Programa para validação de QR Codes em imagens pré-processadas

Bernardo P. Rieger, Caroline P. Rosa, Matheus E. Barkert , Victor F. de Quadros

Departamento de Engenharias, Arquitetura e Computação
Universidade de Santa Cruz do Sul (UNISC) – Santa Cruz do Sul – RS – Brasil

{barkert, bernardorieger, cprosa, victor.quadros}@mx2.unisc.br

Abstract. *This article aims to introduce the theme of image recognition/identification using neighborhood concepts. From the pre-processing of the image to its validation through QR Codes (Quick Response Code) recognition algorithms.*

Resumo. *Este artigo tem como finalidade introduzir o tema de reconhecimento/identificação de imagens utilizando conceitos de vizinhança. Abordando desde o pré-processamento da imagem até a sua validação através de algoritmos de reconhecimento de QR Codes (Quick Response Code).*

1. Introdução

Um Código QR, ou seja, um *Quick Response Code*, nada mais é do que um código de barras bidimensional inventado no Japão no ano de 1994 por uma subsidiária da Toyota chamada Denso Wave. Criado principalmente para otimizar a manufatura dos automóveis e suas peças. Atualmente, os *QR Code* são utilizados das mais diversas formas, principalmente para o compartilhamento de informações via celular, podendo ser para buscar o cardápio digital em um restaurante ou autenticar o login.

Nosso trabalho visa validar se a imagem pré-processada é de fato um Código QR válido por meio de três identificadores laterais, que são quadrados pretos com uma borda levemente distanciada, formados internamente por um pixel preto, um branco, três pretos, um branco e um preto. Esses quadrados estão localizados em três cantos do Código, sendo eles, ambos os cantos superiores e o esquerdo inferior.

Para o desenvolvimento, optamos por utilizar a linguagem de programação C, sem nenhuma biblioteca específica de computação gráfica. O principal objetivo do trabalho será simular uma situação real de leitura de um *QR Code* por meio de uma câmera de celular, sendo escolhidas três imagens para demonstrar capacidade de processamento, sendo elas uma com um código válido, outra com um código inválido e outra com a ausência de um Código QR.

Imagem 1. Foto do QR Code válido

Imagem 2. Imagem sem nenhum QR Code

Imagem 3. Imagem com o QR Code inválido

2. Pré-processamento das imagens

O pré-processamento de uma imagem consiste em transformá-la em algo mais apropriado para a aplicação, utilizando etapas como recorte da imagem, alteração do contraste para as cores e mudança na resolução.

Na imagem abaixo simulamos uma foto de Código QR invádo tirado a partir de um telefone celular, tentando ao máximo simular uma situação real.

Imagem 4. Imagem sem nenhum processamento

Após isso, o recorte da na imagem foi realizado. Em uma situação real, essa etapa seria realizada por meio de um quadrado na interface, para que o usuário encaixe o código QR na área marcada.

Imagem 5. Imagem após o recorte

A seguir, temos o histograma da imagem recortada. É perceptível que a imagem é bem definida em cores claras e escuras, por isso será aplicado um filtro de *threshold*

utilizando o valor 127, transformando todas as cores abaixo desse valor em preto e as acima em branco.

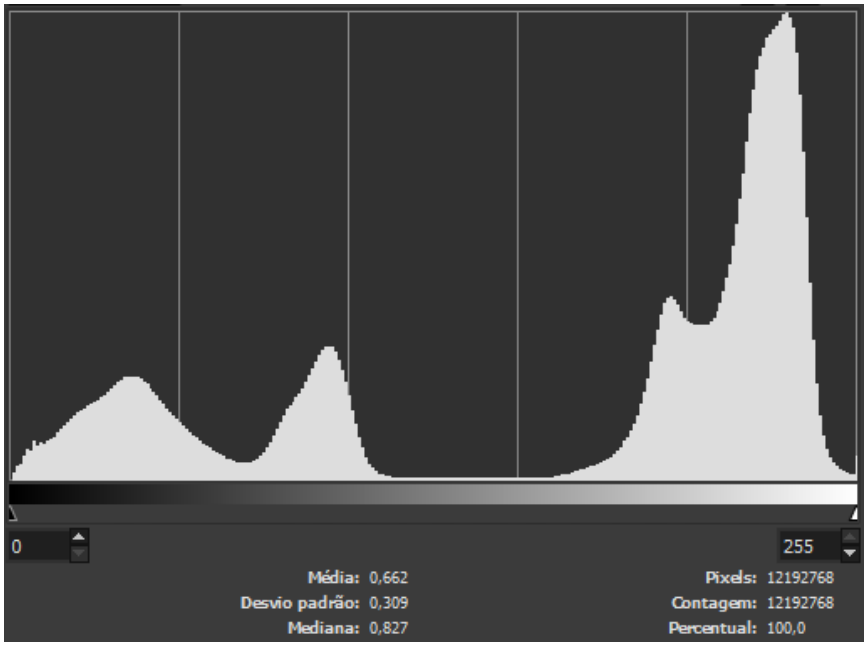

Imagem 6. Histograma obtido da imagem inicial

Assim, temos a imagem do código QR após aplicação do *threshold* de 127, com uma maior definição dos pixels pretos, mas ainda com alguns em tons de cinza.

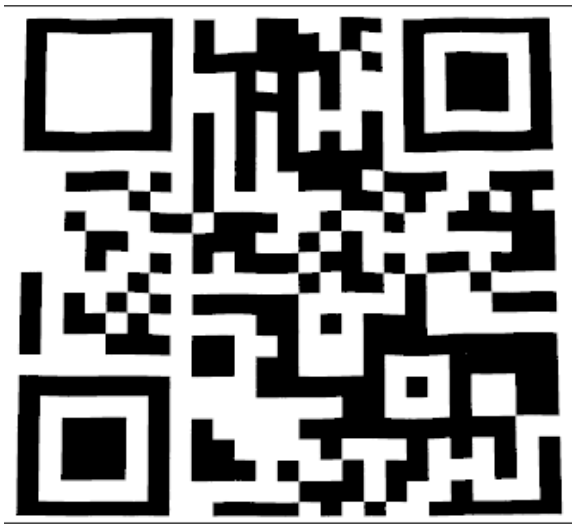

Imagem 7. Imagem após a aplicação de Threshold com valor de 127

A imagem ainda possui uma resolução muito grande para a análise ser realizada, demandando muito poder de processamento. Por isso, a reduzimos para uma resolução de aproximadamente 100 x 100 *pixels*.

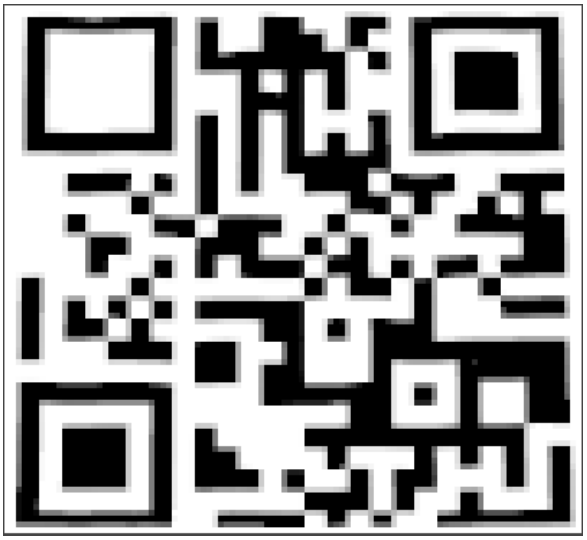

Imagem 8. Imagem após a redução de sua resolução para 100 x 100 pixels.

A última etapa do pré-processamento é a aplicação de mais um *threshold*, dessa vez utilizando o valor 250, para deixar apenas as cores muito claras como branco e partes pretas maiores, facilitando o processamento. Por fim, temos uma imagem com o pré-processamento finalizado e pronta para ser analisada pelo algoritmo.

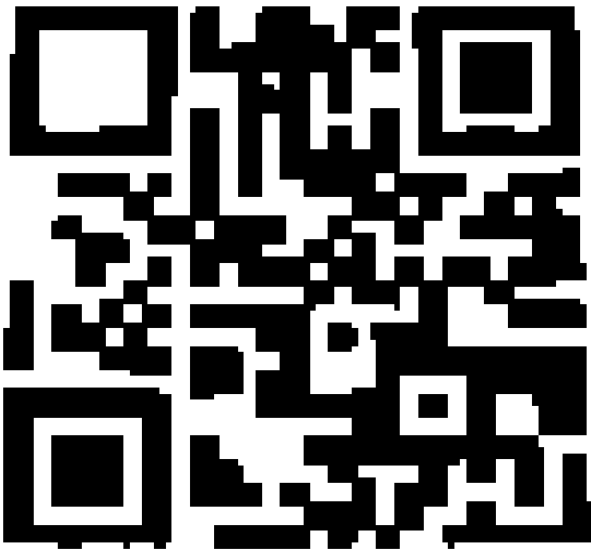

Imagem 9. Imagem após a aplicação de Threshold com valor de 250

3. O programa

```
#define TAM_COLUNA 150
#define TAM_LINHA 150
#define TAM_VALIDO 20

void carregaImagem(int imagem[][TAM_COLUNA]);
void printaImagem(int imagem[][TAM_COLUNA]);
void processaImagem(int imagem[][TAM_COLUNA]);
bool validacao(int imagem[][TAM_COLUNA], int col, int lin);
bool quadradoValida(int imagem[][TAM_COLUNA], int col, int lin);

int tam_lin, tam_col;

int main() {
    int imagem[TAM_LINHA][TAM_COLUNA];

    carregaImagem(imagem);

    printaImagem(imagem);

    processaImagem(imagem);

}
```

Imagem 10. Captura de tela das principais funções do código desenvolvido

O programa foi particionado em pequenas funções, que são chamadas na *main* ou internamente nas três funções principais. A“carregaImagem” é onde abre-se o arquivo e faz a sua leitura, salvando-o em uma matriz que será utilizada; “printaImagem” exibe no console a imagem que será processada; e “processaImagem” que faz todo o processamento e validação do QR Code, por meio de suas subfunções, sendo elas: “validacao”, que valida primariamente a borda do quadrado externo, e depois chama a “quadradoValida”, que tem a responsabilidade de verificar somente o quadrado interno, as quais utilizam o conceito de vizinhança.

Um exemplo da utilização do conceito de vizinhança no programa, é a validação de uma das arestas do quadrado externo, que inicia a verificação quando ele encontra o primeiro pixel preto da imagem, tendo em vista que a partir disso que ele começa a contornar o mesmo analisando de maneira inteligente e evitando desperdício computacional.

```
while(loop && valido) {
    if (imagem[lin][col] == 0) {
        count++;
    }
    if (imagem[lin][col] == 255) {
        loop = false;
        col -= 3;
    }
    col++;
}
colMedia = (colMedia + col)/2;
if (count < TAM_VALIDO) {
    valido = false;
}
loop = true;
count = 0;
```

Imagem 11. PrintScreen de um trecho de código utilizando vizinhança

4. Conclusões

Os tópicos abordados em aula foram de extrema importância para atingir o objetivo do trabalho proposto, com o pré-processamento ajustamos de maneira manual as imagens para que seus processamentos fossem mais eficientes, diminuindo suas resoluções e aplicando threshold para uniformizar possíveis pontos que possuíam cor, tornando assim as imagens pretas e brancas. Dessa forma, foi possível desenvolver um algoritmo que utiliza-se conceitos de vizinhança para validar as imagens que foram capturadas.

Em relação aos resultados obtidos com o programa desenvolvido, as validações de imagens com *QR Codes* válidos, inválidos e sem nenhum código foram assertivas com os testes realizados em relação à proposta do trabalho.

Referências

GOGONI, Ronaldo. O que é QR code? [Como ler e criar o seu]. 2018. Disponível em: https://tecnoblog.net/272214/como-criar-o-seu-proprio-qr-code/. Acesso em: 7 out. 2021.

CÓDIGO QR. 2021. Disponível em: https://pt.wikipedia.org/wiki/C%C3%B3digo_QR. Acesso em: 7 out. 2021.

ORIGIN of QR Codes and Why They’re on the Rise. 2021. Disponível em: https://wp.nyu.edu/dispatch/origin-of-qr-codes-and-why-theyre-on-the-rise/. Acesso em: 7 out. 2021.