UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CRIAÇÃO DE FILTROS DE IMAGENS ATRAVÉS DA UTILIZAÇÃO DE PROGRAMAÇÃO GENÉTICA

LUCAS PAULI SIMÕES

ORIENTADOR: PROF. DR. EDILSON REIS RODRIGUES KATO

São Carlos - SP
Maio/2013

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CRIAÇÃO DE FILTROS DE IMAGENS ATRAVÉS DA UTILIZAÇÃO DE PROGRAMAÇÃO GENÉTICA

LUCAS PAULI SIMÕES

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Inteligência Artificial.
Orientador: Dr. Edilson Reis Rodrigues Kato.

São Carlos - SP
Maio/2013

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

S593cf

Simões, Lucas Pauli.
Criação de filtros de imagens através da utilização de programação genética / Lucas Pauli Simões. -- São Carlos : UFSCar, 2013.
116 f.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2013.

1. Inteligência artificial. 2. Programação genética (Computação). 3. Visão computacional. I. Título.

CDD: 006.3 (20ª)

Universidade Federal de São Carlos

Centro de Ciências Exatas e de Tecnologia

Programa de Pós-Graduação em Ciência da Computação

"Criação de Filtros de Imagens Através da Utilização de Programação Genética"

Lucas Pauli Simões

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação

Membros da Banca:

Prof. Dr. Edilson Reis Rodrigues Kato
(Orientador - DC/UFSCar)

Prof. Dr. Emerson Carlos Pedrino
(DC/UFSCar)

Prof. Dr. Roberto Hideaki Tsunaki
(EESC/USP)

Profa. Dra. Veronica Oliveira de Carvalho
(IGCE/UNESP/Rio Claro)

São Carlos
Junho/2013

Dedico esse trabalho aos meus pais Angela e Antonio,

à minha irmã Maíra, e a toda família

pelo apoio e incentivo ao longo dessa etapa.

AGRADECIMENTO

Dedico meus sinceros agradecimentos:

- À Andrea, pelo apoio, companheirismo, compreensão e incentivo.

- Aos amigos Me. Ricardo Ferreira Martins e Me. Alex Luiz de Sousa, pelo incentivo e ajuda para o ingresso nessa jornada.

- Ao meu orientador Dr. Edilson Kato, pelo tempo dedicado a reuniões e discussões fundamentais que levaram ao sucesso do trabalho.

- À banca de qualificação, Dr. Roberto; Dr. Emerson e Dr. Orides, pelas sugestões que ajudaram a ajustar o foco do trabalho.

- Ao Dr. Emerson pelas reuniões e orientações na área de processamento de imagens.

- Aos meus colegas do TEAR, pelo companheirismo e trocas de conhecimento.

- Aos professores do DC, pelo conhecimento passado e amizade adquirida.

- À banca avaliadora, Dr. Roberto; Dr. Emerson e Dra. Veronica, pelo interesse, observações e questionamentos que ajudaram a melhorar a qualidade do trabalho.

- Ao CNPQ, por possibilitar a dedicação integral a este trabalho.

"Veni, Vidi, Vici."

Gaius Julius Caesar

RESUMO

As técnicas de visão computacional têm sido utilizadas cada vez mais pelas indústrias para o auxílio da automação de seus processos, porém, a implementação das técnicas de visão computacional encontra diversas dificuldades de acordo com a aplicação. Tais técnicas são limitadas ao custo computacional de acordo com sua complexidade. Problemas de identificação de elementos em cenários industriais são exemplos de uma aplicação que pode gerar uma maior complexidade na automação do processo. Pesquisas são realizadas utilizando diversas técnicas para a solução dessas dificuldades. Este trabalho aborda o campo de visão computacional através da utilização de técnicas de inteligência artificial. Aqui são avaliados métodos para criação e replicação de filtros de imagens binárias através da utilização da programação genética com o objetivo de identificação de elementos em um cenário industrial. São avaliados dois métodos que possuem abordagens diferentes, um utiliza operações entre pixels e outra morfologia matemática para a detecção de objetos. Os resultados são apresentados tanto qualitativamente quanto quantitativamente através de imagens comparativas dos métodos avaliados e das medidas estatísticas, respectivamente. Através da utilização de toolbox GPLab em conjunto com o Matlab, foi possível a criação de filtros de forma automática para identificação de objetos, de forma que a detecção e remoção de elementos em imagens possa ser utilizada por outros sistemas de apoio ao funcionamento automático em um ambiente industrial. Os resultados estabeleceram uma forma de criação de filtros eficiente a partir da utilização da programação genética.

PALAVRAS-CHAVE: VISÃO COMPUTACIONAL, INTELIGÊNCIA ARTIFICIAL, PROGRAMAÇÃO GENÉTICA.

ABSTRACT

The techniques of computer vision have been used more and more by the industries in order to aid the automation of their processes; however, the implementation of computer vision techniques has several difficulties according with the application. Such techniques are limited to the computational cost along with its complexity. Problems of elements identification in industrial sceneries are examples of an application that can generate a larger complexity in the process automation. Researches are accomplished using several techniques for solving those difficulties. This work approaches the field of computer vision through the use of artificial intelligence techniques. Here, methods are evaluated for creation and replication of binary images filters through the use of the genetic programming with the objective of elements identification in an industrial scenery. Two methods that possess different approaches are evaluated; one uses operations between pixels and other mathematical morphology for objects detection. The results are presented qualitatively as well as quantitatively through comparative images of the evaluated methods and statistical measures, respectively. Through the GPLab toolbox together with Matlab, the automatic creation of filters for objects identification was possible, so the detection and deletion of elements in images can be used by other support systems to automatic operation in an industrial environment. The results established a efficient way of filter creation with the use of the genetic programming.

KEYWORDS: COMPUTER VISION, ARTIFICIAL INTELLIGENCE, GENETIC PROGRAMMING.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

Capítulo 1

INTRODUÇÃO À VISÃO COMPUTACIONAL

De acordo com Ogata (1998), o primeiro trabalho significativo para o controle automático se passa no séc. XVIII, quando James Watt construiu um controlador de velocidade de uma máquina a vapor. Outro fator importante ocorre a partir de 1960, onde a disponibilidade dos computadores digitais passa a tornar possível a análise de sistemas de controle complexos, dando início à teoria de controle moderna.

As técnicas de visão computacional têm sido utilizadas cada vez mais pelas indústrias para o auxílio da automação de seus processos, tais técnicas são necessárias devido ao nível de complexidade dos problemas a tratar.

Ogata (1998) ilustra as necessidades de robôs de alto nível de complexidade onde é necessária a utilização de um meio óptico (como um sistema de televisão) que é utilizado para realizar uma varredura do ambiente de fundo onde se encontra o objeto, com essa varredura ele reconhece padrões e determina a presença e orientação do objeto.

De acordo com Chin e Harlow (1982), muitas das pesquisas industriais tinham como foco a inspeção visual automática de suas linhas de produção. Os maiores pontos de argumentação eram que a inspeção visual automática livraria os humanos de rotinas repetitivas, pouparia recursos humanos, possibilitaria a inspeção em ambientes desfavoráveis, proporcionaria a redução na demanda de inspetores especialistas humanos, além de possibilitar uma melhor análise estatística assim como uma maior rapidez de inspeção capaz de acompanhar a velocidade da produção.

Para executar tal tarefa, nas pesquisas iniciais, foram utilizadas diversas técnicas de abordagem sendo as de maior destaque: operadores lógicos, morfologia matemática e gabaritos ou máscaras de comparação (CHIN; HARLOW, 1982).

Os processos, de forma geral, encontram diversas dificuldades quando são feitas suas automações, uma delas é a validação constante dos resultados obtidos em uma pequena parte do processo. No trabalho de Besari et al. (2010) é tratado o problema de polimento de superfícies, e a principal dificuldade encontrada nesse processo é que a superfície deve ser analisada diversas vezes durante o processo de polimento para se atingir um resultado satisfatório.

Isso se dá ao fato de que conforme a peça é polida as imperfeições são corrigidas, porém é necessário que o processo seja feito em pequenas etapas para que não ocorra um desgaste excessivo no ponto da peça que está sendo polido. Para tratar tal problema, Besari et al. (2010) propõem a utilização de redes neurais que possam analisar a superfície através da visão computacional, e assim orientar os próximos passos do processo como, por exemplo, aumentar a pressão e diminuir o tempo de polimento naquele local.

Em outra aplicação, como para a automação de máquinas na agricultura, um dos desafios encontra-se na locomoção, onde a distinção entre o caminho a seguir e a área de plantio é crucial para possibilitar esse tipo de automação. Tal problema é tratado por Lulio et al. (2009), através da utilização da técnica de segmentação de imagens JSEG[1] em conjunto com redes neurais para reconhecimento de padrões.

Na área de visão computacional, muitas questões podem ser discutidas, além das dificuldades que envolvem percepção do ambiente e avaliação dos resultados, é necessário também levar em consideração o ambiente onde será operado o sistema.

Duas dificuldades em relação ao ambiente são levantadas por Killing, Surgenor e Mechefske (2009), que apresentam uma proposta para a inspeção de peças, onde a qualidade do resultado de sua proposta está diretamente relacionada ao posicionamento correto da câmera responsável pela alimentação do sistema, como também pela iluminação do ambiente no momento da inspeção.

Além destas dificuldades muitas outras podem ser analisadas, e como a área de visão computacional ainda se encontra em uma fase inicial é possível identificar muitas abordagens já avaliadas, como também outras ainda a serem testadas.

[1] Método de segmentação de imagens coloridas proposto por Deng, Manjunath e Shin (1999).

A este projeto de mestrado aborda o estado da arte do campo da visão computacional, focando principalmente na criação de filtros de imagens. A abordagem utiliza técnicas de programação evolutiva para gerar filtros de imagens, que quando aplicados, resultem no destaque ou remoção de objetos.

Neste projeto serão comparadas duas técnicas e para realizar a comparação, também é feita uma análise de ferramentas disponíveis para a utilização com programação genética. Por fim, também é feita uma análise que visa identificar a qualidade dos resultados obtidos com uma análise secundária dos dados obtidos durante o processo de comparação das técnicas quando aplicadas à geração de filtros de imagens.

1.1 Motivação

A aplicação da visão computacional está cada vez mais presente nas indústrias. Ambientes industriais estão fortemente ligados à necessidade de soluções que exigem respostas em um curto espaço de tempo, para que assim seja possível manter um ritmo próximo ou igual ao já existente no processo em que será empregada a visão computacional.

Para que isso seja possível, ou a solução proposta deve ser simples de ser executada, ou será necessário um grande poder computacional para executá-la. Como o poder computacional mais elevado trás custos, isso gera uma busca por um equilíbrio entre simplicidade de execução e poder computacional.

Ao longo do levantamento bibliográfico sobre o estado da arte das áreas de pesquisa, foi identificada uma grande variedade de técnicas que podem ser aplicadas à visão computacional, sendo que cada técnica apresenta pontos fortes e fracos, cabendo ao pesquisador decidir qual delas melhor se enquadra ao seu problema. Isso implica em analisar técnicas de pré-processamento, processamento morfológico e detecção/segmentação dependendo do que se deseja obter.

Tal análise pode apresentar um nível de complexidade elevado, uma vez que o processamento morfológico gera impacto na detecção/segmentação e o pré-processamento gera impacto no processamento morfológico. Ou seja, se uma

técnica de detecção/segmentação não apresenta bons resultados, o baixo desempenho pode ser consequência de falhas em etapas anteriores.

Outro ponto importante a ser considerado é que cada etapa também pode possuir mais de um elemento. O processamento morfológico pode ser composto, por exemplo, por uma série de operações morfológicas para atingir um determinado resultado, e uma leve variação nessa série de operações pode levar a um resultado melhor que o já obtido.

O mesmo princípio pode ser aplicado a cada etapa de processamento individualmente, como citado no exemplo do processamento morfológico. E se caso cada etapa for analisada individualmente esse espaço de busca cresce ainda mais. Ou seja, mesmo que o estudo tenha foco em apenas uma etapa do processamento de imagens, o espaço de busca pode apresentar-se extenso.

Devido a essa característica, a programação genética apresenta características que indicam ser adequadas ao problema. Dado que a programação genética é capaz de explorar e avaliar um grande espaço de busca, em um tempo inferior ao que seria utilizado para uma varredura de todo espaço de busca através de uma busca exaustiva.

1.2 Objetivo

O objetivo desse trabalho é o comparar duas técnicas de visão computacional que apresentam abordagens distintas a uma mesma classe de problema. Esses métodos podem realizar de forma automática a criação de filtros de imagens binárias, para detecção ou remoção de elementos em imagens, para que outros sistemas que utilizem a visão computacional possam apoiar o funcionamento automático da produção.

Dessa forma, o objetivo deste trabalho está em criar filtros de imagens a partir de resultados pré-determinados (exemplos de treinamento) com auxílio da programação genética. Sendo necessário para a execução do processo, o estabelecimento de um conjunto de operações que teoricamente podem ser utilizados na imagem original para conseguir atingir o resultado esperado (fornecido

pelos exemplos de treinamento), assim como indicar como esse conjunto de operações deve ser aplicado.

O conjunto de operações estabelecido possui operadores lógicos e morfológicos, tais como *and*, *or*, erosão, etc. A maneira de como esses operadores serão sequenciados em uma imagem será estabelecida utilizando técnicas evolucionárias de inteligência artificial, mais especificamente a programação genética. Tais técnicas são aplicadas de acordo com as propostas que são o foco da comparação.

Com o conjunto de operações que podem ser utilizadas e sua forma de aplicação definidos, a programação genética deve ser capaz de encontrar uma combinação de operações dentro desses conjuntos, que quando aplicadas a qualquer imagem pertencente à mesma classe da qual o filtro foi gerado, o resultado seja próximo ou igual ao esperado.

Os testes são embasados em métodos encontrados na literatura, com o propósito de identificar um que seja capaz de produzir resultados que possam atender, com boa qualidade, a demanda desse tipo de problema. A qualidade deve ser mensurável através de métricas quantitativas, fornecendo assim uma forma confiável de avaliação e comparação.

1.3 Organização do Trabalho

A estrutura deste trabalho está dividida em cinco capítulos, no Capítulo 2 está a visão geral do contexto no qual este trabalho está inserido. Nele são discutidas questões sobre sistemas automáticos que utilizam visão computacional, trabalhos envolvendo processamento de imagens, noções teóricas dos algoritmos evolutivos, aplicações envolvendo inteligência artificial no contexto de processamento de imagens e por fim é feito um levantamento das ferramentas mais utilizadas neste contexto.

O Capítulo 3 aborda a proposta e os objetivos deste trabalho, assim como explica qual metodologia foi aplicada para alcançar e avaliar o que foi proposto para o projeto e como isso auxiliou a atingir o objetivo.

A implementação e os resultados estão dispostos no Capítulo 4, onde está descrita de forma detalhada a maneira que os resultados são apresentados, como também a forma de analisá-los. Ainda nesse capítulo, é apresentada uma segunda forma de avaliação dos resultados utilizando métricas estatísticas.

Por fim, no capítulo 5 é feita a comparação dos resultados e dos objetivos, analisando o que foi obtido, assim como se os resultados foram bem sucedidos. Também são feitas observações sobre o que pode ser explorado em trabalhos futuros.

Capítulo 2

REVISÃO BIBLIOGRÁFICA

Neste capítulo são introduzidos os sistemas automáticos utilizando visão computacional como também são apresentadas pesquisas recentes da área.

Em seguida são apresentados alguns conceitos de processamento de imagens, assim como aplicações que utilizam o processamento de imagens para tratar de determinados problemas.

Após a parte de processamento de imagens é introduzida a teoria dos algoritmos evolutivos, para que então sejam apresentadas aplicações que utilizem inteligência artificial (IA) em problemas da área de processamento de Imagens.

Por fim, é apresentado um levantamento bibliográfico de aplicações que utilizam algoritmos evolutivos focados mais especificamente na utilização de programação genética, para assim avaliar quais são as ferramentas voltadas para programação genética que são mais utilizadas nessa área de pesquisa.

2.1 Sistemas Automáticos utilizando Visão Computacional

No início dos anos 90 novas pesquisas relacionadas ao processamento de imagens começam a surgir com o auxílio da inteligência artificial, como a de Terano et al. (1988), que utiliza um método de decisão não-Bayesiano para monitoramento preventivo de portões de represas hidroelétricas visando prever sua vida útil. Neste mesmo período surgem pesquisas que unem inteligência artificial e visão computacional como, por exemplo, a pesquisa de Roth (1990), que utiliza redes

neurais para análise de imagens visando aplicação militar na identificação de alvos de caças aéreos.

As indústrias tem se envolvido com os mais diversos segmentos, tratando problemas antes intratáveis sem a utilização de técnicas de inteligência artificial. Suas abordagens variam desde a utilização dos métodos clássicos como algoritmos genéticos, lógica proposicional e lógica fuzzy, assim como métodos mistos ou variações como neuro-fuzzy. Além da utilização de tais técnicas, são encontradas abordagens com utilização de FPGAs (*Field-programmable gate array*) para tratar problemas simples e complexos diretamente em nível de hardware.

Gordan et al. (2008), apresenta um sistema que auxilia no diagnóstico da deterioração de barragens hidráulicas, com uma abordagem diferente da aplicada por Terano et al. (1988), sua proposta se dá por meio da aplicação de técnicas fuzzy que classificam o nível de deterioração dos setores da barragem. Também com foco na inspeção, Jayakumar e Kanna (2010) sugerem a utilização de redes neurais para auxiliar inspeção de transistores visando detecção de defeitos possibilitando descarte ou correções.

El-Medany e Hussain (2007) sugerem que o controle de semáforos de trânsito pode ser controlado através da utilização de um FPGA sendo que o método proposto foi capaz de atuar em quatro ruas com um total de seis semáforos, o que indica que FPGAs podem ser utilizados em problemas complexos.

Quanto ao tempo de resposta, que é um ponto bastante analisado na utilização de FPGAs, outro exemplo é o apresentado por Birla (2006), onde é utilizado um FPGA para análise sequencial de imagens, onde são obtidas taxas de 50 quadros por segundo sendo que pesquisas anteriores citadas por Birla (2006) alcançavam taxas de apenas 3 quadros por segundo, o que demonstra que dependendo da aplicação FPGAs podem ser bem eficazes, sendo que por terem sua implementação em hardware sua performance tende a ser próxima ao tempo real.

2.2 Processamento de Imagens

Segundo Gonzalez e Woods (2002), o interesse nos métodos de processamento de imagens provém de duas principais áreas de aplicação: melhoria de informação visual para interpretação humana; e processamento de dados de imagens para armazenamento, transmissão, e representação para percepção de máquinas autônomas.

Gonzalez e Woods (2002), afirmam que os métodos de processamento de imagens podem ser divididos em duas classes: os métodos que têm imagens como entrada e como saída; e os métodos que podem ter como entrada imagens, mas terão como saída atributos das imagens de entrada.

Para Gonzalez e Woods (2002), o processamento de imagens pode ser subdividido em 10 processos que podem ser combinados entre si, ou aplicado individualmente, sendo eles: aquisição de imagem; destaque de imagem; restauração de imagem; processamento de cor; transformada *wavelet* e processamento multiresolução; compressão; processamento morfológico; segmentação; representação/descrição e reconhecimento.

Entre os métodos clássicos de destaque e restauração temos como exemplos os filtros espaciais de suavização, que são comumente utilizados para borrar imagens e remover ruídos; filtros espaciais de realce para realçar detalhes da imagem ou detalhes que tenham sido borrados; media de imagem usada para redução de ruídos. Para ilustrar algumas dessas técnicas, foram aplicadas suas formas clássicas em um trecho de uma foto resultando na imagem da Figura 1.

Representados nessa figura temos: (a) Imagem Original; (b) Suavização de (a); (c) Realce de (a); (d) Redução de ruído em (a); (e) Transformação de Cores em (a); (f) (a) em escala cinza.

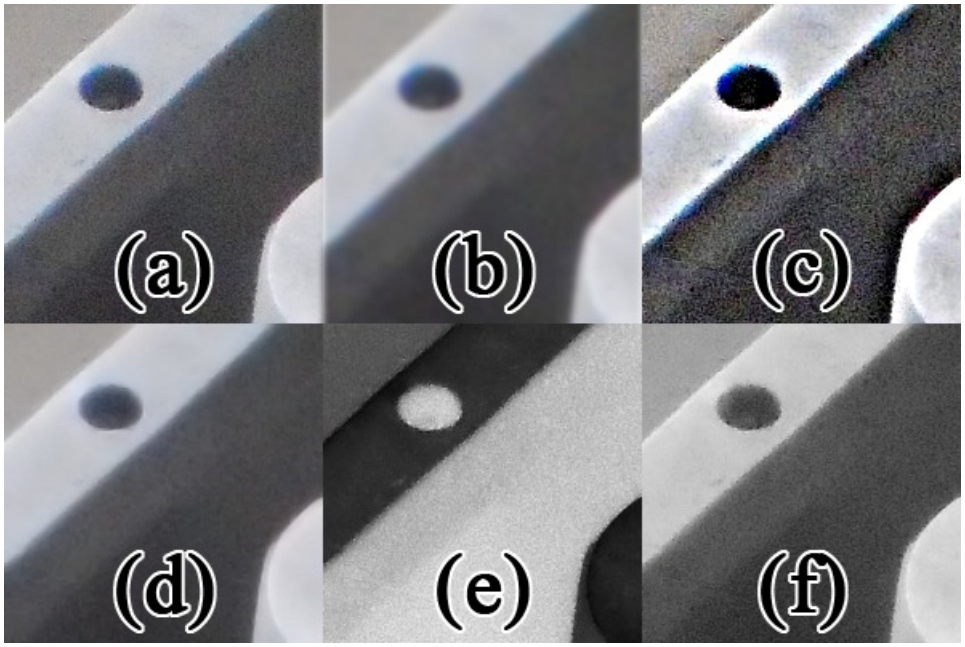


Figura 1 – Ilustração de métodos de processamento de imagens. Elaborada pelo autor.

Outra técnica utilizada no processamento de imagens é a morfologia matemática ou processamento morfológico. Com o processamento morfológico, algumas simplificações ou realces podem ser feitos na imagem assim como efetuar detecção de características.

Para atingir tais objetivos, é necessário aplicar as operações morfológicas, que são baseadas na teoria dos conjuntos, onde cada operação pode ser combinada a outra ou ser utilizada isoladamente para gerar um resultado.

Na Figura 2 estão ilustradas algumas operações morfológicas aplicadas individualmente em uma mesma imagem convertida para formato binário, devido ao fato de que a morfologia matemática foi inicialmente criada para tratar problemas que envolviam apenas imagens binárias, a utilização desse tipo de imagens é mais adequada para ilustrar o conceito.

Nessa figura foi obtido como resultado: (a) a conversão da imagem original para o formato binário, (b) a operação de abertura, (c) fechamento, (d) dilatação, (e) erosão e (f) dilatação seguida de erosão (*Hit and Miss*).

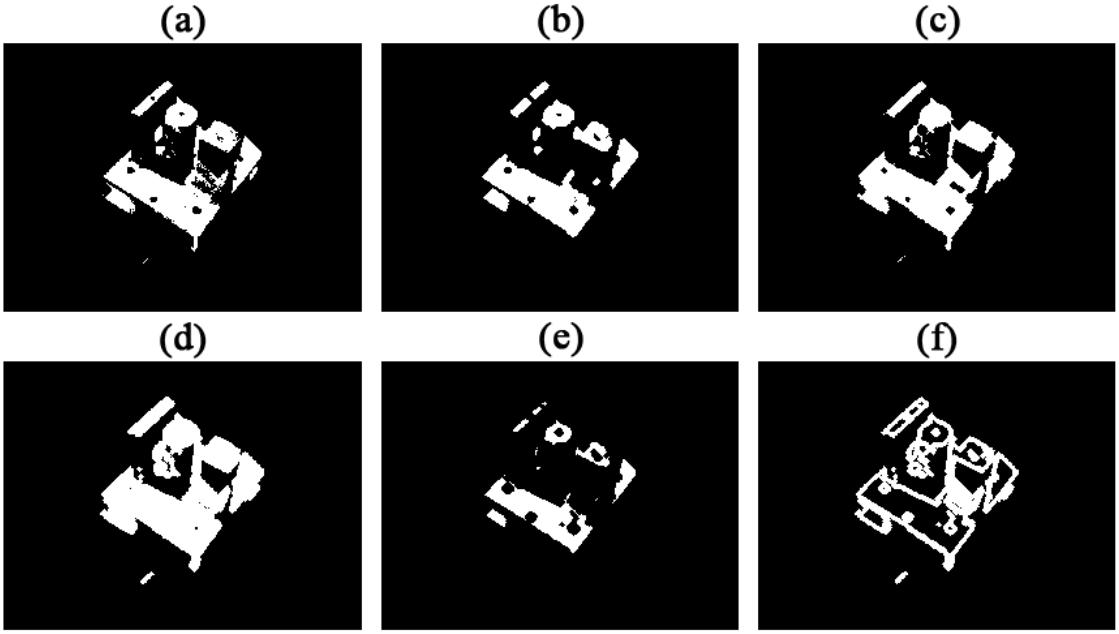


Figura 2 – Ilustração de operações morfológicas. Elaborada pelo autor.

Na segmentação/detecção de imagens, a multiplicação por matrizes é o método mais comum, onde dependendo da matriz aplicada se obtém uma determinada característica realçada e essa técnica é muito utilizada para detecção de pontos, linhas, arestas e vértices.

Ao aplicar tais técnicas isoladamente ou em conjunto o resultado esperado é uma imagem onde as características de maior importância, de acordo com o que está sendo almejado, estejam destacadas quando comparadas ao restante da imagem.

2.3 Aplicações de processamento de imagens

Bhattacharya, Majumder e Sanyal (2010) propõem um método cujo foco é auxiliar na suavização de imagens com o intuito de facilitar a detecção de características. Seu método, batizado Filtro da Máxima Gaussiana, foi comparado com os filtros bilateral e de deslocamento de média apresentando imagens resultantes com deformações mais sutis do que as dos outros dois métodos, como pode ser observado na Figura 3.

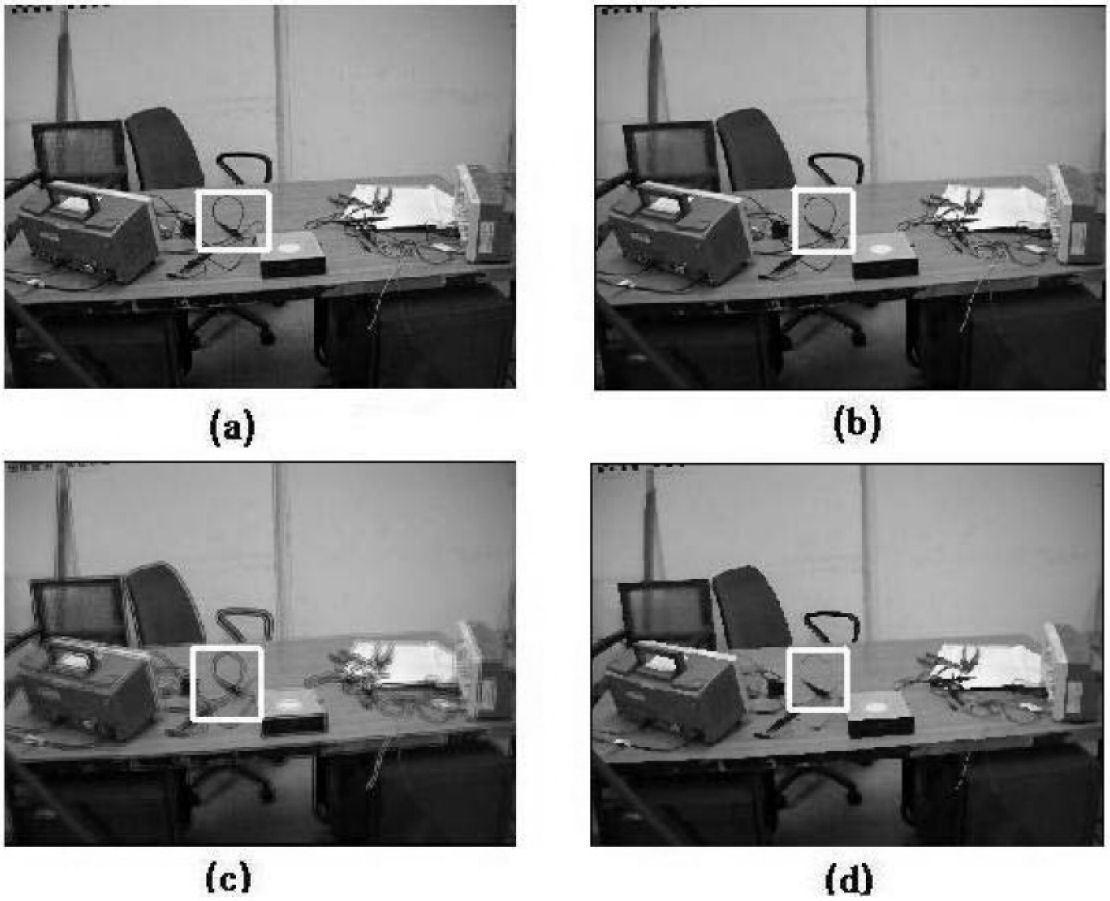


Figura 3 – Exemplo de suavização através dos resultados obtidos por Bhattacharya, Majumder e Sanyal, adaptada de (BHATTACHARYA; MAJUMDER; SANYAL, 2010).

Nessa estão representadas: (a) a imagem original; (b) a imagem original filtrada com o filtro da máxima gaussiana; (c) e (d) a imagem original filtrada respectivamente com o filtro bilateral e deslocamento de média. (BHATTACHARYA; MAJUMDER; SANYAL, 2010).

Como a suavização deste método é mais sutil, dependendo da aplicação, isso pode ser um fator interessante no caso de ruídos pouco expressivos ou imagens de baixa qualidade. Entretanto, nos dois casos é interessante que a suavização não leve a muita perda de informações na imagem alvo, o que poderia comprometer um tratamento ou análise porterior.

Ao trabalhar com imagens coloridas, o espaço de cores[2] deve ser considerado. Wirth e Nikitenko (2010) analisam quais os efeitos desses espaços de cores na aplicação de métodos de realce visando identificar qual o comportamento de cada espaço e para qual aplicação ele seria mais bem empregado.

[2] Indica qual a composição de cores deve ser utilizada como: RGB, YIQ, CIELab entre outros.

Os métodos utilizados por Wirth e Nikitenko (2010) para validação foram a máscara de remoção de realce e realce morfológico fuzzy, e ambos os métodos foram aplicados nos espaços de cores: RGB, YIQ e CIELab. Nos resultados, ao analisar as variações dos métodos aplicados a cada espaço de cores, o método da máscara de remoção de realce apresentou o mesmo resultado indiferente do espaço de cor, já o método de realce morfológico fuzzy apresentou resultados diferentes dependendo do espaço de cores no qual foi aplicado.

Neste trabalho serão tratadas apenas imagens binárias porém, dependendo do método aplicado, podem ocorrer variações de acordo com o espaço de cores. Com essa informação, conclui-se que o espaço de cor deve ser considerado ao longo da aplicação e validação de métodos que utilizem imagens coloridas.

Abordando outro aspecto do pré-processamento, Pompapathi, Krishna e Babu (2010) propuseram um método de remoção de ruídos com preservação de bordas, constituído de duas etapas. A primeira etapa deste método consiste em detectar ruídos gerados durante o processo de aquisição de imagens, para servir de parâmetro para o filtro com preservação de bordas da segunda etapa. Resultados apresentados na Figura 4.

Figura 4 – Exemplo de remoção de ruído através dos resultados obtidos por Pompapathi, Krishna e Babu, adaptada de (POMPAPATHI; KRISHNA; BABU, 2010).

A Figura 4 é composta por: (a) Imagem original; (b) imagem com ruído adicionado artificialmente; imagens tratadas com (c) filtro SMF, (d) filtro DBA, (e) filtro de preservação de bordas e (f) filtro proposto.

Quando comparado com outras técnicas de redução de ruído como SMF (S*imple and efficient soft Morphological Filter*), DBA (*Decision-Based Algorithm*) e preservação de bordas, os autores indicam que o método proposto apresentou um melhor desempenho que os demais métodos. Porém, as imagens apresentadas na Figura 4 para comparação, quando tratadas com os métodos de controle e o método proposto, apresentavam algumas variações, mas apenas o método SMF apresentou um resultado discrepante em relação aos demais, o que dificulta a inspeção visual de resultados.

Apesar de a inspeção visual não ser possível de quantificar, na Figura 4 são apresentados valores referentes ao PSNR (*Peak Signal-to-Noise Ratio*) das imagens resultantes, e mesmo com uma variação pequena nesses valores, seu impacto nas etapas posteriores de processamento de imagens pode causar uma grande influência, logo tais técnicas devem ser observadas e consideradas.

Durante a comparação dos métodos neste trabalho as imagens utilizadas para os testes foram geradas artificialmente, entretanto caso fossem utilizadas imagens de uma linha de produção ou de outro ambiente, provavelmente a aplicação de técnicas de remoção de ruídos se fariam necessárias.

Já as técnicas de pré-processamento são comumente empregadas para auxiliar a detecção de objetos em imagens como proposto por Ott e Everingham (2009). Eles propõem um método baseado em histogramas gradientes para detecção de pedestres, que faz uma leve segmentação da imagem em plano de fundo e objetos. Porém durante os experimentos, foi identificado que o comportamento do método era mais satisfatório em imagens que eram tratadas com normalização de cores, sendo possível perceber mais nitidamente o contorno do pedestre na Figura 5d do que na Figura 5b.

A figura tem disposto da esquerda para direita: (a) imagem original; (b) magnitude gradiente da imagem original; (c) imagem com normalização de cores; (d) magnitude gradiente da imagem normalizada. Adaptada de (OTT; EVERINGHAM, 2009)

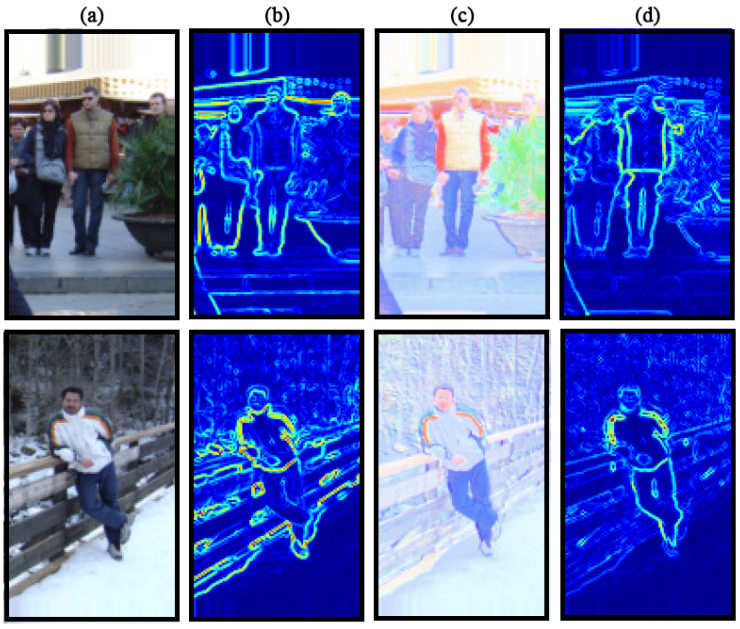


Figura 5 – Exemplo de segmentação através dos resultados obtidos por Ott e Everingham, adaptada de (OTT; EVERINGHAM, 2009).

A segmentação ocorre em imagens coloridas com uma busca por segmentações suaves implícitas na imagem. Depois de encontradas elas são inseridas em um classificador SVM (*Support Vector Machine*) que por fim identifica os pedestres na imagem.

Os resultados apresentados por Ott e Everingham (2009) reforçam a discussão levantada sobre a influência das etapas anteriores nas etapas posteriores, que no caso deles, o pré-processamento com normalização de cores, trouxe um ganho significativo ao resultado final do método.

Outro exemplo da influência do pré-processamento onde é abordada a conversão de cores para escala cinza é o de Lu e Plataniotis (2009), onde a conversão é feita de modo a gerar uma saída que auxilie na detecção facial. Afirmando que a conversão NTSC não é ideal para reconhecimento de faces, após pesquisarem padrões de cores de pixels da pele humana identificaram uma tendência ao tom vermelho. Sendo assim é proposta uma variação da conversão RGB que considere este fator dos tons de vermelho, auxiliando assim na tarefa futura.

Portanto, antes de decidir quais são as técnicas que devem ser empregadas, as pesquisas indicam que uma pré-análise do problema se faz necessária para que os métodos sejam direcionados à resolução do problema, exceto quando o problema não é conhecido ou é muito abrangente.

Essa argumentação é interessante, pois como neste trabalho a programação genética será a responsável pela busca das melhores combinações, os principais fatores para que a técnica seja bem sucedida são que as funções pertinentes ao espaço de busca sejam suficientes e adequadas; e que o *fitness* também esteja de acordo com o problema abordado.

Entretanto, vale ressaltar que é necessário cuidar para não sobrecarregar o algoritmo com todas as técnicas possíveis, o que faria com que o espaço de busca crescesse demasiadamente dificultando o sucesso da programação genética em encontrar uma boa combinação.

2.4 Teoria dos Algoritmos Evolutivos

A Computação Evolutiva (CE) é uma área de pesquisa que tem se expandido rapidamente. De acordo com Gabriel e Delbem (2008), os motivos para isso são: possuir algoritmos capazes de encontrar soluções válidas para problemas complexos através da utilização de métodos simples, chamados de algoritmos

evolutivos (AEs), que utilizam princípios básicos da Teoria da evolução e da Genética que podem ser modelados por poucas linhas de código e serem relativamente fáceis de adaptar para problemas das mais diversas áreas.

Segundo Gabriel e Delbem (2008), a interação de várias frentes de estudo produziu os AEs atuais, e dentre esses, os algoritmos genéticos (AGs) são os mais conhecidos devido à sua utilização em Inteligência Artificial (IA). Gabriel e Delbem (2008), afirmam que na década de 1930, os sistemas evolutivos naturais passaram a ser investigados como algoritmos de exploração de múltiplos picos de uma função objetivo, e nesse contexto, três abordagens de AEs foram desenvolvidas de forma independente: a programação evolutiva (PE), as estratégias evolutivas (EEs) e os algoritmos genéticos (AGs) que deram origem a novas variações como a programação genética (PG) e os Micro-AGs.

De acordo com Russell e Norvig (2010), os algoritmos genéticos são uma analogia à seleção natural. Ao analisar o espaço de busca de um determinado problema, cada solução para este é representada por um indivíduo e um grupo de indivíduos (soluções) define uma população. Cada indivíduo é definido por uma cadeia de caracteres definida com base em um alfabeto finito, sendo mais comumente constituída por 0s e 1s.

As gerações seguintes são criadas a partir de uma primeira gerada aleatoriamente. Os indivíduos da geração anterior são elencados utilizando uma função objetivo, ou função de *fitness*, para então dar início ao processo de reprodução genética a cada par de indivíduos (RUSSELL; NORVIG, 2010).

Segundo Russell e Norvig (2010), com os indivíduos elencados são atribuídas porcentagens a cada indivíduo de acordo com seu *fitness*, sendo que os indivíduos com maior *fitness* (mais aptos) têm uma porcentagem maior que os demais, fazendo assim com que eles sejam selecionados para reprodução com maior frequência.

Com os indivíduos selecionados, é escolhida uma posição aleatória na cadeia de caracteres para efetuar o processo de cruzamento. Tal processo é feito copiando a primeira parte da cadeia dividida de um dos “pais” do indivíduo para o primeiro filho, e a outra parte restante da cadeia do outro “pai” completando assim o primeiro filho. Esse processo é repetido para um segundo filho, porém trocando os trechos genéticos de cada pai. Tal processo está exemplificado na Figura 6, junto ao processo de criação de população, seleção e mutação (RUSSELL; NORVIG, 2010).

Russell e Norvig (2010) descrevem por fim o processo de mutação, onde uma pequena fração da carga genética, mediante a uma porcentagem pequena, tem essa fração alterada aleatoriamente. O processo com suas principais etapas são ilustrados na Figura 6.

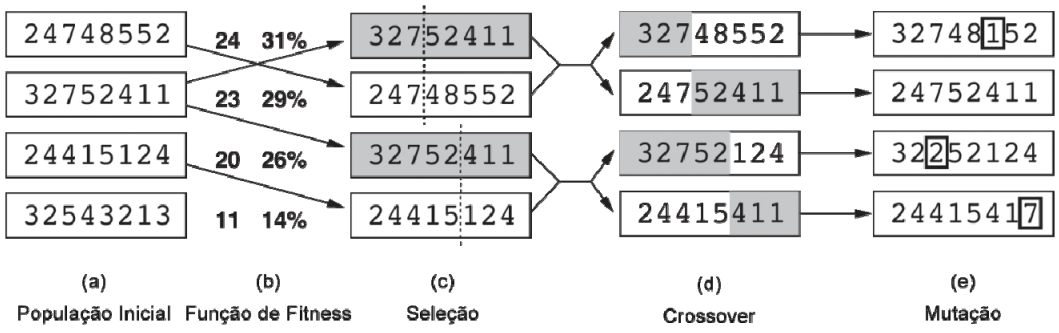


Figura 6 – Processo de mutação, adaptada de (RUSSELL. NORVIG, 2010).

Nesse processo tem-se a população inicial (a) ordenada pela função de *fitness* (b), resultando em pares para reprodução (c), produzindo “filhos” (d) que são submetidos à mutação (e). Após a sequência de ações descrita, a população é re-ordenada e o processo é repetido. Os critérios de parada deste algoritmo são geralmente: quando um número definido de gerações é atingido; ou quando um *fitness* mínimo é alcançado.

Uma variação do algoritmo genético é a programação genética. Esta variação, de acordo com Russell e Norvig (2010), difere do algoritmo genético apenas no que sofre mutação e é combinado, sendo que ao invés de ser uma cadeia de caracteres, são cadeias de funções ou de programas.

2.5 Programação Genética

A programação genética foi introduzida em 1988 por John Koza sob a forma de patente, nela é definido um algoritmo genético não linear para solução de problemas. (KOZA, 1990)

Segundo Koza (1990), sua invenção executa o processo de iteração em uma população de entidades que são soluções de um determinado problema. Primeiramente, as entidades ativas são acionadas produzindo resultados. Os resultados produzem valores que são associados à entidade geradora. Em seguida, as entidades com os valores associados mais altos são selecionadas. As entidades

selecionadas passam por uma operação (cruzamento, reprodução, mutação, etc.) produzindo novos indivíduos. Por fim os novos indivíduos são inseridos na população. Tal execução é ilustrada no fluxograma apresentado na Figura 7.

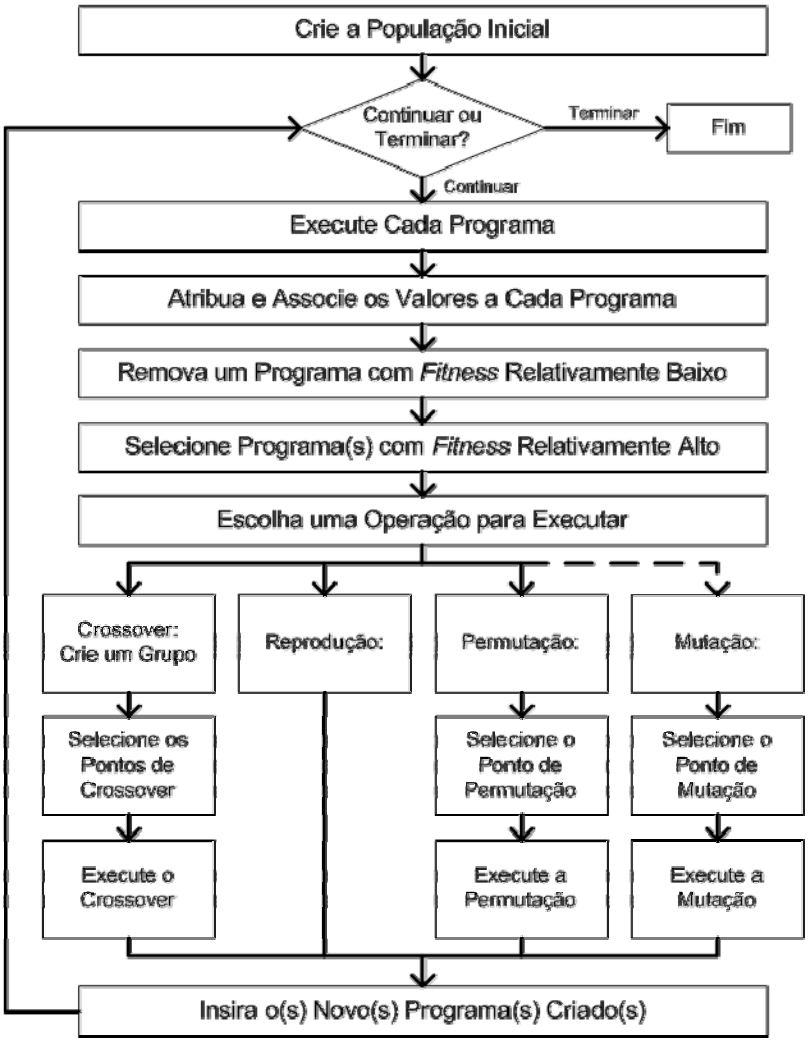


Figura 7 – Diagrama de Fluxo da Programação Genética. Adaptada de (KOZA, 1990)

De acordo com Koza (1992), este processo é equivalente a uma busca no espaço de possíveis programas, pelo programa mais apto à solução do problema proposto. Mais precisamente, o espaço de busca contém todos os programas compostos por funções e terminais pertinentes ao domínio do problema.

Seguindo um exemplo apresentado por Koza (1992), o processo de criação de indivíduos ocorre da seguinte forma. Supondo um conjunto de funções $F = \{+, -, \div, \times\}$ e um conjunto de terminais $T = \{A, B, C, D, E\}$, a criação tem como ponto inicial a seleção de um elemento do conjunto *F* de forma aleatória. Após a seleção

deste elemento, dependendo da aridade[3] da função, são selecionados agora elementos de quaisquer conjuntos até que todas as funções estejam completamente preenchidas. Por exemplo, a criação do indivíduo que representa a expressão $C + (A \times B)$ poderia ser obtida da forma representada na Figura 8.

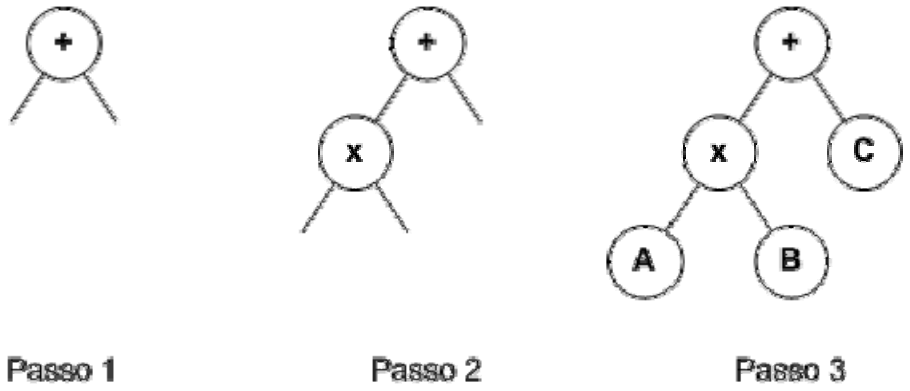


Figura 8 – Processo de geração de indivíduos. Adaptada de (KOZA, 1992)

Quanto aos operadores, as mesmas técnicas dos algoritmos genéticos são aplicadas na programação genética, sendo os operadores mais comuns o cruzamento e a mutação. Para uma melhor compreensão de como tais operadores são aplicados em indivíduos com esse tipo estrutural, os dois estão ilustrados na Figura 9 e Figura 10.

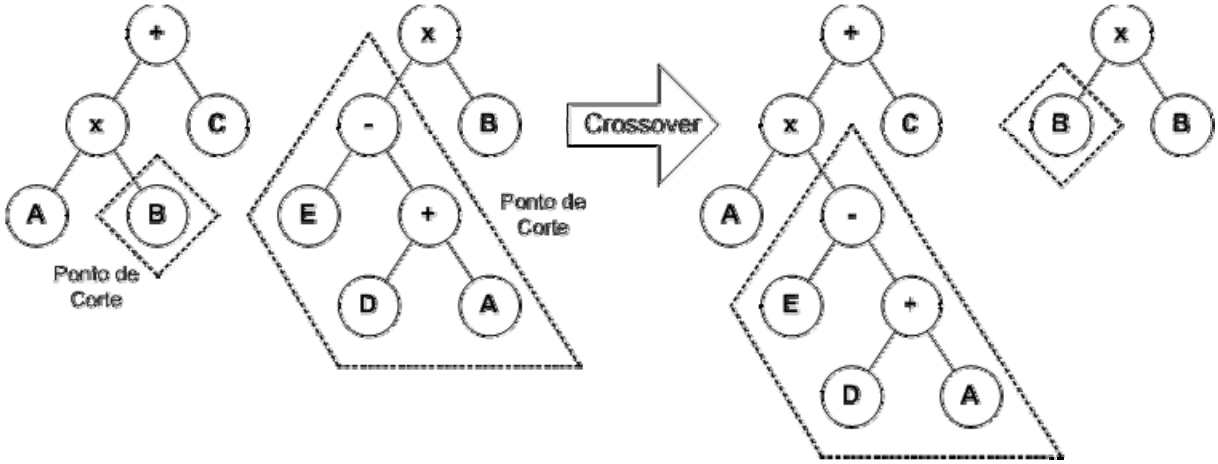


Figura 9 – Ilustração da operação de cruzamento. Elaborada pelo autor.

Kosa (1992) define o cruzamento como a recombinação sexuada, que cria variação na população produzindo uma nova geração que consiste de partes dos seus pais. O cruzamento utiliza dois indivíduos com *fitness* iguais ou próximos, que

[3] Termo matemático que indica em uma função ou operação, qual o número de argumentos ou operandos é utilizado. Por exemplo, a soma utiliza dois operandos, logo sua aridade tem valor dois.

depois de selecionados, aleatoriamente é escolhido um ponto de corte em cada um para troca de suas sub-árvores.

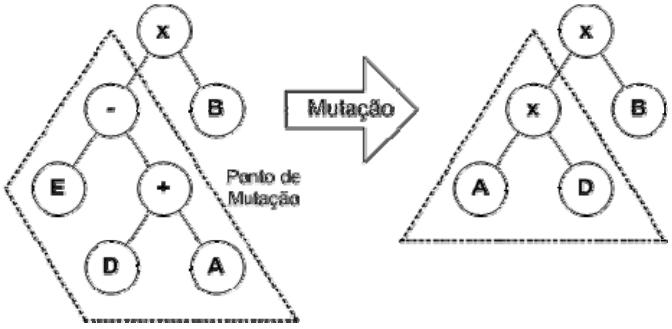


Figura 10 – Ilustração da operação de mutação. Elaborada pelo autor.

Já a mutação, Koza (1992) afirma ser responsável por reintroduzir diversidade à população, sendo que a parte do indivíduo que sofre mutação, é completamente substituída por uma nova sub-arvore gerada aleatoriamente.

Para conclusão do processo, os novos indivíduos gerados pelos operadores são inseridos à população, para que assim o critério de parada seja verificado definindo se o algoritmo atingiu ou não seu estágio final. Ao atingir o estágio final, o indivíduo que apresenta o melhor valor de *fitness* é considerado a melhor solução encontrada para o problema proposto.

Uma variação da programação genética pode ser identificada na literatura, onde é empregada a utilização de uma estrutura linear (sequência de operações) ao invés de em árvore. O objetivo de utilizar esse tipo de abordagem é a implementação direta em hardware do resultado obtido, sem que seja necessária uma reformulação posterior. Tal variação é encontrada, por exemplo, nos trabalhos de Pedrino et al. (2011a) e Wang, Chen e Lee (2008).

De acordo com Banzhaf et al. (1997), o principal impacto da variação na estrutura está na variação que os operadores genéticos trazem aos indivíduos gerados. Esse impacto pode ser ilustrado ao analisar a estrutura da árvore que, considerando uma árvore balanceada, cresce em progressão geométrica.

Ao analisar essa progressão do ponto de vista probabilístico, os nós do último nível da árvore podem ser escolhidos para mutação ou cruzamento com a probabilidade de ½, ao subir um nível, essa probabilidade passa a ser de 1/4, no próximo 1/8 e assim por diante. Esse fator indica que os nós mais distantes da raiz da árvore têm uma probabilidade maior de serem escolhidos para operações, e por estarem mais distantes causam apenas uma pequena variação no indivíduo como

um todo. Porém, Banzhaf et al. (1997) ressalta que apesar desse fator, são tomadas precauções para que quando esse tipo de estrutura é utilizado essa distribuição seja uniforme.

Descartando essa diferença, os métodos variam quanto à estrutura visando apenas se adequar à classe de problema para qual será empregado. As estruturas lineares são normalmente utilizadas para representação de cadeias de bits ou linhas de código de máquina, enquanto as estruturadas em forma de árvore são utilizadas para representação de programas.

Banzhaf et al. (1997) afirma que o método linear tende a apresentar uma execução mais rápida, mas para isso sacrifica a habilidade de testar mudanças na ordem de execução, que pode facilmente ser alterada em uma árvore alternando entre a busca de nós pré-ordem, pós-ordem ou em ordem.

2.6 Aplicações de IA em processamento de Imagens

A aplicação de métodos de IA no processamento de imagens é muito comum e a prática mais comum de aplicação de técnicas de IA, ocorre na identificação ou criação de filtros de imagens, assim como na detecção de características.

Killing, Surgenor e Mechefske (2009), sugerem uma técnica de visão computacional, onde é feita a detecção da falta de elementos de fixação em peças estampadas em aço através da utilização de um método híbrido neuro-fuzzy. O método proposto, quando comparado com técnicas baseadas em limiares (thresholds), apresentou melhor adaptabilidade quando avaliado nos conjuntos de testes. Porém, tal método enfrenta dificuldades quando ocorre variação de iluminação e/ou posição da câmera.

Para que a identificação dos elementos fosse possível, Killing, Surgenor e Mechefske (2009) precisaram isolar alguns fatores que dificultavam o processo como: as variações da luz ambiente ao longo do dia, resolvido com um painel luminoso; movimentação de fundo como movimento de outras máquinas e operários, resolvido com uma placa cobrindo o fundo; e a aparência do elemento em questão variava de acordo com o ângulo em relação a câmera, resolvido com um braço robótico que posicionava a câmera na posição correta.

A área de detecção de objetos em imagens estáticas e sequências de imagens é comumente associada a IA. Em sua proposta, Pinto e Song (2009) definem uma programação genética para construção de programas reconhecedores de movimento em imagens com e sem ruído. Com essa abordagem, é possível contornar as dificuldades indicadas no trabalho de Killing, Surgenor e Mechefske (2009).

A proposta de Pinto e Song (2009) apresenta uma solução para identificar apenas movimentos que sejam de interesse em imagens que apresentam ruído, mesmo sem o conhecimento prévio dos possíveis ruídos que possam aparecer nas imagens. Tal abordagem é capaz de se adaptar inclusive a novas condições climáticas, assim como variações na posição da câmera.

Para alcançar tais resultados, Pinto e Song (2009) utilizaram programação genética aplicada a imagens tratadas com uma variação do método de diferença temporal. O método de diferença temporal analisa cada quadro de uma sequência de imagens, construindo um “mapa” do que foi modificado das imagens anteriores acumulativamente. Essa abordagem possibilita uma identificação facilitada de como a imagem era no passado em relação à imagem atual.

Na Figura 11, Pinto e Song (2009) apresentam resultados significativos considerando todas as adversidades encontradas pela proposta, como câmera em constante movimento, ruídos e variações climáticas. Na figura estão representados os resultados da PG treinada com câmera em movimento sem ruído (a), (b) e (c); e com ruído em (d) e (e). Sendo testada sem ruído em (a), com adição de ruído em (b) e com chuva e adição de ruído em (c); também testada com adição de ruído em (d); com chuva e adição de ruído em (e).

De acordo com os resultados apresentados por Pinto e Song (2009), o conjunto treinado sem ruído apresentou um acerto de 99,33%, o treinado com ruído 98,54%. O que indica que a utilização de programação genética em problemas de visão computacional pode apresentar resultados significativos.

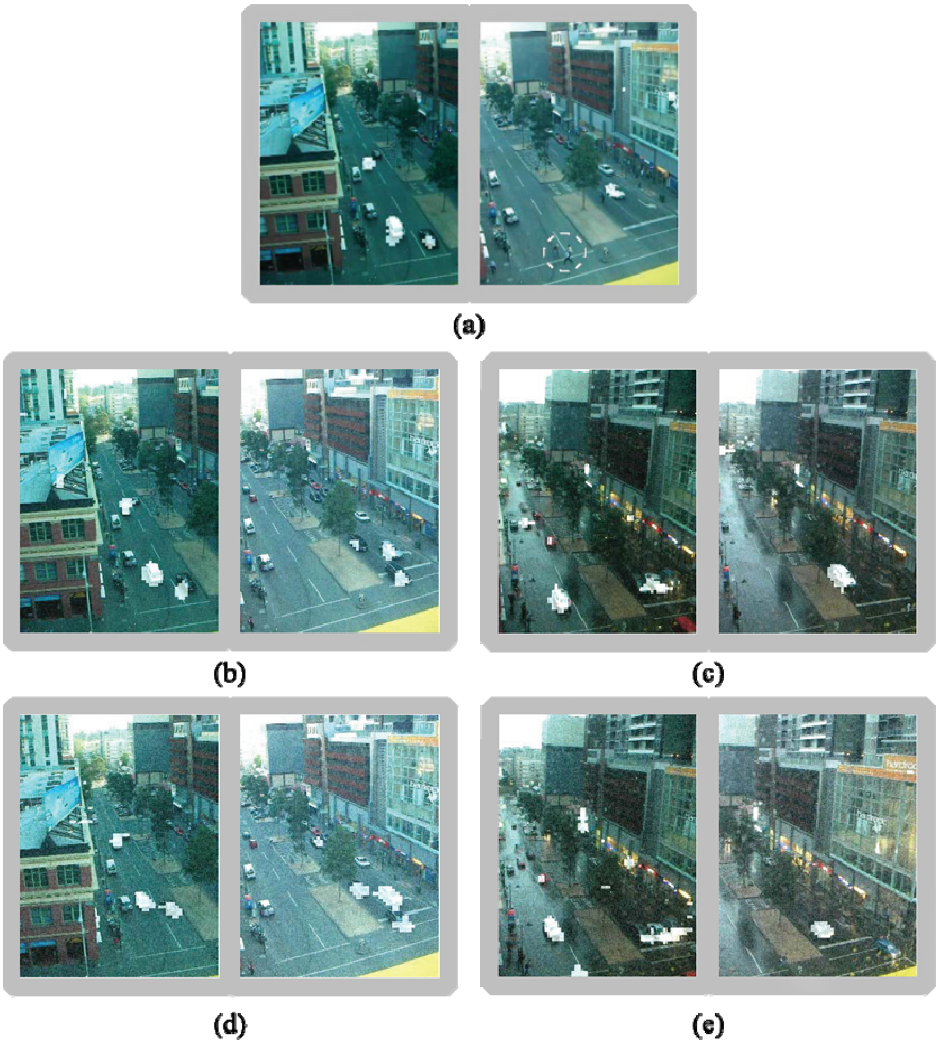


Figura 11 – Resultados apresentados por Pinto e Song (2009). Adaptada de (PINTO; SONG, 2009).

Utilizando também a abordagem de programação genética, Bhowan, Zhang e Johnston (2009) propõem classificação de imagens estáticas, em conjuntos de imagens não balanceados, através de um comparativo entre duas técnicas de programação genética. O primeiro método abordado utiliza uma função de *fitness* adaptativa onde o resultado final é um classificador com grande precisão em uma classe individual, o segundo método utiliza a abordagem com vários objetivos que evolui um conjunto de classificadores simultaneamente, podendo identificar classes predominantes e não predominantes.

Bhowan, Zhang e Johnston (2009) selecionaram dois conjuntos de dados, onde o primeiro conjunto cuja classe alvo eram pedestres possuía apenas 1/5 da amostra sendo imagens de pedestres e o restante imagens de paisagens, e o segundo conjunto cuja classe alvo eram faces, o conjunto de treino possuía 1/10 de imagens de faces e o restante eram imagens de paisagens.

Dentro de cada conjunto selecionado, Bhowan, Zhang e Johnston (2009) dividiram as imagens em quadrantes e, dentro de cada quadrante, fizeram análises estatísticas visando encontrar padrões em cada conjunto de imagens que correspondessem à classe alvo.

Os resultados obtidos por Bhowan, Zhang e Johnston (2009), podem ser considerados satisfatórios, sendo que para o primeiro conjunto, que possuía apenas 1/5 da amostra de treino como a classe alvo, apresentou uma taxa de acerto de 45,1% com um desvio padrão de 15,3%. E para o segundo conjunto cuja classe alvo era faces, mesmo contendo faces em apenas 1/10 da amostra, apresentou uma taxa de acerto de 16,8% com um desvio padrão de 6,9%.

Em uma abordagem mais genérica, Pedrino et al. (2011a) sugerem a utilização da programação genética linear para construção de operadores de imagens a partir de operações lógicas e morfológicas, onde dada uma imagem de entrada e uma imagem objetivo, a programação genética gera uma solução composta por um conjunto de operações que quando executadas na imagem de entrada o resultado seja o mais próximo possível à imagem objetivo.

O espaço de busca proposto por Pedrino et al. (2011a), está contido nas operações morfológicas, porém não contém necessariamente todas operações. Ao iniciar o algoritmo de programação genética, o usuário define as porcentagens de mutação e cruzamento, fornece as imagens de entrada e saída para o treino, assim como define quais instruções/operações morfológicas podem ser utilizadas na geração de indivíduos.

Os resultados apresentados por Pedrino et al. (2011a), indicam que a abordagem mostra uma resposta satisfatória na busca de sequências de operações visando atingir um determinado resultado. Um exemplo é demonstrado na Figura 12 onde um conjunto de operações age como um detector de bordas utilizando um conjunto de treino bem distinto do conjunto de teste apresentando bons resultados.

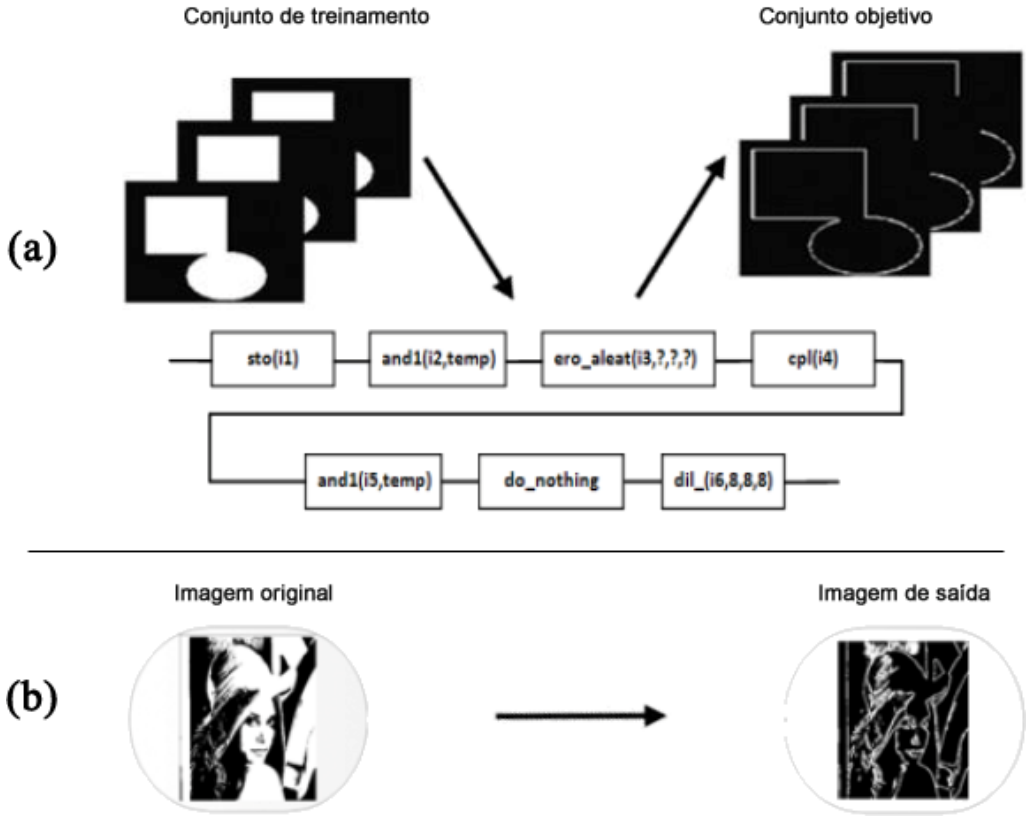


Figura 12 – Exemplo de resultado obtido por Pedrino et al. (2011ª). Adaptada de Pedrino et al. (2011a)

A Figura 12 apresenta em (a) a programação genética na fase de treinamento com o melhor indivíduo representado e em (b) a aplicação do melhor indivíduo encontrado em (a) em uma imagem teste e seu resultado

Outra abordagem para o problema abordado por Pedrino et al. (2011a) é a proposta por Wang, Chen e Lee (2008), onde é utilizada a programação genética, mas utilizando somente operações lógicas (*and*, *or*, *xor*, etc..) entre os pixels da imagem, ao invés de utilizar operadores morfológicos, visando obter um resultado parecido.

A abordagem de Wang, Chen e Lee (2008) tem como foco a utilização em dispositivos embarcados, assim como a de Pedrino et al. (2011a), porém em uma forma simplificada. Cada imagem de entrada é subdividida em partes menores para que então essas partes sejam passadas ao algoritmo genético para que ele tente encontrar uma sequencia de operações entre pixels, que quando aplicadas à subdivisão da imagem, seja obtido um resultado similar ao de Pedrino et al. (2011a).

Os testes de Wang, Chen e Lee (2008), envolvem a formação de filtros capazes, por exemplo, de remover ruídos adicionados artificialmente pelos métodos sal e pimenta e o método gaussiano. Nos resultados apresentados, nos melhores

cenários, foi possível alcançar um valor médio de diferença entre pixels (MDPP) de 2,30 para o filtro sal e pimenta e de 8,94 para o gaussiano.

2.7 Ferramentas voltadas para programação genética

Várias ferramentas auxiliam na implementação de sistemas baseados em programação genética. Foram selecionadas algumas delas através da análise de artigos recentes da área de programação genética com o intuito de identificar qual a tendência de uso para essa finalidade.

Para esta etapa foram selecionados sessenta e cinco artigos que foram analisados apenas em relação à ferramenta utilizada para programação genética. Com essa análise, foi constatado que as principais ferramentas utilizadas no período entre 2008 e 2013 são MatLab e GPLab que é um toolkit desenvolvido para MatLab especificamente para tratar problemas de programação genética.

Os artigos foram obtidos por pesquisas nas bases da IEEE, Elsevier e ACM. A busca em cada base foi feita através da utilização das palavras chave “Genetic Programming” ou “GP”, filtrando apenas por faixa temporal dentro do período de 2008 a 2013.

Tais artigos abordam diversas áreas de pesquisa, mas em um escopo geral, essas áreas de pesquisa podem ser divididas em classificação/predição (18 artigos), geração de soluções (17 artigos), problemas de programação genética (9 artigos), processamento de imagens (19 artigos) e relacionados à ferramentas de programação genética (2 artigos).

2.7.1 Problemas de classificação/predição

Nos problemas de classificação/predição temos, por exemplo, o trabalho desenvolvido por Delfianto, Khodra e Roesli (2011), onde é tratada a classificação de propagandas vinculadas ao conteúdo em que estão associadas. Outra aplicação é a proposta por Hu e Xie (2010) onde é feita a classificação de impressões digitais.

A programação genética é aplicada em diversas áreas, no trabalho de Uto, Kosugi e Ogata (2009), ela é utilizada para classificar fungos de carvalho. Nos

trabalhos de Farinaccio et al. (2010) e Worzel et al. (2009) é utilizada para auxiliar em pesquisas relacionadas ao câncer.

Puente et al. (2009) utiliza a programação genética para sintetizar índices de vegetação e com isso estimar cobertura de solo; Alavi e Gandomi (2012) para avaliação de liquefação solo; Kuo et al. (2009) propõe sua utilização para auxiliar na predição de interação entre proteínas a nível celular.

Outra proposta é a de Xu, Wang e Liu (2012) onde é feita a predição de acidentes em rodovias, ou seja, a programação genética pode ser aplicada a diversas áreas de classificação/predição, o que pode ser observado também nos demais artigos[4] dessa classe.

2.7.2 Problemas de geração de soluções

Para geração de soluções, problemas como o de geração de árvores de decisão proposto por Yi e Wanli (2011) e design de circuitos lógicos proposto por Xiong e Tanik (2011), são apenas alguns dos exemplos de aplicação da programação genética.

Gusel e Brezocnik (2011) propõem a utilização da programação genética para modelagem de materiais, onde utilizam a técnica para gerar uma liga de cobre; Nitsure, Londhe e Khare (2009) a utilizam para estimar a altura de ondas oceânicas; e Islam, Rico-Ramirez e Han (2012) utilizam para inferir parâmetros micro-físicos de distribuições de tamanho de gotas de chuva a partir de medidas de diversidade de polarização.

Dentre as mais variadas aplicações, Alhejali e Lucas (2010) propõem sua utilização na criação de agentes para o jogo Ms. Pac-Man, onde o processo evolutivo trata do comportamento de tais agentes.

Supari, Sawitri e Purnomo (2009) utilizam a programação genética para controlar a velocidade de um motor, avaliando o desempenho da solução gerada com o método mais comum utilizado para tal finalidade, constatando que os resultados são muito similares.

[4] Artigos relacionados à classificação/predição propostos por Trujillo et al. (2011); Purohit, Chaudhari e Tiwari (2010); Coelho, Fernandes e Faceli (2010); Sildam (2010); Shen, Karakus e Xu (2012); Aslam, Zhu e Nandi (2012); Chen, Kao e Tsai (2012); Shokrkar et al. (2012); Al-Madi e Ludwig (2012).

Assim como para classificação, pode-se notar que a programação genética pode ser também utilizada em outras classes de problemas, como também em outras aplicações além das apresentadas[5].

2.7.3 Problemas de programação genética

A programação genética possibilita ainda sua utilização para investigar características próprias, como seleção de função de *fitness* como proposto por Lin (2010) e Li e Zeng (2011).

Como também são encontradas pesquisas que busquem melhorias na programação genética, como é o caso da proposta feita por Kameya, Kumagai e Kurata (2008) que consiste na mineração de sub-árvores mais frequentes visando acelerar o processo evolutivo. Castelli, Vanneschi e Silva (2013) verificam o comportamento da programação genética, onde é considerada a semântica da solução encontrada.

Ou ainda analisar os parâmetros ideais para uma determinada classe de problema, como discutido por Chaudhary e Iqbal (2009), onde são explorados quais são os parâmetros ideais para problemas de regressão não-linear.

A programação genética, por possuir vários componentes, é alvo constante de pesquisas visando encontrar quais são as melhores combinações desses componentes, dado determinado problema, além de outras áreas de estudo[6].

2.7.4 Problemas de processamento de imagens

Porém a área de maior representatividade foi a de processamento de imagens que trata, por exemplo, classificação de objetos como proposto por Tamboli e Shah (2011), até evolução de imagens artísticas como proposto por Solt (2010).

[5] Artigos relacionados à geração de soluções propostos por Aleshunas e Janikow (2011); White, Arcuri e Clark (2011); Wang et al. (2010); Meuth (2010); To e Pham (2009); Borg, Rosner e Pace (2009); Gamage, Silva e Campos (2012); Garg e Tai (2012); Behbahani e Silva (2013); Lones et al. (2013).

[6] Artigos relacionados à programação genética propostos por Perez e Olague (2009); Langdon, Harman e Jia (2009); Xie e Zhang (2013); Weise et al. (2013).

Mahmood et al. (2013) propõe por exemplo a restauração de imagens para sistemas de vigilância.

Olague e Trujillo (2012) utilizam a programação genética multi-objetivos para detectar pontos de interesse em imagens estáticas como contornos e pontos de luz. Na mesma linha, Trujillo et al. (2012) tratam da criação de estimadores capazes de detectar estruturas, sendo essa feita através da programação genética.

Em outra aplicação, Al-Sahaf et al. (2012) propõem a criação de classificadores de imagens a partir da imagem em si, ou seja, sem utilizar regiões para auxiliar a classificação, como é comumente abordado o problema ; Hernández et al. (2012) utiliza a técnica para evoluir a "percepção visual" de um braço mecânico, auxiliando seu posicionamento. Sendo esses apenas exemplos de uma área com aplicabilidade ampla[7] para programação genética.

2.7.5 Específicos sobre ferramentas de programação genética

Castelli et al. (2010) analisam a habilidade de generalização em *frameworks* de programação genética e William e Northern (2008), abordam a terceira versão da *toolbox* GPLab descrevendo suas principais características e seu funcionamento.

2.7.6 Análise dos artigos em relação à ferramenta utilizada

Na Figura 13, estão identificadas as porcentagens referentes às ferramentas utilizadas dentro do conjunto de artigos selecionados e na Figura 14 está ilustrada a presença das ferramentas de acordo com o período analisado.

[7] Artigos de processamento de imagens propostos por Pedrino et al. (2011a); Pedrino et al. (2011b); Olague e Trujillo (2011); Wang e Tan (2011); Pedrino, Saito e Roda (2010); Guo, Bhattacharya e Kharma (2010); Bozorgtabar, Noorian e Rad (2010); Pedrino, Ogashawara e Roda (2010); Takamura, Matsumura e Yashima (2009); Neshatian e Zhang (2009); Spina et al. (2009); Trujillo e Olague (2008)

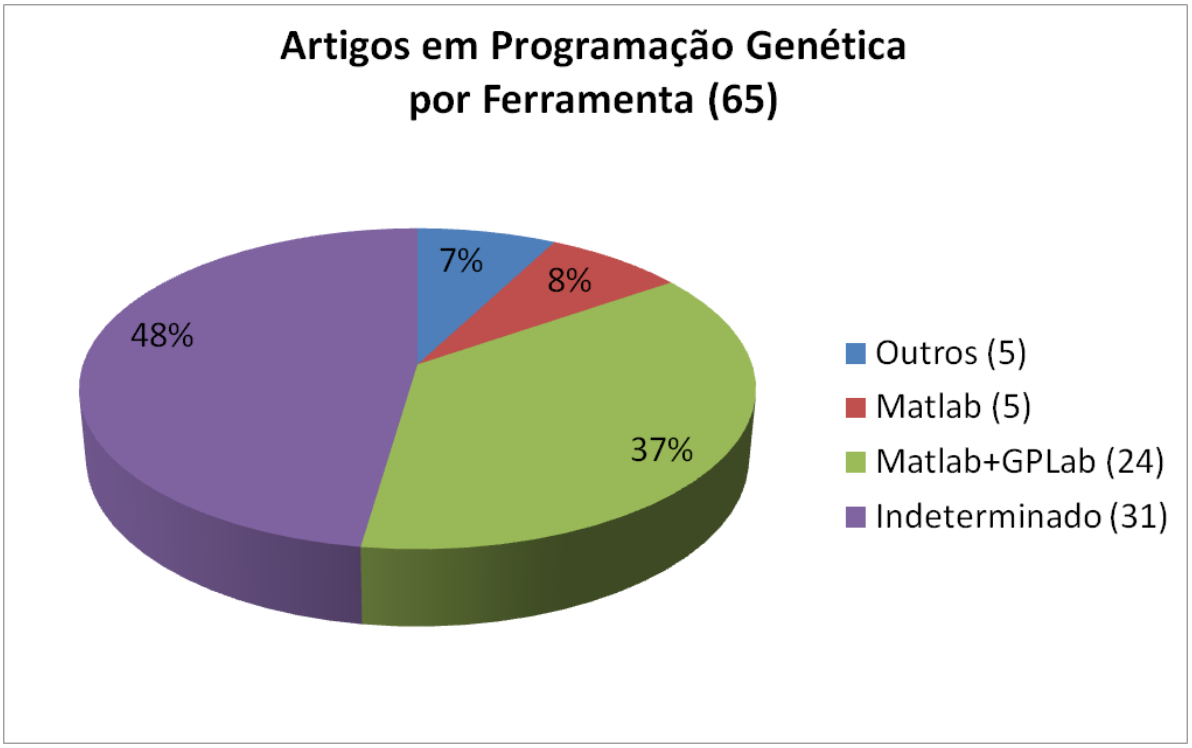
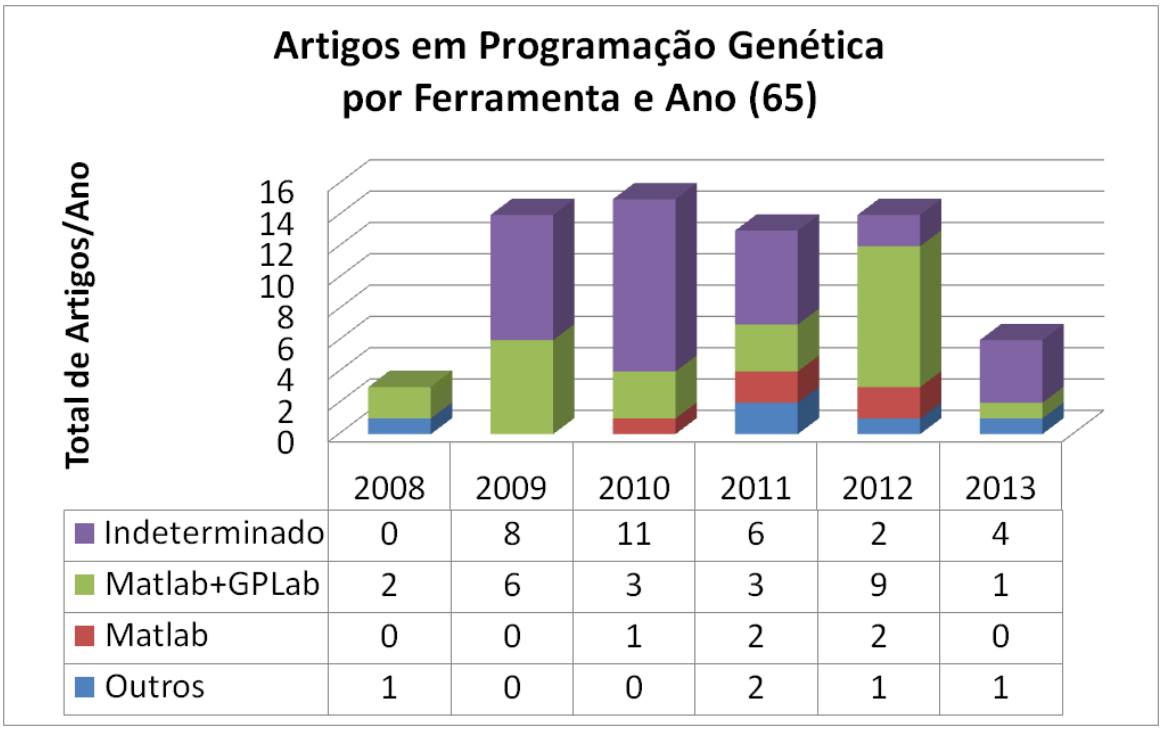


Figura 13 – Porcentagem de cada ferramenta dentro do conjunto de artigos analisados.

Apesar de quase metade dos artigos analisados não explicitar qual ferramenta foi utilizada durante o desenvolvimento, GPLab apresenta uma porção significativa da amostra, indicando ser possivelmente uma boa escolha para utilização no trabalho, e na Figura 14 é apresentada a análise temporal dos artigos.



	2008	2009	2010	2011	2012	2013
Indeterminado	0	8	11	6	2	4
Matlab+GPLab	2	6	3	3	9	1
Matlab	0	0	1	2	2	0
Outros	1	0	0	2	1	1

Figura 14 – Utilização das ferramentas ao longo do período analisado.

A Figura 14, ilustra o fato de, mesmo sem indicação da ferramenta utilizada em todos os artigos, o GPLab esta presente em artigos de todos os anos avaliados sendo o mais citado dentre as ferramentas indicadas nos artigos.

2.7.7 MATLAB

O Matlab é uma ferramenta desenvolvida pela MathWorks que utiliza linguagem de alto nível com foco em computação numérica, visualização e programação. Com essa ferramenta é possível desenvolver algoritmos, analisar dados, criar modelos e aplicativos. (MATHWORKS, 2013).

Por ser uma ferramenta focada em computação numérica tem uma base consolidada para expansão em áreas relacionadas, podendo assim ter suas capacidades estendidas para diversas áreas. A MathWorks vende toolboxes prontas para utilização com Matlab, sendo as de destaque do desenvolvedor as toolboxes para: computação paralela; matemática, estatística e otimização; processamento de sinais e comunicações; processamento de imagens e visão computacional; finanças e biologia. (MATHWORKS, 2013)

Como o Matlab possui suporte nativo a expansões, qualquer pessoa pode desenvolver uma toolbox e adicionar novas funcionalidades. Por ser uma ferramenta já conceituada na área acadêmica, sua utilização diminui os riscos de erros de implementação.

2.7.8 – A *toolbox* GPLab

A *toolbox* GPLab foi desenvolvida inicialmente por Silva e Almeida (2005) com o intuito de ser uma toolbox gratuita para MatLab que pudesse ser utilizada tanto por usuários simples, que quisessem apenas uma “caixa preta” de programação genética onde utilizariam apenas sua funcionalidade, como também por pesquisadores avançados que quisessem testar novos operadores genéticos, ordenadores de população (ranking) ou qualquer outra parte de um algoritmo de programação genética.

A *toolbox* tem como principal característica ser altamente modular, o que possibilita que novas funções e métodos sejam facilmente inseridos e testados no

ambiente gerado, possibilitando assim, ampla aplicabilidade nas mais diversas áreas de pesquisa.

É Composta por três módulos principais, GENPOP, GENERATION e SET VARS. O módulo GENPOP é o módulo responsável pela criação da população inicial, como também pelo cálculo do *fitness* dos mesmos. Os indivíduos são gerados com uma das funções pré-definidas: Grow, Full ou Ramped Half-and-Half, sendo a terceira função a padrão da toolbox. (SILVA E ALMEIDA, 2005)

O *fitness* padrão da toolbox, independentemente do módulo que está em execução, se dá através da soma das diferenças absolutas entre os valores obtidos e os valores esperados em todos os casos. Quanto menor o *fitness*, melhor o indivíduo, entretanto isso pode ser modificado para que a toolbox considere o maior *fitness* como melhor. (SILVA E ALMEIDA, 2005)

A execução deste módulo solicita alguns parâmetros para o módulo SET VARS e, após sua execução, passa o controle de execução para o módulo GENERATION, salvo quando apenas a criação da população inicial é solicitada pelo usuário. (SILVA E ALMEIDA, 2005)

O módulo GENERATION cria novas gerações de indivíduos através da aplicação dos operadores genéticos de cruzamento e mutação. Para essa etapa é executado um método de seleção dos indivíduos, e outro método é executado para calcular o número esperado de “filhos” gerados. A *toolbox* fornece quatro opções de métodos de seleção (*Roulette, SUS, Tournament e Lexictour*) e três métodos para o cálculo da expectativa de “filhos” (Absolute, Rank85 e Rank 89), sendo possível qualquer combinação dos métodos. Na *toolbox*, o *Lexictour* é o método padrão para seleção e o Rank85 para o cálculo de expectativa. (SILVA E ALMEIDA, 2005)

Os indivíduos são gerados por este módulo até que o valor definido para o tamanho da população esteja completamente preenchido. Após o cálculo do *fitness* dos novos indivíduos, o módulo SURVIVAL é executado para que seja feita a seleção de quais indivíduos serão aproveitados para a nova geração, sendo os indivíduos escolhidos de acordo com o parâmetro de elitismo. (SILVA E ALMEIDA, 2005)

Por fim, o módulo SET VARS é o responsável pela inicialização das variáveis com os valores padrões da *toolbox*, ou pela atualização de tais valores com opções do usuário (SILVA E ALMEIDA, 2005). Esse módulo é o que possibilita a inclusão de

novos métodos e funções na *toolbox*, sendo nele que se encontram as indicações de quais funções e métodos devem ser utilizados ao longo da execução do algoritmo.

Um exemplo de utilização foi elaborado durante a implementação e testes, nele está descrito em mais detalhes as principais formas de execução da toolbox, esse exemplo encontra-se no Apêndice A.

2.8 Considerações finais

Este capítulo visou realçar a utilização de algoritmos genéticos e da programação genética no contexto de processamento de imagens, assim como reforçar que são técnicas viáveis e que produzem bons resultados nessa área.

Foram introduzidos sistemas automáticos que utilizam visão computacional através de pesquisas recentes, assim como foram apresentados alguns conceitos de processamento de imagens e algumas aplicações.

Ainda neste capítulo foi apresentada a teoria de algoritmos evolutivos em conjunto com aplicações que utilizam IA para auxiliar problemas de processamento de imagens.

Concluindo o capítulo foram apresentadas pesquisas relacionadas à programação genética, com uma análise focada em quais ferramentas foram utilizadas. Por fim, foi feita uma breve análise das ferramentas que se destacaram.

Neste projeto será utilizada a programação genética em cenários similares às técnicas apresentadas por Pedrino et al. (2011a) e Wang, Chen e Lee (2008), visando identificar qual abordagem apresenta um comportamento mais completo quanto a criação de filtros de imagens. A análise do desempenho do filtro será realizada com base na comparação de propostas citadas, através da implementação adaptada das mesmas para a Toolbox GPLab.

Capítulo 3

DESCRIÇÃO E FORMULAÇÃO

Este trabalho tem como objetivo fornecer uma comparação de técnicas de apoio a sistemas automáticos baseados em visão computacional para identificação de elementos. Mais especificamente, qual técnica é capaz de encontrar filtros de imagens que possam ressaltar qualidades desejadas em uma imagem em um determinado cenário.

Para a criação de filtros de forma automática foi adotada a programação genética convencional, a qual será aplicada a dois métodos existentes na literatura através de suas adaptações. O primeiro método será similar ao proposto por Pedrino et al. (2011a) e o segundo método similar ao de Wang, Chen e Lee (2008).

No trabalho de Pedrino et al. (2011a) são utilizadas operações da morfologia matemática e operadores lógicos, que quando combinados geram uma sequência linear de operações para serem aplicadas a uma imagem.

Para o similar deste método no trabalho, também serão utilizadas operações da morfologia matemática e operadores lógicos, entretanto o resultado da programação genética terá suas operações dispostas em uma estrutura de árvore com as operações a serem aplicadas à imagem.

No trabalho de Wang, Chen e Lee (2008), também é proposta a criação de filtros de forma linear para aplicação direta em hardware. Em uma abordagem diferente da de Pedrino et al. (2011a), Wang, Chen e Lee (2008) propõem a utilização de somente operadores lógicos em uma janela deslizante, que percorre a imagem aplicando a sequência encontrada.

Seguindo a mesma variação utilizada para a comparação com o método de Pedrino et al. (2011a), para comparar com a abordagem de Wang, Chen e Lee

(2008), será utilizado neste segundo método o mesmo conceito de janela deslizante porém, ao invés de uma sequência linear de operações entre pixels ser utilizada dentro da janela, será utilizada uma árvore de operações.

A validação dos resultados obtidos pode ser realizada pela comparação com a programação genética obtida e com a programação genética linear proposta por Pedrino et al. (2011a) e o método que utiliza operações entre pixels proposta por Wang, Chen e Lee (2008).

3.1 Metodologia e Implementação

Os filtros de imagem serão criados a partir do uso da programação genética, seguindo os métodos previamente apresentados. Sua implementação será feita utilizando a *toolbox* de programação genética GPLab em conjunto com Matlab. Com isso será possível obter dados para avaliação dos métodos, e assim analisar seus desempenhos.

Para isso, foram executados diversos testes com configurações variadas, produzindo assim diversos níveis de complexidade para análise e comparação com os demais métodos.

A escolha da programação genética se torna interessante sendo que seu espaço de busca é limitado apenas às funções fornecidas, ou seja, caso algum problema seja identificado e existam novas funções que possivelmente auxiliam na resolução do problema, basta inseri-las à programação genética.

Com a inserção de novas funções é necessário que o processo evolutivo seja executado novamente para que uma nova solução seja encontrada, entretanto essa nova solução pode ou não conter as novas funções inseridas.

O esperado é que as propostas sejam mais rápidas que o processo exploratório manual, considerando um problema sem nenhum conhecimento prévio, sendo capaz de gerar um protótipo de visão computacional com resultados satisfatórios.

Os testes foram realizados em conjuntos de imagens gerados aleatoriamente sendo que, o conjunto deve conter pelo menos duas formas distintas entre si,

possibilitando assim que o algoritmo seja capaz de isolar apenas uma ou mais formas previamente selecionadas.

Esse conjunto de imagens é fornecido à programação genética junto com um gabarito assim como na proposta de Pedrino et al. (2011a), visando assim identificar um conjunto de transformações e efeitos que quando aplicados na imagem original tenham o resultado mais próximo possível ao gabarito.

Para o desenvolvimento dos testes foram utilizados quatro objetos, ilustrados na Figura 15, e esses objetos foram combinados aos pares para que fosse feita a remoção ou realce de um deles na imagem.

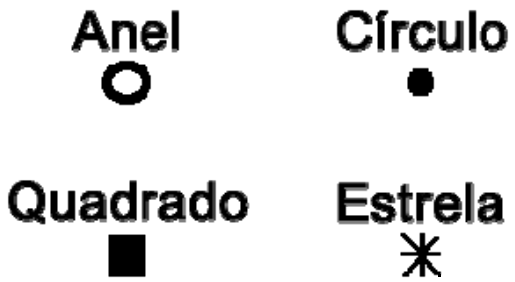


Figura 15 – Objetos utilizados nos testes.

Com esses objetos, foram geradas imagens aleatórias de dimensões 256x256 pixels, contendo 10 cópias de cada um dos dois objetos selecionados distribuídas nesse espaço. Não foram definidas restrições quanto às posições desses objetos, podendo assim ocorrer sobreposição dos mesmos.

Quanto aos métodos utilizados, o primeiro método é a versão em programação genética do método utilizado por Pedrino et al. (2011a), onde é utilizada a programação genética linear.

Esse método consiste na utilização de operadores lógicos (*and*, *or* e *not*) e operadores morfológicos (dilatação e erosão), que quando combinados através da programação genética, gerem uma árvore capaz de realçar/remover objetos de uma imagem.

O segundo método é a versão em programação genética do proposto por Wang, Chen e Lee (2008), onde foi desenvolvida uma abordagem voltada para hardware.

O segundo método é feito através de uma janela de dimensão 3x3 pixels, que percorre a imagem executando apenas operações lógicas (*and*, *or* e *not*) entre os pixels contidos nesta janela, e atribuindo o valor final obtido ao pixel central à janela deslizante.

A *ToolBox* GPLab foi utilizada no projeto, e por ser uma ferramenta de programação genética voltada à regressão linear numérica, apenas alguma adaptação foi necessária para que fosse possível utilizá-la no contexto proposto.

A única alteração que foi necessária na *toolbox* está apenas na forma de leitura dos arquivos de entrada e de objetivo. Na *toolbox* original, tais arquivos são tratados como matrizes contendo dados numéricos. Para a utilização com imagens, foi alterada a leitura para que pudesse ser fornecido um arquivo texto contendo os nomes dos arquivos de imagens a serem utilizadas pelo algoritmo.

Para a execução dos métodos propostos foi desenvolvida uma interface, ilustrada na Figura 16, com o intuito de facilitar a utilização na fase de testes da dissertação. Os detalhes das opções da interface podem ser consultados no Apêndice B. Todos os valores padrão da *toolbox* foram mantidos com exceção do número de gerações e tamanho da população.

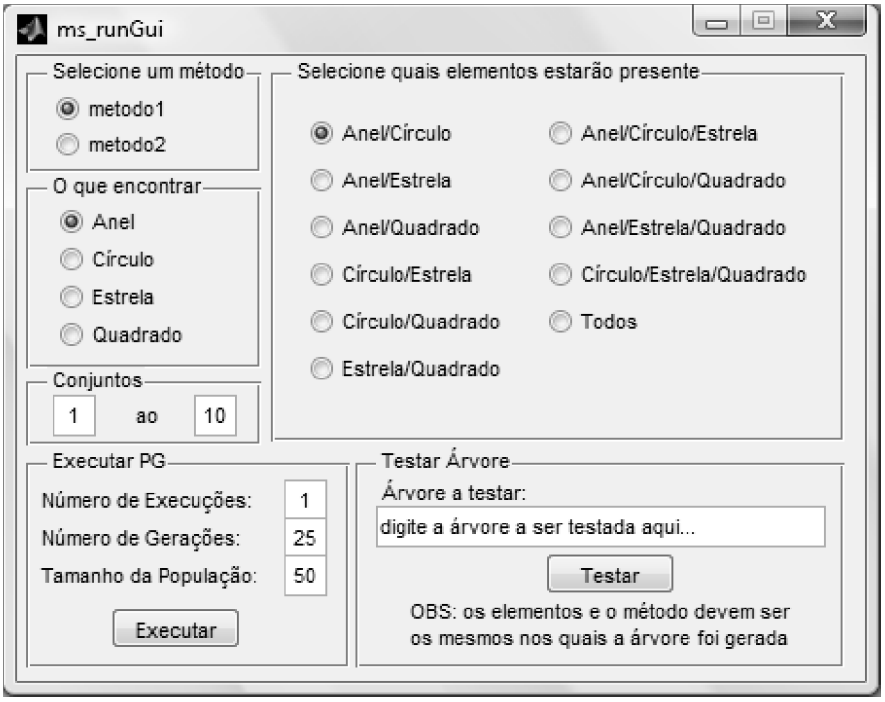


Figura 16 – Interface desenvolvida para os testes.

A execução ocorre inteiramente através desta interface, onde podemos executar o algoritmo visando encontrar uma solução, assim como é possível testar uma solução encontrada previamente.

Independente do objetivo é necessário primeiro selecionar qual método será utilizado, qual objeto será realçado, quais objetos estarão presentes na imagem e por fim, quais conjuntos de imagens deverão ser utilizados. Os conjuntos de imagens já foram previamente gerados aleatoriamente visando acelerar o processo.

Para a execução do algoritmo visando encontrar uma nova solução, é necessário definir também o tamanho da população, o número de gerações e número de execuções, ou seja, quantas vezes o algoritmo deve ser executado por X gerações com Y indivíduos.

Por exemplo, se definirmos o número de execuções como 2, com uma população de 50 indivíduos por 25 gerações para os conjuntos 1 ao 10. O algoritmo encontrará 2 soluções após evoluir por 25 gerações 50 indivíduos, com base nos conjuntos de imagens de 1 à 10.

A árvore gerada poderá então ser testada, por exemplo, nos conjuntos 11 ao 50. Possibilitando assim validar a resposta encontrada, assim como analisar seu desempenho em exemplos desconhecidos previamente.

Durante o teste das árvores é feito o cálculo do custo para então se obter o *fitness* de cada indivíduo, e para esse cálculo foi utilizado e erro médio absoluto que é obtido através da fórmula (1).

$$c(I_O, I_D) = \frac{1}{XY} \sum_{i}^{X} \sum_{j}^{Y} |I_O(i,j) - I_D(i,j)| \quad (1)$$

O custo $c(I_O, I_D)$ tem como parâmetros I_O que é a imagem obtida após a aplicação do filtro encontrado e I_D que é a imagem desejada (objetivo). O seu valor é calculado pelo somatório da diferença entre as duas imagens pixel a pixel, que é então dividido pela dimensão da imagem que é uma matriz de X linhas por Y colunas.

No caso deste trabalho o valor do custo é diretamente utilizado como *fitness* e sendo assim, o valor do *fitness* está sempre no intervalo [0;1]. Quanto menor o valor do *fitness* menor o número de pixels que diferem entre as imagens, ou seja, melhor o indivíduo.

Por estar entre zero e um é possível converter esse valor em uma porcentagem subtraindo o custo de um e multiplicando por 100, onde nessa porcentagem o inverso ocorre, sendo que 100% indicam então o acerto total do filtro para aquele caso. Essa porcentagem é apresentada nas tabelas de análise de *fitness*.

Após a execução de todos os testes foi então feita a análise comparativa do *fitness*, assim como também é feita uma análise adicional utilizando métricas estatísticas comumente utilizadas para avaliar classificadores.

Capítulo 4

RESULTADOS

Para cada objeto de teste serão analisados os comportamentos dos dois métodos tanto para o destaque quanto para a remoção utilizando como métrica apenas o *fitness*. Após essa análise inicial, uma análise adicional é apresentada utilizando métricas de exatidão, sensibilidade, especificidade e precisão, com intuito de avaliar os resultados obtidos.

4.1 Forma de ilustração dos resultados

Os resultados serão ilustrados aos pares de elementos, e separados pelo elemento que está definido com o objetivo de destaque ou remoção. Por exemplo, será mostrado o comportamento da programação genética na tentativa de isolar ou remover o quadrado, quando combinado com os elementos: anel, círculo e estrela.

Essa combinação dois a dois foi realizada com os quatro elementos, abrangendo todas as combinações possíveis de pares que poderiam ser gerados. Com isso é possível analisar o comportamento de cada elemento e deduzir que alguns elementos são mais simples de serem isolados, assim como outros são mais simples de serem removidos.

Durante a análise dos resultados, também é avaliado o comportamento de cada um dos dois métodos apresentados, para que com essa avaliação seja possível analisar a eficiência e indicar qual é mais indicado para aplicação nessa classe de problema.

Ao término da análise aos pares de elementos, é feita uma análise geral, e nessa análise são apresentados gráficos contendo os dados de todos os elementos para os dois métodos. Nos mesmos gráficos está destacado o desvio padrão que é necessário, pois todos os valores apresentados são referentes à média do grupo de imagens utilizado nos testes das árvores encontradas.

Quando uma solução é encontrada, o algoritmo retorna uma árvore contendo um conjunto de operações, que quando aplicadas devem encontrar um resultado próximo ao esperado, ou ainda em alguns casos, exatamente o resultado esperado.

Nos casos apresentados, como foram tratados dois métodos, serão apresentadas duas soluções em duas árvores distintas, onde cada uma representa um dos métodos selecionados. A árvore gerada automaticamente pela *toolbox*, nos casos de maior complexidade, apresenta as informações de forma incompleta ou ilegível, sendo assim as representações das árvores foram alteradas para facilitar a avaliação nesse trabalho, como representado na Figura 17.

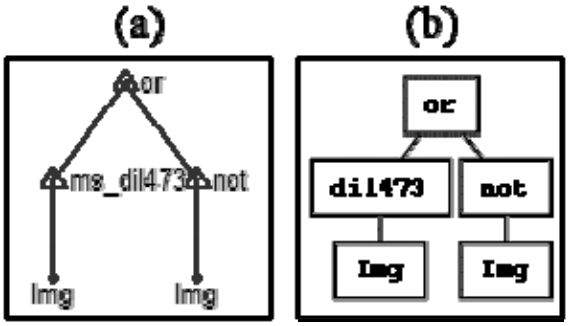


Figura 17 – (a) árvore resultante gerada pelo GPLab e (b) árvore adaptada para este trabalho. Elaborada pelo autor.

Para o método que utiliza operações morfológicas, serão apresentadas árvores contendo operações morfológicas e operações lógicas. Por exemplo, *or*(dil473(Img),*not*(Img)) é uma árvore que representa a seguinte operação; aplique o operador lógico *not* à imagem original e também aplique uma dilatação com o elemento estruturante 473 também à imagem original, por fim aplique o operador lógico *or* aos resultados.

O código do elemento estruturante equivale à sua representação binária, mapeada a uma matriz 3x3, ou seja, o elemento estruturante 473 tem como representação binária 111011001b que quando mapeada em uma matriz 3x3 tem na primeira linha 111, na segunda linha 011 e na terceira 001, logo o elemento estruturante final seria o equivalente a matriz:

$$\begin{matrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{matrix}$$

Após aplicar a solução em uma imagem original, o que será analisado é o resultado dessa aplicação, e para facilitar a visualização será utilizado um esquema de cores para destacar determinadas características que desejamos analisar no resultado, como na Figura 18.

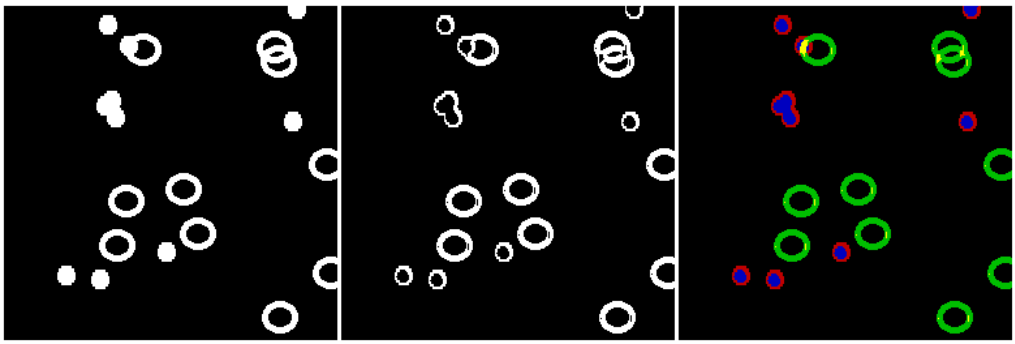


Figura 18 – está representado respectivamente: (a) imagem original a ser tratada, (b) a imagem tratada com a árvore encontrada pela programação genética e (c) a diferença entre (a) e (b).

A representação do resultado será feita no formato exemplificado pela Figura 18c, onde o objetivo neste exemplo é remover os círculos e realçar os anéis, sendo que as cores representam:

- Verde: são os verdadeiros positivos, ou seja, os que continuam na imagem e que deveriam continuar.
- Azul: são os verdadeiros negativos, ou seja, os que foram removidos da imagem corretamente.
- Vermelho: são os falsos positivos, ou seja, os que continuam na imagem, mas não deveriam.
- Amarelo: são os falsos negativos, ou seja, os que foram removidos da imagem, mas não deveriam.

Para melhor entender o significado da Figura 18c, ela será decomposta e explicada de diversas formas, para ilustrar as diversas verificações que podem ser feitas através do mapa de cores. Visando ilustrar melhor essa representação, foi elaborada a Figura 19.

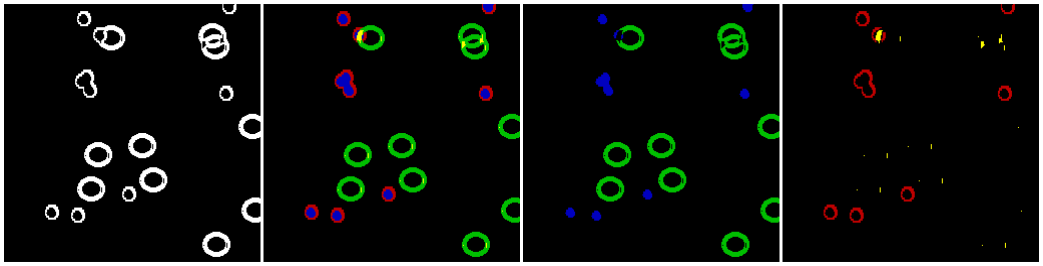


Figura 19 – Decomposição1: (a) imagem resultado, (b) imagem resultado colorida para análise, (c) resultados positivos e (d) resultados negativos

Na Figura 19c estão os positivos da árvore, ou seja, o que está destacado em verde e azul foi executado corretamente. Em seguida (Figura 19d) podemos analisar

os negativos, sendo as partes em amarelo e em vermelho, que são os elementos que a árvore tratou incorretamente.

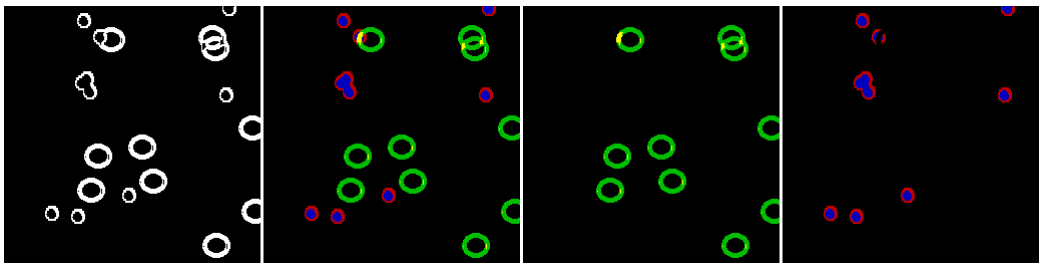


Figura 20 – Decomposição2: (a) imagem resultado, (b) imagem resultado colorida para análise, (c) resultados esperado e (d) resultados inesperado

Outra analise pode ser feita onde analisamos o resultado esperado e o inesperado. Na Figura 20c ao analisarmos o que está destacado em verde e em amarelo, temos o resultado que esperávamos, ou seja, nosso objetivo, e Figura 20d temos o que não esperamos que apareça no resultado, que é o que desejamos remover.

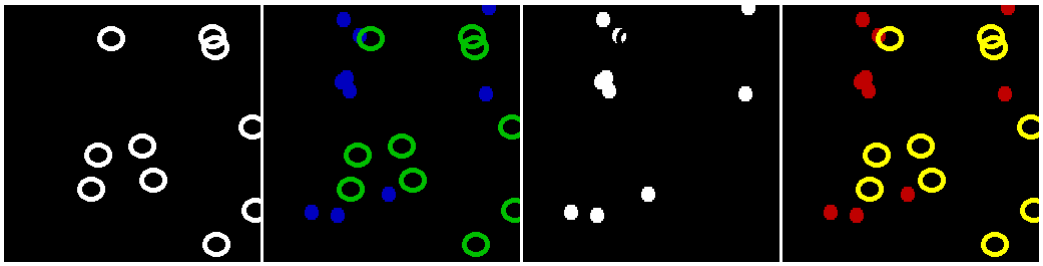


Figura 21 – Resultado gerado artificialmente para ilustrar melhor e pior caso, sendo: (a) o melhor resultado, (b) melhor resultado colorido para análise, (c) pior resultado e (d) pior resultado colorido para análise.

Sendo assim, a resposta ideal seria uma imagem com todos os objetos a serem realçados estarem completamente em verde, e todos os objetos a serem removidos estarem completamente preenchidos de azul como na Figura 21b. Os pontos amarelos são por sua vez, partes do objeto a ser realçado que foram apagadas, e os pontos em vermelho, partes do objeto a ser removido que não foram apagadas, representado o pior caso como na Figura 21d.

As imagens apresentam os resultados dos métodos propostos lado a lado, para assim facilitar a comparação dos resultados. As imagens apresentam o primeiro método (utilizando operações lógicas e morfológicas) à esquerda, e o outro método (que utiliza a janela com operações pixel a pixel) à direita.

Ao fim dos destaques é apresentada uma tabela contendo o *fitness* encontrado para cada um dos métodos. Como o *fitness* varia entre 0 e 1, esse valor é subtraído de 1 e multiplicado por 100 para representar a porcentagem de acerto.

4.2 Resultados utilizando o Objeto Anel

4.2.1 Resultados de Destaque

Quando em conjunto com o círculo, nenhum dos métodos conseguiu destacar completamente o anel. O primeiro método removeu apenas o centro do círculo deixando sua borda, já o segundo método praticamente não alterou a imagem.

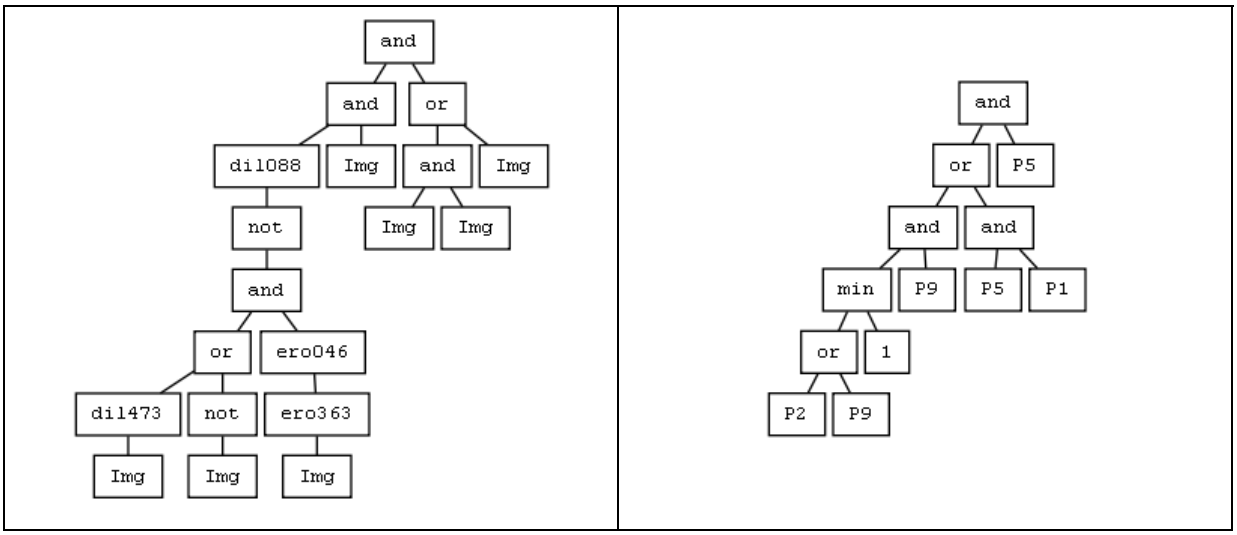


Figura 22 – Árvores resultantes da combinação entre anel e círculo, visando realce do anel. À esquerda, resultado do método 1 e à direita do método 2.

Na Figura 22 estão respectivamente as árvores geradas pelo método 1 e pelo método 2, que produzem os resultados representados na Figura 23 para a análise gráfica.

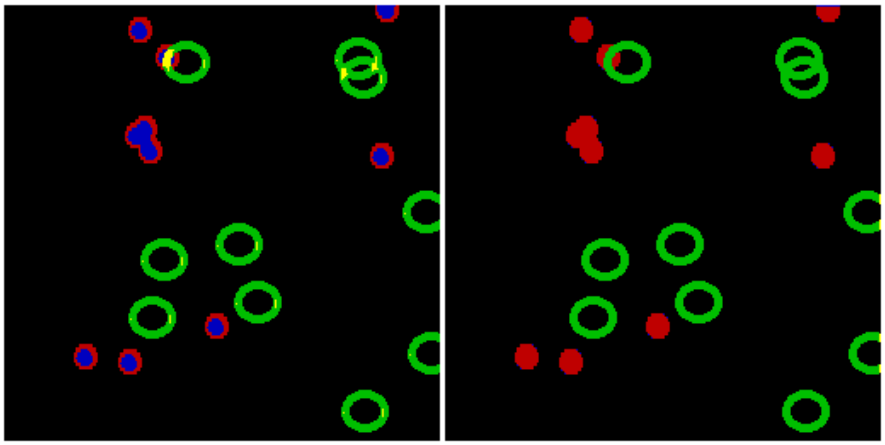


Figura 23 – Combinação entre anel e círculo, visando realce do anel. À esquerda, resultado do método 1 e à direita do método 2.

Ao associar o anel com a estrela, o primeiro método consegue um desempenho quase perfeito, deixando uma pequena falha onde ocorre sobreposição

entre os elementos, já o segundo método, não consegue limpar perfeitamente a estrela, deixando um traço diagonal como também apaga partes das bordas do anel.

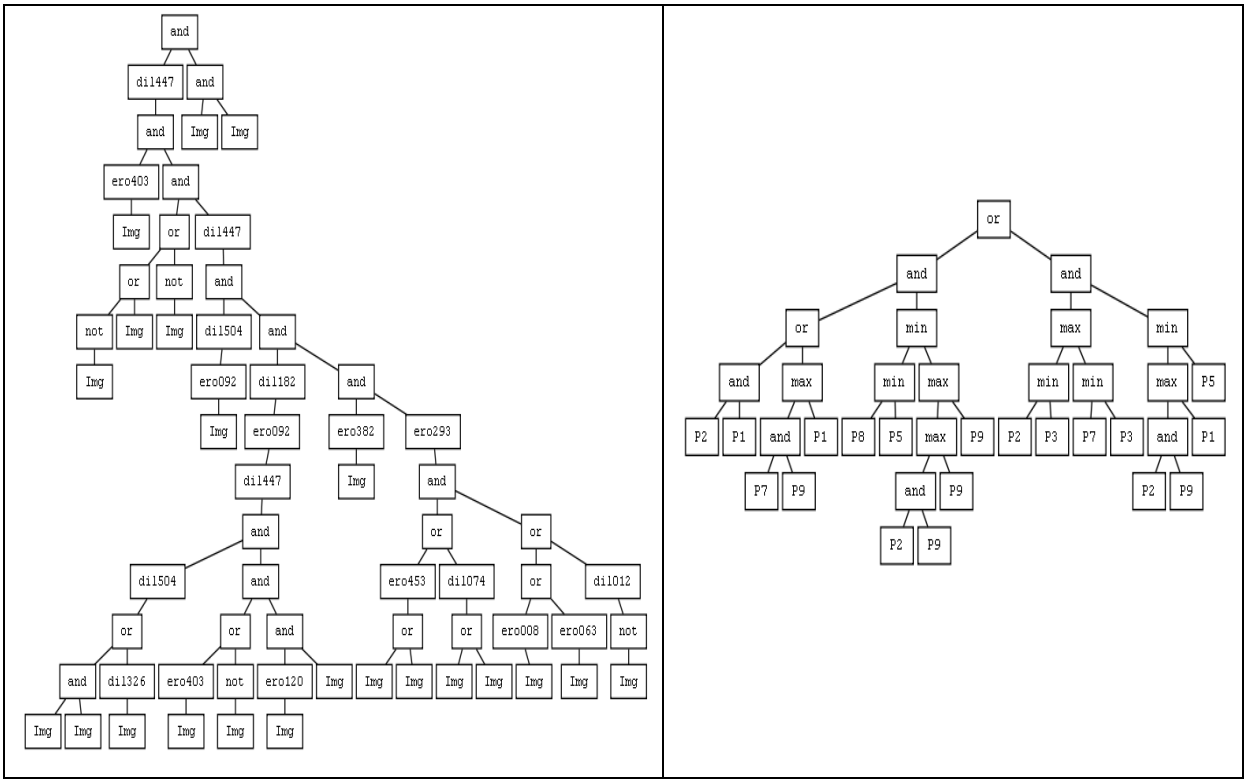


Figura 24 – Árvores resultantes da combinação entre anel e estrela, visando realce do anel. À esquerda, resultado do método 1 e à direita do método 2.

Nessa combinação foi necessária uma árvore mais complexa como visto na Figura 24, e ao observar o resultado na Figura 25, pode-se notar que tal árvore por sua vez consegue gerar bons resultados.

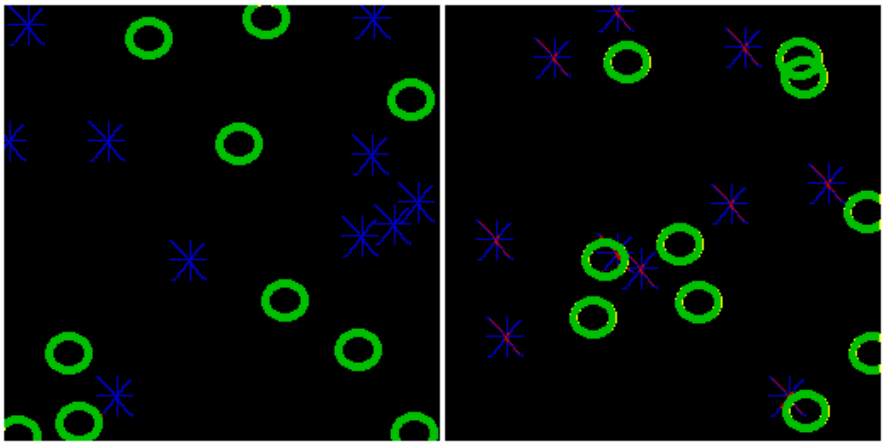


Figura 25 – Combinação entre anel e estrela, visando realce do anel. À esquerda, resultado do método 1 e à direita do método 2.

Quando combinado com o quadrado, os dois métodos apresentaram algumas dificuldades. O primeiro método deixou rastros do quadrado e removeu parte do

anel, em contrapartida o segundo método removeu o quadrado por inteiro, mas também removeu boa parte do anel, deixando apenas alguns resquícios.

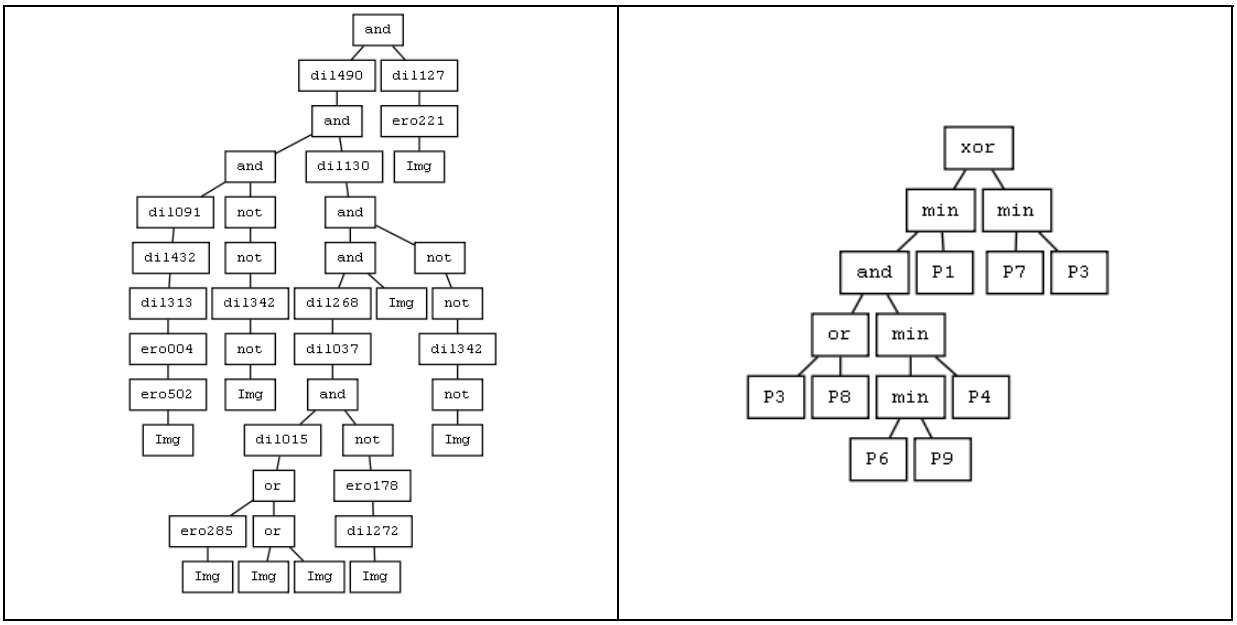


Figura 26 – Árvores resultantes da combinação entre anel e quadrado, visando realce do anel. À esquerda, resultado do método 1 e à direita do método 2.

As árvores geradas para esta combinação, que estão representadas na Figura 26 são distintas na complexidade, e ainda assim independentes quanto à complexidade, não geram resultados satisfatórios como ilustrado na Figura 27. O que indica que a complexidade da árvore não implica necessariamente na qualidade da solução encontrada.

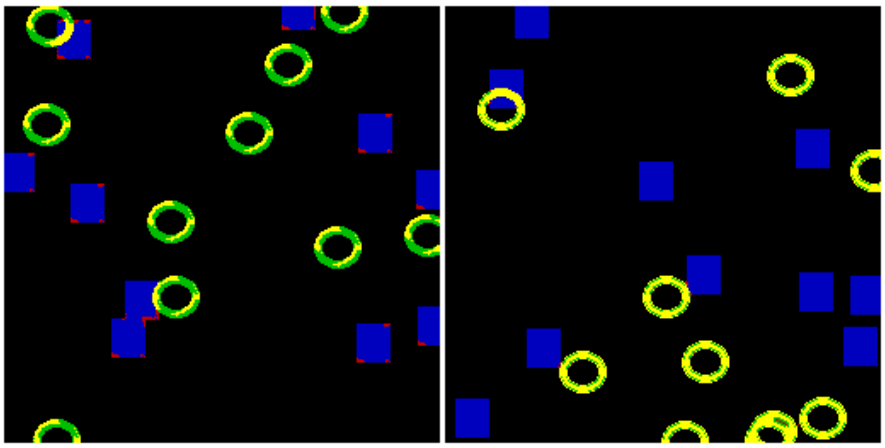


Figura 27 – Combinação entre anel e quadrado, visando realce do anel. À esquerda, resultado do método 1 e à direita do método 2.

Ao analisar os *fitness* desses casos, como ilustra a Tabela 1, é possível afirmar que os indivíduos do primeiro método são melhores que os do segundo método.

Tabela 1 – Tabela de comparação de *fitness* entre os dois métodos para realce do objeto anel.

	***Fitness* Método 1**	**%**	***Fitness* Método 2**	**%**
Anel/Círculo	0,016032	98,40	0,024348	97,57
Anel/Estrela	0,000446	99,96	0,006918	99,31
Anel/Quadrado	0,020892	97,91	0,036136	96,39

Com isso é possível afirmar que, mesmo não sendo 100% exato, o primeiro método é o mais indicado para realce do elemento anel.

4.2.2 Resultados de Remoção

Quando associado ao círculo, o anel foi removido com sucesso pelos dois métodos, porém com o primeiro método, parte da borda do círculo também foi removida. Entretanto o segundo método removeu o círculo inteiro, o que inviabiliza sua utilização, sendo que ao aplicar o segundo método não resta nada na imagem para ser analisado.

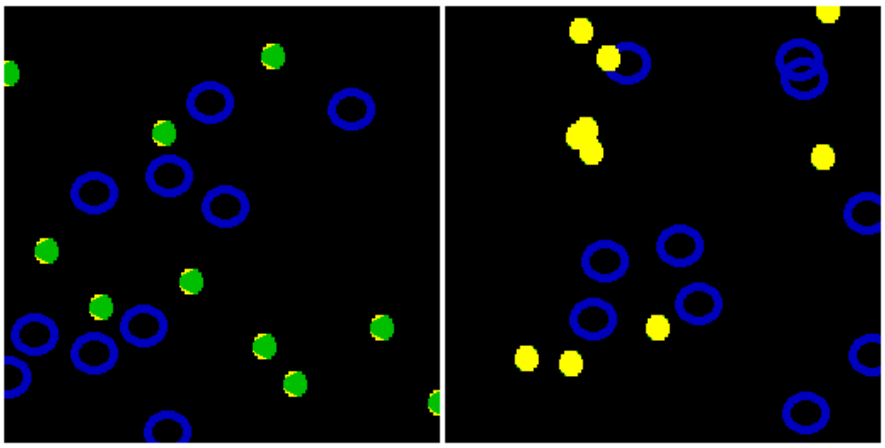


Figura 28 – Combinação entre anel e círculo, visando remoção do anel. À esquerda, resultado do método 1 e à direita do método 2.

Ao remover o anel associado à uma estrela, os dois métodos obtiveram sucesso, porém o primeiro método foi o que mais se aproximou do ideal, tendo em vista que o segundo método removeu uma linha diagonal da estrela, assim como deixou de remover partes da borda do anel.

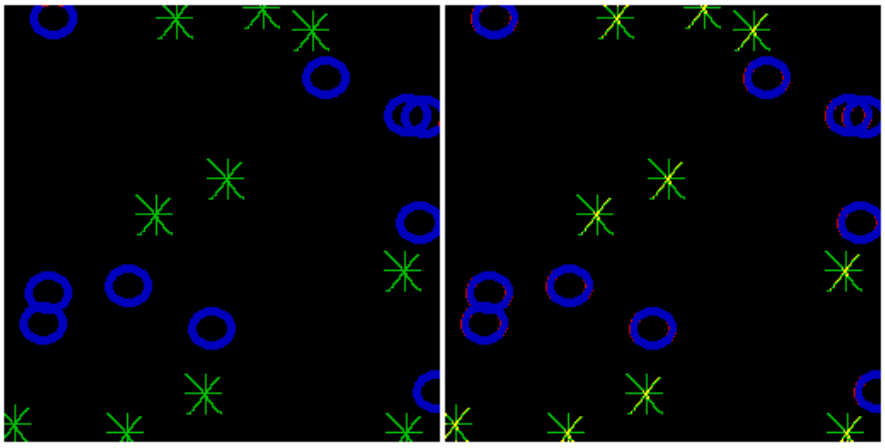


Figura 29 – Combinação entre anel e estrela, visando remoção do anel. À esquerda, resultado do método 1 e à direita do método 2.

Por fim, ao tentar remover o anel quando associado ao quadrado, o primeiro método foi mais eficaz, sendo que removeu os anéis por completo e apenas danificou parte da borda dos quadrados. Já o segundo método, removeu apenas a borda do anel, assim como diminuiu o tamanho do quadrado por inteiro.

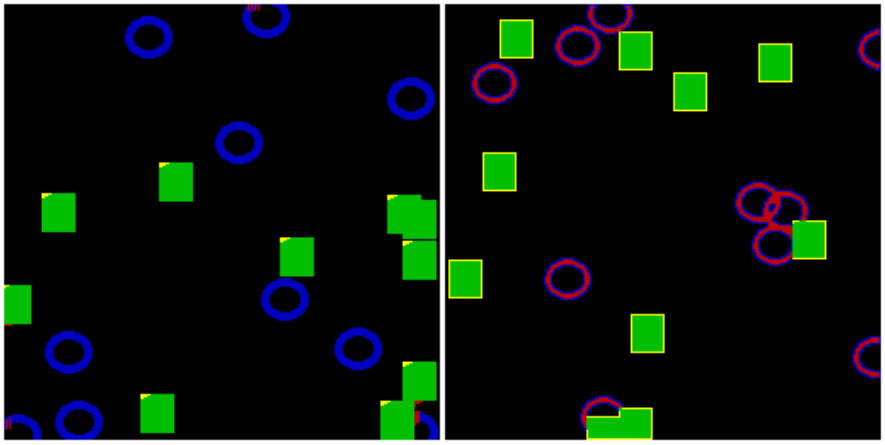


Figura 30 – Combinação entre anel e quadrado, visando remoção do anel. À esquerda, resultado do método 1 e à direita do método 2.

Com essa análise, podemos concluir que o anel é um objeto de fácil remoção, independente do outro objeto presente na imagem, porém essa afirmação é mais verdadeira para o primeiro método, sendo que o segundo apresenta dificuldades nos três casos.

4.3 Resultados utilizando o Objeto Círculo

4.3.1 Resultados de Destaque

Ao associar o círculo ao anel, o primeiro método conseguiu remover o anel completamente, porém, removeu também parte da borda do círculo. Entretanto, o desempenho do primeiro método é claramente superior ao segundo método, sendo que o mesmo removeu tanto o anel quanto o círculo, deixando apenas resquícios quase que imperceptíveis do círculo.

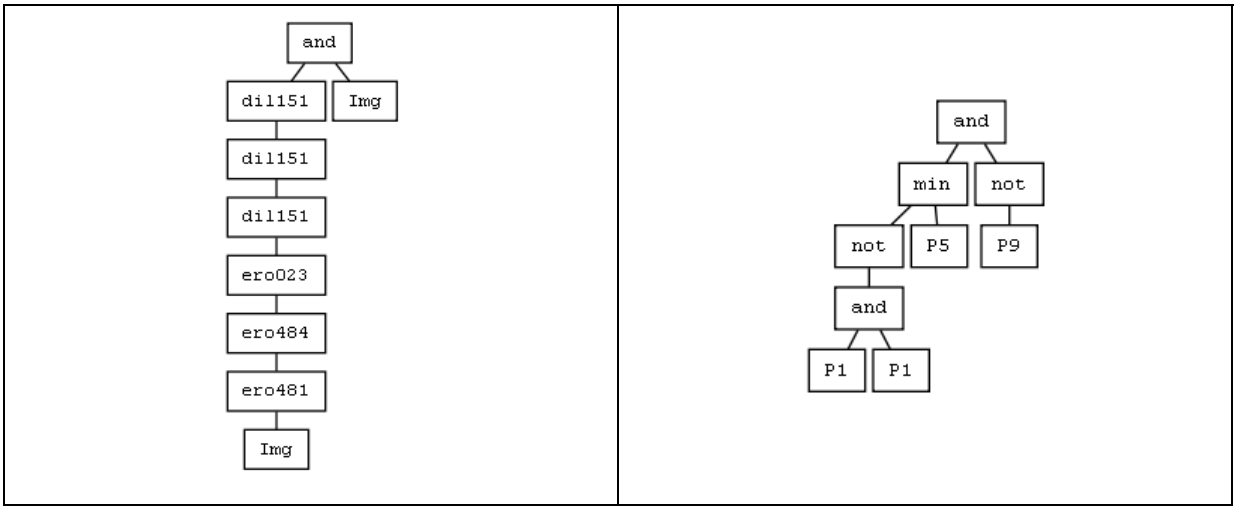


Figura 31 – Árvores resultantes da combinação entre círculo e anel, visando realce do círculo. À esquerda, resultado do método 1 e à direita do método 2.

A Figura 31 ilustra a árvore resultante do processo evolutivo para essa combinação e, na Figura 32 estão os resultados obtidos pela aplicação da solução nas imagens originais.

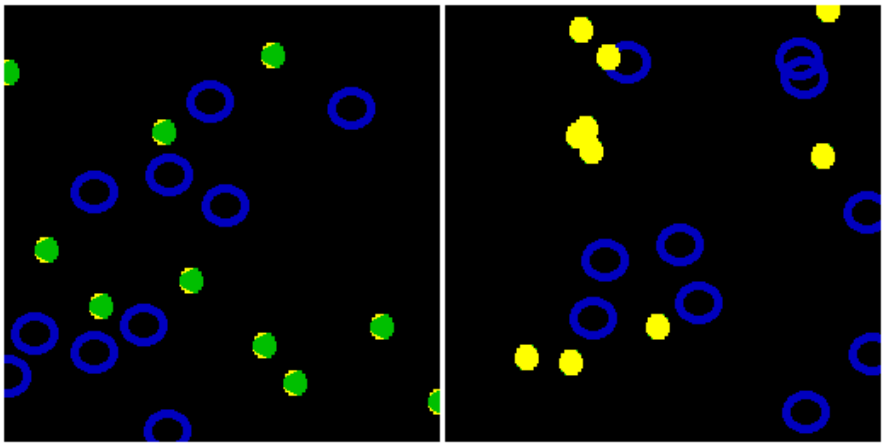


Figura 32 – Combinação entre círculo e anel, visando realce do círculo. À esquerda, resultado do método 1 e à direita do método 2.

Já na associação do círculo com a estrela, ambos os métodos apresentam bons resultados, mas vale ressaltar que o primeiro método consegue um desempenho perfeito, onde o segundo método não consegue limpar perfeitamente a estrela, deixando seu centro sem remoção, assim como remove parte da borda do círculo.

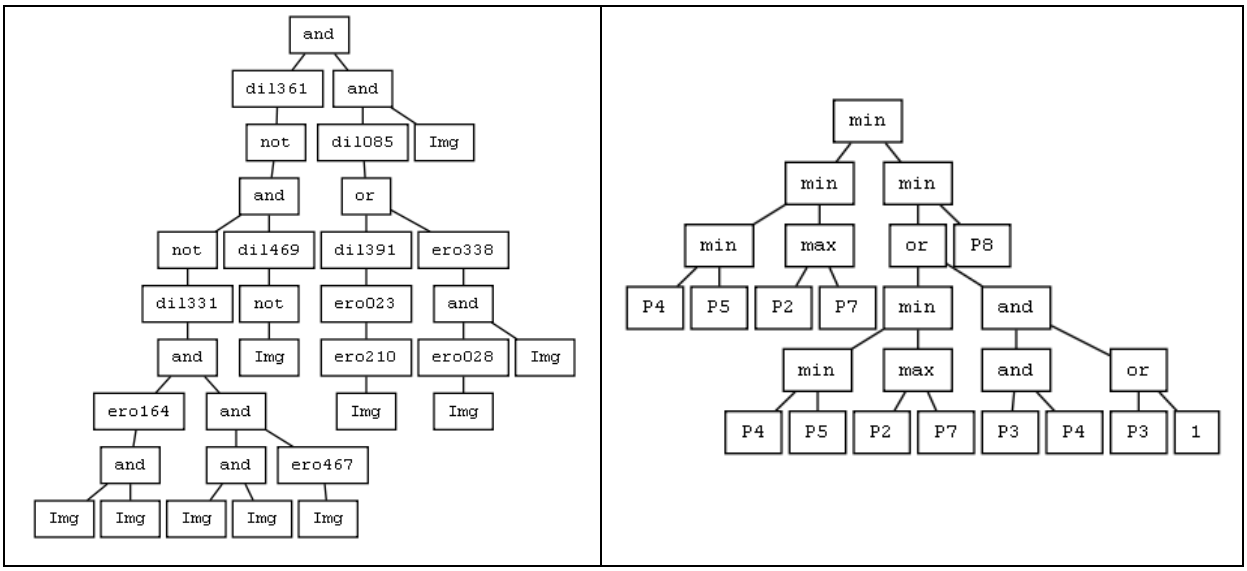


Figura 33 – Árvores resultantes da combinação entre círculo e estrela, visando realce do círculo. À esquerda, resultado do método 1 e à direita do método 2.

Ao analisar a Figura 33, observando a árvore encontrada do segundo método, é possível identificar uma operação descartável onde temos or(P3,1) que poderia ser substituída diretamente pelo valor 1. Com essa observação, pode-se afirmar que as árvores são realmente geradas ao acaso, podendo ainda essa operação ser resultado de um cruzamento com outro indivíduo. Através da aplicação das mesmas são obtidos os resultados da Figura 34.

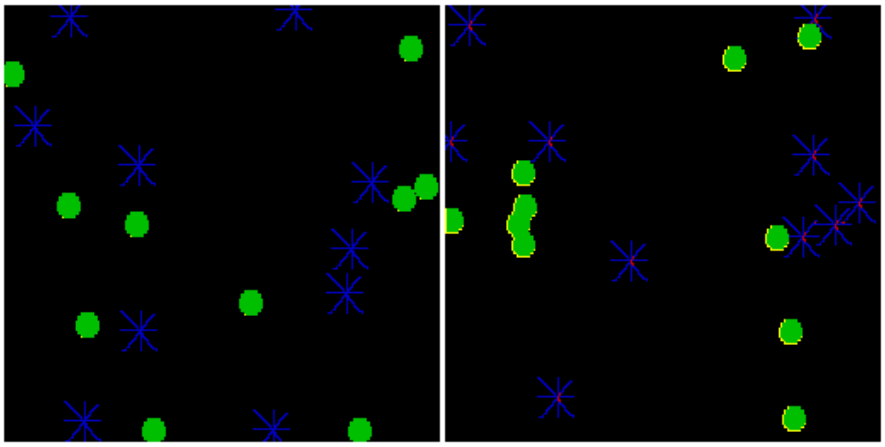


Figura 34 – Combinação entre círculo e estrela, visando realce do círculo. À esquerda, resultado do método 1 e à direita do método 2.

Na combinação com o quadrado, o primeiro método remove perfeitamente o quadrado, mas remove também boa parte do círculo, porém, o segundo método removeu o quadrado por inteiro e também removeu o círculo quase que por completo, deixando apenas alguns resquícios.

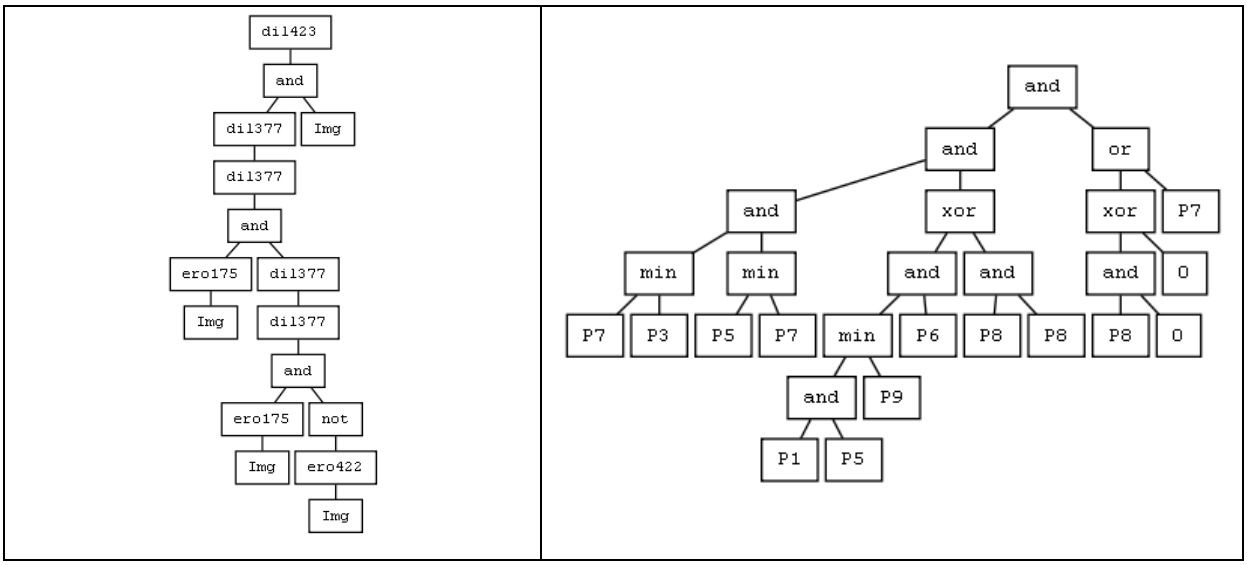


Figura 35 – Árvores resultantes da combinação entre círculo e quadrado, visando realce do círculo. À esquerda, resultado do método 1 e à direita do método 2.

As árvores geradas para esta combinação estão dispostas na Figura 35, que quando aplicadas na imagem original, geram os resultados apresentados na Figura 36.

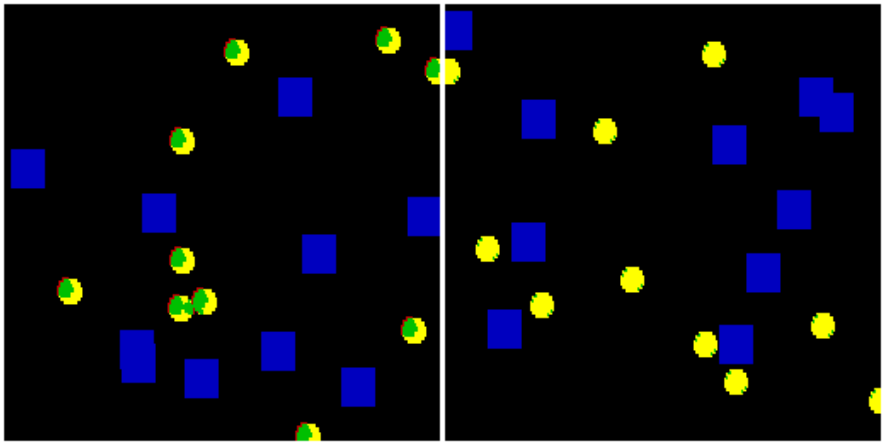


Figura 36 – Combinação entre círculo e quadrado, visando realce do círculo. À esquerda, resultado do método 1 e à direita do método 2.

Analisando os *fitness* desses casos, como ilustrado na Tabela 2, é possível afirmar que os indivíduos do primeiro método são melhores que os do segundo método.

Tabela 2 – Tabela de comparação de *fitness* entre os dois métodos para realce do objeto círculo.

	***Fitness* Método 1**	**%**	***Fitness* Método 2**	**%**
Círculo/Anel	0,005597	99,44	0,025073	97,49
Círculo /Estrela	0,000348	99,97	0,005038	99,50
Círculo /Quadrado	0,018782	98,12	0,024440	97,56

Portanto é possível afirmar que, mesmo não sendo 100% exato, o primeiro método é o mais indicado também para realce do elemento círculo.

4.3.2 Resultados de Remoção

Associado ao anel, a remoção do círculo não foi bem sucedida em nenhum caso, mas ao observar o primeiro método, pelo menos o interior dos círculos foi removido, o que apesar de não ser um resultado ideal, é melhor que o resultado apresentado pelo segundo método, onde a imagem praticamente não sofre alteração.

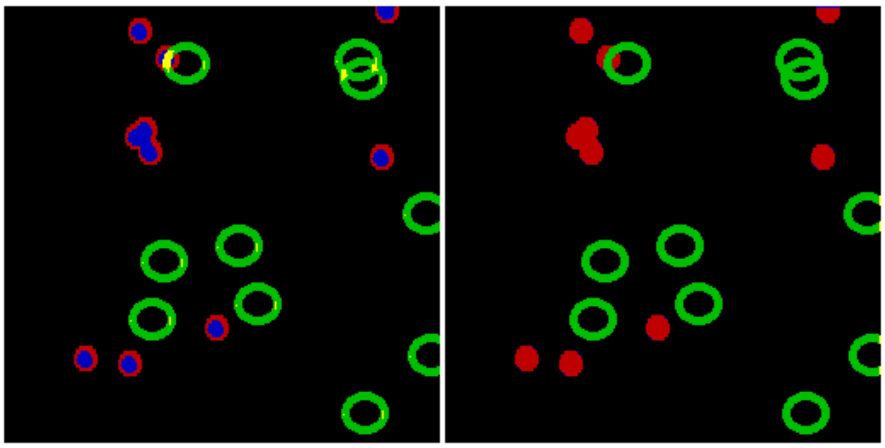


Figura 37 – Combinação entre círculo e anel, visando remoção do círculo. À esquerda, resultado do método 1 e à direita do método 2.

Ao remover o círculo quando associado à estrela, os dois métodos obtiveram sucesso, porém o primeiro método atingiu quase o resultado ideal deixando apenas erros onde ocorre sobreposição, já o segundo método removeu parte do centro da estrela, que deveria permanecer inalterada, como também deixou de remover partes da borda do anel, deixando uma meia lua sem remoção.

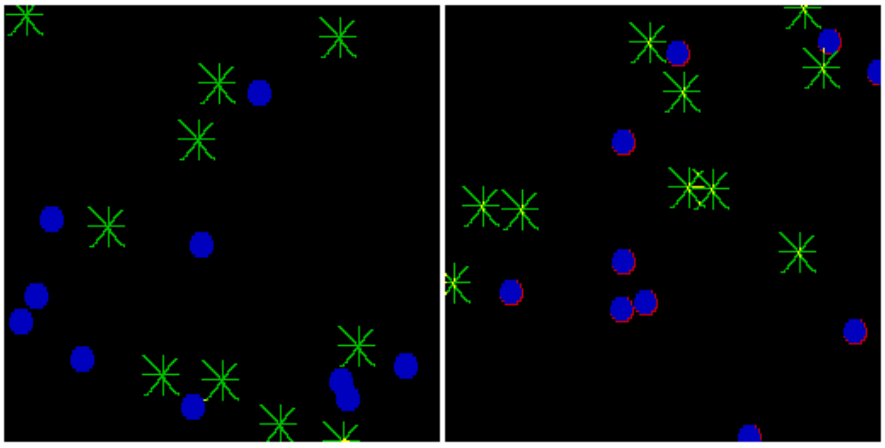


Figura 38 – Combinação entre círculo e estrela, visando remoção do círculo. À esquerda, resultado do método 1 e à direita do método 2.

Ao tentar remover o círculo quando associado ao quadrado, ambos os métodos apresentaram um desempenho insatisfatório, ainda assim o primeiro método foi mais eficaz, apesar de danificar o quadrado e remover apenas parte da borda do círculo, ainda altera melhor a imagem do que o segundo método que apenas remove uma pequena parte da borda dos dois objetos.

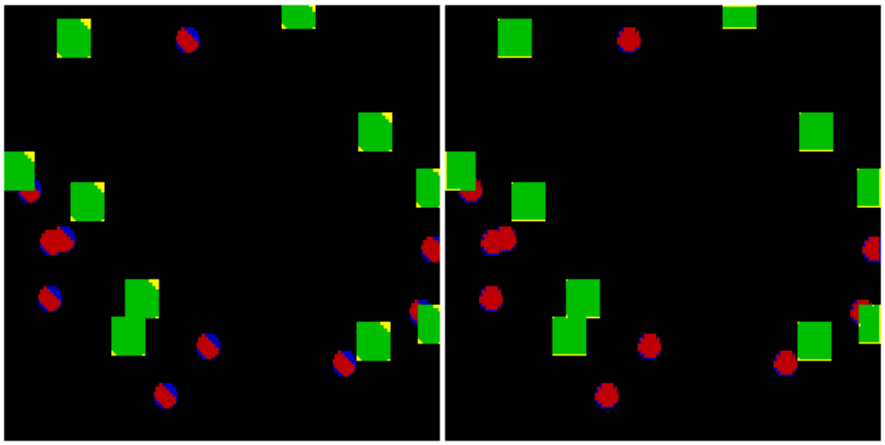


Figura 39 – Combinação entre círculo e quadrado, visando remoção do círculo. À esquerda, resultado do método 1 e à direita do método 2.

Com essa análise, podemos concluir que a remoção do círculo não é trivial, sendo que apenas quando associado à estrela foi removido com sucesso por ambos os métodos, e mesmo com o primeiro método tendo um desempenho melhor nas demais associações, o desempenho está muito distante do ideal.

4.4 Resultados utilizando o Objeto Estrela

4.4.1 Resultados de Destaque

Quando em conjunto com o círculo, os dois métodos apresentaram bons resultados, porém o primeiro método destacou sem falhas a estrela. O segundo método apesar de apresentar do resultado ser bom, não removeu completamente o anel, assim como removeu uma linha diagonal da estrela.

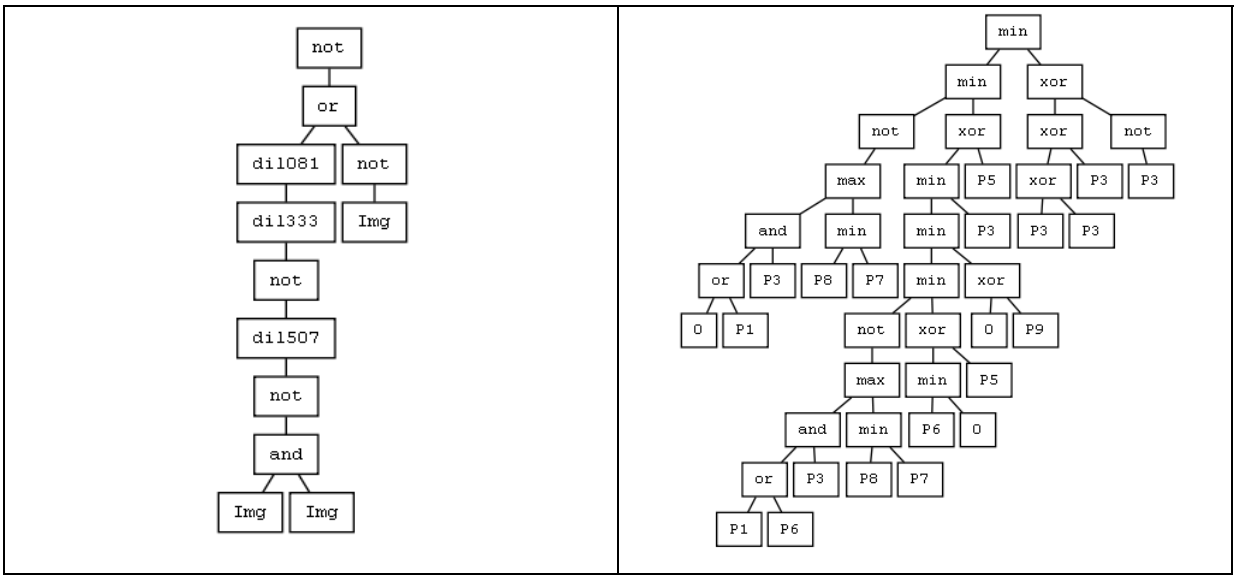


Figura 40 – Árvores resultantes da combinação entre estrela e anel, visando realce da estrela. À esquerda, resultado do método 1 e à direita do método 2.

Na Figura 40 temos as árvores que geram os resultados apresentados na Figura 41 e desta vez, mesmo o método 1 possuindo uma árvore mais simples que a do método 2, apresenta um resultado melhor.

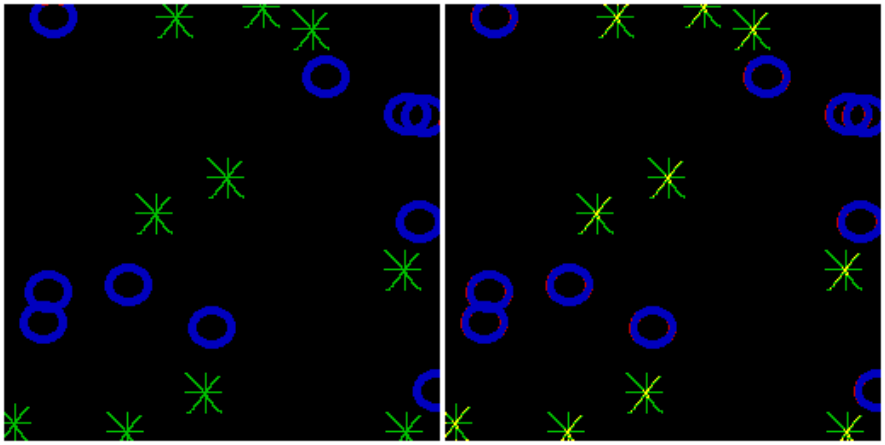


Figura 41 – Combinação entre estrela e anel, visando realce da estrela. À esquerda, resultado do método 1 e à direita do método 2.

Ao associar a estrela e o círculo, o primeiro método consegue um desempenho quase perfeito, deixando apenas um pequeno erro onde ocorre sobreposição entre os elementos, já o segundo método se comporta da mesma forma que na associação anterior, não consegue limpar perfeitamente o círculo, assim como remove uma parte do centro da estrela.

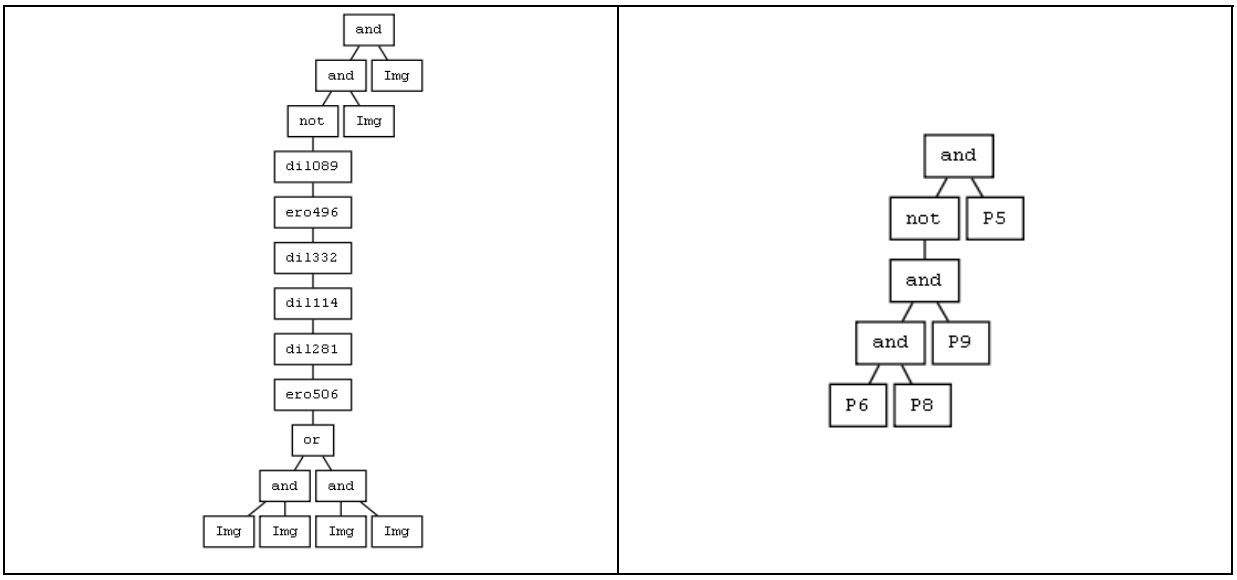


Figura 42 – Árvores resultantes da combinação entre estrela e círculo, visando realce da estrela. À esquerda, resultado do método 1 e à direita do método 2.

Em contrapartida com a combinação anterior, na Figura 42 temos que a árvore do método 2 é mais simples que a do método 1, e ao analisar os resultados na Figura 43, ambos estão muito próximos. Porém a do método 1 continua com um resultado superior.

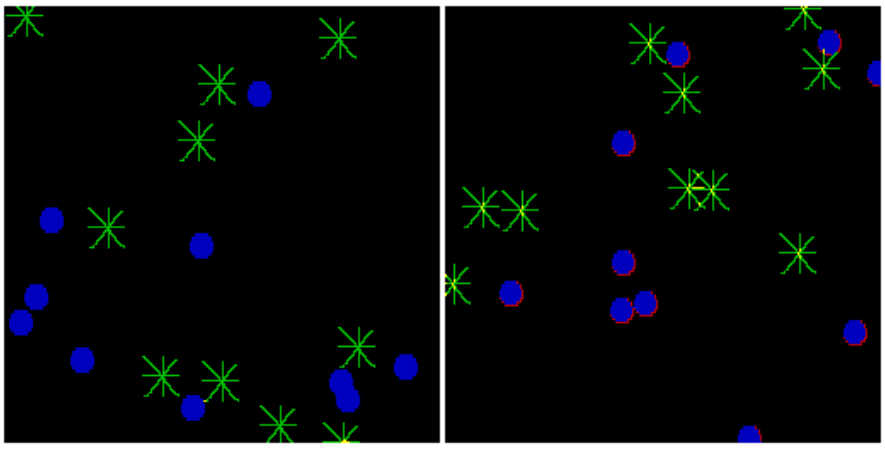


Figura 43 – Combinação entre estrela e círculo, visando realce da estrela. À esquerda, resultado do método 1 e à direita do método 2.

Na combinação com o quadrado, o primeiro método apresenta novamente um resultado quase perfeito, com a mesma dificuldade na sobreposição. Porém neste caso, o segundo método tem um desempenho pior que nos casos anteriores, pois

além de remover o quadrado, remove também boa parte da estrela onde as linhas horizontais e verticais são removidas quase que por inteiro.

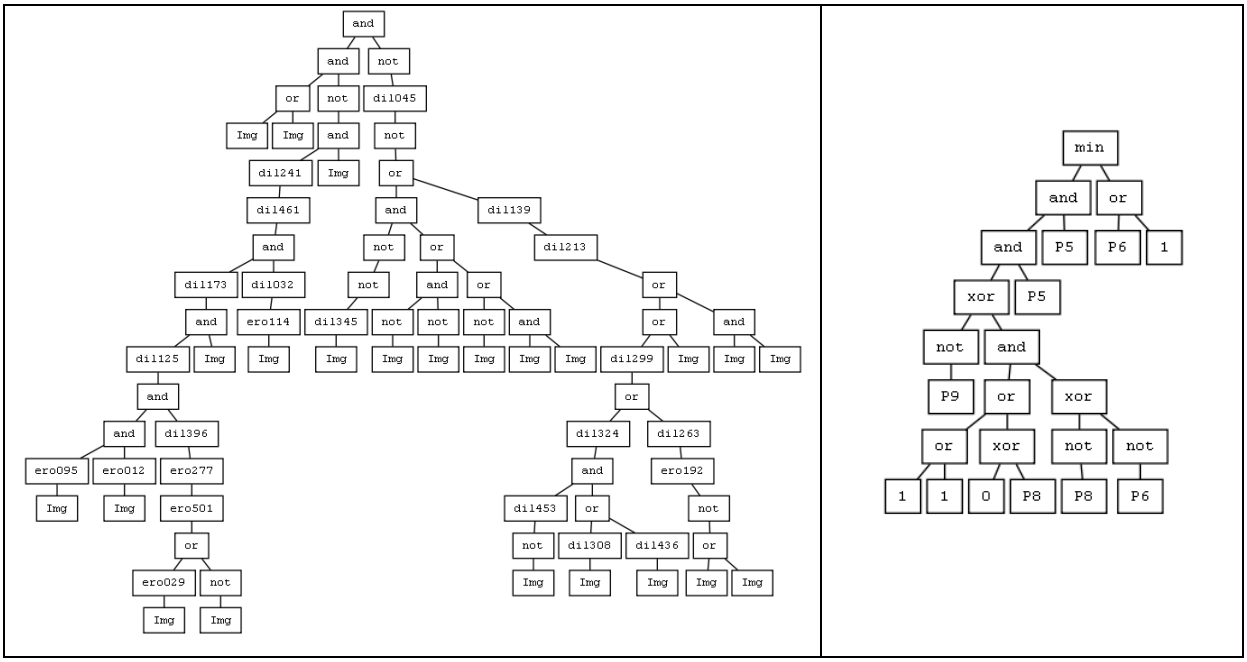


Figura 44 – Árvores resultantes da combinação entre estrela e quadrado, visando realce da estrela. À esquerda, resultado do método 1 e à direita do método 2.

Nessa combinação as árvores da Figura 44 apresentam arvores muito discrepantes quanto à complexidade, entretanto, ao analisar os resultados na Figura 45, o primeiro método apresenta um resultado com qualidade muito superior à do outro método.

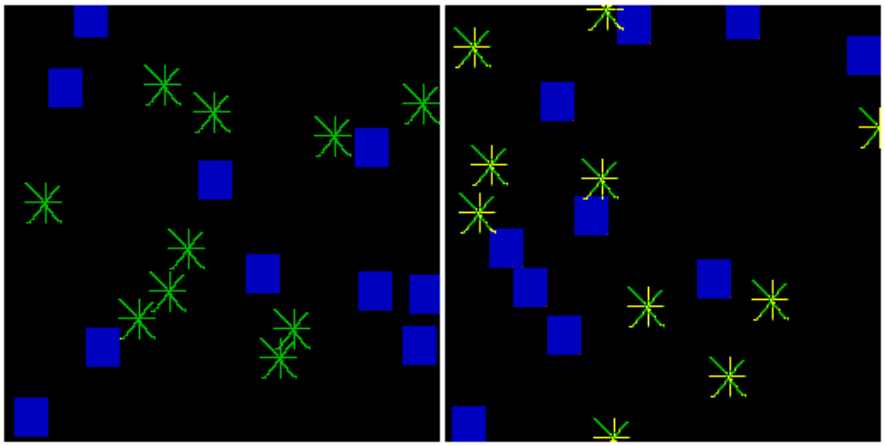


Figura 45 – Combinação entre estrela e quadrado, visando realce da estrela. À esquerda, resultado do método 1 e à direita do método 2.

Ao analisar os *fitness* desses casos, como ilustra a Tabela 3, é possível afirmar que os indivíduos do primeiro método são melhores que os do segundo método, porém com um desempenho mais próximo do que quando comparado com o destaque do anel ou do círculo.

Tabela 3 – Tabela de comparação de *fitness* entre os dois métodos para realce do objeto estrela.

	***Fitness* Método 1**	**%**	***Fitness* Método 2**	**%**
Estrela/Anel	0,001202	99,88	0,007172	99,28
Estrela/Círculo	0,001060	99,89	0,005504	99,45
Estrela/Quadrado	0,001508	99,85	0,008319	99,17

Com isso é possível afirmar que o primeiro método é o mais indicado para realce do elemento anel, apresentando erros apenas onde há sobreposição dos elementos, o que significa que é o resultado ideal, sendo que para a remoção do erro de sobreposição, seria necessário reconstruir a parte sobreposta após a remoção do outro elemento.

4.4.2 Resultados de Remoção

Quando associada ao anel, os dois métodos apresentam bons resultados, a estrela foi removida com sucesso pelo primeiro método sem alteração no anel, porém com o segundo método, parte da borda do anel também foi removida, assim como parte da estrela foi deixada na imagem final.

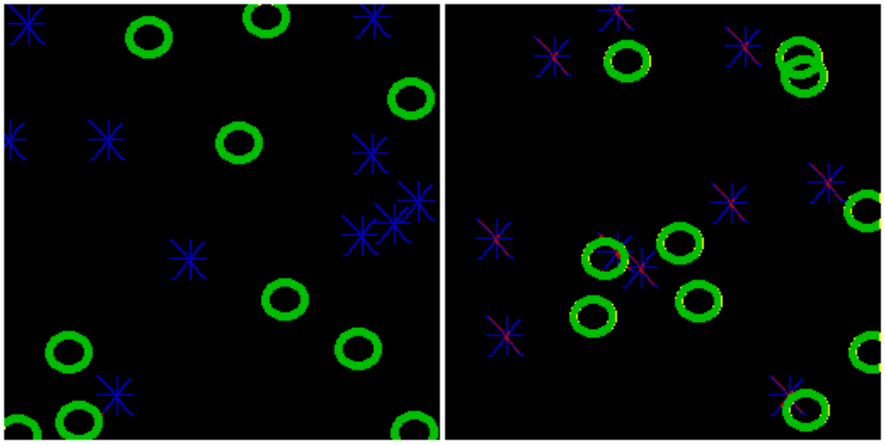


Figura 46 – Combinação entre estrela e anel, visando remoção da estrela. À esquerda, resultado do método 1 e à direita do método 2.

Na associação com o círculo os resultados foram quase ideais. O primeiro método conseguiu o resultado ideal e o segundo método quase ideal, deixando apenas o centro da estrela sem remoção como também removeu parte da borda do círculo, porém o desempenho neste caso é melhor que o apresentado no caso anterior da associação com anel.

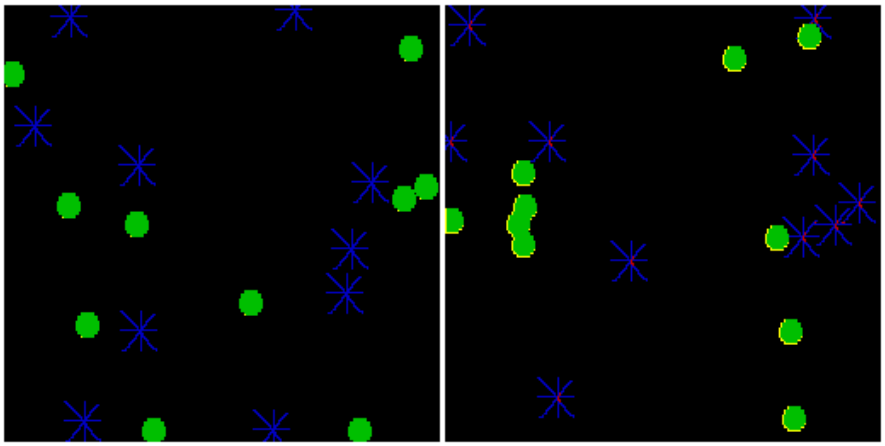


Figura 47 – Combinação entre estrela e círculo, visando remoção da estrela. À esquerda, resultado do método 1 e à direita do método 2.

Em um resultado ainda melhor que os dois anteriores, na associação com o quadrado, os dois métodos apresentaram resultados quase ideais. Porém neste caso, os dois métodos alteraram uma porção ínfima do quadrado, e mais uma vez, o segundo método deixa de remover o centro da estrela.

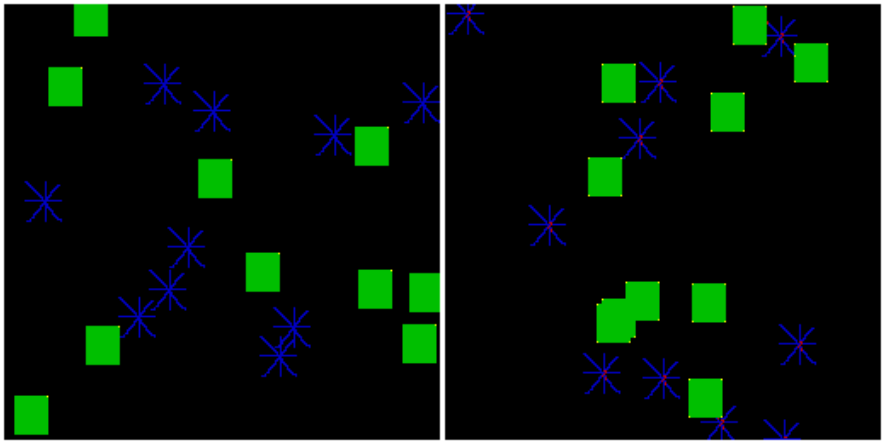


Figura 48 – Combinação entre estrela e quadrado, visando remoção da estrela. À esquerda, resultado do método 1 e à direita do método 2.

Sendo assim, conclui-se que a estrela é um objeto de fácil remoção para ambos os métodos, independente do objeto associado à estrela. Porém o mesmo padrão se repete, onde o primeiro método continua a apresentar melhores resultados quando comparado com o segundo método.

4.5 Resultados utilizando o Objeto Quadrado

4.5.1 Resultados de Destaque

Ao associar ao anel, o quadrado impõe certa dificuldade aos dois métodos. O primeiro método, apesar de remover completamente o anel, danifica um vértice do quadrado em demasia, por outro lado, apesar desse dano, os resultados são melhores do que os apresentados pelo segundo método, onde não só diminui o tamanho geral do quadrado, como também não remove o anel por inteiro.

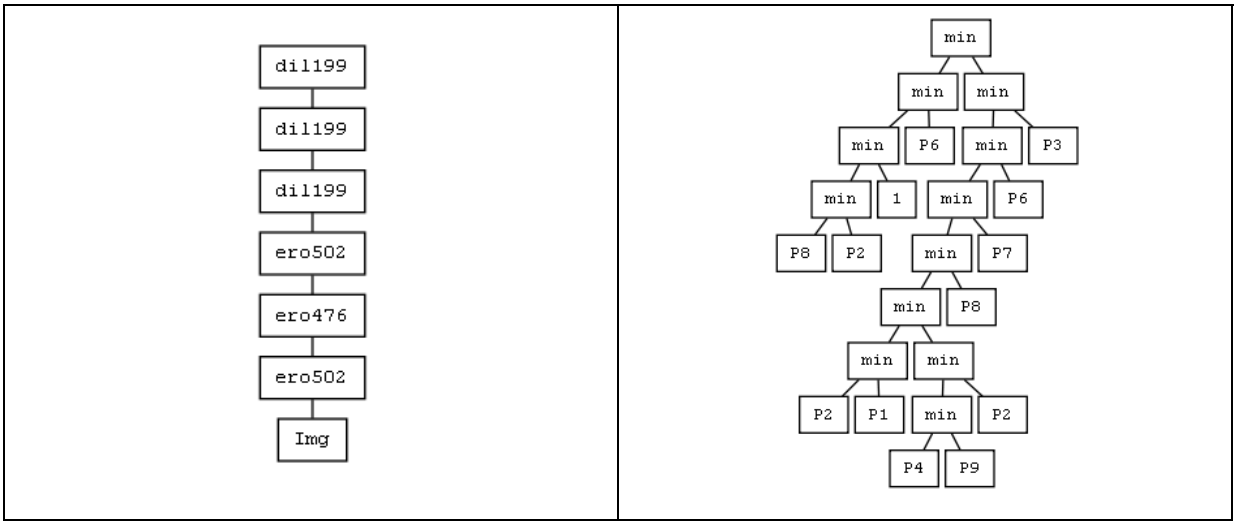


Figura 49 – Árvores resultantes da combinação entre quadrado e anel, visando realce do quadrado. À esquerda, resultado do método 1 e à direita do método 2.

Mais uma vez o método 1 gera uma árvore de menor complexidade, como ilustrado na Figura 49, e com um resultado superior como pode-se observar na Figura 50.

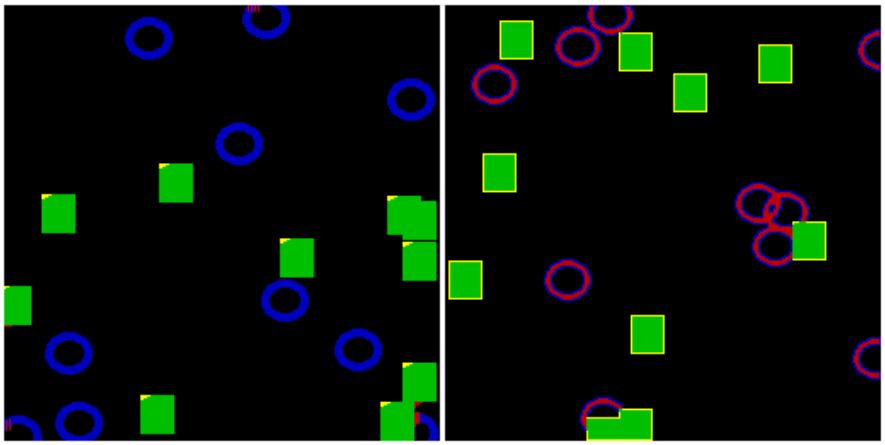


Figura 50 – Combinação entre quadrado e anel, visando realce do quadrado. À esquerda, resultado do método 1 e à direita do método 2.

Quando associado ao círculo, os resultados são piores, sendo que nenhum método consegue apagar por inteiro o círculo, assim como ambos danificam o quadrado. Entretanto, o primeiro método remove mais partes do círculo que o segundo, por outro lado, o segundo danifica o quadrado menos que o primeiro, deixando assim um equilíbrio entre os dois métodos.

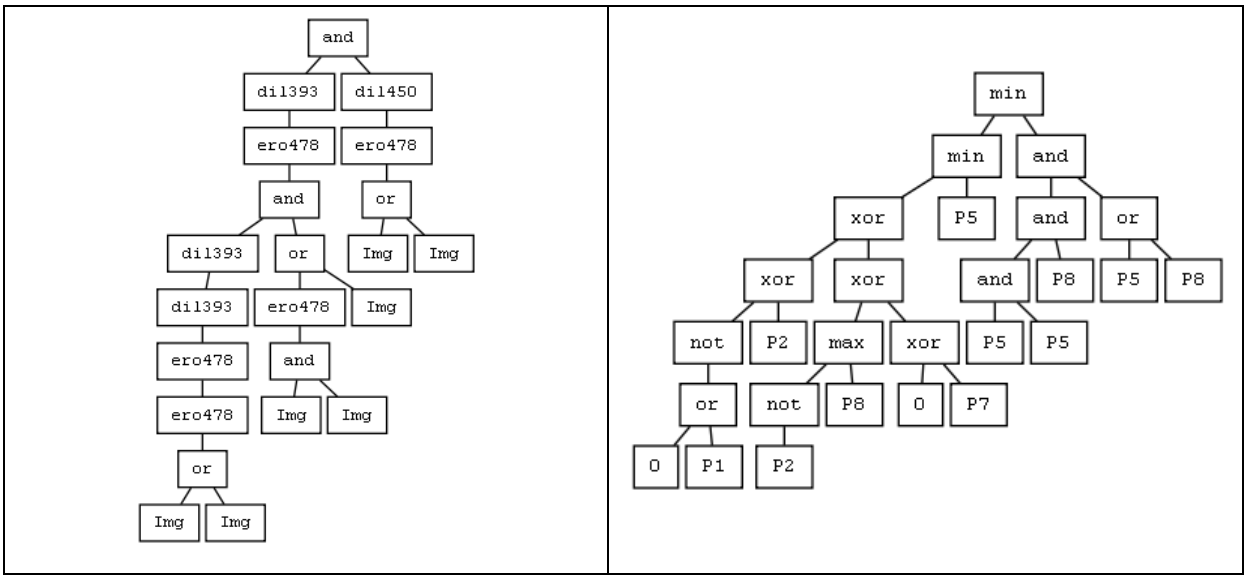


Figura 51 – Árvores resultantes da combinação entre quadrado e círculo, visando realce do quadrado. À esquerda, resultado do método 1 e à direita do método 2.

Neste exemplo, as árvores da Figura 51 estão similares tanto quanto a complexidade, quanto ao resultado gerado que pode ser observado na Figura 52.

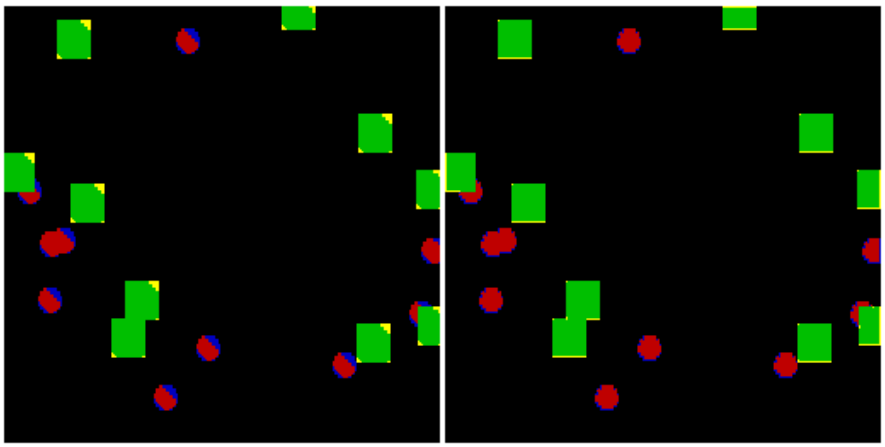


Figura 52 – Combinação entre quadrado e círculo, visando realce do quadrado. À esquerda, resultado do método 1 e à direita do método 2.

O melhor resultado dos dois métodos ocorre na associação do quadrado com a estrela, pois ambos apresentam resultados quase ideais. Mas, mais uma vez, o primeiro método se destaca, tendo em vista que o segundo ainda deixa de apagar parte do centro da estrela.

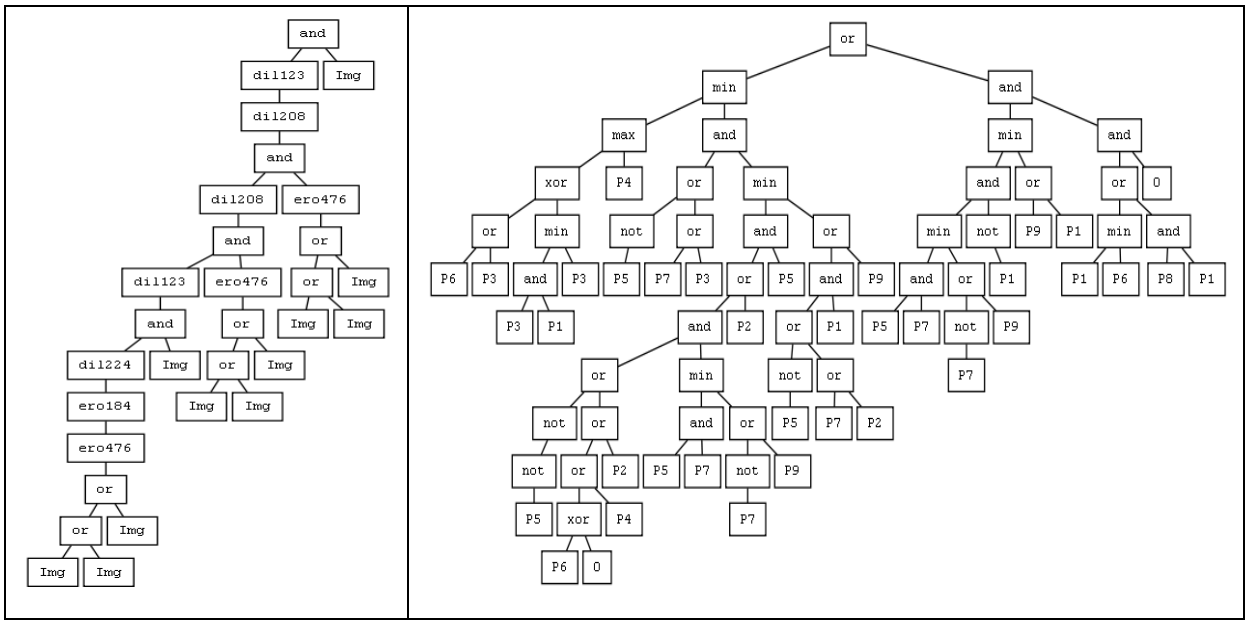


Figura 53 – Árvores resultantes da combinação entre quadrado e estrela, visando realce do quadrado. À esquerda, resultado do método 1 e à direita do método 2.

Na última combinação também foi obtido o equilíbrio entre as árvores (Figura 53) e seus resultados (Figura 54).

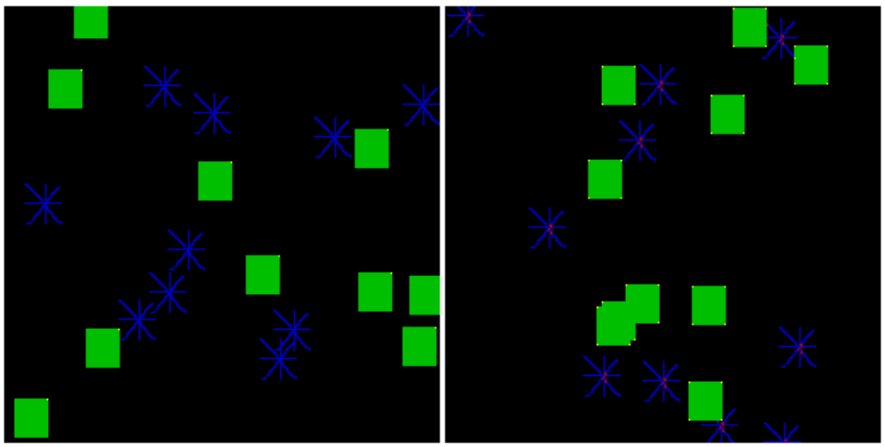


Figura 54 – Combinação entre quadrado e estrela, visando realce do quadrado. À esquerda, resultado do método 1 e à direita do método 2.

Os *fitness* desses casos, como ilustrado a Tabela 4, ilustra o fato de o primeiro método ter melhor desempenho que o segundo. E no caso do anel e da estrela esse desempenho é claramente muito superior ao segundo método.

Tabela 4 – Tabela de comparação de *fitness* entre os dois métodos para realce do objeto quadrado.

	***Fitness* Método 1**	**%**	***Fitness* Método 2**	**%**
Quadrado/Anel	0,004498	99,55	0,032478	96,75
Quadrado/Círculo	0,021042	97,90	0,024124	97,59
Quadrado/Estrela	0,000401	99,96	0,002374	99,76

Neste último objeto analisado, assim como em todos os demais, o primeiro método se mostra superior ao segundo, e mesmo com resultados não tão satisfatórios ainda é o mais indicado para realce do quadrado.

4.5.2 Resultados de Remoção

Combinado com o anel, algumas dificuldades foram apresentadas pelos dois métodos. O primeiro método deixou rastros do quadrado, como também removeu parte do anel, por outro lado o segundo método removeu o quadrado por inteiro, mas também removeu boa parte do anel.

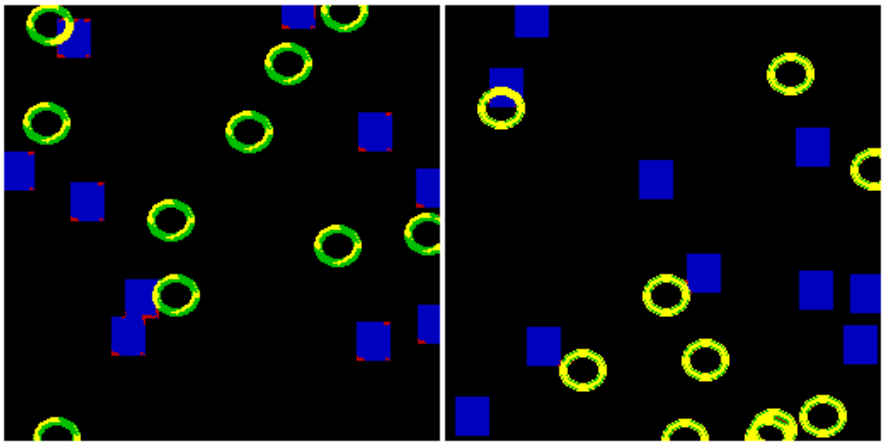


Figura 55 – Combinação entre quadrado e anel, visando remoção do quadrado. À esquerda, resultado do método 1 e à direita do método 2.

Na remoção do quadrado quando associado ao círculo, ambos os métodos apresentaram resultados insatisfatórios. O segundo método removeu os dois objetos da imagem, não deixando quase nada para análise, já o primeiro método removeu completamente os quadrados, porém, deixou parte dos círculos na imagem final, o que caracteriza um melhor resultado.

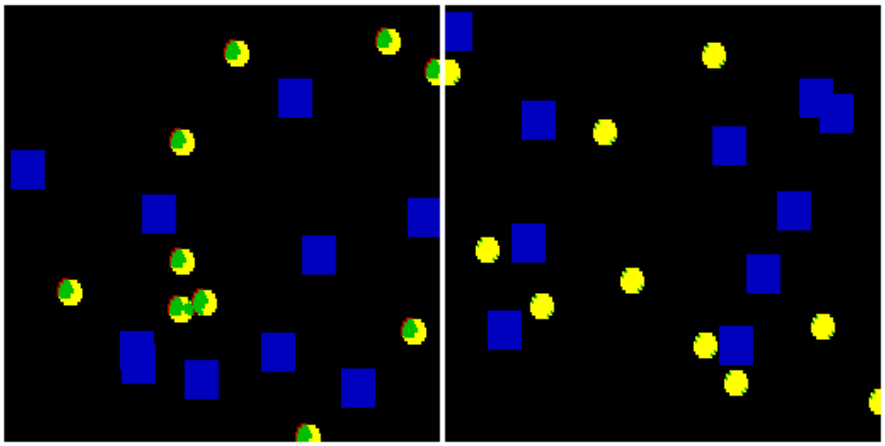


Figura 56 – Combinação entre quadrado e círculo, visando remoção do quadrado. À esquerda, resultado do método 1 e à direita do método 2.

Por fim, ao tentar remover o quadrado quando associado à estrela, ambos os métodos foram eficazes, pois removeram os quadrados por completo e deixaram as estrelas, entretanto o primeiro método deixa as estrelas intactas, já o segundo método, remove as linhas horizontais e verticais das estrelas, descaracterizando o objeto original.

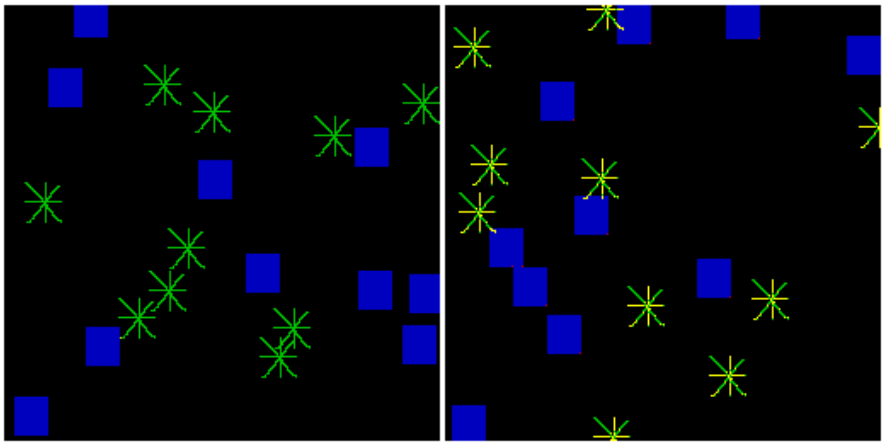


Figura 57 – Combinação entre quadrado e estrela, visando remoção do quadrado. À esquerda, resultado do método 1 e à direita do método 2.

Com base nos dados apresentados nas comparações aos pares de elementos será apresentada nas sessões seguintes gráficos e análises contendo todos os valores de *fitness* apresentados, com o intuito de propiciar uma visão global e facilitar a comparação dos dois métodos como um todo.

4.6 Análise geral de *fitness*

Em uma primeira análise, ao observar a Figura 58 pode-se notar que o primeiro método apresenta o *fitness* com valor menor que o segundo em todos os casos, ou seja, apresenta melhores resultados. Prosseguindo a análise, mesmo considerando o desvio padrão não há sobreposição de *fitness*.

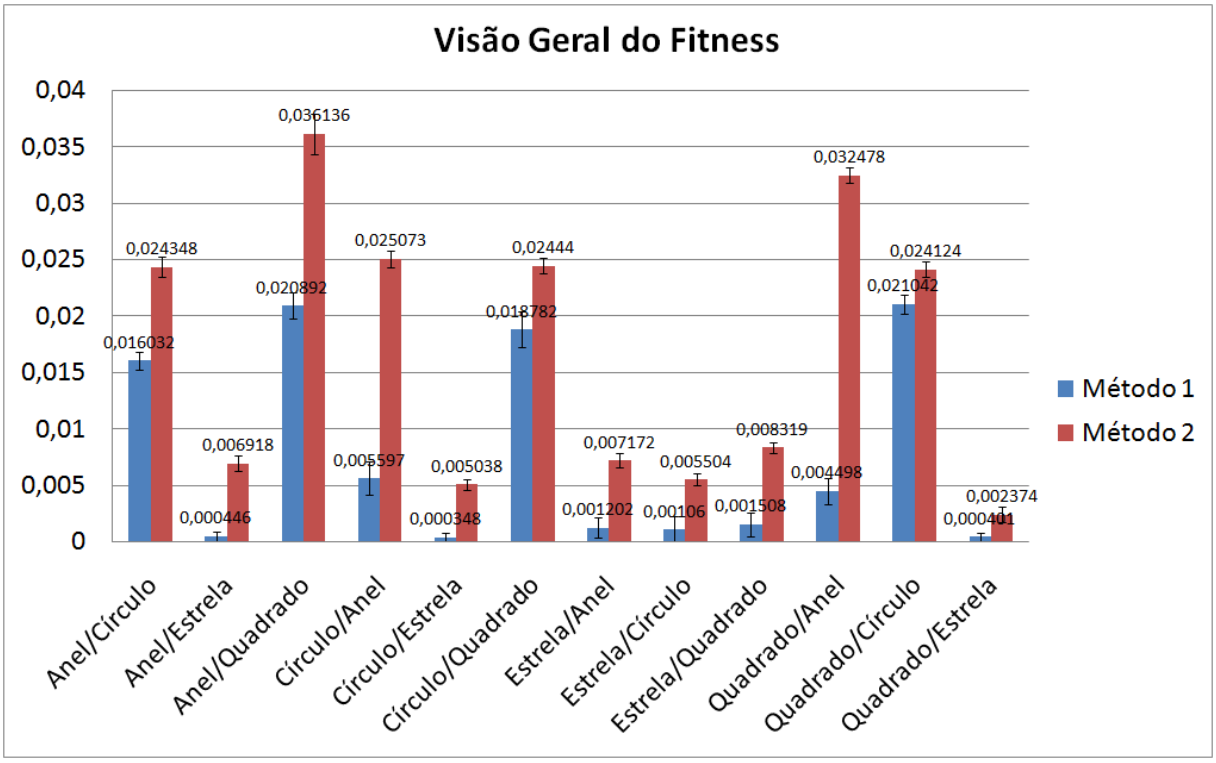


Figura 58 – Visão geral do *fitness*.

Por outro lado, os valores apresentados na Figura 58 estão em uma escala pequena e isso pode influenciar nas observações. Considerando essa possibilidade, outra forma de representação dos *fitness* pode ser feita na forma de porcentagem e, para isso é subtraído o valor do *fitness* de um e o resultado é multiplicado por 100 gerando assim os valores da Figura 59.

Ao analisar esses novos dados, apesar das discrepâncias entre os métodos serem menores, o que foi observado a princípio continua válido mesmo considerando o desvio padrão.

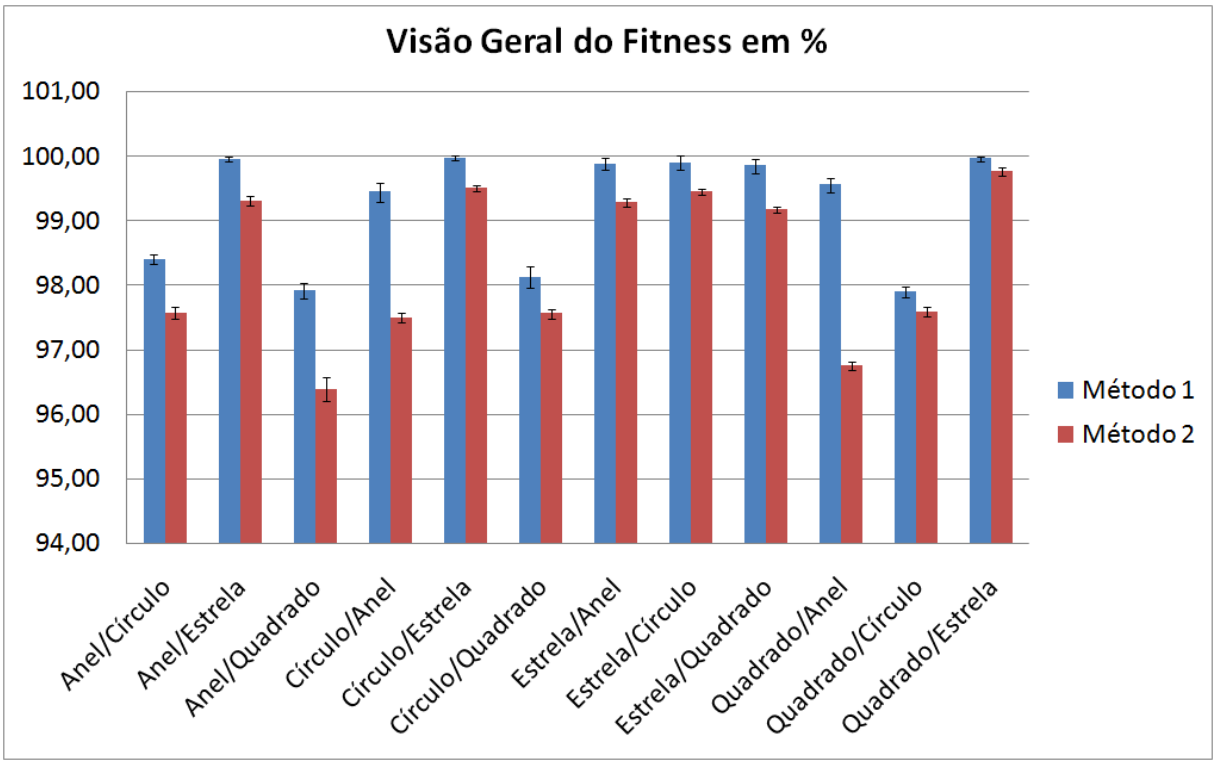


Figura 59 – Visão geral do fitness na forma de porcentagem.

Para verificar a validade das observações, foi feita uma verificação do comportamento do *fitness* com as imagens de controle apresentadas na Figura 60. Essas imagens devem ilustrar como seria o resultado no caso de tentar adivinhar o filtro com um único passo predeterminado.

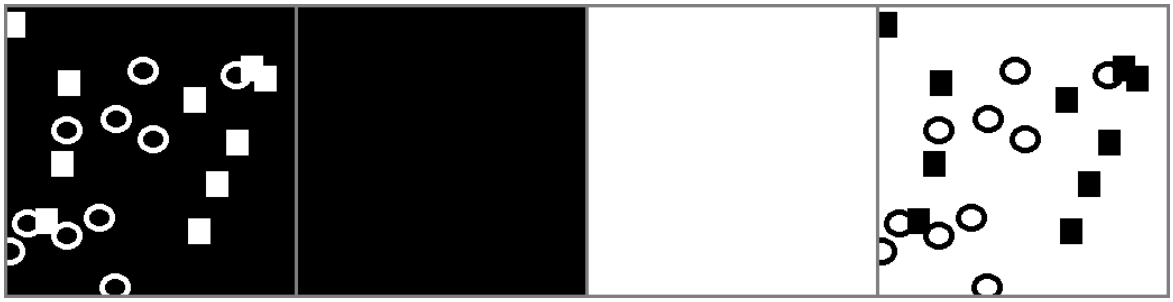


Figura 60 – Exemplos de imagens dos grupos de controle.

A primeira imagem da Figura 60 é a imagem original inalterada, ela testa o caso do filtro não fazer nenhuma alteração; a segunda equivale ao filtro apagar toda a imagem; a terceira preenche a imagem com o brilho máximo; e a última é o inverso da original, onde é apagado o que existe na imagem e o que não existe é preenchido.

Com essas imagens foram calculados os valores de *fitness* para todos os casos. O *fitness* apresentado nos demais gráficos está na forma original sem conversão para porcentagem, ou seja, quanto menor o *fitness* melhor.

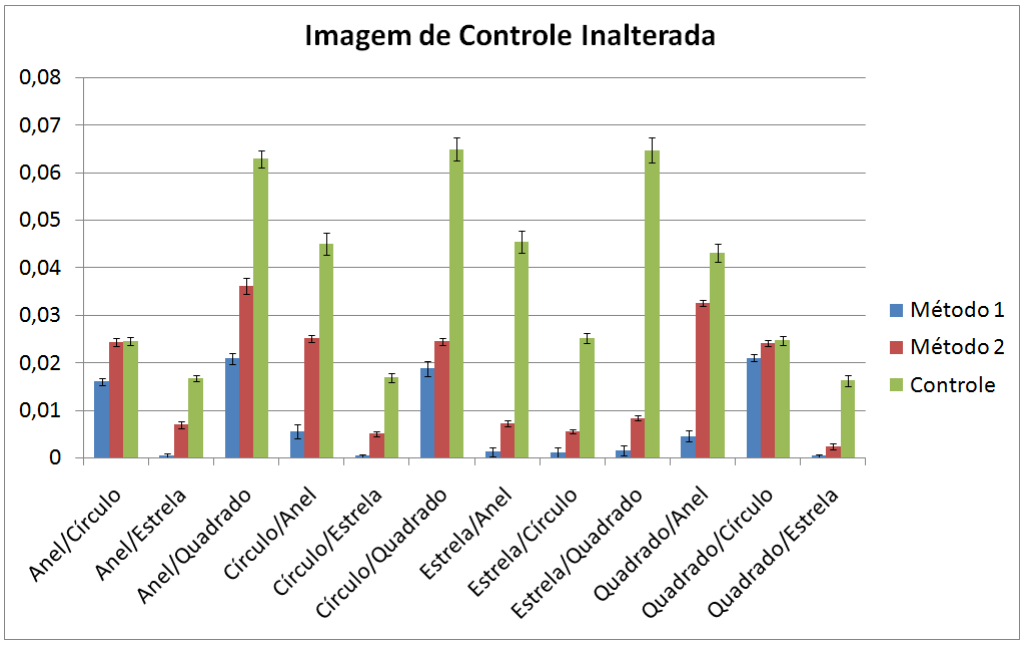


Figura 61 – Visão geral do *fitness* com a imagem de controle inalterada.

Na Figura 61, ao observar o *fitness* da imagem inalterada, percebe-se que em todos os casos a perda é visível quando comparada ao Método 1, mas quando comparado com o Método 2 notam-se casos de valores muito próximos.

Esse é um forte indicativo de que nesses casos o segundo método teve um desempenho muito inferior ao primeiro, mesmo apresentando um valor de *fitness* muito próximo a zero. Como o valor do fitness é muito próximo ao da imagem inalterada, para poupar o processamento de aplicar o filtro seria mais viável manter a imagem original ao invés de aplicar o segundo método nesses casos.

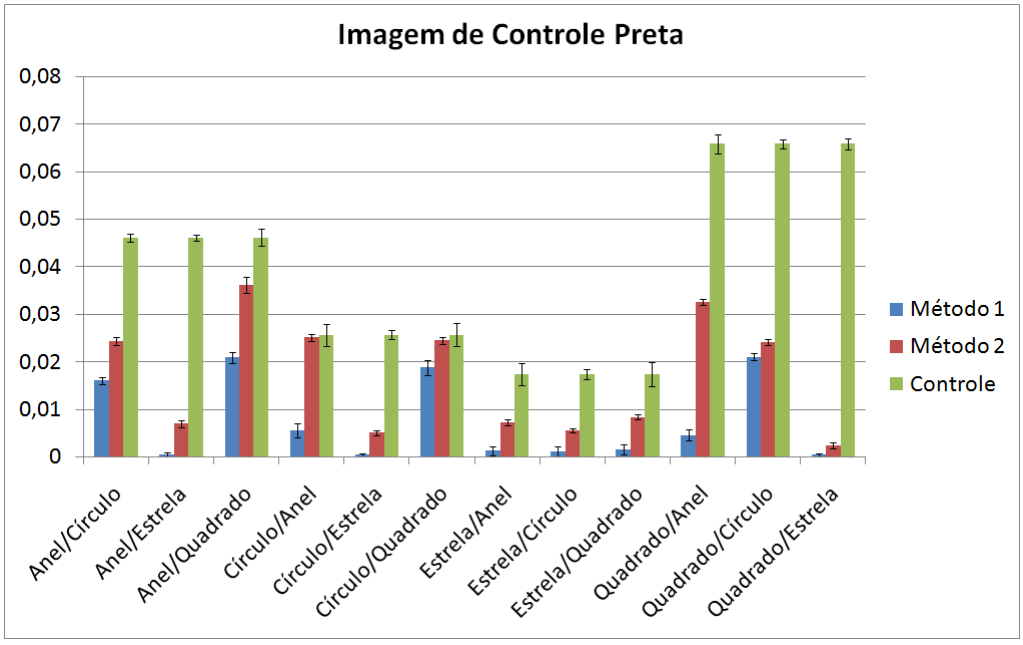


Figura 62 – Visão geral do *fitness* com a imagem de controle preta.

Os dados da Figura 62 apresentam um comportamento similar ao anterior exceto que os valores oscilam menos, porém o segundo método continua a se mostrar inviável ao tentar destacar o círculo quando associado ao anel ou ao quadrado, nesses casos é mais interessante apagar toda a imagem a aplicar o filtro.

Quanto às outras duas imagens de controle, por apresentarem os valores de *fitness* muito discrepantes em relação aos dos métodos, elas serão apresentadas em um mesmo gráfico para apenas demonstrar tal diferença.

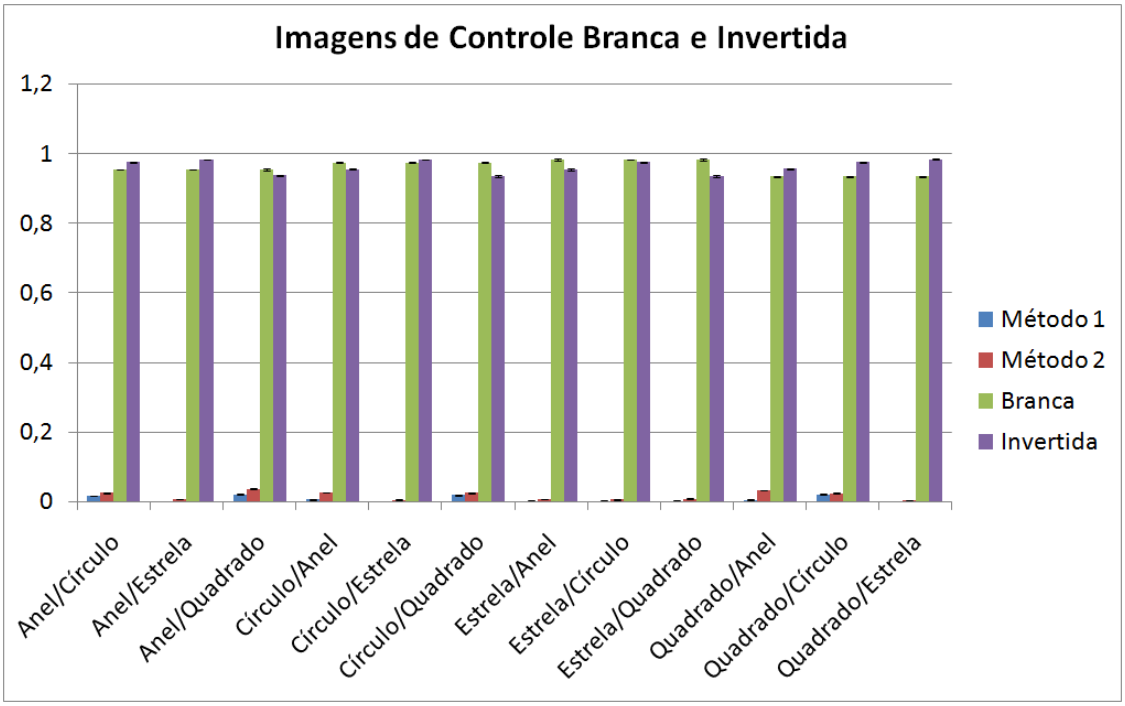


Figura 63 – Visão geral do *fitness* com as imagens de controle branca e invertida.

Analisando tais resultados apresentados na Figura 63, é possível identificar quais são as classes majoritárias e minoritárias das imagens testadas. Quanto mais preto menor o fitness (classe majoritária), quanto mais branco pior (classe minoritária).

Com base nessas análises, ao analisar apenas pelo valor do fitness, o primeiro método é claramente superior ao segundo, porém uma segunda análise foi elaborada para avaliar os resultados obtidos, confirmando ou contestando os resultados até aqui apresentados.

4.7 Análise de Exatidão, Sensibilidade, Especificidade e Precisão

Este tópico da análise com métricas estatísticas tem como intuito propiciar uma visão alternativa para os resultados apresentados anteriormente. A escolha da utilização de métricas estatísticas se deu devido ao fato de serem métricas quantitativas e de uma compreensão possivelmente mais intuitiva.

Essas métricas são baseadas em dados já ilustrados nas figuras deste capítulo, onde serão utilizados para os cálculos os valores: verdadeiros positivos; verdadeiros negativos; falsos positivos; e falsos negativos.

Serão utilizadas as métricas de sensibilidade e especificidade, que são comumente associados a diagnósticos médicos (OBID, 2013), porém também são aplicados em classificadores com dados binários. Por exemplo, tais medidas podem ser utilizadas para analisar o desempenho de um classificador de *spam* como exemplificado por Hastie, Tibshirani e Friedman (2009), assim como serão apresentadas métricas de precisão e exatidão.

Segundo Sammut e Webb (2011), essas métricas são comumente utilizadas para sumarizar o desempenho de classificação dos classificadores. Tais métricas são normalmente obtidas através da elaboração de uma matriz de confusão, que tem como características ser bidimensional e apresentar como índices a verdadeira classe do que deve ser classificado e a classe atribuída pelo classificador.

Para ilustrar melhor como as matrizes de confusão são criadas, será utilizado um exemplo criado por Sammut e Webb (2011). Neste exemplo, são fornecidos 31 objetos a um determinado classificador, e tal classificador deve classificar esses objetos como pertencentes às classes A, B ou C.

Após a execução do classificador, é obtida uma matriz intermediária contendo os resultados do classificador, separando os objetos por sua classe real e classe atribuída pelo classificador, como ilustrado na Tabela 5. Essa matriz intermediária deve conter todas as classes que o classificador é capaz de identificar, para que com base nessa tabela sejam construídas as matrizes de confusão contendo apenas classificações positivas e negativas de acordo com uma única classe que será avaliada.

Tabela 5 – Matriz de Confusão por Classe. Adaptada de (SAMUTT E WEBB, 2011)

		Classe Atribuída pelo Classificador		
		A	B	C
Classe Real do Objeto	A	10	2	1
	B	0	6	1
	C	0	3	8

Com essa matriz, a performance do classificador pode então ser medida para cada classe específica. Por exemplo, essa matriz indica na primeira linha que 13 objetos pertencem a classe A, e desses, 10 foram classificados corretamente pelo classificador como pertencentes a essa classe, os outros três foram classificados incorretamente sendo que dois foram classificados como B e um como C. Com base nessa avaliação é criada a matriz de confusão que será utilizada para o cálculo das métricas.

Essa nova matriz deve conter os dados tendo como referência a classe que será avaliada o desempenho do classificador, ou seja, para se avaliar o classificador quanto a sua capacidade de analisar objetos da classe A, deve-se ter os dados de acerto e erro do mesmo para essa classe. Para a construção da nova matriz é necessário que esses dados estejam discriminados como verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos.

Para ilustrar o raciocínio para criar a nova matriz, foi criada uma nova matriz intermediária que não é obrigatória, mas que serve para facilitar a compreensão do processo. Como temos mais que duas classes, as classes que não estão em evidência (classes B e C) serão unidas em uma nova classe (D), deixando a matriz anterior como na Tabela 6.

Tabela 6 – Matriz Auxiliar para Criação da Matriz de Confusão Final.

		Classe Atribuída pelo Classificador	
		A	D (B+C)
Classe Real do Objeto	A	10	3
	D (B+C)	0	18

No exemplo temos então que para a classe A o classificador classificou corretamente 10 itens como pertencentes à classe A (verdadeiros positivos),

classificou corretamente 18 itens como da classe D (verdadeiros negativos), classificou incorretamente e incorretamente 3 itens da classe A como D (falsos negativos) e não classificou nenhum item incorretamente como A (falsos positivos). Esses dados são de fato os que compõem a matriz de confusão, porém representados com os nomes dos índices alterados para melhor compreensão, sendo a matriz final representada na Tabela 7.

Tabela 7 – Exemplo de Matriz de Confusão.

		Classe Atribuída pelo Classificador	
		Positivo	Negativo
Classe Real do Objeto	Positivo	10	3
	Negativo	0	18

No contexto deste trabalho, podemos identificar quais *pixeis* estão corretamente e incorretamente classificados na imagem final gerada pela árvore. Como explicado no início deste capítulo, o significado de cada cor apresentado nas imagens é referente a valores presentes na matriz de confusão, e com esses valores é possível montar uma tabela para facilitar essa análise, assim relembrando temos:

- Verde: são os verdadeiros positivos (VP), ou seja, os que continuam na imagem e que deveriam continuar.
- Azul: são os verdadeiros negativos (VN), ou seja, os que foram removidos da imagem corretamente.
- Vermelho: são os falsos positivos (FP), ou seja, os que continuam na imagem, mas não deveriam.
- Amarelo: são os falsos negativos (FN), ou seja, os que foram removidos da imagem, mas não deveriam.

E com essa classificação foi elaborada a matriz de confusão na Tabela 8, com as fórmulas dos cálculos das principais métricas representadas. Esses cálculos têm como base os valores da matriz de confusão, e por isso, sua inclusão na tabela facilita relacionar o que cada métrica leva em consideração em sua base de cálculo.

Com esse modelo de tabela, serão apresentados dois exemplos ilustrando um bom e um mau resultado, visando ilustrar como é feito o cálculo e a avaliação dos resultados, porém não serão apresentadas as matrizes de todos os testes, serão apresentados apenas os valores finais das métricas.

Tabela 8 – Cálculo da Matriz de Confusão.

		Objetivo		
		Objeto a ser Destacado	**Objeto a ser Removido**	
Resultado	**Objeto Destacado**	**Destacado Corretamente**	**Destacado Incorretamente (Erro Tipo I)**	**Precisão do Destaque** $= \frac{VP}{(VP+FP)}$
	Objeto Removido	**Removido Incorretamente (Erro Tipo II)**	**Removido Corretamente**	**Precisão da Remoção** $= \frac{VN}{(VN+FN)}$
		Sensibilidade $= \frac{VP}{(VP+FN)}$	**Especificidade** $= \frac{VN}{(VN+FP)}$	**Exatidão** $= \frac{(VP+VN)}{(VP+FP+VN+FN)}$

As definições de cada métrica dadas por Sammut e Webb (2011), foram adaptadas para o contexto da dissertação, visando uma melhor compreensão de cada uma no contexto proposto.

De acordo com Sammut e Webb (2011), a sensibilidade e a especificidade são utilizadas em conjunto para medir o desempenho preditivo de um modelo. No contexto proposto, a sensibilidade indica a fração de objetos destacados corretamente pelo modelo e a especificidade indica a fração de objetos removidos corretamente pelo modelo, ou seja, quanto maior o valor dessas frações melhor é o desempenho do classificador.

Segundo Sammut e Webb (2011), ao invés de utilizar essas duas medidas separadamente, é comum encontrar as duas combinadas em um mesmo coeficiente para medida de performance preditiva. Essa combinação, por não possuir uma nomenclatura específica, aqui denominada SE é feita através do cálculo da multiplicação das duas métricas $\left(\mathrm{SE} = \frac{VP}{(VP+FN)} \times \frac{VN}{(VN+FP)}\right)$.

Para a precisão, independente de ser de remoção ou destaque, Sammut e Webb (2011) definem como a proporção entre as classificações verdadeiras e todas as classificações. Sendo assim a precisão de destaque ou remoção, é a proporção entre o que foi destacado ou removido corretamente e tudo o que foi destacado ou removido.

Sammut e Webb (2011) fornecem ainda outra métrica de desempenho que envolve a combinação da sensibilidade e da precisão de destaque, chamada de medida-F, sendo a mesma obtida pelo cálculo $F_D = \frac{2 \times Sensibilidade \times PrecisãoDeDestaque}{(Sensibilidade + PrecisãoDeDestaque)}$.

Como a sensibilidade e a precisão de destaque estão relacionadas aos valores verdadeiros positivos da matriz de confusão e, a especificidade e a precisão de remoção aos valores verdadeiros negativos, a medida-F poderia ser aplicada também para a remoção através da combinação entre especificidade e precisão de remoção, obtendo assim $F_R = \frac{2 \times Especificidade \times PrecisãoDeRemoção}{(Especificidade + PrecisãoDeRemoção)}$.

Para completar o conjunto de métricas temos a exatidão, que de acordo com Sammut e Webb (2011), indica uma medida para o grau em que as predições do modelo se adéquam à realidade para a qual o mesmo foi modelado.

Nos exemplos apresentados na sequência, será feita uma análise mais detalhada incluindo a matriz de confusão. Entretanto, na análise geral de resultados serão utilizadas apenas as métricas mais completas, sendo elas: SE, F_D, F_R e $exatidão$.

Todos os resultados apresentados, tanto na matriz de confusão quanto nos resultados gerais, serão multiplicados por 100 para indicar a porcentagem de cada métrica, tendo em vista que todas as métricas apresentadas apresentam valores dentro do intervalo $[0,1]$.

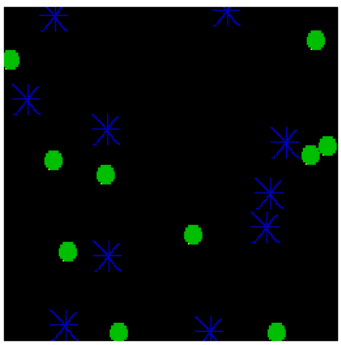


Figura 64 – Exemplo de resultado bom.

A Figura 64 representa um caso onde os objetos que deveriam ser destacados, foram destacados corretamente, assim como os que deveriam ser removidos foram removidos corretamente, os dados estatísticos estão dispostos na Tabela 9.

Tabela 9 – Cálculo da matriz de confusão do caso com resultado bom.

		Objetivo		
		Objeto a ser Destacado	**Objeto a ser Removido**	
Resultado	**Objeto Destacado**	**VP = 1719px**	**FP = 0px**	**Precisão do Destaque** $= \frac{VP}{(VP+FP)} \times 100 = 100,00\%$
	Objeto Removido	**FN = 8px**	**VN = 1135px**	**Precisão da Remoção** $= \frac{VN}{(VN+FN)} \times 100 = 99,30\%$
		Sensibilidade $= \frac{VP}{(VP+FN)} \times 100 = 99,54\%$	**Especificidade** $= \frac{VN}{(VN+FP)} \times 100 = 100,00\%$	**Exatidão** $= \frac{(VP+VN)}{(VP+FP+VN+FN)} \times 100 = 99,72\%$

Analisando os resultados temos a sensibilidade indicando que menos de 0,5% não foi destacado corretamente, o que implica em erro na remoção. A especificidade indica que não houve erro no destaque. Essas duas constatações são confirmadas nas precisões, onde o destaque alcança uma precisão de 100% e a remoção a precisão de 99,30% indicando um erro de 0,70%.

Entretanto, ao analisar a mesma matriz com as outras métricas, os resultados se tornam mais claros e compreensíveis. Ao analisar SE temos $SE = 99{,}54\%$, sendo um primeiro indicativo de que é um bom método para esse caso.

Ao analisar F_D e F_R temos que, $F_D = 99{,}77\%$ indica um bom desempenho de destaque, e com $F_R = 99{,}65\%$ também indicando um bom desempenho de remoção, é possível validar a métrica SE que também se aproxima de 100% confirmando que o método é adequado tanto para destaque quanto para remoção.

Por fim, a exatidão indica a porcentagem de acerto do método considerando o todo, com uma taxa de 99,72%, a métrica acompanha os indicadores anteriores reforçando a qualidade dos resultados concluindo que esse se apresenta um bom método para a remoção e destaque dos objetos da Figura 64.

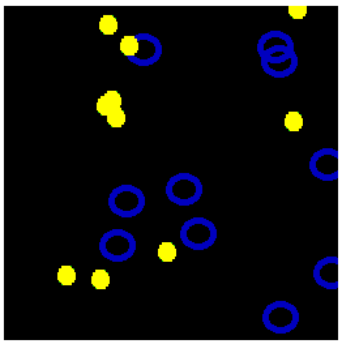


Figura 65 – Exemplo de resultado ruim.

A Figura 65 ilustra um resultado ruim do método onde os objetos que deveriam ser removidos, foram removidos corretamente, entretanto, os objetos que deveriam ser destacados também foram removidos, estando os dados deste exemplo dispostos na Tabela 10.

Tabela 10 – Cálculo da matriz de confusão do caso com resultado ruim.

		Objetivo		
		Objeto a ser Destacado	**Objeto a ser Removido**	
Resultado	**Objeto Destacado**	**VP = 31px**	**FP = 0px**	**Precisão do Destaque** $= \frac{VP}{(VP+FP)} \times 100 = 100,00\%$
	Objeto Removido	**FN = 1573px**	**VN = 3054px**	**Precisão da Remoção** $= \frac{VN}{(VN+FN)} \times 100 = 66,00\%$
		Sensibilidade $= \frac{VP}{(VP+FN)} \times 100 = 1,93\%$	**Especificidade** $= \frac{VN}{(VN+FP)} \times 100 = 100,00\%$	**Exatidão** $= \frac{(VP+VN)}{(VP+FP+VN+FN)} \times 100 = 66,23\%$

Ao analisar os resultados deste exemplo, ao verificar a sensibilidade, vemos que apenas 1,93% foram destacados corretamente, enquanto a especificidade indica que 100% foram removidos corretamente. Em paralelo temos que a precisão de destaque foi de 100%, ou seja, nada foi destacado incorretamente indicando que o que resta na imagem pertence ao objeto a ser destacado, já na precisão da remoção, vemos que 44,00% foram removidos incorretamente.

Seguindo a apresentação das métricas como no primeiro exemplo, ao analisar SE, o valor obtido é $SE = 1{,}93\%$, que é um valor extremamente baixo para um método.

Para validação, ao analisar F_D e F_R os valores obtidos são $F_D = 3{,}79\%$ e $F_R = 79{,}52\%$, o que indica que o método quase não realiza o destaque do objeto,

nas possui uma remoção acima de 50%. Com esses dados, pode-se dizer que SE sofre alta influencia quando é obtido um desempenho ruim tanto em destaque quanto em remoção.

Na conclusão da análise deste exemplo, a exatidão deste caso é de 66,73%, o que indica que a exatidão sofre menor impacto do caso de baixo desempenho do destaque, não prejudicando muito os resultados da remoção.

Com essa análise considerando as quatro métricas sugeridas, pode-se afirmar que elas são complementares. Baseado nos resultados apresentados é possível afirmar que o método é inadequado para a combinação de objetos da Figura 65.

Portanto, por serem métricas complementares que apresentam dados importantes sobre cada aspecto do método, os resultados gerais serão apresentados com as quatro métricas identificadas.

Para a melhor compreensão das métricas, elas serão primeiramente apresentadas individualmente nos casos de estudo deste trabalho para então discutir a avaliação dos resultados de forma mais geral.

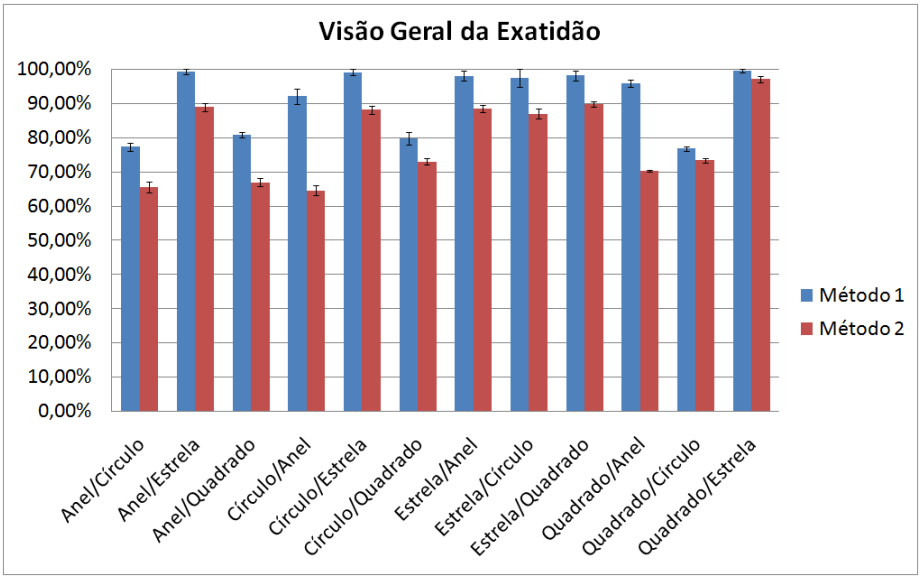


Figura 66 – Visão geral da exatidão.

Os dados apresentados na Figura 66 mostram os dois métodos com comportamentos similares apesar da leve superioridade do primeiro. Quando ocorrem valores elevados os dois são elevados, quando são baixos o mesmo ocorre.

Entretanto isso não ocorre em todas as outras métricas, ao analisar as métricas F_D e F_R o comportamento já começa a sofrer variações em alguns casos onde as diferenças almentam.

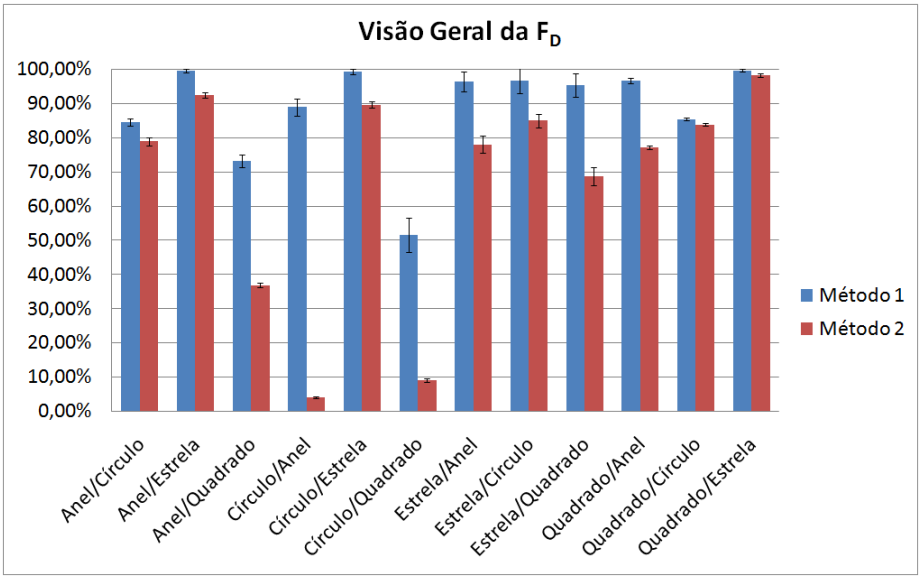


Figura 67 – Visão geral da F_D.

Na Figura 67 a F_D do primeiro método apresenta um desempenho muito superior quando comparado ao segundo, e o mesmo pode ser observado na Figura 68 onde a medida F_R do primeiro método também apresenta melhor desempenho.

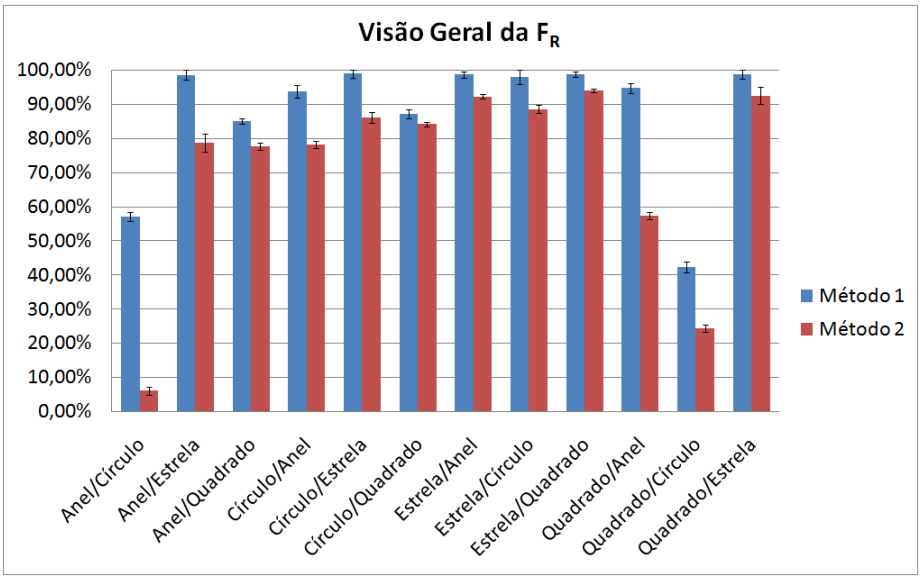


Figura 68 – Visão geral da F_R.

As métricas F_D e F_R apresentam variações em apenas alguns casos, porém ao analisar SE as variações aparecem na maioria dos casos, como é possível observar na Figura 69.

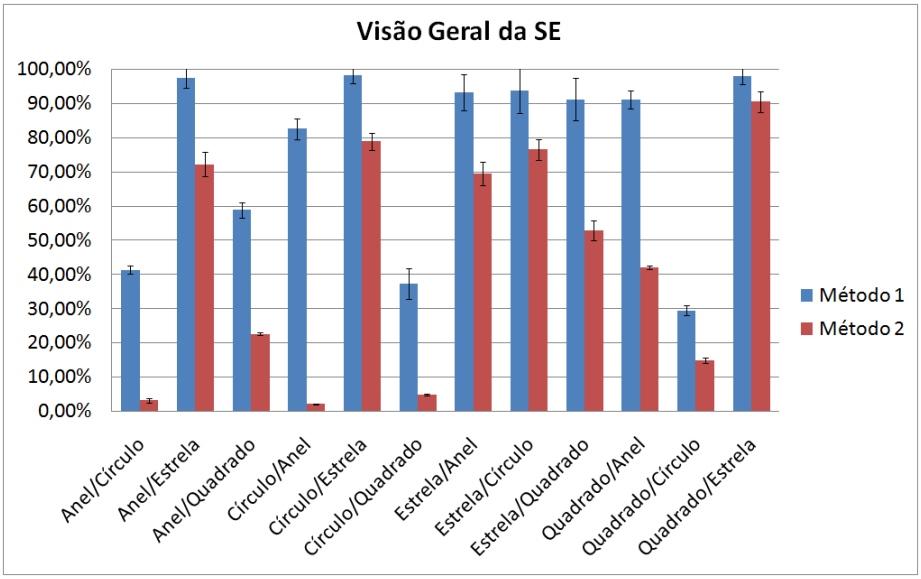


Figura 69 – Visão geral da SE.

Com essa primeira análise, pode-se notar um determinado comportamento de cada métrica. A exatidão é a de maior tolerância, sendo que não sofre muito impacto com erros dos métodos; F_D e F_R são susceptíveis aos erros das respectivas classes (destaque e remoção); já SE é mais penalizada com falhas apresentadas pelo método avaliado.

O comportamento dessas métricas fica mais claro quando analisadas em conjunto. Na Figura 70 estão dispostas todas as métricas citadas com os valores referentes ao primeiro método.

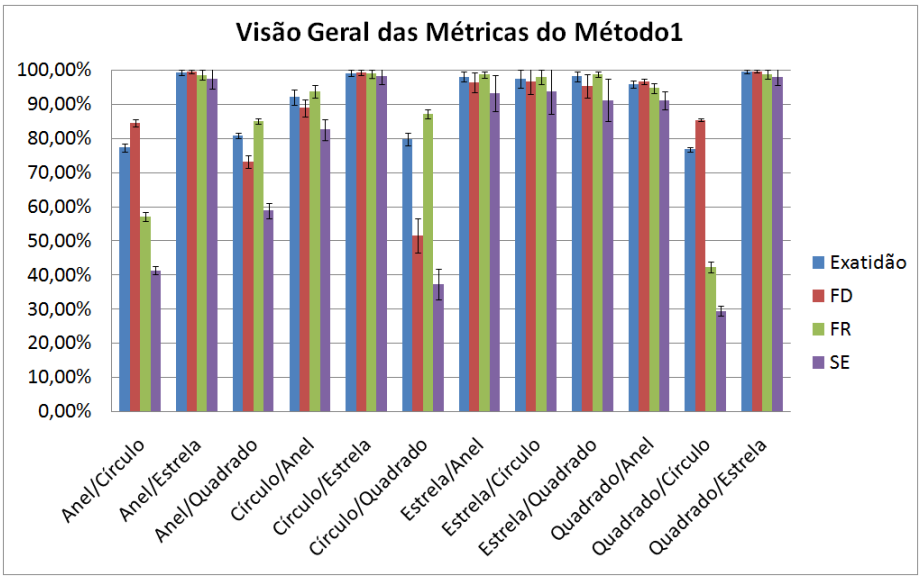


Figura 70 – Visão geral das métricas estatísticas para o primeiro método.

Ao analisar os dados apresentados na Figura 70, o padrão discutido anteriormente fica mais claro onde, a exatidão tem sempre o valor pouco abaixo do

maior F independente de ser F_D ou F_R, já SE tem o comportamento inverso da exatidão onde seu valor é sempre inferior ao F mais baixo independente de ser F_D ou F_R.

Como o primeiro método apresenta valores muito altos em algumas combinações, essas observações ficam nítidas ao observar as métricas do segundo método apresentadas na Figura 71.

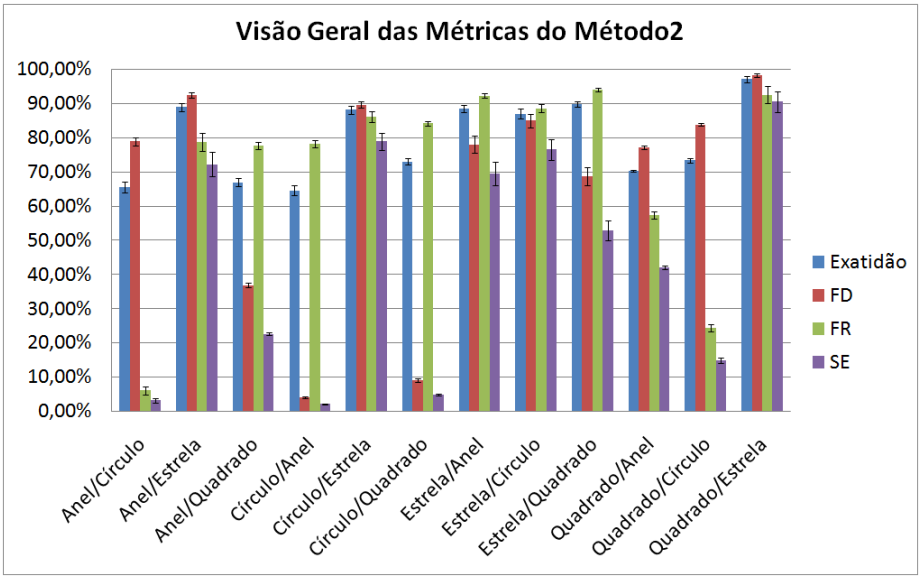


Figura 71 – Visão geral das métricas estatísticas para o segundo método.

Observando os dados da Figura 71, fica claro que a exatidão acompanha o F de valor mais alto e SE o F de menor valor. Com isso algumas aplicações para essas métricas podem ser propostas.

As métricas F_D e F_R só são úteis no caso de apenas ser relevante o desempenho de destaque ou da remoção. Já a exatidão tem uma boa aplicação quando é indiferente se o destaque ou a remoção sejam bons desde que um seja bom, isso ocorre pelo fato da exatidão acompanhar o F mais alto, o que garante que pelo menos um será bom. Por fim SE pode ser utilizado quando é necessário que tanto o destaque quanto a remoção tenham desempenhos bons, sendo que SE somente apresentará um valor alto se o menor F possuir também um valor elevado.

Complementando a analise estatística foi avaliado o comportamento de tais métricas utilizando as imagens de controle. Para ilustrar como são analisadas as imagens de controle, elas estão ilustradas na Figura 72 utilizando o esquema de cores já utilizado ao longo das análises.

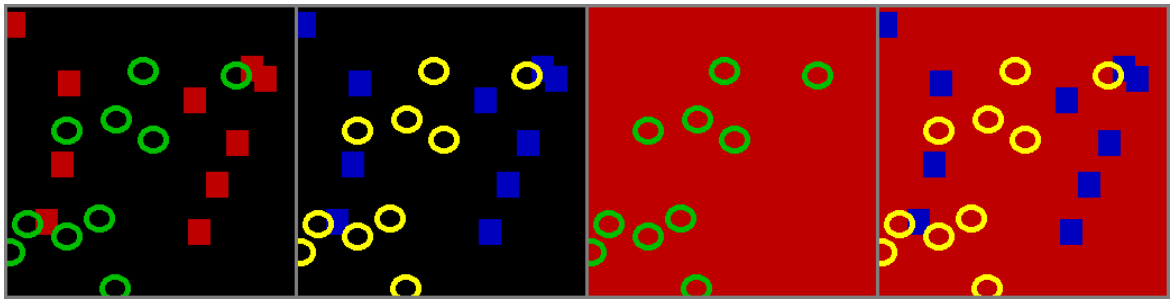


Figura 72 – Exemplos de imagens dos grupos de controle utilizando esquema de cores.

A primeira imagem da Figura 72 é a imagem original inalterada, ela acerta no destaque e erra na remoção; a segunda equivale ao filtro apagar toda a imagem, nela o filtro acerta na remoção e erra no destaque; a terceira preenche a imagem com o brilho máximo acertando o destaque e errando todo o resto; e a última é o inverso da anterior, onde acerta na remoção e erra todo o resto.

Ao analisar a exatidão com base nas imagens de controle inalterada e preta, é possível encontrar em alguns casos valores próximos aos valores encontrados através da execução dos métodos.

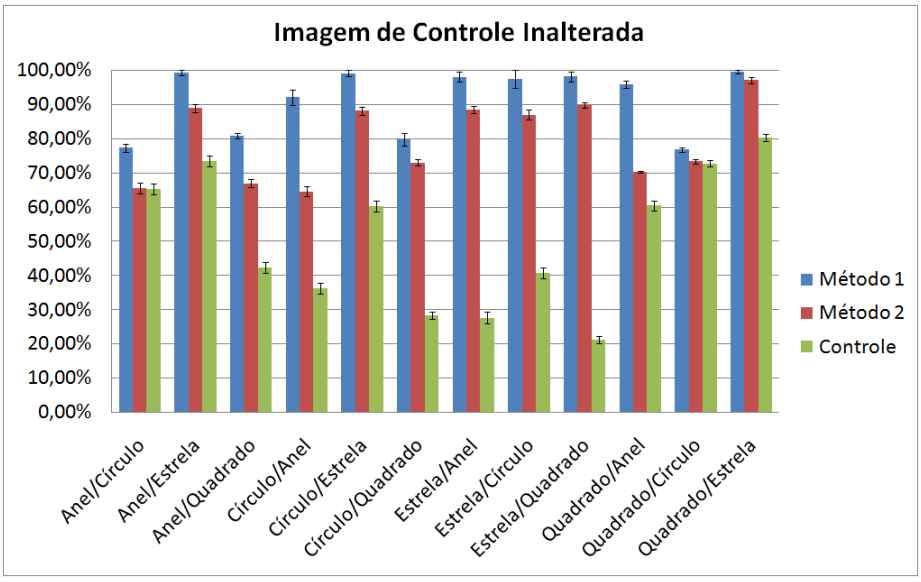


Figura 73 – Visão geral da exatidão com a imagem de controle inalterada.

Na Figura 73, com a imagem de controle inalterada a exatidão do primeiro método está acima das outras duas métricas e em alguns casos o segundo método tem o valor muito próximo ao controle.

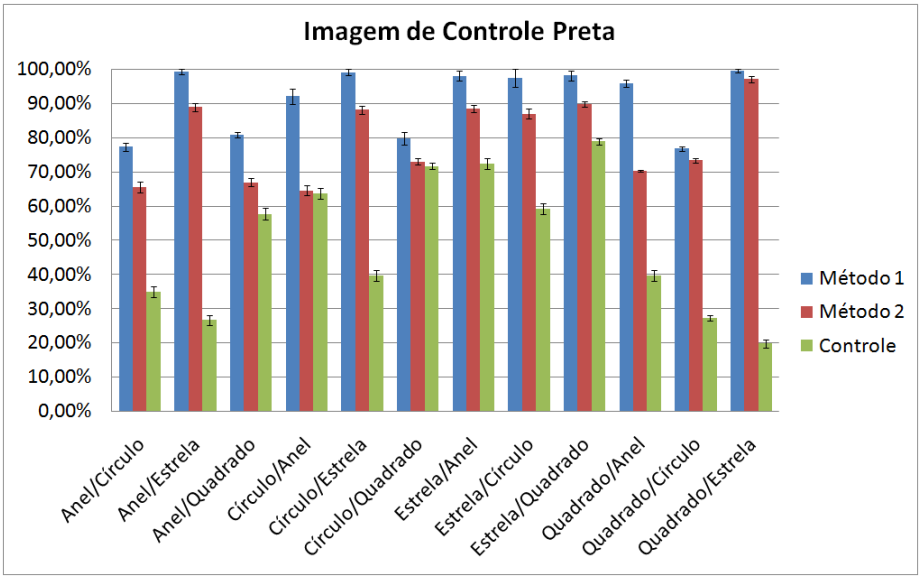


Figura 74 – Visão geral da exatidão com a imagem de controle preta.

O comportamento da exatidão se repete na Figura 74 com a imagem de controle preta. Porém, como analisado anteriormente, a exatidão acompanha a medida F de maior valor e sendo assim mesmo com o segundo método apresentado valores próximos ao controle, a análise das outras métricas se faz necessária antes de qualquer conclusão.

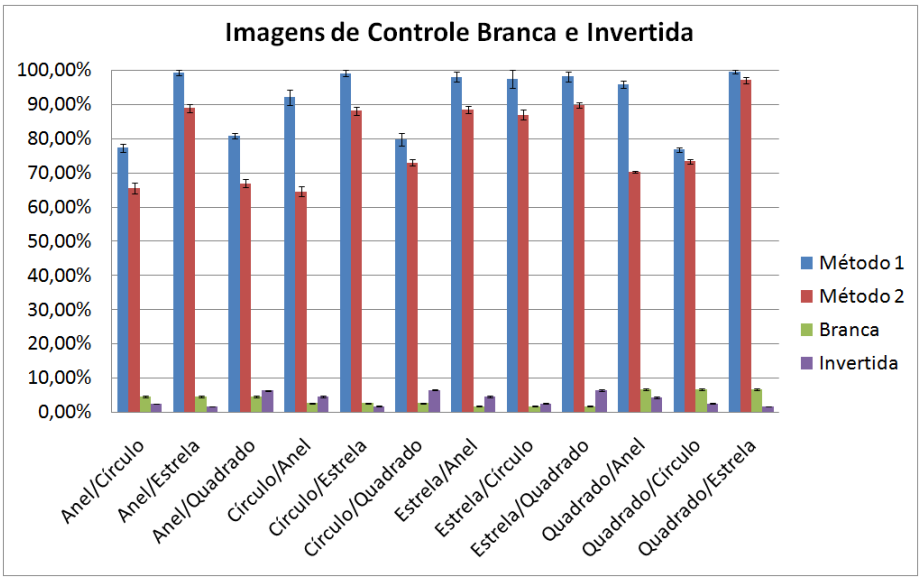


Figura 75 – Visão geral da exatidão com as imagens de controle branca e invertida.

Já Figura 75 ilustra o mesmo comportamento das imagens de controle branca e invertida, quando analisadas junto ao *fitness* na Figura 63. Isso pode ser considerado um forte indicador de que as imagens de controle branca e invertida não são adequadas ao problema.

Em relação à medida F, como já observado na Figura 72, cada imagem de controle apresenta uma alta taxa ou no destaque ou na remoção, mas não em ambos. O impacto desse comportamento na medida F está diretamente relacionado à detecção ou remoção que não apresenta nenhum acerto, sendo que onde não há acerto a medida F apresenta valor zero, ou seja, se a remoção não obteve nenhum pixel de acerto a medida F_R tem também valor zero.

Com essa observação, as imagens de controle inalterada e branca que não apresentam acerto na remoção não possuem valores de F_R diferentes de zero. O mesmo ocorre com as imagens de controle preta e invertida, porém em relação ao destaque, obtendo assim F_D igual a zero. Por esse motivo, os gráficos referentes às medidas F de valor zero não foram apresentados.

Além desses os gráficos, os com as figuras de controle branca e invertida que apresentam valores muito baixos, como no *fitness* e na exatidão, também foram omitidos.

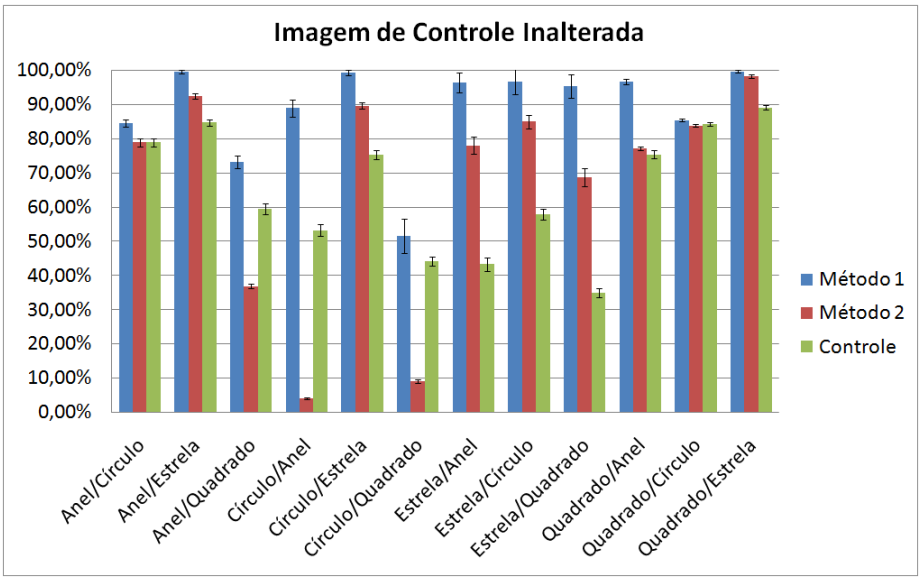


Figura 76 – Visão geral da medida F_D com a imagem de controle inalterada.

Ao comparar a medida F_D dos métodos com o controle na Figura 76, o resultado do controle se destaca por ter valores superiores a alguns casos do segundo método. Porém, isso não indica que o segundo método deve ser descartado, considerando que a imagem de controle não apresenta nenhum acerto na medida F_R.

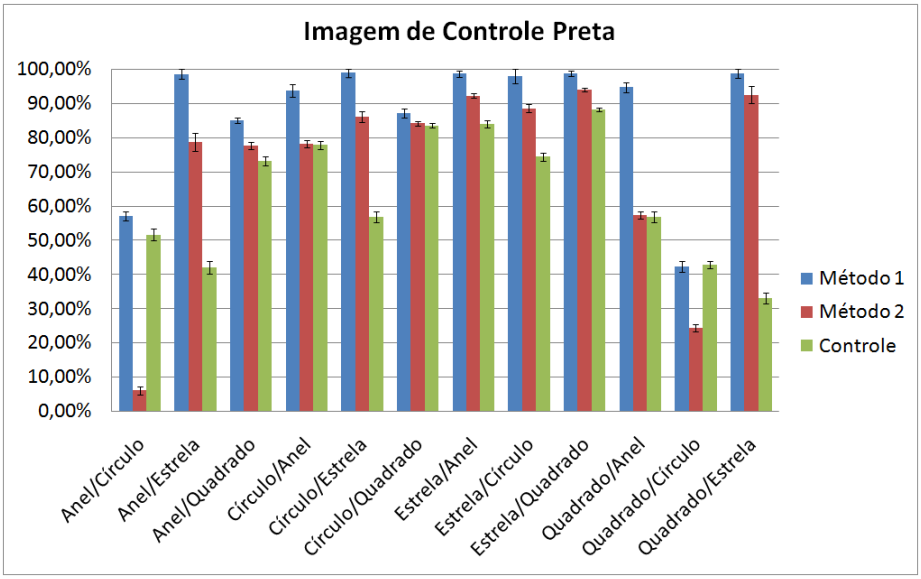


Figura 77 – Visão geral da medida FR com a imagem de controle preta.

Na Figura 77, o mesmo comportamento ocorre e com a mesma conclusão pelo desempenho na medida F_D. Mas essas duas observações em relação às medidas F_D e F_R servem para reforçar que o primeiro método tem desempenho superior.

Em relação à métrica SE, como ela está diretamente relacionada às medidas F_D e F_R ao mesmo tempo estando sempre abaixo da que apresenta o valor mais baixo e, como em todas as imagens de controle uma ou outra tem valor zero, em todos os casos de controle SE tem valor zero. Isso porque SE é uma composição de sensibilidade e especificidade e ambas são compostas por valores de destaque e remoção, e quando um desses valores zera, a métrica também obtém valor zero.

Com base nessa avaliação, podemos reafirmar o que foi discutido anteriormente neste capitulo, sendo que aqui é reforçada a hipótese do primeiro método avaliado ser melhor que o segundo em todos os casos testados, independentemente da métrica avaliada.

4.8 Análise de resultados

Tanto na análise por *fitness* quanto pelas métricas obtidas pela matriz de confusão, o primeiro método apresenta resultados melhores. O que indica que a o primeiro método apresenta um melhor desempenho que o segundo no cenário proposto.

Em outra análise, não foi encontrada uma métrica obtida na matriz de confusão que acompanhasse proporcionalmente o *fitness* obtido, entretanto é possível observar que a métrica escolhida para o cálculo de *fitness* foi adequada ao problema proposto.

As métricas estatísticas apresentaram um alto nível de detalhamento no comportamento de cada método, o que indica que possam ser utilizadas para análise em outros casos. Porém, um estudo mais aprofundado se faz necessário.

Com relação aos valores obtidos, o caso em que os resultados de exatidão estiveram mais próximos foi na combinação de destaque do quadrado em conjunto com a estrela, tendo o primeiro método um desempenho melhor em 2,41% de diferença. No caso mais discrepante, destaque do círculo quando associado ao anel, o primeiro método apresentou resultados 27,57% melhores.

Na métrica SE, o mais próximo também está na combinação de destaque do quadrado em conjunto com a estrela, sendo a diferença igual a 7,41% a favor do primeiro método. E no caso de maior discrepância, também no destaque do círculo quando associado ao anel, o primeiro método teve um resultado 80,51% melhor.

Sendo possível também, com base nessas métricas, afirmar que o primeiro é viável para essa classe de problema, e não somente apresenta melhores resultados, uma vez que na maior parte dos casos o método obteve valores elevados. Sendo que na exatidão, todos valores estão acima de 76%; Em relação à métrica SE apenas quatro dos doze casos está abaixo 82%, sendo que destes um ainda se encontra acima da faixa de 50%.

Capítulo 5

CONCLUSÕES

Os objetivos foram amplamente alcançados visto que foi estabelecida uma forma de comparação sistemática de dois métodos da literatura que possibilitou a identificação e distinção de desempenho entre os dois.

Após a execução das analises quantitativa e qualitativa dos testes, foi possível obter uma quantidade significativa de dados que auxiliassem na avaliação dos métodos testados.

Por terem sido analisados com métricas quantitativas, é possível afirmar que o primeiro método (proposto por Pedrino et al. (2011a)), realmente apresenta um melhor desempenho em relação ao método proposto por Wang, Chen e Lee (2008).

Este trabalho também apresentou um levantamento em relação às ferramentas para utilização da programação genética, sendo o mesmo uma referência para trabalhos futuros que desejam utilizar programação genética.

5.1 Trabalhos Futuros

O trabalho deixa em aberto para trabalhos futuros, a investigação da possibilidade de utilizar métricas como, exatidão, especificidade, sensibilidade e precisão, ou uma composição das mesmas já na execução da programação genética como medida de *fitness*.

Entretanto, novos trabalhos podem explorar também, o comportamento do método (aqui apresentado como adequado) em casos que apresentem um nível de complexidade mais elevado. Por exemplo, com a combinação de mais elementos ao

mesmo tempo, ou seja, utilizar o destaque de dois objetos, ou talvez a remoção de dois e assim por diante, podendo assim tentar identificar até qual nível de complexidade o método é válido.

Outra análise de comportamento do método que pode ser avaliada está na utilização de objetos e/ou elementos estruturantes com outras dimensões, ou ainda validação do mesmo com imagens em escala cinza ou colorida..

REFERÊNCIAS BIBLIOGRÁFICAS

AL-MADI, N.; LUDWIG, S. **Adaptive genetic programming applied to classification in data mining**. In: Nature and Biologically Inspired Computing (NaBIC), 2012 Fourth World Congress on. [S.l.: s.n.], 2012. p. 79-85.

AL-SAHAF, H.; SONG, A.; NESHATIAN, K.; ZHANG, M. **Two-tier genetic programming: towards raw pixel-based image classification**. Expert Systems with Applications, v. 39, n. 16, p. 12291-12301, 2012. ISSN 0957-4174.

ALAVI, A. H.; GANDOMI, A. H. **Energy-based numerical models for assessment of soil liquefaction**. Geoscience Frontiers, v. 3, n. 4, p. 541-555, 2012. ISSN 1674-9871.

ALESHUNAS, J.; JANIKOW, C. **Cost-benefit analysis of using heuristics in acgp**. In: Evolutionary Computation (CEC), 2011 IEEE Congress on. [S.l.: s.n.], 2011. p. 1172-1178. ISSN Pending.

ALHEJALI, A.; LUCAS, S. **Evolving diverse ms. pac-man playing agents using genetic programming**. In: Computational Intelligence (UKCI), 2010 UK Workshop on. [S.l.: s.n.], 2010. p. 1-6.

ASLAM, M. W.; ZHU, Z.; NANDI, A. K. **Automatic modulation classification using combination of genetic programming and knn**. Wireless Communications, IEEE Transactions on, v. 11, n. 8, p. 2742-2750, 2012. ISSN 1536-1276.

BANZHAF, W.; NORDIN, P.; KELLER, R. E.; FRANCONE, F. D. **Genetic Programming : An Introduction : On the Automatic Evolution of Computer Programs and Its Applications (The Morgan Kaufmann Series in Artificial Intelligence)**. [S.l.]: Morgan Kaufmann Publishers, 1997. Hardcover. ISBN 155860510X.

BEHBAHANI, S.; SILVA, C. de. **Mechatronic design evolution using bond graphs and hybrid genetic algorithm with genetic programming**. Mechatronics, IEEE/ASME Transactions on, v. 18, n. 1, p. 190-199, 2013. ISSN 1083-4435.

BESARI, A. R. A.; PRABUWONO, A. S.; ZAMRI, R.; PALIL, M. D. M. **Computer vision approach for robotic polishing application using artificial neural networks**. In: Research and Development (SCOReD), 2010 IEEE Student Conference on. 2010. p. 281-286.

BHATTACHARYA, J.; MAJUMDER, S.; SANYAL, G. **The gaussian maxima filter (gmf)**: A new approach for scale-space smoothing of an image. In: India Conference (INDICON), 2010 Annual IEEE. 2010. p. 1-4.

BHOWAN, U.; ZHANG, M.; JOHNSTON, M. **Genetic programming for image classification with unbalanced data**. In: Image and Vision Computing New Zealand, 2009. IVCNZ '09. 24th International Conference. 2009. p. 316-321. ISSN 2151-2205.

BIRLA, M. **Fpga based reconfigurable platform for complex image processing**. In: Electro/information Technology, 2006 IEEE International Conference on. 2006. p. 204-209.

BORG, C.; ROSNER, M.; PACE, G. **Evolutionary algorithms for definition extraction**. In: Proceedings of the 1st Workshop on Definition Extraction. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009. (WDE '09), p. 26-32. ISBN 978-954-452-013-7.

BOZORGTABAR, B.; NOORIAN, F.; RAD, G. **Comparison of different pca based face recognition algorithms using genetic programming**. In: Telecommunications (IST), 2010 5th International Symposium on. [S.l.: s.n.], 2010. p. 801-805.

CASTELLI, M.; MANZONI, L.; SILVA, S.; VANNESCHI, L. **A comparison of the generalization ability of different genetic programming frameworks**. In: Evolutionary Computation (CEC), 2010 IEEE Congress on. [S.l.: s.n.], 2010. p. 1-8

CASTELLI, M.; VANNESCHI, L.; SILVA, S. **Semantic search-based genetic programming and the effect of intron deletion**. Cybernetics, IEEE Transactions on, PP, n. 99, p. 1-11, 2013. ISSN 2168-2267.

CHAUDHARY, U.; IQBAL, M. **Determination of optimum genetic parameters for symbolic non-linear regression-like problems in genetic programming**. In: Multitopic Conference, 2009. INMIC 2009. IEEE 13th International. [S.l.: s.n.], 2009. p. 1-5.

CHEN, H.-M.; KAO, W.-K.; TSAI, H.-C. **Genetic programming for predicting aseismic abilities of school buildings**. Engineering Applications of Artificial Intelligence, v. 25, n. 6, p. 1103-1113, 2012. ISSN 0952-1976.

CHIN, R. T.; HARLOW, C. A. **Automated visual inspection**: A survey. Pattern Analysis and Machine Intelligence, IEEE Transactions on, PAMI-4, n. 6, p. 557-573, nov. 1982. ISSN 0162-8828.

COELHO, A. L.; FERNANDES, E.; FACELI, K. **Inducing multi-objective clustering ensembles with genetic programming**. Neurocomputing, v. 74, n. 1?3, p. 494-498, 2010. ISSN 0925-2312. Artificial Brains.

DELFIANTO, R.; KHODRA, M.; ROESLI, A. **Content-targeted advertising using genetic programming**. In: Electrical Engineering and Informatics (ICEEI), 2011 International Conference on. [S.l.: s.n.], 2011. p. 1-5. ISSN 2155-6822.

DENG, Y.; MANJUNATH, B.; SHIN, H. **Color image segmentation**. In: Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on. [S.l.: s.n.], 1999. v. 2, p. 451 Vol. 2. ISSN 1063-6919.

EL-MEDANY, W.; HUSSAIN, M. **Fpga-based advanced real traffic light controller system design**. In: Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2007. IDAACS 2007. 4th IEEE Workshop on. 2007. p. 100-105.

FARINACCIO, A.; VANNESCHI, L.; GIACOBINI, M.; MAURI, G.; PROVERO, P. **On the use of genetic programming for the prediction of survival in cancer**. In: Proceedings of the 12th annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM, 2010. (GECCO '10), p. 163-170. ISBN 978-1-4503-0072-8.

GABRIEL, P. H. R.; DELBEM, A. C. B. **Fundamentos de Algoritmos Evolutivos**. ICMC-USP, 2008. Disponível em: < http://www2.icmc.usp.br/~biblio/BIBLIOTECA/not_did/ND_075.pdf >. Acesso em: 09 de Abril de 2013.

GAMAGE, L.; SILVA, C. de; CAMPOS, R. **Design evolution of mechatronic systems through modeling, on-line monitoring, and evolutionary optimization**. Mechatronics, v. 22, n. 1, p. 83-94, 2012. ISSN 0957-4158.

GARG, A.; TAI, K. **Review of genetic programming in modeling of machining processes**. In: Modelling, Identification Control (ICMIC), 2012 Proceedings of International Conference on. [S.l.: s.n.], 2012. p. 653-658.

GONZALEZ, R. C.; WOODS, R. E. **Digital Image Processing**. 2nd. ed. Prentice Hall, 2002.

GORDAN, M.; DANCEA, O.; VLAICU, A.; STOIAN, I.; TSATOS, O. **Computer vision based decision support tool for hydro-dams surface deterioration assessment and visualization using fuzzy sets and pseudo-coloring**. In: Automation, Quality and Testing, Robotics, 2008. AQTR 2008. IEEE International Conference on. 2008. v. 3, p. 207-212.

GUO, P.-F.; BHATTACHARYA, P.; KHARMA, N. **Automated synthesis of feature functions for pattern detection**. In: Electrical and Computer Engineering (CCECE), 2010 23rd Canadian Conference on. [S.l.: s.n.], 2010. p. 1-4. ISSN 0840-7789.

GUSEL, L.; BREZOCNIK, M. **Application of genetic programming for modelling of material characteristics**. Expert Systems with Applications, v. 38, n. 12, p. 15014-15019, 2011. ISSN 0957-4174.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning, Second Edition: Data Mining, Inference, and Prediction**. 0002-2009. corr. 3rd. ed. Springer, 2009. Hardcover. (Springer Series in Statistics). ISBN 0387848576.

HERNÁNDEZ, D.; OLAGUE, G.; CLEMENTE, E.; DOZAL, L. **Evolving a conspicuous point detector based on an artificial dorsal stream: Slam system**. In: Proceedings of the fourteenth international conference on Genetic and

evolutionary computation conference. New York, NY, USA: ACM, 2012. (GECCO '12), p. 1087-1094. ISBN 978-1-4503-1177-9.

HU, J.; XIE, M. **Fingerprint classification based on genetic programming**. In: Computer Engineering and Technology (ICCET), 2010 2nd International Conference on. [S.l.: s.n.], 2010. v. 6, p. V6-193-V6-196.

ISLAM, T.; RICO-RAMIREZ, M. A.; HAN, D. **Tree-based genetic programming approach to infer microphysical parameters of the DSDs from the polarization diversity measurements**. Computers & Geosciences, v. 48, n. 0, p. 20-30, 2012. ISSN 0098-3004.

JAYAKUMAR, S.; KANNA, R. **Inspection system for detecting defects in a transistor using artificial neural network (ann)**. In: Communication and Computational Intelligence, 2010 International Conference on. 2010. p. 76-81.

KAMEYA, Y.; KUMAGAI, J.; KURATA, Y. **Accelerating genetic programming by frequent subtree mining**. In: Proceedings of the 10th annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM, 2008. (GECCO '08), p. 1203-1210. ISBN 978-1-60558-130-9.

KILLING, J.; SURGENOR, B.; MECHEFSKE, C. **A machine vision system for the detection of missing fasteners on steel stampings**. The International Journal of Advanced Manufacturing Technology, Springer London, v. 41, p. 808-819, 2009. ISSN 0268-3768.10.1007/s00170-008-1516-3.

KOZA, J. R. **Non-Linear Genetic Algorithms for Solving Problems**. 19 jun. 1990. United States Patent 4935877. Filed may 20, 1988, issued june 19, 1990, 4,935,877. Australian patent 611,350 issued september 21, 1991. Canadian patent 1,311,561 issued december 15, 1992.

KOZA, J. R. **Genetic Programming**: On the programming of computers by means of natural selection. [S.l.]: MIT Press, 1992. ISBN 0262111705.

KUO, H.-C.; SU, K.-Y.; ONG, P.-L.; HUANG, J.-P. **Learning a prediction model for protein-protein recognition**. In: Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human. New York, NY, USA: ACM, 2009. (ICIS '09), p. 736-741. ISBN 978-1-60558-710-3.

LANGDON, W.; HARMAN, M.; JIA, Y. **Multi objective higher order mutation testing with genetic programming**. In: Testing: Academic and Industrial Conference - Practice and Research Techniques, 2009. TAIC PART '09. [S.l.: s.n.], 2009. p. 21-29.

LI, G.; ZENG, X.-J. **Genetic programming with a norm-referenced fitness function**. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM, 2011. (GECCO '11), p. 1323-1330. ISBN 978-1-4503-0557-0.

LIN, J. Y. **Fitness enhancement of layered architecture genetic programming**. In: Computer Symposium (ICS), 2010 International. [S.l.: s.n.], 2010. p. 700-704.

LONES, M.; FUENTE, L.; TURNER, A.; CAVES, L.; STEPNEY, S.; SMITH, S.; TYRRELL, A. **Artificial biochemical networks: Evolving dynamical systems to control dynamical systems**. Evolutionary Computation, IEEE Transactions on, PP, n. 99, p. 1-1, 2013. ISSN 1089-778X.

LU, J.; PLATANIOTIS, K. **On conversion from color to gray-scale images for face detection**. In: Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on. 2009. p. 114-119.

LULIO, L.; TRONCO, M.; PORTO, A. **Jseg-based image segmentation in computer vision for agricultural mobile robot navigation**. In: Computational Intelligence in Robotics and Automation (CIRA), 2009 IEEE International Symposium on. 2009. p. 240-245.

MAHMOOD, M. T.; MAJID, A.; HAN, J.; CHOI, Y. K. **Genetic programming based blind image deconvolution for surveillancesystems**. Engineering Applications of Artificial Intelligence, v. 26, n. 3, p. 1115-1123, 2013. ISSN 0952-1976.

MATHWORKS. **Matlab Overview**. MathWorks – Accelerating the pace of engineering and science, 2013. Disponível em: <http://www.mathworks.com/products/matlab/>. Acesso em: 25/02/2013.

MEUTH, R. J. **Meta-learning genetic programming**. In: Proceedings of the 12th annual conference companion on Genetic and evolutionary computation. New York, NY, USA: ACM, 2010. (GECCO '10), p. 2101-2102. ISBN 978-1-4503-0073-5.

NESHATIAN, K.; ZHANG, M. **Dimensionality reduction in face detection: A genetic programming approach**. In: Image and Vision Computing New Zealand, 2009. IVCNZ '09. 24th International Conference. [S.l.: s.n.], 2009. p. 391-396. ISSN 2151-2205.

NITSURE, S.; LONDHE, S.; KHARE, K. **Application of genetic programming for estimation of ocean wave heights**. In: Nature Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on. [S.l.: s.n.], 2009. p. 1520-1523.

OBID. **PESQUISAS E ESTATÍSTICAS/Conceitos Estatísticos/Sensibilidade e Especificidade**. Observatório Brasileiro de Informações Sobre Drogas, 2013. Disponível em: <http://www.obid.senad.gov.br/portais/OBID/index.php>. Acesso em: 06/01/2013.

OGATA, K. **Engenharia de controle moderno**. 3a Ed.. Prentice/Hall do Brasil, 1998.

OLAGUE, G.; TRUJILLO, L. **Evolutionary-computer-assisted design of image operators that detect interest points using genetic programming**. Image and Vision Computing, v. 29, n. 7, p. 484-498, 2011. ISSN 0262-8856.

OLAGUE, G.; TRUJILLO, L. **Interest point detection through multiobjective genetic programming**. Applied Soft Computing, v. 12, n. 8, p. 2566-2582, 2012. ISSN 1568-4946.

OTT, P.; EVERINGHAM, M. **Implicit color segmentation features for pedestrian and object detection**. In: Computer Vision, 2009 IEEE 12th International Conference on. 2009. p. 723-730. ISSN 1550-5499.

PEDRINO, E.; OGASHAWARA, O.; RODA, V. **Reconfigurable architecture for mathematical morphology using genetic programming and fpgas**. In: Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on. [S.l.: s.n.], 2010. p. 1-4.

PEDRINO, E.; SAITO, J.; RODA, V. **Architecture for binary mathematical morphology reconfigurable by genetic programming**. In: Programmable Logic Conference (SPL), 2010 VI Southern. [S.l.: s.n.], 2010. p. 93-98.

PEDRINO, E.; SAITO, J.; KATO, E. R. R.; MORANDIN, O.; CURA, L. D. V.; RODA, V.; TRONCO, M.; TSUNAKI, R. **Automatic construction of image operators using a genetic programming approach**. In: Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on. [S.l.: s.n.], 2011. p. 636-641. ISSN 2164-7143.

PEDRINO, E.C.; MORANDIN, O.; KATO, E. R R; RODA, V.O., **Intelligent FPGA based system for shape recognition,** In: *Programmable Logic (SPL), 2011 VII Southern Conference on* , vol., no., pp.197,202, 13-15 April 2011

PEREZ, C. B.; OLAGUE, G. **Evolutionary learning of local descriptor operators for object recognition**. In: Proceedings of the 11th Annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM, 2009. (GECCO '09), p. 1051-1058. ISBN 978-1-60558-325-9.

PINTO, B.; SONG, A. **Detecting motion from noisy scenes using genetic programming**. In: Image and Vision Computing New Zealand, 2009. IVCNZ '09. 24th International Conference. 2009. p. 322-327. ISSN 2151-2205.

POMPAPATHI, M.; KRISHNA, A. S.; BABU, B. **An efficient approach for removal of impulse noise from highly corrupted images by preserving edge details**. In: Signal and Image Processing (ICSIP), 2010 International Conference on. 2010. p. 498-501.

PUENTE, C.; OLAGUE, G.; SMITH, S. V.; BULLOCK, S.; GONZALEZ, M. A.; HINOJOSA, A. **Genetic programming methodology that synthesize vegetation indices for the estimation of soil cover**. In: Proceedings of the 11th Annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM, 2009. (GECCO '09), p. 1593-1600. ISBN 978-1-60558-325-9.

PUROHIT, A.; CHAUDHARI, N.; TIWARI, A. **Construction of classifier with feature selection based on genetic programming**. In: Evolutionary Computation (CEC), 2010 IEEE Congress on. [S.l.: s.n.], 2010. p. 1-5.

ROTH, M. **Survey of neural network technology for automatic target recognition. Neural Networks**, IEEE Transactions on, v. 1, n. 1, p. 28-43, mar 1990. ISSN 1045-9227.

RUSSELL, S. J.; NORVIG, P. **Artificial Intelligence:** A Modern Approach. 3. ed. Pearson Education, 2010. ISBN 0-13-604259-7.

SAMMUT, C.; WEBB, G. **Encyclopedia of Machine Learning**. Springer, 2011. (Springer reference). ISBN 9780387307688.

SHEN, J.; KARAKUS, M.; XU, C. **Direct expressions for linearization of shear strength envelopes given by the generalized hoek/brown criterion using genetic programming**. Computers and Geotechnics, v. 44, n. 0, p. 139-146, 2012. ISSN 0266-352X.

SHOKRKAR, H.; SALAHI, A.; KASIRI, N.; MOHAMMADI, T. **Prediction of permeation flux decline during mf of oily wastewater using genetic programming**. Chemical Engineering Research and Design, v. 90, n. 6, p. 846-853, 2012. ISSN 0263-8762. Special Issue on the 3rd European Process intensification Conference.

SILDAM, J. **Masking of time-frequency patterns in applications of passive underwater target detection**. EURASIP J. Adv. Signal Process, Hindawi Publishing Corp., New York, NY, United States, v. 2010, p. 6:1-6:7, jan. 2010. ISSN 1110-8657.

SILVA, S.; ALMEIDA, J. **Gplab-a genetic programming toolbox for matlab**. In: In Proc. of the Nordic MATLAB Conference (NMC-2003. [S.l.: s.n.], 2005. p. 273-278.

SOLT, P. D. **Artwork evolution**. In: ACM SIGGRAPH 2010 Posters. New York, NY, USA: ACM, 2010. (SIGGRAPH '10), p. 20:1-20:1. ISBN 978-1-4503-0393-4.

SPINA, T.; MONTOYA-ZEGARRA, J.; FALCAO, A.; MIRANDA, P. **Fast interactive segmentation of natural images using the image foresting transform**. In: Digital Signal Processing, 2009 16th International Conference on. [S.l.: s.n.], 2009. p. 1-8.

SUPARI; SAWITRI, D.; PURNOMO, M. **Genetic programming function representation for fuzzy polar speed controller of induction motor**. In: Instrumentation, Communications, Information Technology, and Biomedical Engineering (ICICI-BME), 2009 International Conference on. [S.l.: s.n.], 2009. p. 1-5.

TAKAMURA, S.; MATSUMURA, M.; YASHIMA, Y. **Automatic pixel predictor construction using an evolutionary method**. In: Picture Coding Symposium, 2009. PCS 2009. [S.l.: s.n.], 2009. p. 1-4.

TAMBOLI, A.; SHAH, M. **A generic structure of object classification using genetic programming**. In: Communication Systems and Network Technologies (CSNT), 2011 International Conference on. [S.l.: s.n.], 2011. p. 723-728.

TERANO, T.; SINOHARA, Y.; MATSUI, S.; NAKAMURA, H.. **Dam gate diagnosing advisor-an expert system for steel structures**. In: Artificial Intelligence for Industrial Applications, 1988. IEEE AI '88., Proceedings of the International Workshop on. 1988. p. 129-134.

TO, C.; PHAM, T. **Analysis of cardiac imaging data using decision tree based parallel genetic programming**. In: Image and Signal Processing and Analysis, 2009. ISPA 2009. Proceedings of 6th International Symposium on. [S.l.: s.n.], 2009. p. 317-320. ISSN 1845-5921.

TRUJILLO, L.; LEGRAND, P.; OLAGUE, G.; LÉVY-VÉHEL, J. **Evolving estimators of the pointwise hölder exponent with genetic programming**. Information Sciences, v. 209, n. 0, p. 61-79, 2012. ISSN 0020-0255.

TRUJILLO, L.; MARTÍNEZ, Y.; GALVÁN-LÓPEZ, E.; LEGRAND, P. **Predicting problem difficulty for genetic programming applied to data classification**. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM, 2011. (GECCO '11), p. 1355-1362. ISBN 978-1-4503-0557-0.

TRUJILLO, L.; OLAGUE, G. **Automated design of image operators that detect interest points**. Evol. Comput., MIT Press, Cambridge, MA, USA, v. 16, n. 4, p. 483-507, dez. 2008. ISSN 1063-6560.

UTO, K.; KOSUGI, Y.; OGATA, T. **Evaluation of oak wilt index based on genetic programming**. In: Hyperspectral Image and Signal Processing: Evolution in Remote Sensing, 2009. WHISPERS '09. First Workshop on. [S.l.: s.n.], 2009. p. 1-4.

WANG, J.; CHEN, Q.S.; LEE, C.H.. **Design and implementation of a virtual reconfigurable architecture for different applications of intrinsic evolvable hardware**. Computers Digital Techniques, IET, v. 2, n. 5, p. 386-400, september 2008. ISSN 1751-8601.

WANG, J.; TAN, Y. **Morphological image enhancement procedure design by using genetic programming**. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM, 2011. (GECCO '11), p. 1435-1442. ISBN 978-1-4503-0557-0.

WANG, S.; MA, J.; LIU, J.; NIU, X. **Evolving choice structures for genetic programming**. Information Processing Letters, v. 110, n. 20, p. 871-876, 2010. ISSN 0020-0190.

WEISE, T.; WAN, M.; WANG, P.; TANG, K.; DEVERT, A.; YAO, X. **Frequency fitness assignment**. Evolutionary Computation, IEEE Transactions on, PP, n. 99, p. 1-1, 2013. ISSN 1089-778X.

WHITE, D.; ARCURI, A.; CLARK, J. **Evolutionary improvement of programs**. Evolutionary Computation, IEEE Transactions on, v. 15, n. 4, p. 515-538, aug. 2011. ISSN 1089-778X.

WILLIAM, E.; NORTHERN, J. **Genetic programming lab (gplab) tool set version 3.0**. In: Region 5 Conference, 2008 IEEE. [S.l.: s.n.], 2008. p. 1-6.

WIRTH, M.; NIKITENKO, D. **The effect of colour space on image sharpening algorithms**. In: Computer and Robot Vision (CRV), 2010 Canadian Conference on. 2010. p. 79-85.

WORZEL, W. P.; YU, J.; ALMAL, A. A.; CHINNAIYAN, A. M. **Applications of genetic programming in cancer research**. The International Journal of Biochemistry & Cell Biology, v. 41, n. 2, p. 405-413, 2009. ISSN 1357-2725. Molecular and Cellular Evolution: A Celebration of the 200th Anniversary of the Birth of Charles Darwin.

XIE, H.; ZHANG, M. **Parent selection pressure auto-tuning for tournament selection in genetic programming**. Evolutionary Computation, IEEE Transactions on, v. 17, n. 1, p. 1-19, 2013. ISSN 1089-778X.

XIONG, F.; TANIK, M. **An experiment on evolutionary design of combinational logic circuits using information theory**. In: Southeastcon, 2011 Proceedings of IEEE. [S.l.: s.n.], 2011. p. 379-383. ISSN 1091-0050.

XU, C.; WANG, W.; LIU, P. **A genetic programming model for real-time crash prediction on freeways**. Intelligent Transportation Systems, IEEE Transactions on, PP, n. 99, p.1-13, 2012. ISSN 1524-9050.

YI, L.; WANLI, K. **A new genetic programming algorithm for building decision tree**. Procedia Engineering, v. 15, n. 0, p. 3658-3662, 2011. ISSN 1877-7058. CEIS 2011.

Apêndice A

TUTORIAL DA TOOLBOX DE PROGRAMAÇÃO GENÉTICA GPLAB3

Os principais módulos da *ToolBox* são **SET VARS**, **GEN POP** e **GENERATION**, onde cada um representa um ponto de interação com o usuário.

Em **GEN POP** a população inicial é gerada e seu *fitness* é calculado. Dentro da população gerada, cada indivíduo é representado por uma árvore que pode ser inicializada nos modos: *full*, *grow* e *ramped half-and-half*.

O *fitness* dos indivíduos é, por padrão, o somatório das diferenças absolutas entre os valores obtidos e os valores esperados. Quanto menor o valor do *fitness*, melhor é o indivíduo. Sendo este o padrão para problemas de regressão simbólica e problemas de paridade.

Caso seja requisitada apenas a geração da população inicial, o módulo **GENERATION** não será utilizado.

O módulo **GENERATION** é responsável pela criação de uma nova geração de indivíduos por meio da utilização de operadores genéticos na geração anterior. Os operadores padrões são crossover e mutação, disponibilizados como funções "*plug and play*". Ambos devem ter um grupo de pais para escolha, criado por amostragem que pode ou não ter como base de escolha, o número esperado de descendentes de cada individuo. A *ToolBox* já possui quatro métodos de seleção e três para cálculo do numero esperado de descendentes, que podem ser combinados aleatoriamente aos pares. Tais operadores criam novos indivíduos até o preenchimento da população até atingir o tamanho pré-determinado.

Após o cálculo do *fitness* o módulo **SURVIVAL** é executado, onde é feita a escolha dos melhores indivíduos por parâmetros de elitismo e sobrevivência. O

módulo **GENETATION** executa essa sequência até que uma condição de parada seja alcançada ou o numero máximo de gerações seja atingido.

Por fim o módulo **SET VARS** que pode tanto inicializar os parâmetros com valores padrão, assim como atualizar os parâmetros com opções do usuário.

1.1. Parâmetros/Variáveis

Representado pela estrutura "vars" da *ToolBox*, é constituído de quatro campos:

- params: armazena todas variáveis que determinam a execução do algoritmo. O conteúdo desse campo não deveria variar durante a execução, apesar de ser possível iniciar a execução com uma configuração e continuar com outra.
- state: armazena a representação do estado atual do algoritmo. Este campo é atualizado constantemente durante a execução e não devem ser alteradas pelo usuário.
- pop: representa a população atual, sendo alterado juntamente com a evolução da população.
- data: é o conjunto de dados utilizados pelo algoritmo para o processo evolutivo e, opcionalmente, executar validação cruzada.

2. Utilização

O modo mais simples de utilização é o que não é necessário conhecimento de parâmetros nem como defini-los. Basta que seja definido o número máximo de gerações "g" e a quantidade de indivíduos "n" além de possuir um conjunto de dados para análise. Este modo exige apenas a execução de duas funções:

```
[vars,b]=gplab(g,n);
[vars,b]=gplab(g,vars);
```

Durante a execução da primeira função, o usuário é solicitado a informar a localização dos arquivos com os dados a serem analisados. Esta função executa o algoritmo por “g” gerações com “n” indivíduos e retorna todos os dados necessários para a continuação de execução do algoritmo, assim como todos os parâmetros inicializados em "vars", e o melhor indivíduo da execução em "b". A segunda função pode ser executada com o número de gerações adicionais "g" continuando do ponto

de onde a função anterior parou. Isso é possível devido ao fato de informar o estado anterior com a passagem do parâmetro "vars".

A *ToolBox* tem definida por padrão as operações de soma, subtração, multiplicação, seno, cosseno e logaritmo no espaço de busca. Sendo assim podemos exemplificar encontrando o cálculo de volume de paralelepípedos.

Suponha que tenhamos em posse os valores de largura, altura, comprimento e volume de alguns paralelepípedos (Tabela 1), e gostaríamos de descobrir qual o cálculo necessário para encontrar o volume dados os demais valores.

Tabela 1 – Dados de volume de paralelepípedos

Largura	Altura	Comprimento	Volume
10	30	20	6000
50	10	5.5	2750
35	20	15	10500
13	42	12	6552
23	10	10	2300
20	20	20	8000
1	2	3	6
76	2	10	1520
12	98	6	7056
1.5	2.2	5	16.5

Para isso é necessário criar dois arquivos de texto simples. O primeiro arquivo, “vol_x.txt” deve conter os valores de largura, altura e comprimento que possuímos referentes aos paralelepípedos. O segundo arquivo “vol_y.txt” deve conter o resultado do cálculo que queremos encontrar, ou seja, o volume.

Com os arquivos criados, executamos a primeira função “`[vars,b]=gplab(g,n);`” e quando solicitado, fornecemos os arquivos criados com os dados. Ao término da execução podemos visualizar a representação da melhor resposta encontrada através do comando “`drawtree(b.tree);`”, onde b é o melhor indivíduo encontrado pelo algoritmo (Figura 1).

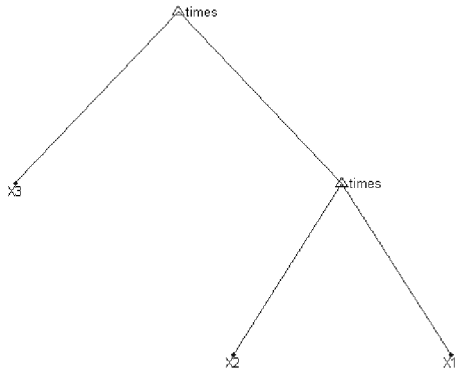


Figura 1 – Representação da seqüência de operações do melhor indivíduo encontrado

Como o arquivo foi montado de acordo com a Figura 1, temos largura(X1), altura(X2) e comprimento(X3) e o cálculo de volume de paralelepípedos como $(largura \times altura) \times comprimento$ que realmente é a fórmula matemática para tal cálculo.

Porém, esse tipo de execução é limitado, pois caso seja necessária a adição de terminais e/ou funções, neste modo de execução não é possível. Por exemplo, para o cálculo do volume de cilindros é necessário utilizar a constante π, e para utilizá-la é necessário alterar alguns parâmetros da *ToolBox*.

2.1. Execução com alteração de parâmetros

É um modo de execução mais elaborado, onde o usuário altera o comportamento do algoritmo definindo valores de parâmetros diferentes dos padrões. Para definir parâmetros, primeiro inicializamos todos os parâmetros com seus valores padrões e assim alteramos os que necessitamos. No caso do exemplo do volume de cilindros precisamos adicionar a constante aos terminais.

```
params=resetparams;
params=setterminals(params,'pi');
[vars,b]=gplab(g,n,params);
```

A primeira função inicia os parâmetros com os valores padrão, a segunda adiciona a constante aos terminais e a terceira executa o algoritmo com o novo conjunto de parâmetros com os valores alterados.

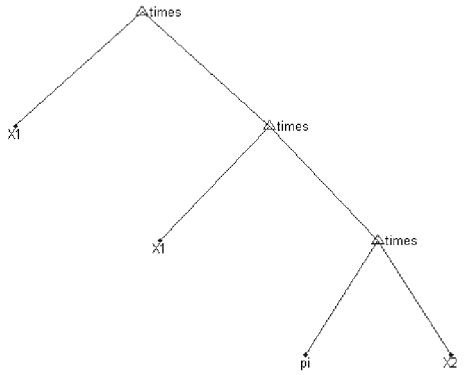


Figura 2 – Representação do cálculo encontrado para volume de cilindros.

O cálculo encontrado na Figura 2 utiliza a constante π para encontrar o resultado. Sendo assim temos como cálculo de volume de cilindros $\pi \times X2 \times X1 \times X1$ porém, em uma análise rápida nota-se que a operação $X1 \times X1$ poderia ser substituída pela função "ao_quadrado", e para isso bastaria criar um arquivo com a função e adicioná-la aos parâmetros através do comando "`params=setfunctions(params,'ao_quadrado',1);`", onde o valor 1 após a função indica sua aridade.

2.2. Execução avançada

Indicado para testes mais aprofundados, como teste de operadores genéticos, métodos de amostragem, métodos de cálculo de *fitness*, etc.. Este modo de execução permite que o usuário utilize toda infra-estrutura da programação genética fornecida pelo *ToolBox* para testes em apenas partes do algoritmo. Isto é possível devido a abordagem "*plug and play*" da *ToolBox*.

Por exemplo, pode-se adicionar novos operadores, além dos de mutação e crossover, através do commando "params=addoperators(params,'newoperator',nparents,nchildren);" onde newoperator é o nome da função que descreve o novo operador, nparents indica o número de pais necessários para a utilização do operador e nchildren indica quantos filhos serão gerados por este operador.

Devido a estrutura "*plug and play*" da *ToolBox*, praticamente qualquer parte do algoritmo pode ser modificada ou substituída por uma versão nova ou alterada.

3. Utilização com processamento de imagens

Para abordagem do problema com o *ToolBox* foi apenas necessário alterar um trecho do código que não é "*plug and play*", que é a verificação dos dados das variáveis no arquivo "checkvarsdata.m" com as seguintes alterações:

```
Linha 42:   x=load(params.datafilex); % load the file
Alterada para:
Linha 42:   fid=fopen(params.datafilex);x=textscan(fid,'%q');x=x{1,1};fclose(fid);
```

Assim como:

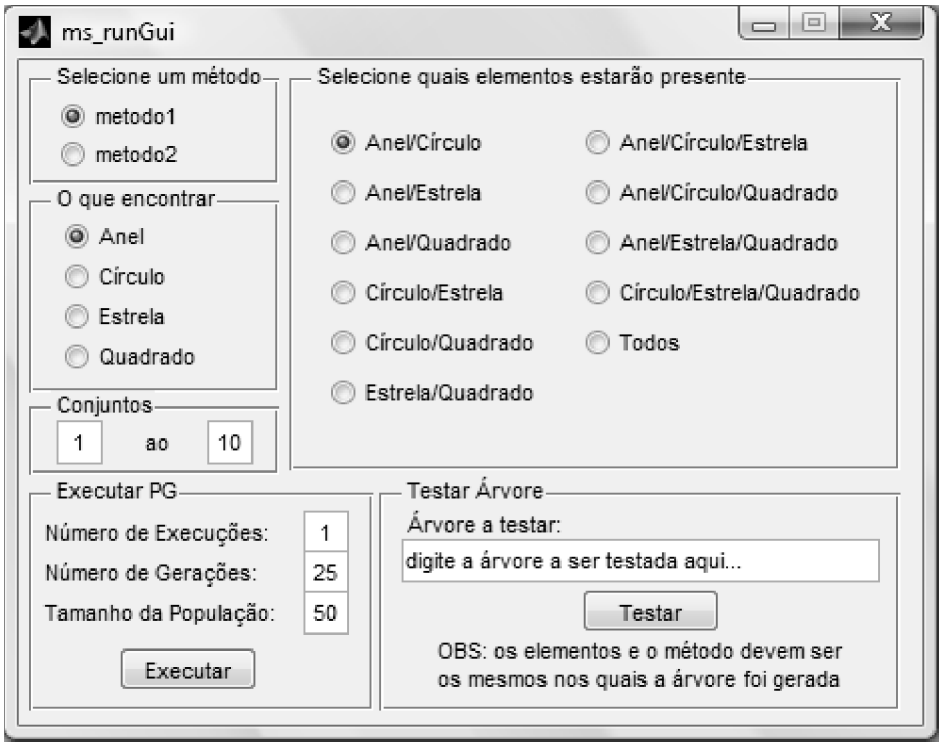
```
Linha 54:   y=load(params.datafiley); % load the file
Alterada para:
Linha 54:   fid=fopen(params.datafiley);y=textscan(fid,'%q');y=y{1,1};fclose(fid);
```

Nenhuma outra alteração foi feita à ToolBox, apenas foi utilizada a estrutura "*plug and play*" para adicionar as funções e terminais necessários para o tratamento do problema

Apêndice B

DOCUMENTAÇÃO DA GUI GERADA

Para execução basta executar o comando ms_run, sem nenhum parâmetro adicional.



As configurações selecionadas são globais exceto nas caixas de execução e teste, por este motivo é necessário sempre selecionar as configurações corretamente.

1) **Seleção do método:** Neste item deverá ser selecionado qual método será utilizado.
2) **O que encontrar:** Neste item será selecionado qual o objeto que será o objetivo de destaque do método.
3) **Seleção de elementos:** Indica quais elementos estarão presentes na imagem inicial que será tratada.
4) **Conjuntos:** Indica quais conjuntos de imagens pré-definidos devem ser utilizados. (geralmente são utilizados conjuntos distintos em cada etapa, por exemplo: para o treinamento utilizar os conjuntos de 1 à 10 e para testes do conjunto 11 ao 50)

5) **Execução:** É necessário indicar o número de execuções onde, por exemplo, caso seja definido o valor 5, serão geradas 5 soluções através do processo evolutivo. Além do número de execuções, definir o número de gerações e o tamanho da população é necessário.

6) **Testes:** Neste modo é apenas necessário copiar a árvore gerada, que se encontra no arquivo newData.ini dentro de cada pasta de solução.

- As soluções geradas, ficam armazenadas dentro da subpasta imgMod de acordo com as opções selecionadas.
    - No exemplo acima as soluções se encontrariam na pasta imgMod\metodo1\An\AnCirc\SolX
- Os testes seguem o mesmo padrão com uma pasta adicional.
    - No mesmo exemplo teríamos os testes em imgMod\metodo1\An\AnCirc\TreeTest\SolX