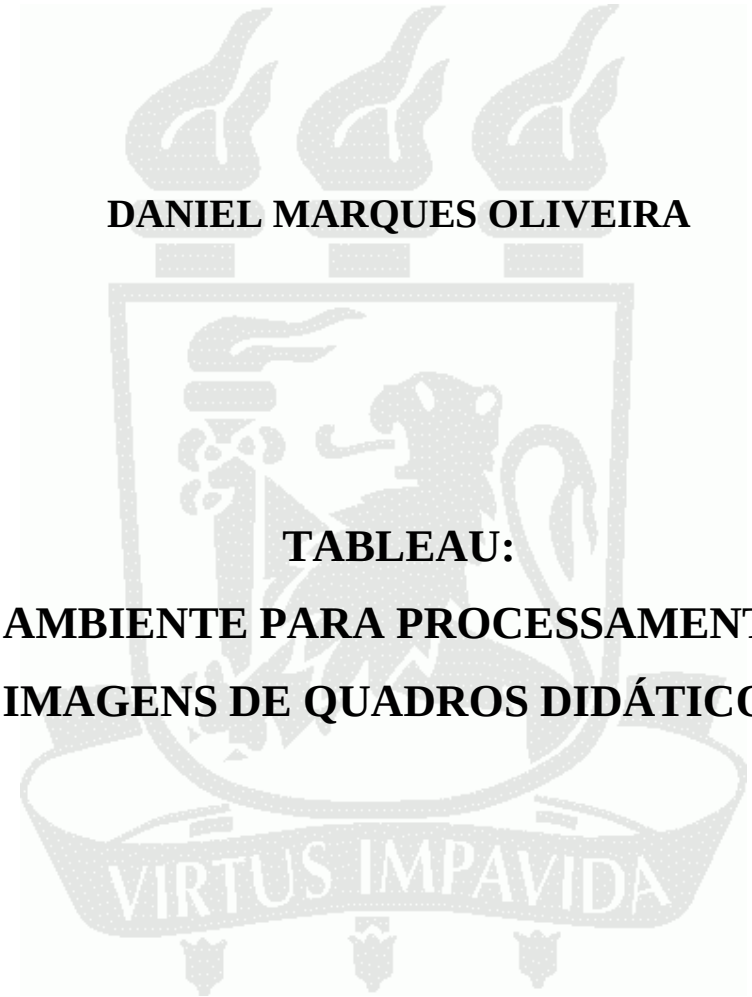
UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA



DANIEL MARQUES OLIVEIRA

TABLEAU:
UM AMBIENTE PARA PROCESSAMENTO DE IMAGENS DE QUADROS DIDÁTICOS

Recife, 6 março de 2009.

UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

TABLEAU:
UM AMBIENTE PARA PROCESSAMENTO DE IMAGENS DE QUADROS DIDÁTICOS

por

DANIEL MARQUES OLIVEIRA

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco como parte dos requisitos para a obtenção do grau de Mestre em Engenharia Elétrica.

ORIENTADOR: RAFAEL DUEIRE LINS, PHD

Recife, março de 2009.

.

O48T **OLIVEIRA, DANIEL MARQUES**

Tableau: um ambiente para processamento de imagens de quadros didáticos / Daniel Marques Oliveira. – Recife: O Autor, 2009.

xv, 180 f.; il., gráfs., tabs.

Dissertação (Mestrado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia Elétrica, 2009.

Inclui Referências bibliográficas e Apêndices.

1. ENGENHARIA ELÉTRICA. 2. DETECÇÃO DE BORDA. 3. NORMALIZAÇÃO DA ILUMINAÇÃO. 4. QUADRO NEGRO. 5. QUADRO DIDÁTICO. 6. SALA DE AULA. 7. SALA DE REUNIÃO. 8. PROCESSAMENTO DE DOCUMENTOS. 9. PROCESSAMENTO DE IMAGENS. I. TÍTULO.

UFPE

621.3 CDD (22. ED.) **BCTG/2009-155**

Universidade Federal de Pernambuco

Pós-Graduação em Engenharia Elétrica

PARECER DA COMISSÃO EXAMINADORA DE DEFESA DE DISSERTAÇÃO DO MESTRADO ACADÊMICO DE

DANIEL MARQUES OLIVEIRA

TÍTULO

"TABLEAU:
UM AMBIENTE PARA PROCESSAMENTO DE IMAGENS DE QUADROS DIDÁTICOS"

A comissão examinadora composta pelos professores: RAFAEL DUEIRE LINS, DES/UFPE, VALDEMAR CARDOSO DA ROCHA JÚNIOR, DES/UFPE e LUIZ EDUARDO SOARES DE OLIVEIRA, CCET/PUC-PR sob a presidência do primeiro, consideram o candidato **DANIEL MARQUES OLIVEIRA APROVADO**.

Recife, 06 de março de 2009.

EDUARDO FONTANA
Coordenador do PPGEE

RAFAEL DUEIRE LINS
Orientador e Membro Titular Interno

LUIZ EDUARDO SOARES DE OLIVEIRA
Membro Titular Externo

VALDEMAR CARDOSO DA ROCHA JÚNIOR
Membro Titular Interno

AGRADECIMENTOS

Desejo expressar os meus sinceros agradecimentos a todas as pessoas que contribuíram, direta ou indiretamente, no trabalho que será apresentado nessa dissertação. Dentre essas pessoas, sou grato em especial:

- Ao meu orientador Rafael Dueire Lins, pela sua paciência e por acreditar no meu trabalho, desde a minha iniciação científica até hoje e por me ensinar a ser um verdadeiro pesquisador.
- Ao professor Luiz Eduardo da PUC-PR, pela sua avaliação e comentários sobre esta dissertação, e por ser um dos organizadores do ICIAR 2007 em Curitiba.
- À minha namorada, pelo seu apoio para o meu desenvolvimento acadêmico e devido as minhas ausências nos finais de semana e eventos familiares. Desejo agradecer, também, a toda sua família.
- A toda minha família, em especial aos meus pais, Sérgio e Glória Oliveira, e minha irmã, Martha Oliveira, que me incentivaram, desde o vestibular, a seguir a minha carreira na área de Tecnologia da Informação. Quero agradecer a minha Avó, Adélia Barbosa, e todos tios que me incentivaram a entrar no mestrado.
- Aos meus amigos de pós-graduação do PPGEE da UFPE, em especial, a João Marcelo Silva, Bruno Ávila, Gabriel França Silva e Ednardo Mariano.
- Ao PPGEE da UFPE, por prover a estrutura necessária para realização do mestrado acadêmico. Em especial, ao professor Valdemar Rocha Jr, pelos seus comentários sobre a dissertação e por suas aulas. Desejo agradecer também ao professor Hélio Magalhães, por ter dado a oportunidade de aprender a cadeira de processamento de sinais.
- Ao Centro de Informática da UFPE, por ter me dado a oportunidade de me formar no curso de Bacharelado de Ciências da Computação.
- Aos meus colegas de trabalho no CESAR, em especial, a Tarciana Mello e Leila Souza por permitirem minha liberar a minha ida no horário de trabalho ao ICIAR 2007 em Curitiba. Desejo agradecer também a Fábio Maia, Camilo Almendra e

Cleber Moura por permitir um horário flexível que foi fundamental para o meu desenvolvimento acadêmico.

- Aos usuários do portal Flickr, Google Picasa e Gilles Rochefort da empresa RealEyes 3D que cederam suas imagens gratuitamente.
- Aos desenvolvedores de componentes e bibliotecas que auxiliaram no desenvolvimento deste estudo, como o ColorChooser (Jeremy Wood), ImageJ (Wayne Rasband) e ColorInspector3D (Kai Uwe Barthel).
- Aos professores do CTG, CCSA e Colégio de Aplicação da UFPE, que permitiram a coleta de imagens com quadros da sua aula.

Resumo da Dissertação apresentada à UFPE como parte dos requisitos necessários para a obtenção do grau de Mestre em Engenharia Elétrica.

TABLEAU:
UM AMBIENTE PARA PROCESSAMENTO DE IMAGENS DE QUADROS DIDÁTICOS

DANIEL MARQUES OLIVEIRA

Março/2009

Orientador: Rafael Dueire Lins, PhD.

Área de Concentração: Telecomunicações.

Palavras-chave: detecção de borda, normalização da iluminação, quadro negro, quadro didático, sala de aula, sala de reunião, processamento de documentos, processamento de imagens.

Número de Páginas: 195.

RESUMO: O uso de câmeras digitais, devido ao seu custo baixo, portabilidade e boa qualidade de imagem, tem se tornado cada dia mais usual. Uma das aplicações não originalmente prevista, que vem se tornando corriqueira, é a fotografia de quadros didáticos e de reuniões para uso posterior. É freqüente que fotos de quadros didáticos englobem não só a lousa, mas também outros objetos que estão na sua circunvizinhança. Outro problema é a iluminação irregular resultante de várias fontes de luzes no ambiente em que a foto foi tirada. Esta dissertação tem como objetivo apresentar novos algoritmos para o processamento de imagens de quadros, os quais foram integrados formando um ambiente chamado Tableau. Tableau é um ambiente amigável ao usuário que permite: remoção automática de bordas; correção de perspectiva; melhoria da imagem de quadros de qualquer cor, o que é uma solução pioneira, pois não existe ferramenta acadêmica ou comercial com tal funcionalidade.

Abstract of Dissertation presented to UFPE as a partial fulfillment of the requirements for the degree of Master in Electrical Engineering.

TABLEAU:
AN ENVIRONMENT FOR DIDACTIC BOARD IMAGE PROCESSING

DANIEL MARQUES OLIVEIRA

March/2009

Supervisor(s): Rafael Dueire Lins, PhD.

Area of Concentration: Telecommunications.

Keywords: boundary detection, border detection, illumination normalization, blackboard, didactic board, classroom, meeting room, document processing, image processing.

Number of pages: 195.

ABSTRACT: The use of digital cameras is becoming widespread due its low-cost, portability and good image quality. One of the applications not originally foresee which is becoming usual is taking picture of didactic and meeting boards for later use. Most of times, board pictures are not restricted to the board writing area, but include other objects around it. Another problem that makes more difficult to process such images is the irregular illumination, due to various light sources in the room when the picture is taken. This dissertation presents new algorithms that solve such problems which are set together into a single image processing tool called Tableau. Tableau is a user-friendly environment that allows: automatic border removal; perspective correction; enhancement of board picture of any background color. These features make of Tableau a pioneer solution both in academic and commercial terms.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

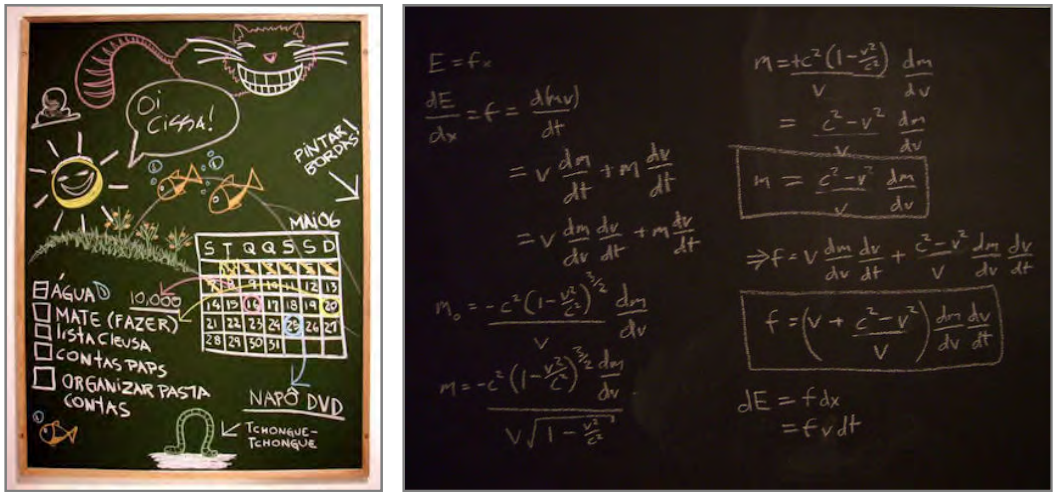
LISTA DE PSEUDOCÓDIGOS

1 Introdução

Quadros de giz (negros ou verdes) são um recurso didático milenar presentes em todo o mundo. Atualmente, esses objetos estão perdendo espaço para os quadros brancos, nos quais não existe o incômodo "pó-de-giz".

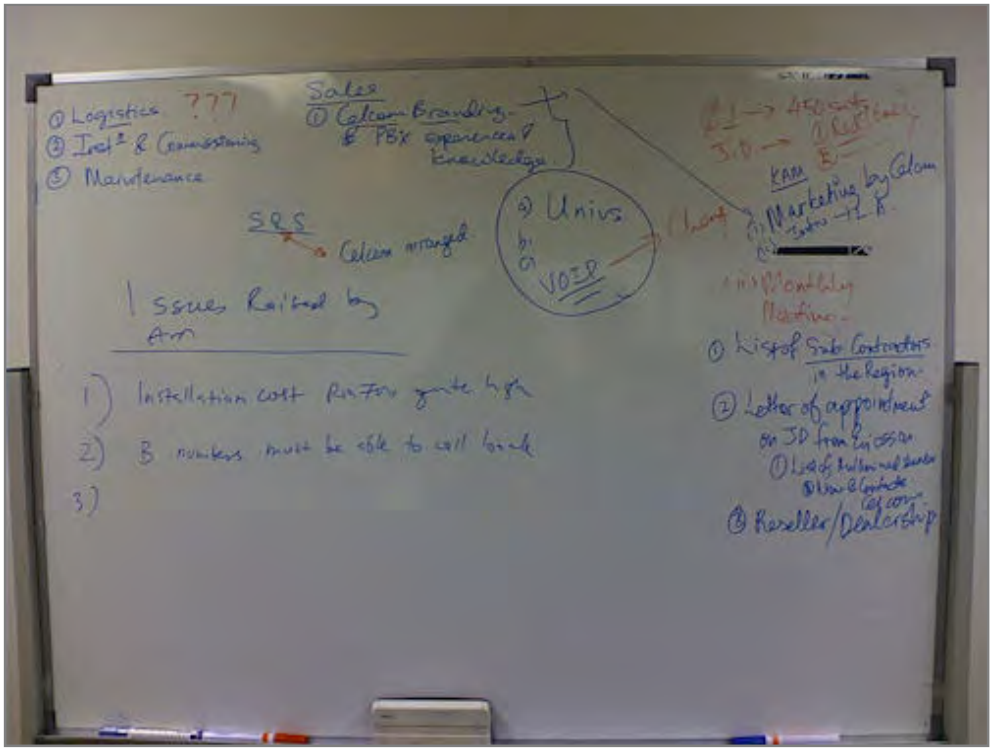
Apesar de terem sido projetadas para tirar fotos de família, o uso de câmeras digitais em salas de aula está se tornando cada vez mais difundido como meio de armazenar o conteúdo escrito no quadro durante a aula (HOUSE *et al*, 2005), tornando-se um objeto de auxilio didático. Isso só se tornou possível devido: ao preço acessível, a boa qualidade das imagens obtidas, à portabilidade, à praticidade e dentre outros fatores. Sem contar que há câmeras digitais integradas a telefones celulares, tendo se tornado um bem de consumo pervasivo, ou seja, onipresente na vida quotidiana de grande parte da população.

O uso de quadros didáticos não está restrito a salas de aulas, sendo também empregado em empresas para troca de informações em reuniões, discussões técnicas etc. Segundo um estudo feito por Cherubini e seus colegas (CHERUBINI *et al*, 2005) com desenvolvedores da Microsoft, os quadros brancos são o meio preferido dentre os programadores quando se deseja tirar dúvidas e em reuniões. Os profissionais fazem anotações das reuniões e tiram fotos dos quadros para ambas serem utilizadas após a reunião. Algumas imagens típicas de quadros didáticos são ilustradas nas Figuras 1.1 e 1.2.

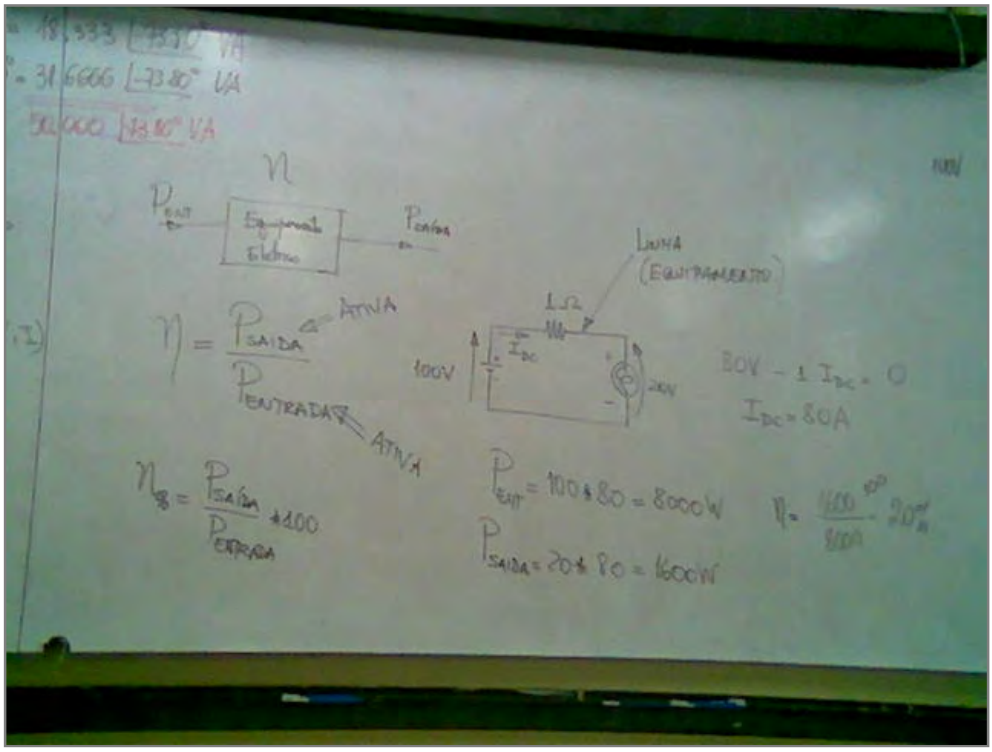


(a) Quadro verde (b) Quadro preto

Figura 1.1 – *Exemplos de imagens de quadros não-brancos*

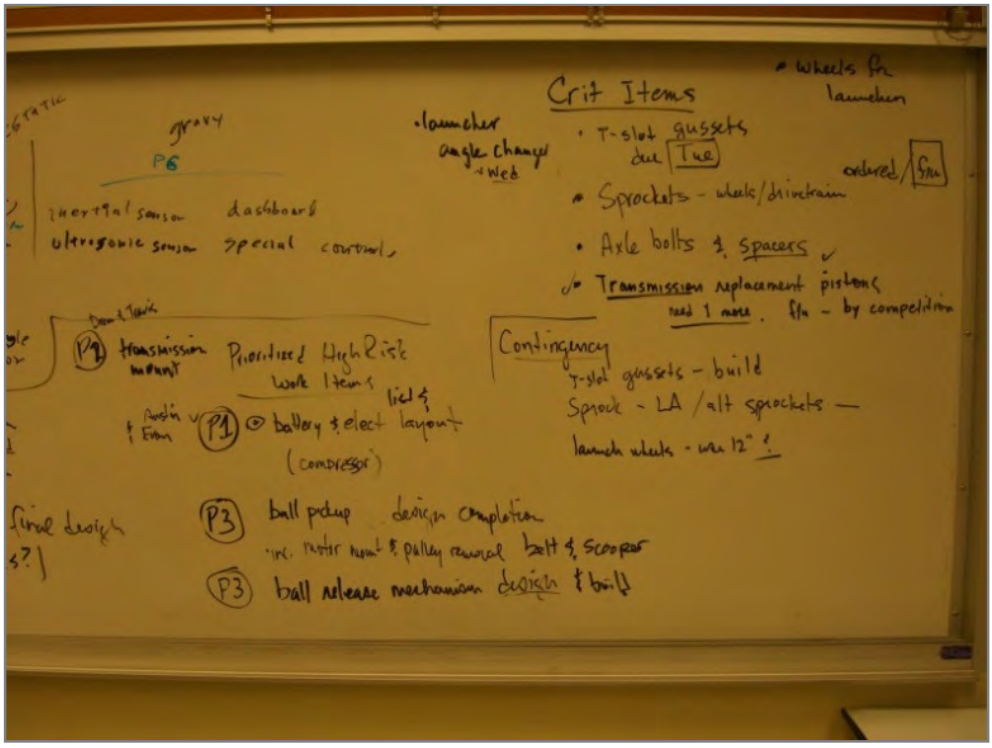


(a) Quadro branco com distorção de lente



(b) Foto com resolução VGA tirada com o Nokia 6230

Figura 1.2 – *Exemplos de imagens de quadros brancos*



(c) Quadro com interferência da iluminação na cor do fundo

Figura 1.2 (cont.)

Os exemplos de quadros apresentados nas Figuras 1.1 e 1.2 permitem observar que:

- A iluminação é irregular e não existe um padrão de iluminação comum.
- A fonte de luz resultante não é necessariamente branca, pois pode ser formada através: da reflexão dos objetos ao redor; filtros coloridos como o vidro fumê; por luz incandescente amarela como na Figura 1.2 (c), etc.
- As fotos apresentam efeito de distorção da lente da câmera. As lentes podem apresentar três tipos de distorção:
    - "olho-de-peixe" (*barrel*), a diminuição da ampliação a medida em que se distancia do eixo focal da imagem (geralmente localizada no centro da imagem), presentes nas Figuras 1.1 (a) e 1.2 (a);
    - "alfineteira" (*pincunshion*), que é o aumento da ampliação ao distanciar-se do eixo focal da imagem;
    - bigode (*mustache*), que é a combinação dos dois anteriores (ADOBE, 2008), caracteriza-se pela predominância do o efeito de "olho-de-peixe" na região próxima ao eixo focal e o predomínio do efeito "alfineteira" na região próxima a periferia da imagem, esta distorção é mais rara.

- Como as fotos de quadros são geralmente tiradas a mão livre, sem suporte mecânico, é comum o aparecimento de uma distorção de perspectiva. Idealmente, a superfície do quadro estaria em plano paralelo ao da lente da câmera. Esta distorção é mais visível na Figura 1.2 (b).
- Pode-se observar que nem todo quadro possui moldura ao redor da lousa, como na Figura 1.2 (b).
- Em muitos casos, a foto é parcial devido ao quadro ser muito largo, como na Figura 1.2 (b). É possível também que a foto apresente apenas o conteúdo do quadro sem a borda, como na Figura 1.1 (b).
- Os riscos impressos são arbitrários e não seguem um padrão bem definido, em contrapartida ao que ocorre num texto impresso, onde os caracteres estão dispostos de uma maneira mais estruturada.
- As imagens tiradas com câmeras de aparelhos celulares são um pouco "embaçadas" devido ao padrão de *Bayer*, que será descrito com mais detalhes no próximo capítulo. Um exemplo deste tipo de imagem está presente na Figura 1.2 (b), tirada com um celular Nokia 6230 com resolução VGA (640x480 *pixels*).

Diante do exposto, identifica-se que a digitalização de quadros didáticos através de câmeras digitais portáteis gera imagens complexas. Este tipo de imagem merece ser tratada com ferramentas específicas, visando facilitar o manuseio desse tipo de imagem por não-especialistas em processamento de imagens.

1.1 Objetivos

Esta dissertação apresenta o ambiente Tableau, uma plataforma de *software* desenvolvida para processar imagens de quadros didáticos obtidas com câmeras portáteis digitais. O Tableau possibilita aprimorar a imagem a partir de três operações:

- Detecção da borda do quadro, que automaticamente encontra a fronteira entre o conteúdo e os objetos externos.
- Correção de perspectiva, que efetua uma transformação geométrica gerando uma imagem como se a foto do quadro tivesse sido obtida frontalmente.
- Realce da escrita, que tornam os riscos mais evidentes, normalizam a iluminação e tornar o fundo do quadro chapado, sem contar que consegue realçar quadros com

qualquer cor de fundo. No caso do realce não há ferramenta comercial ou acadêmica que consigam realçar quadros com qualquer cor de fundo.

Para realizar tais operações foram desenvolvidos novos algoritmos com intuito de aprimorar os resultados dos algoritmos existentes na literatura. Na versão atual, o usuário transfere as fotos para seu computador (*desktop* ou *laptop*) onde Tableau está instalado. A versão atual de Tableau foi desenvolvida como um complemento (*plugin*) do ImageJ (RASBAND, 2008), um sistema de processamento de imagens de código aberto desenvolvido na linguagem Java por Wayne Rasband no Instituto Nacional de Saúde Mental (NIMH), na cidade de Bethesda, Maryland, Estados Unidos. Dessa maneira, o ambiente Tableau se beneficia da portabilidade da linguagem Java, disponível em quase todas as arquiteturas atuais.

ImageJ comporta a maioria das funções comuns aos outros sistemas de processamento de imagens. ImageJ oferece facilidades de desenvolvimento, tais como, carregar e salvar imagens; seleção de áreas de interesse; execução de algoritmos clássicos; cortar imagem; execução de *scripts* desenvolvidos por usuários; dentre muitas outras funções necessárias para o ambiente de processamento de imagens. Além de prover um ambiente produtivo para desenvolvimento de novos algoritmos, permitindo acesso direto a matriz de pixels e integração a outros completos.

O fluxograma de execução das operações do ambiente Tableau pode ser visto na Figura 1.3.

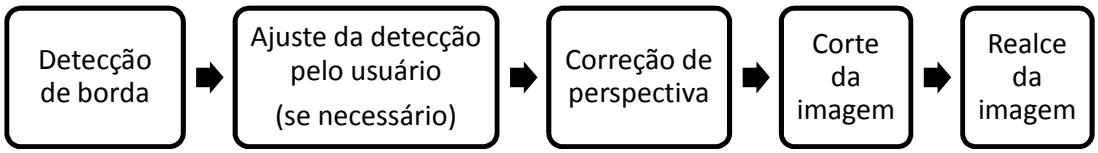


Figura 1.3 – ***Fluxograma da execução das operações do Tableau***

A imagem da Figura 1.4 ilustra a execução Tableau no ImageJ no Windows Vista.

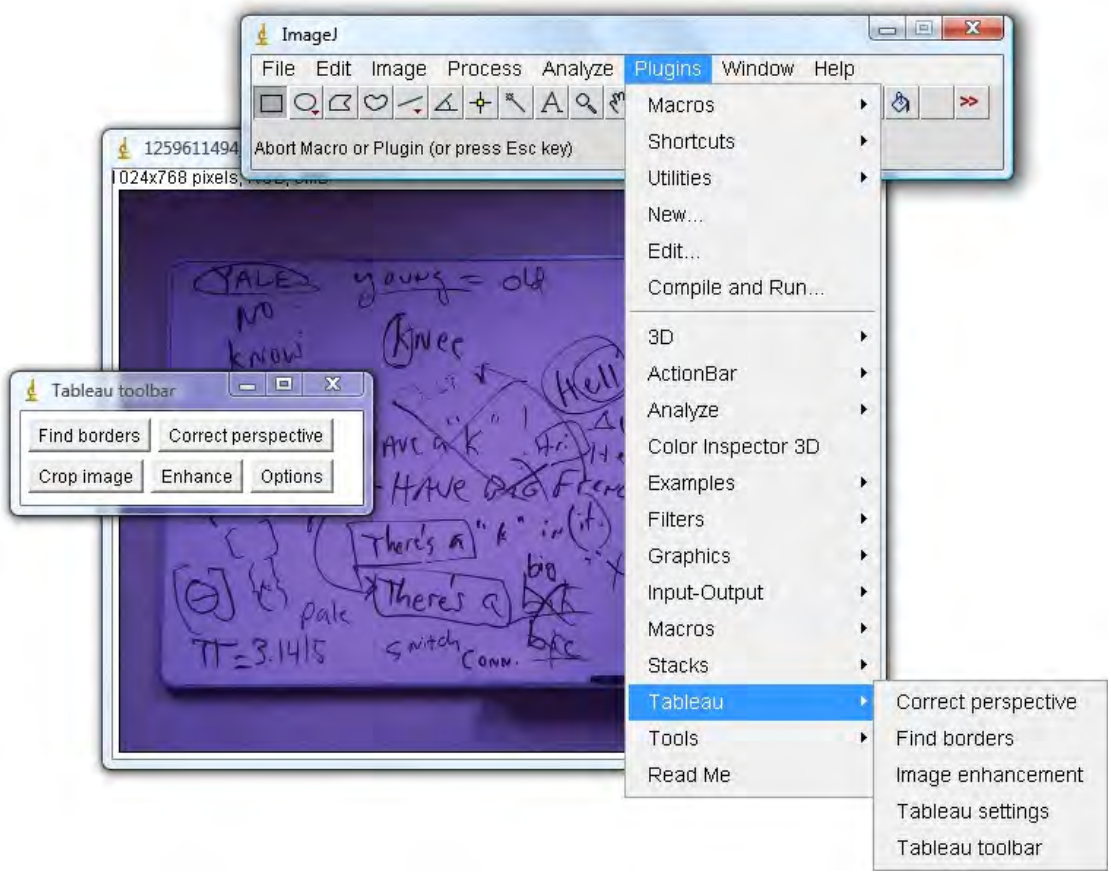


Figura 1.4 – *ImageJ com o Tableau*

Vale salientar que a qualidade da imagem fruto do processamento efetuado por Tableau dificilmente seria obtida com o uso de filtros padrão existentes em ferramentas de propósito geral, inclusive se utilizado por usuário especializado, em face da grande complexidade e especialização dos filtros aqui desenvolvidos.

Observa-se que o ambiente executa quatro operações: detecção de borda, correção de perspectiva, cortar imagem e realce da imagem. Essas operações podem ser tanto executadas através do menu “Plugins” → “Tableau”, como também através da barra de ferramentas do Tableau (“Tableau toolbar”). Ainda há a opção de anexar a barra de ferramentas do Tableau ao menu principal do ImageJ, como mostrado na Figura 1.5.

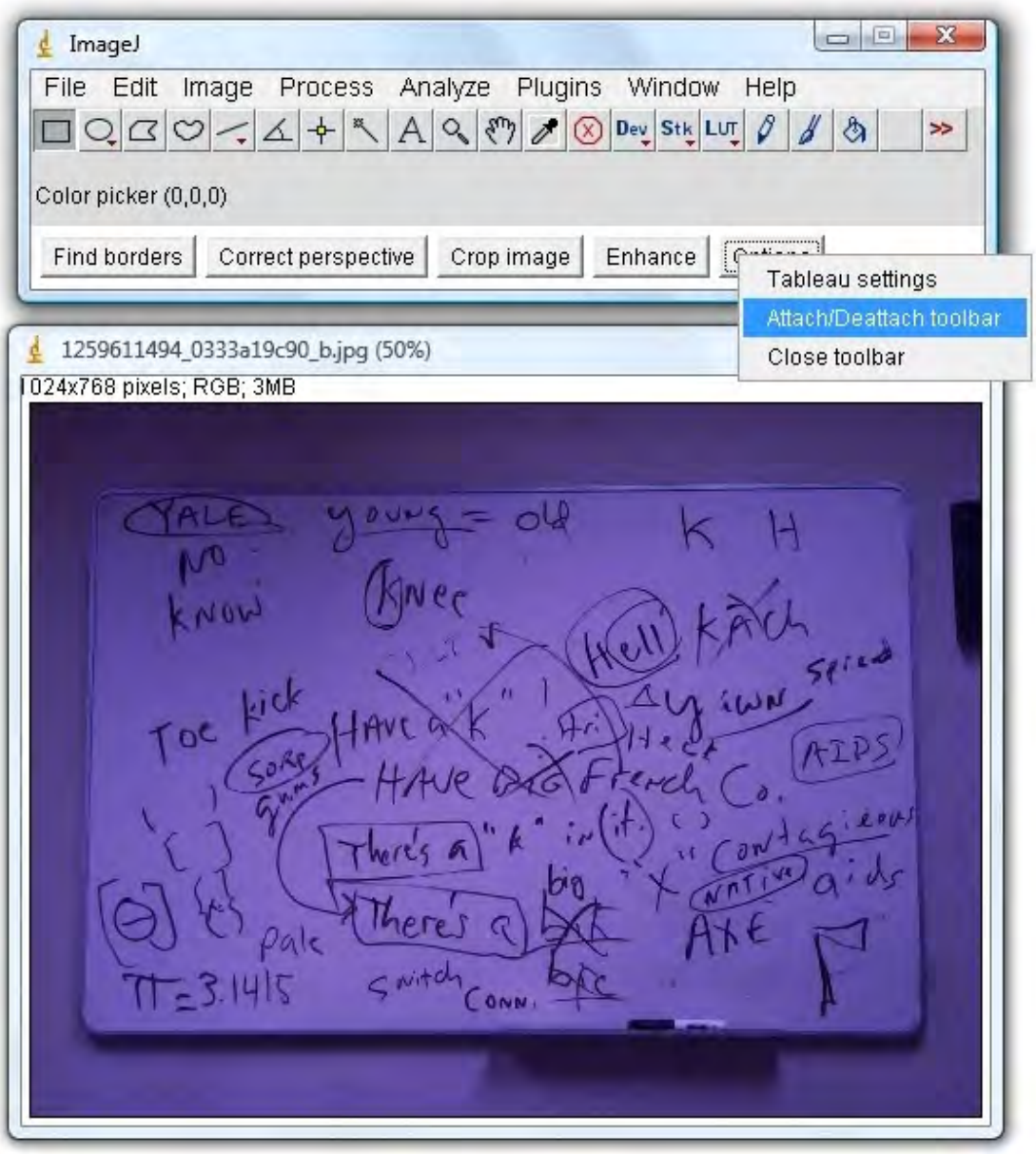


Figura 1.5 – *ImageJ com a barra de ferramentas do Tableau anexada ao menu principal*

A primeira operação a ser executada é a detecção de bordas. Essa opção vai identificar a fronteira do conteúdo e a área externa. O resultado da execução desse mecanismo na imagem do quadro da Figura 1.5 pode ser visto na Figura 1.6.

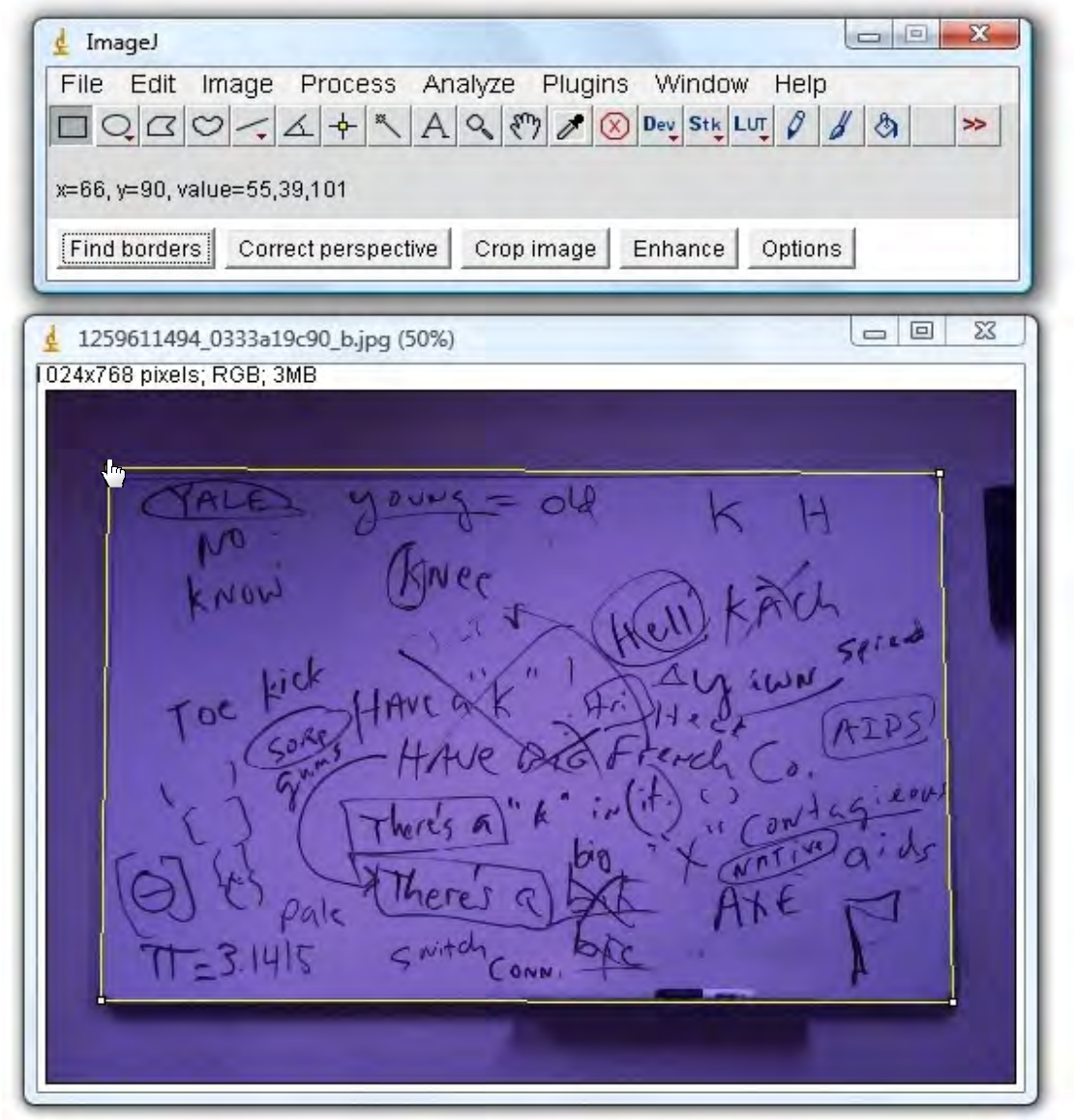


Figura 1.6 – *Borda detectada pelo Tableau*

Uma vez automaticamente detectada pelo ambiente Tableau o usuário pode ajustar conforme necessário. Após o ajuste é possível corrigir a perspectiva do quadro (*Correct perspective*) e cortar a imagem com o conteúdo do quadro. Para o corte utilizou-se a própria rotina do ImageJ com esse fim, com relação a correção de perspectiva foi implementado em Tableau o algoritmo de correção de perspectiva proposto por SILVA e LINS (SILVA, 2006; SILVA and LINS, 2005), detalhado no capítulo 4 desta dissertação. As Figuras 1.7, 1.8 e 1.9 ilustram esse processo.

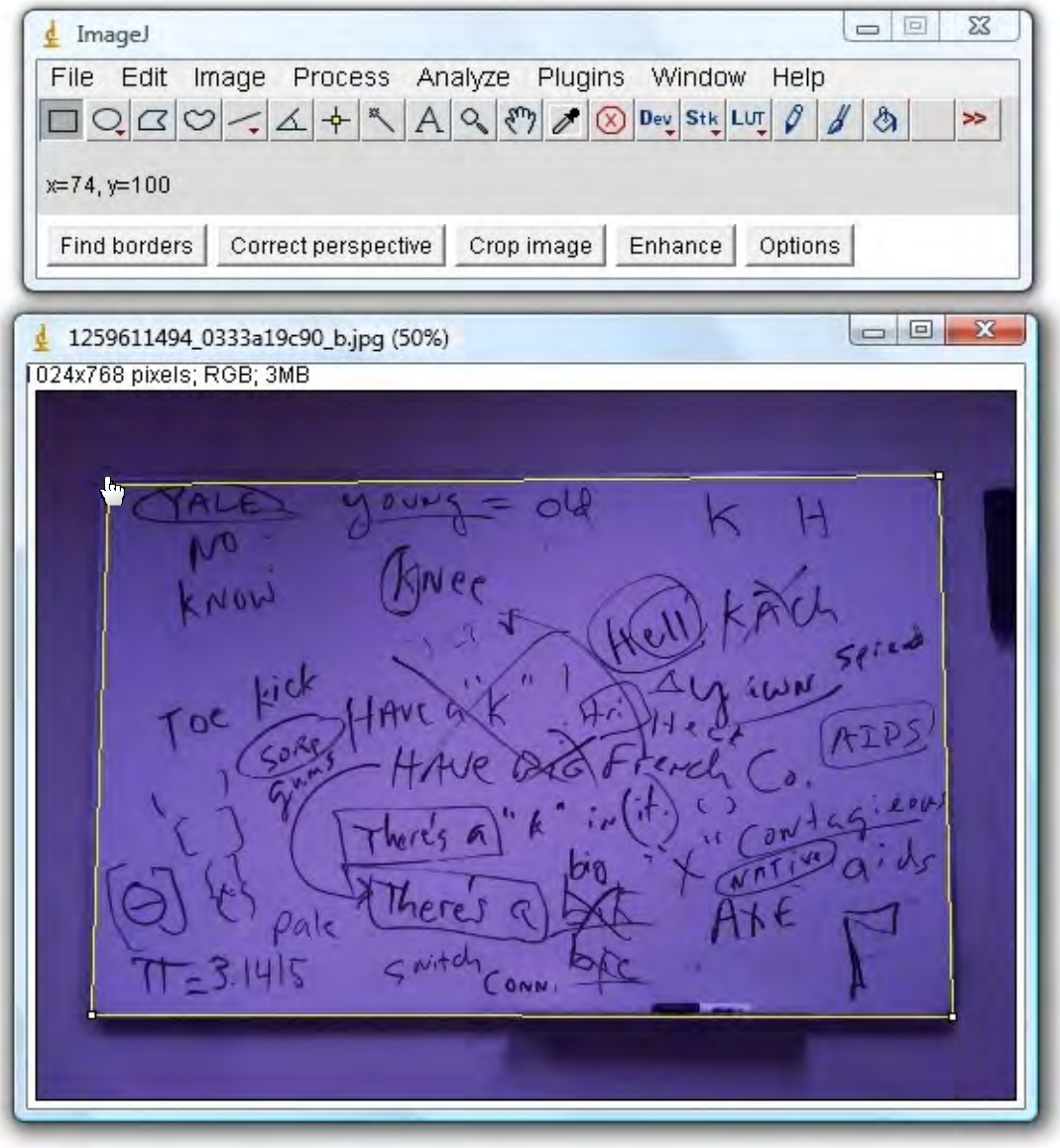


Figura 1.7 – *Borda delimitada após ajuste do usuário*

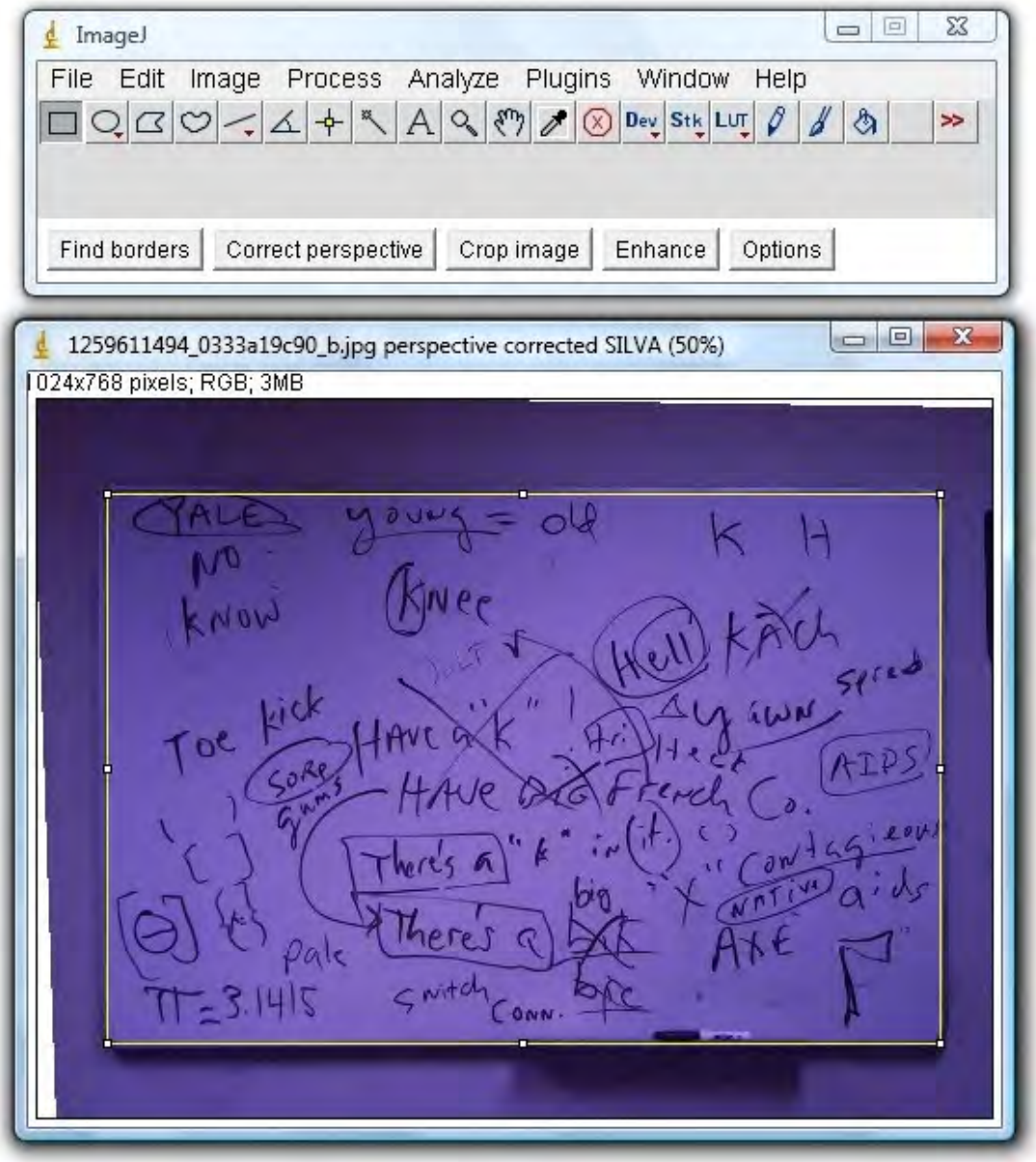


Figura 1.8 – *Perspectiva corrigida com a área selecionada para corte da imagem*

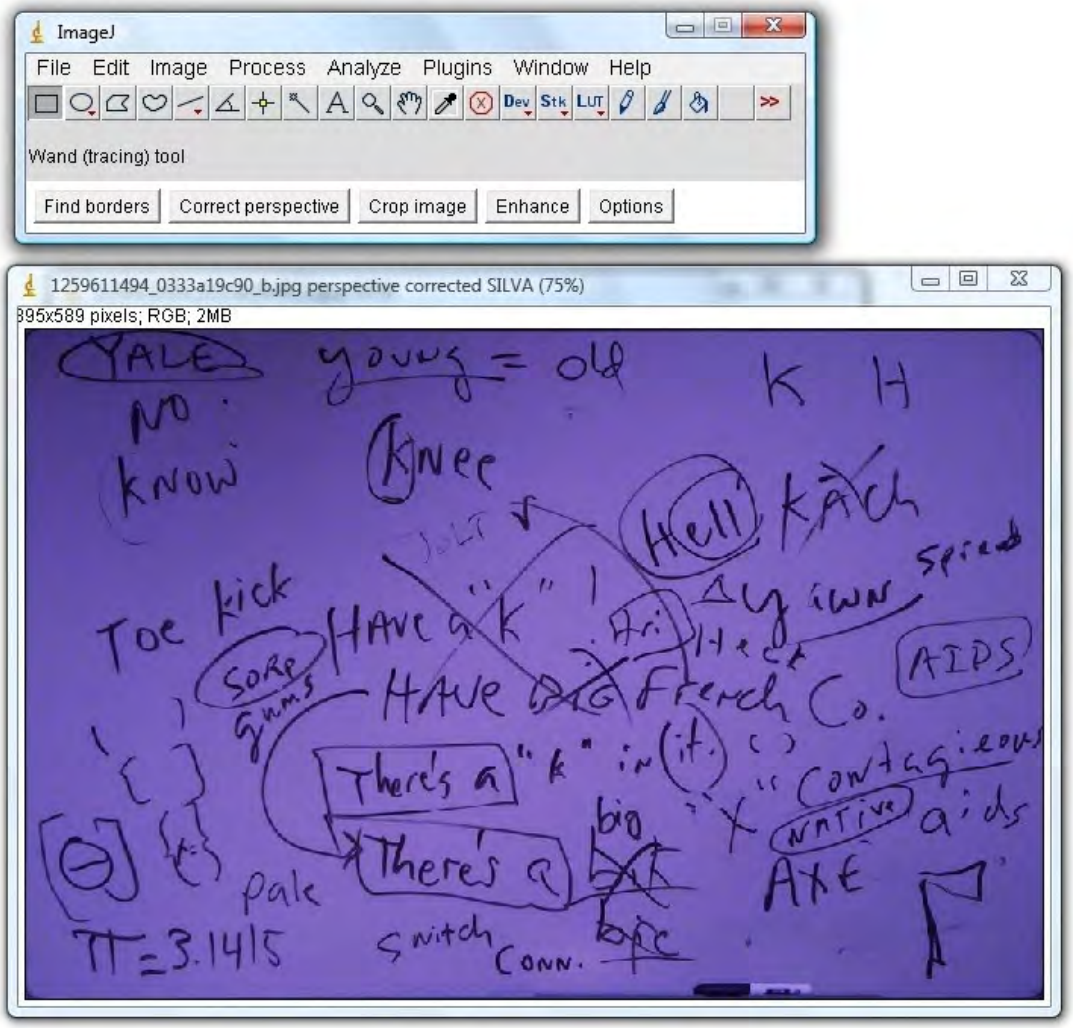


Figura 1.9 – *Imagem cortada*

A imagem uma vez cortada, pode ser melhorada utilizando o algoritmo de realce (*Enhance*). Tableau pergunta ao usuário qual a cor que deva ser aplicada como fundo de final, sugerindo a cor mais próxima da lista de fundos. Caso o usuário deseje, é possível utilizar o fundo inferido a partir da própria imagem ou o fundo sugerido. Nessa tela de sugestão também é possível alterar os parâmetros do algoritmo, bem como pedir para que o sistema não pergunte sobre a cor do fundo. Nesse caso é necessário selecionar uma das opções: utilizar o fundo inferido ou sempre utilizar uma cor da lista de fundos, conforme apresentado na Figura 1.10. A imagem realçada pode ser visto na Figura 1.11.

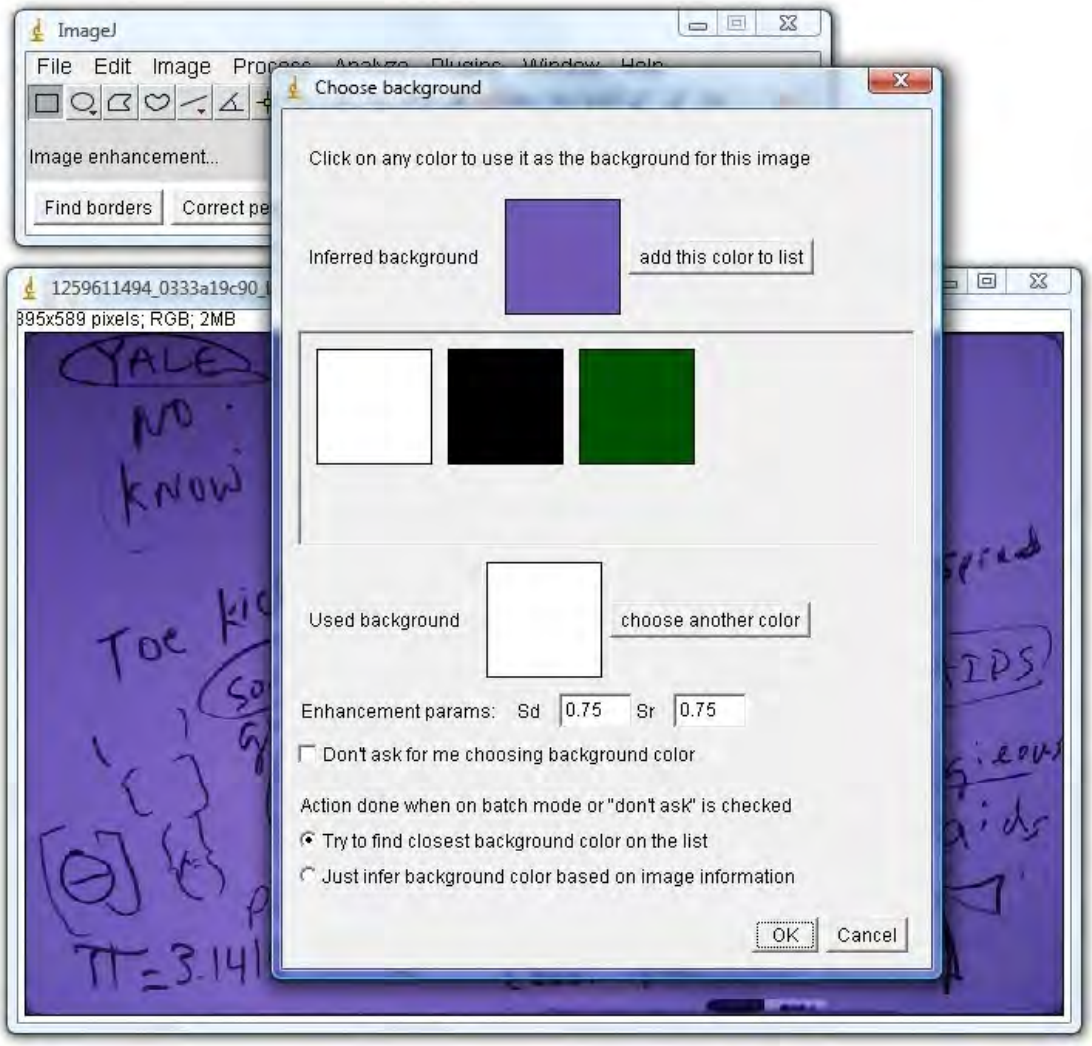


Figura 1.10 – *Tela mostrando ao usuário qual a cor de fundo a ser utilizada*

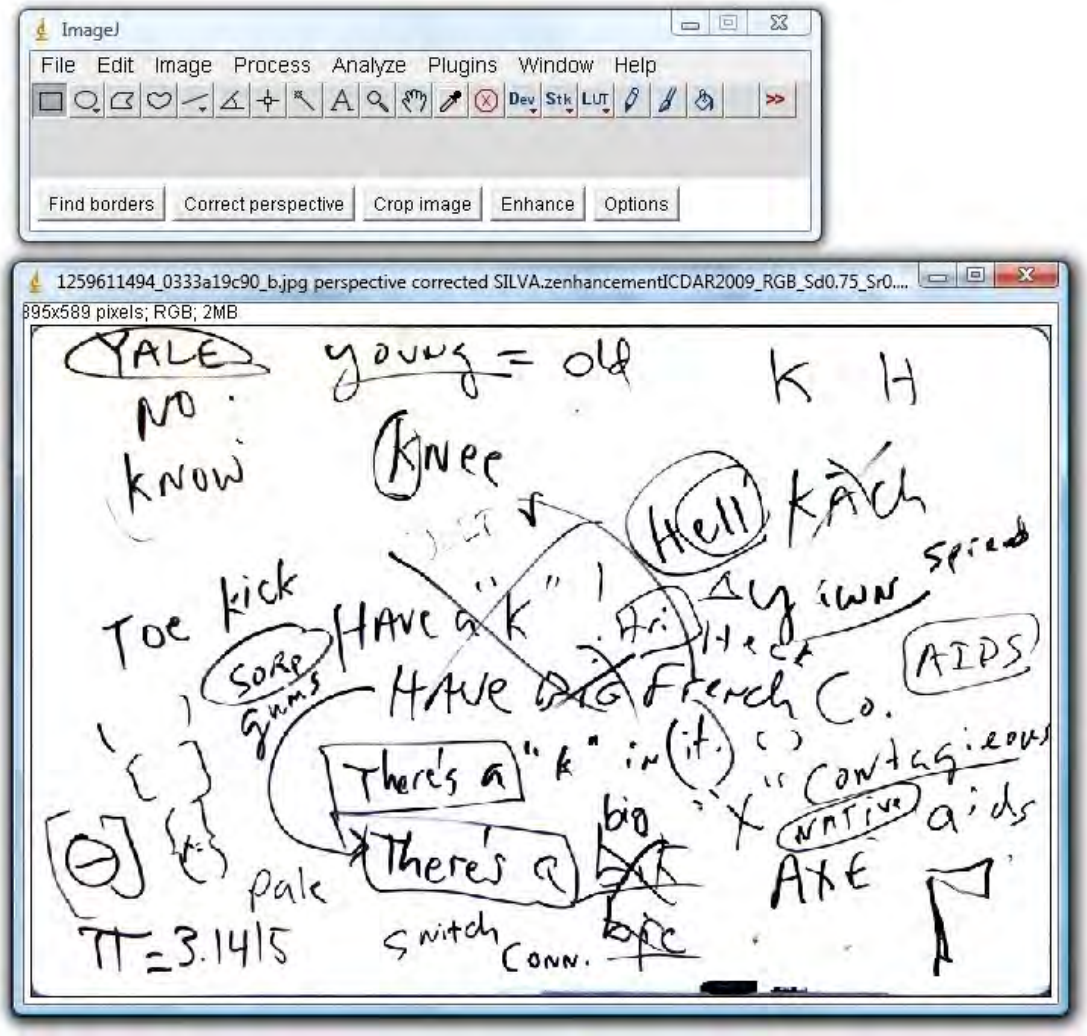


Figura 1.11 – *Imagem melhorada com o realce*

Outro exemplo de realce é dos quadros verde e preto (da Figura 1.1), que são mostrados a seguir nas Figuras 1.12 e 1.13.

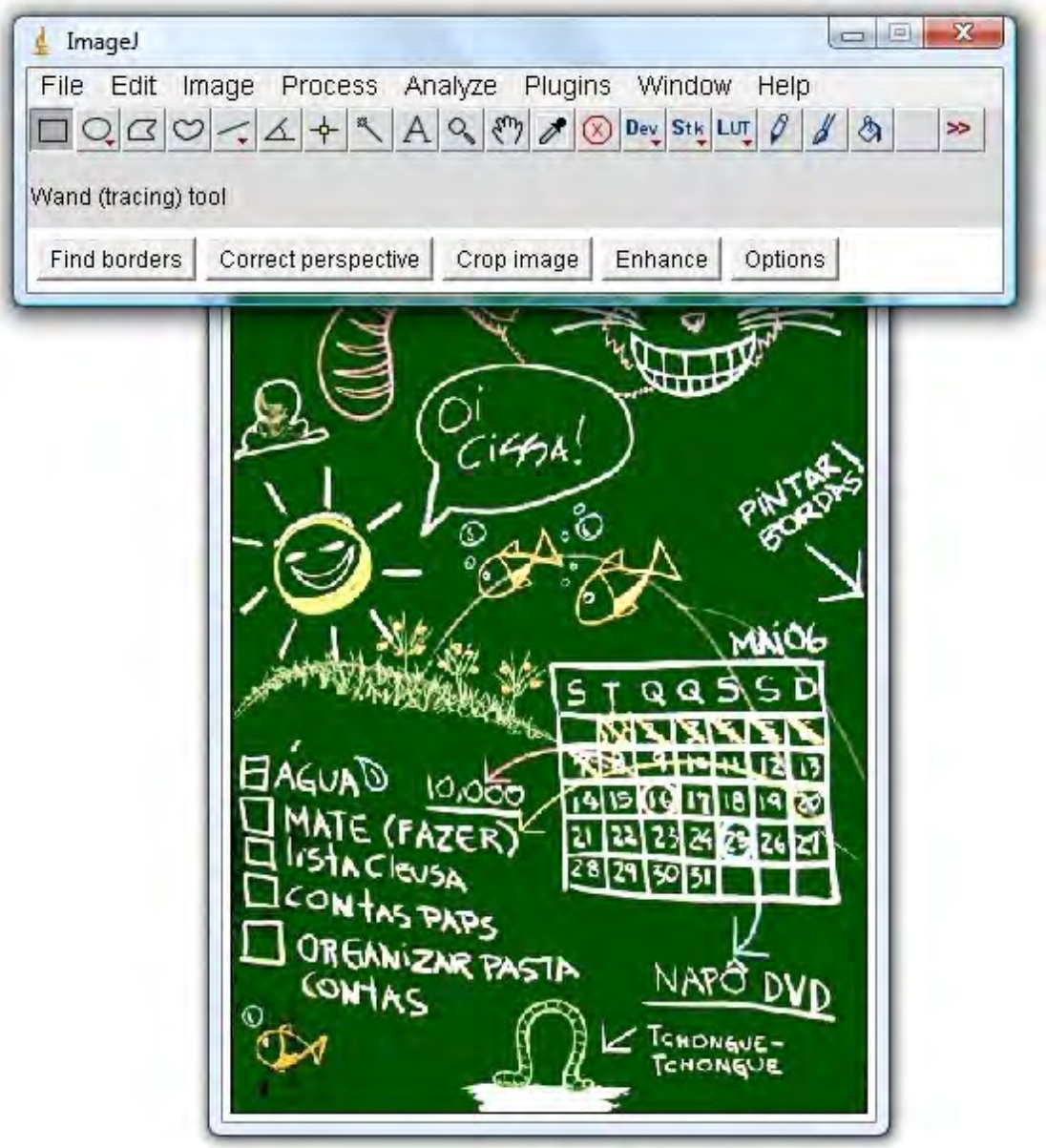


Figura 1.12 – *Imagem realçada de um quadro verde*

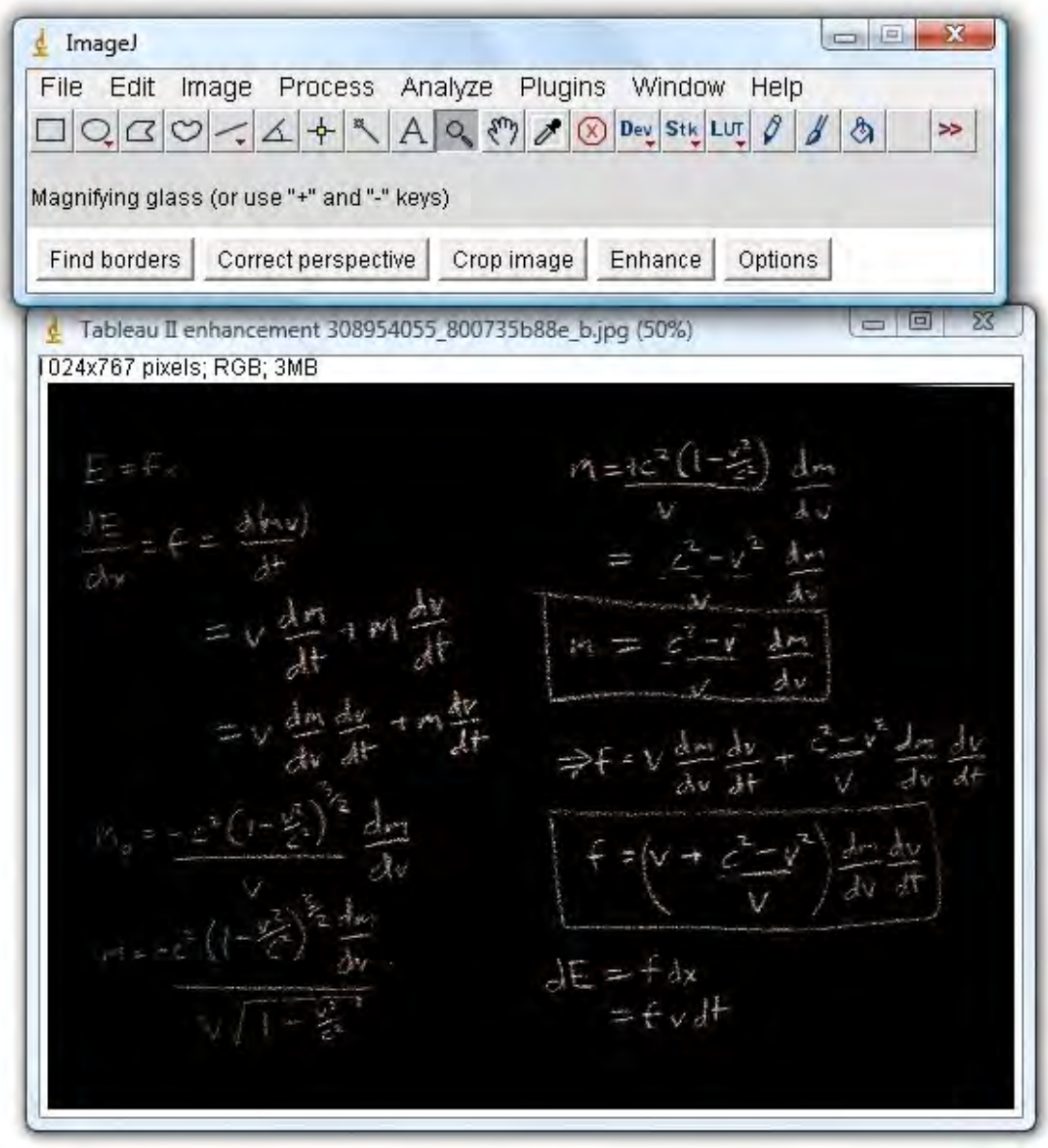


Figura 1.13 – *Imagem realçada de um quadro preto*

Como o objeto de estudo desta dissertação foram quadros didáticos, restringiu-se aos quadros com fundos branco, preto e verde, pois são os mais comuns. Porém o sistema permite que o usuário adicione mais cores à lista de fundos conforme visto na tela da Figura 1.14. Os fundos adicionais poderiam pertencer a cadernos de anotações, cartolinas etc. Ressaltando-se que para selecionar novas cores utilizou-se um componente desenvolvido por Jeremy Wood e pode ser obtido em (JAVANET, 2008).

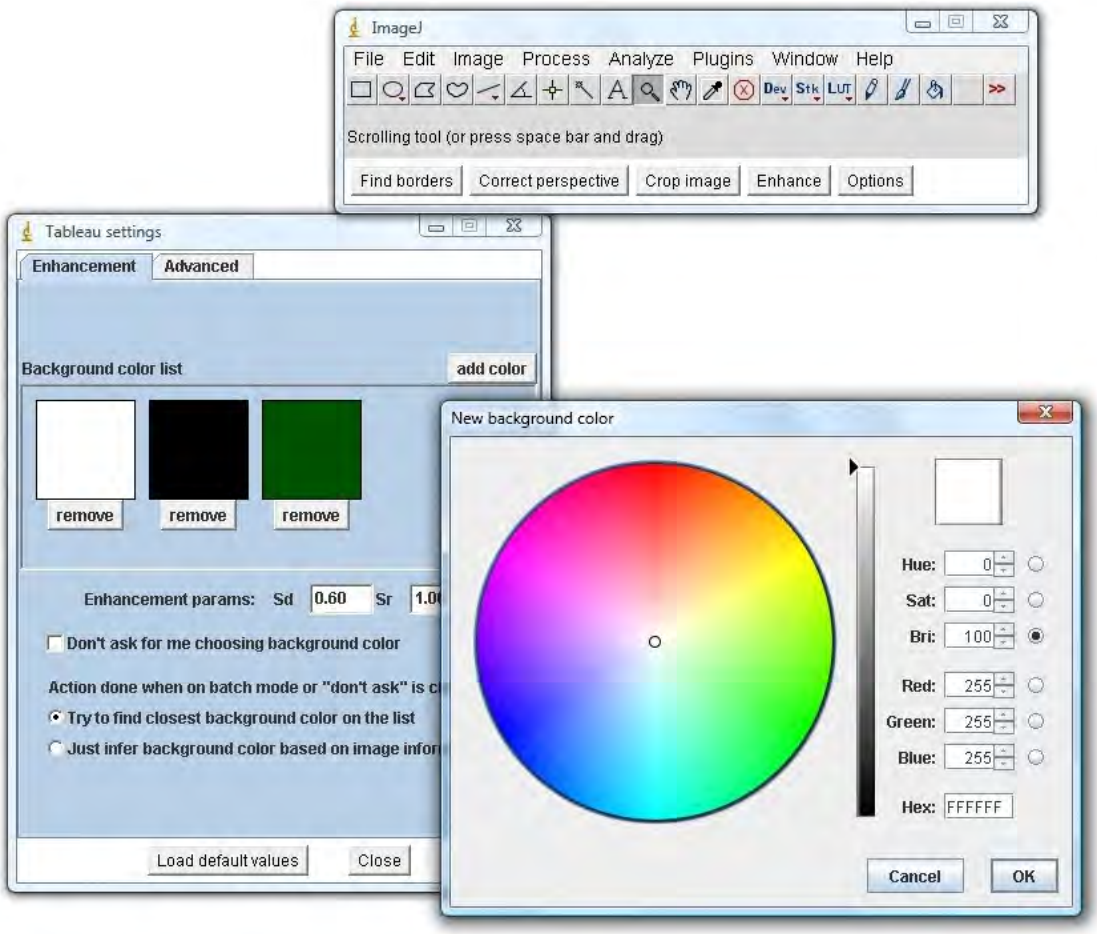


Figura 1.14 – *Tela que adiciona novas cores de fundo e podendo-se personalizar os parâmetros do algoritmo de realce*

O ambiente Tableau tem como principais características: execução multi-plataforma; integração com outros algoritmos que estão no ImageJ; ter como fundo de saída o fundo do quadro, não necessariamente branco; ser gratuito e de código aberto.

1.2 Trabalhos correlatos

Tableau é um ambiente inovador e com diversos aspectos pioneiros, sendo o único trabalho correlato descrito na literatura acadêmica é o sistema *WhiteboardIt!*. Esse é um sistema proprietário usado internamente na Microsoft® desenvolvido exclusivamente para quadros brancos. Os algoritmos de *WhiteboardIt!* estão parcialmente descritos em (LIU *et al*, 2007; ZHANG and TAN, 2001; ZHANG and HE, 2004; ZHANG *et al*, 2006). No entanto, as especificações dos seus algoritmos apresentam falhas que serão abordadas mais adiante.

No mercado há quatro sistemas comerciais com propósito semelhante a Tableau. Dois desses são executados localmente (as rotinas são processadas na própria máquina do usuário) e dois serviços *on-line* na Internet, conforme a Tabela 1.1.

Tabela 1.1 – *Sistema comerciais para processamento de imagens de quadros didáticos*

Sistema	Tipo
Whiteboard Photo (POLYVISION, 2008)	Execução local
ClearBoard (SOFTTOUTHIT, 2008)	Execução local
QipIt® (REALEYES3D, 2008)	Serviço “on-line”
ScanR® (SCANR, 2008)	Serviço “on-line”

Nenhum desses sistemas realça imagens com fundo de cor qualquer. No caso dos sistemas de execução local, o realce pode ser aplicado a imagens de quadro branco, preto e verde, com o fundo de saída branco, útil na impressão da imagem.

O QipIt® e ScanR® se restringem a quadros brancos, conforme dito nos seus endereços na Internet www.qipit.com e www.scanr.com, respectivamente. Mesmo assim, as imagens de quadros pretos e verdes foram testadas não obtendo resultados satisfatórios conforme será visto adiante.

Portanto, Tableau se mostrou um ambiente comparável aos sistemas comerciais, com qualidade similar na detecção de borda e superior no realce. Comparações foram feitas através de inspeção visual, sendo o seu o resultado discutido mais adiante nesta dissertação.

1.3 Contribuições

Esta dissertação tem como contribuição o desenvolvimento de dois novos algoritmos para a detecção de borda de quadros didáticos, e dois novos algoritmos para o realce desse tipo de imagem. No caso do realce da imagem, a segunda proposta é a única na literatura técnica que consegue realçar quadro com qualquer cor de fundo.

A primeira proposta da detecção de de borda foi publicada em (OLIVEIRA and LINS, 2007). A segunda proposta de detecção de borda e a primeira de realce foi publicada em (OLIVEIRA and LINS, 2008). A segunda proposta de realce foi submetida e está em processo de avaliação no momento que esta disertação foi escrita.

1.4 Imagens utilizadas

Para realizar os estudos e teste apresentados nesta dissertação, foram coletadas imagens da Internet dos portais Flickr® (FLICKR, 2008) e Google Picasa® (GOOGLE, 2008), algumas foram cedidas por Gilles Rockefort da RealEyes 3D (empresa que desenvolve o QipIt®) e outras fotos tiradas pelo próprio autor desta dissertação.

As fotos tiradas pelo autor visam verificar a eficácia dos algoritmos em imagens capturadas tanto a partir de câmeras portáteis digitais quanto de câmeras embutidas em telefones celulares, onde a resolução e qualidade são inferiores aos dispositivos de uso específicos. O estudo com as imagens obtidas por telefones celulares é essencial, pois o seu uso está se tornando cada vez mais freqüente em face da sua disponibilidade.

1.5 Visão geral desta dissertação

Esta dissertação contém sete capítulos, incluindo o presente capítulo de introdução.

O segundo capítulo descreve, resumidamente, os princípios de funcionamento de câmeras digitais dedicadas e de aparelhos celulares.

No terceiro capítulo, demonstra-se o problema da detecção da borda do quadro. A importância da delimitação é fundamental para os mecanismos de correção de perspectiva e eliminação das áreas que não pertencem ao conteúdo do quadro.

No quarto capítulo, apresentam-se os algoritmos de correção de perspectiva. Esses algoritmos visam fazer uma transformação na imagem de modo que dêem a impressão que o plano da figura é paralelo ao da lente da câmera.

No quinto capítulo, apresenta-se o desafio de se realçar as imagens de quadros didáticos. É através do realce que o quadro se torna mais legível e retira o fator de iluminação irregular, além de aumentar o contraste entre risco e fundo do quadro.

No sexto capítulo, é feita uma comparação entre os sistemas existentes para processamentos de quadros didáticos. Esses sistemas são a ponte entre o usuário final e os algoritmos abordados. A usabilidade e desempenho também são avaliados neste estudo.

No sétimo capitulo, é apresentada uma visão geral das contribuições desta dissertação, bem como os trabalhos futuros que podem ser desenvolvidos.

Três apêndices também compõem esta dissertação. O primeiro deles apresenta as publicações do autor obtidas ao longo do trabalho:

- *Tableau – Processing Teaching-Board Images Acquired with Portable Digital Cameras*. Proceddings of Second International Workshop on Camera-Based Document Analysis and Recognition, Curitiba, Brasil, 22 de setembro de 2007.
- *Improving the Border Detection and Image Enhancement Algorithms in Tableau*, International Conference on Image Analysis and Recognition, Póvoa Varzim, Portugal, 25-27 de junho de 2008.

Os apêndices B e C são relativos ao sistema CIE L*a*b* e os sistemas de diferenças de cores, respectivamente, sendo esses utilizados no novo algoritmo de realce a ser descrito no capítulo 5.

Em anexo a essa dissertação pode ser encontrado um DVD contendo:

- Base das imagens utilizadas,
- Imagens da base processadas em Tableau e ferramentas similares,
- Código fonte do plugin do ImageJ,
- Código de instalação do ImageJ.

2 Câmeras Digitais

Uma câmera fotográfica pode ser modelada como uma câmera escura, que é representada por uma caixa com um pequeno orifício onde a imagem é projetada de cabeça para baixo no lado oposto ao orifício. A Figura 2.1 ilustra esse modelo, que também é conhecido como *pinhole* (GONZALEZ and WOODS, 2008).

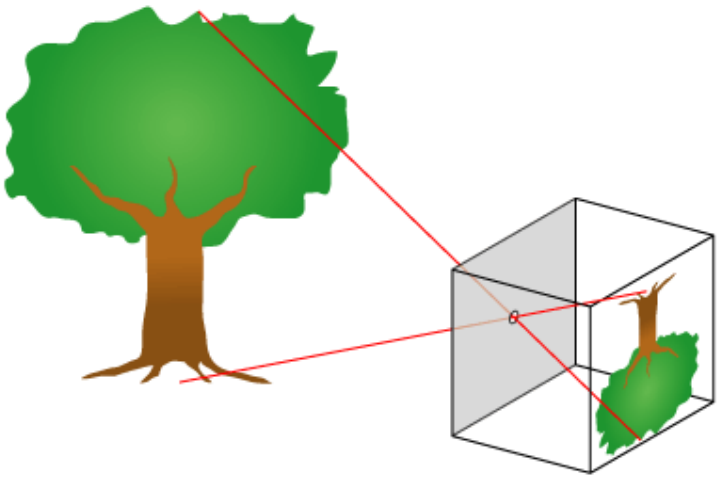


Figura 2.1 – *Modelagem de projeção da imagem a partir de uma câmera escura com furo (MELLISH, 2008)*

Uma câmera fotográfica analógica simples possui os seguintes componentes:

1. Visor – permite ver o que vai ser fotografado
2. Bobina do filme e mecanismo de transporte – serve para rodar o filme à medida que as fotos são tiradas
3. Filme – Na abertura do obturador a luz passa e incide no filme, registrando a cena
4. Obturador – controla o tempo em que o filme é exposto para registrar a foto
5. Abertura – controla a quantidade de luz que é passada para o filme, quanto maior for mais luminosa vai ser a imagem
6. Lente – Concentra a luz do ambiente externo para projetar no filme a imagem. A fabricação de lentes com qualidade envolve tecnologia bastante complexa, o que não será comentado nesta dissertação pois foge ao escopo da mesma.

7. Mecanismo de focalização – desloca a lente para frente ou trás, a fim de enfatizar a parte da cena a ser fotografada. Esse mecanismo pode ser feito de forma manual, automática ou inexistir dependendo do modelo do dispositivo.

Todos esses componentes estão ilustrados na Figura 2.2.

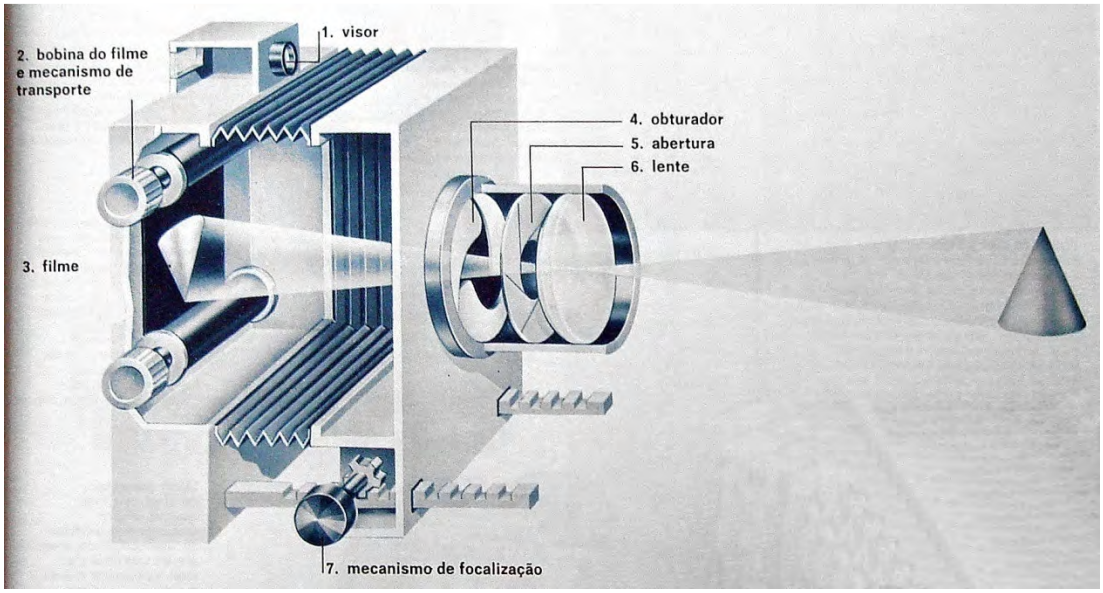


Figura 2.2 – *Diagrama demonstrando uma câmera analógica simples (BUSELLE,1977)*

Numa câmera digital, substitui-se o filme pelo sensor de captura, que transforma luz em sinais elétricos. Após a captura da imagem uma rotina de programação é executada para transformar os sinais elétricos em um arquivo de imagem. As câmeras digitais introduziram os visores LCD em que é possível ver exatamente o que está sendo fotografado, sendo uma alternativa ao visor tradicional.

Nesta dissertação, discorre-se em mais detalhes as tecnologias de sensores de capturas que estão em vigor atualmente, os mecanismos que usam esses sensores para formar a imagem final e o mecanismo de focalização da imagem. Todos os outros componentes possuem uma grande influência no resultado final da imagem, no entanto é necessário um estudo bem mais aprofundado o que foge ao escopo dessa dissertação. Para mais informações sobre os outros componentes o livro de Michael Busselle pode ser consultado (BUSELLE, 1977).

2.1 Sensores de captura

O componente responsável pela digitalização da imagem é o sensor de captura. Neles as imagens são projetadas e transformadas em sinais digitais. Atualmente existem duas tecnologias predominantes no mercado: o CCD e CMOS.

O termo em inglês para CCD é *charge-coupled device*, que quer dizer dispositivo de carga conjunta (acoplada). Essa tecnologia surgiu no final dos anos 60, inicialmente utilizado em câmeras filmadoras, que acabou substituindo o cinescópio (THEUWISSEN, 2007). O CCD é feito de silício e é constituído de uma matriz de pacotes de carga que capturam os pontos da imagem. Depois da exposição, a carga dos pacotes é transmitida para uma unidade comum que converte a carga em sinal digital. Devido a essa arquitetura os CCD funcionam através da leitura/escrita seqüencial uniforme. A transmissão da carga conjunta diminui a interferência de ruídos na imagem.

Já o termo CMOS vem do inglês *complementary metal-oxide-semiconductor*, traduzindo para o português "semicondutor metal-óxido complementar". A tecnologia empregada é a mesma utilizada na fabricação de outros componentes eletrônicos, o que possibilita integração com outros componentes (módulo de pós-captura etc.). Essa tecnologia permite também que cada ponto seja acessado independentemente e a leitura é feita por linhas da imagem.

A principal vantagem do CMOS sobre o CCD é o menor custo de produção, já que na produção do chip pode ser integrado com o circuito que faz o processamento dos dados pós-captura, em contrapartida o CCD é um componente à parte. O CMOS consume menos energia e ocupa um espaço menor. Contudo, existem várias desvantagens do CMOS em relação ao CCD. A relação sinal-ruído (SNR) é maior no CMOS que no CCD; a qualidade da imagem do CMOS não é tão boa quanto ao CCD devido ao primeiro possuir maior sensibilidade a ruídos; o CMOS apresenta dificuldades ao capturar imagens no escuro; o custo de pesquisa do CMOS é mais alto que o CCD, pois se fabrica o CMOS com vários componentes integrados e o CCD é um componente à parte.

Geralmente, as câmeras de telefones celulares usam a tecnologia CMOS, e as câmeras dedicadas (profissionais e amadoras) ambas as tecnologias. O CMOS também é utilizado em inspeções industriais (verificação de soldas etc) onde há intensa exposição à luz. Apesar das referências acadêmicas mais recentes afirmarem a qualidade superior do CCD em relação ao CMOS, as melhores câmeras profissionais utilizam sensores CMOS,

conforme observado no site http://www.dpreview.com (DPREVIEW, 2008), demonstrando a concretização da tendência na superação do CMOS pelo CCD. Por exemplo, as câmeras Nikon D3 (12Mpixel), Sony (10.2Mpixel) e Pentax K20D (14,6Mpixel) são algumas das câmeras que utilizam sensores CMOS (DPREVIEW, 2008),.

Mais detalhes sobre o CCD podem ser vistos em (RAINER, 2007) e (PERES, 2007); sobre o estado da arte do CMOS em (THEUWISSEN, 2007; THEUWISSEN, 2008) e (FARAMARZPOUR *et al*, 2007); sobre as comparações feitas acima em (HAIN *et al*, 2007; EOM *et al*, 2007).

2.2 Mecanismos de capturas de imagens

Os sensores captam a luz numa determinada faixa de freqüência. No caso das câmeras digitais, a luz visível com sensores nas faixas de verde, vermelho e azul. O que diferencia um mecanismo de captura de outro é o arranjo dos sensores, a quantidade de vezes que eles são expostos, o tempo de exposição e como a imagem final é construída. Os mecanismos podem ser usados tanto no CCD ou CMOS. Geralmente os fabricantes informam qual a tecnologia emprega (CCD ou CMOS), tempo de exposição e a distância focal, sendo as outras informações raramente descritas. Todos os métodos citados foram baseados em (PERES, 2007), além desses métodos esta referência apresenta outros não são relevantes para a dissertação.

2.2.1.1 Mecanismos “uma exposição/sensor em linha”

Neste tipo de mecanismo três linhas de sensores são usadas para capturar as linhas da imagem. À medida que a imagem é capturada esse sensor move em uma direção até capturar todo conteúdo da imagem. A principal vantagem desse tipo de mecanismo é que a imagem capturada possui alta resolução e o custo do equipamento é menor em relação aos outros que serão vistos a seguir, porém é necessário que o objeto capturado esteja estático. Este método pode ser encontrado em scanners comerciais e câmeras industriais para verificar linhas de produção. Nesse último caso o rastreamento é feito com alta freqüência.

2.2.1.2 Mecanismos “uma exposição/ sensor em área”

Neste tipo de mecanismo constitui na abertura do feixe de captura uma única vez, onde a luz incide sobre uma matriz de sensores e a imagem é convertida para o formato digital.

O primeiro método é o sistema de captura com uma exposição utilizando o padrão de Bayer (BURNETT, 2006). Esse sensor é constituído em uma matriz com sensores das cores primárias (verde, vermelho e azul) conforme descrito na Figura 2.3.

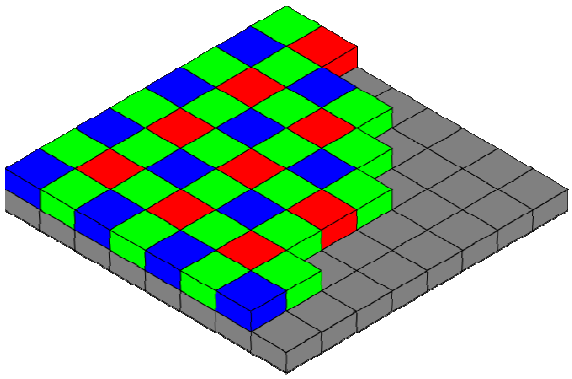
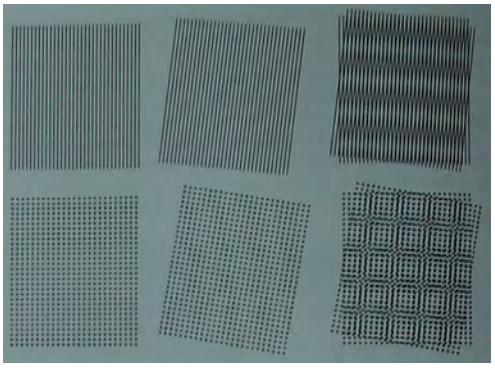


Figura 2.3 – *Padrão de Bayer (BURNETT, 2006)*

A imagem final é constituída através de um processamento com interpolações para se estimar a cor de cada ponto, uma vez que cada ponto define a intensidade de uma componente e deseja-se uma imagem em que cada ponto possui as três componentes. A interpolação pode resultar em imagens com algumas falhas, como o padrão de Moiré, que é a superposição entre o mesmo padrão orientado por ângulos diferentes (GONZALEZ and WOODS, 2008; CBURNETT, 2008), e o aliasing, que é a transição não-suave entre cores (GONZALEZ and WOODS, 2008). Padrão de Moiré pode ser visto pela “onda” na Figura 2.4 (b) e o aliasing na Figura 2.6, nos riscos do quadro.



(a) Exemplos de padrões de Moiré com linhas retas e pontos (GONZALEZ and WOODS, 2008)

(b) Padrão de Moiré nos tijolos de um prédio (CBURNETT, 2008)

Figura 2.4 – *Padrão de Moiré*

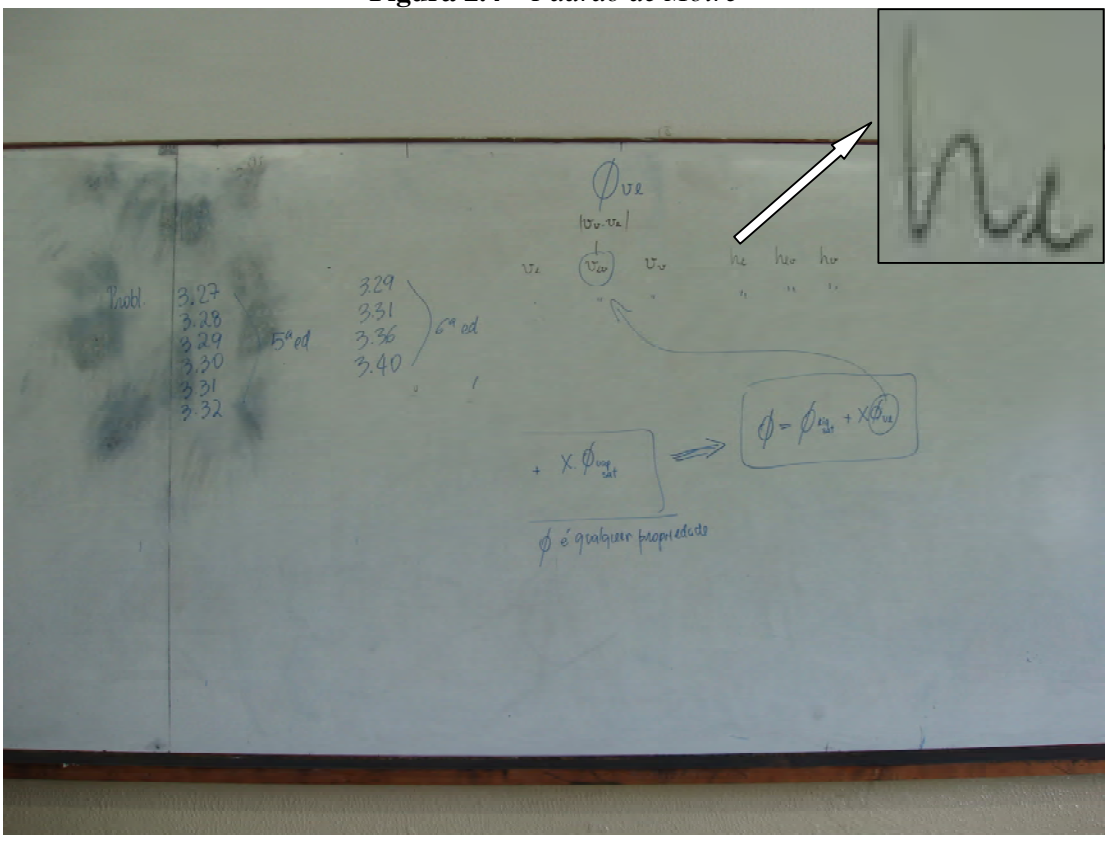


Figura 2.5 – *Aliasing*

Nota-se que a quantidade de verde é maior, pois o olho humano é mais sensível a essa cor e o brilho é formado, em sua maior parte, pelo verde. Com esse método obtém-se uma imagem com qualidade razoável a um baixo custo. No entanto, alguns fabricantes podem indicar como a resolução da imagem através da quantidade de pontos no sensor, ao informar a "resolução interpolada", omitindo a resolução efetiva que é ½ da área de *pixels* para verde e ¼ para azul e ¼ vermelho.

Para deixar a imagem mais agradável, as câmeras executam rotinas de pós-processamento que minimizam o efeito do padrão de Bayer, e as distorções de *aliasing* e *Moiré*. No caso dos quadros didáticos, essas rotinas podem acarretar na perda de informação dos riscos, que é o caso da Figura 1.2 (b) e de algumas figuras que serão vistas a seguir onde o risco está "borrado", sendo provavelmente conseqüência da execução dessas rotinas. No entanto, nas fotos comuns essas rotinas melhoram a qualidade da imagem em relação à imagem sem processamento. O padrão de Bayer não é o único arranjo disponível, outros padrões podem ser vistos em (PERES, 2007).

O segundo método é constituído de uma matriz de pixel para cada componente primária. As matrizes residem em camadas separadas. Cada camada absorve a faixa de freqüência da sua componente. A vantagem é que não se faz necessário o uso de interpolações ou filtros ópticos para se obter a imagem final, no entanto a sua fabricação é mais complexa e cara.

2.2.1.3 Mecanismos "múltiplas exposições/um sensor em área"

Ao invés de expor a imagem uma única vez ao sensor de captura, o mecanismo permitem que a luz entre pelo menos 3 vezes em instantes distintos, o que garante melhor resolução, pois dispensa interpolações. Porém o uso é restrito a imagens estáticas.

O primeiro mecanismo é "três exposições a uma matriz de sensor monocromático". Nesse esquema sensores monocromáticos são expostos três vezes à luz. A cada exposição o filtro óptico da componente é utilizado, deixando passar apenas as freqüências da componente.

O segundo método constitui na exposição da luz 4 vezes em uma matriz com padrão de Bayer. O que muda é o arranjo dos sensores que são "rotacionados" a cada exposição. Quatro exposições são suficientes para capturar todas as componentes do ponto sem interpolações. No entanto, como o verde possui 2 vezes mais amostra que as outras duas

componentes, uma média aritmética é feita entre os valores da componente em cada ponto. O rodízio do padrão está ilustrado na figura abaixo.

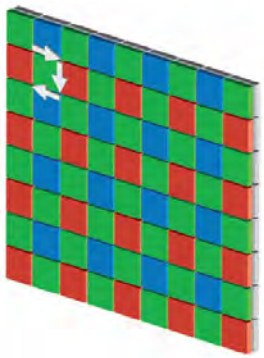


Figura 2.6 – *Rodízio de sensores com 4 exposições (PERES, 2007)*

2.3 Mecanismos de foco

O objetivo do foco é enfatizar uma determinada parte da cena a ser fotografada. O mecanismo mais simples é o manual. Porém é mais prático, na maioria dos casos, que a focalização seja automática. Atualmente existem três tipos de focalização automática: passiva, ativa e preditiva. Todos esses mecanismos foram obtidos das referências (WIKIPEDIA, 2008), (BROWN, 2008) e (NIKON, 2008).

2.3.1 Focalização ativa

A focalização ativa consiste em um dispositivo sensor que emite ondas ultra-som, essas ondas batem no objeto e voltam, obtendo assim o tempo de ida e volta. Como é sabida a velocidade da onda é possível inferir a distância do objeto principal, e daí ajustar a lente para frente ou trás, focalizando a imagem. Também se utilizam ondas infravermelhas, onde a distância é calculada por triangulação. Um problema desse mecanismo ocorre quando há vidros entre o objeto a ser fotografado e a câmera, fazendo com que o mecanismo focalize no vidro e não no objeto. Outro problema ocorre na utilização do infravermelho quando o objeto focalizado é uma vela de aniversário pode confundir o sensor, e se objeto for de cor preta absorve a luz infravermelha. No entanto, com esse sistema é possível focalizar objetos onde a iluminação é precária, sendo eficiente para uso de flash.

2.3.2 Focalização passiva

A focalização passiva consiste em analisar a luz que entra na câmera e ajustar a posição da lente através dessa análise até que se obtenha o foco. Esse mecanismo de subdivide-se em dois: ajuste por contraste e detecção por fase.

No ajuste por contraste calcula-se o nível de contraste à medida que se move a lente. A posição que obter maior contraste é que vai ser utilizada para tirar a foto. Este é um mecanismo barato, presente em muitas câmeras comerciais amadoras. A Figura 2.7 ilustra um exemplo utilizando esse mecanismo, onde o maior contraste se encontra quando a lente está no meio.

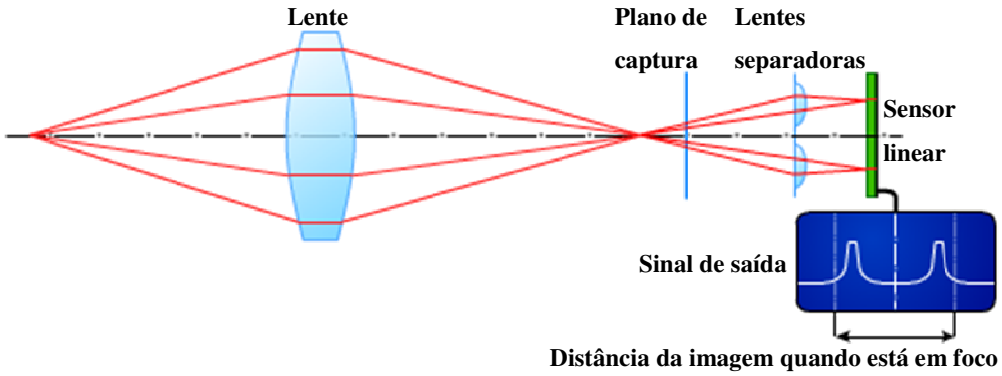
(a) Início do cálculo do contraste

(b) Contraste com maior nível

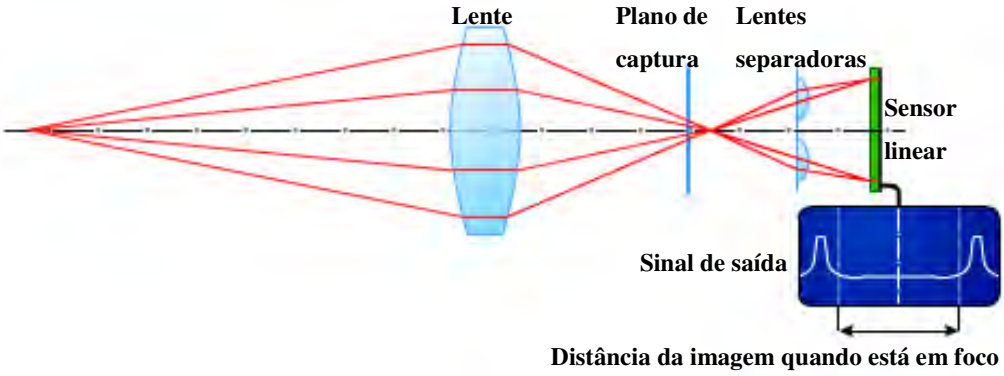
(c) Fim do cálculo do contraste

Figura 2.7 – *Focalização de ajuste por contraste (NIKON, 2008)*

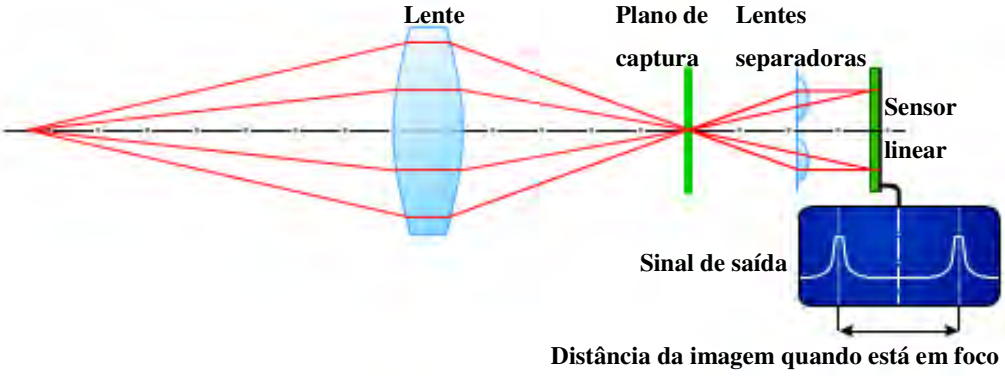
Na detecção por fase, a idéia é projetar duas imagens com origem no mesmo ponto e calcular a distância necessária para alinhar as imagens, que devem ser simétricas numa imagem focalizada. Isso é possível colocando dois sensores lineares em conjunto com lentes separadoras atrás do plano do dispositivo de captura, esses sensores vão capturar dois raios de luz de um mesmo ponto. Com o sinal obtido é possível calcular o quão é necessário no ajuste da lente para focalizar. A Figura 2.8 ilustra os casos em que a imagem é formada antes do plano do sensor de captura, depois do plano, e a mesma está em foco.



(a) A imagem fora de foco que é formada antes do plano do sensor



(b) A imagem fora de foco que é formada depois do plano do sensor



(c) A imagem em foco

Figura 2.8 – *Focalização por detecção por fases (NIKON, 2008)*

Esse processo é extremamente rápido, no entanto é muito mais caro, pois são necessários:

- lentes adicionais (lentes separadoras);
- sensores lineares adicionais;
- colocar um espelho na frente do sensor de captura, que deve ser removido ao capturar a foto, aumentando a complexidade de projeto.

Outro ponto fraco da detecção por fase é a necessidade de calibração do sensor, o que não acontece no ajuste por contraste.

2.3.3 Focalização preditiva

É um mecanismo proprietário desenvolvido pela Nikon para tirar seqüências de fotos em movimento. À medida que o objeto é focalizado, o sistema usa informação passada para prever a focalização futura, diminuindo o intervalo de tempo entre focalizar e tirar foto. Mais detalhes em (NIKON, 2008).

2.3.4 Foco pré-fixado

Já nas câmeras de baixo custo o foco é pré-ajustado na fábrica. Nesse cenário o fotógrafo deve tirar fotos com a distância mínima de 1 metro, que é razoável para amadores, pois este grupo geralmente tira fotos com sua família, de objetos e paisagens. O uso de foco pré-fixado é muito comum em câmeras embutidas em telefones celulares.

2.4 Resumo

Este capítulo teve como objetivo mostrar sumariamente os princípios e componentes de uma câmera digital. Um dos componentes da câmera é o sensor de captura que transforma a luz em dados digitais. Atualmente, existem duas tecnologias predominantes de sensores: o CCD e o CMOS. Em paralelo a essas tecnologias, existem vários mecanismos de captura da luz para obter uma imagem colorida, como visto na seção 2.2.

Visando garantir uma imagem nítida, a câmera digital hoje geralmente incorpora mecanismos de focalização automática. Há vários mecanismos com esse propósito como podem ser vistos na seção 2.3.

3 Detecção de borda

Neste capítulo são mostrados algoritmos para detectar borda da lousa do quadro, que serve de entrada para a correção de perspectiva e corte da imagem. A detecção de borda consiste na delimitação automática entre o conteúdo do quadro e área externa a esse conteúdo.

3.1 Detecção de bordas

A detecção de bordas é uma das áreas de estudo mais antigas na área de processamento de imagens. A idéia básica é realçar as áreas das imagens que apresentam mudanças abruptas de sinais, no caso de imagens coloridas esses sinais são as cores dos pixels. Esta seção tem o intuito de fazer uma introdução breve sobre o assunto, para mais detalhes sobre esse assunto a referência (GONZALEZ and WOODS, 2008) pode ser consultada.

A maioria dos mecanismos tem como base a convolução de uma máscara sobre a imagem. O primeiro ponto que deve ser validado é a direção da borda a ser detectada. Geralmente se consideram as direções: horizontal, vertical e diagonal (em 45º e 135º).

A forma mais simples é através de gradientes das imagens. Gradiente é o vetor de derivadas $\langle g_x, g_y \rangle$. Esse vetor tem como direção a reta normal em relação a borda da imagem como ilustrado na Figura 3.1. A magnitude do vetor pode ser denotada por $M(x, y)$ e o ângulo por $\tan^{-1}$. Em muitos casos o cálculo da magnitude é simplificado para "$|g_x| + |g_y|$", pois o custo computacional é menor e a aproximação é razoável.

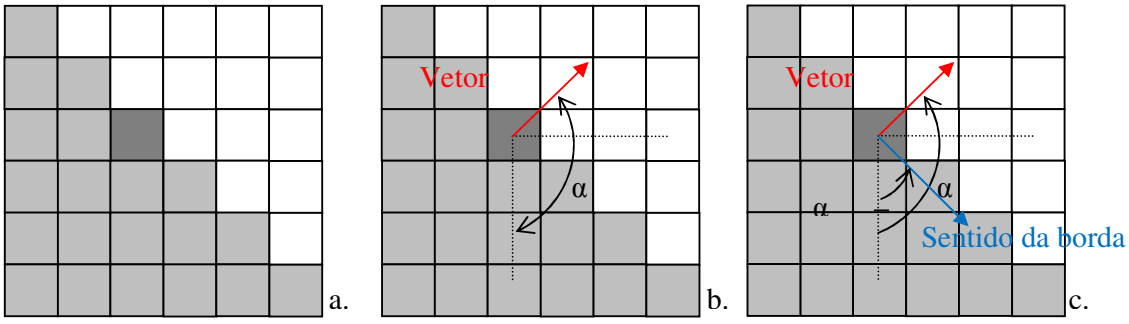


Figura 3.1 – *Ponto analisado (a), vetor gradiente (b), vetor gradiente fazendo um ângulo reto com o sentido da borda (c)*

O vetor gradiente pode ser expresso por

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad 3.1$$

, a magnitude deste vetor por

$$M(x, y) = \sqrt{{g_x}^2 + {g_y}^2} \quad 3.2$$

e o seu ângulo por

$$\alpha(x, y) = \tan^{-1}(g_x, g_y) \quad 3.3$$

.

No caso das imagens, a função f é a matriz da imagem, que pode ser vista como uma função bidimensional com domínio nos número naturais. O cálculo da componente g_x é definido por

$$g_x = \frac{\partial f(x,y)}{\partial x} = f(x+1, y) - f(x, y); \quad 3.4$$

e g_y é definido por

$$g_y = \frac{\partial f(x,y)}{\partial y} = f(x, y+1) - f(x, y) \quad 3.5$$

Observa-se as Equações 3.4 e 4.5 detectam bordas horizontais e verticais. Roberts (ROBERTS, 1965) definiu uma máscara para calcular bordas diagonais que podem ser vistas na Figura 3.2.

$\begin{bmatrix} -1 \\ 1 \end{bmatrix}$ a. $\quad [-1 \quad 1]$ b. $\quad \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$ c. $\quad \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$ d.

Figura 3.2 – *Máscaras para cálculos dos gradientes: vertical (a), horizontal (b), diagonal (c) e (d)*

O problema dessas máscaras se depara na implementação. A definição acima não apresenta qual o ponto central para o cálculo do gradiente. Para contornar essa situação utilizam-se máscaras 3x3 que aproximam melhor o valor da derivada parcial. Uma aproximação do operador de Roberts foi sugerida por Prewitt (PREWITT, 1970). No entanto Söbel (SÖBEL, 1970) modificou essa máscara aumentando a influencia dos pontos centrais, o que torna a convolução menos afetada por ruídos na imagem.

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} -1 & 0 & -1 \\ -1 & 0 & -1 \\ -1 & 0 & -1 \end{bmatrix}$$

Figura 3.3 – ***Máscaras de Prewitt***

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \begin{bmatrix} -1 & 0 & -1 \\ -2 & 0 & -2 \\ -1 & 0 & -1 \end{bmatrix}$$

Figura 3.4 – *Máscaras de Söbel*

Para a detecção de bordas, é comum calcular o valor do gradiente na imagem inteira. Os pontos que apresentarem valores de magnitude $M(x, y)$ maiores que um limite é classificado como borda. Esse método é simples, porém pode não apresentar bons resultados, pois não é muito sensível a ruídos. É possível encontrar na literatura algoritmos mais sofisticados como o de Marr-Hildreth (MARR and HILDRETH, 1980) e Canny (CANNY, 1986), sendo este último o mais referenciado.

O algoritmo de Marr-Hildreth pode ser resumido conforme a seguir:

1. Filtrar a imagem de entrada com um filtro gaussiano passa-baixa
2. Convoluir a imagem resultante do primeiro passo com um filtro laplaciano – que é a segunda derivada da imagem. Um exemplo de uma máscara 3x3 para esse filtro é:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

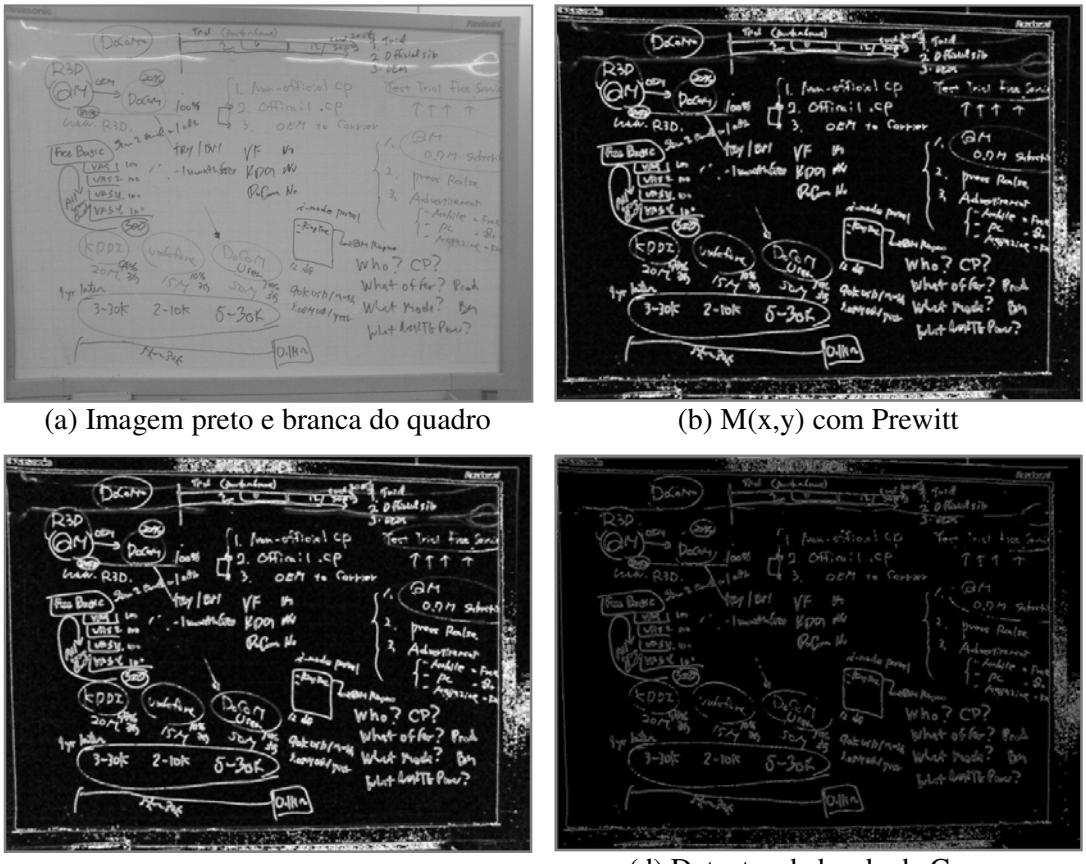
3. Os pontos que apresentarem mudanças de sinais é que vão ser as bordas das imagens

Já o detectar de borda de Canny é bem mais complexo, apresentando melhores resultados, no entanto é bem mais custoso em termos de espaço e desempenho. Este algoritmo pode ser resumido conforme os seguintes passos:

1. Suavizar a imagem de entrada com um filtro gaussiano
2. Computar $M(x, y)$ e $\alpha(x, y)$ utilizando algum operador de gradiente
3. Seja $g_N(x, y)$ uma matriz do tamanho da imagem que é atribuída da seguinte forma:
    a. Identificar os pontos vizinhos que estão na direção (vertical, horizontal ou diagonal) normal ao ângulo α do ponto em questão
    b. Caso a magnitude $M(g_x, g_y)$ do ponto em questão seja menor que qualquer um desses pontos, atribui-se $g_N(x, y) = 0$, caso contrário $g_N(x, y) = M(x, y)$

4. Verifica-se quais pontos em $g_N(x,y)$ possuem valores maiores que T_H (*High threshold*), que deve ser definido a priori, esses pontos são identificados como borda. Executam-se os seguintes passos a seguir.
    a. Procura-se por vizinhos das bordas atuais
    b. Qualquer vizinho que $g_N(x,y)$ for maior que T_L (*Low threshold*) é incluído na lista de bordas atuais
    c. Caso não haja nenhuma nova borda incluída, pára o algoritmo, caso contrário volta para o passo (a).

A figura abaixo mostra a execução dos algoritmos aqui apresentados, com exceção do Marr-Hildreth, em uma imagem de quadro branco. O valor limitante para os algoritmos de Prewitt e Sobel foi 30; os valores para o algoritmo de Canny foram T_L 120 e T_H 360.



(a) Imagem preto e branca do quadro
(b) M(x,y) com Prewitt
(c) M(x,y) com Söbel
(d) Detector de borda de Canny

Figura 3.5 – *Binarização de imagens de quadros brancos*

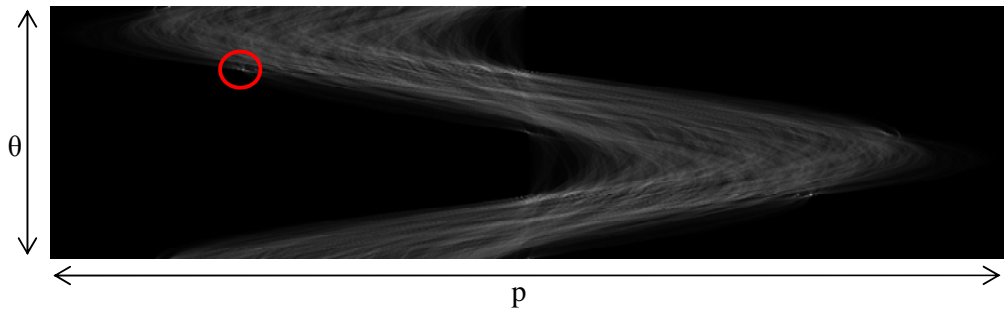
3.2 Transformada de Hough

O propósito inicial da transformada de Hough (HOUGH, 1962) é detectar linhas retas na imagem, mas o mecanismo pode ser estendido para detectar curvas, círculos ou outras formas. Uma reta pode ser definida pela equação $y = a \times x + b$, define-se um espaço de parâmetros a e b, que contém todos os possíveis valores para a e b que definem uma reta na imagem.

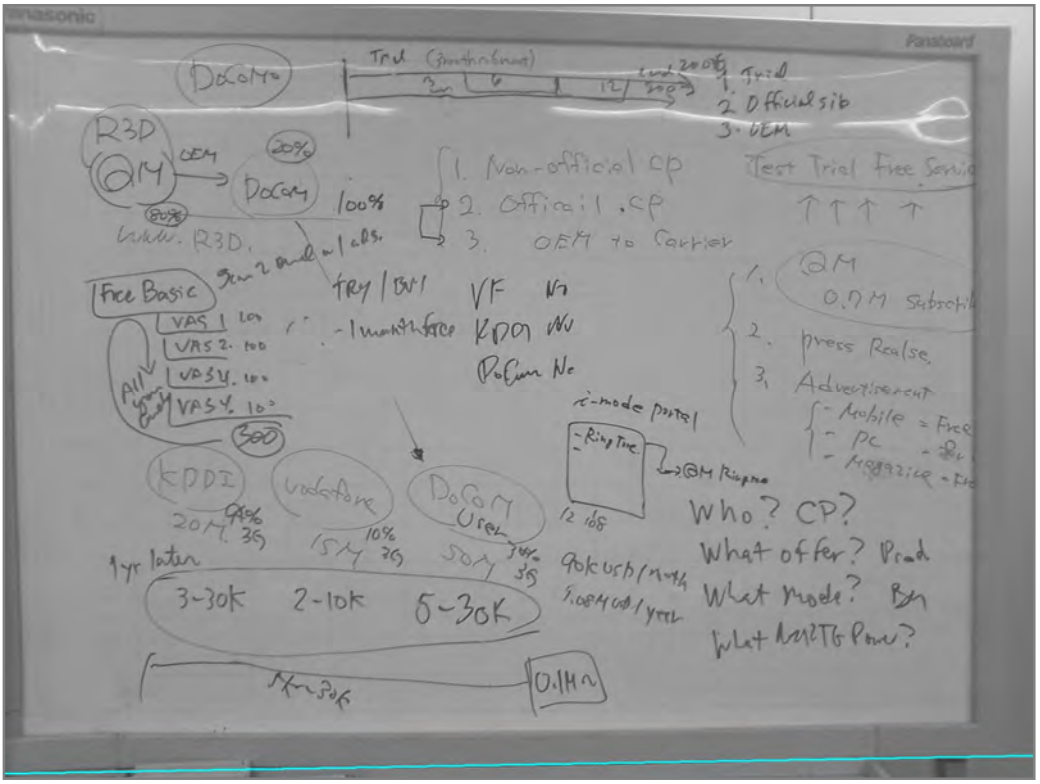
A transformada de Hough necessita de um mecanismo de detecção de bordas, como descrito na seção anterior. Para cada ponto classificado como borda incrementa-se o voto no espaço de parâmetros. Como por um ponto podem passar várias retas, os votos são incrementados nos valores de a e b que definem retas que passem pelo ponto em questão. As coordenadas (a, b) que tiverem muitos votos vão definir as retas presentes na imagem.

Um problema existente neste modelo é a dificuldade na representação das retas próximas ao eixo vertical. O valor de a neste caso tende a infinito, inviabilizando uma representação computacional. Para contornar essa situação emprega-se o uso da representação polar de reta $p = x \times \cos\theta + y \times \operatorname{sen}\theta$ (HART, 1972). A nova representação pode utilizar valores os valores de θ podem variar entre 0º e 359º, bem como p que assume valores $-D$ e D que é o valor máximo entre dois cantos da imagem. Da mesma forma que a proposta inicial, para cada ponto é calculado os valores de p e θ em que a reta passa pelo ponto. No final as células do espaço que apresentarem mais votos definem as retas das imagens.

A Figura 3.6 (a) mostra o espaço de Hough da Figura 3.5 (a). Para obter os valores foi aplicado o algoritmo de Söbel, considerando borda os valores dos pontos em a magnitude $M(x, y)$ fosse maior ou igual a 40. Já a Figura 3.6 (b) ilustra a reta que teve mais votos no espaço de Hough (círculo em vermelho na Figura 3.6 (a)) localizada na borda inferior do quadro.



(a) Votos no espaço de Hough



(b) Figura com a reta (em ciano) que teve mais votos no espaço de Hough

Figura 3.6 – *Definição do Espaço de Hough para a Figura 3.5 (a)*

3.3 WhiteboardIt!

Nesta seção será mostrada a parte do WhiteboardIt responsável pela detecção de borda. A descrição deste algoritmo pode ser vista em (LIU *et al*, 2007; ZHANG and TAN, 2001; ZHANG and HE, 2004; ZHANG *et al*, 2006).

3.3.1 Detecção de borda

O mecanismo de detecção de borda é baseado na transformada de Hough vista na seção anterior em imagens em tons de cinza. Inicialmente cálcula-se o gradiente da imagem. Serão considerados bordas os pontos em que $|G_x| + |G_y| \geq T_g$, ou seja a magnitude do vetor gradiente for maior que um limite. O próximo passo é incrementar os votos no espaço de Hough.

Primeiro, calcula-se o valor do ângulo θ com a formula $\tan^{-1}(g_y, g_x)$. Com esse valor, calcula-se o valor de p da equação polar da reta, $p = x \times \cos\theta + y \times \operatorname{sen}\theta$. Incrementa-se os votos no espaço de Hough na coordenada (p, θ). Nota-se que há uma diferença fundamental entre a idéia inicial de Hough que é incrementar os parâmetros que designam retas que passam pelo ponto atual, no caso do Whiteboard o incremento é baseado no ângulo do vetor da borda e no valor de p obtido pela equação polar da reta.

O próximo passo é achar um conjunto de quarto linhas definidas no espaço de Hough que atendam as seguintes restrições:

- As linhas devem ter no mínimo 5% do número máximo de votos no espaço de Hough.
- Linhas opostas devem ter entre 150 e 210 graus de diferença.
- A diferença do parâmetro p entre as linhas opostas deve ser no mínimo 1/5 da altura ou largura, isso garante que as linhas sejam suficientemente distantes para formar a borda do quadro.
- O ângulo entre duas linhas vizinhas deve ser entre 60 e 120 graus.
- Todas as linhas devem seguir uma orientação no sentido horário ou anti-horário.
- A circunferência inscrita dentro do quadrilátero deve ser maior que $\frac{largura+altura}{4}$. Essa restrição foi retirada conforme está descrita nos documentos do WhiteboardIt, notando-se a falta de precisão, pois dependendo do quadro não há circunferência que tangencie todos os lados do quadro.

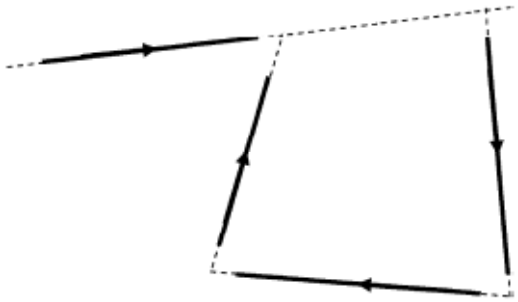


Figura 3.7 – *Exemplo de linhas que não formam um quadrilátero da borda (ZHANG et al, 2006)*

Uma vez definidas quatro linhas que possuem as restrições impostas anteriormente, um quadrilátero pode ser traçado. Para evitar que aconteça o cenário em que um objeto com borda bem definida (uma prateleira que está próxima ao quadro) venha ser detectado como parte da borda do quadro exemplificado na Figura 3.7, é feita a computação em cada lado do quadrilátero da quantidade de pontos que foram classificados como borda. Esse total é dividido pelo comprimento da circunferência inscrita no quadrilátero. O quadrilátero que tiver a maior razão é considerado como borda do quadro.

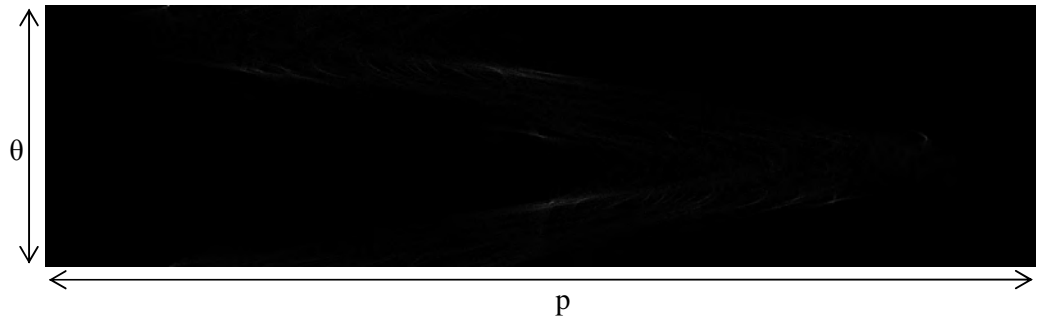
Devido à discretização do espaço de Hough, as linhas encontradas anteriormente não podem ser consideradas como borda do quadro. Portanto, uma busca por pontos classificados como borda é feita ao redor de cada linha em 10 pixels. Detectam-se pontos desconformes (também conhecido como pontos fora da curva ou *outliers*) através do método de quadrado da mediana (ZHANG, 1997). Após a eliminação das desconformidades, a linha da borda é definida através do método de menor erro quadrático médio com os pontos restantes (FAUGERAS, 1993).

É notável a presença de falhas na especificação do algoritmo do WhiteboardIt que impossibilitaram a sua implementação. A restrição sobre a coerência das orientações da linha não é necessária, pois se define como orientação o ângulo α calculado, se as quatros linhas atendem às restrições sobre os ângulos, respeitam também a orientação no sentido horário ou anti-horário. Outro problema encontrado é que não há garantia da circunferência inscrita no quadrilátero tangenciar todos os lados do quadrilátero, conforme dito anteriormente. Além de não considerar o efeito de distorção da lente e não tirar proveito da imagem ser colorida. Outra falha na especificação original se deve ao fato dos autores do WhiteboardIt não considerar as imagens parciais de quadro, como na Figura 1.2. O

desempenho do algoritmo também não é aceitável, com complexidade de $\Theta((largura + altura)^4)$, que será provado a seguir.

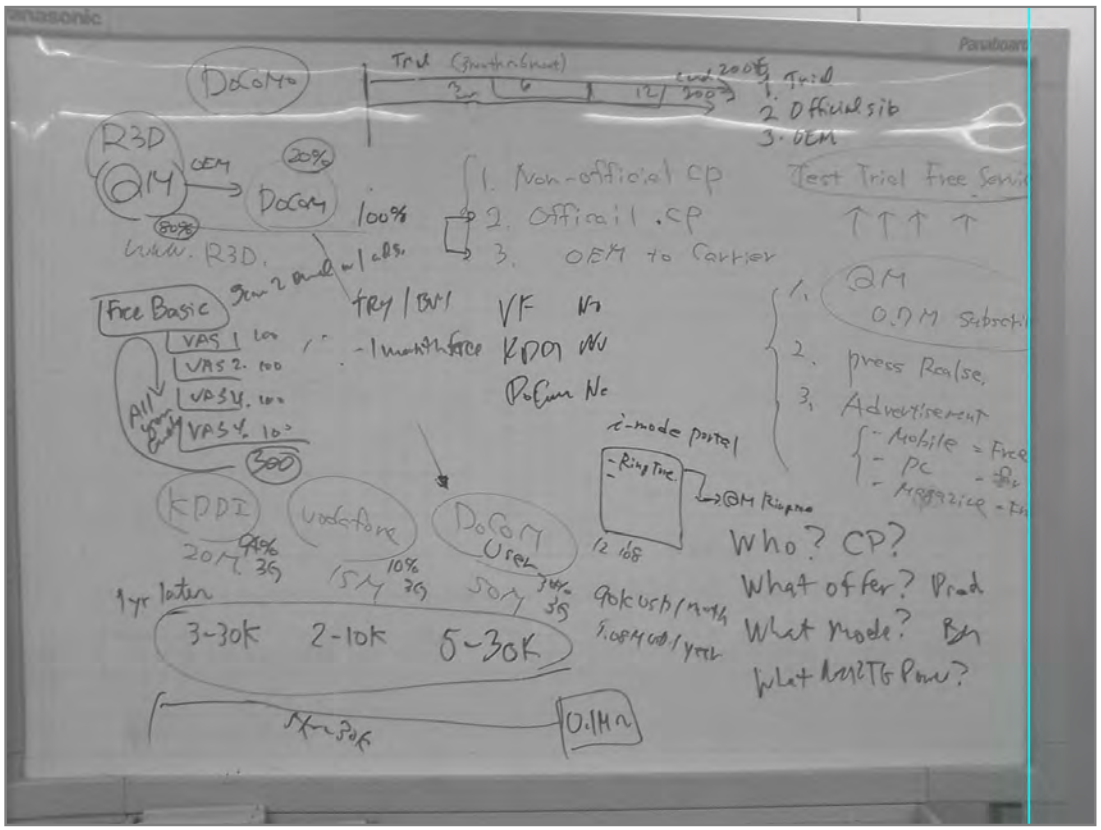
Considerando que o espaço de Hough possui dimensões $\theta \times p$. O valor de θ é fixo, independente do tamanho da imagem. Já o valor de p depende da soma $largura + altura$. Inicialmente, busca-se uma linha que tenha votos superiores a 5% da linha que tem mais votos. Faz-se a busca das outras linhas com as restrições impostas do algoritmo, e nenhuma das restrições reduzem a complexidade das buscas por linhas. Portanto a complexidade do algoritmo pode ser resumida no pior caso em $(largura + altura)^4$ uma vez que são 4 linhas.

No entanto foi possível implementar com desempenho aceitável a transformada de Hough. O θ pode ser calculado facilmente através da função *arctan*, onde os valores G_x e G_y podem retornar o θ variando de 0º a 359º. O espaço de Hough com a modificação do WhiteboardIt da Figura 3.5 (a) pode ser visto logo na Figura 3.8 (a), já em (b) é possível visualizar a reta mais votada do espaço de Hough em ciano localizada na borda do lado direito.



(a) Espaço de Hough de acordo com o WhiteboardIt

Figura 3.8 – *Detecção de borda do WhiteboardIt*



(b) Figura com a reta (em ciano) que teve mais votos no espaço de Hough

Figure 3.8 (cont.)

3.4 Um Algoritmo proposto para detecção de borda

A proposta de detecção de borda a ser apresentada foi publicada em (OLIVEIRA and LINS, 2007).

Observa-se que a estrutura de uma imagem de quadro didático é constituída: pela área disponível para colocar o conteúdo; pelos objetos ao redor do quadro, como tubos de ensaio, projetor, parede etc; e, em muitos casos, pela moldura do quadro. A figura seguinte mostra essas diferentes regiões.

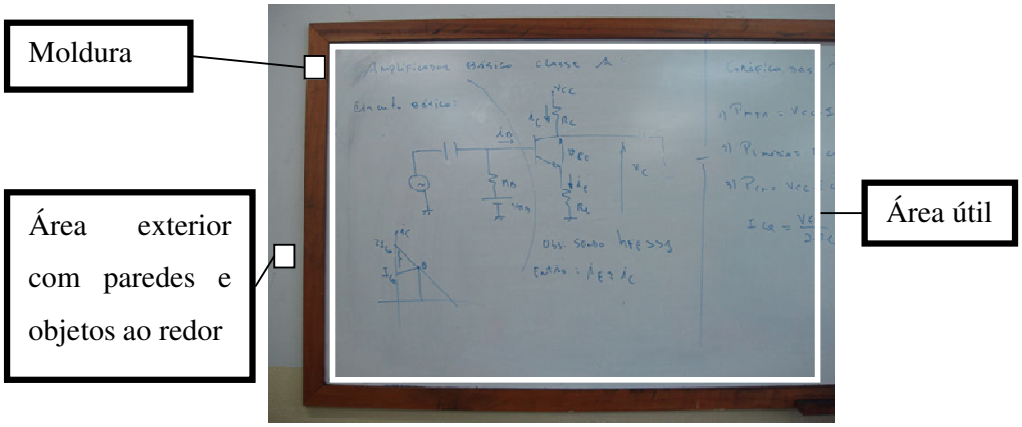


Figura 3.9 – *Regiões típicas de um quadro didático*

Nota-se que a única área que tem uma estrutura comum entre todos os quadros é a área de conteúdo, e que a transição de cores é brusca entre a superfície do quadro e a área ao redor (moldura, parede etc.). Os riscos que compõem o quadro também apresentam mudanças bruscas de cores em relação à superfície do quadro, porém os riscos são bem mais finos que a moldura do quadro. Já a superfície do quadro apresenta transição suave devido ao fator de iluminação.

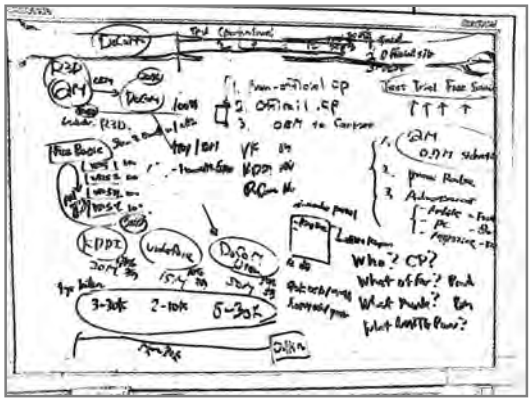
3.4.1 Segmentação

Tableau faz uso de um novo mecanismo de segmentação. Antes de entrar em detalhes de como ele funciona, a notação a ser utilizada é: o *pixel* de origem vai representar o ponto que vai ser classificado como borda ou não. O *pixel* de apoio vai representar o ponto que o *pixel* de origem vai usar como base da comparação. Portanto, para cada ponto existe apenas um ponto de origem e, dependendo do algoritmo, podem existir vários pontos de apoio.

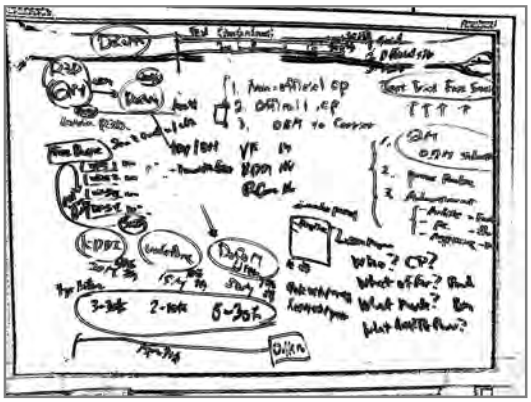
A maioria dos métodos clássicos de detecção de bordas baseia-se em computações entre os pixels circunvizinhos. O novo mecanismo para detecção de borda onde o pixel de apoio é o circunvizinho mais próximo do centro da imagem. Se a diferença as componentes entre o ponto de origem e de apoio ultrapassar um limite previamente estipulado, o pixel é marcado como borda. Como a moldura é bem mais espessa que o risco do quadro, “aumenta-se” a distância entre os pixels de origem e apoio. As regiões da borda dos quadros vão ser caracterizadas com uma região com grande concentração de pontos classificados como borda, ao contrário dos riscos do quadro em que a concentração

é bem menor. As figuras abaixo são resultados da nova segmentação aplicada à Figura 3.5 (a), com distâncias variando de 2 a 64 pixels, onde a borda é identificada pela cor preta.

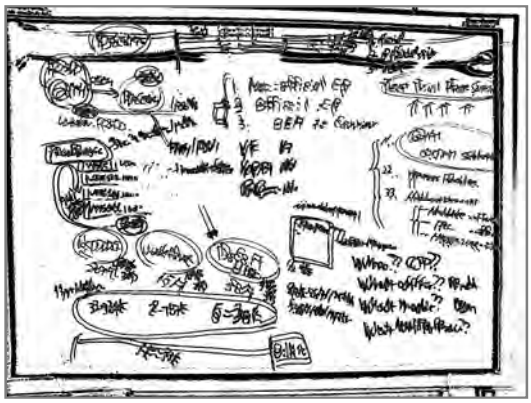
(a) Distância 2 pixels



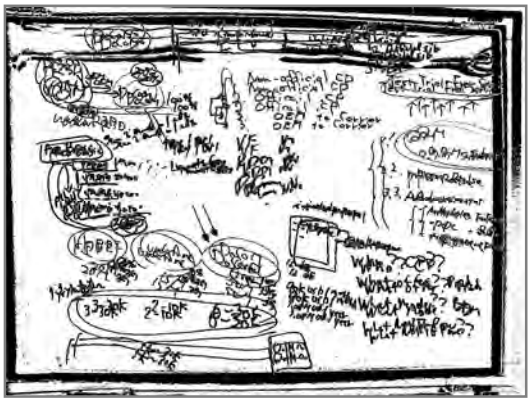
(b) Distância 4 pixels



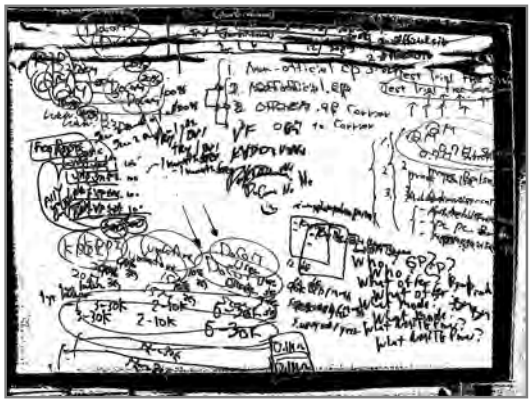
(c) Distância 8 pixels



(d) Distância 16 pixels



(e) Distância 32 pixels



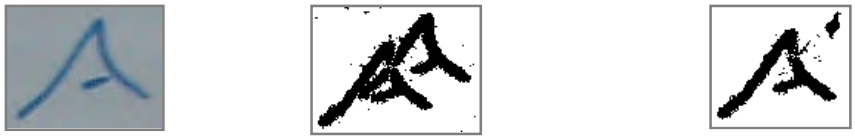
(f) Distância 64 pixels

Figura 3.10 – *Nova segmentação aplicada à Figura 3.5 com diversas distâncias*

Observa-se que, à medida que se aumentou o valor da distância, a moldura se tornou cada vez mais espessa. O intuito do algoritmo é identificar as áreas com grandes concentrações de bordas na imagem segmentada.

Todavia esse aumento da diferença de pixels ocasiona o efeito de contorno duplo dos riscos do quadro. Esse efeito ocorre devido, em um momento um *pixel* do risco do quadro é um *pixel* de origem e em outro é o de destino, em ambos o resultado da diferença é maior que o limite, o que ocasiona que a borda em ser detectada duas vezes.

Para minimizar esse problema, utilizam-se dois pontos de apoio. O outro ponto fica localizado ainda mais próximo do centro da imagem, o ponto de origem vai ser classificado como borda se as duas diferenças (origem e o primeiro ponto de apoio; origem e segundo ponto de apoio) for maior que o limite previamente estipulado. As figuras a seguir ilustram a aplicação dessa pequena modificação.



(a) Imagem original | (b) Segmentação considerando apenas uma diferença | (c) Segmentação considerando duas diferenças

Figura 3.11 – *Detalhe da letra "A"*

Para segmentar a imagem da maneira apresentada anteriormente é necessário cálculo do vetor que aponta para o centro da imagem para cada ponto analisado. Uma simplificação pode ser feita, de modo que se tenha a mesma eficácia do mecanismo anterior para quadros didáticos através da fixação do vetor de distâncias em cada uma das quatro regiões de imagens conforme descrito na Figura 3.12.

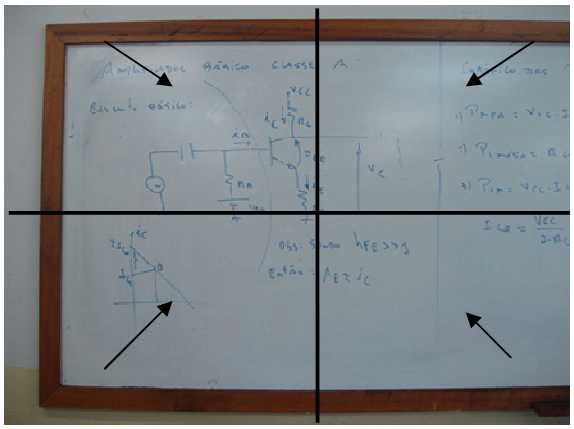


Figura 3.12 – *Divisão do quadro em quatro regiões, as setas indicam os vetores diferenças*

Os passos a seguir indicam como o algoritmo é aplicado incluindo a segmentação.

1. Dividir a imagem em quatro regiões, como apresentado na Figura 3.12.

2. Criar nova imagem binária com dimensões iguais à imagem de entrada
3. H_DIST (a componente X do vetor diferença) e V_DIST (a componente Y do vetor diferença) são definidas em função da resolução da imagem. O valor atribuído é o arredondamento inteiro de 0,91% da largura e altura, respectivamente. Este valor foi obtido empiracamente a partir do tamanho das molduras em pixels em relação ao tamanho da imagem. No caso de uma imagem com resolução de 640x480, $\text{H\_DIST} = 640 \times 0{,}91\% = 6$ e $\text{V\_DIST} = 480 \times 0{,}91\% = 4$.
4. O ponto (x,y) de DIF_IMG(x,y) não é borda se apenas uma das condições procederem (considerando que o valor de cada componente varia na escala de 0 a 255):
    a. A diferença entre todas as componentes dos pontos ORI_IMG(x,y) e ORI_IMG(x±H_DIST, y±V_DIST) é menor que 10 (DIFF_COMP)
    b. A diferença entre todas as componentes dos pontos ORI_IMG(x,y) e ORI_IMG(x±2*H_DIST, y±2*V_DIST) é menor que 10 (DIFF_COMP)

O sinal da diferença deve ser igual ao da tabela abaixo. As células correspondem às regiões dos pontos de origem analisados. O resultado da segmentação pode ser visto a seguir.

Tabela 3.1 – *Sinal usado para cada região da imagem*

X+H_DIST, Y+V_DIST	X-H_DIST, Y+V_DIST
X+H_DIST, Y-V_DIST	X-H_DIST, Y-V_DIST

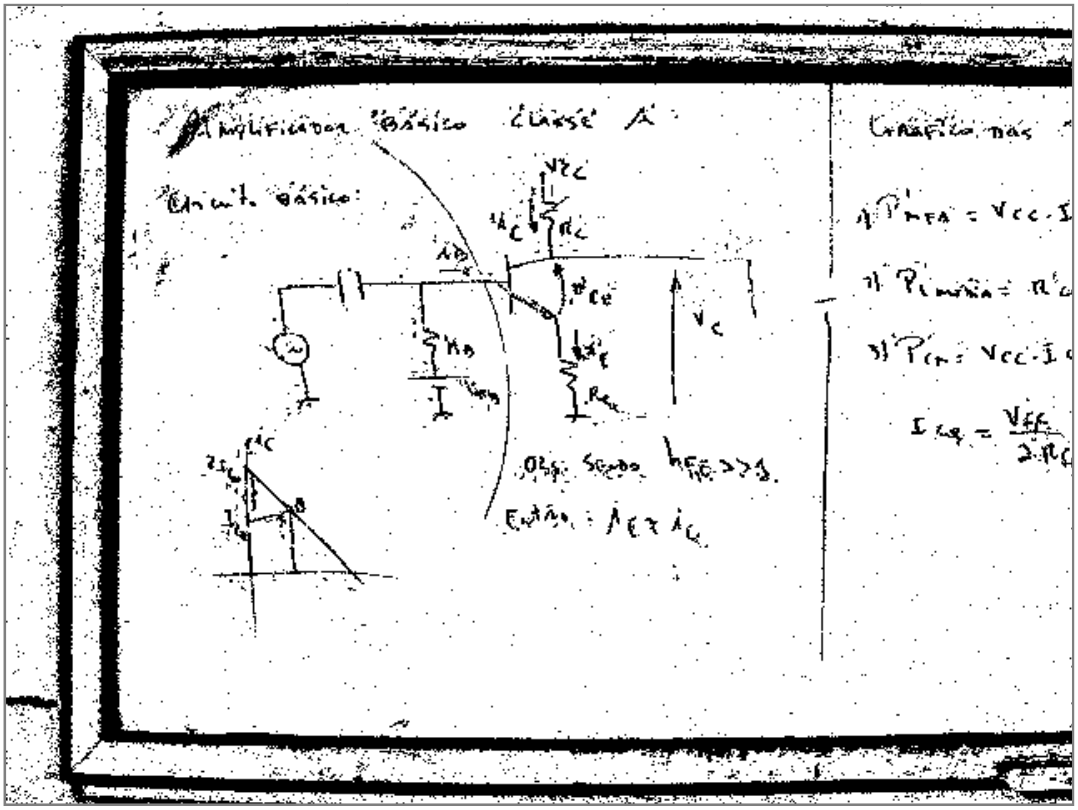


Figura 3.13 – *Imagem resultante após o primeiro passo*

3.4.2 Procurando pontos de controle

Os pontos que possivelmente pertencem à borda do quadro são chamados de "candidatos a pontos de controle". Esse passo tem o objetivo de encontrá-los através da análise da imagem que foi segmentada pelo procedimento anterior.

Para cada direção e sentido mostrado na Figura 3.14 são definidos N linhas igualmente espaçadas que buscam por pontos de controle. Essas linhas partem do centro da imagem para o exterior. Para o conjunto de imagens utilizado o valor de N é igual a nove.

O ponto inicial do rastreamento é o centro da imagem, busca-se o ponto cujos circunvizinhos sejam pretos em sua maioria. No caso da direção vertical os circunvizinhos estão compreendidos em uma máscara 7x1, na horizontal essa máscara é 1x7. Quando acontecer da maioria dos circunvizinhos ser preto (nesse caso se tiver mais de quatro pontos pretos), os passos a seguir são executados. Para simplificar, os passos a seguir denotam a busca pela borda inferior.

1. Calcular a quantidade de pontos pretos ao redor do ponto atual (x,y). O canto superior esquerdo está localizado em (A,B) e o inferior direito em (C,D), onde:

- $A = X - \left[\frac{\text{largura\_imagem}}{N} \times \frac{1}{2}\right]$ 3.6
- $B = Y - \text{altura\_imagem} \times \text{checagem\_interna}$ 3.7
- $C = X + \left[\frac{\text{largura\_imagem}}{N} \times \frac{1}{2}\right]$ 3.8
- $D = Y + \text{altura\_imagem} \times \text{checagem\_externa}$ 3.9

* Onde $checagem\_interna = 0{,}165\%$ e $checagem\_externa = 0{,}91\%$

2. Se no retângulo existirem mais de 65% (PERC_BORDA) pontos classificados como bordas, então (x,y) será marcado como ponto de controle candidato da borda inferior. Senão, o algoritmo continua o rastreamento através da procura da máscara de 7x1 pixels por outro ponto de controle.

Nos outros sentidos, o algoritmo funciona de maneira análoga, um exemplo de pontos de controle achado pode ser visto na figura abaixo em ciano e verde.

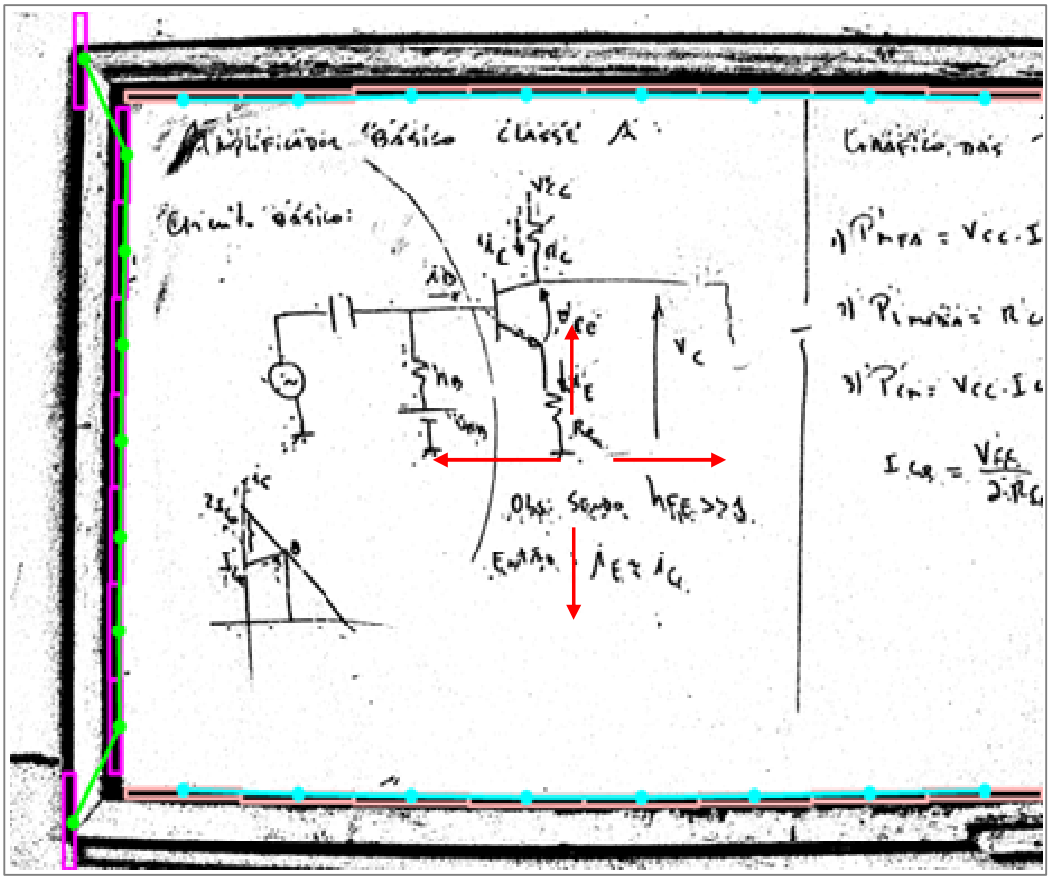


Figura 3.14 – *Direção e sentido do rastreamento das linhas da imagem (seta vermelha), os pontos de controle (pontos verde e ciano) e retângulo de análise (rosa)*

3.4.3 Eliminação de pontos de controle

Os pontos de controle podem delinear a borda do quadro negro equivocadamente devida:

- O ponto pode estar localizado fora da borda do quadro, conforme mostrado na Figura 3.14
- Quando o quadro tem "ruídos", tais como avisos ou propaganda
- A figura foi tirada com flash

Para eliminar esses pontos o seguinte procedimento é executado:

1. Para cada ponto de controle candidato, são calculadas as tangentes dos ângulos formados pelo candidato com os vizinhos de primeiro e segundo grau nos dois lados. O candidato é marcado como ponto de controle se o valor absoluto de pelo menos duas tangentes for menor ou igual a 0,07. Nota-se que esse cálculo não é relativo ao número da vizinhança, então se o candidato tiver apenas dois vizinhos todas tangentes devem ser menor ou igual a 0,07. Um exemplo de uma borda horizontal pode ser visto na Figura 3.15, o ponto candidato é cinza escuro.

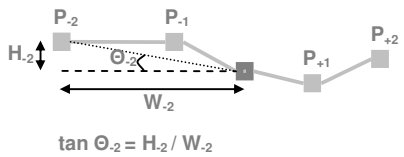


Figura 3.15 – *Seleção de ponto de controle*

2. Depois da execução do passo em todas as direções, os segmentos de reta serão definidos como na figura 3.15. Qualquer candidato fora desse segmento vai ser excluído conforme mostrado nos pontos em vermelho.

A Figura 3.15 e a Figura 3.16 mostram os pontos de controle candidatos, já a Figura 3.17 os pontos de controle eliminados.

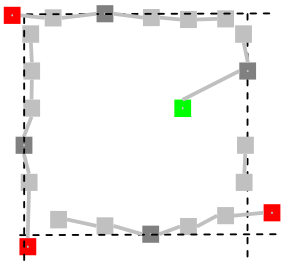


Figura 3.16 – *Ponto de controle verde é eliminado do passo 1, pontos vermelhos no passo 2*

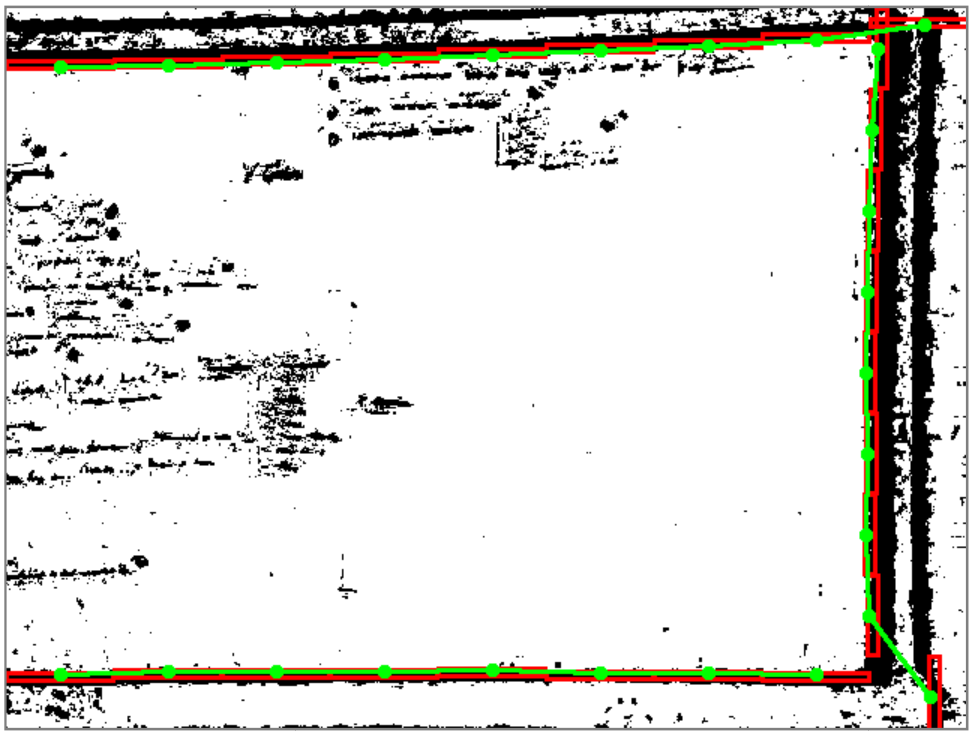


Figura 3.17 – *Imagem do quadro mostrando dois pontos de controle errados que são deletados*

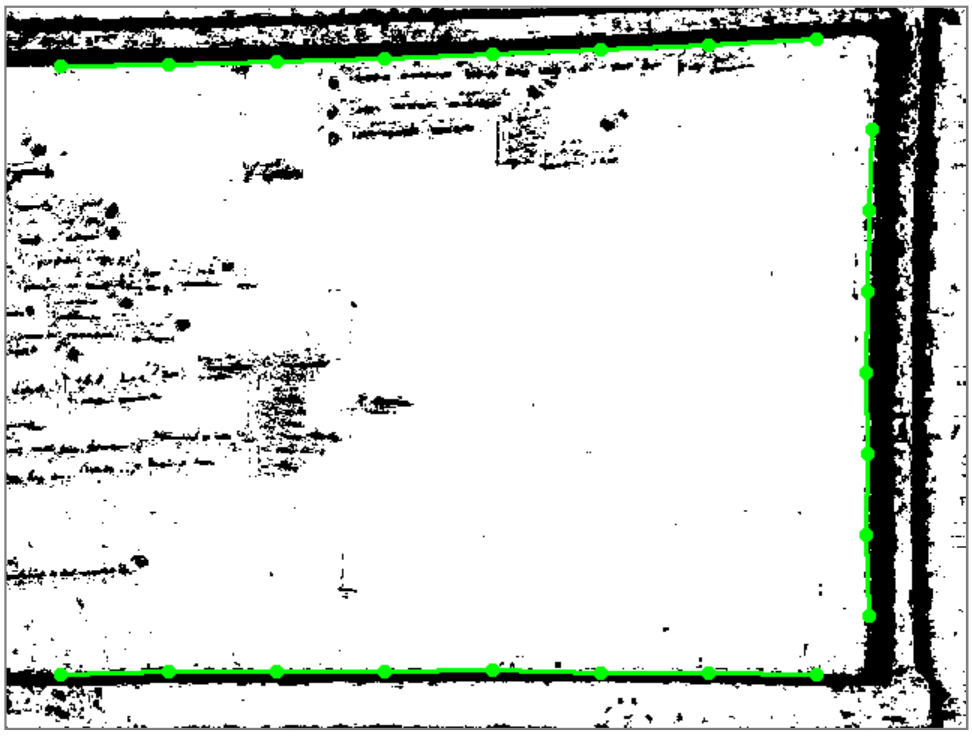


Figura 3.18 – *Imagem do quadro após a eliminação dos pontos de controle*

3.4.4 Correção de perspectiva e corte

Para corrigir a perspectiva é necessário ter as coordenadas dos quatros cantos do quadro. Para obter esses pontos traça-se uma reta entre os dois pontos da extremidade de cada borda detectada. A interseção entre essas retas é que define o canto do quadro. Caso não seja detectada nenhuma borda, a interseção ocorre com a borda da imagem. A Figura 3.19 ilustra a definição desses pontos, e a Figura 3.20 mostra a borda detectada da Figura 3.12. Esses pontos são as entradas para os algoritmos de correção de perspectiva descritos no capítulo seguinte.

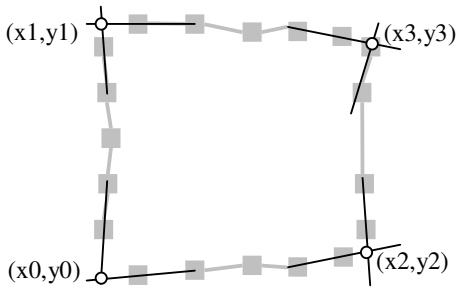


Figura 3.19 – *Quatro pontos dos cantos do quadro que servem de entrada para o algoritmo de correção de perspectiva obtidos através da interseção dos dois pontos da extremidade*

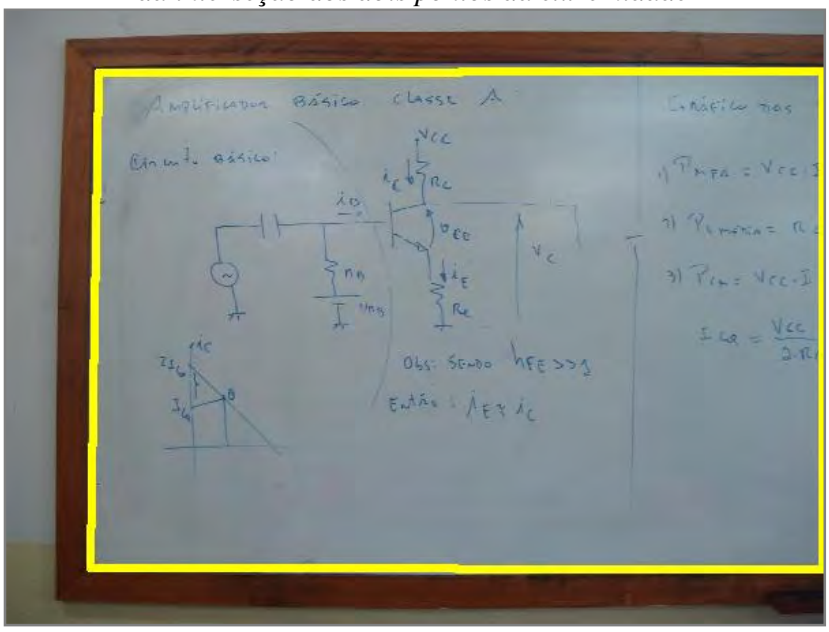


Figura 3.20 – *Borda detectada da Figura 3.12*

3.4.5 Parâmetros do algoritmo

Esta seção tem como objetivo listar todos os parâmetros definidos pelo algoritmo proposto bem como a consequência da sua modificação conforme descrito na Tabela 3.2 para o primeiro passo, na Tabela 3.3 para o segundo passo e Tabela 3.4 para o terceiro passo.

Tabela 3.2 – *Parâmetros definidos no primeiro passo da primeira proposta de detecção de borda*

Parâmetro	Valor sugerido	Observações
H_DIST	0,91% da largura	Componente horizontal do vetor diferença. O aumento deste valor torna a borda vertical mais espessa na imagem segmentada.
V_DIST	0,91% da altura	Componente vertical do vetor diferença. O aumento deste valor torna a borda horizontal mais espessa na imagem segmentada.
DIFF_COMP	10	Indica o valor limite em cada componente para indicar se um ponto na imagem pertence a borda ou não. O quanto menor for este valor, mais fácil o ponto ser detectado como borda.

Tabela 3.3 – *Parâmetros definidos no segundo passo da primeira proposta de detecção de borda*

Parâmetro	Valor sugerido	Observações
N	9	Indica a quantidade de linhas partindo do centro até periferia da imagem. Ao aumentar este valor, aumenta a quantidade de pontos de controle candidatos, porém diminui-se o tamanho da caixa ao redor de cada linha, podendo aumentar a quantidade de pontos de controle falsos.

Parâmetro	Valor sugerido	Observações
checagem_interna	0,165%	Indica o percentual relativo a altura (para bordas horizontais) ou largura (para bordas verticais) da imagem que vai indicar a altura da caixa acima do ponto central encontrado na imagem segmentada. O objetivo é evitar que o ponto de controle informado esteja localizado na moldura do quadro.
checagem_externa	0,910%	Indica o percentual relativo a altura (para bordas horizontais) ou largura (para bordas verticais) da imagem que vai indicar a altura da caixa abaixo do ponto central encontrado na imagem segmentada. Esta medida indica a área pertecente à moldura ou objetos de quadro. O ideal é que esta medida seja menor ou igual que o percentual usado no cálculo de V_DIST e H_DIST, uma vez a parede ao redor do quadro pode ser da mesma cor do fundo do quadro.
PERC_BORDA	65%	Indica a quantidade mínima de *pixels* classificados como borda dentro da caixa.

Tabela 3.4 – *Parâmetros definidos no terceiro passo da primeira proposta de detecção de borda*

Parâmetro	Valor sugerido	Observações
TAN_PC	0,07	Tangente máxima permitida entre um ponto e seus vizinhos de primeira e segunda ordem.

3.5 Um Outro Algoritmo para Detecção de Borda de Quadros

Esta seção descreve outra proposta de algoritmo para a detecção de borda publicada na referência (OLIVEIRA and LINS, 2008).

Esta proposta compartilha alguns aspectos do anterior, porém evita a criação da imagem binária, o que o torna mais eficiente em espaço e tempo. Duas listas de blocos são criadas: uma que guarda o *ranking* de bordas verticais e outra de bordas horizontais. O número total de blocos é fixado em um valor muito menor que a dimensão da imagem. Portanto, o custo espacial da nova abordagem é formado pela soma das dimensões da imagem multiplicado pelo tamanho do bloco. No algoritmo anterior o custo de memória era a imagem segmentada. O novo algoritmo constitui apenas de alterações nos dois primeiros passos do algoritmo anterior para a detecção de borda. Os passos seguintes são idênticos, portanto serão omitidos.

3.5.1 Primeira parte do algoritmo

O primeiro passo do novo algoritmo é:

1. Dividir a largura da imagem em N blocos de tamanho igual. Faz-se o mesmo para a altura da imagem. Neste estudo N é igual 9.
2. Criam-se dois novos *arrays*, um de tamanho (N, largura) e outro (N, altura). Esses novos *arrays* são chamados de V_BORDAS e H_BORDAS, respectivamente.
3. H_DIST e V_DIST são definidos da mesma forma que no passo 3 do algoritmo proposto anterior.
4. Se alguma das condições for verdadeira, incrementa-se o elemento do *array* H_BORDAS[grupoX][y] (considerando que o valor de cada componente varia na escala de 0 a 255):
    a. A diferença entre as respectivas componentes dos pontos ORI_IMG(x,y) e ORI_IMG(x,y±V_DIST) é menor que 8 (DIFF_COMP1).
    b. A diferença entre as respectivas componentes dos pontos ORI_IMG(x,y) e ORI_IMG(x,y±2*V_DIST) é menor que 16 (DIFF_COMP2).
5. Analogamente, se uma das condições seguintes for verdadeira, incremente o elemento do array V_BORDAS[grupoY][X] (considerando que o valor de cada componente varia na escala de 0 a 255).
    a. A diferença entre as respectivas componentes dos pontos ORI_IMG(x,y) e ORI_IMG(x±H_DIST,y) é menor que 8 (DIFF_COMP1).
    b. A diferença entre as respectivas componentes dos pontos ORI_IMG(x,y) e ORI_IMG(x±2*H_DIST,y) é menor 16 (DIFF_COMP2).

É necessário ressaltar que a nova abordagem faz uma diferença separada entre bordas verticais e horizontais, o que torna mais imune a ruídos. A diferença entre limites entre as condições “a” e “b” é maior, pois os pontos em “b” estão mais distantes que em “a”, conseqüentemente, a diferença de iluminação tende a ser maior.

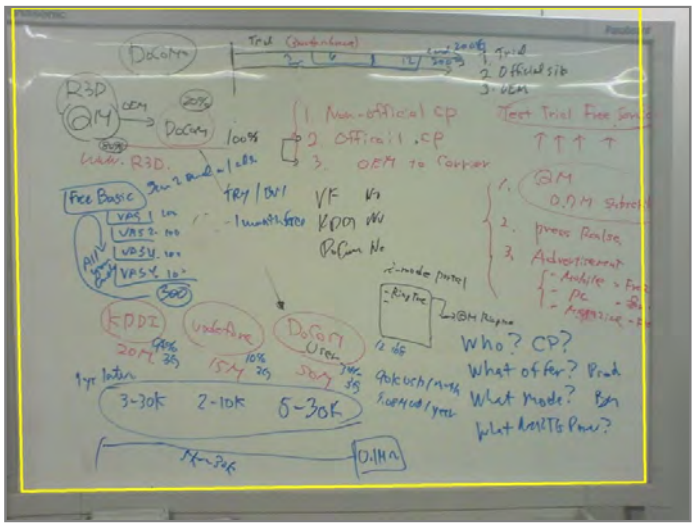
3.5.2 Buscando pontos de controle na imagem (segunda parte)

Depois de executar os passos acima em toda imagem, utiliza-se apenas estrutura de *cluster* montada anteriormente para obter o mesmo resultado. O rastreamento é feito da mesma forma, do centro para fora da imagem. Os passos da segunda parte do algoritmo para encontrar a borda inferior do quadro podem ser vistos abaixo, as outras bordas podem ser achadas de maneira análoga:

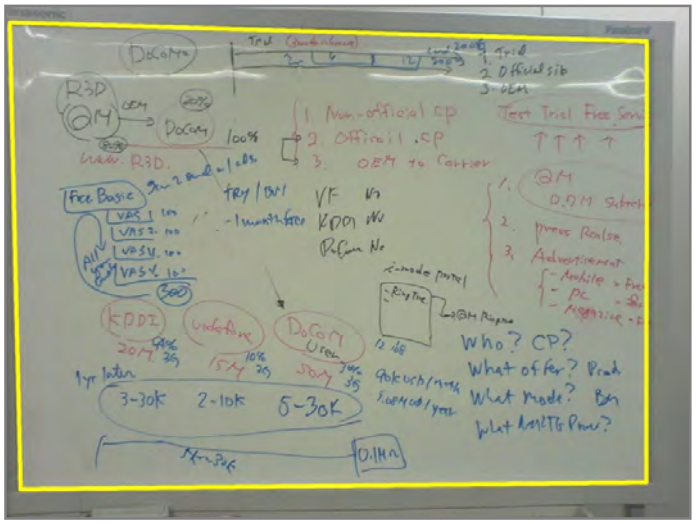
1. Para cada cluster X, atribuí-se y no centro da imagem.
2. Calcula-se a quantidade de pixels que são “possíveis bordas”, adicionando-se elementos do *array* entre os elementos H_BORDAS[grupoX][y+V_DIST/2] e H_BORDAS[grupoX][y]
3. Observa-se que V_DIST é a distância entre (x,y) e a primeira diferença vertical, e ao somar na área do cluster na coordenada y entre y e y+V_DIST/2 – que é a metade de toda área da borda – aumenta-se o limite da densidade de 65%, do algoritmo anterior, para 85% (PERC_BORDA) que torna o resultado mais robusto e menos sensível a ruído.

Uma vez que os pontos de controle forem encontrados, o terceiro passo do algoritmo anterior é executado para a definição dos cantos do quadro. É necessário mencionar que ao aumentar o valor do número de blocos N aumenta-se a sensibilidade rotacional, no entanto a quantidade de pontos de controle achados equivocadamente pode aumentar.

Um exemplo de detecção de borda pode ser visto na Figura 3.21 (b), em contrapartida com a primeira proposta que pode ser observada na Figura 3.21 (a).



(a) Detecção de borda usando o algoritmo (OLIVEIRA e LINS, 2007)



(b) Detecção de borda usando o algoritmo (OLIVEIRA e LINS, 2008)

Figura 3.21 – *Borda detectada com as duas novas propostas*

3.5.3 Parâmetros do algoritmo

Esta seção tem como objetivo listar todos os parâmetros definidos pelo algoritmo proposto bem como a consequência da sua modificação conforme descrito na Tabela 3.5 para o primeiro passo, na Tabela 3.6 para o segundo passo, como o terceiro passo é igual à primeira proposta, os detalhes dos parâmetros pode ser visto na Tabela 3.4.

Tabela 3.5 – *Parâmetros definidos no primeiro passo da segunda proposta de detecção de borda*

Parâmetro	Valor sugerido	Observações
N	9	Indica a quantidade de setores em que a imagem foi dividida verticalmente (para calcular bordas horizontais) e horizontalmente (para calcular bordas verticais). Ao incrementar este valor, aumenta-se a quantidade de candidatos a pontos de controle, podendo também aumentar a quantidade de falsos pontos de controle.
H_DIST	0,91% da largura	Magnitude do vetor horizontal utilizado na detecção de bordas verticais. O aumento deste valor torna a borda vertical mais espessa na estrutura V_BORDAS.
V_DIST	0,91% da altura	Magnitude do vetor vertical utilizado na detecção de bordas horizontais. O aumento deste valor torna a borda horizontal mais espessa na estrutura V_BORDAS.
DIFF_COMP1	8	Indica o valor limite em cada componente para a primeira diferença. O quanto menor for este valor, mais fácil o ponto ser detectado como borda.
DIFF_COMP2	16	Indica o valor limite em cada componente para a segunda diferença. O quanto menor for este valor, mais fácil o ponto ser detectado como borda, ele deve ser maior ou igual a primeira diferença devido ao fato da iluminação variar mais.

Tabela 3.6 – *Parâmetros definidos no segundo passo da segunda proposta de detecção de borda*

Parâmetro	Valor sugerido	Observações
PERC_BORDA	85%	Indica a quantidade mínima de *pixels* classificados como borda para classificar o ponto de controle.

3.6 Resumo

Este capítulo teve o intuito de mostrar os algoritmos de detecção de bordas. Para realizar tal objetivo foi feita uma revisão bibliográfica sobre os algoritmos clássicos de detecção de borda e a transformada de Hough, utilizada para identificação de linhas retas na imagem. Além de mostrar a detecção de borda de quadros do WhiteboardIt, bem como suas falhas de especificações.

Foram apresentados também dois novos mecanismos de detecção de borda de quadros propostos, ambos publicados em conferências internacionais.

4 Correção de perspectiva de quadros didáticos

A captura de fotos a partir de câmeras digitais portáteis, devido à ausência de suporte mecânico, gera fotos com a perspectiva "distorcida". A foto de um quadro idealmente oferece uma visão frontal do mesmo obtida em plano paralelo ao quadro na reta perpendicular à superfície traçada a partir da intersecção das mediatrizes do quadro, conforme mostrado na Figura 4.1. O presente capítulo trata como retirar a distorção de perspectiva da imagem.

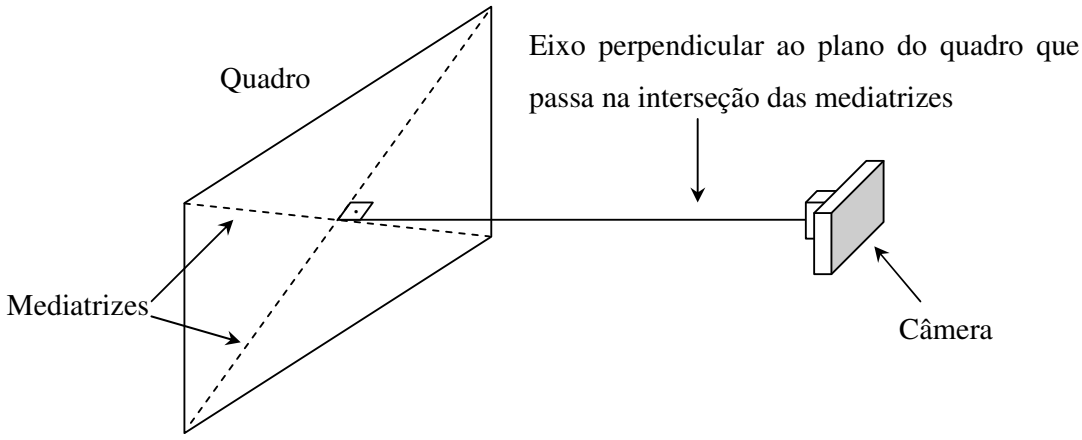


Figura 4.1 – *Câmera na posição frontal do quadro considerada como ideal*

Existem vários algoritmos na literatura (YIN *et al*, 2007; MASALOVITCH and MESTETSKIY, 2007; UCHIDA *et al*, 2007) empregados na correção de distorções (além da perspectiva) de documentos impressos. Esses algoritmos tiram proveito das estruturas de texto conhecidas como: distribuição de palavras numa mesma altura, as linhas igualmente espaçadas etc.

No caso dos quadros didáticos a única coisa em comum é que o quadro é retangular, só que na imagem é um quadrilátero convexo devido à distorção de perspectiva. Uma vez definidos os quatro pontos localizados nas quinas do quadro, busca-se uma transformada linear sobre a imagem original. É possível inferir tal transformada, pois se deseja obter uma imagem de saída em que a borda do quadro vai ser caracterizada por um retângulo sem inclinação.

4.1 Conceitos básicos

Uma câmera fotográfica pode ser modelada como se fosse uma câmera escura com um furo minúsculo onde a imagem é projetada do lado oposto ao furo, esse modelo é conhecido como *pinhole* (GONZALEZ and WOODS, 2008), que foi brevemente introduzido no segundo capítulo desta dissertação. Uma visualização geométrica deste modelo pode ser vista na Figura 4.2.

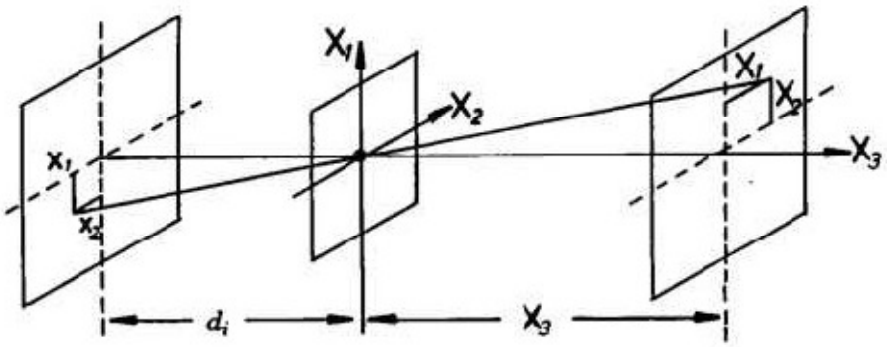


Figura 4.2 – *Formação da imagem com uma câmera pinhole (SILVA, 2006)*

A imagem pode ser modelada através de uma transformação de objetos que estão em três dimensões (mundo real) para um plano com duas dimensões (plano da imagem), ocasionando a perda da coordenada de profundidade. Ao observar a Figura 4.2 é possível estabelecer as relações entre os objetos em três dimensões e a imagem, conforme na Equação 4.1 (SILVA, 2006).

$$(X_1, X_2, X_3) \rightarrow (x_1, x_2) = (\frac{d_i X_1}{X_3}, \frac{d_i X_2}{X_3}) \quad 4.1$$

Ao utilizar coordenadas generalizadas, divide-se a transformação acima por d_i, que é a distância do furo para a imagem projetada, obtendo a transformação:

$$(X_1, X_2, X_3) \rightarrow (x_1, x_2) = (\frac{X_1}{X_3}, \frac{X_2}{X_3}) \quad 4.2$$

4.1.1 Coordenadas homogêneas

As matrizes de coordenadas generalizadas não permitem representar as operações de translação e projeção de perspectiva com as operações de rotação e mudança de escala na mesma matriz. Para isso, utilizam-se matrizes de coordenadas homogêneas. As

transformações elementares presentes na formação da imagem estão ilustradas na Tabela 4.1.

Tabela 4.1 – *Matrizes de transformações na formação da imagem*

$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -T_1 & -T_2 & -T_3 & 1 \end{bmatrix}$ Translação	$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ Rotação no eixo x
$R_y = \begin{bmatrix} \cos\varphi & 0 & \sin\varphi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\varphi & 0 & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ Rotação no eixo y	$R_z = \begin{bmatrix} \cos\beta & \sin\beta & 0 & 0 \\ -\sin\beta & \cos\beta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ Rotação no eixo z
$E = \begin{bmatrix} s_1 & 0 & 0 & 0 \\ 0 & s_2 & 0 & 0 \\ 0 & 0 & s_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ Mudança de escala	$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1/d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$ Projeção de perspectiva

A formação da imagem no sensor da câmera pode ser entendida através da aplicação de cada uma das transformações acima, seqüencialmente. Como a multiplicação de matrizes é associativa, pode-se resumir todo o processo com uma única matriz. Além das matrizes mostradas na Tabela 4.1, é feito uma transformação de corte (descrito pela Equação 4.2), para que a matriz resultante tenha tamanho 4x3, pois o ponto da imagem possui duas dimensões na notação de coordenadas homogêneas. A matriz resultante M de tamanho 4x3 é obtida conforme a Equação 4.3.

$$M = TR_xR_yR_zPEC \tag{4.3}$$

Portanto, a relação entre a imagem projetada no sensor da câmera e o quadro fotografado é denotada por (JAGANNATHAN and JAWAHAR, 2005; HARTLEY and ZISSERMAN, 2003)

$$p = M\mathrm{P} \tag{4.4}$$

, onde p indica o ponto na imagem (tamanho 3x1), P o ponto do quadro no mundo (tamanho 4x1). Tanto a imagem do quadro como o quadro em si são planos, é possível simplificar a relação em

$$q_i' = Hq_i \tag{4.5}$$

onde q_i' e q_i representam vetores 3x1 do mesmo ponto em visões diferentes. Dado quatro pontos correspondentes é possível formar um sistema com oito equações com oito variáveis, logo o sistema pode-se determinar a matriz H. Seja:

$$q_i' = [x_i' \quad y_i' \quad 1]^T \tag{4.6}$$

$$q_i = [x_i \quad y_i \quad 1]^T \tag{4.7}$$

a relação homográfica pode ser traduzida em:

$$x_i' = \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}y_i + h_{32}y_i + h_{33}} \tag{4.8}$$

$$y_i' = \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}y_i + h_{32}y_i + h_{33}} \tag{4.9}$$

Esses quatro pontos da imagem foram obtidos com um dos algoritmos do capítulo anterior. Os pontos correspondentes na imagem final devem ser definidos para que seja possível calcular os elementos da matriz H (HARTLEY and ZISSERMAN, 2003).

O principal fator a ser analisado nos pontos de saída é a manutenção da proporção real do quadro. A outra questão é relativa ao tamanho da imagem, como está se trabalhando com imagens digitais, é desejável que o mapeamento de um ponto da imagem de entrada corresponda a, pelo menos, um ponto da de saída, caso contrário há perda de informação.

Caso a proporção seja conhecida, é trivial definir-se os pontos de saída com o tamanho desejado. Nos outros casos é necessário levar em consideração a informação a priori da forma do quadro ser um retângulo como será visto nos três métodos que serão apresentados a seguir.

4.2 Métodos de interpolação

No mapeamento das transformações geométricas o ponto na imagem transformada não indica, necessariamente, um ponto com coordenadas inteiras na imagem original. Portanto é necessário estimar o valor desses pontos intermediários de modo que a imagem transformada seja o mais natural possível.

A maneira mais simples – e mais rápida – de se estimar o ponto na imagem original é utilizar o ponto vizinho mais próximo. Já os métodos de interpolações fazem o uso de mais de um ponto na imagem original para estimar o valor do ponto desejado. Em todos os métodos é comum aparecimento de distorções indesejáveis nas imagens, como pode ser visto na Figura 4.3 abaixo ao aumentar uma imagem em 400%.

(a) Imagem original

(b) *aliasing* (c) embassamento (d) borda *halo*

Figura 4.3 – *Efeitos de distorções comuns na utilização de métodos de interpolações (CAMBRIDGEINCOLOUR, 2008a)*

Nesta seção serão discutidos dois métodos mais usados na literatura: bilinear e bicúbica. Esses métodos são simples e suficiente para o objetivo deste capítulo, que é a correção de perspectiva. Ambos foram baseados nas referências (BURGER and BURGE, 2008) e (GONZALEZ and WOODS, 2008). Existem outros mais sofisticados na literatura e em sistema proprietários para ampliação de imagens para impressão que podem ser vistos em (CAMBRIDGEINCOLOUR, 2008a) e (CAMBRIDGEINCOLOUR, 2008b).

4.2.1 Interpolação bilinear

Este método utiliza os quatro pontos da imagem do quadrado do pixel em que o ponto desejado está dentro, conforme ilustrado na Figura 4.4. Na direção y obtêm-se dois valores da intensidade através da média ponderada pela distância dos pontos nessa direção que compartilham o valor de y. A Figura 4.4 mostra esses pontos E e F, onde E é calculado a partir de A e B, e F a partir de C e D. Com os pontos E e F, uma nova média ponderada pela distância na direção x é calculada, obtendo o ponto G, que representa a intensidade final do valor desejado. Vale ressaltar que não há diferença no valor final caso o cálculo seja iniciado pela direção x, sendo facilmente provado.

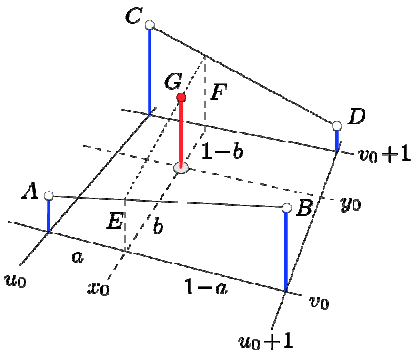


Figura 4.4 – *Cálculo utilizando interpolação bilinear (BURGER and BURGE, 2008)*

4.2.2 Interpolação bicúbica

A interpolação bicúbica utiliza uma janela maior que a bilinear em uma grade 4x4 totalizando 16 pontos. A distribuição desses pontos é feita conforme a Figura 4.5 onde (x_0, y_0) é o ponto que se deseja estimar na imagem, e u_0 é o piso de x_0, e o v_0 é o piso de y_0, lembrando que na Figura 4.5, o sentido do eixo x é da esquerda pra direita e do eixo y de cima pra baixo.

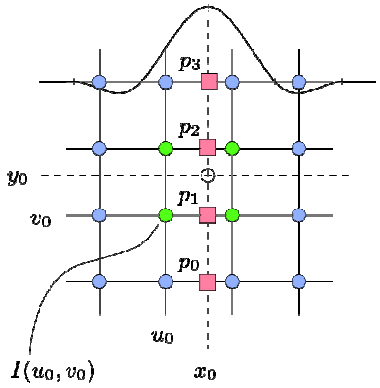


Figura 4.5 – *Distribuição dos pontos na interpolação bicúbica (BURGER and BURGE, 2008)*

O valor de $I'(x_0, y_0)$ é obtido através de uma soma ponderada conforme a expressão

$$I'(x_0, y_0) = \sum_{u=u_0-1}^{u_0+2} \sum_{v=v_0-1}^{v_0+2} I(u, v)\, P_{bic}(x_0 - u, y_0 - v) \qquad 4.10$$

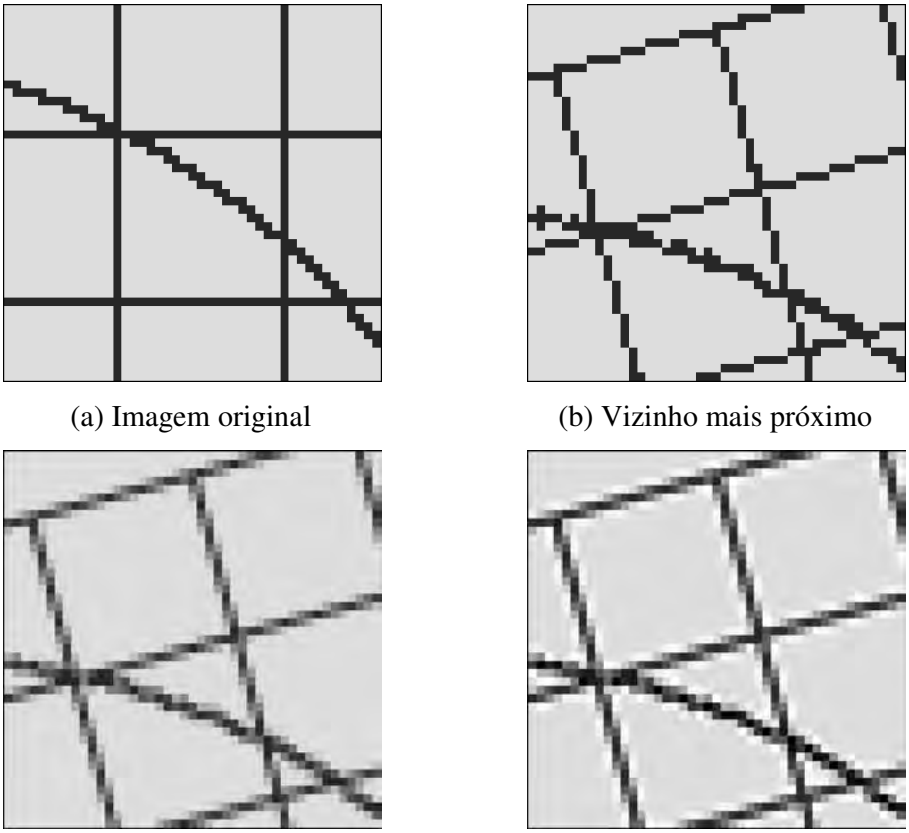
onde $I(u, v)$ indica o valor do ponto (u, v) na imagem original, com domínio em $\mathbb{N} \times \mathbb{N}$. E P_{bic} é o peso a ser utilizado nesse ponto, sendo definido por

$$P_{bic}(x, y) = P_{cub}(x) P_{cub}(y) \qquad 4.11$$

, e P_{cub} é definido por

$$P_{cub}(x,y) = \begin{cases} a \times x^3 + 5a \times x^2 + 8a \times x + 4a & se -2 \leq x < 1 \\ (a-2) \times x^3 + (1-a) \times x^2 + 1 & se -1 \leq x < 0 \\ (2-a) \times x^3 + (1-a) \times x^2 + 1 & se\ 0 \leq x < 1 \\ -a \times x^3 + 5a \times x^2 - 8a \times x + 4a & se\ 1 \leq x \leq 2 \\ 0 & senão \end{cases} \quad 4.12$$

. A Figura 4.6 ilustra a aplicação de duas interpolações e o método do ponto mais próximo na rotação de uma imagem sintética, e notável o aparecimento de distorção indesejável perto nas bordas da imagem como informado na Figura 4.3.



(a) Imagem original (b) Vizinho mais próximo

(c) Interpolação bilinear (d) Interpolação bicúbica

Figura 4.6 – *Resultado da aplicação dos métodos de estimativa de pontos em uma imagem sintética (BURGER and BURGE, 2008)*

4.3 Método de Jagannathan e Jawahar

O primeiro método foi proposto por Jagannathan e Jawahar (JAGANNATHAN and JAWAHAR, 2005). Ele consiste em separar a matriz H no produto de três componentes: similaridade (H_s), afinidade (H_a) e projetiva (H_p). É desejado remover as componentes projetiva e afinidade, para que ter uma figura similar à imagem frontal.

Esse método se baseia no fato que existem pelo menos duas linhas paralelas entre si (duas horizontais, duas verticais) e duas linhas perpendiculares (que se intersectam nas quinas), ambas no plano do quadro.

As linhas paralelas não são mais paralelas no plano da imagem de entrada. Para que essas linhas voltem a ser paralelas deve-se retirar a componente projetiva ao observar a linha que passa na interseção no plano da imagem, que é definido por

$$L = [l_1 \quad l_2 \quad l_3] \tag{4.13}$$

A componente projetiva para o processo que transforma a imagem em frontal pode ser descrita da seguinte forma

$$H_p = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix} \tag{4.14}$$

A intersecção dos pontos das retas paralelas não existe na imagem frontal, portanto pode-se considerar que as retas se intersectam no infinito e sua reta pode ser definida como:

$$L_\infty = [0 \quad 0 \quad 1]. \tag{4.15}$$

O próximo passo é remover a componente de afinidade.

Sejam duas linhas que sua interseção forma uma das quinas dos quadros. Na imagem frontal elas formam um ângulo reto, para achar a transformada H_a utiliza-se o método de cônica absoluta descrita em (HARTLEY and ZISSERMAN, 2003). Uma vez achada a transformada retira-se a componente utilizando sua inversa.

Apesar da idéia desse método ser interessante, ele não especifica claramente a definição da reta L, que mais parece a definição do ponto de interseção do que da reta em si. Além de não definir melhor como achar a componente de afinidade H_a. Logo não foi possível desenvolver o algoritmo para comparar sua eficácia diante os próximos que serão apresentados a seguir.

4.4 Método do WhiteboardIt

A correção perspectiva do WhiteboardIt faz uso dos quatro pontos para calcular a matriz de transformação. A transformação sugerida usa desses pontos para estimar a razão largura/altura do quadro didático. Esse método pode ser visto com mais detalhes em (LIU *et al*, 2007; ZHANG and TAN, 2001; ZHANG and HE, 2004; ZHANG *et al*, 2006).

Inicialmente modela-se a imagem conforme a figura a seguir, onde m_i são pontos na imagem original que serão projetados nos pontos definidos por M_i com coordenada $z = 0$. Essa modelagem é um caso particular do modelo matemático de câmera escura. Os pontos m_i são definidos como: $m_1 = (0,0),\ m_2 = (l, 0),\ m_3 = (0, h)$ e $m_4 = (l, h)$, sendo l e h, a largura e altura da imagem final, respectivamente. Convenientemente, define-se que $\tilde{x}$ é igual a $(x_1, x_2, .., x_n, 1)$ onde $x = (x_1, x_2, .., x_n)$.

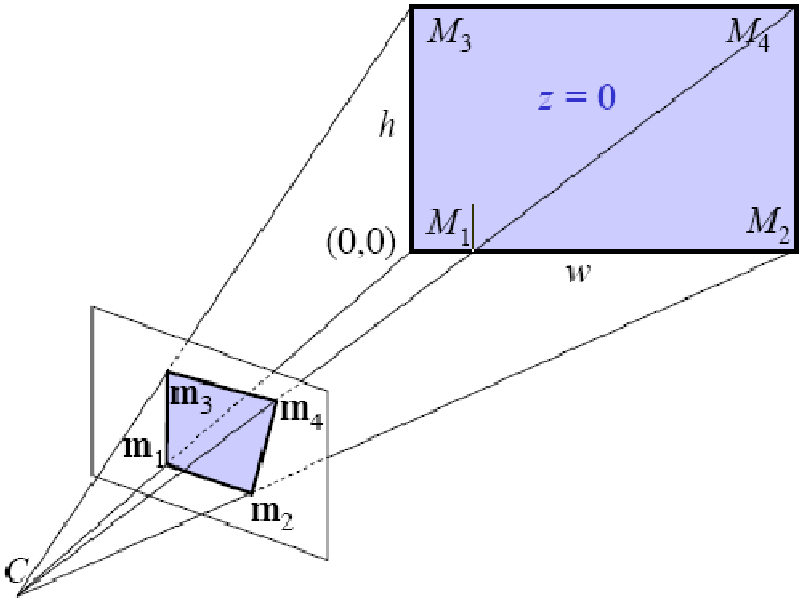


Figura 4.7 – *Geometria do retângulo (ZHANG and HE, 2004)*

Nessa nova modelagem de câmera as relações entre M e m são definidas da seguinte maneira:

$$\lambda \tilde{m} = A[R \quad t]\tilde{M} \tag{4.16}$$

Onde:

$$A = \begin{bmatrix} f & 0 & u_0 \\ 0 & s * f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.17}$$

$$R = [r_1 \quad r_2 \quad r_3] \tag{4.18}$$

O valor de f é a distância focal. O valor de s é a razão largura/altura de um *pixel*, como o *pixel* é quadrado o valor é igual a 1. Já o valor de u_0 e v_0 é a coordenada (x, y) do centro da imagem original. E $[R \quad t]$ define a transformada 3D do sistema de coordenadas do mundo, que é o retângulo definido pelos pontos m_i, e o sistema de coordenadas da câmera. R indica uma matriz de rotação e t o vetor de translação.

O objetivo é achar a matriz A, R e t a partir dos valores de m. Será descrito aqui o cálculo da matriz de uma maneira diferente do artigo do WhiteboardIt, pois o artigo não segue uma seqüência em que as equações dependem exclusivamente das anteriores, dificultando o seu desenvolvimento. No entanto a nomenclatura dos termos será mantida, exceto o termo n' e n'', onde o WhiteboardIt define como n_2 e n_3, respectivamente.

Inicialmente, coloca-se as equações descrito em 4.16, para $\widetilde{M}_1 = (0,0)$

$$\lambda_1 \widetilde{m}_1 = A\ t \tag{4.19}$$

para $\widetilde{M}_2 = (l, 0)$

$$\lambda_2 \widetilde{m}_2 = lAr_1 + A\ t \tag{4.20}$$

para $\widetilde{M}_3 = (0, h)$

$$\lambda_3 \widetilde{m}_3 = hAr_2 + A\ t \tag{4.21}$$

para $\widetilde{M}_4 = (l, h)$

$$\lambda_4 \widetilde{m}_4 = lAr_1 + hAr_2 + A\ t \tag{4.22}$$

Subtrai-se a equação 4.19 em todas as outras.

$$\lambda_2 \widetilde{m}_2 - \lambda_1 \widetilde{m}_1 = lAr_1 \tag{4.23}$$

$$\lambda_3 \widetilde{m}_3 - \lambda_1 \widetilde{m}_1 = hAr_2 \tag{4.24}$$

$$\lambda_4 \widetilde{m}_4 - \lambda_1 \widetilde{m}_1 = lAr_1 + hAr_2 \tag{4.25}$$

Calcula-se ([4.25] – [4.24] – [4.23]):

$$\lambda_4 \widetilde{m}_4 + \lambda_1 \widetilde{m}_1 - \lambda_3 \widetilde{m}_3 - \lambda_2 \widetilde{m}_2 = 0 \tag{4.26}$$

Fazendo o produto vetorial de $\widetilde{m}_4$ dos dois lados da formula acima.

$$0 + \lambda_1 \widetilde{m}_1 \times \widetilde{m}_4 - \lambda_3 \widetilde{m}_3 \times \widetilde{m}_4 - \lambda_2 \widetilde{m}_2 \times \widetilde{m}_4 = 0 \tag{4.27}$$

Fazendo o produto escalar de $\widetilde{m}_3$ dos dois lados da formula acima (observa-se que $\widetilde{m}_1 \times \widetilde{m}_4$ é perpendicular ao ponto $\widetilde{m}_3$).

$$\lambda_1 (\widetilde{m}_1 \times \widetilde{m}_4) \cdot \widetilde{m}_3 - 0 - \lambda_2 (\widetilde{m}_2 \times \widetilde{m}_4) \cdot \widetilde{m}_3 = 0 \tag{4.28}$$

O que pode resultar na razão:

$$k_2 = \frac{\lambda_2}{\lambda_1} = \frac{(\widetilde{m}_1 \times \widetilde{m}_4) \cdot \widetilde{m}_3}{(\widetilde{m}_2 \times \widetilde{m}_4) \cdot \widetilde{m}_3} \tag{4.29}$$

Analogamente, fazendo o produto escalar de $\widetilde{m}_2$ em 4.27.

$$k_3 = \frac{\lambda_3}{\lambda_1} = \frac{(\widetilde{m}_1 \times \widetilde{m}_4) \cdot \widetilde{m}_2}{(\widetilde{m}_3 \times \widetilde{m}_4) \cdot \widetilde{m}_2} \tag{4.30}$$

Sejam n'e n''.

$$n' = (k_2 \widetilde{m}_2) - \widetilde{m}_1 \tag{4.31}$$

$$n'' = (k_3\widetilde{m}_3) - \widetilde{m}_1 \tag{4.32}$$

Divide-se 4.23 por l e multiplica-se a esquerda por A^{-1}, obtendo:

$$r_1 = A^{-1}\frac{1}{l}(\lambda_2\widetilde{m}_2 - \lambda_1\widetilde{m}_1) \tag{4.33}$$

Reescrevendo a equação.

$$r_1 = A^{-1}\frac{\lambda_1}{l}\left(\frac{\lambda_2}{\lambda_1}\widetilde{m}_2 - \widetilde{m}_1\right) \tag{4.34}$$

Que resulta em:

$$r_1 = A^{-1}\frac{\lambda_1}{l}(k_2\widetilde{m}_2 - \widetilde{m}_1) \tag{4.35}$$

Ou:

$$r_1 = \frac{\lambda_1}{l}A^{-1}n' \tag{4.36}$$

Do mesmo modo obtemos r_2, ao dividir 4.24 por $(\lambda_1 l)$ e multiplica-se a esquerda por A^{-1}, obtendo:

$$r_1 = \frac{\lambda_1}{l}A^{-1}n'' \tag{4.37}$$

De acordo com a propriedade de matriz de rotação o produto escalar de r_1e r_2é igual a 0, portanto:

$$\left[\frac{\lambda_1}{l}A^{-1}n'\right]^T\left[\frac{\lambda_1}{l}A^{-1}n''\right] = 0 \tag{4.38}$$

O que pode ser simplificado para:

$$[A^{-1}n']^T[A^{-1}n''] = 0 \tag{4.39}$$

$$n'^T A^{-T}A^{-1}\,n'' = 0 \tag{4.40}$$

A partir da equação 4.40 pode-se calcular o valor do foco.

$$f^2 = -\frac{\begin{pmatrix}[n'_1n''_1 - (n'_1n''_3 + n'_3n''_1)u_0 + n'_3n''_3{u_0}^2]s^2 \\ + \\ [n'_2n''_2 - (n'_2n''_3 + n'_3n''_2)v_0 + n'_3n''_3{v_0}^2]\end{pmatrix}}{n'_3n''_3s^2} \tag{4.41}$$

Daí, obtém-se a matriz A. Agora é possível calcular os elementos de R.

$$r_1 = \frac{A^{-1}n'}{\|A^{-1}n'\|} \tag{4.42}$$

$$r_2 = \frac{A^{-1}n''}{\|A^{-1}n''\|} \tag{4.43}$$

$$r_3 = r_1 \times r_2 \tag{4.44}$$

Para calcular a razão largura/altura, de acordo com a propriedade da matriz de rotação $r_1 \cdot r_1 = 1$ e $r_2 \cdot r_2 = 1$, daí podemos obter:

$$1 = \frac{n'^T A^{-T} A^{-1}\, n'}{(l\lambda_1)^2} \tag{4.45}$$

e

$$1 = \frac{n''^T A^{-T} A^{-1}\, n''}{(h\lambda_1)^2} \tag{4.46}$$

Pode-se inferir o quadrado da razão largura/altura, o que é possível manter a proporção do quadro na imagem final.

$$r^2 = \left(\frac{l}{h}\right)^2 = \frac{(n')^T A^{-T} A^{-1} n'}{(n'')^T A^{-T} A^{-1} n''} \tag{4.47}$$

Para obter a imagem final é necessário definir-se as suas dimensões de modo todo *pixel* presente na imagem original seja mapeado pelo menos uma vez na imagem transformada. Seja o quadrilátero que representa o quadro projetado com larguras l_1 e l_2; e alturas h_1 e h_2. Obtém-se

$$\hat{l} = \max(l_1, l_2) \tag{4.48}$$

$$\hat{h} = \max(h_1, h_2) \tag{4.49}$$

$$\hat{r} = \frac{\hat{l}}{\hat{h}} \tag{4.50}$$

A dimensão da nova imagem vai ser (lembrando que r^2 foi obtido na equação 4.47):

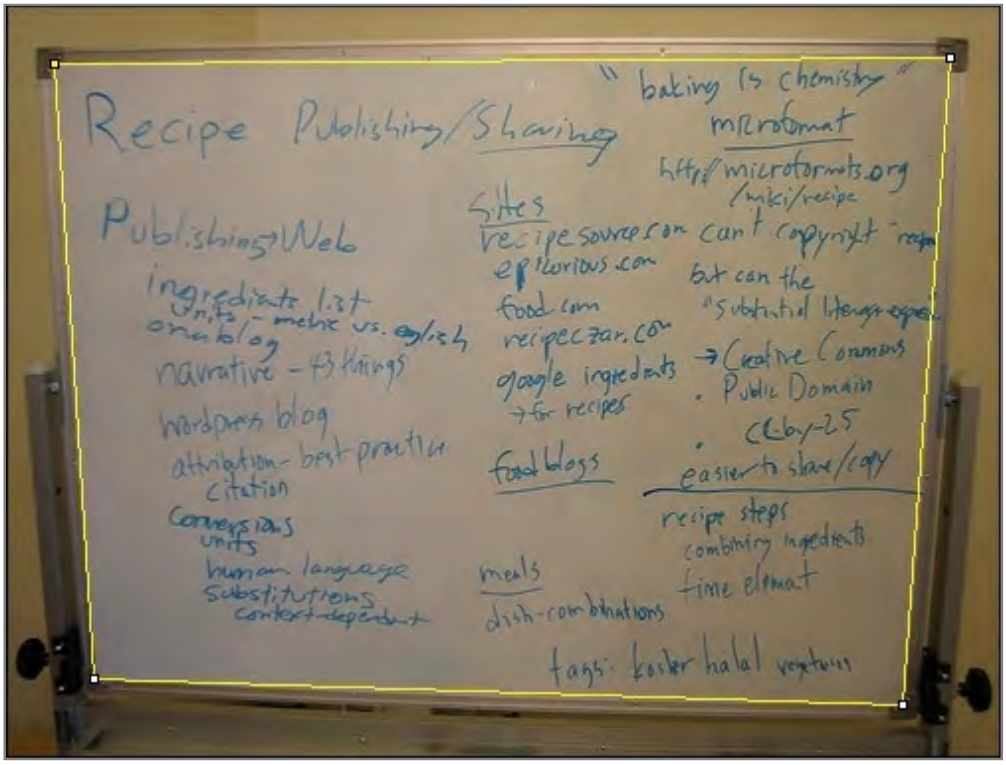
$$\begin{aligned} l = \hat{l},\ \ H = \frac{\hat{l}}{r} \ \ \text{se} \ \ \hat{r} \geq r \\ l = r * \hat{h},\ \ h = \hat{h} \ \ \text{se} \ \ \hat{r} \geq r \end{aligned} \tag{4.51}$$

Agora é possível obter o vetor de translação t.

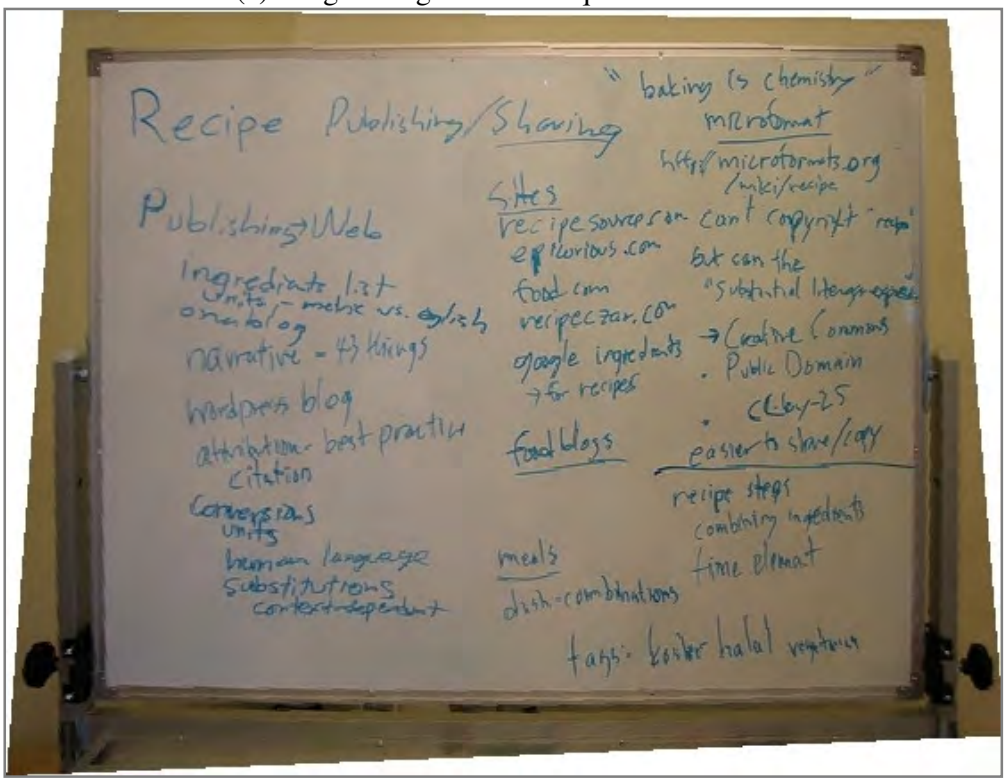
$$\lambda_1 = \frac{l}{(n')^T A^{-T} A^{-1} n'} \tag{4.52}$$

$$t = \lambda_1 A^{-1} \tilde{m}_1 \tag{4.53}$$

Agora é possível realizar a transformação. Um exemplo dessa transformação pode ser vista na abaixo utilizando vários métodos de mapeamento de *pixels*.

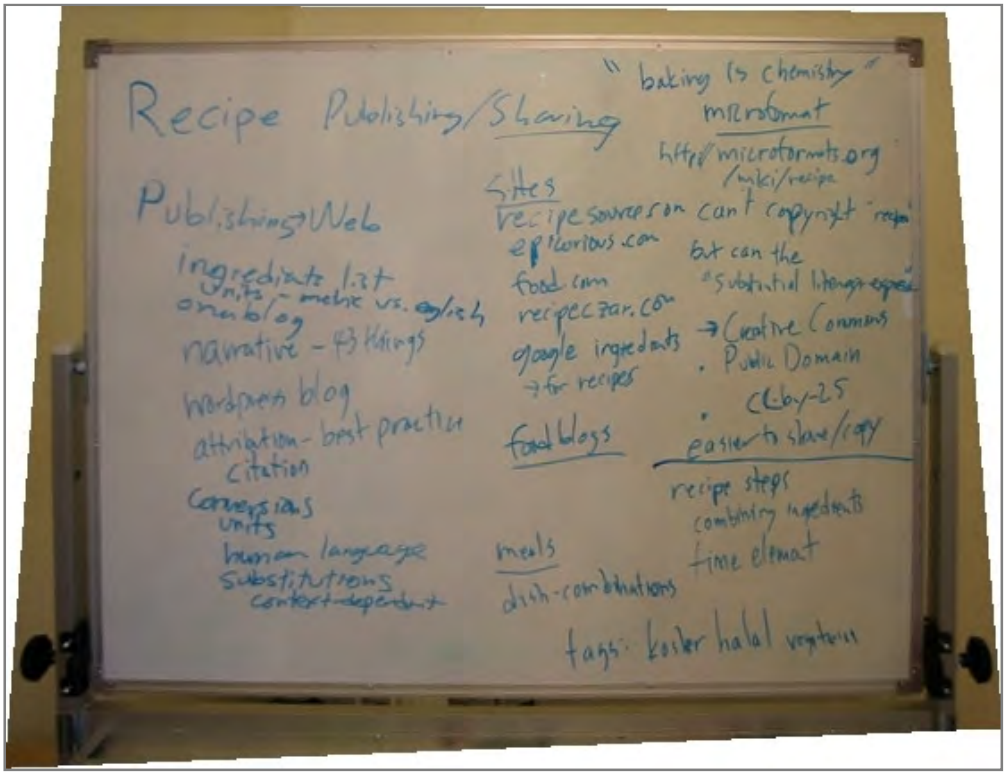


(a) Imagem original com as quinas definidas

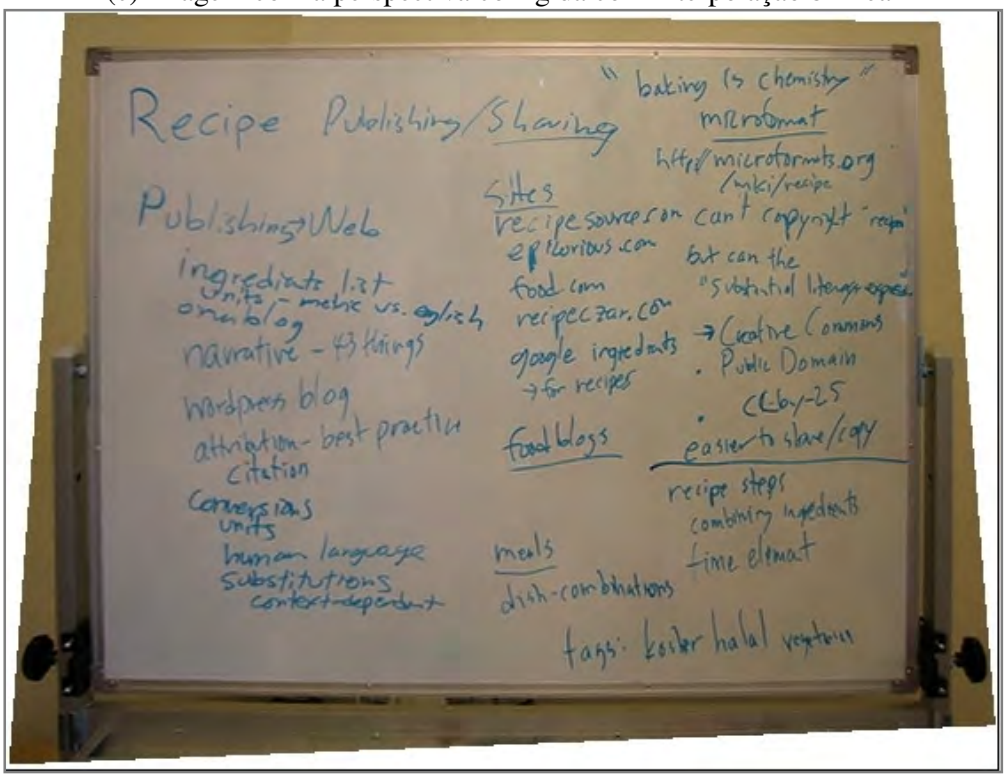


(b) Imagem com a perspectiva corrigida utilizando método do ponto mais próximo

Figura 4.8 – *Correção de perspectiva com o WhiteboardIt com vários mecanismos de interpolação*



(c) Imagem com a perspectiva corrigida com interpolação bilinear



(d) Imagem com a perspectiva corrigida com interpolação bicúbica

Figura 4.8 (cont.)

Este algoritmo apresenta uma falha relativa ao cálculo do foco em alguns casos como o da Figura 4.9 em que o valor de f^2, calculado conforme a formula da equação 4.41, é negativo.

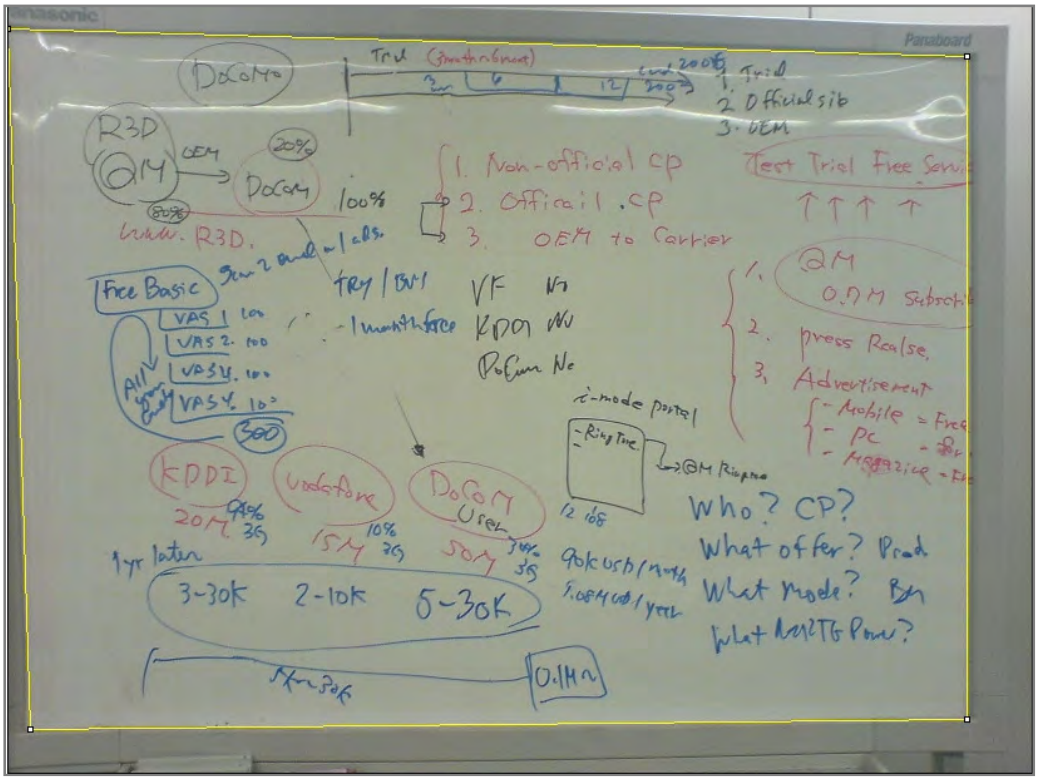


Figura 4.9 – *Exemplo de imagem em que o cálculo* $\boldsymbol{f^2}$ *foi negativo*

Para provar essa questão, as quinas delimitadas na Figura 4.9 possui os valores das coordenadas definidos por

$$\tilde{m}_1 = \begin{bmatrix} 35 \\ 1154 \\ 1 \end{bmatrix}; \tilde{m}_2 = \begin{bmatrix} 1528 \\ 1138 \\ 1 \end{bmatrix} \tilde{m}_3 = \begin{bmatrix} 9 \\ 36 \\ 1 \end{bmatrix} \tilde{m}_4 = \begin{bmatrix} 1529 \\ 81 \\ 1 \end{bmatrix}$$

, e a imagem possui tamanho de 1632x1224. Com esses valores o valor inferido de f^2 é

$$\boldsymbol{f^2} = -32850519{,}327348124$$

. Como o algoritmo foi desenvovido em Java, e essa plataforma apresenta problemas de precisão conhecidos (GOETZ, 2003), optou-se por fazer o mesmo cálculo para f^2 no Matlab®, obtendo o mesmo resultado, confirmando o problema na especificação do algoritmo. Portanto, no desenvolvimento feito pelo autor dessa dissertação utilizou-se o valor absoluto para $\boldsymbol{f^2}$, não ocasionando correção equivocada para o conjunto de imagens dessa dissertação.

4.5 Método de SILVA e LINS

O método proposto por Silva e Lins (SILVA, 2006; SILVA and LINS, 2005) apresenta um algoritmo bem mais simples, mas eficiente, consistindo numa nova estimativa da razão largura e altura. A Figura 4.10 ilustra um exemplo de quadrilátero que representa o contorno do quadro. Suas quinas representadas pelos pontos ES (esquerdo superior), DS (direito superior), EI (esquerdo inferior) e DI (direito inferior).

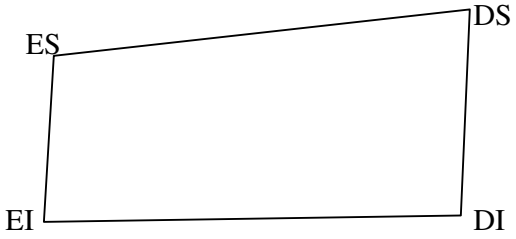


Figura 4.10 – *Quadrilátero que representa o contorno do quadro definido por suas quinas*

A razão largura e altura, que pode ser visto na Equação 4.54, nada mais é que a soma do comprimento da borda inferior e superior dividida pelo comprimento das duas bordas laterais, esquerda e direita.

$$razao_{\frac{L}{H}} = \frac{\|ES - DS\| + \|EI - DI\|}{\|ES - EI\| + \|DS - DI\|} \qquad 4.54$$

Com a razão largura e altura em mãos, é necessário definir as dimensões do quadro na imagem final. SILVA propõe que a largura da imagem final seja definida por $\|EI - DI\|$, nesse caso, pode haver a perda de informação dependendo da imagem de entrada.

A condição necessária para que não haja perda de informação é que cada ponto da imagem de entrada seja mapeado, pelo menos, em um ponto da imagem de saída, conforme dito no início deste capítulo. Calcula-se o teto dos máximos das larguras (L_{max}) e alturas (H_{max}) do quadrilátero:

$$L_{max} = \lceil \max(\|ES - DS\|, \|EI - DI\|) \rceil \qquad 4.55$$

$$H_{max} = \lceil \max(\|ES - EI\|, \|DS - DI\|) \rceil \qquad 4.56$$

. Verifica-se qual dessas dimensões abrange uma maior área, utilizando a $razao_{\frac{L}{H}}$ calculada anteriormente, obtendo a largura e altura final.

$$se\ H_{max} \times razao_{\frac{L}{H}} < L_{max}\ então\ H_{final} = H_{max}\ e\ L_{final} = H_{max} \times razao_{\frac{L}{H}} \qquad 4.57$$

$$senão\ L_{final} = L_{max}\ e\ H_{final} = \frac{L_{max}}{razao_{\frac{L}{H}}}$$

Este cálculo é muito parecido com a estimativa da altura e largura proposta pelo WhiteboardIt descrito na seção anterior, sendo diferente apenas no uso da função teto após obter o máximo entre as bordas verticais e horizontais.

Tendo em mão as dimensões do quadro na imagem final, pode-se definir as coordenadas dos quatro pontos da quina na imagem transformada. Daí resolve-se o sistema de equações, envolvendo as coordenadas da quina da imagem de entrada com a de saída, obtendo a matriz homográfica de transformação H.

Para calcular a transformação, observa-se a relação entre as coordenadas da imagem original e transformada, descritas por

$$x_i' = \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}} \quad 4.58$$

$$y_i' = \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}} \quad 4.59$$

podem ser transformadas em dois sistema de equações:

$$h_{11}x_i + h_{12}y_i + h_{13} - x_i'(h_{31}x_i + h_{32}y_i + h_{33}) = 0 \quad 4.60$$

$$h_{21}x_i + h_{22}y_i + h_{23} - y_i'(h_{31}x_i + h_{32}y_i + h_{33}) = 0 \quad 4.61$$

São dados quatro pontos de origem, e seu correspondente no destino, portanto têm-se oito equações no total, em um sistema com nove variáveis. É possível obter o nono sistema ao observar que todos os termos da soma do denominador e numerador para cálculo de x_i' e y_i' possuem um elemento da matriz H, logo pode se considerar que todos os elementos de H possuem um fator multiplicativo em comum que desaparece na computação de x_i' e y_i' (HARTLEY and ZISSERMAN, 2003). Portanto, pode-se fixar um elemento de H em um valor reduzindo o sistema para oito equações com oito variáveis. O elemento escolhido foi o h_{13}.

Para encontrar os outros elementos da matriz H, resolve-se o sistema

$$Ax = b \quad 4.62$$

achando os valores de x, onde A é igual a

$$A = \begin{bmatrix} x_1 & y_1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 & -x'_1 \\ 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 & -y'_1 \\ x_2 & y_2 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 & -x'_2 \\ 0 & 0 & x_2 & y_2 & 1 & -y'_2x_2 & -y'_2y_2 & -y'_2 \\ x_3 & y_3 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 & -x'_3 \\ 0 & 0 & x_3 & y_3 & 1 & -y'_3x_3 & -y'_3y_3 & -y'_3 \\ x_4 & y_4 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 & -x'_4 \\ 0 & 0 & x_4 & y_4 & 1 & -y'_4x_4 & -y'_4y_4 & -y'_4 \end{bmatrix} \quad 4.63$$

, b é igual a

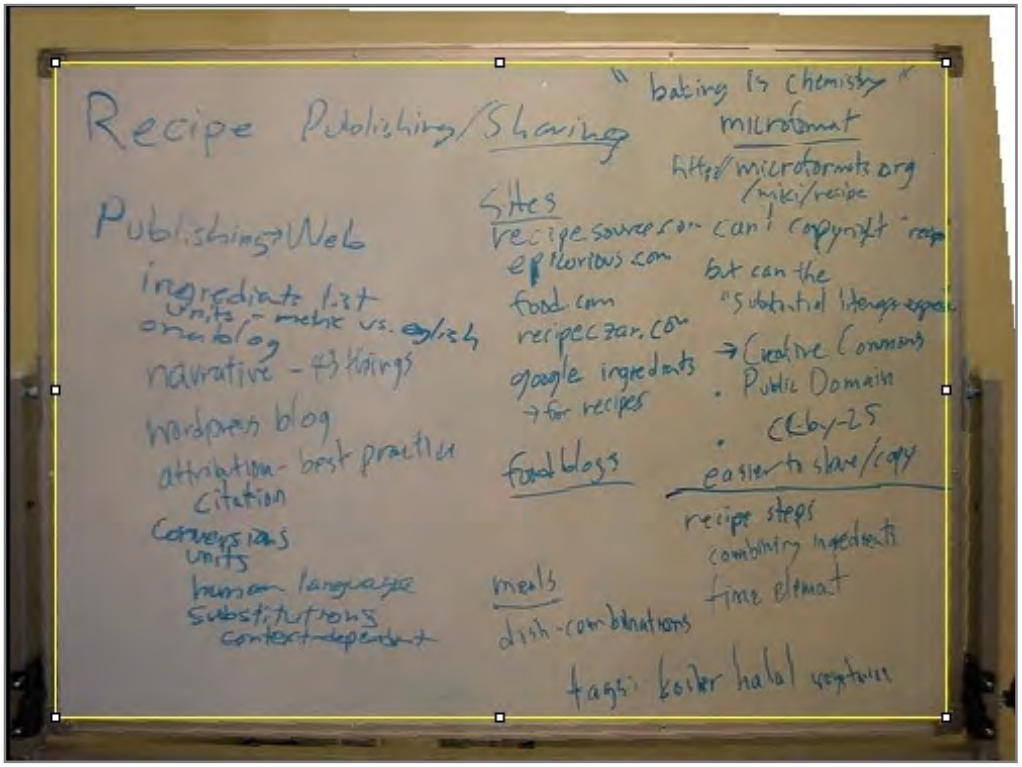
$$b = \begin{bmatrix} -h_{13} \\ 0 \\ -h_{13} \\ 0 \\ -h_{13} \\ 0 \\ -h_{13} \\ 0 \end{bmatrix} \quad 4.64$$

e x é igual a

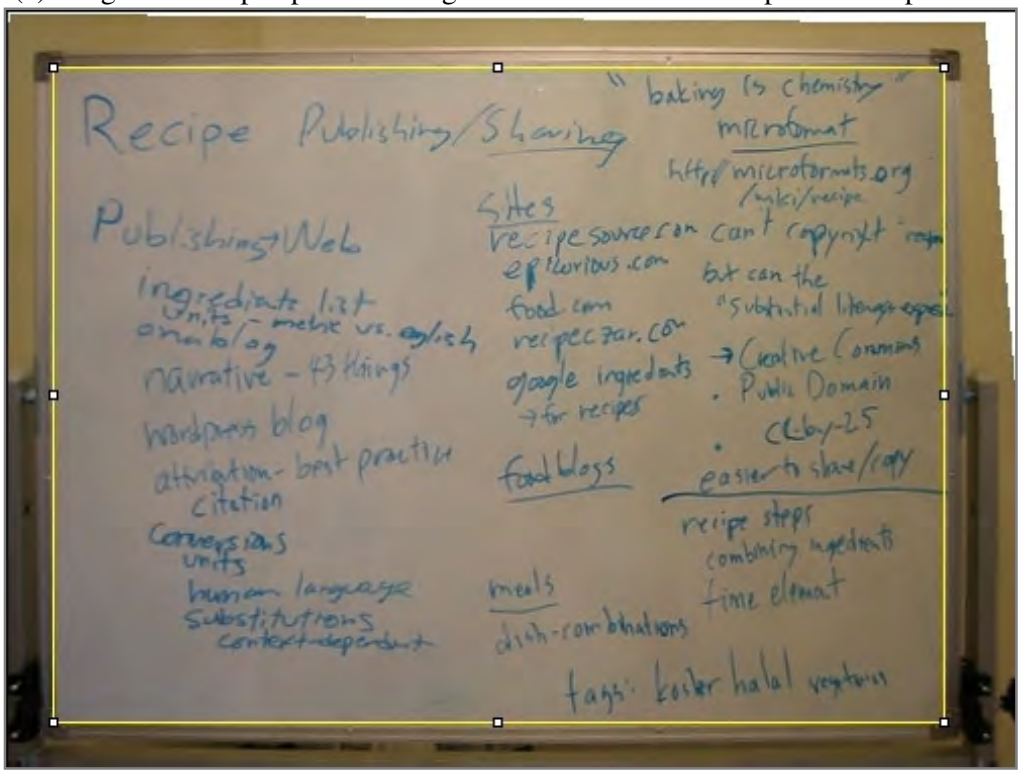
$$x = \begin{bmatrix} h_{11} \\ h_{12} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} \quad 4.65$$

.

Um exemplo desse algoritmo pode ser visto na Figura 4.11, onde a imagem original é a Figura (a).

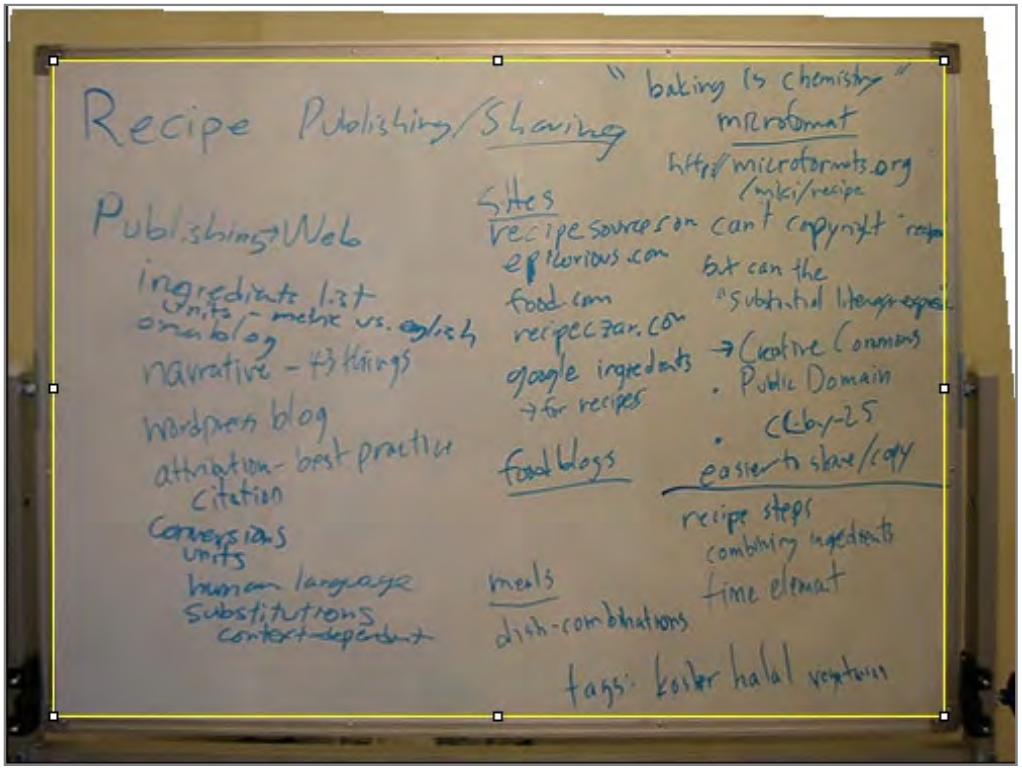


(a) Imagem com a perspectiva corrigida utilizando método do ponto mais próximo



(b) Imagem com a perspectiva corrigida com interpolação bilinear

Figura 4.11 – *Correção de perspectiva com o método proposto por SILVA com vários mecanismos de interpolação*



(c) Imagem com a perspectiva corrigida com interpolação bicúbica

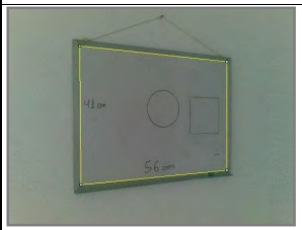
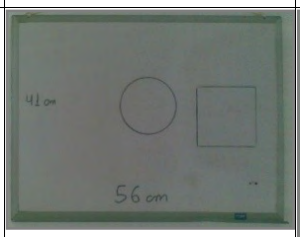
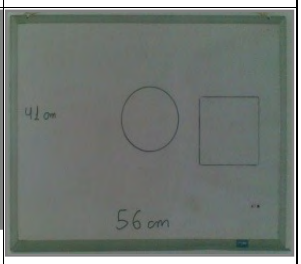
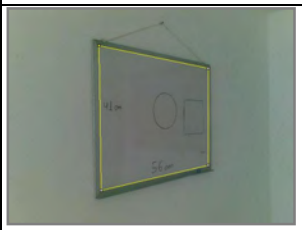
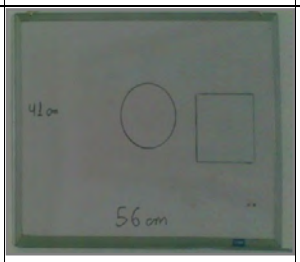
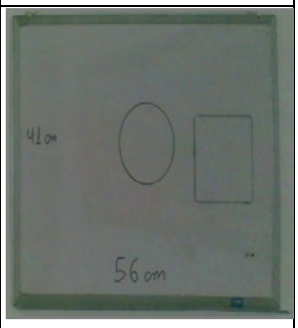
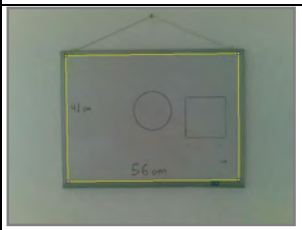
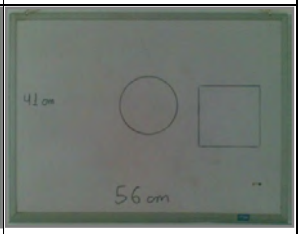
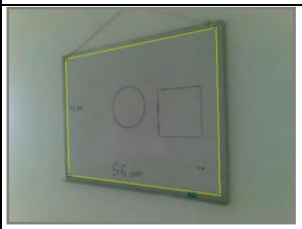
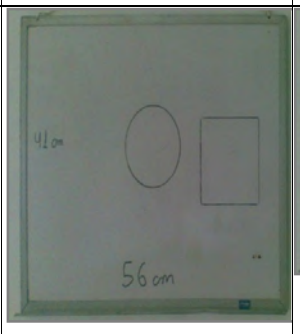
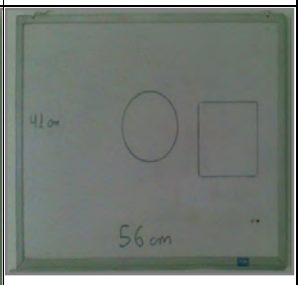
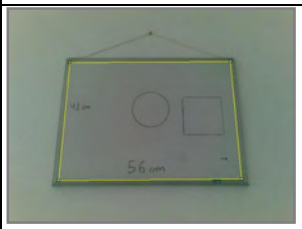
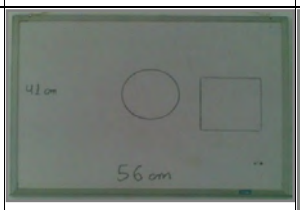
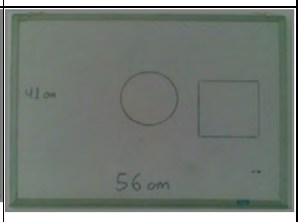
Figura 4.11 (cont.)

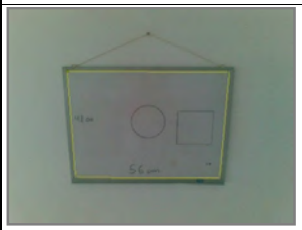
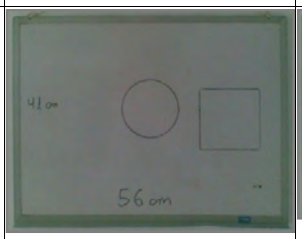
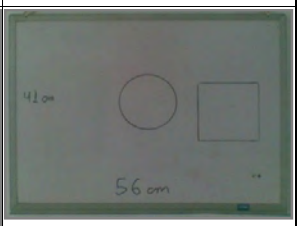
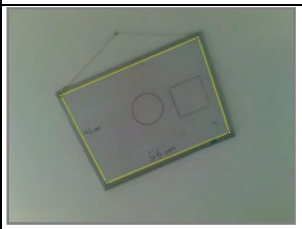
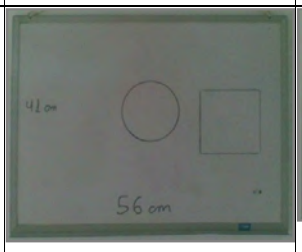
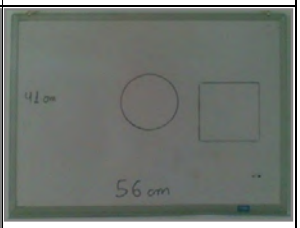
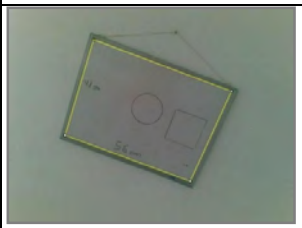
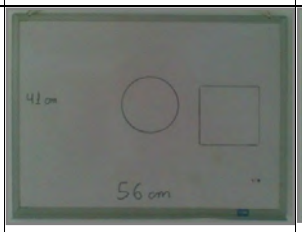
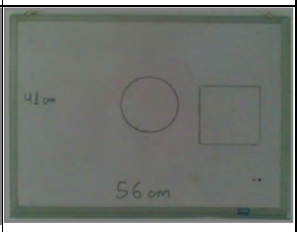
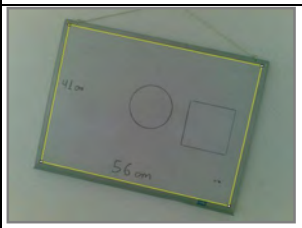
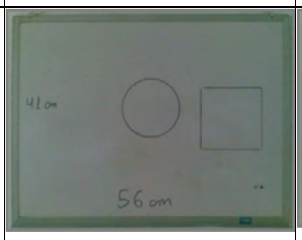
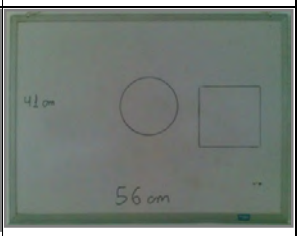
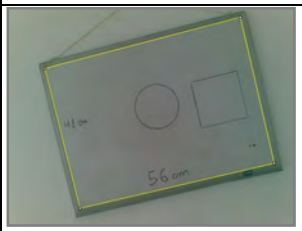
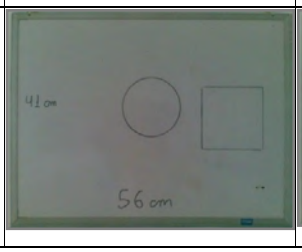
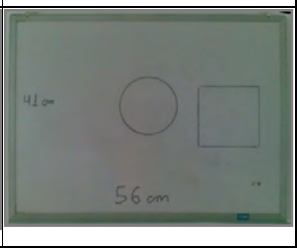
4.6 Comparação entre o método WhiteboardIt e SILVA

Para comparar os métodos de correção de perspectiva propostos pelo WhiteboardIt e por SILVA, foi montado um ambiente com um quadro de 56 cm de largura e 41 cm de altura em uma parede. Nesse quadro foi desenhado um círculo de 12 cm de diâmetro e um quadrado com 12 cm de lado para verificar se na imagem transformada suas formas são preservadas.

As fotos foram tiradas com a câmera embutida no celular Nokia 6120 classic. A tabela Tabela 4.2 mostra as fotos originais com as quinas marcadas e as imagens transformadas cortadas. O corte foi necessário a fim de enfatizar a transformação do conteúdo do quadro.

Tabela 4.2 – *Comparação entre métodos de correção de perspectiva*

Código da imagem	Imagem original	WhiteboardIt	SILVA
0			
1			
2			
3			
4			

Código da imagem	Imagem original	WhiteboardIt	SILVA
5			
6			
7			
8			
9			

Ao observar a Tabela 4.2 acima é possível notar que dois algoritmos apresentam desempenho similar, onde um gera resultados melhores em algumas situações e vice-versa, sem contar no caso em que os dois apresentam desempenho semelhante. Pode ser vista Tabela 4.2, uma comparação entre a largura real do quadro $(56\,\text{cm}/41\,\text{cm}=1{,}3659)$ e as larguras estimadas das imagens da Tabela 4.2 pelos dois algoritmos apresentados, confirmando que não a prevalência de desempenho de um com relação ao outro.

Tabela 4.3 – *Razão largura/altura estimada e comparada com a razão real entre os métodos de SILVA e WhiteboardIt*

Código da imagem	Razão real		1,3659	
	WhiteboardIt		SILVA	
	Razão	Erro	Razão	Erro
0	1,3516	-1,0%	1,2003	-12,1%
1	1,1529	-15,6%	0,9211	-32,6%
2	1,3645	-0,1%	1,3656	0,0%
3	0,9100	-33,4%	1,0732	-21,4%
4	1,5671	14,7%	1,4574	6,7%
5	1,3533	-0,9%	1,4397	5,4%
6	1,3169	-3,6%	1,4107	3,3%
7	1,3801	1,0%	1,4041	2,8%
8	1,3447	-1,5%	1,3685	0,2%
9	1,3587	-0,5%	1,3814	1,1%

O ambiente Tableau possui os dois algoritmos e cada um com os três mecanismos de mapeamento (vizinho mais próximo, interpolação bilinear e interpolação bicúbica), o usuário pode modificar o algoritmo de correção e interpolação na própria tela de configuração.

4.7 Resumo

Este capítulo teve como objetivo mostrar os principais algoritmos de correção de perspectiva para quadros didáticos existente na literatura. O algoritmo proposto por Jagannathan e Jawahar não é preciso o suficiente para que possa ser implementado.

Os algoritmos propostos por SILVA e LINS (SILVA and LINS, 2005) e pelo WhiteboardIt fazem uso de uma única matriz de transformação de perspectiva. Essa matriz resume todas as operações necessárias para a mudança de perspectiva, no entanto para encontrar os elementos da matriz é necessário o conhecimento da proporção largura e altura da imagem.

O algoritmo proposto pelo WhiteboardIt consegue, através de propriedades matemáticas, calcular os elementos da matriz sem o conhecimento da proporção largura e altura. No entanto, existe uma falha no cálculo do foco da matriz.

O algoritmo proposto por SILVA e LINS (SILVA and LINS, 2005) infere a proporção de uma maneira bem mais simples através da soma das bordas horizontais e verticais, dividindo o resultado da soma entre si. Tanto os algoritmos propostos por SILVA e LINS e WhiteboardIt apresentam desempenho similar.

Ortogonalmente a esses mecanismos deve-se considerar a necessidade de fazer interpolações na formação da imagem transformada, uma vez que o *pixel* desejado na imagem transformada, não corresponde necessariamente a um *pixel* com coordenadas inteiras na imagem original, portanto é necessário estimar o valor desse pixel através de interpolações (bilinear ou bicúbica) ou obter o valor do *pixel* mais próximo.

5 FILTROS DE REALCE PARA IMAGENS DE QUADROS DIDÁTICOS

Este capítulo tem como objetivo mostrar algoritmos de filtragem que podem ser utilizados para melhorar a qualidade da imagem de quadros didáticos. Para o conjunto de imagens em estudo, observa-se que a iluminação apresenta-se irregular sem padrão definido. Nota-se, também, que os quadros didáticos em si possuem poucas cores, algumas vezes apenas duas: a do fundo e do risco (giz ou marcador). Um filtro de realce, que seja capaz de compensar o efeito da iluminação irregular, diminui a quantidade de cores na imagem, tornando a imagem chapada, não só melhora a legibilidade da imagem, mas também pode ocasionar uma melhor compressão da imagem final.

O objetivo central do realce é transformar os pontos das imagens pertencentes ao fundo do quadro em uma cor uniforme, sem nuances nem interferências (reflexos indesejáveis) da iluminação. Já a parte referente à escrita deve ser contrastada em relação à cor constante de fundo.

No caso de quadros brancos e pretos, o fundo é bem definido no espaço RGB. Já nos quadros verdes são mais difíceis de serem definidos de maneira universal no espaço RGB, porque dependem da fabricação e do material usados.

5.1 Filtro WhiteboardIt

Nesta seção será apresentado o algoritmo de realce de imagens para quadros brancos do sistema WhiteboardIt o qual foi apresentado nos trabalhos (LIU *et al*, 2007; ZHANG and TAN, 2001; ZHANG and HE, 2004; ZHANG *et al*, 2006).

Este algoritmo foi o primeiro proposto para processamento de quadro branco. Ele define que as imagens são formadas pela cor do objeto "puro" multiplicada pelo coeficiente de iluminação, que depende do local do quadro. O procedimento em que será apresentado adiante infere o fator de iluminação e divide pelo valor da foto original, obtendo assim uma figura "pura" sem a interferência da luz. Os passos deste algoritmo são:

1. Dividir o algoritmo em blocos retangulares de tamanho fixo. Nas referências supracitadas o retângulo é de 15x15 *pixels*.

2. Para cada bloco, as cores do pixel são armazenas ordenando-as de maneira crescente de acordo com o valor da luminância. O algoritmo não especifica qual a equação de luminância, utilizou-se a Equação 5.2 (GONZALEZ, 2008)

$$luminância = 0{,}30 \times r + 0{,}59 \times g + 0{,}11 \times b \qquad 5.1$$

, que é uma das mais usadas para conversão de colorido para tons de cinza.

3. Computar o valor médio de cada componente RGB de 25% dos pixels com maior luminância. Essa média vai ser o valor do fundo local (C_{luz}), em cada componente.
4. Removem-se blocos que estão fora do plano RGB em relação aos outros blocos. Isso acontece provavelmente quando a cor em questão é de um risco do quadro e possui uma largura grande o suficiente para interferir no cálculo do fundo local. Essas cores são rejeitadas, pois são desconformes em relação ao resto. Os fundos locais dessas áreas são calculados através da média dos vizinhos. Como pode ser visto nas referências (LIU *et al*, 2007; ZHANG and TAN, 2001; ZHANG and HE, 2004; ZHANG *et al*, 2006), todo esse passo está mal especificado. Pois, não existe plano RGB, e sim cubo RGB, mesmo assim não há valores informando qual é a maior distância para um bloco não-desconforme. Para remover esses blocos será utilizado o método de remoção de blocos desconformes explicado na segunda proposta de realce do Tableau, a ser vista mais adiante na seção 5.3.2.
5. Processa-se cada pixel da imagem aplicando-se a função S descrita na Equação

$$S(x,p) = \begin{cases} 1 & se\ \ x \geq 1 \\ 0{,}5 - 0{,}5 \times cos(x^p) & se\ \ x < 1 \end{cases} \qquad 5.2$$

, onde x é a razão $C_{entrada}/C_{luz}$, sendo $C_{entrada}$ o valor do pixel original e C_{luz} o valor do fundo local (obtido nos passos 3 e 4), lembrando que essa razão é calculada em cada componente. Nas referências do WhiteboardIt, o valor de p é igual a 0,75.

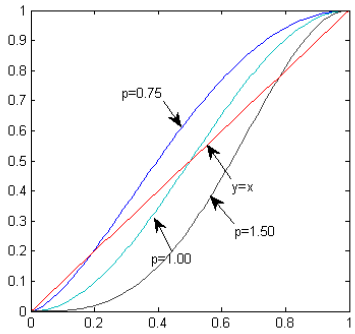


Figura 5.1 - *Função S da Equação 5.2 com diferentes valores para p*

O resultado do algoritmo WhiteboardIt pode ser visto na Tabela 5.1, onde a primeira linha contém duas imagens de quadro branco, a segunda o fundo local inferido e a terceira o resultado do processamento. As áreas brancas das imagens da segunda linha da tabela indicam blocos com valor de fundo local distante do valor da maioria dos blocos.

Tabela 5.1 – *Processamento de imagens de quadros brancos com o WhiteboardIt*

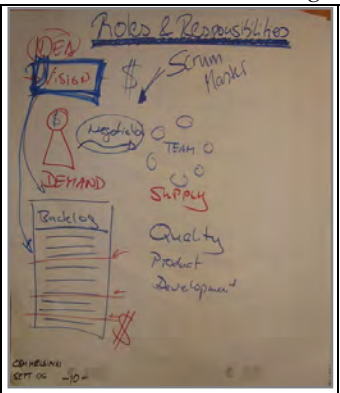
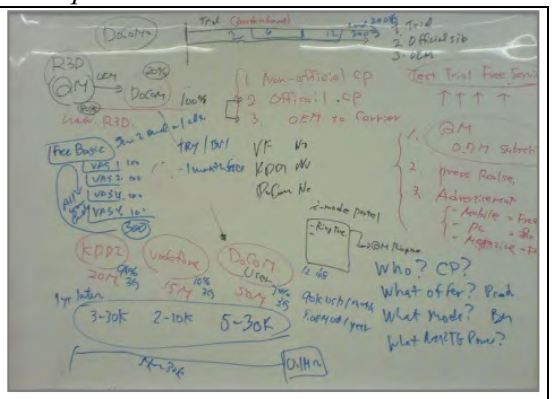
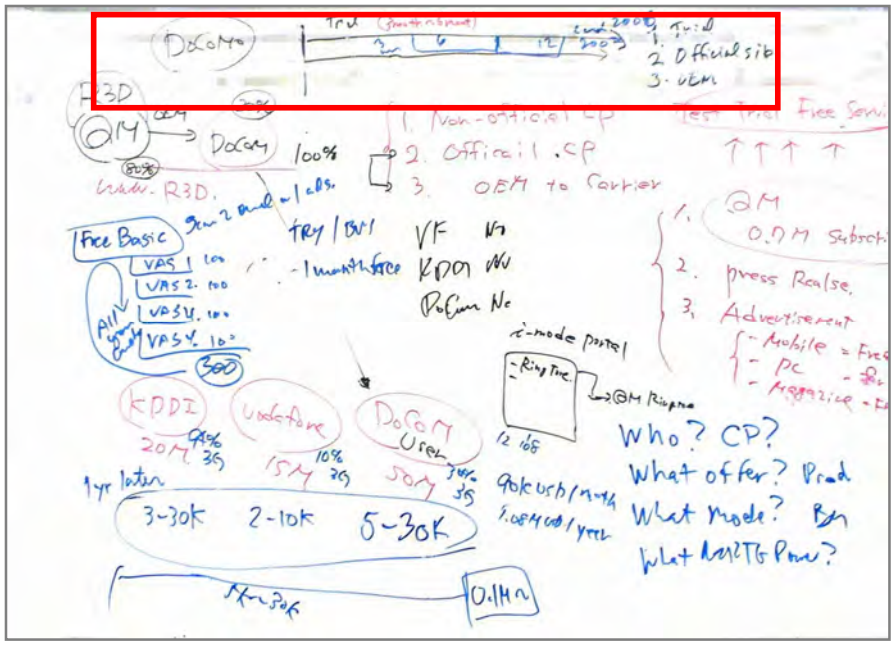
Imagem original		
Fundo local inferido		

Imagem resultante		

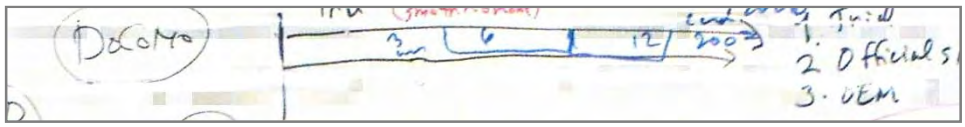
5.2 Um algoritmo de realce para quadros brancos

Apesar dos bons resultados obtidos pelo que o autor desta dissertação interpretou e inferiu ser proposto pelo WhiteboardIt, existem algumas falhas que precisam ser corrigidas. Ao se atribuir o valor de p na função S igual a 0,75, o contraste do risco com o fundo é muito baixo. Para aumentar o contraste, modifica-se o valor de p da função S para 1,50, como visto na figura abaixo, o que ocasiona outro problema: o realce indesejado dos blocos localizados no reflexo das lâmpadas da sala (a parte que está selecionada de vermelho), sendo ampliado na Figura 5.3 (b). Esse reflexo acontece devido ao alto índice de coeficiente especular dos quadros brancos.



(a) Resultado com p igual a 1,50

Figura 5.2 – *Imagem da direita na Tabela 5.1 processada com WhiteboardIt*



(b) Detalhe da imagem processada

Figura 5.2 (cont.)

Um novo algoritmo é proposto para solucionar essas duas falhas. Ele foi nomeado de Tableau-WBIT tendo em vista que este se baseia em algumas idéias do WhiteboardIt. Esse novo algoritmo na referência (OLIVEIRA and LINS, 2008) em conjunto com a segunda proposta para detecção de borda apresentada anteriormente.

O procedimento se baseia na hipótese que os riscos do quadro são finos cercados pelo fundo do quadro. A idéia central é coletar pixels que estão ao redor de cada pixel, estimar o fundo e daí calcular a saída da mesma maneira que o WhiteboardIt.

A coleta de pixels ao redor de cada ponto deve ter um raio maior que a largura do risco, caso contrário tanto o ponto central como os coletados podem pertencer ao mesmo risco. À medida que se coletam mais pontos, mais memória e computação serão necessárias nos passos posteriores. Deste modo coletam-se pixels que estejam em circunferências ao redor do ponto central. Nesse trabalho foram utilizadas três circunferências onde o ângulo entre os pontos varia-se de 22,5º graus, e o raio ($raio\_cmp$) é calculado conforme Equação 5.3.

$$raio\_cmp = RAZAO\_RAIO \times (LARGURA\_IMG + ALTURA\_IMG) \quad 5.3$$

O fator $RAZAO\_RAIO$ deve ser definido empiricamente de modo que obedeça à desigualdade da Equação 5.4.

$$RAZAO\_RAIO > \frac{LARGURA\_RISCO}{LARGURA\_IMG + ALTURA\_IMG} \quad 5.4$$

O resumo da coleta é ilustrado na Figura 5.3

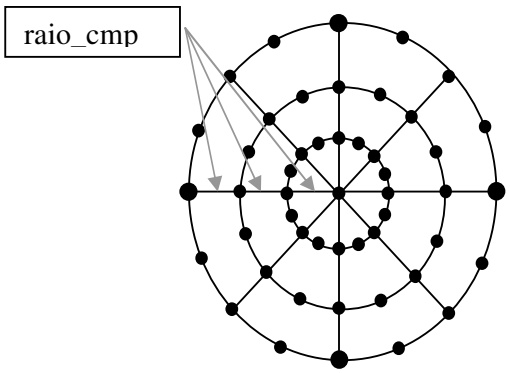


Figura 5.3 – *Posição dos pontos coletados para a estimativa do fundo, onde o pixel corrente está no centro*

Após a coleta de dados é necessário definir-se quais pixels tem uma maior probabilidade de pertencer ao fundo do quadro. Muitas vezes os riscos são pretos, azuis, vermelho, verde e roxo. Nesta etapa ordenam-se os pixels obtidos na seguinte ordem do modelo HSL: saturação, matiz e luminância. Devido ao quadro ser branco a tendência é que o fundo possua saturação baixa, porém algumas vezes a imagem não se comporta dessa maneira devido à reflexão de luz do quadro no ambiente em que a foto foi tirada. Logo, outro fator a ser considerado é que a matiz dos pixels do fundo são próximas. E por fim as medidas de luminância devem ser próximas. Para essas medidas, as escalas foram reduzidas para [0-16] na saturação, [0-359] na matiz e [0-16] para luminância.

Uma vez feita a ordenação é desejado obter-se um conjunto de pixels que são vizinhos no conjunto ordenado e apresentam uma variação baixa. Para o estudo aqui apresentado a redondeza que compreender 25% (PERC_FUNDO) do total de pontos coletados com menor desvio padrão da escala de cinza (conforme a equação de luminância do WhiteboardIt) é que será utilizada como fundo do quadro.

E por final obtém-se a cor do fundo local (C_{luz}) através da média dos valores dos pontos encontrados na etapa anterior. Para obter o pixel de saída utiliza-se a função S definida no algoritmo do WhiteboardIt, utiliza-se o mesmo valor de x, sendo igual a $C_{entrada}/C_{luz}$. O valor de p da função S foi 1,75 ao invés de 0,75, pois apresentaram melhores resultados. Um pseudocódigo do algoritmo apresentado pode ser visto logo abaixo:

```
Para cada ponto da imagem faça
  P <= Obter cores ao redor do pixel central que estão em
circunferências de raio raio_cmp, 2*raio_cmp e 3*raio_cmp espaçados em
22,5° graus
  Ordena-se o conjunto de pixels P com os critérios: saturação, matiz
e luminância
  S <= seqüência de pixels do conjunto P de tamanho igual a 25% do
número de elementos de P com menor desvio padrão da luminância
  B1 <= Média das cores do conjunto S
  Cn <= função-S(Co/C1)
```

Pseudocódigo 5.1 – *Algoritmo proposto para realce de imagens quadros brancos.*

Duas imagens resultantes do segundo realce podem ser observados na Figura 5.4 e na Figura 5.5.

Figura 5.4 – *Resultado do Realce com o Tableau-WBIT para imagem da primeira coluna da Tabela 5.1*

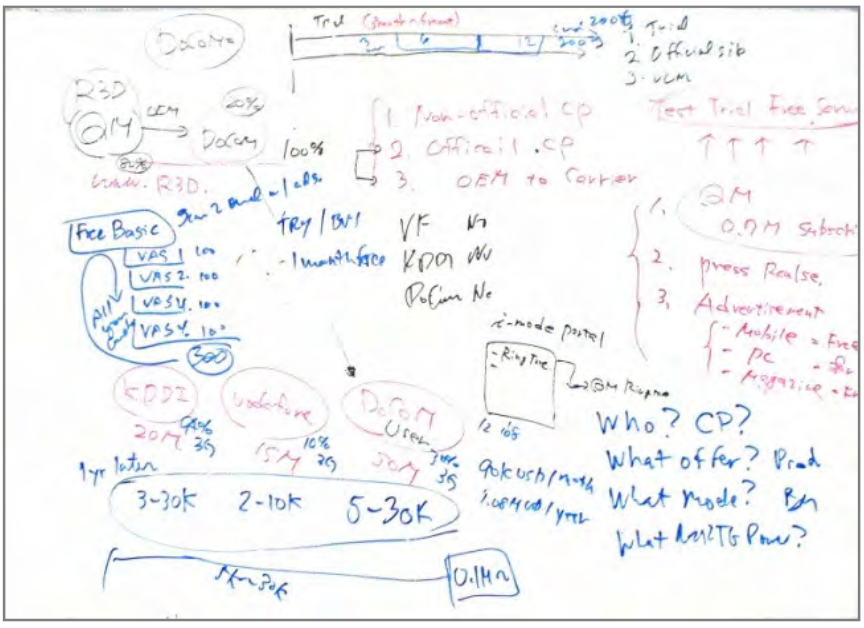


Figura 5.5 – *Resultado do Realce com o Tableau-WBIT para imagem da segunda coluna da Tabela 5.1*

5.2.1 Parâmetros do algoritmo

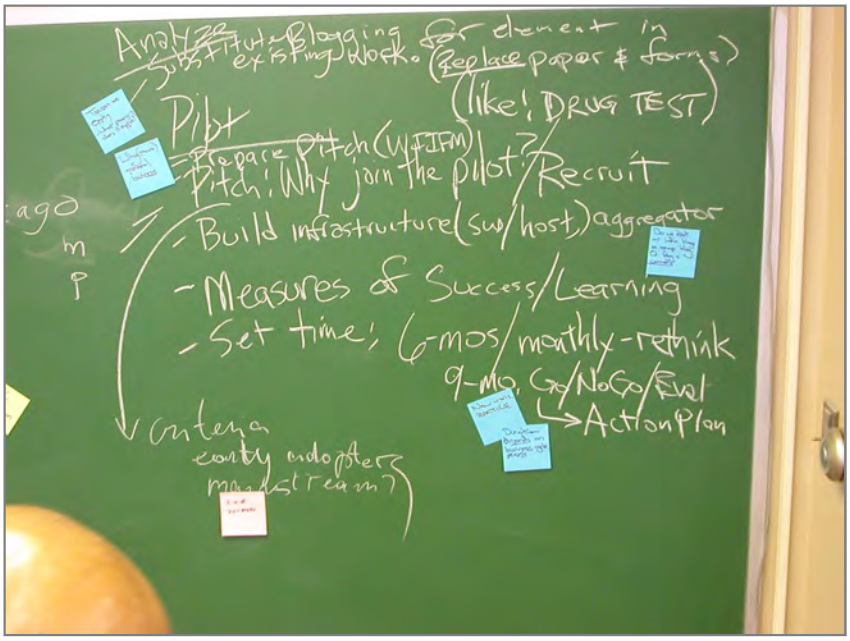
Esta seção tem como objetivo listar todos os parâmetros definidos pelo algoritmo proposto de realce bem como a consequência da sua modificação conforme descrito na Tabela 5.2.

Tabela 5.2 – *Parâmetros definidos na primeira proposta de realce do Tableau*

Parâmetro	Valor sugerido	Observações
RAZAO_RAIO	0,75%	Indica o valor que ao ser multiplicado pela soma da largura e altura da imagem, esse resultado deve ser maior que qualquer risco do quadro. A escolha de um valor pequeno pode acarretar na detecção do fundo local o próprio risco do quadro, no caso em que o valor é muito grande, o fundo estimado pode ser muito diferente do fundo local desejado.
PERC_FUNDO	25% da largura	Percentual onde é feita a análise estatística na lista ordenada. Ao utilizar um percentual baixo pode-se analisar poucos *pixels* entre si, como os riscos apresentam cores próximas pode-se inferir como fundo local do quadro. Ao utilizar um percentual alto, os *pixels* do riscos podem contaminar no calculo do fundo local final.
COLETA DOS PIXELS AO REDOR DO PONTO	Conforme a Figura 5.3	A coleta pode ser feita de várias maneiras, ao obter mais pontos é necessário mais processamento, ao utilizar poucos pontos pode-se estimar equivocadamente os fundos locais.
Valor de p para função S	1,75	Esse valor foi selecionado afim de tornar o risco mais forte. Para valores maiores que 1 o risco mais ficam mais evidentes, porém se o valor for muito alto, os ruídos da imagem podem ser ressaltados.

5.3 Um Algoritmo Geral de Realce para Quadros Didáticos

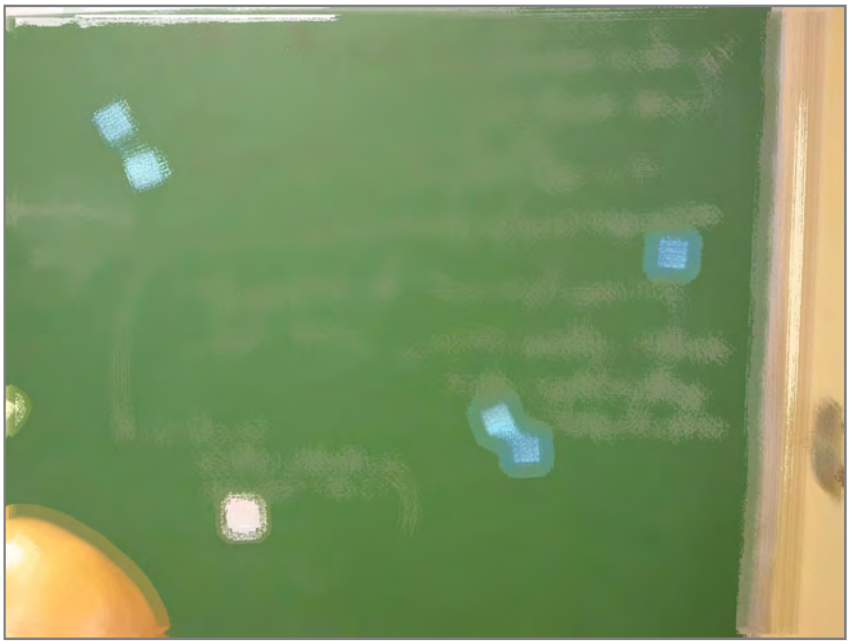
Os algoritmos anteriores assumem a premissa que o fundo do quadro é branco. No caso do WhiteboardIt, o cálculo do fundo local é obtido a partir dos *pixels* que tiverem maior luminância, o que não é verdade para quadros verdes e pretos, pois os riscos brancos possuem maior luminância que o fundo desses quadros. Ao observar os fundos inferidos obtidos pelo WhiteboardIt e Tableau-WBIT, conforme visto na figura a seguir, notou-se a necessidade de desenvolvimento de um novo algoritmo. No caso do fundo inferido do WhiteboardIt, os blocos em preto são os blocos desconformes.



(a) Imagem original

Figura 5.6 – *Inferência do fundo utilizando os algoritmos descritos anteriormente*

(b) Fundo inferido com o WhiteboardIt



(c) Fundo inferido com o Tableau-WBIT

Figura 5.6 (cont.)

Note-se que no WhiteboardIt, como já era de se esperar, os riscos brancos são detectados como parte do fundo. Já no caso do Tableau-WBIT, há casos em que o risco é muito grosso ou há presença de objetos como *Postit*® que interferem no fundo local, todavia a sua inferência de fundo local foi bem melhor que o WhiteboardIt. Portanto, é necessário mudar como os fundos locais são estimados. Isto é feito dividindo-se a lousa em

blocos de tamanho constante. Um processamento é feito em cada bloco identificando quais são possíveis fundos locais. Dentre esses blocos se identifica os que estão em desconformidade com o resto, eliminando-os. E por final, utiliza-se o fundo inferido para uniformizar o efeito da iluminação em toda a imagem. Antes de detalhar o algoritmo, duas hipóteses sobre das imagens de entrada são levantadas.

Hipótese 1. A foto apenas contém a lousa com a informação escrita, ou seja, não há moldura ou qualquer outro objeto ao redor do quadro. Para obter tal imagem basta utilizar um dos algoritmos descritos no capítulo de detecção de borda, seguido da correção de perspectiva e operação de corte.

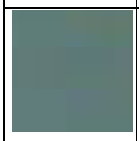
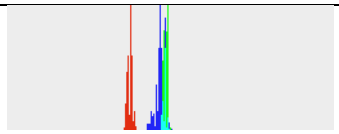
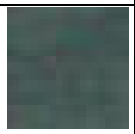
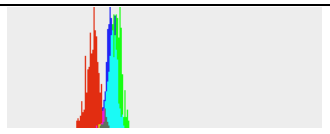
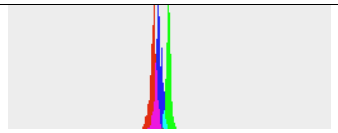
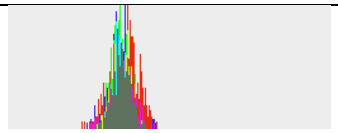
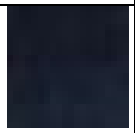
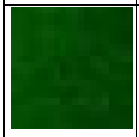
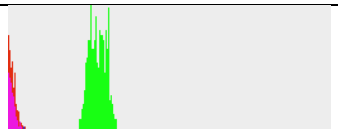
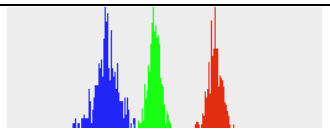
Hipótese 2. O quadro deve ter uma cor predominante, na maioria dos casos verde, branco e preto.

5.3.1 Delimitação de blocos de fundo

Inicialmente divide-se a imagem em blocos retangulares de tamanho constante. Cada bloco é analisado a fim de detectar os que possuem a distribuição uniforme das cores dos seus *pixels*. Um bloco que possui uma distribuição uniforme apresenta o histograma concentrado em todas componentes RGB. A Figura 5.7 exemplifica dois blocos que possuem distribuição uniforme, com concentração ao redor da moda da componente. Já a Tabela 5.3 apresenta vários exemplos de fundos com seus histogramas em cada componente RGB.

Figura 5.7 – *Exemplo de bloco com seu histograma*

Tabela 5.3 – *Diferentes tipos de blocos com seus histogramas*

Bloco	Histograma	σ	Bloco	Histograma	σ
		12,53			9,02
		5,46			5,23
		9,00			9,01
		32,60			36,44

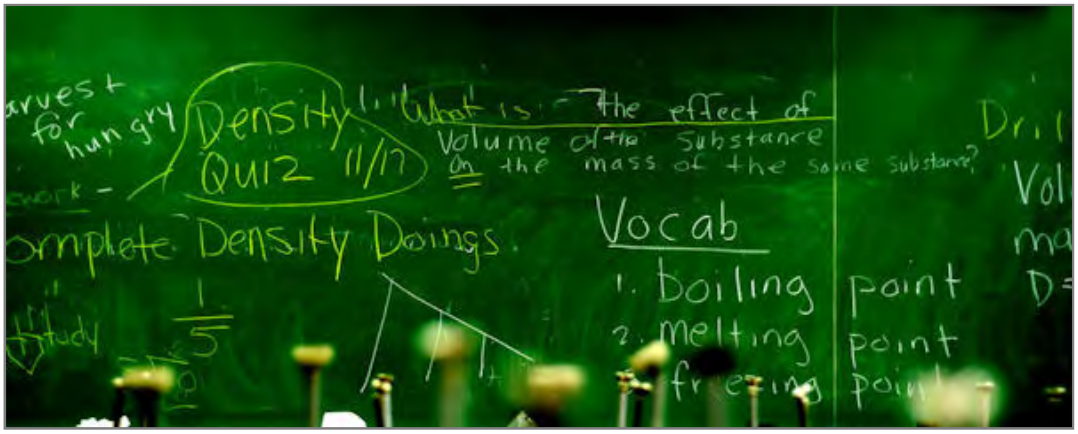
O tamanho do bloco deve ser definido de acordo com: o tamanho dos riscos, o grau de iluminação do conjunto de fotos e a resolução da imagem. Ao fazer uma escolha adequada do tamanho é possível obter histogramas dos blocos parecidos aos da Figura 5.7 e Tabela 5.3. No caso desta dissertação, o tamanho do bloco utilizado foi de 20x20 *pixels*, sendo suficiente em imagens de pequenas (0,3 Mpixels), como imagens maiores, com até 6 Mpixels. No entanto, talvez seja necessário ajustar manualmente este valor para imagens com resoluções superiores.

Para delimitar os blocos que são possíveis fundos locais, excuta-se para cada bloco:

1. Encontrar os histogramas de cada componente RGB.
2. Para cada histograma encontre sua moda ($moda_R$, $moda_G$ e $moda_B$).
3. Para cada componente RGB busca-se Q_X, que é o total de pixels que estão entre $moda_X - \Delta\text{moda}$ e $moda_X + \Delta\text{moda}$, onde Δmoda é igual a 10.
4. Se todas as quantidades Q_R, Q_G e Q_B ultrapassarem 75% (PERC_BL) dos *pixels* dos blocos, este bloco será classificado como possível fundo local, tal bloco será identificado como tendo cor ($moda_R$, $moda_G$, $moda_B$). Caso contrário o bloco não é classificado como fundo local. O uso da moda é devido à sua maior representatividade estatística, segundo (MORONNA, 2006; ROUSEEUW, 1987).

Outro problema que deve ser considerado é que a análise feita por componentes isoladamente pode resultar num fundo local com uma cor não-presente no bloco. Porém, como as componentes estão muito concentradas, ao inferir uma cor inexistente no bloco, esta vai estar bem próxima às cores dos *pixels* presentes no bloco.

A Figura 5.8 e Figura 5.9 apresentam imagens de quadros verdes correspondente aos blocos classificados como possíveis fundos. Os blocos marcados de branco indicam aqueles que não foram identificados como fundo.



(a) Imagem original

(b) Fundo inferido

Figura 5.8 – *Exemplo de fundo inferido de quadro verde*

Figura 5.9 – *Imagem com fundo inferido da Figura 5.7*

5.3.2 Eliminação de blocos desconformes

A Figura 5.8 (b) e Figura 5.9 apresentam blocos identificados como possíveis fundos. Porém, nota-se que alguns blocos da Figura 5.8 (b) foram identificadas como possíveis fundos que não pertencem à lousa do quadro (neste caso, as cadeiras de sala de aula), pois esses blocos apresentaram cores uniformes. Esta seção descreve a eliminação dos falsos "candidatos" a fundo.

Existem vários mecanismos na literatura que separam as imagens em diferentes regiões (ex: céu, prédios, outros objetos etc.) utilizando apenas a sua distribuição de cores, uma comparação entre os algoritmos com esse propósito é feita por Cheng e seus colegas (CHENG *et al*, 2001), onde se destaca o algoritmo proposto por Celenk (CELENK, 1990), que é muito citado por outros artigos. Esse algoritmo utiliza o espaço CIE L*a*b* (em termos de luminância, crominância e matiz) com os valores das cores dos *pixels* de toda a imagem para identificar as regiões da imagem, pois o CIE L*a*b* é um dos espaços mais eficientes na representação perceptual de cores.

No entanto, o propósito desta seção é identificar as cores da lousa do quadro a partir das cores dos fundos locais, que são definidas por $moda_R$, $moda_G$ e $moda_B$ do seu bloco, ao invés dos *pixels* de toda a imagem. A obtenção dos fundos locais separa a imagem em regiões de cores constantes.

A Figura 5.10 (a) denota um caso de quadro branco com anotações tipo *Postit*®. Esses *Postit*® são identificados como fundo, pois apresentam cores mais uniformes que a própria lousa do quadro, a sua distribuição de cores no espaço RGB pode ser vista logo abaixo da figura. Já a Figura 5.10 (b) ilustra os possíveis fundos, sendo denotada abaixo sua distribuição de cores no espaço RGB. É perceptível que a identificação de possíveis fundos ajudou visualmente na separação das regiões de cores constantes que fazem parte ou não da lousa do quadro. As Figuras 5.9 (c) e (d) são geradas pelo complemento do ImageJ, ColorInspector3D disponível em (BARTHEL, 2008).

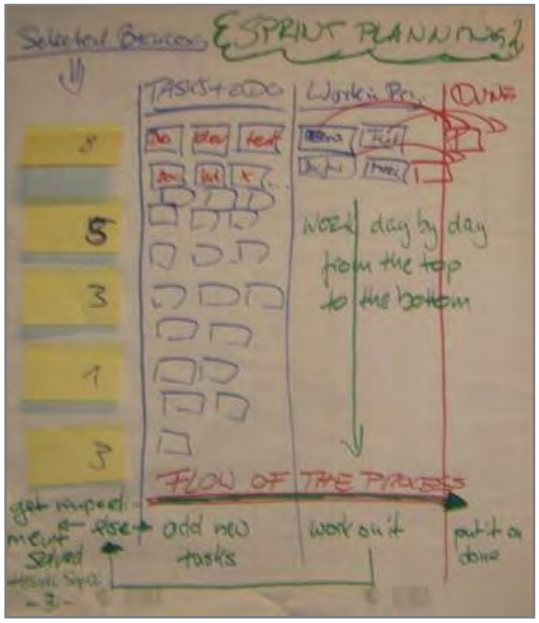
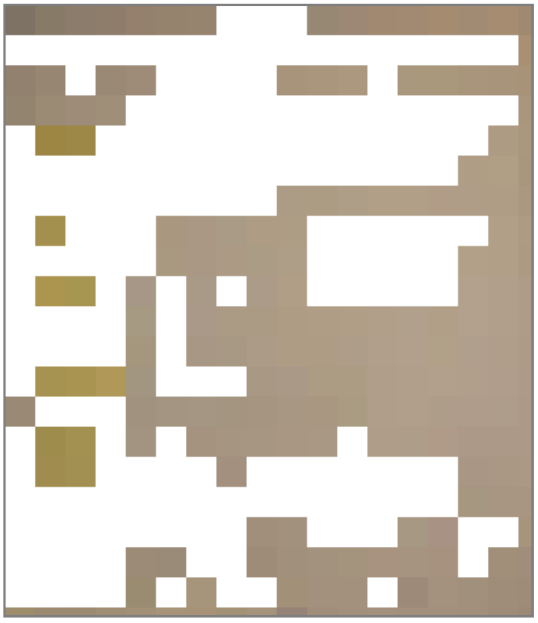
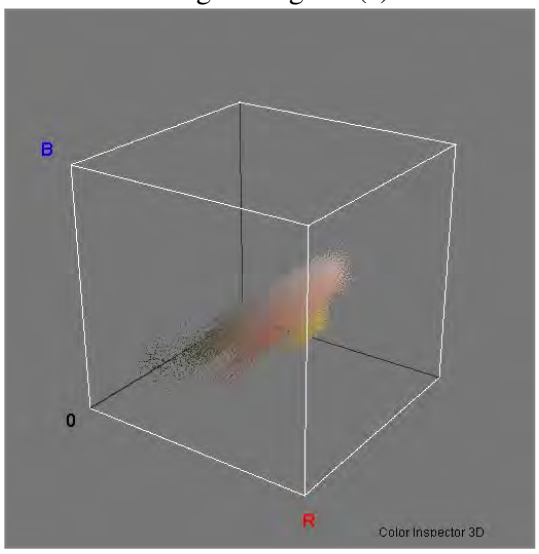


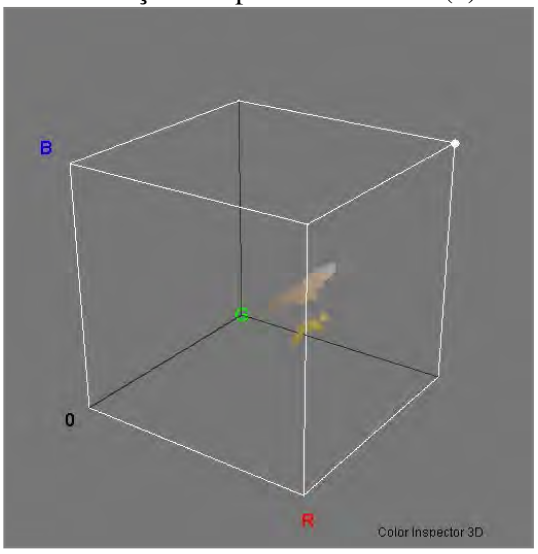
Imagem original (a)



Definição dos possíveis fundos (b)



Cores de (a) no espaço RGB (c)



Cores de (b) no espaço RGB (d)

Figura 5.10 – *Análise da Imagem de um quadro branco com Postit®*

Vislumbrou-se a idéia de utilizar algum dos algoritmos de identificação de agrupamentos, porém todos demonstraram uma complexidade computacional desnecessária, tendo em vista que os pontos já apresentam uma separação mais fácil de ser tratada, comprovada na Figura 5.10. Além da difícil implementação destes algoritmos. O procedimento proposto pode ser visto a seguir, a escala de cada componente está entre 0 e 255.

1. Seja uma lista C com todos os candidatos a blocos

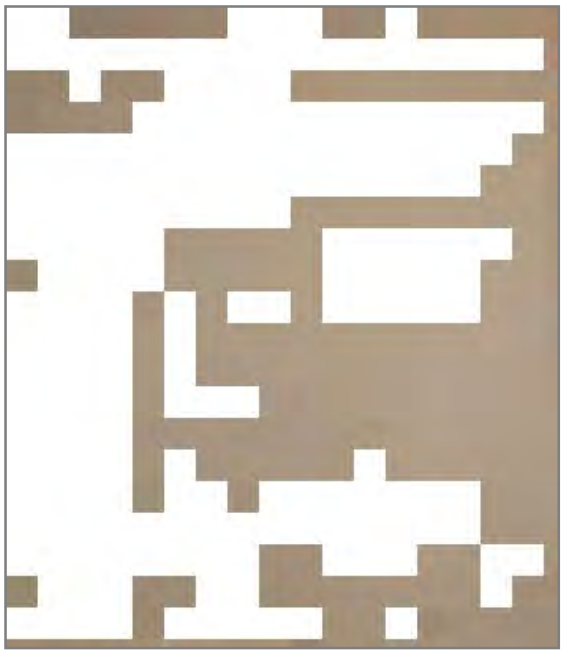
2. Para cada par de blocos C_i e C_j em C, se a diferença entre os valores de cada componente RGB desses dois blocos for menor ou igual a 6 (ARESTA_MAX), encontrada empiricamente, esses dois blocos devem pertencer ao mesmo conjunto.
3. Os conjuntos encontrados com elementos em comum são unidos até que todos sejam disjuntos entre si.
4. O conjunto com maior cardinalidade representa os fundos locais do quadro. O fundo global é encontrado através da média de cada componente RGB desse conjunto.

Ao desenvolver o algoritmo listado acima é possível fazer algumas otimizações. A primeira otimização decorre da comparação entre C_i e C_j ser simétrica, portanto ao comparar-se C_i com C_j, não há a necessidade de se comparar C_j com C_i. A segunda é a execução dos passos 2 e 3 poder ser mesclada, a medida que se compara os blocos C_i e C_j, a união dos conjuntos pode ser feita neste mesmo passo caso a distância entre as cores destes blocos seja menor que 6.

A Figura 5.11 mostra os fundos locais da lousa com a remoção dos fundos desconformes a partir da execução desse passo. É possível notar que o fundo detectado erroneamente da cadeira foi removido da Figura 5.8 (a), assim como o fundo do *Postit*® da Figura 5.10 (a). Já o fundo da Figura 5.9 ficou praticamente inalterado, apenas removendo os fundos da borda inferior da imagem.

(a) Fundo inferido da Figura 5.8 (a)

(b) Fundo inferido da Figura 5.7



(c) Fundo inferido da Figura 5.10 (a)

Figura 5.11 – *Fundo inferido após remoção de blocos desconformes*

5.3.3 Delimitação do fundo global

O passo anterior descreve o reconhecimento dos blocos não-escritos do quadro, conectando-se os "fundos locais". Faz-se necessário definir-se um fundo global. No caso do quadro branco os fundos locais obtidos não são brancos "puros" no espaço RGB, a mesma coisa acontece para os quadros pretos e verdes. Portanto pode-se definir o fundo global de duas maneiras. A primeira é através da média de cada componente dos fundos locais obtidos no passo anterior.

A segunda maneira é utilizar essa média e verificar qual é a cor mais próxima de uma lista de cores "ideais" de fundo. No caso desse estudo a lista de cores é (valores RGB no formato hexadecimal RRGGBB): branco (FFFFFF), preto (000000) e verde escuro (005200). Essas cores podem ser visualizadas na Figura 1.14.

Para identificar qual a cor mais próxima foi utilizado o padrão CIEDE2000 (CIE, 2001). Esse padrão define como calcular as diferenças entre duas cores que estão no formato CIE L*a*b*, que é um espaço de cor perceptual, para mais informações consulte o Apêndice B. No desenvolvimento do algoritmo se considerou as observações de (SHARMA *et al*, 2005), pois a especificação original peca na descrição de alguns detalhes, como a utilização da função tan^{-1} com um argumento para calcular a matiz, que abrange apenas dois quadrantes. Uma das modificações de Sharma define que a tan^{-1} deve utilizar dois argumentos, compreendendo quatro quadrantes, ou seja, todo o círculo trigonométrico.

Existem outros padrões, tanto de sistemas de cores perceptuais como o CIELUV, NC-IIIC, o CIE L*a*b* Normalizado (que é mais uniforme que o CIE L*a*b*). Como também padrões de cálculos de diferenças como o CMC e CIEDE94. Nakayama (2004) faz comparações entre os padrões de diferenças e nota-se que não há uma melhora considerável entre os sistemas ao utilizar o padrão CIEDE2000 com o CIE L*a*b* normalizado. Mais detalhes sobre o padrão CIEDE2000 são apresentados no Apêndice C ou em (NAKAYAMA and IKEDA, 2004; SHARMA *et al*, 2005).

Os pesos utilizados foram $k_L = 1$, $k_C = 0{,}8$ e $k_H = 0{,}8$. Estes pesos são inversamente proporcionais as diferenças de luminância, crominância e matiz entre as duas cores. O peso da matiz e crominância deve ser menor devido ao fato de existirem duas cores acromáticas na lista de cores (branco e preto), sendo observado várias vezes que a cor de fundo verde foi detectada, equivocadamente, onde deveria ser branca. Isso ocorreu devido à proximidade da crominância e matiz do fundo global com a cor verde utilizada.

Num conjunto de 187 imagens, 31 delas apresentaram a detecção equivocada do fundo, totalizando em uma taxa de erro de 17%. Considerando esse erro, optou-se por perguntar ao usuário do sistema o fundo global escolhido, sugerindo o fundo mais próximo da lista. Conforme mostrado na Figura 1.10. O usuário pode optar como o fundo global a média de cada componente dos fundos locais, caso o fundo não esteja na lista apresentada.

O fundo global definido acima é representado pela variável B_g.

5.3.4 Realce dos blocos

Esta seção tem como objetivo mostrar como é feito o mapeamento do pixel da imagem original para a imagem de saída. Assim como os outros algoritmos apresentados, o resultado final deve apresentar o fundo do quadro com uma cor constante contrastando com os riscos dos quadros.

A idéia central do algoritmo é igualar a razão entre o fundo local e o pixel da imagem original, com a do fundo global com o pixel de saída. Os valores do fundo global e local foram anteriormente obtidos, e o pixel original está presente na imagem em si. Portanto a saída é inferida através dessas três variáveis. Essa divisão é feita em cada componente separado.

Todavia, deve-se considerar 3 casos para fazer esse cálculo em cada componente RGB o que vai ser mostrado a seguir. Os níveis das componentes são definidos entre 0 e 1.

5.3.4.1 Caso 1

O primeiro caso ocorre quando o valor da componente do *pixel* original está entre 0 e a componente correspondente do fundo local. A figura abaixo demonstra essa situação.

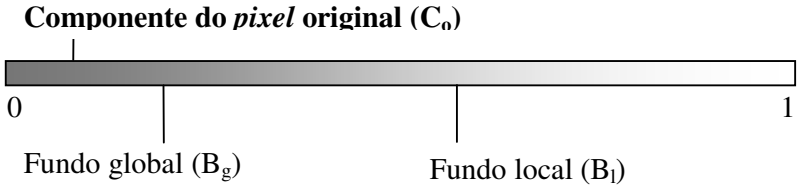


Figura 5.12 – *Caso onde o componente do pixel original está entre 0 e o fundo local*

É desejado igualar as razões entre cada componente do *pixel* original com o fundo local e as respectivas componentes do *pixel* final com o fundo global, conforme descrito na equação abaixo.

$$\frac{C_n}{B_g} = \frac{C_o}{B_l} \tag{5.5}$$

Observa-se que a razão está o intervalo [0, 1). Multiplicando-se B_g nos dois da equação obtém-se uma fórmula fechada para o cálculo de C_n.

$$C_n = \frac{C_o}{B_l} \times B_g \quad 5.6$$

Esta fórmula vai ser reescrita conforme abaixo para ser útil mais adiante. O valor de D_1 significa a diferença entre o fundo local e o pixel original e R_1 a razão da diferença pelo fundo local.

$$D_1 = B_l - C_o \quad 5.7$$

$$R_1 = \frac{D_1}{B_l} \quad 5.8$$

$$C_n = \frac{B_l - D_1}{B_l} \times B_g \quad 5.9$$

$$C_n = \left(1 - \frac{D_1}{B_l}\right) \times B_g \quad 5.10$$

$$C_n = (1 - R_1) \times B_g \quad 5.11$$

5.3.4.2 Caso 2

No Segundo caso, o valor do pixel original está localizado entre o fundo local e o valor 1. A figura a seguir ilustra essa situação.

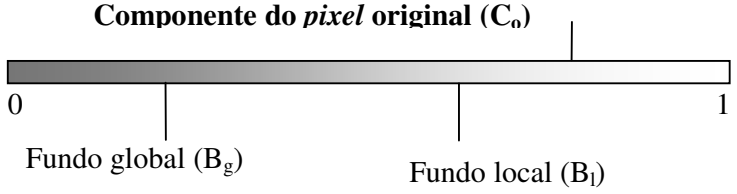


Figura 5.13 – *Caso onde o componente do pixel original está entre o fundo local e o valor 1*

Se a razão a ser considerada for $\frac{C_o}{B_l}$, o seu valor vai tender a infinito à medida que B_l se aproxima de 0. Uma alternativa seria inverter a razão, mas como valor desejado é o *pixel* novo isto não é possível.

A saída da razão anterior ficou entre um e zero porque Co nunca será maior que B_l. Observe que os valores C_o e B_l representam a distância das componentes em relação ao nível 0. É mais conveniente nesse caso utilizar distância do valor para o nível 1, o que

resulta no cálculo dos valores de $1 - C_o$ e $1 - B_l$, pois $1 - C_o$ não é maior que $1 - B_l$ de acordo com a premissa desse caso. Portanto analogamente as duas razões são igualadas.

$$\frac{1 - C_n}{1 - B_g} = \frac{1 - C_o}{1 - B_l} \quad 5.12$$

Da mesma maneira é necessário achar uma fórmula fechada para o cálculo do C_n.

$$1 - C_n = \frac{1 - C_o}{1 - B_l} \times (1 - B_g) \quad 5.13$$

$$C_n = 1 - \left(\frac{1 - C_o}{1 - B_l}\right) \times (1 - B_g) \quad 5.14$$

A equação acima é reescrita numa forma mais conveniente a ser utilizada posteriormente.

$$D_2 = B_l - C_o \quad 5.15$$

$$R_2 = \frac{D_2}{1 - B_l} \quad 5.16$$

$$C_n = 1 - \left(\frac{1 - B_l + D_2}{1 - B_l}\right) \times (1 - B_g) \quad 5.17$$

$$C_n = 1 - \left(1 + \frac{D_2}{1 - B_l}\right) \times (1 - B_g) \quad 5.18$$

$$C_n = 1 - (1 + R_2) \times (1 - B_g) \quad 5.19$$

5.3.4.3 Caso 3

O terceiro caso ocorre quando C_o é igual à B_l, o que torna o pixel resultante igual ao fundo global. Sendo este resultado coerente com os dois casos anteriores. Se $C_o = B_l$ então:

- No primeiro caso

$$C_n = (1 - R_2) \times B_g = (1 - 0) \times B_g = B_g \quad 5.20$$

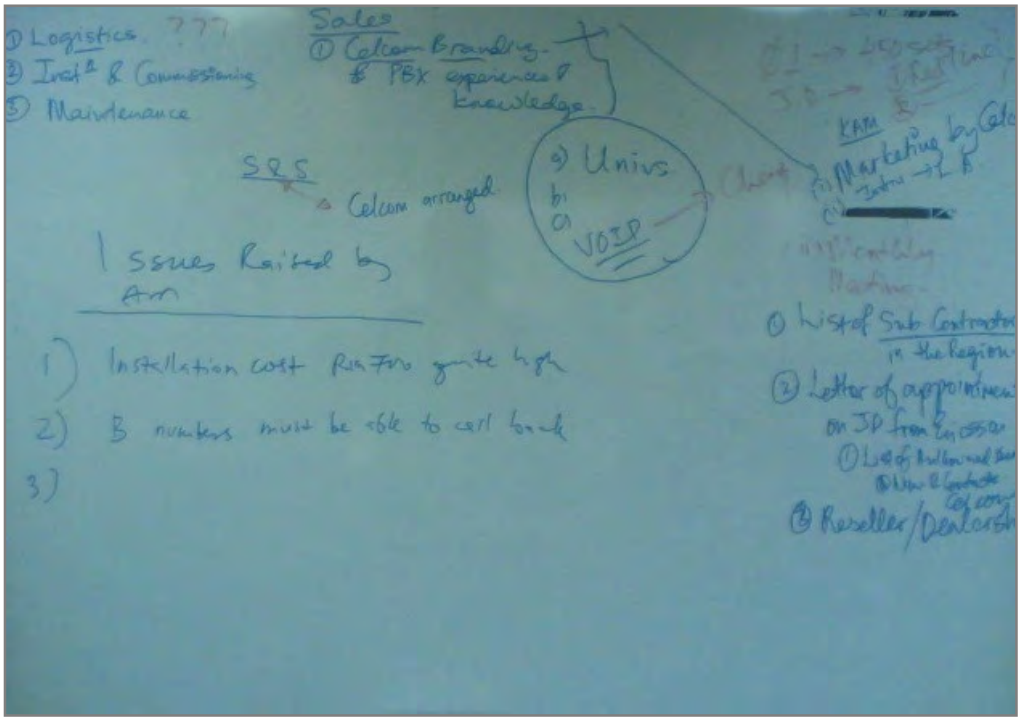
- No segundo caso

$$C_n = 1 - (1 + R_2) \times (1 - B_g) = 1 - (1 + 0) \times (1 - B_g) = B_g \quad 5.21$$

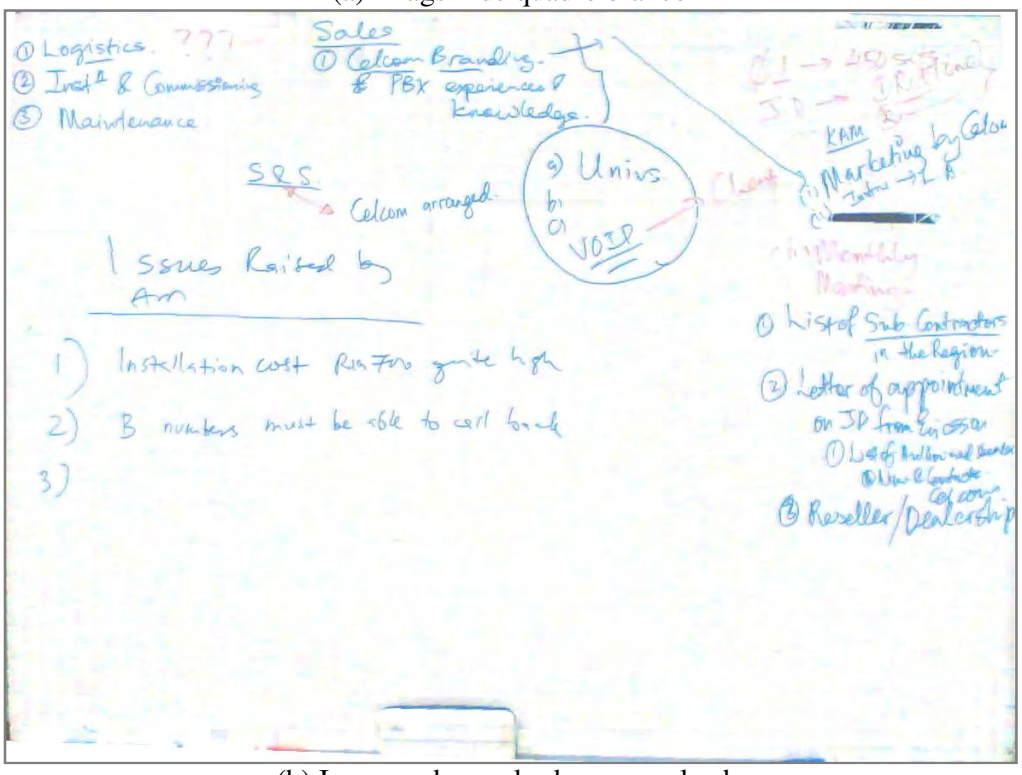
5.3.5 Considerações sobre a equivalência das razões

Esta seção tem como objetivo mostrar imagens resultantes quando executados os passos apresentados anteriormente. A iluminação foi normalizada, porém a saída não é a imagem ideal. No caso da Figura 5.17 a imagem original é melhor que a processada. Na

figura do quadro negro, a os riscos antecedentes são realçados erroneamente. Tais problemas serão contornados na próxima seção.



(a) Imagem de quadro branco



(b) Imagem do quadro branco realçada

Figura 5.14 – *Resultado preliminar do realce*

(a) Imagem original (b) Imagem realçada

Figura 5.15 –*Exemplo de processamento de quadro negro com a nova proposta de realce*

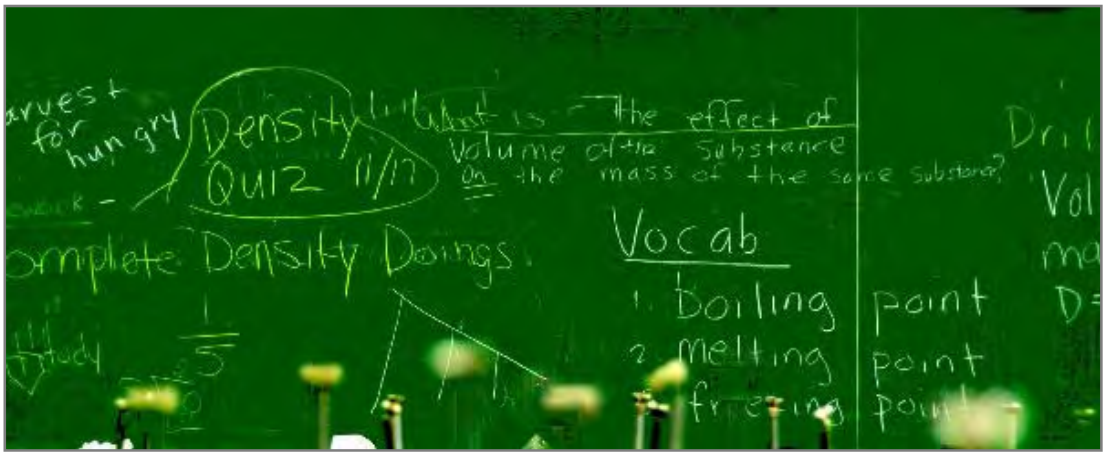


Figura 5.16 – *Figura 5.8 (a) processada utilizando o verde da lista como fundo global*

(a) Figura 5.7 processada utilizando o verde da lista como fundo global

(b) Detalhe da imagem original

(c) Detalhe do mesmo local na imagem processada

Figura 5.17 – *Figura 5.7 processada com a abordagem preliminar do realce*

5.3.6 Melhorando o cálculo da razão

O algoritmo do WhiteboardIt faz uso da função S no cálculo do valor resultante tornando a imagem mais suave e agradável de ser vista. A função S tem o objetivo diminuir sua entrada quando ela está próxima de zero e aumentar quando a mesma está próxima de 1, conforme descrito na Figura 5.1.

O defeito que está presente nas figuras acima pode ser resumido em pequenas mudanças de tons, como o olho humano é sensível ao olhar bordas de imagens elas apresentam um aspecto “desagradável”. Para isso se faz o uso de funções S para suavizar essas transições.

No entanto a função S descrita na Equação 5.2 deve ser modificada para aceitar entradas negativas, conforme descrito na Equação 5.22.

$$S(x) = \text{sinal}(x) \times (0.5 - 0.5 \times \cos(|x|^{p}\pi)) \qquad 5.22$$

Onde a função sinal é definida por

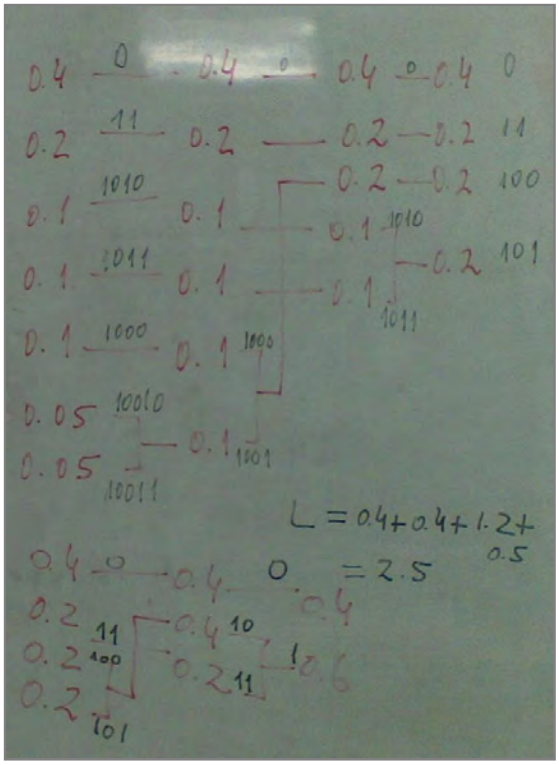
$$\text{sinal}(x) = \begin{cases} -1 & x < 0 \\ 0 & x = 0 \\ 1 & x > 0 \end{cases} \qquad 5.23$$

A primeira função S (S_D) é utilizada no cálculo da diferença entre componentes com o valor do parâmetro p denotado por P_d, essa computação é necessária para que pequenas diferenças não sejam realçadas equivocadamente. Para enfatizar essa situação, considera-se o caso em que $C_o = 0{,}45$, $B_l = 0{,}5$ e $B_g = 0{,}8$, então $C_o = \frac{C_l - D_1}{B_l} \times B_g = \frac{0{,}5 - 0{,}05}{0{,}5} \times 0{,}9$. Ou seja, a diferença na imagem realçada foi de 0,05 para 0,09, quase dobrando. Para atenuar a diferença calcula-se a função S para D_1.

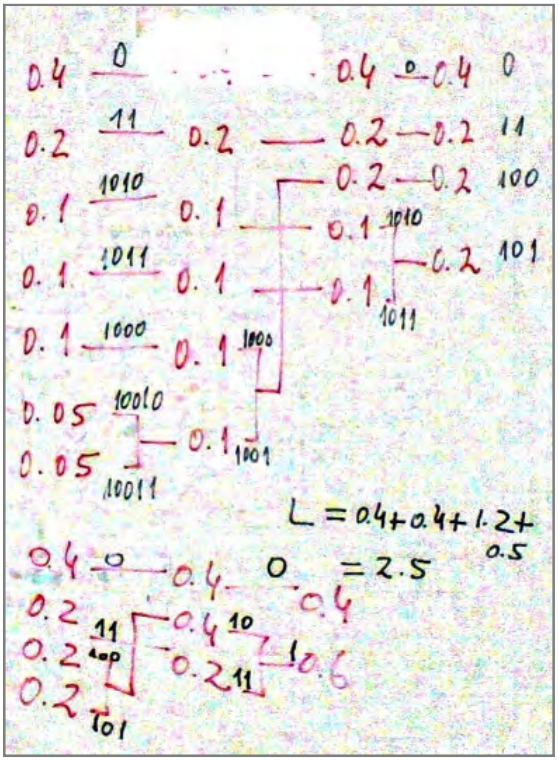
A segunda função S (S_R) é introduzida na razão resultante com $p = P_r$. O benefício provido é dar um maior contraste entre o fundo e o risco do quadro.

A função S_R possuir maior influência que no resultado final. Já a S_D é mais útil para remover vestígios do quadro sem prejudicar o realce final da imagem.

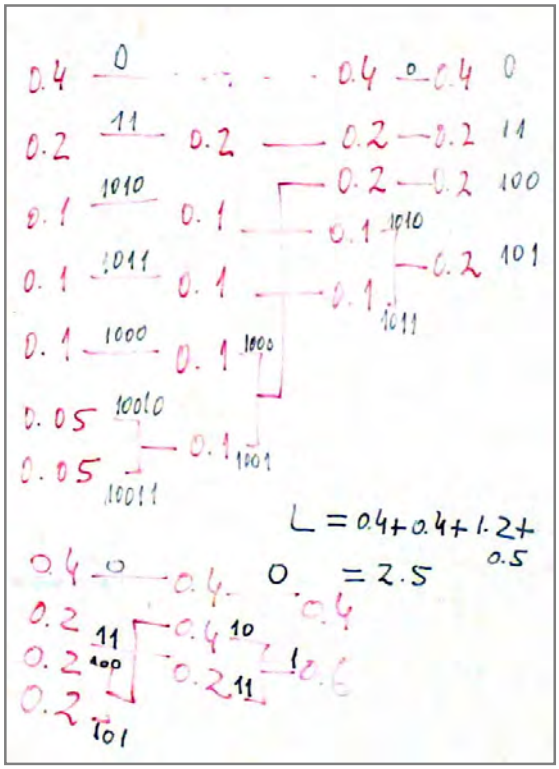
Não há como se definir os valores ideais para P_d e P_r, pois estes dependem do conjunto de imagens. No caso de imagens onde o contraste é baixo, devem ser usar valores baixos para P_d e P_r para aguçar as diferenças de cores, porém se forem muito baixos os ruídos serão realçados. As figuras abaixo demonstram a aplicação do algoritmo numa imagem tirada a partir de um celular Nokia 6120 classic, que apresenta um baixo contraste. A primeira é o resultado para os valores de $P_d = 0{,}5$ e $P_r = 0{,}5$. Já na segunda é o resultado para valores de $P_d = 0{,}6$ e $P_r = 0{,}7$. No primeiro os ruídos foram ressaltados indesejadamente, no segundo apresenta uma relação sinal/ruído melhor.



(a) Imagem original



(b) Processamento com $P_d = 0{,}5$ e $P_r = 0{,}5$



(c) Processamento com $P_d = 0{,}6$ e $P_r = 0{,}7$

Figura 5.18 – *Imagem processada com diferentes valores para* $\boldsymbol{P_d}$ *e* $\boldsymbol{P_r}$

Já há imagens como no caso da Figura 5.15, onde o "ruído" formado pelos riscos remanescentes é muito evidente, só podendo ser removido ao se atribuir valores altos para P_d e P_r. A Figura 5.19 (a) ilustra o resultado do processamento com os mesmos valores utilizados para a Figura 5.18 (c), notando-se o realce dos riscos remanescentes. Para retirá-los completamente foram utilizados valores de $P_d = 1{,}0$ e $P_r = 1{,}0$ conforme visto na Figura 5.19 (b).

(a) Processamento com $\mathrm{P_d} = 0{,}6$ e $\mathrm{P_r} = 0{,}7$ (b) Processamento com $\mathrm{P_d} = 1{,}0$ e $\mathrm{P_r} = 1{,}0$

Figura 5.19 – *Figura 5.15 (a) processada com diferentes valores para* $\boldsymbol{P_d}$ *e* $\boldsymbol{P_r}$

Devido à incorporação da função S as equações para calcular a saída foram escritas em termos de D_k e R_k. O valor de saída com a função S é incorporada conforme a seguir:

- Para o caso 1

$$D_1 = S_D(B_l - C_o) \tag{5.24}$$

$$R_1 = S_R\left(\frac{D_1}{B_l}\right) \tag{5.25}$$

$$C_n = (1 - R_1) \times B_g \tag{5.26}$$

- Para o caso 2

$$D_2 = S_D(B_l - C_o) \quad 5.27$$

$$R_2 = S_R\left(\frac{D_2}{1 - B_l}\right) \quad 5.28$$

$$C_n = 1 - (1 + R_2) \times (1 - B_g) \quad 5.29$$

Logicamente não é necessário ter uma função S no caso em que $C_o = B_l$, pois o resultado é sempre $C_n = B_g$.

Outras funções poderiam ser usadas ao invés da função S, com as restrições:

- Para $x = 0$, a saída também tem que ser 0, para que ela seja contínua quando $P_o = B_l$;
- Para $x = 1$, a saída também tem que ser 1, senão haverá uma diminuição de contraste para valores altos;
- A função deve ser não decrescente;
- A função deve ser ímpar.

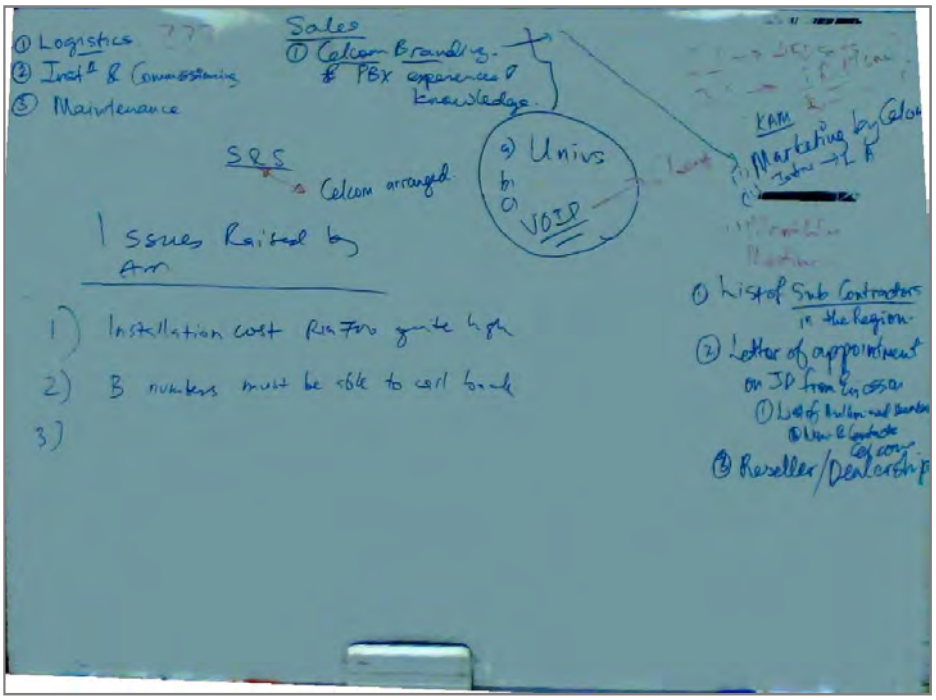
Segue abaixo o pseudocódigo que resume os passos do segundo algoritmo proposto:

```
1° passagem da imagem
Para cada bloco da imagem
  Calcular histograma do bloco
  Calcular a área ±10 (Δmoda) ao redor da moda do histograma da
componente
  Se todas as áreas computadas anteriormente excederem 75% (PERC_BL)
então o bloco atual é classificado como fundo, onde a cor de B1 é
igual as modas de cada componente

Remova blocos desconformes (mais detalhes na seção 5.3.2)

Calcule a média dos fundos locais

Ache a cor mais próxima na lista de fundos da média de fundos locais,
  essa cor vai ser o fundo global

2° passagem da imagem
Para cada ponto da imagem faça
  Co <= valor da componente do pixel
  B1 <= se o pixel está localizado dentro de um bloco de fundo então
use esse valor, senão procure o conjunto de blocos num raio mais
próximo e faça a média de cada componente desses blocos próximos
  caso Co = B1 então
    Cn <= Bg
  caso C < B1 então
```

```
    D1 <= Sd(Co - B1)
    R1 <= Sr(D1/B1)
    Cn <= (1 - R1)*Bg
  caso
    D2 <= Sd(B1 - Co)
    R2 <= Sr(D2/(1 - B1))
    Cn <= 1 - (1 + R2)*(1 - Bg)

ONDE:
  Bg (valor do fundo global),
  B1 (valor do fundo local),
  Co (valor do pixel original)
  Cn (novo valor do pixel)
```

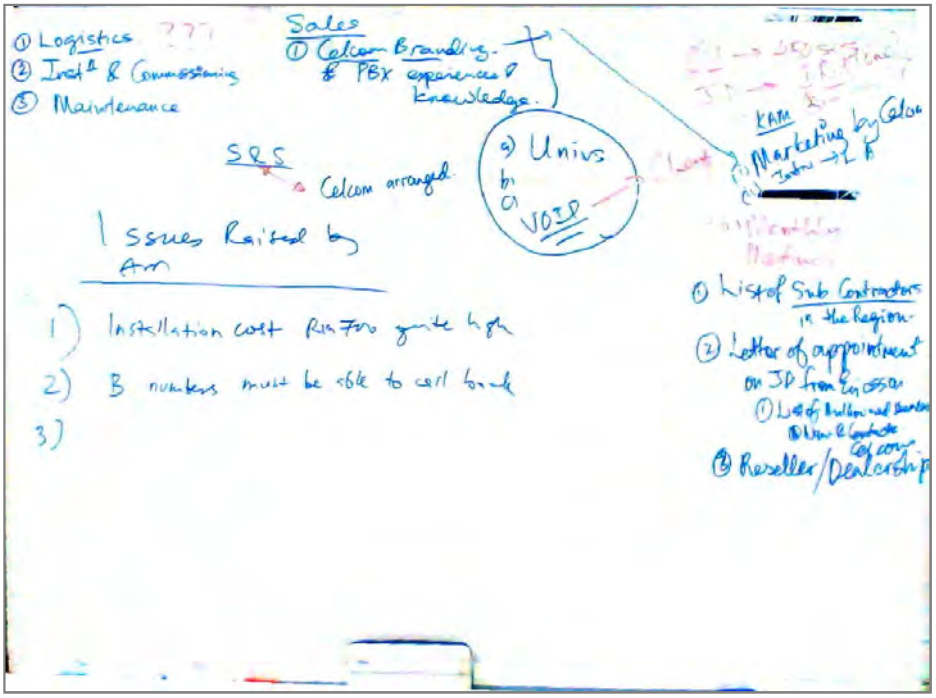
Pseudocódigo 5.2 – *Procedimento para realce de pontos da imagem.*

Alguns resultados da aplicação do segundo algoritmo proposto podem ser vistos a seguir. Para cada imagem é mostrada duas saídas: uma com o fundo inferido através das médias dos locais e outra com o fundo com uma cor mais próxima dessa média. Foram utilizados os valores de $P_d = 0{,}6$ e $P_r = 0{,}7$, pois estes valores reproduziram melhores resultados em geral.



(a) Imagem realçada com fundo global definido pela média dos locais

Figura 5.20 – *Aplicação do segundo algoritmo proposto na Figura 5.14(a)*



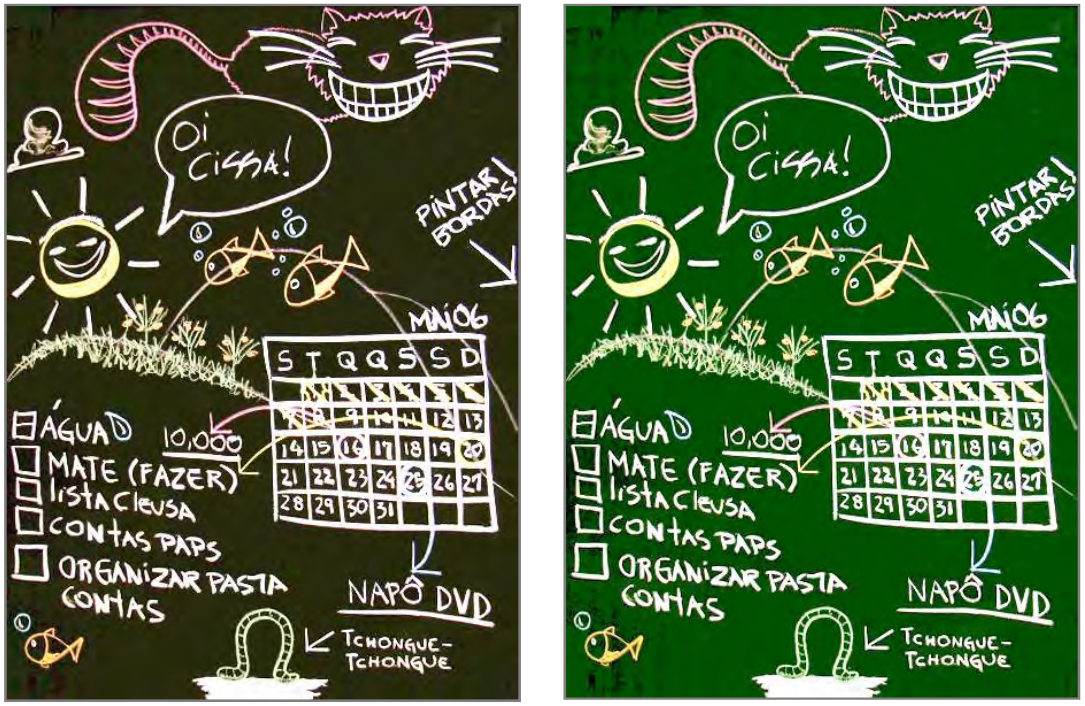
(b) Imagem realçada com o fundo global obtido com a cor mais próxima da lista

Figura 5.20 (cont.)

(a) Realce com o fundo global obtido pela média da lista

(b) Realce com o fundo global obtido pela cor mais próxima da lista

Figura 5.21 – *Aplicação do segundo algoritmo proposto na Figura 5.15*



(a) Realce com o fundo global obtido pela média da lista

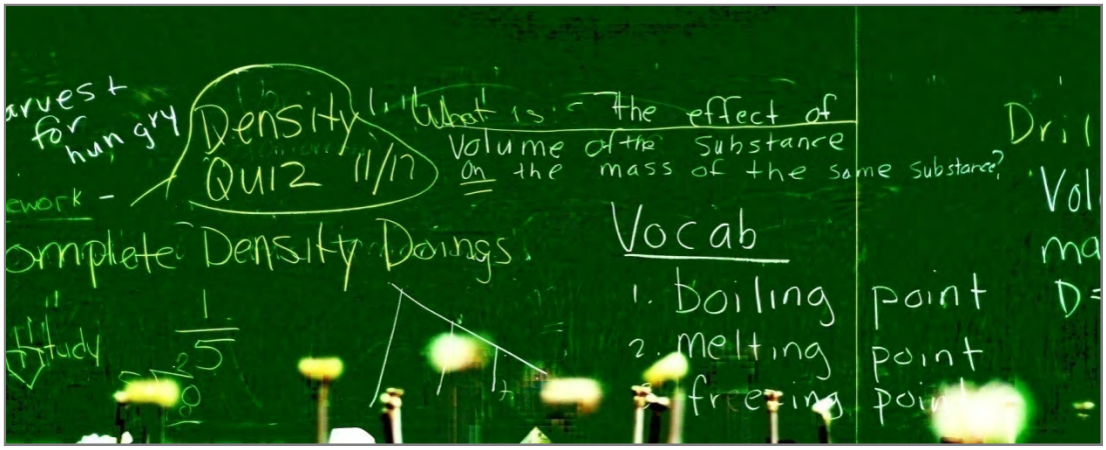
(b) Realce com o fundo global obtido pela cor mais próxima da lista

Figura 5.22 – *Aplicação do segundo algoritmo proposto na Figura 5.7*

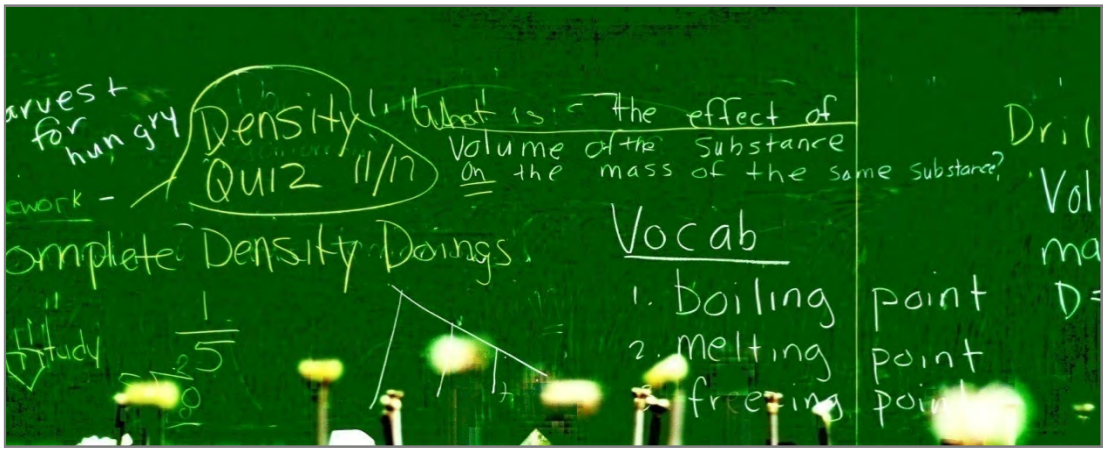
(a) Realce com o fundo global obtido pela média da lista

(b) Realce com o fundo global obtido pela cor mais próxima da lista

Figura 5.23 – *Aplicação do segundo algoritmo proposto na Figura 5.10 (a)*



(a) Realce com o fundo global obtido pela média da lista



(b) Realce com o fundo global obtido pela cor mais próxima da lista

Figura 5.24 – *Aplicação do segundo algoritmo proposto na Figura 5.8 (a)*

5.3.7 Parâmetros do algoritmo

Esta seção tem como objetivo listar todos os parâmetros definidos pelo algoritmo proposto de realce bem como a consequência da sua modificação conforme descrito na Tabela 5.4.

Tabela 5.4 – *Parâmetros definidos a segunda proposta de realce de quadros*

Parâmetro	Valor sugerido	Observações
Δmoda	10	Variação ao redor da moda para calcular o percentual de pixels que está presente no bloco afim de identificar se o bloco é fundo ou não. Quanto menor o valor mais restrito é a classificação de um bloco como fundo, podendo eliminar blocos que fazem parte do fundo, ao utilizar valores maiores pode-se classificar blocos como fundo onde não fazem parte do fundo.
PERC_BL	75%	Percentual mínimo de pixels ao redor da moda da imagem no bloco para definir se o bloco é candidato a fundo local ou não. Ao aumentar essa parâmetro, torna-o menos rigoroso na definição do bloco como fundo.
ARESTA_MAX	6	Tamanho máximo de todas as aresta da caixa formada pela comparação entre dois blocos afim de definir se estes pertencem ao mesmo conjunto de fundos locais ou não. Quanto menor esse valor mais restrito, é possível eliminar blocos pertencente ao fundo do quadro, equivocadamente, caso o valor seja muito grande é possível o aparecimento de blocos falso positivos.
k_L	1	Peso na diferença de cores com o padrão CIEDE2000 (Equação C.1) relativo a diferença entre luminâncias, da cor do fundo inferido e uma cor da lista de cores. Quanto menor o valor mais influente é essa diferença.
k_C	0,8	Peso na diferença de cores com o padrão CIEDE2000 (Equação C.1) relativo a diferença entre crominâncias (distância para o eixo cinza do espaço CIE L*a*b*), da cor do fundo inferido e uma cor da lista de cores. Quanto menor o valor mais influente é essa diferença.

Parâmetro	Valor sugerido	Observações
k_H	0,8	Peso na diferença de cores com o padrão CIEDE2000 (Equação C.1) relativo a diferença entre matizes, da cor do fundo inferido e uma cor da lista de cores. Quanto menor o valor mais influente é essa diferença.
P_d	0,6	Valor do parâmetro p na função S no cálculo da diferença. Valores pequenos aumenta-se o nível de realce, podendo-se ressaltar os ruídos do quadro, valores maiores pode-se eliminar ruídos formado pelos riscos remanescentes no quadro.
P_r	0,7	Valor do parâmetro p na função S no cálculo da razão. O uso de valores menores e maiores apresentam o mesmo resultado mostrado em P_d.

5.4 Resumo

Este capítulo teve o intuito de revelar os algoritmos de realce para quadros didáticos. Foi apresentado a proposta de realce do WhiteboardIt, feita para quadros brancos, ela apresenta falha ao não especificar como remover blocos desconformes e não conseguir realçar áreas onde o reflexo da luz é muito evidente.

A primeira proposta do Tableau para o realce visa eliminar tal problema, como demonstrado neste capítulo. Já a segunda proposta apresenta uma característica inédita na literatura técnica que é realçar quadros com qualquer cor de fundo. Além desta característica, apresenta resultados similares ou superiores ao ser comparado com os outros sistemas, o que será visto no próximo capítulo.

6 Comparando Tableau com Ambientes Correlatos

Este capítulo tem por intuito descrever e comparar os ambientes existentes para processamento de quadros didáticos. Os sistemas se subdividem em: execução local e serviços *on-line* (disponível na Internet). Os de execução local fazem o processamento na própria máquina do usuário. Os serviços *on-line* permitem que os usuários enviem as imagens dos quadros pela Internet para que a empresa fornecedora processe-as colocando o resultado no seu site, sendo necessário o cadastro prévio do usuário para ter o serviço.

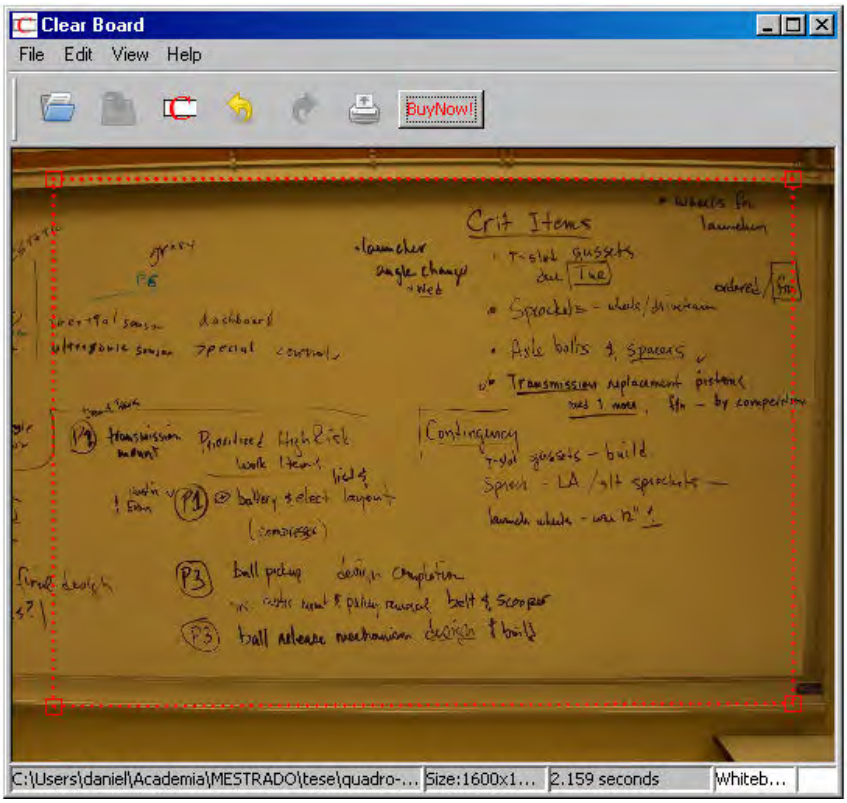
6.1 Ambientes com execução local

Dois ambientes de execução local estão disponíveis atualmente no mercado. O ClearBoard (SOFTTOUCH, 2008) e o Whiteboard Photo (POLYVISION, 2008).

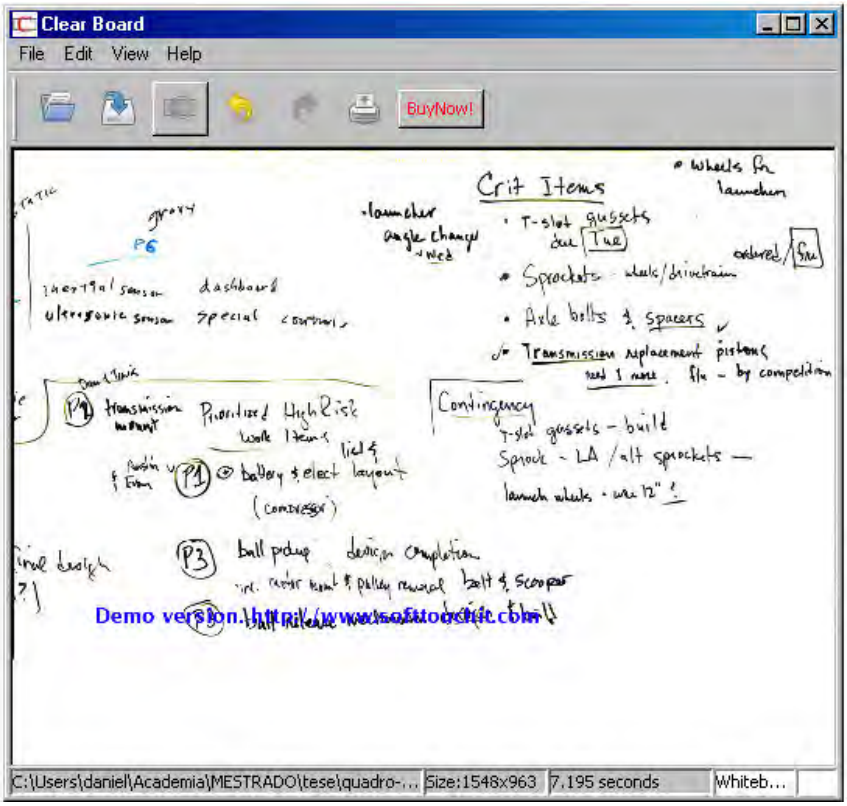
Outro produto que se destaca é o WhiteboardIt, que contém os algoritmos descritos nessa dissertação, no entanto não é possível executá-lo pois é um projeto de pesquisa da Microsoft e não está disponível para ser baixado publicamente. Os diferenciais dele, segundo (LIU *et al*, 2007; ZHANG and TAN, 2001; ZHANG and HE, 2004; ZHANG *et al*, 2006), são: correção de perspectiva; agrupar imagens do mesmo quadro em uma única imagem; realçar imagem; detecção de borda.

6.1.1 ClearBoard

O ClearBoard é um sistema comercial desenvolvido pela empresa australiana SofttouchIt. No dia 15 de novembro, o sistema estava custando 246,20 reais para uso definitivo. No ClearBoard, ao carregar a imagem, inicia-se a detecção de borda do conteúdo da imagem, como pode ser visto na Figura 6.1 (a). Uma vez a borda detectada pode ser ajustada pelo usuário. Uma vez definida a região, o usuário requisita para "limpar" a imagem onde é feita a correção de perspectiva seguida do realce do quadro. Também é possível selecionar a cor do fundo do quadro, no entanto o fundo de saída é sempre branco, sendo útil para a impressão do conteúdo do quadro. Um exemplo de imagem processada pode ser visto Figura 6.1 (b). A versão analisada nesse estudo é a 2.0.2 de demonstração, liberada no dia 19 de fevereiro de 2007, tendo como única diferença em relação à versão paga, a inscrição na imagem resultante informando que é a versão demonstração com o endereço para o site da empresa.



(a) Após o carregamento da imagem

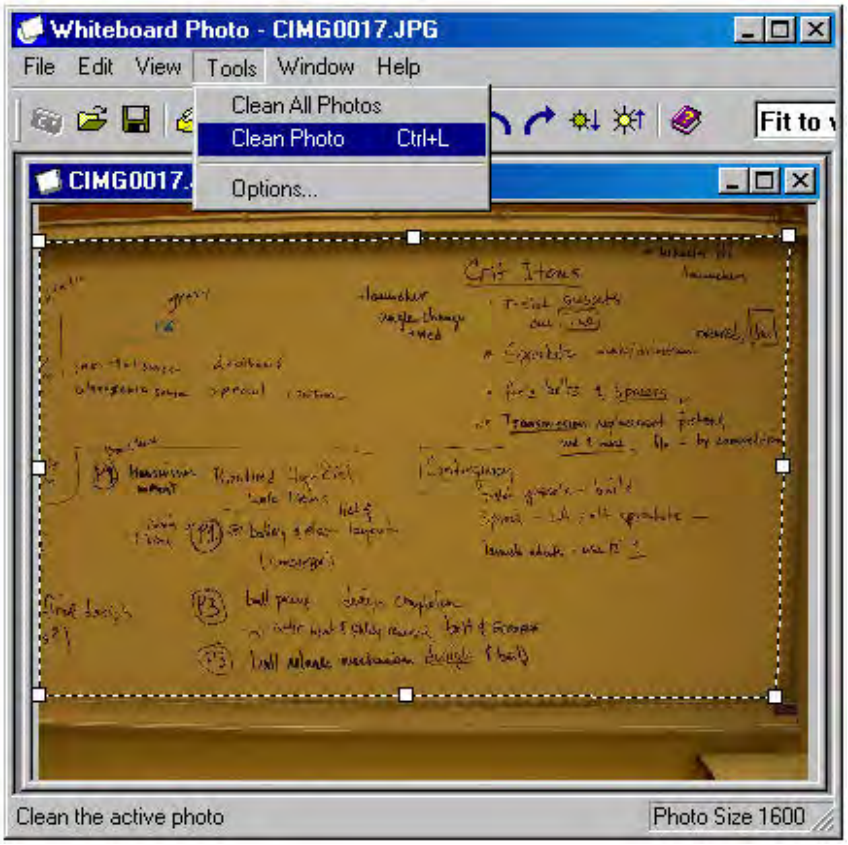


(b) Após processamento da imagem

Figura 6.1 – *Telas do ClearBoard*

6.1.2 Whiteboard Photo

O Whiteboard Photo foi desenvolvido pela empresa americana, PolyVision Corp. Esse sistema também detecta a borda ao carregar a imagem, como pode ser visto na Figura 6.2 (a). Além de definir as quatro quinas do quadro o usuário pode ajustar ponto intermediário entre duas quinas, que é útil na correção da distorção da lente. Ao terminar o ajuste o usuário pode requisitar a "limpeza" da imagem, onde esse sistema corrige a distorção da lente e a perspectiva, seguido do realce da imagem. O sistema fornece a opção de escolha do fundo do quadro, também, no entanto para quadros verdes e pretos são geradas imagens com fundo preto "puro". A tela com um exemplo de imagem processada pode ser vista na Figura 6.2 (b). Foi utilizada nesse estudo a versão de demonstração 1.3.0.5, não sendo informada a data de liberação da versão.



(a) Depois de adicionar a imagem

Figura 6.2 – *Interface do Whiteboard Photo*

(b) Imagem processada

Figura 6.2 (cont.)

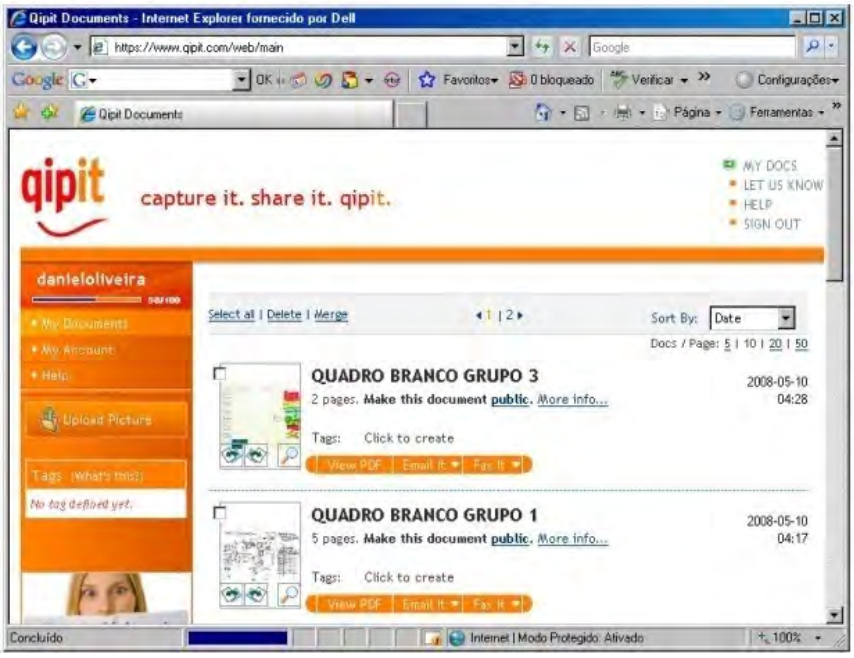
6.2 Serviços *on-line* na Internet

Em alternativa aos ambientes de execução local, existem os serviços *on-line* na Internet, que possuem um pacote gratuito. Os únicos sistemas existentes, até onde o autor desse estudo saiba é o QipIt® e ScanR®.

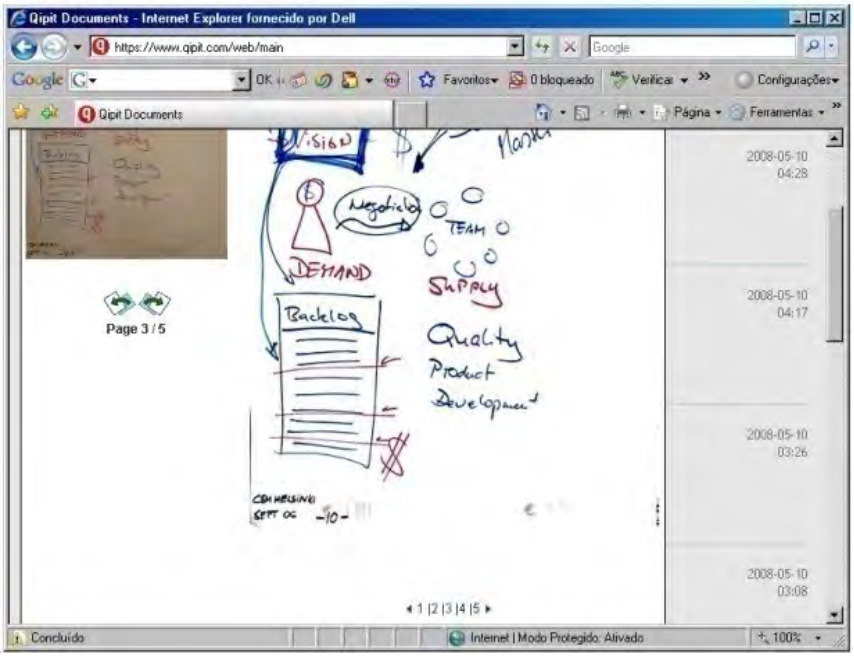
6.2.1 QipIt ®

O QipIt® (REALEYES3D, 2008) é um sistema mantido pela empresa francesa RealEyes3D. Ele permite que usuário envie as fotos por e-mail ou pelo próprio site. Para fazer isso é necessário criar uma conta com nome, e-mail e senha. Ao receber as imagens por e-mail o sistema responde a mensagem com o resultado do processamento em um arquivo anexado no formato PDF. É possível também enviar fax das imagens armazenadas

no sistema. Na opção gratuita é possível enviar até 5 faxes por semana e armazenar até 100 fotos no sistema. As restrições são: não trabalhar com imagens pequenas, mas no seu portal não é informado a resolução mínima necessária; só processar imagens de quadros brancos. A página inicial do sistema e a parte em que é possível visualizar a imagem, podem ser vista na Figura 6.3.



(a) Página inicial

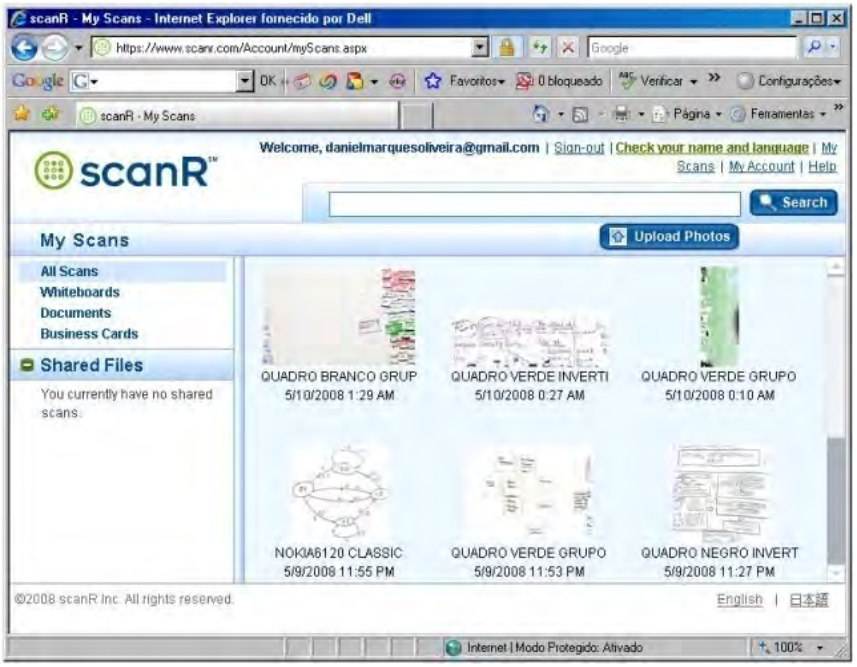


(b) Página de visualização de um grupo de imagens

Figura 6.3 – *Interface do Qipit*

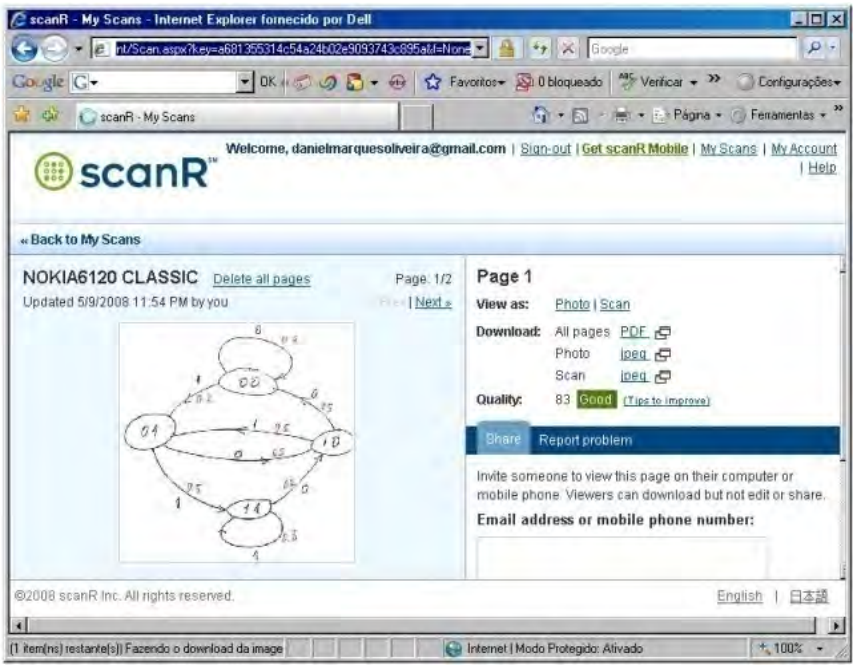
6.2.2 ScanR ®

O ScanR é um serviço que concorre diretamente com o QipIt®. Ele possui as mesmas funcionalidades do QipIt como: envio de fotos por e-mail e pelo próprio site; recebimento das fotos resultantes por e-mail em um arquivo PDF; envio de fax. No entanto, observou-se no dia 28 de janeiro de 2009, a versão gratuita ficou restrita ao envio de uma única imagem para ser processada. Ele também possui a restrição de trabalhar somente com imagens de quadros brancos e com resolução mínima de 1100x800 *pixels*. A interface do ScanR com a página inicial e a tela que contém um grupo de imagens pode na **Figura 6.4**.



(a) Página inicial

Figura 6.4 – *Interface do ScanR*



(b) Página com um grupo de imagens

Figura 6.4 (cont.)

6.3 Comparação entre os ambientes

Esta seção tem o intuito de fazer comparações entre os ambientes supracitados. O primeiro aspecto diz respeito à detecção de borda, sem correção de perspectiva e realce. A segunda comparação leva em consideração a qualidade do realce, onde as imagens de entrada são as imagens pré-cortadas. A última comparação é uma análise geral dos sistemas. As comparações foram feitas num *notebook* Dell 531L, com processador AMD Turion X2 TL-50 com 3 Gb de memória RAM no Windows Vista Business.

Todas as imagens de teste, bem com os resultados do processamento delas utilizando Tableau e os ambientes aqui mencionados, podem se encontradas no DVD em anexo a esta dissertação.

6.3.1 Conjunto de testes

Para testar os sistemas foram separadas em dois conjuntos de imagens diferentes. Um conjunto de 23 imagens para testar a detecção de borda, e 46 para testar o realce, sendo algumas destas imagens comuns aos dois conjuntos. A base completa contém 278 imagens, porém há muitas imagens parecidas, especialmente as imagens geradas pelo autor, pois se

utilizou apenas três modelos de câmeras e as salas de aula apresentam características similares.

Com relação aos serviços on-line foram enviadas as fotos para os dois serviços online a partir de uma conta de e-mail do Gmail® (http://www.gmail.com). O primeiro grupo foi enviado no dia 13 de setembro de 2008 entre 15 e 16 horas, e as fotos foram separadas em cincos subgrupos:

- Três subgrupos de imagens de quadro branco, um contendo três fotos totalizando 3,38Mb, um com cinco imagens totalizando 851kb, um com sete imagens totalizando 1,78Mb;
- Um subgrupo de imagens de quadro negro contendo três fotos totalizando 1,32Mb;
- Um subgrupo de imagens de quadro verde contendo cinco fotos totalizando 2,09Mb.

O segundo grupo foi enviado em maio de 2008, a primeira parte no dia 10, entre 12 e 15 horas; a segunda parte no dia 9 entre 14 e 15 horas. As fotos foram separadas em apenas três subgrupos:

- Um subgrupo com 16 imagens de quadro branco
- Um subgrupo com 10 imagens de quadro verde
- Um subgrupo com 20 imagens de quadro negro

Mesmo o segundo grupo tendo sido enviado bem antes do primeiro, como os serviços on-line fazem todo o processamento, com detecção de borda e realce, não foi notada nenhuma diferença no realce da imagem, não sendo necessário o re-envio do segundo grupo.

6.3.2 Detecção de borda

A comparação entre os mecanismos de detecção de borda foi feita através de inspeção visual. Ao observar as imagens classificam-se as imagens em: detecção correta (DC), detecção correta parcial (DCP) e detecção incorreta (DI). A detecção correta ocorre quando o sistema definiu como borda o limite da área de conteúdo, não incluindo a moldura. A detecção correta parcial ocorre quando pelo menos duas bordas (superior, inferior, direita, esquerda) foram detectadas corretamente sem cortar parte da área do conteúdo. Caso contrário a borda é classificada como detectada incorretamente.

Nota-se que na Tabela 6.1, o total de imagens do ScanR® e do QipIt® não são coerentes com os outros sistemas, pois o ScanR® não conseguiu processar todas imagens devido a restrição de tamanho do quadro e o QipIt® não processou nenhuma de imagem de quadro verde. Para a primeira proposta do Tableau da detecção de borda foi utilizada a notação Tableau CBDAR 2007, já para a segunda nomeou-se como Tableau ICIAR 2008.

Tabela 6.1 – *Comparativo entre detecções de bordas entre os sistema de processamento de quadros didáticos*

Algoritmo	Quadro branco			Quadro preto			Quadro verde			Acertos
	DC	DCP	DI	DC	DCP	DI	DC	DCP	DI	
ClearBoard	1	2	12	0	2	1	0	2	3	1 (4%)
WBPhoto	7	2	6	2	0	1	1	1	3	10 (44%)
QipIt*	2	4	9	2	0	1	0	0	0	4 (17%)
ScanR**	0	4	5	0	0	1	0	0	2	0 (0%)
Tableau CBDAR2007	6	4	5	2	0	1	0	0	5	8 (35%)
Tableau ICIAR2008	8	5	2	2	1	0	2	2	1	12 (52%)

* O QipIt informou que ocorreu um erro no processamento das imagens de quadros verdes

** O ScanR não conseguiu processar 6 imagens de quadro branco, 1 de quadro negro e 3 de quadro verde

Dentre os sistemas apresentados a versão mais recente do Tableau apresentou a melhor taxa de acertos, de 52%, seguido do Whiteboard Photo. Nas figuras a seguir podem ser vistas algumas das imagens usadas. Como não é possível obter a imagem intermediária no processamento do QipIt® e ScanR® estimou-se a detecção de borda através da inspeção visual da imagem resultante. Os pontos verdes, amarelos e cianos no Whiteboard Photo e ClearBoard indicam as quinas do quadrilátero da borda do quadro, bem como o ponto intermediário da curvatura da lente no Whiteboard Photo.

No caso de imagens de quadros negros, nota-se também que na imagem do Tableau CBDAR 2007 o processamento não conseguiu detectar a borda, definindo como fronteira do quadro a borda da imagem, o que não elimina o conteúdo do quadro.

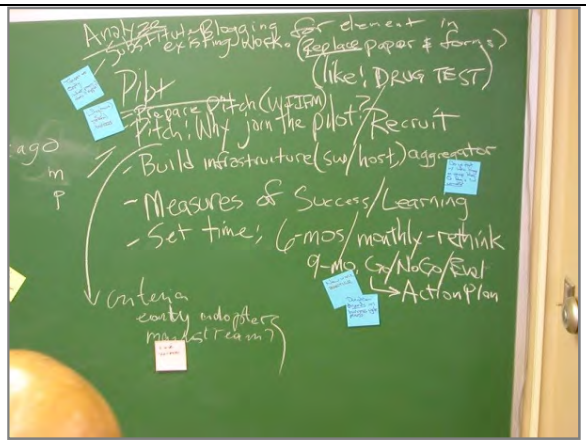
Tabela 6.2 – *Imagens de quadros pretos com bordas detectadas*

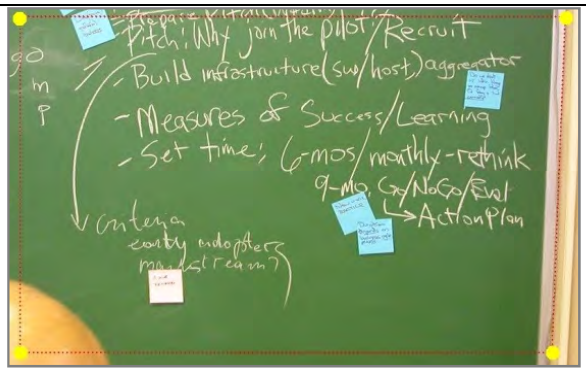
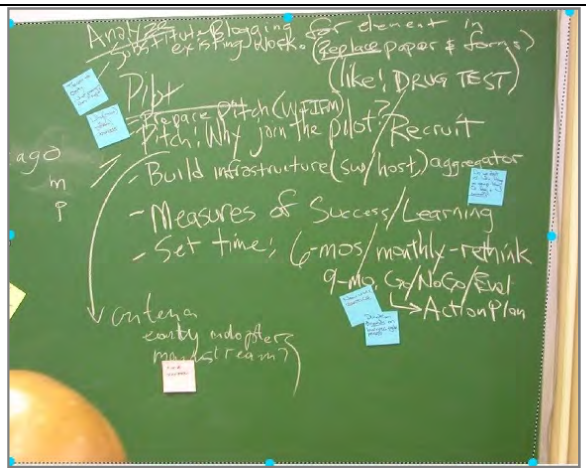
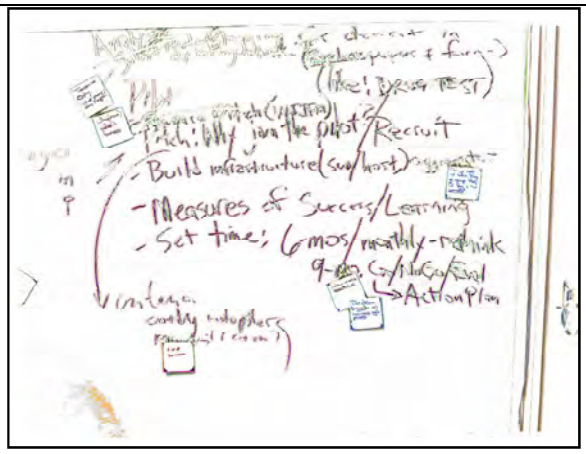
Algoritmo	Imagem A	Classi.	Imagem B	Classi.
Imagem original		-		-
ClearBoard		DCP		DI
Whiteboard Photo		DC		DI
QipIt		DCP		DC

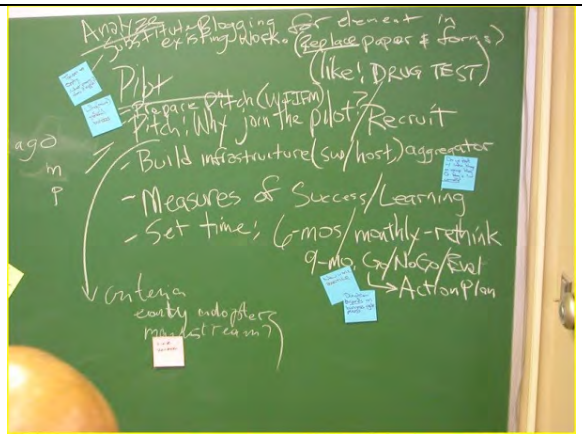
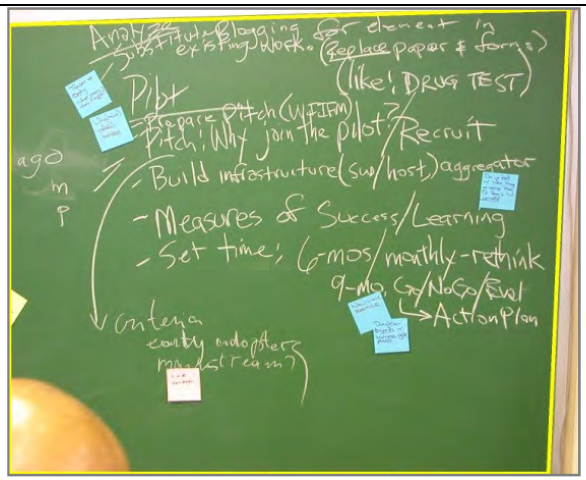
Algoritmo	Imagem A	Classi.	Imagem B	Classi.
Tableau CBDAR2007		DC		DI
Tableau ICIAR2008		DC		DCP

Logo abaixo pode ser visto outro resultado com um quadro verde. Dentre os resultados apresentados, só o Tableau ICIAR 2008 conseguiu detectar a borda corretamente.

Tabela 6.3 – *Imagem de quadro verde com bordas detectadas*

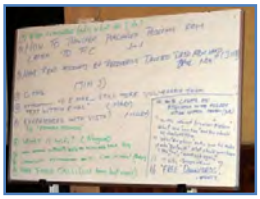
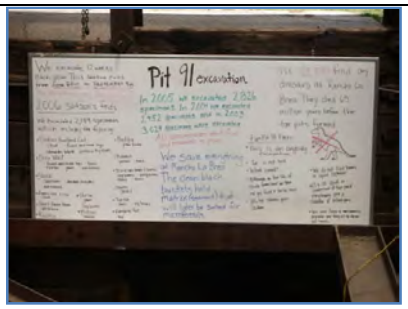
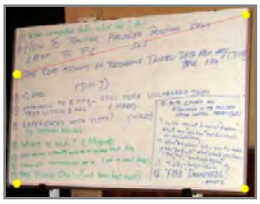
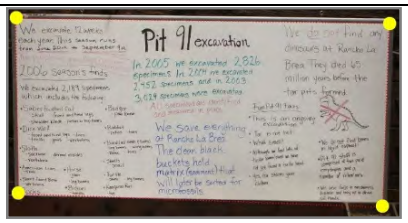
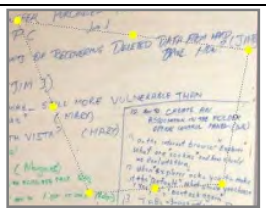
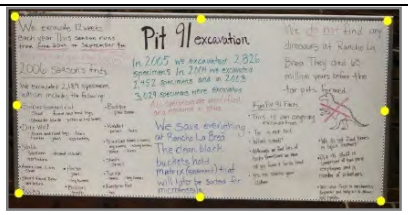
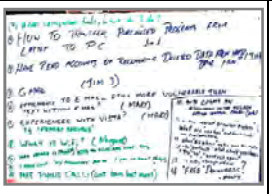
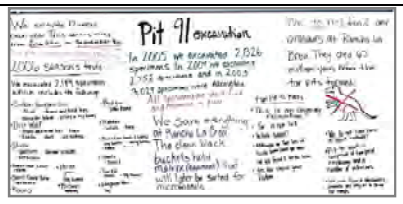
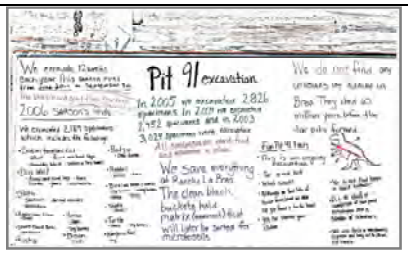
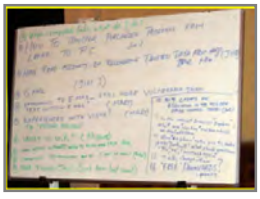
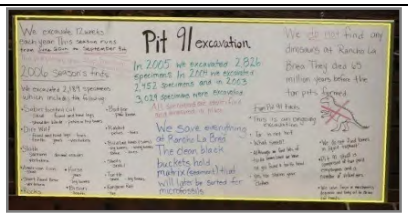
Algoritmo	Imagem C	Classi.
Imagem original		-

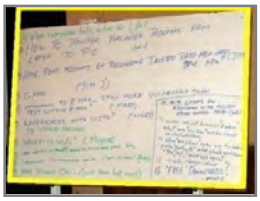
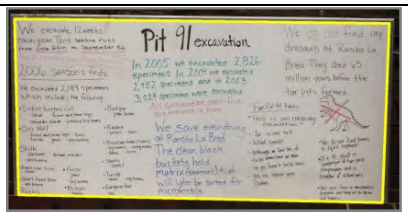
Algoritmo	Imagem C	Classi.
ClearBoard		DI
Whiteboard Photo		DCP
ScanR		DI

Algoritmo	Imagem C	Classi.
Tableau CBDAR2007		DI
Tableau ICIAR2008		DC

Já para imagens de quadros brancos, o resultado pode ser visto logo abaixo. A borda da imagem E só não foi detectada corretamente pelo ScanR, que não conseguiu achar a borda superior. Já no caso da imagem "D", o ScanR não a processou por ser é menor que 1100×800 pixels. O único sistema que conseguiu detectar a borda corretamente dessa imagem foi o QipIt®. O Tableau ICIAR 2008 só não detectou a borda superior.

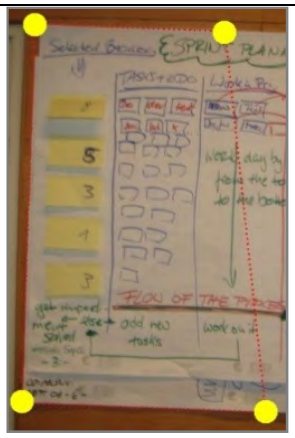
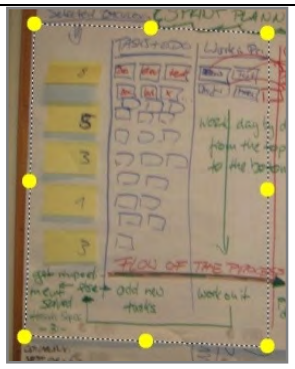
Tabela 6.4 – *Imagens de quadros pretos com bordas detectadas*

Algoritmo	Imagem D	Clas.	Imagem E	Clas.
Imagem original		-		-
ClearBoard		DCP		DC
WBPhoto		DI		DC
QipIt		DCP		DC
ScanR	Menor que 1100 × 800	-		DCP
Tableau CBDAR2007		DI		DC

Algoritmo	Imagem D	Clas.	Imagem E	Clas.
Tableau ICIAR2008		DCP		DC

Com as imagens apresentadas anteriormente é possível notar a evolução do Tableau com relação à detecção de bordas, ao se comparar a versão para o ICIAR 2008 e CBDAR 2007. No entanto, o Tableau ICIAR 2008 apresenta algumas limitações caso a imagem do quadro apresente riscos grossos, se as bordas estiverem muito inclinadas devido à perspectiva do quadro e se o quadro estiver muito sujo. Duas dessas imagens podem ser vistas na tabela as seguir, onde a imagem F índica Figura 5.7 (a) e a imagem G a Figura 5.10 (a). Nesses casos alguns dos outros sistemas conseguem superar o Tableau, que é o caso do Whiteboard Photo na imagem F.

Tabela 6.5 – *Imagens de quadros verdas e pretos com detecção errada pelo Tableau*

Algoritmo	Imagem F	Clas.	Imagem G	Clas.
ClearBoard		DI		DI
WBPhoto		DC		DI
Tableau ICIAR 2008		DI		DI

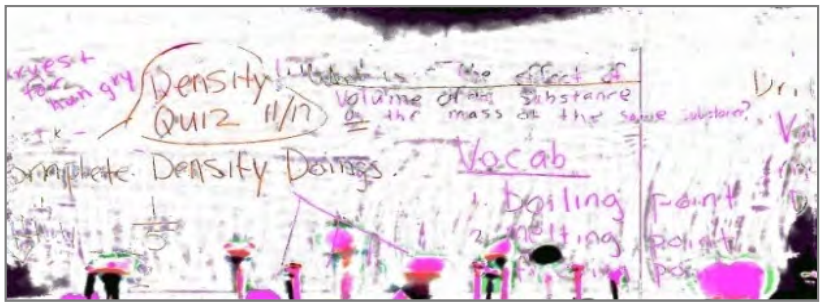
Vale ressaltar que em Tableau, o ambiente foi projetado para que o usuário possa corrigir facilmente a detecção não adequada de bordas e que a detecção automática funciona como sugestão para a qual o usuário deve confirmar a área de corte.

6.3.3 Realce da imagem

Esta seção visa fazer uma análise comparativa entre os realces dos diversos sistemas aqui apresentados. No caso do Tableau, dois algoritmos foram propostos. A primeira proposta do realce será referenciada nesta seção como Tableau-WBIT, pois aplica alguns

conceitos propostos pelo WhiteboardIt. Já a segunda proposta será referenciada como Tableau-II.

A Figura 6.5 apresenta os resultados do processamento de imagens de quadros verdes das Figura 5.8 e Figura 5.7 dos serviços on-line. É visível que os resultados não são satisfatórios, piorando as imagens originais. No entanto esses resultados já eram esperados, pois as próprias ferramentas se propõem, exclusivamente, a processar imagens de quadros brancos. O ScanR não processou a Figura 5.7, pois esta é menor que 1100×800 *pixels*.



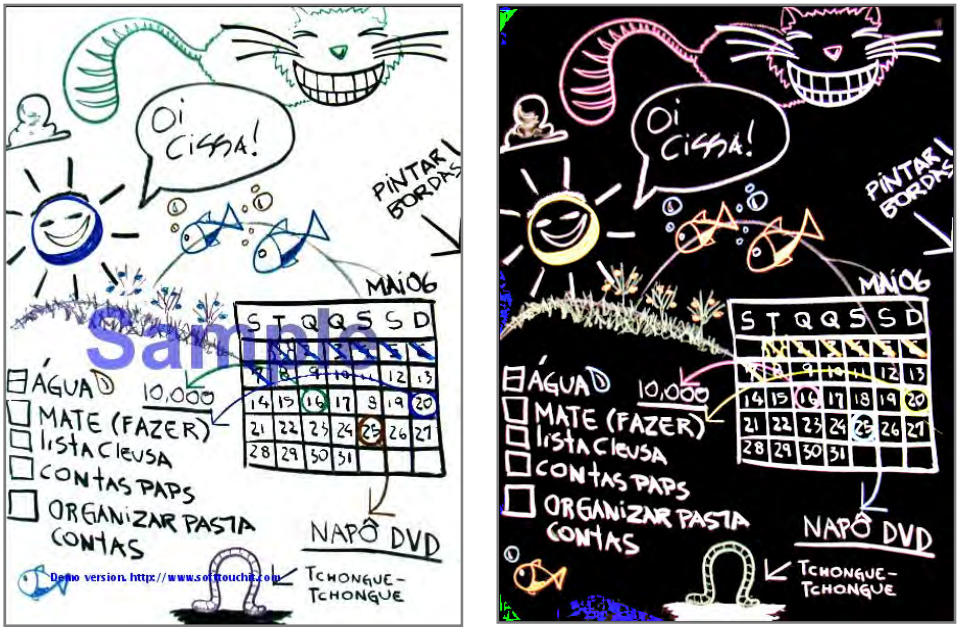
(a) ScanR®

(b) Qipit®

Figura 6.5 – *Resultado do processamento da Figura 5.8 (a)*

Figura 6.6 – *Resultado do processamento da Figura 5.7 com o Qipit® (c)*

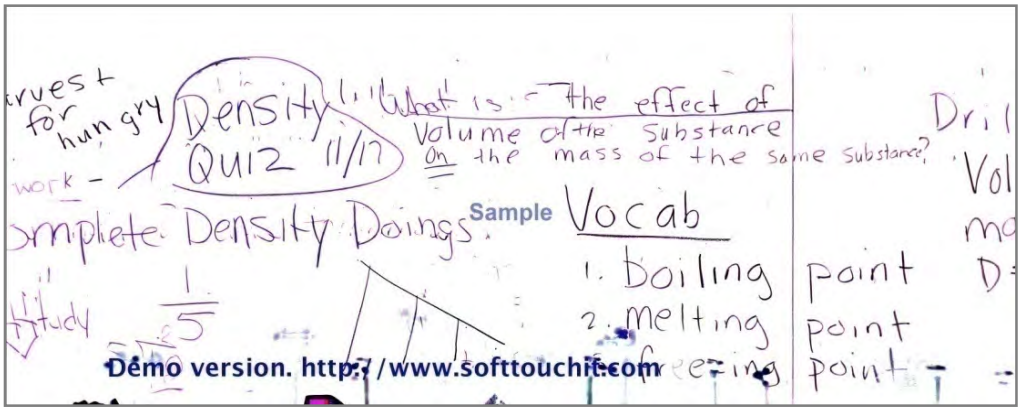
Já os resultados do processamento nos sistemas de execução local das mesmas imagens de quadros verdes da Figura 5.7 e da Figura 5.8(a) podem ser vistos na Figura 6.7 e na Figura 6.8, respectivamente. Ambos os resultados tornaram as imagens mais claras e bordas bem definidas. A diferença principal entre os dois é o fundo resultante, do ClearBoard é branco – útil para impressão – e do Whiteboard Photo é preto, mesmo para quadros verdes.



(a) ClearBoard®

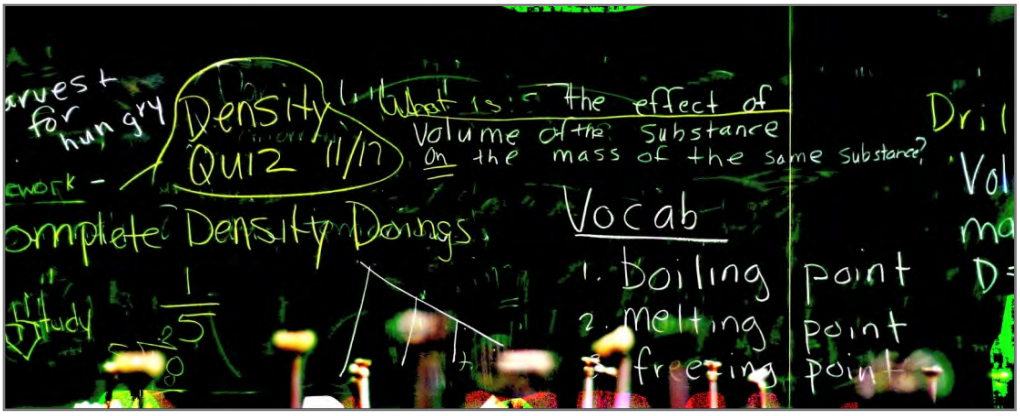
(b) Whiteboard Photo®

Figura 6.7 – *Resultado do processamento da Figura 5.7*



(a) ClearBoard®

Figura 6.8 – *Resultado do processamento da Figura 5.8 (a)*



(b) Whiteboard Photo®

Figura 6.8 (cont.)

Os resultados dessas imagens da segunda proposta do algoritmo de realce do Tableau podem ser visto nas figuras Figura 5.22 e Figura 5.24 do capítulo anterior. O único sistema que apresentou um resultado um pouco melhor foi o ClearBoard® na Figura 5.8. Ao utilizar os valores de P_d e P_r, igual a 1,00 e 0,75, respectivamente, para a mais recente proposta do Tableau obtem-se a imagem abaixo. Essa imagem é de qualidade semelhante quando à obtida pelo ClearBoard®.

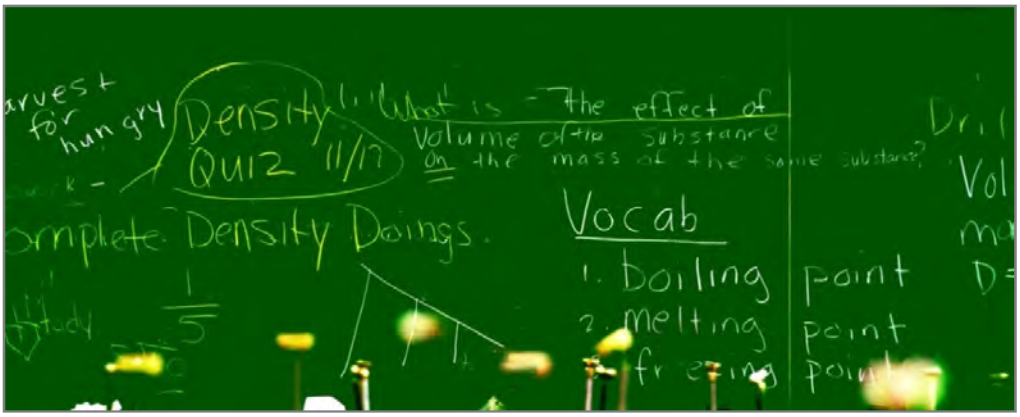


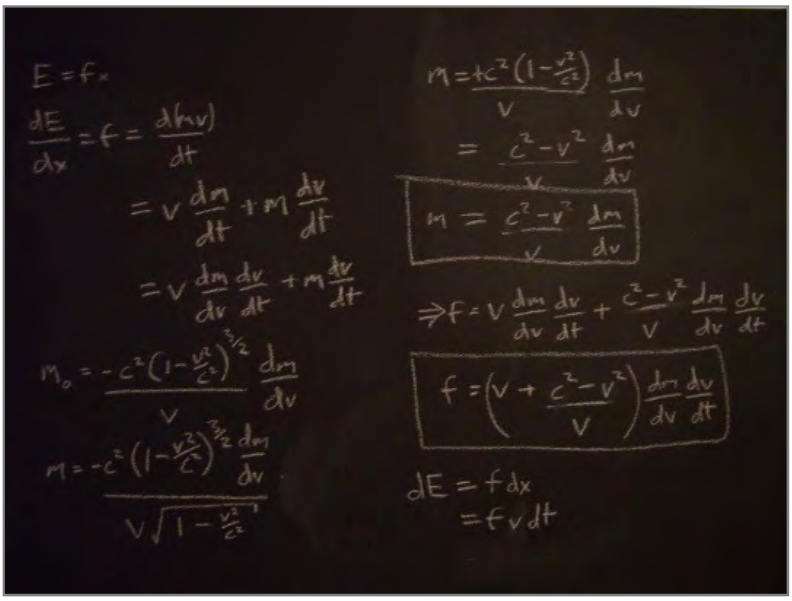
Figura 6.9 – *Resultado do processamento da Figura 5.8 com o Tableau-II utilizando valores dos parâmetros* $\boldsymbol{P_d = 1,00}$ *e* $\boldsymbol{P_r = 0,75}$

No caso dos quadros pretos, o resultado do processamento da Figura 5.15 pode ser visto na Figura 6.10 e Figura 6.11. Observe-se que os riscos remanescentes foram realçados equivocadamente nas figuras processadas pelo ClearBoard – Figura 6.10 (a) – e Whiteboard Photo – Figura 6.10 (b). Esse realce indesejado aconteceu também no processamento do Tableau, que pode ser visto na Figura 5.19 (a). Esses riscos foram eliminados utilizando-se parâmetros mais "agressivos", sendo visível na Figura 5.19 (b).

Outro exemplo de imagem de processamento de quadro negro pode ser visto na e Figura 6.11 onde todos os algoritmos tiveram resultados similares.

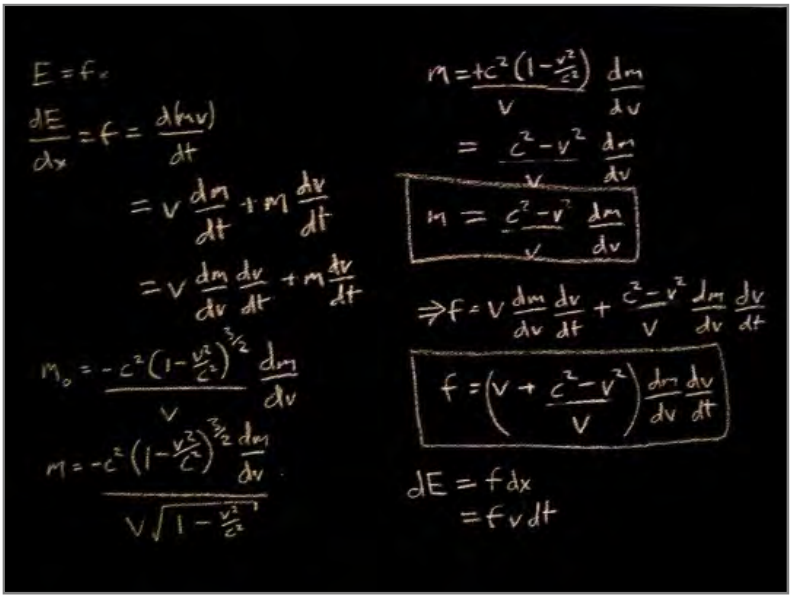
(a) ClearBoard® (b) Whiteboard Photo®

Figura 6.10 – *Resultado do processamento da Figura 5.7 (a)*

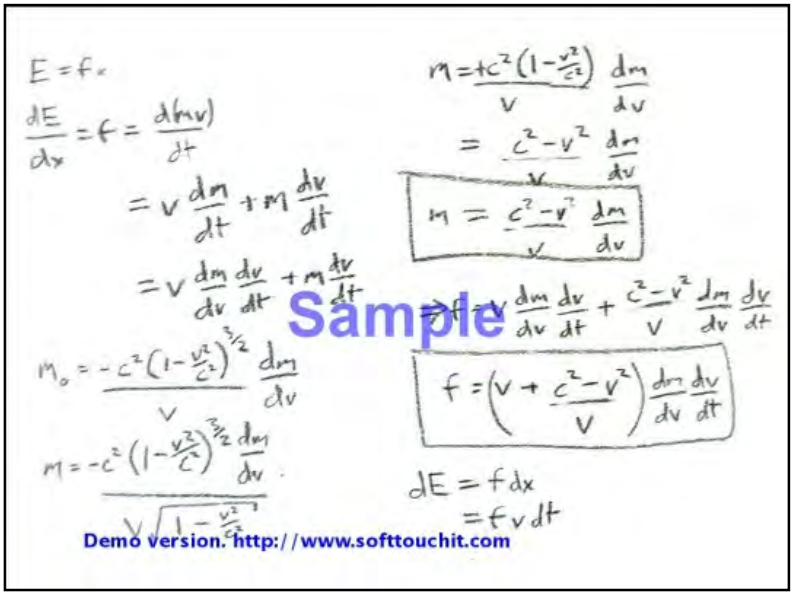


(a) Imagem original

Figura 6.11 – *Processamento de quadro negro*

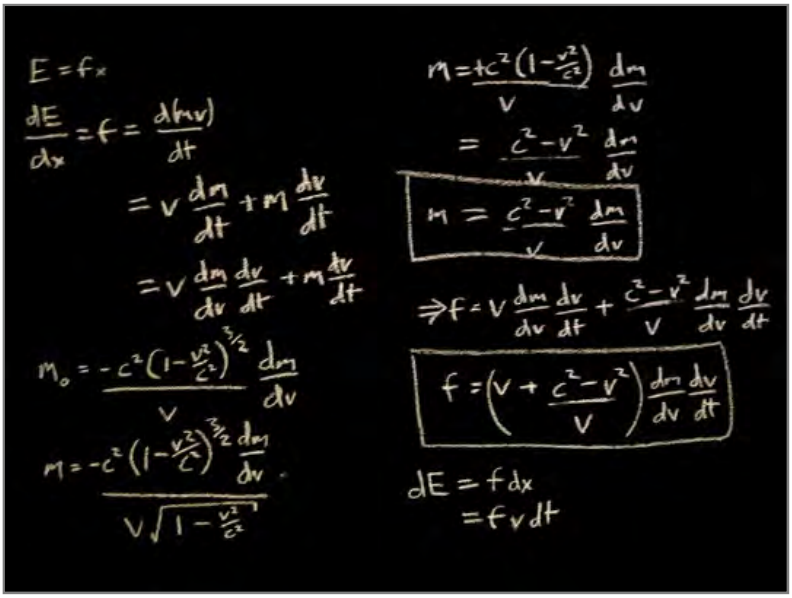


(b) Resultado com segunda proposta de realce do Tableau



(c) ClearBoard®

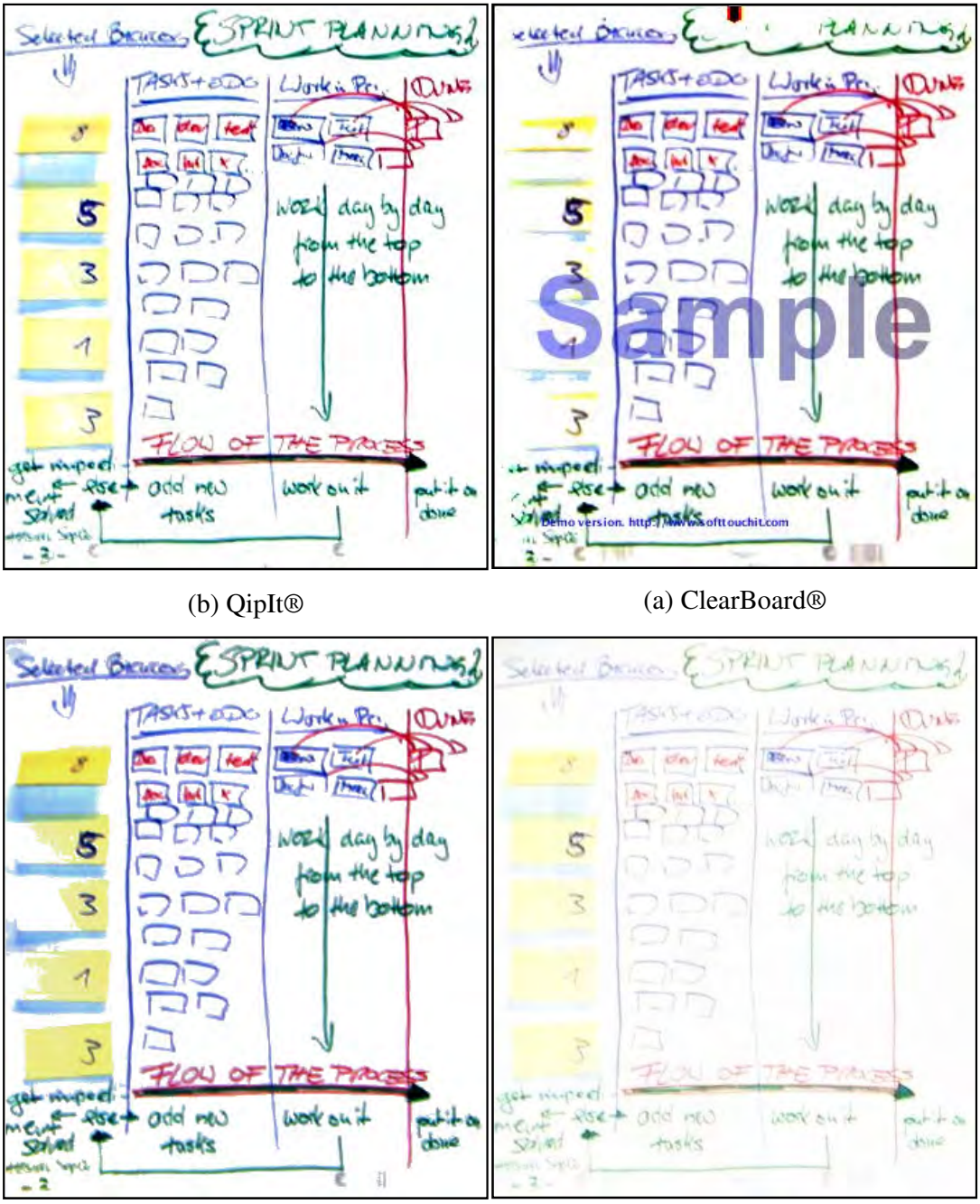
Figura 6.11 (cont.)



(d) Whiteboard Photo®

Figura 6.11 (cont.)

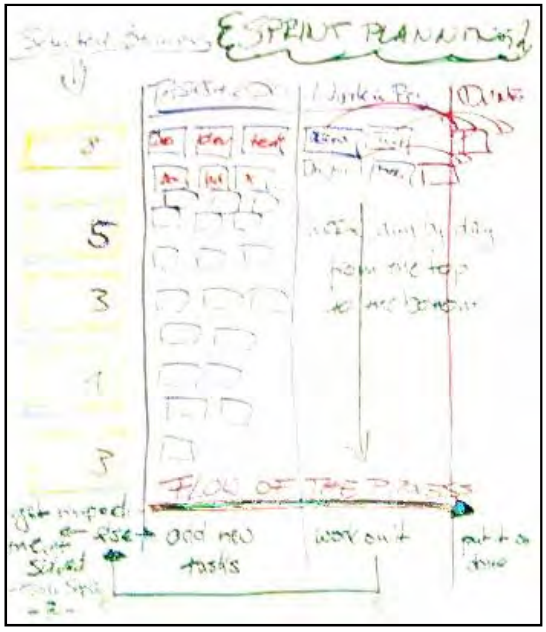
Agora os resultados obtidos com as duas propostas do Tableau e as fotos de quadros brancos serão analisados e comparados com as ferramentas existentes. A primeira análise é relativa à aplicação na Figura 5.10 (a). O ScanR® rejeitou essa imagem, pois ela era menor que o mínimo permitido, com dimensões 349x405 *pixels*. A Figura 6.12 (a) mostra o resultado do algoritmo pelo QipIt®, onde o único defeito é que o centro do *Postit*® ficou esbranquiçado. A Figura 6.12 (b) ilustra o resultado do algoritmo utilizando o ClearBoard, seguido do Whiteboard Photo. Ambos apresentaram resultados bem parecidos, onde o *Postit*® ficou um pouco apagado. Já Figura 6.12 (d) foi utilizado o WhiteboardIt que apresentou a imagem resultante com um contraste muito baixo. E por fim, a Figura 6.12 (f) que contém o resultado do Tableau-WBIT, esse resultado não é tão bom devido aos riscos serem muito grossos. Já o resultado da aplicação da segunda proposta do Tableau para o realce apresentou o melhor desempenho para essa imagem como pode ser visto na Figura 5.23.



(b) QipIt® (a) ClearBoard®

(c) Whiteboard Photo® (d) WhiteboardIt

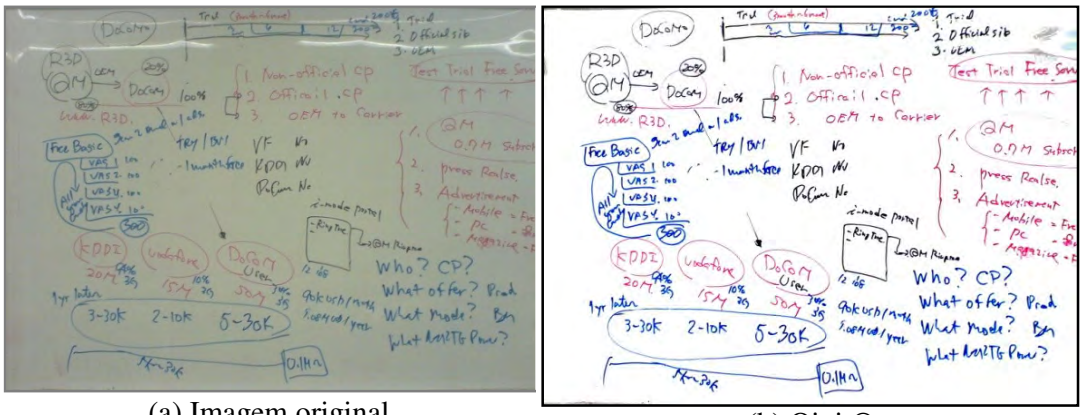
Figura 6.12 – *Processamento da Figura 5.10 (a)*



(e) Tableau-WBIT

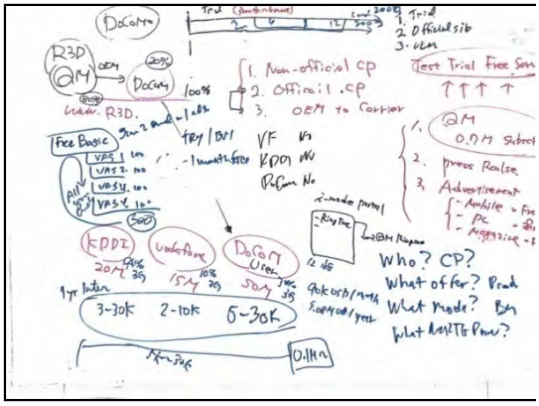
Figura 6.12 (cont.)

Outro exemplo de quadro branco pode ser visto logo a seguir. É possível notar que todas as imagens apresentaram um desempenho similar.

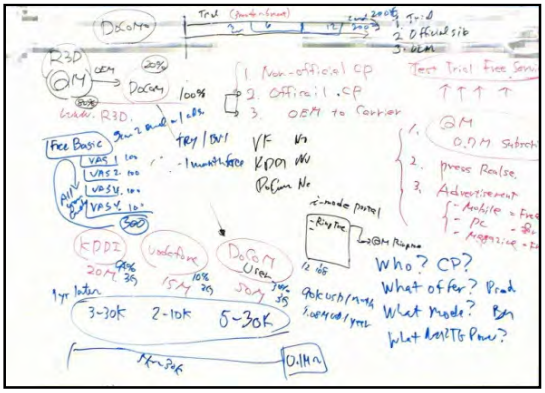


(a) Imagem original (b) Qipit®

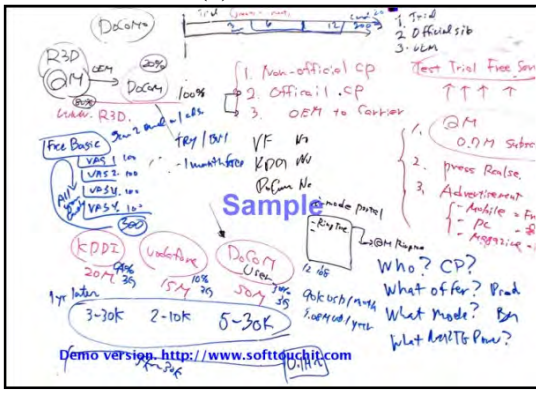
Figura 6.13 – *Processamento de um quadro branco*



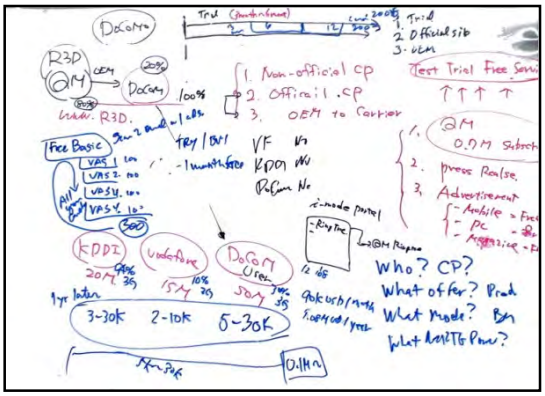
(c) ScanR®



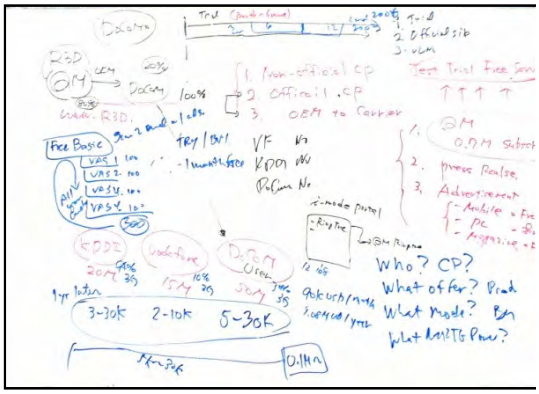
(d) WhiteboardIt®



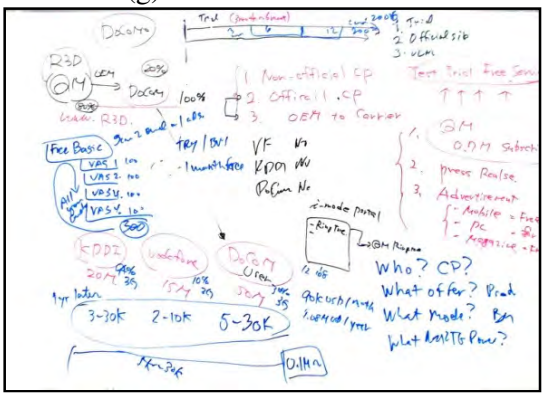
(f) ClearBoard ®



(g) Whiteboard Photo ®



(h) Tableau-WBIT



(i) Tableau-II

Figura 6.13 (cont.)

As imagens mostradas acima evidenciam que o realce do Tableau, especialmente a segunda proposta, tem desempenho similar e em alguns casos melhores que os outros sistemas, tanto os serviços on-line que são projetados para trabalhar em quadros brancos, como nas duas ferramentas de execução local.

6.3.4 Comparação geral

Nesta seção será feita uma comparação geral entre os sistemas de processamento de quadros didáticos. Primeiro deve-se diferenciar os sistemas de execução local e os serviços *on-line* via Internet.

A vantagem dos sistemas de execução local é pela resposta ser mais rápida, pois o processamento é local, não necessitando a transferência da imagem. No segundo é necessário enviar a imagem para o servidor do serviço, para que ele possa processar a imagem. Nas ferramentas de execução local, é possível corrigir eventuais falhas na detecção de borda, o que não acontece nos serviços on-line. Devido a esse fator, os serviços optaram por adotar um algoritmo de detecção de borda mais conservador, onde em muitos casos a imagem em que o realce é aplicado é a imagem sem correção de perspectiva, como pode ser visto neste capítulo nas seções anteriores. Outra vantagem das ferramentas de execução local é tornar possível o processamento de imagens confidenciais de quadros, que não podem ser enviadas pela Internet.

A principal vantagem dos serviços on-line está no armazenamento na organização de resultados em grupos, estes podem ser compartilhado com outras pessoas. Outra vantagem está no envio das imagens por e-mail, no formato PDF. Além de ser possível o envio de imagens por celular.

Na Tabela 6.6 é possível ver vários aspectos analisados anteriormente. Dentre as características, as únicas que o Tableau não possui são: a geração de imagem para impressão para qualquer fundo, correção da distorção de lente e agrupamento de imagens. Esses casos merecem um estudo mais aprofundado no futuro. No entanto, no caso da distorção de lente, nas imagens analisadas a distorção não interfere significativamente no resultado final, tanto que os outros algoritmos não a implementam. Lembrando que o Tableau contempla uma característica exclusiva em relação aos outros que é o realce de quadros com qualquer fundo.

Para calcular o tempo médio de resposta dos algoritmos foram adotadas duas estratégias para as ferramentas de execução local e os serviços on-line. Para o primeiro tipo foi considerado como tempo médio o carregamento da imagem, detecção de borda, correção de perspectiva e realce. A detecção de borda não incluiu o “tempo de interferência do usuário” caso o algoritmo detecte errado.

Para o segundo tipo, as imagens foram enviadas por e-mail os serviços on-line em grupos conforme especificado anteriormente. Verificou-se o tempo de resposta a partir da hora do envio do e-mail e a hora da resposta. Como não há informação dos segundos no Gmail®, a precisão desse tempo ficou limitada aos minutos. No caso do Qipit® o tempo ficou em torno de 1 minuto, houve casos em que foi menor que isso. Já o ScanR® demorou mais em torno de 2 minutos pra responder com os resultados.

Tanto o WhiteboardIt como o Tableau, ambos desenvolvidos pelo autor desta dissertação como complemento do ImageJ, tiveram desempenho (em termos de tempo de execução) superior ao ClearBoard e Whiteboard Photo.

O Whiteboard Photo foi renomeado para WBPhoto e o WhiteboardIt para WBIt nas colunas da tabela abaixo.

Tabela 6.6 – *Comparativo entre os sistema de processamento de quadros didáticos*

	ClearBoard	WBPhoto	WBIt	QipIt	ScanR	Tableau
Detecção de borda / Correção de perspectiva	Sim	Sim	Sim	Sim	Sim	Sim
Tipos de quadro para realce	Negro, verde e branco	Negro, verde e branco	Apenas branco	Apenas branco	Apenas branco	Qualquer com fundo uniforme
Gera imagens para impressão	P/ quadro negro, verde e branco	Só para quadros brancos	Só para quadros brancos	Só para quadros brancos	Só para quadros brancos	Só para quadros brancos
Tempo médio de resposta	20 s	20 s	7 s *	1 min	2 min	7 s
Taxa de acerto na detecção de borda	4%	48%	Não disponível	17%	0%	52%

	ClearBoard	WBPhoto	WBIt	QipIt	ScanR	Tableau
Correção de distorção de lentes	Não	Sim	Não	Não	Não	Não
Agrupamento de imagens	Não	Não	Sim	Não	Não	Não
Tamanho mínimo	Não tem	Não tem	Não tem	1Mpixel	1100 x 800	Não tem

* Sem deteção de borda

6.4 Resumo

Este capítulo teve o objetivo de comparar o Tableau com outros sistemas correlatos. A comparação inclui, dois sistemas comerciais de execução local, o Clearboard® e WhiteboardPhoto® que processam imagens de quadros branco, verde e negro. Dois serviços de Internet, que conseguem apenas processar imagens de quadros brancos, o QipIt® e ScanR®. No entanto, não foi possível obter o ambiente WhiteboardIt, por ser um projeto interno de pesquisa da Microsoft® onde não possível baixar o executável do ambiente.

Dentro os sistemas o Tableau foi o que obteve o melhor desempenho. As únicas funcionalidades presentes em outros sistemas e ausentes no Tableau foram: correção de distorção de lentes (presente apenas no WhiteboardPhoto), junção de várias imagens do mesmo quadro (presente apenas no WhiteboardIt) e geração imagens de quadros não brancos para impressão (presente apenas no Clearboard). Tanto entre esses sistemas como na literatura acadêmica, o Tableau é o único que consegue realçar quadros com qualquer cor de fundo.

7 Conclusões e trabalhos futuros

O quadro didático é uma das ferramentas universalmente utilizadas para transmissão de conhecimento.

Câmeras digitais portáteis estão ficando cada vez mais presentes na vida cotidiana, devido ao seu baixo custo e praticidade. Sem contar com câmeras embutidas em celulares, sendo este um bem que está se tornando pervasivo. Por essas razões, as câmeras digitais portáteis estão sendo usadas para registrar o conteúdo dos quadros didáticos para uso posterior às aulas e às reuniões.

As imagens de quadros didáticos apresentam várias dificuldades inerentes, pois a grande maioria das fotos contém objetos que não pertencem ao conteúdo do quadro. Muitas vezes, a foto é tirada fora da posição ideal, onde o plano da foto é paralelo ao do quadro e a iluminação ocorre de modo irregular sobre o quadro. Deve-se considerar também que não há padrão de conteúdo bem definido, diferente do caso dos documentos impressos. Além desses fatores, as imagens de quadros podem conter todo o quadro, apenas uma parte dele ou só o conteúdo desejado, dependendo do modo como a foto foi tirada.

Tableau é um sistema que propõe ajudar o usuário a melhorar fotos de quadros didáticos. O sistema proposto foi desenvolvido como um complemento do ImageJ, uma ferramenta de processamento de imagens feita em Java, portanto de grande portabilidade, sendo possível rodar praticamente qualquer sistema operacional existente.

Tableau permite ao usuário melhorar a imagem do quadro através de três formas: detecção da borda do quadro, separando o conteúdo do quadro do resto; correção da perspectiva, deixando a imagem do quadro como se fosse vista frontalmente; e realçar a escrita do quadro tornando a cor do fundo uniforme (“chapado”), podendo o quadro ser de qualquer cor de fundo.

Ao comparar-se com outras ferramentas comerciais – ClearBoard®, Whiteboard Photo®, QipIt® e ScanR® – o Tableau superou essas ferramentas em vários aspectos:

- O primeiro aspecto é relativo à maior taxa de acerto com relação à detecção de borda.
- Outro ponto é a maior rapidez no tempo de resposta em relação às alternativas.

- Tableau é o único sistema que permite o realce de quadros de qualquer cor e possui desempenho similar ou superior aos outros algoritmos de realce nos quadros brancos, verdes e pretos.

Com o Tableau não há restrição de tamanho mínimo para aplicação de qualquer algoritmo, sendo esta restrição vista nos serviços na Internet.

No entanto, o Tableau não possui rotinas para correção de distorção causada pelas lentes, sendo esta característica presente apenas no Whiteboard Photo®. O Tableau não possui a funcionalidade de juntar várias fotos de partes de diferentes do mesmo quadro, o que é possível com o WhiteboardIt®. O Tableau não permite geração de imagens mais econômicas para impressão, esta funcionalidade está apenas presente no ClearBoard®. Todas essas funcionalidades ausentes no Tableau servem de inspiração para pesquisas futuras.

Convém, ainda, observar que as imagens de quadros didáticos, quando realçadas, apresentam poucas cores. A partir da imagem realçada, uma imagem binária pode ser obtida, como existem poucas distinções de tonalidades de riscos, sendo possível mapear cada risco do quadro (que na imagem binária é representado por preto) em uma cor predominante. Para gravar tal imagem, é necessária, apenas, a imagem binária e uma tabela informando a cor de cada risco, gerando um arquivo de tamanho menor.

Os algoritmos apresentados nesta dissertação possuem pouco consumo de memória. Como há muitos celulares no mercado com capacidade de processamento e memória suficientes para executá-los, a um custo relativamente baixo, um estudo poderia ser feito para identificar o desempenho dos mesmos nesses celulares. Para exemplificar, pode-se citar o celular Nokia 6120 classic (NOKIA, 2008), com CPU de 369 Mhz, 20 Mb de memória executável, câmera de 2 Mpixels e plataforma Symbian (em que é possível executar código nativo), uma imagem de 2 Mpixels consume 8 Mb de memória. Com 20 Mb é possível carregar duas imagens, sendo suficiente para execução dos algoritmos apresentados.

Para o desenvolvimento de uma versão móvel do Tableau é recomendável o incremento de um mecanismo de detecção de borda e verificação de qualidade da foto em tempo real, para ajudar o usuário a fotografar com a melhor qualidade possível.

A utilização de outros mecanismos de diferença de cores deve ser estudada com o fim de identificar um melhor mecanismo que o CIEDE2000 com uma menor taxa de erro.

O uso do Tableau pode ser estendido para um ambiente na Internet onde seja possível enviar fotos do quadro e compartilhar com outros usuários. Esse sistema permitiria aos usuários criar cursos onde seria possível descrever as informações sobre a cadeira e o conteúdo da aula registrado, através de fotos de quadros didáticos.

Cada foto poderia ser comentada para estimular a discussão em grupo sobre os tópicos abordados nas aulas. Tal funcionalidade não está presente nos serviços de Internet apresentados aqui, esses apenas processam a imagem e compartilham as mesmas, porém não é possível usá-las colaborativamente.

No período entre o término da redação desta dissertação e a defesa desta dissertação, o algoritmo de realce do Tableau foi modificado para contemplar documentos fotografados, sendo publicado no CBDAR (International Workshop on Camera-Based Document Analysis and Recognition) 2009 e podendo ser obtido na referência (OLIVEIRA and LINS, 2009).

8 REFERÊNCIAS

(ACMI, 2008) Australian Centre for the Moving Image. **ADVENTURES in CYBER SOUND, The Camera Obscura : Aristotle to Zahn**. Disponível em: http://www.acmi.net.au/AIC/CAMERA_OBSCURA.html. Acesso em: 12 de novembro de 2008.

(ADOBE, 2008) ADOBE. **About Lens Distortion. Photoshop CS3 Help**. Disponível em: http://livedocs.adobe.com/en_US/Photoshop/10.0/help.html?content=WS04BE5944-BF2A-41c9-84D8-3D447FF965CE.html. Acesso em: 21 de julho de 2008.

(BARTHEL, 2008) BARTHEL, Kai Uwe. 3D Color Inspector/Color Histogram. Acessado em 5 de maio de 2008. Disponível em: http://rsb.info.nih.gov/ij/plugins/color-inspector.html.

(BROWN, 2008) BROWN, Gary; 2008. **How autofocus cameras works**. Disponível em: http://electronics.howstuffworks.com/autofocus.htm/printable. Acesso em 14 de novembro de 2008.

(BURGER and BURGE, 2008). BURGER, Wilhelm; BURGE, Mark J.; 2008. **Digital Image Processing: An Algorithmic Introduction using Java**. ISBN: 978-1-84628-379-6. Springer, New York, 2008. Imagens disponíveis em: http://www.imagingbook.com/. Acesso 8 de dezembro de 2008.

(BURNETT, 2008) BURNETT, Colin. **Image:Bayer pattern on sensor.svg.** Disponível em: http://en.wikipedia.org/w/index.php?title=Image:Bayer_pattern_on_sensor.svg&oldid=210441615. Acesso em: 10 de outubro de 2008.

(BUSELLE, 1977) BUSELLE, Michael; 1977. **Tudo sobre Fotografia**. São Paulo: Pioneira.

(CAMBRIDGEINCOLOUR, 2008a) **Optimizing Digital Photo Elargenment**. Cambridge in Colour, 2008. Disponível em: http://www.cambridgeincolour.com/tutorials/digital-photo-enlargement.htm. Acesso em: 8 de dezembro de 2008.

(CAMBRIDGEINCOLOUR, 2008b) **Understanding Digital Image Interpolation**. Cambridge in Colour, 2008. Disponível em: http://www.cambridgeincolour.com/tutorials/digital-photo-enlargement.htm. Acesso em: 8 de dezembro de 2008.

(CANNY, 1986) CANNY, J.; 1986. **A Computational Approach for Edge Detection.** IEEE Transactions on Pattern Analysis and Machine Intelligence. Volume 8, number 6, pp. 679-698.

(CBURNETT, 2008) BURNETT, Colin M.L. **Image:Moire pattern of bricks small.jpg**. Disponível em: http://en.wikipedia.org/wiki/Image:Moire_pattern_of_bricks_small.jpg. Acesso em: 14 de nov. de 2008.

(CELENK, 1990) CELENK, Mehmet; 1990. **A Color Clustering Technique for Image Segmentation**. Computer Vision, Graphics, and Image Processing. 52 (3) (1990) 145}170.

(CHENG *et al*, 2001) CHENG, H.D.; JIANG, X.H.; SUN, Y.; WANG, Jingli.; 2001. **Color image segmentation: advances and prospects**. Pattern Recognition 34, p. 2259-2281. Elsevier.

(CHERUBINI *et al*, 2007) CHERUBINI, Mauro; VENOLIA, Gina; DELINE, Rob; KO, Andre J.; 2007. **Let's Go to the Whiteboard: How and Why Software Developers Use Drawings**. Proceddings of Computer Human Interaction. San José, CA, USA.

(CIE, 2001) CIE; 2001. **Improvement to industrial colour-difference evaluation**. CIE Publication No. Central Bureau of the CIE, Vienna.

(DUDA and HART, 1972) DUDA, R.O.; HART, P.E.; 1972. **Use of the Hough transformation to detect lines and curves in pictures, Commun**. ACM 15 (1) (1972), p. 11–15.

(EOM *et al*, 2007) EOM, Minyoung; OH, Jaegeun; KIM, Seon Wook; 2007. **Camera interface method in mobile handset and its performance comparison. Em: INTERNATIONAL CONFERENCE ON** PARALLEL PROCESSING WORKSHOPS. Disponível em: http://dx.doi.org/10.1109/ICPPW.2007.27. Acesso em: 10 de out. de 2008.

(FAIRCHILD, 2005) FAIRCHILD, Mark D.; 2005. **Color and Image Appearance Models**. John Wiley and Sons, 340. ISBN 0470012161.

(FARAMARZPOUR *et al*, 2007) FARAMARZPOUR, Naser; DEEN, M. Jamal, SHIRANI, Shahram ; FANG, Qiyin; LIU, Louis W. C.; CAMPOS, Fernando de Souza; e SMART, Jacobus W; 2007. **CMOS based active pixel for low-light-level detection: Analysis and measurements.** IEEE Trans. Electron Devices, vol. 54, p. 3229.

(FAUGERAS, 1993) FAUGERAS, Olivier; 1993. **Three-Dimensional Computer Vision: A Geometric Viewpoint**. MIT Press, Cambridge, MA.

(GOETZ, 2003) GOETZ, Brian; 2003. **Java theory and practice: Where's your point?**. IBM Developer Works. Acessado em 8 de dezembro de 2008. Disponível em: http://www.ibm.com/developerworks/java/library/j-jtp0114/

(GONZALEZ and WOODS, 2008) GONZALEZ, Rafael C.; WOODS, Richard E.; 2008. **Digital Image Processing**. 3rd ed. Prentice Hall.

(HAIN *et al*, 2007) HAIN, Rainer; KÄHLER, Christian J; TROPEA, Cam.; 2007. **Comparison of CCD, CMOS and intensified camera**. Exp Fluids, 2007. doi:10.1007/s00348-006-0247-1.

(HARMS, 2007) HARMS, Douglas; 2007. **A brief comparison of Technologies Available in the Computer Science Classroom**. Em: International Conference on Computer System and Techniques. CompSysTech.

(HARTLEY and ZISSERMAN, 2003) HARTLEY, Richard; ZISSERMAN, Andrew; 2003. **Multiple View Geometry and Computer Vision**. Cambridge University Press, New York.

(HE *et al*, 2003) HE, Li-wei; LIU, Zicheng; ZHANG, Zhengyou; 2003. **Why take notes? Use the whiteboard system**. Em: International Conference on Acoustics, Speech, and Signal Processing (ICASSP2003), p. 776-779.

(HOUGH, 1962) HOUGH, P.V.C; 1962. **Methods and means for recognizing complex patterns**. U.S. Patent 3.069.654.

(HOUSE *et al*, 2005) HOUSE, Nancy Van; DAVIS, Marc; AMES, Morgan; FINN, Megan; VISWANATHAN, Vijay; 2005. **The Uses of Personal Networked Digital Imaging: An Empirical Study of Cameraphone Photos and Sharing. Human Computer Interaction**, p. 1853-1856.

(JAGANNATHAN and JAWAHAR, 2005) JAGANNATHAN, L.; JAWAHAR, C. V.; 2005. **Perspective correction methods for camera based document analysis,** p. 148-154, CBDAR 2005/ICDAR, Korea.

(JAIN, 1989) JAIN, Anil K.; 1989. **Fundamentals of Digital Image Processing**. New Jersey, United States of America: Prentice Hall, p. 68, 71, 73. ISBN 0-13-336165-9.

(JAVANET, 2008) Javanet; 2008. **colorchooser: The SwingLabs Quick Color Chooser**. Acessado em 10 de setembro de 2008. Disponível em: https://colorchooser.dev.java.net/.

(JIN *et al*, 2007) YIN, Xu-Cheng; SUN, Jun; FUJII, Yusaku; FUJIMOTO, Katsushito; NAOI, Satoshi; 2007. **Perspective Rectification for Mobile Phone Camera-Based Documents Using a Hybrid Approach to Vanishing Point Detection**. Em: Proceedings of Second International IAPR Workshop on Camera-Based Document Analysis and Recognition, p.37-44, IAPR.

(LINS *et al*, 2007) LINS, Rafael Dueire; SILVA, André Ricardson Gomes e; SILVA, Gabriel Pereira da Silva; 2007. **Enhancing Document Images Acquired Using Portable Digital Cameras**. Em: ICIAR 2007, LNCS, Springer Verlag, p. 1229-1241.

(LIU *et al*, 2007) LIU, Zicheng; ZHANG, Zhengyou; HE, Li-wei; 2007. **Whiteboard scanning and image enhancement**. Digital Signal Processing, Volume 17, Issue 2, p. 414-432.

(MARONNA *et al*, 2006) MARONNA, Ricardo A.; MARTIN, Douglas R.; YOHAI, Victor J.; 2006. **Robust Statistics: Theory and Methods**. John Wiley & Sons, Ltd, England. ISBN: 0-470-01092-4.

(MARR and HILDRETH, 1980) MARR, D.; HILDRETH, E.; 1980. **Theory of Edge Detection.** Proceedings of the Royal Society of London, Volume B207, p. 187-217.

(MASALOVITCH and MESTETSKIY, 2007) MASALOVITCH, Anton; MESTETSKIY, Leonid; 2007. **Usage of Continuous Skeletal Image Representation for Document Images De-warping**. Em: Proceedings of Second International IAPR Workshop on Camera-Based Document Analysis and Recognition, p.45-52, IAPR.

(MELLISH, 2008) MELLISH, Robert; 2008. **Image:Pinhole-camera.png**. Wikimedia Commons, 2008. Disponível em http://commons.wikimedia.org/w/index.php?title=Image:Pinhole-camera.png&oldid=11832052.

(NAKAYAMA and IKEDA, 2004) NAKAYAMA, Masahura; IKEDA, Koichi; 2004. **Comparison of Perceived Colour Differences with Colorimetric Colour Differences in Uniform Colour Spaces and Colour Appearance Model**. Em: Journal of Light & Visual Environment, Vol.28, No.2, 2004. Disponível em: http://www.jstage.jst.go.jp/article/jlve/28/2/81/_pdf.

(NIKON, 2008) Nikon, 2008. **Predictive Focus Tracking System**. Disponível em: http://www.nikon.com/about/technology/core/software/caf/index.htm. Acesso em 14 de novembro de 2008.

(NOKIA, 2008) Nokia, 2008. **Device Details -- Nokia 6120 classic**. Disponível em: http://www.forum.nokia.com/devices/6120_classic. Acesso em 10 de dezembro de 2008.

(OLIVEIRA and LINS, 2007) OLIVEIRA, Daniel M.; LINS, Rafael D.; 2007. **Tableau – Processing Teaching-board Images Acquired with Portable Digital Cameras**. Em: Proceedings of Second International Workshop on Camera-Based Document Analysis and Recognition, p.79-86, IAPR.

(OLIVEIRA and LINS, 2008) OLIVEIRA, Daniel Marques; LINS, Rafael Dueire; 2008. **Improving the Border Detection and Image Enhancement Algorithms in Tableau**. ICIAR 2008, LNCS 5112, p. 1111-1121, Springer Verlag.

(OLIVEIRA and LINS, 2009) OLIVEIRA, Daniel M.; LINS, Rafael D.; 2009. **A New Method for Shading Removal and Binarization of Documents Acquired with Portable Digital Cameras**. Em: Proceedings of Third International Workshop on Camera-Based Document Analysis and Recognition, p.3-10, IAPR.

(PERES, 2007) PERES, Michael R.; 2007. **Focal Encyclopedia of Photography**. Elsevier.

(PREWITT, 1970) PREWITT, J. M. S.; 1970. **Object Enhancement and Extraction**. Picture Processing and Pyschopictorics, Linkin, B.S. and Resenfeld, A. (eds), Academic Press, New York.

(ROBERTZ, 1965) ROBERTZ, L.G.; 1965. **Machine Perception of Three-Dimensional Solids**. Optical and Electro-Optical Information Processing, Tippet. J. T. (ed.), MIT Press, Cambridge, Mass.

(ROUSEEUW and LEROY, 1987) ROUSEEUW, Peter J.; LEROY, Annick M.; 1987. **Robust Regression and Outlier Detection**. John Wiley & Sons, Inc, United States.

(RUSSO, 2004) RUSSO, Fabrizio; 2004. **Piecewise linear model-based image enhancement**. EURASIP J. Applied Signal Processing, p. 1861-1869.

(SHARMA *et al*, 2005) SHARMA, Gaurav; WU, Wencheng; DALAL, Edul N.; 2005. **The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations**. Color Research & Applications 30 (1), p. 21–30. Wiley Interscience, doi:10.1002/col.20070.

(SILVA and LINS, 2005) SILVA, André Ricardson Gomes; LINS, Rafael Dueire; 2005. **Background Removal of Document Images Acquired Using Portable Digital Cameras**. LNCS 3656, p. 278-285.

(SILVA, 2006) SILVA, André Ricardson Gomes; 2006. **Análise e Melhoria da Qualidade de Documentos Fotografados**. Dissertação de Mestrado. Universidade Federal de Pernambuco, Departamento de Eletrônica e Sistemas, Pernambuco, Recife, 2006.

(SÖBEL, 1970) SÖBEL, I.E.; 1970. **Camera Models and Machine Perception**. Ph.D. dissertation. Stanford University University. Palo Alto. Calif.

(THEUWISSEN, 2007) THEUWISSEN, Albert J.P.; 2007. **CMOS image sensors: State-of-the-art and future perspectives**. Em: 37th European Solid State Device Research Conference (September 11-13, 2007). ESSDERC. IEEE Computer Society, Munich, 2007. DOI= http://dx.doi.org/10.1109/ESSDERC.2007.4430875.

(UCHIDA *et al*, 2007) UCHIDA, Seiichi; SAKAI, Megumi; IWAMURA, Masakazu; OMACHI, Shinichiro; KISE, Koichi; 2007. **Instance-Based Skew Estimation of Document Images by a Combination of Variant and Invariant**. Em: Proceedings of Second International IAPR Workshop on Camera-Based Document Analysis and Recognition, p.79-86, IAPR.

(WIKIPEDIA, 2008). **Autofocus**. Disponível em: http://en.wikipedia.org/w/index.php?title=Autofocus&oldid=250226965. Acesso em 14 de novembro de 2008.

(WOLBERG, 1990) WOLBERG, George; 1990. **Digital Image Warping**. p. 129-131, IEEE, ISBN 0-8186-8944-7.

(ZHANG and HE, 2004) ZHANG, Zhengyou; HE, Li-wei; 2004. **Notetaking with a camera: Whiteboard scanning and image enhancement**. Em: IEEE International Conference on Acoustics, Speech, and Signal Processing, volume 3, pages 533–536, Montreal, Quebec, Canada.

(ZHANG and TAN, 2001) ZHANG, Zheng; TAN, Chew Lim; 2001. **Restoration of images scanned from thick bound documents**. Em: International Conference on Image Processing, volume 1, p. 1074 – 1077, IEEE.

(ZHANG *et al*, 2006) ZHANG, Zhengyou et al. 2006. **System and method for whiteboard scanning to obtain a high resolution image**. U.S. Patent 7,119,816, filed March 31, 2003, issued October 10, 2006.

(ZHANG, 1997) ZHANG, Zheng; 1997. **Parameter estimation techniques: A tutorial with application to conic fitting**. Image Vision Comput, p. 59–76.

(DPREVIEW, 2008) DPREVIEW; 2008. **Digital Photography Review**. Disponível em: http://www.dpreview.com. Acesso 12 de outubro de 2008.

(FLICKR, 2008) Flickr, 2008. **Flickr – compartilhamento de fotos**. Disponível em: http://www.flickr.com. Acesso 12 de outubro de 2008.

(GOOGLE, 2008) Google; 2008. **Picasa, álbuns da web**. Disponível em: http://picasaweb.google.com.br. Acesso 12 de outubro de 2008.

(POLYVISION, 2008) POLYVISION; 2008. **Whiteboard Photo Image Capture System**. Disponível em: http://www.polyvision.com/products/wbp.asp. Acesso 10 de agosto de 2008.

(RASBAND, 2008) RASBAND, Wayne; 2008. **ImageJ – Image Processing and Analysis in Java**. Disponível em: http://rsb.info.nih.gov/ij/. Acesso 12 de outubro de 2008.

(REALEYES3D, 2008) REALEYES 3D; 2008. **Qipit® – Copy and Share Documents**. Disponível em: http://www.qipit.com. Acesso 12 de outubro de 2008.

(SCANR, 2008) SCANR; 2008. **Scanr® – Scan, copy, and fax with your câmera phone**. http://www.scanr.com

(SOFTTOUCHIT, 2008) SOFTTOUCHIT; 2008. **ClearBoard**. Disponível em: http://softtouchit.com/xpe/doc/softtouchit/products/clearboard. Acesso 10 de agosto de 2008.

APÊNDICE A

A.1 Tableau – Processing Teaching-board Images Acquired with Portable Digital Cameras

Publicado no *Proceddings of Second International Workshop on Camera-Based Document Analysis and Recognition*, Curitiba, Brasil, 22 de setembro de 2007

A.2 Improving the Border Detection and Image Enhancement Algorithms in Tableau

Publicado no *Proceddings of International Conference on Image Analysis and Recognition*, Póvoa Varzim, Portugal, 25-27 de junho de 2008.

Tableau - Processing Teaching-board Images Acquired with Portable Digital Cameras

Daniel Marques Oliveira and Rafael Dueire Lins
Departamento de Eletrônica e Sistemas – UFPE – Recife – PE – Brazil
danielmarquesoliveira@gmail.com, rdl@ufpe.br

Abstract

Portable digital cameras are of widespread use today due to good image quality, low cost and portability. Teaching-boards are the most universal classroom equipment throughout the world. This paper presents a software environment for processing images from teaching-boards acquired using portable digital cameras and cell-phones.

Keywords: *Digital cameras, image processing, portable cameras, teaching boards.*

1. Introduction

Portable digital cameras were developed for taking amateur "family photos"; the recent price-performance improvement, low weight, portability, low cost, small dimensions, etc. widened enormously the number of users of digital cameras giving birth to several new applications. One of them, completely unforeseen is using portable digital cameras for digitalizing images from teaching-boards. Teaching boards are present in every classroom throughout the world with small variations: modern ones are white and written with color felt tip markers; some others are black or green and written with white chalk sticks. Students take notes of what teachers write on the board for later revision. Today, some students start to take photos of classroom boards for later reference.

This paper describes a software environment to process images of teaching boards acquired using portable digital cameras operated either by students or teachers. This simple tool provides a natural way to generate digital content for courses, respecting particular aspects of the group such as syllabus, class learning speed, teacher experience, regional content, local culture, etc.

The system consists of three parts. The first is database formation. As soon as the images are transferred from the camera to the PC information is collected to generate a simple database that will organize the images for later content formation. Information such as teacher name, course name, discipline, subject, class number, group number, etc. are requested. The second module is for image processing. This module will improve the image acquired in a number of ways involving background removal, image segmentation, skew correction, image enhancement, etc. The third part of the processing environment deals with outputting the content. Three different ways are under development: printed handouts, webpage generation and slide production. Each of these media receives the information of the processed image part of the environment and makes it suitable to its best use. This paper focuses on the image processing parts of the environment.

2. Image Acquisition

Image acquisition is performed by taking a photograph of the teaching board at a suitable distance, before cleaning up the information. Whenever a photo is taken, special care is needed to keep the readability of the text in the inbuilt camera LCD display. The image processing part takes the images acquired by a portable digital camera and processes them in a number of ways. Very often the photograph goes beyond the board size and incorporates parts of the wall that served as mechanical support for taking the photo of the document. Boards often have frames either made of wood or metal. The second problem is due to the skew often found in the image in relation to the photograph axes, as cameras have no fixed mechanical support very often there is some degree of inclination in the document image. The third problem is non-frontal perspective, due to the same reasons that give rise to skew. A fourth problem is caused by the distortion of the lens of the camera. This means that the perspective distortion is not a straight line but a convex line (in digital camera photos) or concave line (in cell phone photos), depending on the quality of the lens and the relative position of the camera and the document. The fifth difficulty in processing board images acquired with portable cameras is due to non-uniform illumination. White boards have a polished surface to avoid the marker ink to be absorbed by the board surface. This yields non uniform photo illumination as one often finds high intensity bright areas that correspond to reflections of room lighting. Figure 1 presents an example of a white board photographed

with a low-cost mobile phone Nokia 6020, where one may observe the four problems aforementioned: extra borders, image skew, non-frontal perspective distortion, and lens distortion. Besides those problems one may add: uneven multiple source illumination and non-delimited areas. One must remark that all pictures taken for this study and presented herein were from real classes. In the case of the board presented in Figure 1, there is a written area in the top-leftmost area that belonged to a "previous" board area. The lecturer did not respect the board junction that imposes a natural area delimiter. The pictures were taken after lectures without previous information to the lecturer. If informed beforehand, lecturers ought to respect area separation to make easier board segmentation. What is most surprising is that despite the low resolution of the camera of the cell-phone and the non-ideal environment, the image obtained provides readable information.

The board image presented in Figure 2 also exhibits the four problems already described. The photo was taken without natural daylight interference and strobe flash (the HP iPaq has no internal strobe flash). Room illumination was from tube fluorescent lamps. One may notice that the lecturer respected the board junction as a content delimiter.

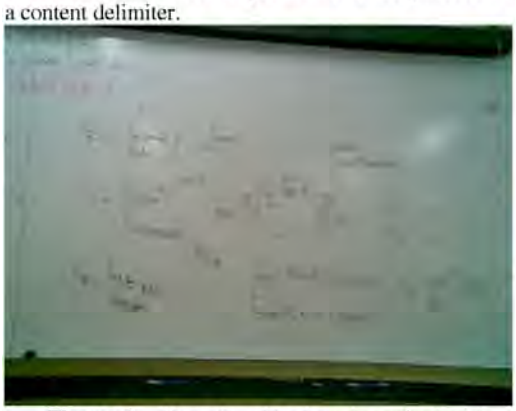


Figure 1. Part of a white-board acquired with internal camera of the cell-phone Nokia 6020, no strobe-flash used, 640x480 pixels, image size 21KB under Jpeg compression, board height 115cm, illumination: natural daylight (indirect) and ceiling tube fluorescent lamps.

An Olympus portable digital camera was used to acquire the board image presented in Figure 3. Two aspects are new in this image. The first is the presence of some blurred areas due to imperfect board cleaning or aging of surface of the board. The second is that the lecturer used vertical lines to split the content of his presentation on different board segments.

Figure 2. Part of a white-board acquired with internal 1.2 Mpixel camera of a HP iPaq rx3700, no strobe-flash used, image size 131KB under Jpeg compression, board height 115cm, illumination: natural ceiling tube fluorescent lamps.

Figure 3. Part of a white-board acquired with portable digital camera Olympus C60, 6.0 Mpixels, no strobe-flash used, image size 1.20MB under Jpeg compression, board height 115cm, illumination: natural ceiling tube fluorescent lamps.

3. Boundary detection

The first step to process teaching board information is to find the limits of the image. As already mentioned, boards often have frames as a decorative "finishing" or mechanical support as may be observed in figures 2 and 3, but that is not always found. Figure 1 is an example of the latter case. Besides that, in real classrooms a board is several meters wide. Thus, the content of a board often claims for several photos to be covered. Figures 2 and 3 exemplify a left section of a board while Figure 1 presents a central slice of a board. One should observe that the non-framed edges bring a

higher-complexity for boundary detection, thus for image segmentation.

Boundary detection is central for all other steps described herein because it isolates the area of interest from its surroundings. Besides that, the detection of the boundaries will allow one be able to correct perspective and skew in the image. Unfortunately, boundary detection has shown itself a much harder task than one may first assume. The algorithm presented in reference [8] [9] used to remove the mechanical background from document images acquired with portable digital cameras is unsuitable for board images, despite the high degree of similarity in the two problems addressed. A new algorithm is presented herein, performing the following steps:

3.1 Segmentation

1. Split the input image (ORI_IMG) into 4 regions as presented in Figure 7
2. Create a new binary image (DIF_IMG) of equal dimensions;
3. H_DIST and V_DIST are defined as functions of the image resolution. They correspond to the rounding-off of the integer part of 0.91% of the width and height in pixels of the original image, respectively. For instance, in the case of a 640x480 pixel image, H_DIST=6 and V_DIST=4.
4. DIF_IMG(x,y) is white if one of the following condition holds, it is black otherwise:

- The difference between each component of ORI_IMG(X,Y) and ORI_IMG(X±H_DIST,Y±V_DIST) is less than 10
- The componentwise gap of ORI_IMG(X,Y) and ORI_IMG(X±2.H_DIST,Y±2.V_DIST) is < 10.
- The pixel differences are local operations, which minimize non-uniform illumination. A difference between non-board areas and board areas is more likely to turn black in DIF_IMG, than if all pixels compared belong to the board.

Two pixels differences are needed to minimize the "double contour" around board writings. The first contour is marked as black when ORI_IMG(X,Y) is located on the teacher writing and the inner pixel is a board background. The second contour is the other way round, thus DIF_IMG is wrongly marked as black. Such behavior may be seen in Figure 5 for the letter "A" obtained from Figure 3. The behavior when considered the two differences is shown in Figure 6.

Figure 4. Letter "A" extracted from Figure 3

Figure 5. Segmentation considering just one difference

Figure 6. Segmentation considering the two differences

The sign is such as always to subtract the outermost value from the innermost one, according to the matching position from the region on Table 1.

Figure 7. Board split into four regions. Arrows show the direction of illumination compensation.

The result of Step 1 applied to the image presented in Figure 7 is shown in Figure 8.

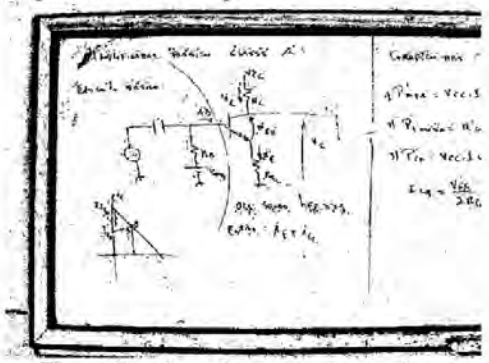


Figure 8. Figure 3 board image after Step 1.

Table 1. H_DIST and V_DIST offset calculation.

X+H_DIST, Y+V_DIST	X-H_DIST, Y+V_DIST
X+H_DIST, Y-V_DIST	X-H_DIST, Y-V_DIST

3.2 Finding Control Points

Points that possibly belong to the board boundaries are called control points. This step will try to spot them by analyzing the binary image.

For each direction, N equally spaced axes are defined to scan the binary image looking for control points from the centers towards the borders. Each of those vertical axes is swept with a 7x1 pixel mask. Similarly, each horizontal axis is scanned with a 1x7 pixel mask. If four or more pixels under the mask are black, go to 2. For the collected images N was set up to 9.

1. Calculate colors ratio in the rectangle around current point (X, Y). The bottom border of the rectangle has the upper left corner located on (A,B) and the lower right corner on (C,D), where:

$$A = X - \left[\frac{IMG\_WIDTH}{N} \times \frac{1}{2}\right]$$

$$B = Y - IMG\_HEIGHT \times INTERNAL\_CHECK$$

$$C = X + \left[\frac{IMG\_WIDTH}{N} \times \frac{1}{2}\right]$$

$$D = Y + IMG\_HEIGHT \times EXTERNAL\_CHECK$$

* Where INTERNAL_CHECK=0.165% and EXTERNAL_CHECK=0.91%

2. If within the rectangle (A,B,C,D) there is more than 65% of black pixels, then (X,Y) is marked as a control point candidate for the bottom border. Otherwise, the algorithm moves outwards looking for another candidate.

For all other directions the algorithm works similarly to the step explained above. An example of control points found for the board image in Figure 8 is presented in Figure 9.

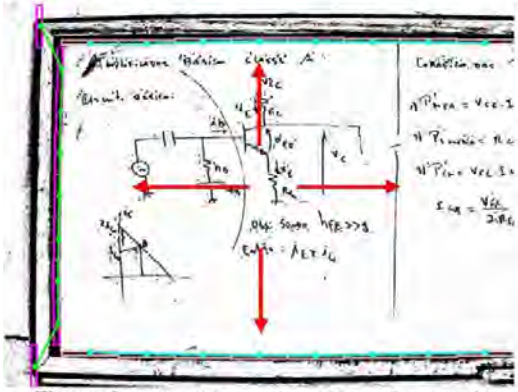


Figure 9. Board image from Figure 8 scanned in red from the center towards the outwards exhibiting the control points on top-bottom-left positions.

3.3 Control point elimination

Control points can be wrongly found, due to:

- The ORI_IMG(X,Y) it is out of the boundary of the board, as shown in Figure 9;
- When the board has added "noise" such as a sign or advertisement;
- The board picture was taken with strobe flash, etc.

To eliminate such points the following procedures are executed:

1. For every control point candidate, tangents of the angles (Θn) with the border axis formed by the candidate and its 2-neighborhood in both sides are calculated. A candidate is selected if the absolute value of at least 2 tangents is lower or equal to 0.07. One may observe that this calculation is not relative to the number of the neighborhood, so if the candidate is the outermost point all tangents should be lower or equal to 0.07. An example of a horizontal border neighborhood is shown in Figure 10, where the candidate is in dark grey.

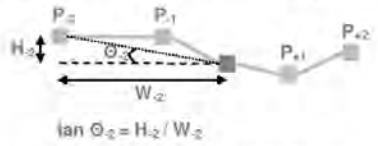


Figure 10. Control point selection

1. After executing step 1 above in all directions, the outermost points will define a line segment as depicted in Figure 11. Any candidate to a control point outside the orthogonal segment defined is excluded.

Figure 12 shows control point candidates. Figure 13 shows the image after the deletion of the wrong ones.

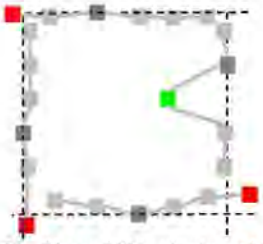


Figure 11. Green CP is eliminated in step one, while red ones are eliminated in step two.

Figure 12. Board image showing two wrong control points that are deleted.

Figure 13. Board image after control point elimination.

4. Perspective Correction and Cropping

The freedom allowed in acquiring board images with portable digital cameras without mechanical support invariably leads to perspective distortion. Perspective transformation must be done in order to make the image appear as a frontal vision. This process is called rectification [1] [3] [4] [5] [6] [10]. Four edges that delimit the original board image are needed. There are four kinds of board images:

1. The image presents no lateral borders as presented in Figure 1.
2. The image presents a left border on the image, as shown in Figures 02 and 03.
3. The image presents a right border on the image.
4. The whole board image fits the photograph.

In any of the cases above four points were taken as reference for perspective correction. Those points were chosen by drawing a line passing through the two outermost points in each direction and finding their intersections, which are named the *reference points*. If no control point is found in any direction the intersection of the lines drawn with the end of the image is taken as a reference point. This often happens in the three first cases above.

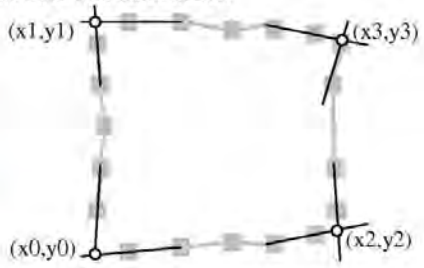


Figure 14. Reference points for perspective correction

One must remark that the technical literature registers other perspective correction techniques in the absence of reference points [4]. The adoption of the choice of reference points as above was done for a matter of uniformity and simplicity, and provided good results as is discussed later on.

Once the four reference points are chosen their equivalent after perspective correction are calculated as:

```
d1=|x0-x2|+|y0-y2|; d2=|x1-x3|+|y1-y3|;
d3=|x0-x1|+|y0-y1|; d4=|x2-x3|+|y2-y3|;
aspect=(d1+d2)/(d3+d4); x'0=x'1=x0;
x'2=x'3=xd0+d1; y'0=y'2=y0;
y'1=y'3=y'0+(d1/aspect);
```

5. Image Enhancement

There are several algorithms in the literature for enhancing images. Image board enhancement has to increase the contrast between the teacher writings and the board background, increasing information readability. Finding a way to cluster similar information to widen the gap between the different clusters is the key to image enhancement in the case of teaching-board images.

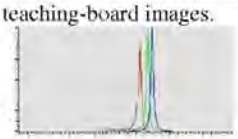


Figure 15. Background histogram of Figure 04

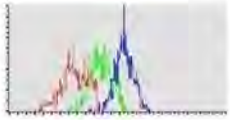


Figure 16. Histogram of Letter of Figure 04

Figure 15 and Figure 16 present the histograms of different selections of Figure 04. The letter "A" has a more representative contribution of the blue

component, and the color-histogram is widespread. The background histogram is narrower and more uniform. An efficient way to increase the contrast, without affecting the feature of an image, is provided by Rayleigh filter [7].

The images were tested against the following classes of algorithms: global and local histogram equalizations sharpen, and mathematical morphology techniques [7]. The Rayleigh filter with parameter $\lambda = 0.5$ consistently provided the best results. Figures 17 to 19 show images after Tableau processing.

Figure 17. Tableau applied to Figure 1

Figure 18. Tableau applied to Figure 2

Figure 19. Tableau applied to Figure 3

6. Tableau in ImageJ

ImageJ [11] is an open source image processing environment in Java developed by Wayne Rasband, is at the Research Services Branch, National Institute of Mental Health, Bethesda, Maryland, USA.It allows the insertion of plug-ins for special purposes. The algorithms presented herein for processing teaching board images was developed as an ImageJ plugin. Figure 20 presents a screen shot of the Tableau interface menu.

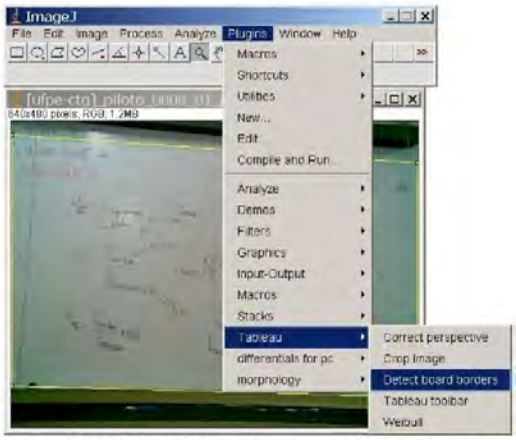


Figure 20. Tableau Plug-in interface in ImageJ

The algorithm for border detection presented above sometimes does not yield the best choice. The Tableau plug-in in ImageJ allows the user to adjust the contour detection for better processing.

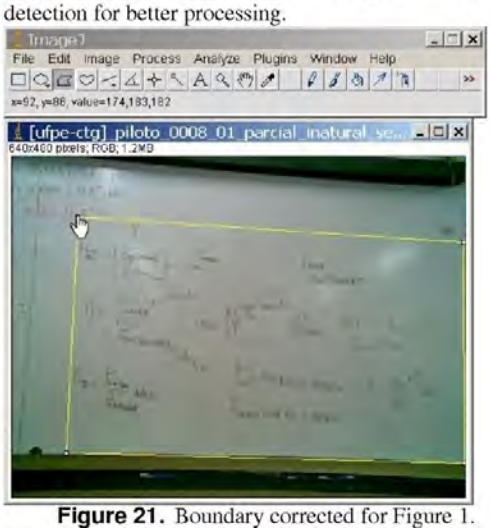


Figure 21. Boundary corrected for Figure 1.

Figure 22. Perspective corrected and cropped image from Figure 21.

Figure 20 presents the automatic boundary selection performed by the algorithm in the background, while Figure 21 shows the operator driven selection for the same image. The perspective corrected/cropped image is in Figure 22. Image in Figure 1 has size of 27.5 Kbytes while the cropped image claims only 15Kbytes of storage space, both compressed in JPEG. Average times for each of the board processing phases are showed on Table 2. They were measured considering the end-user perspective, thus screen refresh and progress bar updates are included in processing times.

Table 2. Average processing times in ImageJ

Tableau	6.0 Mpixel	1.2 Mpixel	300 Kpixel
Border detection	361ms	104ms	22ms
Persp. correction	42857ms	9692ms	2187ms
Image crop	589ms	224ms	79ms
Rayleigh ($\alpha = 0.5\sqrt{2}$)	515ms	198ms	60ms
Total time	44322ms	10218ms	2348ms

Times were measured on processor AMD Sempron 2600+ 1.83 GHz with 512Mb RAM running on Windows XP SP2. One may see that perspective correction in Tableau is very time consuming. This is due to the use of the JAI (Java Advanced Imaging) version 1.1.3 with machine native support [12] which is not incompatible with ImageJ, demanding to-and-from conversion of representations of the two libraries. The development of an ImageJ plug-in for such purpose will certainly yield more efficient code. Bicubic2 interpolation with subsampled bits equal to two was used [1]. One may see that the Rayleigh algorithm takes longer to process than the Border detection algorithm. This is due to the image refreshing window needed by Rayleigh while detection only shows the selection of board corners to the user.

Tableau was tested on 81 images from Olympus C60, 53 from Nokia 6020 and 3 from HP iPaq rx3700.

7. Comparisons with other approaches

Tableau was compared with two document processing web environments: Qipit®[13] and ScanR®[14]. The former was tested with 16 Olympus images, 3 from HP iPaq and 6 from Nokia 6020. The latter was tested on a subset of those images, as ScanR does not handle low-resolution images such as the ones taken with the Nokia 6020. Typical comparative results for the board of Figure 23 may be seen in Figures 24 to 26, working in similar circumstances.

Figure 23. Original image

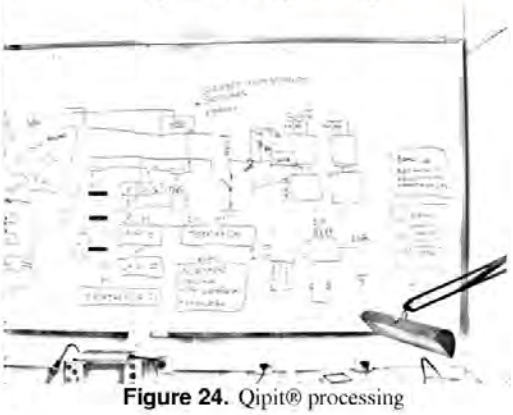


Figure 24. Qipit® processing

Figure 25. ScanR® approach

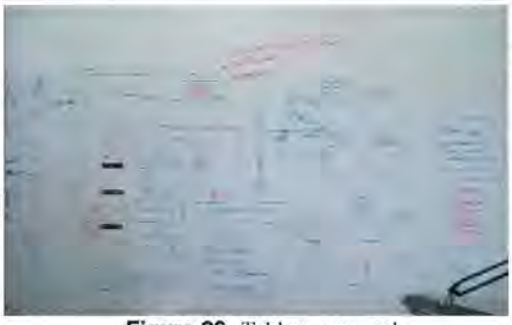


Figure 26. Tableau approach

As Figure 24 shows, Qipit neither detected the board border nor corrected the perspective, but binarized the image. ScanR, as Figure 25 shows, performs a poor border detection and crop, but a good image enhancement. Tableau, as shown in Figure 26, presents the best border detection and crop of the tools studied, but image enhancement falls behind ScanR. Performance comparison could not be done because both Qipit and ScanR systems are on-line Internet services.

8. Conclusions and Further Works

Portable digital cameras are a technological reality today that opens a wide number of challenges in image processing, including board image processing. This paper presented the image processing part of Tableau, an environment for assisting the layman to easily generate digital contents from teaching-board images. The image processing part of Tableau was implemented in ImageJ and finds the edges of board images, allowing perspective correction and image cropping, thus eliminating wall background and board frames. All images analyzed here were obtained from real classrooms without any special care either from lecturers or from the environment (illumination, special framing or marking, etc.). Different cameras were tested (different manufacturers, resolution, quality, etc.). Whenever compared with Qipit® and ScanR®, Tableau provided the best image segmentation and cropping.

Better board image enhancement and image binarization are currently under work in Tableau.

The Tableau code is freely available at: http://www.telematica.ee.ufpe.br/sources/Tableau.

9. Acknowledgements

The work reported herein was partly sponsored by CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico, Brazilian Government.

10. References

[1] D. Liebowitz and A. Zisserman. "Metric rectification for perspective images of planes". In Proc. of the Conference on C. Vision and Pattern Recognition, pp 482–488, 1998.

[2] George Wolberg, Digital Image Warping, 1990, pp 129-131, IEEE, ISBN 0-8186-8944-7.

[3] G. K. Myers, R. C. Bolles, Q.-T. Luong, J. A. Herson, and H. B. "Rectification and recognition of text in 3-D scenes". International Document Analysis and Recognition, 7(2+3) pp 147–158, July 2005.

[4] L. Jagannathan and C. V. Jawahar, "Perspective correction methods for camera based document analysis", pp. 148–154, CBDAR 2005/ICDAR 2005, Korea, 2005.

[5] P. Clark, M. Mirmehdi, "Recognizing Text in Real Scenes", IJDAR, Vol. 4, No. 4, pp. 243-257, 2002.

[6] P. Clark, M. Mirmehdi, "Location and Recovery of Text on Oriented Surf", SPIE CDRR VII, pp.267-277, 2000.

[7] R. C. Gonzalez and R. E. Woods, Digital Image Processing, 2nd ed. Prentice Hall, 2001.

[8] R. D. Lins, A. R. Gomes e Silva and G. P. da Silva. Enhancing Document Images Acquired Using Portable Digital Cameras. ICIAR 2007, LNCS, Springer Verlag, 2007.

[9] R. Gomes e Silva and R. D.Lins. Background Removal of Document Images Acquired Using Portable Digital Cameras. LNCS 3656, p.278-285, 2005.

[10] S. J. Lu, *et al*, "Perspective rectification of document etc.," Image and Vision Computing, V(23):541-553, 2005.

[11] ImageJ http://rsb.info.nih.gov/ij/

[12] JAI (Java Advanced Imaging). https://jai.dev.java.net.

[13] Qipit®. http://www.qipit.com

[14] Scanr®. http://www.scanr.com

Improving the Border Detection and Image Enhancement Algorithms in Tableau

Daniel Marques de Oliveira and Rafael Dueire Lins

Departamento de Eletrônica e Sistemas, Universidade Federal de Pernambuco
Recife – Pernambuco - Brazil

Abstract. Tableau is a software environment for processing images from teaching-boards acquired using portable digital cameras and cell-phones. This paper presents two new algorithms to improve the performance of Tableau. The first one enhances border detection while the second algorithm targets at image enhancement. Tableau has its performance compared to similar commercial softwares.

Keywords: Whiteboard images, border detection, portable cameras.

1 Introduction

Tableau [2] is a software environment to process images of teaching boards acquired using portable digital cameras. This simple tool provides a natural way to generate digital content for courses, respecting particular aspects of the group such as syllabus, class learning speed, teacher experience, regional content, local culture, etc.

The system consists of three parts. The first is database formation. As soon as images are transferred from the camera to the PC, information is collected to generate a simple database that will organize images for later content formation. Information such as teacher name, course name, discipline, subject, class number, group number, etc. are requested. The second module is for image processing. This module improves the image acquired in a number of ways: background removal, image segmentation, skew correction, image enhancement, etc. The third part of the processing environment deals with outputting the content. Three different ways are under development: printed handouts, webpage generation and slide production. Each of these media receives the information of the processed image part of the environment and makes it suitable to its best use.

ImageJ [12] is an open source image processing environment in Java developed by Wayne Rasband at the Research Services Branch, National Institute of Mental Health, Bethesda, Maryland, USA. It allows the insertion of plug-ins for special purposes. The current version of Tableau image processing routines [2] were developed as an ImageJ plug-in and besides its natural portability, it has shown suitable performance.

WhiteboardIt [4, 11, 9, 10] pioneered the research in board image processing and became a commercial product. Tableau is the first open-source environment with similar purposes. This paper focuses on improving two parts of the whiteboard

A. Campilho and M. Kamel (Eds.): ICIAR 2008, LNCS 5112, pp. 1111–1121, 2008.

processing on the Tableau environment. The first one addresses border detection, while the second targets at image enhancement. The new version has its performance compared with our implementation of the latest version of WhiteboardIt [9][10] (with the exception of its border detection algorithm for the reasons explained later on in this paper) and two recently released commercial softwares: Qipit® [14] and Scanr®[15]. Evidences are provided that the new version of Tableau produces similar quality images than those commercial products. Algorithms and tools were tested with 54 images taken with Nokia 6020 low-cost mobile phone, 16 with Motorola Z6 and 60 images were collected from the Internet.

2 Improvements in Boundary Detection

The problem of border detection is well-known in the literature [1][3][5]. References [6] [7] provide approaches for finding the boundaries of documents acquired with portable digital cameras. However, those algorithms have proved not suitable for treating whiteboard images. The unevenness of illumination due to environment conditions and the lack of fixed boundaries (as a whiteboard image may span throughout several photos) makes border detection much harder than on paper documents WhiteboardIt [9][10] uses a boundary detection algorithm based on Hough Transform; Experimenting with it makes one believe that it is robust to noise, but unfortunately neither its code is available, nor there are enough details in [9][10] to grant its implementation by readers.

2.1 The Original Tableau Border Detection Algorithm

The original Tableau reference [2] provides an algorithm for border detection on whiteboard images acquired using portable digital cameras that is explained below. The new algorithm presented herein is based on it. It has 3 steps: segmentation, control point finding and control point elimination. The first two steps are described as follows:

Tableau Step 1: Segmentation

1. Split the input image (ORI_IMG) into 4 regions: upper left, upper right, lower left and upper right.
2. Create a new binary image (DIF_IMG) of equal dimensions;
3. H_DIST and V_DIST are defined as functions of the image resolution. They correspond to the rounding-off of the integer part of 0.91% of the width and height in pixels of the original image, respectively. For instance, in the case of a 640x480 pixel image, H_DIST=6 and V_DIST=4.
4. DIF_IMG(x,y) is white if one (it is black otherwise):
    - The difference between each component of ORI_IMG(X,Y) and ORI_IMG(X±H_DIST,Y±V_DIST) is less than 10, or
    - The component wise gap of ORI_IMG(X±2.H_DIST,Y±2.V_DIST) and ORI_IMG(X,Y) is less than 10.

Tableau Step 2: Finding control points

In each direction, N equally spaced axes are defined to scan the binary image looking for control points from the center towards the borders. Each of those vertical axes is swept with a 7x1 pixel mask. Similarly, each horizontal axis is scanned with a 1x7 pixel mask. If four or more pixels under the mask are black, go to step 1, otherwise it continues the search. For the collected images, N=9.

1. Calculate colors ratio in the rectangle around current point (X, Y). The bottom border of the rectangle has the upper left corner located on (A,B) and the lower right corner on (C,D), where:

$$A = X - \left[\frac{IMG\_WIDTH}{N} \times \frac{1}{2}\right] \qquad B = Y - IMG\_HEIGHT \times INTERNAL\_CHECK$$

$$C = X + \left[\frac{IMG\_WIDTH}{N} \times \frac{1}{2}\right] \qquad D = Y + IMG\_HEIGHT \times EXTERNAL\_CHECK$$

   Where INTERNAL_CHECK=0.165% and EXTERNAL_CHECK=0.91%
2. If within rectangle (A,B,C,D) more than 65% of pixels are black, then (X,Y) is marked as a candidate to a control point for the bottom border. Otherwise, the algorithm moves outwards looking for another candidate.

2.2 The New Border Detection Algorithm

This section describes the new border detection algorithm incorporated into Tableau, which avoids the creation of a segmented image, thus it claims for less memory and provides much better performance. This is done by creating two cluster arrays: one for the vertical border candidates (top and bottom borders) and another for the horizontal borders. The total number of clusters is fixed to a value that is significantly smaller than the image dimensions. Thus, the memory cost of the new approach depends only on the sum of the image width and height times the cluster size, which is much less than the total image area. Alterations are performed to the first two steps of the previous algorithm for border detection. The third step remains unchanged.

The New Algorithm - Step 1: filling horizontal vertical lines cluster spaces

1. Split image width into N equally spaced clusters. Do the same for image height. The larger number of clusters implies on less sensitivity to rotational variation. However, the probability of finding wrong control points may increase. In this implementation, N= 9.
2. Create two arrays, one with size (N, WIDTH) and with size (N, HEIGHT). They are called V_BORDERS and H_BORDERS, respectively.
3. H_DIST and V_DIST are defined in the same way as step 3 of reference [2].
4. If one of the following condition holds, increment array element H_BORDERS[Xcluster][Y]:
   - The difference between each component of ORI_IMG(X,Y) and ORI_IMG(X,Y±V_DIST) is less than 8.
   - The componentwise gap between ORI_IMG(X,Y) and ORI_IMG(X,Y±2.V_DIST) is less than 16.
5. Similarly, if one the following condition holds, increment array element V_BORDERS[Ycluster][X].

- The difference between each component of ORI_IMG(X,Y) and ORI_IMG(X±H_DIST,Y) is less than 8.
- The componentwise gap between ORI_IMG(X,Y) and ORI_IMG(X±2.H_DIST,Y)is less than 16.

It is worth noting that this new approach separates vertical from horizontal borders, thus it becomes more robust against noise. The threshold for the second difference changes due to being further away from point (X,Y) and this may yield a stronger difference in illumination.

The New Algorithm - Step 2: Finding control points in the image

After the steps above are executed on the whole image, as explained earlier on, one has only the structure of the clusters. In this step, small modifications are done to try to get the same information of the previous approach with this structure. The scanning is done in a similar way, starting from the center towards the image borders. The steps of the second part for finding the bottom border can be seen bellow, the other borders can be found in a similar way:

1. For each X-cluster, set Y on image center.
2. Compute the amount of pixels set to "possible border" by adding the elements of the array within H_BORDERS[Xcluser][Y+V_DIST/2] and H_BORDERS[Xcluser][Y].
3. If the cluster area previously computed has 85% of pixels set to "possible border" one sets the current cluster as a control point, otherwise one increments the value of Y and returns to step 2.
4. Observe that V_DIST is the distance between (X,Y) and first vertical difference, and by summing up cluster area on the y-coordinate between Y and Y+V_DIST/2 – which is half than the whole border area – one can increase the density threshold to 85%, yielding a more robust result, thus less sensitive to noise.

Once the control points are found, the third step of the original algorithm is followed to perform a perspective correction. A comparison on the performance of both algorithms may be found in images presented in Figures 1 to 4.

Fig. 1. Original image

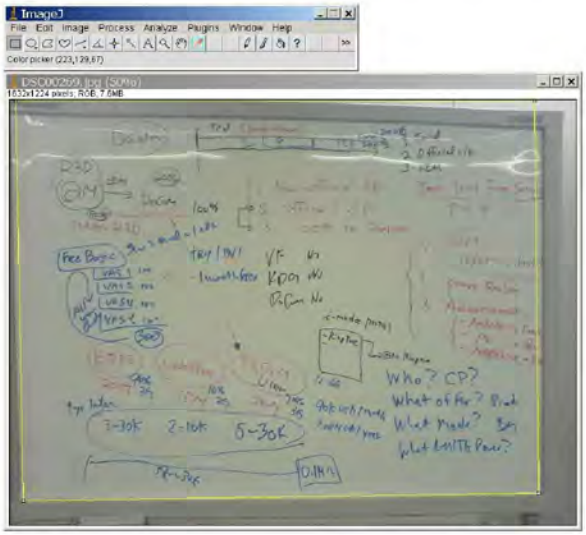


Fig. 2. Tableau border detection algorithm [2] applied to image of Figure 1

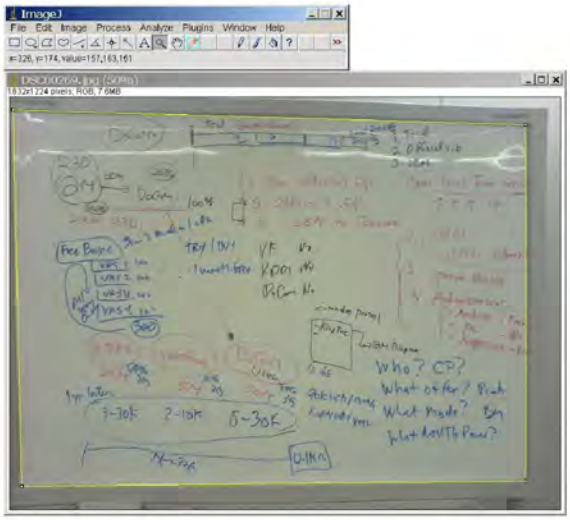


Fig. 3. Image obtained using the new border detection approach on image of Figure 1

One may observe that the new algorithm was able to better detect the whiteboard frame. The top right-hand-side of Figure 1 concentrates a higher luminosity than any other part of the same image. In that crucial area, the new algorithm (Figure 03) was able to find the border accurately, while the old one (Figure 04) overshoot it. Figures 10 and 11 provide the Qipit and Scanr processing of the image of Figure 01.

Fig. 4. Final image after perspective correction on the image shown in figure 3 followed by image crop

3 Tableau-WBIT Image Enhancement

The literature provides several algorithms for enhancing images [5]. References [2] and [4] address ways of improving the quality of whiteboard images. The first uses a Rayleigh filter with parameter $\lambda = 0.5$. References [9] and [10] describe an improved algorithm, summarized below:

1. Divide the whiteboard region into rectangular cells.
2. Sort the pixels in each cell by their luminance values. Average the colors of the pixels in the top 25 percentile; store this average in variable *Clight*.
3. Filter the colors of the cells by locally fitting a plane in the RGB space.
4. Re-map the value of each color channel of each pixel according to an S-shaped curve in equation 1 with S(*Cinput*/*Clight*). Where *Cinput* is the pixel value on original image. This paper suggests the value for p to be equal to 0.75. Function graph can be seen on Figure 5.

$$\text{if } (x >= 1),\ S(x) = 1;\ \text{otherwise},\ S(x) = 0.5 - 0.5 * \cos(\pi * x^{p}) \quad (1)$$

The implementation of this algorithm yielded better results than the classical methods known in the literature [5]. However, the algorithm has two problems. The first one is on step 3, which is needed when most of the cell pixels belong to pen strokes. It is not clear how to identify when such situation occurs to a particular cell. The second problem arises whenever the input image has areas with high illumination changes, turning out into an unpleasant image to the end-user.

A new algorithm based on WhiteboardIt [10] was developed and called Tableau-WBIT, since it borrows some key ideas from WhiteboardIt. The new algorithm is described bellow; these steps are applied for every pixel on the image which is represented as the central pixel:

1. Calculate cmp_radius as shown in equation 2. In the implementation described here RADIUS_RATIO=0.00375. This distance must be larger than the width of a pen stroke.

2. Get the color values of pixels in the region closer to the central pixel. This is done in a coarse way by getting values from points on 3 circumferences with radius: *cmp_radius*, *cmp_radius*2* and *cmp_radius*3*; the angle between points is about 22.5 degrees (see Figure 6 for further details).

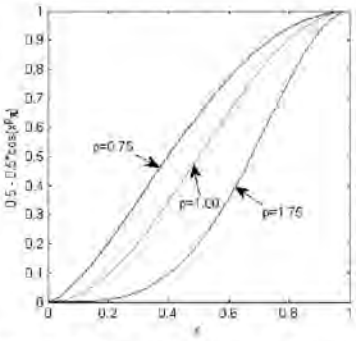


Fig. 5. S shape function with different p values

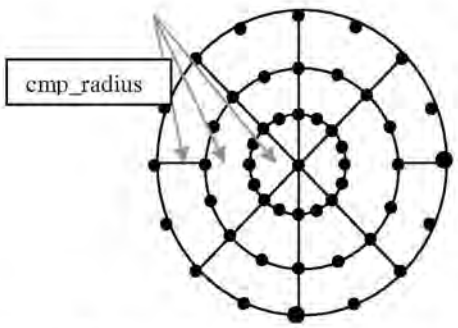


Fig. 6. Pixels positions used to estimate background, current pixel is at the center

A new algorithm based on WhiteboardIt [10] was developed and called Tableau-WBIT, since it borrows some key ideas from WhiteboardIt. The new algorithm is described bellow; these steps are applied for every pixel on the image which is represented as the central pixel:

3. Calculate cmp_radius as shown in equation 2. In the implementation described here RADIUS_RATIO equals 0.00375. This distance must be larger than the width of a pen stroke.
4. Get the color values of pixels in the region closer to the central pixel. This is done in a coarse way by getting values from points on 3 circumferences with radius: *cmp_radius*, *cmp_radius*2* and *cmp_radius*3*; the angle between points is about 22.5 degrees (look at Figure 6 for further details).
5. Map the color values of those pixels from the RGB space onto the HSL space. Place color values onto an increasing value ordered list by: saturation, hue and luminance. In the implementation here the HSL model was used with the following discrete scales: saturation (range 0-16), hue (range 0-359) and luminance (range 0-16). In this step, one aims to get the whiteboard background color value, but without the reflectance of the strobe flash or room light on the board surface.
6. Find the neighborhood of step 3 ordered list with size ((list total elements)/4) that has the lowest standard deviation of the grayscale value defined by equation 3.
7. The neighborhood found on step 4 may represent the whiteboard background. It is calculated by averaging RGB components of this neighborhood, the average result is the new *Clight*.
8. Apply in the same way the remapping with S(*Cinput*/*Clight*) using equation 1.

$$\text{cmp\_radius} = \text{RADIUS\_RATIO}*(\text{IMG\_WIDTH} + \text{IMG\_HEIGHT}) \quad (2)$$

$$\text{gray level} = 0.30*\text{red} + 0.59*\text{green} + 0.11*\text{blue} \quad (3)$$

In order to yield sharper pen-strokes, the value of p=1.75 was adopted here, instead of 0.75 as suggested by WhiteboardIt. Lower values of p yields softer pen strokes as output, due to the color mapping being positioned above line y = x as can be seen in figure 5. Whenever setting p value to 1.75 for WhiteboardIt, it increases the sensitivity to illumination variance. One could change step 3 of new algorithm by getting the top percentile of luminance values as described by second step of WhiteboardIt algorithm [6]. This modification yields strobe flash and room light "noise" sharper, instead of the desired pen-strokes image only (see Fig. 8).

Figures 7-11 show image results after applying the studied algorithms. Figure 7 shows the original WhiteboardIt algorithm applied to the perspective corrected image in Fig.4. Figure 8 shows the same image with the modified third step on the proposed algorithm as explained in previous paragraph. Figure 9 shows the result of the new algorithm. To allow a better visualization, the image was framed in black. For figures 7 to 9, the value of p is 1.75. Figures 10 and 11 present the QipIt [14] and Scanr [15] processing of the board image in Figure 01, respectively.

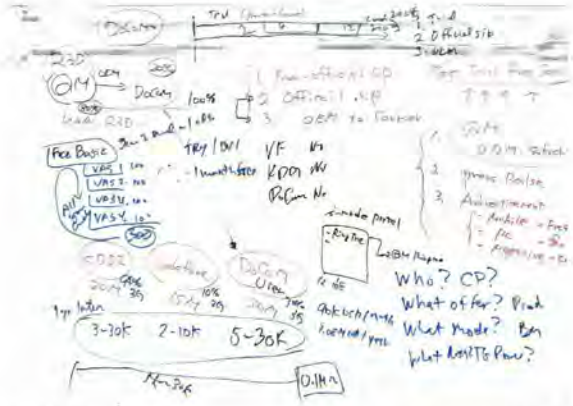


Fig. 7. Whiteboard enhancement applied on the image of Fig 4

Fig. 8. New algorithm with luminance 25% top percentile approach used on WhiteboardIt

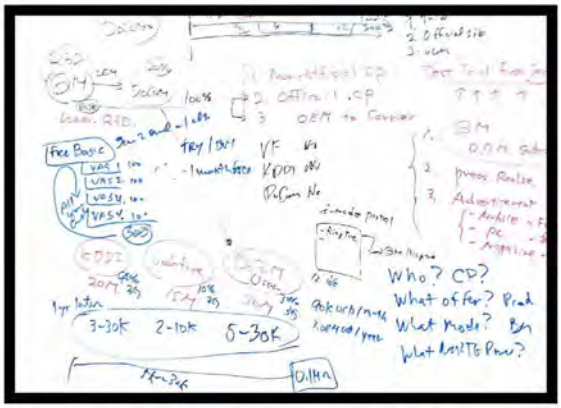


Fig. 9. New algorithm, Tableau-WBIT

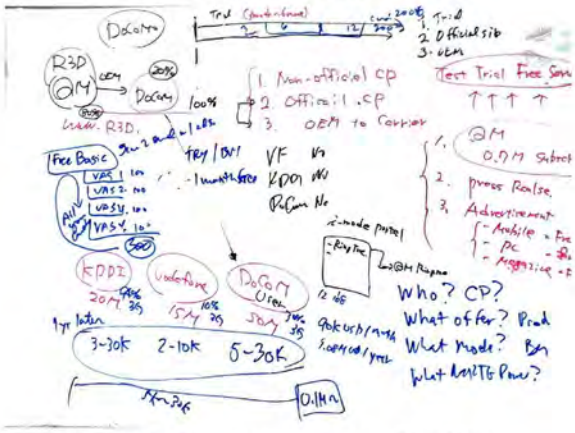


Fig. 10. Final image generated from Qipit [14]

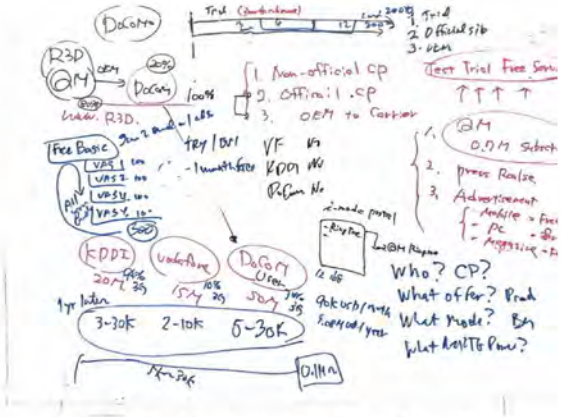


Fig. 11. Final image generated from Scanr [15]

4 Conclusions and Lines for Further Works

This paper introduced two important enhancements to Tableau, a software environment to process images of teaching boards acquired using portable digital cameras. The first one allows for better whiteboard boundary detection while the second one enhances image quality after the removal of the board background. The performance of Tableau was compared with compared with Qipit® [14] and ScanR® [15] two recently released commercial softwares for similar purposes and exhibited results of similar quality.

Image enhancement in Tableau is still possible and at this point the authors are addressing the problem of replacing the different hues of a color with a small number of colors (black, red, green and blue). This will allow a far more compact image representation with impact in storage and network transmission.

The current version of Tableau is implemented as an ImageJ plug-in and its source and executable codes are freely available by requesting to one of its authors.

Acknowledgements

The authors are grateful to Gilles Rochefort of Realeyes3d for providing some of the test images presented herein.

This research was partly sponsored by CNPq – Brazilian Government.

References

[1] Fernandes, L.A., Oliveira, M.M.: Real-time line detection through an improved Hough transform voting scheme. Pattern Recogn. 41(1), 299–314 (2008)

[2] Oliveira, D.M., Lins, R.D.: Tableau – Processing Teaching-board Images Acquired with Portable Digital Cameras. In: Proceedings of Second International IAPR Workshop on Camera-Based Document Analysis and Recognition, IAPR, pp. 79–86 (2007)

[3] Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. 8(6), 679–698 (1986)

[4] He, L., Liu, Z., Zhang, Z.: Why take notes? Use the whiteboard system. In: Inter. Conf. on Acoustics, Speech, and Signal Processing (ICASSP2003), April 2003, pp. 776–779 (2003)

[5] Gonzalez, R.C., Woods, R.E.: Digital Image Processing, 2nd edn. Prentice-Hall, Englewood Cliffs (2001)

[6] Lins, R.D., Gomes e Silva, A.R., da Silva, G.P.: Enhancing Document Images Acquired Using Portable Digital Cameras. In: Kamel, M., Campilho, A. (eds.) ICIAR 2007. LNCS, vol. 4633, pp. 1229–1241. Springer, Heidelberg (2007)

[7] Gomes e Silva, R., Lins, R.D.: Background Removal of Document Images Acquired Using Portable Digital Cameras. In: Kamel, M., Campilho, A.C. (eds.) ICIAR 2005. LNCS, vol. 3656, pp. 278–285. Springer, Heidelberg (2005)

[8] Russo, F.: Piecewise linear model-based image enhancement. EURASIP J. Appl. Signal Process. 1, 1861–1869 (2004)

[9] Liu, Z., Zhang, Z., He, L.: Whiteboard scanning and image enhancement. Digital Signal Processing 17(2), 414–432 (2007)

[10] Zhang, Z., He, L.: Notetaking with a camera: Whiteboard scanning and image enhancement. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, Montreal, Quebec, Canada, May 2004, vol. 3, pp. 533–536 (2004)
[11] Zhang, Z., et al.: System and method for whiteboard scanning to obtain a high resolution image. U.S. Patent 7,119,816, filed March 31, 2003 (issued October 10, 2006)
[12] ImageJ, http://rsb.info.nih.gov/ij/
[13] JAI (Java Advanced Imaging), https://jai.dev.java.net
[14] Qipit®, http://www.qipit.com
[15] Scanr®, http://www.scanr.com

APÊNDICE B – Sistema de cores CIE L*a*b*

O sistema de cores CIE L*a*b* foi desenvolvido pela CIE (*Commission internationale de l'Éclairage*, Comissão Internacional de Iluminação) em 2001 com o objetivo de representar as cores conforme elas são percebidas, sendo mais próxima da visão humana (CIE, 2001). O sistema foi desenvolvido com base em outro sistema de cores: o XYZ.

O sistema XYZ também desenvolvido pela CIE com o objetivo de ter uma representação perceptual das cores. Esse sistema constitui em uma transformação linear do sistema RGB. Os coeficientes da transformadas foram determinados empiricamente. No entanto a representação nesse sistema não é uniforme, pois a visão humana não apresenta uma resposta linear para as cores primárias: verde, vermelho e preto. A conversão do sistema RGB para o XYZ pode ser vista logo abaixo (CELENK, 1990).

$$X = 2{,}7690 \times R + 1{,}7518 \times G + 1{,}1300 \times B \quad \text{B.1}$$

$$Y = 1{,}0000 \times R + 4{,}5907 \times G + 0{,}0601 \times B \quad \text{B.2}$$

$$Z = 0{,}0000 \times R + 0{,}0565 \times G + 5{,}5943 \times B \quad \text{B.3}$$

O sistema CIE L*a*b* visa uniformizar através de transformações não-lineares. É um sistema tridimensional possuindo três componentes: luminância (L^*); vermelho/magenta e verde (a^*); amarelo e azul (a^*). O valor de luminância zero indica a cor preta e cem indica a cor branca, representado a quantidade de luz presente na cor. O valor de a^* negativo indica a cor verde, e o positivo magenta. E o valor de b^* negativo indica azul e positivo amarelo. Para calcular-se o valor da cor no espaço CIE L*a*b* calcula-se os valores no espaço XYZ que são as entradas para calcular as componentes L^*, a^* e b^*

$$L^* = 116 f\left(\frac{Y}{Y_b}\right) - 16 \quad \text{B.4}$$

$$a^* = 500\left[f\left(\frac{X}{X_b}\right) - f\left(\frac{Y}{Y_b}\right)\right] \quad \text{B.5}$$

$$b^* = 500\left[f\left(\frac{Y}{Y_b}\right) - f\left(\frac{Z}{Z_b}\right)\right] \quad \text{B.6}$$

, onde $f(t)$ é

$$f(t) = \begin{cases} \sqrt[3]{t} & t > \left(\frac{6}{29}\right)^3 \\ \frac{1}{3}\left(\frac{6}{29}\right)^2 t + \frac{4}{29} & senão \end{cases} \quad \text{B.7}$$

A função f em dois domínios é necessária para evitar que a derivada de $\sqrt[3]{t}$ ao se aproximar de $t = 0$, tenda a infinito. Os valores de X_b, Y_b e Z_b são os valores do branco de referência no sistema XYZ calculados previamente.

Muitas aplicações usam o sistema CIE L*a*b* em termos da matiz e crominância, ao invés dos valores a* e b*. A matiz define um ângulo formado pela tangente de a* e b*. Cores com mesmo valor de ângulo são diferenciadas pela luminância (componente L*), que identifica a quantidade de luz na cor, e crominância, que é a medida de pureza de uma cor. No caso das cores acromáticas, o valor da crominância é zero. A crominância é calculada pela raiz quadrada da soma dos quadrados de a* e b*.

Para maiores informações sobre o sistema CIE L*a*b* consulte-se (CELENK, 1990; FAIRCHILD, 2008; JAIN, 1989).

APÊNDICE C – Padrão de diferença de cores CIEDE2000

A CIE apresentou em 2001 um novo padrão de diferença entre cores, o CIEDE2000 (CIE, 2001). Esse padrão foi uma evolução dos padrões CIE76 e CIE94. Este apêndice tem como objetivo explicar brevemente o padrão CIEDE2000. Ele obtém a diferença a partir de uma série de equações, no entanto não será explicado como as elas foram obtidas. Como se optou pelas definições das equações a partir das modificações proposta por SHARMA e seus colegas (SHARMA *et al*, 2004), pois a especificação original apresenta perda de informação no cálculo de alguns dos seus valores. Outros detalhes também podem ser vistos em (NAKAYAMA and IKEDA, 2004).

A diferença se baseia na notação do espaço L*a*b* em termos de luminância, matiz (*hue*) e crominância, conforme visto no apêndice anterior. Para cada diferença entre esses termos, atribuem-se os pesos k_L, k_H, e k_C, respectivamente. A equação resumida denotada por ΔE_{00} – que é a diferença CIEDE2000 – pode ser vista logo abaixo:

$$\Delta E_{00} = \sqrt[2]{\left(\frac{\Delta L'}{k_L S_L}\right)^2 + \left(\frac{\Delta C'}{k_C S_C}\right)^2 + \left(\frac{\Delta H'}{k_H S_H}\right)^2 + R_T \left(\frac{\Delta C'}{k_C S_C}\right)\left(\frac{\Delta H'}{k_H S_H}\right)} \qquad \text{C. 1}$$

Portanto, se faz necessário mostrar como os valores de $\Delta X'$, S_X e R_T são obtidos conforme o desenvolvimento a seguir.

Sejam duas cores no espaço CIE L*a*b* com valores (L_1^*, a_1^*, b_1^*) e (L_2^*, a_2^*, b_2^*). Primeiro calcula-se os valores de C_i' e h_i', que é a crominância e matiz, sabendo que $i \in \{1,2\}$.

$$C_{i,ab}^* = \sqrt{(a_i^*)^2 + (b_i^*)^2} \qquad \text{C. 2}$$

$$\overline{C_{ab}^*} = \frac{C_{1,ab}^* + C_{2,ab}^*}{2} \qquad \text{C. 3}$$

$$G = \frac{1}{2}\left(1 - \sqrt{\frac{\overline{C_{ab}^*}^7}{\overline{C_{ab}^*}^7 + 25^7}}\right) \qquad \text{C. 4}$$

$$a_i' = (1 + G)a_i^* \qquad \text{C. 5}$$

$$C_i' = \sqrt{(a_i')^2 + (b_i^*)^2} \qquad \text{C. 6}$$

$$h_i' = \begin{cases} 0 & , \quad se\ a_i' = b_i^* = 0 \\ \tan^{-1}(b_i^*, a_i') & , \quad senão \end{cases} \qquad \text{C. 7}$$

Depois os valores de $\Delta L'$, $\Delta C'$ e $\Delta H'$.

$$\Delta L' = L_1^* - L_2^* \qquad \text{C. 8}$$

$$\Delta C' = C_1' - C_2' \qquad \text{C. 9}$$

$$\Delta h' = \begin{cases} 0 & C_1'C_2' = 0 \\ \Delta h(h_1', h_2') & C_1'C_2' \neq 0 \end{cases} \qquad \text{C. 10}$$

$$\Delta h(h_1', h_2') = \begin{cases} h_2' - h_1' & |h_1' - h_2'| \leq 180° \\ (h_2' - h_1') - 360 & (h_2' - h_1') > 180° \\ (h_2' - h_1') + 360 & (h_2' - h_1') < -180° \end{cases} \qquad \text{C. 11}$$

$$\Delta H' = 2 \times \sqrt{C_1'C_2'} \sin\left(\frac{\Delta h'}{2}\right) \qquad \text{C. 12}$$

O próximo passo é calcular os valores de entrada para ΔE_{00}.

$$\bar{L}' = \frac{L_1^* + L_2^*}{2} \qquad \text{C. 13}$$

$$\bar{C}' = \frac{C_1' + C_2'}{2} \qquad \text{C. 14}$$

$$\bar{h}' = \begin{cases} h_2' + h_1' & C_1'C_2' = 0 \\ \overline{h_{\neq}}'(h_1', h_2') & C_1'C_2' \neq 0 \end{cases} \qquad \text{C. 15}$$

$$\overline{h_{\neq}}' = \begin{cases} \dfrac{h_2{}' + h_1{}'}{2} & |h_1{}' - h_2{}'| \leq 180° \\ \dfrac{(h_2{}' + h_1{}') - 360}{2} & |h_1{}' - h_2{}'| > 180°; (h_1{}' + h_2{}') \geq 360° \\ \dfrac{(h_2{}' + h_1{}') + 360}{2} & |h_1{}' - h_2{}'| > 180°; (h_1{}' + h_2{}') < 360° \end{cases} \quad \text{C. 16}$$

$$T = 1 - 0{,}17 \times \cos\left(\bar{h}' - 30°\right) + 0{,}24 \times \cos\left(2\bar{h}'\right) + 0{,}32 \times \cos\left(3\bar{h}' + 6°\right) - 0{,}2 \times \cos(4\bar{h}' - 63°) \quad \text{C. 17}$$

$$\Delta\theta = 30 \times exp\left(-\left[\frac{\bar{h}' - 275°}{25}\right]\right) \quad \text{C. 18}$$

$$R_C = 2 \times \sqrt{\frac{\bar{C}'^7}{\bar{C}'^7 + 25^7}} \quad \text{C. 19}$$

$$S_L = 1 + \frac{0{,}015(\bar{L}' - 50)}{\sqrt{20 + (\bar{L}' - 50)^2}} \quad \text{C. 20}$$

$$S_C = 1 + 0{,}045\bar{C}' \quad \text{C. 21}$$

$$S_H = 1 + 0{,}015\bar{C}'T \quad \text{C. 22}$$

$$R_T = -\sin(2\Delta\theta)\, R_C \quad \text{C. 23}$$

Daí pode-se obter os valores de ΔE_{00}. Conforme os cálculos acima é possível fazer as seguintes observações:

1. O padrão CIE considera que uma das cores pode ter crominância 0, que é o caso dos tons de cinza. O que pode ser observado na equação do cálculo do $\Delta h'$, que vai ser zero caso uma das cores tenha crominância nula. No âmbito dos quadros didáticos isso é de extrema importância, pois preto e branco "puros" possuem crominância 0, que é a cor do fundo resultante. O fundo de uma imagem dos

quadros brancos e pretos na maioria das vezes possui um valor de crominância muito baixo, mas diferente de zero.

2. A especificação inicial do padrão calcula $\boldsymbol{h_i}'$ como sendo $\mathbf{tan^{-1}}\left(\frac{b_i^*}{a_i'}\right)$, SHARMA e seus colegas (2005) sugerem o cálculo da forma $\tan^{-1}(b_i^*, a_i')$, pois abrangem 4 quadrantes.
3. A computação da diferença dos ângulos de matizes em $\Delta h'$ é modificada para que a diferença entre um ponto que está na origem do plano (a_i',b_i^*) e qualquer outro ponto seja zero. SHARMA e seus colegas explicam que os pontos cinzas estão localizados na origem, logo não há diferença na matiz das cores.

Existem outros comentários presentes na referência (SHARMA *et al*, 2004) que fogem ao escopo deste Apêndice, que tratam de ambigüidades e algumas aspectos para o desenvolvimento correto do padrão que não estão descritas no relatório original presente em (CIE, 2001).

APÊNDICE D – DVD em anexo

O DVD em anexo contém:

- Esta dissertação em PDF;
- Artigos do Tableau publicados em PDF;
- Imagens originais, sem nenhuma modificação (pasta imagens/original);
- Imagens pré-processadas, cortadas com perspectiva corrigida (pasta "imagens/cortada-amao");
- Resultados de processamento processamento de realce e detecção de borda (pasta "resultados");
- Executável do ImageJ integrado com o Tableau versão 1.0 (arquivo "tableau/ImageJTableau.zip");
- Código fonte do Tableau como Plugin do ImageJ (pasta "codigofonte") versão 1.0;
- Instalador do ambiente Java 1.5, não é necessário caso o usuário já tenha instalado uma versão igual ou superior a 1.5 (arquivo "downloads/jdk-1_5_0_17-windows-i586-p.exe").

Para usar o ambiente Tableau, descompacte o arquivo "tableau/ImageJTableau.zip" e execute o arquivo ImageJ.exe.

Observação: As imagens resultantes e originais das pastas "cameras" e "celular" foram compactadas no formato ZIP devido a restrição do sistema de arquivos do DVD com relação a nomes de arquivos.