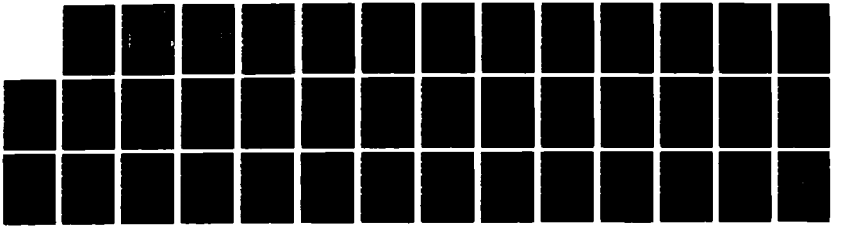
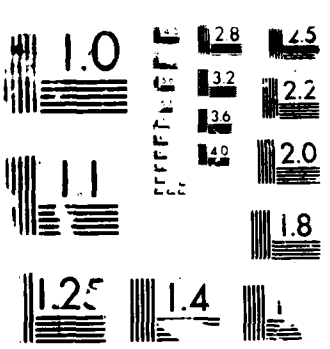
AD-A189 571 CAMERA CALIBRATION(U) ROCHESTER UNIV NY DEPT OF 1/1
COMPUTER SCIENCE I RIGOUTSOS ET AL MAR 86 TR-186
N00014-82-K-0193

UNCLASSIFIED F/G 14/4 NL





Camera Calibration

Isidore Rigoutsos and Christopher M. Brown
Department of Computer Science
The University of Rochester
Rochester, NY 14627

TR 186
March 1986

Department of Computer Science
University of Rochester
Rochester, New York 14627

Camera Calibration

Isidore Rigoutsos and Christopher M. Brown
Department of Computer Science
The University of Rochester
Rochester, NY 14627

TR 186
March 1986

Abstract

Given an image that has been scaled both horizontally and vertically (possibly with different scale factors in the two directions), we determine the camera position and orientation, as well as the scale factors for sampling in the u and v axes of the image plane, from the knowledge of the correspondence between a few known points in space and their location on the image (image plane).

This work was supported in part by the Office of Naval Research and the Defense Advanced Research Projects Agency under Contract N00014-82-K-0193, the National Science Foundation under Grant DCR-8320136, and the U.S. Army Engineering Topographic Laboratories under Contract DACA76-85-C-0001. We thank the Xerox Corporation University Grants Program for providing the equipment used in the preparation of this paper.

AD-A189 571

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: TR 186	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): Camera Calibration	5. TYPE OF REPORT & PERIOD COVERED: Technical Report
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Isidore Rigoutsos and Christopher M. Brown	8. CONTRACT OR GRANT NUMBER(s): N00014-82-K-0193, DACA76-85-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Computer Science Department, University of Rochester, Rochester, NY 14627	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Defense Advanced Research Project Agency, 1400 Wilson Blvd., Arlington, VA 22209	12. REPORT DATE: March 1986
	13. NUMBER OF PAGES: 36
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Office of Naval Research, Information Systems, Arlington, VA 22217	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Distribution of this document is unlimited

Accession For
NTIS GRA&I ☑
DTIC TAB ☐
Unannounced ☐
Justification
By
Distribution/
Availability
Dist
A-1

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES
None

DTIC COPY INSPECTED 6

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Computer imaging, camera calibration, scale, imaging process, robotics

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
Given an image that has been scaled both horizontally and vertically (possibly with different scale factors in the two directions), we determine the camera position and orientation, as well as the scale factors for sampling in the u and v axes of the image plane, from the knowledge of the correspondence between a few known points in space and their location on the image (image plane).

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Abstract

Given an image that has been scaled both horizontally and vertically (possibly with different scale factors in the two directions), determine the camera position and orientation, as well as the scale factors for sampling in the u and v axes of the image plane, from the knowledge of the correspondences between a few known points in space and their locations on the image (image plane).

Introduction

For many practical purposes, it is sufficient to model the imaging process of a camera by a pure point projection process; then the two-dimensional image plane coordinates of a 3-D point, as seen by a camera, can be easily expressed in terms of a 3 x 4 matrix using the notion of the homogeneous coordinate system[2,5]. Because of its elegance and its usual accuracy, this matrix has considerable applications in fields like computer graphics, computer vision, motion tracking, etc. The problem of camera calibration and determination of the above mentioned matrix has received considerable attention in the literature[3,4,5,6]. Camera calibration is simply determining the elements of the matrix that describes the imaging process. Equivalently, it can be thought of as determining certain geometrical and optical constants, such as the direction of the optical axis, the spatial location of the camera's nodal point and the focal length of the lens. The solution that is described in what follows, is the one suggested by Ganapathy ([5]).

1. Formulation of the problem.

Consider a right handed three-dimensional orthonormal coordinate system OXYZ (see Fig. 1). Using the homogeneous coordinate system ([2]) the coordinates of a point A(x, y, z) are represented by a quadruple of the form (ax, ay, az, a) where we introduced the scalar a . In order to obtain the three-dimensional coordinates of a given point that is represented in the homogemeous coordinate system, one merely divides by the last component of the quadruple. In a similar way, a (two-dimensional) point I (u, v) of the image plane (U-V) is represented by the triple (iu, iv, i), for some arbitrary scalar quantity i, and we divide by the third component i to obtain u and v.

Consider a camera having its center at C that is located at (x_o, y_o, z_o) (see Fig.1); the camera looks along a line of sight (optical axis of the camera) CO'P where P is the point at which the optical axis of the camera intersects the X-Y plane and O' the point at which the optical axis pierces the image plane. Taking a right-handed orthonormal coordinate system O"X'Y'Z' centered at C, we can write for any point of the 3-D space the following equation:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \qquad (1)$$

where

$$R \equiv \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

is the rotation matrix, and

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

is the translation matrix.
In the above, (x, y, z) and (x', y', z') are the coordinates with respect to the OXYZ and O"X'Y'Z' coordinate systems respectively.
Using homogeneous coordinates, the above equation (1) can be written as follows:

$$\begin{bmatrix} jx' \\ jy' \\ jz' \\ j \end{bmatrix} = \begin{bmatrix} a & b & c & p \\ d & e & f & q \\ g & h & i & r \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} lx \\ ly \\ lz \\ l \end{bmatrix} \quad (2)$$

where we combined both the translation and rotation matrices in one.

Since the (x_0, y_0, z_0) point of the OXYZ orthonormal system is mapped to the (0, 0, 0) point of the O'UVW system, then using equation (1) it follows that:

$$p = -a * x_0 - b * y_0 - c * z_0 \quad (3)$$
$$q = -d * x_0 - e * y_0 - f * z_0 \quad (4)$$
$$r = -g * x_0 - h * y_0 - i * z_0 \quad (5)$$

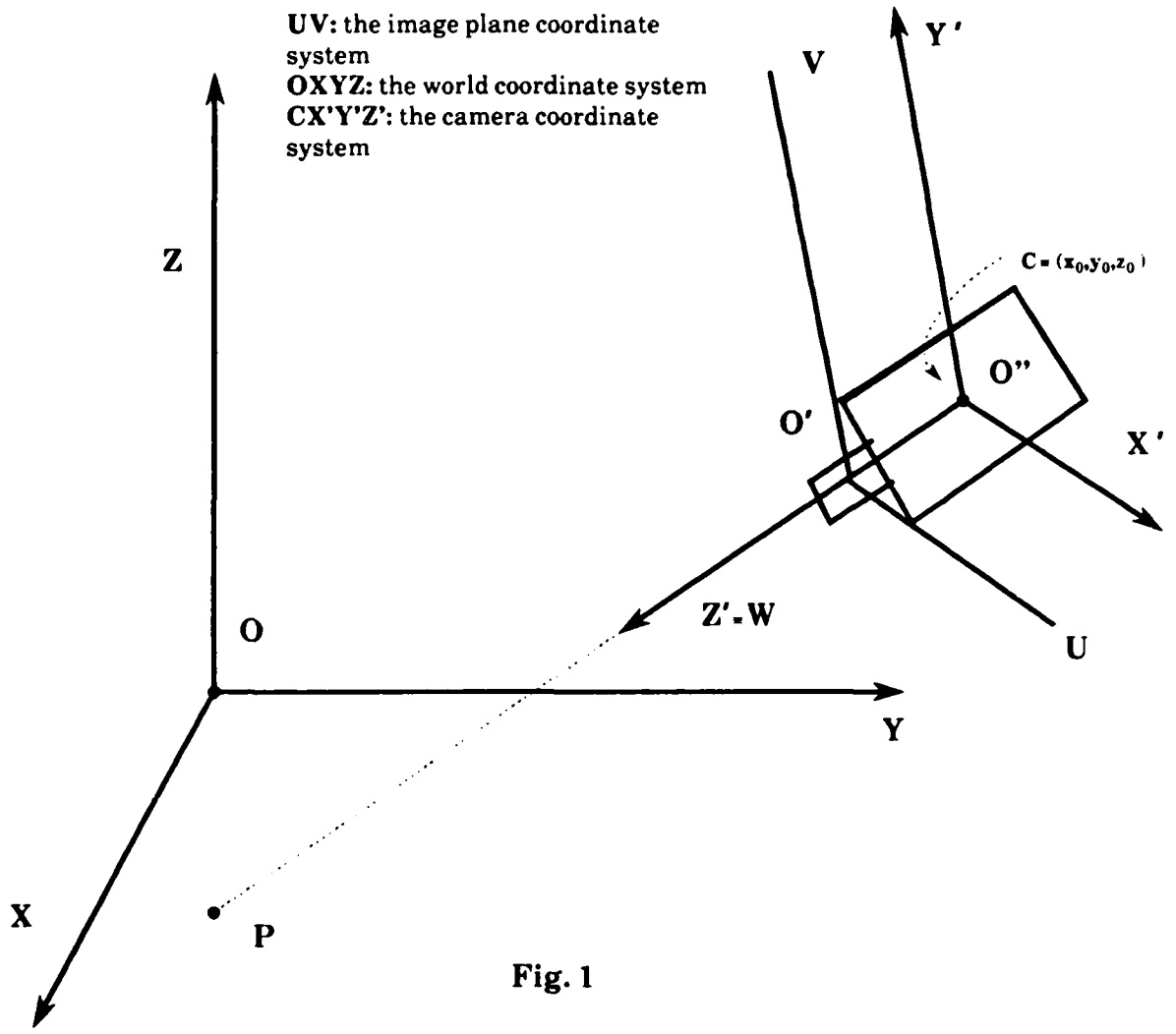


Fig. 1

Substituting into equation (1) (x, y, z) by (x_0, y_0, z_0) and (x', y', z') by (0, 0, 0) and solving for the translation matrix, we have the following equation:

$$\begin{bmatrix} -p \\ -q \\ -r \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \qquad (6)$$

But R, the rotation matrix, is assumed to be an orthonormal rotation matrix and therefore it follows that $R^{-1} = R^t$. Using the latter equation we can write (6) as:

$$\begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix} \begin{bmatrix} -p \\ -q \\ -r \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \qquad (7)$$

This equation can be equivalently written as:

$$x_0 = -p*a \; -q*d \; -r*g \qquad (8)$$
$$y_0 = -p*b \; -q*e \; -r*h \qquad (9)$$
$$z_0 = -p*c \; -q*f \; -r*i \qquad (10)$$

Consider now a point (x', y', z') where the coordinates are taken with respect to the O'X'Y'Z' system. The image plane coordinates of the considered point will therefore be given by the equations:

$$u = \frac{fl * x'}{z'} \quad \text{and} \quad v = \frac{fl * y'}{z'}$$

Here we assumed that the Z' axis is parallel to the optical axis and that the image plane is in front of the camera; fl is the focal length of the camera. Furthemore, X',Y', and Z' axes can be aligned with the U, V, and W axes such that U-axis is parallel to the X'-axis, V-axis is parallel to the Y'-axis, and W-axis is parallel to the Z'-axis. In practice though, the measurements in both the X' and Y' axes are subject to finite resolution; this consequently induces a discrete spectrum of values for the measured quantities. To be general we will assume two scale factors (possibly different) k_u and k_v for sampling in the X' and Y' axes respectively. Moreover, if (u_0,

v_0, 0) are the coordinates, with respect to the UVW coordinate system, of the point that we consider to be the one through which the optical axis passes, then the above equations can be written as:

$$u = u_0 + \frac{k_u * fl * x'}{z'} \quad (11) \quad \text{and} \quad v = v_0 + \frac{k_v * fl * y'}{z'} \quad (12)$$

Representing now the products $k_u * fl$ and $k_v * fl$ by k_1 and k_2 respectively, we can clearly write the above equations, using homogeneous coordinates, as follows:

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \begin{bmatrix} k_1 & 0 & u_0 & 0 \\ 0 & k_2 & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} sx' \\ sy' \\ sz' \\ s \end{bmatrix} \quad (13)$$

or

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = M_1 \begin{bmatrix} sx' \\ sy' \\ sz' \\ s \end{bmatrix} \quad (14)$$

Note that M_1 is a 3x3 matrix whose determinant is equal to $k_1 * k_2 \neq 0$. Therefore it is invertible.

Similarly, equation (2) can be written as:

$$\begin{bmatrix} sx' \\ sy' \\ sz' \\ s \end{bmatrix} = M_2 \begin{bmatrix} lx \\ ly \\ lz \\ l \end{bmatrix} \qquad (15)$$

Observe that M_2 is completely determined by the camera location and orientation. Equations (14) and (15) can be written as :

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = M_3 \begin{bmatrix} sx \\ sy \\ sz \\ s \end{bmatrix} \qquad (16)$$

where $M_3 = M_1 \cdot M_2$ is a 3x4 matrix that is known as the **calibration** matrix. Let us denote its components by t_i / i = 1,2,3,4, ... ,12 , where:

$$t_1 = k_1 * a + u_0 * g$$

$$t_2 = k_1 * b + u_0 * h$$

$$t_3 = k_1 * c + u_0 * i$$

$$t_4 = k_1 * p + u_0 * r$$

$$t_5 = k_2 * d + v_0 * g \qquad \text{equations (17)}$$

$$t_6 = k_2 * e + v_0 * h$$

$$t_7 = k_2 * f + v_0 * i$$

$$t_8 = k_2 * q + v_0 * r$$

$$t_9 = g$$

$$t_{10} = h$$

$$t_{11} = i$$

$$t_{12} = r$$

Since we use the homogeneous coordinates formulation, no change will occur if we divide all the t_i's by t_{12}. Using the notation $T_i \equiv t_i / t_{12}$, we have:

$$T_1 = (k_1 * a + u_0 * g)/r$$

$$T_2 = (k_1 * b + u_0 * h)/r$$

$$T_3 = (k_1 * c + u_0 * i)/r$$

$$T_4 = (k_1 * p + u_0 * r)/r$$

$$T_5 = (k_2 * d + v_0 * g)/r \qquad \text{equations (18)}$$

$$T_6 = (k_2 * e + v_0 * h)/r$$

$$T_7 = (k_2 * f + v_0 * i)/r$$

$$T_8 = (k_2 * q + v_0 * r)/r$$

$$T_9 = g/r$$

$$T_{10} = h/r$$

$$T_{11} = i/r$$

$$T_{12} = 1$$

Therefore equation (16) can be expressed in terms of the eleven T_i 's. Each observation (u, v) together with the corresponding (x, y, z) gives us two linear equations in the eleven T_i 's. Hence a minimum of six observations is needed to determine the entries of M_3.

2. Some useful properties.

Consider again matrix R (rotation matrix). Since the latter is an orthonormal rotation matrix the following identities hold:

$$a^2 + b^2 + c^2 = 1 \qquad (19)$$

$$d^2 + e^2 + f^2 = 1 \qquad (20)$$

$$g^2 + h^2 + i^2 = 1 \qquad (21)$$

$$d * g + e * h + f * i = 0 \qquad (22)$$

$$a * d + b * e + c * f = 0 \qquad (23)$$

$$a * g + b * h + c * i = 0 \qquad (24)$$

Equations (19) - (21) represent the normality of the vectors whereas equations (22) - (24) represent the orthogonality of the vectors that correspond to the rows of the rotation matrix ([7]). Recalling that Det (R) = Det (R^{-1}) = 1 and that $R^{-1} = R^t$, we have the following set of equations :

$$a = e * i - f * h \qquad (25)$$

$$b = f * g - d * i \qquad (26)$$

$$c = d * h - e * g \qquad (27)$$

$$d = c * h - b * i \qquad (28)$$

$$e = c * g - a * i \qquad (29)$$

$$f = b * g - a * h \qquad (30)$$

$$g = b * f - e * c \qquad (31)$$

$$h = c * d - a * f \qquad (32)$$

$$i = a * e - b * d \qquad (33)$$

3. Solving the problem

The equations (19) through (33) we described above are **not** all **independent.** As a matter of fact the existing interdependence can be expressed by the following identity:

$$(a*d + b*e + c*f)^2 = (a^2+b^2+c^2)*(d^2+e^2+f^2) - (a*e-b*d)^2 - (c*d-a*f)^2 - (b*f-e*c)^2$$

Consider now the vectors **A**, **B**, and **C** where **A**, **B**, and **C** are defined as follows:

$$\mathbf{A} \equiv (T_1, T_2, T_3) \qquad (34)$$

$$\mathbf{B} \equiv (T_5, T_6, T_7) \qquad (35)$$

$$\mathbf{C} \equiv (T_9, T_{10}, T_{11}) \qquad (36)$$

Using the definitions of the T_i's, and after some easy computations we have that:

$$\mathbf{A} \cdot \mathbf{A} = \frac{k_1^2 + u_o^2}{r^2} \qquad (37)$$

$$\mathbf{B} \cdot \mathbf{B} = \frac{k_2^2 + v_o^2}{r^2} \qquad (38)$$

$$\mathbf{C} \cdot \mathbf{C} = \frac{1}{r^2} \qquad (39)$$

$$\mathbf{A} \cdot \mathbf{C} = \frac{u_o}{r^2} \qquad (40)$$

$$\mathbf{B} \cdot \mathbf{C} = \frac{v_o}{r^2} \qquad (41)$$

The above equations clearly indicate the method that is to be used in order to solve the problem.

3 a) Computing the parameters of the problem

We *first determine* the entries T_i, $i = 1,2,3,4 \ldots 11$ (recall that $T_{12} = 1$). Then using the definitions (34), (35), (36) the equations (37) thru (41) and equations (18), we can clearly see that:

$$r = \frac{1}{\sqrt{(T_9^2 + T_{10}^2 + T_{11}^2)}} \qquad (42)$$

$$u_o = (T_1 * T_9 + T_2 * T_{10} + T_3 * T_{11}) * r^2 \qquad (43)$$

$$v_o = (T_5 * T_9 + T_6 * T_{10} + T_7 * T_{11}) * r^2 \qquad (44)$$

$$k_1 = \sqrt{(r^2 * (T_1 * T_1 + T_2 * T_2 + T_3 * T_3) - u_o^2)} \qquad (45)$$

$$k_2 = \sqrt{(r^2 * (T_5 * T_5 + T_6 * T_6 + T_7 * T_7) - v_o^2)} \qquad (46)$$

$$g = T_9 * r \qquad (47)$$

$$h = T_{10} * r \qquad (48)$$

$$i = T_{11} * r \qquad (49)$$

$$a = \frac{T_1 * r - u_o * g}{k_1} \qquad (50)$$

$$b = \frac{T_2 * r - u_o * h}{k_1} \qquad (51)$$

$$c = \frac{T_3 * r - u_o * i}{k_1} \qquad (52)$$

$$d = \frac{T_5 * r - v_o * g}{k_2} \qquad (53)$$

$$e = \frac{T_6 * r - v_o * h}{k_2} \qquad (54)$$

$$f = \frac{T_7 * r - v_o * i}{k_2} \qquad (55)$$

$$p = \frac{T_4 * r - u_o * r}{k_1} \qquad (56)$$

$$q = \frac{T_8 * r - v_o * r}{k_2} \qquad (57)$$

In the above the sign of r is assumed to be positive. As a matter of fact the solutions with +r and -r correspond to the image in front of and behind the camera respectively (see [5]). Moreover, k_1 can always be assumed positive. This is not true for k_2 though. The sign of k_2 depends on the polarity of axes U and V (see [5]). We can determine the sign of k_2 using the set of equations {(28), (29), (30)} and {(53), (54), (55)}. Observe from the latter set of equations that the signs of d, e and f depend on the sign of k_2; therefore, the former set of equations will hold only for one of the signs of k_2, that is for the correct one.

3 b) How to determine the table entries

Observe that we deal with a total of 11 unknown table entries and therefore, in principle, a set of at least 5 1/2 points is needed. Since, in general, more than 6 points are used, one needs to apply the method of least-squares. Recalling the fact that we are using the homogeneous coordinate representation, we can clearly see that the u and v coordinates are given by the following equations:

$$u_i = \frac{T_1 * x_i + T_2 * y_i + T_3 * z_i + T_4}{T_9 * x_i + T_{10} * y_i + T_{11} * z_i + T_{12}} \quad / i = 1,2,3 \ldots n \qquad (58)$$

$$v_i = \frac{T_5 * x_i + T_6 * y_i + T_7 * z_i + T_8}{T_9 * x_i + T_{10} * y_i + T_{11} * z_i + T_{12}} \quad / i = 1,2,3 \ldots n \qquad (59)$$

where s, w were assumed to be equal to 1 (see equation 16) and n is the number of the experimental points. Therefore, application of the least-squares method to this case will require that we minimize the quantity **A**, where:

$$\mathbf{A} \equiv \sum_{i=1}^{n} [\,(T_9 * x_i + T_{10} * y_i + T_{11} * z_i + 1) * u_i - (T_1 * x_i + T_2 * y_i + T_3 * z_i + T_4)\,]^2 +$$

$$\sum_{i=1}^{n} [\,(T_9 * x_i + T_{10} * y_i + T_{11} * z_i + 1) * v_i - (T_5 * x_i + T_6 * y_i + T_7 * z_i + T_8)\,]^2$$

Taking the partial derivatives of **A** with respect to T_i, for $i = 1, 2, 3, \ldots 11$, and equating them to 0, we come up with the following set of linear equations :

$$T_1*(\sum_{i=1}^{n} x_i^2*u_i)+T_2*(\sum_{i=1}^{n} x_i*y_i*u_i)+T_3*(\sum_{i=1}^{n} x_i*z_i*u_i)+T_4*(\sum_{i=1}^{n} x_i*u_i)+$$

$$+T_5*(\sum_{i=1}^{n} x_i^2*v_i)+T_6*(\sum_{i=1}^{n} x_i*y_i*v_i)+T_7*(\sum_{i=1}^{n} x_i*z_i*v_i)+T_8*(\sum_{i=1}^{n} x_i*v_i)-$$

$$-T_9*(\sum_{i=1}^{n} x_i^2*(u_i^2+v_i^2))-T_{10}*(\sum_{i=1}^{n} x_i*y_i*(u_i^2+v_i^2))-T_{11}*(\sum_{i=1}^{n} x_i*z_i*(u_i^2+v_i^2))=\sum_{i=1}^{n} x_i*(u_i^2+v_i^2) \quad \textbf{(E1)}$$

$$T_1*(\sum_{i=1}^{n} x_i*y_i*u_i)+T_2*(\sum_{i=1}^{n} y_i^2*u_i)+T_3*(\sum_{i=1}^{n} y_i*z_i*u_i)+T_4*(\sum_{i=1}^{n} y_i*u_i)+$$

$$+T_5*(\sum_{i=1}^{n} x_i*y_i*v_i)+T_6*(\sum_{i=1}^{n} y_i^2*v_i)+T_7*(\sum_{i=1}^{n} y_i*z_i*v_i)+T_8*(\sum_{i=1}^{n} y_i*v_i)-$$

$$-T_9*(\sum_{i=1}^{n} x_i*y_i*(u_i^2+v_i^2))-T_{10}*(\sum_{i=1}^{n} y_i^2*(u_i^2+v_i^2))-T_{11}*(\sum_{i=1}^{n} y_i*z_i*(u_i^2+v_i^2))=\sum_{i=1}^{n} y_i*(u_i^2+v_i^2) \quad \textbf{(E2)}$$

$$T_1*(\sum_{i=1}^{n} x_i*z_i*u_i)+T_2*(\sum_{i=1}^{n} y_i*z_i*u_i)+T_3*(\sum_{i=1}^{n} z_i^2*u_i)+T_4*(\sum_{i=1}^{n} z_i*u_i)+$$

$$+T_5*(\sum_{i=1}^{n} x_i*z_i*v_i)+T_6*(\sum_{i=1}^{n} y_i*z_i*v_i)+T_7*(\sum_{i=1}^{n} z_i^2*v_i)+T_8*(\sum_{i=1}^{n} z_i*v_i)-$$

$$-T_9*(\sum_{i=1}^{n} x_i*z_i*(u_i^2+v_i^2))-T_{10}*(\sum_{i=1}^{n} y_i*z_i*(u_i^2+v_i^2))-T_{11}*(\sum_{i=1}^{n} z_i^2*(u_i^2+v_i^2))=\sum_{i=1}^{n} z_i*(u_i^2+v_i^2) \quad \textbf{(E3)}$$

$$T_1*(\sum_{i=1}^{n} x_i^2)+T_2*(\sum_{i=1}^{n} x_i*y_i)+T_3*(\sum_{i=1}^{n} x_i*z_i)+T_4*(\sum_{i=1}^{n} x_i)-$$

$$-T_9*(\sum_{i=1}^{n} x_i^2*u_i)-T_{10}*(\sum_{i=1}^{n} x_i*y_i*u_i)-T_{11}*(\sum_{i=1}^{n} x_i*z_i*u_i)=\sum_{i=1}^{n}(x_i*u_i) \quad \text{(E4)}$$

$$T_1*(\sum_{i=1}^{n} x_i*y_i)+T_2*(\sum_{i=1}^{n} y_i^2)+T_3*(\sum_{i=1}^{n} y_i*z_i)+T_4*(\sum_{i=1}^{n} y_i)-$$

$$-T_9*(\sum_{i=1}^{n} x_i*y_i*u_i)+T_{10}*(\sum_{i=1}^{n} y_i^2*u_i)+T_{11}*(\sum_{i=1}^{n} y_i*z_i*u_i)=\sum_{i=1}^{n}(y_i*u_i) \quad \text{(E5)}$$

$$T_1*(\sum_{i=1}^{n} x_i*z_i)+T_2*(\sum_{i=1}^{n} y_i*z_i)+T_3*(\sum_{i=1}^{n} z_i^2)+T_4*(\sum_{i=1}^{n} z_i)-$$

$$-T_9*(\sum_{i=1}^{n} x_i*z_i*u_i)-T_{10}*(\sum_{i=1}^{n} y_i*z_i*u_i)-T_{12}*(\sum_{i=1}^{n} z_i^2*u_i)=\sum_{i=1}^{n}(z_i*u_i) \quad \text{(E6)}$$

$$T_1*(\sum_{i=1}^{n} x_i)+T_2*(\sum_{i=1}^{n} y_i)+T_3*(\sum_{i=1}^{n} z_i)+T_4*n-$$

$$-T_9*(\sum_{i=1}^{n} x_i*u_i)-T_{10}*(\sum_{i=1}^{n} y_i*u_i)-T_{11}*(\sum_{i=1}^{n} z_i*u_i)=\sum_{i=1}^{n} u_i \quad \text{(E7)}$$

$$T_5*(\sum_{i=1}^{n} x_i^2)+T_6*(\sum_{i=1}^{n} x_i*y_i)+T_7*(\sum_{i=1}^{n} x_i*z_i)+T_8*(\sum_{i=1}^{n} x_i)-$$

$$-T_9*(\sum_{i=1}^{n} x_i^2*v_i)-T_{10}*(\sum_{i=1}^{n} x_i*y_i*v_i)-T_{11}*(\sum_{i=1}^{n} x_i*z_i*v_i)=\sum_{i=1}^{n}(x_i*v_i) \quad \text{(E8)}$$

$$T_5*(\sum_{i=1}^{n} x_i*y_i)+T_6*(\sum_{i=1}^{n} y_i^2)+T_7*(\sum_{i=1}^{n} y_i*z_i)+T_8*(\sum_{i=1}^{n} y_i)-$$

$$-T_9*(\sum_{i=1}^{n} x_i*y_i*v_i)+T_{10}*(\sum_{i=1}^{n} y_i^2*v_i)+T_{11}*(\sum_{i=1}^{n} y_i*z_i*v_i)=\sum_{i=1}^{n}(y_i*v_i) \quad \text{(E9)}$$

$$T_5*(\sum_{i=1}^{n} x_i*z_i)+T_6*(\sum_{i=1}^{n} y_i*z_i)+T_7*(\sum_{i=1}^{n} z_i^2)+T_8*(\sum_{i=1}^{n} z_i)-$$

$$-T_9*(\sum_{i=1}^{n} x_i*z_i*v_i)-T_{10}*(\sum_{i=1}^{n} y_i*z_i*v_i)-T_{12}*(\sum_{i=1}^{n} z_i^2*v_i)=\sum_{i=1}^{n}(z_i*v_i) \quad \text{(E10)}$$

$$T_5*(\sum_{i=1}^{n} x_i)+T_6*(\sum_{i=1}^{n} y_i)+T_7*(\sum_{i=1}^{n} z_i)+T_8*n-$$

$$-T_9*(\sum_{i=1}^{n} x_i*v_i)-T_{10}*(\sum_{i=1}^{n} y_i*v_i)-T_{11}*(\sum_{i=1}^{n} z_i*v_i)=\sum_{i=1}^{n} v_i \quad \text{(E11)}$$

4. Estimating the "quality" of the results

We would like to lump somehow the non-linear nature (cf. equations (17)) of the original problem in a single parameter. At this point one should also recall the error

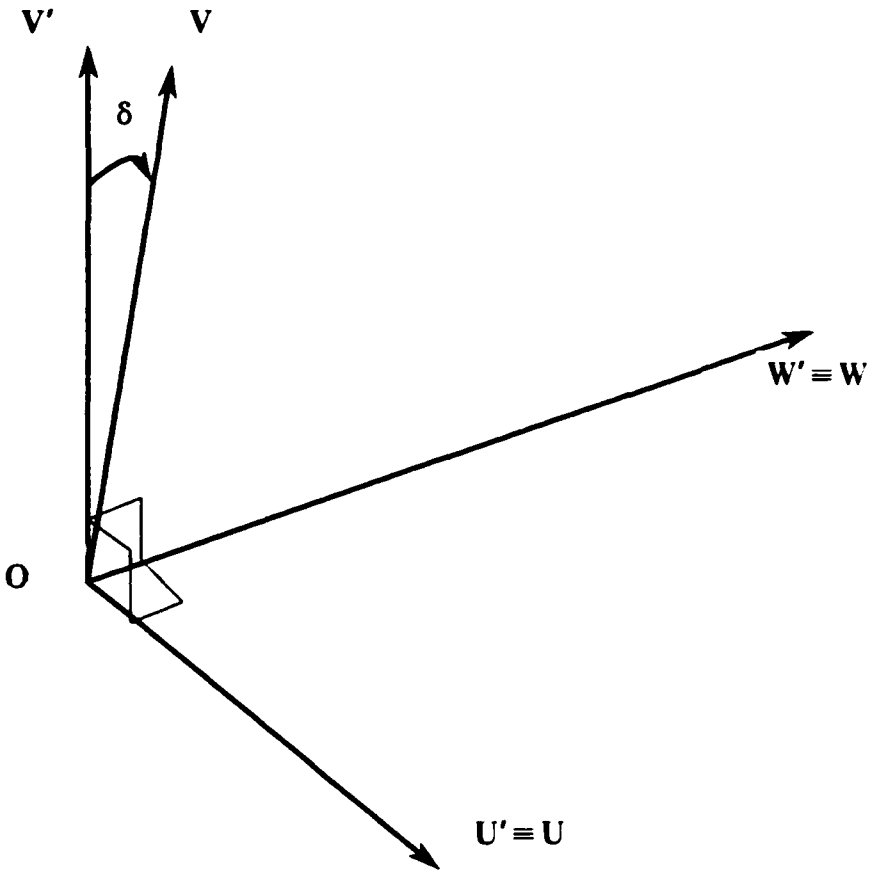


Fig. 2

introduced by the observations (i.e. (u_i, v_i) & (x_i, y_i, z_i)) made as well as the fact that the camera used is not a pin hole one without distortions. Assume that all these errors that are intrinsic to the experimental approach amount to the determination, using equations (47) - (54), to a coordinate system UVW such that axes U and V are not perpendicular to each other but slightly off by a small angle δ (see the above Fig. 2). We furthermore considered axes U'V'W', that constitute an orthonormal

coordinate system, such that the pairs (W , W') and (V, V') of axes represent axes parallel to each other. It is clear that the coordinates u,v,w can still be expressed in terms of x,y,z (see also Fig. 1) as follows:

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} a' & b' & c' \\ d' & e' & f' \\ g' & h' & i' \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad (71)$$

where a', b', c', d', e', f', g', h', i' are the coefficients of a proper rotation matrix. Moreover we observe that we can write:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ \sin\delta & \cos\delta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} \qquad (72)$$

Hence we can write the following:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} a' & b' & c' \\ a'*\sin\delta + d'*\cos\delta & b'*\sin\delta + e'*\cos\delta & c'*\sin\delta + f'*\cos\delta \\ g' & h' & i' \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad (73)$$

Comparing now equations (72) and (73) we observe that we can write:

$$a = a'$$

$$b = b'$$

$$c = c'$$

$$d = a'*\sin\delta + d'*\cos\delta$$

$$e = b'*\sin\delta + e'*\cos\delta$$

$$f = c'*\sin\delta + f'*\cos\delta$$

$$g = g'$$

$$h = h'$$

$$i = i'$$

Therefore,

$$a*d + b*e + c*f = \sin\delta*(a'^2 + b'^2 + c'^2) + \cos\delta*(a'*d' + b'*e' + c'*f') = \sin\delta$$

Recall that from equation (23) – "normality" equation – we have :

$$a*d + b*e + c*f = 0$$

Hence, we can estimate the " goodness " of the calibration using the value of parameter δ. It has been observed, that the closer δ's value is to 0 the better the calibration under consideration. In fact, values of $\delta \sim 10^{-3}$ radians are considered satisfactory for solid state as well as vidicon cameras. In case δ is significantly greater than 10^{-3}, it is better to consider a new set of points than to attempt to recalibrate the camera with the same set of points.

5. Deriving more important parameters of the problem

So far we have solved the linear system that resulted after the consideration of the initial equations of the problem. In other words we have computed the entries of the matrix M_3, or equivalently the unknowns T_i's / $i=1,2,3 \ldots 12$, (recall that $T_{12}=1$). We still need to translate these coefficients into camera parameters, namely the position of the center of the camera in 3-D, the scale factors k_u and k_v as well as the pair (u_0, v_0) (see page 6).

The u_0 and v_0 are computed using equations (44) and (45). Recalling the definitions of k_u and k_v we observe that using the results of equations (45) and (46) we can compute k_u and k_v as follows:

$$k_u = k_1 / fl$$
$$k_v = k_2 / fl$$

where fl is the (known) focal length of the camera.

Furthermore, in what concerns the position of the camera in 3-D, one need only recall equations (8), (9), and (10). These equations directly give us the 3-D coordinates of the center of the camera.

At this point it is worth recalling the discussion about the sign of k_2. As we saw, according to the set of points considered, the sign of k_2 might be different each time. And since the values of d, e, f, and q all depend on the values of k_2 it seems that we must determine each time the sign of k_2, in order to compute x_0, y_0, z_0. However we can observe that all these variables appear in equations (8) , (9) , (10) , as products of pairs and therefore their signs always reduce to " + ". Hence there is no need at all to compute the sign of k_2.

Finally, we could also determine the directions of the axes U, V, and W with respect to the fixed 3-D coordinate sytem OXYZ. In order to do this we only need to consider the (orthonormal) vectors:

$$n_1 = (a, b, c)$$
$$n_2 = (d, e, f)$$

$$\mathbf{n_3} = (g, h, i)$$

As a matter of fact $\mathbf{n_1}$ is the vector that is parallel to U-axis, $\mathbf{n_2}$ is the vector that is parallel to the V-axis and $\mathbf{n_3}$ is the vector that is parallel to the W-axis.

Note: As it has been also pointed out by Ganapathy ([5]), *the computation of* u_0, v_0 *is rather sensitive. Therefore, if one is interested in a quite precise computation of those quantities, one should follow another approach* (see APPENDIX I).

6. Deficiencies of the algorithm

The above described algorithm has one important deficiency. Assume that the considered set of points has a symmetry with respect to any of the planes defined by any two axes of the 3-D coordinate system. If the optical axis lies on this plane, then some of the summations appearing in equations (E1) through (E11) will be equal to 0. Which ones exactly are equal to 0, depends on the symmetry. However, this is enough to reduce the rank of the coefficients' matrix (matrix A_{ij} below). But reduction of the rank of this matrix has as a consequence our inability to solve the linear system of the 11 equations in the 11 unknowns T_i, i=1, 2, 3, .. 11. It is therefore clear that one should not choose such configurations of points. However,

$$\begin{bmatrix} A_{11} & A_{12} \ldots\ldots & & A_{1m} \\ A_{21} & A_{22} \ldots\ldots & & A_{2m} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & & & \\ \cdot & & & \\ & & & \\ A_{n1} & & & A_{nm} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ \cdot \\ \cdot \\ \cdot \\ X_m \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \cdot \\ \cdot \\ \cdot \\ C_m \end{bmatrix}$$

the problem is not necessarily solved. This fact lies on more intrinsic reasons. Let us consider the above linear system of equations.

Definition: The above system of equations is called **ill-conditioned** if the determinant of the matrix of coefficients has a value that is very close to 0.

In such cases, **an error** is introduced in the values of the computed parameters. Moreover, consider the case where the coefficients A_{ij} are measured with an error $\pm \Delta A_{ij}$. This means that the value of A_{ij} lies in an interval $[\, A^*_{ij} - \varepsilon_{ij}\,,\ A^*_{ij} + \varepsilon_{ij}\,]$, $/\, i = 1,2,3, \ldots n \ \&\ j = 1,2,3, \ldots m$. The "stared" values A^*_{ij} are the ones that "correct" measurements should give. It is clear that we cannot be sure that for *no set of values* A_{ij}, the above system become ill-conditioned.

Definition: We call a linear system **critically ill-conditioned** if it is not ill-conditioned, but there is at least one set of values for A_{ij} in the intervals $[\, A^*_{ij} - \varepsilon_{ij},\ A^*_{ij} + \varepsilon_{ij}\,]$, $/\, i = 1,2,3, \ldots n \ \&\ j = 1,2,3, \ldots m$, for which the linear system becomes ill-conditioned.

In a recent work ([1]), that dealt with this problem, a measure of ill-conditioning was determined. In fact, the quantity:

$$\phi = \sum_{i=1}^{n} \sum_{j=1}^{m} |b_{ij}| \cdot \varepsilon_{ij}$$

where $b_{ij} = ((A^{-1})^{T})_{ij}$ $/\, i = 1,2,3, \ldots n \ \&\ j = 1,2,3, \ldots m$, can be used to determine whether the considered system is ill-conditioned. The following hold:

$$\phi < 1 \Rightarrow \text{Not critically ill-conditioned}$$
$$\phi \geq 1 \Rightarrow \text{Critically ill-conditioned}$$

Furthermore, an upper bound for the error ΔX_i $/\, i = 1,2,3 \ldots m$, in the computed values of X_k $/\, k = 1,2,3, \ldots m$, was established:

$$\Delta X_i \leq \frac{\phi}{1 - \phi} \| x \|_1 \ / \ i = 1,2,3 \ldots m$$

where $\| a \|_1$ is the first norm of matrix $a = (a_{jk}) / j = 1,2,3, \ldots n \; \& \; k = 1,2,3, \ldots m$, that is defined as follows:

$$\| a \|_1 \equiv \max_k \{ \sum_{j=1}^{n} | a_{jk} | \} \text{or} \max_j \{ \sum_{k=1}^{m} | a_{jk} | \}$$

One could use the parameter ϕ, to determine whether the system of equations (E1) thru (E11), is ill-conditioned or not, and therefore determine the " quality " of the calibration. However, the necessary overhead does not make the effort worth it, since one can easily tell from the calibration matrix whether the system is ill-conditioned (in which case most of its entries will be equal to 0), or not.

7. Implementation of the described algorithm

The algorithm we just descibed, has been implemented in the University of Rochester, during the spring of 1986. A full description of the program's output together with useful hints and other relevant information can be found in APPENDIX II.

REFERENCES

1. Alloimonos, Y., " On Low level Vision Computations ", PhD. Thesis, University of Rochester, 1986.

2. Ballard, D.H., and Brown, C.M., "Computer Vision", Prentice-Hall, N.J. 1982.

3. Brown, C.M., "Low Level Striping Routines", Department of Artificial Intelligence, University of Edinburgh, November 1974.

4. Brown, C.M., "The Striper Calibration System", Department of Artificial Intelligence, University of Edinburgh, November 1974.

5. Ganapathy, K., "Camera Location Determination Problem", AT&T Bell Laboratories, Holmdell, New Jersey.

6. Gennery, D., "Stereo Camera Calibration", Artificial Intelligence Laboratory, Computer Science Dept, Stanford University, November 1979.

7. Goldstein, H., "Classical Mechanics", Addison-Wesley, Reading Massachusetts, 1981.

APPENDIX I

The following describes a quick method for the determination of the of the coordinates of the pixel on the DataCube display through which the optical axis of the camera in use passes. The idea of the method relies on the notion and the properties of the so-called "vanishing point".

Method: A set of 8 points forming the vertices of a cube is constructed in 3-D space (see **Fig. 1.a**). These points can be constucted using 8 nuts screwed on vertical threaded rods, or using a set of 8 targets like the one shown in **Fig. 1.b**. The camera under consideration is moved until the points A, A', B, B', C, C', D, D' of the image plane (Datacube display) that correspond to the "vertices" of the original cube, are such, that the quadruples (A, A', C, C') and (B, B' D, D') represent sets of points lying on two distinct lines of the image, say ε_1 and ε_2 respectively (see **Fig. 2.a** and **Fig. 2.b**). Lines ε_1 and ε_2 intersect at a point P (the vanishing point). The coordinates of this point are the coordinates we are after. How can one determine the coordinates of P ? Recall that A, A', C, C' lie on the same line, namely ε_1. Using a cursor moving routine, one determines the coordinates of points A, A', C, C'. Then using a simple calculator, and applying a least-squares fitting method, the slope and the offset of line ε_1 are computed. The same procedure is repeated for the case of line ε_2. The slopes and offsets of lines ε_1 and ε_2 being determined, the coordinates of P can be computed; in fact one need only express ε_1 and ε_2 as linear equations in two unknowns, say x and y, and then solve the resulting linear system. The solution of this system will be the pair of coordinates of P .
Improvement of the results can be achieved by repeating the above procedure using other rigid configurations (see **Fig. 3.a** and **Fig. 3.b**).

Note: *Obviously, the above described method is general, and can be applied to any camera and display combination.*

Results: Application of the above described method for the case of a cube gave the following results, for the cameras of our laboratory:

CAMERA TYPE	COORDINATES
vidicon	(353, 247)
ccd	(334, 200)

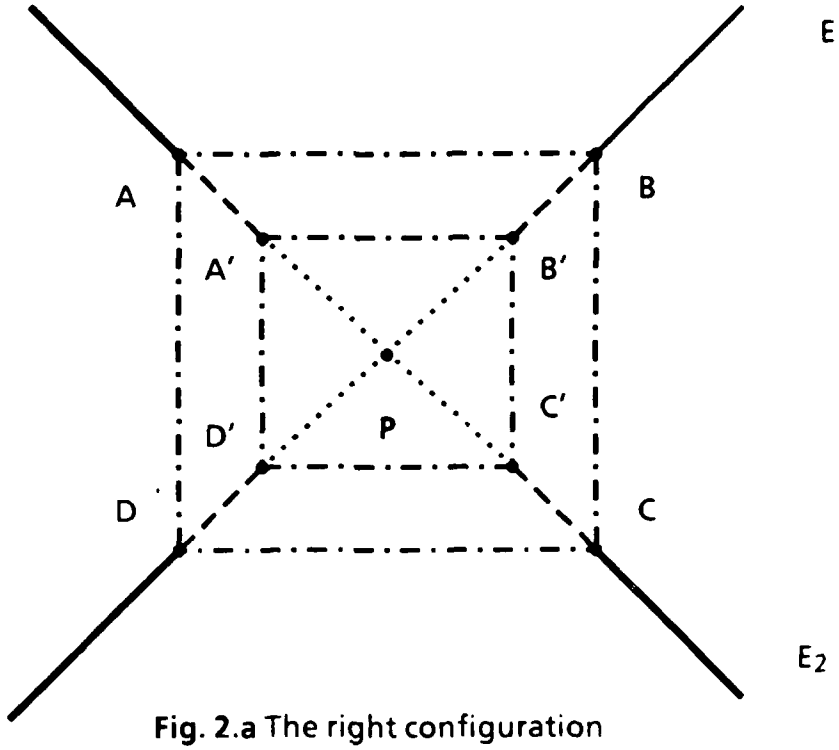


Fig. 2.a The right configuration

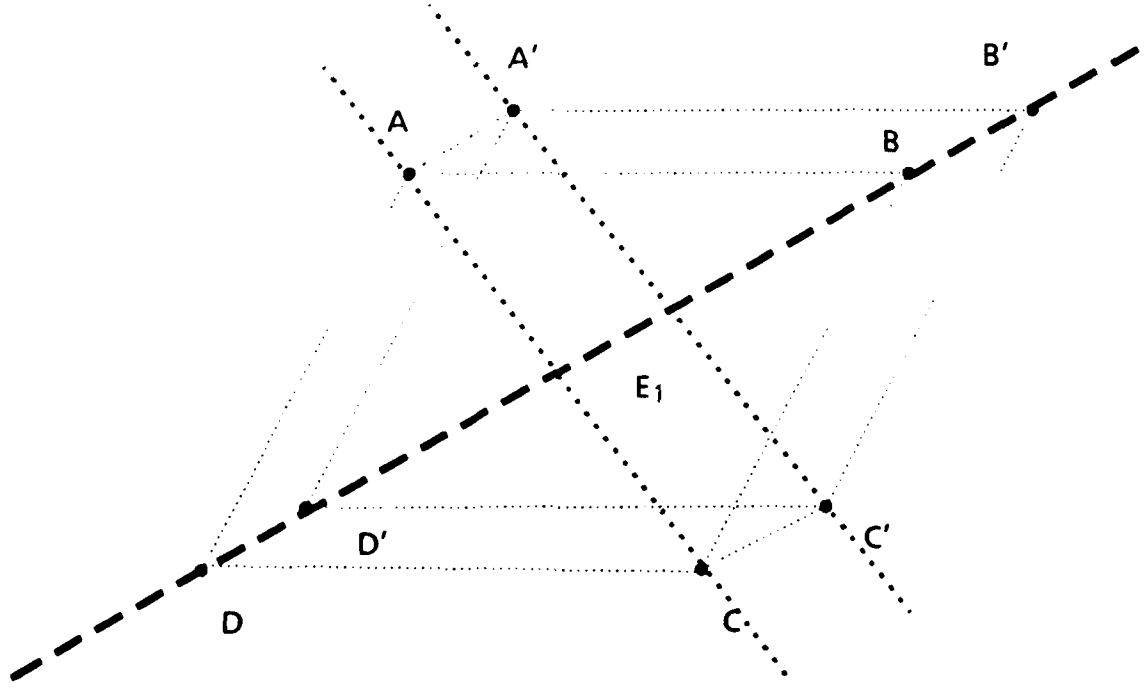


Fig. 2.b A wrong configuration.

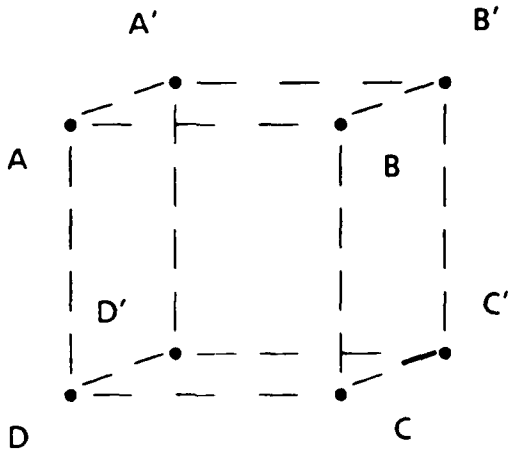


Fig. 1.a

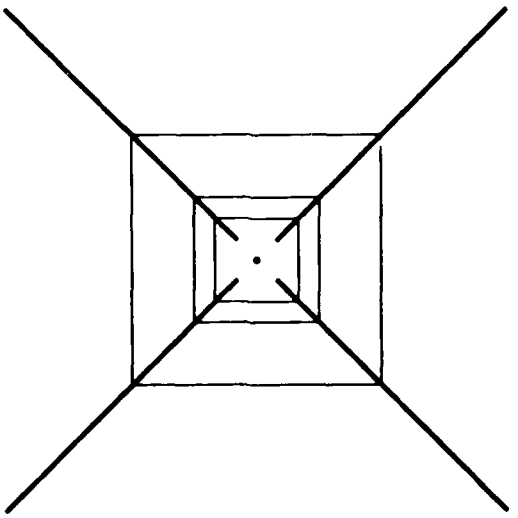


Fig. 1.b

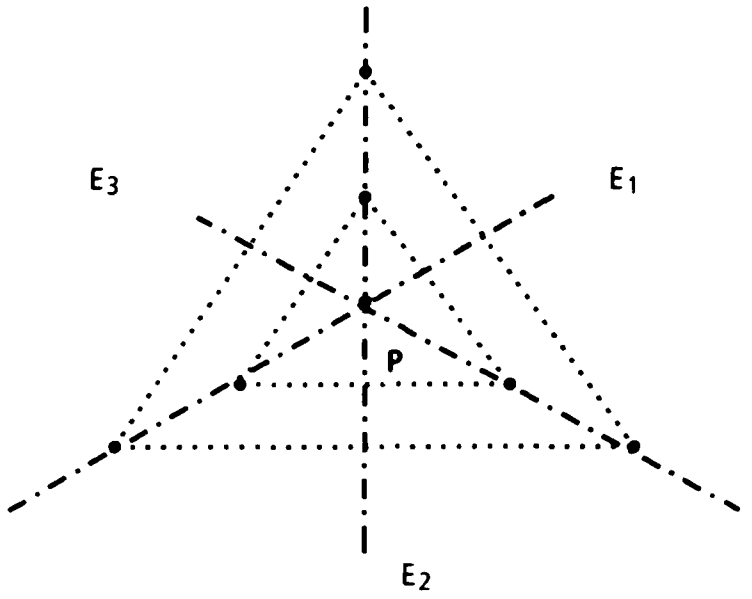


Fig. 3.a Case of a prism.

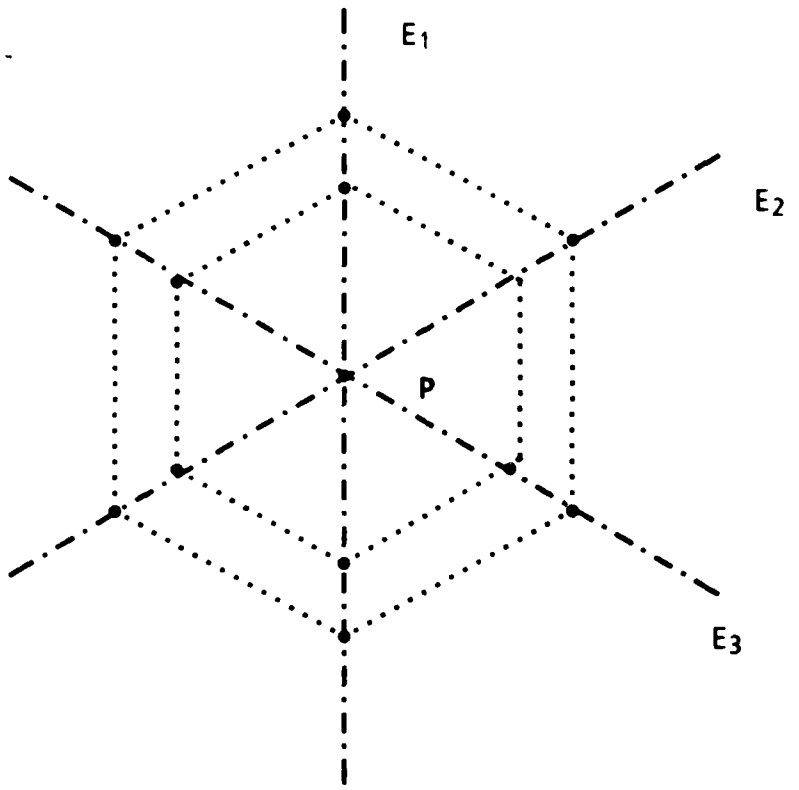


Fig. 3.b Case of a octahedron.

APPENDIX II

This appendix gives a) some very useful observations that have been made as well as hints, and b) the file structures that are needed together with a description of the output of the actual program.

Part I

Choosing the 3-D points

This is the most crucial part of the whole procedure. In fact the selection of the world points can lead to a disaster or a success. Some useful observations have been made:

a) choose a sufficient number of points; although the algorithm requires only 5 $^{1/2}$ points, a "trustworthy" calibration was found to require at least 8 or 9 points.

b) pay attention to the configuration of the 3-D points with respect to the optical axis and the 3-D (fixed) coordinate system; besides some obvious degenerate cases for which the algorithm will not work, there are also other cases that may cause problems. As a general suggestion, though, one should keep in mind that the more randomly the targets are selected, the higher the possibility for a succesful calibration. It is obvious, of course that one should not select configurations that experience some kind of symmetry (especially axial-symmetry) with respect to the optical axis.

c) the 3-D points should be selected "*as far as possible*" from the camera under consideration; of course this is in direct relation with the focal length of the camera. In general, the targets must be placed at such positions, so that all of them are focused. Obviously, if the camera supports apperture setting facilities, one would like to set the apperture to 16 or 22 under intense lighting conditions.

Measuring coordinates

① 3-D space: It has been observed that measuring the 3-D coordinates of the world points with an accuracy of approximately 0.1 inches is sufficiently satisfactory. However, one should keep in mind that all the quantities that have a dimension of $M^0L^1T^0$, e.g. focal length of the camera, world-point coordinates etc., must be expressed in the same **length** units, e.g. all in inches or all in millimeters ...

② Image plane: Using a cursor moving routine, one must determine the image plane coordinates with an inaccuracy of at most ± 3 pixels.

Discovering potential deficiencies of the camera in use

One problem that might arise is the following: "cheap optics" or "not properly mounted on the camera" lens. One could easily figure out whether this is a problem for the camera in use, or not, by placing some targets in front of the camera and trying to focus on them; if their images are displaced in a **non-consistent** way, then there is a high possibility of the camera having the above mentioned problem. Of course it is needless to say that there is no hope of calibrating this camera .

Algorithm's results

It has been observed that the computation of the values u_0, v_0, is rather sensible and related to the direction of the optical axis. In fact, one should use another approach in order to get more robust values for these parameters (see APPENDIX I). Besides that, the remaining parameters are precisely computed. It must be pointed out, though, that if one wants to get reliable results, one must try a few calibrations with the camera being horizontal, and for different camera positions (with respect to the 3-D coordinate frame) and sets of world-points, in order to get a feeling about the the response of the camera in use. Then other positions might be tried (pan, tilt or roll $\neq 0$) as well.

Part II

Description of the necessary file structures

The files that are used by the calibration routine are two:

i) the file " data3D ", that contains the number of experimental points as well as their 3D coordiates. This file must be set up by the user, and should have the following format:

line #1	#of_points_used
line #2	X_1 (space) Y_1 (space) Z_1
line #3	X_2 (space) Y_2 (space) Z_2
line #4	X_3 (space) Y_3 (space) Z3

etc.

Note: The X_i's, Y_i's and Z_i's are read as " double's ".

ii) the file " dataimage " that contains the image plane coordinates of the experimental points. This file is **either** set up by the user, in which case it has to have the following format:

line #1 u_1 (space) v_1
line #2 u_2 (space) v_2
line #3 u_3 (space) v_3
line #4 u_4 (space) v_4

etc.

or by invoking the special command called " dqstorepoint "; the latter is a cursor moving routine that uses the "mouse" of a SUN-Workstation that is connected to the DataCube. The user:

1) selects points by clicking the left button of the mouse
2) confirms a selected point and makes an entry for the corresponding coordinates to the file " dataimage ", by clicking the middle button of the "mouse"
3) " exits " by clicking the right button of the "mouse".

Note: It is worth mentioning that the entries of the (i + 1)-th line of the file "data3D" and the ones of the i-th line of the file "dataimage" correspond to the i-th experimental point, i = 1,2,3,4,5,6, ... n.

A typical session

The following is a script of a typical session. We assume that the files " data3D " and " dataimage " were appropriately set up as described abc . The names of the parameters are consistent with the ones appearing to the description of the algorithm...

```
< ISIDORE >>> % cat data3D
9
552.2 376.5 1989.1
815.75 376.5 1989.1
815.75 169.4 1989.1
351.4 169.4 1689.3
351.4 376.5 1689.3
702.8 169.4 1185.975
```

```
702.8 376.5 1185.975
753.0 169.4 1085.575
753.0 376.5 1085.575
< ISIDORE >>> % cat dataimage
286 164
504 164
505 318
62 327
64 143
432 372
429 89
517 385
517 67
< ISIDORE >>> % dqcalibrate


 Please enter the focal length of the camera in use: 25


 Reading  the 3-D coordinates  Number of points= 9



 Point number 1
 552.200000    376.500000    1989.100000

 Point number 2
 815.750000    376.500000    1989.100000

 Point number 3
 815.750000    169.400000    1989.100000

 Point number 4
 351.400000    169.400000    1689.300000
```

```
Point number 5
351.400000   376.500000   1689.300000

Point number 6
702.800000   169.400000   1185.975000

Point number 7
702.800000   376.500000   1185.975000

Point number 8
753.000000   169.400000   1085.575000

Point number 9
753.000000   376.500000   1085.575000

Reading the image-plane coordinates

Point number 1
286.000000   164.000000

Point number 2
504.000000   164.000000

Point number 3
505.000000   318.000000

Point number 4
62.000000   327.000000

Point number 5
64.000000   143.000000

Point number 6
432.000000   372.000000

Point number 7
```

```
429.000000    89.000000

Point number 8
517.000000    385.000000

Point number 9
517.000000    67.000000



 The solution  matrix is the following :

   -5.288670    -0.060201    -1.428586    3957.598778
   -0.051500    4.650256    -0.929366    -914.765229
   -0.000063    -0.000189    -0.003655    1.000000

 r2 = 74628.615332



 Observe ad + be + cf = 0.011060

 The skew angle estimation parameter is equal (in radians) to
: 0.011060

   u0=415.503321      v0=188.044822

 The resolution parameters are:
   Ku=57.512410     Kv=51.273858



 The remaining parameters are :
   a=-0.999847 b=0.003509   c=0.017124
```

```
   d=-0.008440 e=0.998633   f=-0.051583
   g=-0.017282 h=-0.051723  i=-0.998512

   p=672.994285  q=-235.026810  r=273.182385
```

The directions of the 3 axes with respect to the fixed 3D coordinate system are

```
         U-axis: (-0.999847,0.003509,0.017124)
         V-axis: (-0.008440,0.998633,-0.051583)
(optical)Z-axis: (-0.017282,-0.051723,-0.998512)
```

The position of the center of the camera is at:

```
   Xo=675.628994    Yo=246.473854    Zo=249.128475
```

```
< ISIDORE >>> %
```

END

DATE

3-88

DTIC