UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

ANDRÉ LUIZ MARASCA

# ANÁLISE DE TEXTURA UTILIZANDO DESCRITORES FRACTAIS ESTIMADOS PELA INTERFERÊNCIA MÚTUA DE GRUPOS DE PÍXEIS COLORIDOS

DISSERTAÇÃO

PATO BRANCO
2019

ANDRÉ LUIZ MARASCA

# ANÁLISE DE TEXTURA UTILIZANDO DESCRITORES FRACTAIS ESTIMADOS PELA INTERFERÊNCIA MÚTUA DE GRUPOS DE PÍXEIS COLORIDOS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná, como requisito parcial para obtenção do grau de "Mestre em Engenharia Elétrica" – Área de Concentração: Análise de Sistemas Dinâmicos.

Orientador: Dr. Dalcimar Casanova
Coorientador: Dr. Marcelo Teixeira

PATO BRANCO
2019

M311a Marasca, André Luiz.
Análise de textura utilizando descritores fractais estimados pela interferência mútua de grupos de píxeis coloridos / André Luiz Marasca. -- 2019.
71 f. : il. ; 30 cm

Orientador: Prof. Dr. Dalcimar Casanova
Coorientador: Prof. Dr. Marcelo Teixeira
Dissertação (Mestrado) - Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Engenharia Elétrica. Pato Branco, PR, 2019.
Bibliografia: f. 67 - 71.

1. Automação. 2. Textura - Análise. 3. Fractais. I. Casanova, Dalcimar, orient. II. Teixeira, Marcelo, coorient. III. Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Engenharia Elétrica. IV. Título.

CDD 22. ed. 621.3

Ficha Catalográfica elaborada por
Suélem Belmudes Cardoso CRB9/1630
Biblioteca da UTFPR Campus Pato Branco

Ministério da Educação
**Universidade Tecnológica Federal do Paraná**
Câmpus Pato Branco
Diretoria de Pesquisa e Pós-Graduação
**Programa de Pós-Graduação em Engenharia Elétrica**

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

## TERMO DE APROVAÇÃO

## Título da Dissertação n.° 071

## "Análise de Textura Utilizando Descritores Fractais Estimados pela Interferência Mútua de Grupos de Píxeis Coloridos"

por

**André Luiz Marasca**

Dissertação apresentada às doze horas e trinta minutos, do dia vinte e cinco de junho de dois mil e dezenove, como requisito parcial para obtenção do título de MESTRE EM ENGENHARIA ELÉTRICA, do Programa de Pós-Graduação em Engenharia Elétrica – Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho APROVADO.

Banca examinadora:

**Prof. Dr. Dalcimar Casanova** UTFPR/PB
(orientador)

**Prof. Dr. André Ricardo Backes**
UFU/Uberlândia/MG
(participação à distância)

**Prof. Dr. Pablo Gauterio Cavalcanti**
UTFPR/PB

**Prof. Dr. Gustavo Weber Denardin**
UTFPR/PB

Homologado por:

**Prof. Dr. Gustavo Weber Denardin**
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica - PPGEE/UTFPR

A versão devidamente assinada deste termo, encontra-se em arquivo no PPGEE -UTFPR – Câmpus Pato Branco.

## RESUMO

MARASCA, André Luiz. Análise de textura utilizando descritores fractais estimados pela interferência mútua de grupos de píxeis coloridos. 2019. 71 f. Dissertação. Programa de Pós-Graduação em Engenharia Elétrica. Pato Branco. 2019.

A análise de textura é uma técnica que vem sendo amplamente utilizada em sistemas que se utilizam de reconhecimento automático de padrões por imagens, como alternativa à capacidade cognitiva humana. Aplicações dessa técnica beneficiam diretamente áreas como teoria de controle, manufatura, biomédica, etc. A ideia central é extrair características numéricas de imagens, para assim classificá-las. O principal problema envolvendo a análise de textura é sua acurácia. O fato de não existirem até o momento abordagens 100% precisas de classificação implica que as aplicações que se utilizam da técnica impõem uma margem de erro operacional e, quando essa margem extrapola o limite tolerável a uma dada aplicação, o seu uso, como um todo, é inviabilizado. Recentemente foi proposta na literatura uma abordagem para a análise de textura baseada na teoria fractal, que potencializa os índices de acerto na classificação. Nesse contexto, o presente trabalho contribui com a área da análise de textura via teoria fractal propondo um método de extração de características de imagens de textura baseado em *rótulos*. A proposta consiste em converter uma imagem bidimensional em nuvens de pontos rotulados, que são então dilatados revelando características da imagem de textura. Os descritores de textura são obtidos a partir da relação entre o volume dos pontos dilatados, seus rótulos e os raios de dilatação. A validação do método proposto foi realizada ao estimar a sua acurácia de classificação em bases de imagens de *benchmarks* e ao compará-la com a acurácia de métodos de análise de textura da literatura. Os resultados apresentados mostram que essa combinação proporciona um descritor de textura mais robusto e preciso do que alguns dos resultados no estado da arte, expandindo assim o aspecto prático da técnica.



**Palavras-chave:** Automação. Análise de textura. Teoria fractal. Classificação supervisionada. Reconhecimento de padrões.

## ABSTRACT

MARASCA, André Luiz. Análise de textura utilizando descritores fractais estimados pela interferência mútua de grupos de píxeis coloridos. 2019. 71 f. Dissertação. Programa de Pós-Graduação em Engenharia Elétrica. Pato Branco. 2019.

Texture analysis is a technique that has been widely used in systems that use automatic image pattern recognition as an alternative to human cognitive ability. Applications of this technique directly benefit areas such as control theory, manufacturing, biomedical, etc. The main idea is to extract numerical characteristics of images, to classify them. The main problem involving texture analysis is its accuracy. The fact that there are no 100% accurate classification approaches so far implies that the applications using the technique impose an operational margin of error and, when this margin exceeds the tolerable limit for a given application, its use is unfeasible. Recently, an approach to texture analysis based on fractal theory has been proposed in the literature, which potentiates the indices of correct classification. In this context, the present work presents a contribution to the texture analysis area by fractal theory proposing a method of feature extraction of texture images based on the idea of *labels*. The proposal consists of converting a two-dimensional image into clouds of labeled points, which are then enlarged revealing characteristics of the texture image. The texture descriptors are obtained from the relation between the volume of the dilated points, their labels and the radii of dilation. The validation of the proposed method was performed by estimating its accuracy on benchmark image datasets and comparing it with the accuracy of texture analysis methods in the literature. The results presented show that the proposed approach provides a more robust and accurate texture descriptor than some of the results in the state of the art, thus expanding the practical aspect of the technique.



**Keywords:** Automation. Texture analysis. Fractal theory. Supervised classification. Pattern Recognition.

## LISTA DE SIGLAS

| | |
|---|---|
| 3D | Tridimensional |
| AT | Análise de textura |
| ATBF | Análise de textura baseada em fractais |
| BM | Bouligand-Minkowski |
| EDT | *Euclidean Distance Transform* |
| EEDT | *Exact Euclidean Distance Transform* |
| $FD_{MIC}$ | *Texture analysis using fractal descriptors estimated by the mutual interference of color channels* |
| $FD_{MIG}$ | *Texture analysis using fractal descriptors estimated by the mutual interference of groups of colored pixels* |
| FPGA | *Field Programmable Gate Array* |
| GLCM | *Gray-level Co-occurrence Matrix* |
| LBP | *Local Binary Patterns* |
| LBPRIU2 | *Local Binary Patterns Rotation Invariant Uniform* |
| LDA | *Linear Discriminant Analysis* |
| MVD | *Modified Voronoi Diagram* |
| RGB | Vermelho, verde e azul |
| STEEDT | *EEDT de Saito e Toriwaki* |
| VD | *Voronoi Diagram* |

## LISTA DE TABELAS


TABELA 1 – MATRIZ DE CONFUSÃO PARA UM PROBLEMA DE CLASSIFICAÇÃO DE DUAS CLASSES . . . . . 30
TABELA 2 – RELAÇÃO ENTRE O TAMANHO DO VETOR DE CARACTERÍSTICAS E O PARÂMETRO DE RAIO MÁXIMO DE DILATAÇÃO . . . . . 48
TABELA 3 – COMPARAÇÃO NUMÉRICA DA ACURÁCIA DO MÉTODO PROPOSTO UTILIZANDO DIFERENTES PARÂMETROS SOBRE A BASE DE IMAGENS OUTEX. . . . . 49
TABELA 4 – COMPARAÇÃO NUMÉRICA DA ACURÁCIA DO MÉTODO PROPOSTO UTILIZANDO DIFERENTES PARÂMETROS SOBRE A BASE DE IMAGENS VISTEX. . . . . 49
TABELA 5 – COMPARAÇÃO NUMÉRICA DA ACURÁCIA DO MÉTODO PROPOSTO UTILIZANDO DIFERENTES PARÂMETROS SOBRE A BASE DE IMAGENS USPTEX. . . . . 50
TABELA 6 – COMPARAÇÃO NUMÉRICA DA ACURÁCIA DO MÉTODO PROPOSTO UTILIZANDO DIFERENTES PARÂMETROS SOBRE A BASE DE IMAGENS MADEIRAS. . . . . 51
TABELA 7 – COMPARAÇÃO DA ACURÁCIA DO MÉTODO PROPOSTO UTILIZANDO DIFERENTES PARÂMETROS EM RELAÇÃO A ACURÁCIA MÉDIA DAS BASES DE IMAGENS. . . . . 51
TABELA 8 – COMPARAÇÃO NUMÉRICA DO TEMPO DE EXECUÇÃO MÉDIO (EM MILISSEGUNDOS NO COMPUTADOR INSPIRON 5767, PROCESSADOR INTEL(R) CORE(TM) I7-7500U CPU @ 2.70GHZ 2.90 GHZ, RAM INSTALADA 16,0 GB) DO MÉTODO PROPOSTO UTILIZANDO DIFERENTES VALORES PARA $R_{MAX}$ E QUANTIZAÇÃO $Z$ SOBRE A BASE DE IMAGENS OUTEX_TC_00013. . . . . 52
TABELA 9 – MÉTODOS UTILIZADOS PARA COMPARAÇÃO DE RESULTADOS DE ACURÁCIA. . . . . 54
TABELA 10 – RESUMO DE DESEMPENHO (%) DE ALGUNS MÉTODOS DE AT EM BASES DE IMAGENS DE BENCHMARK. FORMATO: ACURÁCIA $\pm$ DESVIO PADRÃO . . . . . 54

## LISTA DE FIGURAS


FIGURA 1 – VISUALIZAÇÃO DA DEDUÇÃO DA DIMENSÃO DE HAUSDORFF BESICOVITCH . . . . . 15
FIGURA 2 – DILATAÇÃO DO TRIÂNGULO DE SIERPINSKI . . . . . 16
FIGURA 3 – CURVA DE DILATAÇÃO DE UM OBJETO FRACTAL . . . . . 17
FIGURA 4 – DEFINIÇÃO GRÁFICA DA DISTÂNCIA ENTRE UM PONTO E UMA FORMA . . . . . 18
FIGURA 5 – GRADE ORTOGONAL DISCRETIZADA . . . . . 18
FIGURA 6 – VISUALIZAÇÃO DA TRANSFORMADA DE DISTÂNCIA . . . . . 19
FIGURA 7 – TRANSFORMAÇÕES EM UM ALGORITMO DE EEDT . . . . . 20
FIGURA 8 – TRANSFORMAÇÕES EM UM ALGORITMO DE EEDT . . . . . 21
FIGURA 9 – PSEUDOCÓDIGO DE UMA ALGORITMO DE EEDT . . . . . 22
FIGURA 10 – EXEMPLO DE DIAGRAMA DE VORONOI CONVENCIONAL . . . . . 23
FIGURA 11 – EXEMPLO DE DIAGRAMA DE VORONOI MODIFICADO . . . . . 23
FIGURA 12 – LBP VERSÃO SEGUNDA-ORDEM . . . . . 27
FIGURA 13 – PADRÕES DE VIZINHANÇA DO MÉTODO $LBP^{RIU2}_{P,R}$ . . . . . 28
FIGURA 14 – EXEMPLO NUMÉRICO DO PROCEDIMENTO DE CÁLCULO DE UM VD MULTIRÓTULO . . . . . 34
FIGURA 15 – DIAGRAMA DO MÉTODO $FD_{MIC}$ COM O ALGORITMO VDS MULTIRÓTULO . . . . . 40
FIGURA 16 – PROCESSO DE ROTULAÇÃO NO MÉTODO PROPOSTO . . . . . 42
FIGURA 17 – COMPARACAO DA ACÚRACIA DO MÉTODO PROPOSTO EM RELAÇÃO DIFERENTES PARÂMETROS. . . . . 49
FIGURA 18 – COMPARACAO DA ACÚRACIA DO MÉTODO PROPOSTO EM RELAÇÃO DIFERENTES PARÂMETROS. . . . . 50
FIGURA 19 – COMPARACAO DA ACÚRACIA DO MÉTODO PROPOSTO EM RELAÇÃO DIFERENTES PARÂMETROS. . . . . 50
FIGURA 20 – COMPARACAO DA ACÚRACIA DO MÉTODO PROPOSTO EM RELAÇÃO DIFERENTES PARÂMETROS. . . . . 51
FIGURA 21 – COMPARAÇÃO DA ACURÁCIA DO MÉTODO PROPOSTO UTILIZANDO DIFERENTES PARÂMETROS EM RELAÇÃO A ACURÁCIA MÉDIA DAS BASES DE IMAGENS. . . . . 52
FIGURA 22 – COMPARAÇÃO VISUAL DO TEMPO DE EXECUÇÃO MÉDIO (EM MILISSEGUNDOS) DO MÉTODO PROPOSTO UTILIZANDO DIFERENTES VALORES PARA $R_{MAX}$ E QUANTIZAÇÃO $Z$ SOBRE A BASE DE IMAGENS OUTEX. . . . . 53
FIGURA 23 – RESUMO DE DESEMPENHO DO MÉTODO PROPOSTO - OUTEX . . . 55
FIGURA 24 – RESUMO DE DESEMPENHO DO MÉTODO PROPOSTO - USPTEX . . 56

FIGURA 25 – RESUMO DE DESEMPENHO DO MÉTODO PROPOSTO - MADEIRAS 57
FIGURA 26 – RESUMO DE DESEMPENHO DO MÉTODO PROPOSTO - VISTEX . . 58

**SUMÁRIO**


**1 INTRODUÇÃO ..... 9**
1.1 OBJETIVOS ..... 11
1.1.1 Objetivo Geral ..... 11
1.1.2 Objetivos Específicos ..... 12
1.2 JUSTIFICATIVA ..... 12
**2 REFERENCIAL TEÓRICO ..... 14**
2.1 DIMENSÃO FRACTAL ..... 14
2.1.1 Dimensão Fractal de Bouligand-Minkowski ..... 15
2.2 TRANSFORMADA DE DISTÂNCIA ..... 17
2.2.1 Transformada exata de Distância com métrica euclidiana ..... 19
2.3 DIAGRAMAS DE VORONOI TRIDIMENSIONAIS ..... 22
2.4 ANÁLISE DE TEXTURA ..... 24
2.4.1 Descritores Fractais ..... 24
2.4.2 Local Binary Patterns ..... 26
2.5 RECONHECIMENTO DE PADRÕES ..... 28
2.5.1 Classificação Supervisionada ..... 28
2.5.2 Taxa de sucesso ..... 30
2.5.3 Validação cruzada ..... 30
**3 MÉTODO DE ANÁLISE DE TEXTURA PROPOSTO ..... 32**
3.1 O MÉTODO VD MULTIRÓTULO ..... 33
3.2 O MÉTODO $FD_{MIG}$ ..... 41
3.2.1 Cálculo do vetor de características ..... 44
3.3 O MÉTODO $FD_{MIG}$ GRAY ..... 45
**4 ANÁLISE DE RESULTADOS ..... 47**
4.1 BASES DE TEXTURA ..... 47
4.2 DEFINIÇÃO DOS PARÂMETROS PADRÕES PARA O MÉTODO $FD_{MIG}$ ..... 48
4.3 COMPARAÇÕES DE ACURÁCIA COM MÉTODOS CLÁSSICOS DE ANÁLISE DE TEXTURA ..... 53
4.4 COMPARAÇÃO DE ACURÁCIA COM MÉTODOS MODERNOS DE ANÁLISE DE TEXTURA ..... 59
**5 CLASSIFICAÇÃO DE SINAIS UNIDIMENSIONAIS ..... 62**
**6 CONCLUSÃO ..... 66**

# 1 INTRODUÇÃO

Nos últimos anos, avanços paralelos em áreas como a microeletrônica e a aprendizagem de máquina têm viabilizado o uso de percepção artificial como alternativa à capacidade cognitiva humana em diversas áreas do conhecimento, pelas mais diferentes razões (LIMA; PINHEIRO; SANTOS, 2016). Por exemplo, hoje em dia é possível representar aspectos da visão ou do tato humano mediante algoritmos computacionais. Isso permite tomar decisões mais ágeis, com menos riscos e, em geral, mais precisas em relação à percepção humana. Esses diferenciais se aplicam na íntegra na prática da engenharia, principalmente no contexto de ambientes insalubres, ou que requerem esforço repetitivo, ou até mesmo em aplicações para as quais as habilidades humanas podem ser superadas, como é o caso de diagnósticos médicos por imagens (CAVALCANTI; SCHARCANSKI, 2014).

De todo modo, a ideia central desse domínio do conhecimento é a de que um componente mecatrônico possa "perceber" o seu ambiente, ou algum detalhe específico e, então seja capaz de atuar em função de sua percepção. Particularmente no meio industrial, tal fato repercute diretamente nas possíveis políticas de controle e automação, impondo novas estratégias, possibilidades e desafios para a área de engenharia como um todo.

Dentre os possíveis mecanismos que permitem perceber um ambiente, para assim passar a atuar em função dele, estão os transdutores, os dispositivos de sensoriamento remoto, câmeras digitais, etc. A partir desses mecanismos, se torna possível extrair dados (por vezes complexos) que descrevem virtualmente grandezas do mundo real, como sinais elétricos, padrões geográficos, cores, texturas, etc. Esses dados são essencialmente os argumentos para que qualquer dispositivo eletrônicos seja capaz de se integrar ao mundo físico e, dessa forma, quanto mais refinados forem esses dados, mais precisão tende a dispor o dispositivo.

O foco particular desse trabalho envolve os sistemas que percebem o ambiente por meio de imagens digitais. Esses dispositivos realizam aquisição de imagens, que podem então ser interpretadas e enviadas a dispositivos eletrônicos na forma de sinais, para efetiva integração com o universo físico. A abordagem proposta nesse trabalho explora a técnica de análise de textura (AT), que tem por objetivo classificar imagens digitais por meio de seus padrões.

Aplicações computacionais envolvendo a AT se espalham por uma vasta área multidisciplinar, incluindo a medicina (CAVALCANTI; SCHARCANSKI, 2014), a botânica (BACKES; CASANOVA; BRUNO, 2009), a automação (SHAFARENKO; PETROU; KITTLER, 1997), o controle e a manufatura (BROSNAN; SUN, 2002), a cibernética, etc. A reabilitação de pacientes com paralisia de membros, por exemplo, pode-se utilizar da AT para intersectar os campos da cibernética, controle, medicina, mecânica e automação. De fato, nesse exemplo a inspeção visual pode gerar parâmetros para que o sistema de controle de um dispositivo robótico calibre o referencial de controle e, possa aplicar a força adequada para manipular um objeto.

Embora existam muitas aplicações envolvendo a análise de textura, não há uma definição formal do que de fato é textura em imagens digitais. Sabe-se apenas que ela visa mensurar

propriedades percebidas por alguns seres vivos. Esse argumento expressa a distância entre uma percepção por textura e, por exemplo, um sistema automático de controle: a percepção humana não gera diretamente sinais, tampouco os dispositivos eletroeletrônicos são capazes de incorporar a ideia de textura sem que essa passe previamente por tratamento algorítmico e, é justamente nesse contexto que se insere esse trabalho.

De um modo geral, as texturas são padrões visuais complexos compostos por subpadrões que possuem brilho característico, cor, inclinação, tamanho, etc. Assim, a textura pode ser considerada como um agrupamento de semelhanças em uma imagem (PIETIKAINEN; ROSENFELD, 1982). As propriedades visuais desses agrupamentos dão subsídios para a percepção de leveza, uniformidade, densidade, rugosidade e uma série de outras características da textura como um todo. Há quatro questões principais na análise de textura (MATERKA; STRZELECKI et al., 1998):

1. Extração: calcular as características de uma imagem digital que possam descrever numericamente suas propriedades de textura;

2. Discriminação: particionar uma imagem texturizada em regiões, cada uma correspondendo a uma textura perceptualmente homogênea (leva à segmentação de imagem);

3. Classificação: dado um número finito de classes fisicamente definidas (como tecido normal e anormal ou peças de manufatura com ou sem defeito) determinar a qual classe uma textura pertence;

4. Forma: reconstruir a geometria de uma superfície tridimensional (3D) a partir de informações de textura.

A extração de características é a primeira etapa da análise de textura de imagens e seus resultados são usados nas demais etapas. Para extrair uma característica existe uma variedade de técnicas que visam quantificar a variação na intensidade ou padrões da superfície, incluindo alguns que são imperceptíveis ao sistema visual humano. Exemplos de técnicas de extração incluem *Local Binary Patterns* (LBP) ou *Padrões Locais Binários*, *Gray-level Co-occurrence Matrix* (GLCM) ou *Matriz de Co-ocorrência*, *Gabor wavelet*, *Descritores de Fourier*, dentre outros (FLORINDO, 2013).

Todavia nenhuma dessas técnicas de extração de características consegue gerar aplicações 100 % precisas. Isso impacta diretamente nas etapas subsequentes, que fazem uso dessas características para discriminar, classificar ou compor uma forma. Na prática, isso significa que as aplicações que se utilizam da técnica de análise de textura impõem uma margem de erro operacional. Para algumas aplicações essa margem de erro pode ser aceitável enquanto que, para outras, o seu uso se torna inviável. Em um sistema de manufatura, por exemplo, uma margem de 10% de erro na identificação de peças defeituosas pode ser tolerável e contornado via retrabalho. Entretanto, a mesma margem é inaceitável para o reconhecimento de padrões em um veículo autoguiado.

Nesse contexto, se mostra de grande interesse teórico-prático a exploração de descritores de imagens que resultem na extração de características com maior poder discriminativo, para que assim melhore as taxas de classificação. Recentemente foi proposta na literatura uma abordagem para a análise de textura baseada na *teoria fractal*, que potencializa os índices de acerto na classificação (CASANOVA et al., 2016). As características extraídas a partir dessa abordagem, chamadas de descritores fractais, evidenciam maior poder discriminativo em relação a outros métodos na literatura (BACKES; CASANOVA; BRUNO, 2009; BACKES; CASANOVA; BRUNO, 2012; ALATA et al., 2011; MÄENPÄÄ; PIETIKÄINEN, 2004).

Esse trabalho contribui com a área da análise de textura baseada em fractais (ATBF), propondo um método de extração de características pautado na ideia de *rótulo*, nomeado de *Texture analysis using fractal descriptors estimated by the mutual interference of groups of colored pixels* ($\text{FD}_{\text{MIG}}$). A proposta amplia o conceito de descritores fractais já existentes e consiste em, resumidamente, converter uma imagem bidimensional em nuvens de pontos rotulados, que são então dilatados revelando características da imagem de textura. Os descritores de textura são obtidos a partir da relação entre o volume dos pontos dilatados, seus rótulos e os raios de dilatação.

Resultados mostram que essa combinação proporciona um descritor de textura mais robusto e preciso do que os resultados no estado da arte, expandindo assim o aspecto prático da abordagem. Os experimentos foram realizados sobre imagens provenientes de *benchmarks* e sobre uma base construída nesse trabalho. Inicialmente foram implementados alguns métodos da literatura e os resultados foram comparados com o método proposto, que demonstrou acurácia superior aos demais, inclusive superou o método antecessor (CASANOVA et al., 2016). Em seguida foram obtidos valores de acurácia para métodos recentes da literatura e estimou-se a acurácia do método proposto sob as mesmas condições. Os resultados revelam que o método proposto não é capaz de superar a acurácia de métodos de elevado custo computacional, contudo o seu custo computacional justifica seu uso.

## 1.1 OBJETIVOS

### 1.1.1 Objetivo Geral

Propor, implementar e validar uma abordagem para a análise de textura usando descritores fractais, com foco na ideia de rótulo. Gerar um conjunto de novos descritores que possibilitem capturar aspectos específicos de uma textura e que, assim, resultem na extração de características com maior poder discriminativo, melhorando as taxas de classificação em relação a outras abordagens na literatura.

### 1.1.2 Objetivos Específicos

- Definir e implementar a análise de textura baseada em fractais por rótulos;
- Definir um conjunto de rótulos a serem examinados em análise de textura;
- Validar e estimar a eficiência do método proposto por meio de imagens de textura provenientes de *benchmarks*;
- Comparar a eficácia do método em relação a resultados na literatura.

## 1.2 JUSTIFICATIVA

Uma das linhas de pesquisa da ATBF surgiu da aplicação do método *Bouligand-Minkowski* [1] (TRICOT, 1994) em imagens de textura em tons de cinza (BACKES; CASANOVA; BRUNO, 2009), gerando os descritores fractais. Os descritores são obtidos, a grosso modo, mapeando os píxeis da imagem de entrada para uma nuvem de pontos em um espaço 3D e dilatando sucessivamente essa nuvem para um determinado raio crescente $r$. A relação raio de dilatação *versus* volume da nuvem dilatada é chamada de curva de complexidade da textura, e a partir dela são extraídos os descritores fractais.

Todavia o método de Backes, Casanova e Bruno (2009) é limitado a imagens em tons de cinza. Uma extensão desse trabalho foi proposto por Backes, Casanova e Bruno (2012) para analisar imagens coloridas com base nos canais de cores vermelho, verde e azul (RGB) (uma imagem colorida pode e é comumente representada por meio de três canais de cores, vermelho (R), verde (G) e azul (B)). Nessa nova abordagem a nuvem de pontos é mais complexa que a nuvem em Backes, Casanova e Bruno (2009), uma vez que cada ponto pode pertencer a um determinado canal de cor, ou seja, pode possuir rótulos associados.

Embora apresentado o conceito de rótulos pela primeira vez, Backes, Casanova e Bruno (2012) utiliza-se do mesmo processo de dilatação que o formulado por Backes, Casanova e Bruno (2009) para extração dos descritores fractais. Ou seja, embora o conceito de rótulo exista no trabalho de Backes, Casanova e Bruno (2012), ele é apenas indiretamente utilizados na criação da nuvem de pontos e não efetivamente usados no cálculo posterior da dilatação.

Nesse sentido Casanova et al. (2016) modifica o trabalho de Backes, Casanova e Bruno (2012) justamente no processo de dilatação, incorporando e analisando os efeitos que cada rótulo causa sobre o volume dilatado. Os descritores fractais, nesse caso, são calculados como sendo o volume dilatado para cada rótulo e não pelo volume total como em Backes, Casanova e Bruno (2009) e Backes, Casanova e Bruno (2012). O processo, embora muito parecido com os trabalhos antecessores (criação da nuvem de pontos e dilatações sucessivas), apresenta um problema computacional grave: Para analisar uma imagem com 3 rótulos (e.g. R, G e B) são

[1] *Bouligand-Minkowski* foi inicialmente proposto para estimativa da dimensão fractal de objetos, como exposto na Seção 2.1.1

necessárias 3 dilatações e um processamento a *posteriori*, ou seja, o custo computacional cresce à medida que o número de rótulos cresce, inviabilizando a maioria das aplicações em tempo real.

É preciso observar que o trabalho *Texture analysis using fractal descriptors estimated by the mutual interference of color channels* ($FD_{MIC}$) de Casanova et al. (2016), embora inovador e bastante eficaz do ponto de vista do poder discriminativo dos descritores de textura, apresenta uma alternativa que é computacionalmente custosa e não explora outras formas de rotulação, focando-se apenas em 3 rótulos (i.e., RGB).

Os resultados de Casanova et al. (2016) são encorajadores e motivam uma investigação de uma maneira mais eficiente computacionalmente para se chegar nesses resultados e uma investigação mais profunda de outras formas de rotulação que, por conseguinte, tragam descritores fractais com maior poder discriminativo para ATBF.

# 2 REFERENCIAL TEÓRICO

Nesse capítulo são introduzidos os conceitos necessários para compreensão da ATBF rotulada, desde a extração de características até a classificação supervisionada e um método para validação dos resultados. Inicialmente é apresentada a noção de Dimensão Fractal (Seção 2.1) e, o método de aproximação da Dimensão Fractal de Bouligand-Minkowski (Subseção 2.1.1) que é utilizado em Casanova et al. (2016) como ferramenta para o cálculo do vetor de características. Esse método utiliza o operador matemático Transformada Exata de Distância com métrica euclidiana (Seção 2.2). Em $FD_{MIC}$ (CASANOVA et al., 2016) (Seção 2.4.1) foi proposta a ATBF rotulada, que utiliza o conceito de Diagramas de Voronoi (Seção 2.3) para rotulação e segmentação do espaço Bouligand-Minkowski.

Com os conceitos acima já é possível obter o vetor de características de Casanova et al. (2016), que é utilizado para a etapa de classificação de textura, que por sua vez é realizada mediante classificação supervisionada (Seção 2.5.1). Por fim, a validação dos resultados se dá por meio de uma técnica de Validação Cruzada (Seção 2.5.3) e, a taxa de sucesso é obtida como descrito na Seção 2.5.2.

## 2.1 DIMENSÃO FRACTAL

Segundo Mandelbrot e Pignoni (1983), a geometria euclidiana não é suficiente para descrever a irregularidade ou fragmentação de fenômenos do mundo físico, como montanhas e nuvens. Uma compreensão adequada da irregularidade ou fragmentação não pode ser satisfeita com a ideia de dimensão inteira (dimensão topológica). Nesse contexto, surge a ideia de dimensão fractal. Para Mandelbrot e Pignoni (1983) "um fractal é, por definição, um conjunto para o qual a dimensão Hausdorff-Besicovitch excede estritamente a dimensão topológica".

A dimensão de Hausdorff Besicovitch é deduzida do conceito de que, uma reta de comprimento $L$ pode ser dividida em $n$ partes de comprimento $H = L/n$. Para um quadrado de arestas de comprimento $L$, dividindo elas em $n$ partes iguais, são obtidos $n^2$ quadrados de área $H = (L/n)^2$. Para um cubo de arestas de comprimento $L$, dividindo elas em $n$ partes iguais, serão obtidos $n^3$ cubos de volume $H = (L/n)^3$ (ver Figura 1). Generalizando, para um hipercubo na dimensão $d$ com arestas de comprimento $L$, dividindo elas em $n$ partes iguais, são obtidos $n^d$ hipercubos com ocupação espacial $H = (L/n)^d$. Logo a dimensão $d$ de Hausdorff Besicovitch é dada por,

$$d = \frac{\ln(H)}{\ln\left(\frac{L}{n}\right)},$$

em que $L$ é o comprimento da aresta, $n$ é o número de vezes que a aresta pode ser dividida para determinado fractal em uma iteração $i$ e, $H$ é o comprimento do segmento na iteração $i$, para $i \in \mathbb{N}$.

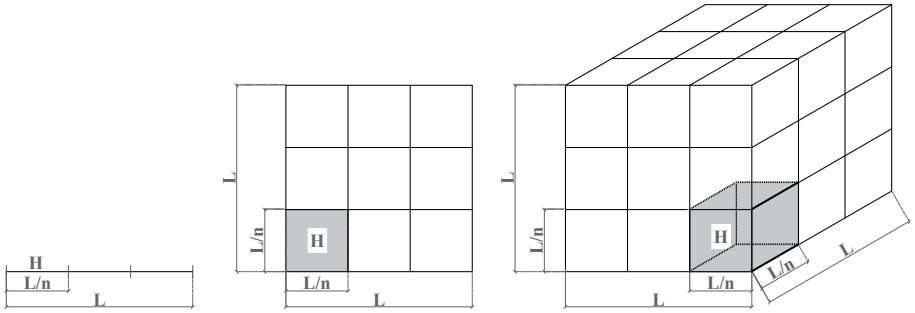



**Figura 1 – Visualização da dedução da Dimensão de Hausdorff Besicovitch. Exemplo com $n = 3$**

Um objeto que apresenta Dimensão Fractal, auto semelhança em escala e, complexidade infinita, pode ser caracterizado como um fractal puramente matemático (BACKES, 2006). Porém, objetos do mundo real estão limitados as leis da física e, dessa forma, não apresentando complexidade infinita (CARLIN, 2000), impedindo o uso da formulação acima para o cálculo de sua dimensão fractal. Entretanto, esses objetos podem ter suas dimensões fractais estimadas mediante métodos de aproximação como *box-counting*, *mass-radius* (COSTA; JR, 2009) e Bouligand-Minkowski (TRICOT, 1994). O último é utilizado nesse trabalho e descrito na subseção 2.1.1.

### 2.1.1 Dimensão Fractal de Bouligand-Minkowski

O método de Bouligand-Minkowski para estimativa da dimensão fractal é o mais sensível a mudanças na estrutura da forma analisada (BACKES, 2006) e um dos mais consistentes e apurados para o cálculo da dimensão fractal (TRICOT, 1994).

Dada uma nuvem de pontos [1] $S \subset \mathbb{R}^3$, todos os pontos $s \in S$ são dilatados sob uma esfera de raio $r \in \mathbb{R}$, gerando a nuvem de pontos dilatados $S_D$. O volume $Vo(r)$ da estrutura dilatada é calculado por:

$$Vo\,(r) = \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{k=1}^{L} X_{Ar}\,(i,j,k), \tag{1}$$

em que, $M$, $N$ e $L$ são os limites superiores para as coordenadas dos pontos $s_D \in S_D$ e $X_{Ar}$ é uma função:

$$X_{Ar}(i,j,k) = \begin{cases} 1 & \text{se} \quad (i,j,k) \in A_r; \\ 0 & \text{caso contrário.} \end{cases} \tag{2}$$

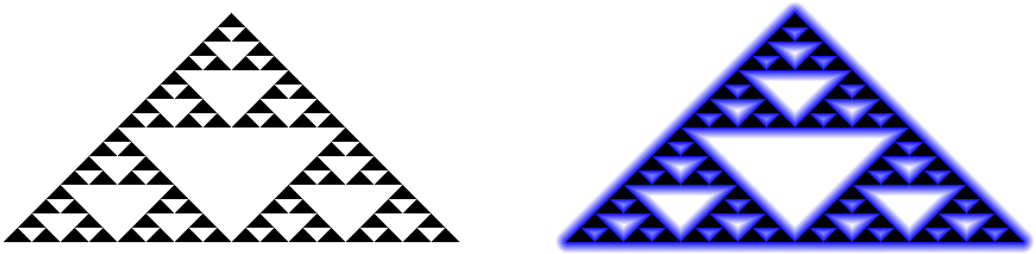
Na equação (2), $A_r \subseteq \mathbb{R}^3$ é um conjunto que inclui pontos dentro da região de influência da nuvem de pontos dilatados $S_D$, o que pode ser definido conforme a equação (3):

[1] Denota-se $\mathbb{R}^n$ o conjunto dos números reais que representam um objeto arbitrário em $n \in \mathbb{N}$ Dimensões. Para maior clareza, assumimos que um elemento $r \in \mathbb{R}^n$ tem a forma $(x_1, x_2, \cdots, x_n)$, em que cada $x_i,\ i = 1, \cdots, n$ identifica a posição em sua respectiva dimensão $i$.

$$A_r = \{p \in \mathbb{R}^3 \mid \|p - s_D\| \leq r\}. \tag{3}$$

Na equação (3), $A_r$ define a região de influência para um dado ponto $s_D \in S_D$ em relação a um delimitador de raio máximo $r$. Essa região é calculada por meio da distância euclidiana entre os pontos $p$ e $s_D$, representada pelo comprimento do vetor $p - s$, que é calculado pelo operador *norma* $L_2$ ($\|$) (ELAD, 2012).

O exemplo mostrado na Figura 2 é usado a seguir para ilustrar o efeito da equação (1) sobre um conjunto de pontos. Nesse exemplo, píxeis na Figura 2 (a) são vistos como pontos $s \in S$. Sem perder a generalidade, assume-se que $S \subset \mathbb{R}^2$ de modo que o exemplo seja mais claramente apresentado em 2 dimensões. Então, usando um algoritmo de dilatação em $S$, obtém-se a região de influência $A$, cujo efeito no exemplo é mostrado na Figura 2 (b).



(a) Triângulo de Sierpinski (original)

(b) Triângulo de Sierpinski e sua área de influência (após dilatação com $r = 7$ píxels)

**Figura 2 – Dimensão fractal ideal do Triângulo de Sierpinski,** $DF_{HB} = ln(3)/ln(2)$

Após se obter a nuvem de pontos dilatados, a dimensão fractal de Bouligand-Minkowski (BM) em $\mathbb{R}^3$, denotada por $DF_{BM}$, pode ser estimada como segue:

$$DF_{BM} = 3 - \lim_{r \to 0} \frac{\ln(Vo(r))}{\ln(r)}. \tag{4}$$

Na prática, o valor $DF_{BM}$ é obtido ao traçar a relação $ln(Vo(r)) \times ln(r)$ e a reta $ln(Vo) = \alpha \cdot ln(R) + \beta$ (Figura 3), em que $Vo$ é o vetor com os valores de volume, $R$ é o vetor com os valores de raio, $\alpha$ e $\beta$ são obtidos pelo *Método dos mínimos quadrados* (*Linear Least Squares*) (GOLAN, 2012). A dimensão fractal $DF_{BM}$ é aproximada usando o coeficiente, $\alpha$, dessa reta. Por exemplo, se $S \subset \mathbb{R}^2$, então $DF_{BM} = 2 - \alpha$; ou ainda, se $S \subset \mathbb{R}^3$, então $DF_{BM} = 3 - \alpha$. O Algoritmo 1 expõem o procedimento para o cálculo da dimensão fractal Bouligand-Minkowski $DF_{BM}$.

Para o exemplo, depois de se obter a região de influência (Figura 2(b) para a Figura 2(a)), foram calculados os valores de $\alpha$ e $\beta$, conforme ilustrado na Figura 3. A dimensão fractal resultante (por BM) é dada por $DF_{BM} = 2 - 0,41 = 1,59$. Esse resultado é próximo à dimensão fractal calculada usando o método analítico de Hausdorff Besicovitch (MANDELBROT; PIGNONI, 1983), o que levaria a $DF_{HB} = \ln(3) / \ln(2) \approx 1,58$.

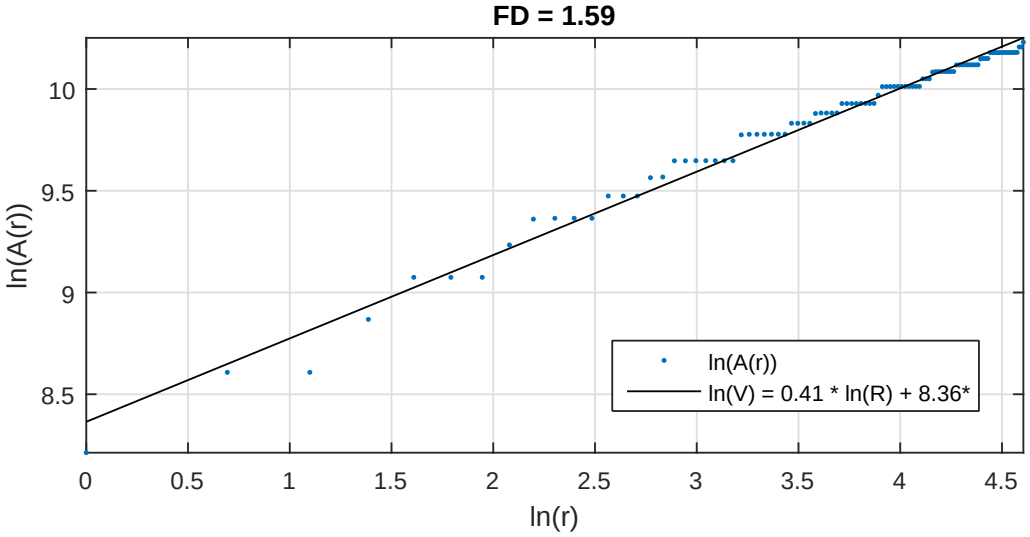



**Figura 3 – Curva** $\ln(Vo(r)) \times \ln(r)$ **obtido por dilatar o objeto da Figura 2, juntamente com a reta de regressão linear.** $DF_{BM} = 2 - 0.41 = 1.59$

**Algoritmo 1:** Cálculo da dimensão fractal de Bouligand-Minkowski

**Entrada:** Objeto a ser analisado $O$; Raio máximo de dilatação $r_{max}$;
**Saída:** Dimensão fractal Bouligand-Minkowski $DF_{BM}$;

1 $S \leftarrow$ Converter objeto em nuvem de pontos$(O)$;
2 $S_D \leftarrow EEDT(S)$;
3 $[Vo, R] \leftarrow$ Calcular Volume$(S_D, r_{max})$;
4 $[\alpha, \beta] \leftarrow$ Calcular método dos mínimos quadrados$(Vo, R)$;
5 $DF_{BM} \leftarrow 3 - \alpha$

Ao usar o método BM para estimar a dimensão fractal para um único ponto (nuvem de pontos com cardinalidade unitária), obtém-se o maior grau de variação no volume dilatado. Isso acontece pois não há outro ponto que influencie o espaço de dilatação, logo obtém-se a reta com o maior coeficiente angular. Em contraste, nos casos em que há pontos próximos uns dos outros, valores menores serão obtidos para o coeficiente angular quando comparado ao ponto isolado. Isso é esperado, uma vez que existe influência mútua no processo de dilatação.

Em geral, o processo de dilatação, como mostrado na Equação (3), é computacionalmente caro. Um meio para realizar a dilatação com menor custo computacional é utilizando o operador *Transformada Exata de Distância com métrica euclidiana* ou *Exact Euclidean Distance Transform* (EEDT) (FABBRI et al., 2008) em um espaço discretizado (ver Seção 2.2). Nesse trabalho, utilizou-se o algoritmo de EEDT proposto por Saito e Toriwaki (1994).

## 2.2 TRANSFORMADA DE DISTÂNCIA

A transformada de distância ou *Euclidean Distance Transform* (EDT) é um operador geométrico fundamental com grande aplicabilidade em visão computacional. Exemplos in-

cluem análise de formas, *skeletonization* (COSTA; JR, 2009), detecção e rastreamento de pista (RUYI et al., 2011), etc. Porém, o cálculo de uma transformada de distância com métrica euclidiana ou EDT é relativamente custoso, pois, além do tamanho da imagem, o seu conteúdo também influencia no desempenho desse algoritmo.

O operador EDT necessita de uma definição para a distância entre uma forma e um ponto qualquer. Seja $S$ uma forma binária bidimensional (conjunto de pontos conectados), seja $P \notin S$ um ponto qualquer, a distância $d(P, S)$ entre o ponto $P$ e a forma $S$ é definida como a menor distância entre $P$ e um ponto $s \in S$ qualquer. Várias métricas de distância podem ser consideradas, como *chessboard*, *city-block*, distância euclidiana (COSTA; JR, 2009), etc. A Figura 4 ilustra a distância $d(P, S)$ utilizando a métrica euclidiana.

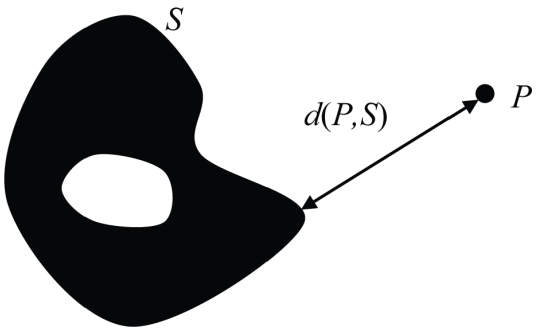



**Figura 4 – Distância $d(P, S)$ euclidiana entre o ponto $P$ e a forma $S$**
**Fonte: Costa e Jr (2009).**

No domínio contínuo existem infinitas distâncias e infinitos pontos dentro de uma área de interesse para se atribuir suas respectivas distâncias. Diferentemente, no domínio discretizado existe um número limitado de pontos e distâncias. Por exemplo, uma imagem de $3 \times 3$ permite apenas seis distâncias diferentes (contando com a distância 0) (COSTA; JR, 2009), como ilustrado na Figura 5.

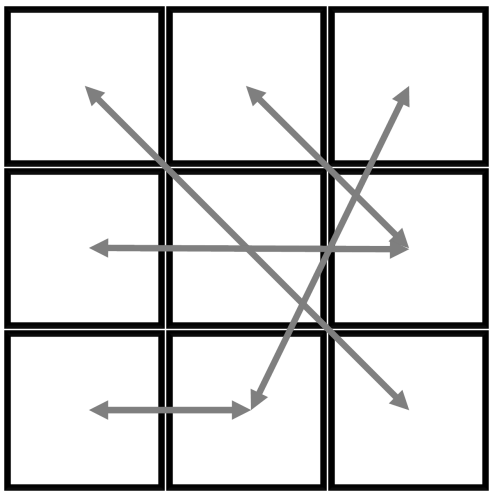
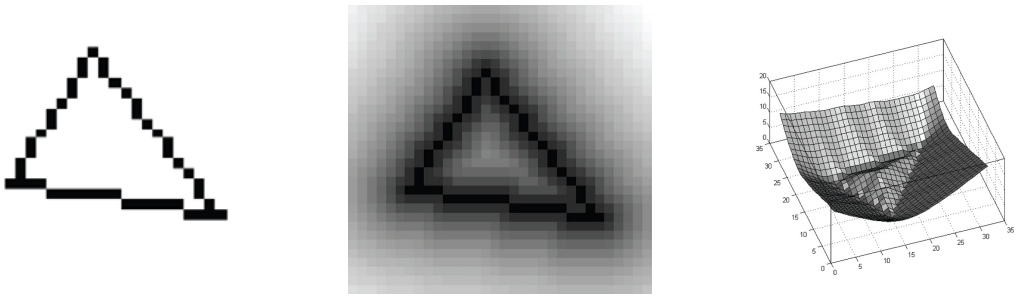


**Figura 5 – As possíveis distâncias em uma grade ortogonal discretizada $3 \times 3$**

O cálculo da EDT em uma malha ortogonal mediante algoritmos força bruta apresenta complexidade polinomial de ordem elevada (complexidade assintótica $O(n^4)$ no pior caso, $\Omega(n^2)$ no melhor caso, $\Theta(n^3)$ no caso médio, sendo $n = \sqrt{N}$, em que $N$ é o número de píxeis da imagem de entrada (FABBRI et al., 2008)), implicando em uma série de dificuldades práticas que motivaram o surgimento de muitos algoritmos simplificados (que utilizam muitas vezes de mecanismos gulosos) para o cálculo da EDT *aproximada*. Porém, os avanços recentes têm permitido o cálculo da EDT *exata* ou EEDT (FABBRI et al., 2008; CUISENAIRE; MACQ, 1999; LOTUFO; ZAMPIROLLI, 2001; SAITO; TORIWAKI, 1994). Mais detalhes sobre essas técnicas podem ser encontrados em Fabbri et al. (2008). A EEDT é praticamente invariante a rotação e/ou forma (tornando suas aplicações robustas) (FABBRI et al., 2008). Em contraste, Transformadas de Distância não exatas e/ou sem métricas euclidianas além de não possuírem tais propriedades, podem não corresponder ao resultado esperado (PAGLIERONI, 1992). A Figura 6 ilustra o resultado da aplicação da EEDT sobre uma forma $S$ qualquer.



(a) (b) (c)

**Figura 6 – Forma S qualquer (a) e sua EEDT representada em níveis de cinza (b) e representada em uma superfície tridimensional (c)**
**Fonte: Costa e Jr (2009).**

Nesse trabalho é utilizado o método de Saito e Toriwaki (1994) para realizar a EEDT. Esse método é apresentado na próxima seção.

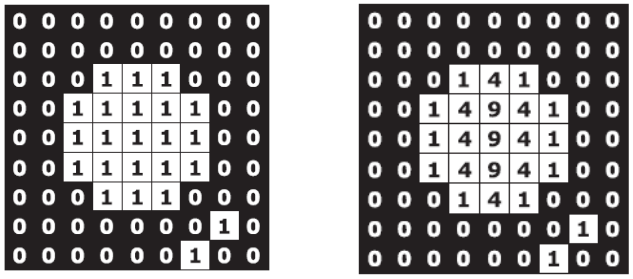
### 2.2.1 Transformada exata de Distância com métrica euclidiana

Em Saito e Toriwaki (1994) foi proposto um algoritmo (*EEDT de Saito e Toriwaki* (STEEDT)) para realizar a EEDT para uma imagem k-dimensional, utilizando k transformações unidimensionais (ou seja, uma transformação para cada eixo geométrico). Esse método garante as seguintes características citadas pelos autores:

- EEDT para o espaço n-dimensional sem grandes alterações no algoritmo;
- Aplicável com poucas modificações para imagens digitalizadas utilizando amostragem com intervalos diferentes em cada eixo de coordenada. A exemplo disso, imagens de raio X, em que os intervalos de aquisição entre as camadas representam distâncias diferentes à distância entre os intervalos dos píxeis;

- Requisitos mínimos de memória. Apenas uma matriz n-dimensional para armazenar a imagem de entrada e um único vetor para trabalho local são necessários para executar a EEDT n-dimensional;
- Algoritmos iterativos e transformações locais sempre unidimensionais;
- Tempo computacional relativamente pequeno, uma vez que o tempo do algoritmo é proporcional tanto ao número de *voxels* da imagem de entrada quanto o seu raio médio;
- Apresenta maior eficiência para computadores de uso geral, porém não necessariamente para computadores de *hardware* específico (exemplo, *Field Programmable Gate Array* (FPGA)), pois não possui máscaras para execução simultânea.

O método STEEDT (SAITO; TORIWAKI, 1994) apresenta quatro versões diferentes, desde a mais básica até a mais eficiente. Todas têm a mesma base matemática, no entanto, o algoritmo mais eficiente possui uma otimização relacionada a redução da região de busca para minimização de distâncias. STEEDT executa uma transformação para cada eixo de coordenadas. Em seu algoritmo apenas a transformada para o primeiro eixo é um caso particular, ou seja, existe um algoritmo específico para o primeiro eixo. A partir do segundo eixo as transformações são generalizadas, ou seja, pode-se utilizar o mesmo algoritmo para os eixos seguintes.




(a) Imagem de entrada

(b) Resultado da transformação 1

**Figura 7 – Ilustração da transformação 1 da versão básica do algoritmo STEEDT**
**Fonte: Fabbri et al. (2008).**

Dada uma imagem bidimensional constituída por $M$ linhas e $N$ colunas, a inicialização do algoritmo é executada marcando-se o plano de fundo (*background*) com $1's$ e pontos pertencentes a figura analisada (*foreground*) com $0's$.

Em sequência, a transformação $1$ (ver Figura 7(b)) consiste em minimizar a distância euclidiana quadrática entre *background* e *foreground* para um determinado eixo, mediante duas varreduras:

- Varredura para frente (*forward scan*), que consiste em marcar o quadrado da distância euclidiana entre *background* e o último ponto encontrado do *foreground*;
- Varredura para trás (*backward scan*), que de fato minimiza a distância entre *background* e o último ponto encontrado do *foreground*.

A saída da transformação 1 é o conjunto $G = \{g_{ij} | 1 \leq i \leq N \text{ e } 1 \leq j \leq M\}$ e, a entrada é uma imagem representada mediante um conjunto de píxeis $F = \{f_{ij}\}$, em que,

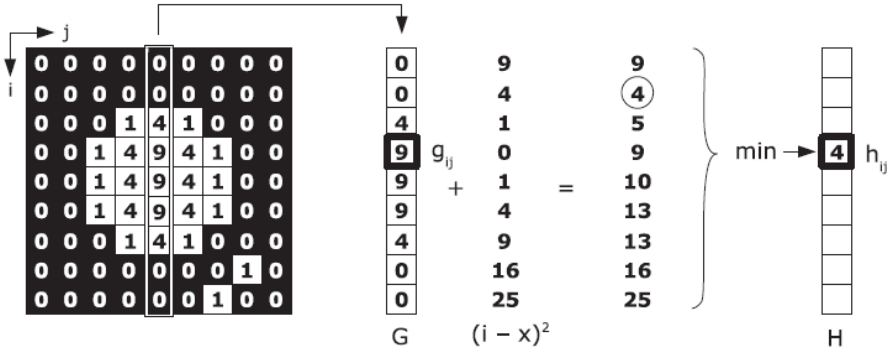
$$g_{ij} = \min \left\{ (i - x)^2; f_{ij} = 0 \text{ e } 1 \leq x \leq N \right\},$$

conforme ilustrado na Figura 7(b).

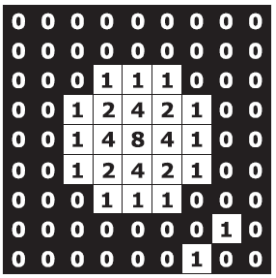
A segunda transformação (e as demais a partir desse ponto), utilizam da saída da transformação anterior como região de trabalho. No caso particular, a saída da transformação 2 é o conjunto $H = \{h_{ij}\}$ e, sua entrada é $G = \{g_{ij}\}$, em que,

$$h_{ij} = \min \left\{ g_{iy} + (j - y)^2; f_{ij} = 0 \text{ e } 1 \leq y \leq M \right\},$$

conforme ilustrado na Figura 8.




(a) Execução da trasnformação 2



(b) Resultado da transformação 2

**Figura 8 – Ilustração da transformações 2 da versão básica do algoritmo STEEDT**
**Fonte: Fabbri et al. (2008).**

A prova matemática da eficácia do método STEEDT pode ser encontrada no artigo original (SAITO; TORIWAKI, 1994). Essa prova mostra que STEEDT minimiza o quadrado da distância euclidiana.

Dada como entrada para uma EEDT bidimensional uma imagem binária com dimensões $X \times Y$ composta por $N$ píxeis, considerando $n = \sqrt{N}$, ao analisar a complexidade assintótica do método STEEDT, a *transformação 1* tem complexidade de $O(n)$, que é desconsiderada

perante a complexidade de ordem superior da *transformação 2*. Na *transformação 2*, o pior caso acontece quando $b - a$ resultar no maior valor possível, que é próximo de $n$ (ver Figura 9). Portanto, a complexidade no pior caso é $O(n^3)$ e, grosso modo, a complexidade no caso médio pode ser estimada por $O(R_m * n^2)$, em que $R_m$ é o raio médio da saída da transformação 1 (i.e., o valor médio de $b - a$).

```
//FORWARD
a = 0;
for (y = 1; y < YYY; y++) {
    if (a > 0) a--;
    if (buff[y] > buff[y - 1] + 1) {
        b = (buff[y] - buff[y - 1] - 1) / 2;
        if (y + b > (YYY - 1)) b = (YYY - 1)- y;
        for (n = a; n <= b; n++) {
            m = buff[y - 1] + (n + 1) * (n + 1);
            if (buff[y + n] <= m) break;
            if (m < A[x][y + n][z]) {
                A[x][y + n][z] = m;
            }
        }
        a = b;
    }
    else {
        a = 0;
    }
}
```

```
//BACKWARD
a = 0;
for (y = YYY - 2; y >= 0; y--) {
    if (a > 0) a--;
    if (buff[y] > buff[y + 1]) {
        b = (buff[y] - buff[y + 1] - 1) / 2;
        if (y - b < 0) b = y;
        for (n = a; n <= b; n++) {
            m = buff[y + 1] + (n + 1) * (n + 1);
            if (buff[y - n] <= m) break;
            if (m < A[x][y - n][z]) {
                A[x][y - n][z] = m;
            }
        }
        a = b;
    }
    else {
        a = 0;
    }
}
```

**Figura 9 – Pseudocódigo para os passos *forward scan* e *backward scan* do algoritmo mais eficiente para EEDT de Saito e Toriwaki (1994)**

De todo modo, uma EEDT pode ser utilizada como ferramenta para gerar um Diagrama de Voronoi (*Voronoi Diagram* (VD)) que é utilizado nesse trabalho para segmentar um espaço com várias nuvens de pontos mantendo a informação de quais regiões do espaço pertencem a uma determinada nuvem.

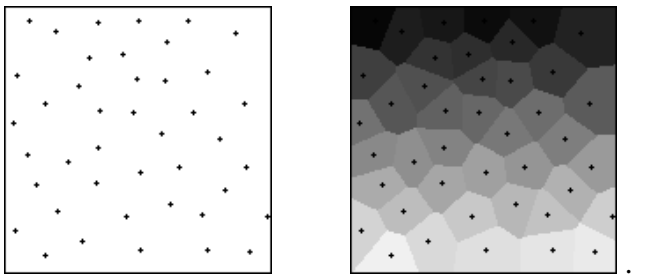
## 2.3 DIAGRAMAS DE VORONOI TRIDIMENSIONAIS

Diagramas de Voronoi (VDs) é o nome de uma técnica de segmentação de um dado espaço (COSTA; JR, 2009). Se cada um dos pontos (também chamados de sementes) de uma nuvem de pontos $S$ qualquer receberem um rótulo distinto e, uma EEDT for executada sobre $S$, então o espaço será segmentado, gerando os VDs e a nuvem de pontos será dilatada gerando o conjunto de pontos $S_D$. Dado $i, j = 1, \ldots, h$, para $i \neq j$, um VD tridimensional consiste em particionar um determinado espaço $\mathbb{R}^3$ em $h$ regiões, denotadas por $R_i$, associadas a cada ponto $s_i \in S$, em que $S_D = \{R_1 \cup R_2 \cup \ldots \cup R_h\}$, de modo que qualquer ponto dentro da região $R_i$ esteja mais próximo do ponto $s_i$ que qualquer outro ponto $s_j$ (COSTA; JR, 2009).

Define-se que, se um ponto $k$ estiver na região equidistante entre $S_i$ e $S_j$, então $k \in R_i$ e $k \in R_j$, logo $R_i \cap R_j \neq \emptyset$. O conceito de diagramas Voronoi pode ser estendido para o chamado Diagramas de Voronoi Modificados (*Modified Voronoi Diagrams* (MVDs)). Para isso, definem-se $g$ rótulos (conjuntos de regiões) $L_u = \{R_a, R_b, \ldots\}$ $u = 1, \ldots, g$, de modo que $S_D = \{L_1 \cup L_2 \cup ... \cup L_g\}$ e $L_1 \cap L_2 \cap ... \cap L_g \neq \emptyset$. Em palavras, cada ponto pertencente a nuvem de pontos dilatados pode ou não ter mais de um rótulo, ou seja, pertencer a mais de um conjunto de regiões.

Esse conceito pode ser melhor compreendido a partir das Figuras 10 e 11. Na Figura 10, cada ponto $s_i \in S$ do espaço $\mathbb{R}^2$ está associado a uma região $R_i$ do espaço $\mathbb{R}^2$, calculado como uma função do conceito de distância dos VDs. Dessa forma, cada ponto $s_i$ marca sua própria região de influência $R_i$ (Figura 10(b)).

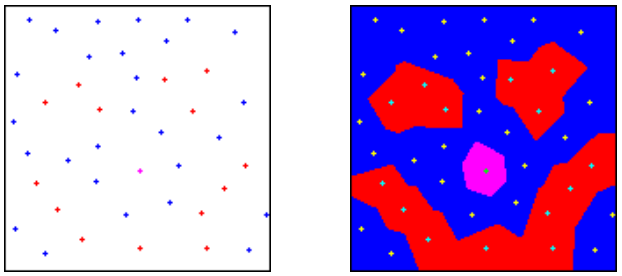
Diferentemente, na Figura 11(a) assume-se que um rótulo é composto por um conjunto de regiões. Nesse caso, tem-se 2 rótulos, $L_1$ e $L_2$ (vermelho e azul na Figura 11(a)), porém podem existir regiões pertencentes aos dois rótulos, ou seja, estar na intersecção desses, representado por magenta em 11(b).



(a) Um conjunto de pontos rotulados isolados.

(b) Diagramas de Voronoi da Figura 10(a)

**Figura 10 – Exemplo de Diagrama de Voronoi convencional. Cada região de influência foi representada por um nível de cinza distinto**



(a) Um conjunto de pontos rotulados isolados.

(b) Diagrama de Voronoi Modificado para Figura 11(a).

**Figura 11 – Exemplo de Diagramas de Voronoi Modificados. Rótulos vermelho e/ou azul, rótulo rosa é empate entre vermelho e azul. Os pontos rotulados na Figura 11(a) são representados por MVD na Figura 11(b)**

Pode-se concluir nesse ponto que, como os Diagramas de Voronoi são obtidos diretamente da EEDT e a EEDT também é usada diretamente para obter a dimensão fractal $DF_{BM}$, ao realizar operações sobre MVDs e EEDTs, logo pode-se inferir informações fractais sobre um conjunto com informações de rótulos.

## 2.4 ANÁLISE DE TEXTURA

A análise de textura é uma área da ciência que tem como objetivo extrair características de imagens de textura com elevado poder de discriminação (i.e., características que aproximam as imagens de mesma classe e as distanciam de classes diferentes) e utilizar essas características em algoritmos de aprendizagem de máquina para realizar a classificação dessas imagens.

Para muitos, o conceito de textura é algo intuitivo, mas ainda não se tem uma definição única aceita na literatura. Segundo Backes (2006), a textura pode ser caracterizada como a repetição de padrões nas variações de cores e tons de iluminação, ou quaisquer outros modelos presentes em uma determinada área, podendo haver pequenas diferenças na repetição dos padrões, como cor ou forma. Para Tamura, Mori e Yamawaki (1978), textura é caracterizada por padrões repetitivos no posicionamento de primitivas ou elementos em uma imagem. Já para Sklansky (1978), uma imagem deve possuir estatísticas locais, ou outras propriedades locais constantes, com variações suaves, para que possua uma textura constante.

### 2.4.1 Descritores Fractais

Um dos métodos mais recentes e promissores de análise de textura via descritores fractais é o ($FD_{MIC}$) proposto em Casanova et al. (2016), que analisa textura mediante técnicas de estimativa da dimensão fractal sobre a interação dos canais de cores.

O primeiro passo do método $FD_{MIC}$ é o mapeamento de cada canal de uma imagem de textura no espaço de cores RGB para uma nuvem de pontos. Para esse mapeamento, um píxel de posição (i, j) e intensidade k, é convertido em um *voxel* de coordenadas (i, j, k), rotulado de acordo com seu canal de cor de origem. Inicialmente é definida $S^c$, que é a nuvem de pontos para o canal $c$ (em que, $c \in Labels$ e $Labels = \{R, G, B\}$), representada por uma matriz tridimensional,

$$S^c(i,j,k) = \begin{cases} 0 & \text{se} \quad (i,j,k) \in CC^c; \\ 1 & \text{caso contrário.} \end{cases} \tag{5}$$

Na nuvem $S^c$, um *voxel* de coordenada $(i, j, k)$ pertence ao *foreground* se a sua coordenada existe no conjunto $CC^c$ e, pertence ao *background* caso contrário, para

$$CC^c = \{(i + r_{max}, j + r_{max}, I^c(i,j) + 1 + r_{max})\}, \tag{6}$$

para $i = 1 \ldots M$, $j = 1 \ldots N$, $I^c(i,j) = 0 \ldots L - 1$, em que os valores $M$ e $N$ são, respectivamente, os números de linhas e colunas que $I$ (imagem de textura) possui e, $L$ é o número máximo de tons de intensidade luminosa que $I$ pode assumir. Em seguida, as dimensões são atualizadas para $M = M + 2\, r_{max}$, $N = N + 2\, r_{max}$ e $L = L + 2\, r_{max}$. Logo, $S^c$ é uma matriz de dimensões $M \times N \times L$.

No segundo passo, é executado o algoritmo de EEDT de Saito e Toriwaki (1994) sobre cada uma das nuvens de pontos $S^R$, $S^G$ e $S^B$, gerando as nuvens de pontos dilatadas $S_D^R$, $S_D^G$ e $S_D^B$. Logo, $S_D^c$ é uma matriz de distância euclidiana. No passo anterior são somados os valores de $r_{max}$ a cada coordenada dos pontos e as dimensões são atualizadas para que a dilatação dos pontos via EEDT não sofram interferência pelas fronteiras durante o processo de dilatação.

O terceiro passo é gerar os Diagramas de Voronoi Modificados. Para isso, primeiramente é calculada a matriz de distâncias $S_D^T$, que pode ser obtida executando-se a EEDT de Saito e Toriwaki (1994) sobre $S^T$, que é proveniente de $CC^T = CC^R \cup CC^G \cup CC^B$. Informalmente, para obter a matriz de distâncias (nuvem de pontos dilatada) $S_D^T$ é preciso colocar todos os pontos de todas as nuvem de pontos juntos em um espaço compartilhado e aplicar o algoritmo de EEDT sobre essa nuvem. Contudo, de forma ineficiente, em Casanova et al. (2016), $S_D^T$ é simplesmente obtida tal que

$$S_D^T(i,j,k) = min\left\{S_D^R(i,j,k), S_D^G(i,j,k), {S^B}_D(i,j,k)\right\}, \tag{7}$$

para $i = 1 \ldots M,\ j = 1 \ldots N,\ k = 1 \ldots L$. Para cada *Voxel*, existe um conjunto $V(i,j,k)$ de rótulos que minimizam a equação (7). Em outras palavras, V é uma matriz tridimensional em que cada um de seus elementos é um conjunto de rótulos, tal que

$$V(i,j,k) = \left\{c \mid S_D^T(i,j,k) = S_D^c(i,j,k)\right\}, \tag{8}$$

para $i = 1 \ldots M,\ j = 1 \ldots N,\ k = 1 \ldots L$.

No quarto passo, tendo-se em mãos a nuvem de pontos dilatados para cada canal em um ambiente compartilhado $S_D^T$, dá-se início a primeira etapa do processo do cálculo da dimensão fractal de Bouligand-Minkowski (ou seja, o cálculo do volume de dilatação para cada raio $Vo^c(r)$), visando a obtenção dos descritores fractais. Porém, com uma pequena modificação:

$$Vo^c(r) = \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{L} X_{A_r^c}(i,j,k), \tag{9}$$

em que, a função que decide se um dado *voxel* será somado ao volume é:

$$X_{A_r^c}(i,j,k) = \begin{cases} \frac{1}{N_e} & \text{se} \quad (i,j,k) \in A_r^c; \\ 0 & \text{caso contrário,} \end{cases} \tag{10}$$

em que, $N_e$ é o número de elementos do conjunto $V(i,j,k)$, conforme a equação (8) (ou seja, o número rótulos que satisfazem a equação (7)).

Na equação (10), $A_r^c \subseteq \mathbb{R}^3$ é um conjunto que inclui pontos dentro da região de influência da nuvem de pontos dilatados $S_D^c$. Isso pode ser definido utilizando-se a equação (11),

$$A_r^c = \{p^c \in \mathbb{R}^3 \ \mid\ \|p^c - s^c\| \leq r \text{ e } p^c = (i,j,k) \text{ e } c \in V(i,j,k)\}, \tag{11}$$

em que, $p^c$ é um ponto tridimensional qualquer e, $s^c$ é um ponto qualquer, tal que $s^c \in CC^c$.

O quinto e último passo é a construção do vetor de características $D$, que é obtido mediante a concatenação do logaritmo dos volumes $ln\left(Vo^c\left(r\right)\right)$ de cada canal de cor, esses logaritmos são denominados descritores fractais,

$$D = ln\left(\left[Vo^R\left(1\right), \ldots, Vo^R\left(r_{max}\right), Vo^G\left(1\right), \ldots, Vo^G\left(r_{max}\right), Vo^B\left(1\right), \ldots, Vo^B\left(r_{max}\right)\right]\right), \tag{12}$$

em que, $0 \leq r \leq r_{max}$ e, $r_{max}$ é definido experimentalmente.

É importante ressaltar que $FD_{MIC}$ como foi apresentado em Casanova et al. (2016) realiza desnecessariamente o cálculo de $S_D^R$, $S_D^G$ e $S_D^B$, conforme a equação (7). Para contornar esse problema é proposto aqui um método que utilizada apenas uma execução da EEDT de Saito e Toriwaki (1994) e, de forma transparente são retornados $S_D^T$ e $V$.

De todo modo, é recomendado que a etapa de aprendizagem da ATBF utilizando $FD_{MIC}$ seja realizada juntamente com o classificador *Linear Discriminant Analysis* (Subseção 2.5.1), pois o $FD_{MIC}$ possui alta correlação entre os atributos do vetor de características.

O método de ATBF proposto nesse trabalho utiliza $FD_{MIC}$ como base e, a proposta principal para rotulação na ATBF, é utilizar conceitos presentes no método $LBP_{P,R}^{riu2}$, que é descrito na subseção 2.4.2.

### 2.4.2 Local Binary Patterns

O LBP (OJALA; PIETIKAINEN; HARWOOD, 1994) é um método de AT invariante a escala de cinza que caracteriza imagens por seu espectro de textura. A ocorrência de distribuição de unidades de textura computadas sobre uma região é chamada de espectro de textura. Mais especificamente, utiliza-se um operador bidimensional sobre uma imagem de textura extraindo padrões binários locais (LBP) de vizinhança que são utilizados para caracterizar imagens.

A versão original do LBP realiza uma análise de vizinhança sobre uma malha de píxeis com dimensões $3 \times 3$ (i.e., análise de segunda ordem, sobre os vizinhos imediatos do píxel central) assim como exposto no pseudocódigo presente no Algoritmo 2.

No método LBP cada padrão de vizinhança representa um código binário, e esse código é convertido em um número inteiro, como ilustrado pelas Figuras 12 (b) e (c) que geram a Figura 12 (d), a soma dos elementos dessa ultima é o número inteiro que representa os padrões de vizinhança de LBP (e.g. no exemplo da Figura 12 o número inteiro que representa o padrão é 169).

O vetor de características é o histograma dos números inteiros que representam todos $2^8 = 256$ os padrões de vizinhança para o operador $3 \times 3$. O LBP é variante a rotação e possui um vetor de características grande, de tamanho 256. Para contornar esse problema, surgiu o método $LBP_{P,R}^{riu2}$ ou LBP invariante a rotação de maneira uniforme (*Local Binary Patterns Rotation Invariant Uniform* (LBPRIU2)) (OJALA; PIETIKAINEN; MAENPAA, 2002).

**Algoritmo 2:** Método LBP

**Entrada:** Imagem de entrada
**Saída:** Histograma de padrões

**para** *Cada pixel da imagem de entrada* **faça**
  //Operador $LBP$ $3 \times 3$;
  **para** *Cada vizinho imediato do píxel central* **faça**
    **se** *o nível de cinza do vizinho é maior ou igual ao do píxel central* **então**
      Marcar $1$ na posição do vizinho;
    **senão**
      Marcar $0$ na posição do vizinho;
    **fim**
  **fim**
  Multiplicar esse código binário pela máscara da Figura 12(c);
  Somar os valores resultantes;
  Acrescentar a soma ao histograma;
**fim**

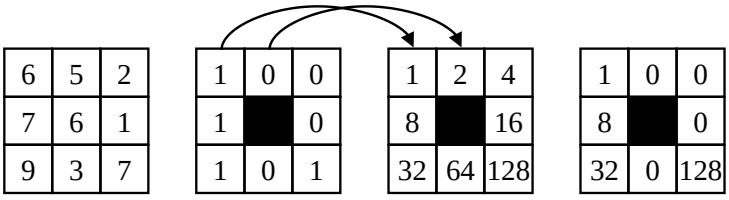



**Figura 12 – LBP versão segunda-ordem, a imagem de entrada (a) é convertida em um código binário (b) que é multiplicado por uma máscara (c) e resulta em (d).**
**Fonte: Adaptado de Ojala, Pietikainen e Harwood (1994).**

O método LBPRIU2 consiste em criar rótulos para determinados padrões de vizinhança encontrado no LBP clássico, diminuindo o número de atributos de 256 para 10 e, tornando o método invariante a rotações na imagem de textura. Para isso, é realizada uma verificação na "uniformidade" dos padrões de vizinhança, no sentido de quantas transições $0/1$ ocorrem no código binário. O método $LBP^{riu2}_{P,R}$ utiliza as equações (13),

$$LBP^{riu2}_{P,R} = \begin{cases} \sum_{p=0}^{P-1} s\left(g_p - g_c\right), & \text{se } U\left(LBP_{P,R}\right) \leq 2 \\ P+1, & \text{senão,} \end{cases} \tag{13}$$

e (14),

$$U\left(LBP_{P,R}\right) = \left|s\left(g_{P-1} - g_c\right) - s\left(g_0 - g_c\right)\right| + \sum_{p=0}^{P-1} \left|s\left(g_p - g_c\right) - s\left(g_{p-1} - g_c\right)\right|. \tag{14}$$

A equação (14) conta o número de transições $0/1$ que ocorrem no código binário para os $P$ vizinhos de um píxel, em que cada *bit* é representado por $s(g_p - g_c)$. Na equação (13),

se o número de transições é menor ou igual a 2, então a codificação $LBP^{riu2}_{P,R}$ do píxel central é obtida contando-se o número de $1's$ no código binário, senão, pelo padrão de vizinhança não ter um significado relevante, ele é tratado como uma exceção e recebe o código $P+1$. A Figura 13 ilustra os possíveis padrões de vizinhança do método $LBP^{riu2}_{P,R}$ juntamente com seus respectivos rótulos. Executando essa transformação para cada píxel da imagem de textura (escala de cinza), é gerada uma matriz $LBP^{riu2}_{P,R}$.

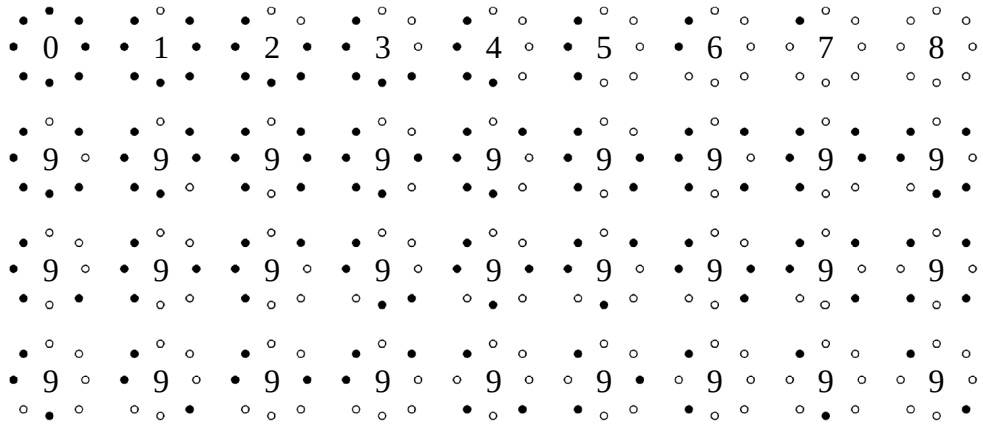



**Figura 13 – Possíveis padrões de vizinhança do método $LBP^{riu2}_{P,R}$ e seus respectivos rótulos. Adaptado de: Ojala, Pietikainen e Maenpaa (2002).**

Por fim, é calculado o histograma da matriz de codificação $LBP^{riu2}_{P,R}$ (ou seja, é realizada a contagem de quantos padrões de cada tipo aconteceram na imagem de textura), resultando em um vetor de características de tamanho $P+2$ $(0, 1, \ldots, P, P+1)$.

## 2.5 RECONHECIMENTO DE PADRÕES

Nessa seção serão abordadas as questões relevantes a reconhecimento de padrões. Serão apresentados conceitos sobre inteligência artificial relacionada a classificação supervisionada, métricas para o cálculo do desempenho de classificação, validação estatísticas dos resultados de acurácia de classificação (validação cruzada) e reconhecimento de padrões na classificação supervisionada.

### 2.5.1 Classificação Supervisionada

Alguns vetores de características obtidos por meio de métodos de AT podem possuir uma elevada correlação entre seus atributos (e.g. métodos $FD_{MIC}$). Nesse contexto, um dos métodos de classificação (aprendizagem de máquina) mais recomendados para esse cenário é o *Linear Discriminant Analysis* (LDA) (GUO; HASTIE; TIBSHIRANI, 2006), que é a maneira clássica de se realizar análise discriminante.

Sejam G populações numéricas, cada uma possuindo distribuição normal multivariada com uma matriz de covariância em comum, $\Sigma$, com dimensões $p \times p$ e vetores de média $\mu_g$

$(g = 1, \ldots, G)$ (GUO; HASTIE; TIBSHIRANI, 2006). Seja $x_i$ uma amostra aleatória de $n$ observações, com rótulo de classe desconhecida, a tarefa é descobrir de qual grupo $g$ essa amostra $x_i$ pertence.

A ideia do LDA é classificar a amostra $x_i$ para uma população $\tilde{g}$ que minimiza $\tilde{g} = (x_i - \mu_g)^T \Sigma^{-1}(x_i - \mu_g)$. Isso equivale a encontrar a população que maximiza a probabilidade de observação, uma vez que se está minimizando a distância da amostra com a média do grupo (i.e., está buscando o grupo que apresenta a média mais próxima da amostra analisada), conforme a equação (15).

$$\tilde{g} = argmin_{\dot{g}}(x_i - \mu_g)^T \Sigma^{-1}(x_i - \mu_g). \tag{15}$$

Seja $\pi_g$ a proporção da população $g$, de tal forma que $\pi_1 + \ldots + \pi_G = 1$, a probabilidade a *posteriori* de que a observação pertença a um grupo específico é maximizada, conforme a equação (16).

$$x_{g,i} \in \text{população} \left( \tilde{g} = argmin_{\dot{g}} \left[ \frac{1}{2}(x_{g,i} - \mu_g)^T \Sigma^{-1}(x_{g,i} - \mu_g) - log\pi_{\dot{g}} \right] \right). \tag{16}$$

Supondo que a matriz de covariância é comum para todas as populações, a equação (15) pode ser simplificada, gerando a equação (18),

$$x_{g,i} \in \text{população}(\tilde{g} = argmin_{\dot{g}} d_{\dot{g}}(x_{g,i})), \tag{17}$$

em que,

$$d_x(x_{g,i}) = x^T \Sigma^{-1} \mu_g - \frac{1}{2} \mu_g^T \Sigma^{-1} \mu_g + log\pi_g. \tag{18}$$

é a chamada função discriminante.

Os parâmetros $\mu_g$ e $\Sigma$ podem ser estimados a partir da amostra. Quase sempre, são usadas as estimativas de probabilidade máxima para esses parâmetros,

$$\hat{\mu}_g = \bar{x}_g = \frac{1}{n_g} \sum_{i=1}^{n_g} x_{g,i}, \tag{19}$$

$$\hat{\Sigma}_g = (X - \hat{X})(X - \hat{X})^T, \tag{20}$$

em que $X$ é uma matriz $p \times n$ com as colunas representando as observações e $\hat{X}$ é uma matriz de mesmas dimensões com cada coluna correspondente ao vetor de médias da amostra da população a que a coluna pertence. Dessa forma, pode-se reescrever a equação (18), como uma versão mais prática:

$$\hat{d}_x(x_{g,i}) = x^T \hat{\Sigma}^{-1} \bar{x}_g - \frac{1}{2} \hat{x}_g^T \hat{\Sigma}^{-1} \bar{x}_g + log\pi_g. \tag{21}$$

### 2.5.2 Taxa de sucesso

A obtenção da acurácia de classificação de um método de análise de textura (também chamada de *taxa de sucesso*) se dá mediante a matriz de confusão. Ela apresenta tanto o número de classificações corretas quanto de incorretas para dadas amostras. A matriz de confusão exibe medidas estatísticas sobre as dificuldades e facilidades que o algoritmo de classificação tem ao discriminar amostras de um determinado conjunto de dados.

Uma matriz de confusão de tamanho $n \times n$ associada à um classificador exibe a classificação prevista e real, em que $n$ é o número de classes diferentes (VISA et al., 2011). A Tabela 1 mostra uma matriz de confusão para $n = 2$, em que os elementos têm os seguintes significados:

- *a* (Verdadeiro Negativo) é o número de previsões negativas corretas;
- *b* (Falso Positivo) é o número de previsões positivas incorretas;
- *c* (Falso Negativo) é o número de previsões negativas incorretas;
- *d* (Verdadeiro Positivo) é o número de previsões positivas corretas.

**Tabela 1 – Matriz de confusão para um problema de classificação de duas classes**

| | Predito Negativo | Predito Positivo |
|---|---|---|
| Real Negativo | a | b |
| Real Positivo | c | d |

A taxa de sucesso é dada pela equação (22) e a taxa de erro de predição é dada pela equação (23).

$$Taxa\ de\ Sucesso = \frac{a+d}{a+b+c+d}; \tag{22}$$

$$Taxa\ de\ erro = \frac{b+c}{a+b+c+d}. \tag{23}$$

### 2.5.3 Validação cruzada

Estimar a acurácia preditiva de classificadores sobre um conjunto de dados de interesse permite que um classificador adequado seja escolhido para uma determinada aplicação, objetivando maximizar a sua segurança e eficácia (WOLPERT, 1992).

O desempenho de um classificador é geralmente medido em termos de erro de predição. Na maioria dos problemas práticos, o erro não pode ser calculado com exatidão e, dessa forma, deve ser estimado. Um estimador de erro de um dado classificador é uma variável aleatória e sua qualidade é medida geralmente mediante seu viés e variância (RODRIGUEZ; PEREZ; LOZANO, 2010).

A validação cruzada por *k-fold* é provavelmente a técnica de estimativa de erro de predição mais popular (RODRIGUEZ; PEREZ; LOZANO, 2010). Resumidamente, esse método consiste em dividir o conjunto total de dados de treinamento em $k$ subconjuntos mutuamente exclusivos de igual tamanho. Então o algoritmo de classificação é treinado utilizando $k-1$ subconjuntos e, o subconjunto $i$ restante é utilizado como instâncias de testes. Esse processo é repetido $k$ vezes, em cada repetição é selecionado outro subconjunto de testes $i$, de forma que $i = 1, \ldots, k$.

A estimativa de erro é o valor médio dos erros cometidos em cada etapa. Assim o estimador de erro mediante validação cruzada por *k-fold* depende de dois fatores: o conjunto de treinamento e a forma com que são criados os $k$ subconjuntos (RODRIGUEZ; PEREZ; LOZANO, 2010).

Adicionalmente, o método *Z-score* é utilizado para a padronização de cada índice do vetor de características, objetivando igualar a influência para todos os elementos desse vetor em tempo de classificação (DEVORE, 2011). Esse método consiste em utilizar a equação (24),

$$z = \frac{x - \mu}{\sigma}, \tag{24}$$

em que $x$ é a variável a ser padronizada e $\mu$ e $\sigma$ são a média e o desvio padrão respectivamente de todas as instâncias de treino em relação a característica à qual pertence a variável analisada.

# 3 MÉTODO DE ANÁLISE DE TEXTURA PROPOSTO

O método proposto, denominado $FD_{MIG}$, é baseado no $FD_{MIC}$, que extrai características fractais de imagens para usos em análise de textura. Como apresentado no capítulo 2 seção 2.4.1, o $FD_{MIC}$ divide a imagem de entrada nos três canais de cores do sistema RGB gerando um espaço tridimensional para cada canal. Em cada espaço tridimensional gerado os píxeis de seu respectivo canal são convertidos em pontos. Em seguida, o $FD_{MIC}$ aplica o operador EEDT em cada um dos três espaços tridimensionais e o resultado é utilizado para criar um VD em um espaço tridimensional compartilhado. Por fim, o $FD_{MIC}$ extrai as características fractais da imagem de entrada diretamente do VD.

Em vez de se utilizar três espaços tridimensionais distintos e posteriormente seguir com operador de união para se obter o VD, poderia ser utilizado apenas um espaço tridimensional rotulado. Existem métodos na literatura capazes de obter um VD sobre um conjunto de pontos rotulados invocando uma única vez o operador EEDT (FABBRI et al., 2008). Contudo, nenhum desses métodos é capaz de resolver os dois problemas encontrados no $FD_{MIC}$, que são: (i) necessidade de um ponto de interesse possuir mais de um rótulo; (ii) tratamento de regiões equidistantes entre pontos de interesse com diferentes rotulações. De agora em diante o problema (i) será denominado como “problema da rotulação múltipla” e o problema (ii) será denominado como “problema do empate”.

No $FD_{MIC}$, o problema da rotulação múltipla ocorre pois píxeis de canais de cores diferentes podem gerar pontos de mesma coordenada. Já o problema do empate ocorre sempre que o problema da rotulação múltipla ocorre, pois existirão pontos de interesse de diferentes rotulações ocupando o mesmo espaço. O problema do empate também pode ocorrer quando pontos de interesse com diferentes coordenadas e rotulações possuírem regiões equidistantes entre si.

Ambos os problemas são resolvidos no $FD_{MIC}$ separando os pontos de cada canal de cor em espaços tridimensionais distintos, ou seja, no caso são 3 rótulos distintos, consequentemente torna-se necessário invocar o operador EEDT separadamente para cada espaço tridimensional para que seja possível calcular o VD. Caso fosse do interesse definir um número maior de rótulos, seria gerado um número maior de espaços tridimensionais, adicionando custo computacional ao método, pois seria necessário uma EEDT para cada rótulo.

Nesse contexto, foi desenvolvido como parte desse trabalho um algoritmo de EEDT denominado como *VD multirótulo* (apresentado na seção 3.1) capaz de resolver o problema da rotulação múltipla e o problema do empate. Esse algoritmo, além de resolver ambos os problemas, também é capaz de reduzir o custo computacional de $FD_{MIC}$ uma vez que não necessita separar os pontos de diferente rotulação em espaços tridimensionais diferentes, um para cada rótulo, e dilatar cada um desses espaços separadamente para posteriormente uni-los.

Além de resolver os problemas já mencionados o *VD multirótulo* permite que o método de análise de textura proposto explore maneiras alternativas/adicionais de realizar a rotulação

em relação a $FD_{MIC}$ sem sofrer alterações no custo computacional. Uma nova abordagem de rotulação pode criar um método capaz de extrair características de imagens adicionais únicas, fazendo com que o $FD_{MIG}$ seja capaz de discriminar imagens de textura de maneira mais acurada.

## 3.1 O MÉTODO VD MULTIRÓTULO

Para obter um VD são necessárias duas entradas: (i) Uma matriz, $S$, que contém informações espaciais sobre quais pontos pertencem (*foreground*) ou não (*background*) a nuvem de pontos analisada (Ver Figura 14 (b)); (ii) Uma matriz, $V$, que armazena a rotulação dos pontos do *foreground* (Ver Figura 14 (d)).

Na matriz $S$ o *foreground* é representado por 0 e o *background* é representado por $\infty$ (Ver Figura 14 (a)). A maneira mais comum na literatura de representar rótulos na matriz $V$ é utilizando um número inteiro (i.e., cada rótulo é representado por um número inteiro). Nesse trabalho, a matriz $V$ continua sendo uma matriz de inteiros, mas os rótulos são representados pelos *bits* desses números inteiros, sendo assim, cada *bit* desse número inteiro (elemento qualquer da matriz $V$) representa a presença ou ausência de um determinado rótulo para um dado ponto do *foreground*.

Cada elemento da matriz $V$ é um número inteiro representado pelo seu código binário. Dessa forma, a fim de atribuir um rótulo para um dado elemento é necessário ativar o *bit* que corresponde a esse rótulo. Portanto, o código binário $0 \ldots 00$ significa que nenhum rótulo foi atribuído. Atribuir um rótulo $U$ significa que o bit $U$ será ativado i.e.:

$$2^U = \text{BitwiseShiftLeft}(1, U).$$

Isso significa que a ativação de dois bits quaisquer $U_1$ e $U_2$ é realizada da seguinte maneira:

$$2^{U_1} + 2^{U_2} = \text{BitwiseOR}(\text{BitwiseShiftLeft}(1, U_1), \text{BitwiseShiftLeft}(1, U_2)).$$

Para as ativações dos bits nos algoritmos são utilizadas as operações Bitwise por possuírem maior eficiência computacional (Ver Algoritmo 3 linha 5). As matrizes $S$ e $V$ são inicializadas por meio de vetores. Esses vetores armazenam as coordenadas e respectivas rotulações dos pontos do *foreground* (Algoritmo 3).

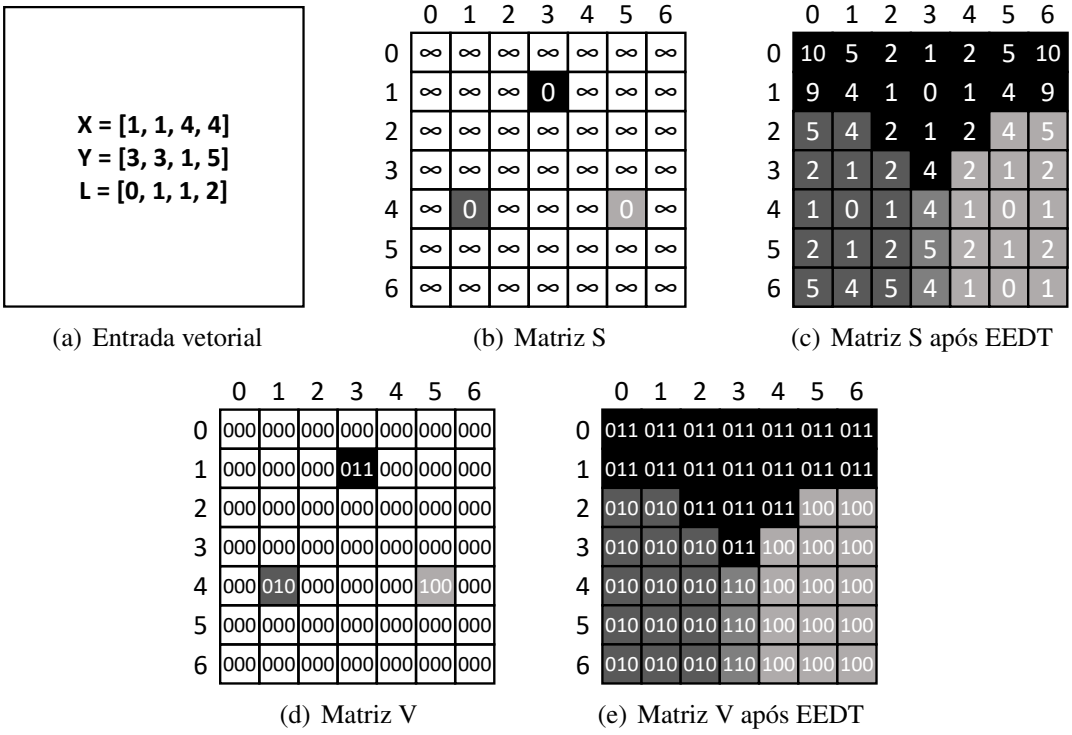
Para melhor explicar o processo de inicialização das matrizes $S$ e $V$ por meio do Algoritmo 3, é desenvolvido a seguir um exemplo numérico baseado na Figura 14. Nesse exemplo, a entrada (Figura 14 (a)) possui quatro pontos de interesse, dois deles com rótulos e coordenadas distintos e os outros dois com rótulos diferentes e coordenadas $(x, y)$ iguais.

Um dos pontos tem coordenadas $(4, 1)$ e o rótulo desejado $1$, outro ponto tem coordenadas $(4, 5)$ e o rótulo desejado $2$, e finalmente os dois outros pontos têm a mesma coordenada $(1, 3)$ e rótulos desejados $0$ e $1$. No *VD multirótulo* para obter o VD é necessário utilizar os

**Algoritmo 3:** Inicialização

**Entrada:** Vetor de coordenadas $X$; Vetor de coordenadas $Y$; Vetor de coordenadas $Z$; Vetor de rotulação $L$;
**Saída:** Matriz de distâncias $S$; Matriz de rotulação $V$;

```
1 S ← ∞;
2 V ← 0;
3 para u ← 0 ... sizeof(L) − 1 faça
4 |   S[X[u]][Y[u]][Z[u]] ← 0;
5 |   V[X[u]][Y[u]][Z[u]] ←
  |     BitOR(V[X[u]][Y[u]][Z[u]], BitShiftLeft(1, L[u]));
6 fim
```




(a) Entrada vetorial (b) Matriz S (c) Matriz S após EEDT

(d) Matriz V (e) Matriz V após EEDT

**Figura 14 – Exemplo numérico do procedimento de cálculo de um VD multirótulo (X representa as linhas e Y representa as colunas). Os vetores representados na Figura 14(a) são convertidos na matriz S (Figura 14(b)) e na matriz V (Figura 14(d)). Após executar o VD Multirótulo, a matriz S é representada pela Figura 14(c) e a matriz V é representada pela Figura 14(e)**

vetores de coordenadas $X = [1, 1, 4, 4]$, $Y = [3, 3, 1, 5]$ e, o vetor de rotulação $L = [0, 1, 1, 2]$ como entrada para o Algoritmo 3.

Por uma questão de ilustração, o exemplo assume a representação binária de apenas 3 bits. Após executar o Algoritmo 3, o primeiro ponto é representado em $V$ por $V_{(4,1)} = 010$ e, o segundo ponto é representado por $V_{(4,5)} = 100$ e os dois pontos seguintes $V_{(1,3)} = 011$. Nota-se que no primeiro ponto que o *bit* 1 foi ativado, no segundo ponto o *bit* 2 foi ativado, e nos demais pontos os *bits* 0 e 1 foram ativados, conforme o rótulos expressos no vetor $L$.

Dessa forma, os valores expressos em $L$ devem ser numéricos e devem representar os *bits* a serem ativados em $V$. Isso ocorre pois é utilizado o operador *Bitwise Shift Left* para selecionar o *bit* e o operador *Bitwise OR* para ativá-lo. Se o próprio usuário do método preferir, para evitar duplicar coordenadas com muitos rótulos, as operações *Bitwise OR* e *Bitwise Shift Left* poderiam ser realizadas anteriormente ao método de inicialização e os vetores seriam $X = [1, 4, 4]$, $Y = [3, 1, 5]$ e, o vetor de rotulação $L = [0...011, 0...010, 0...100]$.

No Algoritmo 4 é efetivamente calculada a EEDT e gerado o VD (ver Figura 14 (c) e Figura 14 (e)). Ele recebe como entrada os vetores $X$, $Y$, $Z$ e $L$ e retorna as matrizes $S$ [1] e $V$ com o VD calculado. Inicialmente, nesse algoritmo é realizada a conversão dos vetores de coordenadas e rotulação para as matrizes $S$ e $V$, conforme explicado nos parágrafos anteriores. Em seguida, são executados os algoritmos de EEDT de Saito e Toriwaki (1994) com as modificações propostas. Lembrando que Saito e Toriwaki (1994) utiliza várias transformações unidimensionais para obter a EEDT, as transformações são chamadas de *Step 1, 2, 3, ..., $N_{eixos}$*.

**Algoritmo 4:** Algoritmo para VDs multirótulo

**Entrada:** Vetor de coordenadas $X$; Vetor de coordenadas $Y$; Vetor de coordenadas $Z$; Vetor de rotulação $L$;
**Saída:** Matriz de distâncias $S$; Matriz de rotulação $V$;

```
1 [S, V] ← Inicialização(X, Y, Z, L);
2 [S, V] ← step1(S, V);
3 [S, V] ← step2(S, V);
4 [S, V] ← step3(S, V);
```

Considerando uma EEDT tridimensional, o *Step 1* é representado pelo Algoritmo 5, o *Step 2* é representado pelo Algoritmo 8, por fim, o *Step 3* é obtido modificando o eixo analisado no algoritmo 8. Por exemplo, no pseudocódigo apresentado no *Step 1* é percorrido o eixo Z, já no *Step 2* é percorrido o eixo Y, consequentemente, no *step 3* é percorrido o eixo X, a ordem não interfere no resultado final. O código em linguagem C está disponível em: https://github.com/andremarasca/FDMIG/blob/master/EDT_FRACTAL.cpp.

O *Step 1* (Algoritmo 5) é feito utilizando o *background* de $S$ como $\infty$. A modificação desse step em relação a Saito e Toriwaki (1994) é a inserção de $V$. Na etapa *Forward Scan* do *Step 1* o rótulo dos pontos pertencente ao *background* são atribuídos de acordo com o último encontrado na mesma linha. Já na etapa *Backward Scan* algumas verificações são necessárias. Se a nova distância calculada para um ponto pertencente ao *background* for menor do que atual, então o rótulo desse ponto é substituído. Senão, se a distância for igual (ocorre um empate), então é utilizado o operador *Bitwise OR* para manter a rotulação atual e unir com a nova rotulação.

[1] Nesse caso $S$ é $S_D^T$, pois ao fim do Algoritmo 4 $S$ já está dilatada

**Algoritmo 5:** Step 1

**Entrada:** Matriz de distâncias $S$; Matriz de MVD $V$;
**Saída:** Matriz de distâncias $S$; Matriz de MVD $V$;

```
1 [M, N, L] ← dimensões de (S)
2 para i ← 0 ... M − 1 faça
3     para j ← 0 ... N − 1 faça
4         //Forward Scan (Algoritmo 6)
5         //Backward Scan (Algoritmo 7)
6     fim
7 fim
```

**Algoritmo 6:** Step 1 Forward Scan

```
1  x ← −1;
2  label ← V[i][j][0];
3  para k ← 0 ... L − 1 faça
4      se S[i][j][k] = 0 então
5          x ← k;
6          label ← V[i][j][k];
7      senão se x > 0 então
8          V[i][j][k] ← label;
9          S[i][j][k] ← (k − x)²;
10     fim
11 fim
```

**Algoritmo 7:** Step 1 Backward Scan

```
1  x ← −1;
2  label ← V[i][j][L − 1];
3  para k ← L − 1 ... 0 faça
4      se S[i][j][k] = 0 então
5          x ← k;
6          label ← V[i][j][k];
7      senão se x > 0 então
8          d ← (k − x)²;
9          se d < S[i][j][k] então
10             V[i][j][k] ← label;
11             S[i][j][k] ← d;
12         senão se d = S[i][j][k] então
13             V[i][j][k] ← BitOR(V[i][j][k], label);
14         fim
15     fim
16 fim
```

O *Step 2* (Algoritmo 8) utiliza como entrada a saída do *Step 1*. Ou seja, a sua entrada é a EEDT unidimensional sobre um dos eixos de $S$ e sua respectiva matriz de rotulação $V$

**Algoritmo 8:** Step 2

```
Entrada: Matriz de distâncias S; Matriz de MVD V;
[M, N, L] ← dimensões de (S)
para i ← 0 ... M − 1 faça
    para k ← 0 ... L − 1 faça
        para j ← 0 ... N − 1 faça
            buff[j] ← S[i][j][k];
            buffV[j] ← V[i][j][k];
        fim
        //Forward Scan (Algoritmo 9)
        //Backward Scan (Algoritmo 10)
    fim
fim
```

**Algoritmo 9:** Step 2 Forward Scan

```
//Forward Scan
a ← 0;
para j ← 1 ... N − 1 faça
    se a > 0 então
        a ← a − 1;
    fim
    se buff[j] > buff[j − 1] então
        b ← (buff[j] − buff[j − 1])/2;
        se b ≥ N − j então
            b ← (N − 1) − j;
        fim
        para n ← a ... b faça
            m ← buff[j − 1] + (n + 1)²;
            se buff[j + n] < m então
                break;
            fim
            se S[i][j + n][k] > m então
                S[i][j + n][k] ← m;
                V[i][j + n][k] ← buffV[j − 1];
            senão se m = S[i][j + n][k] então
                V[i][j + n][k] ← BitOR(V[i][j + n][k], buffV[j − 1]);
            fim
        fim
        a ← b;
    senão
        a ← 0;
    fim
fim
```

atualizada. Assim como no *Step 1*, o procedimento do *Step 2* em relação a rotulação consiste em: se encontrado uma semente (ponto do *foreground*) mais próximo de um dado ponto do

**Algoritmo 10:** Step 2 Backward Scan

```
1  //Backward Scan
2  a ← 0;
3  para j ← N − 2 . . . 0 faça
4      se a > 0 então
5          a ← a − 1;
6      fim
7      se buff[j] > buff[j + 1] então
8          b ← (buff[j] − buff[j + 1])/2;
9          se j − b < 0 então
10             b ← j;
11         fim
12         para n ← a . . . b faça
13             m ← buff[j + 1] + (n + 1)²;
14             se buff[j − n] < m então
15                 break;
16             fim
17             se S[i][j − n][k] > m então
18                 S[i][j − n][k] ← m;
19                 V[i][j − n][k] ← buffV[j + 1];
20             senão se m = S[i][j − n][k] então
21                 V[i][j − n][k] ← BitOR(V[i][j − n][k], buffV[j + 1]);
22             fim
23         fim
24         a ← b;
25     senão
26         a ← 0;
27     fim
28 fim
```

*background* do que já se havia calculado, então o rótulo desse ponto deve ser substituído pelo rótulo da semente. Senão, se encontrado uma semente com a mesma distância de um dado ponto do *background* do que já se havia calculado, então o rótulo desse ponto deve ser mantido e acrescentado o rótulo da semente, utilizando o operador *Bitwise OR*.

No *VD multirótulo*, como os rótulos são representados por *bits* e não por números inteiros, se a arquitetura da máquina utilizada permitir no máximo 64 *bits* e o algoritmo for implementado de maneira que cada elemento de $V$ seja uma única variável inteira de 64 *bits*, então o número máximo de rótulos distintos permitidos é 64. Entretanto, $V$ pode ser implementado de maneira que cada um de seus elementos seja um vetor de inteiros de tamanho variável, ou seja, a memória da máquina torna-se a única limitação do número de rótulos permitidos.

Como vários *bits* podem ser ativados simultaneamente em $V$ para um dado ponto de $S$, isso resolve naturalmente o problema da rotulação múltipla. Por esse mesmo motivo, o método *VD multirótulo* é capaz de receber múltipla rotulação para uma semente e propagar esse rótulo

durante a etapa de dilatação das sementes. Além disso, o método é capaz de identificar e tratar regiões equidistantes entre duas ou mais sementes (i.e., ele é capaz de resolver o problema do empate).

O *VD multirótulo* permite que um número maior de rótulos possa ser escolhido no $FD_{MIC}$ sem afetar o custo computacional, pois independentemente do número de rótulos que serão usados, o método de análise de textura só necessita de uma execução do *VD multirótulo*. Isso significa que o custo computacional de $FD_{MIC}$ é aproximadamente dividido por 3 ao utilizar o método *VD multirótulo*. A Figura 15 ilustra o método $FD_{MIC}$ utilizando e não o método *VD multirótulo*.

O *VD multirótulo* foi desenvolvido para se tornar um operador capaz de gerar um VD a partir de um conjunto de pontos rotulados qualquer. Ele é uma continuação do trabalho de Marasca e Casanova (2016), que é um método inspirado em Saito e Toriwaki (1994) desenvolvido especificamente para reduzir o custo computacional de $FD_{MIC}$.

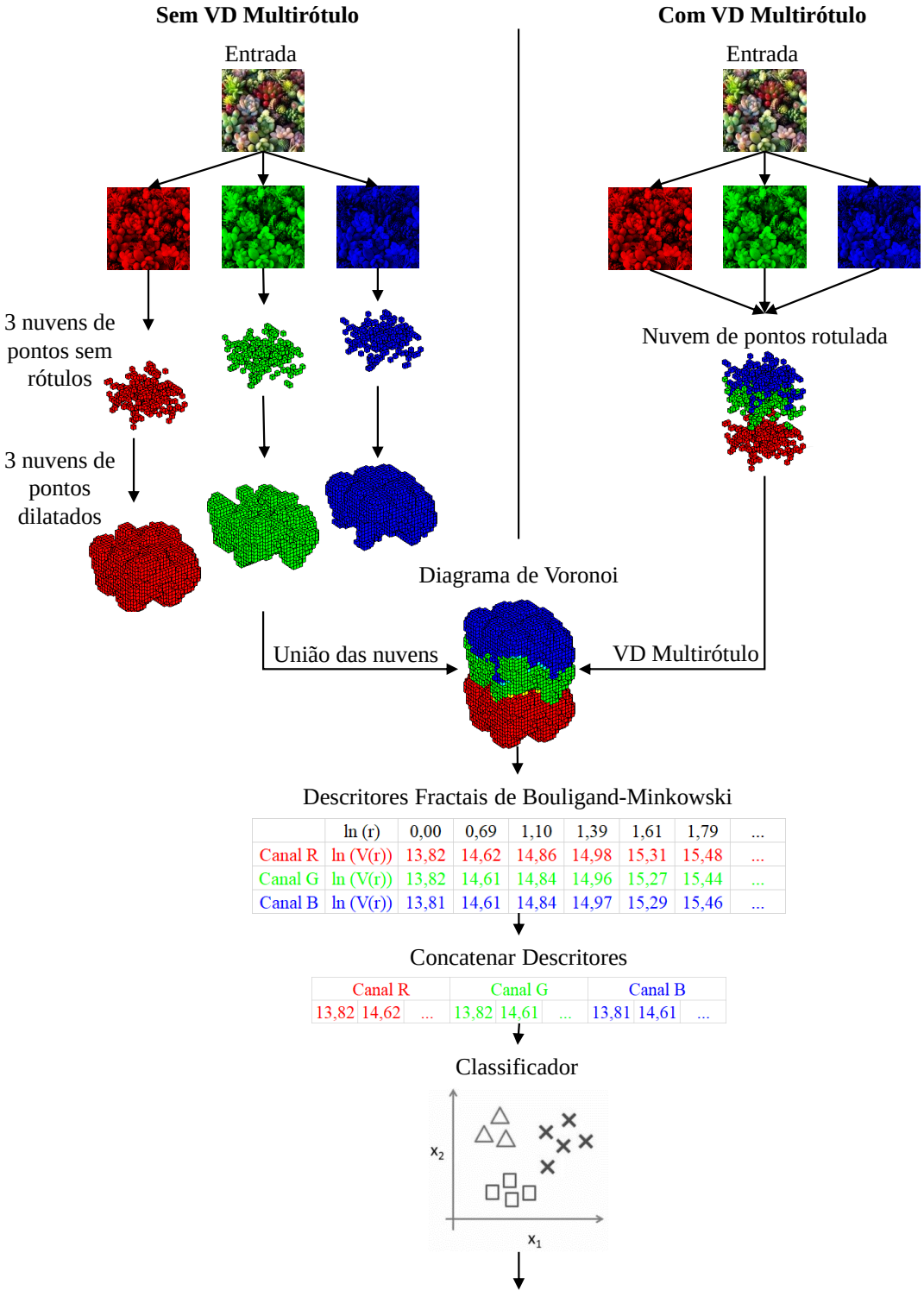



**Figura 15 – Diagrama do método $FD_{MIC}$, a esquerda da linha vertical é a maneira convencional de obter um VD nesse método, a direita dessa linha vertical é utilizando o algoritmo VDs multirótulo**

## 3.2 O MÉTODO $FD_{MIG}$

Como já mencionado anteriormente, a rotulação do método $FD_{MIG}$ proposto toma como base a implementação de $FD_{MIC}$ utilizando o método *VD multirótulo*. Para isso, $FD_{MIC}$ gera a nuvem de pontos $S^T$ convertendo todos os píxeis dos três canais de cores R, G e B em pontos em um espaço tridimensional compartilhado [2] e em seguida aplica o *VD multirótulo*.

No método $FD_{MIG}$, para o mapeamento dos píxeis da imagem de entrada nos pontos da nuvem de pontos $S^T$ [3], propõem-se utilizar, além dos canais de cores do sistema RGB, também o canal de cores *GRAY* definido como média aritmética simples dos canais de cores do sistema RGB: $GRAY = (R + G + B)/3$. Portanto, em $FD_{MIG}$, a nuvem de pontos $S^T$ é proveniente de $CC^T = CC^R \cup CC^G \cup CC^B \cup CC^{GRAY}$, respeitando sempre as equações (5) e (6).

A partir desse momento, tem-se uma maneira de atribuir coordenadas aos píxeis, o próximo passo é definir uma maneira de atribuir rótulos a eles (i.e., definir uma maneira de criar o vetor $L$). Os rótulos são fontes de informações da imagem original, que são transferidas para a nuvem de pontos, no $FD_{MIG}$ foram definidos vários rótulos, que foram agrupados em duas classes de rotulação, nomeadas de: *Cores* e *LBPRIU2* e descritas abaixo. Em conjunto, ambas as classes contém 14 rótulos distintos, como segue:

$$Labels = \{R, G, B, GRAY, LBP_0, LBP_1, \ldots, LBP_9\}\,.$$

Abaixo são descritos os procedimentos para rotular cada um dos píxeis de cada um dos canais de cores (i.e., R, G, B e GRAY) da imagem de entrada para as duas classes de rótulos.

A classe de rotulação *cores* foi proposta em $FD_{MIC}$. Os rótulos de *cores* são obtidos de maneira imediata, o rótulo de um píxel é dado de acordo com o canal de cor em que ele se encontra (R, G, B ou GRAY). Essa classe de rótulos traz a informação de cor para $FD_{MIG}$, tornando o método capaz de diferenciar padrões de coloração em texturas.

A segunda classe de rotulação (*LBPRIU2*) obtém rótulos por meio do método de AT $LBP^{riu2}_{P,R}$. Esse método pode gerar 10 rótulos diferentes para $P = 8$ vizinhos ($0, 1, \ldots, P, P+1$, ou seja, numerados do 0 até o 9), para uma imagem em níveis de cinza. Na prática, para obter os rótulos na segunda classe de rotulação, executa-se o método de AT $LBP^{riu2}_{P,R}$ individualmente para cada canal de cor. Dessa forma, cada píxel de cada canal de cor recebe individualmente um rótulo dentre os 10 valores possíveis. Essa classe de rótulos traz informações úteis sobre os padrões de vizinhança para o $FD_{MIG}$ que o $FD_{MIC}$ não possui.

Após obter a rotulação de cada um dos píxeis para cada um dos quatro canais de cores (R, G, B e GRAY) da imagem de textura, é realizada a etapa de construção dos vetores de coordenadas ($X$, $Y$ e $Z$) e de rotulação ($L$). Para isso, cada píxel de cada canal de cor é

[2] diferentemente do que é feito originalmente no $FD_{MIC}$, em que cada canal de cor gera um espaço tridimensional próprio

[3] o mapeamento dos píxeis da imagem de entrada nos pontos da nuvem de pontos $S^T$ é a conversão dos píxeis da imagem para os vetores $X$, $Y$, $Z$

mapeado na nuvem de pontos, mantendo o rótulo correspondente ao canal de cor. Nota-se que, um ponto $s$ pode ser resultado do encontro de mais de um píxel de mesma posição $i$, $j$, e intensidade $k$. Por exemplo, um píxel de cor branca (255, 255, 255) resulta em um ponto $s$ na posição i, j, k, com rotulação R, G, B e GRAY simultaneamente. O mesmo pode ocorrer para a classe de rotulação LBPRIU2. O tratamento do empate ocorre da mesma forma: o *voxel* $s$ recebe a rotulação de todos os píxeis envolvidos no conflito, que ocorre de maneira imediata no *VD multirótulo*.

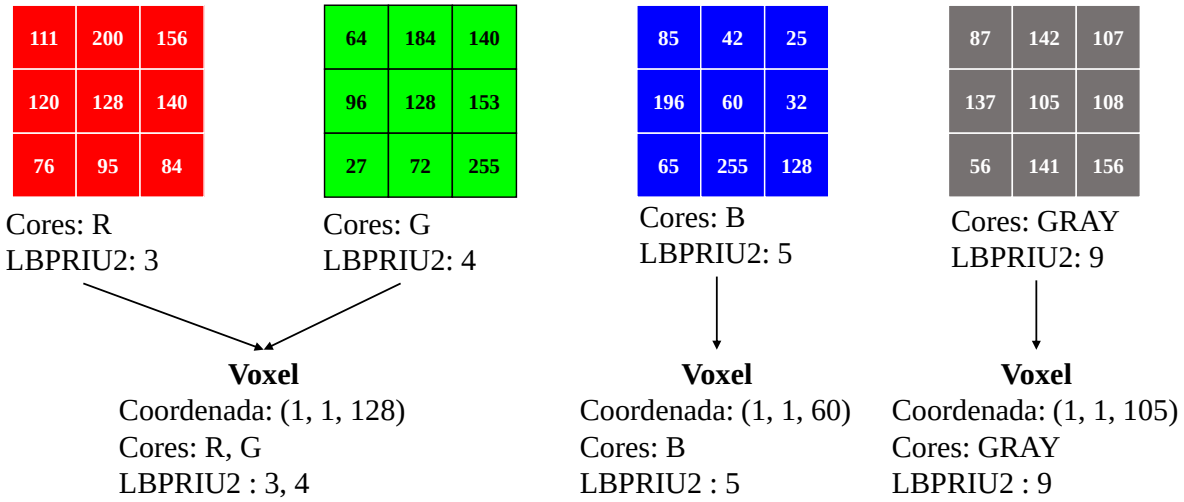



**Figura 16 – Exemplo numérico do processo de conversão de imagem digital para nuvem de pontos utilizando $FD_{MIG}$. No exemplo é ilustrado o processo apenas para o píxel central que tem a coordenada empatada nos canais de cores R e G, gerando apenas um *voxel* para eles com a união de suas rotulações. Os canais B e GRAY geram um *voxel* cada.**

A Figura 16 ilustra o processo de conversão dos píxeis de uma imagem de entrada em uma nuvem de pontos rotulados, nesse caso é ilustrado apenas a rotulação do píxel central. No exemplo, é apresentada uma imagem com 4 canais de cores, R, G, B e GRAY (ilustrados pelas cores vermelho, verde, azul e cinza respectivamente), com dimensões $3 \times 3$. O píxel central dos canais de cores R e G tem a mesma intensidade luminosa. Dessa forma, o processo de conversão da imagem digital para a nuvem de pontos gera apenas um *voxel* na mesma coordenada para ambos os píxeis centrais. Sendo assim, essas rotulações são unidas (união de conjuntos), gerando a rotulação do *voxel*. Os píxeis centrais dos canais de cores B e GRAY possuem valores distintos. Dessa forma, cada um desses píxeis gera seu próprio *voxel*.

O processo de conversão dos píxeis rotulados nos vetores de coordenadas ($X$, $Y$ e $Z$) e de rotulação ($L$) ocorre no Algoritmo 11. Sabe-se previamente que cada píxel pode possuir no mínimo 2 rótulos (um para classe cores e outro para classe LBPRIU2), em lugar de duplicar as coordenadas com rotulações diferentes, para evitar retrabalho e para maior eficiência computacional em termos de memória RAM, decidiu-se aplicar os operadores *Bitwise Shift Left* e *Bitwise OR* para unir os rótulos provenientes da diferentes classes de rotulação no Algoritmo

11, ou seja, imediatamente antes do uso do Algoritmo 3, evitando a necessidade de duplicar as coordenadas do píxel analisado para que ele possua 2 rotulações distintas.

Sendo assim, o Algoritmo 3 é poupado de serviço extra, e continua fazendo o trabalho de unir as rotulações se os *voxels* provenientes de píxeis de diferentes canais de cores colidirem no mesmo ponto. Além disso, como os bits corretos já foram ativados, não necessitando mais utilizar o operador *Bitwise Shift Left* que desloca o *bit* de um rótulo para sua posição correta. Consequentemente, no Algoritmo 3 linha 5, a operação $BitShiftLeft(1, L[u])$ é substituída simplesmente por $L[u]$.

Na linha 2 do Algoritmo 11, é calculada a classe de rotulação ***LBPRIU2***, que é a codificação do método $LBP^{riu2}_{P,R}$ para cada píxel de cada canal de cor (ou seja, não é calculado o vetor de características, apenas a codificação).

**Algoritmo 11:** Conversão da imagem de entrada em uma nuvem de pontos $S$ e uma matriz de rotulação $V$ para o método $FD_{MIG}$

**Entrada:** Imagem de entrada $I$; Valor da quantização $z$; Raio máximo de dilatação $r_{max}$;

**Saída:** Vetor de coordenadas $X$; Vetor de coordenadas $Y$; Vetor de coordenadas $Z$; Vetor de rotulação $L$;

1 $I[:][:][4] \leftarrow (I[:][:][1] + I[:][:][2] + I[:][:][3])/3$;
2 $LBP \leftarrow LBP^{riu2}_{P,R}$ para cada canal de $(I)$;
3 $[M, N] =$ Obter dimensões de $(I)$;
4 $X \leftarrow$ Vetor de zeros $(M * N * 4)$;
5 $Y \leftarrow$ Vetor de zeros $(M * N * 4)$;
6 $X \leftarrow$ Vetor de zeros $(M * N * 4)$;
7 $L \leftarrow$ Vetor de zeros $(M * N * 4)$;
8 $u \leftarrow 1$;
9 **para** $i \leftarrow 0 \ldots M - 1$ **faça**
10  **para** $j \leftarrow 0 \ldots N - 1$ **faça**
11  **para** $k \leftarrow 0 \ldots 2$ **faça**
12  $X[u] \leftarrow i + r_{max}$;
13  $Y[u] \leftarrow j + r_{max}$;
14  $Z[u] \leftarrow (z + 1)\ I[i][j][k]/256 + r_{max}$;
15  $BitsCores \leftarrow BitShift(1, k - 1)$ ;
16  $BitsLBPRIU2 \leftarrow BitShift(1, LBPRIU2[i][j][k])$ ;
17  $L[u] \leftarrow BitsCores$;
18  $L[u] \leftarrow BitOR(L[u], BitShift(BitsLBPRIU2, 4))$;
19  $u \leftarrow u + 1$;
20  **fim**
21  **fim**
22 **fim**

O Algoritmo 11 é capaz de receber uma imagem colorida e retornar os vetores de coordenadas ($X$, $Y$ e $Z$) e de rotulação ($L$).

### 3.2.1 Cálculo do vetor de características

Tendo em mãos o Algoritmo 11 que converte a imagem de entrada na nuvem de pontos rotulados, torna-se possível calcular o vetor de características do método $\text{FD}_{\text{MIG}}$. O procedimento para isso é implementado no Algoritmo 12.

No Algoritmo 12, é inicialmente realizada uma chamada para o o Algoritmo 11 que retorna os vetores de coordenadas e de rotulação. Na sequência (Linha 2), é invocado o método *VD multirótulo* (Algoritmo 4) que calcula a matriz de distâncias $S_D$ e sua respectiva matriz de rotulação $V$.

**Algoritmo 12:** Método $\text{FD}_{\text{MIG}}$ proposto.

**Entrada:** Imagem de entrada $I$; Valor da quantização $z$; Raio máximo de dilatação $r_{max}$;
**Saída:** Vetor de características $D$;
1 $[X, Y, Z, L] \leftarrow$ Algoritmo 11$(I, z, r_{max})$;
2 $[S_D, V] \leftarrow$ Algoritmo 4$(X, Y, Z, L)$;
3 $Vo \leftarrow$ Algoritmo 13$(S_D, V, r_{max})$;
4 $D \leftarrow$ Concatenar volumes$(Vo)$;

O próximo passo é o cálculo do volume (Linha 3). No método proposto a equação (10) (que é utilizada em $\text{FD}_{\text{MIC}}$) é substituída pela equação (25). Essa substituição é necessária pois nesse trabalho existem diferente classes de rótulos, sendo assim, todos os píxel tem mais de um rótulo. Portanto, não divide-se o volume de um *voxel* entre as classes de rótulos. O Algoritmo 13 implementa esse procedimento.

$$X_{A_r^c}(i,j,k) = \begin{cases} 1 & \text{se} \quad (i,j,k) \in A_r^c; \\ 0 & \quad \text{caso contrário,} \end{cases} \tag{25}$$

Em que $A_r^c$ é definido na equação (11), $X_{A_r^c}(i,j,k)$ é utilizado na equação (9) para o cálculo de $Vo^c(r)$.

Nos algoritmos 12 e 13, $Vo$ é definido como uma matriz que representa $Vo^c(r)$ para $c \in Labels$ e $r = 1, \ldots, r_{max}$, em que cada linha de $Vo$ representa um valor que $c$ pode assumir e cada coluna de $Vo$ representa um valor que $r$ pode assumir.

A função *Concatenar volumes* (definida pela equação (26), referenciada no Algoritmo 12, linha 4) converte a matriz $Vo$ no vetor de características $D$. Em palavras, para obter o vetor de características $D$, basta concatenar os valores de volume em um único vetor.

$$\textit{Concatenar volumes}(Vo) = \left(\left[Vo^R(1), \ldots, Vo^R(r_{max}), \ldots, Vo^{LBP_9}(1), \ldots, Vo^{LBP_9}(r_{max})\right]\right), \tag{26}$$

É importante ressaltar que a função *Concatenar volumes*, elimina o elemento do vetor de volume $Vo$ que corresponde à $r_{max}^2 = 0$ e, elimina os elementos do vetor de volume $Vo$ que

**Algoritmo 13:** Calcular Volume

**Entrada:** Matriz de distâncias $S_D^T$; Matriz de rotulação $V$; Raio máximo de dilatação $r_{max}$;
**Saída:** Matriz de Volumes $Vo$;

$N_{Rotulos} \leftarrow$ Calcular número de rótulos utilizados $(V)$;
$Vo \leftarrow$ Matriz de zeros$(N_{Rotulos}, r_{max}^2 + 1)$;
**para** $i \leftarrow 1 \ldots M$ **faça**
  **para** $j \leftarrow 1 \ldots N$ **faça**
    **para** $k \leftarrow 1 \ldots 3$ **faça**
      **se** $S_D^T[i][j][k] \leq r_{max}^2$ **então**
        **para** $u \leftarrow 1 \ldots N_{Rotulos}$ **faça**
          **se** $BitAND(V[i][j][k], BitShiftLeft(1, u))! = 0$ **então**
            $Vo[u][S_D^T[i][j][k] + 1] \leftarrow Vo[u][S_D^T[i][j][k] + 1] + 1;$
          **fim**
        **fim**
      **fim**
    **fim**
  **fim**
**fim**
**para** $i \leftarrow 1 \ldots N_{Rotulos}$ **faça**
  **para** $j \leftarrow 2 \ldots r_{max}^2 + 1$ **faça**
    $Vo[i][j] \leftarrow Vo[i][j-1];$
  **fim**
**fim**

correspondem aos valores de $r_{max}^2$ impossíveis (e.g. o raio $r^2 = 7$ não existe em uma malha tridimensional discretizada, assim como o raio $r^2 = 3$ não existe em uma malha bidimensional discretizada, ver Figura 5).

## 3.3 O MÉTODO FD$_{\text{MIG}}$ GRAY

Algumas aplicações não possuem imagens coloridas, apenas em escala de cinza. Nesse cenário tanto o método FD$_{\text{MIG}}$ padrão quanto o FD$_{\text{MIC}}$ não são capazes de atuar. Para contornar esse problema, é apresentado nessa seção a versão FD$_{\text{MIG}}$ Gray.

Imagens em escala de cinza não possuem canais de cores RGB, logo a nuvem de pontos $S^T$ do método FD$_{\text{MIG}}$ Gray é gerada apenas por píxeis do canal GRAY [4]. Portanto, em FD$_{\text{MIG}}$ Gray, a nuvem de pontos $S^T$ é proveniente de $CC^T = CC^{GRAY}$, respeitando sempre as equações (5) e (6).

A partir desse momento, tem-se uma maneira de atribuir coordenadas aos píxeis, o próximo passo é definir uma maneira de atribuir rótulos a eles (i.e., definir uma maneira de criar o vetor $L$). Nesse cenário, como só existem informações do canal GRAY, só é definida a classe

[4] o mapeamento dos píxeis da imagem de entrada nos pontos da nuvem de pontos $S^T$ é a conversão dos píxeis da imagem para os vetores $X$, $Y$, $Z$

de rótulos *LBPRIU2* que gera 10 rótulos distintos:

$$Labels = \{LBP_0, LBP_1, \ldots, LBP_9\}\,.$$

Para se obter os rótulos da classe *LBPRIU2* é executado o método de AT $LBP^{riu2}_{P,R}$ para o canal GRAY. Dessa forma, cada píxel recebe individualmente um rótulo dentre os 10 valores possíveis. Em seguida, é realizada a etapa de construção dos vetores de coordenadas ($X$, $Y$ e $Z$) e de rotulação ($L$). Para isso, cada píxel é mapeado na nuvem de pontos, mantendo sua rotulação.

O processo de conversão dos píxeis rotulados nos vetores de coordenadas ($X$, $Y$ e $Z$) e de rotulação ($L$) é implementado pelo Algoritmo 14.

Na linha 2 do Algoritmo 11, é calculada a classe de rotulação *LBPRIU2*, que é a codificação do método $LBP^{riu2}_{P,R}$ para cada píxel do canal GRAY (ou seja, não é calculado o vetor de características, apenas a codificação). O restante do método permanece inalterado.

**Algoritmo 14:** Conversão da imagem de entrada em níveis de cinza em uma nuvem de pontos $S$ e uma matriz de rotulação $V$ para o método $FD_{MIG}$ Gray

**Entrada:** Imagem de entrada $I$; Valor da quantização $z$; Raio máximo de dilatação $r_{max}$;
**Saída:** Vetor de coordenadas $X$; Vetor de coordenadas $Y$; Vetor de coordenadas $Z$; Vetor de rotulação $L$;

```
1  LBP ← LBP^{riu2}_{P,R}(I);
2  [M, N] = Obter dimensões de (I);
3  X ← Vetor de zeros (M * N);
4  Y ← Vetor de zeros (M * N);
5  X ← Vetor de zeros (M * N);
6  L ← Vetor de zeros (M * N);
7  u ← 1;
8  para i ← 0 ... M − 1 faça
9      para j ← 0 ... N − 1 faça
10         para k ← 0 ... 2 faça
11             X[u] ← i + r_max;
12             Y[u] ← j + r_max;
13             Z[u] ← (z + 1) I[i][j][k]/256 + r_max;
14             L[u] ← LBPRIU2[i][j][k];
15             u ← u + 1;
16         fim
17     fim
18 fim
```

# 4 ANÁLISE DE RESULTADOS

Nesse capítulo, inicialmente, são apresentadas análises de diferentes parâmetros para o método $FD_{MIG}$ ponderando acurácia e custo computacional. Em seguida, são apresentadas (Na Seção 4.3) análises e comparações da acurácia do método proposto em relação a métodos clássicos de AT utilizando bases de textura de benchmarking. Por fim, são apresentadas análises e comparações da acurácia do método proposto em relação a métodos modernos e clássicos de AT utilizando bases de textura de benchmarking desenvolvidas especificamente para métodos robustos (i.e., bases que necessitam de métodos de AT com elevado poder de generalização).

Na Seção 4.1 são apresentadas as bases de textura utilizadas para as análises de parâmetros do método proposto, que são apresentadas na Seção 4.2, bem como para as comparações do método proposto com os métodos clássicos de AT apresentadas na Seção 4.3.

## 4.1 BASES DE TEXTURA

Para algumas das análises nesse trabalho foram utilizadas as bases de imagens de *benchmarks* Vistex, Outex e Usptex e uma construída nomeada de Madeiras.

A base de imagens Vision Texture (Vistex) (TECHNOLOGY, 2002) foi desenvolvida no *Massachusetts Institute of Technology*. É uma base de dados fora do convencional, no sentido de aquisição, pois o objetivo da Vistex é fornecer imagens de textura que são representativas das condições do mundo real. Foram usadas 54 imagens originais da Vistex com resolução de $512 \times 512$, cada uma dividida em 16 sub-imagens não sobrepostas com dimensões de $128 \times 128$, resultando em 864 sub-imagens.

O conjunto de testes Outex_TC_00013 (OJALA et al., 2002) é uma base de imagens coloridas, que contém uma significativa coleção de texturas de superfície, todas capturadas utilizando os mesmos rígidos critérios de angulação e iluminação. Contém tipos variados de texturas de superfície e cenas naturais e artificiais. Foram usadas 68 imagens originais cada uma correspondente a uma classe, cada imagem dividida em 20 sub-imagens não sobrepostas com dimensões de $128 \times 128$, resultando em 1360 sub-imagens.

A base de imagens Usptex (BACKES; CASANOVA; BRUNO, 2012), foi desenvolvida na Universidade de São Paulo. As condições de aquisição, ou seja, angulação, iluminação, escala da imagem, correção de gama, fontes e primárias da câmera, são não controladas. A base foi construída com 332 fotografias com dimensões $512 \times 384$, cada uma dividida em 12 sub-imagens não sobrepostas com dimensões de $128 \times 128$, resultando em 3984 sub-imagens.

A base de imagens de textura construída chamada *Madeiras*, foi constituída por imagens coletadas de (MEIER, 2008). Ao todo foram extraídas $432$ imagens, cada imagem corresponde a uma espécie de madeira (fotografia da estrutura macroscópica da madeira). Em sequência cada uma dessas fotografias foram recortadas em sub-imagens de dimensões $200 \times 200$ sem sobreposição. Como o tamanho de cada imagem é diferente, foram selecionadas no máximo $8$

sub-imagens por classe, para que as classes se mantivessem balanceadas. Ao todo, obteve-se $3034$ sub-imagens, pois algumas imagens não são grandes o suficiente para se obter $8$ sub-imagens.

Todas as bases de imagens apresentadas nessa seção estão disponíveis no seguinte link: https://goo.gl/h1wiyZ.

## 4.2 DEFINIÇÃO DOS PARÂMETROS PADRÕES PARA O MÉTODO $FD_{MIG}$

O método $FD_{MIG}$ possui dois parâmetros que influenciam a sua acurácia e devem ser ajustados, $r_{max}$ e quantização $z$.

O parâmetro $r_{max}$ refere-se ao valor do raio máximo de dilatação dos pontos no método *VD multirótulo*. O tamanho do vetor de características de $FD_{MIG}$ está diretamente relacionado ao valor de $r_{max}$, assim como apresentado pela equação (26). A Tabela 2 apresenta a relação do número de atributos do vetor de características de $FD_{MIG}$ em função do parâmetro $r_{max}$. Esses valores vem do número de distâncias possíveis em um malha ortogonal multiplicado pelo número de rótulos do método, no caso 14, ex: Para $r_{max} = 5$ o número de distâncias distintas em uma malha ortogonal tridimensional é 22, logo número de atributos é $22 * 14 = 308$

**Tabela 2 – Relação do número de atributos do vetor de características do método proposto em função do parâmetro $r_{max}$.**

| $r_{max}$ | **3** | **4** | **5** | **6** | **7** |
|---|---|---|---|---|---|
| **Número de atributos** | 112 | 196 | 308 | 434 | 588 |

O método $FD_{MIC}$, e consequentemente $FD_{MIG}$, são variáveis ao nível de quantização $z$, da iluminação da imagem. A quantização torna-se um parâmetro, de modo que a imagem colorida $I_{256}$, com 256 níveis de intensidade luminosa, sofre uma transformação linear da seguinte forma:

$$I_z = round((z-1)\frac{I_{256}}{255}),$$

em que, a imagem $I_z$ apresenta $z$ níveis de intensidade luminosa após a transformação. O Algoritmo 11 linha 14 implementa a quantização $z$.

Segundo Casanova et al. (2016), quanto menor o valor de $z$, maior é a quantidade de informações perdidas durante a quantização $z$. Porém, quanto maior o valor de $z$, maiores são as chances de evolução volumétrica constante (isso significa que não são obtidas informações úteis com a dilatação). Além disso, quanto maior o valor de $z$, maior é a distância entre os píxeis, consequentemente, menor é a quantidade de informações relacionais obtidas durante o processo de dilatação.

Para verificar a melhor combinação dos parâmetros $r_{max}$ e quantização $z$ foram estimados os valores de acurácia de $FD_{MIG}$ para as bases de textura Outex_TC_00013, Vistex,

Usptex e Madeiras [1], apresentados nas tabelas 3, 4, 5, 6 respectivamente. As Figuras 17, 18, 19, 20 ilustram os valores das tabelas 3, 4, 5, 6 respectivamente. Por fim, as médias dos valores de acurácia das bases Outex_TC_00013, Vistex, Usptex e Madeiras são apresentados na Tabela 7 e a Figura 21 ilustra esses valores.

**Tabela 3 – Comparação numérica da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Outex. Fomato: acurácia $\pm$ desvio padrão (número de atributos)**

| | **quantização $z$** | | | | |
|---|---|---|---|---|---|
| $r_{max}$ | **64** | **96** | **128** | **192** | **256** |
| **3** | 93.68 $\pm$ 1.44 (112) | 93.53 $\pm$ 1.33 (112) | 93.90 $\pm$ 1.39 (112) | 94.71 $\pm$ 1.58 (112) | 94.41 $\pm$ 1.26 (112) |
| **4** | 94.12 $\pm$ 1.34 (196) | 95.15 $\pm$ 1.05 (196) | 93.75 $\pm$ 1.52 (196) | 94.49 $\pm$ 1.21 (196) | 93.82 $\pm$ 1.97 (196) |
| **5** | 93.97 $\pm$ 1.54 (308) | 94.85 $\pm$ 1.34 (308) | 94.12 $\pm$ 1.47 (308) | 94.56 $\pm$ 1.48 (308) | 93.97 $\pm$ 0.90 (308) |
| **6** | 93.68 $\pm$ 1.31 (434) | 94.63 $\pm$ 1.66 (434) | 93.46 $\pm$ 2.06 (434) | 94.56 $\pm$ 1.52 (434) | 93.09 $\pm$ 0.93 (434) |
| **7** | 93.01 $\pm$ 1.11 (588) | 93.53 $\pm$ 1.58 (588) | 92.57 $\pm$ 1.27 (588) | 93.01 $\pm$ 1.71 (588) | 91.99 $\pm$ 1.64 (588) |

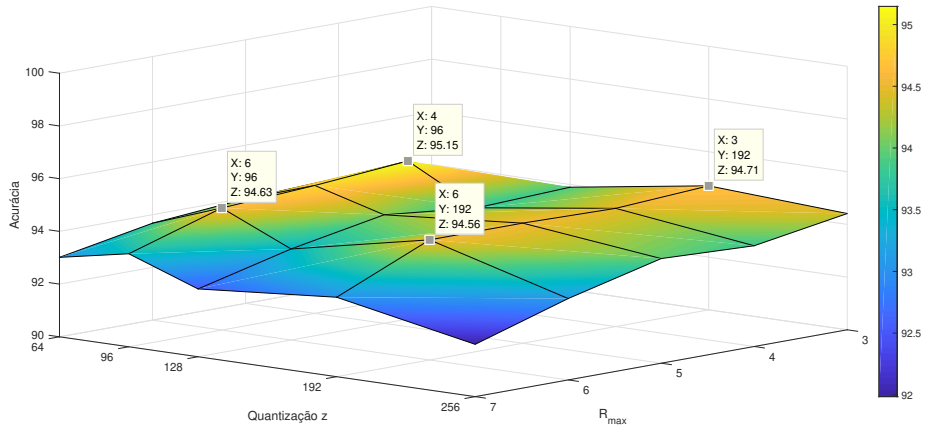



**Figura 17 – Comparação visual da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Outex.**

**Tabela 4 – Comparação numérica da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Vistex. Fomato: acurácia $\pm$ desvio padrão (número de atributos)**

| | **quantização $z$** | | | | |
|---|---|---|---|---|---|
| $r_{max}$ | **64** | **96** | **128** | **192** | **256** |
| **3** | 99.65 $\pm$ 0.56 (112) | 99.65 $\pm$ 0.56 (112) | 99.65 $\pm$ 0.56 (112) | 99.77 $\pm$ 0.49 (112) | 99.89 $\pm$ 0.36 (112) |
| **4** | 99.77 $\pm$ 0.49 (196) | 99.65 $\pm$ 0.56 (196) | 99.65 $\pm$ 0.56 (196) | 99.77 $\pm$ 0.49 (196) | 99.77 $\pm$ 0.49 (196) |
| **5** | 99.77 $\pm$ 0.49 (308) | 99.77 $\pm$ 0.49 (308) | 99.77 $\pm$ 0.49 (308) | 99.54 $\pm$ 0.60 (308) | 99.89 $\pm$ 0.36 (308) |
| **6** | 99.65 $\pm$ 0.78 (434) | 99.77 $\pm$ 0.49 (434) | 99.77 $\pm$ 0.49 (434) | 99.65 $\pm$ 0.56 (434) | 99.65 $\pm$ 0.56 (434) |
| **7** | 99.42 $\pm$ 0.99 (588) | 99.65 $\pm$ 0.56 (588) | 99.19 $\pm$ 0.56 (588) | 99.42 $\pm$ 0.61 (588) | 99.19 $\pm$ 0.78 (588) |

[1] Utilizando classificador LDA, validação cruzada por k-fold ($k = 10$), e padronização z-score

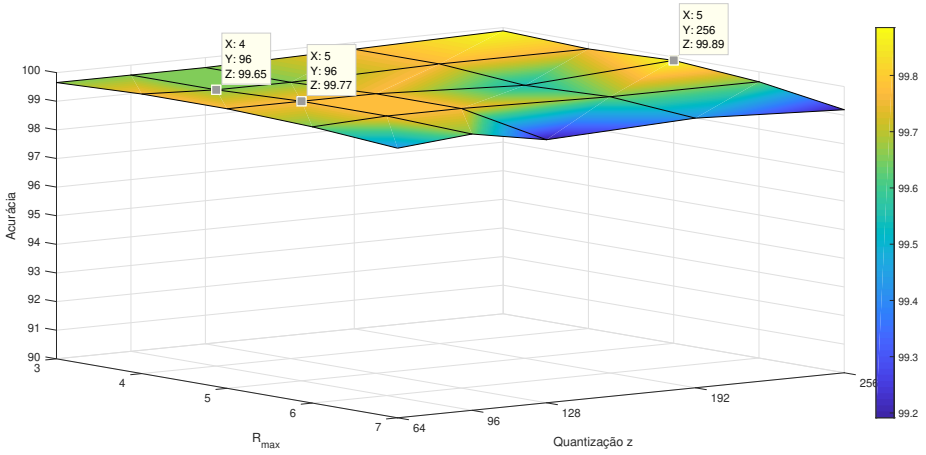



**Figura 18 – Comparação visual da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Vistex.**

**Tabela 5 – Comparação numérica da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Usptex. Fomato: acurácia $\pm$ desvio padrão (número de atributos)**

| | **quantização $z$** | | | | |
|---|---|---|---|---|---|
| $r_{max}$ | **64** | **96** | **128** | **192** | **256** |
| **3** | 98.27 $\pm$ 0.30 (112) | 98.19 $\pm$ 0.31 (112) | 98.17 $\pm$ 0.64 (112) | 98.27 $\pm$ 0.47 (112) | 98.32 $\pm$ 0.65 (112) |
| **4** | 98.39 $\pm$ 0.50 (196) | 98.54 $\pm$ 0.41 (196) | 98.69 $\pm$ 0.37 (196) | 98.57 $\pm$ 0.54 (196) | 98.79 $\pm$ 0.53 (196) |
| **5** | 98.54 $\pm$ 0.51 (308) | 98.77 $\pm$ 0.42 (308) | 98.85 $\pm$ 0.49 (308) | 98.87 $\pm$ 0.53 (308) | 99.02 $\pm$ 0.54 (308) |
| **6** | 98.80 $\pm$ 0.39 (434) | 98.80 $\pm$ 0.37 (434) | 99.05 $\pm$ 0.35 (434) | 98.97 $\pm$ 0.40 (434) | 99.10 $\pm$ 0.40 (434) |
| **7** | 98.92 $\pm$ 0.36 (588) | 98.75 $\pm$ 0.55 (588) | 99.05 $\pm$ 0.26 (588) | 98.85 $\pm$ 0.21 (588) | 99.05 $\pm$ 0.33 (588) |

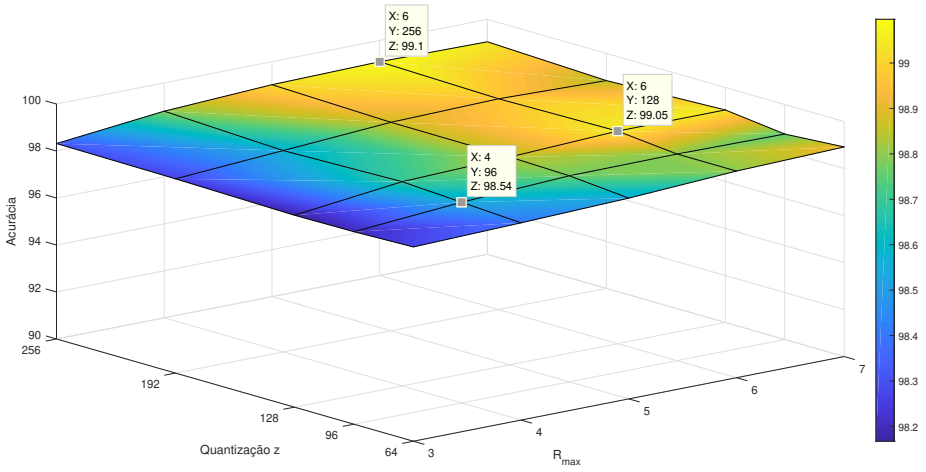



**Figura 19 – Comparação visual da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Usptex.**

**Tabela 6 – Comparação numérica da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Madeiras. Fomato: acurácia $\pm$ desvio padrão (número de atributos)**

| | quantização $z$ | | | | |
|---|---|---|---|---|---|
| $r_{max}$ | **64** | **96** | **128** | **192** | **256** |
| **3** | 95.12 $\pm$ 1.45 (112) | 94.96 $\pm$ 1.64 (112) | 94.76 $\pm$ 1.62 (112) | 94.69 $\pm$ 1.30 (112) | 94.50 $\pm$ 1.47 (112) |
| **4** | 95.52 $\pm$ 1.19 (196) | 95.88 $\pm$ 1.60 (196) | 95.55 $\pm$ 1.52 (196) | 95.98 $\pm$ 1.15 (196) | 95.45 $\pm$ 1.38 (196) |
| **5** | 95.81 $\pm$ 1.14 (308) | 95.75 $\pm$ 1.28 (308) | 95.85 $\pm$ 1.52 (308) | 96.34 $\pm$ 1.12 (308) | 95.65 $\pm$ 1.42 (308) |
| **6** | 95.75 $\pm$ 1.22 (434) | 95.39 $\pm$ 1.55 (434) | 95.81 $\pm$ 1.42 (434) | 96.05 $\pm$ 1.30 (434) | 95.81 $\pm$ 0.88 (434) |
| **7** | 95.78 $\pm$ 1.27 (588) | 95.55 $\pm$ 1.43 (588) | 95.62 $\pm$ 1.55 (588) | 95.72 $\pm$ 1.37 (588) | 95.75 $\pm$ 0.78 (588) |

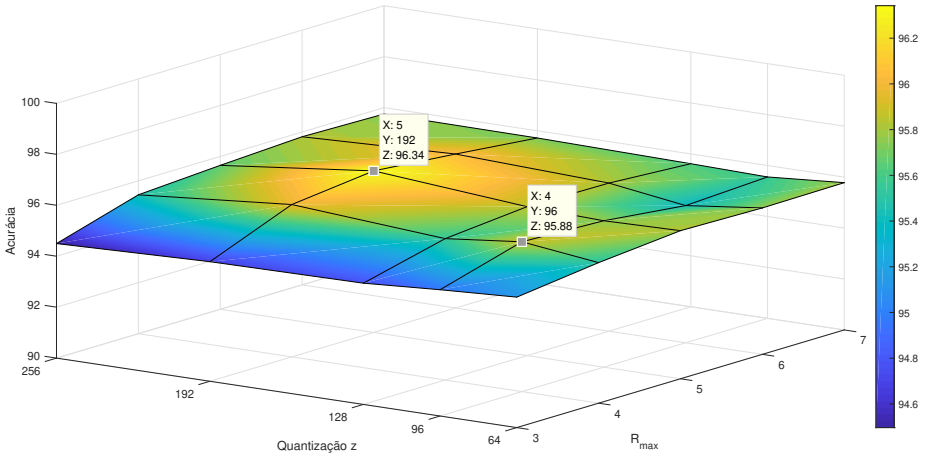



**Figura 20 – Comparação visual da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Madeiras.**

**Tabela 7 – Comparação numérica da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ em relação a média da acurácia das bases de imagens.**

| | quantização $z$ | | | | |
|---|---|---|---|---|---|
| $r_{max}$ | **64** | **96** | **128** | **192** | **256** |
| **3** | 96.68 | 96.58 | 96.62 | 96.86 | 96.78 |
| **4** | 96.95 | 97.31 | 96.91 | 97.20 | 96.96 |
| **5** | 97.02 | 97.28 | 97.14 | 97.33 | 97.13 |
| **6** | 96.97 | 97.15 | 97.02 | 97.31 | 96.91 |
| **7** | 96.78 | 96.87 | 96.61 | 96.75 | 96.49 |

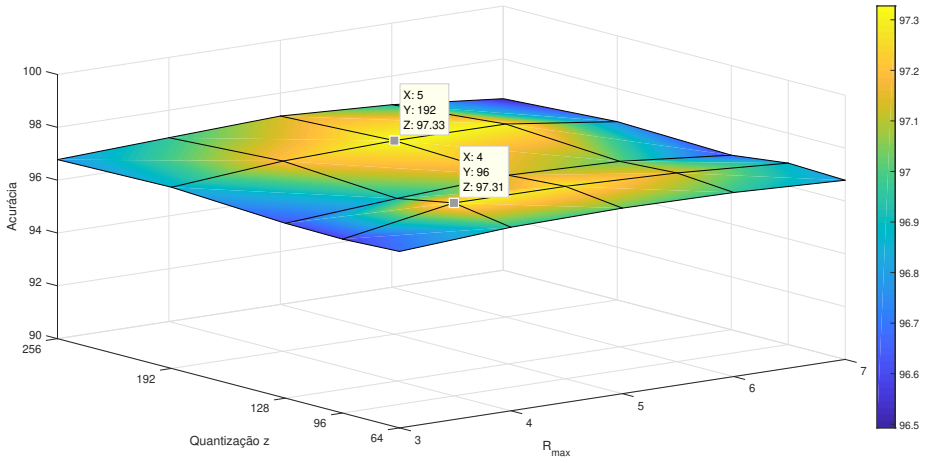



**Figura 21 – Comparação visual da acurácia do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ em relação a média da acurácia das bases de imagens.**

Os valores de $r_{max}$ e quantização $z$ influenciam no tempo de execução do método $FD_{MIG}$, tornando-se um fator de ponderação na escolha dos parâmetros. A Tabela 8 apresenta a comparação numérica do tempo de execução médio (em milissegundos no mesmo computador) do método $FD_{MIG}$ utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Outex_TC_00013 (apresentada em 4.1). A Figura 22 ilustra os valores apresentados na Tabela 8 por meio de uma superfície tridimensional.

**Tabela 8 – Comparação numérica do tempo de execução médio (em milissegundos no computador Inspiron 5767, processador Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz 2.90 GHz, RAM instalada 16,0 GB) do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Outex_TC_00013.**

| | **quantização $z$** | | | | |
|---|---|---|---|---|---|
| $r_{max}$ | **64** | **96** | **128** | **192** | **256** |
| **3** | 192 | 215 | 252 | 291 | 341 |
| **4** | 195 | 223 | 250 | 305 | 360 |
| **5** | 203 | 232 | 262 | 321 | 383 |
| **6** | 209 | 240 | 274 | 336 | 401 |
| **7** | 216 | 249 | 285 | 351 | 417 |

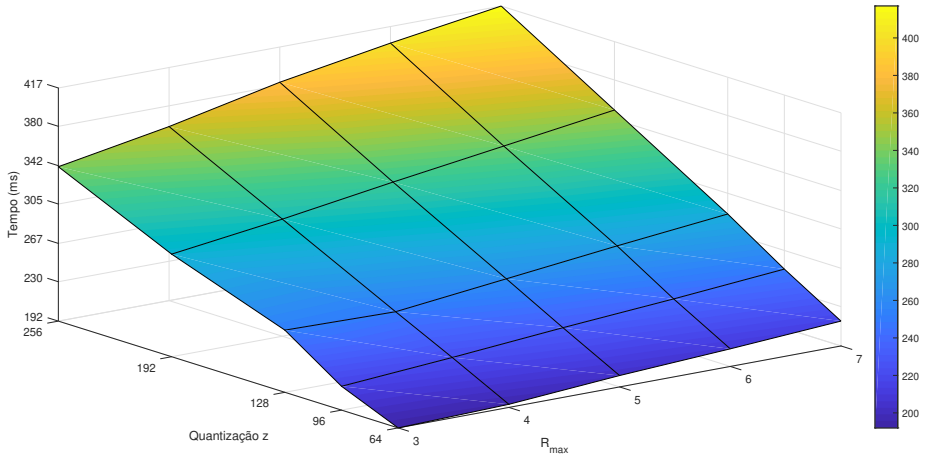



**Figura 22 – Comparação visual do tempo de execução médio (em milissegundos) do método proposto utilizando diferentes valores para $r_{max}$ e quantização $z$ sobre a base de imagens Outex.**

Como pode ser observado na Tabela 7 e Figura 21 o valor de maior acurácia média das bases de imagens analisada é obtido quando utilizados os parâmetros $r_{max} = 5$ e Quantização $z = 192$, resultando em acurácia $97.33\%$, contudo, esse resultado é pouco superior ao obtido quando utilizados os parâmetros $r_{max} = 4$ e Quantização $z = 96$ que resulta em acurácia $97.31\%$. Ao analisar as Tabelas 2 e 8 percebe-se que o custo beneficio de utilizar essa segunda combinação de parâmetros é maior do que o da primeira, uma vez que o custo computacional de extração de características da segunda combinação é cerca de $30\%$ inferior ao custo da primeira, além disso, a dimensionalidade da segunda combinação de parâmetros é cerca de $36.4\%$ inferior do que a dimensionalidade da primeira. Portanto, definiu-se os parâmetros padrões do método proposto como $r_{max} = 4$ e quantização $z = 96$.

## 4.3 COMPARAÇÕES DE ACURÁCIA COM MÉTODOS CLÁSSICOS DE ANÁLISE DE TEXTURA

Nessa Seção, para entrada dos testes de acurácia entre $FD_{MIG}$ e os métodos da literatura são utilizadas as bases de imagens Outex_TC_00013, Vistex, Usptex e Madeiras.

Objetivando padronização na validação dos resultados, os parâmetros de $FD_{MIG}$ foram definidos e fixados como $r_{max} = 4$ e quantização $z = 96$ em todos os testes de acurácia. Os parâmetros para os métodos de validação aplicados ao $FD_{MIG}$ e aos métodos de AT da literatura foram definidos e fixados em todos os testes dessa seção como:

- Número de pastas do método de validação cruzada k-fold com $k = 10$ (valor comum na literatura);
- Transformação dos dados: Padronização z-score;
- Classificador LDA;

**Tabela 9 – Métodos utilizados para comparação de resultados de acurácia.**

| Método | Significado da sigla ou título do trabalho | Referência |
|---|---|---|
| $FD_{MIC}$ | Texture analysis using fractal descriptors estimated by the mutual interference of color channels | (CASANOVA et al., 2016) |
| CN | Texture analysis and classification: A complex network-based approach | (BACKES; CASANOVA; BRUNO, 2013) |
| DCT | Discrete Cosine Transform | (NG; TAN; KITTLER, 1992) |
| Fourier | Texture classification using windowed Fourier filters | (AZENCOTT; WANG; YOUNES, 1997) |
| FirstOrder | First-order histogram | (MATERKA; STRZELECKI et al., 1998) |
| GLCM | Gray Level Co-occurrence Matrix | (HARALICK, 1979) |
| GLDM | Gray Level Difference Matrix | (WESZKA; DYER; ROSENFELD, 1976) |
| Wavelets | Wavelets descriptors | (CHANG; KUO, 1993) |

Os métodos de AT da literatura implementados para serem utilizados nessa seção como comparação de acurácia com o $FD_{MIG}$ foram listados na Tabela 9.

A Tabela 10 apresenta a comparação numérica da acurácia do método proposto em relação a métodos de AT da literatura. As Figuras 23, 24, 25 e 26 ilustra os valores apresentados na Tabela 10.

**Tabela 10 – Resumo de desempenho (%) de alguns métodos de AT em bases de imagens de benchmark. Formato: acurácia $\pm$ desvio padrão**

| Informações do método | | | Acurácia sobre bases de benchmark | | | |
|---|---|---|---|---|---|---|
| **Método** | **Dimensão** | **Ano** | **Outex** | **Usptex** | **Madeiras** | **Vistex** |
| $FD_{MIG}$ | 196 | 2019 | **95.15 $\pm$ 1.05** | **98.54 $\pm$ 0.41** | **95.88 $\pm$ 1.60** | **99.65 $\pm$ 0.56** |
| $FD_{MIG}$ Gray | 140 | 2019 | 85.07 $\pm$ 2.25 | 90.66 $\pm$ 0.93 | 86.62 $\pm$ 2.01 | 98.61 $\pm$ 1.20 |
| $FD_{MIC}$ | 66 | 2016 | 92.57 $\pm$ 1.91 | 96.08 $\pm$ 0.75 | 92.62 $\pm$ 1.55 | 98.96 $\pm$ 0.85 |
| CN | 108 | 2013 | 89.56 $\pm$ 1.83 | 91.16 $\pm$ 1.57 | 86.39 $\pm$ 1.55 | 98.26 $\pm$ 1.13 |
| DCT | 8 | 1992 | 69.26 $\pm$ 4.71 | 52.99 $\pm$ 2.71 | 65.53 $\pm$ 2.74 | 80.56 $\pm$ 2.41 |
| FirstOrder | 18 | 1998 | 56.54 $\pm$ 1.88 | 27.71 $\pm$ 2.68 | 27.69 $\pm$ 2.82 | 62.62 $\pm$ 3.90 |
| Fourier | 64 | 1997 | 85.00 $\pm$ 2.67 | 77.84 $\pm$ 2.60 | 62.49 $\pm$ 2.58 | 92.84 $\pm$ 3.65 |
| GLCM | 32 | 1979 | 83.75 $\pm$ 2.93 | 82.00 $\pm$ 1.96 | 74.95 $\pm$ 2.54 | 94.45 $\pm$ 2.92 |
| GLDM | 60 | 1976 | 87.13 $\pm$ 1.64 | 89.83 $\pm$ 1.06 | 82.07 $\pm$ 2.04 | 97.46 $\pm$ 2.02 |
| Wavelets | 36 | 1993 | 79.49 $\pm$ 2.67 | 74.95 $\pm$ 1.36 | 73.73 $\pm$ 1.74 | 91.44 $\pm$ 2.36 |

Analisando as Tabelas 10, pode-se concluir que o método proposto apresenta elevada acurácia nas bases de imagens em questão, superando os outros métodos da literatura utilizados para comparação. Além disso, nota-se que o método proposto superou seu antecessor $FD_{MIC}$, indicando que a rotulação dos píxeis trouxe informações uteis para a análise de textura.

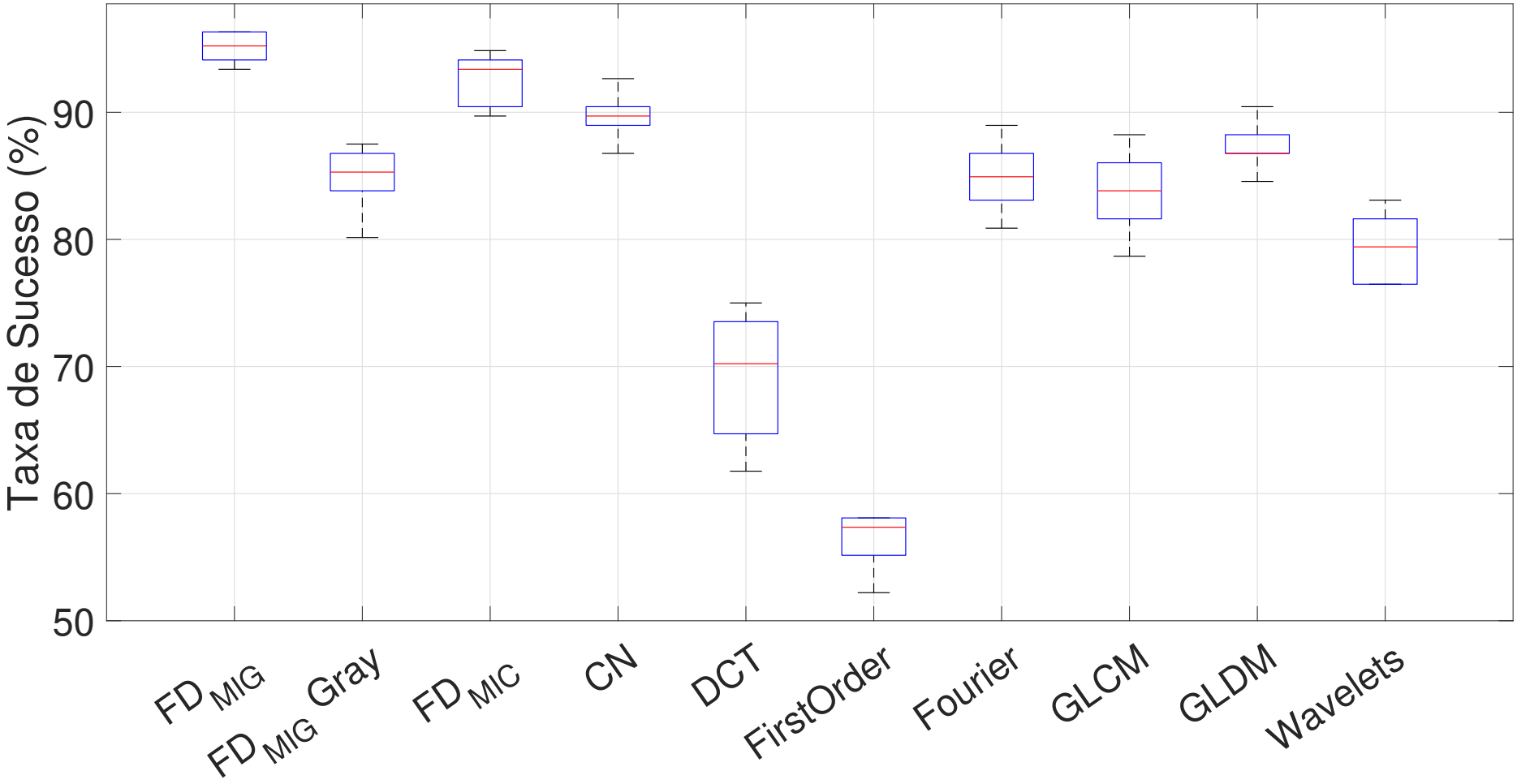



**Figura 23 – Resumo de desempenho (%) de métodos de AT na base de imagens de benchmark Outex. Cada subfigura representa os valores de acurácia para uma base de imagem, cada caixa dentro das delas representam um método de AT. Em cada caixa, a marca central indica a mediana da acurácia, e as bordas inferior e superior da caixa indicam os percentis 25 e 75, respectivamente. As bordas externas estendem-se até os pontos dos dados mais extremos que não são considerados *outliers*, que são representados individualmente pelo símbolo "+". São considerados *outliers* os valores fora do intervalo $\mu \pm 2,7\sigma$, em que $\mu$ é a média e $\sigma$ é o desvio padrão.**

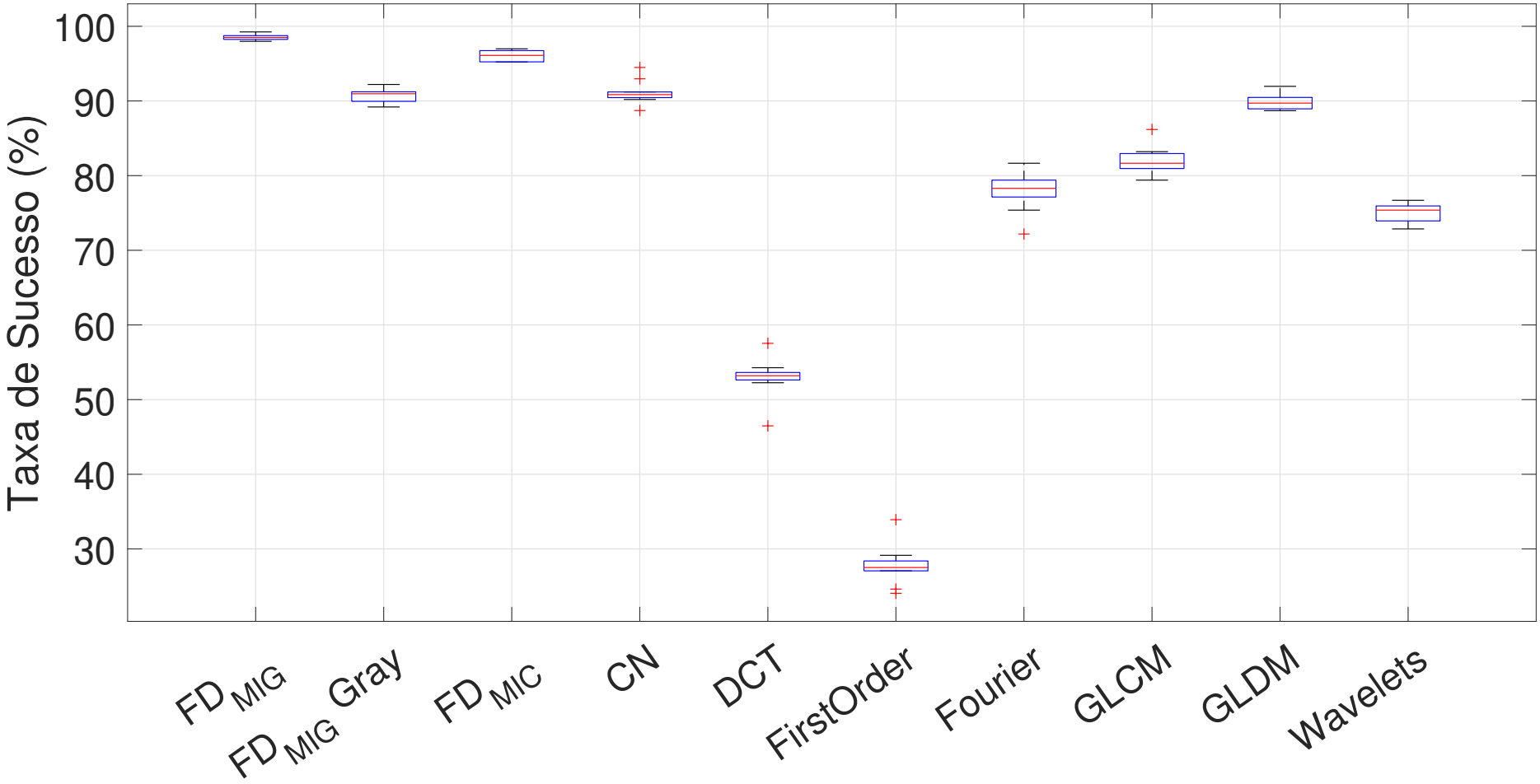



**Figura 24 – Resumo de desempenho (%) de métodos de AT na base de imagens de benchmark Usptex. Cada subfigura representa os valores de acurácia para uma base de imagem, cada caixa dentro das delas representam um método de AT. Em cada caixa, a marca central indica a mediana da acurácia, e as bordas inferior e superior da caixa indicam os percentis 25 e 75, respectivamente. As bordas externas estendem-se até os pontos dos dados mais extremos que não são considerados *outliers*, que são representados individualmente pelo símbolo "+". São considerados *outliers* os valores fora do intervalo $\mu \pm 2,7\sigma$, em que $\mu$ é a média e $\sigma$ é o desvio padrão.**

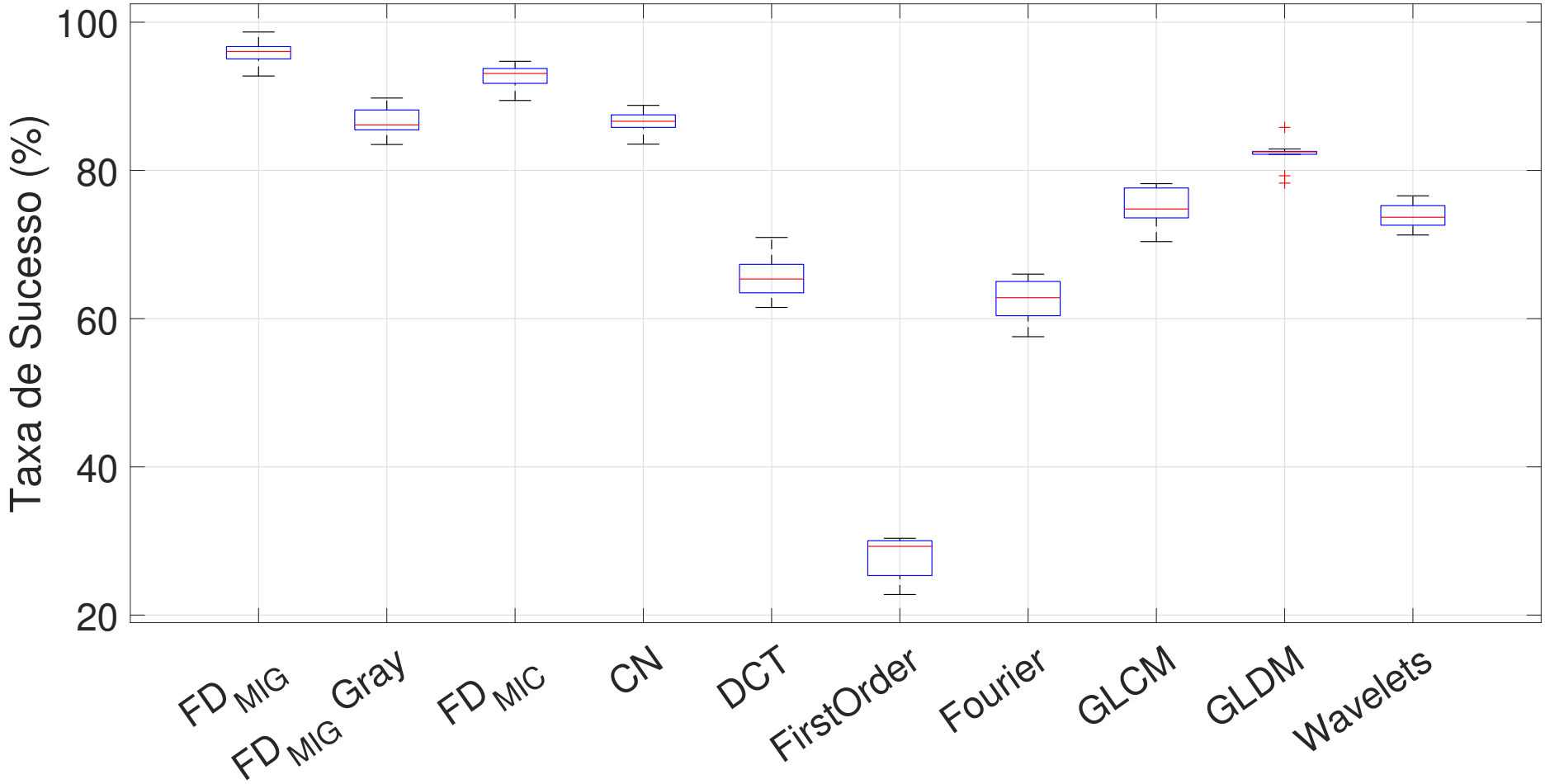



**Figura 25 – Resumo de desempenho (%) de métodos de AT na base de imagens de benchmark Madeiras. Cada subfigura representa os valores de acurácia para uma base de imagem, cada caixa dentro das delas representam um método de AT. Em cada caixa, a marca central indica a mediana da acurácia, e as bordas inferior e superior da caixa indicam os percentis 25 e 75, respectivamente. As bordas externas estendem-se até os pontos dos dados mais extremos que não são considerados *outliers*, que são representados individualmente pelo símbolo "+". São considerados *outliers* os valores fora do intervalo $\mu \pm 2,7\sigma$, em que $\mu$ é a média e $\sigma$ é o desvio padrão.**

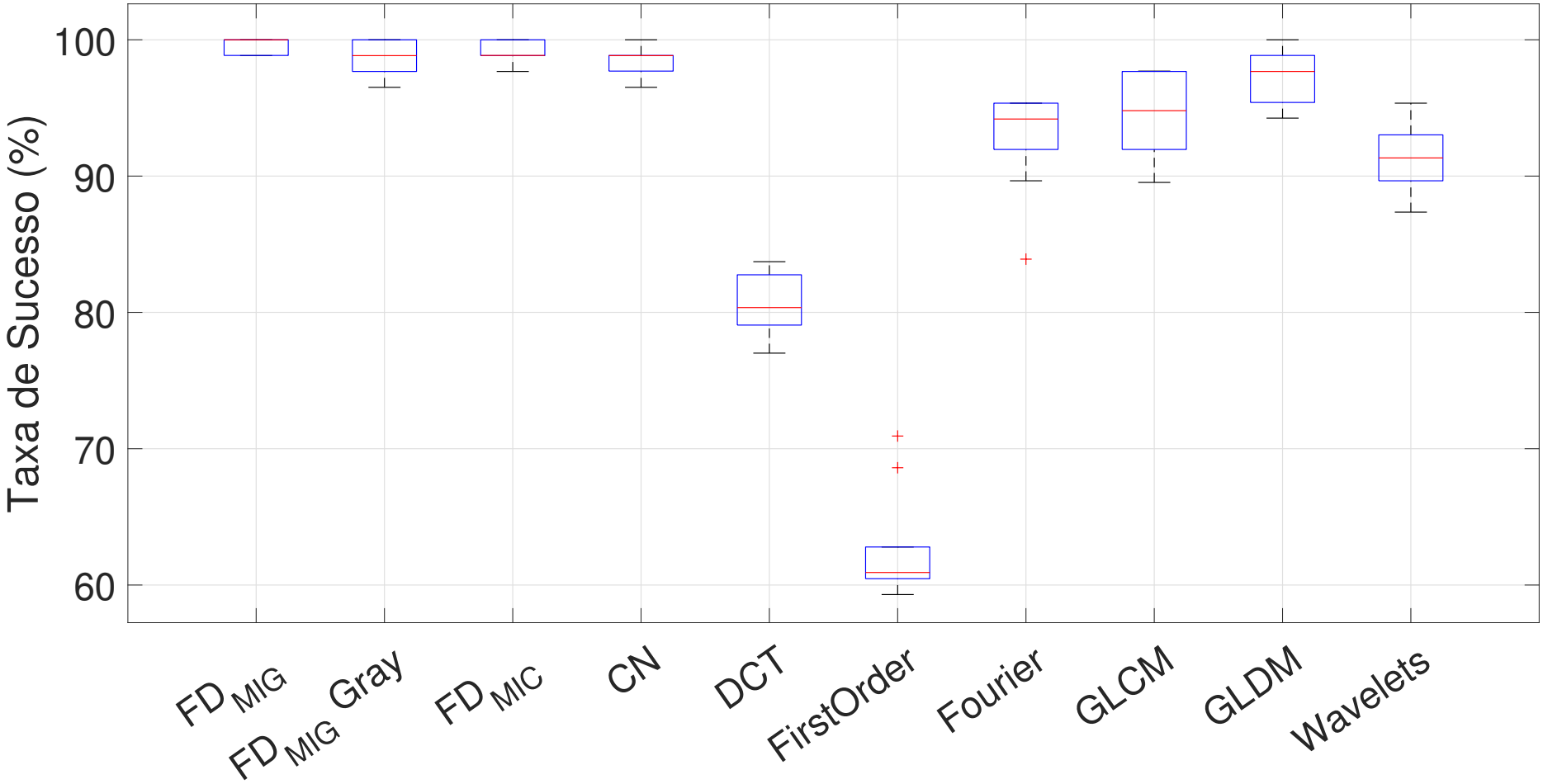



**Figura 26 – Resumo de desempenho (%) de métodos de AT na base de imagens de benchmark Vistex. Cada subfigura representa os valores de acurácia para uma base de imagem, cada caixa dentro das delas representam um método de AT. Em cada caixa, a marca central indica a mediana da acurácia, e as bordas inferior e superior da caixa indicam os percentis 25 e 75, respectivamente. As bordas externas estendem-se até os pontos dos dados mais extremos que não são considerados *outliers*, que são representados individualmente pelo símbolo "+". São considerados *outliers* os valores fora do intervalo $\mu \pm 2,7\sigma$, em que $\mu$ é a média e $\sigma$ é o desvio padrão.**

## 4.4 COMPARAÇÃO DE ACURÁCIA COM MÉTODOS MODERNOS DE ANÁLISE DE TEXTURA

Objetivando verificar e comparar a acurácia do método proposto com a de métodos recentes da literatura utilizou-se a survey de (LIU et al., 2018). Nessa survey é possível encontrar a comparação de acurácia entre vários métodos de análise de textura, que incluem tanto métodos tradicionais (i.e. utilizam extração de características para classificação de imagens de textura) quanto métodos novos baseados em redes neurais convolucionais (CNN) (LIU et al., 2018). Essas comparações ocorrem sobre imagens provenientes de famosas bases de Benchmark, algumas particularmente desenvolvidas para testar algoritmos robustos. A Tabela 12 expõe essas comparações de acurácia obtidas de (LIU et al., 2018) juntamente com a acurácia calculada para o método proposto (FDMIG).

Das bases de Benchmark expostas em (LIU et al., 2018), utilizou-se KTHTIPS, KTH-TIPS2b e ALOT, por serem compatíveis com o método proposto, as informações sobre elas são expostas a seguir:

- KTHTIPS (FRITZ et al., 2004) é uma base de imagens colorida com 10 classes diferentes, apresenta variação em iluminação, posição e escala. Cada classe possui 81 imagens com dimensões aproximadamente $200 \times 200$ podendo haver variações, sendo assim a base possui 810 imagens de textura. Em (LIU et al., 2018) a validação de acurácia sobre essa base é feita utilizando aleatoriamente metade das instâncias para treino e a outra metade para testes.
- KTHTIPS2b (MALLIKARJUNA et al., 2006) é uma base de imagens colorida com 11 classes diferentes, apresenta variação em iluminação, posição e escala. Cada classe é subdividida em 4 amostras, cada amostra possui 108 imagens de textura com dimensões aproximadamente $200 \times 200$ podendo haver variações, existem grandes variações na textura e cor de amostras diferentes para a mesma classe. Cada classe possui 432 imagens, ao todo a base possuí 4752 imagens. Em (LIU et al., 2018) a validação de acurácia sobre essa base é feita utilizando uma das quatro amostras para treino e as outras três para testes.
- ALOT (BURGHOUTS; GEUSEBROEK, 2009; FRITZ et al., 2004) é uma base de imagens coloridas com 250 classes de textura obtidas com critérios de cor de iluminação controlada, angulo de iluminação variável e angulações de aquisição variáveis. Cada classe possui 100 imagens diferentes, totalizando 25000 imagens com dimensões exatamente $384 \times 256$. Em (LIU et al., 2018) a validação de acurácia sobre essa base é feita utilizando aleatoriamente metade das instâncias para treino e a outra metade para testes.

Como as bases testadas não apresentam dimensões fixa, o vetor de características deve ser normalizado pelo número de píxeis da imagem, como sugerido por (MALLIKARJUNA et al., 2006).

**Tabela 11 – Informações sobre os métodos utilizados para comparação com o método proposto.**

| Método | | Significado da sigla ou Titulo do trabalho | Referência |
|---|---|---|---|
| Tradicional | LBPRIU2 | Multiresolution gray-scale and rotation invariant texture classification with local binary patterns | (OJALA; PIETIKAINEN; MAENPAA, 2002) |
| | Lazebnik et al. | A sparse texture representation using local affine regions | (LAZEBNIK; SCHMID; PONCE, 2005) |
| | Zhang et al. | Local features and kernels for classification of texture and object categories: A comprehensive study | (ZHANG et al., 2007) |
| | Patch | A statistical approach to material classification using image patch exemplars | (VARMA; ZISSERMAN, 2009) |
| | BIF | Basic Image Features | (CROSIER; GRIFFIN, 2010) |
| | SRP | Sorted random projections for robust texture classification | (LIU et al., 2011) |
| | Timofte et al. | A Training-free Classification Framework for Textures, Writers, and Materials | (TIMOFTE; GOOL, 2012) |
| | MRELBP | Median robust extended local binary pattern for texture classification | (LIU et al., 2016b) |
| | SIFT | Scale Invariant Feature Transform | (CIMPOI et al., 2016) |
| CNN | ScatNet | Invariant scattering convolution networks | (BRUNA; MALLAT, 2013) |
| | AlexNet | Alex Network | (CIMPOI et al., 2016) |
| | VGGM | Visual Geometry Group Medium | (CIMPOI et al., 2016) |
| | VGGVD | Visual Geometry Group Very deep convolutional networks | (CIMPOI et al., 2016) |
| | BCNN | Bilinear CNN | (GAO et al., 2016) |
| | LFVCNN | Locally-Transferred Fisher Vectors CNN | (SONG et al., 2017) |

A Tabela 11 sumariza as informações sobre os métodos de análise de textura da literatura utilizados para comparação de acurácia na Tabela 12.

Como pode ser observado na Tabela 12, o método LBP supera o $FD_{MIG}$ na base de imagens ALOT. O método $FD_{MIG}$ utiliza informações obtidas de LBP, contudo o valor de acurácia apresentado é para a melhor combinação de parâmetros de LBP para essa base, que é uma mascara com dimensões $19 \times 19$ (apresentado em (LIU et al., 2016a)), e $FD_{MIG}$ utiliza o LBP com uma mascara de $3 \times 3$. Na base KTHTIPS o $FD_{MIG}$ apresenta acurácia próxima aos outros métodos, inclusive aos métodos baseados em CNN. Já na base KTHTIPS2b o $FD_{MIG}$ apresenta acurácia inferior aos outros, pois a metodologia de validação dessa base utiliza imagens de mesma classe visualmente diferentes entre o conjunto de treinamento e o conjunto de testes.

**Tabela 12 – Resumo de desempenho (%) de alguns métodos representativos em conjuntos de dados de textura de referência populares. Todos os métodos usaram a mesma estratégia de divisão para treinamento e teste em cada conjunto de dados. Todos os resultados listados foram obtidos por (LIU et al., 2018) a partir dos documentos originais, exceto aqueles marcados com (*) que são de (ZHANG et al., 2007) e marcados com (@) que são de (LIU et al., 2016a). Para mais informações sobre parâmetros dos métodos consulte (LIU et al., 2018)**

| Informações do método | | | | | Acurácia sobre bases de benchmark | | |
|---|---|---|---|---|---|---|---|
| | Método | Ano | Classificador | Dimensão | KTHTIPS | KTHTIPS2b | ALOT |
| Tradicional | LBPRIU2 | 2002 | Chi Square, NNC | 210 | - | - *4,2 (@) | |
| | Lazebnik et al. | 2005 | EMD, NNC | 40 Clusters | 91,3 (*) | - | - |
| | Zhang et al. | 2007 | EMD, SVM | 40 Clusters | 95,5 | - | - |
| | Patch | 2009 | Chi Square, NNC | Depende do Dataset | 92,4 (*) | - | - |
| | BIF | 2010 | Chi Square, NNC | 1296 | 98,5 | - | - |
| | SRP | 2011 | Chi Square, SVM | Depende do Dataset | 99,3 | - | - |
| | Timofte et al. | 2012 | Raciocínio Colaborativo | 1780 | 99,4 | - | - |
| | MRELBP | 2016 | Chi Square, SVM | 800 | - | - | 99,1(@) |
| | SIFT | 2016 | Linear SVM | 65536 | 99,5 | 70,8 | - |
| | **FDMIG** | **2019** | **LDA** | **196** | **99,0** | **54,0** | **92,0** |
| CNN | ScatNet | 2013 | PCA Classifier | 596 | 99,4 | - | 98,3 (@) |
| | AlexNet | 2016 | Linear SVM | 32768 | 99,6 | 69,7 | 99,1 (@) |
| | VGGM | 2016 | Linear SVM | 65536 | **99,8** | 73,3 | 99,4 (@) |
| | VGGVD | 2016 | Linear SVM | 65536 | **99,8** | 81,8 | **99,5** (@) |
| | BCNN | 2016 | Linear SVM | 262144 | - | 77,9 | - |
| | LFVCNN | 2017 | LFV classifier | 65536 | - | **82,6** | - |

# 5 CLASSIFICAÇÃO DE SINAIS UNIDIMENSIONAIS

Nesse capítulo é apresentado um estudo de caso do método proposto em processamento de sinais unidimensionais. Mais especificamente, reconhecimento de voz.

Em processamento de sinais, uma das áreas de interesse é o processamento de áudio, seja para detecção de propriedades no sinal ou sua manipulação. Dentro do processamento de áudio há o interesse pela voz humana, onde aplicações variam de manipulações no afinamento, como o software Auto-tune, até reconhecimento do timbre de um indivíduo, no caso de sistemas de segurança implementados em *smartphones* modernos, que pode ser aplicado para desbloqueio de *smartphones* ao identificar que a voz pertence ao dono, assim como era feito pelo programa "Google Assistente".

O problema da classificação da fala é muito importante porque sua solução afeta outros problemas de análise, síntese e reconhecimento da fala. Por exemplo, a classificação de fala pode ajudar a melhorar a síntese de fala (LIAO; GREGORY, 1999). Existem vários métodos para o problema de classificação da fala (SHI; FAN, 2017), (LIAO; GREGORY, 1999), (GAIKWAD; GAWALI; YANNAWAR, 2010), etc.

Digitalmente a voz é representada por um sinal unidimensional (ou dois sinais unidimensionais para o som estéreo) como apresentado na Figura 27.

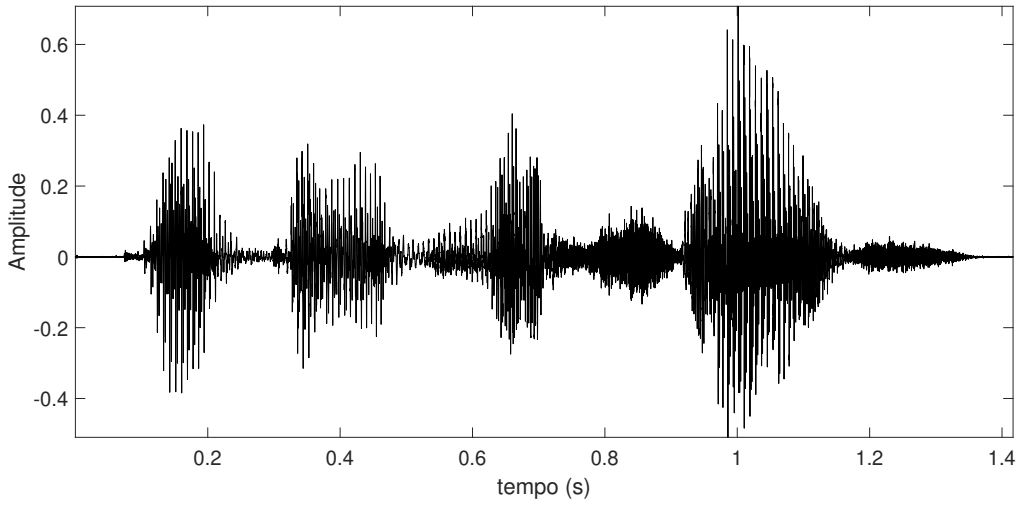



**Figura 27 – Sinal de voz**

Uma das maneiras de se analisar um sinal de voz é por meio de um espectograma. Espectrogramas são gráficos que analisam dinamicamente a densidade espectral de energia de um sinal. Geralmente os espectrogramas são gerados a partir de uma transformada de Fourrier.

O algoritmo disponível em Wojcicki (2010) computa um *short-time Fourier transform* e traça a relação tempo $\times$ frequência $\times$ energia do espectro, gerando um espectograma que pode ser representado como uma imagem em escala de cinza. A Figura 28 é o espetrograma gerado por esse algoritmo usando como entrada a voz representada na Figura 27.

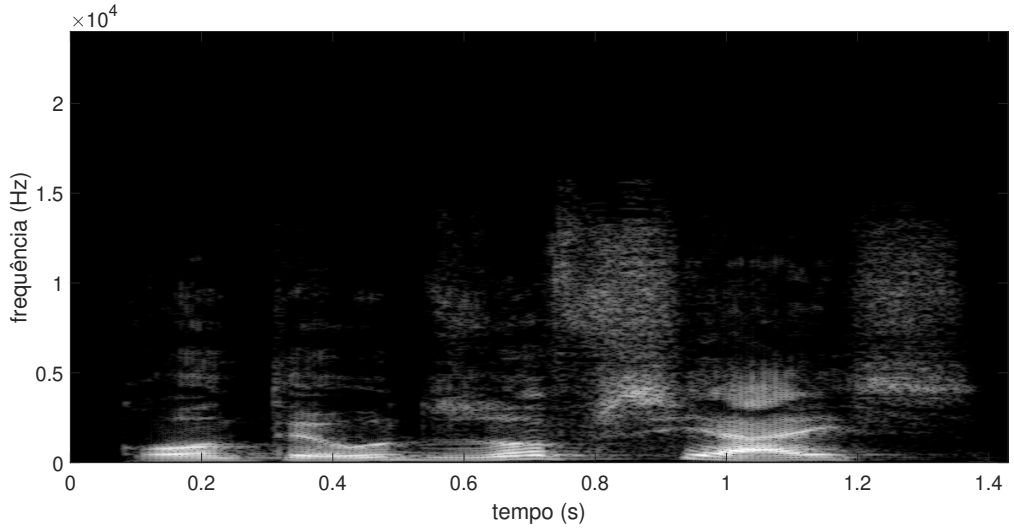



**Figura 28 – Espectrograma do sinal de áudio representado na Figura 27 com $Fs = 48000Hz$**

Nesse contexto, como o sinal de voz pode ser representado como um espectrograma que por sua vez pode ser representado como uma imagem em tons de cinza, levanta-se a hipótese de que o método proposto $FD_{MIG}$ Gray pode ser usado para classificar sinais de voz.

Nesse contexto foi construída uma base de dados com 10 vozes diferentes. Os sinais de áudio foram obtidos de vídeos da plataforma de compartilhamento de vídeos YouTube, conforme a Tabela 13. Objetivando restringir o espaço amostral em relação as frequências analisadas foram selecionados 10 vídeos de homens dialogando sem música de fundo. Para cada vídeo, e consequentemente cada voz, foi realizada a aquisição de 60 amostras de áudio com duração 1 segundo, totalizando 1 minuto de amostragem de áudio. Ao todo a base de áudio é constituída por 600 amostas de duração 1 segundo. Em momento de aquisição definiu-se que todas as amostras selecionadas apresentassem no mínimo 90 % dos valores de energia do espectro contidos no intervalo entre 0 dB e -128 dB.

**Tabela 13 – Origem dos sinais de áudio das vozes utilizadas no estudo de caso**

| Classe | Canal do YouTube / Assunto | URL | Início |
|---|---|---|---|
| 0 | Two Minute Papers | https://youtu.be/KhP7lTLTipc | 0,5 s |
| 1 | Curso em Vídeo | https://youtu.be/J5q7s7l2EuI | 4 m 27 s |
| 2 | Me Salva! / CRC01 | https://youtu.be/JFUBx2QsEhg | 12 s |
| 3 | Me Salva! / SER01 | https://youtu.be/qeQJMnCCSQw | 1 m 7 s |
| 4 | nerckie | https://youtu.be/1zxL3MYdK04 | 20 s |
| 5 | Aulalivre / Português | https://youtu.be/49P7uTXpPOI | 10 s |
| 6 | Canal do Cortella | https://youtu.be/4kk5qKrUfNo | 1 s |
| 7 | Dr. Dayan Siebra | https://youtu.be/0BwOTjtVfGY | 1 s |
| 8 | Território Conhecimento / Leandro Karnal | https://youtu.be/ivkzCmOuoPY | 3 s |
| 9 | Canal do Pirula | https://youtu.be/SAoFkZczm2Y | 1 s |

Os sinais de áudio de todos os vídeos do YouTube foram convertidos para o formato *mp3* com frequência de amostragem $Fs = 48000$, que realiza compressão de áudio com perdas quase imperceptíveis ao ouvido humano.

As dimensões da imagem gerada a partir do espectrograma estão em função da frequência de amostragem ($Fs$), quanto maior a $Fs$ maior é a imagem e mais informações do áudio são mapeadas na imagem. Contudo, o $FD_{MIG}$ analisa imagens extraindo informações de vizinhança, e a frequência da voz humana varia aproximadamente entre $50Hz$ e $3400Hz$ (WIKIPéDIA, 2012), que é um amplo intervalo. Ou seja, para obter melhor aproveitamento da capacidade de discriminação do $FD_{MIG}$ a frequência de amostragem deve ser reduzida, e consequentemente fazendo com que o $FD_{MIG}$ extraia características relevantes da voz humana.

Para reduzir o valor de $Fs$ tomou-se como base a frequência de amostragem do sinal de telefonia, que utiliza $Fs = 8000Hz$. De acordo com o teorema da amostragem de Nyquist–Shannon (CANDÈS et al., 2006) a frequência de amostragem de um sinal senoidal deve ser no mínimo o dobro da maior frequência de interesse, no caso $Fs = 8000Hz$ aceitaria frequências de interesse de até $4000Hz$ (i.e., $600Hz$ acima da maior frequência de interesse que é aproximadamente $3400Hz$). Contudo, após experimentações empíricas notou-se distorções na voz, porém ao dobrar a frequência de amostragem (i.e., utilizando frequência de amostragem $Fs = 16000Hz$) as mesmas distorções não são percebidas de maneira empírica. Portanto, os áudios do estudo de caso foram re-amostrados definindo $Fs = 16000Hz$ por meio da função *resample* do software Matlab, que automaticamente aplica um filtro *antialiasing* FIR passa-baixas para compensar as distorções causadas por essa operação. A Figura 29 ilustra o espectrograma da Figura 27 com $Fs = 16000Hz$. O código para selecionar as amostras de áudio, manipular a frequência de amostragem e gerar espectrogramas está disponível em https://goo.gl/c2nnC3.

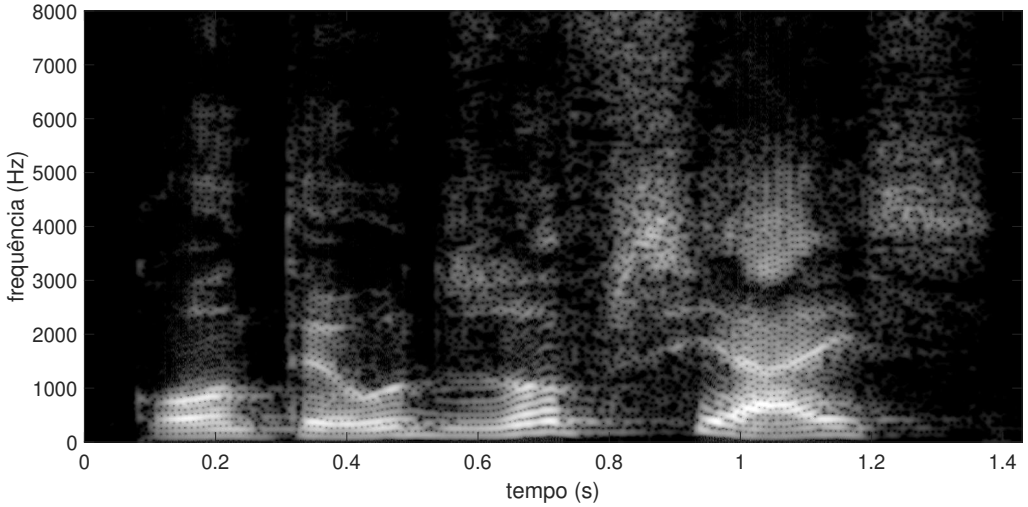



**Figura 29 – Espectrograma do sinal de áudio representado na Figura 27 com $Fs = 16000Hz$**

Os espectrogramas obtidos apresentam valores de frequência fora do intervalo da voz humana que varia aproximadamente entre $50Hz$ e $3400Hz$, então as imagens foram recortadas para apresentar a análise de frequência apenas nesse intervalo, como pode ser visto na Figura 30. Para converter os espectrogramas em imagens os valores de energia do espectro fora do intervalo $-128dB \leq x \leq 0dB$ foram saturados, e em seguida foi realizada uma transformação

linear, $x = (x + 128) * 255/128$, mapeando os valores de energia no intervalo $0 \leq x \leq 255$. Com esse procedimento foi criada uma base de imagens em escala de cinza, disponível em: https://goo.gl/gAtvNN.

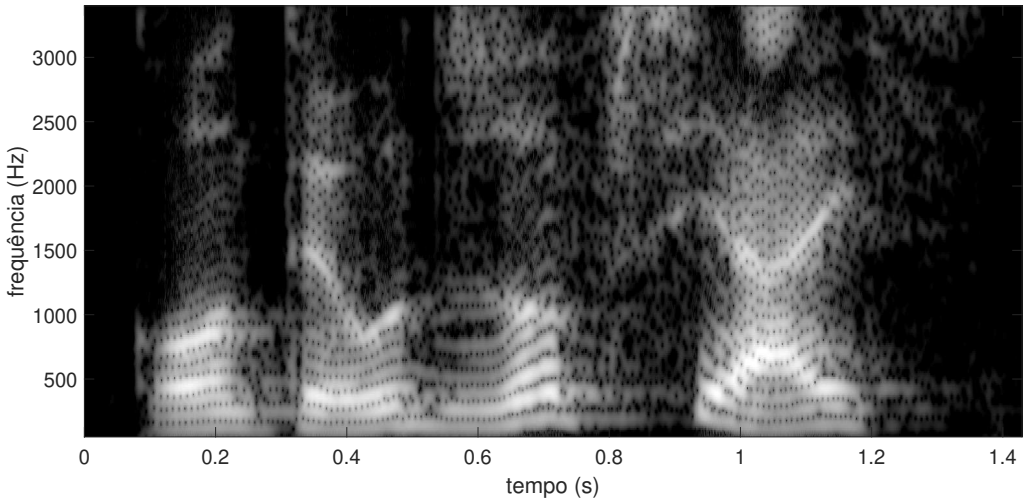



**Figura 30 – Espectrograma do sinal de áudio representado na Figura 27 com** $Fs = 16000Hz$

Tendo em mãos a base de imagens, pode-se aplicar o método $FD_{MIG}$ Gray com as mesmas configurações do método e de validação dos resultados apresentadas na seção 4.3. Portanto, utilizando os parâmetros padrões de $FD_{MIG}$ Gray, validação cruzada 10-fold e normalização z-score, foi obtido um valor de acurácia de 76 % ± 6,3 % com 140 atributos. Esse valor de acurácia indica que o método pode ser suficiente para algumas aplicações de classificações de voz. Por exemplo, aplicações que permitem que vários segundos de áudio sejam classificados sequencialmente, a classificação mais frequente pode ser usada para detectar o interlocutor, aumentando a confiabilidade na classificação.

Não é possível afirmar que o estudo de caso é capaz de classificar a voz presente em cada áudio, uma vez que os sinais de áudio não foram obtidos no mesmo estúdio ou com os mesmos instrumentos, e não há informações se os formatos de armazenamento dos sinais de áudio foram os mesmos antes de realizar o *upload* dos vídeos ao YouTube. Contudo é possível dizer que o estudo de caso consegue identificar a fonte de um sinal de áudio com 76 % ± 6,3 % de acurácia.

# 6 CONCLUSÃO

O presente trabalho desenvolveu um novo método de análise de textura que propõe uma maneira adicional, em relação ao método proposto em (CASANOVA et al., 2016), de realizar a rotulação de píxeis para que sejam utilizados na extração de características de imagens de textura por meio de análise fractal. Os resultados mostraram superar a acurácia de vários dos métodos da literatura estudados nesse trabalho.

Com isso foi mostrado que maneiras alternativas na seleção de rótulos na ATBF pode ser capaz de aumentar a acurácia de classificação de imagens de textura. Dentre as diversas maneiras de realizar a rotulação exploradas no decorrer do desenvolvimento desse trabalho o melhor dos resultados foi apresentado.

Uma contribuição inicial desse trabalho é o método de EEDT para gerar diagramas de Voronoi multirótulo (*VD multirótulo*), que pode ser útil tanto para o objetivo desse trabalho (análise de textura) quanto para outros projetos não necessariamente relacionados com esse objetivo. Como por exemplo o trabalho (MARASCA; CASANOVA; TEIXEIRA, 2018) que usa tais os diagramas de Voronoi multirótulo para realizar estimativas de complexidade de classificação supervisionada de dados. Tal trabalho foi desenvolvido no decorrer desse trabalho aplicando o método de EEDT proposto.

Por fim, salienta-se que tal método de AT pode ser empregado em vários problemas da indústria e da sociedade como um todo. Como exemplo, pode-se citar a análise de peças de manufatura para classificação, tais como madeira (MARASCA; TEIXEIRA; CASANOVA, 2018), ou análise de defeitos, etc.

## REFERÊNCIAS BIBLIOGRÁFICAS

ALATA, O. et al. Choice of a pertinent color space for color texture characterization using parametric spectral analysis. *Pattern Recognition*, Elsevier, v. 44, n. 1, p. 16–31, 2011.

AZENCOTT, R.; WANG, J.-P.; YOUNES, L. Texture classification using windowed fourier filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, v. 19, n. 2, p. 148–153, 1997.

BACKES, A. R. *Implementação e comparação de métodos de estimativa da dimensão fractal e sua aplicação à análise e processamento de imagens*. Dissertação (Mestrado) — University of São Paulo, Brazil, 2006.

BACKES, A. R.; CASANOVA, D.; BRUNO, O. M. Plant leaf identification based on volumetric fractal dimension. *International Journal of Pattern Recognition and Artificial Intelligence*, World Scientific, v. 23, n. 06, p. 1145–1160, 2009.

BACKES, A. R.; CASANOVA, D.; BRUNO, O. M. Color texture analysis based on fractal descriptors. *Pattern Recognition*, v. 45, n. 5, p. 1984–1992, 2012.

BACKES, A. R.; CASANOVA, D.; BRUNO, O. M. Texture analysis and classification: A complex network-based approach. *Information Sciences*, Elsevier, v. 219, p. 168–180, 2013.

BROSNAN, T.; SUN, D.-W. Inspection and grading of agricultural and food products by computer vision systems—a review. *Computers and electronics in agriculture*, Elsevier, v. 36, n. 2, p. 193–213, 2002.

BRUNA, J.; MALLAT, S. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 35, n. 8, p. 1872–1886, 2013.

BURGHOUTS, G. J.; GEUSEBROEK, J.-M. Material-specific adaptation of color invariant features. *Pattern Recognition Letters*, Elsevier, v. 30, n. 3, p. 306–313, 2009.

CANDÈS, E. J. et al. Compressive sampling. In: MADRID, SPAIN. *Proceedings of the international congress of mathematicians*. [S.l.], 2006. v. 3, p. 1433–1452.

CARLIN, M. Measuring the complexity of non-fractal shapes by a fractal method. *Pattern Recognition Letters*, Elsevier, v. 21, n. 11, p. 1013–1017, 2000.

CASANOVA, D. et al. Texture analysis using fractal descriptors estimated by the mutual interference of color channels. *Information Sciences*, Elsevier, v. 346, p. 58–72, 2016.

CAVALCANTI, P. G.; SCHARCANSKI, J. Texture information in melanocytic skin lesion analysis based on standard camera images. In: *Computer Vision Techniques for the Diagnosis of Skin Cancer*. [S.l.]: Springer, 2014. p. 221–242.

CHANG, T.; KUO, C.-C. Texture analysis and classification with tree-structured wavelet transform. *IEEE Transactions on image processing*, IEEE, v. 2, n. 4, p. 429–441, 1993.

CIMPOI, M. et al. Deep filter banks for texture recognition, description, and segmentation. *International Journal of Computer Vision*, Springer, v. 118, n. 1, p. 65–94, 2016.

COSTA, L. d. F.; JR, R. M. C. *Shape classification and analysis: theory and practice*. [S.l.]: Crc Press, 2009.

CROSIER, M.; GRIFFIN, L. D. Using basic image features for texture classification. *International journal of computer vision*, Springer, v. 88, n. 3, p. 447–460, 2010.

CUISENAIRE, O.; MACQ, B. Fast euclidean distance transformation by propagation using multiple neighborhoods. *Computer Vision and Image Understanding*, Elsevier, v. 76, n. 2, p. 163–172, 1999.

DEVORE, J. L. *Probability and Statistics for Engineering and the Sciences*. [S.l.]: Cengage learning, 2011.

ELAD, M. Sparse and redundant representation modeling - what next? *IEEE Signal Process. Lett*, v. 19, n. 12, p. 922–928, 2012.

FABBRI, R. et al. 2D Euclidean distance transform algorithms. *ACM Computing Surveys*, v. 40, n. 1, p. 1–44, 2008.

FLORINDO, J. B. *Descritores fractais aplicados à análise de texturas*. Tese (Doutorado) — Universidade de São Paulo, 2013.

FRITZ, M. et al. *The kth-tips database*. [S.l.]: Citeseer, 2004.

GAIKWAD, S. K.; GAWALI, B. W.; YANNAWAR, P. A review on speech recognition technique. *International Journal of Computer Applications*, International Journal of Computer Applications, 244 5 th Avenue,# 1526, New . . . , v. 10, n. 3, p. 16–24, 2010.

GAO, Y. et al. Compact bilinear pooling. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 317–326.

GOLAN, J. S. *The Linear Algebra a Beginning Graduate Student Ought to Know*. [S.l.]: Springer, 2012.

GUO, Y.; HASTIE, T.; TIBSHIRANI, R. Regularized linear discriminant analysis and its application in microarrays. *Biostatistics*, Oxford University Press, v. 8, n. 1, p. 86–100, 2006.

HARALICK, R. M. Statistical and structural approaches to texture. *Proceedings of the IEEE*, IEEE, v. 67, n. 5, p. 786–804, 1979.

LAZEBNIK, S.; SCHMID, C.; PONCE, J. A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, v. 27, n. 8, p. 1265–1278, 2005.

LIAO, L.; GREGORY, M. Algorithms for speech classification. In: . [S.l.: s.n.], 1999. v. 2, p. 623 – 627 vol.2. ISBN 1-86435-451-8.

LIMA, I.; PINHEIRO, C. A.; SANTOS, F. A. O. *Inteligência artificial*. [S.l.]: Elsevier Brasil, 2016. v. 1.

LIU, L. et al. From bow to cnn: Two decades of texture representation for texture classification. *International Journal of Computer Vision*, Springer, p. 1–36, 2018.

LIU, L. et al. Sorted random projections for robust texture classification. In: IEEE. *Computer Vision (ICCV), 2011 IEEE International Conference on*. [S.l.], 2011. p. 391–398.

LIU, L. et al. Evaluation of lbp and deep texture descriptors with a new robustness benchmark. In: SPRINGER. *European Conference on Computer Vision*. [S.l.], 2016. p. 69–86.

LIU, L. et al. Median robust extended local binary pattern for texture classification. *IEEE Transactions on Image Processing*, IEEE, v. 25, n. 3, p. 1368–1381, 2016.

LOTUFO, R. A.; ZAMPIROLLI, F. A. Fast multidimensional parallel euclidean distance transform based on mathematical morphology. In: IEEE. *Computer Graphics and Image Processing, 2001 Proceedings of XIV Brazilian Symposium on*. [S.l.], 2001. p. 100–105.

MÄENPÄÄ, T.; PIETIKÄINEN, M. Classification with color and texture: jointly or separately? *Pattern recognition*, Elsevier, v. 37, n. 8, p. 1629–1640, 2004.

MALLIKARJUNA, P. et al. The kth-tips2 database. *Computational Vision and Active Perception Laboratory (CVAP), Stockholm, Sweden*, 2006.

MANDELBROT, B. B.; PIGNONI, R. *The fractal geometry of nature*. [S.l.]: WH freeman New York, 1983. v. 173.

MARASCA, A.; CASANOVA, D.; TEIXEIRA, M. Assessing classification complexity of datasets using fractals. *Submetido ao International Journal of Computational Science and Engineering*, 2018. Aceito para publicação.

MARASCA, A.; TEIXEIRA, M.; CASANOVA, D. Utilizando análise de textura baseada em fractais na automação de sistemas de manufatura. *Anais do Computer on the Beach*, p. 701–710, 2018.

MARASCA, A. L.; CASANOVA, D. *Análise de textura através de descritores fractais por meio da utilização de rótulos na transformada de distância 3D*. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2016.

MATERKA, A.; STRZELECKI, M. et al. Texture analysis methods–a review. *Technical university of lodz, institute of electronics, COST B11 report, Brussels*, p. 9–11, 1998.

MEIER, E. *The Wood Database*. 2008. <http://www.wood-database.com/">. Acessado em 08/09/2017.

NG, I.; TAN, T.; KITTLER, J. On local linear transform and gabor filter representation of texture. In: IEEE. *11th IAPR International Conference on Pattern Recognition. Vol. III. Conference C: Image, Speech and Signal Analysis,*. [S.l.], 1992. p. 627–631.

OJALA, T. et al. Outex-new framework for empirical evaluation of texture analysis algorithms. In: IEEE. *Pattern Recognition, 2002. Proceedings. 16th International Conference on*. [S.l.], 2002. v. 1, p. 701–706.

OJALA, T.; PIETIKAINEN, M.; HARWOOD, D. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In: IEEE. *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*. [S.l.], 1994. v. 1, p. 582–585.

OJALA, T.; PIETIKAINEN, M.; MAENPAA, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, IEEE, v. 24, n. 7, p. 971–987, 2002.

PAGLIERONI, D. W. Distance transforms: Properties and machine vision applications. *CVGIP: Graphical models and image processing*, Elsevier, v. 54, n. 1, p. 56–74, 1992.

PIETIKAINEN, M.; ROSENFELD, A. Gray level pyramid linking as an aid in texture analysis. *IEEE TRANS. SYS., MAN, AND CYBER.*, v. 12, n. 3, p. 422–428, 1982.

RODRIGUEZ, J. D.; PEREZ, A.; LOZANO, J. A. Sensitivity analysis of k-fold cross validation in prediction error estimation. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 32, n. 3, p. 569–575, 2010.

RUYI, J. et al. Lane detection and tracking using a new lane model and distance transform. *Machine vision and applications*, Springer, v. 22, n. 4, p. 721–737, 2011.

SAITO, T.; TORIWAKI, J.-I. New algorithms for Euclidean distance transformations of an $n$-dimensional digitized picture with applications. *Pattern Recognition*, v. 27, p. 1551–1565, 1994.

SHAFARENKO, L.; PETROU, M.; KITTLER, J. Automatic watershed segmentation of randomly textured color images. *IEEE transactions on Image Processing*, IEEE, v. 6, n. 11, p. 1530–1544, 1997.

SHI, W.; FAN, X. Speech classification based on cuckoo algorithm and support vector machines. In: *2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA)*. [S.l.: s.n.], 2017. p. 98–102.

SKLANSKY, J. Image segmentation and feature extraction. *IEEE Transactions on Systems, Man, and Cybernetics*, IEEE, v. 8, n. 4, p. 237–247, 1978.

SONG, Y. et al. Locally-transferred fisher vectors for texture classification. In: *ICCV*. [S.l.: s.n.], 2017. p. 4922–4930.

TAMURA, H.; MORI, S.; YAMAWAKI, T. Textural features corresponding to visual perception. *IEEE Transactions on Systems, Man, and Cybernetics*, IEEE, v. 8, n. 6, p. 460–473, 1978.

TECHNOLOGY, M. I. of. *Vision Texture*. 2002. <http://vismod.media.mit.edu/vismod/imagery/VisionTexture/>. Acesso em: 12 dezembro de 2017.

TIMOFTE, R.; GOOL, L. J. V. A training-free classification framework for textures, writers, and materials. In: *BMVC*. [S.l.: s.n.], 2012. v. 13, p. 14.

TRICOT, C. *Curves and fractal dimension*. [S.l.]: Springer Science & Business Media, 1994.

VARMA, M.; ZISSERMAN, A. A statistical approach to material classification using image patch exemplars. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 31, n. 11, p. 2032–2047, 2009.

VISA, S. et al. Confusion matrix-based feature selection. In: *MAICS*. [S.l.: s.n.], 2011. p. 120–127.

WESZKA, J. S.; DYER, C. R.; ROSENFELD, A. A comparative study of texture measures for terrain classification. *IEEE transactions on Systems, Man, and Cybernetics*, IEEE, n. 4, p. 269–285, 1976.

WIKIPéDIA. *Voz humana*. 2012. Disponível em: <https://pt.wikipedia.org/wiki/Voz_humana>.

WOJCICKI, K. *Speech Spectrogram*. 2010. Disponível em: <https://www.mathworks.com/matlabcentral/fileexchange/29596-speech-spectrogram>.

WOLPERT, D. H. Stacked generalization. *Neural networks*, Elsevier, v. 5, n. 2, p. 241–259, 1992.

ZHANG, J. et al. Local features and kernels for classification of texture and object categories: A comprehensive study. *International journal of computer vision*, Springer, v. 73, n. 2, p. 213–238, 2007.