UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ – UTFPR
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JONATAN LIECHESKI

RECONHECIMENTO DE CARTÕES RESPOSTAS POR MEIO DE PROCESSAMENTO DIGITAL DE IMAGENS

TRABALHO DE DIPLOMAÇÃO

MEDIANEIRA
2015

JONATAN LIECHESKI

RECONHECIMENTO DE CARTÕES RESPOSTA POR MEIO DE PROCESSAMENTO DIGITAL DE IMAGENS

Trabalho de Diplomação apresentado à disciplina de Trabalho de Diplomação, do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas – COADS – da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Tecnólogo.

Orientador: Prof. Ricardo Sobjak.

Co-orientador: Prof. Pedro Luiz de Paula Filho.

MEDIANEIRA

2015

“Ninguém é suficientemente perfeito, que não possa apreender com o outro e, ninguém é totalmente destituído de valores que não possa ensinar algo ao seu irmão, Paz e bem” – São Francisco de Assis

AGRADECIMENTOS

A Deus, primeiramente, por guiar meus caminhos durantes estes anos de graduação, me iluminando e dando forças para vencer os obstáculos que a vida nos impõem.

A todos os professores que de alguma forma contribuíram tanto na minha formação acadêmica como pessoal, pois foram essenciais para que me torna-se a pessoa que sou. Em especial aos professores Pedro Luiz de Paula Filho e Ricardo Sobjak, que acreditaram e me auxiliaram na conclusão deste estudo.

Aos meus pais Augustinho Liecheski e Lurdes Fabris Liecheski por me proporcionarem essa oportunidade de um futuro promissor e por tantas vezes que deixaram os seus sonhos para realizar os meus, abrindo mão das suas vontades para realizar meus caprichos, ao meu irmão Jober Augusto Liecheski que sempre esteve ao meu lado nos momentos bons e ruins, e também aos meus familiares, pela confiança e apoio para que de alguma forma meus objetivos fossem alcançados.

A minha namorada Daiane Aline Tomaz, que me auxiliou em muitos momentos da minha vida, nunca deixando que eu desista de meus sonhos e objetivos, que através de conselhos sempre me apoiou na elaboração deste projeto.

Por fim, gostaria de agradecer a todos aqueles que diretamente ou indiretamente contribuíram para o desenvolvimento deste trabalho e da minha formação acadêmica. Meu eterno agradecimento.

RESUMO

LIECHESKI, Jonatan. Reconhecimento de cartões resposta por meio de processamento digital de imagens. 2015. 67 f. Trabalho de Diplomação (Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas). Universidade Tecnológica Federal do Paraná. Medianeira, 2015.

A evolução da computação no mundo moderno é constante. Nesta evolução cita-se o processamento de imagens como um dos fatores de crescimentos. O processamento de imagens é muito utilizado e relevante em diversas áreas, pois por meio deste processo é possível a retirada de informações contidas em uma imagem. O processo de extração de informações em uma imagem não é tarefa fácil, pois requerer o uso de técnicas às vezes muito complexas e necessita dados de uma resolução qualidade para chegar a um resultado satisfatório. Esse projeto tem como objetivo apresentar técnicas de reconhecimento de imagens com o intuito de reconhecer e extrair informações das questões assinaladas em um cartão resposta por meiodo uso da biblioteca gráfica OpenCV, comparando os algoritmos SIFT, SURF e *Template Matching* visando apresentar a técnica que subentende os requisitos em relação ao reconhecimento e correção dos cartões respostas.

Palavras-chaves: Processamento de Imagens, Processamento Digital de Imagens (PDI), Cartão Resposta.

ABSTRACT

LIECHESKI, Jonatan. CARDS RECOGNITION IN RESPONSE DIGITAL IMAGE PROCESSING MEANS. 2015 67 f. Working graduation (Course of Technology in Systems Analysis and Development). Federal Technological University of Paraná. Medianeira 2015.

The evolution of computing in the modern world is constant. This evolution refers to the image processing as one of the growth factors. The image processing is widely used and relevant in various areas, for through this process is possible to remove information contained in an image. The process of information extraction in an image is no easy task as require the use of techniques to very complex and often require data from a resolution quality to reach a satisfactory result. This project aims to present image recognition techniques in order to recognize and extract information from the issues raised in a response card for meiodo use of OpenCV graphics library , comparing the SIFT algorithms , SURF and Template Matching aiming to present the technique that implies the requirements regarding the recognition and correction of the cards answers.

Keywords: Image Processing, Digital Image Processing (PDI), Response Card.

LISTA DE SIGLAS

API	*Application Programming Interface*
BRIEF	*Binary Robust Independent Elementary Features*
BSD	*Berkeley Software Distribution*
FAST	*Features from Accelerated Segment Test*
GB	*GigaBytes*
GFTT	*Good Features To Track Detector*
IDE	*Integrated Development Environment*
MSER	*Maximally Stable Extremal Regions*
OMR	*Optical Marc Recognition*
OpenCV	*Open Source Computer Vision Library*
ORB	*Oriented FAST and Rotated BRIEF*
PDI	Processamento Digital de Imagens
RAM	*Random Access Memory*
RGB	*Red, Green, Blue*
SIFT	*Scale Invariant Feature Transform*
STAR	*Solenoidal Tracker at RHIC*
SURF	*Speeded Up Robust Features*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

SUMÁRIO

1 INTRODUÇÃO

Processamento digital de imagens (PDI) está relacionado ao auxílio humano, apresentando resultados que, aos olhos humanos, não são visualizados. A visão biológica do ser humano é a principal ferramenta para exploração e descoberta dos conhecimentos gerais, assim é de extrema importância e trivial para o homem (PLOTZE; BRUNO, 2007).

Por O uso de técnicas computacionais de aprimoramento de imagens teve seus primeiros estudos por volta de 1960 em *Jet Propulsion Laboratory Massachusetts Institute of Technology – MIT, Bell Labs e University of Maryland*, com aplicação para imagens de satélite, conversão de padrões de *wirephoto*, imagens médicas, videofone, reconhecimento de caracteres e aprimoramento de foto. Somente a partir de 1970, o processamento de imagens cresceu devido ao valor reduzido do custo dos computadores e hardwares dedicados. Atualmente técnicas de processamento de imagens são adicionalmente utilizadas tanto na criação de efeitos especiais como na atenuação de artefatos digitais encontrados em imagens sintéticas (TING, 2009).

O PDI teve um grande crescido nos últimos anos, com a utilização de imagens e gráficos em uma grande variedade de aplicações, possibilitando a utilização de sistemas mais eficientes e mais baratos. Muitas áreas vêm utilizando sistemas de processamento digital de imagens como, por exemplo: reconhecimento de padrões, meteorologia, pesquisas espaciais, medicina entre outros.

Atualmente muitas organizações que realizam concursos, provas ou pesquisas de opiniões, apresentam gabaritos em forma de cartão resposta, os equipamentos para a correção destes cartões possuem um alto custo de aquisição. Neste sentido, o objetivo deste trabalho foi estudar um mecanismo capaz de efetuar o reconhecimento digital de imagens com a finalidade de reconhecer as respostas assinaladas pelo candidato, a partir de imagens digitalizadas com um dispositivo de menor custo como um scanner convencional.

Outros trabalhos foram realizados com propósitos semelhantes, tal como o desenvolvimento de uma biblioteca, na plataforma Java, capaz de avaliar respostas assinaladas, como solução alternativa para o processo convencional conhecido como OMR (*Optical Marc Recognition*) neste trabalho, o reconhecimento ocorre com base em um cartão pré-definido pelo software (OLIVEIRA NETO, 2010). Silva e Paiva. (2013) realizaram um estudo comparativo utilizando diferentes técnicas para detecção de pontos chave e descritores em imagens. As imagens foram analisadas utilizando os detectores SIFT, SURF, FAST,

STAR, MSER, GFTT (com Harris), GFTT e ORB, e os descritores: SIFT, SURF, BRIEF e ORB.

1.1 OBJETIVOS

1.1.1 Objetivo geral

Reconhecer as respostas em cartão resposta por meio de técnicas de processamento de imagens e reconhecimento de padrões.

1.1.2 Objetivos específicos

Os objetivos específicos deste estudo são:

- Estudar técnicas de processamento digital de imagens aplicáveis ao problema em questão;
- Processar, detectar e identificar as imagens utilizando API OpenCV;
- Aplicar algoritmos *Template Matching*, *SIFT* e *SURF* no reconhecimento de respostas de cartões;
- Avaliar a melhor técnica para reconhecimento das respostas.

1.2 JUSTIFICATIVA

A proposta deste estudo se dá pelo fato da dificuldade que se encontra em corrigir cartões respostas de provas que são elaboradas na Universidade, em que estes cartões são enviados para outra cidade para assim efetuar as correções dos mesmos. Com técnicas

elaboradas no processamento de imagens estas correções podem se tornar mais rápidas e simples.

Várias técnicas de processamento são aplicadas com a biblioteca OpenCV que, de acordo com Marengoni e Stringhini (2013) possibilita ao desenvolvedor uma vasta de opção de métodos de manipulações e processamento de imagens digitais, assim como acesso simplificado para a realização de tarefas necessárias com as imagens a serem reconhecidas.

No trabalho de Silva e Paiva (2013) utilizando os detectores SIFT, SURF, FAST, STAR, MSER, GFTT (com Harris), GFTT e ORB, e os descritores: SIFT, SURF, BRIEF e ORB, os parâmetros observados quanto ao tempo de processando e geração de matriz homográfica, mostraram-se capazes de realizar satisfatoriamente a correspondência entre objetos e imagens das cenas.

1.3 ESTRUTURA DO TRABALHO

Este trabalho é estruturado da seguinte maneira:

a. **Capítulo I:** Neste primeiro capítulo é apresentada a introdução do trabalho, bem como seus objetivos e a justificativa para a sua realização.

b. **Capítulo II:** Neste capítulo são apresentados, as técnicas utilizadas na elaboração deste trabalho no formato de revisão de literatura, apresentando conceitos de imagens, processamento de imagens e algoritmos específicos que serão estudados, apresentando seus conceitos e técnicas baseadas e demais abordagens julgadas necessárias para o desenvolvimento deste trabalho.

c. **Capítulo III:** Contém todas as técnicas utilizadas na elaboração deste trabalho apresentando passos específicos para a detecção e correção do cartão resposta.

d. **Capítulo IV:** Apresenta os resultados do trabalho realizado.

e. **Capítulo V:** Último capítulo se tem como objetivo apresentar as considerações finais e sugestões para trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

2.1 IMAGEM DIGITAL

A palavra imagem tem origem do termo latim “imago”, que significa representação visual de um objeto. As imagens podem ter origens diversas, podem ser captadas por comprimento de onda de radiação eletromagnética (máquinas fotográficas) ou por ondas sonoras de altas frequências (ultrassom). Em imagens que são captadas por radiação, elas podem ser obtidas por radiações refletidas por objetos iluminados por fontes, por radiações absorvidas por objetos translúcidos ou diretamente do emissor de radiação (objetos emitentes) (GONZALES; WOODS, 2010).

Como já diz o ditado popular “Uma imagem vale mais que mil palavras”, a comunicação baseada em imagens vem desde o tempo das cavernas, em que o ser humano desenhava para transmitir conceitos diretamente relacionados ao que os cercava (TING, 2009).

Após a energia luminosa ser refletida ou irradiada pelo objeto, e sensibilizar um dispositivo de captação ou visão, seja este, o olho humano, um sensor ou uma câmera, a imagem passa a ser processada e pode ser analisada pelo cérebro ou de forma computacional (GONZALES; WOODS, 2010).

2.2 PROCESSAMENTO DIGITAL DE IMAGENS

De acordo com Spring (1996), processamento de digital de imagens pode ser definido como a manipulação de uma imagem por computador, de modo que a entrada e a saída do processo sejam imagens.

Silva (2001) ainda relata que a função primordial do processamento de imagens é a extração de informações que facilitam a identificação das imagens, assim posteriormente com as informações recolhidas são realizadas as interpretações desejadas.

Segundo Gonzales e Woods (2010), processamento digital de imagens (PDI) pode ser definido como uma função bidimensional, $f(x, y)$, em que x e y são coordenadas espaciais

(plano), e a amplitude de *f* em qualquer par de coordenadas (x, y) é chamada de intensidade de nível de cinza da imagem nesse ponto. Quando (x, y) e os valores de intensidade de *f* são quantidades finitas e discretas chama-se de imagem digital. Uma imagem digital é composta de um número finito de elementos, cada um com localização e valores específicos. Esses elementos são chamados de elementos pictóricos, elementos de imagem e pixels. Pixels é o termo mais usado para representar os elementos de uma imagem digital.

Um sistema de processamento digital de imagens é constituído por um conjunto de etapas, ilustradas na Figura 1 seguindo conceitos de Gonzales e Woods (2010), que apresentam as cinco fases ou passos fundamentais para que a execução de uma tarefa de PDI tenha sucesso.

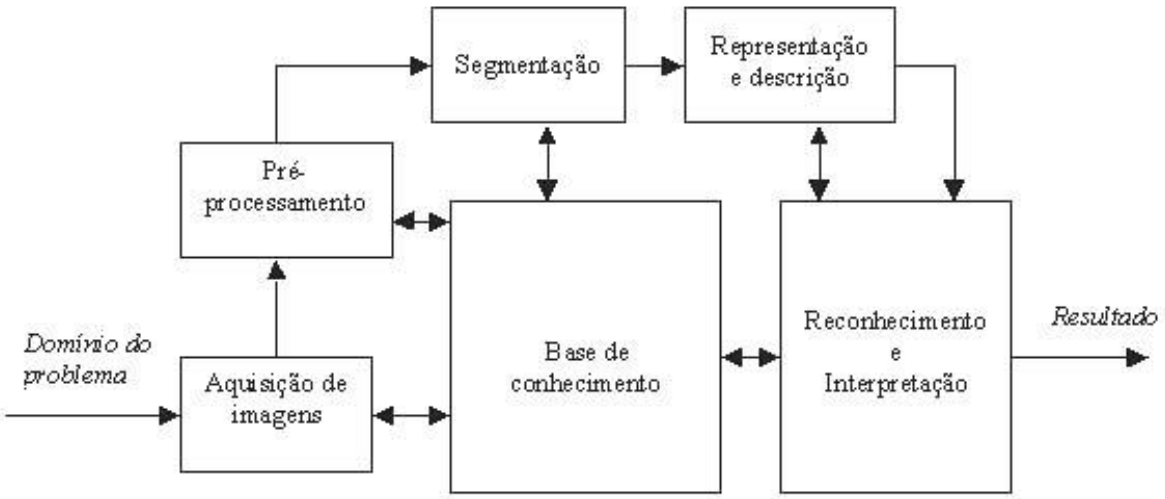


Figura 1 - Etapas de um Sistema de Processamento de Imagens.
Fonte: Gonzales e Woods (2010).

A primeira etapa compreende a aquisição de imagens, que é realizada por intermédio de um dispositivo com capacidade de digitalização e em seguida converte a imagem em uma representação adequada para o processamento da mesma. Os principais dispositivos para a aquisição de imagens são as câmeras de vídeos e scanners (GONZALES; WOODS, 2010).

A função *f (x, y)*, é o resultado do produto entre a luminância e a reflectância do objeto, com a respectiva resposta da luz do ambiente. A digitalização dos valores de coordenadas é chamada de amostragens sendo o grau de detalhes perceptíveis, a resolução espacial (PEDRINI; SCHWARTZ, 2008).

Na aquisição de imagens coloridas cada componente da imagem RGB é armazenada separadamente por sua camada sendo a camada "*Red*", a camada "*Green*" e a camada "*Blue*". A imagem é nada mais, que uma função bidimensional de intensidade da luz, sendo a *f (x, y)*,

em que x e y são as coordenadas, e o valor *f* em qualquer ponto (x, y) proporciona um nível na imagem, (PEDRINI; SCHWARTZ, 2008), conforme ilustrado na Figura 2

Figura 2 - Convenção dos Eixos para Representação de Imagens.
Fonte: PEDRINI; SCHWARTZ (2008). PEDRINI; SCHWARTZ (2008).

A etapa seguinte é o pré-processamento em que o mesmo, consiste em analisar a imagem capturada, na qual serão aplicadas transformações lineares ou não-lineares visando o melhoramento de contraste, remoção de ruído, regiões de interesses, de correlação e codificação das informações entre outras (GONZALES; WOODS, 2010).

A segmentação de imagens particiona a imagem em regiões disjuntas com algum significado para a aplicação. Nesta fase pode ser identificada e separada as áreas de interesses na imagem, a segmentação pode ser considerada um dos processos mais difíceis para o PDI, o mal uso de algoritmos nesta fase pode acarretar muito o resultado final, não condizendo com objetivos esperados (GONZALES; WOODS, 2010).

No processo da representação e descrição, os dados quase sempre partem do resultado da segmentação, que normalmente são dados primários em forma de pixels. A etapa de representação transforma dados primários em uma forma apropriada para o próximo passo no processamento computacional, o processo de descrição realizada a extração de atributos que resultam em informações quantitativas de interesse (MARQUES FILHO; VIEIRA NETO, 1999).

A última etapa envolve o reconhecimento e a interpretação dos componentes da imagem. O reconhecimento é baseado nas características apresentadas pelos descritores. Pedrini e Schwarts (2008) relatam que o reconhecimento atribui um identificador ou rótulo

aos objetos na imagem. A interpretação atribui um significado ao conjunto de objetos reconhecidos.

A etapa da base do conhecimento é dependente na aplicação, ela deve ser utilizada para guiar a comunicação entre os módulos do processamento, assim executando as tarefas no seu momento certo (PEDRINI; SCHWARTS, 2008).

2.2.1 Cor

As cores são uma propriedade importante para a análise em imagens realizadas pelo ser humano, seja com, ou sem o auxílio do computador. A utilização da cor no processamento de imagens é um poderoso descritor que simplifica a identificação do objeto e sua extração de uma cena (GONZALES; WOODS, 2010).

O sistema de cores de Munsell foi criado pelo professor Albert H. Munsell por volta da década de 20, este sistema é perceptualmente uniforme, o qual possibilita um arranjo tridimensional das cores, utilizando os conceitos de matiz, pureza e luminosidade (ADAMS, 2014) como apresentado na Figura 3. Nesta a matiz (*hue*) é disposta no eixo circular, a pureza (*chroma*) no eixo radial e a luminosidade (*value*) no eixo vertical. Estes conceitos deram origem a modelos de cores como HSV (*hue, saturation, value*), HLS (*hue, lightness, saturation*), HSB (*hue, saturation, brightness*), dentre outros (GAMITO, 2005).

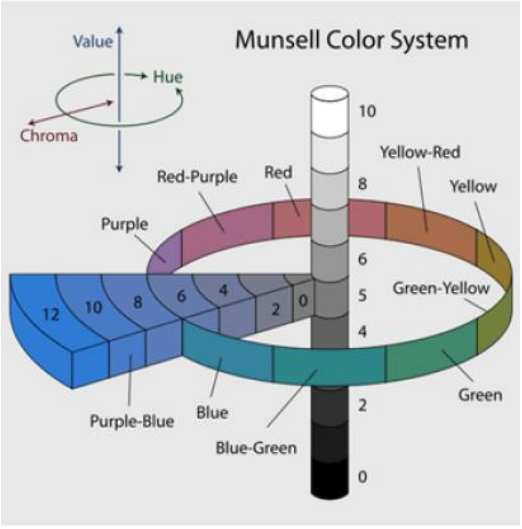


Figura 3 - Sistema de Cores de Munsell.
Fonte Fullcoat (2013)

Para processamento de imagens os modelos de cor mais trabalhados são: RGB, CMY (*Cyan, Magenta, Yellow*) ou também conhecido como CMYK utilizado normalmente em impressor e o modelo HSI (*Hue, Saturation, Intensity*). O RGB representa a cor natural com a combinação das cores vermelho, verde e azul, é um modelo de formação aditivo, em que as cores são criadas por adição e mistura de cores primárias. Baseadas em um sistema de coordenadas cartesianas como está ilustrado na Figura 4 em que os valores são divididos em oito vértices, em que as cores primárias (vermelho, verde e azul) estão representadas em três pontos e as cores secundárias (ciano, magenta e amarelo) estão em outros três vértices, o preto está na origem e o branco está no oposto do preto, localizado no inferior da origem (GONZALEZ; WOODS, 2010).

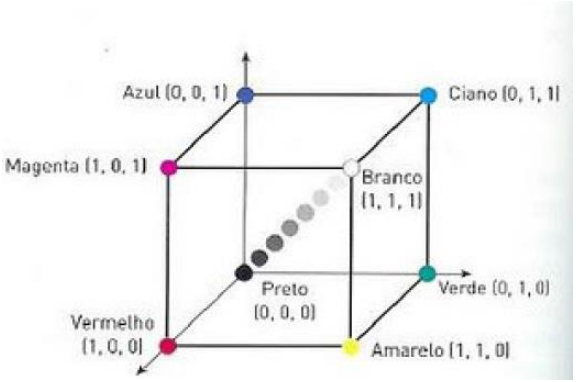


Figura 4 - Coordenadas Cartesiana Modelo RGB.
Fonte: Gonzales e Woods (2010).

Em imagens de modelo de cor RGB, as imagens possuem três canais, um para cada tonalidade de cor, quando combinados estes três canais é produzido uma imagem de cor composta. Cada canal desta cor primária é constituída por um valor de cor de 8 bits nestas condições cada pixel de cor RGB tem uma profundidade de 24 bits. A partir disto tem-se uma imagem *full-color* ou também conhecida como uma imagem colorida (GONZALES; WOODS, 2010).

2.2.2 Binarização (*Threshold*)

O algoritmo de binarização é normalmente utilizando quando é preciso fazer uma separação entre o fundo da imagem com os objetos da cena.

A binarizacão (*Threshold*) ou também conhecida como limiarização é o método mais simples para a segmentação de imagens que separa as regiões de interesse por meio da escolha de um ponto limiar. Existem casos em que não é possível dividir a imagem em apenas um limiar para obter um resultado satisfatório, nestes casos são definidos mais de um ponto de corte na imagem (SANTOS, 2011).

A escolha dos limiares é feitas de acordo com histograma dos pixels da imagem em escala de cinza, como exemplificado na Figura 5.

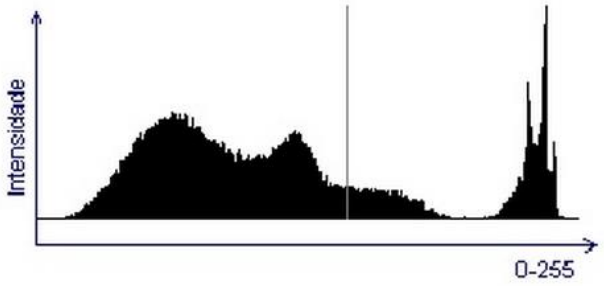


Figura 5 - Histograma de um exemplo de *Threshold.*
Fonte: Santos (2011).

A divisão da imagem em duas classes pode ser resolvida utilizando duas cores que são por padrão, as cores preto e branco. Por exemplo, de acordo com a Equação 1, as cores dos pixels que possuem um valor abaixo do ponto de corte terão sua cor alterada para branco (valor 255), e pixels que possuem valor acima do ponto de corte terão sua cor alterada para preto (valor 0) (SANTOS, 2011).

$$f(x,y) = \begin{cases} 255, se\ f(x,y) < T \\ 0, se\ f(x,y) > T \end{cases} \quad (1)$$

Equação 1- Equação que descreve a divisão das cores em um Threshold.
Fonte: Santos (2011).

Na Figura 6 tem-se um exemplo de binarização de uma imagem de tons de cinza (Figura 6(a)) que após binarizada é apresentada na Figura 6(b).

(a) Imagem original (b) Imagem binária

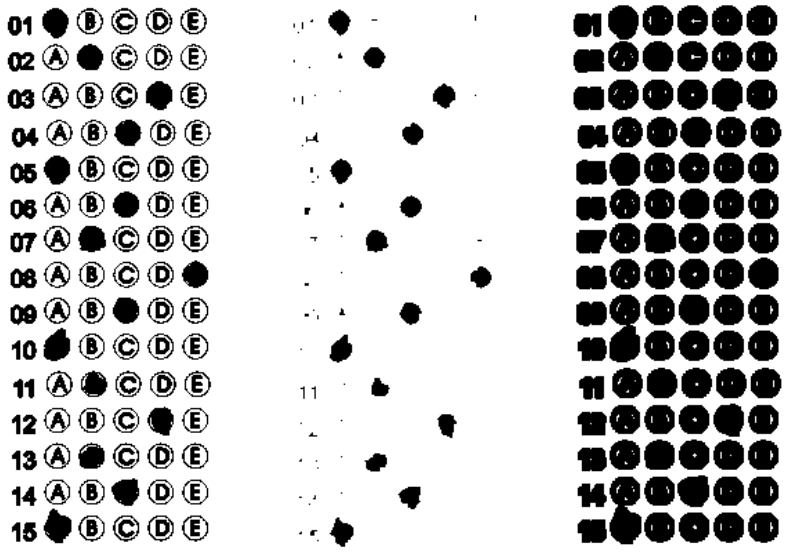
Figura 6 - Exemplo de binarização de uma imagem, (a) Imagem original e (b) Imagem binária. Fonte: Serrilho (2009).

2.2.3 Morfologia Matemática

Na morfologia matemática segundo os conceitos de Woods e Gonzalez (2010), o termo morfologia refere-se à forma, e o termo matemática diz respeito ao fato deste método utilizar conceitos da teoria dos conjuntos para obter informações de uma imagem.

Um dos principais objetivos da morfologia matemática é obter características dos objetos a partir de informações conhecidas de uma malha retangular denominada elemento estruturante (SILVA et al., 2006).

Existem vários métodos que realizam o processamento de imagens dentro da morfologia matemática, dois métodos que se destacam é a erosão e a dilatação, que podem ser consideradas as operações mais básicas da morfologia matemática. Na erosão, os pixels que não atendem um padrão estabelecido pelo elemento estruturante são eliminados, diminuindo os ruídos nas imagens, já na dilatação ocorre o oposto, os elementos são ampliados como se destaca na Figura 7.



(a) Imagem original binária | (b) Imagem com aplicação de erosão | (c) Imagem com aplicação de dilatação

Figura 7 - Exemplo da técnica de erosão e dilatação em uma imagem binária.
Fonte: Autoria própria.

2.2.4 Filtro de Sobel

Segundo Santos (2002) o filtro Sobel é um estimulador de bordas que tem por característica suavizar e detectar bordas ao mesmo tempo, sendo que o operador realça linhas verticais e horizontais mais escuras que o fundo sem realçar pontos isolados.

O filtro Sobel calcula o gradiente da intensidade da imagem em cada ponto, apresentando a maior variação entre o pixel claro, com pixel escuro e a quantidade de variação dos mesmos, assim obtém-se uma noção de como varia a luminosidade em cada ponto. Com isto consegue estimar a presença de uma transição claro-escuro e qual a orientação da mesma, a partir destas variações se tornam bem definidas as fronteiras entre os objetos, conseguindo assim a detecção dos contornos e bordas (GONZALEZ; WOODS; EDDINS, 2004).

O filtro Sobel utiliza duas máscaras deslocadas em 90° para encontrar os gradientes vertical e horizontal das bordas como apresentada na Figura 8. Seara (1998) ainda relata que o operador Sobel é muito menos sensível ao ruído devido às máscaras serem utilizadas a partir de uma matriz de 3x3 que comparado a outros filtros como o filtro de Roberts que possui matriz de 2x2.

A formulação matemática (Figura 8) utiliza duas matrizes 3x3 que são convoluídas com a imagem original para assim calcular as aproximações das derivadas, sendo uma para calcular as variações horizontais e outra para variações verticais (SEARA, 1998).

Sobel

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

$G_x=(z_7+2z_8+z_9)-(z_1+2z_2+z_3)$ $G_y=(z_3+2z_6+z_9)-(z_1+2z_4+z_7)$

Figura 8 - Formulação matemática do filtro Sobel.
Fonte: Gonzales; Woods; Eddins (2004).

Na Figura 9 são apresentados exemplos da aplicação do filtro Sobel. A Figura 9 (a) trata-se da imagem original. A Figura 9 (b) apresenta a aplicação do filtro na horizontal. A Figura 9 (c) é aplicação na vertical e a Figura 9 (d) é a junção das duas direções.

(a) Imagem original

(b) Aplicação do filtro Sobel na horizontal

(c) Aplicação do filtro Sobel na vertical

(d) Imagem formada com a junção das duas direções

Figura 9 - Exemplo de aplicação do filtro Sobel.
Fonte: Santos (2002).

2.2.5 OMR

Optical Mark Reading (OMR) é a tecnologia que permite interpretar e extrair eletronicamente informações de campos marcados (VERDE, 2000).

Um dos usos do algoritmo OMR, utiliza formulários de papel para corrigir provas respondidas em um cartão resposta. Um scanner com OMR corrige e envia as respostas diretamente a uma base de dados facilitando a correção com uma taxa de erro muito baixa e com um alto rendimento de correção (BERGERON, 1998). Na Figura 10 é apresentado um modelo de cartão resposta preenchido, no qual deve se extrair suas respostas.

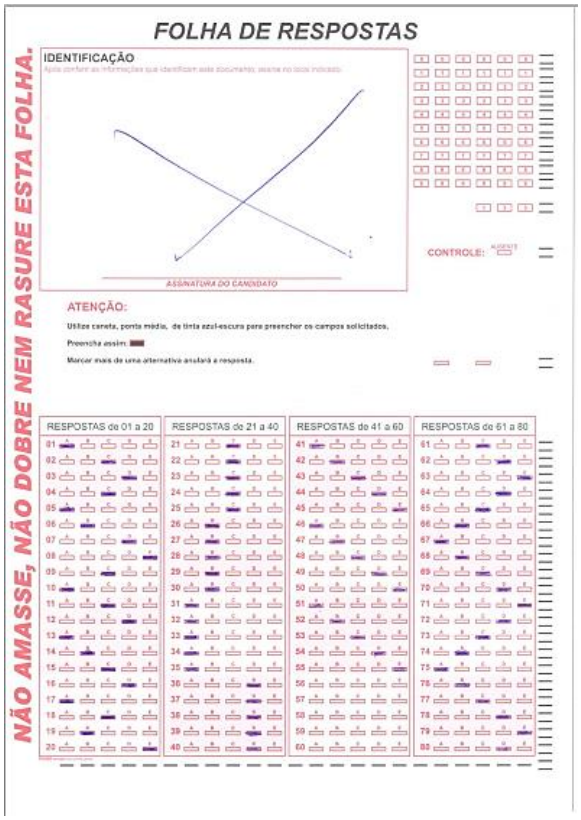


Figura 10 - Exemplo de cartão resposta preenchido para ser corrigido em um dispositivo OMR.
Fonte: Autoria Própria.

O OMR lê determinados tipos de marcas em um conjunto definido de locais em uma página. O *software* do computador usado pelo scanner OMR é desenvolvido para reconhecer o significado das diferentes marcas e para converter as imagens digitalizadas em dados legíveis apresentando as respostas do cartão resposta (VERDE, 2000).

Verde (2000) ainda cita que está tecnologia tem sido amplamente utilizada desde a década de 70 para uma variedade de usos, incluindo provas escolares, censos, pesquisas e loterias, bem como em votação em outros países. Ele também é usado como leitor de código de barras que é facilmente encontrado na maioria dos produtos de varejo.

2.2.6 SIFT

O SIFT (*Scale Invariant Feature Transform*) é um algoritmo que encontra pontos fundamentais na imagem, chamado de *Keypoints*, atribuindo aos mesmos uma assinatura característica (LOPES; SIMÃO, 2011).

Segundo Teixeira (2011), o SIFT é utilizado para detectar e descrever características locais das imagens. David Lowe em 1999 propôs este algoritmo e desde então o SIFT se tornou amplamente utilizado pelo fato de apresentar vários resultados positivos em vários tipos de processamentos de imagens.

O algoritmo é proposto com a intenção de resolver o problema de detectar imagens dentro de imagens quando elas foram sujeitas a rotações, translações e mudanças de escala (LOPES; SIMÃO, 2011) como são exemplificados na Figura 11.

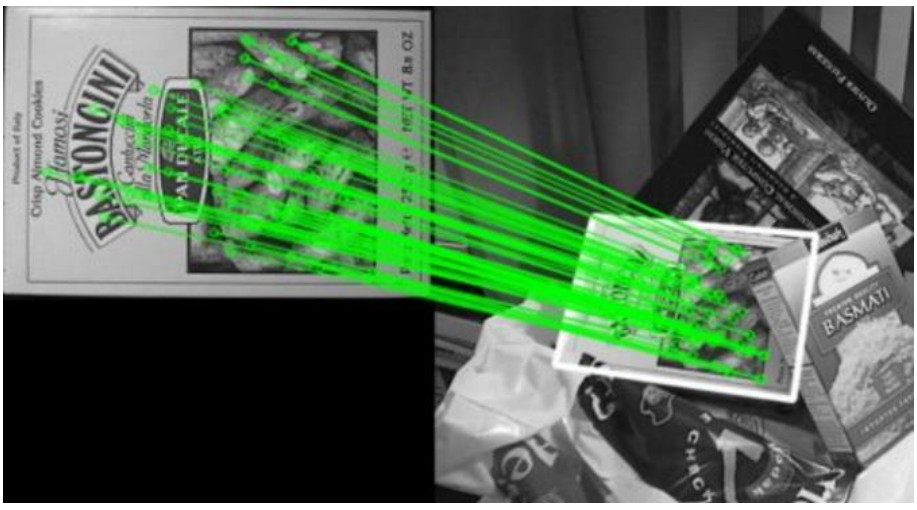
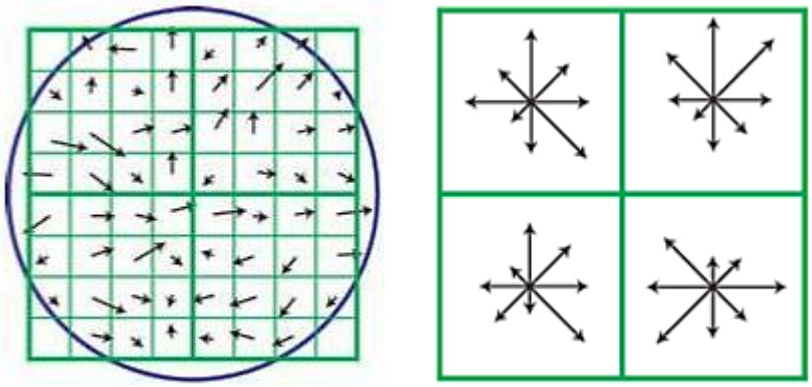


Figura 11 - Exemplo do algoritmo SIFT.
Fonte: Lowe (2004).

O algoritmo SIFT é caracterizado por duas fases, a detecção dos pontos de interesses a extração do descritor visual. O objetivo é encontrar pontos de interesse que representam locais

relevantes da imagem e que se mantenham estáveis em relação às mudanças de escala e rotação.

A função gaussiana é aplicada na região em volta do ponto chave para dar peso à magnitude do gradiente de cada ponto na vizinhança para evitar mudanças na posição e dar menos ênfase a gradientes que estão longe do ponto chave (LOPES; SIMÃO, 2011). Na Figura 12 é apresentada a função gaussiana como um circulo em volta da região.



(a) Imagem gradiente (b) Descritor do algoritmo

Figura 12 - Exemplo do algoritmo SIFT em uma região de 8x8, (a) Imagem gradiente e (b) Descritor do algoritmo.
Fonte: Lowe (2004).

No primeiro passo é obtido o gradiente de cada ponto, em uma região de 8x8, como é apresentado na Figura 12(a), em seguida de cada bloco 4x4 é calculado o histograma com oito direções do gradiente. Cada uma destas regiões é representada por 128 valores (valores obtidos através de 16 regiões de 4x4 multiplicado por oito pontos do histograma). O descritor SIFT, devido às propriedades de cada ponto, é caracterizado por ser invariante a iluminação, rotação e escala. Este descritor apresenta um grau de distinção que permite que funcione como identificador da imagem em questão (BALLESTA et al., 2007; LOWE, 2004).

A partir da obtenção destes dados o algoritmo constrói então os vetores que objetivam descrever a imagem por meio das orientações mais comuns ao redor deles (LOPES; SIMÃO, 2011).

2.2.7 SURF

O SURF (*Speeded Up Robust Features*) é um descritor invariante a rotação e aos efeitos de escala apresentado por Bay, Tuytelaars e Gool no ano de 2008. Este algoritmo foi desenvolvido no ano de 2006 por Hebert Bay, Tinne Tuytelaars e Luc Van Gool, que como o nome sugere, é uma versão acelerada do algoritmo SIFT (BAY; TUYTELAARS; GOOL, 2008).

Nesta técnica, nem o detector de pontos, nem o descritor utilizam dados relacionados com a cor. Na detecção de pontos é utilizado o método de imagens integrais, que efetua o somatório de valores dentro de uma área retangular, reduzindo o tempo de computação (BAY; TUYTELAARS; GOOL, 2008).

Para fazer a detecção de pontos de interesse, diferentemente do SIFT que usa diferenças Gaussianas, o SURF realiza este processo por meio de filtros de Haar que tem formato de caixa e se baseia no determinante da matriz Hessiana computando as derivadas de segunda ordem da Gaussiana, utilizando integrais de imagens que permitem a rápida computação dos filtros, como é apresentada na Figura 13 em que para obter a localização e escala do ponto chave, as localizações são interpoladas procurando descartar os pontos de baixo contraste similar de como é feito no algoritmo SIFT (BAY; TUYTELAARS; GOOL, 2008).

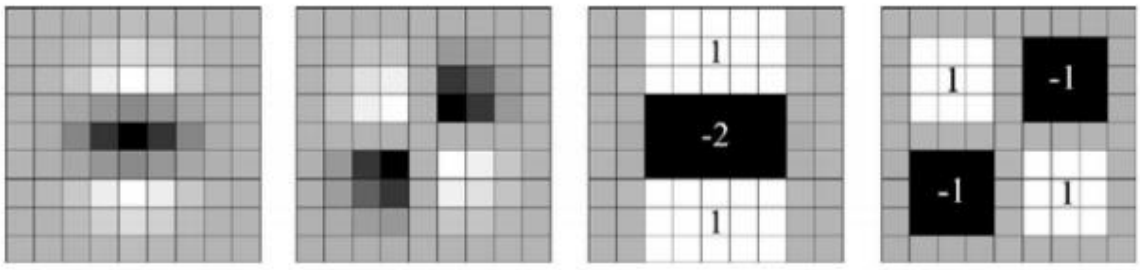


Figura 13 - Derivadas Gaussianas e Filtro Caixa.
Fonte: Bay, Tuytelaars e Gool (2008).

O descritor é computado com base na construção de um histograma de orientação dos gradientes dos pontos presentes nas 16 regiões ao redor do ponto de interesse. Para cada sub-região é analisada a soma das respostas dos filtros e o modulo das respostas nas direções verticais e horizontais como apresentado na Figura 14 resultando em um vetor de 64 dimensões (BAY; TUYTELAARS; GOOL, 2008).

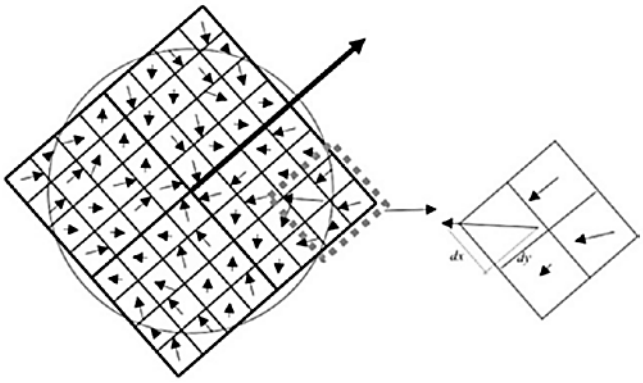


Figura 14 - Descritor SURF.
Fonte: Bay, Tuytelaars e Gool (2008).

O algoritmo SURF define uma escala para cada característica detectada, este fator de escala pode ser usado para definir o tamanho da área em torno do ponto, de tal modo que as informações das vizinhanças irão ser úteis para caracterizar os pontos e assim torna-los distinguível dos outros.

O maior avanço do detector SURF se tem pela velocidade que o mesmo tem em detecção permitindo a utilização até mesmo em tempo real (BAY; TUYTELAARS; GOOL, 2008).

2.2.8 *Template Matching*

Segundo Brunelli (2009) "*Template Matching* é uma técnica em processamento de imagens em que sua principal função é encontrar pequenas partes de uma imagem que correspondem a uma imagem modelo". Basicamente esta técnica consiste em criar um *template* para cada classe do problema em questão e depois comparar o exemplo de teste com todos os *templates* disponíveis. Aquele *template* que tiver a menor distância será a classe escolhida.

Este método é muito utilizado em aplicações atuais, principalmente na área de reconhecimento de face e processamento de imagens médicas. No estudo apresentado por Tahmasebi (2012), é relatado o método implementado em simulação estatística que poderia fornecer um algoritmo rápido de reconhecimento de imagens médicas.

Uma maneira de se fazer a combinação entre as imagens, é percorrer pixel a pixel no *template* e verificar em qual ponto da imagem padrão está localizada a imagem do modelo.

Porém pode se tornar lenta dependendo do tamanho das imagens processadas, pois ela lê e compara pixel a pixel (ALMEIDA, 2007), como é exemplificada na Figura 15.

Figura 15 - Exemplo do algoritmo *Template Matching*.
Fonte: OpenCV (2015).

Ao percorrer a imagem de *template* sobre a imagem original a métrica é calculada para representar sua comparação de valores de pixels e armazenada em uma matriz de resultados, cada local *f (x , y)* contém uma métrica (OPENCV, 2015).

A partir da Figura 15 o resultado desta métrica é apresentado na Figura 16 em que ao deslizar a imagem modelo sobre a imagem original, o algoritmo calcula os locais mais brilhantes, que indicam as correspondências mais altas da imagem apresentando o local marcado pelo circulo vermelho é o local em que existe o maior valor, assim o retângulo é formato a partir das dimensões da imagem de *template* (OPENCV, 2015).

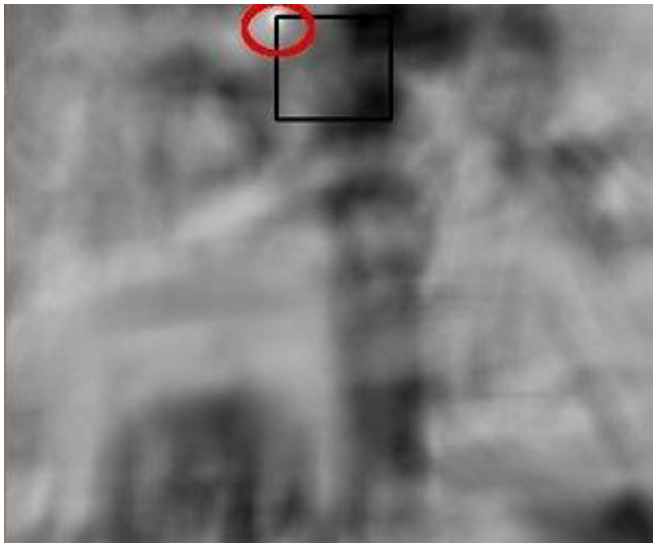


Figura 16 - Métrica de avaliação *Template Matching*.
Fonte: OpenCV (2015).

O *Template Matching* é uma técnica que ao mesmo tempo, segmenta e classifica os objetos de uma determinada classe. O sucesso desta técnica pode ser comprometido devido a uma aquisição de imagem de baixa qualidade, com ruídos que aumentam a variabilidade dos objetos presentes na imagem (ALMEIDA, 2007).

3 METODOLOGIA

Este capítulo apresenta a parte organizacional do projeto e descrição dos métodos utilizados. Para o contexto deste trabalho serão tratadas com exclusividade as etapas do processamento de imagens, ou seja, o foco principal está em aplicar as técnicas do PDI e analisar qual se manifesta melhor no reconhecimento de cartões resposta.

3.1 AMBIENTE DE DESENVOLVIMENTO

Foi escolhida a linguagem de programação C++ para o desenvolvimento do trabalho, utilizando a plataforma de programação Code::Blocks, com a instalação da biblioteca OpenCV.

O *hardware* utilizado no desenvolvimento e experimentos dos códigos foi um notebook com processador Intel Core I3 e 4GB de memória RAM, e um scanner convencional para a aquisição das imagens, com resoluções de 2477 x 3504 pixels.

3.1.1 Linguagem C++

O C++ é uma linguagem de programação de nível médio, que foi desenvolvida em base na linguagem de programação C. O desenvolvimento desta linguagem se deu por volta da década de 80 por Bjarne Stroustrup, em que o objetivo era desenvolver esta nova linguagem para melhorar uma versão do núcleo Unix. No seu início a linguagem usava um pré-processador, mas com novas ideias, Stroustrup desenvolveu um compilador próprio. No desenvolvimento foram acrescentados elementos de outras linguagens na ideia de criar uma linguagem de novos elementos sem trazer problemas para a programação (PACIEVITCH, 2011).

A linguagem C++ foi à única linguagem entre tantas outras que obteve sucesso como sucessora da linguagem C, servindo de inspiração para outras linguagens como Java e C# (PORTO; BITAR, 2005).

3.1.2 Code Blocks

O Code::Blocks é um ambiente de desenvolvimento integrado (IDE), é um *software* de domínio público, ou seja que pode ser usado livremente, pode ser obtido no site do fabricante a versão utilizada neste trabalho foi a versão 13.12. Ele possui suporte com múltiplos compiladores como GCC/MinGW, SDCC, Intel C++, GNU ARM entre outros. Sua arquitetura é orientada a plug-ins, em que suas funcionalidades são definidas pelos plug-ins fornecidos a ele. (CODE::BLOCKS, 2015).

3.1.3 OPENCV

OpenCV é uma biblioteca de visão computacional *Open Source* (código aberto) sob a licença BSD (*Berkeley Software Distribution*), ou seja, é gratuito tanto para o uso acadêmico como para o uso comercial. Possui interfaces em C, C++, Python e Java, para ser desenvolvidas em plataformas Linux, Windows, Mac OS X, iOS e Android. Foi desenvolvida para a eficiência computacional e com um forte foco em aplicações em tempo real, podendo tirar proveito dos processadores *multicore* por ter sido otimizado em C/C++ (OPENCV, 2015).

Esta biblioteca foi desenvolvida nas linguagens de programação C/C++, mas também dá suporte a programadores que trabalham com Java, Python e Visual Basic e desejam incorporar a biblioteca em seus aplicativos (BRADSKI; KAEBLER, 2008).

Pelo fato de ser uma ferramenta livre e a facilidade de obter imagens de grande qualidade por um baixo custo está tornando possível o desenvolvimento de sistemas sofisticados com um baixo investimento e custo de operação (DELAI; COELHO, 2011).

3.2 FLUXO DO PROCESSAMENTO DOS CARTÕES RESPOSTA

O fluxo de execução das atividades envolvidas no estudo de reconhecimento de cartões resposta compreende as seguintes etapas (Figura 17): (1) aquisição da imagem definindo uma imagem para modelo e o restante para a execução de testes no trabalho, (2) aplicação de algoritmos de detecção de características, (3) morfologia matemática e o (4) reconhecimento das alternativas assinaladas.

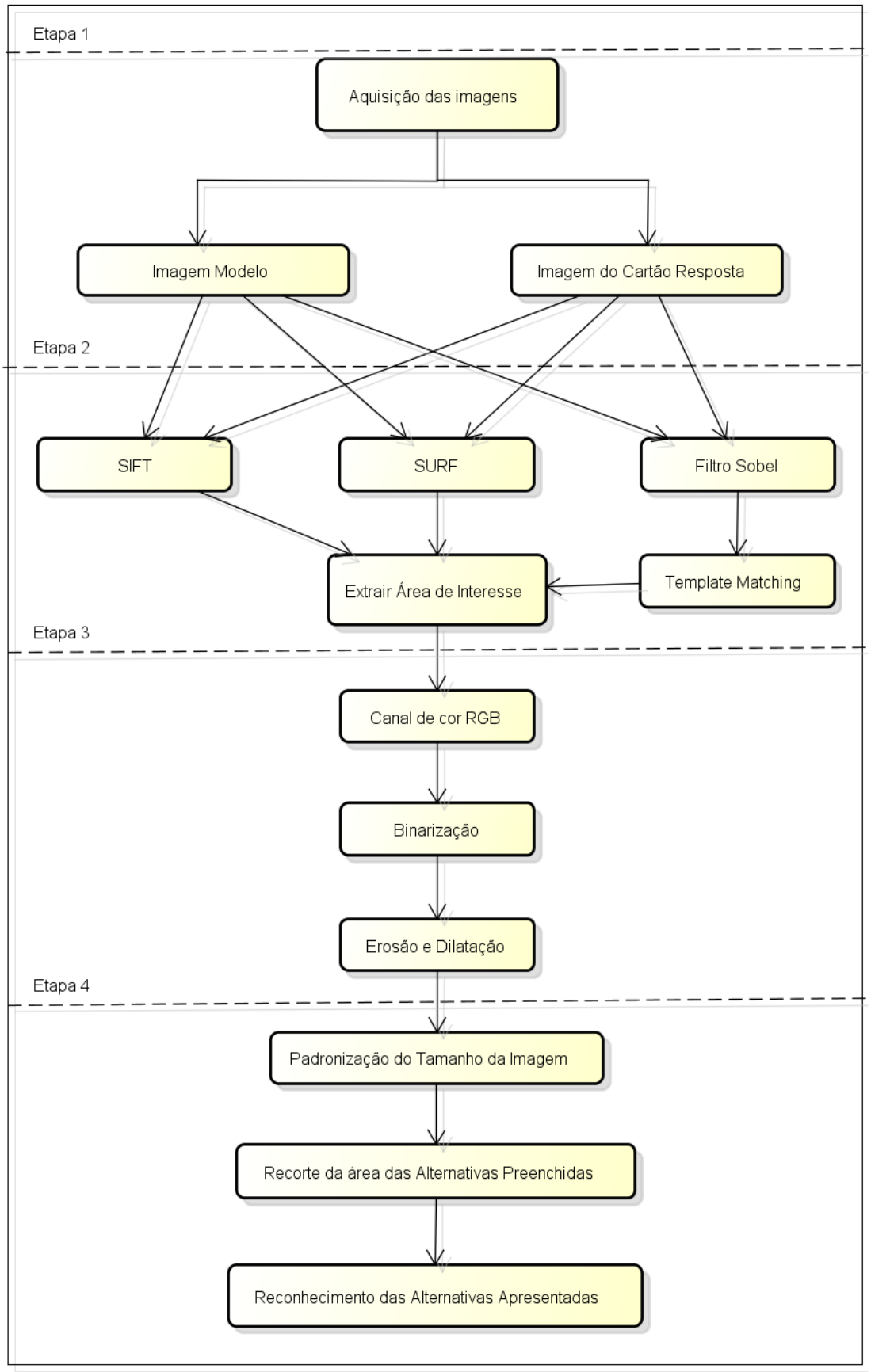


Figura 17 - Sequência de etapas do processamento do cartão resposta.
Fonte: Autoria própria.

3.2.1 Aquisição de imagens

A aquisição das imagens foi realizada por meio de um scanner convencional, com o intuito de digitalizar o cartão resposta, e em seguida convertê-la em uma representação adequada para assim dar início ao processamento da imagem.

Para confecção de um banco de imagens, foram digitalizados 55 cartões respostas com resolução 2477 x 3504 pixels com 30 questões assinaladas.

3.2.2 Modelo do cartão resposta

Visando encontrar um algoritmo capaz de reconhecer a área das respostas em que as mesmas estão assinaladas, são aplicadas três técnicas de reconhecimento a partir de algoritmos da biblioteca OpenCV, sendo os algoritmos SIFT, SURF (detectores e descritores razoavelmente invariante a mudança de iluminação, rotação e ruído) e o algoritmo *Template Matching*.

Os algoritmos SIFT e SURF detectam pontos, também chamados de *Keypoints*, nas duas imagens e os compara calculando sua distância, já o algoritmo *Template Matching* utiliza a técnica de comparar os elementos e suas vizinhanças, em que também necessita de uma imagem exemplo como nos outros algoritmos.

Para isso se concretizar, os algoritmos possuem requisitos e delimitações. Os algoritmos necessitam de uma imagem modelo para utiliza-la em comparação com a imagem, em que o objeto é encontrar a área de interesse no cartão resposta.

3.2.2.1 Algoritmo SIFT

Este algoritmo possui duas funções primordiais para seu funcionamento, sendo que o `SiftFeatureDetector` detecta os pontos chaves nas duas imagens e os armazena em formato de *keypoints*. Na função seguinte o `SiftDescriptorExtractor` extrai dados

adquiridos pela função anterior e os transforma em uma imagem para assim efetuar uma comparação dos pontos da figura de modelo com a imagem do cartão resposta Figura 18.

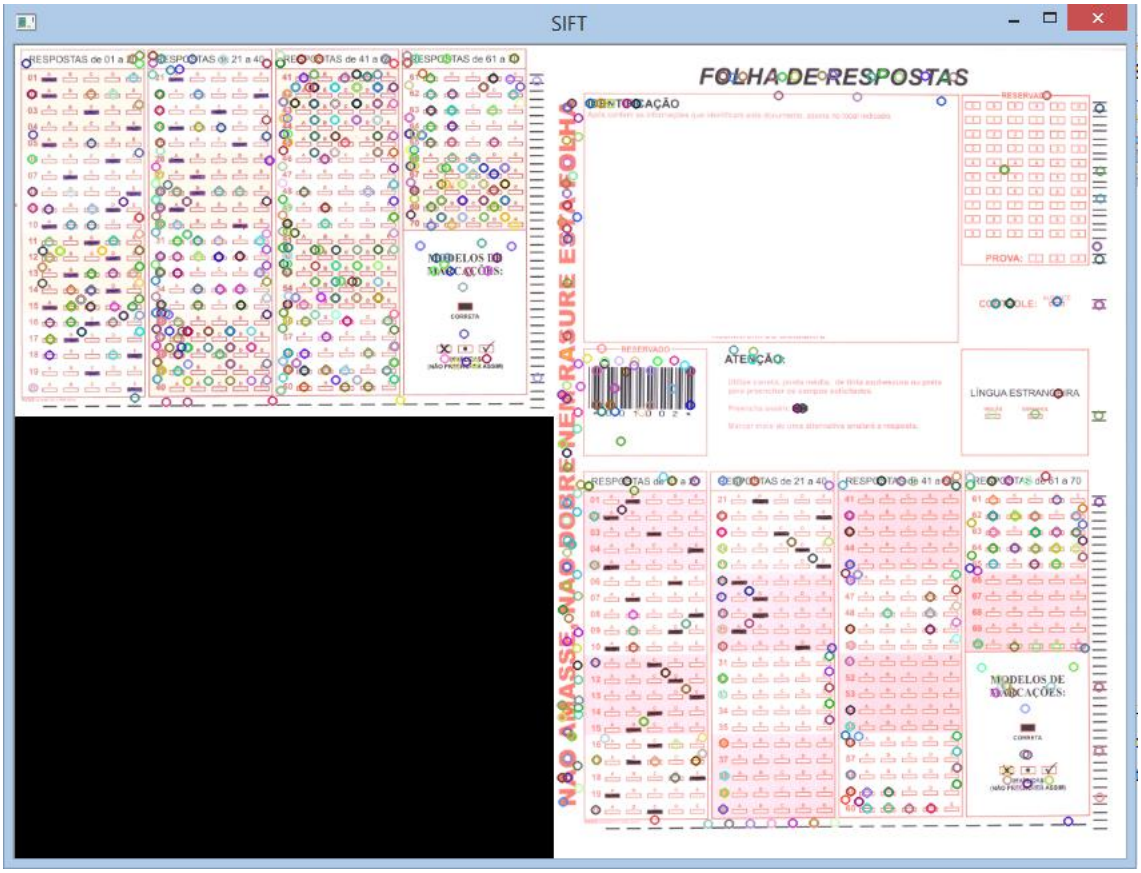


Figura 18 - *Keypoints* extraídos pelo algoritmo SIFT.
Fonte: Autoria própria.

A partir da detecção dos pontos (Figura 18), o algoritmo compara os vários *keypoints* analisando as distâncias dos mesmos, os quais são minerados até só restar os pontos que possuem a maior semelhança entre a imagem exemplar e a imagem do cartão resposta, assim interligando os *keypoints* nas duas imagens como é apresentado na Figura 19.

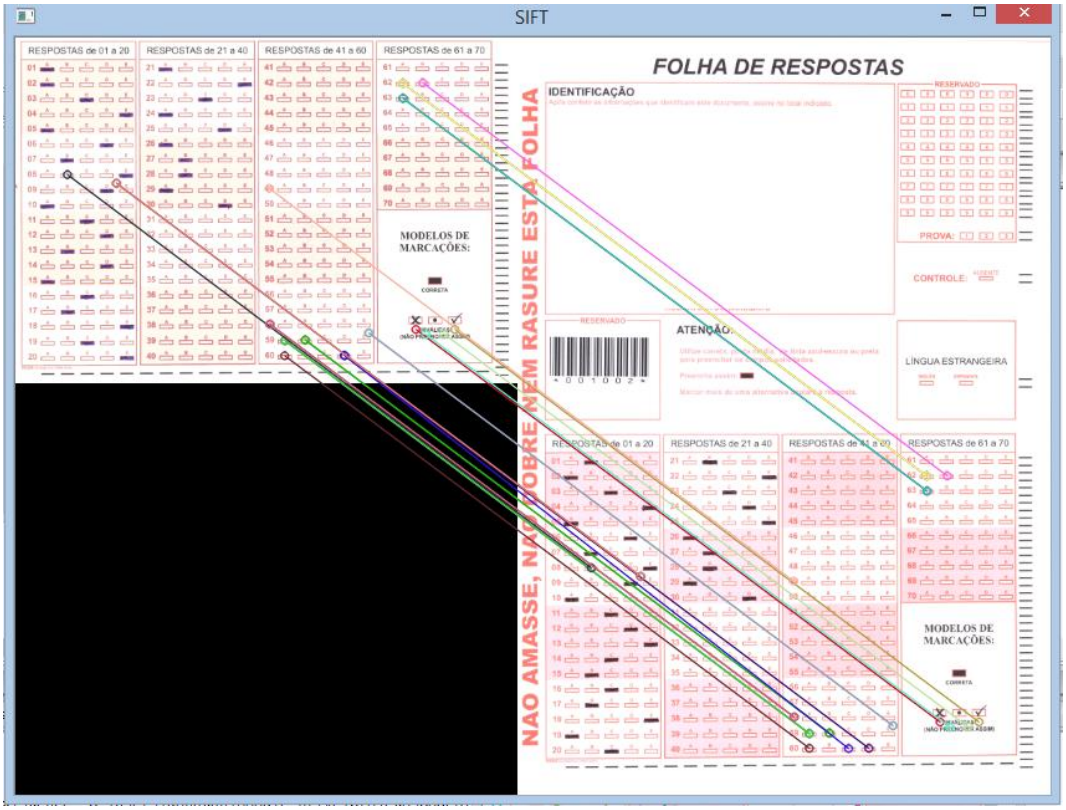


Figura 19 - *Keypoints* Interligados calculados a partir das suas distâncias.
Fonte: Autoria própria.

Após a verificação e interligação dos *keypoints* que possuem as distâncias exigidas pelo programador, o algoritmo localiza os quatro pontos das extremidades do cartão em que possuem uma grande área de *keypoints* e os apresenta a partir de um retângulo.

3.2.2.2 Algoritmo SURF

Como algoritmo SURF é derivado do algoritmo SIFT, seu funcionamento é bastante semelhante, somente alterando sua forma de processamento das funções pois o mesmo também possui as duas funções primordiais, sendo que o `SurfFeatureDetector` detecta os pontos chaves nas duas imagens e os armazena em formato de *keypoints*. Na função seguinte o `SurfDescriptorExtractor` extrai dados adquiridos pela função anterior e os transforma em uma imagem, para assim efetuar uma comparação dos pontos da figura de modelo com a imagem do cartão resposta.

Assim como na extração dos pontos no algoritmo SIFT (Figura 18), o SIFT obtém os pontos detectados em que o algoritmo compara os vários *keypoints*, minera os pontos e interliga os mesmos nas duas imagens (Figura 19).

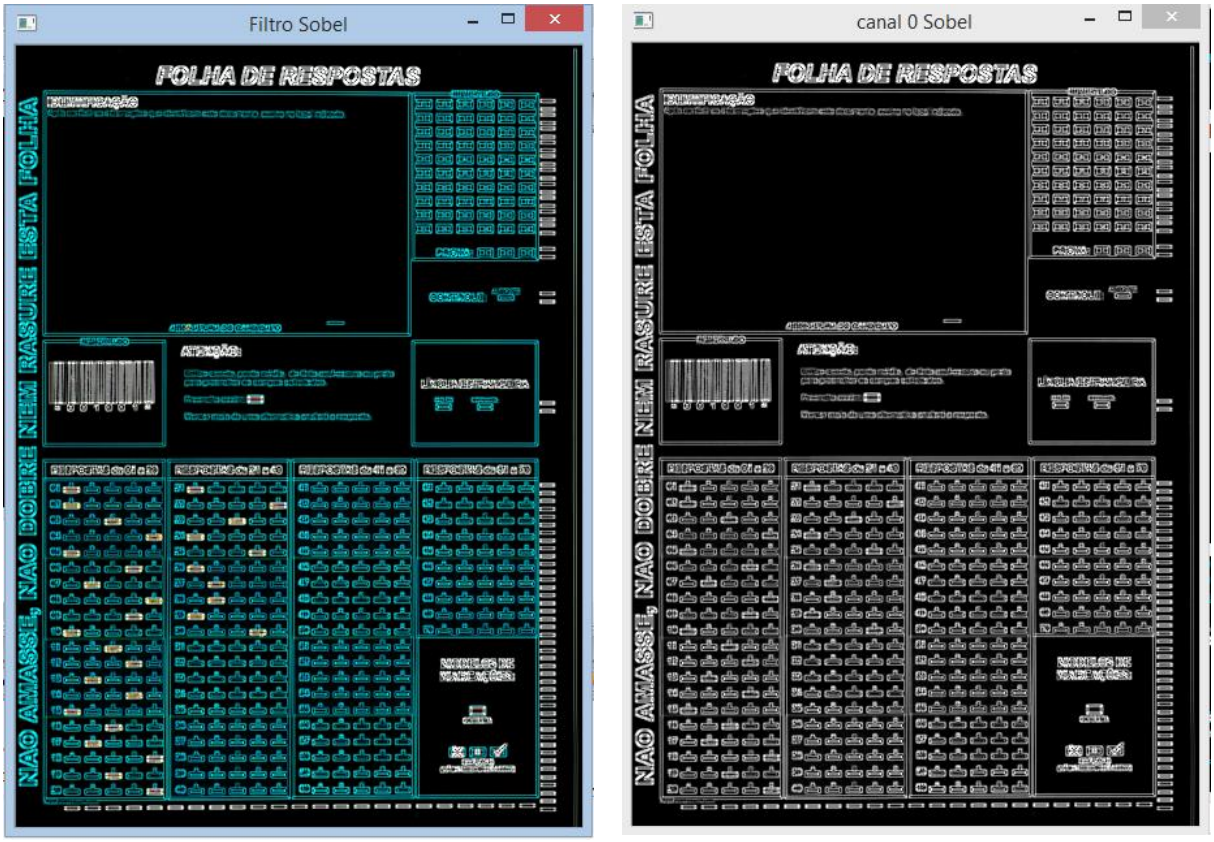
Após a detecção e a extração dos *keypoints* o algoritmo SURF localiza os quatro pontos das extremidades do cartão em que possuem uma grande área de *keypoints* e os apresenta a partir de um retângulo.

3.2.2.3 Algoritmo *Template Matching*

Como as imagens adquiridas não são padronizadas em tamanho e/ou coloração, algumas técnicas de pré-processamento foram aplicadas para assim preparar as imagens para um melhor reconhecimento do algoritmo *Template Matching*.

O operador Sobel foi utilizado para esta melhoria, fazendo com que os cartões ficassem mais padronizados no formato de sua coloração. Sabendo que o algoritmo *Template Matching* possui uma forma de localização de objeto a partir dos pontos dos pixels e suas vizinhanças, como os cartões respostas não possuem pontos exatamente iguais a técnica do operador Sobel tende ser aplicada.

Na Figura 20(a) é ilustrado o cartão resposta com a aplicação do filtro Sobel. Para aperfeiçoar mais ainda o reconhecimento das alternativas assinaladas às imagens que estão em formato RGB são separados seus canais de cores (Figura 20(b)) assim, otimizando o algoritmo e alcançando melhores índices de reconhecimento.



(a) Imagem com aplicação do filtro Sobel (b) Canal *red* da imagem após aplicação do filtro Sobel

Figura 20 - Pré-processamento para aplicação do algoritmo Template Matching, que se tem (a) Imagem com aplicação do filtro Sobel e (b) Canal *red* da imagem após aplicação do filtro Sobel.
Fonte: Autoria própria.

A partir deste processamento a imagem do cartão resposta está apta para ser processada pelo algoritmo *Template Matching*.

O algoritmo assim realiza a operação de verificação dos modelos, percorrendo pixel a pixel na imagem e armazenado o resultado da métrica, que após é normalizada para assim detectar os pontos mais brilhantes da imagem. Utilizando a função `minMaxLoc` para a localização dos valores mínimos e máximos, que indicam as correspondências mais altas da imagem, assim apresentando um retângulo na imagem em que os pontos são encontrados em maior escala.

3.2.3 Extração da área de interesse

Um conceito importante em processamento de imagem é a região de interesse (ROI - *Region Of Interest*). Entende-se como sendo a região definida pelo operador ou automaticamente a partir de parâmetros.

A partir da localização apresentada pelos algoritmos, o algoritmo apresenta os pontos extremos da área. Com os pontos *f (x,y)* da imagem derivadas pelas variáveis `scene_corners[0]` e `scene_corners[2]` a técnica ROI é executada como é apresentada no Quadro 1.

```
Formato = original (Rect (scene_corners[0], scene_corners[2]));
```

Quadro 1 - Código fonte para extração da área de interesse.
Fonte: Autoria própria.

A partir da imagem original é extraída a área de interesse no cartão resposta em que a variável `Formato` recebe apenas a área recortada onde estão apresentadas as alternativas assinaladas.

3.2.4 Escolha do canal de cor

Com o objetivo de destacar as respostas assinaladas em cada uma das questões, realizou-se a divisão de canais de cores RGB da imagem que representa a área de interesse do cartão.

Para realizar a divisão dos canais, utilizou-se a função `split` do OpenCV que a partir de uma imagem de 24 bits (imagem colorida), faz a divisão em três canais de 8 bits, que representam cada um dos canais separadamente.

A partir desta divisão elaborou-se um estudo com as três imagens adquiridas, valorizando o canal R, em que o mesmo apresenta um realce melhor das alternativas assinaladas.

3.2.4.1 Binarização (*threshold*)

Para exercer a binarização da imagem, foi aplicada a função de *threshold*, com o objetivo de destacar e eliminar pontos não pertinentes às respostas preenchidas pelo candidato.

Para isto é necessário um pré-processamento da imagem transformando-a em tons de cinza, após é aplicada a técnica de *threshold*, que verifica a intensidade do pixel calculado baseado no valor de corte, sendo que se o mesmo possuir valor abaixo do ponto de corte sua cor será alterada para a cor preta caso contrário o pixel se tornará de intensidade máxima (valor 255), ou seja, pixel de cor branca, tornando um ótimo destaque para as respostas assinaladas às tornando bem visíveis, sendo assim de fácil reconhecimento para o algoritmo.

O processo foi aplicado sobre a imagem gerada a partir do processo de separação das bandas em RGB, selecionando-se o canal R que apresentou o maior destaque para as respostas preenchidas.

3.2.4.2 Erosão da imagem

A erosão da imagem se dá por conta de ampliar os pixels em que estão sinalizadas as respostas preenchidas no cartão resposta, para assim o sistema encontrar mais facilmente o local da resposta tornando o algoritmo menos vulnerável a erros de correções.

O processo foi aplicado em uma imagem binária, em que a função `erode` do OpenCV amplia os pixels de valores 0 (pixels de tonalidade de cor preta) sobre os outros pixels vizinhos, dando o destaque para as respostas assinaladas.

3.2.5 Padronização do tamanho

No estudo foram trabalhados vários tamanhos de imagem (resoluções de captura), consequentemente erros foram encontrados em que as respostas não eram encontradas com as

devidas proporções, sendo nestas condições o tamanho das imagens deveriam de ser padronizadas, tornando-as assim menos suscetíves a erros de uma má localização do eixo da matriz.

A partir da função `resize` uma imagem pode ser redimensionada e tornando-a padrão informando a sua resolução, deve-se destacar que em muitos processamentos esta técnica não é atribuída pelo fato de se perder e/ou alterar dados das imagens.

3.2.6 Recorte da área das alternativas preenchidas

Com o objetivo de padronizar e alinhar a área extraída do cartão resposta, as extremidades da imagem foram detectadas, seguindo a partir dos marcos do gabarito para assim eliminar os ruídos e pontos que não são importantes para o reconhecimento das respostas, sendo assim o algoritmo localiza os últimos pixels de intensidade de cor preta nos cantos inferior esquerdo e superior direito da imagem, e a partir destas localizações são recortadas às extremidades da imagem, tornando-as com o mesmo formato e tamanho.

3.2.7 Reconhecimento das alternativas apresentadas

O reconhecimento das alternativas se dá pelos pontos indicados pelo candidato no cartão resposta. O sistema elabora uma matriz de pontos em que a partir desta matriz as alternativas podem ser localizadas por meio da sua tonalidade de cor, ou seja, os pontos em que o pixel tem valor equivalente à zero (cor preta) significa ser parte do campo preenchido da resposta, consequentemente é gravado no sistema o valor da alternativa.

Seguindo estes conceitos ao percorrer está matriz o sistema verifica o valor da coluna e valor da linha, assim é a detecção de qual questão e qual alternativa preenchida. Após esta verificação o número da questão e a alternativa preenchida é armazenada em uma variável, para após ser gravada em arquivo TXT.

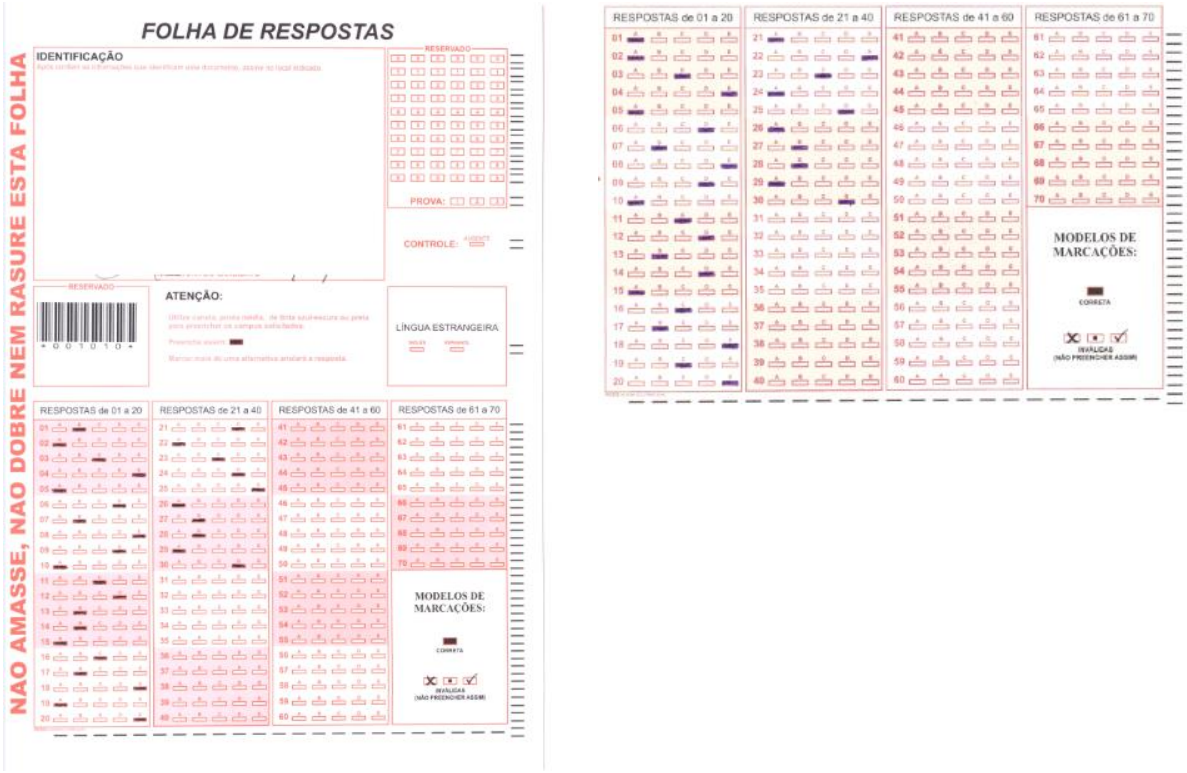
4 RESULTADOS E DISCUSSÃO

Neste capítulo são apresentados os resultados e discussões do trabalho de reconhecimento de cartões respostas, o mesmo é seguido por todos os passos necessários até apresentar as respostas preenchidas pelo candidato no cartão.

Apresentando também os índices de reconhecimento e tempo dos algoritmos, e comparativo dos três algoritmos estudados: SIFT, SURF e *Template Matching*.

4.1 PROCESSO DE RECONHECIMENTO DOS CARTÕES RESPOSTA

A detecção das respostas em cartões respostas de formatos OMR (Figura 21(a)) se dá a partir das aquisições das imagens e, na sequência, a preparação da imagem que serviu como modelo (Figura 21(b)), necessário para a execução dos algoritmos SIFT, SURF e *Template Matching*, visando comparar qual algoritmo possui o melhor índice de reconhecimento da área de interesse no cartão resposta.



(a), imagem original de um cartão resposta

(b) Imagem modelo preparado para utilização nos algoritmos SIFT, SURF e *Template Matching*

Figura 21 - Aquisição de cartões resposta para o processamento das imagens, (a) imagem original de um cartão resposta e (b) Imagem modelo preparado para utilização nos algoritmos SIFT, SURF e *Template Matching*.
Fonte: Autoria própria.

Na Figura 22 são apresentadas os resultados dos algoritmos SIFT, em que se localiza a área de interesse e forma um retângulo a partir da área detectada.

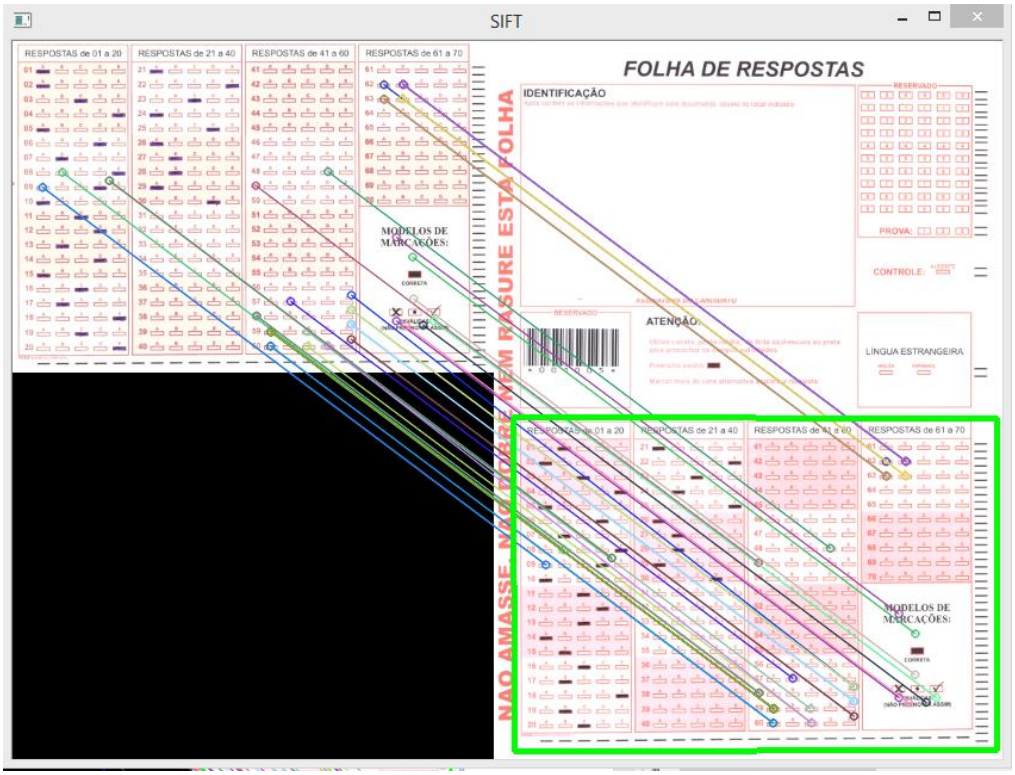


Figura 22 - Resultado do reconhecimento da área de interesse do algoritmo SIFT.
Fonte: Autoria própria.

Na Figura 23 apresenta-se o resultado do reconhecimento da área de interesse do algoritmo SURF, exibido pelas delimitações do retângulo de cor verde.

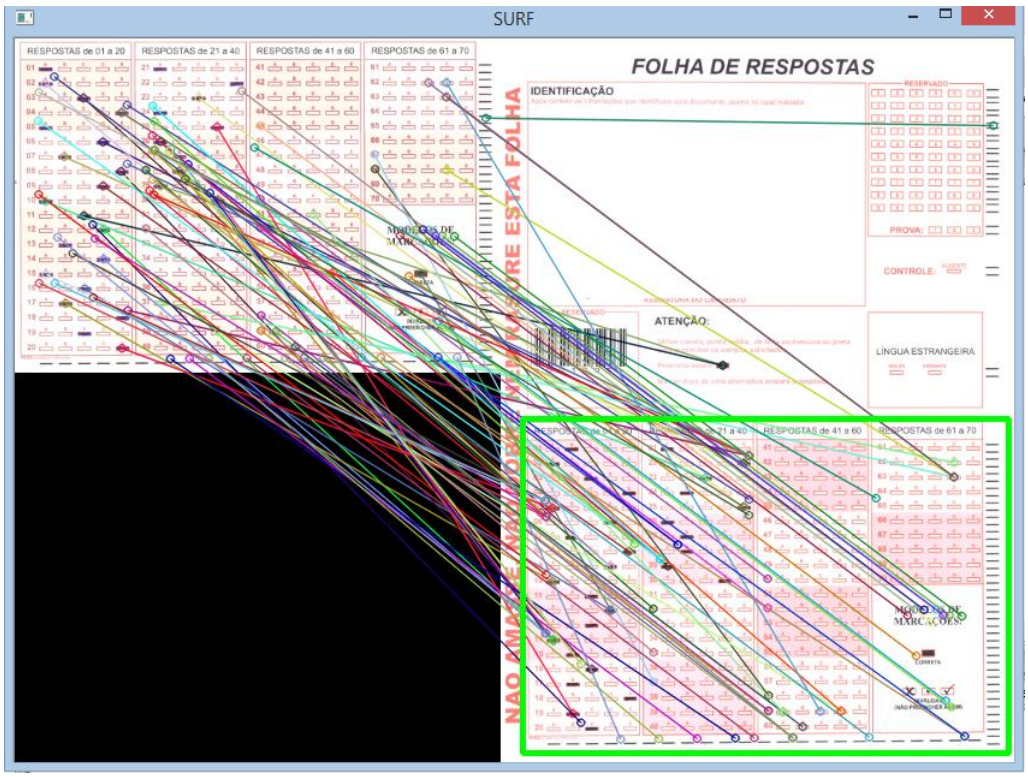


Figura 23 - Resultado do reconhecimento da área de interesse algoritmo SURF.
Fonte: Autoria própria.

Na Figura 24 é apresentado o resultado da detecção da área de interesse no cartão resposta a partir do algoritmo *Template Matching*.

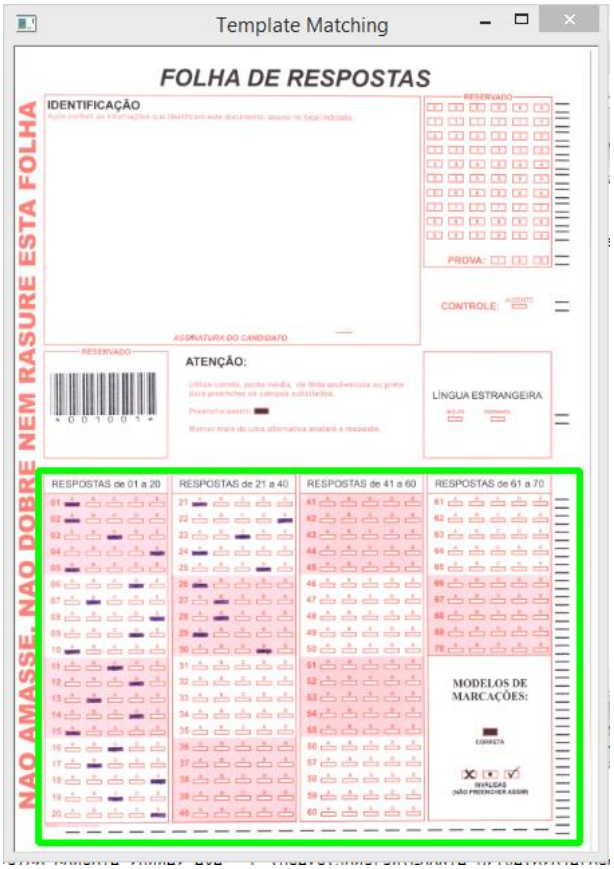


Figura 24 - Área de interesse detectado pelo algoritmo *Template Matching*.
Fonte: Autoria própria.

A partir do reconhecimento da área de interesse reconhecida pelos algoritmos SIFT, SURF ou *Template Matching*, a técnica de ROI (*region of interest*) foi aplicada para extrair a área de interesse (Figura 25), que, no caso dos cartões respostas, é o local em que estão localizadas as respostas assinaladas pelo candidato.

Figura 25 - Resultado da extração da área de interesse do cartão resposta.
Fonte: Autoria própria.

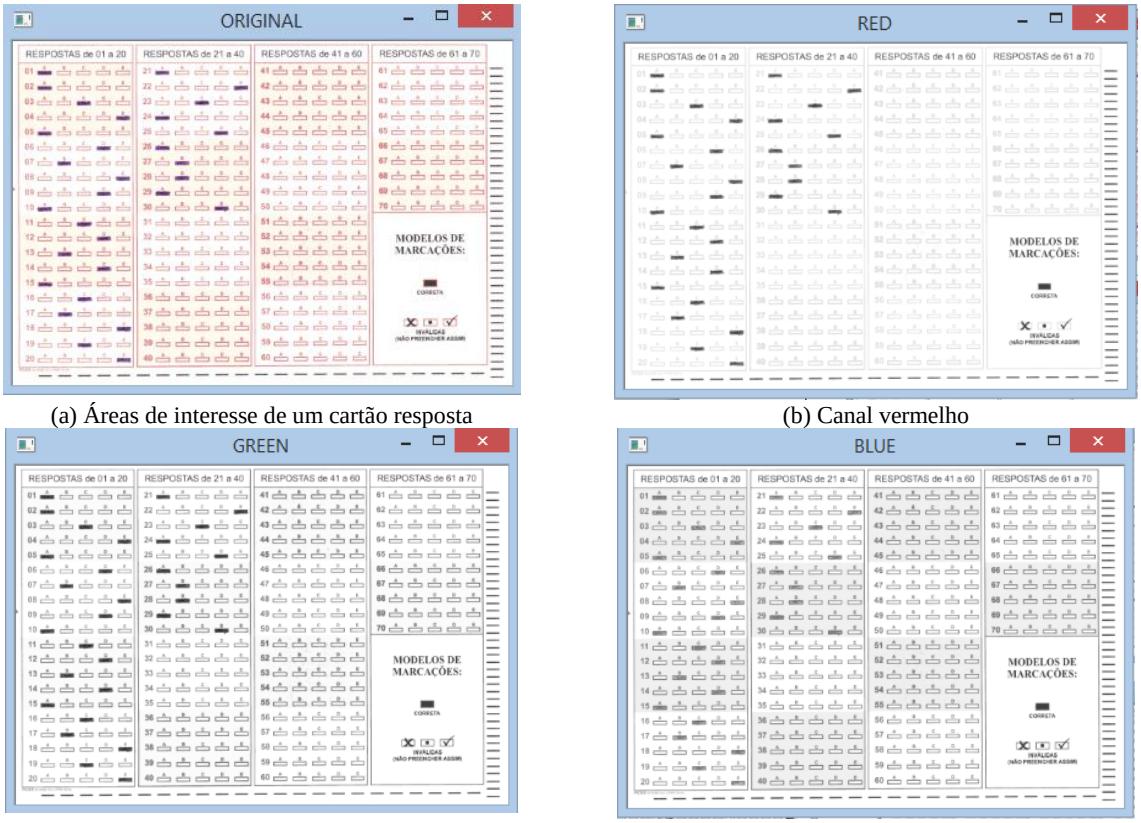
Com a extração da área de interesse dos cartões respostas se dá continuidade a partir da escolha dos canais de cores,

No Quadro 2 é apresentado o código para realizar este processo, em que é declarada uma variável do tipo `<Mat>` em um vetor de três posições, no qual são armazenados os três canais de cor. E em seguida apresentada em novas janelas os três canais RGB.

```
Vector <Mat> canais(3);
Split(Formato, canais);
```

Quadro 2 - Código fonte para divisão de canais de cores.
Fonte: Autoria própria.

Na Figura 26 são apresentadas as imagens com a separação dos canais de cores R (Figura 26 (b)), G (Figura 26 (c)) e B (Figura 26 (d)).



(a) Áreas de interesse de um cartão resposta

(b) Canal vermelho

(c) Canal verde

(d) Canal azul

Figura 26 - Divisão dos canais de cores a partir da área de interesse do cartão resposta, (a) Áreas de interesse de um cartão resposta, (b) Canal vermelho, (c) Canal verde e (d) Canal azul.
Fonte: Autoria própria.

Dentre os canais o canal vermelho (Figura 26 (b)) apresentou maior destaque para as partes preenchidas no cartão, que correspondem à reposta que o candidato assinalou em cada questão. Consequentemente este canal de cor foi escolhido para dar continuidade ao projeto.

Seguindo a partir da imagem selecionada, apenas o canal vermelho (Figura 26 (b)), aplicou-se o *threshold* (binarização) (Figura 27), com o objetivo de fazer a separação entre o fundo da imagem, dos pixels de interesse da imagem (alternativas marcadas pelo candidato). Assim possibilita analisar com maior facilidade as áreas que possuem as respostas assinaladas.

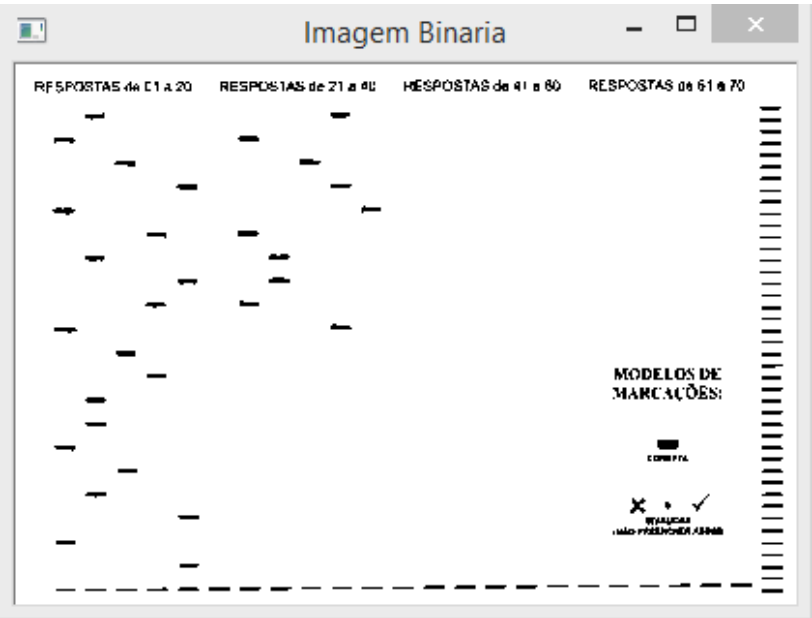


Figura 27 - Cartão Resposta em formato binário.
Fonte: Autoria própria.

A partir da imagem binarizada apresentada na Figura 27, o próximo passo foi efetuar a função `erode` responsável por fazer a erosão da imagem, expandindo os pixels que correspondem às respostas assinaladas pelo candidato. O código implementado para esta funcionalidade, está apresentado no Quadro 3, e contém os valores de *kernel* para executar a erosão da imagem, ajustado o ponto de ancoragem com valor quatro, assim valorizando os campos preenchidos pelo candidato.

```
Mat kernel = Mat::ones(Size(4, 4), CV_8U);

Mat img_binary_erode = img.clone();
erode(img, img_binary_erode, kernel);
```

Quadro 3- Trecho de código fonte, com a função de erosão na imagem.
Fonte: Autoria própria.

O resultado da erosão é apresentado na Figura 28.

Figura 28 - Imagem aplicada erosão em um *kernel* de tamanho 4x4.
Fonte: Autoria própria.

Devido as diferentes resoluções de imagens, surge a necessidade de padronizá-las quanto ao tamanho. Neste sentido, foi aplicada a função para redimensionar a imagem. Na Figura 29 apresenta-se o resultado da imagem com resolução de 1110 x 635 pixels.

Figura 29 - Imagem configurada com resolução padrão de 1110 x 635 pixels.
Fonte: Autoria própria.

Após a padronização do tamanho da imagem, foi necessário recortar e extrair as áreas da imagem que apresentam apenas os resultados das questões assinaladas pelo candidato. Para isso, a imagem foi percorrida a fim localizar os extremos do quadrante (Figura 30 (a)). Com os quatro cantos detectados, a função `rect` (ROI) foi aplicada (Figura 30 (b)).

(a) Imagem de tamanho padrão (b) Imagem recortada

Figura 30 - Recorte da área das alternativas preenchidas, (a) Imagem de tamanho padrão e (b) Imagem recortada.
Fonte: Autoria própria.

A partir da imagem recortada (Figura 30 (b)) eliminando dados que não fazem parte da área de interesse em que estão localizadas as alternativas preenchidas pelo candidato. Seguindo esta imagem o algoritmo realiza uma varredura a partir de um laço de repetição por pontos sinalizado pela Figura 31.

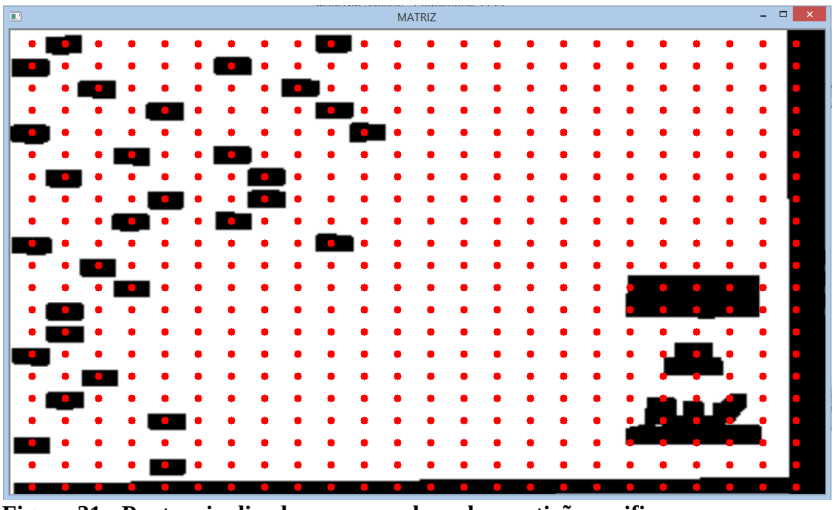


Figura 31 - Pontos sinalizados em que o laço de repetição verifica.
Fonte: Autoria própria.

O algoritmo verifica a intensidade do pixel de localização *f (x, y)* (Quadro 4) se o mesmo for de valor = 0 (cor preto) significa que a localização da matriz está preenchida, caso for de valor = 255 (cor branco) a localização não está preenchida.

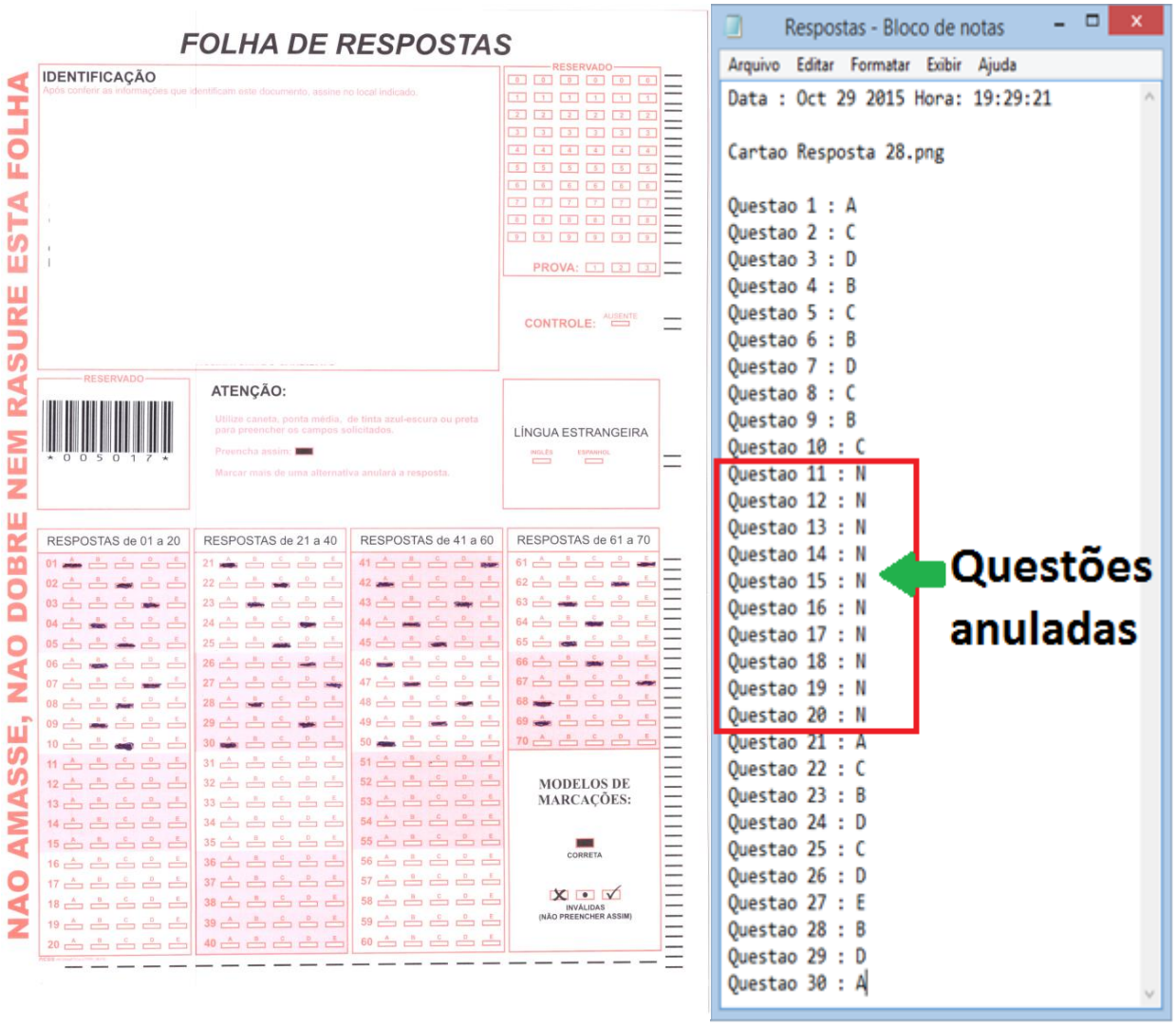
```
if ( imagem.at<uchar>( x , y ) == 0 )
```

Quadro 4 - Código fonte de verificação do valor da intensidade do pixel de posição (x, y).
Fonte: Autoria própria.

Após a verificação dos valores do pixel, o algoritmo armazena em um vetor de 30 posições o número da questão e a alternativa assinalada, sendo A, B, C, D, E ou N para questões anuladas. As questões em que o candidato deixar a alternativa em branco ou preencher duas alternativas em uma só questão, são anuladas apresentando o valor N.

Seguindo está avaliação o algoritmo armazena os valores de todos os cartões respostas em um arquivo de formato TXT.

Na Figura 32(a) é apresentado um cartão resposta preenchido de forma incorreta, em que o candidato, preenchendo as questões 1 a 10, 21 a 30, 41 a 50 e 61 a 69, sendo que as questões que deveriam ser preenchidas seriam as questões 1 a 30 para encaixar com a prova aplicada com isso o sistema desenvolvido para estes testes, possui o intuito de corrigir somente as 30 primeiras questões, sendo assim as questões 11 a 20 devem ser anuladas como é apresentando no arquivo TXT da Figura 32 (b).



(a) Cartão Resposta com as alternativas assinaladas inadequadamente

(b) Resultado apresentado pelo sistema em um arquivo TXT

Figura 32 - (a) Cartão Resposta com as alternativas assinaladas inadequadamente e o (b) resultado do cartão apresentado em um arquivo TXT, contendo dados de hora e data, bem como, nome do arquivo e suas respectivas respostas.
Fonte: Autoria própria.

4.1 ÍNDICE DE DESEMPENHO DOS ALGORITMOS DE DETECÇÃO

Neste capítulo são apresentados os índices de desempenho dos algoritmos de detecção SIFT, SURF e *Template Matching* com nove tamanhos de imagens realizados para cada algoritmo, apresentando três resoluções para imagem modelo e três dimensões para a imagem original, imagem em que será processada a fim de encontrar a área de interesse.

4.1.1 SIFT

Na Tabela 1 são apresentados os índices de acertos da área de interesse, a partir do algoritmo SIFT, em que foram efetuados nove tipos de testes, sendo com três tamanhos de imagem original e três tamanhos de imagem modelo, nas 55 imagens adquiridas para nosso estudo.

Tabela 1 - Índice de acertos do algoritmo SIFT

	Imagem modelo			
Imagem original	**2277 x 1585 px**	**667 x 460 px**	**392 x 273 px**	**Média**
2477 x 3504 px	74,54 %	100,00 %	89,09 %	
723 x 1024 px	98,18 %	100,00 %	74,54 %	78,58 %
420 x 600 px	52,72 %	63,63 %	54,54 %	

Fonte: Autoria própria.

Notas:
px valor da resolução da imagem.

Nos testes apresentados o algoritmo SIFT obteve índices de acertos, razoavelmente bons, porém o mesmo ainda possui erros do reconhecimento, estas falhas de reconhecimentos se dão pela falta de informação que o algoritmo necessita para a localização da área de interesse, e muitas vezes apresentam resultados como a Figura 33.

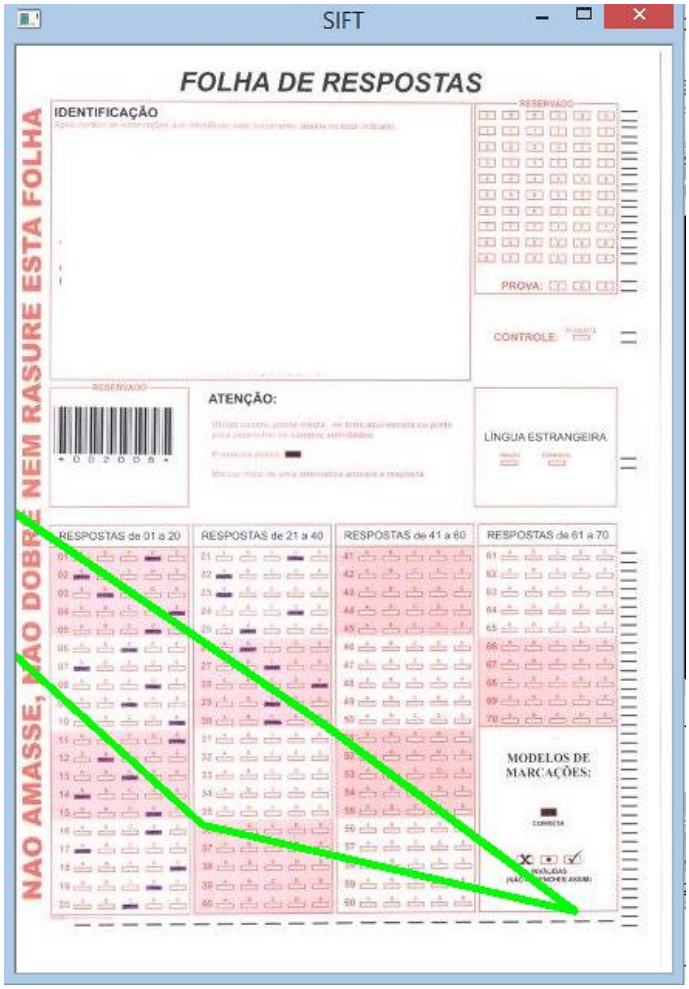


Figura 33 - Exemplo de erro de reconhecimento da área de interesse no cartão reposta a partir do algoritmo SIFT.
Fonte: Autoria própria.

Nos testes realizados apresentam-se os índices de tempo na Tabela 2, o tempo refere-se para cada uma das imagens processadas, está variação de tempo se dá pelo fato das dimensões das imagens, apresentando resultados que variam de 30,449 a 0,328 segundos por imagem processada, com média de 9,590 segundos de processamento.

Tabela 2 - Índice de tempo do algoritmo SIFT

	Imagem modelo			
Imagem original	**2277 x 1585 px**	**667 x 460 px**	**392 x 273 px**	**Média**
2477 x 3504 px	30,449 (s)	12,623 (s)	9,181 (s)	
723 x 1024 px	3,134 (s)	1,144 (s)	0,924 (s)	9,590 (s)
420 x 600 px	1,011 (s)	0,456 (s)	0,328 (s)	

Fonte: Autoria própria.

Notas:
(s) tempo avaliado em segundos.
px valor da resolução da imagem.

4.1.2 SURF

No mesmo seguimento do algoritmo SIFT o algoritmo SURF foi testado com a mesma proporção de 55 imagens de cartões repostas adquiridos. Na Tabela 3 são apresentados os índices de acertos em encontrar a área de interesse nos cartões respostas através do algoritmo SURF.

Tabela 3 - Índice de acertos do algoritmo SURF

	Imagem modelo			
Imagem original	**2277 x 1585 px**	**667 x 460 px**	**392 x 273 px**	**Média**
2477 x 3504 px	65,45 %	90,90 %	74,54 %	
723 x 1024 px	74,54 %	98,18 %	54,54 %	61,00 %
420 x 600 px	16,36 %	3,63 %	70,90 %	

Fonte: Autoria própria.

Notas:
px valor da resolução da imagem.

Na Tabela 4 são apresentados os índices de tempo do algoritmo SURF. A variação de tempo varia pelo fato da quantidade de pontos chaves que o algoritmo localiza, quanto maior a dimensão da imagem maior é a quantidade de pontos chaves também descritos como *keypoints*.

Tabela 4 - Índice de tempo do algoritmo SURF

	Imagem modelo			
Imagem original	**2277 x 1585 px**	**667 x 460 px**	**392 x 273 px**	**Média**
2477 x 3504 px	16,452 (s)	6,945 (s)	5,208 (s)	
723 x 1024 px	1,721 (s)	0,505 (s)	0,323 (s)	7,580 (s)
420 x 600 px	0,353 (s)	0,246 (s)	0,203 (s)	

Fonte: Autoria própria.

Notas:
(s) tempo avaliado em segundos.
px valor da resolução da imagem.

4.1.3 *Template Matching*

Na Tabela 5 são apresentados os índices de reconhecimento do algoritmo *Template Matching*, porém apresentando resultados de apenas seis testes, pois o *Template Matching* não

processa o algoritmo caso a imagem de modelo possuir dimensão maior que a imagem original.

Tabela 5 - Índice de acertos do algoritmo *Template Matching*

	Imagem modelo			
Imagem original	**2277 x 1585 px**	**667 x 460 px**	**392 x 273 px**	**Média**
2477 x 3504 px	100,00 %	0,00 %	0,00 %	
723 x 1024 px	*	100,00 %	0,00 %	50,00%
420 x 600 px	*	*	100,00 %	

Notas:
px valor da resolução da imagem.
(*) o algoritmo *Template Matching* não executa o processo com imagem modelo de dimensões maiores que a imagem original.

Na Tabela 6 são apresentados os resultados do tempo de processamento do algoritmo *Template Matching*, tempo correspondente a cada imagem processada.

Tabela 6 - Índice de tempo do algoritmo *Template Matching.*

	Imagem modelo			
Imagem original	**2277 x 1585 px**	**667 x 460 px**	**392 x 273 px**	**Média**
2477 x 3504 px	4,257 (s)	3,527 (s)	2,804 (s)	
723 x 1024 px	*	0,226 (s)	0,188 (s)	3,520 (s)
420 x 600 px	*	*	0,056 (s)	

Fonte: Autoria própria.

Notas:
(s) tempo avaliado em segundos.
px valor da resolução da imagem.
(*) o algoritmo *Template Matching* não executa o processo com imagem modelo de dimensões maiores que a imagem original.

Obtiveram-se resultados positivos quanto ao tempo de execução do reconhecimento da área de interesse a partir do algoritmo *Template Matching*, obtendo três índices apresentando 100% de reconhecimento dos 55 cartões respostas testados.

4.2 COMPARATIVO ENTRE OS ALGORITMOS

Como forma de comparativo dos algoritmos estudados a Tabela 7 apresenta os resultados dos algoritmos SIFT, SURF e *Template Matching* em testes realizados com objetivo reconhecer áreas de interesses em cartões resposta.

Tabela 7 - Comparativo dos algoritmos SIFT, SURF e *Template Matching*

Resolução da imagem original (px)	Resolução da imagem modelo (px)	Índice de Reconhecimento dos algoritmos			Tempo de execução dos algoritmos		
		SIFT (%)	SURF (%)	*Template Matching* (%)	SIFT (s)	SURF (s)	*Template Matching* (s)
2477 x 3504	**2277 x 1585**	74,54	65,45	100,00	30,449	16,452	4,257
2477 x 3504	**667 x 460**	100,00	90,90	0,00	12,623	6,945	3,527
2477 x 3504	**392 x 273**	89,09	74,54	0,00	9,181	5,208	2,804
723 x 1024	**2277 x 1585**	98,18	74,54	*	3,134	1,721	*
723 x 1024	**667 x 460**	100,00	98,18	100,00	1,144	0,505	0,226
723 x 1024	**392 x 273**	74,54	54,54	0,00	0,924	0,323	0,188
420 x 600	**2277 x 1585**	52,72	16,36	*	1,011	0,353	*
420 x 600	**667 x 460**	63,63	3,63	*	0,456	0,246	*
420 x 600	**392 x 273**	54,54	70,90	100,00	0,328	0,203	0,056
	Média	**78,58**	**61,00**	**50,00**	**9,590**	**7,580**	**3,529**

Fonte: Autoria própria.

Notas:
(s) tempo avaliado em segundos.
(px) valor da resolução da imagem.
(*) o algoritmo *Template Matching* não executa o processo com imagem modelo de dimensões maiores que a imagem original.

A partir da Tabela 7 pode-se analisar que o algoritmo SIFT possuiu o maior índice de eficiência no reconhecimento da área de interesse, apresentando uma média de 78,58% de reconhecimento a partir dos nove testes de resoluções, com testes apresentando 100% de reconhecimento, porém sendo considerado o algoritmo mais lento em tempo de processamento, com média de tempo de 9,590 segundos para cada imagem processada.

O algoritmo SURF sua média de reconhecimento chegou a 61% dos cartões processados, sendo inferior ao índice do algoritmo SIFT. O algoritmo em uma sessão de teste apresentou apenas 3,63% à área de interesse, ou seja, dos 55 cartões testados o algoritmo reconheceu a área de interesse em apenas dois cartões, porém o seu tempo de processamento chega a ser 26% mas ágil que o algoritmo SIFT com uma média de 7,580 segundos para cada processamento.

A seguir o *Template Matching* é o algoritmo que possuiu o melhor índice de tempo de processamento, sendo considerado o algoritmo mais ágil testado, apresentando uma média de 3,539 segundos apresentando agilidade de mais de seis segundos referentes ao algoritmo SIFT, nos testes em que as imagens de modelo correspondem ao tamanho da área de interesse do cartão resposta, os índices de acertos são de 100% com os 55 cartões respostas testados, porém em cartões respostas em que a imagem modelo é inferior as dimensões da área de interesse, o algoritmo não consegue extrair a área nas proporções necessária, apresentando erros em todos os testes de reconhecimento, como é apresentado na Figura 34.

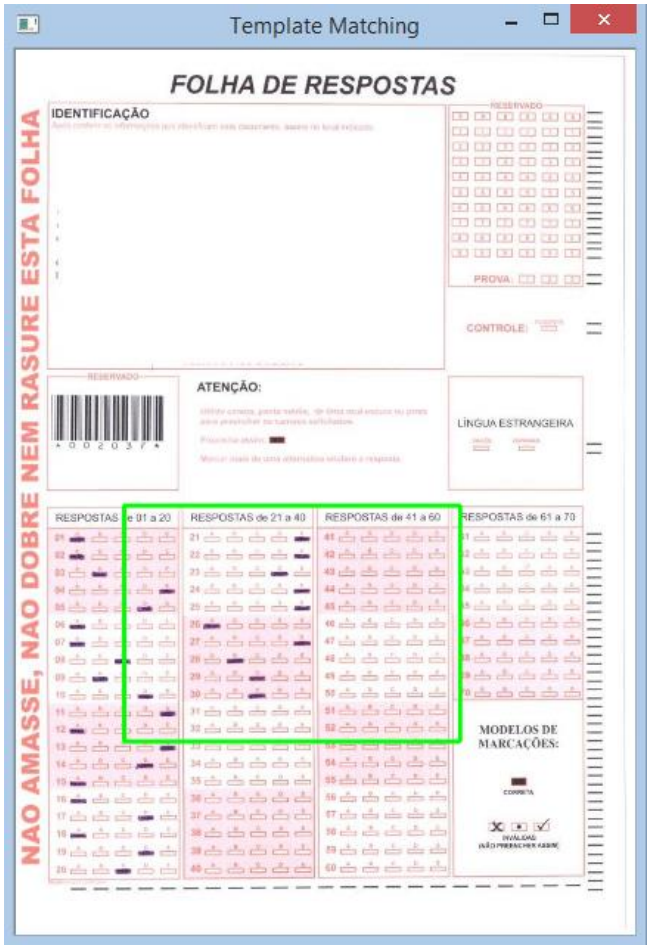


Figura 34 - Erro de reconhecimento com o algoritmo *Template Matching*
Fonte: Autoria própria.

Para este banco de imagens o algoritmo *Template Matching* foi o qual mais se adaptou, apresentando os melhores resultados de reconhecimento e menor tempo de processamento, porém em um exemplo de imagens desalinhadas o algoritmo *Template Matching* não consegue o reconhecimento da área de interesse em que somente os algoritmos SIFT e SURF possuem a capacidade de reconhecer como é apresentado nas Figura 35 e Figura 36.

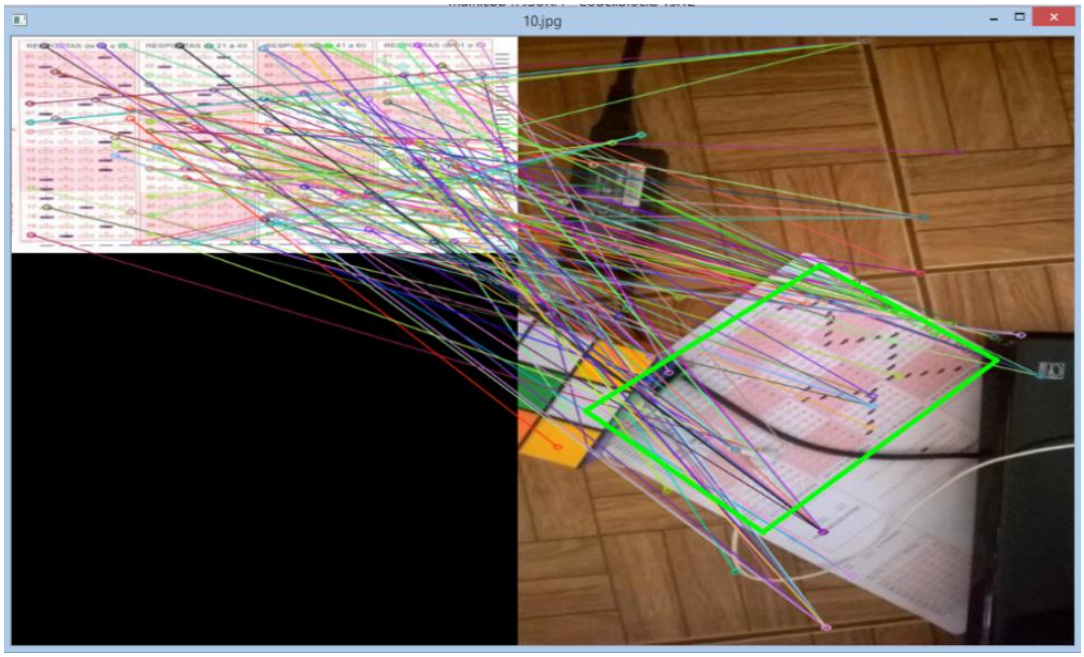


Figura 35 - Exemplo de reconhecimento da área de interesse em imagens mal adquiridas a partir do algoritmo SIFT.
Fonte: Autoria própria.

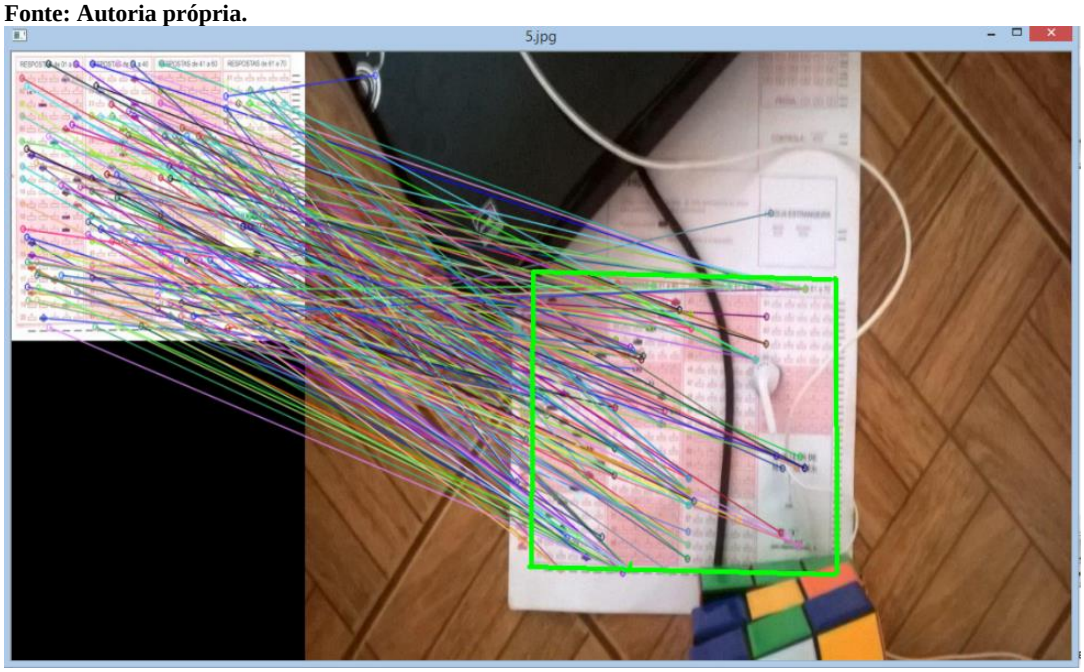


Figura 36 - Exemplo de reconhecimento algoritmo SURF a partir de imagens desalinhadas.
Fonte: Autoria própria.

5 CONSIDERAÇÕES FINAIS

Este capítulo aborda a conclusão obtida após a elaboração e análise do trabalho desenvolvido, bem como ideias para melhorar ou continuar o projeto construído neste trabalho.

5.1 CONCLUSÕES

O trabalho apresentado atendeu as especificações definidas, reconhecendo os cartões respostas, adquiridos por meio de um scanner convencional, processando a partir de algoritmos da biblioteca OpenCV e validando os mesmos, até chegar aos resultados preenchidos no cartão resposta e armazenando estes resultados em arquivos de texto.

Além do processamento dos cartões respostas, foi possível verificar experimentalmente a complexidade computacional dos três algoritmos de reconhecimento estudados: SIFT, SURF e *Template Matching.* O comportamento experimental destes algoritmos foi de acordo com o previsto. Foi possível dentre os algoritmos verificarem a sua eficiência em relação ao reconhecimento nos cartões respostas e a verificação quais são mais ágeis para determinadas situações.

Após análise dos algoritmos pode-se concluir que cada algoritmo possui suas características próprias. Sendo que na correção de cartões respostas, nos padrões de digitalização testados neste trabalho o algoritmo *Template Matching* da biblioteca OpenCV possuiu o melhor resultado, com o menor tempo de processamento e com alto índice de reconhecimento nas imagens apresentando resultados satisfatórios. Porém em casos em que as imagens não estavam alinhadas o algoritmo possui dificuldade em reconhecer as áreas de interesses referentes a este trabalho.

5.2 TRABALHOS FUTUROS

Como trabalho futuro deste projeto, sugere-se a continuação de testes de reconhecimento a partir dos algoritmos apresentados com imagens capturadas por meio de câmeras fotográficas, aplicando técnicas afim de reconhecer, calibrar e corrigir os cartões assim facilitando a aquisição de imagens.

Para trabalho futuro sugere-se em aplicar o projeto proposto em um sistema *online*, para efetuar todoo processo de correção comparando as respostas apresentadas pelo candidato com o gabarito, corrigindo e apresentando a pontuação do candidato.

REFERÊNCIAS BIBLIOGRÁFICAS

ADAMS, L. M. **Análise de textura usando técnicas fractais para reconhecimento de espécies florestais**. 2014. 70f. Trabalho de conclusão de curso, Universidade Tecnológica Federal do Paraná, Medianeira, 2014.

ALMEIDA, I. O. **Metodologia semi-automática para construção 3d de sólidos geométricos baseada em imagem**, SÃO LUIZ – MA, 2007 disponível em: <http://www.tedebc.ufma.br/tde_arquivos/10/TDE-2007-10-19T175201Z-50/Publico/Irlandino%20Almeida.pdf> acesso em 29. Out. 2014.

BALLESTA, M. GIL, A. MOZOS, O.M. REINOSO, O., 2007 Local descriptors for visual SLAM, in ,**Workshop on Robotics and Mathematics**", Portugal:Coimbra

BAY, H. ESS, A. TUYTELAARS, T. GOOL, V. L. **Surf: Speeded Up Robust Features, computer vision and image**, Understanding (CVIU), Disponível em: <ftp://ftp.vision.ee.ethz.ch/publications/articles/eth_biwi_00517.pdf>

BERGERON, B. P. **O reconhecimento de marca óptica. Postgraduate medicin.** 1998, disponível em: <http://www.postgradmed.com/issues/1998/08_98/dd_aug.htm> Acesso em: 07. Jun. 2014.

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

BRUNELLI, R, Técnicas de casamento de modelos em visão computacional: teoria e prática. 2009.

CODE::BLOCKS (2015) Code::Blocks, disponível em: < http://www.codeblocks.org/> acesso em: 15. Out. 2015.

DELAI, R. L; COELHO, A. D. **Aplicação de ferramenta open source em sistemas de visão computacional – desenvolvimento de um veículo seguidor autônomo**. São Caetano do sul – SP, 2011. Disponível em: < http://www.abenge.org.br/CobengeAnteriores/2011/sessoestec/art1674.pdf> acesso em: 12. Set. 2015.

FULCOAT. **Catálogo de cores, padrão Munsell**. Disponível em: < http://fullcoat.com.br/catalogo_munsell.html> acesso em: 27. Out. 2015

GAMITO, M. M. A. **A cor na formação do designer.** 2005, disponível em: <https://www.repository.utl.pt/bitstream/10400.5/5985/1/A%20Cor%20na%20Forma%C3%A7%C3%A3o%20do%20Designer_Disserta%C3%A7%C3%A3o.pdf> acesso em: 12. Out. 2015

GONZALEZ, R. C.; WOODS, R. C. **Processamento digital de imagens.** 3 ed. Pearson Prentice Hall, São Paulo, 2010.

GONZALEZ, R. C.; WOODS, R. C.; EDDINS, S. L. **Digital Image Processing Using MATLAB. Pearson Education**, 2004. 609 p.

LOPES, R. O.; SIMÃO, R. B. **Implementação do algoritmo sift para detecção de objetos em imagens**, Rio de Janeiro, 2011. Disponível em: <http://www.lcg.ufrj.br/Cursos/cos756/trabalhos-2011/trabalhos-2011/rafael-roberto-sift-relatorio.pdf> acesso em 29. Out. 2014.

LOWE, D. G. **Distinctive image features from scale-invariant keypoints.** University of British Columbia, Vancouver - Canadá, 2004. disponível em: <http://www.cs.berkeley.edu/~malik/cs294/lowe-ijcv04.pdf> acesso em: 12. Set. 2014.

MARENGONI, M.; STRINGHINI, D. **Introdução à visão computacional usando opencv**, 2009. Disponivel em: <http://seer.ufrgs.br/rita/article/viewFile/rita_v16_n1_p125/7289> acesso em: 31. Out. 2015.

MARQUES FILHO, O; VIEIRA NETO, H. **Processamento digital de imagens**, Rio de Janeiro, 1999.

OLIVEIRA NETO, V. J. Towards license plate recognition: comparying moving objects segmentation approaches, **International Conference on Image Processing, Computer Vision, and Pattern Recognition**, Las Vegas, 2012.

OPENCV. **Template Matching** 2015. disponível em: <http://docs.opencv.org/doc/tutorials/imgproc/histograms/template_matching/template_matching.html> acesso em 14. Out. 2015.

PACIEVITCH, Y. **Estudo e características da linguagem e programação C/C++**, disponível em: < http://www.infoescola.com/informatica/cpp/> acesso em: 13. Out.2015.

PEDRINI, H.; SCHWARTZ, W. R. **Análise de imagens digitais: princípios, algoritmos e aplicações**. São Paulo: editora Thomson Learning, 2008

PLOTZE, R. O.; BRUNO, O. M. Análise de formas e reconhecimento de padrões por meio da assinatura fractal multi-escala. Infocomp **– Journal of Computer Science**. 100p. , São Carlos, 2007.

PORTO, T. N. BITAR, P. A. S. **Protótipo de software para controle de acesso de funcionários através da autenticação digital**, Belém 2005. Disponível em: <http://www3.iesam-pa.edu.br/ojs/index.php/computacao/article/viewFile/48/45> acesso em: 12. Out. 2015.

SANTOS, E. A. **Utilização de equações diferenciais parciais no tratamento de imagens orbitais**, Presidente Prudente - SP, 2002, disponível em: <http://www2.fct.unesp.br/pos/cartografia/docs/teses/d_santos_ea.pdf> acesso em: 26 set. 2015.

SANTOS, R. M. **Um estudo de processamento de imagens com opencv**, Niterói - SP, 2011 disponível em: <https://pt.scribd.com/doc/95600971/11/BINARIZACAO> acesso em 28. Out. 2014.

SEARA, D. M. **Algoritmos para detecção de bordas**. Florianópolis - SC, 1998. Disponível em: < http://www.inf.ufsc.br/~visao/1998/seara/index.html > Acesso em: 09 set. 2015.

SERRILHO, T. C. Identificação de código de barras em documentos de tramitação interna da unemat com utilização de webcam para coleta de dados. Colider – MT, 2009. Disponível em: <http://livros01.livrosgratis.com.br/ea000670.pdf> Acesso em: 21 Out. 2015.

SILVA, A. M., **Curso processamento digital de imagens de satellite**, Porto Alegre - RS, 2001, disponível em: <http://www.cartografia.org.br/> acesso em 25. Out. 2014.

SILVA, E. A., SANTOS, F. P., LEONARDI, F., GONÇALVES, T. R. **Aplicação de técnicas de morfologia matemática e PDI na detecção semi-automática de feições cartográficas em imagens digitais** In: VII Congresso de Cadastro Técnico Multifinalitário e Gestão Territorial, Florianópolis, 2006.

SILVA, F. A; PAIVA, M. S. V.. **Evaluation of keypoint detectors and descriptors**. In: WORKSHOP DE VISÃO COMPUTACIONAL - WVC 2013, Rio de Janeiro, 2013. disponível em:< http://iris.sel.eesc.usp.br/wvc/Anais_WVC2013/Poster/1/18.pdf> acesso em 23. Jan. 2015.

SPRING, V.; Integrating remote sensing and gis bt object-oriented data modelling, Berlin 1996.

TAHMASEBI, P. **Multiple-point geostatistical modeling based on the cross-correlation functions**, 2012 disponível em:< http://link.springer.com/article/10.1007%2Fs10596-012-9287-1> acesso em 28. Jul. 2015.

TEIXEIRA, A. S. B. **Desenvolvimento de uma interface gráfica para classificadores de imagem**, 2011 disponível em: < http://repositorio.ipcb.pt/bitstream/10400.11/1155/1/disserta%C3%A7ao.pdf> acesso em 28. Out. 2014.

TING, W. S.; **Sistemas de informações gráficas**, Campinas-SP, 2009 disponível em: <http://www.dca.fee.unicamp.br/cursos/EA978/1s2009/notas/ea978_CG.pdf> acesso em 28. Jul. 2014.

VERDE, P. **Sistemas de digitalização óptica**. Ace The Electoral Knowledge Network, 2000 disponível em: <http://www.aceproject.org/main/english/et/et72.htm> Acesso em: 09. Jun. 2014.