Problem Set 1

Assigned: Feb 9, 2006
Due: Feb 23, 2006

Note

The problems marked with "*6.882 only*" are for the students who register for 6.882. (Of course, students registering for 6.098 may do these problems according to their interests, but they will not be graded.) The problems marked "*extra credit*" are for both 6.882 and 6.098 students. For those who are not familiar with image processing/manupulation in MATLAB, please see the appendix to this problem set for a brief introduction.

Students are encouraged to collaborate on the assignments. However, each student should individually do the coding and the write-up. Please hand in solutions on the due date, and email the code and results (in a zipped file) to TA. In the email please also indicate how much time you took on this problem set.

Please go to webpage `http://groups.csail.mit.edu/graphics/classes/CompPhoto06/`, download and unzip file `ps1.zip` under the link "problem sets and solutions".

Problem 1 *Pinhole Camera*

(a) With a pinhole camera, if you photograph a brick wall, with the sensor plane parallel to the brick wall, will the lines of the bricks be parallel or converging in the resulting photograph?

(b) Suppose you move the camera during an exposure. Under each of the following types of camera motion, will the amount that an object is blurred depend on its distance away from the camera (and why)

(i) rotation about a vertical axis through the pinhole (vertical axis assumed to be parallel to the sensor plane).
(ii) rotation about the optical axis.
(iii) translation in a direction parallel to the sensor plane.

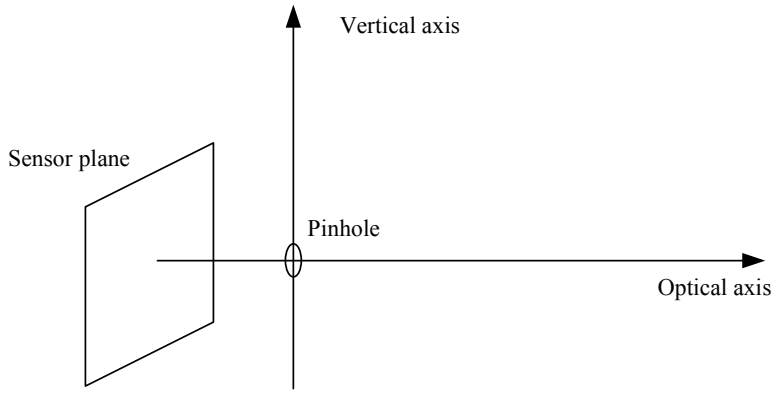

Figure 1: Pinhole camera schematic, defining the vertical and optical axes.

(iv) translation perpendicular to the sensor plane.

(c) For a pinhole camera, how will the image change from one obtained using a small circular aperture (ignoring diffraction effects):

(i) if you double the size of the aperture?

(ii) if you change the aperture to a thin vertical slit?

Problem 2 *Warm-up Matlab Assignment*

Unzip file `ps1.zip`. Load image `street.jpg` into Matlab. This is the output data taken by a CCD color camera before tonescale adjustment. Apply a "gamma" tonescale correction of $I_{out} = I_{in}^{\gamma}$ where $\gamma = 0.5$. Display the two images side-by-side in your answer document.

Problem 3 *Color*

Load image `desk-orange.jpg` into MATLAB. The image is from web: `http://www.scifun.ed.ac.uk/card/images/left/desk-orange.jpg`. This is a photograph taken under Tungsten illumination. (Usually color correction is done before the tonescale adjustment, but for convenience in viewing the images, we will apply it after tonescale adjustment here.)

"Von kries adaptation" refers to restricting the 3x3 color transformation matrix to have zeros on all off-diagonal terms. What 3 entries for "von kries

adaptation" color scaling is needed to provide proper color balance to this image, under the assumption of

(a) "gray world"–the average pixel intensity for each color channel is the same for all 3 color channels. To minimize the overall brightness change, assume that the average pixel value of the color balanced image is $[c, c, c]$ where c is the maximum of the mean value per each uncorrected RGB channel. Show, side-by-side, the uncorrected and color corrected images.

(b) the brightest pixel value in each color channel in the image is white? Show, side-by-side, the uncorrected and color corrected images.

(c) (*6.882 only*) For a generic, color-balanced image (not specific to the image above), what 3x3 matrix, applied to every pixel, will increase the color saturation of the image? Your answer should be a formula for the matrix, parameterized by the amount of color saturation desired. Load image `harbor.bmp` into Matlab, apply the transform, and display the two images side by side.

Problem 4 *Demosaicing*

Given image `watchtower.jpg`:

(a) synthesize the image observed through YGC striped sampling pattern.

(b) form the RGB demosaiced image, using a linear interpolation algorithm for demosaicing.

(c) form the RGB demosaiced image, using the median filter interpolation algorithm.

Assume

$$\begin{bmatrix} Y \\ G \\ C \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0 & 1 & 0 \\ 0 & 0.5 & 0.5 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

(d) (*extra credit*) Put yourself in the "shoes" of an image demosaicing algorithm. Guess (by any means–eyeballing it, by plotting color values, by using a color interpolation algorithm) what the true values are for the missing color samples in `waterfall.bmp`. Assume that an initial color interpolation has been performed for the surrounding pixels and that we only have to infer the missing color values for the 4 central pixels of the image patch, which lies at the center of the image, or `im(64:65,64:65,:)` in MATLAB notation.

Your answer to this problem should be 8 numbers: the 2 missing colors in each of the 4 pixels that are missing color samples in the image patch. (The color sampling was made using a Bayer sampling pattern).

Scoring for this extra credit problem will be a function of the squared error distance between your estimated values and the *actual* pixel values that were present in the original image.

Problem 5 *Color Metamerism (Not required; Extra Credit)*

The Matlab file `CIE.mat` contains the spectra which will be needed for this problem. The vectors c_x, c_y, and c_z give the CIE color matching functions for the X, Y, and Z coordinates, respectively. These functions are sampled at 5nm intervals for wavelengths from 400 through 700 nm.

Suppose you have a test color specified as a linear combination of three spectral primaries at 420, 520, and 620 nm, given by $(a, b, c)^T$, respectively. In terms of a, b, and c, what linear combination of light sources at 460, 510, and 590 nm is needed to give a perceptual match to the test color?

Appendix *Image Operation in MATLAB*

MATLAB is a great tool for scientific computation. There are three reasons we choose it as the programming language for the class.

- MATLAB code is very compact. The learning curve is very short. Because matrix operation is embedded, it is straightforward to code and easy to debug. There are academic papers coded in only a few lines (ICCV) or less than a hundred lines (Siggraph).
- It is cross-platform for Windows, Linux and Mac OS. For most of the code it requires no change to be transplanted from one operating system to another.
- MATLAB provides a rich library of toolboxes which covers a broad range of scientific computation. Besides, there are many public MATLAB code downloadable from the internet.

But MATLAB is also notorious for loops. If you are a Pascal, C or C++ coder, you may find it a little difficult to get used to MATLAB coding at

beginning. Relax! After reading this tutorial you will get a glimpse of how to manipulate images correctly and efficiently in MATLAB.

In `ps1.zip` there is a file called `watchtower.jpg`. Type in the following code in MATLAB command window:

```
im=imread('watchtower.jpg');
im=im2double(im);
imshow(im);
```

Function `imread` can read the typical image format such as bmp, jpg, tiff, gif... It returns an image array of data type `uint8` (which is equivalent to `unsigned char` in C++). But many MATLAB functions require the input image to be `double`. So function `im2double` is called to convert data type from `uint8` to `double`. Function `imshow` is to display the image in a window. Call `imwrite` to save image in arbitrary format.

Now we try the following commands

```
[height,width,nchannels]=size(im);
X=reshape(im,[height*width,nchannels])';
A=[0 0 1;0 1 0;1 0 0]
Y=A*X;
Im=reshape(Y',[height width nchannels]);
figure;imshow(Im);
```

MATLAB stores a matrix in a 1D buffer. For images, the dimension is [height x width x nchannels] where nchannels=1 or 3. Function `reshape` is used to change the dimension, but the 1D buffer is not changed. In MATLAB symbol `'` right after a matrix means transpose. So the first two lines convert the input image to a matrix with three rows. Each row correspond to R,G,B channels. Matrix `A` is defined to switch R and B channels. After matrix multiplication and matrix reshaping we obtain another image `Im`. Note that

```
X=reshape(im,[nchannels,height*width]);
```

is different from `X=reshape(im,[height*width,nchannels])';`. Why? In MATLAB the matrix is stored from the first dimension to the last, or in other word, concatenation of column vectors. And function `reshape` doesn't change the 1D buffer. So in this way the first column of `X` does not correspond to the RGB values of the first pixel, but the R value of the first three pixels in column 1.

Type in the following commands

```
Im=im(:,1:2:end,:);
figure;imshow(Im);
```

MATLAB notation `1:2:end` means to sample every other element from the first to the end. This one together with `repmat` and `kron` are useful tools to squeeze or expand matrices. Essentially `Im=im(:,1:2:end,:)` samples the odd columns from `im`.

Finally,

```
im_interp=reshape([Im;Im],[height,width,3]);
im_interp(:,2:2:end-1,:)=(Im(:,1:end-1,:)+Im(:,2:end,:))/2;
figure;imshow(im_interp);
```

Now we try to expand the width-reduced image `Im` to the original size. In the first line, `im_interp` is simply the duplicate of the columns. In the second line, the even columns are linearly interpolated from the two neighboring odd columns.

All the commands in this tutorial can be found in `appendix.m`. You may run and see how it works! In addition, MATLAB provides a lot of help and examples for the functions and how to use them. Having been using MATLAB for years, I am still using help everyday! Please don't hesitate to write to TA if you encounter any MATLAB problems.