UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE COMPUTAÇÃO
CURSO DE CIÊNCIA DA COMPUTAÇÃO

MAURICIO ANTONIO GOIS DE ALMEIDA

ESTUDO COMPARATIVO DE TÉCNICAS DE DESBORRAMENTO DE IMAGEM

TRABALHO DE CONCLUSÃO DE CURSO

MEDIANEIRA

2022

MAURICIO ANTONIO GOIS DE ALMEIDA

ESTUDO COMPARATIVO DE TÉCNICAS DE DESBORRAMENTO DE IMAGEM

Trabalho de Conclusão de Curso apresentado ao Departamento Acadêmico de Computação da Universidade Tecnológica Federal do Paraná como requisito parcial para obtenção do título de "Bacharel em Ciência da Computação".

Orientador: Prof. Dr. Pedro Luiz de Paula Filho

Co-orientador: Prof. Dr. Arnaldo Candido Junior

MEDIANEIRA

2022

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Diretoria de Graduação e Educação Profissional
Coordenação do Curso de Ciência da Computação

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

TERMO DE APROVAÇÃO

ESTUDO COMPARATIVO DE TÉCNICAS DE DESBORRAMENTO DE IMAGEM

Por

MAURICIO ANTONIO GOIS DE ALMEIDA

Este Trabalho de Conclusão de Curso foi apresentado às 13:00 do dia 12 de Agosto de 2021 como requisito parcial para a obtenção do título de Bacharel no Curso de Ciência da Computação, da Universidade Tecnológica Federal do Paraná, Câmpus Medianeira. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. Pedro Luiz de Paula Filho
UTFPR - Câmpus Medianeira

Prof. Jorge Aikes Junior
UTFPR - Câmpus Medianeira

Prof. Evando Carlos Pessini
UTFPR - Câmpus Medianeira

A folha de aprovação assinada encontra-se na Coordenação do Curso.

RESUMO

ALMEIDA, Mauricio Antonio Gois. ESTUDO COMPARATIVO DE TÉCNICAS DE DESBORRAMENTO DE IMAGEM. 64 f. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade Tecnológica Federal do Paraná. Medianeira, 2022.

A baixa capacidade das câmeras de segurança pode afetar uma investigação pericial devido à problemas de borramento por movimento ou falta de foco. Para solucionar problemas de borramento podem ser utilizadas técnicas de desborramento clássicas ou que utilizam inteligência artificial. O objetivo desse trabalho é comparar as duas abordagens, contribuindo para a análise de imagens. Foram implementados o Filtro de Wiener e o Algoritmo de Richardson-Lucy, e treinada uma Rede Neural Adversária para o desborramento das imagens. A comparação das duas abordagens utilizou duas métricas de qualidade, Relação sinal-ruído de pico (PSNR) e Índice de Similaridade Estrutural (SSIM). A Rede Neural em todos os parâmetros testados, obteve as melhores médias de SSIM e PSNR, variando em média a similariade de 0,6786 a 0,86024, em comparação com as outras duas técnicas. Já comparando somente as técnicas de processamento de imagem, Richardson-Lucy foi a que teve as melhores médias de desempenho, com as médias de SSIM variando de 0,1405 a 0,5642 em relação a Filtro de Wiener que variou de 0,081 a 0,1251.

Palavras-chave: Processamento de Imagens, Redes neurais (Computação), Reconstrução de imagens

ABSTRACT

ALMEIDA, Mauricio Antonio Gois. COMPARATIVE STUDY OF IMAGE DEBLURRING TECHNIQUES. 64 f. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade Tecnológica Federal do Paraná. Medianeira, 2022.

The low capacity of security cameras can affect an forensic investigation due to motion blur issues or lack of focus. To solve blurring problems, classic blurring techniques or those using artificial intelligence can be used. The objective of this work is to compare the two approaches, contributing to image analysis. The Wiener Filter and the Richardson-Lucy Algorithm were implemented, and an Adversary Neural Network was trained to deblur the images. The comparison of the two approaches used two quality metrics, Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). The Neural Network in all tested parameters, obtained the best averages of SSIM and PSNR, ranging on average the similarity from 0.6786 to 0.86024, compared to the other two techniques. Comparing only the image processing techniques, Richardson-Lucy had the best performance averages, with SSIM averages ranging from 0.1405 to 0.5642 in relation to Wiener Filter, which ranged from 0.081 to 0, 1251.

Keywords: Image processing, Neural Networks, Reconstruction of images

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

FFT	Fast Fourier Transform
FW	Filtro de Wiener
MSE	Mean Squared Error
PDI	Processamento Digital de Imagem
PSNR	Peak Signal to Noise Ratio
RL	Richardson-Lucy
RNA	Redes Neurais Artificiais
SSIM	Structural Similarity Index

SUMÁRIO

1 INTRODUÇÃO

A população, movida pela insegurança nos últimos anos, aumentou a procura por serviços de segurança residencial de forma significativa. Em 2017, residências ocuparam 69% dos projetos dos prestadores de serviços, e uma das principais demandas eram voltadas ao monitoramento por vídeo (ABESE, 2018). O modelo bastante utilizado por esses prestadores de serviço é a locação das câmeras, e segundo a presidente da ABESE (Associação Brasileira das Empresas de Sistemas Eletrônicos), Selma Migliori, muitas empresas não atualizam seus equipamentos com as últimas tecnologias e usam câmeras obsoletas que deixam a desejar na qualidade da imagem. Essa baixa capacidade pode afetar uma investigação pericial, e um dos problemas característicos que se enfrenta é o "borramento" por movimento, ou falta de foco. Uma maneira de solucionar isso, é digitalizar as imagens para um software que aplicará técnicas de *deblurring*, para auxiliar na identificação de objetos e/ou pessoas na figura (JERIAN et al., 2007).

Existem diferentes maneiras de solucionar esse problema, podendo ser utilizadas técnicas clássicas ou algoritmos de inteligência artificial. Uma técnica clássica e bastante utilizada é a filtragem de mínimo erro quadrático médio, ou filtro de Wiener, focada na restauração de imagens bidimensionais e sinais unidimensionais, apesar de ser sensível ao ruído, é possível obter bons resultados a partir desse procedimento (ACHARYA, 2007). Outro método clássico que também atrai bons resultados são os métodos iterativos, no qual pode se destacar o algoritmo de Richardson-Lucy, que tem como vantagem a simplicidade e a facilidade na implementação (MOSQUERA, 2015). Uma abordagem mais moderna, que consegue resultados favoráveis no processamento de imagem é o aprendizado profundo (do inglês *deep learning*) e vem se tornando cada vez mais popular, podendo citar como exemplo sistemas que tratam imagens, vídeos e áudios (KAMILARIS; PRENAFETA-BOLDÚ, 2018).

Um dos algoritmos de aprendizado profundo, utilizado no processamento de imagem, é a rede neural convolucional, que tem a capacidade de receber uma imagem e diferenciar objetos contidos nela. Pode ainda aprender, a partir de um treinamento, e extrair características que antes eram extraídas manualmente (Data Science Academy, 2019).

Outra abordagem de aprendizado profundo, que tem a capacidade de lidar com

imagens são as Redes Adversárias Generativas, elas podem conservar detalhes das imagens. Segundo Kupyn et al. (2018), elas possuem bons resultados com problemas envolvendo resolução de imagem e tradução de uma imagem para outra.

Recentemente o aprendizado profundo trouxe avanços para resoluções de problemas como o desborramento (BIGDELI; ZWICKER, 2018).

Outro fator que vale ressaltar é uma demanda proposta pelos peritos da Polícia Federal de integrar tal solução ao Peritus. Peritus é um software proposto por peritos criminais federais, tem como vantagem a rapidez na produção de laudos, permite realizar comparação facial, fazer uma análise do conteúdo de um vídeo e a previsão é que receba novas funcionalidades,(Agência Brasil, 2019). Nesse cenário, busca-se com esse trabalho avaliar, utilizando métricas de similaridade as técnicas clássicas e Redes Neurais Artificiais para restauração de imagens ("desborramento").

1.1 OBJETIVOS GERAL E ESPECÍFICOS

O objetivo geral desse trabalho é utilizar técnicas clássicas de restauração de imagem ("desborramento") e Redes Neurais Artificiais, com o propósito de comparar qual abordagem tem melhores resultados afim de contribuir para a análise de imagens. Esse objetivo principal pode ser dividido nos seguintes objetivos específicos:

- Borrar uma base de imagens com diferentes ângulos de borrão para implementação das técnicas e treinamento de uma Rede Neural Artificial
- Testar o desempenho das técnicas clássicas de *deblurring* na base de imagens;
- Testar o desempenho de uma RNA para o "desborramento" de imagens;
- Comparar e Analisar os desempenhos das técnicas usando a mesma base de dados;

1.2 JUSTIFICATIVA

A restauração da imagem (*deblurring*) é necessária em diversas áreas, como, biomedicina, em imagens astronômicas, na fotografia, imagens médicas, entre outras

(CAMPISI; EGIAZARIAN, 2017). Outra área onde há uma exigência para restauração da imagem, é na investigação criminal, pois uma das principais fontes de informações são as cenas gravadas por câmeras de monitoramento. Porém esse equipamento, devido ao custo, geralmente possui uma baixa qualidade na gravação, o que pode dificultar na investigação (JERIAN et al., 2007). Especificamente, a restauração de imagem a partir de técnicas de *deblurring* ajudam a solucionar o problema. Portanto, esse trabalho visa analisar e comparar o desempenho de algumas técnicas de restauração de imagem afim de simplificar a obtenção da imagem com pouca degradação. . Nesse contexto justifica-se o estudo de diferentes técnicas de desborramento para o auxilio desses problemas.

2 REFERENCIAL TEÓRICO

Nessa seção é descrito o estado da arte do tema escolhido. Primeiramente será exposto um referencial teórico sobre processamento digital de imagens, conceitos de imagem digital, será apresentado a degradação em uma figura e técnicas de restauração, além disso, serão apresentadas noções sobre aprendizado profundo, redes neurais artificiais e convolucionais.

2.1 IMAGEM DIGITAL

Uma imagem é um sinal analógico bidimensional, e é composta basicamente por três elementos: a iluminação, os modelos de refletância das superfícies fotografadas e também do processo de formação da imagem na retina dos olhos humanos ou no plano do sensor da câmera (ACHARYA, 2007).

Matematicamente falando, imagem monocromática pode ser definida por uma função contínua $f(x,y)$, onde x e y são coordenadas espaciais e o resultado da função em qualquer ponto é correspondente à intensidade da luz no local indicado. Porém os computadores não processam imagens contínuas, apenas arranjos de números digitais, logo é exigido demonstrar imagens como vetores bidimensionais de pontos, onde cada ponto é chamado de *pixel* (QUEIROZ; GOMES, 2001).

Para conversão da imagem contínua em digital, são necessários dois processos: a amostragem, que é a digitalização dos valores das coordenadas, e a quantização, que é a digitalização dos valores de amplitude (GONZALEZ; WOODS, 2011).

2.1.1 Amostragem e quantização

Para ilustrar os processos de amostragem e quantização, a Figura 1 representa a ideia básica desses procedimentos. A Figura 1(a) retrata a imagem contínua ao longo do segmento de reta *AB* e a partir dela pode-se aplicar os métodos para digitalização. A Figura 1(b) é uma função unidimensional, representada na forma de gráfico, mostrando os valores de amplitude da imagem ao longo do segmento, vale destacar que as variações aleatórias foram motivadas pelo ruído da imagem. A partir dessa função, é possível realizar a amostragem, dividindo-a em porções iguais ao longo da reta *AB*, como pode ser visto na parte inferior da Figura 1(c), representados por quadrados brancos, formando a função de amostragem. Porém ainda há informação contínua nessa imagem, são os valores de intensidade que precisam ser digitalizados, por meio da quantização que divide em oito partes, variando entre preto e branco, o nível de intensidade atribuindo um valor dependendo da proximidade de uma mostra da marca vertical indicada no lado direito da Figura 1(c). O resultado desses dois processos pode ser visualizado na Figura 1(d). Vale ressaltar que a precisão atingida na quantização depende do ruído do sinal da amostragem (GONZALEZ; WOODS, 2011).

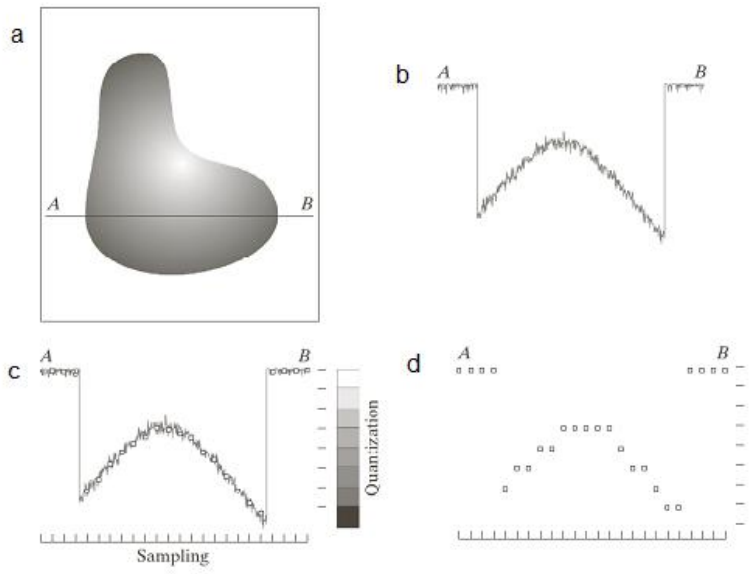


Figura 1 – Produzindo uma imagem digital. (a)Imagem contínua. (b)Linha de varredura de *A* e *B* na imagem contínua utilizada para ilustrar os conceitos de amostragem e quantização. (c)Amostragem e quantização. (d)Linha de varredura digital.

Fonte: Gonzalez e Woods (2011)

Esse processo que foi descrito anteriormente é utilizado quando a imagem é gerada por

um único elemento sensor juntamente com um movimento mecânico, ou seja, o procedimento de amostragem e quantização depende do arranjo de sensores. Caso utilize uma composição de sensores por varredura de linha, a quantidade deles definem as limitações da amostragem na direção da imagem, o movimento mecânico que é formado na outra direção pode ser controlado com mais precisão. Quando um arranjo matricial de sensores é utilizado para geração de imagens, não há um movimento mecânico, e o número de sensores limita a amostragem em todas as direções (GONZALEZ; WOODS, 2011).

Uma outra abordagem define a amostragem com o processo de discretização do domínio de definição da imagem nas direções x e y, gerando uma matriz M x N de amostras, e a quantização como um processo que define um número inteiro de níveis de cinza, se tratando de uma imagem monocromática, permitidos para cada ponto da imagem (PEDRINI; SCHWARTZ, 2008).

2.2 PROCESSAMENTO DIGITAL DE IMAGEM

Segundo Gonzalez e Woods (2011) “o campo do processamento de digital de imagens se refere ao processamento digital de imagens por um computador digital”. O domínio dessa área é bastante amplo, pois pode-se trabalhar com diversos tipos de imagens, geradas por fontes que muitas vezes os humanos não relacionam como uma figura, como ultrassom, microscopia eletrônica entre outras. Logo com um vasto e variado âmbito de utilização, não se sabe ao certo os limites de estudo e onde começam as outras áreas. Uma forma de delimitação é afirmar que se em um processo a entrada e a saída dele for uma imagem, conclui-se que ocorreu um processamento digital de imagem (PDI) porém é uma delimitação artificial, pois alguns procedimentos tem como saída um número, por exemplo, e também são considerados PDI (GONZALEZ; WOODS, 2011).

A reunião de técnicas que envolvem captura, representação ou transformação de imagem, com o auxilio do computador, constitui-se o processamento digital de imagem. A partir da aplicação dessas técnicas é possível extrair informações das imagens, melhorar a qualidade visual, com o objetivo de facilitar a percepção humana ou interpretação de um sistema inteligente (PEDRINI; SCHWARTZ, 2008).

Uma outra maneira para definir PDI, é segundo Gonzalez e Woods (2011), dividir em três níveis, processos de níveis pequeno, médio e alto. Onde o processo de menor nível pode ser

definido da maneira que foi citado no parágrafo anterior, envolvendo procedimentos primários. Já o PDI de nível médio tem como característica que a entrada do procedimento, geralmente, seja uma imagem e a saída atributos que foram retirados da mesma, por exemplo bordas ou contornos. Por fim, o processo de grau mais elevado implica em atribuir um significado a um conjunto de objetos reconhecidos, realizando funções cognitivas, que geralmente estão ligadas à visão, fazendo com que esse nível fique no limite do que é processamento digital de imagem. Logo é possível concluir que PDI abrange métodos onde as entradas e saídas podem ser imagens e também processos que extraem informações da imagem e ainda pode reconhecer objetos de maneira individual (GONZALEZ; WOODS, 2011).

Todo o processo de processamento digital de imagens envolvem várias etapas, constituindo um sistema completo, que inicia desde do momento da captura da imagem. Essa etapa tem a função de adquirir a figura e converter em uma representação adequada para o próximo passo, o pré-processamento, que envolve técnica que melhoram a qualidade da imagem, por exemplo correção de contraste ou brilho. A fase subsequente é definida como segmentação, na qual é realizada a remoção e identificação de áreas na imagem, tendo como exemplo detecção de bordas. Seguindo no sistema tem-se a etapa de representação que serve para armazenar e manipular objetos extraídos da imagem, e descrição que pretende retirar informações ou características, para utilizar na separação de classes dos objetos. O último processo é o de reconhecimento, o qual atribui um identificador aos objetos da imagem, e interpretação, que dá um significado ao conjuntos de objetos reconhecidos (PEDRINI; SCHWARTZ, 2008).

O interesse do processamento de imagens é trabalhar em funções 2D de intensidade ou cores, nas quais existem no espaço real, porém, trabalhar com essas funções no espaço real, muitas vezes pode ser muito complexo. Uma forma de facilitar o processamento é atuar no espaço de frequência e a transformada de Fourier gera um modelo equivalente da função no espaço de frequência, simplificando as soluções de problemas que são mais complexas no espaço real (SOLOMON, 2013).

2.2.1 Transformada de Fourier

Antes de abordar a Transformada de Fourier, é interessante citar uma outra colaboração de Fourier, na qual ele afirma que qualquer função periódica pode ser representada por uma

soma de senos e/ou cossenos de diferentes frequências, sendo a complexidade da função irrelevante, basta ela ser periódica e satisfazer pequenas condições matemáticas, essa soma é denominada de série de Fourier (GONZALEZ; WOODS, 2011).

A transformada e a série de Fourier, apesar de matematicamente serem diferentes, tem o mesmo objetivo, de forma conceitual fazem a mesma coisa. A diferença basicamente consiste em que a série de Fourier decompõe sinais periódicos em funções harmônicas de frequências discretas, e a transformada de Fourier decompõe sinais não periódicos, em funções harmônicas de frequências que variam continuamente (SOLOMON, 2013). Além disso, ambas tem a característica que ao aplicar o processo inverso, o sinal onde foi aplicado os métodos Fourier pode ser totalmente recuperado sem perda de informação (GONZALEZ; WOODS, 2011).

No processamento de imagem, o sistema de captura e as operações de filtragem, dependendo do domínio podem interessar de formas diferentes. No âmbito espacial interessa como ambos impactam nos *pixels* individuais na imagem, no domínio de frequência, o que importa é como eles afetam as componentes harmônicas que formam a imagem (SOLOMON, 2013).

A revolução na área de processamento de sinais aconteceu a partir do desenvolvimento do algoritmo da transformada rápida de Fourier (FFT), permitindo o processamento prático de um série de sinais de grande importância (GONZALEZ; WOODS, 2011). A partir da implementação da FFT, as transformações de funções mais complicadas podem ser calculadas de forma precisa e rápida em um computador (SOLOMON, 2013).

2.2.2 Função de espalhamento de ponto

A função de dispersão pontual (Point Spread Function-PSF), explica a base da formação de uma imagem, indica como uma fonte pontual de luz produz uma imagem espelhada na dimensão espacial (ACHARYA, 2007).

Ao analisar um ponto em um objeto na imagem, a luz que reflete nele e chega até o conjunto óptico, gerando a imagem resultante, é o efeito da convolução da função de dispersão pontual desse conjunto, com a figura original. Essa convolução pode ser definida pela Equação 1. Portanto, o quadro de um ponto do objeto demonstra o PSF do sistema visual (MOSQUERA, 2015).

$$I_{res}(x,y) = I_{id}(x,y) \otimes P(x,y) \tag{1}$$

Na qual I_{res}, é o resultado entre a convolução, representado por $\otimes$, a imagem de entrada I_{id} e a função de dispersão pontual P(x,y), na coordenada (x,y) (ACHARYA, 2007).

Uma suposição comum, utilizada em imagens que são borradas por movimento, utiliza-se uma PSF espacialmente invariável, ou seja os movimentos dos objetos são ignorados, e a função descreve um movimento de forma global (MOSQUERA, 2015).

Segundo Gonzalez e Woods (2011) a função de dispersão pontual se origina pelo fato de que os sistemas ópticos espalham um ponto de luz em determinada extensão, e a quantidade de borramento depende da qualidade dos componentes ópticos.

2.2.3 Degradações de Imagens

A degradação de uma imagem, são problemas que podem ocorrer em imagens, como desfocagem, borrão por movimento, entre outros fatores, pode ser modelada segundo Almeida e Almeida (2010), Amudha, Pradeepa e Sudhakar (2012) e Mosquera (2015), a partir da Equação 2:

$$I_{res}(x,y) = I_{id}(x,y) \otimes P(x,y) + N(x,y) \tag{2}$$

Conforme visto anteriormente na Equação 1, a I_{res}, é a imagem resultante da convolução entre a imagem de entrada I_{id} e a função de dispersão pontual $P(x,y)$, contudo na Equação 2, é adicionado um ruído $N(x,y)$ onde a primeira parte da modelagem foi vista na Equação 1.

Para tentar solucionar a degradação na imagem, é necessário destacar o modelo e a fonte da degradação, a partir desses fatores é possível reverter o processo, para presumir o que ocorreu, geralmente, utiliza-se modelos analíticos, estatísticos ou informações por dedução, em conjunto com conhecimento do conjunto de sensores que gerou a imagem (ANDREWS, 1974).

A causa da degradação de uma figura pode ocorrer por diversos fatores como ruído, borramento por movimento, desfocagem proveniente de vários efeitos específicos, desvio de desempenho de um sistema óptico, entre outros fatores, fazendo com que amplifique as áreas que necessitam do procedimento de restauração de imagens, dentre elas é possível citar astronomia, imagens médicas, microscopia e outras áreas (MOSQUERA, 2015).

2.2.3.1 Desfocagem

Conforme citado no Seção 2.2, o padrão de degradação é bastante importante no processo de solução da degradação na imagem. Existem diversos modelos tratando de borramento, segundo Amudha, Pradeepa e Sudhakar (2012). Dentre eles é possível destacar o desfoque atmosférico, que pode acontecer nas imagens de sensoriamento remoto de satélites (KOPEIKA, 1998), desfoque uniforme, por movimento quando um objeto ou câmera se move durante a aquisição da cena, que pode ser observado na Figura 2 e o desfoque Gaussiano, que é possível notar em fotos tiradas em um nevoeiro, imagens subaquáticas, capturas astronômicas terrestres na atmosfera (FLUSSER et al., 2016), entre outros. Com a dificuldade de se trabalhar com outros tipos de borrões, o desfoque Gaussiano é bastante utilizado para simula-los (FLUSSER et al., 2016).

(a) Imagem original (b) Imagem borrada

Figura 2 – Exemplo de desfoque por movimento. (a)Imagem original. (b)Imagem Borrada. Fonte: Adaptado de Mosquera (2015)

2.2.4 Restauração de Imagem

Na tentativa de reverter o processo de degradação de uma imagem, existem as técnicas de restauração (ANDREWS, 1974). Para um bom resultado, essas técnicas dependem de alguns fatores, como o quanto degenerada foi a figura, se o modelo formulado para recuperação corresponde a degradação ocorrida, e também se o algoritmo de restauração foi desenvolvido da melhor forma possível (ANDREWS, 1974). A recuperação de uma imagem baseia-se em

um conhecimento *a priori* da degradação ocorrida, aplicando o processo inverso (GONZALEZ; WOODS, 2011).

Na prática, muitas vezes não é possível conhecer a PSF do sistema óptico que foi utilizado no modelo de degradação, compreendido na Equação 2. Nesse caso é necessário realizar uma estimativa, usando algoritmos de deconvolução cega, que tem como objetivo restaurar a imagem sem saber a causa da degradação. Eles possuem resultados melhores do que os algoritmos não-cegos, que utilizam modelos teóricos da PSF e dependem de um sistema ideal, o qual não acontece na realidade (PINTO et al., 2015).

A Figura 3 exemplifica o procedimento de recuperação da imagem, no qual H, é a função de degradação e juntamente com um ruído aditivo $\eta(x,y)$, formam a degradação, e a função $g(x,y)$, o modelo formulado para reverter o processo, a partir de algum conhecimento sobre o ruído aditivo e sobre a função de degradação, gerando uma imagem resultante $\hat{f}(x,y)$ (GONZALEZ; WOODS, 2011).

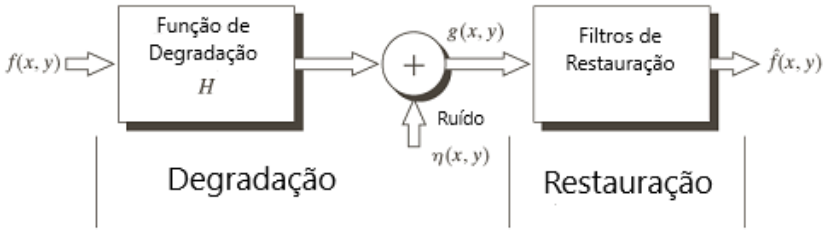


Figura 3 – Modelo de processo degradação/restauração de imagens.
Fonte: Adaptado de Gonzalez e Woods (2011)

2.2.4.1 Deconvolução não-cega

Na deconvolução não-cega, existem muitos algoritmos que tem como propósito solucionar o problema de degradação, dentre os mais famosos estão o filtro de Wiener (FW) e o algoritmo de Richardson-Lucy (RL) (MOSQUERA, 2015). Porém esse tipo de abordagem tem alguns problemas, pois mesmo que a PSF seja conhecida, usar o processo inverso não é recomendado, porque os resultados são afetados por ruído amplificado e artefatos, porém existem métodos para regulagem dos processos (TAO et al., 2013). Na Figura 4, é possível observar esses problemas na imagem resultante. Os artefatos de toque se parecem com

ondulações em torno das bordas fortes na imagem restaurada. A PSF normalmente é limitada por banda, fazendo com que seus valores de frequência resultem em zero ou próximos de zero. Logo, a inversão da PSF com a imagem borrada pode causar amplificação de sinais em componentes que possuem alta frequência, resultando nos artefatos de toque próximos às bordas e a amplificação do ruído, fornecendo resultados aborrecedores (LEE; HO, 2011).

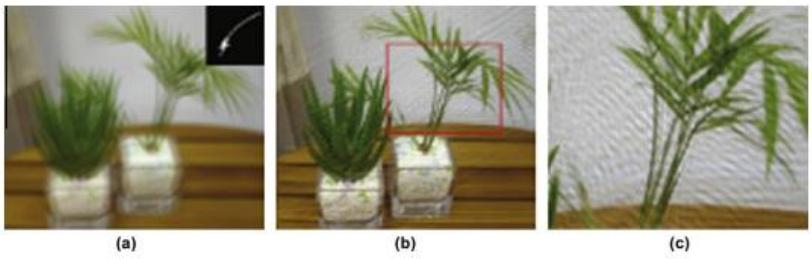


Figura 4 – Artefatos de toque na deconvolução de imagem. (a)Imagem desfocada e a PSF estimada. (b)Resultado da deconvolução. (c)Pedaço ampliado.

Fonte: Lee e Ho (2011)

2.2.4.2 Deconvolução cega

O processo da deconvolução cega tem como objetivo restaurar a imagem sem saber a causa da degradação, que o torna um problema não totalmente compreendido, sendo difícil de ser solucionado (CARASSO, 2001). A maioria das soluções utilizam métodos iterativos, porém podem desenvolver pontos de estagnação ou divergir completamente, e mesmo que o método seja bem desenvolvido, o número de iterações pode ser uma complicação (CARASSO, 2001).

Mesmo com muitos fatores que dificultam o processo de restauração cega de imagem, como o custo computacional, ou ainda, a impossibilidade de obter qualquer informação sobre a fotografia, uma das razões que motivam aplicações, é por ser uma alternativa viável para melhorar a imagem sem precisar de métodos de calibração complicados (KUNDUR; HATZINAKOS, 1996).

A deconvolução cega tem duas abordagens principais, a primeira onde a PSF é identificada separadamente da imagem original, para ser utilizada usando métodos de deconvolução não-cega, esse procedimento tem como vantagem o uso de técnicas mais simples (KUNDUR; HATZINAKOS, 1996). São métodos de identificação de desfoque *a priori*, tendo

como objetivo identificar os parâmetros de degradação mais prováveis a partir da observação. É possível também coletar imagens de fontes pontuais para obter uma estimativa da PSF. Essa abordagem pode ser utilizada em sistemas microscópios, de ultrassom, sensoriamento remoto ou telescópio óptico (CAMPISI; EGIAZARIAN, 2017).

A segunda abordagem une o procedimento de identificação e o algoritmo de restauração, onde se estima a PSF e a imagem original de forma simultânea. Logo os algoritmos dessa abordagens são mais complexos (KUNDUR; HATZINAKOS, 1996). É onde a maioria dos métodos se enquadram. O conhecimento anterior sobre a imagem e a degradação são modelados matematicamente, os parâmetros que descrevem os modelos precisam ser estimados a partir dos dados disponíveis. Normalmente é realizado antes da imagem e identificação da degradação (CAMPISI; EGIAZARIAN, 2017).

2.2.4.3 Estimando os parâmetros de borramento

Quando o *kernel* de borramento não é conhecido, é necessário utilizar abordagens para descobrir o ângulo do movimento do borramento e seu comprimento e a partir da descoberta utilizar algoritmos de restauração de imagem. Segundo Soe e Zhang (2012) existem dois tipos de algoritmos para estimar o ângulo de borramento, o primeiro tenta utilizar várias imagens para tentar estimar os parâmetros, e o segundo tipo tenta decifrar a partir de uma única imagem, este considera que o movimento de cada *pixel* é o mesmo.

Um método para estimar os parâmetros de borramento de uma imagem é analisando a partir da Transformada de Fourier da imagem. Como é possível observar na Figura 5, o espectro da imagem borrada contém linhas paralelas que indicam o movimento de borramento (KHARE; NAGWANSHI, 2011). Para calcular a orientação dessas linhas paralelas, é utilizado a transformada de Radon, segundo Khare e Nagwanshi (2011), um dos métodos mais eficazes para calcular a orientação de linhas em imagens. No âmbito de processamento de imagens, a transformada de Radon é o cálculo da projeção da imagem por todos os ângulos que foram fornecidos, esse processo resulta na soma da intensidade dos *pixeis* em cada direção (HOILUND, 2007). Portanto esse procedimento resulta uma imagem $I(\rho, \theta)$, no qual ρ é a distância e θ é o angulo (HOILUND, 2007). Para encontrar a direção do borramento é necessário encontrar a solução máxima da transformada de Radon em cada ângulo, considerando somente de 0 a 179^{o}, o valor máximo desse processo é o ângulo de movimento do borramento (KHARE; NAGWANSHI, 2011).

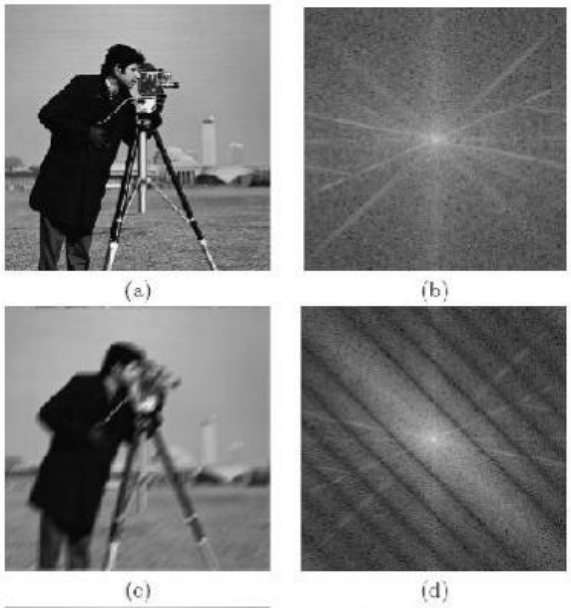


Figura 5 – (a) Imagem Original. (b) Espectro da Imagem Original. (c) Imagem borrada com um angulo de 40º. (d) Espectro da Imagem Borrada.

Fonte: Adaptado de Khare e Nagwanshi (2011)

Outra abordagem para encontrar a orientação do borramento, é utilizando o algoritmo proposto por Liang e Liang (2016), nele a análise é baseada na transformada de Fourier do espectro da imagem borrada, conforme pode ser visto na Figura 6.

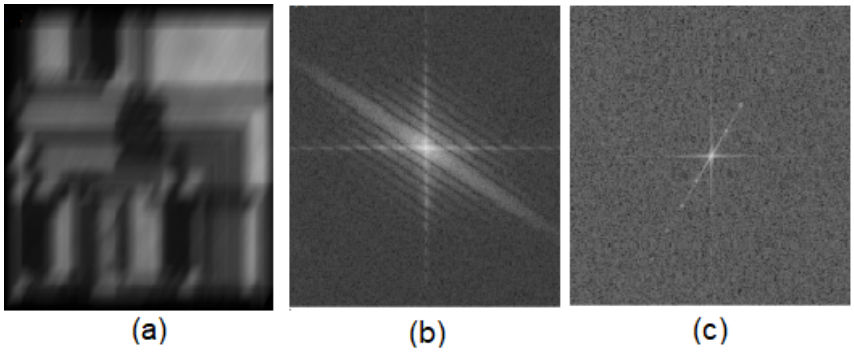


Figura 6 – (a) Imagem Borrada. (b) Espectro da Imagem Borrada, com a origem no centro. (c) Espectro da primeira transformada de Fourier.

Fonte: Adaptado de Liang e Liang (2016)

A linha brilhante diagonal que aparece na Figura 6 (c), mostra o sentido do borramento da imagem, a partir disso, é necessário encontrar o ângulo entre esse linha e a linha horizontal do centro. Para encontrar o ângulo, é necessário calcular, considerando o ponto central da imagem como o ponto final da linha brilhante e a origem do plano, o valor de cinza da linha em comparação a cada ângulo, em um intervalo de 0 a 180º, a linha com o maior valor máximo

de cinza na imagem indica o ângulo do borramento, para realizar esse procedimento de forma rápida e eficaz, pode ser utilizado o Algoritmo de Bresenham (LIANG; LIANG, 2016). O Algoritmo de Bresenham, permite especular a coordenada do próximo ponto a partir de um ponto conhecido, ele determina os *pixeis* em cada linha reta, tem como vantagem de gerar rapidamente uma linha reta a partir da imagem digital (LIANG; LIANG, 2016). Para estimar o comprimento do borramento, Khare e Nagwanshi (2011), propuseram um algoritmo que estima esse parâmetro. Analisando o espectro de Fourier, é possível analisar que quanto maior o comprimento do borramento, mais próximas as linhas escuras ficam umas das outras, é possível observar essa característica na Figura 7 (KHARE; NAGWANSHI, 2011).

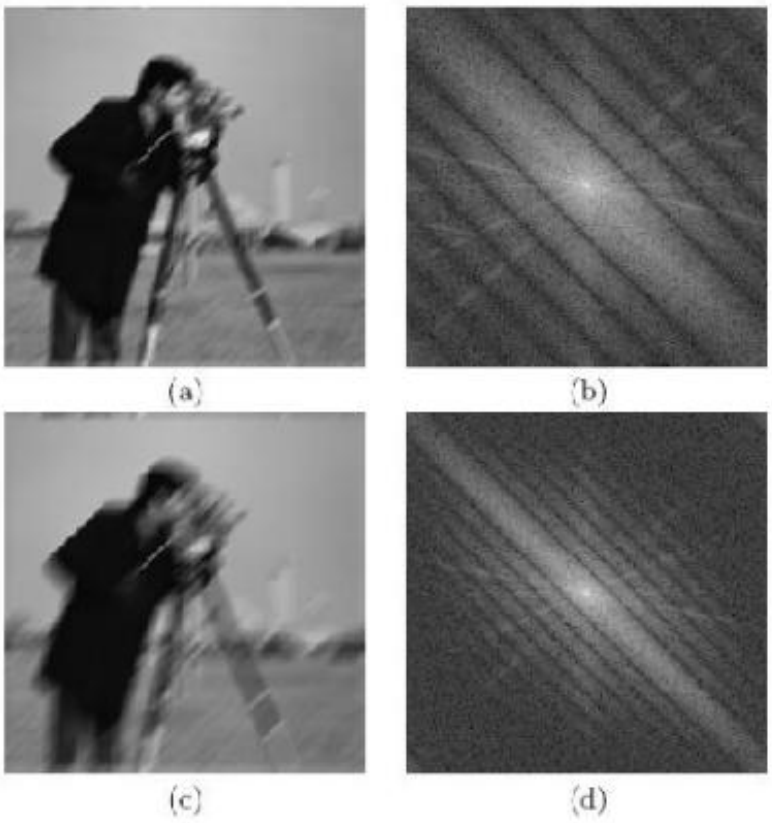


Figura 7 – (a) Imagem Borrada com borramento de comprimento 10 *pixeis*. (b) Espectro da Imagem Borrada (a). (c) Imagem Borrada com borramento de comprimento 20 *pixeis* .

Fonte: Khare e Nagwanshi (2011)

Conhecendo o ângulo de borramento, é necessário aplicar a transformada de Radon, fixada no ângulo conhecido, após esse procedimento, calcula-se a distância entre dois vizinhos máximos (ou mínimos vizinhos), para encontrar o valor da distância (KHARE; NAGWANSHI, 2011). Outro método para estimar o tamanho do *kernel* de borramento, é baseado no domínio cepstral. Segundo Cakır (2010) para calcular o domínio cepstral de uma imagem 2D, é

necessário seguir os seguintes passos:

- Passo 1: Calcular a transformada de Fourier da imagem;
- Passo 2: Calcular os valores absolutos do espectro resultante do passo anterior;
- Passo 3: Calcular o logaritmo natural do passo anterior;
- Passo 4: Calcular a transformada inversa de Fourier do resultado anterior.

Matematicamente pode ser definido a partir da equação: (TIWARI et al., 2013)

$$C\{f(x,y)\} = F^{-1}(\log|F(f(x,y))|) \tag{3}$$

Na qual $C\{f(x,y)\}$ é o dominio cepstral da imagem, $F(f(x,y))$ é a transformada de Fourier da imagem e F^{-1} é a transformada inversa de Fourier. O *kernel* de desfoque de movimento uniforme possui uma característica que, no domínio de frequência possui padrões periódicos que cruzam o zero da função (TIWARI et al., 2013).Esses padrões, no domínio cepstral geram grandes picos negativos, a uma distância d da origem (TIWARI et al., 2013; CANNON, 1976), a distância d do primeiro pico negativo até a origem é o tamanho do *kernel* de borramento. Utilizando esse conceito é possível estimar o tamanho do *kernel* a partir do seguinte algoritmo (TIWARI et al., 2013):

- Passo 1: Converter a imagem para escala cinza gerando uma imagem f(x,y);
- Passo 2: Calcular a transformada de Fourier resultante do passo anterior;
- Passo 3: Calcular os valores absolutos do espectro resultante do passo anterior;
- Passo 4: Calcular o logaritmo natural do passo anterior;
- Passo 5: Calcular a transformada inversa de Fourier do resultado anterior.
- Passo 6: Calcular o logaritmo natural do passo anterior;
- Passo 7: Rotacionar na direção inversa, o espectro, utilizando o ângulo estimado.
- Passo 8: Converter a matriz 2D em 1D, calculando as médias das colunas;
- Passo 9: Encontrar a distancia de colunas entre o primeiro pico negativo e a origem.

  A partir desse algoritmo é possível encontrar o tamanho do *kernel* de desfoque por movimento.

2.2.4.4 Filtro de Wiener

Nesse método, tanto a imagem e o ruído são considerados como varáveis aleatórias, e tem como objetivo encontrar uma estimativa da imagem sem a degradação, de forma que o

erro quadrático médio seja o mínimo possível. Essa medida de erro é dada pela Equação 4 (GONZALEZ; WOODS, 2011).

$$e^2 = E\{(f - \hat{f})^2\} \tag{4}$$

Na qual $E\{.\}$ é o valor esperado do argumento, e ainda entende-se que não haja uma relação entre o ruído e a imagem, que um dos dois tenha média zero e ainda que os níveis de intensidade da estimativa sejam uma função linear dos níveis da imagem degradada. Partindo dessas premissas, é possível expressar o mínimo da função de erro, visto na Equação 4, no domínio da frequência por meio da Equação 5 (GONZALEZ; WOODS, 2011).

$$\begin{aligned} \hat{F}(x,y) &= \Big[\frac{H^*(x,y)S_f(x,y)}{S_f(x,y)|H(x,y)|^2 + S_f(x,y)}\Big] G(x,y) \\ &= \Big[\frac{H^*(x,y)}{|H(x,y)|^2 \frac{S_\eta(x,y)}{S_f(x,y}}\Big] G(x,y) \\ &= \Big[\frac{1}{H(x,y)} \frac{|H(x,y)|^2}{|H(x,y)|^2 + \frac{S_\eta(x,y)}{S_f(x,y}}\Big] G(x,y) \end{aligned} \tag{5}$$

Na qual $H(x,y)$ é a função de degradação, $H^*(x,y)$ é o conjugado complexo de $H(x,y)$, $|H(x,y)|^2$ equivale a $H^*(x,y)H(x,y)$, $S_\eta(x,y)$ é o espectro de potência do ruído, $S_f(x,y)$ é o espectro de potência da imagem não degradada, $G(x,y)$ é a transformada da imagem degradada e $\hat{F}(x,y)$ é a figura restaurada no domínio espacial dada pela transformada inversa de Fourier da estimativa no domínio de frequência (GONZALEZ; WOODS, 2011). Na Equação 5 caso não seja possível saber alguma informação sobre as propriedades estatísticas da imagem degradada, pode-se substituir $\frac{S_\eta(x,y)}{S_f(x,y}$ por uma constante τ e experimentar valores para ela (PETROU; PETROU, 2010). Logo a Equação 5 pode ser simplificada conforme a Equação 6.

$$\hat{F}(x,y) = \Big[\frac{1}{H(x,y)} \frac{|H(x,y)|^2}{|H(x,y)|^2 + \tau}\Big] G(x,y) \tag{6}$$

Em comparação com outro método como o filtro inverso, o de Wiener não tem o problema de ser afetado pelo ruído, mesmo se a função de degradação for zero. Caso o ruído seja zero, esse método se reduz ao filtro inverso (ACHARYA, 2007).

Segundo Petrou e Petrou (2010) um algoritmo a ser utilizado para aplicar o filtro de Wiener na prática, resumidamente:

- Passo 1: Calcular transformada de Fourier da PSF do modelo de degradação;
- Passo 2: Fazer a transformada de Fourier da imagem com degradação;
- Passo 3: Selecionar um valor para a constante τ e multiplique com o espectro da imagem

degradada ponto por ponto de acordo com:

$$\hat{F}(x,y) = \frac{H^*(x,y)}{|H(x,y)|^2 + \tau} \tag{7}$$

- Passo 4: Calcular a transformada inversa de Fourier para recuperar a imagem desfocada.

A partir desse algoritmo é possível implementar o filtro de Wiener.

2.2.4.5 Algoritmo de Richardson-Lucy

O algoritmo de Richardson-Lucy calcula a imagem sem degradação a partir da imagem de entrada com o desfoque (MOSQUERA, 2015). Usando princípios de probabilidade, o algoritmo é um método iterativo que foi desenvolvido com o objetivo de encontrar uma solução de máxima verossimilhança usando o conhecimento da PSF da degradação e o modelo de ruído de Poisson (SOLOMON, 2013). Se tratando de restauração de imagem sua utilização é bastante popular, devido a sua capacidade de reconstruir imagens com uma boa qualidade mesmo com altos níveis de ruído. É amplamente utilizado nos campos da astronomia e de imagens médicas (FISH et al., 1995).

Para aplicação do método na restauração de imagem, o algoritmo se baseia no teorema de Bayes, que estabelece uma relação entre as probabilidades, podendo ser a PSF, as imagens de entrada ou a degradada (MOSQUERA, 2015). Segundo Fish et al. (1995), o teorema de Bayes pode ser dado pela Equação 8.

$$P(x|y) = \frac{P(y|x)P(x)}{\int P(y|x)P(x)dx} \tag{8}$$

no qual $P(y|x)$ é a probabilidade condicional do evento y em relação a x , $P(x)$ é a probabilidade do evento x e $P(x|y)$ é a probabilidade condicional do evento x em relação a y, trazendo para o âmbito de restauração de imagem, $P(x)$ pode ser definido como a distribuição de probabilidade da imagem de saída, $P(x|y)$ identificado como distribuição da probabilidade da PSF e $P(y)$ é a distribuição da imagem degradada (FISH et al., 1995; MOSQUERA, 2015). De acordo com Fish et al. (1995), a função iteração do algoritmo Richardson-Lucy pode ser definida segundo a Equação 9.

$$f_{i+1}(x) = \left\{ \left[\frac{c(x)}{f_i(x) \otimes g(x)} \right] \otimes g(-x) \right\} f_i(x) \tag{9}$$

Em que, $\otimes$ representa a operação de convolução, a PSF conhecida é representada por $g(x)$ e

$c(x)$ é a imagem degradada, nas primeiras iterações a estimativa da imagem real não fica com uma boa qualidade, os detalhes vão se ajustando a cada iteração (FISH et al., 1995).

Uma outra abordagem para função de iteração do algoritmoRichardson-Lucy, partindo do principio que a PSF seja conhecida e o ruído seja orientado apela função densidade de Poisson, segundo Solomon (2013), pode ser dada pela Equação 10.

$$f_{i+1}(x,y) = f_i(x,y) + [g(x,y) - f_i(x,y) \otimes h(x,y)] \tag{10}$$

Na qual, $f_i(x,y)$ é a estimativa da distribuição da imagem de entrada, $g(x,y)$ é a estimativa da distribuição da imagem de saída e $\otimes$ é a operação de convolução entre $f_i(x,y)$ e a função PSF $h(x,y)$ e para iniciar o método usa-se $f_0(x,y) = g(x,y)$ (SOLOMON, 2013).

Um outra solução do algoritmo deRichardson-Lucy, para um ruído gaussiano, segundo Souza (2005), pode ser dada na Equação 11.

$$f_{i+1}(x,y) = f_i(x,y) + H * g(x,y) - H \otimes f_i(x,y) \tag{11}$$

Em que, $*$ é operador de correlação e $\otimes$ é a operação de convolução, H é a PSF da degradação e $g(x,y)$ e $f_i(x,y)$ é a imagem degradada e a estimativa da imagem real respectivamente (SOUZA, 2005). Segundo Mosquera (2015) o pseudcódigo de uma iteração da restauração de Richardson-Lucy pode ser visualizada segundo o Algoritmo 1.

Algoritmo 1: Pseudocódigo para uma iteração do Algoritmo de Richardson-Lucy

Entrada: Imagem observada(Observation) em escala de cinza de tamanho $m \times n$
Função de espalhamento de ponto (psf)
Imagem restaurada na iteração n (I^n), para n = 1 então $I^n \leftarrow 1$;
Saída: Imagem restaurada I^{n+1} na iteração n+1

```
imagem_observada ← I^n ⊗ psf
if imagem_observada = 0 then
    imagem_fator ← 255;
else
    imagem_fator ← Observation ÷ imagem_observada;
imagem_corrente ← imagem_fator ⊗ psf;
I^{n+1} ← I^n * imagem_corrente;
```

A partir do Algoritmo 1 é possível desenvolver o algoritmo de Richardson-Lucy.

2.2.4.6 Restauração de máxima verossimilhança

O problema da restauração cega é que não se conhece a função de degradação, nem o ruído aditivo na imagem. A partir da restauração de máxima verossimilhança, uma estimativa com probabilidade máxima desses parâmetros é obtida, isso quer dizer que o conjunto de parâmetros encontrados, ϕ, faz com que a função densidade de probabilidade da imagem degradada seja maximizada e essa função pode ser definida com a Equação 12 (KUNDUR; HATZINAKOS, 1996).

$$p(g|\phi) = \frac{1}{\sqrt{|2\pi(H\Lambda_F H^H + \Lambda_N)|}} exp\{-\frac{1}{2} g^H (H\Lambda_F H^H + \Lambda_N)^{-1} g\} \tag{12}$$

Na qual H é a matriz da PSF, Λ_F é a matriz de covariança da imagem real sem degradação, $(.)^H$ é a tranposição conjugada de uma matriz, Λ_N é a matriz de covariança do ruído aditivo e g é a imagem degradada (KUNDUR; HATZINAKOS, 1996). Segundo Lagendijk, Tekalp e Biemond (1990) a função de máxima verossimilhança, pode ser estimada pela Equação 13.

$$\hat{\phi_{m1}} = arg\{\max_{\phi \in \theta} \ell*(\phi)\} = arg\{\max_{\phi \in \theta} \log p(g|\phi)\} \tag{13}$$

Em que, $\ell*$ é a função de probabilidade dos parâmetros desconhecidos, representado por ϕ, $p(g|\phi)$ denota a função de densidade da imagem observado, definida na Equação 12, θ especifica o intervalo dos parâmetros ϕ (LAGENDIJK; TEKALP; BIEMOND, 1990).

Devido ao grande número de incógnitas envolvidas na função de verossimilhança, essa solução pode se tornar sem sentido e intratável. Além disso, a estimativa da PSF pode não ser única por causa da utilização de estatísticas de segunda ordem no processo. Logo não se tem informação sobre a fase de degradação. Técnicas de identificação recursiva tem com requisito mínimo ter alguma informação sobre a fase de desfoque. Para solucionar esse problema informações adicionais sobre os parâmetros desconhecidos são adicionados ao processo (KATSAGGELOS; LAY, 1991). Além disso, a otimização do processo é difícil, devido ao alto grau de não-linearidade, porém para estimar os parâmetros desconhecidos da função, utiliza-se o método de maximização de expectativa (KATSAGGELOS; LAY, 1991; KUNDUR; HATZINAKOS, 1996).

Para realizar o passos do método de maximização de expectativa, é necessário definir adequadamente os dados completos e incompletos (KATSAGGELOS; LAY, 1991). O processo é divido em dois passos, o passo E, no qual são calculados a expectativa condicional do log da função densidade de probabilidade dos dados completos condicionada pelos dados

incompletos e a estimativa atual dos parâmetros relevantes, e no passo M essa expectativa é maximizada, os passos E e M podem ser definidos segundo as Equações 14 e 15 respectivamente (KATSAGGELOS; LAY, 1991).

$$Q(\phi, \phi^{(p)}) = E[\log f_z(z;\phi)|y;\phi^{(p)}] \tag{14}$$

$$\phi^{(p+1)} = arg\{\max_{[\phi]} Q(\phi, \phi^{(p)})\} \tag{15}$$

Em que, $Q(\phi, \phi^{(p)})$ é a função de expectativa condicional, $f_z(z;\phi)$ é a função de densidade de probabilidade dos dados completos, y são os dados incompletos e $\phi^{(p)}$ é a estimativa de ϕ na iteração p (KATSAGGELOS; LAY, 1991).

2.3 APRENDIZADO PROFUNDO

Com advento da tecnologia, a quantidade de aplicações que utilizam o aprendizado de máquina aumentam cada vez mais. Elas variam entre filtragem de conteúdo nas redes sociais a recomendações em sites de comércio eletrônicos. E a classe de técnicas que vem sendo utilizada crescentemente é chamada de aprendizado profundo, um subconjunto da aprendizagem de máquina, que basicamente são métodos que permitem que um computador a partir de dados brutos descubra de forma automática as representações necessárias para detecção ou classificação (LECUN; BENGIO; HINTON, 2015).

Um dos principais problemas para soluções que envolvem inteligência artificial é que no mundo real existem muitos fatores variáveis que podem influenciar os dados que são observados. Por exemplo, ao analisar *pixeis* individuais em uma imagem onde o objeto tem um cor escura, e ele está em um fundo preto, as aplicações requerem que fatores desnecessário sejam descartados para simplificar a solução, porém é difícil separar esses recursos abstratos de alto nível (GOODFELLOW; BENGIO; COURVILLE, 2016). Com o aprendizado profundo é possível resolver esse problema, devido que as representações complexas podem ser expressas em várias representações mais simples de lidar, construindo conceitos difíceis a partir de mais simples (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.4 REDES NEURAIS ARTIFICIAIS

Redes Neurais Artificias (RNA) podem ser definidas, segundo Lima, Pinheiro e Santos (2016), “como modelos computacionais com capacidades de adaptar, aprender, generalizar, agrupar ou organizar dados”. Elas são baseadas nas redes neurais biológicas, ambas têm a capacidade de computação paralela, na rede artificial isso é possível graças a organização de processamento paralelo e distribuído em que são processados elementos interconectados por meio de sinais chamados de conexões (EBERMAM; KROHLING, 2018).

Na modelagem das RNAs, o componente principal é chamado de nó, neurônio ou célula. São unidades de processamento que tem como inspiração o neurônio biológico que possui um elemento bastante importante, a sinapse. É por meio dela o neurônio recebe e envia os impulsos, ela ainda tem a função de regular as quantidades de informações que são passadas para um outro componente do neurônio biológico (LIMA; PINHEIRO; SANTOS, 2016).

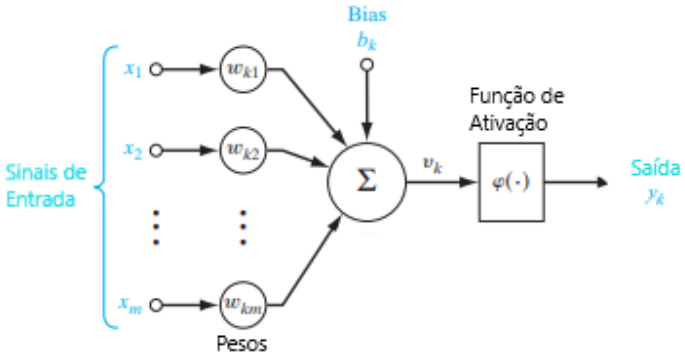


Figura 8 – Modelo não linear de um neurônio artificial

Fonte: Adaptado de Haykin (2009)

Na Figura 8, é possível observar a estrutura de um neurônio artificial, no qual o conjunto de entrada é denotado por $x_1, x_2, ..., x_m$ e seus respectivos pesos, $w_{k1}, w_{k2}, ..., w_{km}$. Na notação dos pesos, o primeiro índice faz referência ao neurônio e o segundo à entrada que ele analisa. O valor de um peso pode ser como positivo ou negativo (HAYKIN, 2009). A soma dessas entradas multiplicadas com seus respectivos pesos definem a função primitiva (FELDMAN; ROJAS, 2013), definida matematicamente pela Equação 16 (HAYKIN, 2009).

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{16}$$

A partir da função primitiva, é possível gerar uma resposta na saída do modelo de neurônio, essa informação é verificada por limiar definido por uma função chamada de ativação (LIMA; PINHEIRO; SANTOS, 2016). Além da função primitiva, é incluído um *bias*, na Figura 8 é indicado como b_k, que tem a finalidade de aumentar ou diminuir a entrada da função de ativação, dependendo se ele for positivo ou negativo, respectivamente (HAYKIN, 2009).

Os valores desses pesos e *biases* são ajustados através de algoritmos de aprendizagem, sendo uma importante característica das redes neurais artificiais, essa capacidade de aprender e ajustar esses valores auxiliam na melhora do desempenho da rede (HAYKIN, 2007). Porém esses ajustes trazem dois desafios na área de aprendizado de máquina, o sobreajuste (*overfitting*) e o sub-ajuste (*underfitting*). O sobreajuste acontece quando a diferença entre o erro de treinamento e o de teste é muito grande, ou seja o a rede obteve um bom desempenho no treino porém não teve o mesmo desempenho ou parecido nos testes, logo informalmente pode-se concluir que a rede memorizou os dados e não aprendeu. Já o sub-ajuste o erro no treinamento é muito grande, a rede não obteve um desempenho satisfatório no treinamento (GOODFELLOW; BENGIO; COURVILLE, 2016). Um método atual para regular o problema de sobreajuste, é utilizando a técnica de *dropout*. É um algoritmo que tem como conceito principal descartar neurônios e suas conexões aleatoriamente, garantindo um melhor desempenho no treinamento da rede (SRIVASTAVA et al., 2014).

2.4.1 Função de Ativação

A função de ativação possibilita que uma pequena alteração nos pesos e no *bias* resulte em uma pequena alteração na saída do neurônio (Data Science Academy, 2019). A função de ativação realiza o processo de transformação não-linear na informação de entrada, fazendo com que a rede neural artificial aprenda e execute tarefas mais complexas (Data Science Academy, 2019). Segundo Datta (2020), além de ser não lineares as funções de ativação devem ser diferenciáveis, limitadas, centradas em zero e ter um baixo custo computacional. Existem diversos tipos de função de ativação, dentre eles é possível citar, Função de Etapa Binária (também chamada de função Limiar, Degrau ou Passo), Função Sigmoide, Função ReLU e a Função Leaky ReLu.

2.4.1.1 Função de Etapa Binária

A função mais simples e de bom entendimento de como funciona uma função de ativação baseado em um limiar, é a função de Etapa Binária. Nela, se o valor da função estiver acima ou abaixo do limite, o neurônio é ativado ou desativado respectivamente (Data Science Academy, 2019).

$$\begin{cases} f(x) = 1, x >= 0 \\ f(x) = 0, x < 0 \end{cases} \tag{17}$$

A Equação 17, mostra matematicamente o funcionamento desse tipo de função, nesse caso o resultado da função é 1 caso o valor do limiar x for maior ou igual a zero e o resultado da função é zero caso o valor do limiar for menor que zero (Data Science Academy, 2019).

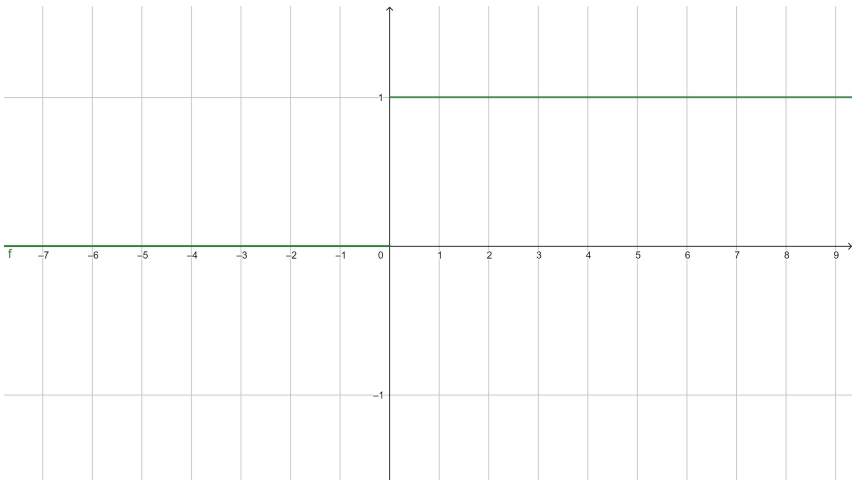


Figura 9 – Representação da Função Etapa binária graficamente
Fonte: Adpatado de Haykin (2007)

Graficamente a função é representada pela Figura 9.

2.4.1.2 Função Sigmoide

A função Sigmoide tem como característica limitar que o resultado da função varie entre 0 e 1, caracterizando assim um gráfico em formato de “S”, como é possível ver na Figura

10 (Data Science Academy, 2019). Além de ter a característica limitadora também tem a propriedade de ser diferenciável e é uma das funções mais utilizadas em redes neurais artificiais (DATTA, 2020).

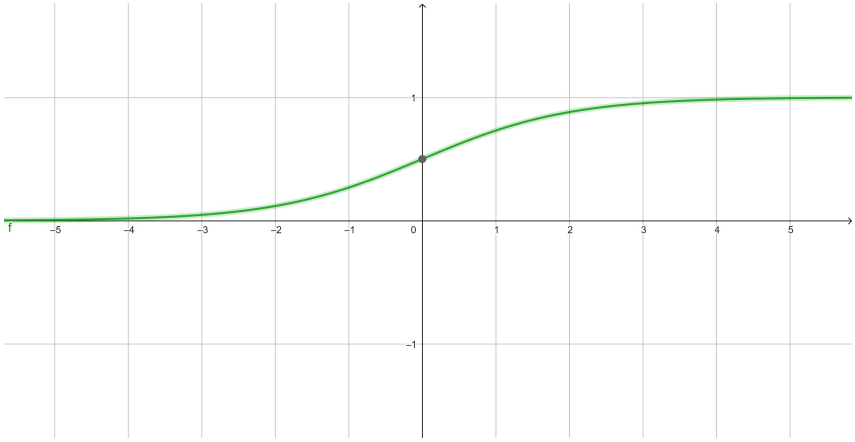


Figura 10 – Representação da Função Sigmoide graficamente
Fonte: Adpatado de Goodfellow, Bengio e Courville (2016)

A função pode ser definida matematicamente segundo a Equação 18.

$$f(x) = \frac{1}{(1+e^{-x})} \tag{18}$$

2.4.1.3 Função ReLu

A função ReLu é bastante eficiente computacionalmente, pois caso a entrada for negativa a função a converte em zero e retorna a própria entrada caso positiva. Isso faz com que todos os neurônios não sejam ativados ao mesmo tempo, tornando-a mais eficiente, tornando a função ReLu uma das mais utilizadas em redes neurais atualmente (Data Science Academy, 2019). Em comparação com a função Sigmoide, a função ReLu pode ser seis vezes mais rápida, ela é geralmente utilizada em camadas ocultas juntamente com outras funções de ativação (DATTA, 2020).

Ela pode ser definida a partir da Equação 19.

$$f(x) = max(0,x) \tag{19}$$

Possui um gráfico mais simples, representado pela Figua 11.

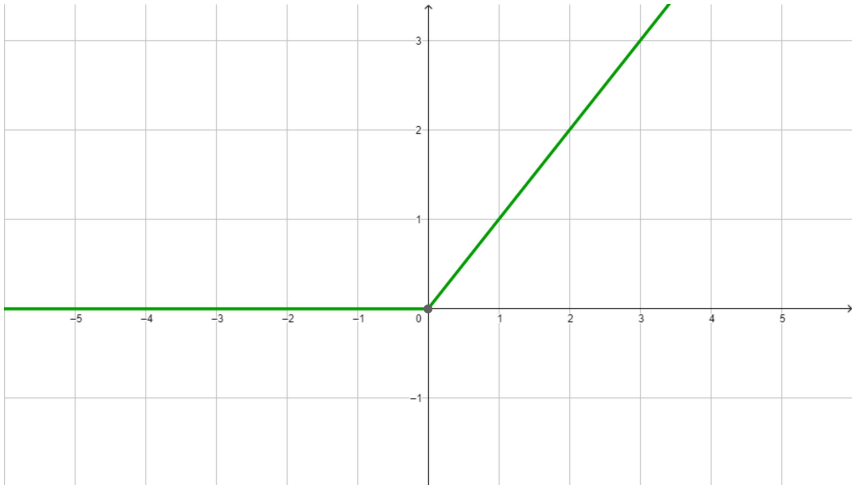


Figura 11 – Representação da Função ReLu graficamente
Fonte: Adpatado de Goodfellow, Bengio e Courville (2016)

2.4.1.4 Função Leaky ReLu

A função Leaky ReLu tem seu funcionamento parecido com a função ReLu, sendo considerada uma versão melhorada, em vez de converter a entrada, caso seja negativa, em 0, desativando os neurônios, a função Leaky ReLu define para entrada menor que zero, um pequeno componente linear da entrada, podendo ser definido como na Equação 19 (Data Science Academy, 2019).

Figura 12 – Representação da Função Leaky ReLu graficamente
Fonte: Autoria Própria

$$\begin{cases} f(x) = x, x >= 0 \\ f(x) = ax, x < 0 \end{cases} \tag{20}$$

O custo computacional desta função é muito baixo, é uma função contínua e como a função ReLu é não limitada, além de ser centrada em zero (DATTA, 2020). Na Equação 20, a é uma constante de valor pequeno, por exemplo $0,01$, evitando que o valor de saída tenha resultado zero (Data Science Academy, 2019) contribuindo para evitar o problema de dissipação do gradiente. A Figura 12, mostra graficamente o que acontece caso a entrada seja menor que zero, diferente do que mostra na Figura 11.

2.4.2 Treinamento de Redes Neurais

Existem diversos algoritmos de aprendizado profundo. O algoritmo base da maioria deles é o algoritmo do gradiente descendente estocástico, uma extensão do algoritmo gradiente descendente (GOODFELLOW; BENGIO; COURVILLE, 2016). Um algoritmo de aprendizado importante para as redes neurais modernas é o *Backpropagation*, ele torna possível o treinamento de redes de aprendizagem profunda nos modelos recentes, tornando o aprendizado mais rápido e eficiente (Data Science Academy, 2019).

O algoritmo do gradiente descendente serve como base para maioria dos algoritmos de aprendizado, pois esses algoritmos estão preocupados em minimizar ou maximizar uma

função, ou seja otimizá-la. Essa função é chamada de função objetiva, se o objetivo for minimizar a função é denominada função de custo, erro ou perda (GOODFELLOW; BENGIO; COURVILLE, 2016). Um dos algoritmos mais utilizados para otimização de função é o Algoritmo do Gradiente Descente (Data Science Academy, 2019).

Trabalhando de forma iterativa, esse algoritmo se torna uma boa ferramenta de otimização de funções complexas, tem como objetivo encontrar o mínimo de uma função arbitrária. Em algumas funções existe apenas um mínimo, porém em funções mais realistas podem haver diversos mínimos, nos quais a maioria são locais, o que se torna um problema, pois o objetivo é encontrar o mínimo global ou que mais se aproxima dele (Data Science Academy, 2019).

A maneira mais rápida de descer na direção do mínimo mais próximo, é estimando valores de gradiente, com uma diferença entre eles, a escolha dessa diferença pode acelerar a otimização no início mas pode causar uma oscilação em torno do mínimo (CHOW; CHO, 2007).

Ao iniciar o treinamento de um rede neural artificial é necessário especificar uma medida de erro, chamada de função de custo, usada durante o treinamento para ajustar um conjunto de pesos $\vec{w}$, sob os dados de treinamento. Existem diversas funções de custo, a Equação 21 define o erro quadrático médio (Data Science Academy, 2019; MITCHELL, 1997).

$$E(\vec{w}) \equiv \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2 \tag{21}$$

Na qual D, é o conjunto de dados de treinamento, t_d é a saída desejada para o d-ésimo dado do conjunto de treinamento e o_d é a saída obtida do neurônio artificial sob o dado d do conjunto de treinamento. Essa equação define E em função de $\vec{w}$, porque a saída obtida o depende desse vetor de peso (MITCHELL, 1997). Para calcular a direção da descida mais ingrime ao longo da superfície do erro, é necessário derivar E em relação a cada peso e *bias* conforme mostra a Equação 22 , o vetor resultante é chamado de gradiente de E em relação a $\vec{w}$ (MITCHELL, 1997).

$$\bigtriangledown E(\vec{w}) \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, ..., \frac{\partial E}{\partial w_n}, \frac{\partial E}{\partial b} \right] \tag{22}$$

O gradiente da Equação 22 define a direção que gera um aumento de E e o negativo desse vetor mostra a direção da diminuição mais acentuada, logo a regra de treinamento da descida do gradiente é definida pela Equação 23.

$$\begin{aligned} w_i &\leftarrow w_i + \triangle w_i \\ \triangle w_i &= -\eta \frac{\partial E}{\partial w_i} \end{aligned} \tag{23}$$

Em que η é uma constante positiva definida como taxa de aprendizado, que indica o tamanho do passo na descida do gradiente, é antecedida por um sinal negativo pois a direção do vetor desejada é na direção em que o custo diminui (MITCHELL, 1997).

O passo a passo do procedimento de otimização utilizando descida do gradiente, segundo Chow e Cho (2007) é o seguinte:

- Passo 1: iniciar os parâmetros de forma aleatória;
- Passo 2: avaliar o gradiente da função de erro em relação a cada parâmetro do modelo;
- Passo 3: ajustar os parâmetros do modelo com determinado tamanho de passo na direção do gradiente mais acentuado;
- Passo 4: repetir os passos 2 e 3 até encontrar o mínimo.

2.4.3 Redes Neurais Convolucionais

As redes neurais convolucionais são uma classe das redes neurais artificiais tipicamente usadas no processamento de imagens. A subdivisão das camadas nessa rede pode ser definida da seguinte forma: as camadas iniciais são voltadas para extração de características dos dados e os neurônios não são totalmente conectados com a próxima camada. Já as camadas finais, que são totalmente conectadas, são focadas em interpretar e gerar uma resposta a partir das características extraídas pelas camadas iniciais (EBERMAM; KROHLING, 2018). Por meio desse tipo de rede neural é possível receber uma imagem, atribuir valor a objetos desta imagem e fazer um procedimento de diferenciação entre estes objetos (Data Science Academy, 2019). Esse tipo de rede tem a capacidade de aplicar filtros em dados visuais, e a importante característica de manter a relação de vizinhança entre os *pixeis* da imagem (VARGAS; PAES; VASCONCELOS, 2016).

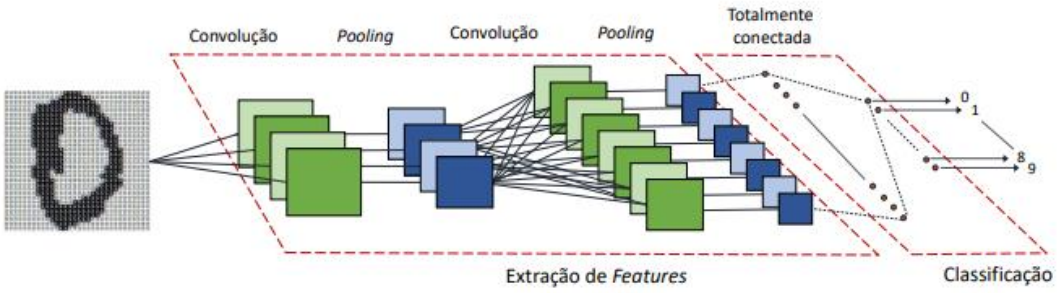


Figura 13 – Arquitetura de uma rede neural convolucional

Fonte: Becker (2017)

Na Figura 13 é possível visualizar arquitetura de um rede neural convolucional para processamento de imagens. Pode-se destacar camadas de grande importância para o seu funcionamento, a de convolução e de agrupamento (*pooling*) e a camada completamente conectada.

Uma importante característica das Redes Neurais Convolucionais, são os campos receptivos locais, localizados nas camadas iniciais da rede, nas quais cada neurônio da primeira camada oculta é conectado a uma pequena região dos neurônios da camada de entrada que é responsável por se conectar com cada *pixel* de uma imagem. Essa conexão aprende um peso e um *bias* geral (Data Science Academy, 2019), compartilhado por outros neurônios. Esse aspecto dos campos respectivos locais força que cada neurônio extraia recursos de uma região da imagem (HAYKIN, 2009).

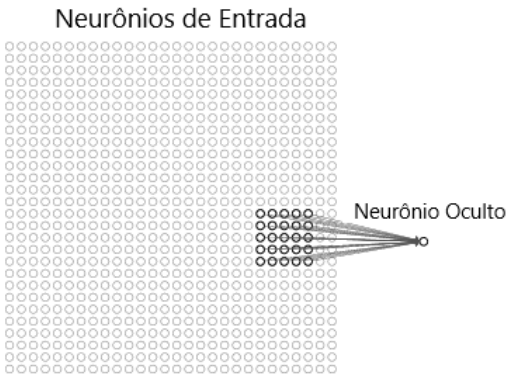


Figura 14 – Campo Receptivo Local

Fonte: Adaptado de Data Science Academy (2019)

Na Figura 14, é possível observar que o neurônio oculto está conectado a uma região de neurônios da entrada (Data Science Academy, 2019).

Esse agrupamento de saídas dos neurônios da camada de entrada pode ser definido como mapa de recursos, cada mapa detecta na imagem uma característica comum, como por exemplo uma borda vertical. Logo, todos os neurônios da primeira camada oculta estão detectando uma mesma característica, em locais diferentes de uma mesma imagem (Data Science Academy, 2019). Para que esse processo aconteça é necessário um compartilhamento de pesos e *bias*, que é uma outra grande caraterística das redes convolucionais (VARGAS; PAES; VASCONCELOS, 2016). Esse compartilhamento tem a grande vantagem de diminuir de forma significativa o número de parâmetros a serem aprendidos além de diminuir o tempo de treinamento da rede.

Na Figura 15 é possível visualizar como é realizado o processo de convolução, em que cada fragmento da entrada é multiplicado pelo peso ou *kernel*, resultando um valor na saída.

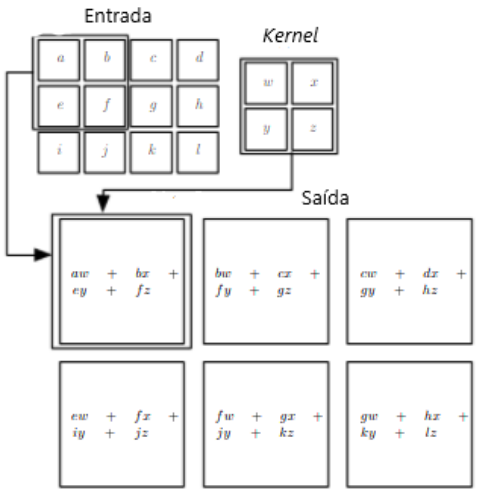


Figura 15 – Processo de convolução

Fonte: Adaptado de Goodfellow, Bengio e Courville (2016)

Outra camada importante nesse tipo de rede é a de *pooling* ou agrupamento. Nela a dimensão da representação dos dados é reduzida, tem como vantagem diminuir a computação necessária para a próxima camada (FERREIRA, 2017). A aplicação desse processo agiliza o treinamento da rede e evidencia o potencial desse tipo de rede (BECKER, 2017). Uma técnica utilizada para essa redução de dimensionalidade, é o *max-pooling*, em que é selecionado o máximo local em um fragmento do mapa de atributos, na Figura 16 é ilustrado esse processo (BECKER, 2017; FERREIRA, 2017).

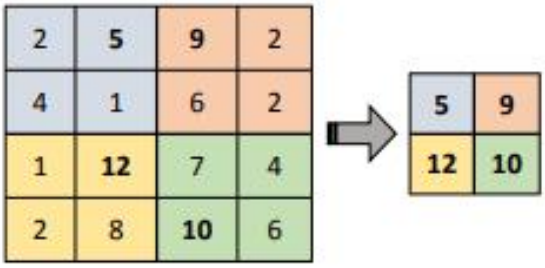


Figura 16 – Exemplo de *max-pooling* com deslocamento 2x2
Fonte: Becker (2017)

2.4.4 Redes Neurais Adversárias Generativas

As redes neurais adversárias generativas possuem uma arquitetura formada por duas redes adversárias, que são colocadas uma contra a outra. Essas duas redes são definidas como geradora e discriminadora (KUPYN et al., 2018). Uma rede adversária generativa tem a capacidade de criar um novo conteúdo, por exemplo gerar uma nova imagem ou compor uma nova música. Logo esse tipo de rede tem um grande potencial se tratando de aprendizado profundo (Data Science Academy, 2019).

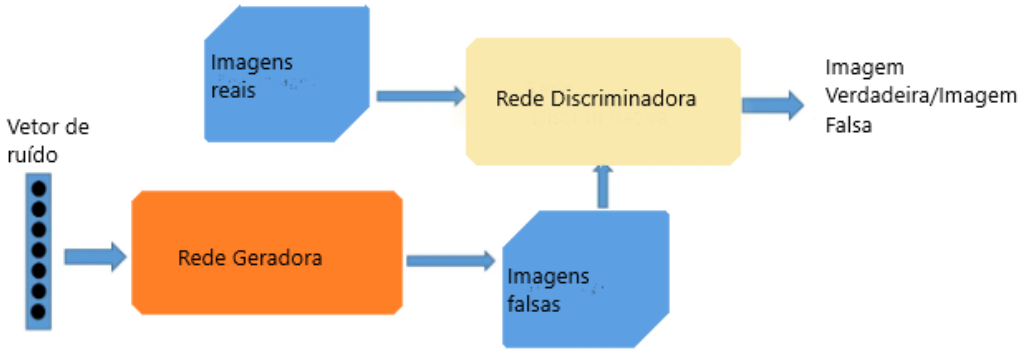


Figura 17 – Exemplo de como funciona uma Rede Neural Adversária Generativa
Fonte: Adpatado de Data Science Academy (2019)

Na Figura 17 é possível visualizar como funcionam as duas redes que compõe o modelo de rede adversária generativa. A rede geradora recebe um ruído como entrada e gera uma imagem que não pertence ao conjunto de imagens de treinamento. Ela tem o objetivo de confundir a rede discriminadora que por sua vez recebe ora uma imagem que pertence ao conjunto de treinamento e ora uma imagem gerada pela outra rede e tenta diferenciar entre elas (Data Science Academy, 2019; KUPYN et al., 2018). A relação das duas redes adversárias pode ser definida matematicamente no qual as duas redes estão tentando otimizar uma função objetivo diferente e oposta (Data Science Academy, 2019).

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{dados}(b)}[\log D(b)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))] \tag{24}$$

A Equação 24 define o modelo de treinamento da rede geradora e descriminadora, no qual $p_{dados}(b)$ é a distribuição da rede geradora sobre os dados b, $p_z(z)$ é uma prévia das variáveis de ruído de entrada, $G(z;\theta g)$ é a representação do espaço de dados, no qual G é uma função diferenciável representada por uma rede multicamadas com parâmetros θg, também é definida a outra rede denominada como D que gera um único escalar e $D(B)$ representa a probabilidade de que o dado b tenha vindo dos conjunto de treinamento ao invés de p_g (GOODFELLOW et al., 2014). Logo, a rede discriminadora D é treinada para maximizar a probabilidade do dado ser verdadeiro e a rede geradora G é treinada simultaneamente para minimizar o acerto da rede D (GOODFELLOW et al., 2014).

2.5 MÉTRICAS DE QUALIDADE DA RESTAURAÇÃO DE IMAGEM

Existem diversos critérios que avaliam a qualidade da restauração de uma imagem desborrada, além da percepção humana ao visualizar a imagem. Dentre as diferentes métricas, o Índice de similaridade estrutural (Structural Similarity Index- SSIM) é um dos mais populares (MOSQUERA, 2015). O SSIM calcula a similaridade da estrutura entre duas imagens, o cálculo desse índice é realizado a partir da Equação 25 (SAHU; LENKA; SA, 2019).

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C2)}{(\mu_x^2 + \mu_y^2 + C1)(\sigma_x^2\sigma_y^2 + C2)} \tag{25}$$

No qual x e y são respectivamente a imagem original e imagem borrada, μ_x e μ_y são as médias de x,y e σ_x e σ_y, a variância de x, y, σ_{xy} é a covariância de x, y e $C1$ e $C2$ são constantes para establizar a divisão (SAHU; LENKA; SA, 2019). Outra métrica bastante utilizada para medir a

semelhança entre a imagem borrada e a restaurada é a Relação sinal-ruído de pico (Peak Signal to Noise Ratio-PSNR). A PSNR depende de uma outra medida, a do erro quadrático médio (Mean Squared Error-MSE) que pode ser calculado usando a Equação 26 (SAHU; LENKA; SA, 2019).

$$MSE = \frac{\sum_{P,Q}(I-L)^2}{P \times Q} \tag{26}$$

No qual P,Q são as dimensões da imagem, I é a imagem original e L é a imagem desborrada (SAHU; LENKA; SA, 2019). Com o valor do MSE, é possível calcular a PSNR usando a partir da Equação 27 (SAHU; LENKA; SA, 2019).

$$PSNR = \frac{m^2}{MSE} \tag{27}$$

No qual m é o valor de intensidade máxima possível. Porém tanto o MSE quanto a PSNR, podem não oferecer uma estimativa verdadeira da qualidade das imagens restauradas (MOSQUERA, 2015).

3 MATERIAIS E MÉTODOS

Após o estudo sobre os conceitos de processamento de imagens, técnicas de desborramento de imagens e sobre algoritmos de *Deep Learning*, serão definidos neste capítulo os materiais e métodos necessários para a implementação de algoritmos de desborramento, e também métodos de análise e avaliação do desempenho das técnicas empregadas.

3.1 MATERIAIS

Nesta seção são apresentados os materiais e as bibliotecas que foram utilizados na realização deste trabalho.

3.1.1 Hardware

O computador utilizado para o desenvolvimento das técnicas clássicas de desborramento de imagem e para o treinamento da Rede Neural Artificial foi um computador em nuvem, o Google Colab, com um processador Intel(R) Xeon(R) CPU @ 2.30GHz, utilizando ambiente de execução TPU, que é uma arquitetura projetada para acelerar o processo de aprendizagem profunda. Para testes iniciais, com as técnicas clássicas de desborramento, o computador que será utilizado é um notebook, com o sistema operacional Windows 10 e um processador Intel Core i5-5200U com dois núcleos de processamento e quatro *threads* com *clock* 2.20 GHz, 6 GB de memória RAM.

3.1.2 Software

Nesta seção são listados a linguagem de programação, o ambiente de desenvolvimento e as bibliotecas utilizadas nesse trabalho.

- Python: A linguagem de programação utilizada foi o Python[1] na versão 3.8, é uma linguagem de código aberto bastante utilizada na área de Inteligência Artificial, e também oferece recursos voltados para computação numérica e científica, e ainda oferece suporte para o processamento de imagens;
- Google Colaboratory: O ambiente de desenvolvimento foi o Google Colab[2] Notebook, é um ambiente de código aberto baseado na Web, nele é possível realizar o compartilhamento de código, visualização e transformação de dados, comunicar os resultados, integrar o código, a documentação e a saída como imagens e gráficos no mesmo arquivo, é utilizado na área de ciência de dados, aprendizagem de máquina entre outras;
- NumPy: Numpy[3] é uma biblioteca Python, tem como uma das principais funcionalidades a manipulação e cálculo com matrizes multidimensionais, executando esses cálculos de maneira rápida e eficiente;
- Matplotlib: Matplotlib [4] é uma biblioteca de código aberto, desenvolvida para o Python, indicada para visualização de imagens, gráficos e dados em geral;
- OpenCV: Para o processamento de imagens foi utilizado OpenCV [5], que é uma biblioteca de visão computacional de código aberto, amplamente utilizada em processamento de imagens. Possui interface para Python e mais de 2.500 algoritmos otimizados voltados para processamento de imagem e aprendizagem de máquina;
- Scikit-Image: Scikit-Image [6] é uma biblioteca desenvolvida em Python, possui diversos algoritmos de processamento de imagem, tem seu código aberto. Neste trabalho serviu como complemento ao OpenCV;
- Keras: Para auxiliar na criação da rede neural artificial foi utilizada a biblioteca Keras[7], é uma API de alto nível que facilita a implementação de aprendizagem profunda, ela executa por meio de bibliotecas de computação numérica, dentre elas o TensorFlow [8] que

[1] https://www.python.org
[2] https://colab.research.google.com/
[3] https://numpy.org/
[4] https://matplotlib.org/
[5] https://opencv.org/about/
[6] https://scikit-image.org/
[7] https://keras.io/
[8] https://www.tensorflow.org/?hl=pt-br

foi utilizado neste trabalho.

3.1.3 Conjunto de dados

Este trabalho utilizou o conjunto de dados UFPR-ALPR. Esse conjunto de dados possui 4.500 imagens de 150 carros, essas imagens foram capturadas em cenários do mundo real, estão em formato PNG e resolução de 1920x1080 (LAROCA et al., 2018). A Figura 18 representa exemplos de imagens contidas no conjunto de dados. No conjunto de dados, cada

Figura 18 – Amostra do conjunto de dados- UFPR-ALPR

Fonte: Laroca et al. (2018)

veículo possui 30 imagens e para evitar que a rede neural tenha o problema de sobreajuste (*overfitting*), no treinamento foram utilizadas 2 imagens de cada veículo e utilizou-se métodos de aumento de dados como giro vertical, rotação das imagens, gerando uma base de treinamento com 3240 imagens.

3.2 MÉTODOS

Os procedimentos deste trabalho foram executados como no diagrama de sequência de atividades, disposto na Figura 19.

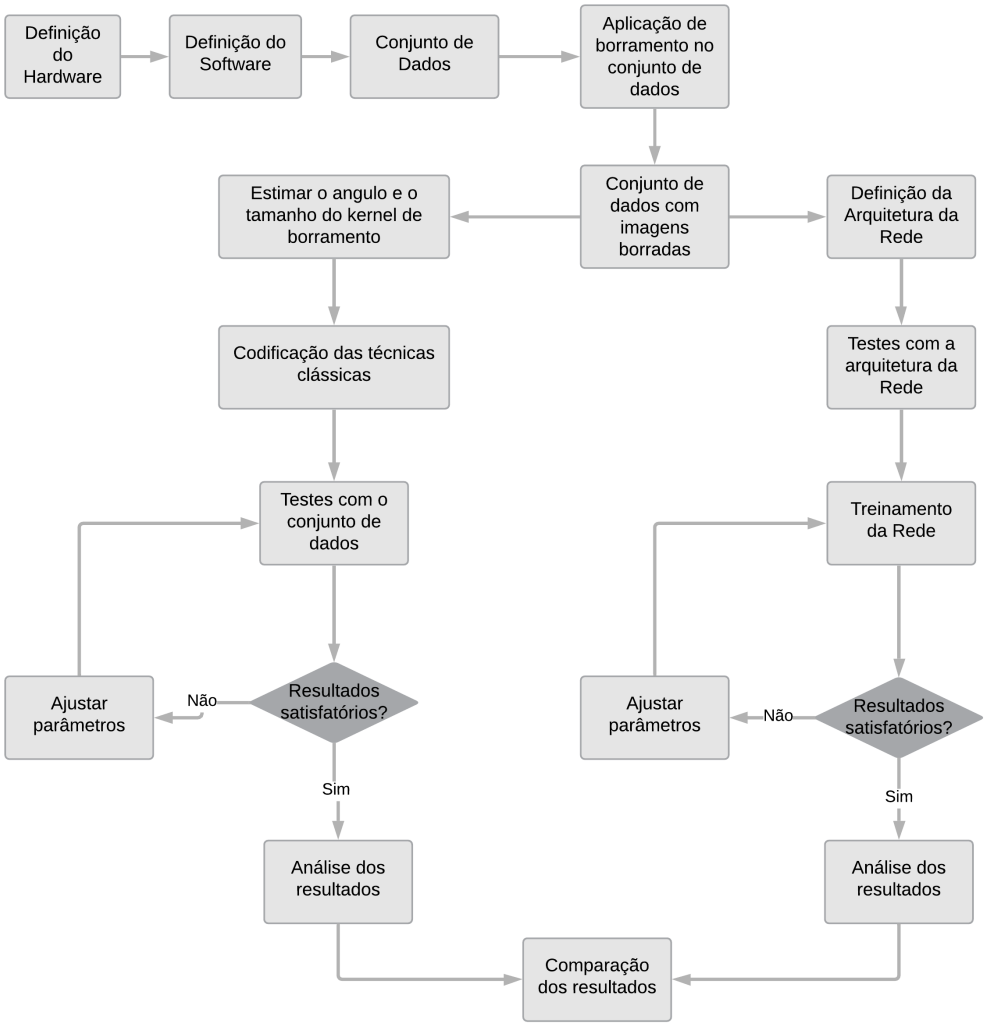


Figura 19 – Diagrama de sequência de atividades
Fonte: Autoria Própria

Inicialmente foi necessário aplicar o borramento no conjunto de dados, foram utilizados os ângulos de 30º, 90º e 120º e tamanhos de *kernel* 15, 27 e 33, esses parâmetros foram definidos empiricamente, por meio de resultados visuais nas imagens. As imagens foram cortadas em 256x256, utilizando como referência o centro da figura. Para degradar o conjunto de dados foi necessário criar um programa utilizando a linguagem de programação Python

e as bibliotecas Numpy e OpenCV que tem como entrada a imagem original, o ângulo do borramento, o tamanho do *kernel* e a saída a imagem borrada.

Conforme discutido na Subseção 2.2.4.1, para utilizar o Filtro de Wiener e o Algoritmo de Lucy-Richardson, foi necessário estimar os ângulos de borramento e o tamanho do *kernel* de degradação. Utilizando os algoritmos propostos na Subseção 2.2.4.3, as bibliotecas de processamento de imagem OpenCV e Scikit-image, Numpy para manipulação das matrizes e cálculo do transformada de Fourier e Matplotlib para visualizar o espectro de Fourier, foram estimados os parâmetros de borramento na imagem degradada. Após conhecer os parâmetros de borramento, foi codificado o Filtro de Wiener e o Algoritmo de Lucy-Richardson, usando as bibliotecas OpenCV e Numpy e aplicando os conceitos vistos nas Subseções 2.2.4.4 e 2.2.4.5.

A arquitetura de Rede Neural Artificial utilizada foi baseada nas redes de Kupyn et al. (2018) e Ramakrishnan et al. (2018), nas quais as redes foram treinadas de forma adversária, conforme visto na Subseção 2.4.4. A rede geradora recebe a imagem borrada e tenta desborrar a imagem, enquanto a rede discriminadora recebe a imagem gerada pela rede adversária e a imagem original sem o borramento. A arquitetura da rede geradora utilizada por Kupyn et al. (2018), pode ser vista na Figura 20. Com o auxilio das bibliotecas Keras e Tensorflow, foi possível definir os parâmetros das duas redes, incluindo detalhes do treinamento, função de custo a ser utilizada, funções de ativação, otimizadores, entre outros. Na arquitetura de Kupyn et al. (2018) a taxa de *dropout*, utilizada após a primeira camada de convolução é de 0.5. Nesta arquitetura são utilizadas as funções ReLu e Leaky Relu, na qual, a constante é 0.2 Um dos otimizadores utilizados em Kupyn et al. (2018) foi o algoritmo Adam.

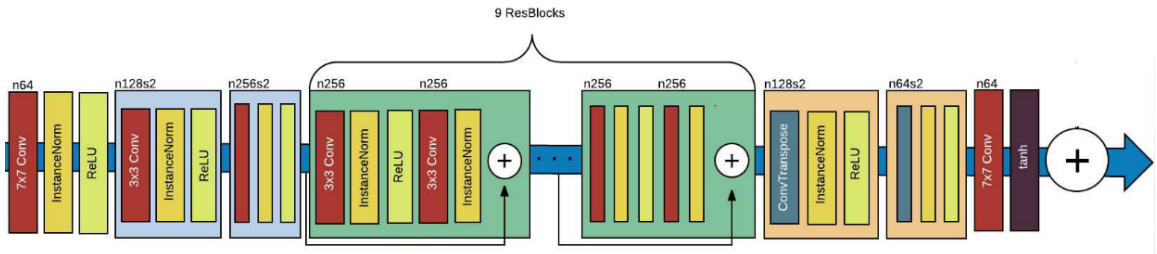


Figura 20 – Arquitetura Rede Geradora

Fonte: Adaptado de Kupyn et al. (2018)

Para analisar os resultados dos métodos, são utilizadas as métricas PSNR e SSIM, discutidas na Seção 2.5.

4 RESULTADOS E DISCUSSÃO

Utilizando os materiais e métodos previamente apresentados, neste capítulo são apresentados os resultados a partir das imagens borradas com diferentes ângulos e tamanhos de *kernel*. A partir de testes prévios com diferentes parâmetros foram definidos empiricamente, nove combinações com 118 imagens em cada uma, resultando em um total de 1062 imagens. As combinações foram: borramento de 30^0, com *kernel* de 15, 27 e 33; borramento de 90^o, com *kernel* de 15, 27 e 33; e borramento de 120^o, com *kernel* de 15, 27 e 33.

4.1 EXPERIMENTO 1: ABORDAGEM UTILIZANDO TÉCNICAS DE PROCESSAMENTO DE IMAGEM

Nesta seção serão apresentados os resultados obtidos utilizando o desborramento com as técnicas de processamento de imagem: Filtro de Wiener e algoritmo de Richardson-Lucy. Para utilizar essas técnicas é necessário estimar o ângulo e o tamanho de *kernel* de borramento.

4.1.1 Estimando ângulo do borramento

Nesse experimento buscou-se a estimativa do ângulo de borramento, para tanto foi utilizado o algoritmo descrito na Seção 2.2.4.3. Foi gerada uma margem de erro para o algoritmo de 2^o para mais e para menos. A Tabela 1 mostra as médias dos ângulos estimados dentre as imagens testadas, o desvio padrão, a taxa de acerto considerando a margem de erro e a quantidade de ângulos estimados fora da margem de 2^o para mais e para menos do real.

O ângulo que teve o melhor resultado foi o de 90^o, independentemente do tamanho

de *kernel*, obteve uma taxa de acerto acima de 99%. Considerando o tamanho de *kernel* o que teve o melhor desempenho, sempre acima de 92% de ângulos estimados corretamente, dentro da margem, foi o tamanho 15.

Tabela 1 – Resultados obtidos para estimação de ângulos

Ângulo - Kernel	Média	Desvio Padrão	Fora da Margem	Taxa Acerto (Dentro da Margem)
30º-15	28,01	4,96	7	93,22%
30º-27	27,60	5,19	13	88,98%
30º-33	29,14	14,29	16	86,44%
90º-15	89,77	8,19	1	99,15%
90º-27	88,21	8,19	1	99,15%
90º-33	88,97	0,33	0	100,00%
120º-15	117,43	16,43	9	92,37%
120º-27	119,24	14,17	17	85,59%
120º-33	117,20	16,05	18	84,75%

Fonte: Autoria Própria

É possível observar no gráfico da Figura 21, que a maioria dos ângulos estimados para a combinação *kernel* 15 e 30º, ficaram bem próximos do real, das 118 imagens 12 foram calculadas dentro da margem, porém somente 7 ficaram fora da margem de 2º.

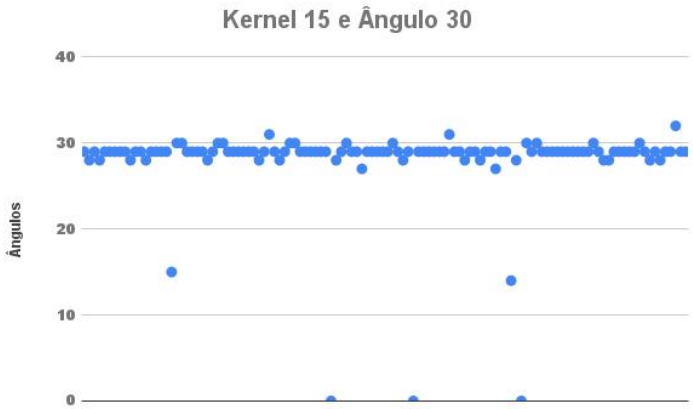


Figura 21 – Gráfico de ângulos estimados para *kernel* 15 e ângulo 30º

Fonte: Autoria Própria

O mesmo comportamento da Figura 21, pode ser visualizado na Figura 22, mesmo com apenas 7 imagens com o ângulo estimado exatamente com o que foi borrado, a maioria ficou dentro da margem, apenas 13 ficaram de fora.

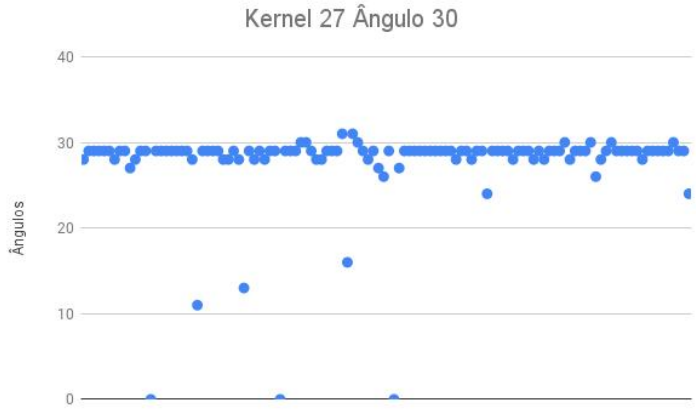


Figura 22 – Gráfico de ângulos estimados para *kernel* 27 e ângulo 30º
Fonte: Autoria Própria

Os resultados para esse algoritmo foram satisfatórios em todos os parâmetros, com altas taxas de acerto.

4.1.2 Estimando o tamanho de *kernel*

Utilizando o algoritmo previamente discutido em 2.2.4.3, da mesma maneira da Seção 4.1.1 foi aplicado uma margem de erro de tamanho 2 para mais e para menos. Para estimar o tamanho de *kernel* de borramento, as imagens borradas utilizando o tamanho de 15 e ângulo 120º, foi possível obter uma taxa de acerto de 91,53%, com a dimensão 15 sempre com a média aproximadamente igual ao que foi borrado.

Na Tabela 2 é possível notar que, quanto maior o tamanho do *kernel*, menor é a taxa de acerto, ou seja, quanto mais borrada a imagem mais difícil de estimar.

Tabela 2 – Resultados obtidos na estimação do tamanho de *kernel*

Ângulo - Kernel	Média	Desvio Padrão	Fora da Margem	Taxa Acerto (Dentro da Margem)
30º-15	15,11	3,08	10	91,53%
90º-15	15,10	3,03	12	89,83%
120º-15	15,09	2,76	9	92,37%
30º-27	22,96	4,26	62	47,46%
90º-27	21,07	4,80	84	28,81%
120º-27	24,12	3,96	52	55,93%
30º-33	23,87	6,00	102	13,56%
90º-33	22,08	5,88	104	11,86%
120º-33	25,49	5,84	95	19,49%

Fonte: Autoria Própria

O ângulo que obteve melhores resultados, foi o de 120º, em cada tamanho de *kernel*.

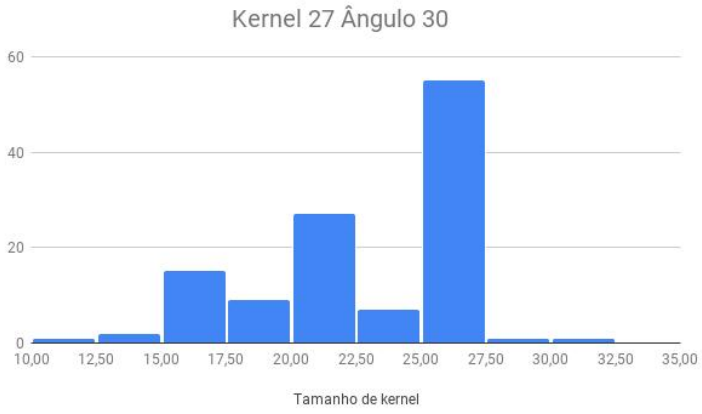


Figura 23 – Histograma para tamanhos estimados para *kernel* 27 e ângulo 30º

Fonte: Autoria Própria

Conforme é possível visualizar na Figura 23, apesar de ter bastante variação entre os tamanhos de *kernel* estimados, a maioria está entre 25 e 27,5. Considerando as imagens com borramento de *kernel* 27 e ângulo 30º, obteve uma taxa de acerto de 47,46 %.

4.1.3 Filtro de Wiener

Nessa seção serão apresentados os resultados das restaurações das imagens utilizando o filtro de Wiener. A partir da imagem borrada, foram estimados os ângulos e tamanho de *kernel*, e aplicado o filtro, de acordo com o que foi apresentado na seção 2.2.4.4.

Na Tabela 3 são apresentados os resultados deste experimento, no qual são apresentados a média do SSIM, e do PSNR de todas as 118 imagens, de cada uma da relação ângulo/*kernel*, além de apresentar o valor máximo obtido, de cada métrica, entre todos os parâmetros testados.

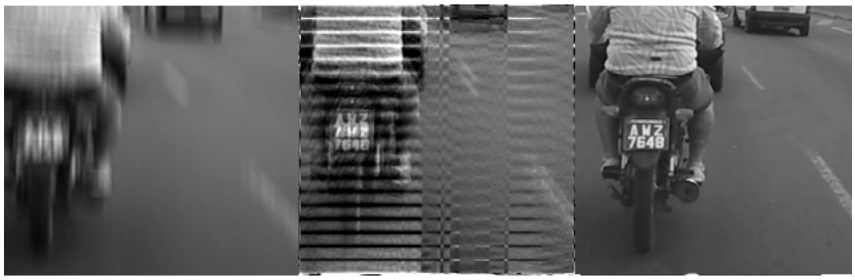
Tabela 3 – Filtro de Wiener

Ângulo - Kernel	Média SSIM	Média PSNR
30º-15	0,12514	13,24461
90º-15	0,16679	13,77124
120º-15	0,11939	13,08978
30º-27	0,09368	11,99615
90º-27	0,12657	12,76448
120º-27	0,08712	11,63789
30º-33	0,09323	11,97344
90º-33	0,12589	12,61717
120º-33	0,08103	11,58006

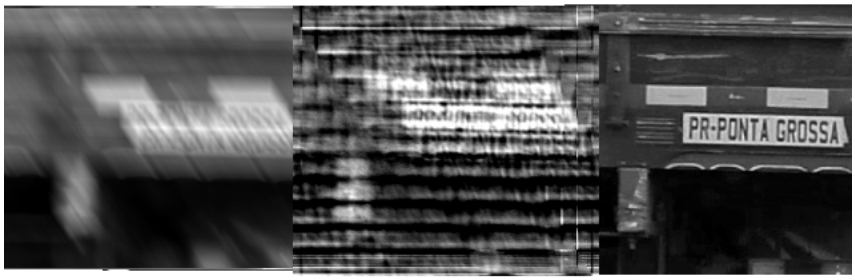
Fonte: Autoria Própria

A média do Índice de Similaridade estrutural foi de aproximadamente 0,11 e de PSNR de 12,51, o maior SSIM obtido foi de 0,66, com a imagem borrada com ângulo de 30º e tamanho de *kernel* 27, de PSNR foi de 21,81, com os parâmetros de borramento com *kernel* 15 e ângulo 120º. O ângulo com melhor desempenho, em cada tamanho de *kernel*, foi de 90 º.

(a) Imagem borrada com *kernel* 15 e ângulo 30º. Imagem restaurada, SSIM=0,21 e PSNR = 15,16. Imagem Original



(b) Imagem borrada com *kernel* 27 e ângulo 90º. Imagem restaurada, SSIM=0,22 e PSNR = 12,61. Imagem Original



(c) Imagem borrada com *kernel* 33 e ângulo 120º. Imagem restaurada, SSIM=0,13 e PSNR = 14,70. Imagem Original

Figura 24 – Exemplos de restauração utilizando filtro de Wiener

Fonte: Autoria Própria

É possível observar pela Figura 24(a) que apesar de um SSIM baixo, é possível identificar o que está escrito na imagem. As imagens resultantes do filtro de Wiener, contém bastante ruído ocasionando um índice de similaridade baixo. Quanto maior o tamanho de *kernel*, mais dificuldade o filtro teve para restaurar a imagem, ocasionando bastante ruído e dificultando a análise.

4.1.4 Algoritmo de Richardson-Lucy

Nesta seção serão exibidos os resultados das restaurações utilizando o algoritmo de Richardson-Lucy apresentado na Seção 2.2.4.5, partindo dos resultados dos algoritmos de estimação de ângulo e tamanho de *kernel*.

Com a mesma estrutura da Tabela 3, os resultados do algoritmo de Richardson-Lucy são apresentados pela Tabela 4. Observando a Tabela 4 é possível notar que os melhores desempenhos desse algoritmo foram com o tamanho de *kernel* 15. Com os parâmetros 90º-15, obteve um SSIM de 0,84 , um PSNR de 31,16 e também atingiu as melhores médias considerando as duas métricas calculadas. Este fato está ligado ao desempenho desses parâmetros nos resultados de estimação de ângulo e tamanho de *kernel*.

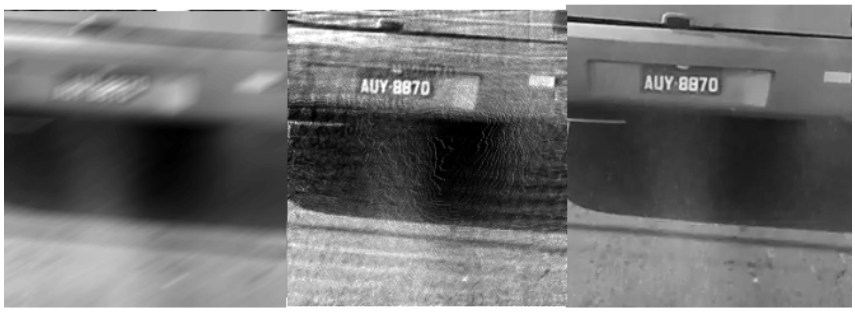
Tabela 4 – Richardson Lucy

Ângulo - Kernel	Média SSIM	Média PSNR
30º-15	0,535160	21,507965
90º-15	0,564253	22,318486
120º-15	0,517644	20,369277
30º-27	0,322764	17,323834
90º-27	0,292129	16,832813
120º-27	0,234871	13,366430
30º-33	0,192649	15,350858
90º-33	0,232327	15,851469
120º-33	0,140579	11,624441

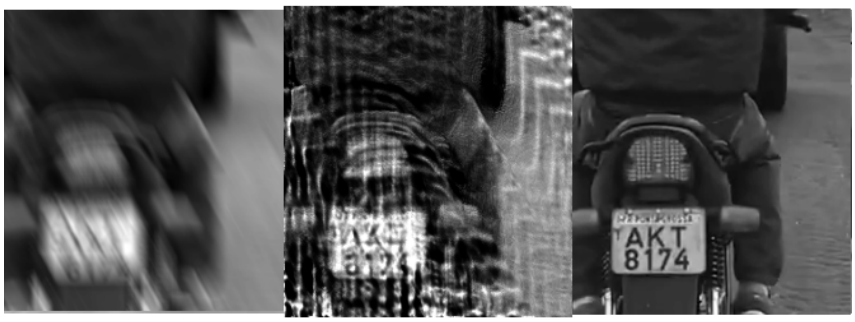
Fonte: Autoria Própria

Na Figura 25(a) é possível observar que a imagem restaurada e a imagem original ficaram visivelmente parecidas. Já na Figura 25b, apesar de um SSIM baixo, é possível visualizar qual a placa do ônibus, a Figura 25(c), é uma exemplificação de como foi o desempenho com os parâmetros de ângulo 120º e kernel 33, com uma média de SSIM 0,14 e PSNR 11,62, a maioria das imagens ficaram visualmente distantes da original.

(a) Imagem borrada com *kernel* 15 e ângulo 90º. Imagem restaurada, SSIM=0,84 e PSNR = 31,16. Imagem Original



(b) Imagem borrada com *kernel* 27 e ângulo 30º. Imagem restaurada, SSIM=0,34 e PSNR = 18,38. Imagem Original



(c) Imagem borrada com *kernel* 33 e ângulo 120º. Imagem restaurada, SSIM=0,21 e PSNR = 15,17. Imagem Original

Figura 25 – Exemplos de restauração utilizando algoritmo de Richardson-Lucy
Fonte: Autoria Própria

Comparando com o Filtro de Wiener, o algoritmo de Richardson-Lucy, foi superior em todos os parâmetros testados, além das imagens restauradas possuírem menos ruído. Os desempenhos dos dois algoritmos estão conectados com os resultados dos procedimentos para estimar os parâmetros.

4.2 EXPERIMENTO 2: ABORDAGEM UTILIZANDO *DEEP LEARNING*

Este experimento utilizou uma rede neural para restaurar as imagens borradas com diferentes parâmetros. Os melhores resultados dentre os parâmetros testados, foram com as imagens borradas com um tamanho de *kernel* 15 e ângulo de borramento 30º, obtendo uma média de SSIM de 0,86, e PSNR 24,60.

Tabela 5 – Rede Neural Adversária Generativa

Ângulo - Kernel	Média SSIM	Média PSNR
30º-15	0,860245	24,606461
90º-15	0,847120	24,510445
120º-15	0,837476	24,142635
30º-27	0,774501	22,166279
90º-27	0,696111	21,691801
120º-27	0,749703	21,562296
30º-33	0,736356	21,233536
90º-33	0,696111	20,788075
120º-33	0,678690	20,422856

Fonte: Autoria Própria

As piores médias das métricas utilizadas, foram com o tamanho de *kernel* 30 e ângulo 120º. Ambos valores máximos obtidos nas duas métricas foram com a degradação da imagem utilizando um tamanho de *kernel* 15, com o ângulo 30º alcançou um valor de SSIM de 0,95, com 120º obteve um PSNR de 34,12.

(a) Imagem borrada com *kernel* 15 e ângulo 120º. Imagem restaurada, SSIM=0,85 e PSNR = 21,90. Imagem Original

(b) Imagem borrada com *kernel* 27 e ângulo 120º. Imagem restaurada, SSIM=0,57 e PSNR = 19,37. Imagem Original

(c) Imagem borrada com *kernel* 33 e ângulo 90º. Imagem restaurada, SSIM=0,83 e PSNR = 25,8. Imagem Original

Figura 26 – Exemplos de restauração utilizando *Deep Learning*

Fonte: Autoria Própria

Mesmo as piores médias de desempenho da rede, em todos os parâmetros foram superiores em comparação com as duas técnicas de processamento de imagem. É possível observar na Figura 26, as imagens restauradas pela rede neural possuem poucos artefatos, em comparação com as técnicas clássicas, gerando um maior índice de similaridade estrutural e um PSNR alto.

4.3 COMPARAÇÃO DOS RESULTADOS

Conforme visto nas seções anteriores, a rede teve melhores médias de PSNR e SSIM em todos os parâmetros, em testados em relação as outras duas técnicas. Como pode ser visto na Figura 27, entre as três restaurações, a que tem melhores métricas e visualmente possui menos ruído e artefatos, é a da rede, em seguida da restauração de Richardson-Lucy e Filtro de Wiener. Ainda observando a Figura 27 , o SSIM da restauração da rede é o dobro do valor do SSIM da recuperação da imagem utilizando Richardson-Lucy.

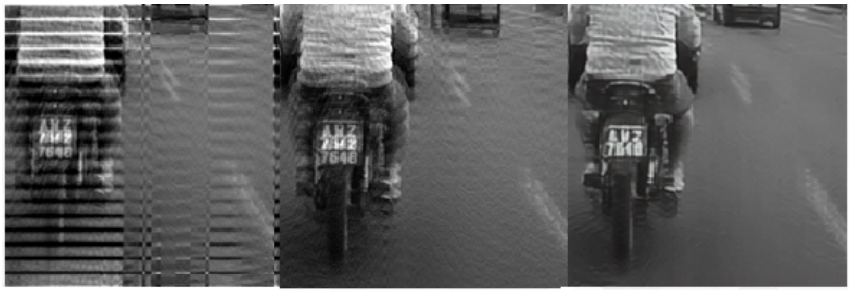


Figura 27 – (a) Filtro de Wiener SSIM:0,2510 PSNR:16,36. (b) Richardson-Lucy SSIM:0,4407 PSNR:20,19. (c) Rede Neural Artificial SSIM:0,8482 PSNR:27,79.

Fonte: Autoria Própria

A restauração de Filtro de Wiener tem bastante ruído o que afeta o índice de similaridade e visualmente fica mais difícil de identificar o objeto na imagem.No Apêndice A é possível observar as restaurações em todos os parâmetros, e em todos o destaque fica para a restauração da rede, em alguns parâmetros Richardson-Lucy, visualmente apresenta boas restaurações, porém quando aumenta a dificuldade pior fica o resultado

5 CONSIDERAÇÕES FINAIS

O presente trabalho comparou técnicas clássicas de *deblurring* e uma Rede Neural Artificial Adversária, contribuindo para a análise das imagens após a restauração. Para alcançar esse objetivo, foram borradas as imagens com diferentes parâmetros de ângulo e tamanho de matriz de degradação, de forma empírica foi observado que quanto maior o tamanho de *kernel* de borramento mais degradada ficava a imagem, gerando uma maior diversidade entre as imagens e também para mensurar o desempenho dos algoritmos de estimação desses parâmetros.

Através de alguns estudos foram definidos duas técnicas clássicas, bastante utilizadas no estudo de restauração de imagens borradas, e uma Rede Neural Adversária Generativa com bons desempenhos no desborramento de imagens. Após a implementação da abordagem clássica e o treinamento da Rede, foi realizado o processo de restauração, o algoritmo de estimar o ângulo da degradação teve um resultado satisfatório, com bons índices de acerto, já o de estimação de tamanho da matriz de borramento, quando o parâmetro era de 15 obteve bons resultado, porém nos outros parâmetros ficou bem abaixo.

O resultado ruim do procedimento de estimar o tamanho, afetou os resultados das técnicas clássicas pois são diretamente relacionadas. Na abordagem utilizando processamento de imagem, o algoritmo de Richardson-Lucy obteve o melhor resultado, porém inferior a Rede Neural que obteve o melhor desempenho em todos os parâmetros testados.

5.1 TRABALHOS FUTUROS

Para possíveis trabalhos futuros, apoiado neste, sugere-se, utilizar um outro algoritmo para estimar o tamanho de *kernel*, utilizar uma Rede Neural para estimar os parâmetros de borramento, fazer uma comparação com a Rede Neural Artificial apresentada com uma outra de restauração de imagem.

REFERÊNCIAS

ABESE. **Pesquisa ABESE mapeia mercado de segurança eletrônica**. 2018. Disponível em: <https://www.abese.org.br/index.php/412-pesquisa-abese-mapeia-mercado-de-seguranca-eletronica>.

ACHARYA. **Image Processing: Principles and Applications [book review]**. [S.l.: s.n.], 2007. 610–610 p. ISSN 1045-9227. ISBN 9780471719984.

Agência Brasil. **Sistema que vai ajudar em investigação de crimes é lançado em SP**. 2019. Disponível em: <http://agenciabrasil.ebc.com.br/geral/noticia/2019-05/sistema-que-vai-ajudar-em-investigacao-de-crimes-e-lancado-em-sp>.

ALMEIDA, M. S.; ALMEIDA, L. B. Blind and semi-blind deblurring of natural images. **IEEE Transactions on Image Processing**, v. 19, n. 1, p. 36–52, 2010. ISSN 10577149.

AMUDHA, J.; PRADEEPA, N.; SUDHAKAR, R. A survey on digital image restoration. **Procedia Engineering**, v. 38, p. 2378–2382, 2012. ISSN 18777058.

ANDREWS, H. C. Digital Image Restoration: A Survey. **Computer**, IEEE, v. 7, n. 5, p. 36–45, 1974. ISSN 00189162.

BECKER, W. E. Uma abordagem de redes neurais convolucionais para análise de sentimento multi-lingual. p. 85, 2017.

BIGDELI, S. A.; ZWICKER, M. Image restoration using autoencoding priors. **VISIGRAPP 2018 - Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications**, v. 5, p. 33–44, 2018.

CAKIR, S. **Cepstral methods for image feature extraction**. Tese (Doutorado) — bilkent university, 2010.

CAMPISI, P.; EGIAZARIAN, K. **Blind Image Deconvolution: Theory and Applications**. [S.l.]: CRC Press, 2017. ISBN 9781420007299.

CANNON, M. Blind deconvolution of spatially invariant image blurs with phase. **IEEE Transactions on Acoustics, Speech, and Signal Processing**, IEEE, v. 24, n. 1, p. 58–63, 1976.

CARASSO, A. S. Direct blind deconvolution. **SIAM Journal on Applied Mathematics**, Society for Industrial and Applied Mathematics, v. 61, n. 6, p. 1980–2007, 2001. ISSN 00361399. Disponível em: <http://www.jstor.org/stable/3061881>.

CHOW, T. W.-S.; CHO, D. S.-Y. **Neural networks and computing: Learning algorithms and applications**. [S.l.]: Imperial College Press, 2007. ISBN 9781860947582.

Data Science Academy. **Deep Learning Book**. 2019. Disponível em: <http://deeplearningbook.com.br/introducao-as-redes-neurais-convolucionais>.

DATTA, L. **A Survey on Activation Functions and their relation with Xavier and He Normal Initialization**. 2020.

EBERMAM, E.; KROHLING, R. A. Uma Introdução Compreensiva às Redes Neurais Convolucionais: Um Estudo de Caso para Reconhecimento de Caracteres Alfabéticos. **Revista de Sistemas de Informação d FSMA**, v. 22, p. 49–59, 2018.

FELDMAN, J.; ROJAS, R. **Neural Networks: A Systematic Introduction**. Springer Berlin Heidelberg, 2013. ISBN 9783642610684. Disponível em: <https://books.google.com.br/books?id=4rESBwAAQBAJ>.

FERREIRA, A. S. Redes Neurais Convolucionais Profundas na Detecção de Plantas Daninhas em Lavoura de Soja. p. 70, 2017. Disponível em: <http://www.gpec.ucdb.br/pistori/orientacoes/dissertacoes/alessandro2017.pdf>.

FISH, D. A.; WALKER, J. G.; BRINICOMBE, A. M.; PIKE, E. R. Blind deconvolution by means of the Richardson–Lucy algorithm. **Journal of the Optical Society of America A**, v. 12, n. 1, p. 58, 1995. ISSN 1084-7529.

FLUSSER, J.; FAROKHI, S.; HÖSCHL, C.; SUK, T.; ZITOVÁ, B.; PEDONE, M. Recognition of images degraded by Gaussian blur. **IEEE Transactions on Image Processing**, v. 25, n. 2, p. 790–806, 2016. ISSN 10577149.

GONZALEZ, R. C.; WOODS, R. E. **Processamento Digital De Imagens**. 3. ed. [S.l.]: ADDISON WESLEY BRA, 2011. ISBN 9788576054016.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [s.n.], 2016. Disponível em: <http://www.deeplearningbook.org>.

GOODFELLOW, I. J.; POUGET-ABADIE, J.; MIRZA, M.; XU, B.; WARDE-FARLEY, D.; OZAIR, S.; COURVILLE, A.; BENGIO, Y. Generative adversarial nets. **Advances in Neural Information Processing Systems**, v. 3, n. January, p. 2672–2680, 2014. ISSN 10495258.

HAYKIN, S. **Redes Neurais: Princípios e Prática**. Artmed, 2007. ISBN 9788577800865. Disponível em: <https://books.google.com.br/books?id=bhMwDwAAQBAJ>.

HAYKIN, S. S. **Neural networks and learning machines**. Third. Upper Saddle River, NJ: Pearson Education, 2009.

HOILUND, C. The radon transform. **Aalborg University**, v. 12, 2007.

JERIAN, M.; PAOLINO, S.; CERVELLI, F.; CARRATO, S.; MATTEI, A.; GAROFANO, L. A forensic image processing environment for investigation of surveillance video. **Forensic Science International**, v. 167, n. 2-3, p. 207–212, 2007. ISSN 03790738.

KAMILARIS, A.; PRENAFETA-BOLDÚ, F. X. Deep learning in agriculture: A survey. **Computers and Electronics in Agriculture**, v. 147, n. February, p. 70–90, 2018. ISSN 01681699.

KATSAGGELOS, A. K.; LAY, K. T. Maximum likelihood blur identification and image restoration using the em algorithm. **IEEE Transactions on Signal Processing**, v. 39, n. 3, p. 729–733, March 1991.

KHARE, C.; NAGWANSHI, K. K. Implementation and Analysis of Image Restoration Techniques. **International Journal of Computer Trends an Technology- May to June**, n. January 2011, p. 1–6, 2011.

KOPEIKA, N. S. Causes of atmospheric blur in remote sensing: a system engineering approach to imaging. In: STROJNIK, M.; ANDRESEN, B. F. (Ed.). **Infrared Spaceborne Remote Sensing VI**. SPIE, 1998. v. 3437, p. 273 – 283. Disponível em: <https://doi.org/10.1117/12.331296>.

KUNDUR, D.; HATZINAKOS, D. Blind image deconvolution. **IEEE Signal Processing Magazine**, v. 13, n. 3, p. 43–64, May 1996.

KUPYN, O.; BUDZAN, V.; MYKHAILYCH, M.; MISHKIN, D.; MATAS, J. DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, p. 8183–8192, 2018. ISSN 10636919.

LAGENDIJK, R. L.; TEKALP, A. M.; BIEMOND, J. Maximum likelihood image and blur identification: a unifying approach. **Optical Engineering**, SPIE, v. 29, n. 5, p. 422 – 435, 1990. Disponível em: <https://doi.org/10.1117/12.55611>.

LAROCA, R.; SEVERO, E.; ZANLORENSI, L. A.; OLIVEIRA, L. S.; GONCALVES, G. R.; SCHWARTZ, W. R.; MENOTTI, D. A robust real-time automatic license plate recognition based on the YOLO detector. In: **International Joint Conference on Neural Networks (IJCNN)**. [S.l.: s.n.], 2018. p. 1–10. ISSN 2161-4407.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, v. 521, n. 7553, p. 436–444, 2015. ISSN 14764687.

LEE, J.-H.; HO, Y.-S. High-quality non-blind image deconvolution with adaptive regularization. **Journal of Visual Communication and Image Representation**, v. 22, n. 7, p. 653 – 663, 2011. ISSN 1047-3203. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1047320311000848>.

LIANG, J.; LIANG, C. Identification of Motion Blur Direction Using Bresenham Algorithm for Straight Line. **Journal of Software Engineering**, Science Alert, v. 11, n. 1, p. 109–115, 2016. ISSN 18194311. Disponível em: <http://dx.doi.org/10.3923/jse.2017.109.115>.

LIMA, I.; PINHEIRO, C.; SANTOS, F. **Inteligência Artificial**. Elsevier Editora Ltda., 2016. ISBN 9788535278095. Disponível em: <https://books.google.com.br/books?id=qjJeBgAAQBAJ>.

MITCHELL, T. **Machine Learning**. McGraw-Hill, 1997. (McGraw-Hill International Editions). ISBN 9780071154673. Disponível em: <https://books.google.com.br/books?id=EoYBngEACAAJ>.

MOSQUERA, O. E. A. Implementação do algoritmo de Richardson-Lucy em arquiteturas reconfiguráveis aplicado ao problema de borramento de imagens. p. 94, 2015. Disponível em: <http://repositorio.unb.br/handle/10482/20034>.

PEDRINI, H.; SCHWARTZ, W. R. **Análise de imagens digitais : princípios, algoritmos e aplicações.** Thomson, 2008. ISBN 9788522105953. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=cat07269a&AN=utfpr.261648&lang=pt-br&site=eds-live&scope=site>.

PETROU, M.; PETROU, C. **Image Processing: The Fundamentals**. 3. ed. [S.l.]: Wiley, 2010. ISBN 9780470745861.

PINTO, C. H. V.; Da Silva, B. C. G.; FREIRE, P. G. d. L.; BERNARDES, D.; TAVARES, J. C.; FERRARI, R. J. Deconvolução Cega Aplicada à Correção de Artefatos de Movimento em Imagens de Vídeo de Microscopia Intravital para Detecção Automática de Leucócitos. **Revista de Informática Teórica e Aplicada**, v. 22, n. 1, p. 52, 2015. ISSN 01034308.

QUEIROZ, J. R. D. E.; GOMES, H. M. Introdução ao Processamento Digital de Imagens. **Rita**, v. 8, n. 1, p. 1–31, 2001. ISSN 85-240-0762-1. Disponível em: <http://files.multimedia-unicv.webnode.com/200000006-16be917b69/Rita-Tutorial-PDI.pdf>.

RAMAKRISHNAN, S.; PACHORI, S.; GANGOPADHYAY, A.; RAMAN, S. Deep Generative Filter for Motion Deblurring. **Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017**, v. 2018-Janua, p. 2993–3000, 2018.

SAHU, S.; LENKA, M. K.; SA, P. K. Blind Deblurring using Deep Learning: A Survey. n. 1, 2019. Disponível em: <http://arxiv.org/abs/1907.10128>.

SOE, A. K.; ZHANG, X. A simple PSF parameters estimation method for the de-blurring of linear motion blurred images using wiener filter in OpenCV. **2012 International Conference on Systems and Informatics, ICSAI 2012**, n. Icsai, p. 1855–1860, 2012.

SOLOMON, C. **Fundamentos de processamento digital de imagens : uma abordagem prática com exemplos em matlab.** LTC, 2013. ISBN 978-85-216-2617-6. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsmib&AN=edsmib.000005578&lang=pt-br&site=eds-live&scope=site>.

SOUZA, P. E. U. Restauração de imagens CCD/CBERS-2 pelo método de Richardson-Lucy modificado. **Simposio Brasileiro de Sensioriamento Remoto**, 2005.

SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I.; SALAKHUTDINOV, R. Dropout: A simple way to prevent neural networks from overfitting. **Journal of Machine Learning Research**, v. 15, n. 56, p. 1929–1958, 2014. Disponível em: <http://jmlr.org/papers/v15/srivastava14a.html>.

TAO, S.; DONG, W.; FENG, H.; XU, Z.; LI, Q. Non-blind image deconvolution using natural image gradient prior. **Optik - International Journal for Light and Electron Optics**, v. 124, p. 6599–6605, 12 2013.

TIWARI, S.; SHUKLA, V.; SINGH, A.; BIRADAR, S. Review of motion blur estimation techniques. **Journal of Image and Graphics**, v. 1, n. 4, p. 176–184, 2013.

VARGAS, A. C. G.; PAES, A.; VASCONCELOS, C. N. Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres. **Proceedings of the XXIX Conference on Graphics, Patterns and Images**, p. 1–4, 2016. Disponível em: <http://gibis.unifesp.br/sibgrapi16/eproceedings/wuw/7.pdf>.

APÊNDICE A – RESULTADOS

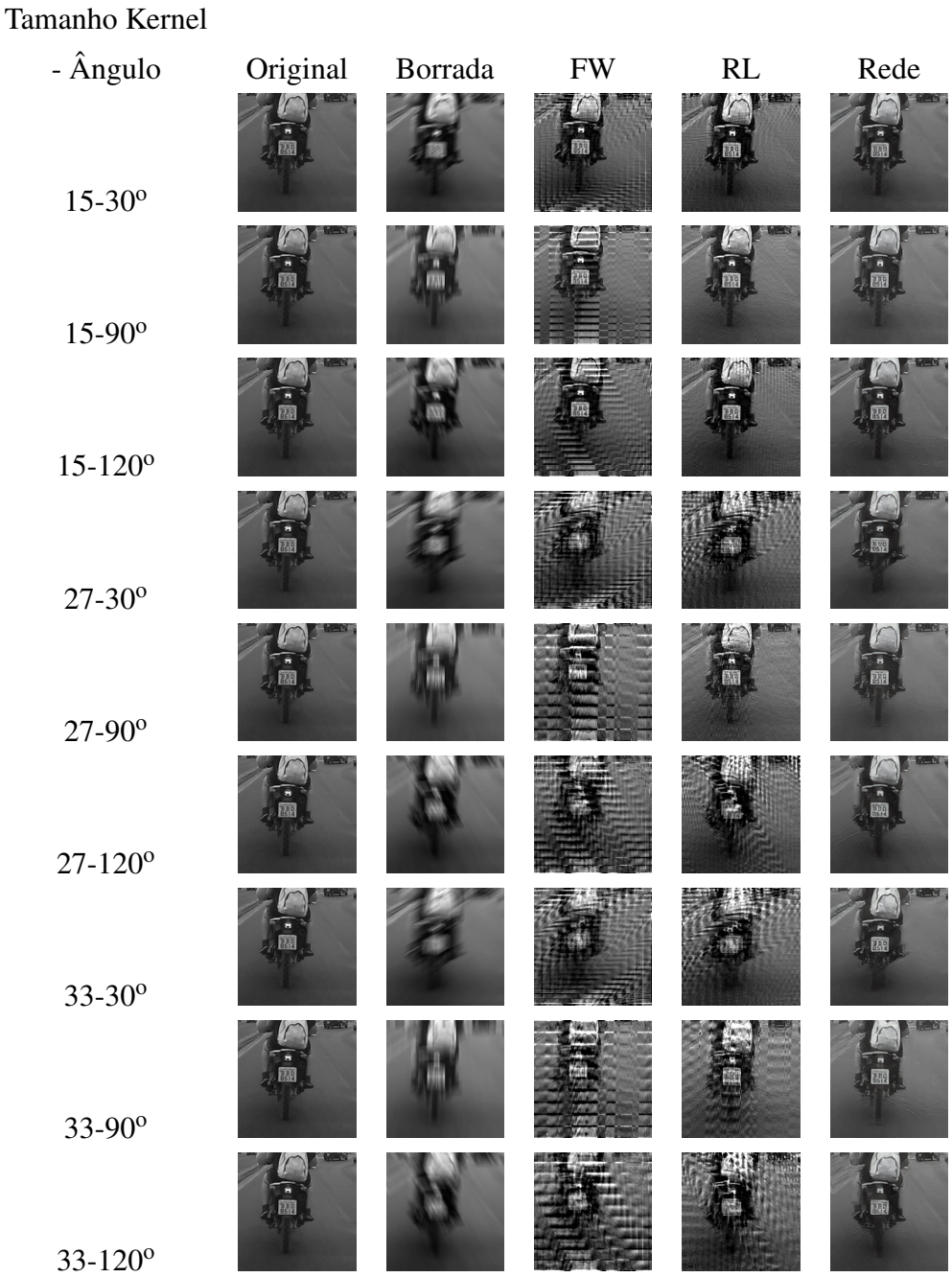


Figura 28 – Restaurações em cada parâmetro

Fonte: Autoria Própria