de realizar a validação, foram desenvolvidos três aplicativos, para a plataforma Android, cada um fazendo o uso de um dos modelos de cena. Um dos aplicativos foi um Navegador de RA similar ao *Nokia City Lens* [24], no qual o aplicativo recupera a localização do usuário via coordenadas GPS e mostra pontos de interesse ao redor do usuário, estes pontos podem ser, por exemplo, bares, restaurantes, pontos turísticos, etc. O segundo aplicativo faz uso de um marcador para exibir um objeto 3D sobre uma superficie qualquer (onde o marcador esteja posicionado). O terceiro aplicativo apresenta um ambiente interno de uma sala 3D, na qual o usuário pode visualizar a cena como se o mesmo estivesse dentro, para isso ele só precisa posicionar o dispositivo na direção desejada.

1.3 Relevância

O desenvolvimento de aplicações de Realidade Aumentada demanda muito tempo e consequentemente isso implica em custos que podem ser minimizados quando adota-se alguma ferramenta que possa simplificar e agilizar este processo. Para os desenvolvedores isso pode ser ainda mais importante, pois além dos custos e o tempo necessário para este desenvolvimento, faz-se necessário conhecimento específico de visão computacional, processamento de imagens, inteligência artificial e conhecimentos sobre os sensores envolvidos em todo o processo.

Com base nestes pontos, uma das principais contribuições deste trabalho está em permitir, através do arcabouço aqui proposto, que os desenvolvedores possam aumentar a produtividade no desenvolvimento de aplicações de RA voltadas para dispositivos portáteis. Em suma,

a contribuição chave do trabalho é fornecer uma ferramenta eficaz que permita a construção de aplicações de RA de forma simples e rápida, baseada no uso de modelos de cenas.

1.4 Organização

Essa dissertação segue estruturada da seguinte forma:

No **Capítulo 2**, apresenta-se a fundamentação teórica, no qual são apresentados os principais conceitos citados durante o trabalho, disponibilizando embasamento teórico para os leitores. No **Capítulo 3**, são apresentados os principais trabalhos realizados no domínio de RA e Fusão de Sensores. No **Capítulo 4**, apresenta-se o Arcabouço para o Desenvolvimento de Aplicações de Realidade Aumentada para Dispositivos Portáteis com o uso de Múltiplos Sensores. No **Capítulo 5**, são apresentados os três estudos de caso desenvolvidos para validar o arcabouço. E, finalmente, no **Capítulo 6** apresentam-se as considerações finais.

Capítulo 2

Fundamentação teórica

2.1 Computação móvel

Segundo Rebolj [40], a computação móvel é o elo que faltava para um uso eficaz da Tecnologia da Informação de um modo integrado e global. Computação móvel é uma das áreas da computação e seu foco principal é o estudo, análise e desenvolvimento de sistemas computacionais que possam ser utilizados pelos seus usuários em aparelhos independente de localização física.

As partes que formam o conjunto da computação móvel são:

- Equipamentos caracterizados como dispositivos portáteis, ou seja, podem ser utilizados em vários ambientes e com vários propósitos. Exemplos de aparelhos nessa categoria são: notebooks, aparelhos celulares, *Smartphones* e *Tablets*.

- Tecnologias computacionais que permitam o desenvolvimento de aplicações que possam ser usadas em equipamentos móveis como linguagens de programação específicas; a exemplo de: Java (Android), Objective-C (iOS) e C/C++ (Android e iOS)

- Meio de comunicação móvel que permita o envio e recebimento de informações entre aparelhos móveis e estações de processamento de informações, ou entre aparelhos móveis. O meio de comunicação intermediário visa a troca das informações, a longa ou pequena distância.

Dentro dos diversos tipos de dispositivos que são utilizados na computação móvel, os

smartphones tem se destacado nos últimos anos, devido as suas características que os colocam equidade aos computadores.

2.1.1 Smartphones

O *smartphone* oferece recursos de computação, conectividade e telefonia integrados. Seria uma união dos PDA's (*Personal Digital Assistant*) com os telefones celulares. No entanto, de acordo com Babin [5], o smartphone é muito mais do que apenas a soma de dois aparelhos. Assim como os PDA's, os *smartphones* podem executar aplicações como agendas, jogos e programas de comunicação e, claro, realizar chamadas telefônicas. Seu objetivo, no entanto, não é apenas limitar o número de aparelhos que transportamos, mas sim o de combinar o aparelho celular com tecnologias computacionais [5]. Alguns exemplos das habilidades de um *smartphone* se encontram na possibilidade de exibir a informação de um contato ou a foto do mesmo durante uma chamada; tirar fotos, adicionar algum texto e enviar instantaneamente para um PC, para um outro *smartphone* ou diretamente para as redes sociais; ouvir sua música favorita enquanto navega na web; entre outros bastante utilizados pelos usuários de *smartphones*.

Algumas das características encontradas em aparelhos do tipo *smartphone*:

- Estes aparelhos utilizam um sistema operacional mais complexo que são, em sua maioria, munidos de multitarefa;
- Oferecem interfaces visualmente ricas, possibilitando uma melhor experiência do usuário com as aplicações;
- Permite o desenvolvimento de aplicações nativas com o uso de linguagens de programação específicas.

2.1.2 Sistemas Operacionais Móveis

No passado, os dispositivos portáteis não precisavam de sistemas operacionais mais aprimorados, pois estes aparelhos eram simples e todos os softwares eram gravados internamente.

Atualmente existem dois sistemas operacionais voltados para *smartphones* que disputam a liderança do mercado, são eles: Android[1] (foco deste trabalho) e iOS[2] (iPhone OS).

Android

O Android é um sistema operacional desenvolvido para dispositivos portáteis, dentre ele os *smartphones*. Inicialmente o projeto foi desenvolvido pela empresa Android Inc. e em 2005, foi adquirido pela Google.

Após dois anos, em novembro de 2007, a Google anunciou o Android como uma plataforma de desenvolvimento e criou a *Open Handset Alliance*[3], um conselho com mais de 33 empresas parceiras. Em outubro de 2008, o Android se transforma em *Open Source* e tem seu código fonte publicado como AOSP[4], na mesma época foi lançado o primeiro *smartphone* comercial com o sistema Android, o HTC Dream ou G1. Em 2011, a Google em parceria com a Motorola lançou o primeiro *tablet*, o Motorola Xoom, que rodava o Android 3.0, codenome Honeycomb, versão exclusiva para *tablets*.

O Android, atualmente na versão 4.3, codinome *Jelly Bean* é baseado na versão 2.6.35 do Kernel do Linux.

iOS

Inicialmente batizado de iPhone OS, o sistema operacional, desenvolvido pela Apple, teve sua primeira versão anunciada ao público em 2007. Em 2010 a empresa mudou o nome do sistema operacional que passou a se chamar iOS.

O iOS é utilizado nos dispositivos portáteis da Apple: iPhone, iPod e iPad e, compartilha conceitos utilizados no Mac OS X, sistema operacional da Apple usado em ambientes desktop.

A proposta da Apple com o desenvolvimento do iOS e do iPhone é a de transformar a forma como os dispositivos portáteis devem suprir as necessidades do usuários. Interfaces amigáveis combinadas com o uso eficiente das funcionalidades providas pelo *hardware* do

[1] http://www.android.com/

[2] http://developer.apple.com/technologies/ios/

[3] http://www.openhandsetalliance.com/

[4] Android Open Source Project

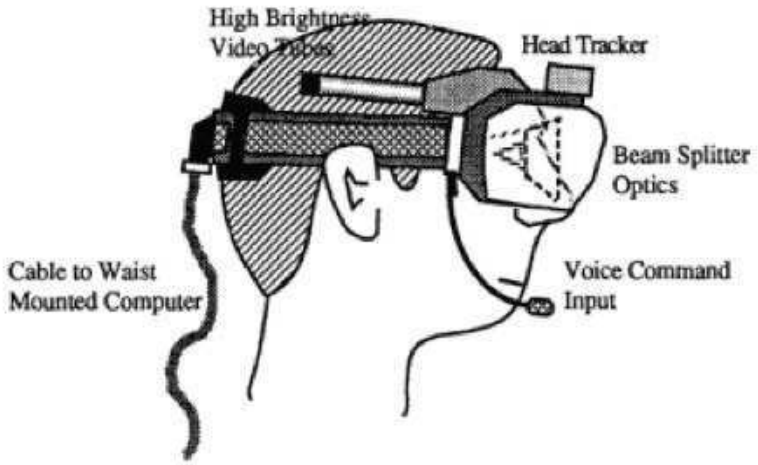
aparelho e aplicações desenvolvidas com um conjunto de regras que maximizem a usabilidade do aparelho é, em essência, a proposta da Apple para o que se considera uma experiência eficiente com um dispositivo portátil.

2.2 Realidade Aumentada

A Realidade Aumentada (RA) é o campo de pesquisa da ciência da computação que combina o mundo real com dados digitais [43]. Em uma combinação de Visão Computacional e Computação Gráfica, é uma tecnologia que permite vislumbrar a "próxima geração de interfaces baseadas na realidade" e está saindo dos laboratórios ao redor do mundo e entrando nas indústrias e nos mercados consumidores. Desde 2007 a RA é reconhecida como uma tecnologia emergente [48] e hoje, com os *smartphones*, os navegadores de RA estão começando a abranger este mais novo tipo de interação homem-máquina. A RA não está limitada a apenas o sentido da visão, podendo ser aplicada a todos os sentidos sensoriais incluindo audição, tato e olfato.

Ao longo dos anos, pesquisadores e desenvolvedores têm encontrado muitas áreas que podem se beneficiar da RA. Os primeiros sistemas foram focados em aplicações militares, industriais e médicas, mas sistemas voltados para uso comercial e de entretenimento apareceram logo depois. Os primeiros protótipos de sistemas de RA, criados pelo pioneiro em computação gráfica Ivan Sutherland e seus alunos na universidade de Harvard e Utah, apareceram na década de 60 e usavam um sistema de visualização do tipo *"see-through"* para apresentar gráficos tridimensionais [46]. Demorou até o início dos anos 90 para que o termo "Realidade Aumentada" fosse utilizado por Caudell e Mizell [7], cientistas da Boeing, os quais estavam desenvolvendo um sistema de RA experimental para ajudar os trabalhadores no trabalho de fiação das aeronaves (Figura 2.1a).

Os sistemas de RA têm como principais características: 1) a capacidade de combinar objetos reais e virtuais em um ambiente real; 2) registrar/alinhar estes objetos uns com os outros de modo a permitir a ilusão de completude da cena; 3) serem executados de forma interativa, fazendo o uso de imagens tridimensionais e em tempo real. No entanto, segundo Siltanen [43], podemos considerar imagens como sendo RA desde que o sistema faça a adição dos objetos em 3D e que exista algum tipo de interação envolvida. Caso alguma



(a) O “HUDset” de Caudell e Mizell [43]

(b) Uso do protótipo [4]

Figura 2.1: O “HUDset” criado por Caudell e Mizell, cientistas da Boeing

dessas características não esteja presente, a aplicação não se enquadra como RA. Devido a essas características, as demandas tecnológicas para o desenvolvimento destes sistemas são muito maiores do que as que são utilizadas para o desenvolvimento de ambientes virtuais (*Virtual Environments* ou VR) [48] e é por isso que a área de RA tem um amadurecimento lento.

Muitas pesquisas tem sido desenvolvidas sobre RA, cada uma destas procura abordar alguma das áreas de aplicação do RA, as quais são apresentadas a seguir:

- **Sistemas de informação pessoal:** Assistência pessoal, Publicidade, Navegação e Turismo. Um exemplo real de aplicação da RA na área de publicidade e *marketing* são os quiosques interativos da LEGO, o qual a criança pode visualizar e interagir com o brinquedo sem a necessidade de tirá-lo da caixa [30], conforme a Figura 2.2.

Figura 2.2: Criança visualizando brinquedo em um quiosque interativo [30]

- **Aplicações industriais e militares:** Projetos, Montagem, Manutenção, Combate e Simulação. Neste tipo de aplicação, a exibição é integrada com o visor protedor dos capacetes, como por exemplo, em treinamento de mecânicos, ao colocar o óculos, o mêcanico verá a explicação de como arrumar as peças ou eventuais problemas do carro.

- **Aplicações médicas:** Simulação e Treinamento. Temos, como exemplo, o uso no apoio a tarefas complexas em cirurgias, inserindo informações adicionais no campo de visão, como tabelas, legendas informativas ou instruções durante um procedimento; visualizando objetos "escondidos", como um Raio-X virtual, baseado em tomografia ou imagens oriundas de ultra-som em tempo real [22].

- **Entretenimento:** Transmissões esportivas e Games. Em transmissões de grandes eventos de natação, na hora da largada, a bandeira do país do atleta é exibida na raia

na qual o mesmo irá nadar e, durante a prova, quando um dos atletas está perto de bater o recorde mundial, uma linha do recorde é exibida. Assim, temos os elementos do mundo real (atletas e piscina), combinados com os virtuais (bandeiras e linha do recorde) como pode ser visto na Figura 2.3.

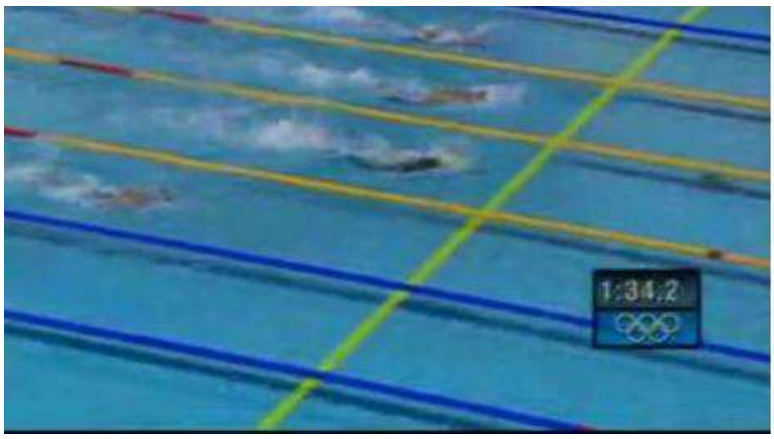


Figura 2.3: Linha do Recorde [11]

- **Realidade aumentada para o escritório:** Colaboração, Comunicação e Coordenação. Usuários podem compartilhar objetos reais e virtuais em meio a um espaço comum no mundo real, não precisando estar diante de um computador.

- **Educação e treinamento: Museus Virtuais**. Museus com atrações aumentadas (Figura 2.4). Nesses museus, os visitantes têm a oportunidade de ficar frente a frente com animais perigosos e que, até mesmo, já foram extintos [16].

Grandes empresas como Google[5], Microsoft[6] [18,50,51] e Nokia[7] [23,32,33] têm investido em pesquisas que envolvam RA e aos poucos os resultados estão sendo disponibilizados ao público em forma de produtos, como é o caso do *Google Goggles*[8], um navegador de RA que usa a captura de imagens (*image scan*) para fazer buscas na web; *Nokia Image Space*[9], aplicação para o compartilhamento de conteúdo (fotos) de forma colaborativa, na qual com as imagens capturadas pelos usuários, um ambiente tridimensional pode ser gerado forne-

[5]http://research.google.com
[6]http://research.microsoft.com/en-us/projects/augmented/
[7]http://research.nokia.com
[8]http://www.google.com/mobile/goggles
[9]http://betalabs.nokia.com/apps/nokia-image-space

Figura 2.4: T-Rex em uma experiência da National Geographic [20]

cendo uma maior imersão do usuário; e do *Nokia City Lens*[10] uma aplicação que fornece informações sobre o local em que o usuário está, como bares, restaurantes etc (Figura 2.5).

Figura 2.5: Nokia *City Lens* [24]

As aplicações de RA em dispositivos portáteis aos poucos estão se tornando uma reali-

[10]http://betalabs.nokia.com/apps/nokia-city-lens

dade e à medida que novos recursos, sensores, maior poder de processamento e o surgimento de redes de dados mais velozes vão surgindo e sendo adicionados a estes dispositivos, esse tipo de aplicação só tende a crescer. Entretenimento, Publicidade, Aplicações Médicas e Educacionais são apenas algumas das áreas que podem se beneficiar de um uso massivo dos sistemas de RA.

No entanto, ainda é complexo desenvolver aplicações de RA para dispositivos portáteis, provavelmente por causa da alta demanda de conhecimento necessário para sua implementação. Em muitos casos se faz necessário um aprofundado conhecimento de processamento de imagens e técnicas de visão computacional, que acabam envolvendo alguns domínios da Inteligência Artificial, sem falar na necessidade de se tratar individualmente cada um dos sensores envolvidos. Estes "pré-requisitos" acabam dificultando o desenvolvimento e a massificação de aplicações de RA disponíveis [6].

Dentre os sensores utilizados para o posicionamento do usuário tem-se o GPS [11], utilizado para o rastreamento em ambientes do tipo *outdoor*; o acelerômetro, o magnetômetro (bússola) e o giroscópio, sensores inerciais, geralmente utilizados em ambientes que não necessitam de preparação; e a câmera, sensor ótico que oferece promissoras abordagens 6DOF[12], para estimativa do posicionamento de usuários e objetos, na maioria das configurações dos sistemas baseados em visualização (*video-based*) [48].

Provavelmente a câmera é um dos dispositivos mais versáteis para se utilizar em sistemas portáteis de RA. Alguns usam homografia[13], quadro a quadro, para estimar o deslocamento e rotação, outros usam o detector de características de Harris [13] para identificar pontos que possam ser utilizados como alvos e alguns outros empregam o algoritmo *Ransac* [14] para uma validação apropriada [21]. A robustez no uso da câmera continua sendo aprimorada e apesar dos custos computacionais serem altos, os resultados desta aproximação baseada apenas em vídeo, de uso geral, para o rastreamento em tempo real é muito promissora.

[11] Global positioning system

[12] Six degrees of freedom: mover para frente/trás, cima/baixo, esquerda/direita e girar nos eixos x, y e z

[13] Verificação de semelhança dos quadros

[14] RANdom SAmple Consensus

Capítulo 3

Trabalhos relacionados

Muito já se foi estudado na área de Realidade Aumentada com o objetivo de fornecer meios que possam ser utilizados no desenvolvimento de sistemas e aplicações, sejam estes em ambientes fechados (*indoor*) ou abertos (*outdoor*). A maioria desses estudos focam-se no desenvolvimento de sistemas que fazem o uso de *hardware* externos, alguns "plugados" ao utilizador como coletes ou óculos, os quais não são considerados portáteis para os padrões que temos hoje. Porém, pouco ou quase nada é referente ao uso de múltiplos sensores em dispositivos portáteis como os *smartphones* e *tablets*.

Nos trabalhos de Krevelen [48] e Azuma [2,4], são apresentadas as principais características da Realidade Aumentada, principais áreas de estudos, problemas e os avanços obtidos ao longo do tempo. Também são apresentadas as vantagens e desvantagens das abordagens tomadas para contornar os problemas existentes e apresenta direcionamentos sobre o futuro da Realidade Aumentada. Por se tratarem de um apanhado de pesquisas (*surveys*) são um ótimo ponto de partida para entender todo o universo da Realidade Aumentada.

Além dos requisitos de rastreamento, registro e interação, base para um sistema de Realidade Aumentada, Krevelen apresenta três outros requisitos para um sistema móvel de RA: arcabouço computacional, rede sem fio e tecnologia de armazenamento de dados.

Krevelen também apresenta as áreas nas quais a RA pode ser aplicada:

- Combate e simulação, aplicações médicas, entretenimento, transmissões esportivas, jogos, educação e treinamento.

Bem como algumas de suas limitações:

- Portabilidade e o uso ao ar livre, rastreamento e (auto) calibração em ambientes sem preparação, percepção de profundidade, sobrecarga visual de informações e aceitação social.

3.1 Fusão de sensores

Em *Global pose estimation using multi-sensor fusion for outdoor augmented reality* [42], é apresentado um sistema (*hardware*) voltado para um ambiente sem preparação (*outdoor*), no qual faz-se o uso do GPS para registrar o posicionamento do usuário e para aumentar a precisão e a qualidade do rastreamento é utilizada a filtragem de Kalman [9, 49] para a fusão dos dados do GPS com o giroscópio, magnetômetro (bússola) e acelerômetro, dessa forma melhorando o resultado dos dados obtidos.

Para compensar as interferências adicionadas aos sensores inerciais giroscópio e acelerômetro, devido ao movimento do dispositivo, e a magnetômetro, por causa dos campos eletromagnéticos do ambiente, é adicionado um rastreador visual que fazendo o uso de técnicas de visão computacional permite a correção das distorções nos três eixos (x, y, z) e dos dados do magnetômetro, dessa forma aumentando a robustez e precisão do sistema.

Apesar de se tratar de um (*hardware*) externo, os autores afirmam que o mesmo pode ser utilizado juntamente com dispositivos portáteis, devido ao seu pequeno peso e formato.

3.2 Orientação espacial do usuário

No trabalho de Lawitzki, *Application of Dynamic Binaural Signals in Acoustic Games* [25], é discutido algumas abordagens técnicas na criação de eventos acústicos espacialmente distintos utilizando dois canais de áudio e apresentado um protótipo de um sistema de áudio que a partir de um único canal (mono) gera áudio de dois canais (stereo). Para que o usuário consiga se localizar espacialmente através do áudio, o protótipo é equipado com um "rastreador de cabeça" (*head-tracking*) que mede a orientação da cabeça do usuário. Este rastreador é utilizado para fornecer os parâmentros espaciais para que o componente responsável pelo áudio possa reagir aos movimentos da cabeça do usuário.

Para atingir o objetivo, o autor utiliza um *smartphone* com sistema Android equipado

com sensores de giroscópio, acelerômetro e magnetômetro (bússola), estes sensores são utilizados para fornecer os dados necessários para medir a rotação da cabeça do usuário de forma rápida e precisa. Para conseguir isso, o mesmo implementa um filtro complementar (Filtro de Kalman) para eliminar o desvio de giro do dados do giroscópio e filtrar o ruído dos dados do acelerômetro e do magnetômetro.

O estudo apresentado neste trabalho pode ser facilmente utilizado em sistemas de realidade virtual (VR), em que o usuário tem sua visão totalmente coberta por um display[1], bem como em sistemas de RA voltados para pessoas com algum tipo de deficiência visual, em ambos os exemplos os usuários teriam o sentido da audição aumentado.

3.3 Realidade Aumentada baseada em marcadores

Em *Theory and Applications of Marker-based Augmented Reality* [43], a autora procura responder algumas das principais dúvidas relativas a Realidade Aumentada do tipo *indoor*, baseada em marcadores, e que usam a câmera com principal meio para registro e rastreamento do ambiente. O trabalho é um apanhado da experiência acadêmica e profissional da autora na área de RA e o mesmo responde as perguntas: O que está por trás do termo "realidade aumentada"? Qual é a tecnologia e os algoritmos que permitem adicionar conteúdo 3D à realidade? Quais são os limites e as possibilidades da tecnologia?

O trabalho explica os algoritmos e métodos que permitem criar a ilusão de uma "convivência" aumentada de conteúdo digital e real. Discutem-se as melhores maneiras de gerenciar interações nos sistemas de RA e, também, mostra os limites e as possibilidades da RA e seu uso.

O foco do trabalho está nas aplicações de RA baseadas em marcadores (*marker-based*), no qual durante as pesquisas a autora desenvolveu métodos para uma rápida e robusta detecção, identificação e rastreamento destes marcadores, tornando possível o uso destes em dispositivos portáteis. As principais áreas tratadas no trabalho são "montagem aumentada" e "design de interiores". O mesmo, também, pode ser considerado como um apanhado de pesquisa (*survey*) de RA baseada em marcadores, pois este fornece uma visão geral sobre o estado da arte, possibilidades e aplicações da RA.

[1] ex.: Oculus Rift: http://www.oculusvr.com/

Além do foco em marcadores, a autora cobre outros métodos alternativos para o rastreamento visual, como rastreamento baseado em características, baseado em modelos e o uso de sensores no rastreamento híbrido. Todos os métodos discutidos no trabalho tem como principais características o uso da câmera como meio principal (quando não o único) para o registro e rastreamento da cena.

3.4 Conclusão

Nesta seção foram apresentados os principais trabalhos na área de Realidade Aumentada que contribuiram de forma direta ou inderatmente para a solução aqui proposta. As soluções apresentadas nestes trabalhos focam-se em resolver problemas distintos da RA, como o uso da câmera para melhorar os resultados obtidos através dos sensores ou o uso do filtro de Kalman para corrigir os dados provenientes dos sensores. Os trabalhos podem ser divididos em dois tipos: 1) os que tratam de RA com sensores e a resolução de seus problemas e, 2) os que tratam de RA com o uso de marcadores, usando para isso a câmera para a obtenção dos dados para registro e posicionamento do usuário e da cena.

Embora os trabalhos tenham a RA em comum, poucos deles tem foco em dispositivos portáteis e o uso de um arcabouço para facilitar ou agilizar o desenvolvimento desse tipo de aplicação também não é visto, especialmente quando o foco são os dispositivos portáveis. Siltanen [43] discute sobre os arcabouços disponíveis para o desenvolvimento de aplicações de RA e cita o ARToolkit[2] como sendo o primeiro deste tipo. Este arcabouço tem versões para as mais diversas plataformas móveis, como Android[3], bem como para desktop e web[4].

A diferença destes trabalhos para o aqui desenvolvido é que através do arcabouço o desenvolvedor poderá criar aplicações de RA com o mínimo de esforço possível, sem precisar se preocupar com a parte mais técnica e de baixo nível da RA, como a aquisição e filtragem dos dados dos sensores ou como ler um marcador e projetar sobre ele um objeto 3D. Com isso, o trabalho no desenvolvimento de aplicações de RA se torna mais simplificado e a complexidade do desenvolvimento será escondida pelo arcabouço.

[2]http://www.hitl.washington.edu/artoolkit/

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

[4]http://www.libspark.org/wiki/saqoosha/FLARToolKit/en

Capítulo 4

mTKar

Neste capítulo, apresenta-se o arcabouço que possibilita o desenvolvimento de aplicações de RA voltada para dispositivos portatéis de forma rápida e simples. Os elementos vistos na Figura 4.1 são explicados em alto nível, de modo a mostrar o funcionamento desde a coleta dos dados dos sensores, passando pela parte processamento dos dados e finalizando na projeção da cena aumentada para o usuário.

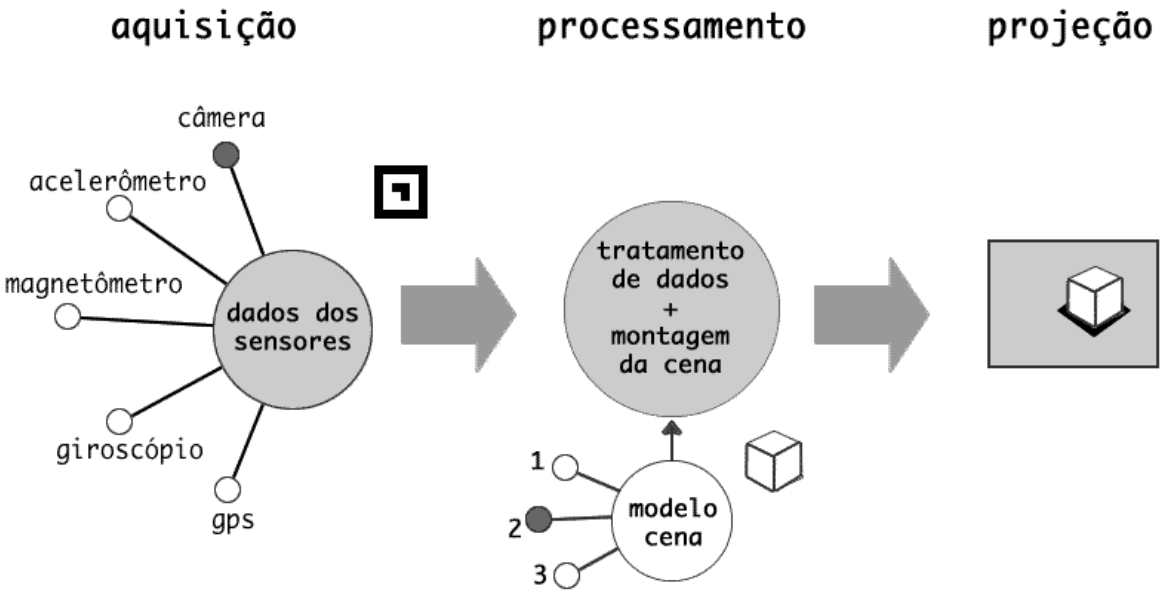


Figura 4.1: Visão geral do Arcabouço

4.1 Visão Geral

O arcabouço desenvolvido neste trabalho disponibiliza três modelos de cenas (*templates*). As cenas, de forma resumida, são definidas como: **Câmera + Marcador**, **Câmera + Sensores** e **Sensores**. Cada uma tem um conjunto de sensores envolvidos, como pode ser visto na Figura 4.2, e foram projetadas para permitir a criação de uma boa variedade de aplicações voltadas para entretenimento, educação, jogos e publicidade. Os sensores foram agrupados em três grupos:

- Sensor de imagem
  - Câmera
- Sensor de posicionamento global
  - GPS
- Sensores inerciais:
  - Acelerômetro
  - Magnetômetro
  - Giroscópio

Figura 4.2: Tipos de cena e os sensores envolvidos

Cada grupo é responsável por fornecer um determinado tipo de dado para o arcabouço, que juntos irão permitir a geração da cena aumentada. A câmera fornece os dados visuais do mundo real e é o único sensor envolvido nas aplicações que fazem o uso de marcadores; o GPS fornece o posicionamento global do usuário, permitindo a criação de aplicações baseadas em localização, como publicidade e navegadores de RA; por último, e não menos importante, temos os sensores inerciais, responsáveis pelo correto posicionamento da cena em relação ao posicionamento do usuário/dispositivo e apesar de existirem três sensores envolvidos, apenas um conjunto de dados é gerado ao final e dependendo da disponibilidade dos sensores no dispositivos apenas dois deles podem ser utilizados, porém com menor precisão nos dados. Neste caso o sensor giroscópio, utilizado para aumentar a precisão dos dados é opcional na utilização do arcabouço.

Abaixo é apresentado como o desenvolvedor escolhe o tipo de cena e como são determinados os sensores a serem utilizados. Na Figura 4.3 é apresentado este processo em forma de um diagrama de blocos.

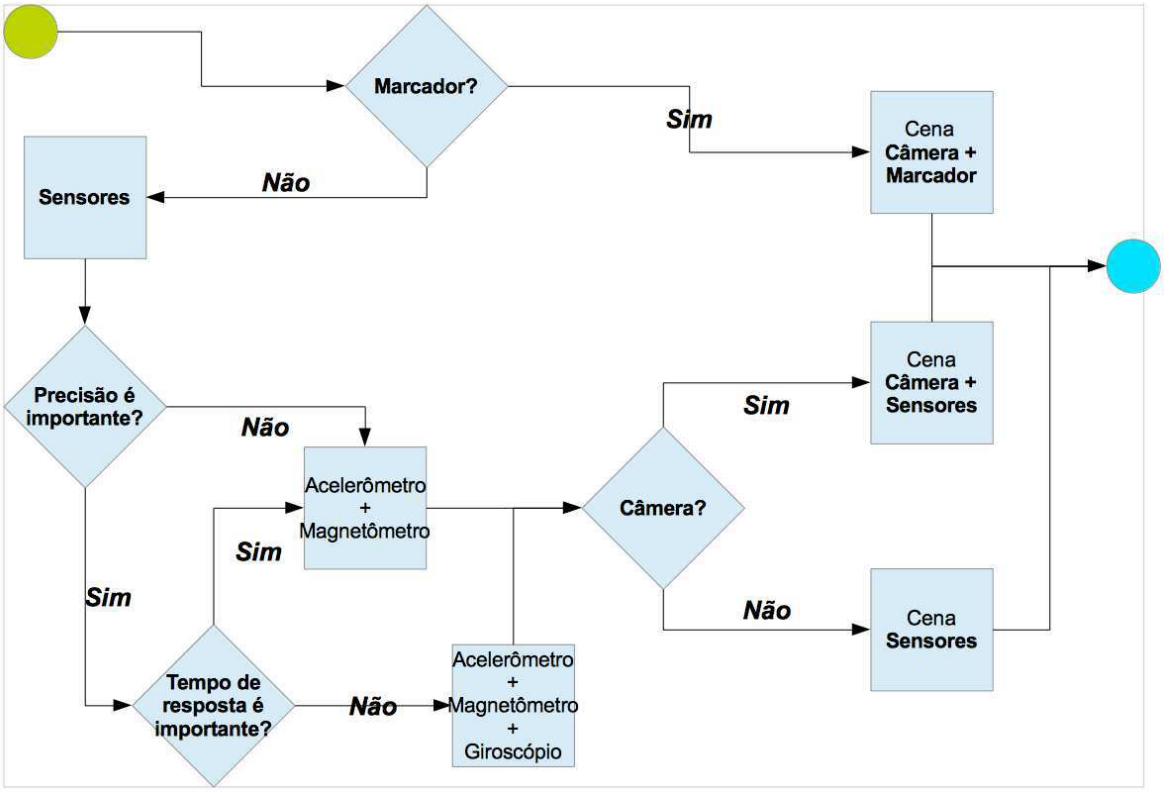


Figura 4.3: Seleção do tipo de cena

4.2 Arquitetura do Arcabouço

Nesta seção, apresenta-se a arquitetura do arcabouço de forma mais detalhada, mostrando como se comporta cada componente do mesmo. O Arcabouço está dividido em quatro módulos: *Aquisição*, *Tratamento*, *Montagem* e *Projeção*. A divisão do mesmo pode ser vista na Figura 4.4.

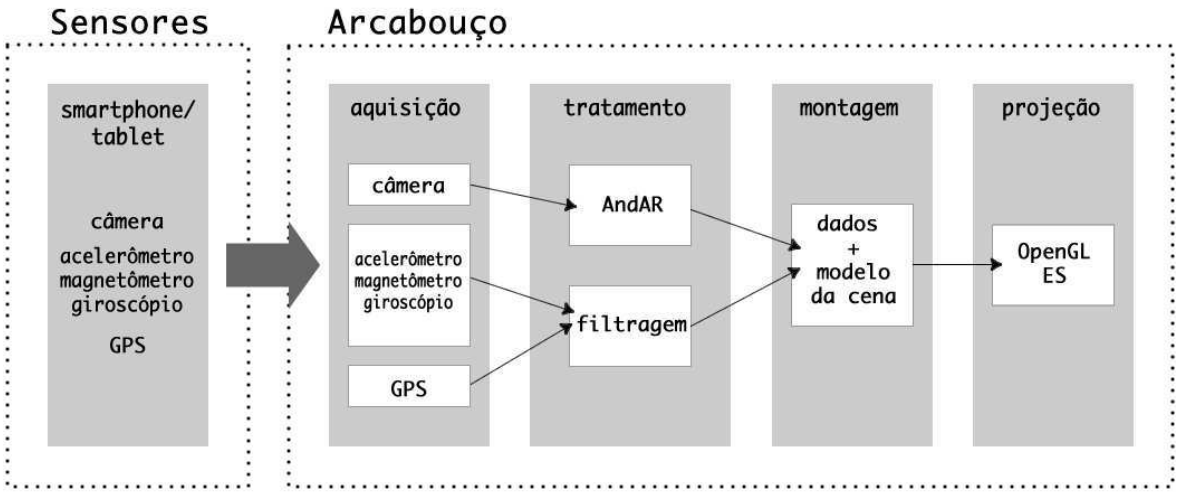


Figura 4.4: Arquitetura do Arcabouço

Inicialmente apresenta-se a *Camada de Sensores* na qual estão presentes os sensores do dispositivo que serão utilizados para a captura dos dados. Esta camada é responsável pela leitura, coleta e envio dos dados dos sensores do dispositivo para o arcabouço através do *Módulo de Aquisição de Dados*. Ao receber os dados, o módulo de *Aquisição* faz a devida filtragem e envia para o *Módulo de Tratamento dos Dados* que é composto de dois submódulos: 1) responsável por efetuar o tratamendo dos dados da câmera, quando a cena envolve marcadores; e 2) responsável por fazer o tratamento dos dados provenientes dos sensores, como a fusão dos dados dos sensores inerciais. Os dados tratados neste módulo são submetidos para o *Módulo de Montagem*, o qual, com base nos dados recebidos irá efetuar a devida montagem dos dados do mundo real (câmera e/ou sensores) com os dados do ambiente virtual (objetos 3D), gerando a cena aumentada. Após a montagem, o *Módulo de Projeção* gera a imagem final que será projetada na tela do dispositivo através da biblioteca

OpenGL ES[1].

Na Tabela 4.1 pode-se ver uma descrição das funcionalidades dos componentes do arcabouço.

Componente	**Descrição**
Módulo de Aquisição	Responsável pelo encaminhamento dos dados para o Módulo de Tratamento;
Módulo de Tratamento	Responsável pelo tratamento e extração dos dados que irão servir para compor a cena;
Módulo de Montagem	Responsável pela junção dos dados obtidos do do mundo real através dos sensores, com as informações e objetos do mundo virtual (objetos 3D);
Módulo de Projeção	Responsável pela projeção, no display do dispositivo, da cena gerada pelo Módulo de Montagem.

Tabela 4.1: Principais componentes da arquitetura do arcabouço

4.2.1 Módulo de Aquisição

Este módulo é o responsável pela aquisição dos dados dos sensores presentes no dispositivo, sua principal responsabilidade é identificar, baseado no tipo de cena, o sensor de origem dos dados para que estes sejam encaminhados ao submódulo correto do *Módulo de Tratamento*.

O Módulo é composto por três classes:

- `DataFromCamera`: responsável pelos dados provenientes da câmera e envio para o submódulo de tratamento de imagens;
- `DataFromSensors`: responsável pelos dados provenientes do acelerômetro, magnetômetro e giroscópio e envio para o submódulo de tratamento de sensores;
- `DataFromGPS`: responsável pelos dados provenientes do GPS e envio para o submódulo de tratamento de sensores.

[1] http://www.khronos.org/opengles/

Estas classes, além de coletar e encaminhar os dados dos sensores, são responsáveis pela detecção e inicialização dos mesmos. Por exemplo, caso não exista câmera traseira presente no dispositivo, o arcabouço não poderá ser utilizado, pois é uma exigência do mesmo. Um outro exemplo, é quando o sensor do giroscópio não está presente no dispositivo, neste caso o arcabouço ignora este sensor, uma vez que o mesmo é um sensor opcional utilizado apenas para uma maior precisão dos dados

Apesar de existirem apenas dois submódulos de tratamento para os dados, faz-se necessária a existência das três classes para a correta inicialização e tratamento dos mesmos.

4.2.2 Módulo de Tratamento

O *Módulo de Tratamento* é o responsável pelo tratamento dos dados provenientes da câmera, sensores inerciais e do GPS, e é composto de dois submódulos: `ImageData` e `SensorData`.

`ImageData`

Módulo responsável pelo tratamento dos dados provenientes da Câmera. É utilizado apenas nas aplicações que fazem o uso de **Câmera + Marcador**.

Este módulo usa a biblioteca AndAR[2], implementação da biblioteca ARToolkit[3] para a plataforma Android. Nele é efetuado todo o trabalho de computação gráfica e visão computacional necessários para a correta detecção, identificação, alinhamento e o rastreamento de objetos nas imagens.

`SensorData`

Este é o principal módulo do arcabouço, sua principal função é efetuar a fusão dos dados provenientes dos sensores inerciais uma vez que os dados do GPS não precisam desta filtragem por não sofrerem de interferências externas ou internas do aparelho. Essa fusão de dados é feita através de uma implementação do Filtro de Kalman[4] apresentada por Lawitzki [25] e

[2] http://code.google.com/p/andar/

[3] http://www.hitl.washington.edu/artoolkit/

[4] http://www.cs.unc.edu/∼welch/kalman/

que pode ser obtida em seu blog pessoal[5].

No Android, o meio utilizado para se obter a orientação tridimensional de um dispositivo é utilizando o método `SensorManager.getOrientation()`, este método irá retornar os dados dos três ângulos. O retorno é baseado nos dados do acelerômetro e do magnetômetro, onde, o acelerômetro fornece o vetor de gravidade (relativo ao centro da Terra) e o magnetômetro faz o papel de bússola. Os dados destes dois sensores são suficientes para calcular a orientação do dispositivo. No entanto, ambos os sensores são imprecisos, especialmente os dados oriundos do magnetômetro que inclui uma grande quantidade de ruído.

O giroscópio presente no dispositivo é muito mais preciso e tem um tempo de resposta relativamente curto. Porém, tem como ponto negativo o **desvio de giro** (*gyro drift* [52]). O giroscópio fornece a velocidade de rotação angular nos três eixos e para se obter a orientação atual do dispositivo, estes valores precisam ser integrados ao longo do tempo. Isto é obtido através da multiplicação das velocidades angulares, com o intervalo de tempo entre a última leitura dos dados com a atual, fornecendo assim um incremento de rotação. A soma de todos os incrementos de rotação fornece a orientação absoluta do dispositivo. Durante este processo, pequenos erros são introduzidos em cada interação. Estes pequenos erros que são adicionados ao longo do tempo, acabam resultando em uma rotação lenta (constante) na orientação calculada, conhecida como desvio de giro.

Assim, para evitar ambos, o desvio de giro e uma orientação cheia de ruídos, os dados do giroscópio são aplicados apenas para as mudanças de orientação ocorridos em pequenos intervalos de tempo, enquanto que os dados do magnetômetro/acelerômetro são utilizados durante longos períodos de tempo. Isto é equivalente a uma filtragem passa-baixa [6] dos sinais do acelerômetro e do magnetômetro e uma filtragem passa-alta dos sinais do giroscópio. Uma visão geral do filtro da fusão dos sensores é apresentada na Figura 4.5.

O funcionamento dos filtros de passa-baixa [7] e passa-alta nos dados dos sensores é o seguinte: os dados dos sensores são obtidos em intervalos regulares (maiores ou menores). O filtro passa-baixa dos dados com ruído do acelerômetro/magnetômetro (*accMagOrientation* na Figura 4.5) são uma média dos ângulos de orientação durante um período de tempo com um intervalo constante. Isto é realizado introduzindo lentamente novos valores do acelerô-

[5] http://www.thousand-thoughts.com/

[6] Permite a passagem de valores acima do limiar pré-definido

[7] Permite a passagem de valores abaixo do limiar pré-definido

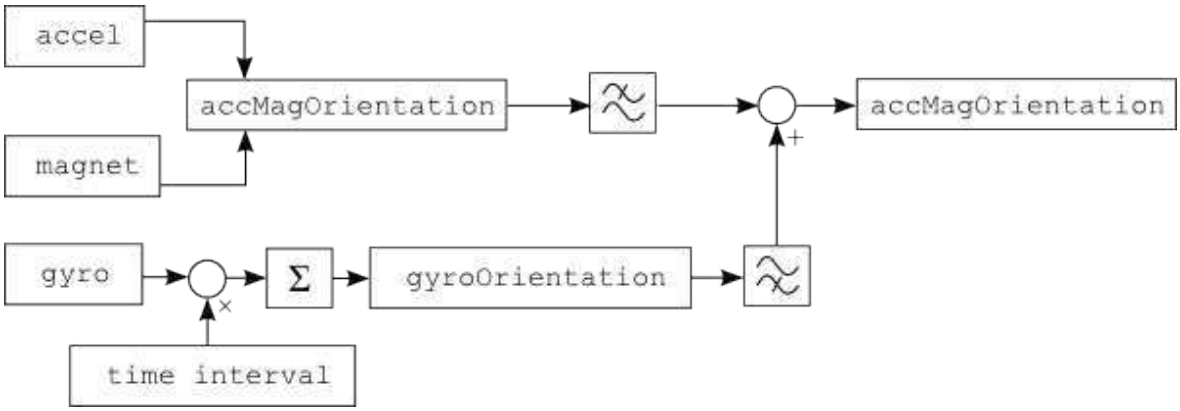


Figura 4.5: Fusão de Sensores [26]

metro/magnetômetro aos dados de orientação absolutos conforme o Código 4.1 abaixo:

Código Fonte 4.1: Filtro passa-baixa

```
1 accMagOrientation = (1 - factor) * accMagOrientation + factor *
    newAccMagValue;
```

A filtragem passa-alta dos dados do giroscópio é realizada ao substituir os componentes filtrados de alta-frequência de *accMagOrientation* pelos valores de orientação correspondentes do giroscópio, conforme o Código 4.2 abaixo pode ser visualizado o filtro final:

Código Fonte 4.2: Filtro passa-alta

```
1 fusedOrientation =
2   (1 - factor) * newGyroValue;   // componente de alta-frequência
3   + factor * newAccMagValue;   // componente de baixa-frequência
```

Um exemplo de funcionamento do filtro: assumindo que o dispositivo é rotacionado 90º em uma determinada direção e após um pequeno intervalo de tempo o mesmo é colocado em sua posição inicial, os dados intermediários no processo de filtragem irá se assemelhar com a Figura 4.6 [8].

[8]Nota: o desvio de giro, integrado ao sinal do giroscópio na Figura 4.6, é o resultado de pequenas irregularidades na velocidade angular original. Estas pequenas derivações são adicionadas à integração e causam uma indesejada lentidão adicional na rotação da orientação baseada no giroscópio.

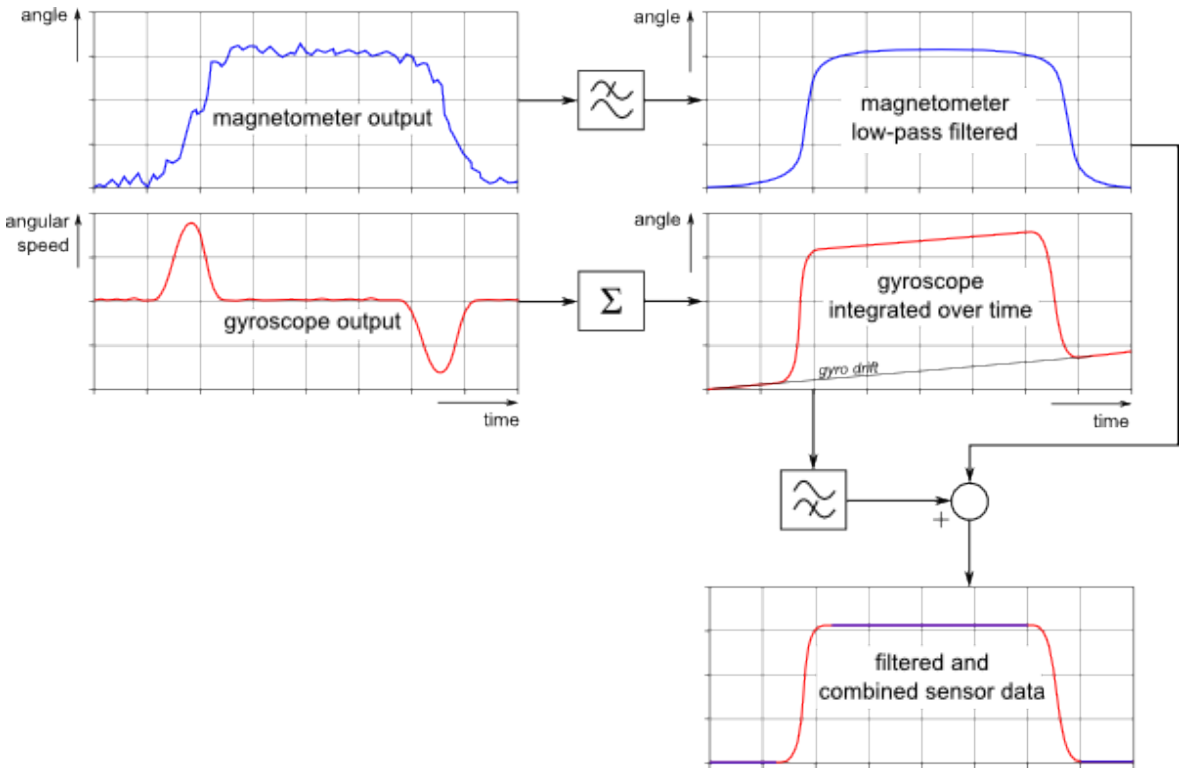


Figura 4.6: Exemplo do funcionamento do filtro de Kalman [26]

4.2.3 Módulo de Montagem

O *Módulo de Montagem* é o módulo com o qual o desenvolvedor interage com o arcabouço, este módulo é o responsável por integrar os dados de localização, oriundos do GPS; os dados de posicionamento, obtidos dos sensores; e os dados de imagem, obtidos da câmera e pre-processados pela biblioteca AndAR, além de parâmentros e dados fornecidos pelo desenvolvedor.

Durante a implementação da cena, alguns métodos precisam ser sobrescritos. De uma forma geral as cenas possuem em comum um conjunto de métodos, listados no Código 4.3 e descritos a seguir: o método `init()` é chamado na inicialização da cena. O método `pause()` é chamado toda vez que alguma interrupção do SO do aparelho “esconde"ou sobrepõe a tela do aplicativo. O método `update()` é chamado a cada *loop* da cena. O método `render()`, é chamado durante a projeção da cena na tela por parte do *Módulo de Projeção*. O método `dispose()` é chamado para liberar os recursos utilizados pela cena, normalmente no encerramento do aplicativo.

Código Fonte 4.3: Métodos padrão dos modelos de cena

```
1   public void init(){    }
2   public void pause(){   }
3   public void update(){   }
4   public void render(){   }
5   public void dispose(){   }
```

No método `update()`, o desenvolvedor poderá interagir diretamente com o módulo, passando, obtendo e atualizando parâmetros que serão utilizados na montagem e atualização da cena.

Cada modelo de cena também possui um conjunto próprio de métodos, que podem ser utilizados para interagir com a mesma.

Modelo 1: Câmera + Sensores

O **Modelo 1** tem como base o uso da Câmera + Sensores, neste modelo a câmera serve apenas para captura o vídeo que será utilizado na plotagem dos objetos virtuais no ambiente real e o desenvolvedor pode interargir com a cena através de quatro métodos.

Os métodos que podem ser utilizados pelo desenvolvedor para interagir com a cena são: o método `getLocation()` irá retornar as coodenadas GPS atuais. O Método `setDistanceUpdate()` é o responsável por passar a distância miníma de deslocamento necessária para a atualização da cena. O método `setData()` é o responsável por passar os dados dos POIs para o arcabouço. O método `setStyle()` permite a personalização das etiquenas exibidas para mostrar os POIs para o usuário da aplicação, no Código 4.4 são apresentandos os métodos com os paramêtros e os tipos de retorno dos mesmos.

Código Fonte 4.4: Métodos do Modelo de Cena 1

```
1   public Point2D getLocation() {   }
2   public void setDistanceUpdate(int meters){   }
3   public void setData(JSONObject data){   }
4   public void setStyle(int bgColor, int titleColor, int textColor}{   }
```

Modelo 2: Câmera + Marcadores

O **Modelo 2** tem como base o uso da Câmera + Marcadores, neste modelo a câmera serve para captura da imagem do marcador, no qual o objeto 3D será exibido e alinhado. O desenvolvedor pode interargir com a cena através do método `loadObject()`.

O método `loadObject()` é utilizado para definir o objeto 3D que será projetado sobre o marcador, o Código 4.5 apresenta o tipo de retorno e o paramêtro necessário para sua utilização.

Código Fonte 4.5: Métodos do Modelo de Cena 2

```
1   public void loadObject(Object3d obj){  }
```

Modelo 3: Sensores

O **Modelo 3** é baseado apenas nos sensores inerciais, neste modelo a câmera não é utilizada, dessa forma poderá ser utilizado em qualquer aparelho que disponha de acelerômetro/magnetômetro. Os objetos são projetados em um ambiente virtual e o deslocamento neste ambiente é feito através dos sensores inerciais do aparelho. O desenvolvedor pode interargir com a cena através de oito métodos.

Dos oito métodos existentes neste modelo de cena, quatro deles são voltados para a movimentação dentro da cena, os principais métodos são: `loadObject()` responsável pelo carregamento do objeto 3D que servirá de base para a cena; `setPosition()` permite o posicionamento do usuário relativo ao objeto/cena; `resetPosition()` reinicia o posicionamento do usuário para o ponto inicial; `getPosition()` devolve um ponto 3D da posição do usuário na cena.

Os métodos utilizados para movimentar o usuário dentro da cena, permitindo que o mesmo possa se locomover pelo ambiente, sem precisar se movimentar nas quatro direções, são: o método `walkToFront()` permite o deslocamento do usuário para a frente; `walkToBack()` deslocamento para trás; `walkToLeft()` deslocamento para a esquerda; e, `walkToRight()` desloca o usuário para a direita. Na Código 4.6 são apresentandos os métodos com os paramêtros e os tipos de retorno dos mesmos.

Código Fonte 4.6: Métodos do Modelo de Cena 3

```
1   // métodos específicos
2   public void loadObject(Object3d obj){  }
3   public void setPosition(Point3D pos){  }
4   public void resetPosition(){  }
5   public Point3D getPosition(){  }
6
7   // métodos utilizados para se deslocar na cena
8   public void walkToFront(int steps) {  }
9   public void walkToBack(int steps) {  }
10  public void walkToLeft(int steps) {  }
11  public void walkToRight(int steps) {  }
```

4.2.4 Módulo de Projeção

O *Módulo de Projeção*, último estágio do arcabouço, é o responsável por mostrar o resultado do processamento realizados nos demais módulos de forma visual para o usuário. O desenvolvedor poderá interagir com este módulo através do método `render()` da cena, para isso precisa apenas sobrescrevê-lo, dessa forma uma camada extra, *layer*, será adicionado ao topo da cena antes dela ser projetada no *display* do dispositivo.

O desenho da cena, *render*, é realizado utilizando a biblioteca OpenGL ES[9], disponível de forma nativa no Android e responsável por acelerar o processamento gráfico nos dispositivos.

De forma resumida, este módulo trabalha pegando os dados processados pelo módulo de tratamento; em seguida é feita a projeção da cena com base nos dados recebidos; então, a matriz de projeção do OpenGL é ajustada para refletir a janela de visualização, neste caso o *display* do dispositivo; caso o usuário tenha fornecido alguma informação extra para o desenho, no método `render()`, uma camada extra é posicionada sobre a cena gerada; e, por último, a cena é então projetada ao usuário através do *display* do dispositivo.

4.2.5 Extensibilidade do Arcabouço

Um Arcabouço deve ter como requisito a extensibilidade como forma de expandir suas funcionalidades principais com o objetivo de dar mais flexibilidade e liberdade ao desenvolvedor

[9]http://www.khronos.org/opengles/

que o utiliza. Esta característica é alcançada com a disponibilização de modelos de cenas que podem ser utilizadas para a criação de novas cenas. No entanto, além destes modelos, o arcabouço fornece um modelo base de cena, que pode ser utilizado para a criação de novas cenas de acordo com necessidades específicas do desenvolvedor. No Código 4.7 é desmostrada a interface `SceneBase` com seus métodos e valores, que são utilizados pelo arcabouço para o devido controle da cena e precisam ser sobrescritos.

Código Fonte 4.7: Modelo base para a criação de novas cenas

```
1  public interface SceneBase {
2    public static final long INTERVAL_FAST = 1000 / 60;
3    public static final long INTERVAL_30fps = 1000 / 30;
4    public static final long INTERVAL_24fps = 1000 / 24;
5    public static final long INTERVAL_15fps = 1000 / 15;
6
7    public static final int STATE_IDLE = 0;
8    public static final int STATE_LOADING = 1;
9    public static final int STATE_PAUSED = 2;
10   public static final int STATE_RUNNING = 3;
11   public static final int STATE_EXIT = 4;
12
13   public static final int ADD_CAMERA = 0xA;
14   public static final int ADD_GPS = 0xB;
15   public static final int ADD_ACCELEROMETER = 0xC;
16   public static final int ADD_COMPASS = 0xD;
17   public static final int ADD_GYROSCOPE = 0xE;
18
19   public void init();
20   public void start();
21   public void stop();
22   public void pause();
23   public void update();
24   public void render();
25   public void dispose();
26   public int getState();
27
28   public void setState(int state);
29   public void setParams(int params);
```

```
30    public void setUpdateInterval(long interval);
31 }
```

4.3 Conclusão

Neste capítulo foi apresentada uma visão geral do Arcabouço para o Desenvolvimento de Aplicações de Realidade Aumentada para Dispositivos Portáteis com o uso de Múltiplos Sensores, denominado mTKar. Inicialmente foi apresentada uma visão geral, através da qual pode-se ter uma idéia global do funcionamento como um todo. Em seguida foi apresentada a Arquitetura e o detalhamento de seus módulos: *Módulo de Aquisição*, *Módulo de Tratamento*, *Módulo de Montagem* e o *Módulo de Projeção*. E para concluir, foi apresentada a interface `SceneBase`, que pode ser utilizada para a criação de cenas com necessidades específicas do desenvolvedor. A prova de conceito e validação do arcabouço foram realizadas por meio de estudos de caso, um para cada tipo de cena, os quais são discutidos no **Capítulo 5**.

Capítulo 5

Estudo de caso

Este capítulo apresenta o desenvolvimento de três estudos de caso para demostrar o uso do arcabouço na criação de aplicações de RA. Os três projetos desenvolvidos utilizam os recursos oferecidos pelo arcabouço e presentes nos dispositivos. O capítulo apresenta uma descrição destes três aplicativos, os tipos de sensores utilizados e a forma como interagem com o arcabouço.

Os protótipos foram desenvolvidos para a plataforma Android[1] utilizando o Android SDK[2] e testados em um *smartphone* Nexus S[3] e em um *tablet* Galaxy Tab 2[4] de 7".

5.1 NavegAR

O **NavegAR** (Figura 5.1) é um navegador de RA, no qual o usuário poderá consultar os restaurantes (pontos de interesse/POIs) ao seu redor. O aplicativo tem funcionamento similar ao *Nokia City Lens* [5], no qual o usuário ao apontar a câmera do dispositivo para um determinado local, os POIs, neste caso os restaurantes, são apresentados contendo nome do estabelecimento e a distância relativa à posição do usuário em metros, esta posição é adquirida através do GPS presente no aparelho do usuário.

O NavegAR foi desenvenvolvido utilizando o **Modelo 1: Navegador de AR**, fornecido

[1] http://www.android.com/

[2] http://developer.android.com/sdk/index.html

[3] http://www.android.com/devices/detail/nexus-s

[4] www.samsung.com/global/microsite/galaxytab2/7.0/index.html

[5] http://betalabs.nokia.com/apps/nokia-city-lens

Figura 5.1: NavegAR: Navegador de RA, faz uso da câmera, GPS e sensores inerciais

pelo arcabouço. Neste modelo, os sensores envolvidos são:

- **GPS** responsável por fornecer o posicionamento global do usuário (sua localização) através da latitude e longitude;
- **Sensores inerciais**: Acelerômetro, Magnetômetro e Girosópio; responsáveis por fornecer a orientação do usuário nos eixos x, y, z;
- **Câmera:** captura a imagem do mundo real para que seja utilizado na composição final da cena, age como uma janela ligando o mundo real ao virtual.

Abaixo, Código Fonte 5.1, é apresentado a forma como o modelo é utilizado:

Código Fonte 5.1: Código base do aplicativo NavegAR

```
1 import mtkar.core.SceneModelA;
2 ...
3 class NavegAR extends SceneModelA {
4   // métodos padrão para todos os modelos de cena
5   // estes métodos precisam ser implementados
6   public void init(){ ... }
7   public void pause(){ ... }
8   public void update(){ ... }
9   public void render(){ ... }
```

```
10    public void dispose(){ ... }
11
12    // métodos específicos
13    public Point2D getLocation() { ... }
14    public void setDistanceUpdate(int meters){ ... }
15    public void setData(JSONObject data){ ... }
16    public void setStyle(int bgColor, int titleColor, int textColor}{ ... }
17  }
```

Para fonecer os pontos de interesse, um *webservice* foi desenvolvido em que a comunicação com o mesmo é feita através de requisições HTTP[6] do tipo GET e com o retorno dos dados em formato JSON[7]. Assim, para solicitar os POIs de um determinado ponto, o aplicativo envia uma solicitação GET ao *webservice* com as coordenadas GPS de sua localização, juntamente com o raio máximo, em metros, desejado:

Código Fonte 5.2: String GET com dados da requisição

```
?lat=-7.216217&lon=-35.904541&radius=1000
```

Como resposta, o *webservice* devolve um objeto em formato JSON com os dados dos POIs encontrados:

Código Fonte 5.3: Objeto JSON com os dados de retorno

```
{
  "hotspots" [
    {
      "id":"10",
      "title":"Restaurante X",
      "lat":-7216303.0115,
      "lon":-35904508.6919,
      "distance":10.205826196472
    },
    {
      "id":"9",
      "title":"Restaurante Y",
      "lat":-7216119.4479,
```

[6] http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol
[7] http://www.json.org/

```
        "lon":-35905426.4261,
        "distance":98.270869024874
      },
      {
      ...
      }
   ],

   "errorCode":0,
   "errorString":"ok"
}
```

Existem vários navegadores de RA disponíveis no mercado e a tendência é só aumentar, nas Figuras 5.2 e 5.3 podem ser vistos dois exemplos desses aplicativos.

Figura 5.2: Navegador de RA Nokia *City Lens* [24]

5.2 Visualizador de Objetos

O **Visualizador de Objetos** (Figura 5.4) é um aplicativo simples de RA, baseado em marcador. O aplicativo tem funcionamento simples, onde o usuário aponta a câmera do dispositivo

Figura 5.3: Navegador de RA Layar [15]

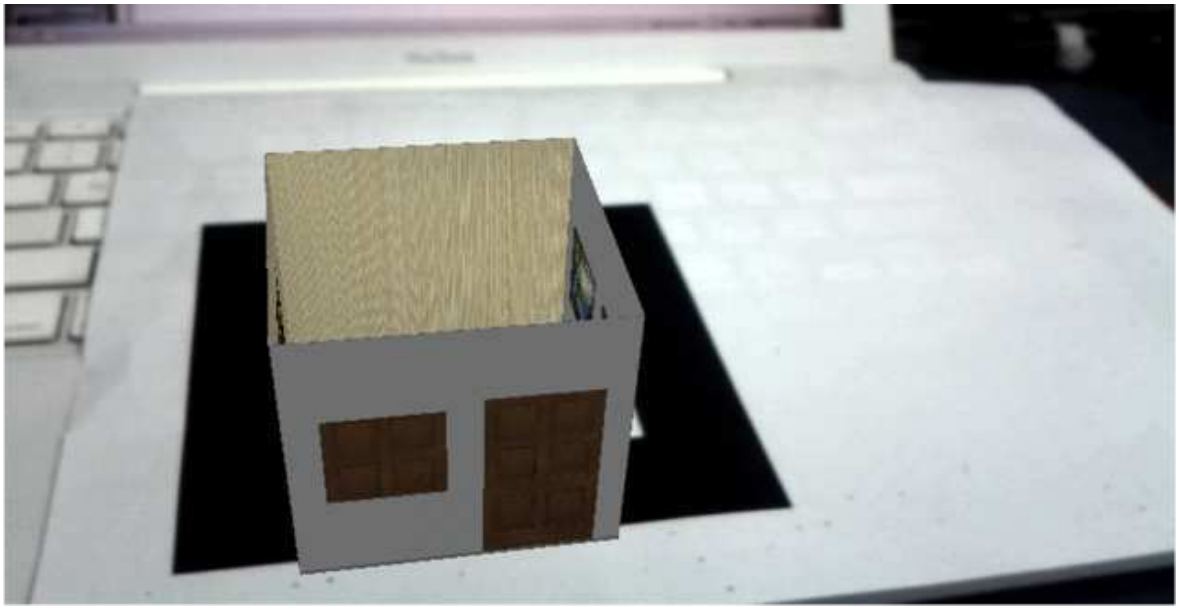


Figura 5.4: Visualizador de Objetos

para o marcador e um objeto 3D é projetado sobre este.

O visualizador foi desenvenvolvido utilizando o **Modelo 2: Visualizador de Objetos**, fornecido pelo arcabouço. Neste modelo, o único sensor envolvido é a câmera, a qual captura a imagem do marcador e envia para o arcabouço, o qual faz todo o processamento e devolve

a cena montada, com a projeção do objeto virtual sobre o marcador.

Abaixo, Código Fonte 5.4, é apresentado a forma com o modelo é utilizado:

Código Fonte 5.4: Código base do Visualizador de Objetos

```
1  import mtkar.core.SceneModelB;
2  ...
3  class ObjectView extends SceneModelB {
4    // métodos padrão para todos os modelos de cena
5    // estes métodos precisam ser implementados
6    public void init(){ ... }
7    public void pause(){ ... }
8    public void update(){ ... }
9    public void render(){ ... }
10   public void dispose(){ ... }
11
12   // métodos específicos
13   public void loadObject(Object3d obj){ ... }
14 }
```

Este tipo de aplicação é muito utilizada na área de jogos, publicidade em revistas, jornais, como pode ser visto na Figura 5.5, e como meio de *marketing* como visto anteriormente, no exemplo das caixas de brinquedos LEGO e na Figura 5.6.

5.3 Navegador Interno de Ambientes

O **Navegador Interno de Ambientes (NIA)** (Figura 5.7) é outro aplicativo simples de RA, porém este é baseado apenas nos sensores inerciais. O aplicativo tem funcionamento simples, após ser calibrado, o usuário poderá navegar internamente por um ambiente 3D, olhando nas diversas direções do mesmo, bastando apenas apontar o dispositivo para a direção desejada e a visão do mesmo será atualizada em tempo real. A tela do dispositivo comporta-se como uma janela para o mundo virtual, no qual o mesmo é projetado de acordo com o posicionamento do dispositivo e do usuário.

O visualizador foi desenvenvolvido utilizando o Modelo 3: Navegador Interno, fornecido pelo arcabouço. Neste modelo, os sensores inerciais são utilizados de forma a fornecer o posicionamento do usuário diante do ambiente virtual.

Figura 5.5: Realidade Aumentada em uma revista [14]

Figura 5.6: Aplicativo para smartphone baseado em marcador [47]

Abaixo, Código Fonte 5.5, é apresentado a forma como o modelo é utilizado:

Código Fonte 5.5: Código base do Navegador Interno de Ambientes

```
1 import mtkar.core.SceneModelC;
```

Figura 5.7: NIA - Navegador Interno de Ambientes

```
2  ...
3  class AmbientView extends SceneModelC {
4    // métodos padrão para todos os modelos de cena
5    // estes métodos precisam ser implementados
6    public void init(){ ... }
7    public void pause(){ ... }
8    public void update(){ ... }
9    public void render(){ ... }
10   public void dispose(){ ... }
11
12   // métodos específicos
13   public void loadObject(Object3d obj){ ... }
14   public void setPosition(Point3D pos){ ... }
15   public void resetPosition(){ ... }
16   public Point3D getPosition(){ ... }
17
18   // métodos utilizados para se deslocar na cena
19   public void walkToFront(int steps) { ... }
20   public void walkToBack(int steps) { ... }
21   public void walkToLeft(int steps) { ... }
22   public void walkToRight(int steps) { ... }
23 }
```

Este tipo de aplicativo é muito utilizado na área de visualização interna de ambientes (design de interiores, por exemplo), em que o usuário pode ter uma visão de como ficará, internamente, um apartamento, quarto, sala, etc, antes mesmo de sua construção e/ou reforma. Também poderá ser utilizado para o desenvolvimento de jogos, um exemplo é o *Space Impact: Meteor Shield*[8], que faz uso do magnetômetro e do acelerômentro presente no *smartphone*; a Figura 5.8 apresenta este aplicativo.

Figura 5.8: *Space Impact: Meteor Shield* [45]

5.4 Conclusão

Este capítulo mostrou com exemplos simples como explorar superficialmente o potencial do arcabouço. Aplicativos mais complexos podem ser desenvolvidos com base nos exemplos aqui apresentados tirando proveito de todo o potencial do arcabouço. Além disso, as cenas podem ser expandidas para as necessidades do desenvolvedor de forma simples e fácil, assim o aplicativo desenvolvido pode ter um melhor aproveitamento do dispositivo em uso, seja esse um *smartphone* ou *tablet* e garantir uma melhor experiência por parte do usuário.

[8] http://www.noknok.tv/2010/04/29/ovi-store-space-impact-meteor-shield-review/

Capítulo 6

Considerações Finais

Atualmente existe uma grande quantidade de trabalhos voltados para o desenvolvimento de aplicações de RA, alguns destes são focados em dispositivos portáteis, como pode ser observado em [43].

No entanto os poucos arcabouços encontrados são voltados apenas para a utilização em ambientes preparados, fazendo o uso da câmera do dispositivo juntamente com um marcador. Não são encontradas soluções em forma de arcabouço para o desenvolvimento de aplicações que façam o uso de múltiplos sensores, muito menos voltados para dispositivos portáteis.

Neste trabalho foi apresentado um arcabouço para o desenvolvimento de aplicações de realidade aumentada para dispositivos portáteis com o uso de múltiplos sensores. Este arcabouço foi elaborado tendo em mente o uso de modelos de cenas e, dessa forma, esconder do desenvolvedor a complexidade envolvida por trás desse tipo de aplicação. Foram criados três modelos de cenas, os quais permitem ao desenvolvedor criar uma boa variedade de aplicações para diversas áreas na qual a RA está envolvida como entretenimento, educação, games e publicidade.

Três estudos de caso foram desenvolvidos para demostrar a utilização dos modelos de cenas disponíveis pelo arcabouço. O primeiro estudo de caso, denominado de *NavegAR*, foi desenvolvido com base no modelo de cena que faz o uso do GPS, dos sensores inerciais e da câmera presentes no dispositivo, este aplicativo tem o objetivo de fornecer os pontos de interesse (POIs) ao redor do usuário, restaurantes, com o nome do estabelecimento e a distância relativa ao usuário. O segundo aplicativo desenvolvido foi um simples visualizador de objetos 3D, este faz o uso da cena baseada em câmera e marcador e o seu objetivo é o

de exibir um objeto 3D sobre o marcador. Por último, o terceiro aplicativo desenvolvido é voltado para a visualização interna de ambientes, este faz uso do modelo de cena baseado apenas em sensores, seu objetivo é exibir um ambiente interno, sala ou quarto, onde o usuário poderá olhar para os diversos cantos do mesmo.

Além das cenas fornecidas pelo arcabouço, um cena base também foi disponibilizada para que os desenvolvedores, caso o queiram, possam expandir o arcabouço adicionando novas cenas com um maior número de funcionalidades, porém mantendo a complexidade escondida do desenvolvedor.

Uma das limitações deste trabalho está no fato do arcabouço estar disponível apenas na plataforma Android. Como trabalho futuro, pode-se fazer o *porting* do mesmo para outras plataformas móveis como o iOS[1], da Apple, e o Windows Phone[2], da Microsoft.

Outro trabalho futuro é adicionar a possibilidade do usuário interagir diretamente com os objetos da cena, não apenas com as cenas, como acontece hoje, dessa forma novos modelos de cenas poderiam ser criados e disponibilizados para o desenvolver.

Atualmente o arcabouço permite o desenvolvimento de aplicações que fazem o uso de marcadores, no entanto existem algumas soluções no mercado voltadas exclusivamente para esse tipo de aplicação, como é o caso do *Vuforia* da Qualcomm[3]. Sendo assim um trabalho futuro, seria focar o arcabouço apenas nos sensores inerciais (giroscópio, acelerômetro, magnetômetro) e de posicionamento global (GPS), garantindo assim um melhor posicionamento diante dos demais arcabouços disponíveis no mercado.

[1] http://www.apple.com/ios/

[2] http://www.windowsphone.com

[3] https://www.vuforia.com/

Bibliografia

[1] Auto Evolution. Vw personnel trained in augmented reality. http://www.autoevolution.com/, acessado em agosto de 2013.

[2] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. Recent advances in augmented reality. *Computer Graphics and Applications, IEEE*, 2001.

[3] R. Azuma and G. Bishop. Improving static and dynamic registration in an optical see-through hmd. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, 1994.

[4] R. Azuma and Others. A survey of augmented reality. *Presence-Teleoperators and Virtual Environments*, 1997.

[5] Steve Babin. *Developing Software for Symbian OS 2nd Edition: A Beginner's Guide to Creating Symbian OS v9 Smartphone Applications in C++*. Symbian Press. John Wiley & Sons, 2008.

[6] R. Ballagas, G. Revelle, K. Buza, H. Horii, K. Mori, H. Raffle, M. Spasojevic, J. Go, K. Cook, E. Reardon, and Others. Electric Agents: combining television and mobile phones for an educational game. In *Proceedings of the 10th International Conference on Interaction Design and Children*, 2011.

[7] P. Castro, P. Chiu, T. Kremenek, and R. Muntz. A probabilistic room location service for wireless networked environments. In *UbiComp 2001: Ubiquitous Computing*, 2001.

[8] J. Chow, H. Feng, R. Amor, and B. Wünsche. Music Education using Augmented Reality with a Head Mounted Display. *Proceedings of the Fourteenth Australasian User Interface Conference*, 2013.

[9] J.A. Corrales Ramón. *Kalman filtering for sensor fusion in a human tracking system*. INTECH, 2010.

[10] K.K. Cox and T.H.R. Meneses. Realidade Aumentada na Alfabetização com o "Jogo das Letras". *Virtual Reality*, 2012.

[11] Ánalise das Mídias. Realidade aumentada nos esportes. http://analisedasmidias.wordpress.com, acessado em agosto de 2013.

[12] L.T. De Paolis and G. Aloisio. Augmented reality in minimally invasive surgery. In *Advances in Biomedical Sensing, Measurements, Instrumentation and Systems*. Springer, 2010.

[13] K.G. Derpanis. The Harris corner detector. *York University, Octorber*, 2004.

[14] Ler ebooks. Layar creator – realidade aumentada em livros e revistas. http://lerebooks.wordpress.com/, acessado em agosto de 2013.

[15] Educadictos. La realidad aumentada y su utilización en educación (layar). http://www.educadictos.com/, acessado em agosto de 2013.

[16] T.S.M.C. Farias. *A2: Um Modelo de Referência para Aplicações de Realidade Aumentada em Redes de Ambiente*. PhD thesis, Universidade Federal de Pernambuco, 2007.

[17] Google. Google glass. http://www.google.com/glass/, acessado em agosto de 2013.

[18] G. Hua, M. Brown, and S. Winder. Discriminant embedding for local image descriptors. In *Proceedings of the 11th International Conference on Computer Vision, IEEE*, 2007.

[19] C.E. Hughes, C.B. Stapleton, D.E. Hughes, and E.M. Smith. Mixed reality in education, entertainment, and training. *Computer Graphics and Applications, IEEE*, 2005.

[20] Hypeness. Realidade aumentada animal. http://www.hypeness.com.br/2011/11/realidade-aumentada-animal/, acessado em agosto de 2013.

[21] S. Julier, R. King, B. Colbert, J. Durbin, and L. Rosenblum. The software architecture of a real-time battlefield visualization virtual environment. In *Proceedings IEEE Virtual Reality*, 1999.

[22] C. Kirner and R. Siscoutto. *Realidade virtual e aumentada: conceitos, projeto e aplicações*. Editora SBC, 2007.

[23] Thommen Korah, Jason Wither, Y.T. Tsai, and Ronald Azuma. Mobile Augmented Reality at the Hollywood Walk of Fame. In *Proceedings IEEE Virtual Reality Conference (VR)*, 2011.

[24] Nokia Beta Labs. Nokia city lens. http://betalabs.nokia.com/apps/nokia-city-lens, acessado em agosto de 2013.

[25] P. Lawitzki. *Application of Dynamic Binaural Signals in Acoustic Games*. PhD thesis, Stuttgart Media University, 2012.

[26] P. Lawitzki. Android sensor fusion tutorial. http://www.thousand-thoughts.com/, acessado em julho de 2013.

[27] A.J.R. Lima, G.G. Cunha, and C.J. Haguenauer. Realidade aumentada no ensino de geometria descritiva. *Realidade Virtual*, 2008.

[28] A. Lutfi and A.B. Raposo. Realidade Aumentada e Publicidade: Até onde pode ir essa relação? http://www.tecgraf.puc-rio.br/, acessado em julho de 2013.

[29] R. Luz, M.W.S. Ribeiro, A. Cardoso, E. Lamounier Jr, H. Rocha, and W. Silva. Análise de aplicações de realidade aumentada na educação profissional: Um estudo de caso no senai DR/GO. In *5º Workshop de Realidade Virtual e Aumentada*, 2008.

[30] Orlando Attactions Magazine. Cool 3d animated digital box at the lego store. http://attractionsmagazine.com/blog/2009/03/27/cool-3d-animated-digital-box-at-the-lego-store/, acessado em julho de 2013.

[31] M. Mohring, C. Lessig, and Ol. Bimber. Video see-through ar on consumer cellphones. In *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2004.

[32] A. Morrison, A. Oulasvirta, P. Peltonen, S. Lemmela, G. Jacucci, G. Reitmayr, J. Näsänen, and A. Juustila. Like bees around the hive: a comparative study of a mobile augmented reality map. In *Proceedings of the 27th international conference on Human factors in computing systems*, 2009.

[33] D.J. Murphy, M. Kahari, and V.V. Mattila. An augmented reality view on mirror world content, with Image Space. In *Proceedings IEEE Virtual Reality Conference (VR)*, 2010.

[34] W. Narzt, G. Pomberger, A. Ferscha, D. Kolb, R. Muller, J. Wieghardt, H. Hortner, and C. Lindinger. Pervasive information acquisition for mobile AR-navigation systems. In *Proceedings of the 5th IEEE Workshop on Mobile Computing Systems and Applications*, 2003.

[35] J.M.L.B. Pereira. A realidade aumentada na engenharia biomédica: Estado da arte. http://ltodi.est.ips.pt/jbraz/, acessado em agosto de 2013.

[36] S.A. Pessoa. *Um Pipeline para Renderização Fotorrealística em Aplicações de Realidade Aumentada*. PhD thesis, Universidade Federal de Pernambuco, 2010.

[37] Military Photos. Helmet mounted display system. http://www.militaryphotos.net/, acessado em agosto de 2013.

[38] W. Piekarski. Sony glasstron hmd. http://www.tinmith.net/wayne/thesis/, acessado em agosto de 2013.

[39] A.B. Raposo and F. Szenberg. Visão estereoscópica, realidade virtual, realidade aumentada e colaboração. *Anais do XXIV Congresso da Sociedade Brasileira de Computação*, 2005.

[40] D. Rebolj and K. Menzel. Mobile computing in construction. *Proceedings of the 8th ISPE International Conference on Concurrent Engineering: Research and Applications (CE 2001)*, 2001.

[41] S.M.V. Romano. Realidade Aumentada Aplicada a Medicina. *Revista Científica FATEF - Revista Automação e Tecnologia da Informação Aplicada à Ciência*, 2011.

[42] G. Schall, D. Wagner, G. Reitmayr, E. Taichmann, M. Wieser, D. Schmalstieg, and B. Hofmann-Wellenhof. Global pose estimation using multi-sensor fusion for outdoor augmented reality. In *Proceedings of the 8th IEEE International Symposium on Mixed and Augmented Reality*, 2009.

[43] S. Siltanen. *Theory and applications of marker-based augmented reality*. VTT Science, 2012.

[44] A. State, Kurtis P. Keller, and H. Fuchs. Simulation-based design and rapid prototyping of a parallax-free, orthoscopic video see-through head-mounted display. In *Proceedings of the 4th IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2005.

[45] All About Symbian. Space impact: Meteor shield. http://www.allaboutsymbian.com/, acessado em julho de 2013.

[46] H. Tamura. Steady steps and giant leap toward practical mixed reality systems and applications. In *Proceeding of the International Status Conference on Virtual and Augmented Reality*, 2002.

[47] UOL Tecnologia. Feira de tecnologia ces 2012. http://tecnologia.uol.com.br/, acessado em agosto de 2013.

[48] D.W.F. Van Krevelen and R. Poelman. A survey of augmented reality technologies, applications and limitations. *International Journal on Virtual Reality*, 2010.

[49] G. Welch and G. Bishop. An introduction to the Kalman filter. 1995.

[50] S. Winder and M. Brown. Learning local image descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[51] S. Winder, G. Hua, and M. Brown. Picking the best daisy. In *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[52] S. You and U. Neumann. Fusion of vision and gyro tracking for robust augmented reality registration. In *Proceedings of the IEEE Virtual Reality*, 2001.

[53] S. Yuen, G. Yaoyuneyong, and E. Johnson. Augmented Reality and Education: Applications and Potentials. In Huang, Ronghuai and Spector, J. Michael, editor, *Reshaping Learning*. Springer Berlin Heidelberg, 2013.

[54] F. Zhou, H.B.L. Duh, and M. Billinghurst. Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2008.