PDF Reader and Text-to-Speech Converter

Overview

This project converts PDF files to text using Optical Character Recognition (OCR) and reads the text aloud. It includes image processing and speech synthesis capabilities.

Installation and Dependencies

This project requires the following dependencies:

- **Poppler**: Used to convert a PDF document to images.
- **Tesseract**: OCR engine to extract text from images.
- **eSpeak**: Command line tool to convert text to speech.

You can install these dependencies using Homebrew on MacOS with the following command: `sh brew install poppler tesseract espeak`

Program Structure

- **HD.pdf**: The source PDF file to be processed.
- **png_files/**: Directory containing images and extracted text of each PDF page.
- **HD_output.txt**: Combined text output from all PDF pages.
- **read_aloud.cpp**: C++ program to read the extracted text aloud.

How it Works

1. **PDF to Image Conversion**:

2. Each page of the PDF (`HD.pdf`) is converted to a separate PNG image using Poppler.

3. **OCR Processing**:

4. Each PNG image is processed using Tesseract to extract text, which is then saved as a `.txt` file in the `png_files` directory.

5. **Text Combination**:

6. All extracted text files are combined to form a single text file (`HD_output.txt`) that contains the entire content of the PDF.

7. **Text-to-Speech**:

8. The C++ program `read_aloud.cpp` reads `HD_output.txt` aloud.
9. Uses eSpeak on Linux, `say` on macOS, and PowerShell on Windows for TTS.
10. The program can be interrupted by pressing the 'z' key.

Running the Program

1. **Compile the C++ Program**:

2. Use the following command: `sh g++ read_aloud.cpp -o read_aloud -lpthread`

3. **Execute the Program**:

4. Run the compiled program with: `sh ./read_aloud`

Stopping the Program

- While the program is reading text aloud, you can stop it by pressing the 'z' or 'Z' key.

Production

- This program was created using a combination of PDF conversion, OCR, and speech synthesis technologies.
- It automates the conversion and reading process with efficient threading and platform-specific command execution.