**DESENVOLVIMENTO DE APLICAÇÃO WEB PARA CORREÇÃO DE PROVAS OBJETIVAS UTILIZANDO PROCESSAMENTO DE IMAGEM**

*Development of a web application for multiple-choice tests correction using image processing techniques.*

|  |  |  |
| --- | --- | --- |
| **Nicolas C. Kelepuris**  *Graduando*  Engenharia da Computação  [*nicolas.kelepuris@ucl.br*](mailto:nicolas.kelepuris@ucl.br) | **Vinicius A. S. Priori**  *Graduando*  Engenharia Elétrica  [*viniciuspriori@ucl.br*](mailto:viniciuspriori@ucl.br) | **Roger A. C. Freitas**  *Orientador*  Faculdade UCL  [*rogerfreitas@ucl.br*](mailto:rogerfreitas@ucl.br) |

A lista completa com informações de autoria está no final do artigo

**RESUMO**

**Objetivo:** O presente trabalho tem como objetivo implementar um sistema *user-friendly* e gratuito com o intuito de agilizar a correção de provas objetivas utilizando técnicas de processamento de imagem e visão computacional integrados a uma aplicação *web*.

**Método:** A abordagem empregada é a de pesquisa exploratória quantitativa, partindo da hipótese de que o processo de correção de provas objetivas das instituições de ensino pode ser otimizado através da implementação de um sistema de correção automática utilizando visão computacional.

**Resultado:** Após o levantamento de hipóteses estipulando o tempo médio gasto por professor para as correções das provas objetivas do fim do semestre, apresentou-se a oportunidade de otimizar o processo de correção, propondo uma solução prática e *user-friendly*. A aplicação desenvolvida se mostrou precisa em 96% das vezes, tendo falhado apenas na situação onde houve rasura.

**Conclusões:** Considerando o tempo do professor pegar a prova em mãos, comparar com o gabarito, realizar a correção, conferir os resultados e finalmente inserir os resultados no sistema, é possível que o aplicativo desenvolvido possa ser utilizado pelos professores interessados da instituição a fim de agilizar a correção de provas de múltipla escolha.

**PALAVRAS-CHAVE:** Aplicação *web*. Processamento de Imagem. Visão Computacional. Automação.

**ABSTRACT**

**Objective:** The present work aims to implement a free and user-friendly system in order to speed up tests correction using image processing techniques and computer vision integrated into a web application.

**Methods:** The employed approach is that of quantitative exploratory research, starting from the hypothesis that the process of correction of objective tests of educational institutions can be optimized through the implementation of an automatic correction system using computer vision.

**Results:** After raising theoretical hypotheses stipulating the average expected time spent per teacher for correcting multiple choice tests held at the end of semester, an opportunity was presented to optimize the correction process, proposing a practical and user-friendly solution. The developed application proved to be accurate in 96% of the time, having failed only in the situation where there was an erasure.

**Conclusions:** Considering the time spent by the teacher by taking the test at hand, comparing the test with the list of answers, checking the results and finally imputing the results in the system, it is feasible that the developed application could be used by the interested teachers at the institution in order to speed up a correction of tests of multiple choice.

**KEYWORDS:** Web application. Image processing. Computer Vision. Automation.

# INTRODUÇÃO

Conforme a digitalização da informação torna-se cada vez mais indissociável às práticas humanas, uma de suas consequências diretas é a de que o homem contemporâneo passe a delegar cada vez mais as tarefas repetitivas e monótonas às máquinas, como tem ocorrido desde o advento da automação industrial, datada no século XVIII com a primeira revolução industrial inglesa, até os dias de hoje.

Nesse contexto, o processamento de imagens, tema central do presente trabalho, também acompanha essa tendência, possuindo atualmente aplicações em quase todos os ramos da atividade humana, entre elas na biologia, na automação, na medicina, na astronomia, no geoprocessamento, dentre muitas outras. [1] O presente trabalho, motivado por essa crescente tecnológica, possui como objetivo desenvolver uma aplicação web para a correção de provas de múltipla escolha de forma simples e rápida, pois sabe-se que a correção de gabaritos é um processo lento e passível de consideráveis índices de erros, além de exigir do professor seu tempo de descanso fora de horário de trabalho para executar tais atividades. [10]

Podemos definir processamento de imagens como um conjunto de técnicas computacionais para análise, aprimoramento, compressão e reconstrução de imagens. Geralmente se refere ao processamento de imagem digital, mas o processamento de imagem óptica e analógica também é possível. [2] O trabalho aqui apresentado trata desta primeira categoria. Entretanto, antes de prosseguir com uma abordagem mais ampla sobre sistemas e técnicas de processamento, faz-se necessário contextualizar conceitos acerca de imagens digitais e sistemas de processamento de imagens, a fim de elucidar como estes se relacionam com a aplicação desenvolvida.

# REVISÃO BIBLIOGRÁFICA

## IMAGENS DIGITAIS

Imagens podem ser definidas como sendo uma função bidimensional, , onde x e y são representadas por coordenadas espaciais e a amplitude de em qualquer ponto representado por essas coordenadas é denominada intensidade, brilho ou nível de cinza da imagem nesse ponto. Quando os valores de e são todos finitos, ou seja, discretos, essa imagem é denominada imagem digital e seus elementos são chamados de *pixels*, do inglês *picture elements*.[3]

Os níveis de brilho ou intensidade normalmente se encontram no intervalo , sendo o número de bits para cada *pixel* da imagem necessário para seu armazenamento. Exemplificando, para uma imagem representada em tons de cinza, ou seja, possuindo apenas um canal de cor, tendo , a imagem em questão terá a capacidade de armazenar até 256 níveis de cinza, onde o preto corresponde ao 0 e o branco ao 255. [4]

As imagens digitais podem ser processadas ou representadas de duas maneiras distintas: matricial ou vetorial. Na forma matricial, a imagem é formada por uma série ordenada de pixels dispostos continuamente em linhas e colunas. Na forma vetorial, a imagem é composta por uma série de instruções matemáticas e a imagem apresentada é gerada a partir do cálculo dessas instruções (Figura 1). [5]

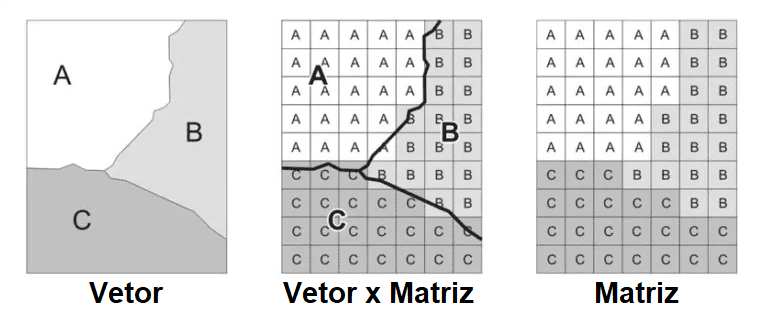


Figura 1: Representações de uma imagem. [6]

O sistema automatizado de correção de provas utilizando processamento de imagem, a ser plenamente detalhado nas próximas sessões, tratará apenas de imagens matriciais em tons de cinza (monocromáticas).

É prática comum em processamento de imagens associar um sistema de coordenadas à imagem, tendo como origem o pixel superior esquerdo. A Figura 2 mostra convenção de orientação dos eixos que foi utilizada tanto no desenvolvimento da aplicação tanto quanto nesta dissertação [7].

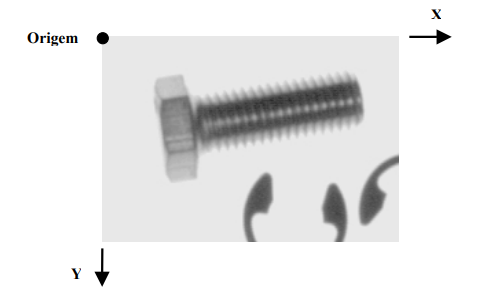


Figura 2: Convenção de eixos de uma imagem matricial bidimensional. [7]

## VISÃO COMPUTACIONAL

A visão computacional é a subárea da computação gráfica que visa a extração de informações úteis a partir de imagens capturadas do mundo real. Esse termo é muitas vezes apresentado como sinônimo de análise ou processamento de imagens, porém, no contexto do presente trabalho, o conceito de processamento de imagens é uma parte constituinte do conceito de visão computacional. [8]

Os sistemas de processamento de imagem possuem atrelados a sua definição um conjunto de processos, não necessariamente obrigatórios, que podem ser resumidamente definidos como: aquisição ou captura, aprimoramento/filtragem, restauração, processamento de cor, *wavelets*, compressão, processamento morfológico, segmentação, representação e descrição e, por fim, reconhecimento. [3]

A fim de simplificar e abreviar esta seção, apenas as etapas utilizadas no presente trabalho serão mais descritas com maior detalhamento abaixo.

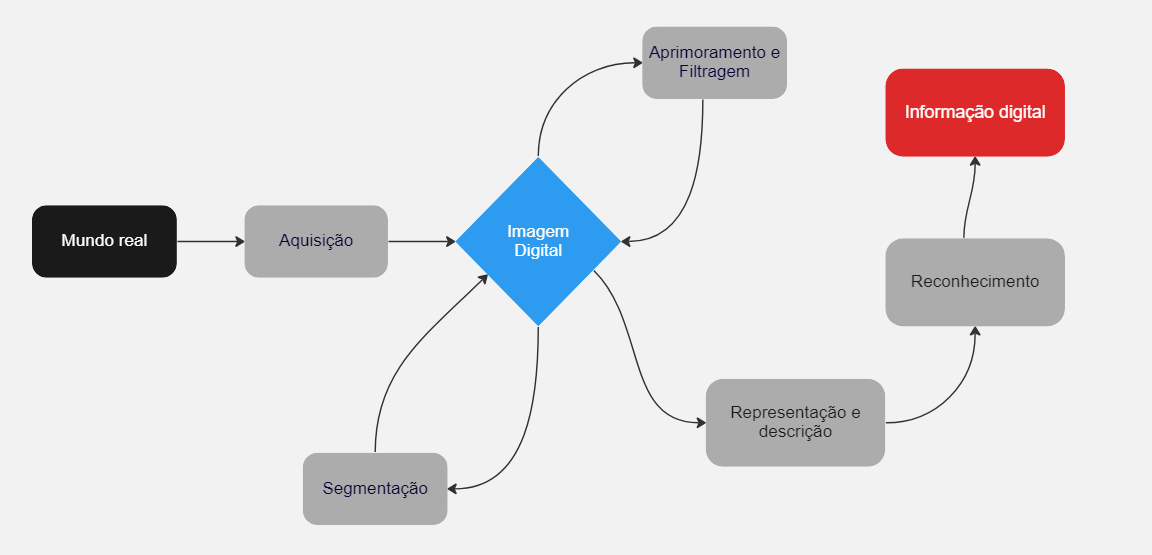


Figura 3: Etapas de um sistema de visão computacional. [8]

Conforme o diagrama da Figura 3, temos os processos a seguir:

1. Aquisição: A captura ou aquisição trata-se da obtenção de uma imagem através de um dispositivo ou sensor. Nos aspectos relacionados nessa etapa, é denominada a configuração do dispositivo de captura, como o formato da imagem digital, as configurações de luminosidade, resolução, número de níveis de cinza ou cores da imagem digital. [9]
2. Aprimoramento/Filtragem: É definida como sendo a manipulação das características da imagem com o objetivo de torná-la mais adequada para o contexto na qual está inserida. Essa etapa pode variar bastante conforme a aplicação, visto que diferentes problemas requerem diferentes formas de tratamento.
3. Segmentação: Trata-se da separação de objetos de interesse de uma imagem, a fim de facilitar a extração de características relevantes para a saída de um determinado sistema. No presente trabalho, por exemplo, a segmentação é utilizada para separar a região das alternativas de múltipla escolha.
4. Representação e descrição: Nesta etapa, que quase sempre ocorre logo após a segmentação, os dados da imagem são convertidos para um formato adequado para o processamento (representação) e em seguida têm seus atributos extraídos para os denominados descritores de imagem. Por fim, estes são comparados com outros descritores de imagem, possibilitando a diferenciação ou correspondência desses atributos (descrição).
5. Reconhecimento: Essa é a etapa final do processamento e consistem em categorizar uma imagem com base nos atributos extraídos da etapa anterior.

## COMPARATIVOS

A utilização de técnicas de processamento de imagem para a correção de provas de múltipla escolha não é um tema inédito. Como o trabalho feito por MACHADO et. al [11], a premissa básica é a de termos uma prova marcada em mãos e em seguida escaneada para ser processada pelo sistema, que por fim enviará o *feedback* da nota dos alunos. Um dos problemas remanescentes que o presente trabalho busca solucionar é o do cadastro do aluno. A aplicação aqui desenvolvida também utiliza técnicas de processamento para identificar a matrícula do aluno, tornando a necessidade de um sistema integrado de cadastro para cada aluno desnecessária, já que o grupo de arquivos enviado pelo professor será contabilizado como uma turma e posteriormente será gerada uma planilha com os resultados das correções por matrícula, o que torna a aplicação mais genérica para diferentes tipos de usuários ao invés de fechada para um sistema específico atrelado a um banco de dados de uma única instituição.

Quanto ao trabalho realizado por BATISTA, et. al [10], uma aplicação desktop foi desenvolvida para realizar a correção das provas. É válido salientar a preocupação dos autores em garantir que a área de interesse seja corretamente identificada através do processo de calibração por eles desenvolvido. Porém, o fato do algoritmo subdividir essa zona demarcada sempre em 5 linhas e 10 colunas torna a aplicação pouco versátil, já que no cotidiano podemos ter uma extensa lista de combinações de questões e alternativas. Pensando nisso, o presente trabalho procurou tornar o número de alternativas e questões como sendo um parâmetro definido pelo usuário ao gerar o modelo de prova. Outro diferencial proporcionado é a possibilidade de acesso por *mobile* e *desktop*, já que se trata de uma aplicação web.

Além disso, incluiu-se a possibilidade da prova ser preenchida de forma on-line, o que demonstra um esforço para a redução de tempo e trabalho envolvidos nos processos de fotografar e escanear a prova.

# METODOLOGIA

O trabalho aqui apresentado trata-se de uma pesquisa exploratória quantitativa sobre processamento de imagens e desenvolvimento de aplicações *web*, a fim de obter familiaridade com os populares *frameworks* e linguagens de programação do mercado, além de aprofundar os conhecimentos sobre processamento de imagens.

Foi realizada uma revisão bibliográfica da literatura sobre imagens digitais e das técnicas de processamento comumente utilizadas, além das linguagens de programação e bibliotecas comumente utilizadas para esses tipos de aplicação. No caso do presente trabalho, foi escolhida a linguagem Python para o back-end da aplicação, devido à popularidade, eficiência e rápida curva de aprendizado que esta possui, além de possuir a biblioteca OpenCV disponível, a mais utilizada atualmente no ramo de processamento de imagens e visão computacional.

Para a interface de usuário, a opção por JavaScript, junto com o *framework* React, surgiu como favorita devido a imensa versatilidade que a mesma proporciona para o desenvolvimento de aplicações *web*. O fato de ser, atualmente, a linguagem de programação mais utilizada no mundo [15], contribuiu para essa decisão, visto que, pelo fato de possuir uma grande comunidade, torna a busca pelo esclarecimento de eventuais dúvidas em fóruns e/ou na documentação menos trabalhosa.

Definidas as linguagens de programação, foram realizados testes de intercomunicação entre as requisições HTTP da interface de usuário e o código do back-end, em Python. O *framework* escolhido para realizar essa intercomunicação foi o FastAPI, devido a sua popularidade e facilidade de uso, somadas a documentação simples e bem exemplificada.

Prosseguindo para a aplicação, após a estrutura dos códigos de back e front-end estarem funcionais, foram gerados modelos de prova, que posteriormente foram impressos, marcados e escaneados, para que a aplicação pudesse ser devidamente testada.

Foi discutida a possibilidade do uso de uma rede neural para identificar os números da matrícula do aluno, que seria escrita ao lado do nome, ao invés de marcada. Porém, após a realização de testes com diferentes parâmetros, constatou-se que a acurácia obtida não seria o suficiente para uma detecção confiável e com boa repetibilidade.

# DESENVOLVIMENTO

O sistema desenvolvido pode ser melhor compreendido se dividido em duas áreas, front-end e back-end, como comumente é feito no meio de desenvolvimento de software. O front-end consiste numa aplicação web em JavaScript, utilizando o *framework* React em conjunto com as linguagens HTML e CSS. Esse segmento da aplicação é destinado a interface do usuário, ou seja, um meio pelo qual este fará suas requisições para o sistema; que por sua vez serão processadas pelo back-end, área na qual estão situadas as lógicas de negócio da aplicação, que no caso do presente trabalho, englobam todos os sub-processos nos quais as imagens são recebidas, tratadas e avaliadas, conforme representado e descrito na Figura 3. O back-end foi desenvolvido em Python, utilizando os recursos da biblioteca OpenCV para a realização do processamento das imagens, da biblioteca Numpy, para a manipulação matricial e numérica, e a biblioteca FastAPI para realizar a intercomunicação entre o front-end e o back-end.

## FRONT-END

Do ponto de vista do usuário, a aplicação é bem simples. O primeiro passo é acessar o endereço do *website*, em seguida, selecionar, conforme desejado, os seguintes itens: quantidade de questões, quantidade de alternativas e quantidade de números presentes na matrícula do aluno, como demonstrado na Figura 4.

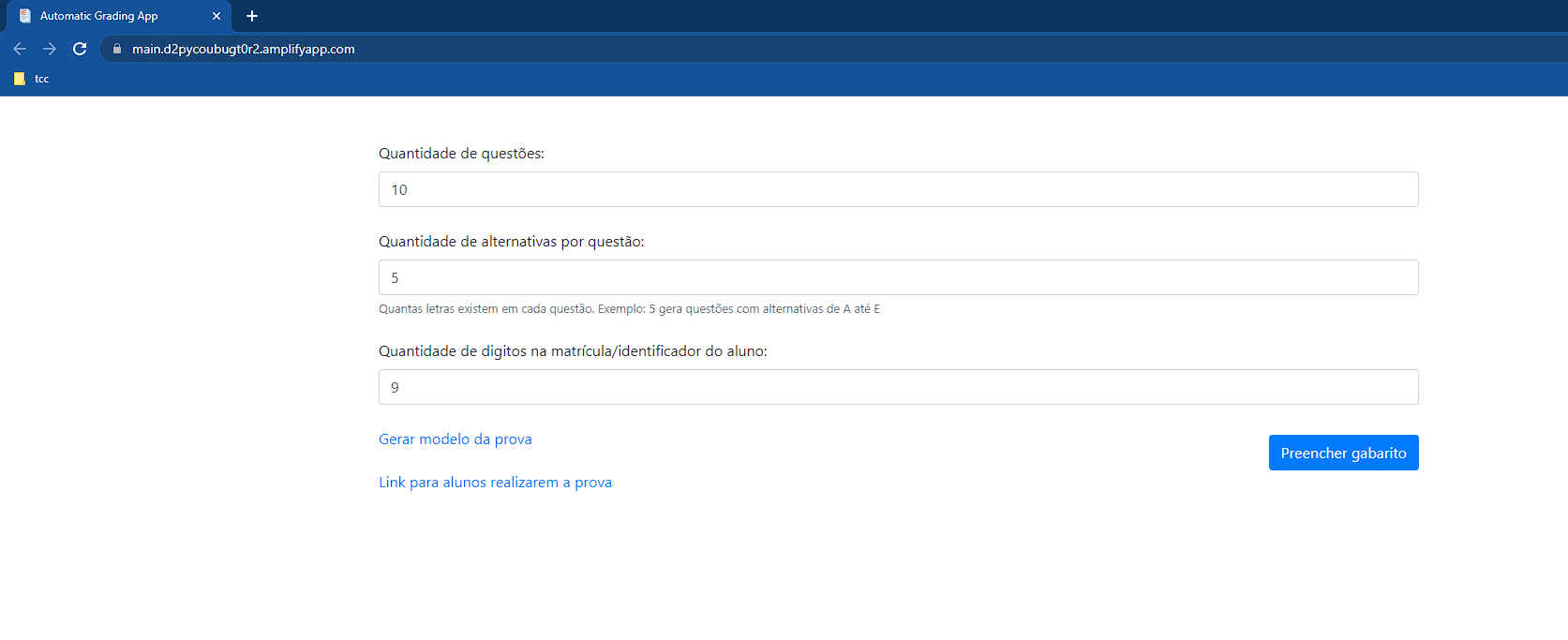


Figura 4: Tela inicial da aplicação. (Autoria própria)

Feito isso, deve-se gerar o modelo de prova, relizar sua impressão e seguir para a página de preenchimento do gabarito (Figura 5).



Figura 5: Preenchimento do gabarito com seus respectivos pesos. (Autoria própria)

Após essa etapa, basta fazer o upload de todas as imagens contendo as marcações das alternativas da prova e do número de matrícula do aluno e aguardar a planilha com a nota de cada aluno em relação ao respectivo número de matrícula preenchido (Figura 6).

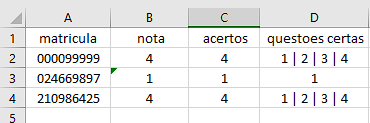


Figura 6: Planilha resultante com as notas dos alunos. (Autoria própria)

## BACK-END

A primeira parte tratada pelo código é o redimensionamento e a aplicação de filtros. A fim de permitir a aplicação de posteriores filtros para auxiliar a detecção de contornos, e reduzir o custo computacional do processo, é realizada a transformação do espaço de cor da imagem, afim de deixá-la em escala de cinza, como demonstrado na Figura 7.

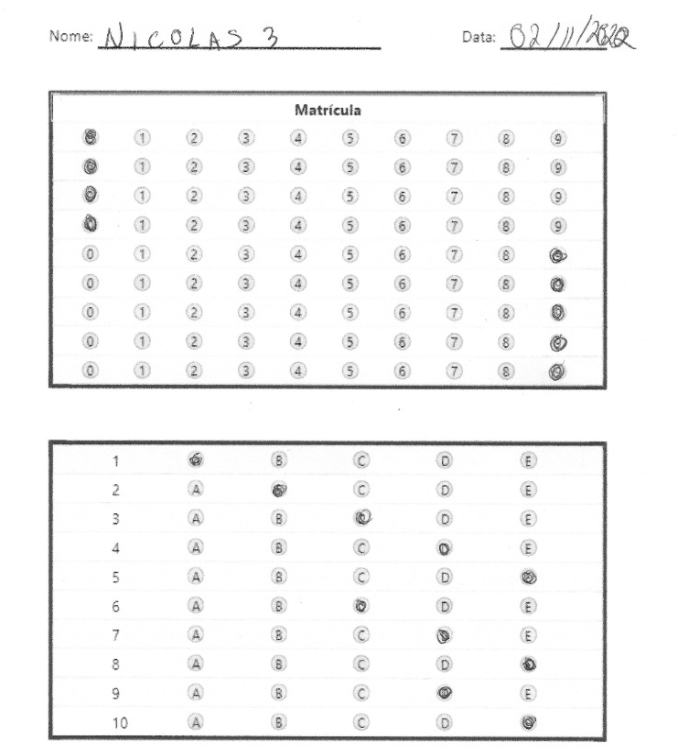


Figura 7: Prova após a transformação do espaço de cor. (Autoria própria)

Após a transformação do espaço de cor, aplica-se o filtro convolucional gaussiano. Em processamento de imagens, convolução é uma operação entre duas matrizes, geralmente bidimensionais, onde uma delas representa a imagem original e a outra representa uma matriz que contém modificações desejadas para imagem [12]. Essa matriz que contém as modificações desejadas é chamada de *kernel*. A convolução gaussiana aplica a distribuição normal (1) nos pixels em volta do *kernel* (matriz de convolução) e, como resultado, temos a atenuação de valores que possuem um desvio padrão mais elevado. [13] Visualmente, o resultado desse filtro é uma imagem um pouco “desfocada”, como demonstra a Figura 5, de uma de nossas provas-teste.

(1)

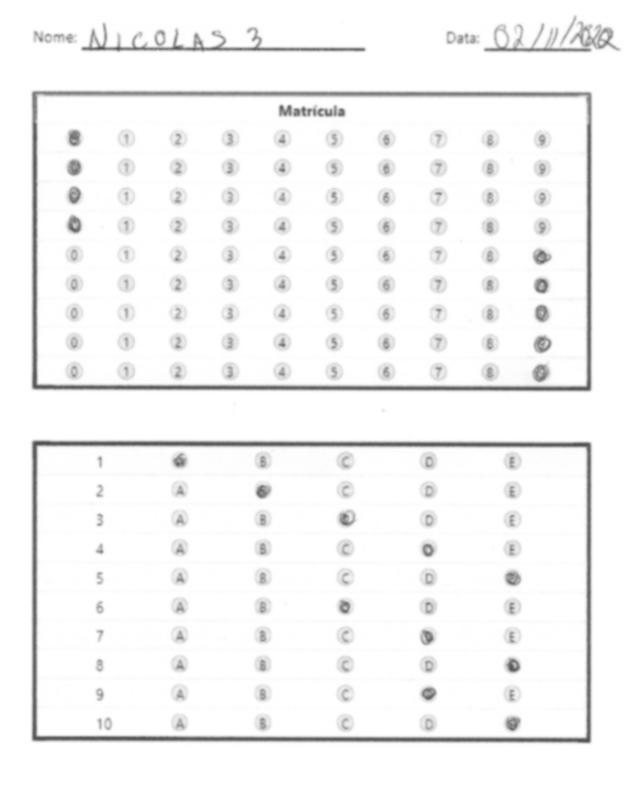


Figura 8: Imagem após a aplicação do filtro gaussiano. (Autoria própria)

Feita a atenuação de altas frequências pelo filtro gaussiano supracitado, realiza-se a detecção de contornos pelo algoritmo de Canny, abstraído pela função da biblioteca OpenCV cv2.Canny(), como é demonstrado na Figura 6. É boa prática de processamento de imagens utilizar o filtro gaussiano antes de começar a detecção de contornos, já que o ruído presente na imagem pode fazer com que o algoritmo falsamente detecte uma região de contorno na imagem.

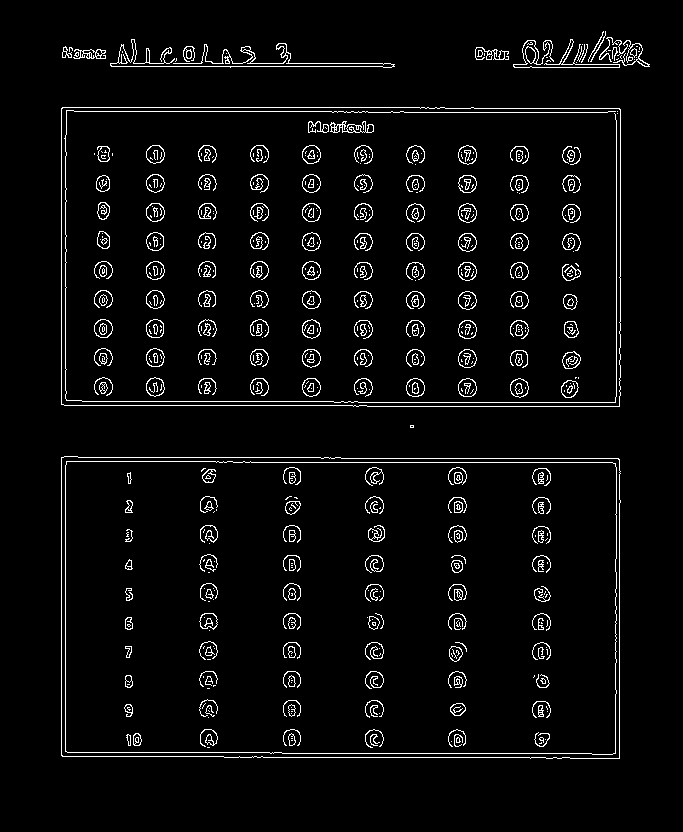


Figura 9: Prova após a detecção de contornos feita pelo algoritmo de Canny. (Autoria própria)

Realizada a detecção de contornos, prossegue-se para a segmentação das zonas de interesse, que no presente trabalho se tratam dos dois maiores retângulos. Dentro do primeiro deles, são marcados os números correspondentes à matrícula e dentro do segundo, as respostas de cada alternativa da prova. Com os contornos corretamente detectados, realiza-se a operação comumente chamada de *threshold*, onde a imagem, que até então estava sendo processada em escala de cinza, passa a assumir apenas dois valores, 0 ou 255. No caso do presente trabalho, foi utilizado o threshold binário inverso com o valor de 170. Em outras palavras, todos os pixels com valores de intensidade maiores do que 170 se tornarão 0, ou pretos. Como demonstra a Equação 2 [14] e a Figura 7.

(2)



Figura 10: Área de respostas após a aplicação do threshold binário inverso. (Autoria própria)

Feita a binarização da imagem, resta subdividir a área de interesse em “caixas”, onde cada uma ocupa o espaço de uma questão. Essa separação é feita realizando a divisão da matriz de pixels em linhas e colunas, através dos métodos vsplit() e hsplit() da biblioteca Numpy. Temos como entrada desse método a matriz da imagem, o número de colunas e o número de linhas, que são o número de questões e o número de alternativas, respectivamente. A saída desse método retorna uma lista contendo as “caixas”, onde cada uma representa uma alternativa ou ID marcado, como demonstra a Figura 8. Feita essa subdivisão, itera-se por cada item da lista aplicando o método cv2.countNonZero(), que nos retornará qual das alternativas possuem um maior número de pixels com valor diferente de 0. Essa alternativa corresponde àquela que o aluno realizou a marcação.

A fim de exemplificar, na Figura 9, tem-se a marcação do número de matrícula e na Figura 10, a matriz de saída com os pixels brancos contados. A primeira linha (0) é ignorada pelo código, pois se trata da linha correspondente ao *header* Matrícula. Da segunda linha em diante (1 a 9), pode-se notar que as regiões com maior densidade de pixels brancos são correspondentes às marcações das alternativas da Figura 9.



Figura 11: Imagem subdividida de forma a ocupar a região de uma única alternativa marcada. (Autoria própria)

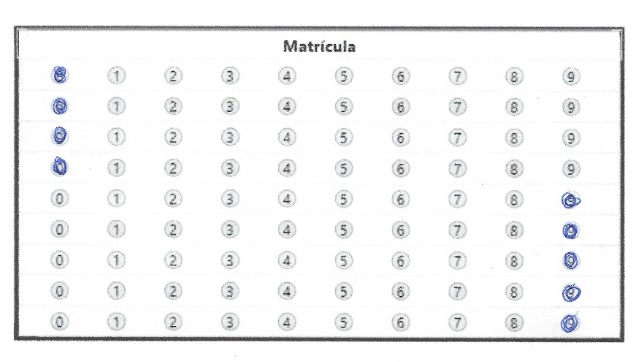


Figura 12: Marcação do número de matrícula. (Autoria própria)

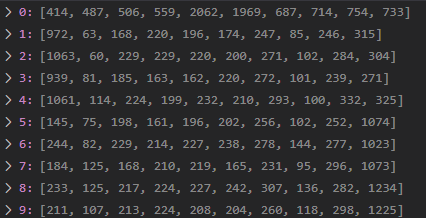


Figura 13: Matriz de saída após a aplicação do método cv2.countNonZero() por todas as alternativas.

(Autoria própria)

# RESULTADO

A aplicação foi testada gerando modelos de provas, marcando as respostas à caneta e escaneando as provas. Foram impressas, preenchidas e em seguida escaneadas uma quantidade de 25 provas. Dessas 25 provas, 23 eram preenchidas corretamente e 2 possuíam rasuras propositais a fim de testar a performance do sistema em situações adversas. Uma das rasuras foi feita próxima à marcação e não gerou alterações na detecção, já que a subdivisão da imagem assume uma região de contorno à alternativa. A outra rasura foi uma segunda marcação numa das questões, o que fez com que o sistema contabilizasse como certa uma das alternativas, ao invés de desconsiderar a questão. Como resultado, obteve-se 96% de acurácia, havendo erro em apenas uma das provas rasuradas.

# CONCLUSÃO

Como conclusão, constatou-se que, dentro das condições previstas, a aplicação funciona de forma satisfatória para a correção automática de provas objetivas. Dentre as melhorias a serem futuramente implementadas, podemos citar: melhoria no algoritmo de detecção das alternativas, a fim de prever questões em branco ou duplamente marcadas. Essa melhoria consiste na inserção de um valor de tolerância correspondente às questões desmarcadas, considerando o valor médio e o desvio máximo dos pixels de cada alternativa. Desta forma, haverá um valor limite de pixels que indica se a alternativa analisada foi de fato marcada, ou não. Também planeja-se o desenvolvimento de um ambiente que disponibiliza a realização e correção da prova por uma turma de alunos de forma integrada e totalmente *online*.

REFERÊNCIAS

[1] MARQUES FILHO, Ogê; VIEIRA NETO, Hugo. **Processamento Digital de Imagens**, Rio de Janeiro: Brasport, 1999. ISBN 8574520098.

[2] MEDEIROS, Caroline Pereira. **Processamento de Imagem e Aprendizagem de Máquina para Classificação de Anomalias em Vias Públicas.** Disponível em: <https://www.cin.ufpe.br/~tg/2018-1/cpm-tg.pdf>

[3] GONZALEZ, R. C., WOODS, R. E., **Digital Image Processing**, 3. ed., Addison-Wesley Publishing Company, 1993.

[4] FALCÃO, Alexandre Xavier. **Introdução ao Processamento de Imagem Digital**, 2003. Disponível em <http://www.dcc.unicamp.br/~cpg/material-didatico/mo443/200302>

[5] ALMERON, Cassio Pinheiro. **Estudo Em Processamento De Imagens Digitais Aplicado Ao Reconhecimento De Placas De Veículos Automotores Brasileiros**. Disponível em: <https://repositorio.ucs.br/xmlui/bitstream/handle/11338/1334/TCC%20Cassio%20Pinheiro%20Almeron.pdf?sequence=1&isAllowed=y> (Apud MANFIO, Felipe Haddad).

[6] GIOVANINI, Adenilson. Imagem Raster e Dados Vetoriais? **Blog Adenilson Giovanini**. Disponível em: <https://adenilsongiovanini.com.br/blog/imagem-raster-e-dados-vetoriais-qual-a-viferenca/>

[7] FREITAS, Roger Alex de Castro. **Sistema de Visão para Robôs Móveis: Uma Aplicação ao Reconhecimento de Referências Geométricas**. (Apud Myler, H. R., Weeks, A. R.).

[8] SAÚDE, André Vital. **Computação Gráfica e Processamento de Imagens**. Editora e Distribuidora Educacional S.A, 2019.

[9] NETO, Euripedes P. Fernandes. **Visão Computacional Para Identificação De Cores Em Tempo Real Com Opencv E Python**, 2020.

[10] BATISTA, M. de A. et al. **Aplicação desktop usando processamento de imagem para correção automática de gabaritos**, 2019. (Apud OCDE)

[11] MACHADO, Mateus F. et al. **Sistema de Correção Automática de Testes de Múltipla Escolha Usando Técnicas de Processamento de Imagens**, 2016.

[12] PEREIRA, Pedro H. **Um Método Para Correção Automática De Cartão De Resposta**

**Utilizando Processamento Digital De Imagem**, 2018.

[13] **Demystifying Gaussian Blur.** Acesso em: 10 nov. 2022. Disponível em: <https://www.adobe.com/creativecloud/photography/discover/gaussian-blur.html>

[14] **Documentação do OpenCV (Open Source Computer Vision).** <https://docs.opencv.org/>. Acesso em 12/11/2022.

[15] Top 10 JavaScript Usage Statistics to Watch Out for in 2022. **Radix Blog**. Disponível em: <https://radixweb.com/blog/top-javascript-usage-statistics>

[16] **Murtaza Hassan Blog**. <https://www.murtazahassan.com/>

NOTAS

**AGRADECIMENTOS**Agradeço a minha família por todo suporte ao longo desses anos.

**CONTRIBUIÇÃO E AUTORIA**

Não se aplica.

**FINANCIAMENTO**  
Não se aplica.

**CONSENTIMENTO DE USO DE IMAGEM**

Não se aplica.

**CURSO**

Engenharia Elétrica, Engenharia da Computação.

**COORDENADOR DO CURSO**

Dayane Corneau Broedel

**DATA DE ENTREGA**

21/11/2022

**BANCA AVALIADORA**

Vitor Jurtelo de Freitas

Ronan Fim Largura

**DECLARAÇÃO DE INEXISTÊNCIA DE PLÁGIO**

Declaro que o trabalho não contém plágio ou autoplágio total ou parcial. Todo o conteúdo utilizado como citação direta ou indireta foi indicado e referenciado.

**LICENÇA DE USO**

Os autores do artigo cedem o direito à divulgação e publicação do material para comunidade acadêmica através de portal da Biblioteca e repositório institucional. Esta autorização permite sua utilização como base para novas pesquisas, caso haja adaptação do conteúdo é necessário atribuir o devido crédito de autoria.