Computer Vision
G22.2271-001
Assignment 2.

October 21, 2012

1 Introduction

This assignment contains a number of problems in geometric computer vision. In addition to the course slides, you may find the following book helpful: Multiple View Geometry in Computer Vision by Richard Hartley and Andrew Zisserman. I have a copy which you can borrow (but not take out of the lab) and the Courant library also has a copy. This assignment also requires you to use SVD matrix factorization, although no in depth knowledge of its inner workings is needed. Hxowever, you can find more information on them at the Mathworld website, or in any linear algebra book.

2 Epipolar geometry

Figure 1 shows a pair of cameras each with focal length unity whose principal axes meet at a point. The y-axes of both cameras are parallel and point out of the page. Assume that the left camera (center C_L) lies at the origin of the world coordinate system.

1. Write down the camera matrices for this configuration and hence compute the fundamental matrix F.

2. Compute the epipolar line in the right image corresponding to the homogeneous point $\mathbf{x} = (1, 1, 1)^T$ in the left from $\mathbf{l} = \mathbf{F}\mathbf{x}$.

3. Using figure 1 determine where potential correspondences to the left image point $(x, y) = (0, 0)$ can lie in the right image.

4. The left camera is rotated about its center with the rotation axis parallel to the vertical direction until its principal axis is parallel with that of the right camera. How does the family of epipolar lines in the right image alter?

3 Estimating the camera parameters

Here the goal is the compute the 3x4 camera matrix P describing a pinhole camera given the coordinates of 10 world points and their corresponding image projections. Then you will decompose P into the intrinsic and extrinsic parameters. You should write a simple Matlab script that works through the stages below, printing out the important terms.

Download from the course webpage the two ASCII files, `world.txt` and `image.txt`. The first file contains the (X,Y,Z) values of 10 world points. The second file contains the (x,y) projections of those 10 points.

(a) Find the 3x4 matrix P that projects the world points $\mathbf{X}$ to the 10 image points $\mathbf{x}$. This should be done in the following steps:

- Since P is a homogeneous matrix, the world and image points (which are 3 and 2-D respectively), need to be converted into homogeneous points by concatenating a 1 to each of them (thus becoming 4 and 3-D respectively).

- We now note that $\mathbf{x} \times P\mathbf{X} = 0$, irrespective of the scale ambiguity. This allows us to setup a series of linear equations of the form:

$$\begin{bmatrix} 0^T & -w_i\mathbf{X}_i^T & y_i\mathbf{X}_i^T \\ w_i\mathbf{X}_i^T & 0^T & -x_i\mathbf{X}_i^T \\ -y_i\mathbf{X}_i^T & x_i\mathbf{X}_i^T & 0^T \end{bmatrix} \begin{pmatrix} P^1 \\ P^2 \\ P^3 \end{pmatrix} = 0 \qquad (1)$$

  for each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$, where $\mathbf{x}_i = (x_i, y_i, w_i)^T$, w_i being the homogeneous coordinate, and P^j is the j^{th} row of P. But since the 3rd row is a linear combination of the first two, we need only consider the first two rows for each correspondence i. Thus, you should form a 20 by 12 matrix A, each of the 10 correspondences contributing two rows.

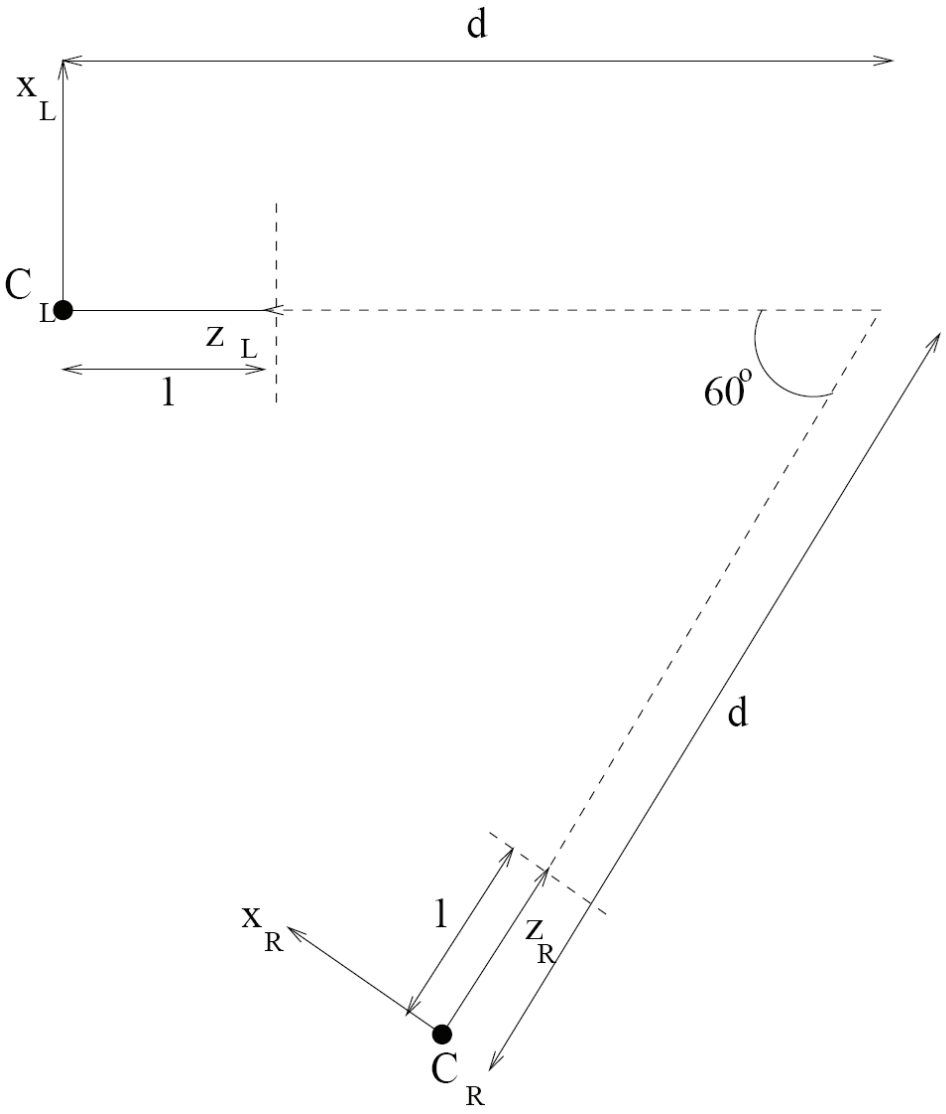

Figure 1: Figure for question 1

This yields $Ap = 0$, p being the vector containing the entries of matrix P.

- To solve for p, we need to impose an extra constraint to avoid the trivial solution $p = 0$. One simple one is to use $||p||_2 = 1$. This constraint is implicitly imposed when we compute the SVD of A. The value of p that minimizes Ap subject to $||p||_2 = 1$ is given by the eigenvector corresponding to the smallest singular value of A. To find this, compute the SVD of A, picking this eigenvector and reshaping it into a 3 by 4 matrix P.

- Verify your answer by re-projecting the world points $\mathbf{X}$ and checking that they are close to $\mathbf{x}$.

(b) Now we have P, we can compute the world coordinates of the projection center of the camera C. There are two different way to do this, which you should do in turn.

First, note that $PC = 0$, thus C lies in the null space of P, which can again be found with an SVD. Cmpute the SVD of P and pick the vector corresponding to this null-space. Finally, convert it back to homogeneous coordinates and to yield the (X,Y,Z) coordinates.

In the alternative route, we decompose P into it's constituent matrices. Recall from the lectures that $P = K[R|t]$. However, also, $t = -R\tilde{C}$, $\tilde{C}$ being the inhomogeneous form of C. Since K is upper triangular, use a QR decomposition to factor KR into the intrinsic parameters K and a rotation matrix R. Then solve for $\tilde{C}$. Check that your answer agrees with the solution from the first method.

4 Structure from Motion

In this section you will code up an affine structure from motion algorithm, as described in the slides of lecture 6. For more details, you can consult page 437 of the Hartley & Zisserman book.

First download the file `sfm_points.mat` from the course webpage. This contains a 2 by 600 by 10 matrix, holding the x, y coordinates of 600 world points projected onto the image plane of the camera in 10 different locations. The points correspond, that is `image_points(:,1,:)` is the projection of the same 3D world point in the 10 frames. The points have been drawn randomly

to lie on the surface of a transparent 3D cube, which does not move between frames (i.e. the object is static, only the camera moves). Try plotting out several frames and the cube shaped structure should be apparent.

To simplify matters, we will only attempt an affine reconstruction, thus the projection matrix of each camera i will have following form:

$$P^i = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} M^i & t^i \\ 0 & 1 \end{pmatrix} \tag{2}$$

where M^i is a 2 by 3 matrix and t^i is a 2 by 1 translation vector.

So given $m = 10$ views and $n = 600$ points, having image locations $\mathbf{x}^i_j$, where $j = 1, \ldots, n$, $i = 1, \ldots, m$, we want to determine the affine camera matrices M^i, t^i and 3D points $\mathbf{X}_j$ so that we minimize the reconstruction error:

$$\sum_{ij} ||\mathbf{x}^i_j - (M^i \mathbf{X}_j + t^i)||^2 \tag{3}$$

We do this in the following stages:

- Compute the translations t^i directly by computing the centroid of point in each image i.
- Center the points in each image by subtracting off the centroid, so that the points have zero mean
- Construct the $2m$ by n measurement matrix W from the centered data.
- Perform an SVD decomposition of W into UDV^T.
- The camera locations M^i can be obtained from the first three columns of U multiplied by $D(1:3, 1:3)$, the first three singular values.
- The 3D world point locations are the first three columns of V.
- You can verify your answer by plotting the 3D world points out using the `plot3` command. The `rotate3d` command will let you rotate the plot.

You should write a script to implement the steps above. The script should print out the M^i and t^i for the first camera and also the 3D coordinates of the first 10 world points.