Leitura Automática de Códigos de Barras em Imagens de Medidores de Energia Elétrica

Vinicius Rogério Araujo Silva*, João Dallyson Sousa de Almeida*,
Aristófanes Corrêa Silva*, Anselmo Cardoso de Paiva*, Geraldo Braz Júnior*,
Eliana Márcia Monteiro†, Bruno Rodrigues Froz†, Ítalo Fernandes Serra da Silva†, Cleman Garcia Mendonça†
*Núcleo de Computação Aplicada
Universidade Federal do Maranhão
São Luís, MA, Brasil
Emails: vinicius.ras@gmail.com, joao.dallyson@ufma.br, ari@dee.ufma.br, paiva@deinf.ufma.br, geraldo.braz@ufma.br
†Companhia Energética do Maranhão - CEMAR e
Centrais Elétricas do Pará S.A - CELPA
São Luís, MA, Brasil
Email: eliana.monteiro, bruno.froz,italo.silva,cleman.mendoca@cemar-ma.com.br

***Abstract*—Barcode technologies constitute an important base for identification processes automation. One of the practical applications for this is using barcodes to identify electrical measuring instruments. The existence of different codification standards makes the identification of barcodes a challenge, which requires the continuous development of new solutions that are efficient for their detection, regardless of the specific standard being detected. This article describes a method for segmentation and reading of barcodes in images of electrical measuring instruments which comply to the *Code 128* standard.**

***Resumo*—Tecnologias de códigos de barras constituem uma importante base para a automação de processos de identificação. Uma das aplicações práticas neste sentido é a utilização dos códigos de barras para a identificação de aparelhos medidores de energia. A existência de diferentes padrões de codificação torna a identificação dos códigos de barras um desafio que requer um desenvolvimento contínuo de novas soluções que sejam eficientes para sua detecção, independente do padrão utilizado. Este artigo descreve um método de segmentação e leitura de códigos de barras em imagens de medidores que seguem o padrão *Code 128*.**

I. Introdução

Medidores de energia elétrica são instrumentos eletromecânicos ou eletrônicos utilizados para averiguar o consumo de energia. No Brasil, as companhias distribuidoras de energia realizam leituras mensais com o intuito de avaliar o consumo nas respectivas unidades consumidoras. A Companhia Energética do Maranhão (CEMAR) e as Centrais Elétricas do Pará S.A (CELPA) utilizam códigos de barras localizados nos medidores de energia elétrica para identificar cada aparelho específico, aos quais podem ser associados dados sobre sua localização, data de substituição, última leitura efetuada, dentre outros.

O procedimento de leitura dos valores dos medidores de energia elétrica da CEMAR e CELPA, em geral, é realizado pelos leituristas em campo, que dispõem de dispositivos móveis com aplicativos específicos para o registro do consumo realizado nas unidades consumidoras e impressoras portáteis. Os leituristas normalmente registram o consumo da unidade no aplicativo e, caso não haja uma inconsistência, a fatura é gerada, impressa e entregue ao cliente no mesmo instante. Porém, em alguns casos podem haver inconsistências, como um consumo fora da média ou a inaptidão para realização da leitura pelo leiturista. Para estes casos, o aplicativo solicita a aquisição de uma imagem do medidor, capturando também sua localização geográfica. Os dados capturados são enviados para o setor de crítica da distribuidora, que utiliza a imagem como uma das formas de analisar inconsistências na leitura. A imagem obtida pelos leituristas deve conter os valores a serem analisados e o código de barras que pode ser utilizado para identificar o medidor de energia específico.

A análise das inconsistências encontradas — que são classificadas como "perdas não técnicas" — pode prevenir perdas financeiras expressivas para as empresas. Um estudo apresentado em [1] aponta que as perdas não técnicas representam um total de 44% da perda total anual para as empresas de distribuição de energia, o que equivale a cerca de 52 Terawatts/hora (TWh), acarretando um custo de aproximadamente R$ 5,5 bilhões — o que pode chegar a R$ 7,3 bilhões quando acrescido dos tributos legais (ICMS, PIS e Cofins).

O volume de imagens que precisam ser analisadas pelo setor de críticas das distribuidoras é grande. Juntas, a CEMAR e a CELPA possuem um volume médio de 30 mil imagens adquiridas e enviadas para análise antes de gerar uma fatura para o cliente. Para aprimorar todo este processo, uma ferramenta computacional denominada Sistema de Validação (SIVAL) está sendo desenvolvida. Seu objetivo é realizar a análise e validação do consumo de energia através da aquisição de imagens, inteligência computacional e uso de dispositivos móveis, minimizando erros humanos advindos do processo da leitura dos medidores de energia, detectando inconsistências e auxiliando nas suas correções. O presente trabalho se insere no contexto de um projeto de Pesquisa e Desenvolvimento

(P&D) contratado pela CEMAR/CELPA (ANEEL PD-00371-0029/2016), executado pelo Núcleo de Computação Aplicada (NCA) da Universidade Federal do Maranhão (UFMA). O SIVAL beneficia os setores de leitura, faturamento, cobrança, serviço de rede e recuperação de energia da CEMAR/CELPA.

O objetivo deste trabalho é propor um método de segmentação e leitura de códigos de barras em imagens de medidores, auxiliando na automatização da leitura dos códigos de barras durante o processo de verificação de inconsistências em medidores de energia, dado o elevado volume de inconsistências a ser analisado mensalmente pelas empresas de distribuição energética.

II. Trabalhos Relacionados

A identificação de regiões de códigos de barras em imagens é uma área que recentemente vem sendo amplamente explorada. Isto se deve em parte à crescente popularização do uso de dispositivos móveis como smartphones e tablets [2]. Tais dispositivos possuem câmeras que permitem fácil acesso a qualquer momento, o que os torna candidatos interessantes ao desenvolvimento de soluções que se utilizem de tecnologias de códigos de barras. Observa-se na literatura e na indústria o surgimento de diversas aplicações e *softwares* que permitem o reconhecimento e leitura de códigos de barras utilizando-se de imagens obtidas através das câmeras embutidas em tais dispositivos móveis.

A. Identificação dos Códigos de Barras

Parte dos métodos de identificação da região dos códigos de barras em imagens mais difundidos é baseada no fato de que existem grandes contrastes de intensidade de *pixels* entre as barras e espaços concentrados na região onde está situado o código, a exemplo de [3], [4] e [5]. Em geral, isto pode ser visto com relativa facilidade ao efetuar o cálculo dos gradientes contidos na imagem onde o código de barras está. Observa-se na literatura a utilização de diferentes métodos para o cálculo destes gradientes. Métodos baseados na utilização de filtragens de imagens do tipo *Bottom-Hat* (também nomeadas *Black-Hat* por alguns) foram utilizados em [3] e [5] para esta finalidade. Por outro lado, em [4] é utilizada outra proposta para a etapa de cálculo do gradiente, substituindo os filtros *Bottom-Hat* pela utilização de operadores de *Sobel*.

A partir da obtenção dos gradientes, é possível observar uma alta intensidade dos mesmos na região onde está localizado o código de barras. Grande parte dos métodos de identificação baseados no cálculo de gradientes utiliza-se deste conhecimento para realizar uma limiarização da imagem que representa o gradiente calculado, obtendo uma imagem binária com uma grande probabilidade de ocorrer um bom destaque nas áreas da imagem onde estão os códigos de barras. É nesta etapa que a maior parte dos métodos propostos executa uma combinação de diferentes operações, em geral envolvendo o uso de operadores morfológicos com finalidades diversas, para tentar aos poucos se aproximar de uma imagem representando uma máscara binária que pode ser utilizada para identificar com relativa precisão a região onde um ou mais códigos de barras se situam na imagem, permitindo o avanço para a etapa seguinte: a leitura do código de barras.

A metodologia descrita no presente trabalho baseia-se em uma combinação dos métodos de identificação dos códigos de barras a partir da utilização dos gradientes e a utilização de conhecimentos específicos acerca da composição do código de barras *Code 128* para tentar identificar e extrair as barras das imagens.

B. Leitura dos Códigos de Barras

Conforme apontado em [6], a etapa de leitura do código de barras contido numa imagem apresenta dois grandes desafios decorrentes das características de obtenção da mesma: o ruído e a deformação da imagem. O ruído é decorrente de problemas como: as características do dispositivo utilizado para capturar a imagem (qualidade, resolução de captura e sensibilidade à luz); as propriedades do material onde o código de barras está impresso, como a qualidade do papel, reflectância do metal ou rugosidade da superfície; as características do ambiente, como a iluminação, presença de poeira, neblina, dentre outros fatores. Já a deformação em geral é proporcionada pelas propriedades das lentes da câmera, e pela posição e orientação do código de barras em relação ao dispositivo de captura. Após a identificação da região onde o código de barras está localizado na imagem, é necessário implementar métodos de leitura que possam lidar com os ruídos e as deformações que estão presentes na mesma.

Em [7], o problema da leitura do código de barras é solucionado através de uma abordagem baseada na utilização de características obtidas da matriz de acumulação gerada a partir da transformada de Hough.

Já em [4] a questão da decodificação é solucionada através de um algoritmo baseado na simulação do funcionamento de um equipamento de leitura de códigos de barras ("*scanner*"), utilizando-se de uma linha perpendicular às barras do código para obter amostras sucessivas da imagem nesta linha. Algoritmos deste tipo são denominados "algoritmos de *scanline*". Estes algoritmos buscam identificar se cada amostra obtida ao longo da linha de leitura corresponde a um espaço ou uma barra, medindo os tamanhos destes elementos e utilizando-se desta informação para tentar efetuar a leitura correta do código.

Os métodos de leitura de códigos de barras citados até agora são baseados no uso de imagens binárias obtidas a partir de um procedimento de limiarização da imagem do código. Um terceiro método de leitura de códigos de barras que pode ser citado é proposto por [2], que buscam solucionar a leitura dos códigos de barras utilizando-se de uma imagem em nível de cinza do código ao invés de uma imagem binária. Neste método também é utilizada uma abordagem do tipo *scanline*, e o padrão gerado pelas amostras obtidas da imagem do código de barras é comparado com o padrão esperado para cada símbolo possível do esquema de codificação que está sendo lido para tentar achar aquele que mais se aproxima do padrão que foi lido. Esta comparação feita entre os padrões tenta "deformar" o padrão obtido de uma forma que ele consiga de aproximar ao máximo do padrão correspondente a um

dos símbolos possíveis do código de barras que está sendo lido. Assim é possível selecionar o símbolo que tem maior probabilidade de estar posicionado em cada local específico do código de barras da imagem.

A metodologia descrita no presente trabalho combina características das metodologias propostas em [4] e [2], além de conhecimentos específicos sobre a composição do esquema *Code 128*, utilizando imagens binárias e tentando efetuar uma leitura baseada em uma medida de similaridade entre os diversos padrões de tamanhos de símbolos do esquema.

III. Método Proposto

O método proposto neste estudo está organizado em três etapas ilustradas na Figura 1. Inicialmente, realiza-se a aquisição da imagem da região dos códigos de barras. Em seguida, segmenta-se os códigos de barras e por último, realiza-se a leitura dos códigos de barras. As etapas de segmentação e leitura dos códigos de barras propostas combinam ideias propostas na literatura, objetivando a maximização do quantitativo de resultados positivos em um banco de dados de imagens previamente adquirido.

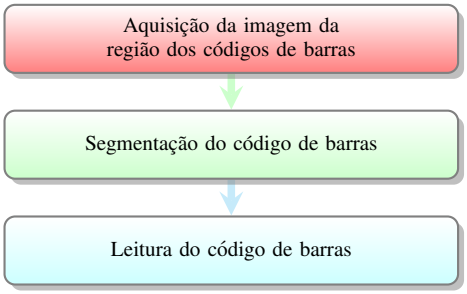


Figura 1. Etapas do método de leitura proposto.

A. Base de Imagens

A base de imagens utilizada possui 1024 arquivos de imagens em formato JPEG, todas contendo códigos de barras extraídos de fotos de medidores de energia, utilizando um padrão de codificação popular para códigos de barras denominado *Code 128*. Cada imagem apresenta apenas um único código de barras. Todos os códigos de barras contidos nas imagens apresentam seus respectivos textos decodificados escritos abaixo ou acima da região onde o código de barras está situado em cada imagem. Os textos representados pelos códigos de barras do banco de dados são constituídos por um conjunto de caracteres contendo apenas números, espaços e hifens. Adicionalmente, os nomes de todos os arquivos contendo os códigos de barras foram modificados e refletem exatamente o texto correspondente ao código contido na imagem. Esta característica é importante porque permite confirmar se a leitura de cada código de barras contido em cada imagem do banco de dados foi correta.

B. Segmentação dos Códigos de Barras

Primeiramente, uma etapa de pré-processamento da imagem é necessária. O objetivo desta etapa é obter uma imagem binária onde o código de barras seja preservado e destacado. Sua localização na imagem então será refinada nas etapas seguintes, até a obtenção de uma máscara correspondente à região onde se situa o código na imagem.

A obtenção da imagem binária durante a etapa de pré-processamento é feita através do cálculo de gradientes da imagem original, convertida em níveis de cinza. Este cálculo do gradiente usa operadores de Sobel para calcular os gradientes horizontais, verticais e diagonais das imagens contendo os códigos. No presente trabalho, tiramos vantagem do fato de que todas as imagens do banco de dados possuem códigos com barras relativamente alinhadas ao eixo vertical da imagem. Esse alinhamento com o eixo vertical nos permite verificar que os gradientes horizontais da região de interesse são muito mais intensos do que os gradientes verticais. Assim, optamos por utilizar operadores de Sobel apenas para o cálculo dos gradientes horizontais, desconsiderando os demais gradientes para fins de obtenção da imagem binária inicial do processo de segmentação. Um passo importante antes do cálculo dos gradientes, é utilizar um filtro Gaussiano para minimizar os ruídos da imagem, que podem afetar negativamente a qualidade dos gradientes obtidos para a segmentação dos códigos de barras ([5] e [4]). Em seguida é realizada a limiarização da imagem do gradiente para obter a imagem binária. O limiar definido para este procedimento é uma porcentagem do valor máximo de intensidade presente nos gradientes calculados. A Figura 2 ilustra os procedimentos para a obtenção da imagem binária.

Do procedimento para obtenção da imagem binária aqui mencionado, surgem três dos parâmetros de execução que afetam significativamente o resultado final da segmentação: o tamanho do *kernel* de convolução utilizado para aplicar o filtro Gaussiano antes do cálculo de gradientes, o parâmetro sigma deste mesmo filtro e a porcentagem do valor máximo dos gradientes calculados a ser utilizada para o cálculo do limiar utilizado para a limiarização da imagem, com o objetivo de transformá-la em uma imagem binária.

A imagem binária gerada tem alta probabilidade de conter o código de barras da imagem original. No entanto, a imagem também irá conter vários *pixels* e ruídos indesejáveis, que não fazem parte da nossa região de interesse. Os próximos passos da segmentação procuram aplicar diversos procedimentos com a finalidade de filtrar e diminuir cada vez mais a quantidade de *pixels* contidos na imagem binária, mantendo apenas aqueles *pixels* que pertencem à região da imagem onde o código de barras está situado.

O primeiro passo para filtrar *pixels* indesejados é baseado no teste de proximidade entre *pixels* realizado utilizando o algoritmo proposto em [5]. Percorremos a imagem binária linha a linha, achando todos os intervalos de colunas que contém *pixels* brancos contidos em cada linha. Tomamos uma distância que será considerada a distância máxima entre

(a) Original

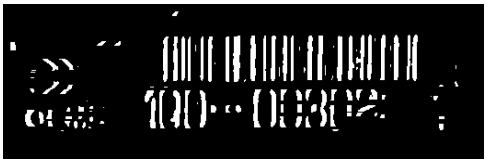
(b) Filtro Gaussiano

(c) Gradiente X

(d) Limiarização

Figura 2. Obtenção da imagem binária inicial.

duas barras adjacentes do código. Comparamos cada tripla de intervalos adjacentes de *pixels* brancos, verificando se as distâncias entre esses intervalos são menores que a distância máxima que definimos e eliminando os *pixels* brancos que não obedecerem a esta "regra de proximidade". Repetimos o teste diversas vezes até que todos os *pixels* brancos da imagem obedeçam à regra. Efetivamente, este passo remove *pixels* brancos da imagem binária que estão longe de outros *pixels* brancos contidos na mesma linha que eles, o que elimina muitos pontos avulsos que não têm chances de pertencer ao código de barras. Adicionalmente, estamos verificando se estes *pixels* brancos podem pertencer a uma zona com pelo menos três barras brancas. O procedimento é executado de duas maneiras: a primeira é feita da esquerda para a direita na imagem eliminando os *pixels* brancos mais à esquerda que não satisfazem a condição, e a segunda é realizada em sentido oposto e eliminando *pixels* à direita. Isso é feito porque na implementação do algoritmo como foi concebido, a comparação das barras de três em três pode eliminar as duas primeiras e as duas últimas barras do código de barras. No final das contas, teremos duas imagens binárias que apresentam muito menos ruídos e que contêm as barras centrais do código de barras. Combinamos as duas máscaras através de uma operação lógica *AND*, gerando uma imagem binária com a interseção das máscaras, conforme ilustrado na Figura 3. É importante notar que neste primeiro procedimento de filtragem dos *pixels* da imagem binária original, nós introduzimos um novo parâmetro de execução do nosso método: a distância máxima a ser considerada entre duas barras adjacentes do código de barras.



(a) Máscara de Proximidade Esquerda-Direita

(b) Máscara de Proximidade Direita-Esquerda

(c) Máscara de Proximidade Combinada

Figura 3. Obtenção da máscara de proximidade.

Sabendo que os *pixels* remanescentes obedecem a uma distância mínima de adjacência entre barras, podemos iniciar um teste baseado no conhecimento acerca do padrão de codificação *Code 128*. Neste padrão, um símbolo é representado por exatamente seis elementos: três barras pretas e três espaços brancos. Assim podemos percorrer a imagem novamente linha a linha descartando os intervalos de *pixels* brancos que não fazem parte de sequências de intervalos brancos e pretos intercalados contendo seis elementos. Este procedimento é executado, novamente, até que restem apenas os intervalos de *pixels* que obedeçam à regra. Obtém-se uma imagem binária conforme ilustrado na Figura 4 contendo o código de barras com poucos ruídos, estes mais fáceis de serem filtrados pelos passos seguintes.

Figura 4. Máscara de Sequências.

O último passo de filtragem de *pixels* indesejados tem como finalidade remover os pequenos ruídos restantes. Neste ponto, observamos que as barras no geral são formadas por grandes regiões verticais e contínuas. Podemos remover então as regiões consideradas muito pequenas, como proposto em [3]. No presente trabalho, optou-se pela introdução de um novo parâmetro de execução referente ao número de *pixels* mínimo

que uma região deve conter para ser mantido na imagem. A Figura 5 ilustra o resultado da remoção das pequenas regiões.

Figura 5. Remoção de regiões pequenas.

Após as etapas de filtragem, iniciam-se as etapas de expansão do algoritmo de segmentação. Nestas etapas, pretendemos recuperar informações que foram perdidas e efetivamente obter a área do código de barras na imagem. Para isso, o primeiro passo é recuperar as duas barras iniciais e duas barras finais do código de barras, que foram excluídas da imagem binária pelas etapas anteriores. Isto é feito percorrendo cada linha da imagem, novamente buscando os intervalos de *pixels* brancos em cada linha. Ao encontrar o intervalo de *pixels* brancos mais à esquerda de uma linha, recuperamos os dois intervalos de *pixels* brancos mais à esquerda utilizando imagem binária original que tínhamos no início do processo, efetivamente recuperando as duas barras mais à esquerda do código de barras ao final deste processo. Executamos um procedimento análogo para recuperar as barras mais à direita do código e assim teremos na imagem binária resultante o código de barras completo, e mais algumas possíveis regiões formadas por elementos semelhantes a barras verticais, como partes de números, por exemplo. O resultado é ilustrado na Figura 6.

Figura 6. Recuperação de barras laterais.

O último passo da segmentação é determinar exatamente a região onde o código de barras tem a maior probabilidade de estar, obtendo a máscara que contém toda sua extensão. Com esta finalidade, pintamos de branco o intervalo do primeiro até o último *pixel* branco de cada linha. Assim teremos uma imagem binária contendo poucas e grandes regiões contíguas. Tomaremos a maior destas regiões como o nosso resultado, conforme indicado na Figura 7.

O resultado final da segmentação é obtido tomando-se um retângulo que contém a região do código de barras. A Figura 8 ilustra uma imagem do banco de dados e o resultado da segmentação feita pelo método descrito no presente trabalho.

C. Leitura do Código de Barras

O algoritmo de leitura do código de barras implementado no presente trabalho é uma versão simplificada da decodificação através de gabaritos deformáveis proposta por

(a) Intervalos ligados.

(b) Maior região encontrada.

Figura 7. Obtenção da máscara final do código de barras.

(a) Imagem inicial.

(b) Imagem segmentada.

Figura 8. Resultado da segmentação.

[2], em combinação com a ideia do algoritmo básico de leitura descrito por [7]. A leitura é realizada através de uma *scanline* horizontal na imagem, obtendo-se um padrão de tamanhos de barras e espaços que é gerado pelas sucessivas amostras das intensidades de *pixels* da imagem binária. O padrão obtido é então comparado com um conjunto de gabaritos pré-definidos de padrões de tamanhos que representam cada símbolo do código de barras. A comparação feita entre o padrão e os gabaritos possíveis leva em consideração que o padrão obtido da leitura do código de barras é afetado pelas características de ruído e deformação da imagem.

A implementação é voltada para a leitura dos códigos de barras que obedecem ao padrão *Code 128*, por isso é importante compreender como funciona este esquema de codificação. Nele, um símbolo é representado por um conjunto de barras e espaços intercalados, totalizando seis elementos, dos quais três são barras e três são espaços. Um símbolo sempre é iniciado com uma barra (*pixel* preto) e sempre termina em um espaço (*pixel* branco). Cada barra e cada espaço possuem um tamanho que varia de uma a quatro unidades. A primeira barra do código de barras é a que possibilita ao leitor do código calcular qual o tamanho básico de uma unidade, em *pixels*. Esta primeira barra corresponde a duas unidades de tamanho das barras.

Para implementar o algoritmo de leitura, então, inicialmente

obtemos a imagem em níveis de cinza correspondente à região de interesse segmentada, onde o código de barras está situado. Em seguida efetuamos uma limiarização desta região com a finalidade de transformar a mesma em uma imagem binária, utilizando um limiar igual à média entre os níveis de cinza da região.

Percorremos em seguida todas as linhas da imagem binária, tentando ler o código de barras a partir dos tamanhos dos intervalos de pretos (barras) e brancos (espaços) da imagem em cada linha. A cada seis tamanhos lidos, teremos três barras e três espaços, que correspondem a um símbolo do código de barras. Procuramos então em uma tabela de gabaritos de tamanhos de símbolos do esquema *Code 128* qual é o símbolo que mais se assemelha ao padrão obtido da imagem.

O padrão obtido da imagem tem o tamanho dado em *pixels*, ao passo que os gabaritos de tamanhos fornecidos pela tabela de símbolos do esquema *Code 128* são dados em tamanhos variando de uma a quatro unidades para cada barra e espaço. Por isso precisamos efetuar a conversão dos tamanhos em *pixels* para tamanhos compatíveis com os gabaritos de tamanhos da tabela *Code 128*, permitindo a comparação entre eles para determinar seu grau de similaridade. Isso é feito dividindo-se o tamanho total em *pixels* dos seis intervalos lidos pelo somatório dos tamanhos do símbolo do esquema *Code 128* com o qual estamos comparando. Assim é possível comparar o padrão obtido com os gabaritos de todos os símbolos do esquema e descobrir qual é o símbolo que tem o maior grau de similaridade com o padrão lido.

O final da leitura ocorre com sucesso caso em alguma das linhas da imagem seja encontrado um caractere especial de parada definido pelo padrão *Code 128*. Neste momento teremos uma série de símbolos lidos, que podemos interpretar e validar através de algoritmos próprios e tabelas de valores definidos pelo padrão *Code 128*.

IV. Resultados e Discussão

Uma solução automatizada para calcular os melhores parâmetros de execução do método de segmentação do código de barras foi inicialmente desenvolvida. O objetivo desta solução é tentar várias combinações para os valores de parâmetros para maximizar a taxa de resultados positivos para as leituras feitas após as segmentações de imagens no banco de dados. Os parâmetros de execução que obtiveram resultado máximo são dados na Tabela I.

Tabela I
Parâmetros de Execução Utilizados.

Tamanho do *kernel* do filtro Gaussiano	7 *pixels*
Sigma do filtro Gaussiano	2.3
Limiar utilizado para o gradiente	10% do valor máximo
Mínimo de *pixels* na região de uma barra	9
Máxima distância entre barras adjacentes	18 *pixels*

Um resultado do método de segmentação proposto pode ser classificado como negativo quando o método não encontra nenhuma área de código de barras na imagem. Resultados podem ser classificados como verdadeiros-positivos, quando a região encontrada contém todas as barras do código e nenhum outro elemento, como imagens da marca da CEMAR/CELPA e caracteres alfanuméricos. Os resultados que possuem uma pequena parte do fundo da superfície onde o código de barras está situado também são considerados verdadeiros-positivos. Por fim, os demais resultados serão considerados falsos-positivos, o que efetivamente indica que o método proposto encontrou uma região que não corresponde ao código de barras da imagem, ou encontrou uma região parcial do código de barras, que não contém todas as barras do código.

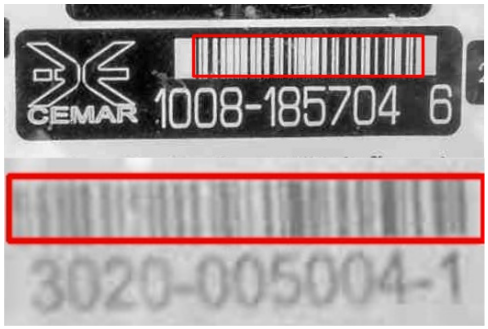
A Figura 9 demonstra exemplos de imagens do banco de dados que se enquadram em cada tipo de classificação explanada. Em 9a podemos observar dois exemplos de verdadeiros-positivos detectados pelo método de segmentação. Já em 9b observamos dois exemplos de falsos-positivos, em que uma das imagens apresenta uma segmentação parcial do código de barras e a outra apresenta uma segmentação falha que detectou os dígitos do número do código. Finalmente, em 9c, podemos observar duas imagens de resultados negativos. Nestas últimas, fica clara a grande dificuldade presente na segmentação dos códigos de barras dos medidores elétricos, dado o desgaste natural que pode ocorrer na região de interesse e às características de qualidade da imagem.

A Tabela II lista os resultados observados utilizando o método de segmentação para o banco de dados do presente trabalho.

Tabela II
Resultados obtidos no método de segmentação proposto.

Tipos de Resultados	**Quantidade Obtida**
Verdadeiros-positivos	831
Falsos-positivos	186
Negativos	7

Com a finalidade de obter parâmetros comparativos para os resultados, foi utilizada uma biblioteca popular especializada na leitura de códigos de barras denominada *ZXing*. Uma bateria de três testes foi executada sobre o banco de dados de imagens utilizado neste trabalho. No primeiro teste, utilizamos apenas os algoritmos de segmentação e leitura propostos. O objetivo foi avaliar os resultados de ambos os algoritmos propostos funcionando de maneira integrada, em comparação com os resultados obtidos pelos algoritmos de segmentação e leitura da biblioteca *ZXing*. A taxa de acertos nas leituras foi de 19,63%. No segundo teste, o algoritmo de segmentação proposto foi utilizado para obter as imagens dos códigos de barras, que foram em seguidas encaminhadas para leitura pela biblioteca *ZXing*. O objetivo deste segundo teste era avaliar apenas o algoritmo de segmentação de códigos de barras, em comparação com o algoritmo de segmentação da biblioteca *ZXing*. Sua taxa de acertos foi de 22,66%. No terceiro teste, a biblioteca *ZXing* processou todas as imagens do banco de dados sozinha. O objetivo deste teste era servir como um



(a) Verdadeiros-positivos.

(b) Falsos-positivos.

(c) Negativos.

Figura 9. Exemplos de resultados da segmentação.

comparativo para ser utilizado nos dois primeiros testes. A taxa de acertos para o último teste foi de 25,00%.

V. Conclusão

Neste trabalho, descrevemos alguns dos métodos recentes propostos para a segmentação e decodificação de imagens de códigos de barras. Em seguida, descrevemos uma adaptação simplificada de uma combinação entre métodos que resolvem cada um dos dois problemas mencionados na tentativa de encontrar uma solução para a leitura de códigos de barras de medidores de energia das companhias de distribuição de energia CEMAR e CELPA.

A partir dos testes executados sobre a implementação proposta neste trabalho pudemos perceber que a biblioteca *ZXing* apresentou melhores resultados em relação ao método de leitura de códigos de barra aqui descritos, apesar de relativamente próximos. O resultado foi o esperado, pois os algoritmos descritos no presente trabalho são bastante sensíveis a ruídos e deformações na imagem, visto que usam um modelo de similaridade bastante simples para efetuar a decisão de quais símbolos são lidos do código de barras extraído da imagem, o que pode acarretar em uma alta taxa de resultados negativos. Em contrapartida, observou-se um bom desempenho e uma interessante eficiência no método proposto para segmentação de imagens contendo códigos de barras, apresentando uma alta taxa de resultados classificados como verdadeiros-positivos.

Oferecer novos métodos para a resolução dos problemas de segmentação e decodificação de códigos de barras é algo importante, a medida que esta é uma tecnologia que se tornou ubíqua no mundo em que vivemos e de extrema relevância para nos mais diversos ambientes e aplicações.

Agradecimentos

Os autores agradecem a CEMAR/CELPA pelo suporte financeiro disponibilizado através do projeto ANEEL PD-00371-0029/2016.

Referências

[1] R. Vidinich and G. Nery, "Pesquisa e desenvolvimento contra o furto de energia," *Revista Pesquisa e Desenvolvimento da ANEEL - P&D*, p. 15, 2009.

[2] O. Gallo and R. Manduchi, "Reading 1d barcodes with mobile phones using deformable templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1834–1843, Sept 2011.

[3] X. J. Juett and X. Qi, "Barcode localization using bottom-hat filter," *NSF Research Experience for Undergraduates*, vol. 19, 2005.

[4] T. R. Tuinstra, "Reading barcodes from digital imagery," Ph.D. dissertation, PhD thesis, Cedarville University, 2006.

[5] M. Katona and L. G. Nyúl, "A novel method for accurate and efficient barcode detection with morphological operations," in *2012 Eighth International Conference on Signal Image Technology and Internet Based Systems*, Nov 2012, pp. 307–314.

[6] T. Pavlidis, J. Swartz, and Y. P. Wang, "Fundamentals of bar code information theory," *Computer*, vol. 23, no. 4, pp. 74–86, April 1990.

[7] R. Muniz, L. Junco, and A. Otero, "A robust software barcode reader using the hough transform," in *Proceedings 1999 International Conference on Information Intelligence and Systems (Cat. No.PR00446)*, 1999, pp. 313–319.