COPPE/UFRJ

CORREÇÃO DE RUÍDO DE PADRÃO FIXO EM VÍDEO INFRAVERMELHO

Daniel Rodrigues Pipa

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientadores: Eduardo Antônio Barros da Silva
Carla Liberal Pagliari

Rio de Janeiro
Dezembro de 2008

CORREÇÃO DE RUÍDO DE PADRÃO FIXO EM VÍDEO INFRAVERMELHO

Daniel Rodrigues Pipa

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

Prof. Eduardo Antônio Barros da Silva, Ph.D.

Prof. Carla Liberal Pagliari, Ph.D.

Prof. Paulo Sergio Ramirez Diniz, Ph.D.

Prof. José Antonio Apolinário Jr., D.Sc.

RIO DE JANEIRO, RJ – BRASIL

DEZEMBRO DE 2008

Pipa, Daniel Rodrigues

Correção de ruído de padrão fixo em vídeo infravermelho/Daniel Rodrigues Pipa. – Rio de Janeiro: UFRJ/COPPE, 2008.

XVII, 105 p.: il.; 29, 7cm.

Orientadores: Eduardo Antônio Barros da Silva

Carla Liberal Pagliari

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2008.

Referências Bibliográficas: p. 102 – 105.

1. Processamento de sinais. 2. Vídeo infravermelho. 3. Correção de ruído de padrão fixo. 4. Gradiente descendente. 5. RLS. I. Silva, Eduardo Antônio Barros da *et al.*. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

À minha família.

Agradecimentos

O autor agradece ao Centro de Pesquisa e Desenvolvimento da Petrobras (CENPES) pelo incentivo e permissão da realização do trabalho.

Agradecimentos também à COPPE/UFRJ, ao IME e ao apoio do Centro Tecnológico do Exército-CTEx e da FINEP, sob o convênio no. 2645/06 FINEP/FAPEB/CTEx.

Agradecimentos especiais aos meus orientadores Eduardo Antônio Barros da Silva e Carla Liberal Pagliari por sua paciência, incentivo, correções e troca de ideias inspiradoras.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

CORREÇÃO DE RUÍDO DE PADRÃO FIXO EM VÍDEO INFRAVERMELHO

Daniel Rodrigues Pipa

Dezembro/2008

Orientadores: Eduardo Antônio Barros da Silva
Carla Liberal Pagliari

Programa: Engenharia Elétrica

IRFPA's (*infrared focal plane arrays*) são sensores ópticos sensíveis à radiação infravermelha que compõem sistemas de imagem e vídeo infravermelho. Os elementos sensíveis são dispostos em forma de matriz no plano focal de um sistema de lentes, daí o seu nome. Apesar dos avanços recentes nessa área, um problema comum à tecnologia IRFPA é o ruído de padrão fixo (*fixed-pattern noise* ou FPN), também chamado de não uniformidade espacial.

Tal ruído se caracteriza pela lenta variação temporal, independência entre os pixels e por possuir componentes aditivas, chamadas de desvio ou *bias*, e multiplicativas, chamadas de ganho ou *gain*. O FPN compromete consideravelmente a qualidade de um sistema de imagens infravermelhas.

Este trabalho apresenta o desenvolvimento de quatro algoritmos recursivos de correção de ruído de padrão fixo. Os algoritmos são baseados em gradiente descendente e algoritmos RLS, e refinam, a cada novo quadro, a estimativa do desvio e do ganho para cada pixel. Após alguns quadros já é possível estimar o FPN de modo a removê-lo das imagens e melhorar substancialmente sua qualidade.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

FIXED-PATTERN NOISE CORRECTION IN INFRARED VIDEO

Daniel Rodrigues Pipa

December/2008

Advisors: Eduardo Antônio Barros da Silva
Carla Liberal Pagliari

Department: Electrical Engineering

IRFPA's (infrared focal plane arrays) are optical sensors designed to operate in the infrared spectrum. Despite recent advances in this technology, a common problem present in all IRFPA's is the fixed-pattern noise (FPN), also known as spatial non uniformity.

This noise is characterized by slow temporal drift, pixel independence and additive and multiplicative components, referred to as bias and gain, respectively. The FPN severely affects the quality of an infrared imaging system.

This work presents the development of four recursive algorithms for FPN correction. The algorithms are based on gradient descent and RLS algorithms. The FPN bias and gain estimates are refined at each new frame. After few iterations the estimates are sufficient to improve considerably the image quality.

Sumário

Lista de Figuras

Lista de Tabelas

Notações

1. k representa tempo discreto e é utilizados como índices subscritos de uma variável, por exemplo x_k, $\mathbf{x}_k$ ou $\mathbf{A}_k$.

2. Vetores coluna são representados por letras minúsculas em negrito, por exemplo $\mathbf{x}$.

3. Matrizes são representadas por letras maiúsculas em negrito, por exemplo $\mathbf{A}$.

4. O elemento a_{ij} de uma matriz ou vetor pode ser representado por $[\mathbf{A}]_{ij}$.

5. $\triangleq$ representa "igual por definição".

6. Estimativas são denotadas por acento circunflexo, por exemplo, $\hat{\mathbf{x}}$.

7. A operação de gradiente em relação a um vetor ou matriz é representado por $\nabla_{\mathbf{x}} f(\mathbf{x})$.

8. O produto Hadamard ou elemento-por-elemento em matrizes ou vetores é representado por $\mathbf{v} = \mathbf{f} \circ \mathbf{g}$, ou seja, $[\mathbf{v}]_i = [\mathbf{f}]_i [\mathbf{g}]_i$.

9. A divisão Hadamard ou elemento-por-elemento em matrizes ou vetores é representada por $\mathbf{v} = \left(\frac{\mathbf{f}}{\mathbf{g}}\right)$, ou seja, $[\mathbf{v}]_i = \frac{[\mathbf{f}]_i}{[\mathbf{g}]_i}$.

10. A função $\text{diag}\,(a_1, \cdots, a_N)$ define a matriz diagonal quadrada cujos elementos da diagonal principal são $(a_1, \cdots, a_N)$.

Acrônimos

FPN Ruído de padrão fixo (*Fixed-pattern noise*)

NUC Correção de não uniformidade (*Non uniformity correction*)

IR Infravermelho (*Infrared*)

IRFPA *Infrared focal plane arrays*

SSIM *Structural similarity*

MSE *Mean square error*

PSNR *Peak signal-to-noise ratio*

LMS *Least-mean-square*

RLS *Recursive least-squares*

Capítulo 1

Introdução

Na atualidade, câmeras infravermelhas e vídeos na faixa do infravermelho são usados em inúmeras áreas como ensaios não-destrutivos para verificação da integridade de equipamentos, visão noturna, segurança, reconhecimento e vigilância aeroespacial, imagens térmicas astronômicas e aplicações militares [2].

No que tange os sensores, as últimas décadas presenciaram o aparecimento de dispositivos cada vez mais precisos e baratos. As FPA's, ou *focal plane arrays*, são sensores de imagem que consistem de uma matriz de sensores ópticos localizada no plano focal de um sistema de lentes. A sua aparição possibilitou a construção de dispositivos para aquisição de imagens e vídeos que são registrados diretamente em formato digital.

Na faixa do infravermelho, as IRFPA's (*infrared focal plane arrays*) têm se tornado o mais proeminente detector usado em sistemas de imagens nos últimos anos. Seu vasto uso é atribuído aos avanços na tecnologia de sensores de estado sólido, que permitiram compacidade, baixo custo e alto desempenho.

Sabe-se que um problema comum a todos os sensores IRFPA é o ruído de padrão fixo (*fixed-pattern noise* ou FPN), também chamado de não uniformidade espacial. De fato, o FPN continua sendo um sério problema apesar dos avanços recentes nessa tecnologia. A origem deste ruído é atribuída ao fato de cada detector da matriz, ou seja, cada pixel possuir uma variação no processo de fabricação. Em outras palavras, cada pixel do detector responde de maneira diferente à mesma quantidade de radiação incidente. O ruído FPN se manifesta aleatoriamente no espaço e está presente em todos os quadros de um vídeo infravermelho independentemente da cena

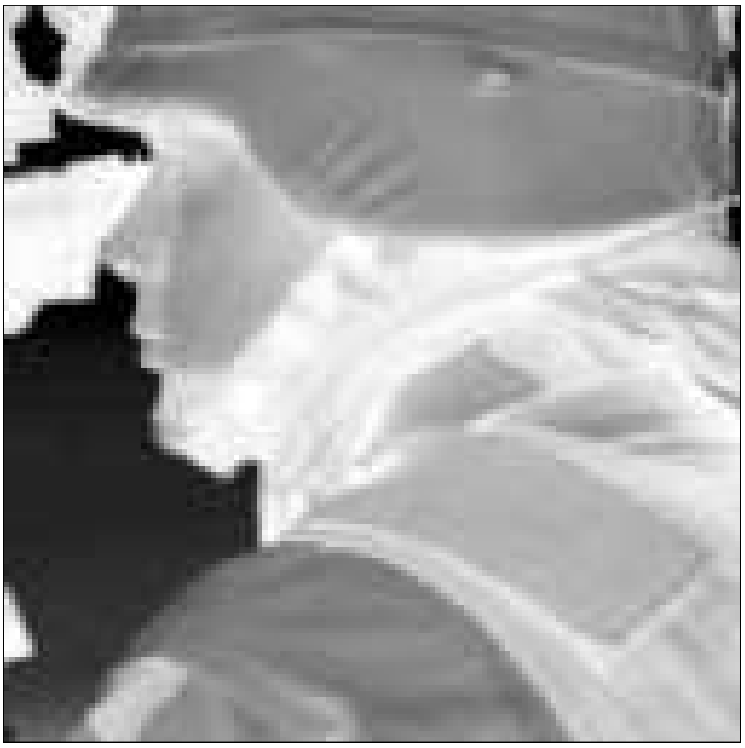

Figura 1.1: Quadro original de um vídeo infravermelho.

ou movimento [2].

Outra característica do ruído FPN é sua lenta variação temporal durante o funcionamento do sensor. Ele pode se tornar significativo numa ordem de grandeza de 30 segundos após uma calibração [3]. Esse desvio temporal é atribuído a variações na temperatura do sensor, material de fabricação, ruído eletrônico de leitura, controle automático de ganho, entre outros. Portanto, uma única calibração é ineficaz e o problema requer estimativa e compensação contínuas durante a operação da câmera [2].

Embora a verdadeira resposta das IRFPA's seja não-linear, ela é em geral modelada linearmente como função da radiância, um ganho e um desvio (*bias*) [2, 4] como

$$y_k(i,j) = a(i,j)L_k(i,j) + b(i,j), \tag{1.1}$$

onde $y_k(i,j)$ é a saída da câmera referente ao pixel (i,j) no instante k, $a(i,j)$ é o ganho associado ao pixel (i,j), $L_k(i,j)$ é a radiância incidente no elemento sensor (i,j) no instante k e $b(i,j)$ é desvio ou *bias* associado ao pixel (i,j).

A Figura 1.1 ilustra um quadro de um vídeo infravermelho sintético onde não existe FPN.

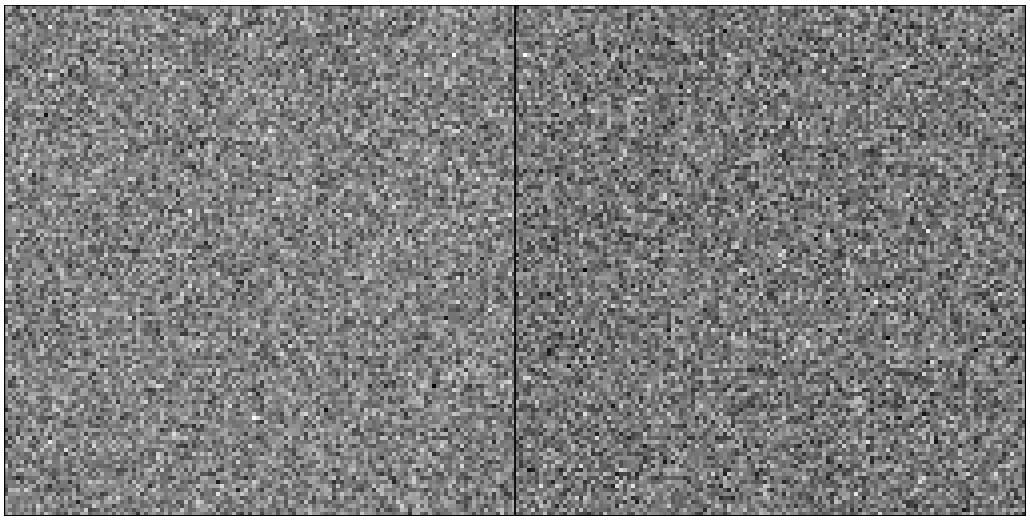

Figura 1.2: Ruído de padrão fixo aditivo e multiplicativo.

A Figura 1.2 mostra um exemplo de FPN de forma isolada, sendo na esquerda a parcela aditiva ou desvio e na direita a parcela multiplicativa ou ganho.

A Figura 1.3 mostra o mesmo quadro contaminado com o FPN da Figura 1.2 segundo o modelo da equação (1.1).

A Figura 1.4 mostra um vídeo real contaminado com ruído de padrão fixo. Nota-se que o FPN degrada consideravelmente a qualidade da imagem.

Este trabalho apresenta o desenvolvimento de quatro algoritmos recursivos de correção de ruído de padrão fixo. Os algoritmos são baseados em gradiente descendente e método de Newton e refinam, a cada novo quadro, a estimativa do desvio e do ganho para cada pixel. Após alguns quadros já é possível estimar o FPN e melhorar substancialmente a qualidade da imagem.

No Capítulo 2 são fornecidas as bases da termometria. Começa-se com definições elementares como o conceito de temperatura e aborda-se gradualmente esse assunto até a apresentação das IRFPA's.

O Capítulo 3 faz uma revisão dos métodos existentes de correção de FPN. São apontadas vantagens e desvantagens de cada abordagem e é indicada a classe na qual os métodos propostos neste trabalha se enquadram.

O Capítulo 4 apresenta o desenvolvimento de um algoritmo para a correção de FPN através da estimativa do desvio de cada pixel. É adotado um modelo simplificado onde somente o desvio é considerado. No mesmo capítulo é apresentado o algoritmo de estimativa de movimento global utilizado, cujas vantagens são simpli-

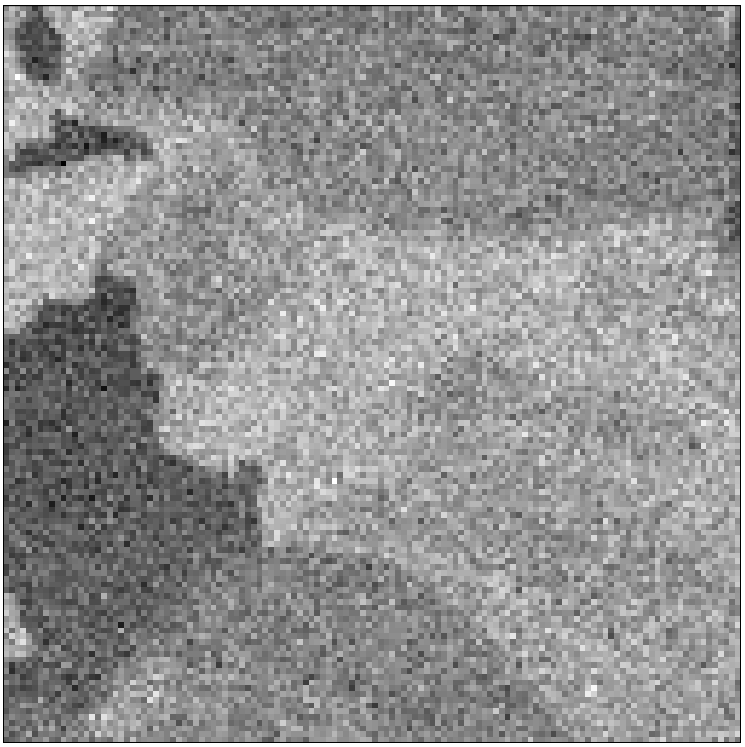

Figura 1.3: Quadro contaminado com ruído de padrão fixo.

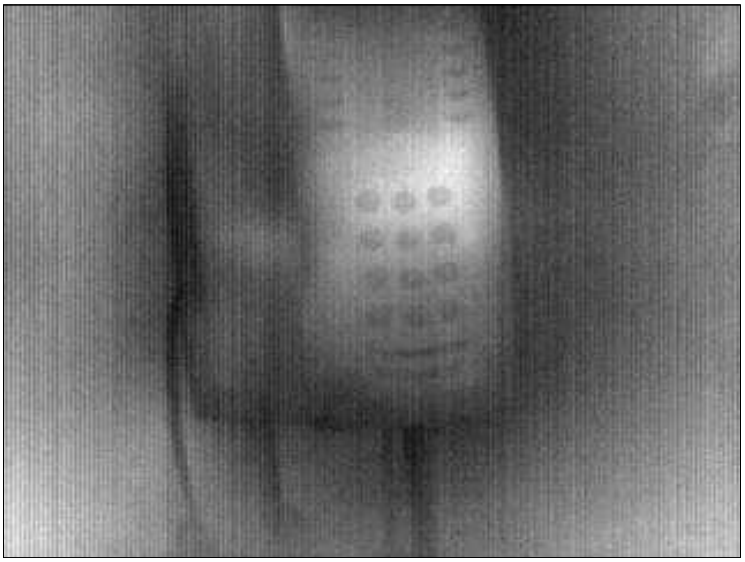

Figura 1.4: Vídeo real com ruído de padrão fixo.

cidade e rapidez computacional.

O Capítulo 5 também trabalha com um modelo simplificado, porém somente o ganho é considerado. Após o desenvolvimento e testes do algoritmo para correção de ganho, são apresentados resultados da aplicação conjunta da correção de desvio e ganho.

O Capítulo 6 aborda o problema do FPN através do modelo completo, considerando um desvio e um ganho para cada pixel. A utilização de uma matriz diagonal de ganhos possibilitou a aplicação do gradiente à função erro de estimativa tanto para o desvio quando para o ganho. O método foi batizado de Tensorial, pois sua solução envolve conceitos de tensores.

No Capítulo 7 é feita uma adaptação do algoritmo Tensorial para os moldes RLS (*recursive least-squares*), chamado então de Tensorial-RLS. O processo também pode ser visto como uma aplicação do método de Newton onde são feitas estimativas de segunda ordem como a matriz Hessiana. Os algoritmos desta família são conhecidos por sua convergência mais rápida quando comparados aos métodos baseados somente em gradiente descendente.

Finalmente no Capítulo 8 é feita uma comparação de todos os algoritmos apresentados neste trabalho. São feitas discussões sobre cada método apontando suas vantagens e desvantagens. São apresentados também os resultados das aplicações dos algoritmos a vídeos reais contendo FPN. Como neste caso a avaliação de qualidade é subjetiva, optou-se por apresentar os resultados em vídeos reais de forma agrupada. Espera-se, assim, facilitar a comparação e o julgamento da qualidade de correção do FPN obtida por cada um dos métodos.

Os Apêndices A, B, C e D contêm, respectivamente, a lista de publicações geradas até o momento pelo trabalho, as equações da métrica de qualidade de imagem SSIM utilizada nos ensaios, a vetorização dos algoritmos tensoriais apresentados nos Capítulos 6 e 7, e discussões sobre a inicialização das variáveis nos algoritmos.

1.1 Conclusões

Este capítulo fez uma breve introdução ao problema de FPN (*fixed-pattern noise*) ou ruído de padrão fixo em vídeos na faixa do infravermelho. Apresentou-se o modelo de

resposta adotado para as IRFPA's amplamente empregado na literatura. O modelo será usado como base para o desenvolvimento dos algoritmos de correção de FPN.

Em seguida, foi feita uma descrição de como está organizado este trabalho e o que é encontrado em cada capítulo. O Capítulo 2 em particular trata das bases da termometria de radiação servindo como motivação para o assunto. A sua leitura traz informações interessantes, porém não essenciais para o entendimento do resto do trabalho.

Capítulo 2

Termometria de radiação

O presente capítulo descreve de forma sucinta as bases da termometria de radiação e a tecnologia de sensores usada atualmente. Primeiramente são feitas algumas definições como temperatura, radiância e corpo negro. Dá-se importância especial a lei de Planck, uma das bases para se avaliar a temperatura de um corpo a partir de sua emissão eletromagnética. São comentadas outras definições como emissividade e refletividade e como as condições do ambiente influem na avaliação da temperatura.

Na Seção 2.3 é discutida a tecnologia atual de sensores infravermelhos, notavelmente as IRFPA's – *Infrared Focal Plane Arrays* – e os problemas intrínsecos desta tecnologia.

Referências sobre termometria de radiação podem ser encontradas em [5, 6, 7, 8] e sensores infravermelhos em [2, 3, 4, 5, 9].

2.1 Introdução

A medição da temperatura em processos industriais é de suma importância do ponto de vista de gerenciamento de energia, produtividade, qualidade de produto e integridade de equipamentos. É igualmente importante em aplicações científicas, onde se deseja confirmar as relações entre teoria e prática. Métodos de radiometria[1] são usados quando o contato com o objeto em questão é indesejado ou impossível. Estas condições ocorrem, por exemplo, quando o objeto se encontra em movimento, está

[1] Radiometria é a ciência que estuda medições de radiação eletromagnética, qualquer que seja a sua origem, incluindo a luz visível.

inacessível ou poderia ser danificado pelo contato, quando a temperatura do objeto pode ser alterada pelo contato do sensor ou quando a temperatura do objeto é tão alta que danificaria o sensor de contato. A medição de temperatura por métodos radiométricos é denominada termometria de radiação.

Um termômetro de radiação é um radiômetro[2] calibrado para indicar a temperatura de um radiador ideal, conhecido como corpo negro (ver Subseção 2.2.3). Fazendo-se uma análise do espectro eletromagnético emitido pela superfície sob medição e conhecendo-se sua emissividade (ver Subseção 2.2.4), é possível estimar qual a temperatura do objeto [6].

2.2 Radiação térmica

2.2.1 Introdução

A temperatura T de um sistema pode ser definida como uma quantidade relacionada à energia cinética média das partículas no referencial do centro de massa[3] do sistema. Assim, a temperatura é definida independentemente do movimento do sistema em relação ao observador [8]. Ou seja, trata-se de um parâmetro físico que, sob o ponto de vista microscópico, mede a energia cinética associada ao movimento aleatório das partículas que compõem a matéria. Supondo que todas as partículas de um sistema têm a mesma massa, pode-se exprimir a energia cinética média por

$$\bar{E} = \frac{1}{2}\, m \left(\frac{1}{N} \sum_{i=1}^{N} v_i^2 \right), \tag{2.1}$$

onde m é a massa de cada partícula em kg, N é o número total de partículas e v_i é a velocidade de cada partícula em $\mathrm{m{\cdot}s^{-1}}$ no referencial do centro de massa.

Com base na definição acima, a temperatura deveria ser expressa em joules/partícula, contudo é costume exprimi-la em graus célsius [8]. A escala de temperatura usada em física é a escala absoluta, o kelvin, representado pela letra K. Um kelvin corresponde a aproximadamente $1,38 \times 10^{-23}\,\mathrm{J}$ por partícula.

O valor médio da energia trocada entre um sistema e suas vizinhanças devido a trocas individuais de energia que ocorrem como um resultado das colisões entre as

[2]Nome genérico de sensores de radiação eletromagnética.

[3]Definindo-se a velocidade no referencial do centro de massa, a contribuição da energia cinética translacional para temperatura é eliminada.

moléculas do sistema e as moléculas das vizinhanças é chamado de calor Q, sempre que não possa ser expresso macroscopicamente [8].

2.2.2 Processo de transferência de calor

A transferência de calor pode ocorrer, de maneira genérica, por condução, radiação, convecção ou uma combinação destas. A condução térmica se dá pela interação de partículas de duas substâncias com temperaturas diferentes. Ocorre a propagação de calor através de choques entre as partículas sem transporte de massa. Na convecção ocorre transferência de calor através do transporte de massa caracterizado pelo movimento de um fluido devido a sua diferença de densidade. A transferência de calor por radiação ocorre quando a superfície do material aquecido emite energia por meio de ondas eletromagnéticas e estas atingem outro corpo. Tais ondas possuem frequência abaixo da luz vermelha visível, daí o nome radiação infravermelha.

A descoberta da radiação infravermelha ocorreu em 1800, quando o Sir William Herschel essencialmente repetiu o famoso experimento do prisma de Newton e detectou calor na região logo abaixo do espectro visível [9].

Radiância e radiância espectral

Radiância é uma medida que descreve a taxa de energia eletromagnética, portanto transferência de calor por radiação, que é emitida por uma determinada área $\mathrm{d}A$ em uma determinada direção (θ, ϕ) e passa através de uma superfície $\mathrm{d}A_n$ determinada por um ângulo sólido $\mathrm{d}\omega$, como ilustra a Figura 2.1. Ela é representada pela letra L, sua unidade é $\mathrm{W{\cdot}sr^{-1}{\cdot}m^{-2}}$ e é definida por

$$L = \frac{\mathrm{d}^2\Phi}{\mathrm{d}A\,\mathrm{d}\omega\cos\theta}, \tag{2.2}$$

onde Φ é o fluxo emitido ou potência em W, θ é o ângulo entre a normal à superfície e a direção desejada, A é a área da superfície em $\mathrm{m^2}$ e ω é o ângulo sólido em sr[4].

A radiância caracteriza emissão ou reflexão em toda faixa do espectro, enquanto a radiância espectral, denotada por L_λ é uma medida em função de comprimento de onda. A unidade da radiância espectral é o $\mathrm{W{\cdot}sr^{-1}{\cdot}m^{-3}}$.

[4]sr significa esferorradiano ou esterradiano e é a medida SI para ângulo sólido. Trata-se de uma grandeza adimensional obtida por $\omega = \frac{A_n}{r^2}$, onde ω é o ângulo sólido, A_n é área da superfície e r o raio da esfera.

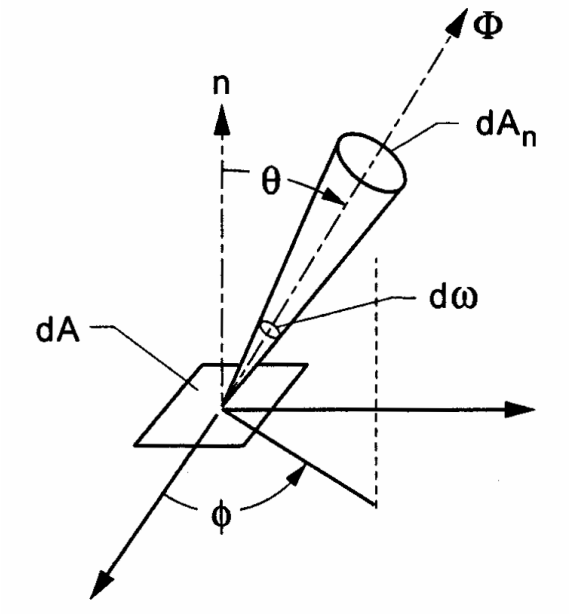

Figura 2.1: Radiação emitida por uma superfície.

A radiância é útil, pois indica quanto da energia emitida ou refletida por uma superfície será recebida por um sistema óptico observando-a de um determinado ângulo. Neste caso, o ângulo sólido de interesse é aquele que se estende abaixo do obturador do sistema óptico. Pelo fato do olho humano ser um sistema óptico, a radiância é um bom indicador de quão brilhante um objeto aparentará.

Exitância e irradiância

A exitância define a potência eletromagnética emitida por uma superfície em todas as direções. Ela é representada pela letra M e sua unidade é o $\mathrm{W{\cdot}m^{-2}}$.

Irradiância é o termo usado para caracterizar a potência eletromagnética incidente em uma superfície vinda de todas as direções. Ela é denotada pela letra E e sua unidade é o $\mathrm{W{\cdot}m^{-2}}$.

2.2.3 Radiação de corpo negro

Corpo negro é um objeto idealizado que tem as seguintes propriedades [6]:

- Um corpo negro absorve toda radiação nele incidente, não importando o comprimento de onda nem a direção de origem.

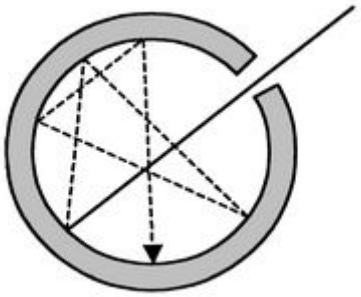

Figura 2.2: Cavidade que simula um corpo negro.

- Dada uma temperatura e um comprimento de onda, nenhuma superfície emite mais radiação térmica que um corpo negro.
- Embora a radiação emitida por um corpo negro seja função do comprimento de onda e da temperatura, ela é independente da direção. Ou seja, um corpo negro é um emissor isotropicamente difuso[5].

A Figura 2.2 materializa um corpo negro ideal. Trata-se de uma cavidade com um pequeno orifício pelo qual entra radiação e, devido a múltiplas reflexões, ela tem baixa probabilidade de sair e tende a ser totalmente absorvida. A energia emitida por um corpo negro depende, portanto, somente da temperatura de suas paredes.

Lei de Planck

A Lei de Planck descreve a distribuição de energia eletromagnética emitida por corpo negro em função de sua temperatura e do comprimento de onda. É uma das mais importantes leis que regem emissão térmica. Ela é dada por

$$L_{\lambda,b}(\lambda, T) = \frac{2hc^2}{\lambda^5}\left(e^{\frac{hc}{\lambda kT}} - 1\right)^{-1}, \tag{2.3}$$

onde $L_{\lambda,b}$ é a radiância espectral (o índice b denota corpo negro – *blackbody*), λ é o comprimento de onda em m, T é a temperatura absoluta em K, h é a constante de Planck que descreve o tamanho dos *quanta* de energia[6] e vale $6,62606896 \times 10^{-34}\,\mathrm{J}\cdot\mathrm{s}$,

[5]Também chamada de reflexão Lambertiana, trata-se de superfícies nas quais a luz incidente é espalhada de tal maneira que seu brilho aparente é o mesmo independentemente do ângulo do observador. Em outras palavras, a luminância da superfície é a mesma independentemente do ângulo de visualização.

[6]Adicionalmente a formulação da lei de distribuição de energia e que a energia do corpo negro não era emitida e absorvida de forma contínua, mas sim discreta na forma de quanta de energia,

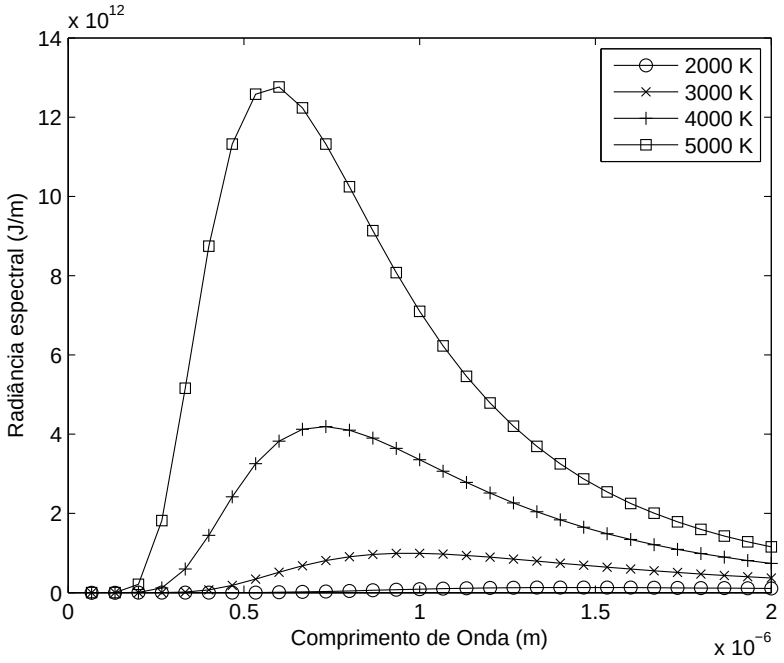

Figura 2.3: Radiância espectral do corpo negro, lei de Planck.

c é a velocidade da luz igual a $299.792.458\,\mathrm{m}\cdot\mathrm{s}^{-1}$ e k é a constante de Boltzmann que relaciona temperatura e energia e vale $1,3806504\times10^{-23}\,\mathrm{J}\cdot\mathrm{K}^{-1}$.

A Figura 2.3 representa a lei de Planck. Para uma dada temperatura, a magnitude da radiação emitida varia com o comprimento de onda e a localização de seu máximo depende da temperatura. Em outras palavras, todo objeto a uma temperatura acima do zero absoluto ($0\,\mathrm{K}$) emite radiação eletromagnética [10].

2.2.4 Superfícies reais

Emissividade

No caso de uma superfície real, deve ser aplicada uma correção na emissão do corpo negro. Tal fator de correção é conhecida como emissividade, representada pela letra ε. De fato, a emissividade é uma propriedade da superfície que mede a capacidade da mesma emitir energia. A emissividade é expressa através da razão entre a radiação emitida pela superfície em questão e a radiação emitida pelo corpo negro nas mesmas condições de temperatura, direção e banda de espectro de interesse.

Max Planck (1858-1947) descobriu que o tamanho de um quantum de energia é proporcional à sua frequência de irradiação, ou seja, $E = hf$.

Tabela 2.1: Emissividade de alguns materiais.

Material	Emissividade ε
Papel	0,93
Óxido de alumínio	0,90
Níquel polido	0,05
Alumínio anodizado	0,03
Aço inox	0,22

Portanto, a emissividade varia entre 0 e 1. A Tabela 2.1 mostra a emissividade de alguns materiais [11].

Se dentro de uma faixa de comprimento de onda a emissividade ε é < 1 porém independente do comprimento de onda, tal superfície é dita emissor cinza (*gray emitter*). Se ε depende muito do comprimento de onda, a superfície é dita emissor seletivo, ou seja $\varepsilon(\lambda)$.

Pode-se ainda definir uma grandeza chamada de emissividade espectral direcional, $\varepsilon(\lambda;\theta,\phi;T)$, como a razão entre a radiância de uma superfície em uma determinada direção, comprimento de onda e temperatura, e a radiância de corpo negro no mesmo comprimento de onda e temperatura, ou seja,

$$\varepsilon(\lambda;\theta,\phi;T) = \frac{L_\lambda(\lambda;\theta,\phi;T)}{L_{\lambda,b}(\lambda,T)}. \tag{2.4}$$

Em casos práticos, pode-se aproximar a emissividade para $\varepsilon(\lambda;\theta)$ [6].

Absorção, reflexão e transmissão

Em termos de radiação incidente, superfícies reais não se comportam como corpo negro. Elas absorvem somente parte da energia nelas incidente, uma parcela é refletida e outra é transmitida. Por isso, certas grandezas precisam ser definidas para caracterizar estas superfícies.

A razão entre o fluxo eletromagnético incidente em uma superfície e o absorvido é chamada de absorbância ou coeficiente de absorção. Ele é dada por

$$\alpha = \frac{\mathrm{d}\Phi_a}{\mathrm{d}\Phi_i}, \tag{2.5}$$

onde Φ_a é o fluxo absorvido e Φ_i é o fluxo incidente.

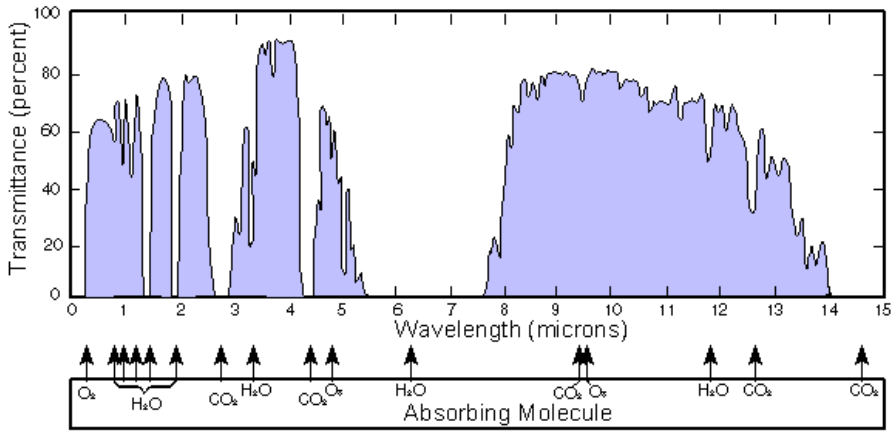

Figura 2.4: Gráfico da transmitância atmosférica para uma parte do espectro infravermelho, com indicações da absorção de algumas moléculas.

A razão entre o fluxo eletromagnético incidente em uma superfície e o refletido é chamada de reflectância ou coeficiente de reflexão. Ele é dada por

$$\rho = \frac{\mathrm{d}\Phi_r}{\mathrm{d}\Phi_i}, \tag{2.6}$$

onde Φ_r é o fluxo refletido e Φ_i é o fluxo incidente.

A razão entre o fluxo eletromagnético incidente em uma superfície e o transmitido é chamada de transmitância ou coeficiente de transmissão. Ele é dada por

$$\tau = \frac{\mathrm{d}\Phi_t}{\mathrm{d}\Phi_i}, \tag{2.7}$$

onde Φ_t é o fluxo transmitido e Φ_i é o fluxo incidente.

A lei de Kirchhoff relaciona o fluxo eletromagnético refletido, absorvido e transmitido em uma superfície através dos respectivos coeficientes por

$$\rho + \alpha + \tau = 1. \tag{2.8}$$

Transmitância atmosférica

A Figura 2.4 mostra a transmitância atmosférica típica para parte do espectro infravermelho. A absorção se dá principalmente pela água e pelo CO_2. Porém, existem duas janelas onde a atmosfera é essencialmente transparente. A janela de faixa média se localiza entre 3 e 5μm, e a de faixa longa se localiza acima de 8μm. A maioria dos sensores e câmeras infravermelhas trabalham em uma dessas duas janelas [11].

2.3 Sensores infravermelhos

Sensores infravermelhos são dispositivos que convertem radiação infravermelha em um sinal mensurável, geralmente de natureza elétrica [5]. Eles são normalmente baseados na detecção passiva de radiação eletromagnética emitida termicamente como descrito pela lei de Planck (ver Seção 2.2.3). Eles podem ser divididos em detectores térmicos e detectores fotônicos [10].

Os detectores térmicos absorvem radiação infravermelha e sofrem um pequeno acréscimo de temperatura que é então medida. A absorção de radiação infravermelha nestes dispositivos eleva sua temperatura e algum parâmetro físico a esta relacionado é modificado, como por exemplo condutividade elétrica. Exemplos destes sensores podem ser:

- Bolômetros - por variação de condutividade elétrica.
- Termopares - a junção de dois metais diferentes gera uma tensão elétrica que é dependente da temperatura.
- Detectores pneumáticos - variação de pressão em um gás causada pela variação de sua temperatura.
- Detectores piroelétricos - propriedade de certos cristais de produzir uma polarização elétrica devido a mudança de temperatura.
- Cristais líquidos - cristais líquidos que, sob efeito de temperatura, alteram sua orientação e refletem luz do vermelho ao violeta quando iluminados com luz branca.

Os detectores fotônicos ou quânticos são geralmente dispositivos semicondutores nos quais a absorção de um fóton infravermelho excita elétrons de um estado inerte a um estado de mobilidade produzindo, portanto, uma mudança de condutividade ou tensão elétrica. O aquecimento do material sensitivo é dispensável.

2.3.1 Infrared focal plane arrays - IRFPA

Uma FPA, ou *focal plane array*, é um sensor de imagem que consiste de uma matriz de sensores ópticos localizada no plano focal de um sistema de lentes, daí seu

nome. Tecnicamente, o termo FPA se refere a uma gama de dispositivos sensores de imagem, porém o uso deste termo é mais comum como referência a dispositivos bidimensionais de detectores fotônicos sensíveis à radiação na região do infravermelho. Ela recebe, então, o nome de IRFPA, ou *infrared focal plane array*. Este trabalho é focado em câmeras de vídeo infravermelho que utilizam a tecnologia IRFPA.

As IRFPA's têm se tornado o mais proeminente detector usado em sistemas de imagens infravermelhas nos últimos anos. O vasto uso desta tecnologia é atribuído aos avanços na tecnologia de sensores de estado sólido, que permitiram compacidade, baixo custo e alto desempenho. Algumas aplicações incluem visão noturna, reconhecimento e vigilância aeroespacial, imagens térmicas astronômicas e aplicações militares [2].

Sabe-se que um problema comum a todos os sensores IRFPA é o ruído de padrão fixo (*fixed-pattern noise* ou FPN), também chamado de não uniformidade espacial. De fato, o FPN continua sendo um sério problema apesar dos avanços recentes nessa tecnologia. A origem deste ruído é atribuída ao fato de cada detector da matriz, ou seja, cada pixel possuir uma variação no processo de fabricação. Em outras palavras, cada pixel do detector responde de maneira diferente à mesma quantidade de radiação incidente. O ruído FPN se manifesta aleatoriamente no espaço e está presente em todos os quadros de um vídeo infravermelho independentemente da cena ou movimento [2].

Outra característica do ruído FPN é sua lenta variação temporal durante o funcionamento do sensor. Porém, ele pode se tornar significativo numa ordem de grandeza de 30 segundos após uma calibração [3]. Esse desvio temporal é atribuído a variações na temperatura do sensor, material de fabricação, ruído eletrônico de leitura, controle automático de ganho, entre outros. Portanto, uma única calibração é ineficaz e o problema requer estimativa e compensação contínuas durante a operação da câmera [2].

Embora a verdadeira resposta das IRFPA's seja não-linear, ela é em geral modelada linearmente como função da radiância, um ganho e um desvio (*bias*) [2, 4] como

$$y_k(i,j) = a(i,j)L_k(i,j) + b(i,j), \tag{2.9}$$

onde $y_k(i,j)$ é a saída da câmera referente ao pixel (i,j) no instante k, $a(i,j)$ é o

ganho associado ao pixel (i, j), $L_k(i, j)$ é a radiância incidente no elemento sensor (i, j) no instante k e $b(i, j)$ é desvio ou *bias* associado ao pixel (i, j).

Desta maneira, é possível corrigir o FPN através de técnicas que estimem estes parâmetros e apliquem-nos aos modelos de modo a corrigi-lo. Tais correções são usualmente chamadas de NUC (*non uniformity correction*). O fato do FPN apresentar uma lenta variação faz com que os parâmetros sejam usualmente modelados como invariantes no tempo. Todavia, a variação existe e as correções devem ser aplicadas continuamente.

Em alguns casos o FPN devido ao parâmetro ganho tende a ser menos significativo e bons resultados são obtidos apenas corrigindo-se o desvio. Este tipo de modelo simplifica bastante os algoritmos de correção e também gera bons resultados [12].

2.4 Conclusões

O objetivo deste capítulo foi dar uma breve introdução ao domínio das câmeras infravermelhas. Inicialmente foram apresentados conceitos fundamentais sobre temperatura, radiação e suas relações. Os assunto foram abordados gradualmente até que finalmente foi apresentada a tecnologia de sensores infravermelhos utilizados nas principais câmeras modernas. Pode-se considerar o carácter do presente capítulo como informativo, pois contém informações não essenciais, porém motivadoras para o que será desenvolvido nos capítulos que seguem.

Capítulo 3

Estado da arte

Este capítulo faz uma revisão dos principais métodos de correção de não uniformidade (NUC) ou correção de ruído de padrão fixo (FPN) disponíveis na literatura.

3.1 Métodos existentes correção de FPN

Vários métodos já foram propostos para a correção do FPN. A seguir são apresentados os principais. Qualquer que seja o método de correção, ele deve ser aplicado repetidamente já que o FPN apresenta uma deriva (*drift*) temporal.

3.1.1 Baseados em calibração (*Calibration-based*)

Trata-se de uma filosofia de calibração onde todos os pixels da IRFPA são expostos a uma temperatura igual de referência, como uma fonte de radiação do tipo corpo negro ou um obturador de temperatura uniforme, de modo a estimar o FPN. Se todos os pixels da IRFPA estão recebendo a mesma quantidade de radiação, deveriam fornecer a mesma saída. Porém, isso não ocorre e o FPN é calculado a partir da diferença das respostas entre os pixels.

O algoritmo mais conhecido de NUC é o de calibração por dois pontos (*two-point calibration*). A ideia é definir um ganho e um desvio (*bias*) para cada pixel de modo a se obter a mesma resposta para toda a matriz quando exposta a mesma quantidade de radiação. A calibração é realizada expondo a IRFPA a dois corpos negros com temperaturas diferentes e conhecidas. Os parâmetros ganho e desvio são obtidos resolvendo-se um conjunto de equações lineares. Após a calibração, a

correção da não uniformidade é obtida subtraindo o desvio relativo e dividindo pelo ganho respectivo a saída de cada pixel da IRFPA, como mostra a equação (3.1).

$$\hat{x}_k(i,j) = \frac{y_k(i,j) - \hat{b}(i,j)}{\hat{a}(i,j)}. \tag{3.1}$$

A principal vantagem desse procedimento é sua grande exatidão. Porém, as desvantagens que ele apresenta são:

- Necessidade de recalibrações regulares devido à deriva do FPN.
- Aumento do peso e o tamanho do dispositivo devido à inclusão de corpos negros ou obturadores.
- A visão é interrompida durante o processo de calibração (pode levar alguns segundos).
- Possível introdução de ruído acústico durante a abertura e fechamento do obturador.
- Potência extra para alimentar o obturador.
- A operação normal do dispositivo é interrompida para a realização da calibração.

Se o modelo adotado para o FPN for de ordem maior de modo a melhor representá-lo (por exemplo, segundo grau), mais temperaturas de referências serão necessárias para se estimar os parâmetros do modelo.

3.1.2 Baseados na cena (*Scene-based*)

É uma família de algoritmos que se baseiam na informação contida na própria cena para estimar e corrigir o FPN. Sua grande vantagem é que não há interrupção no vídeo. Eles se dividem em dois tipos: métodos estatísticos (*statistical*) e *registration-based*[1].

Os métodos estatísticos assumem que todos os pixels são expostos a quantidades de radiação com a mesma distribuição de probabilidade. Esta suposição só é válida se

[1] O autor preferiu manter o termo em inglês por não achar uma tradução adequada ao português.

houver variação significativa na cena, por exemplo elevada movimentação da câmera. A correção é obtida ajustando-se os parâmetros de ganho e desvio de cada pixel de modo que suas estatísticas sejam iguais. Se a assunção estatística for violada, o desempenho destes algoritmos tende a ser baixo. Exemplos podem ser encontrados em [13, 14, 15, 16, 17, 18].

Os algoritmos *registration-based*, por outro lado, necessitam de uma estimativa precisa dos deslocamentos entre quadros consecutivos. Uma vez que esses deslocamentos foram obtidos e os quadros foram alinhados, uma estimativa do FPN é possível comparando-se a resposta de diferentes pixels expostos a um mesmo ponto na cena, ou seja, a mesma quantidade de radiação. A vantagem destes métodos é que não há assunção estatística e a estimativa do FPN e correção são feitas a cada novo quadro. Por outro lado, se a estimativa de deslocamentos contiver erros, não é possível obter uma boa estimativa do FPN. Além disso, os algoritmos tendem a ser mais elaborados e computacionalmente mais custosos. Exemplos destas técnicas podem ser encontrados em [2, 19, 20, 21]. Os algoritmos apresentados neste trabalho são do tipo *registration-based*.

Capítulo 4

Correção do desvio

4.1 Introdução

Neste capítulo é apresentado o desenvolvimento de um algoritmo para a correção do desvio em vídeos na região do infravermelho. É adotado um modelo simplificado do ruído de padrão fixo onde existe apenas uma parcela aditiva de ruído diferente para cada pixel. Após apresentar o modelo de observação, é aplicado o gradiente descendente para estimar de forma gradual o valor do desvio a cada novo quadro. É importante lembrar que, pelo fato do desvio ter uma variação temporal lenta, ele é modelado como invariante no tempo.

Faz-se, então, uma análise da matriz que implementa o movimento global entre dois quadros consecutivos. Em seguida, é apresentado o estimador de movimento global utilizado e são feitas discussões sobre sua aplicação em vídeos contendo FPN. Finalmente são apresentados os resultados de simulações em vídeos sintéticos e ensaios em vídeos de cenas reais obtidos com câmeras térmicas.

4.2 Formulação do problema

Seja o modelo simplificado, considerando apenas o desvio, de observação de um sistema de aquisição de imagens infravermelhas com FPN

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{b}, \tag{4.1}$$

onde $\mathbf{y}_k$ é um vetor que representa a imagem observada no instante k, $\mathbf{x}_k$ é um vetor que representa a imagem real ou radiância incidente no sensor no instante k e $\mathbf{b}$

representa o desvio nos dados observados ou *bias*. Todos os vetores que representam imagens estão, neste trabalho, em ordem lexicográfica[1]. Pelo fato do desvio possuir uma variação muito lenta, ele é modelado como invariante no tempo [2].

Como o desvio não é conhecido, pode-se obter uma estimativa da imagem real através de uma estimativa do desvio através de

$$\hat{\mathbf{x}}_k = \mathbf{y}_k - \hat{\mathbf{b}}. \tag{4.2}$$

Considerando-se um movimento relativo entre dois quadros consecutivos de uma sequência de imagens ou vídeo, é possível escrever

$$\mathbf{x}_k = \mathbf{M}_k \mathbf{x}_{k-1} + \boldsymbol{\gamma}_k, \tag{4.3}$$

onde $\mathbf{M}_k$ é a matriz que implementa o movimento ou deslocamento entre os quadros $(k-1)$ e k, e $\boldsymbol{\gamma}_k$ é o vetor que modela as inovações no quadro seguinte que não podem ser obtidas com simples deslocamento.

Combinando as equações (4.2) e (4.3), desconsiderando o vetor de inovações γ_k já que a informação nele contida não pode ser comparada entre dois quadros, e considerando o deslocamento entre dois quadros , é possível escrever uma estimativa para a imagem real do quadro seguinte como

$$\hat{\mathbf{x}}_k = \mathbf{M}_k \left(\mathbf{y}_{k-1} - \hat{\mathbf{b}} \right). \tag{4.4}$$

Pode-se ainda combinar as equações (4.1) e (4.4) para se obter uma estimativa da próxima imagem observada através de

$$\hat{\mathbf{y}}_k = \mathbf{M}_k \left(\mathbf{y}_{k-1} - \hat{\mathbf{b}} \right) + \hat{\mathbf{b}}. \tag{4.5}$$

Portanto, define-se o erro entre a imagem observada e sua estimativa com base somente na estimativa de movimento e do desvio através de

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k \tag{4.6}$$

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \mathbf{M}_k \left(\mathbf{y}_{k-1} - \hat{\mathbf{b}} \right) - \hat{\mathbf{b}}, \tag{4.7}$$

onde $\boldsymbol{\epsilon}_k$ é o vetor erro de estimativa no instante k. O erro quadrático médio é definido como

$$\varepsilon_k = \frac{1}{N} \sum_{i=1}^{N} \left[\epsilon_k(i) \right]^2 = \frac{\boldsymbol{\epsilon}_k^{\mathrm{T}} \boldsymbol{\epsilon}_k}{N}. \tag{4.8}$$

[1]Entende-se por ordem lexicográfica quando as colunas de uma imagem são empilhadas umas sobre as outras, da primeira à última, para formar um vetor.

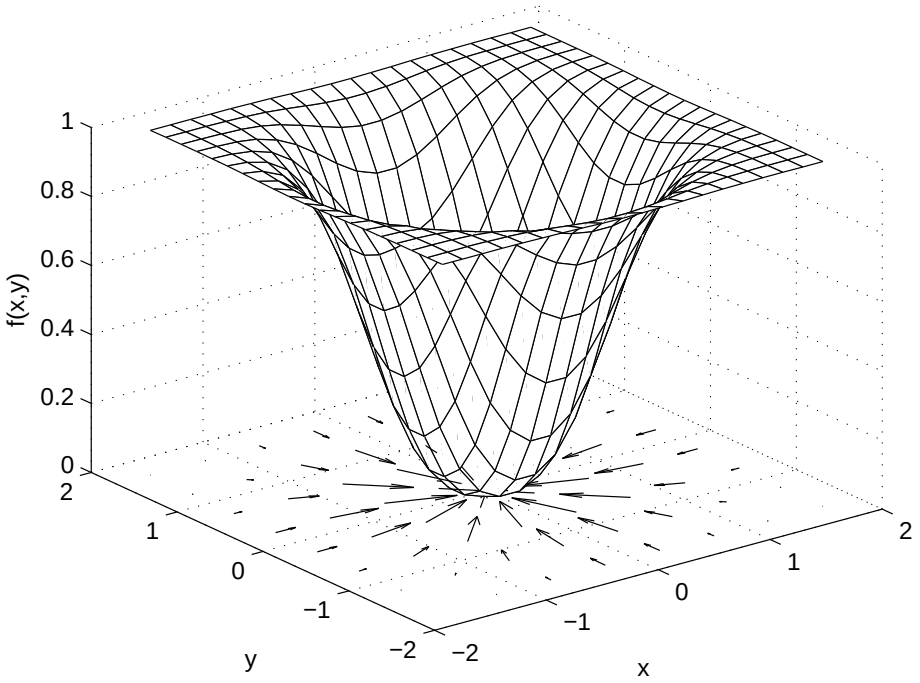

Figura 4.1: Exemplo de função objetivo e projeção gradiente negativo.

4.3 Gradiente descendente

Para obter uma estimativa do desvio, pode-se utilizar o gradiente descendente da seguinte forma [22]

$$\hat{\mathbf{b}}_{k+1} = \hat{\mathbf{b}}_k - \mu \nabla_{\mathbf{b}} \varepsilon_k, \tag{4.9}$$

onde μ representa o passo de atualização, $\nabla_{\mathbf{b}}$ representa o operador gradiente em relação a $\mathbf{b}$ e ε_k é o erro quadrático médio. Neste caso, o índice k aplicado a $\mathbf{b}$ não indica que $\mathbf{b}$ está variando no tempo. A assunção inicial que $\mathbf{b}$ é invariante continua válida e o índice k indica que a estimativa está sendo atualizada a cada novo quadro.

A ideia do gradiente descendente é que, a cada iteração, dá-se um passo na direção oposta ao crescimento da função que se pretende minimizar. A Figura 4.1 ilustra uma função objetivo e a projeção do gradiente negativo. Percebe-se que os vetores do gradiente negativo apontam para uma região próxima ao mínimo da função. É possível atingir o mínimo dando-se passos de tamanho adequado. Passos muito longos podem tornar o algoritmo instável.

Aplicando o gradiente ao erro quadrático médio, tem-se que

$$\nabla_{\mathbf{b}}\varepsilon = \frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{b}}\boldsymbol{\epsilon} \tag{4.10}$$

$$= \left(\mathbf{M}^{\mathrm{T}} - \mathbf{I}\right)\boldsymbol{\epsilon}. \tag{4.11}$$

A demonstração do desenvolvimento acima é a seguinte. Sejam $\mathbf{p}$ e $\mathbf{q}$ dois vetores. Então [23, 24],

$$\nabla_{\mathbf{q}} p_1 \triangleq \frac{\partial p_1}{\partial \mathbf{q}} \triangleq \begin{bmatrix} \frac{\partial p_1}{\partial q_1} \\ \frac{\partial p_1}{\partial q_2} \\ \vdots \\ \frac{\partial p_1}{\partial q_N} \end{bmatrix} \tag{4.12}$$

,

$$\nabla_{q_1} \mathbf{p} \triangleq \frac{\partial \mathbf{p}}{\partial q_1} \triangleq \begin{bmatrix} \frac{\partial p_1}{\partial q_1} & \frac{\partial p_2}{\partial q_1} & \cdots & \frac{\partial p_N}{\partial q_1} \end{bmatrix} \tag{4.13}$$

e

$$\nabla_{\mathbf{q}} \mathbf{p} \triangleq \frac{\partial \mathbf{p}}{\partial \mathbf{q}} \triangleq \begin{bmatrix} \frac{\partial p_1}{\partial q_1} & \frac{\partial p_2}{\partial q_1} & \cdots & \frac{\partial p_N}{\partial q_1} \\ \frac{\partial p_1}{\partial q_2} & \frac{\partial p_2}{\partial q_2} & \cdots & \frac{\partial p_N}{\partial q_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial p_1}{\partial q_N} & \frac{\partial p_2}{\partial q_N} & \cdots & \frac{\partial p_N}{\partial q_N} \end{bmatrix}. \tag{4.14}$$

Logo,

$$\frac{\partial \varepsilon}{\partial \mathbf{b}} = \frac{\partial}{\partial \mathbf{b}} \frac{1}{N} \sum_{i=1}^{N} \left[\epsilon(i)\right]^2 \tag{4.15}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \frac{\partial}{\partial \mathbf{b}} \left[\epsilon(i)\right]^2 \tag{4.16}$$

$$= \frac{2}{N} \sum_{i=1}^{N} \frac{\partial \epsilon(i)}{\partial \mathbf{b}} \epsilon(i) \tag{4.17}$$

$$= \frac{2}{N} \sum_{i=1}^{N} \begin{bmatrix} \frac{\partial \epsilon(i)}{\partial b_1} \\ \frac{\partial \epsilon(i)}{\partial b_2} \\ \vdots \\ \frac{\partial \epsilon(i)}{\partial b_N} \end{bmatrix} \epsilon(i) \tag{4.18}$$

$$= \frac{2}{N} \begin{bmatrix} \frac{\partial \epsilon_1}{\partial b_1} & \frac{\partial \epsilon_2}{\partial b_1} & \cdots & \frac{\partial \epsilon_N}{\partial b_1} \\ \frac{\partial \epsilon_1}{\partial b_2} & \frac{\partial \epsilon_2}{\partial b_2} & \cdots & \frac{\partial \epsilon_N}{\partial b_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \epsilon_1}{\partial b_N} & \frac{\partial \epsilon_2}{\partial b_N} & \cdots & \frac{\partial \epsilon_N}{\partial b_N} \end{bmatrix} \cdot \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_N \end{bmatrix} \tag{4.19}$$

$$= \frac{2}{N} \frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{b}} \boldsymbol{\epsilon}, \tag{4.20}$$

sendo que a parcela $\frac{2}{N}$ desaparece, pois irá compor o passo de atualização.

Derivando-se o vetor erro $\boldsymbol{\epsilon}$ em relação ao vetor dos desvios $\mathbf{b}$ obtém-se

$$\frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{b}} = \frac{\partial}{\partial \mathbf{b}} \left(\mathbf{M}\mathbf{b} - \mathbf{b} \right) \tag{4.21}$$

$$= \left(\mathbf{M}^{\mathrm{T}} - \mathbf{I} \right). \tag{4.22}$$

O transposto na matriz $\mathbf{M}$ aparece, pois são os elementos da primeira coluna de $\mathbf{M}$ que multiplicam o primeiro elemento de $\mathbf{b}$.

A equação final de atualização pode ser finalmente escrita como

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \mathbf{M}_k \left(\mathbf{y}_{k-1} - \hat{\mathbf{b}}_k \right) - \hat{\mathbf{b}}_k \tag{4.23}$$

$$\hat{\mathbf{b}}_{k+1} = \hat{\mathbf{b}}_k - \mu \left(\mathbf{M}_k^{\mathrm{T}} - \mathbf{I} \right) \boldsymbol{\epsilon}_k, \tag{4.24}$$

onde μ define o passo de atualização da estimativa $\hat{\mathbf{b}}$.

4.4 Matriz de movimento global

Trata-se de uma matriz que implementa translação em uma imagem representada lexicograficamente. Quando uma translação de imagem é realizada, os pixels da borda na direção do movimento serão descartados, pois suas novas posições estarão fora da imagem quando forem deslocados. Já nas bordas simetricamente opostas, os pixels são provenientes de fora da imagem, ou seja, não são conhecidos. Para preencher esses espaços é feito um espelhamento ou preenchimento simétrico em relação às bordas. Esse procedimento ficará mais claro nos parágrafos a seguir.

Como simplificação, foi admitido que entre os quadros há apenas translação global, ou seja, movimentos da câmera na horizontal e vertical. É verdade que em vídeos reais existem movimentos mais complexos que estes como rotação, zoom, movimentos dos objetos na cena, entre outros. Porém, como o FPN é estimado em

vários quadros e não em só um par de quadro erros devidos a movimentos mais complexos tendem a se diluir.

Para facilitar o entendimento, pode-se dividi-la em uma parte inteira $\mathbf{M}_\Delta$, que produz translações de número inteiro de pixels com $\{\Delta \in \mathbb{Z}\}$, e uma parte fracionária $\mathbf{M}_\delta$ que produz movimentos menores que um pixel, ou seja, movimentos de subpixel com $\{\delta \in \mathbb{R} | 0 \leq \delta < 1\}$. Tal divisão é válida, pois a matriz de movimentos inteiros $\mathbf{M}_\Delta$ só contém zeros e uns. É a localização dos elementos iguais a um que é responsável pela translação.

O movimento global é descrito, então, por

$$\mathbf{M} = \mathbf{M}_\Delta \mathbf{M}_\delta. \tag{4.25}$$

Sejam uma imagem $\mathbf{X}_k$ de 3×3 pixels e a sua representação lexicográfica $\mathbf{x}_k$. Tem-se no instante $k = 0$

$$\mathbf{X}_0 = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} \quad \text{e} \quad \mathbf{x}_0 = \left[\begin{array}{c} x_{11} \\ x_{21} \\ x_{31} \\ \hline x_{12} \\ x_{22} \\ x_{32} \\ \hline x_{13} \\ x_{23} \\ x_{33} \end{array}\right]. \tag{4.26}$$

O vetor de deslocamento indica qual a direção e o sentido que os elementos devem se deslocar em relação às suas posições na matriz[2]. Supondo um deslocamento global

[2]Para notação de movimento global, adota-se o movimento do elemento $x_k(i, j)$ da matriz em relação às linhas e às colunas, ou seja, $x_{k+1}(i, j) = x_k(i - \Delta_1, j - \Delta_2)$.

inteiro de $\boldsymbol{\Delta} = [-1 \ -1]^{\mathrm{T}}$ tem-se no instante $k = 1$

$$\mathbf{X}_1 = \begin{bmatrix} x_{22} & x_{23} & x_{22} \\ x_{32} & x_{33} & x_{32} \\ x_{22} & x_{23} & x_{22} \end{bmatrix} \quad \text{e} \quad \mathbf{x}_1 = \left[\begin{array}{c} x_{22} \\ x_{32} \\ x_{22} \\ \hline x_{23} \\ x_{33} \\ x_{23} \\ \hline x_{22} \\ x_{32} \\ x_{22} \end{array}\right]. \tag{4.27}$$

A matriz que produz esse movimento global é dada por

$$\mathbf{M}_{\Delta} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.28}$$

Os deslocamento fracionários seguem a mesma lógica dos deslocamentos inteiros. Em outras palavras, o vetor de deslocamento fracionário indica qual o sentido e direção que os elementos da matriz devem se deslocar. O cálculo de cada elemento da matriz de imagem após o deslocamento é feito através de média ponderada onde os pesos são dados pelo vetor de deslocamento nos sentidos horizontal e vertical.

Supondo um movimento global de $\boldsymbol{\delta} = [-0,1 \ -0,3]^{\mathrm{T}}$ do instante $k = 1$ para $k = 2$. A Figura 4.2 mostra como é calculado o novo primeiro elemento da matriz a partir dos pesos obtidos do vetor de deslocamento.

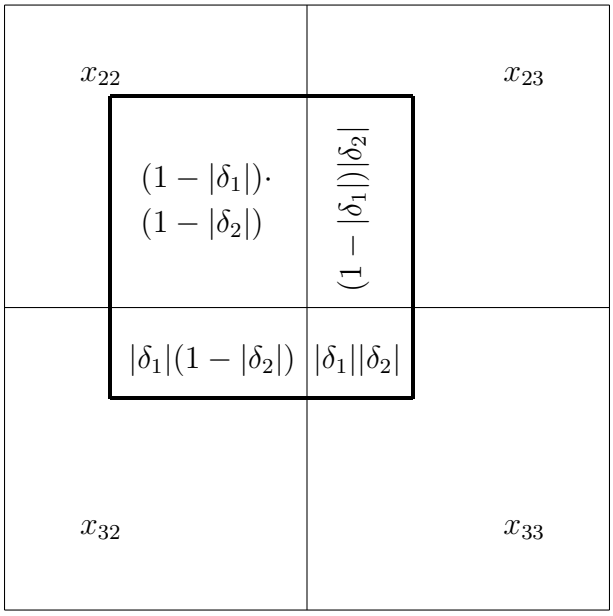

Figura 4.2: Obtenção do primeiro elemento da matriz imagem após deslocamento fracionário.

A imagem no instante $k = 2$ é representada lexicograficamente por

$$\mathbf{x}_2 = \begin{bmatrix} 0,63x_{22} + 0,07x_{32} + 0,27x_{23} + 0,03x_{33} \\ 0,63x_{32} + 0,07x_{22} + 0,27x_{33} + 0,03x_{23} \\ 0,63x_{22} + 0,07x_{32} + 0,27x_{23} + 0,03x_{33} \\ \hline 0,63x_{23} + 0,07x_{33} + 0,27x_{22} + 0,03x_{32} \\ 0,63x_{33} + 0,07x_{23} + 0,27x_{32} + 0,03x_{22} \\ 0,63x_{23} + 0,07x_{33} + 0,27x_{22} + 0,03x_{32} \\ \hline 0,63x_{22} + 0,07x_{32} + 0,27x_{23} + 0,03x_{33} \\ 0,63x_{32} + 0,07x_{22} + 0,27x_{33} + 0,03x_{23} \\ 0,63x_{22} + 0,07x_{32} + 0,27x_{23} + 0,03x_{33} \end{bmatrix} \tag{4.29}$$

e a matriz que produz esse efeito é dada por

$$\mathbf{M}_\delta = \begin{bmatrix} 0.63 & 0.07 & 0 & 0.27 & 0.03 & 0 & 0 & 0 & 0 \\ 0 & 0.63 & 0.07 & 0 & 0.27 & 0.03 & 0 & 0 & 0 \\ 0 & 0.07 & 0.63 & 0 & 0.03 & 0.27 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.63 & 0.07 & 0 & 0.27 & 0.03 & 0 \\ 0 & 0 & 0 & 0 & 0.63 & 0.07 & 0 & 0.27 & 0.03 \\ 0 & 0 & 0 & 0 & 0.07 & 0.63 & 0 & 0.03 & 0.27 \\ 0 & 0 & 0 & 0.27 & 0.03 & 0 & 0.63 & 0.07 & 0 \\ 0 & 0 & 0 & 0 & 0.27 & 0.03 & 0 & 0.63 & 0.07 \\ 0 & 0 & 0 & 0 & 0.03 & 0.27 & 0 & 0.07 & 0.63 \end{bmatrix}. \tag{4.30}$$

O movimento global de $k = 0$ a $k = 2$, totalizando $\mathbf{d} = [-1, 1 \ \ -1, 3]^{\mathrm{T}}$ é dado por $\mathbf{M} = \mathbf{M}_\Delta \mathbf{M}_\delta$, ou seja,

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0.63 & 0.07 & 0 & 0.27 & 0.03 \\ 0 & 0 & 0 & 0 & 0.07 & 0.63 & 0 & 0.03 & 0.27 \\ 0 & 0 & 0 & 0 & 0.63 & 0.07 & 0 & 0.27 & 0.03 \\ 0 & 0 & 0 & 0 & 0.27 & 0.03 & 0 & 0.63 & 0.07 \\ 0 & 0 & 0 & 0 & 0.03 & 0.27 & 0 & 0.07 & 0.63 \\ 0 & 0 & 0 & 0 & 0.27 & 0.03 & 0 & 0.63 & 0.07 \\ 0 & 0 & 0 & 0 & 0.63 & 0.07 & 0 & 0.27 & 0.03 \\ 0 & 0 & 0 & 0 & 0.07 & 0.63 & 0 & 0.03 & 0.27 \\ 0 & 0 & 0 & 0 & 0.63 & 0.07 & 0 & 0.27 & 0.03 \end{bmatrix}. \tag{4.31}$$

4.5 Estimativa de movimento global

Foi utilizado o algoritmo de estimativa de movimento global LIPSE descrito em [2]. Segue uma descrição do método com pequenas adaptações na notação.

A sigla LIPSE significa *linear interpolation projection-based shift estimator*, ou seja, estimador de deslocamento baseado na interpolação linear de projeções (das linhas e das colunas da imagem).

As vantagens de algoritmos baseados em projeção são rapidez computacional e robustez ao ruído temporal e especialmente FPN, que pode vir a ser um importante

impedimento para se alcançar uma estimativa de deslocamento confiável [25]. Para mais informações, consultar [2, 25].

Seja um par de quadros consecutivos $\mathbf{y}_{k-1}$ e $\mathbf{y}_k$ de um vídeo que apresentem entre si apenas deslocamento translacionais de imagem. Se cada pixel da imagem for representado por $y_T(i,j)$, as projeções de linhas e colunas são definidas respectivamente por

$$y_L(j) = \tfrac{1}{N}\sum_{i=1}^{N} y_T(i,j) \quad \text{e} \quad y_C(i) = \tfrac{1}{M}\sum_{j=1}^{M} y_T(i,j). \tag{4.32}$$

A solução será desenvolvida apenas para a projeção das colunas, porém se aplica de forma análoga à projeção das linhas. Daqui em diante o vetor projeção das colunas $y_C(i)$ será chamado apenas de $y(i)$.

Supondo-se que entre dois quadros o deslocamento inteiro não esteja presente, restando apenas movimento de subpixel positivo, ou seja, menor que um pixel. O deslocamento inteiro será elucidado em seguida. Cada elemento da k-ésima projeção pode, então, ser estimado através de

$$\hat{y}_k(i) = (1-\delta_k)y_{k-1}(i) + \delta_k y_{k-1}(i+1), \tag{4.33}$$

onde $0 \leq \delta_k < 1$ é o valor do deslocamento em subpixel.

O erro quadrático médio de estimativa pode ser definido como

$$\varphi_k = \sum_{i=1}^{M-1} \left[y_k(i) - \hat{y}_k(i)\right]^2 \tag{4.34}$$

$$= \sum_{j=1}^{M-1} \left[y_k^2(i) - 2y_k(i)\hat{y}_k(i) + \hat{y}_k^2(i)\right]. \tag{4.35}$$

Substituindo a equação (4.33) em (4.35) e expandindo os termos obtém-se

$$\begin{aligned}\varphi_k = \sum_{i=1}^{M-1} \Big\{ & y_k^2(i) + y_{k-1}^2(i) - 2y_k(i)y_{k-1}(i) + \\ & + 2\delta_k \left[y_k(i)\left(y_{k-1}(i) - y_{k-1}(i+1)\right) + y_{k-1}(i)\left(y_{k-1}(i+1) - y_{k-1}(i)\right)\right] + \\ & + \delta_k^2 \left[y_{k-1}^2(i) - 2y_{k-1}(i)y_{k-1}(i+1) + y_{k-1}^2(i+1)\right]\Big\}. \end{aligned} \tag{4.36}$$

O valor de δ_k que minimiza o erro quadrático médio é obtido por

$$\frac{\partial \varphi_k}{\partial \delta_k} = 0, \tag{4.37}$$

o que resulta em

$$\delta_k = \frac{\psi_k}{\zeta_k}, \tag{4.38}$$

onde

$$\psi_k = \sum_{i=1}^{M-1} \Big\{ y_k(i) \left[y_{k-1}(i) - y_{k-1}(i+1) \right] + \tag{4.39}$$

$$+ y_{k-1}(i) \left[y_{k-1}(i+1) - y_{k-1}(i) \right] \Big\} \tag{4.40}$$

e

$$\zeta_k = \sum_{i=1}^{M-1} \Big\{ y_{k-1}^2(i) - 2 y_{k-1}(i) y_{k-1}(i+1) + y_{k-1}^2(i+1) \Big\}. \tag{4.41}$$

Finalmente, os passos que devem ser aplicados para o obtenção da estimativa de movimento para cada par de quadros consecutivos são:

1. Calcular δ_k pela equação (4.38) para todos deslocamentos inteiros Δ_k's possíveis entre cada par de quadros nas projeções calculadas pela equação (4.32).

2. Calcular todos os erros quadráticos médios φ_k's nas combinações descritas no item anterior.

3. Escolher o deslocamento inteiro Δ_k que gere o menor erro quadrático médio φ_k.

4. Formar o deslocamento total $d_k = \Delta_k + \delta_k$.

5. Repetir os itens anteriores para a projeção das colunas, obtendo, finalmente, o vetor de deslocamentos na horizontal e vertical $\mathbf{d}_k$.

4.6 Métrica de qualidade de imagem utilizada: SSIM

Para a métrica de qualidade de imagem será utilizada neste trabalho a medida SSIM (*Structural SIMilarity*) que tem mostrado melhores resultados para medir qualidade de imagem que os tradicionais MSE ou PSNR quando comparadas com avaliações subjetivas [1]. A ideia é que, ao se comparar duas imagens, as medidas MSE ou PSNR às vezes fornecem um valor que não corresponde diretamente ao critério

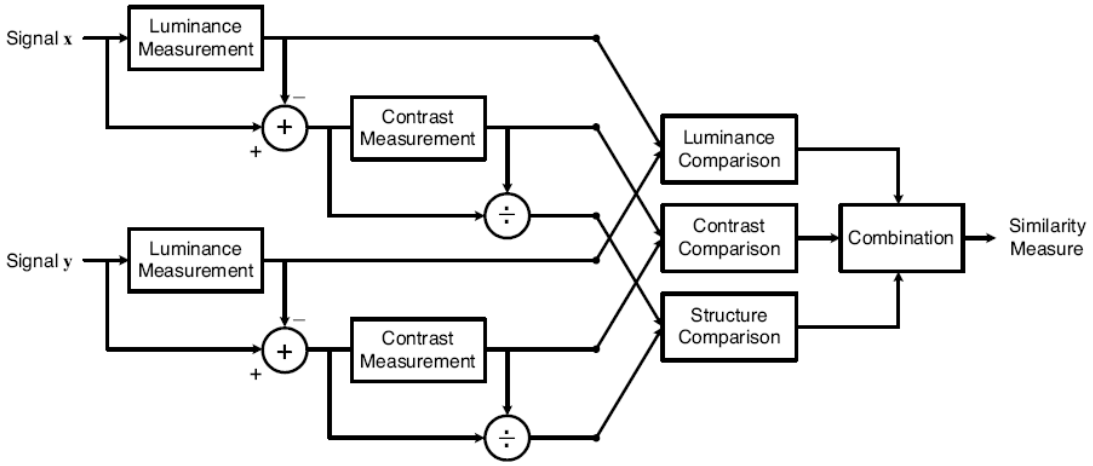

Figura 4.3: Diagrama do cálculo da medida SSIM, reproduzido de [1].

humano de classificar o quanto uma imagem se parece com outra. A medida SSIM tenta contornar esse problema.

A métrica é construída a partir da perspectiva de formação da imagem. Características como luminância e contraste são normalizadas antes da comparação das imagens, porém as diferenças desses parâmetros entre as imagens irão compor a métrica final com pesos adequados. A Figura 4.3 ilustra o processo de obtenção da métrica. As equações utilizadas nos cálculos são apresentadas no Apêndice B. Mais informações acerca da métrica SSIM podem ser encontradas em [1].

Além disso, a métrica SSIM tem as seguintes propriedades interessantes:

1. Simetria: $\mathrm{SSIM}(\mathbf{x}, \mathbf{y}) = \mathrm{SSIM}(\mathbf{y}, \mathbf{x})$;

2. Limitação: $\mathrm{SSIM}(\mathbf{x}, \mathbf{y}) \leq 1$;

3. Único máximo: $\mathrm{SSIM}(\mathbf{x}, \mathbf{y}) = 1$ se, e somente se, $\mathbf{x} = \mathbf{y}$. Ou seja, $x_i = y_i, \forall i$.

Cabe ressaltar que neste trabalho o pixel de cada imagem é representado por um valor real $\{x \in \mathbb{R} | 0 \leq x \leq 1\}$ representados no MATLAB com precisão *double* de 64 bits conforme padrão IEEE 754. Os pixels são ilustrados em escala de cinza com 0 sendo preto e 1 a cor branca.

Voltando à métrica, em suma, quanto mais próxima de 1 é a medida SSIM, mais parecidas as duas imagens comparadas podem ser consideradas. Não é definido um limite inferior para a métrica, porém considerando a faixa de valores utilizados neste trabalho pode-se obtê-lo.

Logo, o limite inferior foi calculado como $\mathrm{SSIM}(\mathbf{1}, \mathbf{0}) = 0,8667$, sendo $\mathbf{1}$ uma imagem composta somente de pixels iguais a 1 e $\mathbf{0}$ uma imagem composta de pixels iguais a zero. O limite inferior próximo de 1 se deve ao fato da medida SSIM ter sido definida para uma faixa dinâmica de $(2^8 - 1) = 255$. Esse fato só afeta a escala e não invalida a comparação entre diferentes métodos de correção de FPN.

De modo a melhor visualizar os resultados será utilizado o valor (1-SSIM). Portanto, quanto mais próximo de zero é a medida (1-SSIM), menos diferença existe entre a imagem estimada e a original e melhor é a estimativa ou o estimador.

4.7 Resultados

Os resultados apresentados estão divididos em simulações e ensaios. Nas simulações são gerados vídeos sintéticos a partir de porções de uma imagem qualquer com FPN acrescentado de maneira controlada. Nos ensaios, os algoritmos são aplicados a vídeos reais contendo FPN. De modo a facilitar a comparação com os demais algoritmos que serão apresentados neste trabalho, os resultados dos ensaios estão compilados na Seção 8.4.

Para as simulações, é acrescentado também ruído eletrônico com distribuição normal completando o modelo de observação que é dado por

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{b} + \mathbf{n}_k, \tag{4.42}$$

onde $\mathbf{n}_k$ é o vetor de ruído eletrônico. Os pixels das imagens têm uma faixa dinâmica de 0 a 1, ou seja, $\{x \in \mathbb{R} | 0 \leq x \leq 1\}$. Tanto $\mathbf{b}$ quanto $\mathbf{n}_k$ são vetores aleatórios com distribuição normal de média zero e desvio padrão $\sigma_{\mathbf{b}}$ e $\sigma_{\mathbf{n}}$ respectivamente. O valor estimado de $\sigma_{\mathbf{n}}$ a partir de vídeos na região do infravermelho reais foi de 2×10^{-4} e será usado esse valor nas simulações salvo quando especificado. Para $\sigma_{\mathbf{b}}$ será utilizado $0,1$ que, apesar de ser superior ao valor estimado, servirá para testar os algoritmos em condições mais adversas. O movimento global, que simula o movimento da câmera, também será um vetor aleatório que representa deslocamentos na horizontal e vertical com média zero e desvio padrão de 2 pixels. Esses deslocamentos são puramente translacionais.

Os vídeos foram gerados a partir de uma janela de 128×128 pixels que caminha sobre uma imagem de 256×256 pixels. Os pixels da janela podem não coincidir

Figura 4.4: Vídeo sintético antes (esquerda) e após (direita) acréscimo de ruído de padrão fixo.

com os da imagem maior, o que gera deslocamentos de subpixel. Inicialmente a janela se encontra no centro da imagem e caminha aleatoriamente sobre a mesma, limitando-se a suas bordas. O resultado final é um vídeo de 128×128 pixels que mostra diferentes visões da mesma cena.

A Figura 4.4 mostra um quadro de um vídeo sintético antes e depois o acréscimo de ruído de padrão fixo sintético com $\sigma_{\mathbf{b}} = 0,1$.

4.7.1 Estimativa de movimento conjugada à correção

É possível aplicar a correção do desvio a um par de quadros antes da estimativa de movimento global de forma a melhorar o desempenho deste estimador. Mesmo o estimador LIPSE sendo robusto à presença de não uniformidades [2], uma pré-correção do desvio utilizando os valores obtidos no passo anterior pode melhorar seu desempenho.

Primeiramente foi estudada a aplicação da estimativa de movimento e da correção do desvio de forma independente (caso 1) e de forma conjugada (caso 2). No caso 1 é feita estimativa de movimento de todo o vídeo e após é aplicada a correção de desvio. No caso 2 a estimativa de movimento é feita nas estimativas de imagem real. Os dois métodos são sintetizados a seguir.

- Caso 1: Estimativa de movimento e correção de desvio de forma independente

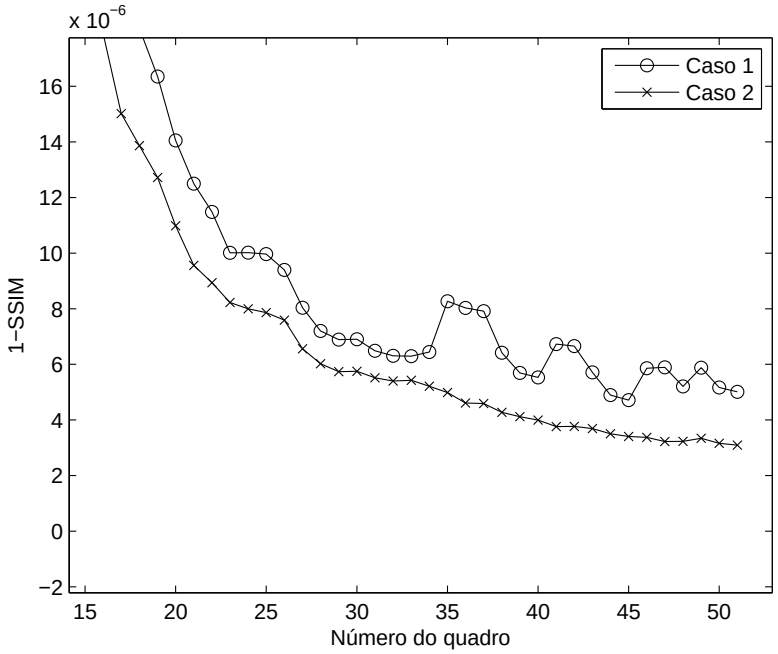

Figura 4.5: Efeito da correção do desvio antes da estimativa de movimento.

1. Estimar movimento a partir das imagens observadas $\mathbf{y}_k$;
2. Estimar o desvio $\mathbf{b}_k$;
3. Aplicar a correção do desvio de modo a obter $\hat{\mathbf{x}}_k$;
4. Repetir passos para todos os quadros do vídeo.

- Caso 2: Estimativa de movimento e correção de desvio de forma conjugada

1. Aplicar a correção do desvio estimado no passo anterior $\mathbf{b}_{k-1}$ em $\mathbf{y}_{k-1}$ e $\mathbf{y}_k$ de modo a obter $\hat{\mathbf{x}}_{k-1}$ e $\hat{\mathbf{x}}_k$. Uma possível interpretação é que retira-se a máscara nas imagens causadas por $\mathbf{b}$;
2. Estimar movimento a partir das estimativas de imagem real $\hat{\mathbf{x}}_{k-1}$ e $\hat{\mathbf{x}}_k$;
3. Estimar o desvio $\hat{\mathbf{b}}_k$;
4. Aplicar a correção do desvio de modo a obter $\hat{\mathbf{x}}_k$;
5. Repetir passos para todos os quadros do vídeo.

A Figura 4.5 mostra os resultados dos dois métodos. Nesta análise, o valor de $\sigma_{\mathbf{b}}$ foi elevado a 10^{-1} para evidenciar o efeito desejado.

Observa-se que, há uma melhora no desempenho geral do algoritmo se a estimativa de movimento for aplicada de forma conjugada à estimativa do desvio. A

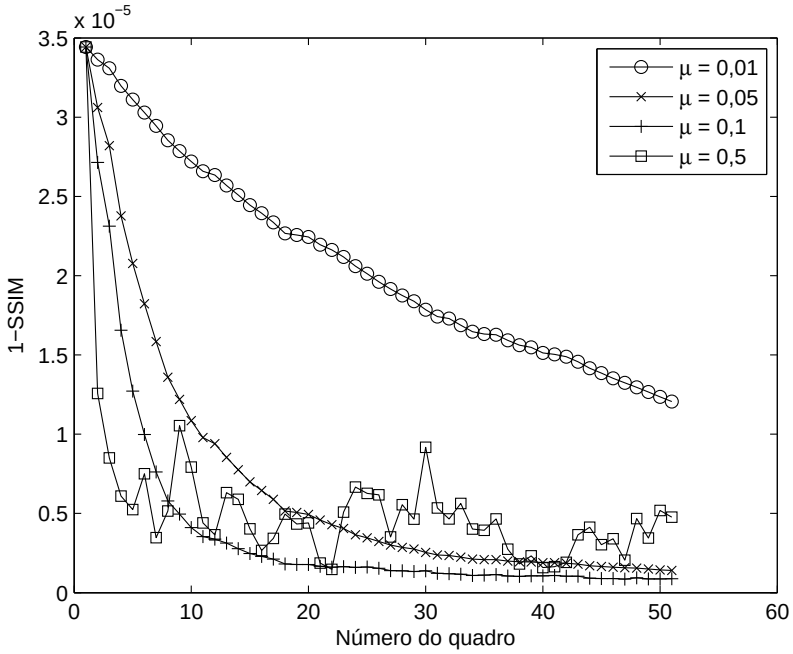

Figura 4.6: Efeito do passo de atualização.

interpretação é que, se a estimativa do desvio for coerente, o movimento obtido a partir das estimativas das imagens reais é mais confiável que o obtido diretamente das imagens observadas. O caso 2 será adotado como padrão para as próximas análises neste trabalho.

4.7.2 Efeito do passo de atualização

Foram realizadas simulações variando-se o passo de atualização μ. A Figura 4.6 ilustra os resultados. Para melhorar a visualização dos valores foi utilizado $(1 - SSIM)$ como medida de qualidade de imagem. Percebe-se que com $\mu = 0,01$ a convergência é muito lenta. Com $\mu = 0,5$ já existe instabilidade. Um valor ideal varia entre $0,05$ e $0,1$.

A Figura 4.7 mostra o quadro 36 de um vídeo sintético antes (esquerda) e após (direta) a correção do desvio com $\mu = 0,1$ e $\sigma_{\mathbf{b}} = 10^{-1}$. A Figura 4.8 também mostra o mesmo caso utilizando outra imagem.

Figura 4.7: Exemplo de correção de desvio.

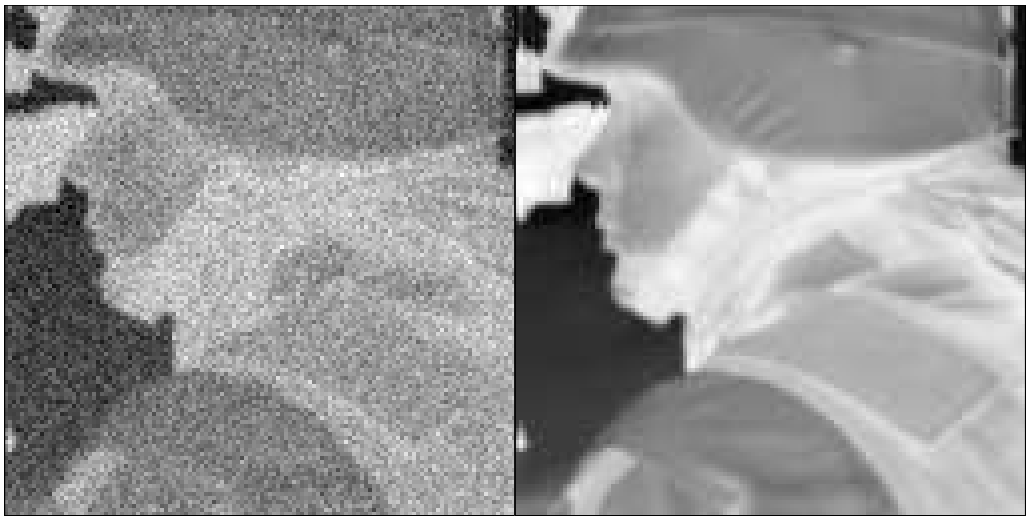

Figura 4.8: Exemplo 2 de correção de desvio.

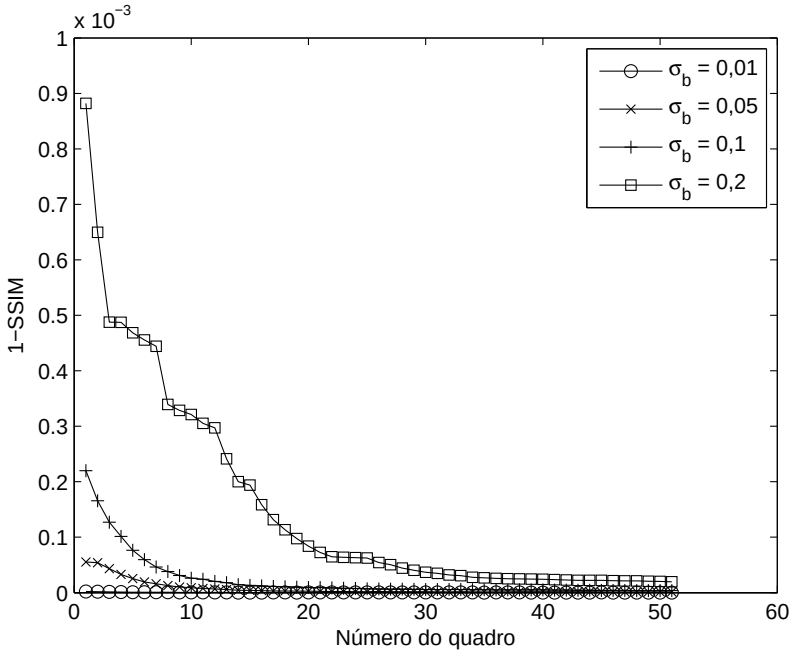

Figura 4.9: Robustez ao FPN.

4.7.3 Robustez ao FPN

Se o valor de $\sigma_{\mathbf{b}}$ for sendo elevado, o algoritmo tem o comportamento mostrado na Figura 4.9.

Percebe-se que para $\sigma_{\mathbf{b}} > 0,1$ a convergência começa a se tornar lenta. O algoritmo só se torna instável a partir de $\sigma_{\mathbf{b}} \geq 1$, provavelmente porque a estimativa de movimento começa a falhar.

O desempenho pode ser melhorado ajustando-se o passo de atualização. Porém, aumentando-se muito o passo de atualização, o algoritmo pode ficar instável.

4.7.4 Robustez ao ruído aleatório

Elevando-se o valor de $\sigma_{\mathbf{n}}$ o algoritmo tem o comportamento ilustrado na Figura 4.10. Para $\sigma_{\mathbf{n}} > 0,01$ a estabilidade do algoritmo começa a ficar comprometida. Porém, este valor é muito acima do estimado com vídeos reais.

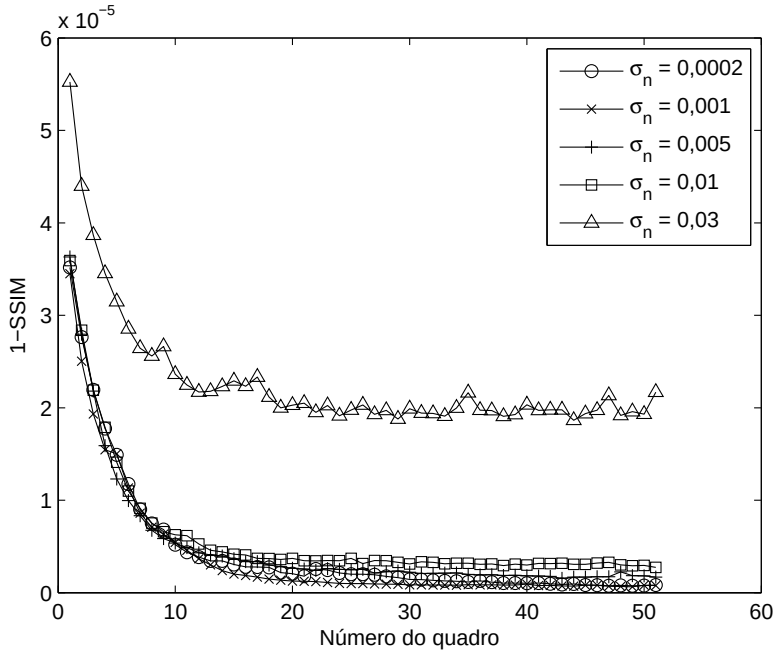

Figura 4.10: Robustez ao ruído aleatório.

4.8 Conclusões

O presente capítulo apresentou o desenvolvimento de um algoritmo para correção do desvio em vídeos infravermelhos com FPN. Primeiramente foi visto como a estimativa de movimento conjugada a correção do desvio pode melhorar o desempenho geral do algoritmo.

Foi apresentado o comportamento do algoritmo proposto em relação a diferentes valores de ruído FPN e sua robustez ao ruído aleatório.

Capítulo 5

Correção do ganho

5.1 Introdução

Este capítulo apresenta o desenvolvimento de um novo algoritmo para a correção do ganho de acordo com o modelo linear de ruído de padrão fixo. A chave da solução adotada reside no uso de matrizes para efetuar as operações de exponencial e logaritmo natural, o que tornou possível uma solução análoga à apresentada no Capítulo 4.

Após apresentar o algoritmo completo, são mostrados os resultados de simulação e ensaios e são feitas análises sobre a aplicação de correção do ganho de forma conjugada à correção do desvio. A solução apresentada é, segundo o conhecimento do autor, inédita para correção de ganho em vídeos com FPN.

5.2 Formulação do problema

Seja o modelo simplificado, considerando apenas o ganho, de observação de um sistema de aquisição de imagens infravermelhas com FPN

$$\mathbf{y}_k = \mathbf{a} \circ \mathbf{x}_k, \tag{5.1}$$

onde $\mathbf{y}_k$ é um vetor que representa a imagem observada no instante k, $\mathbf{a}$ é o vetor que representa os ganhos, "$\circ$" representa o produto Hadamard ou elemento-por-elemento, ou seja, $[\mathbf{y}_k]_i = [\mathbf{a}]_i [\mathbf{x}_k]_i$, e $\mathbf{x}_k$ é o vetor que representa a imagem real no instante k.

Aplicando-se o logaritmo natural elemento-por-elemento em ambos os lados da

equação (5.1) tem-se

$$\ln \mathbf{y}_k = \ln \mathbf{a} + \ln \mathbf{x}_k, \tag{5.2}$$

pois o produto Hadamard se transforma em soma após a aplicação do logaritmo.

Fazendo $\tilde{\mathbf{y}} = \ln \mathbf{y}$, $\tilde{\mathbf{a}} = \ln \mathbf{a}$ e $\tilde{\mathbf{x}} = \ln \mathbf{x}$ tem-se que

$$\tilde{\mathbf{y}}_k = \tilde{\mathbf{a}} + \tilde{\mathbf{x}}_k. \tag{5.3}$$

A equação (5.3) é muito semelhante à equação (4.1). O desenvolvimento que segue é, portanto, semelhante ao do Capítulo 4.

Como o ganho não é conhecido, pode-se obter uma estimativa de $\tilde{\mathbf{x}}$ através de

$$\hat{\tilde{\mathbf{x}}}_k = \tilde{\mathbf{y}}_k - \hat{\tilde{\mathbf{a}}}. \tag{5.4}$$

Conforme previamente descrito, o movimento global relativo entre dois quadros consecutivos do vídeo pode ser escrito como

$$\mathbf{x}_k = \mathbf{M}_k \mathbf{x}_{k-1} + \boldsymbol{\gamma}_k, \tag{5.5}$$

onde $\mathbf{M}_k$ é a matriz que implementa o movimento ou deslocamento entre os quadros $k-1$ e k, e $\boldsymbol{\gamma}_k$ é o vetor que modela as inovações no quadro seguinte que não podem ser obtidas com simples deslocamento.

Contudo, a matriz de movimento global $\mathbf{M}$ não pode ser aplicada diretamente a $\tilde{\mathbf{x}}$, pois foi definida para as imagens reais, ou seja, $\mathbf{x}$. É necessário uma transformação em $\tilde{\mathbf{x}}$ e a equação (5.5) pode ser rescrita como

$$\tilde{\mathbf{x}}_k = \ln \left(\mathbf{M}_k \exp \left(\tilde{\mathbf{x}}_{k-1} \right) \right), \tag{5.6}$$

já desconsiderando o vetor de inovações $\boldsymbol{\gamma}_k$ que não tem utilidade pois não pode ser comparado à imagem anterior.

As operações $\ln(\cdot)$ e $\exp(\cdot)$ elemento-por-elemento são de difícil manuseio. É interessante, logo, defini-las em forma de multiplicação de matriz para facilitar a operação de gradiente que será aplicada à equação de atualização da estimativa do ganho $\tilde{\mathbf{a}}$.

A equação (5.6) pode ser re-escrita como

$$\tilde{\mathbf{x}}_k = \mathbf{L}_k \mathbf{M}_k \mathbf{E}_{k-1} \tilde{\mathbf{x}}_{k-1}, \tag{5.7}$$

sendo que $\mathbf{E}_{k-1}$ e $\mathbf{L}_k$ são matrizes diagonais representam respectivamente a operação de exponencial e de logaritmo natural. Como $\mathbf{E}_{k-1}$ é aplicada a $\tilde{\mathbf{x}}_{k-1}$ pode ser obtida

$$\mathbf{E}_{k-1}\tilde{\mathbf{x}}_{k-1} = \exp\left(\tilde{\mathbf{x}}_{k-1}\right) \tag{5.8}$$

$$\mathbf{E}_{k-1} = \operatorname{diag}\left(\frac{\exp\left(\tilde{\mathbf{x}}_{k-1}\right)}{\tilde{\mathbf{x}}_{k-1}}\right), \tag{5.9}$$

onde $\mathbf{v} = \left(\frac{\mathbf{f}}{\mathbf{g}}\right)$ simboliza divisão feita elemento-por-elemento. Ou seja, $[\mathbf{v}]_i = \frac{[\mathbf{f}]_i}{[\mathbf{g}]_i}$.

Já a matriz $\mathbf{L}_k$ é aplicada a $\mathbf{x}_k = \mathbf{M}_k\mathbf{E}_{k-1}\tilde{\mathbf{x}}_{k-1}$. Logo é obtida por

$$\mathbf{L}_k\mathbf{x}_k = \ln\left(\mathbf{x}_k\right) \tag{5.10}$$

$$\mathbf{L}_k = \operatorname{diag}\left(\frac{\ln\left(\mathbf{x}_k\right)}{\mathbf{x}_k}\right), \tag{5.11}$$

onde a divisão é feita elemento-por-elemento.

Portanto, define-se o erro entre a imagem observada e sua estimativa com base somente na estimativa de movimento e na estimativa do desvio através de

$$\tilde{\boldsymbol{\epsilon}}_k = \tilde{\mathbf{y}}_k - \hat{\tilde{\mathbf{y}}}_k \tag{5.12}$$

$$\tilde{\boldsymbol{\epsilon}}_k = \tilde{\mathbf{y}}_k - \mathbf{L}_k\mathbf{M}_k\mathbf{E}_{k-1}\left(\tilde{\mathbf{y}}_{k-1} - \hat{\tilde{\mathbf{a}}}\right) - \hat{\tilde{\mathbf{a}}}, \tag{5.13}$$

onde $\tilde{\boldsymbol{\epsilon}}_k$ é o vetor erro de estimativa da imagem no instante k. O erro quadrático médio é definido como

$$\tilde{\varepsilon}_k = \frac{1}{N}\sum_{i=1}^{N}\left[\tilde{\epsilon}_k(i)\right]^2. \tag{5.14}$$

5.3 Gradiente descendente

Para obter uma estimativa do ganho, pode-se utilizar o gradiente descendente da seguinte forma:

$$\hat{\tilde{\mathbf{a}}}_{k+1} = \hat{\tilde{\mathbf{a}}}_k - \mu\nabla_{\tilde{\mathbf{a}}}\tilde{\varepsilon}_k, \tag{5.15}$$

onde μ representa o passo de atualização, $\nabla_{\tilde{\mathbf{a}}}$ representa o operador gradiente em relação a $\tilde{\mathbf{a}}$ e $\tilde{\varepsilon}_k$ é o erro quadrático médio. Neste caso, o índice k em $\tilde{\mathbf{a}}$ não indica variação temporal. A assunção inicial que $\tilde{\mathbf{a}}$ é constante continua válida e o índice indica atualização da estimativa a cada novo quadro.

Aplicando o gradiente ao erro ao quadrado, tem-se que

$$\nabla_{\tilde{\mathbf{a}}}\tilde{\varepsilon} = \frac{\partial\tilde{\boldsymbol{\epsilon}}}{\partial\tilde{\mathbf{a}}}\tilde{\boldsymbol{\epsilon}} \tag{5.16}$$

$$= \left[(\mathbf{LME})^{\mathrm{T}} - \mathbf{I}\right]\tilde{\boldsymbol{\epsilon}}. \tag{5.17}$$

A equação final de atualização pode ser finalmente escrita como

$$\hat{\tilde{\mathbf{x}}}_{k-1} = \tilde{\mathbf{y}}_{k-1} - \hat{\tilde{\mathbf{a}}}_k \tag{5.18}$$

$$\mathbf{E}_{k-1} = \mathrm{diag}\left(\frac{\exp(\hat{\tilde{\mathbf{x}}}_{k-1})}{\hat{\tilde{\mathbf{x}}}_{k-1}}\right) \tag{5.19}$$

$$\hat{\mathbf{x}}_k = \mathbf{M}_k\mathbf{E}_{k-1}\hat{\tilde{\mathbf{x}}}_{k-1} \tag{5.20}$$

$$\mathbf{L}_k = \mathrm{diag}\left(\frac{\ln(\hat{\mathbf{x}}_k)}{\hat{\mathbf{x}}_k}\right) \tag{5.21}$$

$$\hat{\tilde{\mathbf{y}}}_k = \mathbf{L}_k\hat{\mathbf{x}}_k + \hat{\tilde{\mathbf{a}}}_k \tag{5.22}$$

$$\tilde{\boldsymbol{\epsilon}}_k = \tilde{\mathbf{y}}_k - \hat{\tilde{\mathbf{y}}}_k \tag{5.23}$$

$$\hat{\tilde{\mathbf{a}}}_{k+1} = \hat{\tilde{\mathbf{a}}}_k - \mu\left[(\mathbf{L}_k\mathbf{M}_k\mathbf{E}_{k-1})^{\mathrm{T}} - \mathbf{I}\right]\tilde{\boldsymbol{\epsilon}}_k, \tag{5.24}$$

onde μ define o passo de atualização da estimativa $\hat{\tilde{\mathbf{a}}}$. Em seguida, obtém-se $\hat{\mathbf{a}} = \exp\left(\hat{\tilde{\mathbf{a}}}\right)$.

5.4 Resultados

Como na Seção 4.7, os resultados apresentados foram divididos em simulações e ensaios. Nas simulações são gerados vídeos sintéticos a partir de porções de uma imagem qualquer com FPN acrescentado de maneira controlada. Nos ensaios, os algoritmos são aplicados a vídeos reais contendo FPN e seus resultados são apresentados na Seção 8.4.

O modelo adotado para geração de vídeos sintéticos foi acrescido de um ruído aleatório que modela o ruído eletrônico do sistema e é representado por

$$\mathbf{y}_k = \mathbf{a} \circ \mathbf{x}_k + \mathbf{n}_k, \tag{5.25}$$

onde $\mathbf{n}_k$ é o vetor de ruído aleatório. Os pixels das imagens têm uma faixa dinâmica de 0 a 1, ou seja, $\{x \in \mathbb{R} | 0 \leq x \leq 1\}$. Os vetores $\mathbf{a}$ e $\mathbf{n}_k$ são vetores aleatórios com

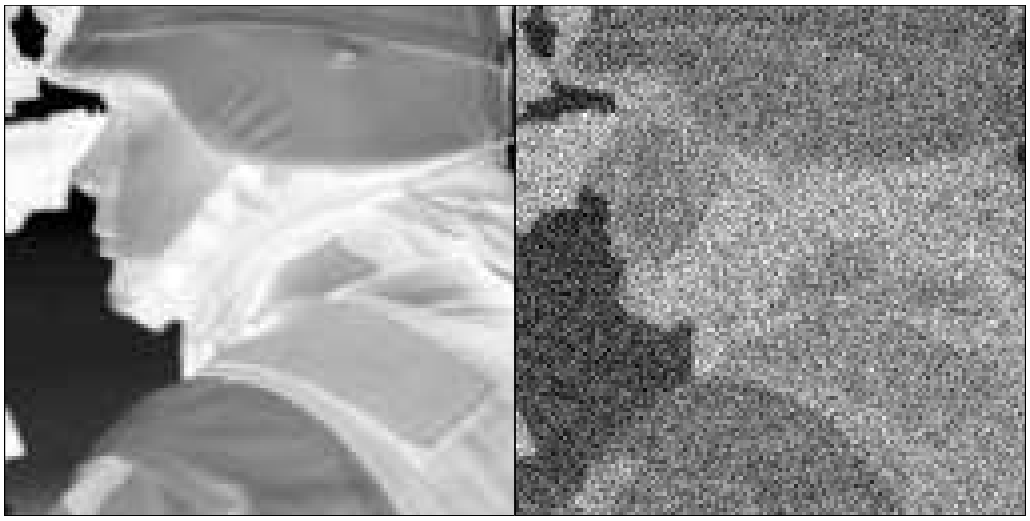

Figura 5.1: Vídeo sintético antes (esquerda) e após (direita) acréscimo de ruído de padrão fixo.

distribuição normal com médias $\mu_{\mathbf{a}} = 1$ e $\mu_{\mathbf{n}} = 0$ e desvio padrão $\sigma_{\mathbf{a}}$ e $\sigma_{\mathbf{n}}$ respectivamente. O valor estimado de $\sigma_{\mathbf{n}}$ a partir de vídeos na região do infravermelho reais foi de 2×10^{-4} e será usado esse valor nas simulações salvo quando especificado de outra forma. Para $\sigma_{\mathbf{a}}$ será utilizado 2×10^{-2} que, apesar de ser superior ao valor esperado na prática, servirá para testar os algoritmos em condições mais adversas. O movimento global, que simula o movimento da câmera, também será um vetor aleatório que representa deslocamentos na horizontal e vertical com média zero e desvio padrão de 2 pixels.

A Figura 5.1 mostra na esquerda um quadro de um vídeo sem ruído e na direita com ruído adicionado segundo a equação (5.25).

A Figura 5.2 mostra um exemplo do algoritmo de correção do ganho do ruído de padrão fixo.

5.4.1 Efeito do passo de atualização

A Figura 5.3 mostra o efeito da variação do passo de atualização no desempenho do algoritmo. Nota-se que para $\mu = 1$ o algoritmo torna-se instável para as condições impostas de $\sigma_{\mathbf{a}}$ e $\sigma_{\mathbf{n}}$. Um valor ideal gravita em torno de $0,5$ e esse valor será usado nas simulações daqui para frente.

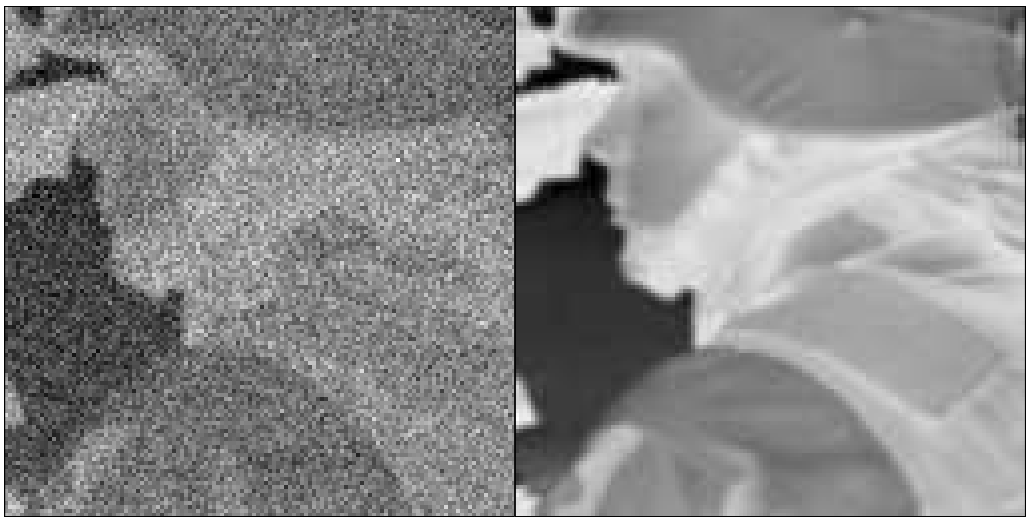

Figura 5.2: Exemplo de correção do ganho, quadro 28, $\sigma_{\mathbf{a}} = 0,02$ e $\mu = 0,5$.

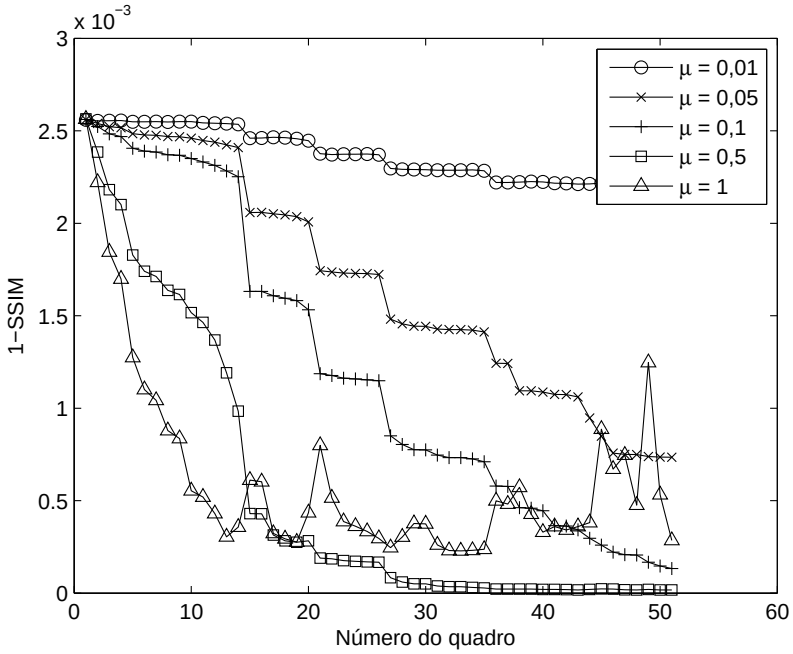

Figura 5.3: Efeito do passo de atualização.

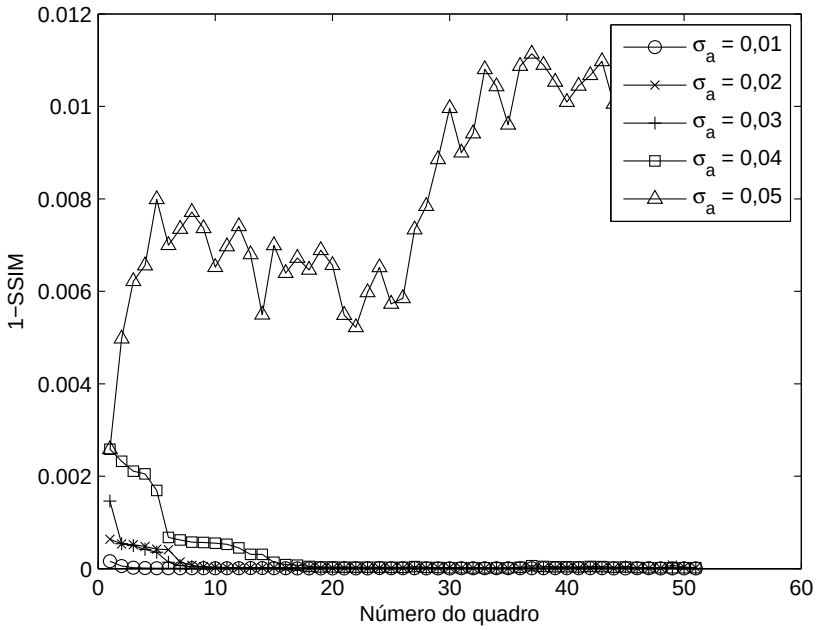

Figura 5.4: Robustez ao ruído multiplicativo de padrão fixo.

5.4.2 Robustez ao FPN

A Figura 5.4 mostra o comportamento do algoritmo a medida que o ganho do ruído de padrão fixo é elevado. Lembra-se que são valores acima do que se espera encontrar na prática. Percebe-se que para $\sigma_{\mathbf{a}} \geq 0,5$ o algoritmo se torna instável, provavelmente pois a estimativa de movimento global começa a falhar.

5.4.3 Robustez ao ruído aleatório

A Figura 5.5 mostra o comportamento do algoritmo com a elevação do desvio padrão do ruído aleatório. Percebe-se que para $\sigma_{\mathbf{n}} \geq 0,05$ o desempenho do algoritmo é afetado.

5.4.4 Aplicação combinada da correção de desvio e ganho

Nesta seção será explorada a aplicação conjugada da correção de desvio desenvolvida no Capítulo 4 com a correção do ganho desenvolvida neste capítulo. Para tanto, será usado o modelo completo de ruído de padrão fixo para geração de vídeo sintético descrito por

$$\mathbf{y}_k = \mathbf{a} \circ \mathbf{x}_k + \mathbf{b} + \mathbf{n}_k. \tag{5.26}$$

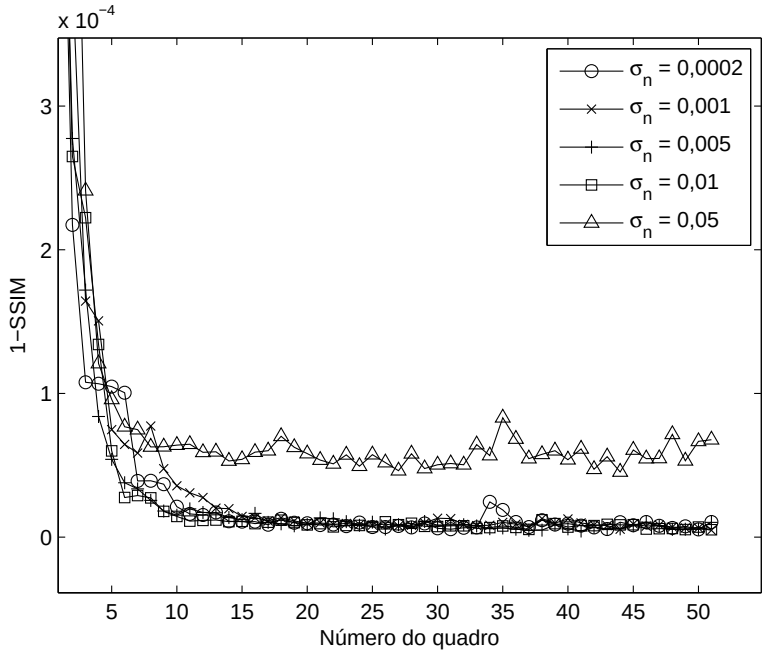

Figura 5.5: Robustez ao ruído aleatório.

A Figura 5.6 mostra um quadro de um vídeo sintético com FPN introduzido segundo a equação (5.26). Foram adotados $\sigma_{\mathbf{a}} = 0,02$, $\sigma_{\mathbf{b}} = 0,1$ e $\sigma_{\mathbf{n}} = 0,0002$.

A Figura 5.7 mostra quatro combinações de aplicação da correção de ganho e desvio. No primeiro caso a correção de desvio é aplicada, no segundo caso o ganho, no terceiro caso primeiro é aplicada a correção de desvio e após de ganho e finalmente no quarto caso é aplicado primeiramente a correção de ganho e após do desvio.

Numa primeira análise aparentemente não há diferença em termos do erro remanescente. Porém, se a região for ampliada percebe-se que as correções conjugadas levam vantagem em relação as outras, como ilustra a Figura 5.8.

Foram comparadas as combinações desvio-ganho e ganho-desvio com desvio e ganho sem re-estimar o movimento global nos casos de aplicação de duas correções. Dessa maneira, não houve vantagem em termos de melhoria na estimativa de movimento.

5.5 Conclusões

Este capítulo apresentou o desenvolvimento de um novo algoritmo para a correção do ganho. Foi utilizado um modelo simplificado onde o desvio foi descartado. Em

Figura 5.6: Exemplo de FPN com desvio e ganho.

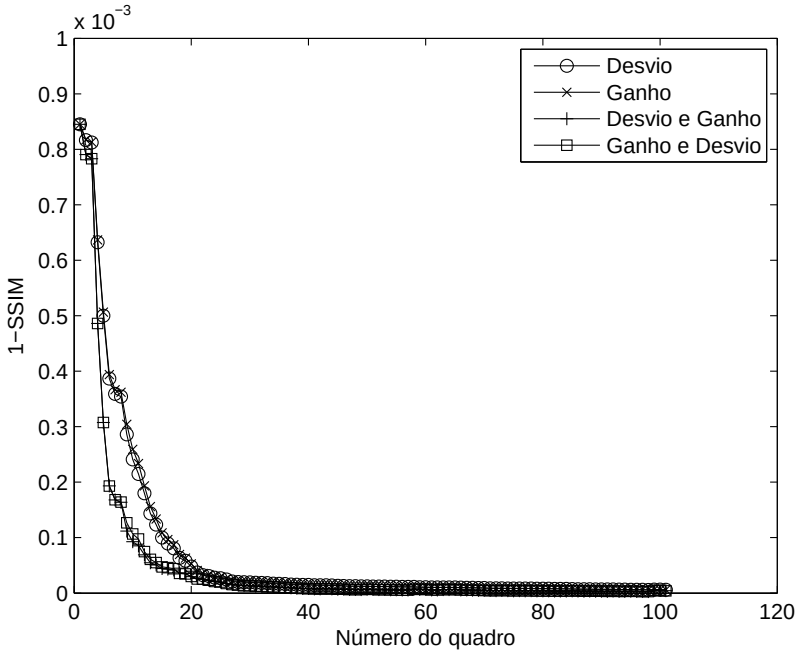

Figura 5.7: Combinações da correção do desvio e ganho.

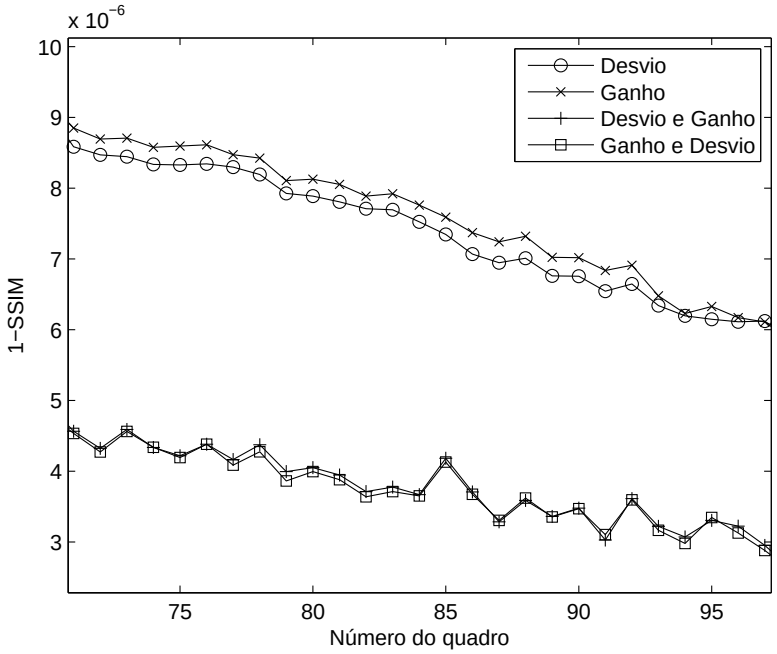

Figura 5.8: Combinações da correção do desvio e ganho com zoom.

seguida, aplicou-se logaritmo de modo a tornar o produto Hadamard em adição. Tal abordagem já havia sido sugerida em [2].

A novidade apresentada foi a utilização de matrizes diagonais para efetuar as operações de logaritmo e exponencial. Através deste artifício foi possível obter uma solução sem simplificações e análoga à apresentada no Capítulo 4.

Foram feitas simulações de modo a se conhecer o comportamento do algoritmo ao ruído FPN e ao ruído aleatório. Análises da aplicação conjugada da correção do ganho e desvio em vídeos sintéticos contendo FPN completo mostraram, como esperado, desempenho superior à correção individual. A sequência de aplicação das correções (desvio-ganho ou ganho-desvio) não mostrou efeitos consideráveis.

Capítulo 6

Correção do desvio e ganho através de método tensorial

6.1 Introdução

Neste capítulo será apresentado o desenvolvimento de um novo algoritmo para correção do desvio e do ganho em vídeos infravermelhos contendo FPN. É adotado o modelo completo de ruído onde são considerados um ganho e um desvio associados a cada pixel. Novamente, os parâmetros do modelo de ruído são assumidos como invariantes no tempo devido a sua lenta variação temporal.

A natureza do algoritmo apresentado é tensorial devido a uma derivada de um vetor em relação a uma matriz. Todavia, é apresentada uma solução para cada elemento de uma matriz diagonal de ganhos. No Apêndice C é apresentada a vetorização dos cálculos, que permite a solução do problema utilizando matrizes.

6.2 Formulação do problema

Seja o modelo de observação de vídeo infravermelho, contendo ruído de padrão fixo e considerando um desvio e um ganho independentes para cada pixel

$$\mathbf{y}_k = \mathbf{A}\mathbf{x}_k + \mathbf{b}, \tag{6.1}$$

onde $\mathbf{y}_k$ representa as imagens observadas, $\mathbf{A} = \text{diag}(a_1, \cdots, a_N)$ é uma matriz diagonal cujos elementos são os ganhos de cada um dos N pixels da imagem, $\mathbf{x}_k$ representa as imagens reais e $\mathbf{b}$ representa o desvio de cada pixel.

Como o ganho e o desvio não são conhecidos, pode-se obter uma estimativa da imagem real através de

$$\hat{\mathbf{x}}_k = \hat{\mathbf{A}}^{-1}\left(\mathbf{y}_k - \hat{\mathbf{b}}\right), \tag{6.2}$$

onde $\hat{\mathbf{A}}^{-1} = \text{diag}(\hat{a}_1^{-1}, \cdots, \hat{a}_N^{-1})$.

Considerando-se um movimento relativo entre dois quadros consecutivos de uma sequência de imagens ou vídeo, é possível escrever

$$\mathbf{x}_k = \mathbf{M}_k\mathbf{x}_{k-1} + \boldsymbol{\gamma}_k, \tag{6.3}$$

onde $\mathbf{M}_k$ é a matriz que implementa o movimento ou deslocamento entre os quadros $(k-1)$ e k, e $\boldsymbol{\gamma}_k$ é o vetor que modela as inovações no quadro seguinte que não podem ser obtidas com simples deslocamento.

Combinando as equações (6.2) e (6.3), desconsiderando o vetor de inovações γ_k e considerando o deslocamento entre dois quadros é possível escrever uma estimativa para a imagem real do quadro seguinte como

$$\hat{\mathbf{x}}_k = \mathbf{M}_k\hat{\mathbf{A}}^{-1}\left(\mathbf{y}_{k-1} - \hat{\mathbf{b}}\right). \tag{6.4}$$

Pode-se ainda combinar as equações (6.1) e (6.4) para se obter uma estimativa da próxima imagem observada através de

$$\hat{\mathbf{y}}_k = \hat{\mathbf{A}}\mathbf{M}_k\hat{\mathbf{A}}^{-1}\left(\mathbf{y}_{k-1} - \hat{\mathbf{b}}\right) + \hat{\mathbf{b}}. \tag{6.5}$$

Portanto, define-se o erro entre a imagem observada e sua estimativa com base somente na estimativa de movimento e na estimativa do desvio através de

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k \tag{6.6}$$

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \hat{\mathbf{A}}\mathbf{M}_k\hat{\mathbf{A}}^{-1}\left(\mathbf{y}_{k-1} - \hat{\mathbf{b}}\right) - \hat{\mathbf{b}}, \tag{6.7}$$

onde $\boldsymbol{\epsilon}_k$ é o vetor erro de imagem. O erro quadrático médio é definido como

$$\varepsilon_k = \frac{1}{N}\sum_{i=1}^{N}\left[\epsilon_k(i)\right]^2. \tag{6.8}$$

6.3 Gradiente descendente

Para obter uma estimativa do desvio, pode-se utilizar o gradiente descendente da seguinte forma:

$$\hat{\mathbf{b}}_{k+1} = \hat{\mathbf{b}}_k - \mu_{\mathbf{b}}\nabla_{\mathbf{b}}\varepsilon_k, \tag{6.9}$$

onde $\mu_{\mathbf{b}}$ representa o passo de atualização, $\nabla_{\mathbf{b}}$ representa o operador gradiente em relação a $\mathbf{b}$ e ε_k é o erro quadrático médio.

Aplicando o gradiente em relação ao erro, tem-se

$$\nabla_{\mathbf{b}}\varepsilon = \frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{b}}\boldsymbol{\epsilon} \tag{6.10}$$

$$= \left[\left(\mathbf{A}\mathbf{M}\mathbf{A}^{-1}\right)^{\mathrm{T}} - \mathbf{I}\right]\boldsymbol{\epsilon}. \tag{6.11}$$

A equação final de atualização pode ser escrita como

$$\mathbf{z}_k = \left(\mathbf{y}_{k-1} - \hat{\mathbf{b}}_k\right) \tag{6.12}$$

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \mathbf{A}\mathbf{M}_k\mathbf{A}^{-1}\mathbf{z}_k - \hat{\mathbf{b}}_k \tag{6.13}$$

$$\hat{\mathbf{b}}_{k+1} = \hat{\mathbf{b}}_k - \mu_{\mathbf{b}}\left[\left(\hat{\mathbf{A}}\mathbf{M}_k\hat{\mathbf{A}}^{-1}\right)^{\mathrm{T}} - \mathbf{I}\right]\boldsymbol{\epsilon}_k. \tag{6.14}$$

Para obter uma estimativa do ganho, pode-se utilizar o gradiente descendente da seguinte forma:

$$\hat{\mathbf{A}}_{k+1} = \hat{\mathbf{A}}_k - \mu_{\mathbf{A}}\nabla_{\mathbf{A}}\varepsilon_k, \tag{6.15}$$

onde $\mu_{\mathbf{A}}$ representa o passo de atualização, $\nabla_{\mathbf{A}}$ representa o operador gradiente em relação a $\mathbf{A}$ e ε_k é o erro quadrático médio.

Aplicando o gradiente em relação ao erro, tem-se

$$\nabla_{\mathbf{A}}\epsilon^2 = \frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{A}}\boldsymbol{\epsilon}. \tag{6.16}$$

A derivada do vetor erro em relação à matriz $\mathbf{A}$ é dada por

$$\frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{A}} = \begin{bmatrix} \frac{\partial \boldsymbol{\epsilon}}{\partial a_{11}} & \frac{\partial \boldsymbol{\epsilon}}{\partial a_{12}} & \cdots & \frac{\partial \boldsymbol{\epsilon}}{\partial a_{1N}} \\ \frac{\partial \boldsymbol{\epsilon}}{\partial a_{21}} & \frac{\partial \boldsymbol{\epsilon}}{\partial a_{22}} & \cdots & \frac{\partial \boldsymbol{\epsilon}}{\partial a_{2N}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \boldsymbol{\epsilon}}{\partial a_{N1}} & \frac{\partial \boldsymbol{\epsilon}}{\partial a_{N2}} & \cdots & \frac{\partial \boldsymbol{\epsilon}}{\partial a_{NN}} \end{bmatrix}. \tag{6.17}$$

Cada termo $\frac{\partial \boldsymbol{\epsilon}}{\partial a_{ii}}$ da matriz na equação (6.17) é, na verdade, um vetor. Logo, tal operação gera um tensor de terceira ordem de tamanho $N \times N \times N$. Isso dá ao método sua natureza tensorial.

Pelo fato da matriz $\mathbf{A}$ ser diagonal e para simplificar o processo, será desenvolvida a solução para o elemento a_{ii} da matriz. Assim sendo, a equação de atualização é dada por

$$\hat{a}_{ii,k+1} = \hat{a}_{ii,k} - \mu_a\nabla_{a_{ii}}\varepsilon_k, \tag{6.18}$$

onde μ_a representa o passo de atualização, $\nabla_{a_{ii}}$ representa o operador gradiente em relação a a_{ii} e ε_k é o erro quadrático médio. A atualização deve ser feita para cada elemento da diagonal de $\mathbf{A}$.

Aplicando o gradiente em relação ao erro, tem-se

$$\nabla_{a_{ii}}\varepsilon = \frac{\partial \boldsymbol{\epsilon}}{\partial a_{ii}}\boldsymbol{\epsilon}. \tag{6.19}$$

Logo,

$$\frac{\partial \boldsymbol{\epsilon}}{\partial a_{ii}} = \frac{\partial}{\partial a_{ii}}\left[\mathbf{y} - \mathbf{AMA}^{-1}\left(\mathbf{y}-\mathbf{b}\right) - \mathbf{b}\right] \tag{6.20}$$

$$= -\frac{\partial\left[\mathbf{AMA}^{-1}\mathbf{z}\right]}{\partial a_{ii}}, \tag{6.21}$$

onde $\mathbf{z} = (\mathbf{y} - \mathbf{b})$.

Antes de continuar a dedução é interessante explicitar o seguinte. Sejam $\mathbf{R}$ uma matriz, $\mathbf{s}$ um vetor e t um escalar. Então [23, 24]

$$\frac{\partial \mathbf{Rs}}{\partial t} = \frac{\partial}{\partial t}\begin{bmatrix} r_{11}s_1 + r_{12}s_2 + \ldots + r_{1N}s_N \\ r_{21}s_1 + r_{22}s_2 + \ldots + r_{2N}s_N \\ \vdots \\ r_{N1}s_1 + r_{N2}s_2 + \ldots + r_{NN}s_N \end{bmatrix} \tag{6.22}$$

$$= \begin{bmatrix} \frac{\partial}{\partial t}\left(r_{11}s_1 + r_{12}s_2 + \ldots + r_{1N}s_N\right) \\ \vdots \end{bmatrix}^{\mathrm{T}} \tag{6.23}$$

$$= \begin{bmatrix} \frac{\partial r_{11}}{\partial t}s_1 + r_{11}\frac{\partial s_1}{\partial t} + \ldots \\ \vdots \end{bmatrix}^{\mathrm{T}} \tag{6.24}$$

$$= \begin{bmatrix} \frac{\partial r_{11}}{\partial t}s_1 + \frac{\partial r_{12}}{\partial t}s_2 + \ldots \\ \vdots \end{bmatrix}^{\mathrm{T}} + \begin{bmatrix} r_{11}\frac{\partial s_1}{\partial t} + r_{12}\frac{\partial s_2}{\partial t} + \ldots \\ \vdots \end{bmatrix}^{\mathrm{T}} \tag{6.25}$$

$$= \left[\left(\frac{\partial \mathbf{R}}{\partial t}\right)^{\mathrm{T}}\mathbf{s} + \mathbf{R}\left(\frac{\partial \mathbf{s}}{\partial t}\right)^{\mathrm{T}}\right]^{\mathrm{T}} \tag{6.26}$$

$$= \mathbf{s}^{\mathrm{T}}\frac{\partial \mathbf{R}}{\partial t} + \frac{\partial \mathbf{s}}{\partial t}\mathbf{R}^{\mathrm{T}}. \tag{6.27}$$

Fazendo a derivada do produto tem-se

$$\frac{\partial\left[\mathbf{AMA}^{-1}\mathbf{z}\right]}{\partial a_{ii}} = \left[\mathbf{MA}^{-1}\mathbf{z}\right]^{\mathrm{T}}\frac{\partial \mathbf{A}}{\partial a_{ii}} + \frac{\partial \mathbf{MA}^{-1}\mathbf{z}}{\partial a_{ii}}\mathbf{A}^{\mathrm{T}}. \tag{6.28}$$

Desenvolvendo a segunda parcela da derivada tem-se que

$$\frac{\partial \mathbf{M}\mathbf{A}^{-1}\mathbf{z}}{\partial a_{ii}} = \left[\mathbf{A}^{-1}\mathbf{z}\right]^{\mathrm{T}} \frac{\partial \mathbf{M}}{\partial a_{ii}} + \frac{\partial \mathbf{A}^{-1}\mathbf{z}}{\partial a_{ii}}\mathbf{M}^{\mathrm{T}} \tag{6.29}$$

$$= \left[\mathbf{z}^{\mathrm{T}}\frac{\mathbf{A}^{-1}}{\partial a_{ii}} + \frac{\mathbf{z}}{\partial a_{ii}}\mathbf{A}^{-1}\right]\mathbf{M}^{\mathrm{T}} \tag{6.30}$$

$$= \mathbf{z}^{\mathrm{T}}\frac{\mathbf{A}^{-1}}{\partial a_{ii}}\mathbf{M}^{\mathrm{T}}. \tag{6.31}$$

Reorganizando os termos chega-se finalmente a

$$\frac{\partial\left[\mathbf{A}\mathbf{M}\mathbf{A}^{-1}\mathbf{z}\right]}{\partial a_{ii}} = \mathbf{z}^{\mathrm{T}}\left[\frac{\partial \mathbf{A}}{\partial a_{ii}}\mathbf{M}\mathbf{A}^{-1} + \mathbf{A}\mathbf{M}\frac{\partial \mathbf{A}^{-1}}{\partial a_{ii}}\right]^{\mathrm{T}}. \tag{6.32}$$

De forma a facilitar a visualização define-se

$$\bar{\mathbf{A}}_k = \mathbf{A}_k^{-1}. \tag{6.33}$$

Desenvolvendo as derivadas chega-se a

$$\dot{\mathbf{A}}_i = \frac{\partial \mathbf{A}}{\partial a_{ii}} = \begin{bmatrix} \mathbf{0} & & \\ & 1_{ii} & \\ & & \mathbf{0} \end{bmatrix}, \tag{6.34}$$

que é uma matriz que só contém zeros exceto o elemento 1_{ii} na ii-ésima posição que é igual a um. Também nota-se que

$$\dot{\bar{\mathbf{A}}}_i = \frac{\partial \mathbf{A}^{-1}}{\partial a_{ii}} = \begin{bmatrix} \mathbf{0} & & \\ & -a_{ii}^{-2} & \\ & & \mathbf{0} \end{bmatrix}, \tag{6.35}$$

onde somente o ii-ésimo elemento da matriz é diferente de zero.

A equação final de atualização dos ganhos pode ser escrita como

$$\mathbf{z}_k = \left(\mathbf{y}_{k-1} - \mathbf{b}\right) \tag{6.36}$$

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \mathbf{A}_k\mathbf{M}_k\bar{\mathbf{A}}_k\mathbf{z}_k - \mathbf{b} \tag{6.37}$$

$$\forall i \tag{6.38}$$

$$\dot{\mathbf{A}}_i = \begin{bmatrix} \mathbf{0} & & \\ & 1_{ii} & \\ & & \mathbf{0} \end{bmatrix} \dot{\bar{\mathbf{A}}}_i = \begin{bmatrix} \mathbf{0} & & \\ & -a_{ii}^{-2} & \\ & & \mathbf{0} \end{bmatrix} \tag{6.39}$$

$$\mathbf{u}_{i,k} = \left[\dot{\mathbf{A}}_i\mathbf{M}_k\bar{\mathbf{A}}_k + \mathbf{A}_k\mathbf{M}_k\dot{\bar{\mathbf{A}}}_i\right]\mathbf{z}_k \tag{6.40}$$

$$\hat{a}_{ii,k+1} = \hat{a}_{ii,k} + \mu_a\mathbf{u}_{i,k}^{\mathrm{T}}\boldsymbol{\epsilon}_k. \tag{6.41}$$

6.4 Aplicação do algoritmo

O algoritmo tensorial é aplicado de forma a refinar a estimativa do desvio e do ganho a cada novo quadro. A primeira etapa é calcular o erro *a priori*, isto é, com o desvio e ganho estimados no passo anterior, como definido na equação (6.7).

Em seguida, a estimativa do desvio e do ganho são atualizadas segundo as equações (6.14) e (6.41) nesta ordem. A estimativa do quadro original é obtido pela equação (6.2).

6.5 Resultados

Como foi feito nas seções 4.7 e 5.4, os resultados foram divididos em simulações e ensaios, sendo os resultados dos ensaios mostrados na Seção 8.4.

Nas simulações foi utilizado o modelo completo de FPN para gerar vídeos sintéticos contendo ruído de padrão fixo segundo a equação

$$\mathbf{y}_k = \mathbf{A}\mathbf{x}_k + \mathbf{b} + \mathbf{n}_k, \tag{6.42}$$

onde $\mathbf{n}_k$ é o vetor de ruído aleatório. Os pixels das imagens têm uma faixa dinâmica de 0 a 1, ou seja, $\{x \in \mathbb{R} | 0 \leq x \leq 1\}$. Como dito anteriormente, a matriz $\mathbf{A}$ é diagonal e implementa os ganhos individuais de cada pixel. Nas simulações ela foi gerada aleatoriamente com distribuição normal, média igual a um e desvio padrão $\sigma_{\mathbf{A}}$.

As Figuras 6.1, 6.2 e 6.3 mostram o mesmo quadro de um vídeo sintético sem FPN, com FPN e após a correção do desvio e do ganho pelo métodos tensorial. Em especial a Figura 6.3 mostra o vídeo original sem FPN e após a correção. Visualmente os vídeos são muito parecidos e pode-se dizer que correção remove totalmente o FPN.

6.5.1 Efeito da ordem de aplicação das correções

Nesta seção será estudado o efeito da ordem de aplicação das correções segundo as equações (6.14) e (6.41). No primeiro cenário a estimativa de desvio é atualizada, o erro é recalculado e em seguida é atualizada a estimativa de ganho. No segundo cenário o ganho é atualizado, o erro recalculado e finalmente a estimativa de desvio

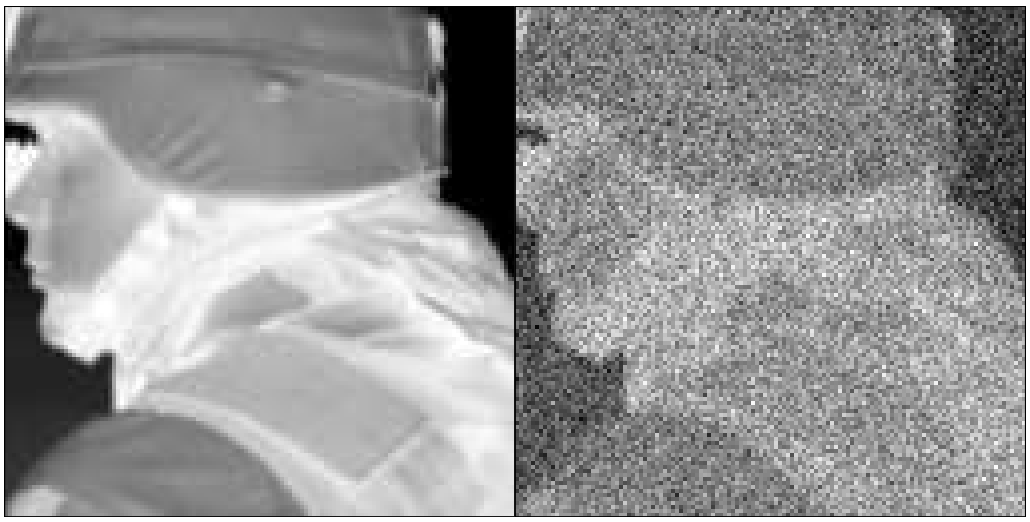

Figura 6.1: Vídeo original sem FPN (esquerda) e com FPN (direita).

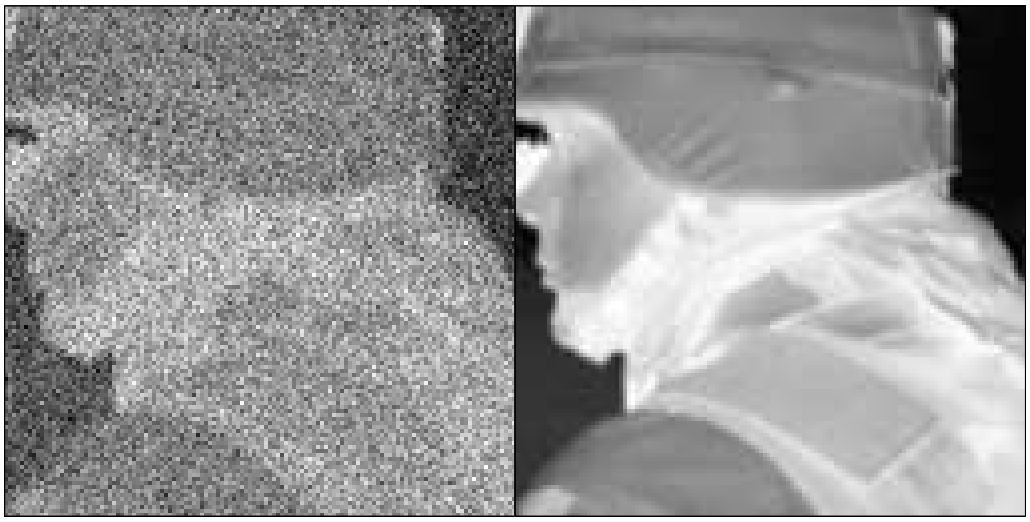

Figura 6.2: Vídeo com FPN (esquerda) e após correção com método tensorial (direita).

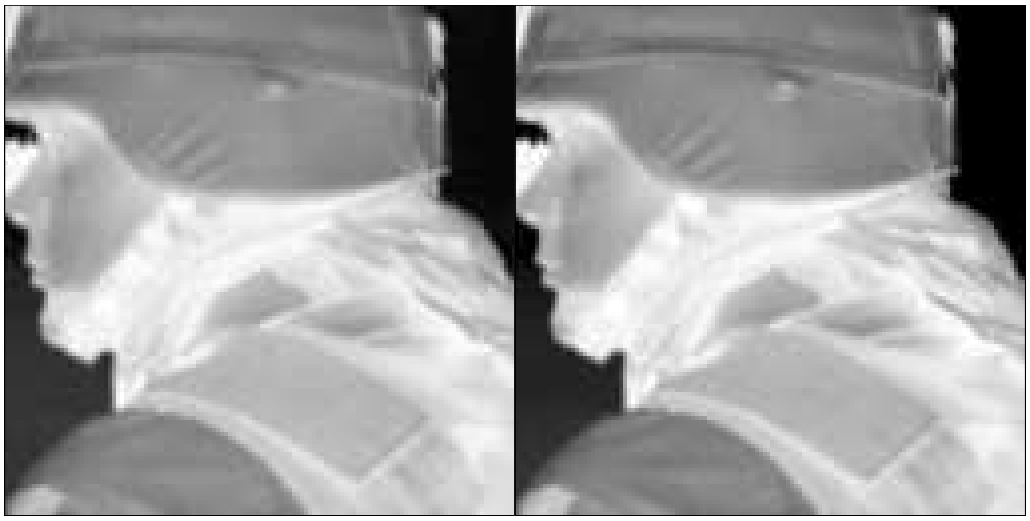

Figura 6.3: Vídeo após correção com método tensorial (esquerda) e original sem FPN (direita).

Tabela 6.1: Comparação da ordem das correções no método Tensorial.

Método	(1-SSIM)10^{-3} médio
Tensorial (desvio-ganho)	$0,7959$
Tensorial (ganho-desvio)	$0,7927$

é atualizada. As sequências serão chamadas respectivamente de desvio-ganho e ganho-desvio.

Foram feitas 50 simulações alterando aleatoriamente a imagem base do vídeo e os parâmetros de ruído FPN e aleatório ($\sigma_{\mathbf{a}} = \{0,2; 0,5\}$, $\sigma_{\mathbf{b}} = \{0,1; 0,5\}$ e $\sigma_{\mathbf{n}} = \{0,0003; 0,005\}$). Os resultados são apresentados na Tabela 6.1.

Percebe-se que a ordem da aplicação das correções no método Tensorial tem pouco efeito. A pequena diferença entre os resultados mostrou que, para os casos estudados, as duas ordens são equivalentes. Daqui em diante será usada a ordem desvio-ganho para os próximos experimentos.

6.5.2 Efeito do passos de atualização

As Figuras 6.4 e 6.5 descrevem o comportamento do algoritmo com a variação dos passos de atualização. Percebe-se que, para as condições impostas, o valor ideal de $\mu_{\mathbf{b}}$ gira em torno de $0,1$, pois acima disso já começa haver instabilidade. Daqui em diante será usado esse valor salvo mencionado contrariamente. No caso de μ_a esse

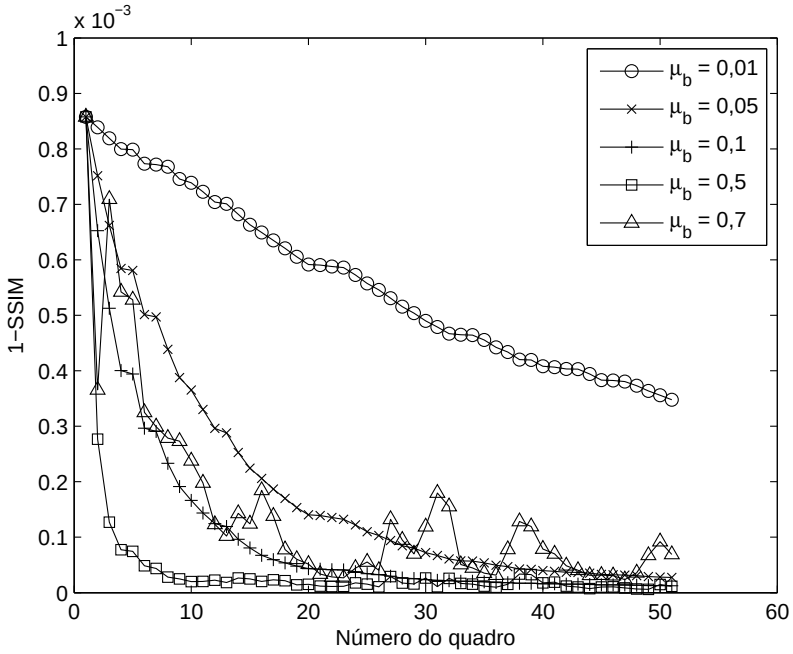

Figura 6.4: Efeito do passo de atualização $\mu_{\mathbf{b}}$, com $\mu_a = 0,001$.

valor ideal orbita em torno de $0,001$ e será usado esse valor daqui em diante.

6.5.3 Robustez ao FPN

A Figura 6.6 mostra a robustez do algoritmo em relação à parcela aditiva do ruído de padrão fixo. Para valores $\sigma_{\mathbf{b}} \geq 1$ não ocorre convergência, provavelmente pois a estimativa de movimento não é satisfatória.

A Figura 6.7 mostra o comportamento do algoritmo a variações de $\sigma_{\mathbf{a}}$. Para valores acima de $0,1$ não ocorre convergência.

6.5.4 Robustez ao ruído aleatório

A Figura 6.8 mostra a robustez do algoritmo ao ruído aleatório ou eletrônico. Nota-se que para valores próximos de $0,05$ o desempenho do algoritmo é reduzido.

6.6 Conclusões

O atual capítulo apresentou o desenvolvimento de um novo algoritmo de correção de desvio e ganho em vídeos infravermelhos contendo FPN. Foi considerado o modelo

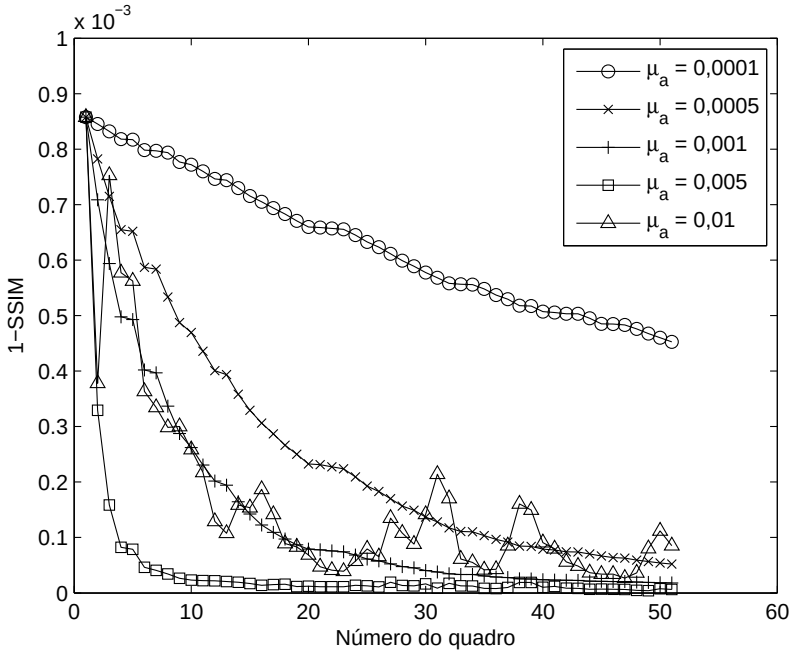

Figura 6.5: Efeito do passo de atualização μ_a, com $\mu_{\mathbf{b}} = 0,1$.

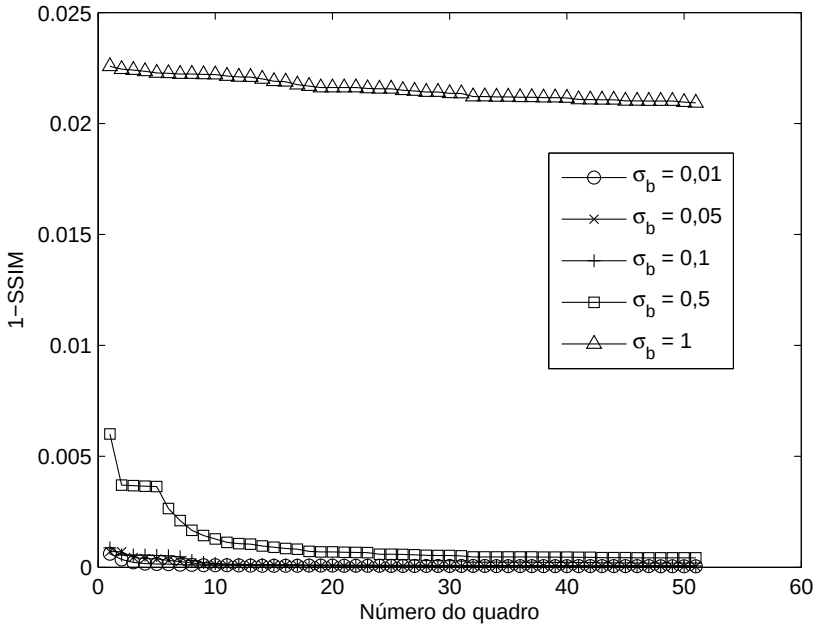

Figura 6.6: Robustez ao FPN (desvio).

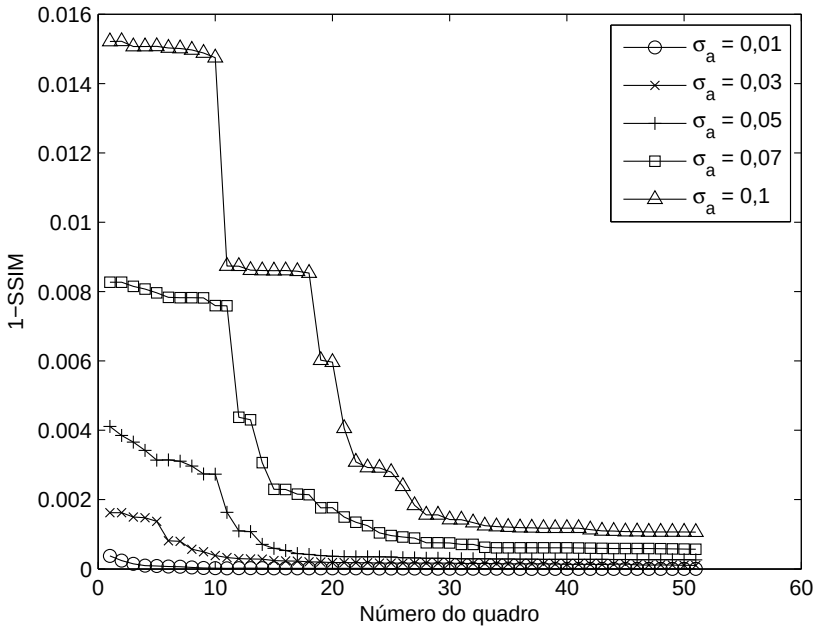

Figura 6.7: Robustez ao FPN (ganho).

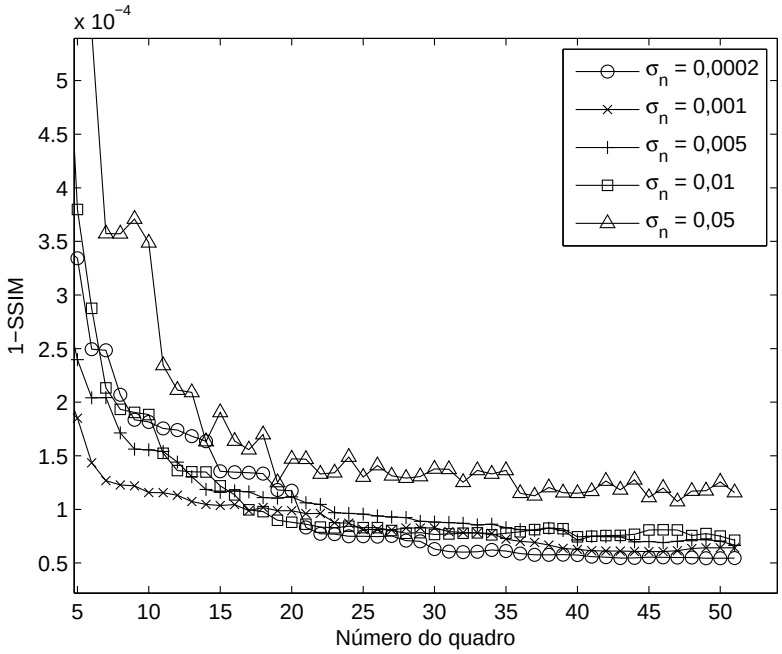

Figura 6.8: Robustez ao ruído aleatório.

completo de observação que associa a cada pixel da matriz de sensores infravermelhos um desvio e um ganho independentes que modificam e distorcem a saída da câmera.

O modelo de observação foi adaptado de modo a conter multiplicações tradicionais de matrizes e vetores. Em seguida foi utilizado o ferramental de álgebra linear para se deduzir as equações de atualização das estimativas de desvio e ganho através da filosofia de minimização do erro quadrático médio por gradiente descendente.

O algoritmo foi batizado de Tensorial, pois a atualização da estimativa dos ganhos em um único passo implicaria no uso de tensores. Alternativamente, foi apresentada a solução para cada elemento da matriz de ganhos, o que evitou o uso de multiplicações com tensores que precisariam ser definidas.

Foram apresentados os resultados de robustez ao FPN e ao ruído aleatório.

Capítulo 7

Correção de desvio e ganho por método Tensorial-RLS

7.1 Introdução

O presente capítulo mostra o desenvolvimento de um algoritmo para correção de desvio e ganho em vídeos contendo FPN batizado de Tensorial-RLS. Trata-se de uma adaptação do algoritmo tensorial desenvolvido no Capítulo 6 para os moldes RLS (*recursive least-squares*). Esta adaptação visou a uma melhora em velocidade de convergência e robustez a ruído, características marcantes da classe de algoritmos RLS. Para mais informações sobre os algoritmos RLS, consultar [26, 27, 28].

Pode-se dizer que o algoritmo RLS é um caso especial do filtro de Kalman [28]. De fato, essa observação já havia sido feita em [21]. A novidade apresentada aqui reside em dois fatores: primeiro um novo caminho de dedução partindo de uma formulação básica até a equação de atualização em vez de utilizar o modelo de filtro de Kalman adaptado para o problema. O segundo fator foi a correção de ganho por RLS que, do conhecimento do autor, trata-se de uma solução inédita. Para mais informações sobre filtro de Kalman consultar [27, 29, 30].

7.2 Formulação do problema

É adotado o mesmo modelo utilizado no Capítulo 6, repetido aqui por conveniência

$$\mathbf{y}_k = \mathbf{A}\mathbf{x}_k + \mathbf{b}, \tag{7.1}$$

onde $\mathbf{y}_k$ representa as imagens observadas, $\mathbf{A} = \text{diag}(a_1, \cdots, a_N)$ é uma matriz diagonal cujos elementos são os ganhos de cada um dos N pixels da imagem, $\mathbf{x}_k$ representa as imagens reais e $\mathbf{b}$ representa o desvio de cada pixel.

Seguindo-se a mesma dedução, define-se o erro entre a imagem observada e sua estimativa com base somente na estimativa de movimento e na estimativa do desvio através de

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k \tag{7.2}$$

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \hat{\mathbf{A}}\mathbf{M}_k\hat{\mathbf{A}}^{-1}\left(\mathbf{y}_{k-1} - \hat{\mathbf{b}}\right) - \hat{\mathbf{b}}, \tag{7.3}$$

onde $\boldsymbol{\epsilon}_k$ é o vetor erro de imagem. O erro quadrático médio é definido como

$$\varepsilon_k = \frac{1}{N}\sum_{i=1}^{N}\left[\epsilon_k(i)\right]^2. \tag{7.4}$$

Para definições precisas de $\mathbf{M}_k$ consultar Seção 4.4, e de $\hat{\mathbf{y}}_k$ e $\hat{\mathbf{A}}^{-1}$ consultar Seção 6.1.

7.3 Algoritmo RLS

A classe de algoritmos RLS tem por objetivo minimizar a soma dos quadrados dos erros entre a saída de um sistema e o sinal desejado [28]. A sigla RLS significa *recursive least-squares* e pode ser traduzida como mínimos quadrados recursivo. Ao contrário dos algoritmos apresentados até o momento neste trabalho, onde o gradiente era calculado a cada nova iteração e o erro era minimizado para o último par de quadros, os algoritmos RLS definem o erro a ser minimizado como [28]

$$\xi_k = \sum_{i=0}^{k}\lambda^{k-i}\varepsilon_i, \tag{7.5}$$

onde $0 \ll \lambda \leq 1$ é chamado fator de esquecimento (*forgetting factor*). Ou seja, o objetivo é minimizar os erros das últimas iterações dando menos peso quando mais

antiga for a saída do sistema de acordo com o fator de esquecimento λ. Portanto, o algoritmo RLS é mais robusto a ruídos e pode se adaptar a mudanças do sistema com a correta escolha do fator λ.

Nota-se que, se $\lambda = 1$ não há esquecimento e as estimativas levam em conta todo o passado. Em sistemas onde existem mudanças dinâmicas é interessante que o estimador "esqueça um pouco o passado" para que se adapte melhor às novas condições. No caso do FPN onde existe uma lenta variação temporal, é interessante escolher $\lambda < 1$.

É possível mostrar após alguma manipulação algébrica [28] que a equação de atualização do algoritmo RLS é dada por

$$\hat{\mathbf{b}}_{k+1} = \hat{\mathbf{b}}_k - \hat{\mathbf{H}}_k^{-1} \nabla_{\mathbf{b}} \varepsilon_k, \tag{7.6}$$

onde $\hat{\mathbf{H}}_k$ é uma estimativa da matriz Hessiana em relação a $\mathbf{b}$ e $\nabla_{\mathbf{b}} \varepsilon_k$ é o gradiente do erro *a priori* [28], exatamente como foi feito no Capítulo 4. Portanto, basta ser calculada uma estimativa da matriz Hessiana a cada nova iteração para se ter o algoritmo RLS.

De fato, o algoritmo apresentado no Capítulo 4 pode ser visto como uma solução LMS (*least-mean-square* ou mínimo quadrático médio) onde apenas o erro da última saída do sistema é considerado. A solução apresentada na equação (7.6) é também chamada, além de RLS, de LMS-Newton, pois se trata de uma versão do algoritmo LMS adaptada para o método de Newton. Este utiliza a inversa da matriz Hessiana para direcionar o gradiente da função erro direto para o erro mínimo invés de somente na direção oposta ao seu crescimento, caso a função erro seja quadrática [28].

A matriz Hessiana é definida como [24]

$$\mathbf{H} \triangleq \nabla_{\mathbf{b}}^2 \xi \tag{7.7}$$

$$\triangleq \frac{\partial^2 \xi}{\partial \mathbf{b} \partial \mathbf{b}^{\mathrm{T}}}. \tag{7.8}$$

A equação (7.5) pode ser re-escrita como

$$\xi_k = \sum_{i=0}^{k} \lambda^{k-i} \varepsilon_i \tag{7.9}$$

$$= \sum_{i=0}^{k-1} \lambda^{k-i} \varepsilon_i + \varepsilon_k \tag{7.10}$$

$$= \lambda \sum_{i=0}^{k-1} \lambda^{k-1-i} \varepsilon_i + \varepsilon_k \tag{7.11}$$

$$= \lambda \xi_{k-1} + \varepsilon_k. \tag{7.12}$$

Portanto, a matriz Hessiana pode ser calculada de uma forma recursiva e incremental aplicando o operador $\nabla^2_{\mathbf{b}}$ à equação (7.12) obtendo-se

$$\hat{\mathbf{H}}_k = \nabla^2_{\mathbf{b}} \xi_k \tag{7.13}$$

$$= \nabla^2_{\mathbf{b}} \left[\lambda \xi_{k-1} + \varepsilon_k \right] \tag{7.14}$$

$$= \lambda \hat{\mathbf{H}}_{k-1} + \frac{\partial^2 \varepsilon_k}{\partial \mathbf{b} \partial \mathbf{b}^{\mathrm{T}}}. \tag{7.15}$$

Desta maneira, a cada novo passo a matriz Hessiana é atualizada.

7.4 Correção do desvio por método Tensorial-RLS

O segundo termo da equação (7.15) é obtido da seguinte maneira:

$$\frac{\partial^2 \varepsilon}{\partial \mathbf{b} \partial \mathbf{b}^{\mathrm{T}}} = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial^2}{\partial \mathbf{b} \partial \mathbf{b}^{\mathrm{T}}} \left[\epsilon(i) \right]^2 \tag{7.16}$$

$$= \frac{2}{N} \sum_{i=1}^{N} \frac{\partial}{\partial \mathbf{b}} \left[\frac{\partial \epsilon(i)}{\partial \mathbf{b}^{\mathrm{T}}} \epsilon(i) \right] \tag{7.17}$$

$$= \frac{2}{N} \sum_{i=1}^{N} \left[\frac{\partial^2 \epsilon(i)}{\partial \mathbf{b} \partial \mathbf{b}^{\mathrm{T}}} \epsilon(i) + \frac{\partial \epsilon(i)}{\partial \mathbf{b}} \frac{\partial \epsilon(i)}{\partial \mathbf{b}^{\mathrm{T}}} \right]. \tag{7.18}$$

A primeira parcela do somatório é igual a zero, pois a derivada parcial da equação (4.22) em relação a $\mathbf{b}^{\mathrm{T}}$ é igual a zero. O outro termo pode ser agrupado finalmente em

$$\frac{\partial^2 \varepsilon}{\partial \mathbf{b} \partial \mathbf{b}^{\mathrm{T}}} = \frac{2}{N} \frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{b}} \frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{b}^{\mathrm{T}}}. \tag{7.19}$$

O termo $\frac{2}{N}$ é uma constante comum a todos os termos da equação (7.15) e pode ser colocado em evidência. Neste ponto, será definida a matriz

$$\hat{\mathbf{H}} = \frac{2}{N}\hat{\mathbf{H}}', \tag{7.20}$$

com

$$\hat{\mathbf{H}}'_k = \lambda\hat{\mathbf{H}}'_{k-1} + \frac{\partial\boldsymbol{\epsilon}_k}{\partial\mathbf{b}}\frac{\partial\boldsymbol{\epsilon}_k}{\partial\mathbf{b}^{\mathrm{T}}}. \tag{7.21}$$

No Capítulo 4, equação (4.20), foi mostrado que

$$\nabla_{\mathbf{b}}\varepsilon = \frac{2}{N}\frac{\partial\boldsymbol{\epsilon}}{\partial\mathbf{b}}\boldsymbol{\epsilon}. \tag{7.22}$$

Combinando as equações (7.6), (7.20) e (7.22) tem-se que

$$\hat{\mathbf{b}}_{k+1} = \hat{\mathbf{b}}_k - \frac{N}{2}\hat{\mathbf{H}}_k^{\prime-1}\frac{2}{N}\frac{\partial\boldsymbol{\epsilon}_k}{\partial\mathbf{b}}\boldsymbol{\epsilon}_k \tag{7.23}$$

$$= \hat{\mathbf{b}}_k - \hat{\mathbf{H}}_k^{\prime-1}\frac{\partial\boldsymbol{\epsilon}_k}{\partial\mathbf{b}}\boldsymbol{\epsilon}_k. \tag{7.24}$$

Além disso, foi mostrado na equação (6.11) que

$$\frac{\partial\boldsymbol{\epsilon}}{\partial\mathbf{b}} = \left(\mathbf{AMA}^{-1}\right)^{\mathrm{T}} - \mathbf{I}, \tag{7.25}$$

e pode ser facilmente mostrado que

$$\frac{\partial\boldsymbol{\epsilon}}{\partial\mathbf{b}^{\mathrm{T}}} = \left(\mathbf{AMA}^{-1}\right) - \mathbf{I}. \tag{7.26}$$

Combinando-se as equações (7.3), (7.21), (7.25) e (7.26) pode-se escrever o algoritmo completo de correção de desvio por método Tensorial-RLS como

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \mathbf{AM}_k\mathbf{A}^{-1}\left(\mathbf{y}_{k-1} - \hat{\mathbf{b}}_k\right) - \hat{\mathbf{b}}_k \tag{7.27}$$

$$\hat{\mathbf{H}}'_k = \lambda\hat{\mathbf{H}}'_{k-1} + \left[\left(\mathbf{AM}_k\mathbf{A}^{-1}\right)^{\mathrm{T}} - \mathbf{I}\right]\cdot\left[\left(\mathbf{AM}_k\mathbf{A}^{-1}\right) - \mathbf{I}\right] \tag{7.28}$$

$$\hat{\mathbf{b}}_{k+1} = \hat{\mathbf{b}}_k - \hat{\mathbf{H}}_k^{\prime-1}\left[\left(\mathbf{AM}_k\mathbf{A}^{-1}\right)^{\mathrm{T}} - \mathbf{I}\right]\boldsymbol{\epsilon}_k. \tag{7.29}$$

O algoritmo apresentado tem o inconveniente da inversão da matriz $\hat{\mathbf{H}}'_k$. Para contornar esse problema, pode ser usado o método iterativo do gradiente conjugado para resolver um sistema de equações e evitar a inversão da matriz [21,31]. Supondo o sistema $\mathbf{u} = \mathbf{H}'\mathbf{v}$ tem-se que $\mathbf{v} = \mathbf{H}^{\prime-1}\mathbf{u}$. Fazendo $\mathbf{u} = \left[\left(\mathbf{AMA}^{-1}\right)^{\mathrm{T}} - \mathbf{I}\right]\boldsymbol{\epsilon}$, obtém-se $\mathbf{v}$ resolvendo-se o sistema.

O algoritmo se torna, então,

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \mathbf{A}\mathbf{M}_k\mathbf{A}^{-1}\left(\mathbf{y}_{k-1} - \hat{\mathbf{b}}_k\right) - \hat{\mathbf{b}}_k \tag{7.30}$$

$$\hat{\mathbf{H}}'_k = \lambda\hat{\mathbf{H}}'_{k-1} + \left[\left(\mathbf{A}\mathbf{M}_k\mathbf{A}^{-1}\right)^{\mathrm{T}} - \mathbf{I}\right] \cdot \left[\left(\mathbf{A}\mathbf{M}_k\mathbf{A}^{-1}\right) - \mathbf{I}\right] \tag{7.31}$$

$$\mathbf{u}_k = \left[\left(\mathbf{A}\mathbf{M}_k\mathbf{A}^{-1}\right)^{\mathrm{T}} - \mathbf{I}\right]\boldsymbol{\epsilon}_k \tag{7.32}$$

$$\mathbf{v}_k = CG(\hat{\mathbf{H}}'_k, \mathbf{u}_k) \tag{7.33}$$

$$\hat{\mathbf{b}}_{k+1} = \hat{\mathbf{b}}_k - \mathbf{v}_k, \tag{7.34}$$

onde CG representa a solução do sistema de equações por gradiente conjugado.

7.5 Correção do ganho por método Tensorial-RLS

A equação de atualização da estimativa do ganho pelo método Tensorial-RLS pode ser escrita, por analogia, como

$$\hat{a}_{ii,k+1} = \hat{a}_{ii,k} - \hat{\boldsymbol{\Gamma}}_k^{-1}\nabla_{a_{ii}}\varepsilon_k, \tag{7.35}$$

onde $\hat{\boldsymbol{\Gamma}}_k$ é uma estimativa da matriz Hessiana em relação a a_{ii} e $\nabla_{a_{ii}}\varepsilon_k$ é o gradiente do erro *a priori* [28], exatamente como foi feito no Capítulo 6. Portanto, basta ser calculada uma estimativa da matriz Hessiana a cada nova iteração para se ter o algoritmo RLS.

A matriz Hessiana é definida neste caso como [24]

$$\boldsymbol{\Gamma} \triangleq \nabla^2_{a_{ii}}\xi \tag{7.36}$$

$$\triangleq \frac{\partial^2\xi}{\partial a_{ii}^2}. \tag{7.37}$$

Analogamente ao que foi feito na equação (7.18) o seguinte termo é desenvolvido como

$$\frac{\partial^2\varepsilon}{\partial a_{ii}^2} = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial^2}{\partial a_{ii}^2}\left[\epsilon(i)\right]^2 \tag{7.38}$$

$$= \frac{2}{N}\sum_{i=1}^{N}\frac{\partial}{\partial a_{ii}}\left[\frac{\partial\epsilon(i)}{\partial a_{ii}}\epsilon(i)\right] \tag{7.39}$$

$$= \frac{2}{N}\sum_{i=1}^{N}\left[\frac{\partial^2\epsilon(i)}{\partial a_{ii}^2}\epsilon(i) + \frac{\partial\epsilon(i)}{\partial a_{ii}}\frac{\partial\epsilon(i)}{\partial a_{ii}}\right] \tag{7.40}$$

$$= \frac{2}{N}\left\{\left[\frac{\partial^2\boldsymbol{\epsilon}}{\partial a_{ii}^2}\right]^{\mathrm{T}} \cdot \boldsymbol{\epsilon} + \frac{\partial\boldsymbol{\epsilon}}{\partial a_{ii}}\left[\frac{\partial\boldsymbol{\epsilon}}{\partial a_{ii}}\right]^{\mathrm{T}}\right\}. \tag{7.41}$$

Os dois últimos termos da equação (7.41) já foram calculados no Capítulo 6 e apresentados nas equações (6.21) e (6.32).

O termo que ainda não havia sido definido neste trabalho e que é chamado de gradiente de segunda ordem, já que não se trata verdadeiramente de uma matriz Hessiana, é dado por [23]

$$\frac{\partial^2 \boldsymbol{\epsilon}}{\partial a_{ii}^2} \triangleq \left[\begin{array}{cccc} \frac{\partial^2 \epsilon_1}{\partial a_{ii}^2} & \frac{\partial^2 \epsilon_2}{\partial a_{ii}^2} & \dots & \frac{\partial^2 \epsilon_N}{\partial a_{ii}^2} \end{array} \right]. \tag{7.42}$$

Aplicando-se o gradiente de segunda ordem ao erro tem-se que

$$\frac{\partial^2 \boldsymbol{\epsilon}}{\partial a_{ii}^2} = \frac{\partial^2}{\partial a_{ii}^2} \left[\mathbf{y} - \mathbf{AMA}^{-1} (\mathbf{y} - \mathbf{b}) - \mathbf{b} \right] \tag{7.43}$$

$$= -\frac{\partial^2 \left[\mathbf{AMA}^{-1} \mathbf{z} \right]}{\partial a_{ii}^2}, \tag{7.44}$$

onde $\mathbf{z} = (\mathbf{y} - \mathbf{b})$.

Antes de continuar a dedução é interessante explicitar o seguinte. Sejam $\mathbf{R}$ uma matriz, $\mathbf{s}$ um vetor e t um escalar. Então [23, 24]

$$\frac{\partial^2 \mathbf{Rs}}{\partial t^2} = \frac{\partial^2}{\partial t^2} \left[\begin{array}{c} r_{11} s_1 + r_{12} s_2 + \ldots + r_{1N} s_N \\ r_{21} s_1 + r_{22} s_2 + \ldots + r_{2N} s_N \\ \vdots \\ r_{N1} s_1 + r_{N2} s_2 + \ldots + r_{NN} s_N \end{array} \right] \tag{7.45}$$

$$= \left[\begin{array}{c} \frac{\partial^2}{\partial t^2} \left(r_{11} s_1 + r_{12} s_2 + \ldots + r_{1N} s_N \right) \\ \vdots \end{array} \right]^{\mathrm{T}} \tag{7.46}$$

$$= \left[\begin{array}{c} \frac{\partial}{\partial t} \left(\frac{\partial r_{11}}{\partial t} s_1 + \frac{\partial r_{12}}{\partial t} s_2 \right) + \ldots \\ \vdots \end{array} \right]^{\mathrm{T}} + \left[\begin{array}{c} \frac{\partial}{\partial t} \left(r_{11} \frac{\partial s_1}{\partial t} + r_{12} \frac{\partial s_2}{\partial t} \right) + \ldots \\ \vdots \end{array} \right]^{\mathrm{T}} \tag{7.47}$$

$$= \left[\begin{array}{c} \frac{\partial^2 r_{11}}{\partial t^2} s_1 + \frac{\partial r_{11}}{\partial t} \frac{\partial s_1}{\partial t} + \ldots \\ \vdots \end{array} \right]^{\mathrm{T}} + \left[\begin{array}{c} \frac{\partial r_{11}}{\partial t} \frac{\partial s_1}{\partial t} + r_{11} \frac{\partial^2 s_1}{\partial t^2} + \ldots \\ \vdots \end{array} \right]^{\mathrm{T}} \tag{7.48}$$

$$= \left[\begin{array}{c} \frac{\partial^2 r_{11}}{\partial t^2} s_1 + \ldots \\ \vdots \end{array} \right]^{\mathrm{T}} + \left[\begin{array}{c} 2 \frac{\partial r_{11}}{\partial t} \frac{\partial s_1}{\partial t} + \ldots \\ \vdots \end{array} \right]^{\mathrm{T}} + \left[\begin{array}{c} r_{11} \frac{\partial^2 s_1}{\partial t^2} + \ldots \\ \vdots \end{array} \right]^{\mathrm{T}} \tag{7.49}$$

$$\frac{\partial^2 \mathbf{Rs}}{\partial t^2} = \left[\frac{\partial^2 \mathbf{R}}{\partial t^2} \mathbf{s} + 2 \frac{\partial \mathbf{R}}{\partial t} \left(\frac{\partial \mathbf{s}}{\partial t} \right)^{\mathrm{T}} + \mathbf{R} \left(\frac{\partial^2 \mathbf{s}}{\partial t^2} \right)^{\mathrm{T}} \right]^{\mathrm{T}}. \tag{7.50}$$

Aplicando-se as regras encontradas na equação (7.50) na equação (7.44) chega-se a

$$\frac{\partial^2 \left[\mathbf{A}\mathbf{M}\mathbf{A}^{-1}\mathbf{z}\right]}{\partial a_{ii}^2} = \mathbf{z}^{\mathrm{T}} \left[2\frac{\partial \mathbf{A}}{\partial a_{ii}}\mathbf{M}\frac{\partial \mathbf{A}^{-1}}{\partial a_{ii}} + \mathbf{A}\mathbf{M}\frac{\partial^2 \mathbf{A}^{-1}}{\partial a_{ii}^2} \right]^{\mathrm{T}}. \tag{7.51}$$

De forma a facilitar a visualização define-se

$$\bar{\mathbf{A}}_k = \mathbf{A}_k^{-1}. \tag{7.52}$$

As derivadas da matriz de ganhos $\mathbf{A}$ e sua inversa são definidas como

$$\dot{\mathbf{A}}_i = \frac{\partial \mathbf{A}}{\partial a_{ii}} = \begin{bmatrix} \mathbf{0} & & \\ & 1_{ii} & \\ & & \mathbf{0} \end{bmatrix} \tag{7.53}$$

,

$$\dot{\bar{\mathbf{A}}}_i = \frac{\partial \mathbf{A}^{-1}}{\partial a_{ii}} = \begin{bmatrix} \mathbf{0} & & \\ & -a_{ii}^{-2} & \\ & & \mathbf{0} \end{bmatrix} \tag{7.54}$$

e

$$\ddot{\bar{\mathbf{A}}}_i = \frac{\partial^2 \mathbf{A}^{-1}}{\partial a_{ii}^2} = \begin{bmatrix} \mathbf{0} & & \\ & 2a_{ii}^{-3} & \\ & & \mathbf{0} \end{bmatrix}, \tag{7.55}$$

onde somente o ii-ésimo elemento de cada matriz é diferente de zero.

O algoritmo completo de correção de ganho pelo método Tensorial-RLS pode, então, ser totalmente descrito como

$$\mathbf{z}_k = \left(\mathbf{y}_{k-1} - \mathbf{b}\right) \tag{7.56}$$

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \mathbf{A}_k\mathbf{M}_k\bar{\mathbf{A}}_k\mathbf{z}_k - \mathbf{b} \tag{7.57}$$

$$\forall i \tag{7.58}$$

$$\dot{\mathbf{A}}_i = \begin{bmatrix} \mathbf{0} & & \\ & 1_{ii} & \\ & & \mathbf{0} \end{bmatrix} \dot{\bar{\mathbf{A}}}_i = \begin{bmatrix} \mathbf{0} & & \\ & -a_{ii}^{-2} & \\ & & \mathbf{0} \end{bmatrix} \ddot{\bar{\mathbf{A}}}_i = \begin{bmatrix} \mathbf{0} & & \\ & 2a_{ii}^{-3} & \\ & & \mathbf{0} \end{bmatrix} \tag{7.59}$$

$$\mathbf{u}_{i,k} = \left[\dot{\mathbf{A}}_i\mathbf{M}_k\bar{\mathbf{A}}_k + \mathbf{A}_k\mathbf{M}_k\dot{\bar{\mathbf{A}}}_i\right]\mathbf{z}_k \tag{7.60}$$

$$v_{i,k} = \mathbf{z}_k^{\mathrm{T}}\left[2\dot{\mathbf{A}}_i\mathbf{M}_k\dot{\bar{\mathbf{A}}}_i + \mathbf{A}_k\mathbf{M}_k\ddot{\bar{\mathbf{A}}}_i\right]^{\mathrm{T}}\boldsymbol{\epsilon}_k + \mathbf{u}_{i,k}^{\mathrm{T}}\mathbf{u}_{i,k} \tag{7.61}$$

$$\gamma_{i,k} = \lambda\gamma_{i,k-1} + v_{i,k} \tag{7.62}$$

$$\hat{a}_{ii,k+1} = \hat{a}_{ii,k} - \gamma_{i,k}^{-1}\mathbf{u}_{i,k}^{\mathrm{T}}\boldsymbol{\epsilon}_k. \tag{7.63}$$

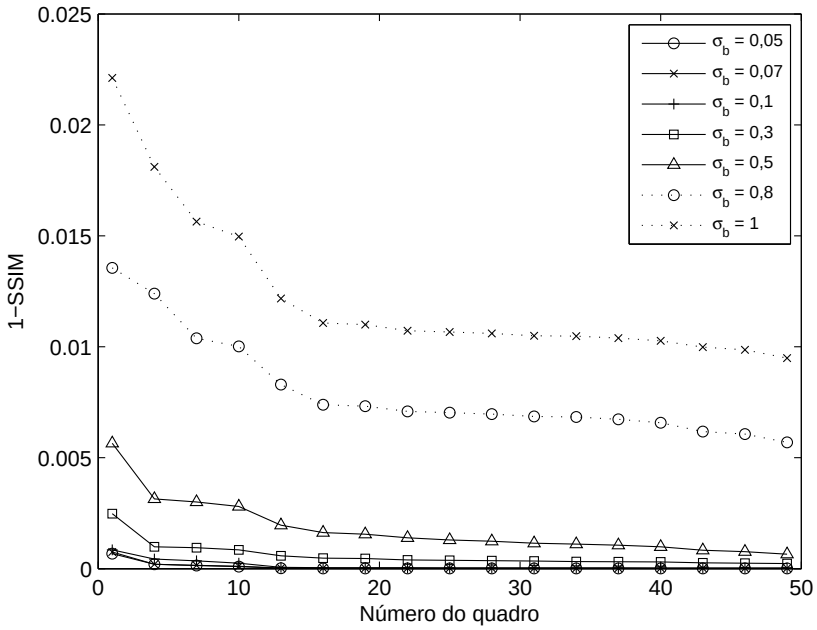

Figura 7.1: Robustez ao FPN (desvio) $\sigma_{\mathbf{A}} = 0,02$ $\sigma_{\mathbf{n}} = 0,0002$.

7.6 Resultados

Os resultados aqui apresentados dizem respeito à robustez do algoritmo Tensorial-RLS em termos de ruído FPN e ruído aleatório. Os resultados da aplicação em vídeos reais se encontram no Capítulo 8. Como o FPN nestas simulações é invariante no tempo, foi utilizado o fator de esquecimento $\lambda = 1$, ou seja, não há esquecimento e o algoritmo busca a melhor solução considerando todas as entradas.

As estimativas de desvio e ganho são atualizadas uma vez a cada novo par de quadros e são feitas na ordem desvio-ganho como definido na Subseção 6.5.1.

7.6.1 Robustez ao FPN

As Figuras 7.1 e 7.2 ilustram o comportamento do algoritmo para diferentes valores de ruído FPN.

Nota-se que mesmo para valores elevados de $\sigma_{\mathbf{A}}$ e $\sigma_{\mathbf{b}}$ não ocorre instabilidade.

7.6.2 Robustez ao ruído aleatório

A Figura 7.3 ilustra a resposta do algoritmo Tensorial-RLS para diferentes valores de $\sigma_{\mathbf{n}}$. Novamente o aumento do valor do ruído aleatório não acarretou instabilidade.

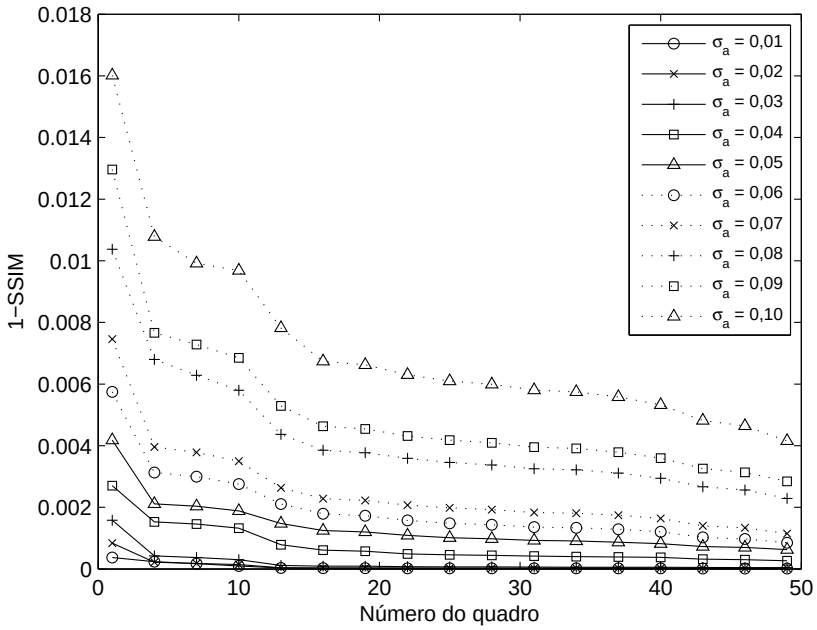

Figura 7.2: Robustez ao FPN (ganho) $\sigma_{\mathbf{b}} = 0,1$ $\sigma_{\mathbf{n}} = 0,0002$.

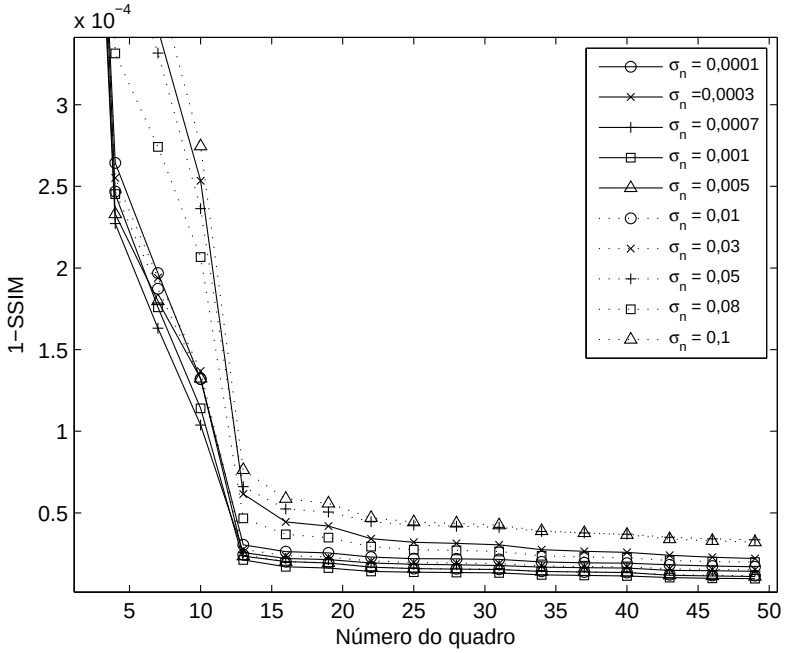

Figura 7.3: Robustez ao ruído aleatório $\sigma_{\mathbf{A}} = 0,02$ $\sigma_{\mathbf{b}} = 0,1$.

7.7 Conclusões

Este capítulo apresentou o desenvolvimento de um algoritmo para correção do desvio e ganho em vídeos contendo FPN batizado de Tensorial-RLS. Partiu-se do algoritmo Tensorial desenvolvido no Capítulo 6 e calculou-se a matriz Hessiana e o gradiente de segunda ordem para se obter a versão RLS.

Uma outra maneira de encarar essa solução seria chamá-la de LMS-Newton [28]. Os filtros adaptativos RLS e LMS-Newton têm versões cujas equações de atualização são idênticas. De fato, acrescentou-se nas equações de atualização uma estimativa da matriz Hessiana no caso da correção de desvio e uma estimativa do gradiente de segunda ordem no caso da correção do ganho. Preferiu-se manter o nome RLS para evidenciar que a solução visa a minimizar os últimos erros quadrados.

Já foi possível notar que a solução Tensorial-RLS apresentou maior robustez em relação aos ruídos FPN e aleatório. Tal resultado ficará mais evidente quando os algoritmos forem comparados no Capítulo 8.

Capítulo 8

Comparação dos métodos apresentados

8.1 Introdução

Este capítulo é dedicado a apresentar uma comparação entre os algoritmo propostos neste trabalho e um algoritmo considerado estado-da-arte na literatura. Serão apresentados os resultados médios de várias simulações e serão feitas discussões de cada caso com o intuito de ressaltar questões relevantes de cada abordagem.

8.2 Parâmetros das simulações e resultados

Os parâmetros escolhidos para cada um dos algoritmos foram aqueles que mostraram o melhor desempenho nas simulações individuais. Por exemplo, os passos de atualização μ foram escolhidos de forma a proporcionar uma boa velocidade de convergência dentro dos limites de estabilidade do algoritmo.

Supõe-se o estado-da-arte, ou seja, a referência de comparação, o algoritmo de correção de desvio desenvolvido em [21] baseado em filtro de Kalman. A escolha dessa referência se fez por sua recente publicação (2007) e por usar um método consagrado na literatura e tido como estimador linear ótimo que é o filtro de Kalman.

Serão aqui apresentados os resultados dos algoritmos sob condições mais severas de ruído, tanto FPN quanto ruído aleatório. Tal escolha foi feita de modo a acentuar as diferenças entre eles, já que em condições mais brandas de ruído as diferenças

Tabela 8.1: Parâmetros para as simulações de comparação.

Parâmetro	Valor
Número de vídeos	50
Quadros por vídeos	75
Resolução do vídeo	64×64
Imagens base (aleatória)	lena, militar ou tubulação
$\sigma_{\mathbf{d}}$ (movimento global)	1
$\sigma_{\mathbf{n}}$ (ruído aleatório)	$0,005$
$\sigma_{\mathbf{a}}$ (FPN ganho)	$0,004$
$\sigma_{\mathbf{b}}$ (FPN desvio)	$0,1$

Tabela 8.2: Comparação dos algoritmos.

Método	Parâmetros do algoritmo	(1-SSIM)10^{-3} médio
Desvio	$\mu_{\mathbf{b}} = 0,1$	$1,035$
Ganho	$\mu_a = 0,0975$	$1,046$
Averbuch/Kalman	$\mathbf{R} = \mathbf{I}$	$0,3681$
Desvio e Ganho	$\mu_{\mathbf{b}} = 0,1$ e $\mu_{\mathbf{a}} = 0,0975$	$0,8050$
Ganho e Desvio	$\mu_{\mathbf{b}} = 0,1$ e $\mu_{\mathbf{a}} = 0,0975$	$0,8075$
Tensorial	$\mu_{\mathbf{b}} = 0,1$ e $\mu_a = 0,001$	$0,7822$
Tensorial-RLS1	$\lambda_{\mathbf{b}} = 1$ e $\mu_a = 0,001$	$0,3362$
Tensorial-RLS2	$\lambda = 1$ (para desvio e ganho)	$0,2882$

encontradas não foram significativas. A Tabela 8.1 mostra as condições de geração dos vídeos sintéticos e outros características das simulações.

O algoritmo Tensorial-RLS foi testado em duas situações. Na primeira, quando ele é chamado de RLS1, somente o desvio é corrigido por RLS, ficando o ganho corrigido pelo método Tensorial tradicional. No caso do RLS2, ambos desvio e ganho são corrigidos pelo método RLS.

A Tabela 8.2 sintetiza os resultados encontrados para cada algoritmo, assim como os parâmetros utilizados em cada um deles. O valor de medida utilizado foi o (1-SSIM) médio, calculado para todos os quadros de todos os vídeos.

As Figuras 8.1 e 8.2 mostram a variação do erro de estimativa (1-SSIM) com

o número do quadro em uma visão geral e outra ampliada respectivamente. Foi escolhido o vídeo que apresentou os erros de estimativa mais próximos da média calculada na Tabela 8.2.

8.3 Discussões

Esta seção fará a discussão dos resultados obtidos, apontando os pontos relevantes e comentando os resultados levando em conta a abordagem e complexidade de cada algoritmo.

8.3.1 Correção do desvio por gradiente descendente

Pela Tabela 8.2 nota-se que o algoritmo proposto neste trabalho para correção de desvio obteve resultados inferiores aos algoritmos propostos em [21]. As vantagens do algoritmo proposto são a simplicidade de sua dedução e o rápido desempenho computacional.

Ao se comparar as equações de atualização em [21] com as propostas neste trabalho para correção de desvio, percebe-se grande semelhança. O algoritmo proposto em [21] é baseado em filtro de Kalman e, a não ser pela matriz de ganho de Kalman, é equivalente ao deste trabalho. De fato, conseguiu-se chegar a um algoritmo simplificado de [21] com melhor desempenho computacional e por uma dedução mais simples.

A título de ilustração, a equação de atualização usada em [21] é dada por

$$\hat{\mathbf{o}}_k = \hat{\mathbf{o}}_{k-1} + \mathbf{K}_k \left(\mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{o}}_{k-1} \right), \tag{8.1}$$

onde $\hat{\mathbf{o}}$ é a estimativa do desvio que equivale ao vetor $\hat{\mathbf{b}}$ deste trabalho, $\mathbf{K}_k = \alpha \mathbf{H}_k^{\mathrm{T}}$ é a matriz ganho de Kalman simplificada e α é uma constante multiplicativa, $\mathbf{z}_k = \mathbf{y}_k - \mathbf{M}_k \mathbf{y}_{k-1}$ e $\mathbf{H}_k = \mathbf{I} - \mathbf{M}_k$ já fazendo a ponte entre os dois trabalhos. Após alguma manipulação algébrica chega-se a

$$\hat{\mathbf{o}}_k = \hat{\mathbf{o}}_{k-1} + \alpha \left(\mathbf{I} - \mathbf{M}_k \right)^{\mathrm{T}} \cdot \tag{8.2}$$

$$\left(\mathbf{y}_k - \mathbf{M}_k \left(\mathbf{y}_{k-1} - \hat{\mathbf{o}}_{k-1} \right) - \hat{\mathbf{o}}_{k-1} \right), \tag{8.3}$$

que é exatamente a equação (4.24). Em suma, os dois algoritmos serão equivalentes quando o filtro de Kalman utilizado em [21] tiver a sua matriz ganho de Kalman

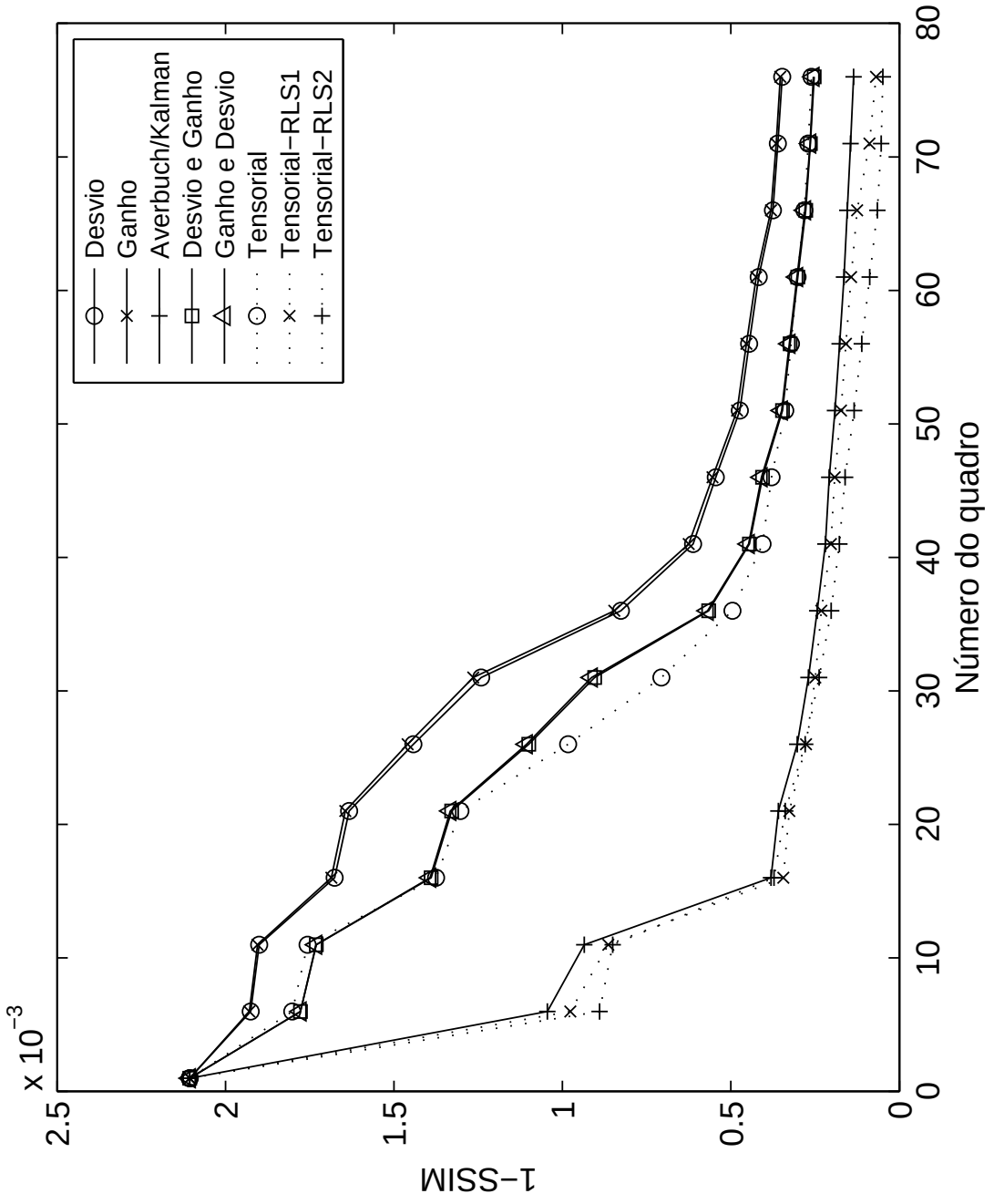

Figura 8.1: Comparação dos erros de estimativa dos algoritmos.

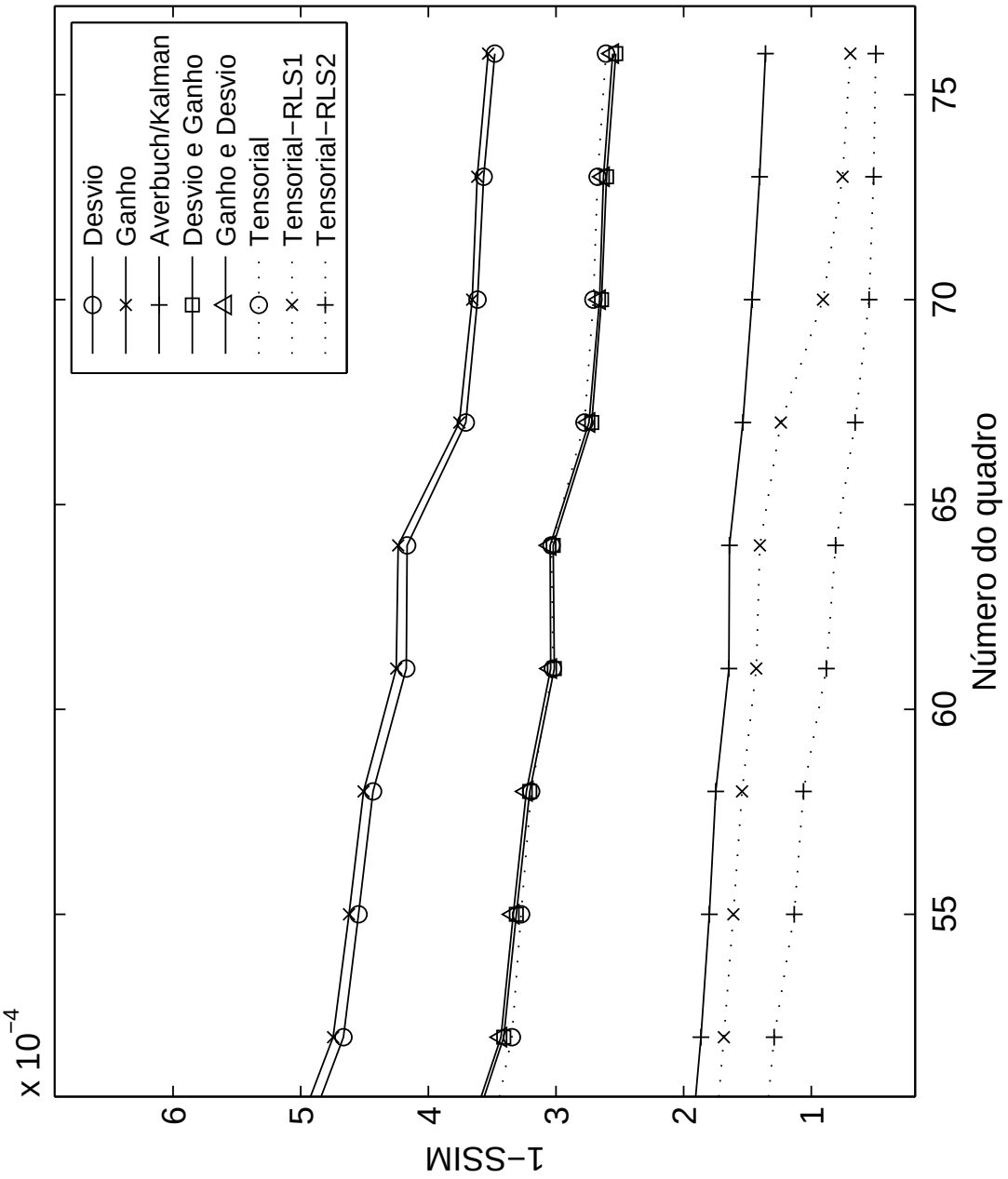

Figura 8.2: Comparação dos erros de estimativa dos algoritmos (gráfico ampliado).

simplificada para $\mathbf{K}_k = \alpha \mathbf{H}_k^{\mathrm{T}}$. A melhora de desempenho computacional obtida pelo algoritmo de correção de desvio proposto deve-se ao fato de não haver a necessidade de inversão de matrizes ou solução de sistema de equação lineares, como é o caso em [21].

8.3.2 Correção do ganho por gradiente descendente

O mesmo pode ser dito sobre o algoritmo de correção de ganho. Em vídeos que contêm parcelas aditivas e multiplicativas de FPN a correção somente do ganho mostrou-se a pior das alternativas. Uma interpretação possível é que foram gerados vídeos com menor teor de ruído multiplicativo que aditivo. Portanto, é razoável se esperar que um algoritmo que corrija somente desvio tenha um menor erro médio de estimativa que um algoritmo que corrige somente o ganho.

8.3.3 Correção de desvio e ganho por gradiente descendente

Nos casos de aplicação conjugada da correção de desvio e ganho percebeu-se uma melhora às correções separadas. A ordem das correções, ou seja, desvio antes que ganho ou ganho antes do desvio, teve um impacto irrelevante. Devido ao pequeno valor desta diferença pode-se se dizer que não é possível concluir qual das duas abordagens é superior ou se existe realmente alguma superioridade.

8.3.4 Método Tensorial

Considerando-se o algoritmo Tensorial percebe-se novamente um pequeno acréscimo na qualidade de correção do desvio e do ganho conjugados. Isso pode ser explicado pelo fato do algoritmo Tensorial considerar o modelo completo de FPN em toda sua dedução e nenhuma simplificação foi imposta, diferentemente dos casos de correção de desvio e ganho. Todavia, os algoritmos de correção de desvio, ganho, desvio e ganho conjugados e Tensorial, todos apresentaram erro médio maior que o algoritmo baseado em filtro de Kalman em [21].

É importante ressaltar que o algoritmo de [21] corrige somente o desvio, mas mesmo assim foi superior aos demais em vídeos com FPN aditivo e multiplicativo. Pôde-se perceber a superioridade do algoritmo baseado em filtro de Kalman em am-

biente com ruído aleatório considerável, superioridade esta em termos de velocidade de convergência, menor erro remanescente e menor erro médio de estimativa.

8.3.5 Métodos Tensorial-RLS

Os algoritmos Tensorial-RLS1 e Tensorial-RLS2 tiveram o melhor desempenho global de todos os algoritmos deste trabalho. O algoritmo Tensorial-RLS1 aplica a filosofia RLS somente à correção do desvio, sendo a correção do ganho feita por gradiente descendente. A adoção do modelo RLS conferiu a esses algoritmos rapidez de convergência e estabilidade a ruído aleatório. De fato, o algoritmo Tensorial-RLS1 sem correção de ganho teve desempenho idêntico ao algoritmo por filtro de Kalman em [21]. Isso comprova o elo entre o algoritmo RLS e Kalman como é sabido na literatura e como descrito em [28]. A vantagem do Tensorial-RLS1 é a correção do ganho que não é feita no algoritmo de [21].

Finalmente o algoritmo Tensorial-RLS2 aplica a filosofia RLS à correção do desvio e do ganho. Com isso foi possível obter estabilidade a ruído aleatório, velocidade de convergência e erro médio de estimativa melhores que os demais algoritmos apresentados. Tal superioridade foi conseguida às custas de uma dedução e a um algoritmo mais complexos.

8.4 Ensaios em vídeos reais

Foram realizadas filmagens utilizando uma câmera infravermelha marca FLIR SYSTEMS modelo ThermaCAM P65. Seu detector é do tipo *focal plane array uncooled microbolometer*. A resolução utilizada foi 320×240 pixels e foram gravados 200 quadros a uma taxa de 60Hz. Para melhor visualização da correção do FPN em cada pixel, será mostrada a sua aplicação à região central de cada vídeo num tamanho de 100×100.

Foi desativada a opção de "Redução do Ruído (*Noise Reduction Off*)" assim como a opção "Período obturação (*Shutter period Off*)". Esta última diz respeito a correção de FPN realizada pela câmera. Quando esta opção esta ativa, a câmera fecha o obturador em períodos de tempo pré-determinados (de 3 minutos a 15 minutos) para efetuar uma calibração como descrito na Subseção 3.1.1.

A primeira cena filmada constituía-se de duas seções de tubos de fibra de vidro com defeitos internos introduzidos artificialmente. Atrás dos tubos foi colocada uma fonte de calor, simulando um fluído interno aquecido durante um ensaio não-destrutivo termográfico. Esse tipo de ensaio objetiva detectar reduções de espessura na parede do duto através de diferenças na temperatura da superfície. A Figura 8.3 ilustra um quadro deste vídeo.

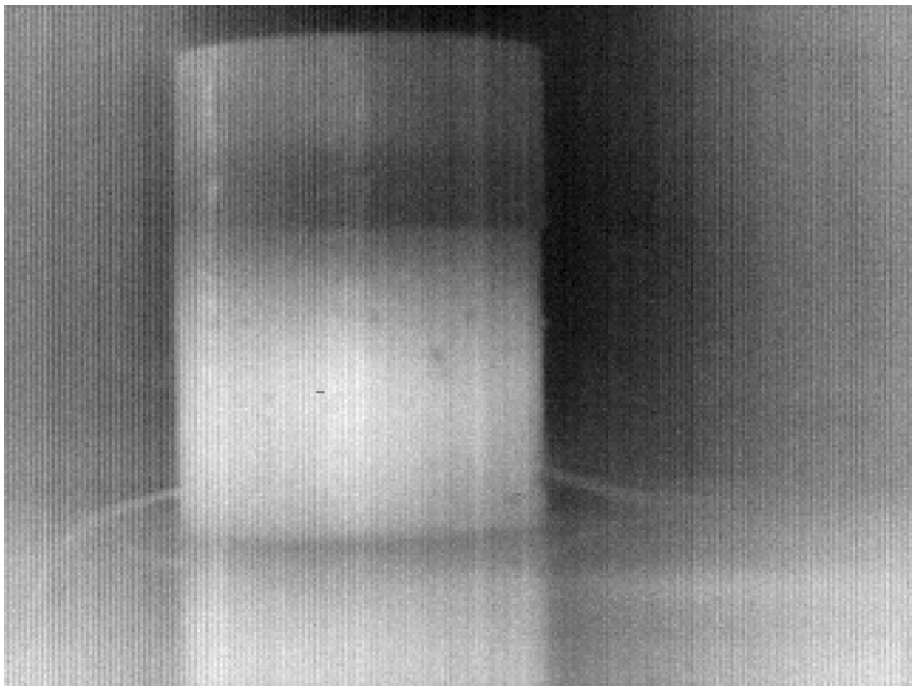

Figura 8.3: Exemplo de vídeo de tubos de fibra de vidro.

A segunda cena foi constituída de um telefone fixado a uma parede. Por todo o telefone estar aproximadamente na mesma temperatura, a diferença da radiação infravermelha deve-se principalmente a variação na emissividade dos materiais componentes do aparelho. A Figura 8.4 ilustra um quadro deste vídeo.

A Figura 8.5 mostra a medida (1-SSIM) aplicada entre o vídeo original dos tubos de fibra de vidro e as saídas dos algoritmos de correção de FPN. Lembra-se que quanto mais próximo de zero é a medida (1-SSIM) mais parecidas são as imagens originais e corrigidas. É importante notar que não se pode concluir nada a respeito da qualidade da correção do FPN a partir desse gráfico. A sua função aqui é indicar o quanto cada algoritmo alterou as imagens originais, independente da melhora de qualidade de imagem em termos de remoção do FPN.

Nos vídeos dos tubos, escolheu-se o quadro 197 por este apresentar o maior valor

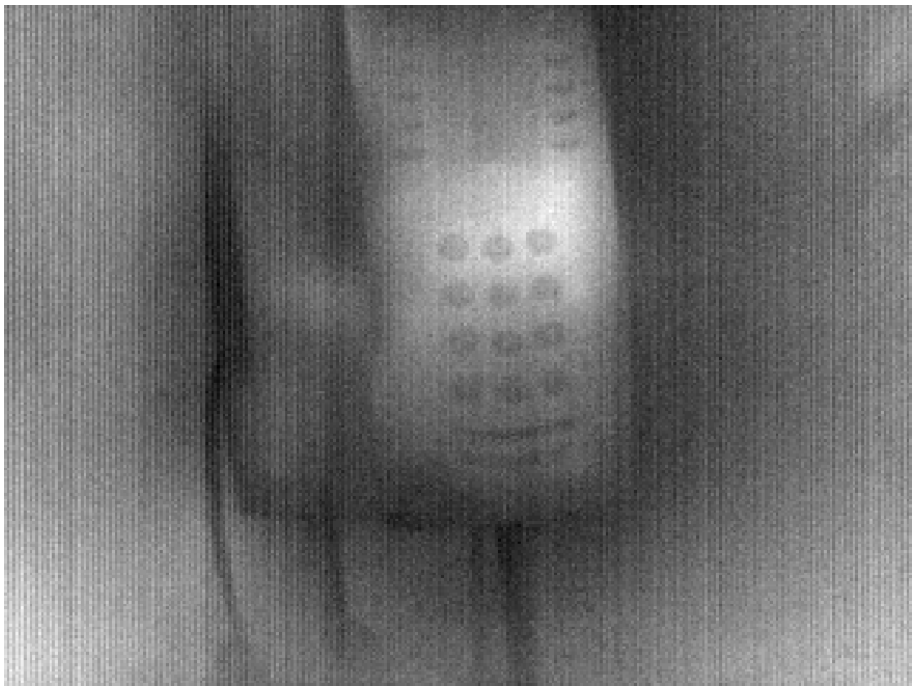

Figura 8.4: Exemplo de vídeo de telefone.

de (1-SSIM), ou seja, a maior diferença entre a imagem original com FPN e as imagens corrigidas. Os resultados são apresentados em pares nas Figuras 8.6, 8.7, 8.8, 8.9, 8.10 e 8.11.

A Figura 8.12 apresenta a métrica (1-SSIM) calculada após a aplicação das correções no vídeo do telefone. Escolheu-se o quadro número 100 para mostrar os resultados, ilustrados nas Figuras 8.13, 8.14, 8.15, 8.16, 8.17 e 8.18.

As Figuras 8.21 e 8.22 mostram respectivamente o quadro 190 do vídeo inteiro dos tubos após correção Averbuch/Kalman e Tensorial-RLS2 com $\lambda = 0,9$.

As Figuras 8.26 e 8.26 mostram respectivamente o quadro 133 do vídeo inteiro do telefone após correção Averbuch/Kalman e Tensorial-RLS2 com $\lambda = 0,9$.

As Figuras 8.31 e 8.31 mostram respectivamente o quadro 172 do vídeo inteiro do telefone após correção Averbuch/Kalman e Tensorial-RLS2 com $\lambda = 0,9$.

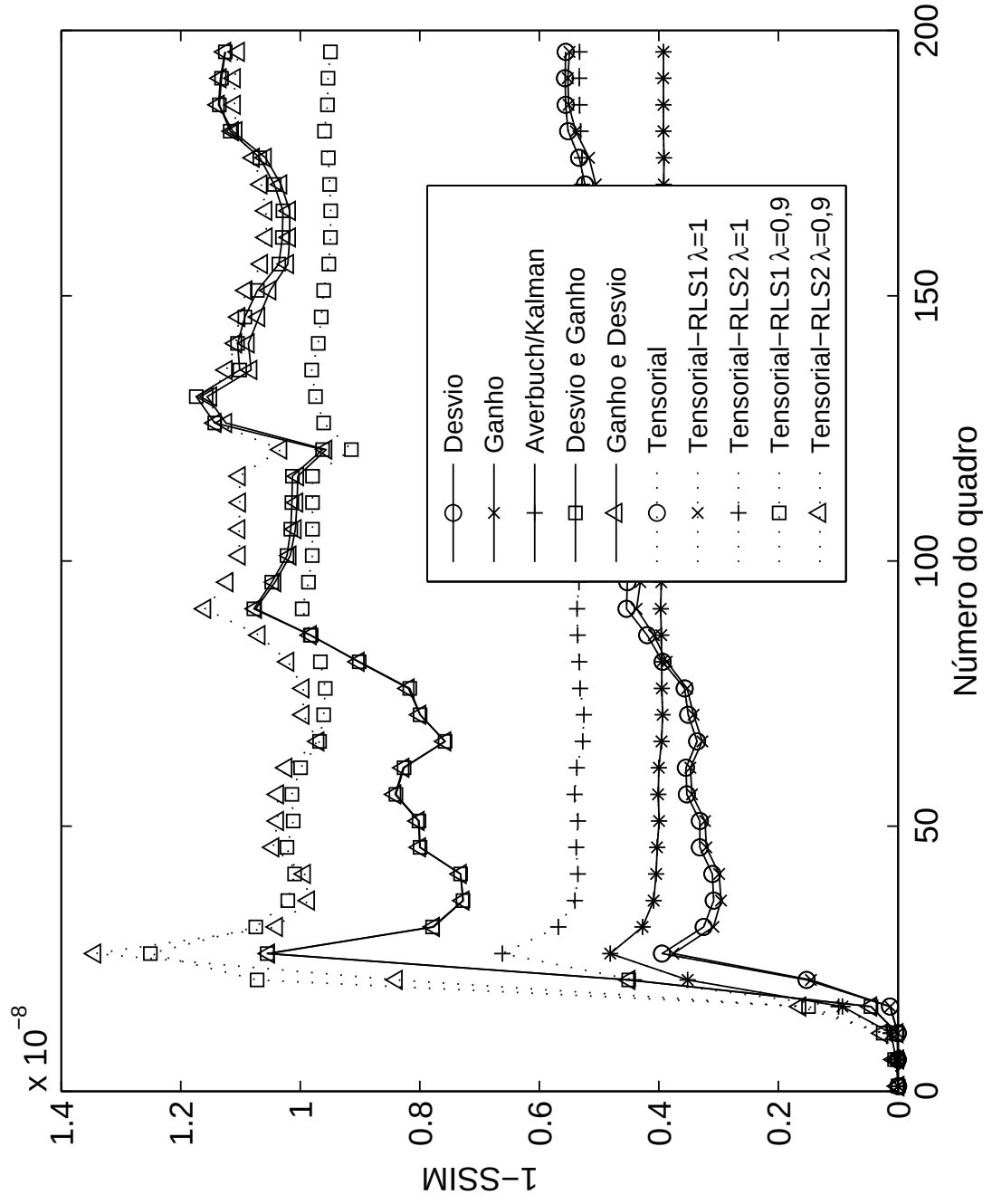

Figura 8.5: Comparação das diferenças em vídeos reais (vídeo de tubos).

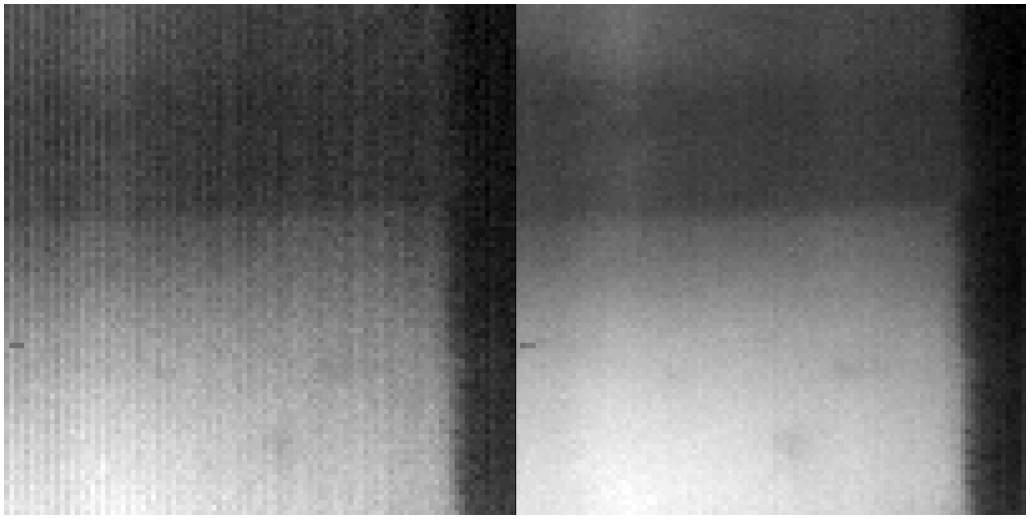

Figura 8.6: Vídeo de tubos: Original e Desvio.

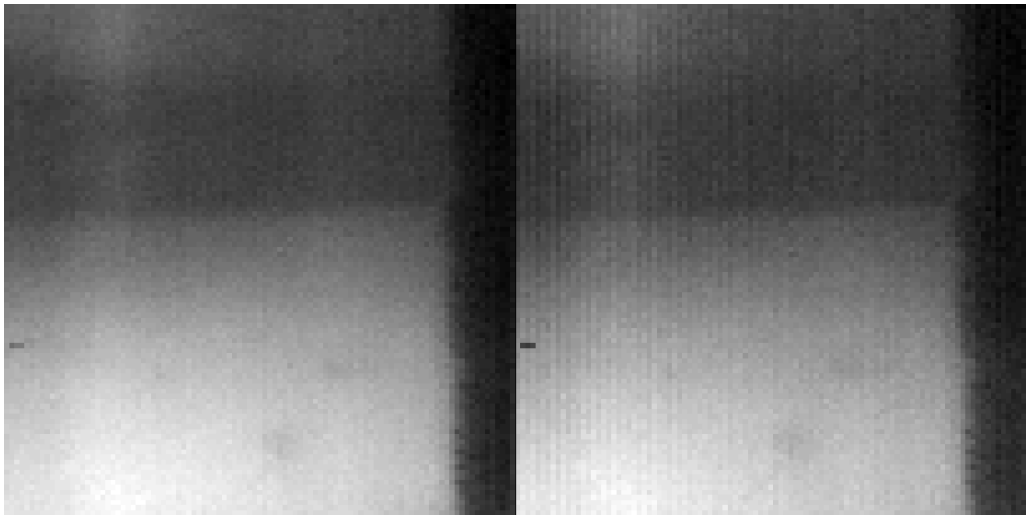

Figura 8.7: Vídeo de tubos: Ganho e Averbuch/Kalman.

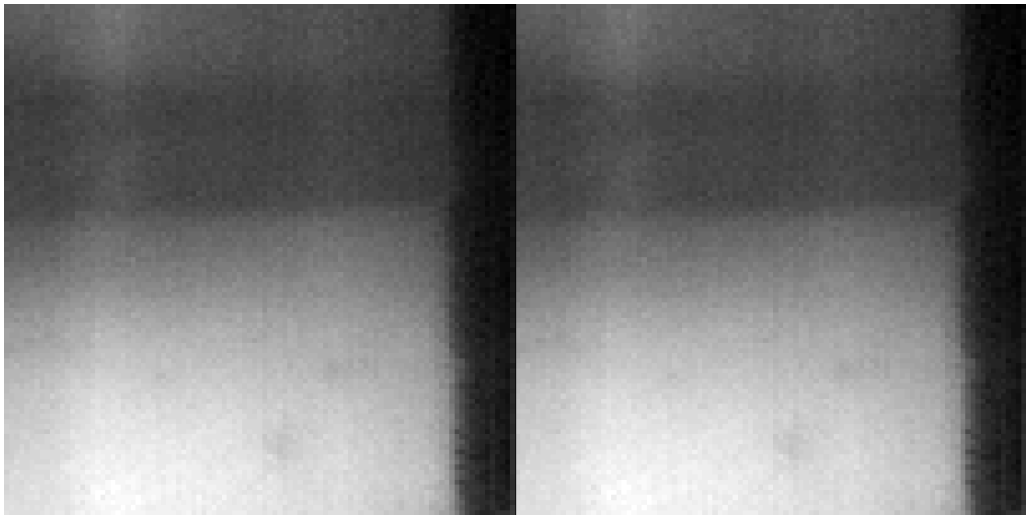

Figura 8.8: Vídeo de tubos: Desvio e Ganho, Ganho e Desvio.

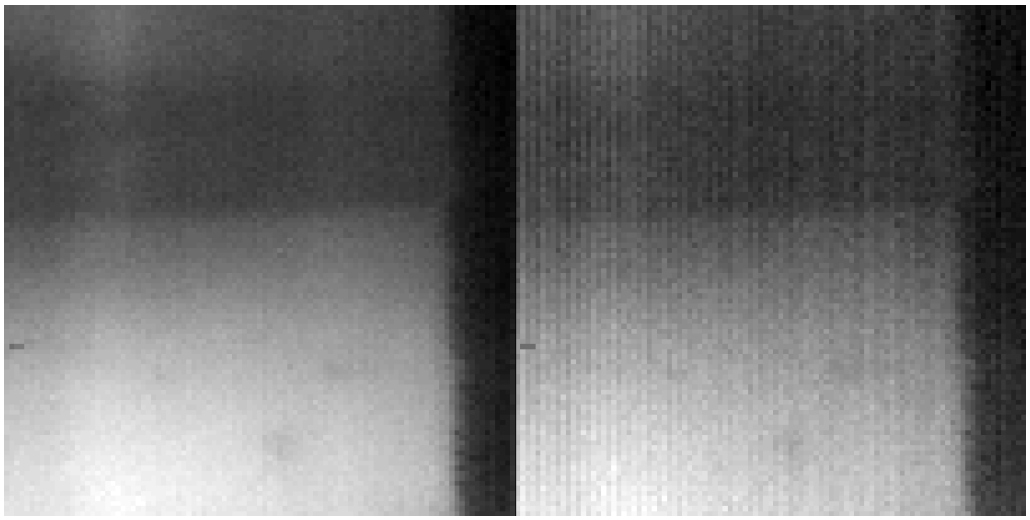

Figura 8.9: Vídeo de tubos: Tensorial, Original.

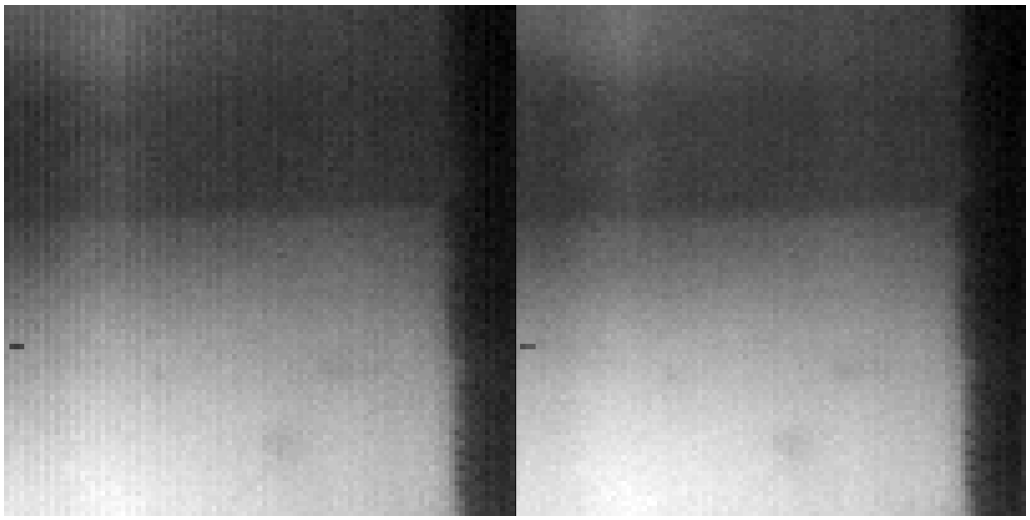

Figura 8.10: Vídeo de tubos: Tensorial-RLS1 $\lambda = 1$, Tensorial-RLS1 $\lambda = 0,9$.

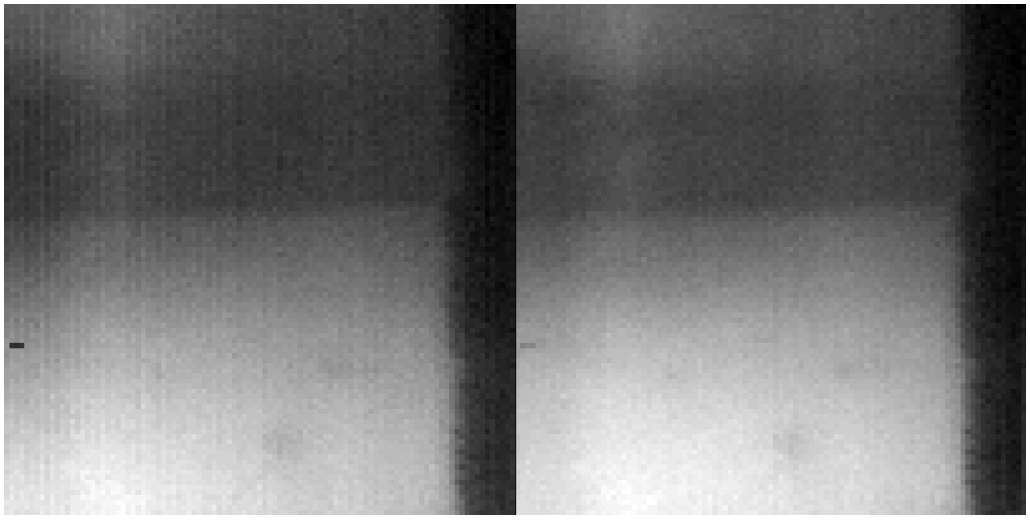

Figura 8.11: Vídeo de tubos: Tensorial-RLS2 $\lambda = 1$, Tensorial-RLS2 $\lambda = 0,9$.

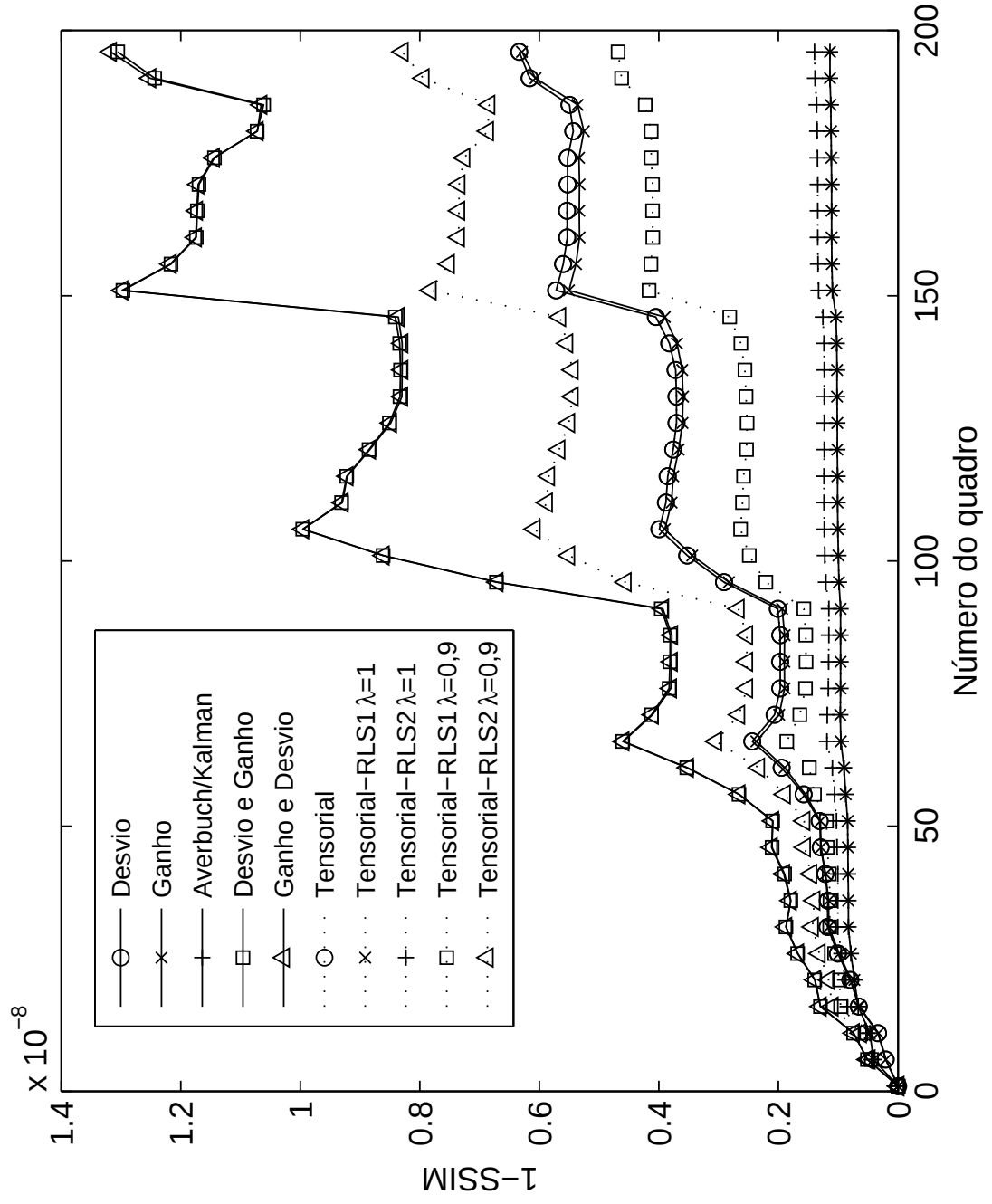

Figura 8.12: Comparação das diferenças em vídeos reais (vídeo de telefone).

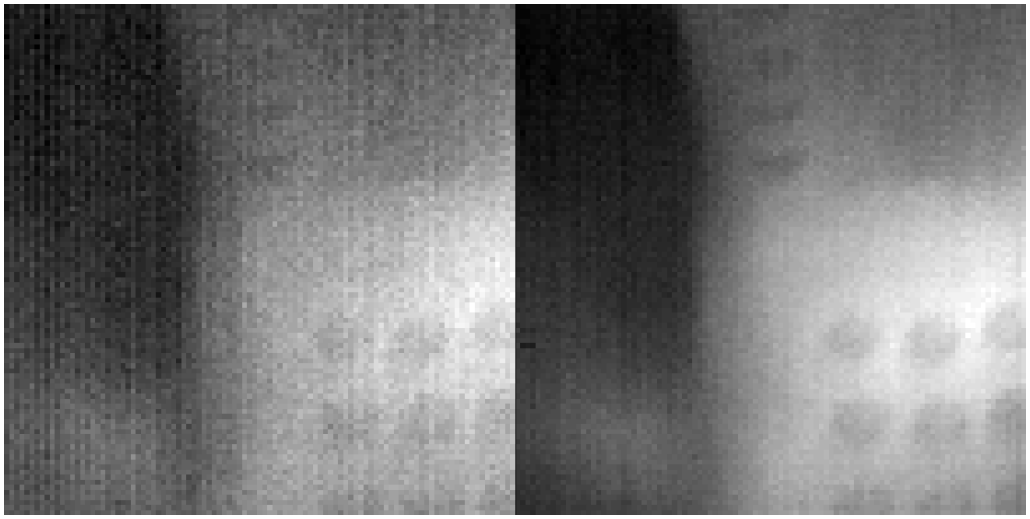

Figura 8.13: Vídeo de telefone: Original e Desvio.

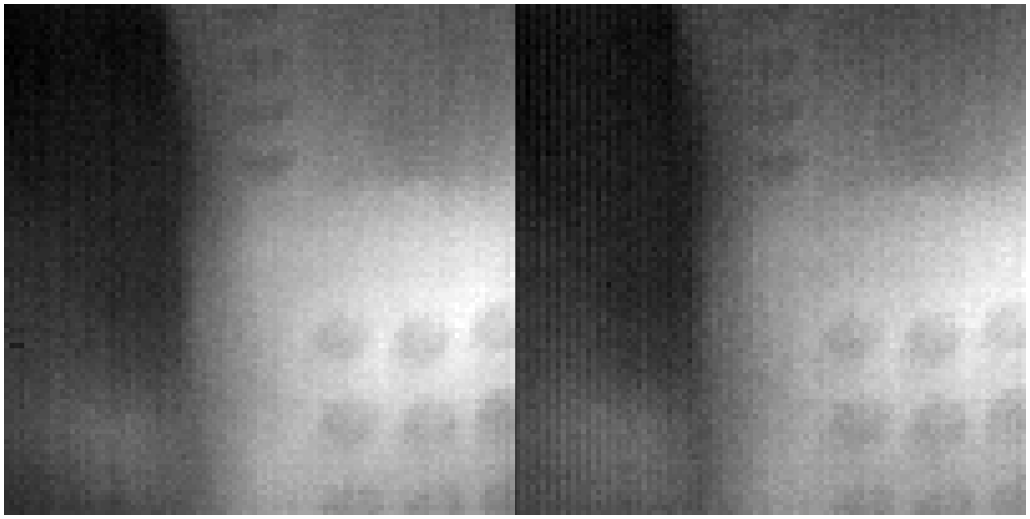

Figura 8.14: Vídeo de telefone: Ganho e Averbuch/Kalman.

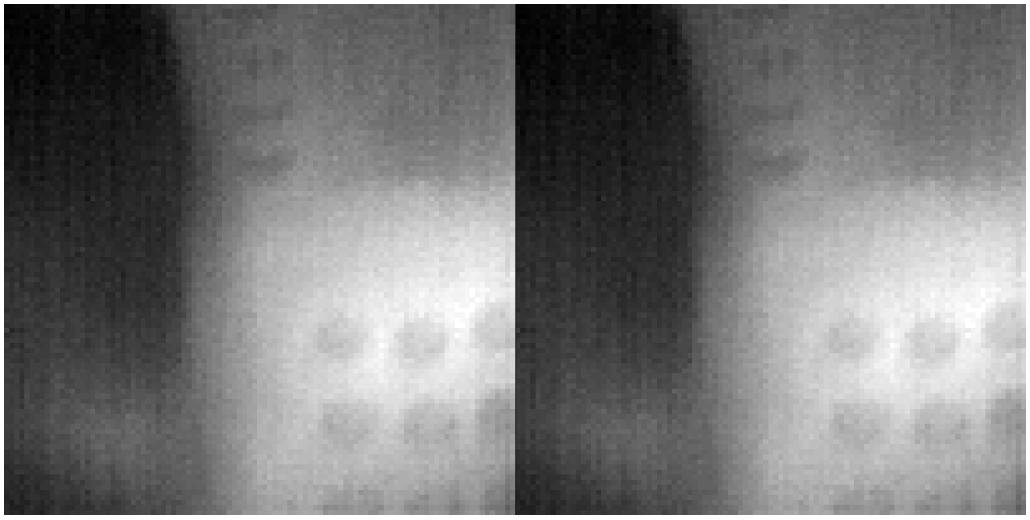

Figura 8.15: Vídeo de telefone: Desvio e Ganho, Ganho e Desvio.

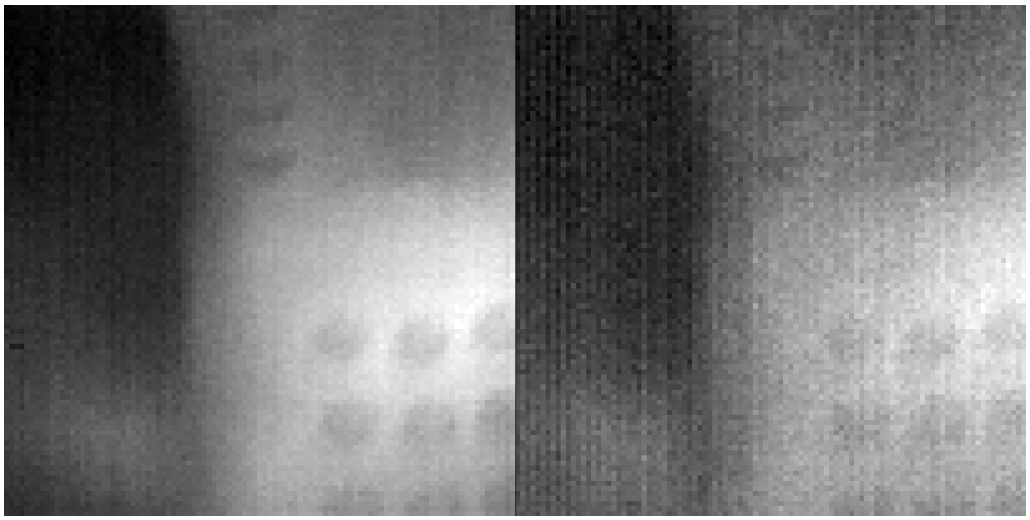

Figura 8.16: Vídeo de telefone: Tensorial, Original.

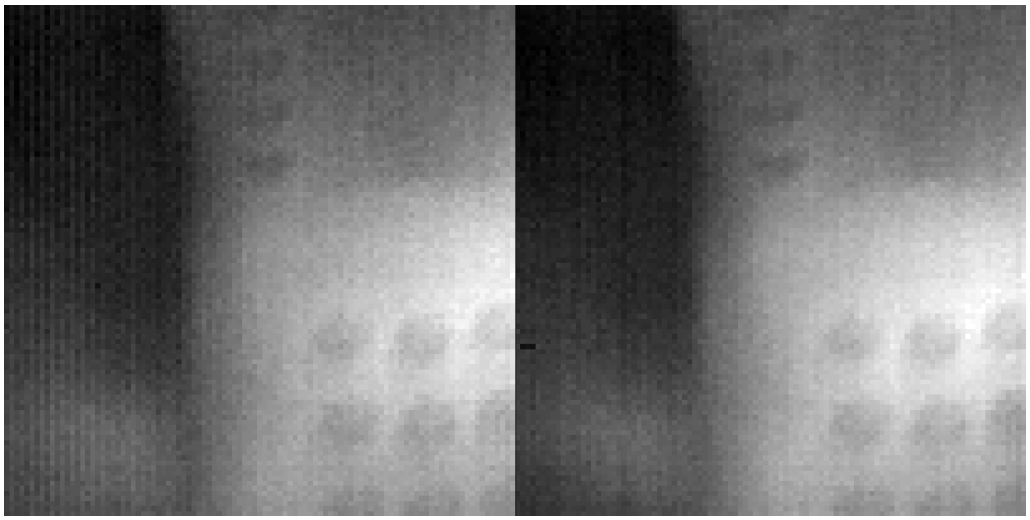

Figura 8.17: Vídeo de telefone: Tensorial-RLS1 $\lambda = 1$, Tensorial-RLS1 $\lambda = 0,9$.

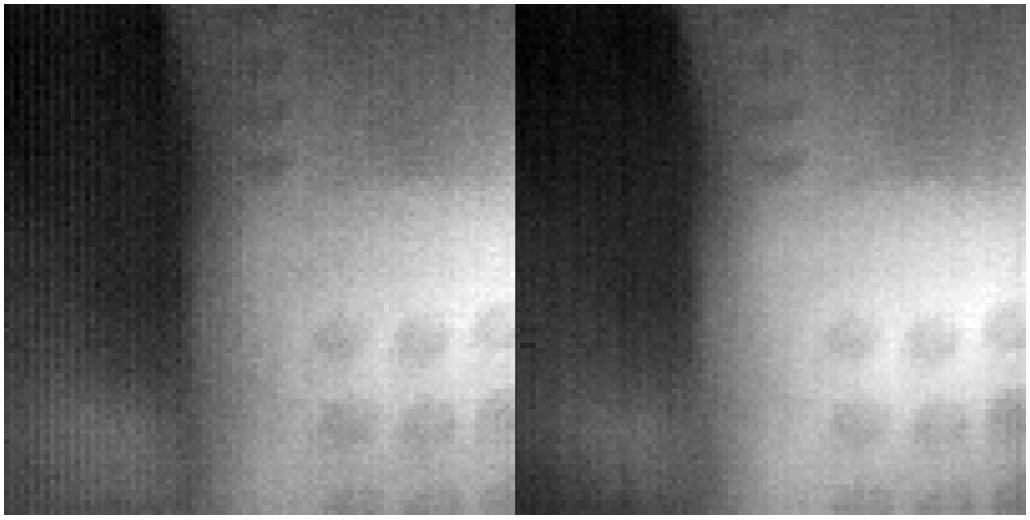

Figura 8.18: Vídeo de telefone: Tensorial-RLS2 $\lambda = 1$, Tensorial-RLS2 $\lambda = 0,9$.

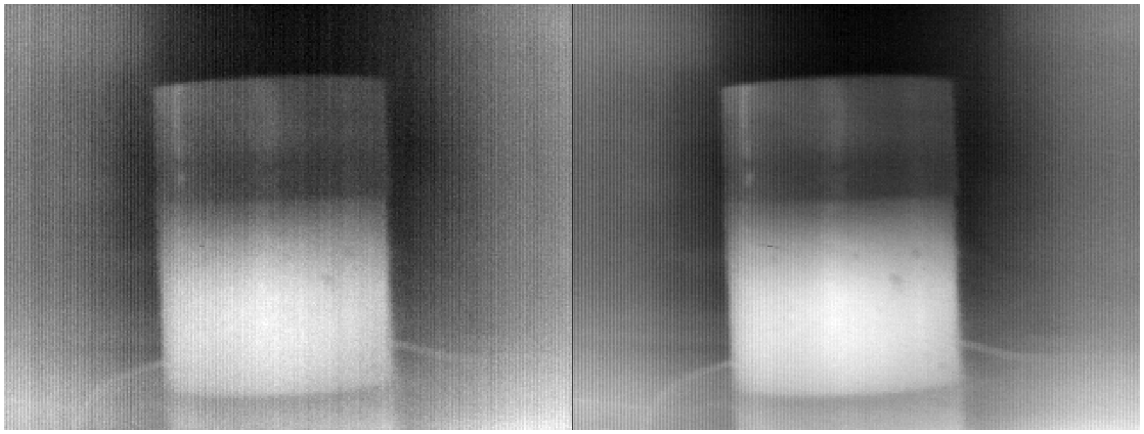

Figura 8.19: Vídeo inteiro dos tubos quadro 190 após correção do desvio.

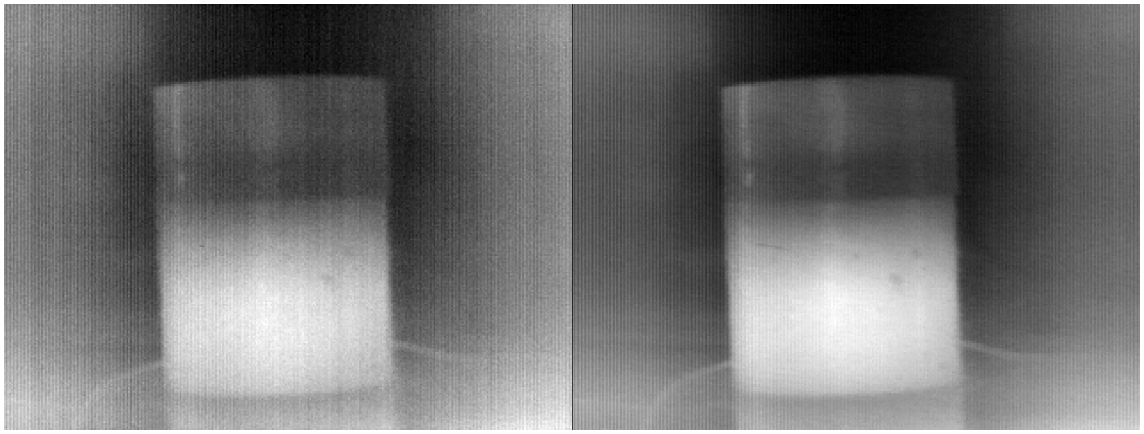

Figura 8.20: Vídeo inteiro dos tubos quadro 190 após correção de desvio e ganho.

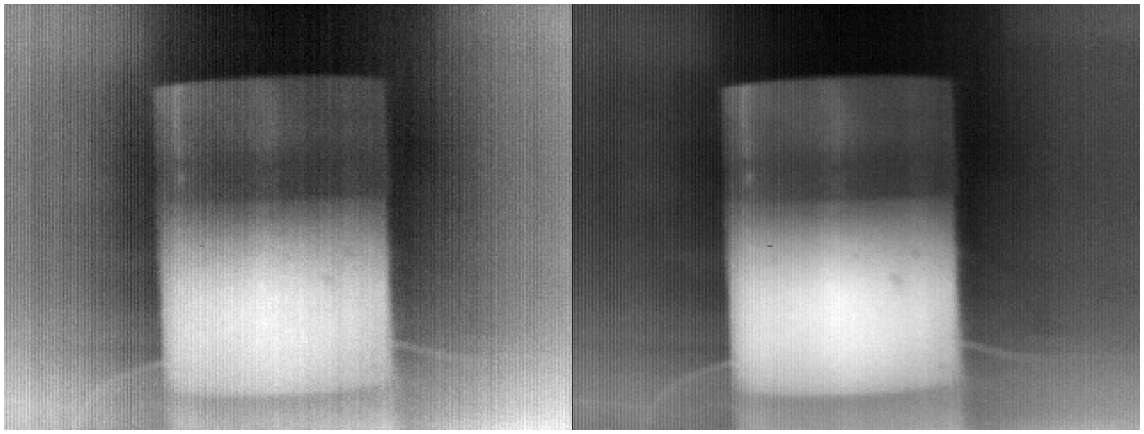

Figura 8.21: Vídeo inteiro dos tubos quadro 190 após correção Averbuch/Kalman.

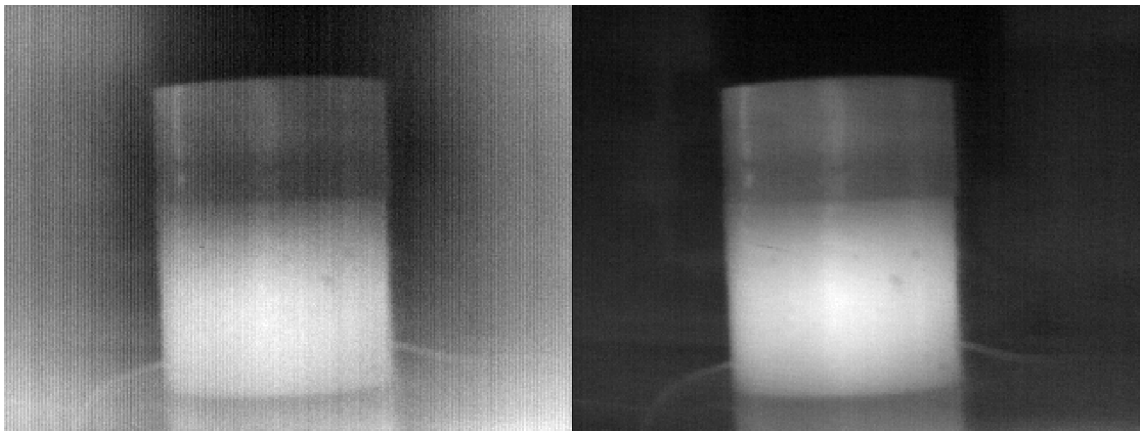

Figura 8.22: Vídeo inteiro dos tubos quadro 190 após correção Tensorial-RLS2 $\lambda = 0,9$.

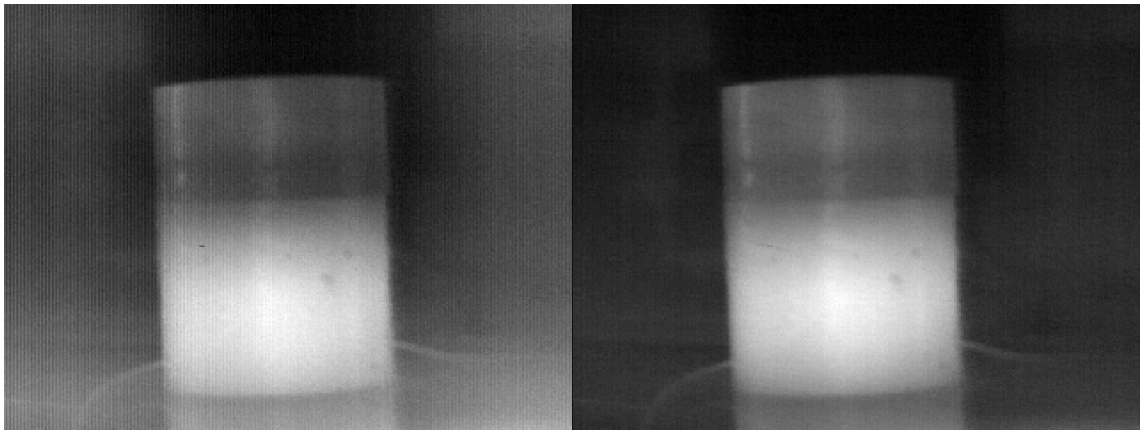

Figura 8.23: Vídeo inteiro dos tubos quadro 190 após correção Averbuch/Kalman (esquerda) e Tensorial-RLS2 $\lambda = 0,9$ (direita).

8.4.1 Discussões

Já com a correção somente do desvio, percebe-se uma melhora na qualidade dos vídeos dos tubos e do telefone. Nota-se que a granulação na imagem diminui e que é possível perceber duas manchas na parte inferior direita dos tubos (defeitos artificiais) e no caso do vídeo do telefone, é possível distinguir com melhor nitidez as teclas do aparelho.

Comparando a correção do desvio com a correção do ganho, não se percebe grande diferença entre as duas. Porém, ao reparar a correção pelo método de Averbuch/Kalman é visível uma qualidade inferior na correção do FPN (Figu-

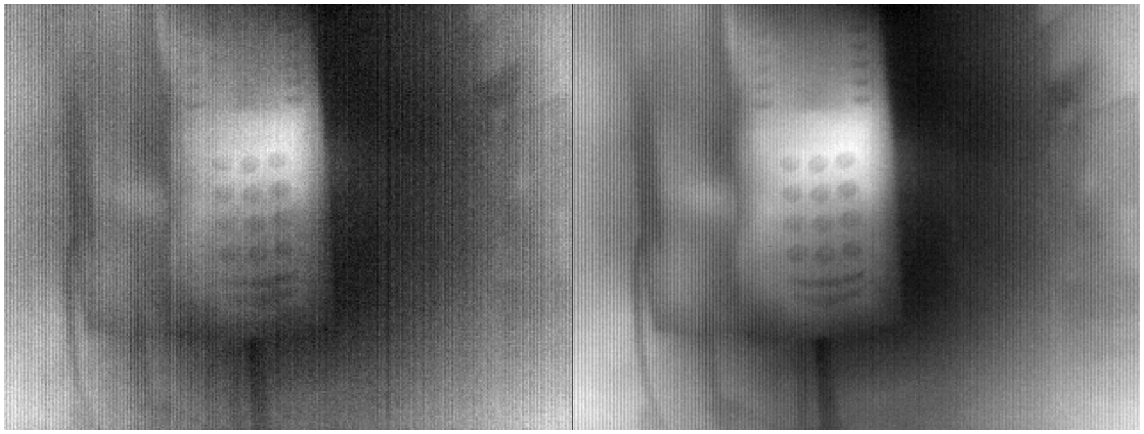

Figura 8.24: Vídeo inteiro do telefone quadro 133 após correção do desvio.

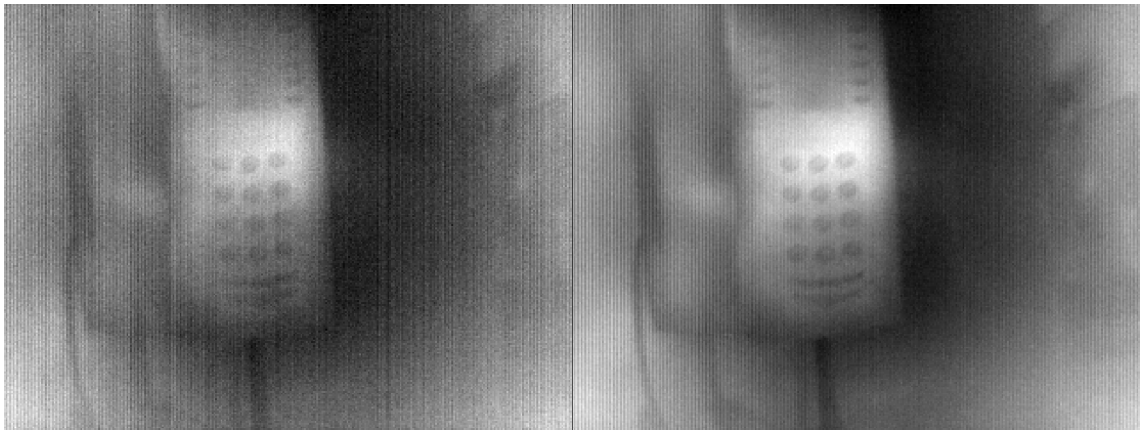

Figura 8.25: Vídeo inteiro do telefone quadro 133 após correção de desvio e ganho.

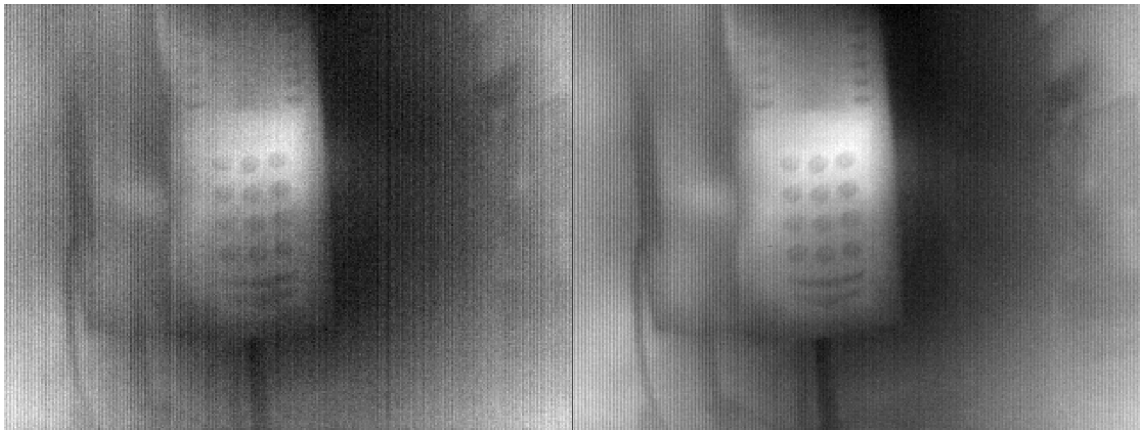

Figura 8.26: Vídeo inteiro do telefone quadro 133 após correção Averbuch/Kalman.

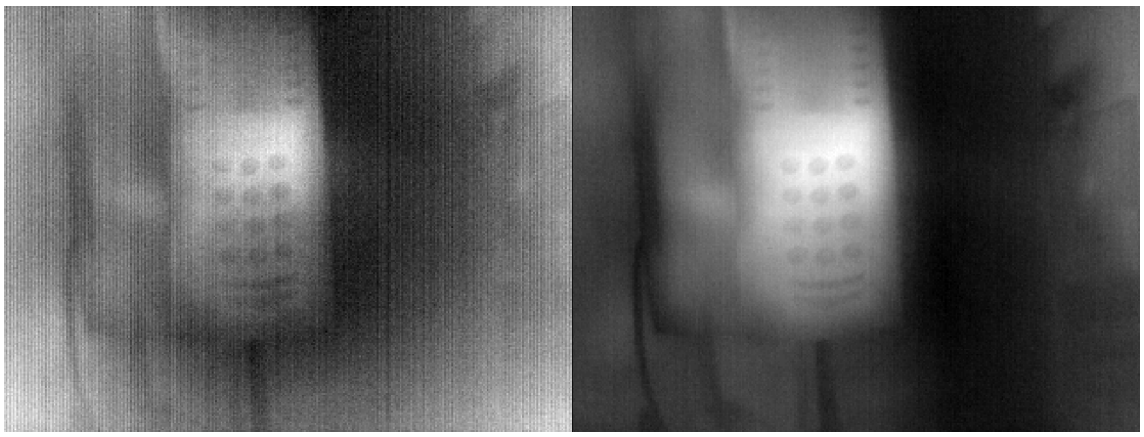

Figura 8.27: Vídeo inteiro do telefone quadro 133 após correção Tensorial-RLS2 $\lambda = 0,9$.

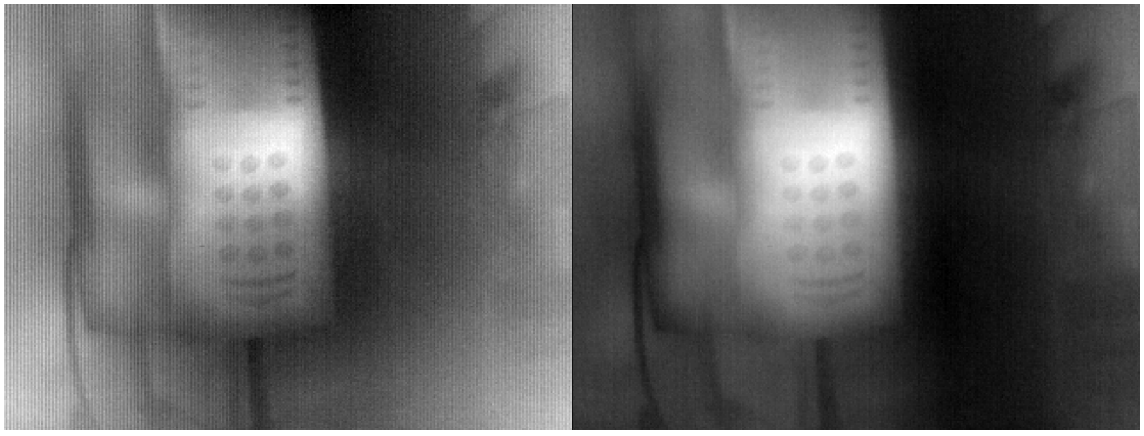

Figura 8.28: Vídeo inteiro do telefone quadro 133 após correção Averbuch/Kalman (esquerda) e Tensorial-RLS2 $\lambda = 0,9$ (direita).

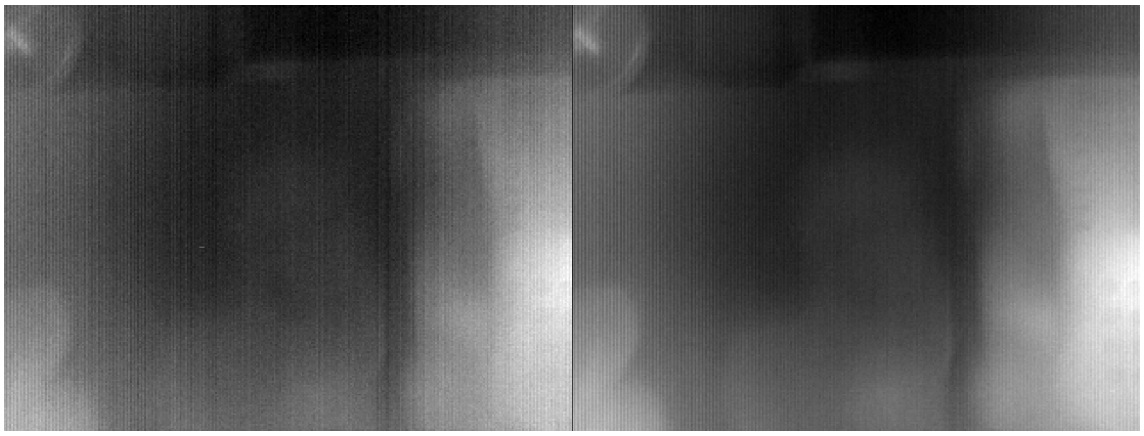

Figura 8.29: Vídeo inteiro do telefone quadro 172 após correção do desvio.

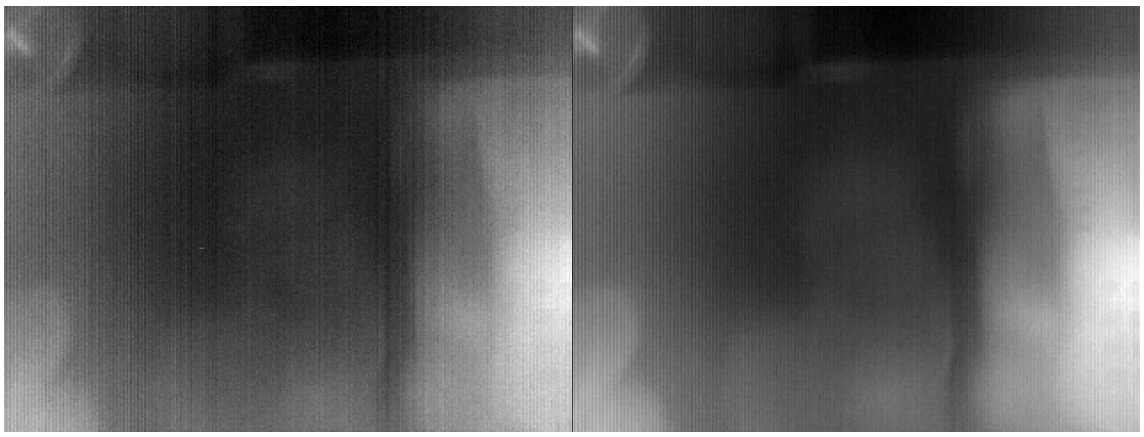

Figura 8.30: Vídeo inteiro do telefone quadro 172 após correção de desvio e ganho.

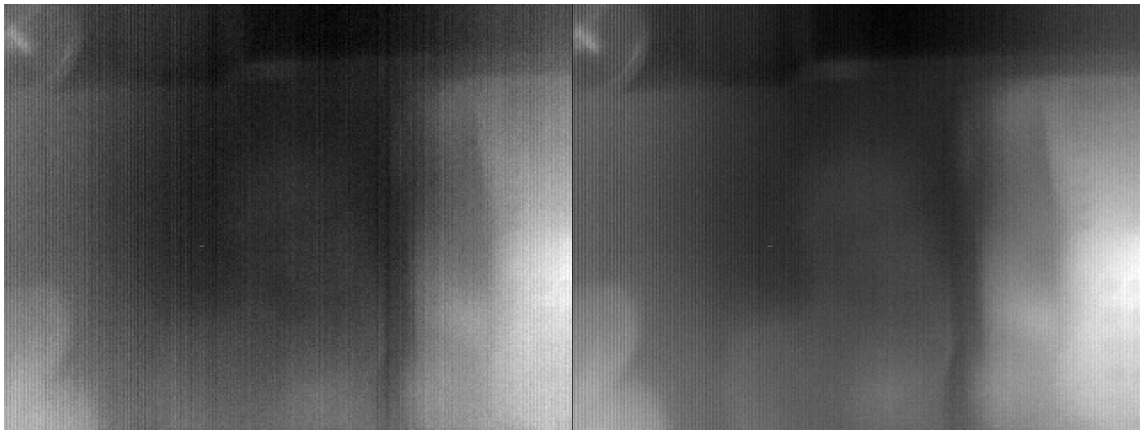

Figura 8.31: Vídeo inteiro do telefone quadro 172 após correção Averbuch/Kalman.

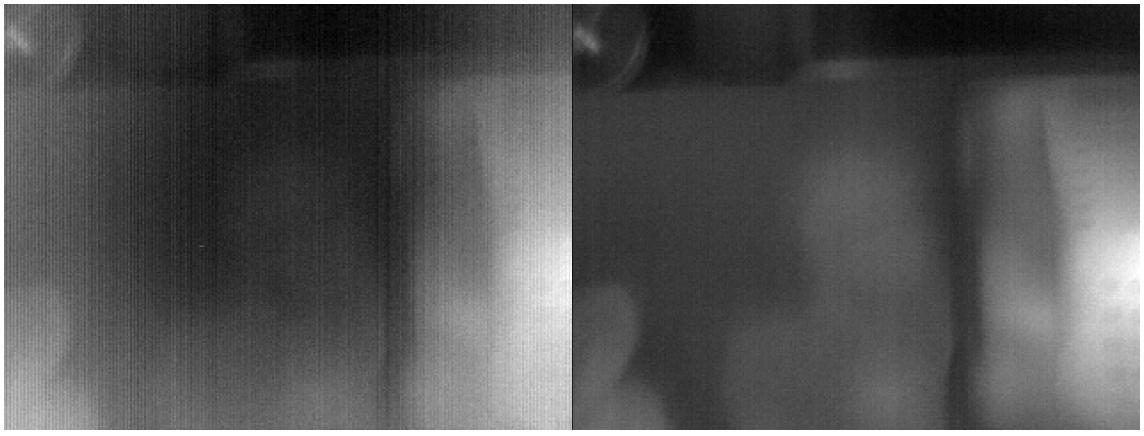

Figura 8.32: Vídeo inteiro do telefone quadro 172 após correção Tensorial-RLS2 $\lambda = 0,9$.

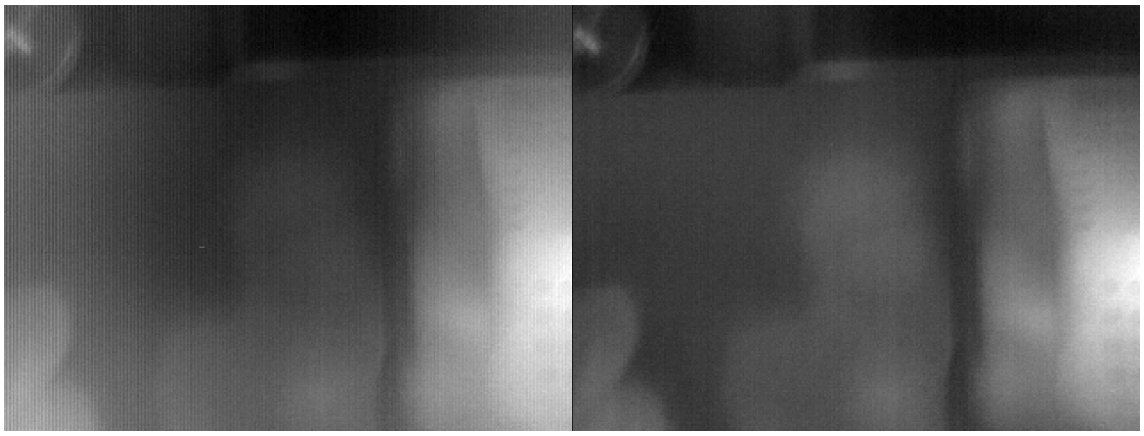

Figura 8.33: Vídeo inteiro do telefone quadro 172 após correção Averbuch/Kalman (esquerda) e Tensorial-RLS2 $\lambda = 0,9$ (direita).

ras 8.7 e 8.14). Ao se reparar nos gráficos das Figuras 8.5 e 8.12 vê-se claramente que o método Averbuch/Kalman alterou pouco as imagens ao longo do vídeo. Tal comportamento se explica pelo fato deste algoritmo supor que o FPN é invariante no tempo e funcionar bem caso isso seja verdade, como foi comprovado nas simulações.

No caso das correções conjugadas desvio/ganho e ganho/desvio é possível notar uma sutil melhora em relação à correção única ou do desvio ou do ganho. Apesar de não ser uma medida de qualidade de correção de FPN, os gráficos das Figuras 8.5 e 8.12 confirmam que o algoritmo provocou mais alteração nas imagens.

As diferenças dos resultados obtidos com o algoritmo Tensorial com os demais é novamente muito sutil. Nota-se uma pequena superioridade em relação a correção somente do desvio ou ganho, porém pode-se considerar que seu resultado é inferior às correções conjugadas, sobretudo pois estas foram capazes de compensar três pixels no lado esquerdo das imagens que demonstraram um comportamento anormal.

Finalmente percebe-se que o algoritmo Tensorial-RLS2 é levemente superior ao Tensorial-RLS1. Tal comportamento já era esperado baseado na estrutura do algoritmo e nos resultados obtidos nas simulações. Contudo, foi constatado nas Figuras 8.10, 8.11, 8.17 e 8.18 que foi obtido uma melhor correção de FPN com o fator de esquecimento mais baixo, ou seja, $\lambda = 0,9$. Tal resultado leva a crer que o FPN introduzido pela câmera estudada apresenta uma variação temporal não desprezível.

Porém, se o modelo de FPN adotado neste trabalho tiver deficiências em representar o FPN real gerado pela câmera que, por exemplo, pode ter uma resposta

não linear de segundo ou maior grau, a seguinte interpretação alternativa é possível. Um algoritmo cujos parâmetros foram ajustados permitindo a minimização do(s) último(s) erro(s) (caso dos algoritmo baseado em gradiente descendente ou com fator de esquecimento pequeno, p.ex. $\lambda = 0,9$) faria uma espécie de linearização por partes.

Em outras palavras, um FPN não linear poderia ser representado por vários modelos lineares, cada um com a sua faixa de trabalho ou atuação. Quando o algoritmo de correção de FPN fica "preso" a um desses modelos (caso de $\lambda = 1$ ou mesmo no caso do algoritmo Averbuch/Kalman) o resultado tende a piorar se os pixels caminharem para outro ponto de operação ou outro sub-modelo linear. Por outro lado, se certa flexibilidade é permitida, os algoritmos podem transitar entre os sub-modelos e oferecer uma melhor correção do FPN.

As Figuras 8.19, 8.20, 8.21, 8.22, 8.24, 8.25, 8.26, 8.27, 8.29, 8.30, 8.31 e 8.32 mostram quadros originais ao lado de correções com os métodos Correção de desvio, Correção de desvio-ganho, Averbuch/Kalman e Tensorial-RLS2 com $\lambda = 0,9$. Nota-se que todos os métodos, porém em destaque o Tensorial-RLS2 proposto neste trabalho, suavizaram consideravelmente o ruído FPN das imagens, deixando-as mais nítidas e possibilitando a percepção de detalhes ocultos nas imagens originais. Finalmente, as Figuras 8.23, 8.28 e 8.33 mostram lado a lado as mesmas imagens após correção Averbuch/Kalman e Tensorial-RLS2 com $\lambda = 0,9$.

8.5 Conclusões

Os resultados aparentemente controversos entre as simulações e os vídeos reais se devem provavelmente à:

- Variação temporal do desvio e/ou ganho mais rápida que o esperado pelo modelo;
- FPN com componentes não lineares significativas.

Pode-se se afirmar que o modelo de FPN utilizado não representa com exatidão o FPN introduzido no vídeo pela câmera utilizada nos ensaios. Não obstante, todos os algoritmos estudados apresentaram alguma melhora na qualidade de imagem no que diz respeito ao FPN.

Capítulo 9

Conclusões

Este trabalho apresentou o desenvolvimento de quatro novos algoritmos recursivos para correção de ruído de padrão fixo em vídeos infravermelhos. Todos os algoritmos consideram os modelos de observação de vídeo infravermelho sugeridos na literatura para o tipo de sensor IRFPA, o mais comum em câmeras infravermelhas modernas.

O Capítulo 4 apresentou o desenvolvimento de um algoritmo para correção do desvio. Como mencionado em [12], o parâmetro ganho tende a ser menos significativo que o desvio. Logo, algoritmos que consideram apenas o desvio apresentam em geral bom desempenho, o que foi confirmado pelos resultados obtidos.

Comparando o algoritmo de correção de desvio com o apresentado em [21], percebe-se grande semelhança. O algoritmo proposto em [21] é baseado em filtro de Kalman e, a não ser pela matriz de ganho de Kalman, é equivalente à correção de desvio proposta neste trabalho. De fato, conseguiu-se chegar a um algoritmo simplificado de [21] com maior rapidez computacional por uma via de dedução mais simples.

Algo importante de se ressaltar diz respeito à etapa de estimativa de movimento global. Uma vez que uma boa estimativa do FPN depende da estimativa de movimento global, as duas podem ser feitas em conjunto de modo a melhorar o desempenho geral do algoritmo. Em outras palavras, é aplicada uma correção do FPN prévia a cada novo par de quadros onde será estimado o movimento relativo. Isto garante uma estimativa de movimento com maior acurácia e um melhor refino do FPN por conseguinte. Existe, de fato, uma interdependência entre as duas etapas.

O Capítulo 5 apresentou o desenvolvimento de um novo algoritmo para correção

do ganho. A chave para se alcançar a solução neste caso residiu em dois fatores: primeiramente a adaptação do modelo de FPN ilustrado pela equação (5.1) que utiliza o produto Hadamard. Isso possibilitou a aplicação do logaritmo ao modelo, transformando o produto em adição. Seguiu-se, então, com uma solução análoga à do Capítulo 4.

Em segundo lugar, o uso de matrizes diagonais para realizar as operações de logaritmo e exponencial permitiram a fácil aplicação do gradiente à função erro. Tais matrizes (**L** e **E** respectivamente) são calculadas a partir dos valores sobre os quais devem operar. Desta maneira conseguiu-se uma equação de atualização simplificada, operando apenas com multiplicação de matrizes.

O Capítulo 6 apresentou a solução do problema batizada de Método Tensorial. Novamente o modelo de FPN foi adaptado e utilizou-se uma matriz diagonal de ganhos **A**. O gradiente em relação ao desvio foi obtido facilmente, já que a matriz de ganho **A** e a sua inversa elemento por elemento $\mathbf{A}^{-1}$ se fundem à matriz de movimento global **M**.

Todavia, para se obter o gradiente da função erro de estimativa em relação a matriz de ganhos recaiu-se sobre a derivada de um vetor em relação a uma matriz. Tal operação gera um tensor. Em outras palavras, o tensor gerado pode ser visto como uma matriz onde cada elemento é na verdade um vetor. Isso confere ao método sua natureza tensorial. De modo a contornar operações com tensor, foi apresentada uma solução para cada elemento da matriz diagonal de ganhos. Em relação a correção conjugada do desvio e do ganho o método Tensorial apresentou pequena melhora em termos de erro de estimativa. Isso é explicado, pois o primeiro apresenta simplificações em sua dedução: a equação de atualização da estimativa do desvio não considera ganho e vice-versa. Já no método tensorial, ambas as equações de atualização consideram o desvio e o ganho.

O método Tensorial-RLS apresentado no Capítulo 7 obteve o melhor desempenho em termos de erro de estimativa entre todos os algoritmos estudados, inclusive superando o algoritmo de referência do Estado da Arte baseado em filtro de Kalman e apresentado em [21]. Isso é explicado pelo fato de ser o único algoritmo a considerar o modelo completo de observação em toda sua dedução assim como os moldes do algoritmo RLS. Os algoritmos da família RLS têm uma estreita ligação com o filtro

de Kalman e são conhecidos pela rápida convergência e robustez a ruído. Sua desvantagem é a maior complexidade computacional e maior tempo de processamento quando comparado aos demais.

Finalmente o Capítulo 8 fez uma comparação entre todos os métodos apresentados neste trabalho. Os algoritmos foram submetidos a vários vídeos contendo FPN e o resultado médio do erro de estimativa foi apresentado. O algoritmo Tensorial-RLS2, que corrige o desvio e ganho ambos pelo método RLS, apresentou o menor erro de estimativa médio. Esse resultado está de acordo com o esperado considerando que a estrutura do algoritmo estava mais preparada para as condições impostas.

Ainda no Capítulo 8 os algoritmos foram aplicados a vídeos reais contendo FPN. Todos os algoritmos alcançaram certo nível de correção de FPN. Os resultados indicaram, porém, diferenças na ordem esperada de classificação dos algoritmos em termos de qualidade de imagem. Foram apresentadas discussões apontando pontos relevantes de cada método assim como possíveis interpretações desses resultados. Nos vídeos das imagens inteiras, notou-se que os métodos, destacando-se o Tensorial-RLS2 proposto neste trabalho, suavizaram o ruído FPN, tornando as imagens mais nítidas e possibilitando a percepção de detalhes antes ocultos nos vídeos originais.

Como sugestão para trabalhos futuros neste segmento pode-se citar a incorporação de um estágio de super-resolução à correção do FPN. Com a aplicação combinada de super-resolução e correção de FPN espera-se obter desempenho superior às suas aplicações individuais em termos de rapidez computacional, devido ao reaproveitamento de etapas de cálculo por ambos os algoritmos possuírem características semelhantes, e exatidão das estimativas, devido a dependência entre super-resolução e um vídeo livre de FPN.

Apêndice A

Publicações

Segue lista de publicações geradas pelo presente trabalho.

- [32] PIPA, D. R., DA SILVA, E. A. B., PAGLIARI, C. L., "Correção de não uniformidade em vídeo infravermelho por gradiente descendente". In: *XXVI SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES - SBrT'08 , 2008.*

Apêndice B

Métrica SSIM

Esta seção apresenta as equações utilizadas para o cálculo da métrica SSIM, que também podem ser encontradas juntamente com sua dedução em [1]. A equação da métrica SSIM é dada por

$$\mathrm{SSIM}(\mathbf{x},\mathbf{y}) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_x^2 + C_2)}, \tag{B.1}$$

onde

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \mu_y = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{B.2}$$

,

$$\sigma_x = \left[\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2\right]^{\frac{1}{2}} \quad \sigma_y = \left[\frac{1}{N-1}\sum_{i=1}^{N}(y_i - \mu_y)^2\right]^{\frac{1}{2}} \tag{B.3}$$

e

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y). \tag{B.4}$$

As constantes C_1 e C_2 servem para evitar instabilidades na medida caso $(\mu_x^2+\mu_y^2)$ ou $(\sigma_x^2 + \sigma_x^2)$ se aproximem muito de zero. Elas são definidas como

$$C_1 = (K_1 L)^2 \tag{B.5}$$

$$C_2 = (K_2 L)^2, \tag{B.6}$$

onde L é a faixa dinâmica dos valores dos pixels (255 para imagens em escala de cinza de 8 bits), $K_1 \ll 1$ e $K_2 \ll 1$ são constantes pequenas.

Apêndice C

Vetorização dos Tensoriais

Esta seção apresenta uma vetorização dos métodos Tensorial e Tensorial-RLS de modo a transformar as operações de laço (*loop*) em operações de álgebra linear. Essa vetorização se aplica às equações de atualização do ganho.

O cálculo do erro *a priori* permanece sendo

$$\mathbf{z}_k = \left(\mathbf{y}_{k-1} - \mathbf{b}\right) \tag{C.1}$$

$$\boldsymbol{\epsilon}_k = \mathbf{y}_k - \mathbf{A}_k\mathbf{M}_k\bar{\mathbf{A}}_k\mathbf{z}_k - \mathbf{b}. \tag{C.2}$$

Os índices k serão ocultados para facilitar a visualização. Define-se $\mathbf{Z} = \mathrm{diag}\left(\mathbf{z}\right)$ e redefine-se as matrizes do Capítulo 7 como

$$\dot{\bar{\mathbf{A}}} = -\mathrm{diag}\left(a_1^{-2}, a_2^{-2}, \cdots, a_N^{-2}\right) \tag{C.3}$$

$$\ddot{\bar{\mathbf{A}}} = 2\mathrm{diag}\left(a_1^{-3}, a_2^{-3}, \cdots, a_N^{-3}\right), \tag{C.4}$$

devendo ser calculadas a cada iteração. As equações de atualização do ganho se tornam então

$$\mathbf{U} = \mathrm{diag}\left(\mathbf{MAz}\right) - \mathbf{ZAM}\dot{\bar{\mathbf{A}}} \tag{C.5}$$

$$\mathbf{W} = \mathbf{Z}\left(2\mathbf{M}\dot{\bar{\mathbf{A}}} - \mathbf{AM}\ddot{\bar{\mathbf{A}}}\right) \tag{C.6}$$

$$\mathbf{v} = \mathbf{W}\boldsymbol{\epsilon} + \left[\sum_i \left[\mathbf{U} \circ \mathbf{U}\right]_{ij}\right]^{\mathrm{T}} \tag{C.7}$$

$$\boldsymbol{\gamma}_k = \lambda\boldsymbol{\gamma}_{k-1} + \mathbf{v} \tag{C.8}$$

$$\boldsymbol{\alpha} = \frac{\mathbf{U}^{\mathrm{T}}\boldsymbol{\epsilon}}{\boldsymbol{\gamma}_k} \tag{C.9}$$

$$\mathbf{A}_k = \mathbf{A}_{k-1} + \mathrm{diag}\left(\boldsymbol{\alpha}\right). \tag{C.10}$$

Apêndice D

Inicialização das variáveis nos algoritmos

Esta seção discutirá a inicialização das variáveis nos algoritmos. Por muitas variáveis serem comuns a todos os métodos, os comentários sobre suas inicializações foram aqui compilados.

1. $\hat{\mathbf{b}}$: O valor do desvio inicial é zero para todos os pixels. Ou seja, se não se tem conhecimento prévio sobre o desvio, o melhor é supor que não há desvio.

2. $\hat{\mathbf{A}}$: O mesmo acontece com o ganho que sempre é inicializado com valor 1 ou ganho neutro para todos os pixels. Na sua representação matricial, adota-se a matriz identidade.

3. $\hat{\mathbf{H}}$: Nos algoritmos RLS, a literatura [28] sugere que a matriz Hessiana seja inicializada com a identidade vezes uma constante δ igual uma estimativa da potência do sinal de entrada. No caso dos algoritmos deste trabalho, foi adotada somente a matriz identidade.

4. $\boldsymbol{\gamma}$: Para a correção de ganho por método Tensorial-RLS, o vetor $\boldsymbol{\gamma}$ tem o papel da matriz Hessiana. Portanto, é inicializado também com vetor unitário.

Referências Bibliográficas

[1] ZHOU WANG, ALAN CONRAD BOVIK, H. R. S., SIMONCELLI, E. P., "Image Quality Assessment: From Error Visibility to Structural Similarity", *IEEE TRANSACTIONS ON IMAGE PROCESSING*, v. 13, n. 4, 2004.

[2] RATLIFF, B. M., *A Generalized Algebraic Scene-based Nonuniformity Correction Algorithm*, Ph.D. Thesis, The University of New Mexico, 2004.

[3] SCRIBNER, D. A., KRUER, M. R., GRIDLEY, J. C., et al., "Physical Limitations to Nonuniformity Correction in IR Focal Plane Arrays", *Soc. Photo-Opt. Instrum. Eng.*, v. 865, pp. 185–202, 1987.

[4] PERRY, D. L., DERENIAK, E. L., "Linear Theory of Nonuniformity Correction in Infrared Staring Sensors", *Optical Engineering*, v. 32, n. 8, pp. 1853–1859, 1993.

[5] MALDAGUE, X. P. V., *Theory and Practice of Infrared Technology for Nondestructive Testing*. Wiley: New York, 2001.

[6] DEWITT, D., NUTTER, G., *Theory and practice of radiation thermometry*. Wiley: New York, 1988.

[7] TIPLER, P. A., *Física, volume 4: Óptica e física moderna*. LTC Editora: Rio de Janeiro, 1995.

[8] ALONSO, M., FIN, E. J., *Física, um curso universitário. Volume 1: Mecânica*. Edgard Blücher: São Paulo, 1972.

[9] SCRIBNER, D. A., KRUER, M. R., KILLIANY, J. M., "Infrared Focal Plane Array Technology", *Proceedings of the IEEE*, v. 79, n. 1, pp. 66–85, 1991.

[10] ÜMÝD TÜMKAYA, *Performance Assessment of Indium Antimonide Photon-detectors on Silicon Substrates*, Master's Thesis, The Middle East Technical University, Ankara, Turkey, 2003.

[11] BREITENSTEIN, O., LANGENKAMP, M., *Lock-in thermography*. Springer: Berlin, 2003.

[12] HARDIE, R. C., DROEGE, D. R., "A MAP estimator for simultaneous superresolution and detector nonuniformity correction", *EURASIP J. Appl. Signal Process.*, v. 2007, n. 1, pp. 206–206, 2007.

[13] HARRIS, J. G., CHIANG, Y.-M., "An Analog Implementation of the Constant Average Statistics Constraint For Sensor Calibration". In: *Advances in Neural Information Processing Systems*, v. 9, p. 699, The MIT Press, 1997.

[14] HARRIS, J. G., "Continuous-time calibration of VLSI sensors for gain and offset variations", 1995.

[15] CHIANG, Y., HARRIS, J., "An analog integrated circuit for continuous-time gain and offset calibration of sensor arrays", 1997.

[16] TORRES, S. N., HAYAT, M. M., "Kalman Filtering for Adaptive Nonuniformity Correction in Infrared Focal Plane Arrays", *Journal of the Optical Society of America*, v. 20, n. 3, pp. 470–480, 2003.

[17] S. N. TORRES, J. E. P., HAYAT, M. M., "Scene-based Nonuniformity Correction for Focal Plane Arrays Using the Method of the Inverse Covariance Form", *Applied Optics*, v. 42, n. 29, pp. 5872–5881, Oct. 2003.

[18] SCRIBNER, D., SARKADY, K., KRUER, M., et al., "Adaptive retina-like preprocessing for imaging detector arrays", *Neural Networks, 1993., IEEE International Conference on*, pp. 1955–1960 vol.3, 1993.

[19] R. C. HARDIE, M. M. HAYAT, E. E. A., YASUDA, B. J., "Scene-based Nonuniformity Correction with Video Sequences and Registration", *Applied Optics*, v. 39, n. 8, pp. 1241–1250, 2000.

[20] B. M. RATLIFF, M. M. H., HARDIE, R. C., "An Algebraic Algorithm for Nonuniformity Correction in Focal Plane Arrays", *Journal of the Optical Society of America*, v. 19, n. 9, pp. 1737–1747, 2002.

[21] AVERBUCH, A., LIRON, G., BOBROVSKY, B. Z., "Scene based non-uniformity correction in thermal images using Kalman filter", *Image Vision Comput.*, v. 25, n. 6, pp. 833–851, 2007.

[22] BOYD, S., VANDENBERGHE, L., *Convex Optimization*. Cambridge University Press, March 2004.

[23] FELIPPA, C. A., *Introduction to Finite Element Methods*. University of Colorado: Boulder, 2004.

[24] DATTORRO, J., *Convex Optimization and Euclidean Distance Geometry*. Meboo: Palo Alto, 2008.

[25] S. C. CAIN, M. M. H., ARMSTRONG, E. E., "Projection-based image registration in the presence of fixed-pattern noise", *IEEE Transactions on Image Processing*, v. 10, n. 12, pp. 1860–1872, 2001.

[26] HAYKIN, S., *Modern filters*. Macmillan Publishing Company: New York, NY, USA, 1989.

[27] HAYKIN, S., *Adaptive filter theory (3rd ed.)*. Prentice-Hall, Inc.: Upper Saddle River, NJ, USA, 1996.

[28] DINIZ, P. S. R., *Adaptive filtering: Algorithms and Practical Implementations (3rd ed.)*. Springer: Boston, MA, 2008.

[29] SORENSON, H. W., "Least-squares estimation: from Gauss to Kalman", *IEEE Spectrum*, v. 7, pp. 63–68, July 1970.

[30] KALMAN, R. E., "A New Approach to Linear Filtering and Prediction Problems", *Transactions of the ASME–Journal of Basic Engineering*, v. 82, n. Series D, pp. 35–45, 1960.

[31] HESTENES, M. R., STIEFEL, E., "Methods of Conjugate Gradients for Solving Linear Systems", *Journal of Research of the National Bureau of Standards*, v. 49, n. 6, pp. 409–436, December 1952.

[32] PIPA, D. R., DA SILVA, E. A. B., PAGLIARI, C. L., "Correção de não uniformidade em vídeo infravermelho por gradiente descendente". In: *XXVI SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES - SBrT'08*, 2008.