FERNANDO DESCHAMPS

CONTRIBUIÇÕES PARA O DESENVOLVIMENTO DE UM SISTEMA DE VISÃO APLICADO AO MONITORAMENTO DO DESGASTE DE FERRAMENTAS DE CORTE – O SISTEMA TOOLSPY

FLORIANÓPOLIS
2004

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

CONTRIBUIÇÕES PARA O DESENVOLVIMENTO DE UM SISTEMA DE VISÃO APLICADO AO MONITORAMENTO DO DESGASTE DE FERRAMENTAS DE CORTE – O SISTEMA TOOLSPY

Dissertação submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a
obtenção do grau de Mestre em Engenharia Elétrica.

FERNANDO DESCHAMPS

Florianópolis, Março de 2004.

CONTRIBUIÇÕES PARA O DESENVOLVIMENTO DE UM SISTEMA DE VISÃO APLICADO AO MONITORAMENTO DO DESGASTE DE FERRAMENTAS DE CORTE – O SISTEMA TOOLSPY

Fernando Deschamps

'Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Engenharia Elétrica, Área de Concentração em *Automação e Sistemas*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.'

Prof. Marcelo Ricardo Stemmer, Dr.-Ing.
Orientador

Prof. Jefferson Luiz Brum Marques, Ph.D.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca Examinadora:

Prof. Marcelo Ricardo Stemmer, Dr.-Ing.
Presidente

Prof. Armando Albertazzi Gonçalvez Jr., Dr. Eng.

Prof. Rolf Bertrand Schroeter, Dr. Eng.

Prof. Rômulo Silva de Oliveira, Dr. Eng.

"O maior elogio que recebi em toda

a minha vida de inventor foi:

isto nunca vai funcionar!"

Thomas Alva Edison

AGRADECIMENTOS

Aos meus pais Genésio e Irma, pelo apoio e amor incondicional e por me ensinarem a importância do conhecimento e aos meus irmãos (e esposas!) Eduardo e Adriana, Marcelo e Andréa e Suzana, pelo sempre importante apoio nas horas difíceis.

À Roberta, pelo amor, apoio e compreensão nos vários momentos difíceis pelos quais passei durante o desenvolvimento deste trabalho. Um pedaço muito grande de ti está neste trabalho. Obrigado de coração.

Ao meu sobrinho Luiz Eduardo e à minha sobrinha Gabriela, por me fazerem acreditar no futuro.

Ao professor e amigo Marcelo Ricardo Stemmer, por toda a força e apoio durante o desenvolvimento deste trabalho. Mais do que um orientador, você foi um grande amigo que já me atura durante quase 5 anos sem nunca reclamar!

Aos meus amigos Fernando Aloísio Carreirão, Charles Alberton Herdt e Carlo Enrico Bressiani, pessoas com quem eu sempre posso contar, em qualquer situação.

Ao amigo Gustavo Bouzon, por sempre topar uma boa corrida, um bom futebol, um bom café e uma boa conversa!

À amiga Josiane Milanez, também por sempre topar um bom café e uma boa conversa (não topando a corrida e o futebol)!

Ao amigo Ricardo Grützmacher (valeu revisor!), que me ensinou o valor do software livre e com quem aprendi muito mais do que ensinei.

Ao professor e amigo Mário Cesar Zambaldi, por sempre me fazer enxergar alternativas.

Ao Marcelo Moraes Minasi, por topar o desafio de uma idéia inovadora – diz aí Minasi, por que agora não integrar o algoritmo de eigenfaces para uso nas telas de login gráfico do GDM?

Ao Lucas Barbosa Sanches, por ter sido a primeira verdadeira cobaia do MtqIPFramework, por ter sido cobaia do S2iIPFramework e por ter aprimorado minhas idéias sobre esta arquitetura.

Aos amigos Alberto Xavier Pavim (valeu revisor), Alexandre Orth e Mário Lucio Roloff, por me ajudarem no desenvolvimento deste trabalho.

Ao amigo Dominic Sack, por adorar o Brasil, adorar caipirinha, adorar churrascarias e pela ajuda na programação (“let’s do some power programming, Fernando!”).

A todo o pessoal do S2i (inclusive os gaúchos!) Adriano Winter Bess (valeu revisor!), Daniel Gomes de Moraes, Eduardo Pigozzi Cabral, Fabio Luis Baldissera, Fábio Pedrotti Terra, Fabricio Forgerini, Guilherme Francisco Mallmann, José Luiz Bittencourt, Marcelo Pires Adur, Mathias José Kreutz Erdtmann, Rafael Moreira Miggiorin e Ricardo Levi Donada, pelas boas gargalhadas e descontração no trabalho.

Resumo da Dissertação apresentada à UFSC como parte dos requisitos necessários para obtenção do grau de Mestre em Engenharia Elétrica.

CONTRIBUIÇÕES PARA O DESENVOLVIMENTO DE UM SISTEMA DE VISÃO APLICADO AO MONITORAMENTO DO DESGASTE DE FERRAMENTAS DE CORTE – O SISTEMA TOOLSPY

Fernando Deschamps

Março de 2004

Orientador: Prof. Marcelo Ricardo Stemmer, Dr.-Ing.
Área de Concentração: Automação e Sistemas
Palavras-chave: sistemas de visão, monitoramento do estado de ferramentas de corte, desgaste de ferramentas de corte, processamento de imagens, visão computacional, células autônomas de produção.
Número de Páginas: 157

A área de desenvolvimento de sistemas para o monitoramento do desgaste de ferramentas de corte vem recebendo grande atenção tanto do meio acadêmico quanto do meio industrial. A complexidade do assunto atrai o interesse da comunidade científica, ao passo que a redução de custos com menor desperdício de matéria-prima e ferramentas e o aumento da qualidade de produtos tornam esta área bastante atrativa aos olhos do setor produtivo. Duas são as principais abordagens para este problema: métodos indiretos, baseados na correlação de algum sinal proveniente do processo com o desgaste da ferramenta e os métodos diretos, baseados na avaliação do desgaste a partir da própria ferramenta. A utilização de um método direto, como o baseado em sistemas de visão, possibilita uma avaliação mais confiável do real estado da ferramenta, fornecendo medidas quantitativas e qualitativas do desgaste. Neste trabalho, o sistema TOOLSPY para o monitoramento do estado de ferramentas de corte é apresentado, sendo ênfase dada a sua cadeia de processamento de imagens e visão computacional e a estrutura do programa que implementa esta cadeia de processamento. O sistema descrito e proposto está em estado de testes e a ponto de poder ser validado através de parceriais com o setor industrial. A arquitetura IPFRAMEWORK bem como a cadeia de processamento de imagens e visão computacional do sistema e os diversos módulos montados para a implementação da mesma junto à biblioteca S2ILIB são apresentados e discutidos. Resultados da aplicação da estrutura IPFRAMEWORK ao programa do sistema TOOLSPY e os resultados obtidos com este na medição de pastilhas de torneamento são mostrados.

Abstract of Dissertation presented to UFSC as a partial fulfillment of the requirements for the degree of Master in Electrical Engineering.

CONTRIBUTIONS TO THE DEVELOPMENT OF A COMPUTER VISION SYSTEM APPLIED TO WEAR MONITORING ON CUTTING TOOLS – THE TOOLSPY SYSTEM

Fernando Deschamps

March/2004

Advisor: Prof. Marcelo Ricardo Stemmer, Dr.-Ing.
Area of Concentration: Automation and Systems
Key words: vision systems, wear monitoring on cutting tools, tool wear, image processing, computer vision, autonomous production cells.
Number of Pages: 157

The development of monitoring systems for the evaluation of tool wear on cutting tools has drawn the attention of both academic and industrial communities. The complexity of this subject attracts the interest of the scientific community. The possible reduction of costs due to the decrease of waste of raw materials and cutting tools and the increase in product quality have a major appeal to the manufacturing industry. There are two main approaches to the wear monitoring problem on cutting tools: indirect methods are based on the correlation between a signal from the manufacturing process with the amount of tool wear generated; direct methods evaluate tool wear by analysis of the cutting tool itself. The use of a direct method, such as a computer vision system, allows a more reliable evaluation of the real tool condition with quantitative and qualitative results about the wear. In this work, the TOOLSPY system for tool condition monitoring is presented. The image processing and computer vision chain of this system as well as the structure of the program that implements this chain are emphasized. This system is in state of being validated through partnerships with industry. The IPFRAMEWORK architecture as well as the image processing and computer vision chain of the TOOLSPY system as well as the several modules programmed for the implementation of this processing chain in the S2ILIB software library are presented and discussed. Results of the application of the IPFRAMEWORK structure to the TOOLSPY program and the results obtained with this final system in the evaluation of tool wear in turning inserts are shown.

Sumário

Lista de Figuras

Lista de Tabelas

Nomenclatura

DOF Profundidade de campo, do inglês *Depth of Field*

FOV Campo de visão, do inglês *Field of View*

R Resolução, do inglês *Resolution*

SS Tamanho do sensor, do inglês *Sensor Size*

WD Distância de trabalho, do inglês *Working Distance*

TOOLSPYOPTIMIZER Programa de computador para a plataforma Windows desenvolvido para a otimização dos parâmetros da cadeia de processamento de imagens do sistema TOOLSPY

ABIMAQ Associação Brasileira dos Fabricantes de Máquinas e Equipamentos

ANSI *American National Standards Institute*, instituto estado-unidense responsável pela definição e manutenção de padrões

APC Célula Autônoma de Produção, do inglês *Autnomoous Production Cell*

API Interface para a Programação de Aplicações, do inglês *Application Programming Interface*

BMP Formato de armazenamento de imagens, do inglês *Bitmap*

CAD Projeto assistido por computador, do inglês *Computer Aided Design*

CBN Nitreto de Boro Cúbico, do inglês *Cubic Boron Nitrite*, um tipo de material possível para ferramentas de corte

CCD Dispositivos de carga acoplados, do inglês *Charged Coupled Devices*, um tipo de sensor empregado em de sistemas de visão

CMOS Semicondutor de óxido-metal complementar, do inglês *Complementary Metal Oxide Semiconductor*, um tipo de sensor empregado em sistemas de visão

CNC Comando Numérico Computadorizado

CVS Controle Concorrente de Versões, do inglês *Concurrent Versions Sysmte*, usado para controlar o desenvolvimento de programas e suas diferentes versões.

DFG Conselho de Pesquisas Alemão, do alemão *Deutsche Forschungsgemeinschaft*, órgão do governo responsável pelo financiamento de pesquisas

FFT Transformada Rápida de Fourier, do inglês *Fast Fourier Transform*

IEEE 1394 Padrão de transmissão digital do IEEE, *Institute of Electrical and Electronic Engineers*, comumente referido como *Firewire*

JPG Também referido como JPEG, é um formato de armazenamento de imagens, do inglês *Joint Photography Expert Group*

LCD Monitor de cristal líquido, do inglês *Liquid Crystal Display*

LMP Laboratório de Mecânica de Precisão da Universidade Federal de Santa Catarina, vinculado ao Departamento de Engenharia Mecânica

LVDS Sinal diferencial de baixa voltagem, do inglês *Low Voltage Differential Signal*, um modo de transmissão de dados do sensor para o subsistema de processamento empregado em sistemas de visão

MNG Formato de armazenamento de seqüências de imagens, do inglês *Multiple Network Graphics*

OpenCV *Open Computer Vision Library*, biblioteca de processamento de imagens e visão computacional desenvolvida inicialmente pela empresa Intel e atualmente mantida pela comunidade de software livre

PCA Análise dos Componentes Principais, do inglês *Principal Component Analysis*, um método para a classificação de padrões baseados em projeções e operações sobre matrizes e vetores.

PNG Formato de armzenamento de imagens, do inglês *Portable Network Graphics*

RNA Rede Neural Artificial, campo de estudo da área de inteligência artificial

ROI Do inglês *Regio of Interest*, é uma região de interesse da qual se deseja obter alguma informação em uma imagem maior

RWTH-Aachen Universidade Técnica de Aachen, do alemão *Rheinisch-Westfälische Technische Hochschule Aachen*

SFB368 Área Especial de Pesquisa 368, do alemão *Sonderforschungsbereich 368*, uma área de pesquisa do Conselho Alemão de Pesquisa que trata da pesquisa em Células Autônomas de Produção

UML Linguagem de Modelagem Unificada, uma linguagem de modelagem para programas concebidos de acordo com o paradigma de programação de orientação a objetos, do inglês *Unified Modeling Language*, uma linguagem de modelagem de software orientado a objetos

WZL Laboratório de Máquinas-Ferramenta e Engenharia Industrial da Universidade Técnica de Aachen, do alemão *Laboratorium für Werkzeugmaschinen und Betriebslehre*

Capítulo 1

Introdução

O problema de monitoramento do desgaste de ferramentas de corte em processos de fabricação metal-mecânica tem recebido grande atenção tanto da comunidade científica quanto do meio industrial nos últimos anos. Dada a complexidade desse tema, muitos artigos são publicados em periódicos e congressos especializados [1, 2, 3, 4, 5]. Grande é também a importância econômica deste problema, visto que o emprego de ferramentas de corte em processos de fabricação é amplo, como em processos de torneamento, fresamento, furação, mandrilamento, entre outros. Produtos comerciais nesta área também já são uma realidade [6]. A figura 1.1 ilustra o faturamento do setor metal-mecânico no Brasil no período de 1995 a 2003.

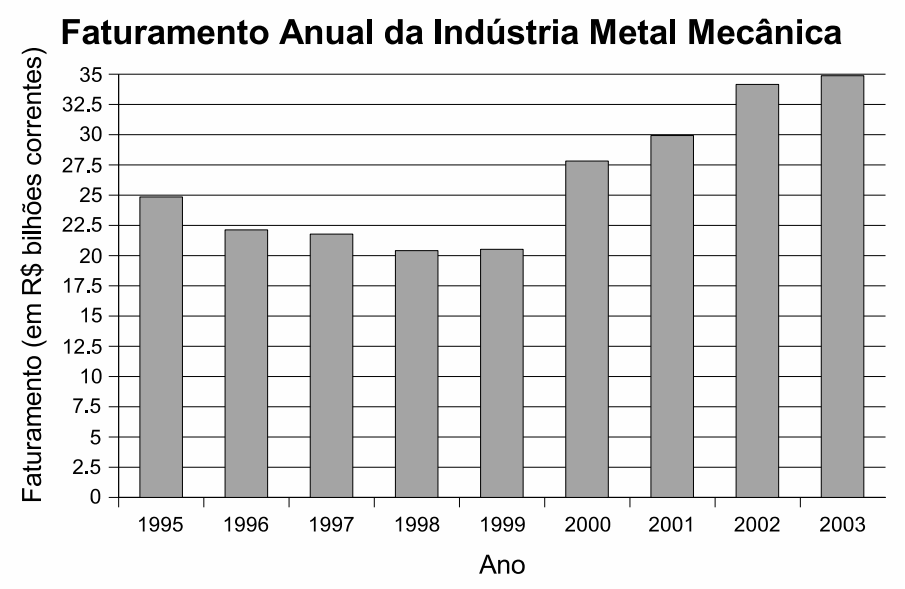


Figura 1.1: *Faturamento da indústria metal-mecânica no Brasil no período de 1995 a 2003.*
Fonte: ABIMAQ - http://www.abimaq.org.br.

Em máquinas-ferramenta modernas, 20% do tempo improdutivo é atribuído a algum tipo de falha da ferramenta, resultando na redução da produtividade e perdas econômicas [5]. Estas perdas estão associadas à parada do processo para a troca de uma ferramenta quebrada, ao descarte de matéria-prima e a danos à máquina-ferramenta pelo uso de uma ferramenta danificada, bem como à perda

da qualidade do produto decorrente do uso de uma ferramenta excessivamente desgastada, sendo este posteriormente descartado pelo próprio processo ou causando insatisfação ao consumidor que o adquire. Um sistema de monitoramento confiável poderia prevenir esses problemas e possibilitar a máxima utilização da ferramenta em seu ciclo de vida, o que é extremamente desejável, aumentando-se o ganho econômico com a ferramenta através da economia de custos de produção.

Além disso, um sistema de monitoramento eficiente pode otimizar o processo produtivo através das informações geradas [4]. De forma sucinta, tal sistema pode classificar o desgaste da ferramenta, indicando para qual atividade de corte ela é mais adequada (desbaste ou acabamento, por exemplo) ou mesmo se ela deve ser descartada. Essas informações podem ainda ser usadas para alterar as condições de usinagem do processo, quando possível, de forma que as causas do desgaste sejam minimizadas.

Vários estudos apontaram a importância da tecnologia de monitoramento no desenvolvimento de sistemas flexíveis de manufatura [7, 8, 9]. Nestes sistemas, o monitoramento do estado das ferramentas de corte assume especial relevância no sentido de aumentar a autonomia do processo sem a necessidade de um operador humano, flexibilizando o uso do sistema através de um melhor controle sobre sua operação.

Existem dois grandes grupos de técnicas para o monitoramento do estado de ferramentas de corte [3]. No primeiro grupo estão as chamadas técnicas diretas, nas quais a própria ferramenta é utilizada como objeto de análise para se estimar seu desgaste, como na medição através de microscopia óptica e na medição através de visão computacional. No segundo grupo estão as chamadas técnicas indiretas, nas quais algum parâmetro do sistema no qual a ferramenta de corte está inserida é avaliado para se estimar o desgaste, como na medição através de sensores de força colocados na base da haste da ferramenta e na medição através de sensores de vibração e de emissão acústica colocados em pontos pré-determinados da máquina-ferramenta. Ambas as técnicas são interessantes e possuem vantagens e desvantagens próprias, como será visto no capítulo 2. A motivação da utilização de um método direto se dá pelo fato de o desgaste estar sendo avaliado a partir da própria ferramenta que é o objeto de estudo, não sendo necessária a geralmente complicada validação de algum modelo empírico ou teórico para a comprovação da relação deste modelo com o desgaste, como ocorre com os métodos indiretos.

Avanços recentes no campo de processamento de imagens levaram ao desenvolvimento de eficientes sensores que podem ser utilizados para se obter informações sobre as ferramentas de corte, bem como sobre as peças usinadas. A velocidade e a ausência de contato físico com a ferramenta fazem deste tipo de monitoramento extremamente interessante. Técnicas de inteligência artificial podem ser integradas ao sistema de visão de forma a fornecer um melhor entendimento do problema do desgaste da ferramenta. Redes neurais artificiais [10, 11], lógica nebulosa [12, 13] e algoritmos genéticos [14, 15, 16] são exemplos de técnicas empregadas na construção de sistemas com a capacidade de generalização. A partir destas técnicas, sistemas com a capacidade de aprendizado também são possíveis.

Esta dissertação de mestrado está inserida no contexto de um projeto de cooperação estabelecido entre o Departamento de Automação e Sistemas (DAS) da Universidade Federal de Santa Catarina

(UFSC) e o Laboratório de Máquinas-Ferramenta e Engenharia Industrial (WZL – *Laboratorium für Werkzeugmaschinen und Betriebslehre*) da Universidade Técnica de Aachen (RWTH-Aachen – *Rheinisch-Westfälische Technische Hochschule Aachen*) na Alemanha. O projeto de cooperação tem por objetivo o apoio no desenvolvimento de um sistema autônomo para a medição e classificação do desgaste de ferramentas de corte em células autônomas de produção (APCs – Autonomous Production Cells) através de um sistema de visão – o sistema TOOLSPY. O título oficial do projeto é "Medidas Técnicas para a Elevação da Autonomia de Células de Produção". Esta parceria já existe desde o ano 2000, sendo que diversos projetos de fim de curso e duas dissertações de mestrado [17, 18] já foram defendidas em seu âmbito.

Em uma dessas dissertações anteriores, Orth [17] abordou de uma forma geral o problema do desenvolvimento de um sistema de visão para a medição e classificação do desgaste de flanco de ferramentas de corte. Apesar do sistema ter tido bons resultados em alguns casos específicos em que foi testado, algumas dificuldades foram encontradas principalmente quanto aos dispositivos utilizados e à tentativa de integração destes em uma máquina-ferramenta, sendo eles:

- O sistema óptico utilizado possuía uma série de deficiências, dentre elas o conjunto de lentes, espaçadores e sensores utilizados, que não eram os mais adequados. Em particular os espaçadores utilizados no sistema eram muito grandes para a aplicação em questão, uma vez que a abertura da íris com a lente utilizada deveria ser pequena para possibilitar uma boa profundidade de campo.

- O sistema de iluminação também não era o mais adequado. Um sistema de iluminação com fibras ópticas direcionáveis foi utilizado para os testes, sendo que este necessitava de constantes ajustes manuais para que bons resultados fossem obtidos, comprometendo a autonomia da solução. Além disso, devido às perdas no sistema óptico, uma grande quantidade de luz era necessária para que se pudesse realçar determinadas características nas peças, demandando um sistema de iluminação bastante especializado.

- A implementação do sistema não levou em consideração um projeto modular para o software que permitisse que novos algoritmos para a cadeia de processamento de imagens fossem testados de uma forma fácil e eficiente, encapsulando também funcionalidades para a seleção e determinação de parâmetros.

- A classificação do desgaste no sistema proposto é realizada após a cadeia de processamento de imagens já ter sido totalmente executada. A idéia é que em casos de quebra da ferramenta não seja necessário efetuar todo o processamento da imagem e que também o sistema possa identificar a presença de desgaste de cratera em determinadas ferramentas, permitindo o ajuste das condições de usinagem.

- A calibração do sistema não foi realizada, sendo que os resultados obtidos não puderam ser corretamente validados.

- O sistema utilizado não foi testado em uma câmera com um processador embarcado, de forma a possibilitar a validação do sistema em condições de produção.

Somando-se a estes problemas, existe ainda a questão da equipe responsável pelo projeto ser internacional, com parte do desenvolvimento sendo feito no Brasil e parte na Alemanha. Protótipos do sistema estão sendo construídos tanto no Brasil quanto na Alemanha, com alguns dispositivos de suporte bastante heterogêneos, como por exemplo as placas de E/S para o acionamento dos dispositivos de iluminação e o sistema operacional utilizado para a execução do programa final. A idéia, no entanto, é que a parte principal do sistema se mantenha sem modificações, como é o caso de dispositivos específicos para os sistemas ópticos e de iluminação e a cadeia de processamento de imagens e seus algoritmos.

Este trabalho tem por objetivo complementar o trabalho anteriormente realizado e também resolver as questões levantadas no parágrafo anterior de forma a aumentar a autonomia do sistema, modularizando-o a fim de que a solução obtida se torne portável e possa ser empregada para diferentes tipos de máquinas, processos e ferramentas. A idéia com este trabalho é atingir duas metas:

- Desenvolver uma arquitetura modular para a cadeia de processamento de imagens do sistema de forma que este, em sua essência, possa ser independente de plataforma de execução e sistema operacional, permitindo que novos algoritmos possam ser testados e os programas do sistema possam ser desenvolvidos com maior facilidade tanto no Brasil quanto na Alemanha.
- Aprimorar o projeto geral da cadeia de processamento de imagens e visão computacional e a documentação do sistema, através de uma revisão bibliográfica mais profunda e o desenvolvimento de um projeto modular para o sistema de iluminação e novos algoritmos para a cadeia de processamento de imagens e visão computacional, permitindo às equipes brasileira e alemã a implementação da mesma estratégia.

Para isto, a análise dos componentes de um sistema de visão para o monitoramento do estado de ferramentas de corte será desenvolvida. Ênfase será dada tanto à parte de software quanto à parte dos dispositivos do sistema, sendo a integração entre ambos descrita em detalhes quando necessário.

1.1 Objetivos

Os objetivos desta dissertação foram delineados no início deste capítulo. De forma sucinta, são:

- Desenvolver uma arquitetura modular para a cadeia de processamento de imagens do sistema de monitoramento do desgaste de ferramentas de corte de forma que este, em sua essência, possa ser o mais independente possível de plataforma de execução e sistema operacional.
- Aprimorar o projeto geral e a documentação do sistema. Fazer também uma revisão bibliográfica mais aprofundada sobre a aplicação de sistemas de visão ao monitoramento do desgaste de ferramentas de corte.

- Desenvolver, junto com os parceiros alemães, um projeto modular para o sistema de iluminação, permitindo às equipes brasileira e alemã a implementação da mesma estratégia para o subsistema de iluminação.

- Testar algoritmos para a cadeia de processamento de imagens a fim de validar a arquitetura modular desenvolvida. Em particular, será testada a aplicabilidade de uma técnica baseada no método de análise dos componentes principais (bastante conhecido na literatura pela aplicação na técnica de *eigenfaces* [19]) ao problema da detecção de quebra e do desgaste de cratera de ferramentas de corte.

- Construir os programas do sistema TOOLSPY através da arquitetura desenvolvida, tanto no Brasil quanto na Alemanha.

1.2 Metodologia

Esta dissertação se baseia na seguinte metodologia para ter seus objetivos alcançados:

- Revisão da bibliografia acerca de sistemas de visão e da aplicação destes ao monitoramento do desgaste de ferramentas de corte para se chegar a um projeto geral do sistema que atenda a todos os seus requisitos.

- Projeto e implementação de software e do sistema utilizando-se a metodologia de engenharia de software e de sistemas e suas etapas clássicas [20, 21, 22, 23]:

  - Especificação dos requisitos: as funcionalidades básicas do sistema são levantadas e listadas.
  - Análise dos requisitos: os requisitos levantados e listados são analisados; prioridades, riscos e uma descrição mais detalhada de cada um deles é feita.
  - Modelagem do sistema: os diversos componentes do sistema são especificados e a integração entre estes é planejada. A linguagem de modelagem escolhida para esta finalidade é a UML (*Unified Modeling Language*), uma vez que o paradigma de programação a ser adotado é o de orientação a objetos e a linguagem UML é um padrão de fato neste caso [24, 25, 26].
  - Implementação do sistema: o sistema é construído. A linguagem de programação C++ [27, 28, 29, 30] será utilizada para esta finalidade, visto o seu caráter multi-paradigma (incluindo o paradigma de orientação a objetos) e a existência de compiladores padrão ANSI que suportam a compilação do código fonte gerado com esta linguagem em diferentes arquiteturas de computadores. Além disso, para a construção dos programas, diferentes bibliotecas de programação para interfaces gráficas serão utilizadas.
  - Testes do sistema: o sistema é testado de forma a se verificar o seu correto funcionamento.
  - Documentação do sistema: o sistema é documentado de forma a possibilitar futuros estudos e modificações.

- Revisão bibliográfica e análise empírica para o caso dos algoritmos a serem especificados na etapa da cadeia de processamento de imagens.

1.3 Resumo dos Capítulos Seguintes

O capítulo 1 tem por objetivo fornecer a justificativa desta dissertação de mestrado, mostrando a importância do problema tanto do ponto de vista econômico quanto do ponto de vista de pesquisa dentro do contexto dos processos de fabricação e das células autônomas de produção. Um panorama dos objetivos e metodologia do trabalho desenvolvido também foi apresentado.

Imediatamente a seguir, no capítulo 2 a problemática do monitoramento do estado de ferramentas de corte começa a ser tratada de forma sistemática. Os conceitos básicos relativos a processos de fabricação, ferramentas de corte e desgaste destas são apresentados. Este capítulo mostra também as diferentes alternativas possíveis para a solução deste problema na literatura e mostra porque uma solução geral é não trivial.

No capítulo 3 são fornecidos os fundamentos sobre sistemas de visão e processamento de imagens necessários para o entendimento do restante deste trabalho. Este capítulo parte de conceitos básicos de sistemas de visão e processamento de imagens, discute uma proposta de metodologia de projeto para sistemas de visão e finaliza analisando as diferentes propostas disponíveis na literatura para a medição do desgaste através de sistemas de visão.

O capítulo 4 inicia com a apresentação dos conceitos básicos sobre células autônomas de produção. Segue discutindo o projeto do sistema TOOLSPY de maneira geral através dos seus módulos e mostra também a API (*Application Programming Interface* – Interface para a Programação de Aplicações) implementada para o desenvolvimento de uma solução padrão utilizando-se sistemas de visão para o problema do desgaste de ferramentas de corte.

A cadeia de processamento de imagens do sistema TOOLSPY é descrita e analisada no capítulo 5, propondo-se e testando-se novos algoritmos e combinações destes para a obtenção de bons resultados para um tipo determinado de ferramenta. Neste capítulo, a biblioteca de processamento de sinais e software S2iLib, mantida e atualizada durante o desenvolvimento deste trabalho para a implementação da cadeia de processamento de imagens e visão computacional é apresentada.

Os resultados obtidos com a API e a cadeia de processamento propostas, além da validação destes sistemas, são apresentados no capítulo 6. Aqui são descritos de forma sucinta os sistemas construídos a partir da API, as versões alemã e brasileira do programa e o programa para a otimização dos parâmetros do sistema. Resultados obtidos com o sistema na medição de um determinado tipo de pastilha são apresentados.

Por fim, no capítulo 7, considerações são feitas sobre o trabalho, no sentido de analisar suas perspectivas e propor alternativas para a continuidade de seu desenvolvimento.

Capítulo 2

Monitoramento do Estado de Ferramentas de Corte

Neste capítulo serão discutidos alguns conceitos básicos sobre ferramentas de corte e desgaste de ferramentas, fornecendo-se assim o material necessário para o posterior entendimento do trabalho e aprofundando a justificativa de seu desenvolvimento.

2.1 Processos de Fabricação

Um processo de fabricação é definido como um conjunto de passos encadeados para a transformação de matérias-primas em produtos acabados, seguindo planos previamente estabelecidos e bem determinados [31, 32]. Os processos de fabricação envolvem a seleção de matéria-prima e outros materiais adequados e a determinação de um plano para sua execução, com base em requisitos técnicos e econômicos de forma a deixar seus produtos finais de acordo com as especificações necessárias e viáveis em termos de custo. Na atualidade, existe um grande número de processos de fabricação para diferentes aplicações industriais, sendo muito difícil e longa uma classificação geral destes. Uma forma simplificada de classificação dos processos de fabricação dentro da área metal-mecânica pode ser vista na figura 2.1, já indicando-se o foco de estudo deste trabalho.

Neste trabalho tem-se interesse geral em processos de usinagem, que são processos de separação de material da peça, material este com forma geométrica irregular (também chamado de cavaco), sob processamento com ferramentas de geometria definida [32]. Tem-se interesse particular no processo de fresamento, que é o processo de usinagem destinado à obtenção de superfícies quaisquer com a utilização de ferramentas em geral multicortantes (com múltiplas arestas de corte). Isto porque este foi o processo escolhido para um estudo aprofundado pelo projeto SFB368, como será discutido na seção 4.1.1.

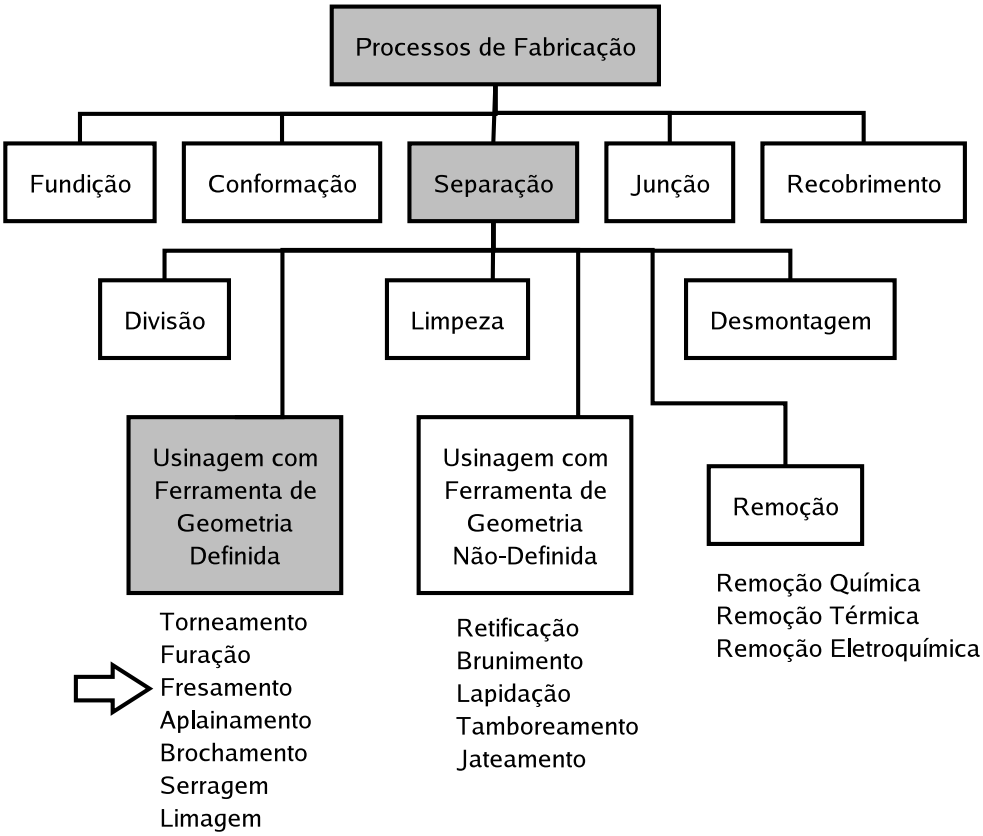


Figura 2.1: *Classificação dos processos de fabricação dentro da área metal-mecânica e o foco de estudo deste trabalho.*
Fonte: König e Klocke [32].

2.2 Ferramentas de Corte

Uma ferramenta de corte é o dispositivo responsável pela remoção de cavaco da peça no processo de usinagem através do contato com esta, de forma controlada e planejada [33]. As ferramentas de corte podem ser de diversas formas e diversos materiais. A figura 2.2 ilustra a terminologia associada a uma ferramenta de corte geralmente utilizada em processos de torneamento. Pastilhas montadas em hastes para a formação de uma ferramenta completa podem ser vistas na figura 2.3. Em processos de fresamento, essas pastilhas também são muito utilizadas quando montadas em um suporte que contém várias destas, caracterizando a sua função multi-cortante, como pode ser visto na figura 2.4.

A figura 2.5 ilustra os diferentes materiais usados nas ferramentas de corte, fornecendo uma idéia qualitativa de suas propriedades.

Ferramentas de metal-duro e metal-duro revestido são bastante utilizadas no meio industrial de forma geral para a usinagem de aço [33]. Elas possuem um custo unitário bastante reduzido. Ferramentas de CBN (Nitreto de Boro Cúbico – *Cubic Boron Nitrite*) e diamante são utilizadas em aplicações mais específicas. O custo unitário destas ferramentas é bastante mais elevado em comparação com o custo de ferramentas de metal-duro.

Para o sistema TOOLSPY, ferramentas de metal-duro simples e metal-duro revestido foram esco-

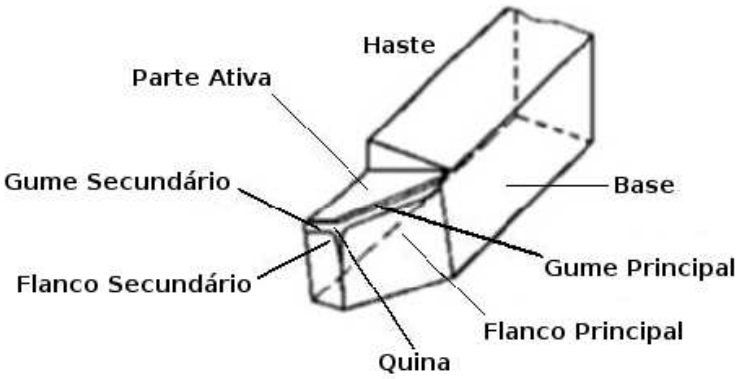


Figura 2.2: *Terminologia associada a uma ferramenta de corte.*
Fonte: König e Klocke [32].

Figura 2.3: *Exemplos de pastilhas para ferramentas de corte.*

lhidas como o foco de estudo inicial, uma vez de sua ampla utilização e do interesse econômico pelo fato de uma melhor utilização destas poder possibilitar uma economia de escala bastante grande, uma vez que são muito utilizadas. No futuro, ferramentas de CBN e diamante também serão estudadas.

2.3 Desgaste de Ferramentas de Corte

2.3.1 Formas de Desgaste

Existem três formas predominantes de desgaste que limitam a vida útil de uma ferramenta: lascamento, desgaste de flanco e desgaste de cratera [33]. O lascamento representa uma falha acidental

Figura 2.4: *Ferramentas multicortantes utilizadas no processo de fresamento. Várias pastilhas são montadas em um único suporte.*
Fonte: Sandvik-Coromant - http://www.sandvik.com.

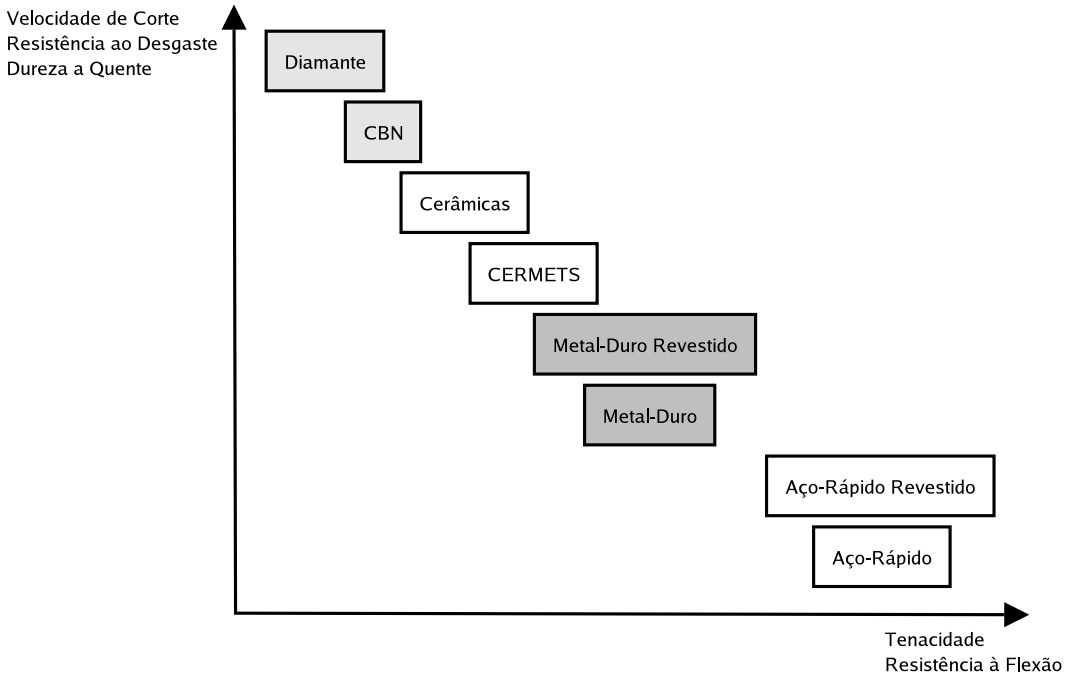


Figura 2.5: *Materiais de ferramentas e suas características.*
Fonte: König e Klocke [32].

e prematura da ferramenta devido a solicitações térmicas ou mecânicas excessivas em seu gume. O desgaste de flanco ocorre na face e no flanco principal da ferramenta e é atribuído ao atrito entre a ferramenta e a peça sendo usinada e às altas temperaturas envolvidas nos processos de usinagem. O desgaste de cratera se forma na face da ferramenta, em geral devido ao atrito da ferramenta com o cavaco sendo retirado da peça.

A figura 2.6 ilustra essas três principais formas de desgaste, juntamente com outras também encontradas.

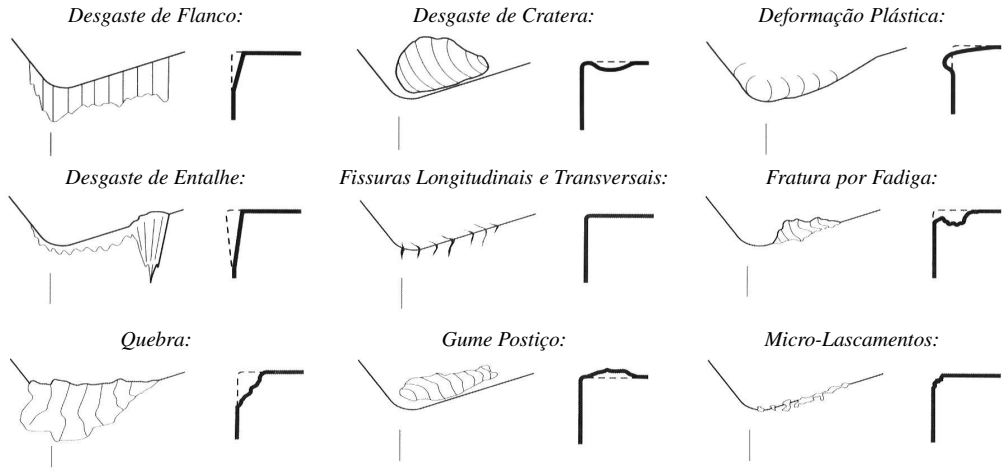


Figura 2.6: *Tipos de desgaste de ferramentas de corte.*
Fonte: König e Klocke [32].

No trabalho de Lanzetta [34] encontra-se uma boa caracterização das diferentes formas de des-

gaste a partir de normas técnicas internacionais e estudos da literatura, como pode ser visto na figura 2.7. Neste trabalho, o autor coloca que pouco esforço tem sido empregado no desenvolvimento de sistemas para a classificação das diferentes formas de desgaste a partir de visão computacional e que o desenvolvimento de um sistema deste tipo passaria por um intenso trabalho no levantamento das principais formas de desgaste encontradas em diferentes ferramentas com diferentes geometrias e de diferentes materiais.

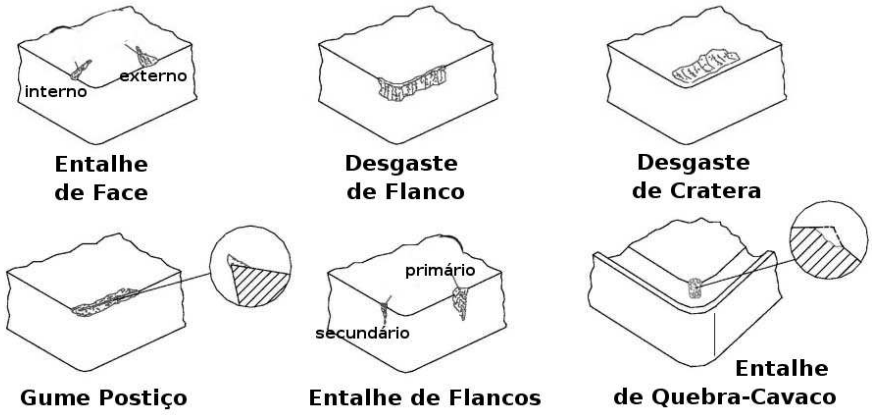


Figura 2.7: *Principais formas de desgaste na classificação de Lanzetta.*
Fonte: Lanzetta [34].

De particular interesse para este trabalho são o desgaste de flanco e o desgaste de cratera. Ambos tipos de desgaste podem ser vistos em maior detalhe na figura 2.8, com os principais parâmetros a serem medidos (desgaste de flanco máximo VB_{max} e desgaste de flanco VB - a 80% da área de desgaste, para maior explicação ver a seção 5.1.4) e os principais ângulos da ferramenta [33].

2.3.2 Principais Causas do Desgaste

Os principais fatores causadores de desgaste são, segundo König e Klocke [32] e Stemmer [33]:

Abrasão: partículas da ferramenta são arrancadas à alta pressão e temperatura devido ao deslocamento com atrito entre a ferramenta e a peça sendo usinada.

Aderência: é a principal causa de formação do gume postiço. Sob alta pressão e temperatura, partículas provenientes da peça ou do cavaco, se caldeiam ao gume da ferramenta de corte.

Difusão: as moléculas que formam a estrutura da ferramenta se movem, misturando-se e formando ligas menos resistentes e mais suscetíveis ao desgaste.

Oxidação: ocorre pela aceleração do processo de oxidação do aço que forma a ferramenta (no caso de ferramentas feitas de aço ou ferro) devido às altas temperaturas envolvidas no processo de usinagem.

Correntes elétricas iônicas: produzidas no contato entre peça e ferramenta durante a usinagem. Para uma melhor discussão sobre esse efeito, ver Stemmer [33].

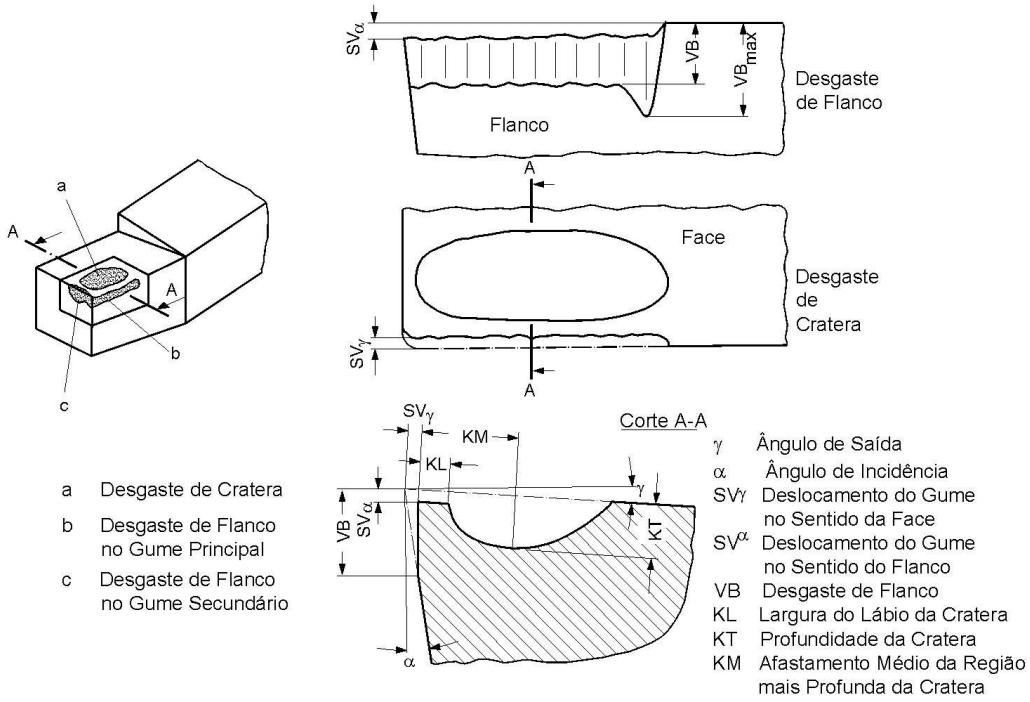


Figura 2.8: *Desgaste de flanco e de cratera, seus principais parâmetros e os principais ângulos da ferramenta.* Fonte: König e Klocke [32].

Choques mecânicos: a ferramenta entra em contato de forma brusca com alguma outra parte da máquina-ferramenta ou da peça. Isto pode acarretar um lascamento ou quebra da ferramenta, ou mesmo deformá-la.

Choques térmicos: a ferramenta sofre uma brusca variação de temperatura (em um curto intervalo de tempo), em geral por estar em uma alta temperatura e entrar em contato com uma peça de trabalho a uma temperatura muito mais baixa ou pelo uso de fluidos lubri-refrigentes de forma intermitente.

A figura 2.9, por exemplo, ilustra a influência da temperatura no desgaste da ferramenta de corte.

O controle destas causas de desgaste pode, muitas vezes, ser contornado pela seleção correta das condições de usinagem, principalmente do avanço e da profundidade de corte e também por outros fatores do processo de usinagem, como o uso de ferramentas revestidas e as características deste revestimento e a seleção do tipo do material da ferramenta a ser usada.

2.4 Métodos de Monitoramento do Estado de Ferramentas de Corte

O monitoramento do estado de ferramentas de corte é uma parte importante de um sistema de manufatura. A partir de um sistema de monitoramento do estado de ferramentas, por exemplo, a quebra da ferramenta pode ser detectada a tempo de reduzir os impactos econômicos acarretados e a troca pode ser providenciada antes do seu estado excessivamente desgastado acarretar a perda da qualidade dos produtos manufaturados.

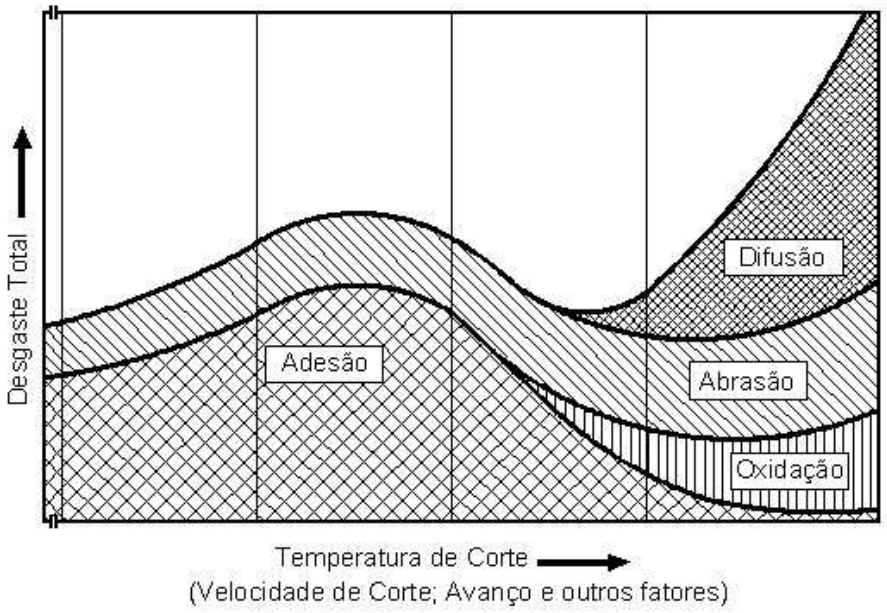


Figura 2.9: *Influência das condições de corte nas causas do degaste.*
Fonte: König e Klocke [32].

Em uma célula autônoma de produção, como será descrito na seção 4.1.1, um sistema de monitoramento do estado de ferramentas é de particular importância, uma vez que a autonomia do sistema só pode ser alcançada a partir do momento em que este possa tomar de forma independente decisões acerca da utilidade ou não de determinada ferramenta.

A literatura classifica os métodos de monitoramento do estado de ferramentas de corte em dois grandes grupos [3]: os métodos diretos, discutidos na subseção 2.4.1, e os métodos indiretos, discutidos na seção 2.4.2. Além desta classificação, os métodos de monitoramento também podem ser classificados em métodos intermitentes[1], quando é necessário que o processo seja parado para que o estado da ferramenta seja monitorado, e métodos contínuos[2], quando o estado da ferramenta pode ser monitorado durante a execução do processo.

Sensores para o monitoramento do estado de ferramentas de corte devem satisfazer os seguintes requisitos [3]:

- Devem realizar a medição o mais próximo possível da usinagem, para realizar medições que reflitam o estado geral do sistema.
- Não devem influenciar a operação da máquina-ferramenta.
- Não devem restringir o espaço de trabalho ou influenciar os parâmetros de usinagem.
- Devem permitir sua fácil manutenção e troca, com baixos custos.

[1] Também chamados de métodos *off-line* ou *in-cycle*.
[2] Também chamados de métodos *on-line* ou *in-process*.

- Devem ser resistentes às condições do ambiente (choque, campos magnéticos, altas temperaturas, entre outros).
- Devem funcionar independentemente de ferramenta ou peça.
- Devem possuir características metrológicas adequadas.
- Devem transmitir o sinal de maneira confiável.

A figura 2.10 fornece um esquema dos métodos diretos e indiretos existentes, indicando os seus principais tipos [1, 3, 5].

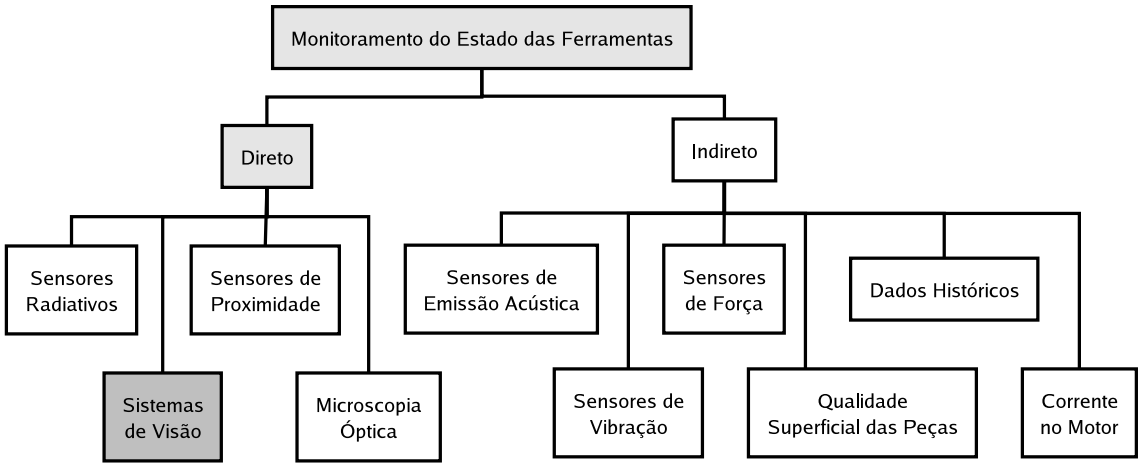


Figura 2.10: *Principais métodos diretos e indiretos existentes.*

2.4.1 Métodos Diretos

Os métodos diretos de monitoramento do desgaste envolvem a determinação do estado da ferramenta através da análise de características e parâmetros provenientes da própria ferramenta utilizada no processo, fornecendo desta forma uma informação bastante precisa de sua condição [3]. Muitos desses métodos são também classificados de métodos *off-line*, pelo fato de necessitarem que a ferramenta não esteja sendo usada no processo, o que muitas vezes implica na parada deste para que aquela seja analisada.

A seguir, são apresentadas as técnicas de monitoramento direto mais utilizadas: os sensores de proximidade, os sensores radioativos, os sensores de visão e a medição através de microscopia óptica.

2.4.1.1 Sensores de Proximidade

Os sensores de proximidade são uma das formas mais antigas para se estimar o desgaste da ferramenta [5]. Esta abordagem se baseia na medição da modificação da distância entre o gume principal (ver figura 2.2) e a peça. Esta distância pode ser medida através de micrômetros elétricos ou provas de toque pneumáticas. Este tipo de medição é afetado pela expansão térmica da ferramenta, pela vibração da peça e da ferramenta devido às forças de corte.

2.4.1.2 Sensores Radioativos

Nos sensores radioativos, uma pequena quantidade de material radioativo é colocada nas faces principal e secundária da ferramenta até uma determinada profundidade e de forma homogênea. Durante o processo de corte, parte do material da ferramenta é removido desta junto com o cavaco formado. Pela monitoração da quantidade de material radioativo presente no cavaco o desgaste de ferramenta pode ser estimado [5].

A necessidade de se coletar o cavaco para posterior análise e o perigo de se trabalhar com materiais radioativos limitam o uso desta técnica a ambientes de laboratório apenas.

2.4.1.3 Sensores de Visão

A aplicação direta de sensores de visão usa a própria ferramenta. Em geral, estes sensores dependem das propriedades reflexivas da área de desgaste, geralmente maiores do que na área não desgastada para a derivação de uma série de parâmetros que caracterizem o desgaste. A maioria das pesquisas nesta área limitou-se na medição apenas do desgaste de flanco [8, 35, 36, 37, 38, 39, 40, 41, 42], enquanto alguns pesquisadores tentaram medir tanto o desgaste de flanco quanto o desgaste de cratera [43, 44, 45]. O desgaste de flanco pode ser avaliado através de uma câmera com sensor CCD padrão. No entanto, o desgaste de cratera necessita de um outro tipo de abordagem para derivar a informação de profundidade, como, por exemplo, a utilização de iluminação estruturada.

Devido às condições hostis dos processos de fabricação, por exemplo pela presença de fluidos lubri-refrigerantes e a geração de cavaco, a utilização de sensores de visão está limitada apenas ao uso intermitente.

A medição do desgaste de ferramentas através de sistemas de visão será detalhada na seção 3.4.

2.4.1.4 Medição Através de Microscópio

A medição através de microscópio é realizada por um operador da máquina-ferramenta ou técnico habilitado para tal finalidade [33]. O processo de fabricação é parado, a ferramenta é removida da máquina e levada até uma estação de medição, em geral afastada do chão-de-fábrica para que o desgaste possa ser corretamente avaliado através da visualização da área de desgaste na ferramenta.

Este método é manual e a avaliação do estado da ferramenta e mesmo a medição da área do desgaste dependem da avaliação subjetiva do operador ou técnico responsável pela medição. Diferentes operadores podem avaliar o desgaste de diferentes formas, uma vez que as próprias características do processo de fabricação podem determinar se a ferramenta pode continuar a ser utilizada ou não.

2.4.2 Métodos Indiretos

Os métodos indiretos são aqueles em que alguma grandeza física relativa ao processo de fabricação ou à máquina-ferramenta é medida, sendo que a condição da ferramenta é relacionada a esta

grandeza [4]. Essas grandezas são influenciadas, muitas vezes, por outros fatores além do desgaste da ferramenta, o que pode levar a previsões erradas se a influência destes fatores for grande. Estes fatores são considerados no modelo proposto por Du et al. [1], como pode ser visto na figura 2.11.

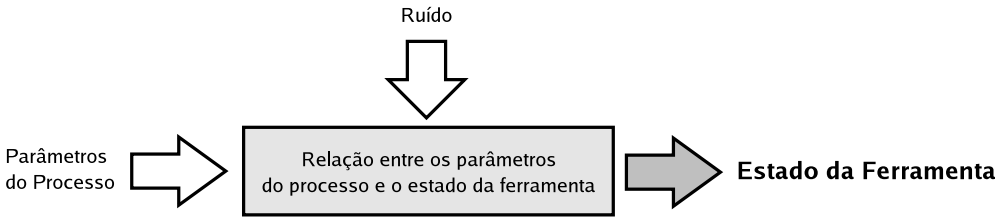


Figura 2.11: *Modelo geral para o monitoramento indireto de processos de fabricação.*

A abordagem baseada em métodos indiretos pode ser implementada durante a execução do processo pelo monitoramento de diversas grandezas relativas a este, em especial quando as grandezas monitoradas são, por exemplo, forças de corte, vibrações e emissão acústica. Os sinais podem ser analisados, então, de diferentes formas, através de técnicas tradicionais ou técnicas de inteligência artificial, como redes neurais artificiais [46].

2.4.2.1 Sensores de Emissão Acústica

Emissão acústica é definida como uma energia elástica transiente espontaneamente liberada por materiais sendo deformados, em processo de fratura ou ambos [47, 48]. Este sinal é geralmente detectado e capturado pela instalação de um transdutor piezelétrico fixado na máquina-ferramenta. A informação do sinal acústico deve ser cuidadosamente analisada para que se separe o sinal relativo ao processo de corte de outros sinais acústicos presentes no ambiente. Isto requer, além do sensor, amplificadores de sinal, filtros e uma eletrônica para pré-processamento. Além disso, a localização do sensor na máquina-ferramenta é bastante problemática: diferentes máquinas possuem diferentes características que devem ser consideradas para o correto posicionamento dos sensores de emissão acústica.

2.4.2.2 Sensores de Vibração

A usinagem com uma ferramenta já desgastada aumenta as flutuações nas forças de corte sobre a ferramenta envolvida no processo [49, 50]. Isto é atribuído pela fricção entre a peça e a ferramenta e também pela estrutura interna da ferramenta, já com algumas fraturas. Devido a essas flutuações, vibrações ocorrem no sistema. Assim, o monitoramento do nível de vibrações pode fornecer informações sobre a condição da ferramenta.

O sensor, neste caso, consiste em um acelerômetro piezelétrico fixado sobre a face da ferramenta, tão perto quanto possível do gume de corte. A saída do sensor é comparada a um valor de referência que, se repetidamente excedido, mostra uma condição de ferramenta bastante desgastada. O posicionamento do sensor é bastante importante neste caso: caso colocado bastante próximo da aresta

de corte, a dispersão do sinal aumenta com a progressão do processo de corte. Caso colocado mais distante da aresta de corte, a amplitude do sinal é reduzida.

2.4.2.3 Sensores de Força

O monitoramento através das forças de corte envolvidas nos processos de fabricação é bastante usado uma vez que a tecnologia de tais sensores está bastante difundida [51, 52, 53]. Em geral, estes sensores são formados por um dinamômetro montado no dispositivo de fixação da ferramenta para monitorar as forças de corte em 1 ou 2 direções ortogonais. O sinal do sensor de força indica o aumento na força de corte para a usinagem à medida que a ferramenta se desgasta. Outras alternativas de sensores também são possíveis, como o uso de discos de medição de forças (através de medidores de força piezo-elétricos ou extensômetros), o uso de extensômetros diretamente no fuso da máquina-ferramenta e outros [3].

A análise do sinal de força fornecido por esse tipo de sensor deve ser feita para que se determine o momento de troca da ferramenta. Esta análise se torna em geral complexa à medida que outros parâmetros também influenciam no sinal do sensor de força. Por exemplo, as propriedades de determinados materiais (como densidade, ductilidade e dureza), a geometria da ferramenta de corte, o efeito da saída do cavaco ou a existência de quebra-cavaco. Todos estes fatores contribuem para a complexidade do desenvolvimento de um sensor de força robusto e confiável para o monitoramento do desgaste de ferramenta.

2.4.2.4 Monitoramento Baseado na Medição da Corrente no Motor e na Potência Efetiva

Sensores para a medição da corrente dos motores e da potência efetiva dos acionamentos da máquina-ferramenta representam a mais simples alternativa do ponto de vista técnico, podendo ser facilmente modernizados [3, 54]. Dependendo do tipo do motor, existe uma grande variedade de opções disponíveis. Essa estratégia, entretanto, possui algumas desvantagens, uma vez que é fortemente dependente da fricção nas guias da máquina e do estado de lubrificação destas, sendo as informações relevantes no sinal mascaradas por esses fatores e de difícil detecção na maioria dos casos.

2.4.2.5 Inspeção da Qualidade Superficial das Peças

Um outro método indireto de medição do desgaste de ferramentas é através da inspeção da qualidade superficial das peças usinadas com determinada ferramenta [3, 55]. A rugosidade e a textura das superfícies usinadas são avaliadas de forma a se verificar se estão em um patamar aceitável, de acordo com o processo sendo executado. Normalmente, quanto pior esses parâmetros, pior o estado da ferramenta, de forma que é possível prever o momento de sua troca. Dois são os principais tipos de sensores usados nesta abordagem: sensores ultra-sônicos e sensores ópticos, que podem ser de dois tipos, dispersão luminosa ou feixe de laser.

Tönshoff et al. [56] propõem uma abordagem contínua bastante inovadora para o monitoramento do processo de fabricação e do estado da ferramenta através da análise de doze parâmetros micromagnéticos da peça sendo usinada por um sensor montado no suporte da ferramenta. Tensão residual e dureza superficial da peça são medidos através dessa técnica e usados tanto para a inspeção da qualidade da peça quanto para o monitoramento do processo de fabricação.

2.4.2.6 Estimativa a Partir de Dados Históricos

A estimativa da vida da ferramenta através de dados históricos não é propriamente um método de monitoramento, mas é uma forma bastante utilizada na prática de se determinar o momento de troca da ferramenta em um processo de fabricação por técnicas de controle estatístico do processo (CEP). Ensaios com um determinado tipo de ferramenta em um determinado tipo de operação são realizados de forma a se determinar o estado médio da ferramenta nos diferentes estágios da usinagem. A partir dessas informações, pode-se determinar o momento da troca da ferramenta.

Em geral, nesses métodos a ferramenta é subutilizada, trocando-se esta muito antes do necessário, de forma a se evitar sua quebra.

2.5 Considerações acerca do Monitoramento do Estado de Ferramentas de Corte

A maioria das pesquisas na área de monitoramento do estado das ferramentas de corte foi realizada em ambientes controlados, em máquinas-ferramenta de propósito geral, como fresas verticais, em laboratórios de pesquisa e não em máquinas utilizadas para produção industrial [9]. Além disso, os sensores usados para pesquisa nessas máquinas não são robustos o suficiente para aplicações industriais. Isto é, com certeza, um fator limitante à aplicação dessas técnicas. Contudo, produtos comerciais para o monitoramento dos processos de fabricação têm sido colocados no mercado [6] e a sua robustez vem aumentando gradativamente com o progresso das pesquisas na área.

Para aumentar a robustez da solução através de um método indireto, múltiplos sensores são utilizados, como, por exemplo, o monitoramento através de emissão acústica combinado com o monitoramento através de vibrações da máquina-ferramenta ou com o monitoramento através das forças de corte. A esta abordagem através da utilização de múltiplos sensores dá-se o nome de fusão de sensores [57, 58, 59, 60]. A figura 2.12 ilustra a utilização desta técnica juntamente com um sistema inteligente baseado em técnicas de inteligência artificial que se comunica com o controle do processo ou da planta, fornecendo informações acerca do estado da máquina, da ferramenta ou da peça. A partir destas informações, o controle pode tomar determinado curso de ação.

O uso de métodos indiretos de monitoramento possui a vantagem de poder ser usado de forma contínua durante o processo de fabricação. Contudo, como já foi mencionado na seção 2.4.2 e como pode ser visto na figura 2.11, esta abordagem possui a desvantagem de ruídos poderem interferir no

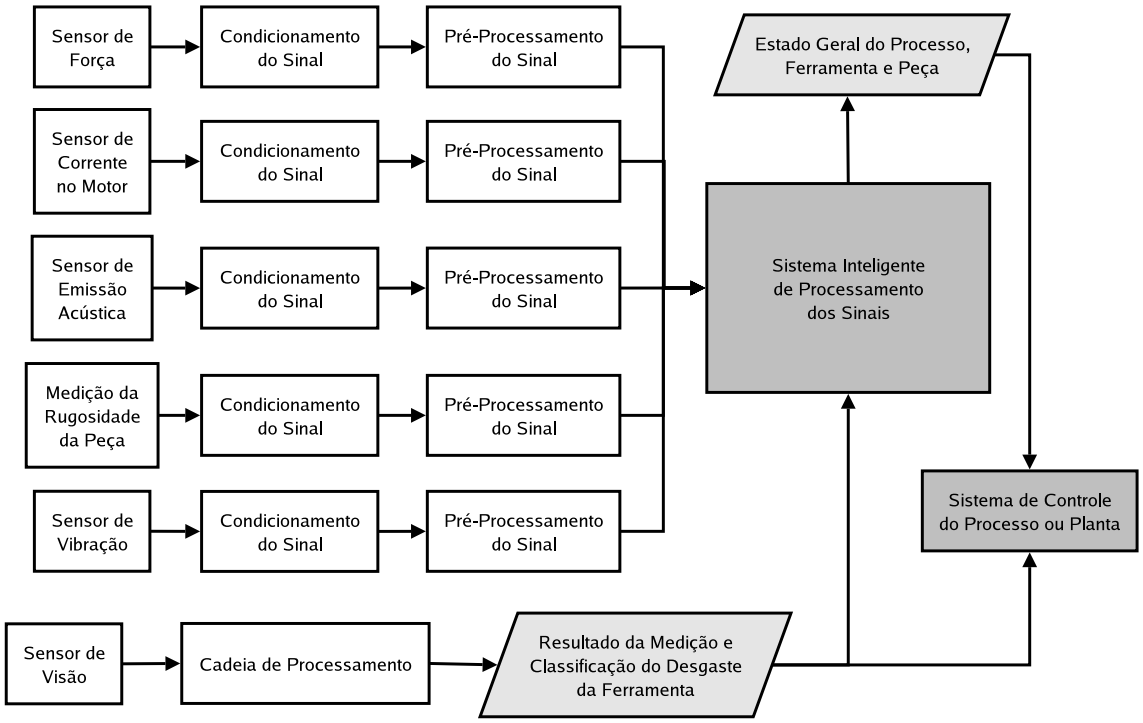


Figura 2.12: *Modelo de uma técnica de fusão de múltiplos sensores usando um sistema de processamento inteligente.*

sinal captado, sendo que a distinção entre o ruído e o sinal de interesse para o sistema devem ser tratados de alguma forma através do modelo utilizado. Isto pode ser, em muitos casos, não trivial. Além disso, os métodos indiretos não fornecem uma medida precisa do desgaste da ferramenta; fornecem apenas informações sobre o estado desta. Por fim, é necessário que o sistema seja calibrado de acordo com a característica dos sinais de cada máquina e de cada processo de fabricação para poder gerar resultados com algum significado [46].

Já em um método direto, a medição é efetuada diretamente a partir da própria ferramenta, o objeto do qual se quer obter as informações acerca do desgaste. A medida fornecida é precisa, como no caso de um sistema de visão, na forma de parâmetros como uma área de desgaste de flanco, do desgaste de flanco máximo e do desgaste de flanco médio, por exemplo, e a incerteza da medição pode ser determinada através da calibração do sistema utilizado. A utilização de um sensor baseado em técnicas de processamento de imagens e visão computacional aumenta a autonomia do sistema, sendo que uma estação para a medição pode ser montada próxima à máquina-ferramenta. A desvantagem deste método é a necessidade de medição intermitente e não contínua, sendo que a ferramenta deve ser removida da máquina-ferramenta e levada a uma estação de medição apropriada. Contudo, com a utilização de máquinas-ferramentas com magazines de ferramentas com uma grande capacidade, com ferramentas às vezes duplicadas, este problema pode ser minimizado.

Capítulo 3

Considerações sobre o Projeto de Sistemas de Visão

Este capítulo é dividido em três partes. Inicialmente, um pouco da terminologia e dos conceitos relativos à área de sistemas de visão serão apresentados juntamente com uma descrição de seus componentes. Posteriormente, uma metodologia para o projeto de sistemas de visão será detalhada. A apresentação de diferentes aplicações de sistemas de visão ao problema do monitoramento do desgaste de ferramentas de corte finaliza o capítulo.

3.1 Terminologia

A definição exata do termo sistemas de visão deve passar por uma análise dos diferentes termos existentes relativos a esta área na literatura especializada. Os termos mais comumente encontrados são *sistemas de visão*, *visão computacional*, *processamento de imagens* e *análise de imagens*. Será adotada, no decorrer deste trabalho, a seguinte terminologia, descrita por Gonzalez e Woods [61]:

Processamento de imagens: se refere ao processamento digital de imagens através de um computador ou outro dispositivo eletrônico no qual tanto as entradas quanto os resultados gerados pelo processamento são imagens. Operações de processamento de imagens são enquadradas como operações de baixo e médio nível, sem um conteúdo semântico associado, como o resultado de um processamento. Como exemplos de operações de processamento de imagens pode-se citar algoritmos de filtragem e algoritmos simples de segmentação de imagens, por exemplo.

Visão Computacional: refere-se ao processamento digital de imagens para a extração de algum tipo de informação relevante, como no caso da identificação de algum objeto da imagem. Envolve operações de mais alto nível, como a classificação de objetos através de uma rede neural artificial.

Análise de Imagens: comporta a área de abrangência de visão computacional e as operações de médio nível de processamento de imagens, como no caso de operações de segmentação que, em vez de retornarem imagens, retornam contornos.

Sistemas de Visão: englobam um sistema completo, como uma solução para um problema, composto tanto por dispositivos físicos quanto por componentes de software. Um sistema de visão realiza tarefas desde a aquisição das imagens até o seu processamento através de software apropriado e a análise do resultado, com conseqüente tomada de decisão, sendo composto por dispositivos de iluminação, ópticos e de posicionamento específicos para estas tarefas e software tipicamente feito sob demanda. Este processamento pode ser um simples processamento de imagens ou alguma técnica de visão computacional.

Na literatura estrangeira encontram-se ainda o termo *machine vision*, empregado quase como um sinônimo de sistemas de visão, referindo-se à incorporação de um comportamento de visão inteligente, capaz de realizar reconhecimento e classificação, por exemplo, similares ao humano, a máquinas e dispositivos variados, como robôs, através de técnicas de visão computacional. A figura 3.1 ilustra a caracterização de todos estes termos.

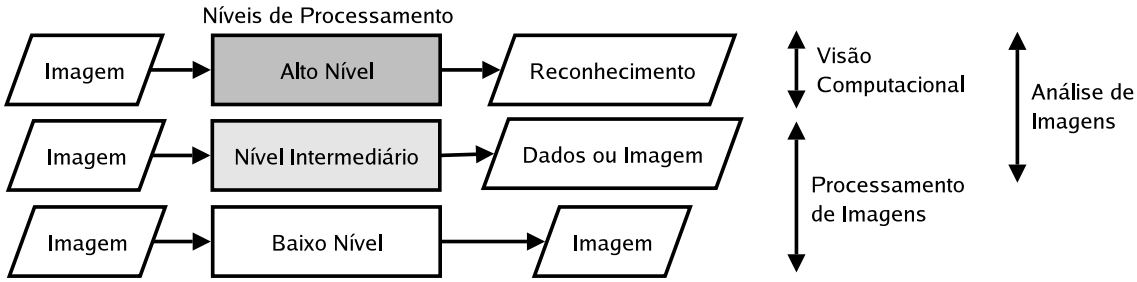


Figura 3.1: *Esquema da terminologia da área de sistemas de visão.*

Na seqüência, os componentes de um sistema de visão serão brevemente discutidos.

3.2 Componentes

Existem diferentes abordagens para a questão da definição dos componentes de um sistema de visão. Um modelo bastante completo é o fornecido por Gonzalez e Woods [61], de acordo com a figura 3.2.

Neste modelo, o computador é a parte central do funcionamento do sistema de visão, sendo responsável pelo controle da execução das diferentes tarefas do sistema. Ele pode ser um computador do tipo PC comum ou um computador próprio para tarefas de processamento de imagens e visão computacional em aplicações que necessitem de um poder de processamento maior ou dedicado, como, por exemplo, a rápida identificação de peças ou objetos transportados através de uma esteira de uma linha de produção a uma alta velocidade ou o controle de tráfego urbano em uma ponte com diversas pistas, incluindo neste caso o reconhecimento das placas dos veículos que por ali trafegam.

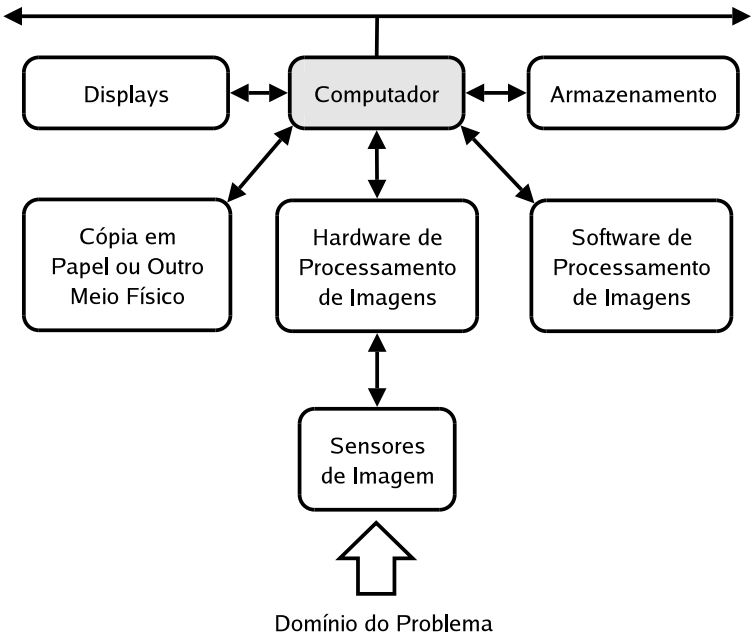


Figura 3.2: *Modelo geral para os componentes de um sistema de visão.*

O computador controla o armazenamento das informações manipuladas pelo sistema de visão, sejam elas imagens ou resultados de processamento. Este armazenamento pode se dar tanto em disco quanto em cópias físicas (por exemplo, papel). Além disso, o computador também é responsável por mostrar os resultados do processamento nos dispositivos adequados como monitores de vídeos, *displays* de diferentes tamanhos (como *displays* LCD) e, em alguns casos, luzes e outros dispositivos de alarme para indicar o funcionamento incorreto de algumas de suas partes. O computador também é o suporte para o software do sistema, devendo se comunicar com os dispositivos próprios para a captura de imagens, por exemplo, que recebem os sinais dos sensores do ambiente externo em diversos formatos.

Jähne [62] agrupa de uma outra forma os componentes essenciais para um sistema de visão, classificando-os da seguinte maneira:

Fonte de iluminação: iluminação apropriada é necessária para que os objetos de interesse possam ser estudados, uma vez que estes geralmente não emitem luz.

Câmera: esta parte é formada por componentes ópticos, responsáveis por transformar a radiação luminosa proveniente do objeto em um sinal adequado para o sensor.

Sensor: é o transdutor do sinal em sua forma de radiação luminosa para um sinal elétrico que possa ser posteriormente transmitido pela câmera e entendido pelo sistema de processamento, geralmente em um formato analógico.

Sistema de processamento: é o sistema que processa o sinal proveniente do sensor e transmitido pela câmera, de forma a extrair as informações necessárias para a análise e tomada de decisões. Pode ser composto também por diferentes dispositivos como, por exemplo, placas de aquisição, quando da necessidade de tradução do sinal proveniente da câmera para um formato digital.

Atuadores: os atuadores são aqueles que reagem ao resultado do processamento. Eles se tornam parte integrante do sistema de visão quando este está ativamente respondendo ao que é observado, como é o caso da navegação autônoma de robôs móveis em um ambiente com obstáculos.

De acordo com esta abordagem, um sistema de visão deve ser autocontido, isto é, deve ser constituído pela sua própria fonte de iluminação e possuir componentes ópticos adequados para que o sinal luminoso do objeto sob estudo possa chegar ao sensor contido na câmera e ser traduzido em um sinal elétrico, sendo transmitido então pela câmera para o sistema de processamento a fim de que seja manipulado de forma adequada, tendo sua informação analisada e por sua vez fornecida aos dispositivos ou pessoas que necessitam desta informação para a tomada de decisão.

Neste trabalho, os sistemas de visão serão abordados de forma híbrida entre as abordagens de Gonzalez e Woods [61] e Jähne [62], sendo que alguns outros elementos relevantes serão acrescentados. Desta forma, um sistema de visão será considerado como constituído pelos seguintes componentes: subsistema óptico, subsistema de iluminação, sensores e câmeras, transmissão de dados, subsistema de processamento e interfaces de suporte, como ilustrado na figura 3.3.

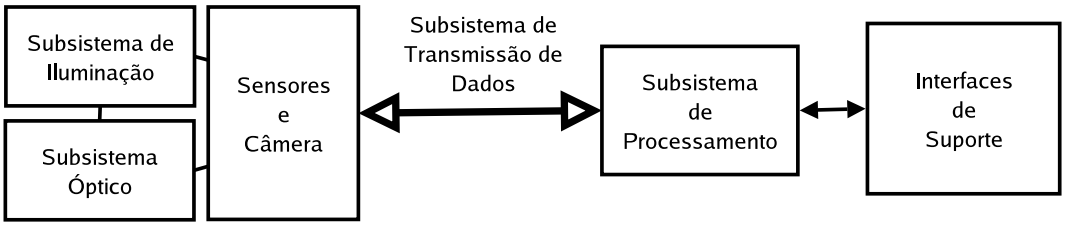


Figura 3.3: *Modelo adotado para os componentes de um sistema de visão.*

3.2.1 Subsistema Óptico

O subsistema óptico de um sistema de visão é formado por prismas, filtros, espelhos, lentes, tubos extensores e outros componentes ópticos que, em conjunto, são utilizados para formar uma imagem do objeto sob estudo sobre o sensor com o objetivo de ressaltar as características que serão analisadas [62]. Tais sistemas são indispensáveis na obtenção de imagens com alta definição e homogêneas em relação à variabilidade de luz no ambiente. A qualidade dos componentes do subsistema óptico é indispensável para que uma imagem adequada do objeto sob estudo seja adquirida.

Os filtros são comumente utilizados para selecionar uma determinada freqüência luminosa que incide sobre o sensor da câmera. Prismas e espelhos são utilizados para manipular a trajetória da luz que incide sobre o sensor da câmera, permitindo que áreas não diretamente acessíveis por este sejam observadas. As lentes são utilizadas para se obter o aumento (ou redução) necessário da região a ser observada, sem perda de foco. Mais informações sobre dispositivos ópticos, seu funcionamento e propriedades podem ser vistas no livro de Smith [63]. Uma discussão geral sobre óptica e formação de imagens pode ser nos vista artigos de Geissler [64] e Haussecker [65, 66, 67].

3.2.2 Subsistema de Iluminação

O objetivo da iluminação num sistema de visão é ressaltar as características de interesse dos objetos sob estudo e atenuar partes da imagem que não interessam à aplicação [62]. Além da iluminação, diversos fatores afetam a quantidade de luz que incide sobre o sensor da câmera, como a abertura da lente, a utilização de tubos extensores, a configuração da câmera, o tempo de exposição, o uso de filtros, entre outros. Todos estes parâmetros precisam ser ajustados e testados em conjunto para se certificar que o projeto está de acordo com as especificações.

Existem diferentes técnicas de iluminação que devem ser consideradas quando do projeto de um sistema de visão. Em geral, a consideração acerca das técnicas de iluminação a serem utilizadas se dá junto com a especificação do subsistema óptico do mesmo. Para uma maior discussão sobre essas técnicas, recomenda-se o artigo de Haussecker [68].

3.2.3 Sensores e Câmeras

A luz emitida pelo subsistema de iluminação, refletida pelo objeto sob observação e que passou através do subsistema óptico de forma a ser adequadamente tratada para destacar alguma característica do objeto, chega ao sensor onde é captada [62]. Os sensores podem ser de diferentes tipos [69]: CCD (*Charged Coupled Devices*), CMOS (*Complementary Metal Oxide Semiconductor*), coloridos de um *chip* (que pode ser CCD ou CMOS), coloridos com múltiplos *chips*, entre outros que fogem ao escopo deste trabalho [70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80].

Considera-se normalmente os sensores estando contidos em uma câmera, que é a junção destes com um um dispositivo óptico, como uma lente, e uma parte do subsistema de transmissão de dados, como será visto a seguir. Existem diversos tipos de câmeras que diferem, além do tipo de sensor, na forma de varredura do sensor para a transmissão dos dados, que pode ser linear, entrelaçada ou não entrelaçada [69]. Câmeras analógicas, digitais, *webcams*, câmeras inteligentes e outros tipos de sensores, como tomógrafos e sensores térmicos são alguns destes tipos.

3.2.4 Transmissão de Dados

Pode-se considerar que uma imagem digital é, na verdade, uma matriz, onde cada elemento possui um valor de nível de cinza ou um conjunto de valores de um formato pré-estabelecido que determinam sua cor [69]. O modo de transmissão dessa matriz para o computador na forma de sinais elétricos é de importância fundamental para o sistema de visão, uma vez que estes sinais limitam o desempenho de todo o sistema, como a taxa máxima de imagens por segundo que pode ser adquirida e, conseqüentemente, processada.

Ressalta-se aqui que se deve tomar muito cuidado ao se projetar um sistema de visão, considerando-se o desempenho dos sinais de transmissão da imagem. Estes sinais possuem limitações quanto à velocidade de transmissão, à resolução da imagem transmitida, à sincronização entre computador e

câmera e à qualidade da imagem. Um sinal de vídeo mal selecionado para uma aplicação crítica, por exemplo, pode inviabilizar todo projeto.

Os modos de transmissão são divididos em analógicos e digitais [69]. Os principais modos analógicos são EIA (RS-170), NTSC, PAL, SECAM, e vídeo composto (Y-C ou luminância e cor). Os principais modos digitais são TTL, RS-422, RS-644 (LVDS ou *Low Voltage Differential Signal*, IEEE 1394, USB, *Camera Link*, *HOTLink*.

3.2.5 Subsistema de Processamento

O subsistema de processamento é composto pelo dispositivo de aquisição do sinal de vídeo transmitido pelo conjunto sensor e câmera (que pode ser um dispositivo de digitalização do sinal de vídeo, quando em um modo de transmissão analógico ou um dispositivo de aquisição digital para a disponibilização do vídeo digital transmitido quando em um modo de transmissão digital), pelo driver deste dispositivo (ou driver do conjunto sensor e câmera quando este englobar um protocolo digital de mais alto nível, não necessitando do dispositivo de aquisição, como é o caso das *webcams* e câmeras que suportam o protocolo IEEE 1394), por bibliotecas de processamento de imagens e visão computacional e pela aplicação desenvolvida, que utiliza as bibliotecas anteriores, juntamente com a informação captada pelo sensor através do driver do dispositivo de aquisição para o processamento da informação para determinada finalidade [62].

O dispositivo de aquisição do sinal de vídeo é uma placa de aquisição de imagens que transforma o sinal de vídeo analógico ou digital recebido em uma informação capaz de ser tratada por um computador digital. Papel importante nesta etapa de tradução é desempenhado pelo driver desse dispositivo, que irá servir como a interface de comunicação da aplicação ou das bibliotecas de processamento de imagens com os sensores do sistema de visão [61]. É a partir desta informação, captada pelo sensor da câmera e colocada em um formato adequado pela placa de aquisição de imagens em conjunto com seus drivers, que as bibliotecas de processamento de imagens irão atuar de uma forma ordenada e previamente estipulada, de acordo com a implementação da aplicação, a fim de gerar seus resultados.

As bibliotecas de processamento de imagens precisam implementar uma série de funções para que as imagens adquiridas sejam adequadamente tratadas [61, 81, 82, 83]. As funções básicas de uma biblioteca de processamento de imagens são a representação das imagens e a sua manipulação através de operadores de rotação, seleção de regiões, *threshold* e normalização, entre outros. Também outras operações devem ser possíveis (aqui estão algumas das operações usadas no desenvolvimento do trabalho com sua referência para consulta dos algoritmos):

- Transformada de Fourier [84]
- Segmentação de imagens de diversos modos, como pirâmides [61] e contornos ativos (*snakes*) [61, 82].
- Filtragem de imagens através de filtros como mínimo, máximo, médio, gaussiano e laplaciano, com uma abertura do filtro adequada [85, 86, 87, 88].

- Interpolação de pontos [89].
- Análise de textura [90] e análise multiresolução [85], como a Transformada *Wavelet* [61].
- Operadores de morfologia matemática para análise da imagem a partir de imagens binarizadas [61, 91]

3.2.6 Interfaces de Suporte

Nas interfaces de suporte do sistema de visão estão incluídos todos os dispositivos que são fundamentais para que o sistema opere de maneira correta, como as interfaces com sistemas de controle de mais alto nível (por exemplo, o sistema de controle supervisório de uma planta industrial), com sistemas de bancos de dados para o registro de eventos de operação ou os resultados do processamento, com sistemas pneumáticos e hidráulicos para o rejeito de determinadas peças ou para o acionamento de outros dispositivos importantes na cadeia de tomada de decisão [61, 62].

Destacam-se aqui as interfaces com sensores de presença indutivos, capacitivos ou ópticos, que auxiliam bastante no processo de aquisição automática de imagens para o processamento.

3.3 Metodologia de Projeto

O projeto de um sistema de visão para uma determinada aplicação é uma tarefa bastante complexa, que envolve a determinação de uma série de parâmetros e o projeto, especificação, seleção e integração de equipamentos e algoritmos para possibilitar a construção do sistema final [61]. A figura 3.4 ilustra a proposta de uma metodologia para o projeto de sistemas de visão [92].

Inicialmente, é necessário a determinação de alguns parâmetros do sistema de visão relativos à área que se deseja visualizar. Estes parâmetros estão ilustrados na figura 3.5 e são:

Campo de visão (*Field of View* – FOV): é o tamanho da área que se deseja visualizar, ou seja, que deve ser captada pelo sensor.

Distância de trabalho (*Working Distance* – WD): é a distância da porção posterior da lente até a área a ser inspecionada. Pode ser especificada como uma faixa de valores, como um valor mínimo e um valor máximo.

Resolução (*Resolution* – R): é o tamanho mínimo do objeto que pode ser distinguido pelo sistema de visão.

Profundidade de campo (*Depth of Field* – DOF): é a máxima profundidade no campo de visão que pode ser mantida inteiramente em foco, a partir da distância de trabalho, ou a distância permitida que o objeto se movimente a partir da distância de trabalho, mantendo-se ainda em foco.

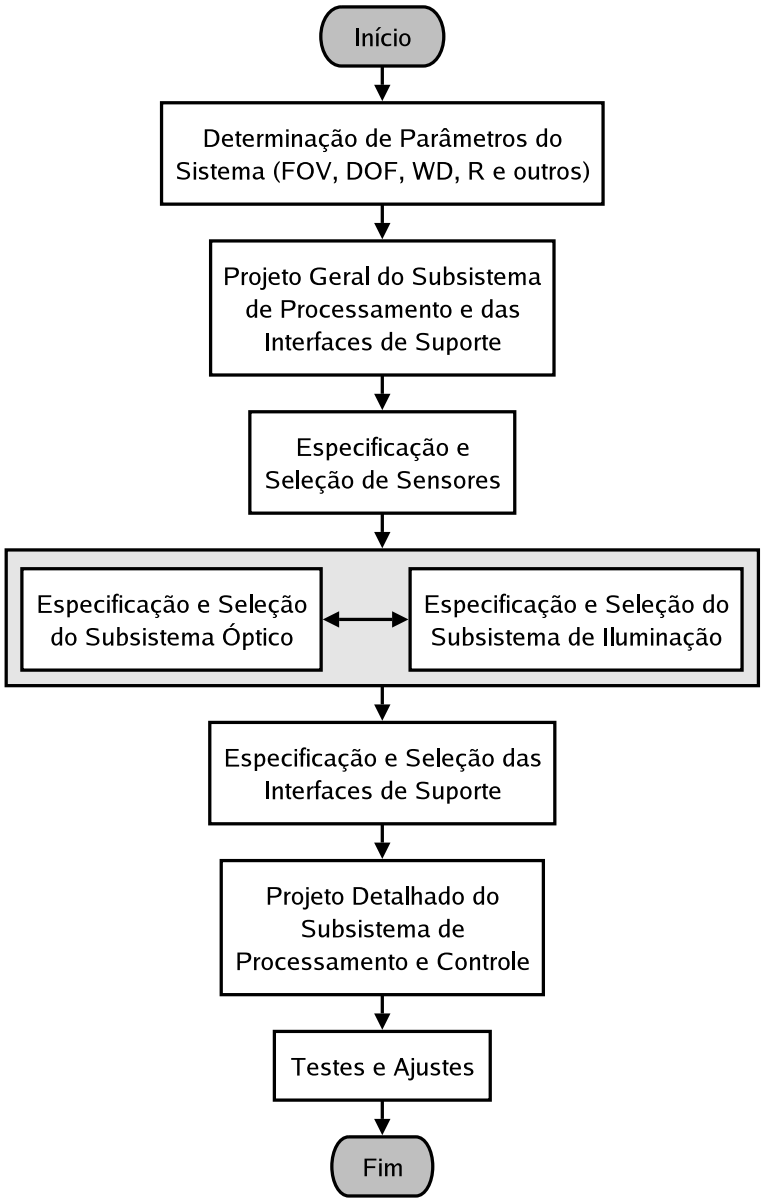


Figura 3.4: *Metodologia para o projeto de um sistema de visão.*

Tamanho do sensor (*Sensor Size – SS*): é o tamanho da área do sensor, tipicamente especificado pela sua dimensão horizontal. Este parâmetro é importante na determinação das características do sistem óptico necessárias para a obtenção de um determinado campo de visão.

O projeto de um sistema de visão inicia-se com o estudo das características do problema, determinando-se as dimensões daquilo que se deseja observar e os parâmetros envolvidos nestas observações, principalmente o campo de visão, a distância de trabalho, a resolução, a profundidade de campo e o tamanho do sensor. A partir destes parâmetros, um pré-projeto do sistema de processamento e das interfaces com os dispositivos externos ao sistema é feito. Esta etapa corresponde a uma etapa de análise dos requisitos do sistema. Geralmente, imagens prévias do objeto que se quer observar para testes já foram adquiridas, e um protótipo do algoritmo é desenvolvido a partir de uma biblioteca de processamento de imagens para se verificar a viabilidade do sistema.

Figura 3.5: *Parâmetros para a especificação de um sistema de visão.*
Fonte: Edmund Optics - http://www.edmundoptics.com.

A partir disto, o sensor mais adequado para se resolver o problema é selecionado, levando-se em consideração o tipo de varredura e a taxa de aquisição de imagens necessárias, juntamente com a resolução da imagem.

Em seguida, o projeto do subsistema óptico é elaborado a partir dos parâmetros do sistema e do sensor, concomitantemente à especificação de uma técnica de iluminação adequada para que as características mais relevantes do objeto observado sejam extraídas. Geralmente, a seleção do subsistema de iluminação se dá juntamente e de forma interativa com a seleção do subsistema óptico, sendo que o mais importante é a combinação de ambos para que boas imagens possam ser adquiridas pelo sensor da câmera.

Após definidos os sensores, o subsistema de iluminação e o subsistema óptico, as interfaces de suporte são especificadas e selecionadas, tipicamente englobando funções de apoio para o subsistema de processamento. Este último é então projetado, levando-se em consideração tanto o seu hardware, como as placas de aquisição de imagens e de entrada e saída, quanto o software geral do sistema e os algoritmos de processamento de imagens e visão computacional a serem programados e incorporados.

A seguir serão feitas algumas considerações sobre o projeto de cada um desses subsistemas de um sistema de visão.

3.3.1 Projeto do Subsistema Óptico

A correta especificação do subsistema óptico de um sistema de visão é extremamente importante para que se alcancem bons resultados no posterior processamento das imagens, sendo determinante para o sucesso da aplicação. O primeiro passo a se tomar quando do projeto do subsistema óptico de

uma aplicação de um sistema de visão é a familiarização com os parâmetros do sistema mostrados na figura 3.5.

A busca por componentes de um subsistema óptico que atendam às necessidades da aplicação passa pela determinação do aumento primário e dos limites de foco do sistema. O aumento primário é o quociente entre o campo de visão (FOV) e o tamanho do sensor (SS), como indicado na equação 3.1. Os limites de foco são obtidos a partir da variação da distância de trabalho. O foco mínimo é obtido com a distância de trabalho mínima e o foco máximo com a distância de trabalho máxima, como indicado nas equações 3.2 e 3.3.

$$PMAG = \frac{FOV}{SS} \tag{3.1}$$

$$f_{min} = \frac{WD_{min}}{(1 + \frac{1}{PMAG})} \tag{3.2}$$

$$f_{max} = \frac{WD_{max}}{(1 + \frac{1}{PMAG})} \tag{3.3}$$

A especificação de lentes se dá a partir da definição destes parâmetros, levando-se em consideração a resolução (R) e a profundidade de campo (DOF) requeridas pelo sistema. Estes dois fatores devem ser observados, de forma que uma lente com a resposta desejada seja selecionada, a partir das curvas de profundidade de campo e resolução por abertura da lente, como ilustrado na figura 3.6.

A determinação de outros componentes para o sistema deve ser realizada de acordo com cada aplicação [63].

3.3.2 Seleção do Subsistema de Iluminação

O projeto do sistema de iluminação é subdividido em três etapas principais [68]:

1. Determinação do tamanho do campo de visão (geralmente já determinado no projeto óptico).
2. Determinação da geometria da luz (posição da fonte, direção, intensidade e espectro freqüencial dos feixes de luz) requerida para ressaltar as características do objeto.
3. Determinação do tipo de fonte luminosa (LED, halogênia, estrobo ou outros) mais adequado a partir dos parâmetros anteriores.

A tabela 3.1 resume os principais tipos de sistemas de iluminação e suas aplicações, auxiliando na seleção de um sistema de iluminação adequado para uma determinada aplicação. A figura 3.7 ilustra esses tipos de iluminação.

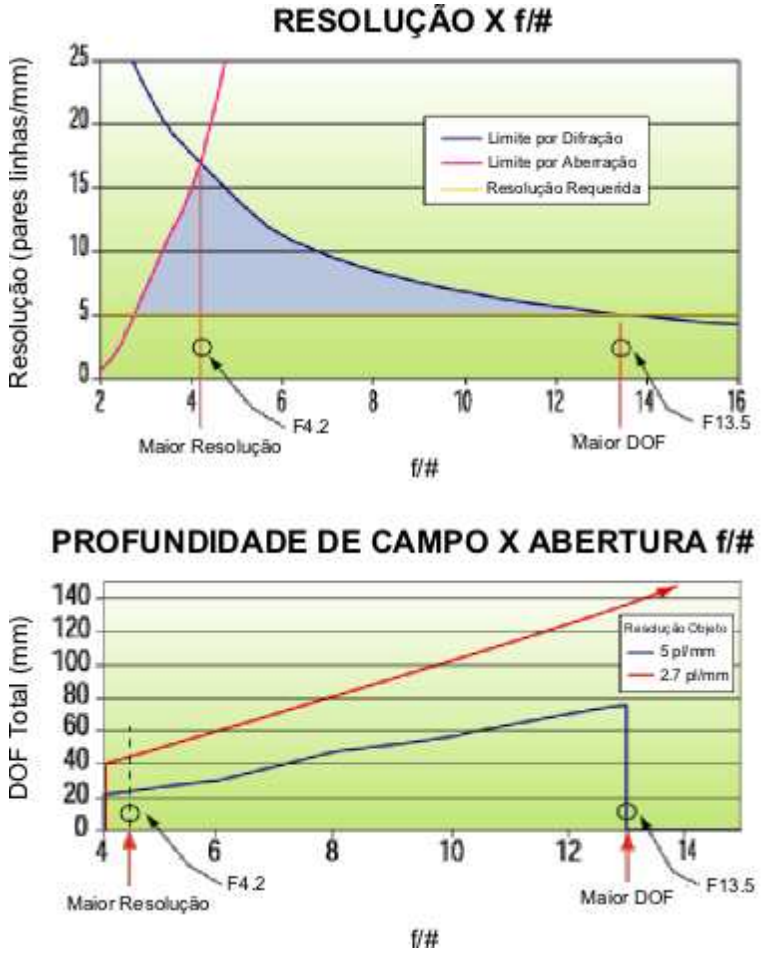


Figura 3.6: *Exemplo de gráficos da profundidade de campo e resolução pela abertura das lentes para a especificação de lentes.*
Fonte: Edmund Optics - http://www.edmundoptics.com.

3.3.3 Seleção de Sensores, Câmeras e a Transmissão de Dados

Na escolha de sensores, câmeras e técnicas e protocolos de transmissão de dados deve-se levar em consideração principalmente o tamanho do sensor necessário para a formação da imagem. O tipo do sensor deve ser definido de acordo com a aplicação, podendo ser de diferentes tipos [69]. Além disso, deve-se levar em consideração a possibilidade de intereferência na transmissão de dados (como no caso de ruídos eletromagnéticos em aplicações de chão-de-fábrica) e as necessidades da aplicação em termos de taxa de aquisição e transmissão de dados.

Para a especificação de uma câmera com sensor do tipo CCD, é importante determinar o tamanho deste em pixels. Para isso, as seguintes relações são úteis:

$$PS_{horizontal} = \frac{FOV_{horizontal}}{R_{horizontal}} \tag{3.4}$$

$$PS_{vertical} = \frac{FOV_{vertical}}{R_{vertical}} \tag{3.5}$$

Tipo de Iluminação	Aplicação
Anel	Iluminação de superfícies planas e difusas.
Campo Escuro	Ilumina apenas o contorno e os detalhes de um objeto.
Cúpula	Iluminação uniforme de superfícies planas e difusas.
Iluminação de Fundo	Ilumina o contorno externo e furos de um objeto.
Dia Nublado	Iluminação uniforme para superfícies muito reflexivas.
Difusa	Reduz sombras e ilumina uniformemente superfícies levemente irregulares.
Direcional	Iluminação de superfícies planas e difusas.
Multi-Direcional	Iluminação uniforme para superfícies planas, criando contraste entre superfícies difusas, especulares e absorventes.

Tabela 3.1: *Principais técnicas de iluminação e suas aplicações.*

3.3.4 Seleção do Subsistema de Processamento

A escolha do subsistema de processamento inicia com a seleção da placa de aquisição de imagens que irá servir para disponibilizar os dados adquiridos pelo sensor ao programa de processamento. Algumas considerações quando da escolha desta placa são [69]:

- O formato de entrada do vídeo (analógico, digital ou de outro tipo) proveniente do subsistema de transmissão de dados.
- A taxa de aquisição da imagem (*frame rate* ou taxa de quadros por segundo).
- O tipo de barramento utilizado para interfaceamento com a estação de processamento.
- Alocação de memória (própria ou pelo sistema operacional).
- Mecanismo de digitalização da imagem.
- Possibilidade de processamento embutido para determinados tipos de operações como filtragem e seleção de regiões de interesse (ROIs – *Regions of Interest*).

A partir destas considerações, o software do sistema é projetado, levando em consideração as bibliotecas de processamento e os drivers para acesso aos dados disponibilizados pela placa.

3.3.4.1 Seleção de Algoritmos de Processamento

A parte de seleção de algoritmos de processamento de imagens é uma das partes mais delicadas do projeto de um sistema de visão. Em primeiro lugar, porque são os algoritmos que irão determinar se uma aplicação irá conseguir realizar determinada tarefa com a informação obtida. Em segundo lugar, porque a seleção dos algoritmos de processamento depende em grande parte da experiência da pessoa responsável pelo sistema.

Algoritmos de processamento podem ser classificados em algoritmos de baixo, médio e alto nível, como sugerido pela figura 3.1. Referências para a seleção de algoritmos podem ser enontradas nos livros de Gonzalez e Woods [61], Jähne [81], Pratt [82], Ritter e Wilson [83] e Jähne et al. [93].

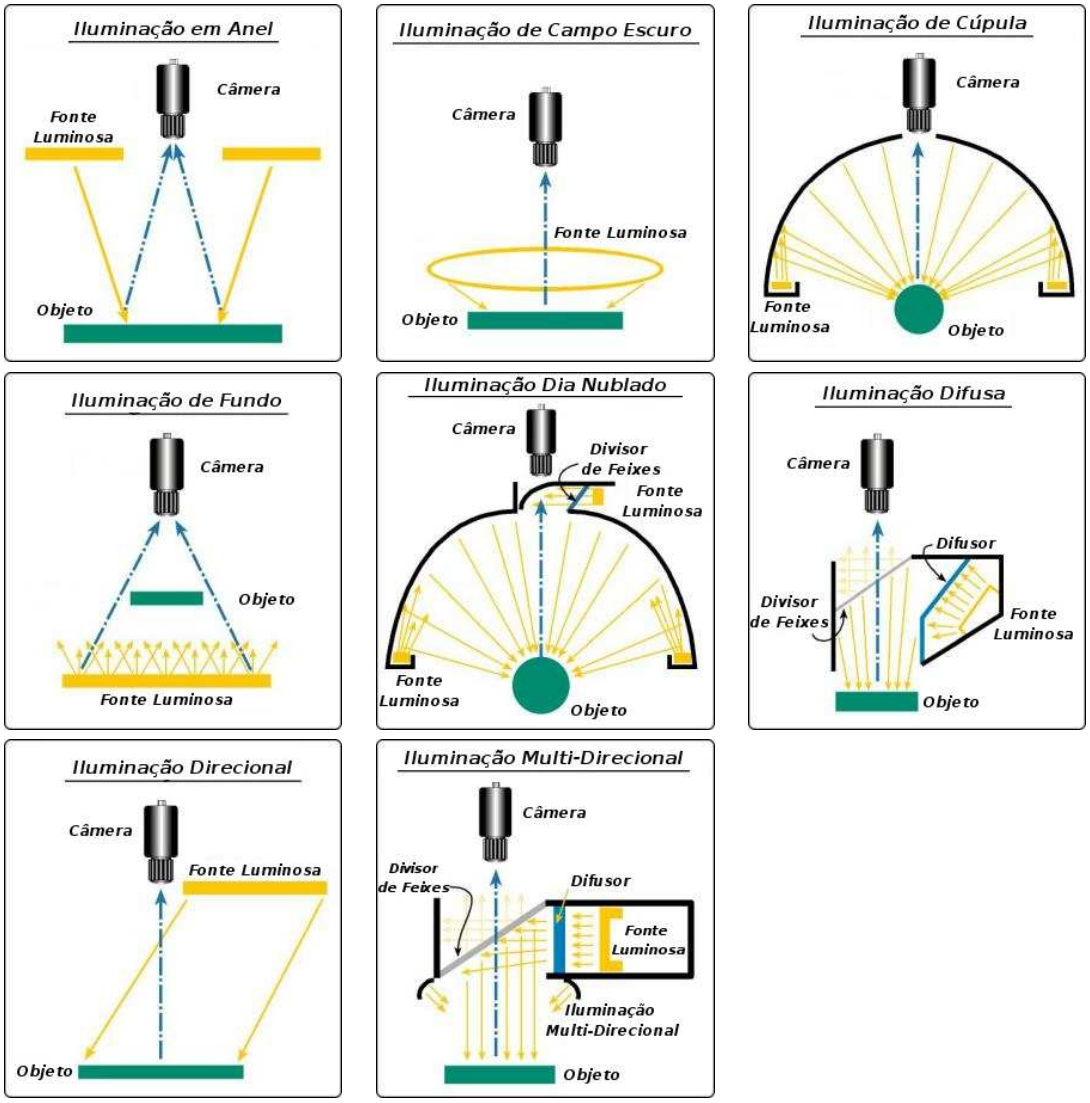


Figura 3.7: *Diferentes tipos de iluminação para sistemas de visão.*
Fonte: RVSI/NER - http://www.nerlite.com.

3.3.4.2 O Projeto do Software

A programação de sistemas de processamento de imagens e visão computacional é uma tarefa que consome bastante tempo e que requer conhecimento especializado sobre os efeitos dos algoritmos de processamento de imagens, bem como conhecimento sobre a implementação e interfaces do sistema. O maior problema no projeto de sistemas de processamento de imagens é a necessidade de código eficiente em tempo de execução e acesso a dispositivos de hardware em baixo nível, em contraponto à necessidade de uma implementação de solução geral e independente de plataforma que forneça tipos de dados e funções de mais alto nível para o processamento das imagens.

De acordo com Paulus et al. [94], os requisitos básicos para o projeto de um sistema de software de propósito geral para o processamento de imagens são:

- Possibilidade de acesso aos dispositivos de hardware como placas de aquisição de imagens e

sinais, interfaces de câmeras e outros dispositivos de entrada e saída. Como o desenvolvimento de novos hardwares ocorre a uma velocidade maior do que a velocidade com que o software pode mudar e porque o reuso de software é desejável, as interfaces devem ser encapsuladas por definições transportáveis.

- O acesso rápido a vetores e matrizes deve ser possível (pelo uso intensivo dessas estruturas nesse tipo de sistema).
- A entrada e saída deve ser rápida, eficiente e independente de máquina. Isto deve ser garantido não apenas para estruturas de baixo nível, como matrizes de imagens, mas também para tipos de dados de mais alto nível, como resultados de operações de segmentação e bases de conhecimento.
- Os módulos de processamento e análise das imagens devem ser o mais independente possíveis da aplicação final de forma a serem reutilizáveis entre diferentes sistemas.

Paulus et al. [94] também propõem uma abordagem para o teste do software de sistemas de visão, como por exemplo:

- Toda linha de código do programa deve ser executada no mínimo uma vez em cada seqüência de testes.
- Todo laço de controle de execução (um *for*, por exemplo) deve ser usado pelo menos duas vezes durante uma seqüência de testes.
- Testar o sistema com dados irregulares, não esperados, errados ou inconsistentes, como, por exemplo, imagens 0x0, 10000x1, imagens com a mesma intensidade de cor em todos os seus pixels.
- Testar o sistema com todo o tipo de imagens, variando todos os parâmetros possíveis, como intensidade e contraste.
- Testar a previsibilidade de alguns casos, como discontinuidades de funções e divisão por números próximos de zero.
- Manter em mente a limitação de recursos como memória e estimar estes recursos, verificando o comportamento do sistema com recursos mínimos ou insuficientes.

3.4 Aplicação de Sistemas de Visão ao Monitoramento do Desgaste de Ferramentas de Corte

Nesta seção descreve-se o estado da arte da tecnologia de monitoramento do estado de ferramentas de corte através de sistemas de visão. Inicia-se a discussão com uma visão geral das primeiras abordagens para em seguida discutir-se as abordagens recentes. Tanto o monitoramento do desgaste

de flanco quanto do desgaste de cratera são abordados, dando-se ênfase ao primeiro tipo. Considerações gerais sobre a aplicação de sistemas de visão ao monitoramento do estado de ferramentas de corte são feitas ao final do capítulo.

3.4.1 Primeiras Abordagens

A primeira tentativa de se usar um sistema de visão para o monitoramento do desgaste de ferramentas é geralmente creditada a Matshushima et al. [95]. A ferramenta de corte, neste sistema, era analisada por uma câmera de TV a cada troca da ferramenta. A imagem capturada era em tons de cinza, sendo posteriormente binarizada utilizando-se um valor de *threshold* determinado de acordo com a ferramenta, aplicação e condições de usinagem. A área do desgaste de flanco era medida diretamente através da contagem do número de pixels na direção mais provável do desgaste, fixada previamente. Devido a variações no posicionamento da ferramenta e na iluminação e a constante necessidade de ajuste do parâmetro de *threshold* este sistema possuía uma aplicação bastante restrita, com repetibilidade não garantida.

No desenvolvimento de seu sistema para o monitoramento do desgaste de ferramentas, Cuppini et al. [96] sofisticaram um pouco mais a solução anterior. Utilizando uma câmera de TV equipada com um sistema óptico e um sistema de iluminação com fibras ópticas direcionáveis, tudo montado em uma máquina-ferramenta. Três diferentes algoritmos de segmentação foram testados, sendo as medições realizadas durante intervalos da produção. Seus resultados não forneceram, contudo, um estudo comparativo das técnicas de segmentação.

Em uma outra abordagem, Lee et al. [97] posicionou a ferramenta sob um microscópio equipado com uma câmera, usando um dispositivo de fixação especialmente projetado para esta finalidade. O processamento da imagem da ferramenta foi feito em duas etapas: primeiramente, um simples algoritmo para melhorar o contraste da imagem foi aplicado a este para então, a imagem ser segmentada interativamente pelo usuário para separar a região desgastada. Esta abordagem, pela etapa de segmentação interativa, ficou limitada a uso em laboratório. Neste trabalho, pela primeira vez, a iluminação foi identificada como um fator chave para o desempenho do sistema.

Giusti et al. [36] desenvolveram um sistema que consistia em dois subsistemas de iluminação distintos com uma única câmera para adquirir uma imagem da face principal e dos flancos principal e secundário da ferramenta pela variação no uso dos sistemas de iluminação. A iluminação de uma fonte difusa foi utilizada para realçar as diferenças entre regiões desgastadas e não desgastadas da ferramenta. A segmentação foi feita através da utilização de uma estratégia local, analisando-se faixas longitudinais da imagem com altura de 10 pixels, como pode ser visto na figura 3.8. Este sistema, com poucas modificações, foi utilizado com relativo sucesso em diversas máquinas sob diferentes condições de corte. Contudo, por capturar uma imagem deslocada da ferramenta e não em um plano paralelo ao plano da câmera, fornece valores não ajustados para o desgaste.

Uma alternativa de sistema de visão com uma fonte de iluminação direcional bastante concentrada a laser foi utilizada por Jeon e Kim [98]. Uma seqüência de passos de processamento de imagens

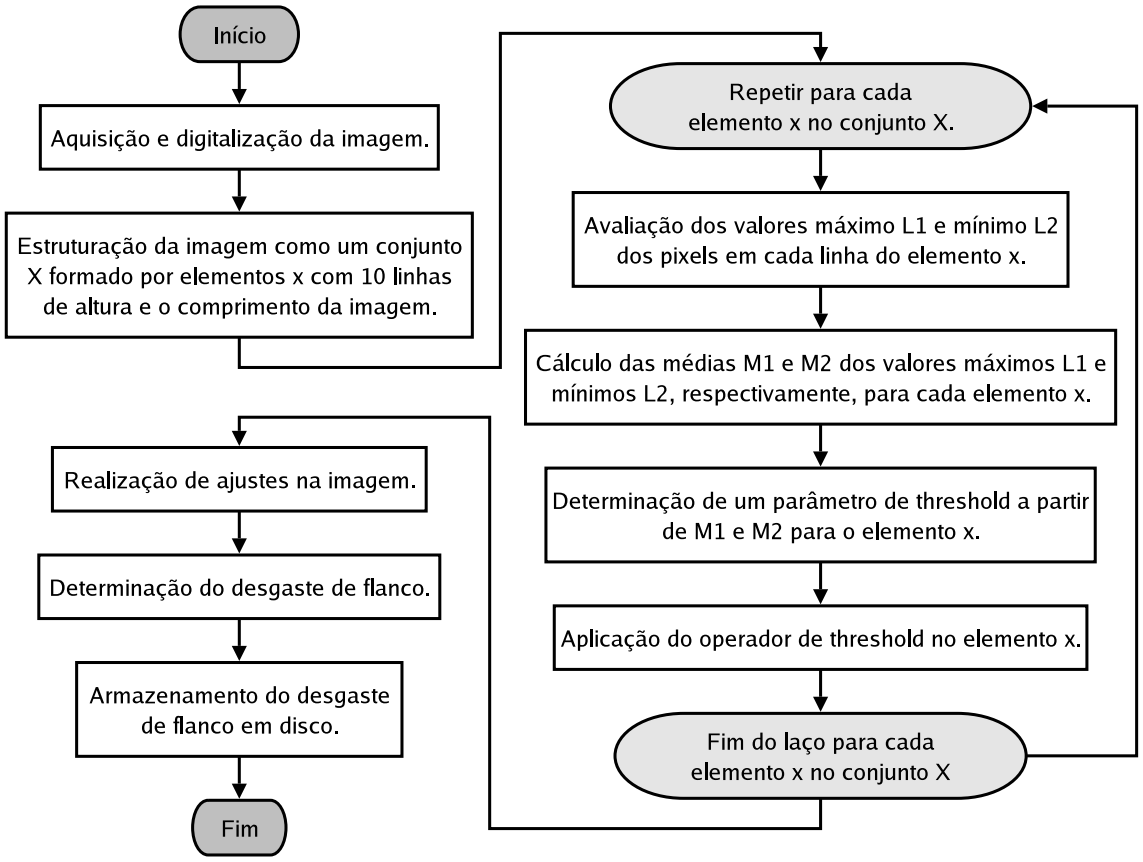


Figura 3.8: *Estratégia de medição do desgaste de flanco através de faixas longitudinais.*

foi aplicada sobre a imagem adquirida já binarizada. Os resultados obtidos com este sistema se mostraram bastante satisfatórios, apesar da pequena região realçada pela fonte de iluminação.

A possibilidade do uso de um sistema de visão montado em uma máquina-ferramenta para a medição do desgaste de ferramentas de corte foi investigado por Pedersen [37]. O sistema constituído de uma câmera e uma lâmpada halogênea foi montado em um torno VDF-Boehringer PNE 480. A região de desgaste de flanco foi delineada através da seleção de um *threshold* a partir de um histograma de níveis de cinza. As medidas consecutivas realizadas pelo sistema com a mesma ferramenta em vários estágios diferentes do processo de usinagem caracterizaram-se estar de acordo com um modelo de desgaste em três estágios[1]. Neste trabalho também as primeiras medidas acerca do tempo de execução do algoritmo para a medição do desgaste foram feitas, chegando a ficar, em média, em 7 segundos em um IBM PC-AT. Um fluxograma do algoritmo deste sistema pode ser visto na figura 3.9.

Teshima et al. [99] foram os primeiros a integrar a um sistema de visão uma rede neural artificial com o intuito de prever o tempo de vida de uma ferramenta. O estado do desgaste de flanco e de cratera determinado através de um sistema básico de processamento de imagens por níveis de brilho em 12 cores diferentes, juntamente com as condições de usinagem eram fornecidas como entradas

[1]Pedersen [37] discute, em seu trabalho, este modelo: o primeiro estágio é o inicial, caracterizado por um desgaste bastante lento. O segundo estágio é o em regime permanente, caracterizado por um desgaste mais rápido, porém constante. O terceiro estágio é o terminal, no qual o desgaste cresce exponencialmente.

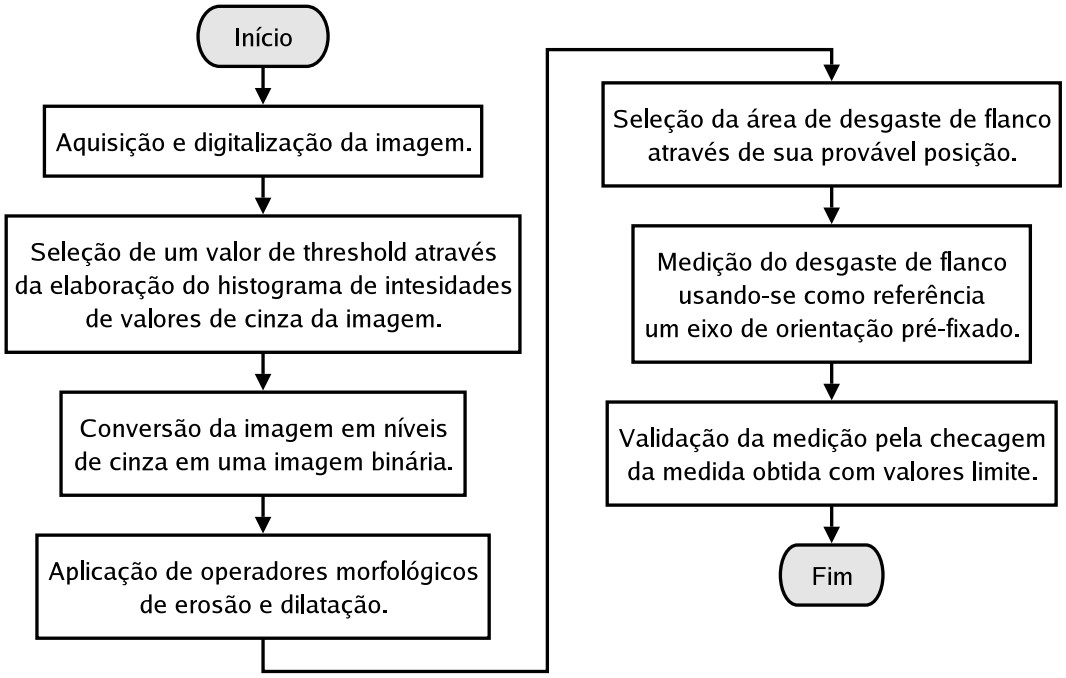


Figura 3.9: *Estratégia de medição do desgaste de ferramenta implementada em um torno VDF-Boehringer PNE 480.*

para uma rede neural de três camadas, que previa o restante do tempo de vida da ferramenta e o tipo de desgaste. Ênfase, neste sistema, era colocada mais no processamento da rede neural do que na avaliação do desgaste.

Por fim, Maeda et al. [100] desenvolveram um sistema de processamento de imagens utilizando-se a diferença entre as imagens da ferramenta desgastada e não desgastada. Os problemas com este sistema eram o posicionamento entre as duas imagens da ferramenta, que deveria ser perfeito para que a estimação pudesse ser feita e também as variações de iluminação na aquisição das duas imagens. Esta abordagem foi usada, posteriormente, por Orth [17] em sua dissertação no desenvolvimento de um algoritmo automático.

A tabela 3.2 resume as primeiras abordagens de sistemas de monitoramento do desgaste de ferramentas de corte através de sistemas de visão.

3.4.2 Abordagens Recentes

Na grande maioria das primeiras abordagens da aplicação de sistemas de visão para o monitoramento do desgaste de ferramentas de corte, a parte de processamento de imagens é extremamente simplificada e suscetível a gerar resultados errados, tendo um mau funcionamento no caso de condições de variação na iluminação ou de mudanças da textura superficial da ferramenta.

Muitos dos pesquisadores usaram a mesma abordagem de segmentação baseada em *threshold* a partir de histogramas de níveis de cinza, destacando duas regiões de interesse na imagem (área desgastada e área não desgastada) antes da derivação de parâmetros morfológicos que quantificassem

Autores	Ano	Tipo de Segmentação	Aplicação
Matshushima et al. [95]	1979	*threshold*	laboratório
Cuppini et al. [96]	1986	algoritmo baseado em *threshold* global	laboratório
Lee et al. [97]	1986	segmentação manual	laboratório
Giusti et al. [36]	1987	algoritmo baseado em *threshold* local	máquina-ferramenta
Jeon e Kim [98]	1988	binarização	laboratório
Maeda et al. [100]	1987	diferença com imagem modelo e *threshold* global	laboratório
Pedersen [37]	1990	algoritmos baseados em *threshold* global	torno VDF-Boehringer PNE 480
Teshima et al. [99]	1993	medição indireta por rede neural	laboratório

Tabela 3.2: *Quadro comparativo das primeiras abordagens para a medição do desgaste de ferramentas de corte utilizando sistemas de visão.*

o desgaste da ferramenta. Park e Ulsoy [101] afirmam que uma eficiente estratégia de segmentação da imagem é um fator chave para a medição satisfatória do desgaste de flanco.

Desta forma, abordagens mais robustas para o processamento das imagens, além da aplicação de um simples *threshold* local ou global, se fazem necessárias para o processamento das imagens em níveis de cinza. O desenvolvimento de abordagens neste sentido caracteriza esta área de pesquisa recentemente. A seguir, as principais abordagens são descritas.

Uma metodologia utilizando-se um processo de segmentação em duas etapas para identificar as três regiões distintas em uma imagem de uma ferramenta desgastada (região desgastada, região não desgastada e fundo da imagem) foi proposta por Oguamanam et al. [40]. A transformada de Hough Gonzalez e Woods [61] que, a grosso modo, identifica linhas retas e arcos na imagem, foi utilizada para determinar a ponta da ferramenta. Foram realizados experimentos com dois tipos de materiais como matéria-prima da usinagem, e a forma da região do desgaste mostrou-se diferente para cada tipo de material. A seleção de parâmetros para os algoritmos de segmentação é um ponto chave neste sistema para a obtenção de bons resultados. A figura 3.10 mostra de forma esquemática a estratégia utilizada para este sistema.

Os seguintes parâmetros são então utilizados para oferecer um diagnóstico do estado da ferramenta: desgaste de flanco máximo, área do desgaste de flanco, perímetro da região do desgaste de flanco, comprimento máximo da região de desgaste de flanco e o ponto mais próximo da ferramenta desgastada em relação à quina da ferramenta não desgastada.

Um outro sistema de monitoramento do desgaste de ferramentas foi proposto por Kurada e Bradley [35] e tem o seu diagrama funcional ou fluxograma exposto na figura 3.11. Este sistema é composto por uma fonte de iluminação de fibra óptica e uma câmera com sensor CCD em conjunto com um microscópio de alta resolução. O sistema utiliza operadores de textura e gradiente para calcular a área da região de desgaste, propondo uma técnica de segmentação de imagens baseada nas diferenças de textura entre as regiões desgastada e não desgastada da imagem.

Em outro trabalho, Kurada e Bradley [5] procuram generalizar a idéia de um sistema de monitoramento do desgaste de ferramentas. Analisando o sistema componente a componente e notando-se que

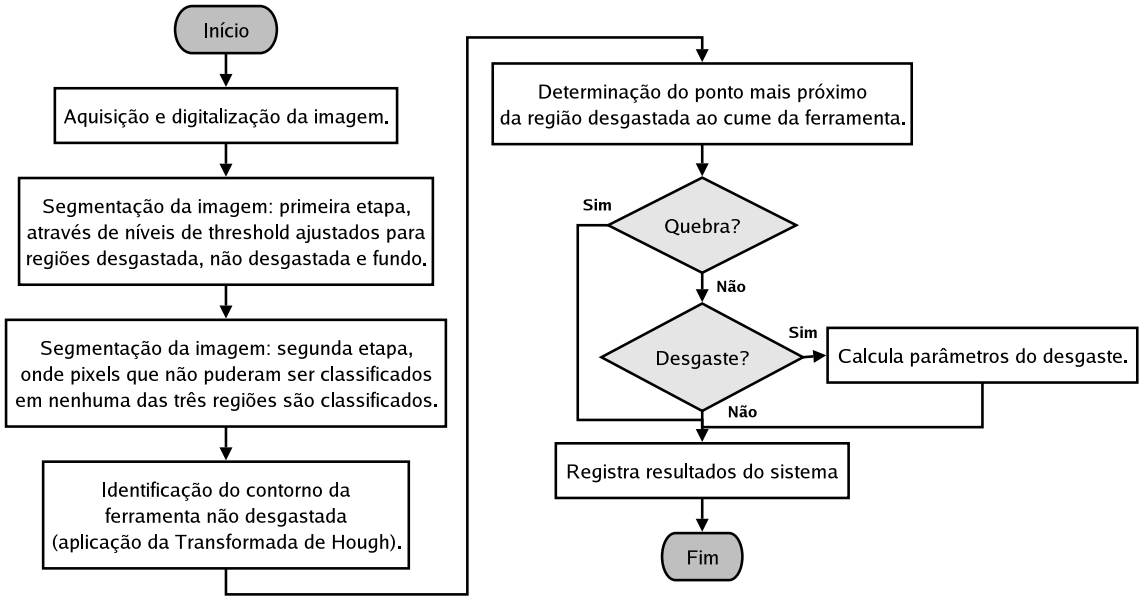


Figura 3.10: *Estratégia de medição do desgaste com duas etapas de segmentação.*

as técnicas de iluminação e segmentação são de extrema importância para o sucesso da abordagem, chega-se a uma proposta de um modelo geral para um tal sistema, como mostrado na figura 3.12.

Lanzetta [34] propõe um esquema de um sensor para a medição e classificação do desgaste de ferramentas de corte baseando-se no esquema exposto na figura 3.13.

Lanzetta [34] também estuda as principais morfologias do desgaste de ferramenta e da quebra de ferramenta e propõe uma abordagem para reconhecer entre os diferentes tipos de desgaste. A abordagem completa é bastante sofisticada e foge ao escopo deste trabalho. Vale notar, no entanto, que o sistema para a classificação do desgaste implementado leva em consideração os parâmetros medidos pelo sistema, como o desgaste de flanco máximo, o desgaste de flanco médio e o perímetro do desgaste de cratera para determinar o tipo de desgaste.

Por fim, Ji et al. [39] desenvolveram um sistema de monitoramento do desgaste de flanco baseado no cálculo de parâmetros conhecidos como distâncias de Mahalanobis da imagem das ferramentas em processo de desgaste. A distância de Mahalanobis é uma medida de distância baseada na relação entre pontos e conjuntos de pontos. Maiores detalhes sobre o algoritmo de Mahalanobis podem ser encontrados no livro de Castleman [102]. A idéia é usar esses parâmetros de forma contínua dentro do próprio processo de usinagem fazendo com que, a partir da mudança dos parâmetros das distâncias de Mahalanobis, o desgaste possa ser classificado. A tabela 3.3 compara essas diferentes abordagens de forma resumida.

Cabe aqui um comentário quanto à aplicação de técnicas 3D para a monitoramento do estado das ferramentas. Estas técnicas têm se limitado à medição de parâmetros do desgaste de cratera. Abordagens da medição do desgaste de cratera através da utilização de iluminação estruturada (pela projeção de franjas, por exemplo) como é ilustrado no trabalho de Lanzetta [34] ou pelo uso de técnicas de visão estereoscópicas, como nos trabalhos de Prasad e Ramamoorthy [44], Karthik et al. [45] e Yang e Kwon [43] são as mais exploradas.

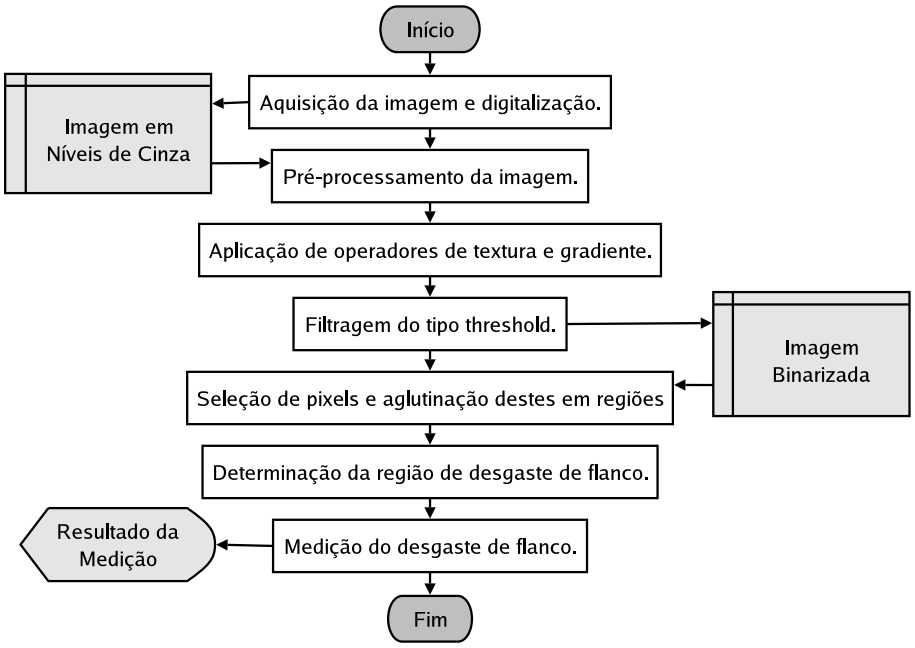


Figura 3.11: *Principais passos do sistema para medição do desgaste de flanco ferramentas de corte através de um microscópio de alta resolução.*

3.5 Considerações Gerais Sobre a Aplicação de Sistemas de Visão ao Monitoramento do Desgaste de Ferramentas de Corte

Pode-se perceber na literatura da área o uso intensivo de técnicas de segmentação baseadas em valores de *threshold*. Em geral, a utilização destes sistemas se deu sob condições controladas e não propriamente em um ambiente de produção, com o sistema montado em uma máquina-ferramenta.

De acordo com Kurada e Bradley [5], alguns pontos importantes a serem levados em conta quanto ao uso de sistemas de visão aplicados ao desgaste de ferramentas de corte são:

- O uso de imagens em escala de níveis de cinza deve ser explorado. A maior quantidade de

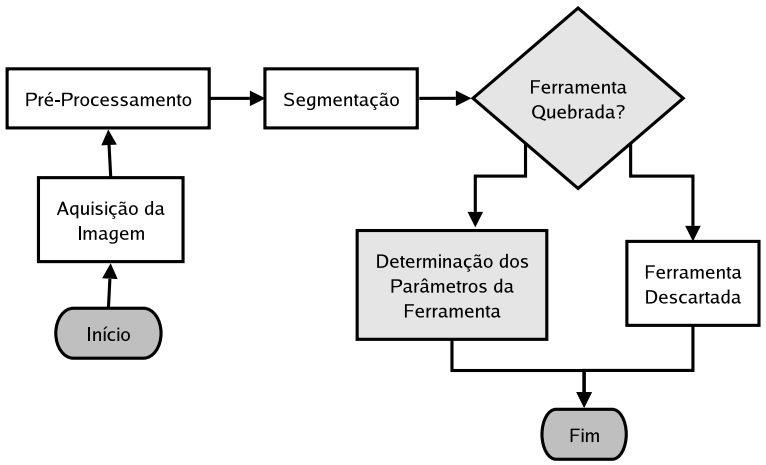


Figura 3.12: *Esquema de um sistema geral para o monitoramento do desgaste de ferramentas.*

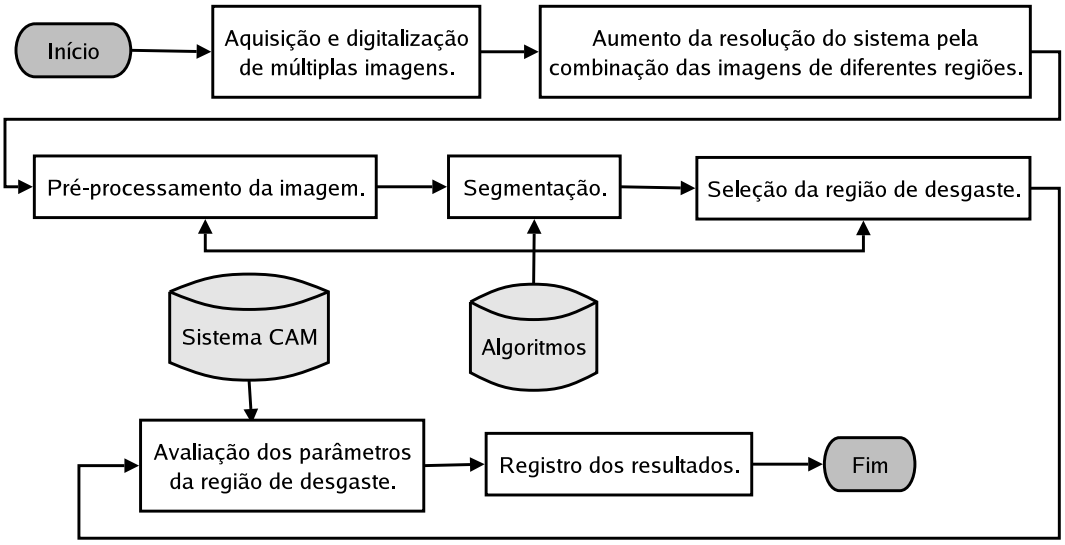


Figura 3.13: *Esquema do sistema para o monitoramento do desgaste de ferramentas proposto por Lanzetta.*

Autores	Ano	Tipo de Segmentação	Aplicação
Oguamanam et al. [40]	1994	segmentação em duas etapas com o uso da transformada de Hough	torno CNC
Kurada e Bradley [35]	1997	operadores de textura e gradiente	laboratório
Lanzetta [34]	2001	operadores de textura	laboratório
Ji et al. [39]	2002	algoritmo de distância de Mahalanobis	laboratório

Tabela 3.3: *Quadro comparativo das abordagens recentes para a medição do desgaste de ferramentas de corte utilizando-se sistemas de visão.*

informação presente neste tipo de imagens proporciona um monitoramento mais eficaz do desgaste da ferramenta sem custo adicional. Características da textura da ferramenta ou ainda técnicas de reconhecimento de padrões podem ser aplicadas a imagens em níveis de cinza sem problemas.

- Informação de textura obtida da peça usinada pela ferramenta deve ser incorporada no modelo de monitoramento do desgaste da ferramenta. Tal informação pode ser vital na seleção das condições de usinagem. Esta informação poderia ser obtida pelo mesmo sistema de visão que obteve a imagem da ferramenta, com algumas pequenas modificações.

- A grande maioria das pesquisas conduzidas utilizando sistemas de visão para a medição do desgaste de ferramentas envolve o uso de *threshold* global. O uso de algoritmos de segmentação mais eficientes, como técnicas baseadas em porções da imagem deveriam ser investigadas, uma vez que estas são menos suscetíveis a erros.

- Para se estabelecer a validade da técnica de monitoramento do desgaste de ferramentas através de sistemas de visão deve-se estudar mais o impacto de materiais de ferramentas e condições de usinagem nestes sistemas. Os resultados devem ser validados através de modelos teóricos propostos no passado, uma vez que as informações obtidas através de técnicas tradicionais é bastante limitada.

- Um estudo comparativo do desempenho do sistema em termos de incertezas deveria ser feito com todos os sistemas de visão implementados para a medição do desgaste de ferramentas.

Segundo Lanzetta [34], seria desejável que um sistema de visão para a medição do desgaste de ferramentas de corte apresentasse as seguintes características:

- Ser automático, ou seja, com capacidade autônoma de decisão.
- Ser flexível, ou seja, capaz de reconhecer um grande número de formas diferentes de desgaste.
- Ser expansível, ou seja, ser capaz de guardar informação de novos tipos de defeitos e de desgaste.

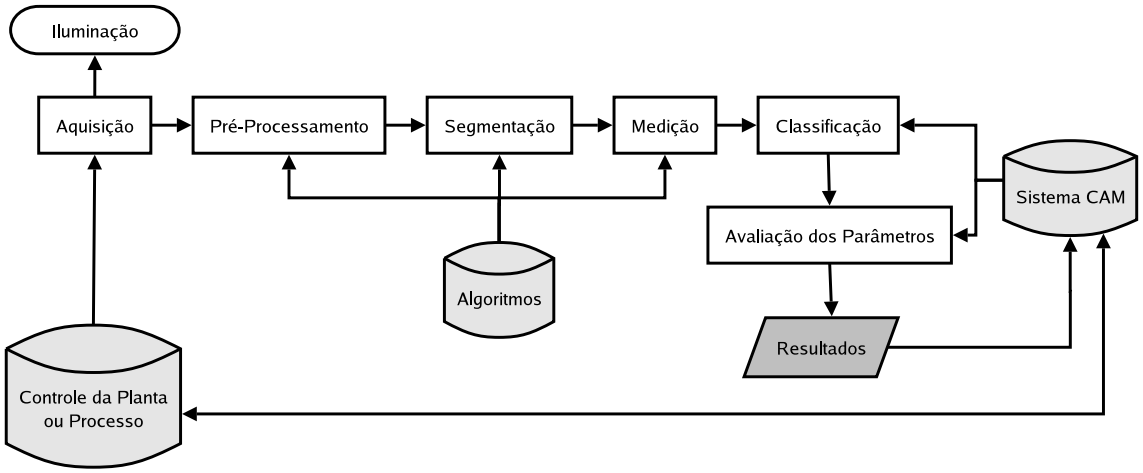


Figura 3.14: *Esquema geral de um sistema de monitoramento do estado da ferramenta por sistema de visão*

.

O efeito e a importância da iluminação em sistemas de monitoramento do desgaste de ferramentas de corte é discutido por Kurada e Bradley [5] e por Lanzetta [34]. Segundo os autores, uma das partes mais importantes da configuração dos dispositivos de tais sistemas é o subsistema de iluminação. É necessário contraste adequado entre a região desgastada da ferramenta e o fundo da imagem. Além disso, os autores colocam a importância da seleção de uma técnica de iluminação adequada para a obtenção de resultados ótimos. As três técnicas de iluminação mais utilizadas em aplicações de sistemas de visão são:

- Iluminação frontal, amplamente utilizada na medição do desgaste de flanco na inspeção superficial da peça.
- Iluminação de fundo, fornecendo um excelente contraste, mas limitando-se a uma imagem apenas do contorno do objeto sendo inspecionado.
- Iluminação estruturada, referindo-se às fontes de iluminação em que se projetam padrões de linhas, por exemplo, sobre o objeto, utilizadas para estimar, por exemplo, a profundidade de uma região de desgaste de cratera.

Não se pode deixar de levar em consideração, em um sistema de medição de desgaste de ferramentas, a integração deste sistema com o sistema geral de controle da máquina e do processo produtivo como um todo. A partir dos modelos analisados e desta observação, pode-se chegar a um modelo mais generalizado para um sistema de monitoramento do desgaste de ferramentas de corte através de sistemas de visão, modificando e estendendo o esquema proposto por Kurada e Bradley [5], como pode ser visto na figura 3.14.

É a partir deste modelo que será desenvolvido o sistema TOOLSPY, levando-se em consideração as observações acerca do sistema de iluminação e de processamento de imagens e visão computacional.

Capítulo 4

O Sistema TOOLSPY e a API IPFRAMEWORK

Neste capítulo, os requisitos operacionais do sistema TOOLSPY para o monitoramento do estado de ferramentas serão descritos e analisados. Em seguida, uma proposta de arquitetura genérica para a implementação do software do subsistema de processamento do sistema de visão TOOLSPY será detalhada. O projeto desta arquitetura será discutido e a as suas interfaces descritas.

4.1 Requisitos do Sistema TOOLSPY

A análise dos requisitos do sistema TOOLSPY começa com uma discussão sobre o papel das necessidades das células autônomas de produção.

4.1.1 Células Autônomas de Produção – APCs

Atualmente, para indústrias de diferentes áreas, processos altamente automatizados e com um alto grau de precisão estão se tornando cada vez mais necessários para que custos sejam reduzidos e a qualidade de produtos seja melhorada, permitindo-se assim maior competitividade no mercado [103]. Devido a estes fatores, o desenvolvimento de equipamentos e instalações industriais que possam executar processos complexos com confiabilidade e com um máximo grau de independência sem a supervisão humana por um longo período de tempo é uma tarefa imprescindível.

Extensões às funcionalidades e às capacidades dos equipamentos de instalações industriais da atualidade ainda são necessárias até que uma máquina-ferramenta possa alcançar o supra-citado grau de independência [104]. É apenas através da integração de atividades produtivas como planejamento, controle de máquina, interface com os usuários, fixação, manuseio e transporte de materiais, supervisão de processos, remoção de subprodutos industriais e prevenção de falhas que uma máquina-ferramenta ou outro equipamento industrial poderá executar suas atividades produtivas de forma independente e confiável.

Esse requisito de integração de equipamentos de instalações produtivas foi tomado como alvo de pesquisa por diversos institutos da RWTH-Aachen e os levou à criação, na Alemanha, de uma área especial de pesquisas (referida, em alemão, como *Sonderforschungsbereich 368* ou simplesmente SFB368) do conselho alemão de pesquisa (*Deutsche Forschungsgemeinschaft* – DFG) em células autônomas de produção [105]. As pesquisas nessa área envolvem desde a integração de aspectos do planejamento da produção como a integração de sensores no nível de chão-de-fábrica.

As células autônomas de produção são definidas como coleções de equipamentos de produção autônomos que lidam com problemas de produção como planejamento, controle, interface com o usuário, fixação, manuseio e transporte de materiais de forma independente e integrada [103, 104]. Para as pesquisas da SFB368, dois processos comuns da engenharia mecânica foram escolhidos para um estudo aprofundado: o processo de fresamento e o processo de soldagem a laser [105]. A partir destes processos foram definidos os subprojetos e suas respectivas áreas de atuação. Esses subprojetos estão representados na figura 4.1.

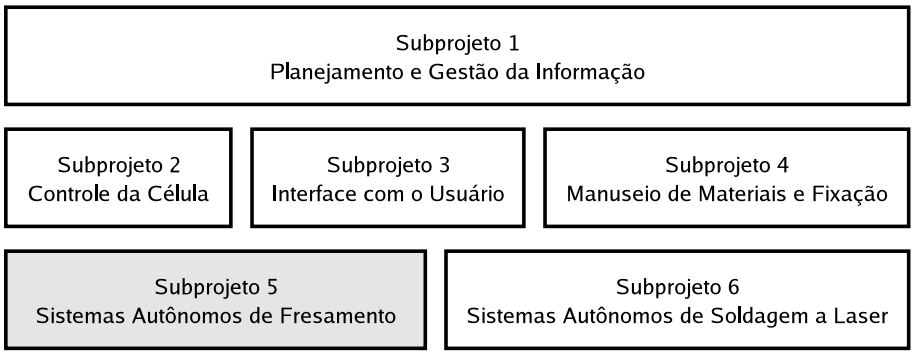


Figura 4.1: *Subprojetos do SFB368, vistos em três níveis de agrupamento: o nível de planejamento, o nível de controle e interface e o nível de aplicação.*

Primeiramente, para a execução independente de um processo de produção, funções de planejamento que produzam dados e programas para o controle da seqüência de produção a partir de modelos CAD e outros dados de entrada como planos de processo e ordens de produção são necessárias [104]. Para isso, no contexto da pesquisa em células autônomas de produção, métodos para a definição automática do planejamento da produção estão sendo desenvolvidos, definindo-se assim parâmetros para o manuseio e transporte de materiais bem como mecanismos de controle automáticos para a modificação desses parâmetros durante o processamento.

Apesar de os dispositivos deverem operar independentemente, intervenções pelo usuário, fazendo uso de seu conhecimento e experiência, devem ser possíveis a qualquer momento [106]. Isso requer que informações de ambos o processo e o equipamento no qual o processo está sendo executado estejam disponíveis localmente para o usuário da máquina em um formato adequado [107]. As interfaces com o usuário devem estar de acordo com estes requisitos, garantindo uma representação permanente dos estados da máquina e do processo e possibilitando a intervenção do usuário no processo a qualquer momento.

De forma a aumentar a flexibilidade dos equipamentos e o espectro de peças capazes de serem manufaturadas, novos sistemas de manuseio, transporte e fixação estão sendo desenvolvidos [108].

Novos dispositivos de fixação que se adaptem passivamente à peça e sistemas de medição integrados às máquinas, utilizados após a etapa de manuseio e transporte para monitorar o estado das peças estão sendo desenvolvidos. Pela realimentação dos resultados da medição após o manuseio das peças às funções de planejamento, este pode ser automaticamente adaptado [104].

Quanto maior a demanda por processamento em um equipamento de produção, maior é o número de produtos que apresentam desvios em termos de qualidade. De forma a compensar este fato, medidas são feitas durante o processamento e são usadas para controlar esses desvios, fornecendo realimentação às outras funções da produção como o planejamento da produção [103]. O objetivo é permitir que a máquina possa reagir a distúrbios no processamento de forma mais independente possível, detectando esses distúrbios na máquina, na ferramenta, nas peças ou no ambiente e tomando ações de forma a compensá-los. Desta forma, todos os parâmetros relevantes do processo como forças de corte, alinhamentos entre peça e ferramenta ou peça e sistema de manuseio e correntes elétricas, por exemplo, devem ser monitorados e analisados de forma conveniente. Esses parâmetros são usados para a determinação dos estados da máquina e do processo e servem para indicar a quebra de uma ferramenta, o carregamento excessivo de uma máquina e temperaturas críticas no processo, entre outros. A resposta a determinados estados deve então ser tomada via um ciclo de controle interno por componentes específicos da máquina. Para que isto seja cumprido, mecanismos de resposta estão sendo integrados às máquinas que possibilitem que a máquina possa adaptar o seu funcionamento após a detecção de um erro [109].

Os objetivos básicos do trabalho da SFB368 foram aqui apresentados. Os resultados gerados serão gradualmente transferidos para a indústria. Alguns resultados, na forma de arquiteturas modulares já estão sendo aplicados na prática industrial, contribuindo para a melhoria de instalações industriais no sentido do aumento de sua autonomia, através da integração de métodos e técnicas que incorporam certa inteligência [106, 110, 111, 112, 113]. O conceito de autonomia para uma célula autônoma de produção é discutido a seguir.

4.1.2 O conceito de Autonomia

O conceito de autonomia para células autônomas de produção possui três significados distintos [103, 104, 114]:

Autonomia pela integração de novas tarefas: a autonomia de uma instalação industrial é melhorada expandindo-se as capacidades dos equipamentos presentes na planta. Desta forma, as funcionalidades que eram anteriormente executadas pelos operadores de tais dispositivos devem ser integradas e incluídas nos equipamentos. O objetivo é dar à célula autônoma de produção a habilidade de executar a tarefa de produção completa que esta deve executar a partir da matéria bruta ao produto acabado baseada apenas em dados do modelo CAD, matéria-prima, dos dados necessários do planejamento do processo e das ordens de produção. Funcionalidades que devem ser integradas aos equipamentos são: funções de planejamento, fixação e controle do processo.

Autonomia por tolerância a falhas: este tipo de autonomia pode ser visto como uma extensão ao primeiro. Um dos aspectos fundamentais de autonomia é a capacidade de um sistema de executar um processo complexo de maneira autônoma em combinação com estratégias de tolerância a falhas, respondendo adequadamente a situações de erro no processo, na máquina ou no planejamento.

Autonomia para o usuário: autonomia não quer dizer simplesmente a substituição do usuário da máquina por um dispositivo que possa operar de forma independente. Pelo contrário, o usuário da máquina é visto como um componente essencial de uma célula autônoma de produção. Uma parte bastante importante da pesquisa em células autônomas de produção versa sobre a interação do usuário com a célula. A célula autônoma de produção fornece funções poderosas para o controle de tarefas de manufatura e de sua estrutura interna. Desta forma, a interface com o usuário de uma APC deve possuir um projeto ergonômico. A máquina deve, de fato, liberar o usuário de tarefas rotineiras e de intervenções contínuas no processo produtivo, fornecendo subsídios para que este possa trabalhar de forma criativa e eficiente no planejamento e controle dos processos de forma específica. Entretanto, o usuário deve continuar tendo a possibilidade de intervir no processo a qualquer momento.

4.1.3 Requisitos para o Sistema TOOLSPY de Monitoramento do Processo de Fresamento

Atualmente, sensores inteligentes programáveis com funções de medição, aquisição de dados, processamento de dados e comunicação fornecem uma excelente plataforma para a modularização do software e dos dispostivos. A tendência é a substituição do processamento centralizado dos dados pela aquisição e processamento distribuído destes, comunicando os resultados do processamento através de algum tipo de rede para sensores e atuadores ou a uma unidade de controle [103]. Esta modularização faz com que projetos maiores se tornem mais fáceis de serem tratados e diminui o tempo para o desenvolvimento de um sistema. Além disso, novas funcionalidades são mais fáceis de serem acrescentadas ao sistema, como por exemplo a adição, remoção, configuração ou troca de sensores, uma vez que, idealmente, um protocolo de comunicação padronizado deve ser utilizado. Esta é uma característica importante de uma célula de produção.

O sistema de monitoramento do desgaste de ferramenta, bem como todos os outros sistemas de monitoramento da máquina-ferramenta, para atender a esses requisitos, deve operar de forma embarcada, em uma câmera inteligente com processamento local.

Para a definição dos requisitos do monitoramento do processo de fresamento, alguns pontos devem ser levados em conta como o usuário da máquina e o comando da máquina-ferramenta (tipicamente um comando CNC). Os requisitos são definidos da seguinte forma:

1. *Possuir uma interface padronizada para a integração em uma rede de sensores e atuadores de uma célula autônoma de produção.*

Prioridade: alta

Descrição: o sistema de monitoramento deve possuir uma interface padronizada definida pelos requisitos para a integração de sensores e atuadores de uma célula autônoma de produção [18, 115]. Isto porque esta rede é o meio pelo qual todas as informações entre o sistema de controle geral da célula e o sistema TOOLSPY serão trocadas. Por exemplo, é a partir desta rede de sensores e atuadores que o sistema TOOLSPY irá receber requisições de medição de determinadas ferramentas e irá comunicar seus resultados para o sistema de controle e para o armazenamento no sistema de banco de dados de ferramentas.

2. *Permitir a adaptação do sistema de processamento de acordo com as necessidades da aplicação.*

   Prioridade: alta

   Descrição: deve ser possível ao usuário adaptar o sistema de software às especificidades de cada aplicação. Em outras palavras, o usuário deve estar habilitado a mudar os algoritmos e ajustar os seus parâmetros. Além disso, o usuário deve poder escolher livremente entre dispositivos como câmeras e componentes de iluminação para o sistema.

3. *Avaliar os parâmetros do desgaste na face e no flanco principal da ferramenta de corte.*

   Prioridade: alta

   Descrição: o gume da ferramenta é a principal área de desgaste da mesma, uma vez que fica em contato direto com a peça durante o processo de usinagem. O gume fica na intersecção da face com o flanco principal da ferramenta. A avaliação dos parâmetros acima nestes dois planos é importante, podendo-se a partir destes parâmetros determinar o real desgaste da ferramenta.

4. *Classificar o desgaste presente na ferramenta de corte.*

   Prioridade: alta

   Descrição: a classificação do desgaste presente na ferramenta é de especial importância para corrigir falhas no processo como condições de usinagem inadequadas. Além disso, a detecção de quebras é de especial importância nesta etapa, para que peças e máquina não sejam desnecessariamente danificadas.

5. *Avaliar os seguintes parâmetros do desgaste de flanco: desgaste de flanco máximo VB_{max}, desgaste de flanco médio VB e área do desgaste A_{VB}.*

   Prioridade: alta

   Descrição: a avaliação destes parâmetros é importante para a determinação do correto estado da ferramenta de acordo com as características de cada processo. A partir desta informação é que as decisões sobre o que fazer com a ferramenta serão tomadas. Os resultados aqui obtidos serão fornecidos para o sistema de controle da planta ou processo e também armazenados em um banco de dados que será mantido com as características da ferramenta e um histórico dos resultados das medições realizadas.

6. *Determinar a presença de desgaste de cratera e seus principais parâmetros, a largura máxima e a largura mínima.*

   Prioridade: alta

   Descrição: o desgaste de cratera não aparece em todos os tipos de ferramentas de corte. Contudo, a sua detecção é importante, uma vez que é um indicativo da possibilidade de quebra da ferramenta e da eventual necessidade de alteração nas condições de usinagem.

7. *Ser integrável a uma máquina-ferramenta industrial em uso em uma célula autônoma de produção.*

   Prioridade: alta

   Descrição: para que o sistema possa ser testado em operação em uma célula autônoma de produção, é necessário que seja possível montar o sistema em uma máquina presente na célula com todos os dispositivos necessários para o seu funcionamento.

8. *Fornecer os resultados em um tempo máximo especificado.*

   Prioridade: média

   Descrição: o sistema de monitoramento do desgaste deve não somente fornecer resultados corretos, mas também deve fornecê-los em determinado intervalo de tempo. Em outras palavras, os resultados devem estar disponíveis antes de um certo limite de tempo (sistema de tempo-real), a ser estabelecido pelas necessidades temporais do sistema de controle da célula.

9. *Possuir um sistema de tratamento de falhas adequado.*

   Prioridade: média

   Descrição: o sistema de monitoramento do desgaste deve possuir uma malha de controle própria e notificar o usuário, o sistema de controle da máquina e o sistema de controle da célula quando uma falha ocorrer.

Todos estes requisitos devem ser levados em consideração quando do desenvolvimento de um sistema de monitoramento do desgaste de ferramentas de corte a ser aplicado em células autônomas de produção.

4.2 O Sistema TOOLSPY

No contexto do projeto SFB368 coube ao DAS, em parceria com o WZL, criar, desenvolver e implementar o sistema de visão para o monitoramento do desgaste de flanco de ferramentas de corte, que convencionou-se chamar de TOOLSPY. O sistema TOOLSPY é constituído por diferentes subsistemas, como pode ser visto na figura 4.2. Estes estão fortemente interligados. O seu projeto levou em consideração a metodologia proposta na seção 3.3 e os requisitos discutidos na seção 4.1.3.

A seguir, esses subsistemas são apresentados e analisados.

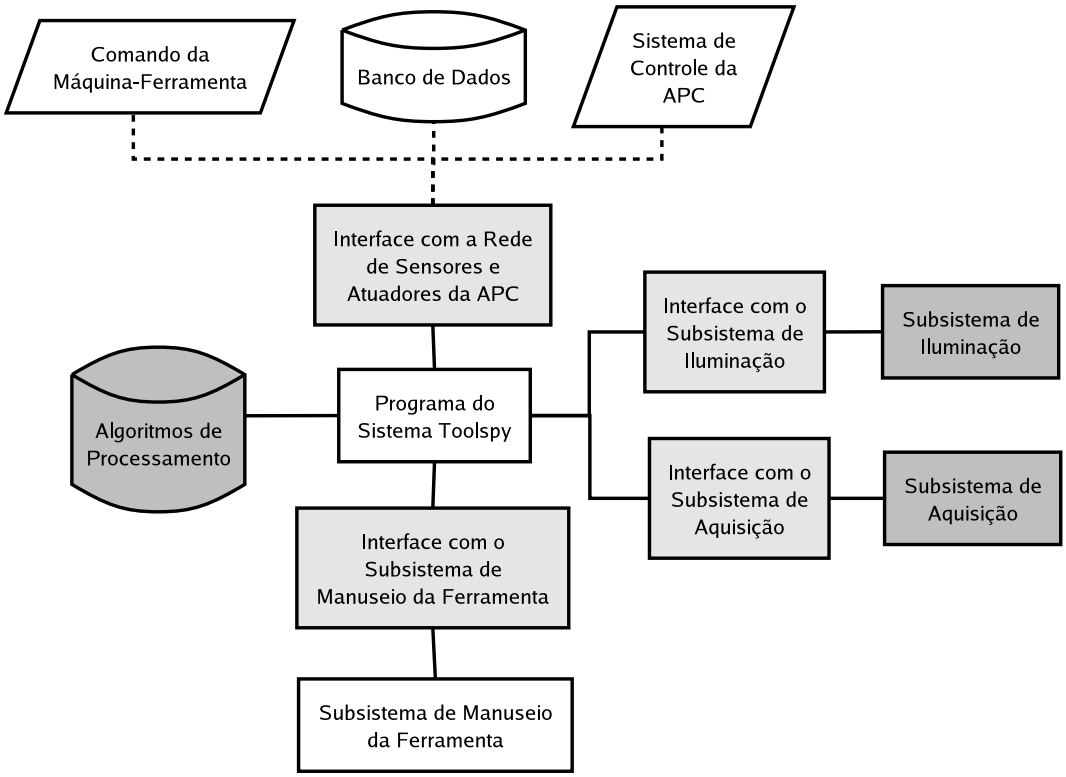


Figura 4.2: *Esquema genérico do sistema* TOOLSPY.
Observação: nesta figura, a linha tracejada representa a rede de sensores e atuadores da célula autônoma de produção. Os blocos mais escuros representam componentes da solução completa para a medição do desgaste de ferramentas sobre o qual o sistema TOOLSPY tem uma dependência direta. Interfaces são necessárias para os diferentes dispositivos.

4.2.1 Interface com a Rede de Sensores e Atuadores

A interface com a rede de sensores e atuadores será responsável pela comunicação do sistema TOOLSPY com o sistema de controle da máquina e o sistema de controle da APC. A partir desta interface é que:

- Requisições serão enviadas ao sistema TOOLSPY para a avaliação do desgaste em determinada ferramenta presente na máquina.
- O sistema TOOLSPY obterá dados necessários para o seu funcionamento a partir do sistema de banco de dados da APC.
- O sistema TOOLSPY fornecerá os seus resultados tanto para o sistema de controle da APC, para o sistema de controle da máquina e para armazenamento no sistema de banco de dados da APC.
- O sistema TOOLSPY estabelecerá comunicação com o sistema de controle da máquina de forma a identificar a ferramenta que deverá ser transportada para a estação de medição e posteriormente medida, possibilitando a correta seleção desta a partir do magazine de ferramentas da máquina.

Estas informações serão trocadas a partir dos protocolos de comunicação estabelecidos pela rede de sensores e atuadores da APC.

4.2.2 Subsistema de Manuseio da Ferramenta

O sistema de manuseio da ferramenta é o sistema responsável pela retirada da ferramenta do magazine da máquina-ferramenta, o seu transporte até a estação de medição montada dentro da máquina-ferramenta e o seu transporte e colocação de volta ao magazine quando do término da medição. A figura 4.3 ilustra o protótipo do subsistema de manuseio da ferramenta do sistema TOOLSPY, atualmente montado como um protótipo em uma máquina-ferramenta no WZL. Nesta figura, à esquerda, está o sistema de transporte e movimentação da ferramenta com 2 eixos; no centro, está a garra para apnahar as ferramentas no magazine da máquina-ferramenta; à direita, está a estação de medição com o sistema de visão. É responsabilidade do sistema TOOLSPY o acionamento deste subsistema.

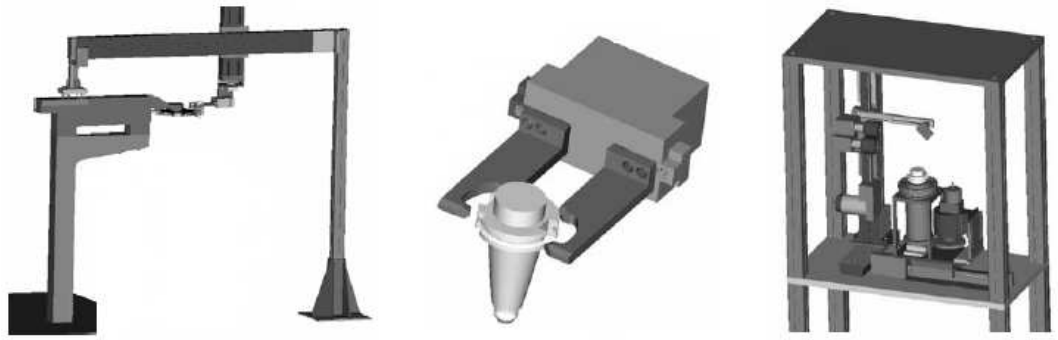


Figura 4.3: *Esquema do sistema de manuseio da ferramenta do sistema* TOOLSPY

É necessário que o sistema TOOLSPY se comunique com esta interface através de sinais que informem que a ferramenta está pronta para o transporte do magazine para a estação de medição e desta novamente de volta para o magazine.

O subsistema de transporte também é responsável, dentro da estação de medição, pela correta colocação da ferramenta em sua posição de medição, de acordo com os ângulos e a geometria de cada ferramenta.

4.2.3 Subsistema Óptico

O projeto do subsistema óptico para o sistema TOOLSPY deve levar em consideração cada ferramenta e suas dimensões. Assim, para a avaliação de diferentes tipos de ferramentas, diferentes conjuntos ópticos devem ser avaliados. Levando-se em consideração experiência com estudos anteriores, pode-se chegar aos seguintes parâmetros para o sistema:

- FOV_{min} = 2.0 *mm* x 1.5 *mm*
- FOV_{med} = 2.4 *mm* x 1.8 *mm*

Distância (mm)		**FOV**					
Mínimo	50	**H** (mm)	**V** (mm)	**H** (mm)	**V** (mm)	**H** (mm)	**V** (mm)
Máximo	200	2,0	1,5	2,4	1,8	2,8	2,1
1/4 polegada		*PMAG*		*PMAG*		*PMAG*	
		1,60		1,33		1,44	
H (mm)	3,2	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)
V (mm)	2,4	30,77	123,08	28,57	114,29	26,67	106,67
1/3 polegada		*PMAG*		*PMAG*		*PMAG*	
		2,40		2,00		1,71	
H (mm)	4,8	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)
V (mm)	3,6	35,29	141,18	33,33	133,33	31,58	126,32
1/2 polegada		*PMAG*		*PMAG*		*PMAG*	
		3,20		2,67		2,29	
H (mm)	6,4	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)
V (mm)	4,8	38,10	152,38	36,36	145,45	34,78	139,13
2/3 polegada		*PMAG*		*PMAG*		*PMAG*	
		4,40		3,67		3,14	
H (mm)	8,8	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)
V (mm)	6,6	40,74	162,96	39,29	157,14	37,93	151,72
1 polegada		*PMAG*		*PMAG*		*PMAG*	
		6,40		5,33		4,57	
H (mm)	12,8	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)	f_{min} (mm)	f_{max} (mm)
V (mm)	9,6	43,24	172,97	42,11	168,42	41,03	164,10

Tabela 4.1: *Tabela com os componentes ópticos necessários de acordo com cada parâmetro do sistema. Observação:* o tamanho do sensor está em polegadas pois é a medida padrão de especificação do mesmo.

- FOV_{max} = 2.8 *mm* x 2.1 *mm*
- WD_{min} = 50 *mm*
- WD_{max} = 200 *mm*
- DOF_{min} = 5 *mm*
- $R_{vertical}$ = 20 μm

A tabela 4.1 fornece os valores do aumento, foco mínimo e foco máximo, com os valores anteriores, para diferentes tamanhos de sensores.

A resolução é determinada a partir da faixa de valores a serem medidos pelo sistema, especificada entre 200 e 600 μm, a uma incerteza de 10%.

Com estes parâmetros em mão, pode-se calcular o tamanho do sensor necessário para a aplicação a partir das equações 3.4 e 3.5. Chega-se então a um valor de 1024x768 pixels para o tamanho dos sensores.

A tabela 4.1 ilustra os diferentes tipos de lentes necessários para cada combinação de parâmetros do sistema.

As placas de aquisição de imagens existentes no mercado em geral possuem especificações técnicas bastante boas para emprego no sistema TOOLSPY. Entre essas especificações, pode-se citar:

- Taxa de aquisição não inferior a 20 quadros por segundo em um canal, 10 quadros por segundo em dois canais.
- Aquisição de imagens a 1024x768 pixels.
- Aquisição de imagens com profundidade de pixels de 8 bits (escala de níveis de cinza de 256 tonalidades).

4.2.4 Subsistema de Iluminação

O maior problema levantado quando da seleção de um subsistema de iluminação adequado para o sistema TOOLSPY foi a intensa reflexão da luz emitida pela fonte de iluminação pela superfície da ferramenta de corte, dada a grande refletividade dos metais. Para isso, uma técnica de pré-processamento da imagem foi proposta e incorporada à cadeia de processamento de imagens do sistema TOOLSPY de forma a minimizar essas reflexões. Esta técnica de pré-processamento das imagens utiliza a iluminação da ferramenta em diferentes ângulos.

O subsistema de iluminação do sistema TOOLSPY é formado por anéis de LEDs montados em uma cúpula, como pode ser visto na figura 4.4, que foi o primeiro protótipo de iluminação montado. O problema deste protótipo era o seu tamanho, bastante grande para que pudesse ser montado em uma estação de medição compacta ao lado de uma máquina-ferramenta de uso industrial.

Figura 4.4: *Protótipo do primeiro subsistema de iluminação desenvolvido para o sistema* TOOLSPY.

O segundo protótipo de iluminação montado, já com duas câmeras, pode ser visto na figura 4.5. Este protótipo também foi montado na estação de medição acoplada à máquina-ferramenta montada no WZL. Contudo, os resultados com este protótipo não foram satisfatórios, uma vez que a dispersão da luz emitida pelos LEDs que consituíam esta solução era grande. Pouca definição era conseguida com esta estratégia.

Ambos os protótipos anteriores exigiam a utilização de uma placa de aquisição de dados específica para a sua utilização, baseando o seu funcionamento fortemente no desempenho desta placa. O terceiro protótipo de iluminação está ilustrado na figura 4.6 e tentou solucionar os problemas dos dois protótipos anteriores, sendo mais compacto e com a luz concentrada, podendo ser acionado a partir

Figura 4.5: *Protótipo do segundo subsistema de iluminação desenvolvido para o sistema* TOOLSPY.

de uma interface própria via a porta paralela. Este foi o protótipo utilizado para os testes do sistema no Brasil.

Figura 4.6: *Protótipo do terceiro subsistema de iluminação desenvolvido para o sistema* TOOLSPY.

A idéia com esta estrutura é possibilitar a implementação do algoritmo proposto por Pfeifer e Wiegers [8], tornando possível que com diferentes padrões de iluminação (ou seja, variável número de LEDs em diferentes posições) sejam acessados em momentos diferentes. Uma imagem é capturada com cada tipo de iluminação e posteriormente processada de forma a minimizar o efeito das reflexões. Maiores detalhes sobre esse algoritmo podem ser vistos na seção 5.1.2.

Estes subsistemas foram desenvolvidos de forma a permitir o seu controle através da porta paralela de um computador tipo PC.

4.2.5 Cadeia de Processamento de Imagens

A cadeia de processamento de imagens do sistema TOOLSPY é a parte do sistema responsável pela realização do subsistema de processamento de um sistema de visão como proposto no capítulo 3. Ela é composta de diferentes etapas, como pode ser visto simplificadamente na figura 4.7.

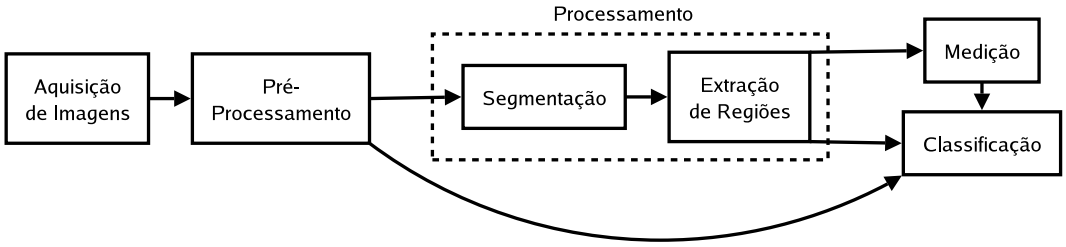


Figura 4.7: *Esquema simplificado da cadeia de processamento de imagens do sistema* TOOLSPY*, ilustrando suas principais etapas.*

Uma discussão aprofundada desta cadeia de processamento de imagens é feita no capítulo 5.

4.3 A API IPFRAMEWORK

A idéia para a construção de uma API para o sistema TOOLSPY surgiu a partir das seguintes observações, obtidas com a experiência na programação de sistemas de visão e análise de bibliotecas de processamento de imagens e visão computacional para o uso em aplicações de diferentes tipos:

- Em geral, a construção de uma aplicação envolvendo funcionalidades de processamento de imagens e visão computacional é extremamente complicada, visto a heterogeneidade de tipos de dados e algoritmos envolvidos na solução e a preocupação do programador com a eficiência dos mesmos.
- A avaliação e validação das aplicações é também difícil, visto que não existe uma estrutura padronizada para a implementação de sistemas de visão.
- A reutilização de componentes não é fácil. Em geral, integração de diferentes bibliotecas com tipos de dados diferentes em uma determinada aplicação complicam ainda mais este fato, visto que muitas vezes os tipos de dados e algoritmos são adaptados de forma a servirem apenas a uma aplicação específica. Desta forma, seria necessário:
  - Uma interface unificada para os algoritmos.
  - Uma interface unificada para as estruturas de dados.
  - Uma interface unificada para os parâmetros, permitindo que estes fossem também otimizados de alguma forma.

Orth [17] desenvolveu o seu sistema tomando por base a biblioteca de processamento de imagens e visão computacional OpenCV, que teve seu desenvolvimento iniciado pela empresa Intel. Esta

biblioteca é uma biblioteca de software livre. Contudo, as funcionalidades desta biblioteca foram agrupadas sob demanda, visto que o nível de agrupamento destas em uma interface padrão ainda era baixo. Para que o sistema TOOLSPY possa ser mais facilmente implementado em diferentes combinações de arquiteturas de computadores e sistemas operacionais e para que os testes do sistema sejam conduzidos de forma mais fácil, mostrou-se necessário o desenvolvimento de uma plataforma padronizada.

Em seguida, uma discussão acerca de algumas tentativas para a padronização de sistemas de visão é feita, fornecendo-se assim o fundamento necessário para o desenvolvimento de uma arquitetura própria para sistemas de visão.

4.3.1 Fundamentos

A idéia de uma arquitetura padronizada para a programação de sistemas envolvendo processamento de imagens e visão computacional não é nova. Carlsen e Haaks [116] descrevem, em seu trabalho, uma arquitetura orientada a objetos para a construção de sistemas de visão chamada de *Iconic Kernel System – IKS*. A preocupação, neste sistema, fica no detalhe da representação dos tipos de dados, operadores, objetos de comunicação e objetos de acesso a dispositivos. A figura 4.8 ilustra esta primeira tentativa de padronização.

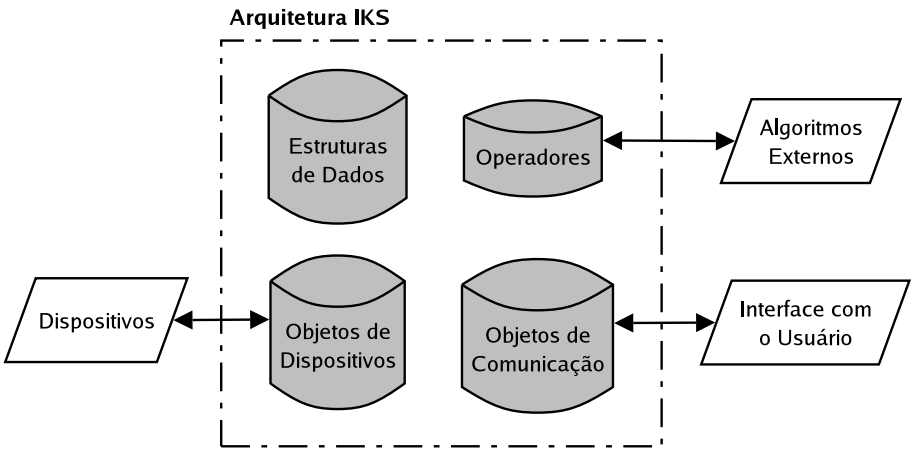


Figura 4.8: *Arquitetura IKS para o software de sistemas de visão computacional.*

No trabalho de Köthe [117], o estado dos atuais sistemas de visão computacional é discutido e analisado. O principal problema levantado é o fato de na área de visão computacional e processamento de imagens, o desenvolvimento de software reutilizável não ser encorajado. Dois motivos principais são identificados como causas desse fato:

Desempenho *versus* flexibilidade: em muitos casos, a flexibilidade como requisito de um sistema introduz um *overhead* em tempo de execução que acaba por degradar o desempenho. Uma vez que muitas aplicações de visão computacional requerem que o sistema processe uma quantidade bastante grande de dados, há um limite bastante pequeno da quantidade de processamento que pode ser sacrificada por flexibilidade.

Acoplamento de interfaces: software para visão computacional é em geral distribuído como parte de arquiteturas mais maduras e estabelecidas como padrões de uso, não permitindo que o seu código seja mesclado com o de outra arquitetura, por exemplo.

Para sanar essas deficiências, um modelo baseado em programação genérica, iteradores, *functors*[1] e tipos de dados genéricos é extensivamente apresentado. Contudo, não se chega a detalhar uma proposta para padronizar a arquitetura do sistema. Para maiores detalhes, recomenda-se ler o artigo de Köthe [117].

Paulus et al. [94] também analisam o fluxo de dados em um sistema de visão computacional e o encadeamento necessário para o processamento desses dados a partir da definição do problema geral de processamento de imagens como sendo a determinação da melhor descrição de uma imagem de entrada (de acordo com o problema em questão). A figura 4.9 ilustra esquematicamente este modelo, colocando ênfase na interação e troca de informações entre os diferentes módulos.

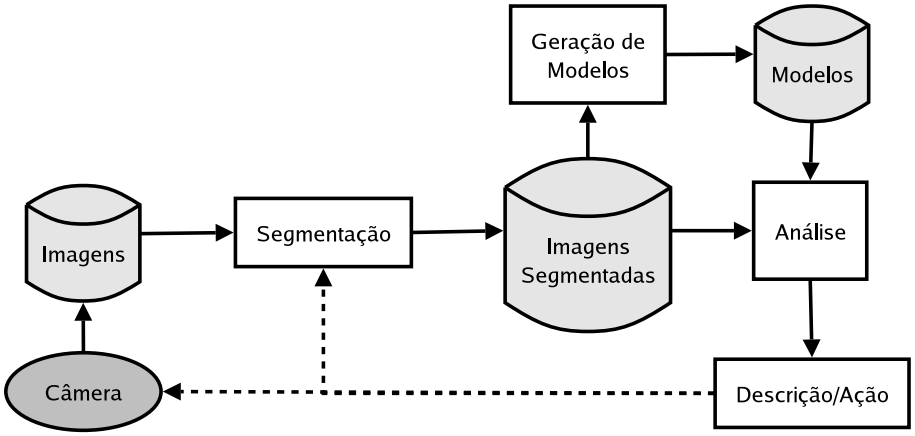


Figura 4.9: *Fluxo de dados em um sistema de processamento de imagens e visão computacional.*

Paulus et al. [94] vão além ao sugerir três hierarquias de classes diferentes para a biblioteca de software de sistemas de visão. Cada uma dessas hierarquias visa representar um dos aspectos do sistema: estruturas de dados, operadores e algoritmos de otimização. As figuras 4.10, 4.11 e 4.12 mostram essas hierarquias.

A hierarquia de estruturas de dados busca uma forma de representar os diferentes tipos de dados que são as entradas e os resultados de operações de processamento de imagens e visão computacional. Através dessa representação, uma forma padronizada de se tratar esses dados é possível, facilitando a manipulação desses diferentes tipos de dados.

A hierarquia de classes de operadores busca padronizar os operadores de forma que cadeias de processamento de imagens e visão computacional possam ser facilmente especificadas. Blocos padrões auxiliam na interface com o projetista e com o usuário do sistema.

[1] Um *functor* é um tipo de abstração de dados que permite a comunicação entre objetos de forma desacoplada. Para mais detalhes, ver os livros de Alexandrescu [118] e Gamma et al. [119].

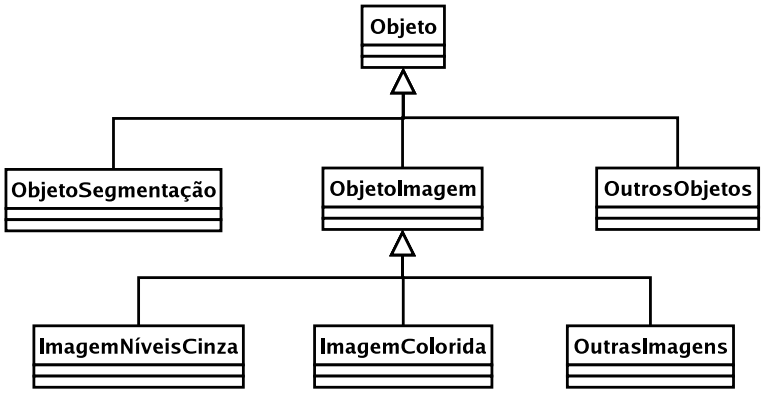


Figura 4.10: *Hierarquia de classes para as estruturas de dados de um sistema de processamento de imagens e visão computacional.*

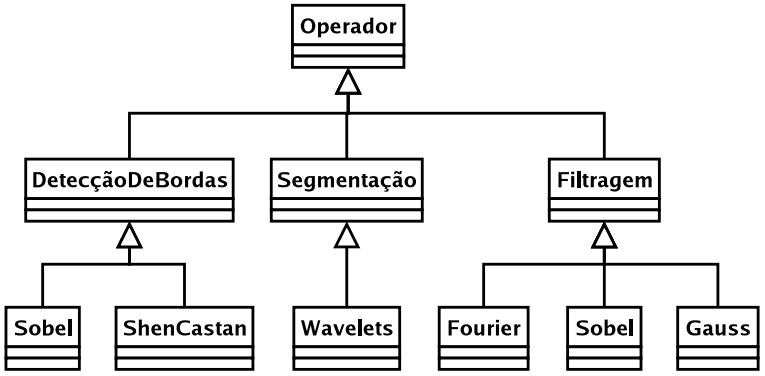


Figura 4.11: *Hierarquia de classes para os operadores de um sistema de processamento de imagens e visão computacional.*

A hierarquia de classes de algoritmos de otimização busca operacionalizar de uma forma padrão tanto os parâmetros quanto a função de avaliação a ser otimizada. A partir dessas classes, diferentes algoritmos de otimização podem ser implementados pelo sistema e os parâmetros podem ser facilmente otimizados, bastando para isso que a função objetivo, juntamente com as restrições a esta sejam fornecidas.

Percebe-se, nessa hierarquia, que os elementos fundamentais para o funcionamento do sistema são a definição de uma hierarquia de operadores e de objetos a serem manipulados pelo sistema de processamento de imagens e visão computacional. Além disso, apesar desta hierarquia não especificar, um outro ponto chave no funcionamento do sistema é a necessidade de padronização do acesso aos parâmetros dos algoritmos de forma que estes possam facilmente serem otimizados pelo sistema.

4.3.2 Requisitos da API IPFRAMEWORK

Os requisitos da API IPFRAMEWORK, juntamente com a análise dos mesmos estão mostrados abaixo:

1. *Oferecer uma interface flexível e um sistema modular para a criação de diferentes arquiteturas de sistemas de visão.*

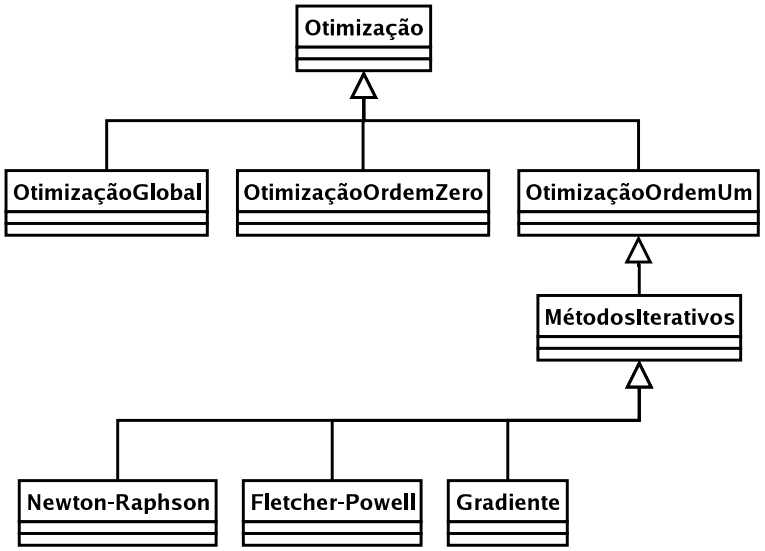


Figura 4.12: *Hierarquia de classes para os algoritmos de otimização de um sistema de visão computacional.*

Prioridade: alta

Descrição: a criação de uma arquitetura modularizada é importante para que seja atendido o requisito de flexibilidade do sistema. Assim, novos algoritmos de processamento de imagens e visão computacional podem ser inseridos na cadeia de processamento de imagens do sistema e testados de acordo com a necessidade. Uma vez que a possibilidade da necessidade de alterações no sistema para a operação com diferentes tipos de ferramenta é grande, isto é de extrema importância para o sistema. Além disso, a modularidade facilita a integração do sistema com dispositivos como placas de aquisição de imagens e placas de entrada e saída digital de dados de diferentes fabricantes, bastando que, para isso, uma interface adequada seja criada para tal finalidade.

2. *Permitir o encapsulamento de algoritmos em estruturas especializadas para a manipulação dos mesmos.*

   Prioridade: alta

   Descrição: o encapsulamento de algoritmos em estruturas especializadas permite que estes sejam especificados e programados para posterior utilização de forma transparente ao usuário. Este encapsulamento deve permitir a posterior produção de resultados em estruturas especializadas e encapsular os parâmetros do algoritmo permitindo uma posterior otimização.

3. *Permitir a criação, inserção e exclusão de elementos da estrutura em pontos específicos de forma transparente para o usuário.*

   Prioridade: alta

   Descrição: a fim de que uma determinada cadeia de processamento de imagens e visão computacional criada com a API seja utilizada para testes de diferentes algoritmos com o sistema, é necessário que os algoritmos possam ser alterados de forma transparente para

o usuário, possibilitando que este conheça apenas a posição em que deseja inserir o algoritmo na cadeia.

4. *Permitir alteração da localização de elementos da estrutura de forma transparente ao usuário.*

   Prioridade: alta

   Descrição: também a fim de que determinada cadeia de processamento de imagens e visão computacional criada com a API seja utilizada para testes, a possibilidade de se mudar a posição de um dos algoritmos no sistema é de extrema importância para que testes possam ser executados.

5. *Oferecer um mecanismo de encapsulamento dos dados que são processados em um sistema de visão.*

   Prioridade: alta

   Descrição: dados de diferentes tipos são utilizados em uma cadeia de processamento de imagens e visão computacional: imagens de diferentes formatos, listas com os contornos encontrados em uma imagem, resultados de processamento de mais alto nível como o reconhecimento de um tipo de desgaste, entre outros.

6. *Oferecer um mecanismo de encapsulamento dos parâmetros de diferentes tipos de cada algoritmo da estrutura.*

   Prioridade: alta

   Descrição: diferentes algoritmos de processamento de imagens possuem diferentes tipos de parâmetros que regulam o seu funcionamento. De forma a que este funcionamento seja ajustado de acordo com cada aplicação (ou com a aplicação do sistema TOOLSPY a diferentes tipos de ferramenta sob diferentes condições), é necessário que estes parâmetros possam ser ajustados.

7. *Permitir o acesso e edição dos parâmetros dos elementos da estrutura.*

   Prioridade: alta

   Descrição: para que os parâmetros possam ser ajustados pelo usuário do sistema, pelo próprio programa do sistema TOOLSPY, pelos próprios algoritmos de processamento e pelos algoritmos de otimização implementados é necessário que estes estejam disponíveis para edição.

8. *Oferecer a possibilidade de se executar as etapas de processamento encapsuladas desejadas de forma transparente ao usuário.*

   Prioridade: alta

   Descrição: de forma a possibilitar o correto funcionamento do sistema, diferentes cadeias de processamento de imagem e visão computacional serão especificadas através desta API. Para que estas cadeias de processamento de imagens possam ser executadas de forma transparente ao usuário, é necessário que a implementação de cada algoritmo leve em

consideração e tenha controle sobre suas entradas e saídas e seus parâmetros e implemente a sua execução sem que o usuário precise se preocupar com os detalhes de implementação de cada um dos algoritmos.

4.3.3 Projeto da API IPFRAMEWORK

O projeto da API levou em consideração o fato de uma cadeia de processamento de imagens ser na verdade um grafo, formado por diferentes etapas e responsável pelo comando dos diferentes componentes do sistema, como por exemplo aqueles ilustrados na seção 3.2.

O primeiro bloco do sistema geralmente é um bloco para a aquisição das imagens. Ele realiza a interface com algum dispositivo, na forma mais básica, como uma placa de aquisição de imagens, mas também pode realizar a interface com outros dispositivos, como por exemplo placas de entrada e saída para o controle de dispositivos de iluminação e também motores para a movimentação e transporte das peças ou objetos cuja imagem se deseja adquirir. Além disso, esse bloco pode também fazer a leitura de uma imagem ou seqüência de imagens em disco. Para isto, ele deve implementar rotinas de leitura de diferentes formatos e tipos de imagens e de seqüências de imagens.

Os blocos internos da cadeia implementam a funcionalidade de algoritmos da cadeia de processamento de imagens e visão computacional. Esta funcionalidade depende, para sua correta operação, de parâmetros definidos do sistema que devem ser especificados pelo usuário ou pelo programa através de algum mecanismo de otimização.

A primeira versão da solução (chamada de IPFRAMEWORK) foi na forma de uma estrutura em cadeia para o processamento de imagens, em que altoritmos de processamento se sucediam nesta cadeia, sendo que resultados do processamento eram armazenados em uma estrutura projetada para esta finalidade. Cada vez que um algoritmo de processamento de imagens era executado, com esta estrutura, ele tipicamente retirava as informações a serem processadas desta estrutura de armazenamento, utilizava esta informação para o processamento de alguns dados para, em seguida, armazenar os resultados de seu processamento novamente nesta estrutura. Além disso, uma estrutura padrão para os parâmetros de cada algoritmo de processamento foi criada de forma a permitir uma interface padronizada de acesso a estes, possibilitando sua futura otimização através de um método previamente estipulado (como algoritmos genéticos paralelos, por exemplo [120]). A esta arquitetura será referido como **MtqIPFramework** ou **IPFramework** versão 1.

A segunda versão da solução (chamada de **S2iIPFramework**) foi um aprimoramento da primeira versão, construída com a mesma estrutura básica da primeira. Aprimoramentos foram feitos no sentido de melhorar a interface com o usuário, facilitando o acesso aos dados do sistema. Esta versão será chamada de **S2iIPFramework** ou **IPFramework** versão 2.

A terceira versão da solução (chamada então de IPFRAMEWORK) foi totalmente modificada. Construída a partir do conceito de que uma cadeia de processamento de imagens possui suas entradas e saídas interligadas de maneira aleatória, utilizou o conceito de que esta cadeia representa uma rede como em um grafo direcionado.

4.3.4 Interfaces da API IPFRAMEWORK

A maneira de utilizar a terceira versão da API está documentada nos diagramas de seqüências UML nas figuras A.6, A.7 e A.8. O fluxograma da figura 4.13 resume o uso da API criada.

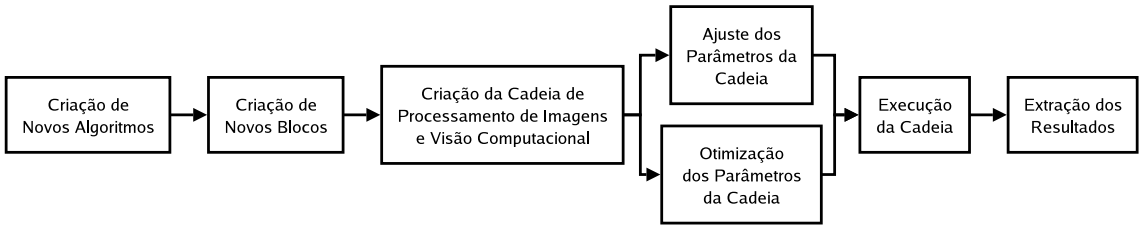


Figura 4.13: *Fluxograma ilustrando o uso da API* IPFRAMEWORK *para a criação e uso de uma cadeia de processamento de imagens e visão computacional.*

Em suma, o uso da cadeia se resume à criação de novos blocos para uma cadeia de processamento de imagens e visão computacional através da derivação a partir da classe **IPFBlock** e do algoritmo programado. A partir daí a cadeia deve ser montada de forma a que as ligações entre os diversos blocos seja feita e os parâmetros de cada algoritmo sejam especificados. A cadeia deve então ser executada e os resultados obtidos com esta podem ser obtidos e analisados.

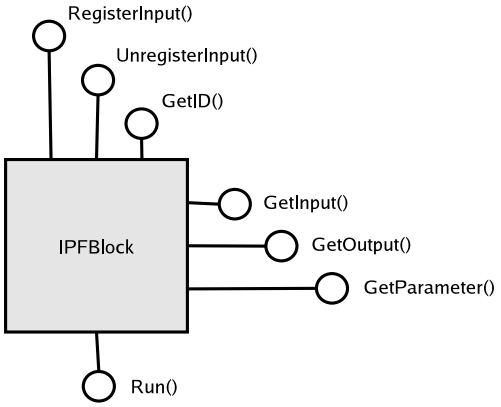


Figura 4.14: *Interface da classe IPFBlock, para a criação de algoritmos da cadeia de processamento de imagens e visão computacional.*

A classe **IPFBlock** encapsula os principais atributos e métodos de um algoritmo de processamento de imagens para a cadeia de processamento de imagens e visão computacional. Esta é uma classe abstrata que deve servir como a classe base para os demais algoritmos de processamento de imagens. Exemplos da criação dos blocos básicos da cadeia de processamento de imagens e visão computacional do sistema TOOLSPY serão dados posteriormente no capítulo 6. A idéia aqui é que apenas o construtor do bloco responsável pelo algoritmo especializado e a principal função, a de execução do algoritmo (função *Run()*) sejam redefinidas de forma a refletir os parâmetros e fluxo de execução do algoritmo. As interfaces desta classe estão mostradas na figura 4.14.

A classe **IPFData** permite a criação de diferentes objetos com tipos de dados diferentes. Uma instância desta classe de acordo com o tipo do objeto a ser manipulado deve ser criada para cada saída da classe especializada para o algoritmo a partir da classe IPFBlock e armazenada no vetor de saídas

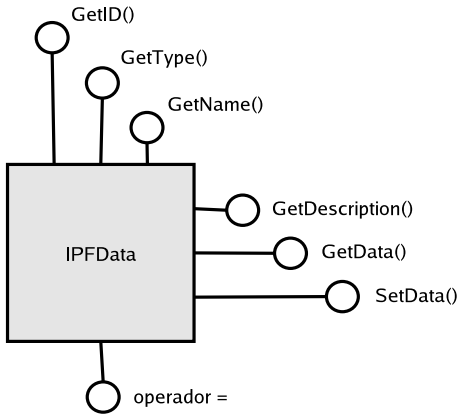


Figura 4.15: *Interface da classe IPFData, para a criação de entradas ou saídas dos algoritmos da cadeia de processamento de imagens e visão computacional.*

correspondente a este bloco. Esta tarefa deve ser feita para cada algoritmo. As interfaces desta classe estão mostradas na figura 4.15.

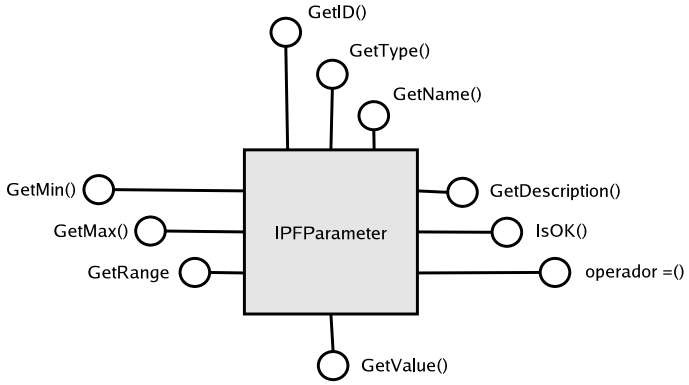


Figura 4.16: *Interface da classe IPFParameter, para a criação dos parâmetros dos algoritmos da cadeia de processamento de imagens e visão computacional.*

A classe **IPFParameter** permite a criação de objetos do tipo parâmetro de acordo com os tipos de parâmetros necessários pelo algoritmo. Aqui também cada classe especializada a partir da classe IPFBlock para cada algoritmo deve se responsabilizar pela criação dos parâmetros do tipo correto. As interfaces desta classe estão mostradas na figura 4.16.

A classe **IPFManager** é a classe a partir da qual objetos que servirão como os gerenciadores da cadeia de processamento de imagens e visão computacional serão instanciados. É nesta classe que os blocos criados para a cadeia serão armazenados, bem como todas as saídas de relevância para facilitar a visualização das mesmas. Esta classe também oferece uma forma transparente de acesso aos parâmetros dos algoritmos do sistema. As interfaces desta classe estão mostradas na figura 4.17.

A criação da cadeia de processamento de imagens completa é feita através da chamada da função *RegisterInput()* para cada bloco do sistema, ligando-se a saída dos diferentes algoritmos especializados a partir da classe **IPFBlock** com as entradas desses, montando a estrutura como se fosse um grafo direcionado. Para isso, é necessário que se obtenham as saídas dos algoritmos através da chamada da função *RegisterOutput()*

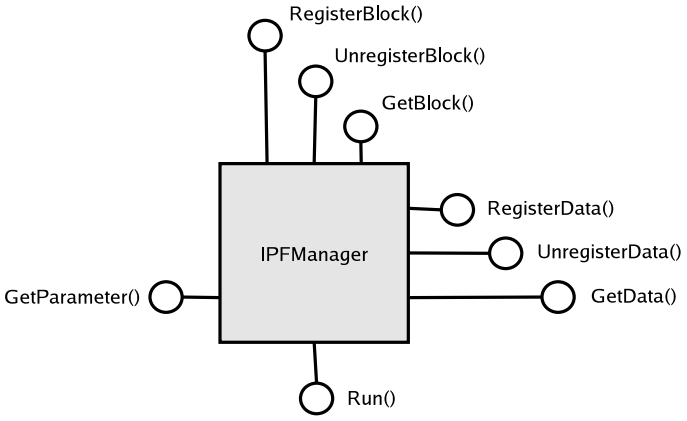


Figura 4.17: *Interface da classe IPFManager, para a criação da cadeia de processamento de imagens e visão computacional.*

A figura 4.18 ilustra o processo de criação de novos blocos para a cadeia de processamento de imagens e visão computacional utilizando a estrutura da API IPFRAMEWORK.

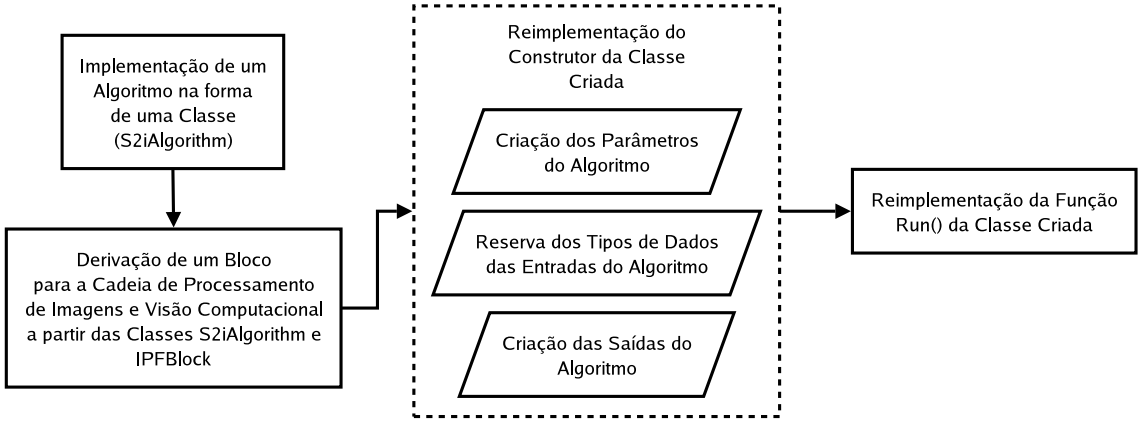


Figura 4.18: *Mecanismo para a criação dos novos blocos de processamento de imagens para a API* IPFRAMEWORK.

Neste capítulo, a estrutura física geral do sistema TOOLSPY foi apresentada, juntamente com uma breve discussão acerca da cadeia de processamento de imagens e a necessidade de uma arquitetura própria para esta cadeia. No capítulo 5 a cadeia de processamento de imagens e visão computacional que faz uso desta estrutura física e da arquitetura para o desenvolvimento de programas (API) desenvolvida será apresentada.

Capítulo 5

A Cadeia de Processamento de Imagens e Visão Computacional

Neste capítulo será apresentada a cadeia de processamento de imagens e visão computacional do sistema TOOLSPY. Em seguida, os principais módulos da biblioteca S2ILIB são apresentados e discutidos, módulos estes desenvolvidos para implementar esta cadeia de processamento.

5.1 A Cadeia de Processamento de Imagens e Visão Computacional do Sistema TOOLSPY

A figura 5.1 fornece um esquema detalhado de toda a cadeia de processamento de imagens e visão computacional do sistema TOOLSPY. Esta cadeia será detalhada a partir de cada uma de suas etapas constituintes e os passos dentro dessas etapas, sendo elas a aquisição de imagens, o pré-processamento, o processamento, a medição e a classificação (como visto na figura 4.7)[1].

5.1.1 Aquisição de Imagens

A etapa de aquisição de imagens é responsável pela aquisição das imagens que serão processadas pelo sistema para a avaliação do desgaste das ferramentas em observação. De forma geral, existem três passos de processamento distintos possíveis para esta etapa, que são o passo de recuperação de uma imagem modelo, o passo de aquisição de uma única imagem e o passo de aquisição de uma seqüência de imagens sob diferentes níveis de iluminação. Em diferentes sistemas esses blocos podem ser implementados de diferentes formas, de acordo com a arquitetura do sistema. Tipicamente na construção de um sistema com base nesta cadeia utiliza-se um passo para a recuperação da imagem modelo e um passo para a aquisição da imagem da ferramenta desgastada, sendo esta uma imagem única ou múltiplas imagens com diferentes níveis de iluminação. A aquisição pode ser feita tanto na

[1] Para alguns dos algoritmos básicos descritos neste capítulo, ver as referências na seção 3.2.5.

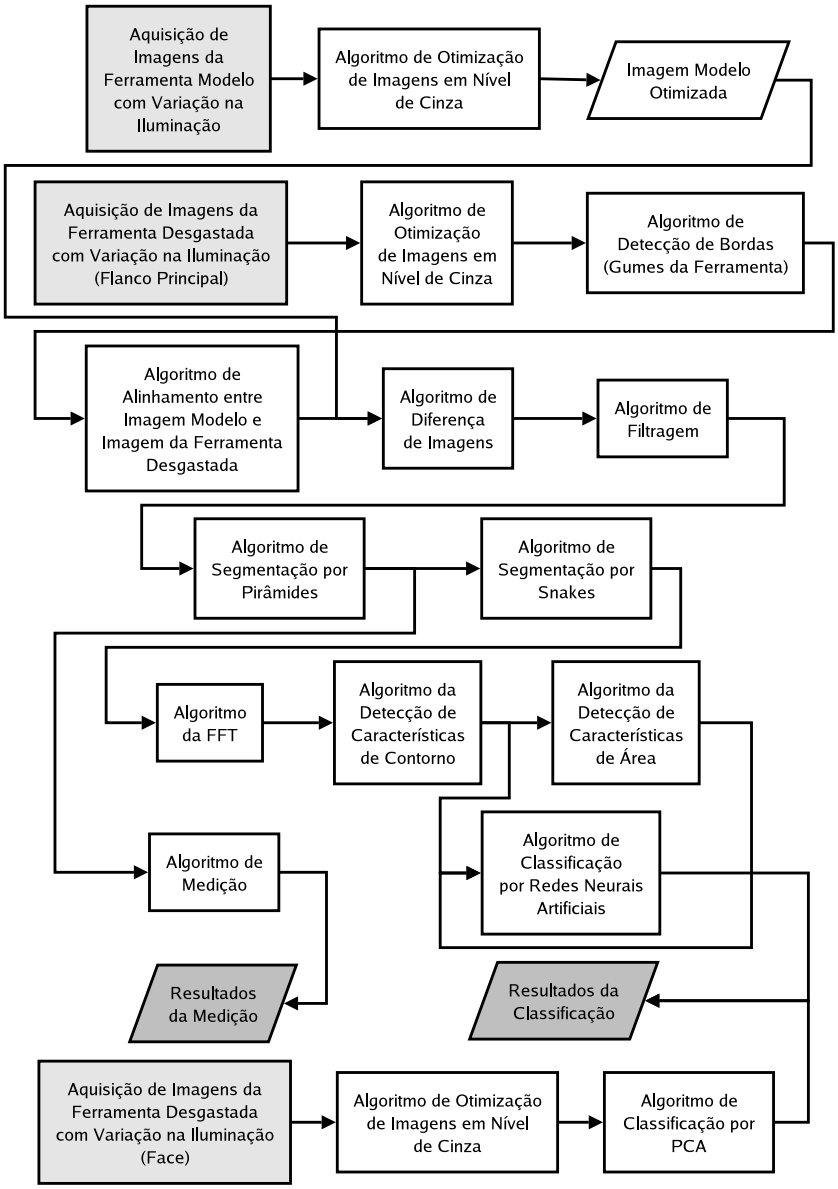


Figura 5.1: *Figura esquemática da cadeia de processamento de imagens e visão computacional do sistema* TOOLSPY.

face quanto no flanco principal da ferramenta. A figura 5.2 ilustra os principais passos da etapa de aquisição.

O passo de recuperação da imagem modelo é responsável por carregar uma imagem armazenada em disco ou em um banco de dados. Esta imagem modelo é previamente adquirida da ferramenta em seu estado não desgastado por um dos passos de aquisição das imagens e armazenada em local adequado. Dependendo do tipo de tratamento posterior, esse passo pode ser o de aquisição de uma única imagem ou o de aquisição de múltiplas imagens com diferentes níveis de iluminação. Essa imagem deve representar a ferramenta sob observação em seu estado não desgastado sendo usada, posteriormente, para a determinação da área do desgaste. No final desse passo, a imagem modelo estará corretamente armazenada em um objeto imagem da cadeia de processamento.

O passo de aquisição de múltiplas imagens é responsável pela aquisição de uma seqüência de

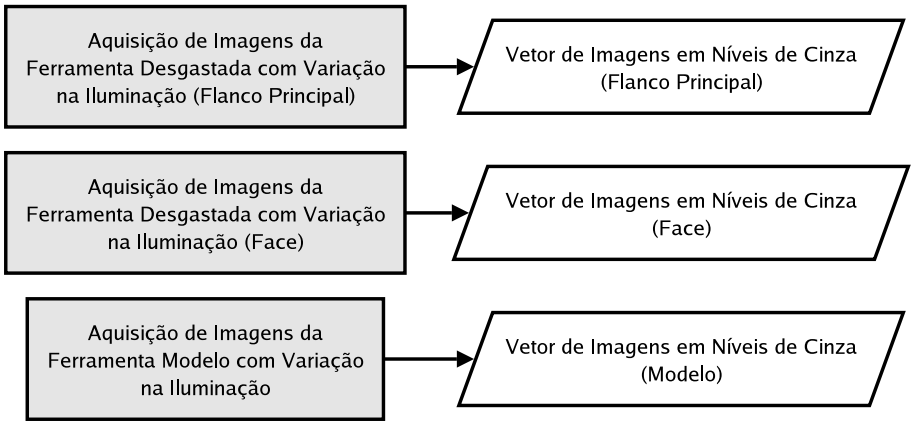
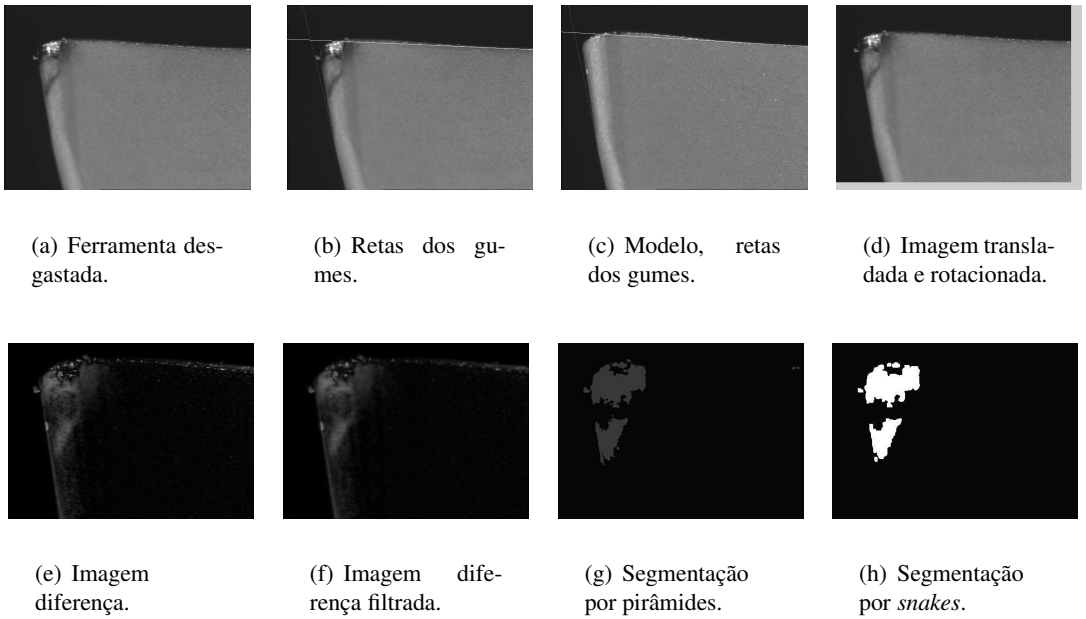


Figura 5.2: *Figura esquemática da etapa de aquisição de imagens do sistema* TOOLSPY.



(a) Ferramenta desgastada.

(b) Retas dos gumes.

(c) Modelo, retas dos gumes.

(d) Imagem transladada e rotacionada.

(e) Imagem diferença.

(f) Imagem diferença filtrada.

(g) Segmentação por pirâmides.

(h) Segmentação por *snakes*.

Figura 5.3: *Múltiplas imagens da mesma ferramenta com diferentes níveis de iluminação.*

imagens da ferramenta desgastada com níveis de iluminação diferentes. Estas imagens serão processadas na etapa seguinte pelo algoritmo de otimização de imagens de níveis de cinza. Além de ter acesso ao dispositivo de aquisição de imagens, para que este passo possa ser executado, o acesso ao dispositivo de entrada e saída para controle da iluminação também é necessário, sendo que aquisição e iluminação devem trabalhar de forma síncrona. Ao fim desse passo, um vetor de imagens em níveis de cinza contendo as imagens da ferramenta desgastada sob diferentes níveis de iluminação é gerado.

O passo de aquisição de uma única imagem é responsável pela aquisição de uma única imagem da ferramenta desgastada com um padrão determinado de iluminação. Também nesse passo é necessário acesso ao dispositivo de aquisição de imagens e ao dispositivo de iluminação. Ao final deste passo, uma imagem em níveis de cinza da ferramenta desgastada é gerada para posterior processamento.

5.1.2 Pré-Processamento das Imagens

A etapa de pré-processamento trabalha com as imagens adquiridas na etapa anterior de forma a tratá-las para que, posteriormente, a área do desgaste da ferramenta possa ser detectada.

A idéia principal nesta etapa é gerar a imagem otimizada em níveis de cinza a partir das imagens adquiridas com diferentes níveis de iluminação e do algoritmo de otimização de imagens em níveis de cinza [8] e posteriormente alinhar a imagem otimizada com a imagem modelo de forma a que a diferença entre as duas gere o menor ruído possível, destacando a área de desgaste do restante da imagem. No caso de apenas uma imagem com iluminação pré-determinada ter sido adquirida na etapa anterior, esta é usada no lugar da imagem otimizada em níveis de cinza. Após a diferença, a imagem resultante é processada de forma a se eliminar os ruídos ainda presentes, restando apenas a região desgastada. São cinco os principais passos dessa etapa: cálculo da imagem otimizada em níveis de cinza, detecção do contorno da ferramenta na imagem modelo e na imagem otimizada, alinhamento da imagem modelo e da imagem otimizada, diferença entre a imagem modelo e a imagem otimizada e filtragem da imagem resultante, como pode ser visto na figura 5.4

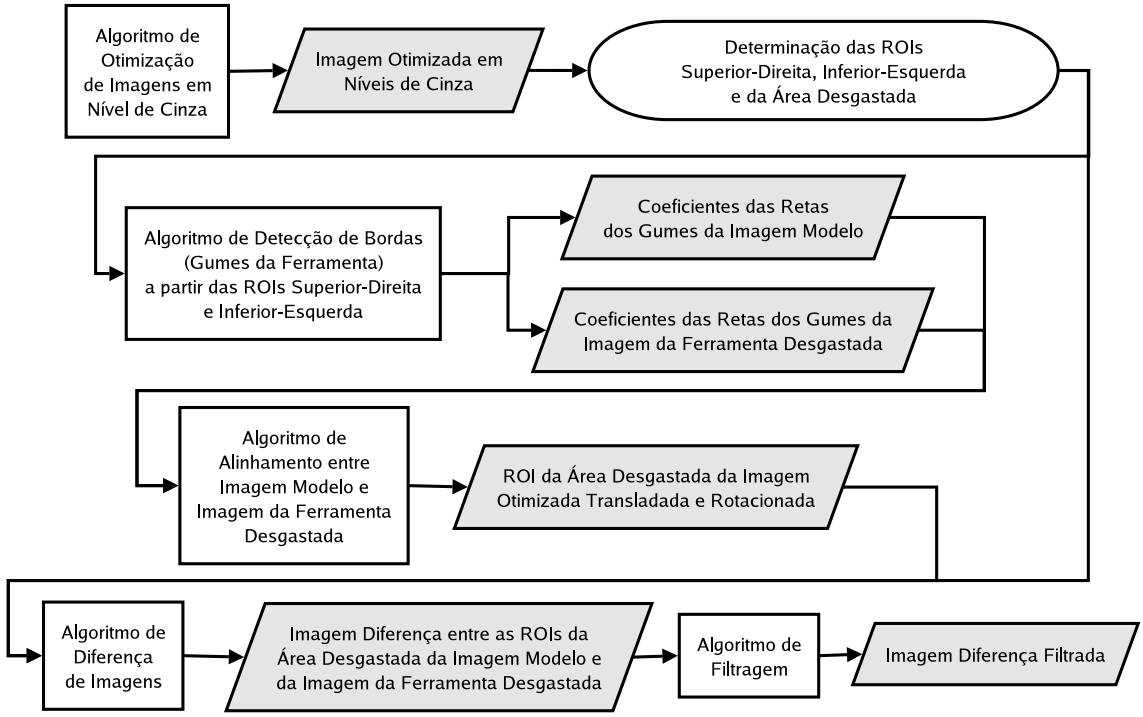


Figura 5.4: *Figura esquemática da etapa de pré-processamento das imagens do sistema* TOOLSPY.

O algoritmo de otimização de imagens em níveis de cinza funciona calculando a imagem média, pixel por pixel, de cada uma das imagens adquiridas através da variação da iluminação. Esse passo tem por objetivo a redução dos ruídos eventualmente causados pela iluminação em cada imagem adquirida, como reflexões excessivas da luz proveniente da fonte luminosa sobre a ferramenta.

O algoritmo utiliza o vetor de imagens em níveis de cinza $I(i)$, $i = 1,\ldots,n$ adquirido pelo passo de aquisição de imagens com diferentes níveis de iluminação (todas as imagens com as mesmas dimensões) e utiliza a relação da equação 5.1 para calcular cada pixel $[x,y]$ da imagem resultante R,

onde κ é o coeficiente de ajuste do otimizador ($\kappa \in [0,1]$, tipicamente, $\kappa = 1$):

$$R[x,y] = \frac{\kappa}{n} \sum_{i=1}^{n} I(i)[x,y] \tag{5.1}$$

O objetivo é aproveitar as partes melhores iluminadas de cada imagem, gerando uma imagem de iluminação mais uniforme da ferramenta, sem os efeitos de reflexões excessivas e sombras. Esta imagem gerada será posteriormente utilzada para processamento. Também a imagem da ferramenta não desgastada é adquirida desta forma para posterior comparação e a imagem resultante após o processamento com o algoritmo de otimização de imagens em níveis de cinza é a imagem modelo.

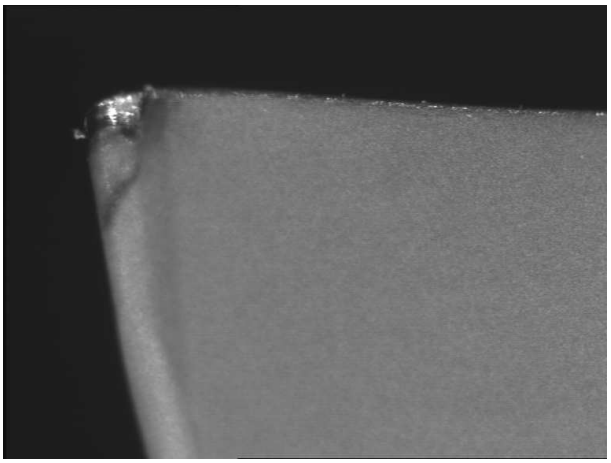


Figura 5.5: *Resultado do processamento com o algoritmo de otimização de imagens em níveis de cinza.*

O algoritmo de detecção do contorno da ferramenta é responsável pela identificação das arestas da ferramenta nas imagens otimizadas da ferramenta desgastada e não desgastada (modelo) de forma a possibilitar o alinhamento entre estas e a futura operação de diferença entre as duas imagens, no caso das imagens do flanco principal da ferramenta.

Inicialmente, as imagens das ferramentas desgastada e não desgastada são utilizadas e três regiões de interesse em cada uma destas imagens são definidas (ver figura 5.6):

ROI superior-direita: para a posterior identificação do gume principal da ferramenta.

ROI inferior-esquerda: para a posterior identificação do gume secundário da ferramenta.

ROI da região de desgaste: para a posterior segmentação da imagem e identificação da área desgastada.

A identificação dos gumes principal e secundário da ferramenta e das retas que os definem é feita a partir de uma seqüência de algoritmos de processamento de imagens nas ROIs superior-direita e inferior-esquerda, tanto para a imagem modelo da ferramenta quanto para a imagem da ferramenta desgastada. Esta seqüência funciona da seguinte forma:

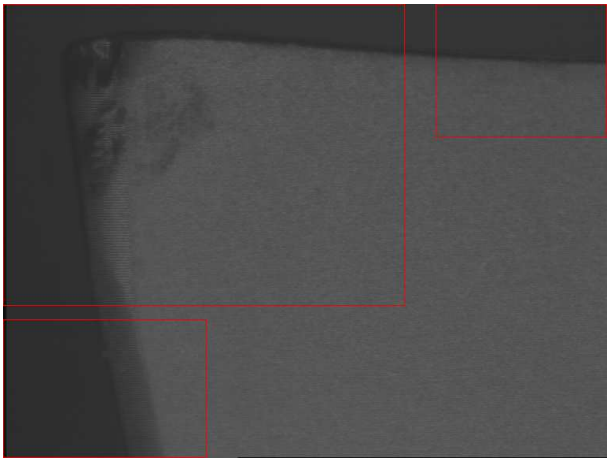


Figura 5.6: *Imagem da ferramenta ilustrando as ROIs usadas pela cadeia de processamento do sistema* TOOLSPY.

1. A imagem é binarizada através da definição de um nível adequado de *threshold* de forma a distinguir a região da ferramenta e a região do fundo da imagem.

2. Aplicação de um operador morfológico de abertura para destacar, na região da borda da ferramenta, a região do fundo da imagem (a região geralmente mais escura da imagem).

3. Aplicação de um operador morfológico de fechamento para destacar, na região da borda da ferramenta, a região da ferramenta (a região geralmente mais clara da imagem).

4. Aplicação de um operador morfológico *top-hat*, para a definição apenas do contorno que separa a região da ferramenta da região do fundo da imagem.

A partir desta imagem apenas com o contorno entre a região da ferramenta e a região do fundo da imagem, os pontos deste contorno são determinados e armazenados em um vetor de pontos. Esses pontos que definem o contorno da ferramenta em cada uma das quatro regiões de interesse das imagens (duas em cada) são processados de forma a serem interpolados e os parâmetros de cada uma das retas ($y = ax + b$) na imagem original, definidos (coeficiente angular a e coeficiente linear b).

Estes coeficientes, para cada umas das retas, são armazenados em estruturas de dados específicas para o uso posterior no passo de alinhamento das imagens.

O passo de alinhamento das imagens é responsável pela rotação e translação da imagem da ferramenta desgastada de forma a alinhar esta com a imagem modelo da ferramenta para possibilitar a operação de diferença entre imagens.

Os coeficientes das retas de cada uma das ROIs de cada uma das imagens são comparados de forma a se obter os fatores de translação e rotação da imagem. O fator de translação é obtido a partir da diferença em cada um dos eixos dos pontos de intersecção das retas que definem os gumes principal e secundário em cada uma das imagens. O fator de rotação é obtido a partir da comparação entre os coeficientes angulares das retas da mesma ROI para as imagens otimizadas da ferramenta desgastada e não desgastada. A imagem otimizada da ferramenta, proveniente do passo de otimização

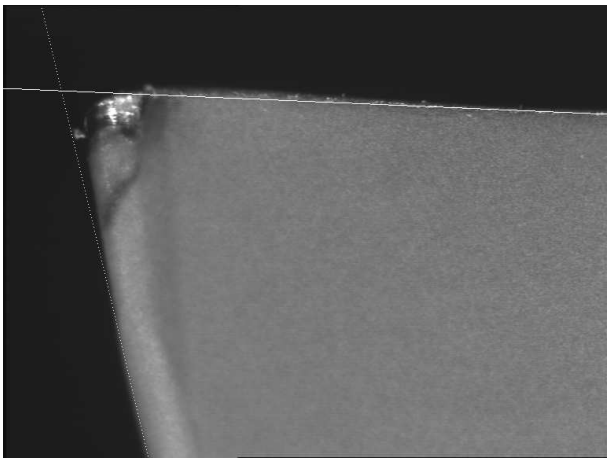


Figura 5.7: *Resultado da aplicação do algoritmo de detecção dos contornos da ferramenta para a imagem da figura 5.5.*

de imagens em níveis de cinza é então rotacionada e transladada de forma a que o posicionamento das duas ferramentas na imagem seja o mais próximo possível.

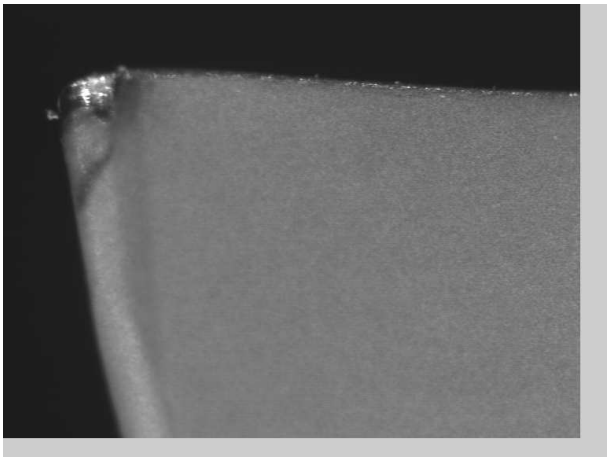


Figura 5.8: *Resultado da translação e rotação da imagem da figura 5.5 para a posição de sua imagem modelo.*

Após as duas imagens estarem alinhadas, o operador de diferença é aplicado entre a imagem otimizada da ferramenta modelo e a imagem otimizada da ferramenta desgastada apenas na ROI da região de desgaste de forma que o desgaste da ferramenta seja realçado. Esta imagem diferença é então armazenada em uma imagem separada. A aplicação deste operador de diferença corresponde ao passo de diferença de imagens.

O passo de filtragem é responsável pela eliminação de ruídos criados pela execução da operação de diferença das imagens. Este passo é necessário para a eliminação de falsas áreas de desgaste provenientes de um não totalmente correto alinhamento entre as imagens da ferramenta modelo e da ferramenta desgastada e pontos de sujeira presentes na ferramenta, em geral oriundos da utilização da ferramenta no processo de fabricação.

A aplicação de filtros é necessária para que apenas a região de desgaste se mantenha realçada. Os filtros suavizam esta região e eliminam as regiões de falso desgaste pela aplicação de operadores de

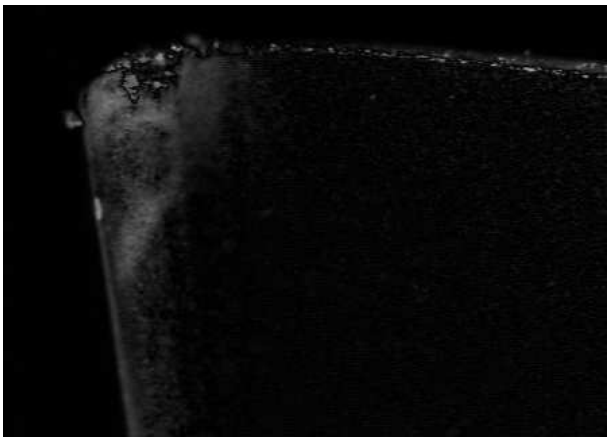


Figura 5.9: *Resultado da aplicação do operador de diferença de imagens na ROI da região de desgaste entre a imagem otimizada da ferramenta desgastada e a imagem otimizada da ferramenta modelo.*

convolução à imagem. Os filtros aplicados são mínimo 3x3, máximo 3x3 e gaussiano 5x5. Ao fim deste passo, a ROI da região de desgaste é armazenada e estará pronta para posterior processamento.

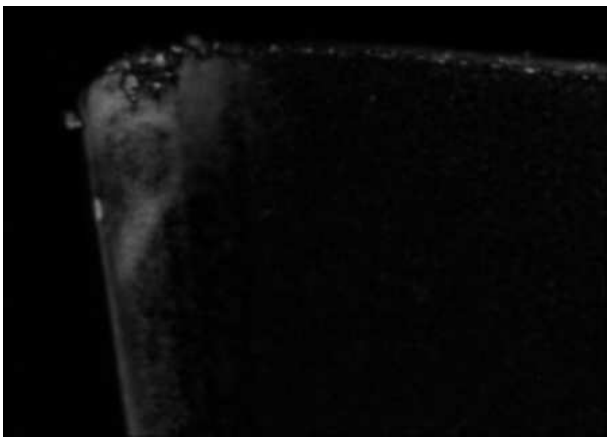


Figura 5.10: *Resultado da aplicação dos filtros à imagem diferença da figura 5.9.*

5.1.3 Processamento

A etapa de processamento é dividida em duas subetapas: a segmentação e a extração de características da região desgastada da ferramenta.

5.1.3.1 Segmentação

Esta subetapa é responsável pela delimitação da área de desgaste através da aplicação de operações de segmentação à ROI da região de desgaste proveniente da etapa anterior após a aplicação do operador de diferença na ROI da região de desgaste entre a imagem da ferramenta desgastada e a imagem da ferramenta modelo. A segmentação visa isolar a região de desgaste da região não desgastada da ferramenta e do fundo da imagem. A partir desta região de desgaste é possível se determinar o

contorno da região desgastada, que será utilizado posteriormente para a classificação do desgaste. A figura 5.11 ilustra o passo de segmentação do sistema TOOLSPY.

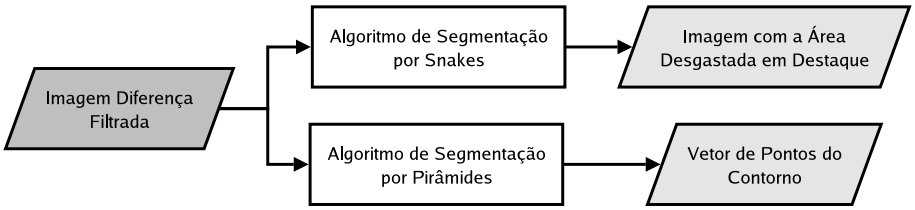


Figura 5.11: *Figura esquemática do passo de segmentação de imagens do sistema* TOOLSPY.

O passo de segmentação por pirâmides é responsável pela identificação da área de desgaste na ROI da região de desgaste a partir da imagem diferença filtrada gerada na etapa de processamento. Ela agrupa os diferentes segmentos encontrados nesta imagem em regiões comuns para especificar a área desgastada.

Neste passo, o algoritmo de segmentação por pirâmides de nível 4, 5 ou 6 é aplicado à imagem, segmentando-a e agrupando as regiões de desgaste de acordo com a intensidade em níveis de cinza das regiões segmentadas. Ruídos remanescentes também são eliminados da imagem. Esta operação deixa apenas a região de desgaste da ferramenta destacada. Ao final deste passo, a imagem segmentada é armazenada em uma estrutura adequada.

Figura 5.12: *Resultado da aplicação do operador de segmentação por pirâmides à imagem diferença filtrada da figura 5.10.*

O passo de segmentação por *snakes* é responsável pela identificação do contorno da região de desgaste. O contorno é posteriormente utilizado para a classificação do desgaste e sua medição.

Neste passo, a imagem segmentada por pirâmides da ROI da região de desgaste é utilizada, aplicando-se o algoritmo de detecção de contornos por *snakes*. Inicialmente, um conjunto de pontos é determinado englobando-se toda a ROI da região da ferramenta desgastada. A partir da determinação destes pontos, o algoritmo de *snakes* começa a funcionar, encolhendo a região do desgaste até que tais pontos entram em contato com a região de desgaste, quando estes então permanecem estáticos. No momento em que todos os pontos do algoritmo entram em um estado estático, a região do desgaste é delimitada.

A partir desta disposição de pontos em um contorno fechado tem-se a representação do contorno da região de desgaste.

Figura 5.13: *Resultado da aplicação do operador de segmentação por snakes à imagem diferença filtrada da figura 5.10.*

Para a classificação, é necessário que toda a região do desgaste seja realçada de uma maneira uniforme. Para que isso possa ser alcançado, é feita uma busca na imagem segmentada por pirâmides para o grupo de áreas de desgaste e muda-se o valor de cada um dos pixels desta região para o mais alto possível (255).

Para a medição, é necessário que o contorno da ferramenta original esteja delimitado pelas arestas da ferramenta (gume principal e secundário da imagem modelo). Para isso, o contorno é desenhado na imagem gerada pelo algoritmo de segmentação por pirâmides.

Assim são gerados aqui dois resultados. O primeiro é um conjunto de pontos que descrevem o contorno da região desgastada. O segundo é a identificação da região desgastada na ROI da região de desgaste em uma estrutura de dados própria (imagem).

5.1.3.2 Extração de Características

Este passo faz uso dos contornos gerados no passo anterior e da ROI da região de desgaste da imagem otimizada da ferramenta desgastada de forma a gerar uma série de coeficientes que irão permitir a classificação do desgaste da ferramenta. Estes coeficientes são gerados de duas maneiras diferentes: através da aplicação da Transformada Rápida de Fourier (FFT) ao contorno da região desgastada, gerado como um conjunto de pontos pelo passo de segmentação por *snakes* no passo anterior e através da análise estatística da ROI da região de desgaste da imagem otimizada da ferramenta desgastada. Esta extração de características é necessária para possibilitar o processo de classificação. A figura 5.14 ilustra a subetapa de extração de características.

Pela aplicação da FFT em um sinal periódico (o contorno fechado), uma nova representação das informações contidas no contorno, no domínio da freqüência, é alcançada.

A representação visual do contorno contém, em geral, um número muito grande de pontos. De forma a reduzir esta quantidade de informações, mantendo-se as características do contorno, muda-se

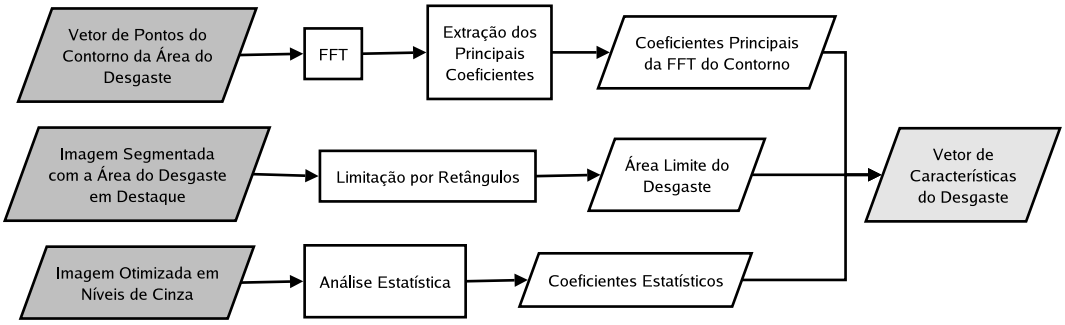


Figura 5.14: *Figura esquemática do passo de extração de características do sistema* TOOLSPY.

a representação do contorno do domínio do espaço para o domínio da freqüência. Esta mudança é possível pelo fato de o desgaste estar delimitado por uma curva fechada. Desta forma, pode-se avaliar o contorno como um sinal periódico no plano cartesiano e pode-se aplicar a FFT a este sinal.

Neste passo, o contorno da área de desgaste determinado no passo de segmentação por *snakes* e processado como um sinal periódico pelo algoritmo da FFT. Apenas os coeficientes de ordem mais baixa são armazenados, pois sabe-se que são estes que possuem a maior quantidade de informação do sinal analisado.

Ao fim deste passo, os coeficientes da aplicação da FFT são armazenados em um vetor de números do tipo ponto flutuante.

O passo de extração de características do contorno é responsável pela seleção dos descritores de Fourier na representação freqüencial do contorno que mais interessam e efetivamente importam para o processo de classificação.

Inicialmente, carrega-se o conjunto de descritores calculados no passo anterior. O primeiro descritor de Fourier está associado ao posicionamento e rotação do contorno e é eliminado. Os demais descritores são aproveitados, sendo que a ordem até a qual serão aproveitados pode ser especificada. Ao final deste passo, um vetor de números em ponto flutuante contendo os descritores de Fourier selecionados é gerado.

A extração de características da área é responsável pela extração de nova informação da superfície da área desgastada da ferramenta para auxílio na classificação do tipo de desgaste encontrado. Esta extração de características é baseada na análise do histograma da ROI da região de desgaste da imagem otimizada em níveis de cinza e também pela determinação de um retângulo que englobe esta área de desgaste a partir da ROI da região de desgaste proveniente do passo de segmentação por pirâmides.

Inicialmente, um retângulo horizontal é desenhado pela ROI da região de desgaste proveniente do passo de segmentação por pirâmides e tenta-se achar um segundo retângulo de área mínima que contenha toda a região do desgaste. Isto é feito através de sucessivas rotações na imagem, determinando-se sempre o retângulo horizontal de mínima área que engloba a região desgastada.

A partir destas coordenadas, o tamanho limite do desgaste é determinado, de forma a ser utilizado como parâmetro para a classificação do desgaste.

Na imagem otimizada em níveis de cinza, a ROI da região de desgaste é analisada de forma a se criar um histograma dos níveis de cinza. Deste histograma, informações podem ser extraídas, como a entropia da superfície da ferramenta, o contraste, valores mínimo, máximo e médio dos pixels, desvio padrão, média e variância. Estas informações são usadas para a tarefa de classificação.

Ao fim deste passo, a segunda parte do vetor para a classificação do desgaste é armazenado e colocado ao final do vetor de valores em ponto flutuante criado no passo de extração de características do contorno.

5.1.4 Medição

Esta etapa realiza a medição do desgaste da ferramenta através da análise das imagens segmentadas e dos contornos do desgaste previamente calculados nos passos anteriores. Ela determina o desgaste de flanco médio VB, o desgaste de flanco máximo VB_{max} e a área do desgaste de flanco A_{VB} para a imagem do flanco principal da ferramenta. A figura 5.15 ilustra os passos envolvidos nesta etapa.

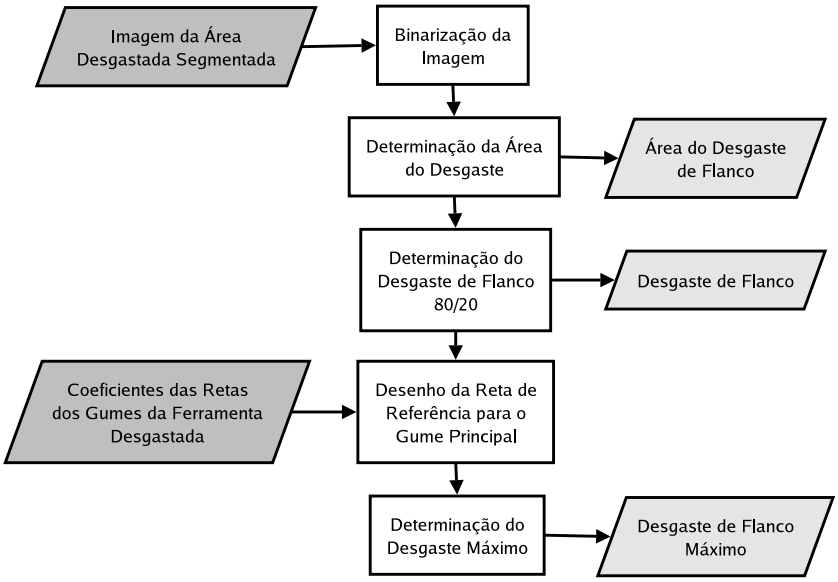


Figura 5.15: *Figura esquemática do passo de medição sistema* TOOLSPY.

Nesta etapa, a imagem do desgaste segmentada com um único rótulo é usada para a determinação das medidas necessárias.

Inicialmente, é necessário que o sistema seja calibrado para se determinar o tamanho dos pixels da imagem a ser processada. É necessário que sejam determinados a altura, a largura e a área destes pixels a partir de algum objeto cujas dimensões sejam conhecidas.

Em seguida, o máximo desgaste de flanco é avaliado determinando-se o valor da distância entre o pixel com a posição mais inferior na imagem da ROI da região de desgaste segmentada por *snakes* e a reta que define o gume principal da ferramenta modelo. A imagem é rotacionada de forma que esta

reta esteja na posição horizontal. Esta distância em pixels é determinada e multiplica-se esta distância em pixels pela taxa de calibração determinada anteriormente para a altura de cada um dos pixels.

O valor da área da região desgastada pode ser avaliado através de uma operação de contagem dos pixels dentro da região de contorno. Isto é feito na ROI da região de desgaste segmentada por pirâmides. Esta contagem de pixels é então multiplicada pela área do pixel.

Uma outra medida é realizada, a qual será denominada de desgaste de flanco 80/20 e será denotada por VB. Para o cálculo do valor deste parâmetro, é necessário dividir a área da região de desgaste em duas áreas distintas através de uma linha, com uma região superior com 80% do desgaste e uma região inferior com 20% do desgaste. O valor pode então ser determinado extraindo-se a distância, através da contagem do número de pixels, entre esta linha e o ponto limite superior, como no caso da determinação do máximo desgaste de flanco.

5.1.5 Classificação

Esta etapa é responsável pela classificação do desgaste da ferramenta a partir das informações de características do contorno do desgaste determinadas na etapa anterior. Ela faz uso da representação freqüencial anterior e utiliza a técnica de redes neurais artificiais (RNA) que é responsável pela classificação final. Também a presença de desgaste de cratera na ferramenta é avaliada através do uso da técnica de análise de componentes principais (PCA), um método de projeção para a classificação de padrões. A figura 5.16 ilustra esquematicamente os passos desta etapa.

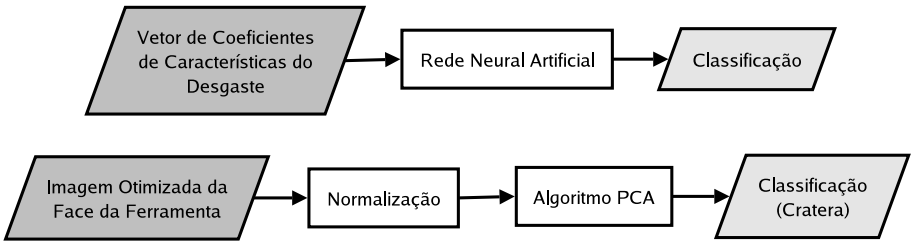


Figura 5.16: *Figura esquemática do passo de classificação das imagens do sistema* TOOLSPY.

O passo de classificação por redes neurais artificiais é responsável pela classificação do tipo de desgaste no flanco principal da ferramenta[2].

Uma rede do tipo *feedforward* é utilizada, com uma função de ativação sigmóide nos neurônios. A rede possui apenas uma camada escondida, recebendo como entrada os valores resultantes da extração das características de superfície e área da etapa anterior. A rede pode ser treinada com algoritmos como *backpropagation* e *quickpropagation*.

Para a correta operação desta etapa, a rede deve ser corretamente treinada e ter o valor dos pesos carregados. A classificação do desgaste é então gerada de acordo com os tipos previamente estipulados e treinados.

[2]Para uma maior discussão sobre RNAs, suas diferentes topologias e algoritmos de treinamento recomenda-se os livros de Freeman e Skapura [10] e Kovács [11] e o relatório de Deschamps [121].

O passo de classificação baseado em PCA é responsável pela detecção ou não do desgaste de cratera da ferramenta a partir da imagem otimizada da ferramenta desgastada normalizada[3]. Por ser um método baseado em projeções, é necessário que, incialmente, um conjunto de imagens representativas do universo a ser classificado seja fornecido de forma que os espaços de classificação para cada uma das classes deste universo possam ser calculados. A partir do cálculo destes espaços de classificação, a cada imagem recebida, esta é classificada como pertencente a um destes espaços ou a nenhum.

5.2 A Biblioteca S2iLib

A partir do ano 2001, o grupo S2i passou a desenvolver uma biblioteca para o processamento digital de sinais e imagens, com foco no desenvolvimento de soluções em sistemas de visão[4]. Esta biblioteca possui diferentes módulos, que implementam as diferentes funcionalidades necessárias para que o processamento das imagens se dê de forma transparente ao usuário. A figura 5.17 ilustra os principais módulos desta biblioteca.

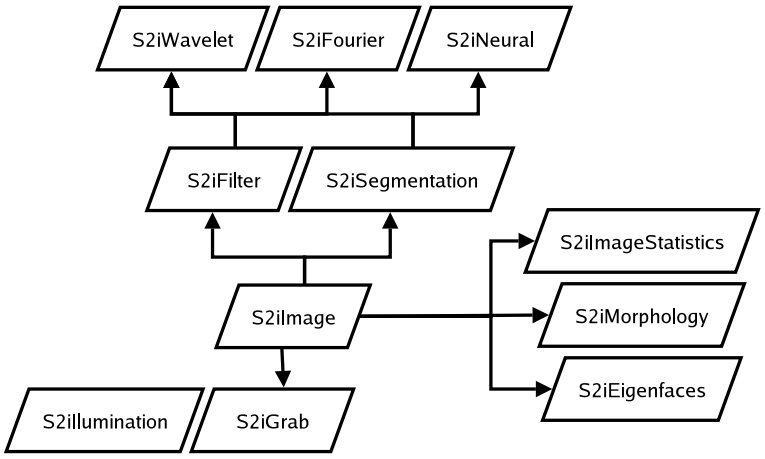


Figura 5.17: *Diagrama com os módulos da biblioteca* S2iLib.
Observação: nesta figura, cada bloco representa um componente da biblioteca S2iLib. As ligações entre os blocos representam as dependências entre eles.

A biblioteca S2iLib tem a sua estrutura baseada nas bibliotecas de software OpenCV[5] e VxL[6]. A biblioteca OpenCV (de *Open Computer Vision*) teve o seu desenvolvimento iniciado pela empresa Intel. Esta biblioteca encapsula uma grande parte de diferentes funções de processamento de imagens e visão computacional. Ela foi programada em C e é utilizada tanto para pesquisa quanto para aplicações comerciais. Em conjunto com a biblioteca IPL (*Image Processing Library*, uma biblioteca de processamento de imagens otimizada para a utilização em processadores fabricados pela Intel) fornece código bastante robusto e rápido. A VxL é um conjunto de diferentes bibliotecas para

[3]Para uma maior discussão sobre o algoritmo de PCA sugere-se os artigos de Pentland e Turk [19] e Yang et al. [122]. O artigo de Minasi et al. [123] fornece maiores detalhamentos sobre esta técnica aplicado ao problema de classificação do desgaste em ferramentas de corte.

[4]Um artigo com uma proposta inicial para a biblioteca S2iLib foi publicado por Minasi et al. [124].

[5]Para maiores informações sobre a biblioteca OpenCV, visitar o endereço http://sourceforge.net/projects/opencvlibrary.

[6]Para maiores informações sobre a biblioteca VxL, visitar o endereço http://vxl.sourceforge.net.

o processamento de imagens e visão computacional, programada em C++. É composta de diversos módulos, como, por exemplo, módulos para a representação de imagens, módulos para a análise de imagens estáticas, módulos para a análise de vídeo, entre outros. Estes módulos são independentes (com algumas exceções) e podem ser usados isoladamente.

Muitos dos módulos da S2ILIB[7] herdaram parte de suas funcionalidades de uma dessas duas bibliotecas. A implementação desses módulos possibilitou que a cadeia de processamento de imagens do sistema TOOLSPY pudesse ser implementada. Para acesso ao código-fonte desta biblioteca, ver o apêndice B.

5.2.1 S2IIMAGE

O módulo S2IIMAGE fornece as estrutura básicas para a representação de imagens, ROIs, retângulos, pontos e linhas para os demais módulos da biblioteca S2ILIB. A biblioteca S2IIMAGE se encontra atualmente em sua segunda versão, utilizando e extendendo as funcionalidades da versão 3 da biblioteca OpenCV. As principais interfaces oferecidas por este módulo da S2ILIB podem ser vistas na figura 5.18.

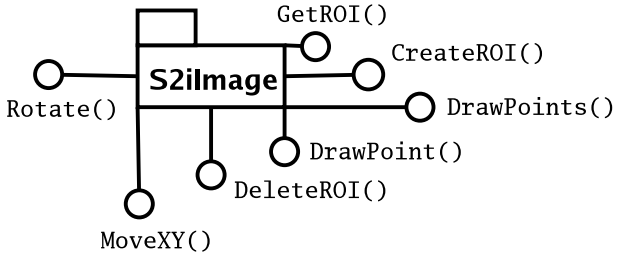


Figura 5.18: *Diagrama com as interfaces do módulo* S2IIMAGE.

Dentre as principais funcionalidades implementadas por este módulo estão operações como a manipulação de ROIs (criação, exclusão, recuperação), a rotação e a translação de imagens e a abertura e o armazenamento das imagens em arquivos em disco nos formatos BMP, JPG, TIF e PNG e a abertura e o armazenamento de seqüências de imagens em arquivos em disco a partir do formato MNG[8].

5.2.2 S2IGRAB

O módulo S2IGRAB fornece a interface com os dispositivos de aquisição de imagens possibilitando que estas possam ser adquiridas a partir de placas de aquisição, *webcams* e *scanners*. Existem duas implementações diferentes para este módulo: uma para a família de sistemas operacionais Windows e outra para o sistema operacional GNU/Linux. Para a família de sistemas operacionais Windows as funcionalidades existentes se baseiam nos *drivers* fornecidos por cada fabricante de placas (atualmente existem interfaces para a placa de aquisição Picolo Pro 2 da Euresys) e no padrão

[7] A documentação completa da S2ILIB e de seus módulos pode ser encontrada no endereço http://s2i.das.ufsc.br/docs.

[8] Maiores informações sobre o formato MNG podem ser obtidas a partir de sua especificação, disponível em http://www.libpng.org/pub/mng/mngdocs.html.

TWAIN de aquisição para *scanners* e *webcams*. Para a família de sistemas operacionais baseados no GNU/Linux as funcionalidades existentes se baseiam nas APIs Video4Linux para o acesso de dispositivos de vídeo como placas de aquisição e *webcams* e SANE para o acesso a dispositivos do tipo scanners. As principais interfaces oferecidas por este módulo podem ser vistas na figura 5.19.

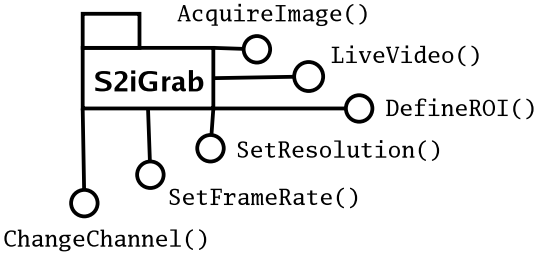


Figura 5.19: *Diagrama com as interfaces do módulo* S2iGRAB.

Dentre as principais funcionalidades oferecidas por este módulo se encontram a aquisição de imagens, a possibilidade de ajustes em parâmetros de aquisição, como brilho e contraste e a definição de regiões de interesse para a aquisição a partir da definição do tamanho do sensor, entre outros.

5.2.3 S2iILUMINATION

O módulo S2iILUMINATION é a interface da biblioteca para dispositivos de iluminação formados por conjuntos de LEDs. Atualmente, este módulo suporta o acionamento deste tipo de dispositivo a partir da porta-paralela, sendo que é usado para acionar, por exemplo, o dispositivo de iluminação mostrado na figura 4.6. As principais interfaces deste módulo estão representadas na figura 5.20.

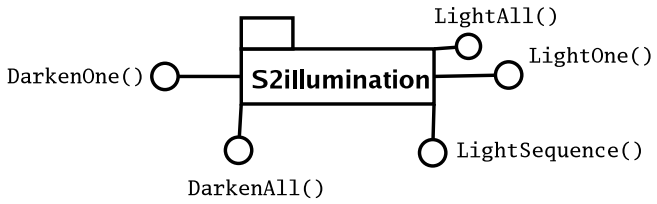


Figura 5.20: *Diagrama com as interfaces do módulo* S2iILUMINATION.

Este módulo oferece a possibilidade de acionar individidualmente cada LED que faz parte do disposisitivo de iluminação, bem como acionar apenas um conjunto deles. Ele fornece a possibilidade de se definir uma seqüência de LEDs a serem acesos de forma que o acionamento de cada uma dessas seqüências para a posterior implementação da aquisição com diferentes níveis de iluminação se torna uma questão de sincronismo.

5.2.4 S2iFOURIER

O módulo S2iFOURIER tem por objetivo o cálculo da Transformada Rápida de Fourier (FFT) de um sinal unidimensional ou bidimensional qualquer. As principais interfaces deste módulo estão representadas na figura 5.21.

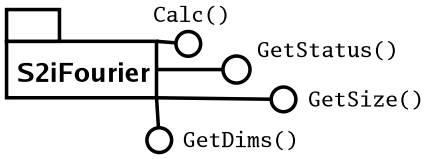


Figura 5.21: *Diagrama com as interfaces do módulo* S2IFOURIER.

Este módulo pode ser usado de diferentes maneiras[9]. Por exemplo, a FFT de um contorno pode ser calculada para posterior processamento dos coeficientes através de uma rede neural artificial; um filtro pode ser implementado de forma que freqüências determinadas presentes na imagem sejam cortadas. Até mesmo um algoritmo de segmentação a partir da FFT pode ser implementado, da mesma forma que um filtro, mas produzindo, por exemplo, uma imagem binarizada a partir de freqüências presentes na análise dos descritores.

5.2.5 S2IWAVELET

O módulo S2IWAVELET[10] é de certa forma independente do restante da S2ILIB. Ele fornece a interface para o cálculo da Transformada de *Wavelets* de um sinal unidimensional ou bidimensional qualquer, a partir de bases de *Wavelet* pré-determinadas. Este cálculo pode ser realizado tanto a partir de bancos de filtros (*Filter Banks*) como a partir de pacotes *Wavelet* (*Wavelet Packets*). As principais interfaces deste módulo podem ser vistas na figura 5.22.

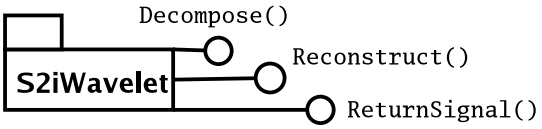


Figura 5.22: *Diagrama com as interfaces do módulo* S2IWAVELET.

Este módulo atualmente está em testes, sendo que posteriormente será aplicado ao problema de segmentação das imagens, em particular para a segmentação da região de desgaste. Dentre a principal funcionalidade deste módulo está o cálculo da transformada *Wavelet* do sinal fornecido como entrada (unidimensional ou bidimensional), sendo que a base de *Wavelets* pode ser fornecida e os parâmetros como nível de detalhe especificados.

5.2.6 S2INEURAL

O módulo S2INEURAL é também independente do restante dos outros módulos da biblioteca e pode ser usado de forma isolada. Este módulo implementa as funcionalidades básicas necessárias

[9] Maiores informações sobre o uso da FFT em processamento de imagens e visão computacional podem ser encontradas no artigo de Cabral et al. [125]

[10] Dois artigos com propostas para a implementação computacional de uma biblioteca baseada na Transformada de Wavelet foram publicados por Baldissera et al. [126, 127].

para a implementação de redes neurais artificiais. Os tipos de redes suportados são *feedforward*, *Self-Organizing Maps* (SOM) e *Basic Associative Memory*. As principais interfaces deste módulo estão representadas na figura 5.23.

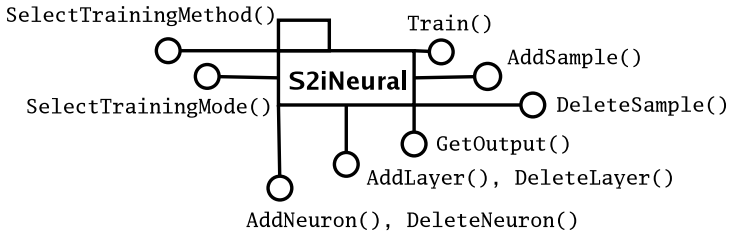


Figura 5.23: *Diagrama com as interfaces do módulo* S2INEURAL.

Pode-se especificar a topologia da rede a ser implementada, como o tipo de neurônio a ser utilizado, o número de camadas e as ligações entre as camadas em redes *feedforward*. Além disso, o algoritmo de treinamento pode ser escolhido, sendo que conjuntos de amostras de treinamento podem ser manipulados e os parâmetros do treinamento especificados. Atualmente esta biblioteca se encontra na sua versão 2. Maiores informações sobre a versão 1 da biblioteca podem ser obtidas no relatório de Deschamps [121].

5.2.7 S2IEIGENFACES

O módulo S2IEIGENFACES implementa a técnica de classificação a partir da análise de componentes principais (*Principal Component Analysis* ou PCA). As principais interfaces deste módulo estão representadas na figura 5.24.

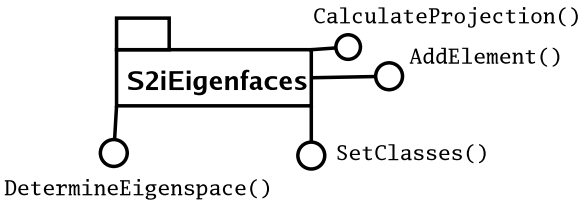


Figura 5.24: *Diagrama com as interfaces do módulo* S2IEIGENFACES.

É possível, a partir deste módulo, adicionar conjuntos de imagens de determinadas classes para o cálculo dos espaços de classificação. A partir do cálculo destes espaços, a técnica de PCA pode ser aplicada para o reconhecimento dos padrões e mesmo para o reconhecimento de faces, por exemplo.

5.2.8 S2IFILTER

O módulo S2IFILTER implementa os principais tipos de filtros para o processamento digital de imagens. Este módulo depende de objetos derivados da classe **S2iImage** do módulo S2IIMAGE. As principais interfaces deste módulo estão representadas na figura 5.25.

Um objeto da classe filtro requerida deve ser instanciado para que o operador possa ser aplicado.

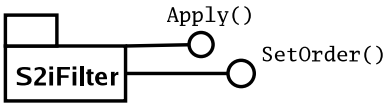


Figura 5.25: *Diagrama com as interfaces do módulo* S2IFILTER.

5.2.9 S2IMORPHOLOGY

O módulo S2IMORPHOLOGY implementa os operadores de morfologia de imagem como erosão, dilatação, abertura, fechamento, *top-hat* e outros. Ele depende do módulo S2IIMAGE pois opera sobre os objetos imagens fornecidos por esta classe. As principais interfaces deste módulo estão representadas na figura 5.26.

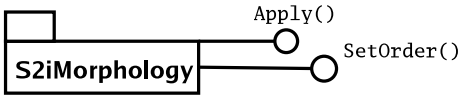


Figura 5.26: *Diagrama com as interfaces do módulo* S2IMORPHOLOGY.

Um objeto da classe do operador de morfologia requerido deve ser instanciada de forma que este possa operar sobre os objetos imagens do usuário. Estes objetos imagens devem ser registrados com a classe e sua alocação e destruição são de responsabilidade do usuário.

5.2.10 S2ISEGMENTATION

O módulo S2ISEGMENTATION fornece a interface para diferentes algoritmos de segmentação de imagens, desde simples operações de *threshold* até operações de segmentação mais sofisticadas, como a segmentação piramidal e a segmentação por *snakes* (contornos ativos). As principais interfaces deste módulo estão representadas na figura 5.27.

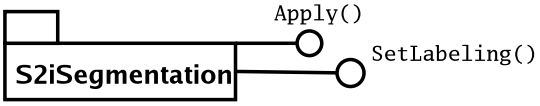


Figura 5.27: *Diagrama com as interfaces do módulo* S2ISEGMENTATION.

Um objeto da classe do operador de segmentação requerido deve ser instanciado para a aplicação deste operador a objetos derivados da classe **S2iImage** do módulo S2IIMAGE.

Capítulo 6

Análise dos Resultados Obtidos

Este capítulo inicia com a descrição das interfaces em forma de blocos para a cadeia de processamento de imagens e visão computacional descrita no capítulo 5, geradas a partir da API IPFRAMEWORK criada para este propósito e apresentada na seção 4.3. A criação dessas interfaces tem como objetivo validar a arquitetura proposta aplicando-a à cadeia de processamento do sistema TOOLSPY. O capítulo termina com os resultados obtidos pela aplicação do programa gerado com esta arquitetura à avaliação do desgaste de um determinado tipo de ferramenta como forma de se determinar o desempenho do sistema TOOLSPY, mais especificamente do conjunto formado pelo subsistema óptico, subsistema de iluminação e subsistema de processamento.

6.1 Validação da API IPFRAMEWORK

A validação da API IPFRAMEWORK se deu através da criação dos blocos básicos para a cadeia de processamento de imagens e visão computacional do sistema TOOLSPY descrita em detalhes no capítulo 5. Esta validação se deu de duas maneiras distintas:

- Para o programa do sistema TOOLSPY para o sistema operacional GNU/Linux, usando a biblioteca de processamento de imagens e visão computacional S2ILIB em conjunto com a API IPFRAMEWORK.
- Para o programa do sistema TOOLSPY para o sistema operacional Windows, usando a biblioteca de desenvolvimento de aplicações MTQLIB[1] desenvolvida no WZL, em conjunto com alguns módulos da biblioteca S2ILIB e a API IPFRAMEWORK.

Estas duas estruturas de blocos são detalhadas a seguir. Os programas para o sistema TOOLSPY gerados a partir de cada uma delas também são brevemente apresentados.

[1] MTQLIB é o nome da biblioteca para desenvolvimento de aplicações desenvolvida e mantida pelo WZL. Ela possui módulos próprios para a representação de tipos de dados e processamento destes, incluindo módulos para a interface com diferentes dispositivos, como placas de aquisição de dados e imagens. Maiores informações sobre esta biblioteca podem ser obtidas em *http://mtq002.wzl.rwth-aachen.de/mtqlib*.

6.1.1 Criação dos Blocos Básicos para o Programa do Sistema TOOLSPY para GNU/Linux

Os blocos[2] básicos para o programa do sistema TOOLSPY para o sistema operacional GNU/Linux foram criados a partir da biblioteca de processamento de imagens e visão computacional S2ILIB e da API IPFRAMEWORK. Todas as classes criadas para a instanciação desses blocos são derivadas da classe **IPFBlock**. A estrutura geral desses blocos juntamente com o esquema das ligações necessárias entre eles podem ser vistos na figura 6.1. Nesta figura, as flechas indicam o fluxo de dados e a seqüência de etapas necessárias (obrigatoriamente) para a execução da cadeia de processamento de imagens e visão computacional.

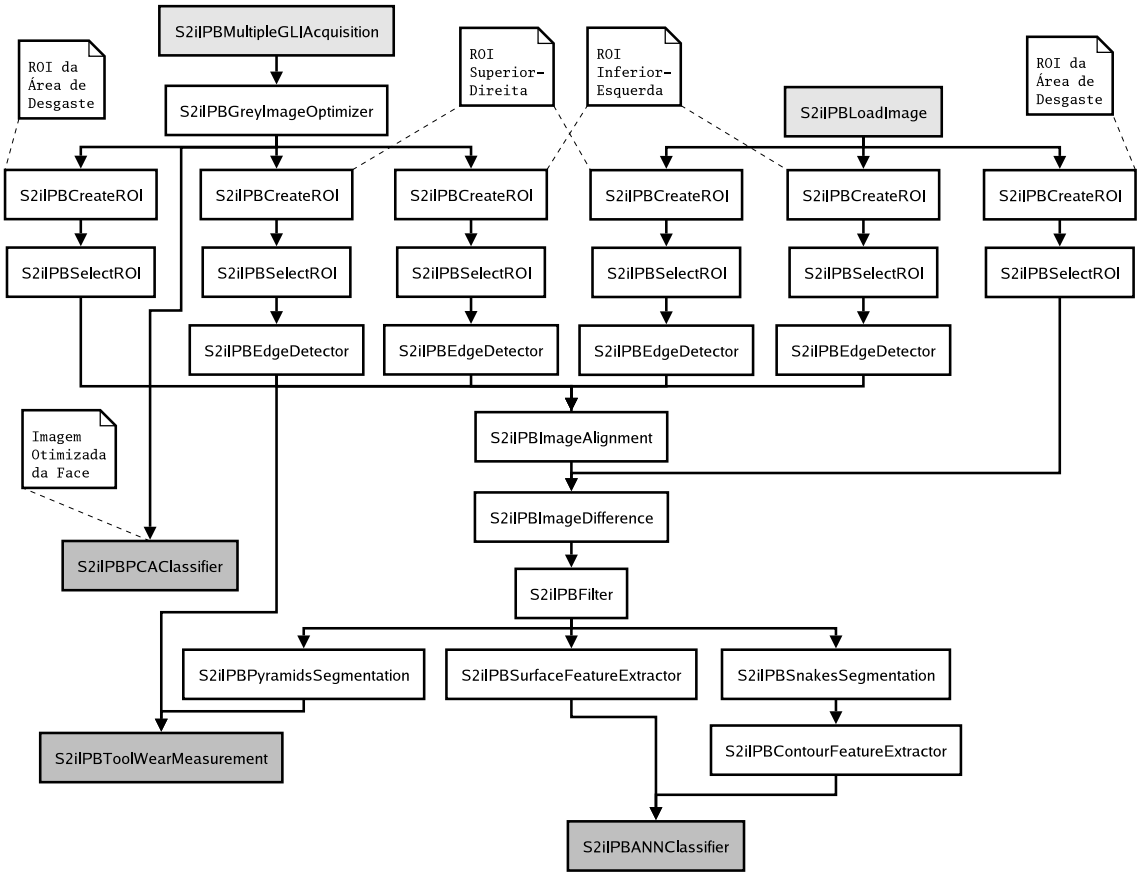


Figura 6.1: *Esquema dos blocos criados para o programa do sistema* TOOLSPY *para GNU/Linux, mostrando o esquema das ligações entre eles.*

S2iIPBMultipleGLIAcquisition: manipula objetos de classes dos módulos S2IGRAB e S2IILUMINATION da biblioteca S2ILIB (como atributos) para acesso aos dispositivos de aquisição das imagens e iluminação. É responsável por acionar o dispositivo de iluminação de forma adequada, de acordo com o padrão de iluminação a ser utilizado em cada aquisição individual definido previamente em um formato adequado como um mapa de LEDs a serem

[2]O termo blocos, quando usado dentro do âmbito do programa do sistema TOOLSPY, pode se referir, de acordo com o contexto, tanto a classes usadas para a instanciação de objetos quanto aos objetos instanciados por estas classes, objetos estes que formarão uma instância da cadeia de processamento de imagens e visão computacional do sistema.

acionados em cada aquisição. Desta forma, múltiplas imagens da ferramenta com diferentes níveis de iluminação em diferentes direções de incidência sobre a ferramenta são capturadas. Pode ser usado tanto para a aquisição de múltiplas imagens da ferramenta não desgastada quanto da ferramenta desgastada.

Entrada: nenhuma.

Saída: vetor de imagens em níveis de cinza (*EIPGreyLevelImagesVector*).

S2iIPBSingleGLIAcquisition: manipula objetos de classes dos módulos S2iGrab e S2iIlumination da biblioteca S2iLib para acesso aos dispositivos de iluminação e aquisição das imagens. Funciona de forma análoga ao bloco **S2iIPBMultipleGLIAcquisition**, adquirindo, contudo, apenas uma imagem. Pode ser usada para a aquisição de uma única imagem tanto da ferramenta desgastada quanto da ferramenta não desgastada.

Entrada: nenhuma.

Saída: imagem em níveis de cinza (*EIPGreyLevelImage*).

S2iIPBLoadImage: recupera uma imagem a partir de um arquivo em disco. Geralmente usado para recuperar a imagem otimizada em níveis de cinza da ferramenta não desgastada (modelo). Usa as funções de recuperação de imagens em disco da classe **S2iImage** do módulo S2iImage da biblioteca S2iLib.

Entrada: nenhuma.

Saída: imagem em níveis de cinza (*EIPGreyLevelImage*).

S2iIPBLoadImagesVector: recupera de um arquivo em disco um vetor de imagens em níveis de cinza obtidas, geralmente, a partir da aquisição com diferentes níveis de iluminação como implementada pelo bloco **S2iIPBMultipleGLIAcquisition**. Usado para recuperar um vetor de imagens da ferramenta desgastada ou da ferramenta não desgastada (modelo) para posterior processamento. Usa as funções de recuperação de seqüências de imagens em disco da classe **S2iImage** do módulo S2iImage da biblioteca S2iLib.

Entrada: nenhuma.

Saída: vetor de imagens em níveis de cinza (*EIPGreyLevelImagesVector*).

S2iIPBSaveImage: armazena uma imagem em um arquivo em disco. Geralmente usado para armazenar a imagem otimizada em níveis de cinza da ferramenta não desgastada (modelo) após aquisição pelo bloco **S2iIPBMultipleGLIAcquisition** e processamento pelo bloco **S2iIPBGreyImageOptimizer**.

Entrada: imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: nenhuma.

S2iIPBSaveImagesVector: armazena um vetor de imagens em um arquivo em disco, imagens estas geralmente obtidas a partir da aquisição com diferentes níveis de iluminação como implementado pelo bloco **S2iIPBMultipleGLIAcquisition**. Usado para armazenar um vetor de imagens

da ferramenta desgastada ou da ferramenta não desgastada (modelo) para posterior processamento.

Entrada: vetor de imagens em níveis de cinza (*EIPGreyLevelImagesVector*).

Saída: nenhuma.

S2iIPBCreateROI: cria uma ROI em uma imagem derivada da classe **S2iImage** do módulo S2IIMAGE da biblioteca S2ILIB para posterior processamento. Esta ROI é adicionada à lista de ROIs da imagem e é criada a partir das coordenadas, na imagem original, da referência inferior esquerda da ROI, da altura da ROI e da largura da ROI (ambas em pixels).

Entrada: imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: a mesma imagem em níveis de cinza (*EIPGreyLevelImage*), só que com a ROI criada adicionada à lista de ROIs da imagem.

S2iIPBSelectROI: seleciona uma ROI de uma imagem para processamento a partir da lista de ROIs da imagem. Esta ROI é selecionada a partir da interface *GetROI()* de classes derivadas da classe **S2iImage** do módulo S2IIMAGE da biblioteca S2ILIB.

Entrada: imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: ROI da imagem em níveis de cinza fornecida como entrada (*EIPGreyLevelImage*).

S2iIPBGreyImageOptimizer: gera a imagem otimizada em níveis de cinza a partir do vetor de imagens em níveis de cinza adquirido com diferentes níveis de iluminação, como implementado pelo bloco **S2iIPBMultipleGLIAcquisition**. Pode ser usado tanto a partir de múltiplas imagens da ferramenta desgastada quanto da ferramenta não desgastada.

Entrada: vetor de imagens em níveis de cinza (*EIPGreyLevelImagesVector*).

Saída: imagem em níveis de cinza (*EIPGreyLevelImage*).

S2iIPBEdgeDetector: detecta o contorno da ferramenta em cada uma das ROIs pré-definidas (superior-direita e inferior-esquerda). Manipula objetos de classes do módulo S2IMORPHOLOGY para a aplicação dos operadores de morfologia adequados (abertura, fechamento e *top-hat*). A partir da imagem resultante, determina os pontos do contorno e os interpola de forma a obter os coeficientes angular e linear das retas deste contorno nesta imagem, tendo como referência o vértice inferior esquerdo.

Entrada: imagem em níveis de cinza.

Saída: coeficientes angular e linear das retas do contorno das ferramentas em cada uma das ROIs analisadas (*EIPLine*).

S2iIPBImageAlignment: alinha a ROI da região de desgaste da imagem da ferramenta desgastada com a ROI da região de desgaste da imagem da ferramenta não desgastada de forma a possibilitar a execução da operação de diferença de imagens. Gera uma nova imagem a partir da imagem da ferramenta desgastada e sua ROI da região de desgaste. Utiliza para isso as interfaces *MoveXY()* (para a translação) e *Rotate()* (para a rotação) de classes derivadas da classe **S2iImage** do módulo S2IIMAGE da biblioteca S2ILIB.

Entrada: coeficientes angular e linear das retas do contorno das ferramentas em cada uma das ROIs analisadas (*EIPLine*) e imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: imagem em níveis de cinza (*EIPGreyLevelImage*).

S2iIPBImageDifference: executa a operação de diferença de imagens entre as ROIs da região desgastada da imagem da ferramenta desgastada e a imagem da ferramenta não desgastada. Utiliza para isso do operador de subtração ($-$) de classes derivadas da classe **S2iImage** do módulo S2IIMAGE da biblioteca S2ILIB.

Entrada: duas imagens em níveis de cinza (*EIPGreyLevelImage*), de mesma dimensões.

Saída: imagem em níveis de cinza (*EIPGreyLevelImage*).

S2iIPBFilter: aplica uma seqüência de filtros à imagem diferença de forma a eliminar ruídos e áreas de falso desgaste. Manipula objetos de classes do módulo S2IFILTER da biblioteca S2ILIB de forma a aplicar os filtros necessários (mínimo, máximo e gaussiano).

Entrada: imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: imagem em níveis de cinza (*EIPGreyLevelImage*).

S2iIPBPyramidsSegmentation: executa a operação de segmentação por pirâmides de forma a determinar a área de desgaste da ferramenta a partir da imagem diferença obtida entre as ROIs da região de desgaste da imagem otimizada da ferramenta desgastada e da imagem otimizada da ferramenta não desgastada. Manipula um objeto da classe **S2iPyramidsSegmentation** do módulo S2ISEGMENTATION da biblioteca S2ILIB.

Entrada: imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: imagem em níveis de cinza (*EIPGreyLevelImage*).

S2iIPBSnakesSegmentation: executa a operação de segmentação por *snakes* de forma a determinar o contorno da área desgastada da ferramenta a partir da imagem diferença obtida entre as ROIs da região de desgaste da imagem otimizada da ferramenta desgastada e da imagem otimizada da ferramenta não desgastada. Obtém-se um conjunto de pontos que descrevem o contorno da área de desgaste. Manipula um objeto da classe **S2iSnakesSegmentation** do módulo S2ISEGMENTATION da biblioteca S2ILIB.

Entrada: imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: vetor de pontos descrevendo o contorno da área desgastada (*EIPPointsVector*).

S2iIPBContourFeatureExtractor: calcula a FFT do contorno da área desgastada da ferramenta proveniente da saída do bloco **S2iIPBSnakesSegmentation** e extrair os coeficientes de interesse para posterior processamento. Manipula um objeto de uma classe derivada da classe **S2iFFT** do módulo S2IFOURIER da biblioteca S2ILIB.

Entrada: vetor de pontos descrevendo o contorno da área desgastada (*EIPPointsVector*).

Saída: vetor de valores em ponto flutuante (*EIPFloatsVector*).

S2iIPBSurfaceFeatureExtractor: extrai informações estatísticas da ROI da região de desgaste da imagem otimizada da ferramenta desgastada para posterior processamento, após uma operação de normalização. Manipula objetos de classes do módulo S2IIMAGESTATISTICS da biblioteca S2ILIB para a obtenção destas informações.

Entrada: imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: vetor de valores em ponto flutuante (*EIPFloatsVector*).

S2iIPBANNClassifier: realiza a classificação do tipo de desgaste a partir de determinados coeficientes. A rede neural deve estar previamente treinada para isso. A topologia da rede e seus parâmetros, como o peso das conexões entre os neurônios podem ser carregados a partir de um arquivo em disco. Manipula um objeto da classe S2iNeuralManager do módulo S2INEURAL da biblioteca S2ILIB para o uso da RNA.

Entrada: vetor de valores em ponto flutuante (*EIPFloatVector*).

Saída: inteiro (*EIPInt*, tipo de uma enumeração representando os possíveis tipos de desgaste previamente definidos e treinados com a rede neural artificial.).

S2iIPBPCAClassifier: realiza a classificação do tipo de desgaste a partir da própria imagem da ferramenta usando a técnica de PCA. Usado para a determinação da presença ou não do desgaste de cratera da ferramenta a partir da imagem otimizada em níveis de cinza da face. Manipula objetos de classes do módulo S2IEIGENFACES da biblioteca S2ILIB.

Entrada: imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: inteiro (*EIPInt*, tipo de uma enumeração representando os possíveis tipos de desgaste previamente definidos, cujas imagens foram fornecidas para a determinação do espaço de classificação).

S2iIPBToolWearMeasurement: realiza a medição do desgaste como descrito na seção 5.1.4.

Entrada: imagem em níveis de cinza (*EIPGreyLevelImage*).

Saída: desgaste de flanco (VB), desgaste de flanco máximo (VB_{max}) e área do desgaste de flanco (A_{VB}) (todos *EIPFloat*).

Os principais tipos de dados usados na conexão entre os diversos blocos foram acima especificados. Estes tipos de dados são detalhados a seguir:

***EIPGreyLevelImagesVector*:** vetor de imagens em níveis de cinza, tipicamente uma seqüência de imagens obtidas com variação na iluminação, como implementado pelo bloco **S2iIPBMultipleGLIAcquisition**.

***EIPGreyLevelImage*:** imagem em níveis de cinza, tipicamente uma imagem que já recebeu algum processamento, como pelo algoritmo de otimização de imagens em níveis de cinza (implementado pelo bloco **S2iIPBGreyImageOptimizer**), pela operação de diferença de imagens

(implementado pelo bloco **S2iIPBImageDifference**), por algum operador de segmentação ou filtragem (implementados pelos blocos **S2iIPBFilter**, **S2iIPBPyramidsSegmentation** e **S2iIPBSnakesSegmentation**, por exemplo), entre outros.

***EIPLine*:** armazena uma linha, com os coeficientes linear e angular. Resultado da detecção de pontos de uma borda e posterior interpolação, como implementado no bloco **S2iIPBEdgeDetector**.

***EIPPointsVector*:** vetor de pontos, em geral um contorno a ser posteriormente processado por um bloco como o responsável pela extração de características do contorno, como implementado por **S2iIPBContourFeatureExtractor**.

***EIPFloatsVector*:** vetor de dados do tipo ponto flutuante, em geral os coeficientes de alguma transformada, como a FFT, implementada pelo bloco **S2iIPBContourFeatureExtractor**, ou o resultado de extração de características estatísticas como implementado pelo bloco **S2iIPBSurfacfeFeatureExtractor**.

***EIPInt*:** um inteiro, em geral de algum tipo de dado enumeração, como o tipo de desgaste, resultado da classificação no caso do algoritmo de classificação por RNAs ou pela técnica de PCA (**S2iIPBANNClassifier** e **S2iIPBPCAClassifier**).

***EIPFloat*:** um número em ponto flutuante, em geral o resultado de uma medição gerado pelo bloco **S2iIPBToolWearMeasurement**.

Cada um desses blocos possui uma série de parâmetros que precisam ser especificados de forma que o sistema funcione corretamente. A determinação destes parâmetros é responsabilidade do usuário e da aplicação em questão. Nas seções 6.1.3 e 6.1.4 descrevem-se as interfaces gráficas criadas de forma a possibilitar a especificação destes parâmetros por parte do usuário.

Blocos como **S2iIPBMultipleGLIAcquisition** e **S2iIPBLoadImage** podem ser substituídos por outros blocos, como por exemplo **S2iIPBLoadImagesVector** ou **S2iIPBSingleGLIAcquisition**, quando conveniente. Além disso, blocos como **S2iIPBSaveImage** e **S2iIPBSaveImagesVector** podem ser colocados na estrutura de forma a possibilitar que imagens sejam armazenadas em disco e posteriormente recuperadas, no caso em que se deseje reavaliar o processamento da cadeia.

Todos os blocos criados foram agrupados em um novo bloco da biblioteca S2ILIB, chamado de S2IIPBLOCKS. Este módulo, dado o seu caráter bastante diverso, realizando interfaces de diferentes módulos da biblioteca S2ILIB, depende de diversos outros módulos desta. Todas estas dependências devem ser satisfeitas de forma que este módulo possa ser utilizado na prática.

6.1.2 Criação dos Blocos Básicos para o Programa do Sistema TOOLSPY para Windows

Os blocos básicos para o programa do sistema TOOLSPY para Windows foram criados a partir dos módulos IPFRAMEWORK e MTQLIB. A estrutura dos blocos criada para o programa do sistema TOOLSPY para Windows é semelhante à estrutura descrita na seção 6.1.1, sendo que não será descrita

em detalhe. Esta estrutrura pode ser vista na figura 6.2 e a equivalência entre os blocos desta estrutura com os blocos da estrutura descrita na seção 6.1.1 pode ser vista na tabela 6.1.

Toolspy (Windows)	**Toolspy (GNU/Linux)**
MtqIPBReadImage	S2iIPBLoadImage
	S2iIPBLoadImagesVector
MtqIPBGreyImgOptim	S2iIPBGreyImageOptimizer
MtqIPMatchedDiffImg	S2iIPBEdgeDetector
	S2iIPBImageAlignment
	S2iIPBImageDifference
	S2iIPBFilter
MtqIPPyramidSeg	S2iIPBPyramidSegmenter
MtqIPSnakeSeg	S2iIPBSnakesSegmenter
MtqIPBFeatureExt	S2iIPBContourFeatureExtractor
	S2iIPBSurfaceFeatureExtractor
MtqIPBNeuralNet	S2iIPBNeuralNetworkClassifier
MtqIPBMeasurement	S2iIPBToolWearMeasurement

Tabela 6.1: *Quadro comparativo entre os blocos do sistema* TOOLSPY *para o sistema operacional Windows e seus equivalentes para o sistema operacional GNU/Linux.*

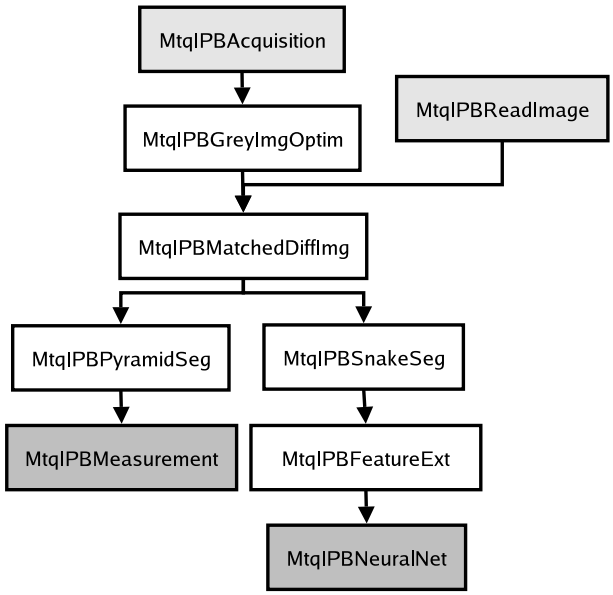


Figura 6.2: *Diagrama com os blocos criados aplicados ao programa* TOOLSPY.

Esta estrutura leva em consideração a hierarquia de classes das estruturas de dados da biblioteca MTQLIB, sendo que algumas funcionalidades da S2ILIB foram pontualmente incorporadas a esta biblioteca para que se pudesse montar esta estrutura, mais especificamente o algoritmo para o cálculo da FFT, a biblioteca de RNAs e os algoritmos de segmentação por pirâmides e *snakes*. Esta foi a primeira estrutura a ser criada para o programa do sistema TOOLSPY usando a API IPFRAMEWORK como suporte, antes da estrutura composta pelos S2IIPBLOCKS descrita na seção 6.1.1. Nesta estrutura, a utilização da imagem da face da ferramenta para a detecção do desgaste de cratera não é realizada. A imagem da face é processada da mesma forma que a imagem do flanco principal, para se avaliar o desgaste de flanco e sua classificação, só que a partir de outro plano.

6.1.3 O Programa do Sistema TOOLSPY para GNU/Linux

O programa do sistema TOOLSPY para GNU/Linux implementa as funcionalidades da cadeia de processamento de imagens e visão computacional descritas no capítulo 5 a partir da estrutura de blocos descrita na seção 6.1.1, baseada na biblioteca S2ILIB e na API IPFRAMEWORK apresentada na seção 4.3.

Com este programa é possível carregar uma cadeia de processamento de imagens e visão computacional pré-determinada, como a cadeia de processamento do sistema TOOLSPY e carregar parâmetros pré-determinados para os blocos que formam esta cadeia. Além disso, é possível visualizar os resultados de cada bloco da cadeia, bem como alterar os parâmetros desta.

Blocos como os responsáveis pela implementação de RNAs e a técnica de PCAs devem ter seus parâmetros determinados separadamente, a partir de outra interface (gráfica ou não). Ou seja, o treinamento da RNA e a determinação do espaço de classificação para a técnica de PCA deve ser realizado separadamente, a partir de um programa simples externo.

6.1.4 O Programa do Sistema TOOLSPY para Windows

O programa do sistema TOOLSPY para Windows implementa as funcionalidades da cadeia de processamento de imagens e visão computacional descrita no capítulo 5 a partir da estrutura de blocos descrita na seção 6.1.2, baseada na biblioteca MTQLIB e na API IPFRAMEWORK apresentada na seção 4.3. A figura 6.3 ilustra a janela principal do programa.

Este programa tem por objetivo demonstrar o funcionamento do sistema desde a aquisição das imagens até o resultado da medição e a classificação, fornecendo, de forma visual e rápida, todos os resultados intermediários do processamento. Ele permite também que os parâmetros do sistema sejam determinados para cada algoritmo da cadeia. Além disso, os parâmetros podem ser salvos e carregados a partir de um arquivo, arquivo este que pode ter sido salvo, por exempo, pelo programa TOOLSPYOPTIMIZER, como visto na seção 6.1.5.

Existem três modos de operação básicos para o sistema:

Inativo: o programa está aguardando que uma cadeia de procesamento de imagens e visão computacional com o seu conjunto de parâmetros seja carregada. Neste modo, o ajuste dos parâmetros desta cadeia pode ser efetuado.

Medição: a cadeia de processamento de imagens e visão computacional está pronta para processar as imagens de uma ferramenta desgastada para realizar a avaliação do desgaste. Neste modo, todos os parâmetros do sistema devem estar corretamente determinados e a rede neural para a classificação do desgaste deve estar corretamente treinada.

Treinamento: a cadeia de processamento de imagens está inativa, sendo que as imagens são processadas até os coeficientes necessários para a entrada da RNA serem calculados. A partir

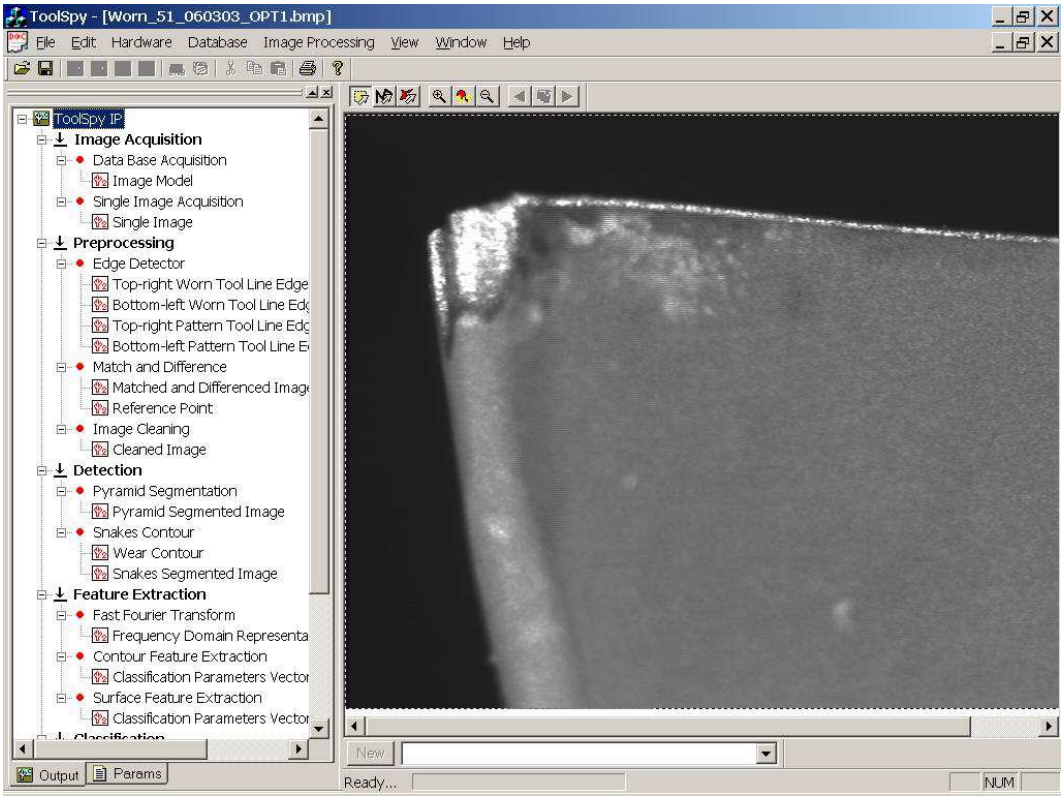


Figura 6.3: *Janela principal do programa* TOOLSPY *ilustrando a cadeia de processamento de imagens e visão computacional (à esquerda).*

deste ponto, exemplos de treinamento compostos pelos coeficientes e a classificação real do desgaste são adicionados ao conjunto de amostras de treinamento da RNA para posterior ajuste dos pesos das conexões entre os neurônios das diferentes camadas para treinamento.

No modo *medição*, clicando-se sob cada algoritmo de processamento na barra da esquerda, aba *Output*, são mostrados os resultados do processamento deste passo. No modo inativo, clicando-se sobre a aba *Params* na barra da esquerda são mostrados os parâmetros, que podem ter os seus valores alterados.

6.1.5 O Programa de Otimização dos Parâmetros – TOOLSPYOPTIMIZER

Um sistema automatizado para a otimização dos parâmetros de um sistema de processamento de imagens e visão computacional foi proposto e utilizado por Deschamps [120] para testar a validade de um algoritmo para a detecção do contorno da área do desgaste da ferramenta aplicado ao sistema TOOLSPY. Este trabalho foi posteriormente incorporado à biblioteca MTQLIB e um programa para a otimização dos parâmetros foi desenvolvido, sendo usada a estrutura fornecida pela API IPFRAMEWORK.

Este sistema é baseado em algoritmos genéticos [14, 15, 16] e processamento paralelo, utilizando *clusters* de computadores para tal fim. Os parâmetros dos algoritmos de processamento da cadeia do

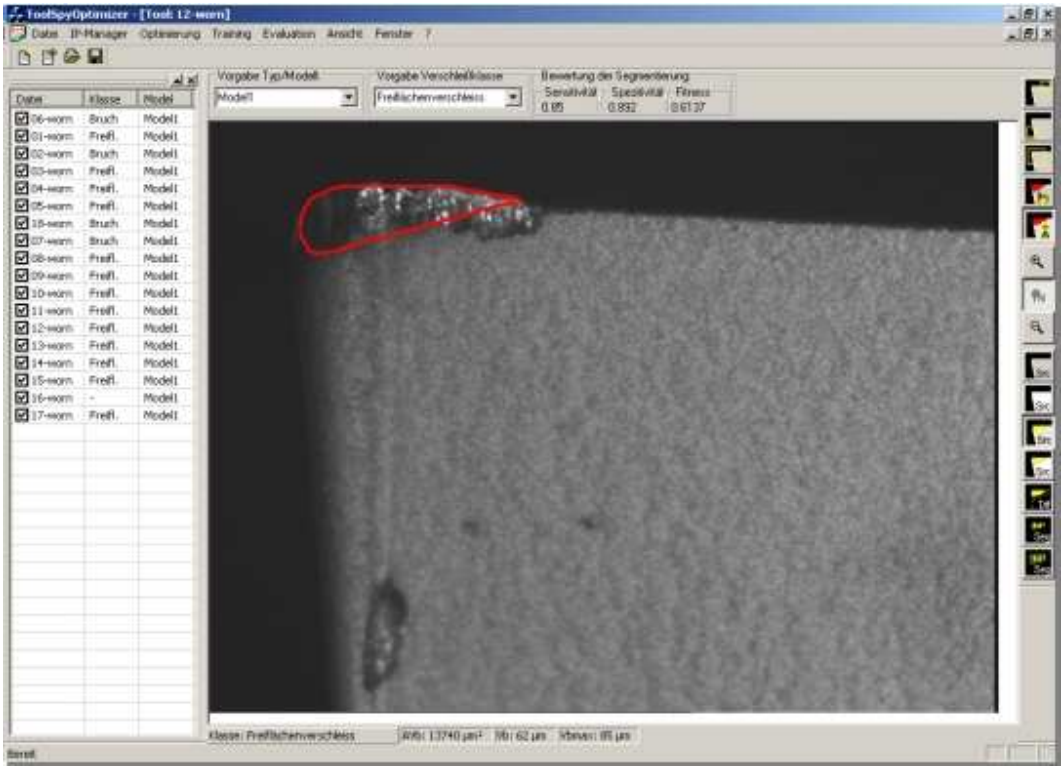


Figura 6.4: *Janela principal do programa* TOOLSPYOPTIMIZER *ilustrando a segmentação manual do contorno da ferramenta.*

sistema TOOLSPY são armazenados em uma estrutura conveniente e a cadeia é constantemente avaliada, tomando o lugar da função de avaliação do algoritmo genético, conhecendo-se antecipadamente os resultados da mesma para um determinado conjunto de imagens a ser analisado. A estrutura geral do sistema pode ser visualizada na figura 6.5.

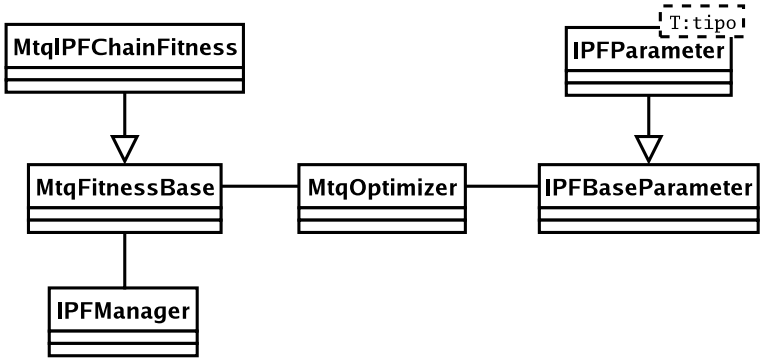


Figura 6.5: *Estrutura geral do funcionamento do programa* TOOLSPYOPTIMIZER*, mostrando a relação com as classes* ***MtqOptimizer*** *e* ***MtqFitnessBase****, que implementam a funcinolidade de algoritmos genéticos da* ***MtqLib****.*

.

Com este programa, também, apenas partes da cadeia de processamento de imagens e visão computacional e seus parâmetros podem ser avaliadas, como por exemplo apenas os algoritmos de segmentação. Para isso, a segmentação manual é feita e os algoritmos são avaliados a partir do resultado da comparação da segmentação manual com a segmentação do algoritmo com diferentes parâmetros.

O programa TOOLSPYOPTIMIZER permite que os parâmetros do algoritmo genético para a otimização dos parâmetros da cadeia de processamento de imagens e visão computacional sejam ajustados de forma a que os melhores resultados sejam obtidos. Por exemplo, o número máximo de gerações, o erro máximo requerido, o número de elementos em cada geração, entre outros, podem ser definidos através desta interface.

6.2 Resultados da Cadeia de Processamento de Imagens Proposta

A partir da estrutura descrita na seção 6.1.1, foi montada a cadeia de processamento de imagens e visão computacional para a realização de testes do sistema TOOLSPY com a interface descrita na seção 6.1.3. Os parâmetros foram ajustados de forma manual (não sendo usado o sistema de otimização de parâmetros descrito na seção 6.1.5, uma vez que este não se encontrava totalmente funcional na época de realização dos testes). Pastilhas de torneamento foram avaliadas, muito semelhantes às mostradas na figura 2.3. Um conjunto com 5 exemplos de ferramentas com desgaste de flanco, 5 exemplos de ferramentas quebradas e 5 exemplos de ferramentas não desgastadas foi usado para o treinamento da rede neural do bloco **S2iIPBANNClassifier** e o cálculo do espaço de classificação para a técnica de PCA do bloco **S2iIPBPCAClassifier**. Os resultados estão mostrados na tabela 6.2. A medição do valor de referência do desgaste foi feita pela utilização de microscópio óptico.

Não havendo número suficiente de imagens de desgaste de cratera, optou-se por se testar o sistema de classificação através da técnica de PCA com a classificação do desgaste em desgaste de flanco ou quebra, comparando-o ao sistema de classificação por RNA.

Número	VB (μm)	A_{VB} (μm^2)	VB_{max} (μm)	VB_{max}^{ref} (μm)	**Erro** (μm)	**Classificação RNA**	**Classificação PCA**	**Referência**
1	419,9	312350	544,4	538,5	5,9	flanco	flanco	flanco
2	472,8	316859	544,4	538,5	5,9	flanco	flanco	flanco
3	407,5	350694	531,9	538,5	6,6	flanco	flanco	flanco
4	348,4	236606	451,1	449,1	2,0	flanco	flanco	flanco
5	388,8	315136	413,8	449,1	35,3	flanco	flanco	flanco
6	360,8	353515	472,8	449,1	23,7	flanco	flanco	flanco
7	376,4	436028	581,7	516,5	65,2	flanco	flanco	flanco
8	488,4	434184	765,3	516,5	248,8	quebra	quebra	flanco
9	351,5	288767	444,8	475,9	31,9	quebra	quebra	quebra
10	264,4	246691	398,2	311,8	86,4	flanco	flanco	flanco
11	591,1	578057	824,4	311,8	512,6	desconhecido	quebra	flanco
12	531,9	578144	681,3	685,8	4,5	quebra	quebra	quebra
13	528,8	570242	709,3	685,8	23,5	quebra	quebra	quebra
14	531,9	617617	671,9	685,8	13,9	desconhecido	quebra	quebra
15	345,3	363044	475,9	475,9	0,0	desconhecido	flanco	quebra
16	307,9	247107	438,6	475,9	37,3	desconhecido	flanco	quebra

Tabela 6.2: *Resultados obtidos com o sistema para a medição de pastilhas de torneamento.*

Descartando-se as medições realizadas nos números 8 e 11, que apresentaram resultados bastante divergentes do desgaste de flanco medido em relação ao desgaste de flanco de referência, o sistema

apresentou uma dispersão de medida de 24,43 μm. O erro máximo foi para a ferramenta 10, com o valor 86,4 μm. O erro mínimo foi para a ferramenta 15, com valor 0,0.

As imagens dos números 1, 2 e 3 são da mesma pastilha com alguns ajustes de posicionamento para a aquisição das imagens, sendo então que três medidas foram realizadas. O erro médio foi de 6,13 μm, sendo que o erro percentual em relação à medição de referência foi de 1%.

As imagens dos números 4, 5 e 6 são da mesma pastilha com alguns ajustes de posicionamento para a aquisição das imagens, sendo então três medidas realizadas. O erro médio foi de 20,3 μm, sendo que o erro percentual em relação à medição de referência foi de 5%.

Para os números 7, 9 e 10 os erros percentuais em relação às medições de referência foram de 13%, 7% e 28%, respectivamente.

As imagens dos números 12, 13 e 14 são da mesma pastilha com alguns ajustes de posicionamento para a aquisição das imagens, sendo então que três medidas distintas foram realizadas. O erro médio foi de 13,97 μm, sendo que o erro percentual em relação à medição de referência foi de 2%.

As imagens dos números 15 e 16 são da mesma pastilha com alguns ajustes de posicionamento para a aquisição das imagens, sendo então que duas medidas distintas foram realizadas. O erro médio foi de 18,65 μm, sendo que o erro percentual em relação à medição de referência foi de 4%.

Para o sistema de classificação por RNA, a taxa de acerto foi de 69%. Para o sistema de classificação por PCA, a taxa de acerto foi de 75%. Para uma melhor comparação entre estas duas técnicas, é necessário que se tenha um maior número de imagens tanto de treinamento, para melhor definir os pesos da RNA e o espaço de classificação da técnica de PCA, quanto de testes, para se obter um estudo estatístico mais preciso sobre a resposta do sistema.

No apêndice C são apresentados os resultados intermediários do processamento dessas imagens.

Capítulo 7

Conclusões e Perspectivas

Neste capítulo serão apresentadas as conclusões e perspectivas futuras para este trabalho. Inicia-se com uma discussão das conclusões obtidas após a análise dos resultados, continuando com a apresentação das perspectivas para o posterior desenvolvimento do sistema TOOLSPY e para o desenvolvimento de novas pesquisas e trabalhos. Conclui-se o capítulo com algumas considerações finais, dentre elas pontos de estudo para futuros trabalhos a serem desenvolvidos nesta área.

7.1 Conclusões

A idéia inicial com este trabalho era desenvolver um sistema para classificar os diferentes tipos de desgaste encontrados em ferramentas de corte (como exemplos de desgaste ver as figuras 2.6 e 2.7) através de técnicas de processamento de imagens e visão computacional. Contudo, com a falta de um número significativo de imagens dos diferentes tipos de desgaste para um mesmo tipo de ferramenta para que o sistema pudesse ser implementado e testado, este estudo sistemático não pôde ser totalmente efetuado. Assim, esta tarefa teve sua prioridade minimizada, sendo que métodos de classificação deverão ser estudados em outro trabalho.

Este trabalho abordou a estruturação de um sistema de visão para o monitoramento do desgaste de ferramentas de corte com ênfase na arquitetura da cadeia de processamento de imagens e visão computacional e seus algoritmos, de forma que o sistema possa ser facilmente adaptado à avaliação do desgaste em diferentes tipos de ferramentas sob diferentes condições de operação, como em uma estação de medição em laboratório e em uma estação de medição montada no lado da máquina-ferramenta.

Testes preliminares para a cadeia de processamento de imagens e visão computacional com algoritmos de segmentação baseados em *threshold* global, como indicado pelas técnicas descritas na seção 3.4.1, mostraram-se bastante ineficientes para o problema em questão, reforçando as conclusões de que técnicas mais sofisticadas de segmentação necessitariam ser experimentadas para que a robustez da solução obtida fosse melhorada. Desta forma, algoritmos de segmentação piramidais e

baseados em *snakes* foram utilizados, em virtude dos bons resultados obtidos. Etapas de filtragem e pré-processamento da imagem se mostraram necessárias para eliminar ruídos da imagem, como regiões mais escuras da ferramenta, que representavam áreas de falso desgaste.

A técnica de classificação baseada na análise de componentes principais com uma etapa de normalização mostrou bons resultados na classificação da imagem do flanco principal da ferramenta desgastada em quebra e desgaste de flanco. Infelizmente, devido à falta de um número significativo de imagens de ferramentas apresentando desgaste de cratera, esta técnica não pode ser aplicada à classificação de imagens da face da ferramenta. Comparativamente à técnica de redes neurais artificiais, também empregada para a análise neste caso, esta técnica mostrou resultados um pouco melhores.

Um dos grandes problemas do sistema é a utilização da imagem modelo para a execução da operação de diferença de imagens entre esta e a imagem da ferramenta desgastada, na ROI da região de desgaste, uma solução adotada por Orth [17] e Maeda et al. [100] e que se mostra melhor do que o simples *threshold* global ou adaptativo. O correto funcionamento desta operação é dependente de que o nível de iluminação, quando da aquisição da imagem da ferramenta modelo e da aquisição da imagem da ferramenta desgastada, sejam semelhantes e de que a imagem da ferramenta desgastada seja corretamente transladada e rotacionada, ficando alinhada com a imagem da ferramenta modelo. Nos testes realizados, problemas foram encontrados quando do posicionamento da ferramenta sob certos ângulos, principalmente em ângulos em que o quebra-cavaco da ferramenta se tornava evidente.

A arquitetura IPFRAMEWORK desenvolvida mostrou-se adequada para a construção da cadeia de processamento de imagens e visão computacional proposta. Ela foi utilizada para a construção dessa cadeia utilizando dois suportes distintos: a biblioteca S2ILIB e a biblioteca MTQLIB. Com futuros desenvolvimentos, é possível que estas duas bibliotecas se aproximem a ponto de terem muitas de suas funcionalidades compartilhadas.

A cadeia de processamento de imagens teve sucesso na medição do desgaste para um determinado tipo de ferramenta, sendo que também classificou corretamente o desgaste da mesma na maioria dos casos.

É necessário que, a partir deste trabalho, um banco de dados de imagens de ferramentas desgastadas de diferentes tipos com diferentes morfologias de desgaste seja formado para que testes mais intensivos possam ser realizados com o sistema. Desta forma, a cadeia de processamento de imagens e visão computacional do sistema poderá ser testada para outros tipos de ferramenta que não a pastilha de torneamento usada para os testes neste trabalho.

7.2 Perspectivas Futuras

Nesta seção serão discutidas as perspectivas futuras para o posterior desenvolvimento deste trabalho. Inicia-se com as perspectivas para o monitoramento do estado das ferramentas no âmbito do projeto SFB368.

7.2.1 Perspectivas para o Monitoramento do Estado das Ferramentas no Âmbito do Projeto SFB368

O monitoramento do estado das ferramentas para aplicação em células autônomas de produção é de especial importância para a operação do sistema sem a supervisão de um operador humano. Como colocado anteriormente como requisito do sistema TOOLSPY, dois são os principais tipos de desgaste que precisam ser avaliados: o desgaste de flanco e o desgaste de cratera e a detecção de quebra da ferramenta.

A medição precisa do desgaste de cratera, de forma geral, não é importante para aplicações industriais. O que se deseja, no entanto, é saber sobre a existência ou não deste tipo de desgaste para que as condições de usinagem sejam ajustadas de forma a evitá-lo, o que pode ser feito através de técnicas de processamento de imagens 2D. Contudo, para a medição precisa do desgaste de cratera, técnicas 3D devem ser empregadas, como métodos de projeção de franjas (Moiré). Além disso, mesmo esses métodos de medição 3D podem ser empregados para a medição do desgaste de flanco da ferramenta, fornecendo informações precisas sobre estes tipos de desgaste.

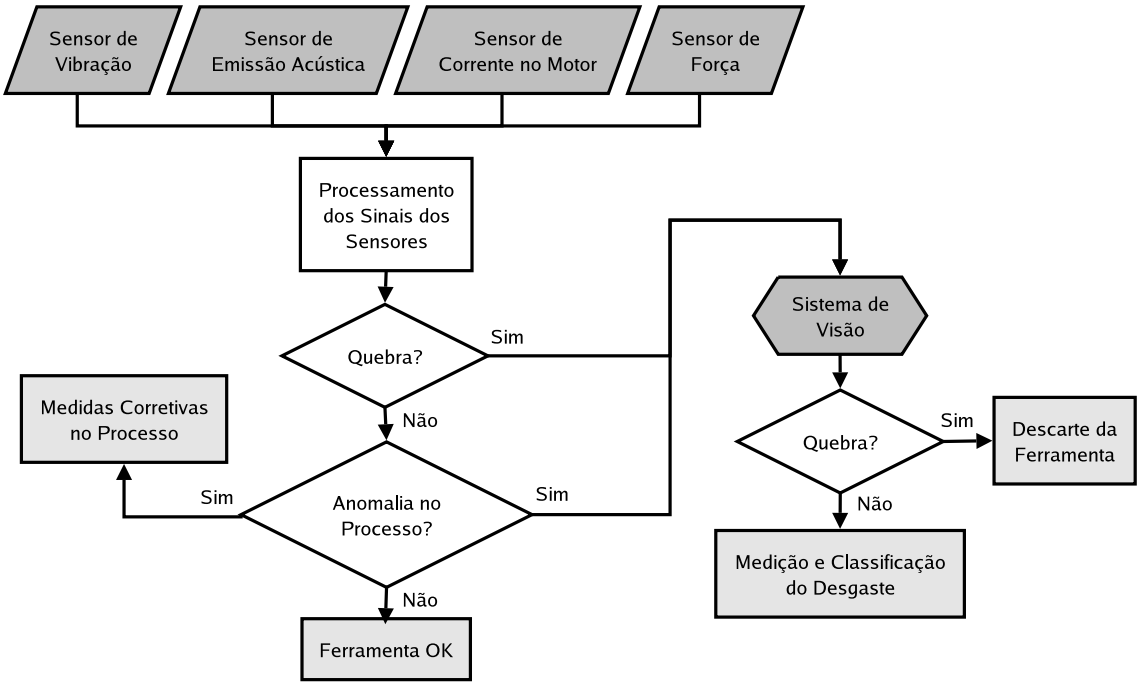


Figura 7.1: *Técnica de fusão de sensores a ser desenvolvida para melhorar a robustez do sistema.*

A utilização de métodos indiretos para o monitoramento do estado das ferramentas pode ser feita em conjunto com a utilização do método direto baseado em um sistema de visão. Por exemplo, uma técnica baseada na fusão de sensores, com o emprego de diversos tipos diferentes de sensores para o monitoramento indireto do estado das ferramentas (como sensores de emissão acústica, vibração e forças) e o processamento dos sinais desses sensores através de técnicas de inteligência artificial como redes neurais artificiais pode fornecer subsídios para a posterior medição do desgaste através de um método direto baseado em um sistema de visão. Por exemplo, o monitoramento através de métodos indiretos pode indicar a imediata quebra da ferramenta, o que só seria detectado posteriormente ao

final do ciclo de trabalho através do método direto baseado em um sistema de visão. O monitoramento indireto também poderia alertar para a possibilidade de existência de microlascamentos na ferramenta, o que comprometeria a qualidade do produto final e o que poderia ser facilmente confirmado por um robusto sistema de classificação em um sistema de visão. Além disso, o método direto baseado em um sistema de visão pode confirmar as informações obtidas através do uso de um método indireto, validando ou não o modelo levantado. A utilização conjunta destas duas técnicas, como pode ser visto na figura 7.1 pode levar ao desenvolvimento de sistemas de monitoramento bastante robustos, que tiram proveito das melhores características de cada um desses métodos (diretos e indiretos).

7.2.2 Perspectivas para a API IPFRAMEWORK

A API IPFRAMEWORK foi fruto de um desenvolvimento contínuo no decorrer deste e de outros trabalhos. A sua aplicação ao sistema TOOLSPY concedeu a este modularidade no projeto dos algoritmos da cadeia de processamento de imagens e visão computacional, possibilitando assim que novos algoritmos fossem testados de forma rápida, robusta e confiável. O foco de atenção no desenvolvimento do sistema deixou de ser a determinação de uma estrutura adequada para o sistema, sendo que os esforços podem ser agora concentrados na determinação de técnicas e algoritmos adequados para que resultados corretos sejam obtidos, testando-se o sistema para diferentes tipos de ferramentas.

A forma genérica pela qual a API IPFRAMEWORK foi implementada garante a aplicabilidade desta não somente ao sistema TOOLSPY, mas também a outros projetos que venham a ser desenvolvidos pelo grupo S2i, pelo laboratório WZL ou por outros grupos de pesquisa e empresas da área de sistemas de visão. Esta API está disponível no servidor CVS do DAS e pode ser obtida através de acesso anônimo pelo protocolo *pserver*. Um dos projetos do grupo S2i no qual se pretende usar a API de forma intensiva é no projeto HARPIA, que visa o desenvolvimento de uma interface gráfica para o desenvolvimento, especificação, prototipagem e testes de sistemas de visão através do paradigma de diagramas de blocos (ver figura 7.2 para um exemplo de diagrama de blocos de um sistema de visão), sendo que blocos de processamento são graficamente colocados em uma janela, ligando-se então as entradas e saídas destes através de conexões próprias para esta finalidade. Alguns produtos comerciais nesta área são o *Framework* da empresa DVT e o *WiT* da empresa Coreco. Contudo, o custo destes sistemas é extremamente alto e a sua aplicabilidade se restringe à aplicabilidade dos dispositivos físicos comercializados por estas empresas. Nenhum destes sistemas foi desenvolvido de acordo com a filosofia de software livre, que é um dos objetivos do projeto HARPIA. A base para este projeto, com a API IPFRAMEWORK e a biblioteca de processamento de imagens e visão computacional S2ILIB está praticamente completa, o que foi demonstrado pela criação dos blocos básicos para os programas do sistema TOOLSPY, tanto para Windows quanto para GNU/Linux.

Alguns ajustes devem ainda ser feitos à forma de implementação interna da API. Uma nova versão deve implementar as funcionalidades internas na forma de um grafo direcionado, permitindo mecanismos de validação formal do sistema, o que seria muito útil quando da tentativa de certificação do funcionamento da estrutura do sistema. Da forma como está implementada hoje, é necessário que o usuário especifique a ordem em que os elementos deverão ser processados pela API. Se implementada com a estrutura de um grafo direcionado, isto não mais será necessário, uma vez tendo-se as

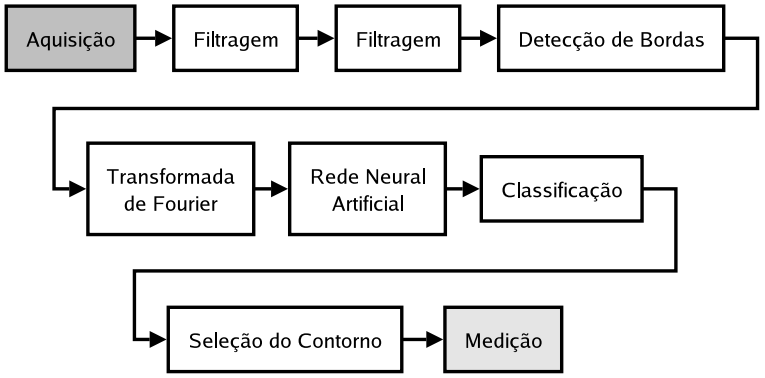


Figura 7.2: *Exemplo de diagrama de blocos de um sistema de visão.*

conexões entre os blocos definidas. Algoritmos como *Topological Sort* podem ser utilizados para a determinação da correta seqüência de processamento. Outras funções da área de teoria de grafos podem também ser adotadas, como mecanismos para a determinação da validade do sistema construído com a API e formalizando a sua idéia.

Finalmente, a API desenvolvida deve ter seu funcionamento testado no sistema embarcado final a ser empregado na estação de medição a ser montada ao lado da máquina-ferramenta. Para isto, não apenas a API mas a própria biblioteca S2ILIB e os blocos da cadeia de processamento de imagens e visão computacional do sistema TOOLSPY devem ser testados em conjunto. Esta tarefa não deve demandar um grande esforço, uma vez que o sistema embarcado adquirido para tal finalidade é um computador do tipo PC, muito semelhante aos computadores de mesa usados na atualidade.

7.2.3 Perspectivas para a Cadeia de Processamento de Imagens

Como já mencionado anteriormente, a utilização da API IPFRAMEWORK facilita a criação de novos blocos de processamento de imagens e visão computacional e a incorporação destes ao subsistema de processamento do sistema de visão de forma que possam ser rapidamente testados e validados. Isto abre caminho para que novas idéias sejam livremente exploradas sem a necessidade do investimento de muito tempo nestas tarefas.

Um dos próximos aspectos a serem explorados é a tentativa de utilização de uma técnica de segmentação baseada em informações de textura, como proposto por Kurada e Bradley [35]. Para isto pode ser usado, por exemplo, o módulo S2IWAVELET da biblioteca S2ILIB. Bases de *wavelets* adequadas devem ser determinadas e testadas. A idéia com uma técnica de segmentação baseada em textura é a eliminação da necessidade de uma imagem da ferramenta modelo, utilizando-se então a imagem diferença entre a imagem da ferramenta modelo e a imagem da ferramenta desgastada. Técnicas de segmentação baseadas em operadores de gradiente também podem ter a sua eficiência testada.

Outro aspecto a ser explorado é a utilização da informação de cores em pastilhas revestidas. Isto pode facilitar a etapa de segmentação da imagem, uma vez que quando um dos revestimentos da ferramenta é totalmente desgastado, muda-se a textura e a intensidade dos pixels. Para isso tanto a

utilização de uma câmera colorida pode ser explorada quanto a utilização de processamento de imagens em níveis de cinza, atentando-se para o fato de que diferentes níveis poderão estar representando diferentes cores.

A classificação do sistema também pode ser expandida, como pode ser visto na figura 7.3. Para isso, no entanto, novos testes devem ser realizados com o sistema. A utilização de um sistema de classificação baseado em uma técnica combinada de redes neurais artificiais e lógica nebulosa pode trazer bons resultados ao sistema. Este mesmo sistema pode ser aplicado à detecção do desgaste de cratera, também em combinação com a técnica de PCA, de acordo com um pré-processamento efetuado. As informações de medição também podem ser utilizadas de forma a melhorar a robustez do sistema de classificação.

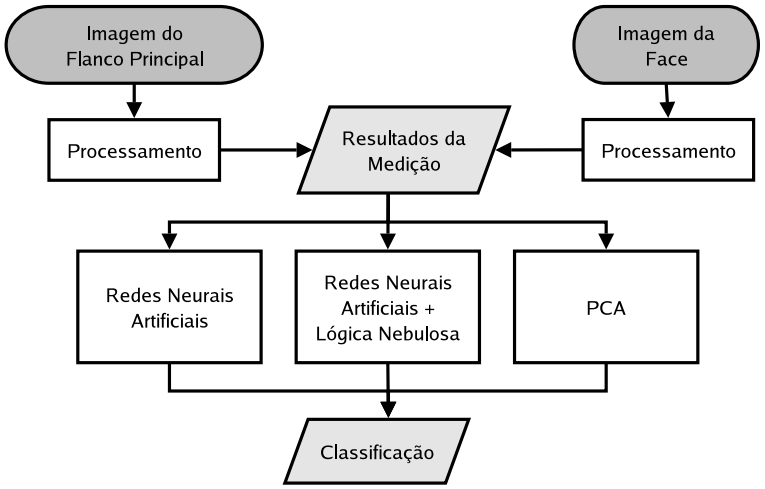


Figura 7.3: *Técnicas de classificação a serem exploradas no futuro desenvolvimento do sistema.*

A própria técnica de classificação baseada em PCA pode ser implementada através de uma RNA de forma a que um algoritmo de aprendizado possa ser utilizado para continuamente treinar a RNA e ajustar os pesos entre os neurônios, que representam, a grosso modo, os coeficientes da matriz de projeção da imagem no espaço de classificação. Assim, a RNA poderia estar sendo treinada a cada vez que um resultado correto fosse encontrado e um operador humano estivesse supervisionando a operação, sendo ela adaptada ao ambiente industrial, aprendendo por exemplos. Um esquema de classificação redundante, utilizando-se a informação de mais de um algoritmo de classificação pode ser estudado para que a robustez do sistema seja aumentada.

O sistema para a determinação do desgaste de cratera pode ser testado de forma a que seja aplicado também à classificação dos demais tipos de desgaste. A técnica baseada em PCA, dependendo da aplicação, pode ser adequada para isto, o que diminuiria bastante o tempo de processamento para que se obtivesse uma informação acerca do tipo do desgaste e, conseqüentemente, do estado da ferramenta.

7.2.4 Perspectivas para o Sistema TOOLSPY

As perspectivas para o sistema TOOLSPY estão na realização de intensivos testes com o sistema sob diferentes condições de operação e com diferentes tipos de ferramentas.

Novos tipos de iluminação devem ser testados pelo sistema de forma a se tentar eliminar a necessidade da aquisição de múltiplas imagens em níveis de cinza. Este é um problema não trivial, uma vez que a superfície da ferramenta reflete de forma bastante irregular a iluminação incidente, sendo que pontos de brilho intenso ocorrem com freqüência.

Alguns tipos de ferramentas diferentes devem ser selecionados e o sistema deve ser ajustado e testado de forma que bons resultados possam ser obtidos para estes determinados tipos de ferramentas. Em particular, a técnica de iluminação desenvolvida deve ser testada, determinando-se a melhor forma de iluminar determinada ferramenta através do uso da cúpula de LEDs (acendendo todos os LEDs de um dos níveis, acendendo apenas uma linha de LEDs de mais de um nível, entre outros). Além disso, ajustes à cadeia de processamento de imagens e visão computacional devem ser realizados, bem como a correta especificação do subsistema óptico para o sistema. O ajuste da cadeia de processamento envolve desde o treinamento da RNA como a determinação do espaço de classificação para a técnica de classificação baseada em PCA, além da seleção dos parâmetros adequados dos demais algoritmos de processamento.

De forma a que o sistema possa ser intensivamente testado, parcerias com empresas da região devem ser firmadas de forma a que estas forneçam algumas ferramentas bastante utilizadas para que o sistema possa ser ajustado para as mesmas. Em fevereiro de 2004, uma parceria com a empresa WEG foi firmada, sendo que está fornecerá ferramentas de metal-duro revestido e ferramentas cerâmicas para que o sistema seja ajustado e testado com elas.

A utilização de uma câmera embarcada é um outro ponto a ser explorado. Para que o sistema possa funcionar de forma autônoma na estação de medição montada junto à máquina-ferramenta, o sistema deve ser executado neste dispositivo. Além disso, o sistema completo, sendo executado nesta câmera embarcada, deve ser implementado na forma de um dos sensores da rede de sensores e atuadores da célula autônoma de produção como descrito por Orth [115] e Roloff [18]. A validade deste modelo deve ser testada em uma máquina-ferramenta em operação no meio industrial, com o sensor conectado à rede de sensores e atuadores, podendo este então ser configurado através da rede pelo protocolo HTTP. Além disso, a construção do protótipo do sistema de transporte da ferramenta também deve ser feita de forma que a estação de medição possa ser efetivamente integrada a uma máquina em condições de operação industrial.

Outro ponto de extrema importância que ainda não foi abordado por nenhum dos trabalhos realizados até agora é um estudo sistemático acerca das incertezas de medição associadas ao sistema. Um levantamento dos erros e deformações associadas ao subsistema óptico, do erro associado à digitalização das imagens e a calibração do sistema devem ser realizadas de forma que um grau de confiabilidade da medição possa ser fornecido juntamente com o resultado final da medição. Sem o conhecimento deste grau de incerteza a informação proveniente deste sensor é incompleta.

7.3 Considerações Finais

Este trabalho não seria possível sem uma estreita integração entre as equipes brasileira e alemã de desenvolvimento do sistema TOOLSPY. Muitas são as dificuldades no desenvolvimento de um trabalho envolvendo equipes separadas por grandes distâncias geográficas, principalmente no tocante à comunicação.

Através do intercâmbio de estudantes entre Brasil e Alemanha, muito da tecnologia desenvolvida e do conhecimento produzido neste projeto pôde ser trocado. Além disso, um *workshop* realizado no início do mês de dezembro de 2003 fez com que muitos aspectos do desenvolvimento do projeto pudessem ser discutidos e aprimorados. Este *workshop* também abriu diversas portas para que contatos fossem estabelecidos com pessoas do setor industrial interessadas no projeto.

Todas estas dificuldades puderam ser superadas devido à grande integração entre as equipes brasileira e alemã do projeto. No presente momento, videoconferências mensais são realizadas para a discussão do andamento do mesmo, bem como e-mails são trocados através de uma lista de discussão.

Uma sugestão para um plano de trabalho para os próximos períodos do projeto, em que determinada áreas devem ser verificadas segue:

- Obtenção de imagens de um determinado tipo de ferramenta em diferentes condições, com diferentes tipos de desgaste e diferentes níveis dos mesmos tipos de desgaste. Estas imagens devem ser classificadas e o desgaste em cada uma das ferramentas deve ser avaliado de forma a que um grande banco de dados com imagens de ferramentas desgastadas seja obtido para testes. Isto deve ser feito em parceria com empresas da região que utilizam ferramentas de corte em grande escala na sua atividade produtiva diária.
- Estudo da influência de diferentes tipos de iluminação no problema. Técnicas de iluminação diferentes podem agora ser testadas a partir da cadeia de processamento de imagens definida para que seus resultados sejam validados. Um estudo sistemático neste sentido deve verificar a influência destes tipos de iluminação para o problema.
- Realização de testes com novos algoritmos de segmentação para o sistema, como por exemplo segmentação através de operadores de textura e gradiente e segmentação através de informações de cores, de forma a que se tente eliminar a necessidade de utilização da imagem de uma ferramenta não desgastada.
- Realização de testes com novos algoritmos para a classificação do desgaste da ferramenta, como, por exemplo, testar a utilização de uma técnica combinada de redes neurais artificiais e lógica nebulosa e o uso da técnica de classificação baseada em PCA para a classificação dos diferentes tipos de desgaste.
- Integração do sistema à câmera embarcada já disponível para testes do sistema, fazendo com que esta se comporte como um dos sensores da célula autônoma de produção. Para isso, o dispositivo de iluminação e aquisição deve ser programado de forma adequada, juntamente com a especificação do subsistema óptico.

Apêndice A

Modelagem da API IPFRAMEWORK

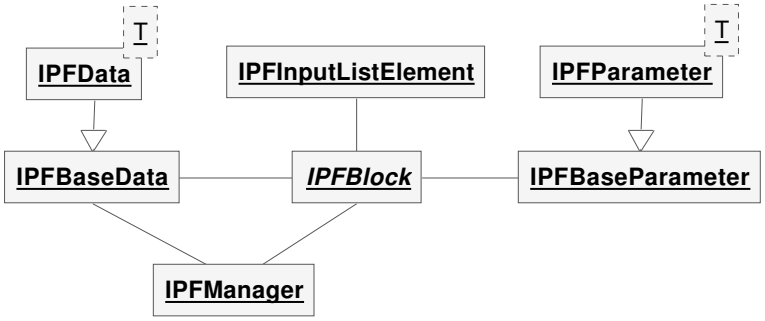


Figura A.1: *Diagrama de classes simplificado da API* IPFRAMEWORK.

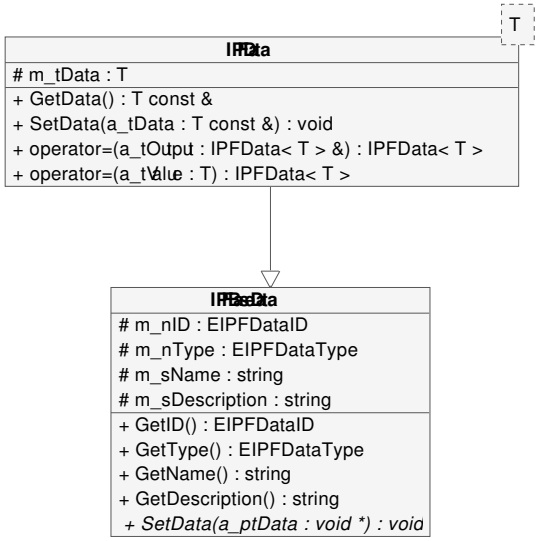


Figura A.2: *Diagrama das classes IPFBaseData e IPFData da API* IPFRAMEWORK.

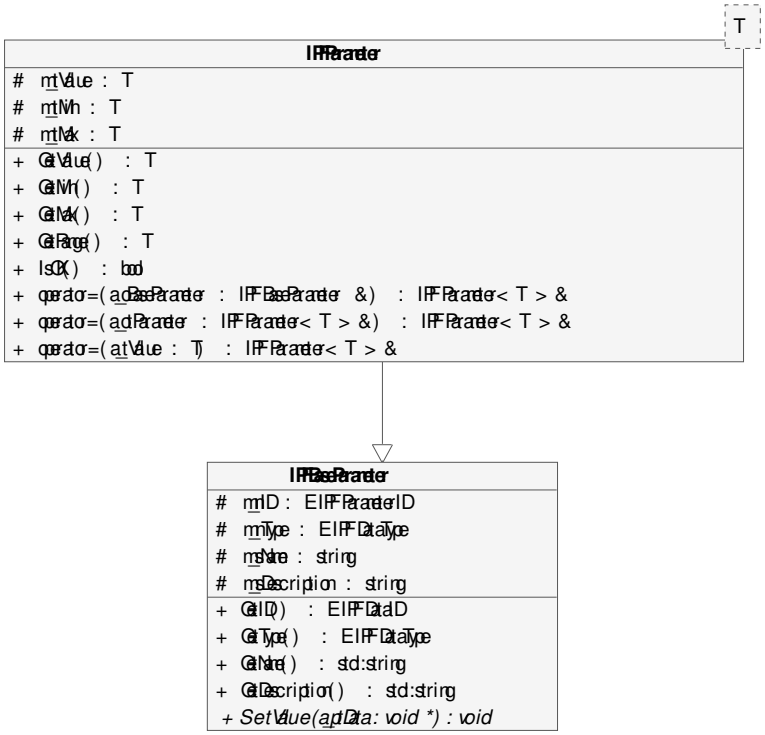


Figura A.3: *Diagrama das classes IPFBaseParameter e IPFParameter da API* IPFRAMEWORK.

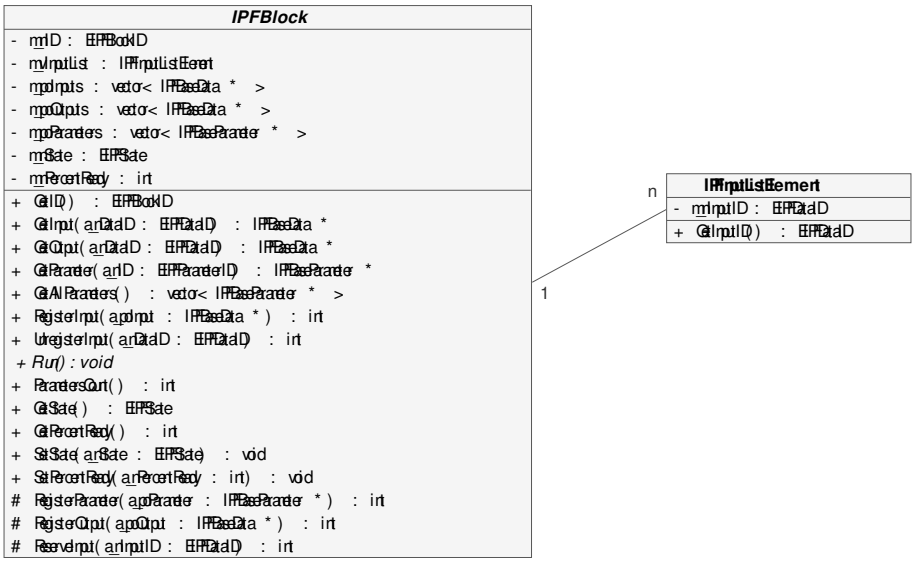


Figura A.4: *Diagrama das classes IPFBlock e IPFInputListElement da API* IPFRAMEWORK.

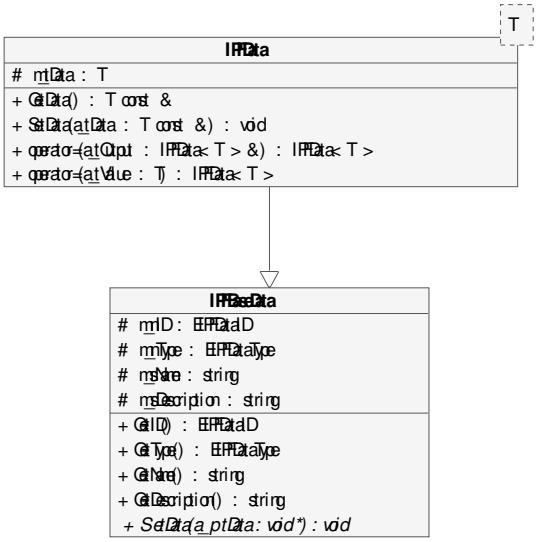


Figura A.5: *Diagrama da classe IPFManager da API* IPFRAMEWORK.

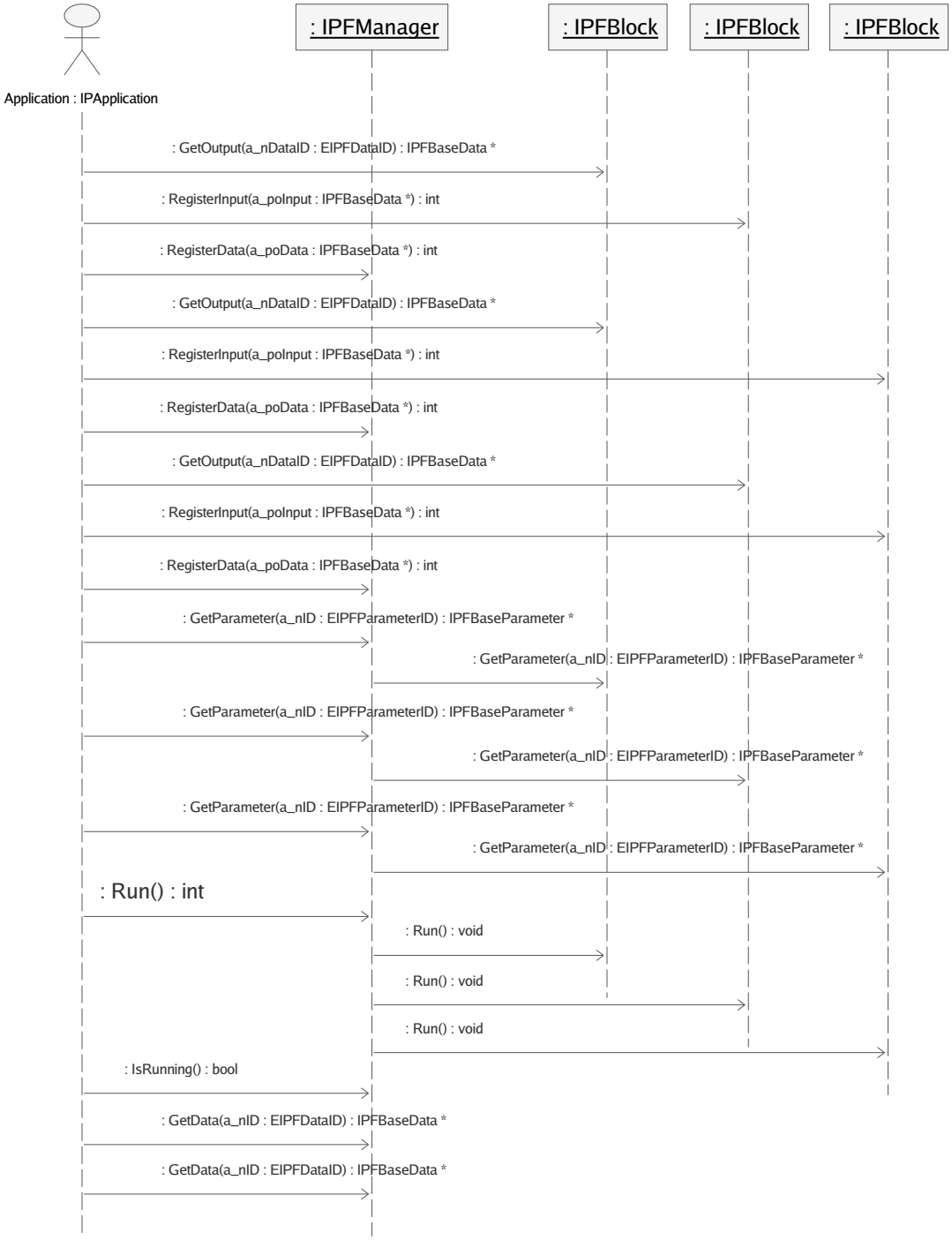


Figura A.6: *Diagrama de seqüências mostrando a criação de uma cadeia de processamento de imagens e visão computacional com a API* IPFRAMEWORK.

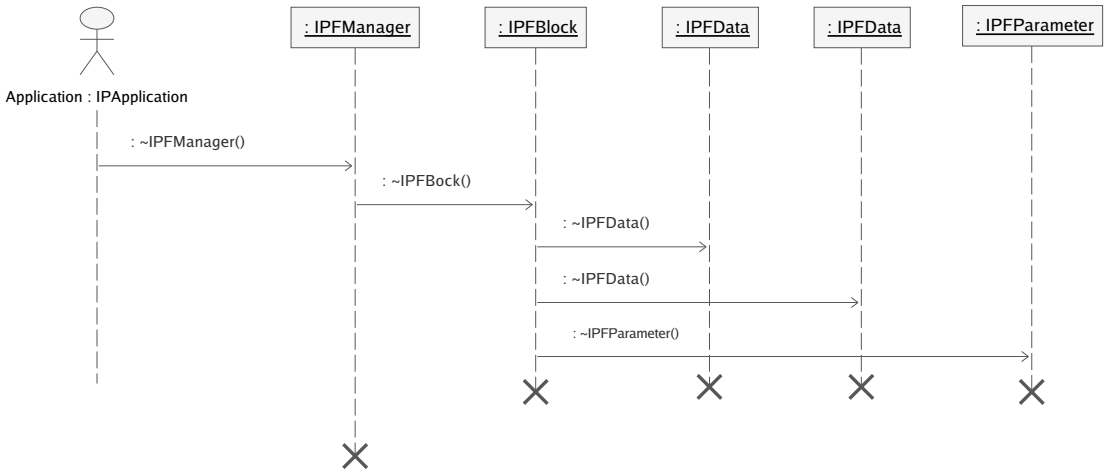


Figura A.7: *Diagrama de seqüências mostrando a criação dos objetos da cadeia de processamento de imagens e visão computacional com a API* IPFRAMEWORK.

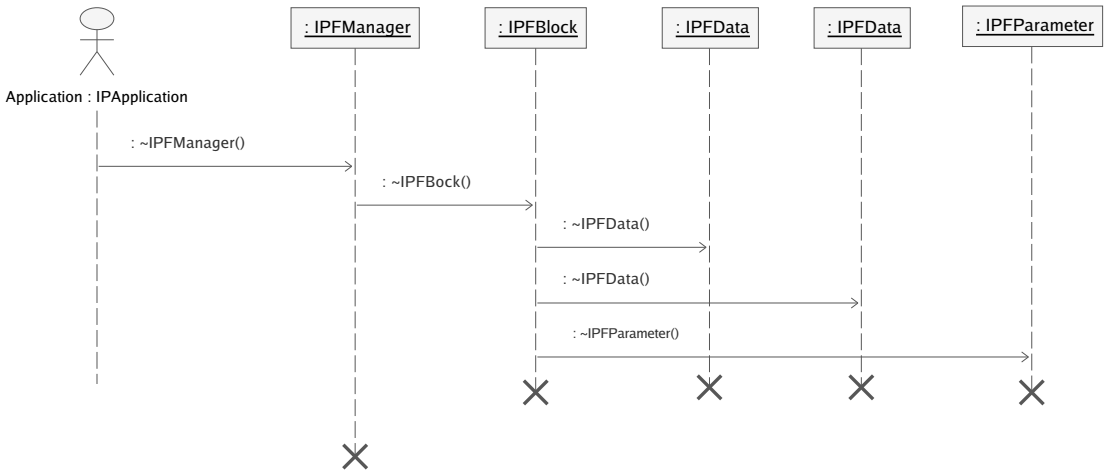


Figura A.8: *Diagrama de seqüências mostrando a destruição dos objetos criados com a API* IPFRAMEWORK.

Apêndice B

Acesso ao CVS do DAS e ao Código-Fonte da S2iLib

O Departamento de Automação e Sistemas disponibiliza um servidor CVS[1] para uso no desenvolvimento de seus projetos. O endereço IP deste servidor é traduzido pelo nome cvs.das.ufsc.br. O modo de acesso é o *pserver*. Para acessá-lo, deve-se ter um programa CVS cliente instalado de forma apropriada na máquina.

Para acesso ao respositório do S2i dentro deste servidor de CVS, a variável CVSROOT deve ser apropriadamente determinada como :pserver:anonymous@cvs.das.ufsc.br:/usr/cvs/s2i. O código-fonte deste repositório está dividido em duas árvores principais:

***generic*:** contém código-fonte escrito pelos membros do grupo S2i para compilação com a família de compiladores GCC em GNU/Linux.

***vc++6:*:** contém código-fonte escrito pelos membros do grupo S2i para compilação com o compilador Visual C++ 6.0 da Microsoft.

Dentro da árvore *generic* existem diferentes bibliotecas e programas. Entre eles, encontramos:

S2iLib: biblioteca para o desenvolvimento de sistemas de visão, composta por diferentes módulos.

GTKRap: programa para o reconhecimento de peças a partir da aquisição das imagens e a utilização de uma RNA.

Toolspy: Programa para a medição do desgaste de ferramentas de corte utilizando a biblioteca S2iLib e os seus diferentes módulos.

[1] Para maiores informações sobre controle de versões através de CVS, visitar o endereço http://www.cvshome.org.

Todos estes programas e bibliotecas podem ser diretamente acessados através da operação de *check-out* do CVS a partir de seus nomes, em letras minúsculas. Dentro da biblioteca S2iLib encontramos ainda os seus diferentes módulos:

S2iImage: módulo para a manipulação de imagens e ROIs e suas operações básicas.

S2iFourier: módulo para o cálculo da FFT de um sinal e sua inversa.

S2iWavelet: módulo para o cálculo da Transformada *Wavelet* de um sinal e sua inversa.

S2iNeural: módulo para a implementação de RNAs.

S2iFilter: módulo para o cálculo de filtros para imagens.

S2iMorphology: módulo para a aplicação de operadores de morfologia matemática a imagens.

S2iEigenfaces: módulo para a implementação da técnica de Eigenfaces.

S2iGrab: módulo para o interfaceamento com disposisitivos de aquisição de imagens como placas de aquisição de imagens, *scanners* e *webcams*.

S2iIlumination: módulo para o interfaceamento com disposisitivos de iluminação.

S2iSegmentation: módulo para a aplicação de operadores de segmentação a uma imagem.

S2iIPBlocks: módulo dos diversos blocos criados a partir da API IPFramework e dos demais módulos da S2iLib.

Estes módulos podem ser diretamente acessados através da operação de *check-out* do CVS a partir de seus nomes, em letras minúsculas.

Dentro da árvore *vc++6* existem diferentes bibliotecas e programas. Dentre eles, encontramos:

S2iLib: biblioteca para o desenvolvimento de sistemas de visão, composta pelos mesmos módulos descritos anteriormente, versão para o compilador Visual C++ 6.0 da Microsoft.

ToolSpy: programa para a medição do desgaste de ferramentas de corte utilizando a biblioteca MtqLib, a biblioteca S2iLib e os seus diferentes módulos.

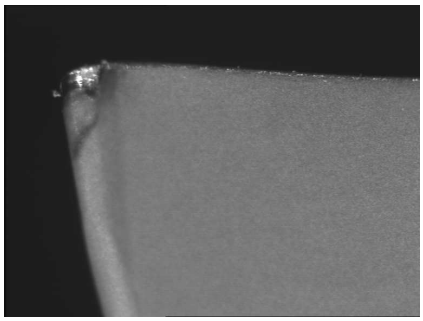
ToolSpyOptimizer: programa para a otimização dos parâmetros da cadeia de processamento de imagens do sistema ToolSpy.

Todos estes programas e bibliotecas podem ser diretamente acessados através da operação de *check-out* do CVS a partir de seus nomes, em letras minúsculas.

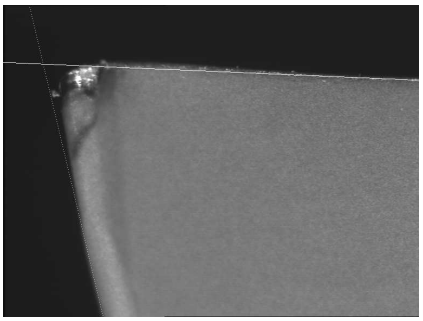
Apêndice C

Imagens das Medições Realizadas

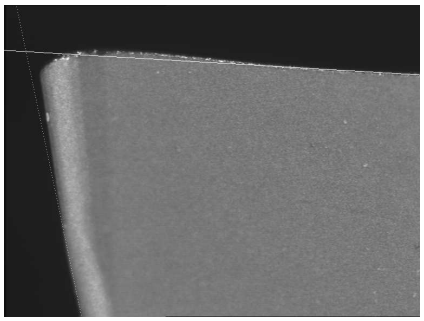
A seguir são apresentadas as imagens intermediárias da aplicação da cadeia de processamento de imagens e visão computacional para a avaliação do desgaste como analisado na seção 6.2. São apresentadas as imagens otimizadas em níveis de cinza das ferramentas desgastadas, as imagens otimizadas em níveis de cinza com as retas dos gumes das ferramentas desgastadas e das ferramentas modelo, as imagens otimizadas em níveis da cinza das ferramentas desgastadas transladadas e rotacionadas, as imagens diferenças das ROIs da região de desgaste entre as imagens otimizadas em níveis de cinza das ferramentas desgastadas e das ferramentas modelo, as imagens diferenças filtradas, as imagens diferenças filtradas após a segmentação por pirâmides e as imagens após a segmentação por *snakes*.



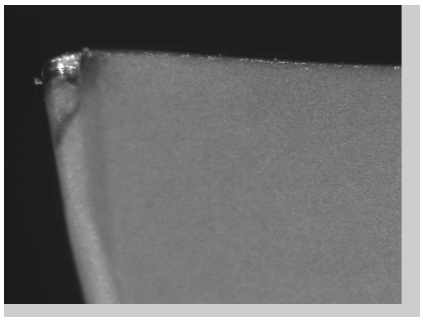
(a) Ferramenta desgastada.



(b) Retas dos gumes.

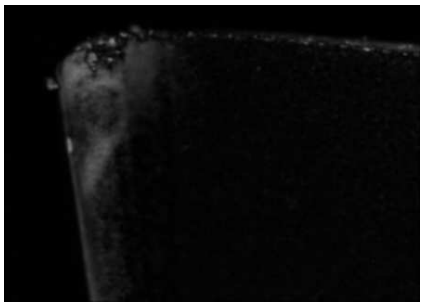


(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.

(e) Imagem diferença.

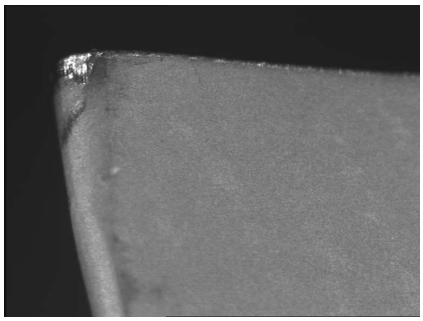


(f) Imagem diferença filtrada.

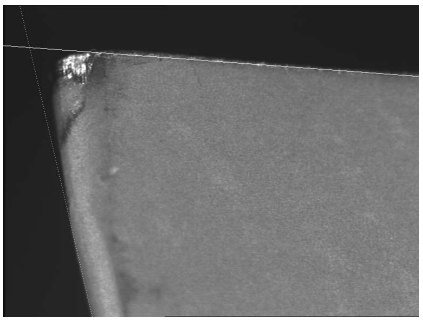
(g) Segmentação por pirâmides.

(h) Segmentação por *snakes*.

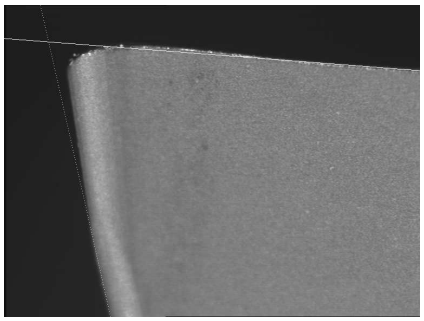
Figura C.1: *Ferramenta 1, medição 1.*



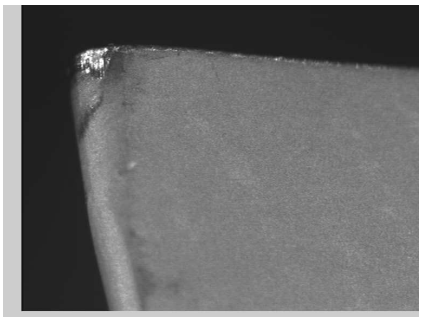
(a) Ferramenta desgastada.



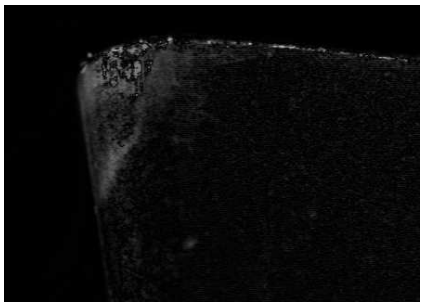
(b) Retas dos gumes.



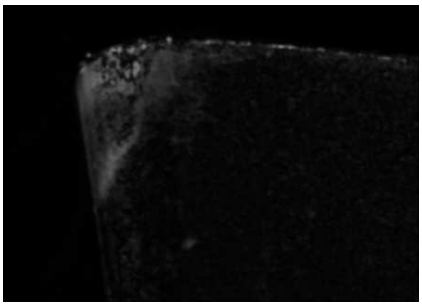
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.



(e) Imagem diferença.

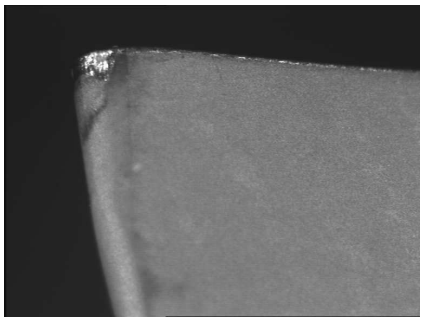


(f) Imagem diferença filtrada.

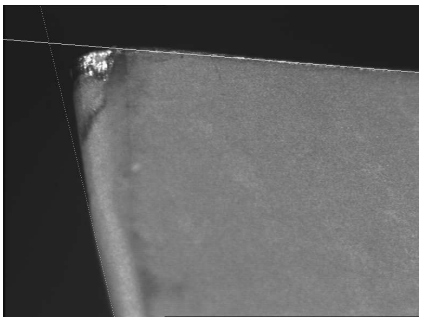
(g) Segmentação por pirâmides.

(h) Segmentação por *snakes*.

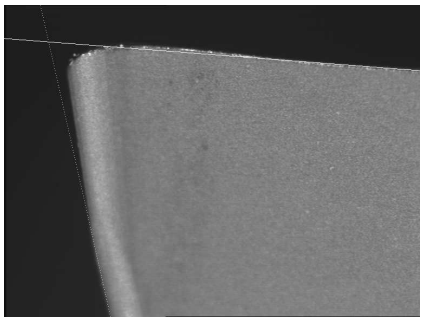
Figura C.2: *Ferramenta 1, medição 2.*



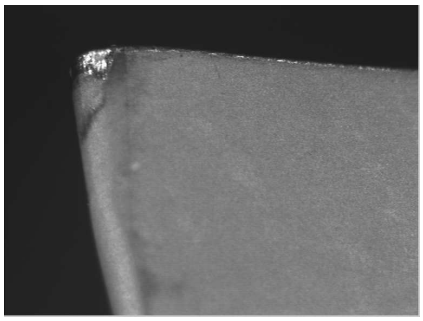
(a) Ferramenta desgastada.



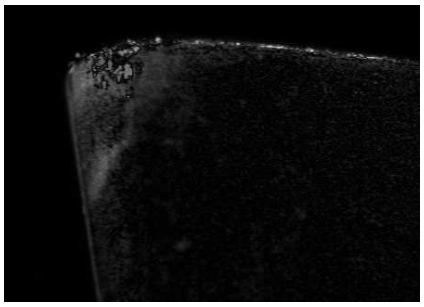
(b) Retas dos gumes.



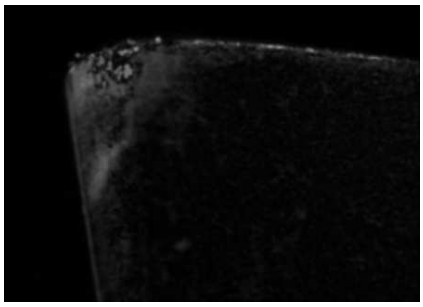
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.



(e) Imagem diferença.

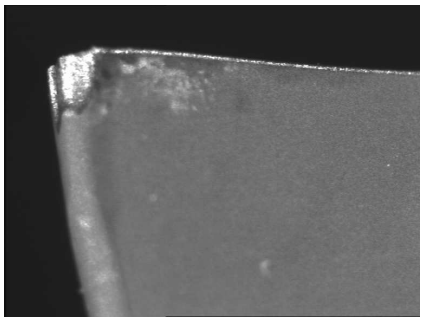


(f) Imagem diferença filtrada.

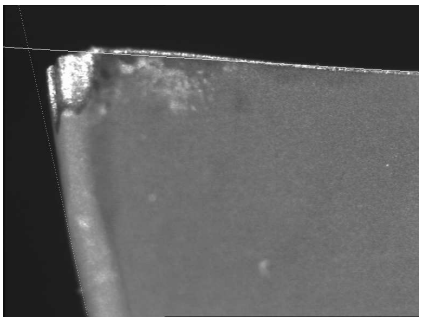
(g) Segmentação por pirâmides.

(h) Segmentação por *snakes*.

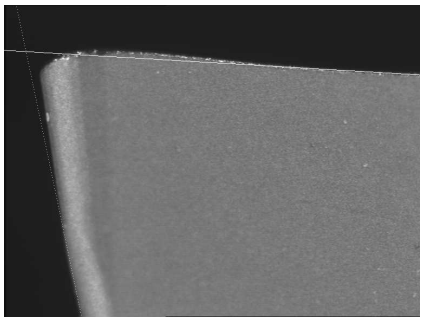
Figura C.3: *Ferramenta 1, medição 3.*



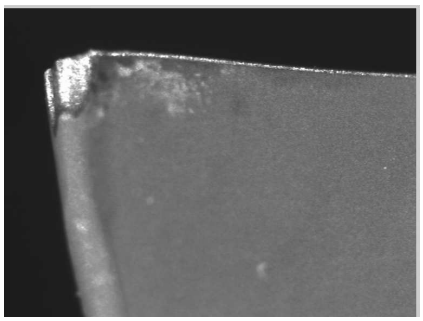
(a) Ferramenta desgastada.



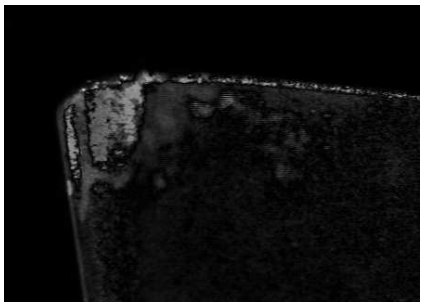
(b) Retas dos gumes.



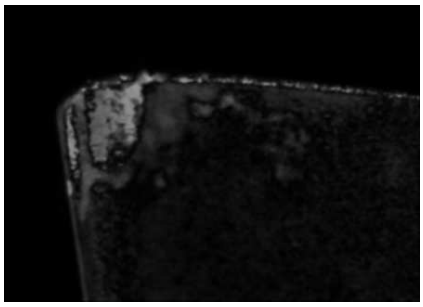
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.



(e) Imagem diferença.

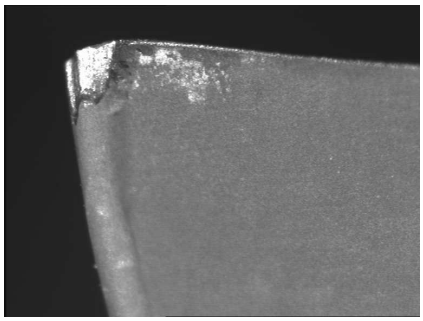


(f) Imagem diferença filtrada.

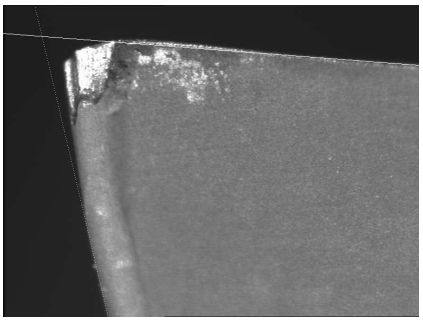
(g) Segmentação por pirâmides.

(h) Segmentação por *snakes*.

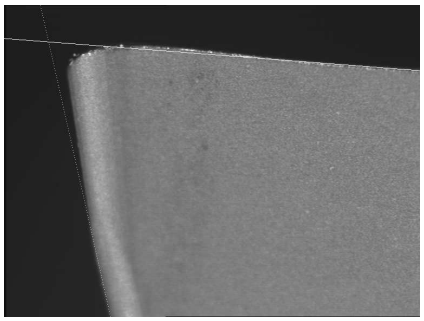
Figura C.4: *Ferramenta 2, medição 4.*



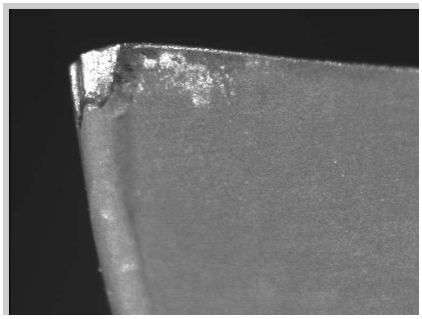
(a) Ferramenta desgastada.



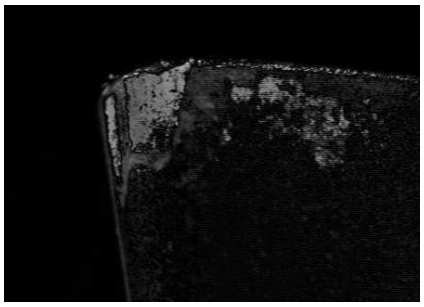
(b) Retas dos gumes.



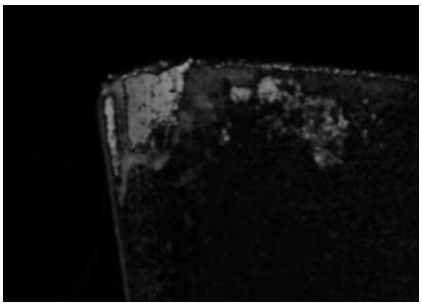
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.



(e) Imagem diferença.

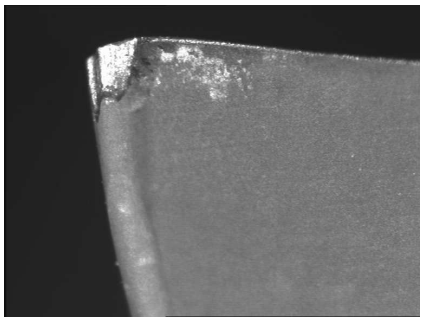


(f) Imagem diferença filtrada.

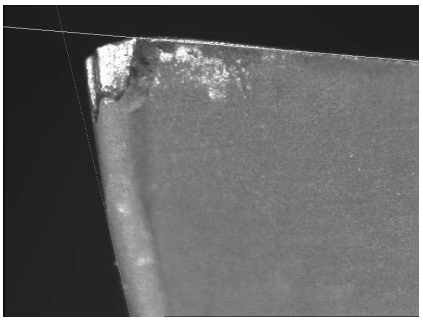
(g) Segmentação por pirâmides.

(h) Segmentação por *snakes*.

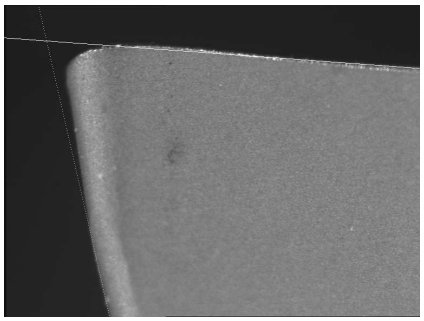
Figura C.5: *Ferramenta 2, medição 5.*



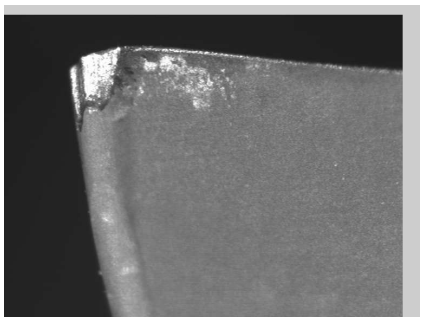
(a) Ferramenta desgastada.



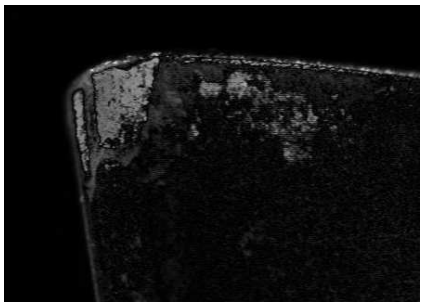
(b) Retas dos gumes.



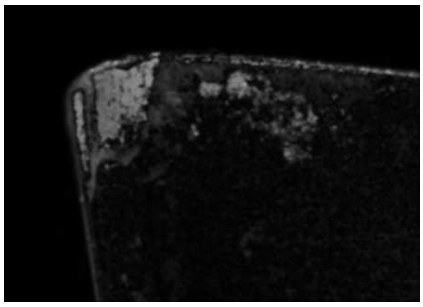
(c) Modelo, retas dos gumes.



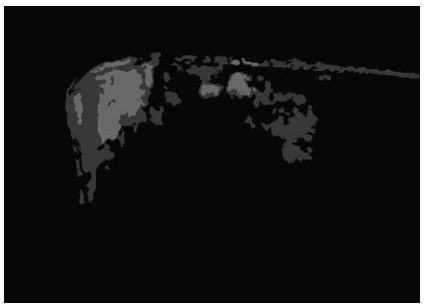
(d) Imagem transladada e rotacionada.



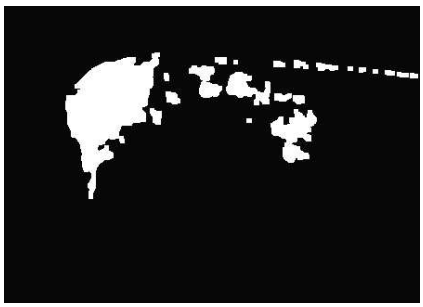
(e) Imagem diferença.



(f) Imagem diferença filtrada.

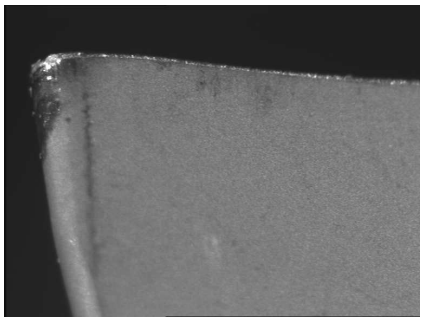


(g) Segmentação por pirâmides.

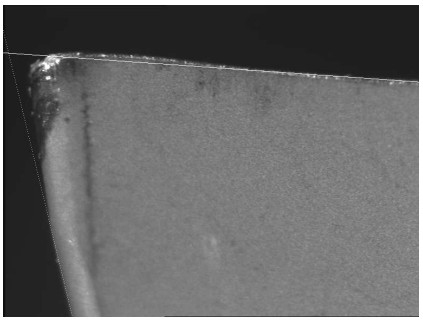


(h) Segmentação por *snakes*.

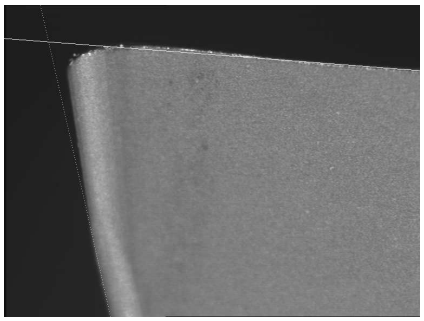
Figura C.6: *Ferramenta 2, medição 6.*



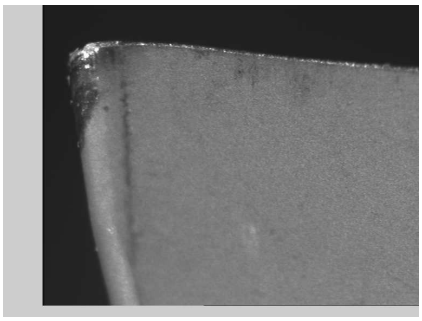
(a) Ferramenta desgastada.



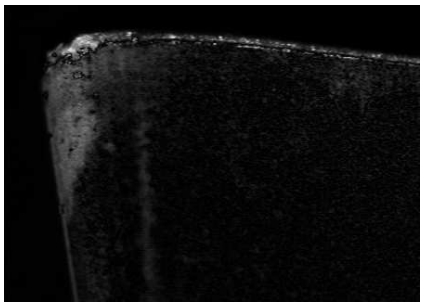
(b) Retas dos gumes.



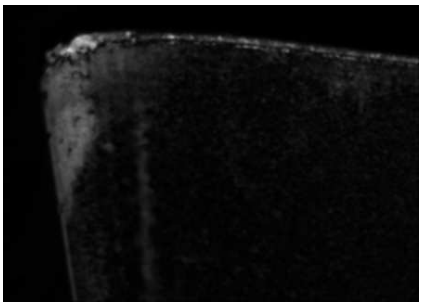
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.



(e) Imagem diferença.

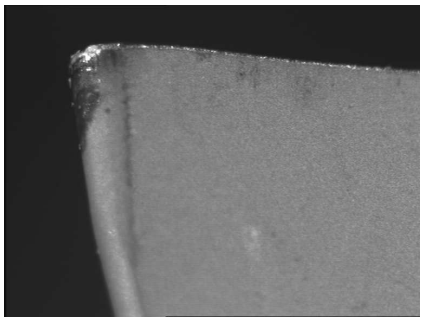


(f) Imagem diferença filtrada.

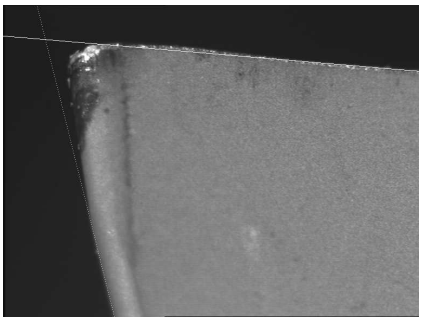
(g) Segmentação por pirâmides.

(h) Segmentação por *snakes*.

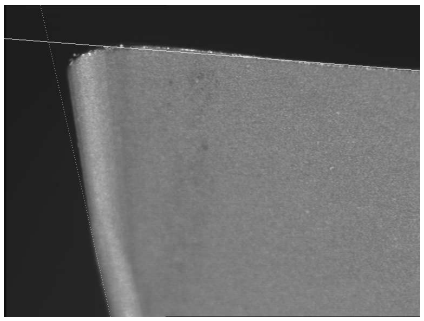
Figura C.7: *Ferramenta 3, medição 7.*



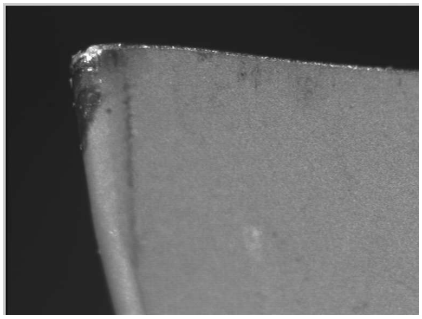
(a) Ferramenta desgastada.



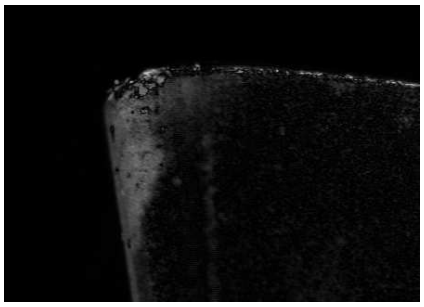
(b) Retas dos gumes.



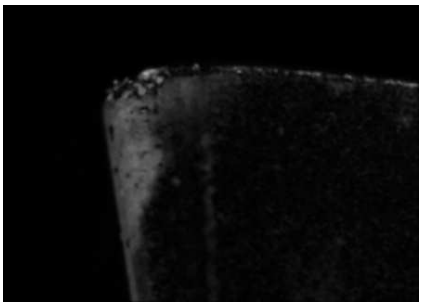
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.



(e) Imagem diferença.

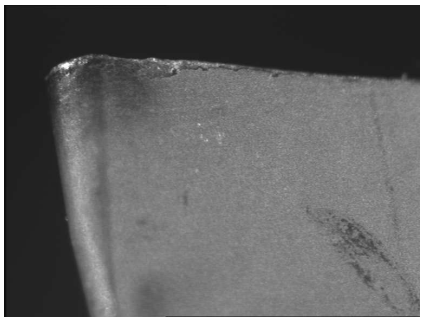


(f) Imagem diferença filtrada.

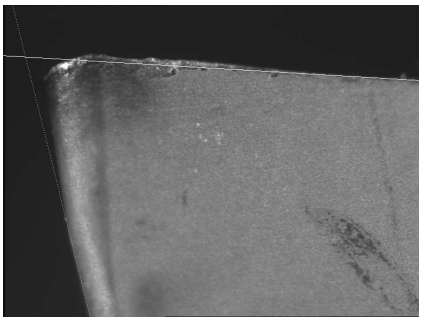
(g) Segmentação por pirâmides.

(h) Segmentação por *snakes*.

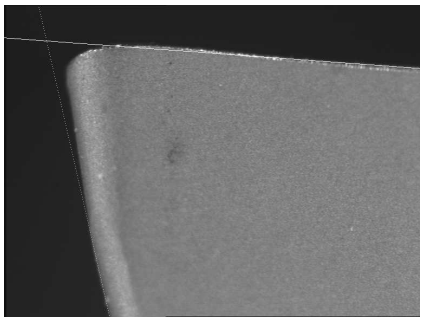
Figura C.8: *Ferramenta 3, medição 8.*



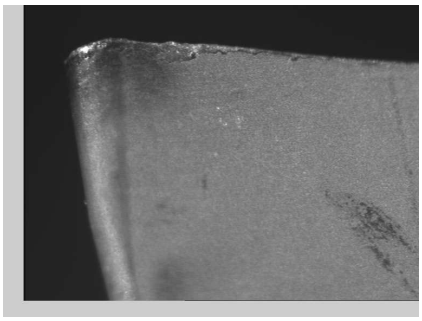
(a) Ferramenta desgastada.



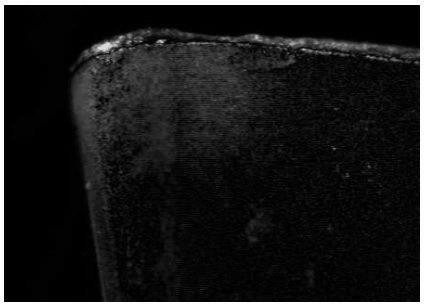
(b) Retas dos gumes.



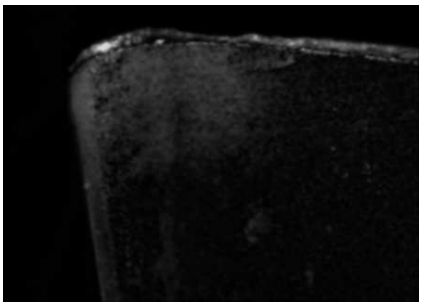
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.

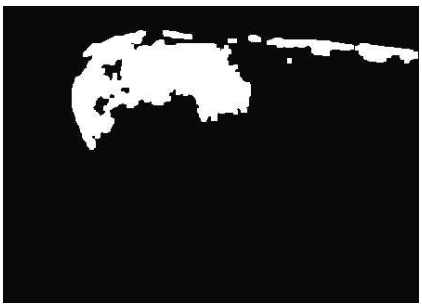


(e) Imagem diferença.



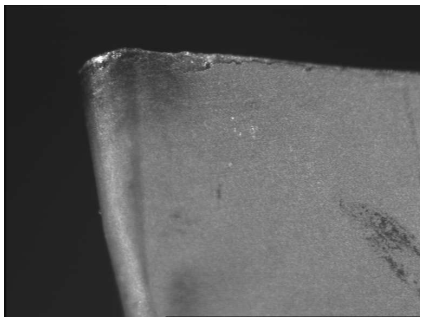
(f) Imagem diferença filtrada.

(g) Segmentação por pirâmides.

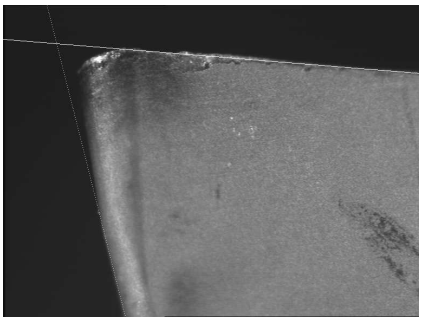


(h) Segmentação por *snakes*.

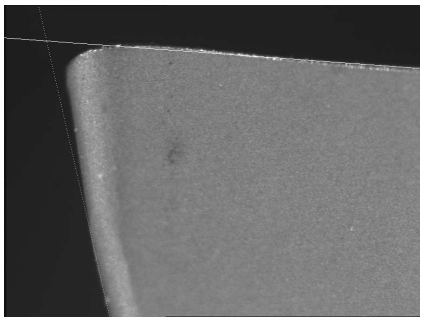
Figura C.9: *Ferramenta 4, medição 10.*



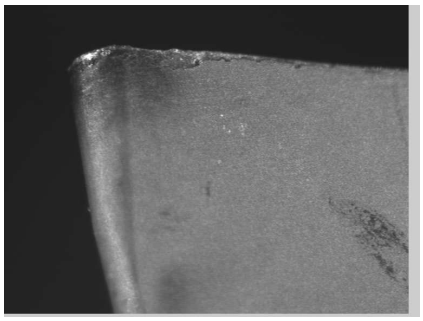
(a) Ferramenta desgastada.



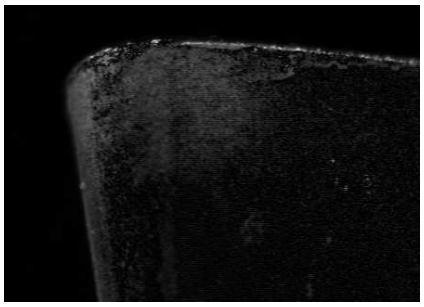
(b) Retas dos gumes.



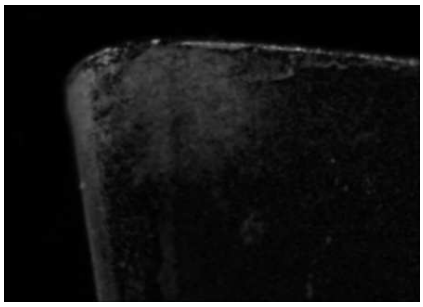
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.

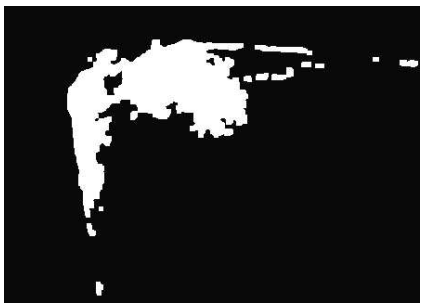


(e) Imagem diferença.



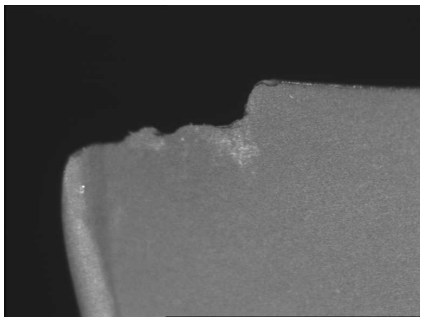
(f) Imagem diferença filtrada.

(g) Segmentação por pirâmides.

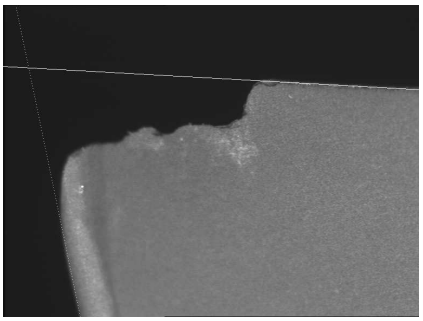


(h) Segmentação por *snakes*.

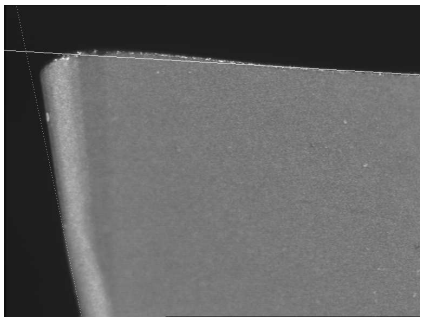
Figura C.10: *Ferramenta 4, medição 11.*



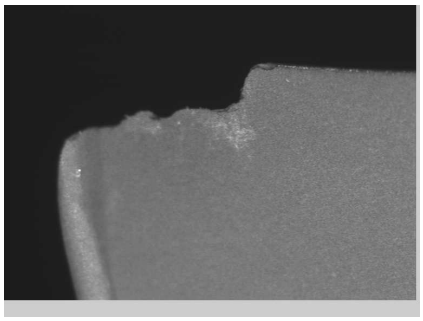
(a) Ferramenta desgastada.



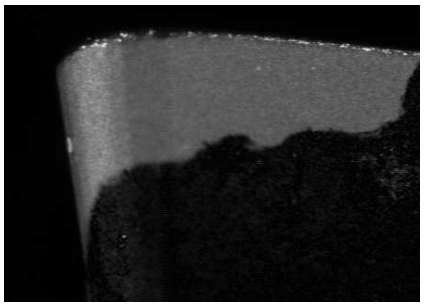
(b) Retas dos gumes.



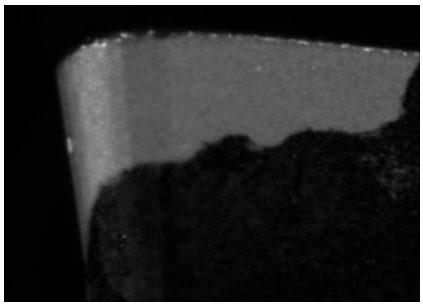
(c) Modelo, retas dos gumes.



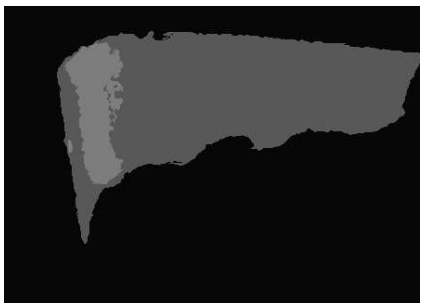
(d) Imagem transladada e rotacionada.



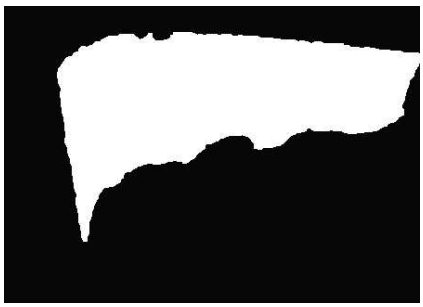
(e) Imagem diferença.



(f) Imagem diferença filtrada.

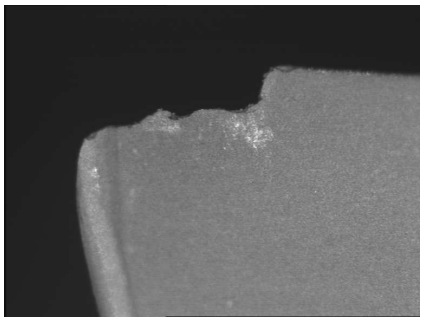


(g) Segmentação por pirâmides.

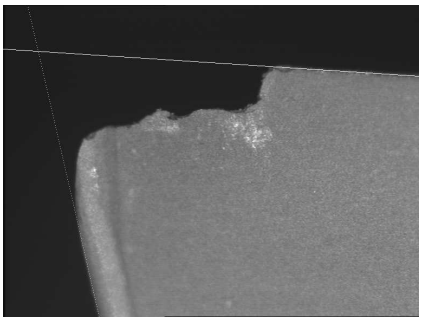


(h) Segmentação por *snakes*.

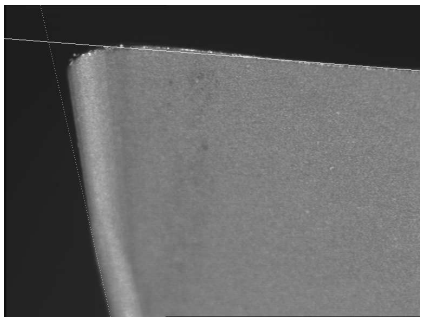
Figura C.11: *Ferramenta 5, medição 12.*



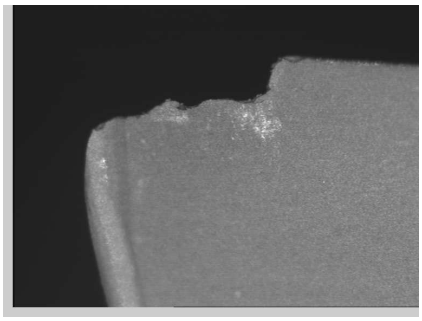
(a) Ferramenta desgastada.



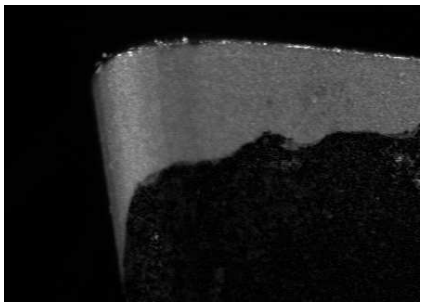
(b) Retas dos gumes.



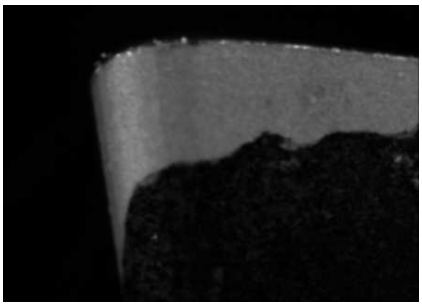
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.

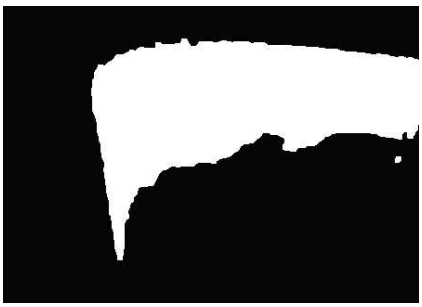


(e) Imagem diferença.



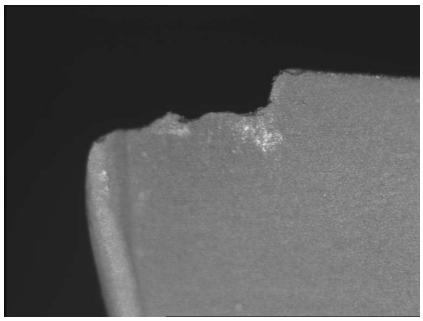
(f) Imagem diferença filtrada.

(g) Segmentação por pirâmides.

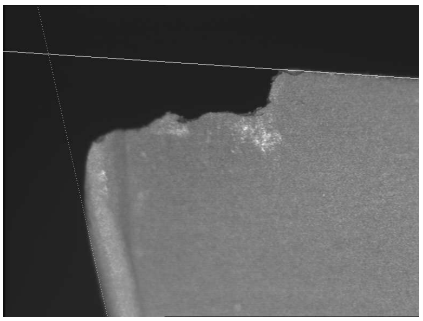


(h) Segmentação por *snakes*.

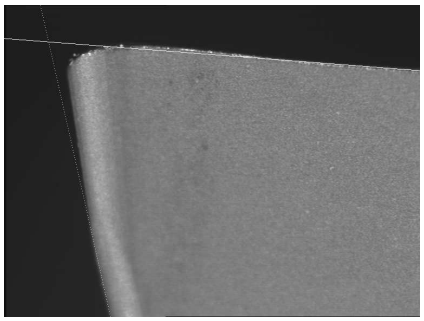
Figura C.12: *Ferramenta 5, medição 13.*



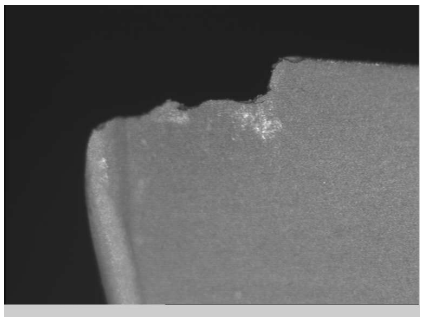
(a) Ferramenta desgastada.



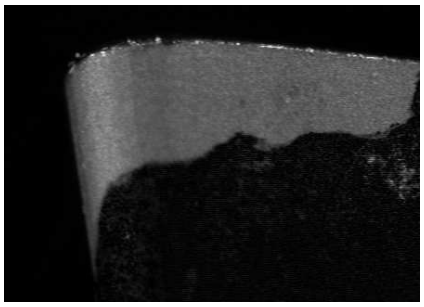
(b) Retas dos gumes.



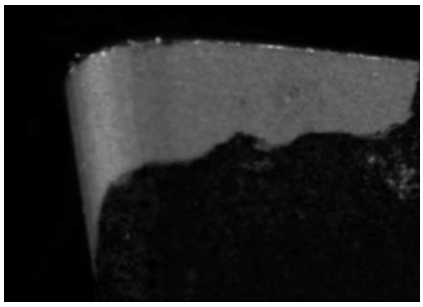
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.



(e) Imagem diferença.

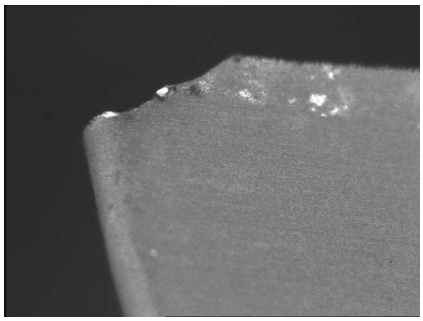


(f) Imagem diferença filtrada.

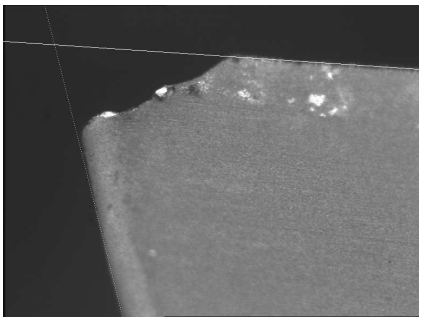
(g) Segmentação por pirâmides.

(h) Segmentação por *snakes*.

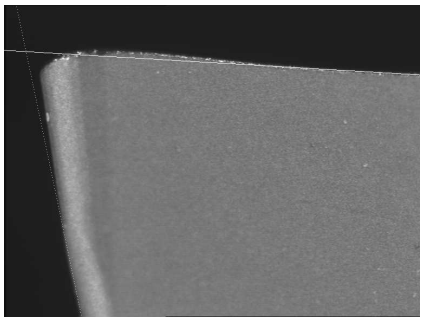
Figura C.13: *Ferramenta 6, medição 14.*



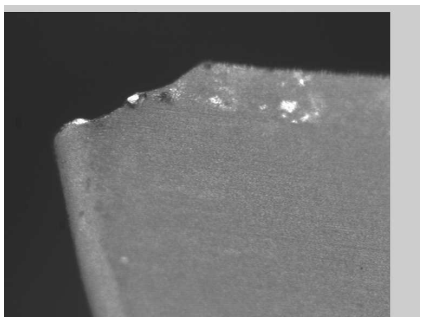
(a) Ferramenta desgastada.



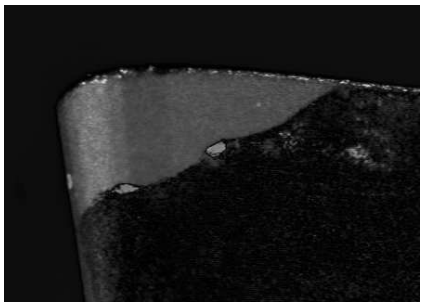
(b) Retas dos gumes.



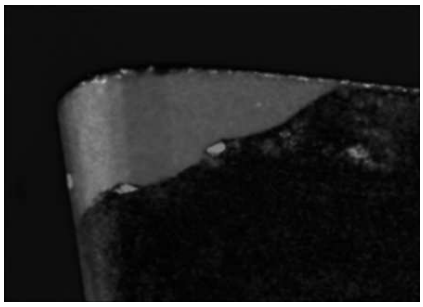
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.

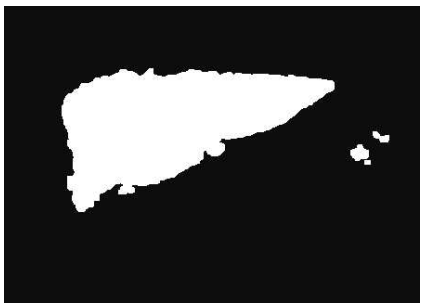


(e) Imagem diferença.



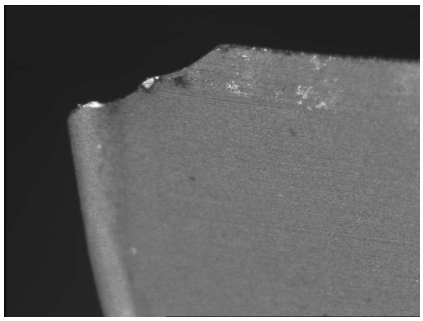
(f) Imagem diferença filtrada.

(g) Segmentação por pirâmides.

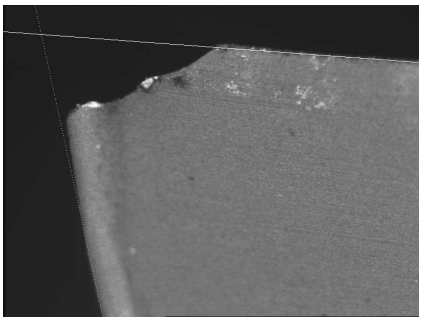


(h) Segmentação por *snakes*.

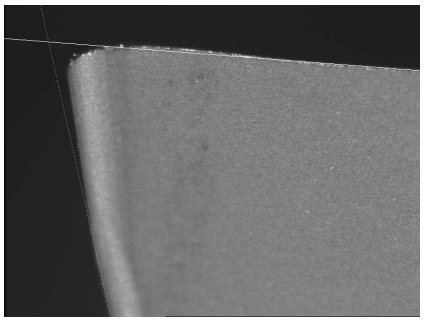
Figura C.14: *Ferramenta 6, medição 15.*



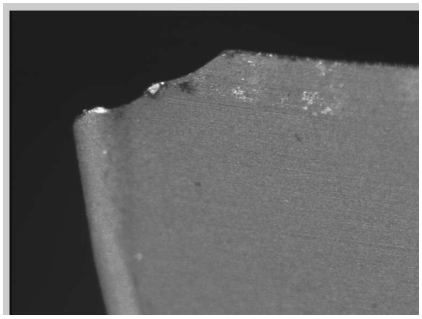
(a) Ferramenta desgastada.



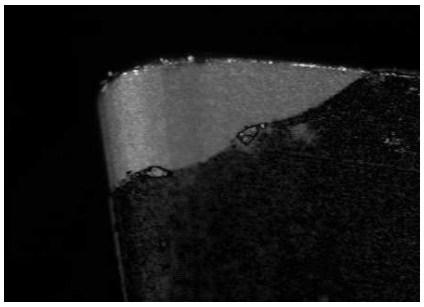
(b) Retas dos gumes.



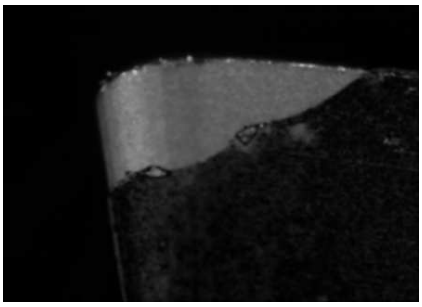
(c) Modelo, retas dos gumes.



(d) Imagem transladada e rotacionada.

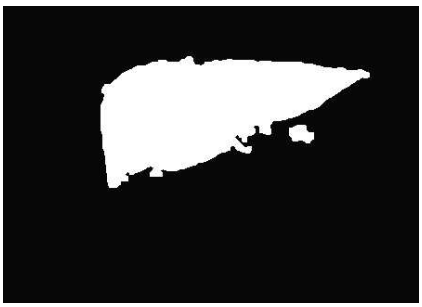


(e) Imagem diferença.



(f) Imagem diferença filtrada.

(g) Segmentação por pirâmides.



(h) Segmentação por *snakes*.

Figura C.15: *Ferramenta 6, medição 16.*

Referências Bibliográficas

[1] R. Du, M. A. Elbestawi, e S. M. Wu. Automated monitoring of manufacturing processes, part 1: Monitoring methods. *Journal of Engineering for Industry*, 117(2):121–132, 1995.

[2] R. Du, M. A. Elbestawi, e S. M. Wu. Automated monitoring of manufacturing processes, part 2: Applications. *Journal of Engineering for Industry*, 117(2):133–141, 1995.

[3] G. Byrne, D. Dornfeld, I. Inasaki, G. Ktteler, W. König, e R. Teti. Tool condition monitoring (tcm) - the status of research and industrial applications. In *Annals of the CIRP*, pág. 541–567, 1994.

[4] B. Sick. On-line and indirect tool wear monitoring in turning with artificial neural networks: a review of more than a decade of research. *Mechanical Systems and Signal Processing*, 16(4): 487–546, 2002.

[5] S. Kurada e C. Bradley. A review of machine vision sensors for tool condition monitoring. *Computers in Industry*, 34:55–72, 1997.

[6] K. Jemielniak. Commercial tool condition monitoring systems. *The International Journal of Advanced Manufacturing Technology*, 15:711–721, 1999.

[7] D. W. Cho, S. J. Lee, e C. N. Chu. The state of machining process monitoring research in korea. *International Journal of Machine Tools and Manufacture*, 39:1697–1715, 1999.

[8] T. Pfeifer e L. Wiegers. Reliable tool wear monitoring by optimized image and illumination control in machine vision. *Measurement*, 28:209–218, 2000.

[9] G. O'Donnel, P. Young, K. Kelly, e G. Byrne. Towards the improvement of tool condition monitoring systems in the manufacturing environment. *Journal of Materials Processing Technology*, 119:133–139, 2001.

[10] J. A. Freeman e D. M. Skapura. *Neural Networks: Algorithms, Applications and Programming Techniques*. Addison-Wesley, 1 ed., 1991.

[11] Z. L. Kovács. *Redes Neurais Artificiais: Fundamentos e Aplicações*. Collegium Cognitio, 2 ed., 1996.

[12] G. Bittencourt. *Inteligência Artificial: Ferramentas e Teorias*. Editora da UFSC, 1 ed., 1998.

[13] E. Rich e K. Knight. *Inteligência Artificial*. Makron Books, 1 ed., 1993.

[14] E. Schöneburg, F. Heinzmann, e S. Feddersen. *Genetische Algorithmen un Evolutionsstrategien*. Addison-Wesley, 1 ed., 1994.

[15] V. Nissen. *Einführung in Evolutionäre Algorithmen: Optimierung nach dem Vorbild der Evolution*. Vieweg, 1 ed., 1997.

[16] H. P. Schwefel. *Evolution and Optimum Seeking*. John Wiley and Sons, 1 ed., 1994.

[17] A. Orth. *Sistema de Visão Aplicado à Medição do Desgaste de Ferramentas de Corte*. Dissertação (mestrado em engenharia elétrica), Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2001.

[18] M. L. Roloff. *Monitoramento Remoto*. Dissertação (mestrado em engenharia elétrica), Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2003.

[19] A. Pentland e M. Turk. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1): 71–86, 1991.

[20] R. S. Pressman. *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 4 ed., 1996.

[21] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, e W. Lorensen. *Modelagem e Projetos Baseados em Objetos*. Campus, 8 ed., 1994.

[22] P. Jalote. *Software Project Management in Practice*. Addison-Wesley, 1 ed., 2002.

[23] P. Jalote. *An Integrated Approach to Software Engineering*. Springer, 2 ed., 1991.

[24] M. Fowler e K. Scott. *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley, 2 ed., 1999.

[25] R. C. Lee e W. M. Tepfenhart. *UML and C++: A Practical Guide to Object-Oriented Development*. Prentice-Hall, 1 ed., 2001.

[26] G. Booch, J. Rumbaugh, e I. Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, 1 ed., 1998.

[27] B. Stroustrup. *The C++ Programming Language*. Addison-Wesley, 3 ed., 2000.

[28] N. M. Josuttis. *The C++ Standard Library: A Tutorial and Reference*. Addison-Wesley, 1 ed., 1999.

[29] V. V. Mizrahi. *Treinamento em Linguagem C++ - Módulo 1*. Makron Books, 1 ed., 1994.

[30] V. V. Mizrahi. *Treinamento em Linguagem C++ - Módulo 2*. Makron Books, 1 ed., 1994.

[31] G. Spur e T. Stöferle. *Handbuch der Fertigungstechnik: Spanen*. Carl Hansen Verlag, 5 ed., 1979.

[32] W. König e F. Klocke. *Fertigungsverfahren - Drehen, Fräsen, Bohren*. Springer, 5 ed., 1997.

[33] C. E. Stemmer. *Ferramentas de Corte I*. Editora da UFSC, 5 ed., 1987.

[34] M. Lanzetta. A new flexible high-resolution vision sensor for tool condition monitoring. *Journal of Materials Processing Technology*, 119:73–82, 2001.

[35] S. Kurada e C. Bradley. A machine vision system for tool wear assessment. *Tribology International*, 30:295–304, 1997.

[36] F. Giusti, M. Santochi, e G. Tantussi. On-line sensing of flank and crater wear of cutting tools. In *Annals of CIRP*, volume 36, pág. 41–44, 1987.

[37] K. B. Pedersen. Wear measurement of cutting tools by computer vision. *International Journal of Machine Tools and Manufacturing*, 30:131–139, 1990.

[38] M. Y. Yang e O. D. Kwon. A tool condition recognition system using image processing. *Control Engineering Practice*, 6:1389–1395, 1998.

[39] S. M. Ji, L. B. Zhang, J. L. Yuan, Y. H. Wan, X. Zhang, L. Zhang, e G. J. Bao. Method of monitoring wearing and breakage states of cutting tools based on mahalanobis distance features. *Journal of Materials Processing Technology*, 129:114–117, 2002.

[40] D. C. D. Oguamanam, H. Raafat, e S. M. Taboun. A machine vision system for wear monitoring and breakage detection of single-point cutting tools. *Computers and Industrial Engineering*, 26(3):575–598, 1994.

[41] W. Weis. Processing of optical sensor data for tool monitoring with neural networks. In *WESCON Conference Proceedings*, pág. 351–355, 1994.

[42] W. Weis. Tool wear measurement on basis of optical sensors, vision systems and neuronal networks (application milling). pág. 134–138.

[43] M. Y. Yang e O. D. Kwon. Crater wear measurement using computer vision and automatic focusing. *Journal of Materials Processing Technology*, 58:362–367, 1996.

[44] K. N. Prasad e B. Ramamoorthy. Tool wear evaluation by stereo vision and prediction by artificial neural network. *Journal of Materials Processing Technology*, 112:43–52, 2001.

[45] A. Karthik, S. Chandra, B. Ramamoorthy, e S. Das. 3d tool wear measurement and visualisation using stereo imaging. *International Journal of Machine Tools and Manufacturing*, 37(11): 1573–1581, 1997.

[46] L. Gong e K. Tang. Monitoring machine operations using on-line sensors. *European Journal of Operational Research*, 96:479–492, 1997.

[47] T. Blum, I. Suzuki, e I. Inasaki. Development of a condition monitoring system for cutting tools using an acoustic emission sensor. *Bulletin of the Japan Society of Precision Engineering*, 22 (4):301–308, 1988.

[48] J. Kopac e S. Sali. Tool wear monitoring during the turning process. *Journal of Materials Processing Technology*, 113:312–316, 2001.

[49] M. E. R. Bonifácio. *Monitoramento do Processo de Torneamento de Acabamento via Sinais de Vibração*. Dissertação (mestrado em engenharia mecânica), Universidade Estadual de Campinas, Campinas, 1993.

[50] C. M. Hara. *Utilização de Redes Neurais na Análise de Sinais de Vibração de Ferramenta de Torneamento*. Dissertação (mestrado em engenharia mecânica), Universidade Estadual de Campinas, Campinas, 1995.

[51] A. Ghasempoor, J. Jeswiet, e T. N. Moore. Real time implementation of on-line tool condition monitoring in turning. *Internation Journal of Machine Tools and Manufacture 39*, 39:1883–1902, 1999.

[52] Q. Liu e Y. Altintas. On-line monitoring of flank wear in turning with multilayered feed-forward neural network. *International Journal of Machine Tools and Manufacture*, 39:1945–1959, 1999.

[53] D. K. Baek, T. J. Ko, e H. S. Kim. Real time monitoring of tool breakage in a milling operation using a digital signal processor. *Journal of Materials Processing Technology*, 100:266–272, 2000.

[54] C. E. Costa. *Monitoramento do Processo de Torneamento de Desbaste via Corrente Elétrica do Motor Principal da Máquina e via Vibração da Ferramenta*. Dissertação (mestrado em engenharia mecânica), Universidade Estadual de Campinas, Campinas, 1995.

[55] A. E. Diniz. *A Rugosidade Superficial da Peça em Processos de Torneamento, Critério de Fim de Vida da Ferramenta e Fatores de Influência*. Tese (doutorado em engenharia mecânica), Universidade Estadual de Campinas, Campinas, 1989.

[56] H. K. Tönshoff, B. Karpuschewski, e C. Regent. Process monitoring in grinding using micromagnetic techniques. *International Journal of Avanced Manufacturing Technology*, 15(10): 694–698, 1999.

[57] Y. M. Niu, Y. S. Wong, e G. S. Hong. An intelligent sensor system approach for reliable tool flank wear recognition. *The International Journal of Advanced Manufacturing Technology*, 14 (2):77–84, 1998.

[58] M. A. Mannan, A. A. Kassin, e M. Jing. Application of image and sound analysis techniques to monitor the condition of cutting tools. *Pattern recognition letters*, 21:969–979, 2000.

[59] H. M. Ertunc e K. A. Loparo. A decision fusion algorithm for tool wear condition monitoring in drilling. *International Journal of Machine Tools and Manufacture*, 41:1347–1362, 2001.

[60] D. Choi, W. T. Kwon, e C. N. Chu. Real-time monitoring of tool fracture in turning using sensor fusion. *International Journal of Advanced Manufacturing Technology*, 15:305–310, 1999.

[61] R. C. Gonzalez e R. E. Woods. *Digital Image Processing*. Addison-Wesley, 2 ed., 2002.

[62] B. Jähne. Introduction. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 1, pág. 1–4. Academic Press, 1999.

[63] W. J. Smith. *Modern Optical Engineering*. McGraw-Hill, 3 ed., 2000.

[64] P. Geissler. Imaging optics. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 64–102. Academic Press, 1999.

[65] H. Haussecker. Radiation. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 7–36. Academic Press, 1999.

[66] H. Haussecker. Interaction of radiation with matter. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 37–62. Academic Press, 1999.

[67] H. Haussecker. Radiometry of imaging. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 103–136. Academic Press, 1999.

[68] H. Haussecker. Illumination sources and techniques. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 63–102. Academic Press, 1999.

[69] P. Seitz. Solid-state image sensing. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 165–222. Academic Press, 1999.

[70] U. Seger, U. Apel, e B. Höfflinger. Hdrc-imagers for natural vision perception. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 223–236. Academic Press, 1999.

[71] B. Schneider, P. Rieve, e M. Böhm. Image sensors in tfa (thin film on asic) technology. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 237–270. Academic Press, 1999.

[72] S. Sedky e P. Fiorini. Poly sige bolometers. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 271–308. Academic Press, 1999.

[73] B. Jähne. Hyperspectral and color imaging. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 309–321. Academic Press, 1999.

[74] D. Uttenweiler e R. H. A. Fink. Dynamic fluorescence imaging. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 323–346. Academic Press, 1999.

[75] H. Stegmann, R. Wepf, e R. R. Schröder. Electron microscopic image acquisition. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 347–386. Academic Press, 1999.

[76] W. Albert e M. Pandir. Processing of ultrasound images in medical diagnosis. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 387–414. Academic Press, 1999.

[77] M. J. Buckingham. Acoustic daylight imaging in the ocean. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 415–424. Academic Press, 1999.

[78] R. Massen. The multisensorial camera for industrial vision applications. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 425–439. Academic Press, 1999.

[79] W. G. Schreiber e G. Brix. Magnetic resonance imaging in medicine. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 563–600. Academic Press, 1999.

[80] A. Haase, J. Ruff, e M. Rokitta. Nuclear magnetic resonance microscopy. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 601–612. Academic Press, 1999.

[81] B. Jähne. *Digitale Bildverarbeitung*. Springer, 4 ed., 1997.

[82] W. K. Pratt. *Digital Image Processing*. John Wiley and Sons, 3 ed., 2001.

[83] G. X. Ritter e J. N. Wilson. *Handbook of Computer Vision Algorithms in Image Algebra*. CRC Press, 1 ed., 1996.

[84] B. Jähne. Spatial and fourier domains. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 2, pág. 35–66. Academic Press, 1999.

[85] B. Jähne. Multiresolutional signal representation. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 2, pág. 67–90. Academic Press, 1999.

[86] B. Jähne. Neighborhood operators. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 2, pág. 92–124. Academic Press, 1999.

[87] B. Jähne, H. Scharr, e S. Körkel. Principles of filter design. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 2, pág. 125–152. Academic Press, 1999.

[88] B. Jähne. Local averaging. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 2, pág. 153–174. Academic Press, 1999.

[89] B. Jähne. Interpolation. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 2, pág. 175–192. Academic Press, 1999.

[90] T. Wagner. Texture analysis. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 2, pág. 275–308. Academic Press, 1999.

[91] P. Soille. Morphological operators. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 2, pág. 627–682. Academic Press, 1999.

[92] R. Jain, R. Kasturi, e B. Schunck. *Machine Vision*. McGraw-Hill, 1995.

[93] B. Jähne, H. Haussecker, e P. Geissler, editores. *Handbook of Computer Vision and Applications*, volume 2. Academic Press, 1 ed., 1999.

[94] D. Paulus, J. Hornegger, e H. Niemann. Software engineering for image processing and analysis. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 77–102. Academic Press, 1999.

[95] K. Matshushima, T. Kawabata, e T. Sata. Recognition and control of the morphology of tool failures. In *Annals of the CIRP*, volume 28, pág. 43–47, 1979.

[96] D. Cuppini, G. D'Errico, e G. Rutelli. Tool image processing with applications to unmanned metal-cutting: a computer vision system for wear sensing and failure detection. *SPIE*, 701: 416–422, 1986.

[97] Y. H. Lee, P. Bandyopadhyay, e B. D. Kaminski. Cutting tool wear measurement using computer vision. In *Proceedings of Sensor '86 Conference*, pág. 195–212, 1986.

[98] J. U. Jeon e S. W. Kim. Optical flank wear monitoring of cutting tools by image processing. *Wear*, 127(2):207–217, 1988.

[99] T. Teshima, T. Shibasaka, M. Takuma, e A. Yamamoto. Estimation of cutting tool life by processing tool image data with neural network. In *Annals of the CIRP*, volume 42, pág. 59–62, 1993.

[100] Y. Maeda, H. Uchida, e A. Yamamoto. Estimation of wear land width of cutting tool flank with the aid of digital image processing techniques. *Bulletin of the Japan Society of Precision Engineering*, 21(3):211–213, 1987.

[101] J. J. Park e A. G. Ulsoy. On-line flank wear estimation using an adaptive observer and computer vision. *Journal of Engineering for Industry*, 115:37–43, 1993.

[102] K. R. Castleman. *Digital Image Processing*. Prentice-Hall, 1996.

[103] T. Pfeifer e H. Thrum. Basis for the integration of sensors and actuators into autonomous production cells. In *Proceedings of the Joint Hungarian-British Mechanotrics Conference*, pág. 527–538, 1994.

[104] E. Eversheim, A. Haufe, W. Koelcheid, M. Walz, e N. Michalas. Integrative structure for flexible product design and manufacturing planning. *Human Factors and Ergnomics in Manufacturing*, 10(3):343–353, 2000.

[105] F. Klocke e A. Kopner. Zerspannprozesse flexibel planen und optimieren. *Werkstatttechnik*, 89 (10):461–464, 1999.

[106] C. Schlick, R. Daude, H. Luczak, M. Weck, e J. Springer. Head-mounted display for supervisory control in autonomous production cells. *Displays*, 17(3-4):199–206, 1997.

[107] H. Luczak, R. Reuth, e L. Schmidt. Development of error-compensating ui for autonomous production cells. *Ergonomics*, 46(1-3):19–40, January 2003.

[108] C. Schlick, R. Reuth, e H. Luczak. A comparative simulation study of work processes in autonomous production cells. *Human Factors and Ergonomics in Manufacturing*, 12(1):31–54, 2002.

[109] S. Kaierle, J. Beersiek, E. W. KReutz, R. Poprawe, J. Guennewig, e H. Rake. Online control of penetration depth in lase beam welding. In *Proceedings of the IECON - Industrial Electronics Conference*, volume 3, pág. 1694–1698, 1998.

[110] N. Brouër e M. Weck. Feature-oriented programming interface for an autonomous production cell. *Control Engineering Practice*, 6(11):1405–1410, 1998.

[111] S. Mann, S. Kaierle, E. W. Kreutz, e R. Poprawe. Autonomous production cell for welding with laser radiation. *Welding and Cutting*, 54(6):292–293, 2002.

[112] S. Kaierle, M. Dahmen, E. W. Kreutz, e R. Poprawe. Autonomous production cells in laser materials processing - a visionary approach. In Laser Institute of America, editor, *Proceedings*, volume 83, pág. E1–E10, 1997.

[113] L. Boske, S. Kaierle, S. Mann, J. Ortmann, e J. Willach. Autonomous production cell for micro- and nano-processing. In The International Society for Optical Engineering, editor, *Proceedings of SPIE*, volume 4977, pág. 16–27, 2003.

[114] N. Michalas. Management von autonomen produktionszellen - fleixibiltät und robustheit durch autonomie. *Tools*, 1, 2002.

[115] A. Orth. *Desenvolvimento e Implementação de uma Estrutura para Sensores e Atuadores Inteligentes em uma Célula Autônoma de Produção*. Projeto de fim de curso (engenharia de controle e automação industrial), Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2000.

[116] I. C. Carlsen e D. Haaks. Iks - concept and implementation of an object-oriented framework for image processing. *Computer and Graphics*, 15(4):473–481, 1991.

[117] U. Köthe. Reusable software in computer vision. In B. Jähne, H. Haussecker, e P. Geissler, editores, *Handbook of Computer Vision and Applications*, volume 3, pág. 103–132. Academic Press, 1999.

[118] A. Alexandrescu. *Modern C++ Design: Generic Programming and Design Patterns Applied*. Addison-Wesley, 1 ed., 2001.

[119] E. Gamma, R. Helm, R. Johnson, e J. Vlissides. *Padrões de Projeto: Soluções Reutilizáveis de Software Orientado a Objetos*. Bookman, 1 ed., 2000.

[120] F. Deschamps. *Otimização Automática de Parâmetros para um Sistema de Processamento de Imagens Aplicado ao Controle do Desgaste de Ferramentas de Corte em Células Autônomas de Produção*. Projeto de fim de curso (engenharia de controle e automação industrial), Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2002.

[121] F. Deschamps. *Projeto, Desenvolvimento e Implementação de uma Biblioteca Computacional de Redes Neurais Artificiais - S2iLibNeural*. Relatório de estágio curricular (engenharia de controle e automação industrial), Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2001.

[122] M. H. Yang, D. Kriegmann, e N. Ahuja. Detecting faces in images: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

[123] M. M. Minasi, F. Deschamps, e M. R. Stemmer. Eigenfaces - uma proposta para o reconhecimento do desgaste de ferramentas. In *Anais do II Simpósio Catarinense de Processamento Digital de Imagens*, volume 1, 2002.

[124] M. M. Minasi, A. Orth, e M. R. Stemmer. S2ilib: uma biblioteca orientada a objetos para processamento de sinais e sistemas de visão. In *Anais do I Simpósio Catarinense de Processamento Digital de Imagens*, volume 1, 2001.

[125] E. P. Cabral, F. Deschamps, e M. R. Stemmer. Algumas aplicações da transformada discreta de fourier no processamento digital de imagens. In *Anais do II Simpósio Catarinense de Processamento Digital de Imagens*, volume 1, 2002.

[126] F. L. Baldissera, A. Orth, e M. R. Stemmer. Análise da transformada wavelet aplicada ao processamento freqüencial de sinais. In *Anais do I Simpósio Catarinense de Processamento Digital de Imagens*, volume 1, 2001.

[127] F. L. Baldissera, F. Deschamps, e M. R. Stemmer. Análise da transformada wavelet aplicada ao processamento freqüencial de sinais. In *Anais do II Simpósio Catarinense de Processamento Digital de Imagens*, volume 1, 2002.

Índice Remissivo