Camera Calibration of Rectangular Textures

Jacky Baltes
Centre for Imaging Technology and Robotics
University of Auckland
Auckland, New Zealand
Email: j.baltes@auckland.ac.nz
http://www.citr.auckland.ac.nz/~jacky

October 11, 2000

Abstract

This paper describes a practical method for the camera calibration given a single image of a regular texture. This paper uses the calibration of images of skyscrapers as an example. The paper introduces two algorithms for the assignment of real world coordinates to feature points. The first algorithm selects five closely connected feature points and determines the orientation of the rectabgular pattern. The second algorithm iteratively sorts the feature points and assigns real world coordinates to them. Lastly, the Tsai camera calibration algorithm is used to compute the camera parameters.

1 Introduction

This paper describes an application of our camera calibration method, which was initially developed for RoboCup.

RoboCup is an international competition of fully autonomous robots playing soccer [3]. The first competiton was organized in 1997, and it has rapidly increased in popularity.

Apart from the obvious challenges in robotics, control theory, path planning, artificial intelligence, and machine learning, RoboCup also presents an interesting domain for real-time computer vison. In the small league, robots are identified using a global vision system. To achieve adequate control, a

vision system must track ten robots, a ball and compute their position, orientation, velocity with a cycle time of less than 20ms. More details about the All Botz videoserver can be found in [1].

This paper shows how the camera calibration method which was originally developed for the RoboCup domain can be used to compute the calibrate images of any regular textures or co-planar feature points using a single image.

Section 2 is an introduction to camera calibration in general and the challenges of calibration using regular textures. The extraction of calibration points is shown in section 3. The Tsai camera calibration used as back end to compute the final calibration parameters is described in 4. The paper concludes with section 5.

2 Camera Calibration

Accurate camera calibration is an essential ingredient in any computer vision system. Therefore, it has been a very active and productive research area. A number of different camera calibration methods have been developed [4]. Most of these methods are based on the pin-hole camera model.

The input to camera calibration methods is a set of image features and their associated real-world attributes. For example, the well known Tsai calibration uses a set of image points with known real world coordinates as input. Haralick proposes a calibration method that use the image coordinates of parallel lines.

Most camera calibration methods use calibration objects, for example cubes with colour patches. These calibration objects allow accurate control over their feature points and allow therefore accurate calibration.

Calibration of pictures of natural scenes requires a different approach because the calibration objects are too small. For example, assume that we need to calibrate the geometry of the images shown in Fig. 1. It is clearly impractical to build a calibration cube of this dimension.

As can be seen though, both images contain regular feature points that can be used for calibration. For example, if the distance between floors and between windows is known, then the position of the centres of the windows can be calculated. This is the basic method in our approach. However, the problem with real world scenes is that the features are hard to extract from the image. Furthermore, there are many feature points and to assign them manually would be time consuming and error prone. Lastly, not all feature points can be extracted from the image and some of them will be missing. The following section discusses a system to automatically deal with these

Figure 1: Sample Calibration Pictures

Skyscraper (Singapore)

Wall Street (New York)

problems.

3 Extraction of Calibration Points

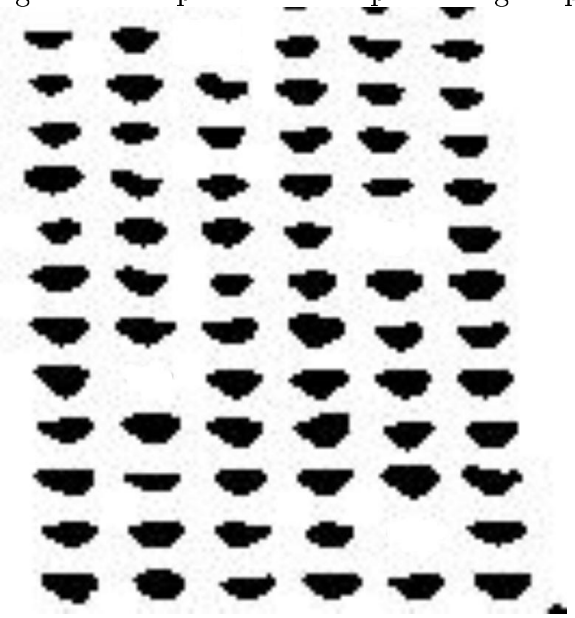
This section describes the algorithm for sorting the feature points and assigning real world coordinates to them. Our system uses a semi-automatic approach. First, a manual preprocessing step is used to segment the image (Subsection 3.1). Secondly, an automatic routine is used to compute the centres of the feature points, to sort them, to correct for missing features, and to assign real world coordinates to the points.

3.1 Image Preprocessing

The first step in the camera calibration routine is a manual preprocessing step to clean up the image, to remove unwanted artefacts, and to select suitable parameter settings for the colour segmentation routines. Figure 2 shows the output of the pre-processing step for the Singapore skyscraper (see Fig. 1).

As can be seen, some of the features were not distinct enough to be recognized and are missing in the preprocessed image.

Figure 2: Output of the Preprocessing Step



3.2 Sorting of Feature Points

This section describes the heart of the calibration routine — the algorithm to assign real world coordinates to the feature points that were extracted in the pre-processing step. The algorithm consists of two main parts: (a) algorithm 1 computes an initial guess of the translation matrix. (b) algorithm 2 is an iterative algorithm which assigns real world coordinates to the feature points and updates the guess of the transformation matrix.

In line 1 of algorithm 1, the centres of the features are extracted. Then in line 1, a number of seed points are selected. These seed points are used to compute the initial transformation matrix. Five seed points are selected to form a cross in the two-dimensional plane. To guarantee a good estimate of the initial transformation matrix, the five centres that form a cross closest to the centre of the image are selected. The routine `computeTransformationMatrix` called in line 1 uses the Boyer-Moore pseudo inverse method to compute a least means square (LMS) approximation for the 3×4 transformation matrix for the perspective projection. Since our system assumes that the distances between squares are not identical, the algorithm checks both possible orientations for the width and the height of the features and selects the best match (Line 1).

Algorithm 1 Algorithm to Assign Real World Coordinates

1: $S = \text{extractCentres}(Image)$
2: $P = \text{selectSeedPoints}(S)$
3: $M_{XY} = \text{computeTransformationMatrix}(S, \text{realWorldCoords}(S))$
4: $M_{YX} = \text{computeTransformationMatrix}(S, \text{swapCoord}(\text{realWorldCoords}(S)))$

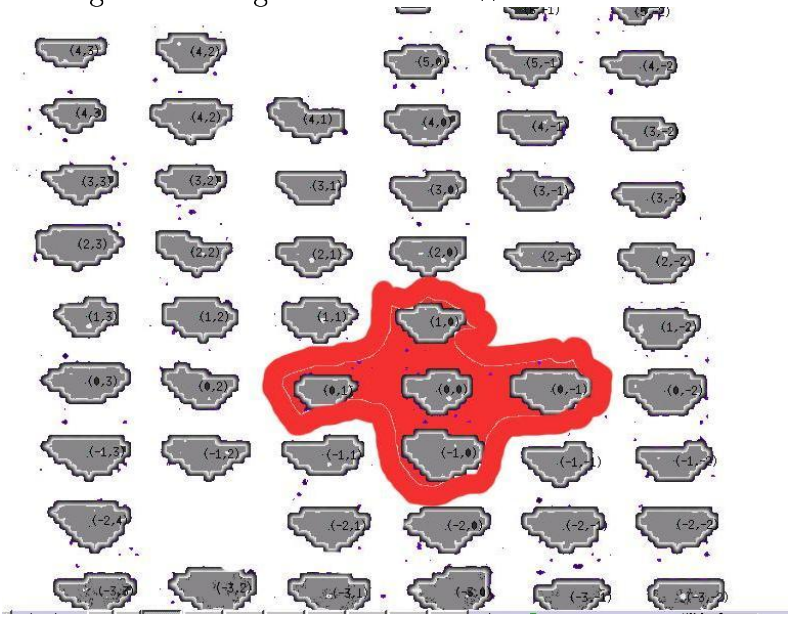
5: **if** $\text{error}(M_{XY}) < \text{error}(M_{YX})$ **then**
6: $\quad M_{Initial} = M_{XY}$
7: **else**
8: $\quad M_{Initial} = M_{YX}$
9: $\quad \text{swapCoordinates}(S)$
10: **end if**
11: $C_S = \text{assignCentres}(S, M_{Initial})$

Algorithm 2 describes the method for the assignment of real world coordinates to the feature points. This algorithm iteratively selects the unassigned feature points that are closest to an already assigned point. In line 2, the variables b_x and b_y contain the closest number of blocks given the current estimate M of the transformation matrix. This estimate of the number of blocks allows the algorithm to compensate for missing feature points. Once the real world coordinates have been assigned to the new point, a new transformation matrix is computed (line 2). This process is repeated until all points have been assigned.

Algorithm 2 Algorithm to Assign Real World Coordinates($S, M_{Initial}$)

1: $M = M_{Initial}$
2: $N =$
3: **while** $S \neq$ **do**
4: $\quad$ **for all** $s \in S$ **do**
5: $\quad\quad s_{North}, s_{East}, s_{South}, s_{West} = \text{findNearestNeighbors}(s, M)$
6: $\quad$ **end for**
7: $\quad$ **for all** $n \in \{s_{North}, s_{East}, s_{South}, s_{West}\}$ **do**
8: $\quad\quad b_x = dist((s-n)/width, M), b_y = dist((s-n)/height, M)$
9: $\quad\quad n_x, n_y = s_x + bx * width, s_y = by * height$
10: $\quad\quad N = N + n, S = S - n$
11: $\quad\quad M = \text{computeTransformationMatrix}(N, \text{realWorldCoords}(S))$
12: $\quad$ **end for**
13: **end while**
14: return N

Figure 3: Assignment of Real World Coordinates



The output of the system is shown in Fig. 3 shows the output of our system. All feature points are correctly aligned and the system correctly corrects for the missing features. The initial five seed points are shown in the gray shaded region.

4 Tsai Camera Calibration

After the computation of the matching points, we use a PD implementation of Tsai's camera calibration to compute the extrinsic and intrinsic parameters of the camera model.

The Tsai calibration method uses a four step process to compute the parameters of a pin hole camera with radial lens distortion.

Firstly, the position (X_T, Y_T, Z_T) and the orientation (R_X, R_Y, R_Z) of the camera with respect to the world coordinate system is computed. This involves solving a simple system of linear equations. This step translates the 3D World coordinates into 3D camera coordinates and computes the six extrinsic parameters of the camera model.

In Step 2, the perspective distortion of a pin hole camera is compensated for. This step is a non-linear approximation and computes the focal length f of the camera. The output of this step are the ideal undistorted image coordinates.

Thirdly, the radial lens distortion parameters (κ_1, κ_2) are computed. These parameters compensate for the pin cushion effect of video cameras, that is straight lines along the edges of the camera are rounded. The output of step 3 are the distorted image coordinates.

Lastly, the image coordinates are discretized into the real image coordinates by taking the number of pixels in each row and column of an image into consideration.

The last three steps compute five intrinsic parameters of the camera model (focal length, lens distortion, scale factor for the rows, and the origin in the image plane).

The Tsai method is a very efficient, accurate, and versatile camera calibration method and is therefore very popular in computer vision.

5 Conclusion

This paper presents a practical system for the accurate calibration of real world scenes that have a regular texture. Our system is currently limited to rectangular textures, but this limitation is due to the current implementation. In theory, any regular texture is suitable.

The Tsai calibration does not provide an efficient method for calculation of the uncertainty factor S_x given co-planar calibration points. We are investigating other suitable methods as the one described in [2].

References

[1] Jacky Baltes. Practical camera and colour calibration for large scale rooms. In *Proceedings of the IJCAI Workshop on RoboCup*, Stockholm, Sweden, July 1999.

[2] J. Batista, H. Araujo, and A.T. Almeida. Monoplanar camera calibration: Iterative multistep approach. In *BMVC93*, page xx, 1993.

[3] Hiroaki Kitano, editor. *RoboCup-97: Robot Soccer World Cup I.* Springer Verlag, 1998.

[4] Roger Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 364–374, Miami Beach, FL, 1986.