Universidade Federal do Rio Grande do Norte
Centro de Tecnologia
Programa de Pós-Graduação em Engenharia Elétrica e de Computação

Sistema de processamento de imagens para o auxílio na obtenção de Funções de Transferências

Camila Nascimento dos Santos

Orientador: Prof. Dr. Allan de Medeiros Martins

Número de ordem: M525
Natal/RN
Junho de 2018

Sistema de processamento de imagens para o auxílio na obtenção de Funções de Transferências

Camila Nascimento dos Santos

Dissertação de Mestrado apresentada ao corpo docente do Programa de Pós-Graduação em Engenharia Elétrica da UFRN como parte dos requisitos para obtenção do título de Mestre em Ciências.

Natal/RN
Junho de 2018

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISBI
Catalogação de Publicação na Fonte. UFRN - Biblioteca Central Zila Mamede

Santos, Camila Nascimento dos.

Sistema de processamento de imagens para o auxílio na obtenção de funções de transferências / Camila Nascimento dos Santos. - 2018.

70 f.: il.

Dissertação (mestrado) - Universidade Federal do Rio Grande do Norte, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica e de Computação. Natal, RN, 2018.

Orientador: Prof. Dr. Allan de Medeiros Martins.

1. Sistema de controle - Dissertação. 2. Função de Transferência - Dissertação. 3. Fórmula de Ganho de Mason - Dissertação. 4. Sistema de processamento de imagens - Dissertação. 5. Segmentação de imagens - Dissertação. 6. Visão Computacional - Dissertação. I. Martins, Allan de Medeiros. II. Título.

RN/UF/BCZM CDU 681.5(043)

Elaborado por Kalline Bezerra da Silva - CRB-15/327

Universidade Federal do Rio Grande do Norte

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

ATA Nº 525

Aos 28 dia do mês de junho do ano de dois mil e dezoito, foi realizada a 525ª sessão de defesa de dissertação de mestrado do Programa de Pós-Graduação em Engenharia Elétrica e de Computação da UFRN, na qual o mestrando Camila Nascimento dos Santos apresentou o trabalho que tem como título:Sistema de Processamento de Imagens para auxílio na obtenção de Funções de Transferência. A sessão teve início às 09h00min, tendo a banca examinadora sido constituída pelos seguintes participantes: Allan de Medeiros Martins (Dr. UFRN, Presidente), Pablo Javier Alsina (Dr. UFRN, Examinador Interno ao Programa), Agostinho De Medeiros Brito Junior (Dr. UFRN, Examinador Externo ao Programa) e Aluisio Igor Rego Fontes (Dr. IRFN, Examinador Externo à Instituição). Após a apresentação do trabalho e o exame pela banca, o mestrando foi considerado APROVADO, tendo sido lavrada a presente ata, que vai assinada pelos examinadores e pelo mestrando. A versão final da dissertação deverá ser entregue ao programa, no prazo máximo de 60 dias, contendo as modificações sugeridas pela banca examinadora e constante na folha de correção anexa. Conforme o Artigo 49 da Resolução 197/2013 - CONSEPE, o candidato não terá o título se não cumprir as exigências acima.

Aluisio Igor R. Fontes

Dr. ALUISIO IGOR REGO FONTES, IFRN

Examinador Externo à Instituição

Agostinho Brito

Dr. AGOSTINHO DE MEDEIROS BRITO JUNIOR, UFRN

Examinador Externo ao Programa

Pablo Javier Alsina

Dr. PABLO JAVIER ALSINA, UFRN

Examinador Interno

Dr. ALLAN DE MEDEIROS MARTINS, UFRN

Presidente

Camila Nascimento dos Santos

CAMILA NASCIMENTO DOS SANTOS

Mestrando

Dedico este trabalho primeiramente a Deus, a minha família, em especial aos meus pais e a minha irmã, pela compreensão e cuidado durante toda a minha vida. Ao meu esposo pelo incentivo e apoio durante este trabalho. Vocês foram o meu porto seguro na presença das dificuldades.

Agradecimentos

Agradeço primeiramente de todo meu coração aos meus familiares, meus pais Nilson Santos e Vilma Santos, e a minha irmã Kaline, que estiveram sempre presentes, me apoiando e me dando forças para eu continuar longo de toda a minha vida.

Ao meu marido, Thiago Farias, por sempre cuidar de mim e estar ao meu lado, me proporcionando sempre os melhores momentos.

Ao meu professor orientador Allan Medeiros Martins, sou grata por toda a ajuda e paciência ao decorrer deste trabalho.

Gostaria de agradecer também a minha grande amiga Taiane Viana de Carvalho, sou muito grata por todo o apoio e ajuda, sem a qual este trabalho não existiria.

Por fim, agradeço aos meus amigos e a todos que, de forma direta ou indireta, contribuíram para este trabalho, muito obrigada!

Resumo

Várias atividades cotidianas utilizam sistemas de controles e eles podem ser modelados e representados pela interconexão de diversos subsistemas, fornecendo uma melhor compreensão da composição de seus componentes. Estes ao serem usados em conjunto com Funções de Transferência (FT), descrevem também as relações de causa e efeito de todo o sistema.

Os diagramas de fluxo de sinal são grafos direcionados usados para representar a relação causal entre componentes do sistema, em que seus nós e arestas representam sinais e FT, respectivamente. Para calcular o ganho final de um diagrama, um método bastante utilizado é a fórmula de ganho de Mason.

Este método requer que o usuário encontre todos os caminhos diretos e *loops* existentes no grafo. Todo esse processamento possui uma complexidade relevante, por envolver análise visual e manipulações matemáticas, tornando este método susceptível a erros.

Dessa forma, o objetivo deste trabalho é descrever o desenvolvimento e implementação de um sistema de processamento de imagem, capaz de processar a imagem de um grafo de fluxo de sinais desenhado em um papel ou quadro branco e executar todas as etapas necessárias, até encontrar todos os caminhos presentes no mesmo e a FT equivalente. Com o objetivo de validar essa abordagem e avaliar sua precisão, foram obtidas 117 amostras de grafos de fluxo de sinal com universitários do curso de Engenharia Elétrica da UFRN.

As principais contribuições propostas neste trabalho consistem em auxiliar tanto a alunos como também a profissionais em calcular a FT final com uma probabilidade superior de acerto, como também agilizar esse processo.

Palavras-chave: Função de Transferência, Fórmula de Ganho de Mason, Processamento de Imagens, Segmentação de Imagens, Visão Computacional

Abstract

Several daily activities utilize control systems and they can be modeled and represented by the interconnection of several subsystems, providing a better understanding of the composition of their components. These, when used in conjunction with Transfer Functions (FT), also describe the cause and effect relationships of the whole system.

Signal flow diagrams are directed graphs used to represent the causal relationship between system components, where their nodes and edges represent signals and FT, respectively. To calculate the final gain of a diagram, a widely used method is the Mason gain formula.

This method requires the user to find all the direct paths and loops in the graph. All this processing has a relevant complexity, since it involves visual analysis and mathematical manipulations, making this method susceptible to errors.

Thus, the aim of this work is to describe the development and implementation of an image processing system, capable of processing the image of a signal flow graph drawn on a paper or whiteboard and performing all necessary steps, until find all the paths present in it and the equivalent FT. In order to validate this approach and evaluate its accuracy, we obtained 117 samples of signal flow graphs with university students of the Electrical Engineering course at UFRN.

The main contributions proposed in this work are to help both students and professionals in calculating the final FT with a higher probability of success, as well as to streamline this process.

Keywords: Transfer Function, Mason's Gain Formula, Image Processing, Image Segmentation, Computer vision

Sumário

Sumário i

Lista de Figuras iii

Lista de Tabelas v

Lista de Abreviaturas vi

Lista de Algoritmos vii

1 Introdução 1

1.1 Motivação . . . 3

1.2 Objetivo . . . 3

1.3 Organização do Trabalho . . . 4

2 Fundamentação Teórica 6

2.1 Fórmula de Mason . . . 6

2.2 Processamento Digital de Imagens . . . 7

2.2.1 Aquisição da imagem . . . 8

2.2.2 Pré-Processamento . . . 10

2.2.3 Segmentação . . . 11

2.2.4 Pós-processamento . . . 13

2.2.5 Extração de atributos . . . 16

2.2.6 Classificação e Reconhecimento . . . 16

3 Abordagem Proposta 18

3.1 Algoritmo Geral . . . 18

3.1.1 Segmentação da Imagem . . . 18

3.1.2 Determinar a posição espacial dos nós . . . 20

3.1.3 Classificação dos nós relevantes . . . 22

3.1.4 Conexão dos nós . . . 24

3.1.5 Detecção das direções das arestas 26
3.1.6 Encontrando os caminhos diretos 27
3.1.7 Encontrando os loops 29
3.1.8 Calculo da Função de Transferência 30
3.2 Exigências do Software 30

4 Testes e Resultados **32**
4.1 Testes 32
4.1.1 Distância Mínima 32
4.2 Resultados 35
4.2.1 Dados coletados 35
4.2.2 Descrição dos Resultados 36
4.2.3 Considerações 50

5 Conclusão **53**

Referências **55**

Lista de Figuras

2.1 Exemplo de um Diagrama de fluxo de sinal, na qual a sua função de transferência final é: $G(s) = G_1(s)G_2(s)G_3(s) + G_4(s)G_3(s) + G_1(s)G_5(s)$ 7
2.2 Etapas de um sistema PDI 9
2.3 Estrutura da vizinhança de 8, usada pelo algoritmo de esqueletização 15
2.4 Exemplo das condições (i) e (ii) 16

3.1 Diagrama das etapas do algoritmo geral proposto 19
3.2 Fluxograma das etapas de segmentação da imagem 20
3.3 Exemplo do esqueleto da imagem e dos *pixels* candidatos a nós 21
3.4 Plot das distâncias e suas diferenças 24
3.5 Ilustração da classificação dos nós como arestas ou bifurcações 25
3.6 Resultado da etapa de classificação dos nós, em que o nó inicial contém um, dois e três ramos partindo dele 31

4.1 Conjunto de dados utilizados para a realização de testes no cálculo da distância mínima 33
4.2 Resultados gerados na etapa de determinação da posição e classificação dos nós utilizando a imagem (1) 34
4.3 Resultados gerados na etapa de determinação da posição e classificação dos nós utilizando a imagem (2) 34
4.4 Resultados gerados na etapa de classificação dos nós utilizando a imagem (3) 35
4.5 Desenhos utilizados para a obtenção dos resultados 36
4.6 Total de amostras coletadas para cada tipo de diagrama 36
4.7 Amostras consideradas com erros de desenho 38
4.8 Amostras consideradas com erros de desenho 39
4.9 Gráfico com a média da quantidade de *pixels* candidatos encontrados por tipo de amostras 41

4.10 Amostras coletadas e os resultados da quantização e classificação dos grafos, respectivamente 42
4.11 Resultados obtidos do grafo 1 do conjunto de dados do tipo (a) . . . 43
4.12 Imagem e Resultado da quantização e classificação do grafo 12 44
4.13 Imagem e Resultado da quantização e classificação do grafo 27 45
4.14 Amostras coletadas 48
4.15 Resultado da quantização e classificação dos grafos 49 e 58, respectivamente 48
4.16 Imagem e Resultado da quantização e classificação do grafo 73 50
4.17 Imagem e Resultado da quantização e classificação do grafo 95 51
4.18 Gráfico com o quantitativo de resultados encontrados por tipo de amostras 52

Lista de Tabelas

4.1 Resumo da quantidade de cada tipo de erro pela categoria de diagrama 39
4.2 Mínimo e Máximo da constante k utilizada no algoritmo de segmentação 40
4.3 Resumo das características encontradas para cada amostra do conjunto do tipo (a) 41
4.4 Resumo das características encontradas para cada amostra do conjunto do tipo (b) 44
4.5 Resumo das características encontradas para cada amostra do conjunto do tipo (c) 46
4.6 Resumo das características encontradas para cada amostra do conjunto do tipo (d) 47
4.7 Resumo das características encontradas para cada amostra do conjunto do tipo (e) 47
4.8 Resumo das características encontradas para cada amostra do conjunto do tipo (f) 49
4.9 Resumo das características encontradas para cada amostra do conjunto do tipo (g) 49
4.10 Resumo das características encontradas para cada amostra do conjunto do tipo (h) 50
4.11 Tempo médio em segundos 52

Lista de Abreviaturas

FT: Função de Transferência

MAT: *Medial Axis Transformation*

MGF: Mason's Gain Formula

PDI: Processamento Digital de Imagem

Lista de Algoritmos

1 Pseudocódigo do algoritmo proposto por Sauvola 13
2 Algoritmo para calcular a distância mínima utilizada na quantização 23
3 Algoritmo para classificação dos nós 25
4 Algoritmo para encontrar a conexão dos nós 26
5 Algoritmo para detectar as direções das arestas 27
6 Algoritmo para encontrar os caminhos diretos 28
7 Algoritmo para encontrar os laços 29

Capítulo 1

Introdução

Os sistemas de controle estão presentes em inúmeras aplicações do dia a dia. Um exemplo disso é o resfriamento de foguetes, o funcionamento de máquinas industriais, veículos autônomos e elevadores prediais (NISE, 2007). Os sistemas de controle complexos podem ser modelados e representados pela interconexão de diversos subsistemas. Isto fornece uma melhor compreensão da composição de seus componentes. Estes ao serem usados em conjunto com funções de transferência (FT), descrevem também as relações de causa e efeito de todo o sistema (DORF; BISHOP, 2011). Uma vez que a resposta de uma única FT foi calculada, é possível obter informações da resposta transitória relativa ao sistema como um todo.

Um método bastante utilizado para a obtenção das FT's é chamado de Fórmula de Ganho de Mason (MGF) (MASON, 1956), um algoritmo clássico e eficiente, que utiliza um grafo direcionado conhecido como diagrama de fluxo de sinal, na qual as FT's são representadas como arestas e os sinais como pequenos nós circulares (NISE, 2007).

Nasir (2014) utilizou a MGF para a obtenção da FT de reguladores lineares digitais operando no modo *low dropout*, em uma rede de distribuição de energia, com o objetivo de analisar a sua estabilidade geral e desenvolver um modelo de controle híbrido de todos os componentes necessários. Já Mohammadian (2015) focou na modelagem do conversor DC-DC e utilizou a MGF para obtenção da FT desse sistema e assim projetar um controlador PID para um conversor buck-boost.

Kotzapetros (2015) utilizou a MGF na geração das equações características de dois sistemas que utilizaram diferentes controladores e com isso fez a comparação da robustez e eficiência dos mesmos. Por fim, Cao (2015) apresentou uma análise em que utiliza a MGF para inferir quatro modos básicos de métodos de modelagem de impedância de saída, usados em sistemas de energia renováveis, com o objetivo de minimizar os impactos da corrente harmônica de baixa frequência na vida útil das

fontes de energia sensíveis a ondulação.

A principal vantagem na utilização da MGF é que ele é um procedimento direto em que não é necessário mover nenhum ponto em volta ou redesenhar o sistema várias vezes como com manipulações de diagramas de bloco.

Contudo, para a obtenção da FT utilizando este método, é necessário encontrar todos os caminhos diretos e os *loops* presentes no grafo (OGATA, 2002). Todo esse processamento possui uma complexidade relevante, por envolver análise visual e manipulações matemáticas, tornando este método susceptível a erros.

Estes problemas podem ser contornados com o uso de um software especifico. Porém, desenhar no computador não é uma tarefa simples, a interface tradicional do usuário não é flexível e pode dificultar a representação dos contornos, o que exige mais habilidade e tempo. Além disso, com o surgimento de novas tecnologias, como *tablets* e telefones celulares, o uso de algoritmos de processamento de imagem para lidar com a entrada do usuário tornou-se bastante comum e atraente (STAHOVICH, 2004; SZUMMER; QI, 2004; GENNARI et al., 2005; KARA; STAHOVICH, 2005; OUYANG; DAVIS, 2007; RABBANI et al., 2016).

Todavia, ao utilizar tecnologias digitais, o sistema requer métodos sólidos de análise de imagens que permitam reconhecer a estrutura e detectar áreas de interesse, facilitando a visualização da imagem e adequá-la para análises quantitativas através da correções de defeitos ou realces das regiões de interesse nas imagens (TRIER; TAXT, 1995).

A simulação por meio do computador deste processo caracteriza os chamados sistemas de visão computacional, cuja aplicabilidade permeia diversos problemas, tais como visão robótica, interface homem-máquina inteligentes, recuperação de informação em bancos de dados multimídia, análise de imagens médicas, segurança e vigilância automatizadas, sensoriamento remoto, entre outros.

Em 2000, Oliver et. al apresentou um sistema para modelar e reconhecer comportamentos humanos em uma tarefa de vigilância visual, em que ele observa e classifica quando ocorrem interações entre pessoas.

Messelodi (2005) desenvolveu um sistema que analisa sequências de imagens monoculares provenientes de câmeras de vídeo, capaz de descrever o caminho de cada veículo detectado, estimar sua velocidade e classificá-lo. Já Kara (2005) apresentou o desenvolvimento de um sistema que reconhece símbolos desenhados a mão, capaz de aprender novas definições a partir de protótipos. Por outro lado, Zhang (2010) propôs um método de reconhecimento facial capaz de ter um bom poder de representação, isto é, ser capaz de reconhecer a face de diferentes humanos em várias

posições.

Casanova (2012) descreveu uma metodologia capaz de reconhecer e representar a linha mediana cerebral, com o objetivo de utilizá-la em diagnósticos de traumatismos cranianos. Em contrapartida, em 2016, Rabbani propôs um método capaz de reconhecer os símbolos elétricos a partir da imagem de um circuito desenhado a mão.

Embora os sistemas atuais de visão computacional sejam limitados em relação ao sistema visual humano, seu projeto e funcionamento também apresentam um elevado grau de complexidade, englobando diversas etapas, desde a aquisição da imagem, passando pelo armazenamento, processamento, segmentação e extração de informações relevantes (GONZALEZ; WOODS, 2000). Dessa forma, o principal objetivo dessas aplicações é segmentar a imagem, extrair informações uteis e utiliza-las de acordo com sua aplicação.

1.1 Motivação

Muitos métodos foram propostos nos últimos anos na área de processamento de imagens, com o intuito de resolver algum problema cotidiano ou até mesmo deixa-lo mais rápido e com a perspectiva de acerto maior. Contudo, ainda há muitos problemas a serem solucionados.

Como foi exposto acima, a fórmula de ganho de Mason ainda é muito utilizada em diversas áreas, como principal forma para obtenção da FT. Porém nenhum trabalho explora a fragilidade deste método, pois quanto maior o sistema, maior a probabilidade de não encontrar um caminho. O que torna esta área propícia para novas pesquisas.

1.2 Objetivo

Dentro do contexto exposto acima, o método proposto neste trabalho tem o objetivo de projetar e implementar um sistema de processamento de imagens, capaz de processar a imagem de um grafo de fluxo de sinais desenhado em um papel ou quadro branco e executar todas as etapas necessárias, até encontrar todos os caminhos presentes no mesmo e a FT equivalente.

As principais contribuições propostas neste trabalho consistem em auxiliar tanto a alunos como também a profissionais em calcular a FT final com uma probabilidade

superior de acerto, como também agilizar esse processo.

Para atingir o objetivo proposto, foram utilizadas algumas técnicas de pré-processamento, segmentação e pós-processamento. Como também foram desenvolvidas várias técnicas capazes de realizar a extração de elementos fundamentais a ponto de encontrar todos os caminhos presentes no grafo e calcular a FT equivalente corretamente.

Sendo assim, a ideia principal foi inicialmente utilizar o filtro morfológico da mediana, para suavização e/ou remoção dos ruídos, segmentar a imagem, utilizando o método de segmentação proposto por Sauvola et. al (2000), em seguida determinar a posição espacial dos nós, classificá-los em bifurcação, arestas, nó inicial ou final, para então encontrar os vizinhos de cada um. Com o grafo montado, foi necessário detectar a direção das arestas, para posteriormente encontrar os caminhos diretos e *loops*, e finalmente calcular a FT equivalente. No decorrer do texto, serão explicados cada etapa, seus problemas e como estes foram resolvidos.

Este trabalho permitirá agregar conhecimento tanto na área de controle como também na de processamento de imagens, além de uma nova ferramenta. Isso poderá contribuir com futuros trabalhos na área de visão computacional, onde pode ser explorado o desenvolvimento desse sistema em uma aplicação mobile ou até mesmo para web. Além disso, as técnicas propostas aqui podem ser exploradas e modificadas para serem utilizadas em alguma outra aplicação específica.

1.3 Organização do Trabalho

Neste capítulo, foram discutidos o cenário, os objetivos e as contribuições da metodologia desenvolvida para detecção dos caminhos diretos e os *loops* presentes em um grafo de fluxo de sinais, para então ser capaz de calcular o ganho final do sistema corretamente.

Uma revisão bibliográfica abrangendo trabalhos correlatos e uma discussão sobre a fórmula de ganho de Mason e processamento de imagens, se aprofundando no método de segmentação utilizado nesse trabalho é apresentado no capítulo 2.

Em seguida, o capítulo 3 consagra-se à apresentação de todos os passos necessários para a implementação do método, descrevendo aspectos teóricos e práticos, como também as restrições impostas pelo sistema, solucionando algumas dificuldades encontradas.

No capítulo 4 serão exibidos alguns testes realizados no sistema desenvolvido, como também será apresentado o conjunto de dados coletados e os resultados obti-

dos. Finalmente, o capítulo 5 é constituído pela conclusão.

Capítulo 2

Fundamentação Teórica

Nesse capítulo encontra-se detalhada a teoria necessária para possibilitar a realização deste trabalho. Ele está dividido em duas partes principais. A primeira contempla a Seção 2.1, que descreve a teoria de diagramas de fluxo de sinal e como, a partir da fórmula de Mason, ele pode ser reduzido a um único ramo entre os nós de interesse. A segunda contempla a Seção 2.2, a qual apresenta alguns conceitos fundamentais de Processamento Digital de Imagens (PDI), procurando introduzir o formalismo e as notações que serão utilizados nos capítulos posteriores.

2.1 Fórmula de Mason

É comum representar sistemas mais complexos como um conjunto de interconexões de diversos subsistemas (NISE, 2007). Estes podem ser representados de duas maneiras: diagramas de blocos e diagramas de fluxo de sinal. Embora nenhuma dessas representações seja restrita a uma técnica específica, o primeiro geralmente é utilizado para análise e projeto no domínio da frequência e o segundo para análise no espaço de estados.

Os diagramas de blocos consistem em blocos operacionais unidirecionais que representam a função de transferência das variáveis de interesse (DORF; BISHOP, 2011).

O diagrama do fluxo de sinal são grafos direcionados que consistem de uma rede em que os nós são conectados por ramificações direcionadas (OGATA, 2002). Ele representa o fluxo de sinais de um ponto do sistema para outro, onde cada nó é uma conexão (ou bifurcação) e cada aresta é um pequeno subsistema que tem uma função de transferência simples e local.

Neste cenário, o objetivo é representar subsistemas múltiplos através de uma única FT, ou seja, encontrar o "ganho final" a partir de um nó inicial particular para

um outro nó final, denominado de FT equivalente.

A Figura 2.1 mostra um exemplo de grafo e sua FT final. Cada aresta desta imagem, tem uma seta para estabelecer a direção do fluxo de sinal, permitindo a presença de *loops* no diagrama que devem ser levados em consideração.

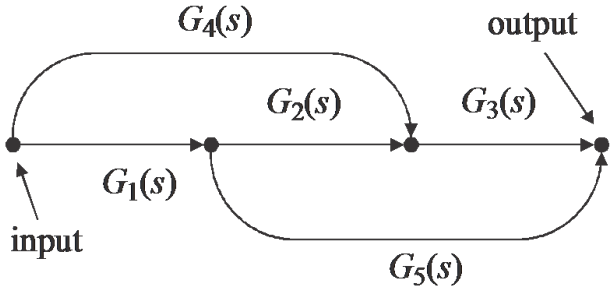


Figura 2.1: Exemplo de um Diagrama de fluxo de sinal, na qual a sua função de transferência final é: $G(s) = G_1(s)G_2(s)G_3(s) + G_4(s)G_3(s) + G_1(s)G_5(s)$

Para o calculo da FT equivalente utiliza-se a fórmula de mason mostrada na Equação 2.1 (MASON, 1956).

$$G = \frac{1}{\Delta}\sum_{i=1}^{N}\Delta_i P_i \tag{2.1}$$

Na Equação 2.1, Δ é o determinante do diagrama dado pela Equação 2.2:

$$\Delta = 1 - \sum_i L_i - \sum_i\sum_j L_i L_j - \sum_i\sum_j\sum_k L_i L_j L_k \cdots \tag{2.2}$$

Os termos L_i's são os ganhos do *loop* e a soma é feita somente com aqueles que não tem nenhuma aresta em comum. P_i's são os ganhos dos caminhos diretos, ou seja, qualquer caminho que vá a partir do nó inicial para o nó final sem nenhum *loop*. Por exemplo, na Figura 2.1, temos três caminhos diretos, apresentados pelos termos da soma. Já Δ_i é calculado como um determinante, mas extraindo todos os *loops* que tem qualquer aresta em comum com os respectivos P_i's.

2.2 Processamento Digital de Imagens

O processamento digital de imagem (PDI) é uma área da eletrônica/teoria de sinais em que imagens são convertidas em uma função bidimensional, $f(x, y)$, onde x e y são coordenadas espaciais que por convenção: $x = [1, 2, \cdots, M]$ e $y = [1, 2, \cdots, N]$. O valor de f no ponto (x, y) é proporcional ao brilho (ou nível de cinza) da imagem

neste ponto, também chamado de *pixel* (abreviação, do Inglês: *picture element*). A partir desta matriz de *pixels* diversos tipos de processamento digital podem ser implementados por algoritmos computacionais. A aplicação destes algoritmos realiza as transformações necessárias para que se possa, por exemplo, obter uma imagem com os realces pretendidos ou extrair atributos e informações pertinentes (GONZALEZ; WOODS, 2000).

O objetivo do processamento de imagens é facilitar a visualização da imagem ou adequá-la para análises quantitativas através de correções de defeitos ou realces das regiões de interesse nas imagens, como também a extração e tratamento de dados quantitativos, feitos pelo próprio computador.

Segundo Mascarenhas e Velasco (1989), para que uma imagem possa ser manipulada por um computador, é necessário que ela seja discretizada ou digitalizada, tanto espacialmente quanto em sua amplitude. Com isso, uma imagem pode ser discretizada através de um processo que envolve dois passos:

1. *Amostragem*: na qual são definidos os valores para as coordenadas x e y da matriz de pontos;

2. *Quantização*: consiste em escolher um valor múltiplo relacionado à banda de frequência da intensidade para cada ponto da imagem.

Alguns autores possuem conceitos diferentes quanto à taxonomia do processo de um sistema PDI. Entretanto, todos seguem uma mesma linha que, neste caso, segue a classificação segundo Gonzalez e Woods (2000), ilustrado na Figura 2.2.

Na Figura 2.2, as setas ao lado do fluxograma indicam os níveis com que se trabalha em cada etapa. No pré-processamento e na segmentação, as operações são feitas nos *pixels* das imagens, resultando em imagens com objetos ou regiões de *pixels* contíguos de valores iguais. As medidas para a geração de dados são feitas na extração de atributos e estas medidas são utilizadas no fim do processo, na etapa de reconhecimento de padrões e classificação, de forma a obter dados mais complexos. Nas subsecções a seguir são apresentadas breves descrições de cada uma destas etapas.

2.2.1 Aquisição da imagem

A etapa de aquisição da imagem tem como função converter uma cena real tridimensional em uma representação numérica adequada para o processamento digital subsequente (FILHO; NETO, 1999).

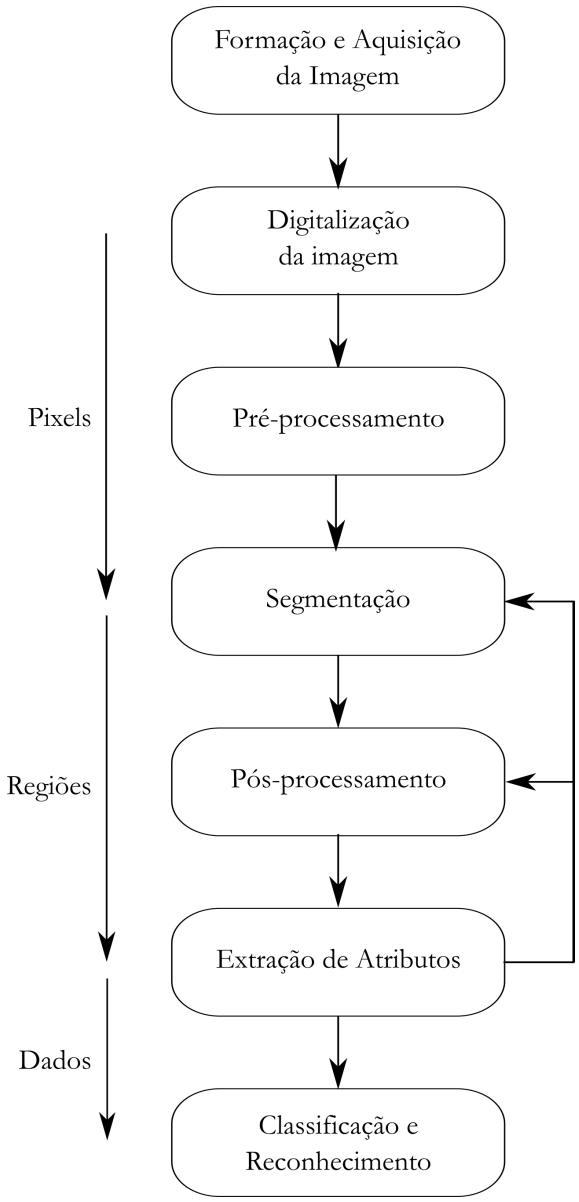


Figura 2.2: Etapas de um sistema PDI

Dois elementos são necessários para a aquisição digital de imagens. O primeiro é um dispositivo físico, que deve ser sensível ao espectro de energia eletromagnética, como por exemplo, ao espectro de raio-x, luz ultravioleta, visível ou infravermelha. Este dispositivo transdutor deve produzir em sua saída um sinal elétrico proporcional ao nível de energia percebido. O segundo, chamado digitalizador, é um dispositivo que converte o sinal elétrico analógico produzido na saída do sensor, em um sinal digital.

2.2.2 Pré-Processamento

A etapa de pré-processamento tem como o objetivo corrigir alguns defeitos e imperfeições provenientes da aquisição da imagem, que podem ter como causa características físicas do sistema, condições deficientes de iluminação, entre outros. Para que a etapa de segmentação tenha resultados satisfatórios, é necessário que a imagem esteja com o mínimo de imperfeições, daí a importância da etapa de pré-processamento.

Esta etapa é caracterizada por soluções específicas, de modo que as técnicas que funcionam bem para um tipo de problema pode se mostrar totalmente inadequadas para outro (GONZALEZ; WOODS, 2000).

Estas técnicas envolvem duas categorias principais: métodos que operam no domínio espacial e no domínio da frequência (ESQUEF, 2002). A primeira baseia-se em filtros que manipulam o plano da imagem, enquanto a segunda se baseia em filtros que agem sobre o espectro da imagem.

Como exemplo de procedimentos eventualmente utilizados, podem ser citados: melhoras no brilho e no contraste, redução de ruídos, correção de iluminação irregular, realce de bordas, entre outros.

2.2.2.1 Filtro da mediana

O filtro da mediana é utilizado para suavização e eliminação do ruído da imagem e seu objetivo é substituir o nível de cinza de cada *pixel*, pela mediana dos níveis de cinza da sua vizinhança, usando uma janela de tamanho $H \times H$ (GONZALEZ; WOODS, 2000). Em geral, este método é eficiente para ruídos do tipo pulso, sendo que a característica a ser preservada é a agudeza das bordas. Se o ruído em uma imagem aparece como erro aditivo, aleatório ou não correlacionado, os *pixels* afetados são corrigidos, com a vantagem de evitar o espalhamento do ruído para os vizinhos.

Para calcular a filtragem por mediana de um *pixel*, primeiramente é selecionado seu valor e de seus vizinhos, para então determinar a mediana e atribuir este valor ao *pixel*. Por exemplo, se for utilizado uma vizinhança com a janela do tamanho 3×3, a mediana é o 5° maior valor. Suponhamos que essa vizinhança tenha os valores $(10, 20, 20, 20, 15, 20, 20, 25, 100)$, então eles ordenados ficam como $(10, 15, 20, 20, 20, 20, 20, 25, 100)$ em que resulta a mediana igual a 20. Assim, a função principal da filtragem por mediana é forçar pontos com intensidades distintas a assemelharem-se a seus vizinhos, eliminando efetivamente ruídos de intensidade que apareçam isolados na área da máscara do filtro.

2.2.3 Segmentação

A segmentação subdivide uma imagem em regiões ou objetos que a compõem (GONZALEZ; WOODS, 2000). Digamos que E represente toda a região espacial ocupada por uma imagem. Pode-se ver a segmentação da imagem como um processo que particiona E em n sub-regiões, E_1, E_2, ..., E_n, de tal forma que:

$\bigcup_{i=1}^{n} E_i = E$
E_i é um conjunto conectado, $i = 1, 2, \cdots, n$
$E_i \cap E_j = \phi$ para todo i e j, $i \neq j$
$P(E_i) = VERDADEIRO$ para $i = 1, 2, \cdots, n$
$P(E_i \cup E_j) = FALSA$ para quaisquer regiões adjacentes E_i e E_j

Em que $P(E_i)$ é um predicado lógico sobre os pontos do conjunto E_i e ϕ é o conjunto vazio.

A imagem de saída da segmentação é uma imagem binária, onde os *pixels* pretos representam o fundo ou objetos que não são de interesse na imagem, e os *pixels* brancos constituem os objetos de interesse.

A segmentação é considerada, dentre todas as etapas do PDI, a mais crítica do tratamento da informação. É nesta etapa que são definidas as regiões de interesse para processamento e análise posteriores. Como consequência, quaisquer erros ou distorções presentes nesta etapa refletem-se nas demais, de forma a produzir ao final do processo resultados não desejados, podendo contribuir de forma negativa para a eficiência de todo o processamento (ESQUEF, 2002).

Diversas técnicas de binarização já foram sugeridas, definindo basicamente dois tipos de abordagem: global, onde o método define um único ponto de corte para toda a imagem, e o local, onde a imagem é dividida em regiões e a binarização é realizada considerando características de cada região (como média e desvio padrão, por exemplo). As técnicas de binarização local são adaptativas, levando em consideração mudanças de luminosidade, por exemplo, ao longo da imagem. Essas técnicas são, em geral, as que apresentam melhor resultado na binarização. Entretanto, são as que demandam mais tempo e processamento para serem realizadas

Embora haja muitos métodos propostos na literatura (SEZGIN; SANKUR, 2004), um grande problema que dificulta a binarização e que a torna um estudo em aberto é que as diversas técnicas de binarização existentes têm desempenhos diferentes para os variados conjuntos de imagens, não havendo ainda uma técnica de binarização geral que tenha um desempenho ótimo para todos os tipos de imagens e seus problemas específicos. Dessa forma, procurando se adequar às características particulares

de cada tipo de imagem e aos objetivos que pretendemos alcançar, após diferentes testes, o método que apresentou melhores resultados e obteve um ótimo desempenho para diversos tipos de imagens foi o proposto por Savoula (2000), apresentado a seguir.

2.2.3.1 Algoritmo proposto por Sauvola

A limiarização adaptativa caracteriza-se por calcular diferentes limiares para cada *pixel* da imagem, através da análise das intensidades dos níveis de cinza dentro de janelas deslizantes.

A técnica de limiarização adaptativa apresentada por Sauvola (2000) calcula um limiar local, utilizando o deslocamento de uma janela retangular, com tamanho $W \times W$, pela imagem. O limiar $T(x, y)$ é calculado para cada *pixel* (x, y) localizado no centro da janela e tem como base as estatísticas locais, como a média, $m(x, y)$, e o desvio padrão, $s(x, y)$, da intensidade do níveis de cinza presentes na janela, tal como exibido na Equação 2.3.

$$T(x, y) = m(x, y) \cdot \left[1 + k \cdot \left(\frac{s(x, y)}{R} - 1\right)\right] \tag{2.3}$$

Na qual R é o *range* dinâmico do desvio padrão, que foi sugerido por Sauvola em seus experimentos práticos como 128 para imagens em escala de cinza e k é uma constante positiva. Nesse método, o valor do limiar é adaptado com a média e o desvio padrão local, de acordo com o contraste da vizinhança local do *pixel*. A constante k controla o nível de adaptação do limiar $T(x, y)$, variando o seu valor (SENTHILKUMARAN; VAITHEGI, 2016).

O algoritmo de Sauvola busca melhorar os resultados da binarização de imagens que foram adquiridas em más condições de iluminação, e que apresentam altas taxas de ruídos. Desse modo, os valores ótimos de R e k são proporcionais ao contraste da imagem, de forma que, para imagens com baixo contraste, se os valores não forem definidos corretamente, pode acarretar em uma perda das regiões de interesse (ARRUDA, 2014).

No Algoritmo 1, é exibido o pseudocódigo do algoritmo proposto por Sauvola, onde inicialmente o usuário entra com a imagem, os valores das constantes e a dimensão da janela, W, com isso, o algoritmo retorna a imagem binarizada.

Algoritmo 1 Pseudocódigo do algoritmo proposto por Sauvola

Entrada: Im: Imagem
W: O tamanho da janela
k: Valor utilizado para a constante k
R: Valor utilizado para a constante R
Saída: $Im_binarizada$: Imagem binarizada
início
  para cada *pixel* **faça**
    w = Definir a janela com o tamanho W centrada no *pixel*;
    m = Média dos *pixels* inclusos em w;
    $desvioPadrao$ = Desvio padrão dos *pixels* inclusos em w;
    $T = m(1 + k(desvioPadrao/R - 1))$;
    se *intensidade do pixel* $< T$ **então**
      $Im_binarizada(i, j)$ = O *pixel* é classificado como objeto;
    fim
    else
      $Im_binarizada(i, j)$ = O *pixel* é classificado como fundo;
    end
  fim
fim

2.2.4 Pós-processamento

Muitas vezes o resultado da segmentação não é adequado para que os grupos de *pixels* segmentados sejam representados e descritos em termo de suas características nas etapas subsequentes. Sendo assim, para a correção de defeitos residuais nas imagens, torna-se necessária a etapa de pós-processamento.

A separação de objetos que se tocam, a eliminação de objetos que não se deseja extrair nenhuma informação e o agrupamento de objetos com o intuito de formar outros mais complexos, são exemplos de procedimentos realizados na etapa de pós-processamento. Estes procedimentos são realizados através de operações lógicas e morfológicas.

As operações lógicas são operações pontuais, isto é, são executadas *pixel* a *pixel* ou orientadas a vizinhança, gerando uma imagem de saída contendo *pixels* preservados ou invertidos em relação aos da imagem de entrada. As principais operações lógicas em processamento de imagens são a interseção (AND), o complemento (NOT) e a união (OR), a partir dos quais combinações podem ser feitas para formar

qualquer outra operação lógica (GONZALEZ; WOODS, 2000).

As operações morfológicas são semelhantes à anterior, nas quais um *pixel* da imagem de saída é função do valor dos *pixels* de uma vizinhança da imagem de entrada. Quando se está trabalhando com imagens binarizadas, a principal aplicação da morfologia é extrair componentes da imagem que sejam úteis na representação e descrição de formatos (FILHO; NETO, 1999). A seguir, serão apresentados dois algoritmos utilizados neste trabalho na etapa de pós-processamento, conhecidos como preenchimento de regiões (*Filling Holes*) e esqueletização (*Skeletonization*).

2.2.4.1 Preenchimento de regiões

Este método é simples e eficiente (SOMASUNDARAM; KALAISELVI, 2010). Inicialmente é calculado a área, em *pixels*, de cada região considerada como fundo. Em seguida, é verificado se está área é menor que um valor limite e caso positivo, estes buracos passam a ser considerados como objeto. O valor limite é um termo definido pelo usuário e seu calculo depende do tipo da aplicação.

Todavia, o objetivo da utilização deste método neste trabalho é a eliminação dos ruídos presentes na borda da imagem após a etapa de segmentação. Sendo assim, foi realizada uma pequena modificação, na qual o algoritmo passa a considerar aquelas regiões determinadas como objetos que tocam nas bordas das imagens como fundo, independente de sua área.

2.2.4.2 Esqueletização

A representação e o entendimento das formas desempenham um papel fundamental nas aplicações de visão computacional atuais (JU; BAKER; CHIU, 2007). Sendo assim, uma importante abordagem para a representação estrutural da forma de uma região plana consiste em reduzi-la a um grafo. Essa redução pode ser realizada obtendo-se o esqueleto da região através de um algoritmo de afinamento (também conhecido como esqueletização) (GONZALEZ; WOODS, 2000).

O esqueleto de uma região pode ser definido usando a Transformação do Eixo Médio (MAT - Medial Axis Transformation), proposta por Blum (1967) e definida como:

"A MAT de uma região R com fronteira B é obtida da seguinte forma: para cada ponto p em R, encontra-se seu vizinho mais próximo em B. Se p tem mais de um vizinho à mesma distância mínima, diz-se que p pertence ao eixo médio (ou esqueleto) de R".

Embora a MAT resulte em um esqueleto aceitável intuitivamente, na prática a sua implementação é impossível quanto a aspectos computacionais, pois envolve o cálculo da distância de cada ponto interior a cada ponto da fronteira de uma região.

Dessa forma, neste trabalho foi explorado um interessante algoritmo para o afinamento de regiões binárias, que consiste de dois passos aos pontos do contorno da região, em que um ponto de contorno é definido como um *pixel* de valor 1 que possua pelo menos um vizinho de oito com o valor 0. A definição de vizinhança de 8 é exibida na Figura 2.3.

p9	p2	p3
p8	p1	p4
p7	p6	p5

Figura 2.3: Estrutura da vizinhança de 8, usada pelo algoritmo de esqueletização

O primeiro passo determina que um ponto p deve ser eliminado se as seguintes condições forem satisfeitas:

(i) $2 \leq N(p_1) \leq 6$;

(ii) $S(p_1) = 1$;

(iii) $p_2 \cdot p_4 \cdot p_6 = 0$;

(iv) $p_4 \cdot p_6 \cdot p_8 = 0$;

Em que $N(p_1)$ é o número de vizinhos não-nulos de p_1 e $S(p_1)$ é o número de transições $0 - 1$. No exemplo da Figura 2.4, $N(p_1) = 4$ e $S(p_1) = 3$.

O passo 1 deve ser aplicado a cada ponto de contorno da região binária da imagem, onde se alguma das condições não for satisfeita o valor do *pixel* não será alterado. No entanto, se todas as condições forem satisfeitas, o ponto será marcado para ser apagado somente após todos os pontos do contorno forem processados.

O passo 2 deve ser aplicado aos dados resultantes exatamente da mesma maneira que o primeiro passo, com as condições (i) e (ii) mantidas, porém as condições (iii) e (iv) passam a ser:

(iii) $p_2 \cdot p_4 \cdot p_8 = 0$;

0	0	1
1	p1	0
1	0	1

Figura 2.4: Exemplo das condições (i) e (ii)

(iv) $p_2 \cdot p_6 \cdot p_8 = 0$;

Com isso, esse procedimento básico deve ser aplicado iterativamente até que não hajam mais pontos a serem apagados, concluindo o algoritmo calculando o esqueleto da imagem.

2.2.5 Extração de atributos

Esta é a etapa em que se inicia a análise da imagem, ou seja, se extrai as informações úteis da imagem processada.

Em geral, a Extração de Atributos pode ser dividida em dois tipos de medida: medidas da imagem como um todo e medidas de região. A primeira se refere ao campo como um todo, como por exemplo, a medição de número de objetos, área total dos objetos, fração de área e etc. Gerando como resultado apenas um valor por medida.

As medidas de região se referem aos objetos individualmente, ou seja, é extraído um parâmetro de cada objeto na imagem, como por exemplo, tamanho, forma e posição das partículas (ESQUEF, 2002).

2.2.6 Classificação e Reconhecimento

Um dos principais objetivos da análise de imagens por computador é dotar uma máquina com capacidade similar à dos seres humanos na realização de tarefas (GONZALEZ; WOODS, 2000). Sendo assim, o objetivo do reconhecimento é realizar de forma automática, a "identificação"dos objetos segmentados da imagem.

Existem duas etapas no processo de classificação de formas: o aprendizado e o reconhecimento propriamente dito. Na maior parte dos sistemas de reconhecimento de formas, os parâmetros provenientes da etapa de extração de atributos são utilizados

para construir um espaço de medida a Z dimensões. Os sistemas de aprendizado irão definir uma função discriminante que separe eficientemente todas as formas representadas neste espaço de medida.

O espaço de medidas é um espaço de dimensão Z onde cada uma corresponde a um parâmetro (atributo). A escolha e a qualidade dos parâmetros para construir este espaço é essencial para a realização de um bom processo de reconhecimento de forma.

Capítulo 3

Abordagem Proposta

Nesse capítulo será apresentado a metodologia utilizada para o desenvolvimento da proposta sugerida neste trabalho. O sistema proposto realiza o processamento de um diagrama de fluxo de sinal desenhado a mão e encontra a sua função de transferência equivalente. Para isso são utilizados os métodos apresentados no Capítulo 2.

Para se chegar a este método que está sendo proposto, foram desenvolvidas e estudadas outras técnicas, que foram sendo aperfeiçoadas. Estes métodos apresentaram vantagens e desvantagens que foram sendo analisadas e melhoradas até se chegar ao método final. Nas seções seguintes serão apresentados os desenvolvimentos de cada item que compõe o sistema como um todo. Inicialmente é descrito o algoritmo utilizado e os passos necessários a partir da aquisição da imagem até encontrar a função de transferência do diagrama de fluxo de sinal. Em seguida descreve-se as restrições impostas pelo software, que solucionam as dificuldades encontradas na hora da implementação e aumentam a probabilidade de obter resultados satisfatórios.

3.1 Algoritmo Geral

O propósito deste trabalho é, a partir da imagem de um diagrama de fluxo de sinal desenhado a mão, encontrar todos os caminhos contidos nele para calcular automaticamente a sua função de transferência equivalente. A metodologia empregada por trás do algoritmo geral envolve muitas etapas. Sendo assim, elas são exibidas no fluxograma da Figura 3.1 e em seguida, descritas nas próximas seções.

3.1.1 Segmentação da Imagem

Como foi apresentado no Capítulo 2.2.3, o objetivo desta etapa é binarizar a imagem de tal forma que só mantenha um único objeto descrevendo o diagrama

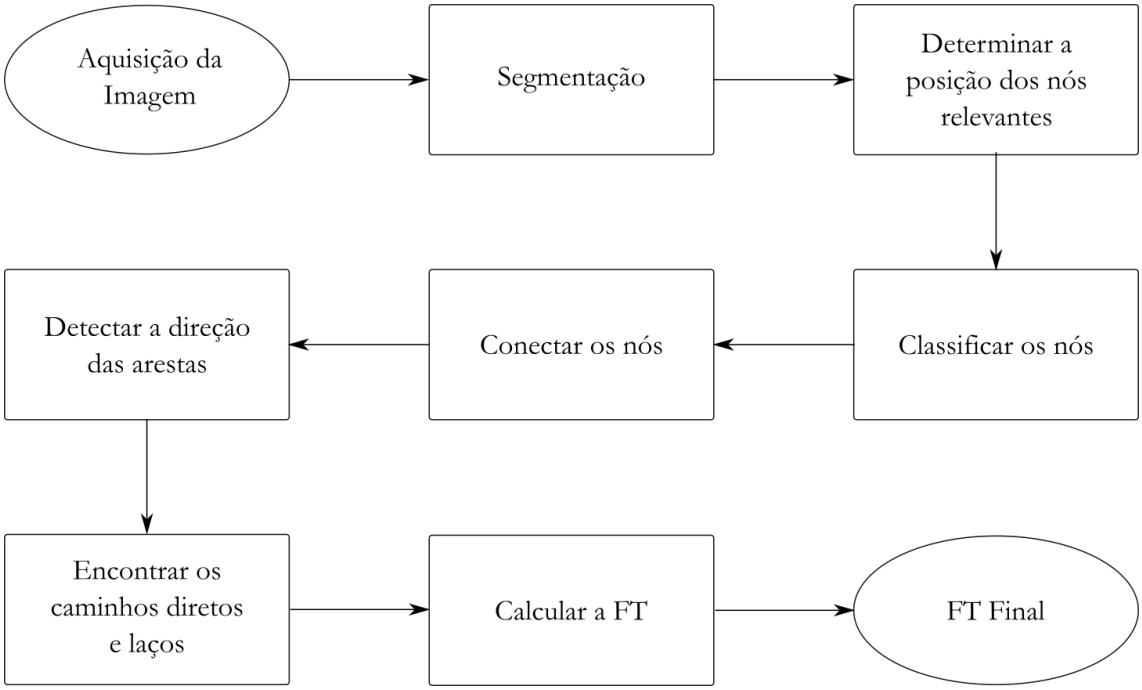


Figura 3.1: Diagrama das etapas do algoritmo geral proposto

dado. Sendo assim, nesta etapa a imagem foi submetida ao processo exibido no fluxograma da Figura 3.2.

Ao analisá-lo percebe-se que inicialmente a imagem é convertida em escala de cinza, para então ser submetida a uma etapa de pré-processamento, onde foi aplicado um filtro morfológico da mediana, utilizado para suavização e eliminação de ruído. No sistema apresentado, o filtro da mediana aplicado possui uma máscara com o tamanho 20×20.

Após a etapa de pré-processamento, o sistema proposto utilizou o método de binarização proposta por Sauvola (2000), onde a janela retangular foi definida com o tamanho 300×300.

Para a etapa que sucede a segmentação, a imagem foi submetida ao procedimento de preenchimento de regiões, como explicado no Capitulo 2.2.4.1, na qual o sistema realiza uma busca em torno da borda da imagem, procurando por uma região considerada como objeto. Uma vez encontrada, o algoritmo realiza um preenchimento nesta região, passando assim a classificá-la como fundo.

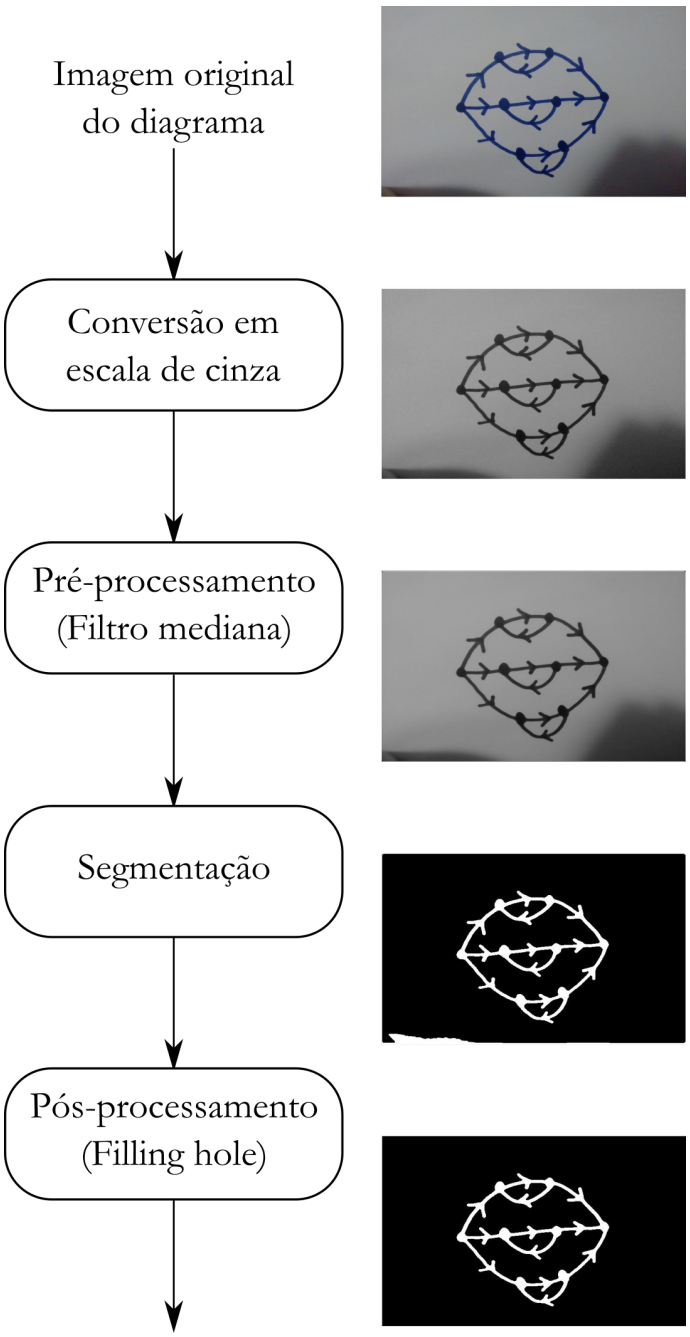


Figura 3.2: Fluxograma das etapas de segmentação da imagem

3.1.2 Determinar a posição espacial dos nós

Para extrair as informações dos nós do diagrama, foi necessário inicialmente localizar os *nós relevantes*. Esses são pontos na imagem que podem ser classificados como um nó do grafo (bifurcação, nó inicial ou final) ou uma aresta. É importante reforçar que, neste ponto do processo, o sistema tem somente um objeto que

representa a imagem binarizada do diagrama, encontrada na etapa anterior. Este processo obedece as seguintes etapas:

1. *Calcular o esqueleto binário do objeto*: Tem como objetivo encontrar um objeto que possui a mesma estrutura de conexão do diagrama original, porém com um *pixel* de largura. Para isso, o sistema executa o algoritmo exposto no Capítulo 2.2.4.2. Um exemplo para o resultado desta etapa pode ser visualizado na Figura 3.3, na qual o esqueleto do diagrama está representado pelos *pixels* na cor vermelha e a área verde representa o objeto original.

2. *Encontrar pixels candidatos*: A técnica proposta nesta etapa consiste em encontrar pontos na imagem que tem mais de dois vizinhos, pois pela característica do esqueleto os que tem três ou mais vizinhos, são aqueles que possuem três ou mais caminhos para se conectarem com outros nós no grafo. A Figura 3.3 ilustra este conceito, onde as setas brancas indicam os *pixels* com mais de dois vizinhos.

3. *Quantizar os pixels em nós relevantes*: Ainda na Figura 3.3, percebe-se que existem nós que podem ter dezenas de *pixels* candidatos a relevantes (e.g. os dois pontos localizados na região superior da imagem). Desta forma, é necessário quantizá-los como nós únicos. Para isso, usa-se um *threshold* e considera-se os pontos que estão dentro desse limite como um único nó.

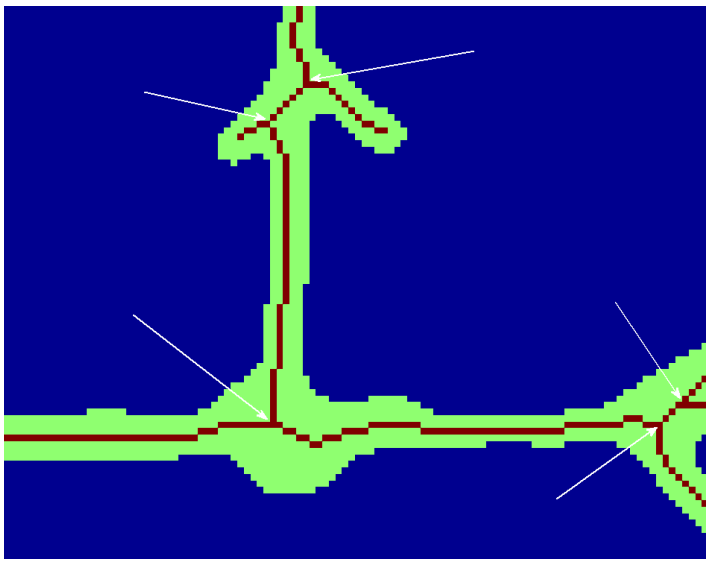


Figura 3.3: Exemplo do esqueleto da imagem e dos *pixels* candidatos a nós

Selecionar um *threshold* incorreto, ocasiona erros no processo de quantização, que é um dos principais problemas nesta abordagem. Para evitar isto, é proposto o Algoritmo 2 para o cálculo do *threshold*, na qual em sua entrada o usuário fornece as posições dos nós candidatos da imagem e assim o algoritmo retorna a distância mínima à ser utilizada. Este algoritmo consiste basicamente nos seguintes passos:

1. Medir e armazenar em um vetor as distâncias entre todos os pares de *pixels* candidatos. Esse vetor será utilizado para o sistema analisar o conjunto de todas as distâncias entre os pares de nós candidatos.

2. Ordenar o vetor, resultando em um vetor crescente, que conterá basicamente dois grupos de distâncias.

    - O primeiro grupo corresponde a pequenas distâncias devido aos *pixels* que pertencem ao mesmo nó. O segundo grupo irá corresponder aos *pixels* que estão distantes.

3. Obter um vetor com o log das distâncias. Este procedimento é realizado para homogeneizar o vetor. Isso significa que grandes distâncias estão mais ou menos na mesma escala que as pequenas, ou seja, se uma grande distância dobrar e uma pequena também dobrar, a diferença entre as duas será a mesma.

4. Encontrar o índice correspondente ao maior aumento no log. Este será o índice da distância que separa os dois grupos (os pequenos dos grandes).

A Figura 3.4 mostra os plots das distâncias ordenadas (gráfico superior) e as derivadas numéricas das distâncias ordenadas (gráfico inferior). Como podemos ver no gráfico de diferenças há um pico no índice 312. Tomando a média da distância nos índices 311 e 313, dará o *threshold* limiar correto que irá quantizar os nós corretamente.

3.1.3 Classificação dos nós relevantes

O objetivo desta etapa é classificar os nós como: Nó inicial, nó final, ligação (corresponde as setas das arestas que indicam a direção do fluxo do sinal) ou bifurcação (pontos em que os subsistemas se conectam entre si).

Uma das dificuldades nesse tipo de aplicação é a heterogeneidade das imagens desenhadas em papel. Buscando uma normatização e padronização mínima dos

Algoritmo 2 Algoritmo para calcular a distância mínima utilizada na quantização

Entrada: *pixelsCandidatos*: Posição espacial dos *pixels* candidatos
Saída: t: Distância mínima
início
  para $i = 1$ ***até*** *quantidade(pixelsCandidatos)* **faça**
    para $j = 1$ ***até*** *quantidade(pixelsCandidatos)* **faça**
      se $i \neq j$ **então**
        $dist$ = distância entre $pixelsCandidatos(i)$ e $pixelsCandidatos(j)$;
      fim
    fim
  fim
  $dist$ = Ordenar o vetor $dist$;
  $distanciasLog$ = Calcular o vetor com o log de $dist$;
  $dDistanciasLog$ = Calcular a diferença do vetor $distanciasLog$;
  i = Índice correspondente ao maior valor do vetor $dDistanciasLog$;
  t = A média entre os valores anterior e posterior ao índice i;
fim

desenhos, o usuário deve seguir as diretrizes indicadas no Capítulo 3.2 ao desenhar o diagrama.

Assim, para o sistema classificar os nós, executa os passos proposto no Algoritmo 3, detalhados a seguir:

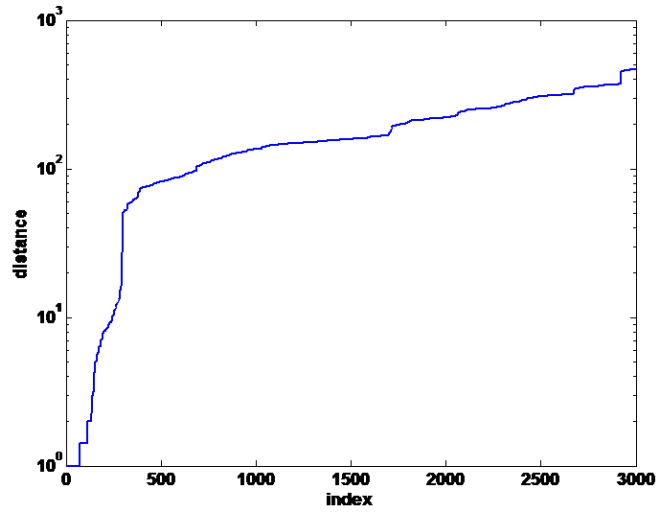
1. Para cada nó, é realizado um preenchimento no esqueleto do grafo a partir do *pixel* central.

2. Esse preenchimento crescerá como uma árvore, do centro do nó até o ponto em que a distância entre os dois *pixels* seja igual a distância mínima calculada na etapa anterior. Esse ponto foi denominado como ponto de conexão;

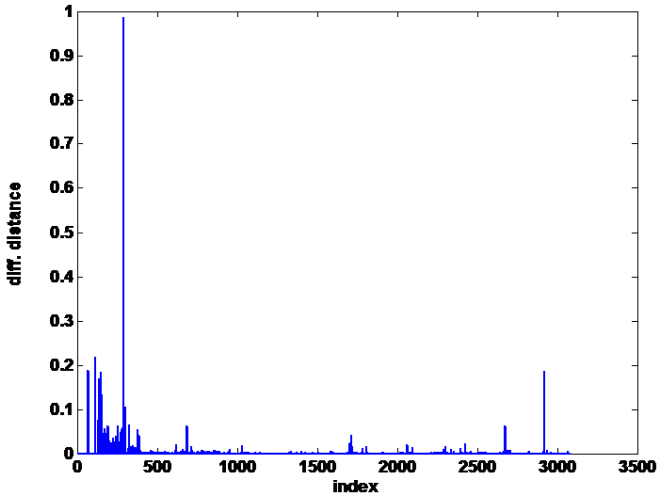
3. Após realizar o preenchimento para todos os caminhos possíveis, calcula-se quantos pontos de conexão o nó possui.

   - Se o nó possuir dois pontos de conexão, ele é classificado como um nó de ligação;
   - Se for três pontos de conexão, ele é classificado como uma bifurcação;

4. Verifica se o nó é o mais a esquerda, caso positivo ele passa a ser classificado como nó inicial;



(a)



(b)

Figura 3.4: Plot das distâncias e suas diferenças

5. Verifica se ele é o mais a direita, caso positivo, ele passa a ser classificado como nó final;

Essas definições podem ser visualizadas na Figura 3.5, onde os círculos azuis indicam os pontos de conexão no esqueleto da imagem que cruzam a distância limite em uma aresta, já os círculos verdes indicam o cruzamento em uma bifurcação.

3.1.4 Conexão dos nós

Feita a classificação dos nós é necessário encontrar os seus vizinhos. Dessa forma, foi selecionado cada ponto de conexão e encontrado quais nós estão conectados a

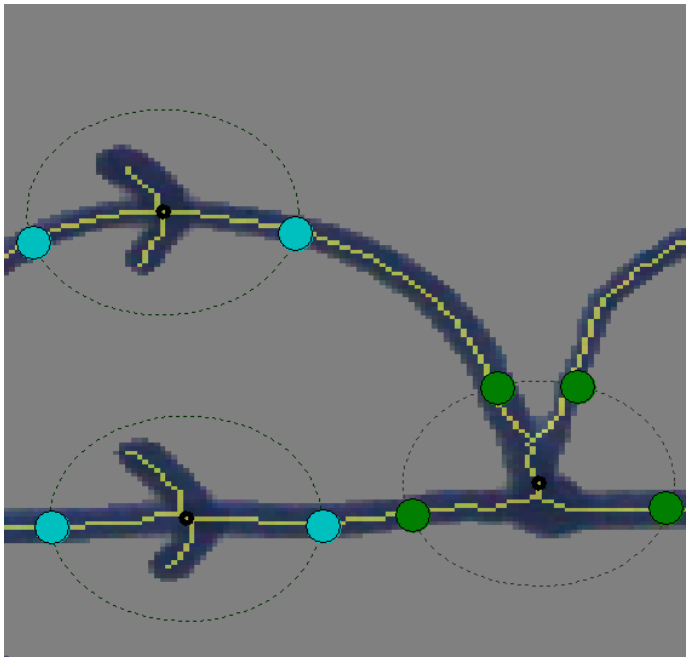


Figura 3.5: Ilustração da classificação dos nós como arestas ou bifurcações

Algoritmo 3 Algoritmo para classificação dos nós

Entrada: n: Posição espacial dos centros dos nós;
t: Distância mínima;
Saída: $ClassificacaoNos$: Vetor que armazena a classificação dos nós
início
  para i = *1* ***até*** *quantidade(n)* **faça**
    $[pontosConexao]$ = Realiza-se um preenchimento no esqueleto a partir do ponto central de $n(i)$;
    se $quantidade(pontosConexao)$ = *2* **então**
      $ClassificacaoNos(i)$ = aresta;
    senão
      $ClassificacaoNos(i)$ = bifurcação;
    fim
    se *(n(i) estiver mais a esquerda)* **então**
      $ClassificacaoNos(i)$ = nó inicial;
    senão
      se *(n(i) estiver mais a direita)* **então**
        $ClassificacaoNos(i)$ = nó final;
      fim
    fim
  fim
fim

ele. Para realizar esta tarefa são executados os passos apresentados no Algoritmo 4, detalhados a seguir:

1. Para cada nó, é realizado um preenchimento no esqueleto a partir de cada ponto de conexão pertencente a ele.

2. Este preenchimento ocorrerá "*pixel* a *pixel*"e crescerá na direção em que a distância para o ponto central do nó seja maior que o *threshold* mínimo, até atingir outro ponto de conexão;

3. O nó pertencente ao ponto de conexão encontrado é conectado ao que iniciou a busca.

Algoritmo 4 ALGORITMO PARA ENCONTRAR A CONEXÃO DOS NÓS

Entrada: n: Posição espacial dos centros dos nós;
$pontoConexao$: Posição espacial dos pontos de conexão;
t: Distância mínima;

Saída: $conexoes$: Índice dos nós vizinhos de cada nó

```
início
    para i = 1 até quantidade(n) faça
        para j = 1 até quantidade(pontoConexao(i)) faça
            noVizinho = Realiza-se um preenchimento no esqueleto a partir
            de pontoConexao(i) até encontrar uma outra conexão
            pertencente a outro nó;
            conexoes(i)(j) = [n(i) noVizinho];
        fim
    fim
fim
```

3.1.5 Detecção das direções das arestas

Após montar o grafo que representa o diagrama, é necessário detectar a direção de suas arestas. Para realização desta tarefa são executados os passos apresentados no Algoritmo 5, detalhados a seguir:

1. Encontrar para cada nó de ligação ambos os *pixels* finais da seta. Ou seja, é iniciado um preenchimento no esqueleto da imagem, a partir de cada nó de ligação, até encontrar dois *pixels* com somente um vizinho;

2. Calcular a posição do ponto médio entre os *pixels* finais;

3. Calcular as distâncias entre o ponto médio e os dois pontos de conexão do respectivo nó de ligação;

4. Determinar a menor distância, pois a direção é definida do ponto de conexão com a menor distância para o com a maior. Por exemplo, para uma aresta horizontal com a seta para a direita, o ponto de conexão mais a esquerda é mais próximo do ponto médio, de modo que a direção definida será da esquerda para a direita.

Algoritmo 5 ALGORITMO PARA DETECTAR AS DIREÇÕES DAS ARESTAS

Entrada: n: Vetor que armazena a posição espacial dos nós;
$pontoConexao$: Vetor que armazena a posição dos pontos de conexão;
$ClassificacaoNos$: Vetor que armazena a classificação dos nós;
Saída: *direcoes*: Vetor que armazena a direção de cada aresta
início
  para i = *1* ***até*** *quantidade(n)* **faça**
    se $ClassificacaoNos(i)$ = *aresta* **então**
      $pxFinais$ = Realiza um preenchimento no esqueleto para encontrar ambos *pixels* finais;
      $pontoMedio$ = Calcula o ponto médio entre $pxFinais$;
      $d1$= Distância entre $pontoMedio$ e $pontoConexao(i)(1)$;
      $d2$= Distância entre $pontoMedio$ e $pontoConexao(i)(2)$;
      se $d1 < d2$ **então**
        $direcoes(i)$ = A direção é definida do $pontoConexao(i)(1)$ para $pontoConexao(i)(2)$;
      senão
        $direcoes(i)$ = A direção é definida do $pontoConexao(i)(2)$ para $pontoConexao(i)(1)$;
      fim
    fim
  fim
fim

3.1.6 Encontrando os caminhos diretos

Para encontrar os caminhos diretos nesta aplicação, foi desenvolvido o Algoritmo 6, detalhado a seguir:

1. Iniciar uma busca a partir do nó inicial;

2. Manter um laço realizando a busca até que todos os caminhos terminem com o nó final;

3. Para cada caminho encontrado, é iniciado um novo laço, em que se obtém os x nós vizinhos do último nó do caminho;

4. São criados x novos caminhos, adicionando cada vizinho ao caminho atual (aquele que gerou os vizinhos);

5. O caminho atual é marcado;

6. Apagar todos os caminhos marcados e os que possuem algum laço;

    - Testar se no conjunto dos caminhos algum nó aparece mais de uma vez;

Algoritmo 6 Algoritmo para encontrar os caminhos diretos

Entrada: n: Posição espacial dos centros dos nós;
$ClassificacaoNos$: Vetor que armazena a classificação dos nós;
conexoes: Índice dos nós vizinhos de cada nó;
direcoes: Vetor que armazena a direção de cada aresta;
Saída: *caminhos*: Vetor que armazena os caminhos diretos;
início
  $idInicial$ = Índice do vetor $ClassificacaoNos$ que indica o nó inicial;
  $idFinal$ = Índice do vetor $ClassificacaoNos$ que indica o nó final;
  $caminhos = [n(idInicial)]$;
  enquanto *todos os caminhos não alcançarem* $n(idFinal)$ **faça**
    id= [];
    para $i = 1$ ***até*** *quantidade(caminhos)* **faça**
      se *o nó final de caminhos*$\{i\}(end) \neq n(idFinal)$ **então**
        $[nos, arestas]$ = Obter o nó vizinho de $caminhos\{i\}(end)$;
      fim
      para $j = 1$ ***até*** *quantidade(nos)* **faça**
        $caminhos\{end + 1\}$ = Adicionar em *caminhos* a $arestas(j)$ e $nos(j)$;
        $id = [id\ i]$;
      fim
    fim
    Remover de *caminhos* aqueles que tem o índice id;
    Remover de *caminhos* os *loops*;
  fim
fim

3.1.7 Encontrando os loops

Para encontrar os laços presentes no grafo, foi desenvolvido o Algoritmo 7 que realiza buscas no esqueleto da imagem até encontrá-los, detalhados a seguir:

1. É iniciada uma busca no grafo a partir de cada nó não classificado como uma aresta;

2. Este procedimento ocorre até atingir o ponto que iniciou a busca, ou então o nó final do laço não ter vizinhos;

3. Durante este processo, são eliminados os caminhos sem *loops* e aqueles que têm um laço, porém o nó final da busca não é o que iniciou.

Algoritmo 7 ALGORITMO PARA ENCONTRAR OS LAÇOS

Entrada: n: Posição espacial dos centros dos nós;
$classificacaoNos$: Vetor que armazena a classificação dos nós;
Saída: *loops*: Vetor que armazena os laços;
início
  para $i = 1$ ***até*** *quantidade(n)* **faça**
    se $classificacaoNos(i) \neq aresta$ **então**
      $loops = [n(i)]$;
      enquanto *(loops não alcançar o seu nó de partida*
      $\|$ $loops\{end\}$ *não ter vizinhos)* **faça**
        id=[];
        para *i=1* ***até*** *quantidade(loops)* **faça**
          se *(nó final de* $loops\{i\} \neq n(i)$*)* **então**
            $[nos, arestas]$ = Obter o nó vizinho de $loops\{i\}(end)$);
          fim
          para $j = 1$ ***até*** *quantidade(nos)* **faça**
            $loops\{end + 1\}$ = Adicionar em *loops* a $arestas(j)$ e
            $nos(j)$];
            $id = [id\ i]$;
          fim
        fim
        Remover de *loops* aqueles com o índice *id*;
        Remover de *loops* os que não possuem laços;
      fim
    fim
  fim
fim

3.1.8 Calculo da Função de Transferência

O cálculo da Fórmula de Mason é o último passo a ser executado. Neste o sistema seleciona todos os nós de ligação encontrados no grafo e solicita as FT's individuais. Ao obter essas informações, é calculada a FT final usando a expressão apresentada na Equação 2.1.

3.2 Exigências do Software

No desenvolvimento do software proposto neste trabalho, uma das dificuldades encontradas foi a heterogeneidade das imagens desenhadas em papel, pois cada pessoa possui uma habilidade de desenhar diferente. Dessa forma, buscando uma normatização e padronização mínima dos desenhos para facilitar a capacidade do sistema distinguir os diferentes padrões, foram impostas algumas diretrizes em que é fundamental serem seguidas para um resultado satisfatório. Essas regras são descritas a seguir:

1. O nó inicial deve ser o mais à esquerda;
2. O nó final deve ser o mais à direita;
3. O nó inicial e final devem conter no mínimo três caminhos partindo deles, pois de acordo com a característica do esqueleto da imagem, só é classificado como um nó aqueles *pixels* que possui três ou mais vizinhos. A Figura 3.6 mostra três exemplos do resultado da etapa de classificação em que ocorreu o caso apresentado, na qual o círculo amarelo representa o nó que foi classificado como inicial, o branco uma bifurcação e o verde uma aresta. A linha fina amarela retrata o esqueleto da imagem e os pequenos quadrados enumerados, a identificação do nó. Analisando as imagens, percebe-se que na primeira e segunda situação, o nó inicial foi identificado por um círculo e no segundo caso foi adicionado um ramo pequeno. Dessa forma, o esqueleto da imagem nesse segmento não teve nenhum *pixel* candidato, então o nó de ligação foi classificado como nó inicial. Enquanto isso, na terceira imagem pelo fato do ponto conter três vizinhos, o nó inicial foi classificado de forma correta;
4. As arestas devem ser identificadas por uma seta, que devem ser abertas em vez das cheias. A seta indica sua direção e garante que a aresta terá uma bifurcação no esqueleto da imagem;

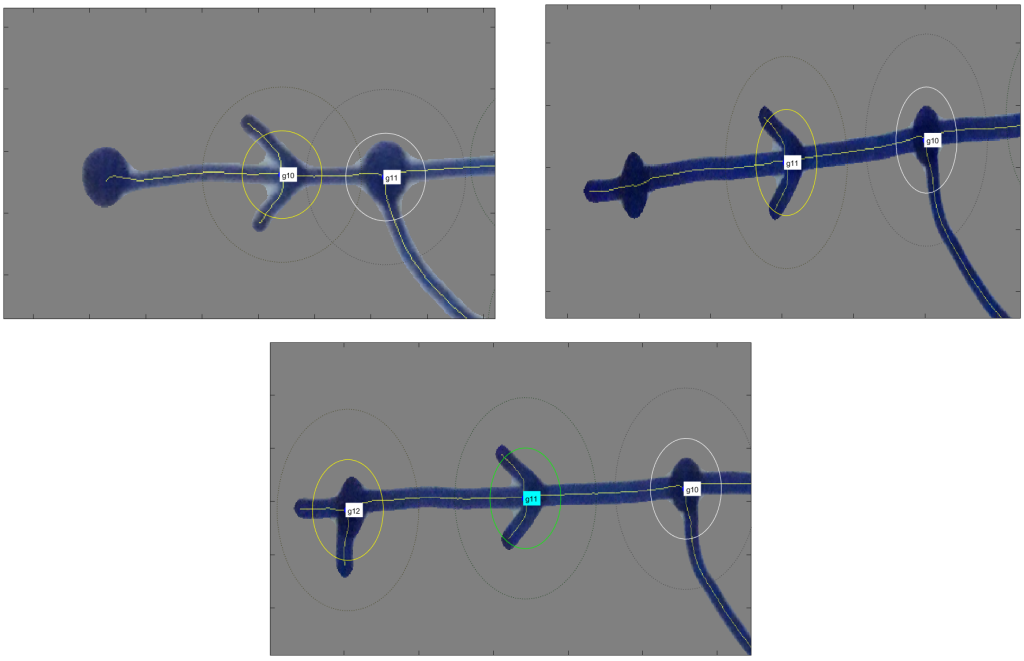


Figura 3.6: Resultado da etapa de classificação dos nós, em que o nó inicial contém um, dois e três ramos partindo dele

5. Os componentes do grafo, ou seja, os nós e as setas que representam as arestas, não devem se sobrepor;

Capítulo 4

Testes e Resultados

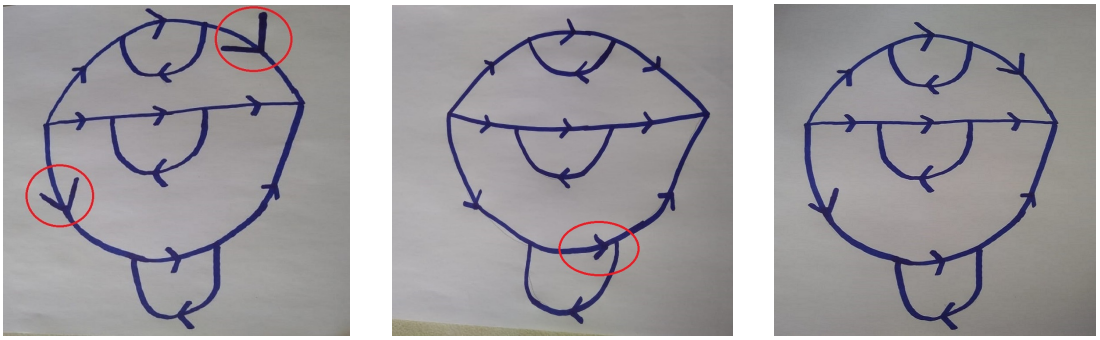
A abordagem apresentada neste trabalho foi implementada na plataforma MATLAB (versão R2016a), e nesta seção serão apresentados testes e resultados obtidos com esse método desenvolvido. Os testes realizados destinam-se a ilustrar o comportamento das etapas mais críticas do software apresentado no Capítulo 3. Ao final, serão apresentados alguns resultados juntamente com uma análise qualitativa sobre o comportamento do sistema em cada situação apresentada.

4.1 Testes

Para ilustrar o funcionamento do método proposto, foram executados testes na etapa em que apresentou limitações no software e a seguir eles serão apresentados. Os testes objetivaram estudar as etapas que determinam a posição espacial e classificam os nós. Mais especificamente, o algoritmo que calcula a distância mínima e como esse parâmetro afeta as etapas subsequentes do sistema.

4.1.1 Distância Mínima

O cálculo do parâmetro da distância mínima é a principal etapa do sistema e caso o resultado encontrado for incorreto, influenciará nos resultados das etapas posteriores. Para realizar este teste foram geradas três situações. A primeira consiste em um grafo na qual algumas setas estão desproporcionais em relação ao resto do desenho. Já a segunda aborda o cenário em que a desproporcionalidade está na distância entre algumas arestas e os nós adjacentes. Finalmente, no último caso todas as distâncias e tamanhos das arestas estão proporcionais. A Figura 4.1 apresenta as situações expostas, na qual o primeiro e segundo cenário está envolvido por um círculo vermelho. É importante ressaltar que todas as imagens possuem a resolução de $725x747$ *pixels*.



(1) (2) (3)

Figura 4.1: Conjunto de dados utilizados para a realização de testes no cálculo da distância mínima

No primeiro caso as setas assinaladas possuem o tamanho maior que as demais, de tal forma que seu tamanho é equivalente a distância entre algumas setas e suas respectivas bifurcações vizinhas. Sendo assim, ao calcular a distância mínima, esta não irá atingir todo o contorno da aresta. Isso fica mais claro nas imagens da Figura 4.2. A primeira exibe o zoom em uma das arestas, em que os nós candidatos são representados pelos pequenos círculos verdes, já os pretos correspondem ao resultado da quantização. A linha fina branca representa o esqueleto da imagem e a área vermelha o diagrama segmentado. Já a segunda imagem ilustra os resultados obtidos na etapa de classificação, onde os círculos amarelo e rosa indicam o nó inicial e final, respectivamente. Os círculos verdes representam os nós de ligação, os brancos as bifurcações, os pretos pontilhados indicam a distância mínima calculada e os pequenos quadrados enumerados simbolizam a identificação do nó. Este resultado foi desenhado na imagem original com o fundo removido para uma melhor visualização.

A distância mínima calculada para esta situação foi $55,6638$ *pixels*. Como pode-se observar, nas duas arestas indicadas, o círculo que aponta a área que inclui a distância mínima finaliza antes do término de suas abas. Por esta razão elas foram consideradas como bifurcações.

Para a segunda situação, o nó de ligação indicado está próximo da bifurcação vizinha, ao ponto de que ao realizar a quantização, eles passarão a ser considerados como um único nó. Para uma simples comparação visual, é possível notar que a distância entre os componentes é menor que o tamanho de algumas arestas. A Figura 4.3 exibe o resultado deste cenário, em que a distância mínima calculada foi $40,1203$ *pixels*.

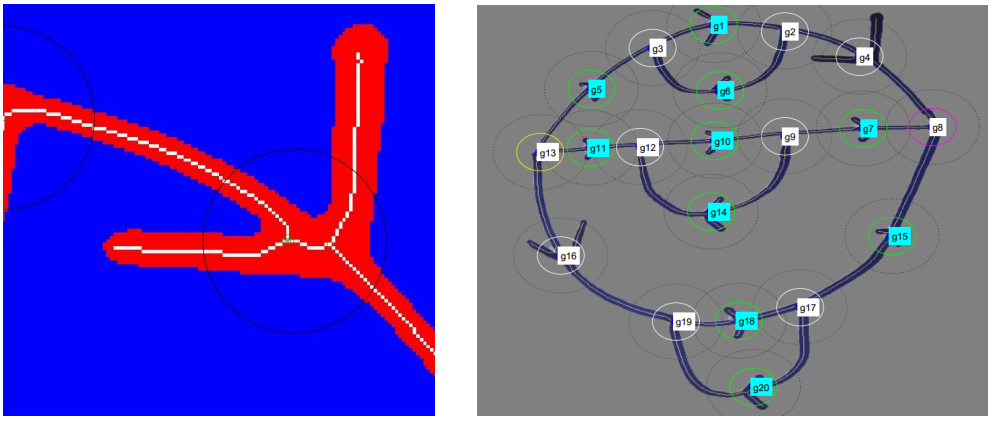


Figura 4.2: Resultados gerados na etapa de determinação da posição e classificação dos nós utilizando a imagem (1)

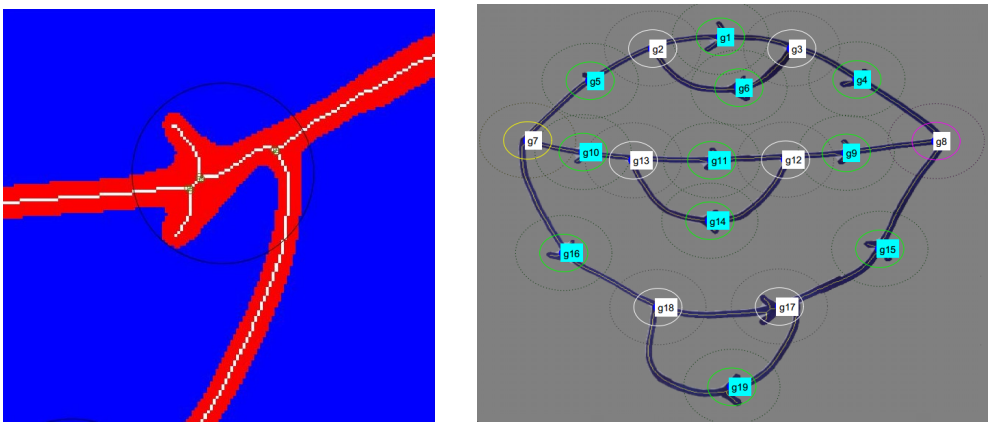


Figura 4.3: Resultados gerados na etapa de determinação da posição e classificação dos nós utilizando a imagem (2)

Para o terceiro caso a distância mínima calculada foi $51,8835$ *pixels* e como pode-se observar na Figura 4.4, todos os nós foram classificados corretamente. Para efeito de comparação, as distâncias calculadas para os três casos são aproximadas, sendo que no primeiro e terceiro caso, elas diferiram em aproximadamente 4 *pixels*. Isso ocorreu devido a diferença na distância da câmera para a imagem ao tirar a foto. Além disso, como no segundo caso a distância entre a aresta e a bifurcação diminuiu, sua respectiva distância também foi menor.

Um fato interessante sobre o sistema é que em todos os testes realizados, não foi encontrado um caso em que se os nós forem classificados corretamente, os resultados para as etapas posteriores do sistemas também serão satisfatórios, ou seja, a função de transferência equivalente encontrada foi exata.

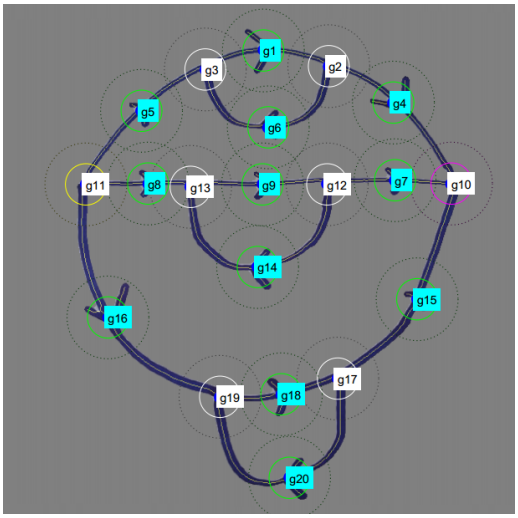


Figura 4.4: Resultados gerados na etapa de classificação dos nós utilizando a imagem (3)

4.2 Resultados

Nesta seção serão apresentados os resultados utilizando o sistema proposto com base no conjunto de dados descrito a seguir. Em cada resultado apresentado, será feito uma análise discutindo os aspectos de dificuldade, qualidade da solução e etc. Serão apresentados e analisados casos particulares em que o usuário não seguiu as diretrizes expostas na Seção 3.2, impossibilitando que o resultado obtido fosse satisfatório. Por fim, serão apresentados casos particulares que representam situações um pouco mais delicadas para o algoritmo proposto.

4.2.1 Dados coletados

Com o objetivo de validar a abordagem proposta neste trabalho e avaliar sua precisão, 50 universitários do curso de Engenharia Elétrica da Universidade Federal do Rio Grande do Norte (UFRN) foram convidados a desenhar um dos oito exemplos de grafos da Figura 4.5, propostos por (NISE, 2007). Estes grafos foram escolhidos porque além de abordar diferentes formas para desenha-los foram consideradas as mais desafiadoras entre os exemplos apresentados por este autor. O conjunto de dados obtidos estão armazenados e disponibilizados na plataforma de armazenamento online Google Drive[1].

Os alunos foram inicialmente instruídos sobre o trabalho e também ficaram res-

[1]<https://drive.google.com/open?id=1zQyRdkD0tTGvJnsu8n3p_8HNg_EflVFr>

ponsáveis por capturar a imagem dos seus produtos, contribuindo para a verificação das diferentes formas possíveis de captura das imagens. Dessa forma, foram obtidas 117 amostras para validação.

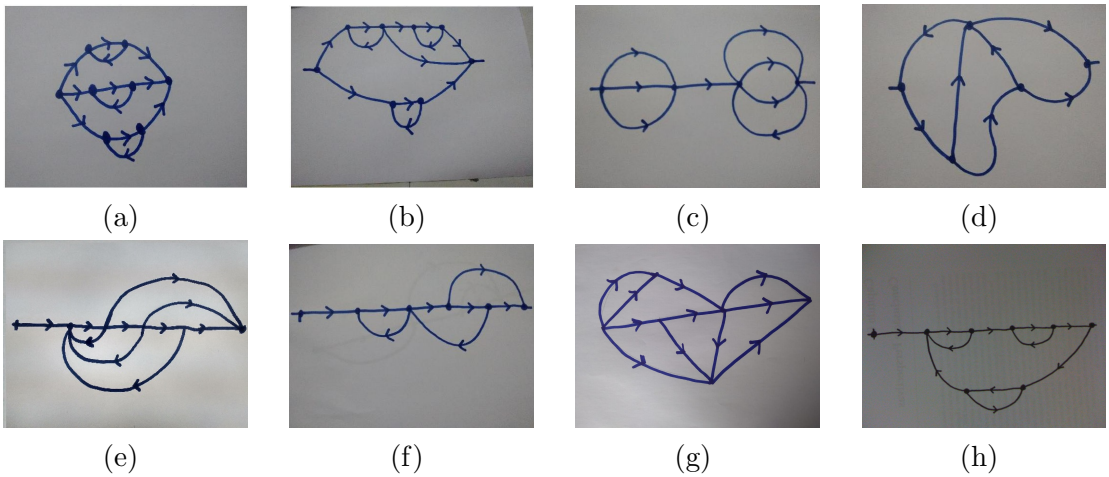


Figura 4.5: Desenhos utilizados para a obtenção dos resultados

No gráfico da Figura 4.6, foi exibido o total de amostras coletadas para cada tipo de diagrama. Como pode-se observar, o conjunto do tipo (c) foi o que se obteve a maior quantidade de amostras $(29,06\%)$, enquanto que o tipo (b) foi a menor $(2,56\%)$, isto ocorreu sem motivos aparentes.

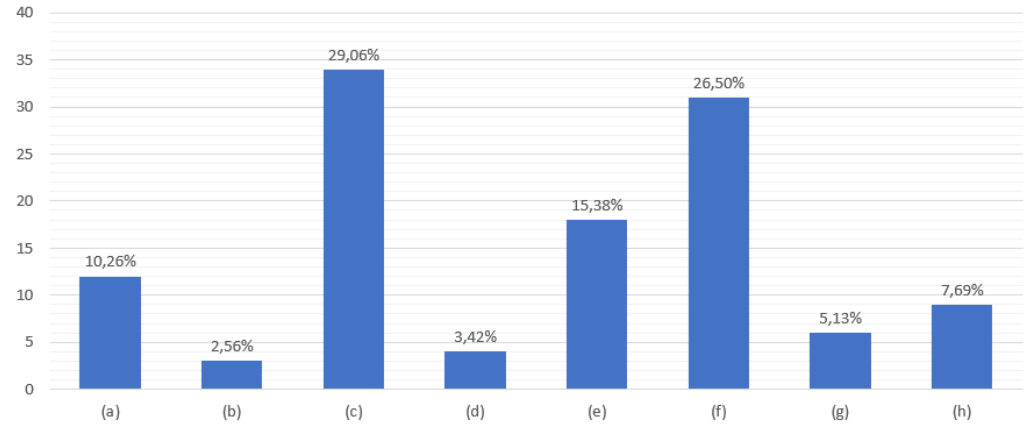


Figura 4.6: Total de amostras coletadas para cada tipo de diagrama

4.2.2 Descrição dos Resultados

Os resultados obtidos utilizando o conjunto de dados descritos anteriormente serão apresentados por etapas. Inicialmente é apresentado um conjunto de imagens

que foram classificadas com erro no desenho. Estas são as que ao serem desenhadas não seguiram as diretrizes apresentadas na Seção 3.2. Em seguida serão expostos os dados quantitativos de cada amostra de dados dos desenhos sem erros.

4.2.2.1 Erros nos Desenhos

Ao analisar as imagens coletadas para a validação desse sistema foi perceptível que alguns desenhos não obedeceram as diretrizes estabelecidas na implementação, apesar de terem sido apresentadas previamente. Consequentemente, estas foram classificadas como imagens com erro no desenho. Isto pode ter acontecido tanto por falta de atenção por parte do usuário como também pela falta de familiaridade com o sistema.

Ao observar estes dados, percebeu-se que os tipos de erros no desenho se repetiram e poderiam ser classificados em cinco. Estas classificações são apresentadaas na Figura 4.7. As imprecisões encontradas estão envolvidas pelo círculo vermelho e são listados a seguir:

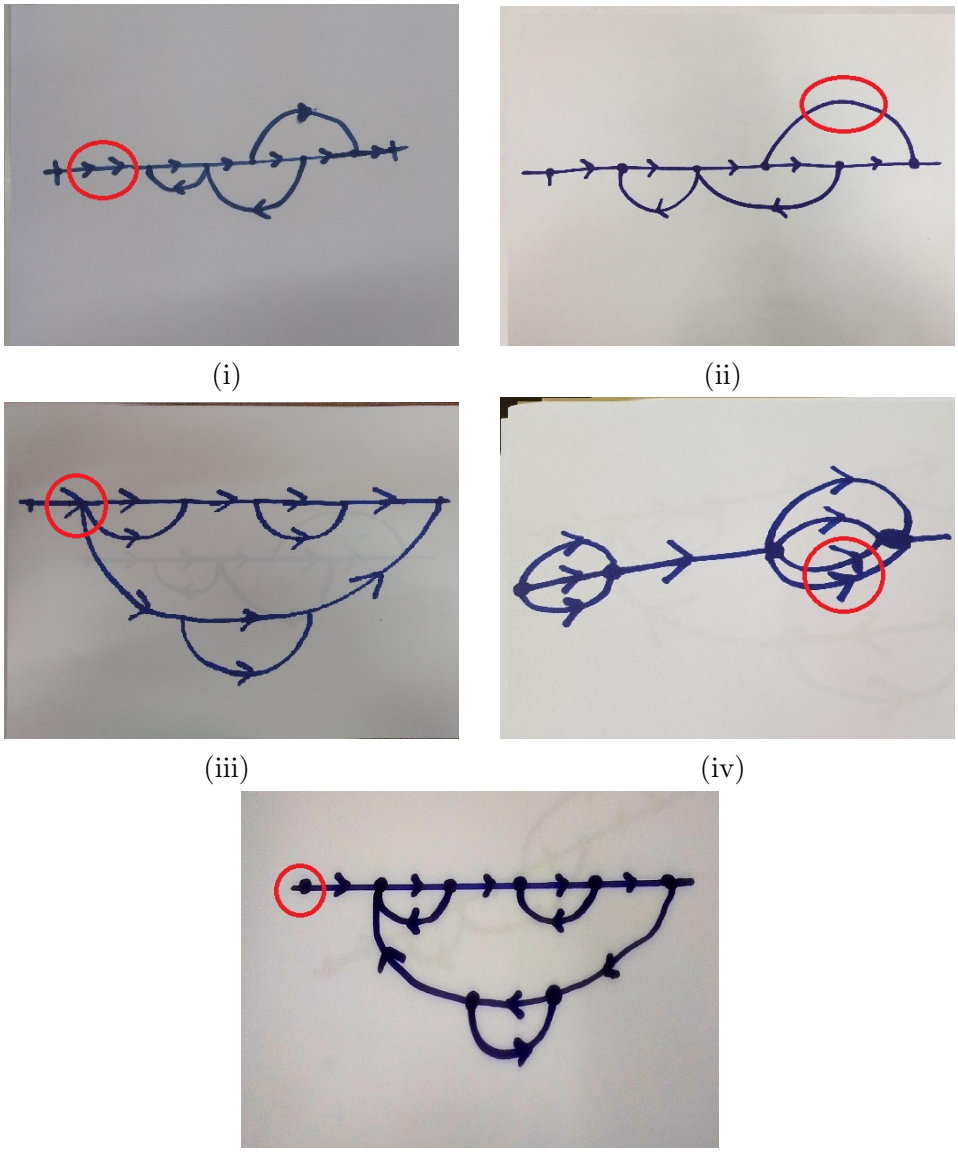
(i) A sintaxe do ramo indicado está errada. Onde deveria ter uma aresta contém duas. Neste caso o sistema encontrou o grafo direcionado e todos os caminhos corretamente, porém, calculou erroneamente o ganho final, pois a estrutura do desenho está diferente da apresentada;

(ii) Não foi desenhada uma aresta no ramo indicado. Isto impossibilita o sistema encontrar a direção do fluxo do sinal neste ramo, pois não existe uma aresta para indicá-la;

(iii) No local apontado há uma seta, que representa o respectivo nó de ligação, posto em cima de uma bifurcação. Dessa forma, ao calcular o esqueleto da imagem, o sistema considerou os dois componentes apenas como um nó;

(iv) Neste caso observa-se que as setas indicadas se tocam, com isso a estrutura do esqueleto do diagrama será calculada erroneamente.

(v) Por fim, como indicado anteriormente, o nó inicial e final devem possuir no mínimo três caminhos partindo dele, caso contrário, o sistema não o identificará como um nó. Dessa forma, pode-se observar que no exemplo o nó inicial não preenche os requisitos estabelecidos, o que causará erros na etapa da classificação dos nós.



(i) (ii)

(iii) (iv)

(v)

Figura 4.7: Amostras consideradas com erros de desenho

A Tabela 4.1 exibe um resumo da quantidade de cada tipo de erro de desenho pela categoria de diagrama. É possível observar que o (ii) possui o maior número de ocorrências. Por outro lado, também pode-se afirmar que o conjunto do tipo (g) apresenta o maior percentual de desenhos com erro, levando em consideração a sua quantidade de amostras. Estes podem ser observados na Figura 4.8, em que os erros nos desenhos estão envolvidos por um círculo vermelho. Analisando-os percebe-se que em três dessas amostras o aluno errou na direção de uma aresta, enquanto que na quarta não foram desenhadas duas arestas. Em geral, pode-se afirmar que esses

erros ocorreram devido a falta de atenção por parte dos usuários.

Tabela 4.1: Resumo da quantidade de cada tipo de erro pela categoria de diagrama

	(i)	(ii)	(iii)	(iv)	(v)
(a)	0	1	0	1	0
(b)	0	0	0	0	0
(c)	0	2	0	1	0
(d)	0	0	0	0	0
(e)	0	1	0	0	0
(f)	1	2	0	0	1
(g)	3	1	0	0	0
(h)	0	0	1	0	2
Total	**4**	**7**	**1**	**2**	**3**

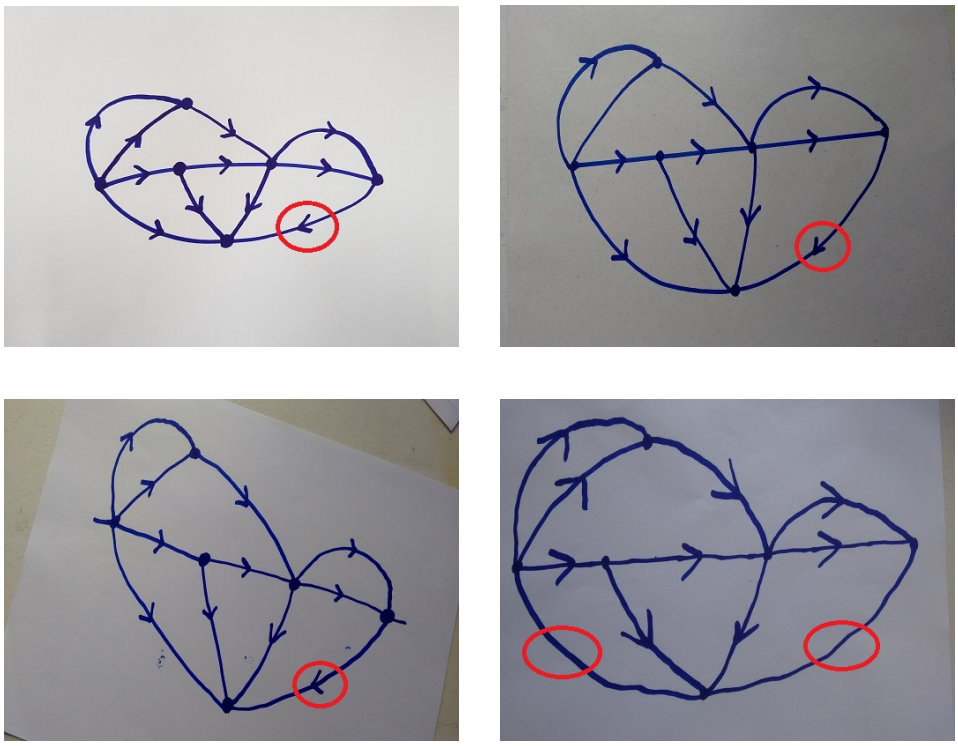


Figura 4.8: Amostras consideradas com erros de desenho

4.2.2.2 Análise dos Resultados

Como apresentado no Capítulo 3, o sistema executa várias etapas até encontrar a função de transferência equivalente ao grafo dado. Inicialmente as imagens foram

convertidas em tons de cinza, em seguida foi aplicado o filtro da mediana, com uma máscara do tamanho 20x20.

Sauvola sugeriu os valores de $R = 128$ e $k = 0.5$ para o algoritmo de segmentação, entretanto, como as imagens foram capturadas a partir de diferentes dispositivos, isto provocou em variações na dimensão, luminosidade e ruído da imagem. Por estes motivos, foram feitos diversos experimentos e como o valor dessas constantes tem o objetivo de controlar a adaptação de T (Equação 2.3), percebeu-se que ao definir R como o máximo valor do desvio padrão dentro da janela e k de acordo com a necessidade da imagem, foram obtidos melhores resultados. Na Tabela 4.2 são exibidos os valores mínimo e máximo da constante k usada para cada tipo de diagrama.

Tabela 4.2: Mínimo e Máximo da constante k utilizada no algoritmo de segmentação

Tipo de Amostra	Constante k (Mín - Max)
(a)	0,5 - 0,9
(b)	0,5 - 0,7
(c)	0,1 - 1,0
(d)	0,5 - 0,7
(e)	0,1 - 0,9
(f)	0,1 - 0,9
(g)	0,3 - 0,9
(h)	0,3 - 0,9

Logo após essa etapa, gera-se o esqueleto da imagem e em seguida encontra-se os *pixels* candidatos, em que pela configuração do esqueleto são pontos que possuem três ou mais vizinhos. No gráfico da Figura 4.9 é exibido a média da quantidade de *pixels* candidatos a nós relevantes para cada tipo de diagrama.

Sendo assim, é perceptível ao observar o gráfico da Figura 4.9, que para cada nó relevante, o sistema encontra de quatro a sete *pixels* candidatos. Isso demonstra a importância na precisão do cálculo da distância mínima, caso contrário, poderá acontecer de em um nó o sistema classificar mais de um *pixel* candidato como relevante.

A seguir, será exibido o passo-a-passo para a obtenção dos resultados do conjunto do tipo (a). Os demais conjuntos realizam os mesmo passos, de maneira que apenas os resultados são diferentes. Sendo assim, os demais tipos de dados serão apenas exibidos e comentados os resultados.

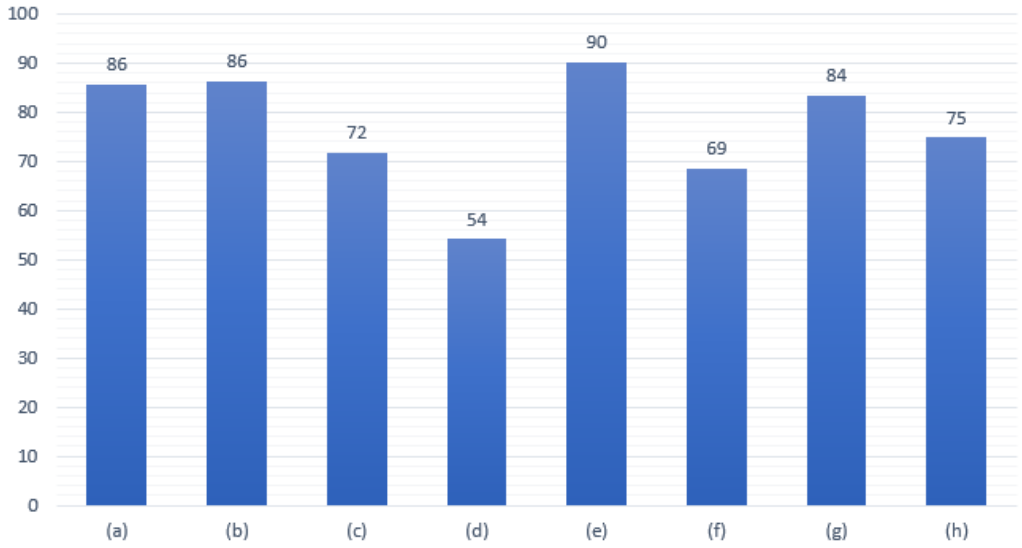


Figura 4.9: Gráfico com a média da quantidade de *pixels* candidatos encontrados por tipo de amostras

Conjunto do tipo (a)

É importante ressaltar que para esse conjunto de dados após a quantização e classificação dos nós, o número de bifurcações e arestas encontradas é igual a seis e doze, respectivamente. Dessa forma, é possível observar na Tabela 4.3 as distâncias mínimas e a quantidade de bifurcações e arestas encontradas para cada amostra.

Tabela 4.3: Resumo das características encontradas para cada amostra do conjunto do tipo (a)

Numeração do Grafo	Distância Mínima	Bifurcações	Arestas
Grafo 1	189,1400	6	12
Grafo 2	136,3805	6	12
Grafo 3	126,3573	6	12
Grafo 4	194,3740	6	12
Grafo 5	113,2733	6	12
Grafo 6	142,0852	6	12
Grafo 7	**182,2766**	**6**	**11**
Grafo 8	**41,7891**	**6**	**11**
Grafo 9	114,3979	6	12
Grafo 10	101,3333	6	12

Ao observar a Tabela 4.3 é perceptível que para os resultados em negrito foram encontrados o número correto de bifurcações, porém 11 arestas. Estas duas amostras

podem ser visualizadas nas Figuras 4.10.(a) e 4.10.(b). Já as Figuras 4.10.(c) e 4.10.(d) exibem os resultados da quantização e classificação dos nós, em que a linha fina amarela indica o esqueleto da imagem. Os círculos amarelo e rosa indicam o nó inicial e final, respectivamente. Os círculos verdes representam os nós de ligação, os brancos as bifurcações e o pontilhado preto simboliza o ponto na imagem em que a distância entre ele até o centro do nó é igual a distância mínima. Por fim, os pequenos quadrados enumerados representam a identificação do nó.

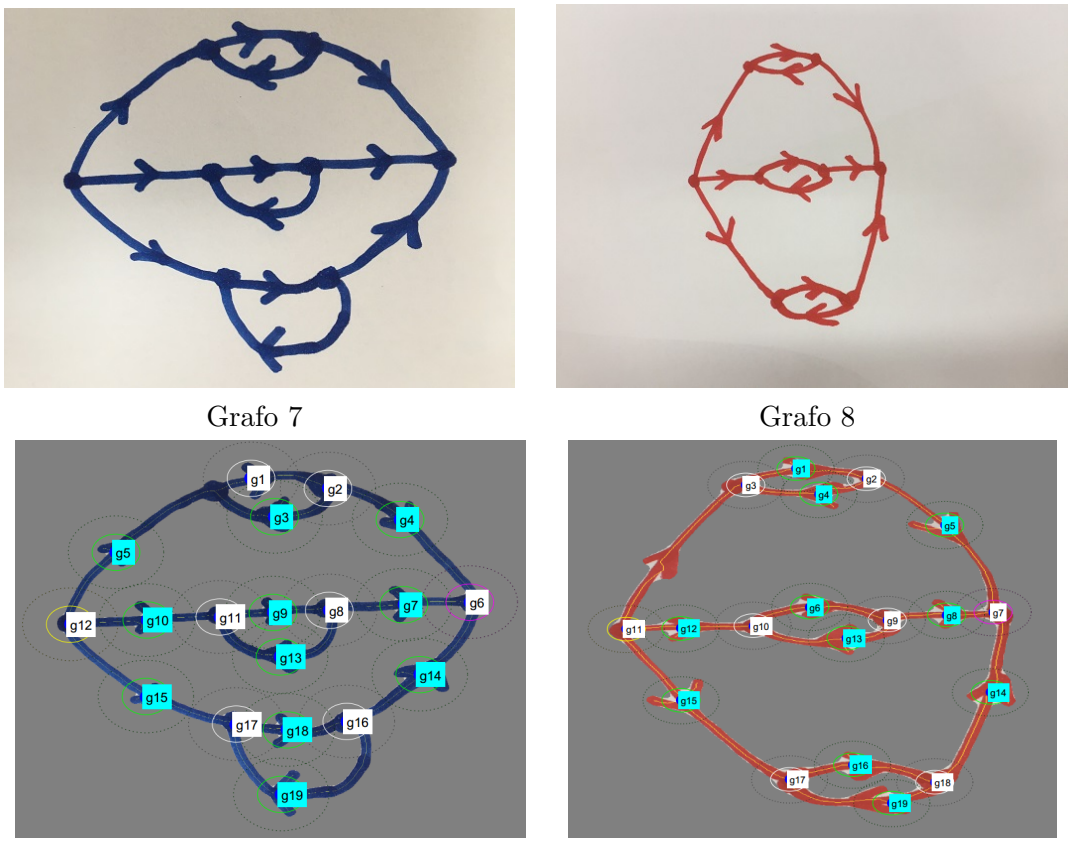
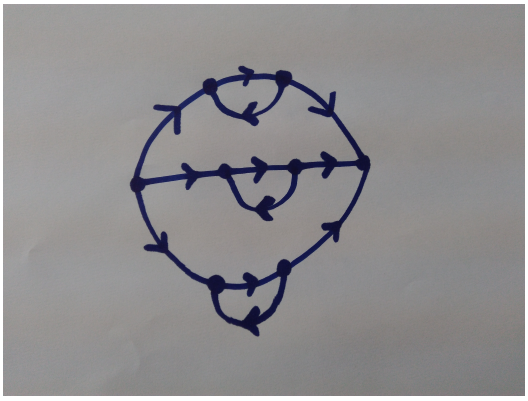


Figura 4.10: Amostras coletadas e os resultados da quantização e classificação dos grafos, respectivamente

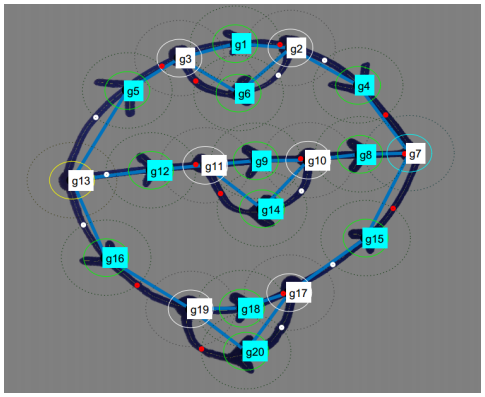
Analisando os resultados apresentados percebe-se que para o grafo 7, a seta no ramo superior não está centralizada. Para uma simples comparação visual, é possível notar que a distância entre ela e a bifurcação vizinha à esquerda é menor do que as distância das outras arestas e seus respectivos vizinhos. Isso se torna mais evidente no resultado da sua quantização (Figura 4.10), em que ao quantizar os *pixels* candidatos o sistema as considerou como uma única bifurcação. Já para o grafo 8, as abas da seta vizinha ao nó inicial na parte superior estão muito fechadas,

dessa maneira, o sistema não calculou o esqueleto da imagem corretamente nesta extensão do grafo e consequentemente não foi identificada uma aresta neste ramo.

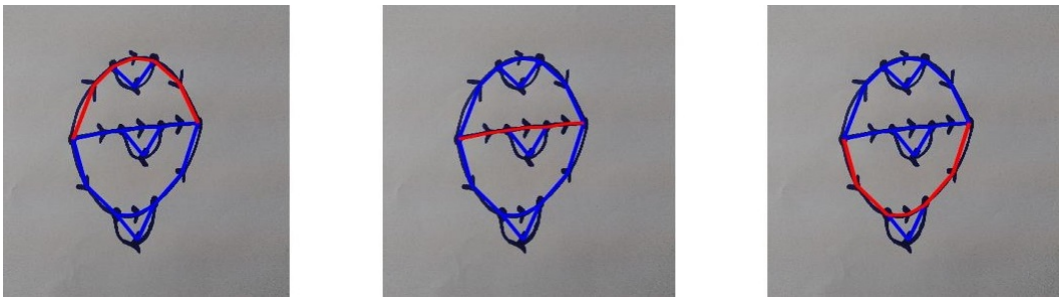
Como já apresentado na Seção 4.1.1, para o conjunto de imagens em que foram detectados e classificados os nós corretamente, o sistema também apresentou a função de transferência precisa. A seguir serão exibidos os resultados obtidos de todas as etapas do Grafo 1.



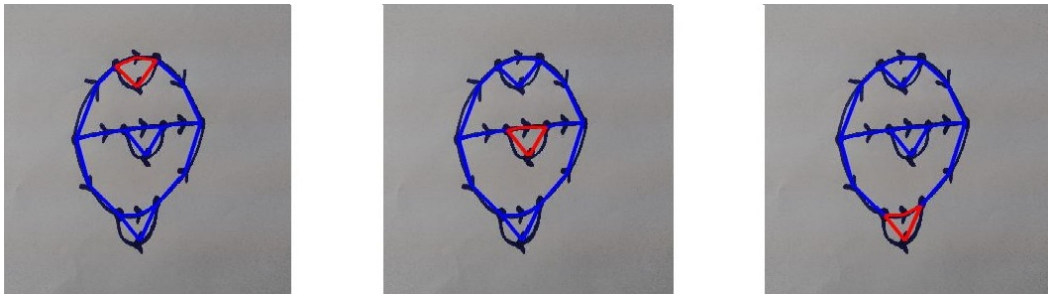
(a) Imagem Original



(b) Resultado das etapas de classificação, conexão dos nós e detecção da direção das arestas



(c) Caminhos diretos encontrados



(d) *Loops* encontrados

Figura 4.11: Resultados obtidos do grafo 1 do conjunto de dados do tipo (a)

A Figura 4.11.(a) exibe a imagem original, já a Figura 4.11.(b) apresenta os resultados obtidos nas etapas de classificação, conexão dos nós e detecção da direção

das arestas. Nesta figura, pode-se observar que ela possui as mesmas referências contidas na Figura 4.10, adicionado as seguintes informações: as linhas azuis representam as conexões dos nós encontradas; cada nó de ligação, representado por um círculo verde, está rodeado por dois círculos preenchidos, na qual a direção da aresta é dada a partir círculo branco para o vermelho. Por fim, as Figuras 4.11.(c) e 4.11.(d) apresentam os caminhos diretos e *loops* encontrados, respectivamente. É possível perceber que as linhas em vermelho representam os caminhos diretos para o primeiro caso e os *loops* para o segundo, já as azuis correspondem aos caminhos restantes.

Conjunto do tipo (b)

A Tabela 4.4 exibe os resultados encontrados para as amostras coletadas. Percebe-se que para o grafo 12, o número de arestas encontradas foi incorreto. Isso pode ser analisado na Figura 4.12, em que a primeira imagem é a amostra coletada e a segunda é o resultado da quantização e classificação dos nós.

Tabela 4.4: Resumo das características encontradas para cada amostra do conjunto do tipo (b)

Numeração do Grafo	Distância Mínima	Bifurcações	Arestas
Grafo 11	132,1584	6	12
Grafo 12	**117,7669**	**6**	**9**
Grafo 13	118,4017	6	12

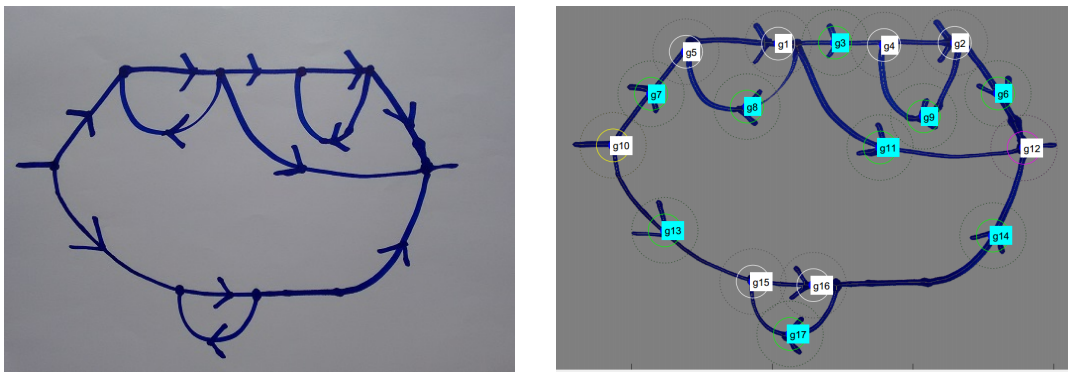


Figura 4.12: Imagem e Resultado da quantização e classificação do grafo 12

Examinando a Figura 4.12, é perceptível que o erro ocorreu devido a uma limitação do sistema já mencionada anteriormente, ou seja, algumas setas estavam muito

próximas de um nó vizinho, gerando em erros no processo de quantização.

Conjunto do tipo (c)

Os resultados para o conjunto do tipo (c) são apresentados na Tabela 4.5. Um ponto importante a ser observado neste conjunto, é que todos estes erros ocorreram por uma única razão: a proximidade de ramos paralelos. A Figura 4.13 exibe a imagem do grafo 27 e o resultado de sua quantização e classificação. Pode-se observar que o círculo que indica o limite da distância mínima calculada para o nó *g1*, está alcançando o ramo paralelo a ele, sendo assim, todos os *pixels* candidatos dentro desse círculo são quantizados em um único nó. O mesmo ocorre para o nó *g9*.

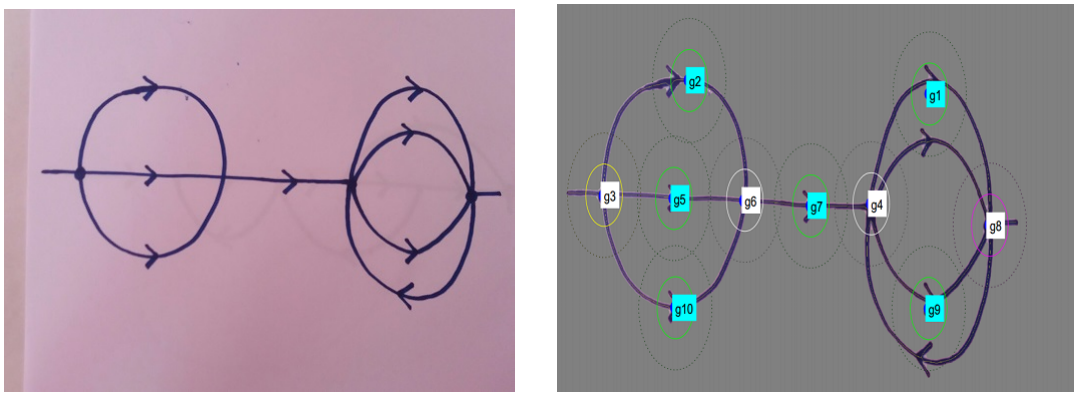


Figura 4.13: Imagem e Resultado da quantização e classificação do grafo 27

Conjunto do tipo (d)

Os resultados do cálculo da distância mínima e a quantidade de bifurcações e arestas para o conjunto do tipo (d) são apresentados na Tabela 4.6. Ao observar estes resultados, percebe-se que todas as arestas e bifurcações foram encontradas corretamente.

Conjunto do tipo (e)

Os resultados para o conjunto do tipo (e) são apresentados na Tabela 4.7. Ao observar estes resultados, é perceptível que seis das dezessete amostras alcançaram um produto incorreto.

Tabela 4.5: Resumo das características encontradas para cada amostra do conjunto do tipo (c)

Numeração do Grafo	Distância Mínima	Bifurcações	Arestas
Grafo 14	221,9397	2	8
Grafo 15	141,2177	2	8
Grafo 16	130,6217	2	8
Grafo 17	208,9812	2	8
Grafo 18	207,1563	2	8
Grafo 19	154,3043	2	8
Grafo 20	59,6570	2	8
Grafo 21	158,4765	2	8
Grafo 22	164,3671	2	8
Grafo 23	185,8722	2	8
Grafo 24	120,3697	2	8
Grafo 25	167,5389	2	8
Grafo 26	171,7943	2	8
Grafo 27	**175,4099**	**2**	**6**
Grafo 28	211,3787	2	8
Grafo 29	219,0750	2	8
Grafo 30	133,0093	2	8
Grafo 31	214,0887	2	8
Grafo 32	**65,4886**	**5**	**8**
Grafo 33	93,0952	2	8
Grafo 34	234,1181	2	8
Grafo 35	**194,6030**	**2**	**6**
Grafo 36	156,4367	2	8
Grafo 37	101,3588	2	8
Grafo 38	144,2470	2	8
Grafo 39	**147,1640**	**2**	**7**
Grafo 40	103,9872	2	8
Grafo 41	196,2009	2	8
Grafo 42	201,9741	2	8
Grafo 43	156,8774	2	8
Grafo 44	161,4416	2	8

Um ponto importante a ser observado neste conjunto são as razões para o percentual de resultados incorretos ser maior que as demais. Isto foi exposto nas Figuras 4.14 e 4.15, em que são exibidos as amostras dos grafos 49 e 58 e seus respectivos resultados. Na imagem do grafo 49, pode-se notar que existe uma bifurcação logo no início, em que os três ramos que estão retornando para ela, foram desenhados se tocando em muitos *pixels*. Com isso, a distância mínima calculada não atingiu toda

Tabela 4.6: Resumo das características encontradas para cada amostra do conjunto do tipo (d)

Numeração do Grafo	Distância Mínima	Bifurcações	Arestas
Grafo 45	226,1189	3	7
Grafo 46	157,3377	3	7
Grafo 47	117,2416	3	7
Grafo 48	151,3255	3	7

Tabela 4.7: Resumo das características encontradas para cada amostra do conjunto do tipo (e)

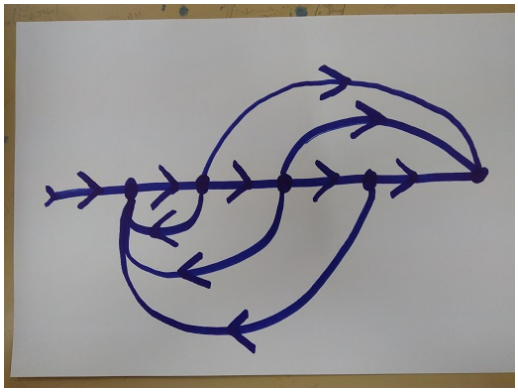
Numeração do Grafo	Distância Mínima	Bifurcações	Arestas
Grafo 49	**163,8702**	**10**	**6**
Grafo 50	90,3819	4	10
Grafo 51	**76,3854**	**8**	**7**
Grafo 52	90,6012	4	10
Grafo 53	125,9787	4	10
Grafo 54	111,1131	4	10
Grafo 55	**93,1005**	**8**	**5**
Grafo 56	124,9091	4	10
Grafo 57	47,8218	4	10
Grafo 58	**133,1387**	**4**	**9**
Grafo 59	81,6180	4	10
Grafo 60	128,4615	4	10
Grafo 61	**91,2627**	**5**	**10**
Grafo 62	126,3884	4	10
Grafo 63	**119,5275**	**7**	**10**
Grafo 64	182,4660	4	10
Grafo 65	96,4426	4	10

essa extensão e a considerou erroneamente como duas ou três bifurcações. O que ocasionou o erro do grafo 58 já foi mencionado anteriormente. Esses pontos ficam mais claros ao analisar os resultados da quantização na Figura 4.15.

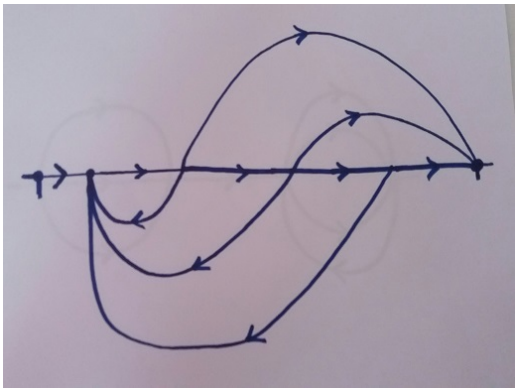
Conjunto do tipo (f)

É possível observar na Tabela 4.8 os resultados obtidos para as amostras do conjunto do tipo (f).

É possível observar que sete entre eles calcularam o total incorreto de bifurcações e/ou arestas. Estes resultados insatisfatórios foram causados por uma situação já exposta anteriormente e para exemplificar, foi escolhida a imagem do grafo 73, na

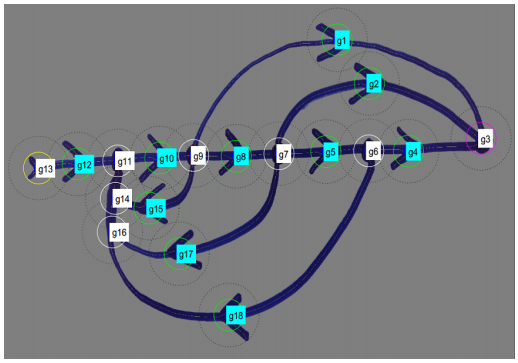


Grafo 49



Grafo 58

Figura 4.14: Amostras coletadas



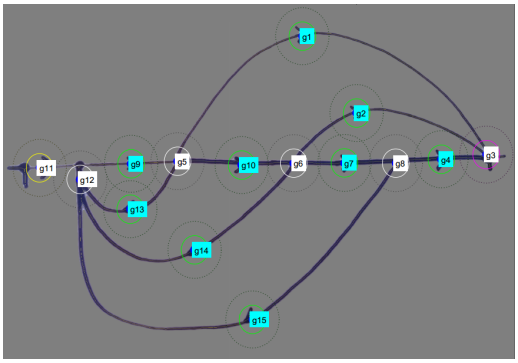


Figura 4.15: Resultado da quantização e classificação dos grafos 49 e 58, respectivamente

qual ela e o seu resultado estão expostos na Figura 4.16.

Conjunto do tipo (g)

Os resultados para o conjunto do tipo (g) são apresentados na Tabela 4.9. Conforme podemos observar, foram encontrados todas as bifurcações e arestas para as amostras com o desenho correto desse tipo de conjunto.

Conjunto do tipo (h)

Na Tabela 4.10 são apresentados os resultados das amostra do conjunto do tipo (h). Observando esta tabela, percebe-se que o sistema encontrou o resultado insatisfatório para três amostras. Os erros encontrados nesses grafos se resumem aos

Tabela 4.8: Resumo das características encontradas para cada amostra do conjunto do tipo (f)

Numeração do Grafo	Distância Mínima	Bifurcações	Arestas
Grafo 66	104,4729	4	8
Grafo 67	58,6582	4	8
Grafo 68	92,8279	4	8
Grafo 69	102,0621	4	8
Grafo 70	94,5143	4	8
Grafo 71	80,1250	4	8
Grafo 72	118,9689	4	8
Grafo 73	**58,6252**	**6**	**6**
Grafo 74	**146,0806**	**5**	**6**
Grafo 75	122,1161	4	8
Grafo 76	72,2908	4	8
Grafo 77	111,0914	4	8
Grafo 78	111,1813	4	8
Grafo 79	191,0613	4	8
Grafo 80	104,7275	4	8
Grafo 81	96,2513	4	8
Grafo 82	**110,2893**	**4**	**7**
Grafo 83	**120,7151**	**4**	**7**
Grafo 84	112,6245	4	8
Grafo 85	**90,7974**	**6**	**5**
Grafo 86	102,6097	4	8
Grafo 87	86,9389	4	8
Grafo 88	96,6816	4	8
Grafo 89	95,3955	4	8
Grafo 90	**149,4666**	**6**	**5**
Grafo 91	122,0487	4	8
Grafo 92	**204,0696**	**7**	**5**

Tabela 4.9: Resumo das características encontradas para cada amostra do conjunto do tipo (g)

Numeração do Grafo	Distância Mínima	Bifurcações	Arestas
Grafo 93	249,4307	4	11
Grafo 94	83,5574	4	11

já descritos anteriormente. Sendo assim, a Figura 4.17 exibi a imagem do grafo 95, como também o resultado da quantização e classificação dos nós.

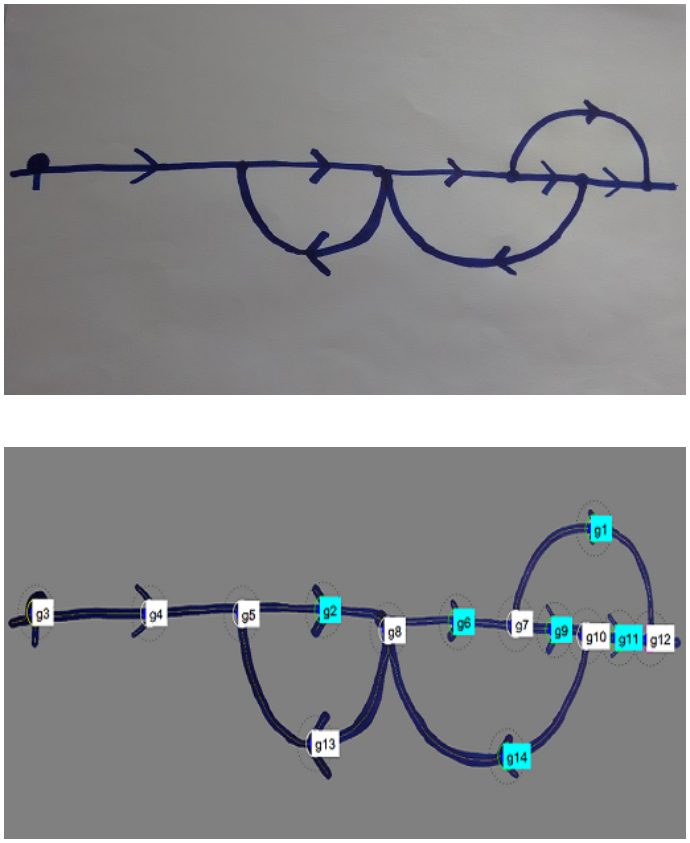


Figura 4.16: Imagem e Resultado da quantização e classificação do grafo 73

Tabela 4.10: Resumo das características encontradas para cada amostra do conjunto do tipo (h)

Numeração do Grafo	Distância Mínima	Bifurcações	Arestas
Grafo 95	**98,7482**	**5**	**10**
Grafo 96	67,4959	6	11
Grafo 97	**81,1193**	**9**	**7**
Grafo 98	**61,9818**	**14**	**4**
Grafo 99	134,3905	6	11
Grafo 100	56,8938	6	11

4.2.3 Considerações

Neste capítulo, foram discutidos os experimentos conduzidos ao longo da pesquisa. Para fins de validação e avaliar a precisão do software proposto, foram obtidas 117 amostras que podem ser classificadas em oito tipos, exibidos na Figura 4.5.

Foi observado que caso o sistema calcule a distância mínima corretamente e os resultados na etapa de quantização e classificação dos nós forem precisos, os

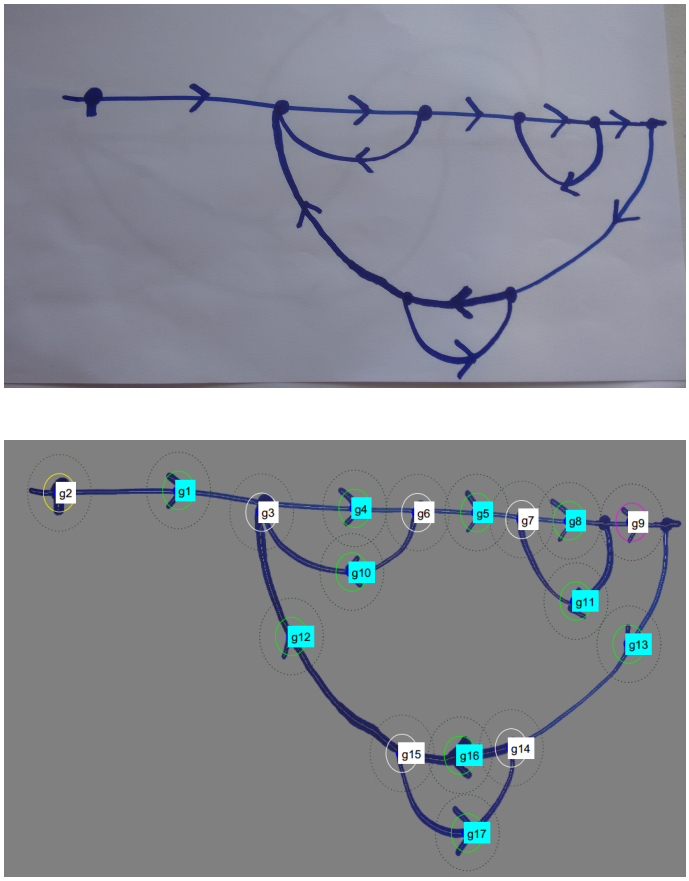


Figura 4.17: Imagem e Resultado da quantização e classificação do grafo 95

resultados nas etapas posteriores também serão satisfatórios. Sendo assim, no gráfico da Figura 4.18, podemos observar a quantidade de desenhos considerados errados, a quantidade de resultados corretos e os incorretos para cada tipo de diagrama, em que a porcentagem exibida nas *labels* do gráfico é referente ao total de amostras do tipo em que ela pertence.

Uma limitação encontrada no sistema foi que o desenho deve ter todos os seus elementos proporcionais, ou seja, a distância entre uma arestas e suas bifurcações vizinhas assim como o tamanho das setas devem ser simétricas, caso contrário, aumenta a probabilidade do resultado não ser satisfatório.

Para verificar a performance do sistema, foi calculado a média do tempo computacional em segundos, exibidos na Tabela 4.11, na qual foi obtido a média de tempo de dez execuções de cada amostra. Para essa simulação foi utilizado um Notebook com um processador Intel Core I5 de 2,3 Ghz, 8 GB de memória RAM (DDR4), placa de vídeo NVIDEA GeForce 920MX e o Sistema Operacional Windows 10.

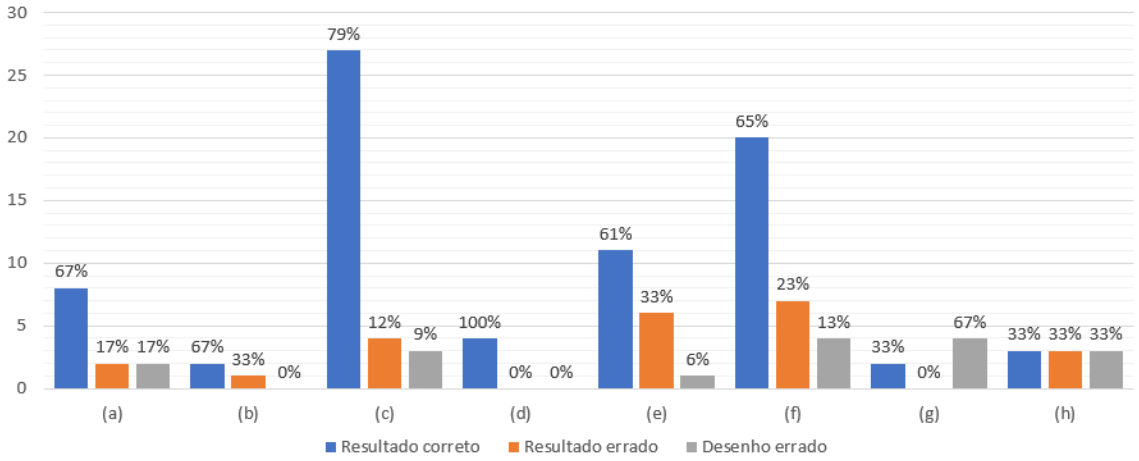


Figura 4.18: Gráfico com o quantitativo de resultados encontrados por tipo de amostras

Nesta tabela, podemos observar que a amostra do conjunto do tipo (g) requer mais tempo de processamento e isso é causado devido a quantidade de caminhos que esse conjunto possui ser maior do que os demais.

Tabela 4.11: Tempo médio em segundos

Tipo de Amostra	**Tempo**
(a)	32,5282
(b)	42,1845
(c)	34,0055
(d)	23,7664
(e)	34,8784
(f)	29,7475
(g)	50,8244
(h)	31,2447

Capítulo 5

Conclusão

Este trabalho teve como objetivo o desenvolvimento de um sistema de processamento de imagem capaz de receber a imagem de um diagrama de fluxo de sinal desenhado a mão e encontrar automaticamente todos os caminhos presentes para então calcular a FT final. Para isso, foi proposto o desenvolvimento de várias técnicas após a etapa de segmentação da imagem: (1) determinar a posição espacial dos nós do grafo; (2) realizar a classificação destes nós, como bifurcação, aresta, nó inicial ou final; (3) Encontrar as conexões dos nós, ou seja, quem são os vizinhos de cada nó; (4) detectar a direção das arestas; (5) encontrar os caminhos diretos; (6) encontrar os *loops*; (7) calcular a FT equivalente.

A técnica de segmentação utilizada neste trabalho foi proposta por Sauvola et. al. (2000), em que apresenta uma limiarização adaptativa e caracteriza-se por calcular diferentes limiares para cada *pixel* da imagem. Esta técnica requer a utilização de alguns parâmetros. O parâmetro W e R não se mostraram críticos, sendo o primeiro definido como [300 300] e o segundo como o máximo valor do desvio padrão dentro da janela, para todas as imagens. Já o parâmetro k se mostrou mais sensível. Como as imagens foram capturadas a partir de diferentes dispositivos, isto provocou em variação na dimensão, luminosidade e ruído da imagem. Dessa forma, percebeu-se que foi necessário utilizar diferentes valores dessa constante para as imagens.

Para a etapa de determinação da posição espacial dos nós, foi possível observar que está foi a etapa mais sensível do sistema, devido ao parâmetro calculado denominado como distância (ou *threshold*) mínima. Nos testes realizados, foi perceptível que os elementos do grafo desenhado devem ser proporcionais, ou seja, a distância entre eles como também o tamanho das setas devem ser simétricos, caso contrário, a probabilidade da distância mínima calculada não ser correta é maior, ocasionando uma FT equivalente errada.

Já para a etapa de classificação, devido a que utiliza a distância mínima calcu-

lada anteriormente para classificar os nós entre bifurcações ou arestas, foi possível perceber que o êxito nesta etapa foi equivalente ao da etapa anterior. Com isso, na realização dos testes e obtenção dos resultados, foi observado que não houve um caso em que se os resultados nestas etapas forem satisfatórios, os produtos obtidos nas etapas subsequentes também serão.

Sobre a base de imagens utilizadas neste trabalho para a validação da abordagem proposta e avaliar sua precisão, aproximadamente 50 universitários do curso de Engenharia Elétrica da UFRN foram convidados a desenhar um dos oito tipos de grafo de fluxo de sinal, propostos por Nise (2007). Dessa forma, foram coletadas 117 amostras. Apesar dos alunos terem sido instruídos sobre o sistema, 17 amostras apresentaram algum erro no desenho.

Os resultados obtidos nesse trabalho foram 67% de acertos para as imagens do conjunto do tipo (a), 67% para as do tipo (b), 79% para as do tipo (c), 100% para as do tipo (d), 61% para as do tipo (e), 65% para as do tipo (f), 33% para as do tipo (g) e por fim, as do tipo (h) obtiveram 33%.

Analisando os resultados obtidos, percebeu-se que o sistema proposto mostrou-se ser útil para calcular a FT final de um diagrama desenhado a mão quando obedece as condições impostas: o nó inicial deve ser o mais à esquerda; o nó final deve ser o mais à direita; as arestas devem ser identificadas por uma pequena seta que deve ser abertas em vez das cheia; as setas não devem tocar nos ramos adjacentes e nos nós vizinhos; por fim, o nó inicial e final devem conter no mínimo três caminhos partindo deles. É importante também levar em consideração que os alunos não tinham nenhuma experiência prévia com o sistema, no entanto, espera-se que à medida que se familiarizem com o sistema, sejam alcançados melhores resultados.

Também sobre a análise dos resultados, foi perceptível que além da desvantagem exposta anteriormente, ou seja, os elementos do grafo serem proporcionais, outra desvantagem encontrada nesta proposta é a necessidade do diagrama ser desenhado com um lápis de ponta grossa, para que o resultado obtido na etapa da segmentação e o esqueleto do diagrama sejam completamente conectados.

Por fim, os resultados indicam que é perfeitamente possível o uso das técnicas propostas para facilitar no cálculo da FT final de sistemas muito complexos, inclusive estas técnicas podem ser implementadas e usadas a partir de dispositivos móveis, como *tablets* e *smartphones*.

Referências Bibliográficas

ARRUDA, A. W. A. d. Binarização de imagens de documentos históricos baseada em imagem de contraste. Universidade Federal de Pernambuco, 2014.

BLUM, H. A transformation for extracting descriptors of shape. MIT press, 1967.

CAO, L.; LOO, K.; LAI, Y. Systematic derivation of a family of output-impedance shaping methods for power converters—a case study using fuel cell-battery-powered single-phase inverter system. *IEEE Transactions on Power Electronics*, IEEE, v. 30, n. 10, p. 5854–5869, 2015.

CASANOVA, D.; COMUNELLO, E. Reconhecimento e representação gráfica da linha mediana cerebral. *Anais SULCOMP*, v. 1, 2012.

DORF, R.; BISHOP, R. *Modern Control Systems, twelfth*. [S.l.]: Addison-Wesley, 2011.

ESQUEF, I. *Tecnicas de entropia em processamento de imagens*. [S.l.]: Rio de Janeiro, 2002.

FILHO, O. M.; NETO, H. V. *Processamento digital de imagens*. [S.l.]: Brasport, 1999.

GENNARI, L. et al. Combining geometry and domain knowledge to interpret hand-drawn diagrams. *Computers & Graphics*, Elsevier, v. 29, n. 4, p. 547–562, 2005.

GONZALEZ, R. C.; WOODS, R. E. *Processamento de imagens digitais*. [S.l.]: Edgard Blucher, 2000.

JU, T.; BAKER, M. L.; CHIU, W. Computing a family of skeletons of volumetric models for shape description. *Computer-Aided Design*, Elsevier, v. 39, n. 5, p. 352–360, 2007.

KARA, L. B.; STAHOVICH, T. F. An image-based, trainable symbol recognizer for hand-drawn sketches. *Computers & Graphics*, Elsevier, v. 29, n. 4, p. 501–517, 2005.

KOTZAPETROS, A. D.; PARASKEVAS, P. A.; STASINAKIS, A. S. Design of a modern automatic control system for the activated sludge process in wastewater treatment. *Chinese Journal of Chemical Engineering*, Elsevier, v. 23, n. 8, p. 1340–1349, 2015.

MASCARENHAS N. D. A. E VELASCO, F. R. D. Processamento digital de imagens. IV EBAI, 1989.

MASON, S. J. *Feedback theory: further properties of signal flow graphs*. [S.l.]: Research Laboratory of Electronics, Massachusetts Institute of Technology, 1956.

MESSELODI, S.; MODENA, C. M.; ZANIN, M. A computer vision system for the detection and classification of vehicles at urban road intersections. *Pattern analysis and applications*, Springer, v. 8, n. 1-2, p. 17–31, 2005.

MOHAMMADIAN, L.; BABAEI, E.; SHARIFIAN, M. B. B. H-ga-pso method for tuning of a pid controller for a buck-boost converter modeled with a new method of signal flow graph technique. *Bulletin of Electrical Engineering and Informatics*, v. 4, n. 1, p. 32–42, 2015.

NASIR, S. B.; LEE, Y.; RAYCHOWDHURY, A. Modeling and analysis of system stability in a distributed power delivery network with embedded digital linear regulators. In: IEEE. *Quality Electronic Design (ISQED), 2014 15th International Symposium on*. [S.l.], 2014. p. 68–75.

NISE, N. S. Control systems engineering, sixth edition. Daniel Sayre, 2007.

OGATA, K. Modern control engineering, fourth edition. Aeeizb, 2002.

OLIVER, N. M.; ROSARIO, B.; PENTLAND, A. P. A bayesian computer vision system for modeling human interactions. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 22, n. 8, p. 831–843, 2000.

OUYANG, T. Y.; DAVIS, R. Recognition of hand drawn chemical diagrams. In: *AAAI*. [S.l.: s.n.], 2007. v. 7, p. 846–851.

RABBANI, M. et al. Hand drawn optical circuit recognition. *Procedia Computer Science*, Elsevier, v. 84, p. 41–48, 2016.

SAUVOLA, J.; PIETIKÄINEN, M. Adaptive document image binarization. *Pattern recognition*, Elsevier, v. 33, n. 2, p. 225–236, 2000.

SENTHILKUMARAN, N.; VAITHEGI, S. Image segmentation by using thresholding techniques for medical images. *Computer Science & Engineering: An International Journal*, v. 6, n. 1, 2016.

SEZGIN, M.; SANKUR, B. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic imaging*, International Society for Optics and Photonics, v. 13, n. 1, p. 146–166, 2004.

SOMASUNDARAM, K.; KALAISELVI, T. A method for filling holes in objects of medical images using region labeling and run length encoding schemes. In: *National Conference on Image Processing (NCIMP)*. [S.l.: s.n.], 2010. p. 110–115.

STAHOVICH, T. F. Segmentation of pen strokes using pen speed. In: *AAAI Fall Symposium Series*. [S.l.: s.n.], 2004. p. 21–24.

SZUMMER, M.; QI, Y. Contextual recognition of hand-drawn diagrams with conditional random fields. In: IEEE. *Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on*. [S.l.], 2004. p. 32–37.

TRIER, O. D.; TAXT, T. Evaluation of binarization methods for document images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, v. 17, n. 3, p. 312–315, 1995.

ZHANG, Q.; LI, B. Discriminative k-svd for dictionary learning in face recognition. In: IEEE. *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. [S.l.], 2010. p. 2691–2698.