GABRIEL AKIRA YANAGUYA

**DESENVOLVIMENTO DE APLICATIVO ANDROID PARA SÍNTESE DE VOZ DE TEXTOS EXTRAÍDOS DE IMAGENS POR MEIO DE OCR**

Projeto de pesquisa apresentado ao Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, *campus* São João da Boa Vista, como requerimento parcial para fins de aprovação da disciplina Metodologia de Pesquisa Científica. Orientador: Prof. Gustavo Aurélio Prieto. Coorientador: Prof. Ricardo Alexandre Neves

SÃO JOÃO DA BOA VISTA

2017

**RESUMO**

YANAGUYA, G. A. (2017). **Desenvolvimento de aplicativo Android para síntese de voz de textos extraídos de imagens por meio de OCR.** Artigo Cientifico - Instituto Federal de São Paulo, São João da Boa Vista, 2017.

A expansão das tecnologias na área de informática e o crescimento dos recursos de hardware e software disponíveis em dispositivos móveis intensificam o potencial que essas tecnologias têm de assumirem o papel de principal ferramenta de auxílio a indivíduos portadores de deficiência, promovendo a sua autonomia, independência e qualidade de vida.

A síntese de fala, que permite a conversão automática de texto digital em áudio, é um recurso recorrente em softwares de tecnologia assistiva. Entretanto, apesar do crescimento considerável de pesquisas e produtos lançados nessa área, ainda é modesta a abrangência e eficiência dessa tecnologia, principalmente em casos onde o texto não se encontra em formato de texto digital editável, como textos impressos ou em formato de imagem ou vídeo.

Este projeto tem como principal objetivo a análise e desenvolvimento de um aplicativo Android para síntese de voz de textos extraídos de fotografias obtidas a partir de câmera fotográfica de smartphones e tablets. Os resultados obtidos após a submissão do aplicativo a experimentos práticos possibilitou, por meio da aplicação de uma métrica de avaliação de processamento de extração óptica de caracteres, determinar aspectos que influenciam negativamente na precisão do OCR, bem como identificar intervenções que minimizam os efeitos desses aspectos.

Palavras-chave: OCR. Tecnologias assistivas. Text-To-Speech. Android.

# 1 LINHA DE PESQUISA

Algoritmos e Processamento de Imagens;

Desenvolvimento para Dispositivos Móveis;

Acessibilidade e Tecnologias assistivas;

# 2 TEMA E SUA DELIMITAÇÃO

Projeto e desenvolvimento de um aplicativo Android para síntese automática de voz de textos extraídos de imagens por meio de OCR[[1]](#footnote-1), como ferramenta de auxílio de leitura para portadores de deficiência visual ou àqueles que apresentarem grau de analfabetismo que impeça ou dificulte a leitura de texto.

# 3 JUSTIFICATIVA

De acordo com pesquisa da Organização Mundial da Saúde (World Health Organization, WHO), cerca de 285 milhões de pessoas sofrem de cegueira ou algum tipo de deficiência visual. Diante da quantidade notável dessa população, a informática cumpre um papel importante na capacitação da leitura mediada por tecnologias assistivas. (REDIG, 2012; RESNIKOFF, 2002; WHO, 2012).

Entretanto, apesar dos esforços para o desenvolvimento de tecnologias assistivas como ferramenta de suporte a indivíduos portadores de deficiência, existe um número restrito de softwares gratuitos para auxílio de leitura, sobretudo aplicativos para dispositivos móveis. EXISTEM SOFTWARES PAGOS ??

Ainda, sendo a tecnologia OCR uma importante ferramenta para a construção de softwares de acessibilidade, a baixa precisão no reconhecimento óptico dos caracteres imagens capturadas por câmeras fotográficas de dispositivos móveis também se destaca como justificativa do desenvolvimento deste projeto.

# 4 OBJETIVOS

Este trabalho apresenta como objetivo geral o projeto, análise e desenvolvimento de um aplicativo para dispositivos móveis, na plataforma Android, que permita a síntese de voz de textos extraídos de imagens por meio de OCR. Esse aplicativo deve atuar como ferramenta de acessibilidade para auxílio de leitura para indivíduos portadores de deficiência visual ou àqueles que apresentem algum grau de analfabetismo.

Os objetivos específicos desta pesquisa são:

* Aplicação de intervenções (pré-processamento de imagem) para favorecer a precisão do Tesseract OCR[[2]](#footnote-2);
* Desenvolvimento de uma metodotogia baseada em experimentos práticos, afim de mensurar, analisar e documentar a influência das intervenções nos resultados obtidos;
* Adoção de uma métrica de mensuração de resultados de processamento OCR, bem como estruturação de testes condizentes com essa métrica;

# 5 FUNDAMENTAÇÃO TEÓRICA

## 5.1 Tecnologia assistiva para auxílio à leitura

Tecnologia assistiva (TA) é definida como o conjunto de recursos e serviços que contribuem para ampliar ou proporcionar as habilidades funcionais de indivíduos portadores de deficiências, proporcionando inclusão e independência aos seus utilizadores. Dentre as diversas aplicações da TA, este trabalho apresenta aspectos das tecnologias de auxílio para a suporte da habilidade visual e recursos que ampliam a informação a pessoas com baixa visão ou cegas, sintetizando voz de textos impressos por meio de TTS e reconhecimento óptico de caracteres (BERSCH, 2008).

## 5.2 TTS

TTS, sigla para *Text-To-Speech*, é uma tecnologia que permite converter texto digital em voz, sintetizando a fala correspondente ao texto de entrada (LECHETA, 2013).

O processo de síntese de voz via TTS é definido a partir dos seguintes passos:

1. A entrada de um dado texto;
2. A normalização do texto recebido;
3. A detecção de derivações de pronúncia;
4. A conversão da representação ortográfica para a representação fonológica;
5. O *parsing* semântico e gramatical;
6. A associação de uma sequência fonológica à cada unidade prosódica;
7. A geração do sinal de fala (saída de áudio);

(ADDISON et. al., 2005; TATHAM, 2005)

A tecnologia TTS vem sendo amplamente empregada na construção de softwares de acessibilidade, permitindo a narração automática de páginas web, softwares e documentos digitais. Diversos sistemas operacionais modernos, como as versões atuais do Microsoft Windows, o Android e o IOs já possuem os recursos de TTS disponíveis como configuração de fábrica.

## 5.3 OCR

OCR (Optical Character Recognition) é uma tecnologia empregada na detecção, identificação e extração de caracteres, a partir de um arquivo de imagem, ou seja, na obtenção de texto editável a partir de textos em formato de imagem, seja ela escaneada, escrita à mão, fotografada, datilografada ou impressa (OCH, 2015).

### 5.3.1 Tesseract OCR

A Tesseract Engine é um *software* de reconhecimento óptico de caracteres *open-source* com distribuição gratuita desenvolvido pela Hewlett Packard (HP) entre 1984 e 1994. Em 1995, ainda em propriedade da HP, foi considerada um dos três *softwares* OCR com maior precisão e desempenho. Em 2006, o projeto recebeu patrocínio oficial da Google, que desde então é responsável pela manutenção e distribuição do *software*, sob licença Apache 2.0 (SMITH, 1995).

Inicialmente desenvolvido na linguagem C e posteriormente migrado para C++, o Tesseract está atualmente disponível para Windows, Linux e Mac OS X e pode reconhecer 38 idiomas, como inglês, alemão, chinês, italiano, japonês, russo, espanhol e outros, podendo ser treinado para reconhecer caracteres de qualquer idioma ou fonte, incluindo textos manuscritos (SMITH, 1995). Segundo Sonia Bhaskar (2010), o Tesseract é considerado o *software* OCR gratuito mais preciso em operação.

## 5.4 Precisão dos resultados

A Tesseract Engine, assim como a maioria dos softwares OCR, foi desenvolvida para extração de caracteres a partir de análise de documentos escaneados. Quando utilizado em fotos capturadas através de dispositivos móveis, alguns fatores, grande parte relacionados a natureza do ambiente, influenciam na captura da imagem, causando distorções e ruídos que podem culminar em perda de precisão no processo de extração. Entre esses fatores, destacam-se a iluminação, as distorções de perspectiva, o contraste entre o texto e o resto do ambiente, o foco da imagem e a possibilidade de presença de objetos que podem ser confundidos com caracteres ou até linhas de texto (RUSIÑOL, 2014; YANAGUYA, 2015).

### 5.4.1 Processamento Básico

Internamente, a Tesseract Engine executa diversas operações de processamento de imagem utilizando a biblioteca *open source* Leptonica antes de realizar o processo de extração e reconhecimento de caracteres. Apesar de esse processamento se apresentar eficiente em muitos casos, existem cenários onde esse processamento não atinge os resultados esperados, resultando em uma redução significativa na precisão**.** Assim, existem operações que quando executadas antes da submissão da imagem ao Tesseract, auxiliam na redução dos problemas dos processos internos da API[[3]](#footnote-3). São eles:

* Redimensionamento da imagem – O Tessetact apresenta resultados melhores em imagens com densidade 300dpiou superior, sendo benéfico redimensionar a imagem afim de atingir essa característica; e
* Conversão da imagem em *grayscale* – O processo de binarização de Otsu[[4]](#footnote-4), utilizado internamente no Tesseract, funciona de forma mais eficiente em imagens em tons de cinza **(**GITHUB.COM, 2017).

### 5.4.2 Filtros de Luminosidade e Contraste

O GPUImage é um framework de processamento de imagens *open source* desenvolvido por Brad Larsonutilizando a biblioteca OpenGL ES 2.0, projetado inicialmente para iOS. Esse framework permite a aplicação de diversos filtros a imagens e vídeos. Os filtros de luminosidade e contraste presentes no GPUImage auxiliam na correção de distorções de luminosidade provenientes do processo de captura, bem como auxiliam na separação morfológica entre os objetos e o fundo comum da imagem.

### 5.4.3 Remoção de ruídos

O ruído é definido como uma variação aleatória de brilho ou cor em uma dada imagem. Pequenos ruídos são detectados e removidos automaticamente pelo processo de binarização. Entretanto, ruídos de maior intensidade podem ser equivocadamente classificados como possíveis caracteres na etapa de binarização. REFERENCIA

X, Y e Z removem ruídos. Dentre esses, destaca-se X, porque.....  
(A IDEIA É COMPARAR).

Portanto, afim de aprimorar a precisão de softwares OCR, utiliza-se operações de processamento de imagens com o objetivo de reduzir a ocorrência de ruídos em uma imagem, como as operações Erode e Dilate da OpenCV[[5]](#footnote-5).

Erode e Dilate são duas operações morfológicas, disponíveis na biblioteca *open source* OpenCV, que processam uma imagem com base em formas. Ambas podem ser aplicadas para remoção de ruído, isolamento de elementos individuais e junção de elementos diferentes em uma imagem. Essas operações permitem que componentes não-desconexos expandam sua área, ocupando espaços de componentes desconexos, minimizando-os ou até mesmo os eliminando por completo da imagem (OPENCV.ORG).

A Figura 1 ilustra a aplicação das operações Erode e Dilate por meio da OpenCV:

**Figura 1: Aplicação das operações Erode e Dilate da OpenCV**



*Fonte: Elaborada pelo autor*

### 5.4.4 Binarização e Algoritmo de limiarização de OTSU

A binarização ou limiarização é o processo de segmentação de uma imagem de acordo com a contextualização de seu conteúdo, subdividindo a imagem em regiões ou objetos distintos. O processo de binarização permite a conversão de uma imagem *grayscale* em uma imagem binária (preto e branco), de forma a separar em duas classes diferentes objetos e regiões significativas (*foreground*) e fundo ou cenário insignificativo (*background*). O processo pode ser definido pela função s = T(r) : compara-se o sinal de entrada com um determinado valor de *threshold* (T). Esse valor pode ser pré-determinado ou calculado dinamicamente de acordo com aspectos da imagem, e serve como referência para a separação dos níveis de cinza. O sinal de saída, 0 (preto) ou 1 (branco) é obtido pela seguinte relação: 1, se r > T, ou 0, se r < T. A ocorrência de igualdade entre r e T é tratada de acordo com a ocasião. (VALA, 2013).

A binarização de Otsu é um algoritmo de binarização desenvolvido por Otsu (1979) que, ao oposto de utilizar um valor arbitrário, calcula o valor de *threshold* automaticamente a partir de análise do histograma da imagem. Considerando uma imagem bimodal (imagem cujo histograma apresenta dois picos), o algoritmo de Otsu calcula um valor entre os dois picos do histograma, e o utiliza como parâmetro *threshold*. A proposta do algoritmo é realizar uma iteração por todos os valores possíveis para o *threshold*, com o objetivo de identificar o valor que minimiza a soma da variância intraclasses da imagem. A variância intraclasse para um possível valor de *threshold* T pode ser calculada de acordo com:



Sendo *W* o peso para cada classe. Esse valor corresponde à probabilidade de um pixel pertencer a classe b (*background*) ou f (*foreground*) (OTSU, 1979; TOROK, 2014).

## 5.5 Métrica de Alves para avaliação de OCR

Para que seja possível avaliar e classificar os resultados de uma ferramenta de reconhecimento óptico de caracteres, é necessário mensurar o quão distante um texto transcrito está em relação ao texto original. Alves (2003) desenvolveu uma métrica para avaliação de OCR baseando-se na Distância de Levenshtein. A Distância de Levenshtein é um algoritmo que permite comparar e determinar a distância entre dois textos, considerando apenas a soma da quantidade de caracteres inseridos, excluídos e/ou substituídos.

Como exemplo, a Distância de Levenshtein entre a palavra “mestre” e a palavra “mastros” é dada da seguinte forma:

**mestre**

mastre – substituição de “e” por “a”;

mastro – substituição de “e” por “o”;

mastros – inserção do caractere “s”;

Portanto, a distância entre “mestre” e “mastros” é equivalente a 3, considerando duas substituições e uma inserção.

Entretanto, esse algoritmo computa a quantidade de erros, sem considerar a localização onde esses erros são gerados. Como a maioria dos softwares de OCR comparam ou recomendam a comparação das palavras do texto extraído com uma base pré-existente de palavras ou um dicionário, a ocorrência de muitos erros em uma única palavra reduz a chance de correção automática nesse processo. A respeito, Alves (2003) declara:

A métrica original de Levenshtein não leva em consideração a onde os erros ocorrem. Por exemplo:

*‘To be or not to be, thaz’s the queztion.’*

*‘To be or not to be, that’s the querkion.’*

A primeira frase do famoso monólogo de Hamlet, de Shakespeare, tem a mesma distância de Levenshtein. Entretanto, o fato da segunda transcrição ter dois erros na mesma palavra (“querkion” em vez de “question”) reduz bastante a probabilidade de recuperação do erro. Então a distância de Levenshtein foi extendida com a taxonomia de classificar os erros e sua posição. (p. 52).

Na métrica proposta por Alves, os erros foram classificados em erros de caractere, de palavra e de linha, sendo:

* Erros de caractere: inclusão (caracteres não-existentes inseridos erroneamente no texto), exclusão (caracteres existentes no texto original excluídos do texto obtido), troca simples (um caractere é substituído por outro), troca múltipla incluinte (um caractere é substituído por dois caracteres, como ‘h’ por ‘l’ e ‘n’), junção de palavras com perda (o espaço separador de palavras mais o próximo caractere são excluídos, resultando em junção de duas palavras com a perda de um caractere) e junção de palavras sem perda (exclusão de um espaço separador de palavras, resultando na junção delas).
* Erros de palavra: inclusão (caracteres não-existentes inseridos erroneamente em uma palavra), exclusão (caracteres existentes no texto original excluídos da palavra), troca simples (um caractere é substituído por outro).
* Erros de linha: inserção (uma linha de texto, inexistente no texto original, é inserida na transcrição) e exclusão (uma linha de texto existente no texto original é removida erroneamente na transcrição).

Assim, são propostas duas métricas para avaliar separadamente a qualidade dos resultados da transcrição do texto: uma considerando os erros de caractere e uma considerando os erros de palavra, dadas da seguinte forma:

MC = 100 – ( ( ERR\_CHAR x 100 ) / TOTAL\_CHAR)

e

MP = 100 – ( (ERR\_PALAVRA x 100) / TOTAL\_PALAVRAS)

Sendo:

ERR\_CHAR – a quantidade total de ocorrência de erros de caractere;

TOTAL\_CHAR - a quantidade total de caracteres do texto original;

ERR\_PALAVRA - a quantidade total de ocorrência de erros de palavra;

TOTAL\_PALAVRA - a quantidade total de palavras do texto original;

TEM A F-SCORE, USADA POR FOONG, 2013. PESQUISAR ??

## 5.6 Trabalhos Correlatos

FOONG (2013) desenvolveu um aplicativo para dispositivos móveis para síntese de voz de textos de placas, sinalizações e avisos, obtidos por meio de OCR, empregando a Tesseract Engine. O objetivo principal do trabalho de Foong é prover maior independência para indivíduos portadores de deficiência visual, que com a ajuda do aplicativo, podem identificar informações impressas em ambientes internos (interior de residências ou edifícios comerciais) como quadros de avisos, placas de alerta ou sinalização de rotas. Foong, afim de aumentar a precisão do processo de OCR, também propôs o processamento da imagem antes da submissão ao Tesseract, convertendo-a em *grayscale*, aplicando filtros de redução de ruído, efetuando a binarização da imagem pelo algoritmo de Otsu, e isolando os caracteres do resto da imagem. Assim, a imagem analisada pelo Tesseract contém apenas caracteres, pois o fundo é removido pelo pré-processamento. O texto extraído pelo Tesseract é então sintetizado a partir do recurso TTS nativo do Android. Foong descreve ainda que a precisão dos resultados foi obstruída por fatores como iluminação e coloração do texto, embora de maneira geral tenham se mostrado muito satisfatórios (FOONG, 2013).

(YANAGUYA, 2015) – É uma boa ideia citar eu mesmo aqui??

Alves (2003) também desenvolveu uma pesquisa na área de reconhecimento óptico de caracteres, tendo como objetivo primário o desenvolvimento de uma métrica baseada no algoritmo da Distância de Levenshtein, afim de mensurar o impacto da aplicação de diferentes filtragens na imagem antes de submetê-la a ferramentas OCR comerciais. Assim, além de quantificar a interferência de cada filtro (brilho, contraste, resolução, rotação, etc), Alves também descreve a métrica utilizada para avaliar os resultados individuais deles (ALVES, 2003).

# 6 METODOLOGIA

A fase de levantamento bibliográfico tornou evidente a existência de fatores externos, relacionados à natureza do ambiente onde a foto é capturada e do próprio processo de captura, que geram ruídos que acabam por influenciar negativamente os resultados da Tesseract Engine. Assim, esse trabalho propõe o processamento da imagem antes de submetê-la ao Tesseract, afim de eliminar ou minimizar a influência desses fatores na imagem capturada.

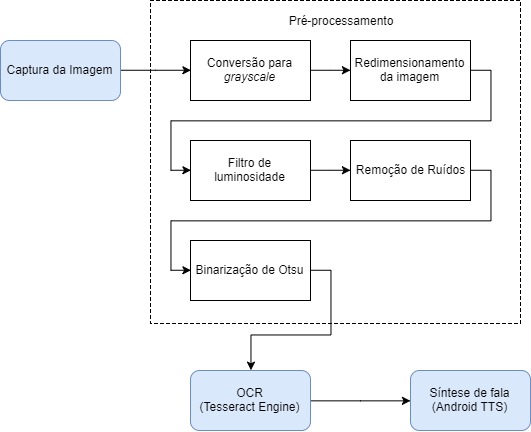
## 6.1 Tecnologias

A construção do software proposto neste trabalho fez uso da linguagem de programação Java (Android SDK 23). O reconhecimento óptico de caracteres foi obtido a partir da Tesseract Engine, encapsulada em um *wrapper* para a linguagem Java na biblioteca Tess4J. Para processamento da imagem afim de aumentar a precisão do OCR, foram utilizadas as bibliotecas OpenCV, GPUImage e Leptonica. Os recursos de captura de imagem e Text-To-Speech são funcionalidades presentes de forma nativa na SDK 23 do Android, não sendo necessário o emprego de softwares adicionais.

## 6.2 Arquitetura

A arquitetura do aplicativo pode ser dividida em cinco etapas distintas de processamento, de acordo com a Figura 2:

**Figura 2: Fluxo de processamento do aplicativo**

****

Fonte: Elaborada pelo autor

A primeira etapa (Captura da Imagem) consiste na obtenção da imagem por meio da câmera fotográfica, recurso normalmente disponível em smartphones e tablets. A captura utiliza as funções de captura nativas da plataforma Android, não sendo necessário o emprego de tecnologias adicionais.

A segunda etapa (Pré-Processamento) é a etapa da aplicação na qual a imagem capturada é processada, de forma a eliminar ou minimizar ruídos e outras interferências, potencializando a precisão do OCR na etapa de detecção e reconhecimento dos caracteres. Os processos envolvidos no tratamento da imagem são:

* Redimensionamento, processo no qual a imagem é redimensionada afim de atingir 300dpi, densidade na qual o Tesseract apresenta melhor precisão;
* Conversão para *grayscale*, processo no qual a imagem, até então colorida, é convertida em escala de cinza, formato no qual o Tessract aoresenta melhor precisão;
* Aplicação de filtros de luminosidade e contraste, que alteram automaticamente os aspectos de iluminação e contraste da imagem, de forma a destacar e definir bordas, evidenciando a separação de blocos de texto do resto da imagem;
* Binarização, processo por meio do qual a imagem é convertida em preto e branco, de forma a separar o que é classificado como ‘background’ (elementos secundários na imagem) e ‘foreground’ (elementos primários, incluindo os textos);
* Remoção de ruídos, processo que permite identificar pequenos componentes desconexos e ruídos na imagem, removendo-os;

A terceira etapa (OCR), é a etapa na qual as linhas de texto são identificadas, e os caracteres presentes são extraídos, analisados e identificados através de comparação da imagem com padrões pré-cadastrados. A tecnologia empregada foi o Tesseract OCR, software de reconhecimento óptico de caracteres desenvolvido pela Hewlett Packard (HP) entre 1984 e 1994 e patrocinado oficialmente pela Google, que desde 2006 é responsável pela manutenção e distribuição do software, sob licença Apache 2.0. Em 1995, ainda em propriedade da HP, a Tesseract Engine foi considerada um dos três softwares OCR com maior precisão e desempenho (SMITH, 1995).

A quarta e última etapa (Síntese de voz), é responsável pela exibição, em áudio, do resultado do processamento OCR. TTS é a API do Android responsável pela conversão automática de texto digital editável em voz (áudio). Nas versões mais atuais do Android, já estão disponíveis as engines de voz em italiano, francês, inglês, alemão e espanhol. Nem todas as versões suportam o idioma português do Brasil, conhecido tecnicamente como *pt-BR*. (LECHETA, 2013)

## 6.3 Experimentos práticos e avaliação dos resultados

Para mensurar e avaliar o impacto das intervenções propostas, foi aplicado um estudo comparativo, baseado no trabalho de Alves (2003), entre os resultados do Tesseract OCR a partir de imagens originais e a partir de imagens que sofreram alterações por meio das intervenções.

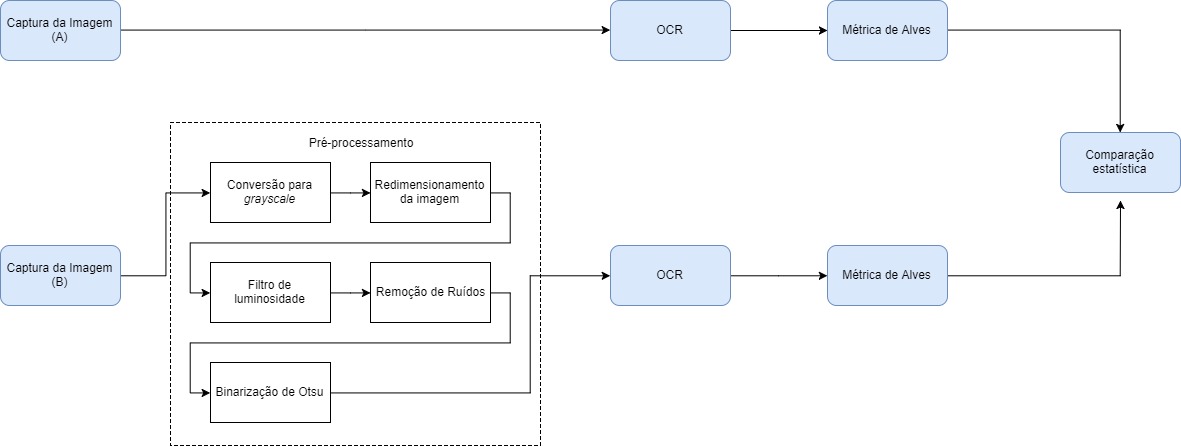
Os experimentos realizados consistiram na extração de caracteres de textos impressos por uma impressora Epson Ecotank L375, em papel sulfite A4.

A tipografia dos textos impressos foi padronizada nas fontes: Arial, Calibri e Adobe Gothic Std B, com tamanhos e cores variadas. A iluminação do local de testes foi abundante, com a fonte de luz posicionada de forma estratégica para não provocar sombras sobre a superfície escrita, e as imagens foram capturadas de forma a não causar distorções de perspectiva.

O aplicativo foi executado em um smartphone Motorola Moto G 1ª Geração, equipado com câmera traseira de 5 MegaPixels. As imagens foram capturadas em tamanho 1944 pixels de largura e 2592 pixels de altura, com 72 DPI. Os recursos de zoom e flash não foram acionados em nenhuma das capturas.

Assim, o processo de avaliação e comparação dos resultados é descrito pelo diagrama apresentado na Figura 3:

**Figura 3: Diagrama do processo de avaliação e comparação dos resultados de OCR**



Fonte: Elaborada pelo autor

Espera-se que o método estatístico possa comprovar que a população B (testes onde foram executadas operações de processamento de imagem antes da extração de caracteres) tende a apresentar resultados melhores que a população A. Assim, torna-se evidente que o processamento da imagem influencia positivamente na precisão dos resultados do Tesseract. MELHORAR E DAR NOME DO METODO ESTATÍSTICO???

TRABALHOS FUTUROS:

* Adicionar processamento de correção de perspectiva;
* Separar texto do resto da imagem e enviar apenas texto para o Tesseract, assim como fez FOONG;
* Melhorar processamento de imagem com fundo complexo. Método estatístico demonstrou que a melhora de performance ainda não atingiu o resultado esperado.

# 7 RESULTADOS

Os experimentos práticos foram conduzidos em ambiente controlado com uma pluralidade de cenários. De maneira geral, os testes apontam um aperfeiçoamento considerável na precisão do processo de OCR quando aplicadas as intervenções propostas.

Explicar cenários onde a precisao aumentou muito ( textos longos ) e onde piorou (fundo complexo) E mostrar tabela de resultados.

**REFERÊNCIAS**

ADDISON, Edwin R. et al. **Text to speech**. U.S. Patent n. 6,865,533, 8 mar. 2005.

ALVES, Neide F. Estratégias para melhoria do desempenho de ferramentas comerciais de reconhecimento óptico de caracteres. 2003.

BERSCH, Rita. Introdução à tecnologia assistiva. **Porto Alegre: CEDI**, 2008.

BHASKAR S. et al., Implementing Optical Character Recognition on the Android Operating System for Business Cards, 2010.

FOONG, Oi-Mean; SULAIMAN, Suziah; LING, Kiing Kiu. Text signage recognition in Android mobile devices. **Journal of Computer Science**, v. 9, n. 12, p. 1793, 2013.

GITHUB.COM. Tessetact-OCR - ImproveQuality. 2017. Disponível em: < https://github.com/tesseract-ocr/tesseract/wiki/ImproveQuality >. Acesso em: 13 junho de 2017.

LECHETA, Ricardo R. Google Android-3ª Edição: Aprenda a criar aplicações para dispositivos móveis com o Android SDK. Novatec Editora, 2013.

**International Conference on Document Analysis and Recognition**, 1995.

OCH F. J. et al. **Optical character recognition**. Estados Unidos, 2015. Acesso em: 29 abr. 2015

OPENCV.ORG. Eroding and Dilating. Disponível em: < http://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion\_dilatation/erosion\_dilatation.html>. Acesso em: 14 de junho de 2017.

OTSU, Nobuyuki. A threshold selection method from gray-level histograms. **IEEE Transactions on systems, man, and cybernetics**, v. 9, n. 1, p. 62-66, 1979.

REDIG, Annie Gomes; DO COUTO JUNIOR, Dilton Ribeiro. A tecnologia assistiva nos processos de leitura e escrita na educação inclusiva. **Informática na educação: teoria & prática**, v. 15, n. 2, 2012.

RESNIKOFF, Serge et al. Global data on visual impairment in the year 2002. **Bulletin of the world health organization**, v. 82, n. 11, p. 844-851, 2004.

RUSIÑOL, Marçal; CHAZALON, Joseph; OGIER, Jean-Marc. Combining focus measure operators to predict ocr accuracy in mobile-captured document images. In: **Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on**. IEEE, 2014. p. 181-185.

SMITH, R. A simple and efficient skew detection algorithm via text row accumulation. **Proceedings of 3rd**

SMITH, Ray. An Overview of the Tesseract OCR Engine. ICDAR. Vol. 7. No. 1. 2007.

TATHAM, Mark, MORTON, Katherine: Developments in Speech Synthesis, John Wiley & Sons Ltd: England, (2005).

TOROK, Leonardo; DE IMAGENS, Análise; CONCI, Aura. Método de Otsu. **Instituto de Computação–Universidade Federal Fluminense, Niterói–RJ–Brasil, notas de aula**, 2014.

VALA, Miss Hetal J.; BAXI, Astha. A review on Otsu image segmentation algorithm. **International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)**, v. 2, n. 2, p. pp: 387-389, 2013.

WHO, 2013. Visual Impairment and Blindness. World Health Organization. Disponível em: < http://www.who.int/mediacentre/factsheets/fs282/en/>. Acesso em: 08 de setembro de 2017.

WOLF, Christian; JOLION, J.-M. Extraction and recognition of artificial text in multimedia documents. Pattern Analysis & Applications, v. 6, n. 4, p. 309-326, 2004.

YANAGUYA, G. A. Aplicação de Tradução OCR Multiplataforma via Web Services, 2015.

GOTO, Hideaki; LIU, Kunqi. Scene Text Detection and Tracking for Wearable Text-to-Speech Translation Camera. In: **International Conference on Computers Helping People with Special Needs**. Springer International Publishing, 2016. p. 23-26.

1. OCR (Optical Character Recognition) é uma tecnologia utilizada para detecção e extração de caracteres a partir de um arquivo de imagem (OCH, 2015) [↑](#footnote-ref-1)
2. Tesseract OCR ou Tesseract Engine é um *software* de reconhecimento óptico de caracteres *open-source* com distribuição gratuita desenvolvido pela Hewlett Packard e mantido pela Google. [↑](#footnote-ref-2)
3. API (Application Programming Interface) é definido como o conjunto de padrões estabelecidos por um software afim de permitir sua utilização por outros softwares ou sistemas computacionais. [↑](#footnote-ref-3)
4. Binarização de Otsu ou simplesmente Método de Otsu, é um algoritmo que permite a limiarização automática de uma imagem, ou seja, a conversão de uma imagem qualquer em uma imagem em preto e branco, afim de separar objetos em evidência de um fundo em comum (OTSU, 1979). [↑](#footnote-ref-4)
5. OpenCV (Open Source Computer Vision Library) é uma biblioteca multiplataforma, gratuita para uso acadêmico e comercial, para o desenvolvimento de softwares na área de Visão Computacional. [↑](#footnote-ref-5)