ASSIGNMENT4

DIGITAL IMAGE PROCESSING (DIP) - CSE 478

DEADLINE: 14TH OCT (FRIDAY), 11:50PM

(1) Given an image, write a generic code to highlight the maximum area square that does not contain any cloud. Run your code without making any changes for images cloud1.jpg, cloud2.jpg and cloud3.jpg. Ideally your code should run in real time. Upload the code along with result images.

(2) A company that bottles a variety of industrial chemicals has heard of your success solving imaging problems and hires you to design an approach for detecting when bottles are not full. The bottles appear as shown in "bottles.tif" image, as they move along a conveyor line past an automatic filling and capping station. A bottle is considered imperfectly filled when the level of the liquid is below the midway point between the bottom of the neck and shoulder of the bottle. The shoulder is defined as the region of the bottle where the sides and slanted portion of the bottle intersect. The bottles are moving, but the company has an imaging system equipped with a illumination flash front end that effectively stops motion, so you will be given images similar to "bottles.tif". Based on the material you have learned in the class, write a program for detecting bottles that are not filled properly. State all the assumptions that you make and that are likely to impact the solution you propose.

(3) Design an algorithm to detect the winner of the popular game tic-tac-toe. Run this algorithm on the given test cases

(4) Camera captured document images are becoming more and more common in the current image processing scenario. A problem relating to document images is binarization of the image into foreground(text) and background(non-text) regions. Your task is to binarize the document images given in the folder into foreground and background regions. You can start with a simple solution and then build upon it (demonstrate the results with improved functions). Additionally, compare your output with the inbuilt MATLAB function for binarization.

(5) Write routines to implement the following three (3) tone-mapping approaches:

1) Linear Mapping: simply scale all radiance values, where min world radiance=0, and max world radiance=255, all values in between are linearly scaled.

2) Erik Reinhard's simple tone mapping approach based on the log-average luminance and a user supplied key 'a'. This technique is from equation1 and equation2 of Reinhard's paper (Reinhard, Erik, et al. "Photographic tone reproduction for digital images." ACM Transactions on Graphics (TOG) 2002).

3) Implement a Local Dodge and Burn technique. You can implement this anyway you want, but Section 3.2 in Reinhards paper is a reasonable starting point although a bit complicated. You can implement this exactly like the Tone Reproduction paper or try your own ideas. Essentially you should apply some heuristic to determine a local region to use for local image scaling based on the image content.

Input Images:AtriumNight.pfm, church.pfm, doll_doll.pfm, lips.pfm, rosette.pfm

Output: inputname_linear.bmp // Simple linear map.
inputname_key_X.bmp // X is the user key value; pick the key value you think is best.
inputname_local.bmp // Your local dodge-burn operator.

For each input image, generate the above three images. This task should have a total of 15 images output images. Several images are provided for you to try. All images in portable float map format. This format encodes world radiance (or what Reinhard et al. call world luminance $L_w(x, y)$). Maltab imread function cannot read this format. Code to read this is provided (getpfmraw.m); try: I=getpfmraw(filename.pfm) in matlab. This function returns back an RGB image, size: Height×Width×3. By the way, you wont be able to view these as images without some sort of tone-mapping.