Text Digitizer

Vincent Dinh

Software Requirements Specification

Document

`

Table of Contents

1. Introduction

The Software Requirements Specification (SRS) document will outline all functions and qualities of the Text Digitizer. The following subsections will provide a background and purpose of the software specified in this document.

1.1 Purpose

The SRS will explain the features of the system, what the system will do, and the constraints under which it must operate. This document is written for the developer to document and clarify the software under development.

1.2 Scope

The specified software will be produced for any person which will allow users to easily create text or pdf files from images of text. Such as obtaining clean text from images of a novel to read on digital devices or editing a printed document where the original digital document is lost.

1.3 Definitions, Acronyms, and Abbreviations.

GUI - Graphical User Interface
OCR - Optical Character Recognition
OSD - Orientation + Script Detection
LAF - Look And Feel
OS - Operating System

2. Requirements

2.1 Product Perspective

2.1.1 System Interfaces

The software will not interact with other systems.

2.1.2 User Interfaces

The software will have a GUI where the user can select files, options, and execute the process.

2.1.3 Hardware Interfaces

The software will not use any hardware interfaces.

2.1.4 Software Interfaces

OpenCV
OpenCV team
https://opencv.org/

NumPy
NumPy community/NumPy's Steering Council
https://numpy.org/

Tesseract
Ray Smith/Google
https://opensource.google/projects/tesseract
https://github.com/tesseract-ocr/tesseract

Python-tesseract(pytesseract)
Matthias Lee
https://pypi.org/project/pytesseract/

2.1.5 Communications Interfaces

This software will not use any interfaces to communicate.

2.2 Product Functions

ID	Natural Language Requirement	Requirement Type
F1	The software shall use Tesseract for OCR	Functional
F2	The software shall use OpenCV to reduce noise in images	Functional
F3	The software shall have Java and Python components interact	Functional
F4	The software should accept PDF containing images as input	Functional
F5	The software shall parse images from PDF files	Functional
F6	The software shall delete any temporary files	Functional
F7	The software shall accept multiple files to be selected at once	Functional
F8	The software shall be able to combine the output of multiple files into one file	Functional
F9	The software shall let users select different output directories	Functional
F10	The software shall output a text file	Functional

F11	The software shall output a PDF file	Functional
F12	The software shall accept JPEG files	Functional
F13	The software shall accept PNG files	Functional
F14	The software shall have a GUI	Functional
F15	The software should have users select files through a file picker	Functional
F16	The software should have users select directories through a file picker	Functional
Q1	The software shall restrict the user to selecting only acceptable file types for input	Quality
Q2	The software shall have a progress bar	Quality
Q3	The software shall have a help section	Quality
Q4	The software shall restrict the user to selecting directories for the output location	Quality
F17	The software shall recognize alphanumeric characters related to English	Functional
F18	The software shall recognize special characters normally used in English	Functional
Q5	The software shall apply the OS's LAF to the GUI	Quality
Q6	The software should recognize white text on a black background	Quality
Q7	The software shall not have the GUI freeze temporarily	Quality
Q8	The software should read through each image within 3 min	Quality
Q9	The software shall allow the user to select options through the GUI	Quality
Q10	The software should implement an icon for the GUI	Quality

2.3 Constraints

* The software will require Python 3.4+
* The software will require Java 8+
* The software will be restricted by the quality of the image the user selects
* The software will follow normal conventions for desktop GUI's
* The software will run at least on Windows OS
* The software will read each image in less than 5 min

2.4 Assumptions

* The OCR engine will not perfectly read text
* The user will have the image upright
* The Tesseract engine will use the default given training data
* The software will not save options between sessions