Lecture Review and Quizzes (Due: Wednesday, January 17, 1:30pm)
Please review what you have learned in class and then complete the online quiz questions for the following sections on OpenEdX:\(^1\):

- Point Operations
- Histograms

---

Homework #1
Released: Monday, January 8
Due: Wednesday, January 17, 1:30pm

1. Displaying High Dynamic Range Images (Total of 9 points)

(a) Assume we have the imaging system shown above. A high dynamic range (HDR) camera takes a picture of the scene with a contrast ratio of 1000:1. The \(\gamma\)-predistortion circuit inside the camera is set to use \(\gamma = 3.0\). The acquired picture is shown on an image display (e.g., CRT or LCD) with \(\gamma = 2.0\). What is the contrast ratio of the image display required to accommodate the full dynamic range of the scene, without saturation in both the dark and bright portions of the image? \((3\ points)\)

(b) On the handouts webpage, you can find two HDR images `hw1_memorial.hdr` and `hw1_atrium.hdr`. Read each color image into MATLAB using function `hdrread` (if you have trouble using `hdrread`, we also provide the data in `hw1_memorial.mat` and `hw1_atrium.mat`, which you can read using function `load`). Convert to a grayscale image using function `rgb2gray`. Show the grayscale image using function `imshow`, submit the displayed image, and comment on which details in the image are easy/difficult to see. \((2\ points)\)

---

\(^1\) https://suclass.stanford.edu/courses/course-v1:Engineering+EE368+Winter2018
(c) Apply a $\gamma$-nonlinearity mapping to each grayscale HDR image from part (b) to reduce its dynamic range, show the new image, and submit the displayed image. For each image, find and report a value of $\gamma$ that allows you to see nearly all the details.

(2 points)

(d) Repeat part (c), but now apply $\gamma$-nonlinearity mappings to each of the red, green, and blue color components. First, use the same value of $\gamma$ for all color components. Then, experiment with different values of $\gamma$ for each of the color components. What is the effect of using different values of $\gamma$ for each color component compared to using the same value of $\gamma$ for all color components? You may submit images to facilitate your explanation.

(2 points)

Note: Please include relevant MATLAB code.
2. Denoising for Astrophotography (Total of 8 points)

Amateur astrophotographers often set up static cameras pointed toward particular regions of the night sky and record for an extended period of time. On the handouts webpage, you can find two videos `hw1_sky_1.avi` and `hw1_sky_2.avi`, which contain two recordings of the night sky each lasting a few minutes. Low light levels cause the video frames to be noticeably noisy.

(a) To generate a single denoised image from each video, compute a running average of the frames $f^t$ ($t=1,2,...$) in the video without frame alignment, according to the following update rule:

\[
\begin{align*}
    f_{\text{average}}^1 &= f^1 \\
    f_{\text{average}}^t &= \frac{t-1}{t} f_{\text{average}}^{t-1} + \frac{1}{t} f^t \quad t = 2,3,\ldots
\end{align*}
\]

To access the video frames in MATLAB, the following code can be used:

```matlab
vidobj = VideoReader('video.avi');
umFrames = get(vidobj, 'NumberOfFrames');
for i = 1 : numFrames
    frame = im2double(read(vidobj, i));
end
```

Display and submit $f_{\text{average}}$ at $t=30$ for each video. Comment on how effectively the noise is reduced and how much the sharp features are blurred by the averaging operation. (4 points)

(b) Now, compute a running average of the frames with frame alignment, according to the following update rule:

\[
\begin{align*}
    f_{\text{average}}^1 &= f^1 \\
    f_{\text{average}}^t &= \frac{t-1}{t} f_{\text{average}}^{t-1} + \frac{1}{t} \text{Align} \left( f^t, f_{\text{average}}^{t-1} \right) \quad t = 2,3,\ldots
\end{align*}
\]
Here, $\text{Align}(f,g)$ aligns frames $f$ and $g$ by minimizing the mean squared difference over a set of horizontal and vertical shifts. The following MATLAB code shows an example of how to horizontally and vertically shift a frame:

```matlab
dx = 1; % pixels
dy = -1; % pixels
A = [1 0 dx; 0 1 dy; 0 0 1];
tform = maketform('affine', A.');
[height, width, channels] = size(frame);
frameTform = imtransform(frame, tform, 'bilinear', ...
    'XData', [1 width], 'YData', [1 height], ...
    'FillValues', zeros(channels, 1));
```

Display and submit $f'_{\text{average}}$ at $t = 30$ for each video. Compare to the result in (a) and comment on how effectively noise is reduced while sharp features are better preserved.

(4 points)

Note: Please include relevant MATLAB code.
3. Image Subtraction for Tampering Detection (Total of 8 points)

Images of paintings are sometimes tampered to introduce subtle and plausible alterations. Please download the following images from the handouts webpage:

- **hw1_painting_1_reference.jpg**: reference image of the original painting *Irises*
- **hw1_painting_1_tampered.jpg**: tampered image of *Irises* with local modifications
- **hw1_painting_2_reference.jpg**: reference image of the original painting *Starry Night*
- **hw1_painting_2_tampered.jpg**: tampered image of *Starry Night* with local modifications

Detect the tampered regions for each painting by subtracting the tampered image from the reference image. Image alignment may be required prior to subtraction. For each painting, submit a binary image where the tampered regions are marked by white pixels and the non-tampered regions are marked by black pixels.

(8 points)

Note: Please include relevant MATLAB code.
4. Nighttime Road Contrast Enhancement (Total of 8 points)

The visibility of lane markings, road signs, and obstacles on the roads is significantly reduced at nighttime. To assist drivers in dark conditions, we can perform contrast enhancement on images captured by the car’s front-facing camera and display the enhanced images to the driver. On the handouts webpage, you can find three images captured at different times on different roads: hw1_dark_road_1.jpg, hw1_dark_road_2.jpg, and hw1_dark_road_3.jpg.

For each image, please perform the following operations and submit the required results.

(a) Plot and submit the histogram (MATLAB function: `imhist`) of the original image’s grayscale values. Briefly comment on the shape of each histogram. (2 points)

(b) Apply global histogram equalization to the original image (MATLAB function: `histeq`). Display and submit the modified image. Plot and submit the histogram of the modified image’s grayscale values. Comment on visually desirable/undesirable regions in the modified image. (3 points)

(c) Apply locally adaptive histogram equalization to the original image (MATLAB function: `adapthisteq`). Display and submit the modified image. Plot and submit the histogram of the modified image’s grayscale values. Choose and report the number of tiles and the clipping limit for attaining higher contrast while avoiding the generation of noisy regions and the amplification of nonuniform lighting effects. Comment on the subjective quality of the modified image compared to the result in (b). (3 points)

Note: Please include relevant MATLAB code.