ENGN1610 & 2605 Image Understanding
Lab08: 3D Reconstruction

The goal of this lab is to

- Learn how to find valid relative pose from an essential matrix.
- Learn how to reconstruct a 3D scene by triangulation.

Problem 1. Scene Reconstruction Given two images, Figure 1, we are interested in reconstructing the scene in a 3D space. This can be achieved by first estimating an Essential matrix of image 2 with respect to image 1 in a RANSAC scheme, followed by a triangulation. The camera intrinsic matrix is provided in a `CalibrationMatrix.mat` file. The necessary steps to follow are:

1. **Resize the images:** The size of the provided image is pretty large which might take a lot of time if we do triangulation on dense point correspondences. Resize the images using `imresize` by a factor of 0.25. Note that the provided calibration matrix is already scaled down by a factor 0.25 from the original calibration matrix; thus only resizing the image is necessary.

2. **Estimate an Essential Matrix in a RANSAC loop:** This step is the same as Problem 3 of the previous lab, *i.e.*, *(i)* extract and match SIFT features, *(ii)* feed the matches into the function `Ransac4Essential`. In this lab, return both a valid essential matrix and indices of all inliers from `Ransac4Essential`. Set the number of RANSAC iterations at least 5000.

3. **Find Valid Relative Pose from an Essential Matrix:** Follow two stages below:

   (a) Decompose the essential matrix `finalE` returned from your `Ransac4Essential` function into rotation matrices R and translation vectors T. There are four sets of possible R and T. You have finished this part in Problem 3 of the previous lab.

   (b) In the lecture, we say that only one out of four sets of R and T is true. To find the true set, loop over all inlier matches (matches indexed by the indices returned by the `Ransac4Essential` function), and compute their respective depths ρ_1 and ρ_2 in image 1 and 2. Return R and T for which the maximal number of inlier matches have positive depths ρ_1 and ρ_2. Please refer to the Appendix for more information on how to find the depths.

4. **Densify Correspondences:** The inlier matches returned by the RANSAC algorithm is usually too sparse for a visually pleasing 3D reconstruction. Thus, densifying the correspondences is necessary. Use the provided matlab function file `Densification.m` and refer to the instructions in that function file to see what the inputs and outputs are.

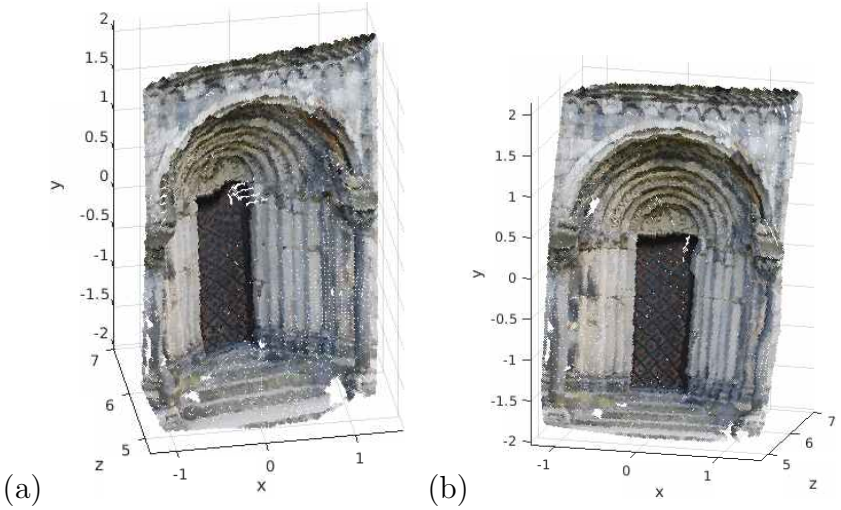
5. **Triangulation:** Using the true R and T obtained from the step 3, loop over all inliers dense matches returned by the `Densification` function and compute their depths ρ_1 and ρ_2. The triangulated 3D points are thus $\Gamma_1 = \rho_1\gamma_1$ under the first camera coordinate, and $\Gamma_2 = \rho_2\gamma_2$ under the second camera coordinate. Note that Γ_1 and Γ_2 are related by

the relative pose R and T, *i.e.*, $\Gamma_2 = R\Gamma_1 + T$.

In the lecture, we say that due to noise, the projection rays from the same 3D point to the two images do not always intersect. A simple but effective way to tackle this is mid-point method: taking the average of Γ_1 and Γ_2 under the same coordinate as the final reconstructed 3D points. Store all 3D points and its RGB values. Use `scatter3` to show your 3D reconstruction results. An example is given in Figure 2. You may use `axis equal` after `scatter3` to properly visualize the reconstruction result. Specify in your report which camera coordinate your 3D reconstruction is under.

(a) (b)

Figure 1: 3D scene reconstruction from different views.



(a) (b)

Figure 2: 3D scene reconstruction from different views.

Problem 2. Answer the Question Take a look at the `Densification.m` function file. How does the function work to make dense correspondences? Also, briefly explain the possible causes of the gaps of regions obtained in the 3D reconstruction.

Problem 3. Triangulation by Non-linear Optimization (This part is mandatory for ENGN2605 students but optional for ENGN1610 students who will get 5 extra points if it is answered correctly.)

In the lecture, we say that the mid-point triangulation is a method that finds an intersection of the projection rays to the two images. This minimizes reprojection errors in a model space (or representation space), *i.e.* the 3D world. However, a better way to find an intersected 3D point is to minimize reprojection errors in an observation space, *i.e.* the 2D images: let γ and $\bar{\gamma}$ be a pair of inlier correspondence we obtained from the two images, and $\hat{\gamma}$ and $\hat{\bar{\gamma}}$ be the ground truth correspondence which are unknown. (Notations are borrowed from the lecture notes.) The goal is that, given the relative pose R and T, find the perturbations of γ and $\bar{\gamma}$ such that the projection rays intersect, while minimizing the extent of the perturbations at the same time, *i.e.*,

$$\min_{\hat{\gamma},\hat{\bar{\gamma}}} \left[|\hat{\gamma} - \gamma|^2 + |\hat{\bar{\gamma}} - \bar{\gamma}|^2 \right], \quad \text{subject to} \quad \hat{\bar{\gamma}}^T E \hat{\gamma} = 0. \tag{1}$$

This is a constrained optimization problem which can be lifted to an non-constrained optimization problem by using a Lagrange multiplier λ:

$$\min_{\hat{\gamma},\hat{\bar{\gamma}},\lambda} \left[|\hat{\gamma} - \gamma|^2 + |\hat{\bar{\gamma}} - \bar{\gamma}|^2 + \lambda \hat{\bar{\gamma}}^T E \hat{\gamma} \right]. \tag{2}$$

Equation 2 is a non-linear function with five unknowns: two from $\hat{\gamma}$, two from $\hat{\bar{\gamma}}$, and one from the Lagrange multiplier λ. You are required to implement the triangulation by numerically minimizing Equation 2 to find the optimal $\hat{\gamma}$ and $\hat{\bar{\gamma}}$. Necessary steps to follow are:

1. **Load the Given Data:** Load three `.mat` files: *(i)* a set of inlier but very noisy correspondences given in the `Two_View_Correspondences.mat` file, *(ii)* the ground truth relative pose R and T in the `RelativePose.mat` file, and *(iii)* an intrinsic matrix provided in the `CalibrationMatrix.mat` file.

2. **Compute an Initial Guess:** For each pair of noisy correspondences, do mid-point triangulation to get the 3D reconstructed point. Project that 3D point to the two images. The projected points are served as an initial guess for the non-linear optimization. For the Lagrange multiplier, you can pick any arbitrary value as its initial guess.

3. **Minimize Equation 2:** For *each* correspondence,

   (a) Construct an energy function

   ```
   function Energy = evaluateEnergyFunction(x, gamma, gamma_bar, E)
   ```

   which takes inputs `x` as an array containing five unknowns, or variables ($\hat{\gamma}$, $\hat{\bar{\gamma}}$, and λ), a pair of observed correspondence (γ and $\bar{\gamma}$), and an essential matrix E, returns the energy value, *i.e.*, the evaluation of Equation 2.

   (b) Do non-linear optimization using Levenberg-Marquardt algorithm. You are allowed to use matlab's built-in function. A sample piece of code is given to you in order to use the matlab non-linear optimizaer:

```
Energy = @(x)(evaluateEnergyFunction(x, gamma, gamma_bar, E));
options = optimoptions('lsqnonlin', 'Display', 'iter');
options.Algorithm = 'levenberg-marquardt';
optimal_points = lsqnonlin(Energy, x0, [], [], options);
```

where `x0` is the initial guess you obtained from step 2, which is also an array containing five variables. `lsqnonlin` is a matlab built-in least square non-linear optimizer which gives you the optimized points.

4. The optimized point correspondences can then be used to find their depths and triangulate to find a reconstructed 3D point which is very close to the intersection of the projection rays. This can be seen from the reprojection error, *i.e.* $|\hat{\gamma} - \gamma|^2 + |\hat{\bar{\gamma}} - \bar{\gamma}|^2$. Compare the mean and median reprojection errors using the mid-point triangulation method and the non-linear optimization method. In what order of magnitude is the reprojection error reduced after the optimization?

Appendix: Depths Computation

Let γ_1 and γ_2 be a pair of correspondences in meters, and their corresponding 3D points which project to γ_1 and γ_2 be Γ_1 and Γ_2 under the first and second camera coordinate, respectively. The relative geometry between two cameras can be captured by a rotation matrix R and a translation vector T such that a 3D point whose representation in two cameras is related as

$$\Gamma_2 = R\Gamma_1 + T, \tag{3}$$

or

$$\rho_2\gamma_2 = R\rho_1\gamma_1 + T, \tag{4}$$

where ρ_1 and ρ_2 are the depths of γ_1 and γ_2. Rewriting Equation 4 in a form of a linear system,

$$\begin{bmatrix} -R\gamma_1 & \gamma_2 \end{bmatrix} \begin{bmatrix} \rho_1 \\ \rho_2 \end{bmatrix} = T, \tag{5}$$

we can solve ρ_1 and ρ_2 given R and T. Note that $\begin{bmatrix} -R\gamma_1 & \gamma_2 \end{bmatrix}$ is a 3×2 non-square matrix, to solve Equation 5 a pseudo-inverse method or a SVD method can be used. Be aware that the computed depths are up to a scale because of metric ambiguity in the geometry of relative pose estimation.