INSTITUTO FEDERAL DE EDUCAÇÃO CIÊNCIA E TECNOLOGIA DE MINAS GERAIS - *CAMPUS* BETIM
BACHARELADO EM ENGENHARIA MECÂNICA

Cícero Friche Neto

DESENVOLVIMENTO DE *SOFTWARE* PARA ANÁLISE DE IMAGENS PARA COMPARATIVO DE FALHAS MECÂNICAS

BETIM
2026

CÍCERO FRICHE NETO

DESENVOLVIMENTO DE *SOFTWARE* PARA ANÁLISE DE IMAGENS PARA COMPARATIVO DE FALHAS MECÂNICAS

Trabalho de Conclusão de Curso apresentado à banca examinadora do curso de Engenharia Mecânica do Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais *Campus* Betim, como parte dos requisitos para obtenção do título de Bacharel em Engenharia Mecânica.

Orientador: Prof. Dr Gabriel Mendes de Almeida Carvalho

BETIM

2026

FICHA CATALOGRÁFICA

F897d Friche Neto, Cícero

Desenvolvimento de software para análise de imagens para comparativo de falhas mecânicas / Cícero Friche Neto. – 2026.

75 f. : il.

Trabalho de conclusão de curso (Bacharelado em Engenharia Mecânica) - Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais, Câmpus Betim, 2026.

Orientação: Prof. Dr. Gabriel Mendes de Almeida Carvalho

1. Análise de falhas. 2. Metalografia. 3. Análise de imagens. 4. Software - Desenvolvimento. 5. Engenharia Mecânica. I. Friche Neto, Cícero. II. Título.

CDU: 658.58

Ficha catalográfica elaborada pelo Bibliotecário Denísio Pereira Marcos CRB-6/3142

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS
Campus Betim
Diretoria de Ensino
Docentes Mecânica
Rua Itamarati - CEP 32677-564 - Betim - MG
3135976360 - www.ifmg.edu.br

ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO

Aos 03 dias do mês de fevereiro do ano de 2026, às 21:20, nas dependências do IFMG - *campus* Betim, reuniu-se a banca examinadora presidida por mim, Gabriel Mendes de Almeida Carvalho e demais membros, Fagner Guilherme Ferreira Coelho e Adriano Ribeiro Marinho. Nesta ocasião o discente Cícero Friche Neto do curso de Bacharelado em Engenharia Mecânica, com registro acadêmico de número 0034222 do IFMG – *Campus* Betim, defendeu seu Trabalho de Conclusão de Curso (TCC) intitulado "DESENVOLVIMENTO DE SOFTWARE PARA ANÁLISE DE IMAGENS PARA COMPARATIVO DE FALHAS MECÂNICAS" e foi APROVADO, com 80 pontos (oitenta) pontos.

Este resultado reflete o cumprimento parcial dos critérios de avaliação estabelecidos pelo curso e reconhece os esforços e a dedicação do discente e seu orientador no desenvolvimento do seu TCC. O lançamento da nota e o consequente encerramento do respectivo processo está condicionado ao cumprimento dos procedimentos pós-defesa conforme previstos nos regulamentos vigentes. Tais procedimentos pós-defesa devem ser finalizados dentro do prazo limite de 20 dias úteis, a contar da data desta ata. O descumprimento destes procedimentos até a data estipulada implicará em atribuição de nota 0 (zero) e consequente reprovação.

A sessão foi encerrada às 22h40. Para constar, eu, Gabriel Mendes de Almeida Carvalho, redigi a presente ata que após lida publicamente, foi aprovada e assinada pelo discente e membros da banca examinadora.

Assinam abaixo, com autenticidade, os membros da banca.

gov.br Documento assinado digitalmente
GABRIEL MENDES DE ALMEIDA CARVALHO
Data: 11/02/2026 21:34:21-0300
Verifique em https://validar.iti.gov.br

Gabriel Mendes de Almeida Carvalho

gov.br Documento assinado digitalmente
FAGNER GUILHERME FERREIRA COELHO
Data: 12/02/2026 11:09:17-0300
Verifique em https://validar.iti.gov.br

Fagner Guilherme Ferreira Coelho

gov.br Documento assinado digitalmente
ADRIANO RIBEIRO MARINHO
Data: 12/02/2026 13:40:36-0300
Verifique em https://validar.iti.gov.br

Adriano Ribeiro Marinho

Dedico este trabalho à minha mãe Luciana, meu pai Cícero, Minhas irmãs Amanda e Tamiris, meu sobrinho Benício e minha companheira de vida Kamilla. É por eles que tento ser uma pessoa melhor.

AGRADECIMENTOS

Tenho muitas pessoas a agradecer por toda a jornada que tive nos estudos até agora, citar todas seria impossível, mas gostaria deixar expresso um agradecimento especial a algumas delas.

A minha família, por todo seu apoio, incentivo e suporte com certeza o carinho e o amor deles é que motiva a querer buscar conhecimento para me tornar uma pessoa melhor em gratidão a tudo que fizeram e ainda fazem por mim. A minha companheira de vida Kamilla tenho mil motivos para agradecer em especial nesse período pela ajuda, pela motivação e por sempre estar comigo quando preciso, sua presença em minha vida é uma das minhas bases.

Ao meu professor orientador Gabriel Mendes de Almeida Carvalho, também expresso minha imensa gratidão por além de transmitir seu conhecimento faz isso com muita dedicação e o carinho de quem realmente ama o que faz. Aos demais professores do IFMG também sou imensamente grato

Agradeço também aos meus colegas de curso pelo aprendizado e ter deixado um legado de amizade em minha vida Brendow Pacheco, Fabricio Bicalho, Henrique Rosendo, Luís Henrique, Samuel Santos sempre serão muito considerados por mim, por terem tornado essa época da vida muito mais divertida e por tudo que me ajudaram.

Um agradecimento também para os dois cãezinhos da casa da nossa família Duque e Balu da maneira deles sempre me ensinaram muitas coisas e estarão eternamente em nossos corações.

RESUMO

Nas áreas de ciências e engenharia, analisar amostras por imagens tem se tornado parte muito importante pois possibilitam a compreensão das propriedades e composição dos materiais. Este trabalho apresenta a criação de um *software* para realizar análises em imagens baseadas em parâmetros sendo elas provenientes de análises metalográficas, ou de análise de falhas em cordões de solda. Esse desenvolvimento parte do entendimento dos conceitos da metalografia e sua importância, uma breve introdução aos conceitos de micrografia e microscopia, as principais etapas de um desenvolvimento de um *software* que será responsável por analisar as imagens por comparação de histogramas mostrando como os arquivos de imagem podem ser manipulados a partir de seu modelo matemático e como se extrair informações e gerar conclusões com relatórios atendendo a documentação.

Palavras-chave: Metalografia; Análise metalográfica; Análise de falhas; Análise de imagens; *Software*; Arquitetura de *Software*;

ABSTRACT

In the fields of science and engineering, image-based sample analysis has become increasingly important, as it enables a better understanding of material properties and composition. This work aims to present the development of software designed to perform image analyses based on defined parameters, applied to metallographic analyses or failure analysis in weld beads. The development is grounded in the understanding of metallography concepts and their importance, including a brief introduction to micrography and microscopy. It also addresses the main stages of software development responsible for analyzing images through histogram comparison, demonstrating how image files can be manipulated based on their mathematical models and how information can be extracted to generate conclusions and documented reports.

Keywords: Metallography; Metallographic analysis; Failure analysis; Image analysis; Software; Software architecture.

LISTA DE ILUSTRAÇÕES

Figura 1 – Exemplo de macrografia corpo de prova.....19
Figura 2 – Ilustração esquemática dos modos de iluminação em microscópio ótico para metalografia. (a) iluminação inclinada ou oblíqua, (b) iluminação paralela ao eixo ou normal e (c) iluminação em campo escuro. 20
Figura 3 – Exemplo de fragmentação de cores.....23
Figura 4 – Histograma formulado a partir da distribuição.....24
Figura 5 – Esquema explicando funcionamento do software 31
Figura 6 – Logo da linguagem Java e da empresa Oracle.....35
Figura 7 – Imagem do arquivo pom.xml.....36
Figura 8 – Diagrama de contexto 37
Figura 9 – Imagem com trinca enviada 60
Figura 10 – Tabela de preços OpenIa.....60

LISTA DE TABELAS

Tabela 1 – Valor da comparação dos histogramas. 32
Tabela 2 – Resultado da média das comparações 32
Tabela 3 – Explicação do conceito de classe e objeto 38
Tabela 4 – Definições dos princípios que compõem o SOLID. 39
Tabela 5 - Resultados comparação.....40
Tabela 6 – Valores médios calculados.....56
Tabela 7 – Resultado da análise do software.....57
Tabela 8 – Custo computacional 59

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
IFMG	Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais
NBR	Normas Técnicas Brasileiras
TCC	Trabalho de Conclusão de Curso
CCC	Cúbica de corpo Centrado
OpenCV	Open Source Computer Vision
CFD	*Calculation of the cumulative distribution function*
EM	Espectro Eletromagnético
I.A	Inteligência Artificial
API	*Application Programming Interface*
IDE	*Integrated Development Environment*

LISTA DE SÍMBOLOS

@	Arroba
©	Copyright

SUMÁRIO

1 INTRODUÇÃO 15
1.1 Justificativa 16
1.2 Objetivos 16
1.3 Organização do Trabalho 16
2 REVISÃO BIBLIOGRÁFICA 17
2.1 Análise metalográfica 17
2.2 Macrografia 18
2.3 Micrografia 19
3 FUNDAMENTAÇÃO TEÓRICA 21
3.1 Processamento de Imagens 21
3.1.1 Conceito de imagem Digital 21
3.1.2 Processamento de Imagens Digitais 22
3.2 Processamento de Histogramas 23
3.2.1 Equalização de histograma 24
3.2.2 Especificação de histograma 25
3.2.2.1 Processamento local de histograma 26
3.2.3 Utilizando estatísticas de histograma para o realce da imagem 27
3.3 *OpenCV* e Aplicações Computacionais na Metalografia 28
3.3.1 Visão Computacional aplicada à Engenharia 28
3.3.2 Ferramentas do OpenCV no Processamento de Imagens 28
3.3.3 Projeção de Fundo (*Back Projection*) 29
3.3.4 Template Matching 29
3.3.5 Detecção de Contornos 30
3.3.6 Envoltória Convexa (*Convex Hull*) 30
4 METODOLOGIA 30
4.1 Funcionamento do sistema 31
4.2 Integração com Inteligência artificial 33
4.3 Arquitetura 33
4.3.1 IntelliJ IDEA 33
4.3.2 Linguagem Java 34
4.3.3 Maven 35

4.3.4 Domain-Driven Design 36
4.3.5 Orientação a objetos 37
4.3.6 Princípios S.O.L.I.D 38
5 RESULTADOS E DISCUSSÕES 40
5.1 Teste Prático 40
5.2 Custo computacional 57
5.3 Integração com Inteligência Artificial 59
6 CONCLUSÃO 60
REFERÊNCIAS 62
APÊNDICE A – SOFTWARE DETALHADO 65

1 INTRODUÇÃO

As análises de um modo geral têm o objetivo de observar algo com mais atenção, como descrito, segundo Ernani Terra em (2011, p.54), “a. ná.li.se s.f. Decomposição de um todo em seus elementos.”

Partindo do princípio de que a análise metalográfica observa com mais atenção decompondo os elementos que constituem o material é, depara-se com o conceito do que são esses materiais e qual a ciência está por trás de descrever todos os conteúdos relacionados. Para melhor entendimento foi consultado o livro Ciência e engenharia de materiais: uma introdução / William D. Callister, Jr., David G. Rethwisch; tradução Sergio Murilo Stamile Soares - 8. ed. - Rio de Janeiro: LTC, 2013.

Como os materiais são compostos por ligações atômicas, podem apresentar diferentes formas e características dependendo de diversas formas de se produzi-los sendo essas variações estudadas e analisadas para que se determine como esses fatores podem influenciar diretamente, segundo Callister e Rethwisch em 2013,

> Muitas vezes, é conveniente subdividir a disciplina Ciência e Engenharia de Materiais nas subdisciplinas Ciência de Materiais e Engenharia de Materiais. Especificamente, a Ciência de Materiais envolve a investigação das relações entre as estruturas e as propriedades dos materiais. Em contraste, a Engenharia de Materiais, com base nas correlações estrutura-propriedade, projeta ou “engenhera” a estrutura de um material para obter nele um conjunto predeterminado de propriedades. (CALLISTER; RETHWISCH, 2013, p. 2).

Portanto a análise metalográfica agrega nessa parte onde é feita uma observação mais especifica de um material composto por vários fatores em sua produção, que resultam em microestruturas no caso dos metais, que por meio da ampliação nos fornece um cenário que torna possível sua avaliação, segundo Colpaert 1951,

> As estruturas que se observam em produtos de aços dependem diretamente de transformações de fases que ocorrem em seu processamento, desde a solidificação (em que a fase líquida se transforma, por exemplo, em fase sólida CCC) até as transformações no estado sólido realizadas em tratamentos termomecânicos. Estas transformações se passam, em geral, com a formação de vários núcleos da(s) nova(s) fase (s) que se forma (m) ao longo da massa de metal em transformação. Na maioria das vezes, as fases formadas são cristalinas (isto é, são compostas por átomos em arranjos regulares, repetitivos). É praticamente impossível que todos os núcleos formados ao longo da massa de metal que se transforma tenham exatamente a mesma orientação cristalográfica, de modo que, quando os cristais formados se encontram, há regiões onde os átomos não se ajustam exatamente à orientação de nenhum dos cristais em crescimento. (COLPAERT HUBERTUS, 1951, p. 26).

Este trabalho apresenta uma solução voltada para realizar análises de imagens geradas digitalmente, tanto a análise metalográfica conhecida e aplicadas amplamente na área de ciência dos materiais, quanto na análise de falhas em cordões de solda. Essas análises são focadas especificamente em situações que a amostra necessita de uma comparação, baseada em imagens definidas como parâmetro.

1.1 Justificativa

Análises comparativas de imagens podem ser bastante demoradas e trabalhosas, principalmente quando feitas em grande volume. Com o intuito realizar essa análise de comparação com agilidade e confiabilidade, foi desenvolvido um software que a partir de imagens de referência determinadas pelo usuário realiza essa comparação de forma rápida e objetiva.

1.2 Objetivos

O trabalho tem o objetivo de apresentar um *software* que realiza a análise de comparação de imagens, definindo imagens de referência chamadas de parâmetro e submetendo outras a comparação, destacando apenas imagens que possuem similaridade.

1.3 Organização do Trabalho

O trabalho apresenta o desenvolvimento e funcionamento de um *software* que atenda a algumas análises feitas na engenharia por meio de imagens. Inicialmente será apresentado quais as análises de imagem podem ser contempladas, como as ferramentas de análise de imagem funcionam explorando como os arquivos são convertidos em modelos matemáticos. Explicação de elementos de arquitetura e boas práticas de programação que guiaram o desenvolvimento do *software* e por fim como os resultados foram alcançados e analisados, com a elaboração de relatórios e análises de viabilidade de aplicação.

2 REVISÃO BIBLIOGRÁFICA

2.1 Análise metalográfica

A análise de metalografia é um processo fundamental para o estudo e compreensão das propriedades e comportamento dos metais. Ela desempenha um papel essencial em diversas áreas da ciência dos materiais, engenharia de materiais e metalurgia. Através dessa técnica, é possível examinar a estrutura microscópica dos metais e ligas metálicas, revelando informações valiosas sobre sua composição e propriedades mecânica, por meio da observação de sua microestrutura. Porém essa análise visual projetada em duas dimensões avalia uma estrutura tridimensional que necessita de diversas considerações, segundo Colpaert 1951,

> Um dos problemas mais interessantes da avaliação de micro e macroestruturas de metais é o fato de que, na maioria das vezes, as técnicas analíticas disponíveis permitem a observação de seções bidimensionais de estruturas que têm características tridimensionais. Esta transformação aparentemente simples requer cuidados especiais na aplicação da técnica metalográfica. Estes cuidados vão desde a seleção das seções a estudar até a avaliação criteriosa dos resultados obtidos na avaliação destas seções. Embora técnicas de reconstrução tridimensional já venham sendo aplicadas ao estudo da estrutura metalográfica, este tipo de análise ainda requer um investimento considerável de recursos materiais e tempo. As primeiras técnicas empregavam seccionamento e preparação de planos sucessivos de amostragem [1], mas já existem diversas outras técnicas que possibilitam recomposição tomográfica automática. (COLPAERT HUBERTUS, 1951, p. 26).

Uma das principais razões para realizar a análise de metalografia é a capacidade de identificar e caracterizar as fases presentes em um material. Essas fases podem ser determinantes para as propriedades do metal, como resistência, dureza, ductilidade e tenacidade. Ao examinar a microestrutura, é possível identificar a presença de diferentes fases, como ferrita, perlita, cementita, martensita, entre outras. Essa informação é essencial para entender o comportamento do metal sob diferentes condições de serviço, como temperatura, pressão e solicitações mecânicas, segundo William D. Callister, Jr. e David G. Rethwisch.

> O comportamento mecânico de um material reflete a relação entre sua resposta ou deformação a uma carga ou força aplicada. Propriedades mecânicas importantes para o projeto são rigidez, resistência, dureza, ductilidade e tenacidade.
> As propriedades mecânicas dos materiais são verificadas através da realização de experimentos de laboratório cuidadosamente planejados, que

> reproduzem da forma mais fiel possível as condições de serviço. (CALLISTER; RETHWISCH, 2013, p. 129).

Além disso, a análise de metalografia é crucial para avaliar a qualidade dos processos de fabricação e tratamentos térmicos utilizados na produção de componentes metálicos. Por exemplo, ao examinar a microestrutura de um material, é possível identificar a presença de inclusões, segregações, porosidade, trincas e outros defeitos que podem comprometer a integridade e o desempenho do metal. Essa análise é especialmente importante em setores como a indústria aeroespacial, automotiva e de petróleo e gás, onde a segurança e a confiabilidade dos materiais são primordiais.

> As propriedades de alguns materiais são significativamente influenciadas pela presença de imperfeições. Portanto, é importante possuir conhecimento dos tipos de imperfeições que existem e dos papéis que elas desempenham ao afetar o comportamento dos materiais. (CALLISTER; RETHWISCH, 2013, p. 78).

Dentre as metodologias existentes dentro da metalografia existem duas principais e se diferem pela execução e resultados que trazem, são denominadas de macrografia e micrografia.

2.2 Macrografia

A macrografia tem como objetivo principal examinar uma amostra a partir de uma seção plana e previamente preparada como mostrado na figura 1, na qual será feita a avaliação da estrutura a vista desarmada, isto é, com a utilização de no máximo uma lupa com a ampliação de no máximo 10 vezes, segundo Colpaert 1951.

> A macrografia consiste no exame do aspecto de uma peça ou amostra metálica, segundo uma seção plana devidamente polida e, em geral, atacada por um reativo apropriado. O aspecto, assim obtido, chama-se macroestrutura. O exame é feito à vista desarmada ou com auxílio de uma lupa. A palavra macrografia é também empregada para designar os documentos que reproduzem a macroestrutura, em tamanho natural ou com ampliação máxima de 10 vezes. Para ampliações maiores, emprega-se o termo micrografia, porque são, em geral, obtidas com o microscópio. (COLPAERT HUBERTUS, 1951, p. 38).

Figura 1 – Exemplo de macrografia corpo de prova

Fonte: Infosolda, 2018

Para a elaboração desta análise existem os protocolos de escolha da seção transversal que será analisado, sua preparação com polimento e armazenamento dos seus resultados, segundo Colpaert 1951,

> De um modo geral, o exame macrográfico de uma peça fundida de aço visa a determinação de rechupes e outras falhas, tais como bolhas, porosidades e trincas, caracterização das dimensões e disposição da estrutura dendrítica e, às vezes, avaliação da segregação.
> Dos ferros fundidos, só os mesclados e coquilhados normalmente apresentam interesse sob o ponto de vista macrográfico; nos primeiros pode-se apreciar as dimensões dos pequenos núcleos de ferro fundido cinzento, sua quantidade e modo de distribuição; nos segundos, a profundidade da parte coquilhada e a maneira como se processa a transição para o resto do material. A posição do corte nesses casos depende.
> naturalmente, da peça e da característica que se descia avaliar. (COLPAERT HUBERTUS, 1951, p. 40).

2.3 Micrografia

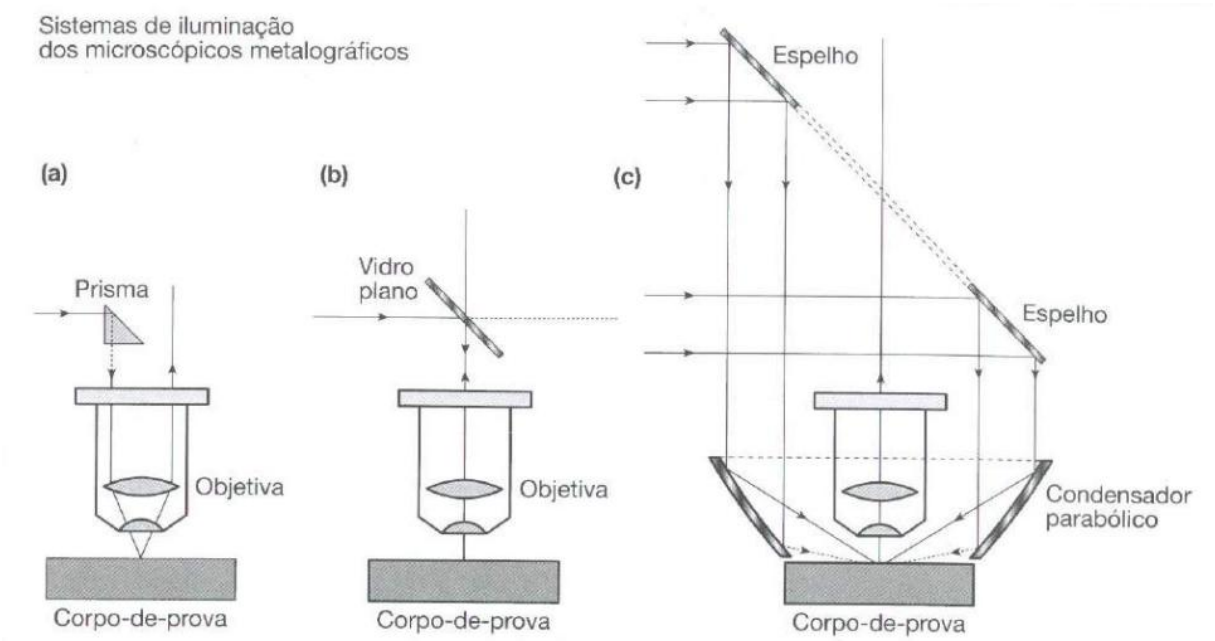
A metodologia que consiste na análise microscópicas de amostras metálicas é denominada de micrografia e possui um método de preparação para análise da amostra parecida com as demais formas de análise, escolhendo a seção transversal

e preparando-a polindo e disponibilizando de aparatos para a realização da análise como acoplar em resinas dando forma ao corpo de prova, segundo Colpaert 1951.

> Há diversas técnicas usuais para observar a estrutura dos aços e ferros fundidos em escala microscópica. Para um grande grupo de técnicas em que se observa a microestrutura através de seções, as técnicas de preparação de amostra são muito semelhantes. (COLPAERT HUBERTUS, 1951, p. 68).

Dentre as técnicas mais comuns utilizadas está a microscopia ótica, essa técnica consiste em incidir luz visível sobre a amostra e interpretar o reflexo que incide sobre a amostra, a resolução da imagem projetada é associada ao comprimento da onda da radiação empregada, como mostra a figura 2.

Figura 2 – Ilustração esquemática dos modos de iluminação em microscópio ótico para metalografia. (a) iluminação inclinada ou oblíqua, (b) iluminação paralela ao eixo ou normal e (c) iluminação em campo escuro.



Fonte: COLPAERT HUBERTUS, 1951, p. 69

Outro parâmetro associado ao comprimento de onda é o foco que depende do comprimento da onda empregada e da distância focal entre as lentes. Por causa dessas características a planicidade da amostra tem extrema importância por isso o cuidado em sua preparação visto que vão ser necessário a precisão dos parâmetros para coletar seu reflexo transmitido ao microscópio.

> Dentre as diversas técnicas de observação da microestrutura dos aços e ferros fundidos, a mais comum é a microscopia ótica. Neste caso, emprega-se luz visível que incide sobre a amostra e é refletida até o observador. A resolução que pode ser obtida em uma imagem depende do comprimento de onda da radiação empregada. Para a luz visível de cor verde, isto resulta em uma resolução de 220 a 250 nm que corresponde a um aumento máximo da ordem de 1400 vezes (1400X). Embora existam microscópios óticos capazes de fornecer aumentos superiores a este valor, tais aumentos são chamados aumentos "vazios" por não fornecerem informação adicional àquela obtida com o aumento máximo de cerca de 1400X.
> Por outro lado, a profundidade de foco também depende do comprimento de onda da radiação empregada na observação e da distância focal das lentes empregadas. (COLPAERT HUBERTUS, 1951, p. 68).

3 FUNDAMENTAÇÃO TEÓRICA

3.1 Processamento de Imagens

3.1.1 Conceito de imagem Digital

As imagens têm um papel muito relevante nos tempos atuais visto que seu registro serve como comprovação e ampara em diversas análises. A obtenção das imagens pode ser de diversas formas e proporciona inúmeras possibilidades de tratá-las, de uma certa forma, essa dinâmica revolucionou o mundo da ciência por adicionar uma forma de registrar os diversos fenômenos no mundo e comparar como descrito por Joly em 1994,

> As imagens e o seu potencial desenvolvem-se em todos os domínios científicos: da astronomia à medicina, das matemáticas à meteorologia, da geodinâmica à física e à astrofísica, da informática à biologia, da mecânica ao nuclear etc.
> Nestes diferentes domínios, as imagens são simplesmente visualizações de fenômenos. O que as distingue fundamentalmente uma das outras (se deixarmos de lado, bem entendido, as tecnologias mais ou menos avançadas que elas utilizam) é que são ou imagens verdadeiras ou reais, ou seja, permitem uma observação mais ou menos direta e mais ou menos sofisticada da realidade e das simulações numéricas.
> As imagens que ajudam a observar e a interpretar os diferentes fenômenos são produzidas a partir do registro de fenômenos físicos: o registro dos raios luminosos, na origem da fotografia, permite, por exemplo, que os satélites vigiem, por teledetecção, o avanço da - 25 - desertificação no planeta ou acompanhar e prever fenômenos meteorológicos; permite que as sondas astronômicas filmem os planetas mais distantes, assim como permite que as microcâmeras filmem o interior do corpo humano. (MARTINE, 1994, p. 24,25).

As imagens por si só já trouxeram uma enorme evolução na ciência, a utilização dos recursos digitais aplicados nessas imagens possibilita ainda mais aplicações. A imagem digital pode ser definida como uma função bidimensional, f (x, y) sendo x e y coordenadas dentro de um plano, e f a amplitude da intensidade em que esse ponto transmite as cores, por ser dentro de um espaço limitado esses pontos são chamados popularmente de pixels, essa quantidade de pares dentro do espaço pode variar o que determina a definição da imagem, segundo Gonzalez, R. C. e Woods, R. C. 2010.

> Uma imagem pode ser definida como uma função bidimensional, f (x, y), em que x e y são coordenadas espaciais (plano), e a amplitude de fem qualquer par de coordenadas (x, y) é chamada de intensidade ou nível de cinza da imagem nesse ponto. Quando x, y e os valores de intensidade de f são quantidades finitas e discretas, chamamos de imagem digital. O campo do processamento digital de imagens se refere ao processamento de imagens digitais por um computador digital. Observe que uma imagem digital é composta de um número finito de elementos, cada um com localização e valor específicos. Esses elementos são chamados de elementos pictóricos, elementos de imagem, pels e pixels. Pixel é o termo mais utilizado para representar os elementos de uma imagem digital. (GONZALEZ; WOODS; MARTINE, 2010, p. 1).

3.1.2 Processamento de Imagens Digitais

A visão humana embora seja um sentido muito poderoso apresenta limitações, o que as imagens digitais possuem de vantagem seria a possibilidade de ir além do que é denominado por Gonzalez, R. C. e Woods, R. C.Joly Martine como espectro eletromagnético (EM).

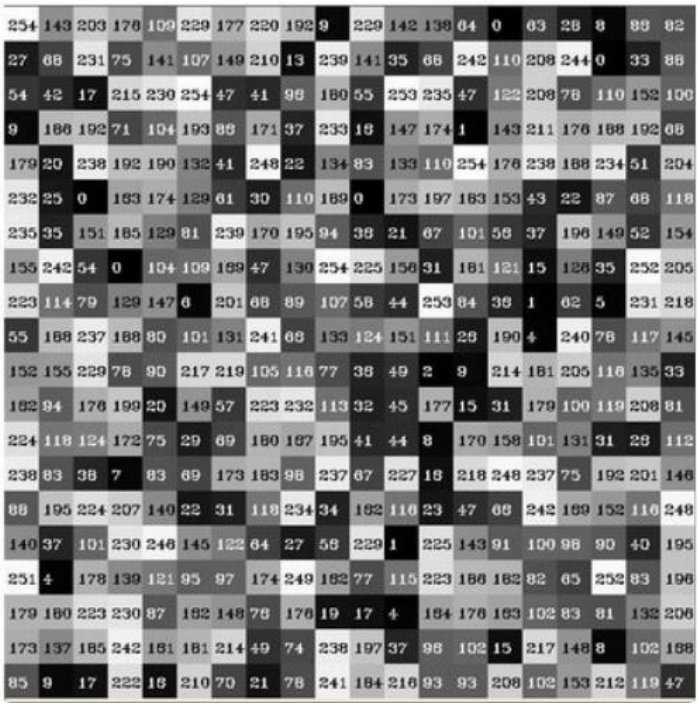
> A visão é o mais avançado dos nossos sentidos, de forma que não é de surpreender que as imagens exerçam o papel mais importante na percepção humana. No entanto, diferentemente dos seres humanos, que são limitados à banda visual do espectro eletromagnético (EM), os aparelhos de processamento de imagens cobrem quase todo o espectro EM, variando de ondas gama a ondas de rádio. (GONZALEZ; WOODS; MARTINE, 2010, p. 1).

Com a possibilidade de aplicar ferramentas, o processamento de imagem pode lidar com imagens geradas por diversos meios como ultrassom, microscopia eletrônica e imagens geradas por computador, podendo ampliar o processamento de imagens já que não há as mesmas limitações da visão humana. Com metodologias de análise essas imagens podem proporcionar ser analisadas, comparadas ou tratadas desde que convertidas para o formato digital.

3.2 Processamento de Histogramas

Histogramas são gráficos utilizados para análises de dados, sendo uma excelente ferramenta para analisar, segmentar, realçar dados desde que sua definição seja clara de quais dados se tratam os números indicados por eles. Na parte de análise de imagens os histogramas são formados pela quantidade de pixels e suas respectivas tonalidades, como evidencia a figura 3, as cores são classificadas de acordo com a informação salva digitalmente, essa informação é o que evidencia a tonalidade da cor que deverá ser exibida no display.

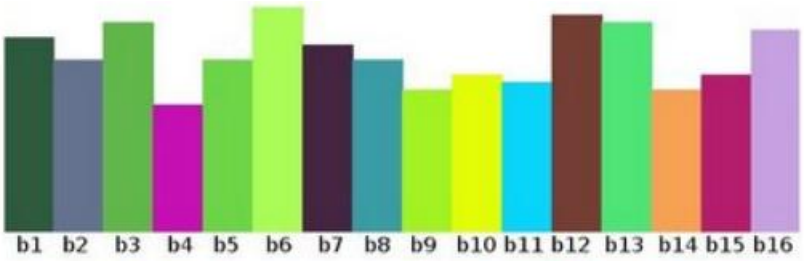
Figura 3 – Exemplo de fragmentação de cores
em uma imagem digital



Fonte: OpenCv, 2025

Organizando as cores da imagem e segregando, é possível formular um histograma da imagem, como apresentado na figura 4, e posteriormente tratando esses dados, eles podem ser analisados usando lógica.

Figura 4 – Histograma formulado a partir da distribuição de cores de uma imagem digital



Fonte: OpenCv, 2026

Com a definição dos histogramas mais clara percebe-se melhor a forma de como é feita a análise digital, que se baseia em uma metodologia denominada comparação de histogramas, para entender como a aplicação dessa lógica foi aplicada, é necessário entender alguns conceitos, sendo eles a Equalização de histogramas, Especificação de Histograma, Processamento local de histograma e a Utilização da estatística de histograma para realce da imagem

3.2.1 Equalização de histograma

A equalização de histograma é uma técnica clássica de processamento de imagens utilizada para melhorar o contraste de imagens digitais. Trata-se de um método de redistribuição dos níveis de intensidade de uma imagem, com o objetivo de expandir a faixa dinâmica e, assim, evidenciar detalhes que poderiam estar ocultos em regiões muito claras ou muito escuras.

No contexto da análise metalográfica, essa técnica é especialmente útil para realçar microestruturas metálicas cujos contrastes são, por vezes, pouco perceptíveis devido às limitações na captura da imagem por microscópios ópticos ou eletrônicos. A aplicação da equalização permite uma visualização mais nítida das fronteiras de grãos, inclusões ou outras imperfeições, facilitando sua identificação automática por sistemas computacionais.

Já no contexto de análise de falha em amostras de cordão de solda, auxiliam no equilíbrio focado no contorno de regiões com fissuras, ou trincas, que apresentam uma progressão de tonalidade que pode ser sútil ou mais abrupta, dependendo do tamanho da interrupção apresentada no material.

O algoritmo de equalização de histograma parte da construção de um histograma da imagem original, representando a frequência de ocorrência de cada nível de cinza. Em seguida, calcula-se a função de distribuição acumulada (CDF, *Cumulative Distribution Function*) e utiliza-se essa função como um mapeamento para redistribuir os níveis de intensidade de forma mais uniforme. Como resultado, os valores de pixels são ajustados para ocupar toda a faixa disponível (geralmente de 0 a 255 em imagens de 8 bits), aumentando a separabilidade visual entre regiões distintas.

Na implementação do *software* proposto, a equalização é realizada por meio da biblioteca *OpenCV*, que oferece a função `equalizeHist()` para esse fim. O uso dessa função permite aplicar a técnica de forma eficiente e padronizada, integrando-se perfeitamente às demais etapas de pré-processamento das imagens analisadas.

De acordo com a documentação técnica do *OpenCV* (2025), a equalização é recomendada como etapa preliminar em *pipelines* que envolvem segmentação, detecção de bordas e reconhecimento de padrões, pois contribui significativamente para a uniformização das imagens de entrada, reduzindo o viés causado por iluminação desigual ou má calibração dos sensores ópticos.

3.2.2 Especificação de histograma

A especificação de histograma, também conhecida como *matching* de histograma, é uma técnica de processamento digital de imagens utilizada para transformar a distribuição de intensidade de uma imagem de forma que ela se aproxime do histograma de uma imagem de referência. Diferentemente da equalização, que busca redistribuir os níveis de cinza de maneira uniforme, a especificação é orientada por um padrão desejado previamente definido.

No âmbito da análise de imagens metalográficas, essa técnica é particularmente útil quando se deseja comparar amostras diferentes sob condições similares de contraste e luminosidade. Ao padronizar os histogramas das imagens analisadas em relação a um modelo de referência — por exemplo, uma micrografia previamente validada — é possível reduzir variações causadas por iluminação ou

configurações do microscópio, mantendo o foco na análise estrutural propriamente dita.

O processo de especificação envolve três etapas principais: (1) cálculo da função de distribuição acumulada (CDF) da imagem original e da imagem de referência; (2) mapeamento dos níveis de intensidade da imagem original para novos valores com base na correspondência entre as duas CDFs; e (3) reconstrução da imagem com os novos níveis de intensidade.

A *OpenCV*, embora não ofereça uma função direta para essa operação, permite sua implementação por meio de rotinas baseadas em funções sendo elas `cv::calcHist()` que calcula o histograma de uma ou mais matrizes, sendo essas matrizes as distribuição de cores das imagens, também é utilizado a função `cv::equalizeHist()` que tem o objetivo de equalizar a distribuição matemática do histograma com o objetivo de normalizar o brilho e contraste da imagem e por fim operações de mapeamento personalizadas.

Existem também bibliotecas auxiliares ou *scripts* personalizados em Python e C++ que integram essa técnica como parte de fluxos automatizados de análise.

A aplicação dessa técnica no *software* proposto contribui para a criação de um ambiente de análise mais padronizado e robusto, principalmente quando se trabalha com bancos de imagens obtidas sob condições laboratoriais distintas.

3.2.2.1 Processamento local de histograma

O processamento local de histograma é uma técnica utilizada para melhorar o contraste de uma imagem considerando pequenas regiões da imagem ao invés de aplicar transformações globais. Ao contrário da equalização de histograma tradicional — que ajusta todos os *pixels* com base no histograma da imagem inteira —, o processamento local trabalha com janelas deslizantes (*tiles*) que percorrem a imagem, aplicando equalizações específicas em cada subárea. Essa abordagem é particularmente útil em imagens que apresentam variações locais significativas de brilho ou contraste.

Em metalografia, onde é comum encontrar microestruturas heterogêneas, variações de iluminação ou regiões parcialmente polidas, o processamento local de histograma permite destacar detalhes que poderiam ser ocultados por um ajuste

global. Por exemplo, grãos metálicos em regiões mais escuras podem ser realçados independentemente de áreas que já estão suficientemente iluminadas, permitindo uma análise mais homogênea da amostra como um todo.

Uma das implementações mais conhecidas dessa técnica é o algoritmo *CLAHE* (*Contrast Limited Adaptive Histogram Equalization*), disponível na biblioteca *OpenCV*. O *CLAHE* funciona dividindo a imagem em blocos menores, aplicando a equalização de histograma em cada bloco e, posteriormente, combinando-os por meio de interpolação para evitar descontinuidades abruptas entre as regiões.

O parâmetro de *clip limit* no *CLAHE* atua como um regulador da amplificação do contraste: valores mais baixos evitam realces exagerados e preservam uma aparência natural. Essa característica é especialmente valiosa na análise de imagens microscópicas, onde artefatos visuais podem comprometer a interpretação dos dados.

No *software* proposto, o uso do *CLAHE* foi considerado uma etapa opcional e ajustável pelo usuário, permitindo personalizar a intensidade do realce conforme as características de cada imagem processada.

3.2.3 Utilizando estatísticas de histograma para o realce da imagem

O histograma de uma imagem digital é uma ferramenta poderosa não apenas para visualização da distribuição de intensidades, mas também como fonte de informações estatísticas capazes de orientar transformações mais sofisticadas de realce. Entre os parâmetros estatísticos comumente utilizados estão a média, o desvio padrão, a moda e os percentis, que permitem realizar ajustes adaptativos em imagens com características distintas.

Na análise de imagens metalográficas, o uso dessas estatísticas é crucial para lidar com amostras com iluminação irregular, contrastes sutis entre fases e texturas complexas. Por exemplo, o cálculo da média e do desvio padrão dos níveis de cinza pode ser empregado para normalizar a imagem em relação a uma faixa de intensidade desejada, aumentando a nitidez de fronteiras de grãos metálicos ou evidenciando imperfeições como porosidades e inclusões.

Outro recurso relevante é a manipulação de percentis, que possibilita a exclusão de extremos muito claros ou escuros — geralmente associados a ruídos ou artefatos —, concentrando o realce na faixa central da distribuição. Essa abordagem, conhecida como *stretching* baseado em percentis (*percentile-based contrast*

stretching), evita saturação e garante um resultado mais fiel à realidade microscópica observada.

Além disso, tais estatísticas podem ser combinadas a regras de decisão automatizadas no *software* desenvolvido, permitindo, por exemplo, aplicar diferentes níveis de contraste conforme a variabilidade estatística local de uma região da imagem. Isso amplia as possibilidades de análise adaptativa, ajustando o processamento conforme as características particulares de cada imagem.

Na biblioteca *OpenCV*, essas estatísticas podem ser obtidas utilizando funções como `cv::meanStdDev()`, `cv::calcHist()` e `cv::minMaxLoc()`, sendo facilmente integradas a pipelines de realce adaptativo.

3.3 *OpenCV* e Aplicações Computacionais na Metalografia

3.3.1 Visão Computacional aplicada à Engenharia

A visão computacional é uma subárea da inteligência artificial que busca simular a capacidade humana de interpretar imagens e vídeos, por meio do uso de algoritmos e modelos matemáticos. Na engenharia, essa tecnologia tem se mostrado uma aliada estratégica na análise automatizada de processos industriais, monitoramento de qualidade e diagnóstico de falhas.

Na metalografia, a visão computacional permite a análise automatizada de microestruturas metálicas, promovendo maior precisão, repetibilidade e agilidade na interpretação de imagens obtidas por microscopia. Por meio da combinação de pré-processamento de imagem, extração de características e reconhecimento de padrões, é possível realizar tarefas antes manuais, como a identificação de grãos, porosidades, trincas e fases metálicas.

3.3.2 Ferramentas do OpenCV no Processamento de Imagens

A biblioteca *OpenCV* (*Open Source Computer Vision*) é uma das mais utilizadas no desenvolvimento de soluções em visão computacional. Por ser multiplataforma e de código aberto, possui ampla documentação e integração com linguagens como C++, Python e Java — esta última empregada no *software* desenvolvido neste trabalho.

Entre as ferramentas relevantes para aplicações em metalografia estão:

- `equalizeHist()` – para realce de contraste;
- `calcHist()` – para cálculo de histogramas;
- `matchTemplate()` – para localização de padrões;
- `findContours()` – para detecção de limites de estruturas;
- `convexHull()` – para fechamento de regiões irregulares.

Essas funções possibilitam montar um *pipeline* automatizado de análise, oferecendo ao usuário recursos prontos para extração de informações relevantes das imagens metalográficas.

3.3.3 Projeção de Fundo (*Back Projection*)

A projeção de fundo é uma técnica de segmentação baseada em histogramas de cor. Ela permite identificar em uma imagem áreas que têm maior probabilidade de conter determinado padrão de cor, previamente aprendido a partir de uma imagem de referência.

No contexto metalográfico, essa técnica pode ser utilizada para realçar regiões com determinada tonalidade característica de uma fase metálica ou de uma inclusão. O algoritmo gera uma imagem em escala de cinza onde os *pixels* com maior probabilidade de pertencer à classe buscada têm valores mais altos, facilitando etapas posteriores como limiarização e segmentação.

3.3.4 Template Matching

O *template matching* é uma técnica de detecção baseada na busca por uma imagem-modelo (template) dentro de uma imagem maior. A função *matchTemplate*() do *OpenCV* compara o *template* com diferentes regiões da imagem de entrada e retorna uma matriz de correspondência com os níveis de similaridade.

Na análise metalográfica, o método pode ser empregado para localizar padrões repetitivos como poros esféricos, inclusões ou defeitos com morfologia conhecida. É uma abordagem útil para comparação com imagens-padrão previamente catalogadas.

3.3.5 Detecção de Contornos

A detecção de contornos é uma técnica clássica de segmentação de bordas, amplamente utilizada na análise de imagens estruturais. A função `findContours()` do *OpenCV* permite identificar os limites de formas distintas em uma imagem binária, retornando os pontos que formam as bordas detectadas.

No campo da metalografia, essa função pode ser utilizada para delimitar grãos metálicos, inclusões ou defeitos, permitindo medições automatizadas de área, perímetro e forma. Com isso, o processo de análise se torna mais objetivo e mensurável.

3.3.6 Envoltória Convexa (*Convex Hull*)

A envoltória convexa, implementada no *OpenCV* por meio da função convexHull(), é uma técnica utilizada para "fechar" uma forma ao redor de um conjunto de pontos, formando o menor polígono convexo que os contém. Na prática, funciona como um envelope que cobre completamente a área externa de uma forma.

Essa função é especialmente útil na metalografia para calcular a regularidade de grãos ou inclusões, comparando a área real com a área de sua envoltória convexa. Tal análise fornece informações relevantes sobre a morfologia das estruturas analisadas.

4 METODOLOGIA

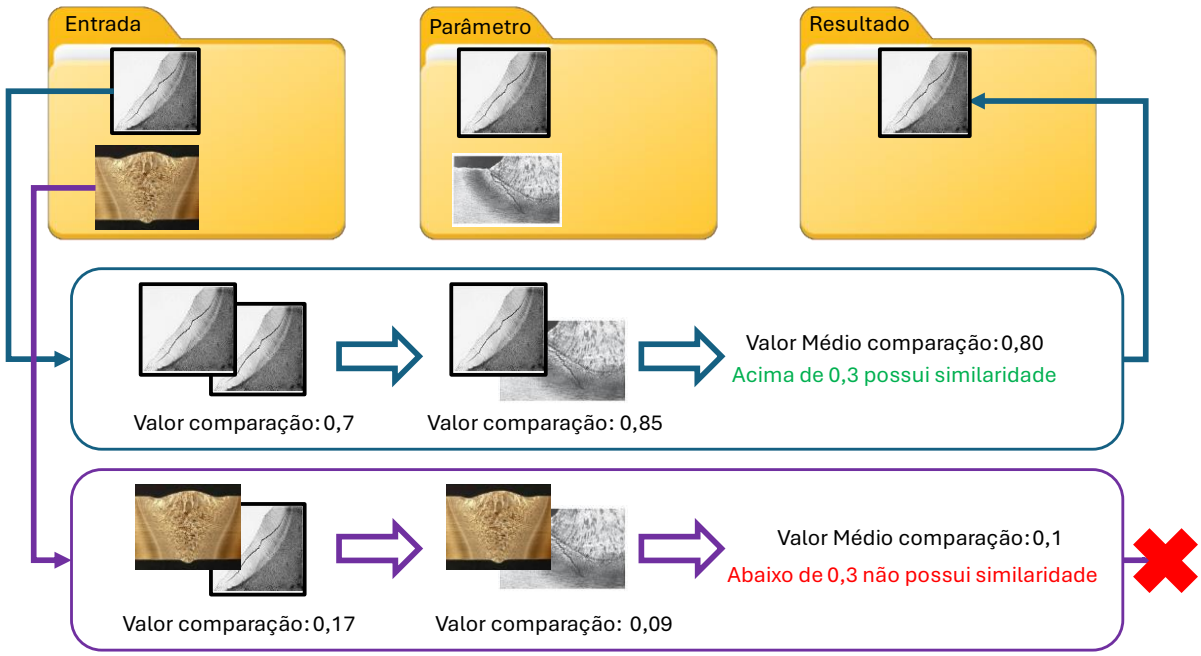
A metodologia apresentada a seguir mostra como o *software* desenvolvido executa as análises de comparação buscando as imagens, comparando-as e exibindo o resultado da análise, posteriormente foi descrito com mais detalhes quais as ferramentas, tecnologias e conceitos foram utilizados para elaborar o sistema que se encontra no Apêndice A, a partir da página 66.

4.1 Funcionamento do sistema

O *software* funciona comparando imagens, para acessar essas imagens é necessário ter um local de referência podendo ser um repositório local ou em nuvem, no caso deste desenvolvimento esse repositório foi criado localmente três pastas utilizadas para alocação das imagens, elas foram denominadas “entrada”, “parâmetro” e “resultado”.

Essas pastas funcionam como uma identificação para que o usuário do *software* possa definir as imagens que serão parâmetro de comparação, inserindo-as na pasta parâmetro. A seguir deve-se inserir na pasta entrada as imagens que serão analisadas e submetidas a comparação. Como apresentado na figura 5, após a análise as imagens que forem consideradas semelhantes pelo sistema serão alocadas na pasta resultado, dessa forma a pasta de resultado conterá somente as imagens contidas na pasta entrada que apresentam semelhança com as imagens contidas na pasta parâmetro, as imagens que não tiverem semelhança permanecerão na pasta entrada.

Figura 5 – Esquema explicando funcionamento do software



Fonte: Elaborado pelo autor, 2026

Cada imagem que está na pasta de entrada será comparada com todas as imagens contidas na pasta parâmetros, portanto o sistema elabora um levantamento das imagens listadas em todas as pastas e a partir daí executa um *looping* que vai manipular esses arquivos de imagem e submetê-las as comparações por meio de funções contidas em uma biblioteca própria para análise de imagens, chamada OpenCV.

A cada interação do *looping* é aplicada a sequência lógica de comparação dos valores do histograma das imagens, essa lógica foi referenciada no apêndice A página 67. Para resultado dessa comparação obtém-se um valor numérico que vai de -1 a 1, sendo esses valores negativos a diferença de contraste. Quanto maior a distância do 0 (zero) maior a similaridade das imagens. Esses valores são registrados em uma lista, como demonstrado na tabela 1, em cada linha é armazenado a imagem contida na pasta entrada na coluna “input”, a imagem comparada na pasta de parâmetro na coluna “parameter”, e o valor resultante na coluna “comparison”.

Tabela 1 – Valor da comparação dos histogramas.

input	***parameter***	***comparison***
imagen1.jpg	imagen1.jpg	1
imagen1.jpg	imagen10.jpg	0,98
imagen1.jpg	imagen11.jpg	0,98

Fonte: Elaborado pelo autor, 2026

Em posse da tabela 1 com todas as interações do looping é calculando a média geral montando uma outra lista, como mostrado na tabela 2, com essa relação entre as imagens inseridas e seus respectivos valores médios.

Tabela 2 – Resultado da média das comparações

imagem	**valor**
imagen1.jpg	70,44
imagen10.png	17,31

Fonte: Elaborado pelo autor, 2026

Com base na tabela 2 apresentada, uma outra classe, nomeada “calculateMediaActivity” presente no apêndice A página 70 segrega as imagens que apresentam apresentarem o valor médio do resultado das comparações maior que o número definido como aceitável, esse número pode ser parametrizável de acordo com a necessidade do usuário. Após as imagens listadas são movidas para a pasta resultado.

Como última etapa do sistema é exportado para uma classe que converte em csv as duas listas geradas no decorrer da análise e salva na raiz do projeto com os nomes “output.csv” e “outputMedia.csv” gerando assim relatórios para facilitar a análise detalhada de como ocorreu o processo de comparação.

4.2 Integração com Inteligência artificial

No projeto foi feita uma implementação enviando uma imagem de parâmetro e solicitando uma análise a I.A. A implementação foi feita por uma API (*Application Programming Interface*) chamando diretamente o serviço web oferecido pela empresa OpenIa, como mostrado no apêndice A página 74 pela classe “CharGptImageClient”. A implementação da função *sendImage*, configura e efetua a chamada a api externa selecionando modelo recuperando a chave e o script juntamente com a imagem a ser analisada como mostra no apêndice A página 73 pela classe “main” integração com I.A.

4.3 Arquitetura

4.3.1 IntelliJ IDEA

Para que seja possível desenvolver o *software* é necessário um ambiente denominado de IDE (Integrated Development Environment), no desenvolvimento em questão foi utilizado o programa IntelliJ IDEA é um produto da empresa JetBrains, focada em sistemas feitos na linguagem Java e Kotlin. Possui a versão chamada *Community* que oferta gratuitamente os recursos para que possibilite o desenvolvimento de *software*. Possui facilidades em integrar com a linguagem Java e no auxílio da análise do sistema.

4.3.2 Linguagem Java

A linguagem Java surgiu durante a década de 90 por um pequeno grupo de desenvolvedores que trabalhavam na empresa Sun Microsystems, ela recebeu esse nome por um fato curioso, segundo Claro D. B. e Sobral J. M. B. 1994,

> Como a equipe de desenvolvimento ingeria muito café enquanto estavam trabalhando, várias xícaras de café foram inseridas até que o projeto estivesse pronto. Finalmente em maio de 1995, a Sun anunciou um ambiente denominado Java (homenagem às xicaras de café). (CLARO; SOBRAL, 2008, p. 12).

Sua invenção é atribuída a James Gosling, e desde seu lançamento em 1996 tem atualizações constantes e está disponível para download no site da empresa Oracle que comprou a linguagem da Sun Microsystems em 2008 por 7 Bilhões de dólares. Seu objetivo inicial era suportar sistemas simples utilizados em eletrônicos, porém com o avanço da tecnologia acabou servindo como base na integração da novidade denominada WWW (World Wilde Web) e ganhou o mercado. É descrita como uma linguagem de aprendizado difícil e possui características interessantes como fácil escalabilidade e por ser amplamente utilizada possui bibliotecas com desenvolvimento amadurecido, e grande material para estudo, segundo Joshua Bloch 2018,

> A linguagem de programação Java é uma linguagem madura, amplamente utilizada, com um sistema de tipos poderoso, coleta de lixo e uma biblioteca padrão rica. (*Joshua Bloch, Effective Java, 3ª edição, Addison-Wesley, 2018, p. 16 (Prefácio)*.

Java é uma linguagem que possui muitas questões que a compõem ou que estão envolvidas a ela de alguma forma, descrevendo resumidamente, ela possui como característica ser orientada por objetos, possui portabilidade e Independência de plataforma e um autogerenciamento de memória. A orientação a objetos trata tudo na linguagem como um objeto, isso possibilita utilizar os conceitos de herança, polimorfismo, encapsulamento entre outros, já a portabilidade e independência significam que a linguagem foi projetada com o conceito de escreva uma vez e execute em qualquer lugar desde que tenha uma máquina virtual java (JVM), essa máquina

dispensa a necessidade de recompilação interpretando os códigos escritos na linguagem, pôr fim a sua capacidade de autogerenciamento de memória significa que ela é capaz de retirar da memória os objetos que não estão sendo utilizados.

Figura 6 – Logo da linguagem Java e da empresa Oracle

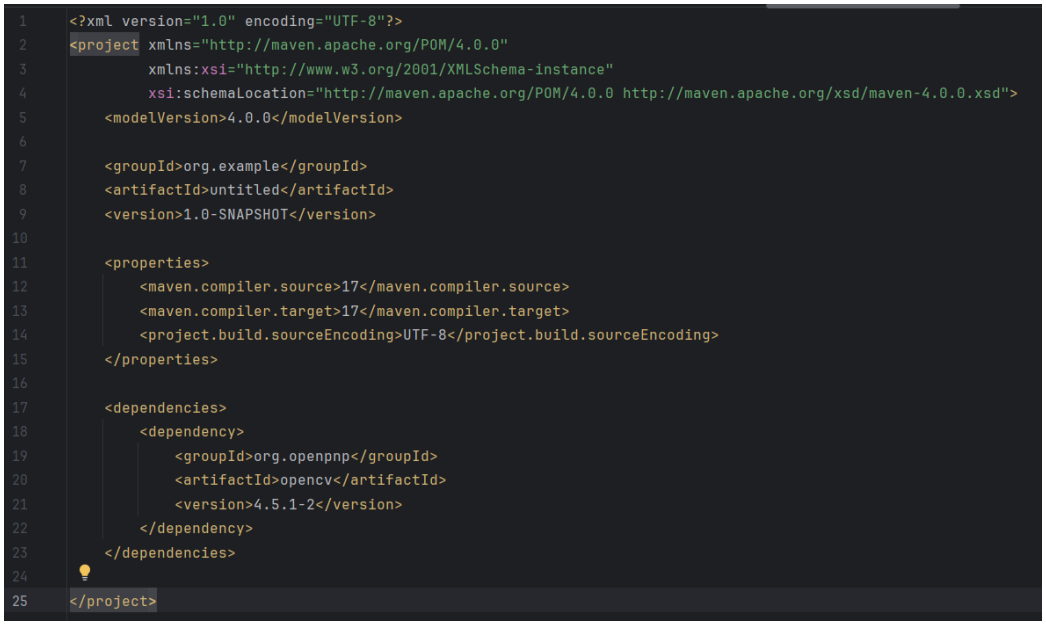
Fonte: Oracle, 2025

4.3.3 Maven

Para auxílio no desenvolvimento de *softwares* existem ferramentas que são capazes de potencializar algumas questões presentes em um sistema, o *maven* é uma dessas ferramentas. Por meio do pom.xml (Project Object Model), um arquivo que disponibiliza informações ao projeto que o *maven* lê baixa as dependências necessárias, compila o código, executa testes e gera o artefato final.

A Figura 7 mostra a utilização de uma dependência no projeto aonde, por meio de uma formatação padrão, é necessário adicionar no escopo de dependências a identificação da dependência chamada de *groupId* refere-se a qual organização ou empresa é responsável pelo pacote, já o *artifactId* é a identificação única do projeto e para finalizar o campo *version* identifica qual a versão está sendo importada. No exemplo é possível ver a dependência *opencv*: Isso indica que o artefato em questão é a biblioteca *opencv*, que está sendo usada pela organização *OpenPnPn* na versão 4.5.1.2.

Figura 7 – Imagem do arquivo pom.xml



Fonte: Elaborado pelo autor, 2026

4.3.4 Domain-Driven Design

No desenvolvimento da aplicação foi adotado o *Domain-Driven Design* também conhecido popularmente como DDD, significa *design* orientado a domínio. É uma abordagem de desenvolvimento de *software* que prioriza entender o modelo a que o sistema se propõe a atender, podendo ser um negócio, como uma biblioteca, uma loja ou uma aplicação que realiza uma função analítica.

Criado por Eric Evans em 2003 foi explicado no livro “Domain-Driven Design: Atacando as Complexidades no Coração do Software” onde o autor conta a motivação do conceito, pois após trabalhar com muitos sistemas complexos e observar casos de sucessos e fracassos o modelo orientado pelo domínio permite uma grande adaptação, permitindo assim chegar a um conceito aplicável.

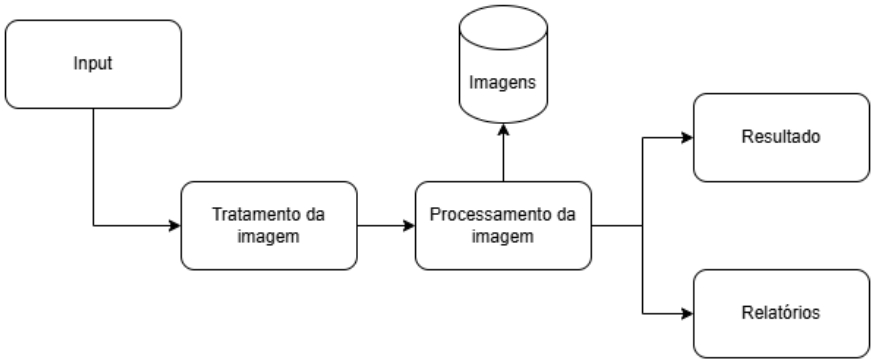
> Passei a última década focado no desenvolvimento de sistemas complexos em diversos domínios técnicos e de negócios. Testei as melhores práticas em design e desenvolvimento, conforme surgiam dos líderes da comunidade de desenvolvimento orientado a objetos. Alguns dos meus projetos foram muito bem-sucedidos; alguns fracassaram. Uma característica comum a todos os sucessos foi um modelo de domínio rico que evoluiu por meio de

> iterações de design e se tornou parte da estrutura do projeto. *(Evans Eric, Domain-Driven Design: Tackling Complexity in the Heart of Software, 1ª edição, Addison-Wesley, 2003, p. 7 (Prefácio).)*.

Com o DDD é necessário, além do domínio, possuir alguns outros conceitos. Como a linguagem Ubíqua um conceito que tem como objetivo definir uma linguagem comum e consistente, possibilitando a interpretação por todas as partes que compõem um projeto. Os contextos delimitados servem a sistemas complexos e objetivam delimitar assuntos entendendo qual conjunto de assuntos melhor se agrupam sem que contextos diferentes corram o risco de se prejudicarem e por fim o mapa de contexto, que é um documento como um diagrama que mostra a relação entre os contextos.

Como mostra o diagrama na figura 8, o contexto da aplicação é sobre o processamento de uma imagem, basicamente a aplicação necessita receber uma imagem, tratar essa imagem preparando para o processamento, executar o processamento acessando as imagens de referência e por fim definir o resultado e gerar um relatório detalhamentos.

Figura 8 – Diagrama de contexto



fonte: Elaborado pelo autor, 2026

4.3.5 Orientação a objetos

O conceito de orientação a objetos trata-se da definição de classes como um objeto que pode conter algumas características que o define, também possui

possibilidade de conter herança e polimorfismo, isso significa estender características e poder ser trabalhada de uma forma ampla.

Com o objetivo de implementar boas práticas de programação a orientação a objetos agrega com conceitos que auxiliam na melhoria de desempenho e facilita implementar a lógica de sistemas complexos. Seus conceitos principais são Abstração, Encapsulamento, Herança e Polimorfismo.

Seguindo a arquitetura, o *software* possui uma classe para iniciar o fluxo de trabalho, esse fluxo iniciado possui acessa a uma classe a qual tem a função de carregar algumas informações montando um objeto que possui definições e características, essa classe é denominada *model*. Esse *model* é o objeto que será submetido aos processos de análise que necessitam de informações contidas nessa classe, essas informações são definidas como Atributos. A imagem abaixo do livro "*Java Básico e Orientação a Objeto de CHAGAS, Clayton Escouper das; BARUQUE, Cássia Blondet; BARUQUE, Lúcia Blondet"* explica essa relação mostrando o que é uma classe, quais podem ser os objetos gerados nela e seus métodos.

Tabela 3 – Explicação do conceito de classe e objeto

CLASSE CARRO		OBJETO CARRO A	OBJETO CARRO B
Atributos de objeto	**Marca**	Ford	Mitsubishi
	Modelo	Fiesta	L-200
	Cor	branco	Azul royal
	combustível	gasolina	diesel
Métodos	ligar		
	acelerar		
	frear		

Fonte: **Java Básico e Orientação a Objeto** página 104

4.3.6 Princípios S.O.L.I.D

Os princípios SOLID foram propostos inicialmente por Robert C. Martin, no livro "Código Limpo". Esses princípios são formados por cinco conceitos de programação orientada a objetos, tem como intenção reduzir a complexidade dos sistemas, diminuir o acoplamento das classes, separar as responsabilidades definindo as relações entre elas. O acrônimo SOLID vem do inglês e abaixo tem seu significado e um resumo de

seu objetivo explicitado em uma tabela encontrada no artigo publicado por “Vinicius Barros Silva Ferreira, Cíntia Antônia Ferreira, Eliana Tani Gomes Grande.”

Tabela 4 – Definições dos princípios que compõem o SOLID.

Nome	Definição
Single Responsibility Principle (Princípio da Responsabilidade Única)	O princípio de responsabilidade única define que uma classe deve ter apenas uma única responsabilidade perante um ator, ou seja, deve conter funções específicas e limitadas que atendama demanda gerada pela funcionalidade (Chebanyuk & Markov, 2016).
Open/Closed Principle (Princípio do Aberto/Fechado)	O princípio do aberto/fechado defende a ideia de que as classes devem ser abertas para extensão e fechadas para modificação, ou seja, devem ser extensíveis sem que isso altere a respectiva classe (Chebanyuk & Markov, 2016; Martin, 2020).
Liskov Substitution Principle (Princípio da Substituição de Liskov)	De acordo com Martin (2020) o princípio da substituição de Liskov define que "para cada objeto de tipo o1 de tipo S, houver um objeto o2 do tipo T, de modo que, para todos os programas P definidos em termos de T, o comportamento de P não seja modificado quando ol for substituído por o2, então S é um subtipo de T". Ou seja, este princípio diz que toda classe que estende uma classe-pai pode ser substituída por outra classe que estende a mesma classe-pai.
Interface Segregation Principle (Princípio da Segregação de Interfaces)	Conforme a definição do princípio de segregação de interfaces, classes que implementam interfaces não devem conter métodos que não serão utilizados. Este princípio auxilia na criação de interfaces específicas para cada contexto, evitando dependências desnecessárias.
Dependency Inversion Principle (Princípio da Inversão de Dependências)	O princípio da inversão de dependências sugere que as dependências de código-fonte se referem apenas a abstrações e não a itens concretos (Martin, 2020). Assim, é possível fazer alterações e expansões de código com mais facilidade.

Fonte: **Clean Architecture e princípios SOLID”,** *Research, Society and Development, v. 11, n. 16, 2022*

5 RESULTADOS E DISCUSSÕES

Com o intuito de mostrar a execução da análise do *software* o capítulo a seguir traz as imagens comparadas, os valores calculados, o resultado e os relatórios registrados. Na sequência o teste de integração com a inteligência artificial.

5.1 Teste Prático

No teste prático foram carregadas 10 imagens na pasta de entrada, em algumas delas existem trincas, já em outras o cordão se encontra íntegro. Na pasta de parâmetro foram adicionadas 12 imagens de análise que apresentam fissuras ou trincas, houve um reaproveitamento nas imagens, com o intuito de aumentar o range de possibilidades, as imagens reaproveitadas foram alteradas no posicionamento e distância do zoom em relação as trincas. A tabela 5 contém os valores resultantes das comparações. Cada comparação armazena apenas o resultado em uma lista para o cálculo médio posteriormente, tratando apenas de comparações locais, sem armazenamento ou modelagens de aprendizado.

Tabela 5 - Resultados comparação

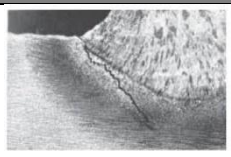
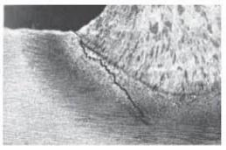
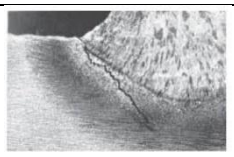
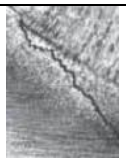
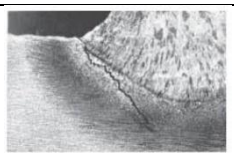
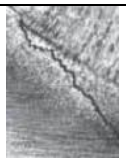
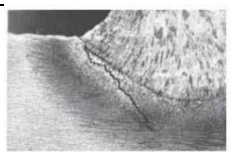
Entrada		Parâmetro		Resultado
Nome arquivo	imagem	Nome arquivo	imagem	
imagem1.jpg		imagem1.jpg		1
imagem1.jpg		imagem10.jpg		0,98
imagem1.jpg		imagem11.jpg		0,98
imagem1.jpg		imagem12.jpg		0,98

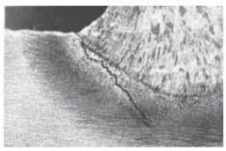
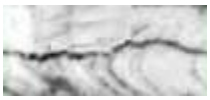
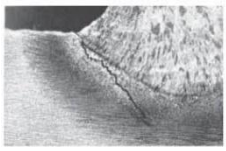
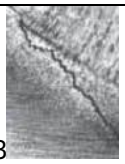
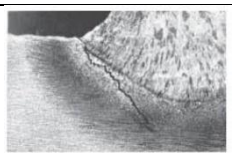
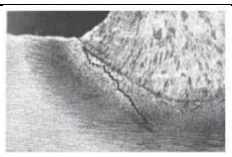
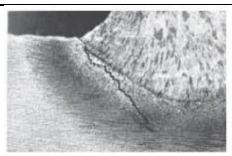
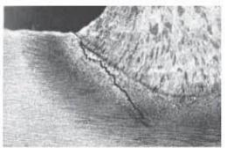
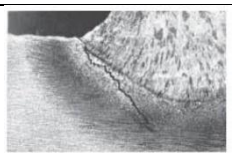
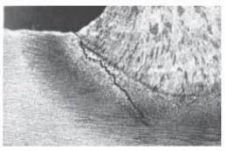
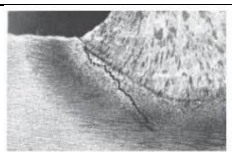
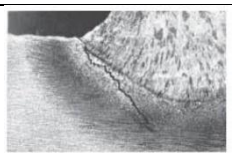
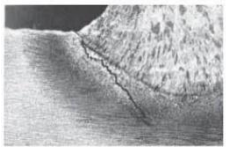
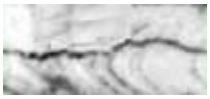
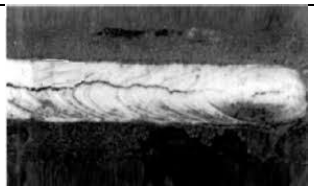
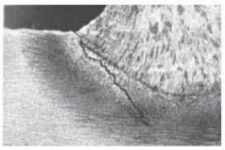
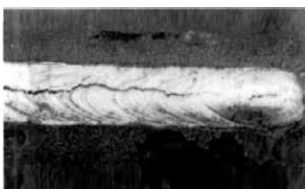
imagem1.jpg		imagem2.jpg		0,13
imagem1.jpg		imagem3.jpg	 3	0,98
imagem1.jpg		imagem4.jpg		1
imagem1.jpg		imagem5.jpg		1
imagem1.jpg		imagem6.jpg		1
imagem1.jpg		imagem7.jpg		0,13
imagem1.jpg		imagem8.jpg		0,13
imagem1.jpg		imagem9.jpg		0,13
imagem2.jpg		imagem1.jpg		0,13
imagem2.jpg		imagem10.jpg		-0,02

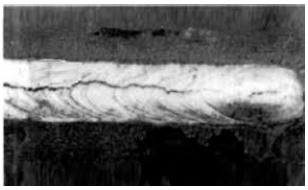
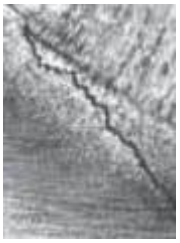
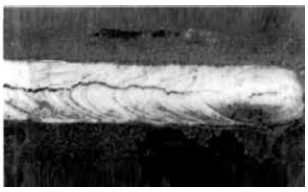
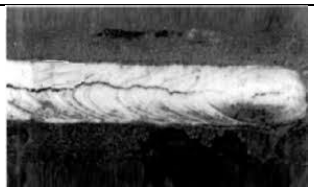
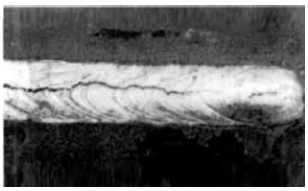
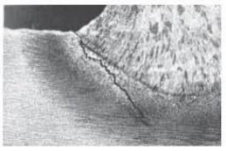
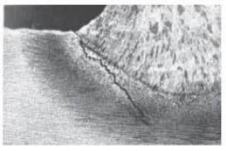
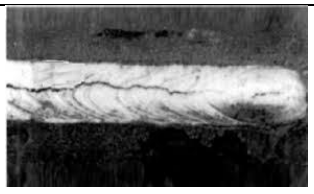
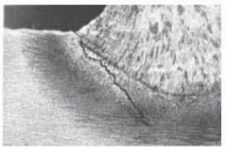
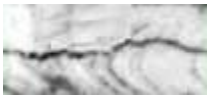
imagem2.jpg		imagem11.jpg		-0,02
imagem2.jpg		imagem12.jpg		-0,02
imagem2.jpg		imagem2.jpg		1
imagem2.jpg		imagem3.jpg		-0,02
imagem2.jpg		imagem4.jpg		0,13
imagem2.jpg		imagem5.jpg		0,13
imagem2.jpg		imagem6.jpg		0,13
imagem2.jpg		imagem7.jpg		1

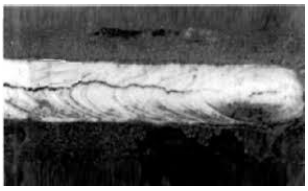
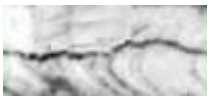
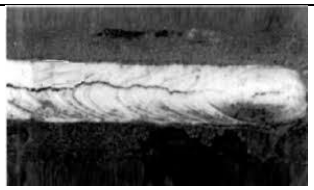
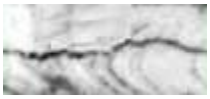
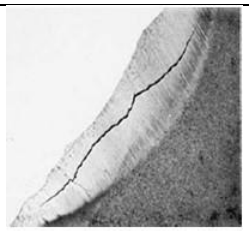
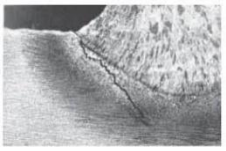
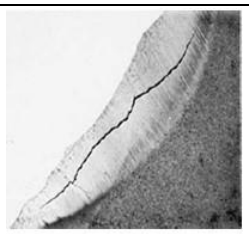
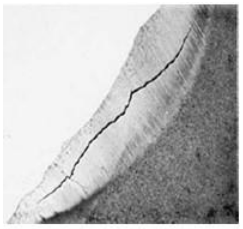
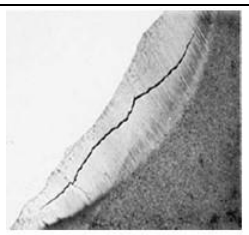
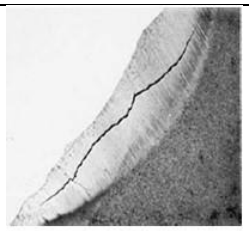
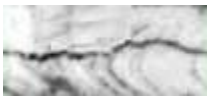
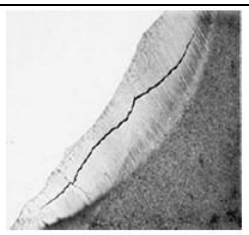
imagem2.jpg		imagem8.jpg		1
imagem2.jpg		imagem9.jpg		1
imagem3.jpg		imagem1.jpg		0,13
imagem3.jpg		imagem10.jpg		-0,02
imagem3.jpg		imagem11.jpg		-0,02
imagem3.jpg		imagem12.jpg		-0,02
imagem3.jpg		imagem2.jpg		1
imagem3.jpg		imagem3.jpg		-0,02

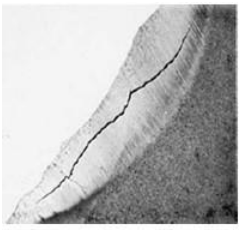
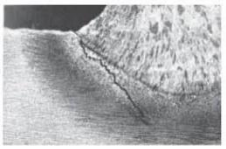
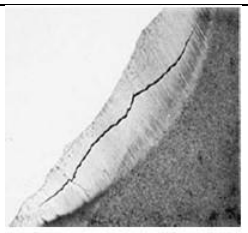
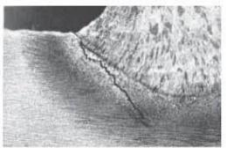
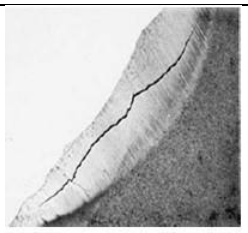
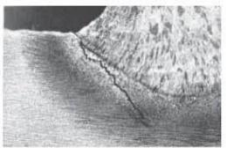
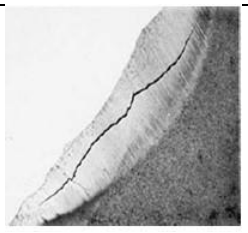
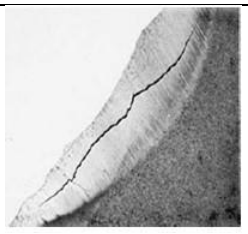
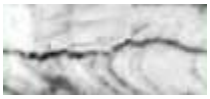
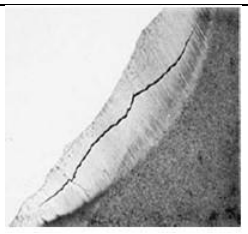
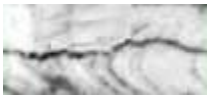
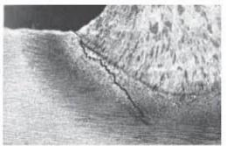
imagem3.jpg		imagem4.jpg		0,13
imagem3.jpg		imagem5.jpg		0,13
imagem3.jpg		imagem6.jpg		0,13
imagem3.jpg		imagem7.jpg		1
imagem3.jpg		imagem8.jpg		1
imagem3.jpg		imagem9.jpg		1
imagem4.jpg		imagem1.jpg		-0,03

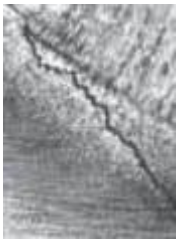
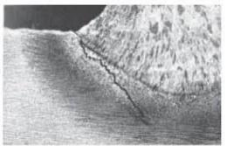
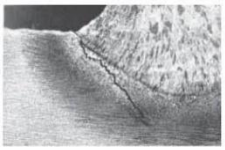
imagem4.jpg		imagem10.jpg		-0,03
imagem4.jpg		imagem11.jpg		-0,03
imagem4.jpg		imagem12.jpg		-0,03
imagem4.jpg		imagem2.jpg		0,04
imagem4.jpg		imagem3.jpg		-0,03
imagem4.jpg		imagem4.jpg		-0,03
imagem4.jpg		imagem5.jpg		-0,03

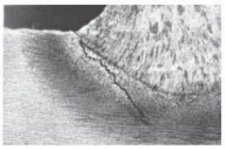
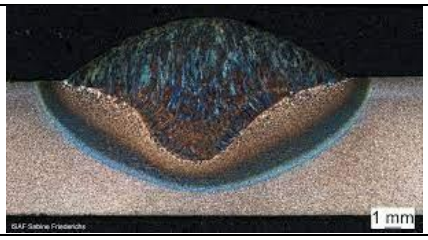
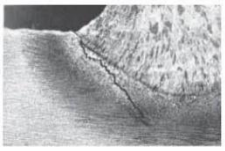
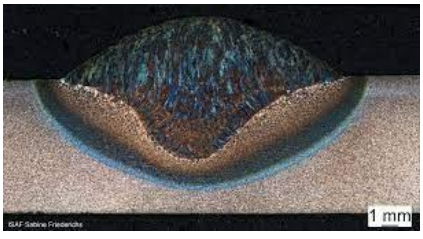
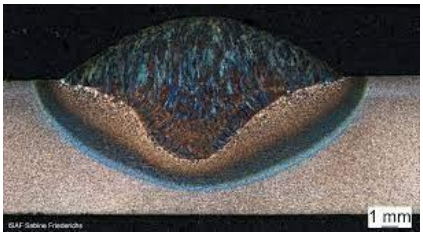
imagem4.jpg		imagem6.jpg		-0,03
imagem4.jpg		imagem7.jpg		0,04
imagem4.jpg		imagem8.jpg		0,04
imagem4.jpg		imagem9.jpg		0,04
imagem5.jpg		imagem1.jpg		0,1
imagem5.jpg		imagem10.jpg		0,1
imagem5.jpg		imagem11.jpg		0,1

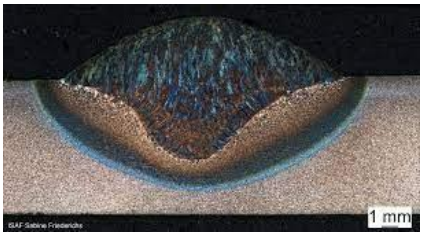
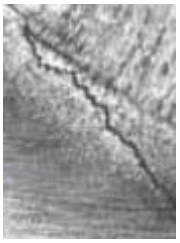
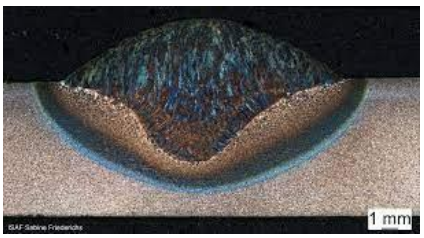
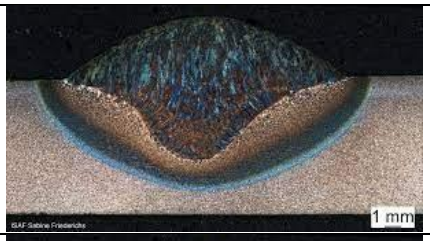
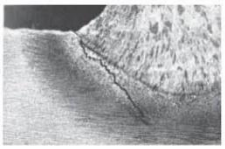
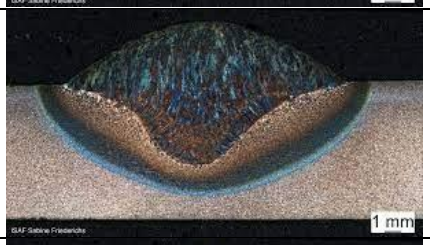
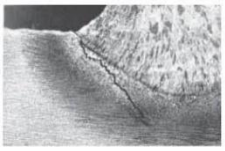
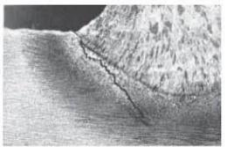
imagem5.jpg		imagem12.jpg		0,1
imagem5.jpg		imagem2.jpg		0,06
imagem5.jpg		imagem3.jpg		0,1
imagem5.jpg		imagem4.jpg		0,1
imagem5.jpg		imagem5.jpg		0,1
imagem5.jpg		imagem6.jpg		0,1
imagem5.jpg		imagem7.jpg		0,06
imagem5.jpg		imagem8.jpg		0,06

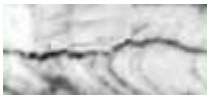
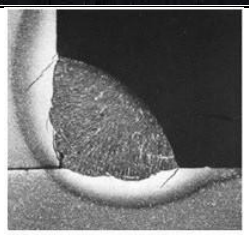
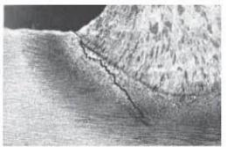
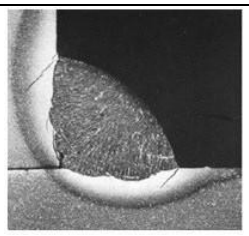
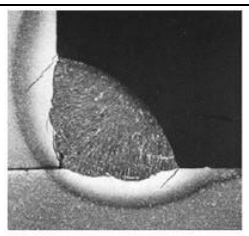
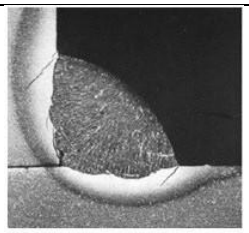
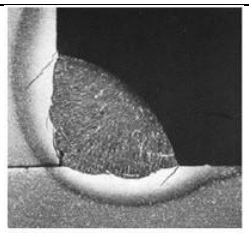
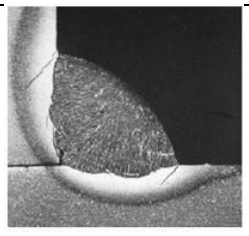
imagem5.jpg		imagem9.jpg		0,06
imagem6.jpg		imagem1.jpg		0,13
imagem6.jpg		imagem10.jpg		-0,02
imagem6.jpg		imagem11.jpg		-0,02
imagem6.jpg		imagem12.jpg		-0,02
imagem6.jpg		imagem2.jpg		1
imagem6.jpg		imagem3.jpg		-0,02

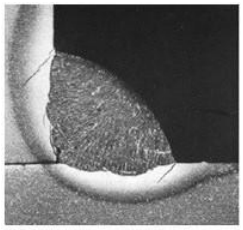
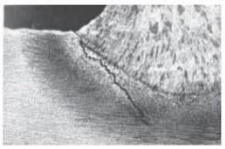
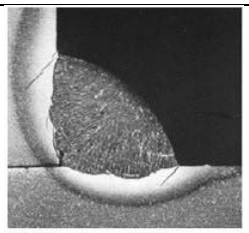
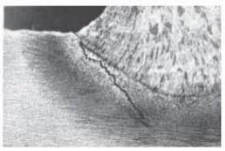
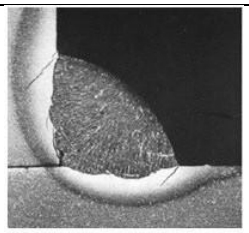
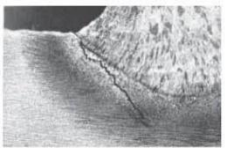
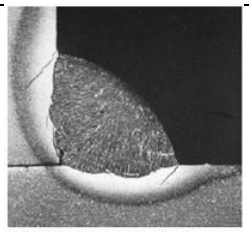
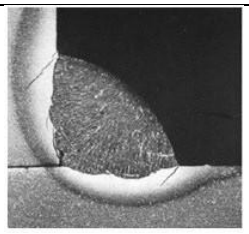
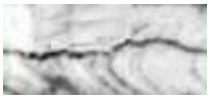
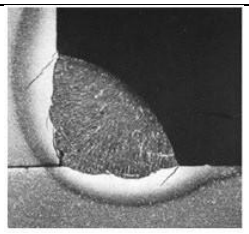
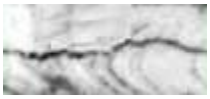
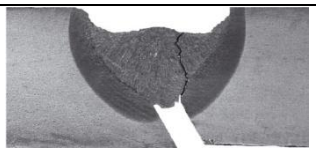
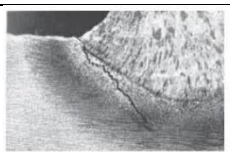
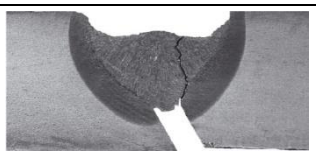
imagem6.jpg		imagem4.jpg		0,13
imagem6.jpg		imagem5.jpg		0,13
imagem6.jpg		imagem6.jpg		0,13
imagem6.jpg		imagem7.jpg		1
imagem6.jpg		imagem8.jpg		1
imagem6.jpg		imagem9.jpg		1
imagem7.png		imagem1.jpg		0,2
imagem7.png		imagem10.jpg		0,11

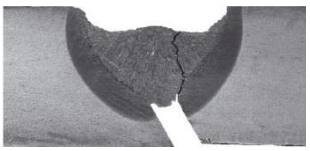
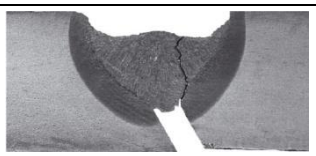
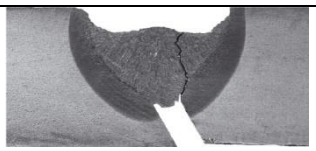
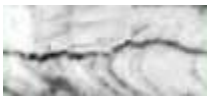
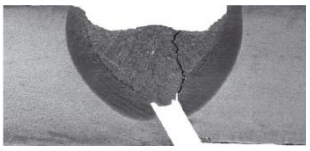
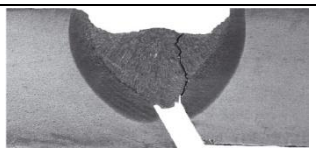
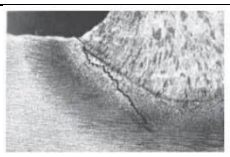
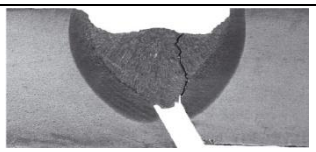
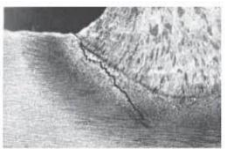
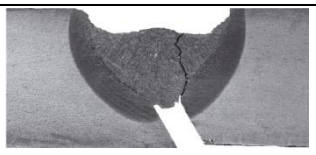
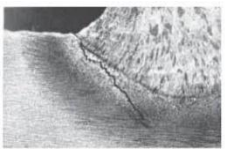
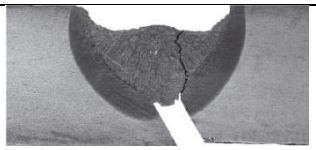
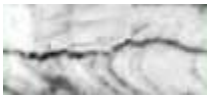
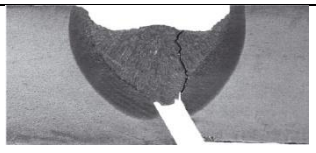
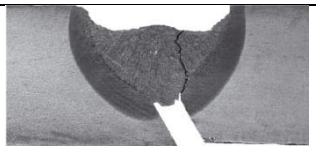
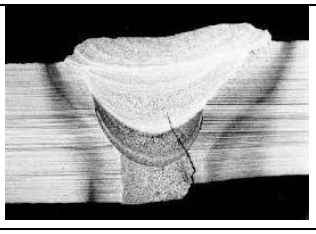
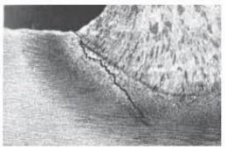
imagem7.png		imagem11.jpg		0,11
imagem7.png		imagem12.jpg		0,11
imagem7.png		imagem2.jpg		0,2
imagem7.png		imagem3.jpg		0,11
imagem7.png		imagem4.jpg		0,2
imagem7.png		imagem5.jpg		0,2
imagem7.png		imagem6.jpg		0,2
imagem7.png		imagem7.jpg		0,2
imagem7.png		imagem8.jpg		0,2
imagem7.png		imagem9.jpg		0,2
imagem8.jpg		imagem1.jpg		0,13

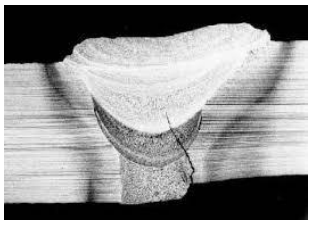
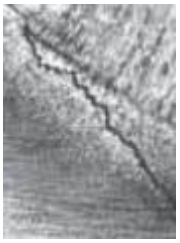
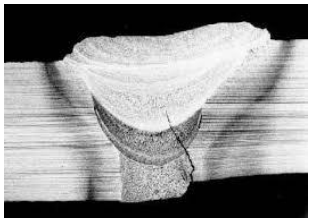
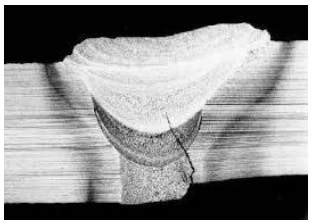
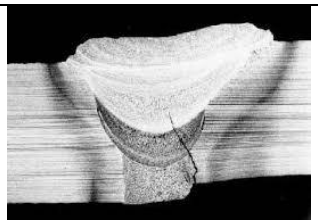
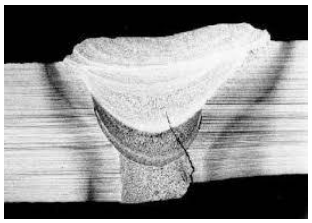
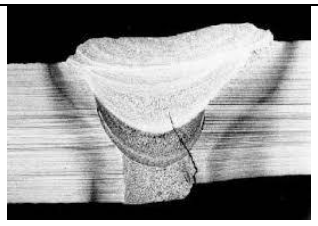
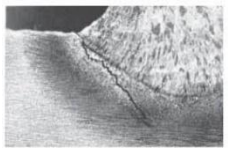
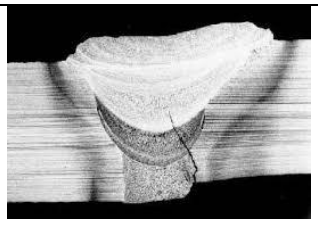
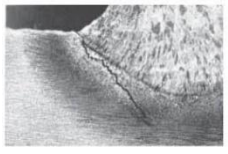
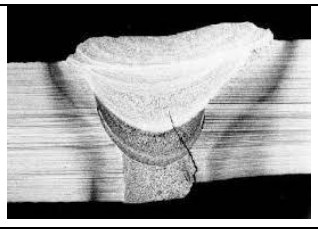
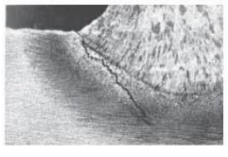
imagem8.jpg		imagem10.jpg		-0,02
imagem8.jpg		imagem11.jpg		-0,02
imagem8.jpg		imagem12.jpg		-0,02
imagem8.jpg		imagem2.jpg		1
imagem8.jpg		imagem3.jpg		-0,02
imagem8.jpg		imagem4.jpg		0,13
imagem8.jpg		imagem5.jpg		0,13
imagem8.jpg		imagem6.jpg		0,13

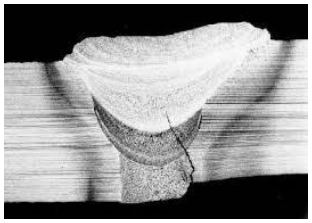
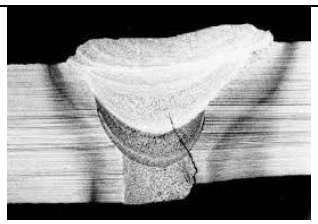
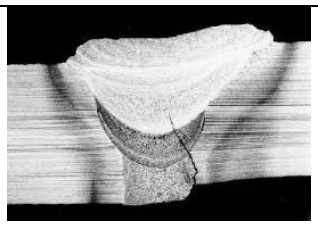
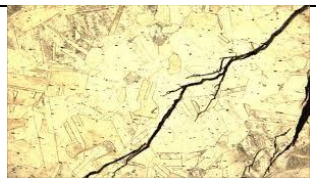
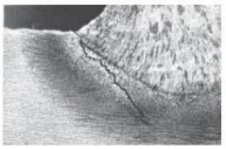
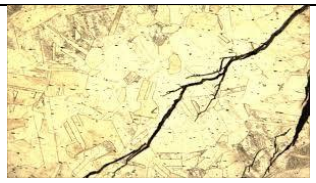
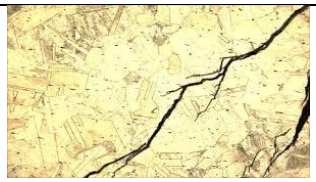
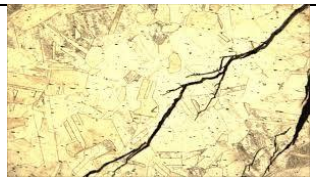
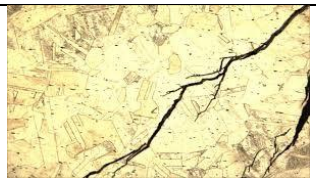
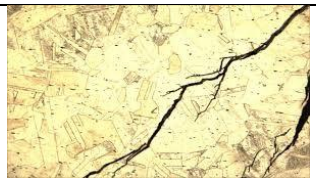
imagem8.jpg		imagem7.jpg		1
imagem8.jpg		imagem8.jpg		1
imagem8.jpg		imagem9.jpg		1
imagem9.jpg		imagem1.jpg		-0,04
imagem9.jpg		imagem10.jpg		-0,03
imagem9.jpg		imagem11.jpg		-0,03
imagem9.jpg		imagem12.jpg		-0,03
imagem9.jpg		imagem2.jpg		-0,03
imagem9.jpg		imagem3.jpg		-0,03

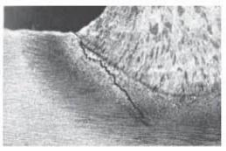
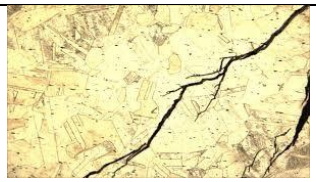
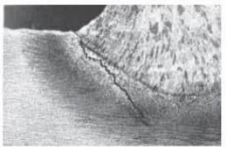
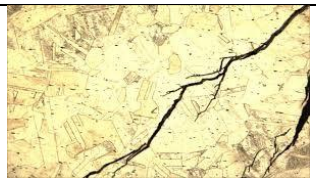
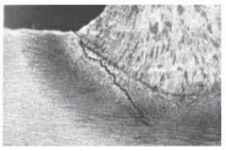
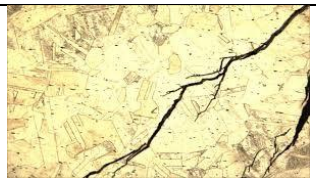
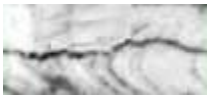
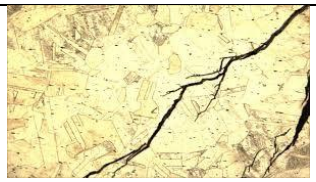
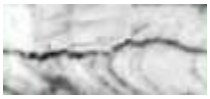
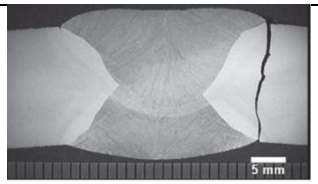
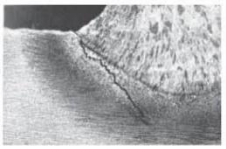
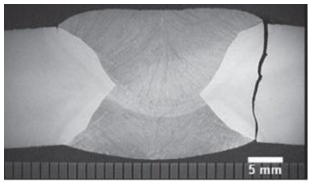
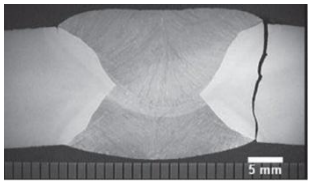
imagem9.jpg		imagem4.jpg		-0,04
imagem9.jpg		imagem5.jpg		-0,04
imagem9.jpg		imagem6.jpg		-0,04
imagem9.jpg		imagem7.jpg		-0,03
imagem9.jpg		imagem8.jpg		-0,03
imagem9.jpg		imagem9.jpg		-0,03
imagem10.png		imagem1.jpg		0,28
imagem10.png		imagem10.jpg		0,26
imagem10.png		imagem11.jpg		0,26

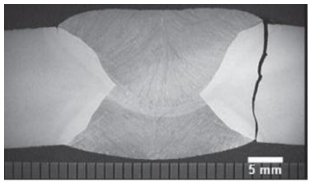
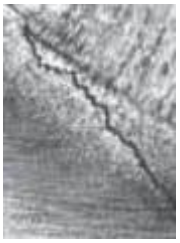
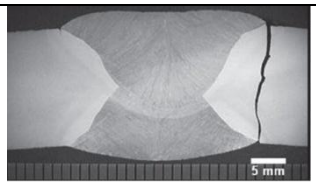
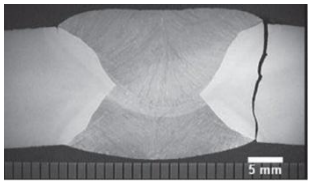
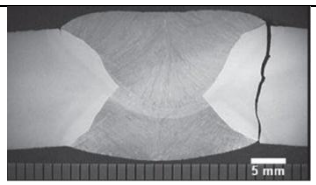
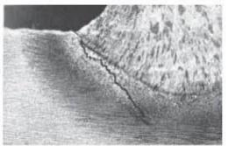
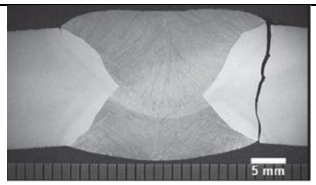
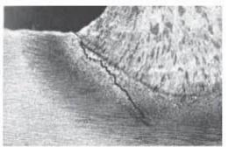
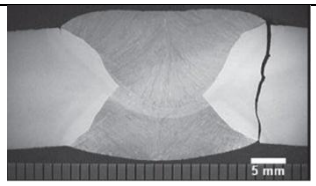
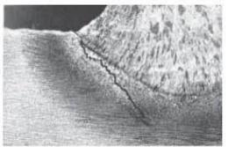
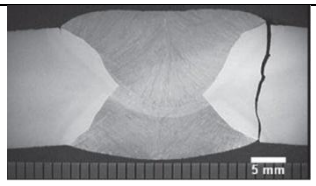
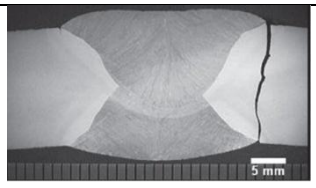
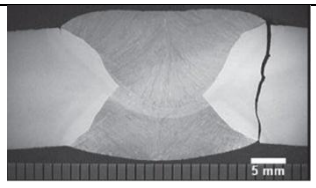
imagem10.png		imagem12.jpg		0,26
imagem10.png		imagem2.jpg		-0,02
imagem10.png		imagem3.jpg		0,26
imagem10.png		imagem4.jpg		0,28
imagem10.png		imagem5.jpg		0,28
imagem10.png		imagem6.jpg		0,28
imagem10.png		imagem7.jpg		-0,02
imagem10.png		imagem8.jpg		-0,02
imagem10.png		imagem9.jpg		-0,02

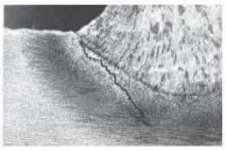
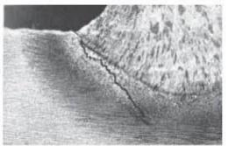
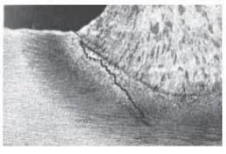
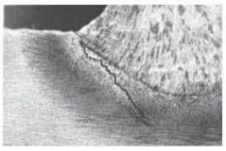
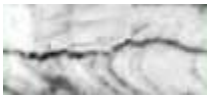
imagem11.jpg		imagem1.jpg		0,13
imagem11.jpg		imagem10.jpg		-0,02
imagem11.jpg		imagem11.jpg		-0,02
imagem11.jpg		imagem12.jpg		-0,02
imagem11.jpg		imagem2.jpg		1
imagem11.jpg		imagem3.jpg		-0,02
imagem11.jpg		imagem4.jpg		0,13
imagem11.jpg		imagem5.jpg		0,13

imagem11.jpg		imagem6.jpg		0,13
imagem11.jpg		imagem7.jpg		1
imagem11.jpg		imagem8.jpg		1
imagem11.jpg		imagem9.jpg		1

Fonte: Elaborado pelo autor, 2026

Ao final da análise é feita a média dos valores das comparações, ou seja, todos os valores resultantes da imagem 1 com as 12 imagens de parâmetro são somados e divididos pela quantidade de imagens de parâmetro. Como mostrado na tabela 6.

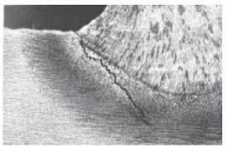
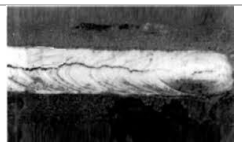
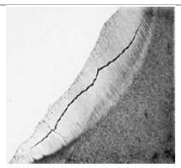
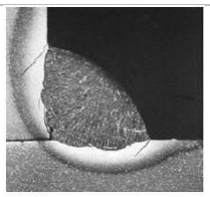
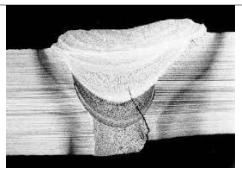
Tabela 6 – Valores médios calculados

IMAGEM	VALOR MÉDIO
imagem1.jpg	70,44
imagem10.png	17,31
imagem11.jpg	37,06
imagem2.jpg	37,06
imagem3.jpg	37,06
imagem4.jpg	-0,71
imagem5.jpg	8,84
imagem6.jpg	37,06
imagem7.png	17,12
imagem8.jpg	37,06
imagem9.jpg	-3,08

Fonte: Elaborado pelo autor, 2026

O valor médio considerado foi a média acima de 30,00. Com esse valor o resultado foi satisfatório pois conseguiu encontrar todas as imagens que possuíam algum tipo de trinca como mostra a tabela 7.

Tabela 7 – Resultado da análise do software

NOME ARQUIVO	VALOR MÉDIO	IMAGEM
imagem1.jpg	70,44	
imagem2.jpg	37,06	
imagem3.jpg	37,06	
imagem6.jpg	37,06	
imagem8.jpg	37,06	
imagem11.jpg	37,06	

Fonte: Elaborado pelo autor, 2026

5.2 Custo computacional

Com informações retiradas da console do projeto ele apresentou o tempo total de execução de 766 ms (milissegundos)

Convertendo para segundos:

$$766\ ms = 0{,}766\ s$$

Portanto:

$$T = 0{,}766\ segundos$$

11 imagens comparadas contra 12 imagens.

Logo:

$$N = 11 \times 12$$

$$N = 132\ comparações$$

Agora calculamos quanto tempo, em média, cada comparação levou.

$$TEMPO\ médio = \frac{TEMPO\ total}{N}$$

Substituindo:

$$TEMPO\ médio = \frac{0{,}766}{132}$$

$$TEMPO\ médio = 0{,}00580\ s$$

Convertendo para milissegundos:

$$0{,}00580\ s = 5{,}80\ ms$$

Cada comparação levou um tempo médio de 5,80 ms. Assumindo um computador local com consumo médio de 65 Watts

Convertendo:

$$65W = 0{,}065\ kW$$

Tempo total em horas:

$$0{,}766\ s = \frac{0{,}766}{3600} = 0{,}000213\ h$$

Energia consumida:

$$E = P \times t$$

$$E = 0{,}065 \times 0{,}000213$$

$$E = 0{,}0000138\ kWh$$

Tarifa média: R$ 1,00/kWh

$$Custo = E \times Tarifa$$

$$Custo = 0{,}0000138 \times 1{,}00$$

$$Custo \approx 0{,}000014$$

A tabela 8 mostra o resumo do tempo, a estimativa do custo de nergia baseada no tempo de execução do projeto.

Tabela 8 – Custo computacional

Métrica	Valor
Tempo total do processamento	766 ms (0,766 s)
Número total de comparações	132
Tempo médio por comparação	5,8 ms
Energia estimada consumida	0,0000138 kWh
Custo aproximado por execução	R$ 0,000014

Fonte: Elaborado pelo autor, 2026

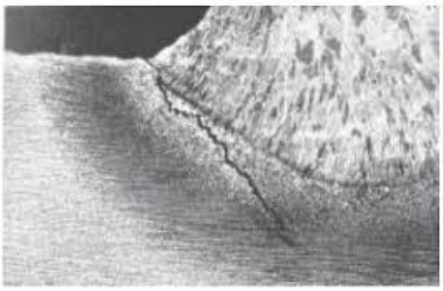
5.3 Integração com Inteligência Artificial

No projeto foi feita uma implementação enviando uma imagem de parâmetro e solicitando uma análise a I.A. A implementação foi feita por uma API (*Application Programming Interface*) chamando diretamente o serviço *web* oferecido pela empresa OpenIa.

A implementação da função *sendImage*, configura e efetua a chamada na API externa selecionando modelo recuperando a chave e o *script* juntamente com a imagem a ser analisada.

A imagem em questão possui uma trinca e foi enviada com o seguinte *prompt*: "Existe alguma trinca na imagem?" e obteve a seguinte resposta dada pelo modelo gpt-4o: Sim — a imagem mostra claramente uma trinca. Ela aparece como uma linha escura e irregular que se propaga a partir da região mais clara (provavelmente a zona afetada pelo calor ou a área fundida) em direção ao material base. O padrão é típico de trinca em solda ou trinca por solidificação/contração térmica.

Figura 9 – Imagem com trinca enviada



Fonte: Infosolda, 2003

As chamadas enviadas possuem um custo determinado pela empresa que fornece a inteligência artificial sendo limitada pela quantidade de *tokens* consumidos, esses *tokens* são mensurados pelas palavras contidas no prompt podendo variar seu consumo.

Figura 10 – Tabela de preços OpenIa

API Realtime

Crie experiências multimodais de baixa latência, incluindo conversão de voz em voz.

Imagem	gpt-realtime	US$ 5,00 / 1 milhão de tokens de entrada	US$ 0,50 / 1 milhão de tokens de entrada em cache	-
	gpt-realtime-mini	US$ 0,80 / 1 milhão de tokens de entrada	US$ 0,08 / 1 milhão de tokens de entrada em cache	-

Fonte: OpenIA, 2026

6 CONCLUSÃO

Como a ideia de desenvolver um software que atenda a uma demanda de situações rotineiras na engenharia, foi possível estudar áreas diversas e abstrair de todas conceitos muito importantes, enriquecer o conhecimento sobre todas as áreas durante as pesquisas e entender durante os registros conceitos e práticas adotadas.

No desenvolvimento do trabalho é possível perceber questões muito importantes sobre diversos aspectos abordados na área de metalografia, soldagem, e desenvolvimento de software, inicialmente é perceptível o valor da análise

metalográfica e o quanto ela agrega no conhecimento do material trabalhado pois possibilita uma visão próxima e criteriosa, que somente com uma observação microscópica é possível ver a composição verdadeira do material. Análises de solda por sua vez mostram a influência do processo em um material e em alguns casos podem identificar falhas e ser essencial para descrever e melhorar processos.

No desenvolvimento do software se mostrou de extrema importância o conhecimento sobre a arquitetura, mesmo em projetos simples, auxilia muito na organização da sequência de execução das etapas e conceitos como a singularidade das classes e objetivo único aumentam a eficiência de manutenção e velocidade de execução. A utilização de bibliotecas gratuitas possibilita explorar ferramentas úteis com muito mais velocidade pois partem de uma lógica complexa e elaborada já testada e validada antes da publicação. Com a aplicação das ferramentas disponibilizadas pela biblioteca openCV, foi necessário entender como o as funções tratam as imagens para aplicar assim a melhor análise voltada especificamente esse formato de arquivo. As teorias envolvendo o tratamento de cores e combinações convertidas para modelos matemáticos explicam perfeitamente de onde partem os princípios das análises das imagens.

Com o resultado satisfatório do teste do software foi possível visualizar como a análise de comparação trabalha e se mostra eficiente em seus resultados encontrados assim que atende perfeitamente ao objetivo e entrega analises seguras e relatórios satisfatórios, com o teste foi possível perceber também que uma base de referência maior pode tornar as análises mais seguras e assertivas, mesmo que isso leve um tempo a mais na execução das analises, o limite de imagens como parâmetro pode crescer ou ser estipulado de acordo com a necessidade determinada pelo usuário.

A existência de um software capaz de analisar as imagens traz vantagens e desvantagens em relação as inteligências artificiais (I.A). Inicialmente a vantagem de se considerar a independência de modelos de I.A, é que não há uma empresa terceira que possa deter o conhecimento dos dados e cobram por serviços, podendo variar a ponto de ser impossível se sustentar e limitar a utilização, possuir o software te dá liberdade de estender as funcionalidades e alterá-lo de acordo com as necessidades. Em contrapartida a I.A pode oferecer resultados satisfatórios pois os modelos além de muito bem desenvolvidos contém um amplo acesso aos arquivos na internet e associa não só imagens, mas também conceitos

REFERÊNCIAS

ANJOS, Paulo N. M. dos; FAZENDA, Alvaro L. Avaliando eficiência energética em padrões de algoritmos para computação científica e de alto desempenho. *In*: WORKSHOP DE INICIAÇÃO CIENTÍFICA - SIMPÓSIO EM SISTEMAS COMPUTACIONAIS DE ALTO DESEMPENHO (SSCAD), 24., 2023, Porto Alegre/RS. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2023. p. 49-56. DOI: https://doi.org/10.5753/wscad_estendido.2023.235760.

CALLISTER, William D; RETHWISCH, Jr., David G. **Ciência e engenharia de materiais: uma introdução.** tradução Sergio Murilo Stamile Soares; revisão técnica José Roberto Moraes d'Almeida. 8 ed. Rio de Janeiro: LTC, 2013.

CHAGAS, Clayton Escouper das; BARUQUE, Cássia Blondet; BARUQUE, Lúcia Blondet. **Java Básico e Orientação a Objeto**: volume único. Rio de Janeiro: Fundação CECIERJ, 2010. 238 p.

Claro D. B. e Sobral J. B. M. **Programação em Java**. Copyleft Pearson Education. Florianopolis, SC, 2008.

CLÁUDIO, E. et al. **Fissuração pelo Hidrogênio "Trincas a Frio"**. [s.l: s.n.]. Disponível em: <http://ftp.demec.ufpr.br/disciplinas/EME733/Arquivos%20da%20disciplina/Trinca%20a%20frio.pdf>.

COLPAERT, Hubertus. **Metalografia dos produtos siderúrgicos comuns.** revisão técnica André Luiz V. da Costa e Silva. - 4 ed. São Paulo: Blucher, 2008.

EVANS, Eric. **Domain-Driven Design: Tackling Complexity in the Heart of Software**. Final Manuscript, April 15, 2003. Disponível em: www.domainlanguage.com.

Gonzalez, R. C.; Woods, R. C. **Processamento Digital de Imagens**. *3*. ed. São Paulo: Pearson Prentice Hall, 2010.

https://g1.globo.com/Noticias/Tecnologia/0,,MUL1091457-6174,00-ORACLE+ANUNCIA+COMPRA+DA+SUN+POR+MAIS+DE+US+BILHOES.html

TOURINHO; TOURINHO. **Ensaio macrográfico - Infosolda**. Disponível em: https://infosolda.com.br/219-ensaio-macrografico/.

Joly, Martine (1994) — **Introdução à Análise da Imagem**, Lisboa, Ed. 70, 2007 — Digitalizado por SOUZA, R.

OPENCV. *Bounding rectangles and circles*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/da/d0c/tutorial_bounding_rects_circles.html. Acesso em: 12 jan. 2026.

OPENCV. *Color space conversions*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/de/d25/imgproc_color_conversions.html. Acesso em: 12 jan. 2026.

OPENCV. *Convex hull*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/d7/d1d/tutorial_hull.html. Acesso em: 12 jan. 2026.

OPENCV. *Finding contours*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/df/d0d/tutorial_find_contours.htm. Acesso em: 12 jan. 2026.

OPENCV. *Histogram back projection*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/da/d7f/tutorial_back_projection.html. Acesso em: 12 jan. 2026.

OPENCV. *Histogram calculation*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/d8/dbc/tutorial_histogram_calculation.html. Acesso em: 12 jan. 2026.

OPENCV. *Histogram comparison*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/d8/dc8/tutorial_histogram_comparison.html. Acesso em: 12 jan. 2026.

OPENCV. *Histogram equalization*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/d4/d1b/tutorial_histogram_equalization.html. Acesso em: 12 jan. 2026.

OPENCV. *Template matching*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/de/da9/tutorial_template_matching.html. Acesso em: 12 jan. 2026.

OPENCV. *Warp affine transformations*. OpenCV Documentation, versão 3.4. Disponível em: https://docs.opencv.org/3.4/d4/d61/tutorial_warp_affine.html. Acesso em: 12 jan. 2026.

ORACLE. **Java Software | Oracle**. Disponível em: https://www.oracle.com/java/.

Tabela de Preços Open Ia. Disponível em: https://openai.com/pt-BR/api/pricing/.

TERRA, Ernani**. Minidicionário da Língua Portuguesa** Ernani Terra. 2. ed. São Paulo: Rideel, 2011.

APÊNDICE A – SOFTWARE DETALHADO

O início do funcionamento do sistema se dá por uma classe denominada “main” nela a principal responsabilidade é direcionar ao processo que fará a análise de fato, chamado de *workflow*. Além disso existem sinalizações sobre o início do processamento e o fim. Como o sistema roda apenas localmente a máquina apenas uma mensagem satisfaz, não sendo necessários *logs*.

```
package org.example;

import org.example.domain.AnaliseImageWorkflow;

public class Main {
    public static void main(String[] args) {
        AnaliseImageWorkflow analiseImageWorkflow = new
AnaliseImageWorkflow();

        System.out.println("inicio projeto!");

        analiseImageWorkflow.execute();

        System.out.println("fim do projeto!");
    }
}
```

fonte: elaborado pelo autor, 2026

A classe de constantes tem a função de ser o local de definição de todas as contantes do código, podendo ser acessadas em diversos pontos.

```
package core;

public class AppConstants {

    public static class fileConstants {
        public static final String CAMINHO_ENTRADA =
"C:\\Users\\cicer\\Desktop\\imagens\\entrada\\";
        public static final String CAMINHO_PARAMETRO =
"C:\\Users\\cicer\\Desktop\\imagens\\parametro\\";
        public static final String CAMINHO_OUTPUT =
"C:\\Users\\cicer\\Desktop\\imagens\\resultado\\";
    }

    public static class calculateConstants {
        public static Double VALOR_MEDIO_MINIMO = 30.0;
        public static Double VALOR_MINIMO_COMPARACAO = 90.0;
    }

}
```

fonte: elaborado pelo autor, 2026

A classe chamada de “AnaliseImageWorkflow” tem a função de definir a sequência das demais operações, definindo alguns parâmetros iniciais como as rotas de onde as imagens serão chamadas sua principal responsabilidade é orquestrar as etapas do sistema na sequência necessária para realização das análises de imagem e geração de relatórios.

```
package domain;

import domain.comparison.CalculateMediaActivity;
import domain.comparison.GetImageActivity;
import domain.comparison.HistogramComparison;
import domain.comparison.SelectOutputImage;
import domain.relatorios.CsvExporter;
import domain.relatorios.model.RetornoComparacaoModel;

import java.util.ArrayList;
import java.util.List;

import static core.AppConstants.fileConstants.*;

public class AnaliseImageWorkflow {


    GetImageActivity getImageActivity = new GetImageActivity();
    HistogramComparison histogramComparison = new HistogramComparison();
    CalculateMediaActivity calculateMediaActivity = new
CalculateMediaActivity();
    CsvExporter csvExporter = new CsvExporter();
    SelectOutputImage selectOutputImage = new SelectOutputImage();

    public void execute(){
        var inputList = getImageActivity.listarImagens(CAMINHO_ENTRADA);
        var parameterList =
getImageActivity.listarImagens(CAMINHO_PARAMETRO);
        List<RetornoComparacaoModel> retornoComparacaoList = new
ArrayList<>();

        inputList.stream()
                        .forEach(input -> parameterList.stream()
                                .forEach(parameter -> {
                                    RetornoComparacaoModel result =
histogramComparison.executeComparison(input, parameter);
                                    retornoComparacaoList.add(result);
                                }));

        var valorMedioImagensList =
calculateMediaActivity.calcularMediaImagens(inputList,retornoComparacaoList
);
        var outputImages =
selectOutputImage.execute(valorMedioImagensList,retornoComparacaoList);
        csvExporter.exportToCsv(retornoComparacaoList, "output.csv");
        csvExporter.exportMediaToCsv(valorMedioImagensList,
"outputMedia.csv");
    }

}
```

fonte: elaborado pelo autor, 2026

Classe para manipular os arquivos.

```
package domain.comparison;

import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.List;

public class GetImageActivity {

    // Função para listar os nomes das imagens em uma pasta
    public static List<String> listarImagens(String caminhoPasta) {
        List<String> nomesImagens = new ArrayList<>();
        File pasta = new File(caminhoPasta);

        // Verificar se o caminho é um diretório válido
        if (pasta.exists() && pasta.isDirectory()) {
            File[] arquivos = pasta.listFiles((dir, nome) -> {
                String nomeLower = nome.toLowerCase();
                return nomeLower.endsWith(".jpg") ||
nomeLower.endsWith(".jpeg") || nomeLower.endsWith(".png");
            });

            if (arquivos != null) {
                for (File arquivo : arquivos) {
                    nomesImagens.add(arquivo.getName());
                }
            }
        }
        return nomesImagens;
    }

    // Função para copiar uma imagem para outra pasta
    public static void copiarImagem(String caminhoOrigem, String
caminhoDestino) {
        Path origem = Paths.get(caminhoOrigem);
        Path destino = Paths.get(caminhoDestino);

        try {
            Files.copy(origem, destino);
            System.out.println("Imagem copiada para: " + destino);
        } catch (IOException e) {
            System.err.println("Erro ao copiar imagem: " + origem + " para
" + destino + ": " + e.getMessage());
        }
    }

}
```

fonte: elaborado pelo autor, 2026

Seguindo o fluxo, a etapa inicial de captar as imagens presente na classe de *workflow* direciona para a classe de “HistogramComparison”, nessa classe o processamento da imagem ocorrerá de fato calculando e normalizando histogramas baseados nas escalas de cores. Essas funções da biblioteca open cv possibilitam resultados numéricos capazes de quantificar e basear em números das cores presentes nas imagens. Convertendo uma imagem que possui dados de cores que são utilizados para serem renderizados nas telas por meio dos *pixels*, em escalas numéricas denominadas histogramas.

```
package domain.comparison;

import domain.relatorios.model.RetornoComparacaoModel;
import org.opencv.core.*;
import org.opencv.imgcodecs.Imgcodecs;
import org.opencv.imgproc.Imgproc;
import static core.AppConstants.fileConstants.*;
import static domain.comparison.SelectOutputImage.*;

public class HistogramComparison {

    public RetornoComparacaoModel executeComparison(String input, String
parameter) {
        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);

        // Carrega as duas imagens
        Mat image1 = Imgcodecs.imread(CAMINHO_INPUT.concat(input));
        Mat image2 = Imgcodecs.imread(CAMINHO_PARAMETRO.concat(parameter));
        System.out.println(String.format("%s x %s: ", input, parameter));

        // Converte para o espaço de cores HSV
        Mat hsvImage1 = new Mat();
        Mat hsvImage2 = new Mat();
        Imgproc.cvtColor(image1, hsvImage1, Imgproc.COLOR_BGR2HSV);
        Imgproc.cvtColor(image2, hsvImage2, Imgproc.COLOR_BGR2HSV);

        // Calcula histogramas
        Mat histImage1 = new Mat();
        Mat histImage2 = new Mat();
        Imgproc.calcHist(java.util.Arrays.asList(hsvImage1), new
MatOfInt(0), new Mat(), histImage1, new MatOfInt(50), new MatOfFloat(0,
256));
        Imgproc.calcHist(java.util.Arrays.asList(hsvImage2), new
MatOfInt(0), new Mat(), histImage2, new MatOfInt(50), new MatOfFloat(0,
256));

        // Normaliza os histogramas
        Core.normalize(histImage1, histImage1, 0, 1, Core.NORM_MINMAX, -1,
new Mat());
        Core.normalize(histImage2, histImage2, 0, 1, Core.NORM_MINMAX, -1,
new Mat());

        // Compara os histogramas usando a métrica de correlação
        double similarity = Imgproc.compareHist(histImage1, histImage2,
Imgproc.CV_COMP_CORREL);
        System.out.println("Similaridade entre as imagens (Correlação de
Histogramas): " + similarity);
```

```
        return new RetornoComparacaoModel(input,parameter,similarity);
    }
}
```

fonte: elaborado pelo autor, 2026

A classe “CalculateMediaActivity” tem a responsabilidade de comparar os valores entregues pela classe de histograma e por meio dessa comparação é possível determinar a similaridade das imagens pelos valores médios apresentados. Em uma escala de cores, na metalografia é possível caracterizar o tipo de amostra pela similaridade das amostras, a olho nu como normalmente é feito o observador compara imagens observando as formas e tamanhos que as cores das imagens mostram. Em modelos matemáticos essa análise necessita ser feito por meio de uma métrica, nesse caso os valores dos histogramas.

```
ackage domain.comparison;

import domain.relatorios.model.RetornoComparacaoModel;
import domain.relatorios.model.ValorMedioImagensModel;

import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;

public class CalculateMediaActivity {

    public List<ValorMedioImagensModel> calcularMediaImagens(List<String>
inputList, List<RetornoComparacaoModel> retornoComparacaoModelList) {
        List<ValorMedioImagensModel> valorMedioImagensList = new
ArrayList<>();

        inputList.forEach(input-> {
            ValorMedioImagensModel valorMedioImagensModel =
getRetornoComparacaoModels(input,retornoComparacaoModelList);
            valorMedioImagensList.add(valorMedioImagensModel);
        });

        return valorMedioImagensList;
    }

    private static ValorMedioImagensModel getRetornoComparacaoModels(String
input, List<RetornoComparacaoModel> retornoComparacaoModelList) {
        var valoresAgrupados = retornoComparacaoModelList.stream().filter(
                retornoComparacaoModel ->
retornoComparacaoModel.getInput().equals(input)).collect(Collectors.toList(
));
        var valor =  somarValores(valoresAgrupados);
        return new ValorMedioImagensModel(input,valor);
    }

    private static Double somarValores(List<RetornoComparacaoModel>
valoresAgrupados) {
        double value = 0;
        for (RetornoComparacaoModel valor : valoresAgrupados) {
```

```
            value += valor.getComparison();
        }
        return (value / valoresAgrupados.size()) * 100;
    }


}
```

fonte: elaborado pelo autor, 2026

Classe para copiar as imagens de resultado.

```
package domain.comparison;

import domain.relatorios.model.RetornoComparacaoModel;
import domain.relatorios.model.ValorMedioImagensModel;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
import static core.AppConstants.calculateConstants.*;
import static core.AppConstants.fileConstants.*;

public class SelectOutputImage {

    GetImageActivity getImageActivity = new GetImageActivity();

    public List<String> execute(List<ValorMedioImagensModel>
valorMedioImagensModelList, List<RetornoComparacaoModel>
retornoComparacaoModelList) {
        List<String> imagensOutputList = new ArrayList<>();

        valorMedioImagensModelList.forEach(input -> {
            if (input.getValor() >= VALOR_MEDIO_MINIMO) {
                imagensOutputList.add(input.getImagem());
            }
        });

        retornoComparacaoModelList.forEach(input -> {
            if(input.getComparison() >= VALOR_MINIMO_COMPARACAO){
                imagensOutputList.add(input.getInput());
            }
        });

        var listaSemDuplicadas = removerDuplicados(imagensOutputList);
        listaSemDuplicadas.forEach(
                output ->
GetImageActivity.copiarImagem(CAMINHO_ENTRADA.concat(output),CAMINHO_OUTPUT
.concat(output))
        );

        return listaSemDuplicadas;
    }

    private List<String> removerDuplicados(List<String> lista) {
        Set<String> set = new HashSet<>(lista);
        return new ArrayList<>(set);
    }
}
```

fonte: elaborado pelo autor, 2026

A classe “CsvExporter” é responsável por gerar relatórios com o resultado dos números comparados, o relatório consiste em três colunas *input* – *parameter* - *comparison* sendo a coluna *input* o campo que identifica qual imagem de *input* será submetida a análise de comparação com as outras imagens que são os parâmetros presentes no campo *parameter*, o resultado dessa comparação fica no campo *comparison*, esse valor em decimal, após convertido demonstra o percentual de semelhança entre os valores das imagens.

```
package domain.relatorios;

import domain.relatorios.model.RetornoComparacaoModel;
import domain.relatorios.model.ValorMedioImagensModel;

import java.io.FileWriter;
import java.io.IOException;
import java.util.List;

public class CsvExporter {

    public void exportToCsv(List<RetornoComparacaoModel>
retornoComparacaoList, String filePath) {
        try (FileWriter writer = new FileWriter(filePath)) {
            writer.append("input;parameter;comparison\n");

            for (RetornoComparacaoModel item : retornoComparacaoList) {
                writer.append(item.getInput()).append(";")
                      .append(item.getParameter()).append(";")

.append(String.format("%.2f",item.getComparison())).append("\n");
            }

            System.out.println("CSV file created: " + filePath);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void exportMediaToCsv(List<ValorMedioImagensModel>
valorMedioImagensModelList, String filePath) {
        try (FileWriter writer = new FileWriter(filePath)) {
            writer.append("imagem;valor\n");

            for (ValorMedioImagensModel item : valorMedioImagensModelList)
{
                writer.append(item.getImagem()).append(";")

.append(String.format("%.2f",item.getValor())).append("\n");
            }

            System.out.println("CSV file created: " + filePath);
        } catch (IOException e) {
            e.printStackTrace();
        }
```

```
    }
}
```

fonte: elaborado pelo autor, 2026

Classe *main* integração com I.A.

```
import com.openai.client.OpenAIClient;
import com.openai.client.okhttp.OpenAIOkHttpClient;
import com.openai.models.responses.Response;
import com.openai.models.responses.ResponseCreateParams;
import domain.ia.ChatGPTImageClient;

public class Main {
    public static void main(String[] args) {

        var API_KEY = "sk-proj-AkZJXZStQuTYMvBBrrDr-
pQyIjCo6ZBVotPjAjKO8kWAZ2KbJtE5IW5-**";
        try {
            ChatGPTImageClient client = new ChatGPTImageClient(API_KEY);

            String resposta = client.sendImage(

"C:\\Users\\cicer\\Desktop\\imagens\\input\\imagen1.jpg",
                        "Existe alguma trinca na imagem?"
            );

            System.out.println(resposta);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

            OpenAIClient client = OpenAIOkHttpClient.fromEnv();

            ResponseCreateParams params = ResponseCreateParams.builder()
                    .input("Say this is a test")
                    .model("gpt-5-nano")
                    .build();

            Response response = client.responses().create(params);
            System.out.println(response.output());
        }


}
```

fonte: elaborado pelo autor, 2026

Service chamada *open I.A*.

```
package domain.ia;

import java.net.URI;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;
import java.nio.file.Files;
import java.nio.file.Path;
import java.util.Base64;
```

```
import org.json.JSONArray;
import org.json.JSONObject;

public class ChatGPTImageClient {

    private final String apiKey;
    private final HttpClient httpClient;

    public ChatGPTImageClient(String apiKey) {
        this.apiKey = apiKey;
        this.httpClient = HttpClient.newHttpClient();
    }

    /**
     * Envia uma imagem ao GPT e retorna a resposta.
     */
    public String sendImage(String imagePath, String prompt) throws
Exception {

        // 1) Ler imagem
        byte[] imageBytes = Files.readAllBytes(Path.of(imagePath));
        String base64Image =
Base64.getEncoder().encodeToString(imageBytes);

        // 2) Construir JSON do conteúdo
        JSONObject contentText = new JSONObject()
                .put("type", "text")
                .put("text", prompt);

        JSONObject contentImage = new JSONObject()
                .put("type", "image_url")
                .put("image_url", new JSONObject()
                        .put("url", "" +
base64Image));

        JSONArray contentArray = new JSONArray()
                .put(contentText)
                .put(contentImage);

        JSONObject message = new JSONObject()
                .put("role", "user")
                .put("content", contentArray);

        JSONObject jsonBody = new JSONObject()
                .put("model", "gpt-4o")
                .put("messages", new JSONArray().put(message));

        // 3) Montar requisição HTTP
        HttpRequest request = HttpRequest.newBuilder()

.uri(URI.create("https://api.openai.com/v1/chat/completions"))
                .header("Authorization", "Bearer " + apiKey)
                .header("Content-Type", "application/json")

.POST(HttpRequest.BodyPublishers.ofString(jsonBody.toString()))
                .build();

        // 4) Enviar requisição
        HttpResponse<String> response =
                httpClient.send(request,
```

```
HttpResponse.BodyHandlers.ofString());

        if (response.statusCode() >= 300) {
            throw new RuntimeException("Erro da API: " +
response.statusCode()
                    + " - " + response.body());
        }

        // 5) Extrair resposta
        JSONObject json = new JSONObject(response.body());

        return json.getJSONArray("choices")
                .getJSONObject(0)
                .getJSONObject("message")
                .getString("content");
    }

}
```

fonte: elaborado pelo autor, 2026