FEDERAL UNIVERSITY OF PARÁ
INSTITUTE OF EXACT AND NATURAL SCIENCES
COMPUTER FACULTY

JOÃO VICTOR DA SILVA DIAS CANAVARRO

Book2Speech: a Low-cost Machine for Converting Textual Content Into Audible Feedback via Synthesized Speech

BELÉM – PARÁ
2021

FEDERAL UNIVERSITY OF PARÁ
INSTITUTE OF EXACT AND NATURAL SCIENCES
COMPUTER FACULTY

JOÃO VICTOR DA SILVA DIAS CANAVARRO

Book2Speech: a Low-cost Machine for Converting Textual Content Into Audible Feedback via Synthesized Speech

Graduation Thesis presented to obtain the degree of Bachelor of Science in Computer Science, of the Institute of Exact and Natural Sciences, of the School of Computing.

BELÉM – PARÁ

2021

Book2Speech: a Low-cost Machine for Converting Textual Content Into Audible Feedback via Synthesized Speech

Este trabalho foi julgado adequado em _____ para a obtenção do Grau de Cientista de Computação, aprovado em sua forma pela banca examinadora que atribuiu o conceito _____.

_____

Prof. Dr. Nelson Cruz Sampaio Neto

ORIENTADOR

_____

Prof. Dr. Carlos Gustavo Resque dos Santos

MEMBRO DA BANCA EXAMINADORA

_____

Me. Cassio Trindade Batista

MEMBRO DA BANCA EXAMINADORA

_____

Profa. Dra. Regiane Silva Kawasaki Francês

DIRETORA DA FACULDADE DE CIÊNCIA DA COMPUTAÇÃO

I dedicate this work to my family and friends.

ACKNOWLEDGEMENTS

First of all, I thank God, for the resilience, wisdom and opportunities that have been conceived for me throughout my life.

To my family, who encouraged me to fight for my dreams and who endeavored to ensure that I had wings to pursue them. In particular, to my father, Lucas Silva Canavarro, and my mother, Rita da Silva Dias, for their unconditional love, dedication, support, confidence and encouragement to my potentials and qualities. Also, to my grandparents and uncles, for the enormous affection, innumerable lessons of life and essential presence in this journey. In special, to my grandparents Edna Canavarro and Robysson Canavarro, and my aunt Ednea Canavarro, for providing me access to technology since 2012 with my first laptop until the computer I use to write this work.

To the Federal University of Pará, for the environment, knowledge and services offered to the community, and for granting scholarships for scientific initiation and research during my university education.

To the FACOMP professors, for sharing their knowledge, valuable experiences and for introducing me to the wonderful world of technology. In particular to my advisor, Prof. Dr. Nelson Cruz Sampaio Neto, and my research colleague, MSc. Cassio Trindade Batista, for the opportunity and trust in the work that I am performing for almost 3 years at LABVIS/FalaBrasil, for the patience and also for the helpful advices during the production of this thesis.

To Matheus Holanda Nascimento, one of the best friends I've ever had, this work is in his memory. And to all my friends, whom I would not be able to list, and for this reason I dare not name them, but without whom the path here would not have been the same.

To my classmates and research colleagues at Labvis, for the countless collaborations and activities carried out during graduation, for the daily sharing of knowledge and for the opportunities to work and experience the most diverse areas of technology.

João Victor da Silva Dias Canavarro

“Pain is temporary. Quitting lasts forever.”
Lance Armostrong

ABSTRACT

Even with the great advances in technology regarding the ease of disseminating information digitally, printed media continues to be an important and frequently used means of conveying knowledge. However, for the visually impaired, there is still a major barrier in accessing physically distributed content, since alternative methods of reading, such as Braille, are often not available, in addition to the lack of literacy in these writing systems among the blind population. To overcome these difficulties, solutions in the Assistive Technology areas have been proposed, both in academic and commercial ambits, aiming at increasing the independence, quality of life and inclusion of this portion of the population. The present work seeks to develop a standalone reading machine to recognize and convert the textual content of books and derivatives into audible feedback via synthesized speech. Based on the Raspberry Pi 3, an embedded microcomputer, equipped with the Pi NoIR (InfraRed) camera module, Book2Speech is a machine that performs the image acquisition and the Optical Character Recognition and Text-to-Speech procedures, making use of modules for image and text processing to improve the representativeness of the synthesized voice, reproduced through an external speaker. Due to the lack of available document-image datasets in Brazilian Portuguese, the ICDAR2015 dataset, composed mostly of English text, was used to evaluate Book2Speech's performance. Also, the processing time and error rate metrics were considered, which are calculated from the difference between the text recognized by the machine and the reference. Regarding the results obtained, the dewarping method using the L-BFGS-B algorithm as the optimizer obtained the lowest word error rate (12.32%), while the average threshold pipeline followed by dewarping with L-BFGS-B obtained the lowest character error rate (13.27%). On the other hand, the spelling correction methods evaluated did not lead to good results, often increasing the book2speech error rate. Finally, it is worth mentioning that the system developed, along with the tools and resources used, are freely available.

Keywords: Assistive Technology. Image Processing. Spelling Correction. Optical Character Recognition. Text-to-Speech. Embedded Systems.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
API	Application Programming Interface
ANPR	Automatic Number Plate Recognition
ASR	Automatic Speech Recognition
AT	Assistive Technology
CBDA	Camera-based Document Analysis
CCA	Connected-component Analysis
CER	Character Error Rate
CLI	Command Line Interface
CSI	Camera Serial Interface
DNN	Deep Neural Network
DSP	Digital Signal Processing
HMM	Hidden Markov Model
HWR	Handwritten Recognition
HCI	Human-Computer Interaction
HP	Hewllet Packard
IBM	International Business Machines Corporation
ICDAR	International Conference on Document Analysis and Recognition
LPF	Low-pass Filter
LSTM	Long Short-Term Memory
NLP	Natural Language Processing
OCR	Optical Character Recognition
OMR	Optical Music Recognition
PWD	People With Disabilities

RNN	Recurrent Neural Network
SBC	Single-board Computers
TTS	Text-to-Speech
VI	Visual Impairment
WER	Word Error Rate
WIL	Word Information Lost

CONTENTS

1 INTRODUCTION

1.1 Contextualization

Most of the technologies available on the market are already economically accessible for a large part of the population. The use of electronic devices, such as smartphones and personal computers, became more common in people's daily lives to help with various tasks. Thus, with the Internet becoming more and more globalized, access to information takes place in a simple and practical way.

During the last decades, technology development has reached a stage where communication between people and computer systems incredibly resembles, for example, the communication established between two people. Thanks to the various works carried out in the area of human-computer interaction (HCI) and artificial intelligence (AI), machines are approaching human behavior in the sense of being able to perform tasks similar to listen, speech, understand, estimate and act automatically and independently.

Despite the increasing focus on information and communication technologies, print media continue to be an important and frequently used means of conveying information [1]. In fact, even with the great advance of the Internet, knowledge continues to be strongly disseminated through physical format, as in articles, books, encyclopedias, newspapers, magazines and more, especially during school and academic education. It has been estimated that more than 200 million books are being published every year, many of which are being distributed and printed on traditional papers [2]. However, given that the availability of haptical written mechanisms such as Braille are not always plentiful, solutions under the concept of Assistive Technology must be constantly explored for people with low vision or total blindness, which have been prevented from accessing regular textual information.

Assistive Technology (AT) is an interdisciplinary field of knowledge dedicated to increasing the independence and mobility of people with disabilities (PWD), encompassing products, methodologies, practices and services that aim to promote their autonomy, quality of life and social inclusion [3, 4]. The term refers to technology that provides assistance to people with different types and degrees of limitation in order to reduce the effects of disabilities and allowing them to actively participate in their respective routines. According to the Convention on the Rights of Persons with Disabilities [5], published by the United Nations in 2006, the eyes of the 21st century had quit seeing PWD as "objects" of charity, medical treatment and social protection, and come to see them under a new perspective, in which the PWD are active "subjects" in society, capable of claiming their rights and to make decisions based on their own consent.

Through AT, it is possible to reduce the difficulties experienced by people who need solutions that do not leave them out of the use of electronic devices. Aiming to reduce the exclusion imposed on PWDs due to the difficulty or total inability to perform specific tasks, accessibility is seen as a fundamental element to raise self-esteem and the degree of independence of this group. In addition, the solutions presented can also be useful for people without any disabilities, since the usage and interaction with various electronic apparatus becomes more practical, comfortable and natural [6]. In fact, technology has the potential to enhance individuals" ability to participate fully in societal activities and to live independently [7].

In this sense, technologies such as optical character recognition (OCR) [8], as well as text-to-speech synthesis (TTS) [9], when combined, can enhance human-computer interaction while turning resources accessible to people with visual impairments (VI). OCR refers to the system that receives a digital image as input, extracts and recognizes textual content, and generates a string of characters as the output. This string might serve as input to a TTS system, which is responsible for synthesising a digital speech signal. By applying such concepts, stand-alone reading systems are independent machines that are able to scan a printed document, and produce an audio version of the document for visually impaired and blind readers. Although there is a decrease in usability and speed of task execution in contrast with standard methods, such alternative interaction techniques provide a much more pleasant and enjoyable experience [6].

Therefore, this work aims at developing an stand-alone reading machine with audible response for the visually impaired, named Book2Speech. By implemeting the OCR-TTS pipeline as a standalone automatic reader, Book2Speech can read aloud the words from pictures taken from book pages or similar physical textual content, The prototype is configured as a portable device over a Raspberry Pi 3 development board. OCR is obtained through the Tesseract open-source engine while speech synthesis is achieved communicating with the gTTS (Google Text-to-Speech) application programming interface (API). Images are taken by the Raspberry Pi NoIR camera module, which presents good results, especially under low light environments. Finally, a regular 8 Ohm external speaker is used to reproduce the synthesised speech generated by the TTS system.

The potential target users of this work are people with visual limitations (partial or complete) with preserved cognitive and superior motor skills. The idea is to provide these users with access to the vast knowledge and information present in books and derivatives that do not have a Braille version available or are difficult to access. It is worth mentioning that Book2Speech is an open-source project, providing all of its implementation, command line interface (CLI), scripts, libraries and packages used publicly on the Internet [10].

1.2 Justification

Disability, by definition, is the partial or total incapacity of a certain person to perform common everyday tasks, when compared to others of a standard group of people. Disability can be the consequence of a physical, mental, sensory, emotional, developmental impairment or a combination of some of these factors. The term "disability", as it is quite extensive, encompasses impediments, limitations in carrying out activities and restrictions on participation in society [11]. Still, people with disability experience poorer health outcomes, have less access to education and work opportunities, and are more likely to live in poverty than those without a disability [12].

As stated by the World Health Organization (WHO), approximately 15% of the whole population of the globe has some kind of disability [13]. This number really speaks volumes, since it reveals that in a population of 7.9 billion people, around one out of seven (over 1 billion people) is a disabled person. Regarding visual disabilities only, approximately 285 million are visually impaired, representing more than 3.86% of the entire population. In Brazil, regarding visual disability in particular, approximately 35.8 million (18.8%) has some kind of problem in their sight, according to the last census of the Brazilian Institute of Geography and Statistics (IBGE) in 2010 [14]. Table 1 shows the profile of the Brazilian population with disabilities.

Table 1 – Profile of the Brazilian population with disabilities.

Disability	**Description**	**Number of People**	**Percentage**
Cognitive	Mental or intellectual problems	2.611.536	1,369%
Auditory	Deafness or general difficulties	9.717.318	5,094%
Motor	Paralysis or general difficulties	13.265.599	6,950%
Visual	**Blindness or general difficulties**	**35.774.392**	**18,754%**

Source: Extracted from 14.

In the last decades, several methods, such as Braille, were invented as an option for the visually impaired to acess the content of engraved text. However, Braille poses two main issues. Firstly, very few books are published along of or modified into a Braille version. Secondly, only a small part of the blind population is dictated to read by this method. In fact, fewer than 10% of the US blind people can read Braille [15]. Such scarcity of resources and literacy has led to a growing need to explore commercial and academic solutions to provide the visually impaired with access to the content distributed in books.

Despite the current existence of numerous instruments geared towards assistive technologies, such as wheelchairs, glasses, and software that promote accessibility when using computers and smartphones, a large portion of PWD still do not have access to these tools. The World Health Organization also estimates that, in underdeveloped countries, for example, only 15% of PWD have access to these assistive technologies [13]. This scenario can be mainly driven by two factors: i) the high cost of some of these instruments; and ii) the low availability given

factors of manufacture and export to other countries. In fact, depending on the needs of individual visually impaired persons, the average cost of such technologies varies from approximately US$1,000 to as much as US$20,000 [16].

In constrast with the available technologies, Book2Speech was designed with cheaper hardware in order to promote more access to the knowledge distributed through physical form to people with visual impairments, especially those who do not have the economic conditions to obtain the available commercial instruments. Still, people with the right knowledge can also replicate Book2Speech on their own setup.

1.3 Objectives

The main objective of this work is to develop a stand-alone reading machine, which is able to help people with different degrees of visual impairment to have access to the content of books and derivatives. At first, as a proof of concept, as a system developed to run on a Linux-based Raspberry Pi 3 with a NoIR Camera module attached and external speaker, it will be possible to synthesize a sound signal from the recognized text referring to the images captured by the prototype. It is also important to note that this work is low-cost and with its system implementation freely available on the Internet [10].

Unfortunately, given the current pandemic scenario, tests with volunteers who are visually impaired could not be conducted. Hence, the proposed work has not been concerned with the usability of the product up to this point. In this sense, Book2Speech's performance was evaluated considering various optical character recognition error rate metrics and processing time measures. Finally, given the scarsity of document-image datasets in Brazillian Portuguese, such evaluation was conducted with an initial focus on the English language.

1.3.1 Specific Objectives

The specific objectives can be divided into five main subgroups: i) study and implementation of assistive technologies; ii) study and implementation of image processing modules; iii) study and implementation of text correction and processing methods; and iv) performance and accuracy tests. A brief description of each topic is given below.

i) Study of assistive technologies systems:
   - Optical character recognition;
   - Text-to-Speech.

ii) Study and implementation of image processing modules:
   - Camera module control;

- Binarization;
- Noise alleviation;
- Distortion correction.

iii) Study and implementation of text correction and processing methods:

- Spelling correction;
- Text tokenization.

iv) Performance and accuracy tests

- Acquisition and preparation of the test dataset;
- Study of OCR performance metrics evaluation;
- Definition of the tests scenarios and parameters;
- Evaluation of the results in order to overcome the disadvantages and possibly improve the system in the next versions.

1.4 Thesis Organization

This chapter provided a brief introduction regarding assistive technologies application and the motivations of this area of research. In addition, it was discussed the justification of this work given the current scenario of people with disabilities, considering mainly those who are in low-income situations.

The next chapters are organized as follows:

Chapter 2 Related Work. This chapter discusses some related work regarding the commercial as well as the academic works of stand-alone reading systems and their derivatives.

Chapter 3 Background. This chapter presents some general background information and theory about all system, tools, algorithms and metrics used in the work.

Chapter 4 Methodology. This chapter details the activities performed to build the project. It will also describe in details the procedures implemented from the image acquisition to the generation of the synthetic speech referring to the recognized text, as well as the details about the physical design of Book2Speech.

Chapter 5 Experiments and Results. This chapter details the evaluation dataset, as well as tests stages performed to calculate the error rate of the proposed work in different scenarios. The results obtained as well as the general difficulties will also be discussed in this chapter.

Chapter 6 Considerations and Future Work. This chapter presents some considerations and shed some lights on what can possibly be the next steps to be followed.

2 RELATED WORK

When it comes to commercial products, several approaches for stand-alone reading systems have been developed over the last years. OrCam MyEye 2 [17] is a wearable device that attaches to the user's glasses and claims to read textual content from books, smartphone screens, and any other surface through a 13 megapixel camera, fully off-line and with support for 20 languages. OrCam MyEye 2 requires the user to have full control over their head and hand movements to send the system activation commands by touching the device's surface, in order to recognize the target text. The latest version of the product is currently costing, in 2021, US$4250 on Amazon's official website.

The CZUR ET16 Plus Scanner [18] functions as a scanning device specializing in the digitalization of books and derivatives. Similar to Book2Speech, the images are captured by a camera that is vertically aligned with the horizontal plane that the book lies on. The device also features several lighting sources to remove possible shadows on the pages, improving the final readability of the captures. On the other hand, the ET16 Plus does not offer voice synthesis capabilities, making it necessary for people with visual impairments to use third-party software such as screen readers and similars to access the digitalized content. The product can be found for sale in price ranges from US$400 to US$425. A cheaper version of the scanner is also available, without some of the previous features, such as artificial lighting and a built-in display to accompany the captures, the CZUR Shine Pro can be found in the average price range of US$129.

Aimed at portability, Scanmarker Air Pen Scanner [19] and C-Pen Reader [20] are two devices similar to a text marker, where the user slides them over the text lines of a document to scan its contents. The first one, aimed at scanning and helping in the learning of a new language, has support for recognition and translation of approximately 40 languages. However, for the user to have access to the TSS output, it is necessary to use a smartphone or computer with the developer's proprietary software installed, since the product itself only performs the OCR process. Meanwhile, C-Pen Reader only supports English, French, Spanish, German and Italian, but has built-in TTS support. Both devices can be found for sale for an average price of US$120 and US$250, respectively.

In regard to academia, several works have been presented on the development of automatic reading systems for the blind and visually impaired. The following strings were applied into the Google Scholar plataform as search criteria to find the candidate works to be included in this revision: "automatic reading system" and "OCR-TTS system".

In [21], a Windows desktop GUI application was proposed for text recognition and speech synthesis. The system is used on an ordinary desktop computer and requires an external

scanner to capture the images. After image acquisition, the program preprocesses the images by applying binarization, denoising, deskewing and segmentation algorithms. Finally, the image is passed to the OCR module followed by the TTS API, which is unspecified, generating the sound signal that is played by the computer speakers.

Also in the context of embedded machines, the FingerReader device [22] is a large ring that slips on the index-finger and features a front facing camera that scans the text that the finger is pointing to. Smart software powering the device recognizes the line of text under the finger and reads only that line. The OCR and TTS procedure is performed by Tesseract and Flite Text-to-Speech engines, respectively.

It was proposed in [15] a system based on the Raspberry Pi 2B as a mobile processing unit connected to glasses adapted with a HD micro-camera and an external Bluetooth headset. To capture the images, the user should hold the book open (two pages) with his hands stretched straight at the level of his eyes to capture the image in order to minimize the curviness of the pages, and then start to move the book toward him slowly but steadily. This movement is necessary for the camera calibration step. After the image is captured, it is binarized and deskewed before the OCR-TTS pipeline.

In [23] was suggested an Arabic reading machine aimed at the visually impaired. The idealized prototype is composed of a dark colored base with a surface barely larger than an A4 page where the printed paper is to be placed. A 3 megapixel digital camera is located above the dark base and takes a picture of the print text in order to send it to the integrated computer. The machine considers the SAKHR software for both OCR and TTS because, according to the author, it presents the most natural synthesized speech. Even describing the processes required to develop such a system, no real prototype was presented. Finally, the author considers one of the following on-board boards as the processing unit of the machine: EPIC, EBX or 5.25“ SBC. The price for such equipment, without considering the camera, speakers and other equipment like cables and connectors, is on average 281 dollars [1].

The proposed work in [24], while not providing much detail about the machine development methodology and evaluation methods, is also based on the Raspberry Pi 3 microboard, as is Book2Speech. The device uses the Pi NoIR camera module for image acquisition, while the Tesseract and Pico engine are responsible for the OCR and TTS procedure, respectively. For controlling the system, an external keyboard is used, while external speakers are necessary for sound output. One point that caught attention was the use of two binarization algorithms in sequence, Global and Otsu’s threshold, before smoothing the image in the pre-processing phase, since already binary images cannot be “rebinarized”.

Another approach to producing a portable reading system for blind people is the prototype Read IT [1], similarly idealized to run on a Raspberry Pi 3 microcomputer. In the proposed

[1] Direct conversion of the board’s price provided by the author, 2480 Moroccan Dirham (MAD), to the US dollar.

work, an image of the text is captured by a video camera positioned in the user's sunglasses. This image is analyzed and the text content identified drives the speech synthesizer card. The resulting speech output is then delivered to the user via a single earphone. The signal processing unit, speech synthesizer card and battery power supply are housed in a small box worn at waist level. Also, manual control is via a small hand-held Braille key pad. As in the other works cited, the engine responsible for the optical character recognition procedure is Tesseract, while an unspecified modular hardware is used for TTS [25].

This research aims to present a solution to reduce the digital exclusion experienced especially by people with varying degrees of visual needs and who are on the margins of the electronic world due to the absence of devices adapted to their needs. In contrast to commercially available solutions, Book2Speech aims to provide a low-cost automatic reading machine for the visually impaired that cannot afford the equipments currently available on the market. Table 2 presents the average price of the components necessary to built the work developed. Regarding the proposed works in the academic field, Book2Speech aims to provide an open-source system that performs the image and text processing procedures, together with the OCR-TTS pipeline, within a single processing board and without the need for users to use their upper movements frequently.

Table 2 – Average price, in dollars, of the components used to build the Book2Speech prototype.

Component	**Price ($)**
Raspberry 3 Model B	35.0
Pi NoIR Camera Module	25.0
Phone Holder	10.0
External Speaker	15.0
Keyboard	18.0
Total	103.0

3 BACKGROUND

This chapter aims to present the theoretical framework of the most important modules for the development of the device proposed in this work, such as: i) optical character recognition; ii) speech synthesis; iii) image processing; iv) spelling correction, v) specifications of the Raspberry Pi 3 and NoIR Camera module; and vi) evaluation metrics.

3.1 Optical Character Recognition

Optical Character Recognition (OCR) is the process of converting scanned images of text into digital documents that can be processed, edited, searched, saved, and copied for an unlimited number of times without any degradation or loss of information [2]. This technology allows machine to recognize the text automatically through an optical mechanism whether from a scanned document, a picture, a scene-photo or from subtitle text superimposed on an image [26]. OCR is a vast field with a number of varied applications such as invoice imaging, legal industry, banking, health care industry, etc. It is also widely used in many other related fields like captcha, Institutional repositories and digital libraries, Optical Music Recognition (OMR), Automatic Number Plate Recognition (ANPR) and Handwritten Recognition (HWR) [27].

In the 60s, the International Business Machines Corporation (IBM) was very active in developing OCR systems, recognizing that this technology was a very important input device for computers. The first commercialized OCR was the IBM 1418, which was designed to read a special IBM font, 407 [28]. By the end of the 1960's, the next targets for document readers were for poor-print-quality characters, and hand-printed characters for a large category character set, such as Chinese characters. These targets have been achieved partially and such commercial OCR systems appeared roughly during the decade from 1975 to 1985 [29].

Since then, several new techniques of optical character recognition have been developed, and, along with the advances in the computational performance of current technologies, new systems with higher accuracy rate and speed have emerged. Various approaches such as matrix matching, fuzzy logic, feature extraction, structural analysis and neural networks where used to design OCR pipelines [30]. However, even today's systems still face challenges when trying to recognize textual content of digital images [2, 8, 31].

While OCR is well-established for majority languages, the same cannot be said for endangered languages [32]. Actual OCR models might struggle to recognize non-ocidental alphabetic characters, diacritical marks and symbols such as japanese Kanjis [33]. Also, the small difference between some characters, for example, the digit "0" and the letter "O", especially in small resolution images, might lead into misrecognitions. Still, images geometric deformations, common artifact on camera-captured documents, make it difficult to trace text lines which

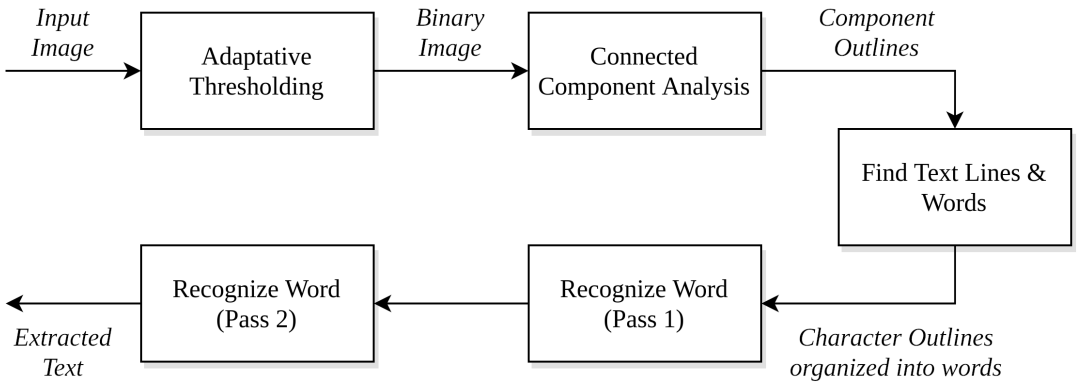
might imply into suboptimal results [31]. Practically, the error rate of OCR systems can fairly become high, occasionally over 10%, if the papers being scanned have numerous defects such as bad physical condition, poor printing quality, discolored materials, and old age papers [2]. In this sense, even with the great advantages during the last decades, OCR systems performance still strongly correlated to the quality of the input image and language resources available for recognition models training.

3.1.1 Tesseract Engine

Tesseract is an free optical character recognition engine developed at Hewllet Packard (HP) in between 1984 to 1994, originally as an PhD research project. It was modified and improved in 1995 with greater accuracy and in late 2005, HP released Tesseract for open source [34]. Since 2006, the engine is developed and maintained by Google as an public repository at GitHub [1] under the Apache 2.0 license.

Writen in C++, Tesseract aims for high portability. Currently supporting over 100 languages and various types of output, such as PDF, XML and HTML, the engine can be accessed through its command line interface (CLI) or by several wrappers developed for various programming languages. Figure 1 presents the main components of Tesseract's text recognition pipeline.

Figure 1 – Top-level block diagram of Tesseract OCR system.



Source: Adapted from 8, 26

Processing starts with the binarization of the original image by applying adaptive thresholding in order to separate the pixels into two groups, white for background and black for text. The main idea is to make the text as clear as possible to be recognized, removing unnecessary noise and artifacts from the image background. Image processing techniques are discussed in more depth in Section 3.3.

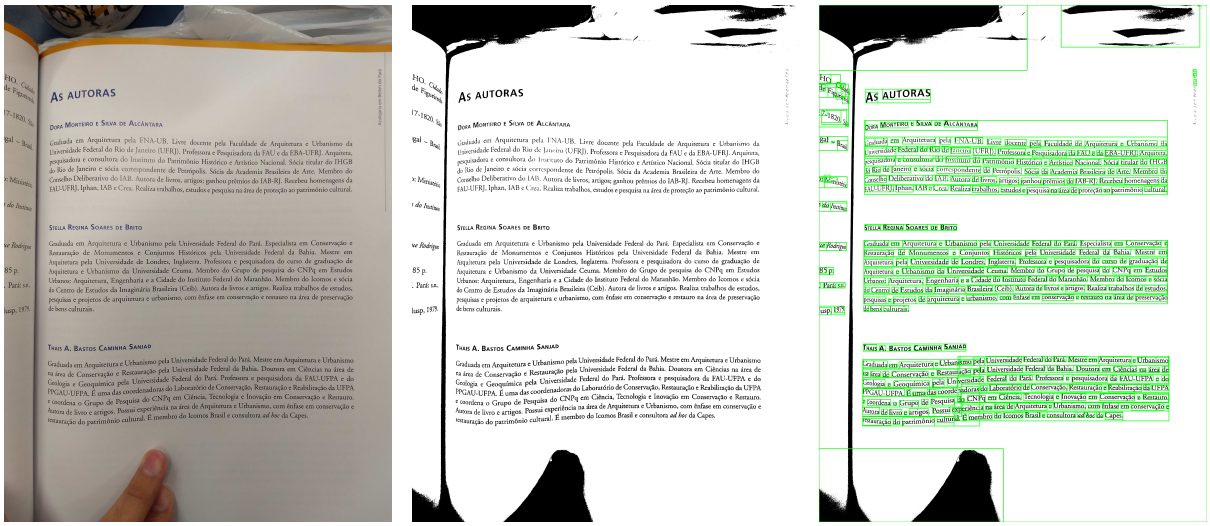
The next step, connected component analysis (CCA), is responsible for recognizing and storing text and words outlines. These outlines are gathered together into *blobs* in order to

[1] https://github.com/tesseract-ocr/tesseract

detect text lines. A blob is a putative classifiable unit, which may be one or more horizontally overlapping CCs, and their inner nested outlines or holes [35]. Tesseract them analyze these regions searching for some fixed pitch area or proportional text size. Text lines are then broken into words according to the kind of character spacing [36]. Figure 2 shows the respective outputs from the steps above.

After finding text lines and words, the image content recognition step is then started as a two-pass process. The first is responsible for attempting to recognize each word in turn. In most cases, a blob corresponds to a character, so the word recognizer first classifies each blob, and presents the results to a dictionary searcher to find a word in the combinations of the classifier choices for each blob in the word. While the result from a word is unsatisfactory, Tesseract attempts to improve the result by chopping the blob with worst confidence from the character classifier. On the other hand, the blobs that are in a dictionary and are not dangerously ambiguous, are passed to an adaptive classifier for training.

Figure 2 – Tesseract adaptative thresholding binarization and connected component analysis detection output.



(a) Original image (b) Binary image (c) Boundary outlines

Source: Author.

Since it may have learned something useful within the end of text to make a contribution near the top of the page, a second pass is run over, in which words that were not recognized well enough (low confidence) are processed again. Finally, a final step resolves fuzzy spaces, and checks alternative hypotheses to locate small-cap text.

Since Tesseract 4.0, a new neural network subsystem was configured as a text line recognizer. It has its origins in OCRopus" Python-based LSTM (Long Short-Term Memory) implementation [2] and can be used with the legacy layout analysis module explained above to analyze text within a large document, improving the recognition accuracy.

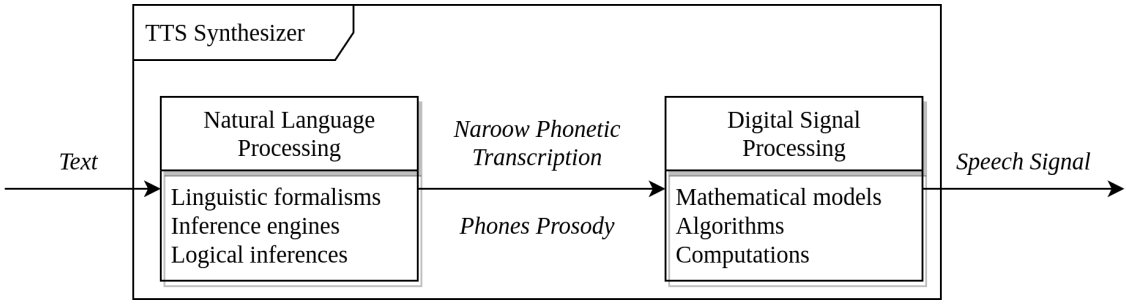
[2] https://github.com/ocropus/ocropy

3.2 Speech Synthesis

Speech synthesis consists of artificially producing human speech, through the automatic generation of the speech signal from text [37]. Several works on speech synthesis have been developed for decades and considerable progress has been achieved. However, the quality in terms of the naturalness of the voice produced still has gaps, especially with regard to the adaptations that speech may suffer considering aspects such as intonation and emotion, which are associated with the expressiveness of the content to be synthesized.

One of the applications of speech generation can be found in Text-to-Speech (TTS) systems, which convert an input text into an artificial voice that is intelligible and as natural as possible. According to [9], TTS systems are mainly composed by two components: Natural Language Processing (NLP) and Digital Signal Processing (DSP). Figure 3 shows the functional diagram of a general text-to-speech system.

Figure 3 – General diagram of a TTS system which converts written text to a phonemic representation, then to waveforms that can be output as sound.



Source: Adapted from 37.

The NLP module (also referred as front-end) is responsible for producing a phonetic transcription of the text, together with the desired intonation and rhythm. The front-end performs the normalization (or pre-processing) of the text, converting symbols, such as numbers and abbreviations, into equivalent words written in full. It also implements grapheme-to-phone (G2P) conversion, syllabification and stressed syllable determination. These tasks assign phonetic transcription to each word and mark the text with prosodic information. Phonetic transcriptions and prosodic information together make up the linguistic symbolic representation that serves as input for the synthesizer.

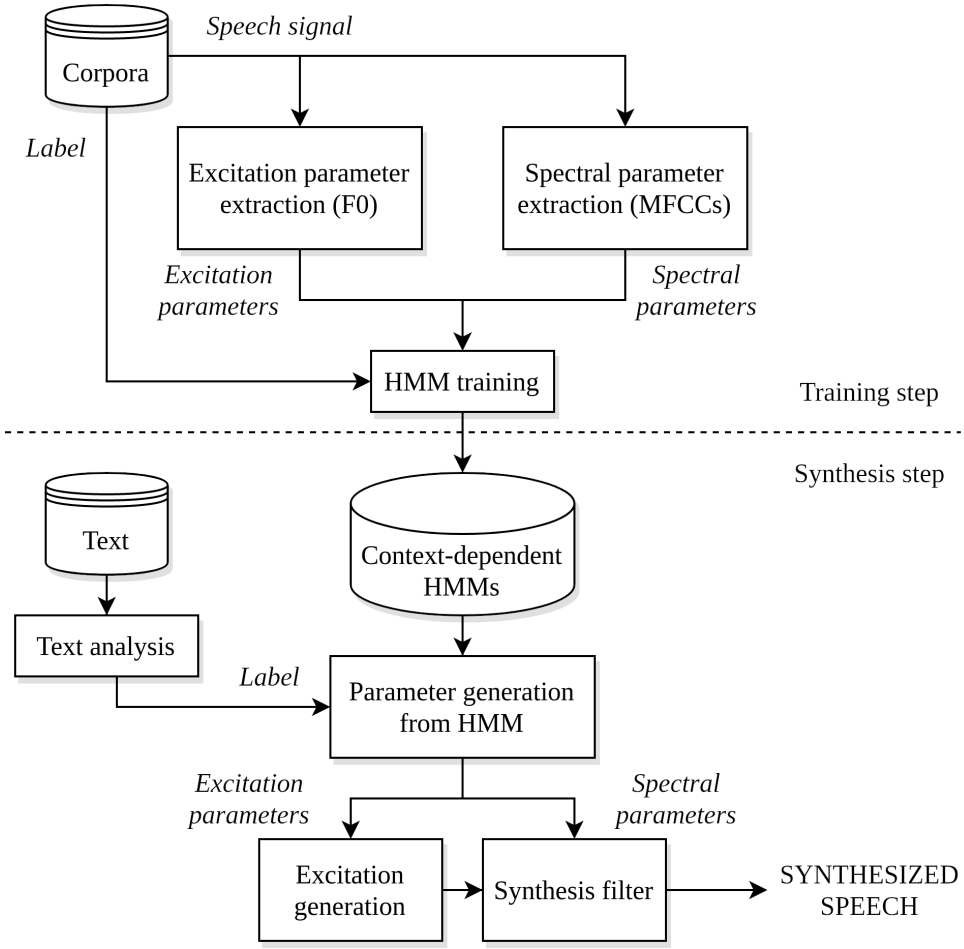
The DSP module (also referred as backend or synthesizer) transforms the symbolic information received into natural-sounding speech. Typically independent of language, the synthesizer is the block that effectively generates the sound, converting the linguistic representation into speech signal.

In recent years, TTS systems were developed as an output device of human-machine interfaces, and can be used in several areas such as in car navigation systems, information retrieval over the telephone, voice mail, assistive technologies and so on. The increasing availability of

large speech databases makes it possible to construct more robust TTS systems, which are referred to as data-driven or corpus-based approach, by applying statistical learning algorithms. These systems can generate natural and high quality synthetic speech by reproducing voice characteristics of the original speakers [38].

For constructing such systems, the use of Hidden Markov Models (HMMs) has arisen largely. Voice synthesis based on HMMs is already quite popular due to its flexibility in synthesizing voice considering characteristics such as style and speaker's speech individuality, in addition to the possibility of expressing emotional aspects in the voice [39]. Figure 4 presents the HMM's training and synthesis step top-level block diagram.

Figure 4 – Top-level scheme of a HMM-based TTS system.



Source: Adapted from 38, 39.

Simply put, a HMM-based TTS system works through the execution of two stages: training and synthesis. During the training phase, parameters related to the voice spectrum and excitation, such as mel-frequency cepstral coefficients (MFCCs) and fundamental frequency (F0), are extracted from a voice base (corpora) and modeled through HMMs, considering phonetic, linguistic and prosodic aspects of speech. In this specific case, there is an HMM for each phone (handset), but the statistical model can be based on larger units, such as biphones, triphones,

words, etc.

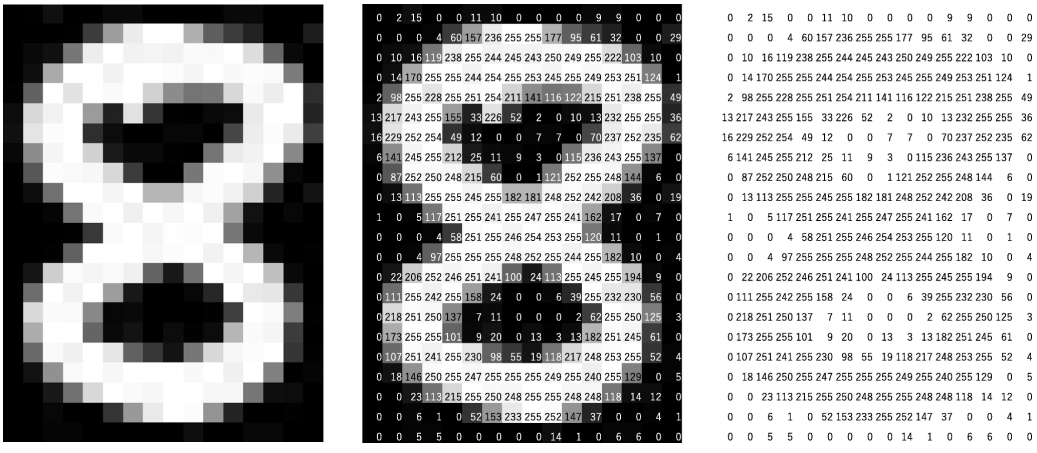
As a result of this first stage, there is a unified framework with models of the excitation and duration of the voice spectra. In the synthesis step itself, the input text is transformed into a sequence of phones where each HMM corresponding to a phone is concatenated with the one of the anterior and posterior phone (depending on the context). After the concatenation process, each HMM issues a set of observations (MFCCs parameters, F0, among others). Finally, a voice waveform is synthesized directly from the estimated parameters.

Currently, several TTS engines with different languages, dialects and specialized vocabularies are available through third-party publishers, for example Google Cloud Text-to-Speech [40], or as open-source software such as Festival, eSpeak and Gnuspeech [41, 42, 43].

3.3 Digital Image Processing

Digital Image Processing refers to processing digital images by means of a computer. A digitalized image is composed of a finite number of elements, generally referred to as picture elements, image elements, pels, or pixels. *Pixel* is the term most widely used to denote the elements of a digital image [44]. In this sense, for each pixel, the imaging device records a number, or a small set of numbers, that describe any of its properties, such as brightness (the intensity of the light) or color. The numbers are arranged in an array of rows and columns that correspond to the vertical and horizontal positions of the pixels in the image. This way, as a digital image is intrinsically a numerical matrix, it is possible to apply several operations and mathematical transformations to it. Figure 5 presents a gray-scale image and the values of the pixels that compose it.

Figure 5 – A digital image with its corresponding pixel matrix representation.

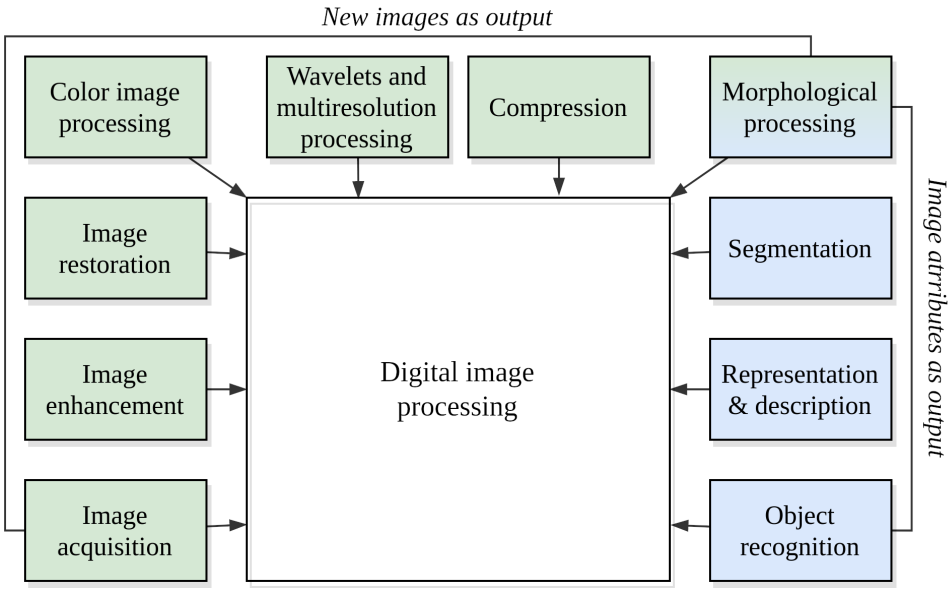


Source: Extracted from 45.

There is no general agreement regarding where image processing stops and other related areas, such as image analysis and computer vision, starts. However, one useful paradigm is to consider three types of computerized processes in this continuum: low-, mid-, and high-level

processes. Low-level processes involve primitive operations such as image preprocessing to reduce noise, contrast enhancement, and image sharpening. A low-level process is characterized by the fact that both its inputs and outputs are images. Mid-level processing on images involves tasks such as segmentation (partitioning an image into regions or objects), description of those objects to reduce them to a form suitable for computer processing, and classification (recognition) of individual objects. A mid-level process is characterized by the fact that its inputs generally are images, but its outputs are attributes extracted from those images (e.g., edges, contours, and the identity of individual objects). Finally, higher-level processing involves "making sense" of an ensemble of recognized objects, as in image analysis, and, at the far end of the continuum, performing the cognitive functions normally associated with vision. Figure 6 shows the principal areas regarding digital image processing.

Figure 6 – Block diagram of the main areas of digital image processing. Such stages can be divided into two large groups according to their generated output: new images or attributes extracted from the source image.



Source: Adapted from 44.

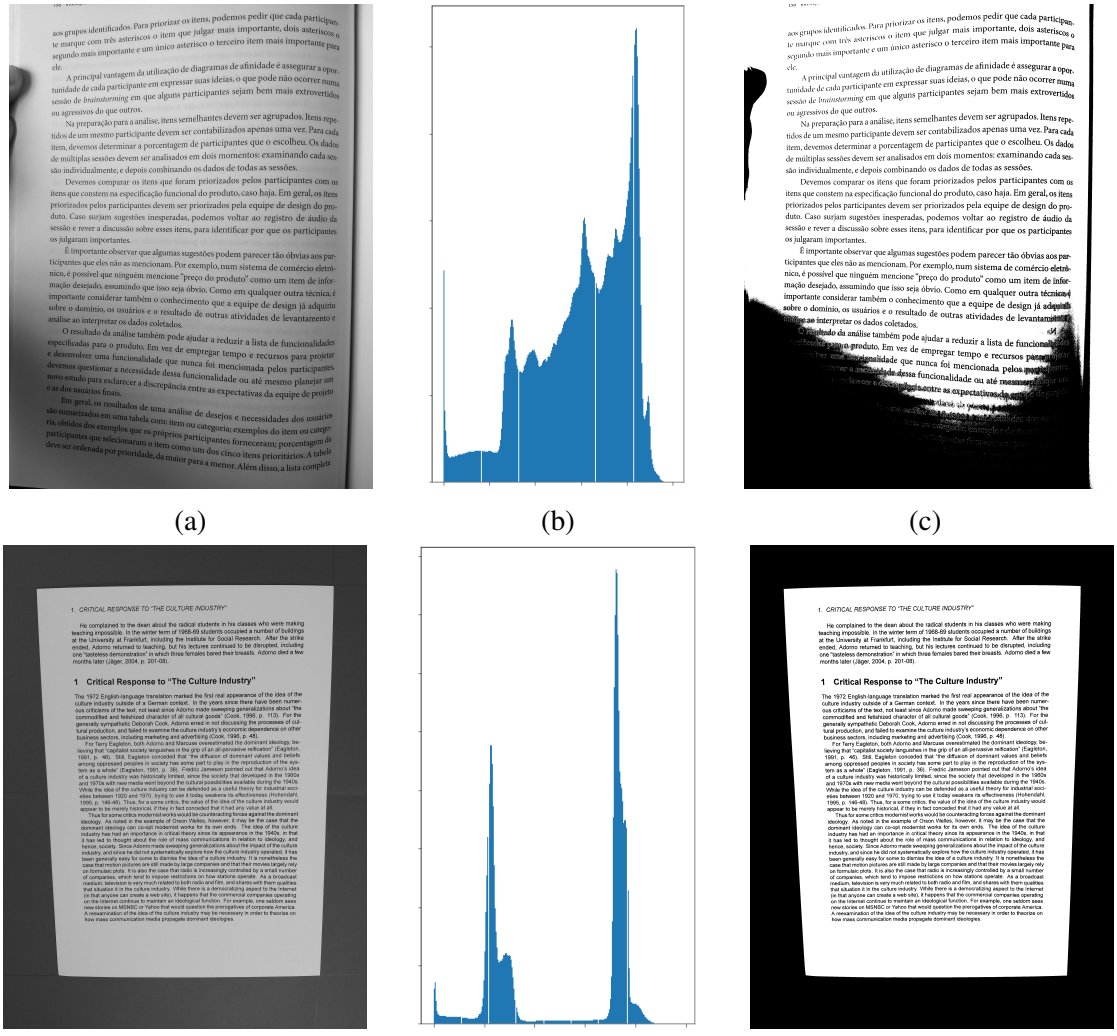
Today, there is almost no area of technical endeavor that is not impacted in some way by digital image processing [46]. In this sense, image processing techniques are used in various applications, such as: i) military operations, by detecting enemy soldiers or vehicles and missile guidance; ii) autonomous vehicles, by detecting obstacles, pedestrians and other vehicles; and iii) medicine, by detecting malign changes in image data, such as tumours and heart diseases, to diagnose patients [47].

3.3.1 Binarization

Image binarization is a fundamental research theme in image processing and an important preprocessing method in optical character recognition and edge/boundary detection. As a

image segmentation subarea, the purpose of binarization is to extract the outlines of different regions in the document made up of pixels that have common attributes, such as brightness or colour, indicating that they might belong to the same object. In this sense, binarization is responsible to convert gray images in a bi-level form in such way that the foreground information is represented by black pixels and the background by white ones, or vice versa [48].

Figure 7 – Results of a global thresholding binarization applied on two images. Note how Figure 7a obtained a suboptimal result due to shadow propagation at the bottom of the page while Figure 7d was well segmented. The histograms shown in Figures 7b and 7e endorse the correlation between the precision of the global thresholding method and the bi-modal characteristic of the images.



(a) (b) (c)

(d) (e) (f)

Source: Author.

The simplest way to acquire a binary image is through a technique called global thresholding, where one threshold value obtained from heuristics or statistics of a image attribute is selected for the entire document. Global thresholding is based on the assumption that the image has a bimodal histogram and, therefore, the object can be extracted from the background by a

simple operation that compares image values with a fixed threshold T [49]. For a gray value image $I(x, y)$ with intensity values between 0 and 1 and a threshold T each image pixel is classified in foreground (labeled as 1) and background (labeled as 0) resulting in the binary image $I_{th}(x, y)$, as shown in Equation 1:

$$I_{th}(x, y) = \begin{cases} 1 \leftrightarrow (x, y) > T \\ 0 \leftrightarrow (x, y) \leqslant T \end{cases} \tag{1}$$

Figure 7 shows that when the background has non-uniform illumination or noise, or the image has a non-bimodal histogram, or the input image has low quality, a fixed threshold value will poorly segment the image [50]. Thus, a local threshold value that changes dynamically over the document might obtain better results.

In this sense, local or adaptive thresholding methods compute a threshold for each pixel in the image on the basis of the content in its neighborhood. Adaptive methods have the characteristic that the value of $T(x, y)$ depends on the local region $R(x, y)$ properties, such as the local mean or standard deviation values [51]. Hence, as opposed to global thresholding, local methods generally perform better for low quality images, but are computationally more expensive [52].

During the last decades, several approaches to both global and local threshold value estimation where presented [48]. Otsu's thresholding method involves iterating through all the possible threshold values and calculating a measure of spread (interclass variation) for the pixel levels each side of the threshold, i.e. the pixels that either fall in foreground or background. In other words, the method's purpose is to find the threshold value where the sum of foreground and background spreads is at its minimum [53].

3.3.2 Denoising

In image analysis, the extraction of information can be significantly altered by the presence of random distortions called noise [54]. The principal sources of noise arise during image acquisition (digitization) and/or transmission, due to the fact that the performance of imaging sensors is affected by a variety of factors, such as environmental conditions during image acquisition and the quality of the sensing elements themselves [44].

The growing need to develop automated systems for image interpretation, such as OCR engines, necessitates that the input to be interpreted should be free from noise and other aberrations. Thus, it is important to perform preprocessing operations on the document so that the resultant image is better suited for the target system [55]. In this sense, as an image enhancement subarea, the ultimate goal of denoising (or smoothing) techniques is to improve an image in some predefined sense by accentuating its features, such as contrast, boundaries, edges, and so on.

The types of noise can be divided into two major classes: multiplicative and additive [46]. An image with variable illumination, the graininess noise from photographic plates and the speckle noise from the imaging systems as in ultrasound imaging are multiplicative in nature. For a digital image function $f(x, y)$ and a multiplicative noise $n(x, y)$, the resulting corrupted image $g(x, y)$ can be modelled as follow:

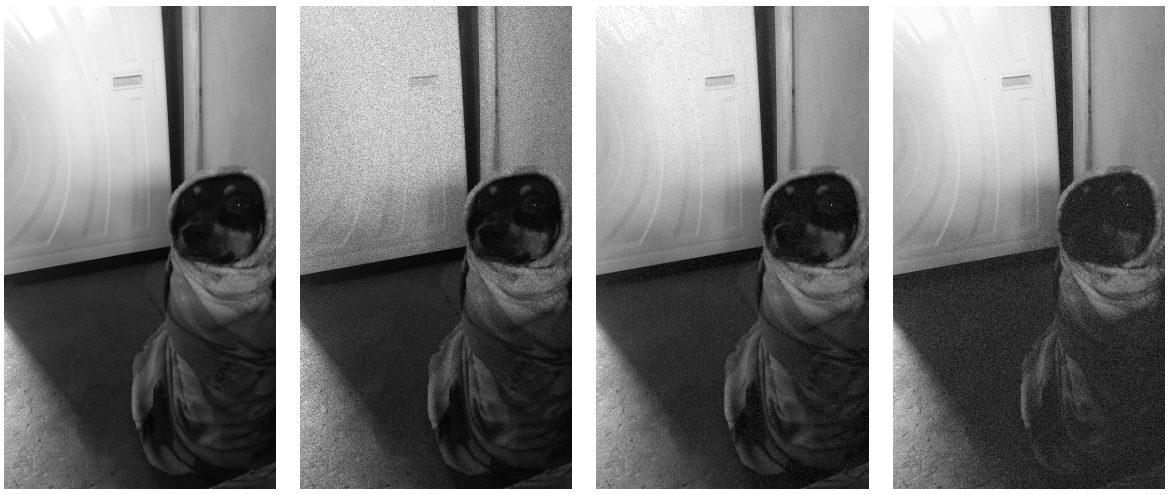
$$g(x, y) = f(x, y) \cdot n(x, y). \tag{2}$$

Additive noise is often assumed to be impulse or *Gaussian* noise. The first, also known as salt-and-pepper noise because of its appearance as white and black dots superimposed on images, alters at random the value of some pixels. Occasionally, the noise generated from digital or analog image transmission system is impulsive in nature, and can be modeled as shown in Equation 3, where $i(x, y)$ is the impulsive noise and p is a binary parameter that assumes the values of either 0 or 1 [55].

$$g(x, y) = (1 - p)f(x, y) + p \cdot i(x, y) \tag{3}$$

Instead of modifing the value of random pixels, additive Gaussian noise means that a value drawn from a zero-mean Gaussian probability density function is added to the true value of every pixel in the image. Figure 8 shows a image corrupted with multiplicative and both impulse and Gaussian noise.

Figure 8 – A sample image corrupted with different types of noise.



(a) Original image. (b) Multiplicative noise. (c) Impulse noise. (d) Gaussian noise.

Source: Author.

A variety of methods for images noise reduction has been developed so far. Most of them successfully remove noise but their edge preserving capabilities are weak [56]. Similarly

to threshold binarization, most denoising algorithms are based on filtering methods applied to the entire image. However, its also possible to perform image denoising using an average or median value of a group of pixels, also called *kernel* (similar to local threshold regions), that moves through the image [57].

Image smoothing (or blurring) is the process of removing different types of noise from a image. In the last decades, several image blurring methods where proposed, such as low-pass filtering. In this method, image denoising is performed by convolving a image with a low-pass filter (LPF) kernel in order to remove high frequency content from the document, such as noise. However, some filters also remove useful information from the image buried in these high frequencies [58].

It is possible to apply several LPF kernels to smooth an image. Median and mean filters generally obtain good results with impulse noise, while Gaussian and bilateral filters are the most suitable for Gaussian noise.

Mean and median blur is achieved by convolving the source image with a normalized box filter, which takes the mean or median value of all the pixels under the kernel area and replaces the central element. By replacing the value of every pixel in an image by the resulting value of the gray levels in the neighborhood defined by the filter mask, this process leads into an image with reduced "sharp" transitions.

Impulse noise filters have the side effect of also blurring edges, a feature that is generally important in some image analysis tasks such as object recognition. In addition, these methods cannot remove Gaussian noise or a mixture of Gaussian and impulse noise. The reason is that Gaussian noise corrupts each pixel in the image, and therefore it is difficult for these methods to find enough original pixels to be able to estimate the noisy ones [59].

Guassian blur consists in convolving the image with a Gaussian function in order to calculate the resulting value of each pixel in the image. The Gaussian distribution of a two dimensional matrix is calculated as follow:

$$G(x, y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{x^2+y^2}{2\sigma^2}}, \tag{4}$$

where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution. Once a suitable kernel has been calculated, the Gaussian smoothing can be performed using standard convolution methods. It is important to note that this method also preserve boundaries and edges better than other uniform kernel filters [60].

In Gaussian smoothing, the pixel value is calculated by the weighted average of the neighborhood, which are inversely proportional to the distance from the center of the kernel. Besides these spatial weights, the bilateral filter calculate a weighted sum of the pixels in a

local neighborhood that weights depends on both spatial and intensity (or tonal) distance [61]. The tonal weighting makes the bilateral filter capable of preserving edges (large differences in tonal value) while smoothing in the more flat regions (small tonal differences). However, since it does not have a linear implementation, the bilateral filter has the disadvantage of being harder to compute, with a longer processing time compared to other filters [62].

3.3.3 Dewarping

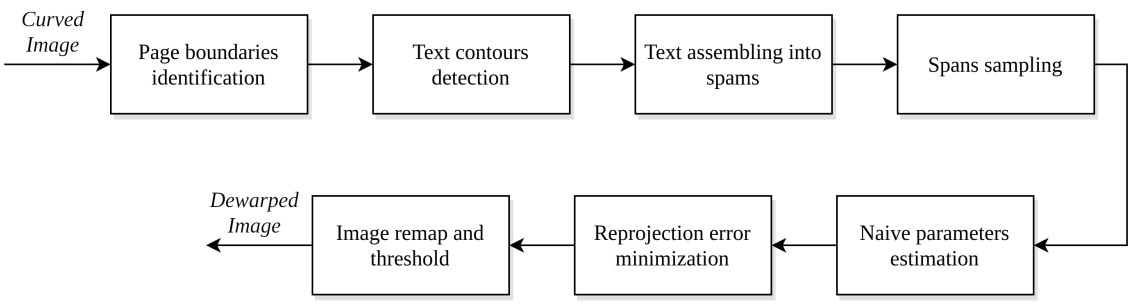
One of the main research directions in camera-based document analysis (CBDA) is to deal with warped images and perspective distortions. Current document analysis and OCR systems do not expect these types of artifacts, and generally show poor results when applied directly to camera-captured documents [63]. In this sense, the goal of image dewarping is to adjust the text lines horizontally by flattening images of curled pages, in order to facilitate the identification of text outlines by OCR and document-analysis systems.

If a thick book is scanned, text lines on the scan may be wrapped near the spine of the book. Also, if a digital camera is used to retrieve the image instead of a scanner, the text lines may be still wrapped due to several possible causes, such as: i) perspective distortions generated by optics of digital cameras; ii) the barrel (fish-eye) effect caused by the finite distance between the camera and the page; and iii) the local curvature of the page in to the image plane [64, 65].

Despite the existence of so many approaches for performing the dewarping of documents, there is no comparative evaluation so far. Still, few algorithms and models that perform correction of such documents are publicly available, making the treatment of camera-based images a more arduous task.

The script described in [66] uses a parametric model to reproject the detected keypoints on horizontal text spans. Starting with a initial guest, the model tries to find the rotation r and translation t vector, the α and β slopes and the horizontal x and vertical y offsets values which minimize the reprojection error of the keypoints. Figure 9 ilustrate the process performed by the script in order to dewarp the input document.

Figure 9 – Page dewarping block diagram.



Source: Author.

Using a fixed horizontal and vertical margin value, the script trims the edges of the

image in order to remove unecessary information that may have been captured by the camera. In the text countours detection step, the image is binarized with an inverse adaptive threshold (background as black and foreground as white), followed by a morphological dilatation and erosion to connect adjacent mask pixels and eliminate single-pixel-high "blips". Morphological transformations apply the same kernel and convolution principles as blurring techniques in order to enhance (dilatation) or degrade (erosion) image features. Finally, similarly to the Tesseract engine, the script performs a connected component analysis to determine the horizontal text contours.

Then, the detected text contours are combined to generate the horizontal page spans. This process consists of generating candidate edges for every pair of text contours and assigning a weight to the connection in order to assemble nearby countours into the same span. Because the parametric model needs discrete keypoints, it is necessary to generate a small number of representative points on each span. The spans sampling step is responsible for choosing one keypoint per twenty pixels of text contour.

With the spans determined, the naive estimation of the parameters is performed by simply calculating the mean orientation of all spans using principal component analysis followed by the reprojection of the keypoints into the image plane.

Finally, it is necessary to minimize the reprojection error previously estimated. To do so, the script uses a minimizer function available at a Python library, called Scipy, as a black-box optmizer. It is also important to note that this step is the principal bottleneck of the script, representing $\sim 99\%$ of the entire execution time.

The dewarped image is calculated by establishing a coordinate transformation with the obtained r, t, α and β parameters. The script projects the page points into a three-dimensional plan and perform a remmaping by transposing the resulting image coordinates to it. The final output is generated by binarizing the resulting page with an adaptive threshold mask.

3.4 Spelling Correction

As computers are used in an ever-widening variety of lexical processing tasks, the problem of error detection and correction becomes more critical [67]. Computer-aided spelling correction has been the subject of investigation since the 1960s. Research has focused on several different areas, from pattern matching algorithms and dictionary searching techniques to OCR systems, to context-dependent text analysis that seeks to identify and correct even real-word spelling mistakes [68].

Today, spelling correction is important for many of the potential NLP applications such as text summarization, plagiarism checking, sentiment analysis and machine translation [69]. Automatic spelling correction is also crucial in search engines as spelling mistakes are very

common in user-generated text. As example, many websites have a feature of automatically giving correct suggestions to the misspelled user queries [70].

In a relentless effort to find a way to better detect and correct misspelled words in text, researchers conceived the dictionary-based error correction methodology, also known as lexical error correction. This approach identifies words not belonging to the dictionary and weights the correction candidates accordingly to a distance metric. Such metric generally represents the number of character-level transformations necessary to convert the misspelled word into the correct one. In this sense, the correction candidate with the lowest distance with respect to the misspelled word is the best to fit as a correction [2].

Although dictionary-based error correction techniques are easy to implement and use, they still have various limitations and drawbacks. The first limitation is that it requires a wide-ranging dictionary that covers as many as possible words in the language. This dictionary should encompass proper nouns, names of countries and locations, scientific terminologies, and technical keywords. The second limitation is that regular dictionaries normally target a single specific language and thus they cannot support multiple languages simultaneously. Finally, the dictionary-based approach tries to look at the misspelled word in isolation, in a sense that it does not take into consideration the context in which the error has occurred.

A common situation encountered during proofreading using the lexical-based approach is to find several possible candidates with equal distance metric. For example, if the word “cafa” was wrong recognized by an OCR system in a Portuguese text, there are many possible candidates, such as “casa”, “cara”, “cama” and “capa”. Since selecting one of these candidates at random seems like a very lazy method, a frequency-based correction of the words usually obtains better results in these situations.

Table 3 – Most frequent words in both English and Portuguese mono and bigram frequency dictionaries.

English		**Brazilian Portuguese**	
Monogram	Bigram	Monogram	Bigram
the	of the	que	que o
of	in the	não	o que
and	to the	o	com o
to	on the	de	com a
in	for the	a	para o

Frequency-based correction requires a dictionary built from a robust text corpora. Unlike a common dictionary, the frequency dictionary also contains the number of occurrences within the corpora assigned to each word. The frequency-based correction method consider the likelihood of a suggestion based on the frequency of its occurrence in the provided dictionary. Thus, in situations where multiple candidates are selected, the spell-checker chooses the most frequent

word within the dictionary. In addition, frequency-based correction can also use bigrams (frequency of two words appearing together), trigrams and so on to segment and compound words. Table 3 shows the most frequent words in a English and Portuguese mono/bigram dictionaries available on the Internet[3,4].

Still, the task of spelling correction is challenging for resource-scarce languages [70]. This is due to the fact that to obtain a very representative dictionary, a large amount of text from a given language is required. Since most acoustic and textual resources are more easily found for languages like English, generating robust spelling correction system for other languages becomes a hard challenge.

3.5 Raspberry Pi 3

Raspberry Pi is a series of small single-board computers (SBC) developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom. As an embedded platform, The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse.

Besides its main operating system, Pi OS, the Raspberry Pi supports a variety of Linux distributions, such as Raspbian (a Debian-based distro), Windows IoT Core, Ubuntu and Arch Linux. Such flexibility has made Raspberry Pi a widely adopted tool for carrying out various tasks in the areas of home automation, robotics and other hardware-bases projects, education and edge computing.

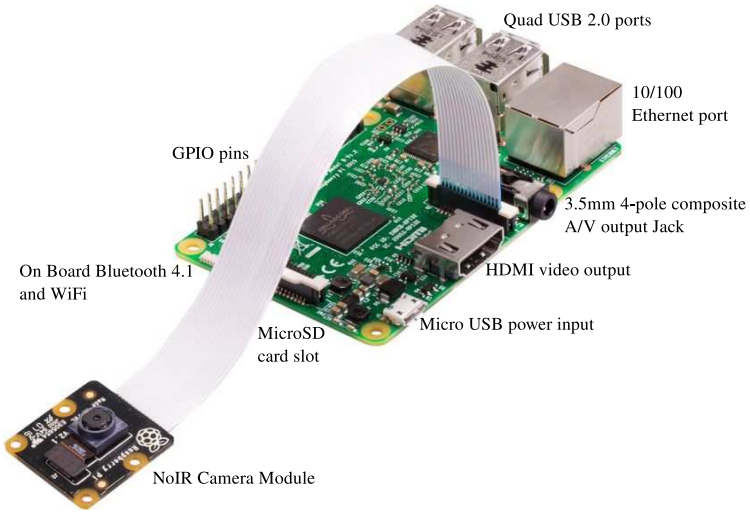
The Raspberry Pi 3 Model B is the earliest model of the third-generation Raspberry Pi, replacing Pi 2 Model B in February 2016. The third generation received an upgrade over its predecessor as the CPU increased in frequency from 900 MHz to 1.2 GHz, while the old 32-bit architecture was upgraded to 64-bit.

Although there are different types of microcomputers and micro components that could be used in the construction of computational systems with different purposes, such as Arduino, Banana Pi, Pc Duino, UDOO, etc. The Raspberry Pi has characteristics that make it a top plate in its functionality, given its hardware that allows: i) fast data transfer; ii) ports for external hardware; iii) internet connection via Ethernet or Wi-Fi cable; and iv) video output (HDMI) and audio (mini-jack) [72]. Also, the Raspberry has several external modules that can increase the functionality of the system, such as: heartbeat, temperature and motion sensors, cameras, GPS receiver and joysticks.

Shortly after the release of the new version of the Raspberry Pi, the infrared Camera Module v2 (Pi NoIR) replaced the original PiNoIR Camera Module in April 2016 as the official "night vision" camera board. The Raspberry Pi NoIR Camera Module v2 is a 8 megapixel Sony

[3] https://github.com/wolfgarbe/SymSpell/tree/master/SymSpell
[4] https://github.com/hermitdave/FrequencyWords/tree/master/content/2018/pt_br

Figure 10 – Main components of the Raspberry Pi 3 board.



Source: Adapted from 71.

IMX219 image sensor custom designed add-on board for the Raspberry Pi, featuring a fixed focus lens.

Such module attaches to the Raspberry Pi by way of one of the small sockets on the board upper surface and uses the dedicated Camera Serial Interface (CSI), designed especially for interfacing to cameras, as shown in Figure 10. It's capable of capturing 3280 x 2464 pixel static images, and also supports 1080p30, 720p60 and640x480p60/90 video recording. Finally, the NoIR Camera has a No InfraRed (NoIR) filter on the lens which improves its representativeness when performing infrared photography and taking pictures in low light environments.

3.6 Evaluation Metrics

In order to evaluate the correctness of the Book2Speech system, three metrics were considered: *Character Error Rate* (CER), *Word Error Rate* (WER) and *Word Information Lost* (WIL) [73]. Such metrics, although generally used in the evaluation of automatic speech recognition (ASR) systems, can be applied in any context where a text-to-text comparison is required, usually between a reference text (also referred as "ground truth") and the hypothesis.

CER is based on the Levenshtein distance, which represent the minimum number of edit operations (character insertions, deletions, and substitutions) needed to correct the hypothesis to match it with the ground truth [74, 75]. Let H_c, S_c, D_c and I_c denote the total number of character hits, substitutions, deletions and insertions and N_c the total number of input characters. CER is defined as:

$$CER = \frac{S_c + D_c + I_c}{H_c + S_c + D_c + I_c} = 1 - \frac{H_c}{N_c} \tag{5}$$

Similarly, the *WER* considers the number of word misrecognitions in the hypothesis text, normalized by the word-length of the reference text. Such metric is sometimes criticized for not having an upper bound, that is, the error rates for a given recognition can go beyond 1. Notably, WER also does not consider whether some words may be more important to the meaning of the message or whether some words might be more predictable than others in a text [76]. To calculate the WER between two texts, consider the same variables from Equation 5 but at a word-instead of character-level. In this sense, the WER is calculated as follow, in Equation 6:

$$WER = \frac{S_w + D_w + I_w}{N_w = H_w + S_w + D_w} \tag{6}$$

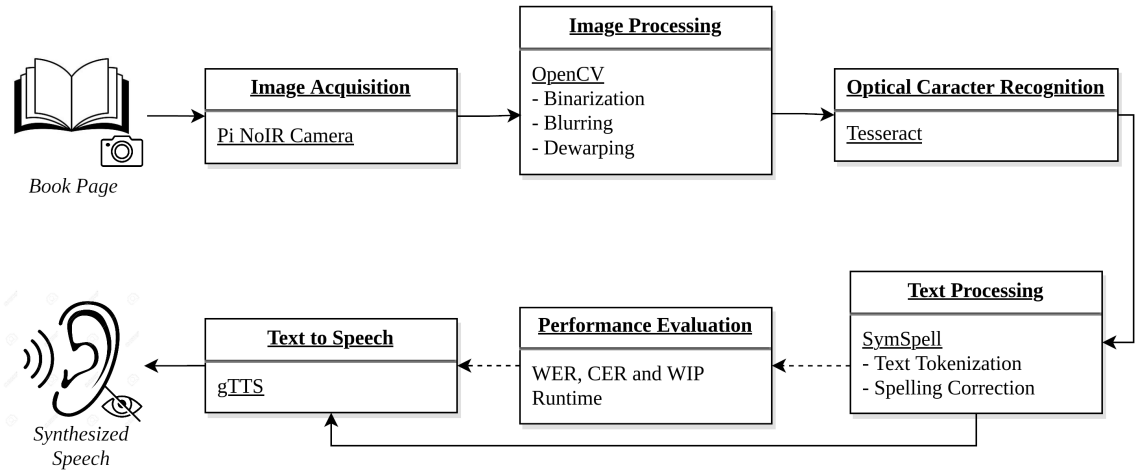
The *WIL* metric does not consider the edit operations necessary to correct the hypothesis text. Instead, it only takes account of the correct matches H squared proportion divided by the size (number of words) of both ground truth and hypothesis, S_{truth} and $S_{hypothesis}$, as shown in Equation 7:

$$WIL = \frac{H_w^2}{S_{truth} \cdot S_{hypothesis}} \tag{7}$$

4 METHODOLOGY

Book2Speech was built with the idea of an accessible tool for digitizing and reproducing the textual content of books through a voice signal for people with visual disabilities. The Raspberry Pi 3 plays the role of the system's main hardware, which captures the images through its camera module and reproduces the voice signal through its P2 Jack audio output. The pipeline of the prototype developed in this work is shown in Figure 11.

Figure 11 – Book2Speech modules.



Source: Author.

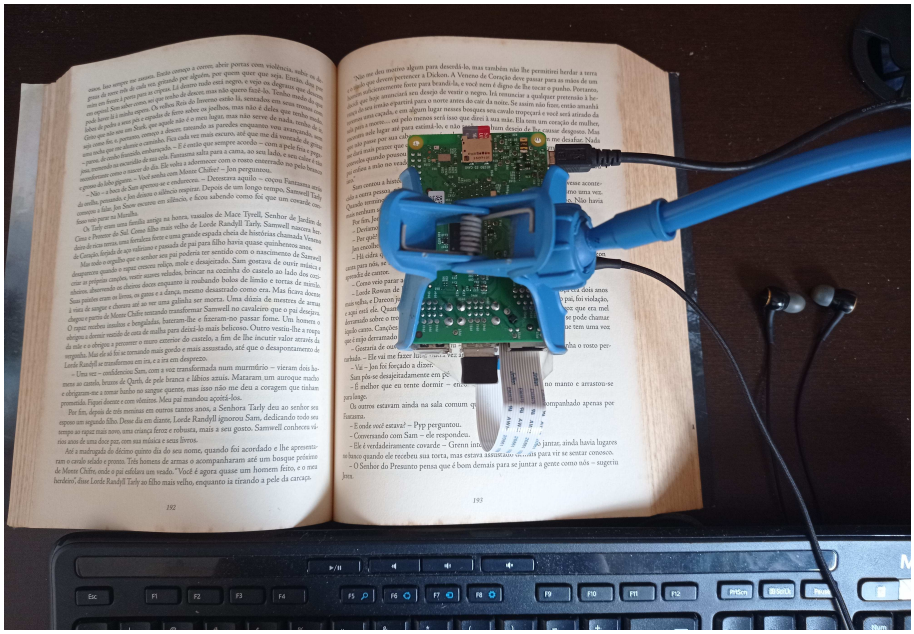
Even though it has a direct pipeline, the prototype has a highly customizable command line interface written in Python 3, providing the more advanced user with several hyperparameters to enable, disable and customize most of the modules present in this work. Appendix A presents a list with all available Book2Speech command line parameters and flags. This work was also developed with the intention of being open-source and low-cost, granting access to an alternative method to acquire the content of books that do not have a Braille version available.

From image acquisition to speech synthesis, each task has its specific module within the prototype implementation. This makes it possible to test and use the modules individually, helping in the process of developing, maintaining and updating the system. For better understanding, this chapter was subdivided according to the Book2Speech modules: i) image acquisition; ii) image processing; iii) optical character recognition; iv) text processing; and v) text-to-speech. Still, Book2Speech also possess a performance evaluation module, responsible for calculating the obtained error rate metrics as well as the pre- and post-processing runtime.

4.1 Image Acquisition

Book2Speech is controlled through a keyboard connected via USB, through which the user performs the image capture command by pressing a single key. The prototype was designed to be vertically parallel to a horizontal plane with the support of a tripod or a flexible phone holder where the target books are simply placed below the prototype in the direction of the NoIR Camera to be captured. Figure 12 shows Book2Speech ready to perform a image capture.

Figure 12 – Book2Speech supported by a cell phone holder above the book to be scanned. The system waits for the capture or cancel command to perform an action.



Source: Author.

OpenCV is the library responsible for capturing the images through the Pi NoIR Camera and pre-processing them before they are analyzed by the OCR module. As it is not a wrapper, this module is very easy to use within Python. Also, it does not require the C++ version of the library to be installed on the system, increasing the processing speed.

Such library provides some methods to access any camera that is connected to the computer, like webcams, bluetooth cameras, and so on. This way, the system starts connecting to the NoIR camera and stays in a waiting state until the user types the “s” key (for shot) when he wants to capture a new image, or the “q” key (for quit) to exit the program. Once the image is acquired, its information is transformed into a matrix of gray-scale pixels and then forwarded to the next module.

Since the prototype is ideally located vertically parallel to a horizontal plane, such as a table, the user only needs to position the book below the camera to perform the reading, and if desired, change the pages to perform new readings without much effort. Although not directly part of the main scope, the user also has the option of providing the path from a previously

captured image, a directory with several images or a text file with the path of various image files so that multiple readings can be performed in sequence.

4.2 Image Processing

The goal of the image processing module is to improve the content of the captured image before it is recognized by the Tesseract engine. To evaluate the performance of the OCR system under different scenarios, several image processing algorithms were implemented. In this sense, blurring and thresholding binarization functions were implemented using the methods provided by the OpenCV library, as well as an improved version of the proposed script by [66] for image dewarping.

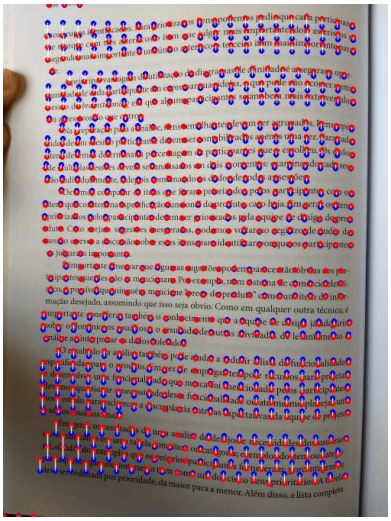
The script for dewarp images, originally written in Python 2, has been adapted to work with more recent versions of the programming language. As mentioned in Section 3.3.3, its optimization step accounted for over 99% of the execution time within the script, taking an average of 24 seconds for image correction. It was found that this processing delay was due to two main factors: i) the script considered many points as text countours, generating many parameters to be optimized; and ii) the default optimization algorithm, Powell [77], took more time than others available within the Scipy minimization function, such as the L-BFGS-B algorithm [78].

To work around this problem, the value of the parameter `TEXT_MAX_THICKNESS`, which represents the max reduced pixel thickness of detected text contour was reduced from 10 to 6. This way, fewer text contours were considered in the optimization process, reducing the module's execution time by about 70%. The other significant change was to make the optimizer algorithm changeable through the command line, helping in the evaluation of its performance. It is also possible to disable the optimization sub-step, using only the parameters of the initial guess (lazy optimization) to perform the curvature correction of the image text. Figure 13 shows the keypoints estimation output before and after reprojection, as well as the resulting dewarped and binarized image.

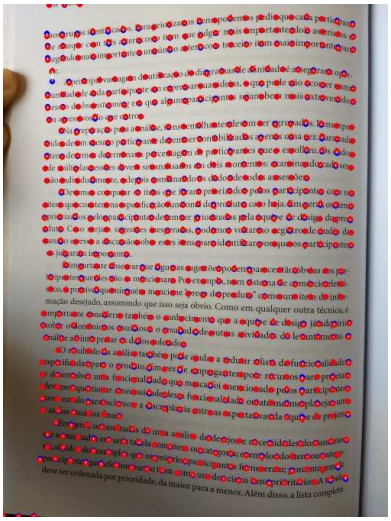
At this point, it is important to note that embedded operating systems like the Raspberry Pi 3 are simplifications of more robust systems, having most of their functionality reduced to fit a smaller platform. So, since the conception of the idea of this work we believe that the system as a whole should have a reduced computational cost, given the specifications of the system where it would be deployed, but also in terms of user experience, who would not want to wait too long to get a feedback from the tool.

To perform the smoothing of the image, the algorithms mean, median, Gaussian and bilateral blur were implemented. Since the process of image acquisition, as well as the conditions of the environment where the image was captured, can generate several noises of different categories, we sought to implement algorithms that encompassed the most common cases of noise that could occur.

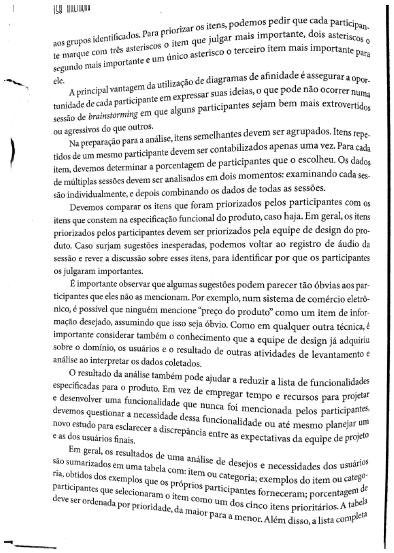
Figure 13 – The red points in Figure 13a and 13b are detected keypoints on text spans, and the blue ones are reprojections through the model. Note that the Figure 13b (final optimization output) has established the page pose/shape well enough to place almost all of the blue points on top of each corresponding red point. It is also remarkable how the binarized image with adaptive thresholding manages to segment the text much better compared to the global threshold shown in Figure 7c.



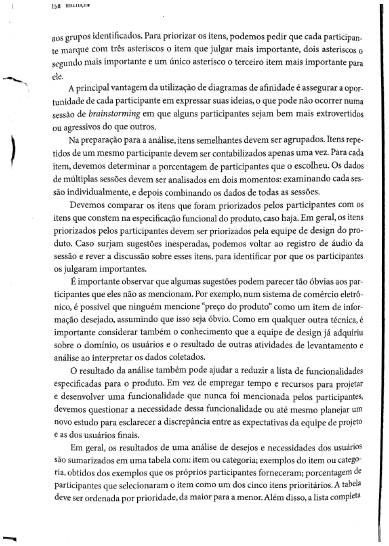
(a) Before reprojection.



(b) After reprojection.



(c) No optimization.



(d) Powell optimization.

Source: Author.

Finally, for the segmentation of the images, the global, Otsu, local average, and local Gaussian threshold algorithms were implemented. Since Tesseract already has an image binarization step, we would like to evaluate its performance with other types of thresholding. Furthermore, given the previously mentioned variables such as the light source and the shadows that can be projected onto the image, we intend to evaluate the performance of the algorithms under these conditions.

The result of the pre-processing step is, in theory, an improved image for the OCR task, since image features such as characters and page outlines will be enhanced, as well as the elimination of possible textlines distortions that might occur. Such a diversity of algorithms will help us in evaluating the accuracy of text recognition preceded by different image processing pipelines.

4.3 Optical Character Recognition

The OCR module is responsible for generating the text string given the provided image. This module can receive the original image that was captured, or the pre-processed version, depending on whether the image enhancement module was disabled or not. The Python API pytesseract is responsible for accessing the Tesseract engine within the Book2Speech's OCR module. It is necessary to previously install Tesseract and the language packages that the system will utilize. Furthermore, if the user wants to recognize non-English texts and has the target language package installed on the system, it is possible to change the Tesseract language via the Book2Speech CLI argument `--lang=LANGUAGE`.

A certain effort was spent looking for ways to speed up the recognition of the Tesseract, which took around 4 to 8 seconds per image. Changing the page segmentation method to that of previous versions produced poor results in the same way when using the recognition mode without the LSTM model. However, it was found that Tesseract performs an extra verification step to check if the input image is already binarized with the background as white and foreground as black. Such stage can be disabled within the pytesseract wrapper by passing the `tessedit_do_invert=0` parameter to the OCR function, which reduced the execution time in about 1.5 seconds.

4.4 Text Processing

When the OCR module fails to recognize a character, a recognition error is produced, commonly causing a spelling mistake in the output text. Such error makes it necessary to post-process the recognized text before the Text-to-Speech step. In this sense, the goal of spelling correction post-processing is to detect and correct linguistic misspellings in the OCR output text after the input image has been captured and processed [2].

Before the text can be corrected, it must undergo a process known as tokenization. In this sub-stage, the string resulting from the OCR module passes through a sequence of transformations, where multiple white spaces, punctuations and special characters such as '`\n`' and '`\t`' are removed. Then, the cleaned string is lowercased and transformed into a word vector. A modified version of the JiWER library [1] is used to tokenize the reference and hypothesis texts,

[1] https://github.com/jvcanavarro/jiwer

as well as calculating the accuracy metrics of the system.

To perform the frequency-based spelling correction, Book2Speech uses three methods implemented in the SymSpellPy library: direct lookup, compound and segmentation. The direct lookup method uses a frequency monogram to correct the words that have been identified as incorrect. It generates a list of the words closest to the incorrect one (fewer number of editions) up to a limit number of operations, and then replaces it with the most frequent one.

Instead of looking from a dictionary of correct words to figure out the closest match for a misspelled word, the SymSpell algorithm [79] flips the problem around by building a map of misspelled words to a probable list of correct words for up to a predetermined edit distance. This makes it extremely fast for lookup, at a cost of more computational memory required. In other words, SymSpell precomputes probable misspellings of the entire dictionary beforehand to make the lookup much faster.

The compound and segmentation methods perform the joining of word fragments that have been recognized apart, and the separation of words that have been incorrectly joined, respectively. For both, it is necessary that in addition to the monogram, a bigram dictionary is provided. Both methods are considerably slower compared to direct correction, either for loading the dictionaries and for the correction process itself, since they also perform the process of direct correcting words separately. Table 4 presents a correction example of each one of the methods implemented on the text-processing module.

Table 4 – Direct lookup, compound and segmentation spelling correction methods examples.

	Direct lookup	**Compound**	**Segmentation**
Original text	Dictiunary qualiti is paramoomt for corrctio qualty	Can yu readthis messa ge despite thehoribleˆsppelingmsitakes	Thequickbrownfoxjump soverthelazydog
Corrected text	Dictionary quality is paramount for correction quality	Can you read this message despite the horrible spelling mistakes	The quick brown fox jumps over the lazy dog

4.5 Text-to-Speech

The Text-to-Speech module is responsible for generating the speech signal referred to the text received from the previous step. The gTTS library is responsible for generating the speech signal from the text by communicating with the Google Translate's text-to-speech API. Unfortunately, Google does not provide a TTS engine for local installation, requiring an internet connection to use its Google Translator API to perform the speech synthesis procedure. Other TTS engines were also considered for this module, however either the resulting synthesized voice had a very low quality or the engine did not support many languages, like Brazilian Portuguese. Overall, this is the module that takes the longest to process.

Once the resulting audio is received by Book2Speech, it is played inside the system for the user. It is also possible to save the audio locally for later access.

5 EXPERIMENTS AND RESULTS

This chapter presents the results obtained with Book2Speech in recognizing the textual content of images. More specifically, evaluation metrics are used to measure the rate of difference between the text generated by Book2Speech and the reference text of the recognized image. In order to measure its accuracy, a sample of 100 images randomly extracted from the ICDAR2015 dataset was used, which is described in Subsection 5.1.

In the tests, the Character Error Rate (CER), Word Error Rate (WER) and Word Information Lost (WIL) metrics as well as the total program execution time from the image loading to the text processing module were calculated as forms of performance comparison.

In addition, it should be noted that, for each image, a single run was performed. This means that instead of passing a list of images in a directory or in a text file to the Book2Speech image acquisition module, each image was passed as a single entry. This was done so that the system would have to reload the OCR engine and the frequency-based dictionaries at each run, not changing the real processing time.

The testing methodology was divided into two parts. In the first, each module was evaluated individually, that is, each image processing algorithm as well as spelling correction methods were used alone to find which ones obtained the lowest error rates. In the second part, the algorithms and methods that achieved the best results in the first phase were combined with each other to form various processing pipelines. In this way, it would be possible to identify which techniques could achieve the best results together.

The scripts used to generate the results of both phases, as well as the results themselves and the list of images used in the tests are available at the Book2Speech repository on GitHub [10]. Finally, the mono- and bigram English dictionaries used in the spelling correction methods are also freely available for download at [79].

Unfortunately, given the current pandemic scenario that persists in Brazil, it was not possible to conduct tests with visually impaired people in order to evaluate the end-user experience and raise important points for improvement.

5.1 Evaluation Dataset

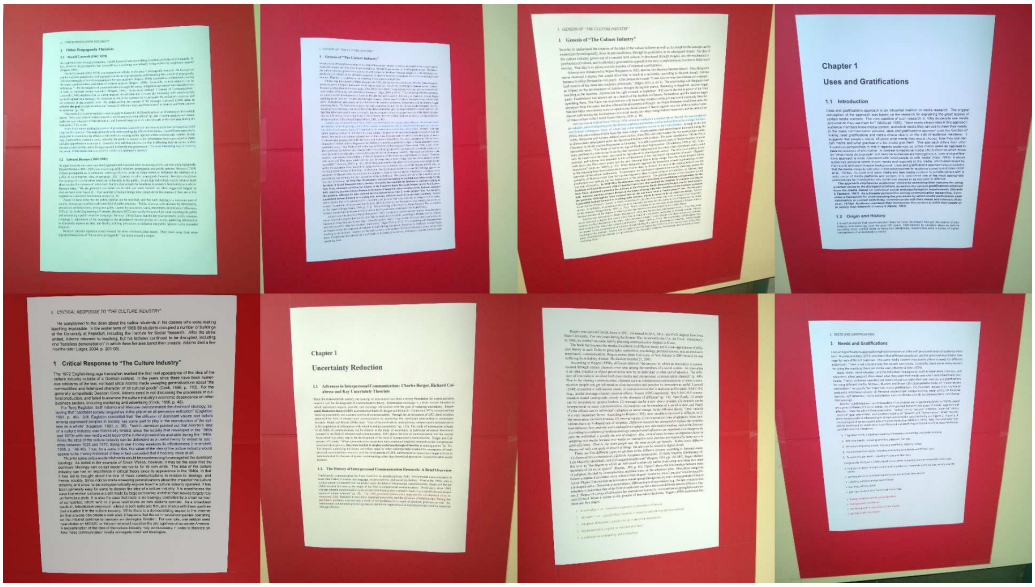
The absence of large-scale document image datasets is a serious problem that has an impact on the current state of research in the OCR area. To the best of our knowledge, there are no publicly available datasets containing images and their ground truth transcription in Brazilian Portuguese, and the hypothesis of producing a new dataset manually for PT-BR was discarded throughout the production of this work as generating such datasets is a time-consuming and

costly procedure [80].

Also, most of the document image datasets we found either did not fit the scenarios we would like to evaluate, because they consisted of images of cropped text lines instead of whole pages [81] or were in a format other than plain text, such as XML [82, 83], or were not publicly available [84].

The 13th International Conference on Document Analysis and Recognition document-image dataset (ICDAR2015) [85] was adopted to measure the accuracy of Book2Speech, since it is composed of images that well represent the most common inputs that the system would receive during personal use. The entire database was generated through smartphone captures under varying conditions (perspective angles, distances, light and blur), causing geometric and photometric distortions that hinder the OCR process. Figure 14 shows a sample composed by 8 images extracted from the referred dataset, presenting the notable distortions distinction between each image.

Figure 14 – Sample images extracted from the ICDAR dataset.



Source: Author.

The dataset has 12100 document images captured from 50 different paper documents with real content from wiki-books and cooking recipes from Internet. All documents contain single column text printed with multiple scales, fonts, font-faces and colors. Original content is mostly English, however some words, sentences and paragraphs have been randomly replaced with random text generated by *lorem* and also by other dictionary words. Finally, at least 240 different images are captured per document, those captures are taken using representative values of different distortions.

5.2 Individual Results

To generate the individual results for the first part of the tests, all the spelling correction and image processing algorithms available in the Book2Speech system were evaluated. In addition, the image dewarp method was evaluated with the standard optimizer (Powell), and with the L-BFGS-B algorithm, as well as without the optimization process, thus performing the text curvature correction only with the initial estimation of the page keypoints (lazy optimization). The L-BFGS-B algorithm was chosen as an alternative optimization method because it showed good results when compared to other algorithms with the same purpose in punctual evaluations performed before the present experiments.

Table 5 – Average error rates and runtime per individual method.

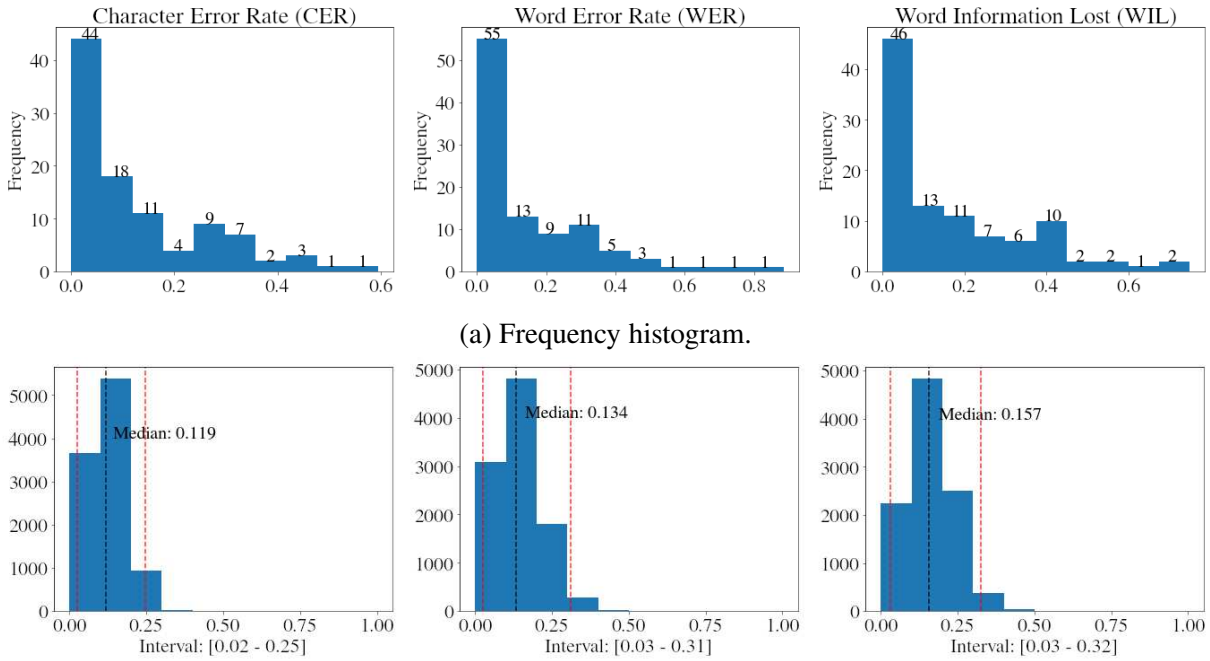
Method	**CER (%)**	**WER (%)**	**WIL (%)**	**Time (s)**
No processing (baseline)	22.62	18.53	23.46	4.41
Spellchecking (direct)	22.62	18.53	23.46	6.62
Spellchecking (compound)	24.82	20.49	26.76	10.25
Spellchecking (segmentation)	23.51	19.65	24.95	7.55
Threshold (global)	21.00	17.98	23.94	**3.77**
Threshold (otsu)	17.53	15.17	19.99	3.94
Threshold (mean)	21.59	17.52	21.67	9.36
Threshold (gaussian)	33.55	25.11	29.49	11.27
Blur (average)	50.02	42.60	46.00	4.16
Blur (median)	18.42	15.79	20.37	5.61
Blur (gaussian)	26.07	20.17	24.91	5.21
Blur (bilateral)	30.56	25.43	29.81	4.50
Dewarp (lazy)	15.16	13.26	17.28	5.73
Dewarp (powell)	14.88	12.76	16.59	10.71
Dewarp (l-bfgs-b)	**14.41**	**12.32**	**16.07**	6.43

Table 5 shows the average results obtained by each individual method, where the error rate values are in percentage and the time measurement in seconds. Also, the best results obtained, i.e. the lowest WER, CER, WIP and runtime are in bold.

The lowest error rates were achieved by the dewarping methods, being approximately 7% lower compared to the results without any processing technique, in all metrics. Compared to each other, they all generated a very similar error rate. However, the Powell optimization achieved an average run time 4 seconds slower than L-BFGS-B, while the second only performed about 2 seconds slower than “raw OCR” (recognition result without any pre- or post-processing method).

Still with regard to image dewarping, Figure 15 shows the frequency histograms as well as the confidance interval of each error metric generated by the dewarping method using the L-

Figure 15 – Histogram plots of the error rates generated by the L-BFGS-B dewarping method.



(a) Frequency histogram.

(b) Confidence interval.

BFGS-B as optmizer algorithm. Looking at the frequency of the first three bars of the histogram subplots, it is noticeable that, on average, the evaluated method obtained better results than raw OCR approximately 73% of the time for all metrics evaluated. Also, the presented histogram shows that the method was able to achieve an error rate less than or equal to 10% in more than half of the performed recognitions, according to the height of the first bar. The confidence interval shown above was generated from an iteration with 10,000 samples, with 95% confidence. In other words, the first confidence interval graph shows that with 95% certainty, given an image pre-processed with the evaluated method, it will achieve a character error rate between 3 and 25%, with the most likely error rate being around 12.5%.

It was noticed that during the recognition of some images, the dewarp module was not able to estimate the minimum number of spans needed to perform the pose estimation step. Since the dewarp implementation has a minimum size in pixels to consider a certain keypoint as text, it is believed that due to the camera-to-page distance in some scenarios, the text was captured with a very small size and thus any keypoint was considered. Hence, changing the minimum text size parameter needs to be evaluated further, since if the minimum value is reduced, more false-keypoints may be identified, leading to higher computational cost and lower accuracy.

Binarization with global thresholding achieved the shortest runtime, followed by Otsu thresholding. This is due to the fact that both algorithms have low complexity compared to the other adaptive segmentation methods. Overall, the other image processing methods were able to improve the results obtained by the OCR module without major compromises in the execution

time.

It is noticeable how the spelling correction methods ended up increasing the error rate of the system. On a more individual analysis, we can verify that this was due to the fact that many times a correct word was either changed because it was not found in the dictionary or wrongly segmented/compounded into another word. This scenario occurred mostly with proper names, names of companies, locations, and non-English words. For example, the Italian region "Calabrese" was erroneously replaced by "calaboose", just as the surname "Heider" was replaced by "header". Also, the time it took to load the dictionaries into the memory considerably increased the system's execution time, resulting in the compound spelling correction being the third slowest among the methods evaluated.

5.3 Combined Results

For the second phase of testing, only the methods that averaged better or very close to the results without processing (raw OCR) were considered. The evaluation process consisted of applying all possible combinations of two or three methods to the same 100 images that were used for producing the individual results. In total, 28 pipelines were evaluated. Thus, the spelling correction module as well as the adaptive Gaussian threshold and average/bilateral blur algorithms were ignored. Table 6 shows the average results obtained by the combined methods.

It is noticeable that none of the pipelines managed to obtain an error rate lower than that of the dewarp with the L-BFGS-B optimization algorithm in the previous stage (14.4% of WER). On the other hand, it is noticeable that applying binarization with the Otsu algorithm along with dewarp generates the best results (15.4% of WER), followed by smoothing the image with median blur, witch obtained an error rate about 3% higher. Finally, regarding execution time, global thresholding plus Gaussian blur achieved the shortest runtime among all tests.

Another notable point from the combined results was the high error rate presented by the pipelines with the global threshold. Even though this method achieved reasonable results in the first evaluation phase, when used in conjunction with other image processing algorithms, it led to a very high inaccuracy, averaging 40% in some cases. Because of this, it was decided to perform a few additional tests switching from the global to the local average threshold algorithm. Table 7 shows the results obtained by the pipelines applying local rather than global thresholding.

It is possible to notice that, in all cases, the error rate was reduced, especially in scenarios where the algorithm was applied together with the gaussian blur, having cases where the average error rate was approximately 25% lower compared to the previous binarization algorithm. In contrast to the higher accuracy, the preprocessing execution time increased, reaching more than 14 seconds in one pipeline, the longest execution time among all the methods considered.

Table 6 – Average error rates for each possible combination of the image processing algorithms that were implemented in Book2Speech that obtained satisfactory results in the previous test phase, along with the average execution time.

Method	**CER (%)**	**WER (%)**	**WIL (%)**	**Time (s)**
Thresh (global) + Blur (gaussian)	37.67	45.42	41.94	**3.11**
Thresh (global) + Blur (median)	26.78	33.13	32.81	3.54
Thresh (otsu) + Blur (gaussian)	23.04	28.59	28.96	3.58
Thresh (otsu) + Blur (median)	17.85	21.07	23.69	3.78
Thresh (global) + Dewarp (lazy)	17.37	20.27	23.43	3.98
Thresh (global) + Dewarp (powell)	16.99	19.85	22.54	9.25
Thresh (global) + Dewarp (l-bfgs-b)	16.24	19.32	21.79	4.68
Thresh (otsu) + Dewarp (lazy)	14.31	16.19	19.12	4.13
Thresh (otsu) + Dewarp (powell)	13.71	15.85	18.26	9.41
Thresh (otsu) + Dewarp (l-bfgs-b)	**13.27**	**15.46**	**17.69**	4.83
Blur (gaussian) + Dewarp (lazy)	20.40	25.17	25.39	4.83
Blur (gaussian) + Dewarp (powell)	19.60	24.46	23.99	10.11
Blur (gaussian) + Dewarp (l-bfgs-b)	19.39	24.41	23.86	5.55
Blur (median) + Dewarp (lazy)	16.33	18.88	20.93	5.20
Blur (median) + Dewarp (powell)	15.40	18.21	19.55	10.49
Blur (median) + Dewarp (l-bfgs-b)	15.06	17.94	19.07	5.90
Thresh (global) + Blur (gaussian) + Dewarp (lazy)	37.74	45.34	42.07	3.24
Thresh (global) + Blur (gaussian) + Dewarp (pow...	36.07	43.72	39.80	8.54
Thresh (global) + Blur (gaussian) + Dewarp (l-b...	36.54	44.36	40.21	3.95
Thresh (global) + Blur (median) + Dewarp (lazy)	26.74	33.04	32.94	3.71
Thresh (global) + Blur (median) + Dewarp (powell)	25.69	31.56	31.23	8.95
Thresh (global) + Blur (median) + Dewarp (l-bfg...	25.64	32.14	31.34	4.40
Thresh (otsu) + Blur (gaussian) + Dewarp (lazy)	24.17	29.44	30.64	3.73
Thresh (otsu) + Blur (gaussian) + Dewarp (powell)	23.00	28.18	28.75	9.00
Thresh (otsu) + Blur (gaussian) + Dewarp (l-bfg...	23.16	28.71	29.08	4.45
Thresh (otsu) + Blur (median) + Dewarp (lazy)	18.65	21.63	24.52	3.91
Thresh (otsu) + Blur (median) + Dewarp (powell)	17.60	20.76	22.96	9.20
Thresh (otsu) + Blur (median) + Dewarp (l-bfgs-b)	17.42	20.73	22.74	4.61

Overall, the results showed that even though some algorithms get good results individually, such methods are very sensitive when mixed together. For example, certain segmentation algorithms lead to poor results when applied in conjunction with binarization methods, and vice versa. However, it was possible to prove that even applying some image processing techniques, Book2Speech was able to obtain better results than raw OCR without much increases in increasing execution time.

Finally, it was also shown how the application of frequency-based spelling correction in the context of optical character recognition led to suboptimal results. However, other text correction techniques should be considered, since this is an essential post-processing step for an OCR-TTS pipeline.

Table 7 – Average error rates and runtime of mean adaptive threshold combined with other pre-processing methods.

Method	**CER (%)**	**WER (%)**	**WIL (%)**	**Time (s)**
Thresh (mean) + Blur (gaussian)	16.52	20.25	21.03	6.82
Thresh (mean) + Blur (median)	16.95	20.26	21.70	6.89
Thresh (mean) + Dewarp (lazy)	16.46	19.45	20.61	8.79
Thresh (mean) + Dewarp (powell)	16.41	19.61	20.15	14.03
Thresh (mean) + Dewarp (l-bfgs-b)	15.86	19.13	19.54	9.47
Thresh (mean) + Blur (gaussian) + Dewarp (lazy)	16.39	20.16	20.83	6.91
Thresh (mean) + Blur (gaussian) + Dewarp (powell)	15.58	19.39	19.57	12.19
Thresh (mean) + Blur (gaussian) + Dewarp (l-bfg...	15.90	19.93	19.87	7.61
Thresh (mean) + Blur (median) + Dewarp (lazy)	16.84	19.83	21.79	7.08
Thresh (mean) + Blur (median) + Dewarp (powell)	16.13	19.26	20.73	12.26
Thresh (mean) + Blur (median) + Dewarp (l-bfgs-b)	15.97	19.28	20.52	7.71

6 FINAL CONSIDERATIONS

This work presented a proposal for a low-cost machine for converting textual content into audible feedback via synthesized speech. Directed to people with partial or total visual impairment, this work aims to provide these users with access to the vast knowledge spread in physical form, such as books and derivatives, given that the availability of haptical written mechanisms such as Braille are not always plentiful. Initially conceived as a system built on a Raspberry Pi 3, this work can be deployed in any device with Linux or Mac OS installed, internet access, and support for image capture and sound reproduction. In adition, the work developed has an open-source proposal, having its entire system freely available on GitHub for download and contributions [10].

In general, Book2Speech image pre-processing methods leaded into better results in comparison to raw OCR procedures. In the tests performed, the average word error rate was reduced from 22.6% to 14.4% by dewarping the image using the L-BFGS-B algorithm as the optimizer method, while only increasing the execution time by approximately 2 seconds. The other error rate metrics considered were also reduced by approximately 7%. In the combined methods results, Book2Speech was able to achieve an average WER of 15.4% with Otsu thresholding plus L-BFGS-B dewarping, while also maintaining an average runtime of 4.82 seconds, only 0.4 seconds slower than pure OCR.

On the other hand, Book2Speech spelling correction methods maintained or increased the error rate of the system. This is mainly due to the fact that the quality of the correction is intrinsically related to the quality and representativeness of the dictionary used. The spell-checker often identified correct words as incorrect ones, these being mostly proper and institutional names (also referred as named entities), which were not part of the dictionary. Furthermore, since the images in the test database had words that did not belong to the English language, the spell-checker ended up replacing them erroneously.

Even as an initial effort, the proposed work proved to be a possible alternative to the current assistive technologies aimed at the visually impaired. Due to its low cost, it is expected that more people with low income or in underdeveloped countries will be able to have access to the developed system in their daily use, helping in the education and dissemination of knowledge.

It is hoped that in future versions the system will achieve greater accuracy in recognizing the text of the captured images. It is also expected that new processing modules for both image and text can be added to the system in order to make it more robust and prepared for the various environmental conditions that may occur during image acquisition. Finally, we believe that porting Book2Speech to be compatible with other operating systems and mobile architectures such as Android is an idea that will add even more value to the work proposed.

6.1 Future Work

We believe that a prior modification that should be made to the proposed work is to change the external control of Book2Speech, as it requires the use of the hands. The application of assistive technologies such as ASR to promote accessibility has been strongly adopted in recent years. Thus, we believe that a possible alternative to the current keyboard-based control would be through voice command, where Book2Speech could be used and configured only by the user's speech. Such an improvement would also be a differential in comparison to other works related to stand-alone reading systems, since all products aimed at automatic reading require the user to have a preserved upper motor skills. Another control alternative would be the use of external actuators, widely applied in ATs directed to people with impaired motor skills.

As an initial effort, the proposed work evaluated only the most common image processing methods in the context of OCR systems. The area of image processing is extremely vast and many other techniques for segmentation, improvement and smoothing of images must be considered. Some of these are: gamma correction, deskewing (text skew correction), thinning and morphological transformations [86].

Still with regard to image processing, we believe that further investigation into page dewarping methods, such as optimization algorithms, text detection and pose estimation techniques, as well as modification of the parameters of the used module are worth remarking on. Since image deformations and curved pages are intrinsic characteristics of book images captured from several devices, developing a system prepared to deal with these types of artifacts is a must. For example, adapting the Book2Speech camera so that it slides over the book, captures various images, and then stitches them together to form a panorama of the book's content should be a considered adaptation.

It was observed that the spelling correction methods used in this work did not produce good results. Thus, it is necessary to consider other correction techniques, especially those that consider the general context of the sentence to perform the replacements. In this sense, solutions in the area of natural language processing and machine learning should generate more satisfactory results. For example, using a n-grams or a Recurrent Neural Network (RNN) language model can be applied as general context analyzer.

Currently, Book2Speech needs an internet connection to produce the speech signal referring to the recognized text. In this sense, as future work we should consider other TTS engines that can be installed locally on the system so that it can work completely offline, removing the need for a stable connection and speeding up the speech synthesis process.

Regarding the overall implementation of the system, we believe that applying thread parallelization methods and translating the system source code to another compiled language, such as C++ or Haskell, should speed up the overall runtime of Book2Speech. Furthermore,

since the compiled file of the aforementioned languages is supported on any type of operating system, such a modification would turn Book2Speech into a multiplataform program.

Regarding the equipment used, we point out that using other more flexible methods to support Book2Speech may help with the portability aspect. The cell phone support used, despite being very extensive and easy to attach to tables, is not very flexible, making it difficult to adjust the angle and distance between the prototype and the plane. Another less important aspect would be to use one of the cases developed for the Raspberry Pi 3 board, because besides protecting the hardware, it helps to keep the camera fixed in a certain angle.

We also believe that some difficulties may also arise regarding the alignment of the book below the prototype camera, leading to poor recognition results. A possible solution to this problem would be to adapt a small platform below Book2Speech, delimiting the place where the camera is focusing, as done in other commercial and acamedic works, such as in [18, 23].

Still, research should also be done on other metrics aimed at evaluating the performance of OCR systems, such as those proposed in [75, 76, 87]. The performance of Book2Speech should also be evaluated in relation to the recognition of double pages, since such a procedure would speed up the book digitization process.

BIBLIOGRAPHY

1 HERSH, M.; JOHNSON, M. A. *Assistive technology for visually impaired and blind people*. [S.l.]: Springer Science & Business Media, 2010. Cited 2 times in pages 12 and 19.

2 BASSIL, Y.; ALWANI, M. Ocr post-processing error correction algorithm using google online spelling suggestion. *arXiv preprint arXiv:1204.0191*, 2012. Cited 5 times in pages 12, 21, 22, 34, and 42.

3 BRASIL. Tecnologia Assistiva: Comitê de Ajudas Técnicas. Subsecretaria Nacional de Promoção dos Direitos da Pessoa com Deficiência, CORDE., Brasília, Brasil, p. 138, 2009. Cited in page 12.

4 GELDERBLOM, G. J.; WITTE, L. P. de. The Assessment of Assistive Technology: Outcomes, Effects and Costs. In: . [S.l.]: IOS Press, 2002. v. 14, n. 3, p. 91–94. Cited in page 12.

5 UN. Convention on the Rights of Persons with Disabilities. *Treaty Series*, United Nations, v. 2515, December 2007. Disponível em: <http://www.un.org/disabilities/convention/conventionfull.shtml>. Cited in page 12.

6 WECHSUNG, I.; NAUMANN, A. B. Evaluating a Multimodal Remote Control: The Interplay Between User Experience and Usability. In: *Quality of Multimedia Experience, 2009. QoMEx 2009. International Workshop on*. [S.l.: s.n.], 2009. p. 19–22. Cited in page 13.

7 HAKOBYAN, L. et al. Mobile assistive technologies for the visually impaired. *Survey of ophthalmology*, Elsevier, v. 58, n. 6, p. 513–528, 2013. Cited in page 13.

8 MITHE, R.; INDALKAR, S.; DIVEKAR, N. Optical Character Recognition. *International journal of recent technology and engineering (IJRTE)*, Citeseer, v. 2, n. 1, p. 72–75, 2013. Cited 3 times in pages 13, 21, and 22.

9 TAYLOR, P. *Text-To-Speech Synthesis*. [S.l.]: Cambridge University Press, 2009. Cited 2 times in pages 13 and 24.

10 CANAVARRO, J. V. *Book2Speech*. February 2021. <https://github.com/jvcanavarro/Book2Speech>. Accessed on 14 May 2021. Cited 4 times in pages 13, 15, 45, and 52.

11 WORLD, D. *Disabled World: Disability News and Information*. <http://www.disabled-world.com/disability/>. Accessed on 20 April 2021. Cited in page 14.

12 ORGANIZATION, W. H. *Disability*. <https://www.who.int/health-topics/disability>. Accessed on 19 April 2021. Cited in page 14.

13 ORGANIZATION, W. H. et al. *WHO global disability action plan 2014-2021: Better health for all people with disability*. [S.l.]: World Health Organization, 2015. Cited in page 14.

14 ESTATíSTICA, I. B. de Geografia e. *Censo Demográfico*. 2010. <www.ibge.gov.br/home/estatistica/populacao/censo2010/>. Accessed on 7 April 2021. Cited in page 14.

15 LIAMBAS, C.; SARATZIDIS, M. Autonomous ocr dictating system for blind people. In: IEEE. *2016 IEEE Global Humanitarian Technology Conference (GHTC)*. [S.l.], 2016. p. 172–179. Cited 2 times in pages 14 and 19.

16 USLAN, M. Barriers to acquiring assistive technology: Cost and lack of information. *Journal of Visual Impairment & Blindness*, SAGE Publications Sage CA: Los Angeles, CA, v. 86, n. 9, p. 402–407, 1992. Cited in page 15.

17 ORCAM. *OrCam MyEye*. September 2018. <https://www.orcam.com/en/myeye2/>. Accessed on 15 May 2021. Cited in page 18.

18 ET, C. *CZUR ET Smart Book Scanner*. October 2018. <https://www.czur.com/product/et16plus>. Accessed on 15 May 2021. Cited 2 times in pages 18 and 54.

19 SCANMAKER. *Scanmarker Air -Black*. October 2021. <https://scanmarker.com/product/scanmarker-air/>. Accessed on 17 May 2021. Cited in page 18.

20 C-PEN. *ReaderPen: A reading pen from C-PEN*. October 2018. <https://cpen.com/product/readerpen/>. Accessed on 17 May 2021. Cited in page 18.

21 SHARMA, A.; SRIVASTAVA, A.; VASHISHTH, A. An assistive reading system for visually impaired using ocr and tts. *International Journal of Computer Applications*, Citeseer, v. 95, n. 2, 2014. Cited in page 18.

22 SHILKROT, R. et al. Fingerreader: a wearable device to explore printed text on the go. In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. [S.l.: s.n.], 2015. p. 2363–2372. Cited in page 19.

23 BOUAZIZI, I.; BOURISS, F.; SALIH-ALJ, Y. Arabic reading machine for visually impaired people using tts and ocr. In: IEEE. *2013 4th International Conference on Intelligent Systems, Modelling and Simulation*. [S.l.], 2013. p. 225–229. Cited 2 times in pages 19 and 54.

24 SONTH, S.; KALLIMANI, J. S. Ocr based facilitator for the visually challenged. In: IEEE. *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*. [S.l.], 2017. p. 1–7. Cited in page 19.

25 ARAVAMUTHAN, A. K.; R, G.; M, R. *A WEARABLE DEVICE TO ASSIST BLIND PEOPLE IN READING TEXT*. [S.l.], May 2017. Cited in page 20.

26 PATEL, C.; PATEL, A.; PATEL, D. Optical character recognition by open source OCR tool teseract: A case study. *International Journal of Computer Applications*, Electronic Publication: Digital Object Identifiers (DOIs), v. 55, n. 10, p. 50–56, 2012. Cited 2 times in pages 21 and 22.

27 SINGH, A.; BACCHUWAR, K.; BHASIN, A. A survey of OCR applications. *International Journal of Machine Learning and Computing*, IACSIT Press, v. 2, n. 3, p. 314, 2012. Cited in page 21.

28 GREANIAS, E. Some important factors in the practical utilization of optical character readers. *Optical Character Recognition*, McGregor & Werner, p. 129–146, 1962. Cited in page 21.

29 MORI, S.; SUEN, C. Y.; YAMAMOTO, K. Historical review of OCR research and development. *Proceedings of the IEEE*, Ieee, v. 80, n. 7, p. 1029–1058, 1992. Cited in page 21.

30 SINGH, S. Optical character recognition techniques: a survey. *Journal of emerging Trends in Computing and information Sciences*, Citeseer, v. 4, n. 6, p. 545–550, 2013. Cited in page 21.

31 BIENIECKI, W.; GRABOWSKI, S.; ROZENBERG, W. Image preprocessing for improving ocr accuracy. In: IEEE. *2007 international conference on perspective technologies and methods in MEMS design*. [S.l.], 2007. p. 75–80. Cited 2 times in pages 21 and 22.

32 SANTOS, E. A. OCR evaluation tools for the 21st century. In: *Proceedings of the Workshop on Computational Methods for Endangered Languages*. [S.l.: s.n.], 2019. v. 1. Cited in page 21.

33 NAZIR, S.; JAVED, A. Diacritics recognition based Urdu Nastalique OCR system. *The Nucleus*, v. 51, n. 3, p. 361–367, 2014. Cited in page 21.

34 SMITH, R. An overview of the Tesseract OCR engine. In: IEEE. *Ninth international conference on document analysis and recognition (ICDAR 2007)*. [S.l.], 2007. v. 2, p. 629–633. Cited in page 22.

35 SMITH, R.; ANTONOVA, D.; LEE, D.-S. Adapting the Tesseract open source OCR engine for multilingual OCR. In: *Proceedings of the International Workshop on Multilingual OCR*. [S.l.: s.n.], 2009. p. 1–8. Cited in page 23.

36 ZELIC, F.; ANUJ, S. *A comprehensive guide to OCR with Tesseract, OpenCV and Python*. February 2021. <https://nanonets.com/blog/ocr-with-tesseract/>. Accessed on 25 April 2021. Cited in page 23.

37 DUTOIT, T. *An introduction to text-to-speech synthesis*. [S.l.]: Springer Science & Business Media, 1997. Cited in page 24.

38 YOSHIMURA, T. Simultaneous modeling of phonetic and prosodic parameters, and characteristic conversion for HMM-based text-to-speech systems. *PhD diss, Nagoya Institute of Technology*, 2002. Cited in page 25.

39 YAMAGISHI, J. et al. Robust speaker-adaptive HMM-based text-to-speech synthesis. *IEEE Transactions on Audio, Speech, and Language Processing*, Ieee, v. 17, n. 6, p. 1208–1230, 2009. Cited in page 25.

40 PERRON, L.; FURNON, V. *Google Cloud Text-toSpeech*. 2017. <https://developers.google.com/optimization/>. Acessed on 27 April 2021. Cited in page 26.

41 TAYLOR, P.; BLACK, A. W.; CALEY, R. The architecture of the Festival speech synthesis system. In: *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*. [S.l.: s.n.], 1998. Cited in page 26.

42 RESEARCH, T. S. *Gnuspeech*. 2002. <https://www.gnu.org/software/gnuspeech/>. Accessed on 27 April 2021. Cited in page 26.

43 DUNN, R. *eSpeak Text to Speech*. February 2006. <http://espeak.sourceforge.net/>. Accessed on 27 April 2021. Cited in page 26.

44 GONZALEZ, R. C.; WOODS, R. E. et al. *Digital Image Processing*. [S.l.]: Prentice hall Upper Saddle River, NJ, 2002. Cited 3 times in pages 26, 27, and 29.

45 ÜNAL, M. O. *Show Images Directly on Terminal: img2sh*. November 2019. <https://mozanunal.com/2019/11/img2sh/>. Accessed on 01 May 2021. Cited in page 26.

46 PETROU, M. M.; PETROU, C. *Image processing: the fundamentals*. [S.l.]: John Wiley & Sons, 2010. Cited 2 times in pages 27 and 30.

47 BOURNE, R. *Fundamentals of digital imaging in medicine*. [S.l.]: Springer Science & Business Media, 2010. Cited in page 27.

48 STATHIS, P.; KAVALLIERATOU, E.; PAPAMARKOS, N. An Evaluation Technique for Binarization Algorithms. *J. Ucs*, v. 14, n. 18, p. 3011–3030, 2008. Cited 2 times in pages 28 and 29.

49 ROGOWSKA, J. Overview and fundamentals of medical image segmentation. *Handbook of medical imaging*, Academic Press, 2000. Cited in page 29.

50 MOKJI, M. M.; BAKAR, S. A. Adaptive thresholding based on co-occurrence matrix edge information. In: IEEE. *First Asia International Conference on Modelling & Simulation (AMS'07)*. [S.l.], 2007. p. 444–450. Cited in page 29.

51 FLICKNER, M. et al. Query by image and video content: The QBIC system. *computer*, Ieee, v. 28, n. 9, p. 23–32, 1995. Cited in page 29.

52 KHURSHID, K. et al. Comparison of Niblack inspired binarization methods for ancient documents. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *Document Recognition and Retrieval XVI*. [S.l.], 2009. v. 7247, p. 72470u. Cited in page 29.

53 GREENSTED, A. *Otsu Thresholding*. June 2010. <http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html>. Accessed on 02 May 2021. Cited in page 29.

54 BLU, T.; LUISIER, F. The SURE-LET approach to image denoising. *IEEE Transactions on Image Processing*, Ieee, v. 16, n. 11, p. 2778–2786, 2007. Cited in page 29.

55 ACHARYA, T.; RAY, A. K. *Image processing: principles and applications*. [S.l.]: John Wiley & Sons, 2005. Cited 2 times in pages 29 and 30.

56 SONIA, S. M. Noise Reduction Techniques using Bilateral Based Filter. *Int. Res. J. Eng. Technol*, v. 4, p. 1093–1098, 2017. Cited in page 30.

57 GEOSPATIAL, L. *Smooth Images*. February 2020. <https://www.l3harrisgeospatial.com/docs/smoothimages.html>. Accessed on 05 May 2021. Cited in page 31.

58 CORPORATION, I. *Smoothing Images*. December 2020. <https://docs.opencv.org/master/d4/d13/tutorial_py_filtering.html>. Accessed on 05 May 2021. Cited in page 31.

59 AWAD, A. Denoising images corrupted with impulse, Gaussian, or a mixture of impulse and Gaussian noise. *Engineering Science and Technology, an International Journal*, Elsevier, v. 22, n. 3, p. 746–753, 2019. Cited in page 31.

60 FISHER SIMON PERKINS, A. W. R.; WOLFART, E. *Gaussian Smoothing*. December 2003. <https://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm>. Accessed on 07 May 2021. Cited in page 31.

61 RAJESH, T.; KARNAN, M.; SIVAKUMAR, R. An Efficient Image Denoising Technique for Various Noisy Images. *International Journal Of Engineering And Computer Science*, v. 3, n. 5064-5066, 2014. Cited in page 32.

62 CHUANG, Y.-Y. *Bilateral Filters*. 2015. <https://www.csie.ntu.edu.tw/~cyy/courses/vfx/10spring/lectures/handouts/lec14_bilateral_4up.pdf>. Accessed on 06 May 2021. Cited in page 32.

63 CHETHAN, H.; KUMAR, G. H. Image dewarping and text extraction from mobile captured distinct documents. *Procedia Computer Science*, Elsevier, v. 2, p. 330–337, 2010. Cited in page 32.

64 MASALOVITCH, A.; MESTETSKIY, L. Usage of continuous skeletal image representation for document images de-warping. In: *Proceedings of International Workshop on Camera-Based Document Analysis and Recognition, Curitiba*. [S.l.: s.n.], 2007. p. 45–53. Cited in page 32.

65 BLOOMBERG, D. *Dewarping Text Pages*. August 2010. <https://tpgit.github.io/UnOfficialLeptDocs/leptonica/dewarping.html>. Accessed on 07 May 2021. Cited in page 32.

66 ZUCKER, M. *Page Dewarping*. August 2016. <https://mzucker.github.io/2016/08/15/page-dewarping.html>. Accessed on 07 May 2021. Cited 2 times in pages 32 and 40.

67 DAMERAU, F. J. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, ACM New York, NY, USA, v. 7, n. 3, p. 171–176, 1964. Cited in page 33.

68 CROWELL, J. et al. A frequency-based technique to improve the spelling suggestion rank in medical queries. *Journal of the American Medical Informatics Association*, BMJ Group BMA House, Tavistock Square, London, WC1H 9JR, v. 11, n. 3, p. 179–185, 2004. Cited in page 33.

69 MURUGAN, S.; BAKTHAVATCHALAM, T. A.; SANKARASUBBU, M. SymSpell and LSTM based Spell-Checkers for Tamil. 2020. Cited in page 33.

70 ETOORI, P.; CHINNAKOTLA, M.; MAMIDI, R. Automatic spelling correction for resource-scarce languages using deep learning. In: *Proceedings of ACL 2018, Student Research Workshop*. [S.l.: s.n.], 2018. p. 146–152. Cited 2 times in pages 34 and 35.

71 BATZ, M. *Raspberry Pi NoIR Camera V2*. February 2017. <https://buyzero.de/products/raspberry-pi-kameramodul-v2-noir>. Accessed on 15 May 2021. Cited in page 36.

72 MOLANO, J. I. R.; BETANCOURT, D.; GÓMEZ, G. Internet of things: A prototype architecture using a raspberry Pi. In: SPRINGER. *International Conference on Knowledge Management in Organizations*. [S.l.], 2015. p. 618–631. Cited in page 35.

73 MORRIS, A. C.; MAIER, V.; GREEN, P. From WER and RIL to MER and WIL: improved evaluation measures for connected speech recognition. In: *Eighth International Conference on Spoken Language Processing*. [S.l.: s.n.], 2004. Cited in page 36.

74 RICE, S. V.; KANAI, J.; NARTKER, T. A. An evaluation of OCR accuracy. *Information Science Research Institute, 1993 Annual Research Report*, Citeseer, v. 9, p. 20, 1993. Cited in page 36.

75 KARPINSKI, R.; LOHANI, D.; BELAID, A. Metrics for Complete Evaluation of OCR Performance. In: *IPCV'18-The 22nd Int'l Conf on Image Processing, Computer Vision, & Pattern Recognition*. [S.l.: s.n.], 2018. Cited 2 times in pages 36 and 54.

76 KAFLE, S.; HUENERFAUTH, M. Evaluating the usability of automatically generated captions for people who are deaf or hard of hearing. In: *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*. [S.l.: s.n.], 2017. p. 165–174. Cited 2 times in pages 37 and 54.

77 POWELL, M. J. Direct search algorithms for optimization calculations. *Acta numerica*, Cambridge University Press, p. 287–336, 1998. Cited in page 40.

78 BYRD, R. H. et al. A limited memory algorithm for bound constrained optimization. *SIAM Journal on scientific computing*, Siam, v. 16, n. 5, p. 1190–1208, 1995. Cited in page 40.

79 GARBE, W. *The SymSpell Algorithm*. August 2020. <https://github.com/wolfgarbe/SymSpell>. Accessed on 14 May 2021. Cited 2 times in pages 43 and 45.

80 STRECKER, T. et al. Automated ground truth data generation for newspaper document images. In: IEEE. *2009 10th International Conference on Document Analysis and Recognition*. [S.l.], 2009. p. 1275–1279. Cited in page 46.

81 KIŠŠ, M.; HRADIŠ, M.; KODYM, O. Brno mobile OCR dataset. In: IEEE. *2019 International Conference on Document Analysis and Recognition (ICDAR)*. [S.l.], 2019. p. 1352–1357. Cited in page 46.

82 MEI, J. et al. MiBio: A dataset for OCR post-processing evaluation. *Data in brief*, Elsevier, v. 21, p. 251–255, 2018. Cited in page 46.

83 ZHARIKOV, I. et al. DDI-100: Dataset for Text Detection and Recognition. In: *Proceedings of the 2020 4th International Symposium on Computer Science and Intelligent Control*. [S.l.: s.n.], 2020. p. 1–5. Cited in page 46.

84 HAKRO, D.; TALIB, A.; MOJAI, G. Multilingual Text Image Database for OCR. *Sindh University Research Journal-SURJ (Science Series)*, v. 47, n. 1, 2016. Cited in page 46.

85 BURIE, J.-C. et al. ICDAR2015 competition on smartphone document capture and OCR (SmartDoc). In: IEEE. *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. [S.l.], 2015. p. 1161–1165. Cited in page 46.

86 ZAITSEV, A. *PRLib*. August 2016. <https://github.com/ZaMaZaN4iK/PRLib>. Accessed on 15 May 2021. Cited in page 53.

87 BELAID, A.; PIERRON, L. Generic approach for ocr performance evaluation. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *Document Recognition and Retrieval IX*. [S.l.], 2001. v. 4670, p. 203–215. Cited in page 54.

Appendix

APPENDIX A – BOOK2SPEECH PARAMETERS

At the beginning of Section 4, I commented on the flexibility of Book2Speech as a command line interface, providing users with the ability to configure the various modules present in the work. This appendix presents the parameters and flags supported by Book2Speech as well as an explanation of their purpose.

Listing A.1 – Book2Speech command line parameters and flags.

```
Usage:
  python core.py [PARAMETERS] [FLAGS]

Basic Parameters:
  -i     --image                path to a previous captured image
  -t     --text                 path to the reference text (required if
                                the metrics module is enabled)
  -m     --monogram             path to the monogram dictionary to be
                                used in text correction
  -b     --bigram               path to the bigram dictionary to be used
                                in text correction
  -o     --output               path to store the resulting text

Flags:
         --dewarp               enable page dewarping submodule
         --play-audio           play the resulting audio
         --disable-tts          disable the TTS module
         --improve-image        enable the image processing module
         --confusion-matrix     plot a confusion matrix referring to the
                                missrecognized characters
         --calculate-metrics    calculate display the CER, WER, WIL and
                                WIP error rate metrics
  -s     --save-results         save the resulting processed image
  -v     --verbose              shows program messages in terminal
         --debug                save resulting metrics and runtime in a
                                separate file
         --help                 show this message

Advanced Parameters:
         --lang                 [eng, por]
         --correction-mode      [direct,  compound, segmentation]
         --transform-mode       [reduced, default, extended]
         --blur-mode            [average, median, gaussian, bilateral,
                                 disable]
         --thresh-mode          [global, otsu, mean, gaussian, disable]
         --optimizer            [Nelder-Mead, Powell, CG, BFGS,
```

	Newton-CG, L-BFGS-B, TNC, COBYLA, SLSQP, trust-constr, dogleg, trust-ncg, trust-exact, trust-krylov, disable]