UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO DE CIÊNCIAS FÍSICAS E MATEMÁTICAS
DEPARTAMENTO DE MATEMÁTICA

Rafael Borges de Souza

Métricas em Processamento de Imagens

Florianópolis
2021

Rafael Borges de Souza

Métricas em Processamento de Imagens

Trabalho de Conclusão de Curso submetido ao Departamento de Matemática da Universidade Federal de Santa Catarina para a obtenção do título de Licenciado em Matemática.
Orientador: Prof. Leonardo Koller Sacht, Dr.

Florianópolis
2021

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Souza, Rafael Borges de
Métricas em Processamento de Imagens / Rafael Borges de Souza ; orientador, Leonardo Koller Sacht, 2021.
102 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro de Ciências Físicas e Matemáticas, Graduação em Matemática, Florianópolis, 2021.

Inclui referências.

1. Matemática. 2. Qualidade de imagens. 3. Métrica SSIM. 4. Otimização. I. Sacht, Leonardo Koller. II. Universidade Federal de Santa Catarina. Graduação em Matemática. III. Título.

Rafael Borges de Souza

Métricas em Processamento de Imagens

Este Trabalho Conclusão de Curso foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Leonardo Koller Sacht, Dr.
Universidade Federal de Santa Catarina

Prof. Diego Fernandes Nehab, Dr.
Instituto de Matemática Pura e Aplicada

Prof. Douglas Soares Gonçalves, Dr.
Universidade Federal de Santa Catarina

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de Licenciado em Matemática.

Coordenadoria do Curso de Graduação
em Matemática

Documento assinado digitalmente
Leonardo Koller Sacht
Data: 20/05/2021 17:03:53-0300
CPF: 057.981.059-38
Verifique as assinaturas em https://v.ufsc.br

Prof. Leonardo Koller Sacht, Dr.
Orientador

Florianópolis, 2021.

Dedicado aos meus pais

AGRADECIMENTOS

Aproveitarei este espaço para agradecer a pessoas que, direta ou indiretamente, contribuíram não somente para a execução deste trabalho mas também para minha formação enquanto licenciando em Matemática.

Primeiro, agradeço aos meus pais, Eliane e Clênio, os principais responsáveis por eu conseguir chegar ao final deste curso, por todo o amor e suporte incondicionais que tenho recebido, em especial, nesses anos de graduação. Ao Thalis, por ser o melhor amigo que eu poderia ter e por me apoiar em qualquer loucura que eu decida embarcar, mesmo sem entender muito bem o que eu faço da vida. À minha família e aos demais amigos de São Joaquim, que estiveram mais perto de mim, pessoalmente ou virtualmente, durante o período de quarentena.

À turma 15.2 de Engenharia Química que me proporcionou muitos bons momentos, desde os meus primeiros dias em Floripa, e me ajudou a descobrir que o que eu gostava mesmo era Matemática. Nunca esquecerei de vocês, Bruna, Mai, Paula, Rauel, Kevin e Andrieli. Um agradecimento especial à Carla, por ser a melhor companhia que eu poderia ter em todos esses anos de UFSC. Você esteve ao meu lado em todos os momentos, me apoiando, sem hesitar, mesmo quando decidi te “abandonar” e deixar a Engenharia.

Ao Yuri, Luana, Lucas, Lara, Gustavo, Ju e Denise, pela amizade desde os primeiros momentos no curso de Matemática, quando cheguei totalmente inseguro, vocês foram muito importantes para mim. À todas as amizades que nasceram em meio a almoços no RU e desesperos causados pelos exercícios de Álgebra. Ao Wander, que nunca hesitou em ficar resolvendo listas comigo até o horário da BU fechar. À Andresa, Luiz e Filipe, por todas as tardes no LEMAT estudando Topologia, que sempre acabavam em açaí e/ou pamonha e/ou caldo de cana. À Deborah, que se tornou a minha principal companhia virtual no período da pandemia, por compartilhar comigo as alegrias (e situações não tão alegres) do Estágio.

A todas as professoras e professores dos quais tive a honra de ser aluno em algum momento do curso, em especial à Alda, Silvia, Fernando, Sônia, Pinho, Boava, Marianna, Royer, Eliezer, Melissa, Virginia, Rosi, Francisco, Bortolan e María, por abrirem os portões do extraordinário mundo da Matemática e da Educação Matemática para mim. Agradeço por todo o carinho, dedicação e paciência que tiveram comigo nessa minha trajetória. Vocês são inspiradores e jamais serão esquecidos.

Ao professor Leonardo, que me orientou por quase 2 anos, por acreditar em mim e me incentivar, pela atenção dedicada a mim e por compartilhar comigo as ideias das quais este trabalho se originou. Obrigado também por todos os conselhos, sejam eles matemáticos ou não, levarei-os sempre comigo.

Aos professores Diego e Douglas, que aceitaram fazer parte da banca de avaliação deste trabalho.

Aos demais servidores e trabalhadores da Universidade, que tornam possível o estudo, não só meu, mas de milhares de alunos da UFSC.

Ao CNPq, por financiar meus estudos durante 2 anos e contribuir para a execução deste trabalho, do qual me orgulho tanto.

“Not all those who wander are lost.
...It’s just wanderlust.”
(Jack Antonoff e Lana Del Rey)

“The greatest films of all time were never made.
The greatest loves of all time are over now.”
(Aaron Dessner e Taylor Swift)

RESUMO

É uma tarefa importante avaliar a qualidade visual de imagens que sofreram algum tipo de distorção durante sua aquisição, processamento, compressão ou armazenamento. Através do conceito de imagem digital, é possível descrever imagens através de matrizes ou funções com domínio no plano e contradomínio num espaço de cores. Isso torna possível a utilização de métricas como um método para prever de forma automática a qualidade de imagens, já que a avaliação visual feita pelo olho humano, apesar de correta, é muito inconveniente. Métricas comuns para essa finalidade incluem o erro quadrático médio (MSE) e a similaridade estrutural (SSIM). Esta última recebeu considerável atenção na última década por comparar duas imagens através de sua informação estrutural, tendo como base estudos que apontam que essa é a principal informação extraída pelo sistema visual humano ao determinar se duas imagens são diferentes ou semelhantes. Neste trabalho, revisamos conceitos fundamentais de processamento de imagens e espaços métricos, além de apresentar duas das métricas mais usadas para avaliação da qualidade de imagens, MSE e SSIM, através de seus aspectos teóricos e implementações, explorando resultados positivos e limitações dessas métricas. Interessamo-nos, particularmente, em melhorar o desempenho da métrica SSIM, usando algoritmos de otimização que nos indicaram o melhor ajuste de parâmetros a ser feito. Observou-se que há potencialidade em alterar o formato e o tamanho da matriz gaussiana usada no cálculo da métrica SSIM, para que esta se assemelhe, da melhor forma possível, à avaliação feita pelo sistema visual humano.

Palavras-chave: Imagem digital. Qualidade de imagens. SSIM. Otimização.

ABSTRACT

It is an important task to evaluate the visual quality of images that suffered some kind of distortion during their acquisition, processing, compression or storing. Through the concept of digital image, it is possible to represent images as matrices or functions whose domains are the plane and target sets are a color space. This allows the use of metrics as a method to automatically predict the quality of images, since the assessment by the human eye, despite correct, is very inconvenient. The mean square error (MSE) and structural similarity (SSIM) are common metrics used for this task. The latter received considerable attention in the last decade for comparing two images through their structural information, being based by studies that indicate that this is the main information extracted by the human visual system to determine if two images are different or similar. In this work, we review the fundamental concepts of image processing and metric spaces, as well as present two of the most used metrics to assess the quality of images, MSE and SSIM, through theoretical aspects and implementations, exploring positive results and limitations of these metrics. We are particularly interested in improving the performance of the SSIM metric, using optimization algorithms that pointed us to a better tuning of its parameters. We have observed that a promising direction is to alter the shape and the size of the Gaussian matrix used in the SSIM metric computation, to make it as similar as possible to the assessment done by the human visual system.

Keywords: Digital image. Image quality. SSIM. Optimization.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

Quando observamos uma imagem qualquer (uma fotografia, por exemplo), perceptualmente recebemos de cada ponto dessa imagem um impulso luminoso que nos fornece uma informação de cor para esse ponto. Assim, de acordo com Gomes e Velho (1994), podemos descrever uma imagem através de um modelo matemático: considera-se uma função com domínio em uma superfície bidimensional que associa cada ponto dessa superfície a um valor em um espaço de cores.

Imagens estão sujeitas a uma grande variedade de distorções durante sua aquisição, processamento, compressão ou armazenamento, qualquer uma das quais pode resultar em degradação da sua qualidade visual. Uma maneira bastante difundida para fazer avaliação da qualidade visual de imagens é chamada de *full-reference* e consiste em comparar a imagem degradada com uma imagem de referência. Dessa maneira, quanto mais próximas visualmente forem as duas imagens, maior é a qualidade visual da imagem degradada. Como afirmam Wang *et al.* (2004), o único método "correto" para avaliar a qualidade de uma imagem é através da avaliação subjetiva (observação humana). Na prática, no entanto, esta avaliação é muito inconveniente, demorada e cara. Dessa forma, o propósito da pesquisa em avaliação objetiva da qualidade de imagens é desenvolver medidas quantitativas que possam prever automaticamente a qualidade visual de uma imagem.

A Figura 1 ilustra uma imagem que recebeu dois tipos diferentes de distorção: compressão do tipo JPEG2000 e borramento gaussiano. Visualmente, é possível medir a qualidade das duas imagens distorcidas por meio de uma nota ou avaliação descritiva mas, como vimos, estaremos interessados em fazer isso de forma automática, por meio de algoritmos que têm como objetivo imitar a avaliação subjetiva feita pelo sistema visual humano.

Diante disso, é possível utilizar o conceito de métrica para calcular distâncias entre duas imagens e interpretá-las como medidas objetivas para a avaliação da qualidade das imagens. Dado um conjunto M não-vazio qualquer, uma métrica sobre M é uma função $d : M \times M \to \mathbb{R}$ que satisfaz algumas condições: a distância entre dois pontos de M é sempre não negativa e simétrica, vale a desigualdade triangular e a distância é nula se, e somente se, os pontos forem iguais. Fixado um conjunto M não-vazio, podem existir muitas métricas sobre M, logo, se os pontos de M são imagens, existem muitas métricas que podem ser usadas para medir a qualidade dessas imagens. Por esse motivo, torna-se relevante estudar métricas já conhecidas e identificar qual possui melhor desempenho na avaliação da qualidade visual. Usa-se como base para essa pesquisa a avaliação subjetiva, ou seja, quanto mais semelhantes forem os

Figura 1 – Imagem distorcida de duas maneiras diferentes. (a) Imagem original, (b) imagem com compressão JPEG2000 e (c) com borramento gaussiano

(a) (b) (c)

Fonte – Ponomarenko *et al.* (2009).

valores calculados pela métrica e os valores obtidos por uma avaliação subjetiva das imagens, melhor desempenho a métrica possui.

Wang *et al.* (2004) desenvolveram um método de avaliação com o propósito de ter desempenho melhor do que os métodos que eram utilizados até então. Essa nova métrica, chamada de SSIM, compara duas imagens através de sua informação estrutural, tendo como base estudos que mostraram que essa é a principal informação extraída pelo sistema visual humano ao fazer a comparação entre duas imagens. Desde quando foi proposta, a métrica SSIM vem recebendo considerável atenção e tem resultados bastante promissores. Apesar disso, suas limitações ainda não são totalmente conhecidas e vêm sendo exploradas cada vez mais. Chen *et al.* (2006) e Pambrun e Noumeir (2015) discutem algumas dessas limitações e Zhang *et al.* (2013) e Golestani e Ghambari (2014) apresentam possíveis soluções para elas. Estes últimos sugerem que alguns parâmetros ocultos na métrica SSIM podem ser ajustados para obter melhor desempenho.

Dessa forma, pretendemos apresentar conceitos fundamentais do processamento de imagens digitais, as diversas ditorções que podem acarretar em perda da qualidade visual dessas imagens e estudaremos algumas métricas já conhecidas que medem essa qualidade visual. Além disso, nos dedicaremos principalmente a abordar alguns ajustes que podem ser feitos na métrica SSIM a fim de melhorar seu desempenho.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Melhorar o desempenho da métrica SSIM na avaliação da qualidade visual de imagens.

1.2.2 Objetivos Específicos

- Revisar conceitos fundamentais de processamento de imagens;
- Apresentar algumas das métricas utilizadas para avaliar a qualidade visual de imagens;
- Entender a importância da métrica SSIM neste contexto;
- Explorar, através de estudos e implementações, pontos de melhoria nesta métrica.

1.3 JUSTIFICATIVA

De acordo com Wang *et al.* (2012), existem sistemas de controle de qualidade de imagens e vídeos que cumprem o papel de monitorar e se ajustar automaticamente para obter os melhores dados de qualidade de imagem e vídeo. A ocorrência de distorções nas imagens, como borramentos ou adição de ruídos, podem ter sua origem na captura, no próprio processamento, na compressão, no armazenamento ou na reprodução de uma imagem.

Dito isso, podemos nos questionar sobre qual seria a melhor técnica para fazer a avaliação da qualidade de uma imagem. Lembrando que, quando estamos nos referindo à qualidade de uma imagem, estamos fazendo a comparação dessa imagem com uma imagem de referência, de acordo com o método *full-reference*.

> Nenhuma técnica apresenta desempenho ótimo para todas as aplicações. [...] A determinação da semelhança entre imagens está condicionada a aspectos objetivos mais elementares tais como forma, cor, textura, tamanho disposição espacial. Por isso, o seu grau elevado de relatividade. Condiciona-se também a aspectos subjetivos como memória, reminiscências e estética, o que lhe confere alto grau de dependência à perspectiva do observador. (TANNÚS, 2008, p. 24).

Sendo assim, a melhor técnica para este tipo de comparação de imagens continua sendo a percepção humana. Porém, na aplicação de processamento de imagens digitais, ainda se considera relevante o estudo de técnicas neste tipo de comparação para implementação computacional. Este tipo de investigação e estudo comparativo proporciona uma melhor compreensão das técnicas já conhecidas, possibilitando que sejam estabelecidas delimitações no campo de atuação de cada uma delas. Os métodos objetivos para avaliação de qualidade de imagens que serão estudadas aqui se baseiam no conceito matemático de métrica.

Através do estudo dos principais detalhes da métrica SSIM (assim como outras métricas) e também da análise dos resultados obtidos na pesquisa investigativa da métrica, pode-se estabelecer seus pontos fortes e fracos em diferentes aplicações, bem como enxergar tais pontos fracos como possíveis pontos de melhoria para aumentar

seu desempenho. Isso tudo resulta em contribuição para eventuais pesquisas futuras e também para o desenvolvimento de outras possíveis técnicas (podendo usar elementos das técnicas conhecidas ou serem totalmente inovadoras) que possam ter um melhor desempenho na aplicação desejada.

Podemos observar a relevância e atualidade do tema em alguns estudos como os de Pambrun e Noumeir (2015), Zhang *et al.* (2013) e Golestani e Ghambari (2014) que, mesmo depois de anos após a publicação da métrica SSIM, continuam investigando maneiras de melhorar seu desempenho e obter resultados melhores na avaliação da qualidade visual de imagens.

1.4 ESTRUTURA DO TRABALHO

O trabalho que desenvolvemos poderia ser estruturado de muitas maneiras distintas, mas optamos por fazê-lo na ordem cronológica em que os assuntos foram estudados e explorados, de uma maneira bem construtiva, que acreditamos ser mais agradável ao leitor.

No segundo capítulo, são apresentados conceitos básicos e terminologias sobre processamento de imagens digitais, que são utilizados ao longo de todo o trabalho. No terceiro capítulo, exploramos o conceito de métrica, juntamente com exemplos de espaços métricos conhecidos e descrevemos como é possível utilizar este conceito para avaliar a qualidade visual de uma imagem. Conhecemos também duas das principais métricas utilizadas para esta finalidade, bem como seus pontos fortes e suas limitações. No capítulo seguinte, apresentamos as primeiras modificações, encontradas na literatura, feitas na métrica SSIM para aumentar seu desempenho. Ao final deste capítulo, surge a necessidade de falar um pouco sobre teoria de otimização, e este é o assunto tratado no capítulo 5. No capítulo 6, realizamos várias tentativas de resolução de um problema de otimização que nos fornece o melhor ajuste de parâmetros possível para a métrica SSIM e, no último capítulo, fazemos as últimas considerações a respeito do assunto. Nos apêndices, o leitor pode encontrar textos complementares que aprofundam assuntos comentados ao longo do trabalho, além de implementações de códigos que utilizamos para gerar nossos resultados.

2 A IMAGEM COMO UM SINAL

Sinais são elementos que estão comumente presentes na nossa interação com o ambiente e podem ser expressos através de funções que contêm informações acerca do comportamento ou da natureza de fenômenos presentes no ambiente. Os sinais podem estar diretamente relacionados aos nossos sentidos, e temos como exemplos sinais luminosos que nos permitem fazer a visualização do ambiente ou sinais sonoros que fazem com que possamos ouvir sons e nos comunicar com outras pessoas. Além disso, há tipos de sinais que são essenciais no desenvolvimento de tecnologias, como sinais eletromagnéticos.

Um sinal se manifesta através de uma variação de uma determinada grandeza física. Essa variação pode ocorrer com relação ao espaço ou ao tempo (GOMES; VELHO, 1994). Neste contexto, podemos tomar como exemplo os sinais de áudio, que nada mais são do que uma variação da densidade do ar através do tempo ou um sinal de vídeo, que além dessa variação no tempo, possui imagens que são compostas por uma variação de cor nos pontos do plano. Uma maneira de descrever matematicamente um sinal se utiliza de um modelo funcional e espacial, e podemos definir da seguinte forma:

Definição 1. *Dados* $n, m \in \mathbb{N}^*$ *e* $U \subseteq \mathbb{R}^m$*, um* sinal *é uma função* $f : U \to \mathbb{R}^n$*.*

Da Definição 1, segue que a grandeza física do sinal é representada por um vetor de dimensão n, que varia em um espaço que possui dimensão m.

A Teoria de Sinais é um campo consolidado com inúmeras aplicações, como vimos através de exemplos de fenômenos que podem ser modelados por sinais. Para o presente trabalho, vamos nos preocupar em particularizar os sinais com imagens, ou seja, utilizar conceitos e ferramentas da Teoria de Sinais para o processamento gráfico.

2.1 IMAGEM CONTÍNUA

Embora existam muitos modelos matemáticos para descrever uma imagem, dependendo da sua utilização e contexto, utilizaremos o modelo mais adequado para aplicações em computação gráfica, de acordo com Gomes e Velho (1994), que é o modelo espacial de imagem. Dado de maneira muito semelhante à definição 1, vamos nos atentar principalmente aos nomes dos conjuntos.

Definição 2. *Dados* $U \subseteq \mathbb{R}^2$ *e C um espaço vetorial, chamamos de* imagem contínua *uma função* $i : U \to C$*. Neste caso, U é chamado de* conjunto suporte da imagem*, enquanto C é um* espaço de cor*. À imagem de i damos o nome de* gamute de cores*.*

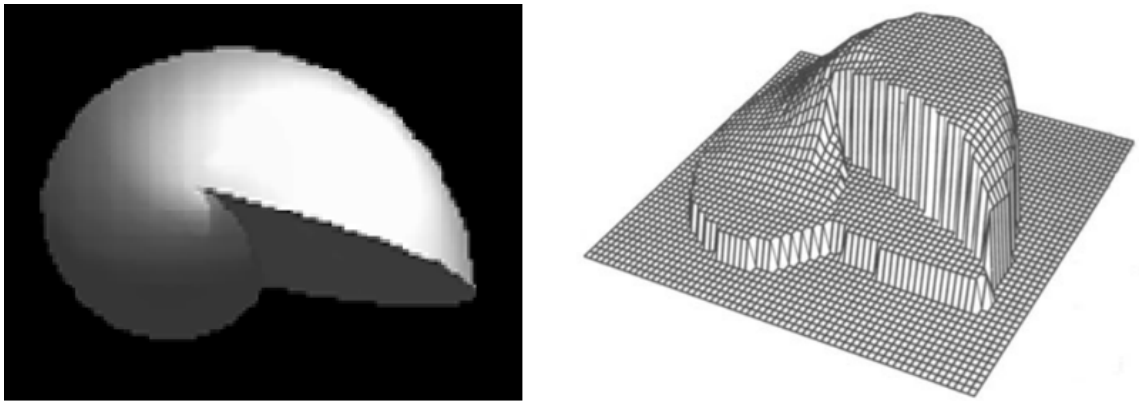
Aqui, vamos tomar atenção ao fato de estarmos usando o termo “imagem contínua” apenas para contrastar com o conceito de imagem digital (discretizada) que será

apresentado em seguida. Nada tem a ver com o fato de i, da definição 2, ser ou não contínua sob o ponto de vista do conceito de "função contínua", como aprendemos em disciplinas de cálculo.

Pode-se considerar que esse modelo é intuitivo se pensarmos sob a ótica da maneira funcionalista de entender como percebemos uma imagem perceptualmente: considerando uma imagem estando contida em um plano, entendemos que cada ponto deste plano nos fornece informações de luz, cor, intensidade, textura etc, que podem ser armazenadas em um vetor de C.

Além disso, no caso em que C é um espaço de cores de dimensão 1, chama-se a imagem de *monocromática*, e assim, cada ponto da imagem nos fornece apenas a informação da intensidade da cor naquele ponto, como podemos ver no exemplo da Figura 2.

Figura 2 – Imagem monocromática e seu gráfico.



Fonte – Gomes e Velho (1994).

2.2 IMAGEM DIGITAL

Como estamos interessados em estudar os conceitos aqui apresentados sob a ótica do processamento gráfico, é usual que se faça uma discretização do conjunto suporte da imagem e do espaço de cores, e então, nesse momento, a imagem passa a ser representada por uma amostragem da imagem contínua no conjunto suporte discreto obtido ao discretizar o conjunto suporte original. Cada ponto, ou coordenada, (x_i, y_i) do conjunto suporte discreto (domínio) é chamado de *pixel*.

Ao fazermos tal discretização da imagem, podemos tomar o conjunto suporte como um retângulo e fazer uma representação matricial da imagem, onde cada elemento da matriz é um vetor de C, que carrega as informações de cor da imagem. Se a imagem for monocromática, como na Figura 2, cada elemento (pixel) é um número real que representa a luminância da cor naquele ponto.

Definição 3. *Dados $n, m \in \mathbb{N}^*$, M uma matriz $n \times m$ e i uma imagem contínua, se M é uma representação matricial para o conjunto suporte de i, chamamos o produto $n \times m$ de* resolução espacial *(ou somente* resolução*) da imagem i.*

A Figura 3 ilustra o conceito de resolução espacial, apresentando uma imagem em diferentes resoluções.

Figura 3 – Imagem com diferentes resoluções espaciais. (a) 512×512 pixels; (b) 64×64 pixels.

(a) (b)

Fonte – Ninassi *et al.* (2006). Modificada pelo autor (2021).

Quanto maior for a resolução de uma imagem, maior será a amostragem da imagem contínua, ou seja, mais pixels temos transmitindo informações de cor na imagem. Perceptualmente, uma imagem com maior resolução apresenta mais detalhes, como podemos ver na Figura 4, que apresenta as mesmas imagens da Figura 3 após "esticarmos" a imagem menor para fazer uma comparação mais justa.

Figura 4 – Imagem com diferentes resoluções espaciais ajustadas para parecerem do mesmo tamanho. (a) 512 × 512 pixels; (b) 64 × 64 pixels.

(a) (b)

Fonte – Ninassi *et al.* (2006). Modificada pelo autor (2021).

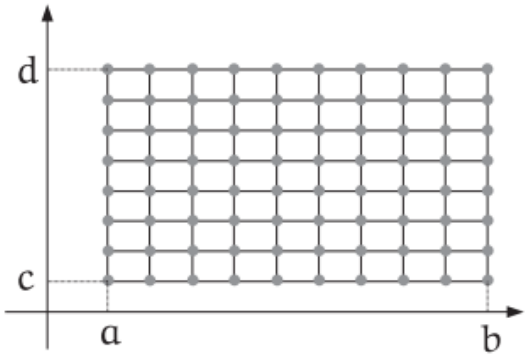
Definição 4. *Dada uma imagem contínua $i : U \to C$, e U' e C' discretizações para U e C, respectivamente, chamamos de* imagem digital *uma função $i' : U' \to C'$ que representa uma amostragem para a imagem contínua.*

Os elementos de uma imagem digital consistem, basicamente, das coordenadas dos pixels e da informação de cor de cada pixel. Assim, para representarmos uma imagem digital em um dispositivo gráfico como o computador, por exemplo, existem duas questões principais a serem resolvidas e que fazem parte do conceito de imagem digital: de que forma fazer a discretização do conjunto suporte da imagem a fim de obter os pixels e de que forma fazer a discretização do espaço de cores da imagem a fim de obter a informação de cor em cada pixel?

2.2.1 Discretização e quantização

Há, na Teoria de Sinais, um estudo aprofundado sobre amostragem de sinais que envolve métodos para otimizar as amostragens e também maneiras de recuperar um sinal contínuo através da reconstrução de um sinal discreto. Em outros contextos, é possível trabalhar com pixels de diferentes tamanhos e formatos, mas ao trabalharmos com dispositivos gráficos, devemos entender a imagem através de suas representações matriciais.

Figura 5 – Reticulado uniforme da representação matricial da imagem.



Fonte – Gomes e Velho (1994).

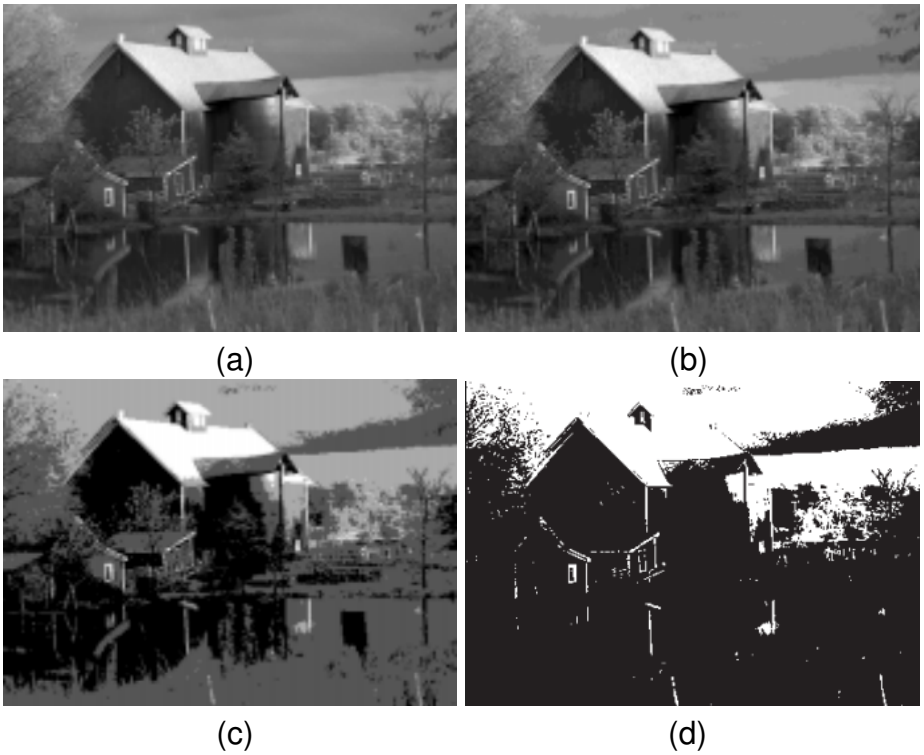
Para entendermos melhor como isso acontece, tome $i : U \to C$ uma imagem contínua, $a, b, c, d \in \mathbb{R}$ e um retângulo U de modo que $U = [a, b] \times [c, d]$. Para discretizarmos o retângulo U, usamos os pontos de um reticulado bidimensional Δ tal que, fixados $\Delta_x, \Delta_y \in \mathbb{R}$, temos

$$\Delta = \left\{ (x_j, y_k) \in U \mid x_j = j \cdot \Delta_x \text{ e } y_k = k \cdot \Delta_y \text{ com } j, k \in \mathbb{Z} \right\}, \tag{1}$$

como mostra a Figura 5. Cada pixel (x_j, y_k) da imagem pode então ser representado pelas suas coordenadas inteiras (j, k) no reticulado. Assim, a imagem pode ser representada de forma conveniente no formato matricial. Nessa representação, ela está associada a uma matriz A, de ordem $m \times n$ (resolução espacial) onde cada elemento (pixel) de A carrega as informações de cor da imagem, ou seja, $A = (a_{jk}) = (i(x_j, y_k))$.

Para exibir uma imagem em um dispositivo gráfico, o gamute de cores da imagem não pode ser maior do que a quantidade de cores que o dispositivo reproduz. Portanto, é necessário fazer uma discretização do espaço de cores da imagem. Além disso, essa discretização se faz necessária também quando queremos reduzir a quantidade de bits necessária para armazenar as informações de cor de uma imagem. Dada uma imagem contínua $i : U \to C$, a discretização do espaço de cores C recebe o nome de *quantização*.

Figura 6 – Imagem quantizada com diferentes níveis de cor. (a) Imagem em 256 tons de cinza, (b) 16 tons de cinza, (c) 8 tons de cinza, 2 tons de cinza.



Fonte – Gomes e Velho (1994).

Para ilustrar o processo de quantização do espaço de cores, considere, por exemplo, uma imagem monocromática cujo espaço de cores possui 256 níveis de intensidade de cinza (8 bits). Uma maneira possível para quantizar tal espaço é identificar qual o nível médio de cinza dentre os 256 disponíveis e fazer a seguinte função de quantização: aqueles valores que estão abaixo do nível médio serão levados no valor mínimo (nível 0) e aqueles que estão acima do valor médio serão levados no valor

máximo (nível 255). Assim, transformamos uma imagem que possuía 256 níveis de cores em uma que agora possui apenas 2. A Figura 6(a)-(d) nos mostram uma mesma imagem monocromática quantizada de quatro maneiras diferentes. Na Figura 6 (a) vemos a imagem em 256 tons de cinza, na (b) reduzimos para 16 tons de cinza, na (c) temos 8 tons de cinza e, por fim, temos apenas 2 tons de cinza na (d).

Na Figura 6(b)-(d), é possível perceber uma curva de fronteira entre níveis de quantização diferentes. Portanto, podemos perceber que embora um número menor de níveis nos garanta uma imagem de resolução de cor menor (e, por consequência, a imagem requeira um número menor de bits para armazenamento), tais curvas de fronteira tornam a imagem cada vez menos natural.

Assim, pode-se fazer o questionamento de qual seria o nível de quantização ideal, ou seja, uma quantização que otimize o tamanho da imagem e também garanta que as curvas de fronteira sejam imperceptíveis. Segundo Gomes e Velho (1994) temos que, em geral, para imagens monocromáticas, 256 níveis de quantização (8 bits) são suficientes para evitar a percepção do contorno de quantização, independente do método de quantização utilizado. Para imagens a cores, utiliza-se uma quantização em 24 bits, ou seja, 8 bits de quantização para cada uma das três componentes do espaço de cores. Mais detalhes sobre espaços de cores e as componentes RGB (vermelho, verde e azul) do contradomínio de imagens digitais coloridas são encontrados no Apêndice A.

2.2.2 Distorções e filtragem

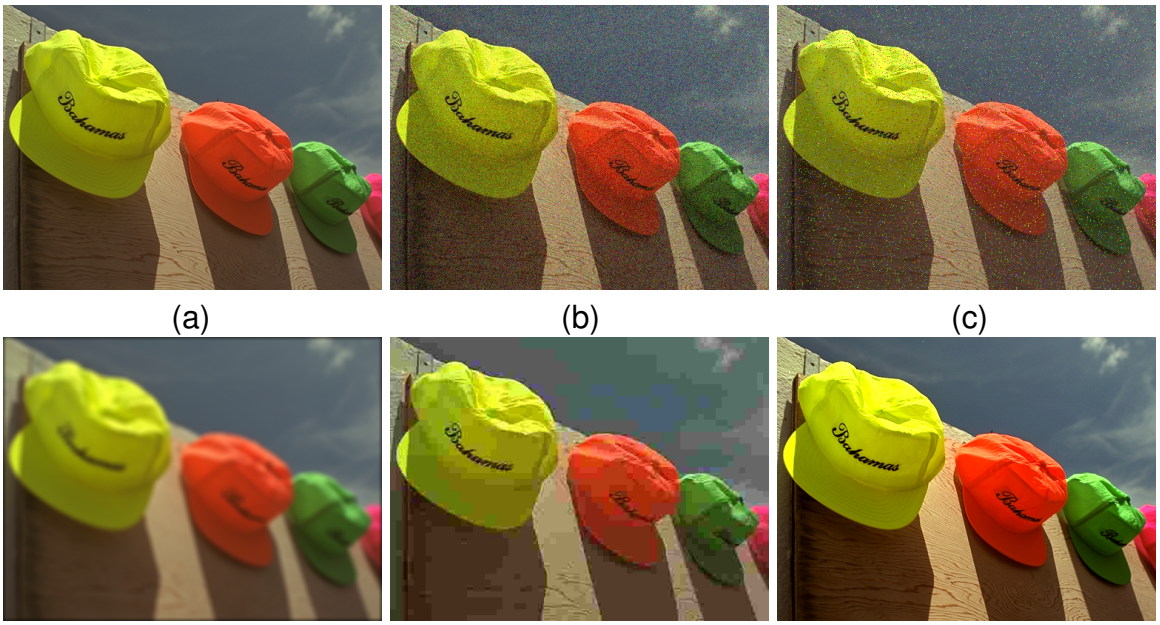
Como já comentado, não é incomum que ocorram distorções em uma imagem ao manipulá-la, seja através do processamento, armazenamento, reprodução ou na própria captura da imagem. Sendo assim, podemos utilizar modelagem matemática para entender e simular o comportamento dessas distorções que causam degradação da qualidade visual de uma imagem.

Damos o nome de *filtragem* a qualquer operação unária realizada em uma imagem digital, ou seja, uma operação que não envolva mais do que uma imagem. É importante fazer essa distinção pois em outros contextos são utilizadas operações em que duas ou mais imagens são combinadas para gerar uma nova. À aplicação T que define uma filtragem no espaço de imagens $I = \left\{ i : U \to C \mid U \subseteq \mathbb{R}^2 \right\}$ damos o nome de *filtro*. Assim, $g(x, y) = T(i(x, y))$ define uma imagem g obtida através de uma filtragem da imagem i com o filtro T.

Podemos encontrar imagens digitais distorcidas “artificialmente”, ou seja, filtradas, em bancos de imagens como o de Ponomarenko *et al.* (2009) e utilizá-las no nosso estudo para investigar métodos que avaliam a qualidade de imagens de forma automática.

A Figura 7 mostra uma imagem que foi filtrada por cinco filtros diferentes: adição de ruído gaussiano, ruído “sal e pimenta” , borramento gaussiano, compressão do tipo JPEG e mudança de contraste. Detalhes sobre filtragem de imagens digitais e descrições de modelos matemáticos para alguns desses filtros são apresentados no Apêndice B.

Figura 7 – Imagem distorcida por cinco filtros diferentes: (a) imagem original, (b) ruído gaussiano, (c) ruído “sal e pimenta” , (d) borramento gaussiano, (e) compressão do tipo JPEG e (f) mudança de contraste.



Fonte – Ponomarenko *et al.* (2009).

3 MÉTRICAS PARA AVALIAÇÃO DA QUALIDADE DE IMAGENS

Os métodos objetivos para avaliação de qualidade de imagens que serão estudadas aqui se baseiam no conceito matemático de métrica, e é este conceito que abordaremos com mais detalhes a seguir.

3.1 ESPAÇOS MÉTRICOS

Ainda na escola, somos introduzidos à ideia de distância entre números reais através da função modular e, dessa maneira, é possível criar uma função para calcular distâncias entre pontos de $\mathbb{R}$ como, por exemplo, $d : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ tal que para um par (x, y) do plano, a distância entre x e y é $d(x, y) = |x - y|$.

Como afirma Kreyszig (1978), é possível fazer uma generalização da ideia de distância entre pontos, como descrito acima, na qual $\mathbb{R}$ pode ser substituído por um conjunto não-vazio qualquer e extrai-se da função modular as propriedades mais fundamentais que a caracterizam como a "distância" entre pontos de $\mathbb{R}$. A escolha de tais características está longe de ser trivial e o conceito que será apresentado a seguir levou mais de sessenta anos para ser formulado.

Definição 5. *Dado um conjunto M não-vazio, uma* métrica sobre M *é uma função $d : M \times M \to \mathbb{R}$ que satisfaz, para quaisquer $x, y, z \in M$:*

1. $d(x, y) \geq 0$*;*
2. $d(x, y) = 0$ *se e somente se $x = y$;*
3. $d(x, y) \leq d(x, z) + d(z, y)$*; (desigualdade triangular)*
4. $d(x, y) = d(y, x)$*. (simetria)*

Ainda nesta definição, dado um conjunto M não-vazio e uma métrica d sobre M, o par (M, d) é dito um *espaço métrico*.

Além da métrica usual em $\mathbb{R}$, apresentada no primeiro parágrafo desta seção, podemos tomar como exemplo o espaço métrico $(\mathbb{R}^n, d)$ com a seguinte métrica d sobre $\mathbb{R}^n$, conhecida como *métrica euclidiana*:

$$d(x, y) = d((x_1, ..., x_n), (y_1, ..., y_n)) = \sqrt{|x_1 - y_1|^2 + ... + |x_n - y_n|^2}, \tag{2}$$

em que n é um número natural não nulo e $x = (x_1, ..., x_n)$ e $y = (y_1, ..., y_n)$ são dois vetores de $\mathbb{R}^n$.

Dado um conjunto qualquer, é possível que existam muitas métricas diferentes sobre este conjunto e a escolha de uma métrica depende de cada aplicação ou problema a ser resolvido. A métrica euclidiana, por exemplo, é bastante conhecida, mas existem outras métricas que podem ser definidas sobre o mesmo conjunto $\mathbb{R}^n$, como

$$d_1(x, y) = d_1((x_1, ..., x_n), (y_1, ..., y_n)) = |x_1 - y_1| + ... + |x_n - y_n|, \tag{3}$$

conhecida como *métrica da soma*.

Para ter ideia do quão geral esse conceito pode ser, podemos tomar outros espaços métricos interessantes, como o conjunto ℓ^∞ das sequências limitadas de números complexos. Assim, dado $x \in \ell^\infty$, x é da forma $x = (x_n) = (x_1, x_2, \dots)$ com $x_n \in \mathbb{C}$ para todo $n \in \mathbb{N}^*$. Como toda sequência em ℓ^∞ é limitada, faz sentido definir uma métrica d_∞ sobre ℓ^∞ da seguinte maneira:

$$d_\infty(x, y) = \sup_{n \in \mathbb{N}^*} |x_n - y_n|. \tag{4}$$

Mais exemplos de espaços métricos e as verificações dos axiomas 1 a 4 (Definição 5) para os espaços métricos citados podem ser encontrados no livro de Kreyszig (1978).

Como vimos anteriormente, estamos estudando o processamento de imagens sob a ótica da teoria dos sinais. Portanto, a partir de agora, vamos nos preocupar em particularizar o conceito de métrica para o caso em que M é um conjunto cujos elementos são sinais (funções). Em particular, considere o conjunto $I = \left\{ i : U \to C \mid U \subseteq \mathbb{R}^2 \right\}$, que chamamos de *espaço de imagens*

De acordo com a Definição 5, para ser chamada de *métrica*, uma função deve satisfazer as quatro propriedades apresentadas. Na prática, no entanto, é possível trabalhar com funções que se "assemelham" a métricas, para as quais algum (ou alguns) dos quatro axiomas não é satisfeito. Esse fato pode acontecer com bastante frequência ao definir funções sobre um espaço de imagens I. Dessa forma, cabem as definições a seguir.

Definição 6. *Dado um conjunto não vazio M e uma função d sobre M, dizemos que d é uma* semimétrica *se d satisfaz as propriedades 1, 2 e 4 da Definição 5.*

Definição 7. *Uma* pseudo-métrica *d sobre um conjunto de imagens I é uma métrica em que é possível ter a igualdade $d(x, y) = 0$ mesmo que duas imagens $x, y \in I$ não sejam iguais do ponto de vista funcional.*

Além disso, já vimos que a nossa percepção de sinais, em particular de imagens, se dá através da relação entre o fenômeno descrito pelo sinal e os nossos sentidos humanos. Por conta disso, e motivado também por um dos axiomas de métrica, pode-se fazer a seguinte definição:

Definição 8. *Dado um conjunto de imagens I, uma métrica d sobre I e $x, y \in I$ dizemos que d é uma* métrica perceptual *se $d(x, y) = 0$ sempre que x e y forem imagens indistinguíveis do ponto de vista do SVH (sistema visual humano).*

Em resumo, dado o problema proposto na introdução deste trabalho, vamos estudar algumas métricas (ou semimétricas ou pseudométricas) utilizadas para avaliar a qualidade de imagens através da distância entre elas. Uma métrica ideal para esse fim seria uma métrica que reproduz exatamente a avaliação feita pelo SVH, o que implicaria numa pseudo-métrica perceptual. Como sabemos que não é possível reproduzir a exata avaliação do SVH como uma métrica, o objetivo do estudo passa a ser procurar algo mais próximo possível da métrica ideal.

Para simplificar a linguagem e evitar possíveis confusões, iremos chamar as funções que serão estudadas a seguir simplesmente de *métricas*, mesmo que, do ponto de vista conceitual, sejam semimétricas ou pseudométricas.

3.2 MSE E PSNR

De acordo com Wang e Bovik (2009), o objetivo de uma métrica para avaliar a qualidade de um sinal é comparar dois sinais, fornecendo uma pontuação quantitativa que descreve o grau de similaridade entre eles, supondo que um deles é o sinal original enquanto o outro é distorcido.

Um método bastante simples, herdado da probabilidade e estatística, muito usado para fazer essa medição é o *erro quadrático médio*, conhecido na literatura pela sigla MSE (*mean squared error*, em inglês). Suponha duas imagens digitais, x e y, com N pixels, tais que x_i e y_i são os valores dos i-ésimos pixels em x e y, respectivamente. Temos que o MSE entre as imagens é dado por

$$MSE(x, y) = \frac{1}{N} \sum_{i=1}^{N} (x_i - y_i)^2. \tag{5}$$

Teorema 9. *Dado um conjunto de imagens I, a função $MSE : I \times I \to \mathbb{R}$ define uma semimétrica sobre I.*

Demonstração. Sejam $x, y, z \in I$ imagens quaisquer. As propriedades 1, 2 e 4 da Definição 5 são verificadas:

- $MSE(x, y) = \frac{1}{N} \sum_{i=1}^{N} (x_i - y_i)^2 \geq 0$, pois $\frac{1}{N} > 0$ e $(x_i - y_i)^2 \geq 0$ para todo $i \in \{0, ..., N\}$.
- – Suponha que $x = y$. Então, $MSE(x, x) = \frac{1}{N} \sum_{i=1}^{N} (x_i - x_i)^2 = 0$
  – Suponha que $MSE(x, y) = 0$. Então $\sum_{i=1}^{N} (x_i - y_i)^2 = 0$. Assim, para cada $i \in \{0, ..., N\}$, temos que $(x_i - y_i)^2 = 0$, ou seja, $x_i - y_i = 0$, o que implica que $x = y$.

- $MSE(x, y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - x_i)^2 = MSE(x, y)$.

□

A desigualdade triangular não vale para a função MSE e isso pode ser observado escolhendo as imagens digitais $x = [1 \quad 2]$, $y = [3 \quad 4]$ e $z = [2 \quad 3]$. Temos que $MSE(x, y) = 4$ e $MSE(x, z) + MSE(z, y) = 1 + 1 = 2$. Note que, se trocássemos $(x_i - y_i)^2$ por $|x_i - y_i|$ na definição da função MSE (Equação (5)), teríamos uma função semelhante à métrica da soma (Equação (3)), na qual valeria a desigualdade triangular e seria, portanto, uma métrica.

Apesar de ser uma semimétrica, do ponto de vista matemático, e ser usado para avaliar a qualidade de uma imagem, o MSE é mais comumente encontrado na literatura em uma versão diferente. Chamada de *relação sinal-ruído de pico* e conhecida pela sigla PSNR (*peak signal-to-noise ratio*, em inglês), essa versão "convertida" do MSE é definida como

$$PSNR(x, y) = 10 \cdot \log_{10} \frac{L^2}{MSE(x, y)}, \tag{6}$$

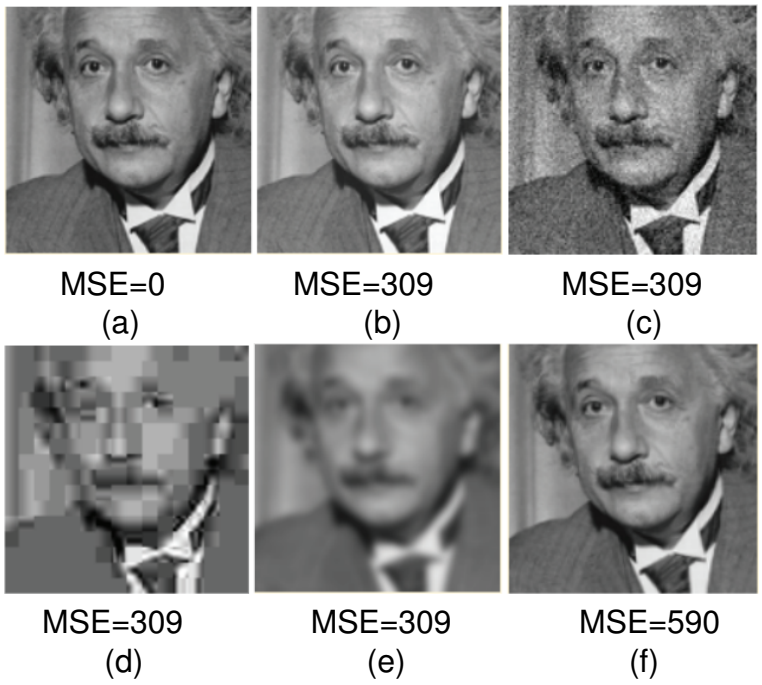
em que L representa o valor máximo alcançado pela intensidade de um pixel nas imagens (para imagens monocromáticas em 8 bits, por exemplo, $L = 255$). É importante notar que, ao contrário do que ocorre com o MSE (e ao contrário da noção intuitiva de métrica vista no início deste capítulo), quanto maior o valor PSNR entre duas imagens comparadas, mais parecidas elas são.

Wang e Bovik (2009) citam algumas razões que motivam a utilização dessa métrica para a avaliação de qualidade de imagens. Além de ter um cálculo pouco complexo, o que implica num método barato de ser implementado do ponto de vista computacional, possui propriedades muito satisfatórias de convexidade, simetria e diferenciabilidade, o que facilita a obtenção de soluções de problemas de otimização que usam o MSE. Possui ainda uma clara interpretação física, pois é a maneira natural de definir a energia do sinal de erro (WANG; BOVIK, 2009). E, finalmente, o fato do MSE ser empregado extensivamente para otimizar e avaliar uma ampla variedade de aplicações de processamento de sinal faz com que algoritmos concorrentes sejam mais frequentemente comparados com o MSE, fornecendo, portanto, um padrão conveniente e abrangente para este tipo de comparação entre métricas, economizando tempo e esforço.

Conhecendo as vantagens de utilizar essa métrica, é natural perguntar se o MSE/PSNR realmente mede a qualidade de uma imagem de maneira similar à avaliação subjetiva feita pelo SVH. A resposta a essa pergunta parece ser negativa para a maioria dos casos. Apesar de obter resultados razoavelmente bons para alguns tipos de distorções, principalmente quando a distorção é uma compressão do tipo JPEG, o MSE possui uma correlação muito baixa quando comparado à avaliação subjetiva

para a maior parte das distorções aplicadas a imagens. Resultados obtidos para vários bancos de imagens podem ser encontrados nos trabalhos de Pedersen e Hardeberg (2012) e Wang e Bovik (2009). A Figura 8 ilustra a baixa correlação entre o MSE e a avaliação subjetiva.

Figura 8 – Comparação dos índices MSE para uma imagem alterada com diferentes tipos de distorção. (a) Imagem original; (b) Mudança de luminância; (c) ruído gaussiano; (d) compressão JPEG; (e) borramento gaussiano e (f) pequena rotação no sentido anti-horário.



Fonte – Wang e Bovik (2009).

Veja que as imagens da Figura 8(b)-(e) apresentam o mesmo índice MSE apesar de serem muito diferentes em termos de qualidade visual: a imagem da Figura 8(b) possui, visualmente, muito mais qualidade do que a imagem da Figura 8(d), por exemplo. Além disso, é possível notar que o MSE é bastante sensível quando uma modificação geométrica é feita na imagem, como na Figura 8(f). Nesta, a perda de qualidade percebida é insignificante, embora seu índice MSE seja maior comparado às outras distorções que apresentam uma maior perda de qualidade.

Algumas alterações feitas em métricas já existentes podem ser úteis a fim de melhorar a correlação entre os índices calculados pela métrica e a avaliação subjetiva. Por exemplo, Pedersen e Hardeberg (2012) mostram que o índice PSNR pode ser calculado em pequenas janelas de tamanho 8×8 pixels na imagem, em vez de fazê-lo na imagem toda de uma vez. A ideia de analisar imagens em janelas (ou blocos) de

pixels, calcular a métrica nessas janelas e só depois agrupar os valores em um único índice foi explorada várias vezes na literatura e será um dos alvos desta pesquisa.

3.3 SIMILARIDADE ESTRUTURAL (SSIM)

De acordo com Wang *et al.* (2004), ao longo das últimas décadas, um grande esforço foi dedicado ao desenvolvimento de métodos de avaliação da qualidade de imagens que tiram proveito de características conhecidas do sistema visual humano. A maioria dos métodos propostos seguiu uma estratégia de modificação do índice MSE, para que os erros na imagem distorcida sejam penalizados de acordo com sua visibilidade.

Nesse sentido, uma suposição amplamente adotada é a de que a perda da qualidade de uma imagem está diretamente relacionada à visibilidade do erro (o erro entre a imagem distorcida e a imagem de referência pode ser entendendido como a subtração entre as duas). Wang *et al.* (2004) reforçam aquilo que vimos ilustrado na Figura 8: duas imagens distorcidas com o mesmo MSE podem ter muitos tipos diferentes de erros, alguns dos quais são mais visíveis que outros. Apontam, ainda, uma série de outros fatores pelos quais a ideia de penalizar a perda de qualidade de uma imagem de acordo com a visibilidade do erro não é a mais adequada quando se quer uma métrica que se assemelhe à avaliação subjetiva.

Com base nesses problemas e também no fato de que, em geral, imagens são altamente estruturadas, ou seja, seus pixels exibem fortes dependências, especialmente quando estão próximos, e essas dependências carregam informações importantes sobre a estrutura dos objetos que compõem a imagem, Wang *et al.* (2004) propuseram uma nova maneira de fazer a avaliação da qualidade de imagens. Chamada de *similaridade estrutural* e conhecida na literatura pela sigla SSIM (*structural similarity*, em inglês), essa métrica tem a finalidade de imitar a avaliação feita pelo SVH, já que este é altamente adaptado para extrair informações estruturais do campo de visualização de uma imagem.

Uma vez que existem elementos em uma imagem que não fazem parte de sua informação estrutural, o índice SSIM leva em conta a estrutura dos objetos na imagem, além de utilizar informações sobre luminância e contraste nas imagens, resultando em uma métrica que depende de três funções: luminância (l), contraste (c) e estrutura (s). Assim, considerando duas imagens x e y, temos

$$SSIM(x, y) = f(l(x, y), c(x, y), s(x, y)). \tag{7}$$

Sejam

$$\mu_x = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{8}$$

e

$$\mu_y = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{9}$$

as intensidades médias das imagens x e y, respectivamente. A média dos pixels de uma imagem pode ser interpretada como luminância de uma imagem, já que, à medida que os valores dos pixels se aproximam de 255 (branco), o valor da média dos pixels também aumenta. Da mesma forma, se temos uma imagem em que seus pixels têm valores próximos de 0 (preto), o valor da média deles também será próximo de 0. Assim, temos que a função que compara a luminância entre duas imagens usa a média dos pixels de cada imagem em seu cálculo e é definida como

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \tag{10}$$

em que C_1 é uma constante usada para evitar instabilidade quando o denominador se aproxima de zero. Essa constante pode ser escolhida como

$$C_1 = (K_1 L)^2, \tag{11}$$

sendo L definido da mesma maneira como feito para a métrica PSNR e K_1 uma constante pequena (Wang *et al.* (2004) tomam $K_1 = 0,01$).

Veja que faz sentido usar a expressão da Equação (10) como uma função que compara a luminância entre as imagens. De fato, podemos reescrever a função como

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{(\mu_x - \mu_y)^2 + 2\mu_x\mu_y + C_1}. \tag{12}$$

Primeiro, veja que

$$(\mu_x - \mu_y)^2 + 2\mu_x\mu_y + C_1 \geq 2\mu_x\mu_y + C_1. \tag{13}$$

Agora, observe que, à medida em que a diferença dos valores de μ_x e μ_y diminui, a diferença entre o numerador e o denominador da fração na Equação (12) também diminui e, portanto, o valor da função se aproxima de 1 (pela esquerda). Por outro lado, quanto maior for a diferença entre μ_x e μ_y, maior é a diferença entre o denominador e o numerador da fração e, portanto, o valor da função se aproxima de 0 (pela direita). Dessa forma, a função $l(x, y)$ atribui às imagens x e y um valor no intervalo real $(0, 1]$, no qual imagens que possuem a mesma luminância recebem 1 e imagens com luminância cada vez mais distintas recebem um valor cada vez mais próximo de 0.

Similarmente, pode-se definir a função que compara o contraste entre as duas imagens como

$$c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \tag{14}$$

em que

$$\sigma_x = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2} \tag{15}$$

e

$$\sigma_y = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(y_i - \mu_y)^2} \tag{16}$$

são os desvios padrões das imagens x e y, respectivamente, e

$$C_2 = (K_2 L)^2, \tag{17}$$

com K_2 sendo uma constante pequena (Wang *et al.* (2004) tomam $K_2 = 0,03$). Faz sentido usar o desvio padrão dos pixels de uma imagem para determinar seu constraste já que uma imagem com alto contraste possui, visualmente, mais discrepância entre regiões claras e regiões escuras. Isso implica em uma discrepância maior nos valores dos pixels também, ou seja, mais eles distam da média dos pixels e, portanto, seu desvio padrão é maior.

Para comparar as estruturas das imagens x e y, Wang *et al.* (2004) usam o *coeficiente de correlação de Person*. Dessa forma, temos que

$$\rho(x, y) = \frac{\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^{n}(x_i - \mu_x)^2} \cdot \sqrt{\sum_{i=1}^{n}(y_i - \mu_y)^2}} = \frac{\sigma_{xy}}{\sigma_x\sigma_y}, \tag{18}$$

em que

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y). \tag{19}$$

Assim, entende-se que imagens mais correlacionadas possuem estruturas mais semelhantes do que imagens pouco correlacionadas.

De maneira semelhante ao que foi feito nas funções l e c, escolhe-se adicionar uma constante $C_3 = \frac{C_2}{2}$ ao numerador e ao denominador da função ρ para evitar instabilidades. Assim, a função que compara as estruturas das imagens x e y é

$$s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}. \tag{20}$$

Finalmente, Wang *et al.* (2004) escolhem utilizar a multiplicação de funções como a função f da Equação (7) e definem a nova métrica. Portanto, das Equações

7, 10, 14 e 20, temos que a métrica SSIM entre as imagens x e y pode ser calculada como:

$$SSIM(x,y) = l(x,y) \cdot c(x,y) \cdot s(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}. \tag{21}$$

Teorema 10. *Para quaisquer $x, y \in I$ duas imagens, temos:*

1. $SSIM(x,y) = 1$ *se, e somente se,* $x = y$*;*
2. $SSIM(x,y) = 0$ *se, e somente se,* $\sigma_{xy} = -\frac{C_2}{2}$*.*

Demonstração. Sejam $x, y \in I$ duas imagens quaisquer.

1. ($\Rightarrow$) Suponha que $SSIM(x,y) = 1$. Temos que

$$l(x,y) \cdot c(x,y) \cdot s(x,y) = 1. \tag{22}$$

Pelas definições das funções l, c e s, sabemos que $0 \leq l(x,y), c(x,y) \leq 1$ e $-1 \leq s(x,y) \leq 1$. Usando esse fato e a Equação (22), temos que

$$l(x,y) = 1, \tag{23}$$

$$c(x,y) = 1 \tag{24}$$

e

$$s(x,y) = 1. \tag{25}$$

Da Equação (23), segue que

$$2\mu_x\mu_y + C_1 = \mu_x^2 + \mu_y^2 + C_1, \tag{26}$$

ou seja,

$$(\mu_x - \mu_y)^2 + 2\mu_x\mu_y = 2\mu_x\mu_y \tag{27}$$

e, portanto,

$$\mu_x = \mu_y. \tag{28}$$

Da Equação (25), temos que as imagens x e y têm a mesma estrutura, ou seja, possuem correlação perfeita. Dessa forma, existe $\alpha \in \mathbb{R}$ tal que

$$y = \alpha x. \tag{29}$$

Das Equações (28) e (29), temos que

$$\sum_{i=1}^{N} x_i = \sum_{i=1}^{N} y_i = \sum_{i=1}^{N} \alpha x_i = \alpha \sum_{i=1}^{N} x_i. \tag{30}$$

Assim, temos que $\alpha = 1$ e, portanto, $x = y$.

($\Leftarrow$) Suponha que $x = y$. Usando a Equação (21), temos que:

$$SSIM(x, y) = \frac{(2\mu_x^2 + C_1)(2\sigma_{xx} + C_2)}{(2\mu_x^2 + C_1)(2\sigma_x^2 + C_2)}. \tag{31}$$

Para sabermos o valor de σ_{xx}, usamos a Equação (19):

$$\sigma_{xx} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2. \tag{32}$$

Além disso, sabemos que:

$$\sigma_x = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2}. \tag{33}$$

Portanto, das Equações (32) e (33), segue que:

$$\sigma_{xx} = \sigma_x^2. \tag{34}$$

Das Equações (31) e (34), segue que:

$$SSIM(x, y) = \frac{(2\mu_x^2 + C_1)(2\sigma_x^2 + C_2)}{(2\mu_x^2 + C_1)(2\sigma_x^2 + C_2)} = 1, \tag{35}$$

já que $C_1 > 0$ e $C_2 > 0$.

2. ($\Rightarrow$) Suponha que $SSIM(x, y) = 0$. Da Equação (21), temos:

$$(2\mu_x\mu_y + C_1) = 0 \text{ ou} \tag{36}$$

$$(2\sigma_{xy} + C_2) = 0. \tag{37}$$

Como $\mu_x \geq 0$, $\mu_y \geq 0$ e $C_1 > 0$, a Equação (36) é um absurdo e então temos que $(2\sigma_{xy} + C_2) = 0$, ou seja,

$$\sigma_{xy} = -\frac{C_2}{2}. \tag{38}$$

($\Leftarrow$) Suponha que $\sigma_{xy} = -\frac{C_2}{2}$. Segue que $2\sigma_{xy} + C_2 = 0$. Assim,

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} = \frac{(2\mu_x\mu_y + C_1)\cdot 0}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} = 0 \tag{39}$$

□

Portanto, no caso em que $SSIM(x, y) = 0$, a correlação entre as imagens x e y é baixa, já que C_2 é uma constante pequena. Em resumo, temos que o índice SSIM possui valores que vão de 0 a 1, em que imagens iguais apresentam índice SSIM igual a 1 e imagens pouco correlacionadas resultam no índice 0.

Observe que as propriedades 1 e 4 da Definição 5 podem ser facilmente verificadas para a função SSIM (a propriedade 1 segue do fato de que o valor da intensidade de cada pixel está no intervalo de 0 a 255, como comentado no capítulo 2, ou seja, são valores não-negativos). Apesar de não valer a propriedade 2, o Teorema 3.3 mostra uma propriedade semelhante que a função possui. Por outro lado, a desigualdade triangular não se verifica, e isso pode ser observado escolhendo duas imagens $x, y \in I$ tais que $SSIM(x, y) < 0,5$. Dessa forma, temos que

$$2 \cdot SSIM(x, y) = SSIM(x, y) + SSIM(y, x) < 1 = SSIM(x, x) \tag{40}$$

Apesar disso, como já foi mencionado, continuaremos chamando a função SSIM de *métrica*, mesmo que ela se pareça mais com uma semimétrica. Outras propriedades matemáticas da métrica *SSIM* foram exploradas no trabalho de Brunet *et al.* (2012), que mostram, inclusive, que as as funções definidas nas Equações 10, 14 e 20 podem ser métricas, sob o ponto de vista conceitual.

Conforme comentado anteriormente, muitos autores fazem o cálculo de métricas já conhecidas de forma local e não global. Isso se deve ao fato de que as variáveis estatísticas de uma imagem utilizadas no cálculo não costumam ser espacialmente estacionárias, ou seja, distorções encontradas em uma imagem podem variar no espaço. Além desse fator, Wang *et al.* (2004) apresentam outros fatores que os levam a escolher a matriz

$$W = 10^{-4} \cdot \begin{bmatrix} 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 3 & 8 & 16 & 2 & 16 & 8 & 3 & 1 & 0 \\ 0 & 3 & 13 & 39 & 77 & 96 & 77 & 39 & 13 & 3 & 0 \\ 1 & 8 & 39 & 120 & 233 & 291 & 233 & 120 & 39 & 8 & 1 \\ 2 & 16 & 77 & 233 & 454 & 567 & 454 & 233 & 77 & 16 & 2 \\ 3 & 20 & 96 & 291 & 567 & 708 & 567 & 291 & 96 & 20 & 3 \\ 2 & 16 & 77 & 233 & 454 & 567 & 454 & 233 & 77 & 16 & 2 \\ 1 & 8 & 39 & 120 & 233 & 291 & 233 & 120 & 39 & 8 & 1 \\ 0 & 3 & 13 & 39 & 77 & 96 & 77 & 39 & 13 & 3 & 0 \\ 0 & 1 & 3 & 8 & 16 & 2 & 16 & 8 & 3 & 1 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{41}$$

de ponderação Gaussiana de tamanho 11×11 pixels (com desvio padrão de 1,5 pixels) para fazer os cálculos de média, desvio padrão e covariância localmente nas imagens, e não na imagem toda. Tal matriz é chamada também de *janela deslizante*, por conta de seu comportamento ao calcular as variáveis estatísticas da imagem, como se estivesse "andando" sobre a imagem toda. Para cada pixel, a janela é centrada neste pixel e então os cálculos são feitos de forma local com os 120 pixels da sua vizinhança. Assim, cada W_i carrega a informação da ponderação do pixel que se encontra na posição i. Dessa maneira, as variáveis estatísticas locais usadas para o cálculo do índice SSIM são calculadas como

$$\mu_x = \sum_{i=1}^{N} W_i x_i, \tag{42}$$

$$\sigma_x = \sqrt{\sum_{i=1}^{N} W_i (x_i - \mu_x)^2} \tag{43}$$

e

$$\sigma_{xy} = \sum_{i=1}^{N} W_i (x_i - \mu_x)(y_i - \mu_y), \tag{44}$$

em que $N = 121$. Dessa forma, obtem-se vários subíndices SSIM locais, usando a fórmula da Equação (21).

Assim, por exemplo, para o cálculo da intensidade média dos pixels dentro de uma janela, usa-se não uma média simples dos 121 pixels, mas sim uma média ponderada de acordo com a distribuição gaussiana, na qual os pixels mais ao centro têm peso maior enquanto os pixels periféricos tem menos peso.

A Figura 9 mostra parte de uma imagem ampliada, duas janelas gaussianas referentes a dois pixels e ilustra a ponderação gaussiana dentro de uma janela.

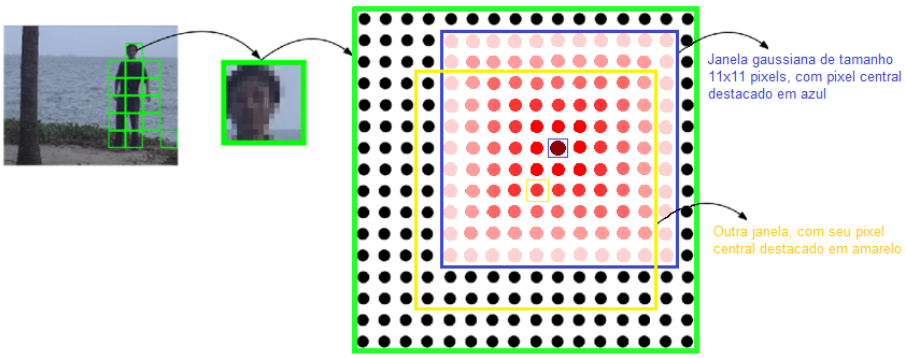
Por último, após calcular vários subíndices SSIM locais, agrupam-se tais índices para ter um índice SSIM global:

$$MSSIM(x, y) = \frac{1}{M} \sum_{j=1}^{M} SSIM(x_j, y_j) \tag{45}$$

em que x_j e y_j representam as "subimagens" encontradas na j-ésima janela das imagens x e y, respectivamente, e M representa o número total de janelas. Como, em geral, usa-se sempre o índice MSSIM, chamaremos este índice simplesmente de SSIM.

Note que distorções que são aplicadas em regiões muito próximas às bordas da imagem têm pouca influência no resultado do cálculo do índice SSIM, já que os pixels que se encontram a uma distância de até 11 pixels da borda nunca serão pixels centrais

Figura 9 – Imagem ampliada e duas janelas gaussianas referentes a dois pixels. A ponderação gaussiana em uma das janelas é ilustrada pela intensidades das cores: os pixels mais ao centro têm peso maior enquanto os pixels periféricos têm menos peso.



Fonte – Xiao e Liu (2016). Modificada pelo autor (2020).

e não terão suas próprias janelas. Esses pixels, quando afetados por distorções, só aparecem no cálculo como pixels com ponderação baixa.

3.3.1 Modificações na métrica SSIM

A ideia de comparar imagens através de sua informação estrutural, como faz a métrica SSIM, vem recebendo considerável atenção desde quando foi proposta e parece ser bastante promissora até o momento. Apesar disso, suas limitações ainda não são totalmente conhecidas e vêm sendo exploradas cada vez mais.

Os resultados experimentais obtidos por Pedersen e Hardeberg (2012) nos mostram que, em geral, o índice SSIM apresenta uma correlação melhor com a avaliação subjetiva do que o MSE/PSNR. Comparam, ainda, os resultados dessas duas métricas com outras métricas pouco usadas, mas que podem ser úteis em contextos específicos e, por fim, concluem que nenhuma das métricas investigadas possui um bom desempenho em todos os bancos de imagens estudados, com todas as distorções estudadas, apesar de algumas métricas terem melhor desempenho do que outras para distorções específicas.

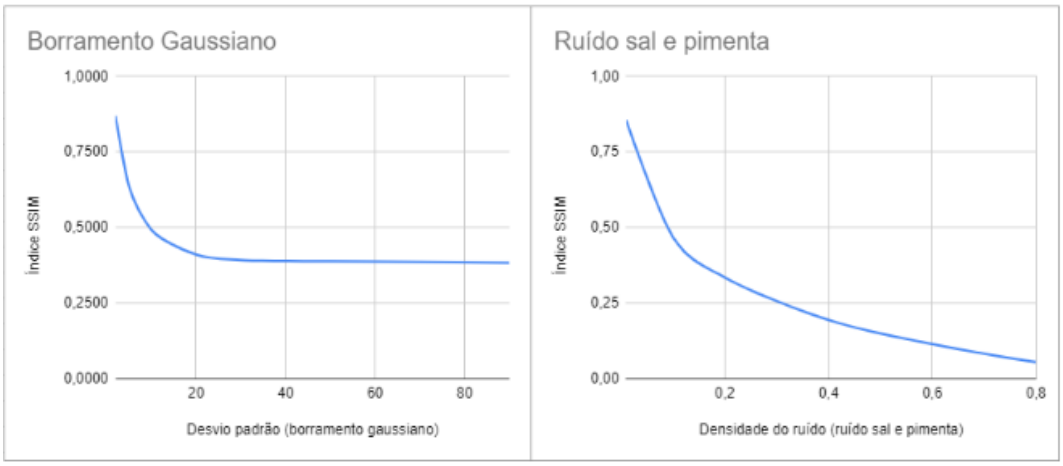
Chen *et al.* (2006) discutem especificamente um problema que o índice SSIM apresenta com relação a comparar imagens nas quais foram aplicadas o borramento gaussiano. As imagens da Figura 10 ilustram esse problema apresentando uma imagem distorcida com ruído e com borramento gaussiano. Primeiro, podemos perceber um problema já comentado, na seção anterior, sobre o índice MSE, já que as duas imagens distorcidas têm o mesmo MSE com relação à imagem original, mas claramente a imagem com ruído possui visualmente mais qualidade do que a imagem borrada. Além disso, os valores SSIM das imagens distorcidas mostram um resultado oposto

Figura 10 – Comparação de imagens com diferentes distorções, todas com MSE=1150. (a) Imagem original; (b) imagem com ruído gaussiano, SSIM=0,2591; (c) imagem com borramento gaussiano, SSIM=0,5114.

Fonte – Chen *et al.* (2006)

à avaliação visual: como podemos ver, a imagem borrada tem um valor SSIM maior do que a imagem com ruído, o que significa que, segundo a métrica SSIM, a imagem borrada tem mais qualidade do que a imagem com ruído.

Figura 11 – Gráficos que mostram o comportamento do índice SSIM quando as distorções são feitas gradualmente em uma imagem. (a) Gráfico para o borramento gaussiano; (b) gráfico para o ruído do tipo sal e pimenta.



(a) (b)

Fonte – Elaborada pelo autor (2020).

O fenômeno que ocorre nos valores SSIM mostrados na Figura 10 pode ser melhor explorado quando aumentamos gradualmente as distorções aplicadas às imagens, como mostram os gráficos da Figura 11. Podemos observar que, no borramento gaussiano, para desvios padrões maiores do que 20, o índice SSIM se mantém pouco alterado, mesmo que o aumento do valor do desvio padrão implique maior perda de qualidade visual. Para entender esse comportamento do índice SSIM quando uma

imagem é contaminada por borramento gaussiano, podemos analisar separadamente cada variável estatística que compõe o índice. Para isso, seja a imagem digital

$$x = \begin{bmatrix} 235 & 111 & 66 & 75 & 66 \\ 109 & 28 & 153 & 81 & 204 \\ 47 & 65 & 181 & 108 & 7 \\ 230 & 104 & 56 & 129 & 236 \\ 249 & 151 & 29 & 21 & 186 \end{bmatrix}$$

e as imagens

$$y_1 = \begin{bmatrix} 160.7562 & 151.8924 & 143.4352 & 135.7266 & 129.0227 \\ 162.6905 & 154.4664 & 146.7239 & 139.7826 & 133.8710 \\ 165.2095 & 157.5376 & 150.4271 & 144.1795 & 138.9975 \\ 168.2243 & 160.9948 & 154.4089 & 148.7541 & 144.2111 \\ 171.5881 & 164.6819 & 158.5005 & 153.3220 & 149.3096 \end{bmatrix},$$

$$y_2 = \begin{bmatrix} 174.5852 & 171.4835 & 168.4041 & 165.3609 & 162.3670 \\ 176.0599 & 173.0384 & 170.0406 & 167.0796 & 164.1685 \\ 177.5585 & 174.6163 & 171.6988 & 168.8192 & 165.9899 \\ 179.0744 & 176.2101 & 173.3717 & 170.5719 & 167.8230 \\ 180.6009 & 177.8126 & 175.0516 & 172.3300 & 169.6598 \end{bmatrix}$$

e

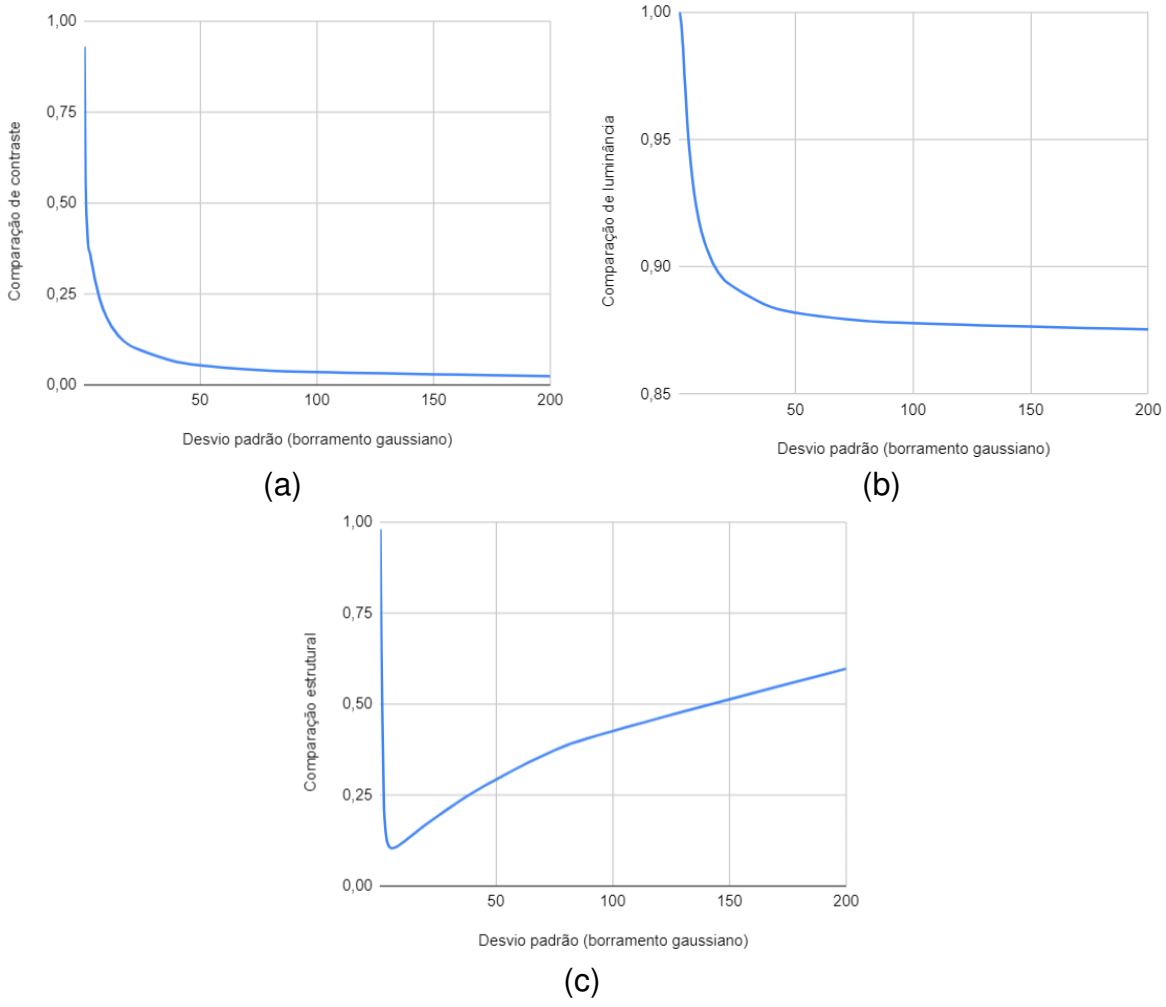
$$y_3 = \begin{bmatrix} 180.2984 & 179.1075 & 177.9179 & 176.7304 & 175.5457 \\ 180.9453 & 179.7659 & 178.5879 & 177.4119 & 176.2388 \\ 181.5935 & 180.4256 & 179.2591 & 178.0947 & 176.9330 \\ 182.2426 & 181.0862 & 179.9312 & 178.7782 & 177.6281 \\ 182.8922 & 181.7472 & 180.6036 & 179.4622 & 178.3235 \end{bmatrix}$$

distorções da imagem x com borramento gaussiano usando desvios padrões de 5, 15 e 40 pixels, respectivamente. Temos que $\mu_{y_1} = 151.2$, $\mu_{y_1} = 171.7$ e $\mu_{y_1} = 179.2$, ou seja, há uma estabilização no valor da média dos pixels a medida que borramos mais a imagem. Isso pode ser melhor visualizado quando analisamos a componente de luminância do índice SSIM, que leva em consideração somente os valores das médias dos pixels, como definido na Equação (10). O gráfico da Figura 12(a) ilustra esse comportamento. Veja que o valor da função l se mantem pouco alterado à medida em que se aumenta o borramento. O mesmo ocorre com os desvios padrões: temos que $\sigma_{y_1} = 9.3$, $\sigma_{y_1} = 3.9$ e $\sigma_{y_1} = 1.7$ e essa estabilização é ilustrada no gráfico da Figura 12(b), que mostra a variação da compenente de contraste do índice SSIM (componente que usa os desvios padrões em seu cálculo). O gráfico da Figura 12(c) mostra que o mesmo não ocorre com a componente estrutural da função SSIM, que

leva em considerção os valores de covariância e desvio padrão. Mas a multiplicação de funções usada para compor o índice SSIM (Equação (21)) implica na estabilização do índice a medida que o borramento aumenta. Já na contaminação por ruído, uma correlação um pouco maior entre o aumento da distorção o e a diminuição do valor SSIM é percebida.

A Figura 13 ilustra a perda de qualidade visual quando tais distorções são aplicadas gradualmente em uma imagem. Para ter uma melhor descrição de como as distorções de borramento gaussiano e ruído gaussiano são aplicadas às imagens, consultar o Apêndice B.

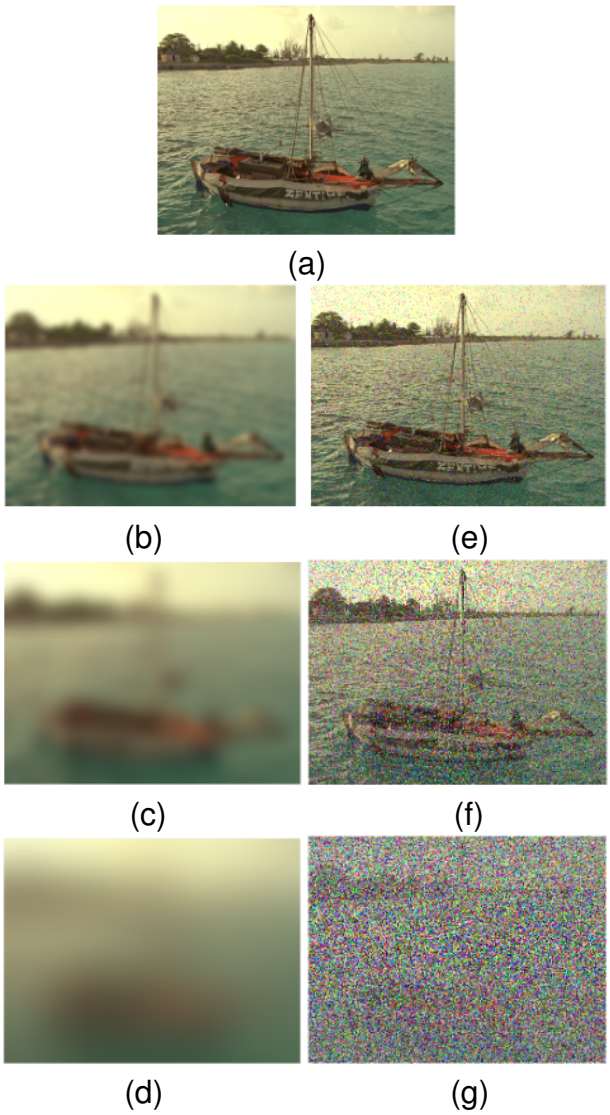
Figura 12 – Gráficos que mostram o comportamento de cada componente do índice SSIM quando usada para comparar uma imagem de referência com uma imagem contaminada por ruído gaussiano. (a) Componente de luminância; (b) componente de contraste e (c) componente estrutural.



Fonte – Elaborada pelo autor (2021)

Além disso, podemos ver em (PAMBRUN; NOUMEIR, 2015) outros problemas apresentados pela métrica SSIM especificamente em imagens usadas para diagnóstico

Figura 13 – Imagem distorcida gradualmente. (a) Imagem original; (b)-(d) borramento gaussiano com desvios padrões de 5, 15 e 40 pixels, respectivamente; (e)-(g) ruído sal e pimenta com densidades 0,05, 0,3 e 0,8, respectivamente.



Fonte – Ponomarenko *et al.* (2009)

médico. Na medicina, a quantidade de imagens geradas por dispositivos de diagnóstico vem aumentando cada vez mais e por isso se faz necessária a compressão destes arquivos. Por este motivo, métricas para avaliar a qualidade das imagens compactadas podem ser importantes neste contexto. Nesse sentido, um dos problemas encontrados na métrica SSIM diz respeito ao fato de que imagens que possuem arestas muito acentuadas, ou seja, mudanças bruscas de contrastes estão presentes na imagem, saturam as componentes $c(x, y)$ e $l(x, y)$ do índice SSIM, o que resulta em uma superestimação não desejada deste índice. Assim, Pambrun e Noumeir (2015) concluem

que, para imagens médicas, essa métrica não é apropriada, já que essas imagens costumam ter tal característica das arestas acentuadas.

Diante desses e outros problemas apresentados pela métrica de similaridade estrutural, vários estudos que propõem modificações no cálculo da métrica vêm sendo propostos com a finalidade de melhorar seu desempenho.

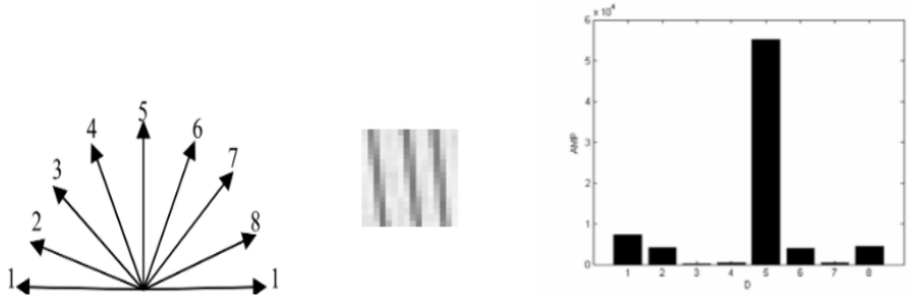
Chen *et al.* (2006) e Zhang *et al.* (2013), por exemplo, falam sobre analisar a similaridade estrutural de duas imagens baseada em suas arestas. Segundo eles, alguns estudos já apontaram para o fato de que o olho humano é muito sensível às informações de arestas e contornos de uma imagem, fazendo com que estas sejam uma das informações mais importantes da estrutura de uma imagem para o ser humano "capturar" a cena. Com base nesse pensamento, surge a proposta de substituir a comparação da estrutura $s(x, y)$, no índice SSIM, pela comparação da estrutura baseada em arestas $e(x, y)$ (CHEN *et al.*, 2006).

Em resumo, uma máscara do operador Sobel (como pode ser visto no Apêndice B) é utilizada sobre cada pixel das imagens para obter a informação das arestas das imagens. Para cada pixel de coordenadas (i, j), é definido um vetor da forma

$$\vec{D}_{i,j} = (d_{x_{i,j}}, d_{y_{i,j}}) \tag{46}$$

em que d_x e d_y são obtidos usando máscaras do operador Sobel vertical e horizontal, respectivamente. Para cada imagem, os vetores $\vec{D}_{i,j}$ são agrupados de acordo com sua direção, de acordo com a Figura 14(a) e, para cada direção, a amplitude dos vetores é somada formando um histograma como na figura Figura 14(c).

Figura 14 – (a) Oito direções discretas pelas quais os vetores são agrupados; (b) imagem 16×16 pixels e seu (c) histograma de vetores em que D representa as direções e AMP as amplitudes somadas.



(a) (b) (c)

Fonte – Chen *et al.* (2006)

É com base nas variáveis estatísticas desses histogramas, um para a imagem original e outra para a imagem distorcida, que a função $e(x, y)$ é obtida. Calculando

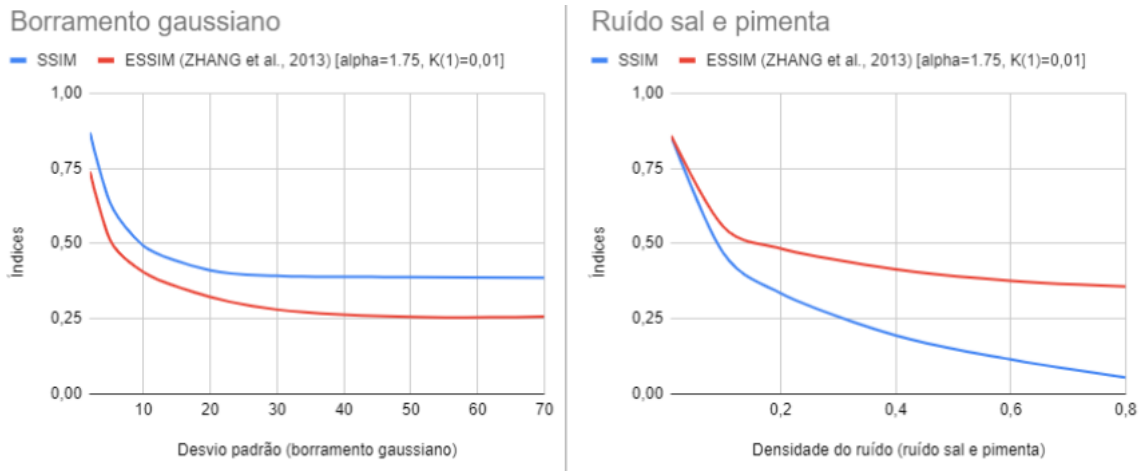
seus desvios padrões, σ'_x e σ'_y, respectivamente, e σ'_{xy} a correlação como na Equação (20), temos que

$$e(x, y) = \frac{\sigma'_{xy} + C_3}{\sigma'_x \sigma'_y + C_3} \tag{47}$$

semelhante ao que foi feito na Equação (20).

Nesse mesmo sentido, também inspirado na similaridade estrutural, Zhang *et al.* (2013) propõem um algoritmo que compara as imagens de acordo com as informações de suas arestas, de maneira muito similar à Equação (47), porém sem levar em consideração as funções $l(x, y)$ e $c(x, y)$. Ambas as métricas são chamadas de *similaridade estrutural baseada em arestas* ou ESSIM (*edge-based structural similarity*, em inglês).

Figura 15 – Gráficos que mostram o comportamento dos índices SSIM e ESSIM (ZHANG et al., 2013) quando as distorções são feitas gradualmente em uma imagem. (a) Gráfico para o borramento gaussiano; (b) gráfico para o ruído do tipo sal e pimenta.



(a) (b)

Fonte – Elaborada pelo autor (2020).

Como podemos ver na Figura 15, a ideia de comparar duas imagens de acordo com suas arestas parece não resolver totalmente os problemas apresentados pela métrica SSIM. Para o cálculo dos índices com uma imagem borrada, vemos que o comportamento do métrica SSIM parece ser o mesmo da métrica ESSIM, ou seja, os valores se mantêm pouco alterados mesmo que o aumento do valor do desvio padrão implique maior perda de qualidade visual. Além disso, para a adição de ruído, o comportamento dos índices ESSIM parece ser pior do que os valores SSIM: a boa correlação entre o índice e a avaliação subjetiva, que observamos com SSIM, não é mais observada ao utilizar ESSIM.

Com base nos problemas de baixa semelhança entre o índice SSIM e a avaliação subjetiva do SVH apresentadas neste capítulo e nas tentativas de modificação da métrica SSIM, estaremos interessados em aproveitar a potencialidade que esta métrica pode oferecer, investigando pontos de melhoria ainda não explorados e modificando seu cálculo com a finalidade de aumentar a semelhança entre os valores calculados pela métrica e a avaliação humana.

4 ALTERAÇÕES NO TAMANHO E NO FORMATO DA JANELA DESLIZANTE

Após o estudo dos conceitos fundamentais necessários, descritos nos capítulos anteriores, passaremos a explorar maneiras para melhorar a eficácia da métrica SSIM na avaliação da qualidade visual de imagens, fazendo alterações no tamanho e no formato da janela deslizante (matriz W da Equação (41)) usada para fazer os cálculos da métrica. É importante dizer que, ao nos referirmos a "alterações no formato" da janela deslizante, estamos querendo dizer que a ponderação de cada pixel dentro da janela será alterada, ou seja, não será mais a distribuição gaussiana (ponderação padrão).

Neste capítulo, serão apresentados os procedimentos realizados e resultados obtidos nas primeiras análises, juntamente com discussões a respeito destes resultados.

4.1 PROCEDIMENTOS

Utilizamos o banco de imagens de Ponomarenko *et al.* (2009), que possui 25 imagens com diferentes características que têm a intenção de representar, de forma geral, os mais variados tipos de cenas que podem ser observadas pelo sistema visual humano. As imagens possuem resoluções espaciais de 512×384 pixels. Além disso, cada imagem foi submetida a 17 tipos de filtros (com 4 níveis de distorção em cada filtro), que representam as diferentes distorções que uma imagem pode sofrer no cotidiano. É importante salientar que cada imagem distorcida vem acompanhada de uma determinada pontuação, chamada pontuação MOS (*Mean Opinion Score*, em inglês), dada por indivíduos reais, em ambientes controlados, que a compararam com a imagem original. O índice MOS é conhecido como uma métrica "de fato", usada para quantificar a qualidade de sinais e comumente usada para comparação com resultados obtidos no cálculo com métricas objetivas. Para imagens, um número de observadores é escolhido (em geral, mais do que 16) e, em um ambiente controlado e com uma distância adequada entre o observador e a imagem, são dadas pontuações para a qualidade visual da imagem distorcida em comparação com a referência. São dadas notas de 1 a 5 (de "péssima" a "excelente") para a imagem distorcida e, em seguida, faz-se uma média das pontuações. A Figura 16 mostra uma imagem filtrada de duas maneiras diferentes e suas respectivas pontuações MOS. Streijl *et al.* (2014) mostram mais detalhes de como o método MOS é aplicado.

Por sua facilidade de implementação de códigos e velocidade na execução, será utilizado o software MATLAB na execução das modificações que serão implementadas. Para a métrica SSIM, será utilizado um código de implementação desenvolvido por Wang *et al.* (2004) e disponível em: `https://ece.uwaterloo.ca/~z70wang/research/ssim/`.

Figura 16 – Índices MOS para imagem com duas distorções: (a) imagem de referência, (b) adição de ruído gaussiano e (c) de borramento gaussiano.

MOS=5 (a) | MOS=2,3 (b) | MOS=1,2 (c)

Fonte – Ponomarenko et al. (2009).

Além disso, é necessário utilizar alguma ferramenta estatística para fazer a comparação entre as pontuações subjetivas do índice MOS com as pontuações obtidas através de cada código. Para cada métrica, dados os índices calculados por ela e os valores MOS associados às imagens distorcidas, usaremos o coeficiente estatístico de *correlação de Pearson*, indicado pela letra grega ρ, para medir a correlação entre os dois índices. Logo, quanto maior for o ρ entre os valores MOS e os valores da métrica, maior o desempenho desta. Lembrando que, chamando de $a = (a_1, \ldots, a_n)$ os índices da métrica e $b = (b_1, \ldots, b_n)$ os índices MOS, temos que

$$\rho(a,b) = \frac{\sum_{i=1}^{n}(a_i - \mu_a)(b_i - \mu_b)}{\sqrt{\sum_{i=1}^{n}(a_i - \mu_a)^2} \cdot \sqrt{\sum_{i=1}^{n}(b_i - \mu_b)^2}} = \frac{\sigma_{ab}}{\sigma_a \sigma_b}. \tag{48}$$

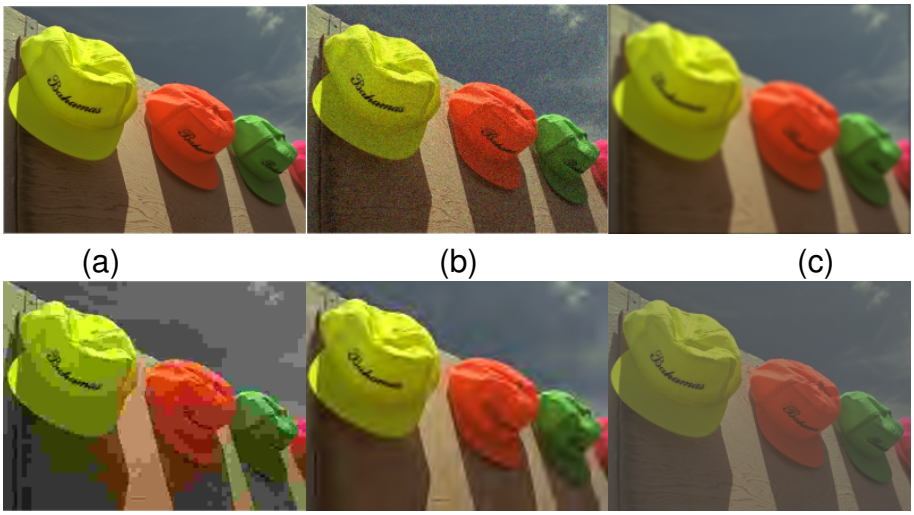
Com base na pesquisa de Golestani e Ghambari (2014), a qual aponta que o tamanho 11×11 (tamanho padrão sugerido por Wang *et al.* (2004)) pode não ser o mais adequado para fazer o cálculo do índice SSIM local com a janela gaussiana, iremos estudar como a variação de parâmetros como o tamanho e o formato da janela deslizante pode afetar o cálculo da métrica, aumentando o coeficiente de correlação de Pearson entre os índides SSIM e MOS. Estudaremos especificamente os pontos descritos a seguir:

- O tamanho da janela deslizante, com variações de 3×3 pixels até 125×125 pixels (GOLESTANI; GHAMBARI, 2014);
- O formato da janela deslizante: análise do desempenho de alguns formatos pré-selecionados disponíveis no software MATLAB, como “quadrado” e “disco”.

4.2 O TAMANHO DA JANELA DESLIZANTE

Como vimos no capítulo anterior, Wang *et al.* (2004) propõem que a métrica SSIM seja calculada de forma local nas imagens, com uma janela de ponderação gaussiana de tamanho 11×11 pixels. Com a intenção de melhorar a correlação (ρ) entre os valores SSIM e as pontuações MOS, Golestani e Ghambari (2014) apontam que o tamanho 11×11 pode não ser o mais adequado para fazer o cálculo local com a janela gaussiana. Resultados indicam que, para algumas distorções, como o filtro que altera o contraste de uma imagem, a correlação (ρ) entre SSIM e MOS é inferior a 0,65. A forma geral do índice SSIM possui alguns parâmetros livres que podem ser ajustados, como as constantes K_1 e K_2 nas Equações (11) e (17), mas Golestani e Ghambari (2014) mostram que a alteração de outros fatores inerentes e ocultos na métrica, como o tamanho da janela gaussiana local da SSIM, pode melhorar a correlação ainda mais.

Figura 17 – Cinco tipos de distorções estudadas por Golestani e Ghanbari (2014). (a) Imagem original (b) adição de ruído gaussiano; (c) borramento gaussiano; (d) compressão JPEG; (e) compressão JPEG2000; (f) mudança de contraste.

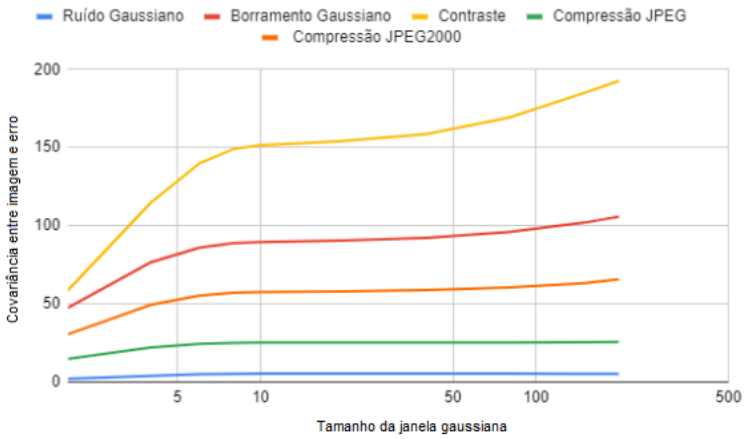


Fonte – Ponomarenko et al. (2009).

Neste estudo, cinco distorções foram analisadas: adição de ruído gaussiano, borramento gaussiano, compressão do tipo JPEG, compressão do tipo JPEG2000 e mudança de contraste, como ilustra a Figura 17. Percebe-se que o tipo de filtro aplicado à imagem tem forte influência sobre o tamanho que a janela gaussiana deve ter para obter um ρ maior quando comparado com o índice MOS. Mais especificamente, Golestani e Ghambari (2014) analisam não a imagem distorcida, mas o erro entre a imagem original e a distorção. Tal erro é calculado através da subtração entre as

duas imagens. Com base nisso, é possível calcular a covariância entre a imagem original e o erro produzido pela distorção, usando a fórmula da Equação (44), na qual a covariância é calculada de forma local através das janelas gaussianas. Finalmente, podemos classificar cada filtro com base nessa covariância. O gráfico da Figura 18 mostra que o filtro que altera o contraste possui maior covariância, enquanto a adição de ruído gaussiano possui menor, independente do tamanho da janela gaussiana utilizada para fazer os cálculos.

Figura 18 – Gráfico que mostra a covariância entre a imagem e o erro causado por diferentes distorções para vários tamanhos usados na janela gaussiana.



Fonte – Elaborada pelo autor (2020).

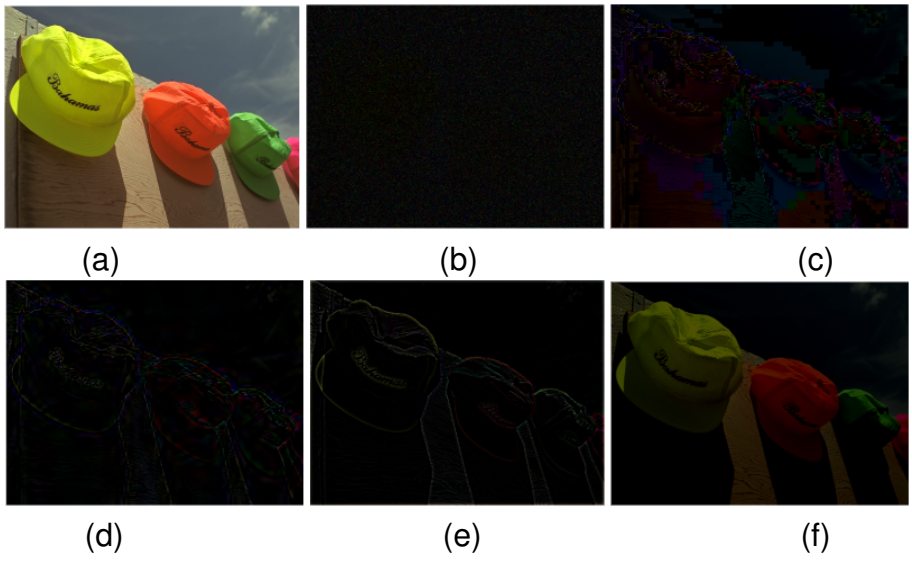
A covariância entre imagem e erro, citada por Golestani e Ghanbari (2014), pode ser visualizada na Figura 19, que mostra o erro entre a imagem original e as cinco distorções aplicadas à imagem. Em linha com a Figura 18, tais filtros se encontram em ordem crescente: da menor para a maior covariância entre imagem e erro.

Dito isso, Golestani e Ghambari (2014) concluem que, para distorções em que a covariância mostrada na Figura 18 é maior, tamanhos maiores de janelas melhoram o desempenho do índice SSIM, enquanto janelas menores são mais indicadas para distorções nas quais tal covariância é menor. Os gráficos da Figura 20 ilustram esse fato, indicando ainda qual tamanho é o ideal para cada um dos cinco filtros.

Perceba que o tamanho padrão para a janela gaussiana (11×11) não é o ideal para nenhum dos cinco filtros estudados, mas possui bom desempenho para as compressões, o que não ocorre com os outros três filtros, já que apresentam ρ significativamente maior para janelas gaussianas de outros tamanhos.

Ainda não estão totalmente claros os detalhes dessa relação entre: (i) qual o tamanho ideal de janela gaussiana para cada distorção e (ii) o erro entre imagem e distorção. O que percebe-se até agora, empiricamente, é que a relação entre (i) e (ii) parece existir, como exposto acima e visto nas três últimas figuras. Os motivos que

Figura 19 – Erro entre imagem e cinco distorções diferentes em ordem crescente: da menor para a maior correlação entre imagem e erro. (a) Imagem original; (b) ruído gaussiano; (c) compressão JPEG; (d) compressão JPEG2000; (e) borramento gaussiano; (f) mudança de contraste.

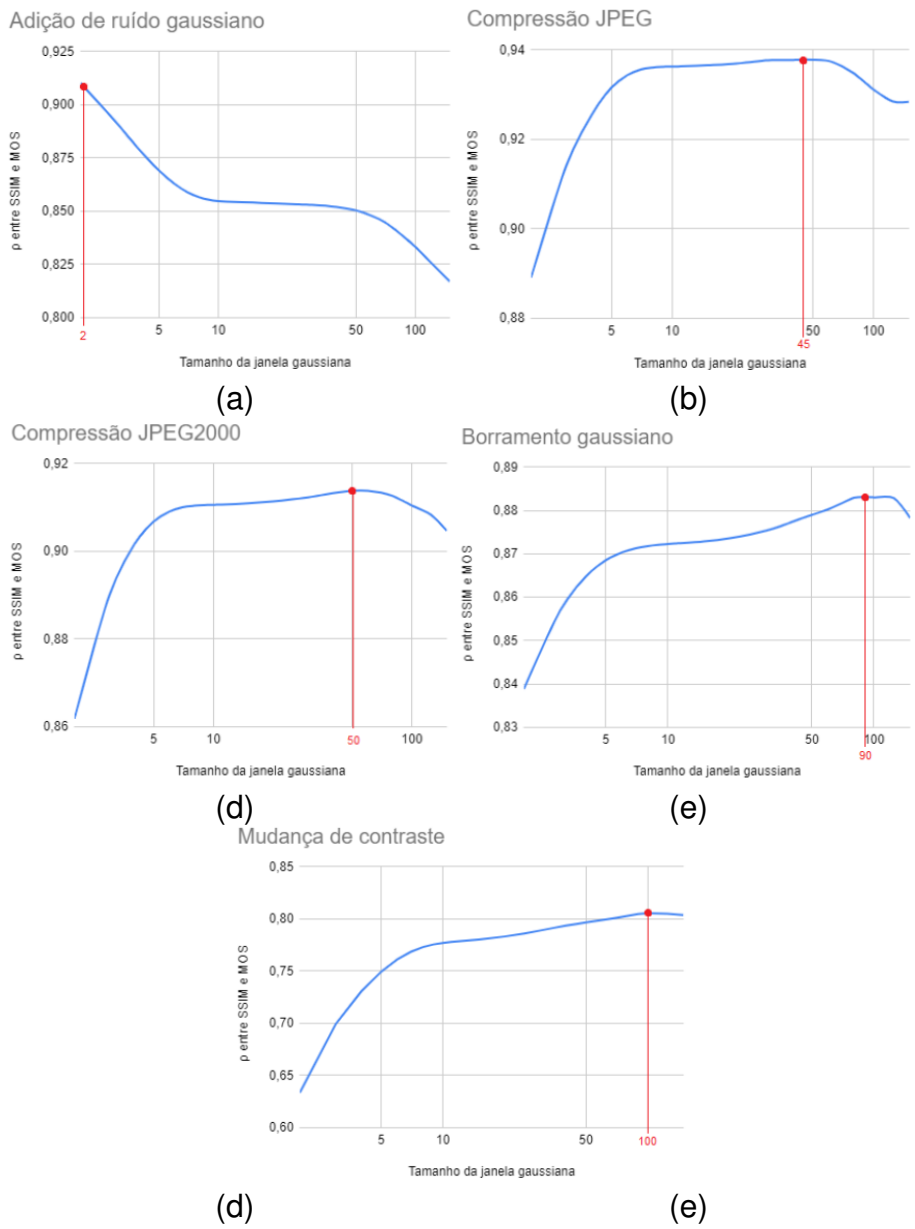


Fonte – Ponomarenko *et al.* (2009)

levam janelas gaussianas maiores terem melhor desempenho para distorções cuja covariância entre imagem e erro é maior poderão ser explorados em trabalhos futuros.

Apesar disso, é natural pensar que, em geral, procura-se por uma métrica com o menor número possível de parâmetros a serem escolhidos ao serem feitos os cálculos. Como vimos anteriormente, Wang et al. (2004) sugerem possíveis valores para as constantes K_1 e K_2, além de apontar a janela gaussiana 11×11 como padrão na implementação da métrica. Da maneira como expomos anteriormente, o tamanho da janela gaussiana poderia ser um novo parâmetro a ser escolhido, na qual janelas maiores são mais indicadas para algumas situações enquanto janelas menores são indicadas para outras. Mas não podemos deixar de observar que o estudo feito por Golestani e Ghambari (2014) levou em consideração somente alguns tipos específicos de distorções aplicadas às imagens e, com base nelas, apontou qual tamanho de janela é melhor. Na prática, podemos estar interessados em usar a métrica para avaliar a qualidade de imagens que foram submetidas a vários tipos diferentes de filtros que não são, necessariamente, nenhum dos cinco estudados e, assim, poderíamos não saber se a correlação entre imagem e erro é alta ou baixa e, por consequência, não saberíamos se é melhor usarmos janelas maiores ou menores. Mesmo se tivéssemos tais informações, pode-se tornar muito caro computacionalmente alterar o tamanho da janela gaussiana toda vez que a imagem for submetida a um filtro diferente. Logo,

Figura 20 – Gráficos que mostram o coeficiente de correlação de Pearson (ρ) entre SSIM e MOS de cinco distorções variando para tamanhos diferentes de janela gaussiana no cálculo do índice SSIM.



Fonte – Elaborada pelo autor (2020).

torna-se natural otimizar o tamanho da janela, procurando por aquele que sirva de maneira melhor possível para vários tipos de filtros ao mesmo tempo.

Considerando que o número de imagens e distorções presentes nos bancos de imagens é bastante expressivo e pode representar, de forma geral, as várias distorções que uma imagem pode ser submetida na prática, geramos 17 curvas (uma para cada tipo de distorção) que mostram como a correlação entre SSIM e MOS varia de acordo com o tamanho de janela utilizado. Foi calculada uma média das 17 curvas obtidas e

o resultado pode ser visto no gráfico da Figura 21. O pico da curva ocorre no tamanho 3×3. Além disso, nota-se que todos os tamanhos de janela menores que 11×11 (padrão adotado para SSIM) apresentam ρ maior do que o tamanho padrão.

Assim, até o momento, observa-se que adotar 3×3 como o tamanho da janela gaussiana utilizada no cálculo da métrica SSIM parece melhorar o desempenho desta, mas não de maneira muito significativa.

Figura 21 – Média das curvas que mostram o comportamento do valor de ρ entre SSIM e MOS (para 17 distorções diferentes) variando conforme o tamanho da janela gaussiana da métrica SSIM.

Fonte – Elaborada pelo autor (2020).

4.3 O FORMATO DA JANELA DESLIZANTE

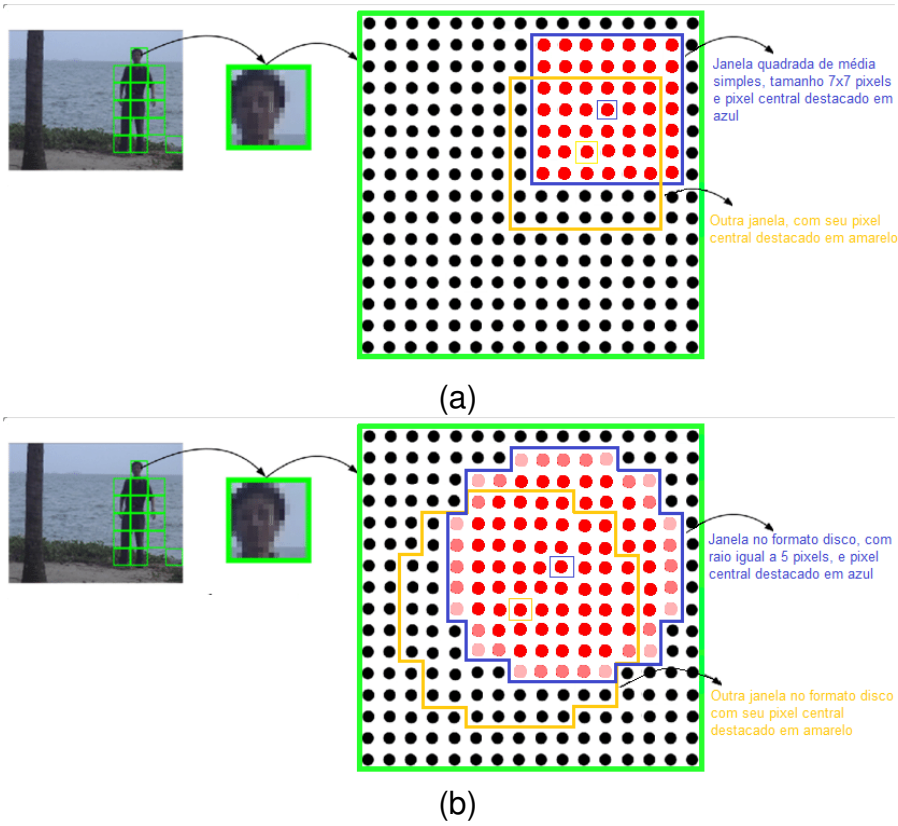
Como comentado, a ideia de ajustar parâmetros dentro do cálculo de uma métrica tem como finalidade principal aumentar a correlação (ρ) entre os valores dados pela métrica e os valores associados à opinião popular (índice MOS). Na métrica SSIM, assim como o tamanho da janela usada para o cálculo dos valores, uma outra característica dessa janela que pode ser ajustada como um parâmetro é o seu formato. A janela quadrada com ponderação gaussiana, como mostra a Figura 9, é sugerida como padrão por Wang *et al.* (2004), mas considerar essa característica como um novo parâmetro a ser estudado pode abrir possibilidades para que o desempenho da métrica possa ser melhorado. Lembrando que estamos entendendo a ponderação da janela como o "formato" da janela, ou seja, janelas com formatos diferentes atribuem diferentes pesos a um mesmo pixel.

Como vimos anteriormente, fazer o cálculo da métrica de forma local, e não global, se justifica pelo fato das variáveis estatísticas da imagem variarem no seu conjunto suporte. Para isso, sobre cada pixel, considera-se uma região a sua volta (chamada aqui de "janela deslizante") onde os cálculos das variáveis estatísticas são

realizados. A Figura 22 ilustra dois formatos simples que podem ser adotados para a janela no cálculo da SSIM.

Na Figura 22(a), observa-se uma janela quadrada, semelhante à utilizada como padrão na SSIM, mas sem considerar a ponderação gaussiana, ou seja, cada pixel dentro da janela possui o mesmo peso para fim dos cálculos das variáveis estatísticas utilizadas na métrica. Já na Figura 22(b), temos uma janela em forma de disco, com raio variável, onde cada pixel pertencente ao disco também possui peso igual, com exceção dos pixels das bordas, que possuem peso menor para se aproximar, de forma mais fiel, à discretização de um disco contínuo.

Figura 22 – Imagem ampliada e duas janelas referentes à dois pixels. (a) Janela no formato quadrado com média simples; (b) janela no formato disco.

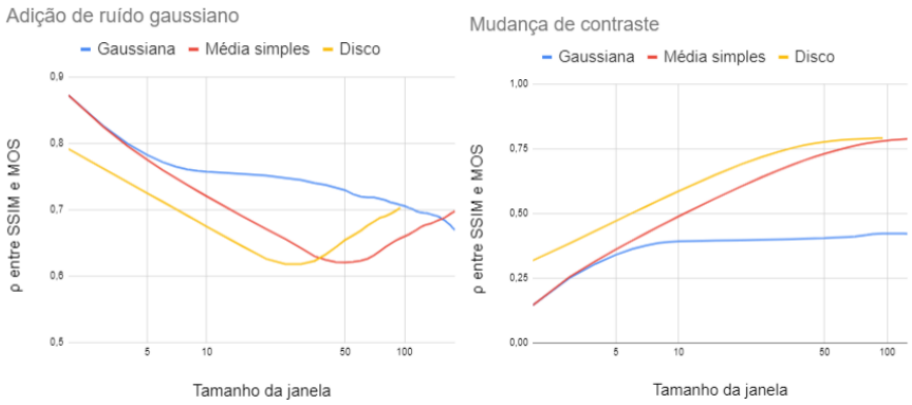


Fonte – Xiao e Liu (2016). Modificada pelo autor (2020).

Utilizando mais uma vez os bancos de imagens, foram feitos cálculos dos valores de ρ entre os valores SSIM e MOS. Fixando cada um dos dois novos formatos de janelas apresentados acima, com seu tamanho variando novamente, observou-se, em geral, um comportamento parecido com o comportamento visto para a janela gaussiana. Assim, viu-se que para distorções em que a correlação entre imagem e erro é baixa (adição de ruído gaussiano, por exemplo), a SSIM com janelas de tamanhos menores tem melhor desempenho, enquanto para distorções nas quais a correlação

de imagem e erro é alta (alteração de contraste, por exemplo) a SSIM tem melhor desempenho com janelas maiores. Esse fato é ilustrado pelos gráficos da Figura 23.

Figura 23 – Gráficos que mostram o coeficiente de correlação de Pearson (ρ) entre SSIM e MOS de duas distorções em função dos tamanhos de janela com três formatos diferentes.



Fonte – Elaborada pelo autor (2020).

Figura 24 – Médias das curvas que mostram o valor de ρ entre SSIM e MOS em função do tamanho da janela com três formatos diferentes.

Fonte – Elaborada pelo autor (2020).

Além disso, da mesma maneira como feito na Figura 21, calculou-se uma média dos valores de ρ obtidos para cada uma das distorções, em cada um dos dois novos formatos de janela, e o resultado é visto na figura Figura 24. É possível observar que,

assim como na janela gaussiana, tamanhos menores de janela apresentam desempenho melhor para os formatos de “média simples” e “disco”. Mais especificamente, a melhor correlação vista para a janela com formato de disco ocorre para o disco com raio de 2 pixels. Já para o formato de média simples, temos o pico no tamanho 3×3, assim como na janela gaussiana. Observa-se ainda que, para todos os tamanhos de janela, o formato de janela gaussiana possui ρ mais alto do que os outros dois formatos, com exceção do tamanho 3×3, no qual o formato de média simples tem $\rho = 0,7984$, enquanto no formato gaussiano obtemos $\rho = 0,7982$.

4.4 NOVOS DESENVOLVIMENTOS

Com base nos resultados obtidos até o momento, observamos que não há ganhos significativos ao alterar o formato e a ponderação da janela deslizante utilizada no cálculo do índice SSIM usando outros formatos pré-selecionados disponíveis no MATLAB, o que poderia justificar a escolha de Wang *et al.* (2004) pela janela quadrada de ponderação gaussiana.

A partir deste momento, pensamos em utilizar o máximo da potencialidade do uso de janelas na métrica SSIM e fomos em busca da janela ideal com base no seguinte problema inverso: qual ponderação deve ter cada pixel dentro da janela gaussiana para que o índice SSIM calculado pela métrica seja igual a pontuação subjetiva dada pelo sistema visual humano? Para tentar responder a essa pergunta, lançamos mão dos resultados obtidos na Seção 4.2 (sobre o tamanho ideal de janela), consideramos uma matriz de variáveis de tamanho 3×3

$$w = \begin{pmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{pmatrix} \tag{49}$$

e fomos em busca dos valores de w_i que nos fornecem a janela ideal. Para avançarmos nesta investigação, precisamos falar um pouco sobre fundamentos e algoritmos de otimização.

5 UM POUCO SOBRE OTIMIZAÇÃO

Resolvemos problemas todos os dias. Alguns destes problemas estão ligados ao conceito de *otimização*, tais como decidir o caminho mais curto para chegar a certo local ou determinar o melhor produto a ser comprado com uma quantia fixa de dinheiro.

Muitos dos problemas que resolvemos podem ser modelados matematicamente e a sua solução pode ser um ponto de mínimo ou máximo (global ou local) de determinada função que descreve o problema em questão. Para isso, é necessário encontrar uma expressão matemática que representa o problema a ser otimizado, além de decidir se ele apresenta alguma *restrição*, isto é, se o valor ótimo deve ser procurado em todo o domínio da função ou apenas em certa região dele. Para ficar mais claro, seguem os elementos que compõem um problema de otimização:

- um vetor de *variáveis* $x = (x_1, ..., x_n)$;
- uma *função-objetivo* $f : D \subset \mathbb{R}^n \to \mathbb{R}$, dependendo de x, que queremos otimizar;
- um *conjunto viável* $C = \{x \in \mathbb{R}^n : c_i(x) \leq 0, c_j(x) = 0\}$, em que $c_i(x)$, $i = 1, \ldots, m$ e $c_j(x)$, $i = 1, \ldots, p$ são funções dependendo de x.

Caso $C = D$, dizemos que o problema de otimização é *irrestrito*. Caso contrário, é um problema *restrito*.

Problemas de otimização podem ser problemas de minimização ou de maximização e, em geral, qualquer problema de minimização pode ser escrito da forma

$$\begin{array}{ll} \min_{x \in D} & f(x) \\ \text{s. a.} & \begin{array}{l} c_i(x) \leq 0 \\ c_j(x) = 0 \end{array} \end{array}, \tag{50}$$

em que “s. a.” é uma abreviação para “sujeito a”. A escrita de problemas de maximização é análoga.

Definição 11. *Dada uma função-objetivo $f : D \subset \mathbb{R}^n \to \mathbb{R}$ associada a um problema do tipo (50), dizemos que x^* é uma solução do problema se $x^* \in D$ tal que $c_i(x^*) \leq 0$, $c_j(x^*) = 0$ e $f(x^*) \leq f(x)$ para todo $x \in C$. Neste caso, x^* também pode ser chamado de mínimo global de f. Caso ocorra a desigualdade $f(x^*) \leq f(x)$ somente para todo x em uma vizinhança de x^* em C, dizemos que x^* é um ponto de mínimo local de f.*

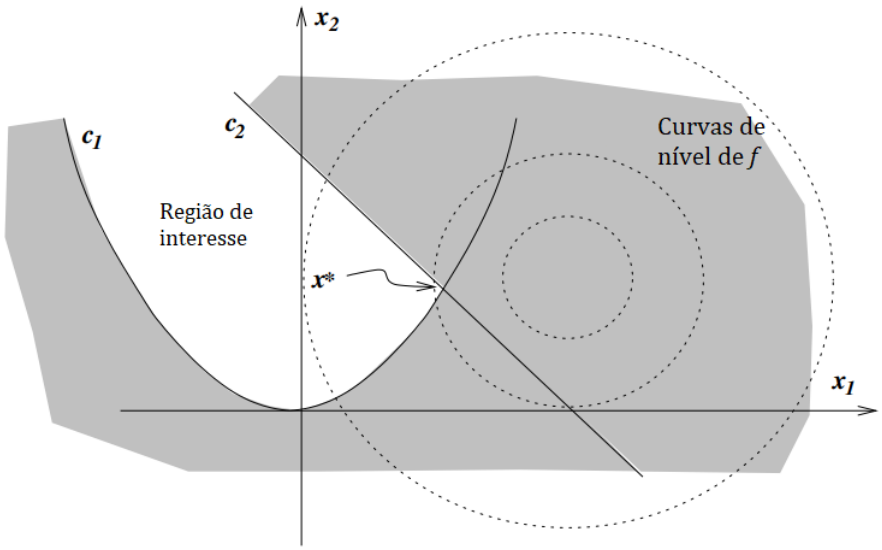
A partir desse momento, sem perda de generalidade, trataremos problemas de otimização por problemas de minimização. A teoria para maximação é a mesma, já que a solução x^* de um problema de maximização com uma função-objetivo f é a mesma solução x^* do problema de minimização, com as mesmas restrições, cuja função-objetivo é $-f$.

Para ilustrar, considere o problema

$$\begin{array}{ll} \min_{x\in\mathbb{R}^2} & (x_1-2)^2+(x_2-1)^2 \\ \text{s. a.} & x_1^2-x_2\leq 0 \\ & x_1+x_2\leq 2 \end{array} \quad . \tag{51}$$

Neste caso, temos que $x = (x_1, x_2)$, a função-objetivo a ser minimizada é $f(x) = (x_1 - 2)^2 + (x_2 - 1)^2$ e o conjunto viável é $C = \{x \in \mathbb{R}^2 : x_1^2 - x_2 \leq 0, x_1 + x_2 - 2 \leq 0\}$. A Figura 25 traz uma representação gráfica do problema (51). Podemos observar as curvas de nível da função objetivo f, que são circunferências centradas em $(2, 1)$ e, como a função-objetivo modela um paraboloide, seu ponto de mínimo x^* seria $(2, 1)$, mas este ponto não satisfaz as restrições. Para x^* satisfazer as restrições, ele deve pertencer a região destacada na Figura 25. Dessa forma, podemos observar onde x^* se encontra, dado o comportamento da função objetivo.

Figura 25 – Representação gráfica do problema de otimização (51).



Fonte – Nocedal e Wright (2006). Modificada pelo autor (2021).

Para entender como um problema prático pode ser modelado em termos de um problema do tipo (50), imagine uma situação em que uma fábrica queira minimizar o custo de metal necessário para produzir uma lata cilíndrica que possui volume de 1 litro (STEWART, 2013). Para isso, deve-se encontrar as medidas de raio e altura do cilindro de 1 litro que possua a menor área de superfície. Temos que a função que modela a área de superfície de um cilindro é $A'(r, h) = 2\pi r^2 + 2\pi rh$, em que r e h são as medidas do raio e da altura do cilindro em centímetros, respectivamente. Além disso, temos a restrição do volume do cilindro: $\pi r^2 h = 1000\ cm^3$. Por fim, como estamos num

problema prático, temos que a altura e o raio precisam ser valores não-negativos. Com isso, temos o problema

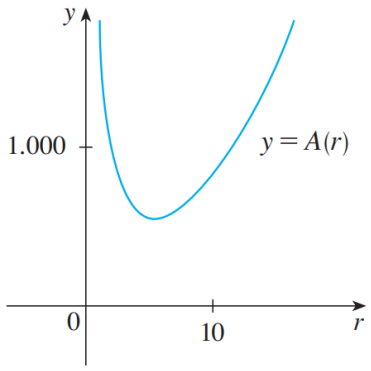
$$\begin{array}{cc} \min_{(r,h)\in\mathbb{R}^2} & 2\pi r^2 + 2\pi rh \\ \text{s. a.} & \pi r^2 h - 1000 = 0 \\ & r \geq 0 \\ & h \geq 0 \end{array} . \tag{52}$$

Podemos simplificar o problema colocando r em função de h na primeira restrição e substituindo na função-objetivo. Então, temos

$$\begin{array}{cc} \min_{r\in\mathbb{R}} & 2\pi r^2 + \frac{2000}{r} \\ \text{s. a.} & r \geq 0 \end{array} , \tag{53}$$

em que o gráfico da nova função-objetivo $A(r)$ é mostrado na Figura 26.

Figura 26 – Gráfico da área de um cilindro com volume igual a 1000 cm^3 em função da medida do raio.



Fonte – Stewart (2013).

É possível observar que a solução do problema (53) está no intervalo $(0, 10)$ e a solução exata $r = \sqrt[3]{500/\pi} \approx 5.42$ pode ser encontrada usando as *condições necessárias de primeira ordem* (NOCEDAL; WRIGHT, 2006) e informações acerca de crescimento e decrescimento da função-objetivo (a resolução completa está em (STEWART, 2013, p. 295-296)).

Problemas de otimização como (51) e (53) podem ser solucionados de maneira relativamente simples. Mas, em geral, problemas de otimização podem ser tão complicados quanto queiramos e, consequentemente, podem não ter soluções fáceis de serem encontradas analiticamente. O processo mais usado para resolver tais problemas é através do uso de algoritmos, com ajuda do computador. Perceba que usei a palavra “algoritmos” no plural, já que não existe um algoritmo universal que resolve qualquer problema de otimização. O que existe são coleções de algoritmos que podem ser úteis na resolução de tipos diferentes de problemas.

Em geral, algoritmos que resolvem problemas de otimização são *iterativos*, ou seja, a partir de determinado ponto de partida x_0, geram uma sequência (x_n) finita de pontos x_n, tais que $f(x_{n+1}) \leq f(x_n)$, até encontrar uma solução para o problema. Cada um dos pontos desta sequência é chamado de *iterado*. A solução encontrada pelo algoritmo pode não ser a solução ótima x^*, mas ele deve ser capaz de parar de iterar a partir do momento que

1. não está conseguindo progredir mais,
2. conseguiu encontrar uma solução com precisão suficiente (tal precisão pode ser especificada pelo indivíduo) ou
3. algum outro critério de parada for satisfeito, como um número máximo de iterações, por exemplo.

Cada algoritmo tem a sua própria maneira de decidir como obter o próximo iterado, dado o iterado atual. Além da iteração atual, é possível que o algoritmo leve em conta informações como as iterações anteriores e características das funções que compõem o problema (função-objetivo e restrições) para decidir a próxima iteração.

Nocedal e Wright (2006) elencam três principais características que se espera de um algoritmo ideal para resolução de um problema de otimização:

- robustez: deve ter bom desempenho em uma grande variedade de problemas dentro da classe de problemas que ele se propõe a resolver, para todos os valores razoáveis do ponto de partida x_0;
- eficiência: não devem exigir muito tempo ou armazenamento do computador;
- precisão: deve identificar uma solução com precisão, sem ser muito sensível a erros nos dados ou aos erros de arredondamento aritmético feitos pelo computador.

Na prática, no entanto, nem sempre é possível obter um algoritmo que tenha as três características ao mesmo tempo. Nesse momento, entra em jogo a subjetividade do indíviduo que precisa resolver o problema, para decidir qual (ou quais) dessas características é mais importante para o problema em questão.

Aspectos teóricos e exemplos de algoritmos que fazem otimização com restrições podem ser encontrados no livro de Nocedal e Wright (2006) e não fazem parte do escopo deste trabalho. A partir de agora, vamos discutir alguns aspectos teóricos e práticos para problemas de otimização sem restrições.

5.1 ESTRATÉGIAS PARA ALGORITMOS DE OTIMIZAÇÃO IRRESTRITA

Como comentado anteriormente, um problema de minimização irrestrito é um problema de minização tal que o conjunto viável coincide com o domínio da função-objetivo. Dessa forma, podemos reformular o problema (50) como

$$\min_{x \in D} \quad f(x) \ . \tag{54}$$

Algoritmos de otimização irrestrita podem usar duas estratégias fundamentais para decidir como encontrar a iteração x_{i+1}, dada a iteração x_i.

Na estratégia de *busca linear*, em cada iteração x_i, o algoritmo escolhe uma direção de descida p_i e procura pela próxima iteração ao longo desta direção. Uma *direção de descida* é um vetor p tal que

$$f(x_i + \alpha p) \leq f(x_i) \tag{55}$$

para todo $\alpha > 0$ suficientemente pequeno. Dessa forma, garante-se que cada iteração terá um valor de função menor que a iteração anterior, estando, dessa forma, mas perto d um ponto de mínimo x^*. Dessa forma, dada uma iteração x_i e uma direção p_i, temos que

$$x_{i+1} = x_i + \alpha p \tag{56}$$

em que α é o escalar que minimiza f ao longo da direção p_i. Assim, o problema de otimização (54) é resolvido através de subproblemas de otimização da forma

$$\min_{\alpha > 0} \quad f(x_i + \alpha p_i) \ , \tag{57}$$

que, em geral, são mais simples de resolver do que o problema original (54). A Figura 27 ilustra a estratégia de busca linear.

A dificuldade deste método se encontra, principalmente, em determinar as direções de descida em cada iteração. Como aprendemos nas aulas de cálculo, a direção

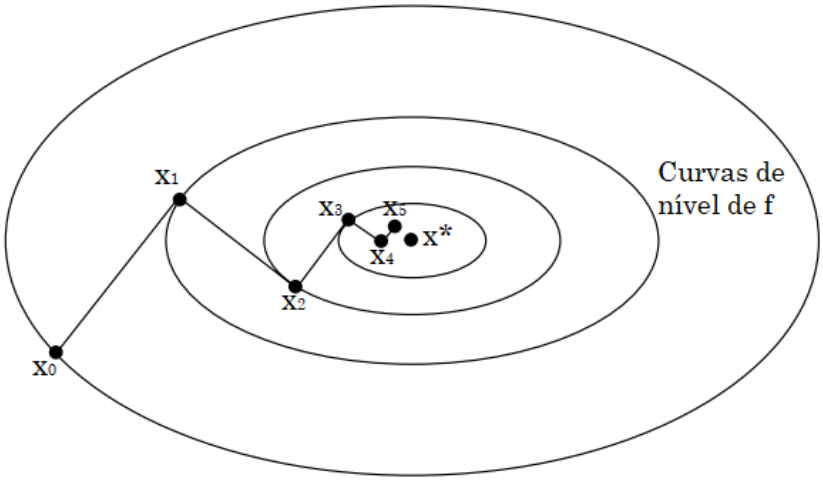
$$p_i = -\nabla f(x_i) \tag{58}$$

é uma direção de descida, além de ser a direção de maior decrescimento de f a partir do ponto x_i. Lembrando que $\nabla f(x_i)$ denota o *gradiente* de f no ponto x_i, isto é,

$$\nabla f(x_i) = \left[\frac{\partial f}{\partial x_1}(x_i), \frac{\partial f}{\partial x_2}(x_i), \ldots, \frac{\partial f}{\partial x_n}(x_i) \right] . \tag{59}$$

Dessa forma, essa direção poderia ser a escolha mais óbvia para fazer a pesquisa linear em cada iteração. Apesar de ter as vantangens de ser necessário calcular apenas as derivadas de primeira ordem da função objetivo em cada iteração e de ser a direção de descida mais rápida (localmente), Nocedal e Wright (2006) mostram que

Figura 27 – Representação gráfica da estratégia de pesquisa linear para otimização de funções irrestritas.



Fonte – Nocedal e Wright (2006). Modificada pelo autor (2021).

a busca linear com a direção $p_i = -\nabla f(x_i)$ pode ser bastante lenta em problemas mais complicados, já que não é incomum que o α_i encontrado em cada busca seja pequeno demais de modo que sejam necessárias mais iterações do que se deseja.

Outras direções que podem ser usadas em algoritmos de busca linear incluem as direções de *Newton* e *Quasi-Newton*, dadas por

$$p_i^N = -(\nabla^2 f(x_i))^{-1} \nabla f(x_i) \tag{60}$$

e

$$p_i^{QN} = -(B_{x_i})^{-1} \nabla f(x_i), \tag{61}$$

respectivamente, em que $\nabla^2 f(x_i)$ denota a matriz Hessiana (matriz das derivadas parciais de segunda ordem) de f no ponto x_i e B_{x_i} denota uma matriz que é uma aproximação de $\nabla^2 f(x_i)$, sem precisar calcular as derivadas de segunda ordem de f. Nocedal e Wright (2006) mostram exemplos de algoritmos que usam as direções citadas acima.

Na segunda estratégia, chamada de *região de confiança*, em cada iteração x_i é usada uma função auxiliar m_i que aproxima a função f em uma região de confiança $T_i \subset Dom(f)$ tal que $x_i \in T_i$. Assim, a ideia é procurar um minimizador para a função m_i em torno de x_i, através da solução de subproblemas da forma

$$\min_p \quad m_i(x_i + p) \ , \tag{62}$$

tal que $x_i + p \in T_i$. Essa restrição é importante pois m_i pode não ser uma boa aproximação da f fora da região de confiança T_i. Caso a solução de (62) não produza um bom decrescimento em f, deve-se diminuir a região de confiança.

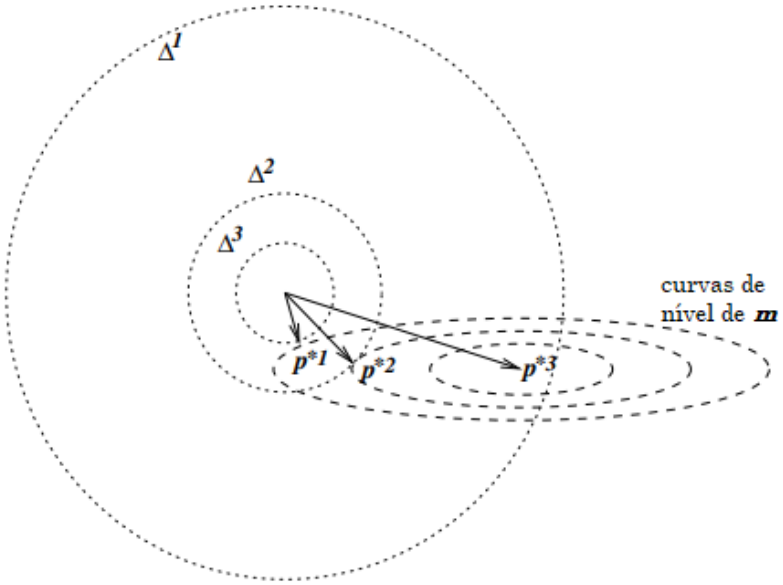
Em geral, usa-se uma bola centrada em x_i e raio Δ_i como T_i e uma função quadrática como m_i. Essa estratégia dá origem ao processo iterativo

$$x_{i+1} = x_i + \Delta_i p_i^* \tag{63}$$

em que p_i^* é a solução de (62). Em problemas práticos, resolver os subproblemas de otimização das aproximações quadráticas m_i é, geralmente, mais simples do que a otimização original (54).

Note que a direção p_i^* pode mudar a medida que o raio da região de confiança diminui, como ilustra o exemplo da Figura 28: podemos ver que, quanto menor for a região de confiança, mais "correta" é a direção escolhida para a próxima iteração. Porém, se Δ_i for muito pequeno, a distância entre as iterações será muito pequena também e, portanto, serão necessárias mais iterações para chegar à solução do problema.

Figura 28 – Representação gráfica da estratégia de região de confiança para otimização de funções irrestritas.



Fonte – Nocedal e Wright (2006). Modificada pelo autor (2021).

Como resumem Nocedal e Wright (2006), as abordagens de busca linear e região de confiança diferem na maneira em que escolhem a direção e a distância do passo de uma iteração para a próxima: a pesquisa linear começa fixando a direção e então identificando uma distância apropriada, ou seja, o "comprimento do passo". Na região de confiança, primeiro escolhemos uma distância máxima (Δ_i) e então

buscamos uma direção que atinja o maior decrescimento da função auxiliar sujeito a esta restrição de distância. Se esta etapa for insatisfatória, reduz-se o raio e faz-se uma nova tentativa.

A seguir, vamos ver, na prática, a teoria descrita neste capítulo ao usarmos algoritmos de otimização com a finalidade de tirar máximo proveito da métrica SSIM, de acordo com o problema apresentado no final do Capítulo 4.

6 A JANELA DESLIZANTE ÓTIMA

O Capítulo 4 deste trabalho foi dedicado às primeiras investigações em parâmetros “ocultos” na métrica SSIM que poderiam fazer com que seu desempenho fosse superior à versão padrão da métrica: o tamanho e o formato da janela deslizante utilizada para realizar os cálculos das variáveis estatísticas usadas no índice SSIM (Equação (21)). Lembrando que “formato” se refere ao peso que cada pixel recebe dentro da janela deslizante. No final do Capítulo 4, chegamos à conclusão que trocar o formato padrão (gaussiano) por qualquer um dos outros dois formatos já existentes no MATLAB não gera grandes ganhos em termos de desempenho da métrica. Com base nisso, criou-se o problema de obter formato ótimo de janela deslizante, ou seja, fixado um tamanho n, gostaríamos de obter uma matriz $n \times n$ que faça com que o uso dessa matriz como a janela deslizante do índice SSIM implique numa reprodução fiel da avaliação subjetiva ao usar a métrica para determinar a qualidade visual de uma imagem.

Neste capítulo, apresentaremos o problema de otimização relacionado a esta investigação juntamente com os procedimentos realizados para resolução deste problema, além dos resultados obtidos e discussões.

6.1 O PROBLEMA DE OTIMIZAÇÃO DA JANELA DESLIZANTE

Novamente, vamos lançar mão do banco de imagens de Ponomarenko *et al.* (2009), que possui diversas imagens submetidas a várias distorções, além de virem acompanhadas pela pontuação MOS (avaliação subjetiva).

De acordo com os resultados que obtivemos no Capítulo 4, para os formatos já existentes no MATLAB, a maior correlação entre o índice MOS e o índice SSIM ocorre ao utilizar janelas deslizantes de tamanho 3×3. Levando em consideração este resultado e também o fato de que aumentar o tamanho da janela deslizante ocasiona um aumento no número e variáves e, por consequência, a dificuldade do problema de otimização, fixaremos este tamanho para a nossa investigação. Assim, seja a matriz

$$w = \begin{pmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{pmatrix}. \tag{64}$$

Dadas duas imagens digitais x e y, de acordo com as Equações (42)-(44), as variáveis estatísticas usadas para calcular $SSIM(x, y)$ podem ser escritas em função

da janela deslizante w. Como os valores de w_i representam pesos que serão usados para o cálculo de variáveis estatísticas, vamos impor a restrição

$$\sum_{i=1}^{9} w_i = 1, \tag{65}$$

que implica nas simplificações

$$\sigma_x^2 = \sum_{i=1}^{9} w_i(x_i-\mu_x)^2 = \sum_{i=1}^{9}(w_i x_i^2-2w_i x_i\mu_x-w_i\mu_x^2) = \sum_{i=1}^{9} w_i x_i^2-2\mu_x \sum_{i=1}^{9} w_i x_i+\mu_x^2 \sum_{i=1}^{9} w_i =$$

$$= \sum_{i=1}^{9} w_i x_i^2 - 2\mu_x^2 + \mu_x^2 = \sum_{i=1}^{9} w_i x_i^2 - \mu_x^2 \tag{66}$$

e

$$\sigma_{xy} = \sum_{i=1}^{9} w_i(x_i - \mu_x)(y_i - \mu_y) = \sum_{i=1}^{9}(w_i x_i y_i - w_i x_i\mu_y - w_i y_i\mu_x + w_i\mu_x\mu_y) =$$

$$= \sum_{i=1}^{9} w_i x_i y_i-\mu_y \sum_{i=1}^{9} w_i x_i-\mu_x \sum_{i=1}^{9} w_i y_i+\mu_x\mu_y \sum_{i=1}^{9} w_i = \sum_{i=1}^{9} w_i x_i y_i-\mu_y\mu_x-\mu_x\mu_y+\mu_x\mu_y =$$

$$= \sum_{i=1}^{9} w_i x_i y_i - \mu_x\mu_y. \tag{67}$$

Assim, temos as variáveis estatísticas usadas na métrica SSIM, da maneira como escrita nas implementações:

$$\mu_x = \sum_{i=1}^{9} w_i x_i, \tag{68}$$

$$\sigma_x = \sqrt{\sum_{i=1}^{9} w_i(x_i - \mu_x)^2} = \sqrt{\sum_{i=1}^{9} w_i x_i^2 - \mu_x^2} \tag{69}$$

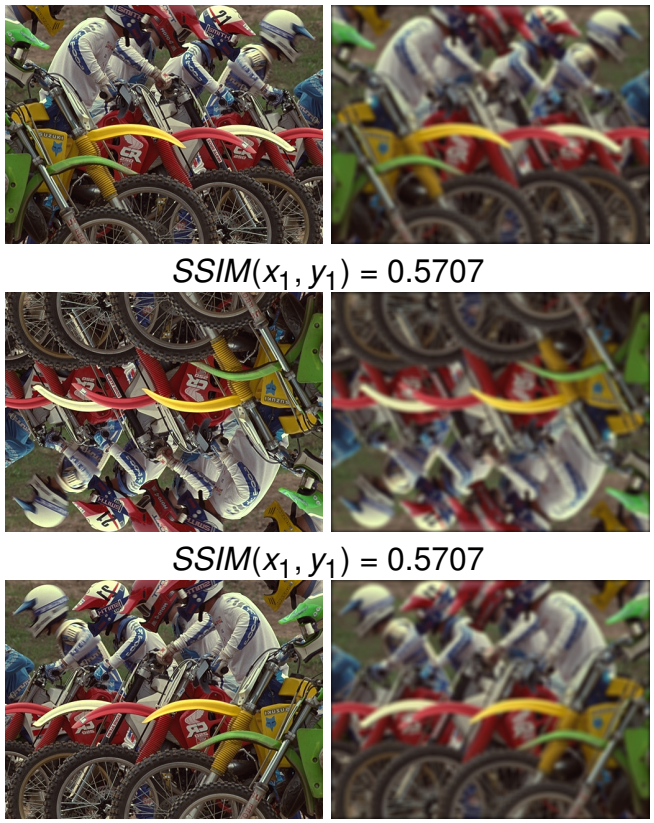
e

$$\sigma_{xy} = \sum_{i=1}^{9} w_i(x_i - \mu_x)(y_i - \mu_y) = \sum_{i=1}^{9} w_i x_i y_i - \mu_x\mu_y. \tag{70}$$

Além disso, para não precisarmos nos preocupar com minimização restrita, vamos usar a Equação (65) para reescrever a matriz w como

$$w = \begin{pmatrix} w_1 & w_2 & w_3 \\ w_4 & 1-\sum_{i=1}^{8} w_i & w_5 \\ w_6 & w_7 & w_8 \end{pmatrix}. \tag{71}$$

Figura 29 – Imagem contaminada por borramento gaussiano e índices SSIM (original) associados à imagem após sofrer movimentos rígidos



Fonte – Ponomarenko *et al.* (2009). Modificada pelo autor (2021).

Além disso, considerando um par de imagens digitais (x, y), veja que, caso aplicássemos algum movimento rígido nas duas imagens ao mesmo tempo (como rotações de 90° ou reflexões), seria esperado que a qualidade visual de uma imagem comparada a outra se mantivesse inalterada. A métrica SSIM original (como proposta por Wang *et al.* (2004)) toma essa preocupação ao utilizar a janela deslizante gaussiana 11×11, a qual possui simetrias (como pode ser visto na Equação (41)) que garantem que o índice se mantenha inalterado mesmo se as imagens rotacionarem ou refletirem. A Figura 29 ilustra esse fenômeno, mostrando movimentos rígidos sendo aplicados às imagens e o índice SSIM (original) se mantendo inalterado após estes movimentos. O mesmo não ocorre se calcularmos o índice SSIM usando uma matriz w que satisfaz a restrição $\sum_i w_i = 1$ mas não preserva simetrias. A Figura 30 mostra esse fenômeno.

Figura 30 – Imagem contaminada por borramento gaussiano e índices SSIM (modificado com uma matriz w tal que $\sum_i w_i = 1$) associados à imagem após sofrer movimentos rígidos

$SSIM(x_1, y_1) = 0.1279$

$SSIM(x_1, y_1) = 0.4517$

$SSIM(x_1, y_1) = 0.6559$

Fonte – Ponomarenko *et al.* (2009). Modificada pelo autor (2021).

Com base nesse comportamento, decidimos impor que a matriz w a ser buscada em nossa otimização deve preservar simetrias, sendo assim, vamos reescrever a matriz de variáveis da Equação (71) como

$$w = \begin{pmatrix} w_1 & w_2 & w_1 \\ w_2 & 1 - 4 \cdot w_1 - 4 \cdot w_2 & w_2 \\ w_1 & w_2 & w_1 \end{pmatrix}, \tag{72}$$

inspirados pelas simetrias da matriz com ponderação gaussiana (Equação (41)). Veja que a nova restrição implica em um número menor de variáveis a serem procuradas. A diminuição no número de variáveis de um problema de otimização, em geral, faz com que ele seja um problema mais simples de ser resolvido.

Por causa das Equações (68)-(70), a função SSIM pode ser entendida como uma função que depende dos valores dos pesos w_i e, portanto, passaremos a usar $SSIM_{x,y}(w)$ ao invés de $SSIM(x, y)$. Além disso, como fixamos a imagem de referência

x e a imagem distorcida y, podemos usar o valor MOS associado a este par de imagens (indicado por $mos_{x,y}$) e definir a função objetivo do nosso problema de otimização como

$$E(w) = [SSIM_{x,y}(w) - mos_{x,y}]^2. \tag{73}$$

Veja que se encontrarmos um ponto de mínimo w^* da função $E(w)$, estaremos encontrando os valores de $w_1^*, \ldots, w_8^*$ que fazem com que o erro entre o índice MOS e o índice SSIM seja o menor possível. Dessa forma, a métrica SSIM estará se comportando da maneira mais semelhante possível à avaliação subjetiva, para o par de imagens (x, y).

Assim, definindo $\mathcal{M}_3'$ como o conjunto das matrizes 3×3 com entradas reais respeitando a restrição mostrada na matriz da Equação (72), temos o problema de otimização:

$$\min_{w \in \mathcal{M}_3'} \quad \mathrm{E(w)} \ . \tag{74}$$

Um último detalhe na definição do nosso problema é que, como a matriz w é utilizada para calcular variáveis estatísticas de média, desvio padrão e covariância, faria sentido procurar por valores de $w_1, \ldots, w_8$ positivos, para que não ocorra, por exemplo, o caso de μ_x ser um valor negativo (o que poderia não fazer sentido na prática). No entanto, tal análise implicaria em utilizar algoritmos de otimização restrita que, em geral, são mais complexos do que algoritmos de otimização irrestrita e não serão abordados neste trabalho. Assim, estaremos interessados em fazer uma primeira abordagem no problema de otimização proposto, isto é, sem considerar as restrições comentadas.

6.2 IMPLEMENTAÇÕES E RESULTADOS

A Figura 31 mostra os pares de imagens (x_1, y_1), (x_2, y_2), (x_3, y_3) e (x_4, y_4) que serão usados nessa investigação, juntamente com suas respectivas pontuações MOS. Toda vez que nos referirmos a um par (x_i, y_i), com $i \in \{1, 2, 3, 4\}$, estaremos fazendo referência à Figura 31. Os índices MOS associados às imagens de Ponomarenko *et al.* (2009) têm valores no intervalo $[0, 9]$ (como vimos antes, normalmente uma pontuação MOS está entre 0 e 5, mas os autores optaram por usar outro intervalo), mas, para fazer uma melhor comparação com os índices SSIM, transformamos estes valores para que estejam no intervalo $[0, 1]$.

Para realizar a otimização com mais de um par de imagens, a função objetivo, mostrada na Equação (73), deve ser alterada para

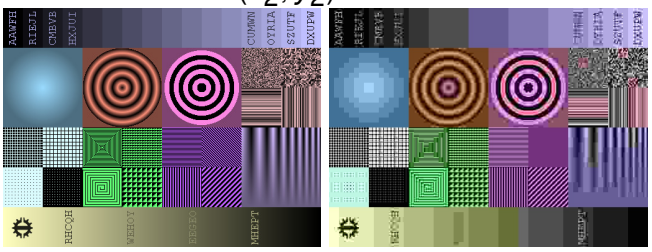
$$E_i(w) = [SSIM_{x_1,y_1}(w) - mos_{x_1,y_1}]^2 + \cdots + [SSIM_{x_i,y_i}(w) - mos_{x_i,y_i}]^2, \tag{75}$$

em que i representa a quantidade de pares de imagens de entrada utilizadas.

Figura 31 – Imagens distorcidas acompanhadas de pontuação MOS

$MOS(x_1, y_1) = 0.12794$

$MOS(x_2, y_2) = 0.46464$



$MOS(x_3, y_3) = 0.09333$

$MOS(x_4, y_4) = 0.40625$

Fonte – Ponomarenko *et al.* (2009).

Uma informação importante, para comparação, é que, usando todos os 1700 pares de imagens de Ponomarenko *et al.* (2009), temos que

$$E_{1700}(W) \approx 163 \tag{76}$$

em que W é a matriz gaussiana 11×11 (Equação (41)), usada por padrão na SSIM. Assim, consideramos que, caso encontremos uma matriz tal que a função E, com $i = 1700$, aplicada nessa matriz resulte em um valor menor do que 163, estaremos melhorando o desempenho da métrica SSIM.

Considerando que o banco de imagens de Ponomarenko *et al.* (2009) é criado com o intuito de representar as mais variadas cenas encontradas em imagens, com as mais variadas distorções que uma imagem pode ser submetida, consideramos

que, caso conseguíssemos realizar o processo de minimização com as imagens deste banco, estaríamos gerando uma matriz ótima w^* que melhoraria o desempenho da métrica SSIM ao máximo para qualquer par de imagens de entrada.

Usaremos o software *MATLAB* R2017b com as funções *fminunc*, que gera iterações a partir do método de *Quasi-Newton* (Equação (61)), e *fminsearch*, que gera iterações a partir de uma estratégia que não faz uso de derivadas (o cálculo das derivadas de uma função-objetivo pode implicar em uma diminuição significativa da eficiência de um algoritmo de otimização, dependendo da complexidade do problema). Para isso, usamos as Equações (68)-(70), juntamente com a definição da função SSIM (21), para implementar um código que consiga buscar soluções w^* do problema. Um exemplo de código que implementamos para gerar os resultados que serão apresentados nessa seção pode ser encontrado no Apêndice C.

Como as imagens são coloridas, vamos usar em nossa implementação apenas a componente "R" do RGB (ver Apêndice A) para realizar as otimizações, para que o problema seja mais simples de ser resolvido. Assim, estaremos assumindo que o índice MOS associado a um par de imagens coloridas (x_i, y_i) é o mesmo índice MOS que estaria associado às imagens x_i y_i em tons de cinza.

Como também serão exibidos resultados a respeito da eficiência dos métodos de otimização utilizados, é importante destacar que os resultados a serem apresentados foram feitos em um computador com processador AMD Ryzen™ 5 3500U 2.10 GHz e memória RAM de 12.0 GB.

Os primeiros resultados que obtivemos são apresentados nas Tabelas 1 e 2.

Tabela 1 – Eficiência e erro da otimização em duas variáveis com diferentes valores de pares de imagens de entrada, usando a função *fminunc* do *MATLAB*

Pares de imagens	**Valores de** $\begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$ $(\approx)$	$\mathbf{E}_i(w^*)$	**Tempo de busca (s)**
$i = 1$	$\begin{pmatrix} 0.1574 \\ 0.1033 \end{pmatrix}$	$2.04 \cdot 10^{-4}$	6
$i = 2$	$\begin{pmatrix} 0.1807 \\ 0.0992 \end{pmatrix}$	0.0870	6
$i = 3$	$\begin{pmatrix} 0.1777 \\ 0.1111 \end{pmatrix}$	0.4460	7
$i = 4$	$\begin{pmatrix} 0.1429 \\ 0.1154 \end{pmatrix}$	0.4872	8

Fonte – Elaborada pelo autor (2021).

Com tais resultados, de fato conseguimos melhorar o desempenho da métrica SSIM na avaliação da qualidade de imagens apenas para os 4 pares de imagens que escolhemos, como pode ser visto na valores da Tabela 3, que compara o desempenho da métrica SSIM usada com a janela W (41) e com as janelas ótimas w^* de acordo

Tabela 2 – Eficiência e erro da otimização em duas variáveis com diferentes valores de pares de imagens de entrada, usando a função *fminsearch* do *MATLAB*

Pares de imagens	**Valores de** $\begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$ $(\approx)$	$\mathbf{E}_i(w^*)$	**Tempo de busca (s)**
$i = 1$	$\begin{pmatrix} 0.4171 \\ 0.0747 \end{pmatrix}$	$7.03 \cdot 10^{-6}$	3
$i = 2$	$\begin{pmatrix} 0.2243 \\ 0.0415 \end{pmatrix}$	0.0870	6
$i = 3$	$\begin{pmatrix} 0.2534 \\ 0.0450 \end{pmatrix}$	0.1394	9
$i = 4$	$\begin{pmatrix} 0.2184 \\ 0.0430 \end{pmatrix}$	0.4692	11

Fonte – Elaborada pelo autor (2021).

com a Tabela 2 (a função *fminsearch* apresentou menor erro entre as duas funções no MATLAB). Para uma comparação mais clara, informamos o valor $\frac{E_i(W)}{E_i(w^*)}$, que pode ser entendido como um "percentual de melhora".

Tabela 3 – Desempenho da métrica SSIM original comparado com o desempenho da métrica SSIM modificada

Pares de imagens	$\mathbf{E}_i(W)$	$\mathbf{E}_i(w^*)$	$\mathbf{E}_i(W)$ / $\mathbf{E}_i(w^*)$
$i = 1$	0.1986	$7.03 \cdot 10^{-6}$	$2.8 \cdot 10^4$
$i = 2$	0.2230	0.0870	2.6
$i = 3$	0.7276	0.1394	5.2
$i = 4$	0.8070	0.4692	1.7

Fonte – Elaborada pelo autor (2021).

Porém, como também pode ser observado, o desempenho da métrica diminui à medida em que aumentamos os pares de imagens de entrada. Isso é um problema já que, como comentamos, estamos interessados em encontrar a janela deslizante que seja ótima para um par arbitrário de imagens de entrada que, teoricamente, seria encontrada ao usar a maior quantidade possível de pares de imagens durante a otimização. Na prática, no entanto, além do processo de otimização se tornar cada vez mais lento ao aumentar a quantidade de pares de imagens, o erro também aumenta de maneira indesejada.

Lembre-se que estamos fazendo as otimizações em duas variáveis, w_1 e w_2, a fim de preservar a avaliação da qualidade das imagens mesmo que elas sofram rotações e reflexões, como discutido anteriormente. Dessa forma, uma possível maneira para contornar o problema do aumento indesejado do erro com mais pares de imagens poderia ser aumentar o tamanho da janela deslizante (nesta seção, adotamos o tamanho 3×3), já que isso implicaria em mais variáveis disponíveis no processo de otimização, o que aumentaria os graus de liberdade para se encontrar a janela ótima.

Para manter as simetrias descritas na seção anterior, usaremos matrizes de variáveis de ordem ímpar j, da forma

$$w = \begin{pmatrix} w_1 & w_2 & \cdots & w_{\frac{j+1}{2}} & \cdots & w_2 & w_1 \\ w_2 & w_3 & \cdots & w_{\frac{j+3}{2}} & \cdots & w_2 & w_2 \\ \vdots & \vdots & & \vdots & & \vdots & \vdots \\ w_{\frac{j+1}{2}} & w_{\frac{j+3}{2}} & \cdots & 1-\sum_i w_i & \cdots & w_{\frac{j+3}{2}} & w_{\frac{j+1}{2}} \\ \vdots & \vdots & & \vdots & & \vdots & \vdots \\ w_2 & w_3 & \cdots & w_{\frac{j+3}{2}} & \cdots & w_2 & w_2 \\ w_1 & w_2 & \cdots & w_{\frac{j+1}{2}} & \cdots & w_2 & w_1 \end{pmatrix}. \tag{77}$$

Os resultados são apresentados na Tabela 4, onde foram fixados os quatro pares de imagens da Figura 31, enquanto variamos a ordem da matriz de variáveis w. Para obter o "percetual de melhora", vamos usar que $E_4(W) = 0.8070$, em que W (Equação (41)) é a matriz gaussiana padrão 11×11 usada por Wang *et al.* (2004).

Tabela 4 – Desempenho da métrica SSIM usada com as janelas w^8 de diferentes tamanhos, encontradas usando a função *fminsearch* do *MATLAB*

Ordem da matriz w	$\mathbf{E_4}(w^*)$	$\mathbf{E_4}(W)$ / $\mathbf{E_4}(w^*)$	**Tempo de busca (s)**
3	0.4692	0,74	9
5	0.1151	6,11	27
7	0.5070	0,61	39
9	0.5456	0,5	59
11	0.1308	5,25	167
13	0.3155	1,59	189

Fonte – Elaborada pelo autor (2021).

De acordo com os resultados obtidos com a otimização em *MATLAB*, as matrizes ótimas que produziram melhores resultados em relação à função SSIM original, foram as matrizes de tamanhos 5×5 e 11×11 dadas por, respectiva e aproximadamente,

$$\begin{pmatrix} -0.0051 & 0.0228 & 0.0255 & 0.0228 & -0.0051 \\ 0.0228 & 0.0255 & 0.0004 & 0.0255 & 0.0228 \\ 0.0255 & 0.0004 & 0.6324 & 0.0004 & 0.0255 \\ 0.0228 & 0.0255 & 0.0004 & 0.0255 & 0.0228 \\ -0.0051 & 0.0228 & 0.0255 & 0.0228 & -0.0051 \end{pmatrix}$$

e

$$10^{-4} \cdot \begin{pmatrix} -2 & -27 & 14 & 7 & 40 & 37 & 40 & 7 & 14 & -27 & -2 \\ -27 & 14 & 7 & 40 & 37 & 23 & 37 & 40 & 7 & 14 & -27 \\ 14 & 7 & 40 & 37 & 23 & 30 & 23 & 37 & 40 & 7 & 14 \\ 7 & 40 & 37 & 23 & 30 & -1 & 30 & 23 & 37 & 40 & 7 \\ 40 & 37 & 23 & 30 & -1 & -21 & -1 & 30 & 23 & 37 & 40 \\ 37 & 23 & 30 & -1 & -21 & 7756 & -21 & -1 & 30 & 23 & 37 \\ 40 & 37 & 23 & 30 & -1 & -21 & -1 & 30 & 23 & 37 & 40 \\ 7 & 40 & 37 & 23 & 30 & -1 & 30 & 23 & 37 & 40 & 7 \\ 14 & 7 & 40 & 37 & 23 & 30 & 23 & 37 & 40 & 7 & 14 \\ -27 & 14 & 7 & 40 & 37 & 23 & 37 & 40 & 7 & 14 & -27 \\ -2 & -27 & 14 & 7 & 40 & 37 & 40 & 7 & 14 & -27 & -2 \end{pmatrix}.$$

É importante ressaltar que, mesmo obtendo desepenho inferior às duas matrizes acima, todas as matrizes ótimas de outros tamanhos encontradas resultaram em uma métrica SSIM com desempenho superior à metrica SSIM original, como mostra a Tabela 4. Ressaltamos, mais uma vez, que tais resultados se aplicam aos quatro pares de imagens da Figura 31. Caso usássemos as duas matrizes acima para calcular $E_{1700}(w^*)$, obtemos os valores 192 e 420, que são superiores a 163 (valor que queremos diminuir). Assim, como comentamos, para obter uma métrica SSIM que seja superior à métrica original para qualquer par de imagens, a matriz ótima deve ser buscada com mais pares de imagens de entrada durante o processo de otimização. Como apontam os valores da Tabela 4, procurar por matrizes de ordem 5 e 11 parece ser um caminho promissor para obter a janela deslizante ótima.

Portanto, escolhemos o tamanho 5×5 (por ter menos variáveis) que nos fornece a matriz de variáveis

$$w = \begin{pmatrix} w_1 & w_2 & w_3 & w_2 & w_1 \\ w_2 & w_3 & w_4 & w_3 & w_2 \\ w_3 & w4 & 1 - \sum_i w_i & w_4 & w_3 \\ w_2 & w_3 & w_4 & w_3 & w_2 \\ w_1 & w_2 & w_3 & w_2 & w_1 \end{pmatrix} \tag{78}$$

para realizar a otimização de nossa função-objetivo (75) no MATLAB com a função *fminsearch*, usando o maior número de pares de imagens de entrada que conseguirmos. Os resultados são apresentados na Tabela 5. Lembrando que o erro entre SSIM (utilizada da forma padrão) e MOS, para todos os 1700 pares de imagens, é de 163, de acordo com (76).

Tabela 5 – Desempenho da métrica SSIM usada com as janelas w^* de tamanho 5×5 encontradas usando a função *fminsearch* do *MATLAB* para diferentes quantidades de imagens de entrada.

Pares de imagens	$\mathbf{E}_{1700}(w^*)$	**Tempo de busca (s)**
50	200	490
150	188	1087
500	186	4625
1000	163	8353
1700	161	12541 ($\approx 3,5$ h)

Fonte – Elaborada pelo autor (2021).

Ao realizarmos a otimização com todos os 1700 pares de imagens conseguimos encontrar uma matriz ótima

$$w^* \approx \begin{pmatrix} -0.004488155 & 0.044300571 & 0.055553294 & 0.044300571 & -0.004488155 \\ 0.044300571 & 0.055553294 & 0.005205373 & 0.055553294 & 0.044300571 \\ 0.055553294 & 0.005205373 & 0.198300201 & 0.005205373 & 0.055553294 \\ 0.044300571 & 0.055553294 & 0.005205373 & 0.055553294 & 0.044300571 \\ -0.004488155 & 0.044300571 & 0.055553294 & 0.044300571 & -0.004488155 \end{pmatrix} \tag{79}$$

que, pela primeira vez, tem desempenho melhor para todo o banco do imagens do que a matriz gaussiana padrão (41): conseguimos um erro total de 161, enquanto o erro total com a matriz padrão é de 163.

Se analisarmos nossa matriz w^*, podemos ver que ela possui algumas características que se assemelham à matriz W (41). Caso usada como uma máscara para filtrar uma imagem digital x, atua como um filtro passa-baixa, no qual fazer a convolução entre w^* e x resulta numa imagem filtrada com aspecto levemente borrado (ver Apêndice B). A Figura 32 ilustra esse fato.

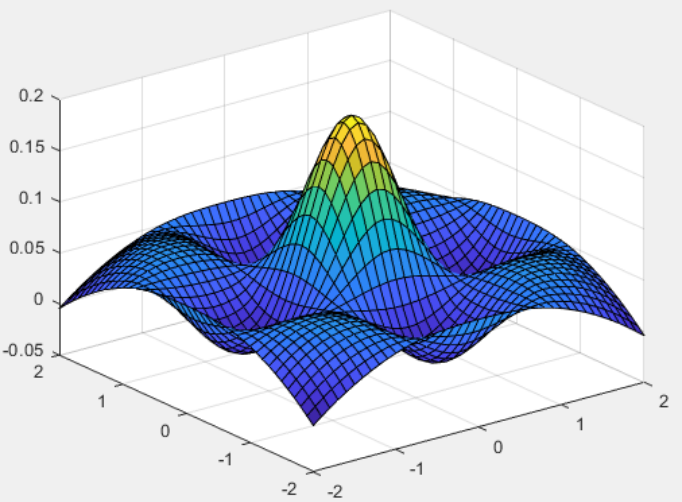
Figura 32 – Imagem filtrada com as matrizes W e w^*: (a) imagem de referência, (b) imagem filtrada com W e (c) imagem filtrada com w^*

(a) (b) (c)

Fonte – Ponomarenko *et al.* (2009). Modificada pelo autor (2021).

Uma representação gráfica de w^* como filtro passa-baixa pode ser visto na Figura 33. Se comparado com o gráfico de um filtro gaussiano, como o da Figura 42, observa-se certa semelhança na distribuição dos pesos, em que pixels centrais têm ponderação mais alta e pixels periféricos têm menos peso.

Figura 33 – Representação gráfica de w^*



Fonte – Elaborada pelo autor (2021).

Por fim, como pode ser observado na Tabela 5, aumentar a quantidade de imagens na busca pela matriz ótima, implica, de fato, em diminuição do erro entre SSIM e MOS. Assim, por mais que tenhamos conseguido melhorar um pouco o desempenho da métrica SSIM, conseguimos ver que há espaço para aumentar o despenho desta métrica ainda mais, e obter uma métrica SSIM que se aproxime da avaliação humana o máximo possível, caso a busca pela janela ótima seja feita com matrizes de outros tamanhos.

7 CONSIDERAÇÕES FINAIS

Conseguimos perceber, ao explorar aspectos teóricos e práticos de algumas métricas, que é possível aplicar o conceito de métrica para comparar imagens digitais e interpretar a distância entre elas como uma medida objetiva para avaliação da qualidade visual de imagens que sofreram algum tipo de distorção.

Em linha com as conclusões obtidas por Pedersen e Hardeberg (2012), foi possível observar que nenhuma das métricas estudadas apresenta alto desempenho para todas as distorções às quais imagens podem ser submetidas. Apesar disso, a ideia de comparar imagens digitais através de sua informação estrutural, proposta na métrica SSIM, foi muito importante para o desenvolvimento de mais técnicas para a avaliação da qualidade visual de imagens e ainda se encontra como uma das métricas mais promissoras para essa finalidade. Explorar aspectos teóricos e limitações apresentadas pelas métricas, em especial a SSIM, fez que com que fosse possível pensar num ajuste de parâmetros “ocultos” que pode resultar em melhoria no seu desempenho.

O cálculo da SSIM utilizando o formato e tamanho padrões para as janelas, sugerido por Wang *et al.* (2004), foi amplamente usado desde seu surgimento até os dias atuais. Juntamente com a alta adesão, um grande número de estudos têm sido feitos com o propósito de melhorar seu desempenho, como o de Golestani e Ghambari (2014), que aponta o tamanho da janela gaussiana como um novo parâmetro. Inspirados por essa nova perspectiva, passamos a entender o formato da janela (ou seja, a ponderação de cada pixel dentro da janela) também como um possível novo ajuste a ser feito na métrica.

Como pode ser observado nos resultados que obtivemos, minimizar a função-objetivo (75) indica um caminho que pode levar à janela ótima, já que, para os quatro pares de imagens estudados, nossas tentativas de otimização resultaram em melhoras bastante significativas no desempenho da métrica. A busca pela janela ótima pode ser mais explorada em trabalhos futuros, ao usar matrizes w de variáveis de outros tamanhos, já que o caminho que tentamos (por limitações no tempo de entrega do trabalho), o tamanho 5×5, implicou em um leve aumento de desempenho e pensamos que há potencialidade para obter resultados melhores ainda.

Estudos sobre o tamanho ideal de janela deslizante, como os de Golestani e Ghambari (2014), podem ajudar neste busca. Como comentado, os resultados experimentais obtidos por eles ainda não estão totalmente claros, apesar de promissores, já que parece existir uma relação entre o tamanho da janela a ser utilizada e o tipo de distorção apresentada pela imagem.

Um próximo passo natural poderia ser buscar uma solução do nosso problema de otimização usando imagens coloridas, já que optamos por usar imagens em tons de cinza a fim de simplificar o problema.

Por fim, deixamos como sugestão usar a métrica SSIM com a matriz $w^\star$ (79) que possui desempenho superior ao desempenho da métrica SSIM como usada por padrão. Lembrando que assumimos que o banco de imagens utilizado foi feito para representar imagens arbitrárias. Ressaltamos mais uma vez que nossos resultados parecem indicar que é possível encontrar uma janela “mais ótima” ainda.

REFERÊNCIAS

BOYAT, A. K.; JOSHI, B. K. A review paper: noise models in digital image processing. **Signal & Image Processing: An International Journal (SIPIJ)**, v. 6, n. 2, 2015.

BREVE, F. A. **Cores e sistemas de cores**. [*S.l.*], 2020. Disponível em: https://www.fabriciobreve.com/trabalhos/cores.pdf. Acesso em: 8 jun. 2020.

BRUNET, D.; VRSCAY, E. R.; WANG, Z. On the Mathematical Properties of the Structural Similarity Index. **IEEE Transactions On Image Processing**, v. 21, n. 4, p. 1488–1499, abr 2012.

CHEN, G.; YANG, C.; PO, L.; XIE, S. Edge-based structural similarity for image quality assessment. **IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**, v. 2, 2006.

FILHO, O. Marques; NETO, H. Vieira. **Processamento Digital de Imagens**. Rio de Janeiro: Brasport, 1999. P. 307.

GOLESTANI, H. B.; GHAMBARI, M. Window size influence on SSIM fidelity. **7th International Symposium on Telecommunication 2014 (IST2014)**, 2014.

GOMES, J.; VELHO, L. **Computação Gráfica**. Rio de Janeiro: IMPA/SBM, 1994.

JUNIOR, J. **Brasil Escola**: O que é infravermelho? [*S.l.*], 2020. Disponível em: https://brasilescola.uol.com.br/o-que-e/fisica/o-que-e-infravermelho.htm. Acesso em: 18 mai. 2020.

KREYSZIG, E. **Introductory Funcional Analysis with Applications**. Nova Iorque: John Wiley & Sons, 1978.

NINASSI, A.; CALLET, P. Le; AUTRUSSEAU, F. Pseudo no reference image quality metric using perceptual data hiding. **SPIE Human Vision and Electronic Imaging**, San Jose, CA, USA, v. 6057-08, 2006.

NOCEDAL, J.; WRIGHT, S. J. **Numerical Optimization**. 2. ed. Nova Iorque: Springer, 2006. P. 664.

OSGOOD, B. **Lecture notes for EE261**: The Fourier transform and its applications. Scotts Valley: CreateSpace, 2014. P. 428.

PAMBRUN, J.; NOUMEIR, R. Limitations of the SSIM quality metric in the context of diagnostic imaging. **International Conference on Image Processing (ICIP)**, 2015.

PEDERSEN, M.; HARDEBERG, J. Y. Full-Reference Image Quality Metrics: classification and evaluation. **Foundations And Trends® In Computer Graphics And Vision**, Now Publishers, v. 7, n. 1, p. 1–80, 2012.

PONOMARENKO, N.; LUKIN, V.; ZELENNSKY, A.; EGIAZARIAN, K.; ASTOLA, J.; CARLI, M.; BATTISTI, F. TID2008: A database for evaluation of full-reference visual quality. **Advances of Modern Radioelectronics**, v. 10, n. 4, 2009.

STEWART, J. **Cálculo**: Volume I. 7. ed. São Paulo: Cengage Learning, 2013. P. 651.

STREIJL, R. C.; WINKLER, S.; HANDS, D. S. Mean opinion score (MOS) revisited: methods and applications, limitations and alternatives. **Multimedia Systems**, Springer Science e Business Media LLC, v. 22, n. 2, p. 213–227, 2014.

TANNÚS, M. T. F. **Comparação de técnicas para a determinação de semelhança entre imagens digitais**. Uberlândia, 2008. Disponível em: `https://repositorio.ufu.br/handle/123456789/14388`. Acesso em: 3 dez. 2020.

VIANA, R. L. **Função delta de Dirac**. [*S.l.*], 2020. Disponível em: `https://blog.ufes.br/jairfreitas/files/2018/03/Texto%5C_RicardoLuizViana%5C_Funcao%5C_Delta%5C_UFPR%5C_2013.pdf`. Acesso em: 15 jul. 2020.

VIOLA, F. **Filtragem no domínio da frequência**. [*S.l.*], 2020. Disponível em: `http://www2.ic.uff.br/~aconci/filtragemdominiofrequencia.pdf`. Acesso em: 6 jul. 2020.

WANG, Z.; BOVIK, A. C. Mean squared error: love it or leave it? a new look at signal fidelity measures. **IEEE Signal Processing Magazine**, v. 26, n. 1, p. 98–117, 2009.

WANG, Z.; BOVIK, A. C.; HAMID, R. S.; SIMONCELLI, E. P. Image quality assessment: from error visibility to structural similarity. **IEEE Transactions on Image Processing**, v. 13, n. 4, 2004.

WANG, Z.; BOVIK, A. C.; LU, L. Why is image quality assessment so difficult? **IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**, 2012.

WANGENHEIM, A. **Encontrando a linha divisória**: Detecção de bordas. Florianópolis, 2020. Disponível em: `http://www.inf.ufsc.br/~aldo.vw/visao/bordas.pdf`. Acesso em: 7 jul. 2020.

XIAO, H.; LIU, Y. Fast l 1-minimization algorithm for robust background subtraction. **EURASIP Journal on Image and Video Processing**, v. 2016, n. 1, 2016.

ZHANG, X.; FENG, X.; WANG, W.; XUE, W. Edge Strength Similarity for Image Quality Assessment. **IEEE Signal Processing Letters**, v. 20, n. 4, p. 319–322, 2013.

Apêndices

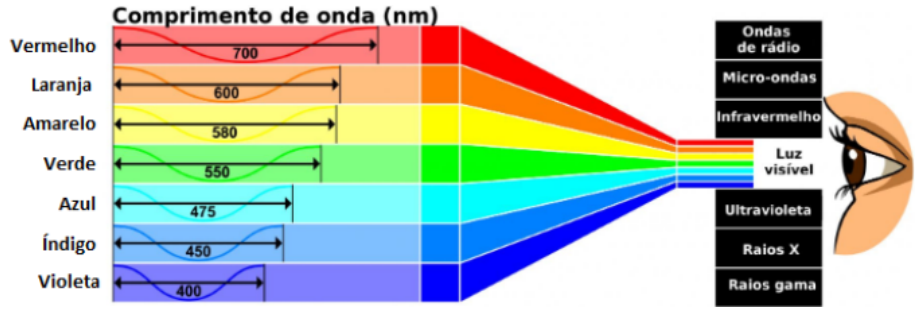
APÊNDICE A – FUNDAMENTOS DE COR

A cor nada mais é do que a manifestação perceptual da luz visível, e como a luz é um sinal eletromagnético, faz sentido estudar a cor sob a perspectiva da Teoria de Sinais.

Sob o ponto de vista da física, a luz é entendida através de um modelo dual que a define como onda e como partícula ao mesmo tempo: um raio luminoso é constituído de partículas, chamadas fótons, e os fótons em movimento determinam uma onda cuja intensidade em cada ponto é igual à probabilidade de se encontrar um fóton nesse ponto.

As cores são distinguidas pelo nosso sistema visual através de diferentes comprimentos de onda. Mas nem todo comprimento de onda é percebido pelo olho humano, apenas ondas de comprimentos entre 380 nm e 780 nm (1 nm = 10^{-9} m) são visíveis para nós. A Figura 34 nos mostra como cada cor se manifesta perceptualmente através de seus comprimentos de onda. Ondas com comprimentos menores do que 380 nm e maiores do que 780 nm são chamadas de *ultravioleta* e *infravermelho*, respectivamente.

Figura 34 – As cores do espectro visível associadas a seus comprimentos de onda



Fonte – Junior (2020).

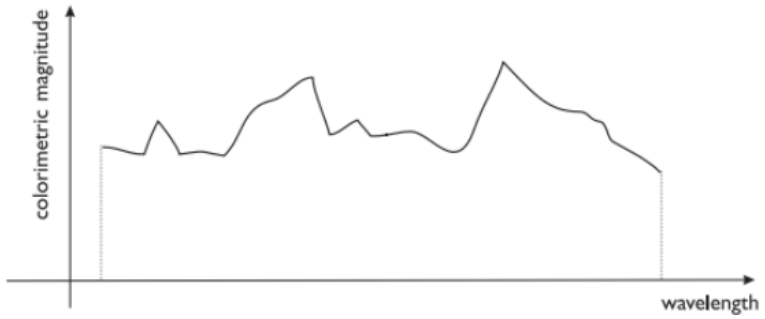
Em nosso cotidiano, percebemos as cores através de vários processos físico-químicos realizados por diversos objetos, alguns destes são fontes de energia luminosa enquanto outros apenas refletem tal energia para gerar a cor resultante na nossa percepção. Tais processos podem ser categorizados em três principais tipos:

- *Processo de formação subtrativa*: a luz que percebemos passa através de algum filtro ou corante antes de chegar ao nosso olho.
- *Processo de formação por pigmentação*: partículas chamadas pigmentos são combinadas, podendo absorver, transmitir e/ou refletir a luz incidida sobre elas.

- *Processo de formação aditiva*: é o processo mais conhecido e também utilizado em dispositivos gráficos, em que a energia da luz que percebemos é resultante de uma combinação de dois ou mais raios luminosos com seus respectivos fótons que são somados. É esse processo que servirá de base ao nosso estudo a partir de agora.

Como visto anteriormente, faz sentido estudar luz e cor sob a perspectiva da Teoria de Sinais, e isso se dá através do modelo espacial de sinal de cor. De acordo com Gomes e Velho (1994), tal modelo associa a cada comprimento de onda a medida da cor dada através de alguma das grandezas de energia radiante. Esse modelo espacial do sinal de cor é chamado usualmente de distribuição espectral e é representado através de uma função $P(\lambda)$, como ilustrado na Figura 35. O processo aditivo de formação de cor é naturalmente associado ao modelo espectral através de sua estrutura vetorial: a soma de duas distribuições espectrais corresponde à adição de cores na natureza, onde a energia dos fótons correspondentes a cada uma das componentes de cor são adicionadas para se obter a cor final.

Figura 35 – Distribuição espectral do sinal de cor em função do comprimento de onda (λ).



Fonte – Gomes e Velho (1994).

Cada pixel de um monitor pode ser entendido como um *sistema físico emissivo de luz*. Esses sistemas emitem um sinal de cor quando eles recebem algum estímulo para tal. Um sistema emissivo possui um número finito de emissores $e_1, ..., e_n$ que se propõem a reproduzir uma determinada cor através do processo de formação aditiva. A emissão básica de cada emissor e_i ($i \in \{1, ..., n\}$) é uma cor com distribuição espectral $P_i(\lambda)$. Essas cores são chamadas de cores primárias do sistema emissivo. Cada cor $C(\lambda)$ reproduzida pelas cores primárias pode ser obtida através de uma combinação linear

$$C(\lambda) = \sum_{n=1}^{n} \beta_k P_k(\lambda), \tag{80}$$

em que o vetor $(\beta_1, \ldots, \beta_n) \in \mathbb{R}_n$ define as *componentes da cor C no sistema emissivo*.

A.1 SISTEMAS DE CORES

A teoria de Young-Helmholtz (século XIX) mostrou que o sistema visual humano é um sistema físico de dimensão três e as moléculas fotossensíveis do olho fazem uma amostragem nas faixas vermelha, verde e azul do espectro, como podemos ver em (GOMES; VELHO, 1994). Assim, tendo como base tais características físicas e biológicas do olho humano, a Comissão Internacional de Iluminação (CIE) adotou, em 1931, uma representação de grande parte do espectro visível que consiste de um espaço tridimensional cuja base de cores primárias são as cores vermelha, verde e azul:

$$P_1(\lambda) = \delta(\lambda - \lambda_1) \text{ em que } \lambda_1 = 700nm \text{ (red)} \tag{81}$$

$$P_2(\lambda) = \delta(\lambda - \lambda_2) \text{ em que } \lambda_2 = 546nm \text{ (green)} \tag{82}$$

$$P_3(\lambda) = \delta(\lambda - \lambda_3) \text{ em que } \lambda_3 = 435,8nm \text{ (blue)} \tag{83}$$

sendo δ a "função" impulso delta de Dirac, que é detalhada no Apêndice B.

Esse sistema aditivo de cores ficou conhecido como CIE-RGB e é amplamente utilizado até hoje em dispositivos gráficos como telas de computador e televisores.

Apesar do sistema de cores CIE-RGB ser o sistema padrão para a exibição de cores em um monitor, existem outros sistemas de cores que também são utilizados, mas em diferentes contextos.

Como mencionamos, o sistema CIE-RGB foi proposto através de estudos do sistema visual humano, mas a exibição de cores através das três primárias (vermelho, verde e azul) não garante que todo o espectro de luz visível seja reproduzido em um monitor, por exemplo.

Como podemos ver em (BREVE, 2020), existe um modelo formado por cores imaginárias (chamadas X, Y e Z) que são definidas matematicamente e servem como um conjunto de primárias que gera todas as cores visíveis. Esse modelo é chamado de CIE-XYZ. Embora essas cores não possam ser reproduzidas fisicamente, o estudo de cálculos para fazer a mudança do sistema CIE-RGB para o sistema CIE-XYZ tem muita importância e serve de base para estudos de mudança entre outros sistemas de cor, como pode ser visto em (GOMES; VELHO, 1994).

Além de sistemas aditivos de cores, existem sistemas subtrativos que também são importantes no estudo de processamento gráfico, como o sistema CMYK. Segundo Breve (2020), uma impressora, por exemplo, não consegue reproduzir cores em um papel através das rimárias do sistema CIE-RGB já que uma folha de papel não emite

luz, apenas a absorve e a reflete. Então, ao transmitir cores de um monitor para um papel, se faz necessário uma mudança de sistema de cores, que passa do CIE-RGB para o CMYK. Este sistema é a base do processo de impressão em quatro cores, a quadricomia, na qual as cores que servem como base de primárias são: ciano (C), magenta (M), amarelo (Y) e preto (K).

Combinando tintas de cores ciano, magenta e amarelo, seria possível reproduzir grande parte do espectro visível. No entanto, na prática, devido a impureza das tintas, a misturas dessas três cores produz um marrom turvo em vez de preto. Portanto, as impressoras adicionam o preto às outras três para produzir as partes mais escuras e cinzas das imagens.

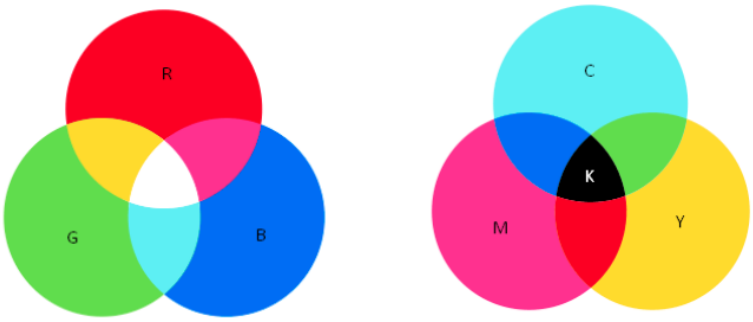
Um fato interessante é que, na teoria, o sistema CMY (sem o preto) é o sistema de cores (aditivo) complementar do sistema CIE-RGB.

Definição 12. *Duas cores são ditas* complementares *se, quando combinadas aditivamente em proporções adequadas, produzem uma cor acromática, isto é, uma cor que possui somente informação de luminância (tons de cinza).*

Assim, um sistema de cores complementar tem como primárias as cores complementares das primárias do sistema de cores original. Como exemplo, partindo do sistema padrão CIE-RGB, obtemos o sistema CMY, onde a cor complementar do vermelho é o ciano, do verde é o magenta e do azul é o amarelo.

O sistema complementar CMY, apesar de ser um sistema aditivo, “simula” o processo subtrativo de formação de cor. Isso quer dizer que a formação de cores nesse sistema se efetua através da subtração de cores primárias da cor branca. Por essa razão, na prática, o sistema complementar do sistema CIE-RGB é usado como padrão na impressão de cores em papel. A Figura 36 mostra a combinação de cores primárias no sistema CIE-RGB e no seu complementar.

Figura 36 – Representação do (a) sistema CIE-RGB e de (b) seu complementar, o sistema CMY.



Fonte – Elaborada pelo autor (2020).

APÊNDICE B – OPERAÇÕES COM IMAGENS E FILTRAGEM

Operações com imagens desempenham um papel muito importante no processamento gráfico e foram amplamente usadas para gerar as imagens distorcidas presentes neste trabalho. Por essa razão, dedicaremos este apêndice para descrever tais operações.

B.1 OPERAÇÕES ARITMÉTICAS

Dada uma imagem digital, vimos como é possível representá-la através de seu formato matricial. Dessa maneira, tal como matrizes (ou funções), imagens podem ser manipuladas numericamente utilizando operações aritméticas.

O conjunto $I = \left\{ i : U \to C \mid U \subseteq \mathbb{R}^2 \right\}$, chamado de *espaço de imagens*, possui uma estrutura natural de espaço vetorial, com as operações de soma pixel a pixel e produto por escalar. Assim, dadas duas imagens $i, g \in I$, temos que:

$$(i + g)(x, y) = i(x, y) + g(x, y) \text{ e} \tag{84}$$

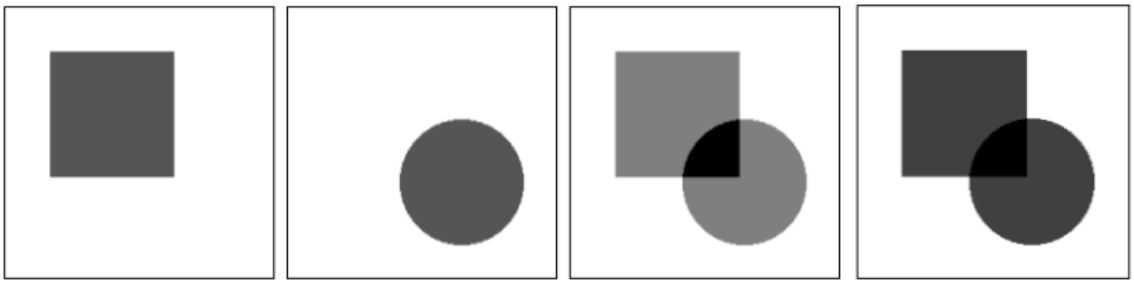
$$(k \cdot i)(x, y) = k \cdot i(x, y), \text{ em que } k \in \mathbb{R}. \tag{85}$$

Na prática, $i + g$ é o resultado da soma das intensidades das imagens i e g. Aplicações diretas dessa operação consistem de redução de ruído e adequação do brilho de uma imagem a um intervalo pré-definido.

Já a imagem $k \cdot i$ é entendida como o produto dos valores de intensidade de i pelo escalar k. Assim, os valores de intensidade de $k \cdot i$ são proporcionais a i por um fator k. Uma aplicação semelhante à adequação de brilho é também conhecida usando essa operação.

A Figura 37 ilustra as duas operações aritméticas com imagens.

Figura 37 – Operações com imagens monocromáticas: (a) i, (b) g, (c) $i+g$ e (d) $(i+g)\cdot k$.



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

Fonte – Filho e Neto (1999).

Como podemos ver em Gomes e Velho (1994), há uma aplicação bastante conhecida, que envolve a operação de adição, para "descolorir" uma imagem. Dada uma imagem $i(x, y)$, para obter uma imagem monocromática $g(x, y)$ a partir de i, devemos, primeiro, considerar i através de suas componentes no sistema RGB (como detalhado no Apêndice A). Assim, $i(x, y) = (R(x, y), G(x, y), B(x, y))$. Em seguida, define-se g como:

$$g(x, y) = 0,176 \cdot R(x, y) + 0,81 \cdot G(x, y) + 0,0011 \cdot B(x, y). \tag{86}$$

A operação realizada na Equação (86) é chamada de *operador de luminância NTSC* e é a maneira mais usada para se obter uma versão em tons de cinza de uma imagem.

B.2 FILTRAGEM

Nem sempre as operações com imagens utilizadas no processamento gráfico são operações que combinam duas ou mais imagens para gerar uma nova. Muitas vezes, operações unárias são realizadas e vamos descrevê-las melhor a partir de agora. Como descrito no Seção 2.2.2, chamamos tais operações unárias de *filtragem* e uma aplicação T que define uma filtragem no espaço de imagens I é chamada de *filtro*.

B.2.1 Resposta de impulso e convolução

Uma maneira usada para estudar o comportamento de filtros, em geral, é aplicar o filtro a um *impulso unitário*. A "função" usada para descrever tal impulso é chamada de *delta de dirac* (δ) e é descrita pelas expressões

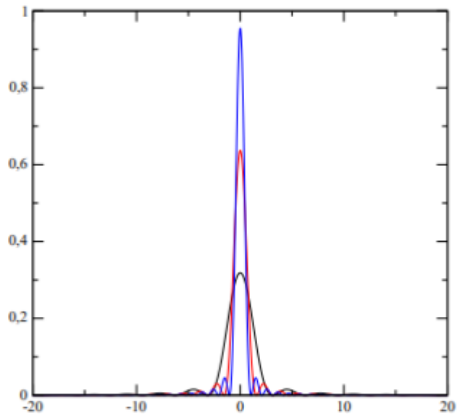
$$\delta(x - x_0) = \begin{cases} \infty, & x = x_0 \\ 0, & x \neq x_0 \end{cases} \quad \text{e} \tag{87}$$

$$\int_{-\infty}^{+\infty} \delta(x)dx = 1 \tag{88}$$

com x_0 fixado no domínio. Note que δ pode ser interpretado como uma imagem com luminância nula em todos os pontos, com exceção de x_0, onde assume um valor branco de intensidade máxima. Normalmente, utiliza-se x_0 como a origem do sistema. A Figura 38 mostra uma aproximação de δ através de limites. Mais detalhes de como a Equação (87) e a Equação (88) são obtidas através de limites, e propriedades de δ, podem ser vistos em Viana (2020).

Apesar de comumente ser chamada de função, δ não o é. Note que δ difere da função identicamente nula somente em um ponto e, portanto, seriam funções *iguais*

Figura 38 – Aproximação de δ através de limites.



Fonte – Viana (2020).

quase sempre. Sabemos, através de propriedades da integração de funções, que funções integráveis e iguais quase sempre têm integrais idênticas. No entanto, sabemos que a integral da função identicamente nula é zero, enquanto a integral de δ é 1, pela Equação (88).

Segundo Gomes e Velho (1994), chamamos de *resposta de impulso* de um filtro T a imagem $h(x, y)$ obtida ao aplicar T no impulso δ. Tal resposta de impulso nos dá a informação de quanto o sistema espalha um "ponto de luz". Por essa razão, ela é chamada também de *função de espalhamento de ponto*.

Uma das aplicações mais importantes da função espalhamento é o fato de que um filtro linear e *espacialmente invariante* (quando obtemos o mesmo resultado ao aplicarmos o filtro antes ou depois de um operador de translação) pode ser totalmente caracterizado pela função espalhamento, ou seja, pela aplicação do filtro ao impulso δ. Essa afirmação é provada através de um teorema que pode ser visto em (GOMES; VELHO, 1994). Esse teorema nos mostra não somente a veracidade da afirmação, mas também qual expressão representa essa caracterização do filtro. Dada uma imagem $i(x, y)$, um filtro T (linear e espacialmente invariante) e sua função de espalhamento h, temos que:

$$T(i(x,y)) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} i(x,y)h(u-x, v-y)dudv \tag{89}$$

Definição 13. *Dadas duas funções i e h, a operação*

$$\int_{-\infty}^{+\infty} i(x,y)h(u-x, v-y)dudv$$

é chamada de convolução entre i e h *e indicada por*

$$i * h.$$

Assim, em resumo, temos que, para um filtro linear e espacialmente invariante, a filtragem de uma imagem i pode ser obtida através da convolução entre a imagem e a função de espalhamento do filtro.

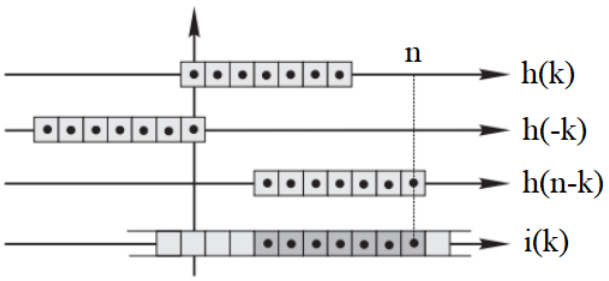
Como na prática trabalhamos com imagens digitais, ou seja, discretas, é necessário estender o conceito de filtragem através da convolução para uma versão discreta. Isso é feito, primeiro, através de uma amostragem da função de espalhamento. Para o caso discreto, a função de espalhamento h é chamada na literatura de *máscara do filtro*. Em seguida, em analogia com a Equação (89), pode ser definida a convolução entre dois sinais discretos i e h unidimensionais como:

$$(i * h)(n) = \sum_{k=-\infty}^{+\infty} i(k)h(n-k) \tag{90}$$

A convergência da soma na Equação (90) é garantida caso i ou h tenha suporte (conjunto de pontos onde o sinal não se anula) finito.

A Figura 39 mostra o cálculo da convolução para o caso em que o filtro tem resposta de impulso finita. A figura ilustra a máscara $(n-k)$ e os pixels da imagem $i(k)$ correspondentes a $h(n-k)$. Para obter a convolução $i * h$ no ponto n, os elementos correspondentes em $h(n-k)$ e $i(k)$ são multiplicados dois a dois e então somados.

Figura 39 – Representação do cálculo da convolução discreta unidimensional.



Fonte – Gomes e Velho (1994).

B.2.2 Filtragem no domínio de Fourier

A técnica de filtragem que vimos até agora é chamada de *filtragem no domínio espacial* pois o filtro atua sobre cada pixel que compõe a imagem original para resultar na imagem filtrada. Uma outra maneira de entender o processo de filtragem é através do *domínio de frequências* ou *domínio de Fourier*. Essa técnica se baseia num operador matemático chamado *transformada de Fourier*, muito usado para resolver problemas de equações diferenciais e também para questões relacionadas ao processamento de imagens, incluindo a filtragem.

Definição 14. *Dado um sinal contínuo integrável $i(x, y)$, sua transformada de Fourier é dada por*

$$\mathbb{F}(i(x,y)) = \hat{i}(u,v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} i(x,y) e^{-2\pi i(ux+vy)} dxdy \tag{91}$$

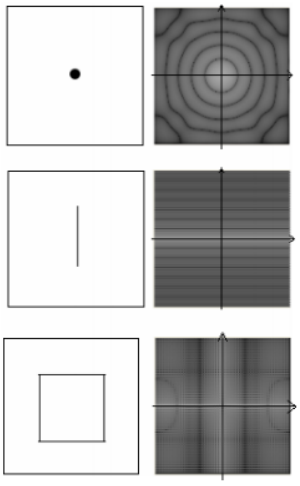
para todo $(u, v) \in \mathbb{R}^2$, desde que a integral convirja.

Como podemos ver em (GOMES; VELHO, 1994) o operador da Equação (91) cumpre o papel de detectar frequências no sinal i.

Uma imagem pode ser entendida tanto da maneira usual (através do domínio espacial) como através do domínio de frequências, depois de passar a imagem pela transformada de Fourier. Assim como o primeiro modelo foi chamado de modelo espacial, este último é chamado de *modelo espectral*.

A Figura 40 mostra alguns exemplos de imagens simples e seus respectivos espectros do domínio de frequências. Mais detalhes de como esses espectros são interpretados podem ser encontrados em Viola (2020).

Figura 40 – Algumas imagens e seus respectivos espectros do domínio de frequências.



Fonte – Viola (2020). Modificado pelo autor (2020).

Teorema 15 (Inversa da transformada de Fourier)**.** *Sejam $i(x, y)$ sinal contínuo integrável e $\hat{i}(x, y)$ de modo que $\hat{i}(x, y)$ seja a transformada de Fourier de $i(x, y)$, então:*

$$i(x,y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \hat{i}(u,v) e^{2\pi i(ux+vy)} dudv \tag{92}$$

Demonstração. (OSGOOD, 2014), p. 72. □

Perceba, da Equação (92), que a operação transformada de Fourier é inversível e, ainda, temos a fórmula da sua inversa.

Uma das razões que motivam o processamento de imagens no domínio de Fourier é o fato de que, neste domínio, a filtragem se dá através de uma operação de multiplicação ao invés da convolução como ocorre no domínio espacial. Mais especificamente, usamos o *teorema da convolução*, para realizar essa operação.

Teorema 16 (Teorema da Convolução)**.** *Sejam i e h dois sinais integráveis e $\mathbb{F}(i)$ e $\mathbb{F}(h)$ suas transformadas de Fourier, respectivamente. Temos que:*

$$\mathbb{F}(i * h) = \mathbb{F}(i) \cdot \mathbb{F}(i) \tag{93}$$

Demonstração. (OSGOOD, 2014), p. 192. □

Portanto, ao aplicarmos a transformada inversa na Equação (93) temos

$$i * h = \mathbb{F}^{-1}(\mathbb{F}(i) \cdot \mathbb{F}(h)) \tag{94}$$

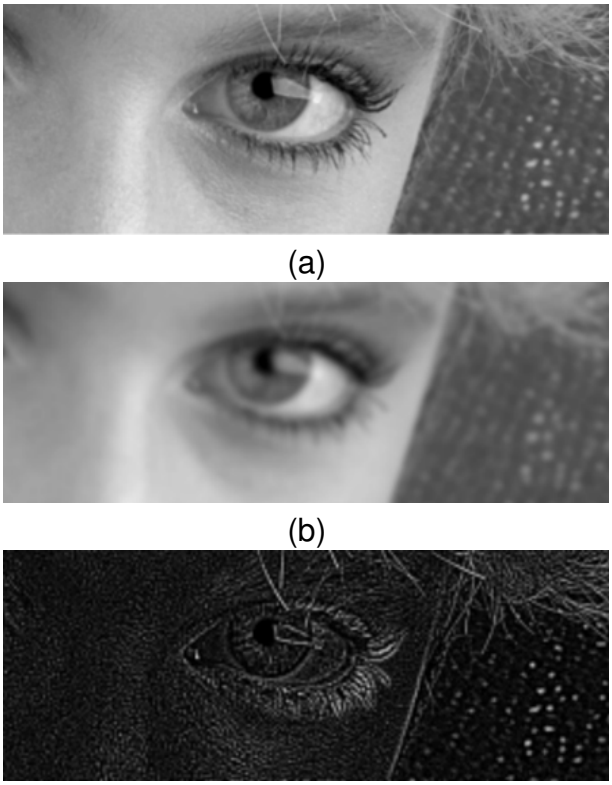
Em resumo, o teorema da convolução nos diz que, no domínio de frequências, aplicar um filtro h a uma imagem i equivale a multiplicar as transformadas de i e h.

Entender melhor como as frequências se comportam em uma imagem é um ponto chave para entendermos como os filtros se comportam no domínio de frequências. A primeira coisa que devemos perceber é que a maior parte da informação de uma imagem (também chamada de *força* de uma imagem) é constituída das baixas frequências do domínio de Fourier, enquanto os detalhes da imagem são responsáveis pelas altas frequências. Em geral, mais de 90% da força de uma imagem se encontra em uma região muito pequena do domínio de frequência, onde as frequências mais baixas se encontram. A Figura 41 mostra uma imagem e o que acontece quando retiramos as baixas e as altas frequências dessa imagem.

Essa informação abre a possibilidade de filtrar uma imagem usando a informação da ocorrência de determinadas frequências nessa imagem. Dessa maneira, existem filtros chamados de *passa-baixa*, *passa-alta* e *passa-banda*.

- Filtros passa-baixa preservam as baixas frequências de uma imagem. Assim, detalhes são perdidos enquanto a força da imagem é preservada, dando o aspecto de imagem borrada, como na Figura 41(b).
- Filtros passa-alta preservam as altas frequências. Dessa maneira, somente detalhes como bordas, traços, lados e outras transições abruptas são deixadas na imagem filtrada, como na Figura 41(c)
- Filtros passa-banda são usados quando se quer preservar apenas uma determinada região do domínio de frequências.

Figura 41 – (a) Imagem original seguida da (b) mesma imagem sem as altas frequências e (c) sem as baixas frequências.



(a)

(b)

(c)

Fonte – Gomes e Velho (1994).

A seguir, vamos apresentar alguns dos filtros mais conhecidos na computação gráfica, bem como sua atuação no domínio de frequências.

B.2.3 Filtro de borramento gaussiano

Um filtro passa-baixa muito conhecido é o filtro de borramento gaussiano. Inspirado na distribuição gaussiana, da probabilidade e estatística, temos que este filtro é definido por:

$$G_\sigma(x, y) = \frac{1}{2\sigma^2\pi} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{95}$$

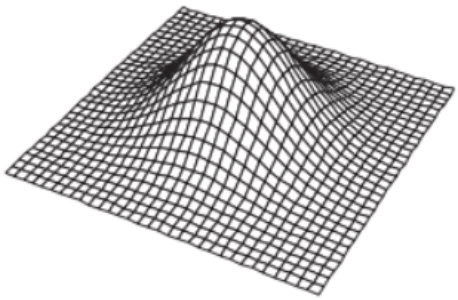
cuja média é 0 e tem variância σ. Seu gráfico pode ser visto na Figura 42.

O filtro gaussiano é um filtro passa-baixa já que ao aplicarmos a transformada de Fourier nele, obtemos novamente uma função gaussiana e assim, as altas frequências da imagem filtrada são atenuadas de maneira exponencial.

Gomes e Velho (1994) apresentam uma máscara para esse filtro, dada pela matriz 5×5:

$$\frac{1}{256} \cdot \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Figura 42 – Gráfico do filtro gaussiano de média 0 e variância 2.



Fonte – Gomes e Velho (1994).

B.2.4 Detecção de bordas

De acordo com Filho e Neto (1999), define-se borda como a fronteira entre duas regiões cujos níveis de cinza predominantes são razoavelmente diferentes. Como vimos anteriormente, ao passar uma imagem do domínio espacial para o domínio de Fourier, a ocorrência de bordas nessa imagem será encontrada nas regiões de altas frequências e há filtros passa-alta bastante conhecidos que são responsáveis por analisar a ocorrência de tais bordas.

Wangenheim (2020) explicam que depois que uma borda é definida (seja por uma mudança no nível de cinza, seja quando ocorre uma descontinuidade na intensidade ou quando o gradiente da imagem tem uma variação abrupta), um operador que é sensível a estas mudanças operará como um detector de bordas. Entre os filtros mais usados se encontram os filtros de Sobel, Roberts, Prewitt e Frei-Chen. Definições e propriedades desses operadores podem ser encontrados em (FILHO; NETO, 1999) e (WANGENHEIM, 2020).

A Figura 43 apresenta máscaras utilizadas para convolução com imagens digitais para se obter uma filtragem de detecção de bordas.

A imagem da Figura 41(c), por exemplo, foi obtida ao utilizar um filtro de detecção de bordas na imagem original (Figura 41(a)).

Figura 43 – Filtros mais utilizados para detecção de bordas.

Operador	Vertical	Horizontal
Roberts	$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Sobel	$\frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
Prewitt	$\frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$
Frei-Chen	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$

Fonte – Filho e Neto (1999).

B.2.5 Adição de ruídos

Durante o processo de captura ou de processamento de uma imagem, é comum a ocorrência de ruídos que adicionam informações indesejadas à imagem, como bordas irrealistas, linhas invisíveis, objetos borrados além de perturbar as cenas de fundo. Geralmente ocorre na forma de uma variação aleatória das informações de brilho ou cor nas imagens. Por ser um processo que adiciona altas frequências, existem muitos estudos, envolvendo filtros passa-baixa, a fim de fazer a remoção desses ruídos. Mas uma parte essencial desses estudos está ligada a entender os ruídos e modelá-los matematicamente. Isso envolve o desenvolvimento de filtros que adicionam tais ruídos à imagem.

O operador que adiciona ruído gaussiano perturba aleatoriamente os valores de cor das imagens digitais. É por isso que o ruído gaussiano é essencialmente modelado e caracterizado pela sua função densidade de probabilidade, que descreve a probabilidade do valor de cor de um pixel ser perturbado, de acordo com a função:

$$P(g) = \sqrt{\frac{1}{2\pi\sigma^2}} e^{-\frac{(g-\mu)^2}{2\sigma^2}} \tag{96}$$

em que g é a variável que representa o valor de cor do pixel, σ é o desvio padrão e μ a média.

Outro tipo de ruído aditivo bastante comum é o ruído impulsivo, também conhecido como ruído do tipo “sal e pimenta”. Ao contrário do ruído gaussiano, no ruído sal e pimenta a imagem não é totalmente corrompida, apenas alguns valores de pixel são alterados na imagem. Neste, os valores de alguns pixel da imagem são substituí-

dos aleatoriamente por valores corrompidos, tanto para o valor máximo quanto para o mínimo, ou seja, 255 ou 0, para imagens em 8 bits.

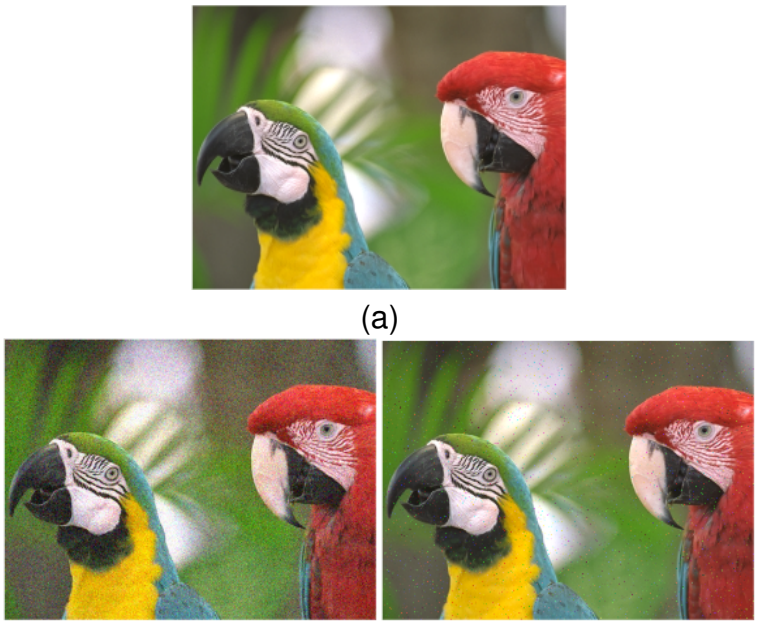
Figura 44 – Imagem digital 3 $\times$ 3 com pixel central corrompido por ruído do tipo “sal e pimenta”

Fonte – Boyat e Joshi (2015).

Por exemplo, vamos considerar matrizes que representam imagens 3 $\times$ 3, como mostradas na Figura 44. Supondo que o valor central das matrizes foi corrompido pelo ruído sal e pimenta, temos que esse valor é substituído pelo valor 0.

A Figura 45 mostra uma imagem corrompida por ruídos do tipo gaussiano e sal e pimenta.

Figura 45 – (a) Imagem original e imagens corrompidas por ruídos do tipo: (b) gaussiano e (c) sal e pimenta.



(a)

(b) (c)

Fonte – Ponomarenko *et al.* (2009).

APÊNDICE C – CÓDIGO DE IMPLEMENTAÇÃO

Apresentamos o código usado para obter os resultados do Capítulo 6. A sintaxe utilizada é a sintaxe do software *MATLAB* R2017b.

Tudo que aparece escrito após um símbolo de porcentagem, como, por exemplo, %Exemplo de comentário, é considerado comentário e não tem efeito na execução do código. O uso de comentários visa auxiliar o entendimento do código.

Esta é uma adaptação do código utilizado, pois chegamos a resolver o problema de otimização com 1700 pares de imagens. Porém, como ficou muito extenso, seria inviável colocá-lo integralmente aqui. Assim, mostramos como implementaríamos se usássemos 136 pares de imagens, ou seja, se minimizássemos a função $E_{136}(w)$. Para minimizar outras funções, como $E_{1700}(w)$, a implementação é análoga.

C.0.1 Procura pela janela deslizante ótima de tamanho 5×5 no MATLAB usando 136 pares de imagens

```
1
2  %Leitura das 2 imagens de referencia
3  for n=1:2
4  image_ref{n} = imread(sprintf('i0%d.bmp',n));
5  image_ref{n} = image_ref{n}(:,:,1);
6  image_ref{n}=double(image_ref{n});
7  end
8
9  %Leitura de 136 imagens distorcidas
10 %Sao 68 imagens distorcidas para cada uma das imagens de
       referencia
11 for n=1:136
12 image{n} = imread(sprintf('image (%d).bmp',n));
13 image{n} = image{n}(:,:,1);
14 image{n}=double(image{n});
15 end
16
17 %Definicao de constantes
18 C1 = 6.5025;
19 C2 = 58.5225;
20
21
22 %Definicao da metrica SSIM com as variaveis estatisticas usadas
       em seu
```

```
23 %calculo
24 for i=1:2
25
26 x{i}=image_ref{i};
27
28 mux{i} =@(w) filter2([w(1) w(2) w(3) w(2) w(1); w(2) w(3) w(4)
      w(3) w(2); w(3) w(4) 1-(4*w(1))-(8*w(2))-(8*w(3))-(4*w(4)) w
      (4) w(3); w(2) w(3) w(4) w(3) w(2); w(1) w(2) w(3) w(2) w(1)
      ], x{i}, 'valid');
29 sigmax_sq{i}=@(w) filter2([w(1) w(2) w(3) w(2) w(1); w(2) w(3)
      w(4) w(3) w(2); w(3) w(4) 1-(4*w(1))-(8*w(2))-(8*w(3))-(4*w
      (4)) w(4) w(3); w(2) w(3) w(4) w(3) w(2); w(1) w(2) w(3) w
      (2) w(1)], x{i}.^2, 'valid') - mux{i}(w).^2;
30
31 end
32
33 for i=1:68
34
35 y{1,i}=image{i};
36
37 muy{1,i} =@(w) filter2([w(1) w(2) w(3) w(2) w(1); w(2) w(3) w
      (4) w(3) w(2); w(3) w(4) 1-(4*w(1))-(8*w(2))-(8*w(3))-(4*w
      (4)) w(4) w(3); w(2) w(3) w(4) w(3) w(2); w(1) w(2) w(3) w
      (2) w(1)], y{1,i}, 'valid');
38 sigmay_sq{1,i}=@(w) filter2([w(1) w(2) w(3) w(2) w(1); w(2) w
      (3) w(4) w(3) w(2); w(3) w(4) 1-(4*w(1))-(8*w(2))-(8*w(3))
      -(4*w(4)) w(4) w(3); w(2) w(3) w(4) w(3) w(2); w(1) w(2) w
      (3) w(2) w(1)], y{1,i}.^2, 'valid') - muy{1,i}(w).^2;
39
40 sigmaxy{1,i}=@(w) filter2([w(1) w(2) w(3) w(2) w(1); w(2) w(3)
      w(4) w(3) w(2); w(3) w(4) 1-(4*w(1))-(8*w(2))-(8*w(3))-(4*w
      (4)) w(4) w(3); w(2) w(3) w(4) w(3) w(2); w(1) w(2) w(3) w
      (2) w(1)], x{1}.*y{1,i}, 'valid') - mux{1}(w).*muy{1,i}(w);
41
42 ssim_map{1,i}=@(w) ((2.*mux{1}(w).*muy{1,i}(w) + C1).*(2.*
      sigmaxy{1,i}(w) + C2))./((mux{1}(w).^2 + muy{1,i}(w).^2 + C1
      ).*(sigmax_sq{1}(w) + sigmay_sq{1,i}(w) + C2));
43
44 ssim{1,i}=@(w) mean(mean(ssim_map{1,i}(w)));
```

```
45
46 end
47
48 for i=69:136
49
50 y{2,i}=image{i};
51
52 muy{2,i} =@(w) filter2([w(1) w(2) w(3) w(2) w(1); w(2) w(3) w
       (4) w(3) w(2); w(3) w(4) 1-(4*w(1))-(8*w(2))-(8*w(3))-(4*w
       (4)) w(4) w(3); w(2) w(3) w(4) w(3) w(2); w(1) w(2) w(3) w
       (2) w(1)], y{2,i}, 'valid');
53 sigmay_sq{2,i}=@(w) filter2([w(1) w(2) w(3) w(2) w(1); w(2) w
       (3) w(4) w(3) w(2); w(3) w(4) 1-(4*w(1))-(8*w(2))-(8*w(3))
       -(4*w(4)) w(4) w(3); w(2) w(3) w(4) w(3) w(2); w(1) w(2) w
       (3) w(2) w(1)], y{2,i}.^2, 'valid') - muy{2,i}(w).^2;
54
55 sigmaxy{2,i}=@(w) filter2([w(1) w(2) w(3) w(2) w(1); w(2) w(3)
      w(4) w(3) w(2); w(3) w(4) 1-(4*w(1))-(8*w(2))-(8*w(3))-(4*w
       (4)) w(4) w(3); w(2) w(3) w(4) w(3) w(2); w(1) w(2) w(3) w
       (2) w(1)], x{2}.*y{2,i}, 'valid') - mux{2}(w).*muy{2,i}(w);
56
57 ssim_map{2,i}=@(w) ((2.*mux{2}(w).*muy{2,i}(w) + C1).*(2.*
       sigmaxy{2,i}(w) + C2))./((mux{2}(w).^2 + muy{2,i}(w).^2 + C1
       ).*(sigmax_sq{2}(w) + sigmay_sq{2,i}(w) + C2));
58
59 ssim{2,i}=@(w) mean(mean(ssim_map{2,i}(w)));
60
61 end
62
63 %Leitura das pontuacoes MOS referentes as imagens distorcidas
64 valores_mos=[0.663400000 0.601855556 0.506177778 0.479366667
       0.682544444 0.643788889 0.604933333 0.546033333 0.533955556
       0.447622222 0.391977778 0.305555556 0.669844444 0.663488889
       0.638100000 0.536511111 0.650788889 0.623455556 0.539677778
       0.393655556 0.496733333 0.469844444 0.447711111 0.367288889
       0.530866667 0.456788889 0.355555556 0.240744444 0.526144444
       0.434922222 0.364200000 0.241833333 0.634922222 0.546033333
       0.333333333 0.133333333 0.660488889 0.574077778 0.367288889
       0.161900000 0.644444444 0.574077778 0.395066667 0.111111111
```

```
    0.713800000 0.478400000 0.456788889 0.376544444 0.463488889
    0.424833333 0.390477778 0.352944444 0.704766667 0.704766667
    0.632711111 0.512344444 0.279366667 0.330066667 0.428566667
    0.475311111 0.660133333 0.625400000 0.654322222 0.539677778
    0.722222222 0.552377778 0.688266667 0.441355556 0.700000000
    0.628477778 0.548611111 0.451611111 0.685822222 0.718755556
    0.609322222 0.559033333 0.537633333 0.448144444 0.405022222
    0.343755556 0.688166667 0.666666667 0.586811111 0.505377778
    0.777777778 0.645166667 0.541666667 0.388888889 0.510422222
    0.468755556 0.444444444 0.388888889 0.635422222 0.473122222
    0.405022222 0.322922222 0.630822222 0.523300000 0.444444444
    0.347222222 0.631944444 0.545144444 0.433688889 0.247311111
    0.680555556 0.645166667 0.381944444 0.284722222 0.562500000
    0.326166667 0.137033333 0.000000000 0.501788889 0.369177778
    0.312500000 0.207888889 0.483866667 0.414811111 0.302088889
    0.408600000 0.695344444 0.618055556 0.576388889 0.461811111
    0.361111111 0.409722222 0.458333333 0.491044444 0.697922222
    0.611111111 0.600700000 0.570366667 0.725700000 0.579866667
    0.570366667 0.468755556];
65
66 %Definicao da funcao-objetivo
67 for i=1:68
68 erro_aux{i}=@(w) (ssim{1,i}(w)- valores_mos(i))^2;
69 end
70
71 for i=69:136
72 erro_aux{i}=@(w) (ssim{2,i}(w)- valores_mos(i))^2;
73 end
74
75 erro=@(w) = erro_aux{1}(w) + erro_aux{2}(w) + erro_aux{3}(w) +
    erro_aux{4}(w) + erro_aux{5}(w) + erro_aux{6}(w) + erro_aux
    {7}(w) + erro_aux{8}(w) + erro_aux{9}(w) + erro_aux{10}(w) +
     erro_aux{11}(w) + erro_aux{12}(w) + erro_aux{13}(w) +
    erro_aux{14}(w) + erro_aux{15}(w) + erro_aux{16}(w) +
    erro_aux{17}(w) + erro_aux{18}(w) + erro_aux{19}(w) +
    erro_aux{20}(w) + erro_aux{21}(w) + erro_aux{22}(w) +
    erro_aux{23}(w) + erro_aux{24}(w) + erro_aux{25}(w) +
    erro_aux{26}(w) + erro_aux{27}(w) + erro_aux{28}(w) +
    erro_aux{29}(w) + erro_aux{30}(w) + erro_aux{31}(w) +
```

```
    erro_aux{32}(w) + erro_aux{33}(w) + erro_aux{34}(w) +
    erro_aux{35}(w) + erro_aux{36}(w) + erro_aux{37}(w) +
    erro_aux{38}(w) + erro_aux{39}(w) + erro_aux{40}(w) +
    erro_aux{41}(w) + erro_aux{42}(w) + erro_aux{43}(w) +
    erro_aux{44}(w) + erro_aux{45}(w) + erro_aux{46}(w) +
    erro_aux{47}(w) + erro_aux{48}(w) + erro_aux{49}(w) +
    erro_aux{50}(w) + erro_aux{51}(w) + erro_aux{52}(w) +
    erro_aux{53}(w) + erro_aux{54}(w) + erro_aux{55}(w) +
    erro_aux{56}(w) + erro_aux{57}(w) + erro_aux{58}(w) +
    erro_aux{59}(w) + erro_aux{60}(w) + erro_aux{61}(w) +
    erro_aux{62}(w) + erro_aux{63}(w) + erro_aux{64}(w) +
    erro_aux{65}(w) + erro_aux{66}(w) + erro_aux{67}(w) +
    erro_aux{68}(w) + erro_aux{69}(w) + erro_aux{70}(w) +
    erro_aux{71}(w) + erro_aux{72}(w) + erro_aux{73}(w) +
    erro_aux{74}(w) + erro_aux{75}(w) + erro_aux{76}(w) +
    erro_aux{77}(w) + erro_aux{78}(w) + erro_aux{79}(w) +
    erro_aux{80}(w) + erro_aux{81}(w) + erro_aux{82}(w) +
    erro_aux{83}(w) + erro_aux{84}(w) + erro_aux{85}(w) +
    erro_aux{86}(w) + erro_aux{87}(w) + erro_aux{88}(w) +
    erro_aux{89}(w) + erro_aux{90}(w) + erro_aux{91}(w) +
    erro_aux{92}(w) + erro_aux{93}(w) + erro_aux{94}(w) +
    erro_aux{95}(w) + erro_aux{96}(w) + erro_aux{97}(w) +
    erro_aux{98}(w) + erro_aux{99}(w) + erro_aux{100}(w) +
    erro_aux{101}(w) + erro_aux{102}(w) + erro_aux{103}(w) +
    erro_aux{104}(w) + erro_aux{105}(w) + erro_aux{106}(w) +
    erro_aux{107}(w) + erro_aux{108}(w) + erro_aux{109}(w) +
    erro_aux{110}(w) + erro_aux{111}(w) + erro_aux{112}(w) +
    erro_aux{113}(w) + erro_aux{114}(w) + erro_aux{115}(w) +
    erro_aux{116}(w) + erro_aux{117}(w) + erro_aux{118}(w) +
    erro_aux{119}(w) + erro_aux{120}(w) + erro_aux{121}(w) +
    erro_aux{122}(w) + erro_aux{123}(w) + erro_aux{124}(w) +
    erro_aux{125}(w) + erro_aux{126}(w) + erro_aux{127}(w) +
    erro_aux{128}(w) + erro_aux{129}(w) + erro_aux{130}(w) +
    erro_aux{131}(w) + erro_aux{132}(w) + erro_aux{133}(w) +
    erro_aux{134}(w) + erro_aux{135}(w) + erro_aux{136}(w);
76
77 %Definicao do ponto inicial
78 w0=[0 0 0 0];
79
```

```
80 %Otimizacao
81 %As funcoes "tic" e "toc" servem para contabilizar o tempo da
       otimizacao
82  tic; w_estrela=fminsearch(erro,w0); toc;
```