Algoritmo para localização, registro e segmentação de QR Code sem enquadramento

Camilla Heleno, Marcelo da Silva Hounsell (colaborador),
Gilmário Barbosa dos Santos (colaborador), Alexandre Gonçalves Silva (orientador)

Departamento de Ciência da Computação
Universidade do Estado de Santa Catarina
Joinville, Santa Catarina, Brasil
Email: camiheleno@gmail.com, {marcelo,gilmario,alexandre}@joinville.udesc.br

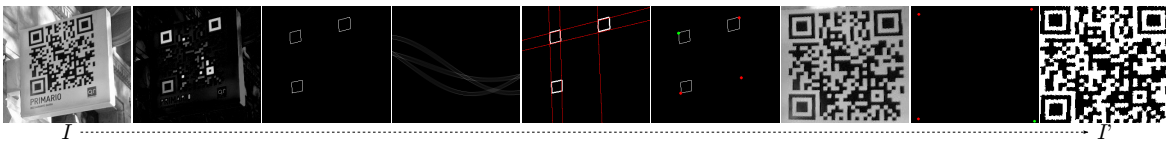


Figura 1. Evolução das etapas do método desenvolvido para registro e segmentação de imagens com QR Code não enquadrado.

***Resumo*—Os códigos de barras são tradicionais representações gráficas para coleta de informações. O QR Code, em especial, é bidimensional e vem se popularizando como método simples e eficiente para obtenção de números, textos e dados binários por visão computacional. Além da identificação de objetos e produtos na indústria, é livremente gerado e interpretado por softwares em diferentes plataformas, incluindo dispositivos móveis com câmera. Os resultados são satisfatórios quando há resolução e enquadramento adequados na aquisição de imagem. Este trabalho propõe um algoritmo para flexibilização destas características, permitindo que o QR Code possa ser automaticamente selecionado, registrado e segmentado em uma imagem fotográfica. Desta forma, o reconhecimento se torna mais ágil, sem necessidade de enquadramento preciso. Com este objetivo, é essencialmente utilizada uma sequência de fechamento de buracos para a detecção de marcadores referenciais, a transformada de Hough para verificação da inclinação e intersecção de retas, e a distorção geométrica de quatro pontos de amarração para registro da região de interesse. O sucesso de decodificação foi 40% superior com uso de imagens resultantes do método desenvolvido.**

***Abstract*—Bar codes are traditional graphical representations to collect information. The QR Code, in particular, is two-dimensional and has been more popular as simple and efficient method for obtaining numbers, texts and binary data by computer vision. Besides identifying objects and products in the industry, it is freely generated and interpreted by a set of software on different platforms, including mobile devices with camera. The results are satisfactory when there is appropriate resolution and guidelines in image acquisition. This paper proposes an algorithm for relaxation of these characteristics, allowing an automatic selection, registration and segmentation of QR Code in a photographic image. Thus, the recognition becomes more agile, without precise guidelines. For this purpose, closing holes is used to detection the reference markers, the Hough transform to check the slope and intersection of straight lines, and geometric distortion of four-point mooring for registration of the region of interest. The decoding success was 40% higher using resulting images of the developed method.**

***Keywords*-morfologia; QR Code; fechamento de buracos**

I. Introdução

Atualmente, diferentes tipos de códigos de barras são usados para armazenar, recuperar e gerenciar informações [1]. Eles permitem o rastreamento de produtos com eficiência e precisão superior à entrada manual de dados [2]. O QR Code, código de barras bidimensional utilizado neste trabalho, é uma representação gráfica de fácil decodificação, contendo expressiva densidade de informações nos sentidos vertical e horizontal. Além de sua utilização em larga escala em celulares [1], visando a disseminação de informações (por exemplo, registro de telefones e endereços em cartões de visita) e interface de campanhas publicitárias (por exemplo, *hyperlinks* de produtos e empresas), os códigos de "resposta rápida" (QR de "*Quick Response*") podem ser usados em atividades de controle de estoque e logística em geral. A utilização de técnicas de processamento de imagens, anteriores ao algoritmo de reconhecimento e interpretação dos dados do código, é importante na viabilização do uso de QR Codes em situações nas quais há chances de insucesso por interferência de características do ambiente ou da própria câmera que realiza a captura. Este trabalho propõe um algoritmo para automaticamente detectar, registrar (enquadrar) e segmentar um QR Code presente em um trecho de uma imagem real mais ampla captada pela câmera, admitindo variações de posição, orientação e perspectiva. Para isto, uma combinação de operações morfológicas e de segmentação, além de transformações geométricas, são utilizadas. O objetivo é ter uma imagem processada que possa ser mais facilmente interpretada (com menor taxa de falha) pelos algoritmos usuais de reconhecimento de QR Codes. A Figura 1 ilustra um resumo das etapas abordadas. Este documento está estruturado da seguinte forma: na Seção II, são apresentadas as principais definições de processamento de imagens, bem como uma breve discussão sobre a codificação em barras; na Seção III,

descreve-se o algoritmo para a operação de registro proposta; na Seção IV, são apresentados os resultados obtidos; e a Seção V é dedicada às conclusões do trabalho.

II. Definições preliminares

As definições de algumas das principais técnicas de processamento de imagens, para posterior aplicação no algoritmo desenvolvido, são apresentadas nesta seção.

A. Imagem digital

Uma imagem em níveis de cinza consiste em uma grade retangular de pixels em (x, y), ou seja, uma matriz de valores (normalmente inteiros sem sinal de 8 bits), representando intensidades luminosas (0 para preto, 255 para branco e valores intermediários para uma tonalidade de cinza). Define-se imagem binária quando se têm apenas duas intensidades, 0 ou 1. As imagens coloridas são formadas pela combinação de três componentes: R (vermelho), G (verde) e B (azul). O negativo de uma imagem I é igual a $I_{neg}(x, y) = L - I(x, y)$, sendo L o maior valor representável (255 para imagens não binárias de 8 bits e 1 para imagem binária), para cada pixel em coordenada $(x, y) \in \mathbb{N}^2$, no domínio $0 \leq x < H$ e $0 \leq y < W$, sendo H a altura (quantidade de linhas) e W a largura (quantidade de colunas) da matriz. A Figura 2 ilustra o sistema de coordenadas adotado, neste trabalho, para uma imagem I qualquer.

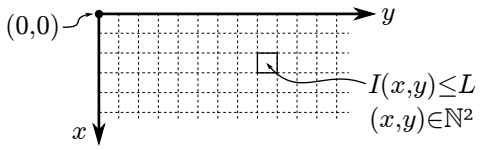


Figura 2. Origem e eixos adotados para representação de uma imagem I

B. Morfologia matemática

Morfologia matemática é uma ferramenta para a manipulação de componentes de imagens em função da representação e descrição da forma de regiões [2]. O processamento de informações geométricas e topológicas utiliza um conjunto completamente definido denominado *elemento estruturante*, contido ou não nos componentes [3]. Um elemento estruturante pode apresentar diversos desenhos e tamanhos, como mostra a Figura 3 (bolinhas brancas são iguais a 1, demais pontos são nulos ou iguais a 0, e o centro é indicado com o fundo cinza), dependendo das formas a explorar na imagem.

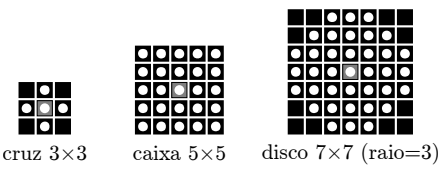


Figura 3. Exemplos de elementos estruturantes utilizados neste trabalho

1) Dilatação e erosão: No que se refere a imagens em níveis de cinza, a dilatação consiste no cálculo do máximo dos vizinhos a cada translação do elemento estruturante E (refletido). A erosão é o complemento da dilatação, ou o mínimo dos vizinhos a cada translação de E [4]. A operação de dilatação binária, por sua vez, é o conjunto de translações na qual há pelo menos um pixel não nulo de E com intersecção às partes não nulas da imagem de entrada e, na operação de erosão, a cada deslocamento do elemento estruturante sobre a imagem, obtém-se, como resultado, todo o conjunto E, se estiver totalmente contido nas partes não nulas da imagem de entrada.

2) Abertura e fechamento binários: A abertura binária é definida como a união de todas as translações de um elemento estruturante B contidas na imagem de entrada A, ou simplesmente a erosão de A por B, seguida de uma dilatação deste resultado por B. Por outro lado, o fechamento binário pode ser considerado a união de todas as translações de B que não estão em A, ou uma dilatação de A por B, seguida de uma erosão deste resultado por B [5] (a abertura e fechamento em níveis de cinza são determinados com base em dilatação e erosão também em níveis de cinza).

3) Reconstrução morfológica: A reconstrução binária de uma imagem consiste em, dentre várias regiões, selecionar apenas aquelas onde há intersecção com um marcador ou semente S, representada por um pixel ou conjunto de pixels [4], considerando o tipo de conectividade (forma como os pixels podem ser ligados diretamente com os pixels vizinhos [6]). O marcador especifica quais componentes da imagem de entrada serão extraídos. Na reconstrução de uma imagem em níveis de cinza, consideram-se todos os componentes da decomposição por limiares [7] que apresentam intersecção com os componentes, também decompostos por limiares, do marcador (agora definido por uma função) [8].

4) Fechamento de buracos: O fechamento de buracos baseia-se na reconstrução morfológica em níveis de cinza utilizando um marcador *frame*, no qual se atribui valor máximo (1 para imagem binária ou 255 para imagem em níveis de cinza) apenas à primeira e última linha e à primeira e última coluna. A ideia é reconstruir as regiões da imagem negativa (valor máximo menos o valor da intensidade, para cada pixel) da entrada que apresentam interface com os limites do domínio da imagem. Ao aplicar um novo negativo, tem-se a imagem original com supressão das regiões mais escuras em forma de ilhas circundadas totalmente por regiões claras. Esta técnica remove mínimos regionais que não estão conectados à borda da imagem [9]. A Figura 4 ilustra as etapas do fechamento de buracos. A imagem original binária (a) é negada em (b). Em seguida, ocorre a reconstrução dos componentes que possuem interface com a borda, por meio do marcador *frame* em (c) e, finalmente, o negativo deste último resultado em (d). Observa-se o desaparecimento de todas as ilhas pretas da imagem original.

(a) (b) (c) (d)

Figura 4. Etapas do fechamento de buracos: (a) imagem original; (b) negação de (a); (c) reconstrução dos componentes presentes nos limites do domínio; (d) negação de (c)

C. Segmentação

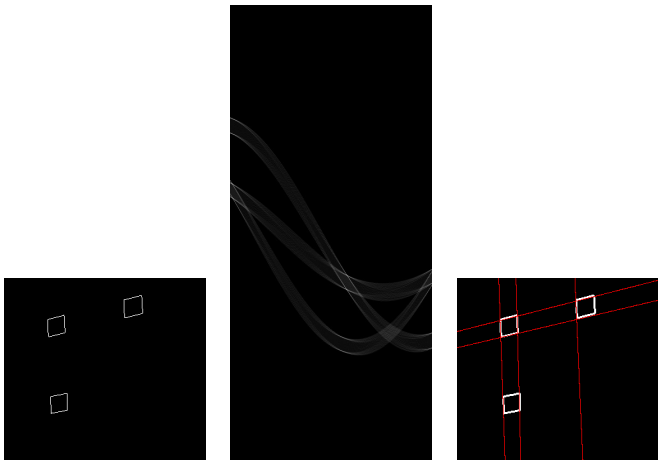
A segmentação consiste na subdivisão da imagem em suas partes ou objetos constituintes [2]. Seguem duas operações úteis neste trabalho, a limiarização e a transformada de Hough.

1) Limiarização: A limiarização ou binarização consiste na produção de uma imagem binária da classificação dos pixels de acordo com a especificação de um limiar. Ou seja, atribui-se 1 aos pixels com valor maior que o limiar e 0 aos iguais ou menores. A seleção de um valor correto de limiar é crucial para um bom resultado, podendo admitir um valor global, na qual um determinado limiar é aplicado sobre toda a imagem, ou local, com base em considerações de vizinhança de um ou mais pixels. Neste trabalho, optou-se pelo limiar aplicado sobre o topo de chapéu (*top-hat*) [9] gerado na subtração da imagem de seu fechamento de buracos. A Figura 5 detalha este resultado. Em (b), tem-se o fechamento de buracos da imagem original (a). Em seguida, tem-se o topo de chapéu (c), resultante da subtração de (b) por (a). Por fim, em (d), a imagem binária resultante da binarização de (c), usando um limiar igual a 35% da máxima intensidade de (c).

(a) (b) (c) (d)

Figura 5. Etapas da limiarização baseada no topo de chapéu do fechamento de buracos: (a) imagem original; (b) fechamento de buracos de (a); (c) subtração de (b) por (a); (d) binarização de (c)

2) Transformada de Hough: A partir de um número de pontos $p_i(x_i, y_i)$ na imagem, o objetivo da transformada de Hough é identificar se há um subconjunto destes pontos pertencente a uma determinada curva. Para o caso particular de segmentos de retas, necessário ao trabalho, o problema consiste em verificar se há pontos colineares [5]. Isto é possível por meio de um espaço discreto de parâmetros, funcionando como células acumuladoras para as retas formadas a cada dois pontos. Prefere-se a equação na forma polar (de modo a evitar indeterminação do coeficiente angular para retas paralelas ao eixo y), $\rho = x\cos\theta + y\text{sen}\theta$, com parâmetro ρ como distância da reta à origem $(0, 0)$ e θ o ângulo formado entre a reta perpendicular e o eixo x. Ao final do processo, as células com os maiores valores somados corresponderão aos parâmetros de retas desejadas. A Figura 6 exemplifica a transformação de Hough (b) de uma imagem binária (a). Algumas retas em (c) são determinadas a partir da seleção de alguns dos parâmetros mais acumulados em (b).



(a) (b) (c)

Figura 6. Exemplo da transformação de Hough para obtenção de retas: (a) imagem binária; (b) transformada de Hough de (a); (c) retas selecionadas (parâmetros mais acumulados) sobre (a)

D. Registro

Registro (enquadramento) é uma técnica de transformação geométrica que transfere um polígono, no caso deste trabalho, formado por quatro vértices de uma região da imagem, para outras posições no espaço, normalmente na forma retangular com os quatro pontos de amarração [2] agora alinhados com os eixos. São utilizados coeficientes para reagrupamento dos pixels dentro da região demarcada pelos pontos de amarração, obtidos pelo método de eliminação de Gauss-Jordan de resolução das equações bilineares [2]. A Figura 7 ilustra um exemplo desta operação.

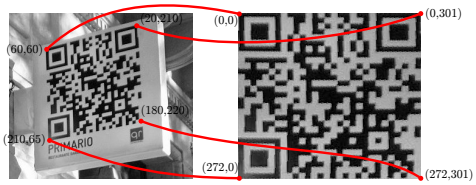


Figura 7. Exemplo de registro efetuado em uma imagem de 273×302

E. Códigos bidimensionais

Os códigos de barras unidimensionais tornaram-se populares devido à sua velocidade de leitura, precisão e características de funcionalidade superior a métodos manuais [10]. Porém, com o passar dos anos e aumento de sua utilização no mercado, a demanda por códigos com maior número de caracteres e impressos em menor espaço aumentou, limitando a adoção de códigos de barras convencionais. Os códigos de barras bidimensionais surgiram para atender a estas necessidades.

O QR Code é um código de barras bidimensional (vide Figura 4(a)) criado pela Denso-Wave em 1994. QR vem das iniciais de "*Quick Response*" (resposta rápida), e visa a decodificação em alta velocidade. Contém informações binárias (quadrados pretos ou brancos) em um área quadrada, podendo representar até 7089 caracteres numéricos, 4296 caracteres de dados alfanuméricos, 2953 bytes de código binário (8 bits) e 1817 caracteres japoneses. O processo de detecção de QR Codes consiste em cinco procedimentos, contando com um pré-processamento, detecção e definição dos três cantos principais do código (marcadores), detecção do quarto canto estimado e verificação do código para criar o tamanho normalizado e binarizado da imagem do código [11].

III. Algoritmo desenvolvido

Nesta seção, é descrita a sequência de processamentos de uma imagem contendo um QR Code, com o objetivo de encontrar as posições dos marcadores do código, definir a aplicação do registro com base nos quatro cantos determinados e criar uma imagem binária final melhor enquadrada para aumentar a taxa de sucesso na decodificação. As etapas do algoritmo são descritas, em detalhes (possibilitando reprodutibilidade), a seguir, sendo ilustradas na Figura 8.

1) Transformação da imagem (eventualmente colorida) para níveis de cinza, usando a luminância $Y(x,y) = 0,2989R(x,y) + 0,587G(x,y) + 0,114B(x,y)$ do modelo $YCbCr$ de cores [2];
2) Normalização, entre 0 e 255 (supondo apenas imagens de 8 bits), do passo 1;
3) Negativo do passo 2;
4) Fechamento de buracos do passo 3;
5) Subtração do passo 4 pelo passo 3;
6) Binarização do passo 5 com limiar equivalente a 35% (obtido por testes empíricos) do valor máximo dos pixels da imagem (garante certa adptabilidade);
7) Fechamento de buracos do passo 6;
8) Subtração do passo 7 pelo passo 6;
9) Remoção de componentes da imagem do passo 8 com área menor que 10 para remoção de pequenos elementos de tamanhos não representativos;
10) Abertura morfológica do passo 9 por um elemento estruturante caixa de tamanho 7×7 para manutenção de apenas regiões com espessura suficientemente pequena;
11) Dilatação do passo 10 por um elemento estruturante cruz de tamanho 3×3;
12) Intersecção do passo 6 com o passo 11;
13) Reconstrução morfológica do passo 6 tendo como marcador o passo 12;
14) Fechamento de buracos do passo 13;
15) Subtração do passo 14 pela erosão, por um elemento estruturante cruz de tamanho 3×3, do mesmo passo 14;
16) Aplicação da transformada de Hough sobre o passo 15;
17) Binarização do passo 16, com limiar equivalente a 70% do valor máximo dos pixels dessa imagem, para seleção dos parâmetros ρ e θ que caracterizem destacadas retas;
18) Dilatação do passo 17, por um elemento estruturante disco de raio 3, ou 7×7, de modo a unir eventuais células próximas;
19) Determinação dos centroides dos componentes do passo 18, ou seja, determinação da coordenada dada pela média dos x e y dos pixels pertencentes a cada componente;
20) Captura dos parâmetros ângulo (θ) e distância (ρ) da coordenada de cada centroide selecionado no passo 19;
21) Representação das retas obtidas;
22) Ordenação das retas do passo 20 primeiramente por ângulos (θ) e, em seguida, por distâncias (ρ), além da seleção das quatro retas mais próximas da origem $(0,0)$;
23) Cálculo dos valores do vértices referentes ao ponto superior esquerdo (do marcador do QR Code) por meio da intersecção da primeira e quarta retas da ordenação gerada pelo passo 22;
24) Dilatação da imagem do passo 15 por um elemento estruturante cruz de tamanho 3×3;
25) Decremento de y, a partir da última coluna, na primeira reta, até encontrar um pixel com valor 1 (branco) na imagem do passo 23, visando determinar x para o vértice superior direito (do marcador do QR Code);
26) Decremento de x, a partir da última linha, na terceira reta, até encontrar um pixel com valor 1 (branco) na imagem do passo 23, visando determinar y para o vértice inferior esquerdo (do marcador do QR Code);
27) Estimativa do valor do vértice inferior direito, por meio dos valores das distâncias entre os vértices determinados nos passos 25 e 26;
28) Registro da imagem a partir dos vértices obtidos nos passos 23 a 26, com folga de 10% em ambas direções (evita selecionar pontos internos ao código, devido à imprecisão das retas, quando na discretização do espaço de parâmetros), retornando a imagem contendo apenas o QR Code redimensionado;
29) Fechamento de buracos do passo 28;
30) Subtração do resultado do passo 29 pelo passo 28;
31) Binarização do passo 30 com limiar equivalente a 50% da média de valores da imagem;
32) Determinação dos três pontos das extremidades dos marcadores do código da imagem (de altura H e largura W), sendo o primeiro mais próximo da coordenada $(0,0)$, o segundo mais próximo de $(0, W-1)$ e o terceiro mais próximo de $(H-1, 0)$;
33) Determinação do quarto ponto extremo, considerando a maior linha e maior coluna pertencente ao código;
34) Novo registro final com os quatro pontos determinados nos passos 32 e 33.

IV. Resultados

Nesta seção, são apresentados os resultados obtidos pela execução do algoritmo desenvolvido para variadas imagens de entradas. Após a realização dos testes, obteve-se a identificação dos três marcadores do QR Code, os quatro cantos do código e seu correto registro para a maior parte

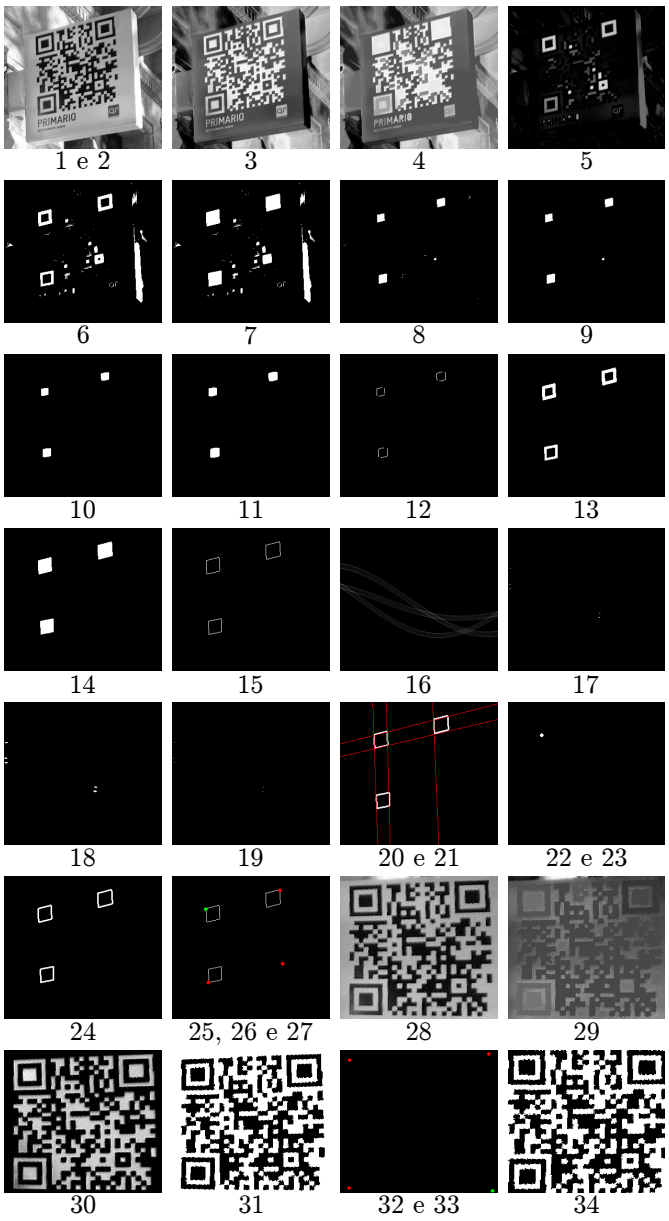


Figura 8. Passos do algoritmo desenvolvido

das imagens. As imagens nítidas que contêm apenas um código de barras apresentam resultado satisfatório. Para o reconhecimento, o ambiente no qual o código de barras estiver deve apresentar suficiente iluminação e não interferência de sombra ou brilho (embora este efeito seja contornado na segmentação com base no topo de chapéu do fechamento de buracos dos passos 4 a 6, e 29 a 31 do algoritmo). A Figura 9 mostra as imagens original e processada pelo algoritmo para cada um dos 20 testes realizados. Observam-se imperfeições do registro nas imagens I_9 (linhas extras indesejadas), I_{20} (manutenção de uma pequena distorção), além de I_{12} e I_{17} (falhas na binarização). No entanto, 16 resultados foram precisos, representando 75% de acerto nesta base, composta somente por imagens com um único QR Code.

Testes com estas 20 imagens, contendo um único QR Code, sem nenhum pré-processamento, em um primeiro momento, e com todos os passos do algoritmo desenvolvido, em um segundo momento, foram realizados. Utilizando uma *webcam* VGA (480×640) de uso doméstico e o reconhecimento *online* de QR Codes [12], no caso de não haver nenhuma etapa de processamento, foi possível o reconhecimento de 50% dos códigos, considerando o rotacionamento e reposicionamento da câmera para um melhor enquadramento. Com as imagens resultantes do algoritmo deste trabalho, houve reconhecimento correto do código de 70% das imagens.

Esta verificação foi repetida utilizando um celular Nokia N95 e, neste caso, também foram decodificadas, de forma correta, 50% da imagens originais e 70% das imagens processadas. As duas últimas colunas da Tabela I identificam as decodificações bem sucessidas ($\surd$) e as que apresentaram falhas ($\times$) no Nokia N95, considerando rotações e reposicionamentos de câmeras suficientes para estas afirmações antes (PRE) e após (POS) o processamento dos 34 passos do algoritmo. Ou seja, além da produção de uma imagem registrada e segmentada, o algoritmo possibilitou um índice 40% maior de sucesso do reconhecimento de conteúdo de um QR Code, considerando o software utilizado. Ainda na tabela, são apresentadas as dimensões $H \times W$ e o tempo de processamento t, em segundos (máquina AMD Phenom II X4 B93 2.8GHz 4GB), de cada imagem. A razão deste tempo pelo número de pixels de entrada é praticamente constante $(0,02 \pm 0,00)$, sinalizando um crescimento de tempo linear em função do tamanho da imagem.

Tabela I
TEMPO DE PROCESSAMENTO (t) E VERIFICAÇÃO DE DECODIFICAÇÃO ANTES (PRE) E APÓS (POS) O REGISTRO E SEGMENTAÇÃO

I_i	H_i	W_i	t_i (s)	$\frac{t_i}{H_i W_i}$ (ms)	PRE_i	POS_i
$i=1$	390	440	3,37±0,03	0,02±0,00	×	$\surd$
$i=2$	345	400	2,71±0,02	0,02±0,00	×	×
$i=3$	320	240	1,62±0,01	0,02±0,00	×	$\surd$
$i=4$	282	445	2,97±0,02	0,02±0,00	$\surd$	$\surd$
$i=5$	240	320	1,65±0,02	0,02±0,00	$\surd$	$\surd$
$i=6$	440	446	3,67±0,05	0,02±0,00	$\surd$	$\surd$
$i=7$	381	510	4,16±0,02	0,02±0,00	$\surd$	$\surd$
$i=8$	387	518	4,30±0,04	0,02±0,00	×	×
$i=9$	446	300	3,02±0,02	0,02±0,00	×	×
$i=10$	342	400	2,71±0,02	0,02±0,00	×	×
$i=11$	300	400	2,53±0,05	0,02±0,00	×	$\surd$
$i=12$	375	500	3,94±0,03	0,02±0,00	$\surd$	×
$i=13$	290	292	1,58±0,02	0,02±0,00	$\surd$	$\surd$
$i=14$	352	288	2,11±0,05	0,02±0,00	×	$\surd$
$i=15$	352	288	2,09±0,01	0,02±0,00	$\surd$	$\surd$
$i=16$	352	288	2,12±0,05	0,02±0,00	$\surd$	$\surd$
$i=17$	240	320	2,09±0,01	0,02±0,00	$\surd$	$\surd$
$i=18$	352	288	2,12±0,03	0,02±0,00	×	$\surd$
$i=19$	240	320	1,63±0,02	0,02±0,00	$\surd$	$\surd$
$i=20$	439	589	5,53±0,04	0,02±0,00	×	×

V. CONCLUSÃO

A solução proposta consiste em um pré-processamento de imagens contendo, cada uma, um único QR Code, com o intuito de possibilitar maior sucesso na interpretação de seu

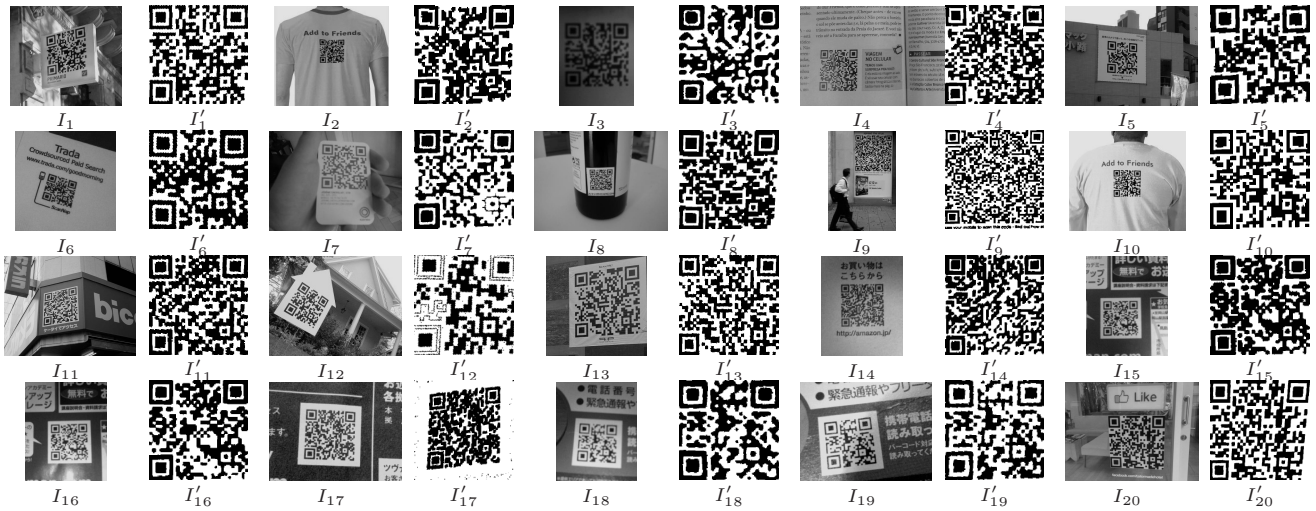


Figura 9. Resultados da aplicação do algoritmo sobre 20 imagens

conteúdo, sem necessidade de reposicionamentos de câmera até um perfeito enquadramento do código. A taxa de acerto que, neste caso, ocorre em ambiente com pouco controle de iluminação, pode ser prejudicada por interferências como sombras ou brilhos. Este problema, no entanto, é reduzido com uso do topo de chapéu do fechamento de buracos que se mostrou útil à detecção das ilhas pretas típicas dos códigos de barras. Aplicado duas vezes em sequência logo no início do algoritmo (passos 4 a 6; passos 7 e 8), praticamente isola os três marcadores do QR Code (vide Figura 8, passo 8). A identificação de pontos colineares pela transformada de Hough é decisiva na desconsideração de eventuais componentes indesejáveis persistentes no processo de binarização, já que normalmente são relativamente pequenos não caracterizando retas importantes. Embora com imprecisões, o registro foi obtido em todos os testes efetuados. Além deste resultado, importante, por exemplo, na recuperação automática de imagens com este tipo de conteúdo (CBIR–content-based image retrieval), houve ainda um ganho de 40% no total de imagens com sucesso de decodificação, se feita após as etapas de processamento sugeridas. As distorções, por exemplo, de códigos em um material flexível como tecidos, são naturalmente corrigidas após o registro, porém persiste a falha na decodificação (vide I_2 e I_{10} na Figura 9 e Tabela I). Em se tratando de superfícies rígidas, a imagem pode estar rotacionada e ser detectada tanto na forma original quanto processada (vide I_6). Ocorre que eventualmente há um tempo maior no acerto do enquadramento no primeiro caso. Por fim, havendo distorção de perspectiva (cisalhamento), o algoritmo promove ganho de sucesso na decodificação (vide I_1 e I_{11}).

Como trabalhos futuros, espera-se estender o algoritmo para imagens contendo mais de um código de barras, e aperfeiçoar a implementação no sentido de torná-la mais robusta (redução de imprecisões do registro e segmentação, e taxa maior de sucesso de decodificação) e rápida (viabilização da aplicação, em tempo real, para dispositivos com capacidade limitada de processamento). Novas etapas devem ser agregadas ao algoritmo para a construção da matriz de bits do código e eventual decodificação autônoma das informações ali contidas. Os resultados desta pesquisa devem integrar a a "inteligência de visão" de um robô móvel para que o mesmo possa efetuar tarefas a partir dos QR Codes dispostos no espaço.

AGRADECIMENTOS

Os autores agradecem o suporte PROBIC/UDESC oferecido à primeira autora.

REFERÊNCIAS

[1] J. Rouillard, "Contextual qr codes," in *International Multi-Conference on Computing in the Global Information Technology*, 2008, pp. 50–55.

[2] R. C. Gonzalez and R. E. Woods, *Processamento de imagens digitais*. Blucher, 2010.

[3] J. Facon, *Processamento e Análise de Imagens*, http://www.ppgia.pucpr.br/∼facon/IndexPrincipalBrCursos.htm (visitado em 06/2011).

[4] A. G. Silva and S. C. Felipussi, "ratoca - mouse by detection of markers using a camera of video in not controlled environment," *Latin America Transactions, IEEE (Revista IEEE America Latina)*, vol. 4, no. 6, pp. 443–448, dec. 2006.

[5] H. Pedrini and W. R. Schwartz, *Análise de Imagens Digitais: Princípios, Algoritmos e Aplicações*. Thomson Learning, 2007.

[6] Q. Wu, F. Merchant, and K. Castleman, *Microscope Image Processing*. Academic Press, 2008.

[7] A. G. Silva and R. A. Lotufo, "Efficient computation of new extinction values from extended component tree," *Pattern Recognition Letters*, vol. 32, no. 1, pp. 79–90, 2011.

[8] L. Vincent, "Morphological Grayscale Reconstruction in Image Analysis: Applications and Efficient Algorithms," *IEEE Transactions on Image Processing*, vol. 2, no. 2, pp. 176–201, April 1993.

[9] E. R. Dougherty and R. A. Lotufo, *Hands-on Morphological Image Processing*. SPIE, 2003.

[10] DENSO WAVE, *About 2D Code*, http://www.denso-wave.com/qrcode/aboutqr-e.html (visitado em 06/2011).

[11] E. Ohbuchi, H. Hanaizumi, and L. Hock, "Barcode readers using the camera device in mobile phones," in *International Conference on Cyberworlds*, 2004, pp. 260–265.

[12] F. Enzenhofer, *Mini QR Reader*, http://miniqr.com/reader.php (visitado em 06/2011).