ABSTRACT

An Implementation of a Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-The-Shelf TV Camera and Lenses

by

Bolang Li

This thesis studies and implements a new versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV camera and lenses developed by Roger Tsai [1]. This technique builds up a unique relationship from the world coordinate system to the computer image coordinate system of calibration points by using a radial alignment constraint. The technique has advantage in terms of accuracy, speed, and versatility over existing techniques.

The fundamental knowledge for using this technique is presented in this thesis first, followed by an overview of the existing calibration techniques, and a detailed description of the new technique. The implementation is then presented step by step and is algorithm-oriented. Finally, the experimental results using real data are reported.

A precise calibration pattern, a CCD camera with zoom lens and a DADACUBE image acquisition system are used for the implementation of the calibration technique.

This thesis supplies the calibrated parameters for researchers who will use the CCD camera in their research, and may pave the way for future research in camera calibration.

AN IMPLEMENTATION OF A VERSATILE CAMERA CALIBRATION TECHNIQUE FOR HIGH-ACCURACY 3D MACHINE VISION METROLOGY USING OFF-THE-SHELF TV CAMERA AND LENSES

by

Bolang Li

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Electrical Engineering

Department of Electrical and Computer Engineering

May 1993

APPROVAL PAGE

An Implementation of a Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-The-Shelf TV Camera and Lenses

Bolang Li

Dr. Y. Q. Shi, Thesis Adviser   Date

Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. E. S. Hou, Committee Member   Date

Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. M. C. Zhou, Committee Member   Date

Assistant Professor of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Bolang Li

Degree: Master of Science in Electrical Engineering

Date: May 1993

Undergraduate and Graduate Education:

- Master of Science in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 1993

- Bachelor of Science in Electrical Engineering,
  University of Guangxi, China, 1984

Major: Electrical Engineering

Dedicated to my parents

ACKNOWLEDGMENT

The author would like to express his sincere gratitude to his advisor, Dr. Y. Q. Shi, for his guidance, friendship, concern and moral support throughout this research.

Special thanks to Dr. E. S. Hou and Dr. M. C. Zhou for serving as members of the committee.

Many thanks are due to Mr. Jingning Pan for his much help.

TABLE OF CONTENTS

Chapter **Page**

Chapter **Page**

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

CHAPTER 1
INTRODUCTION

As the beginning part of the thesis, this chapter introduces the basic concept of camera calibration techniques for 3D machine vision metrology. The important criteria for evaluating a camera calibration technique and the fundamental knowledge for realizing a camera calibration technique are presented.

1.1 The Role of Camera Calibration

High-accuracy measurement of 3D position is an important machine vision task in applications such as automation, robotics and automatic vehicle guidance. In this kind of measurement, parameters of TV cameras' internal geometrical and optical characteristics must be calibrated first so that the accurate measurement can be conducted.

Camera calibration in the context of 3D machine vision is the process of determining the internal camera geometric and optical characteristics (intrinsic parameters) and/or the 3D position and orientation of the camera frame relative to a certain world coordinate system (extrinsic parameters) for the following purposes.

1) Inferring 3D information from computer image coordinates. There are two kinds of 3D information to be inferred. They are different mainly because of the difference in applications.

a) The first kind of 3D information concerns the locations of objects, targets, or features. For simplicity, if the object is a point feature (e.g. a point spot in a mechanical part illuminated by a laser beam, or the corner of a electrical component on a printed circuit board), camera calibration provides a way of determining a ray in 3D space that the object point must lie on, given the computer image coordinates. With two views either taken from two cameras or one camera in two locations, the position of the object point can be determined by intersecting the two rays (see Figure 1-1).

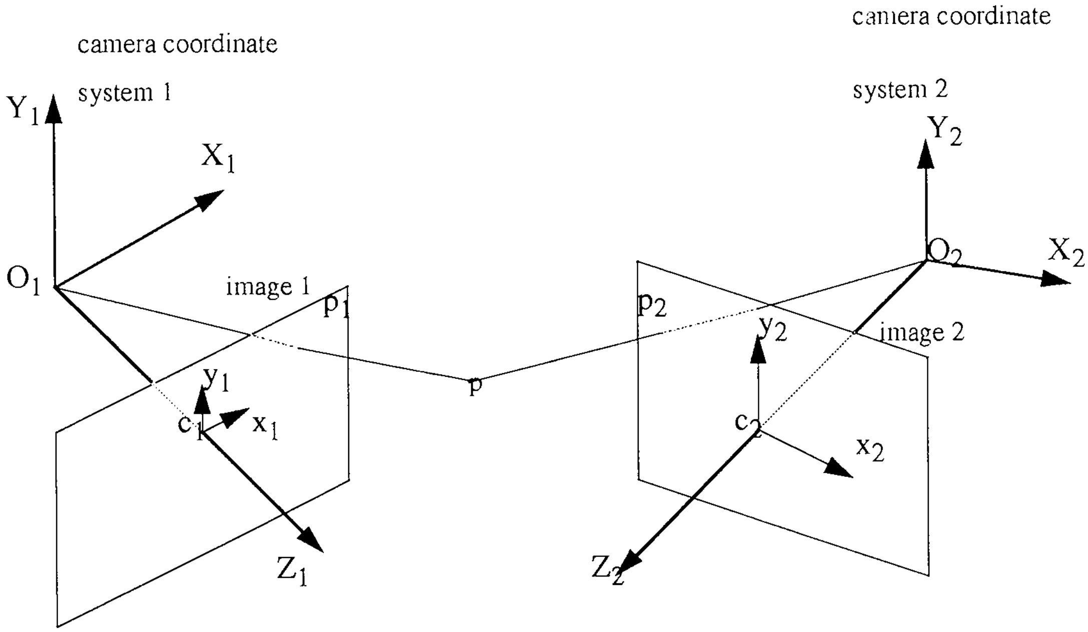


Figure 1-1 Three dimensional localization from two cameras

Both intrinsic and extrinsic parameters need to be calibrated. The applications include mechanical part dimensional measurement, automatic assembly of mechanical or electronics components, tracking, robot calibration and trajectory analysis. In the above applications, the camera calibration need to be done only once.

b) The second kind concerns the position and the orientation of moving camera (e.g., a camera held by a robot) relative to the target world coordinate system. The applications include robot calibration with camera-on-robot configuration and robot vehicle guidance.

2) Inferring 2D computer image coordinates from 3D information. In model-driven inspection or assembly applications using machine vision, a hypothesis of the state of the world can be verified or confirmed by observing whether the image coordinates of the object confirm to the hypothesis. In doing so, it is necessary to have both the intrinsic and

extrinsic camera model parameters calibrated so that the two-dimensional (2D) image coordinate can be properly predicted given the hypothetical 3D location of the object.

1.2 Important Criteria for Camera Calibration

The following criteria for camera calibration are important for serving the purposes stated in the previous section.

1) Autonomous: This means that the calibration procedure should not require operator intervention such as giving initial guesses for certain parameters. or choosing certain system parameters manually.

2) Accurate: Many applications such as mechanical part inspection, assembly, or robot arm calibration require an accuracy that is less than one hundred percent of the working range. Therefore camera calibration technique should have the potential of meeting such accuracy requirements. This requires that the theoretical modeling of the imaging process must be accurate and thus implies that the camera calibration should include lens distortion and perspective rather than parallel projection.

3) Reasonably efficient: Although the calibration needs not operate in real time, the complete camera calibration procedure should not include high dimension (more than five) nonlinear search (this takes a long time to do). Since type b) application mentioned in section 1.1 needs repeated calibration of extrinsic parameters, the calibration approach should allow enough potential for high-speed implementation.

4) Versatile: This means that the calibration technique should operate uniformly and autonomously for a wide range of accuracy requirements, optical setups, and applications.

5) Usage of common off-the-shelf camera and lens only: Most camera calibration techniques developed in the photogrammetric area require special professional cameras and processing equipment. Such requirements prohibit full automation and are labor-

intensive and time-consuming to implement. The advantages of using off-the-shelf solid state or vidicon camera and lens are as follows:

* versatile: this is because solid state cameras and lenses can be used for a variety of automation applications;

* availability: since off-the-shelf solid state cameras and lenses are common in many applications, they are at hand when you need them and need not be custom ordered;

* familiarity, user-friendly: because that not many people have the experience of operating professional metric cameras used in photogrammetry or the tetralateral photodiode with preamplifier and associated electronics calibration, it is convenient to use solid state type camera which is easy to interface with a computer and easy to install.

The purposes mentioned in section 1.1 can be best served if the above criteria for the camera calibration are met.

Among existing camera calibration techniques, a technique called two stage calibration technique introduced by Roger Tsai [1] is the best one according to how well it meet the criteria stated above. An overview of existing camera calibration technique and a description of the calibration technique introduced by Roger Tsai will be seen in chapter 2. This thesis researches work studies and implements this technique.

1.3 Camera Parameters and Transformation

In this section, camera parameters and some basic transformations that will be used in the thesis are defined. Both are the fundamental knowledge for camera calibration.

1.3.1 Camera Parameters

1) Effective focal length

The effective focal length denoted by f, is defined as the distance between front image plane of a CCD camera and the optical center. For thin lens, the effective focal length is fixed. It can be changed in zoom lens.

2) One-pixel width

One-pixel width on image plane denoted by d_x is the center to center distance between two adjacent sensor elements in the X (scan line) direction.

3) One-pixel height

One-pixel height on image plane denoted by d_y is the center to center distance between two adjacent CCD sensor elements in the Y direction.

4) Distortion coefficient and uncertainty scale factor

Distortion coefficient is denoted by k and uncertainty scale factor for image X coordinate is denoted by S_x. These are introduced due to a variety of factors, such as lens distortion, slight hardware timing mismatch between image acquisition hardware and camera scanning hardware, or imprecision of timing of TV scanning itself. Note that if a vidicon type camera is used, the sensor element or pixel mentioned earlier should be regarded as each individual resolution element in the receptor area with the resolution being determined by the sampling rate of computer scanning system. If a solid-state CCD or CID discrete array sensor is used and if full resolution is used, since the image is scanned line by line, the distance between adjacent pixels in Y direction is just the same as d_y, the center to center distance between two adjacent CCD sensor elements in y direction. Therefore, the uncertainty scale factor on image Y coordinate needs not be considered. The situation in X is different. The uncertainty scale factor for X coordinate S_x needs to be and can be calculated approximately by using the equation $S_x = f_c / f_s$, where f_c is the sample frequency of the CCD camera, and f_s is the sample frequency of the A/D converter in acquisition hardware.

1.3.2 Transformation

The following paragraphs review the basic transformations which will be used to build up the relation between the world coordinate system and the computer image coordinate system.

Translation

To translate a point with coordinates (X,Y,Z) to a new location (X_t. Y_t, Z_t) by using displacements (X_0, Y_0, Z_0) is accomplished by using the equations:

$$X_t = X + X_0$$
$$Y_t = Y + Y_0 \qquad (1\text{-}1a)$$
$$Z_t = Z + Z_0$$

or may be expressed in matrix form by writing

$$\begin{bmatrix} X_t \\ Y_t \\ Z_t \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Scaling

Scaling by factors S_x, S_y, and S_z along the X-,Y-, and Z- axes is given by the transformation matrix

$$S = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (1\text{-}2)$$

Rotation

To rotate a point about another arbitrary point in space requires three transformations: the first translates the arbitrary point to the origin, the second performs the rotation, and the third translates the point back to its original position.

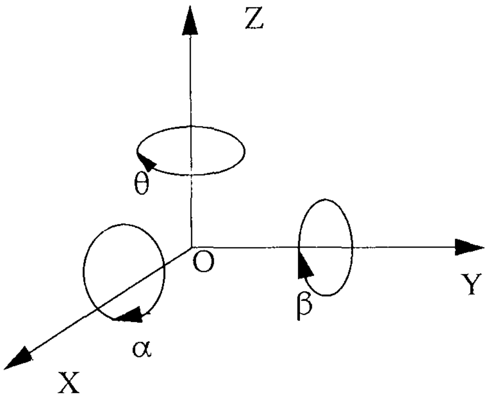


Figure1-2 Rotation about X-, Y-, Z-coordinate axes

With reference to Figure1-1, the rotation about the X coordinate axis by an angle α is achieved by using the transformation

$$R_\alpha = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The rotation angle α is measured clockwise when looking at the origin from a point on the + X axis. This transformation affects only the values of Y and Z coordinates.

Similarly, the rotation of a point about the Y axis by an angle β is performed by using the transformation

$$R_\beta = \begin{bmatrix} \cos\beta & 0 & -\sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (1\text{-}3b)$$

The rotation of a point about Z axis by an angle θ is achieved by using the transformation

$$R_\theta = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(1-3c)}$$

CHAPTER 2
CAMERA CALIBRATION TECHNIQUE

In this chapter, an overview of existing camera calibration technique is presented, followed by a description of the calibration technique that will be used in this thesis.

2.1 Overview of Existing Camera Calibration Techniques

There exist several camera calibration techniques which could be classified into four categories. The following paragraphs will discuss the strength and weakness of each category.

Category I --- Techniques involving full-scale nonlinear optimization: see [2] for example.

Advantage: It allows easy adoption of any arbitrarily accurate yet complex model for imaging. The best accuracy obtained in this category is comparable to the accuracy of the technique proposed by Roger Tsai [1] that is used in this thesis.

Problems: 1) It requires a good initial guess to start the nonlinear search. 2) It needs computer-intensive full-scale nonlinear search.

Following are some representative approaches for this kind of techniques.

* Classical approach: Faig's technique [3] uses a very elaborate model for imaging, uses at least 17 unknowns for each photo, and is very computer-intensive [3]. However, because of the large number of the unknowns, the accuracy is excellent. The root mean square (rms) error can be as good as 0.1 mil (1 inch = 10^3 mils). But this rms error is in photo scale, i.e., the error of fitting the model with the observations in image plane. When transformed into 3D error (i.e., error in 3D world coordinate system), it is comparable to the average error (0.5 mil) obtained by using monoview multiplane calibration technique, which is a typical case among various two-stage techniques. Another reason why such

photogrammetric techniques produce very accurate results is that large professional format photo is used rather than solid-state image array such as CCD. The resolution for such photos is generally three to four times better than that for the solid-state imaging sensor array.

* Direct linear transformation (DLT): Another example is the DLT technique developed by Abdel-Aziz [4]. One advantage of the DLT is that only linear equations need to be solved. However, it was later found by Karara [5], the co-inventor of the DLT, that unless lens distortion is ignored, full-scale nonlinear search is needed. Although Wong [6] mentioned that there are two possible procedures of using DLT (one entails solving linear equations only, and the other requires nonlinear search), the procedure using linear equation solving actually contains approximation. One of the artificial parameters he introduced, k_1, is a function of (x, y, z), the world coordinate and therefore not a constant. Nevertheless. the DLT bridges the gap between photogrammetry and computer vision so that both areas can use the DLT directly to solve camera calibration problem.

* Sobel [7] described a system for calibrating a camera using nonlinear equation solution. Eighteen parameters must be optimized. The approach is similar to Faig's method described earlier. No accuracy results were reported. Gennery [8] described a method that finds camera parameters iteratively by minimizing the error of epipolar constraints without using 3D coordinates of calibration points. It is mentioned in [9] that the technique is too error-prone.

Category II --- Techniques involving computing perspective transformation matrix first using linear equation solution: see [10] for example.

Advantage: No nonlinear optimization is needed.

Problems: 1) Lens distortion can not be considered. 2) The number of unknowns in linear equations is much larger than the actual number of equations (e.g., the unknowns to be solved are not linearly independent). The disadvantage of such redundant parameterization is that erroneous combination of these parameters can still make a good fit between experimental observations and model prediction in real situation when the observation is not perfect. This means the accuracy potential is limited in noisy situation.

Following are some representative approaches for this kind of techniques.

* Sutheriand [11] formulated very explicitly the procedure for computing the perspective transformation matrix given 3D world coordinates and 2D image coordinates of a number of points. It was applied to graphics applications, and no accuracy results are reported.

* Hall et al. [10] used a straightforward linear least square technique to solve for the elements of perspective transformation matrix for doing 3D curved surface measurement. The 2D computer image coordinates were tabulated, but no ground truth was given, and therefore the accuracy is unknown.

Category III --- Two-plane method: see [12] for example.

Advantage: Only linear equations need to be solved.

Problems: 1) The number of unknowns is at least 24 (12 for each plane), much larger than the degrees of freedom. 2) The formula used for the transformation between image and object coordinates is empirically based only.

A general calibration using the two-plane technique was proposed by Isaguirre et al. [13]. Full-scale nonlinear optimization is needed.

Category IV --- Geometric technique: see [12] for example.

Advantage: No nonlinear search is needed.

Problems: 1) No lens distortion can be considered. 2) Focal length is assumed given. 3) Uncertainty of image scale factor is not allowed.

Fischler [12] use a geometric construction to derive direct solution for the camera locations and orientation. However, none of the camera intrinsic parameters can be computed. No accuracy results was reported.

2.2 The Two-Stage Calibration Technique

The Two-Stage Calibration Technique introduced by Roger Tsai [1] has advantage in terms of accuracy, speed, and versatility over other techniques mentioned in Chapter 1. The fundamental basis of the technique is the radial alignment principle (to be described in Section 2.3) by which the following four steps of transformation from 3D world coordinate to computer image coordinate are established. The parameters are calibrated in each step.

2.2.1. The Four Steps of Transformation from 3D World Coordinate to Computer Image Coordinate.

Figure 2-1 illustrates the basic geometry of a camera model. (x_w, y_w, z_w) is the 3D coordinate of an object point P in 3D world coordinate system. (x, y, z) is the 3D coordinate of the object point P in the 3D camera coordinate system, which is centered at point O, the optical center, with the z axis the same as the optical axis. $(X\ O_i\ Y)$ is the image coordinate system center at O_i (the intersection point of the optical axis z and front image plane), and X, Y axes are parallel to x, y axes, respectively. (X_u, Y_u) is the image coordinate of (x, y, z) if a perfect pinhole camera model is used. (X_d, Y_d) is the actual image coordinate which differs from (X_u, Y_u) due to lens distortion. (X_f, Y_f) is the coordinate used in computer which is the number of pixels for the discrete image in the frame memory. To relate the image coordinate in the front image plane to the computer image coordinate system, additional parameters need to be specified and calibrated.

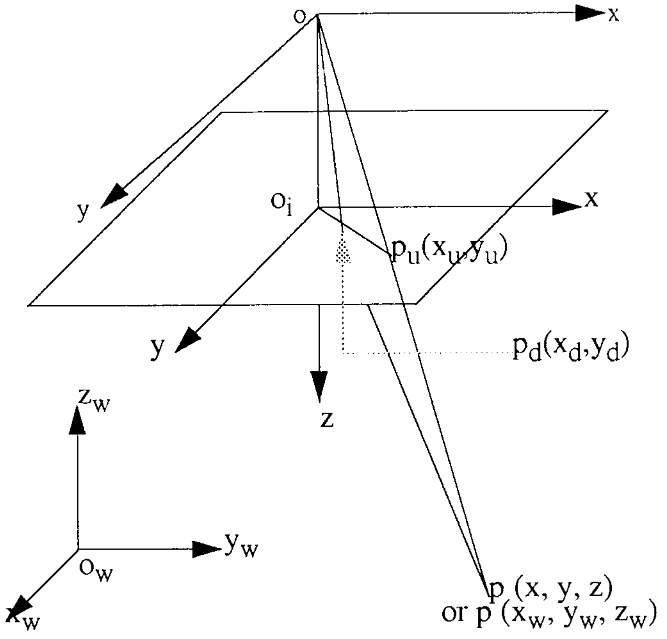


Figure2-1 Camera geometry perspective projection

The overall transformation from (x_w, y_w, z_w) to (X_f, Y_f) is depicted in Figure 2-2. The following is the transformations in analytic form for the four steps in Figure 2-2.

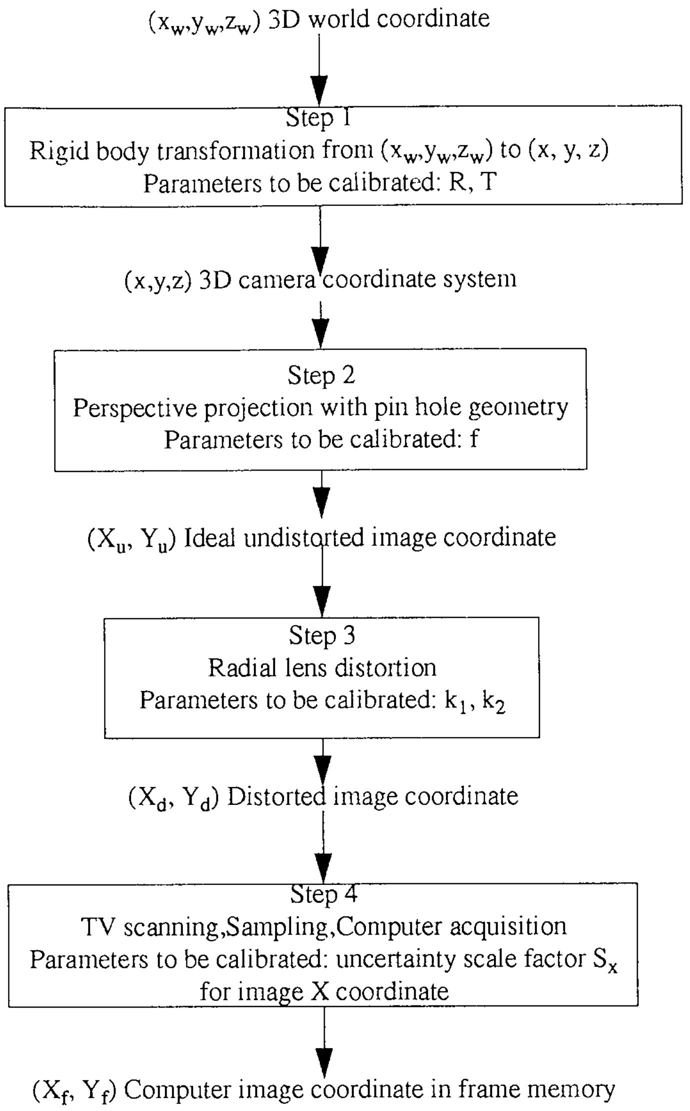

(x_w, y_w, z_w) 3D world coordinate
Step 1
Rigid body transformation from (x_w, y_w, z_w) to (x, y, z)
Parameters to be calibrated: R, T
(x,y,z) 3D camera coordinate system
Step 2
Perspective projection with pin hole geometry
Parameters to be calibrated: f
(X_u, Y_u) Ideal undistorted image coordinate
Step 3
Radial lens distortion
Parameters to be calibrated: k_1, k_2
(X_d, Y_d) Distorted image coordinate
Step 4
TV scanning,Sampling,Computer acquisition
Parameters to be calibrated: uncertainty scale factor S_x for image X coordinate
(X_f, Y_f) Computer image coordinate in frame memory

Figure 2-2 Four steps of transformation from 3D world coordinate to computer image coordinate.

Step 1: Rigid body transformation from the object world coordinate system (x_w, y_w, z_w) to the camera 3D coordinate system (x, y, z).

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = R \times \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + T \qquad (2\text{-}1)$$

where R is 3 x 3 rotation matrix

$$R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \tag{2-2}$$

and T is the translation vector

$$T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \tag{2-3}$$

The parameters to be calibrated in this step are R and T.

The rotation and translation can be viewed as a rotation parallel to the real world coordinate system and a translation into the origin, rotating and translating the camera world coordinate into the real world coordinate (Rogers, Adams [11]).

Step 2: Transformation from 3D camera coordinate (x, y, z) to ideal (undistorted) image coordinate (X_u, Y_u) using perspective projection with pinhole camera geometry, with reference to Figure 2-3.

$$X_u = f\frac{X}{Z} \tag{2-4a}$$

$$Y_u = f\frac{Y}{Z} \tag{2-4b}$$

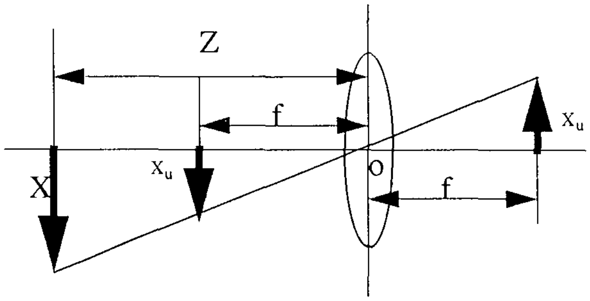


Figure 2-3 Pinhole camera model

The parameter to be calibrated in this step is the effective focal length f.

Step 3: Radial lens distortion. There are two kinds of lens distortion --- radial and tangential. For each kind of lens distortion, an infinite series is required. However according to Roger Tsai [1], for most of industrial machine vision applications where CCD or CID are frequently used, only radial distortion needs to be considered, and only one term is needed. Any more elaborate modeling not only would not help but also would cause numerical instability.

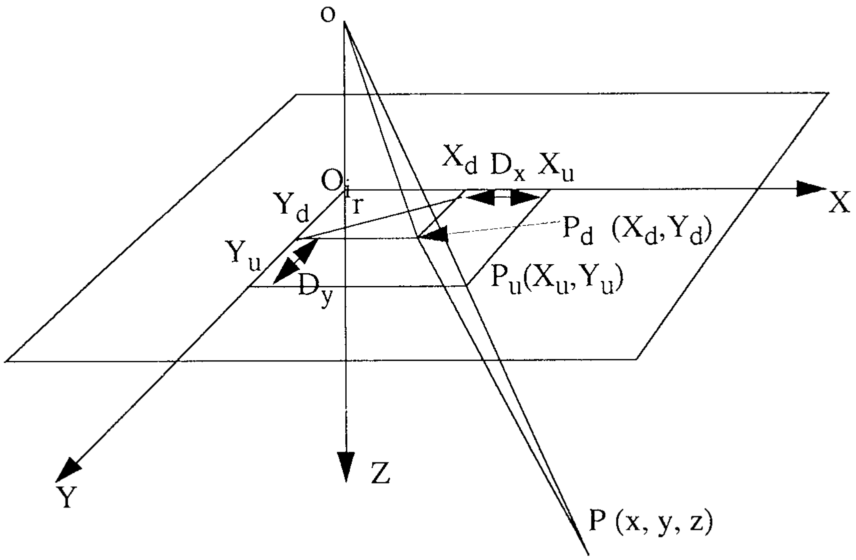


Figure 2-4 Modeling lens distortion

Figure 2-4 shows the modeling of radial lens distortion. The distance between X_d and Y_d is r. The distance between X_d and X_u is D_x. The distance between Y_d and Y_u is D_y. D_x and D_y can be calculated by using the following formula:

$$D_x = X_d\,(k_1 r^2 + k_2 r^4 + \ldots) \cong X_d k_1\, r^2 \qquad \text{(2-5a)}$$

$$D_y = Y_d\,(k_1 r^2 + k_2 r^4 + \ldots) \cong Y_d k_1\, r^2 \qquad \text{(2-5b)}$$

$$r = \sqrt{(X_d)^2 + (Y_d)^2} \qquad \text{(2-5c)}$$

Radial lens distortion is computed by

$$X_d + D_x = X_u \qquad (2\text{-}6a)$$

$$Y_d + D_y = Y_u \qquad (2\text{-}6b)$$

where (X_d, Y_d) is the distorted or real image coordinate on the image plane.

The parameter to be calibrated here is the distortion coefficient k_1.

Step 4. Transformation from real image coordinate (X_d, Y_d) to computer image coordinate (X_f, Y_f)

$$X_f = S_x \, d_x'^{-1} X_d + c_x \qquad (2\text{-}7a)$$

$$Y_f = d_y^{-1} Y_d + c_y \qquad (2\text{-}7b)$$

where X_f, Y_f are the row and column numbers of the image pixel in computer frame memory, respectively, c_x, c_y are the row and column numbers of the center of computer frame memory, respectively. The parameter d_x' and d_y are defined below.

$$d_x' = d_x \frac{N_{cx}}{N_{fx}}$$

where d_x center to center distance between two adjacent sensor elements in X (scan line) direction.

d_y center to center distance between two adjacent sensor elements in the Y direction.

N_{cx} number of sensor elements in the X direction.

N_{fx} number of pixels in a line as sampled by the computer.

The parameter to be calibrated here is the uncertainty image scale factor S_x.

S_x is the horizontal scale factor, an uncertainty scale-factor resulting from a mismatch between computer image acquisition hardware and camera hardware. It affects the one-pixel width in X direction on the image sensor as they appear in the computer frame memory.

$$d_x' = S_x d_x$$

where as defined above d_x is the one-pixel-width on sensor elements in X direction (scan line). d_x' is the one pixel width on computer image frame. S_x can be calculated approximately by using the equation $S_x = f_c / f_s$, where f_c is the sample frequency of the CCD camera, and f_s is the sample frequency of the A/D converter in acquisition hardware.

In this implementation, a CCD discrete array sensor is used. Since the image is scanned line by line, the distance between adjacent pixels in the Y direction is just the same as d_y. Therefore, (2-7b) is the right relationship between Y_d and Y_f. However, the situation in X is different. In TV camera scanning, an analog waveform is generated for each image line by zeroth-order sampling and holding. Then it is sampled by the computer into N_{fx} samples, Therefore

$$X = \frac{X_d N_{fx}}{d_x N_{cx}} \qquad (2\text{-}8)$$

where $X = X_f - c_x$, is the image coordinate.

2.2.2 Equations Relating the 3D World Coordinates to the 2D Computer Image Coordinates.

Now we are in a position to give the general equations relating the 3D world coordinates to the 2D computer image coordinates. By combining Step 2 to Step 4, the following equations relating (X, Y) computer coordinate to (x, y, z), the 3D coordinate of the object point in camera coordinate system can be derived.

$$S_x^{-1} d_x' X + S_x^{-1} d_x' X k_1 r^2 = f \frac{x}{z} \qquad (2\text{-}9a)$$

$$d_y Y + d_y Y k_1 r^2 = f \frac{y}{z} \qquad (2\text{-}9b)$$

where $r^2 = (S_x^{-1} d_x' X)^2 + (d_y Y)^2$, $X = X_f - c_x$, $Y = Y_f - c_y$

Substituting (2-1) into (2-9a) and (2-9b) gives

$$S_x^{-1}\ d_x'X + S_x^{-1}\ d_x'X\ k_1\ r^2 = f\frac{(r_1x_w + r_2y_w + r_3z_w + T_x)}{(r_7x_w + r_8y_w + r_9z_w + T_z)} \quad \text{(2-10a)}$$

$$d_y\ Y + d_y\ Y\ k_1\ r^2 = f\frac{(r_1x_w + r_2y_w + r_3z_w + T_y)}{(r_7x_w + r_8y_w + r_9z_w + T_z)} \quad \text{(2-10b)}$$

where $r^2 = (S_x^{-1}\ d_x'X)^2 + (d_y\ Y)^2$. The last two equations will be used in the next chapter to relate the 3D world coordinate (x_w, y_w, z_w) to the 2D computer image coordinate (X, Y).

2.2.3 Observations for the Two-Stage Calibration Technique

In the following, four observations are made, which are the fundamental basis of the two-stage calibration technique. They are illustrated in Figure 2-5.

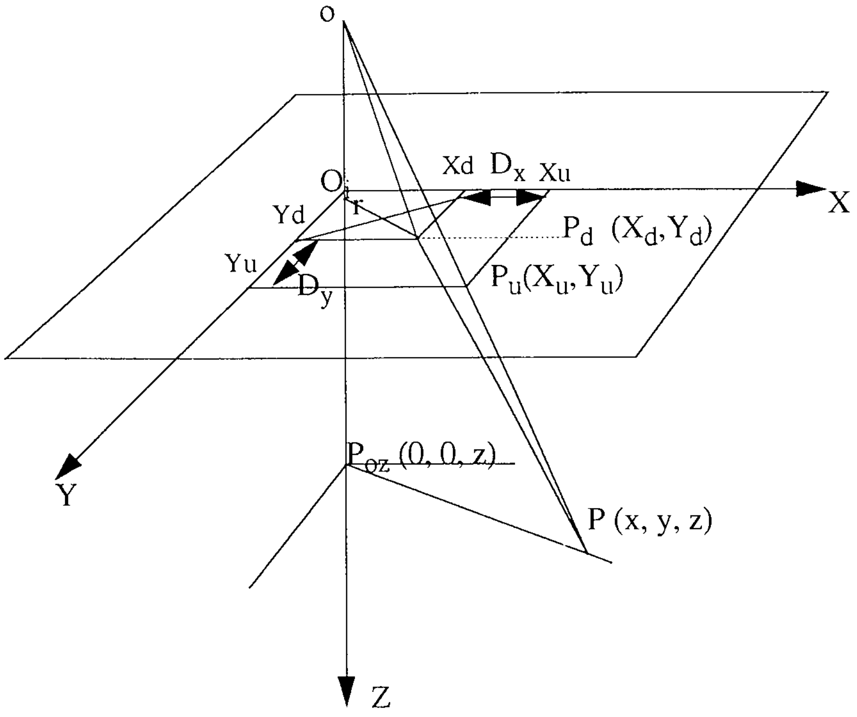


Figure 2-5 The radial alignment constraint

Observation I: Since the distortion is assumed to be radial, the direction of the vector O_iP_d remains unchanged no matter how much the distortion is (i.e., O_iP_d always lies on the line of O_iP_u), and is radially aligned with the vector $P_{oz}P$ extending from the optical axis (i.e., the point P_{oz} on the optical axis whose z coordinate is the same as that for the object point (x, y, z)) to the object point (x, y, z). For proof of this radial alignment constraint, see Roger Tsai [1].

Observation II: The effective focal length f also does not influence the direction of the vector O_iP_d since f scales the image coordinate X_d and Y_d by the same rate.

Observation III: If the world coordinate system is translated and rotated in x and y direction such that O_iP_d // $P_{oz}P$, then translation in z will not alter the direction of O_iP_d. This comes from the fact that, according to Equations (2-4a) and (2-4b), z changes Xu and Yu by the same scale, so that O_iP_u // P_iP_d.

Observation IV: O_iP_d is parallel to $P_{oz}P$ for every point and independent of the radial distortion, the effective focal length, and the z component of 3D translation vector. This allows the rotation matrix R and the x- and y- component of the transformation matrix T to be calculated from the world coordinate system.

For the proof of above radial alignment constraint, see Roger Tsai [1]. What we should know is that not only is the radial alignment constraint sufficient to determine uniquely the extrinsic parameters, but also the computation entails only the solution of linear equations with five to seven unknowns. This means the camera calibration can be done fast and automatically since no initial guess is needed, which is normally required for nonlinear optimization.

CHAPTER 3

IMPLEMENTATION

This chapter describes the implementation of the camera calibration technique introduced by Roger Tsai [1]. The experimental set up is first given, followed by the step by step description of the experimental procedure. The computation algorithm and their implementation are presented after the procedure. Finally, the test results are reported.

3.1 Experimental Set Up for Camera Calibration Using Monoview Coplanar Set of Points

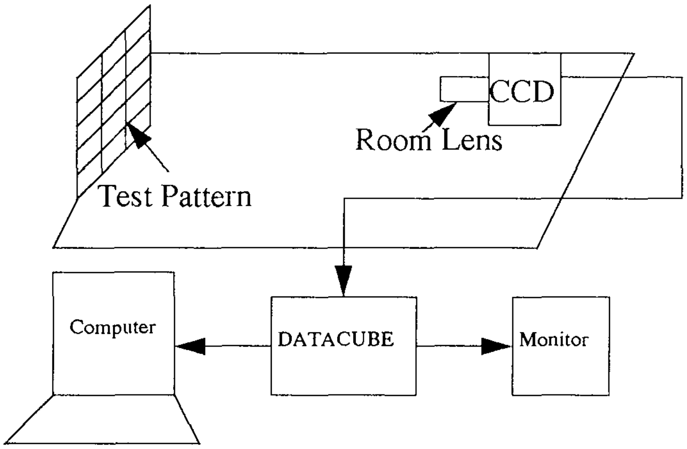


Figure 3-1 Experimental setup

Figure 3-1 shows the experimental set up for the calibration. A monoview coplanar set of points is used for the calibration. The test points are the intersection points of the horizontal lines and the vertical lines on a test post. The world coordinates of each point are known and the corresponding computer image coordinates are measured. In the experiment, a software called *xv* is used to measure the computer image coordinates.

In the actual setup, the (x_w, y_w, z_w) coordinate system are chosen such that $z_w = 0$ (This is realized by setting all test points in the same plane), and the origin is not close to the center of view of the camera coordinate system. The purpose for these is to make sure that T_y is not exactly zero.

3.2 The Computation Algorithm

This section describes the implementation procedure step by step. The presentation will be algorithm-oriented.

1) Stage 1 --- Compute 3D orientation and X- and Y- position of the object points.

i) Compute the distorted image coordinates (X_d, Y_d)

Procedure:

a) Grab a frame into the computer frame memory. Detect the row and column numbers of each calibration point i, call it (X_{fi}, Y_{fi}).

b) Obtain N_{cx}, N_{fx}, d_x, d_y using information of CCD camera and frame memory supplied by the manufacturer (see page 18 for definitions)

For this experiment

$N_{cx} = 492$

$N_{fx} = 512$

$d_x = 0.000525135$ (inch)

$d_y = 0.000679326$ (inch)

Calculate d_x' by using the formula $d_x' = d_x \dfrac{N_{cx}}{N_{fx}}$.

$d_x' = 0.000507504$ (inch)

c) Take (c_x, c_y) to be the center pixel of frame memory.

$$(c_x, c_y) = (256, 256)$$

d) Compute (X_{di}, Y_{di}) using

$$X_{di} = S_x^{-1} d_x'(X_{fi} - c_x)$$

$$Y_{di} = d_y(Y_{fi} - c_y)$$

for i = 1, 2,..., N, where N is the total number of calibration points. $S_x = f_c / f_s$, where f_c = 15.734 KHz, f_s = 15.098 KHz, S_x = 1.042 (see Chapter 2, Section 2.2.2 for definition of f_c and f_s).

ii) Compute the five unknowns $T_y^{-1}r_1$, $T_y^{-1}r_2$, $T_y^{-1}T_x$, $T_y^{-1}r_4$, and $T_y^{-1}r_5$.

Procedure: For each point i having (x_{wi}, y_{wi}), (x_{di}, y_{di}) as the 3D object world coordinate and the corresponding image coordinate (computed above) respectively, formulate the following equation with $T_y^{-1}r_1$, $T_y^{-1}r_2$, $T_y^{-1}T_x$, $T_y^{-1}r_4$ and $T_y^{-1}r_5$ as unknowns:

$$\begin{bmatrix} Y_{di}x_{wi} & Y_{di}y_{wi} & Y_{di} & -X_{di}x_{wi} & -X_{di}y_{wi} \end{bmatrix} \begin{bmatrix} T_y^{-1}r_1 \\ T_y^{-1}r_2 \\ T_y^{-1}T_x \\ T_y^{-1}r_4 \\ T_y^{-1}r_5 \end{bmatrix} = X_{di}$$

With N (the number of object points) much larger than five, an overdetermined system of linear equations can be established and solved for the five unknowns $T_y^{-1}r_1$, $T_y^{-1}r_2$, $T_y^{-1}T_x$, $T_y^{-1}r_4$, $T_y^{-1}r_5$.

iii) Compute $(r_1, r_2,..., r_9, T_x, T_y)$ from $(T_y^{-1}r_1, T_y^{-1}r_2, T_y^{-1}T_x, T_y^{-1}r_4, T_y^{-1}r_5)$:

a) Compute T_y^2 from $T_y^{-1}r_1$, $T_y^{-1}r_2$, $T_y^{-1}T_x$, $T_y^{-1}r_4$, $T_y^{-1}r_5$:

Procedure: Let c be a 2*2 matrix, defined as

$$c = \begin{bmatrix} \frac{r_1}{T_y} & \frac{r_2}{T_y} \\ \frac{r_4}{T_y} & \frac{r_5}{T_y} \end{bmatrix} \qquad (3\text{-}13)$$

If there is not a whole row or column of c that vanishes, then T_y^2 can be computed with the next equation

$$T_y^2 = \frac{S_r - [S_r^2 - 4(r_1'r_5' - r_2'r_4')^2]^{\frac{1}{2}}}{2(r_1'r_5' - r_2'r_4')^2}$$

where $S_r = r_1'^2 + r_2'^2 + r_4'^2 + r_5'^2$, $r_1' = r_1 / T_y$, $r_2' = r_2 / T_y$, $r_4' = r_4 / T_y$, $r_5' = r_5 / T_y$.

Else compute T_y^2 with

$$T_y^2 = (r_i'^2 + r_j'^2)^{-1}$$

where r_i' and r_j' are the elements in the row or column of c which do not vanish.

b) Determine the sign of T_y.

After T_y^2 has been found, T_y can then be obtained if the sign of T_y is determined.

Procedure:

1) Pick up an object point i whose computer image coordinate (X_{fi}, Y_{fi}) is away from the image center (c_x, c_y) and the object world coordinate is (x_{wi}, y_{wi}, z_{wi}).

2) Pick the sign of T_y to be + 1.

3) Compute the following:

$$r_1 = T_y^{-1} r_1 T_y \qquad r_2 = T_y^{-1} r_2 T_y$$

$$r_4 = T_y^{-1} r_4 T_y \qquad r_5 = T_y^{-1} r_5 T_y$$

$$T_x = T_y^{-1} T_x T_y$$

$$x = r_1 x_w + r_2 y_w + T_x \qquad y = r_4 x_w + r_5 y_w + T_y$$

where $T_y^{-1} r_1$, $T_y^{-1} r_2$, $T_y^{-1} T_x$, $T_y^{-1} r_4$, $T_y^{-1} r_5$ are determined in ii).

4) If x and X have the same sign and y and Y have the same sign

then sign $(T_y) = + 1$,

else sign $(T_y) = - 1$.

c) Compute the 3D rotation matrix R, i.e., its entries $r_1, r_2, \ldots,$ and r_9.

Procedure:

Compute R with the following formula:

$$R = \begin{bmatrix} r_1 \; r_2 \; (1 - r_1^2 - r_2^2)^{\frac{1}{2}} \\ r_4 \; r_5 \; s(1 - r_4^2 - r_5^2)^{\frac{1}{2}} \\ r_7 \; r_8 \qquad r_9 \end{bmatrix}$$

where s = - sign $(r_1r_2+r_3r_4)$. Sign (arg) is equal to +1 or -1 depending on the sign of its argument arg. r_7, r_8, r_9 are determined from the outer product of the first two rows using the orthonormal and right-handed property of R. That is,

$$\text{if} \qquad R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \tag{3-16a}$$

then

$$r_7 = (1-r_1^2-r_4^2)^{1/2}$$

$$r_8 = (1-r_2^2-r_5^2)^{1/2}$$

$$r_9 = (1-r_7^2-r_8^2)^{1/2}$$

3) If $f< 0$ (obtained by using (3-16a)), then the sign of the elements r_3, r_6, r_7, r_8 in matrix R must be reversed, i.e.

$$R = \begin{bmatrix} r_1 & r_2 & -r_3 \\ r_4 & r_5 & -r_6 \\ -r_7 & -r_8 & r_9 \end{bmatrix} \tag{3-16b}$$

For the proof of (3-16b), see Roger Tsai[1]

Note that in the experiment, to select a valid one between the two solutions in (3-16a) and (3-16b), we use the linear equation in (3-17) below for computing approximation of f and T_z by ignoring distortion. The wrong one will yield negative f and the right one will yield positive f.

Stage 2 Compute effective focal length f, distortions coefficient $k_1$and displacement along z axis T_z:

d) Compute an approximation of f and T_z by ignoring lens distortion.

Procedure: For each calibration point i, establish the following linear equation with f and T_z as unknowns:

$$[y_i - d_y (Y_{fi} - c_y)\,] \begin{bmatrix} f \\ T_z \end{bmatrix} = w_i d_y (Y_{fi} - c_y) \qquad (3\text{-}17)$$

where $y_i = r_4 x_{wi} + r_5 y_{wi} + T_y$

$w_i = r_7 x_{wi} + r_8 y_{wi}$

With several object calibration points, this yields an overdetermined system of linear equations that can be solved for the unknowns f and T_z. Note that the calibration plane must not be exactly parallel to the image plane, otherwise (3-17) becomes lineally dependent.

e) Compute the exact solution for f, T_z, k_1.

Procedure: Solve (3-10b) with f, T_z, k_1 as unknowns using steepest descent optimization, use the approximation value for f and T_z computed in d) as initial guess, and zero as the initial guess for k_1.

3.3 Mathematical Algorithms and Their Implementation

3.3.1. Approximations Using Least Square Method

The ability to obtain accurate measurements is limited by the error of the measurements and the nature of the measured tools. One approach to overcome this problem is to take much more measurements than needed. If the errors are expected to be random, with much more measured data than unknowns, an overdetermined system can be established and can be used for a least square solution.

Using m observations, an overdetermined system can be established leading to an approximation of n unknowns (with m >> n). Consider a linear system Ax = b, where

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

$$\mathrm{x} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$$

$$b = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_m \end{bmatrix}$$

There may not exist a choice of x that perfectly fits the data b. In other words, probably the vector b will not be a combination of the column vectors of A, it will be outside the column space. The problem is to chose such an x that the error is minimized, and this minimization will be carried out in the least squares sense. The error is E = || Ax - b || (||k|| is the length of a vector k). This is exactly the distance from b to the to the point

Ax in the column space (Note that Ax is the combination of the columns with coefficients $x_1, ..., x_n$). Therefore searching for the least squares solution x, which minimizes E, is the same as locating the point p = Ax which is the closest point to b. With reference to Figure 3-2, p must be the projection of b onto the column space and the error vector (b-Ax) must be perpendicular to that space.

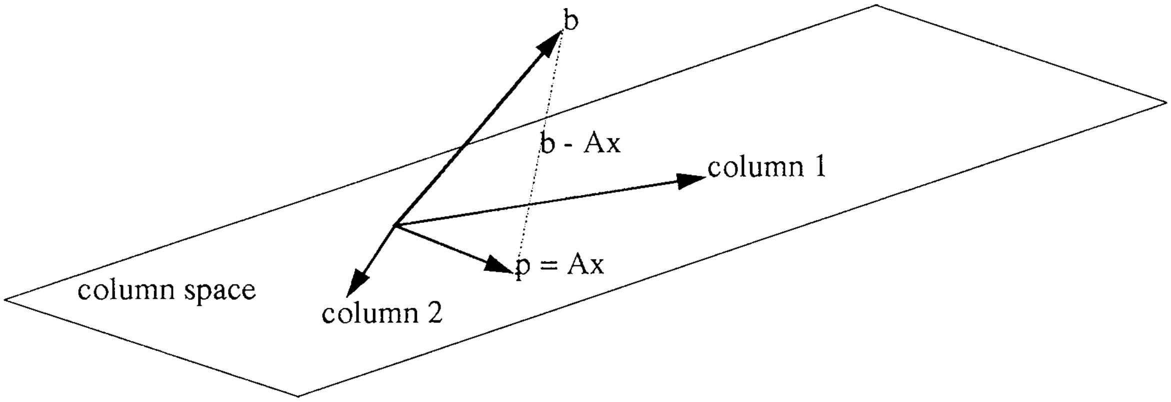


Figure 3-2 Projection onto the column space of a 3 by 2 matrix.

The calculation of x is performed in the following way:

The error vector must be in the nullspace of A^T.

$$A^T(b-Ax) = 0 \qquad \text{or} \qquad A^TAx = A^Tb$$

Therefore, the least squares solution to an inconsistent system Ax = b of m equations with n unknowns satisfies $A^TAx = A^Tb$ and if the column of A are linearly independent, then A^TA is invertible and

$$x = (A^TA)^{-1}A^Tb. \qquad (3\text{-}18)$$

This is the unique least square solution.

3.3.2 Implementation of Least Square Solution

The realization of Equation (3-18) involves matrix transposition, matrix multiplication and matrix inversion. The program for the least square solution is available in computer Lab. To aid those readers who intend to understand the program, the following presentation will be algorithm-oriented.

Matrix Transposition

Matrix transposition is defined for nxn matrix and is easy to implement: the rows and the columns of a square matrix are interchanged, i.e., the element n_{ij} is replaced by the element n_{ji}, and vice versa, while the elements $n_{i=j}$ remain unchanged. The algorithm for program could be stated as following (the program source code can be found in appendix):

```
for i = 0, i < row
    for j = 0, j < n
        transposed [j] [i] = matrix [i] [j]
```

Matrix Multiplication

Matrix multiplication can be performed if the number of rows of the first matrix are identical to the number of columns of the second matrix. It is executed by multiplying the elements of the row of the first array with the elements of the columns of the second array. The " intersection " of the row and the column is the " location " of the element in the new matrix (for example, if c_{ij} is the new matrix, then i is row number of the left-hand matrix and j is the column number of the right-hand matrix). A more accurate definition of matrix C as the product of two matrices A and B is:

$$C_{ij} = \sum_{k=1}^{n} (a_{ik} \times b_{kj})$$

where n is both the number of rows in matrix A, A = $\{a_{ij}\}$ and the number of columns in matrix B, B = $\{b_{ij}\}$.

The algorithm for program could be following (the program source code can be found in appendix):

```
for i = 0, i < column B
for j = 0, j < row A
for k = 0, k < row b
matrix [j] [i] = matrix [j] [i] + matrix [j] [k] * matrix [k] [i]
```

Matrix Inversion

Matrix inversion requires more complex computation than above two operations. It is based on the fact that for a nonsingular matrix A, the following holds:

$$AA^{-1} = A^{-1}A = I$$

where I is the identity matrix. If a matrix A is multiplied by a matrix B and the result is the identity matrix, then $B = A^{-1}$.

To obtain a matrix B which satisfies AB = I, the matrix A is first extended by a identity matrix to the form of

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & \circ & 1 & 0 & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2n} & \circ & 0 & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \circ & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & \circ & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

then performed by using Gaussian Elimination to an upper triangle matrix:

$$\begin{bmatrix} a_{11}' & a_{12}' & a_{13}' & \cdots & a_{1n}' & \circ & b_{11} & b_{12} & b_{13} & \cdots & b_{1n} \\ 0 & a_{22}' & a_{23}' & \cdots & a_{2n}' & \circ & 0 & b_{22} & b_{23} & \cdots & b_{2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \circ & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & a_{nn}' & \circ & 0 & 0 & 0 & \cdots & b_{nn} \end{bmatrix}$$

and continued with Jordan Reduction to transfer matrix A to identity matrix. After these operations, matrix B comes out

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 & \circ & b_{11} & b_{12} & \dots & b_{1n} \\ 0 & 1 & 0 & \dots & 0 & \circ & b_{21} & b_{22} & \dots & b_{2n} \\ \dots & \dots & \dots & \dots & \dots & \circ & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & \circ & b_{m1} & b_{m2} & \dots & b_{mn} \end{bmatrix}$$

The matrix B is nothing but the inverted matrix A.

The algorithm for program could be following (the program source code can be found in Appendix):

```
        form augmented matrix
        for i = 0, i < n-1, do L1
L1:     if matrix [i] [i] == 0
        search for row m >= j with matrix [n] [i]!= 0
        if found then exchange row[i] with row[m]
        else print: no unique solution
        for j = i+1, j < n (j is the next row after i)
        m = matrix [j] [i] / matrix [i] [j]
        for row j do
        matrix [i] [k] = matrix [j] [k] - m * matrix [i] [k]
        if matrix [n-1][n-1] == 0
        then print: no unique solution
        else for j = n-1, j > 0, do L2
L2:     m = matrix [j] [j]
        for l = j, l < 2*n, normalize row matrix [j] [j]:
        matrix [j] [l] = matrix [j] [l] / m
        for i = j-1, i >= 0
        m = matrix [i] [j]
        for k = j, k < 2*n, do Gauss-Jordan Reduction
```

```
matrix [i] [k] = matrix [i] [k] - m * matrix [j] [k]
m = matrix [0] [0]
for l = 0, l < 2*n, normalize row [0] to 1:
matrix [0] [l] = matrix [0] [l] / m
```

3.3.3 Approximation Using Steepest Descent

The idea of this method is to minimize functions with respect to a set of unconstrained parameters. This method requires only computations of the objective function F(v) and gradient vector G(v). Each computation of a direction vector and corresponding minimization in one variable is referred to as an iteration.

Let $F(v) = F(x, y)$ (3-19)

With reference to Figure 3-3, to find the solution x, y which satisfies F (x, y) = 0, we can start with an initial value (x_0, y_0), go along the direction in which the value of F decreases until the value of F approach zero and the desired approximation solution is obtained.

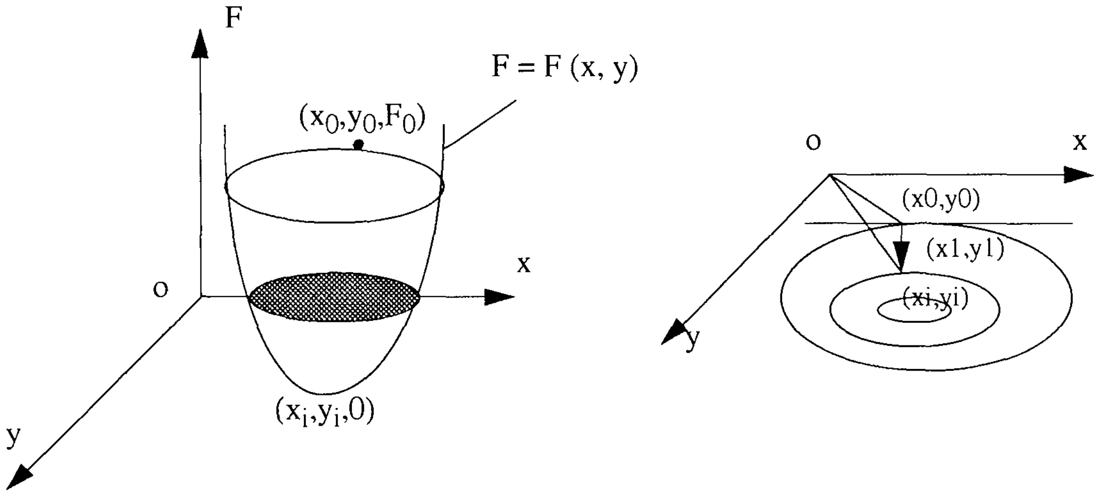


Figure 3-3 Steepest descent method

As we know, the gradient direction at a point (x, y) is defined as

$$(\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y})^T$$

and its direction is the direction in which the value of F raises fastest. Therefore, the opposite direction -g is the direction that the value of F descend fastest. The steepest descent method is to descend the value of F step by step along this direction (-g) until the criteria are satisfied. Above discussion of the steepest descent method is based on two dimensional variables. It can be extended to any dimensional variables directly.

3.3.4 Implementation of Steepest Descent

Let (x_0, y_0, z_0) be the initial approximation solution. Compute the gradient of F at point (x_0, y_0, z_0):

$$g_0 = (g_{x0}, g_{y0}, g_{z0})^T$$

where $g_{x0} = (\frac{\partial F}{\partial x})_{x=0}$

$$g_{y0} = (\frac{\partial F}{\partial y})_{y=0}$$

$$g_{z0} = (\frac{\partial F}{\partial z})_{z=0}$$

The next approximation (x_1, y_1, z_1) is obtained by the following equations:

$$x_1 = x_0 - \lambda g_{x0}$$

$$y_1 = y_0 - \lambda g_{y0}$$

$$z_1 = z_0 - \lambda g_{z0}$$

where λ is a step length.

Use (x_1, y_1, z_1) as a new approximation solution, repeat the above computation until some point (x_i, y_i, z_i) is obtained such that $|F(x_i, y_i, z_i)| < c$, where c is the desired criteria.

The algorithm used in the experiment is as the following (the program source code can be found in Appendix):

```
      input initial value $(x_0, y_0, z_0)$
      for i = 0, i <= size do
L1:   $g_x[i] = (\frac{\partial F}{\partial x})$

      $g_y[i] = (\frac{\partial F}{\partial y})$

      $g_z[i] = (\frac{\partial F}{\partial z})$

      $x[i+1] = x[i] - \lambda g_x[i]$

      $y[i+1] = y[i] - \lambda g_y[i]$
      $z[i+1] = z[i] - \lambda g_z[i]$

      if F[i+1] <= c
      then stop
      else if F[i+1] <= F [i]
          then i = i+1
          else $\lambda = \lambda/2$
      repeat
```

3.4 The Test Result

This section will describe the experiment operation and give the experiment result.

3.4.1 Experiment Hardware

The vision system used in this experiment is build upon a CCD TV camera, a computer station and an image data acquisition hardwares.

The CCD TV camera, SONY M-852 from CHORI company, has a resolution of 510(H) x 492(V) picture elements. A zoom lens is mounted on the TV camera. The image processing software is implemented on a computer station. The image acquisition hardwares are from DATACUBE company.

In this thesis, the calibration points are created by making a set of horizontal lines and a set of vertical lines (see Figure 3-4). The horizontal lines are parallel to each other and are 0.5 inch apart from each other. These rules also apply to the vertical lines. A total of 64 points which are the intersection points of horizontal lines and vertical lines are chosen for the experiment. The CCD camera is mounted 90 inches away from the calibration plane. The output of the CCD camera is connected to the DATACUBE electronic circuit boards and the output of the DATACUBE is connected to both the computer station and the TV screen which is used to monitor the images (see Figure 3-1).

The test pattern is placed on a level board. The origin of the real world coordinate system in the experiment is chosen away from the north-west point shown in Figure 3-4 by an offset of (200, 200) (inches). The X-axis is paralleled to the horizontal lines and Y-axis is parallel to the vertical lines. The X-Y plane of the world coordinate system is on the same plane as the test pattern. The level board must be precisely leveled so that the Z-components of each test point in the world coordinate system are exactly zero. The level board is perpendicular to the working table and is not perpendicular to the optical axis of the CCD camera (see Figure 3-4).

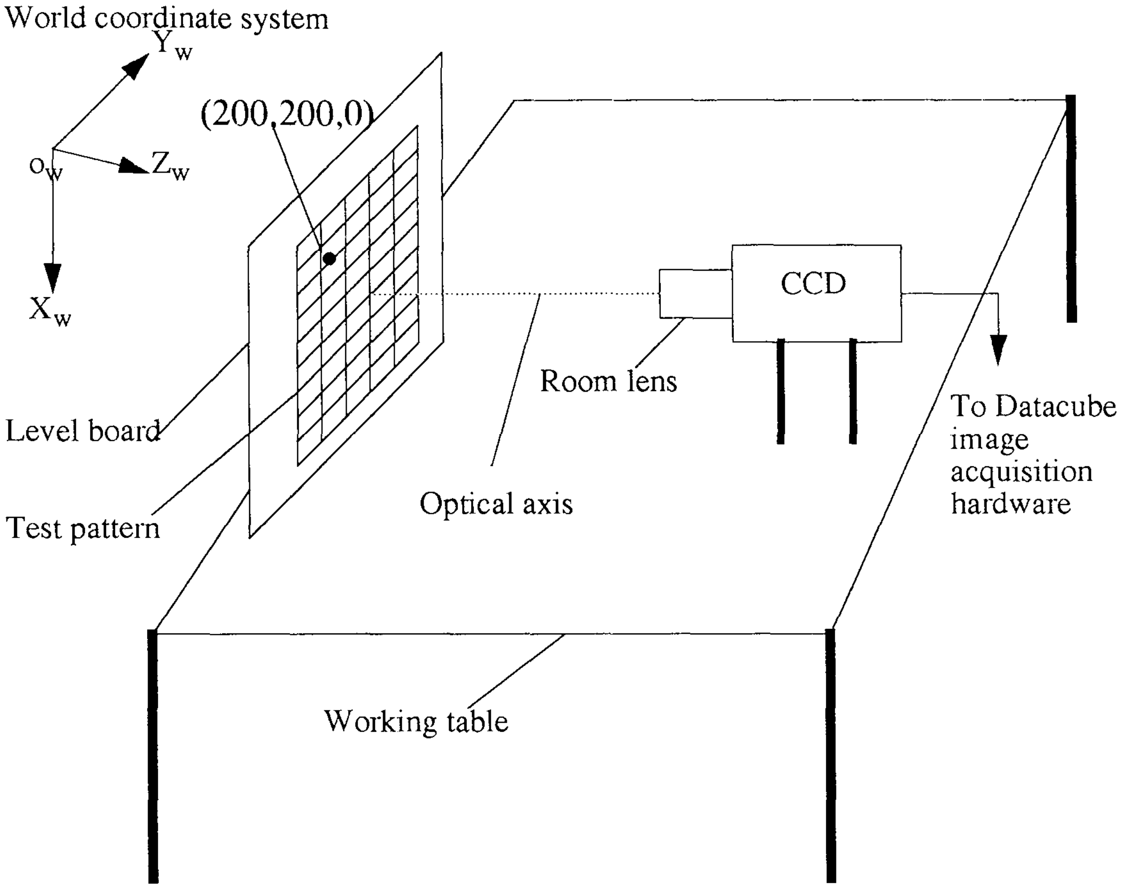
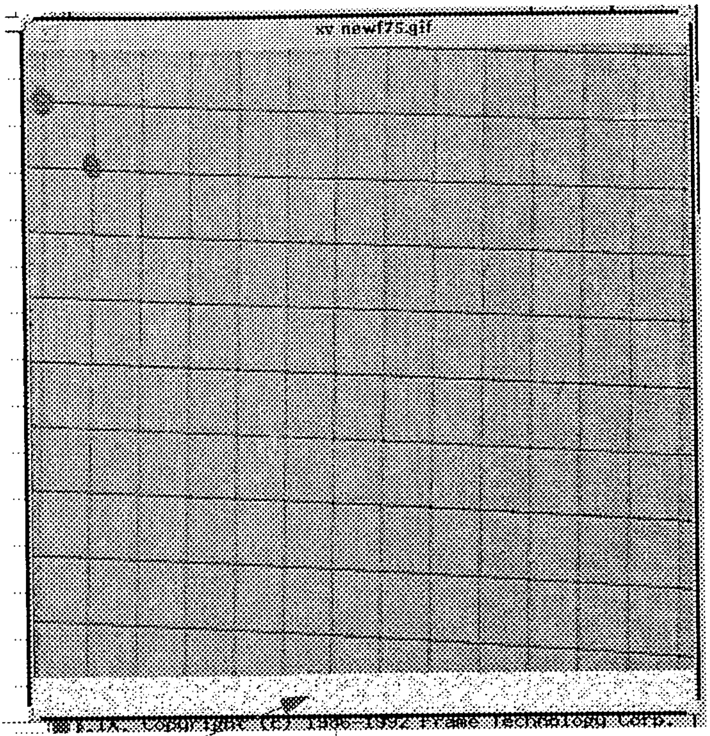


Figure 3-4 The test pattern

3.4.2 Calibrating a TV Camera

This section will present the experimental procedure for calibrating a TV camera.

Login in the Imager which is the name of a computer station used in this experiment. Type command "image1" under the path of /home/imager/bli. A picture of the test pattern will appear on the TV screen. Focus the lenses until the best quality of the picture is obtained. Type command "image2" under the same path. The image will be stored in the computer with the default file name "picture". Change this file into a ".gif " file so that it can be displayed in the computer screen. Figure 3-5 shows the test pattern viewed in computer screen.



blank lines on the bottom of the image

Figure 3-5 Test pattern viewed in computer screen

Obtain the row and the column number of each calibration point in the computer image plane. Obtain the image center by taking the apparent center of the computer image frame buffer offset by the number of blank lines. Type command "cal " to run the calibration program. Input the data according to the prompt set by the program. The final calibration results will appear on the computer screen.

3.4.3 Experiment results

There are four data files available from the camera calibration:

cal1.dat --- Containing the information used to solve the five unknowns $T_y^{-1}r_1$, $T_y^{-1}r_2$, $T_y^{-1}T_x$, $T_y^{-1}r_4$, $T_y^{-1}r_5$.

cal2.dat --- Containing the information used to solve the approximation value of f and T_z.

cal3.dat --- Containing the information of R matrix and T_y.

cal4.dat --- Containing the information of f, T_z and k_1.

For the CCD M-852 TV camera with a zoom lens, set the marked focal length f equal to 75 mm. The image coordinates in the computer frame memory and the corresponding world coordinates are shown in Table 3-1.

Table 3-1 Image coordinates in the computer frame memory and the corresponding world coordinates

X_{fi}(row)	Y_{fi}(column)	X_{wi}(inch)	Y_{wi}(inch)
94	88	200.5	200.0
145	125	201.0	200.5
197	161	201.5	201.0
199	198	202.0	201.0
251	235	202.5	201.5
253	273	203.0	201.5
303	272	203.0	202.0
306	310	203.5	202.0
358	374	204.0	202.5
360	385	204.5	202.5
414	423	205.0	203.0

The calibrated parameters are:

f = 75.538 mm

$T_x = 194.523$

$T_y = 203.377$

$T_z = 159.283$

$k_1 = 1.028$

$$R = \begin{bmatrix} 0.926 & 0.048 & -0.372 \\ 0.322 & 0.996 & 0.080 \\ 0.424 & 0.019 & 0.925 \end{bmatrix}$$

For different focal length of the CCD camera, repeat the same procedure described above.

The measured parameters are:

$\theta = 25^{o}$

$\phi = 0^{o}$

$\psi = 0^{o}$

where θ is the yaw angle, ϕ is the pitch angle, and ψ is the tilt angle for rotation.

f = 75.0 mm

$T_x = 209.0$

$T_y = 205.5$

$T_z = 157.0$

$$R = \begin{bmatrix} 0.906 & 0 & -0.423 \\ 0.423 & 1 & 0 \\ 0.423 & 0 & 0.906 \end{bmatrix}$$

Although the desired accuracy of the calibration technique developed by Roger Tsai was proved theoretically (see Roger Tsai [1]) which is good enough to serve most of practical applications for high accuracy 3D machine vision, it is always not easy to obtain a high accuracy ground truth for camera calibration parameters that can serve as absolute reference. One way to assess the accuracy of the calibrated parameters is to see how well it can sense or measure the 3D world. This could be a future research subject.

CHAPTER 4

SUMMARY AND DISCUSSION

This chapter will give a summary of this thesis and a suggestion for further work.

4.1 Conclusion

In this thesis, we explored several kinds of camera calibration technique for 3D machine vision metrology with a concentration on a new camera calibration technique developed by Roger Tsai. This technique establishes a unique relationship from world coordinate system to computer image coordinate system. The experiment conducted in this thesis demonstrated that this technique is efficient in computation and feasible in implementation. It satisfies several requirements for camera calibration, i.e., autonomous, accurate, reasonably efficient, versatile and usage of common off-the-shelf camera and lens only.

Both theory, concept and detail procedures and algorithms for the implementation of the technique are presented. The effective focal length, the lens correction factor and the image scale factor are successfully calibrated. These calibrated parameters could be used by those who want to use the CCD camera in their research, and the entire work in this thesis may pave the way for further research in camera calibration.

4.2 Future Work

One of the future research subjects is to calibrate a camera using monoview noncoplanar points. The same pattern used in coplanar case can be used, except that it is moved to several different heights by z direction. All procedures are basically the same as those presented in this thesis, except that the linear matrix equation derived from the radial alignment constraint yield solutions for seven unknowns instead of five, since Z_w is no longer identically zero. For more details see Roger Tsai [1].

APPENDIX: PROGRAMS

```
                    PROGRAM LESQ0

   C     LEAST SQUARE METHOD SOLVED FOR THE FIVE UNKNOWNS
```

$T_y^{-1}r_1$, $T_y^{-1}r_2$, $T_y^{-1}T_x$, $T_y^{-1}r_4$, $T_y^{-1}r_5$ C

```
3            INTEGER LDA, LDX, NCA, NRA, NUMEXC,I,J
4           PARAMETER (NCA=5,NRA=64, NUMEXC=1, LDA=NRA, LDX=-
NCA)
     C       NCA IS THE NUMBER OF UNKNOWNS    C
     C       NRA IS THE NUMBER OF CALIBRATION POINTS     C
     C       NUMEXE IS THE COLUMN NUMBER OF B MATRIX      C


6            REAL   X(LDX, NUMEXC)

8            REAL    A(LDA, NCA+NUMEXC)
9            SAVE    A

11           EXTERNAL    LQRRV, SGEMM, WRRRN

12           OPEN(UNIT=1,FILE='R.DAT',STATUS='OLD')

13           DO 15 I=0,LDA
             DO 15 J=1,NCA+NUMEXC

14             READ(1,100,END=16) A(I,J)
100          FORMAT(F)
15           CONTINUE
```

```
            CLOSE(UNIT=1)

16            CALL LQRRV (NRA, NCA, NUMEXC, A, LDA, X, LDX)
110           CALL WRRRN (' SOLUTIONS 1-2 ', NCA,NUMEXC,
     &                     X, LDX, 0)

130   CALL SGEMM('N','N',NRA,NUMEXC, NCA,
     &           1.E0,A,LDA,X, LDX, -1.E0, A(1, NCA+1), LDA)
170           CALL WRRRN (' RESIDUALS 1-2', NRA, NUMEXC,
     &                     A(1, NCA+1), LDA, 0)

     C   USE THE LIBRARY FUNCTIONS    C

200           END




C   DATA INPUT IN THE ORDER OF COLUMNS, THAT IS 1ST,2ND,...-
COLS. C
C    EG. A=(1 2 3 | 4 5)   C
C           (6 7 8 | 9 10)   C
C           (11 12 13 | 14 15)   C
C           (16 17 18 | 19 20)   C
C           (21 22 23 | 24 25)   C
C DATA A/1,6,11,16,21,2,7,12,17,22,3,8,13,18,23,4,9, C
C   &     14,19,24,5,10,15,20,25/                     C
```

```
                 PROGRAM LESQ1
     C    LEAST SQUARE METHOD SOLVED FOR THE APPROXIMATION
            OF THE TOW UNKNOWNS f AND Tz C


            INTEGER LDA,LDX,NCA,NRA,NUMEXC,I,J
            PARAMETER (NCA=2,NRA=64, NUMEXC=1, LDA=NRA, LDX=-
NCA)

     C      NCA IS THE NUMBER OF UNKNOWNS    C
     C      NRA IS THE NUMBER OF CALIBRATION POINTS     C
     C      NUMEXE IS THE COLUMN NUMBER OF B MATRIX      C


            REAL   X(LDX, NUMEXC)
            REAL    B(LDA, NCA+NUMEXC)
            REAL    A(LDA, NCA+NUMEXC)
            SAVE    A



            OPEN(UNIT=3,FILE='F.DAT',STATUS='OLD')


            DO 15 I=1,LDA
            DO 15 J=1,NCA+NUMEXC
            WRITE(*,*) 'OK4'

            READ(3,90,END=66)A(I,J)

            WRITE(*,*) 'OK5'

90          FORMAT(F)
15          CONTINUE
```

```
66          WRITE(*,*) 'OK6'



100           CALL LQRRV (NRA, NCA, NUMEXC, A, LDA, X, LDX)
110           CALL WRRRN (' SOLUTIONS 1-2 ', NCA,NUMEXC,
     &                   X, LDX, 0)

130   CALL SGEMM('N','N',NRA,NUMEXC, NCA,
     &          1.E0,A,LDA,X, LDX, -1.E0, A(1, NCA+1), LDA)
170           CALL WRRRN (' RESIDUALS 1-2', NRA, NUMEXC,
     &                   A(1, NCA+1), LDA, 0)

     C   USE THE LIBRARY FUNCTIONS    C


200           END
```

```
  /* program 1    computing the distorted computer image
coordinate $x_d$,$y_d$   */

#include<math.h>
#include<stdio.h>
#define size1 5
#define size2 5
#define siz 50
#define   c   0.001

 float a1[size1], a2[size1], a3[size1], a4[size1], a5[size1],
b[size1];
 float A1[size2], A2[size2], B[size2];
 int i;

 main()

{
 float xd[size1], xf[size1], yf[size1], yd[size1];
 float sx, dx_1, cx, cy, dy;

 float Sr, r1_1, r2_1, r4_1, r5_1;
 float Ty_1r1, Ty_1r2, Ty_1Tx, Ty_1r4, Ty_1r5;
 float Ty,Ty2,Tx,xi,yi, yii, wii;
 float s,r1,r2, r3,r4,r5, r6, r7, r8, r9,sign;

 float g1[siz], g2[siz], g3[siz], F[siz],r22[siz], f[siz], Tz
[siz],k1[siz];
 float K;
 float tmp1,tmp2,tmp3;
 float xwi, ywi,zwi, Xdi, Ydi,/* an object point i to compute
the sign of Ty */ Xfi,Yfi;
 float xw[size2], yw[size2], zw[size2], x[size2], y[size2],-
```

```
w[size2];



      dx_1= 0.0005075;
      dy= 0.00067667;

      for(i=0; i<size2; i++) {
        zw[i] = 0.0;
      }
      printf( "%s\n", " input sx cx cy" );

      scanf("%f %f %f", &sx, &cx,&cy);

      /*  input sx, cx, cy from the keyboard  */

       for ( i=0; i<size1; ++i ) {

         printf( "xf[%d%s\n yf[%d%s\n",i,"]=?", i, "]=?" );

         scanf( "%f %f",&xf[i],&yf[i] );

      /* input the computer image coordinates (xf[i],yf[i]) */

         printf( "xw[%d%s\n yw[%d%s\n",i,"]=?", i, "]=?" );

         scanf( "%f %f",&xw[i],&yw[i] );

       /* input the world coordinates (xw[i],yw[i]) */


       }

         for (i=0; i<size1; ++i) {
          xd[i]=( 1/sx)*dx_1*(xf[i]-cx);
          yd[i]=dy*(yf[i]-cy);

          a1[i]= yd[i]*xw[i];
          a2[i]= yd[i]*yw[i];
          a3[i]= yd[i];
          a4[i]= - xd[i]*xw[i];
          a5[i]= -xd[i]*yw[i];

       /* compute the A matrix  */

          b[i]= xd[i];

       /* compute the B matrix  */


         }

      for (i=0; i<size1; i++) {

         printf("xd[%d%s%f              yd[%d%s%f\n",
i,"]=",xd[i],
                 i,"]=",yd[i]);

        }

        for (i=0; i<size1; i++) {

         printf("a1[%d%s%f   a2[%d%s%f   a3[%d%s%f
a4[%d%s%f\n", i,"]=",a1[i], i,"]=",a2[i], i,"]=",a3[i],
i,"]=",a4[i]);

         }
         for (i=0; i<size1; i++) {

         printf("a5[%d%s%f   b[%d%s%f\n", i,"]=",a5[i],
i,"]=",b[i]);

         }

     sub1();     /*subroutine to store the data*/

          /*  program 2 computing  Ty   */


       printf("\n");
       printf("%s\n", "enter: r1_1 r2_1 Ty_1Tx r4_1 r5_1");
      scanf( "%f %f %f %f %f", &r1_1, &r2_1, &Ty_1Tx, &r4_1,
&r5_1);

          /* input data from the results of lesq0.for */

       Sr=r1_1*r1_1+ r2_1*r2_1 + r4_1*r4_1 + r5_1*r5_1;

     Ty2=(Sr-sqrt(Sr*Sr-4*(r1_1*r5_1-r4_1*r2_1)*(r1_1*r5_1 -
             r4_1*r2_1)))/
           (2*(r1_1*r5_1-r4_1*r2_1)*(r1_1*r5_1 - r4_1*r2_1));

        Ty=sqrt(Ty2);

         printf("Ty=%f\nSr=%f\n", Ty,Sr);



     /*   program3 computing the Sign of Ty    */
```

```
Ty_1r1= r1_1;
Ty_1r2= r2_1;
Ty_1r4= r4_1;
Ty_1r5= r5_1;


 printf ("%s\n", "input xwi ywi Xfi Yfi");

 scanf ("%f %f %f %f", &xwi , &ywi , &Xfi, &Yfi);

 /* select a point to compute the sign of Ty  */

 r1=Ty_1r1*Ty;
 r2=Ty_1r2*Ty;
 r4=Ty_1r4*Ty;
 r5=Ty_1r5*Ty;
  Tx=Ty_1Tx*Ty;
  xi =r1*xwi+r2*ywi+Tx;
  yi =r4*xwi+r5*ywi+Ty;

  Xdi=Xfi-cx;
  Ydi=Yfi-cy;

 if( xi&&Xdi > 0 && yi&&Ydi > 0 )

    Ty=Ty;

 else Ty= -Ty;

   printf("Ty=%f\n", Ty);



/*     program 4:  computing  r1 ------- r9,   Tx     */
```

```
    s=r1*r4 + r2*r5;

    if (s<0)
     sign= 1;
    else sign = -1;

    r3= - sqrt( 1-r1*r1-r2*r2);
    r6= - sign*(sqrt(1-r4*r4-r5*r5));
    r7= - sqrt(1-r1*r1-r4*r4);
    r8= - sqrt(1-r2*r2-r5*r5);
    r9= sqrt( -1+Sr);

  /*  the sign of r3, r6, r7, r8 are determined by that
       the result of the focal length f is positive */

printf( "r1=%f            r2=%f\n", r1,r2);
printf( "r3=%f            r4=%f\n", r3,r4);
printf( "r5=%f            r6=%f\n", r5,r6);
printf( "r7=%f            r8=%f\n", r7,r8);
printf( "r9=%f             Tx=%f\n", r9,Tx);




     /*    program 5:  computing     y[i]  & w[i]    */



   for( i=0; i<size2; i++ )  {
          y[i]=r4*xw[i] + r5*yw[i] + Ty;
          w[i]= r7*xw[i]+r8*yw[i];

          A1[i]= y[i];
          A2[i]= - dy*(yf[i]-cy);

        /* compute the A matrix  */

          B[i] = w[i]*dy*(yf[i]-cy);

        /* compute the B matrix  */



        }

        for( i=0; i<size2; i++)  {

          printf("w[%d%s%f              y[%d%s%f\n", i,"]=",
w[i],i,"]=", y[i]);
        }



         for( i=0; i<size2; i++) {

           printf("A1[%d%s%f    A2[%d%s%f    B[%d%s%-
f\n",i,"]=",A1[i], i, "]=", A2[i], i, "]=", B[i]);


          }

       sub2(); /* subroutine to store these data */


/*    STEEPEST DECENT PROGRAM to compute f Tz k1    */
```

```
 printf("\n");
 i=0;
 printf("Enter: f(0), Tz(0), k1(0)\n");
 scanf("%f%f%f",&f[i],&Tz[i],&k1[i]);

    /* input the initial value from the results
         computed before */

r22[i]=(dx_1*xf[i]/sx)*(dx_1*xf[i]/sx)+dy*yf[i]*dy*yf[i];

 tmp1=(dy*yf[i] + dy*yf[i]*k1[i]*r22[i] - f[i]*(r4*xw[i] +
r5*yw[i] + r6*zw[i] + Ty)/
           (r7*xw[i] + r8*yw[i] + r9*zw[i] + Tz[i]));

 tmp2=f[i]*(r4*xw[i]+r5*yw[i]+r6*zw[i]+Ty)/(r7*xw[i]+r8*y-
w[i]+r9*zw[i]+Tz[i]);

 tmp3=(r7*xw[i] + r8*yw[i] +r9*zw[i] + Tz[i]);

 F[i]= tmp1*tmp1;


      g1[i]= 2*tmp1*(-tmp2/f[i]);
                                         /*   dF/df   */

      g2[i]=2*tmp1*tmp2/tmp3;
                                         /*   dF/dTz   */

       g3[i]= 2*tmp1*dy*yf[i]*r22[i];   /*  dF/dk1   */

       K=1;
       for ( i=0; i<=siz; ) {
         f[i+1]=f[i]-K*g1[i];
         Tz[i+1]=Tz[i]-K*g2[i];
          k1[i+1]=k1[i]-K*g3[i];

           i++;

r22[i]=(dx_1*xf[i]/sx)*(dx_1*xf[i]/sx)+dy*yf[i]*dy*yf[i];
 tmp1=(dy*yf[i] + dy*yf[i]*k1[i]*r22[i] - f[i]*(r4*xw[i] +
r5*yw[i] + r6*zw[i] + Ty)/
          (r7*xw[i] + r8*yw[i] + r9*zw[i] + Tz[i]));

           F[i]=tmp1*tmp1;
           i--;

           if(F[i+1]<= c )
            break;
           else if ( F[i+1] <= F[i] )
                 i= i+1;
                else K=K/2;

          }

     printf("i= %d\n", i);
     printf("f(i)= %f\n", f[i]);
     printf("Tz(i)= %f\n",Tz[i]);
     printf("k1(i)= %f\n", k1[i]);

 }


    sub1()
   /*  store the data for computing the five unknowns
       $T_y^{-1}r_1$, $T_y^{-1}r_2$, $T_y^{-1}T_x$, $T_y^{-1}r_4$, $T_y^{-1}r_5$   */
    {
```

```
 FILE *fp;

 float rr_data[size1][6];
 int m,n;

if (( fp=fopen("rdata","w")) == NULL)  {
   printf("Error in  opening file %10s \n","rdata ");
   exit();
 }
else {

  for(m=0; m<size1; m++) {
       n=0;
   rr_data[m][n]= a1[m];
       n=1;
   rr_data[m][n]=a2[m];
       n=2;
   rr_data[m][n]=a3[m];
       n=3;
   rr_data[m][n]=a4[m];
       n=4;
   rr_data[m][n]=a5[m];
       n=5;
   rr_data[m][n]=b[m];

   }
    for(m=0; m<size1; m++)
    for(n=0; n<6;    n++)

   fprintf(fp,"%f\n",rr_data[m][n]);

    for( m=0; m < size1; m++)
     for( n=0; n< 6; n++)
     printf("%f       \n",rr_data[m][n]);
    }
  fclose(fp);
  }




     sub2()

/*  store the data for computing the approximation value of
      f and Tz    */
     {
     FILE *fpp;
     float r_data[size2][3];
     int p,q;

     if (( fpp=fopen("fdata","w")) == NULL)  {
      printf("error in opening file %10s \n", "fdata");
      exit();
     }
      else {

       for (p=0;p<size2; p++)  {
         q=0;
         r_data[p][q] = A1[p];
         q=1;
         r_data[p][q] = A2[p];
         q=2;
         r_data[p][q] = B[p];
        }

        for (p=0; p<size2; p++)
        for (q=0; q<3;     q++)
```

```
   fprintf(fpp,"%f\n", r_data[p][q]);


   for (p=0;p<size2;p++)
   for (q=0;q<3;    q++)
     printf("%f    \n",r_data[p][q]);
 }
   fclose(fpp);
}
```

REFERENCES

[1] Roger Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses", IEEE Journal of Robotics and Automation. vol. RA-3. NO. 4. August 1989.

[2] A. Okamoto, "The model construction problem using collinearity condition", Photogrammetric Eng. Remote Science, vol. L, 1984.

[3] W. Faig, "Calibration of close-range photogrammetry systems: mathematical formulation", Photogrammetric Eng. Remote Sensing, vol. 41, 1985.

[4] Y.I. Abdel-Aziz, "Direct linear transformation into object space coordinates in close-range photogrammetry", in Proc. Symp. Close-Range Photogrammetry. Univ. of Illinois at Urbana Champaign, Urbana, 1971.

[5] H. M. Karara, Ed., "Handbook of non-topographic photogrammetry", Amer. Soc. of Photogrammetry, 1979.

[6] K. W. Wong, "Mathematical formulation and digital analysis in close-range photogrammetry", Photogrammetric Eng. Remote Sensing, vol. 41, 1975.

[7] I. Sobel, "On calibrating computer controlled cameras for perceiving 3D scenes", Artificial Intell., vol. 5, 1974.

[8] D. B. Gennery, " Stereo-camera calibration", in Proc. Image Understanding Workshop, 1979.

[9] R. R. Cohen and E. A. Feigenbaum, The Handbook of Artifical Intelligence, vol. III. Heuris Tech. Press, 1982.

[10] E. L. Hall, M. B. K. Tiio, C. A. McPherson, and F. A. Sadjadi, "Curved surface measurement and recognition for robot vision", Conf. Rec. IEEE Workshop on Industrial Applications of Machine Vision, May 1982.

[11] I. Sutherland, "Three-dimensional data input by tablet", Proc. IEEE, vol. 62, 1974.

[12] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting applications to image analysis and automated cartography", Proc. Image University Workshop. April, 1980.

[13] A. Isaguirre, P. Pu, and J. S.Ummers, "A new development in camera calibration: calibrating a pair of mobile cameras", Proc. Inc. Conf. Robotics and Automation. 1985.

[14] Lenz, Reimar K., Roger Tsai, "Calibrating a cartesian robot with eye-on-hand configuration independent of eye-to-hand relationship", Proceedings of Computer Vision and Pattern Recognition, Ann Arbor, MI, June5-9, 1988.

[15] Roger Tsai, "Synopsis of recent progress on camera calibration for 3D machine vision", Robotics Review, MIT Press December, 1988.

[16] Roger Tsai, "A new technique for autonomous efficient 3D robotics hand/eye calibration", IBM Research Report RC 13212 (#59370), 10/20/87.

[17] Roger Tsai, "Overview of a unified calibration trio for robot eye, eye-to-hand, and hand calibration using 3D machine vision", SPIE Advance in Intelligent Robotics Systems, Cambridge, MA., November 7-11, 1988.