Um Método de Calibração Automática para uma Configuração de Múltiplos Marcadores com Oclusão Total

Douglas Coelho Braga de Oliveira

JUIZ DE FORA

MARÇO, 2016

Um Método de Calibração Automática para uma Configuração de Múltiplos Marcadores com Oclusão Total

Douglas Coelho Braga de Oliveira

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Ciência da Computação

Orientador: Rodrigo Luis de Souza da Silva

JUIZ DE FORA

MARÇO, 2016

Um Método de Calibração Automática para uma Configuração de Múltiplos Marcadores com Oclusão Total

Douglas Coelho Braga de Oliveira

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Rodrigo Luis de Souza da Silva
Doutor em Engenharia

Marcelo Caniato Renhe
Mestre em Engenharias de Sistemas e Computação

Rafael Alves Bonfim de Queiroz
Doutor em Modelagem Computacional

JUIZ DE FORA

01 DE MARÇO, 2016

À minha avó.

Resumo

Este trabalho propõe um novo método para automatizar a calibração de múltiplos marcadores fiduciais distribuídos em um sistema de Realidade Aumentada, com objetivo de obter relações estáveis, principalmente entre marcadores que nunca estarão visíveis em uma mesma imagem do vídeo. A principal característica do método é preservar todas as relações entre os marcadores em cada quadro do vídeo. Estas relações serão atualizadas durante a calibração de modo a obter um resultado preciso. Durante a execução, o marcador referencial (base) do sistema pode ser alterado quando aparecerem marcadores com maior qualidade. A aplicação gera como saída o arquivo de configuração de um ambiente de realidade aumentada com múltiplos marcadores no mesmo formato utilizado pelas bibliotecas AVRLib e ARToolKit.

Palavras-chave: Calibração Automática, Configuração Multimarcador, Realidade Aumentada.

Abstract

This work proposes a new method to automate the calibration of multiple fiducial markers distributed over an Augmented Reality scene, in order to obtain stable relations, mainly among markers that will never be visible on the same video frame. The main feature of the method is to maintain all relationships between markers in each video frame. These relationships will be updated during calibration in order to obtain an accurate result. During the execution, the referential (base) marker of the system may change when markers with greater quality appears in the scene. The application produces the output configuration file of a multiple marker augmented reality environment in the same format used by AVRLib and ARToolKit libraries.

Keywords: Automatic Calibration, Multimarker Setup, Augmented Reality.

Agradecimentos

Agradeço, primeiramente, aos meus pais, Elza e Pedro, por todo o apoio, sustento e confiança em mim depositada;

Agradeço também aos meus tios e primos que me ajudaram à me instalar na cidade, principalmente, à minha tia Edir que cedeu espaço em sua casa para eu morar durante certo tempo;

Agradeço ainda à todos os professores que me trouxeram até aqui por meio de seus ensinamentos, aos professores do GETComp por todo o apoio nos projetos que desenvolvi no grupo e, em especial, ao Prof. Rodrigo Luis pelos mais de três anos de orientação.

Também deixo uma agradecimento especial à Universidade Federal de Juiz de Fora e suas políticas de assistência estudantil, que foram, sem dúvidas, muito importantes para minha manutenção e concentração nos estudos.

Por fim agradeço aos colegas de curso, à turma do GETComp e do GCG pelo companheirismo e cooperação durante estes anos, sobretudo, ao Igor Couto por ainda fornecer o modelo 3D utilizado neste trabalho e demais assistências.

"Science is a way to keep from fooling ourselves and each other".

Neil deGrasse Tyson

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

API	Application Programming Interface
CG	Computação Gráfica
GLUT	openGL Utility Toolkit
OpenGL	Opensource Graphics Library
RA	Realidade Aumentada
RM	Realidade Mista
RV	Realidade Virtual
SIGGRAPH	Special Interest Group on GRAPHics and interactive techniques
VA	Virtualidade Aumentada
VST-HMD	Video See-Through Head-Mounted Display

1 Introdução

Em aplicações de Realidade Aumentada (RA), que utilizam rastreamento óptico para encontrar padrões em uma cena, é comum o uso de um cenário com múltiplos marcadores calibrados, i.e., um sistema em que as transformações rígidas entre todos os marcadores que o compõe são previamente conhecidas. Tais sistemas oferecem algumas vantagens sobre os que utilizam um único marcador sem abrir mão do desempenho ao utilizar marcadores fiduciais. Diferente dos marcadores naturais baseados em extração de características, os marcadores fiduciais possuem características conhecidas, tornando os processos de registro e rastreamento rápidos.

A principal vantagem do sistema de múltiplos marcadores fiduciais em relação à marcadores singulares é quanto a visibilidade do sistema. Problemas como oclusão do marcador ou uma má iluminação em um sistema de marcadores singulares podem ocasionar falhas na projeção dos objetos virtuais. Em um sistema de múltiplos marcadores calibrados, desde que ao menos um deles esteja visível em cena, a projeção dos objetos virtuais pode ocorrer adequadamente. Entretanto para obter este resultado é necessário fazer a calibração dos marcadores.

"Para a calibração, é necessário definir um plano no espaço e calcular, para cada marcador utilizado, qual a translação e a rotação em $\mathbb{R}^3$ que este marcador está em relação à origem do plano definido"(Caetano et al., 2013). Normalmente é escolhido um dos próprios marcadores do sistema como o referencial, descartando assim a necessidade de se calcular a transformação para um marcador, como sugerido por Wang et al. (2010). Usualmente este processo é feito de forma manual, sendo necessário o uso de instrumentos para medição de ângulos e distâncias entre os marcadores. Dependendo da complexidade do cenário, a obtenção das relações entre os marcadores que compõem a cena pode ser inviável, tanto pelo trabalho exaustivo quanto pela necessidade de um nível de precisão elevado, que nem sempre é possível obter através de ferramentas de medição. Sendo assim, como será abordado na seção 2.4, diversos métodos para calibração automática de múltiplos marcadores foram propostos nos últimos anos.

1.1 Problema Abordado

A obtenção da relação entre os marcadores é bastante suscetível a erros. Existem vários problemas que fornecem erros de precisão indesejáveis, a começar pela própria precisão da máquina, que é limitada, passando por possíveis problemas no sistema de captura como oclusões temporárias, iluminação irregular na cena, efeitos de reflexões do marcador e etc. Outros problemas estão relacionados à correta captura da profundidade do marcador, os ângulos de inclinação e de rotação. Movimentações na câmera e perda de foco também podem causar erros de precisão no momento da calibração.

O problema se agrava quando tem-se oclusão total entre um par de marcadores, i.e., quando os marcadores estão distribuídos de modo que nem todos podem estar visíveis ao mesmo tempo, como quando distribuídos sobre uma base em formato cúbico ou mesmo sobre toda uma sala. Quando isto ocorre não é possível estimar a transformação de forma direta. Então é necessário encontrar outros meios de obter a relação entre pares de marcadores jamais vistos juntos em uma mesma cena.

1.2 Justificativa

Uma configuração onde todos os marcadores se encontram no mesmo plano, ou no mesmo volume de visualização, nem sempre é a ideal em aplicações de Realidade Aumentada mais realistas. Deseja-se que a câmera possa se movimentar livremente pelo cenário. Assim, uma configuração com marcadores em diversos ângulos e posições aproveita melhor o espaço disponível e as características do ambiente e, se bem calibrados, conseguem manter a mesma projeção preservando o ponto de vista da câmera.

1.3 Hipóteses

As transformações rígidas em um sistema de RA são representadas por matrizes, sendo possível acumular transformações através de uma simples multiplicação matricial. Partindo deste princípio, este trabalho assume ser possível, em algum momento da execução, estimar a relação entre dois marcadores que em nenhum momento foram vistos juntos na

mesma cena, desde que haja um terceiro marcador intermediário presente, em diferentes cenas, com ambos os marcadores que se deseja calcular a relação.

1.4 Objetivos

Este trabalho tem por objetivo desenvolver um método capaz de calcular as relações entre todos os pares de marcadores de uma determinada configuração de entrada. O método deve ser capaz de estimar a posição e orientação de um marcador em relação ao outro, mesmo que esses dois marcadores não estejam presentes no mesmo *frame* de vídeo durante toda a execução do aplicativo de calibração. As relações resultantes da calibração devem ser estáveis o suficiente para que, ao serem usadas em uma aplicação de RA, não deformem ou desalinhem consideravelmente os objetos virtuais projetados.

O trabalho também tem como objetivo definir regras para que, em tempo de execução, escolha o melhor referencial para o sistema, de modo que esta escolha propague um menor erro de precisão para a calibração. Ainda, quer-se comparar tais regras em diferentes cenários para analisar em que situação melhor se enquadra determinada regra. Outra finalidade é estabelecer um critério para aplicar o método que estimará a relação entre os pares com oclusão total durante a calibração.

1.5 Estrutura do Trabalho

O presente trabalho está dividido em cinco capítulos, incluindo a introdução. O capítulo seguinte contém a fundamentação teórica, esta apresenta a área de Realidade Aumentada, evidencia as ferramentas utilizadas para o desenvolvimento do trabalho, descreve os conceitos matemáticos necessários e explana sobre alguns dos trabalhos já realizados que se relacionam à este. No capítulo 3 é detalhado o método proposto para tratar o problema abordado na seção 1.1. O capítulo 4 traz os resultados obtidos pelo método proposto, descrevendo a métrica de erro adotada e discutindo a respeito dos dados levantados. Por fim, o capítulo 5 conclui o trabalho e apresenta potenciais propostas para futuros projetos.

2 Fundamentação Teórica

Neste capítulo serão introduzidos todos os conceitos e fundamentos que sustentam o presente trabalho. Inicialmente será apresentada a área de Realidade Aumentada, no qual este trabalho está inserido, trazendo sua definição e exemplos de sua aplicabilidade. Na sequência são apontadas diversas bibliotecas para o desenvolvimento na área, detalhando o funcionamento da biblioteca escolhida para a construção do método proposto, além de apresentar a motivação para tal escolha. O capítulo ainda descreve toda base matemática necessária ao entendimento do método que o trabalho propõe. Por fim, serão expostos alguns trabalhos que, de alguma forma, se relacionam com o aqui desenvolvido, apresentando um breve resumo e discutindo seus pontos fortes e limitações.

2.1 Realidade Aumentada

Milgram et al. (1994), em seu trabalho, relacionou as áreas de Realidade Aumentada (RA) e Realidade Virtual (RV), termos que estavam em crescente utilização na época, mas que careciam de uma definição formal. Nisso, ele ainda classificou a área de Virtualidade Aumentada (VA) em um mesmo eixo no qual chamou de *Reality-Virtuality Continuum*. De acordo com Milgram, as áreas de RA e VA fazem parte de uma área mais geral denominada Realidade Mista (RM), que trata de sistemas capazes de combinar as realidades física e virtual, gerando um novo ambiente onde objetos físicos e virtuais podem interagir entre si. De modo geral, sistemas de realidade aumentada adicionam elementos virtuais a um ambiente predominantemente real, enquanto que sistemas de virtualidade aumentada inserem elementos reais a um ambiente predominantemente virtual (Figura 2.1).

A VA, assim como a RV, tem como um dos seus principais conceitos a imersibilidade, i.e., o usuário deve sentir-se como parte do mundo ao qual foi inserido. No caso da VA, o mundo criado contém tanto elementos virtuais quanto reais, fazendo ainda com que o usuário sinta esses elementos como parte do mesmo mundo. Uma tecnologia que permite isso é a VST-HMD, um dispositivo composto por câmeras e telas à frente do olho

Figura 2.1: Ilustração do *Reality-Virtuality Continuum* proposto por Milgram et al. (1994). Classifica as áreas de RA e VA em um eixo que se estende desde o ambiente puramente real até o ambiente puramente virtual (realidade virtual).

que captura o ponto de vista do utilizador e exibe as imagens processadas. Em Beato et al. (2009) é proposta uma técnica de *Chroma key*[1] interativa para sistemas de RM.

A RA foi definida mais rigorosamente em Azuma (1997) como sendo um sistema capaz de integrar objetos virtuais ao ambiente físico tridimensional, de modo que é possível interagir com os objetos virtuais em tempo real. A aplicabilidade da área será discutida em subseção própria, onde serão apresentados alguns exemplos das diversas áreas que a RA abrange. O restante desta seção traz ainda as bibliotecas mais utilizadas no desenvolvimento de sistemas de RA.

2.1.1 Aplicações da Área

Arquitetura

A Realidade Virtual tem aplicação na arquitetura para auxiliar no planejamento de imóveis virtuais. Assim, o usuário é imerso, por exemplo, em um apartamento virtual e então pode visualizá-lo em detalhes, antes mesmo de ser iniciada a construção, ajudando

[1] Técnica de processamento de imagens que consiste em remover um fundo homogêneo, que contrasta com os objetos reais da cena, substituindo-o por um cenário virtual.

Figura 2.2: O maior projeto de Realidade Aumentada do mundo (Fonte: (Exame, 2010)).

na decisão da compra e podendo sugerir alterações no projeto. A Realidade Aumentada, por sua vez, é utilizada para projetar maquetes de futuros imóveis vistas em *shopping centers*, por exemplo. Em 2010, entretanto, a construtora Rossi resolveu inovar e criou o maior marcador fiducial do mundo, projetando um prédio em tamanho real sobre o local onde se ergueria (Figura 2.2). Tal feito foi registrado pelo *Guinness Book*[2].

Educação

A educação é, provavelmente, a área com maior gama de aplicações em Realidade Aumentada. São vários os trabalhos que desenvolvem ferramentas de apoio ao ensino. É também a área que mais dissemina a RA, pois abrange todas as áreas do conhecimento. Dentre os inúmeros trabalhos, cita-se alguns mais recentes, como os desenvolvidos em Gomez e Kirner (2015), Ibáñez et al. (2014), Macedo et al. (2012), Queiroz et al. (2014) e Silva et al. (2014). A diversificação das áreas de aplicação é bem exposta nestes exemplos, onde tem-se trabalhos aplicados à Medicina, Física, Química e Geografia. Há, até mesmo, aplicação ao ensino de Computação, como pode ser conferido no trabalho de Machado et al. (2013). Nesse trabalho foi desenvolvida uma ferramenta para apoiar o ensino de algoritmos de aprendizado de máquina, como o *k-means* e técnicas de regressão linear e polinomial. Na Figura 2.3 pode ser conferido o resultado da execução do *k-means* sobre

[2]www.guinnessworldrecords.com/world-records/largest-augmented-reality-mark [Online; acessado em 03/02/2016]

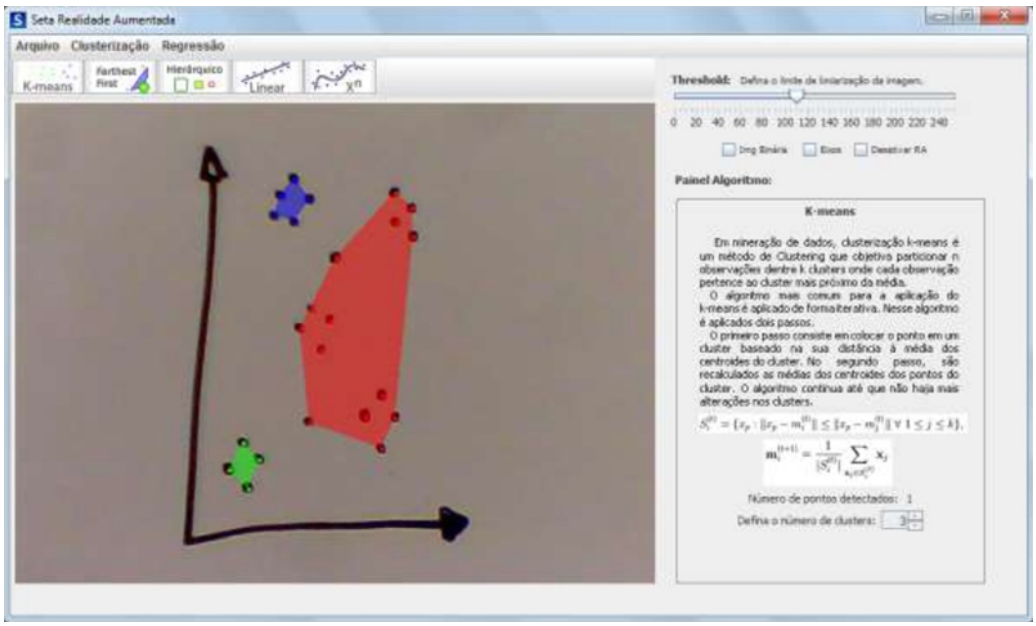


Figura 2.3: Tela da ferramenta SETA. Resultado da execução do algoritmo de clusterização *k-means* (Fonte: (Machado et al., 2013)).

um conjunto de pontos desenhados a mão pelo utilizador. O programa desenvolvido aplica algoritmos de Processamento de Imagens para remover ruídos da imagem e técnicas de Visão Computacional para obter a posição dos pontos e eixos.

Entretenimento

A indústria do entretenimento é outra área com imensa aplicabilidade à Realidade Aumentada. Vários jogos em RA podem ser encontrados nas lojas de aplicativos dos principais sistemas *mobile*. E para mostrar como esta área só tende a crescer nos próximos anos, recentemente a *Microsoft* apresentou uma versão do famoso jogo *Minecraft* em Realidade Aumentada (UOL, 2015). A apresentação foi para demonstrar o poder dos óculos inteligentes, o *HoloLens*, que está sendo desenvolvido pela companhia.

Psicologia

A Realidade Aumentada pode ajudar também na Psicologia, em especial no atendimento a pacientes com fobias. Juan et al. (2005) propõe em seu trabalho o uso da Realidade Aumentada para expor os pacientes a situações que lhes são traumáticas. Essa técnica é chamada dessensibilização sistemática e é utilizada na terapia cognitiva-comportamental. Por se tratar de objetos virtuais, os pacientes se sentem seguros a enfrentar seu maior

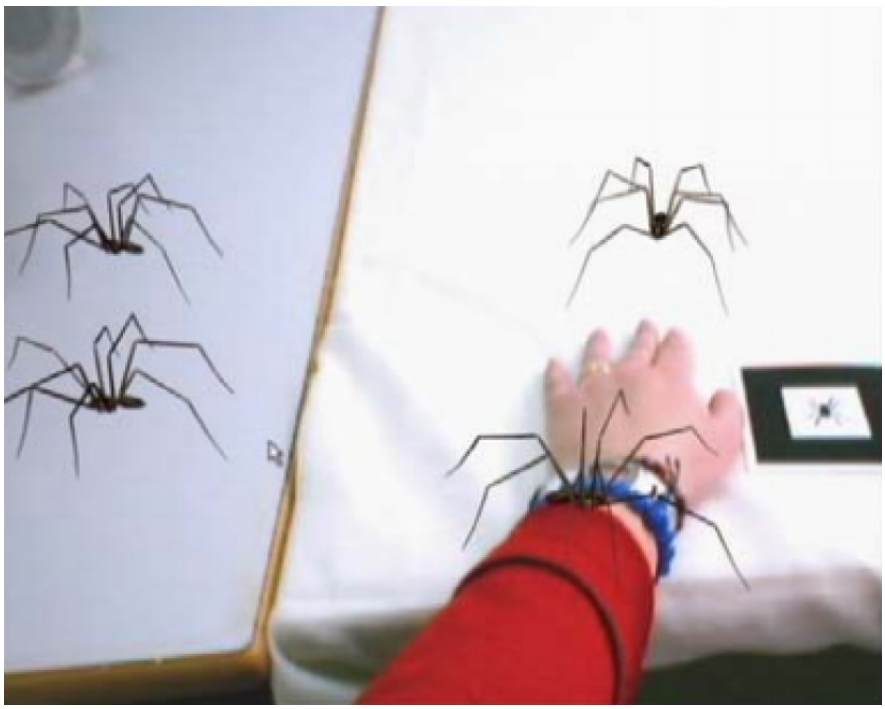


Figura 2.4: Tratamento de um paciente que sofre de aracnofobia utilizando Realidade Aumentada (Fonte: (Juan et al., 2005)).

medo, permitindo o contato com objetos que lhes causam pavor (Figura 2.4).

Publicidade

É normal a indústria da publicidade buscar métodos criativos e atrativos para chamar a atenção dos consumidores. Com a crescente popularização dos *smartphones*, que possuem cada vez mais poder de processamento, a Realidade Aumentada tem sido utilizada para promover produtos. Um exemplo é a promoção realizada para o lançamento do filme *Happy Feat Two* (Figura 2.5). Utilizando o aplicativo *Zappar*[3], ao apontar a câmera do aparelho para o cartaz do filme, a personagem "ganha vida" e começa a dançar pela tela do *smartphone*.

Uma ação promovida pela Axe em 2011 fez "anjos caírem" na estação Victoria em Londres. Uma grande estampa do novo desodorante da marca com a frase "*Look Up*" (olhe para cima) foi usada como marcador e, quando as pessoas seguiam a ordem, viam-se ao lado de anjos (Figura 2.6). Outra ação famosa foi a da *Doritos*, que colocou um marcador na embalagem de um de seus produtos. Os consumidores registravam o padrão no site da promoção e um "monstrinho" 3D virtual era projetado (Projectual, 2011).

[3]www.zappar.com [Online; acessado em 14/02/2016]

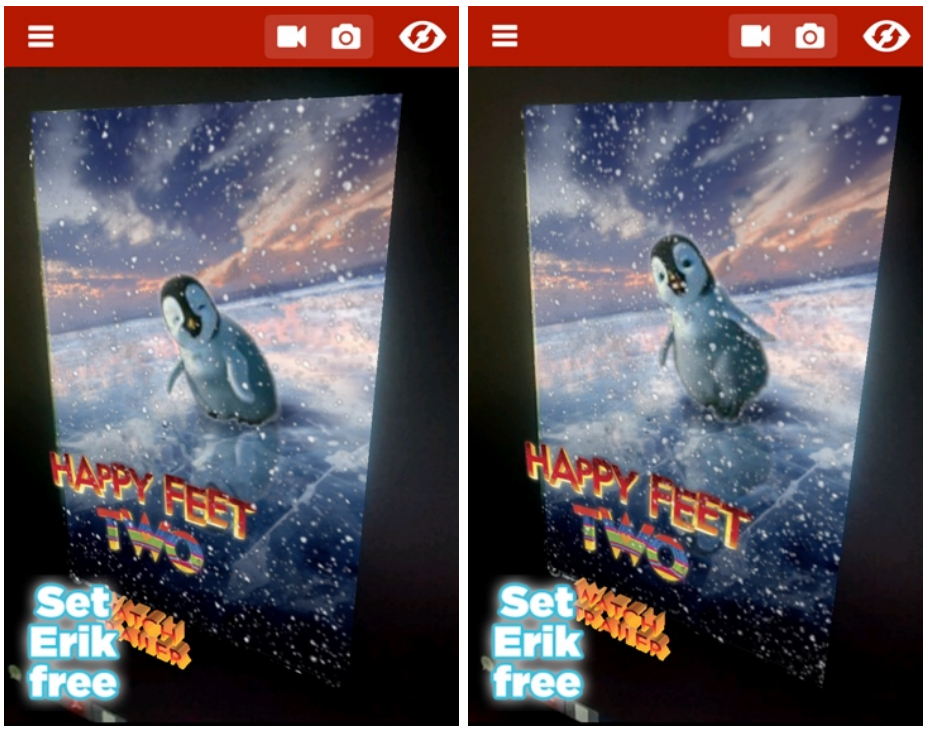


Figura 2.5: *Screenshots* da promoção realizada para o filme *Happy Feet Two* com o aplicativo de Realidade Aumentada *mobile Zappar*. Personagem dança pela tela do aparelho.

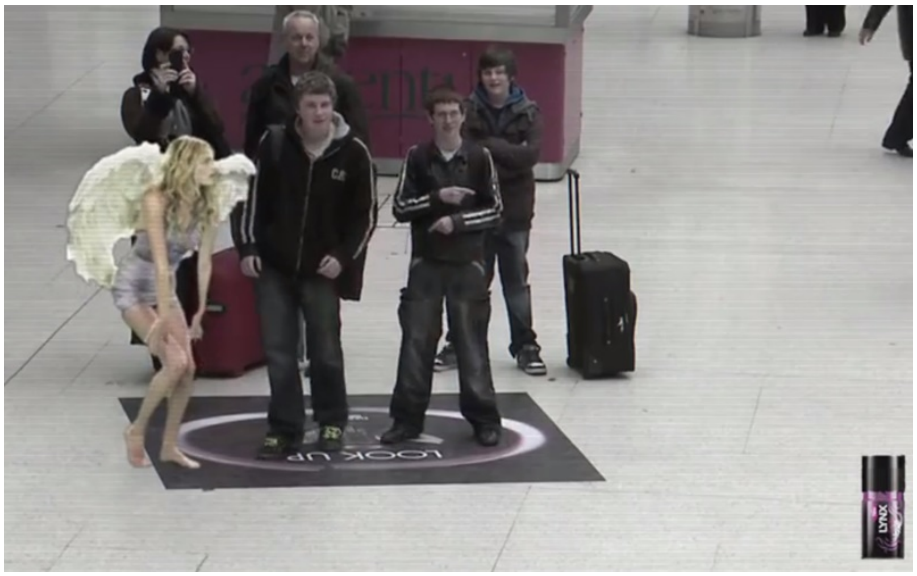


Figura 2.6: Promoção do desodorante *Axe Lynx Excite*, feita na estação Victoria em Londres (Fonte: www.youtube.com/watch?v=rFuUFeQIdpk [Online; acessado em 16/06/2015]).

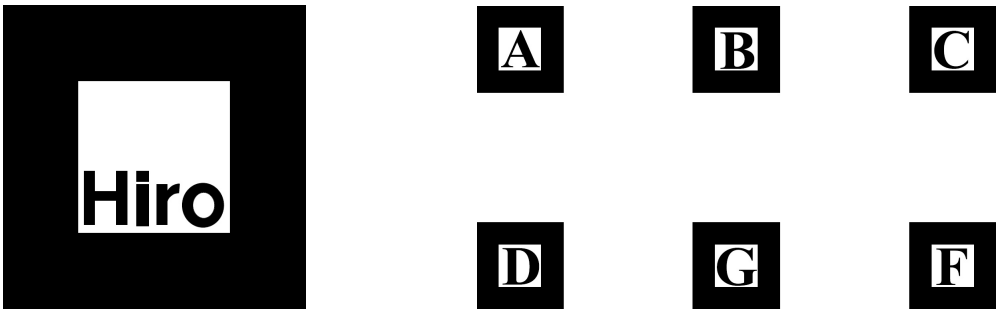
Enfim, a gama de opções que a RA disponibiliza tanto à publicidade quanto às demais áreas é imensa. Diversos outros exemplos de aplicações poderiam ser mencionados, mas esse não é o foco deste trabalho.

2.2 Bibliotecas

2.2.1 ARToolKit

Uma das principais bibliotecas de RA utilizadas atualmente foi apresentada na conferência SIGGRAPH [4] em 1999. Neste evento, os pesquisadores Hirokazu Kato e Mark Billinghurst apresentaram a biblioteca ARToollKit [5] como parte do projeto "*Shared Space*". A partir daí, várias aplicações e modificações desta biblioteca foram desenvolvidas e disponibilizadas para utilização pública. Versões comerciais desta biblioteca também foram desenvolvidas pela empresa ARToolworks [6].

O ARToolKit é uma biblioteca multiplataforma desenvolvida na linguagem C/C++ com o paradigma de programação estruturada. Ela contém algoritmos de visão computacional para calcular a posição e pose de marcadores fiduciais (Figura 2.7a) em relação à câmera real. Deste modo a biblioteca é capaz de definir o espaço do marcador na cena para então realizar a projeção do objeto virtual sobre ele. A biblioteca apresenta utilitários para criação de marcadores próprios e calibração da câmera. Ainda, fornece uma configuração simples de múltiplos marcadores (Figura 2.7b), além de algumas aplicações-exemplo mostrando como desenvolver na biblioteca.



(a) (b)

Figura 2.7: Exemplos de marcadores fiduciais da biblioteca ARToolKit: (a) Padrão *hiro*; (b) Configuração de múltiplos marcadores (Fonte: (Kato, 2005)).

Algumas das modificações e adaptações para outras linguagens do ARToolKit podem ser encontradas em Cunha e Fernandes (2010), Geiger et al. (2002), Haller et al. (2002) e em Reimann et al. (2003). A versão comercial do ARToolKit desenvolvida pela

[4] SIGGRAPH - Special Interest Group on GRAPHics and Interactive Techniques
[5] ARToolKit - www.hitl.washington.edu/artoolkit/ [Online; acessado em 25 de Maio de 2015]
[6] ARToolWorks - www.artoolworks.com/ [Online, acessado em 25 de Maio de 2015]

empresa ARToolWorks é encontrada em Wagner e Schmalstieg (2007). Outras bibliotecas também baseadas em marcadores fiduciais, mas que não são modificações da ARToolKit são encontradas em Fiala (2005) e em Jurado et al. (2014). A biblioteca que será utilizada no presente trabalho é uma outra modificação do ARToolKit e se chama AVRLib (Oliveira et al., 2013), que será detalhada na seção 2.2.2.

Sistema de Rastreamento

Para realizar a detecção dos marcadores na cena, o ARToolKit conta com seu próprio sistema de rastreamento óptico. O funcionamento básico de sistemas de RA baseados neste tipo de rastreamento está ilustrado na Figura 2.8. Como é mostrado, são necessários somente uma câmera, um visor (como monitores ou óculos inteligentes), o *software* de rastreamento e um objeto físico com características previamente definidas, neste caso, os marcadores fiduciais. A câmera captura cenas do mundo real, que então são processadas pelo *software* à procura das características conhecidas do objeto. Então, técnicas de visão computacional são aplicadas para calcular a posição e a pose do objeto na cena. Finalmente, realiza-se o processamento gráfico que dará origem aos objetos virtuais sobre a imagem do mundo real, projetando o resultado no visor.

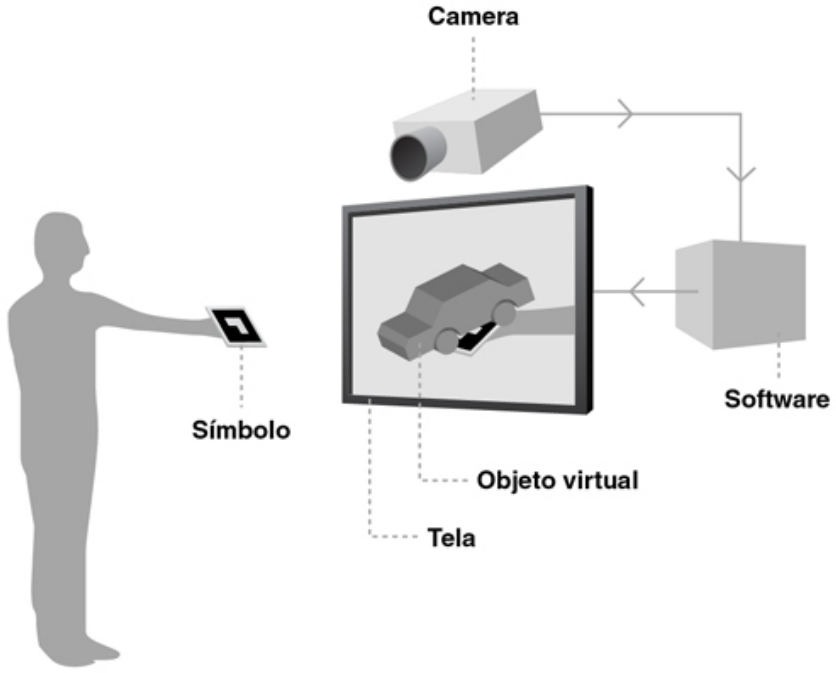


Figura 2.8: Esquema do funcionamento de um sistema de rastreamento óptico (Fonte: www.agenciadda.com.br/realidade-aumentada-ra [Online; acesso em 15/06/2015]).

Múltiplos Marcadores

O uso de cenários com múltiplos marcadores é importante quando se deseja que a câmera possa se movimentar livremente. O movimento da câmera leva a diferentes orientações do marcador e a diferentes condições na visualização. Em um determinado ângulo a luz incidente pode ser maior, por exemplo. Isso afeta o rastreamento e, por consequência, a projeção. Com múltiplos marcadores é possível explorar as características do cenário para que sempre haja um marcador em boas condições de visualização.

Ambientes multimarcadores são registrados no ARToolKit a partir de um arquivo como o da Figura 2.9. Este arquivo contém, para cada marcador do cenário, o caminho para o seu arquivo de registro, sua largura em milímetros, seu ponto central e sua matriz de transformação formada por submatrizes de rotação e translação.

```
Data/multi/patt.a           # Arquivo do marcador
40.0                        # Dimensão (mm)
0.0 0.0                     # Centro
1.0000 0.0000 0.0000 0.0000 # R11 R12 R13 Tx
0.0000 1.0000 0.0000 0.0000 # R21 R22 R23 Ty
0.0000 0.0000 1.0000 0.0000 # R31 R32 R33 Tz

Data/multi/patt.b
40.0
0.0 0.0
1.0000 0.0000 0.0000  100.0000
0.0000 1.0000 0.0000    0.0000
0.0000 0.0000 1.0000    0.0000
```

Figura 2.9: Parte do arquivo de configuração do padrão multimarcador do ARToolKit, mostrando as informações dos marcadores A (patt.a) e B (patt.b)

Observa-se na Figura 2.9 que o marcador A é a origem do sistema de coordenadas (referencial). Portanto, todos os demais marcadores são mapeados em relação a A. O centro do marcador B está localizado 100 unidades à direita de A e está na mesma altura e mesma profundidade. Seu plano é o mesmo do marcador A, pois não há rotação. Com essas informações torna-se possível manter a projeção do sistema, mesmo que haja vários marcadores não rastreados, apenas um deles precisa estar visível.

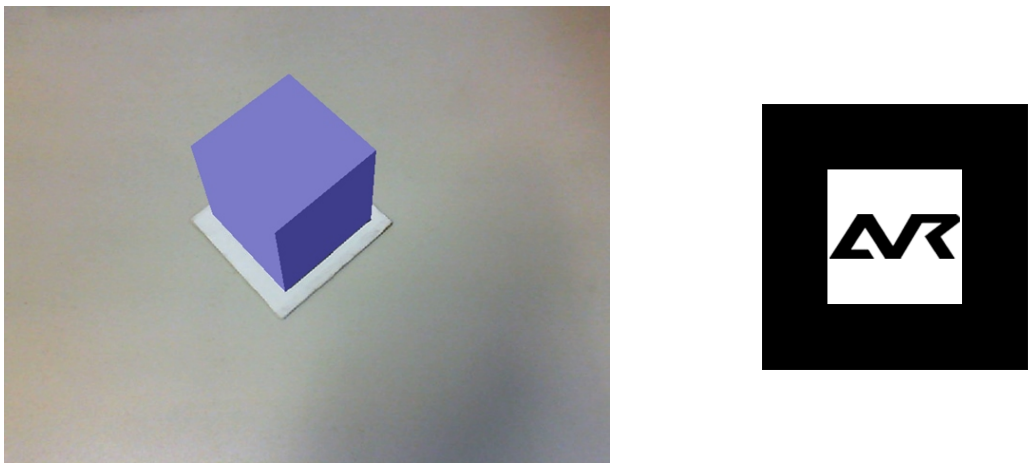
2.2.2 AVRLib

A AVRLib (*Augmented and Virtual Reality Library*) é uma biblioteca desenvolvida em C/C++ para criação de aplicações em Realidade Aumentada. Esta biblioteca faz uso dos recursos de Visão Computacional fornecidos pela biblioteca ARToolKit. A AVRLib apresenta uma interface de programação orientada a objetos de alto nível e usa a biblioteca OpenGL como API gráfica. A biblioteca é ainda *open source* e livre para fins acadêmicos e comerciais. Dentre os motivos que fizeram a biblioteca ser escolhida para o desenvolvimento deste trabalho destacam-se:

- Automatização de grande parte dos códigos necessários a toda aplicação de RA;
- Possibilidade de criação de sistemas de marcadores próprios, que são incorporados à biblioteca por meio da abordagem orientada a objetos.
- Familiaridade, pois a biblioteca foi desenvolvida na UFJF pelo grupo de realidade virtual e aumentada da instituição.

A AVRLib herda da biblioteca ARToolKit seu sistema de rastreamento óptico. Esta etapa está encapsulada na biblioteca e é executada automaticamente. Deste modo a escrita de aplicações simples é bastante simplificada, sendo necessário realizar apenas algumas configurações iniciais e escrever a função de renderização. O código abaixo é um exemplo reduzido de uma aplicação de RA bem simples desenvolvida com a AVRLib. Esta aplicação realiza a renderização de um cubo sobre o marcador AVR (Figura 2.10).

```
1  #include <avrApplication>
2
3  int main(int argc, char **argv) {
4     avrApplication* singleObjApp = new avrApplication();
5     // Arquivos de configuracao da camera
6     singleObjApp->setCameraFiles("Camera.xml", "CameraParams.dat");
7     // Registro do marcador e da callback de desenho
8     singleObjApp->addPattern("Data/avr.patt", 50.0, NULL, draw);
9     singleObjApp->setThreshold(100);
10    // main loop
11    singleObjApp->start();
12    return 0;
13 }
14
15 static void draw() {
16    glTranslatef( 0.0, 0.0, 25.0 );
17    glutSolidCube(50.0);
18 }
```



(a) (b)

Figura 2.10: (a) Exemplo da aplicação SingleObject (AVRLib) ou SimpleTest (ARToolKit) (b) Marcador utilizado para projeção (Fonte: (Oliveira et al., 2013)).

O exemplo do ARToolKit análogo a este, simplificado de modo a obter o mesmo resultado, ultrapassa facilmente 100 linhas de código. O alto nível que a AVRLib fornece poderia limitar o desenvolvimento de aplicações mais complexas. Entretanto, ela ainda permite ao desenvolvedor criar módulos para gerenciamento de marcadores e acoplá-los à biblioteca através da derivação da classe virtual *avrSystemMarker*. Junta-se a isso a possibilidade de utilização das funções do ARToolKit adaptadas para as classes auxiliares presentes na AVRLib. Assim é possível mudar o comportamento da biblioteca, aumentando a complexidade das aplicações, mas mantendo o alto nível de desenvolvimento.

Funcionamento

Na Figura 2.11 pode-se observar o fluxo de execução de aplicações de RA desenvolvidas com a AVRLib. O estado inicial, **Configuração**, é responsável por inicializar a câmera, registrar os marcadores e as *callback's*, que tratam os eventos de teclado e mouse, além da *callback* que realiza a renderização dos objetos. Esta etapa é realizada pelo desenvolvedor e corresponde a tudo que foi feito na função *main* do código-exemplo acima.

Os demais estados agora são realizados dentro do *loop* principal da biblioteca. O primeiro deles, **Rastreamento**, recebe um *frame* capturado pela câmera e faz a busca pelos marcadores registrados no estado anterior.

Definidos os marcadores visíveis na cena, técnicas de visão computacional computam, para cada um, a matriz de transformação que posiciona o marcador no sistema

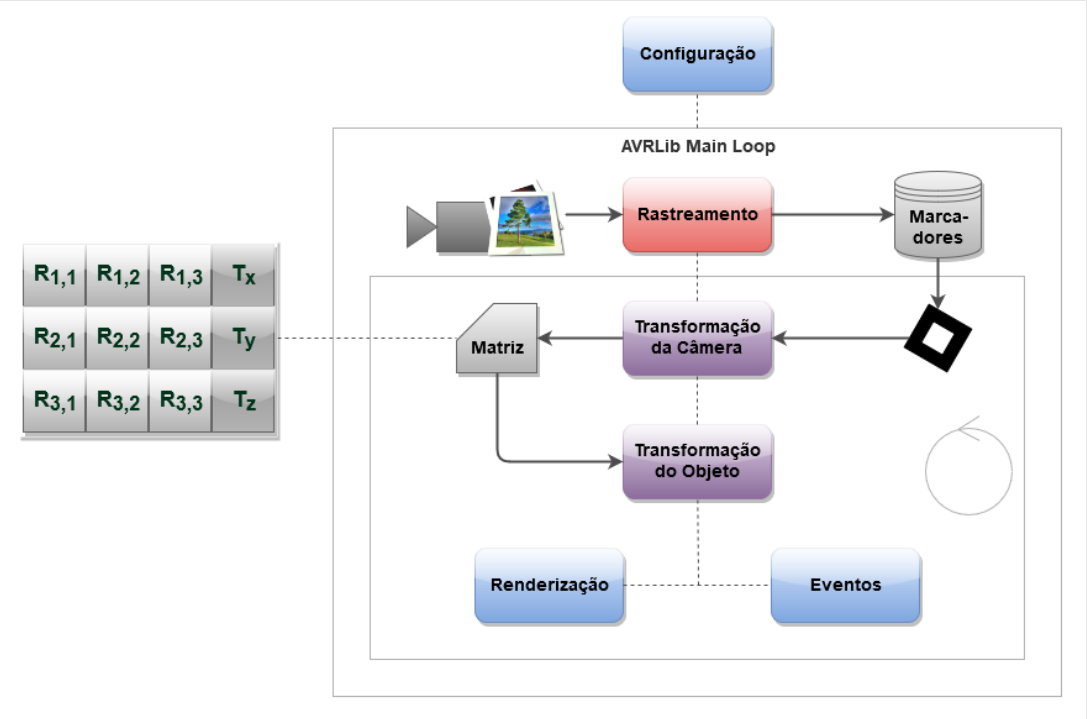


Figura 2.11: Fluxo de execução básico de aplicações que usam a AVRLib. Caixas em azul representam estados desenvolvidos pelo usuário. As vermelhas são internas à biblioteca e executadas automaticamente. As roxas também estão encapsuladas, mas pode-se modificá-las por meio da derivação da classe *avrSystemMarker*

de coordenadas da câmera. Essa fase é representada no estado **Transformação da Câmera**. A função que calcula a matriz chama-se *avrGetTransMat* e esta ainda retorna um valor real de acurácia da transformação. Esse valor será utilizado neste trabalho como parâmetro para definir a qualidade de um marcador na cena. A função também é encontrada sob o nome *avrGetTransMatCont*, porém essa leva em consideração também a matriz obtida para o *frame* anterior, produzindo acurácias melhores. Os sistemas de coordenadas da câmera e do marcador estão relacionados de acordo com a Figura 2.12.

O estado designado **Transformação do Objeto** realiza ajustes iniciais na API gráfica referentes ao posicionamento do objeto virtual e aos algoritmos necessários para a sua correta projeção na cena. O fluxo passa, então, à etapa de renderização e tratamento de eventos, ambos administrados pela biblioteca GLUT (*OpenGL Utility Toolkit*).

A seção 2.3 aborda os conceitos matemáticos envolvidos nas transformações, de modo a complementar o conhecimento a respeito do funcionamento da biblioteca e de sistemas de RA em geral.

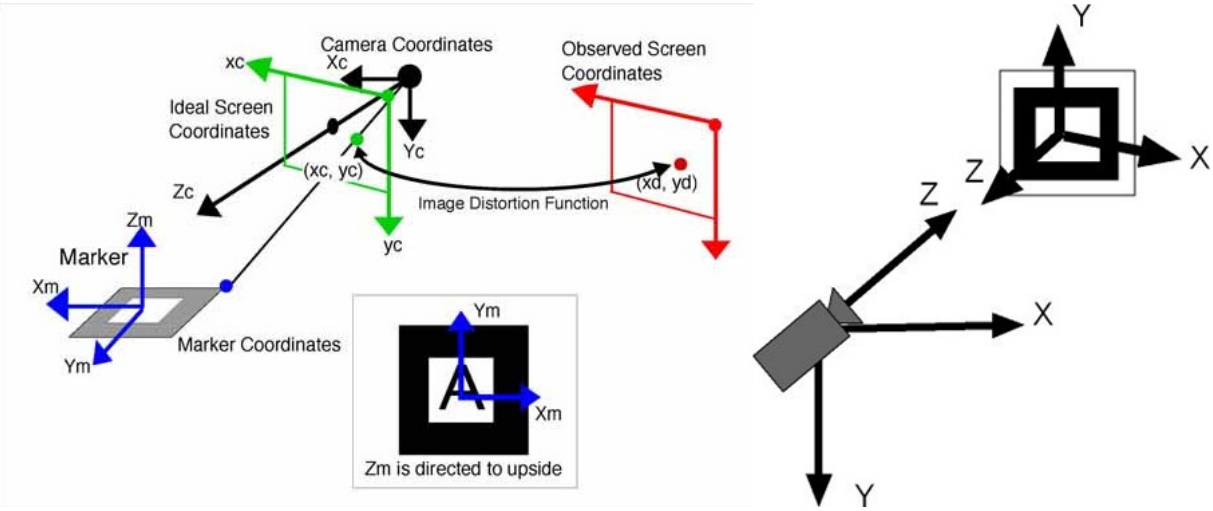


Figura 2.12: Sistemas de coordenadas. O eixo z da câmera está na mesma direção do eixo z do marcador, porém em sentido contrário (Fonte: (Kato, 2005)).

2.3 Fundamentos Algébricos

Esta seção contém toda base matemática necessária ao desenvolvimento deste trabalho. Serão abordados alguns tipos de transformações algébricas, suas representações matriciais, *quaternions* e propriedades dessas estruturas. A principal fonte consultada para construção desta seção foi o livro "*Mathematics for Computer Graphics*" (Vince, 2006), em especial o capítulo 7. Demonstrações das equações aqui apresentadas são encontradas nesse livro.

2.3.1 Definições

Em Álgebra uma transformação é uma função que mapeia vetores de um determinado espaço vetorial em outros vetores do mesmo espaço. Uma transformação é dita linear se preserva a soma de vetores e a multiplicação por um escalar, i.e., sejam $T : \mathbb{U} \rightarrow \mathbb{U}$ uma transformação, $\mathbf{a}, \mathbf{b}$ vetores no espaço $\mathbb{U}$ e α um escalar, se $T(\mathbf{a}) + T(\mathbf{b}) = T(\mathbf{a} + \mathbf{b})$ e $\alpha T(\mathbf{a}) = T(\alpha\mathbf{a})$ $\forall \mathbf{a}, \mathbf{b} \in \mathbb{U}$, então T é linear. Exemplos de transformações lineares incluem as operações de escala e de rotação, enquanto que a translação não é linear. Essas e outras transformações são bastante úteis na Computação Gráfica para a modificação de propriedades dos objetos virtuais, como posição, orientação, forma etc.

Uma terminologia usada em Computação Gráfica é o de Transformação Afim,

que é uma transformação linear seguida de uma translação, algebricamente definida como $T_{afim}(\mathbf{a}) = T(\mathbf{a}) + \mathbf{d}$, onde $\mathbf{d}$ é o vetor cujas componentes são os fatores de deslocamento. Outra nomenclatura empregada é o de Transformação Rígida, que consiste em transformações que não alteram a forma nem a dimensão do objeto transformado. Rotação e translação são alguns exemplos de transformações rígidas. A escala, como pode-se deduzir, não se encaixa nessa definição e, portanto, não é rígida.

2.3.2 Representação Matricial

Transformações podem ser representadas em forma de matrizes. Essa característica é muito útil, pois permite aplicar transformações eficientemente através de multiplicações matriciais, além de ser possível combinar várias transformações em uma única matriz. Para a translação, entretanto, não existe uma matriz da mesma dimensão do vetor, que possa aplicar a transformação através de uma multiplicação. Nesse caso a matriz deve ser somada ao vetor, o que impede de combiná-la com outras transformações em uma única matriz. É necessária uma dimensão extra para ser possível combinar a translação com outras transformações. Para isso, em vez de coordenadas cartesianas são usadas coordenadas homogêneas.

Um ponto $A \in \mathbb{R}^2$, cujas coordenadas cartesianas são $A(x,\ y)$, em coordenadas homogêneas é dado por $A_h(x_h,\ y_h,\ h), h \neq 0$ onde $x_h = x/h$ e $y_h = y/h$. Por conveniência define-se $h = 1$ e assim $A_h(x,\ y,\ 1)$. Para vetores a coordenada $h = 0$, pois são definidos pela subtração de dois pontos. Utilizando coordenadas homogêneas é possível aplicar operações de translação, escala e rotação combinadas, conforme mostrado abaixo. Como a multiplicação matricial não é uma operação comutativa, a ordem de aplicação das transformações importa. No exemplo a seguir, primeiro aplica-se a escala, em seguida a rotação e por fim a translação.

$$[translação] \times [rotação] \times [escala] \times [ponto]$$

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

A representação no espaço tridimensional é análoga à bidimensional, nesse caso as matrizes possuem dimensão 4x4. A rotação passa a ser em relação a um eixo em vez de um ponto, assim existem três tipos de rotação nesse espaço, uma para cada eixo cartesiano. Essas rotações recebem um nome específico, a rotação em torno do eixo z é chamada *roll* e as rotações em volta de Oy e Ox são chamadas, respectivamente, de *yaw* e *pitch*. Então, para rotacionar um objeto sobre um eixo específico é preciso encontrar os ângulos *roll*, *yaw* e *pitch* que, combinadas as respectivas rotações, produzem o efeito desejado. As matrizes dessas rotações, além de uma completa demonstração, podem ser conferidas em Vince (2006) entre as páginas 67 e 72.

Existem formas mais eficientes de representar rotações de objetos no espaço 3D, uma delas é através de uma estrutura denominada *Quaternion*. Um *quaternion* é uma quádrupla de números reais que podem ser usadas para definir um eixo arbitrário em $\mathbb{R}^3$ e um ângulo para a rotação. Assim é possível rotacionar o objeto em torno de qualquer eixo do $\mathbb{R}^3$ em uma única transformação por *quaternion*. Também é possível converter um *quaternion* em uma única matriz de rotação. Essa estrutura é utilizada pela AVRLib para a representação de rotações e está detalhada na subseção a seguir.

2.3.3 *Quaternions*

Buscando por uma maneira de generalizar os números complexos em 3 dimensões, após 15 anos de trabalho, o matemático Sir Willian Rowan Hamilton descobriu que uma estrutura quadridimensional, com certas propriedades, levava a um sistema fechado como o dos complexos no plano. A essa estrutura foi dado o nome *Quaternion*. Com isso Hamilton, não só conseguiu a generalização dos números complexos para o espaço tridimensional que buscara, como também deu origem ao ramo da matemática conhecido hoje como Álgebra Vetorial (Menon, 2009).

Após essa descoberta, formalizada por Hamilton em 1843, matemáticos têm mostrado que *quaternions* podem ser usados para rotacionar pontos em torno de eixos arbitrários e assim representar a orientação de objetos virtuais e da câmera (Vince, 2006).

Definição

Um *quaternion* é definido como $\mathbf{q} = [s, \mathbf{v}] = [s + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}]$, sendo s um escalar real e x, y, z as componentes reais do vetor $\mathbf{v}$ com 3 unidades imaginárias $\mathbf{i}, \mathbf{j}$ e $\mathbf{k}$ que possuem as seguintes propriedades:

- $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$
- $\mathbf{ij} = \mathbf{k}$, $\mathbf{jk} = \mathbf{i}$, $\mathbf{ki} = \mathbf{j}$
- $\mathbf{ji} = -\mathbf{k}$, $\mathbf{kj} = -\mathbf{i}$, $\mathbf{ik} = -\mathbf{j}$

A descoberta dessas propriedades foi o ponto chave do trabalho de Hamilton, o que assegurou a propriedade de fechamento ao *quaternion*.

Operações

Quaternions possuem operações de soma, multiplicação por escalar e multiplicação por outro *quaternion*. Também estão definidos o conjugado, a inversa e a norma. A maioria dessas operações são análogas às mesmas operações com vetores do $\mathbb{R}^4$, como a soma, multiplicação por escalar e a norma. O conjugado, assim como para os números complexos, é a negação da parte imaginária, nesse caso, um vetor tridimensional (Equação 2.1).

$$\overline{\mathbf{q}} = [s, -\mathbf{v}] = [s - x\mathbf{i} - y\mathbf{j} - z\mathbf{k}] \tag{2.1}$$

Uma operação importante para a aplicação de transformações e que não está definida para os vetores é a inversa. Ela é obtida por meio da divisão do conjugado pelo quadrado da norma euclidiana (Equação 2.2). Nota-se que para $\mathbf{q}$ unitário a inversa é igual ao conjugado.

$$\mathbf{q}^{-1} = \frac{[\, s, -\mathbf{v}\,]}{\parallel \mathbf{q} \parallel^2} = \frac{[\, s - x\mathbf{i} - y\mathbf{j} - z\mathbf{k}\,]}{s^2 + x^2 + y^2 + z^2} \tag{2.2}$$

Por fim, a mais notável e importante operação no contexto das transformações, a multiplicação entre dois *quaternions*, $\mathbf{q}_1 = (s_1 + x_1\mathbf{i} + y_1\mathbf{j} + z_1\mathbf{k})$ e $\mathbf{q}_2 = (s_2 + x_2\mathbf{i} + y_2\mathbf{j} + z_2\mathbf{k})$, é definida por:

$$\mathbf{q}_1\mathbf{q}_2 = (s_1 s_2 - x_1 x_2 - y_1 y_2 - z_1 z_2)$$

$$+ (s_1x_2 + s_2x_1 + y_1z_2 - y_2z_1)\ \mathbf{i}$$
$$+ (s_1y_2 + s_2y_1 + z_1x_2 - z_2x_1)\ \mathbf{j}$$
$$+ (s_1z_2 + s_2z_1 + x_1y_2 - x_2y_1)\ \mathbf{k}$$

Simplificando os termos, define-se a multiplicação entre *quaternions* em função dos produtos escalar e vetorial (Equação 2.3).

$$\mathbf{q_1q_2} = [\ (s_1s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2), (s_1\mathbf{v}_2 + s_2\mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2)\ ] \tag{2.3}$$

Aplicação em Rotação

A rotação de um ponto em torno de um eixo arbitrário utilizando *quaternions* é bem simples. Primeiro deve-se representar o ponto a ser rotacionado na forma de *quaternion*. Seja $P(x, y, z) \in \mathbb{R}^3$ um ponto, sua representação em *quaternion* é $\mathbf{p} = [0 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}]$.

Depois é preciso definir o eixo de rotação e o ângulo da rotação, respectivamente representados pelo vetor $\mathbf{u} = (x', y', z')$ e θ. Após isso, o *quaternion* de rotação $\mathbf{q}_r$ é construído de acordo com a Equação 2.4.

$$\mathbf{q}_r = \left[\cos(\theta/2),\ \sin(\theta/2)\ \frac{\mathbf{u}}{\|\mathbf{u}\|_2}\right] \tag{2.4}$$

Com os *quaternions* definidos basta multiplicar $\mathbf{q}_r$ por $\mathbf{p}$ e o resultado pela inversa de $\mathbf{q}$ como definido na Equação 2.5. O *quaternion* resultante $\mathbf{p}_r$ será igual ao ponto rotacionado representado na forma de *quaternion*, bastando, então, extrair o vetor imaginário e convertê-lo para um ponto do $\mathbb{R}^3$. Para recuperar o ponto original, aplica-se as multiplicações na ordem inversa (Equação 2.6).

$$\mathbf{p}_r = \mathbf{q}_r\ \mathbf{p}\ \mathbf{q}_r^{-1} \tag{2.5}$$

$$\mathbf{p} = \mathbf{q}_r^{-1}\ \mathbf{p}_r\ \mathbf{q}_r \tag{2.6}$$

Como dito anteriormente, é possível representar *quaternions* como uma matriz de transformação. Também é possível fazer o caminho inverso, dada uma matriz de rotação transformá-la em *quaternion*. Essas conversões são dadas abaixo nas Equações 2.7 e 2.8,

respectivamente, sendo $\mathbf{q} = [s + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}]$ um *quaternion* de rotação unitário e R uma matriz de rotação normalizada.

$$R(\mathbf{q}) = \begin{bmatrix} 1 - 2(y^2 + z^2) & 2(xy - sz) & 2(xz + sy) \\ 2(xy + sz) & 1 - 2(x^2 + z^2) & 2(yz - sx) \\ 2(xz - sy) & 2(yz + sx) & 1 - 2(x^2 + y^2) \end{bmatrix} \tag{2.7}$$

$$\begin{aligned} &\mathbf{q}(R) = [\ (w/2) + ((R_{3,2} - R_{2,3})/2w)\mathbf{i} + ((R_{1,3} - R_{3,1})/2w)\mathbf{j} + ((R_{2,1} - R_{1,2})/2w)\mathbf{k}\ ] \\ &\text{onde} \quad w = \sqrt{1 + R_{1,1} + R_{2,2} + R_{3,3}} \end{aligned} \tag{2.8}$$

2.4 Trabalhos Relacionados

Em Baratoff et al. (2002) foi proposto um método semi-automático de calibração, com base em imagens previamente tiradas e ajustamento de feixes perspectivos (*bundle adjustment*). Os experimentos obtiveram um tempo de calibração próximo a um segundo. Foi possível avaliar até quarenta fotos e escolher o melhor par para realização dos cálculos. Porém este método só é eficiente quando todos os marcadores se encontram próximos o suficiente para que a maioria deles estejam visíveis na mesma imagem, como observado em Klopschitz e Schmalstieg (2007). Por esse motivo o autor propôs um método semelhante aplicável a áreas maiores. A proposta foi mover uma câmera pelo cenário que faz a captura dos marcadores e de características naturais na cena, armazenando as relações em uma base de dados.

No trabalho desenvolvido por Uematsu e Saito (2005), foi proposto um método capaz de calcular as relações entre cada um dos marcadores que tiveram seus sistemas de coordenadas independentes combinados em um espaço projetivo único. O sistema captura duas imagens aleatórias de posições diferentes, ambas as imagens devem ser obtidas tendo todos os marcadores visíveis para que o espaço seja projetado. Como nem sempre todos os marcadores poderão estar visíveis na mesma cena, não importando o ângulo em que são capturadas as imagens, essa restrição torna-se indesejável.

Em Siltanen et al. (2007) foi proposta uma abordagem utilizando uma estrutura

de grafos para interconectar marcadores que não são visíveis na cena ao mesmo tempo. Dessa forma, sempre que existir um caminho no grafo que conecte-os por intermédio de outros marcadores do caminho, a relação entre eles pode ser obtida acumulando-se as transformações do trajeto. O sistema mantém o referencial fixo e utiliza marcadores temporários em cenários muito grandes a fim de criar uma rota entre os pares distantes. Uma desvantagem dessa abordagem é o número desnecessário de cálculos para a obtenção do resultado final em algumas situações. Um menor caminho composto por n marcadores implica em $n - 1$ multiplicações matriciais. A isso soma-se o custo de encontrar o menor caminho em um grafo.

Uma outra abordagem de calibração de marcadores distribuídos aleatoriamente na cena é sugerida em Caetano et al. (2013). Neste trabalho o marcador usado como referência é alterado de acordo com sua qualidade em cada *frame* de vídeo. Diversos métodos são sugeridos como forma de definir a qualidade dos marcadores. O objetivo do trabalho é eliminar a fase de pré-processamento, fazendo a calibração em tempo de execução direto na aplicação. Entretanto o método obtém somente as relações entre marcadores visíveis ao mesmo tempo, o que é uma limitação do trabalho.

O presente trabalho é uma continuação do método desenvolvido em Caetano et al., trazendo como principal contribuição o método de calibração entre pares de marcadores com oclusão total entre si. Fazendo uma analogia ao trabalho de Siltanen et al., o método proposto busca, a cada *frame*, um caminho que conecte dois marcadores não visíveis através de um único marcador intermediário. Mesmo quando houver um par distante o suficiente para que dois ou mais marcadores sejam necessários nesse caminho, o método vai criando ligações em todas as direções, de modo que em algum momento as duas extremidades poderão ser conectadas por intermédio de um só marcador.

O uso de marcadores fiduciais em aplicações de RA ainda é muito frequente por propiciar o registro preciso dos marcadores que comporão o cenário e exigir baixo poder de processamento se comparado a abordagens que utilizam características naturais da cena como os trabalhos propostos em Chen e Li (2010), Kanbara et al. (2001), Yuan et al. (2006) e em Zhao et al. (2008).

3 Método Proposto

Este capítulo tem como objetivo descrever o método de calibração proposto pelo trabalho. Para isso, foram adotadas algumas nomenclaturas. O símbolo **b** representa o marcador base do sistema, que é escolhido conforme alguns critérios descritos na seção 3.2. Adotam-se ainda as seguintes notações:

- $M(S)$ é o conjunto de marcadores registrados no sistema;
- Mi é a matriz de transformação de $i \in M(S)$ em relação à câmera;
- Mib é a matriz de transformação de $i \in M(S)$ em relação à base $b \in M(S)$;
- Cf_i e Cf_{ib} são os valores de confiança das transformações supracitadas;

As demais seções deste capítulo detalham cada uma das etapas do fluxo de execução geral da aplicação desenvolvida, sendo essas etapas realizadas na sequência ilustrada na Figura 3.1. O capítulo ainda descreve a fase de renderização, apresentando a forma como as relações são exibidas ao usuário durante o processo de calibração.

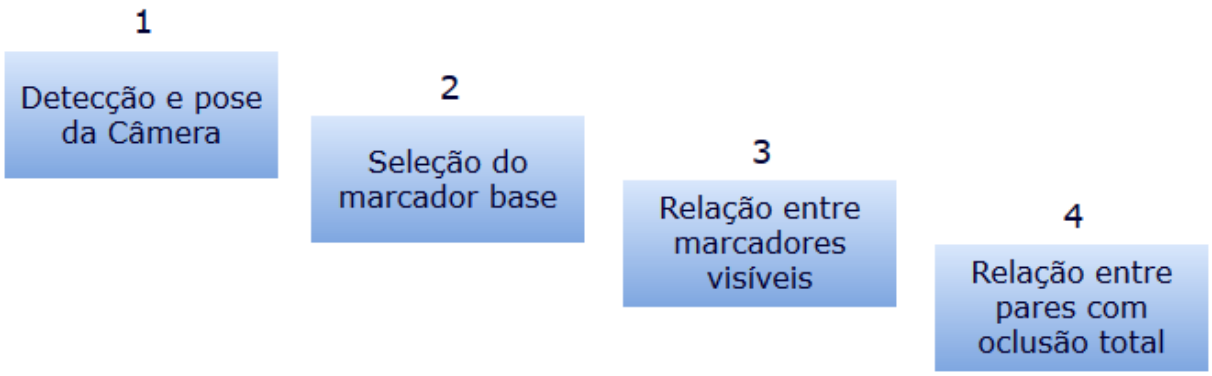


Figura 3.1: Etapas do método desenvolvido

3.1 Detecção e Pose da Câmera

Esta etapa consiste em analisar cada *frame* de vídeo a procura dos marcadores previamente registrados no sistema e calcular, para cada um deles, sua transformação em relação

à câmera. Como dito na seção 2.2.2 a *AVRLib* executa essa tarefa de forma automática em seu *loop* principal. Porém, ela permite alterar seu comportamento a partir da derivação da classe abstrata *avrSystemMarker*. Desta forma, foi criada uma especificação dessa classe para gerir os marcadores e estimar suas transformações.

Uma outra notação utilizada nesta seção é $M(D)$, que representa o conjunto de quadrados detectados pela AVRLib em um *frame* do vídeo que podem vir a ser alguns dos marcadores de $M(S)$. A biblioteca repassa esse conjunto de possíveis marcadores à classe desenvolvida, que identificará quais deles são marcadores de fato.

A análise dos elementos de $M(D)$ é feita pela simples comparação de identificadores, que foram atribuídos a cada elemento de $M(S)$ pela própria biblioteca. Quando o ID do marcador corresponder ao ID de um quadrado detectado na imagem, este é o marcador procurado, logo ele é configurado como visível no sistema.

Continuando a execução, a transformação rígida da câmera é calculada através de duas funções da biblioteca, a *avrGetTransMat* e a *avrGetTransMatCont*, sendo que essa última leva em consideração a matriz de um quadro anterior em que o marcador estava visível. Ambas as funções retornam um valor de *erro* da transformação calculada. Esse $erro > 0$ é usado na definição do valor de confiança Cf_i dado na Equação 3.1. O uso da matriz de um quadro anterior é sugerido em Siltanen et al. (2007) e serve como forma de minimizar o erro de projeção.

$$Cf_i = \frac{1}{erro(M_i)} \tag{3.1}$$

A Figura 3.2 contém o fluxograma de toda essa etapa.

3.2 Seleção do Marcador Base

Como discutido nos capítulos anteriores, sistemas de múltiplos marcadores possuem um referencial global, de modo que cada marcador do sistema é posicionado e orientado em relação a esse ponto de referência. Usar um dos marcadores do próprio sistema como referência para os demais elimina a calibração para ele mesmo (Wang et al., 2010). É possível variar esse marcador, chamado de *base*, de acordo com a qualidade do mesmo na

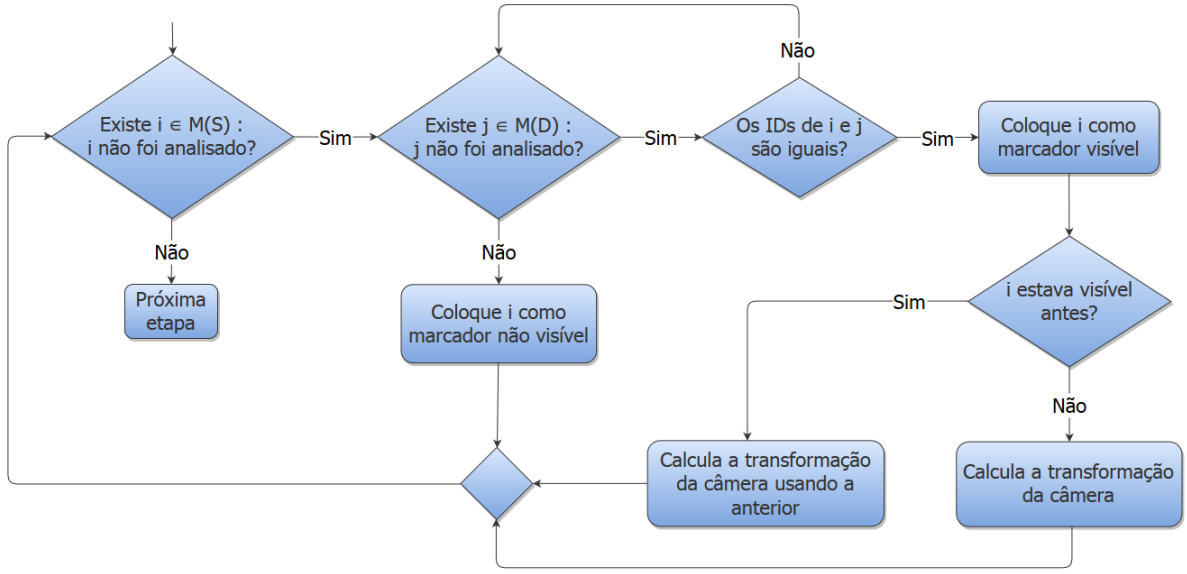


Figura 3.2: Fluxograma da etapa de análise dos marcadores e transformação da câmera

cena, de modo a tentar diminuir o erro da calibração entre a base e os demais marcadores. Em Caetano et al. (2013) são propostas algumas políticas de troca de marcador base.

Para efeito de comparação de resultados, serão utilizados três critérios de definição do melhor marcador da cena. O marcador base será substituído quando existir um marcador com qualidade $X\%$ superior à da base atual. Esse valor de porcentagem é um parâmetro do sistema e para este trabalho foi definido como $X = 10$, baseado no trabalho de Caetano et al.. O restante da seção descreve esses critérios e a Figura 3.3 faz um resumo desta etapa.

3.2.1 Maior Confiança

Como dito anteriormente, as funções que calculam a transformação rígida da câmera retornam um valor de erro, que representa a margem de erro da matriz. Esse valor é usado para obter a confiança dessa transformação usando a relação da Equação 3.1. Portanto, este método define como o melhor marcador da cena o que tiver o maior valor de confiança.

3.2.2 Menor Inclinação

De acordo com Kato (2005) o rastreamento é afetado pelo ângulo de inclinação do marcador em relação à câmera. Como é desejável que os marcadores estejam inclinados para um melhor aproveitamento do cenário, considerar o marcador menos inclinado como base pode aumentar a acurácia da calibração. Logo, esse é um dos critérios que pode ser utilizado pelo sistema. O cálculo da inclinação é feito conforme em Caetano et al..

3.2.3 Menor Inclinação e Erro Aceitável

Este método é uma combinação dos dois critérios anteriores. Primeiramente são escolhidos os marcadores que possuem um erro de transformação aceitável, i.e., pertencente ao intervalo $(0, 1)$. Em seguida, o melhor marcador da cena é definido para aquele que estiver com o menor ângulo de inclinação em relação à câmera.

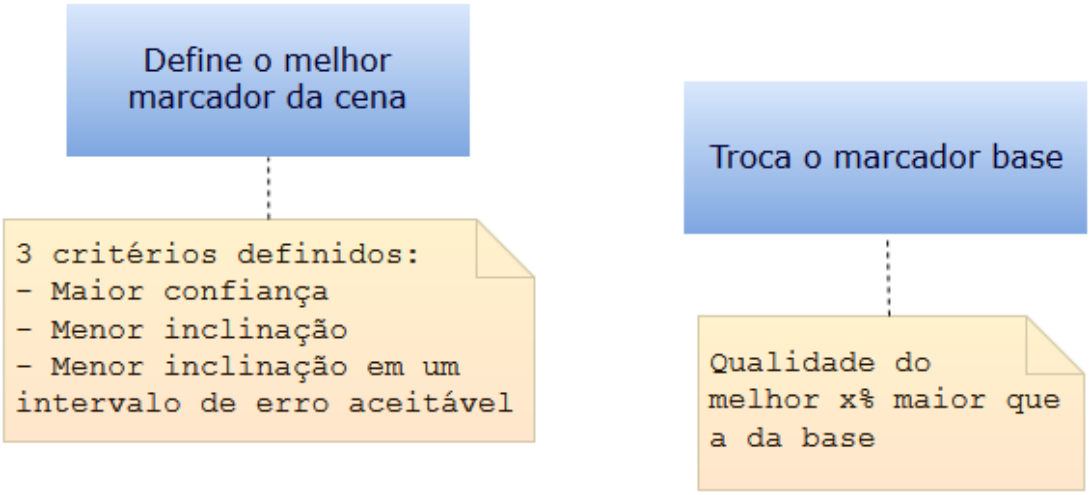


Figura 3.3: Resumo da etapa de seleção do marcador base

3.3 Relação entre Marcadores Visíveis

Quando mais de um marcador é visível, um deles é definido como base e a transformação Mib_t, $i \in M(S)$ e $i \neq b$, é obtida pela Equação 3.2. O novo índice t nas matrizes indica o quadro de vídeo corrente.

$$Mib_t = Mb_t^{-1} \times Mi_t \tag{3.2}$$

De modo análogo, obtêm-se a transformação inversa, ou seja, a transformação da base b em relação a i, pela Equação 3.3.

$$Mbi_t = Mib_t^{-1} = Mi_t^{-1} \times Mb_t \tag{3.3}$$

O valor de confiança dessas relações foi definido como sendo o inverso da média aritmética entre o erro da transformação da câmera de cada marcador. Pode-se escrever este cálculo como na Equação 3.4.

$$Cf_{ib_t} = Cf_{bi_t} = \frac{2}{erro(Mi_t) + erro(Mb_t)} \tag{3.4}$$

As três equações anteriores são usadas para obter apenas a transformação e confiança do *frame* corrente, mas as informações que são armazenadas decorrem de todas as informações obtidas em *frames* anteriores. O motivo para isso é simples, o objetivo do método é obter as relações entre todos os marcadores, independente dos mesmos estarem ou não visíveis em um determinado *frame*. Esses marcadores devem estar distribuídos na cena e não deve haver mudança na posição e orientação de um marcador em relação a outro, logo devem ficar estáticos. Isso implica que as relações não mudam com o tempo, porém, não é o que acontece na prática devido aos vários problemas com relação a precisão já discutidos neste trabalho. Em um *frame* t qualquer a transformação obtida pode estar muito próxima a real, porém em um outro *frame* t' a transformação pode conter grande imprecisão por fatores externos. Em vista disso, a abordagem utiliza o cálculo da média ponderada dos valores obtidos anteriormente, tanto para a transformação final quanto para a confiança correspondente.

A confiança final é a média ponderada entre os valores de confiança nos *frames* $t-1$ e t. O peso de Cf_{ibt-1} é dado pelo número de vezes que a transformação final foi atualizada até o *frame* $t-1$ (denotado por *upd*) e o peso de Cf_{ib_t} é 1, visto que ela é válida apenas no *frame* corrente (Equação 3.5).

$$Cf_{ib} = Cf_{bi} = \frac{upd * Cf_{ib_{t-1}} + Cf_{ib_t}}{upd + 1} \tag{3.5}$$

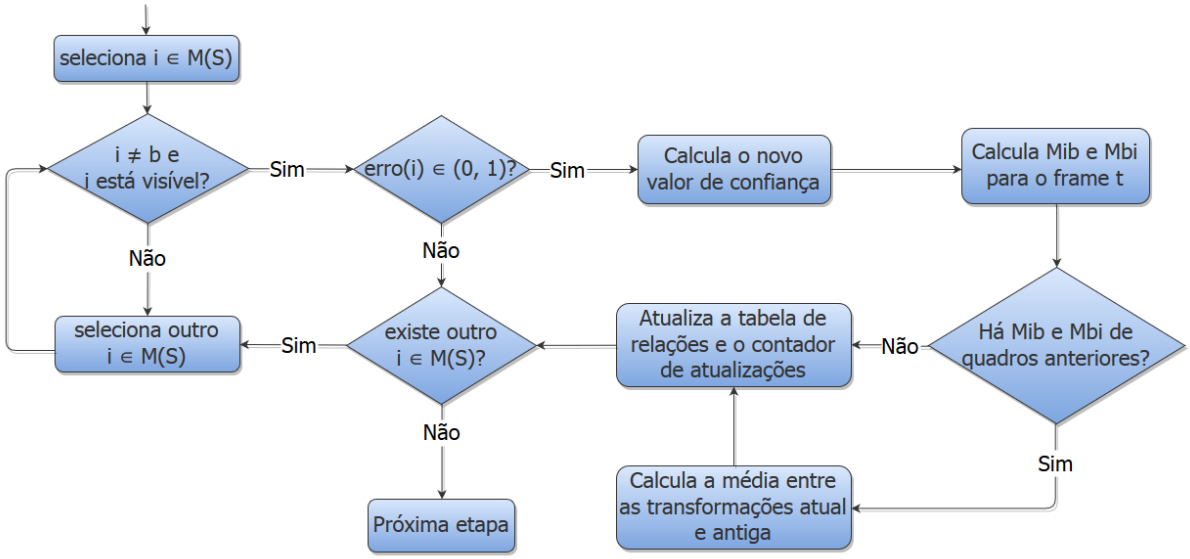


Figura 3.4: Fluxograma da etapa de transformação entre os marcadores visíveis

O mesmo se aplica ao cálculo da matriz de transformação final. A Equação 3.6 mostra esse cálculo para Mib. Para a matriz Mbi o cálculo é exatamente o mesmo bastando-se trocar a posição dos índices i e b.

$$Mib = \frac{upd * Mib_{t-1} + Mib_t}{upd + 1} \tag{3.6}$$

O fluxo de execução dessa etapa está simplificado no fluxograma da Figura 3.4. Em linhas gerais, percorre-se todo o conjunto $M(S)$ calculando as transformações Mib_t e Mbi_t, $\forall i$ visível. Se é a primeira vez que essas matrizes são calculadas, as próprias já são as matrizes finais para esse *frame*, senão calcula-se a média. Em ambos casos, os cálculos só serão realizadas se $Cf_{ib_t}^{-1}, Cf_{bi_t}^{-1} \in (0, 1)$, que é o intervalo de erro aceitável para o método. Ao final o contador de atualizações da transformação será atualizado.

3.4 Relação entre Pares com Oclusão Total

Em uma distribuição aleatória de marcadores em um cenário qualquer, pares de marcadores com oclusão total entre si poderão aparecer. Nesses casos, apenas a aplicação do método anterior não é o suficiente para completar a calibração. Portanto é preciso uma maneira diferente de estimar tais relações.

A Figura 3.5 ilustra um grafo de visibilidade de uma potencial configuração de marcadores. Observa-se que nessa configuração existem três pares de marcadores que jamais serão visualizados em um mesmo quadro do vídeo. Esses pares são $\{(1,3),(1,4),(2,4)\}$. Entretanto, quaisquer outros pares podem ser encontrados juntos ao mesmo tempo. Adotando como exemplo os pares $(1,2)$ e $(2,3)$, pelo método anterior, em momentos distintos o sistema computa as matrizes M_{12}, M_{23}, M_{32} e M_{21}. Analisando a construção dessas matrizes definida na Equação 3.2, observa-se que é possível obter as relações M_{13} e M_{31} usando o marcador 2 como intermediário.

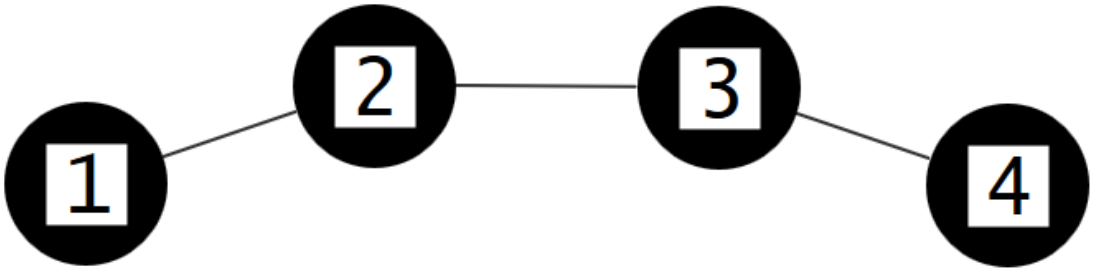


Figura 3.5: Representação da visibilidade entre os marcadores como um grafo. Arestas indicam que há um ponto de vista onde ambos os marcadores conectados são visíveis.

A matriz M_{12} posiciona o marcador 1 no sistema de coordenadas do marcador 2. Do mesmo modo, a matriz M_{23} posiciona o marcador 2 em relação à 3. Então é possível posicionar 1 no sistema do marcador 3 aplicando ambas as transformações em sequência. O processo é análogo para obter M_{31}, mas usa-se as matrizes M_{21} e M_{32}. A demonstração formal desse processo vem a seguir.

Sejam A, B e C marcadores de uma configuração tal que em seu grafo de visibilidade não existe a aresta $E = \{(A,C)\}$. As matrizes M_{AB}, M_{BC}, M_{CB} e M_{BA} são obtidas da Equação 3.2. Multiplicando M_{BC} por M_{AB} e, M_{BA} por M_{CB}, na ordem em que foram citadas, visto que a multiplicação matricial não é uma operação comutativa, obtém-se:

$$M_{BC} \times M_{AB} = (M_C^{-1} \times M_B) \times (M_B^{-1} \times M_A)$$
$$M_{BA} \times M_{CB} = (M_A^{-1} \times M_B) \times (M_B^{-1} \times M_C)$$

Pela associatividade da multiplicação matricial, reescrevemos as equações como:

$$M_{BC} \times M_{AB} = M_C^{-1} \times (M_B \times M_B^{-1}) \times M_A$$
$$M_{BA} \times M_{CB} = M_A^{-1} \times (M_B \times M_B^{-1}) \times M_C$$

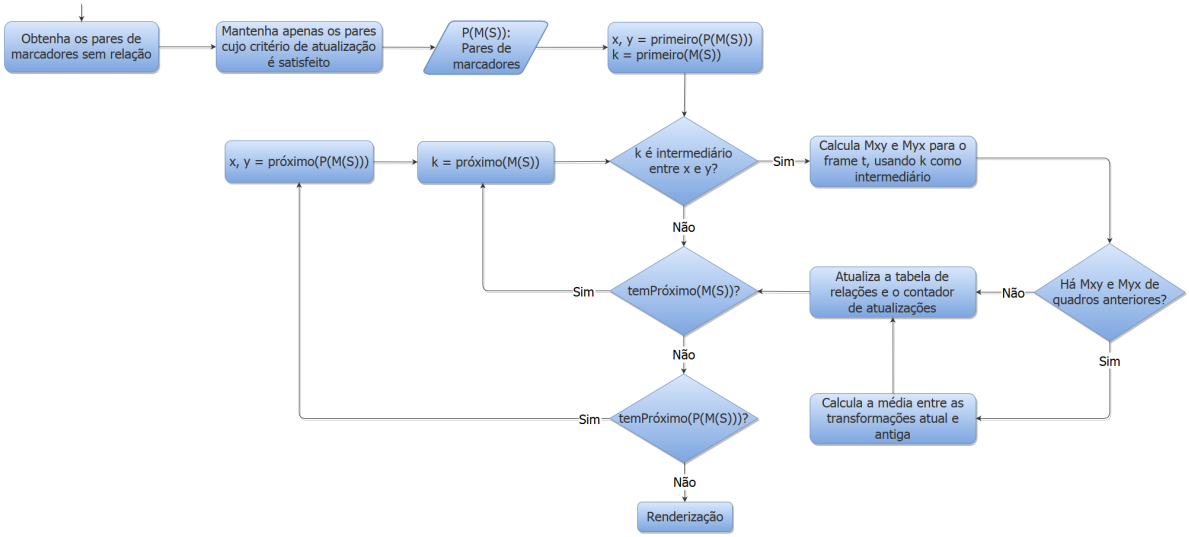


Figura 3.6: Fluxograma da etapa de transformação entre os marcadores não visíveis

Considerando que o produto $M_B \times M_B^{-1}$ resulta na matriz identidade I, obtemos:

$$M_{BC} \times M_{AB} = M_C^{-1} \times M_A = M_{AC}$$
$$M_{BA} \times M_{CB} = M_A^{-1} \times M_C = M_{CA}$$

Durante a execução do sistema, a estimativa da transformação entre A e C não se dará apenas por um marcador intermediário, mas por todos os marcadores que servirem como intermediários. Isto é, $\forall k \in M(S), k \neq A, k \neq C$, se existem as matrizes M_{Ak} e M_{kC} e estas tem o valor de confiança definido no intervalo $(0, 1)$, então k é usado como intermediário. Desse modo as Equações 3.5 e 3.6 também são válidas nessa etapa.

Existe um critério para esse método ser executado. Como ele utiliza mais operações, a propagação de erro tende a ser maior, então é conveniente evitá-lo quando possível. Para a execução do método, definiu-se um número mínimo de atualizações para todas as relações necessárias ao cálculo. Logo, para cada par de marcadores $x, y \in M(S)$ sem relação, o método somente é aplicado quando, para todo intermediário k, as matrizes Mxk e Mky tiverem sido atualizadas ao menos N vezes. Assim, ganha-se um tempo para que os outros pares diferentes apareçam e, mesmo quando não aparecerem, o sistema garante que as relações tenham um valor de confiança elevado. O fluxo de execução pode ser conferido no fluxograma da Figura 3.6.

3.5 Visualização da calibração

Na *callback* de renderização as relações são exibidas no vídeo por meio de quadrados verdes que contornam os marcadores que tiveram sua relação com a base calculada. Sobre o marcador base são desenhados os eixos Ox e Oy e o contorno é feito na cor azul. A Figura 3.7 mostra um quadro da execução da aplicação de calibração com todas as relações já calculadas. Nesse exemplo foi usada uma configuração em que os marcadores são dispostos sobre uma base cúbica.

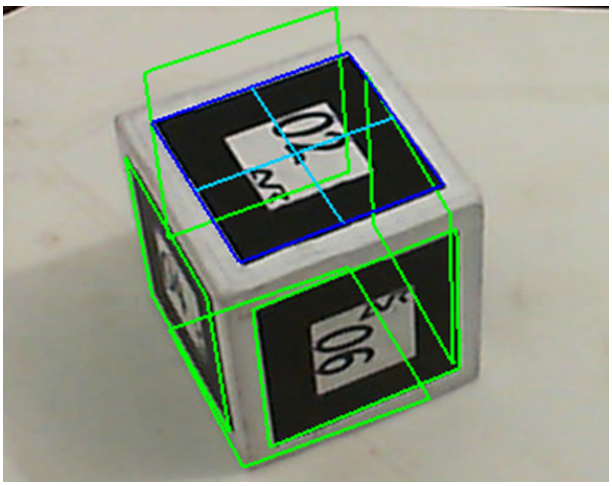


Figura 3.7: Quadro da execução do sistema para o cubo com todas relações calculadas.

4 Resultados

Este capítulo apresenta os resultados obtidos dos experimentos realizados a fim de validar o método proposto. Os experimentos foram executados em um notebook com processador Intel Core i5 2410m 2ª geração, 6GB HDR3 e placa de vídeo integrada Intel Graphics 3000. A câmera de vídeo utilizada para captura das imagens do mundo real foi uma Microsoft LifeCam 720p. As configurações de marcadores testadas estão ilustradas nas fotos da Figura 4.1. Todos os testes foram executados em uma faixa entre 27fps e 30fps. As seções seguintes apresentam a métrica de erro utilizada e os testes para cada configuração.

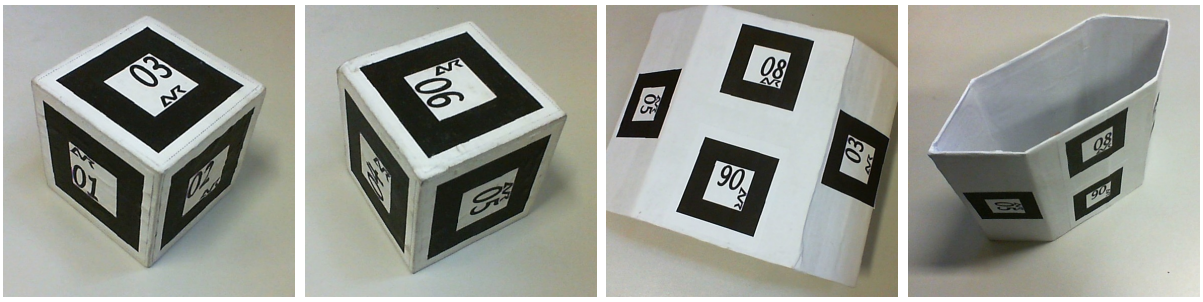


Figura 4.1: Configurações de marcadores utilizadas neste trabalho. As duas à esquerda mostram a configuração sobre uma base cúbica. As duas à direita mostram a configuração sobre um tipo de cilindro hexagonal irregular

4.1 Métrica de Erro

Para calcular o erro de projeção foram seguidas as propostas sugeridas nos trabalhos de Siltanen et al. (2007) e Uematsu e Saito (2005). Esses trabalhos utilizaram a média das distâncias, em milímetros, entre os vértices de dois cubos projetados sobre o marcador, um usando a transformação da câmera deste marcador e o outro usando a transformação estimada da câmera, calculada segundo a Equação 4.1 (Figura 4.2). Porém, foi necessária uma modificação nesse cálculo para o presente trabalho, visto que todas as relações são armazenadas e que o referencial do sistema pode variar.

Dessa forma, calcula-se normalmente a média das distâncias entre os vértices para cada marcador visível junto à base corrente. Em seguida, uma segunda média é calculada

referente ao número de marcadores visíveis. Esse processo é realizado para cada marcador que em algum momento assumiu o papel de referencial do sistema. Esse cálculo é repetido em cada *frame* e então encontramos o valor do erro de projeção máximo para cada base durante toda a execução do sistema.

$$M_{b_{est}} = M_i \times M_{bi} \tag{4.1}$$

$$e_{proj} = \frac{1}{8n} \sum_{\substack{i=1 \\ i \neq b \\ i \text{ visível}}}^{n} \sum_{k=1}^{8} \| M_b \times V_k - (M_i \times M_{bi}) \times V_k \|_2 \tag{4.2}$$

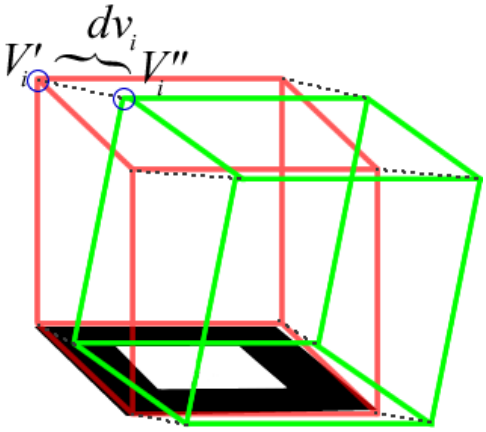


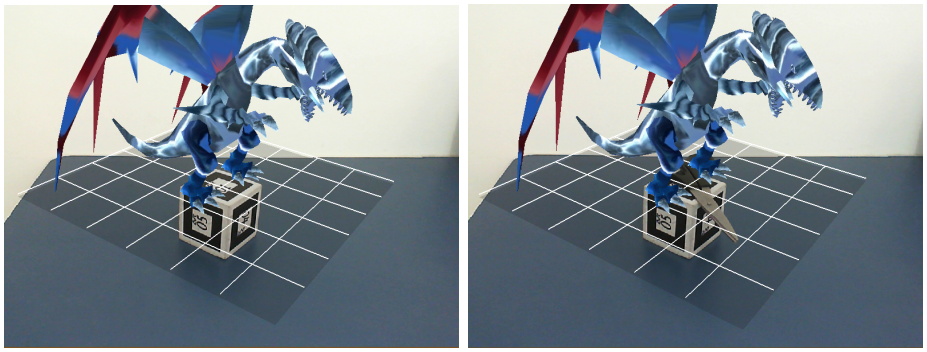
Figura 4.2: Ilustração do cálculo do erro de projeção para um marcador

4.2 Resultados Qualitativos

O método implementado tem como saída arquivos de configuração de um ambiente de realidade aumentada multimarcador. Uma aplicação, chamada *DragonTest*, foi criada a fim de testar os arquivos gerados pelo sistema. Essa aplicação realiza a renderização de um dragão sobre um plano.

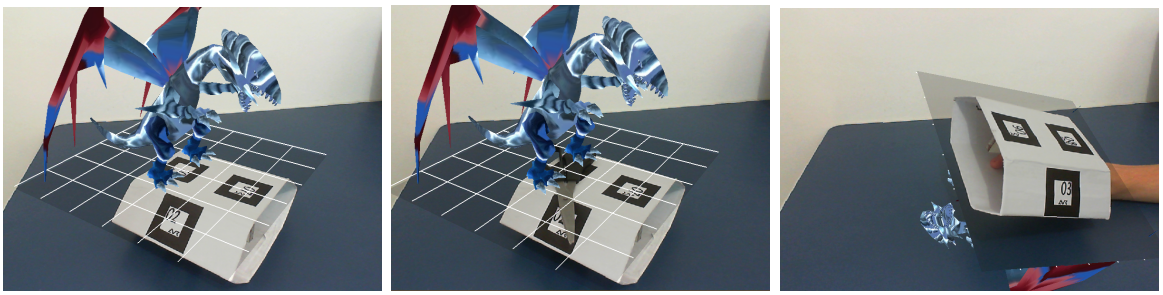
Para o cubo, a base do sistema gerado é o marcador 1. O resultado pode ser conferido na Figura 4.3. É possível notar uma leve alteração visual na projeção quando a base é coberta (Figura 4.3.b).

Para a segunda configuração de marcadores, a base do sistema é o marcador 2, entretanto, a projeção foi realizada sobre o marcador 1 a fim de manter o plano na horizontal. Para obter este efeito, a matriz de transformação do marcador 2 em relação à



(a) (b)

Figura 4.3: Execução da aplicação DragonTest para o marcador cúbico. Em (a) o marcador base está visível. Em (b) a base e outro marcador, antes visíveis, são cobertos o que acarreta em uma pequena alteração na projeção.



(a) (b) (c)

Figura 4.4: Execução da aplicação DragonTest para o marcador hexagonal. Em (a) a base do sistema está visível. Em (b) a base e o marcador onde a projeção foi aplicada estão cobertos. Em (c) o cilindro é rotacionado até que não apareça nenhum marcador do mesmo plano de visão da base.

câmera foi multiplicada pela matriz M_{12} gravada em arquivo. O resultado é a matriz de transformação estimada do marcador 1 em relação à câmera.

$$M_2 \times M_{12} = M_2 \times (M_2^{-1} \times M_1) = M_{1_{est}} \tag{4.3}$$

Assim, aplicando a transformação $M_{1_{est}}$ é possível estimar a posição do marcador 1 na cena em relação à câmera e, com isso, realizar a projeção sobre ele. O resultado pode ser verificado na Figura 4.4. Novamente, nota-se que a oclusão do marcador base acarreta em uma alteração na projeção que, apesar de ainda ser aceitável, é mais visível que a alteração sofrida no cubo. Um resultado interessante é que a projeção se manteve possivelmente próxima ao marcador 1 mesmo quando o cilindro foi rotacionado, exibindo os marcadores opostos a ele e sem a base estar presente.

4.3 Análise Quantitativa

Na seção 4.1 foi explicada a métrica de erro usada neste trabalho. Essa métrica foi aplicada a cada uma das configurações de marcadores, trabalhadas aqui, para cada método de seleção do marcador base. Os dados gerados estão organizados nas Tabelas 4.1 e 4.2, sendo a primeira referente ao marcador cúbico e a segunda ao cilindro hexagonal irregular. Essas tabelas mostram o erro máximo de projeção, em milímetros, encontrado durante a execução do sistema para cada marcador da configuração que em algum momento foi base do sistema.

É preciso levar em consideração alguns fatores:

- Só é possível calcular o erro de projeção das matrizes de transformação entre a base corrente e os marcadores visíveis no momento. Tomando o cubo como exemplo, isso implica que nos quadros onde o marcador 1 foi a base, a relação entre 1 e 6 não foi levada em consideração no cálculo do erro em nenhum quadro.

- O valor do erro refere-se a apenas um quadro de toda a execução: o quadro onde a base corrente obteve o seu maior erro de projeção.

Tabela 4.1: Erro de projeção máximo, em milímetros, para cada base e cada critério de seleção com a configuração cúbica (6 marcadores)

Base	1	2	3	4	5	6
Maior Confiança	**65.91**	73.28	**62.81**	**68.59**	62.49	52.26
Menor Inclinação	81.77	**60.19**	77.61	77.07	89.95	**41.46**
Inclinação e Erro	102.18	70.71	83.46	77.65	**62.26**	91.22

Tabela 4.2: Erro de projeção máximo, em milímetros, para cada base e cada critério de seleção com a configuração hexagonal (8 marcadores)

Base	1	2	3	4
Maior Confiança	1308.94	768.80	**35.17**	**26.46**
Menor Inclinação	109.13	552.04	55.03	53.01
Inclinação e Erro	**61.14**	**38.30**	59.91	527.91

Base	5	6	7	8
Maior Confiança	1009.43	27.16	305.86	129.25
Menor Inclinação	**63.45**	32.48	**53.51**	**16.12**
Inclinação e Erro	745.22	**22.51**	237.85	35.42

É possível perceber que para o cubo, em todos os métodos de seleção, não houve grande variação do erro para cada marcador. É visível também que o método de *Maior Confiança* se sobressaiu aos demais obtendo, em sua maioria, erros menores. Entretanto a diferença dos valores não é suficientemente grande para afirmar com exatidão que essa proposta de seleção de base seja a melhor para essa configuração.

Em relação ao marcador hexagonal, é visível a grande variação do erro para cada base. Percebe-se que os resultados obtidos nos métodos onde a inclinação em relação à câmera é levada em consideração, são consideravelmente melhores que o método de seleção de *Maior Confiança*. Os erros de cada marcador são, em sua maioria, menores nesses métodos que consideram a inclinação. A variação do erro, interna a cada método, também é muito menor nesses critérios de seleção.

Esses resultados fazem um certo sentido quando observamos as características de cada configuração. Os ângulos de inclinação entre os marcadores da base cúbica são uniformes, sendo assim a inclinação do marcador base em relação à câmera não varia muito entre os *frames*, seja qual for a base corrente. Então, é possível esperar que o ângulo não interfira muito na escolha do marcador base e, portanto, não consiga maximizar a qualidade das relações.

Em contrapartida, o marcador hexagonal possui diferentes ângulos de inclinação inter-marcador. A inclinação é grande entre alguns pares, ficando próxima aos 180 graus. Ao mesmo tempo, existem outros pares no mesmo plano. Essa grande variação na inclinação se refletiu nos resultados, principalmente, quando não considerada na escolha do marcador base.

5 Conclusão e Trabalhos Futuros

Este trabalho propôs um novo método para obter as relações de uma configuração de múltiplos marcadores em uma cena. Esse método é potencialmente útil em cenários onde existam pares de marcadores que não podem ser vistos juntos no mesmo volume de visualização. Nesse caso, o método define um ou mais marcadores intermediários e calcula a relação entre esses pares com oclusão de forma indireta. O trabalho ainda adotou um sistema de variação do marcador base durante o processo de calibração, como feito em Caetano et al. (2013), a fim de aumentar a confiança das relações obtidas. Critérios de seleção do marcador base foram propostos tendo em vista as características dos tipos de cenários considerados. Nos experimentos foram utilizadas duas configurações de marcadores e apresentou-se uma métrica de erro como forma de avaliar os resultados.

Os resultados qualitativos mostraram que o método desenvolvido conseguiu obter relações estáveis entre os pares de marcadores das configurações testadas, cumprindo assim o que se propunha o trabalho. A análise quantitativa teve o objetivo de determinar quais critérios de seleção do marcador base melhor se aplicam a determinados cenários. Para a configuração com diferentes ângulos de inclinação, os dados obtidos destacaram os métodos que consideram a menor inclinação dos marcadores em relação à câmera. Já em cenários onde a inclinação é uniforme, nenhum critério se sobressaiu de fato, tendo o método de confiança apresentado resultados ligeiramente superiores.

Uma limitação do trabalho desenvolvido é a falta de praticidade no manuseio da aplicação. Isso ocorre, principalmente, devido a problemas no rastreamento dos marcadores feito pela biblioteca, causados pela ambiguidade dos padrões a certa escala, movimentações bruscas da câmera ou dos padrões e demais problemas discutidos no início deste trabalho. Isso gera uma transformação errônea que se propaga nas relações calculadas pelo método que muitas vezes se estabilizam ainda incorretas.

Uma forma de corrigir tal problema e, portanto, uma proposta de trabalho futuro, pode ser utilizar outro sistema de rastreamento. Bibliotecas como a ARTag (Fiala, 2005) e ARUco (Jurado et al., 2014) são baseadas em rastreamento de padrões digitais em vez

dos padrões arbitrários do ARToolKit. Conforme os trabalhos citados, essa abordagem possui taxas de falsos-positivos e de confusão inter-marcador muito baixas em comparação com o sistema utilizado neste trabalho.

Outras sugestões de trabalhos futuros para melhoria e otimização do sistema desenvolvido são:

- A implementação de métodos que identifiquem de forma mais precisa as relações obtidas em cada *frame*, impedindo atualizações que possam propagar erros;
- Novos critérios de escolha do marcador base que diminuam a incidência de erros na projeção do objeto virtual;
- Tratamento de ponto flutuante, considerando a precisão da máquina como precisão das variáveis reais do sistema, a fim de minimizar os erros de cálculo devido às limitações do *hardware*;
- Outra interface com o usuário que eleve sua usabilidade.

Os resultados apresentados e discutidos neste trabalho foram publicados no *Symposium on Virtual and augmented Reality* (SVR' 2014) e são disponibilizados pela *IEEE Xplore Digital Library* (Oliveira et al., 2014).

Referências Bibliográficas

Azuma, R. A survey of augmented reality. **Teleoperators and Virtual Environments**, v.6, p. 355–385, 1997.

Baratoff, G.; Neubeck, A. ; Regenbrecht, H. **Interactive multi-marker calibration for augmented reality applications**. In: Proceedings of the 1st International Symposium on Mixed and Augmented Reality, ISMAR '02, p. 107–, Washington, DC, USA, 2002. IEEE Computer Society.

Beato, N.; Zhang, Y.; Colbert, M.; Yamazawa, K. ; Hughes, C. E. Interactive chroma keying for mixed reality. **Computer Animation Virtual Worlds**, v.20, p. 405–415, jun 2009.

Caetano, F. A.; Oliveira, D. C. B. ; Silva, R. L. S. Automatic calibration of multiple markers in an augmented reality environment. **X Workshop de Realidade Virtual e Aumentada (WRVA)**, p. 48–53, 2013.

Chen, Z.; Li, X. **Markless tracking based on natural feature for augmented reality**. In: Educational and Information Technology (ICEIT), 2010 International Conference on, volume 2, p. V2–126–V2–129, Sept 2010.

Cunha, C. H.; Fernandes, S. M. Prototipação de ambientes físicos com realidade aumentada. **Symposium on Virtual and Augmented Reality**, p. 4, 2010.

Exame. **Construtora cria maior projeto de realidade aumentada do mundo**, 07 2010. Disponível em www.exame.abril.com.br/tecnologia/noticias/construtora-cria-maior-projeto-realidade-aumentada-mundo-574938 [Online; acessado em 04/02/2016].

Fiala, M. **Artag, a fiducial marker system using digital techniques**. In: Computer Vision and Pattern Recognition, volume 2, p. 590–596. IEEE, June 2005.

Geiger, C.; Reimann, C.; Sticklein, J. ; Paelke, V. **Jartoolkit - a java binding for artoolkit**. In: Augmented Reality Toolkit, The First IEEE International Workshop, 2002.

Gomez, W. L.; Kirner, C. **Desenvolvimento de aplicações educacionais na medicina com realidade aumentada**. Technical report, Universidade Federal de Lavras, 2015. repositorio.ufla.br/jspui/handle/1/9628.

Haller, M.; Hartmann, W.; Luckeneder, T. ; Zauner, J. **Combining artoolkit with scene graph libraries**. In: Augmented Reality Toolkit, The First IEEE International Workshop, 2002.

Ibáñez, M. B.; Serio, A.; Villarán, D. ; Kloos, C. D. Experimenting with electromagnetism using augmented reality: Impact on flow student experience and educational effectiveness. **Computers & Education**, v.71, 2014.

Juan, M. C.; Alca, M.; Monserrat, C.; Botella, C.; Ba, R. M. ; Guerrero, B. Using augmented reality to treat phobias. **IEEE Computer Graphics and Applications**, v.25, p. 31–37, 2005.

Jurado, S. G.; Salinas, R. M.; Cuevas, F. M. ; Jiménez, M. M. Automatic generation and detection of highly reliable fiducial markers under occlusion. **The Journal of the Pattern Recognition Society**, v.47, p. 2280–2292, June 2014.

Kanbara, M.; Yokoya, N. ; Takemura, H. **A stereo vision-based augmented reality system with marker and natural feature tracking**. In: Proceedings of the Seventh International Conference on Virtual Systems and Multimedia (VSMM'01), VSMM '01, p. 455–, Washington, DC, USA, 2001. IEEE Computer Society.

Kato, H. **Artoolkit 2.33 documentation (alpha version)**, 2005. Disponível em www.hitl.washington.edu/artoolkit/documentation/ [Online; acessado em 10-Maio-2015].

Klopschitz, M.; Schmalstieg, D. **Automatic reconstruction of wide-area fiducial marker models**. In: Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR '07, p. 1–4, Washington, DC, USA, 2007. IEEE Computer Society.

Macedo, S. H.; Fernandes, F. A.; Leite, E. S.; Lima, J. V. ; Biasuz, M. C. V. **Uso de realidade aumentada como apoio ao ensino do campo girante de um motor de corrente alternada**. In: XXIII Simpósio Brasileiro de Informática na Educação (SBIE 2012), 2012.

Machado, R. K.; Farinhaki, R. ; Silva, T. A. **Seta: Ambiente de realidade aumentada para o ensino de algoritmos de aprendizado de máquina**. 2013. Bacharelado - Universidade Tecnológica Federal do Paraná.

Menon, M. J. Sobre as origens das definições dos produtos escalar e vetorial. **Revista Brasileira de Ensino de Física**, v.31, 2009.

Milgram, P.; Takemura, H.; Utsumi, A. ; Kishino, F. **Augmented reality: a class of displays on the reality-virtuality continuum**. In: Proc. SPIE Telemanipulator and Telepresence Technologies, volume 2351, p. 282–292, 1994.

Oliveira, D. C. B.; Caetano, F. A. ; Silva, R. L. S. Avrlib - an object oriented augmented reality library. **X Workshop de Realidade Virtual e Aumentada (WRVA)**, p. 54–59, 2013.

Oliveira, D. C. B.; Caetano, F. A. ; Silva, R. L. S. A method to automate the calibration of a multiple fiducial marker setup. **XVI Symposium on Virtual and Augmented Reality (SVR)**, 2014.

A realidade aumentada na publicidade. Disponível em www.projetual.com.br/a-realidade-aumentada-na-publicidade/ [Online; acessado em 04/02/2016].

Queiroz, A. S.; Oliveira, C. M. ; Rezende, F. S. Realidade aumentada no ensino da química: Elaboração e avaliação de um novo recurso didático. **Revista Eletrônica Argentina-Brasil de Tecnologias da Informação e da Comunicação**, 2014.

Reimann, C.; Engel, S. ; Paelke, V. **Object-oriented toolkit for augmented reality**. In: Augmented Reality Toolkit Workshop, 2003. IEEE International, p. 32–34, 2003.

Siltanen, S.; Hakkarainen, M. ; Honkamaa, P. **Automatic marker field calibration**. In: Virtual Reality International Conference (VRIC2007), p. 261–267, 2007.

Silva, M.; Vilar, E.; Reis, G.; Lima, J. P. ; Teichrieb, V. **Ar jigsaw puzzle: Potencialidades de uso da realidade aumentada no ensino de geografia**. In: XXV Simpósio Brasileiro de Informática na Educação (SBIE 2014), 2014.

UOL. **Microsoft quebra barreiras e traz realidade aumentada para minecraft**, 06 2015. Disponível em www.jogos.uol.com.br/ultimas-noticias/2015/06/15/hololens-leva-mapas-de-minecraft-para-o-mundo-real.htm [Online; acessado em 04/02/2016].

Uematsu, Y.; Saito, H. **Ar registration by merging multiple planar markers at arbitrary positions and poses via projective space**. In: Proceedings of the 2005 international conference on Augmented tele-existence, ICAT '05, p. 48–55, New York, NY, USA, 2005. ACM.

Vince, J. **Mathematics for Computer Graphics**. 2. ed., Springer, 2006.

Wagner, D.; Schmalstieg, D. **ARToolKitPlus for Pose Tracking on Mobile Devices**, p. 139–146. Citeseer, 2007.

Wang, L.; Springer, M.; Hauke, Heibel, T. H. ; Navab, N. **Floyd-warshall all-pair shortest path for accurate multi-marker calibration.** In: ISMAR, p. 277–278. IEEE, 2010.

Yuan, M. L.; Ong, S. K. ; Nee, A. Y. C. Registration using natural features for augmented reality systems. **IEEE Transactions on Visualization and Computer Graphics**, v.12, n.4, p. 569–580, Jul 2006.

Zhao, C.; Yu, Z.; Tian, Q. ; Dalin, Z. **Augmented reality for planar scene via affine transform based natural features tracking**. In: Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on, p. 5085–5089, June 2008.