Guidelines for Programming Assignments
Computer Vision, Fall 2016

This document aims to clarify important issues relevant to programming assignments for the course. Following the guidelines explained here will greatly simplify submission, testing, and grading of your work.

Academic Honesty Policy

The course requires you to read the Code of Academic Integrity [1] as stated by The Fu Foundation School of Engineering and Applied Science. For each homework, you must formally agree to the academic policy by electronically signing your name and UNI in the designated file.

You are permitted and encouraged to discuss ideas with other students. However, you may not share answers or your code with others. Also, copying code/solutions from other sources, including online files, MATLAB sources or solutions from other educational institutions, but not limited to these is prohibited. We will be monitoring source code for individuality. Ramifications for students who violate the Code of Academic Integrity may include an "F" grade for the course, documentation on your academic file and even probation.

Software

Environment

The official software environment for this course is MATLAB R2015a on Windows 7 (64 bit), which can be accessed from the computers in CUIT's computer labs across the campus. A copy of MATLAB can also be downloaded and installed on your personal computer, as described in the Programming Languages section.

MATLAB supports all major operating systems, and is platform independent for the most part. We do not anticipate any platform or release specific issues for the programming required in this course. We strongly recommend you to test your solution in the official environment before submission.

Programming Languages

The official programming language for this course is MATLAB. A copy of MATLAB can be obtained from the School of Engineering's MATLAB portal [2]. Installation instructions are also available at the same location. The best resource to learn MATLAB is the MATLAB documentation [3], which is an excellent database of commands and examples. There are several ways to access the documentation:

- **From the web.** The entire MATLAB documentation is available online.

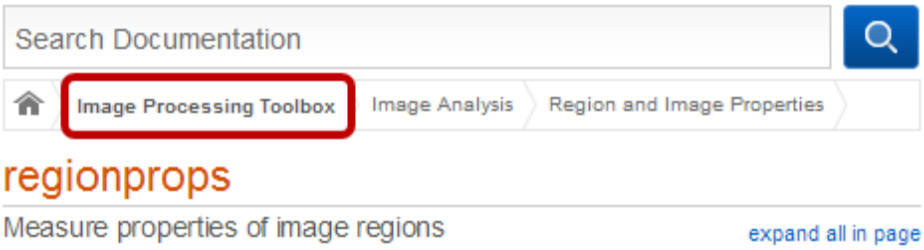
- **From the MATLAB application help menu.**
- **At the MATLAB command prompt.** Type `doc function_name` to display the documentation of function_name in the help browser, or simply type `doc` to launch the documentation browser.

Several additional learning resources are listed in the Resources section. If you are new to MATLAB, the official Getting Started Guide and tutorials [4] are good resources to get you started.

Special Notes on Using Built-in MATLAB Functions

MATLAB comes with a large collection of functions and toolboxes. In the context of this course, some of the classic algorithms have been implemented as built-in functions in Imaging Processing Toolbox or Computer Vision Toolbox. **Thus, only basic MATLAB functions are allowed in assignments, unless otherwise specified.**

In particular, **you may not use any functions from MATLAB's Image Processing or Computer Vision toolboxes**, except for functions that are explicitly permitted. To check if a particular function is from one of these toolboxes, type `doc function_name` into the Command Window to view the function's documentation. At the very top of the documentation window, the source of the function will be listed. For example:



In addition, the usage of one or more **specific built-in MATLAB commands may be barred**. Special instructions regarding allowed or disallowed functions will be explicitly stated in the description of each programming problem.

Pay attention to the instructions specified in each assignment. If you are unsure whether a built-in function can be used or not, please ask before assuming! You will receive no credit for using built-in functions where you should be coding them. When in doubt, ASK.

Working with Images in MATLAB

- **Image memory representation.** An image is represented as a matrix in MATLAB. In general, a grayscale image is represented as an m (rows) x n (columns) matrix, where m corresponds to the height of the image in number of pixels, and n corresponds to the width of the image in number of pixels. Similarly, a color (RGB) image is represented as an m (rows) x n (columns) x 3 matrix, where the three "slices" of the matrix represent the red, green and blue channels of an image.
- **Image coordinate system.** As mentioned above, an image is represented as a matrix in MATLAB. Therefore, we can retrieve the intensity of a pixel in the same

way we retrieve an element of a matrix. Suppose we want to retrieve the intensity of a pixel P at location x along the X-axis and y along the Y-axis of an image A, it is conventional in literature and in our course notes to write as `P=A(x,y)`. However, the ordering of pixel indices is reversed in MATLAB. **To retrieve the intensity of pixel P in the XY coordinate system, the correct MATLAB expression is** `A(y,x)`**, not** `A(x,y)`**!** This is often a source of confusion (and may be a source of bugs in your code).

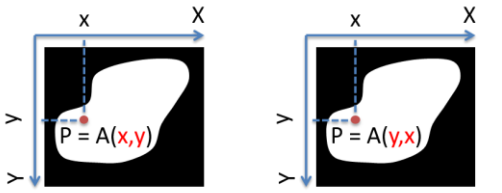


Image A in Literatures and Lecture Notes | Image A in MATLAB

Comparison of Pixel Index Ordering in Lecture Notes and in MATLAB

Running and Testing

For each homework, a file named **runHwX.m** will be provided (where X indicates the homework number). This file will serve as a framework for the homework and guide you through the individual programming milestones. In some cases, you will need to edit this file to complete the skeleton code or fill in certain parameters. It also serves as the interface to all functions/programs you need to implement and will help you run and test your solution. Therefore, your functions must strictly adhere to the specified signatures.

runHwX.m contains a set of sub-functions prefixed with the string "walkthrough" or "challenge," which are test cases for walkthroughs and challenges. To execute a test, type `runhwX("test_name_string")` at the MATLAB command prompt. To execute the entire set of tests, type `runHwX("all")` at the MATLAB command prompt. Make sure you can run **runHwX.m** without errors before submission.

Submission

Homeworks should be submitted in two parts:

1. The analytical portion should be submitted in *hard copy*, **in class** on the day it is due.
2. The programming portion should be submitted electronically **before the beginning of class** (by 10:10 a.m.) on the due date. Submission should contain:
    - **MATLAB** source files
    - **runHwX.m**. This file is provided in each assignment. Make sure to fill in your own name, UNI, filenames, parameters, etc.
    - **README**. In this file you should briefly explain your design decisions, the parameters/constants you used in the algorithms and describe any features

that you have incorporated into your programs in addition to what was required.

Include all the files in a zip file named **hwX_yourUNI.zip** (where X is the homework number) and upload the zip file to CourseWorks.

References

[1] Columbia Engineering Academic Integrity. [Online]. http://gradengineering.columbia.edu/academic-integrity-1

[2] Columbia Engineering MATLAB Portal. [Online]. http://portal.seas.columbia.edu/matlab/

[3] MATLAB Documentation Center. [Online]. http://www.mathworks.com/help/documentation-center.html

[4] MATLAB Tutorials. [Online]. http://www.mathworks.com/academia/student_center/tutorials/launchpad.html

[5] MATLAB Image Coordinate System. [Online]. http://www.mathworks.com/help/images/image-coordinate-systems.html?nocookie=true

[6] MATLAB Documentation Center. [Online]. http://www.mathworks.com/help/documentation-center.html