CSE152 – Computer Vision – Assignment 1 (SP16)
(Revision 2)
Instructor: Ben Ochoa
Maximum Points : 50
Deadline : 11:59 p.m., Thursday, 21-April-2016

Instructions:

- This assignment must be solved, and written up individually
- There is no physical hand-in for this assignment.
- Coding for this assignment must be done in MATLAB
- All code developed for this assignment must be included in the appendix of the report. If the appendix does not contain your code, points may be deducted.
- Submit your assignment electronically by email to Akshat Dave `[akdave@ucsd.edu]` **and** Mihir Patankar `[mpatanka@eng.ucsd.edu]` with the subject line *CSE152-Assignment-1*. The email should contain one attached file named [`CSE_152_HW1_<student-pid>.zip`]. This zip file must contain the following two artifacts:
  1. A pdf file named [`CSE_152_HW1_<student-pid>.pdf`] containing your writeup. Please clearly state the author's full name and student identity in the report. The report must contain all the result images. Images not embedded in the report will not be considered for evaluation. In the case of figures containing text and labels, it is your responsibility to ensure that the text is readable; points may be deducted if these are not readable. Regarding code, in general, MATLAB code does not have to be efficient. Focus on clarity, correctness and functionality here, we can worry about speed in another course.
  2. A folder named [`CSE_152_HW1_<student-pid>_code`] containing all your matlab code files

Introduction

The purpose of this assignment is to gain some insights into the mathematical underpinnings of projective geometry, image formation and rendering. While some problems are theoretical, others require development of code. It is permitted to solve the mathematical problems on paper, scan them and attach these as images into the report. For the programming problems, do not forget to include your code in the appendix and your results in the main report.

1 Geometry [20 points]

Consider a line in the 2D plane, whose equation is given by $ax + by + c = 0$. This can equivalently be written as $\tilde{\mathbf{l}} \cdot \bar{\mathbf{x}} = 0$, where $\tilde{\mathbf{l}} = (a, b, c)^T$ and $\bar{\mathbf{x}} = (x, y, 1)^T$. Noticing that $\bar{\mathbf{x}}$ is a homogeneous representation of $\mathbf{x} = (x, y)^T$, we can view $\tilde{\mathbf{l}}$ as a homogeneous representation of the line $ax+by+c = 0$. We see that the line is also defined up to a scale since $(a, b, c)^T$ and $\sigma[a, b, c]^T$ with $\sigma \neq 0$ represents the same line. All points (x, y) that lie on the line $ax + by + c = 0$ satisfy the equation $\tilde{\mathbf{l}} \cdot \bar{\mathbf{x}} = 0$.

$$\text{A point } \bar{\mathbf{x}} \text{ lies on the line } \tilde{\mathbf{l}} \Leftrightarrow \tilde{\mathbf{l}} \cdot \bar{\mathbf{x}} = \bar{\mathbf{x}} \cdot \tilde{\mathbf{l}} = 0 \qquad \texttt{(Statement 1)}$$

1. [4 points] Using Euclidean coordinates, find the equation of the line perpendicular to the family of lines $y = x + \lambda,\ \lambda \in (-\infty, \infty)$ and at a distance D from the origin. Your answer should be represented only in terms of the given parameters.

2. [6 points] Prove the following two statements (using homogeneous coordinates) that follow from `(Statement 1)`:
   (a) The cross product between two points gives us the line joining the two points
   (b) The cross product between two lines gives us their point of intersection
3. [4 points] What is the line, in homogeneous coordinates, joining the points $(1, 4)$ and $(4, 5)$.
4. [6 points] When a rectangle $ABCD$ is observed under pinhole perspective, the image will be an arbitrary quadrilateral $A'B'C'D'$. Answer the following questions working with homogeneous representations.
   (a) [3 points] For any arbitrarily imaged rectangle $ABCD$ with non zero area, can $A'B'C'D'$ ever be a non-convex quadrilateral? Explain the intuition behind your answer. (Note : A convex polygon is a simple polygon (not self-intersecting) in which no line segment between two points on the boundary ever goes outside the polygon.)
   (b) [3 points] Let $A' = (t, t)$, $B' = (t, 6t)$, $D' = (2t, 4t)$ and $C' = (4t, 6t)$ be the vertices of the image. Find all the vanishing points of the quadrilateral (i.e. the points of intersections of pairs of opposite lines through $\{A'B', C'D'\}$ and $\{B'C', A'D'\}$) given $t = 1$.

2 Image formation [10 points]

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

$$ {}^{C}\boldsymbol{P} = R\ {}^{W}\boldsymbol{P} + \boldsymbol{t} \tag{1} $$

where ${}^{C}\boldsymbol{P}$ is the point position in the target (camera) coordinate system, ${}^{W}\boldsymbol{P}$ is the point position in the source (world) coordinate system, R is the rotation matrix from the world frame to the camera frame, and $\boldsymbol{t}$ is the position vector of the origin of world coordinate system expressed in the camera coordinates. The rotation and translation can be combined into a single 4×4 *extrinsic parameter* matrix, T, so that ${}^{C}\boldsymbol{P} = T\ {}^{W}\boldsymbol{P}$ (where ${}^{C}\boldsymbol{P}$ and ${}^{W}\boldsymbol{P}$ are now in homogeneous coordinates). Once transformed, the points can be photographed using the *intrinsic camera* matrix, K which is a 3×3 matrix. Once these are found, the image of a point, ${}^{W}\boldsymbol{P}$, i.e. $\boldsymbol{p}$, can be calculated as $\boldsymbol{p} = K(Id\ \boldsymbol{0})\ T\ {}^{W}\boldsymbol{P} = K(R\ \boldsymbol{t})\ {}^{W}\boldsymbol{P} = K(R\ \boldsymbol{t})\ {}^{W}\boldsymbol{P}$. We will consider four different settings of focal length, viewing angles and camera positions below.

Camera Settings :

1. [**No rigid body transformation**]. Focal length = 1. The optical axis of the camera is aligned with the z-axis, and pointing in the positive direction.
2. [**Translation**] Focal length = 1. $\boldsymbol{t} = (0, 0, 2)^T$. The optical axis of the camera is aligned with the z-axis.
3. [**Translation and rotation**]. Focal length = 1. R encodes first a 60 degree rotation around the z-axis and then 45 degrees around the x-axis. $\boldsymbol{t} = (0, 0, 2)^T$.
4. [**Translation and rotation, long distance**]. Focal length = 5. R encodes a 60 degrees rotation around the z-axis and then a 45 degrees rotation around the x-axis. $\boldsymbol{t} = (0, 0, 15)^T$.

For each of these settings, calculate and report :

(i) The extrinsic transformation matrix T,

(ii) The intrinsic camera matrix K under the perspective (pinhole) camera assumption. 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 other words: the only parameter in the intrinsic camera matrix under the perspective assumption is f.

(iii) Calculate the image of the four vertices and plot using the supplied plotsquare.m function (see e.g. output in figure 1).

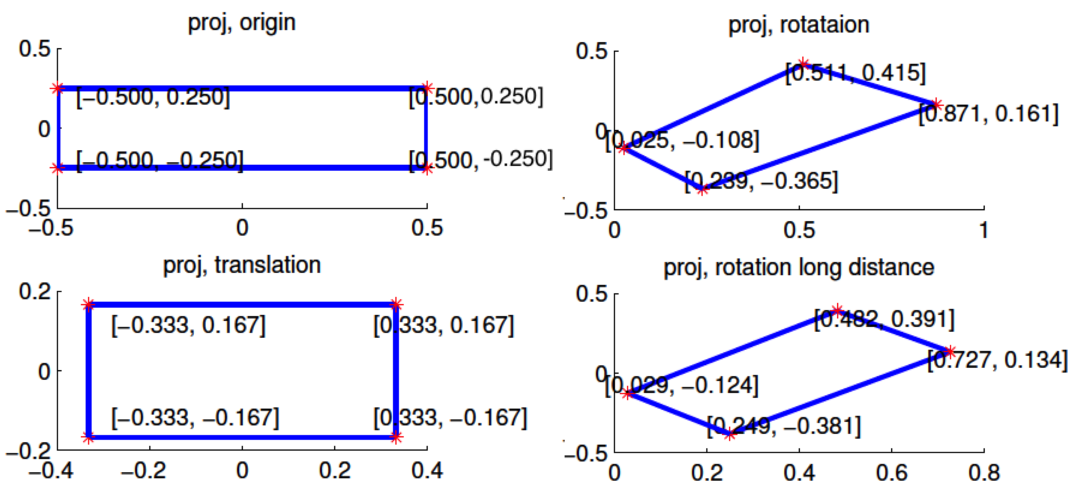

Figure 1: Example output for image formation problem. Note: the angles and offsets used to generate these plots are different from those in the problem statement, it's just to illustrate how to report your results.

In your report, include an image like Figure 1. Each correct image is worth 2 points (4 images i.e. 8 points). Presentation and discussion is worth 2 points (Explaining why you observe any distortions in the projection, if any, under this model).

3 Rendering [20 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 albedos are given in `facedata.mat`. Each row of the `lightsource` variable encode a light location). [Note: Please make good use out of `subplot.m` to display related image next to each other]

1. Plot the face in 2-D [2 points] : Plot both albedo maps using `imshow.m`, explain the results

2. Plot the face in 3-D [2 points] : 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 [8 points]: Calculate the surface normals and display them as a quiver plot using `quiver3.m`. Consider downsampling for better presentation. Recall that for a surface

$f(x,y)$, the surface normals are given by: $[-\frac{\delta f}{\delta x}, -\frac{\delta f}{\delta y}, 1]^T$. Also, recall, that each normal vector should be unitized.

4. Render images [8 points]: 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:

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

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