AUTOMATIZAÇÃO DA CORREÇÃO DE PROVAS UTILIZANDO PROCESSAMENTO DIGITAL DE IMAGENS

João Marcos C. de Lima Costa*, Marcus Vinicius de Oliveira Brito*

* *Universidade Federal do Rio Grande do Norte*
Departamento de Engenharia Elétrica
Natal, RN, Brasil

Emails: jmcosta944@gmail.com, mvobrito@gmail.com

Abstract— The presented technique sequence will process an answer sheet's photo - belonging to a standard model - intending to obtain the essential informations contained on the sheet: student's registration number, the answers and score. All the procedure described was embedded into Android platform for smartphones through the application Kremlin for automating test correction.

Keywords— photo, answer sheet, Android, correction

Resumo— A sequência de técnicas apresentada irá processar a foto de um gabarito - pertencente a um modelo único - no intuito de obter as informações essenciais contidas na folha: matrícula do aluno, suas respostas e pontuação. Todo o procedimento descrito foi embarcado na plataforma Android para smartphones pelo aplicativo Kremlin para automatizar a correção de provas.

Palavras-chave— foto, gabarito, Android, correção

1 Processamento digital da foto

A aplicação responsável por processar a foto tirada pelo usuário foi, inicialmente, desenvolvida na linguagem de programação C++, fazendo uso da biblioteca OpenCV. Na execução da aplicação - via linha de comando do Ubuntu -, o usuário passa como parâmetro o nome do arquivo da foto, que está contida no mesmo diretório do programa.

A folha de respostas é padronizada no tamanho A4, contendo duas tabelas centralizadas: a superior serve para o aluno marcar os espaços correspondentes a sua matrícula, e a inferior serve para marcar as respostas. Existem legendas para orientar o aluno em ambas as tabelas: na matrícula, a primeira coluna indica que algarismo está sendo marcado (de 0 a 9), e o aluno deverá preencher a tabela da direita para esquerda, marcando apenas uma casa por coluna, de maneira análoga a própria escrita da matrícula; a tabela de respostas segue a lógica convencional, na qual a primeira coluna indica o número da questão (de 1 a 10), e a primeira linha indica a alternativa (de A a E).

Dispostos nas extremidades (próximos aos vértices), estão quatro quadrados totalmente pretos, com 1 centímetro de lado cada. O quadrado superior esquerdo dista de 17.5 centímetros do quadrado superior direito, e 26.5 centímetros do quadrado inferior esquerdo. O quarto quadrado (inferior direito) está a 17.5 centímetros do quadrado inferior esquerdo, e 26.5 centímetros abaixo do quadrado superior direito. Por fim, temos uma distribuição simétrica destas marcas.

1.1 Aquisição da imagem

A preparação da cena e o posicionamento correto da câmera condicionam o funcionamento correto do algoritmo. A câmera deve estar próxima da folha de modo que capture uma região ligeiramente interna as suas bordas laterais, de modo que não apareça o limite entre a borda e a superfície sobre a qual pousa a folha. Essa mesma estratégia se aplica a margem superior, fazendo que a foto só mostre a superfície na parte abaixo da margem inferior. Se houver interferência do fundo nas margens laterais e superior, a leitura será afetada, à não ser que este seja branco e não se perceba nenhum contorno ou sombra. Fica claro que essa configuração de posição só é possível com o smartphone paralelo à folha.

Quanto à iluminação, o usuário deve evitar sombras fortes na cena, e prezar por uma iluminação homogênia. Vale salientar que o flash da câmera não pode ser utilizado, mesmo que a luz sobre a folha esteja fraca.

1.2 Melhoramento

Uma vez que a foto tenha sido capturada corretamente, o algoritmo irá importá-la em tons de cinza com o comando

```
imread(argv[1],CV_LOAD_IMAGE_GRAYSCALE);
```

, visto que seria inadequado e desnecessário usar as informações de cores do modelo RGB. Depois dessa etapa, os processos de limiarização e borramento são realizados, um após o outro, 3 vezes

seguidas, utilizando os comandos

```
blur(image,image,Size(2,2));
threshold(image,image,0,255,
CV_THRESH_BINARY | CV_THRESH_OTSU);
```

, sendo o *blur* responsável por borrar a imagem e o *threshold* pela limiarização, utilizando o método de Otsu.

1.2.1 Método de Otsu

Limiarizar a imagem consiste em fixar um limite e substituir o valor do pixel em questão por 0 ou 255 (preto ou branco, respectivamente). Esses dois valores podem variar, e essa operação é do tipo:

$$f(x,y) = \begin{cases} 255 & \text{se } f(x,y) > limite \\ 0 & \text{se } f(x,y) < limite \end{cases}$$