Relatório de Projeto

Mestrado em Engenharia Informática – Computação Móvel

Camera Reading For Blind People

Roberto Ferreira Neto

Leiria, *março de 2014*

Relatório de Projeto

Mestrado em Engenharia Informática – Computação Móvel

Camera Reading For Blind People

Roberto Ferreira Neto

Relatório de Projeto de Mestrado realizada sob a orientação do Doutor Nuno Fonseca, Professor da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria.

Leiria, *março de 2014*

Resumo

A ausência de visão torna a vida de um invisual bastante difícil. No entanto, a utilização da tecnologia pode melhorar ligeiramente pequenos aspectos do dia-a-dia. Nesse contexto, o trabalho que se apresenta tem como objetivo a descrição do processo de desenvolvimento de uma aplicação para cegos. O projeto chama-se *Camera Blind For Blind People,* e tem como propósito final o desenvolvimento de uma aplicação que permita a um utilizador cego utilizar um dispositivo móvel para obter a leitura de um texto que esteja escrito numa folha de papel, num sinal, numa parede ou noutro suporte.

Pretende-se com este projeto conceber o protótipo de uma aplicação para iOS, construída a partir da utilização conjunta e integrada de *frameworks* de reconhecimento óptico de caracteres (OCR) e de *frameworks* de sintetização de voz (TTS), que possibilite ao utilizador, recorrendo à câmara de um dispositivo móvel, captar uma imagem e obter a leitura do texto que exista nessa mesma imagem.

O processo de reconhecimento do texto através do OCR será optimizado através da aplicação de filtros nas imagens captadas, realizando desta forma um pré-processamento que permita obter melhorias nos resultados do OCR, para que o texto reconhecido se aproxime, o mais possível, do texto original existente no suporte fotografado. Os filtros utilizados são os disponibilizados pelo sistema operativo iOS, através da sua *framework* de manipulação de imagem, *Core Image*.

A aplicação será afinada e optimizada de forma a oferecer uma melhor usabilidade aos seus utilizadores, nomeadamente nas situações em que a leitura OCR apresente resultados insatisfatórios que tornem imperceptíveis a leitura dos textos reconhecidos.

Palavras-chave: Camera Reading, Blind People, OCR, TTS, iOS

Abstract

Blindness makes life rather difficult for people who suffer from this malfunction. However, the use of technology can slightly improve some smaller aspects of day-to-day life. In this context, the present work focuses on the description of the development of an application for the blind. The project is called Camera Reading for Blind People, and has as its ultimate purpose the development of an application that allows a blind user, using a mobile device, to read a text that is written on a sheet of paper, a signal, a wall or other support.

The aim of this project is to design a prototype of an application for iOS, built from a set of integrated frameworks of optical character recognition (OCR) and frameworks of Text to Speech Synthesis (TTS), which enables the user, using a mobile camera, to take a picture and to get the text reading that exists in the same picture.

The process of recognition of the text by OCR will be optimized by applying filters on the captured images, applying a pre-processing installation to obtain improvements in OCR results, so that the recognized text approximates, as closely as possible, the existing text in the original photographed support. The filters used are provided by the iOS operating system through its framework of image manipulation, *Core Image*.

The application will be tuned and optimized to offer better usability to users, particularly in situations in which reading OCR present unsatisfactory results that make imperceptible reading of recognized texts.

Key-Words: Camera Reading, Blind People, OCR, TTS, iOS

À Minha Família

Agradecimentos

Agradeço à minha família, por tudo. Pelo apoio, pela coragem e segurança que me transmitem e por acreditarem sempre em mim, mesmo nos momentos em que as coisas parecem não correr tão bem.

Agradeço à Sandra, pela companhia, pela dedicação, carinho, amor e por tudo o que tem feito por mim. Tem sido muito importante para a minha motivação, pelo apoio e pelo suporte que me tem dado ao longo do projeto.

Agradeço também ao meu Orientador, pelo apoio e orientação, pela paciência que teve e por me proporcionar a realização deste aliciante projeto.

Índice de Figuras

Índice de Quadros

Lista de Siglas

BLOB –Binary Large OBject, basic large object. Trata-se de uma coleção de dados binários armazenados como uma única entidade.

GPU –Graphics Processing Unit, unidade de processamento gráfico.

iOS – Sistema operativo da Apple utilizado nos dispositivos móveis da marca, como iPhone, iPad ou iPod Touch.

MPEG-4 - Moving Picture Experts Group. Padrão utilizado primeiramente para compressão de dados digitais de áudio e vídeo

OCR –Optical Character Recognition. Refere-se ao reconhecimento óptico de carateres em imagens, extraindo o texto existente nesssas imagens, que pode posteriormente ser editado e utilizado em outras aplicações.

OS X – Operative System X. Sistema operativo da Apple, destinado aos computadores Mac.

PDF – Portable Document Format, trata-se de um formato de ficheiro criado pela empresa Adobe Systems para que qualquer documento seja visualizado, independente de qual tenha sido o programa que o originou.

PNG –Portable Network Graphics, trata-se de um formato de ficheiros gráficos bitmap. Criado para fornecer uma alternatia livre ao formato GIF.

POST - Part of Speech Tagging, também conhecido como marcação gramatical, é o processo de marcação de uma palavra num texto (corpus) como correspondendo a uma parte específica do discurso, baseada tanto sua definição, bem como o seu contexto.

RTF - Rich Text Format, trata-se de um formato de arquivo de documento desenvolvido pela Microsoft, que tem como grande vantagem a possibilidade de ser lido em aplicações de diversos sistemas operativos (Windows, Mac, Linux, Unix, etc.).

SDK - Software Development Kit, ou seja, pacote de desenvolvimento de software.

TIF – O mesmo que TIFF. Tagged Image File Format, que é um formato de ficheiro gráfico bitmap.

TTS –Text-To-Speech. Trata-se da sintetização da voz humana, convertendo texto em linguagem humana audível.

UNLV – Universidade de Nevada, Las Vegas

URL - Uniform Resource Locator, trata-se do o endereço de um recurso (como um arquivo, uma impressora etc.), disponível numa rede; seja a Internet, ou mesmo uma rede corporativa como uma intranet.

Índice

1 – Introdução

Os indivíduos que apresentam uma deficiência visual, não têm capacidade para realizar tarefas visuais (ler, reconhecer rostos). A leitura de textos ou avisos pode ser feita apenas se os mesmos estiverem no sistema de leitura braille, ou através da audição de livros, se estes estiverem em formato áudio ou em formato digital (usando sistemas text-to-speech).

A generalidade das obras publicadas em papel não contemplam a versão braille ou a versão áudio, e as versões digitais ainda são uma clara minoria. Por outro lado, os avisos existentes nas paredes ou nos sinais que nos rodeiam também não possibilitam a leitura por parte de um cego. Assim, o desenvolvimento de uma aplicação que possa realizar a leitura oral de textos, quer estejam numa parede, numa folha de papel ou noutro suporte escrito, assume grande potencialidade e utilidade.

A tecnologia de reconhecimento óptico de caracteres (OCR) possibilita o reconhecimento de textos escritos em suportes físicos variados. Esta tecnologia tem sido largamente utilizada na digitalização de documentos ou fotografias, convertendo-os em cópias electrónicas que podem ser editadas, onde podem ser realizadas pesquisas, reproduzir o seu conteúdo e transportá-lo com facilidade (ELMORE, 2008).

Por seu lado, a tecnologia de síntese de voz (TTS) possibilita que um texto em formato digital (escrito diretamente pelo utilizador, existente numa caixa de texto ou na descrição de um botão), seja sintetizado em voz humana e reproduzido num sistema áudio. O objetivo do TTS é a conversão automática de frases, sem restrições, num discurso falado em linguagem natural, que se assemelhe à forma falada, do mesmo texto, por um indivíduo nativo da língua. Esta tecnologia tem tido avanços significativos ao longo da última década, com muitos sistemas a serem capazes de gerar um discurso sintético muito próximo da voz natural. A pesquisa na área da síntese de voz tem crescido, fruto da sua crescente importância em muitas e novas aplicações (THOMAS, 2007).

O projeto *Camera Reading for Blind People* é um projeto realizado no âmbito do Mestrado em Engenharia Informática - Computação Móvel, desenvolvido na Escola Superior

de Tecnologia e Gestão do Instituto Politécnico de Leiria.

O objetivo deste projeto consiste elaboração de um protótipo de uma aplicação para dispositivos móveis (iOS), que poderá ser utilizada num iPhone/iPod Touch/iPad, que permita a um utilizador invisual utilizar a câmara do dispositivo e obter a leitura de um texto escrito existente na imagem captada.

A realização do trabalho pressupõe a utilização de tecnologia OCR e TTS já existentes, denominadas por *frameworks*, combinando-as para que em conjunto possam fornecer os resultados pretendidos. Assim, este projeto consistirá na construção de uma aplicação constituída por várias partes.

As *frameworks* OCR e TTS para iOS existentes serão analisadas, testadas e optimizadas no sentido de obter a melhor combinação de ambas que possibilitem realizar as tarefas de reconhecimento e sintetização de voz numa única aplicação. O objetivo do trabalho não passa pela investigação de métodos OCR ou TTS, nem aprofundar o conhecimento existente nestas áreas de investigação mas sim a interligação dos módulos já existentes e a sua optimização.

A aplicação resultante deste projeto estará na língua inglesa, e será pensada para o reconhecimento de textos em inglês, ficando a inclusão de outras línguas em aberto para possível trabalho futuro a realizar.

É importante salientar que dificilmente será obtido um sistema perfeito, fruto de várias limitações inerentes ao OCR que, mesmo com imagens em condições perfeitas, apresenta por vezes algumas dificuldades no reconhecimento dos textos, logo, no caso particular deste projeto, conhecerá ainda maiores dificuldades, uma vez que a qualidade das imagens captadas com a câmara de um telemóvel podem influenciar o resultado final, fruto das condições de captação das fotos, nomeadamente ao nível da luz, da focagem ou de cortes nos textos que se pretendem reconhecer.

Mesmo com as limitações que se afiguram este projeto parece interessante e representa uma oportunidade de melhorar a qualidade de vida aos cegos, uma vez que lhes permite a leitura de textos, mesmo que não estando no sistema de leitura braille. Os invisuais passam a ter uma ferramenta que lhes permite ter mais autonomia na leitura e percepção dos textos que os rodeiam, melhorando assim a sua capacidade de compreender e comunicar com o mundo que os rodeia.

1.1 Objetivos

O projeto que tem como objetivo o desenvolvimento de uma aplicação que permita a pessoas cegas escutar a leitura de textos escritos em papel, sinais, paredes ou outros suportes impressos. O utilizador cego inicia a aplicação, aponta um iPhone para o suporte escrito, é feita a captação da imagem do suporte através de um clique no ecrã, realiza-se o reconhecimento do texto através da ferramenta OCR que converte o texto da imagem em texto que pode ser editado e utilizado e a aplicação procede à leitura do texto, através da sintetização de voz.

1.2 Metodologia

Este projeto desenvolve-se em diversas fases, iniciando-se com a introdução ao iOS, nomeadamente com a familiarização com a linguagem Objective-C e o ambiente Xcode.

Seguidamente inicia-se o desenvolvimento da aplicação, com o implementação da funcionalidade de captação de imagem com a câmara e seu registo para posterior utilização.

A próxima fase prende-se com a introdução às ferramentas OCR, através da pesquisa e testes de *frameworks* OCR para iOS, seguindo-se a escolha da *framework* a utilizar no projeto.

Após concluído o processo de reconhecimento óptico do texto constantes nas imagens torna-se necessário desenvolver a sintetização desse mesmo texto em voz humana, audível. Assim, a fase a desenvolver é a implementação do sistema Text-to-Speech.

Por fim, são realizados diversos testes e procura-se optimizar a aplicação nas diversas funcionalidades, melhorando os resultados obtidos a usabilidade e a amigabilidade.

De seguida é apresentado um esquema que representa o funcionamento da aplicação.

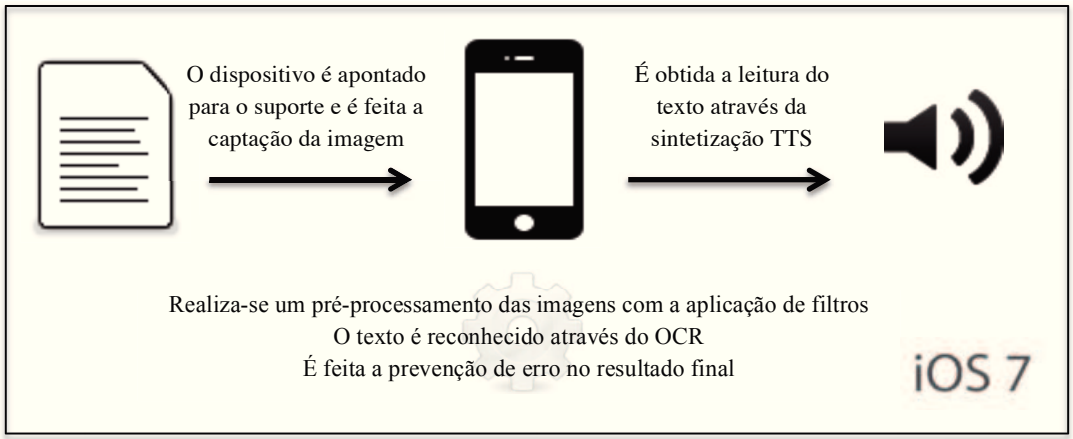


Figura 1 - Esquema de funcionamento da aplicação

2 – Estado da Arte

Neste capítulo serão abordados os conceitos relativos às tecnologias utilizadas na aplicação, nomeadamente o OCR e o TTS. Será feita uma análise individualmente a cada uma das tecnologias, seguindo-se uma análise da sua utilização combinada.

As tecnologias serão analisadas a nível científico, através da análise de trabalhos realizados e estudos desenvolvidos, através da compreensão e estudo do seu funcionamento e também através da análise de aplicações existentes em cada uma destas áreas.

Para cada uma das referidas tecnologias serão estudadas e analisadas diversas *frameworks* disponíveis, que podem ser utilizadas no desenvolvimento de aplicações para equipamentos móveis.

2.1 Reconhecimento óptico de caracteres

O reconhecimento óptico de caracteres, normalmente denominado pela sigla OCR, consiste no processo de reconhecimento e conversão automática de carateres existentes na imagem de um suporte escrito para o formato de texto, que pode ser posteriormente utilizado em diversas aplicações. O OCR tem sido amplamente estudado, verificando-se avanços em relação ao desempenho e à precisão dos resultados obtidos (FUJISAWA, 2007).

O reconhecimento óptico de caracteres pode processar-se de duas formas distintas:

- *Externamente* à aplicação, neste caso o processo de reconhecimento é realizado em servidores externos relativamente ao dispositivo onde se pretende obter o texto. A imagem é captada localmente e enviada para o servidor, sendo depois o texto reconhecido enviado de volta para o dispositivo inicial. Este processo pressupõe a existência de uma ligação à Internet.
- *Localmente,* na própria aplicação, neste caso o reconhecimento do texto é realizado no dispositivo local, sem necessidade de uma ligação à Internet.

O processo de reconhecimento óptico de caracteres pode ser sintetizado como um

processo que decorre num conjunto de passos (EIKVIL, 1993), representados pelo esquema seguinte:

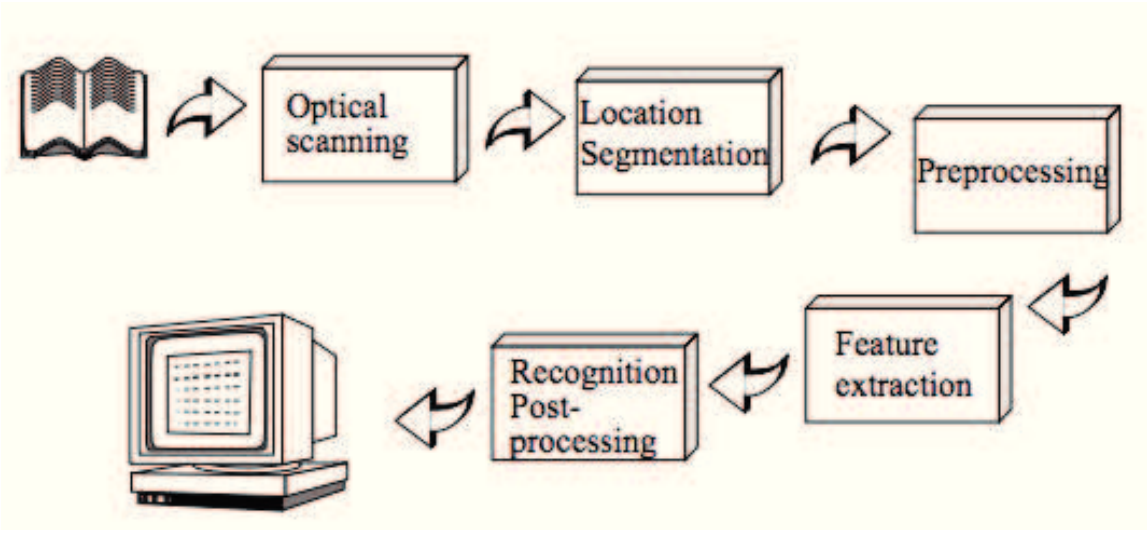


Figura 2 - Componentes de um sistema OCR (EIKVIL, 1993)

Seguidamente são apresentados os passos do reconhecimento óptico, que foram esquematizados na figura anterior e descritos em (EIKVIL, 1993):

Aquisição ótica da imagem

- Realiza-se a aquisição ótica da imagem de um documento, podendo esta ser uma imagem já existente, obtida através da digitalização ou da obtenção de uma imagem através da câmara fotográfica de um dispositivo. Esta etapa está necessariamente dependente do utilizador.

Localização e segmentação

- Neste passo procede-se à localização das regiões que contêm texto, em cada uma dessas regiões é feita a segmentação e são extraídos os símbolos. Nesta fase podem se verificar algumas dificuldades:
    - Obtenção de caracteres unidos ou fragmentados, o que pode fazer com que dois ou mais caracteres unidos sejam interpretados como sendo um único caractere ou um pedaço de um caractere seja interpretado como um símbolo completo.
    - Distinção entre texto e ruído, uma vez que os sinais de pontuação podem ser interpretados como uma imagem, ou como ruído. O seu contrário também acontece, uma vez que uma imagem ou ruído podem ser interpretados como sinais de pontuação.
    - Interpretar imagem ou gráfico como texto, o que inclui no resultado final

reconhecido texto que não existe.

  - Interpretar texto como gráfico ou imagem, fazendo com que o OCR não reconheça o texto que efetivamente existe. Esta situação é mais frequente quando o texto e a imagem encontram-se muito próximos.

Pré-processamento

- De forma a evitar os problemas anteriormente elencados, cada símbolo extraído pode passar por um pré-processamento que envolve a eliminação de todo o ruído possível a fim de facilitar a extração das características deste símbolo.

Extração de características

- O objetivo deste passo é a obtenção das características essenciais dos símbolos reconhecidos, que é uma das principais dificuldades do processo de reconhecimento de padrões.
- A forma mais direta de descrever o caractere é pela leitura dos *pixels* que a compõem.
- Outra forma de o fazer é extrair determinados atributos que caracterizam os símbolos, deixando de parte atributos menos importantes. Estas técnicas dividem-se em três grupos:
  - Distribuição dos pontos
  - Transformações e expansões em série
  - Análise estrutural

Classificação

- Os símbolos obtidos são identificados e classificados ao comparar as características pertencentes ao mesmo, com as descrições das classes de símbolos que são obtidas através de uma fase de aprendizagem, ocorrida antes do início do processo de OCR.

Pós-processamento

- O resultado obtido no reconhecimento é um conjunto de símbolos, que por si só não contêm informação suficiente. Pretende-se pois, associar os símbolos que pertençam à mesma palavra, construindo o texto. O processo de associação de símbolos em *strings*, palavras, é normalmente referido como agrupamento (do inglês *grouping*). Este agrupamento é feito com base na localização desses símbolos no documento. Assim, símbolos que estão mais próximos uns dos outros são agrupados numa só *string*.
- O texto final é então convertido num documento com o formato pretendido (*rtf,*

txt, pdf, etc.).

O texto obtido pode ser utilizado para impressão, para introdução em outros documentos ou num processo de sintetização de voz para obtenção da leitura do texto. Esta última utilização possibilita a pessoas cegas o acesso à informação escrita sem a necessidade de intervenção de outras pessoas.

Em *OCRdroid: A Framework to Digitize Text Using Mobile Phones* (JOSHI, 2009), é abordado funcionamento do sistema de digitalização de imagens para obtenção do texto nelas contido. Trata-se de um projeto cujo objectivo é o desenvolvimento de uma aplicação para *Android*, que faz um pré-processamento das imagens aquando da obtenção das mesmas, corrigindo a luminosidade existente e verificando se as imagens estão corretamente posicionadas para melhor reconhecimento do texto, e de seguida envia a imagem obtida para um servidor externo onde se processa o reconhecimento OCR.

2.1.1 Tesseract OCR

O Tesseract é um sistema OCR de código aberto, desenvolvido pela HP entre 1984 e 1994. O seu aparecimento deu-se pela primeira vez em 1995 na UNLV (Universidade de Nevada, Las Vegas) *Annual Test of OCR Accuracy*.

Tesseract começou por ser um projeto de pesquisa realizado no âmbito de um doutoramento no HP Labs e Bristol e ganhou importância como um possível software e/ou hardware adicional para a linha de scanners de mesa da HP. A motivação ficou a dever-se ao facto de os sistemas OCR comerciais da época estarem ainda numa fase muito embrionária e apresentavam muitas falhas no que diz respeito à qualidade.

Fruto de um trabalho conjunto entre a HP Labs Bristol e a divisão de scanners da HP no Colorado, o Tesseract apresentava resultados significativamente melhores do que outros produtos comerciais, mas não se tornou num produto. Em 1995 o sistema foi submetido ao UNLV para o evento *Annual Test of OCR Accuracy* (Teste Anual de Precisão de OCR) onde demonstrou o seu valor em relação aos outros sistemas comerciais da altura. Em 2005, a HP disponibilizou o Tesseract como sendo código-aberto (SMITH, 2007).

O funcionamento do Tesseract pode ser sintetizado da seguinte forma (SMITH, 2007):

- São analisados os contornos das imagens existentes
- Esses contornos são reunidos em Blobs (do inglês, *Binary Large OBject, basic large object*)
- Os Blobs estão organizados em linhas de texto

- As linhas de texto são analisadas, e as palavras que as constituem são divididas de acordo com o tipo de espaçamento entre caracteres
- É então feita a tentativa de reconhecer uma palavra de cada vez
- Cada palavra considerada como satisfatoriamente reconhecida é classificada de forma adaptativa como dados de treino. Esta classificação e aprendizagem possibilitam, numa segunda análise, o reconhecimento mais eficaz das palavras que não foram reconhecidas na primeira análise.
- Seguidamente são resolvidas as situações de espaços distorcidos, e são feitas tentativas de reconhecimento de carateres em maiúsculas pequenas (*small caps*)
- O texto digital é finalmente obtido em formato digital para posterior utilização.

Durante os passos anteriormente indicados, o Tesseract faz uso de vários algoritmos:

- Algoritmos para detecção de linhas de texto numa página cuja imagem esteja enviesada;
- Algoritmos para detecção de palavras proporcionais e não proporcionais (considera-se palavra proporcional uma palavra em que todos os caracteres têm a mesma largura);
- Algoritmos para separar carateres unidos e para associar caracteres partidos, e a respetiva análise linguística para identificar a palavra mais provável tendo por base conjuntos de caracteres;
- Dois classificadores de caracteres: um classificador estático e um classificador adaptativo que emprega os dados de treino, e que é melhor na distinção entre letras maiúsculas e letras minúsculas.

2.1.2 Abbyy OCR

A ABBYY foi fundada em 1989 por David Yang, que é atualmente o presidente do conselho de administração da ABBYY. A sede da empresa encontra-se em Moscovo existindo escritórios de representação em diversos países da Europa (Alemanha, Reino Unido, Chipre, Rússia, e Ucrânia) América (EUA e Canadá), Ásia (Japão) e Austrália.

Um dos seus produtos é o ABBYY FineReader (*Abbyy FineReader 11 Professional Edition*), que é um software OCR comercial para reconhecimento de imagens e respetiva criação de ficheiros com textos pesquisáveis e editáveis. O sistema está disponível em duas versões: *Professional*, indicada para utilizadores individuais e a *Corporate*, mais indicada para utilização empresarial. A maior diferença é o suporte fornecido aos utilizadores da versão *Corporate*.

ABBYY também dispõe de uma versão SDK que permite integrar o seu mecanismo OCR em outras aplicações de reconhecimento e conversão de texto.

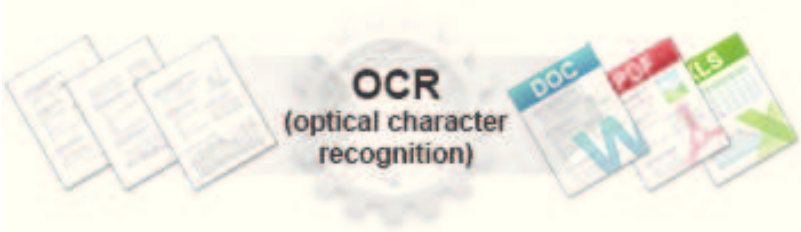


Figura 3 - Esquema funcionamento ABBYY OCR

Das versões SDK existentes, salienta-se a versão do sistema OCR disponível para dispositivos móveis que permite a integração da tecnologia OCR ABBYY em aplicações para estes dispositivos. Um exemplo da utilização deste SDK é em aplicações que permitem realizar a tradução de textos reconhecidos de imagens, TextGrabber + Translator (TEXTGRABBER + TRANSLATOR).

Os mecanismos exatos que permitem aos seres humanos reconhecer objetos ainda estão para serem entendidos, porém, os três princípios básicos são bem conhecidos pelos cientistas – integridade, fim e adaptabilidade (IPA). Esses princípios constituem o centro do ABBYY FineReader OCR permitindo que este replique o reconhecimento natural como o ser humano.

O funcionamento do ABBYY FineReader é o seguinte: primeiro, o programa analisa a estrutura da imagem do documento. Então, divide a página em elementos como blocos de texto, tabelas, imagens, etc. As linhas são divididas em palavras e, então, em caracteres. Uma vez separados os caracteres, o programa compara-os com um conjunto de imagens padrão. São avançadas inúmeras hipóteses sobre esse caractere. Com base nessas hipóteses, o programa com tecnologia OCR analisa variantes diferentes de quebra de linhas em palavras e palavras em caracteres. Após processar um enorme número de tais hipóteses prováveis, o programa finalmente age e toma uma decisão, apresentando o texto reconhecido (Abbyy).

A utilização da SDK ABBYY está dependente da solicitação de uma licença *trial* (licença de demonstração que permite utilizar o SDK por um período de tempo) realizada através do preenchimento de um formulário online com consequente contacto com a empresa. A licença fornecida destina-se a um utilizador e tem uma duração de três meses. O custo da licença para utilização do SDK ABBYY sem limitações tem o custo de 4.900,00€.

2.1.3 Leadtools OCR

A empresa que desenvolve o sistema OCR Leadtools, a LEAD Technologies, Inc., foi fundada por Mohammed Daher e Rich Little, em 1990. A sede da empresa situa-se em Charlotte, na Carolina do Norte, EUA.

A gama de produtos Leadtools inclui um conjunto de ferramentas abrangentes para ajudar os programadores a integrar sistemas de reconhecimento óptico de documentos, de imagens médicas, imagens multimédia e vetoriais em aplicações destinadas ao uso em computadores desktop e dispositivos móveis (LEADTOOLS).

Entre os diversos produtos fornecidos por esta empresa, destaca-se o Leadtools *Professional OCR Module*, que pode ser integrado em aplicações móveis para reconhecimento óptico de caracteres em imagens.

A utilização da SDK Leadtools está dependente do preenchimento de um formulário com o consequente pedido de uma licença *trial* (licença de demonstração que permite utilizar o SDK por um período de tempo). O custo da licença de utilização do SDK é de 995$ (aproximadamente 735,00 €).

2.1.4 OpenRTK OCR SDK

A empresa que desenvolveu o sistema OpenRTK é a ExperVision, fundada na Califórnia em 1987 (EXPERVISION).

O sistema OCR OpenRTK é independente do sistema operativo e pode ser migrado entre sistemas operativos. Neste momento encontra-se disponível para os dispositivos iPhone, iPad, Android, WinCE, entre outros (OpenRTK OCR SDK).

A utilização deste SDK está dependente do pedido de uma licença de utilização trial, seguida da compra da licença. O preço indicado para aquisição da licença na página da empresa é de 5.190$.

2.1.5 Online OCR API

A ferramenta *Online OCR API* diverge das restantes por se tratar de um sistema de reconhecimento realizado externamente à aplicação. Ou seja, a aplicação móvel realiza a captação da imagem, e envia essa imagem para o servidor online que realiza o reconhecimento óptico, devolvendo de seguida o texto reconhecido. O sistema permite o

reconhecimento de imagens em diversos formatos: jpg, png, gif, tif and pdf.

Uma vez que a plataforma online apenas realiza o reconhecimento do texto existente nas imagens, não realizando nenhum pré-processamento das mesmas, para obtenção do melhor resultado possível, são fornecidas algumas indicações que as imagens a ser enviadas devem cumprir:

- Boa iluminação;
- Fundo branco;
- Alta definição com carateres grandes;
- O espaçamento do texto não deve ser muito pequeno;
- O texto deve estar alinhado na horizontal.

A subscrição e utilização do serviço pressupõe a compra de pacotes que contêm um determinado número de operações de reconhecimento. Por exemplo, por 50$ é possível realizar 3000 reconhecimentos (OCRAPISERVICE).

2.2 Síntese de voz

A síntese de voz, definida como TTS (sigla para *Text-To-Speech*) é um sistema computacional que deve ser capaz de ler em voz alta qualquer texto, independentemente da sua origem (DUTOIT, 1997).

A utilização do TTS é a tentativa de produzir a voz humana de forma artificial. O sistema informático reproduz os sons, fazendo a leitura de um texto, procurando recriar uma voz natural. A entoação confere um maior realismo à leitura, quando a sintetização inclui corretamente pausas para respirar e/ou respeita a pontuação do texto.

A síntese de voz é um processo complexo, sendo necessários algoritmos complexos para produzir um resultado inteligível e natural.

No contexto da síntese TTS é impossível gravar e armazenar todas as palavras de uma determinada língua. Portanto, é mais adequado definir o TTS como produtor automático da fala, através da transcrição grafema para fonema da frase a proferir (DUTOIT, 1997).

A sintetização TTS faz uso de técnicas do Processamento de Linguagem Natural. Sendo o texto a ser sintetizado a primeira entrada do sistema, deverá ser o primeiro a ser processado. Segundo (CROCHIERE ,1990), o processo de síntese de voz pode ser esquematizado através da figura apresentada na página seguinte:

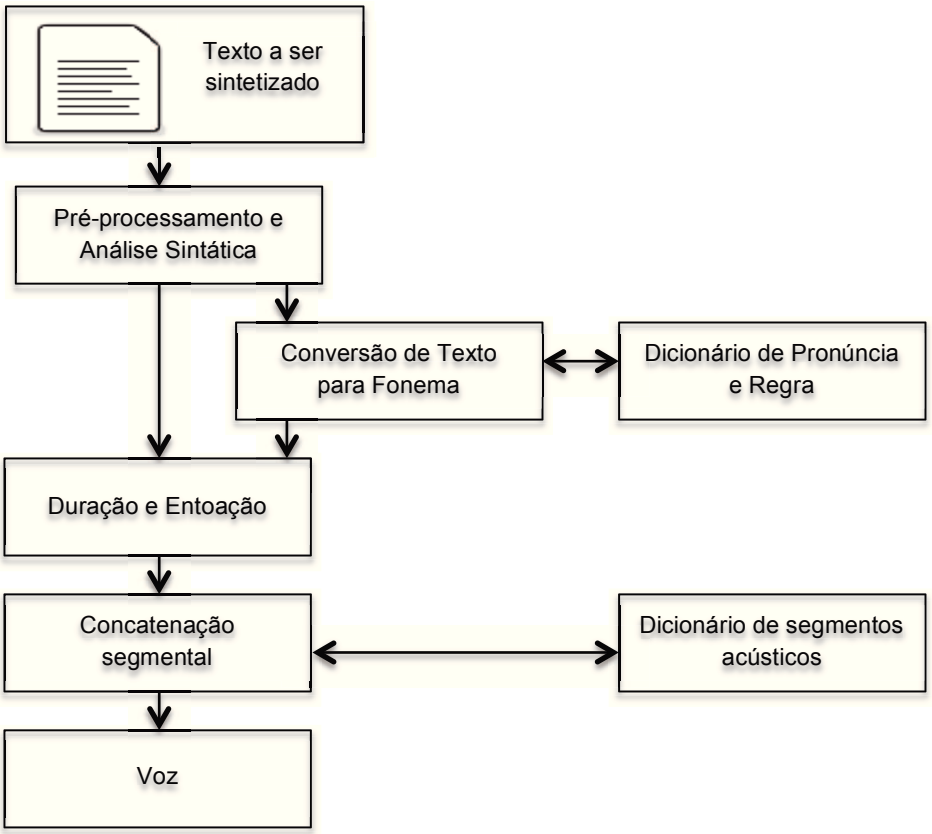


Figura 4 - Elementos de um sistema de síntese de voz a partir de texto

O processo é sequencial e acontece em várias fases (U.D REICHEL , 2006), seguidamente descritas:

Através da normalização do texto, o texto de entrada é adaptado de forma a ser sintetizado. Trata-se pois de um pré-processamento.

A segmentação da frase torna-se um processo um tanto complexo, na medida que tem que lidar com os sinais de pontuação, mas pode ser obtida através da utilização de árvores de decisão simples. Existem no entanto situações mais complexas, passíveis de gerar ambiguidades, que exigem métodos mais complexos. Alguns exemplos dessas dificuldades são a existência do ponto, que pode ser utilizado para indicar o fim da frase mas também uma abreviatura ou a dificuldade em diferenciar letras maiúsculas em nomes próprios e no início de frases, as abreviaturas, acrónimos, etc.

A tokenização é utilizada para decompor o texto em cada unidade que o compõe. O texto é dividido nos espaços entre palavras e entre os sinais de pontuação. Este processo é realizado recorrendo a um processo de análise sintática, que através da análise da sequência de entrada determina a sua estrutura gramatical segundo uma determinada gramática formal. Através da análise lexical é obtido um conjunto de *tokens,* utilizados pelo analisador sintático, que recorrendo a um conjunto de regras constrói uma árvore sintática da estrutura.

A próxima fase é o POST (do inglês, *Part of Speech Tagging*), que consiste em atribuir classes a cada palavra. Nesta fase é necessário lidar com palavras desconhecidas e palavras com etiquetas POST ambíguas (mesma estrutura da frase) como nomes, verbos e adjetivos. De referir que um nome pode tomar o papel de adjetivo e um adjetivo pode tomar o papel de nome, como por exemplo:

- O Roberto Ferreira Branco é um indivíduo simpático. (A palavra "Branco" é um nome).
- O Roberto Ferreira está branco, estará mal disposto? (A palavra "branco" é um adjetivo).

Na próxima fase, a da conversão do grafema para fonema, é assinalado a cada *token* o conjunto fonético correto. De referir que este é um processo dependente do idioma, uma vez que as transcrições fonéticas de cada *token* são influenciadas pelas transcrições do *token* vizinho.

A fase final é inerente à pronúncia das palavras, que é igualmente inerente à língua que se esteja a trabalhar. Trata-se de identificar a sílaba tónica, em inglês *word stress*. Esta identificação é fortemente influenciada pelas características fonológicas, morfológicas e de classe das palavras (U.D REICHEL , 2006).

Para gerar a voz sintetizada, segundo (CONDADO, 2009) existem quatro principais técnicas, seguidamente descritas:

Síntese articulatória

- Este método utiliza algoritmos que simulam o sistema vocal humano, simulando a articulação das cordas vocais, da glote, da mandíbula, da língua e dos lábios. Apesar desta técnica ser extremamente interessante, visto que a criação de modelos computacionais que simulem os diversos órgãos do nosso sistema vocal abre imensas possibilidades, não é a mais utilizada. Pode-se dizer que a utilização de outros métodos em detrimento deste está relacionada com o facto da síntese articulatória requerer muitos recursos computacionais e, pelo menos na atualidade, não apresentar resultados satisfatórios ao nível da naturalidade e de fluência do discurso gerado.

Síntese por formantes

- Contrariamente à síntese articulatória, que apresenta a desvantagem inerente a uma técnica que tenta modelar o sistema de produção de voz dos seres humanos (system-models), as técnicas que se baseiam na modelação dos sinais resultantes do processo de fala (signal-models) são bastante mais utilizadas na área da síntese

de voz.

- Uma das técnicas que é amplamente utilizada nesta área é conhecida pela denominação síntese por formantes. Este método baseia-se num conjunto de regras que determinam os parâmetros necessários para sintetizar uma dada expressão, nomeadamente os relacionados com as frequências de formantes, anti-formantes, e com a amplitude.
- Os sintetizadores por formantes são especialmente utilizados em aplicações onde o output não é previsível, como sistemas TTS ou leitores de ecrã, gerando resultados de elevada inteligibilidade. Apesar do discurso gerado não ser completamente natural, notando-se que é gerado por uma máquina, os recursos computacionais exigidos por este processo de síntese são aceitáveis.

Síntese por concatenação

- Este método de síntese é mais adequado para ser implementado em aplicações onde se conhece previamente os inputs possíveis, como nos sistemas de navegação, nos sistemas de atendimento automático, nas caixas multibanco, e em sistemas semelhantes que desejem ter a naturalidade fornecida por registos vocais pré-gravados. A técnica de síntese por concatenação, também conhecida por síntese concatenativa, baseia-se na união de segmentos de fala que se encontram armazenados numa base de dados. A qualidade resultante deste método de síntese depende da forma como a junção desses segmentos (fonemas) é realizada, nomeadamente se é aplicada a estratégia de seleção de unidades, de difones, ou a do domínio específico. Claro que, aliados a este factor, também existem outros que têm impacto no resultado de um sintetizador baseado nesta técnica, como a eficiência do algoritmo implementado para efetuar a pesquisa na base de dados.

Síntese baseada nos modelos ocultos de Markov

- A técnica de síntese baseada nos modelos ocultos de Markov (Hidden Markov Models) tem sido muito utilizada nos últimos anos por apresentar bons resultados. Em linhas gerais, este método apresenta resultados de qualidade inferior à técnica de síntese concatenativa por seleção de unidades, mas tem a vantagem de requerer bases de dados de menor dimensão. Um modelo oculto de Markov é um modelo estocástico em que se modela um processo Markoviano cujos estados são desconhecidos.

As principais técnicas de síntese, acima abordadas, são os métodos mais utilizados no estudo e no desenvolvimento de sistemas de síntese de voz. Contudo, uma forma de tirar

partido das vantagens inerentes a cada técnica é utilizar um híbrido das várias técnicas no desenvolvimento de futuros sistemas de síntese de voz.

A qualidade de um síntese de voz pode ser determinada pela naturalidade e a inteligibilidade, sendo a naturalidade a característica que descreve o quão próximo o som obtido pelo TTS está da voz humana e a inteligibilidade refere-se à facilidade com que o som é compreendido em situações complexas (DUTOIT, 1997 e LEMMETTY, 1999).

Outros autores defendem que a qualidade de um sistema TTS é frequentemente determinado por quatro características (COHEN, 2004):

- Inteligibilidade: a facilidade como um utilizador entende o que foi dito. Este é o factor de medição de qualidade mais importante num discurso sintetizado.

- Naturalidade: o quanto o discurso sintetizado parece com voz humana.

- Precisão: está relacionado com a correção do que foi sintetizado, ou seja, se o texto é corretamente pronunciado.

- Facilidade de escuta: relacionado com a facilidade ou prazer em ouvir o discurso, não havendo fadiga em fazê-lo.

Observando estas quatro características, percebe-se que estão relacionadas, não sendo nenhuma independente das restantes. Por exemplo, se forem verificados erros na precisão, a inteligibilidade será menor, o discurso será percebido como sendo menos natural e a facilidade de escuta do discurso sintetizado fica pior.

2.2.1 OpenEars TTS

OpenEars TTS (OPENEARS) é uma *framework* iOS de código partilhado, que permite realizar o reconhecimento de voz e a síntese de fala (TTS).

Permite a sintetização de voz em Inglês no iPhone, iPod e iPad e utiliza o código aberto CMU (*Carnegie Mellon University*) Pocketsphinx, CMU Flite e bibliotecas CMUCLMTK, que podem ser consultadas online em http://www.cmusphinx.sourceforge.net/wiki/cmuclmtkdevelopment e é livre para utilização em aplicações para iPhone, iPad ou iPod.

Trata-se de uma *framework* que pode ser utilizada localmente no dispositivo onde está instalada a aplicação, ou seja, sem necessidade de uma ligação de dados, e tem sido destaque em livros de desenvolvimento, tais como *O'Reilly's Basic Sensors in iOS by Alasdair Allan and Cocos2d for iPhone 1 Game Development Cookbook by Nathan Burba*.

2.2.2 iSpeech TTS API

A *API iSpeech* (ISPEECH) permite o desenvolvimento de aplicações que implementem a sintetização de voz e o reconhecimento automático de voz, em dispositivos que possuam ligação à Internet.

O envio de dados pode ser feito através dos métodos GET e POST do HTML. Os pedidos podem ser codificadas em URL, JSON, ou formatos de dados XML, podendo o formato desejado ser especificado pelo programador. A voz obtida através do processo de sintetização pode variar em vários aspectos, tais como várias vozes, formatos, velocidades, etc., que são igualmente definidas pelo programador.

Trata-se de uma *framework* que necessita de uma ligação de dados para ser utilizada.

A subscrição e utilização do serviço pressupõe a compra de pacotes que contêm um determinado número de créditos que permitem a realização de operações de síntese de voz, sendo que cada crédito permite realizar a sintetização de uma palavra. Assim, e a título de exemplo, por 200$ é possível sintetizar 10000 palavras.

2.2.3 Google TTS Library for iOS

O Google TTS possibilita a criação de aplicações iOS, que utilizem os serviços Google TTS para realizar a sintetização de voz. Apresenta uma limitação de 100 caracteres para cada sintetização.

A sintetização de voz é feita externamente à aplicação, e apresenta uma limitação de 100 caracteres para cada operação. A aplicação envia o texto para o servidor da Google, que o sintetiza em voz e cria um ficheiro mp3 e de seguida volta a enviar o ficheiro mp3 para a aplicação cliente.

Existe um projeto que adaptou esta *framework* para iOS, permitindo que seja utilizada localmente no próprio dispositivo, e sem restrições quanto ao limite de caracteres (GOOGLE TTS). A utilização da *framework* resume-se à descarga de um pacote de arquivos, a sua adição no projeto, a importação da biblioteca e a chamada do método que realiza a sintetização.

```
- (IBAction)synthesize:(id)sender {

    self.google_TTS_BySham = [[Google_TTS_BySham alloc] init];
    [self.google_TTS_BySham speak:@"Camera Reading for Blind People"];

}
```

Figura 5 - Exemplo de utilização da *framework* Google TTS

2.2.4 Acapela TTS

Acapela TTS (ACAPELA) para iOS é um sistema desenvolvido para a comunidade de desenvolvimento iPod Touch, iPhone e iPad, oferecendo um sistema de sintetização de voz.

O funcionamento deste sistema é o que de seguida se apresenta (ACAPELA).

- Para reproduzir o som natural de cada linguagem, um narrador regista uma série de textos (poesia, notícias políticas, resultados desportivos, atualizações de bolsa, etc.), que contêm todos os sons possíveis no idioma escolhido.
- Essas gravações são então cortadas e organizadas numa base de dados acústicos.
- Durante a criação da base de dados, todo o discurso gravado é segmentado em: fonemas, sílabas, morfemas, palavras e frases.
- Para reproduzir palavras de um texto, o sistema TTS começa pela realização de uma análise linguística que transpõe o texto escrito em texto fonético.
- A análise gramatical e sintática, em seguida, permite que o sistema defina como pronunciar cada palavra, a fim de lhe dar sentido. Chama-se a isso de prosódia, ou seja, dar ritmo e entonação a uma frase.
- Finalmente, o sistema produz informações associando a escrita fonética com o tom e o comprimento necessário da pronúncia .
- A cadeia de análise termina e o som é gerado selecionando as melhores unidades que se encontram na base de dados acústica.

Através do registo no website da empresa, é possível obter uma versão da SDK para avaliação. Para poder utilizar comercialmente a SDK é necessária a inscrição como desenvolvedor, que tem o custo de 1000,00€ por ano, e ainda o pagamento de *royalties* que variam percentualmente com base no preço da aplicação e na quantidade de aplicações vendidas.

2.3 Combinação OCR e TTS

Existem alguns trabalhos sobre a temática da captação de imagem com a câmara de um dispositivo móvel, a obtenção do texto existente nessa imagem para diversas utilizações, entre elas, a sua sintetização através do TTS. No entanto, esses trabalhos têm sido fundamentalmente pensados para sistemas Android, embora naturalmente o que mais interessa seja a sua fundamentação e a análise científicas, o que possibilita que seja feita a transposição para sistemas iOS.

Em *The Phone Reader* (BIHINA, 2012), é feita a abordagem a um sistema desenvolvido para o sistema Android, que permite reconhecer o texto existente numa foto, utilizando a OCR Tesseract, e através do TTS procede à leitura do texto traduzido na língua que for selecionada. O processamento OCR e a tradução são feitos num servidor externo, o que implica uma ligação de dados para a comunicação entre o servidor e o dispositivo.

Na publicação, *Optical Character Recognition* (MITHE, 2013), é apresentada uma aplicação para dispositivo móvel Android, que realiza o reconhecimento do texto existente em fotos obtidas com o dispositivo através do OCR Tesseract, e após o seu reconhecimento é feita a sintetização de voz. Nesta aplicação, todo o processamento é feito localmente no dispositivo móvel, não havendo lugar a qualquer pré-processamento ou reconhecimento de limites das imagens, ficando estes definidos para trabalho futuro.

2.4 Aplicações existentes

Existem atualmente diversas aplicações que realizam o reconhecimento óptico de textos. Algumas apresentam resultados satisfatórios, permitindo que o texto obtido seja utilizado de diversas formas, no entanto a maioria destas ferramentas possibilita apenas a utilização do texto para edição e posterior reutilização, verificando-se existir poucas aplicações realizam a sintetização do texto em voz humana.

2.4.1 Aplicações OCR

Nesta secção serão apresentadas as principais ferramentas disponíveis para iPhone.

Scanner Pro, disponível online em https://itunes.apple.com/PT/app/id333710667, é uma aplicação de digitalização que possibilita a digitalização de documentos, cartões de visita, recibos, notas e quadros brancos. Scanner Pro permite trabalhar com várias páginas e inclui ferramentas para mover, excluir ou combinar páginas. Quando terminar a digitalização, o arquivo PDF resultante pode ser partilhado através da Dropbox, Evernote, iDisk, ou qualquer servidor WebDAV. O documento resultante pode também ser enviado por email.

Figura 6 - Ícone da aplicação Scanner Pro

JotNot Scanner Pro, disponível online em https://itunes.apple.com/PT/app/id307868751, inclui suporte multi-página, a deteção de limites de página, suporte para fax, estabilização de câmara, suporte a PDF / PNG / JPEG. A aplicação permite o envio das digitalizações para a Dropbox, iDisk, Google Docs, Box.net.

Figura 7 - Ícone da aplicação JotNot Scanner Pro

Image to Text – OCR, disponível online em https://itunes.apple.com/PT/app/id431757093 permite tirar uma foto de uma folha papel e reconhecer o texto via OCR. Mas neste caso o processo de OCR real não acontece no dispositivo local, mas em servidores separados, o que torna o processo muito mais rápido e mais fácil em sua CPU, embora exiga uma ligação de Internet. O texto obtido pode ser enviado por e-mail ou utilizados numa conta Evernote. Trata-se de uma aplicação gratuita.

Figura 8 - Ícone da aplicação Image to Text – OCR

DocScanner, disponível online em https://itunes.apple.com/PT/app/id312391317, possui suporte OCR e melhoramentos de imagem, permitindo o corte, afiação, e correção do balanço de cores e da distorção. A aplicação permite a partilha dos arquivos através Dropbox, Evernote, Google Docs, WebDAV, e iDisk, mas também através da uma ligação Wi-Fi local.

Figura 9 - Ícone da aplicação DocScanner

CamScanner Free, disponível online em https://itunes.apple.com/PT/app/id388627783, é uma aplicação gratuita que permite a partilha do documento obtido pela leitura OCR com Google Docs, Evernote, Dropbox, iDisk, e-mail e Box.net. A aplicação permite a leitura de várias páginas, realizar cortes e ajustes de cor e brilho.

Figura 10 - Ícone da aplicação CamScanner Free

Genius Scan – PDF Scanner, disponível online em https://itunes.apple.com/PT/app/id377672876, é mais uma opção gratuita para o iPhone. A aplicação tem suporte multi-página, partilha via Wi-Fi, integração iBooks e suporte PDF / JPG.

Figura 11 - Ícone da aplicação Genius Scan - PDF Scanner

Perfect OCR, disponível online em https://itunes.apple.com/PT/app/id363095388, é uma aplicação de OCR que, segundo a descrição fornecida, apresenta muita precisão de leitura, sendo um scanner de documentos de alta qualidade, com uma precisão de 90% ou mais. Possibilita editar, copiar e armazenar o texto reconhecido, e enviar por email documentos como arquivos PDF com texto, imagem, ou ambos.

Figura 12 - Ícone da aplicação Perfect OCR

TurbosScan, disponível online em https://itunes.apple.com/PT/app/id342548956, é uma aplicação que inclui suporte AirPrint, detecção de limites, numeração de documento e funcionalidade "SureScan", que trata-se de um método que permite tirar várias fotos, e a aplicação automaticamente seleciona a melhor. A única opção de partilha dos documentos é através do email.

Figura 13 - Ícone da aplicação TurbosScan

Pocket Scanner - Documents on the go, disponível online em https://itunes.apple.com/PT/app/id359793767, permite ajustar a exposição, suporta a leitura de várias páginas, scans de visualização em formato PDF, scan de emails e importar imagens a partir da biblioteca de fotos.

Figura 14 - Ícone da aplicação Pocket Scanner - Documents on the go

Page Scanner, disponível online em https://itunes.apple.com/PT/app/id368331275, é uma aplicação que tanto faz de scanner como de aplicação OCR. A leitura OCR apresenta bons resultados, embora apresente limitações em questões como detecção de limites de páginas. A aplicação permite realizar o scan de emails, o envio através do iTunes, ou o acesso a uma rede Wi-Fi local.

Figura 15 - Ícone da aplicação Page Scanner

Doc Scan Pro - Scanner to Scan PDF, Print, Fax, Email, and Upload to Cloud Storages, disponível online em https://itunes.apple.com/PT/app/id478861246, é uma aplicação que tem como forte características a detecção das das páginas de livros e criar pontos onde as páginas curvam. Assim a aplicação verifica que está a digitalizar a partir de um livro e adapta-se a essa situação. Permite também ajustar o brilho e o contraste depois da digitalização de uma página, como forma de obter melhores resultados. Doc Scan Pro suporta o envio e partilha de PDFs e imagens para Dropbox, Box, Evernote, SkyDrive e Google Drive.

Figura 16 - Ícone da aplicação Doc Scan Pro

FineReader Touch, disponível online em https://itunes.apple.com/PT/app/id565102038, é uma aplicação que permite reconhecer textos em diversas línguas, possibilitando que os resultados sejam apresentados em diversos formatos tais como DOC/DOCX, TXT, PDF entre outros. Permite também recortar, girar e agrupar documentos. Os resultados podem ser exportados diretamente para o Evernote e Yandex.Disk.

Figura 17 - Ícone da aplicação FineReader Touch

2.4.2 Aplicações OCR + Text to Speech

Existem disponíveis na *App Store* da Apple algumas aplicações que realizam o reconhecimento de textos através de OCR textos e realizam a sua leitura. Em todos os casos as aplicações recorrem à funcionalidade *VoiceOver* para obter a leitura dos textos.

SayText, disponível online em https://itunes.apple.com/PT/app/id376337999, é uma aplicação que permite tirar fotografias a textos e após fazer o reconhecimento do texto, permite o envio do mesmo por email ou obter a leitura do texto, recorrendo à funcionalidade do sistema *VoiceOver*. É uma aplicação gratuita.

Figura 18 - Ícone da aplicação SayText

Talking Camera Pro - for visually impaired/blind, disponível online em https://itunes.apple.com/PT/app/id544295997, é mais uma aplicação que permite o reconhecimento de textos através de OCR e a sua leitura através da funcionalidade *VoiceOver*. Esta aplicação tem um custo de 1,79€.

Figura 19 - Ícone da aplicação Talking Camera Pro

Prizmo - Escaneamento, OCR & Fala, disponível online em https://itunes.apple.com/PT/app/id366791896, é uma aplicação permite reconhecer documentos de texto, cartões de visita e imagens, podendo o resultado ser exportado como PDF/Texto, vCard ou JPEG/PNG. A aplicação permite o processamento das imagens obtidas, podendo o utilizador rodar, recortar, corrigir a perspectiva e limpar a imagem (remoção de iluminação desigual, textura do papel...). Através da funcionalidade *VoiceOver* é possível a leitura dos textos. Esta aplicação tem um custo de 8,99€.

Figura 20 - Ícone da aplicação Prizmo

2.5 Conclusão

Não obstante o facto de existirem diversas aplicações que utilizam *frameworks* OCR e TTS, separadamente ou de forma mais ou menos combinada, verifica-se que as soluções existentes apresentam limitações diversas, que se prendem com a sua fraca performance, ou com o seu custo elevado ao utilizador.

O projeto *Camera Reading for Blind People*, reveste-se assim de importância e pertinência, uma vez que vem lançar as bases para o desenvolvimento de uma aplicação que possa servir de forma mais capaz, e com menos custos, as necessidades de invisuais.

De referir que, ao contrário da generalidade das aplicações e projetos desenvolvidos, o projeto que aqui se apresenta está diretamente dirigido e pensado para utilizadores invisuais, podendo desta forma cumprir de forma mais eficaz a missão a que se destina.

3 – Metodologia

Nesta secção será apresentada e descrita a metodologia desenvolvida na prossecução do projeto, desde as tarefas mais elementares até às mais complexas.

Uma vez que o objetivo deste trabalho é o desenvolvimento do protótipo de uma aplicação para cegos, pretende-se tirar o máximo partido da utilização da tecnologia OCR associada à síntese de voz, para que em conjunto possa ser encontrada uma solução capaz de responder ao desafio proposto.

3.1 Introdução ao iOS

Tratando-se de projeto a realizar para um Sistema Operativo com o qual não tinha contactado anteriormente, foi imperativo proceder a um estudo introdutório de forma a conhecer a linguagem Objective-C, e conhecer o ambiente de desenvolvimento da Apple, o Xcode.

Figura 21 – Imagem do símbolo logótipo do *Xcode Welcome Screen*

Objective – C é uma linguagem de programação utilizada no desenvolvimento de aplicações para dispositivos iOS e para o sistema operativo Mac OSX. Esta linguagem foi desenvolvida pela empresa NeXT Software, fundada pelo visionário Steve Jobs.

A linguagem é uma extensão da linguagem C, que incorpora os conceitos de orientação aos objetos da linguagem Smalltalk (FONSECA et al., 2012)

Nesta fase foram de grande importância os tutoriais sobre desenvolvimento de

aplicações para o iOS 4(iPhone, iPad, iPad Touch), desenvolvidos pelos professores Catarina Reis, Catarina Silva, Luis Marcelino, Nuno Fonseca e Vitor Carreira, disponibilizados através da endereço online http://itunesu.ipleiria.pt/, para além do livro desenvolvido pelos mesmos autores (FONSECA et al., 2012).

Para testar aplicações no dispositivo físico e não apenas no simulador do ambiente XCode, é necessário possuir uma conta de desenvolvimento Apple. De referir que este passo é de extrema importância uma vez que a aplicação a desenvolver tem subjacente a necessidade da utilização da câmara fotográfica, que não é possível de utilizar no simulador. Uma vez que o IPL aderiu ao programa *Developer* da *Apple*, foi possível disponibilizar uma conta para realização dos testes no dispositivo físico.

Procedeu-se então à minha inscrição como *Apple Developer*, e a consequente obtenção do Certificado, fornecido pelo meu orientador de projeto, e registo do iPhone para que este pudesse ser utilizado como dispositivo para as aplicações a desenvolver.

Realizei diversos testes e programas de forma a familiarizar-me com o Objective-C, o Xcode e as possibilidades de desenvolvimento de aplicações.

3.2 Dispositivo utilizado

O equipamento utilizado para testar a aplicação foi um Apple iPhone 5. O ecrã tem a dimensão de 4,0', e uma resolução de 640 x 1136 pixels. A câmara fotográfica do dispositivo tem uma resolução de 8 megapixéis, com uma resolução de 3264x2448 pixels, focagem automática e Flash LED.

A aplicação está desenvolvida para a mais recente versão do sistema operativo da Apple para este equipamento, o iOS7.

As características da câmara fotográfica permitem recolher imagens com bastante qualidade, podendo no entanto, estas serem influenciadas pelas condições de luz e enquadramento do dispositivo em relação ao objeto que se quer fotografar.

3.3 Captação de imagem com a câmara

A câmara do dispositivo móvel é fundamental para a utilização da aplicação, uma vez que será o meio através do qual o utilizador irá obter a imagem dos suportes que contenham texto a ser reconhecido e sintetizado.

3.3.1 Classe AVCaptureSession

O objetivo da aplicação é o reconhecimento do texto escrito em suportes variados, normalmente papel, sinais, paredes ou outros. Como tal, a captação de imagem é o primeiro passo necessário para alcançar o objetivo final da aplicação.

Assim, iniciou-se o projeto com o desenvolvimento do sistema de captação de imagens através da câmara. Uma vez que os utilizadores alvo da aplicação são pessoas cegas, foi pensada uma forma de a captação das imagens ser feita em tempo real, sem que tenha que ser feita a captação de uma foto que de seguida teria que ser aberta para realizar o reconhecimento OCR.

O iOS fornece várias possibilidades para realizar a captura de imagens, mas uma vez que se pretende a captura de imagens em tempo real a opção escolhida é a implementação da Classe *AVCaptureSession* da *framework AVFoundation*.

A *framework AVFoundation* disponibiliza serviços essenciais para trabalhar e criar conteúdos multimédia baseados no tempo, nos sistemas iOS e OS X. Através do interface em Objective-C, é possível ver, capturar ou editar filmes audiovisuais tais como filmes *QuickTime* e arquivos MPEG-4 .

Um objeto *AVCaptureSession* representa uma sessão que coordena o fluxo de dados dos dispositivos de entrada AV para as saídas. Através da utilização deste objeto, é possível a captura de imagens em tempo real.

Para tal, adicionam-se os dispositivos de captura e as saídas para a sessão e inicia-se o fluxo de dados, enviando mensagens de *startRunning*, e para-se o fluxo com uma mensagem *stopRunning*. Utiliza-se a propriedade *sessionPreset* para configurar o nível de qualidade, o *bitrate* (taxa de transferência de bits) ou outras definições para a saída que se considerem importante para a captação das imagens.

3.4 Implementação de ferramentas OCR

Após obtida a imagem, o próximo passo é o reconhecimento óptico do texto existente na mesma, através da utilização de uma *framework* OCR.

O reconhecimento óptico de caracteres, normalmente conhecido como OCR, é a interpretação de imagens que tenham textos, e que depois de digitalizadas, o texto nelas contido pode ser reconhecido, editado e utilizado.

Verificou-se que existem algumas *frameworks* disponíveis para iOS, tendo sido três

dessas selecionadas para a realização de testes. Assim, foram testadas a *framework Open-Source* (código aberto) Tesseract e as *frameworks* comerciais Leadtools e Abbyy.

De referir que a utilização das *frameworks* comerciais requereu o contacto com as respetivas empresas e o preenchimento de formulários para obtenção de licenças de três meses para avaliação dos produtos.

3.5 Escolha da ferramenta OCR

A decisão da *framework* OCR a utilizar obedeceu à análise de fundamentalmente dois aspetos: o custo e a eficácia na leitura dos textos. Naturalmente a análise dos dois aspetos foi pensada de forma global e integrada, salvaguardando a possibilidade de não se optar pela solução mais acessível, caso o custo de uma das *frameworks* mais caras fosse considerado aceitável e caso a eficácia da leitura fosse muito superior.

Relativamente ao primeiro aspeto, o custo, verificou-se que uma das soluções é *Open-Source* e as outras exigem uma licença comercial. Assim, a *framework* Tesseract não apresenta custos de utilização e as restantes soluções apresentam custos, sendo estes no caso da solução Leadtools (pode ser consultado online em http://www.leadtools.com/resources/weborder.htm) de 995$ (aproximadamente 720,00 €) e no caso da *framework* ABBYY (pode ser consultado online em http://www.abbyy-developers.eu/en:business:pricing) de 4.900,00€.

Para comparar a eficácia do reconhecimento de cada uma das *frameworks*, foram recolhidas trinta fotos de textos, com formatos, tamanhos e *layouts* diferenciados. Procurou-se obter um conjunto de imagens que representassem uma variedade de situações o mais abrangentes possível, com textos em formatos, tamanhos e cores diferentes, alinhamentos diferentes para as imagens e diferentes exposições à luz no momento da captação das imagens, na tentativa de obter a mais fiel representação de situações reais que seguramente acontecerão aquando da utilização da aplicação.

De seguida, o texto constante em cada uma das fotos foi transcrito para ficheiros de texto individuais, no sentido de poder realizar a comparação entre o texto obtido após reconhecimento óptico da imagem e o texto original.

As fotos utilizadas nos diversos testes realizados podem ser visualizadas na secção anexos.

3.5.1 Distância de Levenshtein

Para obter valores de medição que possam ser utilizados na comparação entre os textos, o reconhecido opticamente e o original, foi desenvolvida uma função que reproduz o algoritmo de comparação de *strings Levenshtein distance*.

O algoritmo *Levenshtein distance* foi criado pelo cientista russo Vladimir Levenshtein, em 1965. O princípio básico do algoritmo é a medição da similaridade entre duas *strings*. Isto é feito através do cálculo do número de operações básicas, ou alterações necessárias para tornar as duas *strings* iguais. As alterações podem consistir numa substituição, apagamento ou inserção de um carácter. A distância de *Levenshtein d(p,t)* (p e t representam duas *strings*) entre duas *strings* é o menor número de operações necessárias para tornar p = t. Por exemplo, se p = "moose" e t = "moody" a distância *Levenshtein* entre p e t será *d(p,t)* = 2, uma vez que as *strings* podem tornar-se iguais utilizando duas substituições (troca da letra s pela letra d e troca da letra e pela letra y (MANGNES, 2005).

Tomando outro exemplo, a distância *Levenshtein* entre as palavras inglesas "kitten" (gato) e "sitting" (sentando-se) é 3, já que com apenas 3 edições conseguimos transformar uma palavra na outra, e não há maneira de o fazer com menos de três edições (SILVA, 2013)

1. kitten
2. sitten (substituição de 'k' por 's')
3. sittin (substituição de 'e' por 'i')
4. sitting (inserção de 'g' no final)

O algoritmo pode ser descrito nos seguintes passos.

- Determina-se n como tamanho de p e m como sendo o tamanho de t. Se ambas forem zero, o processo termina;
- Cria-se uma matriz com m linhas e n colunas, e inicializa-se a primeira linha e a primeira coluna com 0...m e 0..n respetivamente;
- Examina-se cada carácter de p e t de 1 até n e de 1 até m;
- Se p[i] = t[j], os caracteres são iguais e a transformação custa 0. Se p[i] != de t[i] os caracteres não são iguais, e a transformação custa 1;
- O valor da célula d[i,j] é definido para o mínimo de d[i-1, j] + 1 (a célula acima), d[i, j-1] + 1 (a célula da esquerda +1), ou d[i-1, j-1] + custo (a célula diagonalmente acima e à esquerda);
- Os passos 3 a 5 são repetidos até que a pontuação da distância seja encontrada na

célula d[n,m] (MANGNES, 2005).

A distância zero indica que as *strings* são idênticas. Naturalmente, quanto maior for a distância, menor será a similaridade entre as *strings*.

Na secção anexos apresenta-se um exemplo completo da utilização do algoritmo *Levenshtein distance* para determinar a distância entre duas *strings*.

O valor obtido através da implementação do algoritmo *Levenshtein distance* forneceu dados importantes no sentido de definir uma métrica de medição para verificar a eficiência de cada um dos OCRs.

Sendo obtido o valor numérico da distância que separa o texto original do texto obtido pelo reconhecimento OCR, torna-se possível obter um valor percentual para a diferença entre o original e o reconhecido, havendo assim um valor numérico passível de ser comparado e estatisticamente analisado. O número total de caracteres existentes em cada texto é conhecido, e foi obtido através do objeto *lenght*, da classe *NSString*.

Desenvolveram-se três aplicações (uma para cada *framework*) que procedem ao reconhecimento dos textos das imagens e à leitura do texto original que constava em ficheiros de texto, e recorrendo a uma função desenvolvida tendo por base o algoritmo de *Levenshtein* apresentava as respetivas distâncias.

Apresenta-se de seguida a implementação do algoritmo *Levenshtein distance* em Objective-C*:*

```
// Return the minimum of a, b and c - used by compareString:withString:
-(NSInteger)smallestOf:(NSInteger)a andOf:(NSInteger)b andOf:(NSInteger)c
{

    NSInteger min = a;
    if ( b < min )
        min = b;

    if( c < min )
        min = c;

    return min;


}
```

```
//Return the smallest of a or b
-(NSInteger)smallestOf:(NSInteger)a andOf:(NSInteger)b
{

    NSInteger min=a;

    if (b < min)
        min=b;


    return min;

}
```

```objc
//String distance
-(NSInteger) calculaDistancia:(NSString*)stringOriginal calculaDistancia:(NSString*)stringLida
{
    // Normalize strings
    NSString *originalString = stringOriginal;
    NSString *comparisonString = stringLida;
    [originalString stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceAndNewlineCharacterSet]];
    [comparisonString stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceAndNewlineCharacterSet]];
    originalString = [originalString lowercaseString];
    comparisonString = [comparisonString lowercaseString];

    // Step 1
    NSInteger k, i, j, cost, * d, distance;
    NSInteger n = [originalString length];
    NSInteger m = [comparisonString length];

    if( n++ != 0 && m++ != 0 ) {
        d = malloc( sizeof(NSInteger) * m * n );
        // Step 2
        for( k = 0; k < n; k++)
            d[k] = k;
        for( k = 0; k < m; k++)
            d[ k * n ] = k;
        // Step 3 and 4
        for( i = 1; i < n; i++ )
            for( j = 1; j < m; j++ ) {
                // Step 5
                if( [originalString characterAtIndex: i-1] ==
                   [comparisonString characterAtIndex: j-1] )
                    cost = 0;
                else
                    cost = 1;
                // Step 6
                d[ j * n + i ] = [self smallestOf: d [ (j - 1) * n + i ] + 1
                                            andOf: d[ j * n + i - 1 ] +  1
                                            andOf: d[ (j - 1) * n + i - 1 ] + cost ];

                // This conditional adds Damerau transposition to Levenshtein distance
                if( i>1 && j>1 && [originalString characterAtIndex: i-1] ==
                   [comparisonString characterAtIndex: j-2] &&
                   [originalString characterAtIndex: i-2] ==
                   [comparisonString characterAtIndex: j-1] )
                {
                    d[ j * n + i] = [self smallestOf: d[ j * n + i ]
                                               andOf: d[ (j - 2) * n + i - 2 ] + cost ];
                }
            }
        distance = d[ n * m - 1 ];

        free( d );
        //return the string distance
        return distance;
    }
    return 0;
}
```

Figura 22 – Implementação do Algoritmo *Levenshtein Distance* em Objective-C

3.5.2 Análise de resultados

Os valores obtidos foram registados numa folha de cálculo para que fosse possível analisá-los de forma estatística. Assim, registaram-se os nomes das imagens, o total de caracteres dos textos nos textos originais dessas mesmas imagens, a distância obtida entre a leitura do texto original e o texto reconhecido e a percentagem de erros, calculada na divisão entre a distância obtida e o total de carateres do texto. Foi então calculada valor da mediana dos valores das distâncias obtidas.

A mediana é definida como o valor que ocupa a posição central num conjunto de dados ordenados. Logo, tem a propriedade de dividir um conjunto de observações em duas partes iguais quanto ao número dos elementos: o número de dados que são menores ou iguais à mediana é o mesmo que o número de dados que são maiores ou iguais à mediana. Desta forma, pode afirmar-se que 50% das observações que compõem um conjunto qualquer de dados estatísticos são menores ou iguais ao valor correspondente à sua mediana, e, por sua vez os 50% restantes, são observações maiores ou iguais a esta medida. Ao contrário da média, a mediana não é influenciada por valores extremos, visto que é uma medida essencialmente vinculada à posição que ocupa no conjunto ordenado. Assim, se algum valor for demasiado grande ou pequeno – valores extremos –, estes não afectarão o cálculo da mediana, já que não alterarão a ordem.

Por exemplo, se analisarmos dois conjuntos de dados A e B, sendo A = { 1, 2, 1000 } e B = { 1, 2, 10 }, em ambos, a mediana é 2, ou seja, ao se trocar o valor 10 (conjunto B) por 1000 (conjunto A), o valor da mediana não se altera. Por isso, quando se trabalha com observações que apresentam valores extremos, como é o caso dos valores da distância obtidos nas leituras OCR, a opção recaiu na utilização da mediana ao invés da média, pois assim é possível obter uma melhor representação dos dados que apresentam esse diferencial.

As características anteriormente enunciadas definiram a escolha da mediana como valor de referencia para comparar os resultados.

Os resultados obtidos apresentam-se de seguida, através da sua representação em tabelas e a respetiva representação gráfica.

Resultados obtidos após testes realizados com a *framework* Tesseract.

Imagem	N.º total de caracteres	Distância	Percentagem da distância
IMG 0	1032	**289**	28,00%
IMG 1	591	**399**	67,51%
IMG 2	495	**234**	47,27%
IMG 3	242	**118**	48,76%
IMG 4	308	**180**	58,44%
IMG 5	258	**124**	48,06%
IMG 6	331	**78**	23,56%
IMG 7	808	**91**	11,26%
IMG 8	761	**594**	78,06%
IMG 9	599	**399**	66,61%
IMG 10	136	**81**	59,56%
IMG 11	137	**46**	33,58%
IMG 12	446	**2533**	567,94%
IMG 13	767	**417**	54,37%
IMG 14	588	**45**	7,65%
IMG 15	604	**60**	9,93%

Mediana da percentagem Distancia	
Valor da Mediana	**23,45%**

IMG 16	603	**108**	17,91%
IMG 17	419	**210**	50,12%
IMG 18	653	**275**	42,11%
IMG 19	1041	**67**	6,44%
IMG 20	848	**69**	8,14%
IMG 21	1074	**133**	12,38%
IMG 22	947	**126**	13,31%
IMG 23	688	**114**	16,57%
IMG 24	956	**223**	23,33%
IMG 25	737	**44**	5,97%
IMG 26	981	**71**	7,24%
IMG 27	623	**53**	8,51%
IMG 28	1855	**288**	15,53%
IMG 29	738	**62**	8,40%

Tabela 1 – Resultados obtidos após aplicação do algoritmo de *Levenshtein* , utilizando o OCR Tesseract

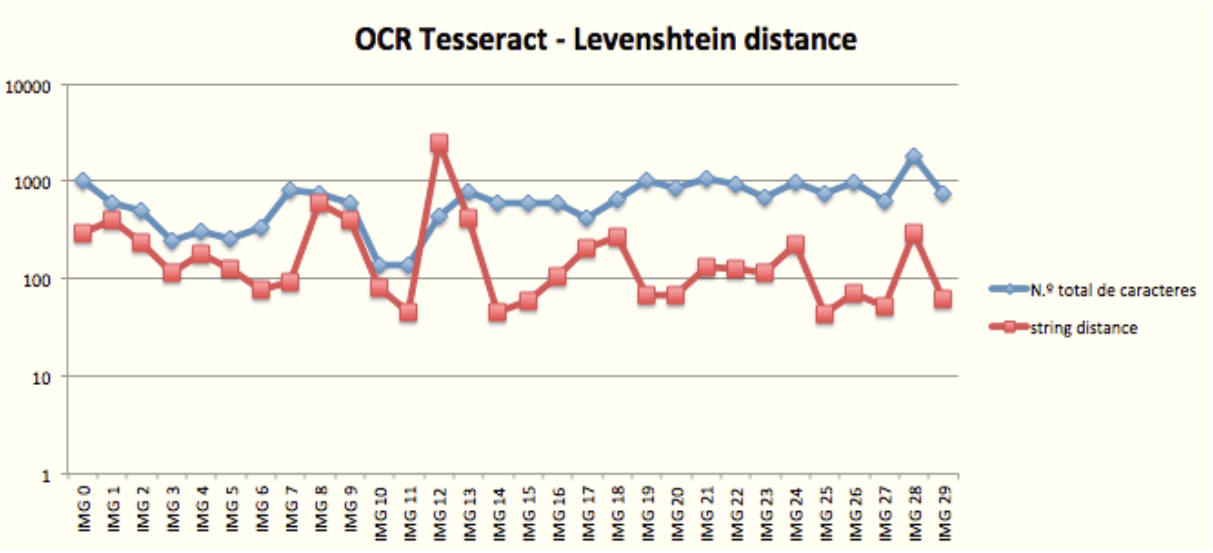


Figura 23 – Representação gráfica dos resultados para o OCR Tesseract

Resultados obtidos após testes realizados com a *framework* ABBYY.

Imagem	N.º total de caracteres	Distância	Percentagem da distância
IMG 0	1032	**461**	44,67%
IMG 1	591	**105**	17,77%
IMG 2	495	**192**	38,79%
IMG 3	242	**96**	39,67%
IMG 4	308	**227**	73,70%
IMG 5	258	**113**	43,80%
IMG 6	331	**110**	33,23%
IMG 7	808	**108**	13,37%
IMG 8	761	**157**	20,63%
IMG 9	599	**303**	50,58%
IMG 10	136	**57**	41,91%
IMG 11	137	**57**	41,61%
IMG 12	446	**154**	34,53%
IMG 13	767	**152**	19,82%

Mediana da percentagem Distancia	
Valor da Mediana	**18,81%**

IMG 14	588	**137**	23,30%
IMG 15	604	**101**	16,72%
IMG 16	603	**117**	19,40%
IMG 17	419	**60**	14,32%
IMG 18	653	**119**	18,22%
IMG 19	1041	**138**	13,26%
IMG 20	848	**120**	14,15%
IMG 21	1074	**161**	14,99%
IMG 22	947	**109**	11,51%
IMG 23	688	**156**	22,67%
IMG 24	956	**168**	17,57%
IMG 25	737	**76**	10,31%
IMG 26	981	**145**	14,78%
IMG 27	623	**104**	16,69%
IMG 28	1855	**203**	10,94%
IMG 29	738	**102**	13,82%

Tabela 2 - Resultados obtidos após aplicação do algoritmo de *Levenshtein* , utilizando o OCR Abbyy

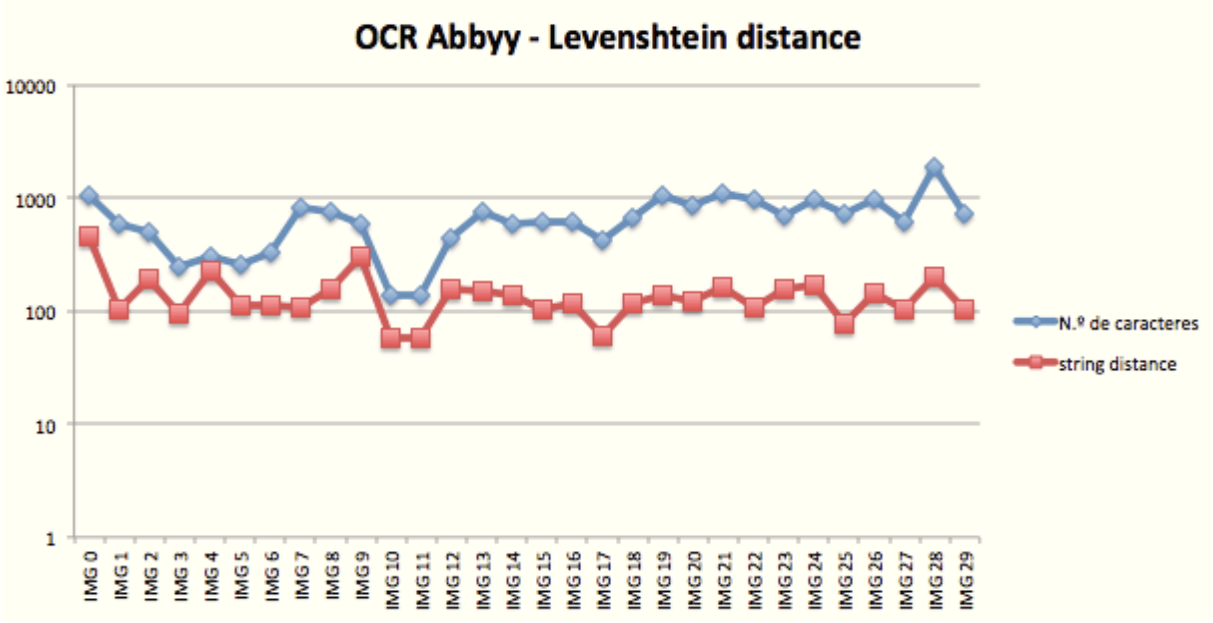


Figura 24 - Representação gráfica dos resultados para o OCR Abbyy

Resultados obtidos após testes realizados com a *framework* Leadtools.

Imagem	N.º total de caracteres	Distância	Percentagem da distância
IMG 0	1032	**284**	27,52%
IMG 1	591	**322**	54,48%
IMG 2	495	**208**	42,02%
IMG 3	242	**97**	40,08%
IMG 4	308	**165**	53,57%
IMG 5	258	**157**	60,85%
IMG 6	331	**82**	24,77%
IMG 7	808	**81**	10,02%
IMG 8	761	**1131**	148,62%
IMG 9	599	**391**	65,28%

Mediana da percentagem Distancia	
Valor da Mediana	**18,76%**

IMG 10	136	**52**	38,24%
IMG 11	137	**501**	365,69%
IMG 12	446	**4862**	1090,13%
IMG 13	767	**160**	20,86%
IMG 14	588	**98**	16,67%
IMG 15	604	**97**	16,06%
IMG 16	603	**82**	13,60%
IMG 17	419	**156**	37,23%
IMG 18	653	**300**	45,94%
IMG 19	1041	**68**	6,53%
IMG 20	848	**63**	7,43%
IMG 21	1074	**86**	8,01%
IMG 22	947	**51**	5,39%
IMG 23	688	**111**	16,13%
IMG 24	956	**144**	15,06%
IMG 25	737	**25**	3,39%
IMG 26	981	**70**	7,14%
IMG 27	623	**58**	9,31%
IMG 28	1855	**238**	12,83%
IMG 29	738	**50**	6,78%

Tabela 3 - Resultados obtidos após aplicação do algoritmo de *Levenshtein* , utilizando o OCR Leadtools

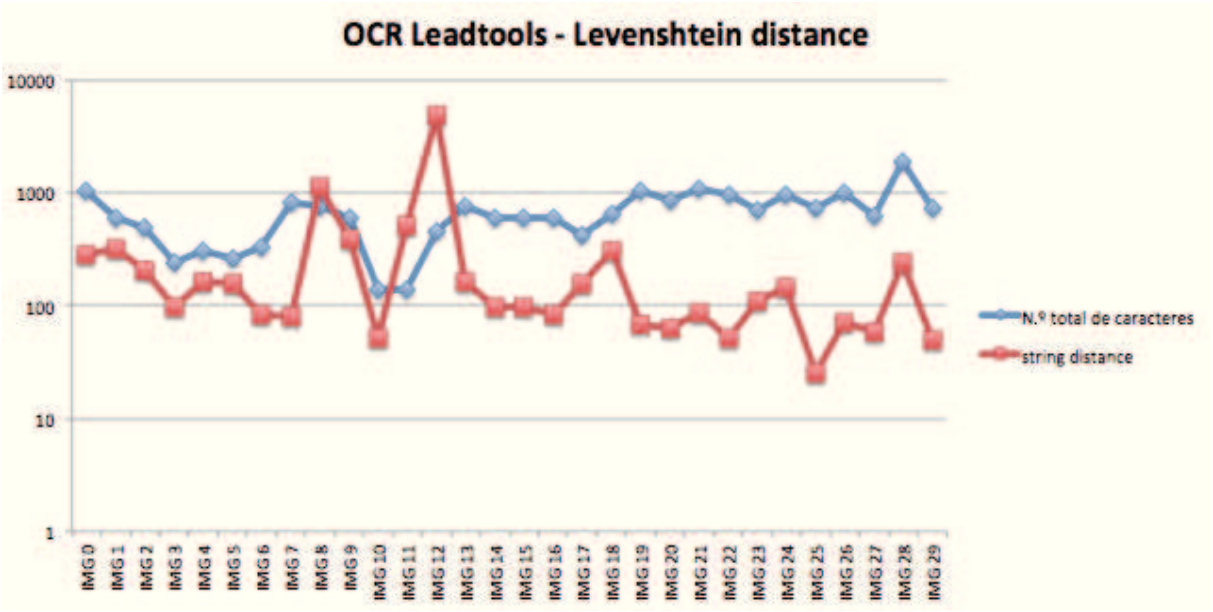


Figura 25 - Representação gráfica dos resultados para o OCR Leadtools

A análise comparativa das distâncias de *Levenshtein* obtidas através da utilização das três *frameworks* testadas encontra-se representada graficamente na próxima página:

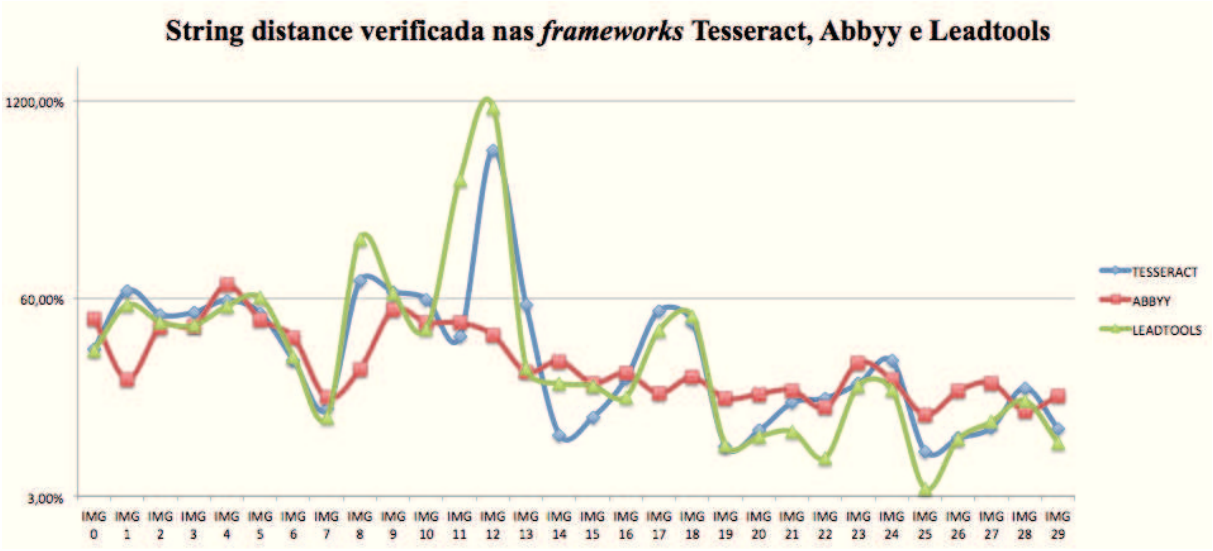


Figura 26 - Valores da string distance para as frameworks Tesseract, Abbyy e Leadtools

Apresenta-se de seguida o comparativo do resultado da mediana das distâncias verificadas em cada uma das *frameworks* analisadas.

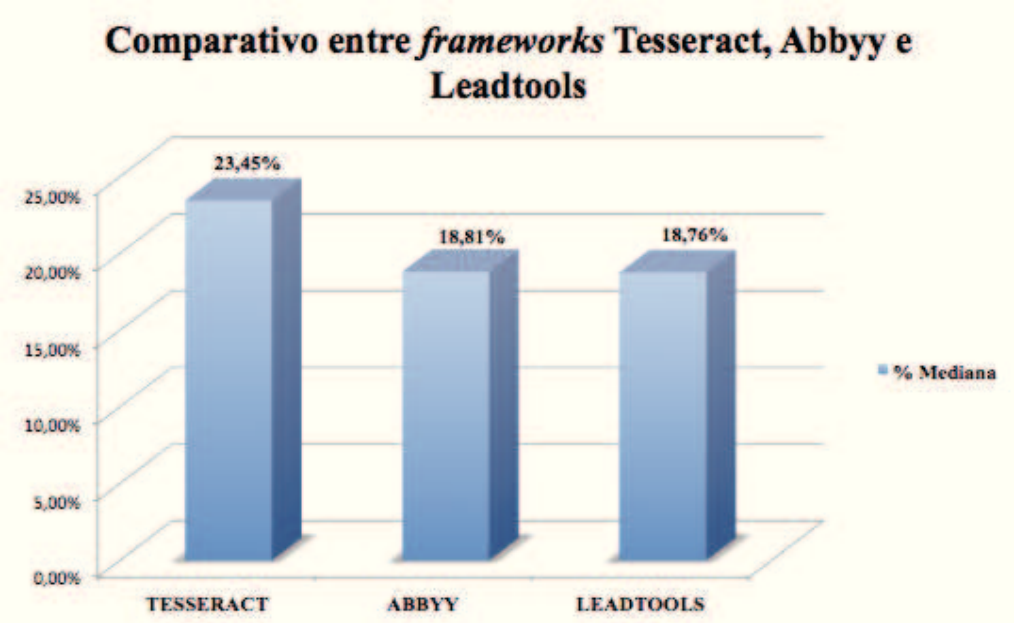


Figura 27 - Comparativo do valor da mediana entre as frameworks Tesseract, Abbyy e Leadtools

Como pode ser verificado pela análise dos resultados, as *frameworks* comerciais apresentam melhores resultados que a *framework* gratuita (OCR Tesseract – 23,45%, OCR ABBYY – 18,81% e OCR Leadtools 18,76%).

Não obstante esta diferença de resultados verificados entre as *frameworks*, decidiu-se que a *framework* a utilizar no projeto seria a Tesseract, uma vez que não apresenta custos, e possibilita o desenrolar do projeto por um período temporal mais alargado, uma vez que as

licenças de demonstração disponibilizados pelas *frameworks* comerciais possibilitariam apenas o desenvolver do projeto durante três meses.

3.6 Implementação da ferramenta Text-To-Speech

Concluído o processo de captação das imagens e da obtenção do texto constante nessas imagens através do reconhecimento OCR, o próximo passo é a transformação do texto reconhecido em linguagem oral.

Para essa transformação utiliza-se uma *framework* Text-to-speech (TTS). A *framework* TTS realiza uma síntese de voz utilizada para criar um som falado que reproduz o texto obtido de um documento, ou no caso deste projeto, do texto obtido através do reconhecimento óptico de uma imagem.

Existem disponíveis para utilização diversas *frameworks*, algumas gratuitas como são os casos da GoogleTTS e da OpenEars ou outras comerciais como são os casos da iSpeech ou da AcapelaTTS.

Foram realizados testes com as *frameworks* gratuitas, verificando-se resultados muito similares na sintetização da voz humana com as *strings* fornecidas, sendo no entanto a *framework* GoogleTTS mais fácil de utilizar e implementar.

Não obstante os resultados obtidos com as *frameworks* TTS serem considerados bastante positivos, verificando-se que as sintetizações dos textos eram satisfatórias, uma vez que permitiam uma compreensão fácil das leituras realizadas, verificou-se que após a atualização do sistema operativo iOS para a versão 7, foi incluída na *framework AVFoundation* uma nova classe, a classe *AVSpeechSynthesizer*, e após realizados testes com esta nova classe, que revelaram-se igualmente bastante satisfatórios, a opção a ser utilizada na aplicação será esta classe, uma vez que permite obter a sintetização de voz pretendida para a aplicação, sem necessidade de recorrer a *frameworks* externas.

A classe *AVSpeechSynthesizer* permite produzir discursos sintetizados a partir de textos, num dispositivo iOS, e fornece métodos para controlar e monitorizar o progresso do discurso em curso.

Para utilizar esta classe é necessário criar uma instância da classe *AVSpeechUtterance* com algum texto. Depois, essa instância é passada para o método *speakUtterance* que sintetiza o texto em voz humana.

A classe *AVSpeechUtterance* representa a unidade básica de síntese de voz. É feito o encapsulamento de uma certa quantidade de texto a ser falado e um conjunto de parâmetros

que afetam o seu discurso: a voz, afinação, ritmo, e atraso.

Apresenta-se de seguida a função que, recorrendo à classe *AVSpeechSynthesizer* sintetiza o texto fornecido:

```
//Speech Systhesizer
//iOS7 new feature
//used for speech text recognized
-(void) speakText:(NSString*) text
{

    AVSpeechSynthesizer *synthesizer = [[AVSpeechSynthesizer alloc] init];
    AVSpeechUtterance *utterance = [AVSpeechUtterance speechUtteranceWithString:text];
    utterance.voice = [AVSpeechSynthesisVoice voiceWithLanguage:@"en-US"];
    [utterance setRate:0.2f];

    //speak
    [synthesizer speakUtterance:utterance];

}
```

Figura 28 – Implementação da classe *AVSpeechSynthesizer*

3.7 Melhorias de usabilidade da aplicação

Para que a aplicação tenha uma melhor apresentação no dispositivo móvel, foi necessário proceder ao desenvolvimento de um ícone que pudesse identificar a aplicação no dispositivo assim como a imagem a apresentar no ecrã assim que a aplicação esteja em execução.

Figura 29 - Ícone da aplicação

Figura 30 - Ecrã inicial da aplicação

Foi necessário pensar no nome que pudesse ser apresentado no dispositivo, uma vez que se esse nome ultrapasse os 11 caracteres não seria totalmente apresentado. Uma vez que o nome do projeto ultrapassa largamente o número de carateres referido, foi pensado o nome

Read4Blind, que totaliza 10 caracteres e representa um nome intuitivo e permite identificar a aplicação sem que o nome fique incompleto no ecrã.

Procedeu-se então à alteração do campo *Bundle display name,* existente no separador *Info* da aplicação, para o nome pretendido. A imagem seguinte exemplifica a forma como o ícone e o nome da aplicação são visualizados no ecrã.

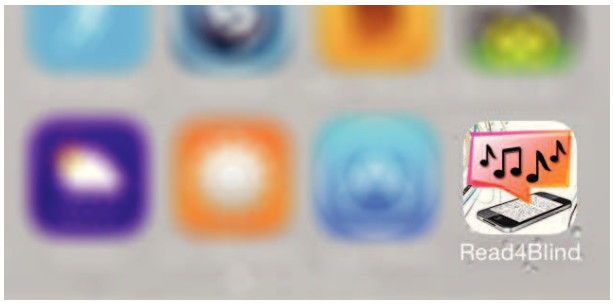


Figura 31 - Aspeto do ícone da aplicação no ecrã de um iPhone

De forma a proporcionar uma experiência de utilização mais agradável aos utilizadores cegos, foram introduzidas instruções de voz nos diversos ecrãs da aplicação e um alerta vibratório após o processo de reconhecimento, para os guiar na utilização da mesma. Uma vez que a aplicação foi desenvolvida para a língua inglesa, todas as instruções são fornecidas em inglês.

Assim, ao iniciar a aplicação é apresentada uma mensagem de boas vindas com instruções de como começar a utilizar a aplicação, aceder ao ecrã de instruções ou aceder ao ecrã de menu. Para iniciar a utilização da aplicação basta ao utilizador realizar um dos gestos de arrastar o ecrã para a direita ou para a esquerda (*swipe right or swipe left*), para aceder ao ecrã das instruções basta realizar o gesto de arrastar ao longo do ecrã para cima (*swipe up*) e para aceder ao menu que possibilita realizar alterações de opções da aplicação, basta realizar o gesto de arrastar ao longo do ecrã par abaixo (*swipe down*).

Assim que o utilizador navega até ao ecrã de instruções, escutará as instruções necessárias para utilizar a aplicação. É indicado ao utilizador que deverá apontar o dispositivo móvel para o texto, clicar no ecrã e aguardar pela vibração do dispositivo até o reconhecimento estar concluído, sendo depois realizada a leitura do texto. São também fornecidas instruções relativas às operações que o utilizador pode executar após o reconhecimento, sendo estas a repetição da escuta do texto reconhecido, o envio do texto reconhecido por email ou o regresso ao ecrã de captação de imagem, através do qual poderá repetir o processo de captação e reconhecimento. O utilizador poderá repetir a escuta das instruções, realizando o gesto *swipe up*, ou voltar ao ecrã inicial através dos gestos *swipe right* ou *swipe left*.

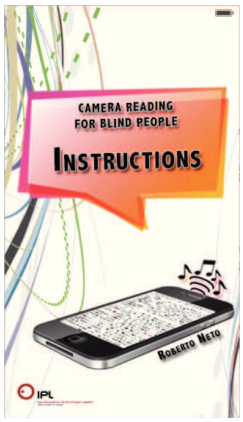


Figura 32 – Ecrã de instruções da aplicação

Se o utilizador aceder ao ecrã menu, escutará as instruções indicativas de que poderá visualizar, e se pretender, alterar o valor da percentagem para prevenção de reconhecimento inválido, que está definido por defeito para o valor de 5% (ver ponto 4.3 – Prevenção de erro no resultado final). O utilizador pode alterar o valor inicial, ajustando-o como pretender, aumentando ou diminuindo esse valor em intervalos de 0,5%. O valor alterado fica guardado no sistema através da classe *NSUserDefaults* da classe *Cocoa Foundation*. A classe *NSUserDefaults* é uma classe que permite que uma aplicação facilmente defina e aceda ao que normalmente se designa como as suas próprias preferências. As definições são guardadas e ficam disponíveis em todo o código da aplicação, persistindo entre chamadas à aplicação (FONSECA et al., 2012).

O ecrã é constituído por uma *label* que apresenta o valor atual mas também os valores que forem sendo alterados, um *stepper* que fornece ao utilizador um interface para incrementar ou decrementar o valor e um botão, que permite ao utilizador guardar a eventual alteração realizada ao valor.

Após proceder à alteração do valor da percentagem, o utilizador pode clicar no botão *Save* e voltar ao ecrã inicial, ou então, caso opte por não realizar qualquer alteração pode voltar ao ecrã inicial através do gesto *swipe right* ou *swipe left*.

Figura 33 – Ecrã de menu da aplicação

Estando no ecrã inicial, o utilizador poderá aceder ao ecrã de captação de imagem realizando o gesto *swipe right* ou *swipe left*.

Uma vez no ecrã de captação de imagem, o utilizador poderá a qualquer momento voltar ao ecrã inicial da aplicação, bastando para tal clicar na parte de cima do seu ecrã, num botão inserido para o efeito, criando-se assim um sistema simples e intuitivo de navegação entre os ecrãs da aplicação. Ainda no ecrã de captação de imagem o utilizador pode clicar uma vez no ecrã, no botão central, representado por uma mira, e proceder à captura da imagem com texto a ser reconhecido.

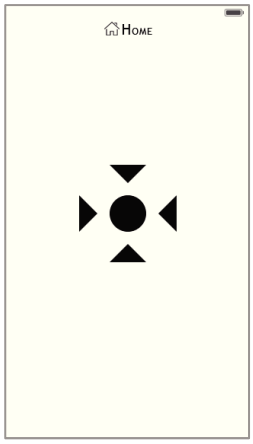


Figura 34 - Ecrã de captação de imagem da aplicação

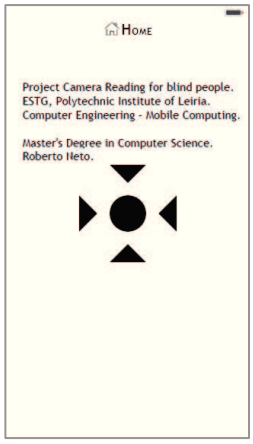


Figura 35 - Ecrã de captação de imagem, com texto a ser reconhecido

Assim que o utilizador clica no botão central, é captada a fotografia da imagem que é apresentada nesse momento no ecrã, são aplicados os filtros (ver ponto 4.1 – Melhoria dos resultados OCR) e inicia-se o processo de reconhecimento óptico dos caracteres existentes na imagem. Enquanto decorre o processo de reconhecimento óptico, o botão central desaparece, para que o utilizador não volte a iniciar o processo sem que obtenha o resultado anterior.

Finalizado o processo de reconhecimento o equipamento vibra e ocorre uma de duas possibilidades: caso se verifique um reconhecimento válido, é realizada a sintetização de voz do texto reconhecido e o utilizador escuta a leitura do mesmo; caso se verifique um reconhecimento inválido (ver ponto 4.3 – Prevenção de erro no resultado final), o utilizador recebe uma mensagem indicativa do erro no processamento, sendo então solicitado a repetição do processo de captação.

Após realizado o processo de reconhecimento com sucesso e a respetiva sintetização, será apresentado ao utilizador três botões, que lhe possibilitam voltar a ouvir o texto reconhecido, enviar o texto reconhecido por email, ou voltar ao ecrã de captação de imagem para iniciar novo processo de reconhecimento.

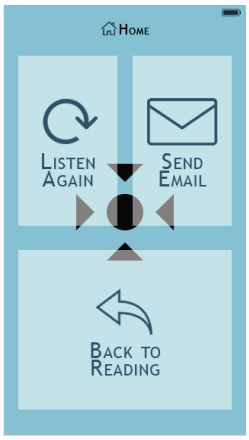


Figura 36 - Ecrã com três botões com opções após reconhecimento

Caso o utilizador pretenda enviar o texto por email, ser-lhe-á apresentado ecrã de email, com indicações das ações que pode tomar.

Figura 37 - Ecrã de email da aplicação

O utilizador terá que preencher os campos de destinatário (*To*) e assunto (*Subject*) do email, e clicar em enviar (*Send email*). Depois do utilizador clicar em enviar, será aberta a aplicação de email do dispositivo, com os campos destinatário e assunto preenchidos com os dados introduzidos na janela anterior, e no corpo do texto, o texto reconhecido pelo OCR.

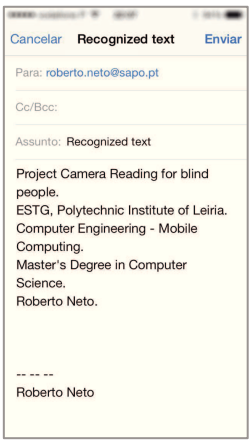


Figura 38 - Janela da aplicação de email com o texto reconhecido

Após enviado o email, é novamente apresentada o ecrã de email, e o utilizador poderá enviar novo email, ou através dos gestos de arrastar o ecrã para a direita ou para a esquerda (*swipe right or swipe left*) voltar ao ecrã de captação de imagem, sendo estas últimas opções fornecidas ao utilizador através de instruções de voz.

O esquema de navegação entre ecrãs da aplicação, encontra-se esquematizado na figura seguinte:

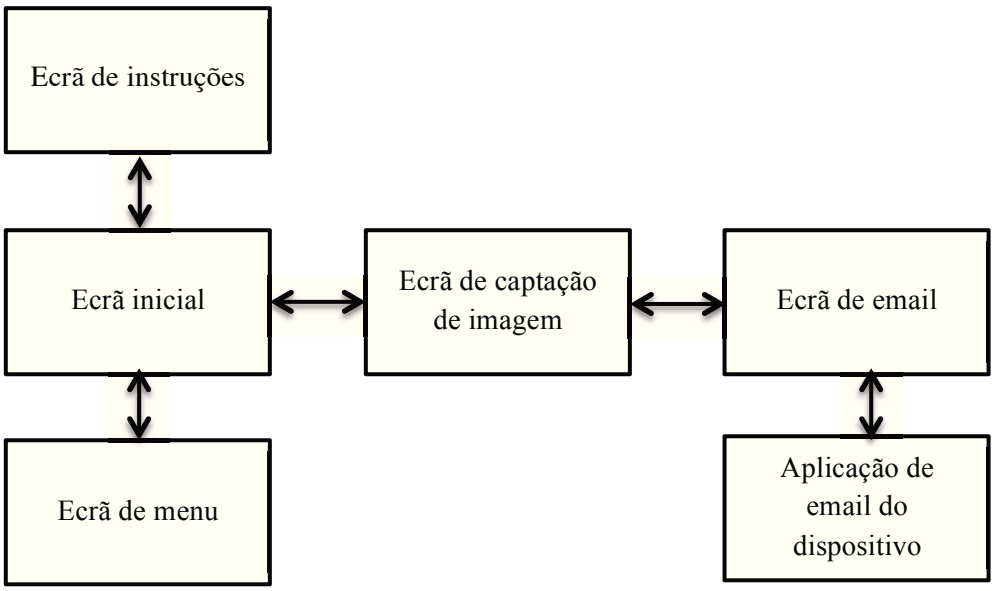


Figura 39 – Esquema de navegação entre ecrãs da aplicação

3.7 Conclusão

O projeto desenvolveu-se segundo uma metodologia iniciada com a introdução ao sistema operativo utilizado pelos dispositivos Apple para os quais está a ser desenvolvida a aplicação e apresentando-se de seguida o dispositivo utilizado, fundamentais para o conhecimento do ambiente de desenvolvimento, da linguagem de programação a utilizar e do equipamento que serve de suporte à aplicação. Seguiram-se os passos necessários ao desenvolvimento da aplicação propriamente dita, tais como a forma como se procede à captação das imagens com a câmara do dispositivo, a implementação de ferramentas OCR, que obedeceu a uma criteriosa escolha e a implementação da ferramenta TTS, fundamental para a sintetização dos textos reconhecidos.

Por fim foram elencadas as diversas melhorias de usabilidade da aplicação, que se revelam de grande importância para que a aplicação seja mais fácil de utilizar e configurar por parte dos utilizadores invisuais, que são os destinatários finais da aplicação.

4 – Optimização do Sistema

A partir do momento em que se encontram combinadas as *frameworks* OCR e o TTS numa aplicação que possibilita o reconhecimento de textos e a sua sintetização em voz, verificou-se ser necessária a procura de optimizações para o sistema tentando obter ainda melhores resultados.

4.1 Melhoria dos resultados OCR

Após escolhido a *framework* OCR Tesseract, procedeu-se a um conjunto de testes no sentido de procurar obter melhores resultados no reconhecimento das imagens.

Tendo em conta que as características da imagem utilizada no processo de reconhecimento óptico tem impacto no resultado, nomeadamente a qualidade da imagem, a luminosidade, o contraste de cores, a focagem entre outras características, procedeu-se à tentativa de realizar alterações às imagens captadas pelo dispositivo, no sentido de melhorar o resultado obtido na sua leitura OCR.

Assim, procedeu-se à utilização de filtros no sentido de procurar realizar um pré-processamento das imagens que resulte na obtenção de melhores resultados.

Utilizando as imagens recolhidas para testes, e aplicando o algoritmo de *Levenshtein* para determinar a *string distance* obtendo assim um valor passível de medição, procederam-se a diversos ajustes, sendo aplicados vários filtros às imagens na busca de um resultado mais satisfatório.

Foram utilizados filtros disponibilizados pela *framework Core Image*. Esta framework pode ser utilizada no tratamento de imagem, nomeadamente para:

- Processar imagens utilizando filtros de imagem existentes;
- Criar cadeias de filtros e arquivá-los para posterior utilização;
- Detectar características (tais como, rostos e olhos) em imagens e vídeo, e rastrear

os rostos em imagens de vídeo;

- Analisar as imagens para obter um conjunto de filtros de ajuste automático.

Quando se manipula imagens, há três classes principais da *framework Core Image* que são utilizadas (FONSECA et al., 2012):

- *CIFilter* – é um objecto mutável que representa um efeito que se pretende aplicar, com vários parâmetros de entrada (imagem ou valores numéricos) e que produz uma imagem de saída em função do estado do filtro;
- *CIImage* – é um objecto mutável que representa uma imagem à qual pode ser aplicado um filtro ou cadeia de filtros, ou que resulta deste filtro ou cadeia de filtros
- *CIContext* – é o objeto através do qual o *Core Image* desenha o resultado, baseado no GPU ou CPU.

Desde logo foram excluídos alguns filtros, que manifestamente nada acrescentariam à melhoria pretendida para os resultados, uma vez que o efeito aplicado às imagens é de distorção ou de alterações sem utilidade prática para este projeto tais como *CIBumpDistortion* (cria uma saliência num ponto especificado na imagem), *CIConstantColorGenerator* (gera uma cor sólida), *CIGlassDistortion* (distorce uma imagem aplicando uma textura tipo vidro) ou *CIKaleidoscope* (produz um caleidoscópio através de uma imagem de origem) entre outros.

Procurou-se então encontrar filtros que pudessem introduzir melhorias nos resultados do reconhecimento OCR, através da alteração de características das imagens que tornasse o texto nela contidos mais facilmente reconhecíveis, como o brilho, a cor, o contraste, a saturação ou o ajuste da exposição tal como numa máquina fotográfica.

Assim, consideraram-se para a realização de testes os filtros *CIToneCurve* (ajusta a resposta dos canais RGB), *CIHueAdjust* (altera a cor da generalidade dos pixéis) *CIGammaAdjust* (ajusta o brilho intermédio), *CIColorControls* (ajusta valores de saturação, brilho e contraste), *CIColorMonochrome* (remapeia as cores de uma imagem para que essas cores fiquem dentro da tonalidade de apenas uma cor), *CIExposureAdjust* (ajusta a exposição da imagem de forma idêntica ao de uma máquina fotográfica).

O filtro *CIColorControls* permite ajustar a saturação, o brilho e o contraste numa imagem. Recebe como parâmetro uma imagem do tipo *CIImage*, e podem ser alterados os valores para os atributos atrás referidos.

Apresenta-se de seguida, uma imagem ilustrativa do resultado da aplicação do filtro *CIColorControls:*

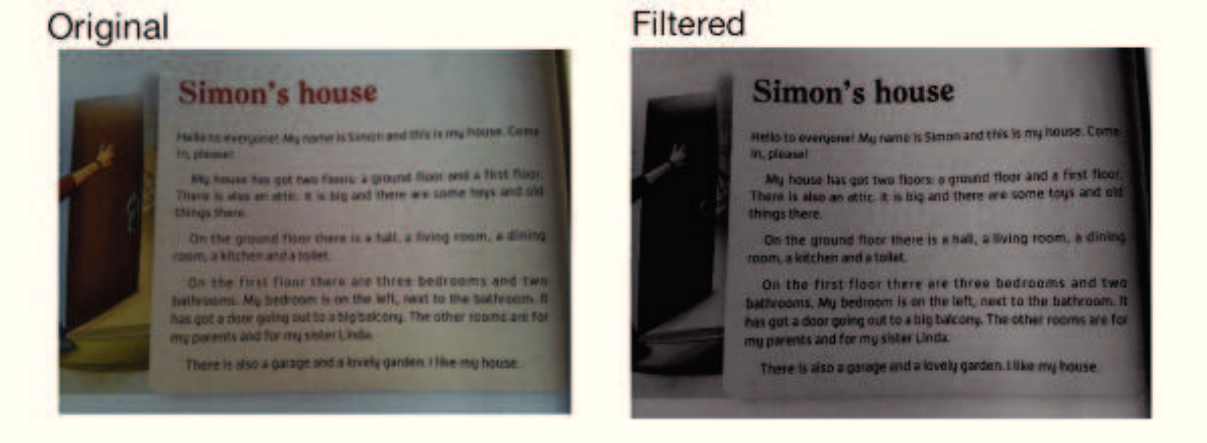


Figura 40 - Exemplo do resultado obtido após aplicação do filtro *CIColorControls*

O filtro *CIGammaAdjust* permite ajustar o brilho intermédio de uma imagem. Recebe como parâmetros uma imagem do tipo *CIImage* e pode ser alterado o valor do *inputPower*, que corresponde ao valor gama da imagem.

Apresenta-se de seguida uma imagem representativa do resultado da aplicação do filtro *CIGammaAdjust*:

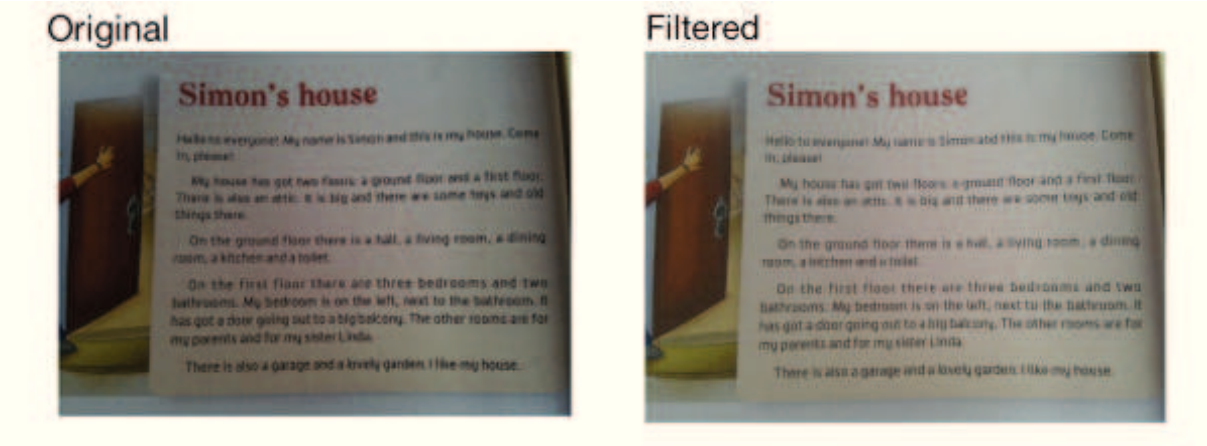


Figura 41 - Exemplo do resultado obtido após aplicação do filtro *CIGammaAdjust*

O filtro *CIExposureAdjust* permite ajustar a definição de uma imagem de forma semelhante à realizada com o ajuste F/Stop numa máquina fotográfica, ou seja, permite ajustar a quantidade de luz necessária para a captação da imagem. Recebe como parâmetro uma imagem do tipo *CIImage* e pode ser alterado o valor da exposição da luz.

Apresenta-se de seguida, uma imagem ilustrativa do resultado da aplicação do filtro *CIExposureAdjust:*

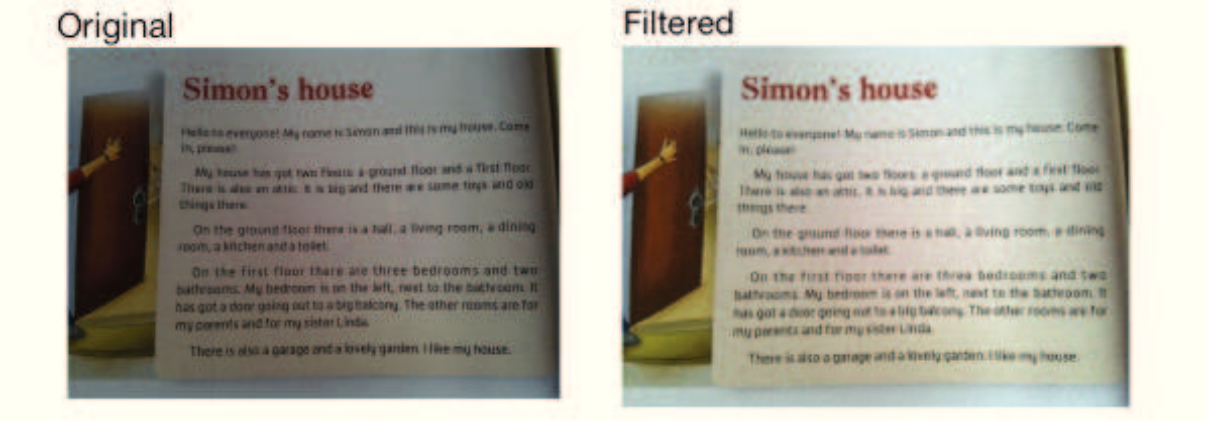


Figura 42 - Exemplo do resultado após aplicação do filtro *CIExposureAdjust*

O filtro *CIColorMonochrome* permite remapear as cores de uma imagem para que essas cores fiquem dentro da tonalidade de apenas uma cor. Recebe como parâmetros uma imagem do tipo *CIImage*, e podem ser alterados os valores para a cor de entrada e para a intensidade.

Apresenta-se de seguida, uma imagem ilustrativa do resultado da aplicação do filtro *CIColorMonochrome.*

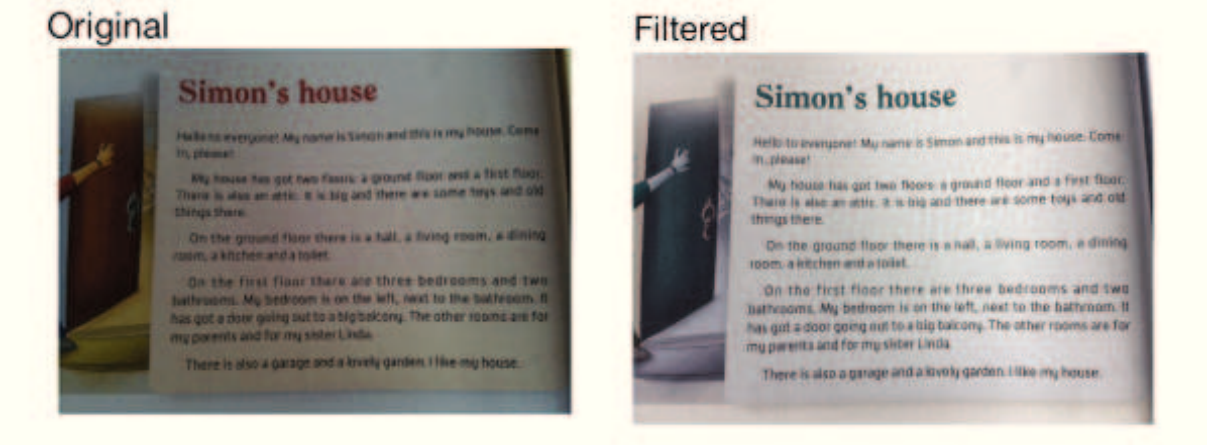


Figura 43 – Exemplo do resultado obtido após aplicação do filtro *CIColorMonochrome*

O filtro *CIToneCurve* permite ajustar a resposta dos canais RGB de uma imagem. Recebe como parâmetros uma imagem do tipo *CIImage* e podem ser alterados os valores de 5 pontos(x,y).

Apresenta-se de seguida, uma imagem ilustrativa do resultado da aplicação do filtro *CIToneCurve.*

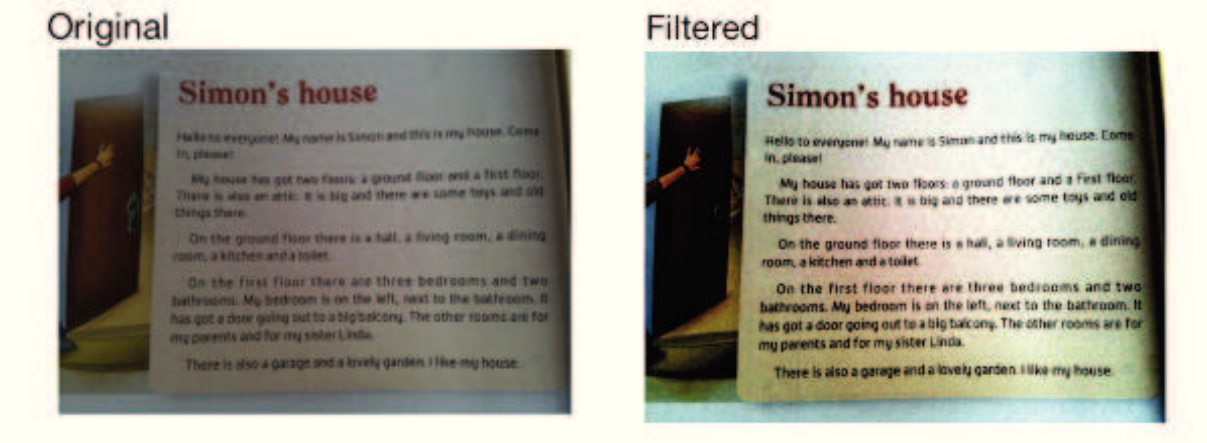


Figura 44 - Exemplo do resultado obtido após aplicação do filtro *CIToneCurve*

O filtro *CIHueAdjust*, altera a cor da generalidade dos pixéis de uma imagem. Recebe como parâmetros uma imagem do tipo CIImage e um ângulo, que determinará a rotação do cubo de cor em torno do eixo neutro.

Apresenta-se de seguida, uma imagem ilustrativa do resultado da aplicação do filtro *CIHueAdjust.*

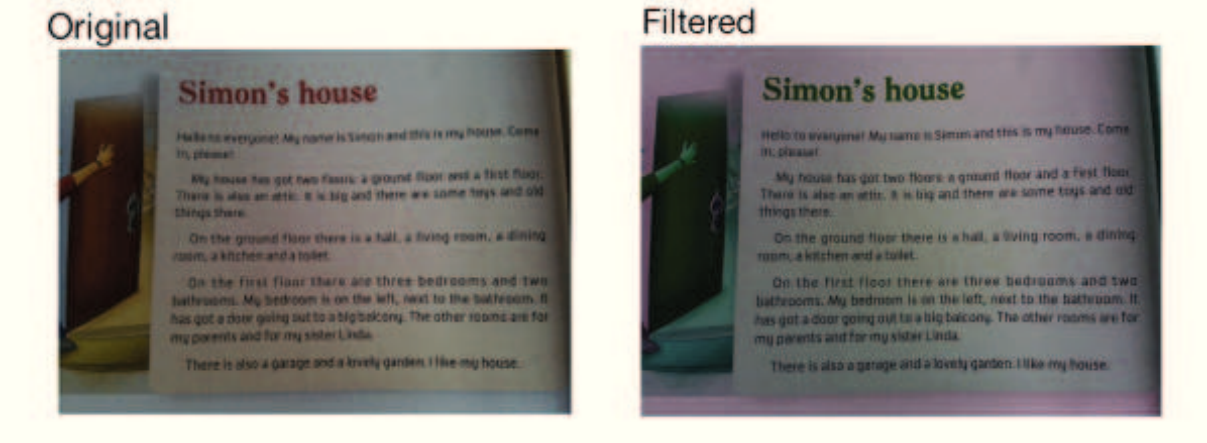


Figura 45 - Exemplo do resultado obtido após aplicação do filtro *CIHueAdjust*

Procedeu-se ao desenvolvimento de uma aplicação no XCode através da qual fosse possível obter a mediana do valor percentual da distância de *levenshtein,* resultante da diferença entre o texto original e os textos reconhecidos, após aplicação de cada um dos referidos filtros, no conjunto de 30 imagens recolhidas para testes.

De referir que nos testes para cada filtro, os valores dos seus diversos parâmetros foram sendo alterados em ciclo, procurando obter os melhores resultados. Foram desta forma apurados os valores para os parâmetros de cada filtro que permitiram a obtenção de melhores resultados.

Os resultados apresentam-se de seguida, em formato de tabela e em representação gráfica, para melhor visualização dos diferentes valores obtidos.

Filtro Aplicado	% da Mediana da distância
Sem Filtro	23,45%
CIGammaAdjust	22,97%
CIToneCurve	26,36%
CIColorMonochrome	20,42%
CIHueAdjust	22,78%
CIExposureAdjust	23,27%
CIColorControls	20,83%

Tabela 4 - Valores da mediana da distância obtido em imagens com filtro e sem filtro

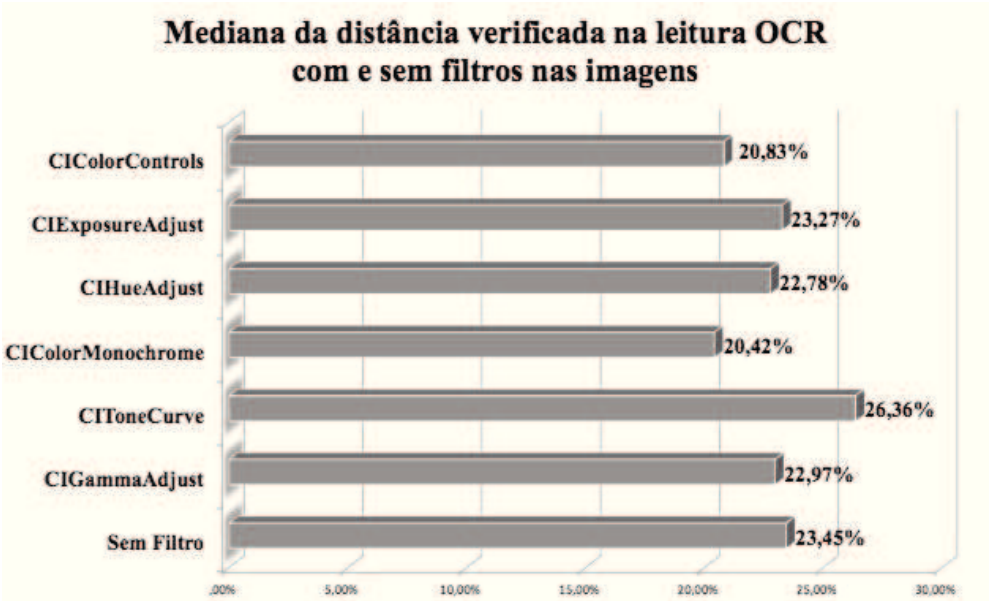


Figura 46 - Gráfico da mediana da distância obtida em imagens com filtro e sem filtro

Como pode ser verificado, à exceção do filtro *CIToneCurve*, a aplicação dos restantes filtros resultou em melhores resultados, tendo mesmo os filtros *CIColorMonochrome* e *CIColorControls* apresentado resultados bastante satisfatórios.

Verificados os resultados obtidos com a aplicação dos filtros individualmente, procedeu-se à utilização combinada de filtros, procurando melhorar ainda mais os resultados obtidos.

Foram realizados diversos testes combinando os filtros anteriormente mencionados, tendo sido registados vários resultados, uns com melhoria nos mesmos outros com resultados piores que o obtidos com a aplicação de apenas um filtro. Os valores definidos para cada parâmetro dos filtros combinados, foram os valores que forneceram os melhores resultados aquando da utilização dos filtros individualmente, sendo posteriormente ligeiramente

alterados esses valores até serem obtidos os melhores resultados.

Seguidamente, são apresentados alguns dos testes realizados, sendo igualmente apresentados o código da aplicação dos filtros, já com os valores que apresentaram os melhores resultados.

Combinação de filtros que altera as cores da imagem para preto e branco, aplicando sequencialmente os filtros *CIColorControls* e *CIExposureAdjust,* e os respetivos resultados obtidos, quer em tabela quer em representação gráfica.

```
//Create context
CIContext *context = [CIContext contextWithOptions:nil];

//Filter definition
CIImage *blackAndWhite = [CIFilter filterWithName:@"CIColorControls" keysAndValues:kCIInputImageKey,
    firstImage, @"inputBrightness", [NSNumber numberWithFloat:0.0], @"inputContrast", [NSNumber
    numberWithFloat:1.025], @"inputSaturation", [NSNumber numberWithFloat:0.0], nil].outputImage;

//filtered image
CIImage *filteredImage = [CIFilter filterWithName:@"CIExposureAdjust" keysAndValues:kCIInputImageKey,
    blackAndWhite, @"inputEV", [NSNumber numberWithFloat:0.7], nil].outputImage;


//Apply image in context
 cgiimage = [context createCGImage:filteredImage fromRect:filteredImage.extent];
 newImage = [UIImage imageWithCGImage:cgiimage];
```

Figura 47 – Aplicação dos filtros *CIColorControls* e *CIExposureAdjust*

Resultados obtidos após aplicados os filtros *CIColorControls* e *CIExposureAdjust* que resultam na transformação das imagens para preto e branco.

Imagem	N.º total de caracteres	Distância	Percentagem da distância
IMG 0	1032	**334**	32,36%
IMG 1	591	**359**	60,74%
IMG 2	495	**247**	49,90%
IMG 3	242	**115**	47,52%
IMG 4	308	**203**	65,91%
IMG 5	258	**135**	52,33%
IMG 6	331	**86**	25,98%
IMG 7	808	**84**	10,40%
IMG 8	761	**1209**	158,87%
IMG 9	599	**414**	69,12%
IMG 10	136	**62**	45,59%
IMG 11	137	**766**	559,12%
IMG 12	446	**3883**	870,63%
IMG 13	767	**184**	23,99%
IMG 14	588	**100**	17,01%
IMG 15	604	**95**	15,73%
IMG 16	603	**78**	12,94%
IMG 17	419	**156**	37,23%
IMG 18	653	**267**	40,89%
IMG 19	1041	**58**	5,57%
IMG 20	848	**73**	8,61%
IMG 21	1074	**81**	7,54%

Mediana da percentagem Distancia	
Valor da Mediana	**22,56%**

IMG 22	947	**50**	5,28%
IMG 23	688	**128**	18,60%
IMG 24	956	**202**	21,13%
IMG 25	737	**27**	3,66%
IMG 26	981	**58**	5,91%
IMG 27	623	**61**	9,79%
IMG 28	1855	**240**	12,94%
IMG 29	738	**46**	6,23%

Tabela 5 – Resultados obtidos após aplicação dos filtros *CIColorControls* e *CIExposureAdjust*

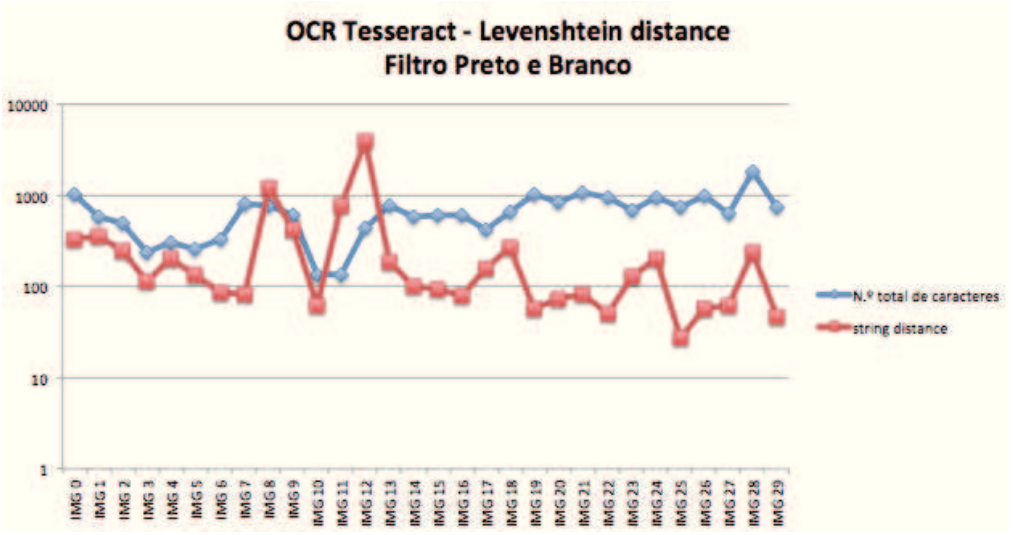


Figura 48 – Representação gráfica dos resultados obtidos após aplicação dos filtros *CIColorControls* e *CIExposureAdjust*

Após aplicação deste filtro, verificou-se uma ligeira melhoria nos resultados, com o valor da mediana da distância entre os textos a diminuir de 23,45% para 22,56%.

Segue-se a combinação do filtro preto e branco (utilizado anteriormente) e do filtro *CIColorControls* novamente, procurando alterar os valores da saturação, brilho e contraste.

```
//Create context
CIContext *context = [CIContext contextWithOptions:nil];

//Filter black and white definition
CIImage *blackAndWhite = [CIFilter filterWithName:@"CIColorControls" keysAndValues:kCIInputImageKey,
    firstImage, @"inputBrightness", [NSNumber numberWithFloat:0.0], @"inputContrast", [NSNumber
    numberWithFloat:1.025], @"inputSaturation", [NSNumber numberWithFloat:0.0], nil].outputImage;

//filtered image
CIImage *filteredImage = [CIFilter filterWithName:@"CIExposureAdjust" keysAndValues:kCIInputImageKey
    , blackAndWhite, @"inputEV", [NSNumber numberWithFloat:0.7], nil].outputImage;

//Filter CICOLORCONTROLS
CIFilter *brightnesContrastFilter = [CIFilter filterWithName:@"CIColorControls"];
[brightnesContrastFilter setDefaults];
[brightnesContrastFilter setValue: filteredImage forKey: @"inputImage"];
[brightnesContrastFilter setValue: [NSNumber numberWithFloat:0.5f]
                           forKey:@"inputBrightness"];
[brightnesContrastFilter setValue: [NSNumber numberWithFloat:2.14f]
                           forKey:@"inputContrast"];

CIImage *output = [brightnesContrastFilter valueForKey: @"outputImage"];

 //Apply image in context
CGImageRef cgiimage = [context createCGImage:output fromRect:output.extent];
UIImage *newImage = [UIImage imageWithCGImage:cgiimage];

CGImageRelease(cgiimage);
```

Figura 49 – Aplicação dos filtros preto e branco seguido do *CIColorcontrols*

Resultados obtidos após aplicados os filtros preto e branco e o filtro *CIColorControls*.

Imagem	N.º total de caracteres	Distância	Percentagem da distância
IMG 0	1032	**334**	32,36%
IMG 1	591	**307**	51,95%
IMG 2	495	**227**	45,86%
IMG 3	242	**100**	41,32%
IMG 4	308	**206**	66,88%
IMG 5	258	**155**	60,08%
IMG 6	331	**55**	16,62%
IMG 7	808	**78**	9,65%
IMG 8	761	**375**	49,28%
IMG 9	599	**414**	69,12%
IMG 10	136	**53**	38,97%
IMG 11	137	**70**	51,09%
IMG 12	446	**6267**	1405,16%
IMG 13	767	**237**	30,90%
IMG 14	588	**100**	17,01%
IMG 15	604	**95**	15,73%
IMG 16	603	**62**	10,28%
IMG 17	419	**150**	35,80%
IMG 18	653	**225**	34,46%
IMG 19	1041	**64**	6,15%
IMG 20	848	**73**	8,61%
IMG 21	1074	**84**	7,82%
IMG 22	947	**55**	5,81%
IMG 23	688	**127**	18,46%
IMG 24	956	**170**	17,78%
IMG 25	737	**34**	4,61%
IMG 26	981	**65**	6,63%
IMG 27	623	**54**	8,67%
IMG 28	1855	**171**	9,22%
IMG 29	738	**55**	7,45%

Mediana da percentagem Distancia	
Valor da Mediana	**18,12%**

Tabela 6 - Resultados obtidos após aplicação dos filtro preto e branco e *CIColorControls*

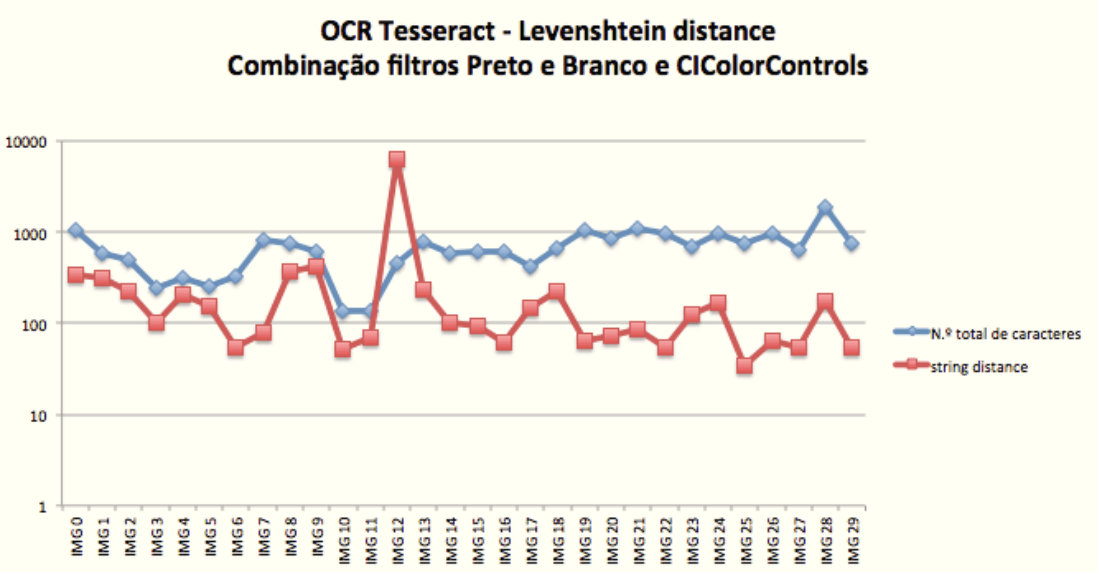


Figura 50 - Representação gráfica dos resultados obtidos após aplicação dos filtro preto e branco e *CIColorControls*

Após aplicação da combinação de filtros anterior, verificou-se uma substancial melhoria nos resultados, com o valor da mediana da distância entre os textos a diminuir de 23,45% para 18,12%.

Segue-se a combinação do dos filtros *CIColorsControls* e *CiColorMonochrome*

```
//Create context
CIContext *context = [CIContext contextWithOptions:nil];

//Filter CICOLORCONTROLS
CIFilter *brightnesContrastFilter = [CIFilter filterWithName:@"CIColorControls"];
[brightnesContrastFilter setDefaults];
[brightnesContrastFilter setValue: imagemTeste forKey: @"inputImage"];
[brightnesContrastFilter setValue: [NSNumber numberWithFloat:0.5f]
                           forKey:@"inputBrightness"];
[brightnesContrastFilter setValue: [NSNumber numberWithFloat:2.14f]
                           forKey:@"inputContrast"];

CIImage *outputImage = [brightnesContrastFilter valueForKey: @"outputImage"];

//Filter CIColorMonochrome
CIImage *output = [CIFilter filterWithName:@"CIColorMonochrome" keysAndValues:kCIInputImageKey,
    outputImage, @"inputIntensity", [NSNumber numberWithFloat:0.77], @"inputColor", [[CIColor alloc]
    initWithColor:[UIColor whiteColor]], nil].outputImage;

 //Apply image in context
CGImageRef cgiimage = [context createCGImage:output fromRect:output.extent];
UIImage *newImage = [UIImage imageWithCGImage:cgiimage];

CGImageRelease(cgiimage);
```

Figura 51 - Aplicação da combinação dos filtros *CIColorsControls* e *CiColorMonochrome*

Os resultados obtidos após aplicação desta combinação de filtros foram os seguintes:

Imagem	N.º total de caracteres	Distância	Percentagem da distância
IMG 0	1032	**375**	36,34%
IMG 1	591	**334**	56,51%
IMG 2	495	**257**	51,92%
IMG 3	242	**89**	36,78%
IMG 4	308	**192**	62,34%
IMG 5	258	**144**	55,81%
IMG 6	331	**55**	16,62%
IMG 7	808	**80**	9,90%
IMG 8	761	**381**	50,07%
IMG 9	599	**377**	62,94%
IMG 10	136	**44**	32,35%
IMG 11	137	**65**	47,45%
IMG 12	446	**3941**	883,63%
IMG 13	767	**242**	31,55%
IMG 14	588	**94**	15,99%
IMG 15	604	**76**	12,58%
IMG 16	603	**81**	13,43%
IMG 17	419	**150**	35,80%
IMG 18	653	**208**	31,85%
IMG 19	1041	**60**	5,76%
IMG 20	848	**63**	7,43%
IMG 21	1074	**96**	8,94%
IMG 22	947	**42**	4,44%
IMG 23	688	**131**	19,04%
IMG 24	956	**123**	12,87%
IMG 25	737	**39**	5,29%
IMG 26	981	**61**	6,22%

Mediana da percentagem Distancia	
Valor da Mediana	**17,83%**

IMG 27	623	**68**	10,91%
IMG 28	1855	**184**	9,92%
IMG 29	738	**61**	8,27%

Tabela 7 - Resultados obtidos após aplicação da combinação dos filtros *CIColorsControls* e *CiColorMonochrome*

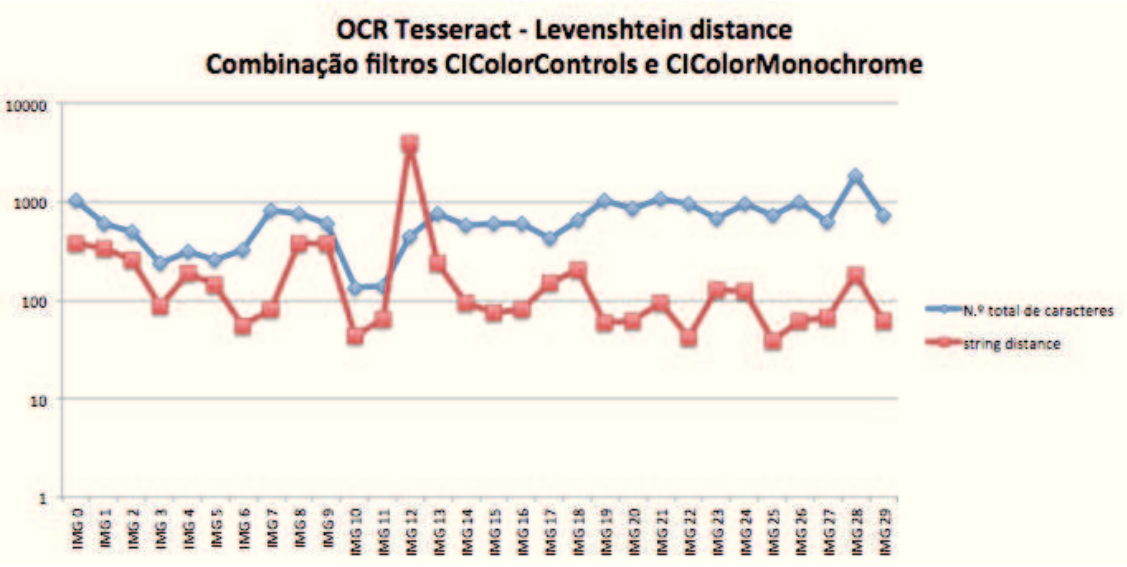


Figura 52 - Representação gráfica dos resultados obtidos após aplicação da combinação dos filtros *CIColorsControls* e *CiColorMonochrome*

Após aplicação desta combinação de filtros, verificou-se uma substancial melhoria nos resultados, com o valor mediano da distância entre o texto original e o reconhecido a diminuir de 23,45% para 17,83%.

Verificou-se que a combinação dos filtros *CIColorControls* e *CIColorMonochrome* originou os melhores resultados, pelo que esta combinação de filtros será sempre aplicada às imagens captadas pelo dispositivo, imediatamente antes do processo de reconhecimento OCR.

É apresentado de seguida um exemplo do resultado obtido numa imagem, após aplicação da combinação de filtros indicada:

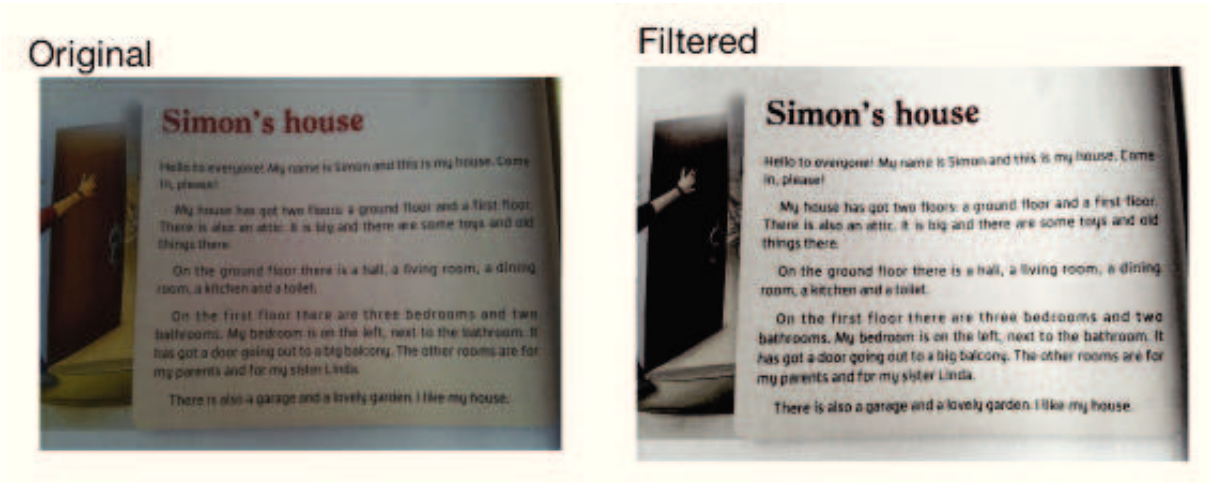


Figura 53 - Exemplo do resultado obtido após aplicação dos filtros *CIColorControls* e *CIColorMonochrome*

Após realizados os testes com filtros e a sua aplicação nas imagens, foram realizados alguns testes alterando as variáveis internas do próprio OCR *Tesseract*. No entanto, os testes realizados não resultaram em qualquer alteração dos resultados.

Limitações de tempo não permitiram realizar testes a todas as variáveis internas do *Tesseract*, tendo sido escolhidas variáveis que aparentemente poderiam exercer algum tipo de influência nos resultados.

Foram realizados testes com as seguintes variáveis internas do Tesseract:

- *image_default_resolution* (alterando a resolução das imagens). O valor por defeito do OCR é de 300 dpi, tendo sido feitos testes com maiores e menores valores.
- *textord_space_size_is_variable* (se o seu valor for verdadeiro, os espaços que delimitam as palavras são assumidos como variável, mesmo nos casos em que os caracteres têm tamanho fixo). O valor por defeito do OCR é "0", e foram testes com alteração desse valor para "1".
- *textord_oldbl_merge_parts* (une partições de caracteres que estejam separados mas que aparentemente deveriam estar unidos). O valor por defeito do OCR é "1" e foram feitos testes alterando esse valor para "0".

Tal como foi já referido, os testes realizados não acrescentaram melhorias no sistema, não se verificando diferenças nos resultados, logo não se realizam alterações às variáveis internas do OCR na aplicação final.

4.2 Alteração automática da orientação das imagens

A captação de imagens utilizando a classe *AVCaptureSession* apresentou um pequeno obstáculo: sempre que o dispositivo móvel estiver na orientação *Portrait*, as imagens captadas ficam numa orientação que não permite o reconhecimento OCR.

De frisar que a aplicação pode ser utilizada em qualquer das orientações do dispositivo, pelo que tornou-se imperativo realizar a rotação das imagens de forma automática e transparente ao utilizador.

Assim, em cada alteração da posição do dispositivo a aplicação determina a orientação atual do mesmo através da propriedade *UIInterfaceOrientation*, da classe *UIApplication*. Esta classe fornece um ponto centralizado de controle e coordenação para as aplicações iOS. A função desenvolvida que atualiza o estado refletido numa variável do tipo *BOOL PortraitOrientation* é apresentada na página seguinte.

```objc
//iPhone Orientation
//Indication of device orientation

- (void) adjustViewsForOrientation:(UIInterfaceOrientation) orientation {


    if (UIDeviceOrientationIsPortrait([UIDevice currentDevice].orientation))
    {
        //load the portrait view
        NSLog(@"Portrait");
        //[appdelegate.audioPlayerPortrait play];
        PortraitOrientation = YES;
    }
    else if (UIDeviceOrientationIsLandscape([UIDevice currentDevice].orientation))
    {
        //load the landscape view
        NSLog(@"Landscape");
        //[appdelegate.audioPlayerLandscape play];
        PortraitOrientation = NO;

    }

}
```

Figura 54 – Função que determina a orientação atual do dispositivo móvel

Conhecendo em cada momento a orientação atual do dispositivo, criou-se uma função que, baseada nessa orientação procede à rotação necessária da imagem, para que esta possa ser reconhecida pelo OCR.

A implementação dessa função em Objective-C encontra-se seguidamente apresentada.

```objc
//Image Rotation
//According to the device orientation, image is rotated for OCR recognition
//Return image rotated

-(UIImage *)imageOrientation:(UIImage *)image {

    CGImageRef imageRef = [image CGImage];
    CGImageAlphaInfo alphaInfo = CGImageGetAlphaInfo(imageRef);
    CGColorSpaceRef colorSpaceInfo = CGColorSpaceCreateDeviceRGB();

    if (alphaInfo == kCGImageAlphaNone)
        alphaInfo = kCGImageAlphaNoneSkipLast;
    //size of final image
    int width, height;
    width = 1024;
    height = 1024;

    CGContextRef bitmap;

    if (image.imageOrientation == UIImageOrientationUp | image.imageOrientation == UIImageOrientationDown) {
        bitmap = CGBitmapContextCreate(NULL, width, height, CGImageGetBitsPerComponent(imageRef),
            CGImageGetBytesPerRow(imageRef), colorSpaceInfo, (CGBitmapInfo)alphaInfo);
    } else {
        bitmap = CGBitmapContextCreate(NULL, height, width, CGImageGetBitsPerComponent(imageRef),
            CGImageGetBytesPerRow(imageRef), colorSpaceInfo, (CGBitmapInfo)alphaInfo);
    }
    if (image.imageOrientation == UIImageOrientationLeft) {
        NSLog(@"image orientation left");
        CGContextRotateCTM (bitmap, 90 * M_PI / 180);
        CGContextTranslateCTM (bitmap, 0, -height);
    } else if (image.imageOrientation == UIImageOrientationRight) {
        NSLog(@"image orientation right");
        CGContextRotateCTM (bitmap, -90* M_PI / 180);
        CGContextTranslateCTM (bitmap, -width, 0);
    } else if (image.imageOrientation == UIImageOrientationUp) {
        NSLog(@"image orientation up");
        CGContextRotateCTM (bitmap, -90* M_PI / 180);
        CGContextTranslateCTM (bitmap, -width, 0);
    } else if (image.imageOrientation == UIImageOrientationDown) {
        NSLog(@"image orientation down");
        CGContextTranslateCTM (bitmap, width,height);
        CGContextRotateCTM (bitmap, -180 * M_PI / 180);
    }
```

```
    CGContextDrawImage(bitmap, CGRectMake(0, 0, width, height), imageRef);
    CGImageRef ref = CGBitmapContextCreateImage(bitmap);
    UIImage *result = [UIImage imageWithCGImage:ref];

    CGContextRelease(bitmap);
    CGImageRelease(ref);

    return result;
}
```

Figura 55 – Função que procede automaticamente à rotação da imagem

4.3 Prevenção de erro no resultado final

Durante a realização de testes da aplicação, nomeadamente quando se utiliza a captação de imagens com a máquina fotográfica do dispositivo, verificou-se que em determinadas leituras, o resultado do reconhecimento OCR era um conjunto de caracteres sem sentido, e que em nada correspondiam ao texto constante das imagens. Tal resultado era originado pela deficiente qualidade da imagem captada, ou por erros de reconhecimento do OCR.

No entanto, uma vez que esse conjunto de caracteres correspondiam a texto era realizada a sintetização oral, o que revelava-se enganador e mesmo desagradável ao utilizador.

Depois de realizados vários testes, e observados os textos resultantes do reconhecimento óptico, verificou-se que o padrão normal das *strings* nos casos de leituras incorretas apresentava vários caracteres especiais, seguidamente indicados:

( ) ; * ~ » “ . =

O conjunto de carateres indicado baseou-se na observação de vários exemplos testados, podendo naturalmente em versões futuras ser alterados, ou até serem incluídos novos caracteres a este conjunto, bastando para tal incluí-los na função.

Para evitar as situações de reconhecimento inválido procedeu-se à utilização de expressões regulares para testar se o texto devolvido pelo OCR correspondia a um texto legível ou se devolvia muitas ocorrências dos caracteres especiais indicados. Assim, a função criada recebe o texto reconhecido pelo OCR, conta o total de caracteres do texto, e conta o total de ocorrências dos carateres especiais supracitados, calculando um valor percentual desses carateres especiais em relação ao total de caracteres do texto, denominado por percentagem de erro, obtido através da fórmula seguinte:

$$Percentagem\ de\ erro = \frac{Ocorrência\ de\ caracteres\ especiais \times 100}{Total\ de\ caracteres}$$

Se o valor da percentagem de erro ultrapassar um determinado valor, que pode ser configurado pelo utilizador, ao invés de a aplicação realizar a sintetização do texto, apresenta uma mensagem de erro no processamento, solicitando ao utilizador que repita o processo de captura da imagem.

Apresenta-se de seguida a função anteriormente descrita.

```
//Function for testing result of OCR recognition
//Counts number of special chars, and total chars
//Return the percent of special chars in a text

- (double)detect:(NSString*)texto {

    //Regular expression counting special characteres that appears on OCR error

    //Definition of special chars
    NSRegularExpression *regex = [NSRegularExpression regularExpressionWithPattern:@"\\;|\\*|\\~|\\=|\"|\\.|
        \\}|\\{|\\=" options:NSRegularExpressionCaseInsensitive error:NULL];

    //contar o numero de ocorrências
    NSUInteger numberOfMatches = [regex numberOfMatchesInString:texto options:0 range:NSMakeRange(0, [texto
        length])];

    //obtem percentagem de caracteres
    double percentagem = ((double)numberOfMatches/(double)[texto length]) *100;

    NSLog(@"Percentagem: %f\n%i\n%i",percentagem, numberOfMatches, [texto length]);

    //return percent of special chars in the text
    return percentagem;
}
```

Figura 56 – Função que obtém a percentagem de caracteres especiais existentes num texto.

No sentido de aferir o valor percentual inicial para a aplicação, e definir o intervalo dos valores a serem alterados, procedeu-se a um conjunto de testes utilizando o conjunto de imagens já utilizadas nos restantes testes.

Assim, procedeu-se a testes de reconhecimento do texto nas imagens, sendo definidos os valores percentuais de caracteres especiais permitidos, e nos casos em que esses valores fossem ultrapassados, as situações eram consideradas como erro de processamento.

Foram assim assinaladas as imagens cujo reconhecimento foi considerado como erro de processamento, logo não sendo consideradas para os resultados finais, tendo como consequência a alteração da percentagem da mediana da distância de *Levenshtein,* uma vez que os textos com piores resultados foram sendo retirados. De referir que o valor da percentagem de erro obtido na imagem que apresenta a maior distância é de 18,54%, e o valor seguinte mais alto de 7,78%, sendo o mais baixo de 0,75%. Para a realização dos testes foram considerados valores entre 10% e 1%, em intervalos de 0,5%.

Os valores obtidos nos testes apresentam-se de seguida:

Percentagem de erro	Percentagem do valor da mediana da distância	Imagens descartadas	Quantidade de imagens descartadas
10%	16,62%	IMG 12	1
9.5%	16,62%	IMG 12	1

9%	16,62%	IMG 12	1
8.5%	16,62%	IMG 12	1
8%	16,62%	IMG 12	1
7.5%	16,30%	IMG 5, IMG 12	2
7%	16,30%	IMG 5, IMG 12	2
6.5%	15,99%	IMG 5, IMG 9, IMG 12	3
6%	15,99%	IMG 5, IMG 9, IMG 12	3
5.5%	15,99%	IMG 5, IMG 9, IMG 12	3
5%	13,43%	IMG 1, IMG 4, IMG 5, IMG 9, IMG 12	5
4.5%	13,15%	IMG 1, IMG 2, IMG 4, IMG 5, IMG 9, IMG 12	6
4%	12,87%	IMG 0, IMG 1, IMG 2, IMG 4, IMG 5, IMG 9, IMG 12	7
3.5%	12,87%	IMG 0, IMG 1, IMG 2, IMG 4, IMG 5, IMG 9, IMG12	7
3%	12,87%	IMG 0, IMG 1, IMG 2, IMG 4, IMG 5, IMG 9, IMG 12, IMG 17, IMG 26	9
2.5%	9,91%	IMG 0, IMG 1, IMG 2, IMG 4, IMG 5, IMG 6, IMG 9, IMG 12, IMG 13, IMG 14, IMG 15, IMG 16, IMG 17, IMG 18, IMG 23, IMG 26	16
2%	8,94%	IMG 0, IMG 1, IMG 2, IMG 3, IMG 4, IMG 5, IMG 6, IMG 7, IMG 8, IMG 9, IMG 10, IMG 12, IMG 13, IMG 14, IMG 15, IMG 16, IMG 17, IMG 18, IMG 23, IMG 26	20
1.5%	6,60%	IMG 0, IMG 1, IMG 2, IMG 3, IMG 4, IMG 5, IMG 6, IMG 7, IMG 8, IMG 9, IMG 10, IMG 12, IMG 13, IMG 14, IMG 15, IMG 16, IMG 17, IMG 18, IMG 21, IMG 23, IMG 24, IMG 26, IMG 28, IMG 29	24
1%	25,94%	IMG 0, IMG 1, IMG 2, IMG 3, IMG 4, IMG 5, IMG 6, IMG 7, IMG 8, IMG 9, IMG 10, IMG 12, IMG 13, IMG 14, IMG 15, IMG 16, IMG 17, IMG 18, IMG 19, IMG 20, IMG 21, IMG 23, IMG 24, IMG 25, IMG 26,IMG 27, IMG 28, IMG 29	28

Tabela 8 – Resultados da realização de testes utilizando a função de prevenção de erro no resultado final

A representação gráfica dos valores da mediana, após realizados os testes podem ser visualizados no gráfico da página seguinte, sendo visível a curva descendente desses valores.

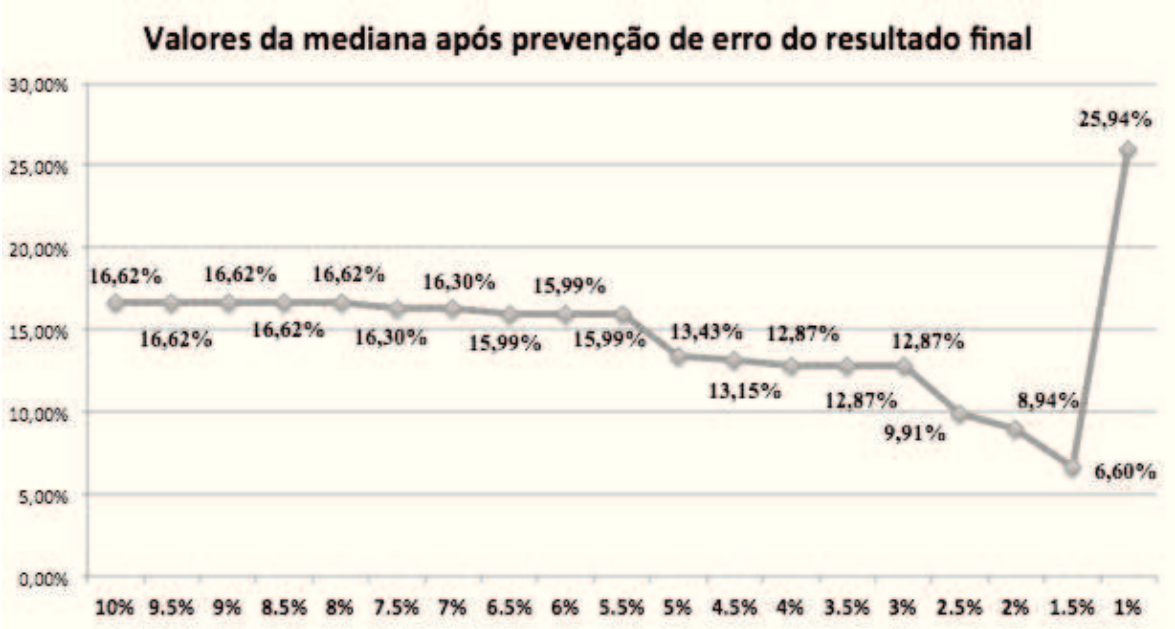


Figura 57 – Representação gráfica do resultado dos testes utilizando a função de prevenção de erro no resultado final

Através da análise dos resultados, pode verificar-se que quanto menor for a percentagem de caracteres especiais permitidos, melhor são os resultados do valor percentual da mediana do erro obtido. Tal decorre do facto de as imagens com piores resultados no reconhecimento OCR serem descartadas. A exceção na trajetória descendente do valor percentual da mediana verifica-se no valor de 1%, uma vez que neste valor apenas são consideradas como válidas a leitura de duas imagens, não podendo esse valor ser considerado como um valor de referência para concluir da sua utilização.

Verificou-se que para percentagens inferiores a 3% a quantidade de imagens descartadas é muito elevada, ultrapassando 50% do total das 30 imagens testadas. A utilização de uma percentagem tão baixa poderá tornar a aplicação pouco eficaz, apresentando muitas vezes a mensagem de erro de reconhecimento.

O valor por defeito na primeira utilização da aplicação será de 5%, uma vez que, após realizados os testes utilizando este valor como referência, verificou-se que, embora sendo descartadas 5 imagens do conjunto das 30 testadas, a que corresponde uma percentagem de aproximadamente 16,7%, o valor da percentagem da mediana da distância obtido é de 13,43%, a que corresponde uma melhoria substancial face ao valor verificado sem a prevenção do erro, 17,83%.

Naturalmente o utilizador poderá optar por alterar o valor definido para a percentagem, podendo aumentar a mesma, e obter mais resultados incorretos e menos mensagens de erro, ou diminuir a percentagem, e obter resultados mais eficazes mas maior número de situações de erro de reconhecimento.

4.4 Conclusão

O processo de optimização do sistema revelou-se de extrema importância para a obtenção de uma melhor aplicação.

Foram realizados diversos testes, alguns bastante morosos como foi o caso dos testes dos filtros, cuja execução dos programas teste, com ciclos para verificação dos diversos parâmetros dos filtros ao mesmo tempo que percorria o conjunto das 30 imagens do banco de imagens, demoraram várias horas a executar e outras tantas a analisar os resultados a fim de obter as melhores conclusões.

A alteração automática da rotação das imagens revelou-se de grande importância uma vez que possibilita que a aplicação seja executada nas diversas orientações do dispositivo móvel, possibilitando a captação de textos que ocupem uma maior área horizontal ou vertical.

A aplicação de uma função de prevenção de erro no resultado final evita o desconforto da obtenção da leitura de caracteres estranhos, ao invés da leitura do texto correto pretendido. Verificou-se, após realização de testes em que o valor da prevenção de erros foi definido para 5%, que a percentagem da mediana da distância entre os textos originais e os textos reconhecidos, apresentou uma diminuição, passando de 17,83% para 13,43%, o que representa um acréscimo substancial da eficácia da aplicação. A possibilidade de esse valor ser configurado pelo utilizador permite que cada utilizador possa adaptar a aplicação ao seu próprio critério.

Após realizados os testes e optimizações, verificam-se melhorias substanciais na aplicação, tornando-a mais fácil de utilizar, mais eficiente e eficaz. O melhor resultado do sistema foi obtido aplicando às imagens captadas a combinação de filtros *CIColorControls* e *CIColorMonochrome* e definindo um valor para a prevenção de erro no resultado final de 5%.

5 – Conclusão

Várias foram as fases ultrapassadas com sucesso no desenrolar deste projeto. Desde o conhecimento e familiarização com novas plataformas tecnológicas, as plataformas móveis, até o conhecimento de uma nova linguagem de programação e um ambiente igualmente novo e desconhecido, o Xcode passando pelo conhecimento de *frameworks* OCR e TTS, e a sua utilização, quer de forma isolada quer em conjunto, até a criação de uma aplicação, que foi sendo melhorada e afinada ao longo do projeto, após ter sido sujeita a um conjunto de testes e optimizações.

Foi realizado um trabalho de pesquisa relativamente às tecnologias OCR e TTS que foram utilizadas no desenvolvimento aplicação, procurando conhecer os métodos que estão por detrás das mesmas, e conhecer com maior profundidade os mecanismos que realizam o reconhecimento óptico de caracteres em imagens e a sintetização de voz de textos.

Foi também feita uma análise das aplicações já existentes no mercado, similares à proposta neste trabalho ou que utilizam tecnologias OCR e TTS, verificando-se a existências de trabalhos já realizados, mas também de lacunas várias no que diz respeito a aplicações que realizem o reconhecimento e/ou sintetização de voz sem limitações, ou a custos baixos, dando mais sentido ao trabalho desenvolvido neste projeto de Mestrado.

Houve igualmente a necessidade de conhecer um dos algoritmos mais utilizados na comparação de *strings*, o algoritmo da distância de *Levenshtein*. Trata-se de um algoritmo concebido há vários anos, mas com uma atualidade e utilidade bem presentes. O algoritmo permite comparar *strings*, contabilizando o número de operações necessárias para, partindo de uma *string*, chegar até à outra. Quanto menor for o número dessas operações, mais idênticos serão as *strings* testadas, correspondendo a distância zero a *strings* idênticas.

Foram ultrapassadas com sucesso várias fases na implementação, primeiro de *frameworks* OCR, depois de *frameworks* TTS e por fim a combinação de ambas numa aplicação iOS. O trabalho permitiu obter conhecimentos destas *frameworks*, a nível científico

e técnico e verificar que conjugadas podem criar uma aplicação interessante e de grande utilidade para utilizadores invisuais.

O projeto consistiu portanto na construção de uma aplicação constituída por várias partes, integrando o sistema de captação de imagem do dispositivo móvel, que é utilizado por uma *framework* OCR para reconhecimento do seu texto, que é seguidamente sintetizado através de um processo de TTS.

As optimizações levadas a cabo para melhoria dos resultados tornaram a aplicação mais eficiente e mais capaz de responder ao repto lançado pelo tema do projeto: uma câmara que leia textos para cegos (*Camera Reading for Blind People*). Uma vez verificado que a utilização da imagem captada pelo dispositivo apresentava resultados pouco satisfatórios no que concerne ao reconhecimento do texto nela constante, procedeu-se a um pré-processamento da imagem antes desta ser submetida ao reconhecimento óptico. Após analisados diversas possibilidades de tratamento das imagens, a opção recaiu na utilização de uma combinação de filtros (*CIColorControls* e *CIColorMonochrome)* que demonstraram melhorar substancialmente os resultados.

Outra das optimizações levadas a cabo para melhorar o projeto, foi através da análise do resultado do reconhecimento óptico, procurando aferir da sua utilidade enquanto resultado a apresentar. Verificou-se que em diversas situações, fruto de leituras incorretas do OCR, o resultado final representava um conjunto de caracteres sem sentido, mas como representavam texto, o TTS realizava a sua sintetização, o que se tornava incómodo ao utilizador. Assim, foi desenvolvida uma função, que recorrendo a expressões regulares, analisa o resultado do reconhecimento OCR, e nos casos em que esse resultado apresente uma percentagem de caracteres especiais elevada, o sistema indica que houve erro de leitura pedindo ao utilizador que repita o processo.

A experiência de realização do trabalho foi bastante positiva, uma vez que foi possível desenvolver um trabalho de pesquisa relevante para a minha formação e ao mesmo tempo poder conceber uma aplicação que poderá vir a ser muito importante para indivíduos invisuais, melhorando substancialmente a sua autonomia e qualidade de vida.

Embora a aplicação final não represente ainda uma solução plenamente utilizável e totalmente fiável para os invisuais no dia a dia, o programa desenvolvido e os resultados obtidos podem considerar-se já interessantes, verificando-se que em algumas imagens, se reunidas as condições ideias, esses resultados são mesmo bastante interessantes, decorrendo um reconhecimento e consequente leitura muito eficazes e consistentes com o texto original.

O resultado final alcançado não é perfeito, uma vez que apresenta algumas lacunas no

que diz respeito ao reconhecimento em tempo real das imagens, verificando-se por vezes alguma lentidão no processo de reconhecimento de grandes e complexos textos. Também o facto de o processo de captação das imagens não estar dotado de um sistema automático de ajuda ao utilizador, que o oriente para que a captação da imagem seja o mais correta possível, apresenta-se como uma limitação desta aplicação, uma vez que a aplicação destina-se a invisuais que poderão ter mais algumas dificuldades em realizar essa tarefa. No entanto, essa limitação poderá e deverá ser alvo de melhoria em trabalhos futuros.

O trabalho de pesquisa, implementação e optimização desenvolvidos, permitiu conceber uma aplicação gratuita, que se encontra já num estado de possível utilização, mesmo com as limitações atrás referenciadas, permitindo a leitura de textos, desde que as condições de luz sejam as ideias para captação da imagem e o equipamento seja corretamente direcionado ao texto que se pretende ouvir, para que o reconhecimento e a leitura sejam o mais satisfatória possível.

5.1 Trabalho futuro

Embora tenha sido já percorrido um caminho importante, podem sempre ser alcançados mais e melhores resultados no desenvolvimento deste projeto. A aplicação pode e deve conhecer melhorias no futuro.

A *framework open source Tesseract* utilizada no projeto, pode ser substituída por *frameworks* comerciais, uma vez que, embora represente custos, a qualidade dos resultados obtidos no reconhecimento óptico poderão melhorar substancialmente. De referir que a utilização das *frameworks* comerciais apresentou melhores resultados que a *framework* gratuita, quando utilizadas sem qualquer filtro, o que leva a crer que a utilização dessas *frameworks* em imagens que tenham sido submetidas a filtros para melhorar o reconhecimento levará a resultados ainda mais satisfatórios.

Também os parâmetros alterados para obter melhores resultados podem ser alvo de alterações e realizados novos testes, que possam permitir alcançar melhores resultados. Por exemplo o alargamento da base de dados de imagens, procurando obter um conjunto mais vasto e representativo de situações reais de utilização podem fornecer dados mais consistentes, mas também a realização de mais testes com novas combinações de filtros, ou com mais e exaustivas alterações aos parâmetros internos do OCR.

Um outro teste que poderá ser importante para aferir a qualidade do sistema, é solicitar a um utilizador que ouça a leitura de um texto, que tenha sido reconhecido e sintetizado pelo

sistema, que o escreva, e posteriormente seja feita a comparação entre o texto compreendido pelo utilizador e o texto original.

O projeto poderá ser alvo de alterações, como por exemplo no que diz respeito ao alinhamento da imagem a ser captada. Recordar que se trata de uma aplicação para invisuais, que devem poder utilizar autonomamente a aplicação, pelo que no futuro a aplicação deve auxiliar o utilizador no alinhamento do dispositivo com o texto a reconhecer. Essa ajuda pode passar pela detecção da área onde se encontra o texto, e através de sinais sonoros dar indicações ao utilizador para orientar o dispositivo e conseguir captar todo o texto corretamente. O reconhecimento pode eventualmente ser feito através de um processo de leitura dos pixéis da imagem, procurando encontrar o limite entre padrões mais claros, onde possivelmente não haja texto, como nas margens de uma folha, e padrões mais escuros, onde potencialmente se encontre texto.

Outras das questões que deve ser alvo de melhorias é no reconhecimento em tempo real, uma vez que para um utilizador invisual a possibilidade de obter uma leitura em tempo real significaria um acréscimo de qualidade de vida, diminuindo o tempo que precisa para ler um documento mais extenso.

O suporte para outras linguagens deverá ser outra das questões a serem revistas. Nesta fase o projeto foi pensado e desenvolvido para a língua inglesa, mas no futuro devera permitir a utilização em outras línguas, alargando assim o número de pessoas que podem beneficiar das vantagens da aplicação.

Fica também reservado para trabalho futuro o incremento das opções de menu, permitindo ao utilizador alterar outras opções da aplicação tais como a língua em que esta funciona ou configurações de sons e orientações.

6 – Bibliografia

- ABBYY. *ABBYY FineReader for Personal Use*. Disponível em http://finereader.abbyy.com/
- ABBYY. *O que é OCR.* Disponível em http://www.abbyy.com.br/ocr/
- ACAPELA. *Acapela TTS for iOS*. Disponível em http://www.acapela-group.com/acapela-for-ios/
- Acapela, *How does it work?*. Disponível em http://www.acapela-group.com/voices/how-does-it-work/
- BIHINA, M. M. B. *The Phone Reader.* Grahamstown, South Africa, November, 2012.
- COHEN M., GIANGOLA J., AND BALOGH J., *Voice User Interface Design*. Addison Wesley, 2004
- CONDADO, P., *Quebra de barreiras de comunicação para portadores de paralisia cerebral.* Doutoramento em Engenharia Electrónica e Computação, Ciências de Computação, Universidade do Algarve, 2009
- CROCHIERE,R.E.; FLANAGAN,J.L. *Speech Processing: an Evolving Technology*. AT & T Technical Journal, v. 65, n. 5, p. 2-11, Sept/Oct. 1990.
- DUTOIT, T. *A Short Introduction to Text-to-Speech* Kluwer Academic Publishers, Dordrecht, Boston, London, 1997.
- EIKVIL, LINE. OCR Optical Character Recognition, 1993
- ELMORE, M. AND MARTONOSI, M. *A morphological image preprocessing suite for ocr* on natural scene images, 2008
- EXPERVISION. Disponível em http://en.wikipedia.org/wiki/ExperVision
- EXPERVISION. *OCR SDK for Mobile and Embedded System.* Disponível em http://www.expervision.com/ocr-sdk-toolkit/ocr-sdk-for-embedded-mobile-system-iphone-ipad-android-wince

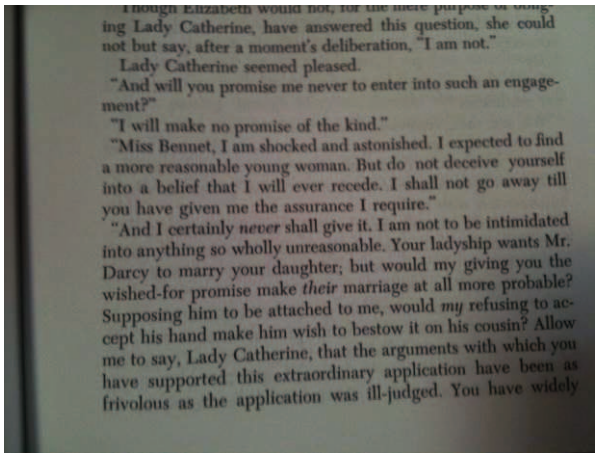
- FONSECA, N., REIS, C., SILVA, C., MARCELINO, L., CARREIRA, V. *Desenvolvimento em iOS iPhone, iPad e iPod Touch*, Lisboa, FCA – Editora de Informática, Lda, 2012.
- FUJISAWA, H. A view on the past and future of character and document recognition. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition* – Volume 01, pages 3-7, Washington, DC, USA. IEEE Computer Society, 2007.
- GOMES, MIGUEL. *Reconhecimento Óptico de Dígitos Impressos,* Instituto Superior de Engenharia de Lisboa, 1999.
- GOOGLE TTS, *Google-TTS-Library-For-iOS*. Disponível em https://www.github.com/shamsudheen/Google-TTS-Library-For-iOS
- JOSHI, A., ZHANG, M., KADMAWALA, R, DANTU, K., PODURI S. AND SUKHATME, G.S. *OCRdroid: A Framework to Digitize Text Using Mobile Phones*. Computer Science Department, Electrical Engineering Department University of Southern California, Los Angeles, CA 90089, USA, 2009
- iOS – *Documentation and API Reference*. Disponível em https://developer.apple.com/library/ios/navigation/
- ISPEECH. *iPhone/iPad SDK Documentation*. Disponível em https://www.ispeech.org/developers/iphone
- LEADTOOLS. *About LEAD Technologies, Inc.* Disponível em http://www.leadtools.com/corporate/corporate.htm
- LEMMETTY, S. *Review of Speech Synthesis Technology*. Masters Thesis, Helsinki University, 1999.
- LEVENSHTEIN, V. I. *Binary codes capable of correcting deletions, insertions, and reversals*. Doklady Akademii Nauk SSSR, 163(4), 845–848, 1965.MANGNES, B. *The use of Levenshtein distance in computer forensics*. Department of Computer Science and Media Technology Gjøvik University College, 2005
- MANGNES, B. *The use of Levenshtein distance in computer forensics*. Master's Thesis. Master os Sciente in Information Security. Department of Computer Science and Media Technology, Gjøvik University College, 2005
- MESHESHA, MILLION. Recognition and Retrieval from Document Image, 2008
- MITHE, R., INDALKAR S., DIVEKAR, N. *Optical Character Recognition*. International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-2, Issue-1, March 2013

- OCRAPISERVICE. *Online OCR API, Cloud based and pay as you go.* Disponível em http://www.ocrapiservice.com/
- OPENEARS. *OpenEars: free speech recognition and speech synthesis for the iPhone.* Disponível em http://www.politepix.com/openears/
- PATEL, UMAL. *An Introduction to the Process of Optical Character Recognition*. International Journal of Science and Research (IJSR), India Online, 2013.
- PISONI D. B., "Perception of synthetic speech", in Progress in Speech Synthesis. New York, USA: springer-Verlag, 1997.
- SILVA, A. *Uma Avaliação Experimental da Combinação de Métricas de Similaridade para o Alinhamento de Ontologias através de Mineração de Dados*. UNIRIO, 2013. Dissertação de Mestrado. Departamento de Informática Aplicada, UNIRIO.
- SMITH, R. An Overview of the Tesseract OCR Engine. Google Inc, 2007.
- SPEECH SYNTHESIS (Speech at CMU). Speech Synthesis Introduction. Disponível em http://www.speech.cs.cmu.edu/comp.speech/FAQ5.html, e em http://en.wikipedia.org/wiki/Speech_synthesis
- TESSERACT. *Wiki de suporte da framework* Tesseract, disponível em https://code.google.com/p/tesseract-ocr/w/list
- TEXTGRABBER + TRANSLATOR. *TextGrabber + Translator para iPhone, iPod touch e iPad na App Store no iTunes.* Disponível em https://itunes.apple.com/PT/app/textgrabber/id438475005
- THOMAS, S. *Natural Sounding Text-To-Speech Synthesis Based On Syllable-Like Units*, Department Of Computer Science And Engineering Indian Institute Of Technology Madras, 2007
- U.D REICHEL AND H.R. PFITZINGER, *Text processing for speech synthesis* in Proceedings of the TC-STAR Workshop on Speech-to-Speech Translation, (Barcelona) , June 2006

7 – Anexos

7.1 Fotos utilizados nos testes

Em anexo apresentam-se as fotos recolhidas em diversos livros e utilizadas em toda a fase de testes com as diversas *frameworks* OCR e posteriormente na otimização da OCR Tesseract.


ing Lady Catherine, have answered this question, she could not but say, after a moment's deliberation, "I am not."

Lady Catherine seemed pleased.

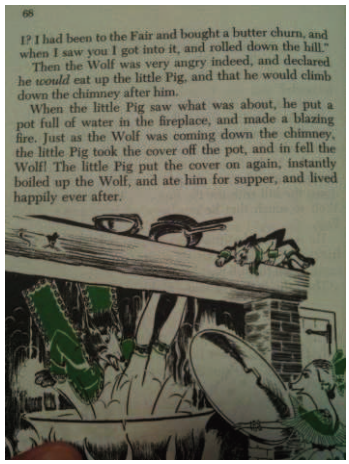
"And will you promise me never to enter into such an engagement?"

"I will make no promise of the kind."

"Miss Bennet, I am shocked and astonished. I expected to find a more reasonable young woman. But do not deceive yourself into a belief that I will ever recede. I shall not go away till you have given me the assurance I require."

"And I certainly *never* shall give it. I am not to be intimidated into anything so wholly unreasonable. Your ladyship wants Mr. Darcy to marry your daughter; but would my giving you the wished-for promise make *their* marriage at all more probable? Supposing him to be attached to me, would *my* refusing to accept his hand make him wish to bestow it on his cousin? Allow me to say, Lady Catherine, that the arguments with which you have supported this extraordinary application have been as frivolous as the application was ill-judged. You have widely

img0.jpg

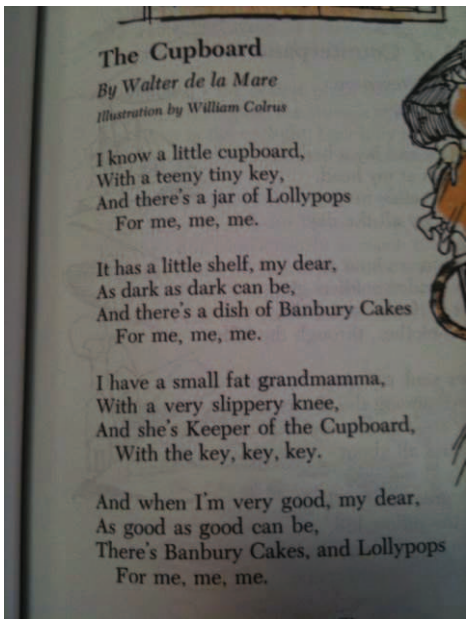

68

I? I had been to the Fair and bought a butter churn, and when I saw you I got into it, and rolled down the hill."

Then the Wolf was very angry indeed, and declared he *would* eat up the little Pig, and that he would climb down the chimney after him.

When the little Pig saw what was about, he put a pot full of water in the fireplace, and made a blazing fire. Just as the Wolf was coming down the chimney, the little Pig took the cover off the pot, and in fell the Wolf! The little Pig put the cover on again, instantly boiled up the Wolf, and ate him for supper, and lived happily ever after.

img1.jpg


The Cupboard

By Walter de la Mare

Illustration by William Colrus

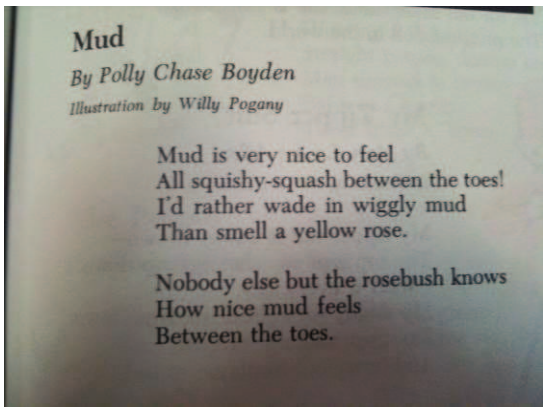
I know a little cupboard,
With a teeny tiny key,
And there's a jar of Lollypops
For me, me, me.

It has a little shelf, my dear,
As dark as dark can be,
And there's a dish of Banbury Cakes
For me, me, me.

I have a small fat grandmamma,
With a very slippery knee,
And she's Keeper of the Cupboard,
With the key, key, key.

And when I'm very good, my dear,
As good as good can be,
There's Banbury Cakes, and Lollypops
For me, me, me.

img2.jpg


Mud

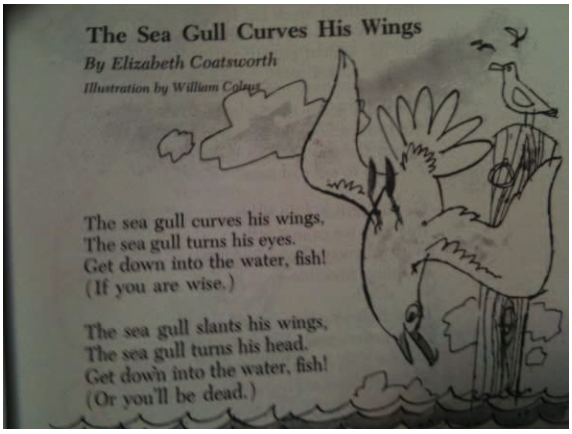
By Polly Chase Boyden

Illustration by Willy Pogany

Mud is very nice to feel
All squishy-squash between the toes!
I'd rather wade in wiggly mud
Than smell a yellow rose.

Nobody else but the rosebush knows
How nice mud feels
Between the toes.

img3.jpg



The Sea Gull Curves His Wings

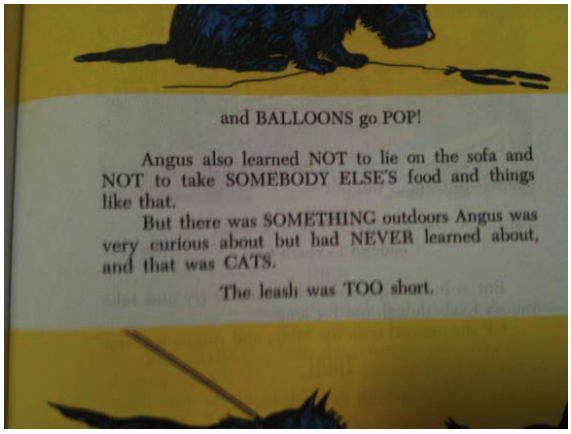
By Elizabeth Coatsworth

Illustration by William Colvin

The sea gull curves his wings,
The sea gull turns his eyes.
Get down into the water, fish!
(If you are wise.)

The sea gull slants his wings,
The sea gull turns his head.
Get down into the water, fish!
(Or you'll be dead.)

img4.jpg



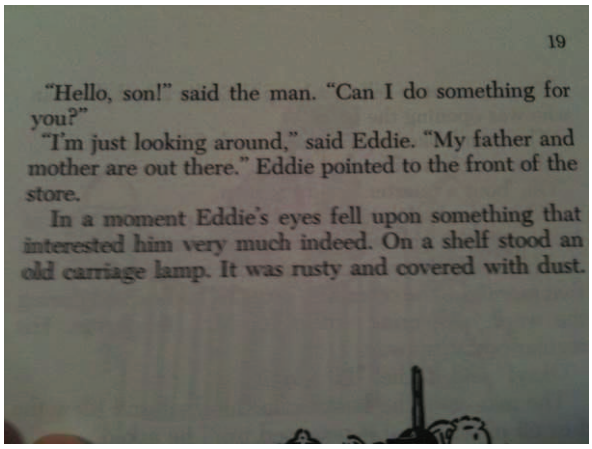
and BALLOONS go POP!

Angus also learned NOT to lie on the sofa and NOT to take SOMEBODY ELSE'S food and things like that.

But there was SOMETHING outdoors Angus was very curious about but had NEVER learned about, and that was CATS.

The leash was TOO short.

img5.jpg



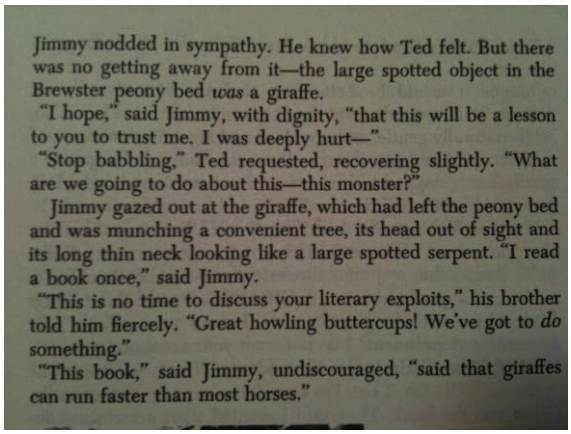
19

"Hello, son!" said the man. "Can I do something for you?"

"I'm just looking around," said Eddie. "My father and mother are out there." Eddie pointed to the front of the store.

In a moment Eddie's eyes fell upon something that interested him very much indeed. On a shelf stood an old carriage lamp. It was rusty and covered with dust.

img6.jpg



Jimmy nodded in sympathy. He knew how Ted felt. But there was no getting away from it—the large spotted object in the Brewster peony bed *was* a giraffe.

"I hope," said Jimmy, with dignity, "that this will be a lesson to you to trust me. I was deeply hurt—"

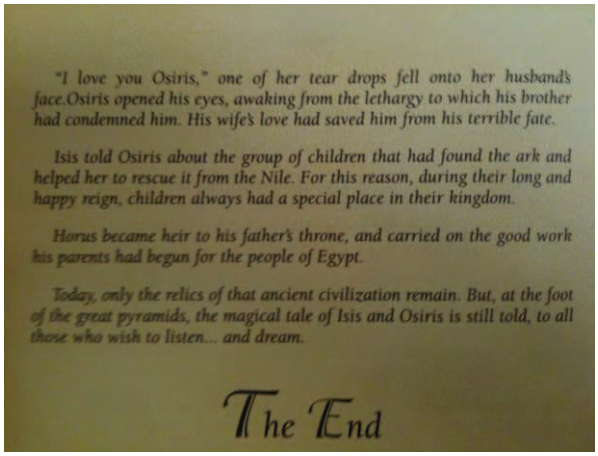
"Stop babbling," Ted requested, recovering slightly. "What are we going to do about this—this monster?"

Jimmy gazed out at the giraffe, which had left the peony bed and was munching a convenient tree, its head out of sight and its long thin neck looking like a large spotted serpent. "I read a book once," said Jimmy.

"This is no time to discuss your literary exploits," his brother told him fiercely. "Great howling buttercups! We've got to *do* something."

"This book," said Jimmy, undiscouraged, "said that giraffes can run faster than most horses."

img7.jpg



"I love you Osiris," one of her tear drops fell onto her husband's face.Osiris opened his eyes, awaking from the lethargy to which his brother had condemned him. His wife's love had saved him from his terrible fate.

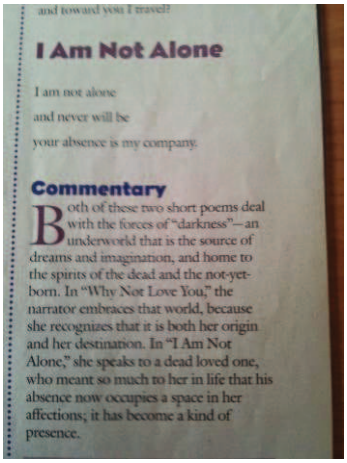
Isis told Osiris about the group of children that had found the ark and helped her to rescue it from the Nile. For this reason, during their long and happy reign, children always had a special place in their kingdom.

Horus became heir to his father's throne, and carried on the good work his parents had begun for the people of Egypt.

Today, only the relics of that ancient civilization remain. But, at the foot of the great pyramids, the magical tale of Isis and Osiris is still told, to all those who wish to listen... and dream.

The End

img8.jpg



and toward you I travel?

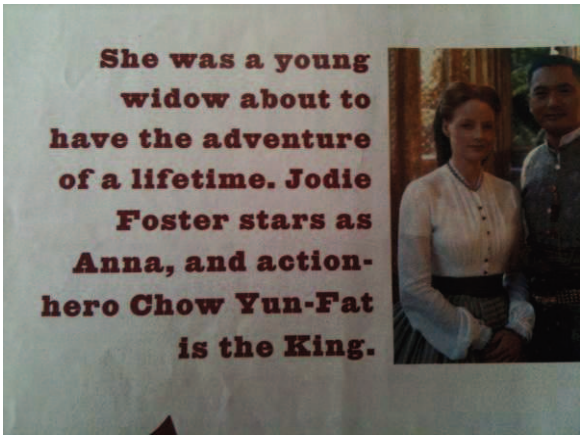
I Am Not Alone

I am not alone
and never will be
your absence is my company.

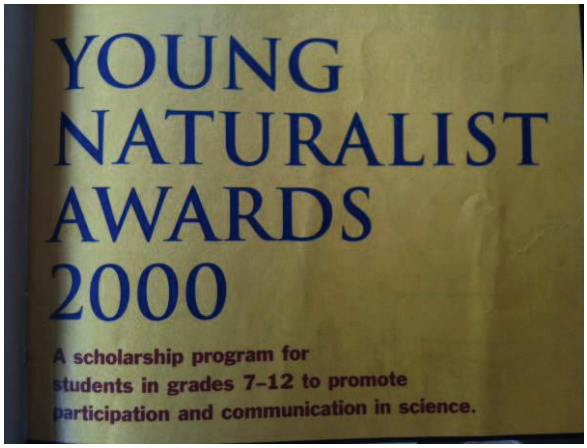
Commentary

Both of these two short poems deal with the forces of "darkness"—an underworld that is the source of dreams and imagination, and home to the spirits of the dead and the not-yet-born. In "Why Not Love You," the narrator embraces that world, because she recognizes that it is both her origin and her destination. In "I Am Not Alone," she speaks to a dead loved one, who meant so much to her in life that his absence now occupies a space in her affections; it has become a kind of presence.

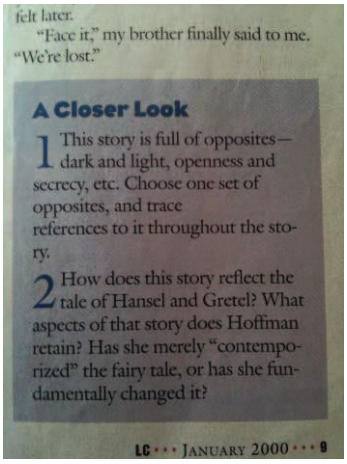
img9.jpg



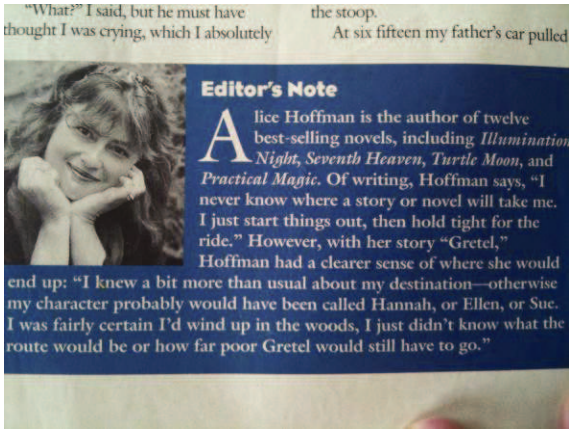
img10.jpg



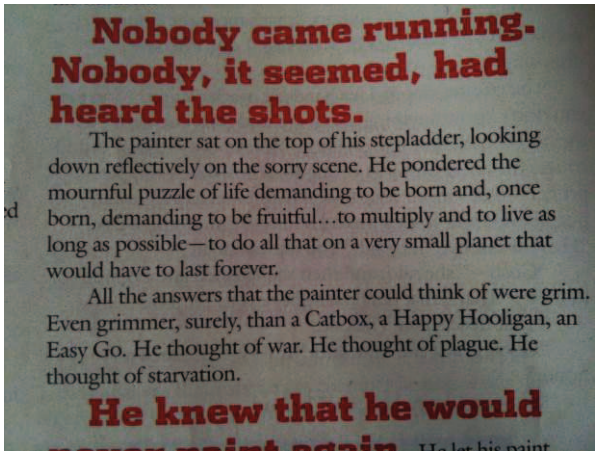
img11.jpg



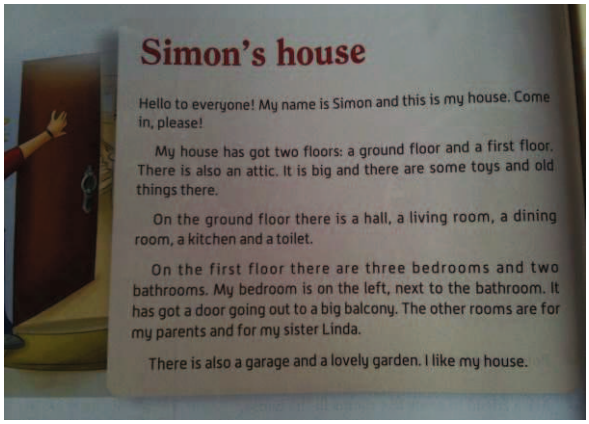
img12.jpg



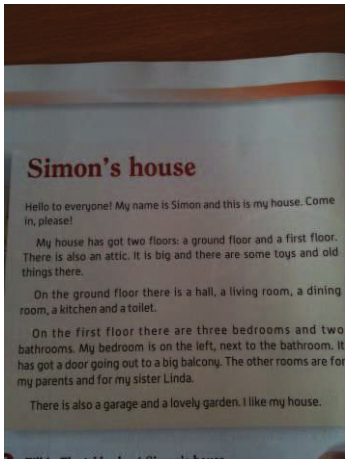
img13.jpg



img14.jpg



img15.jpg



Simon's house

Hello to everyone! My name is Simon and this is my house. Come in, please!

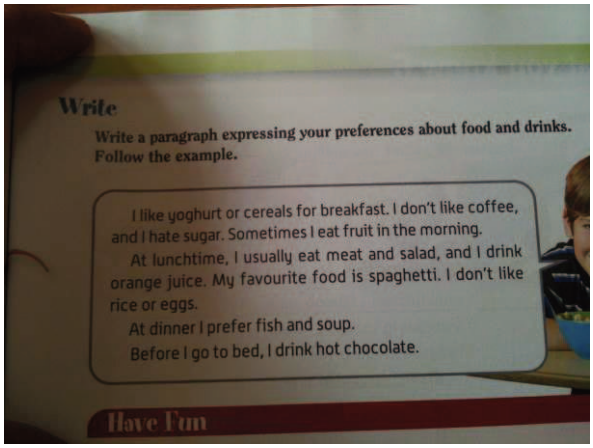
My house has got two floors: a ground floor and a first floor. There is also an attic. It is big and there are some toys and old things there.

On the ground floor there is a hall, a living room, a dining room, a kitchen and a toilet.

On the first floor there are three bedrooms and two bathrooms. My bedroom is on the left, next to the bathroom. It has got a door going out to a big balcony. The other rooms are for my parents and for my sister Linda.

There is also a garage and a lovely garden. I like my house.

img16.jpg



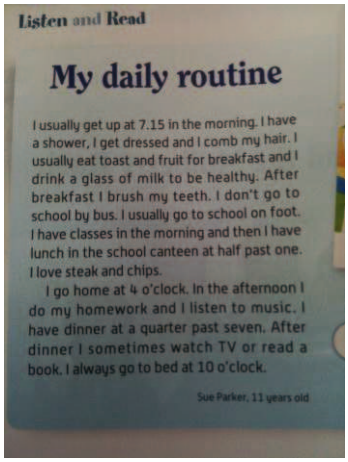
Write

Write a paragraph expressing your preferences about food and drinks. Follow the example.

I like yoghurt or cereals for breakfast. I don't like coffee, and I hate sugar. Sometimes I eat fruit in the morning.
At lunchtime, I usually eat meat and salad, and I drink orange juice. My favourite food is spaghetti. I don't like rice or eggs.
At dinner I prefer fish and soup.
Before I go to bed, I drink hot chocolate.

Have Fun

img17.jpg



Listen and Read

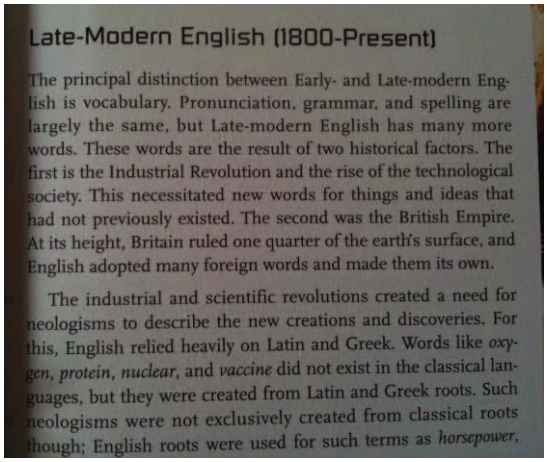
My daily routine

I usually get up at 7.15 in the morning. I have a shower, I get dressed and I comb my hair. I usually eat toast and fruit for breakfast and I drink a glass of milk to be healthy. After breakfast I brush my teeth. I don't go to school by bus. I usually go to school on foot. I have classes in the morning and then I have lunch in the school canteen at half past one. I love steak and chips.

I go home at 4 o'clock. In the afternoon I do my homework and I listen to music. I have dinner at a quarter past seven. After dinner I sometimes watch TV or read a book. I always go to bed at 10 o'clock.

Sue Parker, 11 years old

img18.jpg

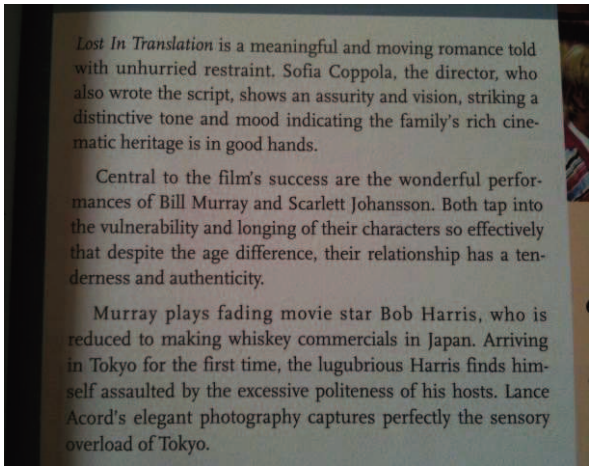


Late-Modern English (1800-Present)

The principal distinction between Early- and Late-modern English is vocabulary. Pronunciation, grammar, and spelling are largely the same, but Late-modern English has many more words. These words are the result of two historical factors. The first is the Industrial Revolution and the rise of the technological society. This necessitated new words for things and ideas that had not previously existed. The second was the British Empire. At its height, Britain ruled one quarter of the earth's surface, and English adopted many foreign words and made them its own.

The industrial and scientific revolutions created a need for neologisms to describe the new creations and discoveries. For this, English relied heavily on Latin and Greek. Words like *oxygen*, *protein*, *nuclear*, and *vaccine* did not exist in the classical languages, but they were created from Latin and Greek roots. Such neologisms were not exclusively created from classical roots though; English roots were used for such terms as *horsepower*,

img19.jpg

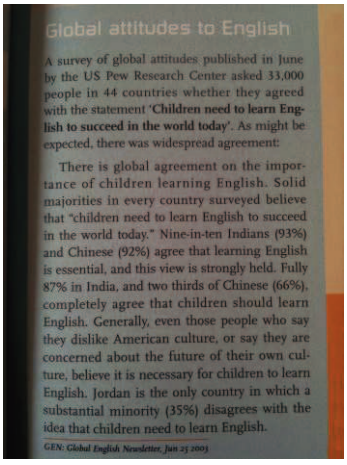


Lost In Translation is a meaningful and moving romance told with unhurried restraint. Sofia Coppola, the director, who also wrote the script, shows an assurity and vision, striking a distinctive tone and mood indicating the family's rich cinematic heritage is in good hands.

Central to the film's success are the wonderful performances of Bill Murray and Scarlett Johansson. Both tap into the vulnerability and longing of their characters so effectively that despite the age difference, their relationship has a tenderness and authenticity.

Murray plays fading movie star Bob Harris, who is reduced to making whiskey commercials in Japan. Arriving in Tokyo for the first time, the lugubrious Harris finds himself assaulted by the excessive politeness of his hosts. Lance Acord's elegant photography captures perfectly the sensory overload of Tokyo.

Img20.jpg



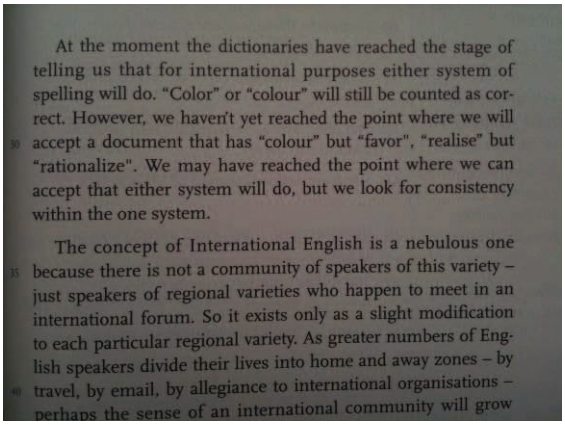
Global attitudes to English

A survey of global attitudes published in June by the US Pew Research Center asked 33,000 people in 44 countries whether they agreed with the statement '**Children need to learn English to succeed in the world today**'. As might be expected, there was widespread agreement:

There is global agreement on the importance of children learning English. Solid majorities in every country surveyed believe that "children need to learn English to succeed in the world today." Nine-in-ten Indians (93%) and Chinese (92%) agree that learning English is essential, and this view is strongly held. Fully 87% in India, and two thirds of Chinese (66%), completely agree that children should learn English. Generally, even those people who say they dislike American culture, or say they are concerned about the future of their own culture, believe it is necessary for children to learn English. Jordan is the only country in which a substantial minority (35%) disagrees with the idea that children need to learn English.

GEN: Global English Newsletter, Jun 23 2003

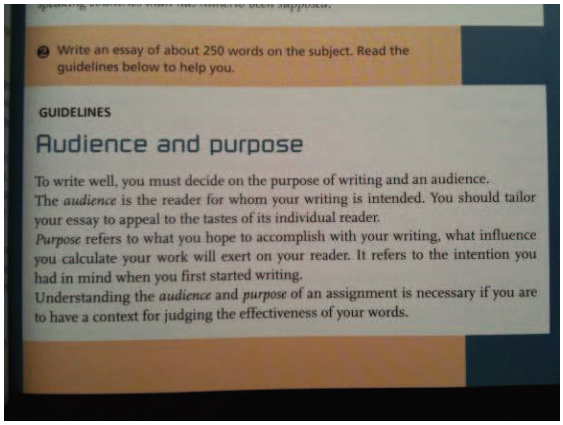
img21.jpg



At the moment the dictionaries have reached the stage of telling us that for international purposes either system of spelling will do. "Color" or "colour" will still be counted as correct. However, we haven't yet reached the point where we will accept a document that has "colour" but "favor", "realise" but "rationalize". We may have reached the point where we can accept that either system will do, but we look for consistency within the one system.

The concept of International English is a nebulous one because there is not a community of speakers of this variety – just speakers of regional varieties who happen to meet in an international forum. So it exists only as a slight modification to each particular regional variety. As greater numbers of English speakers divide their lives into home and away zones – by travel, by email, by allegiance to international organisations – perhaps the sense of an international community will grow

img22.jpg



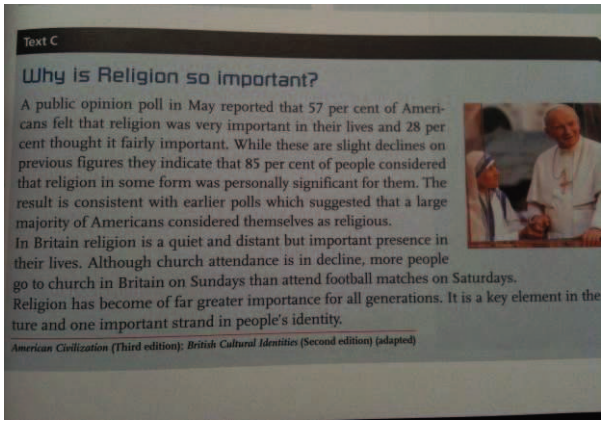
Write an essay of about 250 words on the subject. Read the guidelines below to help you.

GUIDELINES

Audience and purpose

To write well, you must decide on the purpose of writing and an audience.
The *audience* is the reader for whom your writing is intended. You should tailor your essay to appeal to the tastes of its individual reader.
Purpose refers to what you hope to accomplish with your writing, what influence you calculate your work will exert on your reader. It refers to the intention you had in mind when you first started writing.
Understanding the *audience* and *purpose* of an assignment is necessary if you are to have a context for judging the effectiveness of your words.

img23.jpg



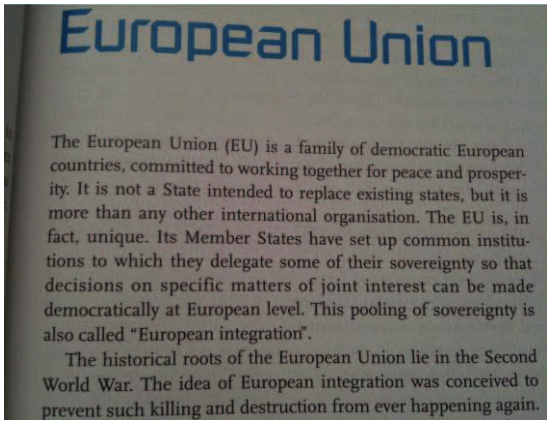
Text C

Why is Religion so important?

A public opinion poll in May reported that 57 per cent of Americans felt that religion was very important in their lives and 28 per cent thought it fairly important. While these are slight declines on previous figures they indicate that 85 per cent of people considered that religion in some form was personally significant for them. The result is consistent with earlier polls which suggested that a large majority of Americans considered themselves as religious.
In Britain religion is a quiet and distant but important presence in their lives. Although church attendance is in decline, more people go to church in Britain on Sundays than attend football matches on Saturdays. Religion has become of far greater importance for all generations. It is a key element in the ture and one important strand in people's identity.

American Civilization (Third edition): *British Cultural Identities* (Second edition) (adapted)

img24.jpg

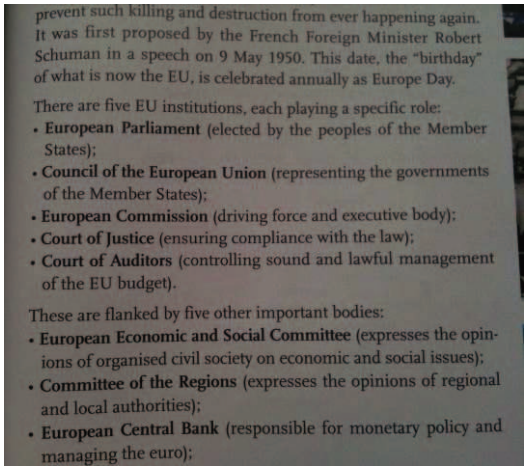


European Union

The European Union (EU) is a family of democratic European countries, committed to working together for peace and prosperity. It is not a State intended to replace existing states, but it is more than any other international organisation. The EU is, in fact, unique. Its Member States have set up common institutions to which they delegate some of their sovereignty so that decisions on specific matters of joint interest can be made democratically at European level. This pooling of sovereignty is also called "European integration".

The historical roots of the European Union lie in the Second World War. The idea of European integration was conceived to prevent such killing and destruction from ever happening again.

img25.jpg



prevent such killing and destruction from ever happening again. It was first proposed by the French Foreign Minister Robert Schuman in a speech on 9 May 1950. This date, the "birthday" of what is now the EU, is celebrated annually as Europe Day.

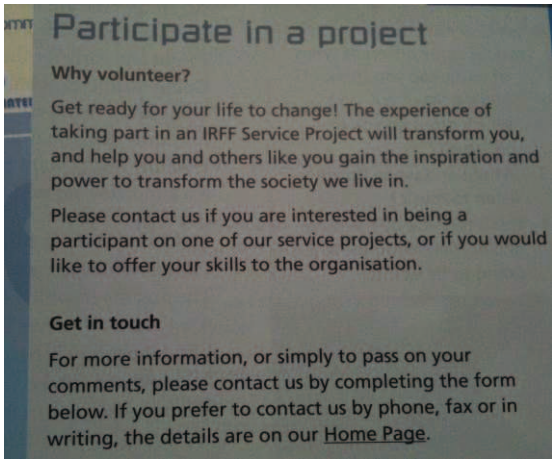
There are five EU institutions, each playing a specific role:

- **European Parliament** (elected by the peoples of the Member States);
- **Council of the European Union** (representing the governments of the Member States);
- **European Commission** (driving force and executive body);
- **Court of Justice** (ensuring compliance with the law);
- **Court of Auditors** (controlling sound and lawful management of the EU budget).

These are flanked by five other important bodies:

- **European Economic and Social Committee** (expresses the opinions of organised civil society on economic and social issues);
- **Committee of the Regions** (expresses the opinions of regional and local authorities);
- **European Central Bank** (responsible for monetary policy and managing the euro);

img26.jpg



Participate in a project

Why volunteer?

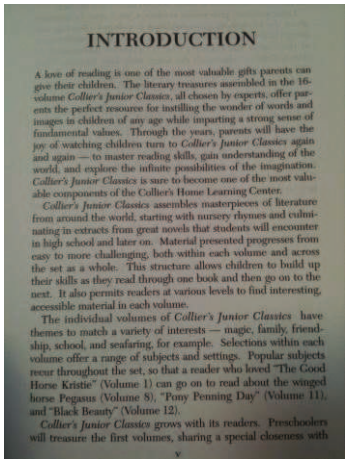
Get ready for your life to change! The experience of taking part in an IRFF Service Project will transform you, and help you and others like you gain the inspiration and power to transform the society we live in.

Please contact us if you are interested in being a participant on one of our service projects, or if you would like to offer your skills to the organisation.

Get in touch

For more information, or simply to pass on your comments, please contact us by completing the form below. If you prefer to contact us by phone, fax or in writing, the details are on our Home Page.

img27.jpg



INTRODUCTION

A love of reading is one of the most valuable gifts parents can give their children. The literary treasures assembled in the 16-volume *Collier's Junior Classics*, all chosen by experts, offer parents the perfect resource for instilling the wonder of words and images in children of any age while imparting a strong sense of fundamental values. Through the years, parents will have the joy of watching children turn to *Collier's Junior Classics* again and again — to master reading skills, gain understanding of the world, and explore the infinite possibilities of the imagination. *Collier's Junior Classics* is sure to become one of the most valuable components of the Collier's Home Learning Center.

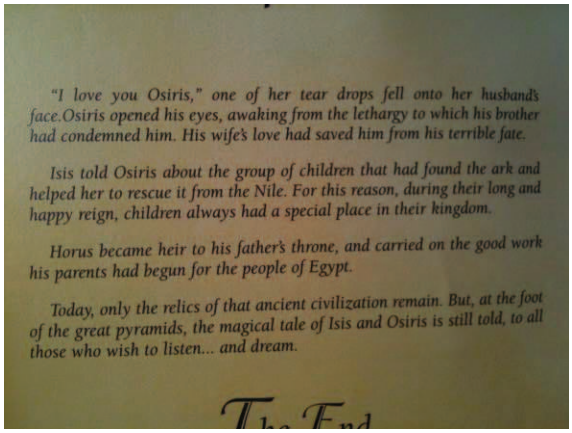
Collier's Junior Classics assembles masterpieces of literature from around the world, starting with nursery rhymes and culminating in extracts from great novels that students will encounter in high school and later on. Material presented progresses from easy to more challenging, both within each volume and across the set as a whole. This structure allows children to build up their skills as they read through one book and then go on to the next. It also permits readers at various levels to find interesting, accessible material in each volume.

The individual volumes of *Collier's Junior Classics* have themes to match a variety of interests — magic, family, friendship, school, and seafaring, for example. Selections within each volume offer a range of subjects and settings. Popular subjects recur throughout the set, so that a reader who loved "The Good Horse Kristie" (Volume 1) can go on to read about the winged horse Pegasus (Volume 8), "Pony Penning Day" (Volume 11), and "Black Beauty" (Volume 12).

Collier's Junior Classics grows with its readers. Preschoolers will treasure the first volumes, sharing a special closeness with

v

img28.jpg



"I love you Osiris," one of her tear drops fell onto her husband's face. Osiris opened his eyes, awaking from the lethargy to which his brother had condemned him. His wife's love had saved him from his terrible fate.

Isis told Osiris about the group of children that had found the ark and helped her to rescue it from the Nile. For this reason, during their long and happy reign, children always had a special place in their kingdom.

Horus became heir to his father's throne, and carried on the good work his parents had begun for the people of Egypt.

Today, only the relics of that ancient civilization remain. But, at the foot of the great pyramids, the magical tale of Isis and Osiris is still told, to all those who wish to listen... and dream.

The End

img29.jpg

7.2 Exemplo prático da aplicação do algoritmo *Levenshtein Distance*

Considerem-se duas strings:

- A: cama
- B: caixa

Através das *strings* indicadas, obtêm-se duas dimensões

- M: tamanho de A, no caso 4
- N: tamanho de B, no caso 5

A Matriz inicial começa então ser preenchida e fica com o seguinte aspeto:

```
Para i= 0 até m fazer
        Matriz[i,0] = i
Fim para

Para j = 0 até n fazer
        Matriz[0,j] = j
Fim para
```

		C	A	I	X	A
	0	**1**	**2**	**3**	**4**	**5**
C	**1**					
A	**2**					
M	**3**					
A	**4**					

O preenchimento da Matriz é feito utilizando o seguinte loop

```
Para i = 1 até m fazer
        Para j = 1 até n fazer
                //determinar custo
                //determinar mínimo
                //preencher posição [i][j] da matriz
        fim para
fim para
```

Determinar Custo	Calcular valor mínimo
Se A[i] = B[j] então	Matriz [i,j] = Mínimo (
Custo = 0	Matriz [i-1, j] + 1 //apagar
Senão	Matriz [i, j-1] +1 //inserir
Custo = 1	Matriz [i-1, j-1] + custo //substituir)

Passo 1

```
Se A[1] = B[1] então
        Custo = 0
Senão
        Custo = 1

CUSTO = 0
```

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0				
A	2					
M	3					
A	4					

Matriz [1,1] = Mínimo (
Matriz [0,1] + 1 = 2
Matriz [1,0] + 1 = 1
Matriz [0,0] + 0 = 0)

Passo 2

Se A[1] = B[2] então
Custo = 0
Senão
Custo = 1

CUSTO = 1

Matriz [1,2] = Mínimo(
Matriz [0,2] + 1 = 3
Matriz [1,1] + 1 = 1
Matriz [0,1] + 1 = 2)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1			
A	2					
M	3					
A	4					

Passo 3

Se A[1] = B[3] então
Custo = 0
Senão
Custo = 1

CUSTO = 1

Matriz [1,3] = Mínimo(
Matriz [0,3] + 1 = 4
Matriz [1,2] + 1 = 2
Matriz [0,2] + 1 = 3)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2		
A	2					
M	3					
A	4					

Passo 4

Se A[1] = B[4] então
Custo = 0
Senão
Custo = 1

CUSTO = 1

Matriz [1,4] = Mínimo(
Matriz [0,4] + 1 = 5
Matriz [1,3] + 1 = 3
Matriz [0,3] + 1 = 4)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	
A	2					
M	3					
A	4					

Passo 5

Se A[1] = B[5] então
    Custo = 0
Senão
    Custo = 1

CUSTO = 1

Matriz [1,5] = Mínimo(
    Matriz [0,5] + 1 = 6
    Matriz [1,4] + 1 = 4
    Matriz [0,4] + 1 = 5)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2					
M	3					
A	4					

Passo 6

Se A[2] = B[1] então
    Custo = 0
Senão
    Custo = 1

CUSTO = 1

Matriz [2,1] = Mínimo(
    Matriz [1,1] + 1 = 1
    Matriz [2,0] + 1 = 3
    Matriz [1,0] + 1 = 2)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1				
M	3					
A	4					

Passo 7

Se A[2] = B[2] então
    Custo = 0
Senão
    Custo = 1

CUSTO = 0

Matriz [2,2] = Mínimo(
    Matriz [1,2] + 1 = 2
    Matriz [2,1] + 1 = 2
    Matriz [1,1] + 0 = 0)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0			
M	3					
A	4					

Passo 8

Se A[2] = B[3] então
Custo = 0
Senão
Custo = 1

CUSTO = 1

Matriz [2,3] = Mínimo(
Matriz [1,3] + 1 = 3
Matriz [2,2] + 1 = 1
Matriz [1,2] + 1 = 2)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1		
M	3					
A	4					

Passo 9

Se A[2] = B[4] então
Custo = 0
Senão
Custo = 1

CUSTO = 0

Matriz [2,4] = Mínimo(
Matriz [1,4] + 1 = 4
Matriz [2,3] + 1 = 2
Matriz [1,3] + 1 = 3)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	
M	3					
A	4					

Passo 10

Se A[2] = B[5] então
Custo = 0
Senão
Custo = 1

CUSTO = 0

Matriz [2,5] = Mínimo(
Matriz [1,5] + 1 = 5
Matriz [2,4] + 1 = 3
Matriz [1,4] + 0 = 3)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3					
A	4					

Passo 11

Se A[3] = B[1] então
Custo = 0
Senão
Custo = 1

CUSTO = 1

Matriz [3,1] = Mínimo(
Matriz [2,1] + 1 = 2
Matriz [3,0] + 1 = 4
Matriz [2,0] + 1 = 3)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2				
A	4					

Passo 12

Se A[3] = B[2] então
Custo = 0
Senão
Custo = 1

CUSTO = 1

Matriz [3,2] = Mínimo(
Matriz [2,2] + 1 = 1
Matriz [3,1] + 1 = 3
Matriz [2,1] + 1 = 2)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1			
A	4					

Passo 13

Se A[3] = B31] então
Custo = 0
Senão
Custo = 1

CUSTO = 1

Matriz [3,3] = Mínimo(
Matriz [2,3] + 1 = 2
Matriz [3,2] + 1 = 2
Matriz [2,2] + 1 = 1)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1	1		
A	4					

Passo 14

Se A[3] = B[4] então
    Custo = 0
Senão
    Custo = 1

CUSTO = 1

Matriz [3,4] = Mínimo(
    Matriz [2,4] + 1 = 3
    Matriz [3,3] + 1 = 2
    Matriz [2,3] + 1 = 2)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1	1	2	
A	4					

Passo 15

Se A[3] = B[5] então
    Custo = 0
Senão
    Custo = 1

CUSTO = 1

Matriz [3,5] = Mínimo(
    Matriz [2,5] + 1 = 4
    Matriz [3,4] + 1 = 3
    Matriz [2,4] + 1 = 3)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1	1	2	3
A	4					

Passo 16

Se A[4] = B[1] então
    Custo = 0
Senão
    Custo = 1

CUSTO = 1

Matriz [4,1] = Mínimo(
    Matriz [3,1] + 1 = 3
    Matriz [4,0] + 1 = 5
    Matriz [3,0] + 1 = 4)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1	1	2	3
A	4	3				

Passo 17

Se A[4] = B[2] então
Custo = 0
Senão
Custo = 1

CUSTO = 0

Matriz [4,2] = Mínimo(
Matriz [3,2] + 1 = 2
Matriz [4,1] + 1 = 4
Matriz [3,1] + 0 = 2)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1	1	2	3
A	4	3	2			

Passo 18

Se A[4] = B[3] então
Custo = 0
Senão
Custo = 1

CUSTO = 1

Matriz [4,3] = Mínimo(
Matriz [3,3] + 1 = 2
Matriz [4,2] + 1 = 3
Matriz [3,2] + 1 = 2)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1	1	2	3
A	4	3	2	2		

Passo 19

Se A[4] = B[4] então
Custo = 0
Senão
Custo = 1

CUSTO = 1

Matriz [4,4] = Mínimo(
Matriz [3,4] + 1 = 3
Matriz [4,3] + 1 = 3
Matriz [3,3] + 1 = 2)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1	1	2	3
A	4	3	2	2	2	

Passo 20

Se A[4] = B[5] então
    Custo = 0
Senão
    Custo = 1

CUSTO = 0

Matriz [4,5] = Mínimo(
    Matriz [3,5] + 1 = 4
    Matriz [4,4] + 1 = 3
    Matriz [3,4] + 0 = 2)

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1	1	2	3
A	4	3	2	2	2	2

Passo Final

Preenchida a matriz, obtém-se o valor da distância de Levenshtein. Verifica-se que o número de operações para transformara string "Cama" em "Caixa" é 2, a que corresponde uma substituição e uma inserção.

		C	A	I	X	A
	0	1	2	3	4	5
C	1	0	1	2	3	4
A	2	1	0	1	2	3
M	3	2	1	1	2	3
A	4	3	2	2	2	2

7.3 Paper a ser submetido para a conferência HCist 2014

Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 00 (2014) 000–000

Procedia Technology

www.elsevier.com/locate/procedia

CENTERIS 2014 - Conference on ENTERprise Information Systems / ProjMAN 2014 - International Conference on Project MANagement / HCIST 2014 - International Conference on Health and Social Care Information Systems and Technologies

Camera Reading For Blind People

Roberto Neto[a], Nuno Fonseca[b]

[a] Polytechnic Institute of Leiria, Leiria 2411-901 Leiria, PORTUGAL
[b] Polytechnic Institute of Leiria, Leiria 2411-901 Leiria, PORTUGAL

Abstract

A ausência de visão torna a vida de um invisual bastante difícil. No entanto, a utilização da tecnologia pode melhorar ligeiramente pequenos aspectos do dia-a-dia. Nesse contexto, o trabalho que se apresenta tem como objetivo a descrição do processo de desenvolvimento de uma aplicação para cegos. O projeto chama-se *Camera Blind For Blind People*, e tem como propósito final a criação de uma aplicação que permita a um utilizador cego utilizar a câmara de um dispositivo móvel para obter a leitura de um texto contido na imagem captada, quer seja texto impresso em papel, cartazes, sinais, avisos, etc.
Pretende-se com este projeto conceber o protótipo de uma aplicação para iOS, construída a partir da utilização conjunta e integrada de *frameworks* de reconhecimento óptico de caracteres (OCR) e de *frameworks* de sintetização de voz (TTS), que possibilite ao utilizador, recorrendo à câmara de um dispositivo móvel, captar uma imagem e obter a leitura do texto que exista nessa mesma imagem.

Peer-review under responsibility of the Organizing Committees of CENTERIS/ProjMAN/HCIST 2014

Keywords: Camera Reading, Blind People, OCR, TTS, iOS

1. Introdução

Os indivíduos cegos não têm capacidade para realizar tarefas visuais (ler, reconhecer rostos). A leitura de textos ou avisos pode ser feita apenas se os mesmos estiverem no sistema de leitura braille, ou através da audição de livros, se estes estiverem em formato áudio ou em formato digital (usando sistemas text-to-speech).

A generalidade das obras publicadas em papel não contemplam a versão braille ou a versão áudio, e as versões digitais ainda são uma clara minoria. Por outro lado, os avisos existentes nas paredes ou nos sinais que nos rodeiam também não possibilitam a leitura por parte de um cego. Assim, o desenvolvimento de uma aplicação que possa

Peer-review under responsibility of the Organizing Committees of CENTERIS/ProjMAN/HCIST 2014.

realizar a leitura oral de textos, quer estejam numa parede, numa folha de papel ou noutro suporte escrito, assume grande potencialidade e utilidade.

A tecnologia de reconhecimento óptico de caracteres (OCR) possibilita o reconhecimento de textos escritos em suportes físicos variados. Esta tecnologia tem sido largamente utilizada na digitalização de documentos ou fotografias, convertendo-os em cópias electrónicas que podem ser editadas, onde podemos realizar pesquisas, reproduzir o seu conteúdo e transportá-lo com facilidade [5].

Por seu lado, a tecnologia de síntese de voz (TTS) possibilita que um texto em formato digital (escrito diretamente pelo utilizador, existente numa caixa de texto ou na descrição de um botão), seja sintetizado em voz humana e reproduzido num sistema áudio. O objetivo do TTS é a conversão automática de frases, sem restrições, num discurso falado em linguagem natural, que se assemelhe à forma falada, do mesmo texto, por um indivíduo nativo da língua. Esta tecnologia tem tido avanços significativos ao longo da última década, com muitos sistemas a serem capazes de gerar um discurso sintético muito próximo da voz natural. A pesquisa na área da síntese de voz tem crescido, fruto da sua crescente importância em muitas novas aplicações [15].

Este paper descreve o processo de elaboração de um protótipo de uma aplicação para dispositivos móveis (iOS), que poderá ser utilizada num iPhone/iPod Touch/iPad, que permita a um utilizador invisual utilizar a câmara do dispositivo e obter a leitura de um texto escrito existente na imagem captada.

A realização do trabalho pressupõe a utilização de tecnologia OCR e TTS já existentes, utilizando frameworks já existentes, combinando-as para que em conjunto possam fornecer os resultados pretendidos.

Este paper está organizado na seguinte estrutura: a secção 2 são apresentados conceitos relativamente ao estado da arte no que respeita a tecnologias OCR e a TTS; na secção 3 é apresentada a metodologia de desenvolvimento do projeto; na secção 4 são apresentados os passos para optimizar o sistema, e finalmente são apresentadas alguma conclusões e trabalho futuro a realizar.

2. Visão Geral e Estado da Arte

Apresenta-se de seguida uma introdução às tecnologias OCR e TTS, utilizadas neste projeto, assim como são apresentados estudos e trabalhos desenvolvidos no âmbito da temática abordada neste projeto.

2.1. Visão Geral da tecnologia utilizada

2.1.1. OCR – Optical Character Recognition

O reconhecimento ótico de caracteres, normalmente denominado pela sigla OCR, consiste no processo de reconhecimento e conversão automática de carateres existentes na imagem de um suporte escrito para o formato de texto, que pode ser posteriormente utilizado em diversas aplicações. O OCR tem sido amplamente estudado, verificando-se avanços em relação ao desempenho e à precisão dos resultados obtidos [7].

O processo de reconhecimento ótico de caracteres pode ser sintetizado como um processo que decorre num conjunto de passos [4], seguidamente indicados:

- Aquisição ótica da imagem
- Localização e segmentação
- Pré-processamento
- Extração de características
- Classificação
- Pós-processamento

O texto final é então convertido num documento com o formato pretendido (rtf, txt, pdf, etc.).

2.1.2. TTS – Text-to-Speech

A síntese de voz, definida como TTS (sigla para Text-To-Speech) é um sistema computacional que deve ser capaz de ler em voz alta qualquer texto, independentemente da sua origem [3].

A utilização do TTS é a tentativa de produzir a voz humana de forma artificial.

A síntese de voz é um processo complexo, sendo necessários algoritmos complexos para produzir um resultado inteligível e natural.

A sintetização TTS faz uso de técnicas do Processamento de Linguagem Natural. Sendo o texto a ser sintetizado a primeira entrada do sistema, deverá ser o primeiro a ser processado.

Para gerar a voz sintetizada, segundo [2] existem quatro principais técnicas, seguidamente indicadas:

- Síntese articulatória
- Síntese por formantes
- Síntese por concatenação
- Síntese baseada nos modelos ocultos de Markov

As principais técnicas de síntese acima indicadas, são os métodos mais utilizados no estudo e no desenvolvimento de sistemas de síntese de voz. Contudo, uma forma de tirar partido das vantagens inerentes a cada técnica é utilizar um híbrido das várias técnicas no desenvolvimento de futuros sistemas de síntese de voz.

A qualidade de uma síntese de voz pode ser determinada pela naturalidade e a inteligibilidade, sendo a naturalidade a característica que descreve o quão próximo o som obtido pelo TTS está da voz humana e a inteligibilidade refere-se à facilidade com que o som é compreendido em situações complexas [3,8].

2.2. Estado da Arte

Existem trabalhos e estudos que versam sobre sobre a temática da captação de imagem com a câmara de um dispositivo móvel, a obtenção do texto existente nessa imagem para diversas utilizações, entre elas, a sua sintetização através do TTS. No entanto, esses trabalhos têm sido fundamentalmente pensados para sistemas Android.

Em The Phone Reader [1], é feita a abordagem a um sistema desenvolvido para o sistema Android, que permite reconhecer o texto existente numa foto, utilizando a OCR Tesseract, e através do TTS procede à leitura do texto traduzido na língua que for selecionada. O processamento OCR e a tradução são feitos num servidor externo, o que implica uma ligação de dados para a comunicação entre o servidor e o dispositivo.

Na publicação Optical Character Recognition [10], é apresentada uma aplicação para dispositivos móveis Android, que realiza o reconhecimento do texto existente em fotos através do OCR Tesseract, e após o seu reconhecimento é feita a sintetização de voz. Nesta aplicação, todo o processamento é feito localmente no dispositivo móvel, não havendo lugar a qualquer pré-processamento ou reconhecimento de limites das imagens, ficando estes definidos para trabalho futuro.

Existem disponíveis na App Store da Apple algumas aplicações que realizam o reconhecimento de textos através de OCR textos e realizam a sua leitura. Em todos os casos as aplicações recorrem à funcionalidade VoiceOver para obter a leitura dos textos.

De seguida, apresentam-se exemplos das referidas aplicações:

- SayText [12], é uma aplicação que permite tirar fotografias a textos e após fazer o reconhecimento do texto, permite o envio do mesmo por email ou obter a leitura do texto, recorrendo à funcionalidade do sistema VoiceOver. É uma aplicação gratuita.
- Talking Camera Pro - for visually impaired/blind [14], é mais uma aplicação que permite o reconhecimento de textos através de OCR e a sua leitura através da funcionalidade VoiceOver. Esta aplicação tem um custo de 1,79€.
- Prizmo - Escaneamento, OCR & Fala [11], é uma aplicação permite reconhecer documentos de texto, cartões de visita e imagens, podendo o resultado ser exportado como PDF/Texto, vCard ou JPEG/PNG. A aplicação permite o processamento das imagens obtidas, podendo o utilizador rodar, recortar, corrigir a perspectiva e limpar a

imagem (remoção de iluminação desigual, textura do papel...). Através da funcionalidade VoiceOver é possível a leitura dos textos. Esta aplicação tem um custo de 8,99€.

Não obstante existirem estudos aplicações que abordam a temática, verifica-se que as soluções existentes apresentam limitações diversas, que se prendem com a performance, com o custo elevado ou pelo facto de serem e estudadas para o sistema Android.

O projeto Camera Reading for Blind People, reveste-se assim de importância e pertinência, uma vez que vem lançar as bases para o desenvolvimento de uma aplicação para o sistema iOS que possa servir de forma mais eficaz, e com menos custos, as necessidades de invisuais.

3. Metodologia

Nesta secção é apresentada a metodologia seguida na prossecução do projeto, desde a captação de imagem com a câmara, passando pela implementação das ferramentas OCR e TTS.

A arquitetura geral do sistema é representada pela figura seguinte:

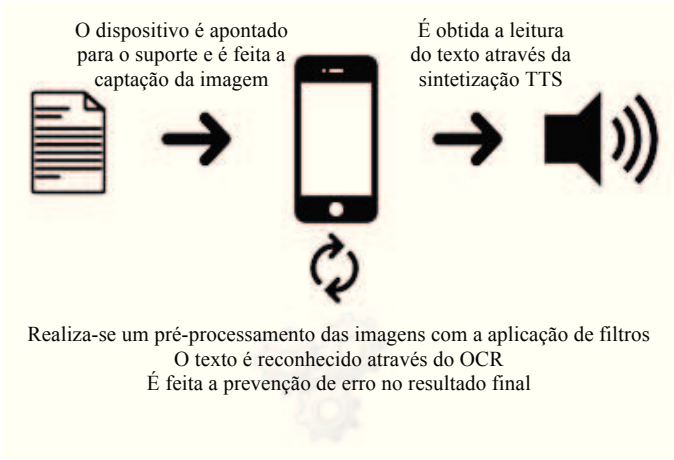


Figura 1 - Arquitetura do Sistema

3.1. Captação de imagem com a câmara

A câmara do dispositivo móvel é fundamental para a utilização da aplicação, uma vez que será o meio através do qual o utilizador irá obter a imagem dos suportes que contenham texto a ser reconhecido e sintetizado.

Uma vez que os utilizadores alvo da aplicação são pessoas cegas, foi pensada uma forma de a captação das imagens ser feita em tempo real, sem que tenha que ser feita a captação de uma foto que de seguida teria que ser aberta para realizar o reconhecimento OCR.

3.2. Implementação de ferramentas OCR

Após obtida a imagem, o próximo passo é o reconhecimento ótico do texto existente na mesma, através da utilização de uma framework OCR.

O reconhecimento óptico de caracteres, normalmente conhecido como OCR, é a interpretação de imagens que tenham textos, e que depois de digitalizadas, o texto nelas contido pode ser reconhecido, editado e utilizado.

Verificou-se que existem algumas frameworks disponíveis para iOS, tendo sido três dessas selecionadas para a realização de testes. Assim, foram testadas a framework Open-Source (código aberto) Tesseract e as frameworks comerciais Leadtools e Abby. A framework Tesseract não apresenta custos de utilização e as restantes soluções apresentam custos, sendo estes no caso da solução Leadtools (pode ser consultado online em

http://www.leadtools.com/resources/weborder.htm) de 995$ e no caso da framework ABBY (pode ser consultado online em http://www.abbyy-developers.eu/en:business:pricing) de 4.900,00€.

De referir que a utilização das *frameworks* comerciais requereu o contacto com as respetivas empresas e o preenchimento de formulários para obtenção de licenças de três meses para avaliação dos produtos.

3.3. Escolha da ferramenta OCR

Para comparar a eficácia do reconhecimento de cada uma das frameworks, foi recolhido um conjunto de trinta fotos de textos, com formatos, tamanhos e layouts diferenciados. Procurou-se obter um conjunto de imagens que representassem uma variedade de situações o mais abrangentes possível, com textos em formatos, tamanhos e cores diferentes, alinhamentos diferentes para as imagens e diferentes exposições à luz no momento da captação das imagens, na tentativa de obter a mais fiel representação de situações reais que seguramente acontecerão aquando da utilização da aplicação.

Seguidamente apresentam-se algumas imagens utilizadas nos testes:

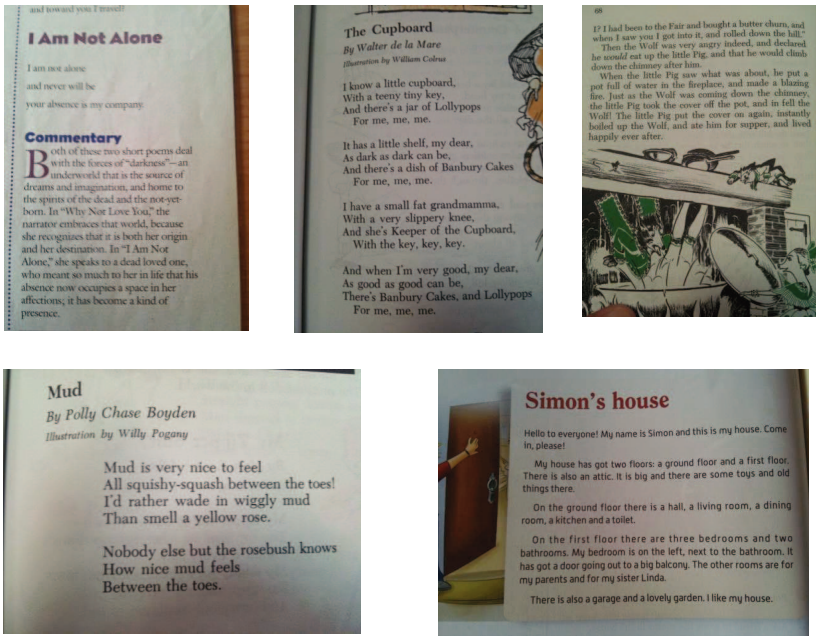


Figura 2 – Exemplos de imagens utilizadas nos testes

De seguida, o texto de cada uma das fotos foi transcrito para ficheiros de texto individuais, no sentido de poder realizar a comparação entre o texto obtido após reconhecimento ótico da imagem e o texto original.

3.4. Levenshtein distance

Para obter valores de medição que possam ser utilizados na comparação entre os textos, o reconhecido oticamente e o original, foi desenvolvida uma função que reproduz o algoritmo de comparação de strings Levenshtein distance.

O algoritmo Levenshtein distance [9] foi criado pelo cientista russo Vladimir Levenshtein, em 1965. O princípio básico do algoritmo é a medição da similaridade entre duas strings. Isto é feito através do cálculo do número de operações básicas, ou alterações necessárias para tornar as duas strings iguais. As alterações podem consistir numa substituição, apagar ou inserção de um carácter. A distância de Levenshtein d(p,t) (p e t representam duas strings) entre duas strings é o menor número de operações necessárias para tornar p = t. Por exemplo, se p = "moose" e t =

"moody" a distância Levenshtein entre p e t será d(p,t) = 2, uma vez que as strings podem tornar-se iguais utilizando duas substituições (troca da letra s pela letra d e troca da letra e pela letra y).

Tomando outro exemplo, a distância Levenshtein entre as palavras inglesas "kitten" (gato) e "sitting" (sentando-se) é 3, já que com apenas 3 edições conseguimos transformar uma palavra na outra, e não há maneira de o fazer com menos de três edições [13]:

- kitten
- sitten (substituição de 'k' por 's')
- sittin (substituição de 'e' por 'i')
- sitting (inserção de 'g' no final)

Os valores obtidos foram registados numa folha de cálculo para que fosse possível analisá-los de forma estatística. Assim, registaram-se os nomes das imagens, o total de caracteres dos textos nos textos originais dessas mesmas imagens, a distância obtida entre a leitura do texto original e o texto reconhecido e a percentagem de erros, calculada na divisão entre a distância obtida e o total de carateres do texto.

Foi então calculada valor da mediana dos valores das distâncias obtidas. Ao contrário da média, a mediana não é influenciada por valores extremos, visto que é uma medida essencialmente vinculada à posição que ocupa no conjunto ordenado. Assim, se algum valor for demasiado grande ou pequeno – valores extremos –, estes não afectarão o cálculo da mediana, já que não alterarão a ordem.

Por exemplo, se analisarmos dois conjuntos de dados A e B, sendo A = { 1, 2, 1000 } e B = { 1, 2, 10 }, em ambos, a mediana é 2, ou seja, ao se trocar o valor 10 (conjunto B) por 1000 (conjunto A), o valor da mediana não se altera.

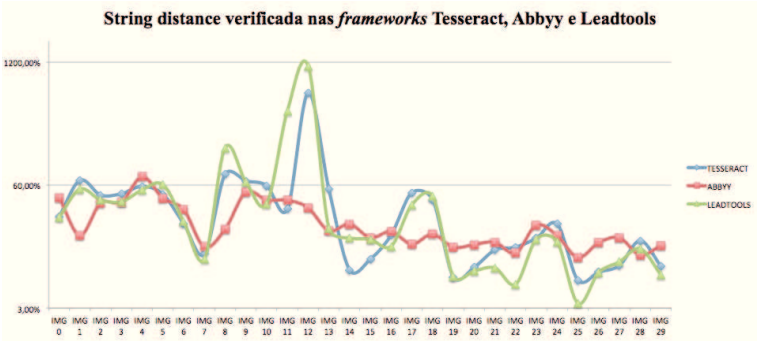


Figura 3 – Comparativo da string distance das imagens testadas nas frameworks Tesseract, Abbyy e Leadtools

Apresenta-se de seguida o comparativo do resultado da mediana das distâncias verificadas em cada uma das frameworks analisadas.

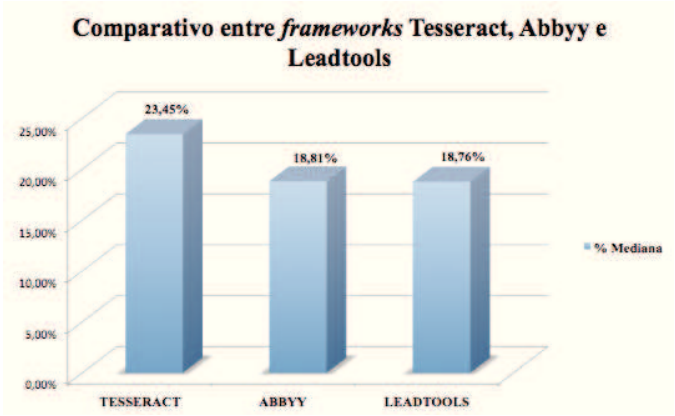


Figura 4 – Comparativo do valor da mediana da string distance das imagens testadas nas frameworks Tesseract, Abbyy e Leadtools

Como pode ser verificado pela análise dos resultados, as frameworks comerciais apresentam melhores resultados que a framework gratuita (OCR Tesseract – 23,45%, OCR ABBY – 18,81% e OCR Leadtools 18,76%).

Neste projeto será utilizada a framework Tesseract, uma vez tratar-se de uma solução sem custos.

3.5. Implementação da ferramenta Text-To-Speech

Não obstante o facto de existirem diversas frameworks TTS, desde a atualização da versão 7 do sistema iOS, foi incluída na framework AVFoundation uma nova classe, a classe AVSpeechSynthesizer que será utilizada, uma vez que permite obter a sintetização de voz pretendida para a aplicação, sem necessidade de recorrer a frameworks externas.

4. Optimização do Sistema

Após escolhido a framework OCR Tesseract, procedeu-se a um conjunto de testes no sentido de procurar obter melhores resultados no reconhecimento das imagens, e maior eficácia nas utilizações da aplicação.

4.1. Aplicação de filtros

Tendo em conta que as características da imagem utilizada no processo de reconhecimento ótico tem impacto no resultado, nomeadamente na qualidade da imagem, na luminosidade, no contraste de cores, na focagem entre outras características, procedeu-se à tentativa de realizar alterações às imagens captadas pelo dispositivo, no sentido de melhorar o resultado obtido na sua leitura OCR.

Assim, procedeu-se à utilização de filtros no sentido de procurar realizar um pré-processamento das imagens que resulte na obtenção de melhores resultados.

Procedeu-se à realização de um conjunto de testes em que foi obtido a mediana do valor percentual da distância de Levenshtein, resultante da diferença entre o texto original e os textos reconhecidos, após aplicação de cada um dos referidos filtros, no conjunto de 30 imagens recolhidas para testes.

De referir que nos testes para cada filtro, os valores dos seus diversos parâmetros foram sendo alterados em ciclo, procurando obter os melhores resultados. Foram desta forma apurados os valores para os parâmetros de cada filtro que permitiram a obtenção de melhores resultados.

Foram testados diversos filtros disponibilizados pela framework Core Image do iOS [6], nomeadamente os filtros CIToneCurve (ajusta a resposta dos canais RGB), CIHueAdjust (altera a cor da generalidade dos pixéis) CIGammaAdjust (ajusta o brilho intermédio), CIColorControls (ajusta valores de saturação, brilho e contraste), CIColorMonochrome (remapeia as cores de uma imagem para que essas cores fiquem dentro da tonalidade de apenas uma cor), CIExposureAdjust (ajusta a exposição da imagem de forma idêntica ao de uma máquina fotográfica).

Após realizados diversos testes verificou-se que a combinação dos filtros CIColorControls e CIColorMonochrome originou os melhores resultados, pelo que esta combinação de filtros será sempre aplicada às imagens captadas pelo dispositivo, imediatamente antes do processo de reconhecimento OCR.

O filtro CIColorControls permite ajustar a saturação, o brilho e o contraste numa imagem. Recebe como parâmetros uma imagem do tipo CIImage, e podem ser alterados os valores para os atributos atrás referidos. Utilizaram-se os valores de 1.00 para a saturação, 0.50 para o brilho e 2.14 para o contraste.

O filtro CIColorMonochrome permite remapear as cores de uma imagem para que essas cores fiquem dentro da tonalidade de apenas uma cor. Recebe como parâmetros uma imagem do tipo CIImage, e podem ser alterados os valores para a cor de entrada e para a intensidade. Utilizaram-se os valores 0.77 para a intensidade e a cor branca.

Os resultados obtidos após aplicação desta combinação de filtros foram os seguintes:

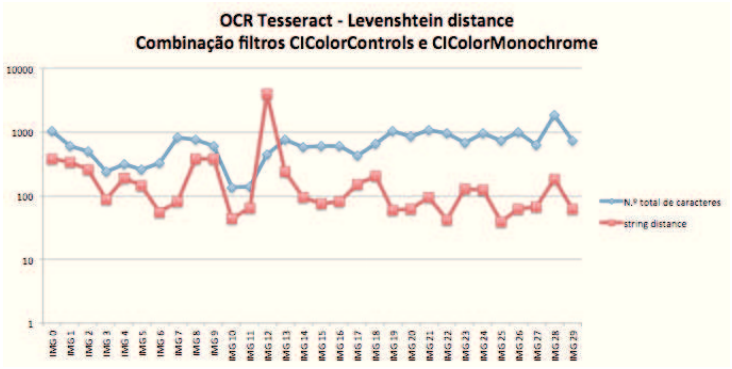


Figura 5 – Valores da string distance obtidas após aplicação da combinação de filtros CIColorControls e CIColorMonochrome

Após aplicação desta combinação de filtros, verificou-se uma substancial melhoria nos resultados, com o valor mediano da distância entre o texto original e o reconhecido a diminuir de 23,45% para 17,83%.

4.2. Prevenção de erro no resultado final

Durante a realização de testes da aplicação, nomeadamente quando se utiliza a captação de imagens com a máquina fotográfica do dispositivo, verificou-se que em determinadas leituras, o resultado do reconhecimento OCR era um conjunto de caracteres sem sentido, e que em nada correspondiam ao texto constante das imagens. Tal resultado era originado pela deficiente qualidade da imagem captada, ou por erros de reconhecimento do OCR.

No entanto, uma vez que esse conjunto de caracteres correspondiam a texto era realizada a sintetização oral, o que revelava-se enganador e mesmo desagradável ao utilizador.

Depois de realizados vários testes, e observados os textos resultantes do reconhecimento óptico, verificou-se que o padrão normal das strings nos casos de leituras incorretas apresentava vários caracteres especiais, seguidamente indicados:

( ) ; * ~ » “ . =

O conjunto de carateres indicado baseou-se na observação de vários exemplos testados, podendo naturalmente em versões futuras ser alterados, ou até serem incluídos novos caracteres a este conjunto, bastando para tal incluí-los na função.

Para evitar tal situação procedeu-se à utilização de expressões regulares para testar se o texto devolvido pelo OCR correspondia a um texto legível ou se devolvia muitas ocorrências dos caracteres especiais indicados. Assim, a função criada recebe o texto reconhecido pelo OCR, conta o total de caracteres do texto, e conta o total de ocorrências dos carateres especiais supracitados, calculando um valor percentual desses carateres especiais em relação ao total de caracteres do texto, denominado por percentagem de erro, obtido através da fórmula seguinte:

$$Percentagem\ de\ erro = \frac{Ocorrência\ de\ caracteres\ especiais \times 100}{Total\ de\ caracteres}$$

Figura 6 – Fórmula de cálculo da percentagem de erro

Se o valor da percentagem de erro ultrapassar o valor definido, que pode ser configurado pelo utilizador, ao invés de a aplicação realizar a sintetização do texto, apresenta uma mensagem de erro no processamento, solicitando ao utilizador que repita o processo de captura da imagem.

No sentido de aferir o valor percentual inicial para a aplicação, e definir o intervalo dos valores a serem alterados, procedeu-se a um conjunto de testes utilizando o conjunto de imagens já utilizadas nos restantes testes, sendo definidos os valores percentuais de caracteres especiais permitidos, e nos casos em que esses valores fossem

ultrapassados, as situações eram consideradas como erro de processamento. Os valores obtidos nos testes apresentam-se de seguida:

Tabela 1. Valores da percentagem da mediana da distância após aplicação da função de prevenção de erro

Percentagem de erro	Percentagem do valor da mediana da distância	Quantidade de imagens descartadas
10%	16,62%	1
9.5%	16,62%	1
9%	16,62%	1
8.5%	16,62%	1
8%	16,62%	1
7.5%	16,30%	2
7%	16,30%	2
6.5%	15,99%	3
6%	15,99%	3
5.5%	15,99%	3
5%	13,43%	5
4.5%	13,15%	6
4%	12,87%	7
3.5%	12,87%	7
3%	12,87%	9
2.5%	9,91%	16
2%	8,94%	20
1.5%	6,60%	24
1%	25,94%	28

A representação gráfica dos valores indicados apresneta-se de seguida:

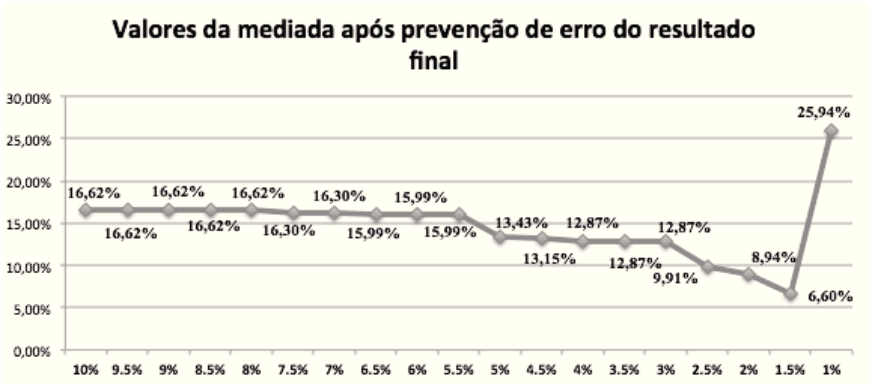


Figura 7 – Valores da mediana da distância após aplicação da função de prevenção de erro

Através da análise dos resultados, pode verificar-se que quanto menor for a percentagem de caracteres especiais permitidos, melhor são os resultados do valor percentual da mediana do erro obtido. Tal decorre do facto de as

imagens com piores resultados no reconhecimento OCR serem descartadas. A exceção na trajetória descendente do valor percentual da mediana verifica-se no valor de 1%, uma vez que neste valor apenas são consideradas como válidas a leitura de duas imagens, não podendo esse valor ser considerado como um valor de referência para concluir da sua utilização.

Verificou-se que para percentagens inferiores a 3% a quantidade de imagens descartadas é muito elevada, ultrapassando 50% do total das 30 imagens testadas. A utilização de uma percentagem tão baixa poderá tornar a aplicação pouco eficaz, apresentando muitas vezes a mensagem de erro de reconhecimento.

O valor por defeito na primeira utilização da aplicação será de 5%, uma vez que, após realizados os testes utilizando este valor como referência, verificou-se que, embora sendo descartadas 5 imagens do conjunto das 30 testadas, a que corresponde uma percentagem de aproximadamente 16,7%, o valor da percentagem da mediana da distância obtido é de 13,43%, a que corresponde uma melhoria substancial face ao valor verificado sem a prevenção do erro, 17,83%.

Naturalmente o utilizador poderá optar por alterar o valor definido para a percentagem, podendo aumentar a mesma, e obter mais resultados incorretos e menos mensagens de erro, ou diminuir a percentagem, e obter resultados mais eficazes mas maior número de situações de erro de reconhecimento.

5. Conclusão

Este paper apresenta o desenvolvimento do projeto Camera Reading for Blind People, passando pelo conhecimento de frameworks OCR e TTS, e a sua utilização, quer de forma isolada quer em conjunto, até a criação de uma aplicação, que foi sendo melhorada e afinada ao longo do projeto, após ter sido sujeita a um conjunto de testes e optimizações.

Foi realizado um trabalho de pesquisa relativamente às tecnologias OCR e TTS que foram utilizadas no desenvolvimento aplicação, procurando conhecer os métodos que estão por detrás das mesmas, e conhecer com maior profundidade os mecanismos que realizam o reconhecimento óptico de caracteres em imagens e a sintetização de voz de textos.

O projeto consistiu na construção de uma aplicação constituída por várias partes, integrando o sistema de captação de imagem do dispositivo móvel, que é utilizado por uma framework OCR para reconhecimento do seu texto, que é seguidamente sintetizado através de um processo de TTS.

As optimizações levadas a cabo para melhoria dos resultados tornaram a aplicação mais eficiente e mais capaz de responder ao repto lançado pelo tema do projeto: uma câmara que leia textos para cegos.

Uma vez verificado que a utilização da imagem captada pelo dispositivo apresentava resultados pouco satisfatórios no que concerne ao reconhecimento do texto nela constante, procedeu-se a um pré-processamento da imagem antes desta ser submetida ao reconhecimento ótico. Após analisados diversas possibilidades de tratamento das imagens, a opção recaiu na utilização de uma combinação de filtros (CIColorControls e CIColorMonochrome) que demonstraram melhorar substancialmente os resultados.

Outra das optimizações a que o projeto foi submetido foi uma análise do resultado do reconhecimento óptico, procurando aferir da sua utilidade enquanto resultado a apresentar. Verificou-se que em diversas situações, fruto de leituras incorretas do OCR, o resultado final representava um conjunto de caracteres sem sentido, mas como representavam texto, o TTS realizava a sua sintetização, o que se tornava incómodo ao utilizador. Assim, foi desenvolvida uma função, que recorrendo a expressões regulares, analisa o resultado do reconhecimento OCR, e nos casos em que esse resultado apresente uma percentagem de caracteres especiais relevante, o sistema indica que houve erro de leitura pedindo ao utilizador que repita o processo.

Embora a aplicação final não represente ainda uma solução plenamente utilizável e totalmente fiável para os invisuais no dia a dia, o programa desenvolvido e os resultados obtidos podem considerar-se já interessantes, verificando-se que em algumas imagens, se reunidas as condições ideias, esses resultados são mesmo bastante interessantes, decorrendo um reconhecimento e consequente leitura muito eficazes e consistentes com o texto original.

O resultado final alcançado não é perfeito, uma vez que apresenta algumas lacunas no que diz respeito ao reconhecimento em tempo real das imagens, verificando-se por vezes alguma lentidão no processo de reconhecimento de grandes e complexos textos. Também o facto de o processo de captação das imagens não estar

dotado de um sistema automático de ajuda ao utilizador, que o oriente para que a captação da imagem seja o mais correta possível, apresenta-se como uma limitação desta aplicação, uma vez que a aplicação destina-se a invisuais que poderão ter mais algumas dificuldades em realizar essa tarefa. No entanto, essa limitação poderá e deverá ser alvo de melhoria em trabalhos futuros.

O trabalho de pesquisa, implementação e optimização desenvolvidos, permitiu conceber uma aplicação gratuita, que se encontra já num estado de possível utilização, mesmo com as limitações atrás referenciadas, permitindo a leitura de textos, desde que as condições de luz sejam as ideias para captação da imagem e o equipamento seja corretamente direcionado ao texto que se pretende ouvir, para que o reconhecimento e a leitura sejam o mais satisfatória possível.

5.1. Trabalho Futuro

A framework open source Tesseract utilizada no projeto, pode ser substituída por frameworks comerciais, uma vez que, embora represente custos, a qualidade dos resultados obtidos no reconhecimento óptico poderão melhorar substancialmente. De referir que a utilização das frameworks comerciais apresentou melhores resultados que a framework gratuita, quando utilizadas sem qualquer filtro, o que leva a crer que a utilização dessas frameworks em imagens que tenham sido submetidas a filtros para melhorar o reconhecimento levará a resultados ainda mais satisfatórios.

Também os parâmetros alterados para obter melhores resultados podem ser alvo de alterações e realizados novos testes, que possam permitir alcançar melhores resultados. Por exemplo o alargamento da base de dados de imagens, procurando obter um conjunto mais vasto e representativo de situações reais de utilização podem fornecer dados mais consistentes, mas também a realização de mais testes com novas combinações de filtros, ou com mais e exaustivas alterações aos parâmetros internos do OCR.

Um outro teste que poderá ser importante para aferir a qualidade do sistema, é solicitar a um utilizador que ouça a leitura de um texto, que tenha sido reconhecido e sintetizado pelo sistema, que o escreva, e posteriormente seja feita a comparação entre o texto compreendido pelo utilizador e o texto original.

O projeto poderá ser alvo de alterações, como por exemplo no que diz respeito ao alinhamento da imagem a ser captada. Recordar que se trata de uma aplicação para invisuais, que devem poder utilizar autonomamente a aplicação, pelo que no futuro a aplicação deve auxiliar o utilizador no alinhamento do dispositivo com o texto a reconhecer. Essa ajuda pode passar pela detecção da área onde se encontra o texto, e através de sinais sonoros dar indicações ao utilizador para orientar o dispositivo e conseguir captar todo o texto corretamente. O reconhecimento pode eventualmente ser feito através de um processo de leitura dos pixéis da imagem, procurando encontrar o limite entre padrões mais claros, onde possivelmente não haja texto, como nas margens de uma folha, e padrões mais escuros, onde potencialmente se encontre texto.

Outras das questões que deve ser alvo de melhorias é no reconhecimento em tempo real, uma vez que para um utilizador invisual a possibilidade de obter uma leitura em tempo real significaria um acréscimo de qualidade de vida, diminuindo o tempo que precisa para ler um documento mais extenso.

O suporte para outras linguagens deverá ser outra das questões a serem revistas. Nesta fase o projeto foi pensado e desenvolvido para a língua inglesa, mas no futuro devera permitir a utilização em outras línguas, alargando assim o número de pessoas que podem beneficiar das vantagens da aplicação.

Fica também reservado para trabalho futuro o incremento das opções de menu, permitindo ao utilizador alterar outras opções da aplicação tais como a língua em que esta funciona ou configurações de sons e orientações.

Referências

- [1] BIHINA, M. M. B. The Phone Reader. Grahamstown, South Africa, November, 2012.
- [2] CONDADO, P., Quebra de barreiras de comunicação para portadores de paralisia cerebral. Doutoramento em Engenharia Electrónica e Computação, Ciências de Computação, Universidade do Algarve, 2009
- [3] DUTOIT, T. A Short Introduction to Text-to-Speech Kluwer Academic Publishers, Dordrecht, Boston, London, 1997.
- [4] EIKVIL, LINE. OCR Optical Character Recognition, 1993
- [5] ELMORE, M. AND MARTONOSI, M. A morphological image preprocessing suite for ocr on natural scene images, 2008

- [6] FONSECA, N., REIS, C., SILVA, C., MARCELINO, L., CARREIRA, V. Desenvolvimento em iOS iPhone, iPad e iPod Touch, Lisboa, FCA – Editora de Informática, Lda, 2012.
- [7] FUJISAWA, H. A view on the past and future of character and document recognition. In Proceedings of the Ninth International Conference on Document Analysis and Recognition – Volume 01, pages 3-7, Washington, DC, USA. IEEE Computer Society, 2007.
- [8] LEMMETTY, S. Review of Speech Synthesis Technology. Masters Thesis, Helsinki University, 1999.
- [9] MANGNES, B. The use of Levenshtein distance in computer forensics. Master's Thesis. Master os Sciente in Information Security. Department of Computer Science and Media Technology, Gjøvik University College, 2005
- [10] MITHE, R., INDALKAR S., DIVEKAR, N. Optical Character Recognition. International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-2, Issue-1, March 2013
- [11] PRIZMO, Prizmo - Escaneamento, OCR & Fala. Disponível online em https://itunes.apple.com/PT/app/id366791896
- [12] SAYTEXT, Saytext. Disponível online em https://itunes.apple.com/PT/app/id376337999
- [13] SILVA, A. Uma Avaliação Experimental da Combinação de Métricas de Similaridade para o Alinhamento de Ontologias através de Mineração de Dados. UNIRIO, 2013. Dissertação de Mestrado. Departamento de Informática Aplicada, UNIRIO.
- [14] TALKING CAMERA PRO, Talking Camera Pro - for visually impaired/blind, disponível online em https://itunes.apple.com/PT/app/id544295997
- [15] THOMAS, S. Natural Sounding Text-To-Speech Synthesis Based On Syllable-Like Units, Department Of Computer Science And Engineering Indian Institute Of Technology Madras, 2007.