Problem Session 4

Feb. 3rd 2017

Orly Liba
Topics

• High dynamic range images
• Tone mapping
• Super-resolution
HDR & Tone mapping

Several 8-bit images acquired at different exposures ([0 255])

HDR algorithm

64-bit image
That has all important information

Tone mapping

8-bit image
That has all important information and can be viewed on a regular screen
Task 1: High dynamic range

(Debevec’s method)
Task 1: High dynamic range

Linearize the images using gamma of 2.2
Task 1: High dynamic range image

Computing the weights: we want to give a higher weight to pixels that are close to the center of the dynamic range at each exposure $k$.

$$w_{k,ij} = \exp\left(-4 \left(\frac{I_{k,ij} - 0.5}{0.5^2}\right)^2\right)$$

Compute this per color channel
Task 1: High dynamic range

Weights using Debevec’s method (all 3 color channels)
Task 1: High dynamic range image

Find a good estimation, $\hat{X}$, for the “true image”, $X$, using an optimization problem:

Minimize the difference, in log scale, between your result, $\hat{X}$, at different exposures $t_k$ and the acquired images at different exposures ($I_{lin_k}$). Multiplied by weight to indicate what’s more important.

$$\minimize_\hat{X} O = \sum_k w_k \left( \log(I_{lin_k}) - \log(t_k \hat{X}) \right)^2$$

Calculating the derivative of $O$, we get:

$$\hat{X} = \exp \left( \frac{\sum_k w_k \left( \log(I_{lin_k}) - \log(t_k) \right)}{\sum_k w_k} \right)$$
Task 1: High dynamic range

Fuse all exposures into a single, floating-point HDR image.
Normalize by the maximal value in the image (values range [0 1]).
Display the image (imshow). It won’t look very good on your screen, because it’s an LDR display.
Adjust gamma ($\gamma$) and scale ($s$) to show that more details are captured in the HDR image than in any of the individual exposures.

$$I_{HDR} = (s \times I_{HDR\_linear})^\gamma$$
Task 1: High dynamic range

Adjusting $\gamma$

$\gamma = 1/10$

$\gamma = 1/6$

$\gamma = 1/4$
Task 1: High dynamic range

Optimizing for maximal detail

\[ \gamma = \frac{1}{5}, s = 3 \]

Try to get better results than this!
Task 2: Tone mapping

HDR image: more than 256 levels (x3 for three channels).
Tone mapping is done to display the image on an LDR monitor so that most detail of the image will be visible.

Implement three types of tone mapping.
Template code is provided.

The template and worksheet provide very detailed hints.
Task 2: Tone mapping

Method #1: Global tone mapping
Apply a similar gamma on all color channels (same as previous question).

Applying a gamma curve to the image in linear scale = scaling on the image in log scale (perceptually linear scale).
When the power $\gamma < 1$, we get compression.

Save and plot the resulting image and report your choice for gamma. Comment on which details are visible and also on the color saturation of the tone mapped image.
Task 2: Tone mapping

Why are the colors washed out?

Because, we’ve compressed them along with the intensity!

The two alternative methods take care of that.
Task 2: Tone mapping

Method #2: Global gamma on intensity only
Intensity = weighted average of all color channels
\[ I_{\text{intensity}} = \frac{(20R+40G+B)}{61} \]
Chrominance = (R/I, G/I, B/I)

Apply gamma on the intensity and then merge the tone mapped intensity with the original chrominance.

Apply gamma correction of 2.2 to the result.

Comment on which details are visible and also on the color saturation of the tone-mapped image.
Task 2: Tone mapping

Method #3: Local tone mapping

Apply a non-linear bilateral filter to the intensity. Only the “large” edges (base) are compressed by applying a gamma curve.

- We look at the log of the intensity,

$$L = \log_{10}(I_{int}).$$
Task 2: Tone mapping

Method #3: Local tone mapping

- Calculate the bilaterally filtered image (base layer, $B$). Only large edges remain.
- You can use the bilateral filter from HW2
- You should figure out the parameters (both sigmas) to achieve good filtering. Use this result as an example.
Task 2: Tone mapping

Method #3: Local tone mapping

• The detail layer is what’s left after subtracting the base from the original intensity.
• $D = L - B$

→ We don’t want to compress texture
Task 2: Tone mapping

Method #3: Local tone mapping

- Apply offset and scale to the base layer.
  \[ \text{BB} = (B - o) \times s \]

O: offset such that the maximal intensity is 1 in linear scale (zero in log scale)

S: scale the base layer. \( s \) is the ratio of the target (\( dR \)) and the contrast in the base layer. Larger values preserve more contrast. Divide by the full range of \( B \) (\( \max(B) - \min(B) \)) and multiply by \( dR \).

- Keep in mind: scaling in log domain is power in linear domain

Notice the scale
Task 2: Tone mapping

Method #3: Local tone mapping

• Merge with the detail layer and convert back to linear scale.
• Merge back with the chrominance.
• Apply gamma of 2.2

• Don’t forget to describe the results
Tone mapping, Method #3: different scaling \((dR)\)
Task 3: Superresolution

Several shifted low-resolution images  \rightarrow \text{High resolution image}
Task 3: Superresolution

Implement super-resolution. Provided are four low-resolution images and also a reference high-resolution image that these were generated from. Each pixel of the low-resolution images averages over four pixels in the target image. See schematic below.
Task 3: Superresolution

4 Low resolution images
Task 3: Superresolution

Formulate as an optimization problem:

\[ x = \text{high resolution image we’re trying to find} \]
\[ b = \text{provided sub-sampled images} \]
\[ A = \text{the subsampling operator} \]

Minimize this residual:
\[ res = F(x) = \sum (Ax - b)^2 \]

Find \( x \), such that when subsampled we get \( b \)
Task 3: Superresolution

Minimize this residual:

$$\text{res} = F(x) = \sum (Ax - b)^2$$

Solve with gradient descent

Gradient descent step:

$$x_{\text{new}} = x - \alpha \nabla F(x)$$
$$x_{\text{new}} = x - \alpha A^T(Ax - b)$$

https://en.wikipedia.org/wiki/Matrix_calculus

The step update
Task 3: Superresolution

Super-resolved

Ax() subsampling

Atx() down-sampling

Project the low resolution images back to the grid of the high resolution image and average

Down-sampled
Task 3: Superresolution

What you will learn in this task is to set up and solve a large-scale linear equation system via a **simple iterative inverse method: gradient descent**.

For this method, **don’t** compute the matrix explicitly - large-scale linear systems will not allow you to do that in practice.

Rather you write two functions $A\mathbf{x}(\cdot)$ and its inverse, $A^T\mathbf{x}(\cdot)$.

Template code is provided and a part of the function $A\mathbf{x}(\cdot)$ is already implemented for you.

You need to write the function $A^T\mathbf{x}(\cdot)$ (hints: use imresize, with option ‘nearest’ and circshift).

Clarify $A^T(x)$ is inverse of $A(x)$. Either copy pixels or use imresize. The circshifted rows need to be zero.
Task 3: Superresolution

Calculate the residual and PSNR as a function of the iteration number.

The residual from iteration to iteration should always decrease and the PSNR should always increase.

Notice that saving to png introduces some errors, and that PSNR is higher when sampling directly.
Task 3: Superresolution
Task 3: Superresolution

More hints:

• Initial guess of $x$ can be zero

• The step size, is usually a key parameter of gradient descent, use $\alpha = 1$. Feel free to play with this parameter and see what happens (is convergence faster or slower? Are there step sizes in which there is no convergence?)

• Number of iterations can be limited by 100

• Concatenate the four subsampled images to one large image, so that the functions $Ax$ and $Atx$ take care of all of them

• The calculation is done on all three color channels simultaneously
Bonus: your own HDR image

Around ~10 images with different exposures
Have a nice weekend!

And good luck with the homework!

Stanford Memorial Church (HDR)  https://www.flickr.com/photos/scotlofesness/4334766965