MESTRADO PROFISSIONAL EM AUTOMAÇÃO E CONTROLE DE PROCESSOS

MARCUS VINICIUS BARBOSA DE MORAIS

SISTEMA DE VISÃO COMPUTACIONAL PARA VERIFICAÇÃO DE PALETES PARA CONTROLE DE QUALIDADE

SÃO PAULO
2019

MARCUS VINICIUS BARBOSA DE MORAIS

SISTEMA DE VISÃO COMPUTACIONAL PARA VERIFICAÇÃO DE PALETES PARA CONTROLE DE QUALIDADE

Dissertação apresentada ao Instituto Federal de Educação, Ciência e Tecnologia de São Paulo – IFSP, *Campus* São Paulo, como parte dos requisitos para conclusão do curso de Mestrado Profissional em Automação e Controle de Processos.

Orientador: Prof. Dr. Ricardo Pires

Coorientadora: Profa. Dra. Sara Dereste dos Santos

SÃO PAULO
2019

Catalogação na fonte
Biblioteca Francisco Montojos - IFSP Campus São Paulo Dados fornecidos pelo(a) autor(a)

M827s Morais, Marcus Vinicius Barbosa de
Sistema de visão computacional para verificação de paletes para controle de qualidade / Marcus Vinicius Barbosa de Morais. São Paulo: [s.n.], 2019.
80 f.

Orientador: Ricardo Pires
Co-orientadora: Sara Dereste dos Santos

Dissertação (Mestrado Profissional em Automação e Controle de Processos) - Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, IFSP, 2019.

1. Visão Computacional. 2. Automação Industrial. 3. Controle de Qualidade. 4. Palete. I. Instituto Federal de Educação, Ciência e

CDD 629.8

MINISTÉRIO DA EDUCAÇÃO
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO
CAMPUS SÃO PAULO
DIRETORIA GERAL DO *CAMPUS* SÃO PAULO
Coordenadoria de Registros Escolares de Pós-Graduação

ATA DE EXAME DE DEFESA DE DISSERTAÇÃO

Nome do Programa: **Mestrado Profissional em Automação e Controle de Processos**

Nome do(a) Aluno(a): Marcus Vinicius Barbosa de Morais

Nome do(a) Orientador(a): Prof. Dr. Ricardo Pires

Nome do(a) Coorientador(a): Sara Dereste dos Santos

Título do Trabalho: "Sistema de visão computacional para verificação de paletes para controle de qualidade"

Abaixo o resultado de cada participante da Banca Examinadora

Nome completo dos Participantes Titulares da Banca	**Sigla da Instituição**	**Aprovado / Não Aprovado**
Prof. Dr. Ricardo Pires – Orientador	IFSP – SPO	aprovado
Prof. Dr. Alexandre Brincalepe Campo – Membro Interno	IFSP – SPO	Aprovado
Prof. Dr. Carlos Eduardo de Brito Novaes – Membro Externo	INSPER	Aprovado
Nome completo dos Participantes Suplentes da Banca	**Sigla da Instituição**	**Aprovado / Não Aprovado**
Prof. Dr. Eduardo Alves da Costa – Membro Interno	IFSP – SPO	—
Prof. Dr. Leandro Poloni Dantas – Membro Externo	INSPER	—

Considerando-o: [X] APROVADO
[ ] NÃO APROVADO

Assinaturas

São Paulo, 18 de fevereiro de 2019

Ricardo Pires
Presidente da Banca

Membro Interno

Membro Externo

Observações:

DEDICATÓRIA

Dedico a toda minha família, principalmente aos meus pais Sebastiana e Paulo Sérgio, e meus irmãos Paulo Felipe e César Matheus, por todo apoio durante o curso. Dedico também a Tainá, pelo apoio, motivação e companheirismo.

Dedico aos meus colegas de classe que contribuíram com meu crescimento acadêmico e pessoal durante o curso com as trocas de ideias e discussões sempre relevantes.

Por fim, dedico a todos os pesquisadores, que dedicam seu tempo em prol de encontrar soluções para os problemas existentes, contribuindo para que tenhamos um mundo melhor de se viver.

AGRADECIMENTOS

Agradeço ao professor e orientador Dr. Ricardo Pires por todo suporte dado ao longo do caminho. Sua paciência e seu conhecimento são imensuráveis e são exemplos para todos que desejam se tornar pesquisadores.

Agradeço à professora e coorientadora Dra. Sara Dereste por sua visão e contribuição ao longo da jornada. Sua presença mostrou-se fundamental e seu trabalho em conjunto com o Dr. Ricardo Pires me proporcionou as ferramentas necessárias para condução deste trabalho.

Agradeço ao professor Dr. Alexandre Brincalepe Campo por me apresentar o programa e me incentivar no ingresso do mesmo para desenvolvimento deste trabalho.

Agradeço aos professores Dr. Alexandre Simião Caporali e Dr. Thomas Edson Filgueiras Filho pelas contribuições e sugestões dadas ao longo do desenvolvimento deste trabalho.

Agradeço a empresa Hartness SEE Sistemas pelo apoio dado no ingresso ao programa, pelo espaço e material cedido para geração dos dados analisados neste trabalho.

Agradeço a todos os professores do programa, gerando e compartilhando seus conhecimentos, vocês proporcionam crescimento e evolução da sociedade.

RESUMO

Sistemas de Produção Flexíveis estão se tornando cada vez mais presentes nas indústrias, que buscam mais agilidade e, consequentemente, a automação de processos têm crescido. Um desses processos é a paletização robotizada, que visa a organizar e empilhar cargas de modo a facilitar sua movimentação e transporte. No entanto, os processos automatizados ainda apresentam problemas que acabam gerando parada de produção e intervenção humana. No caso da paletização, a qualidade dos paletes de madeira é um desses problemas. A fim de solucionar os problemas causados por esses paletes sem condições de uso para aplicação nas células de paletização robotizadas, esta dissertação apresenta um sistema de visão com programa computacional desenvolvido através da plataforma gratuita OpenCV, que pode diferenciar paletes em condições de uso, de paletes sem condições de uso, utilizando a técnica de aprendizagem de máquina *Support Vector Machine*, evitando paletes malformados, paradas e intervenções humanas. Durante o desenvolvimento do sistema foram procurados métodos de otimizar os valores dos parâmetros presentes nas técnicas escolhidas para o processamento das imagens. Com a otimização desses valores, foi realizada a etapa de classificação, onde as funções Kernel Linear, Polinomial, Sigmoidal e RBF foram comparadas. Os percentuais mais elevados de classificação foram obtidos com as funções Polinomial e RBF, sendo esta a única a atingir 100% de acerto na classificação dos paletes.

Palavras-chave: Visão Computacional, Automação Industrial, Controle de Qualidade, Palete

ABSTRACT

Flexible Production Systems are becoming more and more present in the industries, which are looking for more agility and, consequently, the automation of processes have grown. One such process is robotized palletizing, which aims to organize and stack loads in order to facilitate their movement and transportation. However, the automated processes still present problems that end up generating stop of production and human intervention. In the case of palletizing, the quality of the wooden pallets is one such problem. In order to solve the problems caused by these pallets without condition for application in robotized palletizing cells, this dissertation presents a vision system with a computer program developed through the free OpenCV platform, which can differentiate pallets in conditions of use, for the cited application, from pallets without conditions of use, using the machine learning technique Support Vector Machine, avoiding malformed pallets, stops and human interventions. During the development of the system we searched for methods to optimize the values of the parameters present in the techniques chosen for the image processing. With the optimization of these values, the classification stage was performed, where the Linear Kernel, Polynomial, Sigmoidal and RBF functions were compared. The highest percentages of classification were obtained with the Polynomial and RBF functions, being the last the only one to reach 100% accuracy in the classification of the pallets.

Keywords: Computer Vision, Industrial Automation, Quality Control, Pallet

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

LOG - *Laplacian of a Gaussian*

RBF - *Radial-Basis Function*

SNR - *Signal to noise ratio*

SPF - Sistemas de Produção Flexíveis

SVM - *Support Vector Machine*

VIM - *Visual Inspection Machine*

LISTA DE SÍMBOLOS

a1 - quantidade de linhas presentes com ângulo entre -90° e -30°

a2 - quantidade de linhas presentes com ângulo entre -30° e 30°

a3 - quantidade de linhas presentes com ângulo entre 30° e 90°

B - constante de parametrização

C - parâmetro Função Kernel

c1 - constante que maximiza a variável Λ

cp1 - quantidade de linhas presentes com comprimento menores que 800 pixeis

cp2 - quantidade de linhas presentes com comprimento entre 800 e 1600 pixeis

cp3 - quantidade de linhas presentes com comprimento maiores que 1600 pixeis

cx - comprimento da coordenada X

cy - comprimento da coordenada Y

d - valor do acumulador da Transformada de Hough

f(x) - função imagem

G(x) - função filtro

$gamma$ - parâmetro Função Kernel

h - parâmetro Função Kernel

k - Constante ($\frac{X_{max}}{W}$)

M - quantidade de pontos de uma imagem

m - quantidade de pontos escolhidos de maneira aleatória em uma imagem

n_0 - amplitude média do ruído

Nθ - quantidade de θ detectados

Nρ - quantidade de ρ detectados

O() - quantidade de operações para Transformada de Hough

r - parâmetro Função Kernel – coeficiente 0

r - razão entre o erro na detecção e o erro por múltiplas respostas

X - eixo das abscissas

X1 - coordenada das abscissas do ponto inicial do segmento detectado pela Transformada de Hough

X2 - coordenada das abscissas do ponto final do segmento detectado pela Transformada de Hough

xi - coordenada das abscissas dos pontos da imagem

x_{max} – distância entre dois máximos na função ruído

Y - eixo das ordenadas

Y1 - coordenada das ordenadas do ponto inicial do segmento detectado pela Transformada de Hough

Y2 - coordenada das ordenadas do ponto final do segmento detectado pela Transformada de Hough

yi - coordenada das ordenadas dos pontos da imagem

W - tamanho do operador

α - constante de parametrização

θ - ângulo formado entre a reta normal e o eixo das abscissas

θi - ângulo processado a partir de um ponto da imagem

Λ - variável de medição de desempenho dependente do filtro

ρ - comprimento do segmento de reta normal ao segmento de reta da imagem

ρi - comprimento processado a partir de um ponto da imagem

Σ - variável de medição de desempenho dependente do filtro

SUMÁRIO

1. INTRODUÇÃO

A automação industrial tem como principal fundamento o aumento de produção, fazendo os processos e produtos com mais qualidade e em menor tempo (GOEKING, 2013). E esse tem sido o principal objetivo das indústrias para investir em automação. Aguiar (2013) argumenta que a outra motivação para a implementação de sistemas automatizados é a substituição de processos rudimentares e prejudiciais aos trabalhadores. Porém, reforça que a indústria tem necessidade de maior agilidade e desempenho em diversos processos, entre eles a paletização.

Dessa necessidade, surgem os Sistemas de Produção Flexíveis (SPF), que estão se tornando cada vez mais presentes nas fábricas, mas ainda necessitam ser aperfeiçoados em diversos aspectos. Um exemplo de SPF é a célula de paletização robotizada, que tem como objetivo organizar diferentes tipos de cargas de maneira a melhorar seu transporte e armazenagem. Da experiência profissional do autor, ao participar de diversos projetos de paletização robotizada, percebeu-se que existem problemas que limitam a produtividade da máquina e, algumas vezes, gera custo, como por exemplo:

- Produtos a serem paletizados mal acabados, exemplo: caixa fora do esquadro;
- Produtos a serem paletizados fora de especificação, exemplo: caixa fora de dimensão;
- Insumos do cliente fora de especificação, exemplo: vazão de ar comprimido;
- Qualidade dos paletes de madeira, exemplo: paletes com ripas faltantes ou quebradas.

Dentre os problemas citados, o problema mais recorrente é com relação à qualidade dos paletes de madeira, pois os mesmos tendem a serem danificados naturalmente com o uso. Os paletes sem condições de uso para uma célula de paletização, quando acabam entrando nos sistemas misturados com os paletes em condições de uso geram outros problemas, como por exemplo:

- Paletes enroscados nos transportadores;
- Queda dos produtos sobre os paletes, podendo levar a avaria dos mesmos;
- Colisões entre o órgão terminal do robô e um palete fora das dimensões.

A inspeção humana para evitar que paletes sem condições sejam barrados antes de entrarem no sistema é muito utilizada, porém a precisão dos humanos em tarefas de inspeção é próxima de 68% apenas. Dentre os fatores que influenciam negativamente a capacidade de inspeção humana estão fadiga, fatores emocionais e até mesmo doenças como gripe ou resfriado (HUBER *et al apud* PATRICIO e MARAVALL, 2006). É possível aplicar algumas soluções mecânicas baseadas no conceito *poka yoke* (CALARGE e DAVANSO, 2014) para

evitar problemas causados por paletes ruins. No entanto, essas soluções não conseguem abranger toda a necessidade de verificação para garantir que um palete está em condições de uso para o sistema, pois um palete pode apresentar diversos defeitos como lascas, rachaduras, fissuras, elementos quebrados ou fora de dimensão (PATRICIO e MARAVELL, 2006).

Com o desenvolvimento de um sistema de visão para verificação de paletes é possível ter um sistema que abrange as necessidades de verificação dos paletes e que agrega grande valor ao sistema, evitando a queda de produtividade das células de paletização. Além disso, a utilização da técnica de aprendizagem de máquina *Support Vector Machine* garante ao sistema de visão a flexibilidade de atuar com diferentes padrões de paletes.

Sistemas de visão já estão presentes na indústria, exercendo, por exemplo, funções de verificação de produtos e peças. Esses sistemas têm como objetivo a identificação de padrões e auxiliam nos processos de qualidade das empresas. Portanto um sistema de visão para verificação de paletes nada mais é que uma vertente de sistemas de verificação, que identificará padrões e auxiliará na qualidade e desempenho das indústrias.

1.1. OBJETIVO

Desenvolver um sistema de visão para verificação e seleção de paletes de madeira em condições de uso em uma célula de paletização robotizada.

1.2. OBJETIVOS ESPECÍFICOS

- Selecionar técnicas que possibilitem resultados mais precisos para o processamento das imagens através de estudos teóricos;
- Verificar na prática a validade dos conceitos teóricos das técnicas de detecção de borda e caracterização linhas;
- Desenvolver método de otimização para definição dos parâmetros de detecção de borda e de caracterização de linhas nas imagens de paletes;
- Validar método de cálculo de comprimentos e ângulos de linhas a partir de suas coordenadas iniciais e finais;
- Reduzir lacuna de trabalhos com desenvolvimento e testes de sistemas de verificação de paletes.

1.3. MOTIVAÇÃO

A motivação deste projeto é a necessidade de reduzir a quantidade de paradas das células de paletização robotizadas, causadas por paletes sem condições de uso, especificamente paletes com ripas, que são as madeiras superiores, danificadas ou ausentes, já que por experiência do autor esse tipo de defeito ocorre com alta frequência e é muito prejudicial para o sistema.

Por ser um sistema presente no final da linha de produção, as paradas das células de paletização sempre são muito prejudiciais para todo o processo fabril, tornando as células de paletização um fator limitante da capacidade produtiva das fábricas.

Paletes sem condições de uso para a citada aplicação geram diversas paradas para intervenção de operadores que precisam corrigir algum problema causado por eles.

A falta de ripas ou tocos em um palete pode ser prejudicial na formação final, pois pode levar a queda e até perda de produtos, devido à má formação das camadas. Com a queda de produtos na linha de transporte, é necessária intervenção de operadores para retirar os produtos para que a linha volte a funcionar normalmente.

Com isso, o sistema proposto poderia reduzir o número de paradas nas células de paletização, contribuindo, assim, para maior produtividade das fábricas.

1.4. ORGANIZAÇÃO DO TEXTO

Para melhor entendimento da presente dissertação, o texto foi dividido em 6 capítulos.

No capítulo 1, o capítulo introdutório, foi apresentado o problema de pesquisa, os objetivos propostos, a relevância do tema, motivação para a execução deste trabalho e por fim a organização do texto.

No capítulo 2 é apresentada a fundamentação teórica, na qual são caracterizados tópicos e técnicas relevantes ao trabalho, como imagem, pixel, célula de paletização, sistema de visão computacional, detecção de borda, caracterização de linhas e classificação de dados.

O capítulo 3 consiste na revisão bibliográfica referente ao tema de desenvolvimento do trabalho, que são os sistemas para verificação de paletes. Nele são apresentados sistemas de verificação por contato e por imagem que foram desenvolvidos ao longo dos anos. É importante destacar que, como descrito no capítulo 4, existe uma lacuna na literatura sobre sistemas de verificação de paletes.

Para o desenvolvimento dos capítulos 2 e 3 foi utilizado majoritariamente o portal de periódicos da CAPES, o CAFe, disponível para alunos do Instituto Federal de Educação,

Ciência e Tecnologia de São Paulo. Os idiomas utilizados nas pesquisas foram português e inglês.

No capítulo 4 os materiais e métodos utilizados no desenvolvimento do trabalho são apresentados e caracterizados, detalhando a captação das imagens, o desenvolvimento do programa computacional do sistema de verificação e testes executados.

No capítulo 5 são apresentados os procedimentos experimentais realizados e a análise dos resultados obtidos.

E por fim, no capítulo 6 são apresentadas as conclusões obtidas bem como sugestões de melhorias e continuação dessa pesquisa.

2. FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo, são caracterizados os principais tópicos que estão envolvidos no projeto, como: descrição de uma célula de paletização robotizada e de sistema de visão computacional, detalhando suas principais características, além das técnicas utilizadas no processamento e classificação das imagens, sendo o detector de bordas Canny para detecção de bordas, a Transformada de Hough para caracterização de linhas e a *Support Vector Machine* para classificação das imagens. A Figura 1 apresenta um resumo com a estrutura da proposta de funcionamento do sistema de visão.

Figura 1 – Proposta de funcionamento do sistema de visão

(Fonte: Autor)

As pesquisas de cada uma das técnicas utilizadas no processamento e classificação das imagens foram iniciadas com o levantamento dos artigos ou patentes que deram origem a essas técnicas. Na sequência, foi estudada a evolução delas ao longo do tempo. E, por fim, trabalhos que avaliaram e validaram a utilização delas foram estudados.

2.1. CÉLULA DE PALETIZAÇÃO ROBOTIZADA

Segundo Aguiar (2013), existe a necessidade de introduzir maior agilidade nos sistemas de produção industriais. Com isso, surgem os SPF, onde os robôs manipuladores desempenham um papel fundamental, executando tarefas de carga/descarga, pintura, solda e manipulação.

Paletização consiste na ação de organizar caixas entre outros produtos em camadas, formando um paralelepípedo que é chamado de palete ou palete completo, para diferenciar da estrutura sobre a qual os produtos são organizados (AGUIAR, 2013). O principal objetivo da paletização é melhorar o transporte de carga. Existem diversas formas de paletização, são elas: manual, semiautomática, automática convencional e automática robotizada. Para a paletização robotizada, utilizam-se robôs manipuladores de 4 eixos com diferentes tipos de órgãos terminais, dependendo da aplicação.

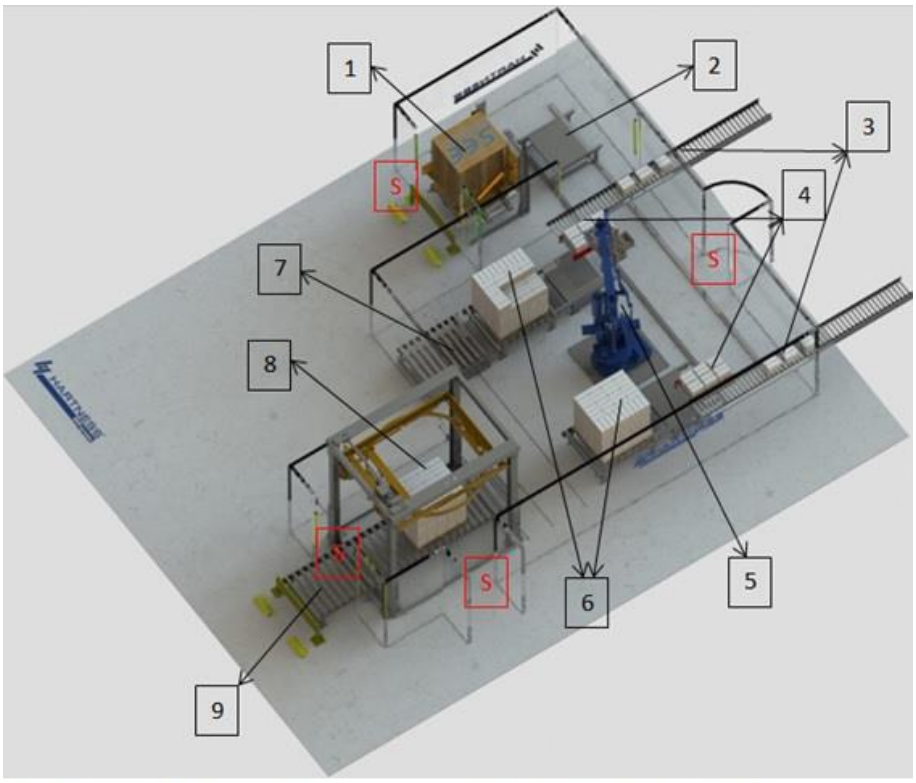
Por ser um SPF, as células de paletização podem ter elementos e tamanhos variáveis, com os seguintes elementos e processos:

- alimentação de produtos;

- alimentação de paletes vazios;
- saída de palete cheio;
- robô manipulador;
- sensores para automação;
- sensores para segurança.

A Figura 2 apresenta alguns dos elementos de células de paletização robotizadas.

Figura 2 – Caracterização de uma célula de paletização



(Fonte: Autor)

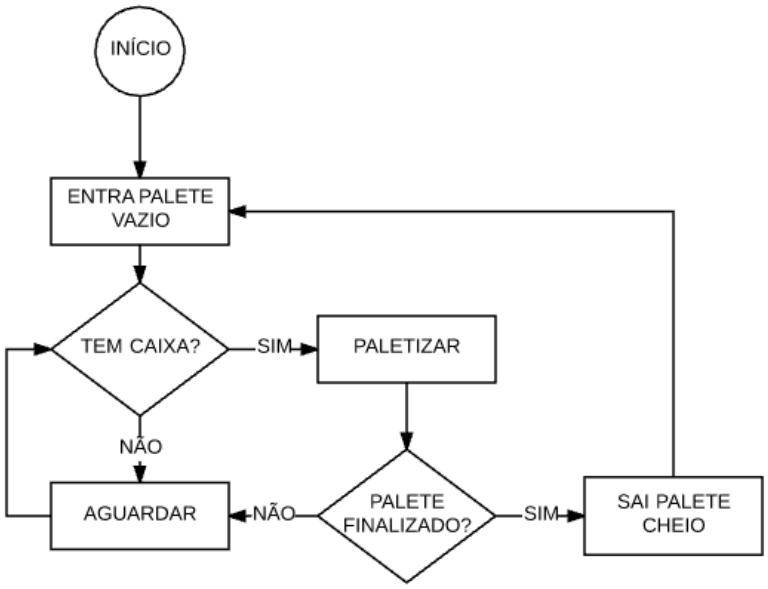
Os itens destacados possuem as seguintes características:

1. É o ponto de alimentação da pilha de paletes vazios. Essa pilha, normalmente formada por 10 paletes, é dosada através de uma parte da máquina chamada magazine de paletes. Os paletes dosados um a um são entregues no item 2;
2. O carro de translação de paletes vazios é uma máquina com rodas guiadas por um trilho que faz o movimento de ir e vir, entregando os paletes vazios nos transportadores das posições 6;
3. A esteira de alimentação de produtos é interligada na linha de produção e leva os produtos até as posições 4;

4. O ponto de pega do robô é local no qual o robô vai buscar os produtos pré-arranjados para depositar nos paletes vazios nas posições 6;
5. O robô manipulador, responsável pela manipulação das caixas das posições 4 até as posições 6;
6. A estação de formação de paletes é um transportador no qual o palete vazio é indexado para receber os produtos;
7. É fisicamente igual ao carro de translação de paletes vazios, porém transporta paletes completos;
8. A estação de envolver o palete com filme ajuda a manter a estabilidade dos paletes, evitando queda de produtos ao longo do transporte;
9. Por fim a estação de saída do palete completo, na qual o palete será retirado e enviado para estoque ou transporte.

O fluxograma de funcionamento básico de uma célula de paletização robotizada é apresentado na Figura 3.

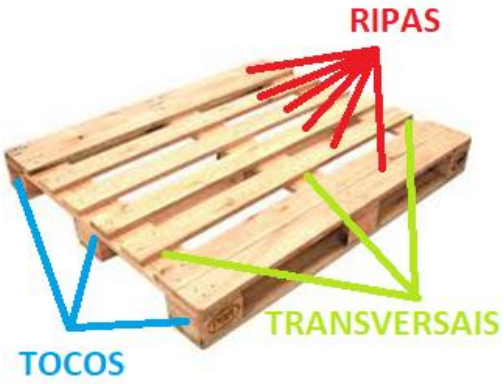
Figura 3 – Fluxograma célula de paletização robotizada



(Fonte: Autor)

O palete, estrutura normalmente de madeira sobre a qual os produtos são depositados de maneira ordenada, possui três componentes: ripas, tocos e transversais. Um palete com seus componentes pode ser visto na Figura 4.

Figura 4 – Palete e seus componentes



(Fonte: Autor)

2.2. SISTEMAS DE VISÃO COMPUTACIONAL

Imagem digital é uma função bidimensional, com variáveis binárias que representam coordenadas espaciais, como cor e forma, permitindo seu armazenamento e reprodução por meios eletrônicos (DJAJADI *et al*, 2010). Pixel é a menor unidade de uma imagem digital, é cada um dos pontos que formam a imagem.

Sistemas de visão computacional podem utilizar recursos de aprendizado para tomar decisões baseadas nas entradas visuais do sistema para emular a visão humana (GONZALEZ; WOODS, 2002 *apud* SEMIM, 2012) e muitas vezes encontrar detalhes imperceptíveis aos olhos humanos. Eles têm como objetivo obter um conjunto de técnicas e metodologias para dar suporte às aplicações práticas baseadas em imagens (SEMIM, 2012). Eles são desenvolvidos e utilizados para aplicações que necessitam de funções visuais, capturando e reconhecendo objetos visíveis e padrões (DJAJADI *et al*, 2010).

Segundo Semim (2012), a tarefa de inspeção visual é uma tarefa muito comum. Esse processo busca informações como dimensão, posição, forma entre outros. Os defeitos são um dos alvos dos sistemas de visão (DAVIES, 2005 *apud* SEMIM, 2012).

Ao fazer a inspeção visual, normalmente, é tomada uma decisão baseada no reconhecimento de padrões. Segundo Castro e Prado (2002), padrões são as propriedades que possibilitam o agrupamento de objetos semelhantes em uma classe ou categoria.

Existem três fases no reconhecimento de padrões: representação dos dados de entrada e mensuração, extração das características e identificação e classificação do objeto (CASTRO e PRADO, 2002).

Sensores de visão, processadores de digitalização de imagem e iluminação são os elementos que compõem um sistema de visão computacional. Além disso, é desenvolvido um algoritmo em um computador, para analisar a imagem e encontrar as informações procuradas

nela. A iluminação é um ponto crucial dos sistemas de visão, dado que é um desafio realizar a captação e análise da imagem com a variação de luz natural do ambiente. Por isso, é necessário ter um elemento de iluminação associado ao sistema de visão (MATO *et al*, 2013).

Os sensores de visão, normalmente câmeras, podem ser divididos em dois grupos (SONKA *et al*, 2014):

- Baseadas no princípio de emissão de fótons: exploram o fenômeno fotoelétrico;
- Baseadas no princípio fotovoltaico: muito utilizado em semicondutores.

Existem diversos tipos de câmeras, digitais e analógicas, baseadas nesses dois princípios e a utilização de cada uma delas varia de acordo com a aplicação.

Dentre os diversos procedimentos para processar as imagens, Djajadi *et al* (2010) listam os seguintes:

- Redução de ruído – Ao retirar o ruído, que atrapalha na análise da imagem, é possível focar a atenção apenas no objeto desejado;
- Remoção de fundo – Nesse procedimento, duas imagens são utilizadas, uma sem o objeto de interesse e outra com o objeto. Pixeis correspondentes das duas imagens são comparados, levando assim ao destaque do objeto desejado;
- Erosão e dilatação – Utilizadas para retirar pequenos ruídos da imagem. Enquanto o método de erosão retira pixeis, o método de dilatação os adiciona.

Sonka *et al* (2014) destacam ruído, brilho, contraste, tamanho e posição como os principais problemas a serem tratados pelos algoritmos de processamento de imagens. Ruído está presente em praticamente todo tipo de medida e normalmente precisa ser eliminado para que a imagem seja analisada. Brilho e contraste, em geral, estão ligados à iluminação. O tamanho está diretamente ligado ao processamento, mas, com o avanço da tecnologia de processadores, imagens maiores podem ser analisadas sem problemas. A posição é essencial, pois a imagem deve mostrar o objeto em questão em condições de ser analisado.

Ao ser processada por um computador, a imagem é digitalizada e, após esse processo, pode ser representada em uma matriz retangular (SONKA *et al*, 2014). Nessa matriz, informações como o brilho de cada ponto são armazenadas. O que o computador "vê" são apenas os números dessa matriz (BRADSKI e KHAELER, 2008).

2.3. DETECÇÃO DE BORDAS

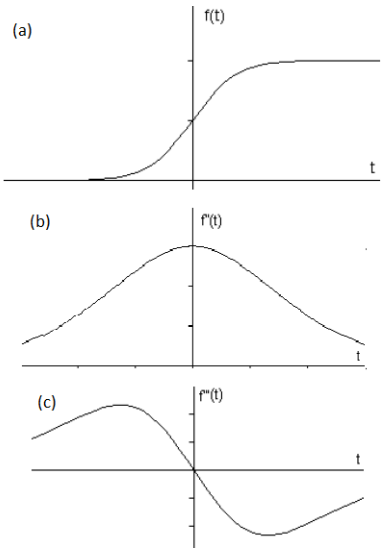
O processo de detecção de bordas é utilizado para simplificar a imagem, reduzindo a quantidade de dados a serem processados, mas mantendo informações estruturais importantes dos limites dos objetos (CANNY, 1986).

Segundo Maini e Aggarwal (2009), detecção de bordas é o processo de identificar e localizar descontinuidades em uma imagem, dadas por mudanças abruptas nas intensidades dos pixeis, caracterizando uma borda. Em geral, a técnica mais utilizada é a convolução entre a imagem e um operador. Existem duas classes de operadores de detecção de bordas, são elas:

- Baseadas em Gradientes: Procuram por pontos de máximo e mínimo na primeira derivada da imagem;
- Baseadas no método Laplaciano: Procuram por cruzamentos no eixo X na segunda derivada da imagem.

A Figura 5 apresenta um exemplo de sinal (a), sua primeira derivada (b) e sua segunda derivada (c).

Figura 5 – Métodos de Detecção de Borda: (a) sinal da imagem, (b) primeira derivada, (c) segunda derivada



(Fonte: Maini e Aggarwal, 2009)

No método baseado em gradientes, os pontos de máximo são comparados com um limiar determinado. Caso o ponto analisado possua valor acima do valor do limiar, ele é considerado como pertencente à borda. Quando se obtém um ponto máximo na primeira derivada, esse ponto irá cruzar o eixo X na segunda derivada. E nisso se baseia o método Laplaciano, buscando pontos que cruzam o eixo X na segunda derivada.

Segundo Nadernejad *et al* (2008) a qualidade da detecção de bordas depende de diversos fatores como: iluminação; presença de objetos de intensidades similares; densidade

das bordas na imagem e ruído. Em geral, esses fatores podem ser corrigidos alterando o valor do limiar do operador de detecção, porém esses valores devem ser alterados manualmente e variando de caso a caso, dado que não existe nenhuma técnica para alterá-los automaticamente.

A maioria dos operadores de detecção de bordas utilizam imagens em tons de cinza, pois dessa maneira a quantidade de dados a serem processados é reduzida, pois em tons de cinza a imagem fica com apenas um canal, ao invés dos três canais (RGB) das imagens coloridas. A exceção se dá por operadores específicos para trabalhar com cores. (NADERNEJAD *et al*, 2008).

Maini e Aggarwal (2009) apresentam em seu estudo uma comparação entre as seguintes técnicas de detecção de bordas:

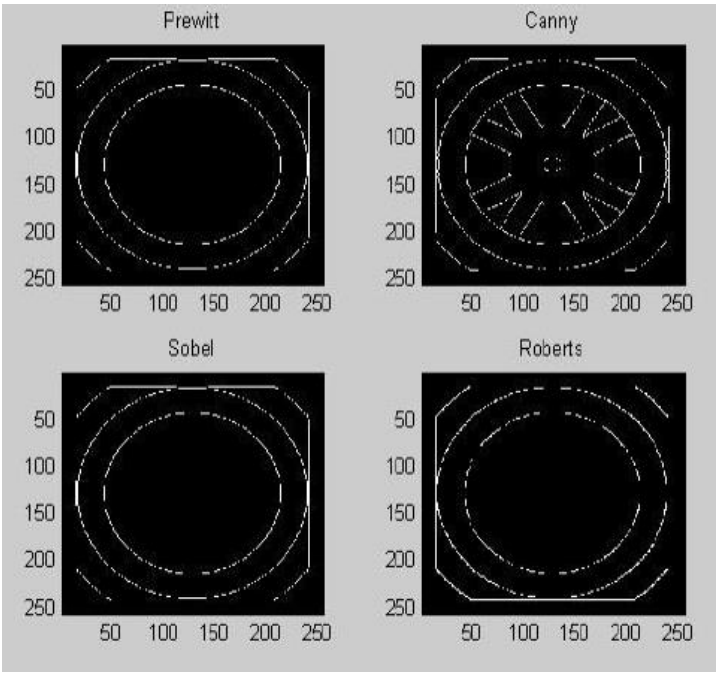
- Operador Sobel;
- Operador Robert Cross;
- Operador Prewitt;
- LOG (*Laplacian of a Gaussian*);
- Canny.

Os autores concluíram, através de análises visuais nas imagens resultantes dos testes com aplicação dos detectores de borda, que o detector de bordas de Canny é o que gera os melhores resultados em praticamente todos os cenários de teste, porém dependendo do ajuste dos seus parâmetros. Uma desvantagem do detector de bordas de Canny em comparação com os demais é que ele exige mais processamento, no entanto, em grande parte por conta da histerese de limiares presentes na técnica, ela é menos sujeita a ruído em comparação com as demais. A Figura 6 apresenta uma das imagens utilizadas pelos autores na realização dos testes, na qual os operadores de detecção de bordas foram aplicados, e a Figura 7 apresenta o resultado do teste dos autores.

Figura 6 – Imagem original para teste com diversos operadores de detecção de bordas

(Fonte: Maini e Aggarwal, 2009)

Figura 7 – Resultado do teste com diversos operadores de detecção de bordas



(Fonte: Maini e Aggarwal, 2009)

Por fim, Maini e Aggarwal (2009) ordenam os classificadores, levando em consideração análises visuais nas imagens sob condições de ruído, da seguinte maneira:

1. Canny;
2. LOG (*Laplacian of a Gaussian*);
3. Operador Sobel;
4. Operador Prewitt;
5. Operador Robert Cross.

Nadernejad *et al* (2008) também apresentam um estudo comparativo entre diferentes técnicas de detecção de bordas, são elas:

- Canny;
- Canny adaptado para cores;
- Booleano;
- Marr-Hildreth;
- *Euclidean Distance and Vector Angle*.

Os autores testaram as técnicas em diferentes imagens, alternando entre fotos reais e imagens geradas virtualmente e analisaram caso a caso, levando em consideração os seguintes critérios:

- Probabilidade de falsos positivos;
- Probabilidade de falsos negativos;

- Erro na estimativa de ângulo das bordas;
- Média da raiz quadrada da distância entre a borda real e a borda detectada;
- Tolerância a bordas distorcidas, junções e curvas a 90°.

Os autores concluíram que o detector Booleano apresentou resultados próximos do detector de bordas de Canny, porém ainda assim o detector de Canny foi preferido, por produzir bordas mais contínuas e de largura de um pixel. Os detectores que utilizam cores têm potencial para serem melhores que os detectores que utilizam escala de cinza, porém, até mesmo para o detector de bordas de Canny adaptado para cores, é necessário encontrar uma maneira ótima para unir os três canais, pois a aplicação é realizada em cada um dos três separadamente. Os autores utilizaram um método aditivo para unir os três canais, mas dizem que deve haver algum método mais eficiente.

2.3.1. DETECTOR DE BORDAS DE CANNY

O detector de bordas de Canny (1986) foi desenvolvido com base em três critérios, são eles:

- Boa detecção;
- Boa localização;
- Evitar múltiplas respostas para a mesma borda.

Segundo Nadernejad *et al* (2008), o detector de bordas de Canny é conhecido como o detector de bordas padrão para a indústria.

Canny (1986) tomou como base trabalhos que estavam sendo desenvolvidos na época com a necessidade de detecção de bordas para considerar os dois primeiros critérios. O terceiro critério foi descoberto como necessário com o andamento da pesquisa.

O primeiro critério se trata de eliminar ruído para não detectá-lo como borda, que seriam falsos positivos, nem perder bord as reais por se parecerem ruído, que seriam falsos negativos. O segundo critério leva em consideração a distância entre os pontos que realmente pertencem à borda de um ponto que possivelmente seja da borda para garantir que a localização de todos os pontos classificados como borda seja coerente. O terceiro critério diz respeito à quantidade de respostas de uma única borda. Ela deve ser limitada, pois evitando múltiplas respostas para uma mesma borda diminui a chance de considerar ruído como parte daquela borda (CANNY, 1986).

Em seu trabalho, Canny (1986) descreve métodos matemáticos para cada um dos critérios, descreve a problemática para definição do melhor filtro a ser utilizado e apresenta resultados de sua técnica.

O autor formulou matematicamente cada um dos critérios, são eles:

- Boa detecção;

$$SNR = \frac{\left|\int_{-W}^{+W} G(-x)f(x)dx\right|}{n_0\sqrt{\int_{-W}^{+W} f^2(x)dx}}$$

Onde:

SNR – *Signal to noise ratio;*

$G(x)$ – Função filtro;

$f(x)$ – Função imagem;

n_0 - amplitude média do ruído.

- Boa localização;

$$Localização = \frac{\left|\int_{-W}^{+W} G'(-x)f'(x)dx\right|}{n_0\sqrt{\int_{-W}^{+W} f'^2(x)dx}}$$

- Evitar múltiplas respostas para a mesma borda.

$$N_n = \frac{2W}{x_{max}} = \frac{2}{k}$$

Onde:

W – Tamanho do operador (limite das integrais dos critérios anteriores);

x_{max} – Distância entre dois máximos na função ruído;

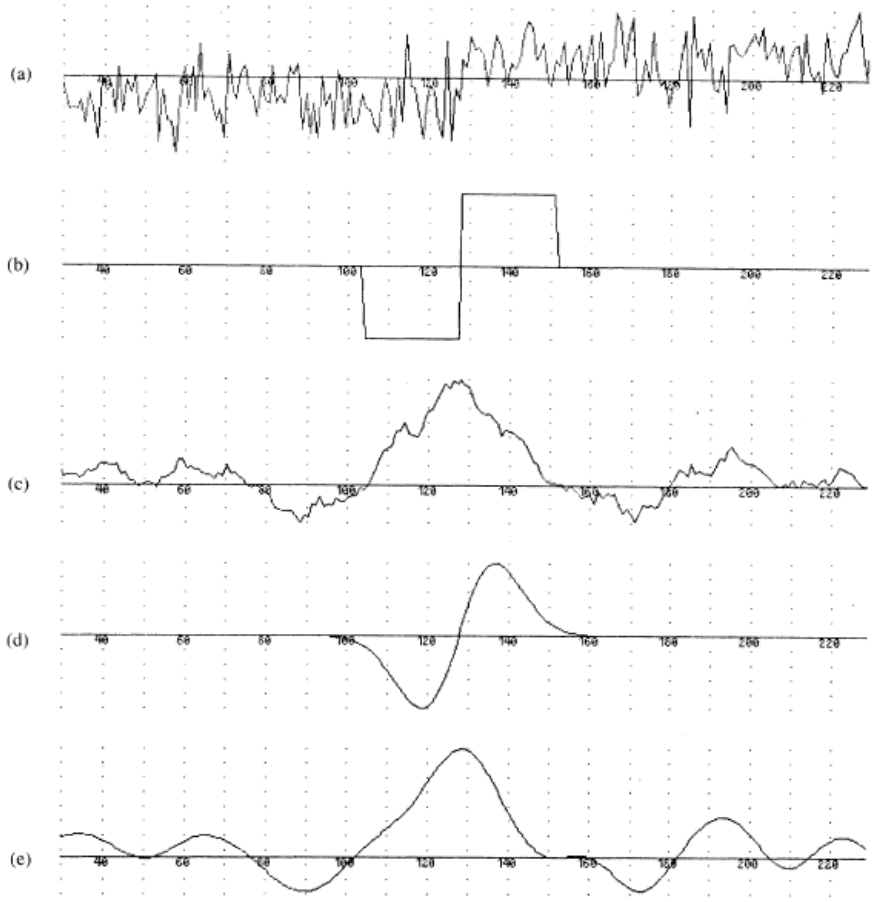
k = Constante ($\frac{X_{max}}{W}$).

Canny (1986) também coloca algumas condições para as imagens, são elas:

- Análise em uma dimensão, considerando a seção transversal local como uma constante em certa direção;
- A imagem consiste em um degrau com ruído Gaussiano branco;
- Para a análise, é realizada a convolução entre o sinal da imagem e um filtro;
- Na função resultante dessa convolução, os pontos de máximo serão considerados como pertencentes à borda.

A Figura 8 apresenta um exemplo de sinal da imagem (a), dois filtros diferentes (b e d) e a função (c) resultante da convolução de (a) com (b) e a função (e) resultante da convolução de (a) com (d).

Figura 8 – Exemplo de convolução entre função imagem e função filtro, onde: (a) sinal da imagem, (b) exemplo de filtro, (c) resultante da convolução de (a) com (b), (d) primeira derivada de uma Gaussiana (e) resultante da convolução de (a) com (d)



(Fonte: Canny, 1986)

Dado que cada filtro resultará em um sinal diferente e uma infinidade de filtros poderia ser utilizada, Canny (1986) definiu o critério para escolher o filtro mais próximo do ideal para sua técnica. Para isso, ele definiu *r* como sendo a razão entre o erro na detecção e o erro por múltiplas respostas. Dos testes feitos com diversos filtros, que não foram definidos pelo autor, foi gerada a Tabela 1.

Tabela 1 – Desempenho de filtros

N	x_{max}	ΣΛ	r	α	ω	β
1	0,15	4,21	0,215	24,59550	0,12250	63,97566
2	0,3	2,87	0,313	12,47120	0,38284	31,26860
3	0,5	2,13	0,417	7,85869	2,62856	18,28800
4	0,8	1,57	0,515	5,06500	2,56770	11,06100
5	1,0	1,33	0,561	3,45580	0,07161	4,80684
6	1,2	1,12	0,576	2,05220	1,56939	2,91540
7	1,4	0,75	0,484	0,00297	3,50350	7,47700

(Fonte: Canny, 1986)

As variáveis α, ω e β presentes na Tabela 1 são definidas pelo autor como constantes desconhecidas. O filtro que gerou o melhor resultado segundo o critério de Canny (valor de *r* mais próximo de 1) foi o filtro número 6, que, segundo Canny, pode ser aproximado para a primeira derivada de uma Gaussiana. No entanto, ele deixa claro que essa opção de filtro não deve ser considerada a única e nem mesmo a melhor possível para detecção de bordas, mas foi a escolhida por ele para satisfazer aos seus critérios.

Canny (1986) percebeu que além de não existir um filtro perfeito (que teria o valor de *r*=1), existe uma correlação entre os dois primeiros critérios que impede que a detecção de bordas seja perfeita.

Considerando um degrau como função da imagem, tem-se:

$$SNR = \frac{A\left|\int_{-W}^{0} f(x)dx\right|}{n_0\sqrt{\int_{-W}^{+W} f^2(x)dx}}$$

$$Localização = \frac{A|f'(0)|}{n_0\sqrt{\int_{-W}^{+W} f'^2(x)dx}}$$

Isolando a amplitude do degrau e a amplitude média do ruído, duas funções foram definidas, Σ e Λ.

$$SNR = \frac{A}{n_0}\Sigma(f)$$

$$Localização = \frac{A}{n_0}\Lambda(f')$$

Logo:

$$\Sigma(f) = \frac{\left|\int_{-W}^{0} f(x)dx\right|}{\sqrt{\int_{-W}^{+W} f^2(x)dx}}$$

$$\Lambda(f') = \frac{|f'(0)|}{\sqrt{\int_{-W}^{+W} f'^2(x)dx}}$$

O autor conclui que um filtro mais amplo daria uma resposta melhor para o primeiro critério, enquanto um filtro mais limitado daria uma resposta melhor para o segundo critério e o mais surpreendente é que ambos os critérios são inversamente proporcionais, crescendo ou diminuindo numa taxa de $\sqrt{w}$, onde w é uma constante que deve ser uma fração de W .

$$\Sigma(f_w) = \sqrt{w}\Sigma(f)$$

$$\Lambda(f'_w) = \frac{1}{\sqrt{w}}\Lambda(f')$$

Portanto, por não ser possível melhorar os dois critérios simultaneamente, Canny (1986) definiu que deveria encontrar um valor c1 para o primeiro critério que daria o melhor resultado para o segundo critério.

$$\Sigma(f_1) = c1, no\ qual\ \frac{1}{\sqrt{w}}\Lambda(f')\ \text{é}\ maximizado$$

Segundo Canny (1986), o valor do operador w é definido automaticamente por seu operador em implementações computacionais, pois, em cada caso, esse valor para otimização será diferente. Para calculá-lo, é realizada uma estimativa da distribuição probabilística do ruído. Estimar o ruído é muito importante, pois o desempenho do sistema depende dessa estimativa, que é utilizada nos dois primeiros critérios.

Para estimar o que é ruído na imagem, o autor utilizou o filtro de Wiener. Após a estimativa e o cálculo médio do valor, é realizada uma distribuição probabilística de ruído ao longo da imagem. No entanto, por melhor que seja essa estimativa, não apenas por conta da correlação entre os critérios e o filtro não ser o ideal, o sistema pode detectar ruído como borda por utilizar limiares (CANNY, 1986).

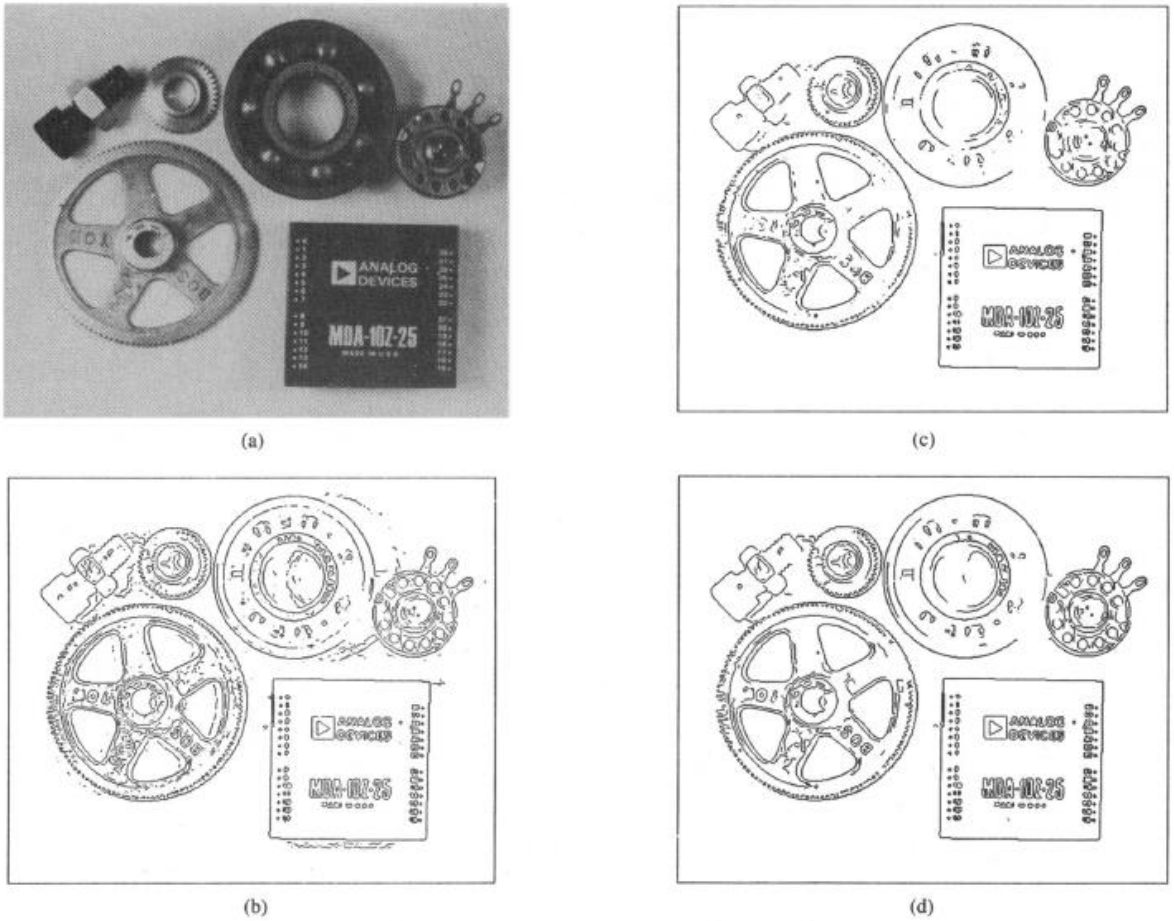
Na realidade, a maneira como Canny utiliza limiares tende a ser mais eficiente do que a dos demais detectores de borda, pois a maioria deles utiliza apenas um limiar. Nesse caso, quando o valor do ponto detectado for maior que o limiar ele é considerado borda, quando for abaixo não é considerado borda. Dessa maneira, encontrar um valor para o limiar que não detecte ruído como borda e ao mesmo tempo não perca informações importantes é mais complexo em comparação ao método de utilizado por Canny. O detector de bordas de Canny utiliza 2 limiares (superior e inferior) e faz a seguinte análise:

- Caso um ponto esteja acima do limiar superior, será considerado borda;
- Caso um ponto esteja abaixo do limiar inferior, não será considerado borda;

- Caso um ponto esteja entre os dois limiares, será borda se o ponto imediatamente ao lado for borda.

A Figura 9 mostra a aplicação da detecção de bordas em uma imagem (a) com um limiar baixo (b) limiar alto (c) e os dois limiares do detector de bordas de Canny (d).

Figura 9 – Aplicação do detector de bordas de Canny, onde: (a) imagem base, (b) resultado da aplicação com um limiar baixo, (c) resultado da aplicação com limiar alto, (d) resultado com os dois limiares do detector de bordas de Canny



(Fonte: Canny, 1986)

É possível perceber que, com um limiar baixo, uma grande quantidade de ruído é considerada borda e que com um limiar alto, bordas importantes são perdidas. Apesar de não ser perfeita, detector de bordas de Canny de utilizar 2 limiares, que ele chama de histerese, é mais efetiva que as outras duas opções (CANNY, 1986).

Canny (1986) explica também que, para cada imagem, será necessário aplicar valores de limiares diferentes, dependendo da quantidade de ruído, e conclui que seu operador é capaz de detectar bordas arbitrárias de maneira otimizada.

2.4. TRANSFORMADA DE HOUGH

Existem diversos métodos de detecção de objetos a partir de imagem, seja para um objeto bidimensional ou tridimensional. Segundo Borrmann *et al* (2011) a Transformada Hough é um dos métodos mais utilizados para detecção de linhas e círculos de objetos bidimensionais e é a base para diversos outros métodos de detecção de objetos, tanto bidimensionais quanto tridimensionais.

Na análise de imagens, é muito comum que se utilizem linhas para caracterizar os objetos, porque até mesmo formas mais complexas podem ser descritas como um conjunto de linhas. A Transformada de Hough é a técnica mais popular para detecção de linhas em imagens (GUERREIRO e AGUIAR, 2012).

Hough (1962) descreveu a técnica de reconhecimento de padrões complexos que deu origem ao método de caracterização de formas, que hoje é conhecido como Transformada de Hough. Mais especificamente, ele desenvolveu um método para que máquinas possam fazer o reconhecimento de linhas, descritas como complexas, em fotografias.

A principal motivação para o desenvolvimento dessa técnica foi a necessidade de enxergar fenômenos físicos em dimensões cada vez menores. No caso, o principal objeto de estudo eram os rastros criados por partículas carregadas em uma câmara de bolhas. Especialistas passavam horas analisando as fotografias tiradas para identificar padrões nesses rastros e a intenção de Hough foi criar uma maneira para que um computador pudesse fazer essas análises.

Para que a análise fosse feita, Hough definiu uma série de regras a serem aplicadas em todos os pontos das linhas encontradas nas imagens. Essas regras fazem uma correlação entre o plano real (no caso uma imagem) e o plano de transformação. Então, ao processar uma imagem, ela será passada para o plano de transformação de acordo com as regras e o resultado será analisado, definindo as características das linhas encontradas (HOUGH, 1962).

As regras para a transformação são:

- Para cada ponto de um segmento de reta no plano real, será desenhada uma linha no plano de transformação;
- O ângulo de inclinação da reta no plano de transformação, tendo como referência a vertical e não a horizontal como de costume, possui tangente proporcional ao deslocamento vertical com relação à linha central horizontal no plano real, sendo que:
    - Caso o ponto esteja no topo do plano real, a inclinação será de 45° para a direita;

- Caso o ponto esteja no meio do plano real, a reta será vertical (0°);
- Caso o ponto esteja na base no plano real, a inclinação será de 45° para a esquerda.

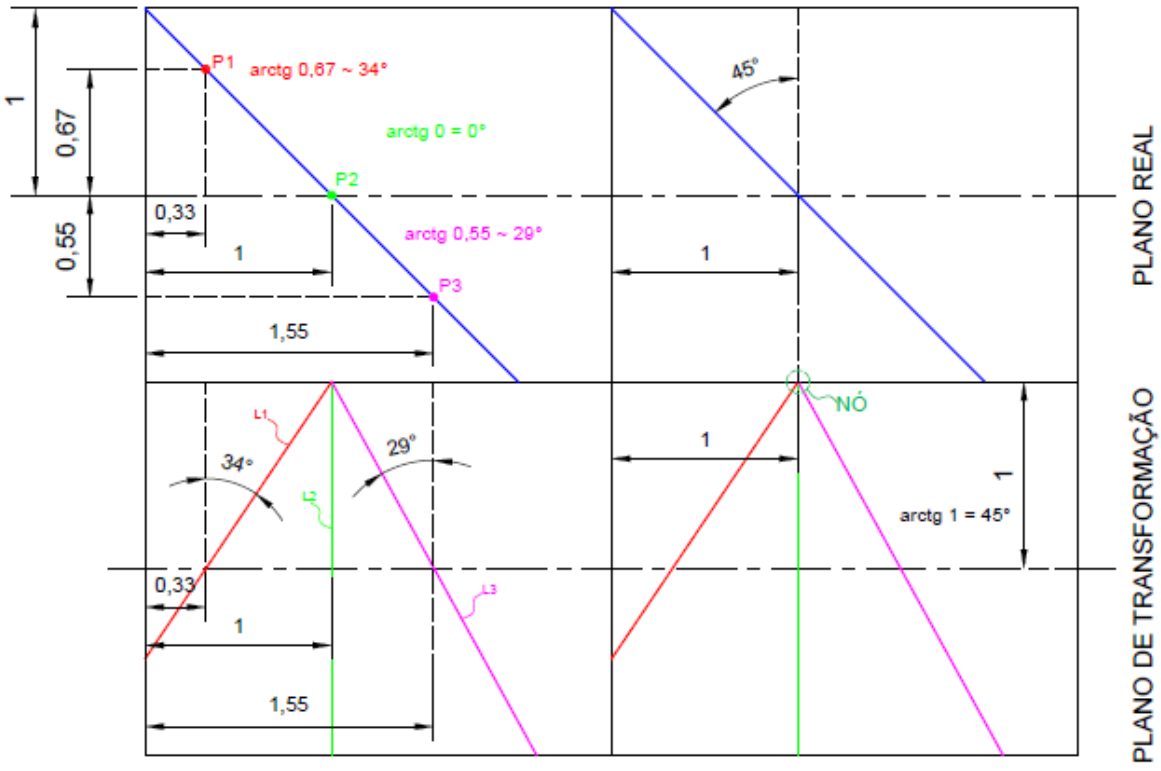
- As linhas no plano de transformação cruzam a linha central horizontal do plano na coordenada horizontal do ponto no plano real.

Se os pontos no plano real estiverem em uma linha reta, então as linhas desenhadas no plano de transformação se cruzarão em único ponto, denominado nó. Os nós, que são, portanto, o ponto de cruzamento das retas traçadas no plano de transformação, têm as seguintes características:

- A coordenada horizontal do nó é igual à coordenada horizontal onde o segmento de reta original cruza a linha central horizontal do plano real;
- A distância do nó para a linha central horizontal do plano de transformação é proporcional à tangente do ângulo de inclinação do segmento de reta no plano real, tendo também como base a vertical.

A Figura 10 ilustra as regras descritas por Hough através de 3 pontos selecionados, indicando suas coordenadas no plano real e a correlação delas com as características das linhas geradas por esses pontos no plano de transformação.

Figura 10 – Plano real e plano de transformação



(Fonte: Autor)

Dessa maneira, é possível transformar todos os pontos identificados em uma imagem para o plano de transformação e identificar quais deles pertencem a segmentos de retas, bem como sua localização (HOUGH, 1962).

Hough (1962) sugere que o processo pode ser feito eletronicamente, porém em sua patente ele apresenta apenas o conceito de sua técnica, fazendo os exemplos manuscritos.

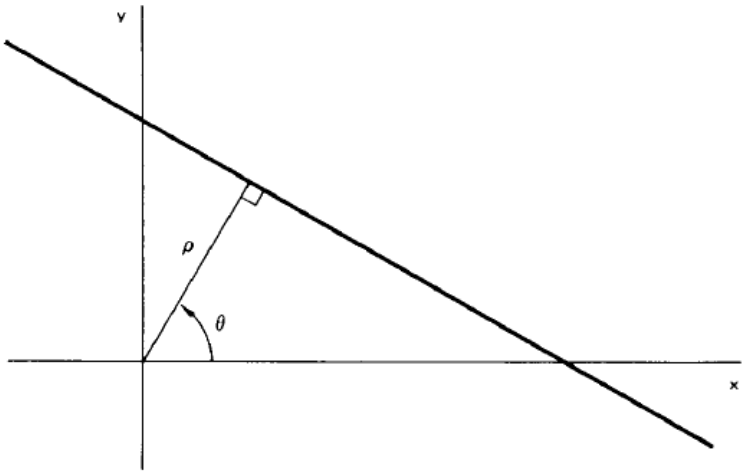
Duda e Hart (1972) apresentam a Transformada de Hough como ela é utilizada atualmente, baseando-se na técnica apresentada por Hough, porém aprimorando-a para que possa ser aplicada de maneira mais simples e eficiente em um computador.

Nessa versão da transformada, é proposto que se faça a transformação com base em distâncias e ângulos e não em inclinação e interceptação como Hough sugeriu. Os autores nomearam esse conjunto de distâncias e ângulos de parametrização normal.

Todo e qualquer segmento de reta presente em um plano pode ser representado pelo conjunto (θ,ρ), onde ρ é o comprimento do segmento de reta normal ao segmento de reta da imagem, isto é, forma um ângulo de 90° entre elas, que representa a distância da origem até a reta e θ é o ângulo formado entre a reta normal e o eixo das abscissas (DUDA E HART, 1972).

A Figura 11 apresenta a parametrização sugerida.

Figura 11 – Parametrização normal de um segmento de reta



(Fonte: Duda e Hart, 1972)

A equação que representa essa geometria é dada por:

$$x \cos\theta + y \operatorname{sen}\theta = \rho$$

Restringindo o valor de θ ao intervalo [0,π], então o conjunto (θ,ρ) será único para toda e qualquer linha e, assim, cada segmento de reta no plano x-y será representado por apenas um ponto no plano de parâmetros θ-ρ.

Supondo m pontos em uma imagem, o objetivo é saber quais deles são colineares. Para isso, os autores propõem que para cada ponto (x_i,y_i) sejam traçados os possíveis conjuntos (θ,ρ) de todas as retas possíveis para esse ponto no plano de parâmetros, através da equação:

$$x_i \cos\theta + y_i \operatorname{sen}\theta = \rho$$

Dessa maneira, as curvas correspondentes aos pontos colineares terão um ponto de intersecção em comum. Esse ponto define o segmento de reta no plano da imagem.

Os autores resumem o processo nas seguintes propriedades:

- Um ponto no plano da imagem corresponde a uma curva no plano de parâmetros;
- Um ponto no plano de parâmetros corresponde a um segmento de reta no plano da imagem;
- Pontos colineares no plano da imagem correspondem a curvas que passam por um ponto em comum no plano de parâmetros;
- Pontos presentes na mesma curva no plano de parâmetros correspondem a linhas que passam por um ponto em comum no plano da imagem.

Os autores pontuam, no entanto, que aplicar esse processo para uma imagem com alto número de pontos seria exaustivo, já que o processamento necessário cresce de maneira quadrática em relação ao número de pontos. Uma alternativa para aliviar essa carga de processamento seria admitir um erro tanto em θ quanto em ρ.

Assim, cada ponto da imagem gera diversos pares (θ,ρ). Cada vez que um determinado par (θ_i,ρ_i) é processado, é somado 1 em no acumulador referente ao par em questão, no caso (θ_i,ρ_i). Essa etapa é chamada de etapa dos votos, o processo de somar 1 em determinado acumulador chama-se voto. Se ao final do processo esse acumulador possui valor d, ou seja, possui d votos, isso significa que o segmento de reta caracterizado por aquele acumulador (θ_i,ρ_i) possui d pontos, com uma certa margem de erro adotada (DUDA E HART, 1972).

Os acumuladores formam na realidade um histograma, no qual apenas os pares com mais votos representam linhas reais na imagem, pois podem existir diversos pontos muito próximos a alguns segmentos de reta que geram valores altos, mas que não fazem parte do segmento, e que, portanto, não devem ser considerados.

Os autores salientam que, quanto menores forem os erros admitidos para θ e ρ, mais valores serão computados e mais processamento será necessário. Dessa maneira, a resolução do resultado em geral será melhor, exceto quando muitos pontos próximos forem considerados como pertencentes a um segmento de reta que não deveriam. Outro ponto levantado é que a técnica encontra grupo de pontos colineares sem levar em consideração a

continuidade do segmento de reta em questão e que, por conta disso, o resultado pode ser distorcido por interferência de pontos não pertencentes àquele segmento de reta, mas que estão presentes em outra região da imagem de maneira quase colinear.

Por fim, Duda e Hart (1972) concluem que a técnica por eles apresentada pode ser generalizada e utilizada para outras formas além de segmentos de retas, desde que escolhidos os parâmetros corretos. Eles exemplificam com um círculo que pode ser representado da seguinte maneira:

$$(x - a)^2 + (y - b)^2 = c^2$$

A transformação deverá ser feita do plano x-y para o plano de parâmetros tridimensional a-b-c. O acumulador tridimensional com maior valor representa um círculo na imagem. E assim, a técnica pode ser utilizada para toda e qualquer curva parametrizável.

Ao longo dos anos, diversos estudos envolvendo a Transformada de Hough foram desenvolvidos, gerando também diversas versões da técnica de Paul Hough. Mukhopadhyay e Chaudhuri (2014) realizaram um estudo geral sobre o assunto, realizando um levantamento bibliográfico em mais de 200 artigos e trabalhos publicados até então, levantando as diferentes versões da transformada, suas características e aplicações.

Segundo os autores, as principais versões da Transformada de Hough são:

- Transformada de Hough Padrão;
- Transformada de Hough Generalizada;
    - Transformada de Hough Generalizada Dinâmica;
    - Transformada de Hough Generalizada com Perspectiva de Transformação Invariante.
- Transformadas baseadas em probabilidade;
    - Transformada de Hough Probabilística;
        - Transformada de Hough Probabilística Progressiva.
    - Transformada de Hough Aleatória.
- Transformada de Hough Digital;
- Transformada de Hough *Fuzzy.*

Dentre as principais motivações para a geração das diferentes versões da Transformada de Hough estão redução do tempo de processamento e tentativa de detecção de objetos mais complexos.

2.4.1. TRANSFORMADA DE HOUGH PROBABILÍSTICA

A Transformada de Hough Probabilística, apresentada por Kiryati *et al* (1990), foi desenvolvida com o objetivo de reduzir o tempo de processamento em comparação com a Transformada de Hough padrão, mas com a mesma capacidade de desempenho. Ela foi desenvolvida com base no conceito de parâmetros sugerido por Duda e Hart (1972), porém com o conceito de utilizar uma quantidade reduzida de pontos como entrada para a transformada.

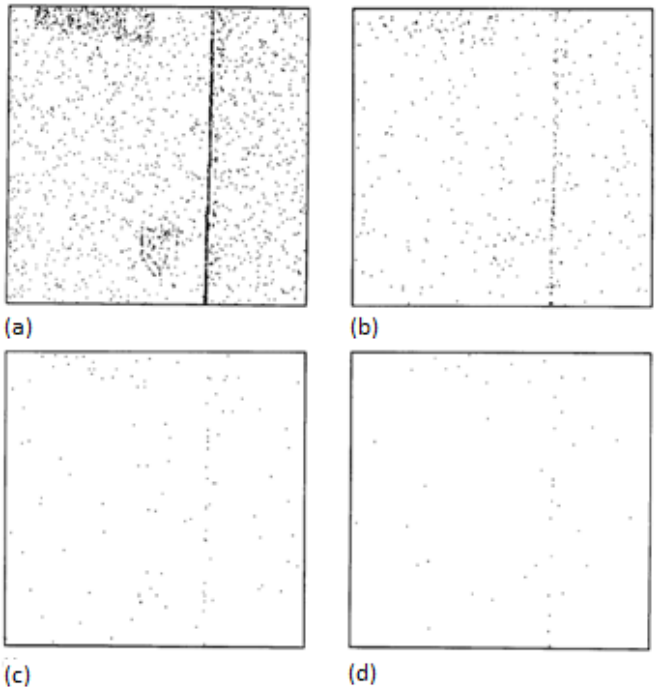
Kiryati *et al* (1990) explicam que o algoritmo proposto por Duda e Hart possui uma característica "um para muitos", isto é, um ponto no plano real (imagem) se torna muitos pontos no plano de transformação. Outras versões da transformada utilizam o conceito "muitos para um" ou "um para um". Por possuir essa característica de "um para muitos" o tempo gasto para realizar a transformação pode ser extremamente grande, dependendo da quantidade de pontos M.

Os autores pontuam que o longo número de operações necessárias, que estão diretamente ligadas ao tempo de processamento, era um obstáculo para que a transformada fosse mais utilizada e que as possíveis soluções envolviam conjunto de equipamento caro e volumoso. A sugestão deles foi então utilizar uma quantidade m de pontos ($m < M$) escolhidos de maneira aleatória, fazendo com que a quantidade de operações para a fase da implementação (fase dos votos) seja reduzida de $O(M \times N_\theta)$ para $O(m \times N_\theta)$. Para a fase de localização dos pares mais votados, a quantidade de operações é dada por $O(N_\rho \times N_\theta)$. Os autores sugerem utilizar o método de Gerig (GERIG, 1987) de detecção de picos, que necessita aproximadamente de $m \times N_\theta$ operações.

Vários testes foram feitos para demonstrar a efetividade da técnica, todos tomando como base imagens com resolução de 256 x 256 e os resultados apresentados tomam como base 100 repetições de cada experimento. Para medir o desempenho do algoritmo, o critério escolhido foi a quantidade de falsos negativos ou falsos positivos. Um ponto importante é que, em todos os casos, os autores sabiam a quantidade total de pontos nas imagens e a quantidade de pontos pertencentes às linhas (KIRYATI *et al*, 1990).

A Figura 12(a) apresenta uma imagem desenvolvida para teste da transformada probabilística com uma única linha e diversos pontos de ruído. As Figuras 11(b), 11(c) e 11(d) representam a mesma imagem, porém com 20%, 5% e 2% do total de pontos da Figura 12(a), respectivamente.

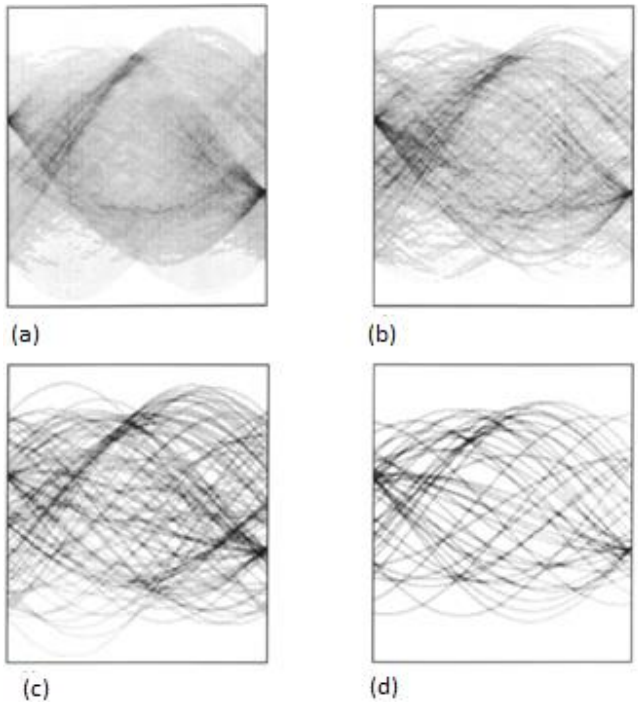
Figura 12 – Imagem para teste da transformada de Hough probabilística, onde: (a) imagem original, (b) imagem com 20% dos pontos de (a), (c) imagem com 5% dos pontos de (a), (d) imagem com 2% dos pontos de (a)



(Fonte: Kiryati *et al*, 1990)

A Figura 13(a) apresenta o plano de transformação referente a imagem 11(a) e assim sucessivamente para as imagens 12(b), 12(c) e 12(d).

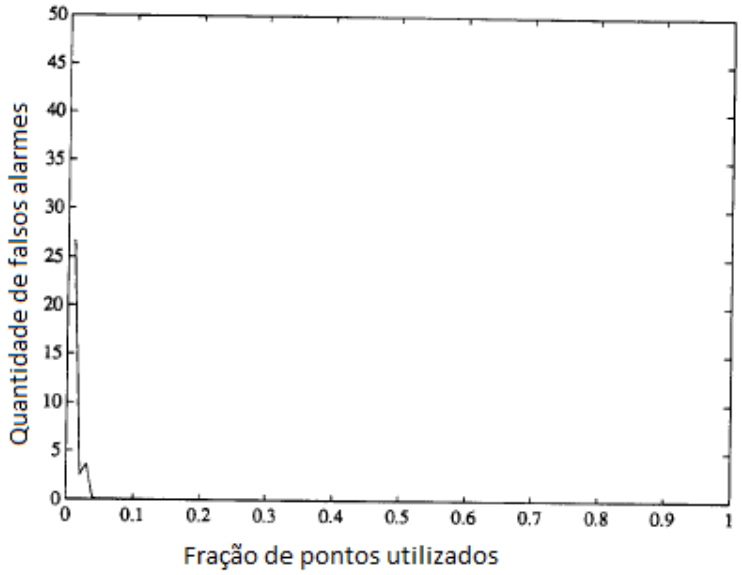
Figura 13 – Plano de transformação transformada de Hough probabilística, onde: (a) referente à Figura 12(a), (b) referente à Figura 12(b), (c) referente à Figura 12(c), (d) referente à Figura 12(d)



(Fonte: Kiryati *et al*, 1990)

A Figura 14 apresenta a quantidade média de falsos alarmes em função da quantidade de pontos utilizados como fonte para a transformada.

Figura 14 – Quantidade de falsos alarmes por fração de pontos utilizados



(Fonte: Kiryati *et al*, 1990)

Os falsos alarmes ocorrem com mais frequência quanto menor for a fração de pontos utilizados para a transformada (KIRYATI *et al*, 1990).

Os autores concluem com os experimentos que normalmente é possível reduzir a quantidade de processamento, mesmo em imagens com muito ruído e inclusive formas inseridas digitalmente para "distrair" o algoritmo, reduzindo a quantidade de pontos utilizados como entrada para a transformada.

2.4.2. TRANSFORMADA DE HOUGH PROBABILÍSTICA PROGRESSIVA

Matas *et al* (1999) desenvolveram a Transformada de Hough Probabilística Progressiva, sendo que a principal diferença entre ela e a Transformada de Hough Probabilística é que ao invés de executar a Transformada de Hough Padrão com a quantidade de pontos reduzida como na probabilística, eles exploram a fração de votos necessários para detectar linhas de maneira confiável com diferentes número de pontos de suporte.

Outro ponto destacado é que não é necessário nenhum conhecimento prévio da imagem, como na probabilística, na qual era necessário conhecer a quantidade de pontos pertencentes à linha para ajustar o limiar. Matas *et al* (1999) destacam que ao longo dos anos foram desenvolvidas algumas soluções para aplicar a Transformada de Hough Probabilística sem conhecimento prévio dos dados da imagem, porém a vantagem com relação à Transformada de Hough Padrão não é tão grande.

Os autores denominam que linhas com comprimentos longos são linhas com suporte forte, por possuir muitos pontos. Segundo eles, com uma pequena fração dos pontos pertencentes a essas linhas é possível detectá-las. Para linhas menores, é necessária uma fração muito maior dos pontos. E em alguns casos, quando a quantidade de pontos pertencentes à linha é quase igual a quantidade de pontos de ruído na mesma região, a transformada deve ser completamente executada, ou seja, em todos os pontos da linha ou região.

Essa diferenciação com relação às características das linhas é ajustada dinamicamente, controlando o limiar de aceitação para cada linha, visto que essa versão da transformada é capaz de identificar quais pixeis pertencem a linhas no momento em que eles são identificados.

No algoritmo, pontos são selecionados de maneira aleatória e seus votos são considerados. A cada ponto é realizada uma verificação em todos os acumuladores, para verificar se algum deles excedeu a quantidade de votos necessários. Caso sim, o algoritmo dá uma linha como detectada e os votos pertencentes aos pontos dessa linha são retirados. Caso não, o limiar é ajustado e a seleção aleatória de pontos continua (MATAS *et al*, 1999).

Os autores destacam que a execução do algoritmo pode ser parada a qualquer momento e ainda assim haverá resultado, pois as linhas que já foram detectadas não são perdidas.

O algoritmo funciona da seguinte maneira:

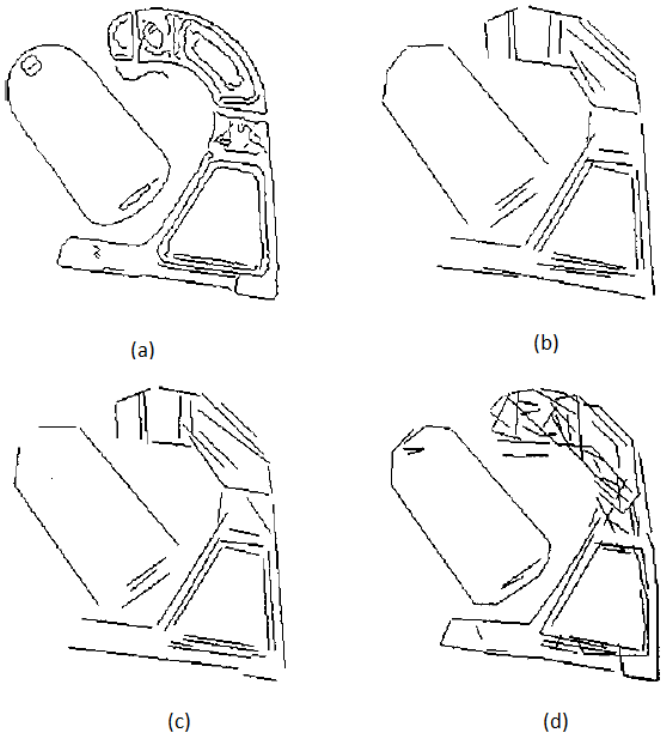
1. Verificar a imagem. Se estiver vazia, então finalize a execução;
2. Atualize os acumuladores com um pixel selecionado de maneira aleatória;
3. Retire esse pixel da imagem;
4. Verifique se os acumuladores modificados pelo ponto estão acima do limiar. Se não, voltar para 1;
5. Verifique o maior segmento contínuo ou que possui uma lacuna menor que o limite definido no corredor apontado pelo acumulador selecionado;
6. Retire os pixeis correspondentes ao segmento da imagem;
7. Retire os votos de todos os pixeis pertencentes ao segmento retirado da imagem;
8. Se o segmento for maior que o comprimento mínimo definido, adicione-o à lista de saída;
9. Voltar para 1.

Os autores provaram através de testes que existe uma relação direta entre a quantidade de votos e o tempo necessário para o processamento da imagem, chegando a um coeficiente de correlação de 99,8%. Em outro teste os autores mostraram que o algoritmo desenvolvido por eles foi superior à Transformada de Hough Padrão em quantidade de falsos negativos e falsos positivos em centenas de imagens geradas digitalmente contendo de 2 a 20 linhas com 100 pixeis de comprimento cada, posicionadas aleatoriamente em uma imagem de 256x256.

Por fim, diversos testes foram executados com imagens reais tendo os resultados comparados com a Transformada de Hough Padrão para mensurar a qualidade e com a Transformada de Hough Aleatória (MUKHOPADHYAY E CHAUDHURI, 2014) para mensurar a velocidade de processamento. Eles concluíram que a quantidade de votos necessários para a versão deles da transformada necessita em média de 25% dos votos necessários para a transformada padrão, independente da complexidade da imagem. Com relação à versão aleatória, a versão deles foi sempre superior, sendo que em imagens mais complexas a diferença é maior do que em imagens mais simples (MATAS *et al*, 1999).

A Figura 15 apresenta um dos testes executados pelos autores, sendo representadas as bordas (a), o resultado da Transformada de Hough Probabilística Progressiva (b), o resultado da Transformada de Hough Padrão (c) e o resultado da Transformada de Hough Aleatória (d).

Figura 15 – Comparação entre transformadas, onde: (a) bordas originais, (b) Transformada de Hough Probabilística Progressiva, (c) Transformada de Hough Padrão, (d) Transformada de Hough Aleatória



(Fonte: Matas *et al*, 1999)

Para o teste apresentado, a quantidade de votos necessários para a Transformada de Hough Padrão foi 5347, enquanto para a Transformada de Hough Probabilística Progressiva foi de 1439. Enquanto a Transformada de Hough Aleatória precisou de 6,2 segundos para ser executada, a Transformada de Hough Probabilística Progressiva foi executada em 1,8 segundos (MATAS *et al*, 1999).

Por fim, os autores concluem brevemente que a versão da Transformada de Hough apresentada por eles possui vantagens com relação à Transformada de Hough Padrão e que seu algoritmo é adequado para aplicações em tempo real, com um determinado tempo de processamento previsto.

2.5. *SUPPORT VECTOR MACHINE*

Cortes e Vapnik (1995) apresentaram o conceito de *Support Vector Machine* (SVM), que foi definida por Noble (2006) como um algoritmo computacional para classificação que aprende através de exemplos.

Aplicações comuns para SVM são aquelas onde é necessário realizar uma separação entre dois grupos de dados, como por exemplo, os grupos vistos na Figura 16 (a). Noble (2006) cita aplicações em análise de operações com cartões de crédito para descobrir quais eram fraudulentas e quais não, por exemplo.

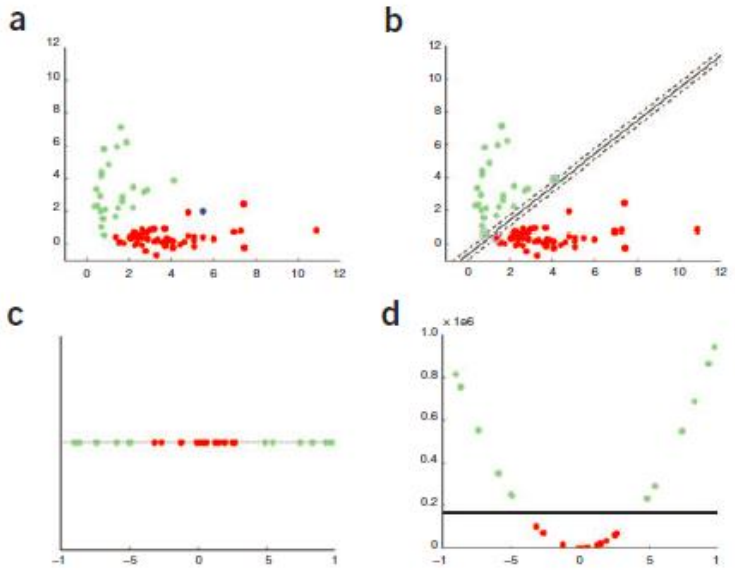
Segundo Noble (2006) existem quatro conceitos principais por trás da SVM, são eles:

- Hiperplano de separação: é o que de fato separa as duas amostras de dados diferentes de maneira linear. Imaginando amostras de apenas uma dimensão, o hiperplano de separação é um ponto. No caso de duas dimensões, uma linha. Para três dimensões, um plano. Para mais dimensões, apesar de não ser possível representar graficamente, é possível realizar a separação através do hiperplano;
- Máxima margem de separação: é uma característica que garante uma qualidade diferenciada para SVM com relação a outros classificadores. Existem diversas possibilidades de separação entre duas amostras, porém apenas uma garante a maior distância possível dos pontos mais próximos das duas classes em questão, ela proporciona a máxima margem de separação, como exemplificado na Figura 16 (b). A ideia por trás dessa característica é maximizar a chance de acertar a classificação de dados mais extremos;
- Margem suavizada: permite que uma quantidade controlada de componentes de uma amostra esteja do lado oposto ao que deveria sem interferir no resultado do

posicionamento do hiperplano de separação. Uma opção para não adicionar margens suavizadas seria aplicar uma Função de Kernel;

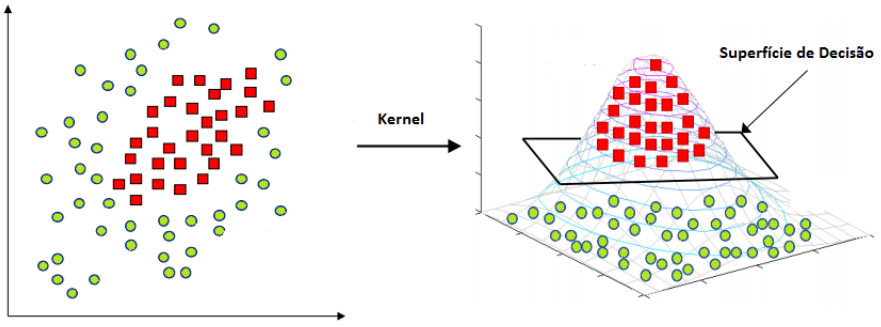
- Função de Kernel: permite mapear a amostra para um espaço com mais dimensões para que, nesse novo espaço, o hiperplano de separação possa ser traçado. Porém, esse recurso deve ser utilizado com cautela, pois adicionar muitas dimensões à amostra pode resultar em um hiperplano que não represente de fato a melhor separação, e que pode, na verdade, dificultar a classificação dos dados. Segundo Noble (2006), é possível provar que para toda e qualquer amostra de dados existe uma Função Kernel que garantirá que a separação dos dois grupos poderá ser feita de maneira linear. As Figuras 15 (c) e 15 (d) mostram um exemplo desse tipo de separação. Após aplicar uma Função Kernel na amostra e traçar o hiperplano de separação, como pode ser visto na Figura 17, é possível retornar a representação para a quantidade de dimensões originais, representando o hiperplano como separador, possuindo outro formato, exemplificado na Figura 18.

Figura 16 – Conceitos *Support Vector Machine*, onde: (a) dois grupos de dados, (b) máxima margem de separação, (c) função impossível de ser separada linearmente e (d) função de (c) após aplicação de uma Função Kernel



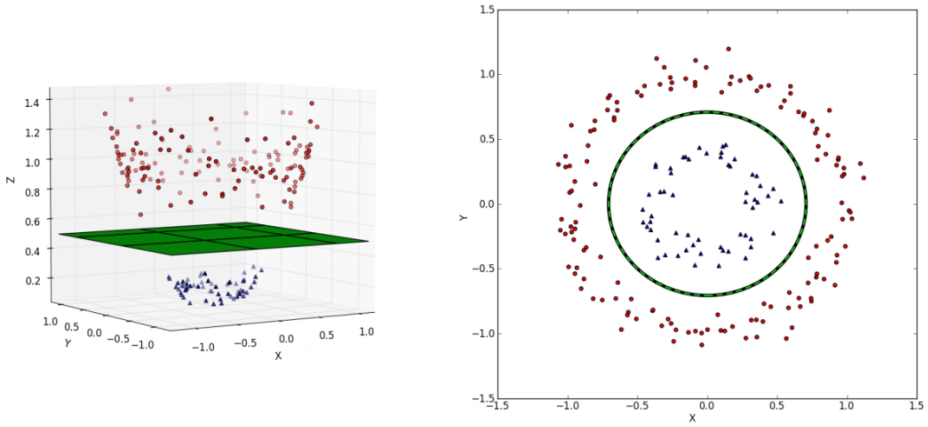
(Fonte: Noble, 2006)

Figura 17 – Aplicação da Função Kernel



(Fonte: Jain, 2017)

Figura 18 – Retorno do hiperplano separador para o plano original após aplicação de uma Função Kernel



(Fonte: Kim, 2017)

2.5.1. TIPOS DE FUNÇÃO KERNEL

Existem diferentes tipos de Função Kernel, as quais possuem diferentes parâmetros que devem ser determinados pelo usuário a fim de obter a melhor classificação possível. Segundo Min e Lee (2005) e Lorena e Carvalho (2007) as Funções Kernel mais utilizadas são:

- Linear: $u'v$;
- Polinomial: $(gamma\, u'v\ +\ r)^{grau}$;
- Sigmoidal: $\tan(h)(gamma\, u'v\ +\ r)$;
- RBF (*Radial-Basis Function*): $\exp(-gamma|u-v|^2)$.

Todas Funções Kernel possuem o parâmetros de penalidade, que é representado pelo símbolo C. A Função Kernel Linear não possui parâmetros adicionais, a Função Kernel RBF possui um parâmetro adicional, o parâmetro $gamma$. A Função Kernel Polinomial possui três parâmetros adicionais, o grau polinomial a ser adotado, $gamma$ e o coeficiente0, representado pelo símbolo r. A Função Kernel Sigmoidal possui três parâmetros adicionais, $gamma$ e r e o h, que define se será ou não utilizada heurística de encolhimento.

3. REVISÃO BIBLIOGRÁFICA

Nesse capítulo, são apresentados sistemas de verificação de paletes com suas características, funcionamento e aplicações detalhadas. Apenas quatro trabalhos foram localizados, sendo um deles um sistema de verificação por contato, apresentado na primeira parte do capítulo Na segunda parte, são apresentados os sistemas de verificação por imagem. Dos quatro trabalhos citados, três foram encontrados em forma de patente, o que dificulta a análise do sistema, uma vez que as patentes apresentam apenas uma ideia de como seria o sistema, não detalham o seu desenvolvimento e não apresentam resultados. Existem, portanto uma lacuna com relação a técnicas utilizadas em desenvolvimento de sistemas de visão para verificação de paletes, bem como detalhamento de desenvolvimento e resultados.

3.1. SISTEMAS DE VERIFICAÇÃO DE PALETES POR CONTATO

O sistema automático para inspeção de paletes desenvolvido por Gatteschi (2005) é composto por uma célula robotizada, com uma entrada e uma saída de paletes, uma posição para acúmulo de paletes bons, uma posição de acúmulo de paletes ruins e a posição de verificação. O sistema

Durante o ciclo, o robô prioriza a inspeção de paletes. Então, se a posição de saída da célula estiver ocupada, o robô deposita o palete nas posições de acúmulo, formando uma pilha de paletes. Quando uma das pilhas de paletes fica cheia, é acionado um aviso para o operador, para que ele retire os paletes da célula (GATTESCHI, 2005).

3.2. SISTEMAS DE VERIFICAÇÃO DE PALETES POR IMAGEM

Patrício e Maravall (2006) estudaram a aplicação de um sistema de visão para detectar pequenas fissuras, de aproximadamente 1mm, nos paletes de madeira. Dentre as principais dificuldades citadas estão: as veias naturais da madeira e a quantidade de irregularidades apresentadas pelos paletes após alguns meses de uso, que podem gerar falsos alarmes, além de encontrar a técnica de segmentação mais adequada para fazer a detecção das fissuras. Eles concluíram que todas as técnicas existentes geravam um número de falsos alarmes acima do aceitável, então propuseram um conceito novo, nomeado Histograma de Elementos Conectados. Esse conceito é um derivado dos histogramas tradicionais, porém com funções espaciais mais avançadas.

O sistema de visão em si, nomeado pelos autores de *Visual Inspection Machine* (VIM), é composto por um dispositivo mecânico de manipulação, sistema de iluminação, oito

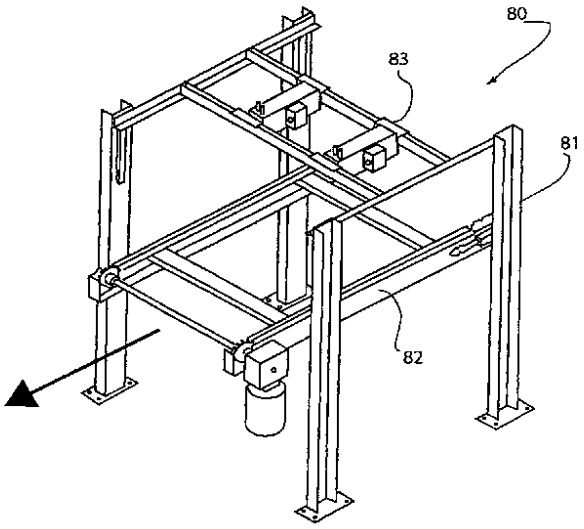
câmeras, dois cartões digitalizadores, um computador e um controlador lógico programável. O palete é verificado no sentido normal e girado 180° dentro de uma cabine preta, para evitar reflexões.

O sistema de verificação é dividido em 4 módulos: calibração, segmentação, medida e classificação. No módulo da calibração, é realizada uma correspondência entre a quantidade de pixeis da imagem com a dimensão do palete, criando um palete virtual e determinando as medidas reais do palete. A precisão do módulo de calibração é submilimétrica. No módulo de segmentação, é obtido o contorno de cada um dos elementos do palete em análise. A principal dificuldade é diferenciar os pixeis do palete dos pixeis do fundo. Para facilitar essa identificação, é utilizada uma imagem em escala de cinza. Com os dados obtidos pelo módulo de calibração e de segmentação, o módulo de medidas determina as dimensões de cada elemento do palete. O último módulo verifica os elementos à procura de fissuras (PATRÍCIO E MARAVALL, 2006).

Os autores concluíram que as fissuras têm um valor diferente dos outros elementos da madeira na escala de cinza, portanto o princípio do funcionamento foi baseado em encontrar elementos que poderiam ser fissuras e comparar com um padrão de veias da madeira, entre outras características, para concluir se o elemento achado era uma fissura ou não.

Townsend e Lucas (2010) descrevem uma célula de inspeção de paletes automática, na qual um ou dois robôs manipuladores fazem a movimentação de paletes alimentados por uma esteira, levando-os para dispositivos de inspeção ou reparo. Esses dispositivos seriam *stand-alone*, não estando associados a nenhum outro equipamento da célula e, portanto, podem ser utilizados tanto com alimentação manual quanto automática. A primeira opção de dispositivo de verificação descrita é composta por um transportador e um sistema de sensores de presença posicionados linearmente em diferentes posições, para detectar presença/ausência de ripas de madeira. Com isso, é possível mapear o palete. O dispositivo é mostrado na Figura 19.

Figura 19 – Sistema para verificação de palete por sensores



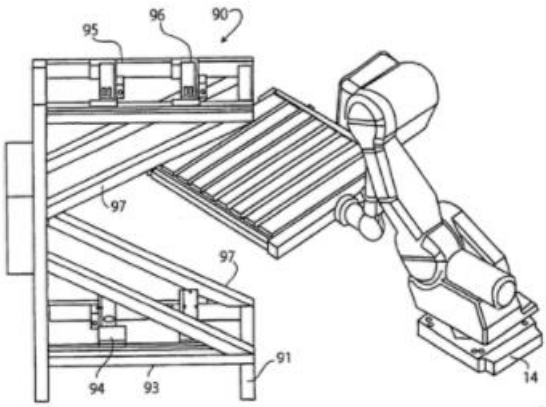
(Fonte: Townsend e Lucas, 2010)

Na Figura 19, o palete entra no dispositivo de estrutura 81 na direção 80 sobre o transportador 82. Os sensores de presença estão na posição 83.

O segundo dispositivo descrito possui a mesma estrutura física 81, porém, ao invés de sensores de presença, é usada uma câmera a laser na posição 83, ou linear com espelhos realizando movimentos ou em formato de leque, para realizar o mapeamento do palete. Com esse dispositivo, é possível adquirir imagens tridimensionais, utilizando mais uma câmera e realizando a triangulação das imagens (TOWNSEND E LUCAS, 2010).

O terceiro e último dispositivo descrito pelos autores é um sistema de verificação sem esteira, no qual o robô manipulador passa com o palete pelos sensores de presença, realizando movimentos lineares. Esse dispositivo realiza o mapeamento bidimensional do palete, apenas. A Figura 20 representa o terceiro dispositivo.

Figura 20 – Sistema para verificação de palete manipulado por robô



(Fonte: Townsend e Lucas, 2010)

Na Figura 20, o palete entra no dispositivo de estrutura 97 na direção 90 sendo manipulado pelo robô 14. Os sensores de presença estão nas posições 93, 94, 95 e 96.

Townsend e Lucas (2015) apresentam um sistema para inspeção e reparo de paletes composto por um emissor de laser e uma câmera. A câmera captura o que é refletido do laser, formando uma imagem em duas dimensões. Essa imagem é processada por um computador, que identifica os elementos dos paletes e os localiza através de coordenadas. Cada elemento é analisado pelos seguintes critérios:

- Individuais (presença/ausência, tamanho, localização e integridade);
- Entre elementos (espaçamento e sobreposição);
- Tipo de palete (presença de itens desnecessários ou ausência de itens necessários).

Segundo os autores, para o processamento da imagem, são utilizados dois programas. O primeiro realiza a aquisição da imagem e o segundo o processamento propriamente dito. Para analisar a imagem, é aplicado um filtro para remoção do fundo, resultando em uma imagem apenas da superfície do palete. Para isso, são identificados os quatro pontos de extremidade do palete, nomeados como PP0, PP1, PP2 e PP3. Conhecendo esses pontos, o programa calcula as dimensões (x,y) do palete.

Na sequência, são aplicados os critérios de inspeção do palete e, caso o sistema identifique defeitos no palete, ele gera uma lista dos itens que devem ser reparados ou removidos. Essa lista é enviada para a estação de reparo, onde um robô manipulador executa a receita gerada através da lista, ou reparando o palete para uso novamente, ou desmontando o palete e separando as partes que podem ser usadas em um reparo futuro de outro palete (TOWNSEND E LUCAS, 2015).

4. MATERIAIS E MÉTODOS

Para a execução do projeto proposto, os seguintes materiais e programas foram utilizados:

- Notebook;
- Paletes;
- Iphone SE;
- Software IDLE com linguagem Python;
- Biblioteca OpenCV;
- Biblioteca LIBSVM.

O sistema se baseia em um programa computacional desenvolvido via biblioteca OpenCV, o qual realiza a análise de imagens de paletes através do detector de bordas de Canny e da Transformada de Hough, com o objetivo de detectar linhas nas imagens. Ao identificar as linhas, é gerado um histograma com os dados do palete, no qual as classes são referentes aos comprimentos e ângulos das linhas detectadas. O sistema trabalha com aprendizagem de máquina, ou seja, primeiro é treinado com a apresentação de quais paletes estão em boa condição para uso em uma célula de paletização e quais não, gerando uma base de dados. Em situação de operação, o sistema identifica paletes aptos ou não para aplicação na célula de paletização, através da classificação pelo sistema de aprendizagem de máquina.

Ao longo do capítulo 2, foram detalhadas as características das técnicas de processamento das imagens, destacando então as que apresentam melhores resultados e são consideradas padrões para determinadas aplicações. Ao longo do capítulo 5, são apresentados os testes realizados para validar as características descritas na teoria. Foram testadas diversas situações para demonstrar o funcionamento de cada uma das técnicas, o efeito ao selecionar diferentes valores para seus parâmetros, validar o comportamento das técnicas com base na descrição teórica de cada uma delas e, por fim, detalhar o modo de seleção dos parâmetros utilizados no sistema de visão.

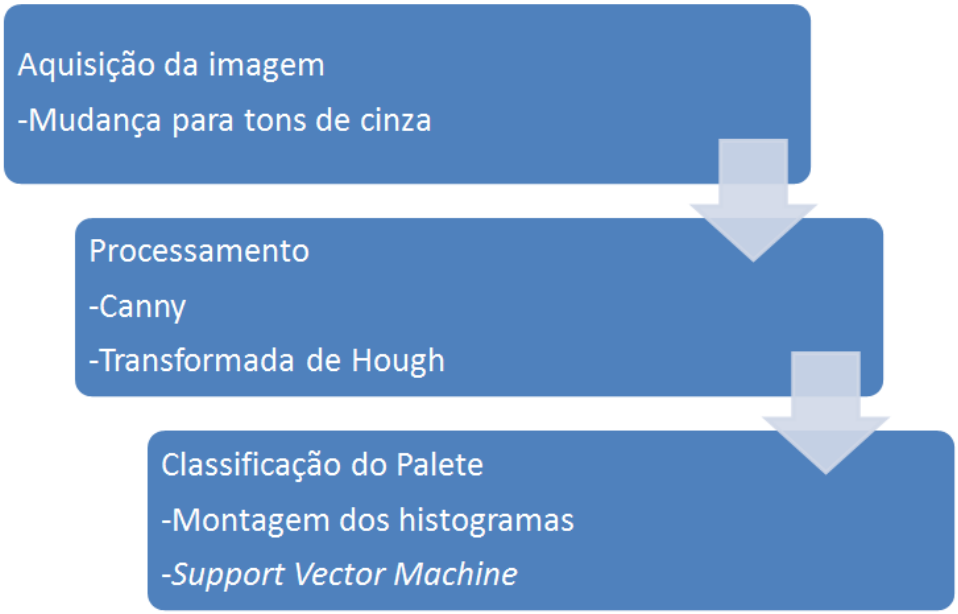
O detector de bordas de Canny foi selecionado por ter sua superioridade de desempenho, conforme seção 2.3, além de ser considerado a solução padrão para aplicações de caracterização de linhas na indústria, conforme seção 2.3.1. A Transformada de Hough é a técnica mais popular para caracterização de linhas, conforme seção 2.4. A Transformada de Hough Probabilística Progressiva foi selecionada pelo seu desempenho e eficiência descritos na seção 2.4.2.

Dentre diversas possibilidades de classificadores de dados, a SVM foi selecionada por possuir as características necessárias para o sistema em questão, ou seja, uma amostra com duas classes, e por possibilitar a otimização na separação com sua máxima margem de separação, o que maximiza a chance de acerto na classificação de um dado que esteja próximo à região divisória entre as duas classes. No caso, a opção foi feita pela utilização da biblioteca original da LIBSVM (CHANG E LIN, 2011) ao invés de utilizar a presente na OpenCV, que é baseada na LIBSVM (OPENCV, 2018).

O que motivou essa escolha foi o fato que a biblioteca original possibilita a utilização de ferramentas não disponíveis na versão presente na OpenCV, entre elas, a função *grid* que auxilia na definição dos parâmetros ótimos para a Função Kernel RBF.

A Figura 21 apresenta os passos do sistema de visão.

Figura 21 – Estrutura do sistema de Visão



(Fonte: Autor)

4.1. OPENCV

OpenCV é uma biblioteca fornecida livremente, voltada para visão computacional. Foi desenvolvida nas linguagens de programação C e C++, mas também já disponível em Python e Java. Ela funciona nos sistemas operacionais Linux, Windows e Mac OSX (BRADSKI e KHAELER, 2008).

A biblioteca contempla mais de 2500 funções e tem como objetivo disponibilizar uma estrutura simples, com a qual possam ser criados sistemas de visão avançados e sofisticados (OPENCV, 2018).

Lançada em 1999, a biblioteca permite que qualquer pessoa a utilize para fins educativos ou comerciais, sem a obrigação de disponibilizar o código desenvolvido ou retornar melhorias para a biblioteca (BRADSKI e KHAELER, 2008). O projeto da biblioteca foi iniciado pela Intel, mas, oficialmente, não existe um dono da OpenCV.

A OpenCV é dividida em cinco grupos:

- CV: básico de processamento de imagens e algoritmos de visão computacional avançados;
- MLL: Classificadores estatísticos e ferramentas de agrupamento;
- HighGUI: Rotinas de entrada e saída e funções de armazenar e carregar vídeos e imagens;
- CXCore: Dados estruturais;
- CvAux: algoritmos experimentais.

4.2. DADOS

Os dados utilizados para desenvolvimento e teste do sistema de visão para verificação de paletes consistem em um conjunto de 120 imagens de paletes, sendo 60 dessas imagens com paletes em condições para uso em uma célula de paletização robotizada e as outras 60 com paletes apresentando defeitos que ocorrem com alta frequência, segundo experiência do autor, e que prejudicariam ou impossibilitariam a utilização deles em uma célula de paletização robotizada.

As imagens foram obtidas com a câmera de um celular Iphone SE com objetivo de utilizar uma câmera comum. Essa escolha vai de encontro com o desenvolvimento de um sistema acessível no qual foram utilizados programas computacionais e bibliotecas gratuitas, possibilitando um alto valor agregado com custo reduzido.

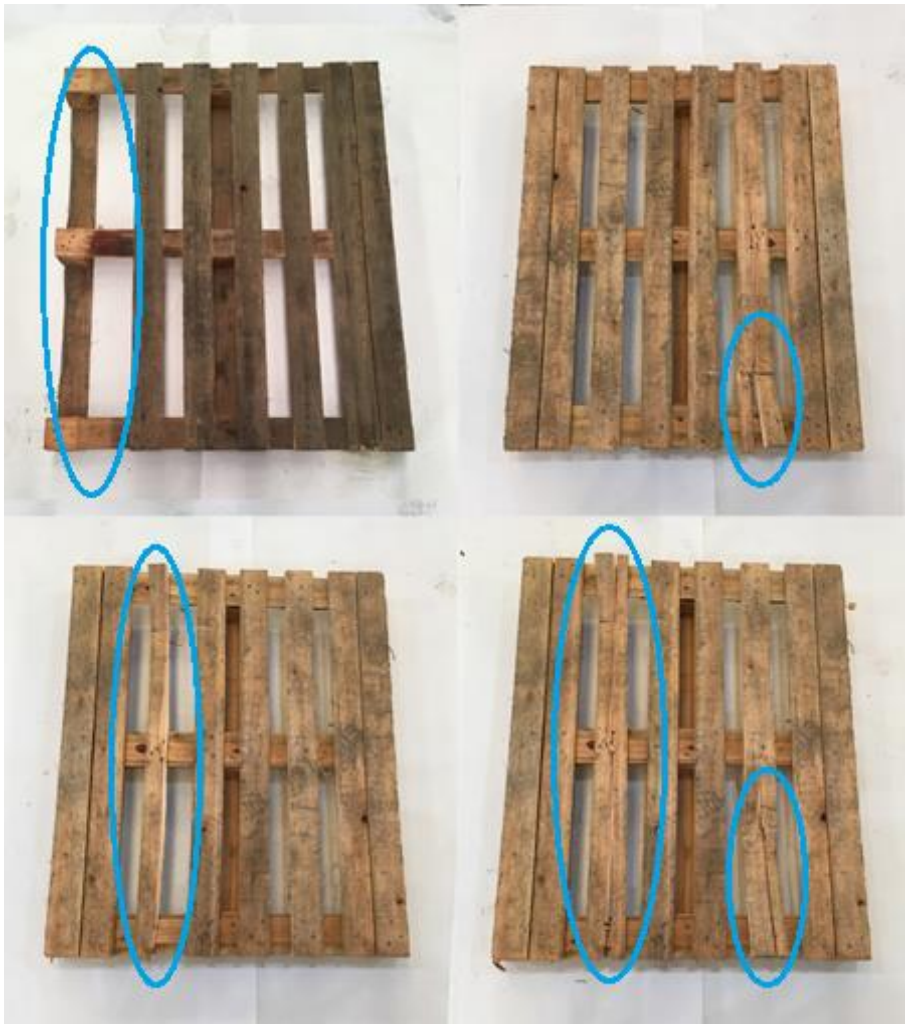
O ambiente de captura das imagens possui fundo branco e não foi utilizado nenhum tipo de iluminação especial ou forçada, apenas a iluminação ambiente da fábrica onde se localizavam os paletes. Foram realizadas marcações no fundo branco para posicionamento padronizado do palete, porém não foram colocadas barreiras físicas para limitar a posição.

Após captura, as imagens que possuíam dimensões 3024 x 4032 pixeis foram reduzidas para 2918 x 3284 pixeis para melhor enquadramento dos paletes, porém sem realizar nenhum tipo de ampliação ou redução, que poderiam levar a distorções nos objetos alvo. Também não foi utilizada nenhuma técnica de processamento de imagem que realizasse

alguma alteração na estrutura das imagens, como remoção de fundo, erosão ou dilatação, citadas na seção 2.2.

Considerando que o foco do sistema de visão proposto está em localizar defeitos nas ripas dos paletes, são esses elementos que aparecem defeituosos ou são ausentes nas imagens de paletes sem condições para a citada aplicação, como pode ser visto na Figura 22.

Figura 22 – Exemplos de defeitos nos paletes



(Fonte: Autor)

Das 120 imagens do conjunto, 90 serão utilizadas para treinamento do sistema, sendo 45 delas com paletes em condições e 45 com paletes sem condições para uso em uma célula de paletização robotizada. As outras 30 imagens serão utilizadas apenas para teste e validação do sistema.

5. PROCEDIMENTOS EXPERIMENTAIS

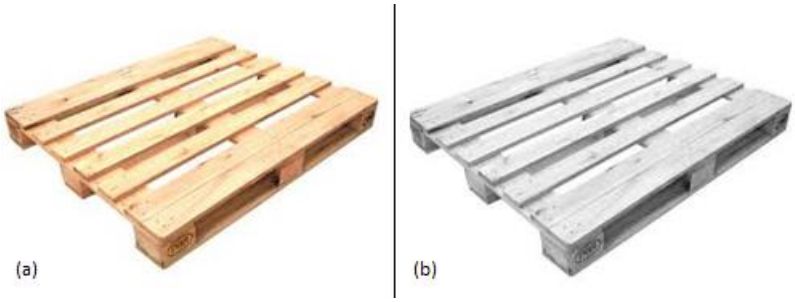
Para o desenvolvimento do programa computacional do sistema de visão, foi utilizada a linguagem de programação Python e fazendo uso da biblioteca OpenCV.

A primeira etapa no processamento é a aquisição da imagem, que pode ser realizada em tempo real, porém, para o desenvolvimento do sistema, foram utilizadas imagens existentes carregadas no computador.

Como explicado na seção 2.3, o ideal para detectores de borda é utilizar imagens em tons de cinza para os operadores que não são específicos para aplicações com imagens coloridas.

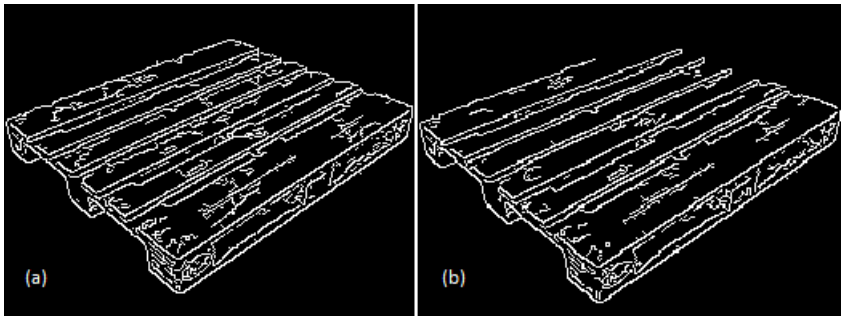
Foram efetuados testes com o objetivo de mostrar a diferença no resultado da aplicação do detector de bordas de Canny com a mesma imagem colorida e em tons de cinza. Para o teste, foi utilizada uma imagem de um palete. O limiar inferior e superior foram mantidos em 50 e 200 respectivamente, para os dois casos. A imagem importada colorida pode ser vista na Figura 23(a) e a imagem importada em tons de cinza pode ser vista na Figura 23(b). Os resultados da aplicação do detector de bordas de Canny nas imagens importadas colorida e cinza podem ser vistos na Figura 24(a) e na Figura 24(b), respectivamente.

Figura 23 – Palete para teste importado, onde: (a) colorido e (b) tons de cinza



(Fonte: Autor)

Figura 24 – Aplicação do detector de bordas de Canny imagem da Figura 23, onde: (a) colorida e (b) tons de cinza



(Fonte: Autor)

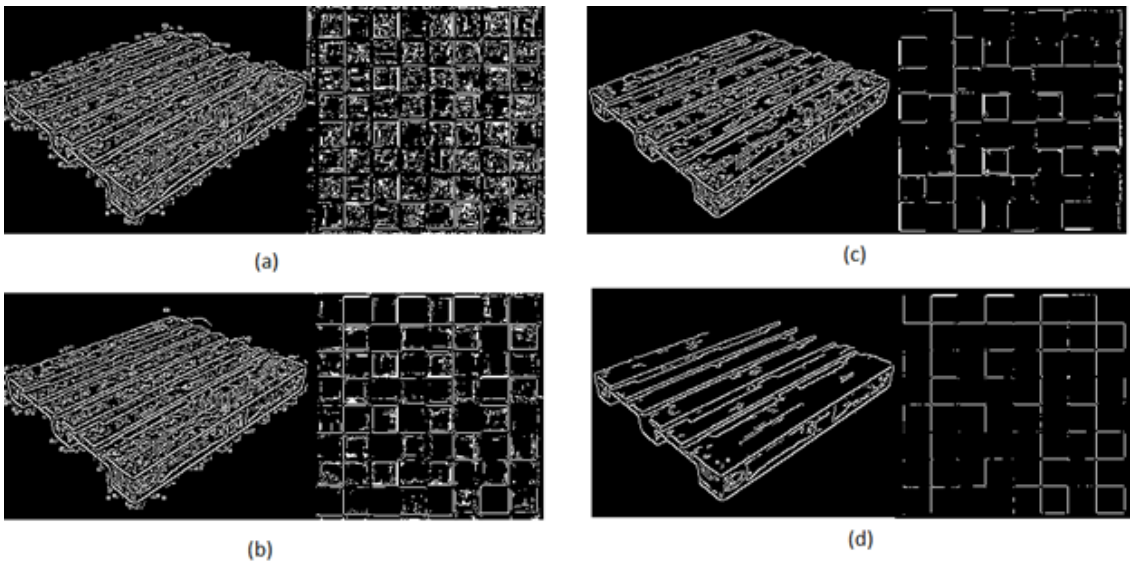
A aplicação do detector de bordas de Canny na imagem colorida apresenta uma quantidade maior de ruído no resultado com relação à aplicação na imagem em tons de cinza, que em contrapartida não detectou todas as bordas existentes do palete para os limiares definidos.

Como concluído na seção 2.3, a definição dos valores limiares deve ser feito manualmente, de acordo com o nível de ruído presente na imagem analisada. Para demonstrar esse ajuste, foram realizados testes comparando os resultados na aplicação do detector de bordas de Canny com diferentes valores dos limiares com a Figura 23(b) e com a Figura 25, que apresenta um padrão de Tsai, que é uma imagem quadriculada na qual as cores branca e preta são alternadas, formando um tabuleiro de damas ou xadrez. Essa imagem é conhecida como padrão de Tsai por ser utilizada por Tsai no desenvolvimento de um dos seus métodos de calibração de câmeras. A Figura 26 apresenta os resultados, sendo que os valores adotados para o limiar mínimo e máximo foram, respectivamente, 1 e 5 (a), 5 e 25 (b), 25 e 75(c) e 75 e 200(d).

Figura 25 – Padrão de Tsai

(Fonte: Autor)

Figura 26 – Teste de limiares comparativo em imagens com nível de ruído diferentes, onde os limiares são: (a) 1 e 5, (b) 5 e 25, (c) 25 e 75 e (d) 75 e 200



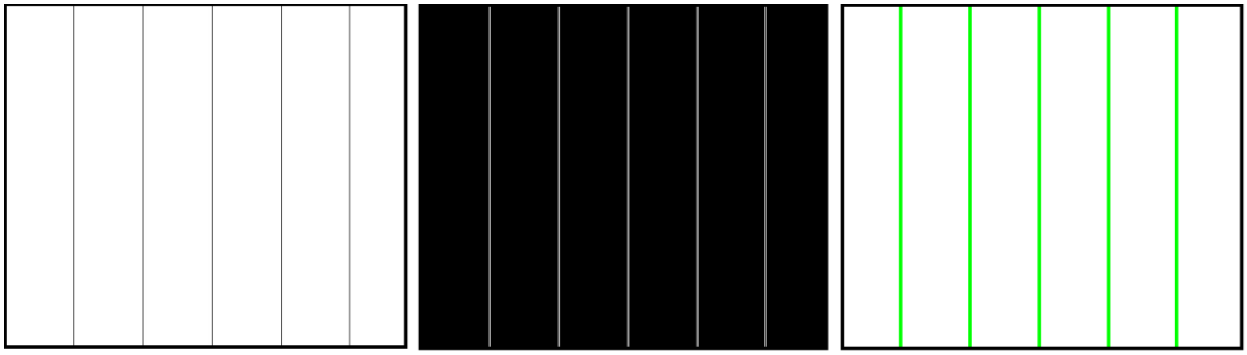
(Fonte: Autor)

É possível notar que para limiares muito baixos, como na Figura 26(a) o nível de ruído detectado como borda é muito alto em ambas as imagens. Já na Figura 26(b), o nível de ruído no padrão de Tsai já não é tão alto quanto no palete. Na Figura 26(c) o nível de ruído no palete ainda está alto, enquanto o padrão de Tsai já começou a perder informações relevantes. E, na Figura 26(d), ambas as imagens já perderam informações de bordas relevantes.

Como, no caso do sistema de visão, todas as imagens serão parecidas, pois as imagens serão capturadas sempre em condições similares de posição e iluminação, será possível definir valores coerentes para os limiares de acordo com o nível de ruído presente.

Após aplicação do detector de bordas de Canny, o próximo passo é aplicar a técnica de caracterização de linhas, no caso a Transformada de Hough Probabilística Progressiva. Para validar o algoritmo antes de aplicá-lo a imagens mais complexas, foi realizado um teste com uma imagem apenas com linhas geradas digitalmente. A Figura 27 apresenta a imagem original (a), a aplicação do detector de bordas de Canny (b) e a aplicação de Hough (c).

Figura 27 – Validação da Transformada de Hough Probabilística Progressiva, onde: (a) imagem gerada digitalmente, (b) aplicação do detector de bordas de Canny, (c) aplicação da Transformada de Hough Probabilística Progressiva



(Fonte: Autor)

A aplicação resultou na detecção perfeita das linhas, sendo interessante notar que o detector de bordas de Canny detectou duas bordas em cada uma das linhas, bem como a Transformada de Hough Probabilística Progressiva. Sendo assim, foram detectadas 10 linhas.

Assim como o algoritmo do detector de bordas de Canny, esse algoritmo também possui parâmetros que podem gerar resultados mais ou menos precisos, levando em consideração as linhas detectadas em comparação com as linhas existentes, de acordo com seus ajustes.

No caso da Transformada de Hough Probabilística Progressiva, o resultado dado consiste em uma matriz de 4 colunas pela quantidade de linhas detectadas. As colunas mostram as coordenadas X e Y do ponto inicial da linha e as coordenadas X e Y do ponto

final da linha. No caso da aplicação dada na Figura 27, a matriz resultado possui 4 colunas e 10 linhas e foi representada na Tabela 2.

Tabela 2 – Resultado da Transformada de Hough Progressiva Probabilística aplicada à Figura 27(a), onde: X1 = coordenada X do ponto inicial da linha; X2 = coordenada X do ponto final da linha; Y1 = coordenada Y do ponto inicial da linha; Y2 = coordenada Y do ponto final da linha.

X1	Y1	X2	Y2
300	498	300	0
402	498	402	0
202	498	202	0
400	498	400	0
498	498	498	0
102	498	102	0
302	498	302	0
200	498	200	0
500	498	500	0
100	498	100	0

(Fonte: Autor)

Com esse resultado, a partir da coordenada dos dois pontos extremos das linhas detectadas, é possível calcular os comprimentos das linhas e o ângulo de inclinação de cada uma delas. Para isso o primeiro passo é determinar o comprimento de cada coordenada subtraindo o valor do ponto final do valor do ponto inicial. Com isso têm-se:

$$cx = X2 - X1$$

$$cy = Y2 - Y1$$

Onde:

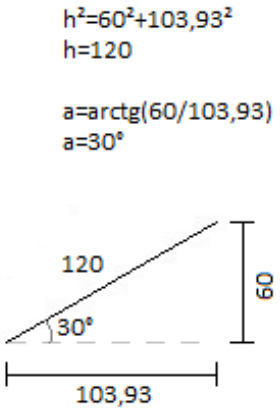
cx = componente X do comprimento

cy = componente Y do comprimento

Considerando que cx e cy formam entre si um ângulo reto, a linha detectada, cujas coordenadas dos pontos inicial e final geraram cx e cy, é a hipotenusa do triângulo retângulo formado entre cx, cy e linha em questão. Conhecendo o comprimento de dois lados de um triângulo retângulo é possível então calcular o lado desconhecido através do Teorema de Pitágoras e o ângulo de inclinação da linha através de trigonometria, no caso, para utilizar

apenas dados obtidos e não calculados, foi utilizada arco tangente para cálculo do ângulo. A Figura 28 apresenta uma ilustração do caso.

Figura 28 – Exemplo de cálculo do comprimento e inclinação das linhas



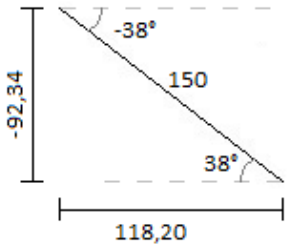
(Fonte: Autor)

O domínio da função arco tangente é de -90° a 90° e essa será, portanto, a faixa à qual os ângulos calculados pertencerão. Os casos específicos no cálculo dos ângulos são:

1. Linha com inclinação de 90°;
2. Linhas com inclinação entre 90° e 180°.

Para o primeiro caso foi adicionada uma exceção no cálculo para que, em caso de cx = 0, considerar o ângulo como reto. Para o segundo, cy será naturalmente negativo e com isso o cálculo da arco tangente resultará na inclinação entre -90° e 0°. A Figura 29 apresenta uma ilustração do segundo caso.

Figura 29 – Exemplo de cálculo do comprimento e inclinação das linhas para um caso específico

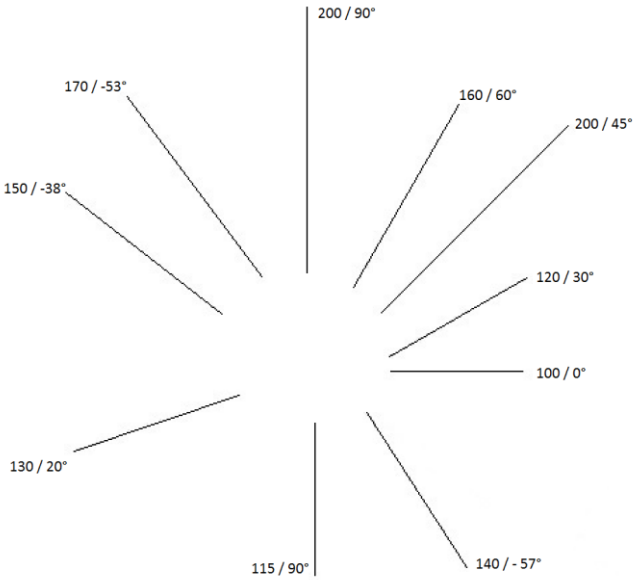


(Fonte: Autor)

Considerando o conceito do triângulo retângulo, não faz sentido ter um ângulo negativo. Para um triângulo, sendo cy um lado seu valor seria positivo e o ângulo encontrado, consequentemente, também, como indicado na Figura 29. Porém, considerando a inclinação de uma linha, sua orientação faz diferença e por isso que é necessário ter um referencial e é com relação a esse referencial que as linhas que possuem inclinação entre 90° e 180° serão consideradas como pertencentes ao quarto quadrante, e não ao segundo quadrante como normalmente, com inclinação variando entre -90° e 0°.

Para validação do cálculo dos ângulos e comprimentos, foi realizado um teste com uma imagem gerada digitalmente com linhas de diferentes comprimentos e ângulos, que pode ser vista na Figura 30. A Tabela 3 apresenta os valores resultantes da detecção.

Figura 30 – Teste de cálculo de ângulos



(Fonte: Autor)

Tabela 3 – Resultado do teste de cálculo de ângulos realizado com base na Figura 30

cx (pixeis)	cy (pixeis)	Inclinação Calculada (°)	Inclinação Esperada (°)	Comprimento (pixeis)
138	138	45	45	195,16
138	138	45	45	195,16
0	199	90	90	199
0	199	90	90	199
99	-132	-53,13	-53	165
100	-137	-53,87	-53	169,61
77	134	60,11	60	154,54
80	138	59,89	60	159,51
116	-91	-38,11	-38	147,43
116	-91	-38,11	-38	147,43
0	115	90	90	115
0	115	90	90	115
104	60	29,98	30	120,06
101	58	29,86	30	116,46
47	-115	-57,23	-57	136,75
75	-116	-57,11	-57	138,13
123	44	19,68	20	130,63
101	0	0	0	101

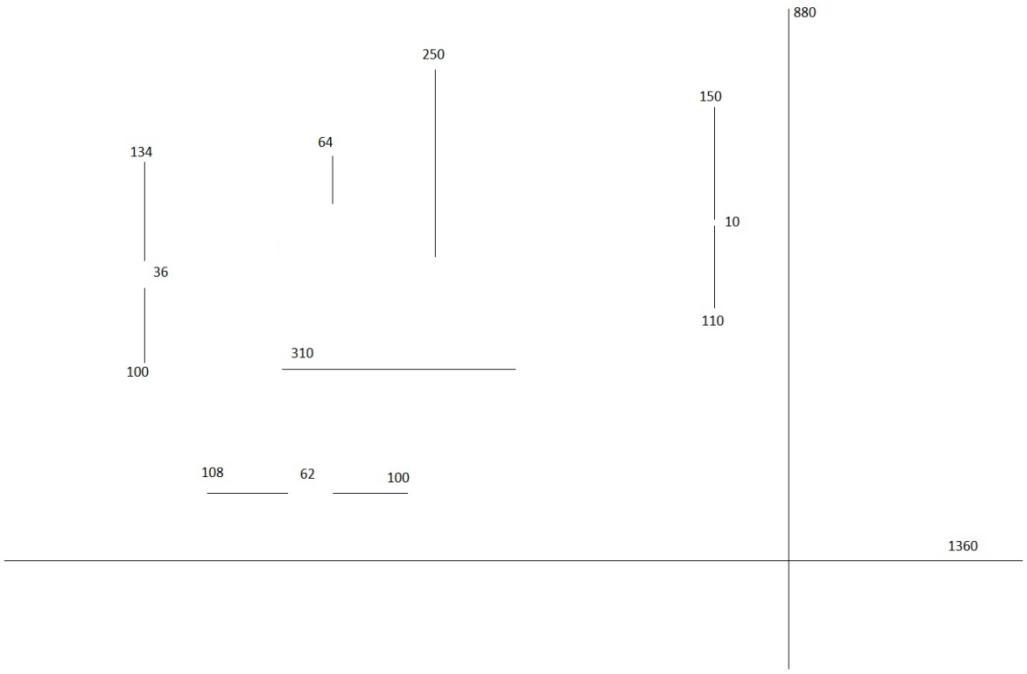
(Fonte: Autor)

As diferenças encontradas se dão por conta da resolução mínima de um pixel tanto no desenho quanto na leitura dos dados. Como a leitura dos catetos é arredondada por conta dessa resolução mínima, o cálculo tanto dos ângulos quanto dos comprimentos das hipotenusas, ou seja, das linhas apresentam diferença no valor.

O algoritmo da Transformada de Hough Probabilística Progressiva possui os seguintes parâmetros: o limiar (L); o menor comprimento a ser considerado como linha (MLL); a maior lacuna entre dois segmentos para que sejam considerados uma única linha (MLG); resolução da varredura de ângulos(θ); resolução da varredura de comprimentos(ρ). A variação desses parâmetros pode levar a detecção de falsos positivos ou a falsos negativos. Em todos os testes realizados, os valores para varredura foram mantidos mínimos, ou seja, 1 pixel para ρ e 1° para θ, pois ao aumentar esses valores, passa-se a admitir um erro maior. Conforme seção 2.4, cada ponto localizado na imagem faz com que sejam incrementados todos os possíveis pares (θ,ρ) que fazem relação com aquele ponto. Ao aumentar os valores das resoluções de varredura, a quantidade de acumuladores é reduzida e, consequentemente, a velocidade de processamento aumenta, pois a fase de incremento dos acumuladores é menor. Porém a precisão do resultado também será menor, dado que ao variar o ângulo a cada 2°, por exemplo, impede a localização precisa de linhas que seriam detectadas com θ ímpar.

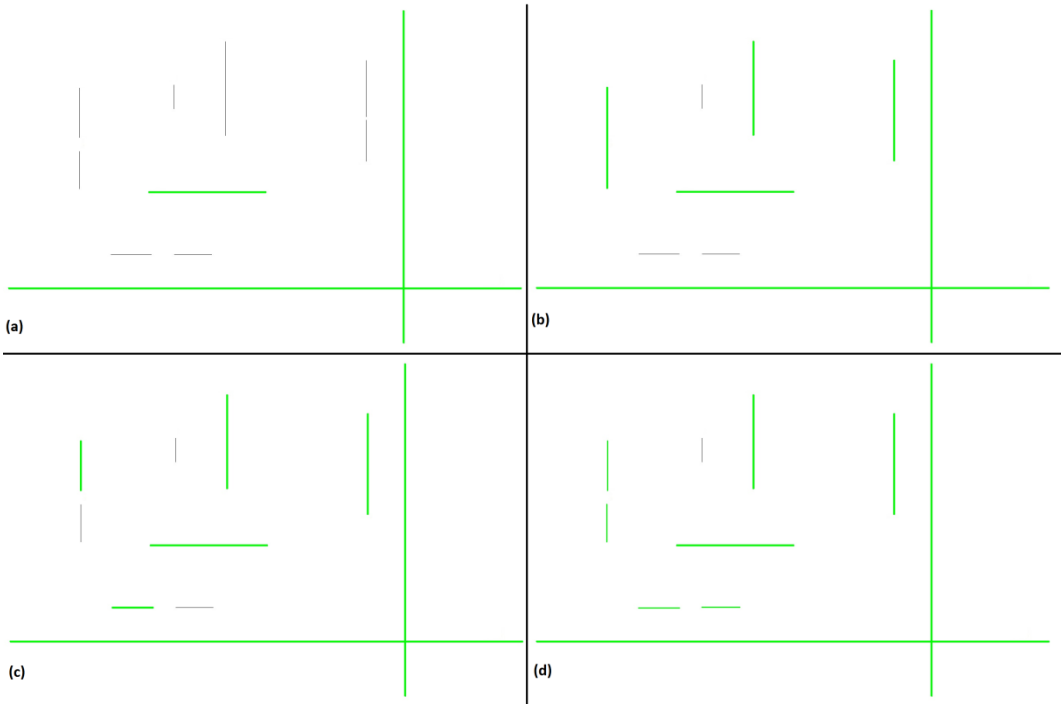
Ainda com objetivo de validar o algoritmo da Transformada de Hough Probabilística Progressiva, foi realizado um teste com uma imagem com linhas e lacunas de diferentes comprimentos geradas digitalmente. A Figura 31 apresenta a imagem com as linhas e seus respectivos comprimentos, bem como os comprimentos das lacunas. Os resultados dos testes podem ser vistos na Figura 32, onde: (a) L=255, MLL=300 e MLG=100, (b) L=220, MLL=240 e MLG=100, (c) L=80, MLL=100 e MLG=30 e (d) L=70, MLL=100 e MLG=30.

Figura 31 – Base para teste da Transformada de Hough Probabilística Progressiva com diferentes comprimentos e lacunas



(Fonte: Autor)

Figura 32 – Teste da Transformada de Hough Probabilística Progressiva com diferentes comprimentos e lacunas, onde: (a) L=255, MLL=300 e MLG=100, (b) L=220, MLL=240 e MLG=100, (c) L=80, MLL=100 e MLG=30 e (d) L=70, MLL=100 e MLG=30



(Fonte: Autor)

Como pode ser observado, os resultados são condizentes com o ajuste dos parâmetros e com a teoria. Na Figura 32(a), são detectadas seis linhas (duas bordas de cada uma das

linhas), as únicas com comprimento acima de 300 pixeis, conforme o MLL. Na Figura 32(b), além das seis linhas anteriores, mais seis linhas são detectadas, uma com comprimento de 250 pixeis e as outras duas com comprimento de 270 pixeis, somando os dois trechos mais a lacuna de cada uma, pois o comprimento da lacuna é menor que o MLG. A linha com segmentos de 108 e 100 pixeis com lacuna de 62 pixeis também tem o comprimento maior que o MLL e lacuna menor que o MLG, mas ela não foi detectada porque a quantidade de pixeis total (108+100 = 208) é menor que o limiar, que no caso é 220. Na Figura 32(c) ao reduzir MLG para 30, a lacuna de 36 pixeis não é mais considerada como aceitável para compor uma linha, porém as duas linhas de 100 pixeis deveriam ser detectadas, pois o limiar com valor 80 e MLL=100 são ajustes nos quais elas se enquadram. O que pôde ser observado é que é mais difícil detectar linhas menores e, quando detectadas, o algoritmo não localiza duas linhas (as bordas), mas sim apenas uma. Na Figura 32(d), o limiar foi alterado para 70, valor que permitiu a localização das duas linhas de 100 pixeis.

No teste da Figura 31, foram calculados os comprimentos e ângulos para as linhas detectadas em cada uma das situações da Figura 32. O resultado é apresentado na Tabela 4.

Tabela 4 – Comprimentos e ângulos das linhas detectadas na Figura 31 de acordo com ajustes mostrados na Figura 32

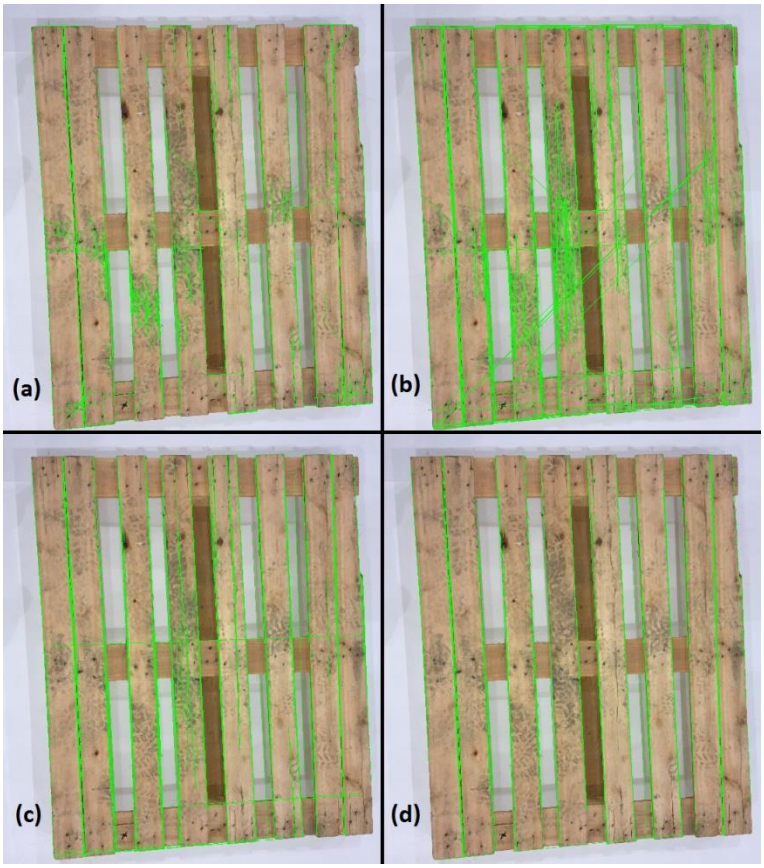
	(a)		(b)		(c)		(d)	
	Comprimento (pixeis)	Ângulo (°)	Comprimento (pixeis)	Ângulo (°)	Comprimento (pixeis)	Ângulo (°)	Comprimento (pixeis)	Ângulo (°)
1	1360	0	1360	0	1360	0	1360	0
2	1359	0	1359	0	1359	0	1359	0
3	880	90	880	90	880	90	880	90
4	881	90	881	90	881	90	881	90
5	311	0	311	0	311	0	267	90
6	311	0	311	0	267	90	249	90
7			267	90	311	0	311	0
8			267	90	249	90	101	0
9			249	90	132	90	132	90
10			249	90	267	90	311	0
11			268	90	109	0	109	0
12			268	90	132	90	267	90
13					249	90	100	90
14					109	0	249	90

(Fonte: Autor)

Conforme pode ser observado na tabela, o comprimento de algumas linhas apresentam um desvio de ± 2 pixeis para linhas que são contínuas e ± 3 pixeis para linhas detectadas com uma lacuna dentro do ajuste aceitável.

O ajuste dos parâmetros para detectar ou não uma linha em uma imagem simples, sem presença de ruído e com comprimentos e lacunas bem definidos e conhecidos, é possível de ser realizado sem maiores problemas. Porém, para uma imagem real, esses ajustes tendem a ser mais complicados. Para demonstrar os ajustes dos parâmetros da Transformada de Hough Probabilística Progressiva, foi realizado um teste com uma foto de um palete capturada pelo autor aplicando diferentes valores para o limiar, MLL e MLG. O resultado é apresentado na Figura 33, onde: (a) L=60, MLL=200 e MLG=50, (b) L=110, MLL=415 e MLG=140, (c) L=160, MLL=665 e MLG=170 e (d) L=255, MLL=300 e MLG=155.

Figura 33 – Diferentes ajustes nos parâmetros da Transformada de Hough Probabilística Progressiva, onde: (a) L=60, MLL=200 e MLG=50, (b) L=110, MLL=415 e MLG=140, (c) L=160, MLL=665 e MLG=170 e (d) L=255, MLL=300 e MLG=155



(Fonte: Autor)

Em uma análise inicial, é possível perceber a relação entre um valor baixo no limiar e a alta presença de falsos positivos. Também é possível que um valor alto de MLG permita a presença de falsos positivos longos como na Figura 33(b). Com um conjunto de valores de

limiar e MLL baixos, surgem muitos falsos positivos curtos em determinadas regiões, como pode ser visto na Figura 33(a). A Figura 33(d) é a que mais se aproxima de representar as ripas dos paletes sem a presença de falsos positivos como nas demais figuras.

Como demonstrado nas Figuras 26 e 29, o ajuste manual dos parâmetros do detector de bordas de Canny e da Transformada de Hough Probabilística Progressiva deve ser feito de maneira cuidadosa para evitar a presença de falsos positivos e falsos negativos em ambas as fases do processamento das imagens. Para otimizar a definição dos valores dos parâmetros, foi gerado um algoritmo cíclico para gerar imagens com variações deles. A cada ciclo, os valores são incrementados e, após a execução do algoritmo, as imagens foram analisadas e, dentro da faixa de valores nos quais os melhores resultados foram obtidos, o algoritmo foi executado novamente, porém dessa vez com um menor incremento entre um ciclo e outro.

Para o detector de bordas de Canny, inicialmente, foram realizados testes para verificar um limite máximo útil dos parâmetros. Foi verificado que, para as imagens dos paletes coletadas pelo autor, não seria válido ter limiares superiores a 270, pois com esse valor a imagem já estaria totalmente comprometida. A primeira execução foi iniciada com os limiares inferior e superior variando de 0 a 240 e 10 a 270, respectivamente. Na segunda execução, os valores dos limiares inferior e superior foram de 60 a 140 e de 70 a 150, respectivamente. Após análise das imagens resultantes, foram adotados os seguintes valores: limiar inferior = 90 e limiar superior = 100.

Para a Transformada de Hough Probabilística Progressiva, o mesmo processo foi realizado, porém o algoritmo foi executado uma terceira vez. Os valores de varredura de pixeis e ângulos foram mantidos mínimos. Inicialmente, limiar, menor comprimento da linha e máxima lacuna permitida variaram de 60, 100 e 50 até 460, 965 e 170, respectivamente. Na segunda execução, os valores foram de 200, 300 e 100 até 300, 800 e 160, respectivamente. Por fim, na terceira execução, os valores iniciaram em 255, 300 e 155 e foram até 275, 600 e 165. Após análise das imagens resultantes, foram adotados os seguinte valores: limiar = 255, menor comprimento da linha = 300 e máxima lacuna permitida = 165, sendo que o resultado com valor de máxima lacuna permitida igual a 155 também pode ser considerado para utilização na fase subsequente.

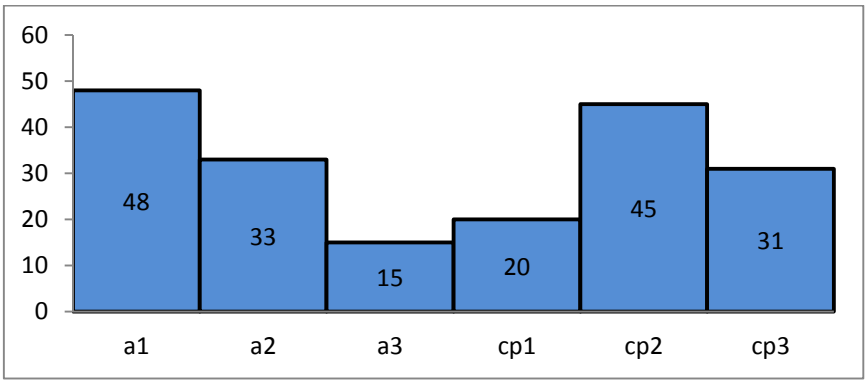
Com os valores dos parâmetros da etapa de processamento das imagens definidos, o passo seguinte foi organizar os dados para a etapa de classificação. Essa organização foi feita em formato de um histograma que apresenta classes de ângulos e comprimentos.

Inicialmente tanto comprimentos quanto ângulos, que variam de -90° a 90°, foram divididos em três classes, sendo:

- a1 = quantidade de linhas presentes com ângulo entre -90° e -30°;
- a2 = quantidade de linhas presentes com ângulo entre -30° e 30°;
- a3 = quantidade de linhas presentes com ângulo entre 30° e 90°;
- cp1 = quantidade de linhas presentes com comprimento menores que 800 pixeis;
- cp2 = quantidade de linhas presentes com comprimento entre 800 e 1600 pixeis;
- cp3 = quantidade de linhas presentes com comprimento maiores que 1600 pixeis.

Ao processar cada uma das imagens, o sistema gera um histograma com a organização conforme apresentado na Figura 34.

Figura 34 – Histograma



(Fonte: Autor)

No formato utilizado na LIBSVM, o primeiro dado é o rótulo, que é o que informa para o sistema se o palete está ou não em condições para a utilização na célula de paletização robotizada. O seu valor será ‘+1’ caso o palete esteja em condições para a citada aplicação e ‘-1’ caso não esteja. O histograma foi formatado conforme necessário para utilização na LIBSVM (CHANG E LIN, 2011).

Conforme informado na seção 4.2, foram gerados dois arquivos de dados. O primeiro deles para realizar a etapa de treinamento da SVM, com 45 histogramas de paletes em condições e 45 de paletes sem condição para a aplicação em questão. O segundo deles para realizar os testes e validação do sistema, com 15 histogramas de cada uma das classes de paletes.

Com os dados de treinamento e teste gerados, a função *grid*, presente na LIBSVM (CHANG E LIN, 2011), foi utilizada para otimizar a determinação dos parâmetros C e $gamma$ da Função Kernel RBF. A utilização dessa função como padrão foi dada visto que segundo Min e Lee (2005) ela possui menor velocidade de processamento comparado com a Função Kernel Polinomial, além de muitas vezes apresentar resultados melhores que as demais Funções Kernel tendo como base estudos anteriores. Em seu próprio trabalho Min e Lee (2005) apresentam uma comparação de desempenho entre as principais Funções Kernel, na qual a RBF apresentou desempenho superior com relação as demais.

Para utilização da função *grid*, é necessário determinar alguns valores, são eles: o intervalo de teste e incremento de $\log_2 C$, intervalo de teste e incremento de $\log_2 gamma$ e a quantidade de divisões dos dados de treinamento para realização da validação cruzada presente na função. A função *grid* realiza então diversas simulações de treinamento com os valores dentro dos intervalos e variando conforme o incremento determinado retornando através da validação cruzada a capacidade de previsão da SVM com aquele determinado conjunto de parâmetros.

O que a função faz é separar aleatoriamente os dados de treinamento na quantidade determinada pelo usuário. Ela então destaca um desses grupos, treina o sistema com os demais dados e com valores dos parâmetros dentro do intervalo determinado e testa a capacidade de previsão no grupo previamente separado. Na sequência, a função realiza o mesmo processo, porém alterando o grupo destacado para testar a capacidade de previsão. Essa é a chamada validação cruzada. Após finalizar esse processo para determinados valores nos parâmetros, esses valores são alterados e o processo inicia novamente.

A função *grid* retorna ao final do processo o par de valores que resultaram na capacidade de previsão mais efetiva e uma imagem no estilo curvas de nível com as diferentes combinações e resultados.

Caso não sejam determinados o intervalo e o incremento para um dos parâmetros, é mantido conforme padrão da LIBSVM. O mesmo acontece para a quantidade de grupos em que os dados de treinamento devem ser divididos. Nesse caso, a quantidade padrão é 5.

Foi realizado um teste com o objetivo de verificar qual a quantidade de grupos que retorna parâmetros capazes de gerar uma estimativa mais efetiva. Durante o teste os intervalos de ambos parâmetros foram mantidos entre -10 e 10 e o incremento em 1. Os dados utilizados foram os histogramas utilizados para treinamento gerados das imagens dos paletes. O resultado desse teste é apresentado na Tabela 5.

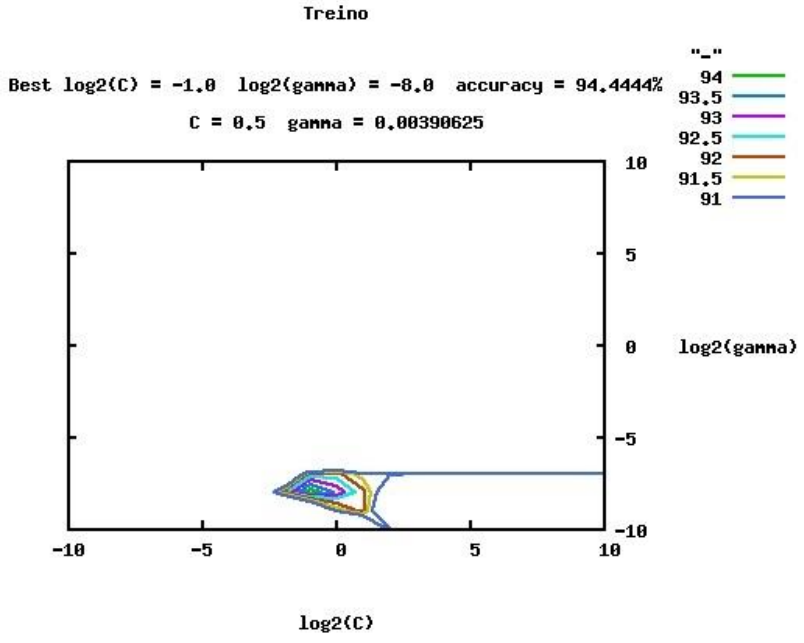
Tabela 5 – Teste com diferentes quantidades de grupos com a ferramenta *grid* com intervalos de C e gamma -10 a 10

Quantidade de Grupos	Resultados		
	C	$gamma$	Acurácia
2	1	0,00390625	88,89%
3	0,5	0,00390625	92,22%
4	0,5	0,0078125	94,44%
5	0,5	0,00390625	94,44%
6	0,5	0,0078125	93,33%
7	0,5	0,00390625	94,44%
8	1	0,00390625	94,44%
9	0,5	0,00390625	93,33%
10	0,5	0,00390625	93,33%
11	1	0,0078125	94,44%
12	0,5	0,00390625	94,44%
13	1	0,0078125	92,22%
14	1	0,0078125	93,33%
15	0,5	0,00390625	93,33%

(Fonte: Autor)

Conforme a Tabela 5, com seis valores diferentes de quantidade de grupos foi possível obter 94,44% de acurácia na classificação com os valores dos parâmetros da Função Kernel RBF C e $gamma$ variando de 1 a 0,5 e de 0,00390625 a 0,0078125, respectivamente. A Figura 35 apresenta as curvas de nível resultantes do teste com quantidade de grupos de dados padrão da LIBSVM, ou seja, cinco.

Figura 35 – Resultado *grid* intervalos C e gamma -10 a 10



(Fonte: Autor)

Para saber se é possível encontrar valores para os parâmetros da Função Kernel RBF que gerem resultados com maior acurácia, um segundo teste foi realizado, reduzindo o incremento de 1 para 0,5. Na Figura 35 é possível notar que os intervalos de $\log_2 C$ e $\log_2 gamma$ podem ser menores e como o incremento teve um redução considerável, os intervalos também foram reduzidos, poupando assim tempo de processamento. O intervalo de $\log_2 C$ utilizado foi entre -2 e 2 e o intervalo de $\log_2 gamma$ foi utilizado entre -10 e -6. Assim como no primeiro teste, foram utilizados diferentes quantidade de grupos para validação cruzada e o resultado pode ser visto na Tabela 6.

Tabela 6 – Teste com diferentes quantidades de grupos com a ferramenta *grid* com intervalo C -2 a 2 e gamma -10 a -6

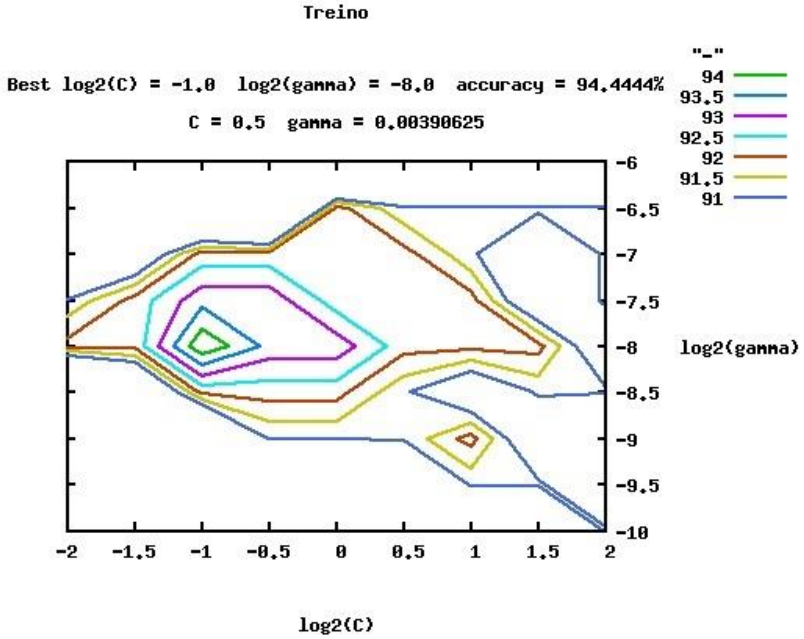
Quantidade de Grupos	Resultados		
	C	$gamma$	Acurácia
2	1	0,005524272	90,00%
3	0,707106781	0,00390625	93,33%
4	0,5	0,0078125	94,44%
5	0,5	0,00390625	94,44%
6	0,353553391	0,005524272	94,44%
7	0,5	0,00390625	94,44%
8	1	0,00390625	94,44%
9	0,353553391	0,00390625	93,33%
10	0,5	0,00390625	93,33%
11	1	0,0078125	94,44%
12	0,5	0,00390625	94,44%
13	0,353553391	0,0078125	92,22%
14	0,353553391	0,00390625	93,33%
15	0,5	0,005524272	94,44%

(Fonte: Autor)

A Tabela 6 mostra que, conforme esperado, ao reduzir os valores de incremento a LIBSVM obtém resultados mais precisos, sendo que com oito valores diferentes de quantidade de grupos foi possível obter 94,44% de acerto na classificação. Porém não foi possível obter com nenhuma quantidade de grupos um resultado mais preciso.

Para treino da SVM, foram utilizados os valores dos parâmetros encontrados com cinco grupos por esse ser o valor padrão da LIBSVM. Os valores utilizados, portanto, foram: C = 0,5 e $gamma$ = 0,00390625. A Figura 36 apresenta as curvas de nível resultantes do teste que geraram os valores adotados.

Figura 36 – Resultado *grid* intervalo C -2 a 2 e gamma -10 a -6



(Fonte: Autor)

Após a definição dos parâmetros de treinamento da SVM, foi realizado o treinamento com os dados dos paletes utilizando a Função Kernel RBF e, na sequência, o teste de classificação dos paletes. O teste foi dividido em duas partes. Na primeira parte, foram utilizados os mesmos dados treinados para a SVM realizar a previsão das classes. Na segunda parte, foi realizada então a classificação com os dados separados para teste.

A primeira parte apresentou 95,56% de acerto na classificação, acertando 86 dos 90 histogramas apresentados, sendo que 50% dos erros foram em histogramas de paletes em condições para uso em uma célula de paletização robotizada e os outros 50% em histogramas de paletes sem condições. Esse resultado está sintetizado na matriz de confusão apresentada na Tabela 7.

Tabela 7 – Matriz de confusão teste de classificação com dados de treino

		Valor Previsto	
		Positivo	Negativo
Valor Real	Positivo	43	2
	Negativo	2	43

(Fonte: Autor)

A segunda parte apresentou 93,33% de acerto na classificação, acertando 28 dos 30 histogramas apresentados. Nesse caso, os dois histogramas com classificações de maneira errada são de paletes sem condições para utilização na citada aplicação. Esse resultado está sintetizado na matriz de confusão apresentada na Tabela 8.

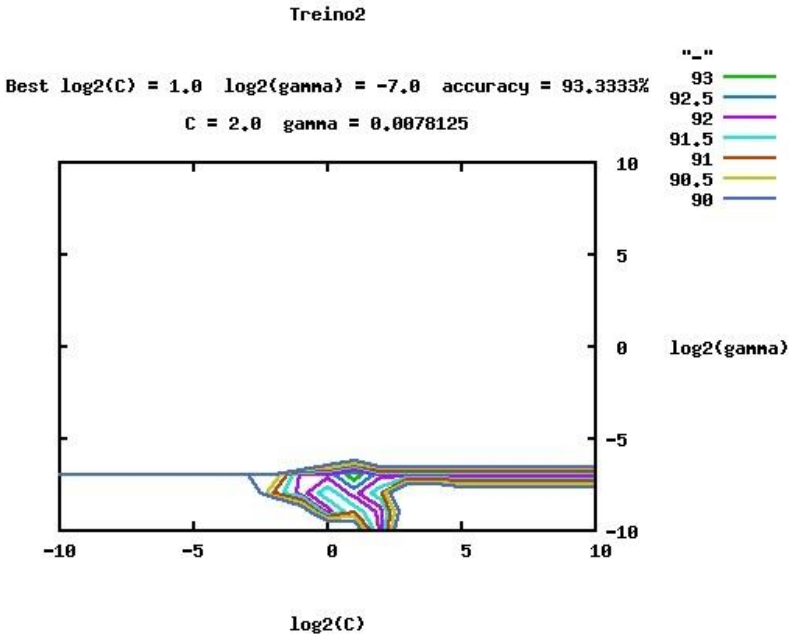
Tabela 8 – Matriz de confusão teste de classificação com dados de teste

		Valor Previsto	
		Positivo	Negativo
Valor Real	Positivo	15	0
	Negativo	2	13

(Fonte: Autor)

Como o resultado ficou abaixo do esperado, foi realizado então um segundo teste de classificação, porém utilizando o valor 155 no parâmetro de máxima lacuna permitida. Para determinação dos parâmetros da Função Kernel RBF foi utilizada novamente a função *grid*, adotando intervalos de ambos parâmetros entre -10 e 10 e o incremento em 1. A quantidade de grupos utilizada foi 5, visto que foi o valor que gerou os melhores resultados nos testes anteriores. A Figura 37 apresenta as curvas de nível resultantes do teste descrito.

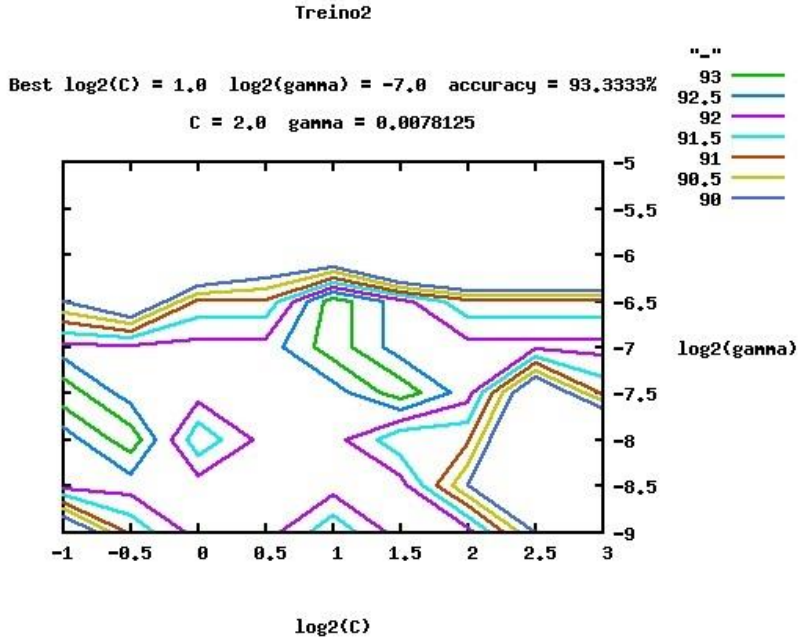
Figura 37 – Resultado segundo teste *grid* intervalos C e gamma -10 a 10



(Fonte: Autor)

A acurácia obtida foi de 93,33% com valores de C = 2 e $gamma$ = 0,0078125. Na sequência foi realizado outro teste, reduzindo o incremento de 1 para 0,5. O intervalo de $\log_2 C$ utilizado foi entre -1 e 3 e o intervalo de $\log_2 gamma$ foi foi utilizado entre -9 e -5. A Figura 38 apresenta as curvas de nível resultantes deste teste.

Figura 38 – Resultado segundo teste *grid* intervalo C -1 a 3 e gamma -9 a -5



(Fonte: Autor)

Novamente a acurácia obtida foi de 93,33% com valores de C = 2 e $gamma$ = 0,0078125. Esses foram, portanto, os valores adotados no treino da SVM. O teste seguiu exatamente os mesmos procedimentos do teste anterior, testando inicialmente os mesmos dados treinados para a SVM realizar a previsão das classes e na sequência realizando a classificação dos dados separados para teste.

A primeira parte apresentou 98,89% de acerto na classificação, acertando 89 dos 90 histogramas apresentados. Conforme apresentado na matriz de confusão da Tabela 9, a classificação errada nesse caso configura um falso negativo, ou seja, um palete que tinha condições de uso para a citada aplicação foi classificado como se não tivesse condições. Para a aplicação em questão, falsos negativos são menos prejudiciais que falsos positivos, pois um palete sem condições ser classificado como se tivesse condições levaria a problemas na célula de paletização.

Tabela 9 – Matriz de confusão segundo teste de classificação com dados de treino

		Valor Previsto	
		Positivo	Negativo
Valor Real	Positivo	44	1
	Negativo	0	45

(Fonte: Autor)

No teste com os dados separados apenas para classificação, o resultado obtido foi de 100% de acerto. O resultado do segundo teste está sintetizado na matriz confusão apresentada na Tabela 10.

Tabela 10 – Matriz de confusão do segundo teste de classificação com dados de teste

		Valor Previsto	
		Positivo	Negativo
Valor Real	Positivo	15	0
	Negativo	0	15

(Fonte: Autor)

Para comparação do desempenho das diferentes Funções Kernel citadas na sessão 2.5.1, foram realizados testes com os mesmos dados de treinamento e teste utilizados no desenvolvimento do sistema com a Função Kernel RBF. Para todas as funções o valor do parâmetro C foi 2. Nas funções que utilizam o parâmetro $gamma$ o seu valor foi 0,0078125. O parâmetro r foi mantido em 0, como padrão da LIBSVM. O grau polinomial foi variado entre 2 e 5. O parâmetro h foi mantido em 1, como padrão da LIBSVM. Os resultados obtidos estão sintetizados na Tabela 11.

Tabela 11 – Comparação de desempenho entre Funções Kernel

Função Kernel	C	$gamma$	Grau polinomial	Acerto	
				Dados Treinados	Dados de Teste
RBF	2	0,0078125	-	98,89%	100%
Linear	2	-	-	86,67%	80%
Polinomial	2	0,0078125	2	91,11%	83,33%
	2	0,0078125	3	95,56%	93,33%
	2	0,0078125	4	93,33%	83,33%
	2	0,0078125	5	86,67%	83,33%
Sigmoidal	2	0,0078125	-	40%	46,67%

(Fonte: Autor)

Conforme citado com base em estudos anteriores e demonstrado por Min e Lee (2005) a Função Kernel RBF possui desempenho superior às demais. A Função Kernel Polinomial apesar de apresentar maior tempo de processamento comparado com a RBF apresentou um resultado interessante quando considerado grau polinomial 3. Foi realizado então mais um teste com a função Polinomial com grau 3, variando r entre 0 e 3. O resultado pode ser visto na Tabela 12.

Tabela 12 – Desempenho Função Kernel Polinomial

r	Acerto	
	Dados Treinados	Dados de Teste
0	95,56%	93,33%
1	100%	96,67%
2	100%	96,67%
3	100%	96,67%

(Fonte: Autor)

Com o valor de r maior que zero o resultado obtido de 100% de acerto na classificação dos dados treinados e 96,67% de acerto na classificação dos dados mostra que a Função Kernel Polinomial pode apresentar resultados similares aos resultados apresentados pela Função Kernel RBF, porém com maior tempo de processamento. Foi elaborada uma matriz de confusão para sintetizar o resultado na classificação dos dados de teste com a Função Kernel Polinomial com grau 3 e r = 1. A matriz de confusão pode ser vista na Tabela 13.

Tabela 13 – Matriz de confusão Função Kernel Polinomial dados de teste

		Valor Previsto	
		Positivo	Negativo
Valor Real	Positivo	14	1
	Negativo	0	15

(Fonte: Autor)

6. CONCLUSÃO

O estudo bilbiográfico sobre sistemas de visão mostrou que existe, conforme descrito no capítulo 3, uma lacuna em relação ao detalhamento das técnicas utilizadas. Além disso, também não existem trabalhos com sistemas de visão aplicados na análise de paletes de madeira. Este trabalho pode ajudar a preencher ambas as lacunas, contribuindo com futuros estudantes e pesquisadores no desenvolvimento de sistemas de visão e aplicação de técnicas de análise e processamento de imagem.

No desenvolvimento do sistema de visão para verificação de paletes, procurou-se selecionar as técnicas de análise e processamento de imagem que apresentaram os melhores resultados baseado em estudos passados e que possuem características necessárias para solucionar o problema em questão, identificar os paletes sem condições para uso em uma célula de paletização robotizada. Os conceitos teóricos de cada uma das técnicas selecionadas, bem como o funcionamento de cada um dos seus parâmetros, foram validados através de testes que comprovaram a eficácia dessas técnicas, permitindo sua aplicação no sistema de visão aqui estudado. Foram validados também os cálculos dos dados gerados pela Transformada de Hough, usados posteriormente na etapa de treino e classificação.

Importante destacar que, em todo processo, foi considerado que, pelo fato de as imagens serem obtidas em condições similares, a etapa de classificação com aprendizagem de máquina poderia resolver por si só imperfeições das etapas anteriores.

Na etapa de treino e classificação dos dados, a Função Kernel RBF apresentou 100% de eficiência na classificação com os dados de teste, ou seja, dados que foram separados especificamente para serem testados e que não foram utilizados na etapa de treinamento. Na classificação dos dados de treinamento, o resultado apresentado foi de 98,89% de acerto, apresentando um falso negativo.

A função Kernel Polinomial apresentou 100% de acerto na classificação dos dados treinados e 96,67% de acerto na classificação dos dados de teste, apresentando também um falso negativo. Esse resultado indica que, voltando nas etapas anteriores, pode ser possível obter um resultado de 100% na classificação dos dados de teste também com a função Polinomial. No entanto, para o desenvolvimento do sistema de visão, a Função Kernel RBF foi escolhida como padrão e atendeu às necessidades. Além disso, por apresentar maior tempo de processamento, a Função Kernel Polinomial é menos interessante que a RBF.

O sistema de visão desenvolvido eliminaria o uso de paletes sem condições de uso nas células de paletização robotizadas, visto que obteve 100% de acerto na classificação dos

paletes testados. Deve ser levado em consideração também que as classificações erradas, tanto com a Função Kernel RBF quanto com a Polinomial, foram falsos negativos, que na prática significa retirar do sistema um palete em condições de uso. Para a aplicação esse erro não causaria danos ou problemas na célula de paletização robotizada. Com isso, paradas, intervenção humana e possíveis perdas de produtos seriam reduzidas, aumentando a eficiência das células de paletização e possibilitando até que elas atinjam a capacidade produtiva máxima.

6.1. SUGESTÕES PARA ESTUDOS FUTUROS

O fato de utilizar aprendizagem de máquina torna o sistema capaz de analisar e classificar diferentes tipos de paletes, seja na distribuição das ripas, seja nas dimensões e até mesmo no material do palete. É preciso, no entanto, realizar testes para validar quão eficiente seriam os sistemas nessas diferentes condições.

O método cíclico para otimização dos parâmetros se mostrou eficiente, porém ele depende da análise visual de uma pessoa para selecionar as faixas de valores a serem testados. Uma possibilidade é desenvolver um algoritmo que realize a análise de maneira automática. Uma sugestão para isso seria testar os ajustes com diferentes parâmetros em todo processo até a classificação dos paletes, pois essa é a única etapa na qual pode ser medida eficiência. Ao obter 100% na classificação, o sistema retornaria então os valores encontrados de cada um dos parâmetros.

Como continuidade do trabalho desenvolvido, é possível realizar testes obtendo as imagens em uma aplicação real, com a célula de paletização em funcionamento, analisando essas imagens ainda com a utilização de um computador, ou embarcando a aplicação em um Raspberry Pi, por exemplo.

REFERÊNCIAS BIBLIOGRÁFICAS

AGUIAR, H. T.; *Desenvolvimento de um Sistema de Paletização Robotizado*. Viseu: Instituto Politécnico de Viseu, 2013.

BORRMANN, D.; ELSEBERG, J.; LINGEMANN, K.; NÜTCHER, A.; *The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design*. Berlim: 3D research, vol. 8, 2011.

BRADSKI, G.; KAEHLER, A.; *Learning OpenCV: Computer Vision with the OpenCV Library*. Sebastopol: O'Reilly Media, Inc., 2008.

CALARGE, F. A.; DAVANSO, J. C.; *Conceito de Dispositivos à Prova de Erros Utilizados na Meta do Zero Defeito em Processos de Manufatura*. Piracicaba: Unimep, 2014.

CANNY, J.; *A Computational Approach to Edge Detection*. Piracicaba: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. Pami-8, No. 6, November, 1986.

CASTRO, A. A. M. de; PRADO, P. P. L. do; *Algoritmos para Reconhecimento de Padrões*. Taubaté: Revista de Ciências Exatas, v. 5-8, p.129-145, 2002.

CHANG, C. C.; LIN, C. J.; *LIBSVM: a library for support vector machines*. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011.

CORTES, C.; VAPNIK, V.; *Support-Vector Networks*. Boston: Machine Learning, Kluver Academic Publishers, 20, p.273-297, 1995.

DJAJADI, A.; LAODA, F.; RUSYADI, R.; PRAJOGO, T.; SINAGA, M.; *A Model Vision of Sorting System Application Using Robotic Manipulator*. Tangerang: Journal Terakreditasi Dikti, Telkomnika, vol. 8 n° 2, p 137-148, 2010.

DUDA, R. O.; HART, P. E.; *Use of the Hough Transformation to Detect Lines and Curves in Pictures*. California: Graphics and Image Processing, vol. 5 n° 1, p 11-15, 1972.

GATTESCHI, E.; *Automated system for checking pallets*. Patente Estados Unidos, 2005.

GERIG, G.; *Linking Image-Space and Accumulator-Space: A New Approach for Object-Recognition*. Londre: IEEE First International Conference of Computational Vision, p 112-117, 1987.

GOEKING, W.; *Da máquina a vapor aos softwares de automação*. São Paulo: Atitude Editorial, 2010.

GUERREIRO, R. F. C.; AGUIAR, P. M. Q.; *Connectivity-Enforcing Hough Transform for the Robust Extraction of Line Segments*. Charlottesville: IEEE Transactions on Image Processing, p. 4819 – 4829, v. 21, 2012.

HOUGH, P. V. C.; *Method and Means for Recognizing Complex Patterns*. Patente Estados Unidos, 1962

JAIN, R; *Simple Tutorial on SVM and Parameter Tuning in Python and R*. Disponível em: <https://www.hackerearth.com/blog/machine-learning/simple-tutorial-svm-parameter-tuning-python-r/>. Acesso em: 26 de março de 2018.

KIRYATI, N.; ELDAR, Y.; BRUCKSTEIN, M.; *A Probabilistc Hough Transform*. Israel: Pattern Recognition, v. 24, no. 4, p. 303 - 316, 1990.

KIM, E.; *Everything You Wanted to Know about the Kernel Trick (But Were Too Afraid to Ask*). Disponível em: <http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick_blog_ekim_12_20_2017.pdf>. Acesso em: 02 de abril de 2018.

LORENA, A. C.; CARVALHO, A. C. P. L. F. de; *Uma Introdução às Support Vector Machines*. Porto Alegre: RITA, vol. XIV, no. 2, p. 43 – 67, 2007.

LIFTCOM; *Paletes*. Disponível em: <https://www.liftcom.com.br/notcias/iyz3s0jc22/Paletes> . Acesso em: 26 de fevereiro de 2018.

MAINI, R.; AGGARWAL, H.; *Study and Comparison of Various Image Edge Detection Techniques*. Punjabi: International Journal of Image Processing (IJIP), Volume (3), 2009.

MATAS, J.; GALAMBOS, C.; KITTLER, J.; *Robust Detection of Lines Using the Progressive Probabilistic Hough Transform*. Elsevier, Computer Vision and Image Understanding 78, p. 119 – 137, 1999.

MATO, J.L.; SOUTO, M.A.; BESTEIRO, R.; MOLEDO, J.A.; *Automated Counting of Palletized Slate Slabs Based on Machine Vision*. Viena: IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society, p. 2378 – 2383, 2013.

MIN, H. J.; LEE, Y. C.; *Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters*. Elsevier, Expert Systems with Applications 28, p. 603 – 614, 2005.

MUKHOPADHYAY, P.; CHAUDHURI, B. B.; *A Survey of Hough Transform*. Singapura: Elsevier, Pattern Recognition, p. 993 – 1010, 2014.

NADERNEJAD, E.; SHARIFZADEH, S.; HASSANPOUR, H.; *Edge Detection Techniques: Evaluation and Comparisons*. Babol: Applied Mathematical Sciences, Vol. 2, no. 31, 1507 – 1520, 2008

NOBLE, W. S.; *What is a support vector machine?* Londres: Nature Biotechnology, Vol. 24, no. 12, p.1565 – 1567, 2006

OPENCV; *About*. Disponível em: <https://opencv.org/about.html>. Acesso em: 21 de fevereiro de 2018.

PATRICIO, M. A.; MARAVALL, D.; *A novel generalization of the gray-scale histogram and its application to the automated visual measurement and inspection of wooden Pallets.* Elsevier, 2006.

SEMIM, R. C., 2012. *Sistema de Visão para Guiar um Robô de Manipulação de Cabeçotes Fundidos*. Joinville: Universidade do Estado de Santa Catarina, 2012.

SONKA, M.; HLAVAC, V.; BOYLE, R., 2014. *Image Processing, Analysis, and Machine Vision*. Stamford: Cengage Learning, Ed. 4, 2014.

TOWNSEND, S.; LUCAS, M. D.; *Automated pallet inspection and repair*. Patente Estados Unidos, 2010

TOWNSEND, S.; LUCAS, M. D.; *Software and methods for automated pallet inspection and repair*. Patente Estados Unidos, 2015