Assignment 3

November 13, 2021

Andreas Kaufmann
nethz Username: ankaufmann
Student ID: 15-545-585

2.2 Calibration

Brief explanation of the DLT algorithm: The goal of DLT is to estimate all parameters of an affine camera, using at least 6 3D points to 2D image location mappings. We differentiate between two types of parameters. Extrinsic and intrinsic parameters. The extrinsics describe where the camera is located and where it is looking at. The intrinsics are camera internal parameters. Intrinsic parameters are the camera constant and the principal point. DLT can estimate these parameters, specifically it estimates R (Rotation), X_0 (Position) and K (Camera Calibration) It can not estimate parameters such as lens distortions (non linear parameters). DLT uses the following equation:

$$x = PX \tag{1}$$

Where $x = [x_1, x_2, 1]$ are the 2D-Points on the image, $X = [X_1, X_2, X_3, 1]$ are the 3D Points and P is a 3×4 matrix, containing the parameters we want to estimate. We can rearange $x = PX$ such that we can build coefficient vectors with the known points as follows:

$$\begin{array}{cccccccccccccc}
p_{11} & p_{12} & p_{13} & p_{14} & p_{21} & p_{22} & p_{23} & p_{24} & p_{31} & p_{32} & p_{33} & p_{34} & & \\
0 & 0 & 0 & 0 & -X_1 & -X_2 & -X_3 & -1 & x_2 * X_1 & x_2 * X_2 & x_2 * X_3 & x_2 * 1 & = & 0 \\
X_1 & X_2 & X_3 & 1 & 0 & 0 & 0 & 0 & -x_1 * X_1 & -x_1 * X_2 & -x_1 * X_3 & -x_1 * 1 & = & 0
\end{array}$$

We get 2 coefficient vectors per point (one for x_1 and one for y_1 dimension). As P consists of 12 Parameters, we need 6 points to calculate P, which is the null space of the system of linear equations, defined by this 6 points. We solve the system by applying SVD. When P is found, we can decompose it to get R, X_0 and K

Picture of result:

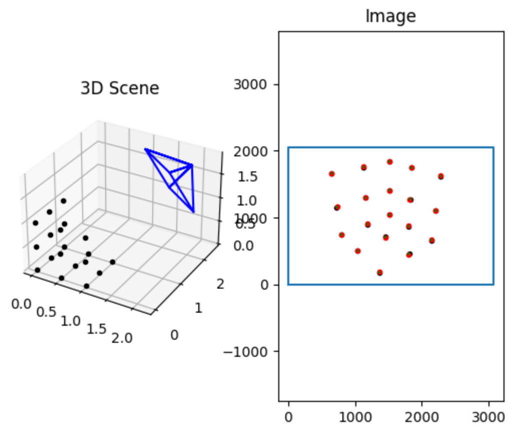

Figure 1: DLR Result

Which part of the full model is not calibrated with our approach? Non linear parameters can not be estimated with the DLT algorithm. This could be parameters such as lens distortion.

a) **Data Normalization: Explain problems, if we skip the normalization:** We want to avoid getting huge and very small numbers. This could easly happen by multiplying small by small numbers or big by big numbers. Big numbers would dominate the error we have in the model. We therefore normalize by centering and scaling the data.

b) DLT: **How many independent constraints can you derive from a single 2D-3D correspondence?** I can derive 2 constrains from one 2D-3D correspondence. An informal explanation therefore is that equation 1 gets two results, x_1 and x_2, out of which we can build the two constraints shown in the table above. As each point x provides two constraints we need at least 6 points as we have 11 degrees of freedom.

c) Optimizing reprojection errors:

- **How does the reported reprojection error change during optimization?** The optimization function uses a minimization function from scipy, which works with the BFGS Method. The reported reprojection error decreases during optimization.
- **Difference between algebraic and geometric error** The algebraic error arises from the residual vector $\epsilon = x \times PX$. The components of this vector consist of the individual correspondences of the givne 3D-2D correspondences. I.e. each correspondence $x_i \longleftrightarrow x'_i$ contributes a partial error vector ϵ_i towards the full error vector ϵ. The vector ϵ_i is called the algebraic error vector associated with the point correspondence $x_i \longleftrightarrow x'_i$ and the homography P. On the other hand, the geometric error is the difference between the measured and the estimated image coordinates.
- **Problem with the error measure** $e = x \times PX$ The error gets bigger, the more point correspondences we have. I.e. there is noise in the point measurements, in our case the system is over determined as we have more then 6 points given. Thus there is no exact solution. This is the reason why we have to minimize the error.

d) Denormalizing the projection matrix: -

e) Decomposing the projection matrix

- **Computed K, R and t**

$$K = \begin{bmatrix} 2.713 + e03 & 3.313e + 00 & 1.481e + 03 \\ 0.0e + 00 & 2.710e + 03 & .654e + 02 \\ 0.0e + 00 & 0.0e + 00 & 1.0e + 00 \end{bmatrix} \tag{2}$$

$$R = \begin{bmatrix} -0.774 & 0.633 & -0.007 \\ 0.309 & 0.369 & -0.877 \\ -0.552 & -0.681 & -0.481 \end{bmatrix} \tag{3}$$

$$t = \begin{bmatrix} 0.047 \\ 0.054 \\ 3.441 \end{bmatrix} \tag{4}$$

- **Discuss the reported re-projection errors before and after nonlinear optimization**
  - Error before optimization: 0.0006316426059796245
  - Error after optimization: 0.0006253538899291193
  - Difference: 6.288716050505258e-06

  Projection errors are in the area of 10^{-4} and get slightly decreased by applying the optimization step.
- **Do your estimated values seem reasonable?** Yes, they are reasonable as the error is small.

2.3 Structure from Motion

Explanation of the SfM approach: In structure from motion we use multiple images from the same subject with given keypoints and keypoint correspondences from one image to another to do a 3D reconstruction of the scene. Using epipolar geometry, we can Triangulate a 3D-Point from two corresponding keypoints. This is done for every image pair, to get a 3D point for every matching keypoint. How the process works is briefly explained as follows:

- First images with keypoints have to be loaded. The keypoints are already predefined.
- Then, for every image pair a list of matching keypoints is loaded. This gives the keypoint correspondences.
- We start with an image pair (here with image 3 and 4). As we are in the calibrated setting (we first transform every point $\hat{x} = K^{-1}x$), we can directly compute the essential matrix with the equation $\hat{x_1} E \hat{x_2} = 0$. x_1 and x_2 are corresponding keypoints from the two images. This equation can be built from properties, arising from epipolar geometry. The fundamental matrix F can be written as $F = [e']_x H_\pi$, where H_π is the transfer mapping from one image to another via any plane π. $[e']_x$ is the skew-symmetric matrix of the epipole and has rank 2. Thus the Fundamental matrix is of rank 2. The computation of E uses similar tecniques as DLT.
- We can now decompose the essential matrix into rotation R and translation t. There is four possible such decompositions. To find out which of the four is correct, we have to set one camera as the center and by triangulating all corresponding points, we have to check with which decomposition, most triangulated 3D points are in front of both cameras. The triangulation of the points is already implemented and not explained here in detail.
- After deriving the correct R and t, we iterate over the other pictures, estimate their image pose and execute the triangulation over every already processed image to get new 3D points. Every newly derived 3D point is added to the scene.
- Finally the derived 3D points of all image pairs are plotted such to get a reconstruction of the scene. Please find the plot of the scene reconstruction below in figure 2

Picture of results:

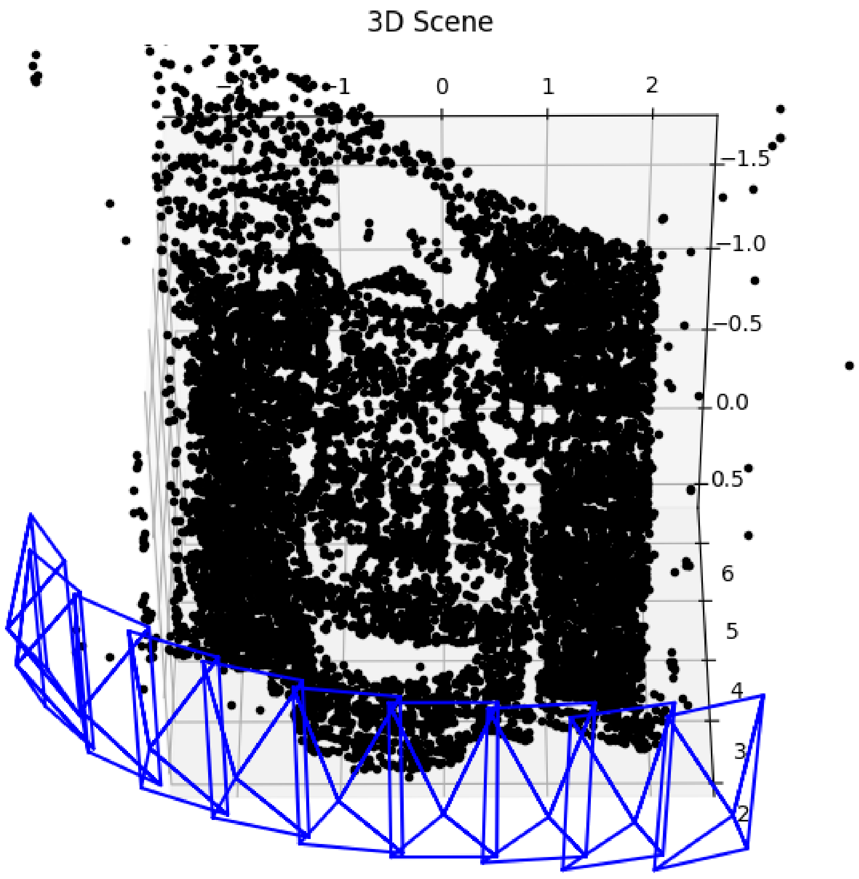

Figure 2: Result Structure From Motion 3D Reconstrucion of scene