Computer Vision, 2nd assignment

Mathis Lamarre

Normalization

In order to normalize, the centroid is computed. c_x and c_y are the means of the x- and y-coordinates of the 2D points:

$$c_x = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad c_y = \frac{1}{n}\sum_{i=1}^{n} y_i$$

c_X, c_Y and c_Z are the means of the x-, y- and z-coordinates of the three dimensional points:

$$c_X = \frac{1}{n}\sum_{i=1}^{n} X_i \qquad c_Y = \frac{1}{n}\sum_{i=1}^{n} Y_i \qquad c_Z = \frac{1}{n}\sum_{i=1}^{n} Z_i$$

Then, the scale is computed. s_{2D} and s_{3D} are the scaling factors for the 2D and 3D points:

$$s_{2D} = \sqrt{\frac{\frac{1}{n}\sum_{i=1}^{n}[(x_i - c_x)^2 + (y_i - c_y)^2]}{2}}$$

$$s_{3D} = \sqrt{\frac{\frac{1}{n}\sum_{i=1}^{n}[(X_i - c_X)^2 + (Y_i - c_Y)^2 + (Z_i - c_Z)^2]}{3}}$$

The transformation matrices T and U, used for the 2D and 3D points respectively, are:

$$T = \begin{bmatrix} s_{2D} & 0 & c_x \\ 0 & s_{2D} & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \qquad U = \begin{bmatrix} s_{3D} & 0 & 0 & c_X \\ 0 & s_{3D} & 0 & c_Y \\ 0 & 0 & s_{3D} & c_Z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}$$

The set of points can then be normalized by multiplying them by the matrices as such: $\hat{x}_i = Tx_i$ and $\hat{X}_i = UX_i$.

Direct Linear Transform

To implement the DLT algorithm, the A matrix si computed as such:

$$A = \begin{bmatrix} w_i\hat{X}_i^T & 0^T & -x_i\hat{X}_i^T \\ 0^T & -w_i\hat{X}_i^T & y_i\hat{X}_i^T \end{bmatrix}$$

The P matrix can be obtained from the right null-vector of the single value decomposition of A. It is then denormalized using the T and U matrices from last section. K, R and t can be obtained from the QR decomposition. The 2D projection points are computed by multiplying the 3D coordinates by the P matrix while making sure the last coordinate is 1.

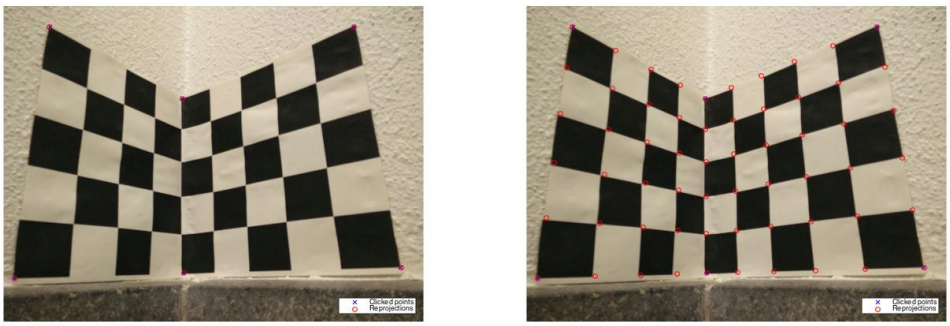

Fig. 1: Projection of points with the DLT algorithm. Only the 6 corners were used for the calibration. The projection of only these corners and all the points are shown.

When using only 6 points for the calibration (Fig.1), it doesn't make a notable difference to use the normalized or unnormalized points. However, when using all the points to calibrate, if the entries are not normalized, they have very different amplitudes. The projection is therefore very poor without normalization, see Fig. 2.

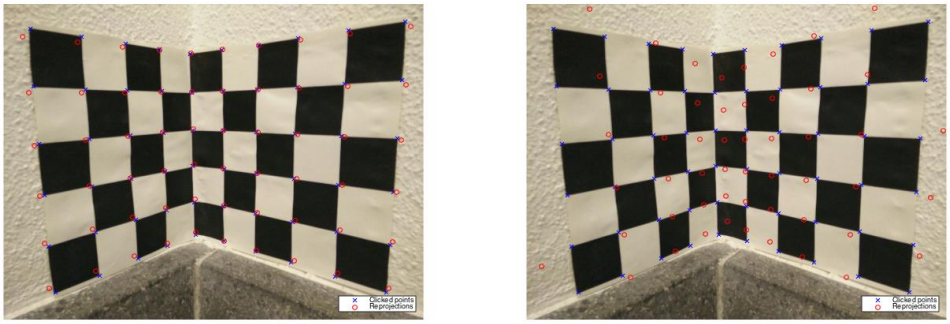

Fig. 2: Projection of points with the DLT algorithm. All the points were used for the calibration. The left side shows result with normalization and the right side without normalization.

The resulting calibration parameters are:

$$K = 10^3 \cdot \begin{bmatrix} 1.3053 & 0.0181 & 0.8267 \\ 0 & 1.2824 & 0.6004 \\ 0 & 0 & 0.0010 \end{bmatrix}$$

$$R = \begin{bmatrix} -0.5987 & 0.8005 & -0.0269 \\ -0.2292 & -0.2034 & -0.9519 \\ -0.7675 & -0.5637 & 0.3053 \end{bmatrix}$$

$$t = \begin{bmatrix} -0.0395 \\ 0.1150 \\ 0.3538 \end{bmatrix}$$

Gold Standard Algorithm

In order to run the Gold Standard Algorithm, the DLT algorithm is ran. Then, the P matrix is reassemble and unnormalized to get the parameters from the K matrix. The normalized version of the P matrix is used to get the projected points $\hat{x}_i = PX_i$ and the distance between the $\hat{x}_i$ and the clicked points x_i is minimized. To do so, this cost function was used:

$$\sum_{i}^{n} d(x_i, \hat{x}_i)^2 + ws^2 + w(\alpha_x - \alpha_y)^2$$

with α_x and α_y taken from the K matrix.

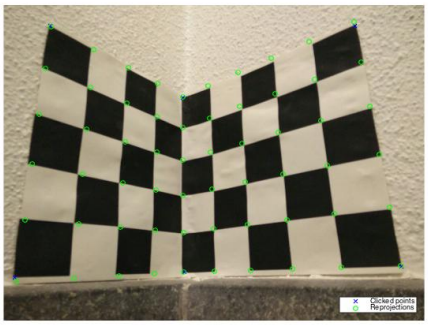

Fig. 3: Projection of points with the Gold Standard algorithm. The 6 corners were used for calibration.

The resulting calibration parameters are:

$$K_{gold} = 10^3 \cdot \begin{bmatrix} 1.2706 & 0.0000 & 0.8151 \\ 0 & 1.2706 & 0.5980 \\ 0 & 0 & 0.0010 \end{bmatrix}$$

$$R_{gold} = \begin{bmatrix} -0.6036 & 0.7964 & -0.0383 \\ -0.2167 & -0.2101 & -0.9534 \\ -0.7673 & -0.5672 & 0.2994 \end{bmatrix}$$

$$t_{gold} = \begin{bmatrix} -0.0357 \\ 0.1153 \\ 0.3495 \end{bmatrix}$$

As expected, if we compare to the results of the DLT algorithm, we notice that in this case we have $\alpha_x = \alpha_y$ and that the skew factor $s = 0$.

Bouget Calibration

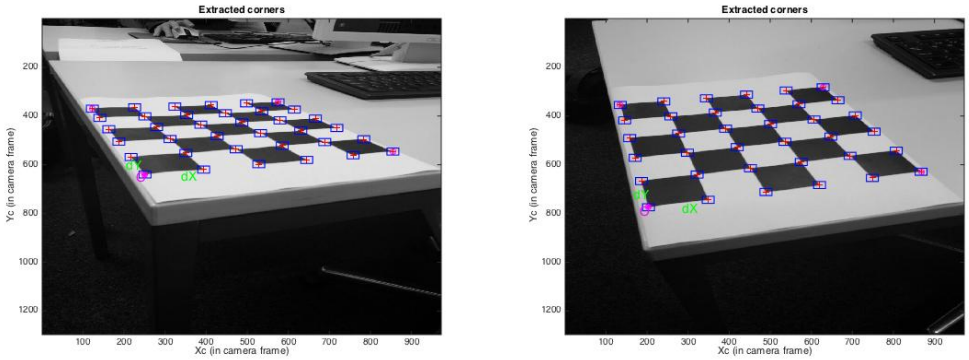

Fig. 4: Extracted corners with the Bouget Toolbox.

Unfortunately the Bouget Toolbox would not let me use more than 2 images for the calibration. With the extracted corners shown in Fig. 4, these are the results I got:

$$\alpha_x = 875.96894 \pm 473.30951, \alpha_y = 1055.56499 \pm 299.60910$$

They are not very precise but the intervals contain the values previously found.