Introduction to Computer Vision (ECSE 415)
Assignment 3

Due: March 13^{th}, 11:59PM

Please submit your assignment solutions electronically via the mycourses assignment dropbox. Students are expected to write their own code. (Academic integrity guidelines can be found at `https://www.mcgill.ca/students/srr/academicrights/integrity`). Assignments received up to 24 hours late will be penalized by 30%. Assignments received more than 24 hours late will not be graded.

There are three questions in this assignment. The questions permit you to explore segmentation, stereo vision and motion algorithms discussed in class. Attempt all parts of this assignment. The assignment will be graded out of total of **60 points**. The first question requires that you implement several algorithms from scratch. You can use library-implemented functions for the rest of the assignment.

Submission Instructions

1. Submit a single jupyter notebook consisting solution of the entire assignment.
2. Comment your code appropriately.
3. Do not forget to run Markdown cells.
4. Submit a pair of stereo images selected for the second question. Do not submit input images for other questions.
5. Assume input images are kept in a same directory as the codes.
6. Do not submit output images. Output images should be displayed in the jupyter notebook itself.
7. Make sure that the submitted code is running without error. Add a README file if required.
8. If external libraries were used in your code please specify its name and version in the README file.

9. Answers to the reasoning questions should be comprehensive but concise.

10. Submissions that do not follow the format will be penalized 10%.

1 Segmentation

You will now explore K-means and EM for the task of image segmentation. Segmentation will be performed on the image: 'yellowlily.png' (Figure 1).

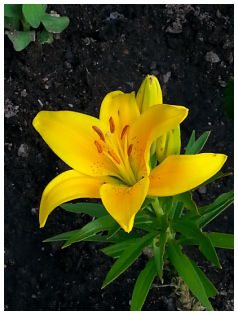

Figure 1: Image to be used to explore segmentation algorithms.

1.1 K-means clustering (10 points)

- Implement the K-means clustering algorithm using only the numpy library. You can use opencv and matplotlib libraries only to read and display images but not for clustering.
- Apply K-means to segment the image using [R,G,B] features. (R,G,B) represents color channels normalized between 0 and 1.
- Display the resulting segmented images for the first 10 iterations.

1.2 Expectation Maximization - Gaussian Mixture Models (15 points)

- Implement the EM algorithm using only the numpy library. You can use opencv and matplotlib libraries only to read and display images but not for the overall EM algorithm.
- Apply GMM to the provided image using the [R,G,B] features. (R,G,B) represents color channels normalized between 0 and 1.
- Display the resulting segmented images for the first 10 iterations.

1.3 Reasoning question (2 points)

Under what data distribution conditions would K-means and EM give the same solution, provided similar initializations?

2 Stereo Vision - Epipolar Geometry (25 points)

Use a stereo image-pair of your choice from the Middlebury dataset: `http://vision.middlebury.edu/stereo/data/scenes2014/` for the following questions. You can use functions from OpenCV and matplotlib for this question.

- Compute matching SIFT keypoints from a stereo image pair. (5 points)
- Compute and display the epipolar lines for both images. (5 points)
- Pick any one keypoint in the left image which has a correct match in the right image, and is on the corresponding epipolar line. Extract a patch of size (5×5) around this keypoint in the left image. (2 points)
- Match the extracted patch to every 5×5 patch along the corresponding epipolar line in the right image. Use normalized cross correlation metric for matching. (7 points)
- Plot normalized cross correlation values (on y-axis) against index of the patch in the left image (on x-axis)(refer Lecture 13 slide 57). Find the matching point with maximum normalized cross correlation value. Display found matching points in both the images. (4 points)
- Did you find exactly one matching point or multiple matches? Is the matching point you found the correct one? Explain. (2 points)

3 Motion Algorithm - Multi-resolution Lucas-Kanade optical flow estimation (8 points)

You will now explore the multi-resolution Lucas-Kanade optical flow estimation algorithm discussed in class. Use the given two video frames 'frame1.png' and 'frame2.png' shown in Figure 2 for this question. You can use functions from OpenCV and matplotlib for this question. Follow each of the steps below:

- Extract good points to track from 'frame1.png' using the Harris corner detector. Use the openCV function `goodFeaturesToTrack` and set the parameter value `maxCorners`=500. Search for the optimal values for the parameters `qualityLevel, minDistance, blockSize`. (2 points)
- Compute the optical flow between 'frame1.png' and 'frame2.png' for the above detected points. Use the openCV function `calcOpticalFlowPyrLK`. Set `winSize`=10, `TERM_CRITERIA_EPS`=0.03 and `TERM_CRITERIA_COUNT`=10.

(a) (b)

Figure 2: Video frames to be used for testing motion algorithm: (a) frame1.png (b) frame2.png

Experiment with the maximum pyramid level by variying `maxLevel` parameter from 0 to 10. For each pyramid level, compute the mean of the tracking error returned by `calcOpticalFlowPyrLK` function for points whose correspondence search is successful. (2 points)

- Display the optical flow for each setting of maximum pyramid level. Comment on the quality of the results. (2 points)
- Plot the mean of the error (on y-axis) vs. pyramid level (on x-axis). Discuss the trends you observe in the plot. (2 points)