CSC 589 Introduction to Computer Vision Final Project

Due dates:

Final project presentation will be on April 30th, 2015
Final deliverables are due: end of day, May 4th, 2015.
Proposals are due April 10th, 2015.
Proposals must be an email message to me on what you plan to work on, sketching a plan you want to pursue, and who will be your teammates.

Percentage of final project: 20% + Extra credits (up to 10% of the total grades). This is your chance to earn credits over the course and try to do as much as you can here in this project.

Best reference: your text books (both Chapter 3 and 4 of Szeliski) contain many potential ideas. The Solem book provided implementation of several algorithms that you should feel free to consult with.
Here is the pdf:

http://szeliski.org/Book/drafts/SzeliskiBook_20100903_draft.pdf

Instructions

- You must finish your Final project before April 30th. This will be the last day of class.

- Team work: highly encouraged! But maximum number of a team is 3. You must clearly specify each member's contribution.

- It is OK to use OpenCV or Skimage or any other existing codes from the Internet, but you must implement significantly amount of your own code for the topic of your choice. You can, on the other hand, learn from other code and refer to your textbook for resources. However, the more you implement on your own, the better grade you will receive. You should upload your final code onto GitHub and contributed to the community.

- Your deliverable is a website, similar as your homework. You can use the same templates as the homework. Please specify your name and your team member's name.

- Provide clearly instructions of how to run your code. Name your folder with your names.

- Please cite your sources. If you want to use any existing code to compare your own, cite the source and specify where you get the code. Please also cite the literature.

- Project presentation: each team makes a 5-8 mins youtube video explaining your projects.

- Team contribution must be specified clearly.

Datasets

There exists many computer vision datasets for test your algorithm:
You can find most of them here:

http://www.cvpapers.com/datasets.html

Grading policy

Here are the grading criterion:

A (Excellent)
You implement significantly amount of your code with an existing paper/method. The paper/method could be ones we discussed in class or new ones. You provided clearly instructions on how to run your code. Your code runs with smooth user interface and is well commented and well written and your deliverables are well presented. Your results are good and you discussed the strength and weakness of your algorithm. You impressed me with the originality of your work, your website is informative, and your presentation is solid!

B/B+. (Good)
You implemented some of your own code with a combination of existing packages. However, your problem is novel and approach is interesting. The scope of your implementation is less significant than the A category but still solid. . Your code runs with smooth user interface and is well commented and well written and your deliverables are well presented. Even though the scope of your implementation is small, you discussed extensively and compared with off-the-shell methods.

C. (Acceptable)
You didn't implement much of your own code or the code you implemented is not quite doing what you aim for. However, you tested a few off-the-shell methods. Your code runs with smooth user interface and is well documented and well commented. You have discussed with me about the project through out.

D. (Not acceptable) I do not expect any of you would get this grade. However, the following scenario could result in such a grade:

1. You didn't really implement anything original and the efforts you put in the project is negligible.
2. You failed to propose a project until the very last week.
3. Your project has not generated any results.
4. Your code didn't run and you didn't talk to me early enough.
5. The final website is not complete and not well written.

F (Fail) You didn't submit your work on time or you didn't show up for the final presentation.

Rubic:

60% Code implementation and algorithm.
20% Coding style, accuracy, user-interface, results, plots.
10% Final presentation
10% Final deliverable of the website.

Extra credits:
1. Have written valuable and efficient Python codes or Python wrapper for existing c binaries and contributed to the community.
2. Collect your own calibrated images using your own cameras.
3. Implemented more than two major methods.
4. Impressing me with a new intellectual angle of the problem or the methods.

Project Topics:

For the course final project, you have the following choices:

Make significant extension to any of the methods/projects we discussed in class. If you want to focus on one method, you MUST implement your own Python code. You cannot just run an existing algorithm you found on the Internet and test on some images. You should pick at least one paper and try to implement it. The selected topics include:

Alternatively:

1. Pyramid methods in image editing. For example this is a great paper to implement: http://web4.cs.ucl.ac.uk/staff/j.kautz/publications/exposure_fusion.pdf
2. Other image editing methods. You can extend what we have done with Pyramid blending with Poisson image editing:

Ref:
http://www.cs.jhu.edu/~misha/Fall07/Papers/Perez03.pdf

3. Edge detection (You can implement your own and compare with Canny). You can also extend this to colors (see Chapter 4.2 for details) https://hal.inria.fr/inria-00548615/document
4. Boundary detection. Implement any existing boundary detector such as crispy boundary detection: http://web.mit.edu/phillipi/pmi-boundaries/
   This code does not exist in Python. So it would be great to code it up in Python. You can then compare this with other methods you find on the Internet such as Martin, Fowlkes, and Malik's pB boundary detector (2004). Chapter 4.2 in your text book.
5. Feature descriptors (SIFT, contrast-normalized patches, Shape context, etc, See Chapter 4.1) Implement any of the feature descriptors. Extend it to do object recognition is a huge plus!
6. You can also work on setting up external websites and collect visual psychophysical data on "the dress" with crowd sourcing. But this involves coding up website for Amazon Mechanical Turk. The novelty here is that you must design original experiment, create sensible website, send the data to AMT and analyze the data. Talk to me early if you want to pursue this.

7. Extend the feature detection, description, matching algorithm to do automatic paranoma image stitching. Solem's book Chapter 2/3 has some good code to start. Again, you have to implement your own code. You can implement the method from this paper: http://cs.brown.edu/courses/csci1290/asgn/proj6/resources/ImageMatching.pdf

8. Finally but not lastly, you can contribute by writing Python wrappers for the c binaries VLfeat. Currently, the python wrappers are sparse and not so good. Maybe along the course project, you can help. VLfeat is here: http://www.vlfeat.org/

Here is an alternative project if you don't want to pursue any of the above:

Interests point detector

Implement one or more feature detectors and compare their performances (with your own or a classmate's or a method downloaded from the internet).

A. Laplacian or Difference of Gaussian.
B. Forsner-Harris (try different formula variants given in formula 4.9-4.11 of your text book, page 188-189)

C. Oriented/steerable filter, looking for either second-order high second responses or two edges in a window (Koethe 2003) as discussed in Section 4.1.1.
D. Compute the detections on a sub-octave pyramid and find 3D maxima.
E. Implement non-maximal suppression, such as the adaptive techqnieu discussed in Brown, Szeliski and Winder (2005) and page 188 on Chapter 4.1 of your text book.
F. Test on your own images and discuss extensively of your results.