Computer Vision **CS 543 / ECE 549**

Homework 1
Due Feb 15, 2011

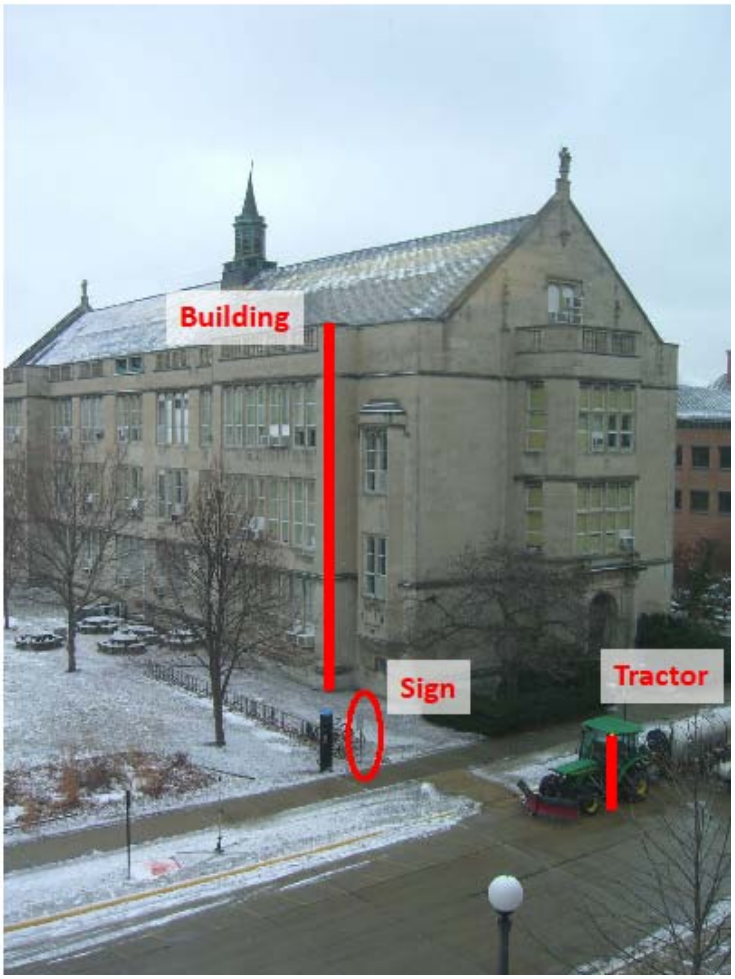
Answer the following questions and explain solutions. Numbers in parentheses give maximum credit value. You can work in small groups, but turn in individual solutions and indicate collaborators. Do not use code from the Internet.

Turn in assignments on Tuesday, Feb 15. *Electronic materials (e.g., pdf) should be submitted by 12:00pm by e-mailing Ian Endres with the subject line: CS543:HW1. Hard copies can be handed in at the start of class, 2pm.* You can turn in a mix of hard copy and electronic (e.g., hard copy for writing down equations, electronic for images), but if you do so, indicate on your hardcopy which portions you sent in electronic format.

1. **Single-View Metrology (50%)**
   A. For the Kyoto Street image, shown above, estimate the positions (in the image plane) of the three major orthogonal vanishing points, corresponding to the building orientations. Use at least three manually selected lines to solve for each vanishing point. The included code getVanishingPoint_shell provides an interface for selecting and drawing the lines, but the code for computing the vanishing point needs to be inserted.
      - Plot the points and the lines used to estimate them on the image plane. (5%)
      - Specify the points (u,v). (5%)
      - Plot the ground horizon line and specify its parameters in the form $au + bv + c = 0$. Normalize the parameters so that: $a^2 + b^2 + c^2 = 1$. (5%)

   B. Use the fact that the vanishing points are in orthogonal directions to estimate the camera focal length (f) and optical center (u_0, v_0). Show all work. (10%)

C. Show how to compute the camera's rotation matrix when provided with vanishing points in the X, Y, and Z directions. (5%)
Now, compute the rotation matrix for this image, setting the vertical vanishing point as the Y-direction, the right-most vanishing point as the X-direction, and the left-most vanishing point as the Z-direction. (5%)



D. The above photo is of the University High building, taken from the third floor in Siebel Center facing south. Estimate the horizon and draw/plot it on the image. Using a reference object, such as the sign, estimate the heights of the tractor, the building, and the camera (in meters). This can be done with powerpoint, paper and a ruler, or Matlab. You can measure the reference object with a tape measurer (feel free to share measurements with other students).
   - Turn in an illustration that shows the horizon line, and the lines and measurements used to estimate the heights of the building, tractor, and camera. (8%)
   - Report the height of your reference object, as well as the estimated heights of the building, tractor, and camera. (7%)

2. Lighting (20%)

A. Answer the following regarding the image on the left (photo credit: *dolmansaxlil* from Flickr). Consider shadows, specularities, albedo, surface orientation, light sources, etc. (10%):
   1. Why is (2) brighter than (1)? Each points to the asphault.
   2. Why is (4) darker than (3)? 4 points to the marking.
   3. Why is (5) brighter than (3)? Each points to the side of the wooden block.
   4. Why isn't (6) black, given that there is no direct path from it to the sun?

B. In the photo on the right (photo credit: *fictures* from Flickr), circle the location of a specularity. Open the image in a photo editor or Matlab and use a specularity to estimate the color of the light. Report the color as normalized RGB with $R^2 + G^2 + B^2=1$. Explain how you got this estimate and any reasons it might be inaccurate. (6%)

C. Suppose you are looking down at a small lake on a bright day, and you see the reflection of the trees on the other side of the lake. Why do you see the reflection and not what is in the lake, even though water is highly translucent? (4%)

3. Image Filtering (20%)

Choose an image that has an interesting variety of texture (from Flickr or your own images). The image should be at least 640x480 pixels.

1) Convert it to grayscale.
2) Create a Gaussian pyramid with five levels and a scale factor of 2. Turn in the images (within a document) and pseudocode. (7%)
3) Using the result of (2), create a Laplacian pyramid with at least five levels. Turn in the images and pseudocode. (7%)
4) Compute the 2D FFT for the original image, the Gaussian pyramid, the Laplacian pyramid, and the Gaussian filter. Display the images, and explain the frequency interpretation of the Gaussian and Laplacian pyramid (6%).

You may find the following Matlab functions useful: `imagesc(im,[minval maxval])`, `rgb2gray`, `im2double`, `fft2`, `subplot`

Your displayed images for the Gaussian and Laplacian pyramids may look something like this:

4. Vision and its Applications (10%)

Think of one important task that you use vision for. Write down the goal of that problem. Then, write down all of the visual subtasks that you need to solve to accomplish the goal. Then, write down the subtasks of the subtasks until you feel that you have specified all of the steps necessary to perform the high-level task. Write a high-level pseudocode algorithm.

Below, I provide a simple example. You must choose a different example. Your answer should be roughly the same length. Careful not to choose too vague of a task, such as scene understanding; otherwise, you may have to write a book ☺.

Example: Reading a Page of Text

When we read books, the goal is to extract information from printed pages. Reading requires that we decompose the page into the background and into characters. The characters need to be grouped together into words or short phrases and recognized. If the words are unfamiliar, it may be necessary to identify the individual characters in the word and to recognize each one, possibly by computing edge features and comparing the features to known characters from the past. It may be possible to recognize familiar words and phrases as a single pattern. Once I have converted the image of the document into plain text, I would then need to apply natural language skills and background knowledge to draw information from the document.

Algorithm to read a page of text:

1) Extract characters from the page
2) Group the characters into words and short phrases (still image patches at this point)
3) Recognize individual characters and/or whole words, translating pixels into letters and words
    a. Compute features, such as edge patterns for each token to be recognized
    b. Apply a categorizer (assume one is pre-trained) to recognize each token
4) String together the text to create a full document. Apply NLP tools to extract information.