SISTEMA DE RECONHECIMENTO DA GRAFIA BRAILLE PARA A LÍNGUA PORTUGUESA (PT-BR)

Jackson Moreira Oliveira[1]

Eixo: Tecnologia Assistiva e Educação Especial
Comunicação oral

RESUMO

A sociedade tem dificuldades para realizar a comunicação com pessoas cegas, através da grafia Braille. Neste contexto o sistema a ser projetado vem ajudar as pessoas com visão / baixa visão / cegos, a serem capazes de compreender o Braille sem qualquer conhecimento, através da detecção a partir de imagens, onde os pontos são processados um a um utilizando um algoritmo especial que separa os pontos em caracteres, a fim de traduzi-los. A metodologia desenvolvida foi capaz de capturar imagem e processá-las.

Palavras-Chaves

Braille, imagens, detecção, traduzi.

INTRODUÇÃO

O Sistema Braille é um processo de leitura e escrita em relevo, com base em 64 (sessenta e quatro) símbolos resultantes da combinação de 6 (seis) pontos, dispostos em duas colunas de 3 (três) pontos cada. É também denominado Código Braille [6]. Pode-se fazer a representação tanto de letras, como algarismos e sinais de pontuação. Ele é utilizado por pessoas cegas ou com baixa visão, e a leitura é feita da esquerda para a direita, ao toque de uma ou duas mãos ao mesmo tempo. Para uma maior compreensão, os pontos são enumerados na coluna da esquerda, do alto para baixo como 1, 2 e 3; já na coluna da direta, do alto para baixo como 4,5 e 6 (figura 1).

[1] Instituto Federal do Pará (IFPA) – Campus Industrial Marabá; Email: jackson.oliveira@ifpa.edu.br

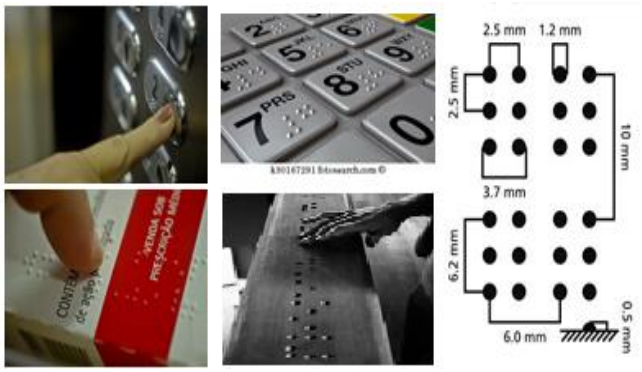


Figure 1.Exemplos de diferentes textos escritos em Braille e distância padrão entre os pontos e celas.

O Brasil conhece o sistema desde 1854, data da inauguração do Instituto Benjamin Constant, no Rio de Janeiro, chamado, à época, Imperial Instituto dos Meninos Cegos. Fundado por D. Pedro II, o instituto já tinha como missão a educação e profissionalização das pessoas com deficiência visual. "O Brasil foi o primeiro país da América Latina a adotar o sistema, trazido por José Álvares de Azevedo, jovem cego que teve contato com o Braille em Paris", conta a pedagoga Maria Cristina Nassif, especialista no ensino para deficiente visual da Fundação Dorina Nowill.

O Braille hoje já está difundido pelo mundo todo e, segundo pesquisa "Retratos da Leitura no Brasil", de 2008, do Instituto Pró-Livro, 400 mil pessoas leem Braille no Brasil. Não é possível, segundo o Instituto Dorina Nowill, calcular em porcentagem o que esses leitores representam em relação à quantidade total de deficientes visuais no país. Isso porque o censo do ano 2000, realizado pelo IBGE (Instituto Brasileiro de Geografia e Estatística), aponta que há 169 mil pessoas cegas e 2,5 milhões de pessoas com baixa visão. No entanto, este último grupo é muito heterogêneo - há aqueles que enxergam apenas 1% e, portanto, poderiam ler

apenas em Braille, como pessoas que enxergam 30% e podem utilizar livros com letras maiores.

Podemos perceber possíveis dificuldades por parte de professor e aluno no procedimento de ensino e aprendizagem. “O atendimento dos alunos com necessidades especiais nas escolas do ensino regular de ensino aumenta em termos de desafio como resultado da formação docente. A maioria dos professores egressos dos cursos de formação está mal preparada para lidar com tal heterogeneidade escolar” [12]. Os alunos apresentam dificuldades tácteis iniciais e tendem a fazer analogias com a escrita a tinta, nomeadamente quanto ao feitio das letras e às posições dos pontos que compõem a "cela braille".

As pessoas necessitam se comunicar ente si. A partir de imagens contendo uma mensagem na grafia Braille, podemos traduzir os caracteres em texto Português-BR, usando técnicas de processamento de imagem, para que as pessoas normais/baixa visões/cegas possam entender de forma rápida e fácil este código, de forma flexível e fácil de usar, reduzindo o esforço entre as pessoas cegas e deficientes visuais, ajudando professores a ensinar alunos cegos, e os pais a acompanhar o estudo de seu filho.

Os pontos refletem a luz de forma diferente devido ao ambiente e diferentes texturas, e alguns ajustes precisam ser feitos principalmente por causa do ruído na imagem. Uma difícil tarefa é agrupar os pontos em padrões de caracteres (cela) Braille através da medição de espaços entre os pontos, devido a escala, rotação e distorção da imagem.

OBJETIVO

Desenvolver um aplicativo que possa traduzir a grafia Braille para a Língua Portuguesa (PT-BR), para que pessoas que interagem com cegos possam se comunicar através dos pontos, sem ter um conhecimento prévio do Braille.

METODOLOGIA

O trabalho foi escrito na linguagem de programação Java e a biblioteca JAI (Java Advanced Imaging), que podem ser usadas para criar aplicações de processamento e visualização de imagens. Implementado na IDE NetBeans e executado na JVM/JAR.

A imagem capturada em sua forma original não é ideal para realizar o processo de tradução. É importante que apenas as informações dos pontos estejam presentes. Neste caso, a imagem encontra-se na forma colorida (RGB). A forma ideal é processá-la, onde destacam os pontos (em preto) do resto da imagem (o fundo em branco). O processo realizado nesta aplicação, envolve duas etapas: Pré-processamento de imagem; Reconhecimento de Padrões.

É necessário realizar o processamento da imagem para um posterior reconhecimento dos padrões, para que seja resolvido algumas situações quanto a captura da imagem, tais como: luz e sombra de ponto, binarização da imagem, distância entre os pontos dos caracteres para separar cada caractere na mesma linha e em linhas diferentes, detectar algum erro na imagem, detectar a escala e rotação/inclinação da imagem Braille, e as celas incompletas, de acordo com a combinações de caracteres.

As etapas fundamentais do processamento da imagem estão ilustradas na Figura 2 [5].Portanto, a aplicação permitir abrir/capturar a imagem Braille a partir da mídia interna do computador, realizar o Pré-Processamento (converter uma imagem de RGB para escala de cinza, binarizar a imagem, melhorar a imagem usando técnicas de morfologia: eliminar ruído, imagem negativa, dilatação dos pontos e eliminar bordas da imagem); o Reconhecimento (filtrar os caracteres, realizar a segmentação de caracteres em linhas e colunas, combinar os caracteres Braille

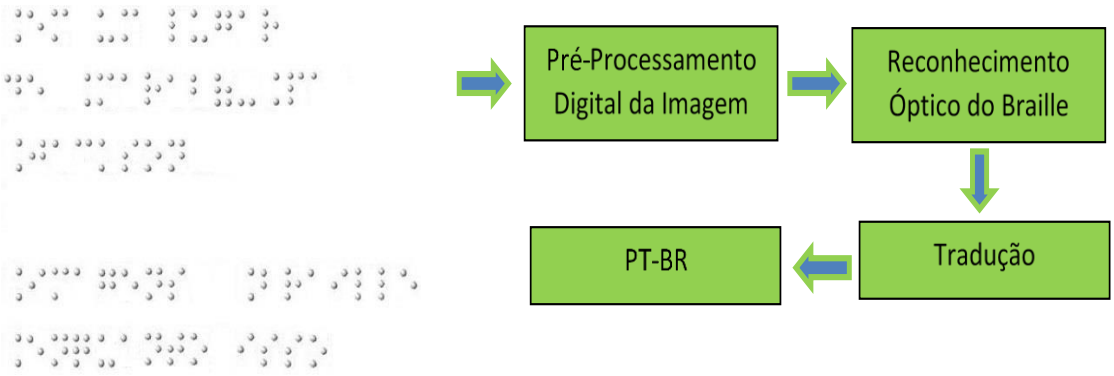


Figure 2. Metodologia adotada para realizar a tradução.

com seus respectivos códigos binários – Braille ASCII) e Traduzir para o Português-BR (figura 2).

A tabela Braille ASCII, foi criada a partir de um estudo sobre Atendimento Educacional Especializado [1], Grafia Braille para a Língua Portuguesa [2], Grafia Braille para Informática [3] e Normas Técnicas para a Produção de Textos em Braille [8].

DESENVOLVIMENTO, RESULTADOS E DISCUSSÕES

Tratar imagens digitalmente é uma coisa cada vez mais comum, seja para fazer correções de luz, cor, retirar imperfeições ou tratá-las para uma determinada finalidade, como identificar elementos na imagem de forma automática.

ETAPAS DO PROCESSAMENTO DIGITAL DA IMAGEM

Escala de Cinza

Primeiro realizamos o processamento da imagem, converte a imagem original em tons de cinza (figura 3), que descarta informações desnecessária sobre a imagem e foca na intensidade da luz refletida pelos os pontos. Cada pixel da imagem é composto de três valores, cada um deles corresponde ao nível de intensidade das cores primárias azul (B, blue), vermelho (R, red) e verde (G, green).

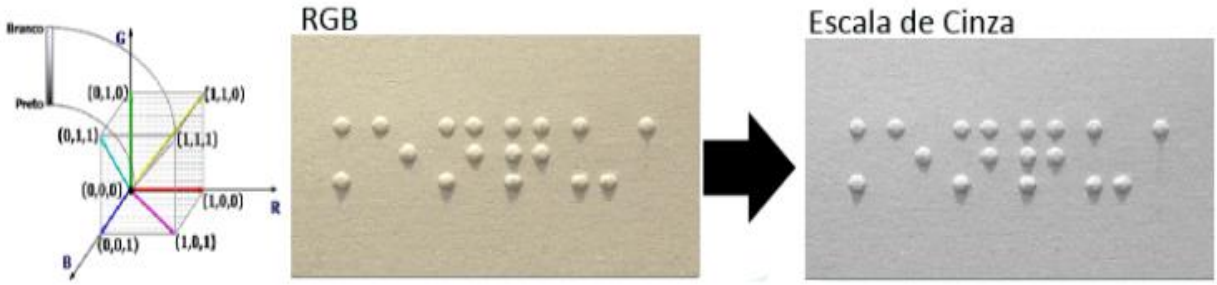


Figure 3. Imagem colorida para Escala de Cinza.

Para converter uma imagem de cor RGB para a imagem em tons de cinza, ou seja, uma imagem colorida terá apenas pixels brancos, pretos ou cinzas, pode-se usa a seguinte fórmula: I = 0,299 * R + 0,587 * G + 0,114 * B (algoritmo 1).

Algoritmo 1: Escala de Cinza
Entrada: Informações das cores primárias na imagem original.
Saída: Cinza.

```
// Captura os pixels R, G, B
alpha = new Color(original.getRGB(i, j)).getAlpha();
red = new Color(original.getRGB(i, j)).getRed();
green = new Color(original.getRGB(i, j)).getGreen();
blue = new Color(original.getRGB(i, j)).getBlue();
red = (int) (0.299 * red + 0. 587 * green + 0. 114 * blue);
// Voltar ao formato original
newPixel = UtilidadesImagem.corParaRGB(alpha, red, red, red);
```

```
// Atribui pixels na imagem
cinza.setRGB(i, j, newPixel);
return cinza;
```

Binarização / Liminarização

O próximo passo é a binarização (ou liminarização, do inglês, thresholding) da imagem, que consiste em dividir os pixels, a partir da imagem escala de cinza, em apenas dois níveis, preto e branco, de acordo com um valor limite T. O objetivo é deixar uma imagem em preto e branco, onde cada pixel da imagem original se transforma em um pixel totalmente branco (RGB #FFFFFF) ou totalmente preto (RGB #000000). Para determinar se um pixel da imagem original se tornará preto ou branco é usado um limiar, um valor que determina a partir de qual ponto um pixel se tornará preto ou se tornará branco [11]. Se seu nível cinza for menor a T, o valor será 0 (preto=0), caso contrário será 1 (branco=255). A partir da imagem em tons de cinza, utilizou-se o algoritmo Binarização (algoritmo 2) usando imagem integral, para gerar a imagem binária (preto e branco).

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

Equação 1. Equação que descreve a valoração das cores preto e branco.

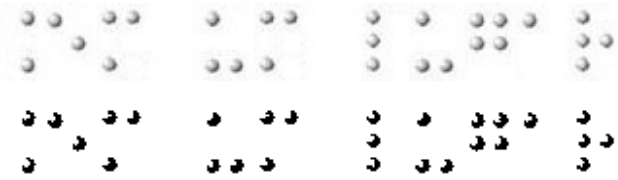


Figura 4. Imagem binarizada com método thresholding.

Algoritmo 2: Binarização

Entrada: Escala de Cinza.

Saída: Binária.

```
colorAux=new Color(this.original.getRGB(i, j));
T = 120;        mediaPixel=(int)((colorAux.getRed()+colorAux.getGreen()+colorAux.getBlue())/3);
    if (mediaPixel<T) { // limiar = 120; por exemplo
          colorSRGB=(0 << 16) | (0 << 8) | 0;
          binaria.setRGB(i, j,colorSRGB);
          //arreglo[i][j]=Color.BLACK;
        } else {
        colorSRGB=(255 << 16) | (255 << 8) | 255;
            binaria.setRGB(i, j,colorSRGB);
              }
        }
```

Técnicas Morfológicas

Eliminar Ruído

A imagem binária é analisada e se encontrar um pixel com cor preta, a posição é processada usando algoritmo de vizinhança de pixels (viz_4) procurando seus limites. Pontos demasiado pequeno ou grande são considerados como ruído na imagem e, portanto, descartado.

Filtro Mediana é usado para eliminar ruídos da imagem. O filtro de mediana é um filtro de vizinhança onde o valor de um pixel (x,y) é substituído pela mediana dos pixels da sua vizinhança, usado um ambiente de tamanho 3x3. Se houver um pixel preto (ruído) em meio há vários pixels brancos a aplicação do filtro eliminará este pixel preto, tirando o ruído da imagem.

Ordenar níveis de cinza em ordem crescente de intensidade, tomar o valor do centro da sequência (Mediana).

Figura 5. Procedimento do Filtro Mediana 3x3.

Negativo, Dilatação e Eliminar Bordas

Após eliminar os ruídos, podemos realizar o filtro Negativo onde a cor de cada pixel da imagem binária se transforme na cor inversa (ex: pixel branco se transforma em pixel preto). A cor inversa é o valor da subtração entre 255 e o valor RGB (figura 6).

Vamos usar a transformação negativo sobre a imagem obtida a partir do passo anterior, para pıxels como pontos brancos e, assim obter os histogramas verticais e horizontal.

Algoritmo 3: Negativo
Entrada: Binária sem ruídos
Saída: Negativo

```
Color color = new Color(neg.getRGB(i, j));
int r = color.getRed();
int g = color.getGreen();
int b = color.getBlue();
neg.setRGB(i, j, new Color(255-r,255-g,255-b).getRGB());
```

1.1.1.1 Dilatação

Na dilatação, a cada ponto da imagem adicionam-se pontos ao redor, de forma que a distância entre componentes da imagem passe a ser o dobro da original. Além disto, tais pontos são adicionados em toda a vizinhança do ponto original da image.

Aplicar esta operação na imagem para completar os pontos devido a binarização ter gerado pontos incompletos. Vamos usar um elemento estrutural 3x3.

Algoritmo 4: Dilatação dos pontos

Entrada: Negativo

Saída: Dilatada

```
Kernel kernel = new Kernel(3, 3, new float[] {
        1f, 1f, 1f,
        1f, 1f, 1f,
        1f, 1f, 1f
});
ConvolveOp op = new ConvolveOp(kernel);
op.filter(this.imagemAtual, dil);
```

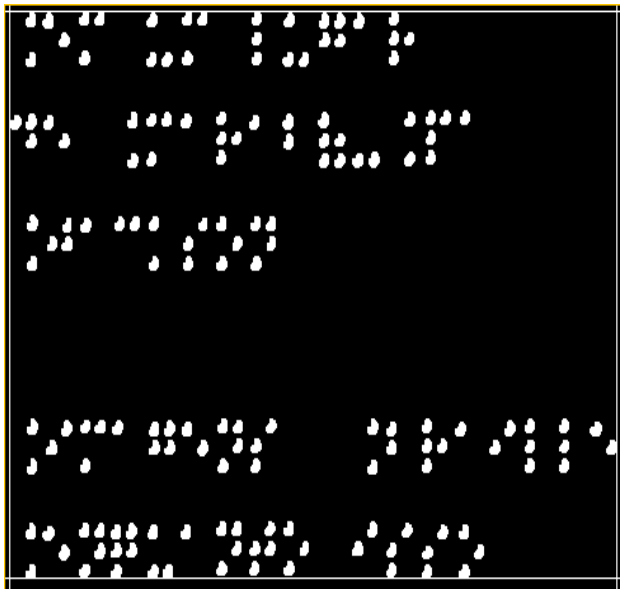


Figura 6. Imagem Negativa com índices detectados para eliminar as bordas.

Índices e Eliminar Bordas

Distâncias verticais entre os pontos e as bordas da imagem são detectados, e dividi o texto em linhas de Braille. Distâncias horizontais entre os pontos e as bordas são medidos, definindo as colunas.

Para fazer isso, a imagem é rastreada de cima para baixo (índices 1 e 2) e da esquerda para a direita (índices 3 e 4) procura o primeiro e último pixel branco em cada caso. Temos que analisar os caracteres Braille em que não possui pontos brancos em suas extremidades (primeira e última linha, e coluna esquerda e direita),nos casos de caracteres incompletos. Com esta etapa, podemos detectar a região de interesse, ou seja, o local da imagem que possui os códigos Braille, recortando a imagem a partir dos índices detectados.

RECONHECIMENTO DOS CARACTERES

Nesta etapa realiza a segmentação da imagem em linhas e colunas, encontrando os caracteres Braille, e posteriormente é dividido horizontalmente e verticalmente cada ponto do caractere, separando-os em duas colunas e três linhas (figura 1), para se obter o vetor de padrões (caracteres) utilizado para gerar a Tabela Braile ASCII, e assim traduzir para o Português-BR.

Detecção das Linhas e Caracteres

Cada pixel de uma imagem tem uma cor que foi produzida por uma combinação de cores primárias (vermelho, verde e azul, ou RGB). Cada uma dessas cores pode ter um brilho que varia de 0 a 255 em uma imagem digital com profundidade de bits de 8-bits.

Um histograma de uma imagem digital é um gráfico que representa os valores das tonalidades dos pixels numa imagem. Da mesma forma que é possível representar graficamente a altura das pessoas numa sala de aulas, é

possível representar a luminosidade dos pixels numa imagem. Um histograma RGB é produzido quando o computador varre a imagem em cada um desses valores de brilho RGB e conta quantos pixels há em cada nível de 0 a 255, com uma luminosidade particular e representa os números no gráfico.

Realiza a segmentação da imagem em linhas, de três em três, utilizando a função de histograma vertical da imagem, analisando os casos em que uma cela possui apenas pontos 1, 1x2, 2x1, 1x3, 2x2.

Para a segmentação dos caracteres, rastreia a imagem da esquerda para a direita, através do histograma horizontal de dois em dois. A zero valores que encontramos vamos ter encontrado um caractere Braille, como cada um tem 2 colunas.

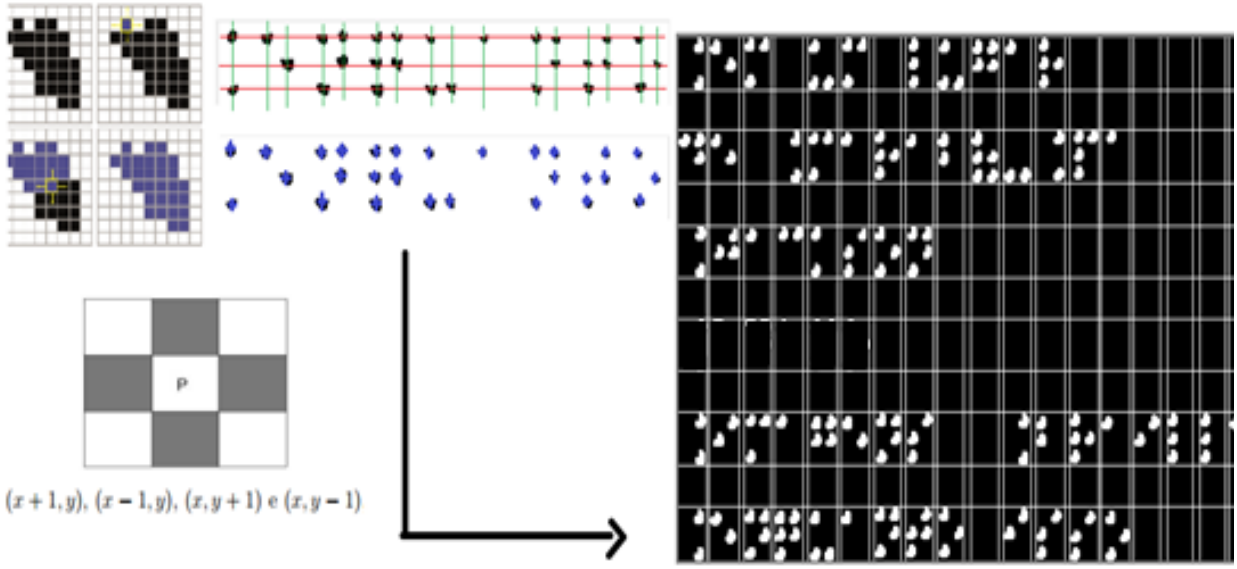


Figure 7. Segmentação em linhas e colunas para detecção das celas Braille.

.

Distância entre os pontos de duas colunas é medido, e em seguida, decidido se a distância for grande o suficiente será dois caracteres ou não (figura 8). Esta

técnica cria uma lista de pontos detectados na imagem a partir do centro de cada ponto, separando os caracteres para uma posterior divisão e detecção da matrix 3x2 de cada caractere. O resultado final do algoritmo é mostrado na Figura 7.

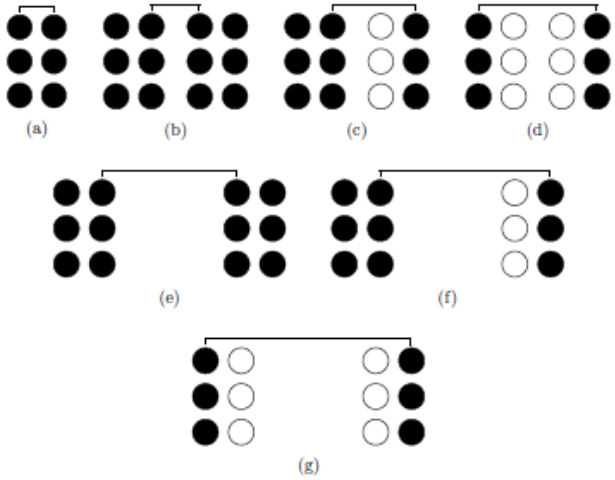


Figura 8. Possíveis distâncias entre as colunas e celas.

Divisão dos Caracteres em 3x2 pontos (Horizontal e Vertical) e Braille ASCII

Os pontos são agrupados em caracteres que por sua vez e montado uma tabela Braille ASCII correspondente a cada cela, como exemplo: Cela (R): 111010; Cela (U): 101001; Cela (I): 010.100 (figura 10). Cada carácter é representada como uma sequência de seis valores (matriz 3x2) binários, em que 1 indica que uma dada posição é gravada em relevo e 0 ponto não revelo (figura 9).

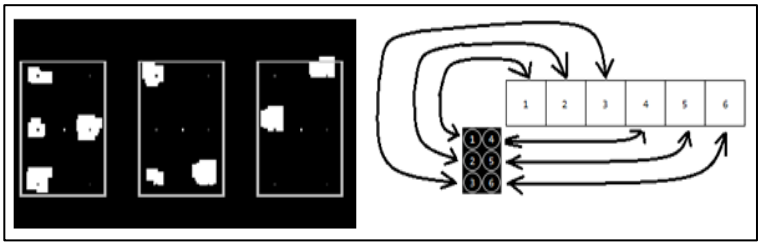
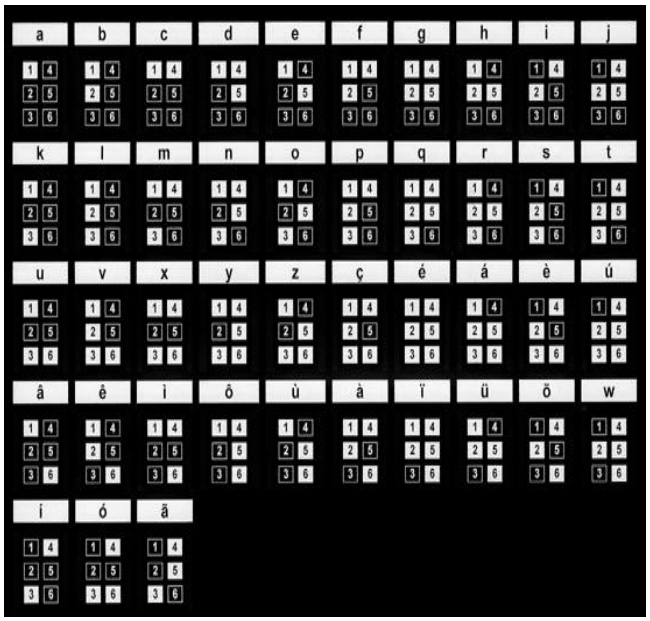


Figure 9. Método para criar a tabela Braille ASCII.

TRADUÇÃO DA GRAFIA BRAILLE PARA O PORTUGUÊS-BR

Na figura 10 podemos observar o Alfabeto Braille para o português, utilizada para elaborar a tabela da figura 12. Após realizamos a tradução a partir do vetor de padrões, conforme figura 11.



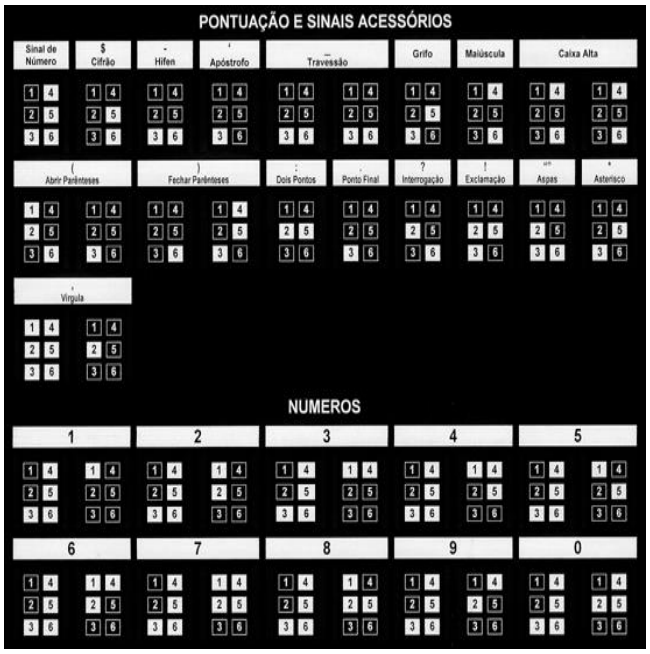


Figure 10. Grafia Braille da Língua Portuguesa (IBC 2016).

No exemplo o único ponto da cela em revelo é o primeiro. No final da verificação de cada ponto, a variável recebe o valor ‘100000” representando a existência de ponto na divisão localizada na parte superior esquerda, e a inexistência nos outros. No vetor de padrões, ao localizar o valor “100000”, verifica-se que este representa o caractere “a”.

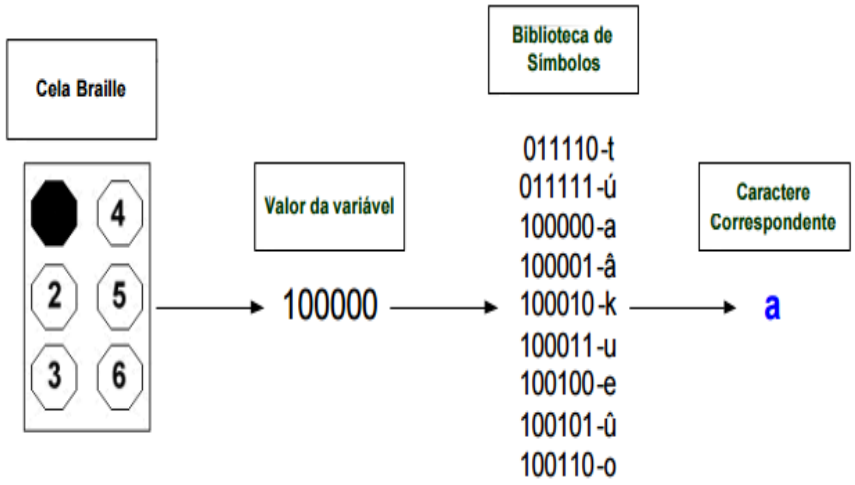


Figura 11. Passos para tradução de uma cela Braille.

Desta forma e olhando para o exemplo anterior, o símbolo “t” pode ser codificado como “011110” onde cada ponto preto é um ponto ativo ou uma “montanha” no papel e eles se tornam “1” no número binário. Esta forma de binarização de texto em Braille torna o sistema global independente da linguagem do documento e facilmente configurável para adicionar alfabetos diferentes. A saída desta etapa será um arquivo com cada caractere codificado como um número binário; Só teremos que traduzir cada número para a sua letra equivalente em texto normal para obter a saída final como um arquivo de texto.

CONCLUSÃO

O projeto centra-se na criação de um aplicativo para traduzir a grafia Braille para o Português-BR, permitindo a ler e compreender textos normalmente utilizados por cegos, sendo capaz de reduz os esforços entre as pessoas cegas, deficientes visuais e pessoas com visão. A aplicação foi testado com a plataforma JVM (figura 13) que pode selecionar uma imagem a partir da mídia interna do computador e

realizar o processamento da imagem e tradução. Outra vantagem é a sua simplicidade e eficiência na execução. Foi necessário estudar as regras da grafia Braille para montar a tabela Braille ASCII (Figura 12), a biblioteca JAI, e IDE NetBeans com Linguagem de Programação Java, para conseguimos projetar, implementar e reconhecer a grafia Braille a partir de imagens digitais.

Imagem Braille é uma imagem sensível, o que significa que devem ser capturados em situação adequada, a fim de obter um bom resultado. Técnicas de morfologia pode ajudar a melhorar a imagem de um ruído. A imagem capturada sempre tem um ângulo de inclinação. É possível programar um aplicativo para o Android usando JAVA.

Braille Dots	Braille Glyph	Braille Meaning		Braille Dots	Braille Glyph	Braille Meaning
000000		(space)		110001		ê
100000		a		100111		ô
110000		b		001110		ã
100100		c		010101		õ
100110		d		001000		,
100010		e		011000		;
110100		f		010010		:
110110		g		001000		.
110010		h		010001		?
010100		i		011010		!
010110		j		001001		-
101000		k		110101		à
111000		l		000001		/
101100		m		001100		í
101110		n		011011		=
101010		o		001010		*
111100		p		000011		$
111110		q				Maiúsculo
111010		r				Números
011100		s				
011110		t				
101001		u				
111001		v				
010111		w				
101101		x				
101111		y				
101011		z				
111101		ç				
111111		é				
111011		á				
011111		ú				
100001		â				

Figure 12. Tabela Braille ASCII.

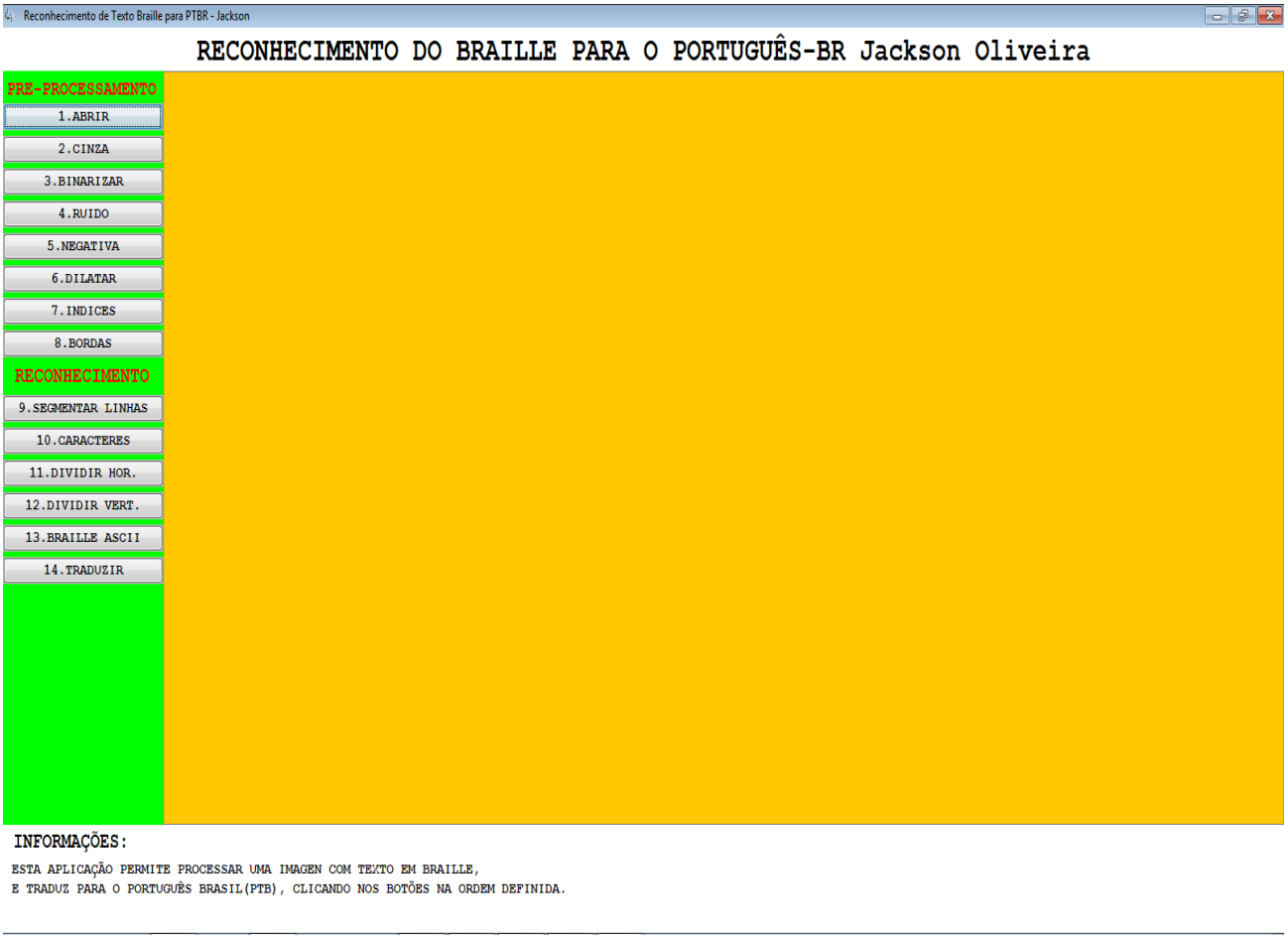


Figure 13. Interface gráfica do Aplicativo BrailleToPTB(BR) – Jackson Oliveira.

.

TRABALHOS FUTUROS

O trabalho teve resultados importantes, mas para o desenvolvimento de um sistema robusto é necessário a continuação do desenvolvimento do mesmo para

obter uma mobilidade da aplicação. Pretende-se desenvolver um aplicativo para a Plataforma Móvel Android, que utilizará a câmera do Smartplhone para a tradução em tempo real, com possível leitura sonora do texto.

REFERENCIAS

[1] **Atendimento Educacional Especializado. Deficiência Visual**. Secretaria de Educação Especial. Brasília: SEESP, 2007. Disponível em: http://portal.mec.gov.br/seesp/arquivos/pdf/ aee_dv.pdf. Acesso em: 12 jun. 2016.

[2] **Grafia Braille para a Língua Portuguesa** / elaboração: Cerqueira, Jonir Bechara... [et al.]. Secretaria de Educação Especial. Brasília: SEESP, 2006. 106p. Disponível em http://portal.mec.gov.br/seesp/arquivos/pdf/ grafiaport.pdf. Acesso em junho 2016.

[3] **Grafia Braille para Informática** / Coordenação: Lêda Lucia Spelta, Maria Glória Batista da Mota; Autores: Antônio Carlos Hildebrandt ... [et al.] . Brasília: MEC, SEESP, 2004. 52 p.

[4] GARCIA, G. B.; SUAREZ, O. D.; ARA NDA, J. L. E.; TERCERO, J. S.; GRACIA, I. S.; ENANO, N. V. **Learning Image Processing with OpenCV**. Packt Publishing, 2015.

[5] GONZALEZ, R. C; WOODS, R. E. [2008]. **Digital Image Processing**, 3rd ed., Prentice Hall, Upper Saddle River, NJ.

[6] IBC. INSTITUTO BENJAMIN CONSTANT. **Grafia Braille da Língua Portuguesa.** 2005a. Disponível em http://www.ibc.gov.br/?catid=112&blogid=1&itemid=344. Acesso em junho 2016.

[7] HOWSE, Joseph. **Android Application Programming with OpenCV 3**. Packt Publishing, 2015.

[8] **Normas técnicas para a produção de textos em Braille/elaboração: Edison Ribeiro Lemos... [et al]**. – Brasília: Ministério da Educação, Secretaria de Educação Especial, 2006. Disponível em http://portal.mec.gov.br/seesp/arquivos/pdf/textosbraile.pdf. Acesso em junho 2016.

[9] **OPENCV. Biblioteca gráfica**. Disponível em: http://opencv.org/. Acessado junho de 2016.

[10] SALIL, Kapur; NISARG, Thakkar. **Mastering OpenCV Android Application Programming**. Packt Publishing, 2015

[11] **Processamento Digital de Imagens (PDI) com Java (Negativo)**. Disponível em: https://oxenteexception.wordpress.com/2011/01/21/processamento-digital-de-imagens-pdi-com-java/. Acesso em: 13/06/2016.

[12] SANTOS, M. de J. dos. **A escolarização do aluno com deficiência visual e sua experiência educacional**. Dissertação (mestrado) - Universidade Federal da Bahia, 2007. Disponível em: https://repositorio.ufba.br/ri/bitstream/ri/10613/1/Miralva%20dos%20Santos.pdf. Acessado em: 13/06/2016.