Inferring 3D world points (synthetic data)

Overview. In this assignment, you will estimate the coordinates of a set of 3-D points in space from their images on multiple cameras (i.e., multiple views). Here, you will use an approach that assumes that both the intrinsic and extrinsic parameters of the cameras are known (i.e., Λ, Ω, and $\boldsymbol{\tau}$ are known). The image points will be collected from three different views, i.e., three cameras or a single camera that moved and captured images of the object from three different positions. In addition to determining the coordinates of the 3-D points, you will also produce a simple analysis of the robustness of the 3-D reconstruction method to noise. The analysis will be based on a measure of the pixel re-projection error. Figure 1 illustrates a 3-camera rig similar to the one you will use in this assignment.

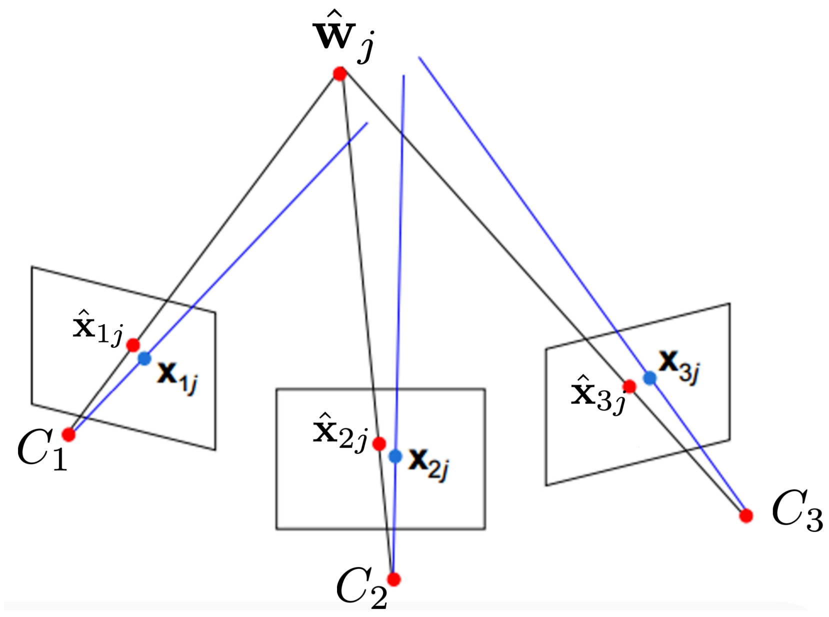

Figure 1: Re-projection error for a given 3-D point seen by three cameras. The 3-D point $\hat{\mathbf{w}}_j$ is the estimated location of the point that was captured by the cameras. The image point $\mathbf{x}_{ij}$, is the projection of the j-th 3-D point onto the image of the i-th camera. The process to calculate $\hat{\mathbf{w}}_j$ uses the pixel coordinates $\mathbf{x}_{ij}$ for $i = 1, \dots, 3$ as input. Given the estimated 3-D point $\hat{\mathbf{w}}_j$, we can then re-project it onto each camera using the camera models. The reprojected pixel coordinates is given by $\hat{\mathbf{x}}_{ij}$. The error between the measured pixel coordinates $\mathbf{x}_{ij}$ and the re-projected pixel coordinates $\hat{\mathbf{x}}_{ij}$ is called the *reproduction error*.

The camera rig. The cameras will be arranged in a similar pattern as the cameras in Figure 1 (see previous assignments for camera descriptions). You can re-use the camera specifications from the previous assignments. In this case, the central camera (i.e., Camera 2) is aligned with the global coordinate frame and is not rotated or translated. Camera 1 (to the left of Camera 2) is rotated by 35 degrees about the v-axis, translated along the u-axis by a vector $\boldsymbol{\tau}_1 = (200, 0, 0)^\mathsf{T}$. Camera 3 is positioned to the right of Camera 2, and is rotated by -35 degrees about the v-axis and translated along the u-axis by a vector $\boldsymbol{\tau}_3 = (-200, 0, 0)^\mathsf{T}$. All three cameras point towards the 3-D object, which is the same cube used in the previous assignments.

Measurements. Generate a set of synthetic image points as seen by each camera. For each 3-D point $\mathbf{w}_j$, use the camera model equations for each camera to generate the corresponding image points $\mathbf{x}_{ij}$ for $i = 1, \dots, 3$, i.e.:

$$\mathbf{x}_{ij} = \text{pinhole}\left[\mathbf{w}_j, \Lambda_i, \Omega_i, \boldsymbol{\tau}_i\right]. \tag{1}$$

Repeat the calculate in Equation 1 for all 3-D points from the cube. You can store all image points in two matrices as follows:

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1N} \\ x_{21} & x_{22} & \dots & x_{2N} \\ x_{31} & x_{32} & \dots & x_{3N} \end{bmatrix} \quad \text{and} \quad Y = \begin{bmatrix} y_{11} & y_{12} & \dots & y_{1N} \\ y_{21} & y_{22} & \dots & y_{2N} \\ y_{31} & y_{32} & \dots & y_{3N} \end{bmatrix} \tag{2}$$

Estimation method. Use the image points as input to the solution described in Section 14.6 (Prince) to reconstruct the 3-D position of the original points of the cube. To estimate the 3-D points, you will use only the image points, the cameras' intrinsic parameters , and the cameras' extrinsic parameters. The solution is given by solving Equation 14.42, which is a linear system of equations of the type $A\mathbf{x} = b$. Note that Equation 14.42 uses normalized image coordinates and not the original image coordinates. The coordinates of estimated 3-D points should be close to the original coordinates of the 3-D points of the cube.

Sensitivity analysis. Now that you have your estimated 3-D points and the estimation method at hand, plot the re-projection error for increasing levels of noise added to the input pixel coordinates. The re-projection error is given by:

$$E = \sum_{i=1}^{M} \sum_{j=1}^{N} \|\mathbf{x}_{ij} - \hat{\mathbf{x}}_{ij}\|^2, \tag{3}$$

where $\|\cdot\|$ is the Euclidean norm, and $\hat{\mathbf{x}}_{ij}$ is calculated using Equation 1 given the estimated 3-D points $\hat{\mathbf{w}}_j$ as input, i.e., run the estimated points $\hat{\mathbf{w}}_j$ through the same cameras to generate the corresponding $\hat{\mathbf{x}}_{ij}$.

In Matlab, you can add some Gaussian noise to the "image coordinates" as follows:

```
% Image points
X = [ 1 2 3 4; 0 3 4 0 ]

% Amount of noise (variance)
sigma = 0.5;

% Create a matrix of noise perturbations. Matrix has same size as the data matrix X.
N = rand(size(X,1),size(X,2)) * sigma

% Then, create the noisy image coordinates
X_noisy = X + N
```

Note that you want to have a different random noise value for each coordinate and not the same value added to all coordinates. To create the plot for the sensitivity analysis, simply add noise to the pixel coordinates incrementally and then calculate the re-projection error.

Python should have an equivalent way to add noise. I found the following Python example online:

```
import pandas as pd
# create a sample dataset with dimension (2,2)
# in your case you need to replace this with
# clean_signal = pd.read_csv("your_data.csv")
clean_signal = pd.DataFrame([[1,2],[3,4]], columns=list('AB'), dtype=float)
print(clean_signal)
"""
```

```
print output:
    A    B
0  1.0  2.0
1  3.0  4.0
"""
import numpy as np
mu, sigma = 0, 0.1
# creating a noise with the same dimension as the dataset (2,2)
noise = np.random.normal(mu, sigma, [2,2])
print(noise)

"""
print output:
array([[-0.11114313,  0.25927152],
       [ 0.06701506, -0.09364186]])
"""
signal = clean_signal + noise
print(signal)
"""
print output:
          A         B
0  0.888857  2.259272
1  3.067015  3.906358
"""
```