Introduction to Computer Vision (ECSE 415)
Assignment 3

Due: November 23^{rd}, 11:59PM

Please submit your assignment solutions electronically via the myCourses assignment dropbox. Attempt all parts of this assignment. The assignment will be graded out of total of **26 points**. 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.

Submission Instructions

1. Prepare and submit a single Google Colab notebook containing answers to all three questions.
2. Comment your code appropriately.
3. Do not submit input/output images. The output images should be displayed in the notebook. Assume the input image folders are kept in a same directory as the codes.
4. Make sure that the submitted code is running without error. Add a README file if required.
5. Answers to reasoning questions should be comprehensive but concise.
6. Submissions that do not follow the format will be penalized 10%.

1 Image Segmentation using K-means

Implement K-means algorithm using only the numpy library. You can use opencv and matplotlib libraries only to read and display images. Apply K-means to the images 'home' and 'flower' shown in Figure 1. Try K=2 and K=3. Run the algorithm for 10 iterations and display the resulting segmented images in each case. **(10 points)**

(a) (b)

Figure 1: Segment above images using K-means algorithm.

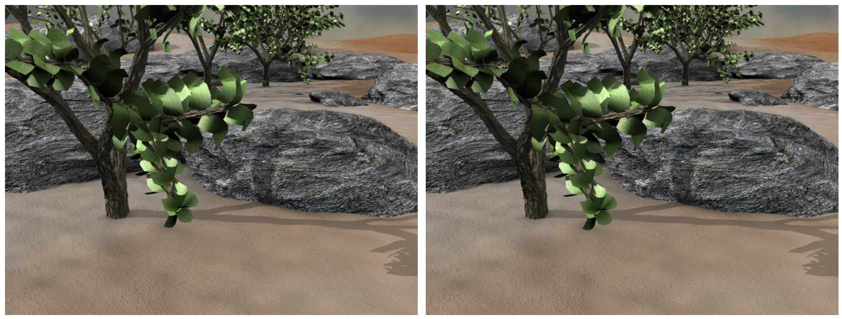
(a) (b) (c)

Figure 2: A pair of stereo images (a) left image (b) right image (c) disparity map (expected output).

2 Disparity

In this section, we will compute disparity map D from a pair of stereo images captured using parallel cameras. The images are shown in Figure 2(a) and 2(b). We will solve correspondence problem with the window search algorithm. Refer to slides 58-59 in *Lecture 18 - Stereo Vision*. Instead of searching for a matching window on the entire scanline, we will restrict the search on a small region on the scanline.

1. Extract a 5×5 window centered at each pixel-location $(i, j)_L$ in the left image. Let's call these windows reference windows. **(2 points)**
2. For each reference window in the left image do the following.
   (a) On the right scanline, create a search region bounded by pixel-locations $(i, j-47)_R$ and $(i, j)_R$. Extract 5×5 windows centered at every pixel-location in this search region. **(2 points)**
   (b) For few boarder pixel-locations either the reference window or the search region lie outside the boundary of the image. Set disparity



(a) (b)

Figure 3: Input frames for optical flow computation. (a) frame1 (b) frame2.

$D(i,j) = 48$ for these pixel-locations. For the remaining locations do the following. **(2 points)**

(c) Compute sum-of-square-difference(SSD) between the windows in the search region and the reference window. **(2 points)**

(d) Find a location $(i', j')_R$ with minimum SSD and compute disparity $D(i,j) = j_L - j'_R$. (Note that $0 \leq D(i,j) \leq 47$ as the search region contains 48 pixel-locations.) **(1 point)**

3. Display the final disparity map D with the cmap argument of `plt.imshow` set to `'gray_r'`. The expected output is shown in Figure 2(c). **(1 point)**

3 Optical Flow

In this section, we will observe the effect of the window-size on the prediction accuracy of optical flow. The input frames are shown in Figure 3(a-b) and the ground-truth flow in given in 'flow10.npz' file. Read ground truth flow as follows: `gt = np.load('flow10.npz')['flow']`

1. Use `calcOpticalFlowFarneback` from OpenCV to compute optical flow between the input frames with the arguments set as follows. **(2 points)**
   - flow=None, pyr_scale=0.5, levels=3, iterations=3, poly_n=5, poly_sigma=1.2 and flags=0.
   - Very winsize from 5 to 21 in the steps of 2.
2. For each setting of winsize, measure mean squared error(MSE) between estimated optical flow and the ground truth optical flow. Plot MSE (y-axis) vs winsize (x-axis). **(2 points)**
3. Do you observe any trend in the plot above? Does the error increase or decrease with increasing window-size? Explain the effect of window-size on the prediction error. **(2 points)**