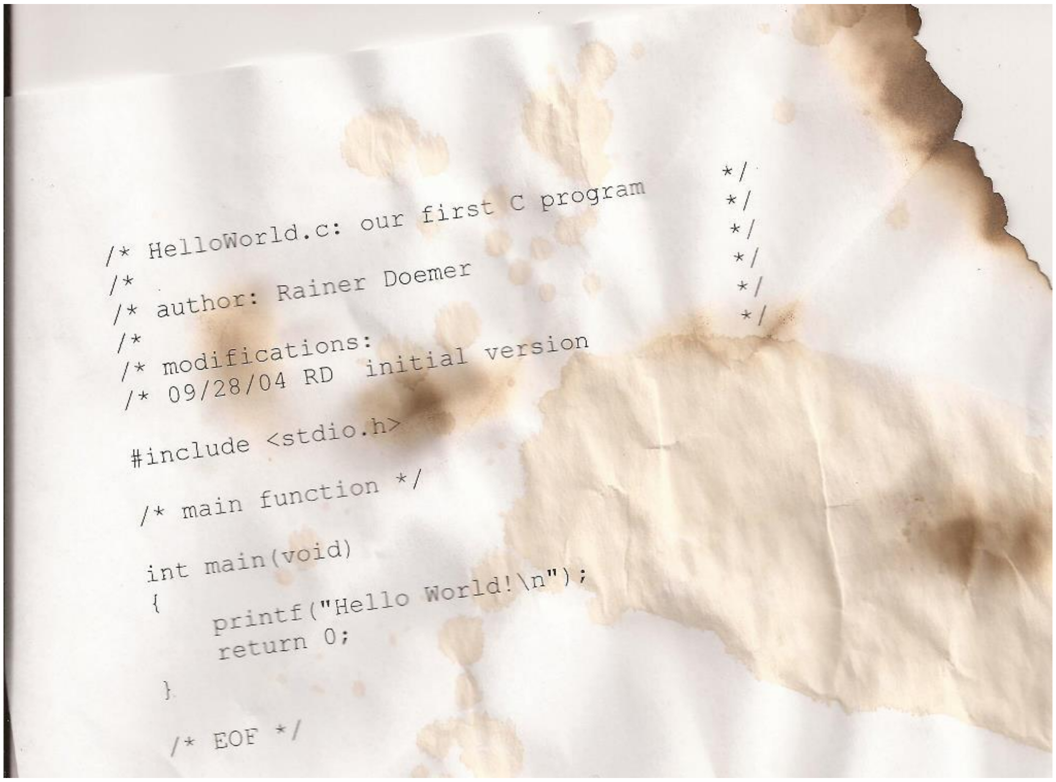


```
/* HelloWorld.c: our first C program       */
/*                                          */
/* author: Rainer Doemer                    */
/*                                          */
/* modifications:                           */
/* 09/28/04 RD  initial version             */

#include <stdio.h>

/* main function */

int main(void)
{
    printf("Hello World!\n");
    return 0;
}

/* EOF */
```

Optical Character Recognition

User Manual

Version 1.0

Team 3

Quan Chau

Hanchel Cheng

Kevin Duong

Jamie Lee

Ryan Morrison

Eric Rodriguez

Andrew Trinh

Table of Contents

Glossary

Digital image processing (DIP) – functions used to edit the image such as sharpening, rotation, resizing, and etc.

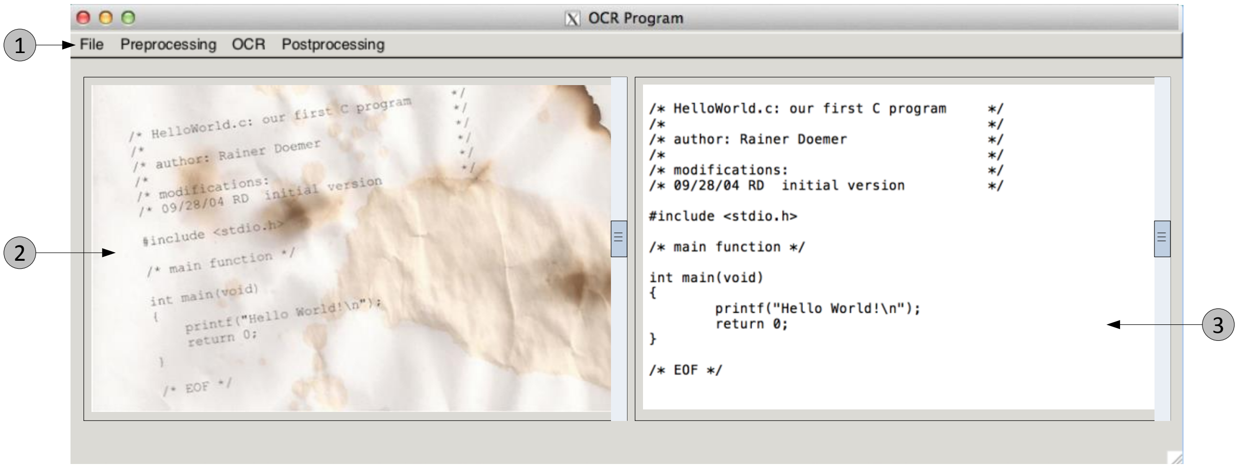
Optical Character Recognition (OCR) – program that matches characters on an input image to produce a text result

Post-processing – editing of text file produced by the OCR

Pre-processing – editing done to the input image to prepare for OCR program

1: Optical Character Recognition (OCR)

1.1 Usage scenario



```
/* HelloWorld.c: our first C program      */
/*                                        */
/* author: Rainer Doemer                  */
/*                                        */
/* modifications:                         */
/* 09/28/04 RD  initial version           */

#include <stdio.h>

/* main function */

int main(void)
{
        printf("Hello World!\n");
        return 0;
}

/* EOF */
```

Figure 1: Screenshot of the OCR program in use.

The image above depicts the OCR program in use. There are three primary components with which the user interacts:

1. Menu Bar – this is where the user initiates all commands for the program. The four main categories are File, Preprocessing, OCR, and Postprocessing. (See section 3 for detailed explanations of the various commands available in the menu bar.)
2. Image Pane – this pane displays the image loaded by the user; the current image in this pane is available for Preprocessing and OCR commands.
3. Text Pane – this pane displays the resultant text after OCR has been performed; the current text in this pane is available for Postprocessing commands.

A typical workflow of the program will look like this:

a) User selects the scanned image file with the File→Open option in the menu bar.

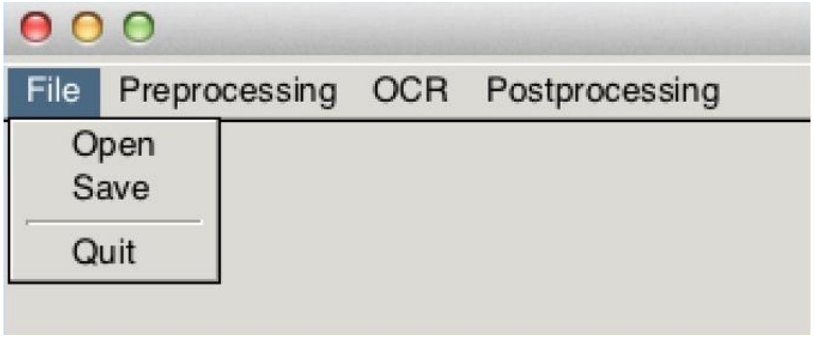


Figure 2: The File menu bar option. File management and quitting the program are accessed here.

b) The program loads the original image in the Image Pane (see Figure 1 above).

c) User selects from several preprocessing functions to perform:

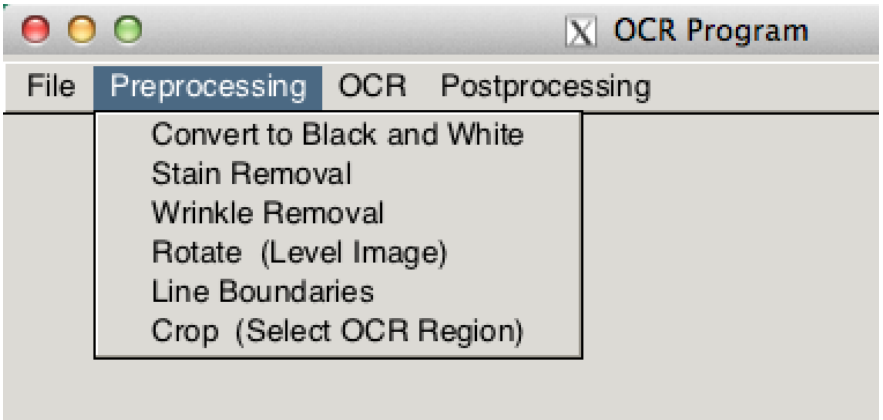


Figure 3: The Preprocessing menu bar option. All functions for image editing are accessed here.

d) User selects the font of the document.

e) The program performs OCR on the image by selecting OCR→Perform OCR. The text result is displayed in the Text Pane (see Figure 1 above). At this step, ambiguous characters will be highlighted and the user can modify it in the text area.

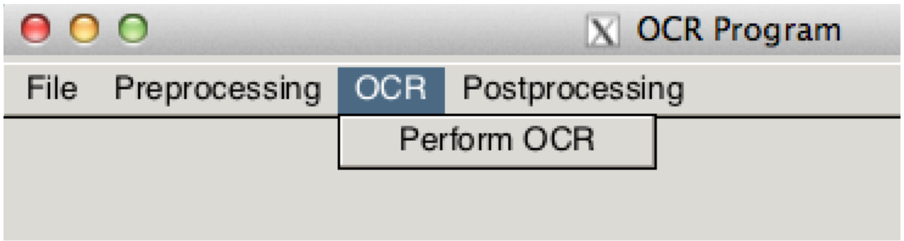


Figure 4: The OCR menu bar option. Text is extracted from the image with the Perform OCR function.

f) If user is not satisfied with the result, user can go back to step c). Otherwise, user will save the text result into a file with File→Save.

g) If the user intends to edit the resulting text file, this is achieved by selecting Postprocessing→Edit OCR Text Output. Postprocessing→Dictionary Settings is also available to apply corrections to OCR misreads.

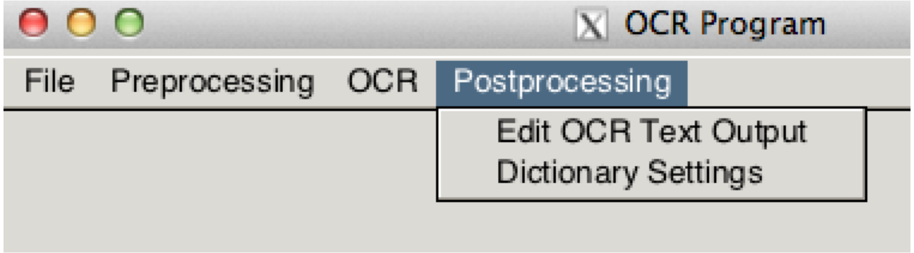


Figure 5: The Postprocessing menu bar option. Functions for editing text are selected here.

h) Selecting File→Save will save the processed text, File→Quit exits the OCR program.

1.2 Goals

The goal of the OCR is to most accurately translate the input from an image file into a text file through preprocessing of the image and postprocessing of the resultant text.

1.3 Features

The OCR will load a scanned input image from the user and go through various preprocessing DIP techniques such as stain and wrinkle removal, rotation, cropping, and black and white conversion. The OCR will then identify and crop individual characters and match these by comparing to reference character images to store in a text file. Then, the text will go through post processing by comparing with a preloaded dictionary for the most accurate results.

The OCR will also support different image input formats (such as .ppm, .bmp, .jpg, .jpeg and .png), combine multiple input images into a single text file as the user desires, and support different text faces, styles, and sizes.

The user will be able to upload an image to the OCR through an interactive graphical user interface. It will prompt the user to select stained areas and also identify where the first line begins and ends, so that the OCR is able to more accurately crop and isolate the characters for recognition. The option of adding another image will be available so that multiple images can be output in a single text file rather than multiple text files that the user will have to concatenate by hand.

2: Installation

2.1 System requirements

- Hardware: PC Hardware (x86_64 server)
- Operating system: Linux OS (RHEL-6-x86_64)
- Dependent third party software:
  i. gcc
  ii. GNU make
- Dependent libraries:
  i. GTK+-2.0 or GTK+-3.0 for graphical user interface
  ii. Netpbm library for image processing
  iii. Uthash library for dynamic hash table, array and string

2.2 Setup and configuration

- GTK+ library installation: details are found at www.gtk.org
- Netpbm library installation: details are found at netpbm.sourceforge.net
- Uthash library installation: not necessary because it's already included in the package
- The software comes in a tar.gz package. After downloading, extract the package by running:
  ```
  tar –zxvf OCR.tar.gz
  ```
- Change into the directory by running:
  ```
  cd OCR
  ```
- Compile the program by running:
  ```
  setenv PKG_CONFIG_PATH /usr/share/pkgconfig
  make OCR
  ```

2.3 Uninstalling

- Change into the directory by running:
  ```
  cd OCR
  ```
- Compile the program by running:
  ```
  make clean
  ```

3: OCR Program Functions and Features

3.1 Detailed Description of Image Preprocessing

Image preprocessing includes:

- Stain and wrinkle removal: being able to recognize the differences between stains, wrinkles, and other marks on the scanned page in order to change these areas to white in order to isolate the text
- Rotation: leveling the scanned image so that the OCR is later able to crop each letter for comparison
- Cropping: leaving out the extra white (or marred) spaces that do not contain characters so that the OCR will not need to look through unnecessary space
- Black and white conversion: changing the input image so that the OCR will be able to match the distribution of pixels to the set of character images
- Line boundaries setup: user select three boundaries to help the program detect the text lines easier:
  - The top of first line
  - The bottom of first line
  - The top of second line

3.2 Detailed Description of OCR Functions

The OCR includes these features:

- Isolate and crop characters based on the differences between black and white pixels
- Match the isolated images to a preloaded set of character images in order to output each one as text

3.3 Detailed Description of Text Postprocessing

Text postprocessing includes:

- Storing the recognized characters into a data structure
- Comparing to a preloaded dictionary to fix any ‘typos’ of similar-looking characters for the most accurate results
- Displaying the recognized text

Back Matter

1. Copyright

   This software is licensed under GNU General Public License version 3. Details can be found on http://www.gnu.org/licenses/gpl.html

2. Error message:

- Lines boundaries not yet selected: This error means the user didn't provide the line boundaries yet
- Invalid image format: This error means the image selected doesn't have a supported format
- Unable to perform OCR: This error means the preprocessed image isn't clear enough for the OCR to detect any characters. To solve this problem, user will have to do some more preprocessing on image before retrying OCR

3. Index