EECS 22L: Project 2 Grading Criteria

Prepared by: Che-Wei Chang, Weiwei Chen, and Prof. Rainer Dömer

March 12, 2013

The follow items are mandatory for the *Final* version of the *OCR* program:

1. For the Final release, there should be two deliverables like we had in the **chess** project.
   - A complete and clean tarball, namely `OCR_Source.tar.gz` as the project source deliverable with all items listed in **Item 5**
   - A complete and clean tarball, namely `OCR.tar.gz` as the user deliverable with:
     - The `bin` directory where the binary executable file of the `OCR` and the related resources, such as images for the library, are put
     - A *pdf* file named `OCR.pdf` in the `doc` directory as the user manual of the *OCR* program. Screenshots of the program functions are desirable to have in this document
2. For the Final release, we are expecting an OCR program with minimum capability of transforming the clean color image into a compilable C program source code file. Also, interactive user interface (ASCII or Graphic) is also required. The input sample source images are:
   `eecs22/ocr_scans/30a_ColorWorld_Clean300DPI.jpg`,
   `eecs22/ocr_scans/30b_ColorWorld_Clean200DPI.jpg`,
   `eecs22/ocr_scans/30c_ColorWorld_Clean100DPI.jpg`.
   You can use any one of them as the input for your OCR program.
3. Also, any advanced features, such as stain/wrinkle removal, image rotation, keyword dictionary, and multiple fonts support, are preferable. You can find the sample images in directory `eecs22/ocr_scans/` as the input. The name of the file also explains the imperfection in the image.
4. In addition to the functionality of the whole OCR function, in the Final release, we are also looking for the unit test for every module. Every module in the project should be tested independently. For every module in the project, a main function should be created to test all functions defined in the module. For example, for the DIPs function in the OCR, the main function should provide source image to the DIPs, generate processed images (binarized, black-and-white, sharpened image, etc.) and write the those images into files.
5. A **well structured** project folder with **compilable** source code files and corresponding documentations are also mandatory for the beta version. Specifically, we are looking for:
   - A complete and clean tarball with all the project files in proper directories
   - Proper project file hierarchy (as what was presented in week3's discussion session)
   - An `INSTALL` file with the descriptions of two installation options:
     - tarball extraction
     - CVS checkout information, i.e. linux command that can be used to get the project checkout

     Proper instructions on how to install the program.
   - A `README` file with the information of the authors, program version, date, and general information / description about the software.
   - A `COPYRIGHT` file with authors and copyright information

- A working top-level `Makefile` with at least three targets, i.e. ‘all’, ‘test’, and ‘clean’.
  Note: this file should be different from the `Makefile` in the *src* directory.
- The `src` directory with all the properly documented program source code files.
- The `bin` directory with the binary executable file of the OCR program.
- The `doc` directory with all the documentation files for this project, i.e. `OCR_SW_Spec.pdf`
- An ASCII text file named `OCR.1` in directory `man/cat1/` as the static text file for the *OCR* program’s manual page
- The `test_dir` directory with the unit test modules, a Makefile for unit testing and, instructions on how to do the unit testing. **Note:** it is desirable to use the ‘make test’ target in the top-level makefile to execute all the test targets in the Makefile in the `test_dir` directory.

6. The detailed grading criteria is listed below:

   - OCR which fulfils the basic requirement (capable of converting clean image into compilable file): 50 (Minor mismatch is acceptable, but at least 50% of the content should be correct)
   - Preprocessing, including binarization, stain/wrinkle removal, and image rotation: 15
   - Post-processing, such as keyword-dictionary function: 10
   - Support for multiple fonts and image formats: 10
   - Support for multiple pages input: 10
   - Unit test for all modules in the project: 10
   - Well structured project deliverable folder which meets the requirements in **Item 5**: 10
   - Well structured user deliverable folder and detail documentation for the program in `OCR.pdf`: 10
   - Extra points for preferable features like **high recognition rate** and **friendly GUI**.
   - Also, if your OCR program has any **innovative** features we do not list in the handout, please document it in detail in `README`, `OCR_SW_Spec.pdf`, and `OCR.pdf`. Extra credits will be rewarded for that.

Good luck and Happy Coding!