Computational Photography
Project 2, Fall 2014

October 2, 2014

In this project you will capture high dynamic range (HDR) images, tone map them using bilateral filtering, and explore other applications of the bilateral filter. The project needs to be turned in by uploading on ILIAS. Submit material as described in the *turn-in instructions* for each part. The deadline is October 16, 14:00.

1. Capturing Multiple-Exposure HDR Images (3 points)

Your first goal is to capture your own multiple-exposure HDR images. You can work on this part in groups of up to three people. Please identify your collaborators. Contact Siavash Bigdeli bigdeli@iam.unibe.ch to borrow our camera if necessary. Second, find and download an application to create HDR images. Some examples we found are
Photosphere `http://www.anyhere.com/` Linux, Mac
Picturenaut `http://www.hdrlabs.com/picturenaut/` Mac, Windows
easyHDR `http://www.easyhdr.com/` Windows XP/Vista/7
Luminance HDR `http://qtpfsgui.sourceforge.net/` Mac, Windows

To capture multiple-exposure HDR images, use the automatic exposure bracketing mode (AEB) on your camera. Consult the manual of your camera to find out how to enable AEB or configure the camera for each shot manually. A sequence of three pictures should be sufficient. Find a scene with high contrast, for example a scene with both indoor and outdoor parts. Check first that the dynamic range of the scene is too large for a single exposure. Take a single picture and check on the back LCD that part of it is under- or overexposed. When this is confirmed, enable AEB and capture the exposure bracketing sequence. Ideally, use a remote shutter release to avoid camera shake.

Finally, you assemble a high-dynamic-range image using the software you found. Play with the parameters the application offers you and figure out which gives the best output for your scene. Save it into as an .hdr.

Turn-in: Your input images and the hdr files in a zip archive.

2. Tone Mapping using the Bilateral Filter (4 points)

Tone mapping reduces the dynamic range of HDR images to display them on low-dynamic range devices, such as conventional computer displays. In this assignment you implement

a tone mapping algorithm based on the bilateral filter. This algorithm only modifies the luminance of an image. It first decomposes the luminance image into a large-scale and a detail layer using the bilateral filter. Then it reduces the contrast of the large-scale layer while preserving local detail. All computation on luminance is performed in the log domain.

A key property of the bilateral filter is that it is edge preserving, i.e., it blurs images except across strong edges. For each pixel x, the output $J(x)$ of the filter is a weighted average of the neighboring pixels where the weight depends both on the spatial distance and the intensity difference:

$$J(x) = \frac{1}{k(x)} \sum_{\xi \in N_x} \exp\left\{-\frac{(\xi - x)^2}{2\sigma_s^2}\right\} \exp\left\{-\frac{(f(\xi) - f(x))^2}{2\sigma_r^2}\right\} I_\xi,$$

where $k(x)$ is a normalization term

$$k(x) = \sum_{\xi \in N_x} \exp\left\{-\frac{(\xi - x)^2}{2\sigma_s^2}\right\} \exp\left\{-\frac{(f(\xi) - f(x))^2}{2\sigma_r^2}\right\}.$$

Here, N_x is the neighborhood of x, x and ξ are pixel coordinates and $f(x)$ and $f(\xi)$ are the pixel intensities. The main parameters of the bilateral filter are the spatial standard deviation σ_s and the range standard deviation σ_r. The neighborhood N_x is usually a square window of size $(2w+1) \times (2w+1)$, where w is related to the spatial standard deviation by $w = \lceil 1.5\sigma \rceil$.

Tone mapping using bilateral filtering is summarized in the following pseudocode:

```
input_intensity = 1/61*(R*20+G*40+B)
r = R/input_intensity, g = G/input_intensity, b = B/input_intensity
log_base = bilateral(log(input_intensity))
log_detail = log(input_intensity)-log_base
compressionfactor = log(output_range)/(max(log_base)-min(log_base));
log_offset = -max(log_base) * compressionfactor;
log_output_intensity = log_base * compressionfactor+log_offset+log_detail
R_output = r*exp(log_output_intensity), same for G and B
```

The main parameter is the output range, which depends on the amount of remaining large-scale contrast that you want in the output. A value of 10 to 30 works well.

1. Implement bilateral filtering in a Matlab function `out=bfilt(im,sigma_s,sigma_r)`. The argument `im` is the input image, and `sigma_s` and `sigma_r` correspond to σ_s and σ_r in the above equations. You may realize that the bilateral filter can be quite slow. Therefore, make sure to use only two loops over image pixels. Do not implement filtering of each pixel using two more for loops. You can also speed up your filter using `parfor`, the parallel for-loop statement of Matlab, for the outer loop. Consult the Matlab help for details. Note that you need the Parallel Computing Toolbox of Matlab to take advantage of this.

2. Experiment with your implementation to understand the effect of the σ_s and σ_r parameters. Use an image (not an HDR image) of your choice. What happens when σ_r is large, what when it is small?

3. Implement tone mapping using bilateral filtering as described in the pseudo-code above. You can load HDR image using the Matlab function `hdrread`. Reasonable parameters for the bilateral filter are $\sigma_s = 2$ and $\sigma_r = 0.12$.

4. Implement tone mapping using Gaussian filtering instead of bilateral filtering. You can use a Gaussian filter generated using `fspecial('gaussian',21,8)` and `imfilter`.

5. Compare the results of tone mapping using bilateral and Gauss filtering. Use the HDR images you captured or the ones we provide. What do you observe?

Turn-in: Matlab code, a pdf file containing example images, the comparison between Gaussian and bilateral filtering, and your observations.

3. Two-Scale Photographic Tone Adjustment (3 points)

The bilateral filter has many other applications in image processing beyond tone mapping. In this part we will experiment with a simple algorithm to adjust photographic tone. A common technique in the photographic printing process, known as dodging and burning, is to manipulate the exposure of selected areas to increase or decrease the large scale contrast. Dodging decreases the exposure for areas of the print that the photographer wishes to be lighter, while burning increases the exposure to areas of the print that should be darker.

You will implement an algorithm that automatically performs tone adjustment. The main idea is to perform automatic tone adjustment by transferring the characteristics of an example image, called the *model*, to an input image. The bilateral filter is used again to separate both the model and the input image into a large scale and a detail layer. The tone of the model is then transferred to the input using histogram matching on the large scale layer. The algorithm proceeds as follows:

- Obtain the two-scale decomposition of both the input and model image.
- Transfer the histogram of the large-scale layer of the model to the large-scale layer of the input.
- Scale the detail layer of the input by a user adjustable factor.
- Obtain the output by summing the histogram matched large-scale and scaled detail layer of input.

You can obtain image histograms using `imhist` and perform histogram matching using `histeq`. For color images, transform them to YUV color space, process only on the Y channel, and then transform back.

1. Implement Matlab code for automatic tone adjustment using the algorithm described above.

2. Experiment with your implementation using different model and input images, both color and black and white. Document your experiments using images of the intermediate steps of the algorithm (large scale and detail layers, histogram matched large scale layer).

3. Compare your results with a simpler approach that directly matches the histograms of the model and the input images, without the two-scale decomposition and detail boosting. Describe your observations.

Turn-in: Matlab code. A pdf file containing results of your experiments as described above and text describing your observations.

4. Bonus (up to 1 point)

You can obtain one bonus point by completing at least two of the following four additional experiments with your code from exercises 2 and 3.

- Find a (free) software tool on the internet to perform tone mapping. Compare the results of this tool and your algorithm using five images. Write down your observations about the differences in a few sentences.
- Write a Matlab script that allows you to tone map an image with different sets of parameters. Run the script for at least three different sets of σs using each input image. Do this for 5 images. Write down your observations about the influence of the parameters in a few sentences.
- Find two "interesting" model histograms for tone adjustment. Match the histogram of five input images with the two models. Play also with the detail scale. Send us a gallery of "artistically" interesting results.
- Try applying histogram matchimg to color images by matching each RGB color chanel separately. Write down your observations in a few sentences.

Turn-in: A pdf file containingthe the results of your experiments and a few sentences describing your observations.