Universidade Federal do Ceará
Departamento de Engenharia de Teleinformática
Programa de Pós Graduação em Engenharia de Teleinformática

Rodrigo Carvalho Souza Costa

Um novo algoritmo para interação homem-dispositivo portátil multiplataforma baseado em fluxo óptico

Fortaleza
2012

Rodrigo Carvalho Souza Costa

Um novo algoritmo para interação homem-dispositivo portátil multiplataforma baseado em fluxo óptico

Tese de Doutorado apresentada à Coordenação do Curso de Pós-Graduação em Engenharia de Teleinformática da Universidade Federal do Ceará como parte dos requisitos para obtenção do grau de **Doutor em Engenharia de Teleinformática**. **Área de Concentração:** Sinais e Sistemas

Orientador : Prof. Dr. Paulo César Cortez

Fortaleza
2012

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca de Pós-Graduação em Engenharia - BPGE

C875n Costa, Rodrigo Carvalho Souza.
Um novo algoritmo para interação homem-dispositivo portátil multiplataforma baseado em fluxo óptico / Rodrigo Carvalho Souza Costa. – 2012.
115 f. : il. color., enc. ; 30 cm.

Tese (doutorado) – Universidade Federal do Ceará, Centro de Tecnologia, Departamento de Engenharia de Teleinformática, Programa de Pós-Graduação em Engenharia de Teleinformática, Fortaleza, 2012.
Área de Concentração: Sinais e Sistemas.
Orientação: Prof. Dr. Paulo Cesar Cortez.

1. Teleinformática. 2. Percepção visual do movimento. 3. Interação homem-máquina. I. Título.

CDD 621.38

RODRIGO CARVALHO SOUZA COSTA

UM NOVO ALGORITMO PARA INTERAÇÃO HOMEM-DISPOSITIVO PORTÁTIL MULTIPLATAFORMA BASEADO EM FLUXO ÓPTICO

Tese submetida à Coordenação do Curso de Pós-Graduação em Engenharia de Teleinformática, da Universidade Federal do Ceará, como requisito parcial para a obtenção do grau de Doutor em Engenharia de Teleinformática, área de concentração Sinais e Sistemas.

Aprovada em 06/09/2012.

BANCA EXAMINADORA

Prof. Dr. Paulo César Cortez (Orientador)
Universidade Federal do Ceará - UFC

Prof. Dr. Danielo Gonçalves Gomes
Universidade Federal do Ceará - UFC

Prof. Dr. Ernane José Xavier Costa
Universidade de São Paulo - USP

Profa. Dra. Maria Elizabeth Sucupira Furtado
Universidade de Fortaleza - Unifor

Prof. Dr. Francisco Carlos de Mattos Brito Oliveira
Universidade Estadual do Ceará- UECE

Resumo

A diminuição dos custos de hardware associado ao aumento da capacidade de processamento dos dispositivos portáteis tem causado um crescimento muito acelerado do mercado consumidor no Brasil e no mundo, de maneira que estes dispositivos tornaram-se uma parte indispensável no nosso cotidiano. Contudo, o crescente fator de miniaturização de componentes gerou problemas em relação à interação eficiente com periféricos de entrada/saída (IO) tradicionais, como os cursores e o teclado. Assim, para evitar custos adicionais, uma alternativa é utilizar componentes já existentes no dispositivo, como sua câmera integrada. Neste contexto, o objetivo geral desta tese é o desenvolvimento de um novo algoritmo para a interação humano-dispositivo portátil multiplataforma baseado em Fluxo Óptico. Embora o Fluxo Óptico (FO) de Horn e Schunck (1981) ser conhecido por ser um algoritmo de extrema complexidade computacional, nesta tese é proposto um conjunto de adaptações neste algoritmo para tornar possível seu processamento em tempo real em dispositivos portáteis. O algoritmo proposto é desenvolvido em C ANSI e embarcado em dispositivos com sistemas operacionais Qualcomm REX, Symbian e Android. O algoritmo implementado é comparado com métodos presentes na literatura especializada para detecção de movimentos através de simulações computacionais e testes de usabilidade. Os resultados mostram que o algoritmo proposto possui um baixo esforço computacional associado a uma efetiva detecção de movimentos, mostrando que é possível o uso do método de FO como sensor para interação em sistemas embarcados. Além disto, através da metodologia de implementação proposta nesta tese, é possível utilizar as funcionalidades desenvolvidas em diversos tipos de sistemas operacionais.

Palavras-chaves: Detecção de Movimento, Fluxo Óptico, Interação com dispositivos portáteis.

Abstract

The decrease in hardware costs associated with improvement of processing power of embedded devices has caused a rapid growth of the consumer market, which make these devices an indispensable part of our daily life. However, the miniaturization of these components leads to problems in the usability of mobile devices, especially with traditional input interfaces, such as the cursors and keyboard. One solution to avoid this kind of problem without increase production cost of these devices is use resources available, like the embedded camera. Following this idea, the objective of this thesis is the development of a new multiplatform human-handheld interaction algorithm based in optical flow. Although the traditional optical flow algorithms have high computational effort, in this thesis, adaptations and optimizations of this algorithm are proposed to overcome hardware limitations of embedded systems. The proposed algorithm is implemented in ANSI C and is embedded at devices with Android, Rex and Symbian OS. The implemented algorithm is compared with traditional motion detection algorithm through computational simulations and usability tests. The results shown that the proposed algorithm associates a low computational effort associated with effective motion detection. So it is possible to use the optical flow as sensor to interact with handheld devices. Furthermore, through the implementation methodology of this thesis, is possible use the developed functionalities in various operational systems.

Palavras-chaves: Motion Detection, Optical Flow, human-handheld interaction.

Dedico este trabalho a Deus e Todos Meus Irmãos Espirituais.

Agradecimentos

Agradeço primeiramente a Deus por todas as bençãos derramadas durante toda minha vida e ao meus guias espirituais por toda ajuda, inspiração e proteção.

À minha querida esposa e melhor amiga, Antonia Daniele pelo carinho, apoio e incentivo que sempre me dá forças para continuar trabalhando para alcançar os objetivos e o progresso.

Ao Professor Dr. Paulo César Cortez, meu Orientador Científico, pela amizade e a oportunidade de receber um pouco de sua sabedoria, pela disponibilidade apresentada e pelas condições que me proporcionou na realização deste trabalho.

Aos professores Dr[a] Maria Elizabeth Furtado, Dr. Ernane José Xavier Costa, Dr. Francisco Carlos de Mattos Brito Oliveira e Dr. Danielo Gonçalves Gomes por participar de banca examinadora de tese pelas preciosas contribuições que enriqueceram bastante este trabalho.

Aos amigos e companheiros bolsistas e professores membros do projeto SMTRG, em especial ao Cincinato e ao Fábio, pela amizade, os quais pude conviver diariamente e me ajudaram de várias formas à desenvolver este trabalho.

Aos amigos, John, Thomaz, Arimateia e Tarique dentre outros integrantes do sub-grupo de pesquisa em Engenharia Biomédica do LESC, pela amizade e esforço compartilhado durante o curso de doutorado.

Aos amigos Márcia Tonieto, Carlos Manso, Alzir Falcão, pela amizade e pelas pequenas palavras que me fizeram crescer muito e obter força para que possamos fazer cada vez melhor nosso trabalho.

Aos engenheiros de desenvolvimento do SIDI, Giovanni, Miguel Lizarraga, Nelson Sasaki, Giovani Balen e Ariston Carvalho pelo embarque dos algoritmos na plataforma de testes.

Por fim, mas não menos importante, à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP) pelo suporte financeiro através do concedimento da bolsa de Doutorado.

"Não existe um caminho para a felicidade. A felicidade é o caminho."

Mahatma Gandhi

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

API	Interface de Programação de Aplicativos (*Application Programming Interface*)
BDS	BREW *Distribution System*
BREW	*Binary Runtime Environment for Wireless*
CAMShift	*Continuously Adaptive Mean Shift*
CCF	Função de Correlação Cruzada (*Cross-Correlation Function*)
DCT	Transformada Discreta do Cosseno (*Discrete Cosine Transform*)
DT	Diferenciação Temporal
FFT	Transformada Rápida de Fourier (*Fast Fourier Transform*)
FO	Fluxo Óptico
FPS	quadros por segundo (*Frames per Second*)
IO	entrada/saída(*input/output*)
GUI	Interface Gráfica do Usuário(*Graphical User Interface*)
GPS	*Global Positioning System*
IA	Inteligência Artificial
IDE	Ambiente de Desenvolvimento Integrado (*Integrated Development Environment*)
IIHC	Interação Humano-Computador
IHC	Interfaces de Interação Humano-Computador
LESC	Laboratório de Engenharia de Sistemas de Computação
LMS	Mínimos Quadrados (*Least Mean Square*)

MC	Controle Mestre(*Master Control*)
MPEG	Padrão de compressão do *Moving Picture Experts Group*
MSE	Erro Médio Quadrático(*Mean Squared error*)
MSM	*Mobile Station Modem*
NDK	Kit de Desenvolvimento Nativo (*Native Development Kit*)
OpenCV	*Open Source Computer Vision Library*
PC	Computador Pessoal(*Personal Computer*)
PDI	Processamento Digital de Imagens
QCIF	*Quarter Common Intermediate Format*
QVGA	*Quarter Video Graphics Array*
QR *Code*	*Quick Response Code*
RAM	Memória de Acesso Aleatório(*Random Access Memory*)
REX	Sistema Operacional *Real Time Executive*
RGB	Formato de Cor Vermelho Verde Azul (*Red Green Blue*)
SAD	Soma das Diferenças Absolutas (*Sum of Absolute Difference*)
SDK	*Software Development Kit*
SIDI	Samsung Instituto de Desenvolvimento para Informática
SO	Sistema Operacional
SSD	Soma dos Quadrados das Diferenças (*Sum of Squared Differences*)
VC	Visão Computacional
UI	Interfaces de Usuário (*User Interface*)
USB	*Universal Serial Bus*
UFC	Universidade Federal do Ceará

Capítulo 1

Introdução

Pesquisas recentes mostram que o mercado de dispositivos portáteis, como telefones celulares e *tablets*, teve um crescimento muito acelerado no Brasil e no mundo nos últimos anos. A associação entre a diminuição dos custos e de volume de *hardware*, bem como o aumento da capacidade de processamento e armazenamento tornaram este tipo de dispositivo mais popular (WANG *et al.*, 2006; TEXEIRA *et al.*, 2005). Além disto, as inúmeras funcionalidades adicionadas, como tocadores de música, calendário eletrônico e galeria de imagens, tornaram estes aparelhos uma parte indispensável de nosso dia-a-dia, podendo ser considerado por muitas pessoas como uma extensão do corpo humano (YIM *et al.*, 2011; SILVA, 2007).

A cada nova geração, tais dispositivos estão cada vez mais leves, tornando-os mais fácil de carregar e transladar. Contudo, o crescente fator de miniaturização de componentes levou a problemas em relação à interação eficiente com interfaces tradicionais como o teclado e o *touch screen* (YIM *et al.*, 2011; ELIAS *et al.*, 2009).

Atualmente, há um grande interesse de construir celulares com telas maiores, mas seu tamanho é limitado pelo tamanho dos cursores. A cada dia que passa, diminui-se o número de teclas nos celulares e têm suas dimensões reduzidas, que torna, em algumas vezes, difícil e lenta sua interação (YIM *et al.*, 2011; HANNUKSELA *et al.*, 2007; ELIAS *et al.*, 2009). O trabalho de Parhi *et al.* (2006) comprova que os objetos presentes na tela precisam ser maiores que 9,2 mm para poderem ser pressionadas de maneira efetiva e eficaz.

Por causa disto, existe uma série de pesquisas para tornar a interação com dispositivos que possuem telas de toque mais rápida e efetiva. O trabalho de

Dunlop e Levine (2012) descreve uma forma de disposição de teclas de teclados touch-screen para melhorar a eficiência e velocidade de pressionamento de teclas. O trabalho realiza sua comparação com métodos layouts de teclado tradicionais através da medição do número de palavras digitadas e distância percorrida durante a escrita de palavras.

Já em relação à tela de toque, os dedos do usuário geralmente encobrem uma parte da informação exibida. Como consequência, é necessária uma habilidade maior na captura das informações e seleção de menus de forma imediata (YIM *et al.*, 2011; HANNUKSELA *et al.*, 2006).

Atualmente, os dispositivos podem vir acompanhados com diversos componentes adicionais, como acelerômetros, câmeras digitais, *Global Positioning System* (GPS), rede sem fio e *Bluetooth* (ELIAS *et al.*, 2009). Nas últimas décadas, tem havido uma variedade de pesquisas para desenvolver formas de interação usando acelerômetros (MASUI; SIIO, 2000; RUIZ *et al.*, 2011), sensores de toque (HINCKLEY *et al.*, 2000; HARRISON, 1998) e de proximidade (HINCKLEY *et al.*, 2000) com o intuito de melhorar a forma de utilização dos aparelhos. Um exemplo disto é a utilização do acelerômetro como forma de interação com dispositivos móveis. No trabalho desenvolvido por Ruiz *et al.* (2011), é proposto o uso de reconhecimento de gestos para disparar ações no dispositivo, como atender uma chamada ou navegar em um mapa. Neste trabalho, o usuário realiza um movimento com o dispositivo, gerando uma interface de interação através da translação e rotação em três dimensões.

Apesar das soluções que utilizam a estratégia de incluir sensores ser eficiente, muitos fabricantes optam por não lançar este tipo de solução ao mercado devido ao aumento do valor agregado de venda(WANG *et al.*, 2006). Considerando-se modelos atuais, um dispositivo portátil com todos esses sensores chega ao mercado com valor de venda próximo a 400 reais, enquanto que um dispositivo mais básico com apenas Bluetooth e câmera chega ao mercado por cerca de 200 reais.

Idealmente, um sistema de interação humano-dispositivo portátil não deve requerer componentes adicionais a serem acoplados nos aparelhos disponíveis no mercado. Pelo contrário, deve utilizar os componentes de fábrica. Assim, uma vez que os dispositivos possuem uma câmera digital embutida e seu valor de venda são extremamente baixos, uma abordagem mais amigável pode ser alcançada através da aquisição e do reconhecimento de movimentos utilizando técnicas de Visão Computacional (SHAN *et al.*, 2007; BARNARD *et al.*, 2007; GRUNNET-JEPSEN *et al.*,

2006).

O aumento crescente do poder de processamento dos dispositivos portáteis torna possível processar as imagens capturadas pela câmera integrada e oferecer novos aplicativos ao usuário como, por exemplo, detecção de movimento, reconhecimento de padrões, detecção de cor, dentre outros. Estes aplicativos possibilitam que telefones celulares e *smartphones* sejam utilizados em aplicações, tais como, assistência a deficientes visuais, segurança ou jogos de realidade virtual.

Outra possível aplicação dos métodos de Visão Computacional (VC) seria o desenvolvimento de interface baseada em movimento, ou seja, um componente capaz de interagir diretamente com os dispositivos, com funções equivalentes a botões, alavancas, sensores de pressão, etc. Isto permita ao usuário realizar a interação do dispositivo com elementos baseados em movimento ou no mapeamento do corpo desta pessoa (CAETANO *et al.*, 2010), fazendo com que a interação com tal dispositivo seja feita similar ao que hoje em dia temos no mouse de um computador.

Pesquisas recentes mostram que, apesar dos usuários não estarem familiarizados com o conceito de interfaces baseadas em movimento, contudo, este tipo de interação possui uma grande aceitação dos usuários, sendo uma das principais motivações deste trabalho (BENYON, 2011; YIM *et al.*, 2011).

1.1 Objetivos e Hipóteses Gerais da Tese

Esta tese tem como objetivo apresentar os resultados científicos que comprovam as hipóteses de que **o algoritmo de Fluxo Óptico (FO) pode ser processado por dispositivos portáteis em tempo real**, de que este algoritmo pode ser utilizado como base para a construção de uma interface de interação por movimento utilizando a câmera do próprio dispositivo e de que a metodologia de implementação pode ser utilizada de forma independente ao tipo de sistema operacional do dispositivo portátil.

Desta forma, o objetivo geral desta tese é o desenvolvimento de um novo algoritmo de detecção de movimentos baseado em FO de Horn e Schunck (1981), possibilitando a criação de uma interface de entrada para dispositivos portáteis baseado no movimento do próprio dispositivo, de forma que a metodologia possa ser aplicada em diferentes Sistemas Operacionais (SOs).

1.2 Objetivos específicos

Para atingir o objetivo da tese, faz-se necessário o desenvolvimento dos seguintes objetivos específicos:

- pesquisa e estudo de métodos de Interfaces de Interação Humano-Computador (IHC) baseados em Visão Computacional como, por exemplo, os métodos de detecção de movimentos;
- desenvolvimento de uma metodologia para avaliar quantitativamente detecção de movimentos (utilizando vídeo sintético);
- implementação de um sistema computacional para avaliar computacionalmente o método proposto e compará-lo com algoritmos presentes no estado da arte;
- desenvolvimento de uma modificação de realce de imagens para melhorar a eficiência da detecção de movimento em um ambiente de apresentação;
- desenvolvimento de um método baseado em Fluxo Óptico para estimar os deslocamentos feitos pelo dispositivo com baixo esforço computacional;
- desenvolvimento de um algoritmo para classificar o movimento realizado pelo dispositivo como rotação horária, anti-horária, aproximação e distanciamento utilizando os vetores de velocidade estimados pelo Fluxo Óptico;
- implementação, adaptação e otimização dos códigos em C ANSI portável em diversos sistemas operacionais;
- implementação de algoritmos presentes na literatura especializada compará-los com método proposto em testes de desempenho e usabilidade;
- contribuir para a produção de produtos científicos e tecnológicos, listados na seção 1.3.

1.3 Produção Científica

Como resultados dos métodos estudados e desenvolvidos, durante o desenvolvimento desta tese, foram publicados os seguintes trabalhos científicos:

1. **R. C. S. Costa**, P. C. Cortez, A. D. Bruno Costa, Contrast Enhancement Method for Movement Detection in Presentation Rooms, Interaction South America 2011, Belo-horizonte, 2011;

2. Freitas, R. F., **Costa, R. C. S.**, Cortez, P. C. Estudo comparativo de técnicas de detecção de cantos em imagens digitais. Revista Tecnologia (UNIFOR), v.31, p.263 - 278, 2010.

Além dos trabalhos científicos, foram geradas as seguintes produções tecnológicas:

1. **R. C. S. Costa** et al. Algoritmo De Fluxo Óptico Modificado Para Rastreamento De Deslocamentos Em Câmeras De Baixa Resolução, Patente PI0605829-9. Depósito: 28/12/2006. Publicação do pedido de patente: 19/08/2008;

2. **Costa, R. C. S.**; Cortez, P. C.; Soares, J. M.; Siqueira, R. S.; Leite Neto, C. F.; Silva, K. L. . Sistema Computacional para Detecção e Rastreamento de Objetos de Formato Específico e Cor pré-Determinada em Sequência de Imagens de Vídeo de Baixa Resolução. Depósito: 05/12/2008. Publicação do pedido de patente: 21/09/2010;

3. **R. C. S. Costa** et al. Sistema Classificador de Movimento Utilizando o Algoritmo de Fluxo Óptico Modificado em Câmeras de Baixa Resolução para Dispositivos Portáteis. Depósito: 18/12/2008. Publicação do pedido de patente: 21/09/2010;

4. Cortez, P. C.; **Costa, R. C. S.**; Soares, J. M.; Siqueira, R. S.; Leite Neto, C. F.; Mattos, C. L. C.; Silva, K. L.; Valente, I. R. S.; Ribeiro, F. C. Método de Rastreamento Facial e Identificação de Feições em Dispositivos Portáteis. Depósito: 18/12/2008. Publicação do pedido de patente: 14/09/2010;

5. Cortez, P. C.; **Costa, R. C. S.**; Soares, J. M.; Siqueira, R. S.; Leite Neto, C. F.; Freitas, R. F.; Barros, A. C. S.; Ribeiro, F. C. . Sistema de reconhecimento de gestos da mão utilizando visão artificial aplicado à interação com dispositivos portáteis. Depósito: 18/12/2008. Publicação do pedido de patente: 14/09/2010.

Além disto, como resultados adicionais do curso de doutorado, foram realizadas as seguintes orientações:

1. Co-orientação do trabalho de conclusão de curso do aluno **Alex Torquatro Carneiro** intitulado *Identificação e Localização de Pontos Críticos em Imagens Digitais* junto ao Curso de Graduação em Engenharia de Teleinformática da Universidade Federal do Ceará - UFC. 2007;

2. Orientação do trabalho de conclusão de curso do aluno **Rodrigo Fernandes Freitas** intitulado *Métodos de Detecção de Cantos em Tempo Real utilizando Câmeras de Baixa Resolução* junto ao Curso de Tecnologia Mecatrônica Industrial do Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE. 2009;

3. Co-orientação do trabalho de conclusão de curso do aluno **Raphael Torres Santos Carvalho** intitulado *Estudo Comparativo de Técnicas de Extração de Características para Reconhecimento de Fonemas* junto ao Curso de Graduação em Engenharia de Teleinformática da Universidade Federal do Ceará - UFC. 2009;

4. Orientação do aluno **Tiago da Silva Venâncio**, intitulado *Utilização da JNI para integração entre sistemas de computação gráfica C e Java* junto ao Curso de Sistemas de Informação junto à Faculdade Lourenço Filho. 2011.

E por fim, durante o doutorado foram gerados os seguintes artigos, a partir da interação com membros do grupo de pesquisa:

1. R. T. S. Carvalho, **R. C. S. Costa**, P. C. Cortez, Reconhecimento de Fonemas usando Predição Linear para movimento labial de modelos faciais humanóides virtuais, Interaction South America 2010, 2010, Curitiba.

2. R. F. Freitas, P. C. Cortez, **R. C. S. Costa**, A. C. S. Barros, Sistema de Rastreamento da Mão Humana Utilizando Visão Computacional para Aplicações Embarcadas, Interaction South America 2010, 2010, Curitiba.

3. R. F. Freitas, **R. C. S. Costa**, P. C. Cortez, Virtual Wheel: Proposta de Um Método de Interação Gestual para Jogos de Corrida, Interaction South America 2010, 2010, Curitiba.

4. **R. C. S. Costa**, A. C. S. Barros, F. R. Marciel, A. R de Alexandria, G. L. B. Ramalho, Avaliação de Métodos de Extração de Características para Inspeção Automática de Laranjas Destinadas à Produção de Suco Utilizando

Classificadores Não Paramétricos, XVIII Seminário de Computação, 2009, Blumenal.

5. A. T. S. Carneiro, P. C. Cortez, **R. C. S. Costa**, Correction of high lighting using histogram matching for skin segmentation on white background images, Interaction 09 - South America, 2009, São Paulo.

6. F. R. Marciel, A. C. S. Barros, R. F. Freitas, **R. C. S. Costa**, P. C. Cortez, J. M. Soares, M. G. L. Espinosa, Estudo Comparativo de Formas de Representação da Mão Humana para o Reconhecimento de Gestos Baseado em Redes Neurais Artificiais, Momentos de Hu e Atributos de Forma, XVIII Seminário de Computação, 2009, Blumenal.

7. A. T. S. Carneiro, P. C. Cortez, **R. C. S. Costa**, Reconhecimento de Gestos de LIBRAS com Classificadores Neurais a partir dos Momentos Invariantes de Hu, Interaction South America 09, 2009, São Paulo.

8. J. H. S. Felix, **R. C. S. Costa**, P. C. Cortez, M. A. Holanda, *Avaliação computacional de enfisema pulmonar em tomografia computadorizada: comparação dos resultados obtidos entre um sistema desenvolvido localmente e o programa Osiris.* Jornal Brasileiro de Pneumologia, 2009.

9. J. H. S. Felix, P. C. Cortez, A. R de Alexandria, **R. C. S. Costa**, M. A. Holanda, *Identification and Quantification of Pulmonary Emphysema through Pseudocolors. Lecture Notes in Computer Science*, v. 5317, p. 957-964, 2008;

10. J. H. S. Felix, P. C. Cortez, M. A. Holanda, **R. C. S. Costa**, *Automatic Segmentation and Measurement of the Lungs in healthy persons and in patients with Chronic Obstructive Pulmonary Disease in CT Images.* IFMBE Proceedings, v. 18, p. 370-373, 2007;

11. R. F. Freitas, **R. C. S. Costa**, A. C. S. Barros, R. S. Siqueira, P. C. Cortez, J. M. Soares, *Algoritmos para Segmentação da Pele Utilizando Modelo de Cores RGB em Ambiente Matlab/Simulink.* Conexões: Ciência e Tecnologia, v. 1, p. 65-71, 2007.

1.4 Organização do trabalho

Esta tese está organizada em sete Capítulos. O segundo capítulo apresenta uma retrospectiva de métodos de estimação de movimento. O Capítulo 3 realiza

uma revisão sobre sistemas interação baseados em VC, descrevendo as principais características e especificações deste tipo de sistema.

Com base nos conhecimentos adquiridos na revisão bibliográfica, no Capítulo 4 são descritos métodos propostos para realce e para detecção de movimentos com quatro graus de liberdade: três de translação (x,y e z) e um de rotação ao redor do eixo z.

O Capítulo 5 apresenta a metodologia utilizada para avaliar os métodos estudados. O Capítulo 6 descreve a análise e a discussão dos resultados obtidos através da metodologia proposta e finalmente, o Capítulo 6.8 apresenta as considerações finais e as perspectivas futuras.

Capítulo 2

Detecção de Movimentos

A percepção visual do movimento é uma ferramenta imprescindível para os seres humanos e animais extraírem informação relevante dentro de um cenário, possibilitando a realização de tarefas simples como atravessar uma rodovia, baseando-se no tempo de aproximação de um veículo que trafega nesta via ou até em cenários críticos como um veículo em sentido contrário (GONZALEZ; WOODS, 2010).

A Visão Computacional (VC) é uma área muito ampla relacionada com a obtenção de uma determinada tarefa, a qual está baseada na transformação de dados a partir de uma imagem estática ou de um vídeo em uma nova forma de representação. Atualmente, a aplicabilidade desta área tem crescido, sendo usada em uma grande variedade de aplicações desde vigilância, controle de tráfego até aplicações médicas (BRADSKI, 1998; GONZALEZ; WOODS, 2010).

Dentre as diversas áreas da VC, a percepção visual do movimento, também conhecida como detecção de movimentos, pode ser utilizada para localizar objetos em movimento e separá-los do fundo, resultando em uma imagem binária, mostrando apenas as regiões em movimento (DENMAN *et al.*, 2007; TRUYENQUE, 2005).

A capacidade de detectar objetos em movimento em vídeo é fundamental em muitos sistemas de visão computacional. Essa capacidade permite que os sistemas foquem a atenção nos objetos que estão em movimento e que, possivelmente, possuem uma importância fundamental na execução da tarefa para a qual são projetados (TRUYENQUE, 2005).

A detecção de movimento em uma sequência de imagens é um problema muito estudado nos últimos anos na área de VC, podendo ser realizada utilizando duas

metodologias: perseguição de um modelo ou independente de modelos.

Neste Capítulo são descritos os principais algoritmos de cada uma das categorias de métodos de estimação de movimentos. Inicialmente, descrevem-se os métodos baseado em rastreamento de modelos e em seguida, os métodos independentes de modelos.

2.1 Detecção baseada em rastreamento

A categoria de métodos baseados na perseguição de um modelo consiste em definir um objeto a ser rastreado, seja ele uma região, uma borda, um ponto e, a partir disto, são aplicadas técnicas de segmentação de imagens e extração de características para permitir, em seguida, rastrear o movimento do objeto (PARAGIOS; DERICHE, 1997).

Nessa categoria, destacam-se o rastreamento de objetos e a subtração de fundo. O primeiro é baseado em um conhecimento prévio do objeto e utiliza algoritmos de segmentação específicos (cor, textura, forma, etc.). Já o segundo, detecta os objetos que estão em movimento através da subtração entre o quadro atual e uma estimativa do fundo da imagem. Este método é muito sensível a pequenos movimentos do fundo e a mudança na iluminação, sendo muito utilizado para situações que possuem um fundo relativamente constante (GONZALEZ; WOODS, 2010).

Em ambos os casos, uma vez localizados os objetos em movimento, algumas informações devem ser extraídas dos mesmos, tais como, centro de massa ou qualquer característica de forma. Estas são utilizadas para o rastreamento, mas dependem do objeto a ser rastreado (SONKA *et al.*, 2008).

Dentro desta categoria de métodos destacam-se os trabalhos de Suryanto *et al.* (2011), Han *et al.* (2011) e Lee *et al.* (2012). Os dois primeiros realizam a detecção de movimentos de objetos selecionados previamente. O terceiro realiza navegação de robôs submarinos através da perseguição de objetos de cor vermelha. Já os trabalhos propostos por Yang *et al.* (2012), Xue *et al.* (2012) e Mandellos *et al.* (2011) realizam a detecção de objetos através da subtração de fundo.

Uma mudança no objeto a ser rastreado requer uma mudança dos algoritmos de segmentação ou de extração de características. Além disso, caso um objeto com características diferentes das previamente definidas se mova, seu movimento não é detectado.

Esses métodos possuem limitações relacionadas com o ambiente e com a necessidade da existência de um objeto específico presente na cena. Nesta tese é proposto um método de estimação de movimentos independentes do ambiente ou de objetos pré-determinados, sendo que os principais algoritmos dentro desta categoria estão descritos na seção a seguir.

2.2 Detecção Independente de Modelos

Uma grande vantagem desta estratégia é não depender do conhecimento prévio do objeto. Por causa disto, nesta abordagem é possível detectar o movimento de dois objetos com características diferentes que se movem em instantes de tempo distintos.

Nesta seção são descritos os principais algoritmos utilizados nesta abordagem, em que o movimento é estimado através de técnicas de correlação, do Fluxo Óptico (FO) (HORN; SCHUNCK, 1981; DENMAN *et al.*, 2007) e da Diferenciação Temporal (DT) entre dois instantes de tempo (GONZALEZ; WOODS, 2010; SOARES *et al.*, 2004).

Estes métodos são bastante eficientes em casos de grandes deslocamentos ou de objetos com textura. Contudo, quanto mais complexo o algoritmo, maior seu esforço computacional e, consequentemente, mais difícil de implementá-lo em tempo real, principalmente em dispositivos portáteis (PARAGIOS; DERICHE, 1997).

2.2.1 Diferença Temporal

A Diferenciação Temporal (DT) é uma abordagem que utiliza a diferença por *pixel* entre dois ou mais quadros (*frames*) em uma sequência de imagens para identificar os objetos em movimento. A diferença (d_{ij}) entre dois *frames* pode ser definida como

$$d_{ij}(x,y) = \begin{cases} 1, & \text{se } |I(x,y,t_j) - I(x,y,t_i)| > T, \\ 0, & \text{caso contrário,} \end{cases} \tag{2.1}$$

em que, $I(x,y,t)$ é a imagem em um dado instante de tempo t, t_i e t_j são dois instantes de tempos diferentes e T é um limiar específico. Neste caso, os *pixels*(x,y) que possuem d_ij igual a um (1) são considerados pertencentes aos objetos em movimento (WANG *et al.*, 2003; GONZALEZ; WOODS, 2010).

Em um ambiente controlado, com pouca variação de luminosidade, a diferença

entre os dois quadros sucessivos tem valores iguais ou muito próximos à zero nas regiões em que existe pouca ou nenhuma variação na intensidade luminosa, ou seja, o cenário de fundo permanece estático (SOARES *et al.*, 2004).

Esse tipo de método é pouco custoso computacionalmente e garante a detecção do movimento entre dois *frames* (MIGLIORE *et al.*, 2006; WANG *et al.*, 2003). Apesar disso, este método está sujeito a falhas causadas pela taxa de aquisição de quadros e a velocidade dos objetos como pode ser observado na Figura 2.1. Em vários momentos é possível observar o aparecimento de buracos dentro dos objetos em movimento, ou a detecção de movimento gerada pela saída do objeto.

(a) (b) (c)

Figura 2.1: exemplo de diferenciação no tempo, a) *frame* no instante t_0; b) no instante $t_0 + 1$; e c) resultado da diferença no tempo. Adaptada de Cheng e Chen (2006).

Outro problema do método em análise é quando um objeto em movimento pára. Consequentemente a região em movimento desaparece (objetos estáticos não produzem diferenças entre 2 quadros). Este problema pode ser resolvido com um controle da posição de regiões em movimento, assumindo que estas devem entrar e sair da cena.

Um parâmetro a ser ajustado em um sistema que utiliza DT é o valor de limiarização da diferença entre as duas imagens, conforme mostrado na Figura 2.2. Este limiar indica o que é realmente uma diferença entre as duas imagens. Tudo que for acima do limiar é considerado como objeto, e tudo que for abaixo do mesmo pode ser considerado como ruído, sendo, portanto, crítico para este método.

Valores muito baixos de limiar produzem diferenças por toda imagem, uma vez que duas imagens sucessivas não são perfeitamente iguais devido a erros introduzidos no processo de aquisição de imagens e variações na iluminação (SOARES *et al.*, 2004).

Uma forma de evitar alguns dos problemas anteriormente mencionados é realizar uma análise por região, ao invés de analisar as diferenças *pixel* a *pixel*, utilizando as técnicas baseadas em correlação.

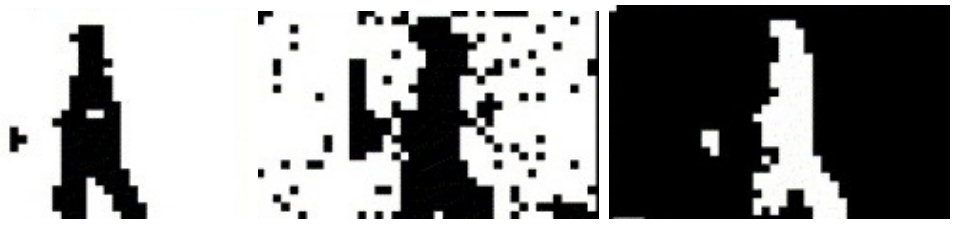


Figura 2.2: exemplo de diferenciação no tempo com diferentes condições de limiar, a) limiar baixo e b) limiar alto. Adaptado de Cheng e Chen (2006).

2.2.2 Correlação

Os métodos baseados na correlação tradicional procuram determinar o local em uma dada região, centrada no *pixel* (i,j), no instante de tempo $t-1$, é encontrada no instante de tempo atual t, podendo ser utilizado como elemento informação para a compressão de vídeo (MURATET *et al.*, 2005; YU *et al.*, 2010).

Cada bloco no instante de tempo atual t é comparado com todos os possíveis blocos dentro de uma janela de busca (com tamanho p x p) de tamanho pré-definido em $t-1$, utilizando métricas de erro, conforme ilustrado na Figura 2.3. O deslocamento (a,b) que possui maior similaridade, ou seja, menor erro, é utilizado como estimativa do movimento do bloco para o instante de tempo atual t (HUSSY, 1991).

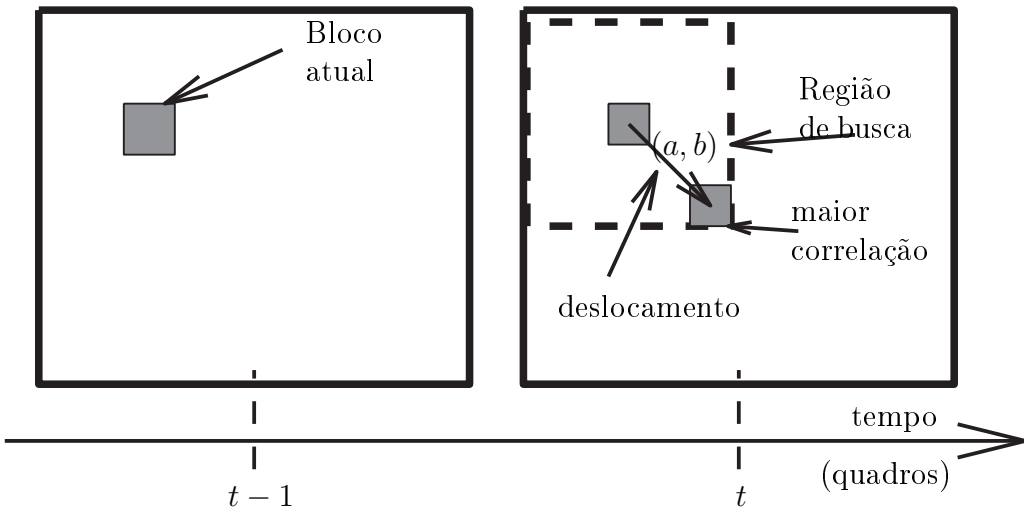


Figura 2.3: exemplo do método de correlação.

Os métodos baseados em correlação utilizam principalmente três medidas de erro: o Erro Médio Quadrático (MSE), Soma dos Quadrados das Diferenças (SSD)

e a Soma das Diferenças Absolutas (SAD). Estas medidas são determinadas por (YU *et al.*, 2010; BARJATYA, 2004)

$$MSE = \sqrt{\sum_{x}\sum_{y}\left(I(x,y,t) - I(x+a,y+b,t)\right)^2}, \tag{2.2}$$

$$SAD = \sum_{x}\sum_{y}\left|\left(I(x,y,t) - I(x+a,y+b,t)\right)\right|. \tag{2.3}$$

$$SSD = \sum_{x}\sum_{y}\left|\left(I(x,y,t) - I(x+a,y+b,t)^2\right)\right|. \tag{2.4}$$

Geralmente, este tipo de método é utilizado para estimar movimentos lineares. Contudo, através de modificações no cálculo de erro, outros tipos de movimentos podem ser detectados.

Os métodos de correlação são conhecidos por possuir um alto custo computacional, devido a grande quantidade de comparações realizadas para o cálculo do coeficiente de correlação. O mais simples dos métodos, também conhecido como busca completa, realiza a verificação de cada um dos pixels dentro da região de busca, o que totaliza p^2 comparações (SHENOLIKAR; NAROTE, 2009).

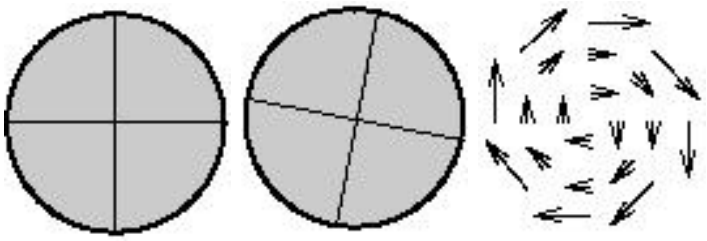
Para torná-lo viável computacionalmente, este cálculo é feito em uma janela de dimensões pequenas, o que restringe os métodos a movimentos suaves (MURATET *et al.*, 2005). Uma forma similar consiste em sub-amostrar a imagem e aplicar o mesmo procedimento com uma janela de busca com dimensão pequena, de forma a detectar movimentos menos suaves. Uma outra forma de melhorar a eficiência consiste em utilizar uma busca otimizada, como, os métodos de busca por três passos, de busca logarítmica ou de busca binária. Todas diminuem o tempo de busca às custas da redução da eficiência de detecção. Por exemplo, o método de busca por três passos é cerca de dez (10) vezes mais rápido que o busca completa. Contudo, estas otimizações são ineficientes para estimar movimentos de baixa velocidade (SHENOLIKAR; NAROTE, 2009; ROHS; ESSL, 2007).

Por causa disto, é necessário utilizar métodos com menor custo computacional para desenvolver um método de interação em sistemas embarcados como, por exemplo, os métodos de estimação do Fluxo Óptico (FO).

2.2.3 Estimação do Fluxo Óptico

O Fluxo Óptico (FO) é a distribuição das componentes de velocidade dos padrões de brilho em uma imagem, ou seja, é um campo de vetores de velocidade associado a uma sequência de imagens. Nesse sentido, o FO consiste em um campo denso de velocidade, em que a cada *pixel* no plano da imagem está associado um único vetor de velocidade (BARBOSA *et al.*, 2005).

Para fins de visualização, o campo é amostrado em uma malha e chamado de diagrama de agulhas. Um exemplo de visualização do FO é ilustrado na Figura 2.4. O primeiro e último *frames* do movimento de rotação no sentido horário de um círculo em torno do seu eixo, são mostrados nas Figuras 2.4(a) e 2.4(b). O FO do movimento é mostrado na Figura 2.4(c). Observa-se uma distribuição radial dos vetores de velocidade, em que os vetores de mesmo módulo formam um círculo no sentido horário e à medida que os vetores vão se distanciando do centro, seu módulo também aumenta.



(a) (b) (c)

Figura 2.4: exemplo de estimação de FO. a) *frame* no instante t_0, b) *frame* no instante $t_0 + 1$ e c) FO. Adaptado de Sonka *et al.* (2008).

Se for conhecido o intervalo de tempo entre duas imagens consecutivas, os vetores de velocidade podem ser convertidos em vetores de deslocamento e vice-versa (BARBOSA *et al.*, 2005).

O FO contém informações importantes utilizadas em diversas aplicações como segmentação de regiões ou bordas, análise de formas, interpretação de cena, navegação exploratória, acompanhamento de objetos, avaliação de tempo para colisão, codificação de vídeo, visão de robôs, etc (HORN; SCHUNCK, 1981; LI, 2000; BARBOSA *et al.*, 2005; FERNÁNDEZ-CABALLERO *et al.*, 2010; TRACKING, 2012).

Os métodos de estimação de FO fornecem meios de determinar e representar o movimento em uma sequência de imagens. Os métodos para a computação do FO

podem ser classificados em três principais grupos de técnicas: correlação, diferenciais e baseadas em frequência/energia (BEAUCHEMIN; BARRON, 1995). Esta última categoria é computacionalmente complexa por analisar as imagens no domínio da frequência, não sendo tratado nesta tese.

Apesar das diferenças entre os métodos, os algoritmos para estimação FO podem ser divididos conceitualmente em três estágios de processamento (BARRON; KLETTE, 2002):

i) pré-filtragem com filtros passa-baixas ou passa-banda para extrair os sinais de interesse ou melhorar a relação sinal ruído, podendo ser aplicados no domínio do tempo ou no domínio da frequêcia;

ii) extração de medidas básicas como, por exemplo, as derivadas espaço-temporais ou superfícies locais de correlação (similar ao descrito na seção 2.2.2);

iii) processamento das medidas para produzir o campo de velocidades.

Nas técnicas diferenciais, a hipótese inicial para a computação do FO considera que a intensidade do brilho entre quadros diferentes em uma sequência de imagens é aproximadamente constante em um intervalo de tempo pequeno (BARBOSA *et al.*, 2005; PARAGIOS *et al.*, 2005; HORN; SCHUNCK, 1981). Nos métodos diferenciais, a velocidade é estimada utilizando derivadas espaço-temporal da imagem original ou realçada utilizando filtros passas-baixas ou passa-bandas (BEAUCHEMIN; BARRON, 1995).

Neste caso, a hipótese inicial na medição de movimento de imagem é que o padrão de brilho da cena (I) permaneça constante numa região da imagem, num curto espaço de tempo. Assim, definindo $I = (\vec{x}, t)$, formalmente tem-se (HORN; SCHUNCK, 1981)

$$I(\vec{x}, t) = I(\vec{x} + \delta\vec{x}, t + \delta t), \tag{2.5}$$

em que $\vec{x} = (x, y)^T$ é o vetor de posição, t é o tempo, $\delta\vec{x}$ é um pequeno deslocamento em uma região da imagem, δt é a diferença temporal e $(\cdot)^T$ é o operador de transposição de matriz. Em outras palavras, em um pequeno intervalo de tempo o deslocamento é desprezível. Expandindo a parte direita da equação 2.5 através da

série de Taylor, resulta em (BEAUCHEMIN; BARRON, 1995)

$$I(\vec{x},t) = I(\vec{x},t) + \nabla I \cdot \partial\vec{x} + \partial t I_t + O^2, \tag{2.6}$$

em que $\nabla I = (I_x, I_y)$ e I_t são as derivadas parciais de primeira ordem de $I(\vec{x},t)$, i.e., o gradiente espacial da intensidade, e O^2 são os termos de maior ordem, os quais assume-se que são desprezíveis. Subtraindo o termo $I(\vec{x},t)$ de ambos os lados, ignorando os termos de maior ordem (O^2) e dividindo os termos restantes por ∂t, resulta em

$$\nabla I \cdot \frac{\partial x}{\partial t} + I_t = 0 \Leftrightarrow \nabla I \cdot \vec{v} + I_t = 0 \tag{2.7}$$

no qual $\vec{v} = \frac{\partial x}{\partial t}(v_x, v_y)$ é o campo de velocidade e v_x e v_y são as velocidades nas direções x e y, respectivamente. Esta equação é conhecida como **equação de restrição do movimento** (BEAUCHEMIN; BARRON, 1995; HORN; SCHUNCK, 1981).

Para um movimento 2D, a equação 2.7 pode ser re-escrita por

$$I_x \cdot v_x + I_y \cdot v_y + I_t = 0, \tag{2.8}$$

em que I_x e I_y são, respectivamente, os gradientes nas direções x e y.

É importante destacar que a equação 2.7 isoladamente não é capaz de determinar unicamente as velocidades v_x e v_y, pois, a mesma possui duas incógnitas. Então, esse problema tem infinitas soluções, que podem ser encontradas em qualquer lugar da linha pontilhada mostrada na Figura 2.5, que é perpendicular ao gradiente ∇I.

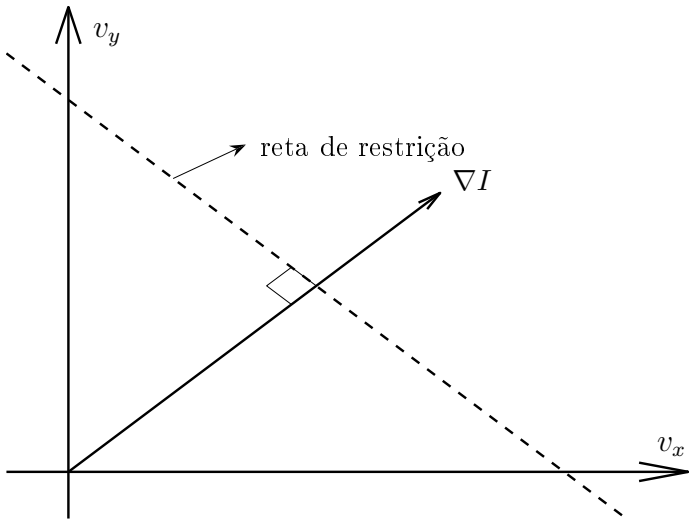


Figura 2.5: reta de restrição de FO.
Fonte: Horn e Schunck (1981).

Neste caso, é necessária a inserção de restrições especiais através de novas equações para conseguir estimar as componentes de velocidade. Por exemplo, pode-se considerar como restrição adicional que o fluxo óptico é constante em uma região consistindo de um grupo. Outros exemplos são considerar que as derivadas do fluxo óptico são constantes, que a luminância é constante em uma dada superfície, qualquer que seja a direção de observação, ou que é realizado apenas um movimento de translação paralelo ao plano da imagem (BEAUCHEMIN; BARRON, 1995; BARBOSA *et al.*, 2005).

Na literatura especializada, cada um dos trabalhos desenvolvidos por Horn e Schunck (1981), Nagel (1983), Nagel (1987), Lucas e Kanade (1981) e Fleet e Jepson (1990) impõem uma restrição adicional para resolver esse problema, tornando possível determinar a velocidade.

2.2.4 Método de Horn e Schunck (1981)

Horn e Schunck (1981) foram os primeiros a impor uma restrição adicional à equação 2.7, que consiste em um considerar que os vetores de fluxo alteram-se de maneira suave ao longo da imagem, ou seja, um suavizador global. Desta forma, em um determinado objeto opaco, os pontos que o compõem possuem velocidades semelhantes (LAUREANO; PAIVA, 2006).

A restrição apresentada, conhecida como restrição de suavidade do fluxo óptico, tem por objetivo limitar o campo de velocidade estimado, minimizando a seguinte expressão (BEAUCHEMIN; BARRON, 1995)

$$\int_D \left(I_x\delta_x + I_y\delta_y + I_t\right)^2 + \lambda^2 \left(\|v_x\|_2^2 + \|v_y\|_2^2\right) dx, \tag{2.9}$$

calculados sobre um domínio de interesse (D), em que $\overrightarrow{v} = (v_x, v_y)$, λ é um fator de suavização, e $\|v_x\|$ e $\|v_y\|$ são os módulos das velocidades nas direções x e y. A solução do problema é realizada através de um conjunto de equações de Gauss-Seidel, resolvidas interativamente e resultam nas seguintes expressões

$$v_x^{k+1} = \overline{v_x}^k - I_x \frac{I_x\overline{v_x}^k + I_y\overline{v_y}^k + I_t}{\lambda^2 + I_x^2 + I_y^2}, \tag{2.10}$$

$$v_y^{k+1} = \overline{v_y}^k - I_y \frac{I_x\overline{v_x}^k + I_y\overline{v_y}^k + I_t}{\lambda^2 + I_x^2 + I_y^2}, \tag{2.11}$$

em que $k+1$ representa a próxima interação (a ser calculada), k é a última interação calculada, $\overline{v_x}$ e $\overline{v_y}$ são os valores médios da velocidade dos vizinhos do *pixel* em que é calculada a velocidade nas direções x e y (BEAUCHEMIN; BARRON, 1995).

A partir de duas imagens consecutivas, o método de computação do FO proposto por Horn e Schunck (1981) realiza uma série de interações a fim de estimar as componentes de velocidade. Inicialmente, o algoritmo considera que as componentes de velocidade inicialmente são nulas e a cada interação, as componentes de velocidade são recalculadas usando as equações 2.10 e 2.11. O processo é repetido até que a diferença de velocidade estimada seja menor do que um certo limiar ou atingir um número máximo de interações, conforme descrito no Algoritmo 2.1.

Algoritmo 2.1 Algoritmo de Horn e Schunck (1981).

Entrada: $v_x = v_x(t)$ e $v_y = v_y(t)$ as componentes de velocidades de um pixel (x, y) em um dado instante de tempo t

Entrada: I_T, I_x e I_y os gradientes temporais e espaciais calculados entre os instantes de tempo t e $t-1$

Entrada: MAX é o número máximo de interações permitidas

$v_x^0 \leftarrow 0$ {Velocidade na interação inicial (0)}
$v_y^0 \leftarrow 0$
$k \leftarrow 0$
repetir
  $\overline{v_x} \leftarrow media(v_x)$
  $\overline{v_y} \leftarrow media(v_y)$
  $v_x^{k+1} \leftarrow \overline{v_x}^k - I_x \frac{I_x \overline{v_x}^k + I_y \overline{v_y}^k + I_t}{\lambda^2 + I_x^2 + I_y^2}$
  $v_y^{k+1} \leftarrow \overline{v_y}^k - I_y \frac{I_x \overline{v_x}^k + I_y \overline{v_y}^k + I_t}{\lambda^2 + I_x^2 + I_y^2}$
  $k \leftarrow k + 1$
até $v_x^k - v_x^{k-1} < \delta$ **e** $v_y^k - v_y^{k-1} < \delta$ **ou** $k < MAX$

2.2.5 Método de Lucas e Kanade (1981)

A restrição adicional para a equação 2.7 proposta por Lucas e Kanade (1981) é um método não iterativo que assume um FO local constante, isto é, o fluxo é constante em pequenas janelas. Assim, todos os *pixels* dentro da janela possuem o mesmo deslocamento, conforme mostrado na Figura 2.6.

Da forma proposta, a imagem é dividida em janelas de tamanho N x N, cada uma com $p = N^2$ *pixels*. A restrição de movimento é usada para formar um sistema sobre determinado conjunto de p equações com 2 variáveis, substituindo os valores

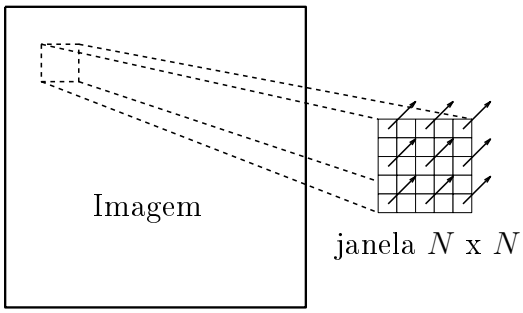


Figura 2.6: restrição proposta por Lucas e Kanade (1981). Adaptado de Aires *et al.* (2008).

de intensidade em cada *pixel* da janela na equação 2.8, resulta em (AIRES *et al.*, 2008)

$$\begin{aligned} I_{x_1} \cdot v_x + I_{y_1} \cdot v_y + I_{t_1} &= 0, \\ I_{x_2} \cdot v_x + I_{y_2} \cdot v_y + I_{t_2} &= 0, \\ &\cdots \\ I_{x_p} \cdot v_x + I_{y_p} \cdot v_y + I_{t_p} &= 0, \end{aligned} \quad . \tag{2.12}$$

que na forma matricial produz

$$\underbrace{\begin{bmatrix} I_{x_1} & I_{x_2} & \cdots & I_{x_p} \\ I_{y_1} & I_{y_1} & \cdots & I_{y_p} \end{bmatrix}^T}_{A} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \underbrace{\begin{bmatrix} -I_{t_1} & -I_{t_2} & \cdots & -I_{t_p} \end{bmatrix}}_{-B}. \tag{2.13}$$

O sistema de equações descritos nesta equação pode ser resolvido utilizando o método dos Mínimos Quadrados (LMS - Mínimos Quadrados), através de (BARRON; KLETTE, 2002)

$$\mathbf{A}^T \cdot \mathbf{A} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \mathbf{A}^T \cdot (-\mathbf{B}), \tag{2.14}$$

sendo que a estimação do FO pode ser obtida através do cálculo da pseudo inversa

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \left(\mathbf{A}^T \cdot \mathbf{A}\right)^{-1} \cdot \mathbf{A}^T \cdot (-\mathbf{B}), \tag{2.15}$$

podendo esta equação ser reescrita como

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \underbrace{\begin{bmatrix} \sum_{i=1}^{p} I_{x_i}^2 & \sum_{i=1}^{p} I_{x_i} I_{y_i} \\ \sum_{i=1}^{p} I_{x_i} I_{y_i} & \sum_{i=1}^{p} I_{y_i}^2 \end{bmatrix}^{-1}}_{\mathbf{C}} \underbrace{\begin{bmatrix} -\sum_{i=1}^{p} I_{x_i} I_{t_i} \\ -\sum_{i=1}^{p} I_{y_i} I_{t_i} \end{bmatrix}}_{\mathbf{D}}. \tag{2.16}$$

Esta equação possui uma operação de inversão de matriz. Para fins de implementação, é necessário verificar se esta matriz é não-singular. Para isto, calculam-se os autovalores da matriz $\mathbf{C}$ (λ_1 e λ_2). Quando os autovalores são menores do que um limiar de redução de ruído (τ), o FO não pode ser estimado e considera-se que as componentes de velocidade v_x e v_y são nulas (BARRON; KLETTE, 2002).

Com base no conhecimento dos principais métodos de detecção de movimentos, é possível compreender o funcionamento de um sistema de interação baseados em técnicas de VC para computadores pessoais e para sistemas embarcados, descritos no próximo capítulo.

Capítulo 3

Interfaces de Interação Humano-Computador (IHC) por Visão Computacional

A Interação Humano-Computador (IIHC) é uma área multidisciplinar relacionada aos aspectos relacionados com a interação entre usuários e sistemas e está preocupada com o design, avaliação e implementação de sistemas interativos usáveis, seguros e funcionais. Outro assunto relacionado à IIHC é o desenvolvimento de um componente responsável por mapear a entrada de dados do usuário em ações realizadas no sistema (ROCHA, 2003).

Por muito tempo, a pesquisa em Interfaces de Interação Humano-Computador (IHC) tem sido restrita à utilização do monitor, do teclado e do mouse. Os dispositivos de interação são tradicionalmente construídos utilizando diferentes dispositivos de rastreamento óptico, magnético ou mecânico ligados ao computador e/ou colocados no corpo do usuário (TRUYENQUE, 2005), conforme está exemplificado na Figura 3.1.

Figura 3.1: interfaces de interação tradicional. Adaptado de Truyenque (2005).

Nas interfaces de interação tradicionais, a presença de um dispositivo físico é importante para traduzir os comandos do usuário para o computador. Se o usuário manipula o dispositivo (por exemplo, muda sua posição) o computador recebe o sinal da mudança e manipula o objeto lógico ligado ao dispositivo (TRUYENQUE, 2005).

Em certas situações, as interfaces tradicionais não são capazes de satisfazer a necessidade dos usuários de interagirem livremente com os computadores sem a necessidade de suportes ou cabos (ZHOU *et al.*, 2007). Por exemplo, a interação de uma pessoa com telas projetadas em dispositivos tradicionais é limitada devido a existência de fios.

Uma solução para esse tipo de problema é utilizar metodologias que eliminem a presença desses dispositivos. Por exemplo, ao utilizar o movimento da mão para realizar ações na tela ou utilizar o movimento do dispositivo torna a forma de interação mais natural (TRUYENQUE, 2005).

Os sistemas operacionais e jogos portáteis são aplicações que podem ser melhor controlados com este tipo de técnicas de interação inovadoras, por causa disto, nos últimos anos tem crescido a quantidade de trabalhos que se destinam à desenvolvimer de métodos de interação baseados em Visão Computacional (VC) para dispositivos portáteis (HANNUKSELA *et al.*, 2005).

Apesar dos rápidos avanços tecnológicos, os dispositivos portáteis dificilmente terão um desempenho equivalente a um Computador Pessoal (PC) (CHEN *et al.*, 2007). De uma maneira geral, os dispositivos /portáteis apresentam restrições nos recursos computacionais disponíveis, nos custos ou mesmo nas condições de operação, limitando sobremaneira as técnicas que podem ser utilizadas.

Uma solução alternativa para solucionar o problema da capacidade de processamento é enviar a imagem para um PC com alta capacidade de processamento com o objetivo de que este processe a imagem. Contudo, por causa da latência na rede (tempo para enviar e receber a informação processada) e da largura de banda disponível para *upload*, este tipo de aplicação se torna inviável para aplicações em tempo real. Desta forma, uma solução para aplicar as técnicas de VC embarcadas em dispositivos portáteis requer adaptações ou um melhor conhecimento da arquitetura utilizada para conseguir obter o máximo de desempenho (CHEN *et al.*, 2007; HANNUKSELA *et al.*, 2007).

Uma limitação adicional ao emprego de VC na detecção e na correta localização de objetos em dispositivos portáteis é resolução da imagem. Em sua maioria, as câmeras são de baixa resolução que dificultam a tarefa de detectar objetos na imagem. Isto faz com que a pesquisa para desenvolver métodos para esta área seja mais precisa para favorecer o desenvolvimento de novas tecnologias que possam chegar ao mercado consumidor.

Por causa destas inúmeras resrições, existem poucos trabalhos relacionados à detectar o movimento do próprio dispositivo através de VC que simulam o movimento do mouse. Contudo, todas possuem limitações como, por exemplo, depender de objetos pré-determinados ou requisitar uma fase prévia de inicialização de sistema (PYROFFERS, 2007; GRUNNET-JEPSEN *et al.*, 2006; BALLAGAS *et al.*, 2005; BETKE *et al.*, 2002; GORODNICHY *et al.*, 2004; CROWLEY; COUTAZ, 1995).

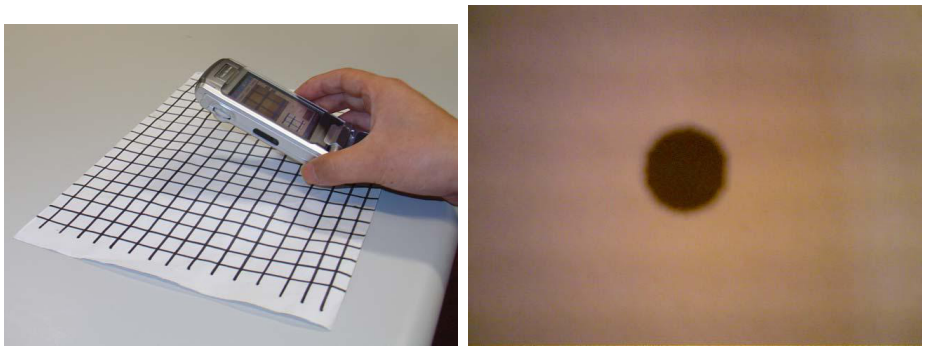
Neste capítulo são descritos os principais métodos de interação com dispositivos portáteis baseado em VC. Inicialmente, são descritos os métodos dependentes de modelos pré-determinados, conhecidos como marcadores e, em seguida, são descritos os métodos independente de modelos.

3.1 Técnicas Baseadas em Marcadores

Os sistemas de interação baseado em VC podem utilizar o conhecimento prévio do objeto a ser rastreado, similarmente ao que fora descrito na seção 2.1. Nesta seção, são descritos os métodos presentes na literatura especializada que utilizam padrões pré-definidos, conhecidos como marcadores, códigos ou padrões visuais Grunnet-Jepsen *et al.* (2006), Hachet *et al.* (2005), Qin (2006), Manduchi *et al.* (2008) e Ballagas *et al.* (2005) .

O trabalho proposto por Qin (2006) utiliza dois tipos de padrão desenhados em uma folha em branco. O primeiro consiste em uma malha de quadrados, enquanto que o segundo é um círculo preto, mostrados na Figura 3.2. Estes padrões possuem um alto contraste com o papel, facilitando a utilização de um método de limiarização. Contudo, requer que o dispositivo portátil esteja próximo ao papel.

Atualmente, o *Quick Response Code* (QR *Code*) mostra-se muito popular de diversos usuários da plataforma Android. Este é um dos tipos de marcadores capazes de armazenar uma informação dentro de um marcador visual 2D, mostrados na Figura 3.3.



(a) (b)

Figura 3.2: códigos visuais utilizados no trabalho de Qin (2006). Adaptado de Qin (2006).

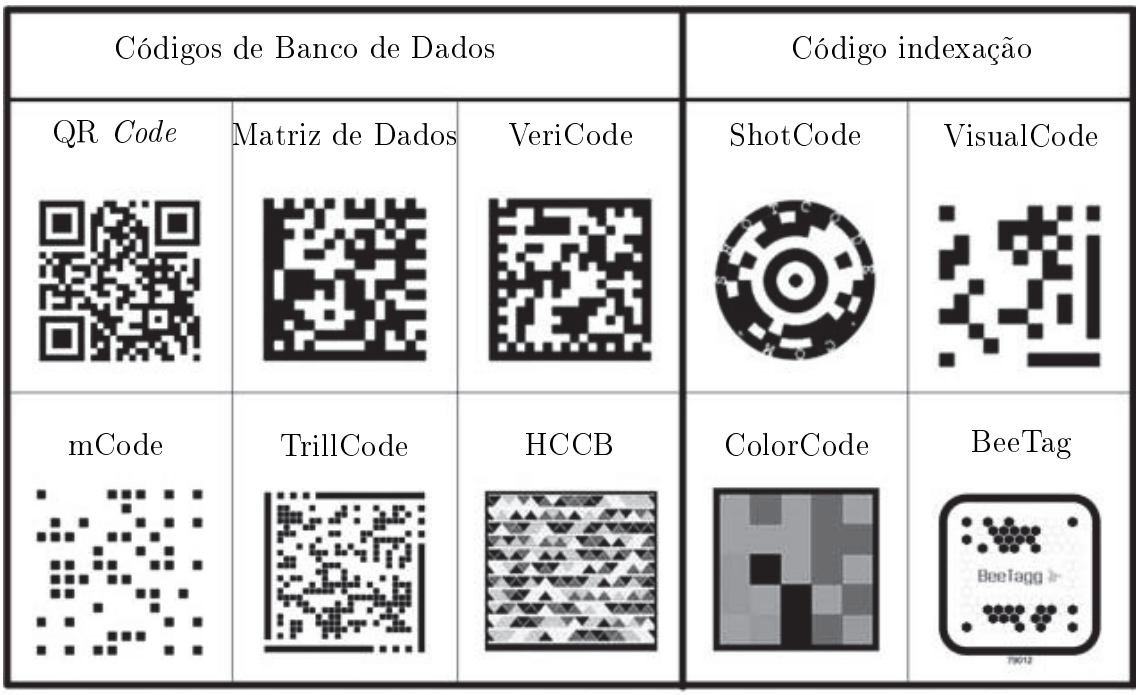


Figura 3.3: exemplos de códigos de barras usados em aplicações para dispositivos portáteis. Fonte: Korato *et al.* (2010).

Apesar deste existir desde 1994, apenas nos últimos anos este tipo de marcador tem se tornado popular no Brasil. Os QR *Code*s armazenam, geralmente, um endereço de uma página web, sendo indispensáveis para dispositivos portáteis com acesso à internet (KORATO *et al.*, 2010). Contudo, o mesmo é utilizado apenas como fonte de informação, não sendo geralmente utilizado como sensor de movimento.

Dentre os diversos tipos, os códigos visuais (*visual codes*) desenvolvidos por Ballagas *et al.* (2005) podem armazenar informações e possibilitam a determinação da posição relativa da câmera em relação ao padrão. O princípio de funcionamento

deste código visual, as características reconhecidas pelo método e um exemplo de aplicação são ilustrados, respectivamente, nas Figuras 3.4 e Figuras 3.5.

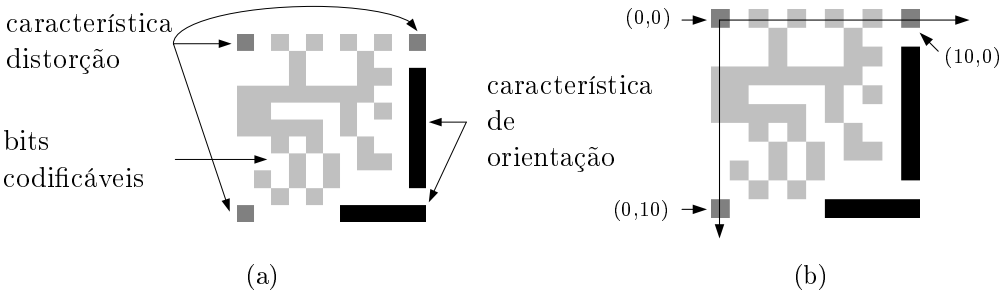


Figura 3.4: código visual (*visual codes*) a) características e b) sistemas de coordenadas. Adaptado de Rohs e Zweifel (2005), Rohs (2007).

Os pontos coloridos em cinza escuro mostrados na Figura 3.4(a) definem o sistema de coordenadas, mostrado na Figura 3.4(b). Já as linhas pretas identificam a orientação do código que permitem, através do seu reconhecimento, determinar a distância, orientação e a inclinação, conforme mostrado na Figura 3.5(a). Além disso, o código visual, colorido em cinza claro, pode utilizar até 83 bits para codificar a informação. A localização deste tipo de código visual e a identificação da informação codificada é feita através de técnicas de baixo custo computacional, tais como a limiarização e as transformações geométricas.

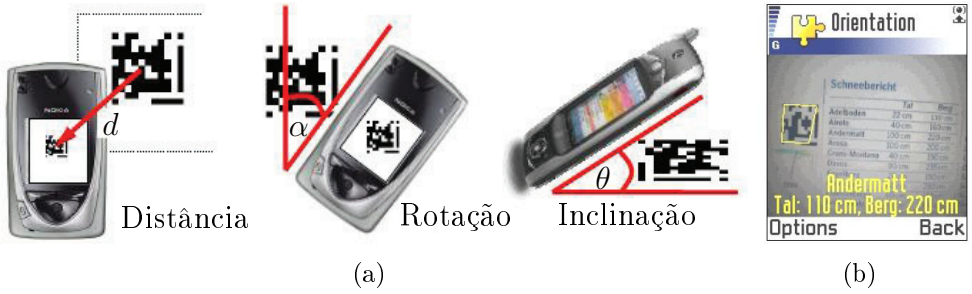


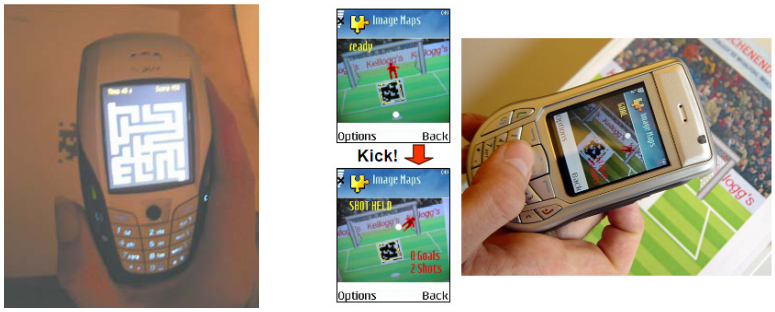
Figura 3.5: reconhecimento de informações através do código visual, a) tipos de movimentos reconhecidos; e b) exemplo do funcionamento. Adaptado de Rohs e Zweifel (2005), Rohs (2007).

Com base nas informações adquiridas, mostradas na Figura 3.4(a), é possível determinar um sistema de interação com quatro graus de liberdade (três de translação e um de rotação), conforme mostrado na Figura 3.5(a) (BALLAGAS *et al.*, 2005).

Um exemplo de aplicação deste padrão visual, mostrado na Figura 3.5(b), é identificar informações dentro de uma tabela presentes em um servidor de

informações. A posição, em que o padrão é encontrado, indica a linha e a orientação/rotação determina qual coluna da tabela a ser exibida na tela (ROHS; ZWEIFEL, 2005).

Devido à alta taxa de acerto e ao baixo custo computacional, esse método é utilizado como sistema de interação para jogos em telefones celulares, descritos por Bucolo *et al.* (2005) e Rohs (2007), mostrados respectivamente nas Figuras 3.6(a) e 3.6(b). O primeiro utiliza o código visual para controlar um jogo



(a) (b)

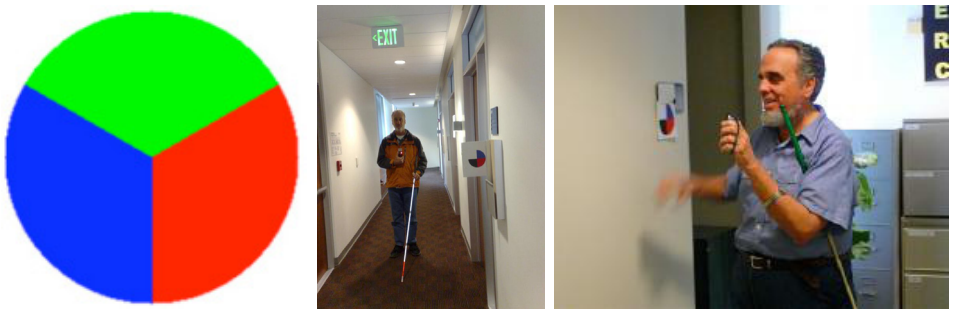
Figura 3.6: aplicações dos marcadores visuais em jogos de dispositivos portáteis a) labirinto portátil (BUCOLO *et al.*, 2005) e b) cobrança de pênaltis. Adaptado de Rohs (2007).

de labirinto, enquanto que o segundo utiliza os códigos visuais para vários jogos de realidade virtual aumentada.

Trabalho similares, propostos por Manduchi *et al.* (2008) e Chan *et al.* (2007), utilizam um marcador circular colorido, mostrado na Figura 3.7(a) com diâmetro de 12 cm impressos em um papel em branco. O objetivo deste é auxiliar deficientes visuais a se guiarem em corredores e a localizarem salas e objetos utilizando um celular Nokia N95, conforme mostrado na Figura 3.7.

Em ambos os casos, é utilizada uma segmentação baseada em limiarização adaptativa. Em seguida, o objeto é rastreado, tornando capaz de simular o movimento do mouse em duas dimensões. No primeiro padrão é rastreado os cantos da malha do quadrado e no segundo o centro de massa. O sistema de interação é testado em um aparelho celular Sony Ericson P910i. Contudo, o trabalho não apresenta uma avaliação sobre a eficiência e precisão do rastreamento, bem como, o uma análise de esforço computacional dos métodos analisados.

Um fator limitante de todas estas técnicas é a dependência completa do marcador visual para que o dispositivo opere em qualquer tipo de ambiente. Além disto,



(a) (b) (c)

Figura 3.7: exemplo de utilização do marcador visual colorido para auxílio a deficiente visual a) marcador a ser localizado; b) utilização por um cego em um corredor e; c) localização da sala desejada. Fonte: Chan *et al.* (2007), Manduchi *et al.* (2008).

para que seja possível a detecção de movimentos, a câmera tem que estar à uma distância pequena do local onde está impresso o marcador, pois só assim, seus detalhes continuam visíveis. Tais restrições contribuíram para o desenvolvimento de outras técnicas alternativas.

3.1.1 Técnicas Baseadas em Segmentação de Pele

Na busca de propostas de interação que sejam independentes da existência de marcadores, nesta seção são descritos métodos que utilizam a métodos baseados em segmentação de pele (HANSEN, 2005) ou detecção de bordas (GALLO *et al.*, 2008).

O método proposto por Gallo *et al.* (2008) adquire o *frame* no modelo de cor Formato de Cor *Red Green Blue* (RGB) e, em seguida, calcula o valor da cromaticidade vermelha $Cr(x, y)$ do modelo YCbCr. Logo após, são calculados as projeções de gradiente horizontal $gradY$ e vertical $gradX$, dados por (GALLO *et al.*, 2008)

$$gradY(c) = \sum_{i} |Cr(i, c) - Cr(i - n, c)|, \tag{3.1}$$

$$gradX(l) = \sum_{j} |Cr(l, j) - Cr(l, j - n)|, \tag{3.2}$$

em que i e j são as coordenadas e n é a distância, em *pixels*. É usado um valor de $n > 1$ por causa da transição entre o fundo e o objeto não ser abrupta devido ao borramento causado pelas características da câmera ou movimento.

A partir das projeções calculadas anteriormente, a ponta do dedo é localizada através da detecção do primeiro ponto de máximo em $gradY$ e $gradX$, representando

o ponto mais acima e mais à direita do quadrado envolvente (desenhado em linhas pontilhadas), conforme mostrado na Figura 3.8.

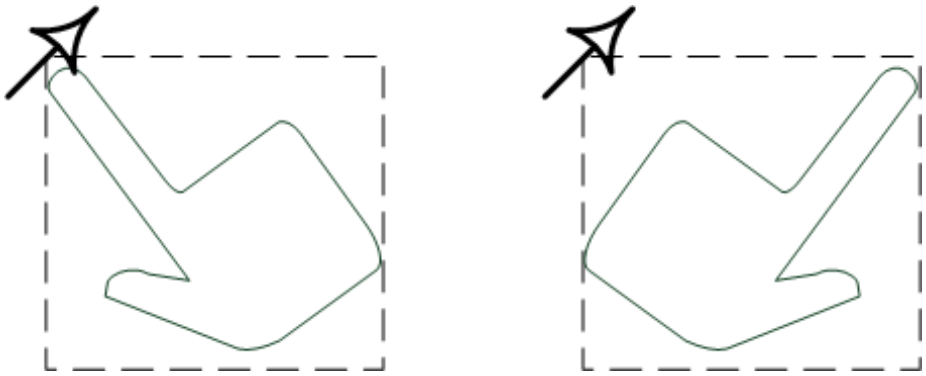


Figura 3.8: localização através do método proposto por Gallo *et al.* (2008) na mão; a) direita; e b) esquerda. Adaptado de Gallo *et al.* (2008).

Uma grande desvantagem desse método é requer que a mão do usuário esteja posicionada conforme mostrado na Figura 3.8(a), limitando o uso para apenas a mão direita, pois, é cansativo posicionar a mão esquerda da forma desejada. Caso o usuário coloque a mão esquerda, o ponto é localizado fora da região da mão, conforme mostrado na Figura 3.8(b). Desta forma, é necessário alterar o código para reconhecer a outra mão.

Outro sistema de interação proposto por Hansen (2005) realiza a detecção e o rastreamento do movimento da cabeça do usuário, utilizando a câmera do dispositivo portátil empregada em vídeo-chamadas, conforme mostrado na Figura 3.9.

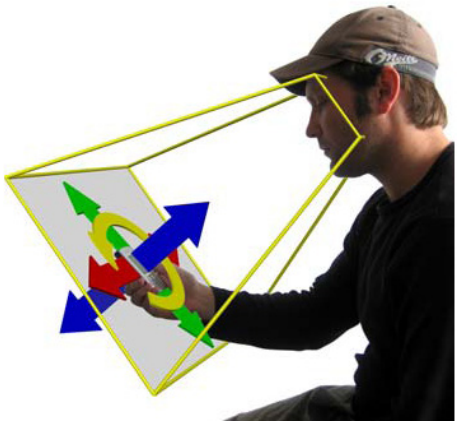


Figura 3.9: interação com dispositivos portáteis através da detecção da pose da cabeça. Adaptado de Hansen (2005).

No seu trabalho, Hansen (2005) utiliza dois métodos de rastreamento: um baseado em Transformada de Hough (XU *et al.*, 1990) e outro baseado no *Continuously Adaptive Mean Shift* - CAMShift (BRADSKI, 1998). Contudo, o autor não descreve os custos computacionais dos algoritmos e não realiza nenhuma análise da eficiência da detecção de movimento, sendo um artigo que apenas apresenta a proposta do uso do método.

3.1.2 Técnicas Baseadas no Movimento do Dispositivo

Uma forma mais natural para interagir com um dispositivo portátil é movê-lo no ar e utilizar o movimento realizado para controlar as aplicações no mesmo (HANNUKSELA *et al.*, 2006), conforme mostrado na Figura 3.10.

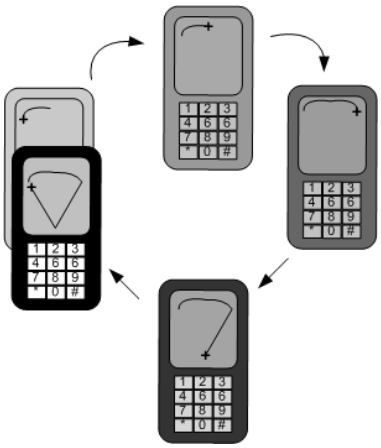


Figura 3.10: forma natural de interação com dispositivos portáteis.

O trabalho de Hildreth (2006) realiza a detecção de cantos e bordas e, em seguida, realiza o casamento dos objetos detectados em quadros (*frames*) adquiridos em instantes de tempos distintos. Com base na quantidade de pontos detectados, é possível determinar de 1 a 6 movimentos realizados pelo dispositivo portátil.

Dentre os diversos tipos de algoritmos, uma outra abordagem é proposta por Mory (2000), Drab e Artner (2005) e por Hannuksela *et al.* (2007) que realizam a detecção de movimentos a partir do movimento da câmera (também conhecido como *egomotion*).

O trabalho proposto por Mory (2000) utiliza como base a estimação do movimento baseado no método de casamento de blocos para detectar o movimento realizado pela câmera. Este método consiste na extensão do método de correlação (seção 2.2.2) que divide a imagem em blocos e, para todos estes, realiza a

determinação do deslocamento que possui menor correlação.

O objetivo deste é fornecer descritores de contexto para a codificação MPEG-7 (*Moving Picture Experts Group*). Este algoritmo é, comumente, utilizado para estimar o vetor de movimento de blocos entre quadros em vídeos nos formatos MPEG-1 e MPEG-2.

O método proposto por Mory (2000) modela o movimento em três componentes translacionais (T_x, T_y, T_z) e três rotacionais(R_x, R_y, R_z), conforme mostrado na Figura 3.11 e descrito por

$$\begin{aligned} \bar{X} &= -T_x + R_y \cdot Z + R_z \cdot Y, \\ \bar{Y} &= -T_y + R_z \cdot X + R_x \cdot Z, \\ \bar{Z} &= -T_z + R_x \cdot Y + R_y \cdot X, \end{aligned} \tag{3.3}$$

em que (X, Y, Z) e $(\bar{X}, \bar{Y}, \bar{Z})$ são as coordenadas de um ponto no plano da imagem no instante de tempo anterior e no atual, respectivamente. Através de uma série de cálculos, estima-se uma função de custo que determina os valores mais prováveis das

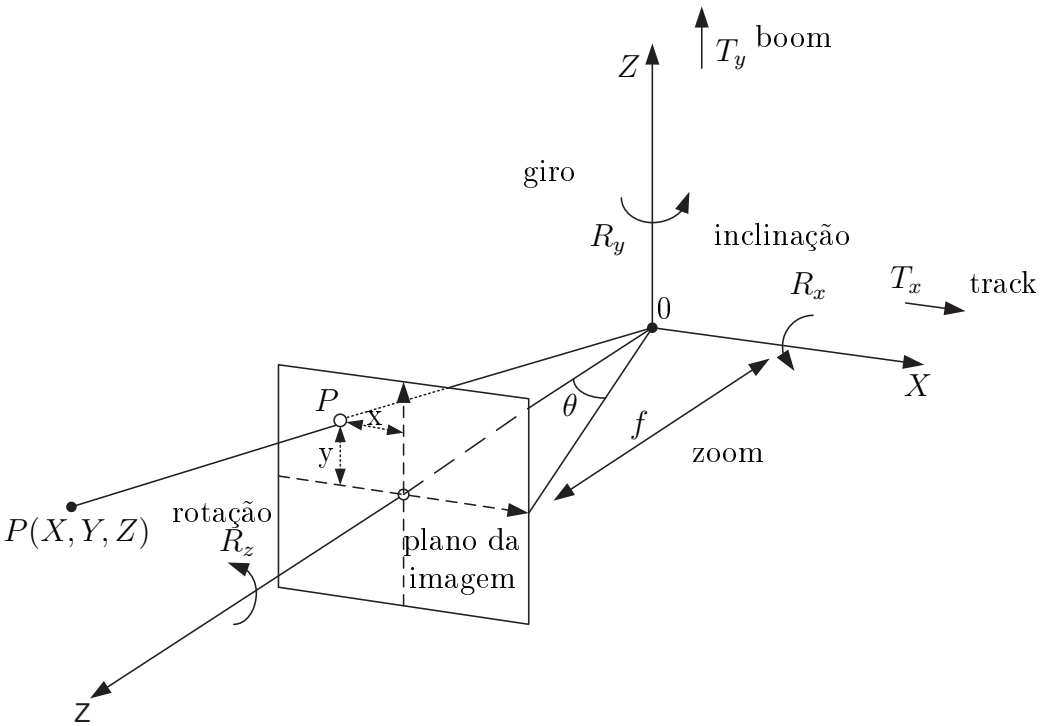


Figura 3.11: movimentos detectados pelo método de Mory (2000). Adaptado de Mory (2000).

componentes rotacionais (R_x,R_y e R_z) para em seguida, determinar as componentes T_x, T_y e T_z.

Dentre as soluções similares propostas para dispositivos portáteis destacam-se os trabalhos de Drab e Artner (2005), Rohs (2005) e Hannuksela *et al.* (2007). Os fundamentos de cada um dos dois métodos é descrito nas subseções a seguir e servem como base de comparação para o método proposto.

3.1.3 Drab e Artner (2005)

O trabalho proposto por Drab e Artner (2005) realiza a detecção de movimentos lineares (Δ_x,Δ_y) através da análise de deslocamentos por projeção (*projection shift analysis*). Este método consiste em converter a imagem bidimensional em dois vetores, as projeções em x e em y, dadas pela soma da intensidade de cada pixel na mesma coluna e na mesma linha, através de:

$$p_x(x) = \sum_{i=0}^{h-1} I(x,i) \quad e \quad p_y(y) = \sum_{j=0}^{w-1} I(j,y), \tag{3.4}$$

em que w e h são respectivamente o comprimento e a largura da imagem, i e j são variáveis usadas para cálculo da soma. O resultado deste procedimento é mostrado na Figura 3.12.

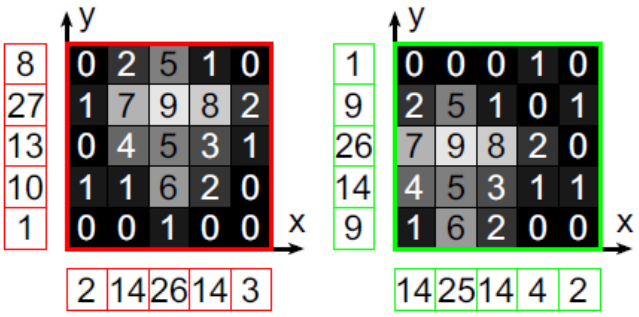


Figura 3.12: exemplo de projeção de imagens. Adaptado de Drab e Artner (2005).

Para realizar a detecção de movimento utilizam-se as projeções nas direções x e y em dois instantes de tempo consecutivos (t_1 e t_2).

O movimento realizado entre os dois instantes de tempo é estimado através da avaliação do deslocamento que gera maior correlação entre os dados de dois instantes

de tempo diferentes, através de:

$$SSD_x(\delta) = \sum_{x=0}^{w-1} p_x(x + \Delta_y, t_2) - p_x(x, t_1),$$
$$SSD_y(\delta) = \sum_{y=0}^{h-1} p_y(y + \Delta_y, t_2) - p_y(y+, t_1),$$

em que $p_x(x,t)$ e $p_y(y,t)$ são as projeções nas direções x e y no instante de tempo t e δ é o deslocamento avaliado. O valor de δ que gera menor valor é considerado como deslocamento realizado. Para fins de normalização o Soma dos Quadrados das Diferenças (SSD) é normalizado pelo número total de combinações, resultando em uma métrica de ajuste.

Um exemplo deste cálculo é mostrado na Figura 3.13, em que a correlação é calculada para deslocamentos iguais à -1, 0 e 1.

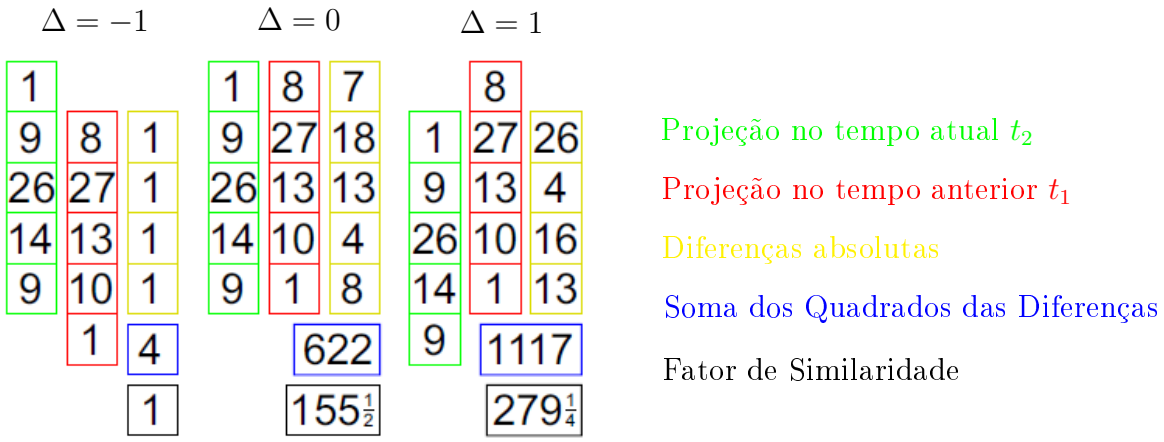


Figura 3.13: cálculo de correlação com três diferentes valores de deslocamento. Adaptado de Drab e Artner (2005).

Observando a Figura, percebe-se que o deslocamento igual à -1 possui a SSD e é considerado como deslocamento realizado entre os dois instantes de tempo.

Como este método realiza a correlação entre cada uma das possíveis correlações, o mesmo possui um esforço computacional reduzido e requer uma pouca quantidade de memória. Contudo, o método é impreciso para movimentos rápidos.

3.1.4 Rohs (2005)

Os métodos desenvolvidos por Rohs (2005) e Wang *et al.* (2006) são baseados em técnicas de casamento de blocos para detectar o movimento realizado por

dispositivos portáteis. O algoritmo detecta os movimentos lineares (Δ_x,Δ_y) e o movimento rotacional Δ_α (ROHS; ESSL, 2007), conforme mostrado na Figura 3.14.

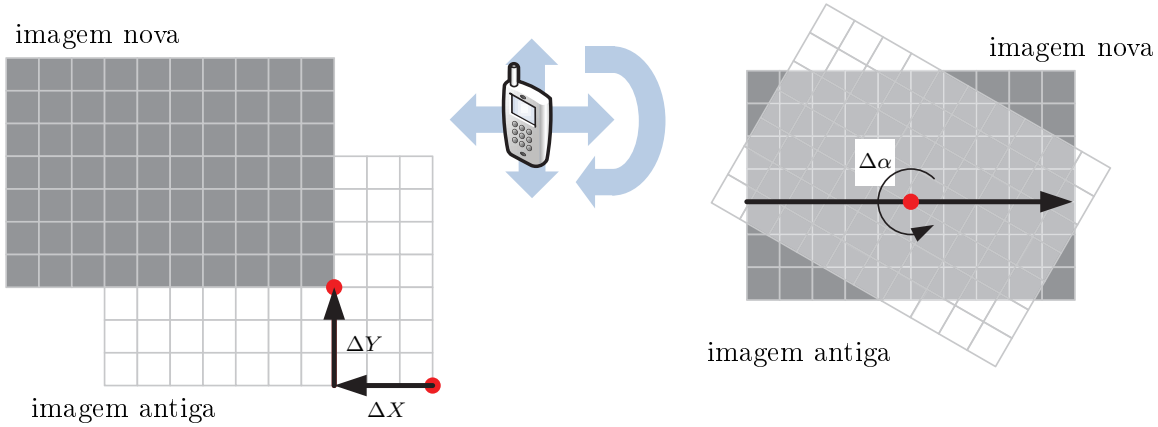


Figura 3.14: movimentos relativos detectados pelo método proposto por Rohs (2005). Adaptado de Rohs (2005).

Nesse algoritmo, a imagem é subdividida em blocos de tamanho $B \times B$ (por exemplo, 16x16). Para melhorar o esforço computacional, esse bloco é sub-amostrado à cada N pixels. O método utiliza a média dos tons de cinza do bloco sub-amostrado para avaliar a correlação deste com os blocos vizinhos no instante de tempo anterior.

O bloco vizinho que possuir a maior similaridade (menor diferença entre os tons de cinza) identifica o deslocamento realizado entre os *frames* adquiridos. São avaliados os vizinhos dentro de uma janela 7x7, ou seja, os possíveis valores para os deslocamentos nas linhas e colunas (Δ_x e Δ_y) variam de -3 a 3. É considerado como deslocamento realizado (o menor valor de distância) que encontra-se mais próximo do centro da janela de busca.

Para detectar os movimentos rotacionais, cada bloco no *frame* atual é rotacionado com um ângulo Δ_α, variando entre -24° e 24° em intervalos de 6°, o qual é comparado com *frame* anterior, posicionado na mesma posição (X,Y) no mesmo local. Por causa da sub-amostragem, este método é pouco sensível a pequenos deslocamentos da câmera, sendo capaz de reconhecer movimentos bruscos. Dependendo do aparelho utilizado, o algoritmo é capaz de processar entre 5 (ROHS, 2005) e 15 quadros por segundo (FPS) (ROHS; ESSL, 2007).

Na mesma direção, o método desenvolvido por Wang *et al.* (2006) é similar ao

método de Rohs (2005), sendo que o *frame* é subdividido em blocos 8x8 e cada bloco no quadro atual tem seus *pixels* comparados com os blocos adjacentes no *frame* anterior através de medidas de correlação: Erro Médio Quadrático (MSE), Soma das Diferenças Absolutas (SAD) ou Função de Correlação Cruzada (CCF), descritas na seção 2.2.2.

O bloco que possuir maior correlação identifica o deslocamento realizado. A diferença entre os métodos de Rohs (2005) e Wang *et al.* (2006) é que no primeiro, a comparação é feita *pixel* a *pixel* do *frame* sub-amostrado. Enquanto que, no segundo, a comparação é feita para todos os *pixels* de cada bloco. Segundo o autor, a taxa de processamento está em torno de 15 FPS em um Motorola v710.

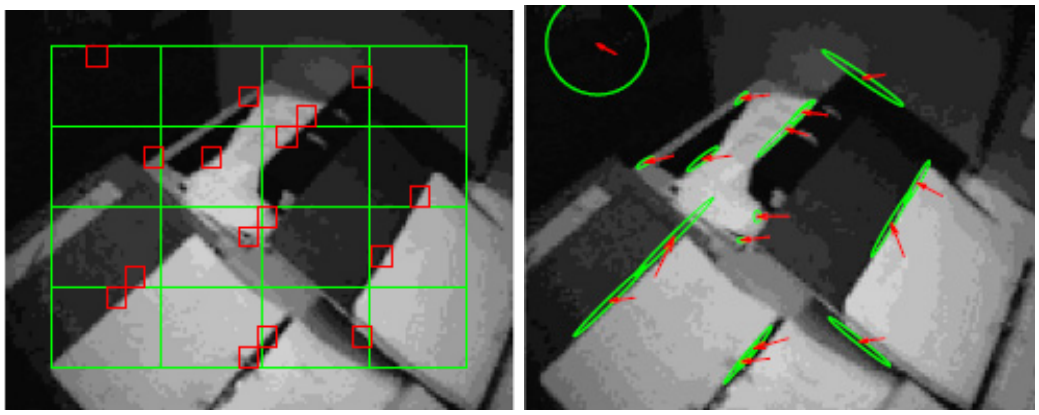
Os últimos métodos realizam a divisão em blocos para permitir a utilização em um dispositivo portátil. Esta divisão realiza a diminuição da resolução da imagem e faz com que estes métodos não sejam capazes de reconhecer movimentos suaves.

3.1.5 Hannuksela *et al.* (2005)

O trabalho de Hannuksela *et al.* (2011) propõe o uso de uma metodologia por duas formas de detecção de movimento: uma usando o movimento do dispositivo portátil e outra o rastreamento de objetos. Para ambos os casos, o sistema realiza a detecção de movimentos baseados na detecção do movimento das bordas e dos cantos localizados na primeira imagem adquirida.

Neste sistema, a imagem inicialmente é dividida em regiões não sobrepostas e tem os pontos de interesse localizados dentro de cada uma das janelas, conforme mostrado na Figura 3.15(a). Para cada um desses pontos, utiliza-se o método de casamento de blocos através da busca exaustiva (seção 2.2.2). Esta busca fornece uma forma para avaliar qual deslocamento e a incerteza da medida, os quais são mostrados respectivamente como uma seta vermelha e como um círculo verde na Figura 3.15(b) (HANNUKSELA *et al.*, 2011).

As informações extraídas são utilizadas por um modelo de similaridade que estima a translação e a rotação. Em seguida, esta informação é passada para uma rede de Markov que identifica o movimento realizado. O primeiro teste realizado pelo autor é feito com um Nokia 3650 que contem um processador de 104 MHz embarcado com o Symbian 6.1 OS, com uma taxa de processamento de 10 quadros por segundo.



(a) (b)

Figura 3.15: extração de pontos relevantes proposto por Hannuksela *et al.* (2011), a) pontos detectados e; b) estimação de movimento dos pontos relevantes. Adaptado de Hannuksela *et al.* (2007).

O sistema adquire quadros de tamanho 160 x 120, utilizando codificação de 12 bits por pixel, os quais são convertidos em imagens com 16 níveis de cinza. Esta imagem é utilizada como base para a extração de pontos relevantes. Uma região 25x25 em torno do ponto relevante é utilizada para realizar a busca exaustiva da sua nova localização.

3.2 Resumo

Nesta seção são resumidas as características das diversas técnicas utilizadas como sistema de interação para dispositivos portáteis. Para cada um destes, são apresentados na Tabela 3.1 os algoritmos utilizados a quantidade de graus de liberdade, tempo de processamento e dispositivo.

Convém ressaltar que nem todos os trabalhos descrevem todas estas características, representando informações não disponibilizadas na Tabela como o símbolo N/D.

Dentre os diversos trabalhos estudados neste capítulo, destacam-se os métodos de Drab e Artner (2005), Hannuksela *et al.* (2011) e Rohs e Essl (2007) pelo seu baixo custo computacional. Contudo, o primeiro método só é capaz de detectar movimentos translacionais enquanto que os outros dois métodos são capazes de detectar movimentos mais complexos como aproximação e distanciamento e rotação anti-horária e horária.

Nesta tese, é proposto um sistema de interação em tempo real com o dispositivo

Tabela 3.1: resumo dos métodos de interação desenvolvidos para dispositivos portáteis.

Autores	Método	Objeto de Referência	Graus de Liberdade	Dispositivo	Tempo de Processamento (ms)
Rohs e Zweifel (2005)	VisualCodes	Marcador	5	N/D	N/D
Gallo *et al.* (2008)	Projeção do Gradiente	Mão (Pele)	2	Nokia N95	5
Hansen (2005)	CAMShift	Face	4	N/D	N/D
Hildreth (2006)	Fluxo Óptico	Cantos	6	N/D	N/D
Drab e Artner (2005)	Projeção da Intensidade	Nenhum	2	N/D	N/D
Wang *et al.* (2006)	TinyMotion	Nenhum	2	Motorola v710	15
Rohs (2005)	*Sweep and Tilt*	Nenhum	3	Nokia N95 Nokia 6600	4 20
Hannuksela *et al.* (2007)	Casamento de Blocos	Cantos	2	Nokia 3650	N/D
Winkler *et al.* (2007)	ARToolkitPlus CAMShift TinyMotion	Marcador Face Nenhum	6 4 2	HP RW6828 PocketPC	30 27 5
Qin (2006)	Autolimiarização	Marcadores	2	Sony Ericson P910i	N/D

portátil através dos mesmos movimentos detectados pelos métodos de Hannuksela *et al.* (2011) e Rohs e Essl (2007). O sistema proposto é baseado em técnicas de detecção de movimentos por Fluxo Óptico de Horn e Schunck (1981), apesar de diversos autores reforçarem que o método de Fluxo Óptico (FO) possui um custo computacional elevado.

O método de Horn e Schunck (1981) estima o fluxo óptico através das diferenças espaço-temporais, que são dependem diretamente do contraste da imagem, por causa disto, é proposto um método de realce para aumentar o contraste de imagens em situações de baixa luminosidade, melhorando a estimação das componentes de velocidade nesta condição.

No próximo capítulo são descritas as etapas do sistema de testes e os detalhes dos métodos e das adaptações propostas.

Capítulo 4

Métodos Propostos

Este capítulo descreve o método para avaliação dos algoritmos estudados durante o desenvolvimento desta tese. Nesta tese, a otimização de um algoritmo clássico de estimação do fluxo óptico Fluxo Óptico (FO) proposto por Horn e Schunck (1981) está organizada em etapas mostradas na Figura 4.1.

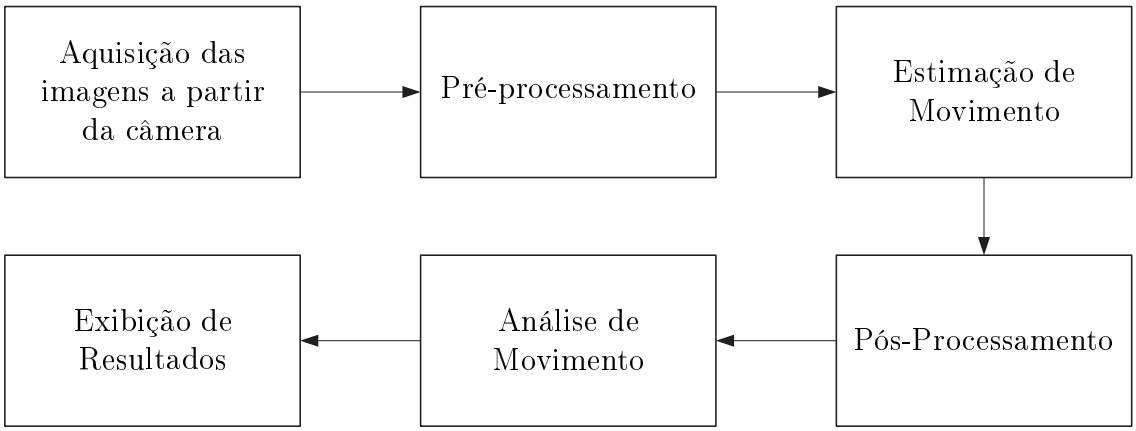


Figura 4.1: principais passos em processamento de imagem e vídeo.

O primeiro passo realiza a aquisição das imagens da câmera integrada ao dispositivo. As imagens adquiridas são convertidas para um formato RGB com seus três canais de cores separados, codificados com 16 bits por cor.

A segunda etapa é responsável por melhorar a qualidade ou ressaltar informações relevantes ao propósito desejado. Como o método de Horn e Schunck (1981) necessita da estimação de gradiente espacial e de gradiente temporal, nesta etapa são aplicados os filtros de Roberts para estimação dos gradientes espaciais (I_x e I_y) e, a partir

da diferença temporal, é aplicado um filtro de média para suavização da diferença espacial.

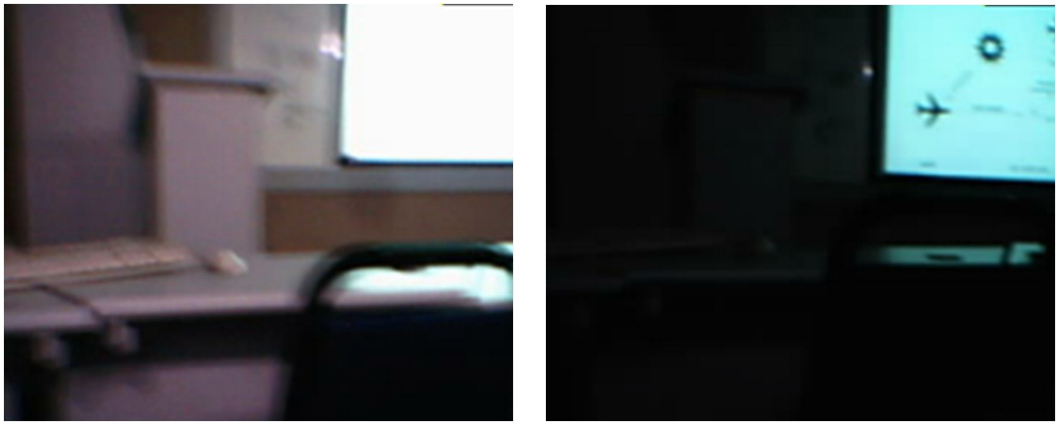
Como o método estudado é baseado em gradiente, o contraste das bordas é fundamental para a estimação de movimento. Assim, para situações de baixa luminosidade é proposto um realce de imagens baseado no ajuste de gama, descrito na seção 4.1. Na terceira etapa, os vetores de velocidade são estimados em cada um dos pixels da imagem através da otimização do método de Horn e Schunck (1981).

Na próxima etapa, os vetores de velocidades são analisados por dois diferentes métodos descritos respectivamente nas seções 4.2 e 4.3. Na seção 4.2 é descrita a modificação do algoritmo de FO para detecção de movimentos feitos pela câmera e a modificação do algoritmo para detectar movimentos de rotação e zoom é descrita na seção 4.3.

No último passo, há a exibição de resultados. Sobre o *frame* adquirido é desenhado um objeto que se move com o deslocamento realizado.

4.1 Ajuste de Contraste

Em ambientes escuros, até mesmo o olho humano sofre dificuldade de detectar objetos devido ao baixo contraste existente. Na Figura 4.2 é apresentado o mesmo ambiente com diferentes condições de luminosidade. No campo de visão da câmera existe um teclado, mouse, mesas, cadeiras e um quadro de projeção que são bastante nítidos quando as lâmpadas estão ligadas, conforme mostrado na Figura 4.2(a).



(a) (b)

Figura 4.2: ambiente de testes com as luzes a) acesas e b) apagadas.

Ao desligar as lâmpadas, os objetos presentes na cena são difíceis de serem

visualizados devido ao baixo contraste das bordas da imagem. Por outro lado, quando as luzes estão desligadas, é possível ver a informação projetada na tela, conforme mostrado na Figura 4.2(b).

Nos algoritmos de FOs baseados em gradiente, as bordas atuam como um elemento fundamental na detecção de movimentos. Uma solução muito utilizada é aplicar uma transformação para aumentar o contraste, realçando as bordas (STARCK *et al.*, 2003).

Uma forma de aumentar o contraste em imagens coloridas é converter a imagem para um espaço de cor em que a informação de brilho é separada da informação de cor, como por exemplo, o HSV e o YCbCr.

Visto que muitas câmeras utilizam o formato de cor YCbCr para adquirir as imagens, neste trabalho é aplicado um realce de contraste apenas no canal relacionado com o brilho de cada pixel (Canal Y). Caso a câmera não utilize este modelo, a imagem precisa ser convertida para o modelo YCbCr e ao final do realce é convertido para o modelo de cor utilizado pelo algoritmo de fluxo óptico.

Nesta tese é proposto um realce baseado em uma modificação do ajuste de gama (GONZALEZ; WOODS, 2010)

$$s = c \times (r + \epsilon)^{\gamma}\,, \tag{4.1}$$

em que r e s são os valores antes e depois do realce de imagem, c e γ são constantes positivas, e ϵ é uma pequena constante.

Para melhorar as características da imagem e possibilitar um melhor reconhecimento de movimento, nesta tese é proposta uma transformação seletiva dependendo da intensidade dos pixels de entrada r. Quando a intensidade é baixa (entre zero e cinquenta) é realizado um ajuste linear, transformando-os em uma intensidade entre zero e cento e cinquenta. A transformação é exponencial quando r é maior que cinquenta e para valores de entrada maiores que cinquenta a transformação tende à 255. Para obter esse objetivo, proposto o ajuste de gama modificado, dado por

$$s = c \times \left\{1 - \left[1 - (r + \epsilon)^{\Gamma}\right]\right\}, \tag{4.2}$$

em que todas as constantes são as mesmas da equação 4.1.

Uma comparação gráfica mostrando o relacionamento entre o ajuste de gama tradicional (com γ iguais a 1, 1/2 e 1/4) e o ajuste proposto é mostrada na Figura 4.3.

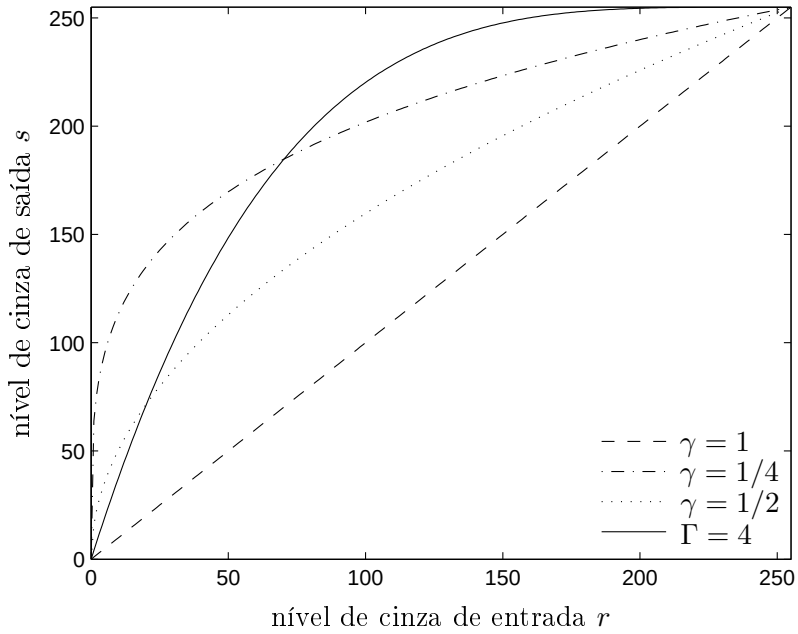


Figura 4.3: ajuste de gama tradicional com $\gamma = 1$, $1/2$ e $1/4$ e ajuste proposto com $\Gamma = 4$.

Através desta Figura, observa-se que para o ajuste de gama tradicional, no início da escala o ajuste tem crescimento exponencial. Dentre os diversos ajustes de gama, o método proposto possui as características desejadas.

4.2 Algoritmo de Fluxo Óptico Modificado

O algoritmo proposto é uma adaptação do Fluxo Óptico de Horn e Schunck (1981) que assume que a câmera se move suavemente, imageando uma cena sem nenhum movimento. De acordo com esta restrição, as componentes de velocidade na vizinhança de um *pixel*, no instante de tempo anterior, podem ser utilizadas como estimativa inicial da velocidade do *pixel* no instante de tempo atual (COSTA *et al.*, 2006), ou seja,

$$\begin{cases} \overline{v_x}\,(t-1) = F * v_x\,(t-1) \\ \overline{v_y}\,(t-1) = F * v_y\,(t-1) \end{cases} \tag{4.3}$$

em que $*$ representa a operação de convolução, F é uma máscara suavizadora, dada por, $\begin{bmatrix} 1/12 & 1/6 & 1/12 \\ 1/6 & 0 & 1/6 \\ 1/12 & 1/6 & 1/12 \end{bmatrix}$, $v_x(t-1)$ e $v_y(t-1)$ são as matrizes de velocidade calculada no instante de tempo $t-1$. Por fim, $\overline{v_x}(t-1)$ e $\overline{v_y}(t-1)$ são matrizes de velocidade média da vizinhança.

Levando em conta a consideração que as componentes de velocidade estimadas no instante de tempo anterior passam a ser ótimas estimativas da velocidade, as equações 2.10 e 2.11 podem ser reescritas como

$$v_x(t) = v_x(t-1) - I_x \cdot \frac{I_x \cdot v_x(t-1) + I_y \cdot v_y(t-1) + I_t}{\lambda^2 + (I_x^2 + I_y^2)}, \tag{4.4}$$

$$v_y(t) = v_y(t-1) - I_y \cdot \frac{I_x \cdot v_x(t-1) + I_y \cdot v_y(t-1) + I_t}{\lambda^2 + (I_x^2 + I_y^2)}, \tag{4.5}$$

em que $v_x(t)$ e $v_y(t)$ são as matrizes de velocidade. Desta forma, o método deixa de ser interativo, sendo capaz de detectar o movimento realizado através do cálculo das equações 2.10 e 2.11, conforme mostrado no Algoritmo 4.1.

Algoritmo 4.1 algoritmo de fluxo óptico proposto neste trabalho.

Entrada: v_x e v_y - as componentes de velocidades de um pixel (x, y) em um dado instante de tempo t

Entrada: I_T, I_x e I_y - os gradientes temporais e espaciais calculados entre os instantes de tempo t e $t-1$

$\overline{v_x} \leftarrow media(v_x(t))$ {devido à restrição adicional do movimento uniforme da cena}

$\overline{v_y} \leftarrow media(v_y(t))$

$v_x(t) \leftarrow v_x(t-1) - I_x \cdot [I_x \cdot v_x(t-1) + I_y \cdot v_y(t-1) + I_t] \div \left[\lambda^2 + (I_x^2 + I_y^2)\right]$

$v_y(t) \leftarrow \overline{v_y}^k - I_y \cdot [I_x \cdot v_x(t-1) + I_y \cdot v_y(t-1) + I_t] \div \left[\lambda^2 + (I_x^2 + I_y^2)\right]$

Para melhorar a detecção, um fator α é utilizado como ponderação da velocidade estimada, o que resulta em

$$v_x(t) = \alpha \cdot v_x(t-1) - I_x \cdot \frac{I_x \cdot v_x(t-1) + I_y \cdot v_y(t-1) + I_t}{\lambda^2 + (I_x^2 + I_y^2)}, \tag{4.6}$$

$$v_y(t) = \alpha \cdot v_y(t-1) - I_y \cdot \frac{I_x \cdot v_x(t-1) + I_y \cdot v_y(t-1) + I_t}{\lambda^2 + (I_x^2 + I_y^2)}. \tag{4.7}$$

Este fator (α) é que permite diminuir a histerese gerada pelo algoritmo original principalmente, quando funciona a baixas frequências de atualização dos *frames* de vídeo. Além disto, durante um movimento contínuo, o fator α limita o crescimento da velocidade, pois utiliza uma parte da velocidade anterior e possibilita uma melhor adaptação aos movimentos que se modificam com o tempo.

As matrizes de velocidade são utilizadas para estimar o deslocamento da câmera

$(\Delta x,\Delta y)$, calculando a média dos vetores de velocidade não nulos, dados por

$$\begin{cases} \Delta x = \sum_{x=0}^{M} \sum_{y=0}^{N} v_x(x,y,t) & | \quad v_x(x,y,t) \geq v_{\min}, \\ \Delta y = \sum_{x=0}^{M} \sum_{y=0}^{N} v_y(x,y,t) & | \quad v_y(x,y,t) \geq v_{\min}, \end{cases} \tag{4.8}$$

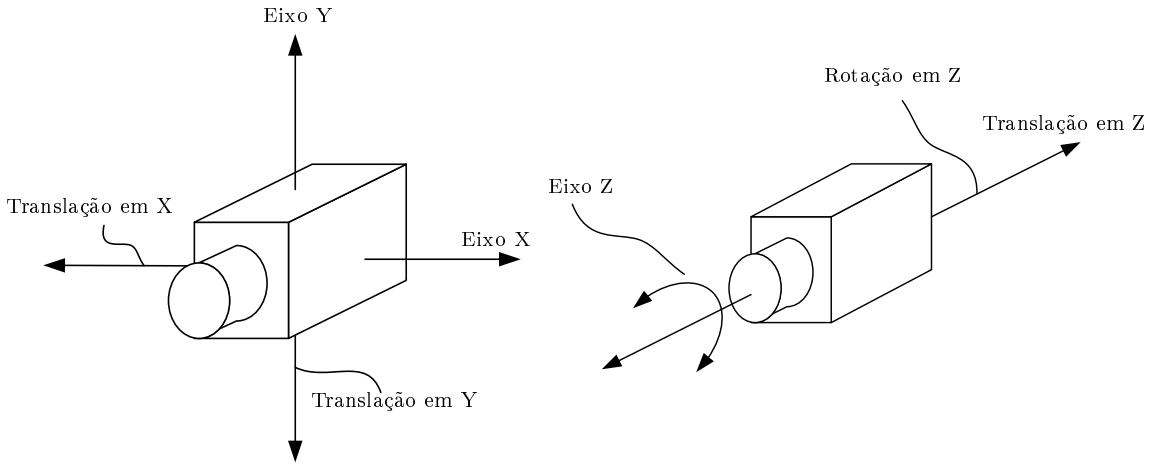
em que M e N são as dimensões da imagem e v_{min} é o limiar de velocidade. Velocidades menores que este limiar são consideradas ruído e são desprezadas. O sinal $-$ foi usado devido ao movimento da câmera ter direção oposta ao movimento óptico. Por exemplo, se a cena está se movendo para esquerda, a câmera está se movendo para direita.

4.3 Classificação dos vetores de velocidade

O método descrito na seção 4.2 realiza um estudo simplificado dos vetores de velocidade, contemplando a estimação de movimento apenas nos eixos X e Y. Um estudo mais profundo da distribuição dos vetores de velocidade é capaz de identificar movimentos simples e complexos conforme mostrado na Figura 4.4. Exemplos de movimentos simples são os movimentos de translação na direção X e Y e, bem como exemplos de movimentos complexos são o movimento de distanciamento e de aproximação (*zoom in/out* ou translação no eixo Z) e rotação em torno do eixo Z (no sentido horário e anti-horário).

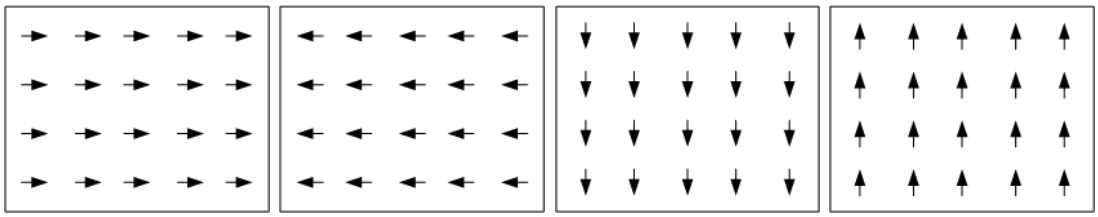
A partir do conhecimento teórico do FO, para cada movimento, pode-se determinar a distribuição dos vetores de velocidade (COSTA *et al.*, 2008). Por exemplo, durante o movimento de translação nos eixos X e Y, a maior parte dos vetores de velocidade estão alinhados no sentido contrário ao movimento, conforme mostrado nas Figuras 4.5(a) à 4.5(d) (PARAGIOS *et al.*, 2005).

Para esse tipo de movimento, a direção do movimento pode ser calculada pela média dos vetores de velocidade. Contudo, para movimentos de rotação e translação no eixo Z (aproximação/distanciamento), o vetor de velocidade é aproximadamente nulo. Durante um movimento de aproximação (*zoom in*), os vetores de velocidade apontam do centro para fora, enquanto que no de distanciamento (*zoom out*) apontam de fora para o centro. Durante movimentos de rotação, eles se distribuem de maneira radial, fazendo um círculo fechado, conforme mostrado nas Figuras 4.6(a) à 4.6(d).



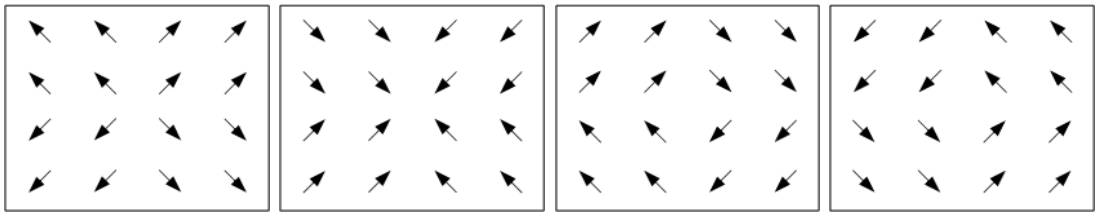
(a) (b)

Figura 4.4: movimentos detectáveis pelo a) fluxo óptico modificado, e b) por classificação de velocidades.



(a) (b) (c) (d)

Figura 4.5: ilustração da distribuição dos vetores de velocidades para movimentos de translação para, a) esquerda, b) direita c) cima, e d) baixo.



(a) (b) (c) (d)

Figura 4.6: ilustração da distribuição dos vetores de velocidades para movimentos de a) aproximação, b) distanciamento, c) rotação anti-horária e d) rotação horária.

Com base nestas informações, o algoritmo proposto realiza a comparação de cada um dos vetores de velocidade estimados pelo FO Modificado com um conjunto formado por modelos de distribuição de vetores de velocidades de cada um dos movimentos ilustrados nas Figuras 4.5 e 4.6 dados por (COSTA *et al.*, 2008)

$$\left(\theta^{referncia_i} - \theta^{estimado}\right) \leq \theta^{tolerncia}, \tag{4.9}$$

em que $\theta^{referncia_i}$ representa o ângulo da velocidade de referência no modelo i, $\theta^{estimado}$ é o ângulo do vetor de velocidade estimado e $\theta^{tolerncia}$ é um limiar de tolerância.

A quantidade de vetores de velocidades gerados pelo algoritmo de FO é, em geral, elevada. Caso todos fossem utilizados, seria necessário um grande esforço computacional para determinar o movimento. Isto inviabilizaria a implementação em dispositivos portáteis.

Para isto, as matrizes de velocidade são dividas em regiões e tem seu vetor médio calculado. Cada um destes vetores médio de regiões é comparado com vetores de referência, tornando viável a aplicação desta medida.

No próximo capítulo são descritos os equipamentos e as formas de avaliação dos algoritmos estudados que possibilitam uma análise comparativa dos algoritmos presentes no estado da arte e dos propostos nesta tese.

Capítulo 5

Materiais e Métodos

Neste capítulo são apresentados a metodologia utilizada para o desenvolvimento e os testes dos algoritmos estudados, conforme apresentado na Figura 5.1.

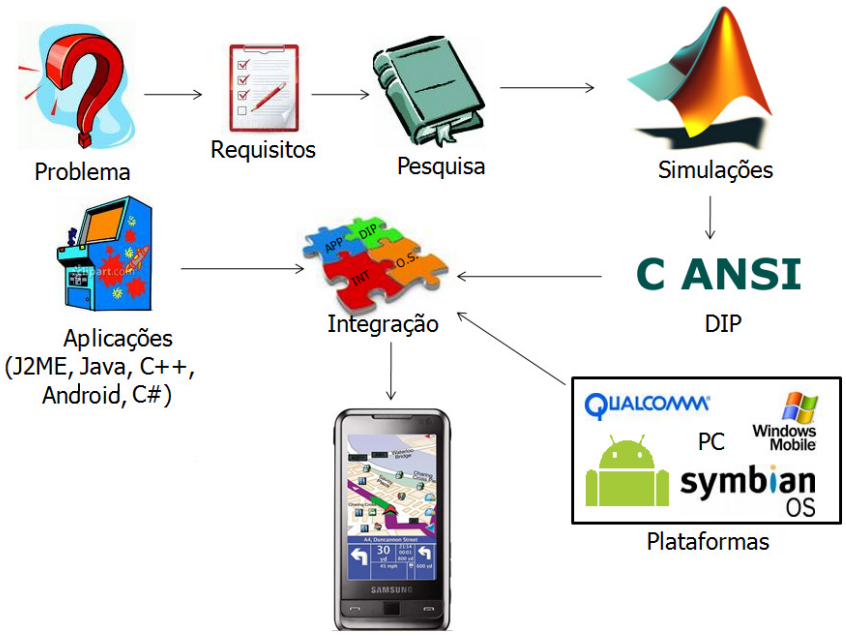


Figura 5.1: metodologia utilizada no desenvolvimento desta tese.

Como o interesse desta tese é desenvolver um método de detecção de movimentos com o requisito de possuir uma detecção eficaz associado a um baixo custo computacional. Com base nestes requisitos, foi feita uma pesquisa que culminou na proposta de um novo algoritmo de fluxo óptico, descrito no capítulo anterior.

Os métodos são então simulados no Simulink e então os algoritmos são implementados como módulos de um algoritmo em C ANSI que posteriormente são

integrados à uma aplicação desenvolvida no sistema portátil para enfim, poderem ser avaliados seu custo computacional e sua eficiência de detecção de movimento.

Neste capítulo, inicialmente, os equipamentos, os softwares de simulação e ferramentas de desenvolvimento utilizados nesta tese. Em seguida, são apresentadas as metodologias para avaliação de resultados.

5.1 Equipamentos Utilizados

Os algoritmos foram inicialmente desenvolvidos em um Computador Pessoal (PC) e, posteriormente, foram adaptados e embarcados nos dispositivos portáteis. As características como, Sistema Operacional (SO), resolução de câmera, tamanho de memória RAM, dos equipamentos utilizados no desenvolvimento desta tese são resumidamente descritos na Tabela 5.1.

Tabela 5.1: equipamentos utilizados no desenvolvimento desta tese.

	Notebook	Telefones Celulares		
Fabricante	Dell	Samsung	Nokia	Samsung
Modelo	Precision M65	SGH-A706	N95	Galaxy 5
Processador	Intel Core2Duo T7200	ARM9 à 143 MHz	Arm 11 à 330MHz	Arm 11 à 600 MHz
RAM	3GB DDR2 667MHz	64 MB	64 MB	128 MB
SO	Windows XP com SP2	Qualcomm REX	Symbian S60	Android 2.2.2
Câmera	1.3 MPix Microsoft NX-3000	2 MPix	5 MPix Carl Zeiss	2 MPix

5.2 Ambientes de Desenvolvimento

Nesta seção são descritos, em linhas gerais, os ambientes utilizados para o desenvolvimento desta tese. Inicialmente, o ambiente para desenvolvimento das simulações (Matlab). Em seguida, os programas necessários para o desenvolvimento dos algoritmos de Visão Computacional (VC) em C ANSI são descritos e por fim, os sistemas operacionais dos dispositivos portáteis são descritos.

5.2.1 Matlab

O Matlab e seu pacote de simulação dinâmica (Simulink) (MATHWORKS, 2009) são os ambientes de programação e de simulação mais utilizados no ramo de controle automático em geral e de processos específicos. Várias ferramentas dentro da área de controle têm sido desenvolvidas para esta plataforma como, por exemplo, o controle preditivo, o controle robusto, a identificação de sistemas, as redes neurais, a lógica fuzzy, etc. Além disto, este programa pode ser útil para os engenheiros simularem a dinâmica de vários processos (WARD, 2007).

A estrutura modular do Simulink permite o agrupamento de modelos dentro de hierarquias que provê uma visão geral do sistema e uma fácil manutenção de componentes e de sistemas complexos. O Simulink utiliza um ambiente gráfico baseado em diagramas de blocos que suportam diferentes operações, como por exemplo, funções aritméticas, entrada e saída de dados, funções de transferências, modelos de estado de espaços, dentre outras (KALAGASIDIS *et al.*, 2007).

A partir de pequenos blocos, disponíveis nesse ambiente, um conjunto de bibliotecas de várias áreas pode ser desenvolvido. Cada biblioteca é formada por vários sub-blocos pequenos ou submodelos que representam uma função específica de uma área específica. Nesse ambiente, partir do arranjo de vários destes blocos na tela e em seguida, através da conexão de blocos com algumas variáveis e constantes, é possível simular vários sistemas de equação (TAYLOR *et al.*, 2004).

Dois pacotes do Matlab são muito importantes para possibilitar a simulação de algoritmos de VC: aquisição de imagens (MATHWORKS, 2006a) e Vídeo e Imagem (MATHWORKS, 2006b).

O pacote de aquisição de imagens fornece uma interface com vários tipos de câmeras digitais ou placas de aquisição de dados, realizando a configuração, o controle de interrupção de forma a possibilitar a aquisição de *frames* em formato Formato de Cor *Red Green Blue* (RGB). Então é possível adquirir os vídeos com tamanhos variados, dependendo da capacidade de aquisição da câmera.

O pacote de vídeo e de imagens possui um conjunto de funções úteis para a comunidade de processamento de imagens compostas por funções de aquisição, filtragem, análise e extração de características, bem como a exibição de resultados (MATHWORKS, 2006b). Este pacote é capaz de adquirir vídeos compatíveis (AVI e WMV) e imagens (BMP, PNG, TIF, JPEG), que serão processados por um

conjunto de blocos. Dentro das funções de processamento de imagens deste destacam-se funções de conversão de formatos de imagens, limiarização automática pelo método de (OTSU, 1979), correção, convolução 2D, filtros FIR e da mediana, operações morfológicas como erosão e dilatação, transformadas matemáticas como a Transformada Rápida de Fourier (FFT) e a Transformada Discreta do Cosseno (DCT), dentre outras (MATHWORKS, 2006b). Por fim, através de funções de exibição, os vídeos processados são exibidos ou salvos em arquivo.

5.2.2 Modelagem de processamento de imagem em Simulink

Ao desenvolver uma aplicação em processamento de imagens, a estrutura ilustrada na Figura 5.2 sempre deve ser seguida. No primeiro passo, a aquisição pode ser realizada pelos pacotes de processamento vídeo e imagens, aquisição de imagens, ou uma outra função desenvolvida do Matlab (**mfunction**), ou funções específicas para o Simulink (**sfunction**).

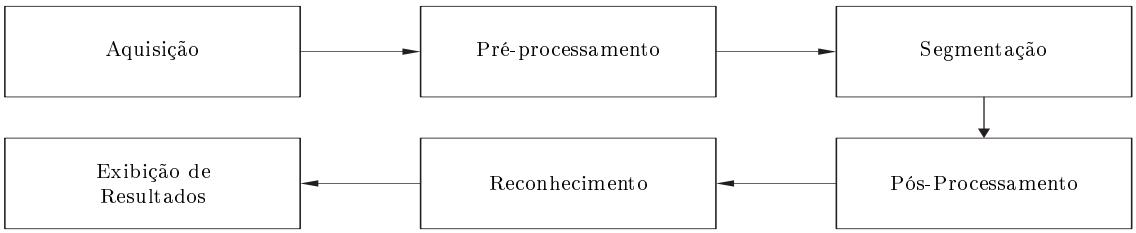


Figura 5.2: principais passos em processamento de imagem e vídeo.

No último passo, a exibição de resultados é realizada pelo pacote de processamento vídeo e imagens, exibindo no monitor, salvando no arquivo, ou exportando para a linha de comando (*Shell*) do Matlab.

Os processamentos intermediários de pré e pós-processamento, segmentação, extração de características ou análise são intimamente ligados à aplicação. Ao desenvolver a mesma, existem três fases típicas.

Na primeira fase, estimam-se quais etapas são necessárias e uma lista de métodos potencialmente indicados para resolver o problema. Na segunda fase, são implementados algoritmos para cada uma das etapas presentes na aplicação. Na última etapa, é desenvolvida a aplicação, em que os algoritmos que possuírem melhor custo benefício (performance x custo computacional) são implementados, utilizando uma linguagem de programação adequada.

Devido o fato de o Simulink operar via diagrama de blocos, através de sua

utilização, pode-se modelar e caracterizar com bastante precisão as várias etapas de processamento, sendo possível gerar modelos complexos a partir da especialização de modelos simples.

Após o modelo caracterizado, o Simulink oferece a possibilidade de testar várias combinações de métodos em várias etapas de processamento, sendo necessário apenas trocar o bloco de um algoritmo de uma etapa específica. Isto possibilita, não só uma avaliação do desempenho do algoritmo, mas também uma estimativa do esforço computacional através de um bloco de estimação de taxa de *frames* (fps). Neste bloco é possível ainda, investigar a influência do tamanho da imagem para o processamento, permitindo a escolha do tamanho de imagem e processamentos viáveis para serem utilizados em aplicações em tempo real.

5.2.3 Code::Blocks

Este Ambiente de Desenvolvimento Integrado (IDE) foi concebido para trabalhar com códigos C/C++. Este programa possui o diferencial de ser código aberto e multiplataforma, possuindo implementações para Windows e Linux. Ele suporta diversos tipos de compiladores, mas uma boa alternativa é utilizar o compilador GCC e G++ do MinGW, pois ele já possui uma versão de instalador que possui esse compilador integrado.

5.2.4 *Open Source Computer Vision Library* (OpenCV)

O OpenCV é uma biblioteca concebida, especialmente para processamento e análise de imagem em tempo real, implementa os principais algoritmos de Processamento Digital de Imagens (PDI), VC e Inteligência Artificial (IA).

Devido seu vasto conjunto de funções em C/C++, esta biblioteca pode ser utilizada como base de diversas aplicações como interface homem-máquina, identificação de objetos, segmentação e reconhecimento, reconhecimento de faces, detecção de movimento, rastreamento do movimento (REIS; TAVARES, 2007).

Durante o desenvolvimento dos algoritmos, o OpenCV foi utilizado apenas como meio de acesso à câmera. Todas as etapas de processamento foram desenvolvidas em C ANSI para torná-las independente dos dispositivos e das plataformas.

Após o desenvolvimento e o teste da implementação feita utilizando o OpenCV, o código é integrado ao sistema de desenvolvimento, o qual é utilizado nos dispositivos móveis para que os testes de desempenho possam ser efetuados.

5.3 Sistemas Operacionais para dispositivos móveis

Nesta seção são descritos os detalhes dos Sistemas Operacionais (SOs) presentes nos dispositivos móveis utilizados nesta tese. Inicialmente descreve-se o sistema operacional do dispositivo com menor performance computacional, passando para o Symbian e depois finalizando a seção através da descrição do mais novo sistema operacional usado em dispositivos móveis, o Android.

5.3.1 Qualcomm REX

O *Real Time Executive* (REX) é um SO multitarefa preemptivo produzido pela Qualcomm, sendo fornecido às empresas fabricantes de dispositivos que utilizam os processadores da família *Mobile Station Modem* (MSM) produzidos pela Qualcomm. O diagrama de blocos deste SO é mostrado na Figura 5.3 (CHO; JEON, 2007).

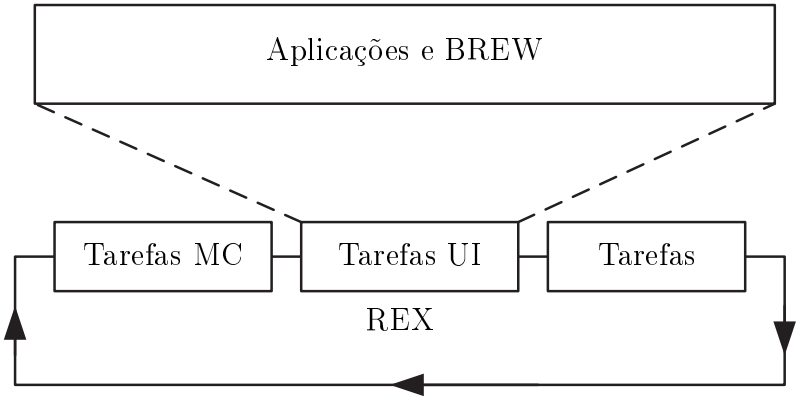


Figura 5.3: diagrama de bloco de aplicativos baseados no REX.

O REX consiste em um conjunto de tarefas e cada uma destas realizam um laço infinito. O sistema seleciona as tarefas de acordo com uma prioridade baseada em agendamento. Isto é, o REX controla as tarefas através de operações de inicialização e finalização da execução de cada uma das tarefas (CHO; JEON, 2007). O REX possui tarefas especializadas para Interfaces de Usuário (UI) e de Controle Mestre (MC). As tarefas de UI realizam a interação com o usuário, enquanto que tarefas MC controlam outras tarefas.

Uma característica importante do REX é a sua baixa ocupação de memória, o sistema operacional consome apenas 5KB de ROM e ocupa na memória poucos KB para realizar suas tarefas como sincronização de tarefas, atendimento de chamadas de procedimentos e gerenciamento de memória dinâmica.

O ambiente de desenvolvimento de software é fechado, isto é, o desenvolvimento de aplicações é feito pelos fabricantes de dispositivos portáteis através de seus colaboradores. É possível desenvolver aplicações do contexto de *internet* através do *Binary Runtime Environment for Wireless* (BREW). Este sistema foi projetado para permitir que o desenvolvedor escolha a linguagem para a codificação dos programas. Oferece suporte nativo à linguagem C/C++, mas aceita a integração de *browsers* e as aplicações em outras linguagens, inclusive Java e XML.

O BREW inclui plataforma de aplicativos e ferramentas para embarcar as aplicações pelos fabricantes de dispositivos, um kit de desenvolvimento de software (BREW SDK) e um sistema de distribuição (BDS). O último é controlado e gerenciado pelas operadoras, o que permite obter aplicativos dos desenvolvedores, comercializá-los e coordenar os processos de faturamento e pagamento (JESUS; NASCIMENTO, 2006).

O BREW não fornece proteção de memória para cada processo em execução. Isto possibilita o acesso à qualquer área de memória do dispositivo e compartilhar dados com outros aplicativos, contudo pode ocasionar travamentos no sistema (BREW, 2009; JESUS; NASCIMENTO, 2006). Por este motivo, o custo de desenvolvimento no BREW é mais alto, pois toda aplicação tem que ser testada e certificada pela Qualcomm para garantir que não se introduza na rede aplicações mal intencionadas ou simplesmente mal feitas e, com isso, ponha-se em risco toda a credibilidade da plataforma BREW (JESUS; NASCIMENTO, 2006; BREW, 2009).

5.3.2 Symbian

O Symbian é um sistema operacional desenvolvido a partir do sistema operacional EPOC para dispositivos portáteis, produzido pela companhia PSION. O Symbian foi projetado para considerar as características de dispositivos portáteis, ou seja, é otimizado para a limitação de recursos, sendo um sistema econômico no consumo de recursos, em especial, a memória e a bateria (CHO; JEON, 2007; IIDA, 2006).

Este sistema possui plataforma de desenvolvimento aberta e suas aplicações podem ser desenvolvidas em diferentes linguagem de programação, como Symbian C++, Perl, Java e Flash (CRUZ, 2008).

Na plataforma nativa do Symbian OS utilizada no desenvolvimento das aplicações é baseada no C++. O Symbian C++ é uma linguagem complexa e permite ao programador ter um grande controle sobre o código de sua aplicação,

facilitando a criação de otimizações mais refinadas (CRUZ, 2008).

Ao realizar o desenvolvimento na plataforma nativa do Symbian OS, implica na possibilidade do uso direto de API's nativas do sistema, que são bastante otimizadas. O desenvolvimento nesta plataforma também oferece o acesso completo aos recursos de *hardware* ao programador (CRUZ, 2008).

5.3.3 Android

O Android é uma plataforma com código aberto para dispositivos portáteis com recursos computacionais limitados. A plataforma possui sua arquitetura baseada em máquinas virtuais. Cada aplicação é executada dentro de uma instância exclusiva da máquina virtual chamada Dalvik, projetada para ser portátil entre os vários processadores ARM (EHRINGER, 2010).

O Dalvik consiste em uma máquina virtual interpretada capaz de executar um conjunto próprio de *bytecodes*, compilados a partir de um código Java. Esta máquina foi escrita em sua totalidade em C e possui bibliotecas C e C++ e é executada sobre o núcleo Linux, responsável por gerenciar a memória de baixo nível, os processos, as *threads* e a segurança dos arquivos e pastas, além do acesso à rede, *drivers* e bibliotecas de sistema (LECHETA, 2010; EHRINGER, 2010; KHAN *et al.*, 2009).

Um grande diferencial deste sistema para os demais é a sua plataforma de desenvolvimento aberta que permite o desenvolvimento de aplicativos com uma grande facilidade. A Google oferece o Android *Software Development Kit* (SDK) gratuitamente, de maneira a permitir que os usuários de dispositivos produzam aplicações, as quais são capazes de incorporar qualquer um dos recursos disponíveis ou serviços disponíveis no sistema (KHAN *et al.*, 2009).

O SDK é excelente para desenvolver aplicações. Contudo, em alguns momentos é necessário desenvolver códigos com desempenho computacional elevado ou acesso direto ao *hardware*, como por exemplo, a placa aceleradora gráfica utilizando a biblioteca OpenGL. Para isto, a Google fornece um segundo *kit* de desenvolvimento nativo (NDK), que permite o desenvolvimento de aplicações C e C++, facilmente integráveis com os códigos Java.

Nesta tese, os códigos desenvolvidos são portáveis entre os diferentes SOs. Assim, os códigos desenvolvidos em C ANSI são compilados utilizando a Android NDK, e integrados à interfaces gráficas desenvolvidas através do SDK.

5.4 Implementação

Os métodos desenvolvidos neste trabalho foram testados em dispositivos com diferentes poderes computacionais, memória e sistemas operacionais. Apesar das diferenças, os Sistemas Operacionais dos dispositivos utilizados possuem uma base em comum que é a linguagem C e C++. Essa linguagem é utilizada para desenvolvimento de muitos SOs, pois, combina várias características importantes como portabilidade de código, eficiência computacional, além de possuir a habilidade de acessar endereços específicos do hardware. O desenvolvimento de aplicações nestes SOs possuem linguagens específicas, mas em todos é possível integrar códigos C ANSI. Por causa disto, os métodos de VC nesta tese foram desenvolvidos em C ANSI, sem a utilização de bibliotecas ou recursos e funcionalidades do SO.

O diagrama da implementação em C ANSI dos métodos de interação baseado em VC são mostrados na Figura 5.4.

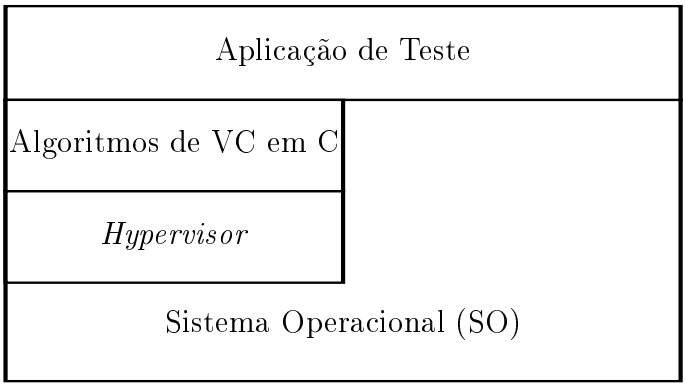


Figura 5.4: diagrama da implementação em C dos algoritmos.

Para utilização do método proposto em várias plataformas (Linux, Windows, Windows Mobile, REX e Symbian, Andorid) foi desenvolvido um *hypervisor*[1] (conhecido como, camada de virtualização). Esta camada de abstração tem o objetivo de facilitar o acesso aos dispositivos de hardware necessários aos algoritmos de Visão Computacional (VC), como entradas de teclado, câmera, sistemas de arquivo, dentre outros. Assim, ocorre uma separação entre o algoritmo de VC e

[1] O *hypervisor*, ou Monitor de Máquina Virtual (VMM), é uma camada de software entre o hardware e o sistema operacional. O VMM é responsável por fornecer ao sistema operacional visitante a abstração da máquina virtual. É o hypervisor que controla o acesso dos sistemas operacionais visitantes aos dispositivos de hardware. É interessante ressaltar que o VMM não executa em modo usuário, pois é ele que deve executar, ou simular a execução, das instruções privilegiadas requisitadas pelo sistema operacional visitante.

o *hardware*, podendo ser portado em diferentes plataformas através da construção de um *hypervisor* compatível. Isto traduz o acesso ao *hardware* em uma interface conveniente aos algoritmos desenvolvidos, conforme mostrado na Figura 5.5.

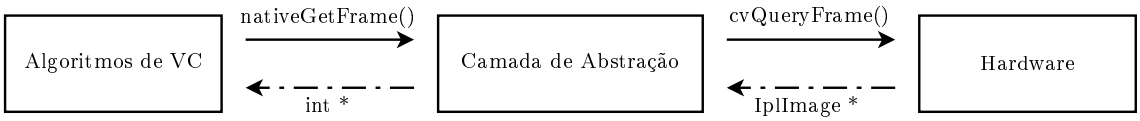


Figura 5.5: exemplo de interface de acesso à câmera no PC.

Por exemplo, para os testes serem realizados no PC, a interface de acesso ao *hardware* da câmera foi totalmente desenvolvida utilizando comandos existentes na Interface de Programação de Aplicativos (API) do OpenCV, os quais, após alguns processamentos necessários, convertem os *frames* adquiridos em um vetor ponto fixo (inteiro), o qual é utilizado como base para o processamento em C ANSI.

Ao final do processamento, algoritmos de VC enviam para a camada de virtualização, os quadros processados que são convertidos em formato adequado ao hardware. Assim, ocorre um isolamento de hardware na camada de abstração de hardware, o que proporciona a utilização em diferentes sistemas operacionais a partir do desenvolvimento de um conjunto de comandos de acesso ao meio físico e que dependem de cada plataforma.

Para testar os algoritmos, aplicações foram desenvolvidas, utilizando todas as funcionalidades disponíveis no SOs e realizam a execução dos algoritmos de VC. No ambiente PC foi desenvolvida uma aplicação em linha de comando (*Console*) que inicia os algoritmos de detecção de movimentos e entra em um laço que adquire um *frame* da câmera, executa o algoritmo proposto e exibe os resultados até que uma tecla seja pressionada. Após sair do laço, a aplicação fecha os algoritmos de VC e termina a aplicação.

No ambiente Symbian ou Android, foi desenvolvida uma aplicação utilizando Interface Gráfica do Usuário (GUI) que realiza um procedimento similar ao ambiente PC, sendo instalados no dispositivo portátil através de uma conexão Bluetooth ou via cabo *Universal Serial Bus* (USB). Já no ambiente REX, os algoritmos foram compilados em C ANSI e são enviados para o grupo de desenvolvedores do SIDI. Os desenvolvedores implementaram a interface de acesso ao *hardware* e integraram o algoritmo em uma aplicação nativa do SO, deixando disponível o conjunto de

algoritmos para serem testados e avaliados.

5.5 Métodos avaliados

Nesta seção é descrito a metodologia utilizada apara realizar uma análise comparativa entre os diversos algoritmos estudados. Para realizar uma análise comparativa do método de pré-processamento proposto na seção 4.1 com métodos tradicionais de realce de imagens, foram implementados os algoritmos de equalização de histograma e ajuste de gama.

Além desta, neste trabalho é feita uma análise comparativa do desempenho dos métodos propostos com algoritmos de estimação de movimento para dispositivos móveis: Drab e Artner (2005), Hannuksela *et al.* (2011) e Rohs e Essl (2007). A metodologia utilizada para avaliação destes algoritmos é apresentada na seção a seguir.

5.6 Formas de Avaliação

As formas de avaliação de algoritmos em geral, dependem de suas aplicações. Neste caso, inicialmente, a qualidade de estimação de movimento dos métodos estudados é avaliada a partir de um vídeo sintético. O movimento realizado no vídeo sintético é previamente conhecido e, então, utilizado como referência para uma análise comparativa do movimento que gerou o vídeo sintético com o movimento estimado. Em seguida, a metodologia para avaliar os métodos computacionalmente é apresentada para avaliar sua possível utilização como interface de interação.

Por fim, a última seção deste Capítulo descreve a metodologia utilizada para análise de usabilidade dos métodos de estimação de movimento.

5.6.1 Análise com Vídeos Sintéticos

Para avaliação inicial dos algoritmos simulados, um vídeo sintético é gerado utilizando uma janela deslizante sobre uma imagem estática de alta resolução, mostradas na Figura 5.6. Cada *frame* (176 x 144) ou (320 x 240) é obtido através de operações de interpolação, sub-amostragem e adição de ruído gaussiano.

A imagem de resolução (800x600) consiste em um ambiente de escritório, envolvendo objetos com uma pequena distância em relação à câmera.

Figura 5.6: imagem estática utilizada neste trabalho.

O processo de geração de vídeo sintético é mostrado na Figura 5.7. Sobreposta a imagem, a região desenhada em linha preta contínua é a região em que são adquiridos o primeiro e último quadro. A seta preta indica o sentido do movimento, a linha branca pontilhada indica a trajetória e os quadrantes desenhados de linha preta limitam quatro quadros do vídeo sintético gerado.

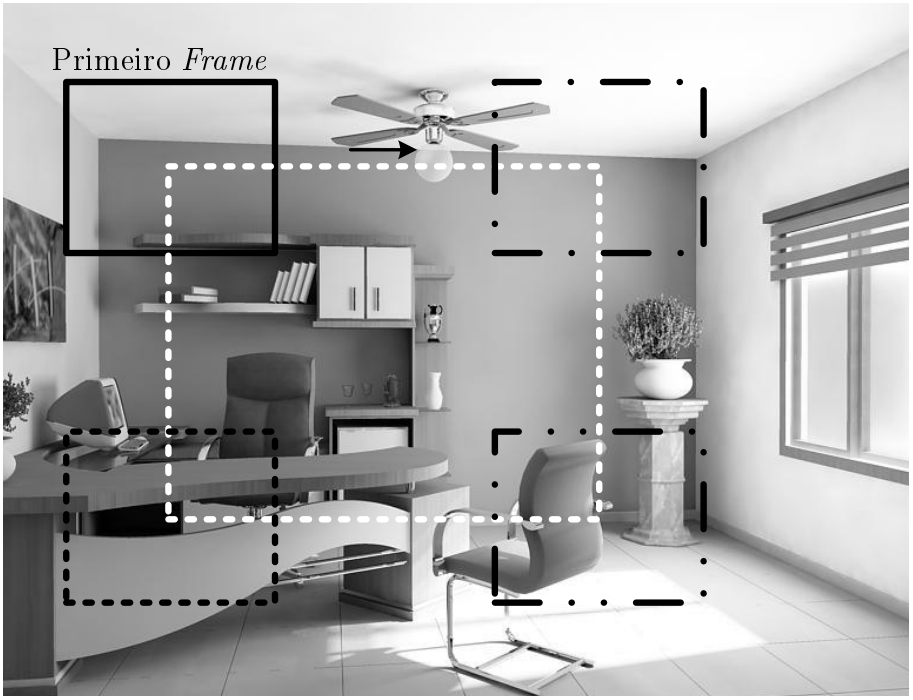


Figura 5.7: geração de um vídeo controlado utilizando uma imagem estática.

São adquiridos vídeos utilizando dois movimentos básicos (retangular e circular), adotando-se os dois sentidos horários e anti-horário, totalizando quatro movimentos, mostrados na Figura 5.8.

O movimento retangular é composto por quatro movimentos lineares na seguinte ordem: baixo, esquerda, cima e direita. O movimento circular consiste em um

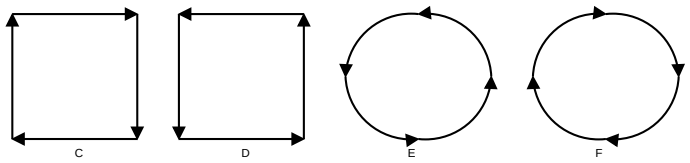


Figura 5.8: movimentos realizados para geração do vídeo sintético.

movimento circular com raio constante e velocidade angular constante por *frame*. Os vídeos são armazenados e, posteriormente, processados por meio da implementação dos algoritmos estudados.

Para testar os métodos que utilizam movimentação 3D (*zoom in/out*, rotação horária e anti-horária) foi utilizado uma implementação similar, empregando sub-amostragem e interpolação para fazer a simulação dos efeitos citados.

5.6.2 Avaliação da Trajetória

Para avaliar a eficiência na detecção de movimento 2D, os deslocamentos dos algoritmos simulados são comparados com os deslocamentos que geraram o movimento do vídeo sintético. Para realizar esta comparação, os deslocamentos (Δx,Δy) são utilizados para calcular a posição absoluta (x, y) do movimento dadas por

$$x\left(t\right) = \sum_{k=1}^{t} \Delta x\left(t\right) = \Delta x\left(t\right) + x\left(t-1\right), \tag{5.1}$$

$$y\left(t\right) = \sum_{k=1}^{t} \Delta y\left(t\right) = \Delta y\left(t\right) + y\left(t-1\right), \tag{5.2}$$

em que t indica o instante de tempo do quadro. A Figura 5.9 mostra uma comparação gráfica do movimento reconhecido com o movimento sintético (referência). A linha preta sólida representa a posição de referência, a pontilhada representa o movimento estimado e a área desenhada em cinza representa o erro de posicionamento.

Nesta tese, o movimento é feito pela câmera. Neste tipo de movimento, a forma é mais importante do que a amplitude, sendo possível corrigir este problema de amplitude através da aplicação de um ganho variável durante a detecção de movimento. Desta forma, antes de calcular o erro, a posição absoluta estimada (x_e,y_e) por um dado algoritmo e a posição absoluta do modelo de referência (x_r,y_r)

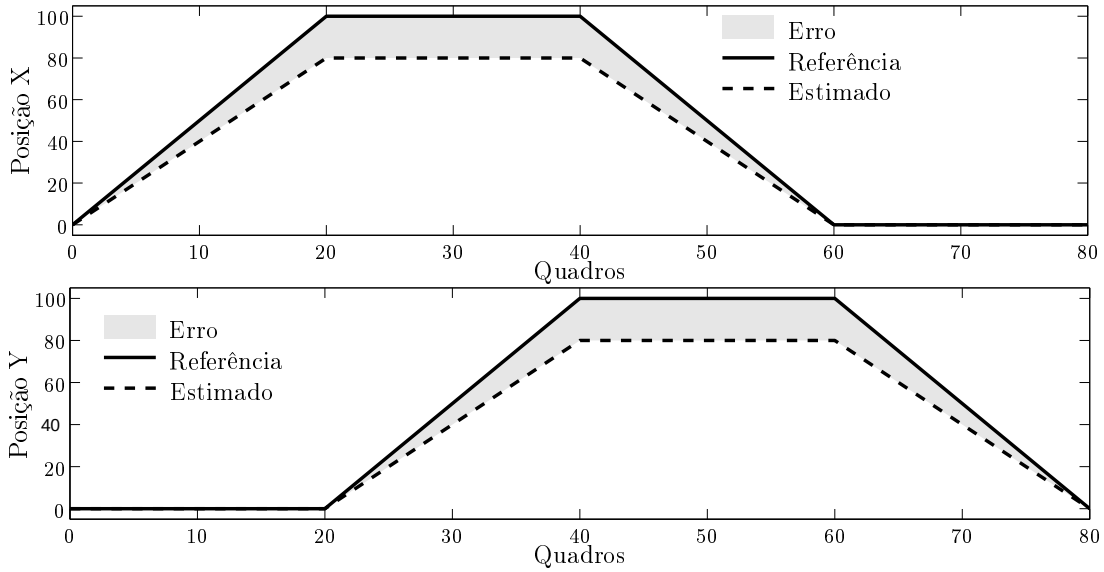


Figura 5.9: comparação gráfica do movimento estimado em relação ao movimento de referência.

são normalizadas através de

$$x(t) = \frac{x(t) - x(1)}{max(x) - min(x)}, \tag{5.3}$$

em que $max(x)$ é a maior posição do movimento e $min(x)$ é a menor posição obtida no movimento. A partir da posição estimada, o erro de detecção de movimento é obtido através do Erro Médio Quadrático (MSE) (POULARIKAS, 1999)

$$MSE = \frac{\sqrt{\sum_{i=1}^{N} (x_e - x_r)^2 + (y_e - y_r)^2}}{N}. \tag{5.4}$$

5.6.3 Esforço Computacional

O tempo de resposta é uma informação do desempenho de qualquer sistema e consiste no tempo necessário entre uma requisição e o instante em que a resposta é fornecida para o usuário (MACHADO; MAIA, 2007).

Segundo NIELSEN (2000) *apud* Paes (2010), o tempo de resposta em sistemas interativos é crítico na forma em que o usuário vai percebê-lo. Com um tempo de resposta menor que um décimo de segundo, o usuário percebe que o sistema está reagindo instantaneamente aos comandos inseridos, caracterizando como tempo real.

Entre 0,1s e 1s o usuário identificam a demora pelo tempo de resposta. Quando o tempo de resposta for maior do que isto, o usuário reconhece uma grande demora na resposta de sistema e pode desistir do uso da interface.

Com isso, ao desenvolver um sistema de interação é de fundamental importância a avaliação computacional dos métodos desenvolvidos, podendo considerar como eficaz o sistema de interação baseado em visão que é capaz de capturar as imagens e detectar os movimentos em menos que 1 milésimo de segundo.

O custo é medido, computacionalmente, pelo tempo médio de execução do método. Foi desenvolvida uma aplicação que executa cada um dos métodos estudados em um período de 20 segundos. Ao final da execução, o sistema apresenta a quantidade de *frames* processados, a média e o desvio padrão do tempo de processamento.

Para cada um dos algoritmos estudados é realizado um estudo comparativo com diferentes tamanhos de imagens e parâmetros que são dispostos em uma Tabela para análise. Esta avaliação deve determinar quais dos métodos desenvolvidos são capazes de responder em tempo real à interação do usuário, ou seja, são capazes de realizar todas as etapas mostradas na Figura 4.1 com tempo menor ou igual a 0,1 ms.

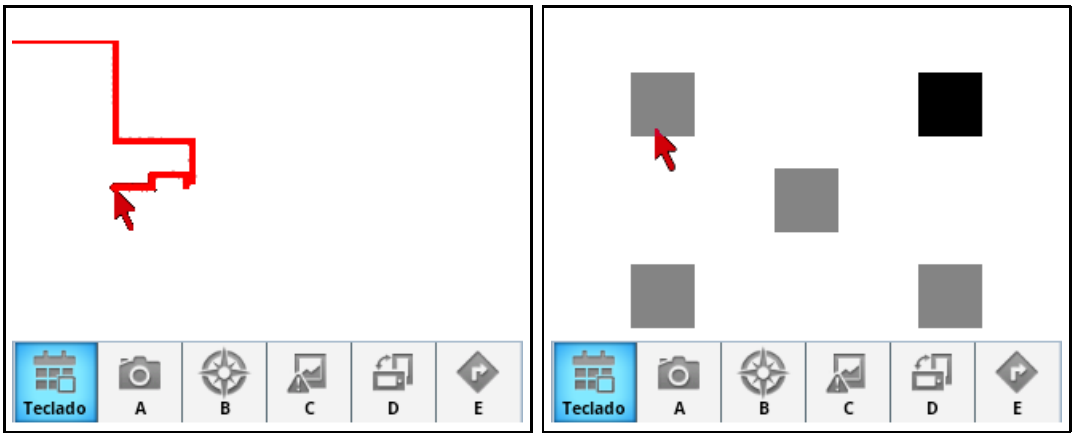
5.6.4 Usabilidade

O tempo de resposta é uma característica importante, mas caso a interface possua um esforço muito grande para realizar a tarefa, ela fica sem utilidade, pois os usuários não utilizariam. Por causa disto, um dos objetivos da tese é desenvolver uma interface de interação capaz de ser facilmente utilizada por usuários que utilizam dispositivos portáteis.

A usabilidade consiste em uma série de características que definem a facilidade com que as pessoas podem empregar uma ferramenta ou objeto (BENYON, 2011). O objetivo dos testes de usabilidade é medir a reação e o desempenho dos futuros usuários ao utilizar a interface. Os métodos estudados são avaliados através da implementação de aplicações, mostradas nas Figuras 5.10 e 5.11.

Estas aplicações utilizam os algoritmos de detecção de movimentos estudados para interagir com um objeto apontador da interface gráfica. Os usuários examinam o resultado de detecção a fim de determinar em qual dos métodos estudados existe

uma melhor interação.



(a) (b)

Figura 5.10: exemplos de aplicações para teste de a) movimentação e b) usabilidade.

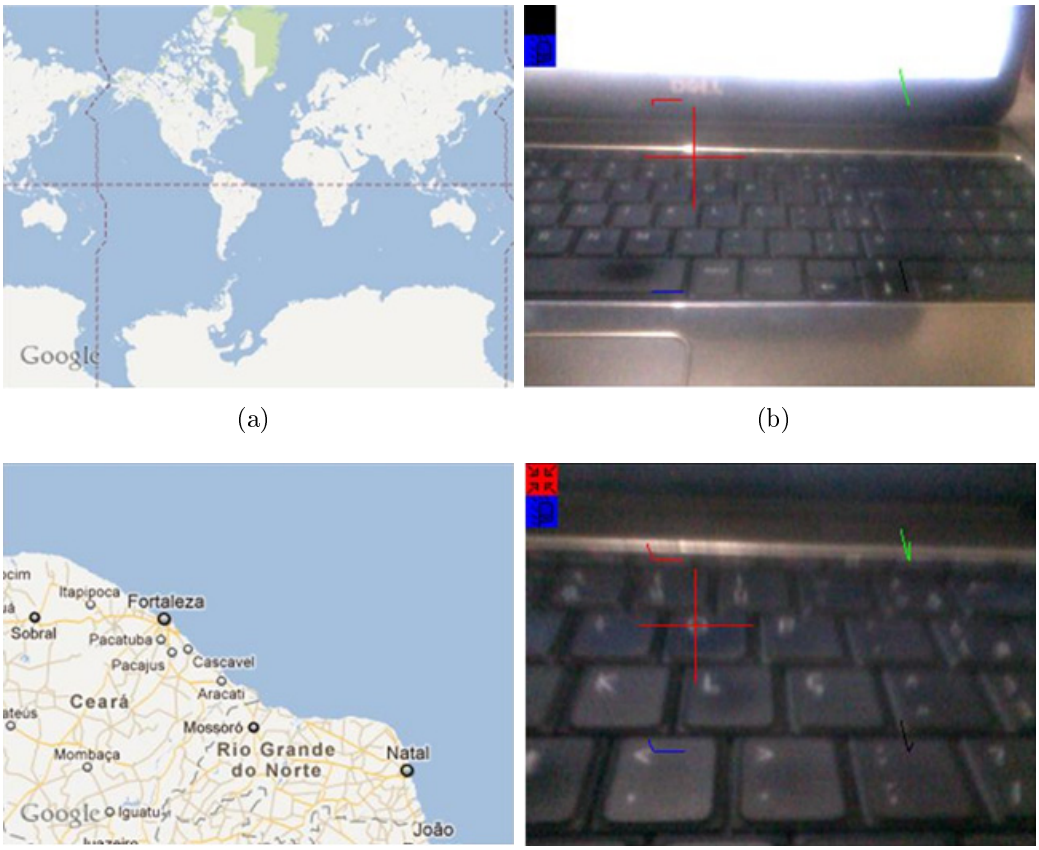
Em todas as aplicações de teste desenvolvidas, ao pressionar o botão menu do dispositivo Android, surge um menu para escolher a forma de interação com o dispositivo. Para possibilitar uma análise imparcial, os métodos a serem analisados foram rotulados com letras: A - Fluxo Óptico (FO) modificado, B - Rohs e Essl (2007), C - (DRAB; ARTNER, 2005) e D - Hannuksela *et al.* (2005). Assim, quando o usuário for realizar o teste de usabilidade da aplicação, o mesmo deverá testar cada um dos métodos estudados.

A primeira aplicação consiste em um sistema para desenhar o rastro do movimento realizado pelo dispositivo, conforme mostrado na Figura 5.10(a). O objetivo desta aplicação é avaliar a precisão da detecção de movimento do dispositivo como entrada de dados para o usuário ao utilizar o sistema como ferramenta de desenho.

A segunda aplicação consiste em um sistema para avaliar a eficiência do tempo de resposta do usuário em realizar uma sequência de movimentos. A aplicação mostrada na Figura 5.10(b) é composta por uma série de quadrados. Nesta aplicação, o movimento do dispositivo faz com que o cursor desenhado na tela se mova. O usuário deve posicionar o cursor sobre os quadrados na sequência indicada na tela. O sistema analisa o tempo necessário para fazer toda sequência pré-determinada, bem como cada movimento linear da sequência.

A terceira aplicação tem como objetivo avaliar a percepção do usuário em uma aplicação de maior porte através da sua utilização com um sistema de mapas para

testar os eventos de detecção de movimentos de translação e os de aproximação e de distanciamento do dispositivo, conforme mostrado na Figura 5.11.



(a) (b)

(c) (d)

Figura 5.11: aplicação de interação com mapas através do movimento. a) mapa mostrado na tela do dispositivo no instante de tempo t b) imagem adquirida no instante de tempo t, c) mapa mostrado na tela do dispositivo no instante de tempo $t+\Delta$ e d) imagem adquirida no instante de tempo $t+\Delta$

A cada instante de tempo, os quadros são adquiridos e quando há uma detecção de movimento, o mapa tem sua posição modificada. Em um determinado instante de tempo, a aplicação exibe uma região do mapa, é mostrado na Figura 5.11 a). A Figura 5.11 b) mostra o quadro adquirido neste instante de tempo.

Ao aplicar um movimento de aproximação da câmera, o mapa tem seu nível de zoom aumentado, conforme mostrado na Figura 5.11 c), pois o sistema detectou um movimento de aproximação entre os quadros mostrados na Figura 5.11 b) e Figura 5.11 d). Os usuários são apresentados à forma de interação proposta, sem terem nenhum conhecimento adicional sobre a metodologia. O formulário utilizado no teste de usabilidade é apresentado no Apêndice A e está dividido em quatro

partes.

A primeira parte do questionário, tem como objetivo conhecer o grau de experiência do usuário em operar dispositivos portáteis. A segunda utiliza a aplicação de desenho, mostrada na Figura 5.10(a), para verificar a intuitividade da interface de interação por movimento e a precisão da detecção de movimentos para interagir com o dispositivo portátil, possibilitando a avaliação de qual método obteve o rastreamento mais próximo ao movimento realizado.

A terceira descreve o teste de tempo de resposta e esforço físico para realizar uma tarefa pré-determinada, utilizando a aplicação mostrada na Figura 5.10(b). A quarta e última parte avalia a satisfação do usuário ao utilizar o método de classificação de movimentos para interagir com o mapa. Os resultados obtidos a partir desta proposta são descritos no próximo capítulo.

Capítulo 6

Resultados

Neste Capítulo são descritos os resultados dos métodos desenvolvidos através da metodologia apresentada nos Capítulos anteriores. Em linhas gerais, os resultados das análises são descritas na seguinte ordem:

i) qualidade de estimação dos vetores de velocidade do método proposto;

ii) comparação da qualidade de estimação de movimento e esforço computacional dos métodos clássicos de Fluxo Óptico (FO) , utilizando vídeos sintéticos e reais;

iii) classificação dos vetores de velocidade para estimação de movimentos de translação em X, Y e Z e rotação horária e anti-horária;

iv) qualidade da estimação da trajetória do movimento dos métodos de detecção de movimentos em dispositivos portáteis empregando vídeos sintéticos;

v) esforço computacional dos métodos de detecção de movimentos em dispositivos portáteis em dispositivos com SOs Android, Symbian e REX;

vi) usabilidade dos algoritmos de detecção de movimentos estudados utilizando as aplicações de teste desenvolvidas em Android;

vii) método proposto de realce para melhoria na detecção de movimentos em ambientes com pouca luminosidade.

6.1 Estimação dos Vetores de Velocidade

Os resultados de estimação de movimento do método proposto são apresentados nesta seção. Inicialmente, são apresentados os vetores de velocidade estimados durante a realização de um movimento linear, em seguida são apresentados os resultados obtidos em movimentos de rotação, aproximação (*zoom in*) e distanciamento (*zoom out*).

Para melhorar a exibição dos vetores de velocidade, as matrizes de velocidade são divididas em 300 regiões com tamanho 16x16, sendo exibida a velocidade média de cada uma dessas regiões.

6.1.1 Movimentos Lineares

Durante um movimento linear para baixo com velocidade de um pixel por *frame*, e adquirida uma sequência de quadros em que dois deles em instantes de tempo consecutivos são apresentados na Figura 6.1.

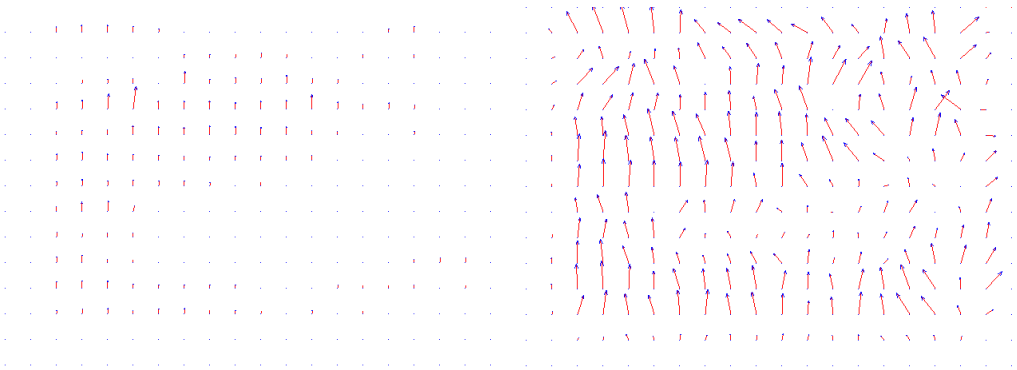
(a) (b) (c)

Figura 6.1: exemplificação da diferença temporal entre dois quadros adquiridos em dois instantes de tempo a) t e b) $t+1$ durante a realização de um movimento com velocidade de um *pixel* por quadro; e c) diferença entre os dois quadros.

Por causa dessa pequena diferença entre os *frames* ser de apenas um *pixel*, visualmente não é possível observar a diferença entre as imagens. Por isso, na Figura 6.1(c) é apresentada a diferença entre os dois quadros.

Apesar disso, o método proposto é capaz de detectar os vetores de velocidade corretamente, conforme mostrado na Figura 6.2. Na Figura 6.2(a) são apresentados os vetores de velocidades, normalizados em função da maior velocidade estimada.

Pode-se perceber que cerca de 41% dos vetores de velocidade são maiores que 10% do que a maior velocidade estimada, mas todos estão apontando para cima. Observa-se que os vetores de velocidade de maior módulo estão próximos às bordas



(a) (b)

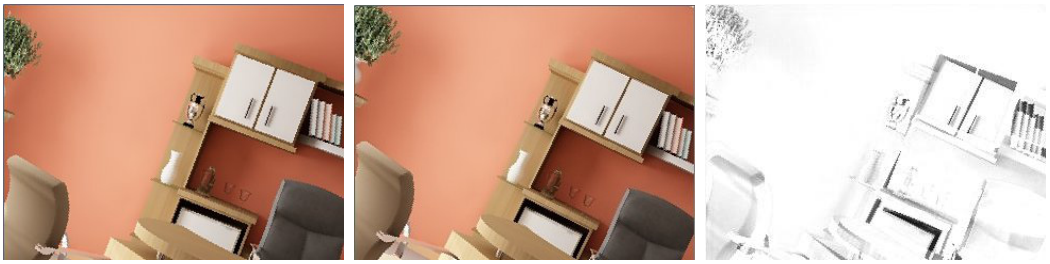
Figura 6.2: FO estimado pelo método proposto normalizado, a) pelo maior valor de velocidade; e b) pela amplitude do próprio vetor.

dos objetos e das regiões em que existem diferenças entre os quadros. Este efeito pode ser observado devido à natureza do FO proposto ser baseado em gradientes.

Para fins de ilustração, a Figura 6.2(b) apresenta os vetores normalizados por sua amplitude. Pode-se observar que apesar de sua baixa amplitude, a maioria também está apontando para a direção correta, resultando em um deslocamento positivo em relação à Y e nulo em relação à direção X, indicando que a câmera está em movimento retilíneo para cima.

6.1.2 Movimentos Rotacionais

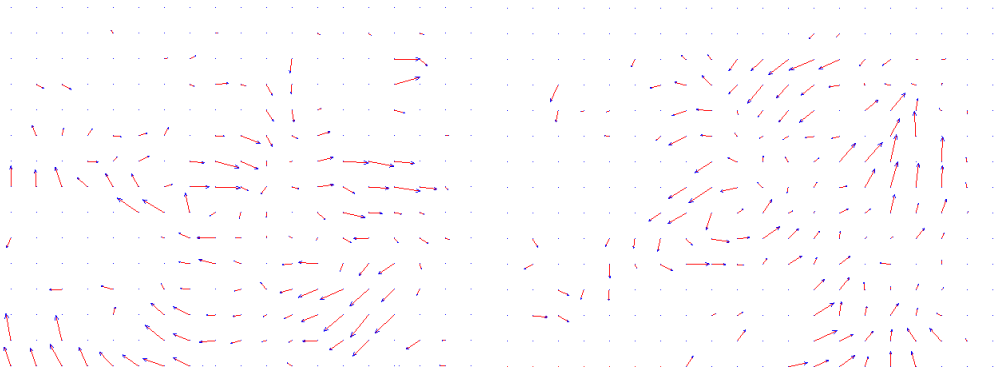
Durante a realização de um movimento de rotação horária em torno do eixo da câmera com velocidade angular de 5 graus por quadro no sentido anti-horário, é adquirida uma sequência de quadros em que dois *frames* consecutivos são apresentados na Figura 6.3.



(a) (b) (c)

Figura 6.3: diferença temporal durante a realização de um movimento de rotação com velocidade angular de 5 graus por quadro. a) e b) quadros adquiridos em dois instantes de tempo consecutivos; c) diferença temporal.

Os vetores de velocidade estimados para movimentos de rotação horária e anti-horária são mostrados na Figura 6.4. Observando as Figuras, pode-se perceber que os vetores de velocidade possuem uma distribuição radial e que formam círculos fechados, similares a distribuições mostradas nas Figuras 4.6(c) e 4.6(d).



(a) (b)

Figura 6.4: campo de velocidades estimadas para movimentos de rotação a) horária e b) anti-horária.

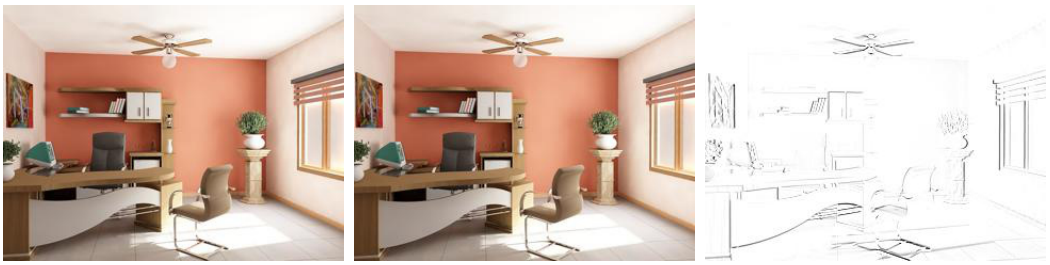
A diferença entre a distribuição prevista e estimada é gerada devido à inexistência de informações de textura e diferenças em determinadas regiões da imagem. Na Figura 6.4(a), apenas 52% dos pixels são capazes de gerar vetores de velocidade superiores a 10% da maior velocidade, nos demais a velocidade é inferior a isto e considera-se como ruído ou regiões sem movimentos.

6.1.3 Movimentos de aproximação e distanciamento

Similarmente ao realizado nas seções anteriores, nesta seção, são apresentados os resultados obtidos da detecção de movimentos feitos através da aproximação e do distanciamento da câmera, realizados a partir do redimensionamento de uma imagem estática.

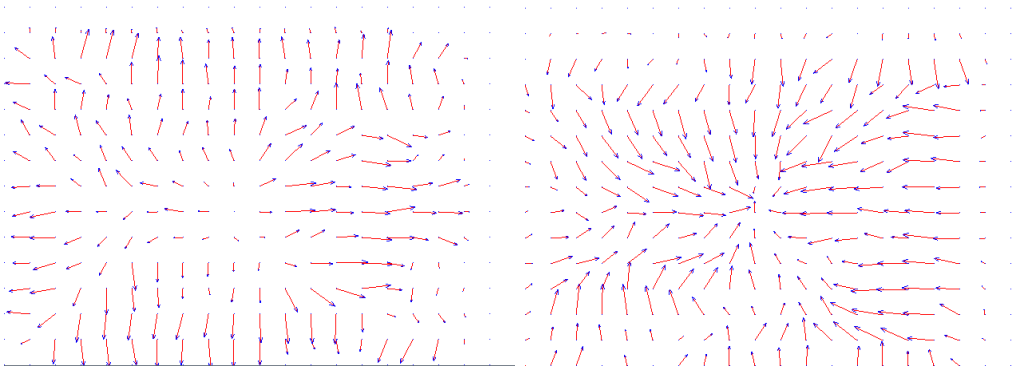
Dois *frames* adquiridos durante a operação de *zoom-in* são apresentados na Figura 6.5. Não é possível observar visualmente a diferença entre as imagens, mas é possível perceber um número maior de diferenças na Figura 6.5(c) que nos movimentos analisados anteriormente.

Os vetores de velocidade estimados durante os movimentos de translação no Eixo Z (aproximação e distanciamento) são mostrados na Figura 6.6. Pode-se perceber que tais vetores estão bem próximos ao previsto nas Figuras 4.6(a) e 4.6(b).



(a) (b) (c)

Figura 6.5: diferença temporal durante a realização de um movimento de rotação com velocidade angular de 5 graus por quadro. a) e b) quadros adquiridos em dois instantes de tempo consecutivos; c) diferença temporal.



(a) (b)

Figura 6.6: campos de velocidade estimados durante o movimento de a) aproximação e b) distanciamento.

Em outras palavras, durante a operação de distanciamento, os objetos vão ficando menores e se movem em direção ao centro da imagem. Por isso, os vetores de velocidade convergem em direção ao centro da imagem. Já na operação de aproximação ocorre o contrário, os objetos tornam-se maiores. Por isso, os vetores são iniciados no centro da imagem e apontam para as extremidades da imagem.

Em ambos os casos, a similaridade com o vetor de velocidade, a similaridade entre os vetores de velocidade previsto e estimado é possível, pois, a grande quantidade de diferenças entre as imagens fez com que fosse possível estimar velocidades consideráveis (maiores que 10% do máximo), em quase 85% de todos os pontos da imagem. Além disto, percebe-se que esses 15% de pontos com baixa velocidade localizam-se na região da periferia da imagem, em que existem problemas de estimação de velocidade devido às bordas da imagem.

6.1.4 Conclusões

Através destes exemplos, percebe-se que a restrição proposta para a estimação do FO é capaz de estimar os movimentos em apenas uma interação, utilizando dois quadros consecutivos, o que diminui consideravelmente o tempo computacional do método de Fluxo Óptico.

6.2 Comparação dos Algoritmos de Fluxo Óptico clássicos

Nesta seção, os algoritmos de FO são avaliados para dois tipos de movimentos: retangular e circular, os quais foram adquiridos conforme descrito na seção 5.6.1 e avaliado por meio da metodologia descrita na seção 5.6.2.

Os algoritmos de estimação de FO são descritos na Tabela 6.1. Para cada método são testados diferentes valores de parâmetros intrínsecos, os quais são mostrados, respectivamente, na terceira e segunda coluna desta Tabela. Além destes parâmetros, é avaliada a influência da velocidade mínima de detecção v_{min}, a qual é usada para calcular o deslocamento nas equações 4.8, cujos valores variam conforme o vídeo analisado.

Tabela 6.1: algoritmos de estimação de FO avaliados, seus respectivos parâmetros de configuração e valores.

Fluxo Óptico	Parâmetros	Valores
Baseado em Correlação	Tamanho da Janela	3 ; 5 ; 7
	Deslocamento Máximo	3 ; 5 ; 7 ; 9 ; 11 ; 15 ; 17
Horn e Schunck (1981)	λ	1 ; 0,3162 ; 0,1 ; 0,03162
Lucas e Kanade (1981)	τ	1 ; 0,5 ; 0,1 ; 0,05 ; 0,01 ; 0,001
FO Proposto	λ	1 ; 0,3162 ; 0,1 ; 0,03162
	α	0,1; 0,5 ; 0,9

O estudo paramétrico do algoritmo proposto na seção 4.2 para o movimento retangular com velocidade de um (01) *pixel* por *frame* é mostrado na Figura 6.7. Os valores λ utilizados são mostrados na Tabela 6.1 e os valores de v_{min} são 0,01 ; 0,1 e 0,7.

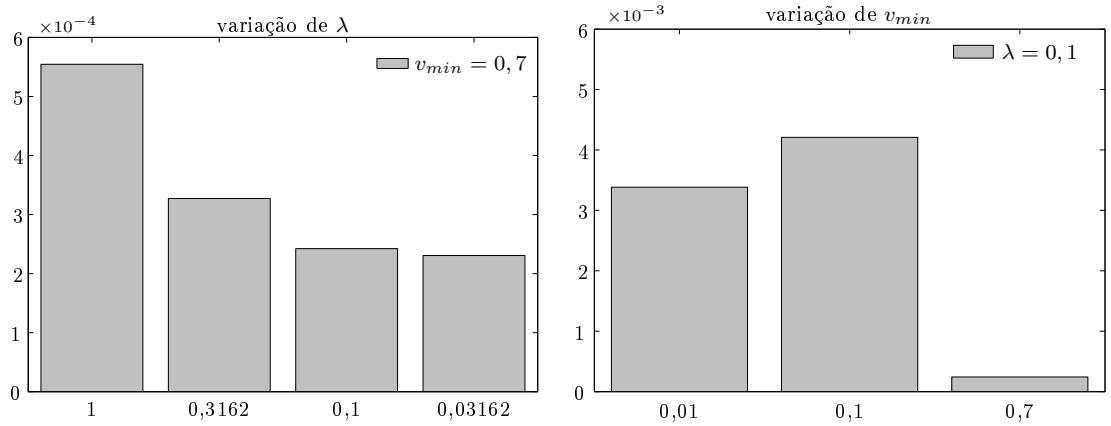


Figura 6.7: MSE de estimação de trajetória utilizando o algoritmo proposto.

Os resultados mostram que a detecção melhora com a diminuição de λ. Além disto, observa-se que o vetor v_{min} filtra a imperfeição do movimento estimado, possuindo os melhores valores para $v_{min} = 0,7$.

Um estudo paramétrico similar é realizado para cada um dos métodos mostrados na Tabela 6.1. Este estudo pretende determinar o melhor conjunto de parâmetros para cada um dos movimentos estudados.

6.2.1 Movimento Retangular

Os resultados obtidos para o movimento retangular para velocidades de 1, 2, 5 e 10 *pixels* por *frame*. O erro de estimação de trajetória (MSE) em função da velocidade do movimento simulado é mostrado na Figura 6.8. Nesta Figura, o eixo Y representa a ordem do MSE, sabendo-se que, quanto menor a ordem, mais próximo do movimento de referência está o movimento estimado. Observando a Figura, percebe-se que para movimentos lentos (velocidade de um *pixel* por *frame*), o algoritmo proposto possui menor erro (2,8 $\times 10^{-4}$). Os demais algoritmos possuem erro na ordem de 10^{-3}.

O valor de MSE aumenta com a velocidade do movimento simulado para todos os métodos analisados. O aumento do erro é esperado devido o fato de qualquer método de estimação de FO requerer que o padrão de brilho de uma região da imagem permaneça constante, conforme estabelecido na equação 2.5. Dentre os algoritmos, o método baseado em correlação mantém a ordem do erro (10^{-2}), possuindo a melhor detecção para movimentos mais rápidos.

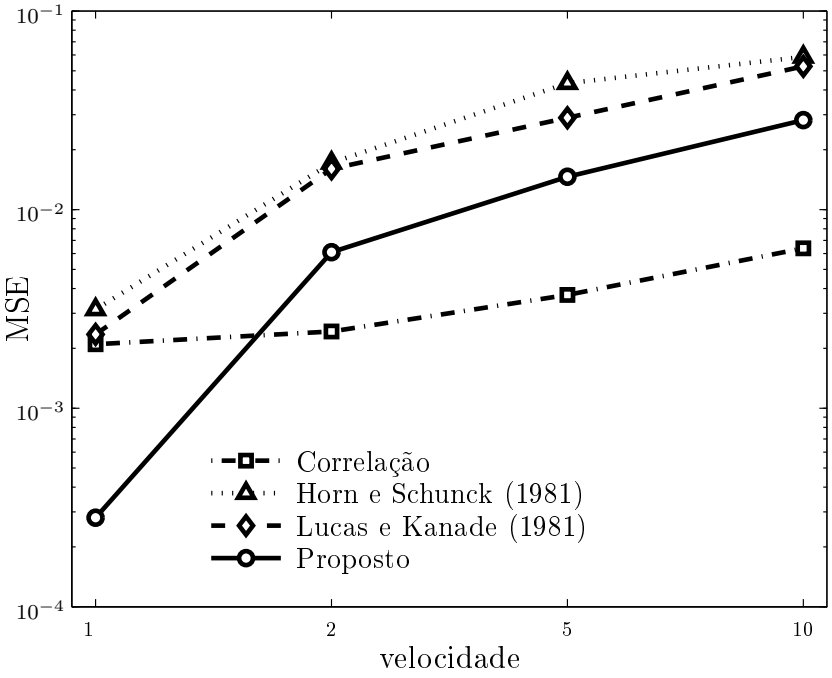


Figura 6.8: comportamento do MSE em função da velocidade simulada para o movimento retangular.

Desta forma, os algoritmos de correlação e o método proposto possuem as melhores detecções enquanto que os piores resultados são obtidos pelos métodos propostos por Horn e Schunck (1981) e Lucas e Kanade (1981). Estes resultados mostram que a modificação no método de Horn e Schunck (1981) melhorou a detecção do movimento da câmera.

Para demonstrar a eficiência da estimação de erro proposta na seção 5.6.2, a comparação visual da trajetória do movimento estimado e de referência para o movimento retangular é mostrada na Figura 6.9. Esta Figura representa a detecção de um movimento retangular com velocidade de um *pixel* por *frame*. De acordo com esta Figura, a trajetória estimada com menor MSE (pelo algoritmo proposto), o qual é bem próximo do movimento simulado (referência).

O segundo melhor método (baseado em correlação), consegue detectar corretamente aproximadamente 3/4 do movimento, isto é, durante os dois primeiros segmentos lineares a detecção é bem próxima da referência e nos dois últimos começa a ter uma pequena diferença entre o movimento estimado e a referência.

Já os métodos de Horn e Schunck (1981) e Lucas e Kanade (1981) começam a gerar falhas de detecção durante o segundo segmento linear, exibindo um movimento similar a um retângulo deformado. Observa-se que quanto menor o valor de MSE,

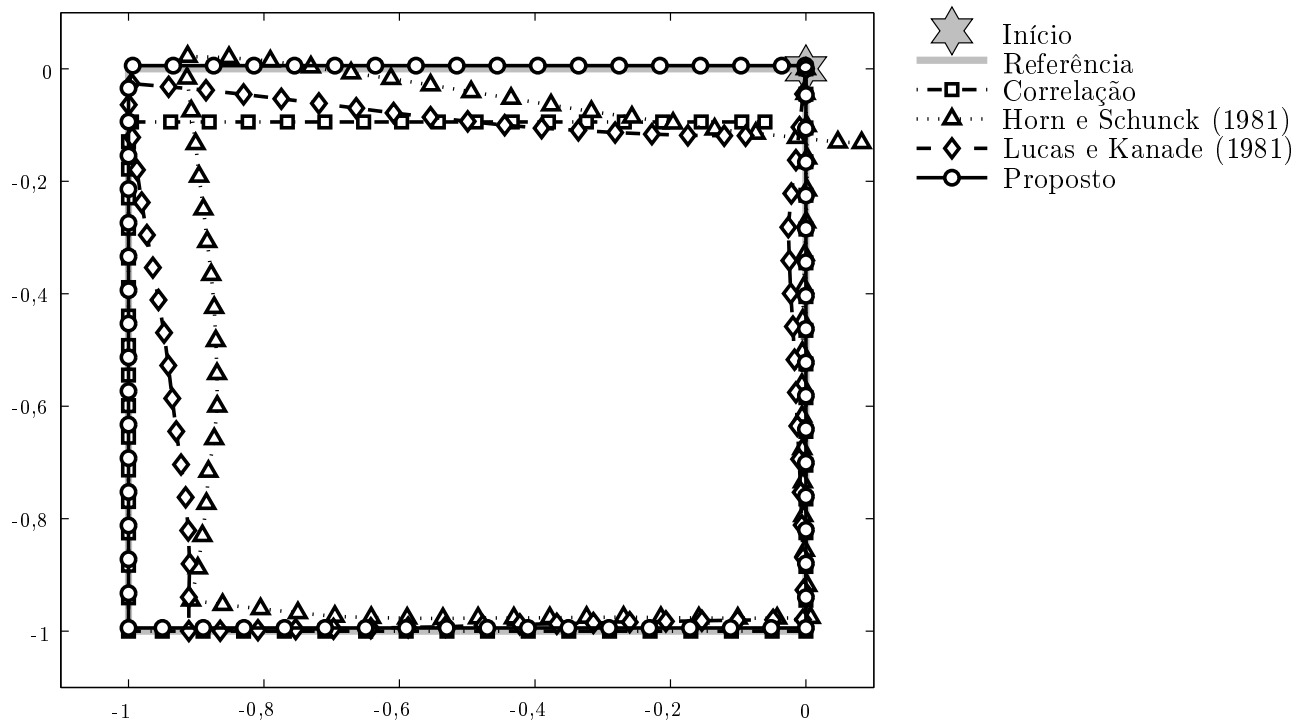


Figura 6.9: comparação da forma da trajetória estimada para movimento retangular horário com velocidade de um pixel por *frame*.

mais próxima da referência é o movimento estimado, de maneira a mostrar que esta característica consegue determinar com eficiência a qualidade da estimação do movimento.

De acordo com a Figura 6.9, a trajetória estimada pelo método proposto é muito próxima da referência, tendo uma aparência quase quadrada. O método proposto por Lucas e Kanade (1981) e de Horn e Schunck (1981) estimam com pequeno erro somente durante o inicio do movimento até o começo do terceiro movimento que compõe o movimento retangular.

6.2.2 Movimento Circular Uniforme

Os resultados obtidos para o erro de estimação de trajetória (MSE) durante a realização de um movimento circular com velocidade angular de 1, 2, 5 e 10 *pixels* por *frame*, é mostrado na Figura 6.10. Estes resultados mostram um comportamento similar ao obtido através do estudo realizado com o movimento retangular, em todas as velocidades analisadas. Isto é, para velocidade igual a 1 *pixel* por *frame*, a melhor detecção é obtida pelo algoritmo proposto e para velocidades maiores, o método

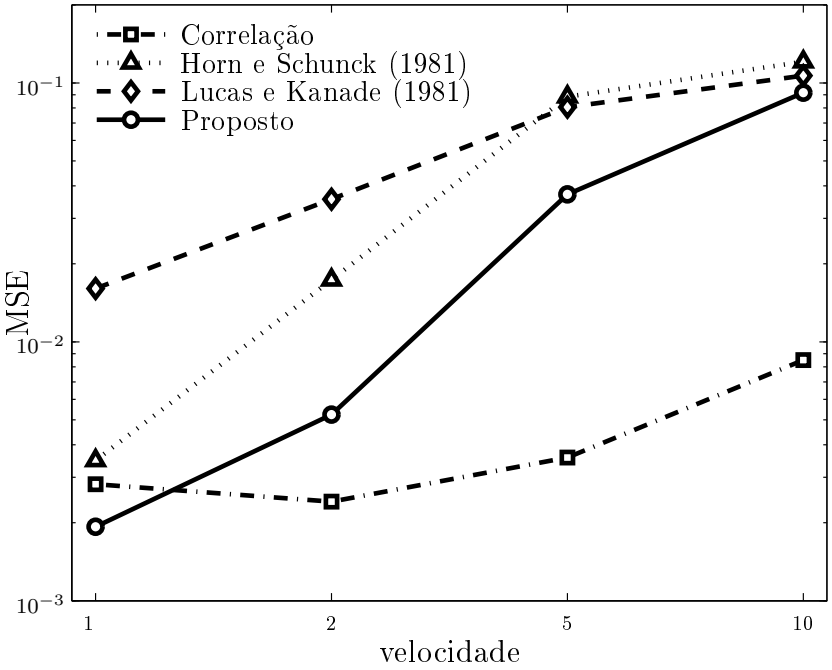


Figura 6.10: comportamento do MSE em função da velocidade simulada para o movimento circular.

baseado em correlação possui as melhores detecções.

Por fim, os resultados reforçam a importância das restrições propostas, as quais conseguiram melhorar a estimação do movimento em relação ao método de Horn e Schunck (1981) para todas as velocidades.

6.2.3 Avaliação de Esforço Computacional

Os resultados obtidos a partir da avaliação de esforço computacional dos algoritmos estudados mostram que dentre os algoritmos estudados, o método baseado em correlação sofre uma grande variação de desempenho em função da variação de parâmetros.

Para valores pequenos de deslocamento máximo, o Simulink é capaz de processar (em média) 21 quadros por segundo (FPS), enquanto que para valores grandes, a capacidade de processamento cai para 8 FPS. Fixando um valor de deslocamento máximo, a taxa de quadros diminui ligeiramente em função do tamanho de janela. Para este método, os parâmetros de tamanho de janela igual a cinco (5) e o deslocamento máximo igual a dezessete (17) resultaram no melhor resultado de detecção.

A análise comparativa do esforço computacional dos algoritmos estudados é

mostrada na Tabela 6.2. Nesta Tabela, o esforço computacional é medido através do conjunto de parâmetros que geraram os melhores resultados de detecção mostrados nas subseções 6.2.1 e 6.2.2.

Tabela 6.2: taxa de FPS dos algoritmos de estimação de FO estudados.

	Tamanho do *Frame*		
Método	128x96	176x144	320x240
Correlação	15,47	8,02	2,853
Horn e Schunck (1981)	48,75	25,54	9,249
Lucas e Kanade (1981)	98,45	54,04	20,38
Proposto	70,74	37,03	14,02

O método proposto por Horn e Schunck (1981) é capaz de processar (em média) 25 *frames* de tamanho 176 x 144 em um segundo. Contudo, o tempo de processamento é altamente dependente do número de iterações para convergência. Quando o *frame* modifica-se pouco, o método necessita interagir menos vezes, reduzindo o tempo de processamento. Quando o *frame* possui uma maior variação, o algoritmo necessita realizar várias iterações, possuindo a capacidade de processar entre 22 e 70 FPS.

Os métodos de Lucas e Kanade (1981) e o proposto não sofrem grandes variações de custo computacional em relação aos parâmetros, sendo capaz de processar respectivamente 54,04 e 37,03 FPS um vídeo de *frames* de tamanho 176 x 144.

Observando a Tabela 6.2, pode-se perceber que o método de Lucas e Kanade (1981) possui o melhor desempenho computacional em todas as resoluções analisadas. Contudo, não possui uma estimação de movimento eficiente. Já o método baseado em correlação, a boa qualidade de detecção tem como preço um alto custo computacional, sendo capaz de processar poucos FPS. Este fato corresponde às expectativas, pois, aumentando-se a complexidade de um algoritmo, aumenta o seu esforço computacional.

O método proposto conseguiu diminuir em média 47.3 % o custo computacional em relação ao método de Horn e Schunck (1981) e está associado a uma eficiente detecção de movimento, principalmente para movimentos sutis ou altas taxas de aquisição.

6.2.4 Vídeo Real

Os resultados de estimação de movimento de um vídeo adquirido pela câmera do celular são mostrados na Figura 6.11.

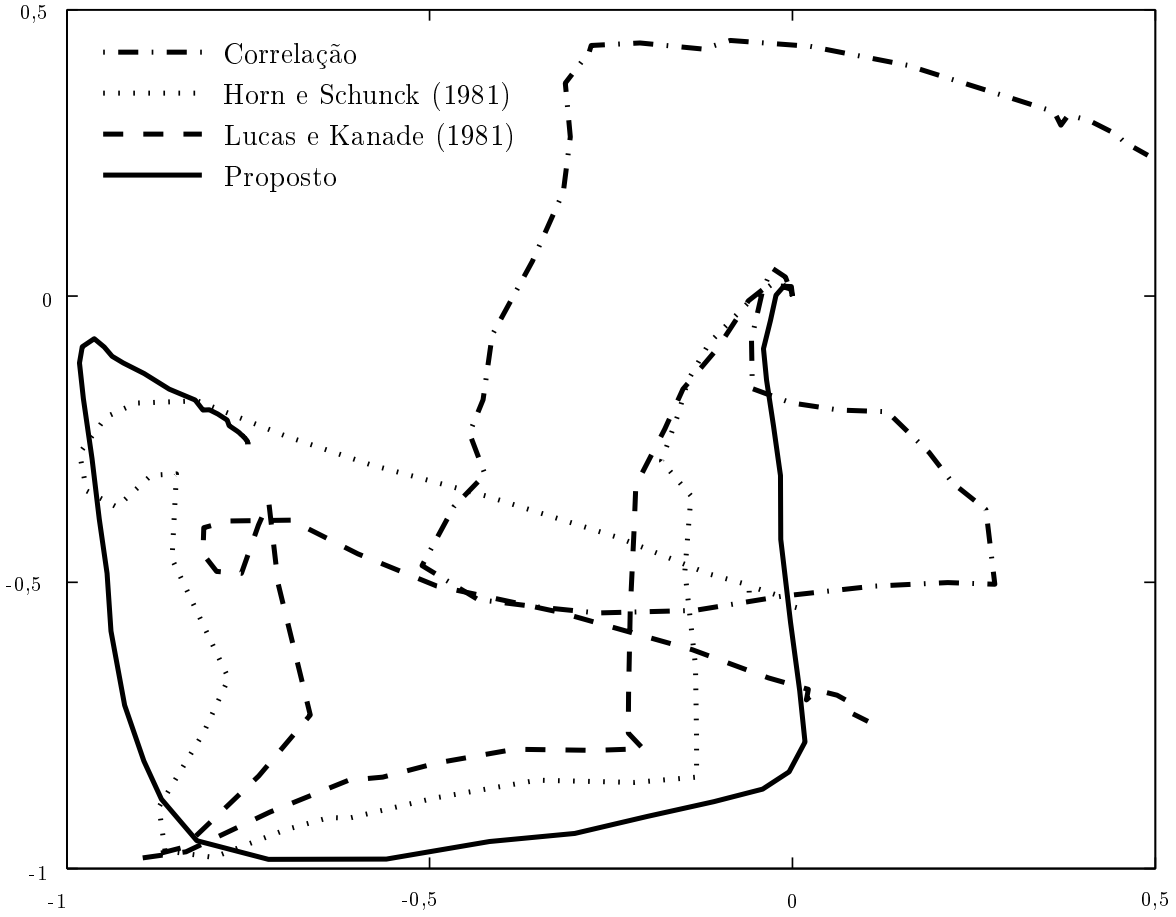


Figura 6.11: trajetória estimada do vídeo adquirido.

Neste experimento, uma sequência de movimentos lineares na seguinte ordem baixo, esquerda, cima, direita é realizada com um dispositivo portátil. Os resultados mostram que dentre os métodos estudados, o método proposto possui a trajetória mais similar a um quadrado, composto por três segmentos lineares (baixo, esquerda e cima) com transição suave entre os movimentos. O método baseado em correlação possui um desempenho intermediário, produzindo boa detecção para os três últimos trechos. Contudo, no primeiro a detecção é insatisfatória. Os métodos de Horn e Schunck (1981) e -Lucas e Kanade (1981) possuem a transição do movimento abrupta e também uma maior distorção da trajetória durante os movimento para cima, em relação ao método de correlação.

Com base nos resultados, pode-se observar que o método proposto associa uma boa estimação de movimento e baixo custo computacional, mostrando-se um excelente candidato a ser embarcado em dispositivos portáteis.

6.2.5 Sub-amostragem

Visando a melhoria computacional da estimação de movimentos, avalia-se o efeito da sub-amostragem dos vetores de velocidade do método proposto, cujo resultado é mostrado na Figura 6.12. Sem realizar a sub-amostragem, os vetores de velocidade são calculados pixel-a-pixel.

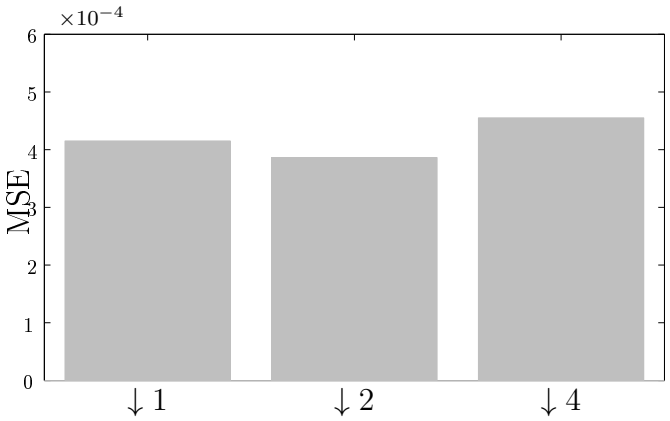


Figura 6.12: trajetória estimada do vídeo adquirido.

Ao realizar uma sub-amostragem de X pixels, os vetores de velocidade são estimados a cada X pixels, o que evita o cálculo de $X^2 - 1$ vetores de velocidade, tornando o método mais rápido à medida que se aumenta o efeito de sub-amostragem. Com base nesta Figura, observa-se que, mesmo ao aplicar uma sub-amostragem de quatro (04) pixels, os valores de MSE permanecem próximos, significando que o algoritmo é pouco sensível à modificação da sub-amostragem, podendo ser utilizado com um valor de sub-amostragem sem prejuízo da detecção de movimento.

6.3 Classificação dos vetores de velocidade

Na seção 4.3 foi descrito o algoritmo proposto para detectar movimentos de rotação, aproximação e distanciamento utilizando os vetores de velocidade estimados pelo FO. Para verificar sua eficiência, são utilizados os mesmos vídeos sintéticos utilizados na seção anterior.

Durante esta verificação, a imagem é dividida em seis (6) regiões, conforme mostrado na Figura 6.13(a). Então, calcula-se a média dos vetores de velocidade em cada uma dessas regiões. As direções das velocidades médias são comparadas

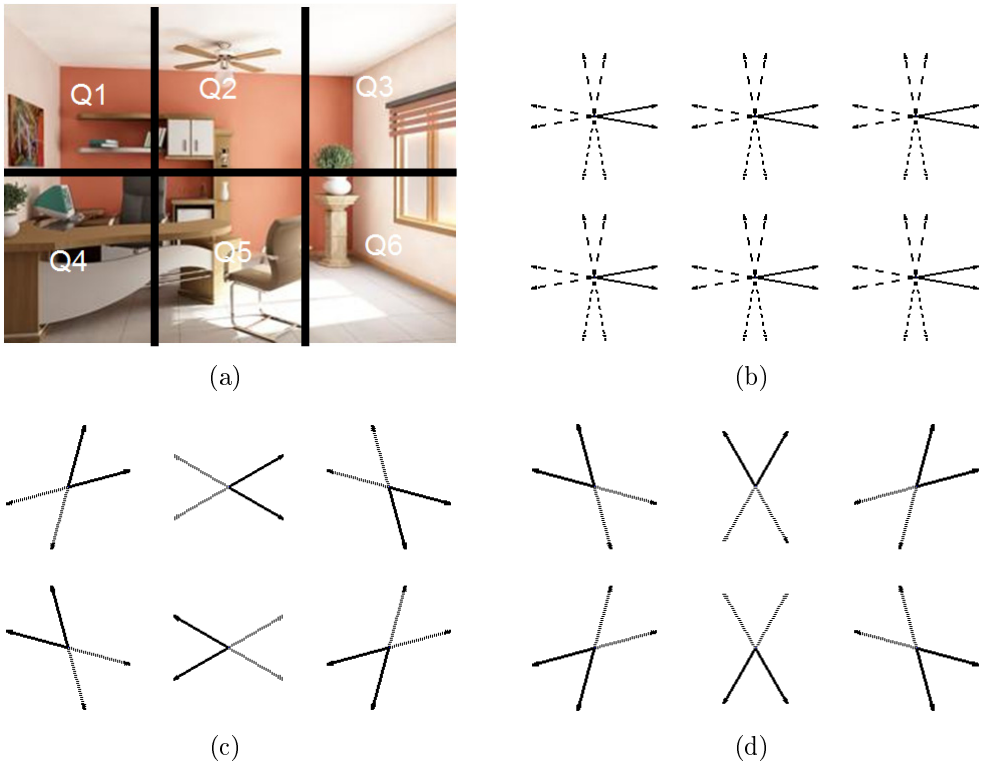


Figura 6.13: a) regiões utilizadas para o cálculo da média de velocidades e modelos de vetores de velocidade para os movimentos b) direcionais (cima, baixo, esquerda e direita) c) rotacionais (horária e anti-horária) e d) aproximação e distanciamento.

com um conjunto formado por modelos de distribuição de vetores de velocidades para cada um dos movimentos, considerando como movimento estimado aquele que possuir a maior quantidade dentro de um limiar de tolerância.

Conforme descrito na seção anterior, observa-se que a estimação de velocidades é mais uniforme nos movimentos direcionais que os demais movimentos. Por isso, foi considerado como limiar de tolerância uma variação entre 10^o e 30^o para os demais movimentos. Nas Figuras 6.13(b) e 6.13(d) as regiões de tolerância dos movimentos direcionais, rotação e aproximação e distanciamento, respectivamente.

Os gráficos que indicam a quantidade de vetores que estão dentro do limiar de tolerância para três vídeos são mostrados na Figura 6.14. O primeiro vídeo consiste em um movimento retangular, composto por quatro movimentos lineares (cima, baixo, esquerda e direita). No segundo e no terceiro vídeo são realizados

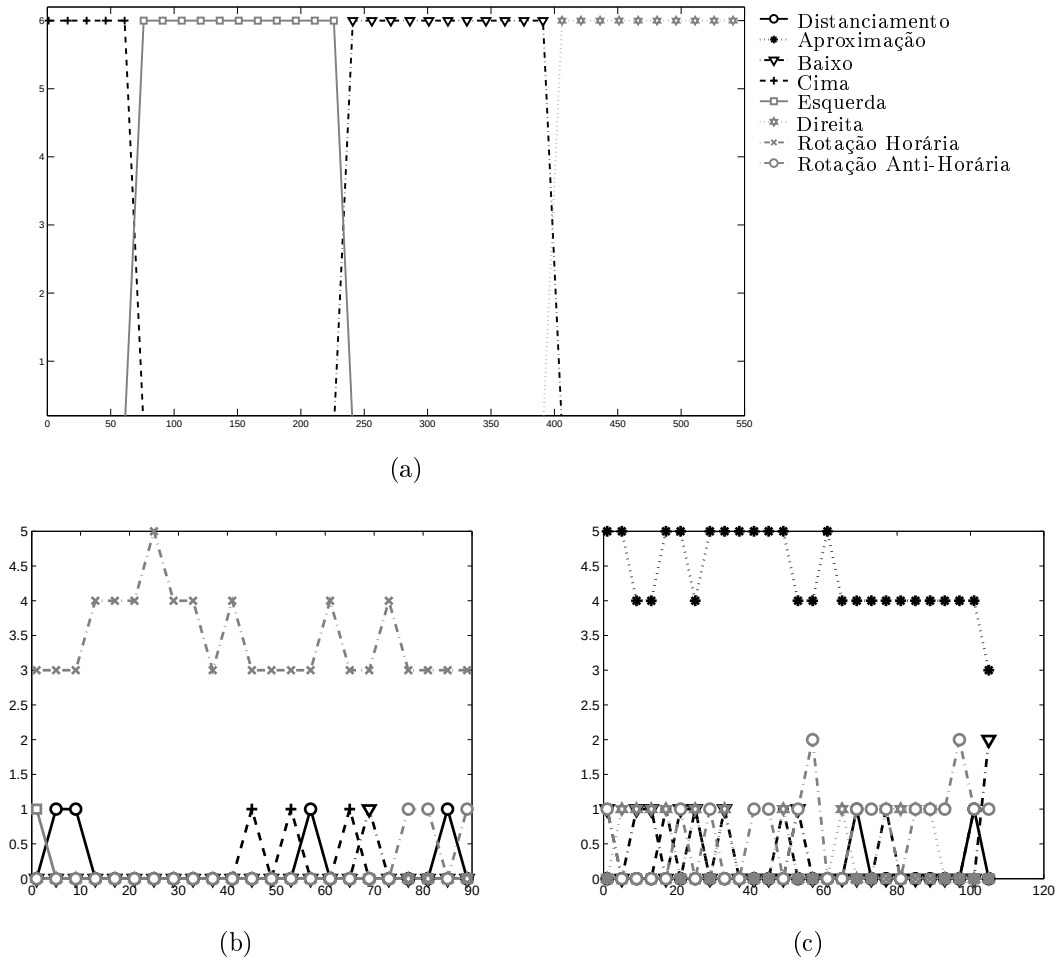


Figura 6.14: quantidade de vetores de velocidade dentro dos limiares de tolerância para cada um dos movimentos analisados. a) movimentos lineares, b) rotação e c) distanciamento.

movimentos de distanciamento e rotação horária, respectivamente.

A contagem de vetores durante a realização do primeiro movimento é mostrada na Figura 6.14(a). Durante a realização do vídeo, entre cada movimento linear existe uma região de transição. Conforme se pode perceber na Figura, durante a realização do movimento linear todos os vetores médios estão dentro do limiar de tolerância.

Devido ao baixo limiar de tolerância, durante a realização dos movimentos lineares, apenas os movimentos direcionais são reconhecidos, não existindo vetores de velocidade dentro do limiar de tolerância para os demais movimentos.

Durante a realização dos movimentos de rotação, existem, em média, 3 vetores dentro do limiar de tolerância dos movimentos de rotação horária, conforme mostrado na Figura 6.14(b). Para os demais movimentos, apenas um vetor de

movimento é reconhecido dentro do limiar de tolerância.

Durante a realização do movimento de distanciamento foi observado que, em média, a quantidade de quatro velocidades dentro do limiar de tolerância. Enquanto que os demais movimentos possuem apenas um dentro do limiar de tolerância. É possível observar nas Figura 6.13(b) e 6.13(d) que existem regiões em comum dos movimentos de aproximação e distanciamento para os movimentos direcionais para cima e para baixo, justificando-se a detecção de pelo menos um vetor de velocidade direcional.

Os vetores de velocidade média estimadas neste vídeo e sua comparação com as regiões de tolerância são mostrados na Figura 6.15. Observa-se que nas regiões

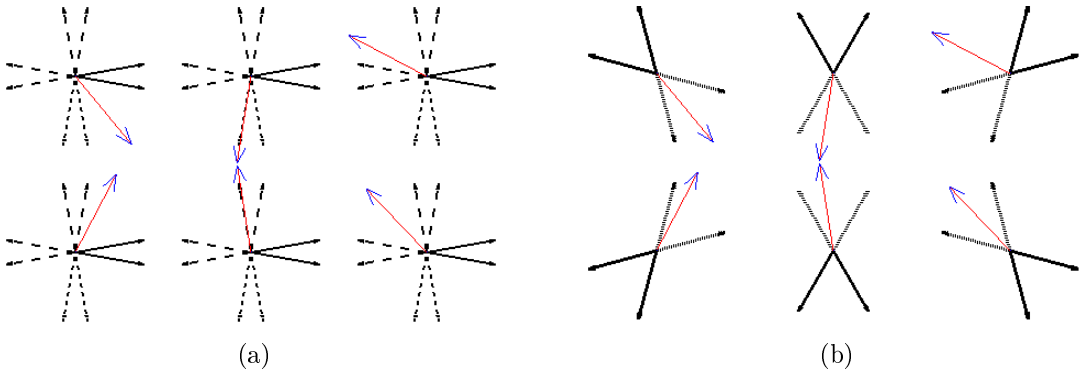


Figura 6.15: distribuição dos vetores de velocidade média e sua comparação com as regiões dos movimentos a) direcionais e b) aproximação e distanciamento.

Q2 e Q4 (mostradas na Figura 6.15), os vetores médios estão dentro da região de tolerância de movimentos lineares e rotacionais simultaneamente.

Já na região Q3, houve um pequeno desvio de direção durante a estimação, fazendo com que o mesmo fosse reconhecido como pertencente a região do movimento para baixo e não do movimento de distanciamento. Isto foi gerado devido a pouca quantidade de informação dentro das regiões.

Através deste resultado, observa-se que pelo menos 50% das médias de velocidade são corretamente estimadas para todos os movimentos, indicando que este limiar. Este resultado indica que é suficiente para distinguir os movimentos realizados. Por consequente, determinar o movimento realizado em todos os quadros analisados.

Devido à natureza do movimento linear, sua detecção é mais fácil, pois o método sempre é capaz de identificar uma maior quantidade de correlação entre as médias

estimadas e a distribuição de velocidade esperada. Em contrapartida, o limiar de tolerância é o menor dentre os movimentos estudados.

Já o movimento de rotação necessita de um maior nível de tolerância durante a classificação, pois este movimento apresenta uma menor capacidade de detecção, conforme observado na Figura 6.14(b). Apesar disto, o método foi capaz de classificar corretamente em todos os quadros processados, não existindo confusão de detecção para os demais movimentos.

Vale a pena ressaltar que além desta informação, é de fundamental importância o cálculo do módulo de velocidade, o qual é obtido através da soma do módulo dos componentes de velocidade de cada pixel da imagem. Esta informação diz respeito à amplitude do movimento realizado, quando este valor é maior que um limiar, pode-se considerar que houve movimentos significativos e neste momento o movimento é classificado.

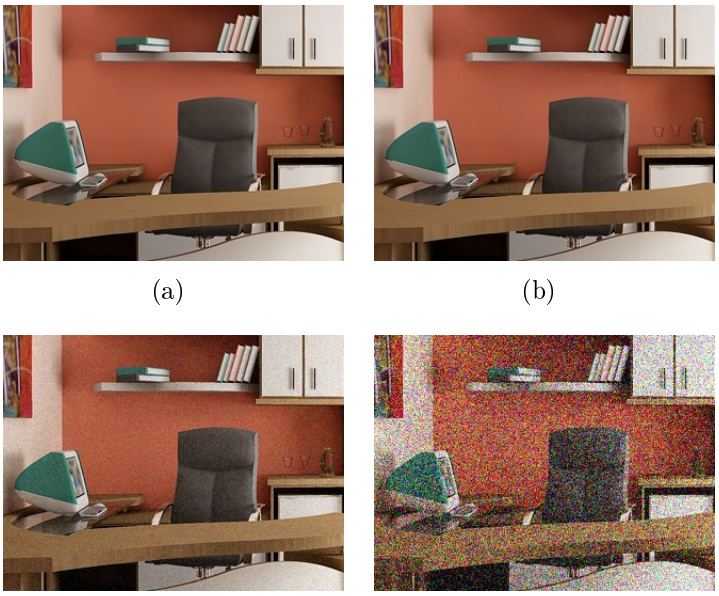
6.4 Algoritmos de Detecção de Movimentos em Dispositivos Portáteis

Esta seção apresenta os resultados de detecção de movimentos, utilizando métodos presentes na literatura especializada de detecção de movimento em dispositivos portáteis, similar ao realizado na seção anterior. Para tanto, é avaliada a qualidade da estimação de movimentos do algoritmo proposto e são comparados com os métodos propostos por Hannuksela *et al.* (2011), Drab e Artner (2005) e Rohs e Essl (2007).

Nesta seção é feita ainda uma analise similar à realizada na seção anterior, em que são avaliados o Erro Médio Quadrático (MSE) da estimação de trajetória para os movimentos retangular e circular com velocidade de um pixel por *frame*, utilizando vídeos com condições diferentes de ruído gaussiano, mostrados na Figura 6.16.

A partir do vídeo sem ruído, em que um *frame* é mostrado na Figura 6.16(a), são inseridos o ruído gaussiano com média 0 e diversas condições de intensidade, em cada um dos três canais de cor. Na Figura 6.16(a) não é possível perceber visualmente a inserção de ruído, devido à pequena variância deste, considerando-se uma aquisição com baixo nível do mesmo.

Para avaliar o ruído de aquisição de dispositivos portáteis, utilizam-se duas imagens adquiridas com os dispositivos iPAD2 e Galaxy 5, mostradas na Figura 6.17.



(a) (b)

(c) (d)

Figura 6.16: imagens de quadros dos vídeos sintético a) sem ruídos e b) com ruído gaussiano com desvio padrão de um nível, c) 10 níveis e d) 50 níveis.

Através das imagens, é possível calcular o desvio padrão do ruído a partir do valor do desvio padrão de regiões homogêneas. Com esta análise, obtém-se o valor de 9.98 e 13, respectivamente, muito próximo do ruído simulado.

(a) (b)

Figura 6.17: fotos adquiridas através da câmera traseira do a) iPAD2 e b) Galaxy 5

Em todos os casos, os métodos são avaliados em sua condição de maior custo computacional. O método proposto realiza o cálculo do FO em todos os pixels da imagem adquirida. O método de Rohs e Essl (2007) utiliza blocos de tamanho 1x1 e realiza a correlação em cada um dos pixels da imagem. Nas seções a seguir é avaliado o desempenho na estimação de movimentos lineares e circulares uniforme.

6.4.1 Movimentos Lineares

Os resultados obtidos para o MSE de estimação do movimento quadrangular para os métodos implementados no dispositivo portátil, os quais estão resumidos na Tabela 6.3.

Tabela 6.3: MSE de estimação de movimento utilizando vídeo sintético dos dispositivos portáteis em função do ruído gaussiano para um movimento retangular.

Método	$\sigma = 0$	$\sigma = 1$	$\sigma = 10$	$\sigma = 50$
Proposto	0.0062	0.0063	0.0076	0.0242
Rohs e Essl (2007)	0.0030	0.0030	0.0030	0.0051
Drab e Artner (2005)	0.0030	0.0023	0.0023	0.0069
Hannuksela *et al.* (2011)	0.0105	0.0110	0.0112	0.0155

Nos testes realizados com o método proposto, é avaliada a variação de valores de MSE ao realizar a sub-amostragem dos vetores de velocidade. No primeiro caso, em que não foi realizada sub-amostragem, o MSE foi igual à 0.0062, já ao utilizar uma sub-amostragem a cada dois pixels, o erro foi de 0.0061. Em outras palavras, o erro inferior neste caso é inferior a 2%, e por causa disto, o método proposto avaliado só calcula o vetor de velocidade a cada 2 pixels, representando uma economia de 75% no custo computacional.

Pelos resultados, nota-se que o método proposto possui um MSE com pequena variação (na ordem de 6×10^{-3}) com ruídos com desvio padrão inferior à 10 níveis. No vídeo com o pior nível de ruído, o método proposto piora consideravelmente, a detecção de movimento, aumentando em quatro vezes o erro de estimação de trajetória.

Apesar disto, os melhores resultados são apresentados no método proposto por Rohs e Essl (2007), que um método baseado em correlação. Com um MSE constante perante pequenos e médios ruídos. Além disto, mesmo com um ruído de alta intensidade, este método possuiu um melhor desempenho do que o método proposto sem condições de ruído. Neste caso, o método realiza o cálculo de correlação em cada um dos pixels da imagem, o que torna o custo computacional bastante elevado, devido ao número de comparações.

O método proposto por Drab e Artner (2005) possui desempenho similar ao método de Rohs e Essl (2007), sendo melhor apenas na situação de baixo e médio

ruído. Mesmo assim, este método possui performance inferior ao método proposto.

O método de Hannuksela *et al.* (2011) apresenta o maior MSE dos métodos estudados. Contudo, permaneceu praticamente constante com ruídos com desvio padrão inferior à 10 níveis. Apesar desse elevado erro, o método conseguiu ser mais eficiente sob a presença do ruído de alta intensidade do que o método proposto.

É sabido que muitos métodos de FO utilizam um filtro gaussiano antes de realizar a detecção de bordas ou a diferenciação no tempo para suavizar os ruídos da imagem. Por causa disto, no caso de existir a necessidade do método proposto ser utilizado sob esta condição extrema de ruído, seria necessário utilizar um filtro passa-baixa de pré-processamento.

Ao aplicar um filtro gaussiano de tamanho 5x5 antes de detectar o movimento, obteve-se uma MSE igual à 0.0137, o que é cerca de 56 % menor do que a estimação de movimento sem a aplicação do filtro, mostrando que com sua aplicação é possível melhorar a detecção de movimento do método proposto.

6.4.2 Movimento Circular

Os resultados obtidos a partir de um vídeo sintético com movimentação circular, cujos resultados estão resumidos na Tabela 6.4.

Tabela 6.4: MSE de estimação de movimento utilizando vídeo sintético dos dispositivos portáteis em função do ruído gaussiano para um movimento circular.

Método	$\sigma = 0$	$\sigma = 1$	$\sigma = 10$	$\sigma = 50$
Proposto	0.0149	0.0121	0.0314	0.0302
Rohs e Essl (2007)	0.0048	0.0081	0.0048	0.0121
Drab e Artner (2005)	0.0069	0.0067	0.0066	0.0042
Hannuksela *et al.* (2011)	0.0211	0.0182	0.0178	0.0138

Os valores de MSE observados para este tipo de movimentação são maiores devido a natureza do movimento. Contudo, mostram um comportamento similar que o observado na seção anterior, em que os melhores resultados são obtidos pelo método de Drab e Artner (2005) e Rohs e Essl (2007).

Da mesma forma, o método proposto possui um desempenho comprometido na presença de ruídos de alta intensidade. Todavia, em baixa intensidade de ruído, o método proposto possui o valor de MSE com uma variação. O método de Hannuksela

et al. (2011) possui uma melhoria no MSE, à medida que o ruído aumenta, há uma pequena variação do MSE. Entretanto, este método não consegue superar os métodos de Drab e Artner (2005) e Rohs e Essl (2007).

6.5 Avaliação Computacional dos Métodos em dispositivos portáteis

Os métodos propostos foram embarcados em três dispositivos portáteis: A706, N95 e Galaxy 5. Em cada um, a câmera é capaz de adquirir os quadros em resolução diferente. No A706 e N95, os vídeos são obtidos na resolução QCIF (176 x 144). Já o Galaxy 5 por possuir processadores e câmeras melhores, o vídeo adquirido possui resolução QVGA (320x240).

Os resultados do tempo de processamento de um *frame* dos métodos estudados são apresentados nas Tabelas 6.5 à 6.7. Nas Tabelas constam o tempo, em milissegundos, que é necessário para processar um quadro da imagem, bem como entre parênteses, o desvio padrão da medida, os quais estão organizados nas seções a seguir.

Inicialmente, a seção 6.5.1 descreve uma análise do esforço computacional do método proposto por Rohs e Essl (2007) modificando, o tamanho do bloco e a forma de sub-amostragem dentro do bloco. Em seguida, a seção 6.5.2 faz um estudo similar para o método proposto e, por fim, a seção 6.5.3 descreve o estudo comparativo do esforço computacional dos métodos de estimação de movimentos estudados.

6.5.1 Método proposto por Rohs e Essl (2007)

Alguns métodos estudados possuem parâmetros que estão diretamente associados ao custo computacional. Conforme descrito na seção 3.1.4, o método de Rohs e Essl (2007) possui dois parâmetros básicos: o tamanho do bloco (B) e a sub-amostragem dentro do bloco (realizado a cada N pixels).

Um estudo do esforço computacional do método de Rohs e Essl (2007) em função do tamanho do bloco (1x1, 2x2, 4x4 e 8x8) é mostrado na Tabela 6.5.

Em todos os casos, observa-se que ao realizar a correlação de cada um dos pixels da imagem, o custo computacional é extremamente elevado. Em todos os aparelhos são capazes de processar cerca de 3 quadros por segundo. A medida que o tamanho

Tabela 6.5: tempo de processamento (em ms) do método de Rohs e Essl (2007) com diferentes tamanhos de blocos sem sub-amostrar os pixels.

		Galaxy 5	
Método	A706	Normal	Modo Avião
pixel à pixel	259 (69)	355 (75)	344 (49)
blocos 2x2	61 (26)	99 (27)	96 (21)
blocos 4x4	26 (20)	40 (18)	37 (15)
blocos 8x8	20 (16)	21 (10)	20 (11)

de bloco aumenta, há uma diminuição do custo computacional, devido a diminuição brusca da quantidade de pixels utilizados para a correlação.

Ao utilizar blocos de tamanho 2x2, evita-se a estimação de deslocamento em 75% dos pixels. Por esta razão, em cada uma das análises, houve uma diminuição em 1/4 do tempo de processamento em todos os dispositivos. Ao utilizar blocos de tamanho 4x4 e 8x8, evita-se o cálculo de deslocamento em 93% e 98%, ou seja, o custo computacional de ambos não está relacionado ao custo computacional do cálculo de correlação.

A análise de deslocamentos é feita a partir de um *frame* em níveis de cinza. Assim, o tempo de processamento neste caso é fortemente relacionado à conversão entre o espaço de cor do quadro adquirido (YCbCr) e o espaço de cor necessário (nível de cinza).

Nos resultados, observa-se um elevado desvio padrão na estimação de tempo (superiores à 10 ms). Observa-se que quanto maior o custo computacional do método de Rohs e Essl (2007), maior o desvio padrão da estimação de tempo. Isto ocorre porque os todos os Sistema Operacionals (SOs) dos dispositivos são multiprogramáveis.

Apesar do Android e *Real Time Executive* (REX) possuírem escalonadores de diferentes naturezas (circular e prioridades), quando o aparelho encontra-se no modo padrão, os desvios de estimação de tempo são próximos. Este fato é causado devido às preempções geradas por eventos do sistema.

Assim, quando se utiliza blocos de tamanho 1x1, o tratamento de eventos e a atualização de dados presentes na tela são bastante afetados, o que causa um aumento no tempo de resposta do dispositivo. Ao utilizar blocos grandes, (4x4

ou 8x8), percebe-se uma diminuição considerável da variação de tempo. Isto ocorre porque os vários eventos que estão sendo tratados pelo sistema operacional interrompem a execução da rotina analisada.

Ao ativar o modo avião do Galaxy 5, houve uma considerável diminuição no desvio padrão de medição de tempo, indicando que o dispositivo passa a gerenciar menos recursos, diminuindo o desvio em 68% ao avaliar a correlação de cada um dos *pixels* da imagem.

Um estudo do esforço computacional do método de Rohs e Essl (2007) em função da forma de sub-amostragem realizada é mostrado na Tabela 6.6, considerando que

Tabela 6.6: tempo de processamento (em ms) do método de Rohs e Essl (2007) com diversos valores de sub-amostragem dos pixels.

		Galaxy 5	
Método	A706	Normal	Modo Avião
sem sub-amostragem	26 (20)	40 (18)	37 (15)
com sub-amostragem de 2 pixels	25 (19)	38 (16)	32 (15)
com sub-amostragem de 4 pixels	25 (18)	37 (15)	32 (15)

os blocos são de tamanho 4x4, ou seja, $B = 4$.

Neste caso, a diferença de esforço computacional está relacionada ao cálculo da média do bloco. No A706, observa-se uma menor variação de estimativa de tempo. Ao realizar uma sub-amostragem de 75% ou 50% dos pixels do bloco, há uma variação de apenas 1 ms de em relação ao cálculo com todos os pixels do bloco.

Nota-se que em ambos os casos, praticamente não há, estatisticamente, uma variação do esforço computacional ao utilizar a sub-amostragem de dois (2) e quatro (4) pixels não há uma variação de esforço computacional. No Galaxy 5 houve uma variação de 1 ms que está dentro da margem de tolerância da medida.

Observou-se que no Galaxy 5 houve uma maior variação do esforço computacional que no A706 ao utilizar a amostragem por causa da diferença de resolução de imagem. O A706 adquire 77% de pixels a menos que o Galaxy 5. Desta forma, ao evitar o cálculo de deslocamentos em 50% dos *pixels* da imagem, propiciou uma diminuição de 14% no tempo de processamento.

6.5.2 Método proposto

Nesta seção, o método proposto é avaliado em função do parâmetro que influencia diretamente o custo computacional. Conforme descrito na seção 4, a quantidade de vetores de velocidade sub-amostrados irá modificar a quantidade de pontos em que o fluxo óptico será calculado.

O tempo médio necessário para processar um quadro em cada um dos dispositivos portáteis em função da forma de sub-amostragem dos vetores de velocidade é apresentado na Figura 6.18. No primeiro caso, o fluxo óptico é calculado em todos os pontos da imagem e no último caso, o FO é calculado em apenas 6 % dos pixels da imagem.

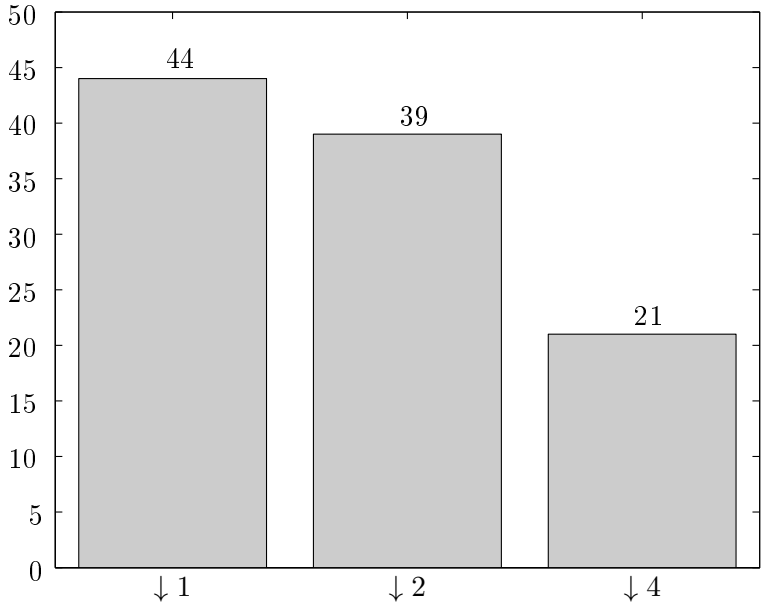


Figura 6.18: tempo de processamento (em ms) do método proposto com diferentes tamanhos de blocos sem sub-amostrar os pixels.

Através dos resultados, observa-se que na situação em que há maior custo computacional, o método proposto possui um custo computacional pequeno (44 ms), o que resultaria em uma taxa de processamento 22 quadros por segundo. Neste caso, existe a possibilidade do dispositivo gastar 66 ms para demais funções desde a aquisição de imagens e realizar as demais rotinas do SO para garantir o processamento dentro do limite de usabilidade (100 ms).

Ao aumentar a sub-amostragem, observa-se a diminuição do custo computacional

no limite em que, ao calcular o FO em 25%, o custo computacional cai pela metade. No Android, observa-se que ao realizar o processamento em todos os pixels da imagem, o custo computacional é elevado. Assim, o método proposto é capaz de processar apenas 6 quadros por segundo. Apesar deste elevado custo, o método proposto é cerca de 52 % mais rápido que o método de Rohs e Essl (2007) nas mesmas condições de sub-amostragem e complexidade computacional.

Conforme descrito na seção 6.2.5, apesar de variar a quantidade de vetores sub-amostrados, a qualidade da estimação de trajetória não se altera. Assim, o método proposto pode ser configurado para calcular o FO apenas 25 % dos pixels da imagem sem prejudicar a detecção de movimentos. Neste caso, o método atende com sucesso o requisito básico de custo computacional, possuindo o tempo de processamento por *frame* de 41 ms em média.

Isto mostra que o método do FO proposto é capaz de ter um custo computacional reduzido e pode ser utilizado como forma de interação para os dispositivos portáteis analisados.

6.5.3 Avaliação dos métodos estudados

A comparação dos esforços computacionais dos métodos estudados está resumida na Tabela 6.7. Na formulação desta Tabela, o método de Rohs e Essl (2007) realiza a estimação de deslocamentos utilizando blocos de tamanho 4x4 sem realizar a sub-amostragem dentro do bloco.

O método proposto realiza a estimação do FO em 25% dos pixels da imagem. O método de Drab e Artner (2005) realiza a projeção de todos os pixels da imagem e avalia os deslocamentos dentro de um intervalo de -16 a 16. O método de Hannuksela *et al.* (2011), por sua vez, utiliza os 16 blocos de tamanhos 6x6 mais representativos da imagem para calcular os deslocamentos.

Tabela 6.7: tempo de processamento (em ms) dos métodos estudados.

Método	A706	N95	Galaxy Y	Galaxy 5
Restante de processamento	133 (48)	200 (50)	56 (26)	36 (30)
Proposto	39 (23)	13 (5)	29 (9)	46 (21)
Rohs e Essl (2007)	26 (20)	15 (7)	26 (8)	40 (18)
Drab e Artner (2005)	18 (7)	4 (2)	18 (7)	24 (9)
Hannuksela *et al.* (2011)	98 (31)	33 (8)	41 (16)	78 (22)

Dentre os métodos estudados, o método proposto por Hannuksela *et al.* (2011) possui o maior esforço computacional. Em qualquer dos dispositivos testados, o mesmo possui a capacidade de processar mais que 10 quadros por segundo. Considerando, que além deste tempo, é necessário incluir o tempo de aquisição de imagens. Desta forma, no A706, o seu desempenho é abaixo do ideal para funcionamento como método de interação.

Ao aplicá-lo em um dispositivo mais moderno, mesmo aumentando a resolução da imagem, o tempo de processamento é reduzido consideravelmente. Ao avaliar o método em um processador mais moderno (Galaxy), o custo computacional do tempo de processamento é quase duas vezes menor. Isto é até esperado, pois o processador é cerca de 5 a 8 vezes mais veloz que o A706.

Nos aparelhos Galaxy os quadros são adquiridos na resolução máxima em que a câmera é capaz de adquirir quadros. Contudo, o custo computacional é inferior a 55 ms, restando 45 ms para o dispositivo adquirir os quadros e realizar o processamento relacionado à gerência do sistema.

Os demais algoritmos possuem um baixo esforço computacional, podendo processar pelo menos 17 quadros por segundo. O método proposto possui o terceiro maior esforço computacional, estando a frente do método de Rohs e Essl (2007) e Drab e Artner (2005).

Apesar disto, esse não é o menor tempo de processamento do método proposto, pois, segundo a sessão anterior, o método proposto ao realizar a sub-amostragem a cada 4 pixels possui tempo de processamento de 9 ms (quase 100 quadros por segundo).

O objetivo da análise é possuir um método que seja capaz de processar os quadros em no máximo 100 ms, atendendo os requisitos mínimos previsto para um algoritmo utilizado como forma de interação com o dispositivo. Com os parâmetros descritos nesta seção o método de Rohs e Essl (2007) possui um custo computacional abaixo de 40 ms, podendo processar 25 quadros por segundo.

Apesar disto, a diminuição do esforço computacional do método de Rohs e Essl (2007) tem um preço grande em termos de estimação de movimento. Na seção 6.4, o método que possui uma ótima qualidade de detecção de movimentos tem o custo computacional elevado (capaz de processar pelo menos 4 quadros por segundo).

O método de Drab e Artner (2005) possui o melhor desempenho dentre todos os

métodos avaliados. O mesmo associa a melhor capacidade de detectar movimentos de translação com o menor custo computacional, sendo capaz de processar pelo menos 40 quadros por segundo, quase 40 vezes mais rápido que o tempo de resposta necessário para ser utilizado como forma de interação.

6.5.4 Avaliação Computacional da classificação de movimento

Nesta seção descrevemos o resultado da avaliação computacional do método de classificação de movimentos baseado no fluxo óptico. Para fins de otimização computacional, as matrizes de velocidade (v_x e v_y) foram divididas em quatro regiões e estas são utilizadas para compará-las com os vetores de velocidade de referência para cada um dos métodos estudados.

Foi testado um número diferente de movimentos a serem reconhecidos. Apesar desta variação, o tempo médio de processamento de um quadro no Galaxy 5 foi de 63 ms, no Galaxy Y foi de 51, com desvio padrão de 20 e 13 ms respectivamente. Como a diferença entre o método de classificação de movimentos com a determinação de deslocamentos pelo FO modificado é apenas a estimação de movimentos, observa-se que cerca de 65% do tempo de processamento do método de classificação de movimento é causada pela estimação do FO. Em outras palavras, o método proposto o reconhecimento mais elaborado causou apenas um aumento de 15ms no tempo de processamento.

Por causa disto, observa-se que os dois métodos propostos atendem os requisitos de esforço computacional para construir uma interface de interação por movimento, sendo necessária sua avaliação de usabilidade.

6.5.5 Variações devido às diferenças entre dispositivos

Durante a realização desta avaliação, algumas observações puderam ser feitas. Devido a forma de gerenciamento de memória dos sistemas operacionais, ao executar os algoritmos no A706, não ocorreram limitações de memória devido ao mesmo estar rodando dentro do ciclo de tarefas, dividindo os recursos do sistema operacional.

Ao executar os métodos nos demais dispositivos (N97 e Galaxy 5), foi necessário uma otimização de uso de memória, pois as mesmas trabalham em modo protegido, possuindo maiores limitações de acesso à recursos. Na Tabela 6.7 são mostrados o tempo de processamento em dois celulares Android, um de 600 MHz com 256 MB (Galaxy 5) e 800 MHz com 384 MB (Galaxy Y).

Ao executar os métodos propostos, foram utilizados cerca de 50 % da RAM do Galaxy Y. Comparativamente, o Galaxy Y é capaz de processar os quadros à uma velocidade de cerca entre 25 e 30 % mais rápido que o Galaxy 5. Isto foi gerado devido aos 200 MHz e 184 MB que o Galaxy Y possui a mais que o Galaxy 5. Um ponto positivo do Galaxy 5 é a existência dos cursores e botões para interação com o dispositivo, enquanto que o outro dispositivo só possui 3 botões, conforme mostrado na Figura 6.19.

(a) (b)

Figura 6.19: cursores dos dispositivos a) Galaxy 5 e b) Galaxy Y.

Por causa disto, pode-se utilizar qualquer um dos métodos estudados como forma de interação com o Galaxy 5, servindo como forma alternativa para a manipulação do dispositivo. Na próxima seção, serão descritos os resultados obtidos através da análise de usabilidade dos métodos.

6.6 Testes de Usabilidade

Nesta seção, são descritos os resultados dos testes de usabilidade dos métodos propostos nesta tese. O teste foi realizado com um grupo reduzido de voluntários que em sua maioria nunca tiveram contato com algum sistema de interação baseado em visão. As características principais dos usuários consistem em 10 jovens de 18 a 30 anos, estudantes da área de computação, com bons conhecimentos da computação e contato com programas e Internet.

Os testes de usabilidades foram realizados em duas seções de duas horas em uma sala de aula de práticas de laboratório em computação. No primeiro momento do teste, descobriu-se que, em sua maioria, os usuários utilizam os dispositivos portáteis por pelo menos 1 hora e meia por dia. Apenas dois usuários utilizam o dispositivo por uma grande quantidade de tempo, superior a 5 horas por dia.

Dentre os entrevistados, 57 % nunca utilizaram qualquer interface baseada em acelerômetro, apenas 28 % dos usuários possuía dispositivo com acelerômetro, mas cerca de 86% deles possuía um dispositivo portátil com câmera.

Após concluir a etapa de conhecimento do perfil de usuário, foi entregue um dispositivo em que o usuário teve a oportunidade de utilizar a aplicação de desenho (mostrada na Figura 5.10(a)). O objetivo desta seção de testes foi verificar se os usuários conseguiam compreender o objetivo da interface proposta. Todos os voluntários analisados foram capazes de perceber a detecção de movimentos na interação. Nenhum dos usuários foi contra a utilização da interface e em sua maioria, perceberam a forma de interação como auto-explicativa e de fácil utilização, comprovando sua elevada aceitação dos usuários.

Ao questionar os usuários sobre a forma de reconhecimento de movimentos, 43% dos usuários pesquisados acharam a qualidade de detecção de movimentos acima do que eles próprios esperavam, o que levou ao pensamento de que o método estava associado à outra forma de detecção de movimento.

Após a realização desta etapa de familiarização com a interface baseada em movimentos, foi solicitado ao usuário avaliar a detecção de seis tipos de movimentos (mostrados na Figura 5.8) em cada um dos algoritmos analisados. A maioria dos usuários (71%) percebeu que o método proposto (I) possuía o menor tempo de resposta dentre os demais resultados. Enquanto que, o método de Drab e Artner (2005) foi percebido como melhor por 28% dos usuários. Contudo, em termos de detecção de movimento, o algoritmo (III) proposto por Drab e Artner (2005) mostrou uma melhor detecção dos movimentos (mostrou um melhor comportamento para os movimentos I, II, V e VI). Já o método proposto se mostrou melhor para a detecção dos movimentos quadrangulares (III e IV).

Com base nesta análise, percebe-se que apesar do método de Drab e Artner (2005) possuir um menor tempo de processamento que o método proposto, os usuários não perceberam esta diferença, devido ao tempo de processamento ser menor do que o perceptível pelo ser humano, indicando que o método proposto atende com satisfação o tempo de resposta do usuário.

Após conhecerem e utilizarem a detecção de movimento de cada um dos métodos analisados, foi solicitado à realização da avaliação de desempenho na realização de uma tarefa. Foi solicitado para os usuários indicarem em qual dos métodos foi possível obter o menor tempo para realizar a trajetória na primeira utilização.

Neste cenário, o fluxo óptico modificado e o método de Rohs e Essl (2007) possibilitaram a obtenção do menor tempo de tarefa na primeira tentativa para 43 % dos usuários, realizando a tarefa em no mínimo 13 segundos e no máximo

25 segundos. Na sua maioria, os usuários não perceberam diferenças entre o esforço necessário para interagir com os métodos avaliados. Os poucos usuários que conseguiram perceber alguma diferença no esforço, perceberam que o método proposto necessitava um de menor esforço do que os demais métodos.

No último teste realizado, o método de classificação de movimento foi apresentado ao usuário. Os usuários classificaram o sistema como agradável e divertido. Os usuários relataram que o sistema tinha uma grande precisão na realização dos movimentos de translação e de distanciamento. A detecção dos movimentos foi considerada a cima da média (bom e muito bom).

Apenas dois usuários avaliados não ficaram satisfeitos com a detecção de movimentos de aproximação, a falha de detecção pode estar relacionada à ausência de informação da região imageada pela câmera no momento da realização do movimento. Além disto, em termos de tempo de resposta apenas um usuário não ficou satisfeito com o tempo de resposta. Em seu formulário, o mesmo achou que poderia ser melhorada a sensibilidade do método. Desta forma, este usuário conseguiu perceber a limitação de um sistema de visão computacional, pois a câmera visualizar um ambiente com bordas para que o FO possa detectar os movimentos. Ao final do seu teste, o usuário foi informado desta limitação e sua respectiva solução de apontar o dispositivo portátil para uma região onde existam objetos.

Com base nestes resultados, observa-se que o método proposto propicia uma satisfação do usuário no uso da interface baseada em movimento, podendo ser integrado aos dispositivos dos voluntários analisados, visto que apenas 1 dos usuários possuía aparelho celular sem câmera.

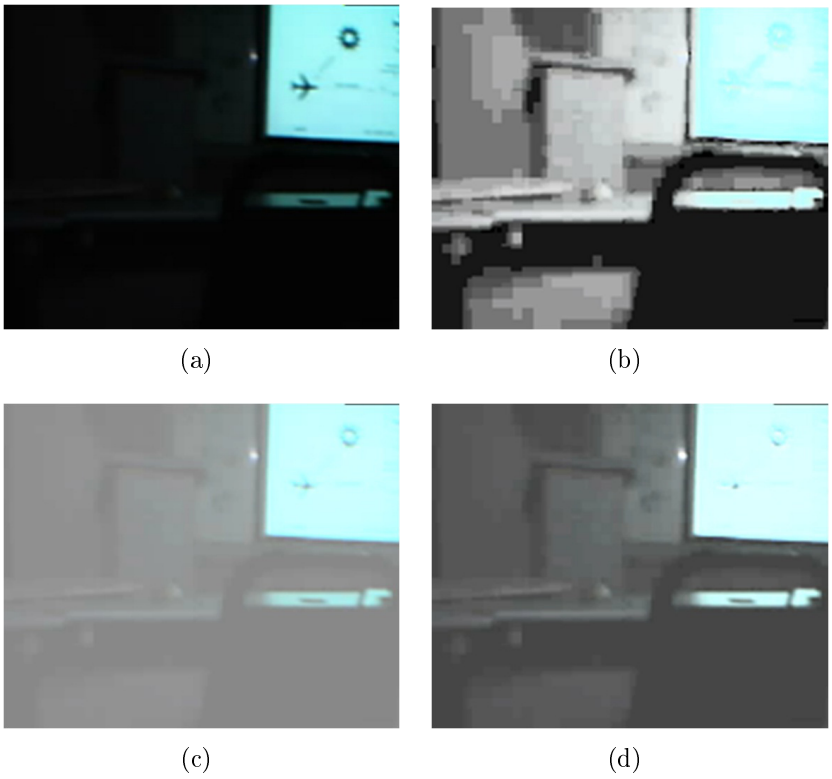
Com base no baixo custo computacional do método e interesse dos participantes, o algoritmo proposto poderia ser facilmente adaptado aos dispositivos portáteis dos usuários que realizaram o teste.

6.7 Realce de Imagens

Nesta seção são descritos os resultados da análise do método de realce de imagens. Em um primeiro momento, é avaliado a melhoria da imagem realizada e em seguida é avaliada sua influência na detecção de movimentos, bem como o esforço computacional que o método acresce na detecção de movimentos.

6.7.1 Análise do realce

É sabido que a detecção de movimentos é altamente influenciada pela diferença temporal e pela presença de bordas na imagem. Quando não existe este tipo de informação, não é possível detectar movimentos nas regiões da imagem. Isto pode ser causado por dois fenômenos, ausência de informação visual das bordas ou baixa iluminação do ambiente. Na segunda situação, é possível realizar alguma melhoria na imagem através do método proposto na seção 4.1. Na Figura 6.20 é apresentada o realce de imagem feito sobre a Figura 6.20(a).

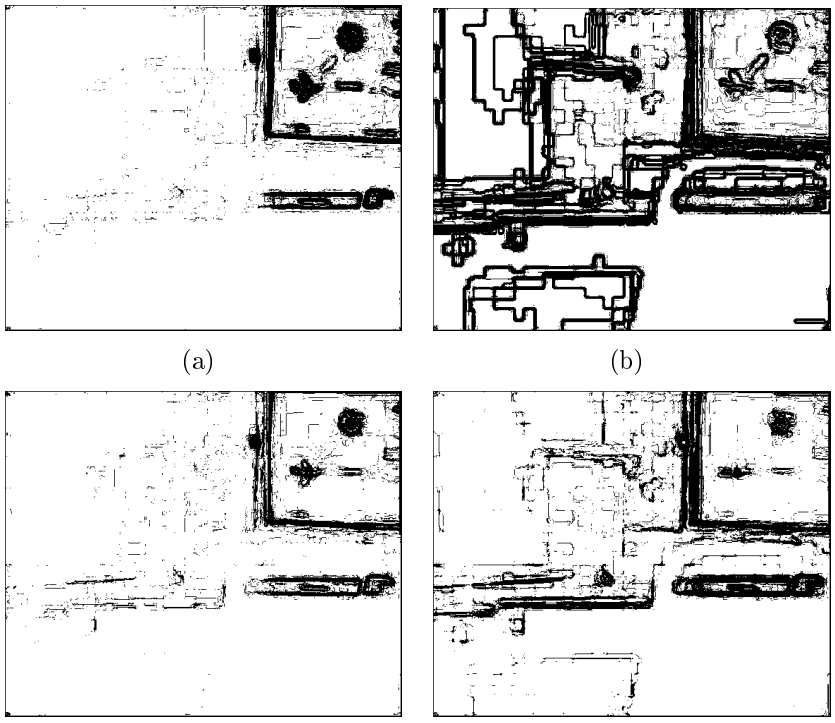


(a) (b)

(c) (d)

Figura 6.20: operação de realce nas imagens, a) imagem original com baixo contraste; b) equalização de histograma; c) ajuste de gamma; e d) ajuste de gamma modificado.

Na imagem original, não é possível observar visualmente os objetos presentes na imagem. Para poder observar melhor as bordas, na Figura 6.21 é mostrado o resultado da aplicação de um filtro de Roberts nas imagens mostradas na Figura 6.20. Em todas as imagens, são mostradas as bordas que possuem intensidade maior que 8 % da borda de maior intensidade. Observando a Figura 6.21(a), percebe-se a existência de poucas bordas na imagem e estas estão concentradas na região de fronteira entre a imagem projetada e o fundo ou dentro da área de projeção.



(a) (b)

(c) (d)

Figura 6.21: detecção de bordas aplicados nas imagens presentes na Figura 6.20.

Com base em uma análise visual, observa-se que na Figura 6.20(b), a equalização de histograma possuiu o melhor ganho de contraste da imagem dentre os métodos estudados, realçando todos os objetos da cena. Por outro lado, após o realce, algumas regiões homogêneas mostram uma grande variação de níveis de cinza, o que faz surgir bordas de grande intensidade na imagem, conforme mostrado na Figura 6.21(b).

Um problema desse efeito é o aparecimento ou desaparecimento de bordas devido à pequenas variações da iluminação, podendo prejudicar a detecção de movimentos a partir do FO.

Os resultados do ajuste gama é mostrado na Figura 6.20(c). A aplicação desse ajuste propícia o pior ajuste de contraste dos métodos estudados, em que os objetos presentes na cena são vistos com dificuldade e, além disto, a imagem tem uma qualidade saturada de branco. Observando a Figura 6.21(c), percebe-se que há uma pequena melhoria na detecção de bordas em relação à imagem original. Neste caso agora é possível observar que algumas bordas que tornam-se mais fortes entre o teclado e a mesa.

Os resultados do ajuste de gamma modificado é apresentado na Figura 6.20(d). Pode-se observar que o método proposto possui um ajuste de contraste com qualidade intermediária, ressaltando levemente as bordas do objetos e não gera falsas bordas no *background*. Pode ser visto na Figura 6.21(d) que o método proposto realça com eficiência as bordas, aparecendo bordas entre a mesa e o teclado ou entre a mesa e a cadeira.

6.8 Influência no realce de imagem na detecção de movimentos

Nesta seção são apresentados os resultados obtidos na detecção de movimentos aplicados nos vídeos após o realce de imagens, cujos resultados são mostrados na Figura 6.22. No vídeo testado, é realizado um movimento quadrangular, composto pelos movimentos cima para baixo, esquerda para direita, baixo para cima e direita para esquerda, respectivamente, nesta ordem.

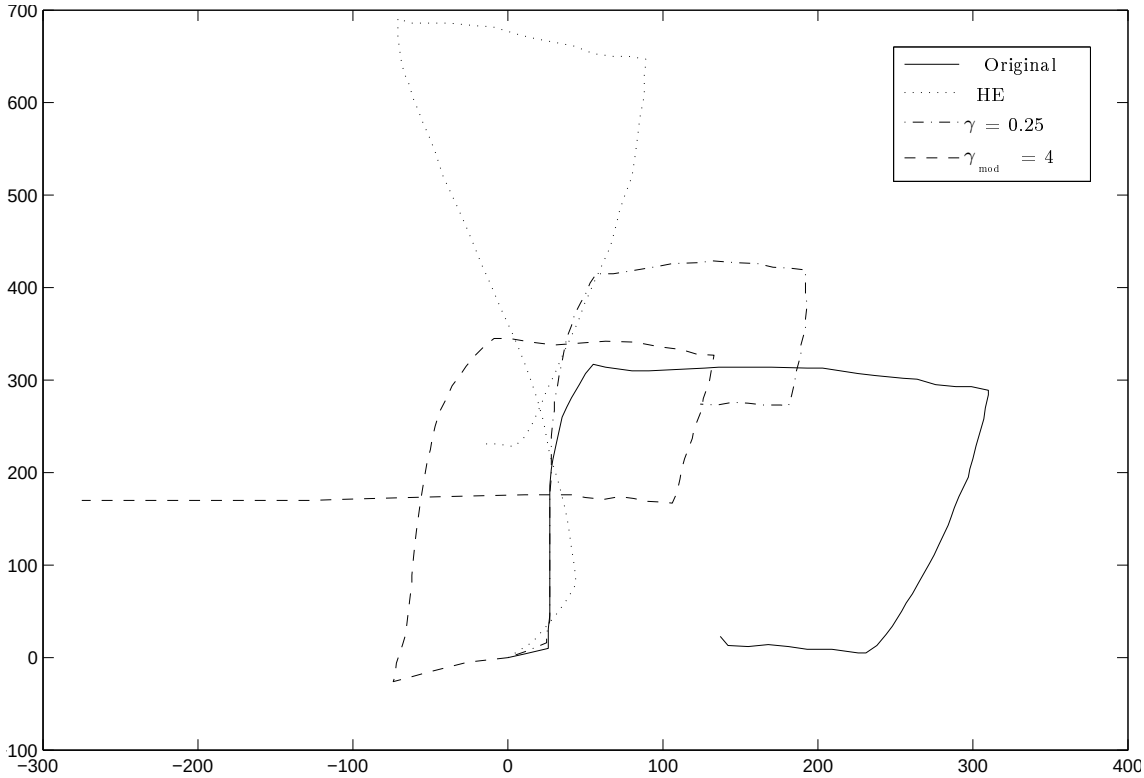


Figura 6.22: estimação de FO a partir do vídeo realçados pelos algoritmos estudados.

De acordo com esta Figura, percebe-se que a detecção de movimentos sem a aplicação do realce (linha sólida) é capaz de detectar apenas os três primeiros movimentos lineares. Ao realizar o último movimento, não houve uma efetiva

detecção de movimentos.

Após a aplicação da equalização de histograma, a detecção de movimentos não possui um bom desempenho. Neste caso, a trajetória estimada (linha pontilhada) assemelha-se a um triângulo, isto ocorre durante a realização dos movimentos de cima para baixo ou de baixo para cima, foram detectados movimentos diagonais.

Os resultados após a aplicação do ajuste de gamma são mostrados como uma linha com traço e ponto. Estes mostram que existe uma melhoria na detecção de movimentos durante a realização do primeiro movimento, ou seja, na região em que há a menor quantidade de luz na imagem. Nos outros dois movimentos há uma estimação de movimento pior do que na imagem original.

O último resultado (linha tracejada) representa o resultado da detecção de movimentos a partir do ajuste de gamma proposto. Observando os resultados percebe-se que o método realça as bordas durante a realização do primeiro e do último movimento, as regiões que possuem a menor quantidade de iluminação, ou seja, estão dentro das condições em que o realce é necessário. Em outras palavras, ao realizar o realce de imagens apenas quando necessário, o resultado ótimo é obtido, de forma conseguir uma trajetória estimada com aparência próxima à um quadrado perfeito.

Conclusões

Esta tese apresenta um novo algoritmo para interação com dispositivos portáteis baseado Fluxo Óptico (FO). Apesar dos algoritmos de FO tradicionais serem conhecidos por possuir um elevado custo computacional, através do conjunto de otimizações, é possível sua utilização em dispositivos com capacidade de processamento limitada e pouca memória. Estas otimizações estão relacionadas ao cálculo do fluxo óptico em apenas uma interação e leva em consideração o tipo de problema abordado nesta tese. Para realização da análise da complexidade dos algoritmos estudados, foram utilizadas as métricas de erro de estimação de trajetória, esforço computacional e usabilidade. O algoritmo proposto é comparado com diversos algoritmos de FO e em seguida comparado com diversos algoritmos de detecção de movimentos em dispositivos portáteis. Na primeira análise comparativa realizada nesta tese, o método proposto é confrontado com os algoritmos de FO de Horn e Schunck (1981) e de Lucas e Kanade (1981). Através da análise do Erro Médio Quadrático (MSE) de estimação de trajetória e da análise de esforço computacional, percebe-se que o método de correlação é eficiente na detecção de movimentos, mas possui o maior custo computacional. O método de Lucas e Kanade (1981) possui o menor esforço computacional, mas também apresenta o maior MSE de estimação de trajetória, ou seja, o pior resultado de detecção de movimento. Dentre os métodos de FO estudados, o algoritmo proposto associa uma eficaz detecção de movimentos e um baixo esforço computacional. Com base no algoritmo de FO proposto nesta tese, é desenvolvido um método para classificação de movimentos a partir da velocidade média de blocos que possibilita o reconhecimento de movimentos em quatro graus de liberdade (um de rotação e três de translação). Desta forma, o método proposto estende a detecção de movimentos do FO modificado, originalmente o método de FO

é capaz de detectar apenas os movimentos de translação nas direções (X e Y). A partir da análise dos resultados, mostra-se que a classificação de movimento proposta nesta tese é capaz de identificar movimentos em mais dois eixos de movimento, um de aproximação e distanciamento (translação na direção Z) e rotação (em torno do eixo Z). Após verificar a eficiência de detecção de movimento do método proposto, este é, então, comparado com métodos para interação com dispositivos portáteis. A análise de MSE de trajetória é realizada em diferentes condições de ruído e mostra a eficiência do método de Drab e Artner (2005) na detecção de movimentos de translação. Isto porque associam uma eficaz detecção de movimentos e um baixo esforço computacional, sendo capaz de detectar movimentos com um menor erro que o método proposto em condições sem ruído. O método de Rohs e Essl (2007) possui uma detecção com baixo MSE associado a um maior custo computacional.

Os métodos de detecção de movimentos para dispositivos portáteis têm seu esforço computacional avaliado através da medição de seu tempo de processamento médio. Observando-se os resultados, percebe-se que o método proposto possui um menor esforço computacional. Esta característica atende os requisitos mínimos de tempo de resposta do usuário, visto que o método proposto é capaz de processar um quadro a menos de 40 ms no dispositivo de menor poder computacional, posto que a interação deva ocorrer num tempo menor do que 100 ms. Assim, existe uma margem de 60 ms disponível para o dispositivo realizar atividades de controle da rede e aquisição de quadros de imagem. Por fim, os algoritmos de detecção de movimento de dispositivos são submetidos a um teste de usabilidade. Com base nos resultados, observa-se que o método proposto atende os requisitos de tempo de resposta e obtém uma boa aceitação dos usuários. Nos testes feitos, mostra-se que é possível a interação dos usuários com o sistema de mapas através do movimento que gerou satisfação e divertimento no uso deste tipo de interação. As otimizações, propostas no novo método, permitem tornar o algoritmo de FO de Horn e Schunck (1981) computacionalmente adequado para ser utilizado como interface de interação mesmo sob condições de baixa luminosidade devido ao ajuste de gamma proposto neste trabalho, comprovado pelos resultados obtidos.

As principais contribuições desta tese são:

- modificação do ajuste de gamma para realçar características em ambientes com baixa luminosidade e melhorar a detecção de movimento;

- desenvolvimento de uma interface de interação com quatro graus de liberdades, satisfazendo as características de tempo de resposta e satisfação do usuário;
- desenvolvimento de uma metodologia de implementação multiplataforma, tornando possível o método proposto ser aplicado em diferentes Sistemas Operacionais (SOs) de dispositivos móveis;
- verificação da aceitação de interfaces baseadas detecção de movimentos a partir da câmera embarcada do dispositivo;
- implementação e avaliação comparativa do desempenho de algoritmos de detecção de movimentos para dispositivos portáteis.

Como trabalhos futuros sugerem-se:

i. implementação de um algoritmo de filtragem passa-baixa para realçar as imagens para ser aplicado antes de realizar a estimação do FO;

ii. melhorar a detecção de movimentos de aproximação e distanciamento através do aumento do número de regiões avaliadas pelo método de classificação de movimentos proposto;

iii. combinação de algoritmos de detecção de movimentos para concepção de uma interface robusta;

iv. comparação dos métodos propostos com interfaces de interação baseada em objetos de cor pré-definida, rastreamento da mão e reconhecimento de feições;

v. implementação de método de estimação de velocidade utilizando uma abordagem multi-resolução, auxiliando a detecção mais efetiva de movimentos em regiões com pouca informação;

vi. avaliar a detecção de movimentos em canais isolados (R, G ou B) ou outro espaço de cor que favoreça a detecção de movimentos;

vii. adaptar o sistema implementado para sistemas operacionais não utilizados nesta tese como o BADA e o iOS;

Apêndice A

Formulário do Teste de Usabilidade

Universidade Federal do Ceará (UFC)

Laboratório de Engenharia de Sistemas de Computação (LESC)

Teste de usabilidade dos métodos de IHC baseados em movimento

M.Sc. Rodrigo Carvalho Souza Costa

Este formulário tem como objetivo auxiliar a realização dos testes de usabilidade de métodos de interação por movimentos. O documento está dividido em seções para realizar o procedimento.

1. Conhecendo o usuário

Primeiramente é necessário conhecer o usuário de testes, para isto, por favor responda as seguintes questões.

I Qual é a sua principal ocupação?

II Cerca de quantas horas por semana, você acha que gasta utilizando o celular?

III Qual percentual de tempo aproximado que você utiliza para seu divertimento como e-mail, navegação e jogar?

☐ Menos que 10% ☐ 30% ☐ 50% ☐ Mais que 70%

IV Quais são os recursos disponíveis em seu aparelho celular?

☐ touch screen ☐ teclado querty ☐ cursores ☐ GPS
☐ acelerômetro ☐ bluetooth ☐ câmera

V Você já utilizou o movimento realizado pelo celular para realizar controlar alguma funcionalidade ou jogo?

☐ sim ☐ não

VI Se sim, responda o que você achou dessa experiência?

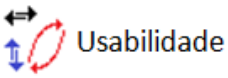
_____

_____

VII Você já utilizou a câmera do seu celular para realizar alguma função além de tirar fotos ou gravar vídeos?

☐ sim ☐ não

Neste momento, o usuário irá receber um dispositivo que possui as aplicações a serem testadas. Favor abrir a aplicação:



O usuário poderá utilizá-la por três minutos, podendo pressionar qualquer tecla do dispositivo ou qualquer botão da tela. Em seguida, o usuário responderá as questões da seção 2.

2. Primeira Aplicação

2.1. Conhecendo o conceito

I Na sua opinião, o que a aplicação faz?

_____

_____

II Qual a sua impressão sobre a forma de interação?

_____

III Você desejaria que esta funcionalidade estivesse presente em seu celular?

☐ sim ☐ não ☐ não importa

IV Você acredita que o movimento foi detectado apenas pela câmera?

☐ sim ☐ não

V O que você achou do tempo de resposta do sistema?

_____

_____

VI Utilizando apenas um adjetivo, descreva a forma de interação.

_____

_____

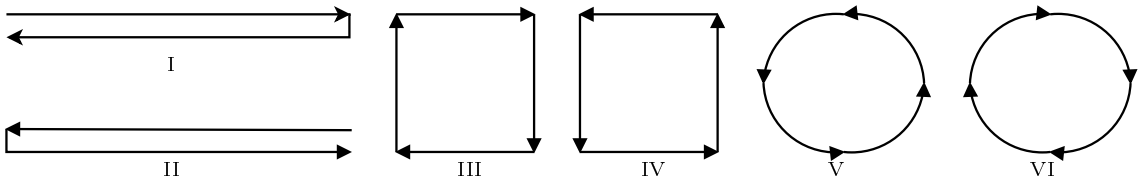
VII Você tem alguma observação ou pergunta até o momento?

_____

_____

2.2. Testes de interação

Neste momento, o usuário deverá realizar cada um dos movimentos mostrados a seguir e irá avaliar qual dos métodos de interação possui o melhor tempo de resposta e qualidade de detecção.



Para fazer isto, clique no botão menu do dispositivo e escolha um dos métodos, o dispositivo então irá lhe solicitar que faça os movimentos realizados. Ao final de cada movimento pressione o botão central do dispositivo para finalizar. Por favor, repita este procedimento para os quatro métodos implementados.

2.3. Resultados dos Testes de interação

I Qual possui o melhor tempo de resposta?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

II Qual possuiu o melhor rastreamento para o movimento A?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

III Qual possuiu o melhor rastreamento para o movimento B?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

IV Qual possuiu o melhor rastreamento para o movimento C?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

V Qual possuiu o melhor rastreamento para o movimento D?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

VI Qual possuiu o melhor rastreamento para o movimento E?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

VII Qual possuiu o melhor rastreamento para o movimento F?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

Segunda Aplicação: Testes de desempenho

Você agora utilizará os métodos de interação para posicionar o cursor no quadrado pintado de branco até concluir um circuito pré-definido. O usuário poderá ficar utilizando a aplicação por 10 minutos e deverá ao final do procedimento indicar o tempo necessário para realizar o circuito utilizando cada uma das formas de interação.

i. Qual possuiu você consegui realizar em menos tempo?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

ii. Qual possuiu você sentiu que realizou o circuito com o menor esforço?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

iii. Qual possuiu você sentiu que realizou o circuito que lhe causou um maior esforço?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

iv. Qual você achou que possuiu uma detecção de movimento mais estável?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

v. Qual você achou mais divertido?

☐ A ☐ B ☐ C ☐ D ☐ sem diferença

Terceira Aplicação: Utilização do método proposto em mapas

Por fim, favor testar o método proposto para interação com o mapa, utilizando a aplicação

Ao final da interação, favor indicar a opção sobre a sua impressão do método proposto em termos de:

i. precisão na detecção de movimentos de translação:

☐ insuficiente ☐ ruim ☐ bom ☐ muito bom ☐ otimo

ii. precisão em detectar a aproximação:

☐ insuficiente ☐ ruim ☐ bom ☐ muito bom ☐ otimo

iii. precisão em detectar o distanciamento:

☐ insuficiente ☐ ruim ☐ bom ☐ muito bom ☐ otimo

iv. tempo de resposta:

☐ insuficiente ☐ ruim ☐ bom ☐ muito bom ☐ otimo

v. divertimento:

☐ insuficiente ☐ ruim ☐ bom ☐ muito bom ☐ otimo

vi. Através de uma frase, descreva sua impressão sobre a interação com a aplicação?

_____

_____

vii. Você teria alguma sugestão para melhorar o sistema?

_____

_____

Referências Bibliográficas

AIRES, K. R. T.; SANTANA, A. M.; MEDEIROS, A. A. D. Optical flow using color information: Preliminary results. In: *SAC08*. Fortaleza: ACM, 2008. p. 1607 – 1611.

BALLAGAS, R.; ROHS, M.; SHERIDAN, J. G. Sweep and point & shoot: Phonecam-based interactions for large public displays. In: *In Proc of CHI 2005*. New York: ACM, 2005. p. 2–7.

BARBOSA, R. L. *et al.* Optical flow computation for land-based mobile mapping system images. *Revista Brasileira de Cartografia*, v. 57, n. 2, p. 72–78, aug 2005.

BARJATYA, A. *Block Matching Algorithms For Motion Estimation*. 2004. Article. Disponível em: <http://www.mathworks.com/matlabcentral/fileexchange/8761>.

BARNARD, M. *et al.* A vision based motion interface for mobile phones. In: *The 5th International Conference on Computer Vision Systems*. Bielefeld, Germany: Pergamon Press, 2007.

BARRON, J.; KLETTE, R. Quantitative color optical flow. In: *Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02)*. Washington, DC: IEEE Computer Society, 2002. v. 4, p. 251 – 255.

BEAUCHEMIN, S. S.; BARRON, J. L. The computation of optical flow. *ACM Computing Surveys (CSUR)*, v. 27, n. 3, p. 433 – 466, sep 1995.

BENYON, D. *Interação humano-computador*. 2ª ed. ed. São Paulo: Pearson do Brasil, 2011.

BETKE, M.; GIPS, J.; FLEMING, P. The camera mouse: Visual tracking of body features to provide computer access for people with severe disabilities. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, v. 10, n. 1, p. 1–10, Mar 2002.

BRADSKI, G. R. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, 1998.

BREW: Bynary runtime environment for wireless. 2009. Disponível em: <http://brew.qualcomm.com>.

BUCOLO, S.; BILLINGHURST, M.; SICKINGER, D. User experiences with mobile phone camera game interfaces. In: *MUM '05: Proceedings of the 4th international conference on Mobile and ubiquitous multimedia*. New York, NY, USA: ACM, 2005. p. 87–94. ISBN 0-473-10658-2.

CAETANO, A. C. M. *et al.* Câmera interativa para cybertv. In: INSTITUTO DE ARTES DA UNIVERSIDADE DE BRASÍLIA. *9º Encontro Internacional de Arte e Tecnologia*. Brasília, 2010. p. 27 – 35.

CHAN, K.-Y.; MANDUCHI, R.; COUGHLAN, J. Accessible spaces: navigating through a marked environment with a camera phone. In: *Assets '07: Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*. New York, NY, USA: ACM, 2007. p. 229–230. ISBN 978-1-59593-573-1.

CHEN, W.-C. *et al.* Efficient extraction of robust image features on mobile devices. In: *6th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*. Nara: IEEE Society Press, 2007. p. 287 – 288.

CHENG, F.-H.; CHEN, Y.-L. Real time multiple objects tracking and identification based on discrete wavelet transform. *Pattern Recognition*, v. 39, n. 6, p. 1126 – 1139, jun 2006.

CHO, Y. C.; JEON, J. W. Current software platforms on mobile phone. In: *International Conference on Control, Automation and Systems*. Seoul: [s.n.], 2007. p. 1862–1867.

COSTA, R. C. S. *et al. Patente PI0605825-6: Algoritmo de fluxo óptico modificado para rastreamento de deslocamentos em câmeras de baixa resolução*. dez 2006.

COSTA, R. C. S. *et al. Sistema classificador de movimento via algoritmo de fluxo óptico modificado utilizando câmeras de baixa resolução em dispositivos de comunicação móveis*. dez 2008.

CROWLEY, J. L.; COUTAZ, J. Vision for man machine interaction. In: *Proceedings of the IFIP TC2/WG2.7 Working Conference on Engineering for Human-Computer Interaction*. Londres: Chapman & Hall, Ltd, 1995. v. 45, p. 28 – 45.

CRUZ, V. M. *GINGA-NCL para Dispositivos Portáteis*. Dissertação (mestrado) — Pontífica Universidade Católica do Rio de Janeiro - PUC RJ, Rio de Janeiro, 2008.

DENMAN, S.; CHANDRAN, V.; SRIDHARAN, S. An adaptive optical flow technique for person tracking systems. *Pattern Recogn. Lett.*, Elsevier Science Inc., New York, NY, USA, v. 28, n. 10, p. 1232–1239, 2007. ISSN 0167-8655.

DRAB, S. A.; ARTNER, N. M. Motion detection as interaction techinique for games & aplications on mobile devices. In: *Proc. of PERVASIVE 2005*. Munich, Germany: Springer, 2005. (Lecture Notes in Computer Science, v. 3468).

DUNLOP, M. D.; LEVINE, J. Multidimensional pareto optimization of touchscreen keyboards for speed, familiarity and improved spell checking. In: , PUBLISHER = ACM Press, NOTE = , ABSTRACT = , KEYWORDS = ,. *CHI 2012*. Austin, USA, 2012.

EHRINGER, D. *The Dalvik Virtual Machine Architecture*. 2010. Disponível em: <http://blog.davidehringer.com/android/dalvik-virtual-machine/more-24>.

ELIAS, I. C. R.; VICTOR, J. L.; RAMPINELLI, M. A. *PESQUISA E DESENVOLVIMENTO PARA DISPOSITIVOS MÓVEIS*. Monografia (TCC) — CENTRO UNIVERSITÁRIO VILA VELHA, Vilha Velha, 2009.

FERNÁNDEZ-CABALLERO, A. *et al.* Optical flow or image subtraction in human detection from infrared camera on mobile robot. *Robotics and Autonomous Systems*, v. 58, n. 12, p. 1273–1281, 2010.

FLEET, D. J.; JEPSON, A. D. Computation of component image velocity from local phase information. *International Journal of Computer Vision*, v. 5, p. 77–104, 1990.

GALLO, O.; ARTEAGA, S. M.; DAVIS, J. E. A camera-based pointing interface for mobile devices. In: *15th IEEE International Conference on Image Processing, 2008.* San Diego: IEEE Society Press, 2008. p. 1420–1423.

GONZALEZ, R. C.; WOODS, R. E. *Processamento Digital de Imagens.* 3ª. ed. São Paulo: Pearson do Brasil, 2010.

GORODNICHY, D. O.; MALIK, S.; ROTH, G. Nouse "use your nose as a mouse" – a new technology for hands-free games and interfaces. *Image and Vision Computing*, v. 22, n. 12, p. 931–942, oct 2004. Disponível em: <http://www.cv.iit.nrc.ca/research/Nouse>.

BLAKELY SKOLOFF TAYLOR & ZAFMAN. A. Grunnet-Jepsen, K. Salsman e J. Sweetser. *Handleld Device For Handheld Vision Based Absolute Pointing System.* 2006. US 2006/0152487, 21 jul. 2005, 13. Jul 2006.

HACHET, M.; POUDEROUX, J.; GUITTON, P. A camera-based interface for interaction with mobile handheld computers. In: *I3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games.* New York, NY, USA: ACM, 2005. p. 65–72. ISBN 1-59593-013-2.

HAN, Z.; YE, Q.; JIAO, J. Combined feature evaluation for adaptive visual object tracking. *Computer Vision and Image Understanding*, v. 115, n. 1, p. 69 – 80, 2011.

HANNUKSELA, J. *et al.* Camera-based motion recognition for mobile interaction. *ISRN Signal Processing*, p. 1 – 12, 2011.

HANNUKSELA, J.; SANGI, P.; HEIKKILÄ, J. A vision-based approach for controlling user interfaces of mobile devices. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Workshop on Vision for Human-Computer Interaction (V4HCI).* San Diego, CA, USA: IEEE Society Press, 2005. 6 p.

HANNUKSELA, J.; SANGI, P.; HEIKKILÄ, J. Motion-based handwriting recognition for mobile interaction. In: *The 18th International Conference on Pattern Recognition (ICPR'06).* Washington: IEEE Computer Society, 2006.

HANNUKSELA, J.; SANGI, P.; HEIKKILA, J. Vision-based motion estimation for interaction with mobile devices. *Computer Vision and Image Understanding*, v. 108, p. 188–195, 2007.

HANSEN, T. R. Mixed interaction spaces - a new interaction technique for mobile devices. In: *UbiComp 2005*. New York, NY: ACM: Association for Computing Machinery, 2005. p. 1933 – 1936.

HARRISON, B. L. Squeeze me, hold me, tilt me! an exploration of manipulative user interfaces. In: *In Proc of CHI 1998*. New York: ACM Press/Addison-Wesley Publishin, 1998. p. 17–24.

Evan Hildreth. *Optical Flow Based Tilt Sensor*. 2006. US 0177103, 06 jan. 2006, 10 aug. 2006.

HINCKLEY, K. *et al.* Sensing techiniques for mobile interaction. In: *Proc of UIST 2000*. New York: ACM, 2000. p. 91–100.

HORN, B. K.; SCHUNCK, B. G. Determining optical flow. *Artificial Intelligence*, v. 17, p. 185–203, 1981.

HUSSY, J. H. Optical flow in image sequence coding. *Physica Scripta*, T38, p. 113–116, 1991.

IIDA, R. F. *Desenvolvimento Symbian para Plataforma Série 60*. Dissertação (mestrado) — Universidade de Brasília - UNB, Brasília, set 2006.

JESUS, A. D.; NASCIMENTO, R. G. *Sistemas Operacionais em Telefones Celulares*. Cuiabá, mai 2006.

KALAGASIDIS, A. S. *et al.* The international building physics toolbox in simulink. *Energy and Buildings*, v. 39, p. 665–674, 2007.

KHAN, S. *et al. Analysis of Dalvik Virtual Machine and Class Path Library*. Pakistan, Nov 2009.

KORATO, H.; TAN, K. T.; CHAI, D. (Ed.). *Barcodes for Mobile Devices*. Cambridge: Cambridge University Press, 2010.

LAUREANO, G. T.; PAIVA, M. S. V. de. Detecção de obstáculos utilizando fluxo óptico em seqüências de imagens. In: *II Workshop de Visão Computacional (WVC 2006)*. São Carlos - SP: USP, 2006. p. 295–298.

LECHETA, R. R. *Google Android*. [S.l.]: Novatech, 2010.

LEE, D. *et al.* Vision-based object detection and tracking for autonomous navigation of underwater robots. *Ocean Engineering*, v. 48, n. 1, p. 59 – 68, jul 2012.

LI, S. Z. *Markov Random Field, Modeling in Computer Vision*. 2nd. ed. London: Springler, 2000.

LUCAS, B. D.; KANADE, T. An iterative image registration technique with an application to stereo vision. In: *Proceedings of Imaging Understanding Workshop*. McLean: Science Applications, 1981. p. 121–130.

MACHADO, F. M.; MAIA, L. P. *Arquitetura de Sistemas Operacionais*. 4. ed. Rio de Janeiro: Grupo Gen: LTC, 2007.

MANDELLOS, N. A.; KERAMITSOGLOU, I.; KIRANOUDIS, C. T. A backgroundsubtraction algorithm for detecting and tracking vehicles. *Expert Systems with Applications*, v. 38, n. 3, p. 1619–1631, mar 2011.

MANDUCHI, R.; COUGHLAN, J.; IVANCHENKO, V. Search strategies of visually impaired persons using a camera phone wayfinding system. *Lecture Notes In Computer Science*, v. 5105, p. 1135 – 1140, 2008.

MASUI, T.; SIIO, I. Real–world graphical user interfaces. *Lecture Notes In Computer Science*, v. 1927, p. 72–84, 2000.

MATHWORKS. *Image Acquisition Toolbox - User Guide*. 3 Apple Hill Drive Natick, MA, sep 2006.

MATHWORKS. *Video and Image Processing Blockset - Userguide*. 3 Apple Hill Drive Natick, MA, sep 2006.

MATHWORKS. *Matlab/Simulink*. 2009. Disponível em: <http://www.mathworks.com>. Acesso em: 04 de fevereiro de 2009.

MIGLIORE, D. A.; MATTEUCCI, M.; NACCARI, M. A revaluation of frame difference in fast and robust motion detection. In: *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*. New York, USA: ACM, 2006. p. 215 – 218.

KONINKLIJKE PHILIPS ELECTRONICS N.V. Benoît Mory. *Camera Motion Parameters Estimation Method*. 2000. US 6349114, 24 dez. 1999, 20 jul. 2000.

MURATET, L. *et al.* A contribution to vision-based autonomous helicopter flight. *Robotics and Autonomous Systems*, v. 50, p. 195 – 209, 2005.

NAGEL, H. H. Constraints for the estimation of displacement vector fields from image sequences. In: *International Joint Conference on Artificial Intelligence*. [S.l.: s.n.], 1983.

NAGEL, H. H. On the estimation of optical ?ow: relations between different approaches and some new results. *Artificial Intelligence*, v. 33, p. 299–324, 1987.

NIELSEN, J. *Projetando Websites*. Rio de Janeiro: Elsevier, 2000.

OTSU, N. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, v. 9, n. 1, p. 62 – 66, 1979.

PAES, F. de A. *Sites com Interfaces Gráficas Acessíveis a Deficientes Visuais: um Estudo de Caso de Usabilidade*. Monografia (Especialização) — Unidade Lapa Scipião, São Paulo, 2010.

Mathematical models in computer vision: The handbook. In: PARAGIOS, N.; CHEN, Y.; FAUGERAS, O. (Ed.). Munich, Germany: Springler, 2005. cap. 15 - Optical Flow Estimation, p. 1–24.

PARAGIOS, N.; DERICHE, R. Detection of moving objects: A level set approach. *International Symposium on Intelligent Robotic Systems*, p. 265–274, 1997.

PARHI, P.; KARLSON, A. K.; BEDERSON, B. B. Target size study for one-handed thumb use on small touchscreen devices. In: , PUBLISHER = ACM Press, NOTE = , ABSTRACT = , KEYWORDS = ,. *Mobile HCI 06*. Helsinki, Finland, 2006.

POULARIKAS, A. D. (Ed.). *Probability and Stochastic Processes - The Handbook of Formulas and Tables for Signal Processing*. Boca Raton: CRC PRESS, 1999.

PYROFFERS. *Nokia 6230i Bluetooth Mouse Mod*. 2007. Disponível em: <http://www.pyrofersprojects.com/blog/?page_id=41>. Acesso em: 04 de fevereiro de 2009.

QIN, Y. *The Smart Phone as a Mouse*. Dissertação (Mestrado) — The University of Waikato, Nova Zelândia, nov 2006.

REIS, I. M. de S.; TAVARES, J. M. R. S. *Interface para processamento de imagem em C++ utilizando Visual Studio .NET 2005*. Porto, out 2007.

ROCHA, H. *Design e Avaliação de Interfaces Humano-Computador*. Campinas: UNICAMP, 2003.

ROHS, M. Real-world interaction with camera phones. *Lecture Notes in Computer Science*, v. 3598, p. 74–89, 2005.

ROHS, M. Marker-Based Embodied Interaction for Handheld Augmented Reality Games. *Journal of Virtual Reality and Broadcasting*, v. 4, n. 5, mar. 2007. `urn:nbn:de:0009-6-7939`, ISSN 1860-2037.

ROHS, M.; ESSL, G. Camus2 optical flow and collaboration in camera phone music performance. In: *Proceedings of the 2007 Conference on New Interfaces for Musical Expression*. New York: NIME, 2007. p. 160 – 163.

ROHS, M.; ZWEIFEL, P. A conceptual framework for camera phone-based interaction techniques. In: *Proc. of PERVASIVE 2005*. Munich, Germany: Springer, 2005. (Lecture Notes in Computer Science, v. 3468).

RUIZ, J.; LI, Y.; LANK. User-defined motion gestures for mobile interaction. In: *In Proc of CHI 2011*. Canada: ACM Press/Addison-Wesley Publishin, 2011.

SHAN, C.; TAN, T.; WEI, Y. Real-time hand tracking using a mean shift embedded particle filter. *Pattern Recognition*, v. 40, n. 7, p. 1958 – 1970, jul 2007.

SHENOLIKAR, P. C.; NAROTE, S. P. Different approaches for motion estimation. In: *INTERNATIONAL CONFERENCE ON CONTROL, AUTOMATION, COMMUNICATION AND ENERGY CONSERVATION*. [S.l.]: IEEE, 2009. p. 1–4.

SILVA, S. R. "eu não vivo sem celular": sociabilidade, consumo, corporalidade e novas práticas nas culturas urbanas. *Intexto (UFRGS)*, v. 2, p. 1, 2007.

SOARES, A. B.; FIGUEIRÓ, T.; SUSIN, A. Caracterização do desempenho de métodos de detecção de movimento aplicado a localização de pessoas através de visão computacional. *Congresso Brasileiro de Automática*, 2004.

SONKA, M.; HLAVAC, V.; BOYLE, R. *Image Processing, Analysis, and Machine Vision*. 2nd ed.. ed. [S.l.]: CENGAGE LEARNING INT, 2008.

STARCK, J.-L. *et al.* Gray and color image contrast enhancement by the curvelet transform. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, v. 12, n. 6, p. 706–717, Jun. 2003.

SURYANTO *et al.* Spatial color histogram based center voting method for subsequent object tracking and segmentation. *Image and Vision Computing*, v. 29, n. 12, p. 850 – 860, oct 2011.

TAYLOR, C. J. *et al.* Macroscopic traffic flow modelling and ramp metering control using matlab/simulink. *Environmental Modelling & Software*, v. 19, p. 975–988, 2004.

TEXEIRA, M. R.; POLIZELLI, D. L.; OZAKI, A. M. Detecção de obstáculos utilizando fluxo óptico em seqüências de imagens. In: *VIII Semead FEA - USP*. São Paulo - SP: USP, 2005.

TRACKING, Q. based optical flow estimation for robust object. Erkang chen and yi xu and xiaokang yang and wenjun zhang. *Digital Signal Processing*, 2012.

TRUYENQUE, M. A. Q. *Uma Aplicação de Visão Computacional que Utiliza Gestos da Mão para Interagir com o Computador*. Dissertação (Mestrado) — Pontífica Universidade Católica do Rio de Janeiro, Rio de Janeiro, Mar 2005.

WANG, J.; ZHAI, S.; CANNY, J. Camera phone based motion sensing: Interaction, techiniques, applications and performance study. In: *Proceedings of the 19th annual ACM symposium on User interface software and technology*. Montreux, Switzerland: ACM, 2006. p. 101–110.

WANG, L.; HU, W.; TAN, T. Recent developments in human motion analysis. *Pattern Recognition*, v. 36, p. 585–601, 2003.

WARD, C.-C. Y. J. D. Population balance modeling in simulink pcss. *Computers and Chemical Engineering*, 2007.

WINKLER, S. *et al.* Intuitive application-specific user interfaces for mobile devices. In: *Mobility '07: Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology*. New York, NY, USA: ACM, 2007. p. 576–582. ISBN 978-1-59593-819-0.

XU, L.; OJA, E.; KULTANEN, P. A new curve detection method: Randomized Hough Transform (RHT). *Pattern Recognition Letters*, v. 11, n. 5, p. 331–338, 1990.

XUE, G.; SUN, J.; SONG, L. Backgroundsubtraction based on phase feature and distance transform. *Pattern Recognition Letters*, v. 33, n. 12, p. 1601–1613, Sep 2012.

YANG, J.-B.; SHI, M.; YI, Q.-M. A new method for motion target detection by backgroundsubtraction and update. *Physics Procedia*, v. 33, p. 1768–1775, sep 2012.

YIM, S.; LEE, S.; CHOI, S. Evaluation of motion based interaction for mobile devices: A case study on image browsing. *Interacting with Computers*, v. 23, n. 2, p. 268–278, 2011.

YU, F. *et al.* The application of improved block-matching method and block search method for the image motion estimation. *Optics Communications*, v. 283, n. 23, p. 4619–4625, dec 2010.

ZHOU, H.; XIE, L.; FANG, X. Visual mouse SIFT detection and PCA recognition. In: *International Conference on Computational Intelligence and Security Workshops*. Harbin , China: IEEE Society Press, 2007. p. 263–266.