CSE 252A – Computer Vision – Homework 1
Instructor: Ben Ochoa
Maximum Points : 50
Deadline : 11:59 p.m., Tue, 20-October-2015

Instructions:

- Work in groups of 2 persons for this assignment (not individually)
- Submission instructions :

  1. Submit your homework electronically by email to both Nick Kinkade [`nkinkade@eng.ucsd.edu`] and Akshat Dave [`akdave@eng.ucsd.edu`]
  2. Set the subject line to be `CSE 252A Homework 1`.
  3. The email should have one file attached, named `CSE252A_HW1__studentid1__studentid2.zip`. The contents of the file should be:
     (a) A `pdf` file with your writeup. This should have all code attached in the appendix. Name this file: `CSE252A_HW1__studentid1__studentid2.pdf`.
     (b) All of your source code in a folder called `code`.

  The code is thus attached *both* as text in the writeup appendix and in the `code` folder of the compressed archive.
- There is no physical hand-in for this assignment.
- Coding is to be done only in MATLAB.
- In general, MATLAB code does not have to be efficient. Focus on clarity, correctness and function here, and we can worry about speed in another course.

1 Perspective Projection [6 pts]

Consider a perspective projection where a point $P = [x, y, z]^T$ is projected onto an image plane Π' represented by $k = f' > 0$ as shown in the Fig. 1.

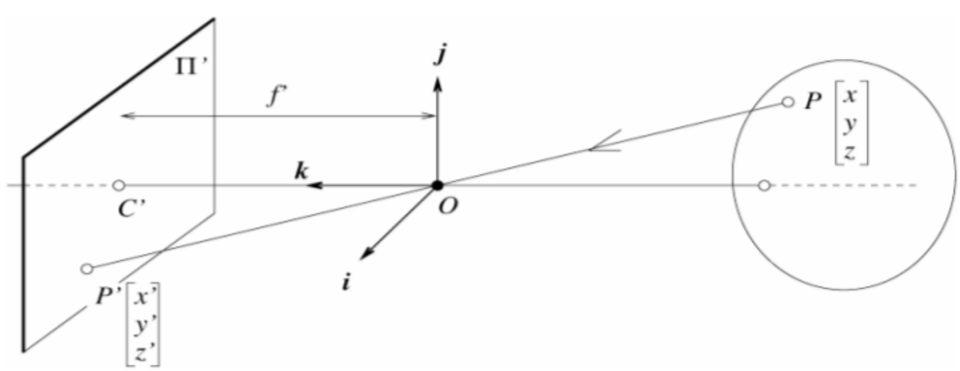

Figure 1: Image formation

Given the coordinate frame $e = [\mathtt{i}, j, k]^T$, consider the projection of a ray Q in the world coordinate system

$$Q = \mathbf{p_1} + s\ \mathbf{p_2}$$

where $\mathbf{p_1} = \left[4, -7, 0\right]^T$, $\mathbf{p_2} = \left[0, 2, 1\right]^T$, and $s \in (-\infty, -1]$.

1. Calculate the coordinates of the endpoints of the projection of the ray onto the image plane. [2pt]

2. Find the equation of a ray $L \neq Q$ that is parallel to Q. How did you arrive at this equation? [2pt]

3. Can you find the point where the rays L and Q meet? If so, what is that point? [Hint:Think in terms of projective geometry] [2pt]

2 Image formation and rigid body transformations [12 points]

In this problem we will practice rigid body transformations and image formations through the projective camera model. The goal will be to 'photograph' the following four points given by ${}^A\mathbf{P}_1 = (-1, -0.5, 2)^T$, ${}^A\mathbf{P}_2 = (1, -0.5, 2)^T$, ${}^A\mathbf{P}_3 = (1, 0.5, 2)^T$, ${}^A\mathbf{P}_4 = (-1, 0.5, 2)^T$ in world coordinates. To do this we will need two matrices. Recall, first, the following formula for rigid body transformation

$$ {}^B\mathbf{P} = {}^B_A\mathbf{R}\ {}^A\mathbf{P} + {}^B\mathbf{O}_A \tag{1}$$

where ${}^B\mathbf{P}$ is the point coordinate in the target (B) coordinate system, ${}^A\mathbf{P}$ is the point coordinates in the source (A) coordinate system, ${}^B_A\mathbf{R}$ is the rotation matrix from A to B, and ${}^B\mathbf{O}_A$ is the origin of coordinate system A expressed in the B coordinates. The rotation and translation can be combined into a single 4×4 *extrinsic parameter* matrix, $\mathbf{E}$, so that ${}^B\mathbf{P} = \mathbf{E}\ {}^A\mathbf{P}$. Once transformed, the points can be photographed using the *intrinsic camera* matrix, $\mathbf{K}$ of dimensions 3×3. Once these are found, the image of a point, ${}^A\mathbf{P}$, can be calculated as $\mathbf{K}[\mathbf{I}|\mathbf{0}]\ \mathbf{E}\ {}^A\mathbf{P}$. We will consider four different settings of focal length, viewing angles and camera positions below. For each of these calculate:

- the extrinsic transformation matrix E,
- intrinsic camera matrix K under the perspective camera assumption.
- Calculate the image of the four vertices and plot using the supplied `plotsquare.m` function (see e.g. output in figure 2).

Camera Settings:

1. **No rigid body transformation [3 pts]:** . Focal length = 1. The optical axis of the camera is aligned with the z-axis.

2. **Translation [3 pts]:** ${}^BO_A = (0, 0, 1)^T$. The optical axis of the camera is aligned with the z-axis.

3. **Translation and rotation [3 pts]:**. Focal length = 1. ${}^B_AR$ encodes a 60 degrees around the z-axis and then 45 degrees around the y-axis. ${}^BO_A = (0, 0, 1)^T$.

4. **Translation and rotation, long distance [3 pts]:**. Focal length = 15. ${}^B_AR$ encodes a 60 degrees around the z-axis and then 90 degrees around the y-axis. ${}^BO_A = (0, 0, 13)^T$.

Note: we will not use a full intrinsic camera matrix (e.g. that maps centimeters to pixels, and defines the coordinates of the center of the image), but only parameterize this with f, the focal length. In your report, include a image like Figure 2.

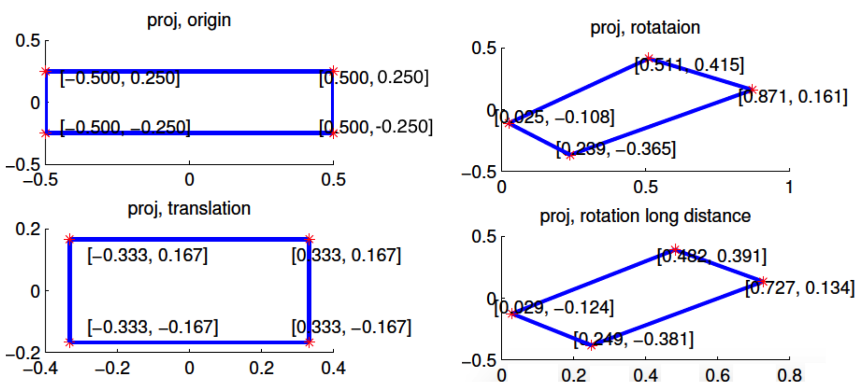

Figure 2: Example output for image formation problem. Note: the angles and offsets used to generate these plots are different from those in the problem statement, it's just to illustrate how to report your results.

3 Rendering [14 points]

In this exercise, we will render the image of a face with two different point light sources using a Lambertian reflectance model. We will use two albedo maps, one uniform and one that is more realistic. The face heightmap, the light sources, and the two albedo maps are given in `facedata.mat` (each row of the `lightsource` variable encode a light location).

Note: Please make use out of `subplot.m` to display related image next to each other.

1. **Plot the face in 2-D [2 pts]:** Plot both albedo maps using `imagesc.m`. Explain what you see.

2. **Plot the face in 3-D [2 pts]:** Using both the heightmap and the albedo, plot the face using `surf.m`. Do this for both albedos. Explain what you see.

3. **Surface normals [5 pts]:** Calculate the surface normals and display them as a quiver plot using `quiver3.m`. You may have to subsample while using `quiver3.m` to make it look nice. Recall that the surface normals are given by

$$[-\frac{\delta f}{\delta x}, -\frac{\delta f}{\delta y}, 1].$$

   Also, recall, that each normal vector should be normalized to unit length.

4. **Render images [5 pts]:** For each of the two albedos, render three images. One for each of the two light sources, and one for both light-sources combined. Display these in a 2×3 subplot figure with titles. Recall that the general image formation equation is given by

$$I = a(x, y)\langle \hat{n}(x, y), \hat{s}(x, y)\rangle \frac{s_0}{d^2(x, y)}$$

   where $a(x, y)$ is the albedo for pixel x, y, $\hat{n}(x, y)$ the surface normal, $\hat{s}(x, y)$ the light source direction, s_0 the light source intensity, $d(x, y)$ the distance to the light source, and $\langle \cdot, \cdot \rangle$ denotes the dot product. Use `imagesc.m` to display these images. Let the light source intensity be 1 and do *not* make the 'distant light source assumption'.

4 Photometric Stereo [18 points]

The goal of this part of the assignment is to implement an algorithm that reconstructs a surface using the concept of photometric stereo. Your program will take in multiple images as input along with the light source direction for each image.

Implement the photometric stereo technique described in Forsyth and Ponce 5.4 and the lecture notes. Your program should have two parts:

(i) Read in the images and corresponding light source directions, and estimate the surface normals and albedo map.

(ii) Reconstruct the depth map from the normals. You can first try the naïve scanline-based shape by integration method described in the book. If this does not work well on real images, you can use the matlab implementation of the Horn integration [source-link] technique.

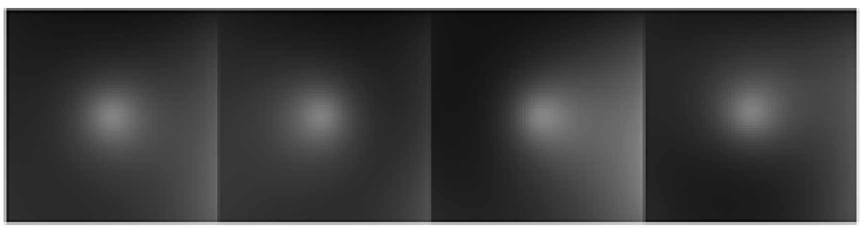

Figure 3: Synthetic Data

Try this out on the synthetic dataset (synthetic_data.mat) with :

(i) A subset of 3 images, viz. {`im1`, `im2`, `im4`}

(ii) All four images (Most accurate)

In your report, include the following results:

(i) Estimated albedo map [3 pts $\times 2 = 6$ pts]

(ii) Estimated surface normals by either showing [3 pts $\times 2 = 6$ pts]

- Needle map (you will need to subsample the image to get a needle map which can be displayed). You can use the matlab functions `meshgrid` and `quiver3`
- Three images showing three components of surface normal

(iii) A wireframe of a depth map for which you may use MATLAB's `surf` function. [3 pts $\times 2 = 6$ pts]

Resources

All files are on the course webpage.