Homework 3: Computation of the Camera Matrix Solution

I. Introduction

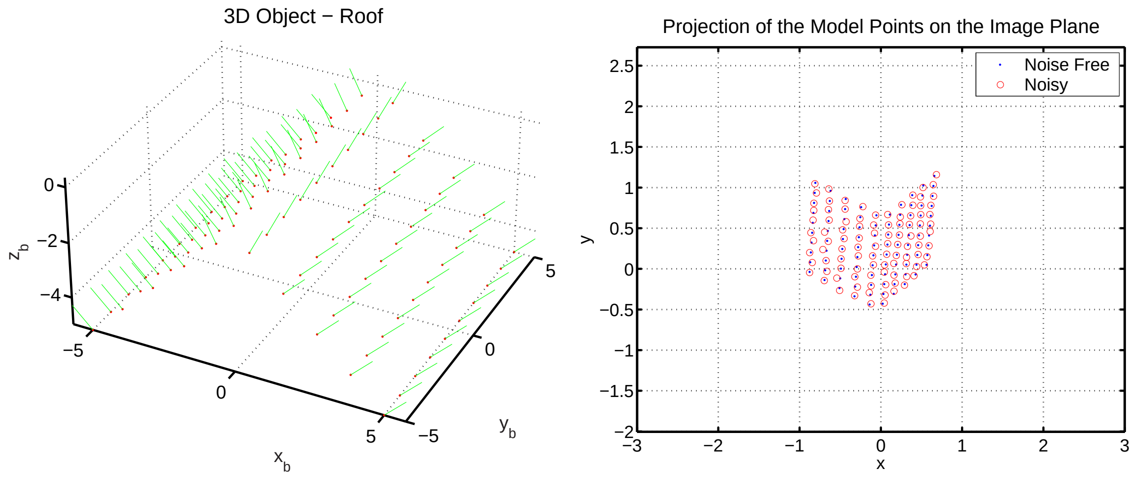

Fig. 1. The left images shows the point of the 3D model in the object coordinate system. The lines represent the normals (with respect to the underlaying surface). The right image shows the same points projected on the image plane. The dots represent the noise free projection, whereas the circles represent a noisy version of the projected points (obtained adding Gaussian noise with zero mean and $\sigma = 1.5 \cdot 10^{-2}$).

In this exercise you will be given a set of n point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$ where $\mathbf{x}_i$ denotes the homogeneous coordinates of a point on the camera image plane and $\mathbf{X}_i$ denotes the homogeneous coordinate of the same point in the 3D space. Your goal is to determine the camera matrix $P \in \mathbb{R}^{3\times 4}$ such that:

$$\mathbf{x}_i = P\mathbf{X}_i = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \mathbf{p}_3^T \end{bmatrix} \mathbf{X}_i$$

for $1 \leq i \leq n$. To carry out this task you will implement the Direct Linear Transformation (DLT) algorithm. Most of the material needed to solve this exercise can be found in the third Handout. From the web you can download the file `data.mat` which contains the matrices x and X. On the i^{th} column of these matrices you can read the homogeneous coordinates of the corresponding points $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$.

II. The Math

It is easy to see that $\mathbf{x}_i = P\mathbf{X}_i$ implies $\mathbf{x}_i \times P\mathbf{X}_i = 0$ (the outer product between two parallel vectors is always 0: why?). This condition will be used to derive a matrix whose null space will allow you to reconstruct P.

Question 1: Show the steps to derive the three linear equations (with respect to the components of the matrix P) that are obtained from the relation $\mathbf{x}_i \times P\mathbf{X}_i = 0$. How many of them are linearly independent? Why?

Suggestion: the outer product between two vectors $\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^T$ and $\begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix}^T$ can be obtained calculating the formal determinant:

$$\begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{vmatrix}$$

Note that in our case the scalars y_i are given by the inner product of the j^{th} row of P and the vector $\mathbf{X}_i$, i.e. $\mathbf{p}_j^T\mathbf{X}_i$.

Answer 1: The definition of the formal determinant allow us to write:

$$\begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ x & y & 1 \\ \mathbf{p}_1^T\mathbf{X} & \mathbf{p}_2^T\mathbf{X} & \mathbf{p}_3^T\mathbf{X} \end{vmatrix} = 0 \Rightarrow \begin{cases} y\mathbf{p}_3^T\mathbf{X} - \mathbf{p}_2^T\mathbf{X} = 0 \\ \mathbf{p}_1^T\mathbf{X} - x\mathbf{p}_3^T\mathbf{X} = 0 \\ x\mathbf{p}_2^T\mathbf{X} - y\mathbf{p}_1^T\mathbf{X} = 0 \end{cases}$$

There are only two linearly independent equations (the third one can be generated by summing the first to the second): we will drop the third one. If $\mathbf{X} = \begin{bmatrix} X & Y & Z & 1 \end{bmatrix}^T$, then the first and second equation can be rewritten as:

$$XP_{2,1} + YP_{2,2} + ZP_{2,3} + P_{2,4} - yXP_{3,1} - yYP_{3,2} - yZP_{3,3} - yP_{3,4} = 0 \quad (1)$$
$$XP_{1,1} + YP_{1,2} + ZP_{1,3} + P_{1,4} - xXP_{3,1} - xYP_{3,2} - xZP_{3,3} - xP_{3,4} = 0 \quad (2)$$

or equivalently, in matrix form:

$$\begin{bmatrix} X & Y & Z & 1 & 0 & 0 & 0 & 0 & -xX & -xY & -xZ & -x \\ 0 & 0 & 0 & 0 & X & Y & Z & 1 & -yX & -yY & -yZ & -y \end{bmatrix} \mathbf{p} = 0$$

where $\mathbf{p}$ is defined in Question 2.

Question 2: Consider the vector $\mathbf{p} = \begin{bmatrix} \mathbf{p}_1^T & \mathbf{p}_2^T & \mathbf{p}_3^T \end{bmatrix}^T$ obtained by concatenating the transpose of the three rows of P (similarly to what you did in Homework 2 to estimate the homography H). Using the first two equations for each pair $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$ construct and write explicitly the matrix $A \in \mathbb{R}^{2n\times 12}$ such that:

$$A\mathbf{p} = 0$$

Answer 2: We just have to stack one on top of the other the equations (1) obtained for each of the point correspondences.

Question 3: Why the matrix P has only 11 degrees of freedom? What is the minimum number of points n_{min} that you need to estimate $\mathbf{p}$? Explain.

Answer 3: Since P relates two homogeneous vectors the scale does not matter, therefore one of the elements of the matrix can be fixed. Since $\mathbf{p} \in \mathbb{R}^{12}$ but it has only 11 degrees of freedom we need 5 point correspondences plus “half” correspondence, in the sense that we can use just the equation that relates the x or y coordinates. In general we can write $n_{min} = 6$ or, with a little abuse of notation, $n_{min} = 5\frac{1}{2}$.

III. The Implementation

In this section you will implement the DLT algorithm using `Matlab`. The final goal is to write a function `P = get_camera_matrix(x, X)` which computes the camera matrix P given the correspondences in the matrices `x` and `X`. The three main steps of the functions are:

1) Compute the matrix A from the point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$.
2) Solve the problem:

$$\hat{\mathbf{p}} = \min_{\|\mathbf{p}\|_2=1} \|A\mathbf{p}\| \tag{3}$$

3) Return the matrix $\hat{P}$ built form $\hat{\mathbf{p}}$.

The critical step of the algorithm is the second one. Read and understand how the `Matlab` function `svd` to compute the *singular value decomposition* of a matrix works (this will be useful also for the face recognition project). It can be shown that the vector $\mathbf{p}^*$ that solves (3) is given by the *last column* of the matrix `V`.

Question 4: Provide the `Matlab` code that implements the function `get_camera_matrix` and write the numerical expression of the camera matrix $\hat{P}$ obtained applying your routine. Check your result by plotting on the image plane the original points $\mathbf{x}_i$ and the projection of the points $\mathbf{X}_i$ according to $\hat{P}$, i.e. $\hat{\mathbf{x}}_i = \hat{P}\mathbf{X}_i$.
Suggestion Check the function `hold` and remember to convert from homogenous to cartesian coordinates.

Answer 4: The `Matlab` code necessary to compute the camera matrix is the following:

```
function A = get_A(x, X)

n = size(x, 2);

h = 1;
for k =1:n

    A(h, :) = [X(1, k) X(2, k) X(3, k) 1 ...
            0 0 0 0 ...
            -x(1, k)*X(1, k) -x(1, k)*X(2, k) ...
            -x(1, k)*X(3, k) -x(1, k)];
    A(h + 1, :) = [0 0 0 0 ...
            X(1, k) X(2, k) X(3, k) 1 ...
            -x(2, k)*X(1, k) -x(2, k)*X(2, k) ...
            -x(2, k)*X(3, k) -x(2, k)];

    h = h + 2;

```

```
end;

return
```

and:

```
function P = get_camera_matrix(x, X)

A = get_A(x, X);
[U S V] = svd(A);
P = V(:, size(V,2));

P = reshape(P, 4, 3)';

return
```

The original camera matrix is given by:

$$P = \begin{bmatrix} 0.14317810330863 & 0.00842890215678 & 0.02213099998823 & -0.00256938875583 \\ 0.00877652062655 & 0.10707672122171 & -0.09756200734510 & 0.05336716889261 \\ 0.00041241486525 & -0.00182986227068 & -0.00197121902156 & 0.97724937743711 \end{bmatrix}$$

whereas the camera matrix estimated using the previous code is:

$$\hat{P} = \begin{bmatrix} 0.14053566766049 & 0.00817447347994 & 0.02175591741225 & -0.00295581323561 \\ 0.00824411820581 & 0.10589782478380 & -0.09508514703888 & 0.05529468747919 \\ -0.00007706358473 & -0.00141827585856 & 0.00315672752683 & 0.97790994601782 \end{bmatrix}$$

Figure 2 shows the reprojected points.

Question 5: Let $\hat{\mathbf{x}}_i = \hat{P}\mathbf{X}_i$ be the projection of the point $\mathbf{X}_i$ on the image plane according to the matrix $\hat{P}$. Using `Matlab` compute the reprojection error:

$$E = \sum_{i=1}^{n} \|\mathbf{x}_i^{cartesian} - \hat{\mathbf{x}}_i^{cartesian}\|_2$$

where the superscript $cartesian$ indicates that the points are represented using cartesian coordinates.

Answer 5: $E = 0.04280366724876$

Question 6: Recalculate (and write) $\hat{P}$ using n_{min} point correspondences and compute the reprojection error. How does this value compare to the value you obtained in Question 5? Did you expect this? Why?
Suggestion Make sure you don't pick the points in some singular configuration, i.e. a set of collinear points.

Answer 6: By choosing the points with indices $1, 7, 12, 55, 63, 78$ we obtained:

$$\hat{P}_{min} = \begin{bmatrix} 0.12566042436172 & 0.00846780239953 & 0.01404267468514 & -0.01219122698349 \\ 0.00587863213522 & 0.09734221723896 & -0.08494521979584 & 0.04848021704513 \\ -0.00471289936000 & -0.00222770658181 & 0.03149803392517 & 0.9816804692578 \end{bmatrix}$$

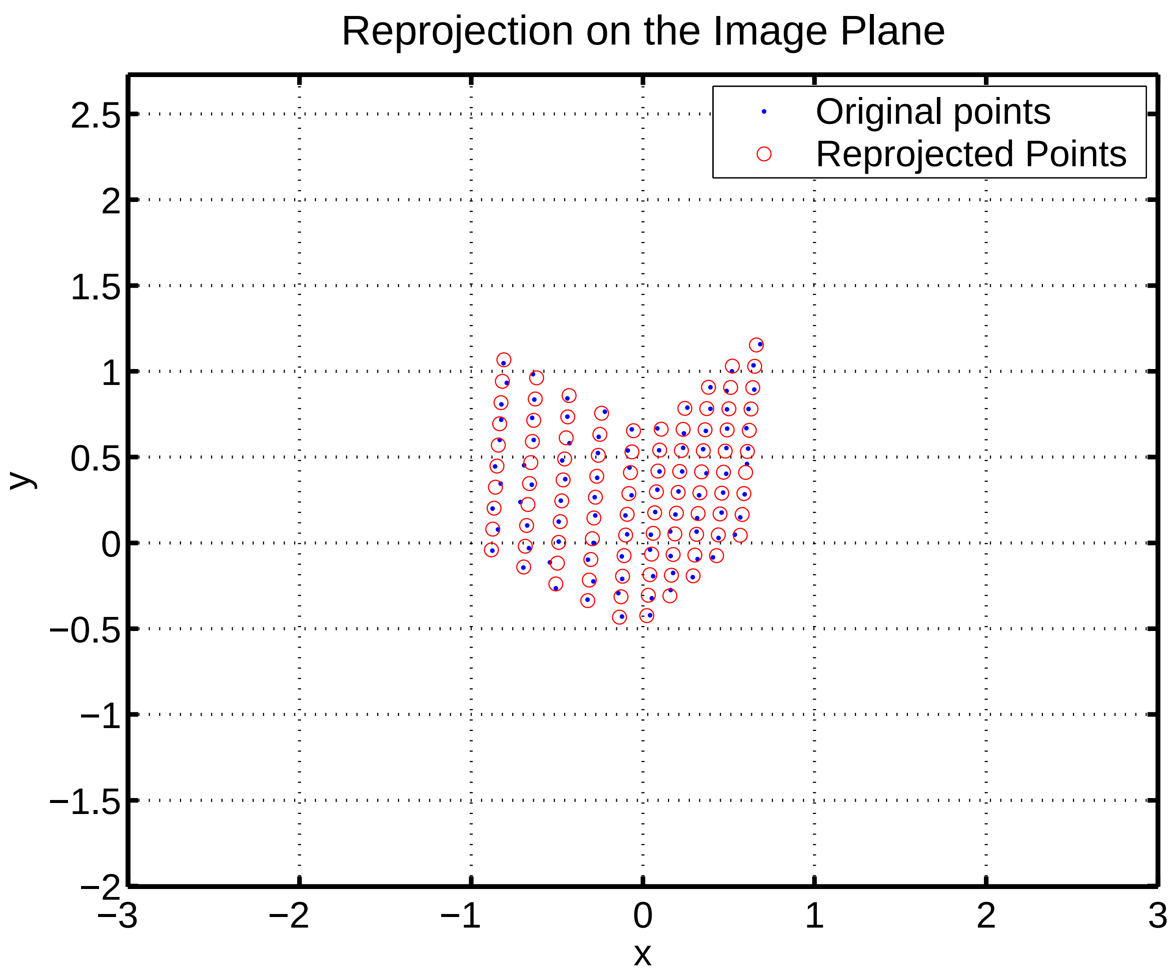

Fig. 2. The image shows the reprojected points $\hat{\mathbf{x}}_i$ on the camera image plane.

and consequently $E_{min} = 0.40401973632080$. Note that $E_{min} > E$ and this is expected since we are using less data in the least square estimation of the matrix P (and therefore the noise has a larger impact on the estimate).