CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CAMPUS TIMÓTEO

Pedro Rodrigues Pinto

UTILIZANDO MÉTODOS EXATOS NO RECONHECIMENTO E RASTREAMENTO DE OBJETOS

Timóteo

2015

Pedro Rodrigues Pinto

UTILIZANDO MÉTODOS EXATOS NO RECONHECIMENTO E RASTREAMENTO DE OBJETOS

Monografia apresentada à Coordenação de Engenharia de Computação do Campus Timóteo do Centro Federal de Educação Tecnológica de Minas Gerais para obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Odilon Corrêa da Silva

Timóteo

2015

...

Agradecimentos

A todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

“And once the storm is over, you won’t remember how you made it through, how you managed to survive. You won’t even be sure, whether the storm is really over. But one thing is certain. When you come out of the storm, you won’t be the same person who walked in. That’s what this storm’s all about.”

Haruki Murakami

Resumo

Apesar de ter grande aplicabilidade, fazer com que o computador seja capaz de reconhecer objetos em imagens continua sendo um dos grandes desafios da computação. Este trabalho propõe um processo de reconhecimento e rastreamento de objetos por cor e forma, em imagens ou sequências de imagens capturadas em tempo real. Para tanto, foram aplicadas técnicas de processamento digital de imagens e métodos exatos. Para validar o funcionamento do processo proposto, desenvolveu-se um *software* utilizando recursos da biblioteca OpenCV. Nos testes realizados, o *software* apresentou uma taxa de acerto superior a 90%, e conseguiu manter o tempo de resposta inferior à 33 milissegundos. Por meio de estudos de caso, foram evidenciadas a simplicidade, eficiência e flexibilidade do processo proposto. Isso mostra a viabilidade do uso de processamento digital de imagens e métodos exatos no reconhecimento e rastreamento de objetos em problemas reais.

Palavras-chave: Processamento digital de imagens. Reconhecimento de objetos. Métodos exatos. OpenCV.

Abstract

Despite having wide applicability, the ability of a computer to recognize objects in images remains one of the great challenges of computing. This essay proposes a process to recognize and track objects by color and shape, that can be used on either still images or real-time video. To achieve this end, digital image processing techniques and exact methods were used. To validate the operation of the proposed process, a software was developed using the resources provided by the OpenCV library. In tests, the software demonstrated over 90% accuracy rate in object recognition, while still maintaining the response rate under 33 milliseconds. Through case studies, we demonstrate the relative simplicity, efficiency and flexibility of the proposed process. This shows the feasibility of using digital image processing and exact methods in the recognition and object tracking on real problems.

Keywords: Digital imagem processing. Exact methods. Object recognition. OpenCV.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

PDI	Processamento Digital de Imagens
RNA	Redes Neurais Artificiais
fps	*Frames Per Second*
OpenCV	*Open Source Computer Vision*
RGB	*Red, Green and Blue*
HSV	*Hue, Saturation, and Value*
SDK	*Software Development Kit*
GPU	*Graphics Processing Unit*

Sumário

1 Introdução

A visão é o mais avançado dos cincos sentidos do ser humano e por este motivo a imagem é o principal recurso utilizado na percepção humana do mundo ao seu redor (GONZALES; WOODS, 2010). As imagens capturadas pelo olhos humanos são informações que transitam entre os neurônios até que algum significado seja produzido. Alguns dos processos envolvendo raciocínio lógico-matemático, que um dia foram exclusivos do cérebro humano, já fazem parte do cotidiano da computação, então, é uma evolução natural tentar dotar os computadores das outras capacidades do nosso cérebro, dentre elas, a capacidade de percepção de mundo por meio de imagens.

As possibilidades são inúmeras, porém o desenvolvimento de sistemas computacionais que se assemelham aos seres humanos em suas capacidades de interpretar e reagir ao que se vê ainda é um grande desafio. O cérebro humano, com sua grande capacidade e diferentes métodos de processamento de informação, ainda não foi plenamente compreendido e replicado pelas tecnologias atuais (PEDRINI; SCHWARTZ, 2008).

A área de estudo que lida com o processamento de dados de imagens para armazenamento, transmissão e representação considerando a percepção automática por máquinas é denominada de processamento digital de imagens (PDI). O PDI tem seus primórdios na década de 1920, quando começaram a transmitir as primeiras fotografias entre jornais de Londres e Nova York por meio de um cabo cruzando o oceânico Atlântico. Atualmente, o PDI atinge diversas áreas, como a medicina, exploração espacial, segurança, biologia, defesa e indústria (GONZALES; WOODS, 2010). Em geral, não existe um consenso entre os pesquisadores sobre onde o PDI termina e outras áreas relacionadas, como a análise de imagens e a visão computacional, começam (PEDRINI; SCHWARTZ, 2008).

O reconhecimento de objetos em imagens é um dentre os vários escopos que envolvem PDI. Neste caso, o desafio é fazer com que o sistema computacional conheça padrões de um objeto e classifique regiões de imagens de acordo com estes padrões, sendo que essas regiões de imagens correspondem aos objetos em questão (GONZALES; WOODS, 2010). Quando o reconhecimento de objetos é feito de forma contínua em um conjunto de imagens sequenciais denomina-se como rastreamento de objetos em imagens. Se essa funcionalidade aumenta ainda mais a aplicabilidade do PDI, também aumenta a complexidade dos problemas a serem solucionados.

Um sistema computacional que lide com PDI pode ser utilizado no reconhecimento e rastreamento de objetos em imagens e aplicado a diversos contextos. Entre outros, podem ser citados:

- Classificação de lesões de pele (OLIVEIRA, 2012);
- Controle de qualidade em linha de produção (CARVALHO et al., 2013);

- Monitoramento de ciclistas e pedestres (HEIKKILÄ; SILVÉN, 2004);
- Rastreamento dos jogadores e da bola em um jogo de futebol (SEO et al., 1997);
- Reconhecimento de caracteres (PERMALOFF; GRAFTON, 1992);
- Reconhecimento facial (SERRE et al., 2002).

Cada contexto apresenta restrições e características específicas com as quais o PDI deve lidar. Por exemplo, o projeto desenvolvido por Carvalho et al. (2013) realiza o reconhecimento de objetos no contexto de um sistema de tempo real. Sistemas de tempo real são sistemas de computação que monitoram, respondem ou controlam um ambiente externo que naturalmente possui um comportamento de tempo real (SHAW, 2003). Ou seja, além de utilizar as técnicas de PDI no controle de qualidade em linhas de produção, o contexto requer que estas técnicas atendam restrições impostas pelo mundo externo com qual o problema está relacionado, no caso, restrições temporais. Desta forma, o desempenho é um fator crítico que deve ser considerado no desenvolvimento do mecanismo de reconhecimento e rastreamento de objetos.

Com o amadurecimento e evolução do PDI, pesquisas foram desenvolvidas com o objetivo de criar recursos computacionais para otimizar e facilitar as análises de imagens. Revisando a literatura, algumas ferramentas foram encontradas:

- BoofCV - Biblioteca *open source* Java para visão computacional em tempo real[1];
- MATLAB - Software de alta performance voltado para o cálculo numérico[2];
- OpenCV - Visão computacional *open source*[3];
- SimpleCV - Plataforma de visão computacional usando Python[4];
- VXL - Bibilotecas C++ para implementação e pesquisa de visão computacional[5].

Essas ferramentas fornecem recursos que podem ser utilizados no desenvolvimento de soluções eficientes em diversas abordagens. No caso do reconhecimento e rastreamento de objetos em imagens, existe a possibilidade de serem usadas técnicas de casamento de padrões, Redes Neurais Artificiais (RNA), métodos exatos e outros. A escolha da estratégia ideal deve ser feita de acordo com o problema a ser trabalhado.

[1] <https://http://boofcv.org/>
[2] <https://www.mathworks.com/products/image/>
[3] <http://http://opencv.org/>
[4] <http://simplecv.org/>
[5] <http://vxl.sourceforge.net/>

1.1 Definição do problema

Influenciado por esta gama de aplicações e recursos disponíveis relacionados a PDI, este trabalho propõe-se a especificar, implementar e avaliar o desempenho de um processo para reconhecimento e rastreamento de objetos em imagens ou sequências de imagens em tempo real, baseando-se em métodos exatos. Com o intuito de alcançar este fim serão utilizadas técnicas de PDI para adquirir, realçar, comprimir, segmentar, sintetizar e realizar outros processamentos ou análises em imagens disponibilizadas por uma ferramenta computacional auxiliar.

O escopo deste trabalho compreende primariamente um cenário bidimensional de iluminação uniforme e controlada, com objetos de geometria simples (círculos, triângulos e retângulos) de cores únicas dispostos por completo em um ambiente com fundo de cor que o diferencie dos objetos a serem detectados. Alguns dos estudos de caso apresentados podem extrapolar de alguma forma as restrições mencionados, mas estes devem ser tratados como exceções.

1.2 Motivação

A busca de um equivalente computacional que se aproxime da cognição humana e habilidade do ser humano em tomar decisões de acordo com as informações providas por imagens pode auxiliar a resolução de problemas complexos. Aliado a isso, o crescente avanço da tecnologia associado ao desenvolvimento de novos algoritmos tem permitido um número de aplicações cada vez maior do PDI. Além dos trabalhos já mencionados, alguns dos domínios de conhecimento que envolvem a utilização de PDI na resolução de problemas incluem (GONZALES; WOODS, 2010):

- Área militar;
- Arqueologia;
- Astronomia;
- Automação Industrial;
- Biologia;
- Entretenimento;
- Medicina;
- Microscopia;
- Sensoriamento remoto.

Muitos dos problemas encontrados nestas áreas podem envolver de alguma forma o reconhecimento e rastreamento de objetos em imagens. Dentro outros, estas são algumas das várias situações que podem ser beneficiadas com os resultados deste projeto:

- Acompanhamento da trajetória de um projétil;
- Contagem de organismos em uma amostra de sangue;
- Controle de qualidade de linhas de produção;
- Identificação de estrelas e constelações.

Nesses e em outros problemas, o principal desafio é identificar regiões de uma imagem como objetos e classificá-los através de suas características individuais. Para isto, a utilização de métodos exatos é uma proposta atraente, já que estes classificam objetos de interesse baseado em suas características peculiares, tais como: comprimento, perímetro, raio, cor, altura, rotação, etc (SILVA, 2014).

Como os objetos a serem detectados ou rastreados neste trabalho possuem características bem definidas e constantes, o uso de métodos exatos é ideal para manter a simplicidade do processo a ser especificado. Além disso, o processo acaba se tornando mais flexível, uma vez que na necessidade de reconhecer objetos fora do escopo proposto, basta que sejam identificadas quais as características particulares deste novo objeto e explicitá-las na etapa de classificação do processo. Outro motivo que reforça a vantagem do uso de métodos exatos é o seu baixo custo computacional, visto que eles se baseiam na realização de operações simples sobre imagens. Este aspecto é importante quando o processo for utilizado em problemas com requisitos de tempo real.

Deste modo, o processo especificado neste trabalho pode ser aplicado em diversos problemas que envolvam reconhecimento e rastreamento de objetos em imagens. Uma vez que muitos dos trabalhos previamente mencionados propõem soluções para problemas específicos, a complexidade e alto custo computacional destes podem comprometer sua utilização em diferentes problemas. Então, a especificação de um processo de natureza simples, flexível e eficiente é relevante para incentivar e auxiliar futuros trabalhos e assim ampliar a quantidade de problemas e situações capazes de serem beneficiadas por PDI e recursos computacionais.

1.3 Objetivos

1.3.1 Objetivo geral

Este trabalho tem como objetivo geral especificar, implementar e avaliar um processo de reconhecimento e rastreamento de objetos em imagens ou sequências de imagens, que atuará em um ambiente controlado com objetos de características específicas e constantes. Para tanto, foram utilizadas técnicas de métodos exatos e ferramentas computacionais auxiliares.

1.3.2 Objetivos específicos

1. Investigar técnicas de PDI para reconhecimento de objetos;
2. Especificar um processo de reconhecimento de objetos em imagens;
3. Adaptar o processo para rastrear objetos em imagens sequenciais;
4. Implementar o processo com o auxílio de ferramentas computacionais;
5. Refinar a implementação para atender requisitos de sistemas de tempo real;
6. Realizar e registrar testes de desempenho do processo implementado;
7. Especificar, executar e avaliar estudos de caso para avaliação da aplicabilidade do processo especificado.

2 Materiais e métodos

Os materiais utilizados como referencias para estudo de PDI incluem artigos, livros, tutoriais e trabalhos de conclusão de curso. Estes materiais foram obtidos através do orientador, na biblioteca da própria instituição de ensino e por pesquisas na internet.

As tarefas de codificação deste projeto foram feitas com a linguagem de programação C++ (versão 11) e a biblioteca OpenCV (versão 2.4.9) no ambiente de desenvolvimento integrado Microsoft Visual Studio 2013 utilizando o compilador Microsoft C/C++ Otimizing Compiler Version 18.00.30501 for x86. A codificação, compilação e execução de todo *software* produzido foi realizada em um computador com as seguintes especificações:

- Processador: Intel Core i5 3320M @ 2.60GHz;
- Memória RAM: 2x4 GBytes DDR3 798.1MHz Dual Boost;
- Disco Rídigo: 500GB (5400 RPM).

As imagens e vídeos elaborados no processo de reconhecimento e rastreamento de objetos deste trabalho foram feitos com uma *webcam* Microsoft modelo LifeCam HD 3000 disposta em um suporte que permite que a câmera esteja ortogonalmente alinhada com um plano. Este plano consiste em uma superfície lisa, não inclinada que esteja coberta por folhas de papel EVA iluminadas uniformemente por uma lâmpada fluorescente. Os objetos de interesse presentes nas imagens e vídeos, incluem:

- Peças de brinquedo pedagógica de encaixe de figuras geométricas;
- Camisas diversas;
- Objetos geométricos coloridos feitos de papel EVA.

O projeto foi desenvolvido durante aproximadamente 11 meses. As atividades realizadas durante o desenvolvimento são descritas abaixo. Especificadamente para este trabalho algumas dessas tarefas, apesar de estarem listadas sequencialmente, foram realizadas simultaneamente objetivando o aumento da produtividade, como pode ser visto no cronograma da Tabela 1.

Tarefas realizadas:

1. Estudo direcionado sobre PDI;
2. Pesquisa e avaliação de métodos para reconhecimento de objetos por cor;
3. Pesquisa e avaliação de métodos para reconhecimento de objetos por formas;
4. Pesquisa e avaliação de ferramentas computacionais de PDI;
5. Estudo da ferramenta computacional escolhida (OpenCV);
6. Implementação de software para reconhecimento de objetos por cor;
7. Implementação de software para reconhecimento de objetos por formas;
8. Refinamento do software implementado;
9. Elaboração e realização de testes para validação do software implementado;
10. Documentação das atividades desenvolvidas;
11. Apresentação do trabalho para a banca examinadora;
12. Correções do trabalho sugeridas pela banca examinadora.

Tabela 1 – Cronograma de atividades

	Atividade											
Mês	1	2	3	4	5	6	7	8	9	10	11	12
fev/15	X	X								X		
mar/15	X	X	X							X		
abr/15	X		X	X	X					X		
mai/15				X	X	X				X		
jun/15						X	X			X		
jul/15	X							X		X	X	X
ago/15	X							X	X	X		
set/15	X					X			X	X		
out/15						X	X			X		
nov/15							X			X		
dez/15										X	X	X

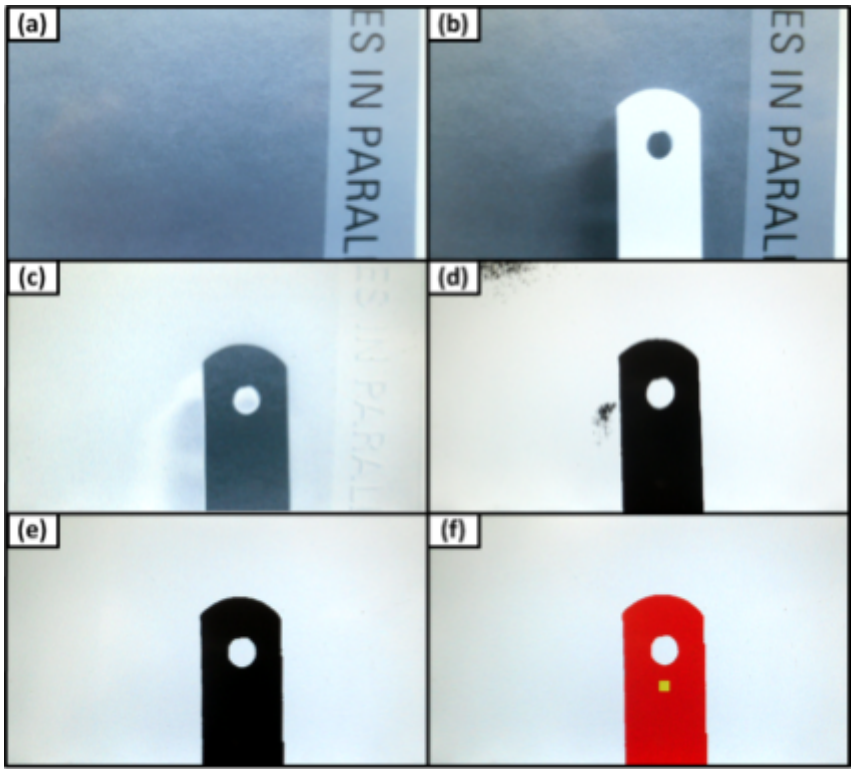
Fonte: Elaborado pelo autor

3 Estado da arte

Devido ao potencial e as possibilidades de atribuir ao computador a capacidade do ser humano de interpretar imagens, vem-se crescendo o número de trabalhos que tentam realizar tal feito. A área de PDI, especialmente quando se trata de reconhecimento de objetos, é recente e isso incentiva a pesquisa e publicação de novos trabalhos. Na literatura, foram encontrados vários trabalhos e alguns são descritos a seguir.

Ferreira (2012) propõe uma arquitetura em hardware para a detecção de objetos em movimento baseada no algoritmo de subtração de fundo. Nesta abordagem, inicialmente é necessário o armazenamento de uma imagem (em tons de cinza) correspondente ao fundo da cena de interesse. A imagem a ser analisada é capturada e filtrada junto com a imagem de fundo para eliminar possíveis ruídos. Em seguida é realizada a subtração entre as duas imagens e uma segmentação no resultado dessa subtração, gerando uma imagem binária. Esta imagem binária é refinada pela aplicação de filtros morfológicos e posteriormente utilizada no cálculo do centro de gravidade do objeto presente na cena. Um exemplo deste procedimento pode ser visualizado na Figura 1.

Figura 1 – Detecção de movimento mediante subtração de fundo.
(a) Imagem de fundo. (b) Imagem atual. (c) Resultado da subtração. (d) Segmentação do movimento. (e) Imagem filtrada. (f) Representação final do objeto em movimento em vermelho, ressaltando o centro de gravidade em amarelo.



Fonte: (FERREIRA, 2012)

Por basear-se em um algoritmo simples e por ter sido implementado em hardware, o resultado final do trabalho apresentou excelente performance. Este alto desempenho é interessante para aplicações de tempo real, mas a simplicidade do algoritmo especificado apresenta algumas restrições. O uso da técnica de subtração de fundo inclui a necessidade de armazenamento de uma imagem que servirá como referência para o fundo da cena, o que pode ser crítico em sistemas com pouca memória disponível ou com memória de baixo desempenho. Reconhecer objetos pelo cálculo do centro de massa de uma imagem binária tem baixo custo computacional, porém não é possível extrair outras informações (cor, formato, tamanho, etc.) do objeto de interesse que não for sua posição na imagem. Além disso, não é possível identificar mais de um objeto simultaneamente, já que o centro de massa vai ser único para cada imagem. Na área da saúde, um método computacional para auxiliar médicos dermatologistas nos diagnósticos de lesões de pele em imagens digitais seguindo as regras de classificação ABCD (Assimetria, Borda, Cor e Diâmetro) é proposto por Oliveira (2012). Neste método utiliza-se difusão anisotrópica na eliminação de ruídos nas imagens e o método de contorno ativo sem bordas (modelo Chan-Vese) para a segmentação das lesões. Na imagem já segmentada são aplicadas operações morfológicas para eliminar orifícios e ruídos externos e também para suavizar as bordas. As características de assimetria, borda, cor e textura são extraídas dessas imagens e utilizadas como entradas para o classificador SVM (Máquina de Vetor de Suporte), técnica baseada em aprendizado estatístico, que caracteriza e classifica as lesões de pele.

Pela natureza da técnica de classificações de lesões de pele, os objetos de interesse da imagem possuem um número considerável de atributos com valores que podem variar muito. Consequentemente, o reconhecimento deve ser mais preciso, e isso implica na criação de uma solução complexa, que passa a exigir o uso de técnicas de classificação mais apuradas, como aprendizado estatístico. Imagina-se que a tarefa de classificação de lesões de pele seja algo pontual, ou seja, o tempo de resposta não é um ponto crítico da solução. Neste caso, o alto custo computacional não é considerado como fator prejudicial, desde que este contribua para uma melhor precisão nos resultados. Da mesma forma como no trabalho de Ferreira (2012), neste também só é possível reconhecer um único objeto, ou no caso, uma lesão, a cada imagem analisada.

Um alternativa que tenta se aproveitar do baixo custo computacional da subtração de fundo e da maior precisão provida por técnicas de aprendizado de máquina foi proposta por Heikkilä e Silvén (2004). Neste projeto, a subtração de imagens é realizada na fase de preprocessamento para tornar mais simples as imagens que ainda passarão por um filtro de Kalman e servirão de amostras para um algoritmo de aprendizagem por quantização vetorial. Tendo como objetivo a contagem e classificação de pedestres e ciclistas em uma via de tráfego exclusivo e misto, que também inclui carros, caminhões e demais veículos, o uso desta abordagem rendeu resultados com precisões que variam entre 80% e 90%.

Neste caso foi possível o reconhecimento e rastreamento de diferentes objetos simultaneamente em um fluxo contínuo de imagens. Porém, o fato de que ciclistas e pedestres podem ter características que os diferenciem, como grupo e como indivíduos, exige que seja realizada uma análise mais apurada, o que justifica o uso de um algoritmo de aprendizado.

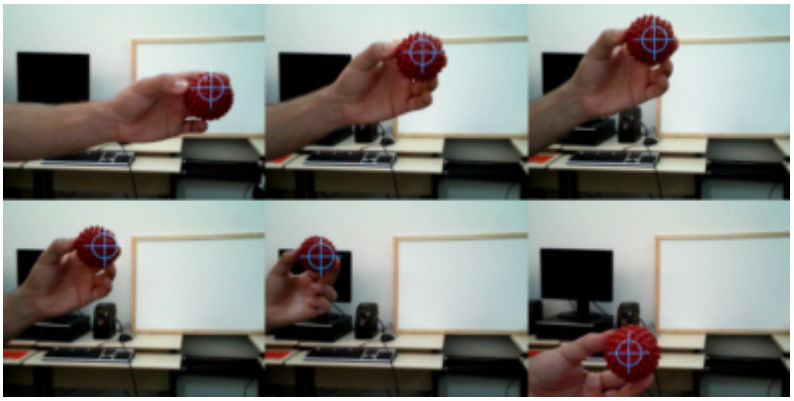
Apesar do processamento ter sido realizado em uma sequência contínua de imagens, o reconhecimento não é feito em tempo real, pois as imagens não são processadas em tempo de captura. As imagens a serem manipulas são *frames* extraídos de um vídeo previamente gravado por um determinado período de tempo. O tempo de processamento de cada *frame* não possui nenhuma restrição temporal, visto que os autores priorizaram precisão à tempo de resposta.

Com um escopo que se aproxima mais ao deste trabalho, Souza (2011) propõe o uso de RNA para a detecção de padrões e do algoritmo de rastreamento de cor conhecido como Camshift para a detecção de trajetória de objetos com formato geométrico simples (triângulos, quadrados e círculos) em um vídeo previamente gravado ou a partir da captura de imagens com uma *webcam*. Para realizar o preprocessamento das imagens e criar a rede de treinamento foram utilizados recursos computacionais disponibilizados pela biblioteca OpenCV.

Mesmo lidando com objetos simples, o autor optou por trabalhar com redes neurais pela capacidade de generalização da abordagem, porém, os testes realizados não foram variados o suficiente para confirmar esta premissa. A amostra testada é muito pequena, oito vídeos para cada formato geométrico, e os objetos a serem detectados são sempre os mesmos. Além do formato e posição, as outras características dos objeto não são identificadas, como tamanho ou cor, mesmo que se esteja utilizando um algoritmo de rastreamento baseado na identificação de cores.

Uma forma de conseguir lidar com cores no reconhecimento de objetos em imagens é apresentada por Oliveira et al. (2013). O autor sugere a conversão da imagem do padrão RGB (*Red, Green and Blue*) para HSV(*Hue, Saturation, and Value*) para facilitar a descoberta de valores de limite inferior e superior que definem o que é uma cor em uma determinada imagem. Esses valores definidos são utilizados no processo de limiarização que gera uma imagem binária contendo somente regiões da cor definida. Esta imagem binária é utilizada no cálculo do centro de massa do objeto encontrado. Na Figura 2 este algoritmo pode ser visto em execução.

Figura 2 – Sequência de quadros que demonstram o rastreio de um objeto de cor vermelha.



Fonte: (OLIVEIRA et al., 2013)

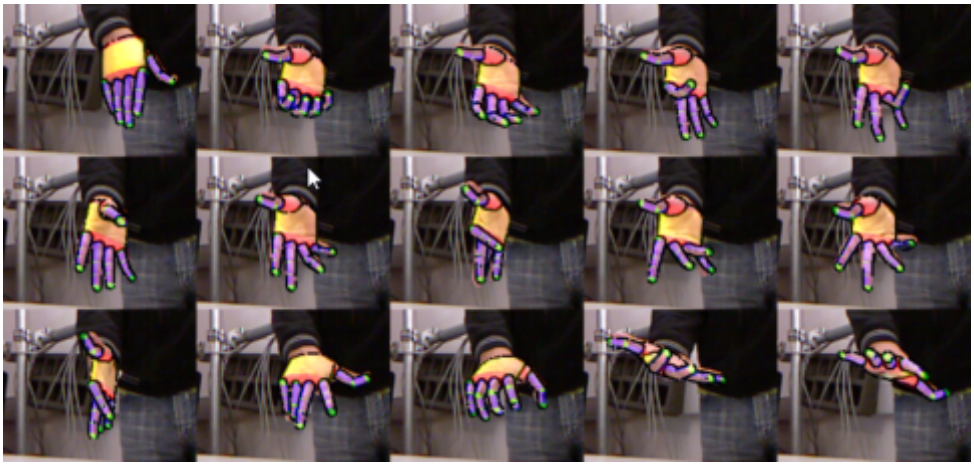
O custo computacional desta abordagem é muito baixo, permitindo que a aplicação satisfaça com facilidade requisitos de tempo real, contudo, as características identificadas do objetos limitam-se somente a cor e posição do centro massa. Pela forma como foi implementada, também não é possível identificar dois ou mais objetos distintos de uma mesma cor, já que o centro de massa é único para cada imagem.

Por último, um conhecido dispositivo que lida com reconhecimento de objetos é o Kinect desenvolvido pela Microsoft, que além de vários outros recursos, tem como destaque sua capacidade de identificar a profundidade dos objetos em cena. Atualmente em sua segunda versão, é um poderoso sensor de movimentos desenvolvido para o *videogame* Xbox One que permite que os jogadores usem o corpo para interagir com o jogo eletrônico, eliminando a necessidade do uso de controles convencionais. Apesar de ser popularmente reconhecido por sua funcionalidade nos videogames, o Kinect possui compatibilidade com computadores e um *Software Development Kit* (SDK) oficial para que projetos possam ser desenvolvidos utilizando os recursos do sensor.

Entre os vários projetos que podem ser mencionados, o desenvolvido por Oikonomidis, Kyriazis e Argyros (2011) especifica um modelo para rastreamento 3D de articulações da mão humana usando o *Kinect*. Devido à complexidade do rastreamento realizado e pela quantidade de informações disponibilizadas pelo sensor, para se alcançar uma performance que aproxime o rastreamento ao tempo real foi necessário uma implementação baseada em GPU (*Graphics Processing Unit*). Então, percebe-se que apesar de poderoso, o Kinect exige uma maior capacidade de processamento para que a aplicação possua um bom tempo de resposta.

A adoção do Kinect como ferramenta de captura de imagens só se torna interessante quando há necessidade do mapeamento de objetos em três dimensões. Além do custo computacional, o sensor possui um custo financeiro consideravelmente alto se comparado à câmeras mais simples que também podem ser utilizadas em projetos de PDI. O resultado final do projeto pode ser verificado na Figura.

Figura 3 – Sequência de quadros que demonstram o rastreio de uma mão e suas articulações.



Fonte: (OIKONOMIDIS; KYRIAZIS; ARGYROS, 2011)

4 Referencial teórico

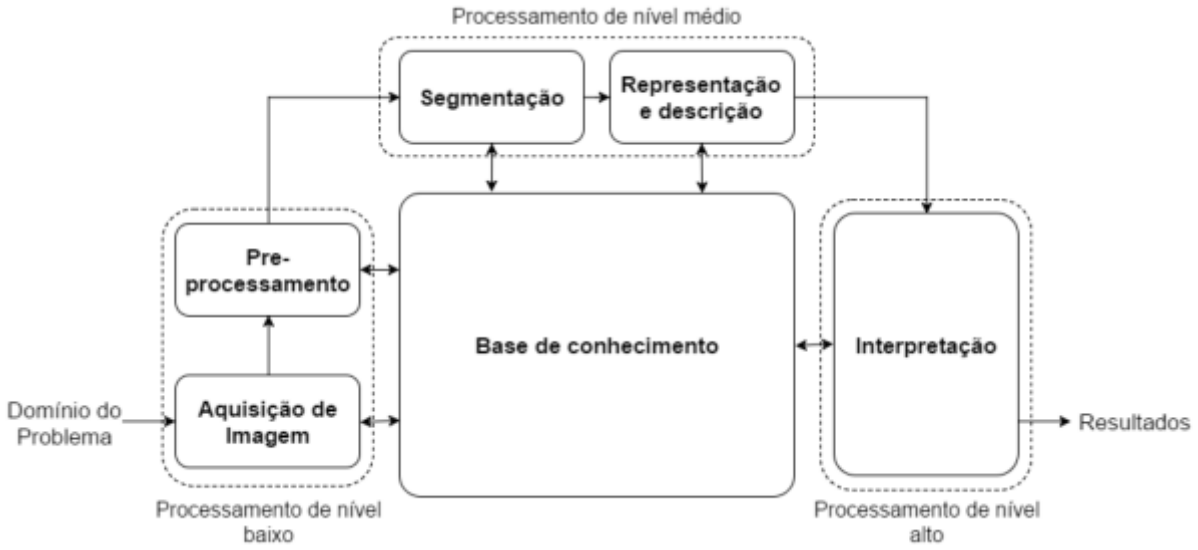
4.1 Processamento digital de imagens

Segundo Gonzales e Woods (2010), PDI constitui a realização de processos digitais em imagens digitais por um computador digital. Basicamente, estes processos podem ser divididos em três categorias: processos de nível baixo, médio e alto.

Processos de nível baixo cobrem a aquisição da imagem e seu tratamento por meio do preprocessamento. Processos de nível médio envolvem a segmentação da imagem em áreas contendo os objetos de interesse, suas definições das formas de representação e por fim, a suas descrições. Processos de nível alto tentam interpretar, “dar sentido” ao conjunto de objetos descritos.

Todos os processos mencionados estão organizados em etapas que constituem o fluxo básico do PDI. O PDI pressupõe a existência de uma base de conhecimento sobre o problema a ser resolvido, cujo tamanho e complexidade podem variar. Como pode ser observado na Figura 4, essa base de conhecimento é utilizada em todas as etapas do processo. (FILHO; NETO, 1999).

Figura 4 – Diagrama geral do PDI.



Fonte: Adaptado de Gonzales e Woods (2010).

Nas subseções a seguir são apresentados em mais detalhes as etapas básicas do PDI, tendo como enfoque o reconhecimento de objetos em imagens.

4.1.1 Aquisição de imagens

A aquisição de uma imagem pode ser tão simples quanto receber uma imagem que já esteja em formato digital ou então envolver a digitalização de sinais analógicos transmitidos por um sensor que converterá informações ópticas do mundo real em sinal elétrico, transformando uma imagem real em uma imagem digital. Em geral, o estágio de aquisição de imagens envolve algum tipo de transformação na imagem, como o redimensionamento ou conversão do padrão de cores (GONZALES; WOODS, 2010).

4.1.1.1 Imagem digital

De acordo com Gonzales e Woods (2010) uma imagem pode ser definida como uma função bidimensional $f(x, y)$ que representa a amplitude f de intensidade do ponto indicado pelo par de coordenadas (x, y). Para uma imagem ser interpretada pelo computador é necessário converter a função $f(x, y)$ para uma forma discreta, este processo é denominado de digitalização.

A digitalização de uma imagem envolve dois passos fundamentais, aquisição e quantização, o resultado gerado é o que se denomina de imagem digital. Na amostragem discretiza-se o domínio de definição da imagem nas direções x (horizontal) e y (vertical), gerando uma matriz de $M \times N$ amostras, respectivamente. Na quantização, escolhe-se o valor inteiro I de intensidade para cada ponto da imagem (PEDRINI; SCHWARTZ, 2008). Desta forma, é possível representar a imagem como uma matriz, como mostrado em 4.1.

$$F(x, y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(M-1,0) \\ f(0,1) & f(1,1) & \cdots & f(M-1,1) \\ \vdots & \vdots & \ddots & \vdots \\ f(0,N-1) & f(1,N-1) & \cdots & f(M-1,N-1) \end{bmatrix} \tag{4.1}$$

Cada ponto desta matriz é denominado *pixel* (aglutinação do termo em inglês *picture element*) e representa a menor unidade de uma imagem digital. A imagem digital possui uma quantidade horizontal e uma quantidade vertical de *pixels*, a esses valores dá se o nome de resolução da imagem (SOLOMON; BRECKON, 2011).

Em uma imagem digital monocromática cada valor do *pixel* corresponde a um escalar de intensidade, permitindo a representação da cor branca, preta ou tons de cinza. Tipicamente, em um imagem colorida cada *pixel* é decomposto em uma tripla de valores que combinados representam a cor naquele determinado ponto. Existem vários modelos que definem a forma de decomposição de cores em imagens digitais, dentre eles, os padrões RGB e HSV.

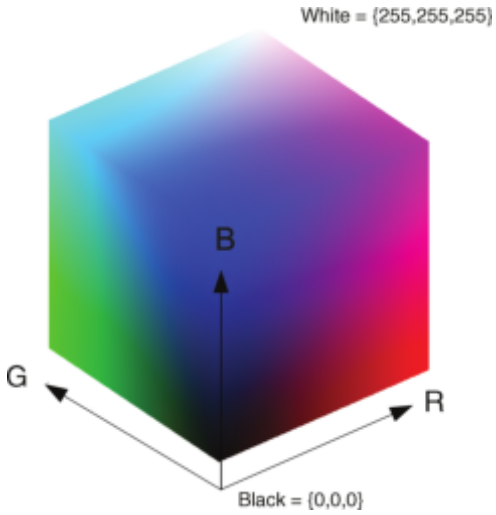
4.1.1.1.1 Padrão RGB

Imagens no padrão RGB são matrizes 3D que podem ser visualizadas como três matrizes 2D distintas, cada um correspondendo a um dos três canais de cor:

- ***R**ed* - Vermelho;
- ***G**reen* - Verde;
- ***B**lue* - Azul.

Considerando que todas as cores possam ser representadas com a descrição RGB, percebe-se que o espaço RGB é um espaço de cor 3D (cubo) com eixos R, G e B, como pode ser visto na Figura 5.

Figura 5 – Espaço de cor RGB como um cubo 3D.

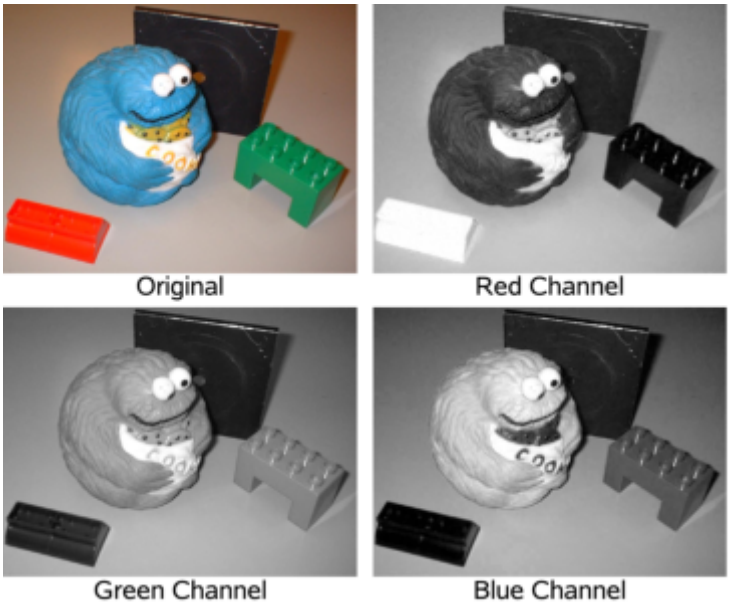


Fonte: (SOLOMON; BRECKON, 2015)

Nas imagens digitais, para cada *pixel* é reservado um espaço de armazenamento de 3 x 1 *byte*. Com 1 *byte* é possível representar o valor de intensidade de uma cor primária em uma faixa de 0 à 255, sendo 0 completamente escuro e 255 completamente intenso. Essa separação de canais pode ser visualizada na Figura 6.

O padrão RGB é o mais utilizado pois corresponde às três cores primárias que são combinadas para exibição de imagens em um monitor ou dispositivos similares (SOLOMON; BRECKON, 2011).

Figura 6 – Imagem transformada e exibida no espaço RGB.



Fonte: (SOLOMON; BRECKON, 2015)

4.1.1.1.2 Padrão HSV

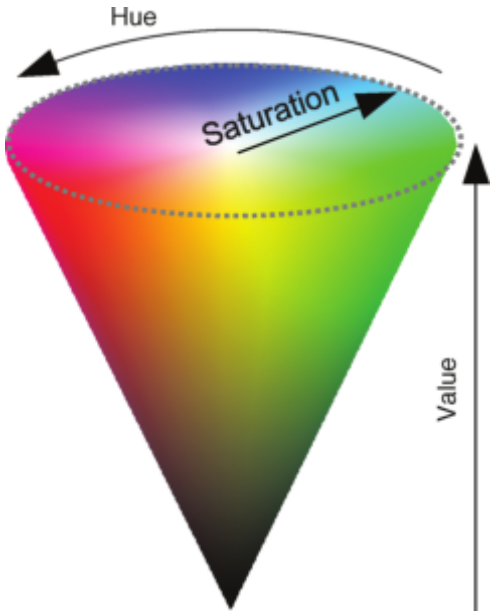
O espaço contendo todas as cores no padrão HSV geralmente é representado como um cone 3D, como pode ser visto na Figura 7. Tal qual o modelo RGB, este também possui três parâmetros distintos que podem ser descritos da seguinte forma:

- ***H****ue* - Matiz ou tonalidade;
- ***S****aturation* - “pureza da cor” (quantidade de branco misturado com a cor);
- ***V****alue* - Brilhância ou luminância da cor.

Estes valores podem ser obtidos a partir da conversão de valores RGB e vice-versa Gonzales e Woods (2010).

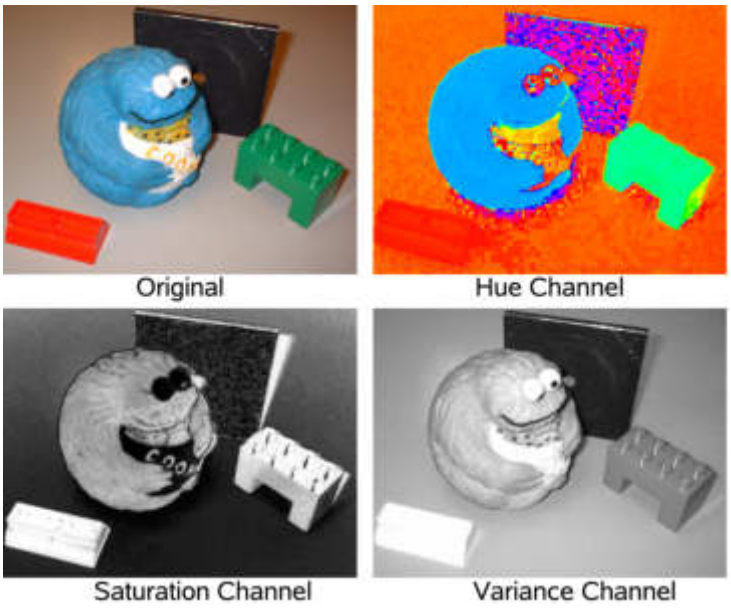
Este modelo é uma alternativa para representar imagens em cores reais de forma mais natural à percepção e compreensão humana das cores. Este fenômeno pode ser facilmente visualizado ao examinar cada canal de cor da imagem dividida pelos canais RGB (Figura 6) e HSV (Figura 8). Nota-se que os objetos da imagem estão contidos de forma mais consistente no campo de matiz (*Hue Channel*) no padrão HSV, do que nos canais da representação RGB, apesar de condições variáveis de iluminação (SOLOMON; BRECKON, 2011). Por este motivo o HSV é mais utilizado em sistemas de PDI baseados nos modelos de percepção de cor pelo ser humano (FILHO; NETO, 1999).

Figura 7 – Espaço de cor HSV como um cone 3D.



Fonte: (SOLOMON; BRECKON, 2015)

Figura 8 – Imagem transformada e exibida no espaço HSV.



Fonte: (SOLOMON; BRECKON, 2015)

4.1.1.2 Redimensionamento

O redimensionamento (ou *resizing*) de imagens é um processo do PDI cujo objetivo é a alteração das dimensões de uma imagem. A utilidade do redimensionamento inclui, entre outros, ajuste de imagens para exibição adequada em diferentes dispositivos e redução do tamanho de uma imagem para diminuir o custo computacional de operações aplicadas sobre ela. Quando feito de forma proporcional (ampliação ou redução por um fator), o redimensionamento de imagens leva o nome de dimensionamento (ou *scaling*) (FILHO; NETO, 1999).

Para realizar o redimensionamento de uma imagem, em geral, trata-se a imagem como uma função de duas variáveis que mapeia uma coordenada (x, y) para uma cor e interpola-se essa função. Porém, como a interpolação de imagens é uma técnica que estima a cor de um determinado *pixel*, a imagem resultante pode sofrer vários efeitos colaterais, como ruídos, variação de contraste, não preservação de bordas ou texturas e suavização exagerada (WITTMAN, 2005).

Há várias tipos de interpolação que podem de ser utilizados no redimensionamento de imagens, cada um tendo suas vantagens e desvantagens, tanto em relação ao custo computacional quanto a intensidade da presença de efeitos colaterais. Na Figuras 9, 10 e 11 é possível visualizar exemplos de redimensionamento usando diferentes tipos de interpolação, estes estão organizados da seguinte forma:

(a) Imagem reduzida em 50% do tamanho original;

(b) Imagem original;

(c) Imagem ampliada em 50% do tamanho original.

Os valores numéricos explicitados correspondem ao tempo gasto no redimensionamento das imagens em milissegundos (ms).

Figura 9 – Redimensionamento utilizando interpolação por vizinho mais próximo.

(a) 1 ms

(b)

(c) 6 ms

Fonte: Adaptado de Gonzales e Woods (2010).

Figura 10 – Redimensionamento utilizando interpolação linear.

(a) 5 ms

(b)

(c) 9 ms

Fonte: Adaptado de Gonzales e Woods (2010).

Figura 11 – Redimensionamento utilizando interpolação Lanczos sobre uma vizinhança de (8×8) *pixels*.

(a) 5 ms

(b)

(c) 20 ms

Fonte: Adaptado de Gonzales e Woods (2010).

4.1.2 Preprocessamento

O objetivo básico desta etapa é manipular a imagem de forma que o resultado obtido seja mais adequado do que o original para uma aplicação específica. É importante ressaltar que no PDI "ser adequado" é algo bastante subjetivo, pois o processo de adequação de uma imagem está diretamente relacionamento ao problema em que ela está envolvida (GONZALES; WOODS, 2010). Algumas das técnicas utilizadas na etapa de preprocessamento serão apresentadas a seguir.

4.1.2.1 Suavização

Suavização é a etapa de PDI onde são aplicados filtros de suavização que causam borramento e, por consequência, removem ruídos em imagens (GONZALES; WOODS, 2010). Ruído é o nome dado a uma pequena variação (aleatória) sofrida pelo sinal em torno do seu valor verdadeiro, em decorrência de fatores externos ou internos durante o PDI. Em geral, é possível dividir os tipos de ruído em duas categorias distintas:

Ruído pimenta e sal: (Figura 12b) causado pela introdução de *pixels* puramente brancos ou puramente pretos em posições aleatórias na imagem. Atualmente este tipo de ruído é incomum devido ao avanço da tecnologia de sensores de captura de imagens, mas ainda podem ser vistos na forma de falhas em alguns sensores de câmeras (SOLOMON; BRECKON, 2011).

Ruído gaussiano: (Figura 12c) neste caso, a variação aleatória do sinal em torno do seu valor verdadeiro segue uma distribuição gaussiana ou normal. É modelo de ruído mais usado no PDI por ser capaz de descrever a maioria dos ruídos existentes (SOLOMON; BRECKON, 2011).

Figura 12 – Comparação dos dois tipos básicos de ruído em imagem.

(a) Imagem original. (b) Ruído Pimenta e Sal. (c) Ruído Gaussiano.

Fonte: Adaptado de Gonzales e Woods (2010).

Em geral, os filtros de suavização estimam um novo valor para um *pixel* a partir de seus *pixels* de vizinhança. O cálculo deste valor pode ser feito utilizando diversas abordagens, como (SOLOMON; BRECKON, 2011):

- Cálculo do valor médio da vizinhança (Filtragem pela média);
- Cálculo da mediana estatística da vizinhança (Filtragem pela mediana);
- Cálculo utilizando funções gaussianas (Filtragem gaussiana);

Dentre todos os tipos de filtros citados, a filtragem gaussiana provavelmente é a mais útil de todas, apesar de não ser a mais rápida (BRADSKI; KAEHLER, 2008). A justificativa para isso pode ser observada na Figura 13. Dentre as três filtragens exemplificadas, a gaussiana foi a que melhor se saiu em eliminar ruídos e simultaneamente manter um alto nível de nitidez da imagem.

Figura 13 – Comparação entre os tipos de filtros de suavização.

(a) Imagem original.

(b) Filtragem pela média.

(c) Filtragem pela mediana.

(d) Filtragem gaussiana.

Fonte: Adaptado de Gonzales e Woods (2010).

No filtro gaussiano a imagem é filtrada a partir da especificação de dois parâmetros livres:

1. Dimensões da vizinhança ($N \times N$);
2. Desvio padrão da função gaussiana (σ);

Onde o grau de suavização é controlado pelo valor do desvio padrão. O custo computacional da aplicação do filtro gaussiano está diretamente relacionado ao tamanho da vizinhança utilizada (SOLOMON; BRECKON, 2011).

4.1.2.2 Operações morfológicas

Tendo como base a teoria dos conjuntos, a morfologia matemática tem como princípio básico identificar e extrair as informações relativas à geometria e à topologia de um conjunto desconhecido (uma imagem), por transformações através de outro conjunto definido, chamado elemento estruturante (FILHO; NETO, 1999).

O elemento estruturante é uma matriz retangular de *pixels* contendo valores 0 ou 1. Um *pixel* central desta matriz (de valor 1) servirá referência e os outros (também de valor 1) definem uma vizinhaça de *pixels* que será levada em consideração durante as operações morfológicas. A escolha de um elemento estruturante adequado é um dos desafios do processamento morfológico (SOLOMON; BRECKON, 2011).

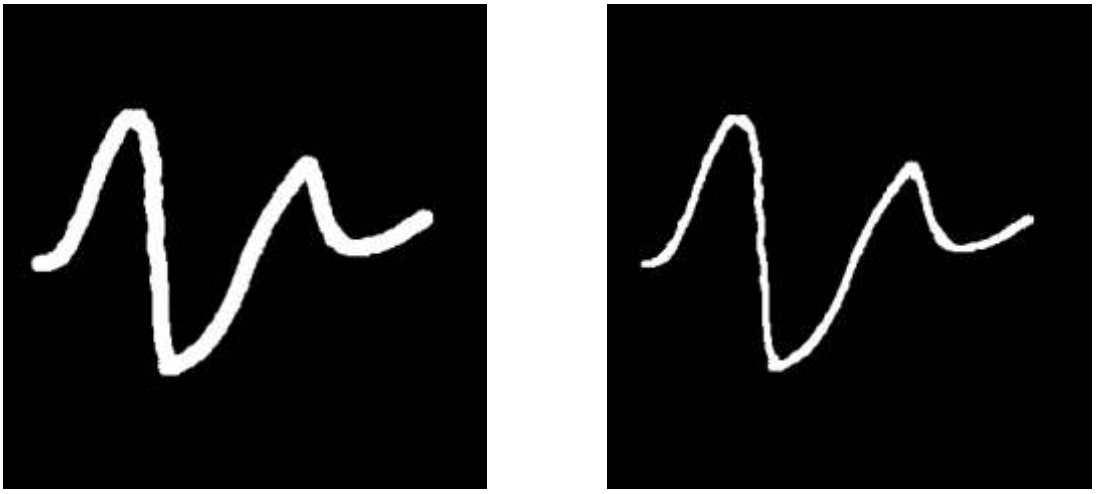
Segundo Solomon e Breckon (2011), operações morfológicas podem ser aplicados em qualquer tipo de imagem, mas geralmente são utilizadas em imagens binárias. Em imagens binárias, os conjuntos em questão são membros do espaço Z^2 em que cada elemento é um vetor bidimensional, cujas coordenadas são (x, y) de um *pixel* branco (por convenção, de valor 1) de uma imagem. Os operadores morfológicos mais básicos e importantes são os de dilatação e de erosão, outros procedimentos morfológicos mais sofisticados podem ser reduzidos à sequências dessas operações básicas (GONZALES; WOODS, 2010).

4.1.2.2.1 Erosão

No processo de erosão de uma imagem binária posiciona-se o *pixel* central do elemento estruturante em cada *pixel* branco da imagem. Se qualquer um dos *pixels* de vizinhança relativa ao elemento estruturante for preto, o *pixel* branco passará a ser preto. Formalmente, define-se a erosão de uma imagem A pelo elemento estruturante B como $A \ominus B$ (SOLOMON; BRECKON, 2011).

O efeito visual da aplicação da operação de erosão em uma imagem pode ser observado na Figura 14. Nota-se que a operação fez com que as áreas em branco fossem reduzidas.

Figura 14 – Aplicação da operação de erosão em imagem binária.



(a) Imagem original. (b) Imagem após a erosão.

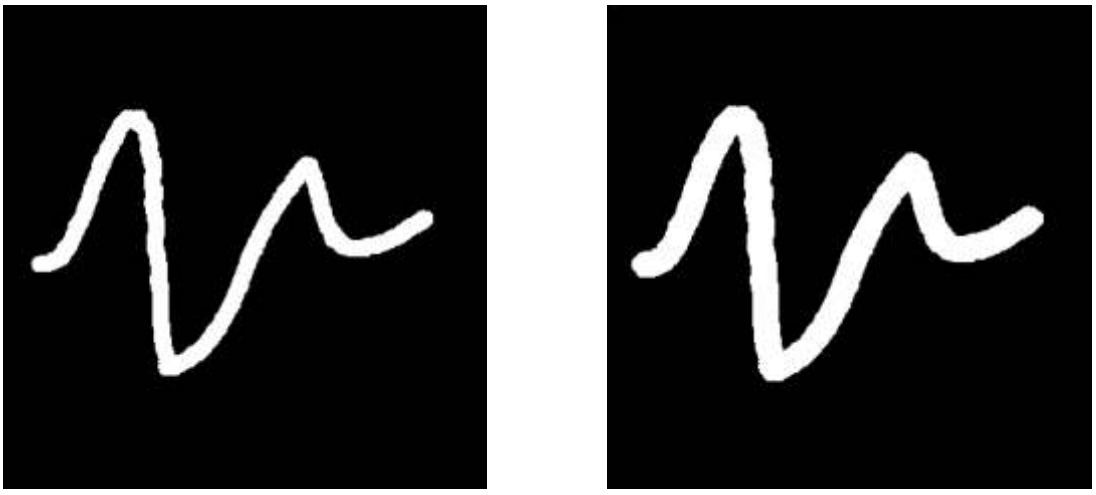
Fonte: Elaborado pelo autor.

4.1.2.2.2 Dilatação

No processo de dilatação de uma imagem binária posiciona-se o *pixel* central do elemento estruturante em cada *pixel* preto da imagem. Se qualquer um dos *pixels* de vizinhança relativa ao elemento estruturante for branco, o *pixel* preto passará a ser branco. Formalmente, define-se a dilatação de uma imagem A pelo elemento estruturante B como $A \oplus B$ (SOLOMON; BRECKON, 2011).

O efeito visual da aplicação da operação de dilatação em uma imagem pode ser observado na Figura 15. Nota-se que a operação fez com que as áreas em branco expandissem.

Figura 15 – Aplicação da operação de dilatação em imagem binária.



(a) Imagem original. (b) Imagem após a dilatação.

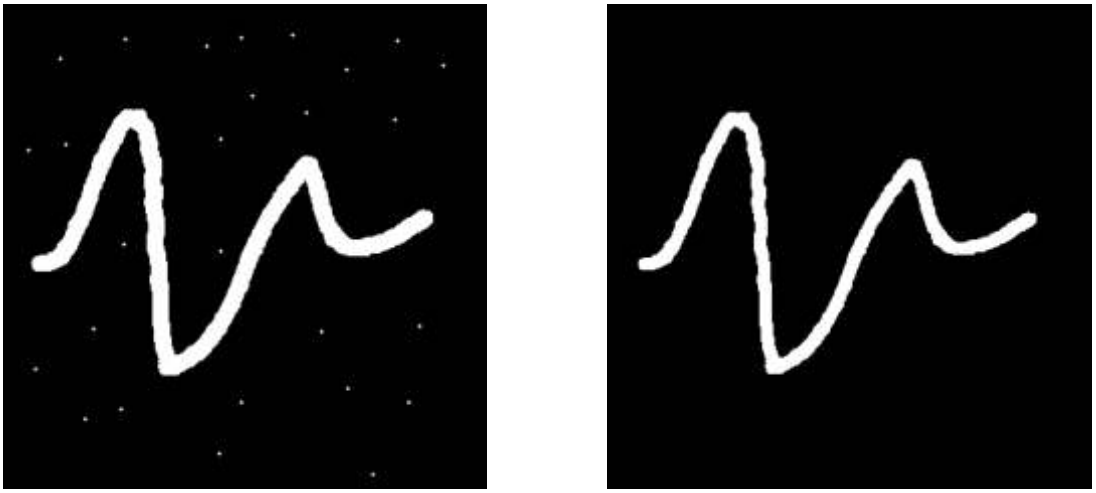
Fonte: Elaborado pelo autor.

4.1.2.2.3 Abertura morfológica

Abertura é o nome dado à operação morfológica de erosão seguida de dilatação utilizando o mesmo elemento estrutural. Formalmente, define-se a abertura formológica de uma imagem A pelo elemento estruturante B como $A \circ B = (A \ominus B) \oplus B$ (SOLOMON; BRECKON, 2011).

O efeito visual da aplicação da abertura morfológica em uma imagem pode ser observado na Figura 16. Nota-se que a operação fez com que pequenas áreas em branco, que poderiam ser consideradas como ruídos, fossem reduzidas sem alterar bruscamente o formato da área principal.

Figura 16 – Aplicação da operação de abertura morfológica em imagem binária.



(a) Imagem original. (b) Imagem após a abertura morfológica.

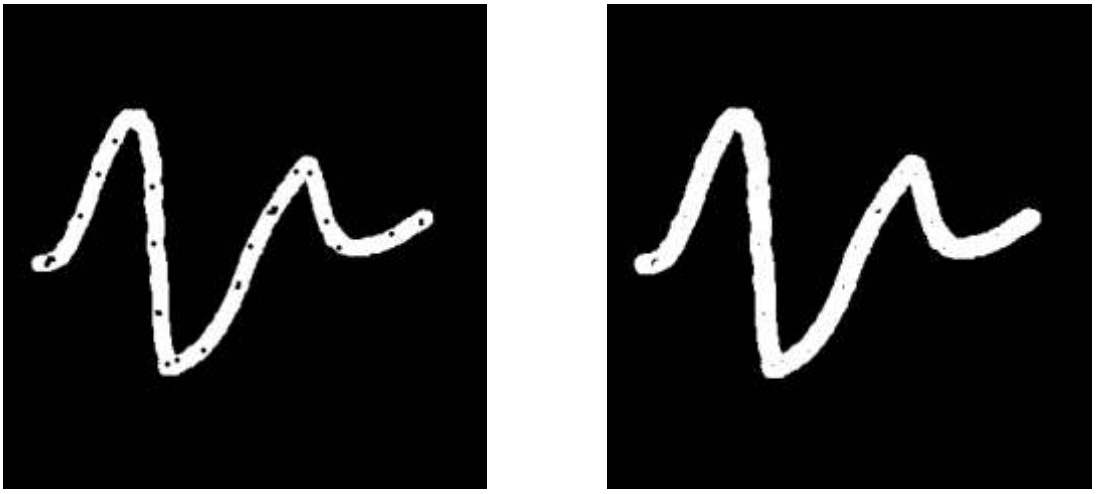
Fonte: Elaborado pelo autor.

4.1.2.2.4 Fechamento morfológico

Fechamento é o nome dado à operação morfológica de dilatação seguida de erosão utilizando o mesmo elemento estrutural. Formalmente, define-se a abertura formológica de uma imagem A pelo elemento estruturante B como $A \bullet B = (A \oplus B) \ominus B$ (SOLOMON; BRECKON, 2011).

O efeito visual da aplicação do fechamento morfológica em uma imagem pode ser observado na Figura 17. Nota-se que a operação fez com que pequenas áreas em preto, que poderiam ser consideradas como falhas, fossem preenchidas sem alterar bruscamente o formato da área principal.

Figura 17 – Aplicação da operação de fechamento morfológico em imagem binária.



(a) Imagem original. (b) Imagem após o fechamento morfológico.

Fonte: Elaborado pelo autor.

4.1.3 Segmentação

A segmentação é e etapa que subdivide uma imagem em regiões ou objetos que a compõe. O nível de detalhamento em que essa subdivisão é realizada dependerá do problema ser a resolvido. Então, a segmentação deve parar quando os objetos ou regiões de interesse da aplicação forem detectados (GONZALES; WOODS, 2010). Por exemplo, se o objetivo for a inspeção automatizada de frutas, o detalhamento da segmentação se limita ao reconhecimento de manchas que possivelmente signifiquem uma má qualidade do fruto.

Segundo Gonzales e Woods (2010), a segmentação de imagens não triviais é uma das tarefas mais difíceis no PDI. A precisão da segmentação irá definir o sucesso ou o fracasso dos procedimentos da análise computadorizada de imagens. Quando viável, é recomendado adotar medidas para se estabelecer um ambiente controlado possível, porém, quando não se tem esse controle sobre o ambiente, deve-se considerar a escolha de sensores que irão realçar os objetos de interesse na cena e diminuir a representação de detalhes irrelevantes. Como exemplos de situações em que a escolha de um sensor específico beneficia a o processo de segmentação, podem ser citados as aplicações de detecção de tropas em movimento utilizadas pelos militares que fazem uso de sensores infravermelho ao invés de câmeras convencionais.

A maioria dos algoritmos de segmentação baseiam-se em uma das seguintes propriedades básicas de valores de intensidade: descontinuidade e similaridade. Na primeira categoria, a abordagem consiste em dividir uma imagem com base nas mudanças bruscas de intensidade, como as bordas. As abordagens da segunda categoria são baseadas na divisão da imagens em regiões de acordo com um conjunto de critério pré-definidos, como acontece no caso de limiarização de imagens (GONZALES; WOODS, 2010).

4.1.3.1 Limiarização

Dentre as várias técnicas de segmentação, a limiarização é uma das mais simples. A limiarização consiste na classificação dos *pixels* de uma imagem de acordo com a especificação de um ou mais limiares (PEDRINI; SCHWARTZ, 2008). A partir da definição desses limiares é possível separar a imagem em grupos de interesse, como foi no caso da Figura 18. Uma imagem no padrão HSV foi limiarizada de acordo com valores que segmentam regiões de cor vermelha (Figura 18b) e regiões de cor azul (Figura 18c), diferenciando-as do fundo de cor branca.

Figura 18 – Imagem limiarizada a fim de segmentar objetos por cor.

(a) Imagem original. (b) Objetos vermelhos. (c) Objetos azuis.

Fonte: Elaborado pelo autor.

A limiarização de imagens em padrão HSV é útil pois facilitam a descoberta de regiões de determinadas cores em uma imagem. Este tipo de limiarização pode ser formalmente definida pela equação 4.2, onde $(H, S, V)_{min}$ é o conjunto de valores que define o limiar inferior para os canais *Hue*, *Saturation* e *Value*, da mesma forma que $(H, S, V)_{max}$ define o limiar superior para os mesmo canais. Cada *pixel* de coordenada (x, y) tem seus valores do conjunto $(H, S, V)_{(x,y)}$ avaliados a fim de definir se estão dentro dos limiares ou não, ou seja, se este *pixel* faz parte ou não do grupo definido.

$$f(x,y) = \begin{cases} 1 & (H,S,V)_{min} \leq (H,S,V)_{(x,y)} \leq (H,S,V)_{max} \\ 0 & \text{caso contrário} \end{cases} \tag{4.2}$$

4.1.3.2 Detecção de bordas

A detecção de bordas é um dos aspectos mais relevantes e mais estudados do PDI. Se for possível a determinação de uma fronteira ou contorno de um objeto pela identificação de todas as suas bordas, é possível segmentá-lo. Conceitualmente, a detecção de bordas parece um problema bastante trivial, já que as bordas são regiões de transição de intensidade entre um objeto e outro, contudo, apesar da simplicidade conceitual, a detecção de bordas de bordas ainda é um campo de pesquisa ativo (SOLOMON; BRECKON, 2011).

Dentre os vários algoritmos desenvolvidos em pesquisas, o método proposto por Canny (1986) é atualmente reconhecido como o método mais completo de detecção de bordas (SOLOMON; BRECKON, 2011). Canny (1986) desenvolveu um método de detecção de bordas que satisfaz três critérios básicos:

1. Baixa taxa de erro;
2. Distância mínima entre os *pixels* encontrados e os da borda real;
3. Deve haver somente uma resposta a uma borda.

Este algoritmo faz uso de um limiar máximo e um liminar mínimo para considerar ou rejeitar certas bordas identificadas. Diferentes limiares resultam em diferentes bordas detectadas, como pode ser visto na Figura 19. A escolha destes limiares deve ser feita de acordo com as bordas que se deseja detectar, mas em geral, Canny (1986) recomenda a escolha de valores que criem faixas de proporção 1:2 ou 1:3.

Figura 19 – Aplicação do algoritmo de Canny (1986) para detecção de bordas.

(a) Imagem original. (b) L. MIN: 30; L. MAX: 90. (c) L. MIN: 60; L. MAX: 180.

Fonte: Elaborado pelo autor.

4.1.4 Representação e descrição

Realizada a segmentação de uma imagem em regiões de interesse, o agregado de *pixels* segmentados resultante deve ser representado e descrito de forma adequada para o futuro processamento computacional (GONZALES; WOODS, 2010).

Para representar uma região é necessário escolher de que forma será realizada esta representação. Em geral, existem duas opções: (1) representá-la em termos de sua características externas (contornos) ou (2) em termos de suas características internas (pixels que a constituem). Após escolhida forma de representação, é necessário definir como as regiões serão descritas (GONZALES; WOODS, 2010). Por exemplo, uma região pode ser representada por sua fronteira e esta pode ser descrita por uma lista de seus pontos extremos.

Algumas das técnicas envolvidas no processo de representação e descrição de regiões de imagens serão detalhadas nas subseções a seguir.

4.1.4.1 Seguidores de fronteira (contorno)

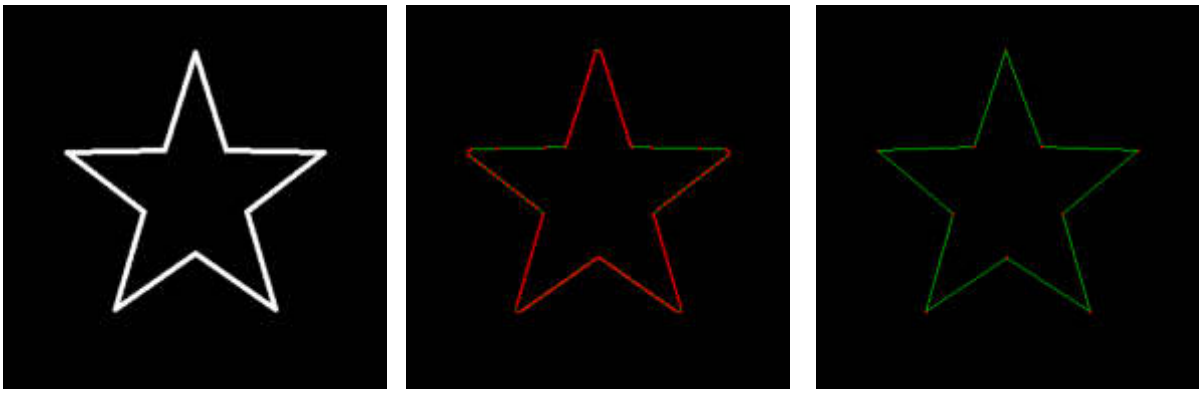
Seguidores de fronteiras são técnicas fundamentais no PDI. Essas técnicas são caracterizadas pela capacidade de identificar e percorrer os contornos presentes em imagens. Quando contornos são conexos, estes são denominados de fronteiras. Em geral, fronteiras podem ser definidas como uma lista de pontos que representam uma curva em uma imagem. Então, um algoritmo que seja capaz de percorrer uma fronteira é capaz de descrever os pontos que a compõem (GONZALES; WOODS, 2010).

O desafio de descrever computacionalmente as bordas presentes em uma imagem é recorrente no PDI e revisando literatura é possível encontrar diversas propostas de algoritmos que realizam esta tarefa, entre eles, os propostos por Suzuki et al. (1985) possuem grande credibilidade por sua precisão, qualidade de descrição e desempenho. Estes algoritmos são capazes de descrever contornos em imagens binárias e organizá-lo em estruturas de dados (listas ou árvores) de forma que possam ser analisados computacionalmente com facilidade.

Na Figura 20b é possível visualizar o contorno de um objeto descrito pelos algoritmos de Suzuki et al. (1985). Nota-se que a curva definida é compatível com a fronteira do objeto da imagem original (Figura 20a), porém, a quantidade de pontos encontrados é muito grande, o que implica em uma maior complexidade nas análises a serem feitas sobre o contorno descrito.

Se as curvas descritas pelas técnicas de seguidores de fronteira forem simplificadas, consequentemente, simplificam-se os processos que analisam estas curvas. Para isso, aplica-se algoritmos que aproximem as curvas encontradas pelos seguidores de fronteiras a uma curva com menos vértices. Ramer (1972) propõe um algoritmo para encontrar uma curva aproximada com a precisão definida por uma distância máxima entre a curva original e a nova curva. Observa-se na Figura 20c que o algoritmo é capaz de simplificar a curva previamente encontrada (Figura 20b) de forma satisfatória, pois ela ainda consegue descrever bem a fronteira da imagem original (Figura 20a).

Figura 20 – Aplicação de algoritmo seguidor de fronteira e simplificação de curvas.



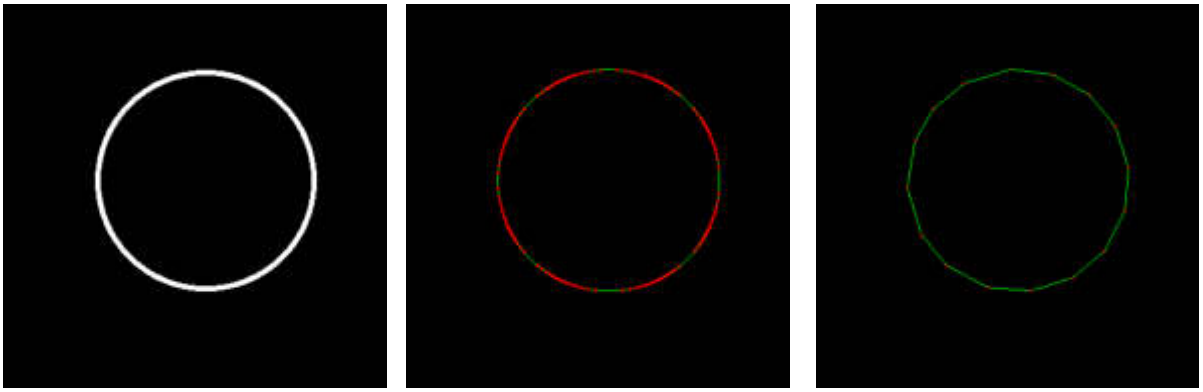
(a) Imagem original. (b) 381 pontos encontrados. (c) 10 pontos encontrados.

Fonte: Elaborado pelo autor.

4.1.4.2 Descrição de círculos

Em alguns casos, a lista dos pontos que compõe uma fronteira talvez não seja a melhor opção de representação de certas regiões, como por exemplo, em círculos. Como pode ser notado na Figura 21b, a aplicação dos algoritmos de Suzuki et al. (1985) resulta em uma descrição precisa da fronteira, porém com um número muito grande de pontos. Se aplicado o algoritmo de Ramer (1972) para diminuir este número de pontos pode-se observar na Figura 21c que a descrição da fronteira é próxima da original (Figura 21a), mas que não é tão precisa.

Figura 21 – Aplicação de algoritmo seguidor de fronteira e simplificação de curvas em círculos.



(a) Imagem original. (b) 288 pontos encontrados. (c) 16 pontos encontrados.

Fonte: Elaborado pelo autor.

A forma mais simples de definição de um círculo baseia-se na sua posição central do plano e o valor do seu raio, esta também é a representação ideal para a descrição de círculos no PDI. Para tanto, Yuen et al. (1990) propõe um algoritmo de detecção e descrição de círculos utilizando a transformada de Hough. Este utiliza limiares inferiores e superiores para definir a aceitação ou não de certos círculos como reais e resulta na descoberta dos pontos centrais e raios dos círculos descobertos na imagem. Com estes valores, é possível definir de forma mais precisa o perímetro do círculo com menos informações, como pode ser visto na Figura 22b.

Figura 22 – Aplicação do algoritmo de Yuen et al. (1990) para detectar e descrever círculos.

(a) Imagem original. (b) Círculo encontrado.

Fonte: Elaborado pelo autor.

4.1.5 Interpretação

Ao final da etapa de representação e descrição é criado um conjunto contendo os possíveis objetos de interesse identificados na imagem, porém, o ato de reconhecimento vai além do processo de identificação da presença dos objetos na imagem, é necessário validar e classificar esses objetos, fazer com eles “tenham sentido”. É na etapa de interpretação onde esses processos acontecem (GONZALES; WOODS, 2010).

A validação e a classificação de objetos estão intrinsecamente relacionadas, sendo que a primeira ocorre por consequência da segunda. Em geral, os objetos de interesses da imagem podem ser divididos em categorias distintas bem definidas. Se não for possível classificar um objeto do conjunto como pertencente a uma dessas categorias, este é descartado, ou seja, não é validado pelo sistema.

Essas categorias geralmente são definidas pelas características únicas dos objetos de interesse. Por exemplo, em um conjunto de frutas, pode-se definir uma categoria rotulada de “Maçãs” contendo todos os objetos de cor vermelha e uma outra categoria rotulada de “Bananas” contendo todos os objetos de cor amarela. Nota-se que distinguir maçãs de bananas pelas cores é uma tarefa simples, porém, não se deve contar com a presença exclusiva de bananas e maçãs nas imagens a serem analisadas. Seguindo as regras de classificação definidas para “Bananas” e “Maças” um melão também seria reconhecido como uma banana, já que também é da cor amarela, mas sabe-se bem que melões não são bananas. Uma solução para este problema de falsa classificação é a identificação e especificação de mais características que definem um objeto no processo de classificação. Ou seja, para um objeto ser classificado como uma banana, além de ser amarelo, ele também deve ter o formato de lua crescente e para ser uma maçã, além de ser vermelho, deve ter o formato de circunferência.

Quando os objetos são muito diversos e irregulares, pode ser necessário desenvolver algum tipo de inteligência que consiga organizar os objetos e agrupá-los em categorias. Em geral, o desenvolvimento dessa inteligência está relacionado ao uso de técnicas de aprendizado de máquina e com isso acrescenta-se ao sistema de PDI toda a complexidade do processo de aprendizagem computacional. Alguns exemplos de sistemas de PDI que aplicam essas técnicas podem ser encontrados no capítulo Estado da Arte deste documento (GONZALES; WOODS, 2010). Quando os objetos a serem avaliados possuem características constantes e bem definidas, existem outras alternativas para resolver o problema de classificação.

Uma da alternativas mais simples é o uso de métodos exatos. Esses métodos se baseiam em características peculiares dos objetos para definir se determinado objeto pertence ou não a uma determinada classe (SILVA, 2014). O descobrimento dessas características peculiares é um dos desafios no desenvolvimento de sistemas de PDI. Enquanto o uso de técnicas de aprendizado de máquina naturalmente já produz conjuntos de características que definem tipos de objetos, os métodos exatos dependem de um processo de identificação empírica para realizar essa descoberta. No exemplo já mencionado de classificação de frutas, a descoberta de que bananas são amarelas e possuem formato de lua crescente e que maças são vermelhas e possuem formato de circunferência aconteceu de forma empírica. Descobertas estas características peculiares, elas servem de base na definição de um algoritmo de classificação

de objetos. Nota-se que este processo classifica de forma mais rígida os objetos, mas que também é capaz de criar certas aproximações de classificação, por exemplo, vários tons de cor amarelo podem ser a cor de um objeto do tipo banana.

Ao fim desta etapa, todos os objetos válidos reconhecidos pelo sistema estão definidos e descritos. O que será feito com as informações desses objetos depende do contexto em que o sistema está envolvido, porém, essas etapas já estão fora do escopo básico do PDI.

4.2 OpenCV

OpenCV é uma biblioteca *open source* de visão computacional. Escrita em C e C++ que funciona em Linux, Windows e Mac OS X e que possui interfaces para C, C++, Python, Ruby, Java, MATLAB e outras linguagens. O OpenCV foi originalmente projetado visando eficiência computacional com foco em aplicações de tempo real (BRADSKI; KAEHLER, 2008).

Com sua primeira versão lançada em 2006, atualmente a biblioteca possui mais de 2500 algoritmos otimizados que podem ser utilizados para detectar e reconhecer rostos, identificar objetos, classificar ações humanas em vídeos, extrair modelos 3D de objetos, seguir os movimentos dos olhos, reconhecer cenários e estabelecer marcadores para cobri-los com realidade aumentada, remover olhos vermelhos de imagens tiradas com *flash*, etc (BRADSKI; KAEHLER, 2008). A biblioteca é estruturada em cinco componentes, sendo eles:

- **CV**: possui as funcionalidades básicas de PDI e algoritmos de visão computacional;
- **ML**: possui algoritmos de aprendizado de máquina, *clustering*, classificação e análise de dados;
- **HighGUI**: possui funções relacionadas à construção de interfaces e a captura e armazenamento de imagens e vídeos;
- **CXCORE**: núcleo de estruturas e algoritmos básicos do OpenCV.
- **CVAUX**: possui algoritmos de visão computacional em fase de experimentação.

É válido mencionar que o OpenCV é utilizado extensivamente por grandes organizações, como Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda e Toyota, ou por *startups* de visão computacional (BRADSKI, 2000).

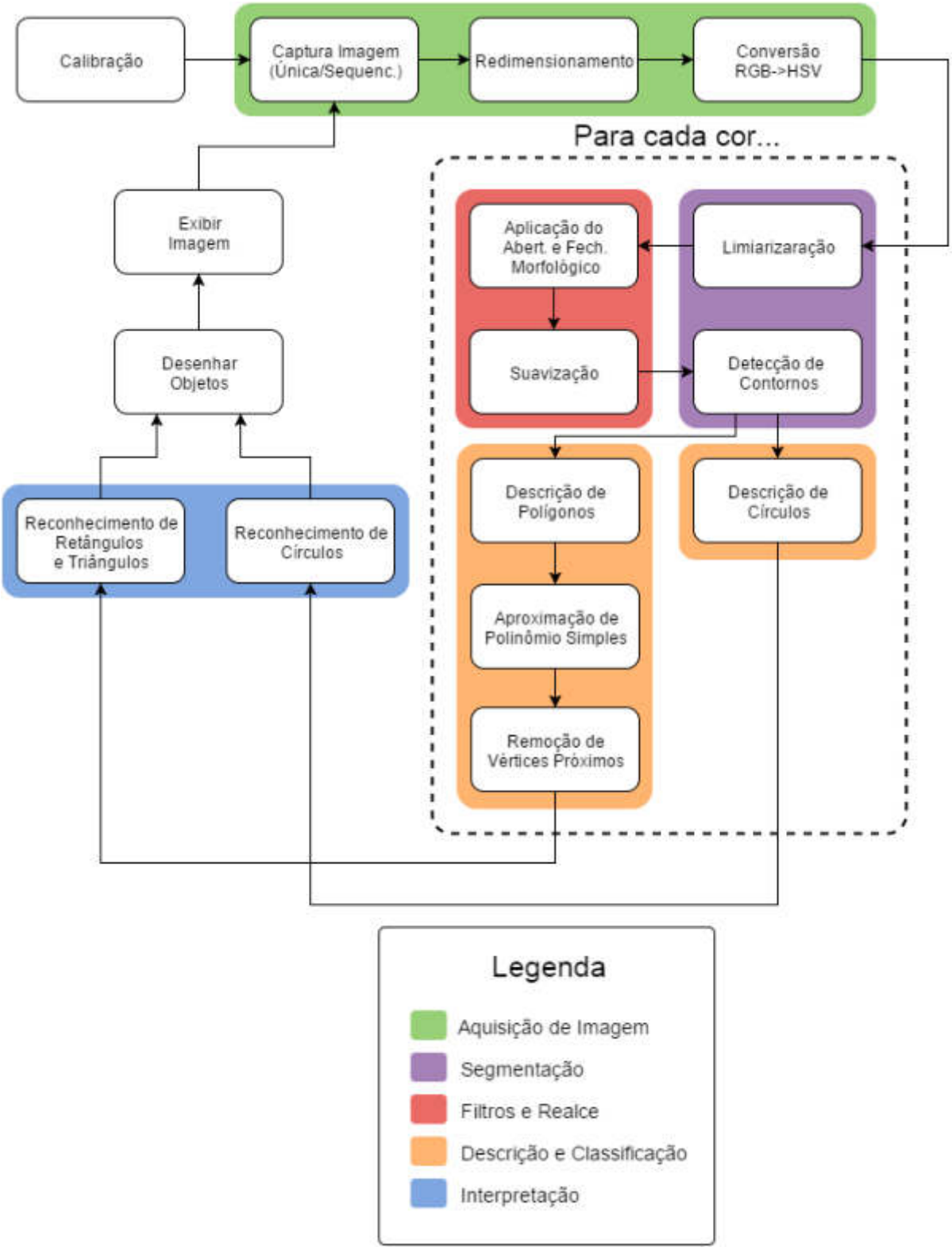
5 Especificação do processo

Tomando como base todo o referencial teórico e os trabalhos já realizados com PDI, foi desenvolvido um processo capaz de reconhecer objetos geométricos simples (triângulos, círculos e retângulos) de qualquer tamanho, de qualquer cor, em uma imagem ou em uma sequência contínua de imagens digitais, limitando-se às seguintes restrições:

- Toda a cena que compõe a imagem deve estar iluminada uniformemente;
- A câmera que captura as imagens deve estar fixada de forma que seu eixo central esteja perpendicular a um plano;
- As cores deste plano devem se diferenciar das cores dos objetos a serem reconhecidos;
- As cores dos objetos a serem reconhecidos devem ser previamente calibradas;
- Os objetos a serem reconhecidos devem estar totalmente contidos na imagem;
- Os objetos a serem reconhecidos não devem estar sobrepostos.

Um visão geral deste processo pode ser visualizado no fluxograma da Figura 23. Nas seções a seguir serão descritas todas as etapas deste fluxograma com o objetivo de esclarecer o funcionamento do processo proposto.

Figura 23 – Fluxograma do processo proposto.



Fonte: Elaborado pelo autor.

5.1 Calibração

Nesta etapa são definidas quais são as possíveis cores dos objetos a serem reconhecidos na imagem. Definir o que é uma cor em uma imagem a partir de procedimentos computacionais é uma tarefa complexa e custosa (FILHO; NETO, 1999), diferente do ser humano, que é capaz de identificar cores independentemente da iluminação ou ângulo de visão da cena. Por este motivo, optou-se por criar uma etapa de calibração onde são definidos, de forma manual, os valores utilizados pelo processo na identificação de cores em uma imagem.

Então, para cada cor deve-se definir valores únicos para os seguintes atributos:

- Nome da cor;
- Tripla de limiares inferiores de valores do padrão HSV;
- Tripla de limiares superiores de valores do padrão HSV.

A escolha do uso do padrão HSV deu-se pelo fato de que o modelo de representação das cores neste padrão é semelhante à percepção humana das cores (FILHO; NETO, 1999). O ser humano não identifica uma cor pela quantidade de vermelho, verde e azul que ela possui (padrão RGB), mas sim pelo tom, pureza e quantidade de iluminação incidente na cor (padrão HSV). Isto facilita a identificação visual mais precisa dos valores limiares.

Os valores limiares inferiores e superiores do HSV formam faixas que definem o que é uma determinada cor em uma imagem. A partir dessas faixas é possível realizar operações de limiarização que segmentam as regiões de uma determinada cor do resto da imagem, como pode ser visto na Figura 18. Desta forma, é necessário que a identificação destes valores seja feita empiricamente tendo como referência uma imagem que esteja em condições (iluminação ou distância da câmera) semelhantes às da imagem utilizada no reconhecimento de objetos ou, preferencialmente, a mesma imagem que será analisada. Além disso, essa faixa de valores deve ser a menor possível, para garantir uma melhor precisão na distinção entre as cores.

5.2 Aquisição de imagem

Após a calibração das cores, é necessário definir qual imagem será analisada a fim de reconhecer os objetos presentes nela.

5.2.1 Captura de imagem

É neste momento que se escolhe qual imagem será analisada. Por conveniência, espera-se uma imagem no padrão RGB, o tipo mais comum em imagens digitais (FILHO; NETO, 1999). Esta imagem pode ser individual ou pertencer a um conjunto de imagens sequenciais. Após a obtenção desta imagem é necessário que sejam aplicadas algumas transformações antes de avançar para as próximas etapas do processo.

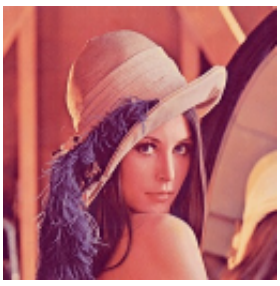
5.2.2 Redimensionamento

A maioria das operações feitas em imagens envolvem funções aplicadas em todos os seus *pixels* (GONZALES; WOODS, 2010). Em geral essas operações possuem complexidade de no mínimo $O(N \times M)$, onde N é o número de *pixels* verticais e M é o número de *pixels* horizontais da imagem. De acordo com o tamanho da imagem ou da complexidade das operações realizadas, o custo computacional pode exceder o desejado.

Uma forma simples de diminuir este custo é redimensionar a imagem por um fator R que esteja entre 0 e 1, ou seja, diminuir o tamanho da imagem. Uma imagem de tamanho $N \times M$ que for redimensionada por um fator R passa a ter as dimensões $(N \cdot R) \times (M \cdot R)$ e consequentemente, as operações aplicadas à imagem redimensionada passam a ter complexidade de no mínimo $O((N \cdot R) \times (M \cdot R))$, diminuindo em $(1 - R)\%$ o custo computacional.

A escolha de um valor para o fator R é uma tarefa delicada. Uma redução brusca do tamanho da imagem, por exemplo, por um fator $R = 0,15$, pode ser atrativa por diminuir em 85% o custo computacional, mas ao mesmo tempo pode-se perder detalhes de interesse na imagem resultante. Analisando a Figura 24 percebe-se que a imagem reduzida pelo fator $R = 0,15$ (Figura 24c) ficou pequena a ponto de não ser possível distinguir suas características fundamentais a olho nu, o mesmo se aplica ao computador. Enquanto isso, na imagem reduzida pelo fator $R = 0,5$ (Figura 24a) ainda é possível identificar essas características. Então, deve-se tentar encontrar um fator R pequeno o suficiente que reduza ao máximo o custo computacional, desde que na imagem reduzida ainda seja possível identificar plenamente suas características de interesse. Por fim, o fator R deve ser armazenado para que no futuro, as posições dos objetos encontrados possam ser recalculadas e relacionadas à imagem original e não à imagem redimensionada.

Figura 24 – Diferentes tipos de redimensionamento.



(a)

(b) Imagem original.

(c)

Fonte: Adaptado de (GONZALES; WOODS, 2010).

Outra escolha criteriosa a ser feita é o algoritmo de interpolação que será utilizado no redimensionamento da imagem. Como já explicitado pelas Figuras 9, 10 e 11, cada tipo de interpolação possui um determinado custo computacional e geram imagens com diferentes níveis de nitidez. Quanto mais nítida uma imagem estiver, melhores serão os resultados do reconhecimento de objetos na imagem.

5.2.3 Conversão do padrão RGB para HSV

Após a obtenção da imagem e do seu redimensionamento, é preciso convertê-la do padrão RGB para o padrão HSV. Esta conversão é necessária pois os parâmetros limiares que identificam as cores (definidas durante a etapa de calibração) foram definidos em relação aos canais *Hue*, *Saturation* e *Value* de cada *pixel* da imagem.

5.3 Segmentação e preprocessamento

Diferenciando-se um pouco do fluxo tradicional do PDI encontrado na literatura (GONZALES; WOODS, 2010), o processo especificado neste trabalho possui duas etapas de preprocessamento realizadas entre as de segmentação. As seções a seguir descrevem essas etapas indicando a qual categoria elas pertencem.

5.3.1 Limiarização (Segmentação)

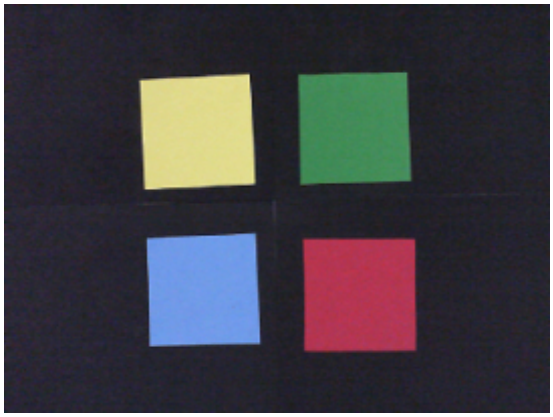
Muitos dos procedimentos utilizados no reconhecimento de objetos em imagens recomendam o uso de imagens em tons de cinza para fins de simplificação, já que a imagem em tons de cinza possui um único canal de cor (GONZALES; WOODS, 2010). Contudo, utilizar imagens em tons de cinza implica em perder a possibilidade de identificar as cores dos objetos contidos na imagem. O processo descrito neste trabalho aborda o problema de uma forma alternativa, garantindo uma simplificação superior ao do uso de imagens em tons de cinza e ainda permitindo a identificação de cores.

Ao invés de analisar uma única imagem, subdivide-se a imagem original em várias imagens binárias distintas. Para se obter essas imagens binárias são realizadas operações de limiarização utilizando os limiares das cores definidas na etapa de calibração. Ou seja, cada limiarização resultará em uma imagem binária distinta contendo somente a marcação das regiões da imagem original que contém determinada cor. A Figura 25 demonstra a limiarização de uma imagem por três cores distintas.

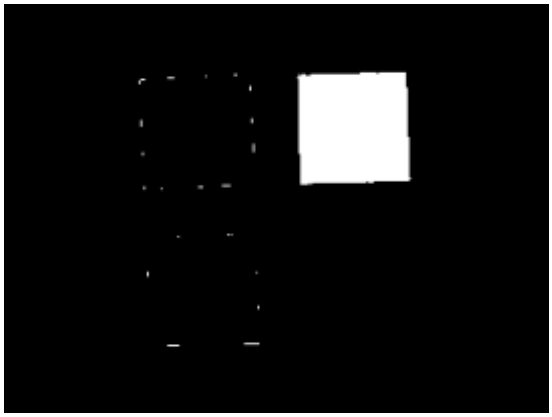
Consequentemente, o número de imagens binárias resultantes será igual ao número de cores definidas na etapa de calibração. Com essa abordagem já é possível determinar quais as cores dos possíveis objetos presentes em uma imagem. Além disso, eliminam-se elementos que não sejam de interesse, estes incluem: o fundo da imagem, outros objetos, ou até mesmo sombras. A eliminação destes elementos diminui as chances da presença de falsos positivos nos resultados do reconhecimento de objetos. Como pode ser observado na Figura 26, foi aplicado o algoritmo de detecção e descrição de círculos de Ramer (1972) em

imagens binárias (Figura 26a) e em uma imagem em tons de cinza (Figura 26b). Os círculos encontrados em cada situação foram desenhados na imagem original.

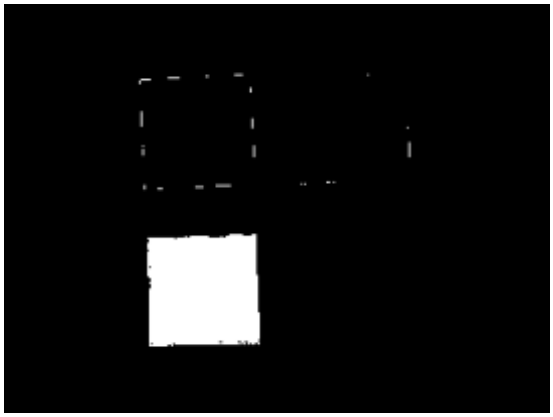
Figura 25 – Limiarização por três cores distintas.



(a) Imagem original.



(b) Limiarização pela cor verde.

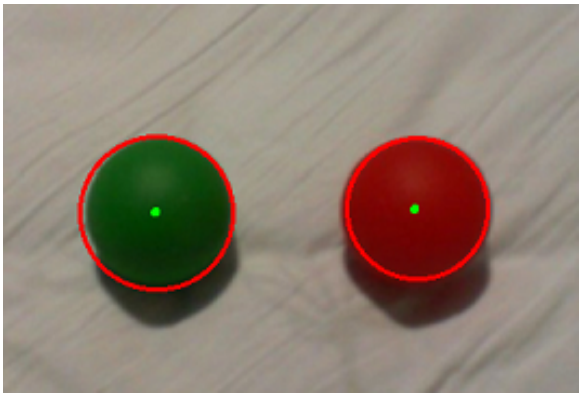


(c) Limiarização pela cor azul.

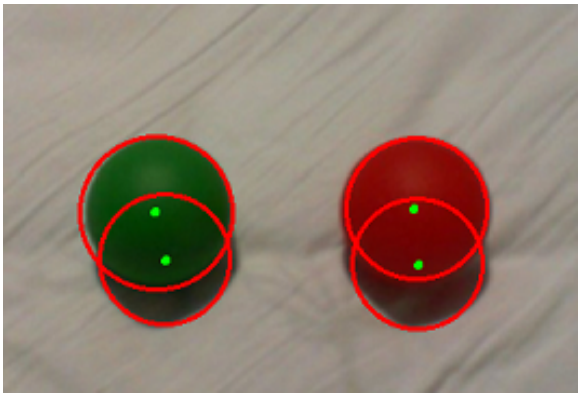
(d) Limiarização pela cor vermelha.

Fonte: Elaborado pelo autor.

Figura 26 – Aplicação do algoritmo de Ramer (1972) em diferentes tipos de imagem.



(a) Dois círculos encontrados.



(b) Quatro círculos encontrados.

Fonte: Elaborado pelo autor.

A partir desse momento, todas as transformações ou análises são realizas nas imagens binárias de forma independente. Ou seja, além de permitir a determinação de cores e eliminar elementos que não sejam de interesse da imagem, essa abordagem permite e incentiva o uso de programação paralela para realizar as etapas seguintes. Essa técnica pode melhorar o desempenho do processo como um todo, tornado-o mais apto a ser utilizado em situações onde o tempo de resposta é um fator determinante, por exemplo, em sistemas de tempo real.

5.3.2 Aplicação da abertura e do fechamento morfológico (Preprocessamento)

Em cada imagem binária gerada na etapa anterior são aplicadas operações de abertura e fechamento morfológico. A aplicação destas operações garante uma maior consistência das regiões segmentadas da imagem, eliminando pequenas falhas ou ruídos causados na etapa de limiarização devido a alguma imprecisão na definição dos limiares das cores.

A escolha do elemento estruturante utilizado na aplicação das operações morfológicas deve ser feita de forma que não cause grandes deformações nos objetos, principalmente em suas bordas. Este fenômeno pode ser observado na Figura 27.

Figura 27 – Abertura e fechamento morfológico por diferentes elementos estruturantes.

(a) Imagem original. (b) Elem. estru.: Elipse 3×3. (c) Elem. estrut.: Elipse 20×20.

Fonte: Elaborado pelo autor.

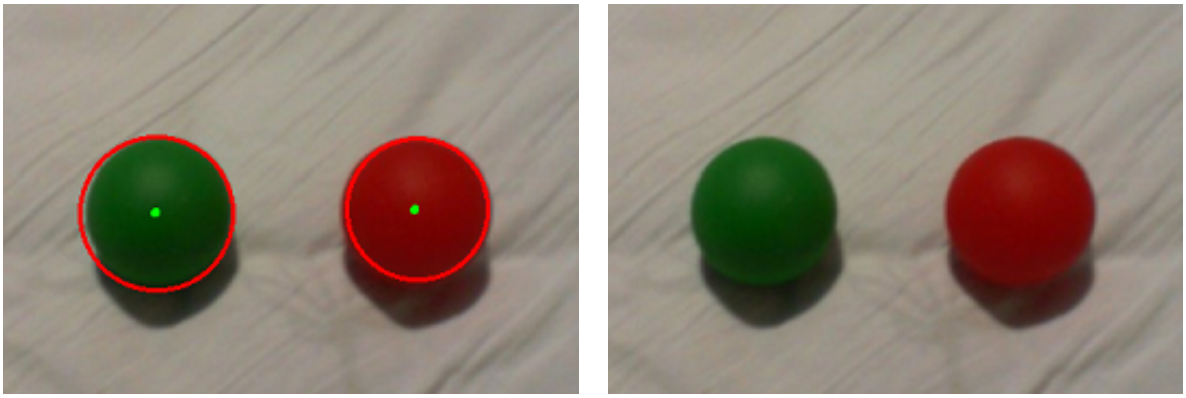
5.3.3 Suavização (Preprocessamento)

Para finalizar a etapa de preprocessamento, deve-se aplicar um filtro de suavização nas imagens resultantes da etapa anterior. Este filtro também ajuda na eliminação de ruídos, mas a sua principal função é suavizar as regiões de formato arredondado, fazendo com que possíveis círculos sejam reconhecidos mais facilmente.

Apesar de possuir um custo computacional mais alto, recomenda-se fortemente o uso do filtro gaussiano. Esse filtro abrange um número maior de tipos de ruídos e consegue manter a nitidez na imagem. Essa nitidez deve ser priorizada durante a escolha dos parâmetros do filtro gaussiano. Se a suavização for muito intensa, pode-se prejudicar o reconhecimento dos objetos no geral, mas se for muito fraca, pode prejudicar o reconhecimento de círculos, especificadamente. Este fenômeno pode ser visualizado na Figura 28. Neste caso, foi aplicado

o algoritmo de Ramer (1972) nas imagens binárias moderadamente suavizadas (Figura 28a) e nas imagens binárias intensamente suavizadas (Figura 28b). Os círculos encontrados foram desenhados na imagem original.

Figura 28 – Impacto da suavização no reconhecimento de círculos.



(a) Dois círculos encontrados. (b) Nenhum círculo encontrado.

Fonte: Elaborado pelo autor.

5.3.4 Detecção de contornos (Segmentação)

Neste momento, espera-se que as imagens estejam nas melhores condições possíveis (sem ruídos, sem falhas e com bordas bem definidas) para passarem pela última etapa de segmentação, a detecção de contornos. Esta etapa consiste na aplicação de um método de detecção de bordas nas imagens resultantes da etapa anterior.

Por ser mais completo e com maior aceitação na literatura, recomenda-se o uso do algoritmo proposto por Canny (1986) para esta tarefa. A definição de seus limiares é uma tarefa simples, já que neste ponto existem menos elementos nas imagens a serem analisadas. Então, seguindo as recomendações do autor de manter uma proporção 1:2 ou 1:3, espera-se obter resultados satisfatórios.

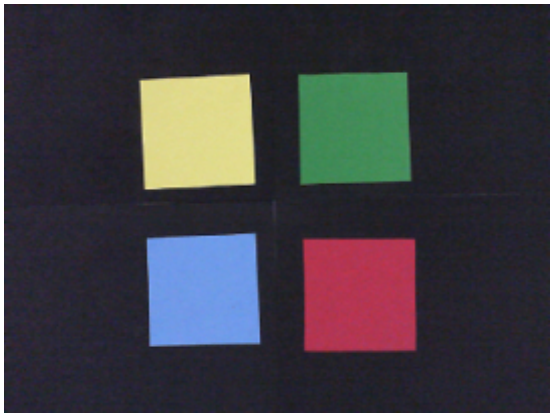
Além da detecção das bordas nas imagens, é necessário que essas bordas encontradas sejam as únicas regiões presentes nas imagens resultantes desta etapa, eliminando o preenchimento das regiões nas imagens binárias (Figura 29). Isto melhora a precisão dos algoritmos de descrição que são utilizados na etapa seguinte.

5.4 Representação e descrição

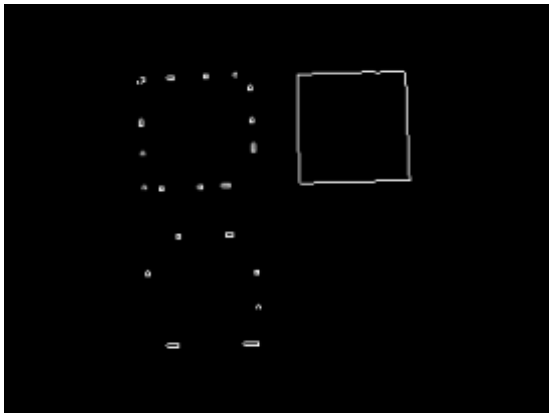
Foram realizados tratamentos para realçar as regiões onde possivelmente se encontram os objetos de interesse na imagem, resultando em imagens binárias compostas somente por contornos. Então, a tarefa desta etapa consiste em definir uma forma de representação para estes contornos e descrevê-los com esta representação.

Os objetos de interesse do escopo deste trabalho são círculos, triângulos e retângulos, ou seja, polígonos. Na literatura, é possível encontrar algoritmos que descrevam qualquer tipo

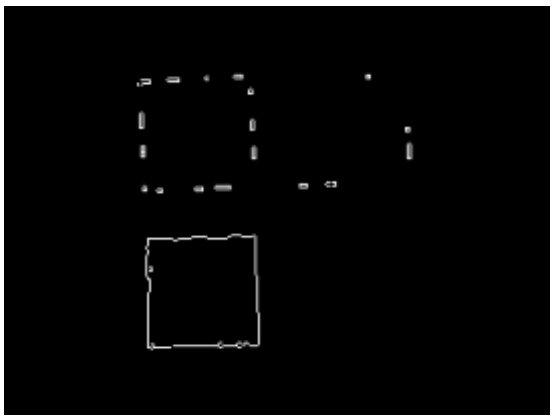
Figura 29 – Detecção de contornos em imagens limiarizadas por cor.



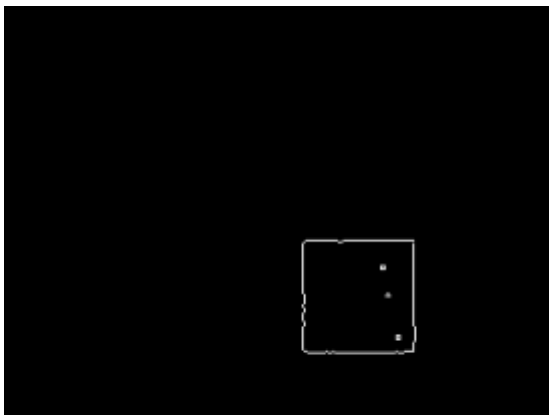
(a) Imagem original.



(b) Bordas das regiões de cor verde.



(c) Bordas das regiões de cor azul.



(d) Bordas das regiões de cor vermelha.

Fonte: Elaborado pelo autor.

de polígono presente em uma imagem a partir de contornos, mas para aumentar a precisão do reconhecimento, optou-se por também aplicar um algoritmo especialista na descrição de círculos.

Para cada imagem binária analisada é criada uma lista de objetos do tipo Círculo ou Polígono que são associados à mesma cor utilizada no processo de segmentação que gerou a respectiva imagem binária. Além disso, todos os valores gerados nesta etapa (pontos, distâncias, tamanhos) devem ser ajustados de acordo com o fator R utilizado no redimensionamento da imagem, fazendo com que os valores não estejam associados à imagem redimensionada, mas sim à imagem original.

5.4.1 Descrição de polígonos

A descrição de polígonos deve ser feita por um algoritmo seguidor de contornos que irá identificar e descrever todos os contornos presentes nas imagens binárias geradas na etapa anterior. Orienta-se a aplicação dos algoritmos propostos por Suzuki et al. (1985) por serem específicos para imagens binárias. Esta especialidade torna-os capazes de realizar uma descrição mais precisa dos contornos presentes na imagem. Dentre as possíveis formas de re-

presentação dos polígonos encontrados pelo algoritmo seguidor de contorno, a lista de pontos é ideal para manter a simplicidade da representação, manipulação e análise dos resultados encontrados.

O volume de pontos resultante da aplicação do algoritmo de Suzuki et al. (1985) é um complicador para a etapa de identificação decidir qual é o tipo de polígono encontrado. Uma forma de diminuir esta complexidade é a aplicar algoritmo de suavização de curvas, como o proposto por Ramer (1972), sobre o polígono encontrado. Além da aplicação deste algoritmo, recomenda-se verificar se cada distância entre os pontos encontrados ultrapassa um valor de distância mínima previamente definido, caso contrário, deve-se eliminar os pontos que compõe esta distância inferior à mínima e definir um novo ponto posicionado entre os dois eliminados. Desta forma, diminui-se ao máximo o número de pontos de um polígono e por consequência, a complexidade das análises realizadas sobre ele. Um exemplo da descrição simplificada de um polígono pode ser vista na Figura 20.

Cada lista de pontos gerada a partir da análise de uma determinada imagem binária deve relacionada a um objeto do tipo Polígono que deve ser incluído na lista de objetos associada à imagem binária referente.

5.4.2 Descrição de círculos

Diferentemente de triângulos e retângulos, círculos são formas geométricas mais complexas de serem descritas pelo seu contorno. Computacionalmente descreve-se o perímetro de um círculo por um conjunto finito de retas pequenas o suficiente a ponto de gerar a impressão de que se tem a aparência de uma curva. Como pode ser visto na Figura 21-b, a aplicação de um algoritmo de descrição de polígonos em contornos de círculos gera uma quantidade muito alta de pontos, o que torna inviável qualquer análise ou manipulação feita sobre o polígono encontrado. Este número de pontos também pode ser reduzido utilizando o algoritmo proposto por Ramer (1972), mas o resultado pode gerar um conjunto de pontos que não descreve a circunferência com muita fidelidade, este fenômeno pode ser observado na Figura 21-c.

Por ter esta particularidade, existem maneiras mais eficientes de representar um círculo do que pelos pontos que formam o seu contorno. A mais simples destas envolve a descrição do ponto central do círculo e o seu raio. O trabalho feito por Yuen et al. (1990) propõe um algoritmo seguidor de contornos em imagens em tons de cinza, baseado na transformada de Hough e especializado na descrição de círculos. A restrição do uso de imagens em tons de cinza não é um problema, já que as imagens geradas na etapa anterior são binárias e uma imagem binária é uma simplificação de uma imagem em tons de cinza. Pelo fato da descrição dos círculos ser feita pela sua representação mais simples, esta solução, simplifica qualquer tipo de análise posterior feita sobre os círculos e descreve os círculos de maneira mais precisa, já que a partir de um ponto central e um raio, é possível deduzir o perímetro de um círculo de forma mais exata. Este resultado pode ser visualizado na Figura 22.

Os resultados gerados pela aplicação do algoritmo de Yuen et al. (1990) em cada uma das imagens binárias da etapa anterior devem ser relacionados a objetos do tipo Círculo e estes devem ser incluídos na lista de objetos associada à imagem binária referente.

5.5 Identificação

A última etapa do processo de reconhecimento de objetos é encarregada de analisar todos os objetos presentes nas listas associadas às imagens binárias para defini-los como objetos reais presentes na imagem original e classificá-los mais precisamente. Cada objeto é avaliado e classificado de acordo com três conjuntos de critérios, caso ele não se encaixe em nenhum desses conjuntos, este não é considerado um objeto real e é descartado da lista.

Ao final desta etapa é gerada uma lista contendo todos os objetos restantes das listas das imagens binárias. Pelo fato das imagens serem avaliadas paralelamente por um algoritmo de descrição de polígonos e por um de descrição de círculos, é possível que sejam detectados dois objetos equivalentes a um único objeto real, especificadamente círculos que também são reconhecido como polígonos. Para evitar este fenômeno é necessário estabelecer uma distância mínima entre dois objetos na imagem, ou seja, uma distância mínima entre os seus centroides e eliminar da lista os que não atenderem a esse requisito. A remoção de objetos da lista deve ser feita dando preferencia para a permanência de objetos do tipo Círculos pois a probabilidade deste fenômeno acontecer com eles é mais alta.

Por fim, o conteúdo desta lista equivale aos objetos reconhecidos na imagem original e cada objeto contém tributos suficientes (cor, formato, centroide, vértices que o constituem, caso polígono, ou raio, caso círculo) para sua descrição e classificação. Esse tipo de abordagem evidencia a simplicidade do uso de métodos exatos no reconhecimento objetos.

Os conjuntos de critérios para avaliação e classificação de objetos são detalhados a seguir:

5.5.1 Reconhecimento de triângulos

Para ser considerado um triângulo, o objeto deve:

- Ser do tipo Polígono;
- Ser um polígono convexo;
- Possuir uma área igual ou maior do que a área mínima previamente definida;
- Possuir três vértices.

Se o objeto atender todos esses critérios, ele é considerado como um objeto real e este deve receber um novo atributo com o valor correspondente ao seu centroide.

5.5.2 Reconhecimento de retângulos

Para ser considerado um retângulo, o objeto deve:

- Ser do tipo Polígono;
- Ser um polígono convexo;

- Possuir uma área maior ou igual do que a área mínima previamente definida;
- Possuir ângulos internos próximos a 90^o;
- Possuir quatro vértices.

Se o objeto atender todos esses critérios, ele é considerado como um objeto real e este deve receber um novo atributo com o valor correspondente ao seu centroide.

5.5.3 Reconhecimento de círculos

Para ser considerado um círculo, o objeto deve:

- Ser do tipo Círculo;
- Possuir um raio maior ou igual do que o raio mínimo previamente definido;
- Possuir um raio menor ou igual do que o raio máximo previamente definido.

Se o objeto atender todos esses critérios, ele é considerado como um objeto real e este deve receber um novo atributo com o valor correspondente ao seu centroide.

5.6 Rastreamento de objetos

Sendo o rastreamento de objetos equivalente ao reconhecimento de objetos de forma contínua, para realizá-lo basta que todas as etapas do processo especificado (exceto a de calibração) sejam realizadas continuamente utilizando uma sequência de imagens contínuas como fonte de dados. Estas podem ser em formato de vídeo ou providas por alguma fonte de imagens em tempo real (câmera, *webcam*, etc).

Especificadamente para o segundo caso, existe a necessidade de que o processo especificado apresente um certo nível de desempenho que atenda a uma determinada frequência de captura de imagens. Acredita-se que este seja o caso e isto será demonstrado nos resultados dos testes realizados em uma *software* desenvolvido que implemente o processo especificado neste trabalho.

6 Avaliação do processo

6.1 Implementação do processo

Objetivando validar o funcionamento e a eficiência do processo de reconhecimento e rastreamento de objetos proposto por este trabalho, desenvolveu-se um *software* seguindo todos os passos e recomendações especificados. Este *software* é capaz de analisar imagens únicas ou imagens sequenciais em formato de vídeo ou capturadas de uma *webcam* em tempo real. As seguintes tecnologias foram utilizadas em seu desenvolvimento:

- OpenCV 2.4.9;
- C++11;
- Microsoft ® C/C++ Otimizing Compiler Version 18.00.30501 for x86.

A escolha do OpenCV deu-se pelo fato dele ser umas das biblioteca de processamento de imagens mais bem estabelecidas e suportadas disponíveis atualmente na computação. Todos os algoritmos necessários para realizar as etapas do processo especificado estão implementados e documentados na biblioteca. A linguagem C++ foi escolhida por ser a linguagem base do OpenCV e o compilador da Microsoft foi utilizado pelo fato do desenvolvimento ter sido feito em uma máquina com sistema operacional Windows 8.1, utilizando a IDE Microsoft Visual Studio Ultimate 2013.

Alguns parâmetros do *software* desenvolvido podem citados:

- Coeficiente de redimensionamento $R = 0,5$;
- Elemento estruturante das operações morfológicas: Elipse de tamanho 6×6 *pixels*;
- Tamanho da vizinhança do Filtro Gaussiano: 3×3 *pixels*;
- Desvio padrão do Filtro Gaussiano: 1 pixel;
- Limiar inferior para o algoritmo de Canny (1986): 30 *pixels*;
- Limiar superior para o algoritmo de Canny (1986): 90 *pixels*;
- Área mínima dos polígonos: 100 *pixels*;
- Raio mínimo dos círculos: 10 *pixels*;
- Raio máximo dos círculos: 50 *pixels*;
- Distância mínima entre vértices: 10 *pixels*;

6.2 Testes em imagem

Para validar o *software* desenvolvido, foram realizados testes utilizando imagens de tamanho 640×480 *pixels* em padrão RGB providas por uma *webcam*. Estas imagens são compostas por objetos de interesse dispostos em um plano de cor preta com iluminação uniforme, a aproximadamente um metro da câmera. A relação dos objetos utilizados nos testes pode ser vista na Tabela 2.

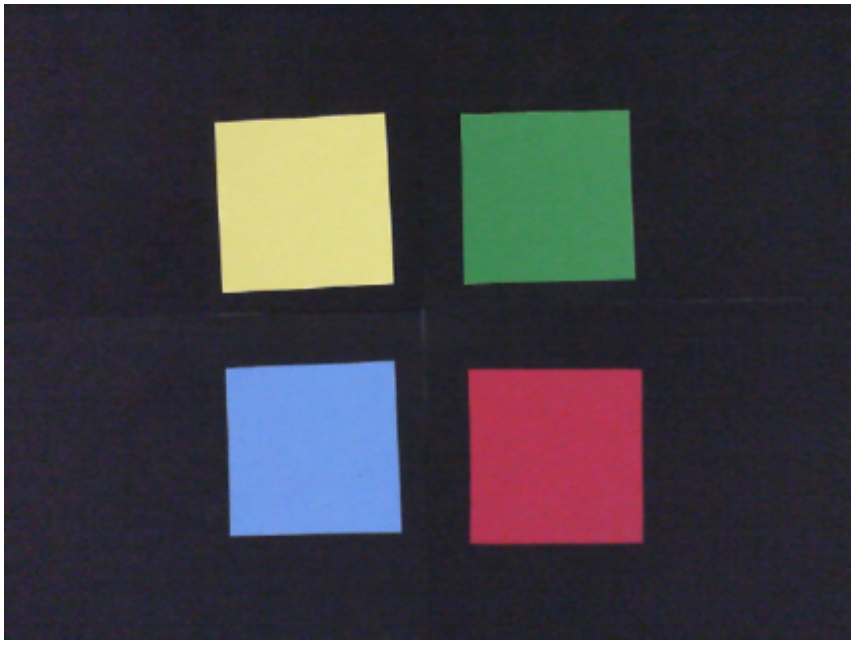
Tabela 2 – Objetos utilizados nos testes.

			Cor			
Categoria	**Formato**	**Tamanho**	**Amarelo**	**Azul**	**Verde**	**Vermelho**
Círculo	Círculo	Pequeno	2	1	1	1
		Grande	1	2	1	1
Retângulo	Quadrado	Pequeno	2	1	1	1
		Grande	1	1	2	1
	Retângulo	Pequeno	1	1	1	2
		Grande	1	1	1	2
Triângulo	Triângulo	Pequeno	1	1	2	1
		Grande	1	2	1	1

Fonte: Elaborado pelo autor.

A imagem utilizada durante a etapa de calibração pode ser visualizada na Figura 30. Os limiares identificados durante a calibração podem ser observados na Tabela 3. Todos os testes realizados nesta seção utilizaram estes mesmos valores de calibração.

Figura 30 – Imagem utilizada na etapa de calibração.



Fonte: Elaborado pelo autor.

Tabela 3 – Limiares identificados na etapa de calibração.

	Limiar					
	Inferior			**Superior**		
Cor	**H**	**S**	**V**	**H**	**S**	**V**
Amarelo	109	202	194	208	255	255
Azul	75	85	89	247	176	172
Verde	63	89	32	132	170	136
Vermelho	61	0	103	123	73	255

Para cada imagem testada (**IMG**), registra-se:

- **Objetos de Interesse (OI)**: Número de objetos reais a serem reconhecidos na imagem.
- **Objetos Reconhecidos (OR)**: Número dos objetos reconhecidos ao final da análise da imagem. Para um objeto ser considerado reconhecido, este deve:
  - Ser identificado pela mesma cor que o objeto real;
  - Ser identificado pelo mesmo formato que o objeto real;
  - Ter seu contorno definido aproximado ao contorno do objeto real.
- **Falso Positivo (FP)**: Número de objetos reconhecidos que não possuem um correspondente real. Este categoria também inclui:
  - Objetos identificados pelo formato errado;
  - Objetos identificados pela cor errada;
  - Objetos com contorno muito discrepante em relação ao objeto real;
- **Falso Negativo (FN)**: Número de objetos reais que não foram reconhecidos.
- **Tempo de resposta médio (TRM)**: O tempo de resposta consiste no tempo entre a captura de imagem até o término da criação da lista de objetos reconhecidos na imagem. Este valor é utilizado para estimar o desempenho do processo implementado. Para que esta avaliação de desempenho seja justa, cada imagem foi processada dez vezes seguidas afim de se obter um tempo de resposta médio. Além disso, todas as análises incluem tentativas de identificação de qualquer tipo de objeto de qualquer uma das cores pré-definidas.

No total foram analisadas 24 imagens divididas igualmente entre 6 categorias de testes (**CAT**). Cada categoria é composta por imagens com características específicas. Esta divisão visa avaliar separadamente determinadas capacidades da implementação realizada. As categorias e as imagens que as compõe são descritas a seguir.

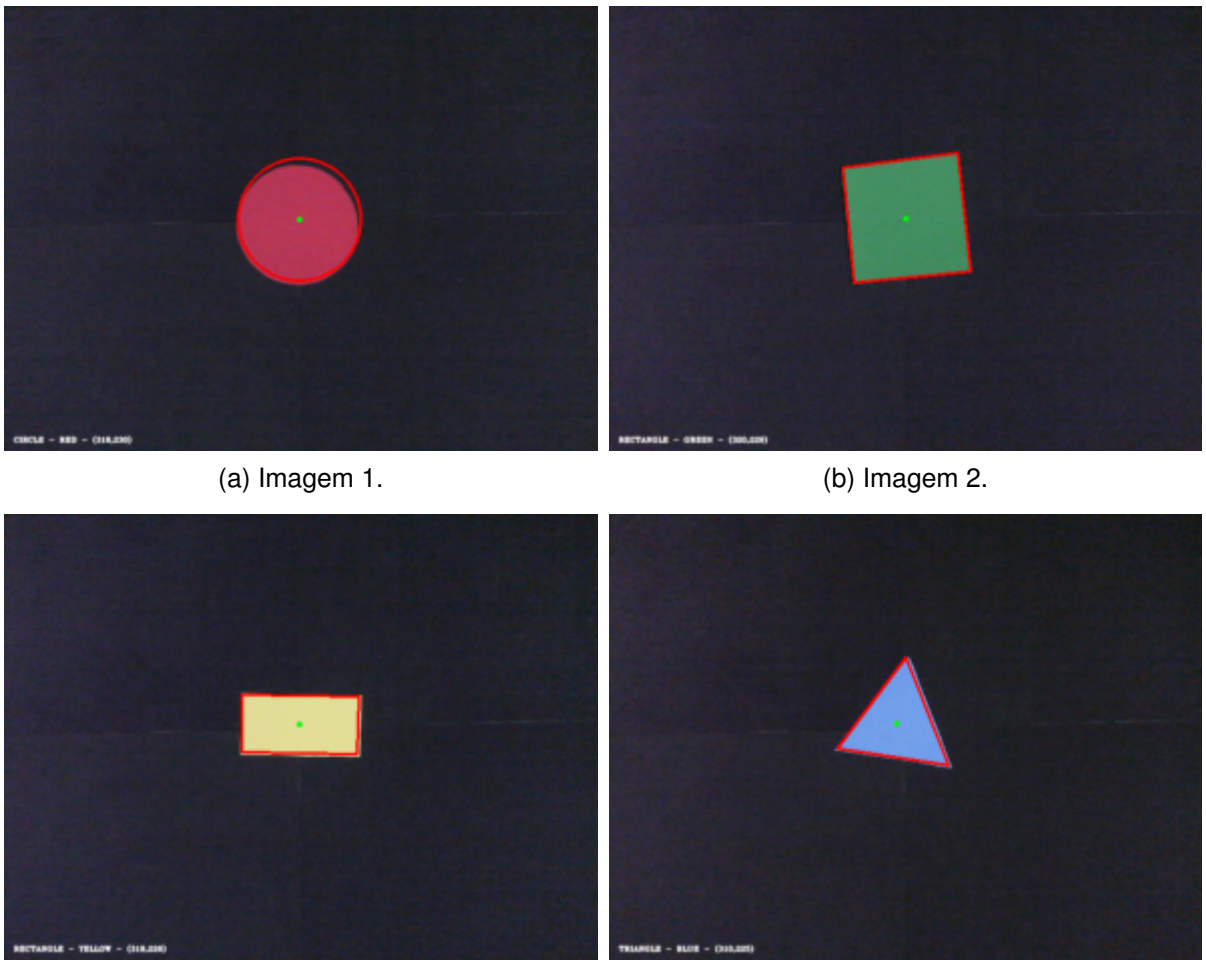
6.2.1 Categoria 1

Os testes desta categoria avaliam a capacidade da implementação de reconhecer um único objeto em uma imagem. Para tanto, as imagens utilizadas possuem as seguintes características:

- Possuem um único objeto.

As imagens resultantes dos testes podem ser vistas na Figura 31 e os dados registrados podem ser vistos na Tabela 4.

Figura 31 – Imagens da Categoria 1.



(a) Imagem 1. (b) Imagem 2.

(c) Imagem 3. (d) Imagem 4.

Fonte: Elaborado pelo autor.

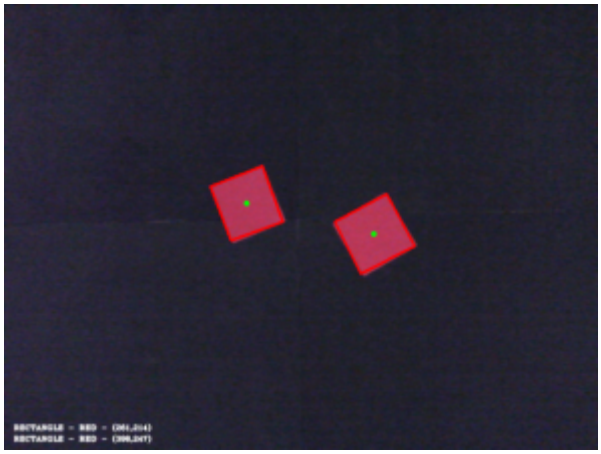
6.2.2 Categoria 2

Os testes desta categoria avaliam a capacidade da implementação de reconhecer múltiplos objetos iguais em uma única imagem. Para tanto, as imagens utilizadas possuem as seguintes características:

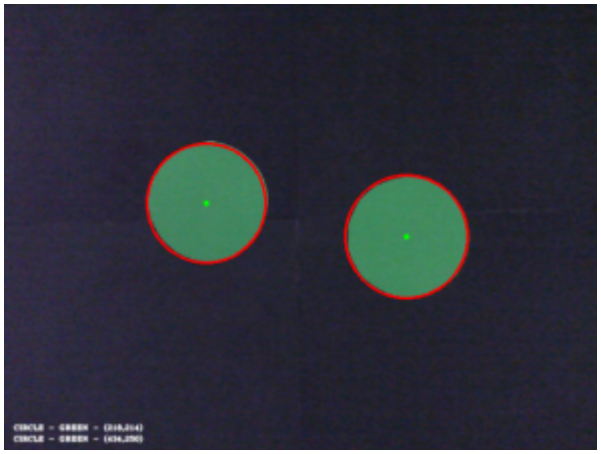
- Possuem dois objetos;
- Os objetos são da mesma cor;
- Os objetos são do mesmo formato;
- Os objetos são do mesmo tamanho.

As imagens resultantes dos testes podem ser vistas na Figura 32 e os dados registrados podem ser vistos na Tabela 4.

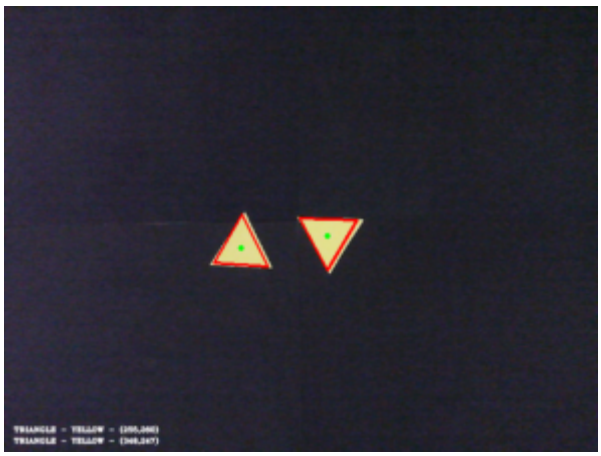
Figura 32 – Imagens da Categoria 2.



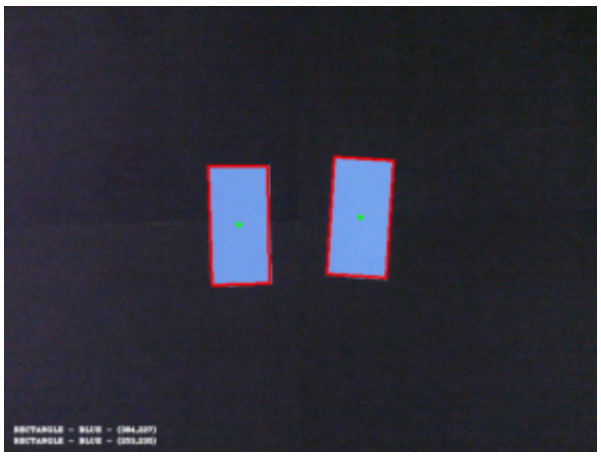
(a) Imagem 5.



(b) Imagem 6.



(c) Imagem 7.



(d) Imagem 8.

Fonte: Elaborado pelo autor.

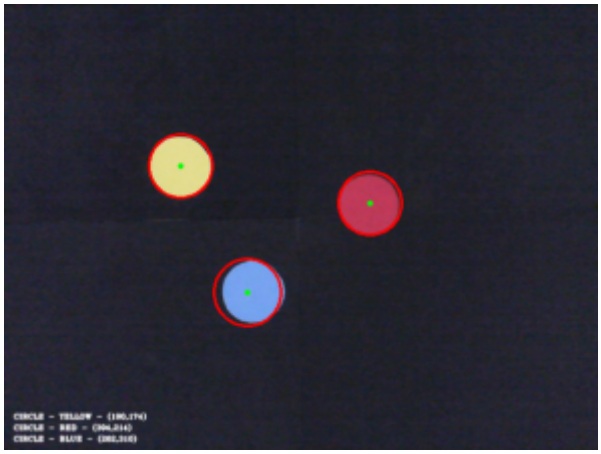
6.2.3 Categoria 3

Os testes desta categoria avaliam a capacidade da implementação de reconhecer múltiplos objetos do mesmo formato e tamanho, porém, de diferentes cores, simultaneamente, em uma única imagem. Para tanto, as imagens utilizadas possuem as seguintes características:

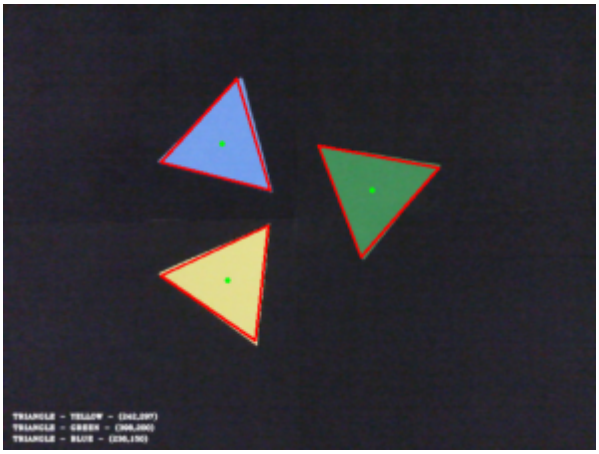
- Possuem três objetos;
- Os objetos são de cores diferentes;
- Os objetos são do mesmo formato;
- Os objetos são do mesmo tamanho.

As imagens resultantes dos testes podem ser vistas na Figura 33 e os dados registrados podem ser vistos na Tabela 4.

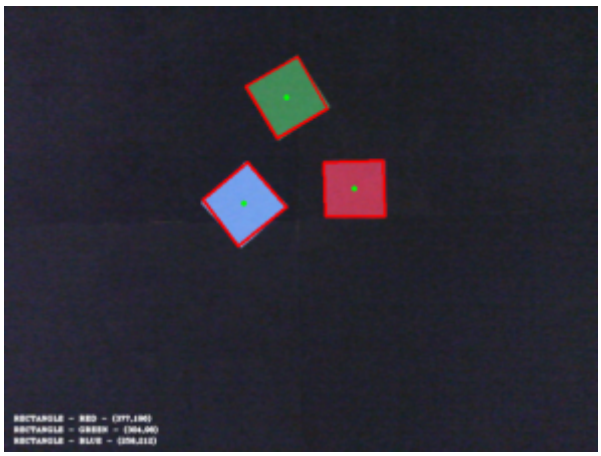
Figura 33 – Imagens da Categoria 3.



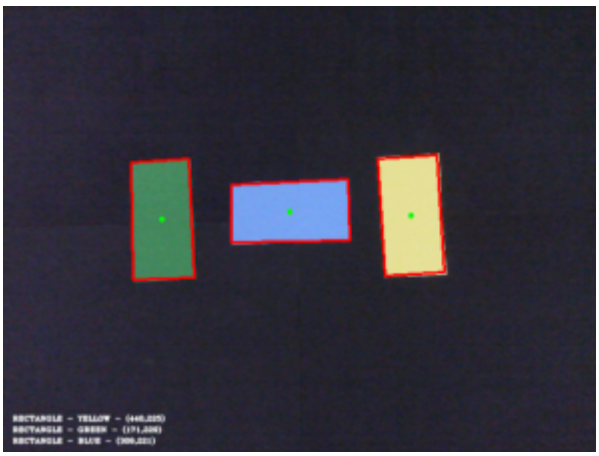
(a) Imagem 9.



(b) Imagem 10.



(c) Imagem 11.



(d) Imagem 12.

Fonte: Elaborado pelo autor.

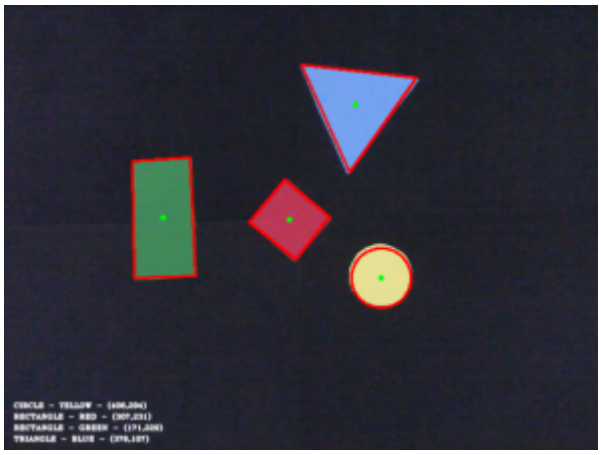
6.2.4 Categoria 4

Os testes desta categoria avaliam a capacidade da implementação de reconhecer múltiplos objetos de diferentes cores e formatos, simultaneamente, em uma única imagem. Para tanto, as imagens utilizadas possuem as seguintes características:

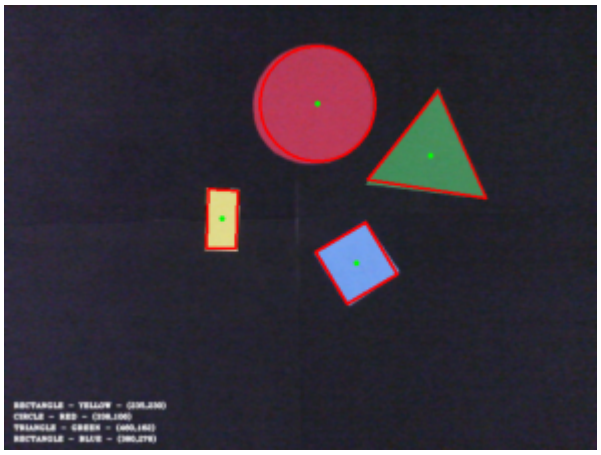
- Possuem quatro objetos;
- Os objetos são de cores diferentes;
- Os objetos são de formatos diferentes.

As imagens resultantes dos testes podem ser vistas na Figura 34 e os dados registrados podem ser vistos na Tabela 4.

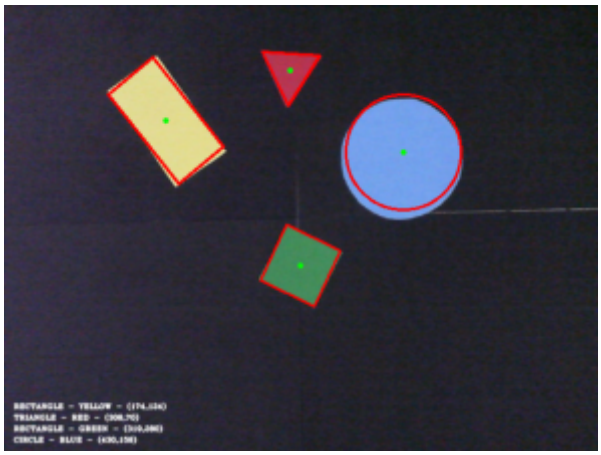
Figura 34 – Imagens da Categoria 4.



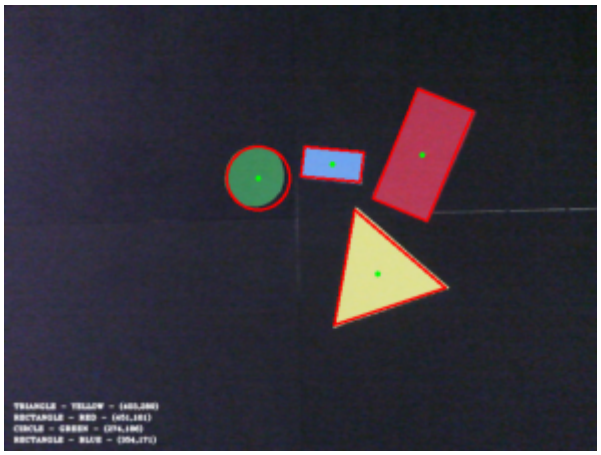
(a) Imagem 13.



(b) Imagem 14.



(c) Imagem 15.



(d) Imagem 16.

Fonte: Elaborado pelo autor.

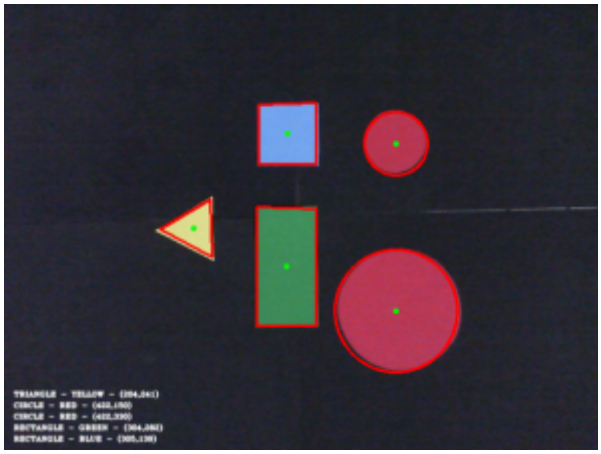
6.2.5 Categoria 5

Os testes desta categoria avaliam a capacidade da implementação de reconhecer múltiplos objetos de diferentes cores, formatos e tamanhos, simultaneamente, em uma única imagem. Para tanto, as imagens utilizadas possuem as seguintes características:

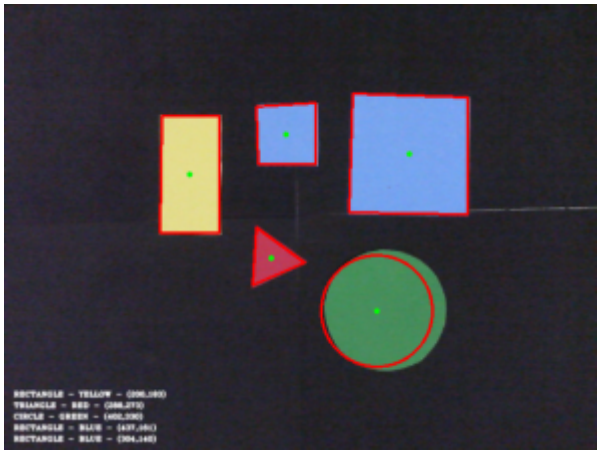
- Possuem cinco objetos;
- Os objetos são de cores diferentes;
- Os objetos são de formatos diferentes;
- Os objetos são de tamanhos diferentes.

As imagens resultantes dos testes podem ser vistas na Figura 35 e os dados registrados podem ser vistos na Tabela 4.

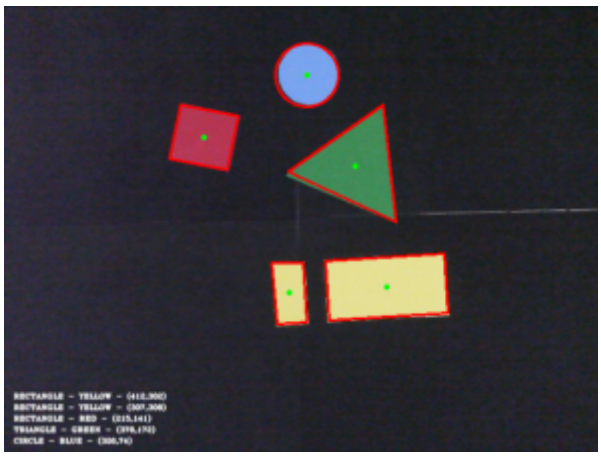
Figura 35 – Imagens da Categoria 5.



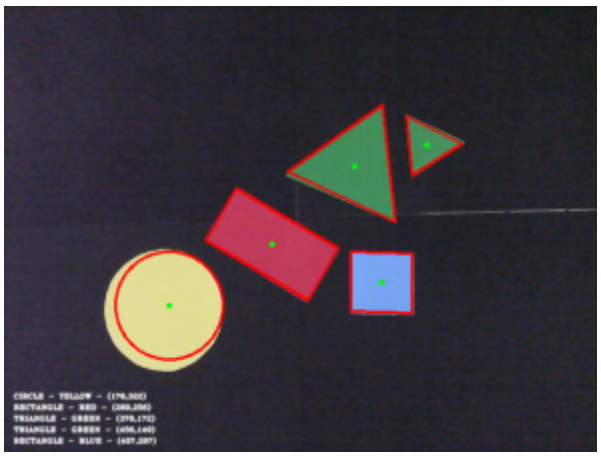
(a) Imagem 17.



(b) Imagem 18.



(c) Imagem 19.



(d) Imagem 20.

Fonte: Elaborado pelo autor.

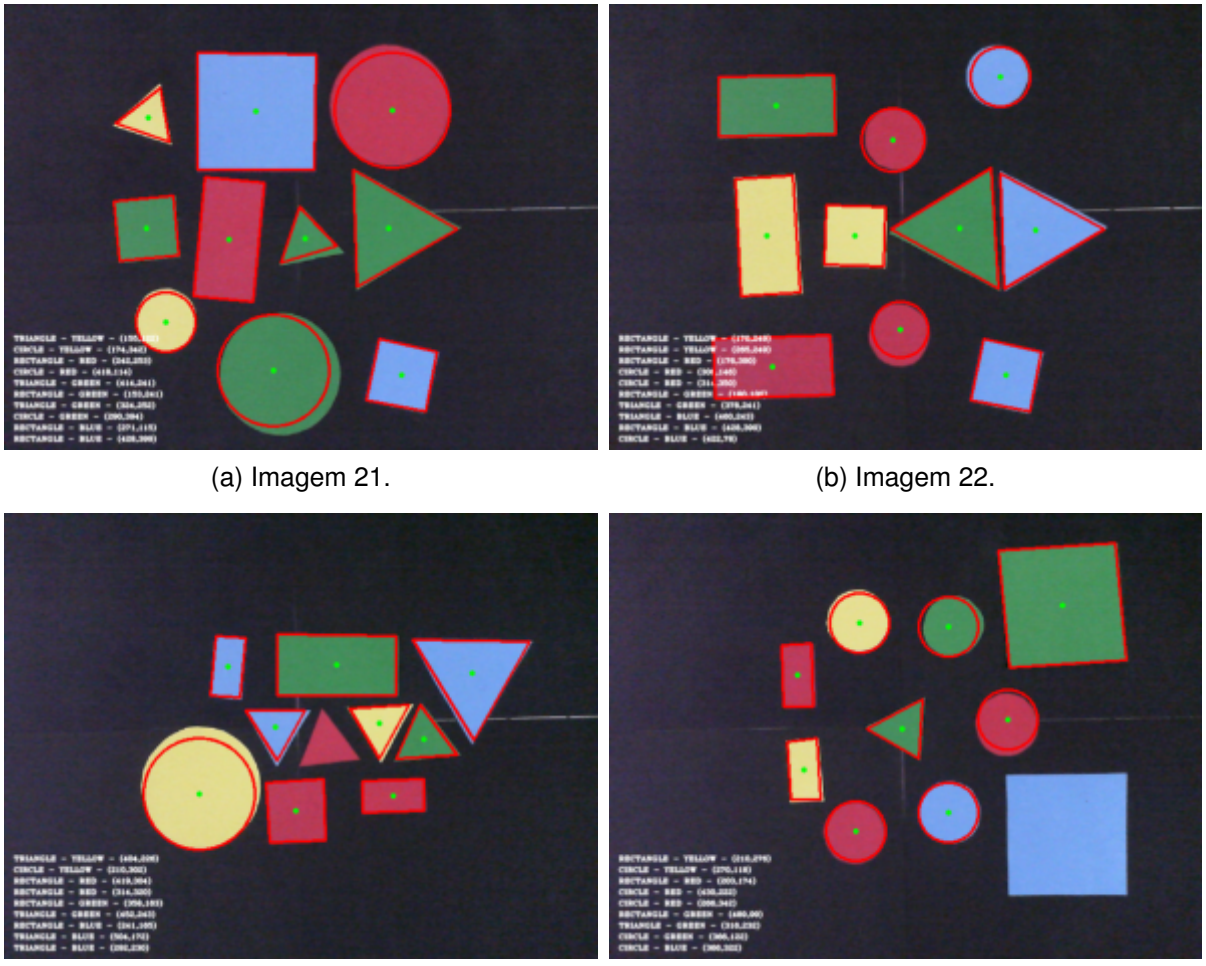
6.2.6 Categoria 6

Os testes desta categoria avaliam a capacidade da implementação de reconhecer um número maior de objetos de diferentes cores,formatos e tamanhos, simultaneamente, em uma única imagem. Para tanto, as imagens utilizadas possuem as seguintes características:

- Possuem dez objetos;
- Os objetos são de cores diferentes;
- Os objetos são de formatos diferentes;
- Os objetos são de tamanhos diferentes.

As imagens resultantes dos testes podem ser vistas na Figura 36 e os dados registrados podem ser vistos na Tabela 4.

Figura 36 – Imagens da Categoria 6.



(a) Imagem 21. (b) Imagem 22.

(c) Imagem 23. (d) Imagem 24.

Fonte: Elaborado pelo autor.

6.2.7 Resultados

Tabela 4 – Resultados dos testes em imagens.

CAT	IMG	OI	OR	OR(%)	FP	FP(%)	FN	FN(%)	TRM(ms)	TRM(ms) por CAT
1	1	1	1	100,00%	0	0,00%	0	0,00%	10,44	12,75
	2	1	1	100,00%	0	0,00%	0	0,00%	12,33	
	3	1	1	100,00%	0	0,00%	0	0,00%	13,89	
	4	1	1	100,00%	0	0,00%	0	0,00%	14,33	
2	5	2	2	100,00%	0	0,00%	0	0,00%	13,89	14,17
	6	2	2	100,00%	0	0,00%	0	0,00%	13,78	
	7	2	2	100,00%	0	0,00%	0	0,00%	16,78	
	8	2	2	100,00%	0	0,00%	0	0,00%	12,22	
3	9	3	3	100,00%	0	0,00%	0	0,00%	13,87	15,27
	10	3	3	100,00%	0	0,00%	0	0,00%	15,76	
	11	3	3	100,00%	0	0,00%	0	0,00%	14,00	
	12	3	3	100,00%	0	0,00%	0	0,00%	17,45	
4	13	4	4	100,00%	0	0,00%	0	0,00%	15,65	14,83
	14	4	4	100,00%	0	0,00%	0	0,00%	12,33	
	15	4	4	100,00%	0	0,00%	0	0,00%	13,89	
	16	4	4	100,00%	0	0,00%	0	0,00%	17,44	
5	17	5	5	100,00%	0	0,00%	0	0,00%	13,86	13,49
	18	5	5	100,00%	0	0,00%	0	0,00%	13,88	
	19	5	5	100,00%	0	0,00%	0	0,00%	14,23	
	20	5	5	100,00%	0	0,00%	0	0,00%	12,00	
6	21	10	10	100,00%	0	0,00%	0	0,00%	15,78	14,69
	22	10	10	100,00%	0	0,00%	0	0,00%	14,66	
	23	10	9	90,00%	0	0,00%	1	10,00%	13,67	
	24	10	9	90,00%	0	0,00%	1	10,00%	14,66	
Média	-	4,17	4,08	99,17%	0	0,00%	0,08	0,83%	14,20	

Elaborado pelo autor.

Analisando os dados apresentados na Tabela 4 nota-se que o *software* obteve resultados satisfatórios nas tarefas propostas. Das 24 imagens testadas, somente duas não obtiveram 100% dos seus objetos reconhecidos. Em cada uma dessas duas imagens, um único objeto de interesse não foi reconhecido, implicando em uma média de 0,83% falsos negativos por imagem. Não foi registrada nenhuma ocorrência de falsos positivos, logo, a média geral de objetos reconhecidos por imagem foi de 99.17%.

Além de obter resultados satisfatórios no reconhecimento de objetos, a implementação apresentou uma boa performance, gastando em média 14,20 ms para processar cada imagem. Em geral, capturas de vídeo em tempo real são feitas a 30 *frames* por segundo, ou seja, o espaço de tempo disponível entre a captura de *frames* é de 33,33 ms. Gastar 14,20 ms, em média, para reconhecer objetos implica em ter 19,13 ms disponíveis para realizar qualquer tipo de operação lógica sobre os objetos reconhecidos sem prejudicar a frequência de captura das imagens. Estes resultados são indicativos de que o processo implementado pode ser aplicado em sistemas de tempo real.

Também é notável que os tempos médios de processamento por categoria de imagem possuem valores próximos. Este fenômeno pode ser explicado pelo fato de que o processo considera todas as cores previamente calibradas e todos os formatos geométricos durante a análise das imagens, independentemente das características dos objetos de interesse presentes.

6.3 Testes em imagens sequenciais

Para demonstrar que o *software* desenvolvido é capaz de reconhecer continuamente, ou seja, rastrear, objetos em tempo real, foram realizados testes em imagens sequenciais. Essas imagens sequenciais equivalem aos *frames* capturados em tempo real por uma *webcam*. As mesmas configurações (padrões de imagem, parâmetros e cores) dos testes em imagem foram utilizadas neste caso.

Neste contexto, um conjunto de *frames* capturados pela *webcam* em um determinado período de tempo será denominado de sequência de imagens. Então, para cada sequência de imagens (**SEQ**), registra-se:

- **Objetos de Interesse (OI)**: Número de objetos reais a serem reconhecidos na sequência de imagens, sendo que todos os *frames* contidos da sequência possuem sempre os mesmos objetos de interesse que estão se movimentando constantemente.
- ***Frames* Processados (FP)**: Número de *frames* da sequência de imagens.
- ***Frames* "Corretos"(FC)**: Número de *frames* da sequência de imagens em que o reconhecimento de objetos foi pleno. Ou seja, todos os objetos de interesse foram reconhecidos corretamente e não foi registrada a presença de falsos positivos ou falsos negativos.
- **Tempo de Processamento (TP):** O tempo (em segundos) de duração do processamento completo da sequência de imagens.
- ***Frames per Second* (FPS):** Média de *frames* processados por segundo em uma sequência de imagens.

No total foram analisadas 5 sequências de imagens, cada uma com características equivalentes a uma das 5 primeiras categorias (**CAT**) dos testes em imagem. Os resultados dos testes podem ser visualizados na Tabela 5.

6.3.1 Resultados

Observando os resultados apresentados na Tabela 5 nota-se que o *software* realiza o rastreamento de objetos de forma satisfatória. Das 5 sequências de imagens testadas, em nenhuma a taxa de *frames* onde os objetos foram reconhecidos plenamente ficou abaixo de 95%. Além da alta precisão de rastreamento, o *software* apresentou performance suficiente para ser capaz de processar uma alta quantidade de *frames* que fez com que o FPS dos

Tabela 5 – Resultados dos testes em imagens sequenciais.

CAT	**SEQ**	**OI**	**FP**	**FC**	**FC(%)**	**TP(s)**	**FPS**
1	1	1	1794	1704	94,98%	60	29,90
2	2	2	1730	1653	95,55%	60	28,83
3	3	3	1687	1663	98,58%	60	28,12
4	4	4	1694	1631	96,28%	60	28,23
5	5	5	1710	1683	98,42%	60	28,50
Média	-	-	1723	1667	96,76%	-	28,72

Elaborado pelo autor.

testes se aproximassem muito do que geralmente acontece em capturas de vídeos em tempo real (30 FPS).

7 Estudos de caso

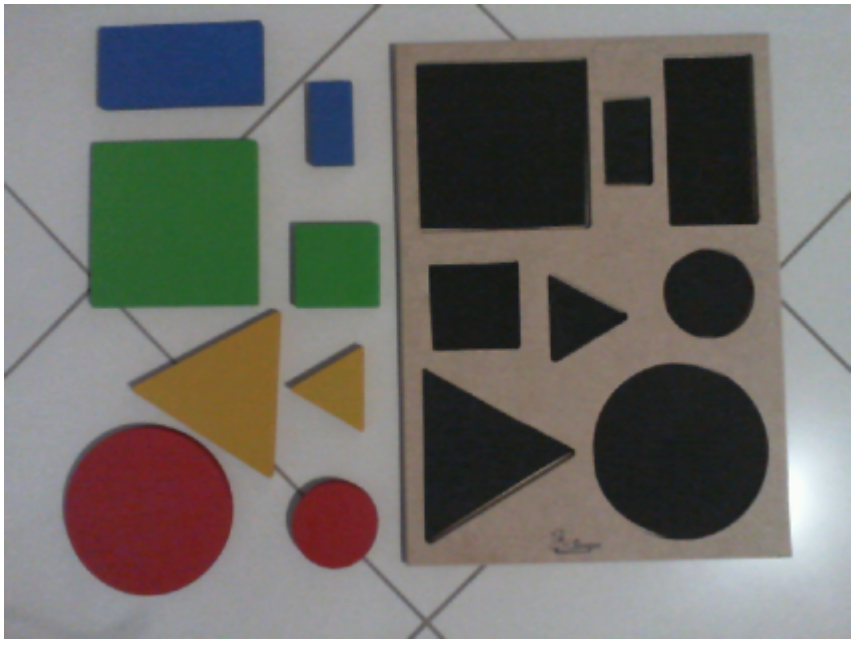
Uma das motivações para o desenvolvimento deste trabalho é proporcionar um processo simples, flexível e eficiente que seja capaz de ajudar na resolução de problemas que possam ser beneficiados pelo reconhecimento e rastreamento de objetos. Objetivando demonstrar que o processo realmente atende a estes requisitos, foram elaborados e executados três estudos de caso que são apresentados nas seções a seguir.

7.1 Resolução computacional de um brinquedo pedagógico de encaixe de figuras geométricas

O reconhecimento de objetos em sua essência é uma funcionalidade útil, mas ela se torna ainda mais interessante quando algum tipo de lógica é aplicada sobre os objetos reconhecidos. Para demonstrar que o processo especificado neste trabalho dá suporte para a realização de tal feito, este estudo de caso foi desenvolvido.

Neste estudo de caso, desenvolveu-se um *software* capaz de apontar a solução de um brinquedo pedagógico de encaixe de figuras geométricas direcionado à crianças de 3 a 5 anos de idade. Como pode ser visto na Figura 37, o brinquedo é composto por 8 figuras geométricas de formatos, cores e tamanhos diferentes e por um peça de madeira com buracos onde se encaixam essas figuras. O desafio do brinquedo está em descobrir em qual buraco se encaixa determinada figura geométrica.

Figura 37 – Brinquedo pedagógico de encaixe de figuras geométricas.



Fonte: Elaborado pelo autor.

Por possuírem cores únicas e serem formas geométricas simples, o *software* desenvolvido para a avaliação do processo especificado neste trabalho já é capaz de analisar uma imagem do brinquedo e reconhecer como objetos tanto as figuras geométricas quanto os buracos onde elas se encaixam, porém, não é capaz de criar uma relação entre eles. Para tanto, o *software* deve ser adaptado para realizar as seguintes tarefas:

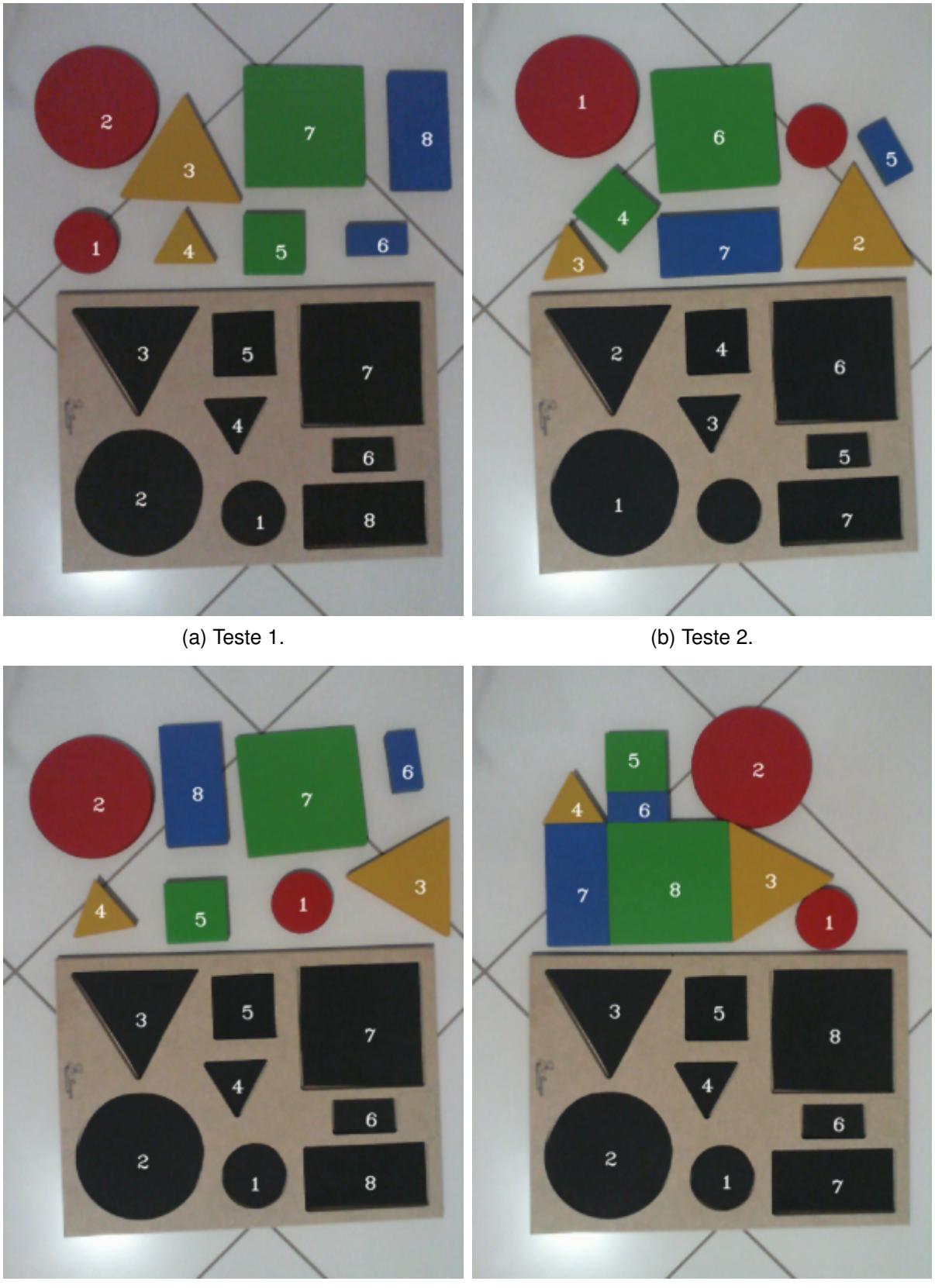
1. **Distinguir as figuras geométricas dos buracos de encaixe:**
   Enquanto as figuras geométricas são de cor vermelha, azul, verde ou amarela, os buracos de encaixe são pretos. Ou seja, a distincão entre eles é feita pela cor: os buracos são da cor preta e as figuras geométricas de quaisquer outras cores.

2. **Analisar as semelhanças entre a figura geométrica e o buraco de encaixe:**
   Além da cor, a figura geométrica e o buraco de encaixe possuem outras duas características fundamentais: formato e tamanho. Pela natureza do brinquedo, são essas as características que devem ser avaliadas na determinação da semelhança entre os objetos. A identificação do formato dos objetos já acontece, mas não a do tamanho. Para que isso aconteça, basta analisar os valores dos descritores de cada objeto. O tamanho de um círculo pode ser estimado pelo valor de seu raio e o tamanho de um triângulo ou retângulo pode ser estimado pelas distâncias entre seus vértices.

3. **Associar a figura geométrica ao seu respectivo buraco de encaixe:**
   A partir da análise de semelhança entre as figuras geométricas e os buracos de encaixe, é necessário associar os pares que melhor se correspondem. Além disso, deve-se definir um marcador visual para que esta associação fique clara na imagem. Por exemplo, pode-se numerar as associações encontradas.

Feitas as devidas adaptações, testes foram realizados e as imagens resultantes podem ser visualizadas na Figura 38. Observa-se que o *software* apresenta resultados satisfatórios, já que é capaz de fazer a associação correta na maioria dos casos (Um único círculo não foi associado na Figura 38b). Nota-se também que a associação é feita independendo da posição ou da rotação dos objetos, o que é uma competência relevante.

Figura 38 – Resolução computacional de um brinquedo pedagógico de encaixe de figuras geométricas.



(a) Teste 1. (b) Teste 2.

(c) Teste 3. (d) Teste 4.

Fonte: Elaborado pelo autor.

7.2 Classificação de camisas

Uma das mais importantes características do processo especificado neste trabalho é o uso de métodos exatos na interpretação dos objetos reconhecidos em imagens. Esta abordagem permite que empiricamente se identifique as características peculiares de cada objeto e que estas sejam facilmente explicitadas na etapa de classificação. Devido a esta simplicidade, o processo torna-se mais flexível em relação à definição de categorias as quais pertencem os objetos de interesse das imagens. Para demonstrar esta flexibilidade, este estudo de caso foi desenvolvido.

Neste estudo de caso, desenvolveu-se um *software* capaz de identificar a cor e o formato de uma camisa. O reconhecimento de cores (devidamente calibradas) é uma funcionalidade intrínseca do processo especificado, mas reconhecer formatos de camisas extrapola o escopo básico definido. Esta é uma situação onde é necessário descobrir quais são as características peculiares que definem o formato de uma camisa e utilizá-las na criação das regras de seu respectivo classificador. Neste contexto, considera-se dois formatos de camisa:

1. **Camisa básica:** Camisa com manga curta;
2. **Camisa regata:** Camisa sem mangas.

Por serem objetos complexos, em uma primeira impressão, a classificação pode também parecer complexa, mas certos recursos podem ser utilizados para que este problema não ocorra. Ajustando alguns parâmetros e critérios da detecção de polígonos do *software* desenvolvido para a avaliação do processo especificado neste trabalho, percebe-se nos resultados obtidos que as camisas podem ser reconhecidas como polígonos de 5 ou 7 vértices. Por um processo de dedução empírica, nota-se que o causador da diferença do número do polígono reconhecido é exatamente o formato da camisa. Camisas básicas são reconhecidas como polígonos de 7 vértices e camisas regatas como polígonos de 5 vértices. Identificadas essas características peculiares, basta que elas sejam utilizadas como regras de classificação.

Criadas essas novas regras de classificação, testes foram realizados e as imagens resultantes podem ser visualizadas na Figura 39. Observa-se que o *software* apresentou resultados satisfatórios, sendo capaz de classificar todas as camisas pela cor e formato correto. Nota-se também que a classificação por cores é feita corretamente mesmo que a camisa possua estampas, ou detalhes que façam com que ela não seja completamente de uma única cor, já que nestes casos foi considerada a cor prenominante da camisa.

Figura 39 – Classificação de camisas.

(a) Teste 1.

(b) Teste 2.

(c) Teste 3.

(d) Teste 4.

Fonte: Elaborado pelo autor.

7.3 Cursores para desenho

Um dos parâmetros utilizados para avaliar o desempenho do reconhecimento de objetos é o tempo de reposta. Um tempo de resposta suficientemente baixo permite que o reconhecimento de objetos seja aplicado em situações mais críticas, como por exemplo, casos de rastreamento de objetos em uma sequência de imagens capturadas em tempo real. Este estudo de caso foi desenvolvido para demonstrar que o processo especificado neste trabalho pode ser aplicado nestes casos.

Neste estudo de caso foi desenvolvido um *software* capaz de reconhecer e rastrear cursores em tempo real enquanto estes desenham ou apagam figuras sobre uma cena filmada por uma *webcam*. Este tipo de problema deve ser abordado com critérios de sistemas de tempo real porque a movimentação do cursor é em tempo real e deseja-se rastrear a sua movimentação da forma mais precisa possível. No contexto deste trabalho, são considerados os seguintes cursores:

- **Pincel verde**: um quadrado de qualquer tamanho de cor verde;
- **Pincel vermelho**: um quadrado de qualquer tamanho de cor vermelha;

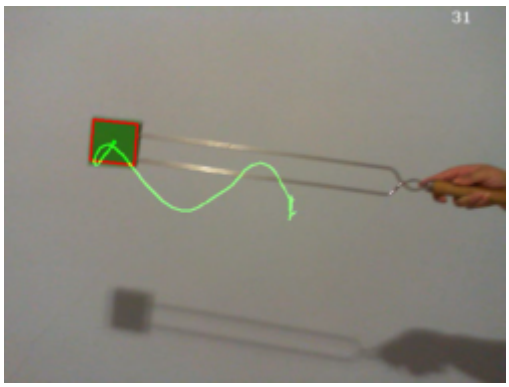
- **Borracha**: um quadrado de qualquer tamanho de cor preta.

Cada um desses cursores movimenta-se em frente a uma superfície uniformemente iluminada de cor branca. Quando o cursor que estiver se movimentando for algum pincel, sua trajetória é rastreada e tracejada (com a cor do pincel) sobre a imagem que estiver sendo capturada. Quando o cursor for a borracha, o mesmo rastreamento acontece, porém, será apagado qualquer tracejado presente em sua trajetória.

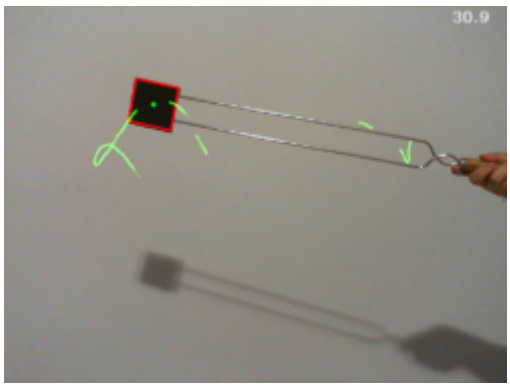
O *software* desenvolvido para a avaliação do processo especificado neste trabalho é capaz de reconhecer os objetos que são considerados como cursores e apresenta um baixo tempo de resposta. Então, basta adequá-lo para que a trajetória dos cursores seja registrada continuamente e utilizada na realização de suas respectivas ações (desenhar ou apagar tracejados).

Feitas estas adequações, testes foram realizados e amostras dos resultados obtidos podem ser visualizadas na Figura 40. Em geral, os testes realizados apresentaram uma taxa média de 29,7 *frames* processados por segundo durante a captura de imagens. Sendo 30 *frames* por segundo a taxa padrão de captura de vídeos em tempo real, pode-se concluir que a alta performance de processamento do *software* desenvolvido tornou o capaz de atuar como um sistema de tempo real.

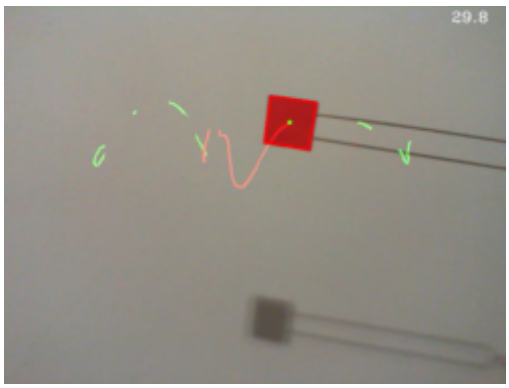
Figura 40 – Cursores para desenho.



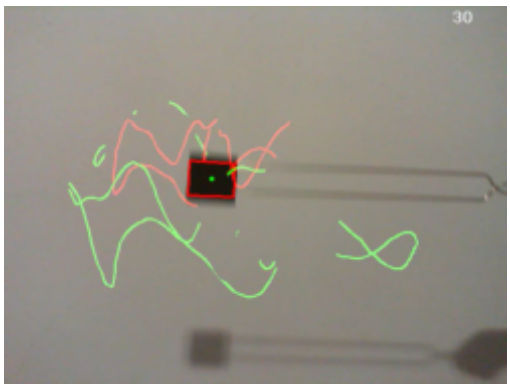
(a) Tracejando em verde.



(b) Apagando tracejado verde.



(c) Tracejando em vermelho.



(d) Apagando tracejado verde e vermelho.

Fonte: Elaborado pelo autor.

8 Conclusão

Com base nos estudos e nas pesquisas realizadas, conforme referencial bibliográfico, este trabalho propôs um processo capaz de reconhecer e rastrear objetos. Este processo descreve um mecanismo que utiliza métodos exatos e é capaz de reconhecer objetos coloridos com formas geométricas simples (triângulos, retângulos e círculos) em uma imagem ou em uma sequência de imagens. Deste modo, o objetivo geral dessa dissertação foi alcançado.

Os objetivos específicos também foram alcançados. A implementação do processo especificado foi capaz de reconhecer objetos tanto em imagens quanto em sequências de imagens capturadas em tempo real. A simplicidade, flexibilidade e eficiência do método proposto foram demonstradas por meio de testes e estudos de caso. Isto permitiu que o processo fosse avaliado em diferentes contextos.

O desenvolvimento desta pesquisa apresentou algumas dificuldades. Uma das mais marcantes está relacionada à extração das características dos objetos presentes nas imagens. Além disso, a preocupação em especificar um processo eficiente o suficiente para atender requisitos de tempo real levou à investigação de técnicas que reduzam a complexidade do reconhecimento de objetos e consequentemente, seu custo computacional. Entretanto, a busca de soluções para esses problemas enriqueceram o desenvolvimento deste trabalho.

De forma geral, a especificação de um processo de natureza simples, flexível e eficiente é relevante para incentivar e auxiliar futuros trabalhos e assim ampliar a quantidade de problemas e situações capazes de serem beneficiados por recursos computacionais. Sendo assim, as principais contribuições deste trabalho são:

- Levantamento, avaliação e documentação das etapas do processamento digital de imagens;
- Avaliação do OpenCV como uma ferramenta computacional auxilar no processamento digital de imagens;
- Especificação e implementação de um processo de reconhecimento e rastreamento de objetos de baixo custo computacional;
- Implementação de protótipos de *software* de reconhecimento de objetos que atuam cenários reais.

Entre os trabalhos futuros, considerados para a extensão deste projeto, estão:

- Propor alternativas para a automatização da identificação de cores em imagens;
- Especificar classificadores de objetos com características complexas;

- Avaliar e comparar a implementação do processo utilizando outras ferramentas computacionais;
- Adaptar o processo para mapear profundidades durante o reconhecimento de objetos.
- Implementar o processo especificado em uma arquitetura de *hardware*.

Referências

BRADSKI, G. *Dr. Dobb's Journal of Software Tools*, 2000. Citado na página 43.

BRADSKI, G.; KAEHLER, A. *Learning OpenCV: Computer vision with the OpenCV library*. [S.l.]: "O'Reilly Media, Inc.", 2008. Citado nas páginas 33 e 43.

CANNY, J. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, n. 6, p. 679–698, 1986. Citado nas páginas 7, 39, 51 e 56.

CARVALHO, G. et al. Um sistema de monitoramento para detecçao de objetos em tempo real empregando câmera em movimento. *Proc. Simp. Brasileiro de Telecomunicações, Fortaleza, Brazil*, 2013. Citado nas páginas 13 e 14.

FERREIRA, C. S. *Implemantação do algoritmo de subtração de fundo para detecção de objetos em movimento, usando sistemas reconfiguráveis*. 2012. Dissertação (Mestrado) — Universidade de Brasília, 2012. Citado nas páginas 21 e 22.

FILHO, O. M.; NETO, H. V. *Processamento digital de imagens*. [S.l.]: Brasport, 1999. Citado nas páginas 25, 28, 29, 34 e 46.

GONZALES, R. C.; WOODS, R. E. *Processamento Digital de Imagens*. third. São Paulo: Pearson Prentice Hall, 2010. Citado nas páginas 13, 15, 25, 26, 28, 30, 31, 32, 33, 34, 37, 39, 40, 42, 47 e 48.

HEIKKILÄ, J.; SILVÉN, O. A real-time system for monitoring of cyclists and pedestrians. *Image and Vision Computing*, Elsevier, v. 22, n. 7, p. 563–570, 2004. Citado nas páginas 14 e 22.

OIKONOMIDIS, I.; KYRIAZIS, N.; ARGYROS, A. A. Efficient model-based 3d tracking of hand articulations using kinect. In: *BMVC*. [S.l.: s.n.], 2011. v. 1, n. 2, p. 3. Citado na página 24.

OLIVEIRA, B. A. S. et al. Detecçao e rastreamento de objetos coloridos em vídeo utilizando o opencv. *VII Semana de Ciência e Tecnologia IFMG - campus Bambuí - VII Jornada Científica e I Mostra de Extensão, Bambuí, Brazil*, 2013. Citado na página 23.

OLIVEIRA, R. B. *Método de detecção e classificação de lesões de pele em imagens digitais a partir do modelo Chan-Vese e máquina de vetor de suporte*. 2012. Dissertação (Mestrado) — Universidade Estadual Paulista, 2012. Citado nas páginas 13 e 22.

PEDRINI, H.; SCHWARTZ, W. R. *Analise de Imagens Digitais: Princípios, Algoritmos e Aplicações*. São Paulo: Thomson Learning, 2008. Citado nas páginas 13, 26 e 38.

PERMALOFF, A.; GRAFTON, C. Optical character recognition. *PS: Political Science & Politics*, Cambridge Univ Press, v. 25, n. 03, p. 523–531, 1992. Citado na página 14.

RAMER, U. An iterative procedure for the polygonal approximation of plane curves. *Computer graphics and image processing*, Elsevier, v. 1, n. 3, p. 244–256, 1972. Citado nas páginas 7, 40, 41, 48, 49, 51 e 53.

SEO, Y. et al. Where are the ball and players? soccer game analysis with color-based tracking and image mosaick. In: SPRINGER. *Image Analysis and Processing*. [S.l.], 1997. p. 196–203. Citado na página 14.

SERRE, T. et al. On the role of object-specific features for real world object recognition in biological vision. In: SPRINGER. *Biologically Motivated Computer Vision*. [S.l.], 2002. p. 387–397. Citado na página 14.

SHAW, A. C. *Sistemas e software de tempo real*. [S.l.]: Bookman, 2003. Citado na página 14.

SILVA, B. R. d. A. e. *Sistema de contagem automática de objetos utilizando processamento digital de imagens em dispositivos móveis*. 2014. Dissertação (Mestrado) — Universidade do Estado do Rio Grande do Norte, 2014. Citado nas páginas 16 e 42.

SOLOMON, C.; BRECKON, T. *Fundamentals of Digital Image Processing: A practical approach with examples in Matlab*. [S.l.]: John Wiley & Sons, 2011. Citado nas páginas 26, 27, 28, 32, 34, 35, 36, 38 e 39.

SOLOMON, C.; BRECKON, T. *Fundamentals of Digital Image Processing - High Resolution Book Figures*. 2015. Disponível em: <http://www.fundipbook.com/>. Citado nas páginas 27, 28 e 29.

SOUZA, L. R. de. *Algoritmo para reconhecimento e acompanhamento de trajetória de padrões em vídeo*. 2011 — Universidade Federal do Vale do São Francisco, 2011. Citado na página 23.

SUZUKI, S. et al. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, Elsevier, v. 30, n. 1, p. 32–46, 1985. Citado nas páginas 40, 41, 52 e 53.

WITTMAN, T. Mathematical techniques for image interpolation. *Report Submitted for Completion of Mathematics Department Oral Exam, Department of Mathematics, University of Minnesota, USA*, 2005. Citado na página 30.

YUEN, H. et al. Comparative study of hough transform methods for circle finding. *Image and vision computing*, Elsevier, v. 8, n. 1, p. 71–77, 1990. Citado nas páginas 7, 41 e 53.