M. TECH. (COMPUTER SCIENCE) DISSERTATION SERIES

CAMERA CALIBRATION FOR STEREO VISION USING IMAGING GEOMETRY

a dissertation submitted in partial fulfillments of the requirements for the M. Tech.(Computer Science) degree of the Indian Statistical Institute

by

PRASENJIT PAL

under the supervision of

DR. BHABATOSH CHANDA

INDIAN STATISTICAL INSTITUTE

203, Barrackpur Trunk Road

Calcutta - 700 035

Contents.

1. Introduction

Determining camera location from image to space point correspondences (i.e., 2-D and 3-D correspondences) is a very basic problem in image analysis and cartography. It can be applied to aircraft location, robot calibration and so on. Roughly, this problem can be considered as estimatimg the three dimensional location from which an image was taken by a set of recognized landmark appearing in the image. The camera location determination problem is formally defined as follows: "Given a set of m control points, whose three dimensional co-ordinates are known in some co-ordinate frame, and given an image in which some subseet of m control points is visible, determine the location (relative to the co-ordinate system of the control points) from which the image was obtained". In solving this problem, we shall assume that the focal length of the camera is known and the 2-D to 3-D line or points correspondences are given.

Here, we are not finding the absolute location of the camera, rather the location of one camera relative to an another camera, i.e. the relative orientation of two cameras.

This report is organized as follows. A brief description of perspective projection, camera model and stereo imaging is sited in section 2. Camera calibration is defined in the section 3. Section 4 presents formal description of the problem and section 5 describes the algorithm used here. Section 6 will give some experimental results.

2. Prerequsites

2.1 Perspective transformation

Perspective transformation projects 3D-points onto a plane. We define a camera co-ordinate system (x,y,z) as having the image plane parallel to the x-y plane , the optical axis along z-axis. Thus the centre of the image plane is at the co-ordinate (0,0,f) and the centre of the lens is at the origin, f being the focal length of the lens. Assume that the camera co-ordinate system (x,y,z) is aligned with the world co-ordinate system (X,Y,Z).

Let (X,Y,Z) be the world co-ordinate of any point in 3-D scene . We wish to obtain the image co-ordinate (x',y') of the projection of the point (X,Y,Z) onto the image plane. By simple mathematical deduction we get,

$$x' = f \frac{X}{Z}$$

$$y' = f \frac{Y}{Z}$$

The homogeneous co-ordinates of a point with cartesian co-ordinate (X,Y,Z) are defined as (kX,kY,kZ,k), where k is an arbitrary non-zero constant. Clearly, conversion of homogeneous co-ordinates back to cartesian co-ordinates is acomplished by dividing the first three homogeneous co-ordinates by the forth. Then we can obtain the perspective tranformation matrix :

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/f & 0 \end{pmatrix}$$

2.2 Camera model

To get the image co-ordinate of a point in perspective transformation we assumed that the camera and the world co-ordinate systems are coincident. Now we consider a more general situation in which the two co-ordinate systems are allowed to be separate.

Let the co-ordinate of the centre of projection (i.e., lens) is (X_0, Y_0, Z_0). Pan, the angle between x and X axis is a and tilt, the angle between z and Z axis is ß .

After a translation and rotation with respect to x and z axis we can aligned the camera co-ordinates with the world co-ordinates. So the image co-ordinates p(x',y') can be obtained by

$$c = P.R_{\beta}.R_{a}.G.W \quad ....(*)$$

where P = projection transformation matrix ;

R_a , R_β = rotation matrices;

G = translation matrix;

W = homogeneous world point (X,Y,Z,1), taking k = 1;

c = homogeneous co-ordinate of the image point.

Writing explicitly,

$$x' = f[(X-X_0)\cos a + (Y-Y_0)\sin a]/z' \quad ....(1)$$

$$y' = f[-(X-X_0)\sin\alpha\cos\beta + (Y-Y_0)\cos\alpha\cos\beta + (Z-Z_0)\sin\beta]/z'; \quad \text{.........(2)}$$

where

$$z' = -(X-X_0)\sin\alpha\sin\beta + (Y-Y_0)\cos\alpha\sin\beta - (Z-Z_0)\cos\beta$$

2.3 Stereo imaging

The mapping of a 3-D scene onto a image plane is a many-to-one transformation. That is, an image point does not uniquely determine the location of a corresponding world point. The missing depth information can be obtained by using stereoscopic imaging technique. Stereo imaging involves obtaining two separate image views of an object of interest. The objective is to find the co-ordinate (X,Y,Z) of a point P given its image points (x'_1,y'_1) and (x'_2,y'_2).

The transformation between two camera stations can be treated as a rigid body motion and can thus be decomposed into a rotation and a translation. If $r_l = (x_l,y_l,z_l)$ is the position of point P measured in the left camera co-ordinate system and $r_r = (x_r,y_r,z_r)$ is the position of the same point measured in the right camera co-ordinate system, then

$$r_r = R.r_l + r_0$$

where R is a 3x3 orthonormal matrix representing the rotation and r_0 is the offset vector corresponding to the translation.

Once R and r_0 are known, we can compute the position of a point with known left and right image co-ordinates. If (x'_l,y'_l) and (x'_r,y'_r) are these co-ordinates, then

$$\left(r_{11}\frac{x'_l}{f}+r_{12}\frac{y'_l}{f}+r_{13}\right)z_l+r_{14}=\frac{x'_r}{f}z_r,$$

$$\left(r_{21}\frac{x'_l}{f}+r_{22}\frac{y'_l}{f}+r_{23}\right)z_l+r_{24}=\frac{y'_r}{f}z_r,$$

$$\left(r_{31}\frac{x'_l}{f}+r_{32}\frac{y'_l}{f}+r_{33}\right)z_l+r_{34}=z_r.$$

We can use any two of these equations to solve for z_l and z_r and then we can compute the 3-D co-ordinate by

$$r_l=(x_l,y_l,z_l)=\left(\frac{x'_l}{f},\frac{y'_l}{f},1\right)z_l,$$

$$r_r=(x_r,y_r,z_r)=\left(\frac{x'_r}{f},\frac{y'_r}{f},1\right)z_r.$$

3. Camera calibration

In eqution (1) and (2) we obtained explicit equations for the image co-ordinates (x',y') of a world point $w(X,Y,Z)$. Implementation of these equations requires knowledge of camera offsets and the angle of pan and tilt. While these parameters can be measured directly, it is often more convenient to determine one or more of these parameters by using the camera itself as a measuring device. This requires a set of image points whose world co-ordinates are known and the computational procedure used to obtain the camera parameters using these known points is often referred to as camera calibration.

With reference to the equation $c = P.R_\beta.R_a.G.W$ ....(3.1) let $A = P.R_\beta.R_a.G$. The elements of A contains all the camera parameters and from the equation (3.1) we have

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

So the cartesian co-ordinate of the image point (x',y') is given by

$$x' = \frac{c_1}{c_4} \quad \text{and} \quad y' = \frac{c_2}{c_4}.$$

Writing explicitly,

$$a_{11}X + a_{12}Y + a_{13}Z - a_{41}x'X - a_{42}x'Y - a_{43}x'Z - a_{44}x' + a_{14} = 0$$
$$a_{21}X + a_{22}Y + a_{23}Z - a_{41}y'X - a_{42}y'Y - a_{43}y'Z - a_{44}y' + a_{24} = 0$$

These two equations consists of 12 unknowns, so the camera calibration procedure then consists of:

1. Obtaining more than 6 world points with known co-ordinates (X_i, Y_i, Z_i), $i = 1,2..6$.

2. Corresponding image points (x'_i, y'_i).

3. Using this points solve the above equations for the unknowns a_{ij}'s.

In practice, more than 6 points are taken and solution is achieved by least square estimation.

However, as suggested in section 2.3, to compute depth by stereo imaging we need rotation and translation parameters between two camera (left and right) co-ordinate systems. We call the procedure for computing these relative parameters as relative calibration. There are two different ways of relative calibration as given below.

3.1 Relative calibration using camera co-ordinates

a. Here we assume that the actual three dimensional co-ordinates $r_l = (x_l, y_l, z_l)$ and $r_r = (x_r, y_r, z_r)$ with respect to the left and right camera co-ordinate systems respectively, are known. We can expand the relation $r_r = R.r_l + r_0$ to

$$r_{11}*x_l + r_{12}*y_l + r_{13}*z_l + r_{14} = x_r$$

$$r_{21}*x_l + r_{22}*y_l + r_{23}*z_l + r_{24} = y_r$$

$$r_{31}*x_l + r_{32}*y_l + r_{33}*z_l + r_{34} = z_r$$

where $[r_{ij}]$, $i,j = 1,2,3$, are the elements of rotation matrix R and r_{14}, r_{24} and r_{34} are the offset r_0. Given a set of corresponding world points (x_l, y_l, z_l) and (x_r, y_r, z_r) we have to determine the coefficients r_{ij}'s. Here we can incorporate the six constraints of orthonormality ie.

$$R.R^T = I.$$

b. In above case we assumed that the 3-D co-ordinates with respect to left and right camera co-ordinate system are known, but actually we do not know the points $r_l = (x_l, y_l, z_l)^T$ and $r_r = (x_r, y_r, z_r)^T$ themselves, only their projections in the image. Given the focal length of the cameras we can determine the ratios of x and y to z using $x'_l/f = x_l/z_l$ and $y'_l/f = y_l/z_l$ and similerly for x'_r and y'_r. we can regard z_l and z_r as additional unknowns. Each pair of corresponding points now provide only one constraints, not three. Unfortunately the equations are non-linear in nature and this makes them harder to solve and will give multiple solutions. For a given point pair (x'_l, y'_l) and (x'_r, y'_r) we have

$$r_{11}*x'_l + r_{12}*y'_l + r_{13}*f + r_{14} * \frac{f}{z_l} = x'_r * \frac{z_r}{z_l}$$

$$r_{21}*x'_l + r_{22}*y'_l + r_{23}*f + r_{24} * \frac{f}{z_l} = y'_r * \frac{z_r}{z_l}$$

$$r_{31}*x'_l + r_{32}*y'_l + r_{33}*f + r_{34} * \frac{f}{z_l} = f * \frac{z_r}{z_l}$$

There are 3 equations in forteen unknowns $r_{11}..r_{34}$, z_l and z_r. Each additional point pair provides three more equations but also introduce two more unknowns.

Now we can use the fact that the rotation matrix $R=[r_{ij}]$ should be orthonormal. This introduces six additional constraints. Given n points pairs, we have (12 + 2n) unknowns and 7 + 3n constraints. A solution is thus possible if we have five points pairs, provided that the equations are independent.

4. Problem definition

Our problem is to find the relative orientation of the two camera co-ordinate system. We could do this after calibrating the cameras according to the procedure described above. But here we are not using this procedure for two reasons.

1. Since we are using two cameras so we have to calibrate two cameras independently then find their relative orientations. In each of the three steps some amount of error will be incurred.

2. We have not sufficient equipments to find the known world points.

For this reason we are directly finding the relative orientations of the cameras without calibrating them independently.

To find the relative orientations of two cameras we are to face the problem of solving non-linear equations. To avoid this we took a different approach. Here we are finding the rotational matrix $R[r_{ij}]$ and translation matrix r_0 without solving the non-linear equations. The method is described in the algorithm.

5.1 Description of the algorithm

To find the relative orientation which determines the transformation between two camera co-ordinate systems. We will use a Grid which is fixed relative to world co-ordinate system.

Suppose we have the corresponding junction points of two diffrent images and also the centre points of two image .To avoid the solving of non-linear equations we will take another approch which is described below by an example.

Let AOB be a straight line in the original grid, where the 3-D co-ordinates of A, O, and B are (x_n, y_n, z_n), (x_0, y_0, z_0) and (x_{-n}, y_{-n}, z_{-n}) respectively. suppose the image points of A, O and B are $A'(x'_n, y'_n)$ $O'(x'_0, y'_0)$ and $B'(x'_{-n}, y'_{-n})$ respectively, in the left camera co-ordinate system Let (a,b,c) be the direction cosine of AOB and l be the length of OA as well as OB. Then

$$\begin{pmatrix} x_n \\ y_n \\ z_n \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} + l \begin{pmatrix} a \\ b \\ c \end{pmatrix} \quad \text{........(1)}$$

and

$$\begin{pmatrix} x_{-n} \\ y_{-n} \\ z_{-n} \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} - l \begin{pmatrix} a \\ b \\ c \end{pmatrix} \quad \text{........(2).}$$

We know that if (x',y') be the image point of the point whose 3-D co-ordinate is (x,y,z) ,then

$x' = f*x/z$, $y' = f*y/z$, where f is the focal length.

Using this ,

$$x'_n = \frac{f*x_n}{z_n} = \frac{f*(x_0 + la)}{(z_0 + lc)} \quad \text{..............(3)}$$

$$y'_n = \frac{f*y_n}{z_n} = \frac{f*(y_0 + lb)}{(z_0 + lc)} \quad \text{..................(4)}$$

$$x'_n - x'_0 = \frac{f*(x_0 + la)}{(z_0 + lc)} - \frac{f*x_0}{z_0} = \frac{f*l*a}{z_0 + lc} \quad \text{......(5).}$$

Similerly,

$$x'_0 - x'_{-n} = \frac{f*l*a}{z_0 - lc} .$$

$$\frac{x'_n - x'_0}{x'_0 - x'_{-n}} = \frac{z_0 - lc}{z_0 + lc} = k' \text{ (say)}.$$

Therefore

$$c = \frac{(1 - k')}{(1 + k')*l} * z_0 = k_1 * z_0 \text{ (say)}. \quad \ldots\ldots\ldots(6).$$

From equation (1)

$$a = \frac{(x'_n - x'_0)(1 + l*k_1)}{f*l} * z_0 = k_2 * z_0 \quad \ldots..(7) .$$

Similerly from

$$y'_n - y'_0 = \frac{f * l * b}{(1 + l * k_1) * z_0}$$

$$b = \frac{(y'_n - y'_0)*(1 + l*k_1)}{f * l} * z_0 = k_3 * z_0 \quad \ldots..(8).$$

Since $a^2 + b^2 + c^2 = 1$,

$$z_0^2 = 1/(k_1^2 + k_2^2 + k_3^2)$$

Therefore from (6) , (7) and (8) we get the value of a, b and c

This is our basic procedure. To minimising the error for calculating z_0 we may consider as many lines as possible. here we considered 16 oblique lines passing through the centre of the grid and 18 grid lines: horizontal and vertical.

Now, consider a boundary junction point $r_1 = (x_1,y_1,z_1)$ in the left image and $r_r = (x_r, y_r, z_r)$ be the corresponding point in the right image. (x_{0r}, y_{0r}, z_{0r}) and (x_{0l}, y_{0l}, z_{0l}) be the centre of right and left image respectively. Then we have

$$r_r = R.r_l + r_0 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (9).$$

Using equation (1)

$$\begin{pmatrix} x_{0r} + l.a_r \\ y_{0r} + l.b_r \\ z_{0r} + l.c_r \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \begin{pmatrix} x_{0l} + l.a_l \\ y_{0l} + l.b_l \\ z_{0l} + l.c_l \end{pmatrix} + \begin{pmatrix} r_{14} \\ r_{24} \\ r_{34} \end{pmatrix} \quad \ldots\ldots (10)$$

where (a_l, b_l, c_l) and (a_r, b_r, c_r) be the direction cosine of the same line in the left and right image respectively.

Or,

$$\begin{pmatrix} x_{0r} \\ y_{0r} \\ z_{0r} \end{pmatrix} + l \begin{pmatrix} a_r \\ b_r \\ c_r \end{pmatrix} = \begin{pmatrix} R \end{pmatrix} \begin{pmatrix} x_{0l} \\ y_{0l} \\ z_{0l} \end{pmatrix} + \begin{pmatrix} R \end{pmatrix} .l. \begin{pmatrix} a_l \\ b_l \\ c_l \end{pmatrix} + \begin{pmatrix} r_{14} \\ r_{24} \\ r_{34} \end{pmatrix}$$

$$\ldots\ldots (11).$$

From here we get

$$\begin{pmatrix} a_r \\ b_r \\ c_r \end{pmatrix} = \begin{pmatrix} R \end{pmatrix} . \begin{pmatrix} a_l \\ b_l \\ c_l \end{pmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (12).$$

So we proceed as follows:

step 1. Find the direction cosines of the oblique lines passing through the centre using above algorithm and store them in an array.

step 2. For each grid line find the point of intersections with the opposite middle grid line to get the centre point of the former grid line. then find the direction cosine of the line using the previous algorithm. Store these direction cosines also.

step 3. Using these data solve the equation (12) for the values of $R[r_{ij}]$ applying least square estimation.

step 4. For each line whose direction cosine is found, obtain the value of (x_{0l}, y_{0l}, z_{0l}) and (x_{0r}, y_{0r}, z_{0r}) using equations (3) and (4), since at the time of finding the direction cosine we found the value of z_0 which we can use now.

step 4. Solve the equation (11) for r_{14}, r_{24} and r_{34} using equation (12) and the values of (x_{0l}, y_{0l}, z_{0l}) and (x_{0r}, y_{0r}, z_{0r}).

5.2 Finding junction points

In the two transformed images of the grid we will use the points in the left image whose corresponding image co-ordinate in the right image are known.That is why we will use only the junction points of the grid. So here we find the corresponding junction points of two images.

Inputs : Two transformed images of the grid.

Outputs :1. The boundary junction points of two images and their correspondences.

2. Centre points of two images.

step 1 : Find the corner points of the image scanning the image by a straight line of the form x/a + y/b = 1. For each corner points change the value of a and b accordingly. Say for the first corner point, initialize a and b by 1 and increse both of them after each unsuccessful search.

step 2: Starting from the first corner traverse the boundary of the grid to find the boundary junction points. while traversing boundary , for each boundary pixel find the number of 8-neighbours. If the number of 8-neighbours is greater than 2 then from there stack the points until a point is reached whose number of 8-neighbours is two.take the average of the points those are stacked, to get a junction point.

For the each corner we will get two junction points which are misleading. To overcome this we will ignore the points having more than two 8-neighbours but the number of elements in the stack is 1.

```
xxxxxx*xxxxxx          xxxxxx*
      x                      x
      x                      x
      x                      x
      x                      x
```

figure 3. Example of boundary junction points

step 3: Suppose arr[0..31] is the set of boundary junction points. To get the centre point we will take two straight lines joining the points arr[4], arr[20] and arr[12], arr[28]. Find the point of intersection of these two straight lines to get the centre point.

6. Experimental results

Stage 1. Create a square grid containing 64 squares each of size 64x64 pixels.

Stage 2. Create two images of the grid from different positions and different angles. This is done in two steps.

Inputs : Position of the camera ie,co-ordinate of the centre of the image plane, the value of pan and tilt, focal length and height of the grid.

Output : Transformed image of the grid.

step1: Using camera model and corresponding transformation transform the junction points of the grid.

step2: Join the transformed junction points according as they are joined in the original grid.

Stage 3. Find the corresponding boundary junction points of two transformed images and also their centre points by the algorithm described in 5.2.

Stage 4. Find the rotation matrix R and translation vector r_0 by the algorithm described in 5.1.

Here we have 34 lines, so first we start from equation (12) and by least square estimation we will get 9 linear equations in 9 unknowns r_{ij}, $i,j = 1,2,3$. To incorporate the 6 normalizing conditions we multiply two equations of the above 9 equations (not necessarily distinct) to get 45 equations in 45 unknowns, each being a quadratic term of the form r_{11}^2, $r_{11}r_{12}$ etc. Totally, we get 51 equations in 45 unknowns. In matrix form,

$$A_{51x45}\ \beta_{45x1} = Y_{51x1}$$

From this we get,

$$\beta_{45x1} = [A'A]^{-1}{}_{45x45}\ A'_{45x51}\ Y_{51x1}$$

Now, considering the values of r_{11}^2, r_{12}^2, r_{13}^2, $r_{11}r_{12}$, $r_{12}r_{13}$, $r_{13}r_{11}$, we will get two sets of values of (r_{11},r_{12},r_{13}). We take one which satisfies equation (12) more accurately than the other.

Similerly, we get (r_{21},r_{22},r_{23}) and (r_{31},r_{32},r_{33}).

Then solve equation (11) for transition vector (r_{14},r_{24},r_{34}).

7. Conclusion We have presented a method for computing relative orientation between two cameras, by which we can find the depth of an object [see 2.3]. This algorithm uses point and straight line correspondences of two images with distinct views. The main advantage of this method is that it decouples rotation and translation, and hence reduces computations. With respect to error in the solution due to noise in the input image data we have emperically observed that :

(1) Adding more feature correspondences to the solution reduces the error.

(2) All the lines should not be parallel to x-y plane.

References :

1. Robot Vision by B. K. P. Horn.

2. Robotics : Control, Sensing, Vision and Intelligence by Fu, Gonzales and Lee.

3. IEEE Transaction on Pattern Analysis and Machine Intelligence, Liu, Huang and Faugeras, Jan, 1990.

RESULT

The square grid is in first quadrant in 3-D co-ordinate system having one of it's corner at (0,0,100) and the plane of the grid is parallel to X-Y plane.

For the Left image :

Camera co-ordinate = (256,256,0).

Focal length = 30.

For the Right image :

Camera co-ordinate = (300,300,0).

Focal length = 30.

Result without iteration :

r11= 1.004868 r12= -0.059084 r13= 0.115108

r21= 0.011318 r22= 0.998758 r23= -0.048527

r31= -0.000446 r32= 0.121124 r33= -0.992637

r11**2 + r12**2 + r13**2 = 1.026500

r21**2 + r22**2 + r23**2 = 1.000000

r31**2 + r32**2 + r33**2 = 1.000000

r11*r21 + r12*r22 + r13*r23 = -0.053224

r11*r31 + r12*r32 + r13*r33 = -0.121865

r21*r31 + r22*r32 + r23*r33 = 0.169138

After first iteration :

r11= 0.991594 r12= -0.059084 r13= 0.003318

r21= 0.065144 r22= 0.998758 r23= -0.048527

r31= -0.000446 r32= 0.048259 r33= 0.992637

r11**2 + r12**2 + r13**2 = 0.986761

```
r21**2 + r22**2 + r23**2 = 1.004116
r31**2 + r32**2 + r33**2 = 0.987658
r11*r21 + r12*r22 + r13*r23 = 0.005425
r11*r31 + r12*r32 + r13*r33 = -0.000000
r21*r31 + r22*r32 + r23*r33 = 0.000000
```

After second iteration :

```
r11= 0.998247    r12= -0.059084    r13= 0.003324
r21= 0.059154    r22= 0.996695    r23= -0.048527
r31= -0.000446    r32= 0.048657    r33= 0.998835

r11**2 + r12**2 + r13**2 = 1.000000
r21**2 + r22**2 + r23**2 = 0.999255
r31**2 + r32**2 + r33**2 = 1.000039
r11*r21 + r12*r22 + r13*r23 = -0.000000
r11*r31 + r12*r32 + r13*r33 = -0.000000
r21*r31 + r22*r32 + r23*r33 = 0.000000
```

After third iteration :

```
r11= 0.998247    r12= -0.059084    r13= 0.003323
r21= 0.059176    r22= 0.997069    r23= -0.048527
r31= -0.000446    r32= 0.048638    r33= 0.998815

r11**2 + r12**2 + r13**2 = 1.000000
r21**2 + r22**2 + r23**2 = 1.000003
r31**2 + r32**2 + r33**2 = 0.999998
r11*r21 + r12*r22 + r13*r23 = 0.000000
```

```
r11*r31 + r12*r32 + r13*r33 = -0.000000
r21*r31 + r22*r32 + r23*r33 = 0.000000

After forth iteration :
r11= 0.998247    r12= -0.059084    r13= 0.003323
r21= 0.059176    r22= 0.997067    r23= -0.048527
r31= -0.000446    r32= 0.048638    r33= 0.998816

r11**2 + r12**2 + r13**2 = 1.000000
r21**2 + r22**2 + r23**2 = 1.000000
r31**2 + r32**2 + r33**2 = 1.000000
r11*r21 + r12*r22 + r13*r23 = -0.000000
r11*r31 + r12*r32 + r13*r33 = -0.000000
r21*r31 + r22*r32 + r23*r33 = -0.000000

Translation (r14,r24,r34) = (45.613932  52.916884 11.852839)
```

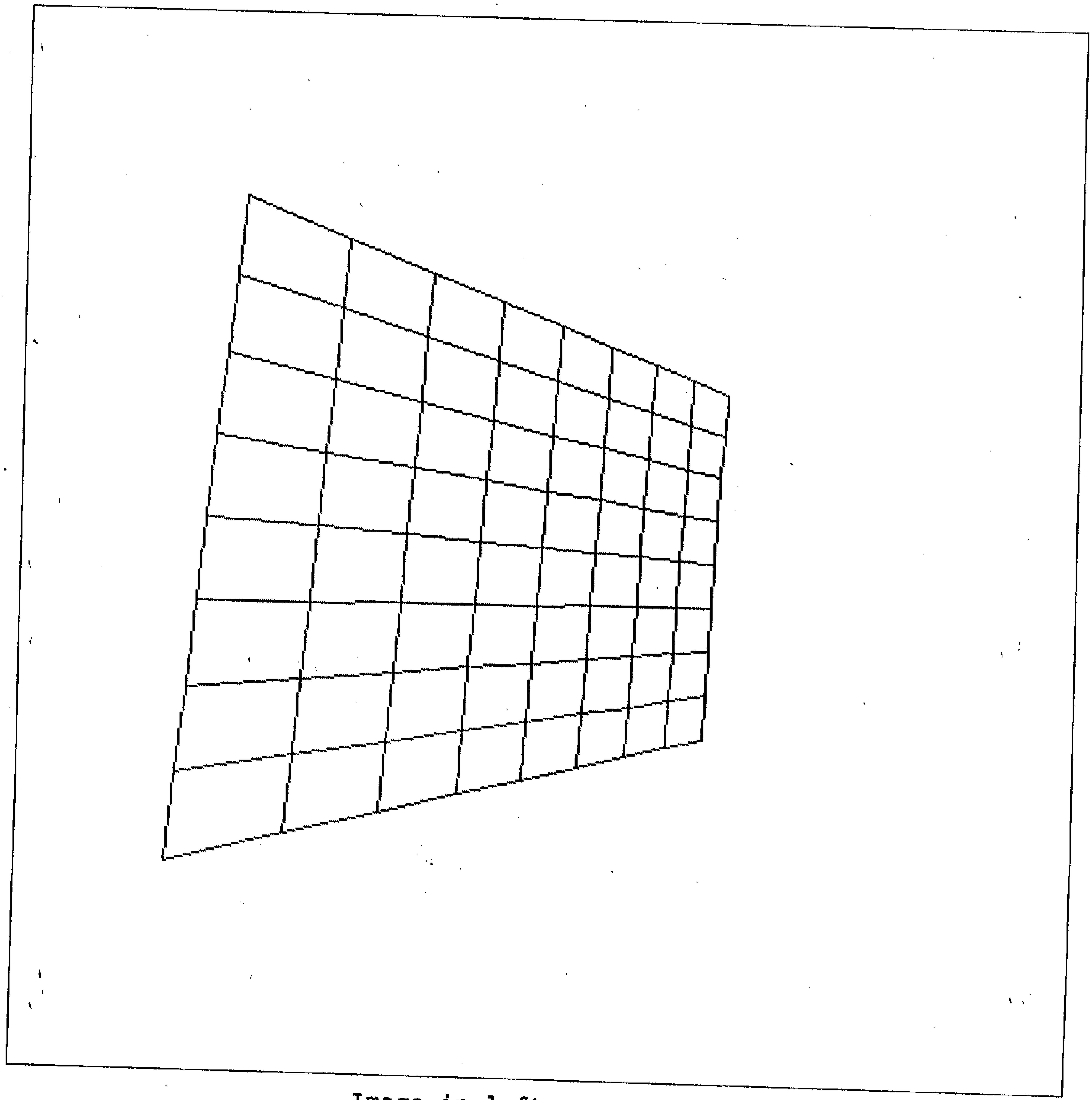

Image in left camera.

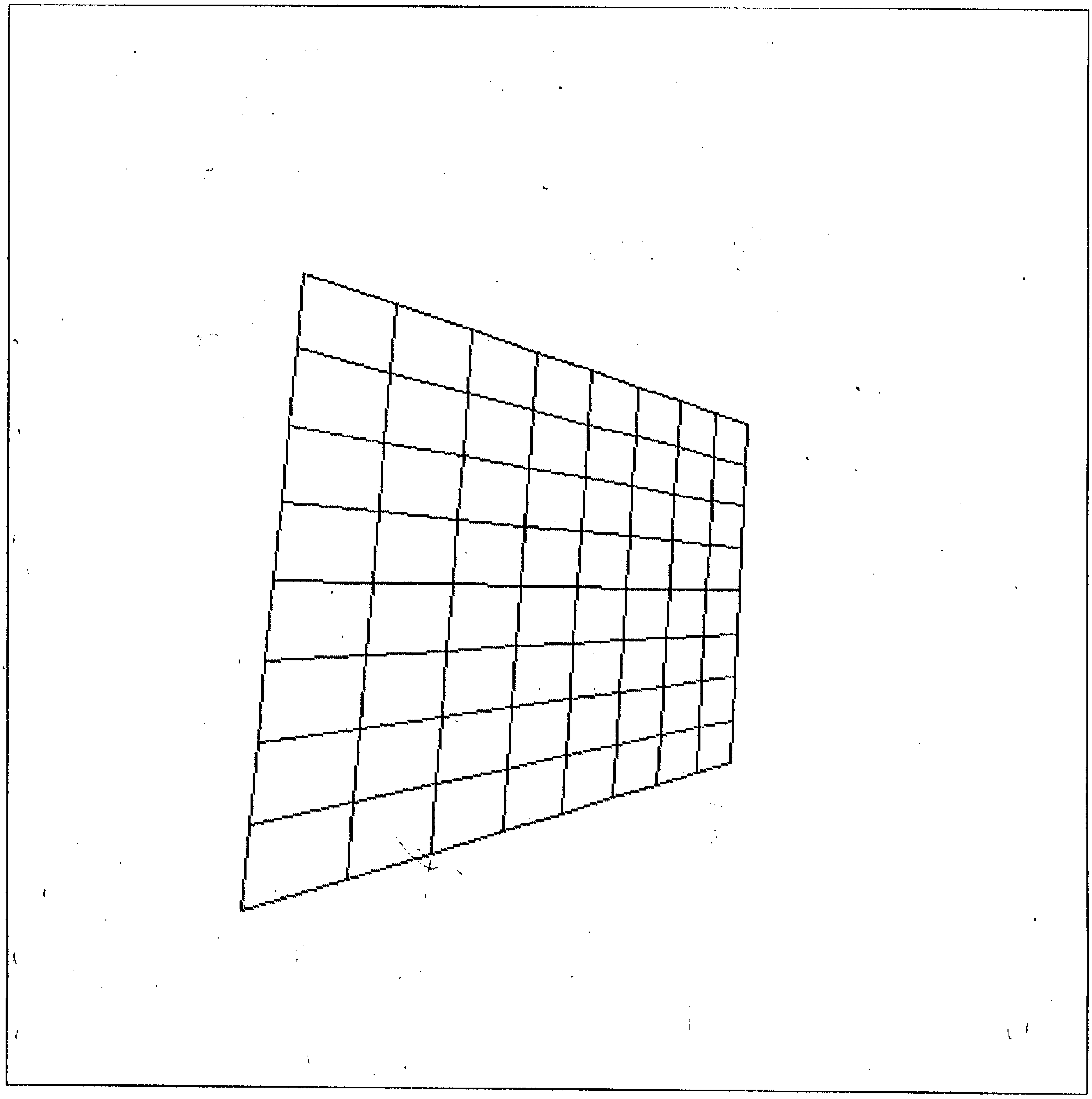

Image in right camera.