CS791E Computer Vision – Assignment 03 – Calibration

Jorge H. Usabiaga
e-mail: `usabiaga@cs.unr.edu`

March 28, 2003

1 `calibration_data.tar.gz`

The file `calibration_data.tar.gz` contains the following set of files:

- `capXX.bmp`

  These are the calibration images. 15 pictures of the same chessboard taken with the same camera at different angles.

- `pixelXX.txt`

  These text files contain the pixel coordinates of each of the 96 corners of the chessboard that are to be used to calibrate the camera. The file `pixel12.txt` corresponds to the image `cap12.bmp`.

- `worldXX.txt`

  These text files contain the world coordinates of each of the 96 corners of the chessboard that are to be used to calibrate the camera. The file `world12.txt` corresponds to the image `cap12.bmp`.

1.1 `pixelXX.txt`

As there are 96 points per image, each of the `pixelXX.txt` files contains 96 2D pixel coordinates. So we have 192 *double* numbers in the text file.

The first number is the x coordinate of the first point, the second number is the y coordinate of the first point, the third number is the x coordinate of the second number, and so on.

1.2 `worldXX.txt`

There is no restriction on how we choose the *World Reference Frame*, so, in order to make things easier, we choose it with its origin at one of the corners of the chessboard, with the X and Y axis parallel to the squares of the board and the Z axis perpendicular to the chessboard plane. Therefore, all the points will have coordinate $Z = 0$ as it happens with the pixel coordinates. That is why we, again, have 192 points in each of the `worldXX.txt` files. More importantly, the points will be ordered as in the correspondent `pixelXX.txt` (for example, the pixel coordinates of the projection of the seventh point in `world12.txt` correspond to the coordinates of the seventh point in `pixel12.txt`.

Although the Z coordinate is 0 for all the points of the chessboard in the *World Reference Frame* it is necessary to store the points as CvPoint2D32f (floats) or CvPoint3D64d (doubles) since the function `cvCalibrateCamera` expects the coordinates of the point in the world system as 3D points. This is resolved by reading in the X and Y coordinates and then adding zeros for all the Z coordinates.

2 OpenCV

I have taken a look at the two functions that do the camera calibration in OpenCV. The only difference between them is that one works with *floats* and the other one with *doubles*. The prototype of one of them looks as follows:

```
void cvCalibrateCamera( int numImages, int* numPoints, CvSize imageSize,
                        CvPoint2D32f* imagePoints32f, CvPoint3D32f* objectPoints32f,
                        CvVect32f distortion32f, CvMatr32f cameraMatrix32f,
                        CvVect32f transVects32f, CvMatr32f rotMatrs32f,
                        int useIntrinsicGuess );


numImages
   Number of the images.

numPoints
   Array of the number of points in each image.
   (In our case it would be: int numPoints[numImage] = {96}, since all the images
   have 96 points each)

imageSize
   Size of the image.

imagePoints32f
   Pointer to the images.
   In our case it would be an array of \textit{numImages x numPoints} elements
   (an array of CvPoint2D32f [CvPoint2D64d] with all the points in all the files
   pixelXX.txt ordered by the image number, so the pixel coordinates of the points
   in the first image would be the first ones).

objectPoints32f
   Pointer to the pattern.
   In our case it would be an array of \textit{numImages x numPoints} elements
```

(an array of CvPoint3D32f [CvPoint3D64d] with all the points in all the files worldXX.txt ordered by the image number, so the world coordinates of the points in the first image would be the first ones)

`distortion32f`
: Array of four distortion coefficients found.

`cameraMatrix32f`
: Camera matrix found.

`transVects32f`
: Array of translate vectors for each pattern position in the image.

`rotMatrs32f`
: Array of the rotation matrix for each pattern position in the image.

`useIntrinsicGuess`
: Intrinsic guess. If equal to 1, intrinsic guess is needed.

3 World2Pixel.jpg

The image `World2Pixel.jpg` is an example of one of the images, with the corners drawn in red. The crosses over the corners are the projected points calculated from the 3D points after doing calibration.