1 Introduction

Computation of the position and orientation of a camera (pose estimation) from a single image with respect to a known object has important applications in camera calibration, object recognition, and photogrammetry from aerial imagery. When the relative geometry of n feature points is used, this problem is called the Perspective-n-Point problem (PnP) [3, 12, 7].

This paper focuses on the degenerate case in which the points are *coplanar*. It is important in practice to be able to solve this degenerate case. In aerial imagery, for example, the spread of feature points may be large compared with the elevations of the points. Even if the map shows that the ground is not planar or that feature points can be taken both on top of buildings and at ground level, these feature points should be considered coplanar if the matrix describing the geometry of feature points can be considered of rank 2 instead of 3; this decision can be taken by comparing the respective amplitudes of the singular values of the matrix in a singular value decomposition [9]. A general algorithm which assumes that these feature points are noncoplanar and fails to detect that this case is degenerate would probably produce inaccurate camera pose estimations.

For the case of coplanar feature points, researchers have formulated closed form solutions for configurations of three feature points and four feature points. The P3P problem (with three noncollinear points) can have as many as four possible solutions [6, 4, 11]. On the other hand, the P4P problem has a single theoretical solution [3, 12, 1, 5] when the coplanar points are in an ordinary configuration (no three collinear scene points, noncollinear image points).

Clearly, there is a problem with closed form calculations claiming a single solution for four coplanar points. The problem can be detected by the following reasoning:

- With a scaled orthographic projection, there are always two acceptable solutions; the two poses are mirror images with respect to a plane parallel to the image plane.
- For configurations where the object's distance to the camera is large compared with its depth along the optical axis direction, scaled orthographic projection is known to

be a good approximation of true perspective projection.

- Therefore, for these configurations, closed form calculations should also produce two solutions.

The single closed form solution to the P4P problem for coplanar points relies on foreshortening information from the perspective image to select one of the two poses. However, if the ratio of camera distance over object depth is large, this foreshortening may be smaller than the noise level. With a small amount of added random error in the image, the single exact analytic solution will flip to either pose and will have a good chance of ending with the wrong pose. Therefore, in such configurations, analytic methods that provide a single pose for coplanar points are not reliable and should probably be avoided.

This paper presents an iterative algorithm that performs equally well for short and long distance imagery. It is an application of our previous work [2] to the case of coplanar points. Starting the computation with a scaled orthographic projection approximation, the process is able to find two solutions that are both acceptable when the ratio of camera distance over object depth is large. In this case, only several iterations (correcting the effects of the scaled orthographic projection approximation) are necessary to converge on solutions that satisfy the perspective projection model.

If, on the other hand, the camera is close to the observed object, the image has strong perspective and the algorithm requires a few more iterations to converge on a single possible solution.

One may object that one is not better served by an algorithm that provides two equally probable poses than by an algorithm that chooses a single pose among these two poses and is wrong 50% of the time. However, consider an hypothetical computer vision system designed to assist a pilot in landing on an aircraft carrier; far from the carrier, the image of the runway may not contain enough information to allow a definite answer about the pose of the aircraft with respect to the runway. We contend that the algorithm providing two possible poses is more useful than the algorithm that is wrong 50% of the time; by providing two possible poses, the first algorithm effectively warns the system that more information is required

to lift the ambiguity, for example from an inertial sensor of the aircraft. This additional information can be used to reject one of the poses, and the system can then incorporate the other pose in its planning of the landing. With the algorithm providing a single pose, there is no warning that the pose may be wrong. Additional information can still be used to check the pose, but if the pose is rejected, the system is left with no pose information for its landing plan.

2 Notation

In Fig. 1, we show the classic pinhole camera model, with its center of projection O, its image plane G at distance f (the focal length) from O, its axes Ox and Oy pointing along the rows and columns of the camera sensor, and its third axis Oz pointing along the optical axis. The unit vectors for these three axes are called $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$. In this paper, the focal length and the intersection of the optical axis with the image plane (image center C) are assumed known.

An object with feature points M_0, $M_1, \ldots, M_i, \ldots, M_n$ is located in the field of view of the camera. The object coordinate frame of reference is $(M_0\mathbf{u}, M_0\mathbf{v}, M_0\mathbf{w})$. We call M_0 the reference point for the object. The coordinates (U_i, V_i, W_i) of the points M_i in the object coordinate frame of reference are known. The images of the points M_i are called m_i, and their image coordinates (x_i, y_i) are known. The coordinates (X_i, Y_i, Z_i) of the points M_i in the camera coordinate system are unknown, because the pose of the object in the camera coordinate system is unknown.

In the following, we show how to find the rotation matrix and translation vector of the object directly, without solving explicitly for the coordinates (X_i, Y_i, Z_i) of the points M_i. The approach implicitely uses the scaled orthographic projections p_i of the points M_i. To construct p_i, we draw a plane K through M_0 parallel to the image plane G. This plane is at a distance Z_0 from the center of projection O. The points M_i are projected on K at P_i by an orthographic projection. Then the points P_i are projected on the image plane G at p_i by a perspective projection. The same result would have been obtained if the object had been flattened into the plane K: approximating perspective projection with a scaled orthographic

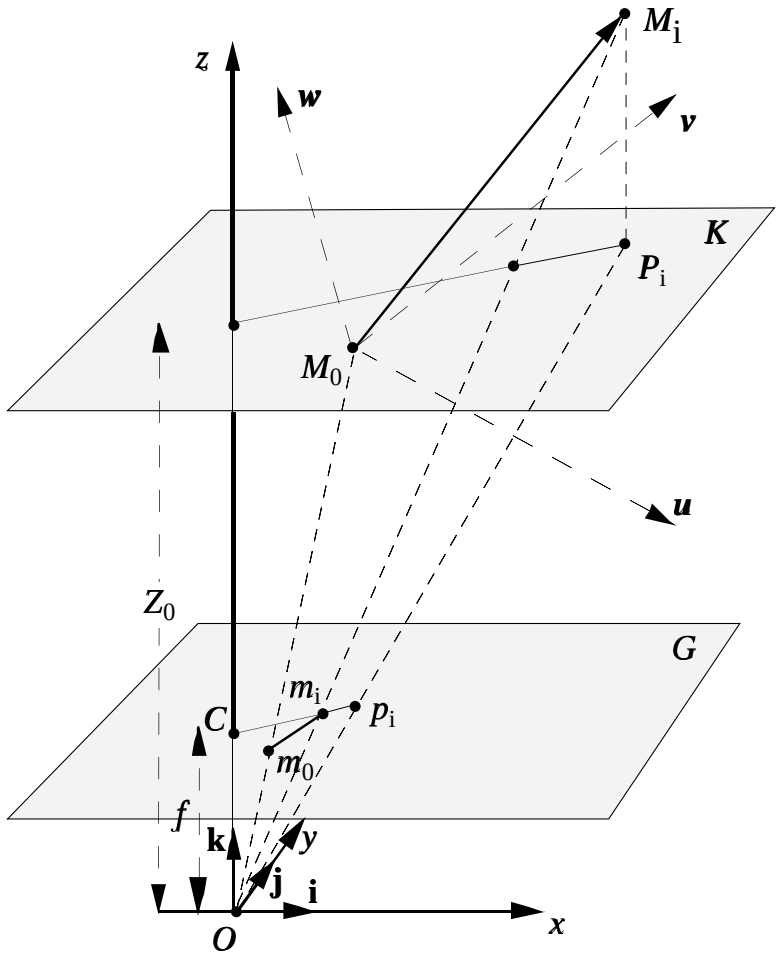


Figure 1: Perspective projection (m_i) and scaled orthographic projection (p_i) for an object point M_i and a reference point M_0.

projection amounts to assuming that the depths Z_i of different points M_i of the object with camera coordinates (X_i, Y_i, Z_i) are not very different from one another, and can all be set to the depth Z_0 of the reference point M_0 of the object.

3 Problem Definition

Our goal is to compute the rotation matrix $\mathbf{R}$ and translation vector $\mathbf{T}$ of the object. The rotation matrix $\mathbf{R}$ for the object is the matrix whose rows are the coordinates of the unit vectors $\mathbf{i}, \mathbf{j}, \mathbf{k}$ of the camera coordinate system expressed in the object coordinate system

(M_0u, M_0v, M_0w). The rotation matrix can be written as

$$\mathbf{R} = \begin{bmatrix} i_u\ i_v\ i_w \\ j_u\ j_v\ j_w \\ k_u\ k_v\ k_w \end{bmatrix} = \begin{bmatrix} \mathbf{i}^T \\ \mathbf{j}^T \\ \mathbf{k}^T \end{bmatrix}$$

To compute the rotation, we only need to compute $\mathbf{i}$ and $\mathbf{j}$ in the object coordinate system. The vector $\mathbf{k}$ is then obtained by the cross-product $\mathbf{i} \times \mathbf{j}$.

The translation vector, $\mathbf{T}$ is the vector $\mathbf{OM}_0$ between the center of projection, O, and the reference point M_0, the origin of the object coordinate frame of reference. Its camera coordinates are X_0, Y_0, Z_0. Since the image of M_0 is the known image point m_0, this translation vector $\mathbf{T}$ is aligned with vector $\mathbf{Om}_0$ and is equal to $\frac{Z_0}{f}\mathbf{Om}_0$. Therefore to compute the object translation, we only need to compute its z-coordinate Z_0. Thus the object pose is fully defined by $\mathbf{i}, \mathbf{j}, x_0, y_0$, and Z_0.

The relationship between the coordinates of feature points M_i in the camera and object coordinate systems can be expressed by

$$\begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} = \begin{bmatrix} \mathbf{i}^T \\ \mathbf{j}^T \\ \mathbf{k}^T \end{bmatrix} \begin{bmatrix} U_i \\ V_i \\ W_i \end{bmatrix} + \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix} \tag{1}$$

4 Fundamental Equations

We define an exact pose as an object pose for which the object points M_i fall on the lines of sight of the image points m_i. This condition can be expressed by the equalities

$$x_0 = f\frac{X_0}{Z_0}\,, \quad x_i = f\frac{X_i}{Z_i}$$

and similar equalities for the y coordinates. The second equality can be expanded by use of Eq. 1 into

$$x_i = f\frac{\mathbf{M}_0\mathbf{M}_i \cdot \mathbf{i} + X_0}{\mathbf{M}_0\mathbf{M}_i \cdot \mathbf{k} + Z_0}$$

A division of both terms of the fraction by Z_0 leads to

$$x_i = \frac{\mathbf{M_0M_i}\cdot\frac{f}{Z_0}\mathbf{i} + x_0}{\frac{1}{Z_0}\mathbf{M_0M_i}\cdot\mathbf{k} + 1}$$

Therefore, a necessary and sufficient condition for a pose defined by $\mathbf{i}, \mathbf{j}, x_0, y_0$, and Z_0 (where x_0 and y_0 define the location of the image of the object origin) to be an exact pose is that these quantities satisfy, for all points M_i, the equations

$$\mathbf{M_0M_i}\cdot\mathbf{I} = x_i(1+\varepsilon_i) - x_0, \tag{2}$$

$$\mathbf{M_0M_i}\cdot\mathbf{J} = y_i(1+\varepsilon_i) - y_0 \tag{3}$$

with

$$\mathbf{I} = \frac{f}{Z_0}\mathbf{i}, \quad \mathbf{J} = \frac{f}{Z_0}\mathbf{j}$$

$$\varepsilon_i = \frac{1}{Z_0}\mathbf{M_0M_i}\cdot\mathbf{k} \tag{4}$$

and $\mathbf{k} = \mathbf{i}\times\mathbf{j}$

5 POSIT Algorithm

We first note that in the right hand sides of the fundamental equations, the terms $x_i(1+\varepsilon_i)$ and $y_i(1+\varepsilon_i)$, are actually the coordinates x_i' and y_i' of the points p_i, which are the scaled orthographic projections of the feature points M_i (Fig. 1). Indeed, in the expression of ε_i, the dot product $\mathbf{M_0M_i}\cdot\mathbf{k}$ is the z-coordinate of $\mathbf{M_0M_i}$, $Z_i - Z_0$; therefore

$$(1+\varepsilon_i) = \frac{Z_i - Z_0}{Z_0} + 1 = Z_i/Z_0$$

Also, in perspective projection $x_i = fX_i/Z_i$. Therefore

$$x_i(1+\varepsilon_i) = fX_i/Z_0$$

The point p_i is the perspective projection of the point P_i which has same x coordinate X_i as M_i, and a z-coordinate equal to Z_0. Therefore the x-coordinate x_i' of p_i is precisely

$$x_i' = fX_i/Z_0$$

The basic idea behind the proposed method is that if values are given to ε_i, Eqs. 2 and 3 provide linear systems of equations in which the only unknowns are respectively the coordinates of $\mathbf{I}$ and $\mathbf{J}$. Once $\mathbf{I}$ and $\mathbf{J}$ have been computed, $\mathbf{i}$ and $\mathbf{j}$ are found by normalizing $\mathbf{I}$ and $\mathbf{J}$, and Z_0 is obtained from either the norm of $\mathbf{I}$ or $\mathbf{J}$. We call this algorithm, which finds an approximate pose by solving linear systems, *POS* (Pose from Orthography and Scaling). Indeed, finding the pose of the object by using fixed values of ε_i in Eqs. 2 and 3 amounts to finding the pose for which the points M_i have as scaled orthographic projections the image points p_i with coordinates $x_i(1+\varepsilon_i)$ and $y_i(1+\varepsilon_i)$, as we have just seen.

The solutions of the POS algorithm are only approximations if the values given to ε_i are not exact. But once the unknowns $\mathbf{i}$ and $\mathbf{j}$ have been computed, *more exact values can be computed for the* ε_i using Eq. 4, and the equations can be solved again with these better values. Initially, we set $\varepsilon_i = 0$. Assuming ε_i null implies $x'_i = x_i, y'_i = y_i$ and amounts to assuming that p_i et m_i coincide, i.e. that the image points are scaled orthographic projections of the object points. Fig. 2 describes this configuration. We call this iterative algorithm *POSIT* (POS with Iterations). This algorithm generally makes the values of $\mathbf{i}, \mathbf{j}$ and Z_0 converge toward values which correspond to a correct pose in a few iterations.

The iterative pose algorithm can be described by the following pseudocode:

1. $\varepsilon_{i(0)} = 0, n = 1$

2. Beginning of loop.
   Solve for $\mathbf{i}, \mathbf{j}$, and Z_0 using Eqs. 2 and 3 (see next section).
   When the object points are coplanar, the additional equality $\mathbf{i} \cdot \mathbf{j} = 0$ must be used, and two approximate poses are found.

3. Compute $\varepsilon_{i(n)} = \frac{1}{Z_0}\mathbf{M}_0\mathbf{M}_i \cdot \mathbf{k}$, with $\mathbf{k} = \mathbf{i} \times \mathbf{j}$. When the object points are coplanar, two sets of ε_i with opposite signs are found (see Section 8).

4. If $|\varepsilon_{i(n)} - \varepsilon_{i(n-1)}| <$Threshold, Exit.
   Else $n = n + 1$. Go to step 2.

5. Exact pose(s) = last approximate pose(s).

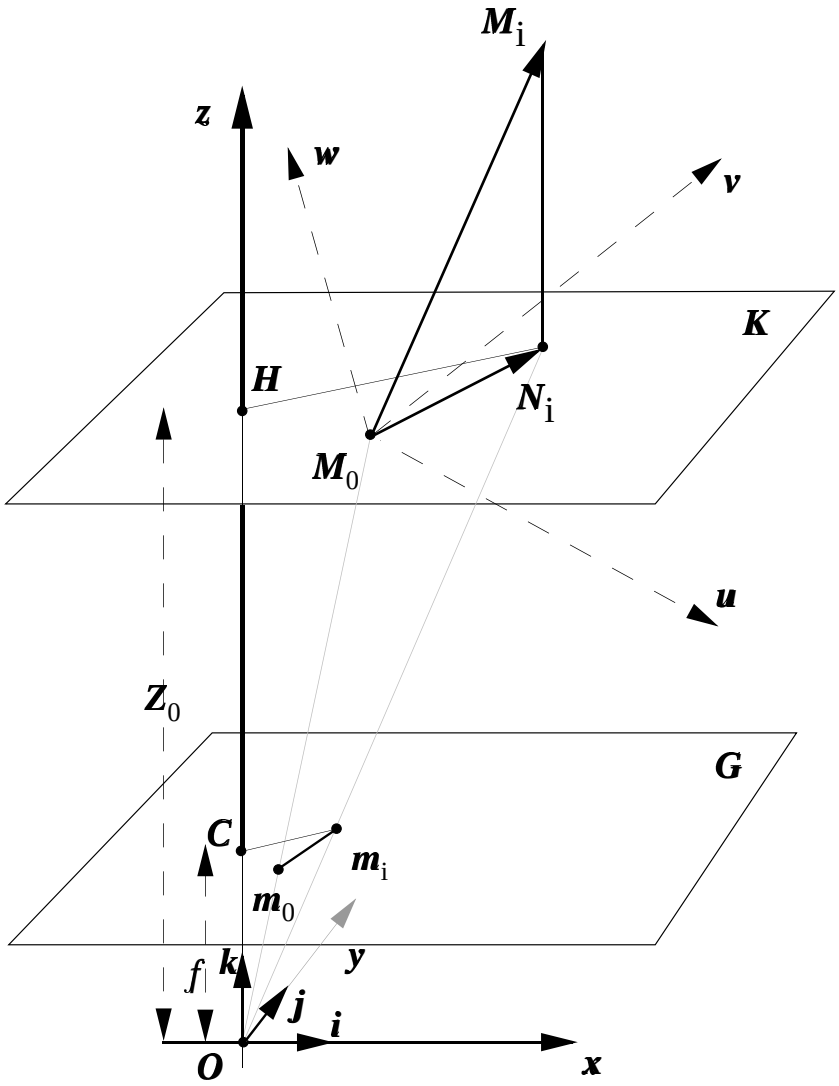


Figure 2: The initial loop in POSIT looks for a pose of the object such that the points m_i are the scaled orthographic projections of the object points M_i.

For a geometric interpretation of this iterative algorithm, see [2].

6 Solving the Systems of Equations (POS algorithm)

Within the iterative algorithm described in the previous section, we have to solve the equations

$$\mathbf{M}_0\mathbf{M}_i \cdot \mathbf{I} = x_i(1+\varepsilon_i) - x_0,$$

$$\mathbf{M}_0\mathbf{M}_i \cdot \mathbf{J} = y_i(1+\varepsilon_i) - y_0,$$

with

$$\mathbf{I} = \frac{f}{Z_0}\mathbf{i}, \mathbf{J} = \frac{f}{Z_0}\mathbf{j},$$

and the terms ε_i have known values at each iteration. We express the dot products of these equations in terms of vector coordinates *in the object coordinate frame of reference:*

$$[U_i\ V_i\ W_i][I_u\ I_v\ I_w]^T = x_i(1+\varepsilon_i) - x_0,\ \ [U_i\ V_i\ W_i][J_u\ J_v\ J_w]^T = y_i(1+\varepsilon_i) - y_0 \qquad (5)$$

These are linear equations where the unknowns are the coordinates of vector **I** and vector **J**. The other parameters are known: x_i, y_i, x_0, y_0 are the known coordinates of m_i and m_0 (images of M_i and M_0) in the camera coordinate system, and U_i, V_i, W_i are the known coordinates of the point M_i in the object coordinate frame of reference.

Writing Eq. 5 for the n object points $M_1, M_2, M_i, \ldots, M_n$ and their images, we generate linear systems for the coordinates of the unknown vectors **I** and **J**:

$$\mathbf{A}\,\mathbf{I} = \mathbf{x}',\ \ \mathbf{A}\,\mathbf{J} = \mathbf{y}' \qquad (6)$$

where **A** is the matrix of the coordinates of the object points M_i in the object coordinate frame of reference, $\mathbf{x}'$ is the vector with i-th coordinate $x_i(1+\varepsilon_i) - x_0$ and $\mathbf{y}'$ is the vector with i-th coordinate $y_i(1+\varepsilon_i) - y_0$. If we had at least three visible points other than M_0, and all these points were *noncoplanar*, matrix **A** would have rank 3, and the solutions of the linear systems in the least square sense would be given by

$$\mathbf{I} = \mathbf{B}\,\mathbf{x}',\ \ \mathbf{J} = \mathbf{B}\,\mathbf{y}' \qquad (7)$$

where **B** is the pseudoinverse of the matrix **A**.

We call **B** the *object matrix*. See [2] for details on the case of noncoplanar points. In this paper, we concentrate on the case where the object points are known to be *coplanar*. In this case, matrix **A** has rank 2, and the set of equations is ill-determined even with an overdetermined set of equations. Then additional constraints are required. We examine this degenerate situation using a geometric interpretation.

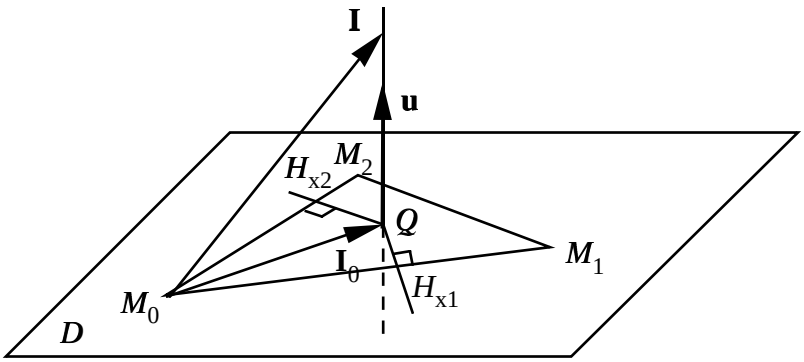


Figure 3: All vectors $\mathbf{I}$ whose head projects onto plane D in Q project onto $\mathbf{M}_0\mathbf{M}_1$ in H_{x1} and onto $\mathbf{M}_0\mathbf{M}_2$ in H_2.

7 A Geometric Point of View

We found the following equations for $\mathbf{I}$:

$$\mathbf{M}_0\mathbf{M}_i \cdot \mathbf{I} = x_i',$$

with $x_i' = x_i(1+\varepsilon_i) - x_0$. Geometrically, this expression states that if the tail of $\mathbf{I}$ is taken to be at $\mathbf{M}_0$, the head of $\mathbf{I}$ projects on $\mathbf{M}_0\mathbf{M}_i$ at a point H_{xi} defined by the algebraic measure

$$\overline{M_0 H_{xi}} = \frac{x_i'}{|\mathbf{M}_0\mathbf{M}_i|}$$

In other words, the head of $\mathbf{I}$ must belong to the plane perpendicular to $\mathbf{M}_0\mathbf{M}_i$ at H_{xi} (Fig. 3). If the object had four noncoplanar feature points, M_0, M_1, M_2, M_3, then $\mathbf{I}$ would be completely defined as the vector with its tail at M_0 and its head at the intersection of the three planes perpendicular to $\mathbf{M}_0\mathbf{M}_1$, $\mathbf{M}_0\mathbf{M}_2$ and $\mathbf{M}_0\mathbf{M}_3$ at H_{x1}, H_{x2} and H_{x3} respectively. Analytically, we would solve a system of three equations, and the matrix of the system would have rank 3.

However, if the feature points belong to the same plane D (but are not aligned), then the vectors $\mathbf{M}_0\mathbf{M}_1$, $\mathbf{M}_0\mathbf{M}_2$, etc. are coplanar and the planes perpendicular to them at H_{x1}, H_{x2}, etc. (defined above) all intersect at a single line or at close parallel lines that are perpendicular to plane D. The rank of matrix $\mathbf{A}$ is only 2. The simplest of such configurations occurs when we use only a triangle of features $M_0M_1M_2$. The pseudoinverse solution (7) of the system is a point Q also located in plane D which minimizes its distance

to the planes, and we call the corresponding vector solution $\mathbf{I}_0$ (Fig. 3). Clearly, this solution to the system is not unique, since all vectors $\mathbf{I}$ whose heads project onto plane D at Q still project onto $\mathbf{M}_0\mathbf{M}_1$ at H_{x1}, onto $\mathbf{M}_0\mathbf{M}_2$ at H_{x2}, etc.. In other words any vector $\mathbf{I}$ with its tail at $\mathbf{M}_0$ and its head on the line perpendicular to plane D at Q is a solution (Fig. 3).

8 System Solutions for Coplanar Points

Such solutions can be written as

$$\mathbf{I} = \mathbf{I}_0 + \lambda \mathbf{u} \tag{8}$$

where $\mathbf{u}$ is a unit vector normal to D and λ is the coordinate of the head of $\mathbf{I}$ along $\mathbf{u}$. Similarly,

$$\mathbf{J} = \mathbf{J}_0 + \mu \mathbf{u} \tag{9}$$

and λ and μ are unknown. Since the vector $\mathbf{u}$ is normal to the plane D of the object, we have $\mathbf{M}_0\mathbf{M}_i \cdot \mathbf{u} = 0$. Thus the vector $\mathbf{u}$ is the unit vector of the null space of the matrix $\mathbf{A}$. It can be obtained as the column vector corresponding to the smallest singular value in the second orthonormal matrix provided by the singular value decomposition of $\mathbf{A}$ [9]. This method of computing $\mathbf{u}$ provides a mean direction, and seems to be useful in cases where the object points are not exactly coplanar. This computation is performed only once for a given distribution of points, at the same time as the computation of the object matrix $\mathbf{B}$.

In contrast to the general case of noncoplanar feature points [2], we now have to use the additional fact that $\mathbf{I}$ and $\mathbf{J}$ must (1) be perpendicular and (2) be of the same length, in order to compute the unknowns λ and μ. The first condition yields

$$\lambda \mu = -\mathbf{I}_0 \cdot \mathbf{J}_0,$$

and the second condition yields

$$\lambda^2 - \mu^2 = \mathbf{J}_0^2 - \mathbf{I}_0^2$$

Huttenlocher [8] finds the same two equations for the restricted case of three object points using a completely different approach, and solves them by squaring the first equation and

eliminating one of the unknowns between the two conditions. Squaring an equation introduces new solutions, so that all solutions must be checked against the original equations. We propose an alternative method that does not require squaring the first equation.

We remark that the square of the complex number $C = \lambda + i\mu$ is $C^2 = \lambda^2 - \mu^2 + 2i\lambda\mu$, i.e.,

$$C^2 = \mathbf{J}_0^2 - \mathbf{I}_0^2 - 2i\mathbf{I}_0 \cdot \mathbf{J}_0$$

Therefore we can find λ and μ as the real and imaginary parts of the square roots of the complex number C^2. Finding the square roots requires writing C^2 in polar form:

$$C^2 = [R, \Theta], \text{ with}$$

$$R = ((\mathbf{J}_0^2 - \mathbf{I}_0^2)^2 + 4(\mathbf{I}_0 \cdot \mathbf{J}_0)^2)^{1/2}, \text{ and}$$

$$\Theta = \text{Arctan}(\frac{-2\mathbf{I}_0 \cdot \mathbf{J}_0}{\mathbf{J}_0^2 - \mathbf{I}_0^2}), \text{ if } \mathbf{J}_0^2 - \mathbf{I}_0^2 > 0, \text{ and}$$

$$\Theta = \text{Arctan}(\frac{-2\mathbf{I}_0 \cdot \mathbf{J}_0}{\mathbf{J}_0^2 - \mathbf{I}_0^2}) + \pi, \text{ if } \mathbf{J}_0^2 - \mathbf{I}_0^2 < 0$$

$$(\text{if } \mathbf{J}_0^2 - \mathbf{I}_0^2 = 0, \text{ we have } \theta = -\text{Sign}(\mathbf{I}_0 \cdot \mathbf{J}_0)\frac{\pi}{2}, \text{ and } R = |2\mathbf{I}_0 \cdot \mathbf{J}_0|)$$

There are two roots C for this number, $C = [\rho, \theta]$, and $C = [\rho, \theta + \pi]$, with

$$\rho = \sqrt{R}, \text{ and } \theta = \frac{\Theta}{2}$$

λ and μ are the real and imaginary parts of these numbers

$$\lambda = \rho\cos\theta, \quad \mu = \rho\sin\theta, \text{ or}$$

$$\lambda = -\rho\cos\theta, \quad \mu = -\rho\sin\theta$$

These values yield the two solutions for $\mathbf{I}$ and $\mathbf{J}$

$$\mathbf{I} = \mathbf{I}_0 + \rho(\cos\theta)\mathbf{u}, \quad \mathbf{J} = \mathbf{J}_0 + \rho(\sin\theta)\mathbf{u}, \text{ and}$$

$$\mathbf{I} = \mathbf{I}_0 - \rho(\cos\theta)\mathbf{u}, \quad \mathbf{J} = \mathbf{J}_0 - \rho(\sin\theta)\mathbf{u}$$

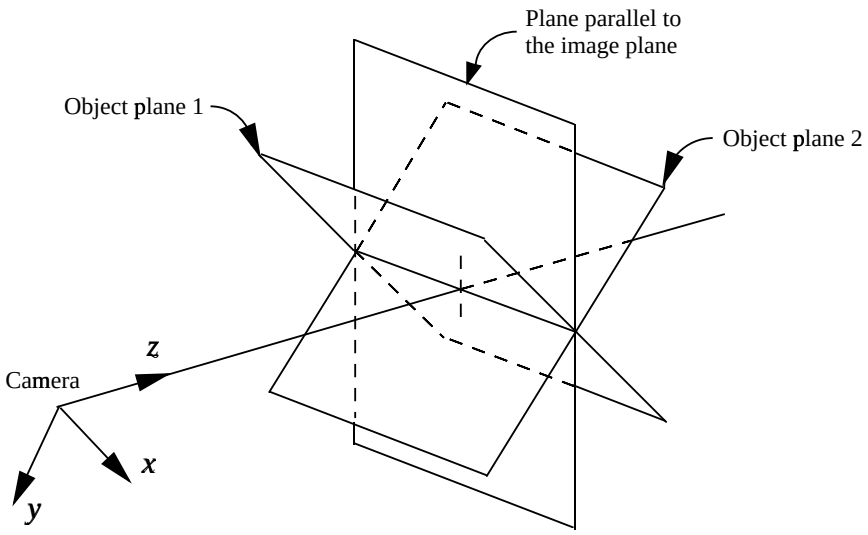


Figure 4: Two object poses giving the same image under SOP approximation.

Notice that since $\mathbf{u}$ is perpendicular to the object plane, the above solutions show that the pair $(\mathbf{I}, \mathbf{J})$ of the first solution is symmetrical to the pair $(\mathbf{I}, \mathbf{J})$ for the second solution. If we anchor our point of view to the camera, this is equivalent to the observation that the two solutions for the pose of the object plane are symmetrical with respect to a plane parallel to the plane $(\mathbf{I}, \mathbf{J})$, in other words symmetrical with respect to a plane parallel to the image plane (see Fig. 4).

9 From System Solutions to Approximate Pose Solutions

Next we use these two solutions for $\mathbf{I}$ and $\mathbf{J}$ to find the corresponding rotation matrices and translation vectors. First we find the depth Z_0 of the reference point M_0, by dividing the focal length by the norm of $\mathbf{I}$ or $\mathbf{J}$; note that the solutions for $\mathbf{I}$ and for $\mathbf{J}$ all have the same norm, because we imposed the condition $|\mathbf{I}| = |\mathbf{J}|$. Therefore there is a unique solution Z_0. Then we get $X_0 = \frac{Z_0}{f} x_0$, $Y_0 = \frac{Z_0}{f} y_0$. Thus, there is a single translation solution. However there are two rotation matrices corresponding to the two solutions for $\mathbf{I}$ and $\mathbf{J}$ (for each set, normalizing $\mathbf{I}$ and $\mathbf{J}$ yields the first two rows of a rotation matrix and the last row is obtained by the cross product of the first two rows). These two solutions correspond to the two symmetrical positions of the plane of object points, with respect to a plane parallel to the image plane, that lead to the same image (see Fig. 4). However, they may not both

be feasible. We have to verify, for each pose, that all the object points are in front of the camera (all $Z_i > 0$). If it is not the case, the pose has to be discarded.

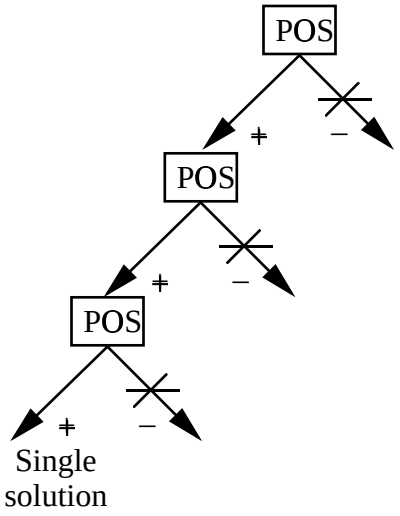


Figure 5: Case 1: POS yields a single feasible pose at each step of the process (+: feasible pose, −: unfeasible pose).

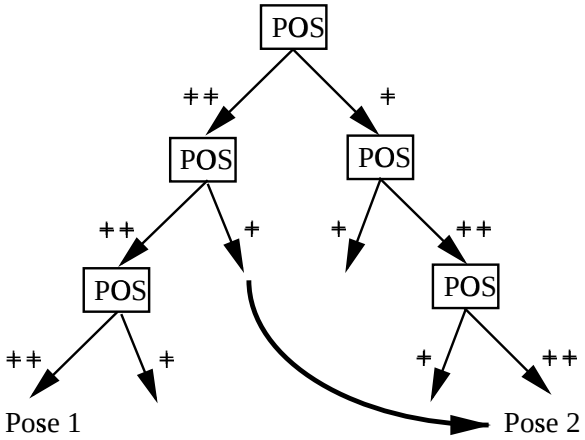


Figure 6: Case 2: POS yields two feasible poses at each step of the process (++: best quality, +: lower quality pose).

10 Iterating from Approximate Pose to Exact Pose

For coplanar scene points, the POS algorithm produces two poses at each iteration of the POSIT algorithm. Therefore it would seem at first that we would need to explore a tree of poses and might end up with 2^n poses after n iterations (see Figs. 5 and 6 for illustrations of these trees). However, in practice we find that we have to only follow one or two branches, and end up with one or two feasible solutions. Two situations occur:

- In the first situation (see Fig. 5), at the first iteration step we compute the two poses, but find that one of the poses is not feasible and has to be discarded because some scene points are placed behind the camera in that pose. This situation is then found to occur at every subsequent step. Therefore we are left with a single path to follow and convergence to a single feasible pose.

- In the second situation (see Fig. 6), both poses of the first iteration step are feasible, and we pursue the iterations for both branches. At the second step, each branch still

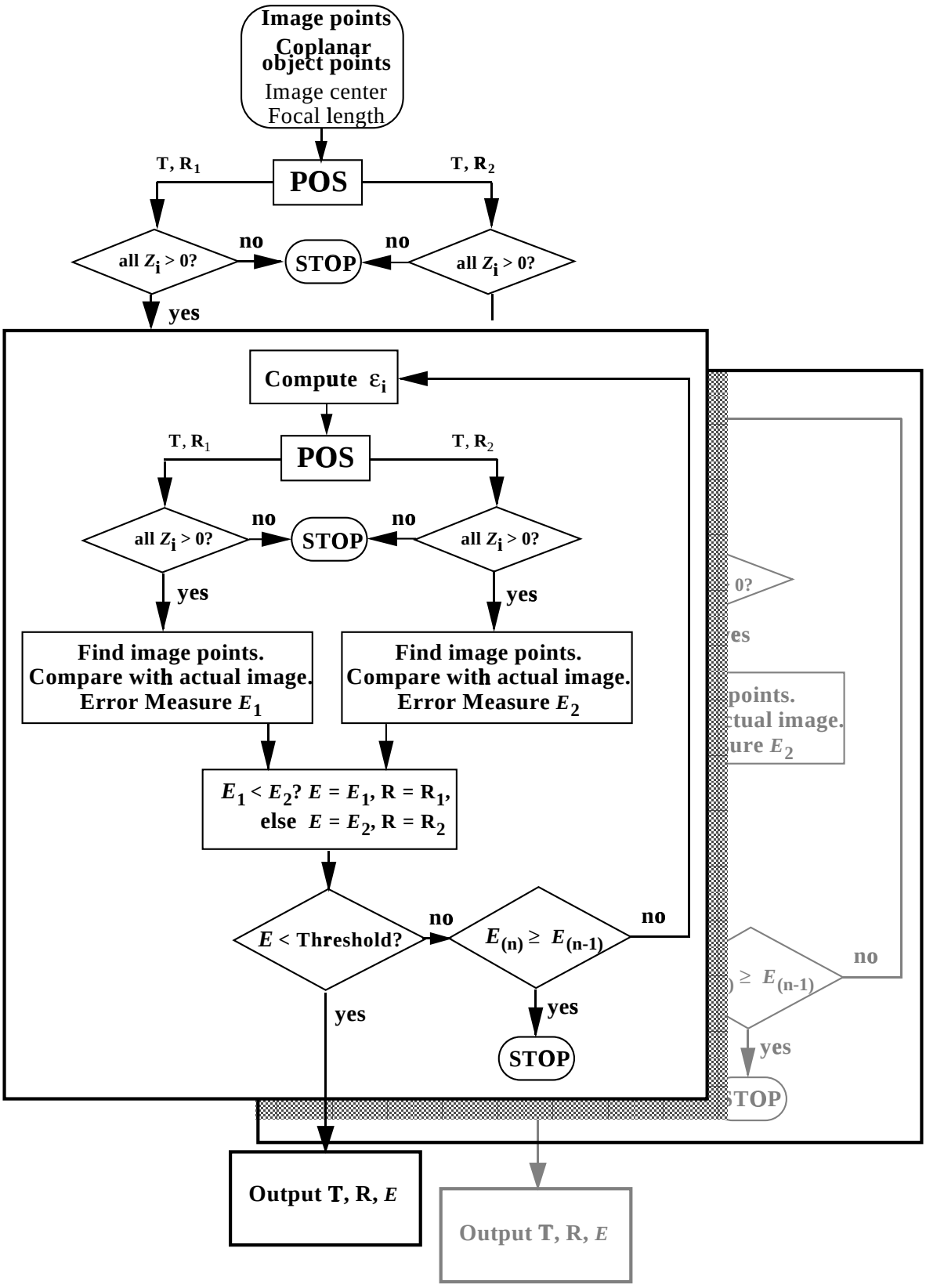


Figure 7: The POSIT algorithm for coplanar scene points.

provides two feasible poses, but for each branch *we keep only the better pose*. This strategy is justified by the fact that exploring a lower quality branch would not be fruitful: experiments show that either we would end up with one of the same two poses that this strategy would produce (but at a much slower convergence rate) or we would have to stop the iteration because the process diverges. The exploration of a lower quality branch is illustrated by the curved arrow in Fig. 6.

The measure of quality we use to select the better pose is the distance measure E, the average distance between actual image points and projected points from the computed pose.

The flow chart for the POSIT algorithm for coplanar scene points is shown in Fig. 7. It shows the two branches produced at the first iteration step. One branch may have to be dropped if the z-components of some scene points for the corresponding pose are negative (points behind camera). At the second and following iteration steps, the processes for each branch are similar, and only the process for the left branch is shown in the foreground of the figure. This process includes choosing the better of two poses at each iteration, and checking if the distance measure E has fallen below a threshold predefined in relation to the estimated noise in the image. If this is the case, the pose for the branch is output, along with the final distance measure E; otherwise the pose is used at the next iteration loop to recompute the ε_i. An example of computation with four coplanar points in a configuration where both poses are acceptable can be found in Appendix.

11 Performance Characterization

In this section, we evaluate the orientation and position errors of the POSIT algorithm for a planar object or scene; in photogrammetric applications, the object is the scene in front of the camera. We consider two objects with coplanar feature points, at four distance ratio from the camera. The distance ratio is defined as the ratio of the distance from the camera to the object, over the size of the object. For each object, synthetic images are obtained using a number of azimuth and elevation angles for the camera and three levels of image noise. The camera poses are computed by POSIT from these images and compared with the actual camera poses, and errors in position and orientation are plotted against the elevation

angles of the camera for various distances from the camera to the object along the optical axis. We also study the number of solutions found by the algorithm for each configuration, the effect of errors in positioning the image center, as well as the effect of assuming that an object is planar when it is not actually planar.

11.1 Objects

The first object comprises four coplanar points; two points are located at diagonally opposite corners of a 100m square, and the other two points are at arbitrary locations inside the square. The second object has ten coplanar points and is used in most of the tests; for this object, two points are also located at diagonally opposite corners of the 100 m square, and eight points are positioned randomly inside the square. Thus for both objects, the size used to compute the distance ratio is 100m. The origin O of the coordinate system $(O\mathbf{u}, O\mathbf{v}, O\mathbf{w})$ of the objects is located at the center of the square, and the plane $W = 0$ is the plane of the square (see Fig. 8).

11.2 Image Generation

We obtain synthetic images by perspective projection with a focal length of 760 pixels. Three levels of random noise in the image are specified. At image noise level 1, the real numbers computed for the coordinates of the perspective projections are rounded to integer pixel positions. At noise level 2, these integer positions are disturbed by vertical and horizontal perturbations of ± 1 pixel. These are created by a uniform random number generator. At noise level 3, the amplitude of the perturbations is ± 2 pixels. Note that when the camera is at a distance ratio of 20 from the object and at a 10 degree elevation, the image of the object extends over as few as 30 pixels horizontally and 6 pixels vertically: a perturbation of two pixels on each side of the image is relatively large in comparison to the size of the projected object.

11.3 Camera Poses

The camera always points toward the origin of the object coordinate frame. It is successively located at four distance ratios from the origin: 2, 5, 10 and 20.

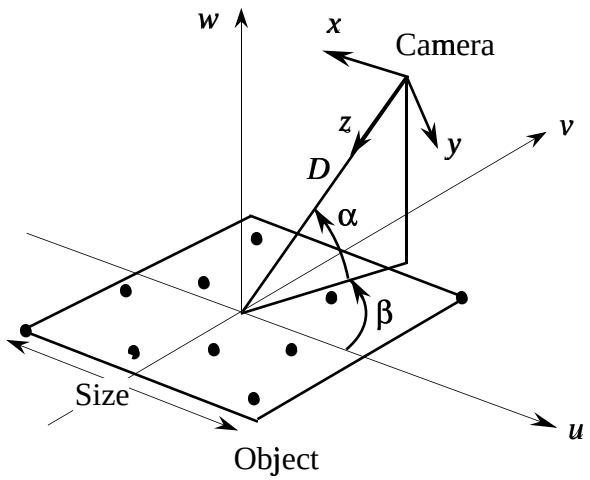


Figure 8: Elevation angle α and azimuth angle β for the camera in the experiments.

For each of these distance ratios, 17 elevation angles, from 10 to 90 degrees, are considered. These camera elevation angles are denoted by α in Fig. 8. For the last elevation angle, the camera is at nadir. All the pose errors are plotted against these elevation angles.

Fig. 8 also shows the camera azimuth angle, denoted by β. In most tests, our goal is to obtain results that reflect the performance of the method itself rather than the camera azimuths and the distribution of the object points. Therefore, we take the average of the pose errors obtained for 72 camera azimuths β, in 5 degree increments. We display these average errors and their standard deviation bars against the 17 elevation angles for the three noise levels and the four camera distance ratios.

We also study the occurence of double solutions as a function of the camera elevation for the four camera distance ratios and three noise levels. For these tests, we choose a single camera azimuth equal to zero, and we plot for each elevation the probability of obtaining a double pose solution by examining the proportion of double solutions obtained for 72 noisy images.

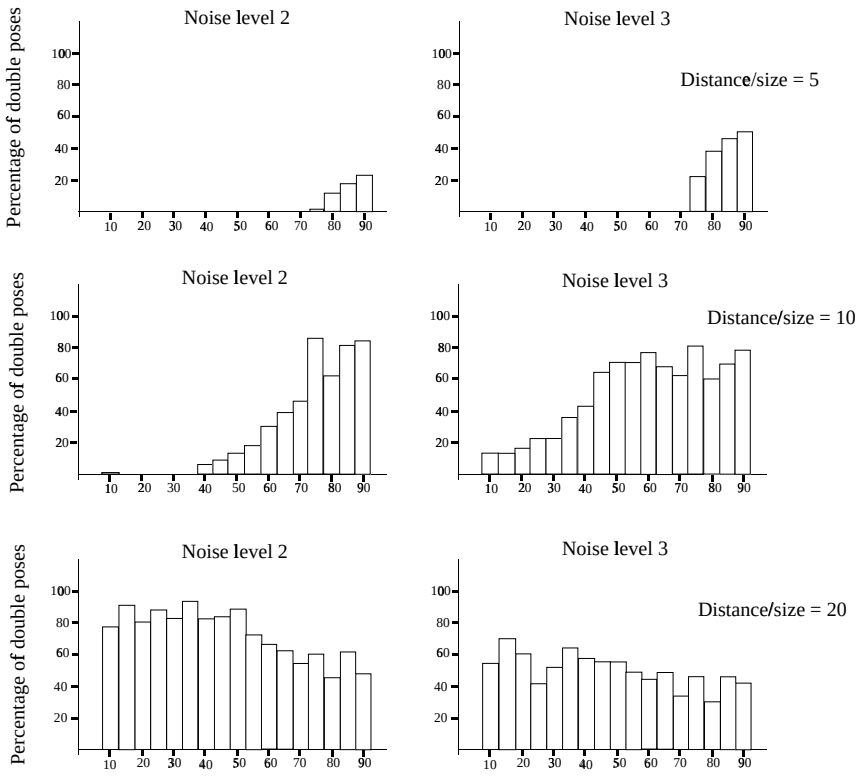


Figure 9: Probabilities of obtaining two acceptable poses, plotted against elevation angles α, for four coplanar points.

11.4 Number of Acceptable Poses

We now look at the number of "acceptable" poses found by the algorithm. For these experiments the camera is pointed toward the object at various elevation angles at a single azimuth, but for each elevation, 72 random image noise configurations are generated. A pose is called *acceptable* according to the following definition: we consider the offsets between image points and projected object points; the pose is acceptable is all these offsets are smaller than the amplitude of the image noise (1.5 pixel along x and y for level 2, and 2.5 pixels for level 3). This is simply an acknowledgement of the fact that, in practice, one cannot know whether an offset between the image points and the projections of the feature points for the computed pose is due to actual image points shifted by the image noise or to projected feature points shifted by a poor pose, and one has to give the benefit of the doubt to the pose computation if the offsets fall within the level of the image noise.

The diagrams of Figs. 9 and 10 present results in the forms of percentages of double pose

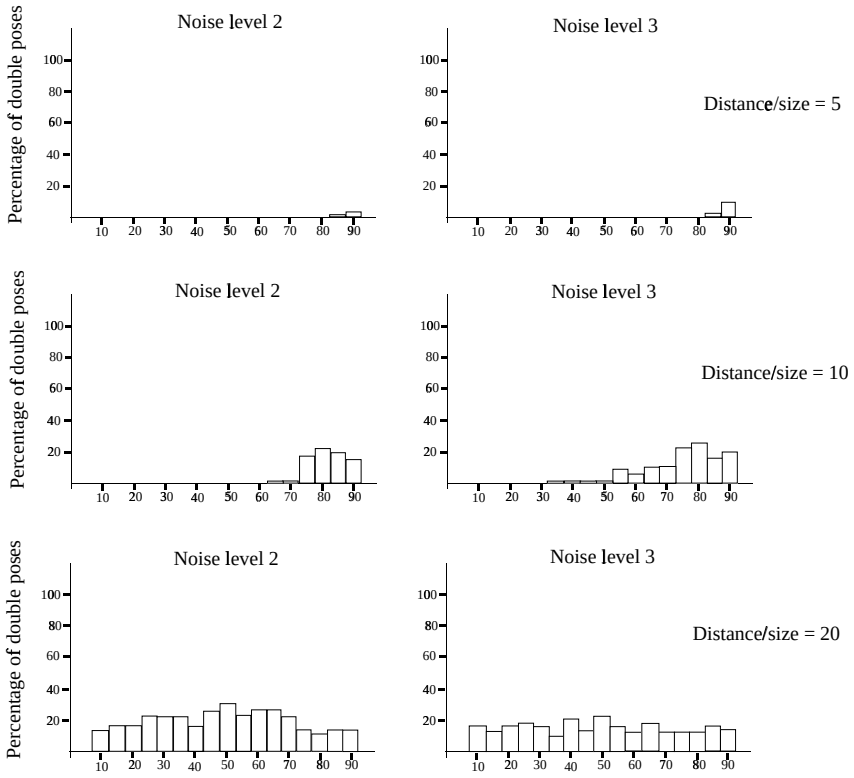


Figure 10: Probabilities of obtaining two acceptable poses, plotted against elevation angles α, for ten coplanar points.

solutions for elevation angles from 10 to 90 degrees. Each data is the percentage of occurrence of two pose solutions for 72 noisy images obtained by following the procedure described in Section 11.2. These results can be interpreted as probabilities of finding two acceptable poses instead of a single pose. In these diagrams, only noise level 2 and noise level 3 are shown. Noise level 1 is less interesting and is not shown because it is deterministic quantization noise; for this type of noise, the 72 noisy images used to compute the percentages are identical and all provide the same poses, so that the percentage of occurence is 100% when two acceptable poses are found and 0% otherwise.

We see from these diagrams that there is more chance to find two poses when the ratio of camera distance over the depth of the object along the optical axis is large. Indeed when this condition applies, we know that scaled orthography is a good approximation to perspective, and that with this approximation there are always two solutions for coplanar feature points. This condition is verified when the object is far from the camera, when the camera faces the object (then the dimension along the optical axis is zero and the ratio is infinite), or for a

combination of a moderate distance and a non-grazing incidence angle. We find two poses for the shortest distance ratios only when the camera nearly faces the object. For intermediary distance ratios, we start finding two poses for intermediary angles; for the largest distance ratio, the probability of finding two poses is practically independent of the camera angle.

Note that the occurrence of double solutions does not necessarily increase with the image noise level. The reason may be that two mechanisms with opposite effects are at play. On one hand, the larger the noise, the easier it is for a pose to be acceptable (see definition of acceptable poses in the beginning of this section). On the other hand, the larger the noise, the more distorted the image may be, and the more offset the pose may be.

Comparing the diagrams of percentages for four points and ten points, we see that the probabilities of obtaining two acceptable poses are dramatically higher for four points than for ten points. We have not found a convincing explanation for this difference. One of the advantages of performing multiple experiments on synthetic data (more than 14,000 experiments for these two diagrams) is that it brings to light properties which would have been difficult to predict analytically or discover from a few experiments with real data.

11.5 Computation of Pose Errors

For each pose estimated by the algorithm, the position and orientation of the camera with respect to the object are compared to the actual position and orientation used to obtain the image. We compute the axis of rotation to align the coordinate system of the object in its actual orientation with the coordinate system of the object in its orientation computed by POSIT with respect to the camera coordinate system. The *orientation error* is defined as the rotation angle in degrees around this axis required to achieve this alignment. The *position error* is defined as the norm of the translation vector required to align the actual and computed positions of the origin of the object coordinate frame. This distance is normalized by the distance between the origin of the camera coordinate frame and the origin of the object coordinate frame. Thus the position error is a relative error, whereas the orientation error is a measure in degrees.

11.6 Average Pose Errors for the Ten-Point Object

The plots of Fig. 11 and Fig. 12 show the orientation and position errors of the camera obtained by averaging data for 72 azimuth angles around the ten-point object.

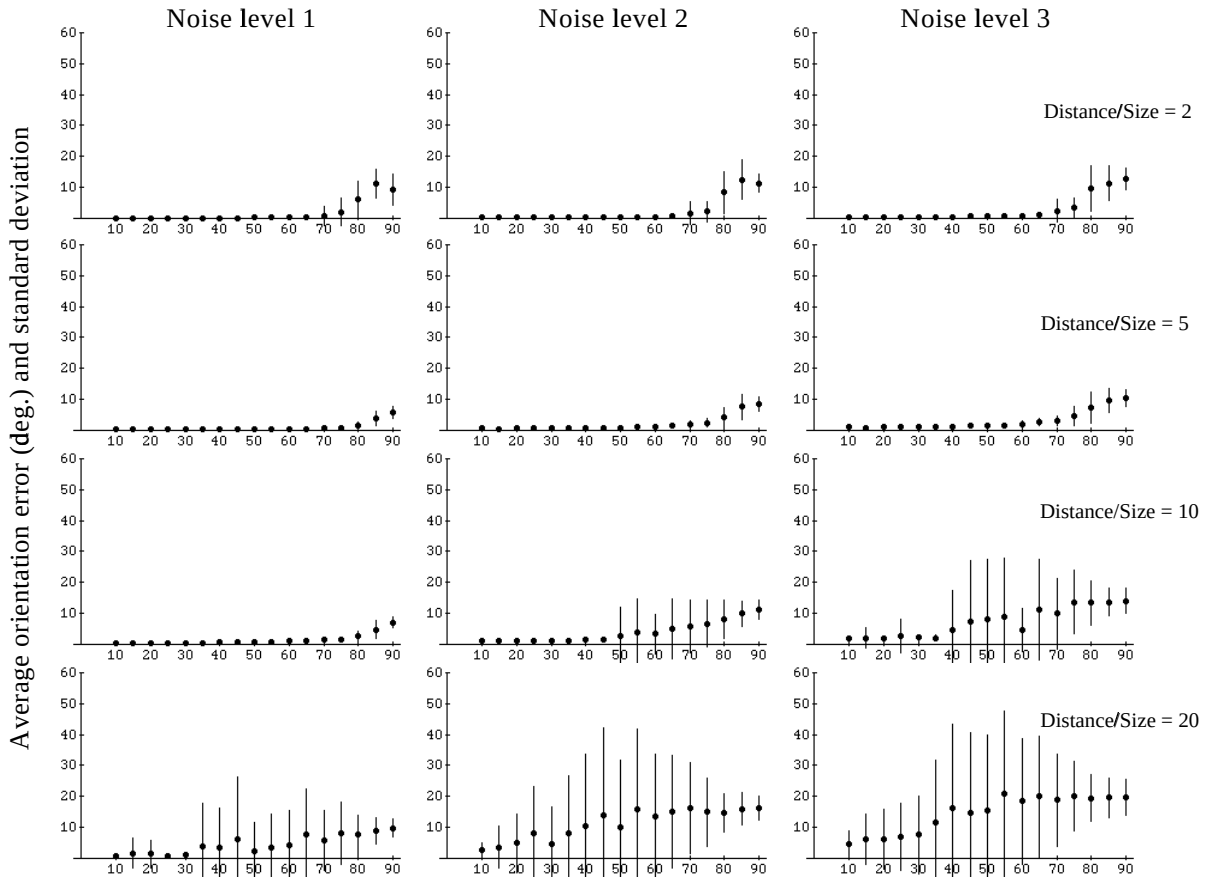


Figure 11: Average orientation errors and standard deviations against elevation angles α for ten coplanar feature points. These results are obtained from 72 azimuth angles β.

For camera distance ratios up to 10 and elevation angles up to 35 degrees, the orientation errors are typically less than 3 degrees even for the largest image noise. The largest orientation errors occur for the largest image noise level when the elevation is close to nadir. The orientation computations are then very sensitive to image noise. Indeed, at nadir, all the rotations around axes in the plane of the object displace the feature points in directions which are close to the directions of the lines of sight, and are difficult to detect because they produce few changes in the image. Only rotations around axes parallel to the optical axis are easy to detect. The points on the rotation error plots reflect the average interpretations

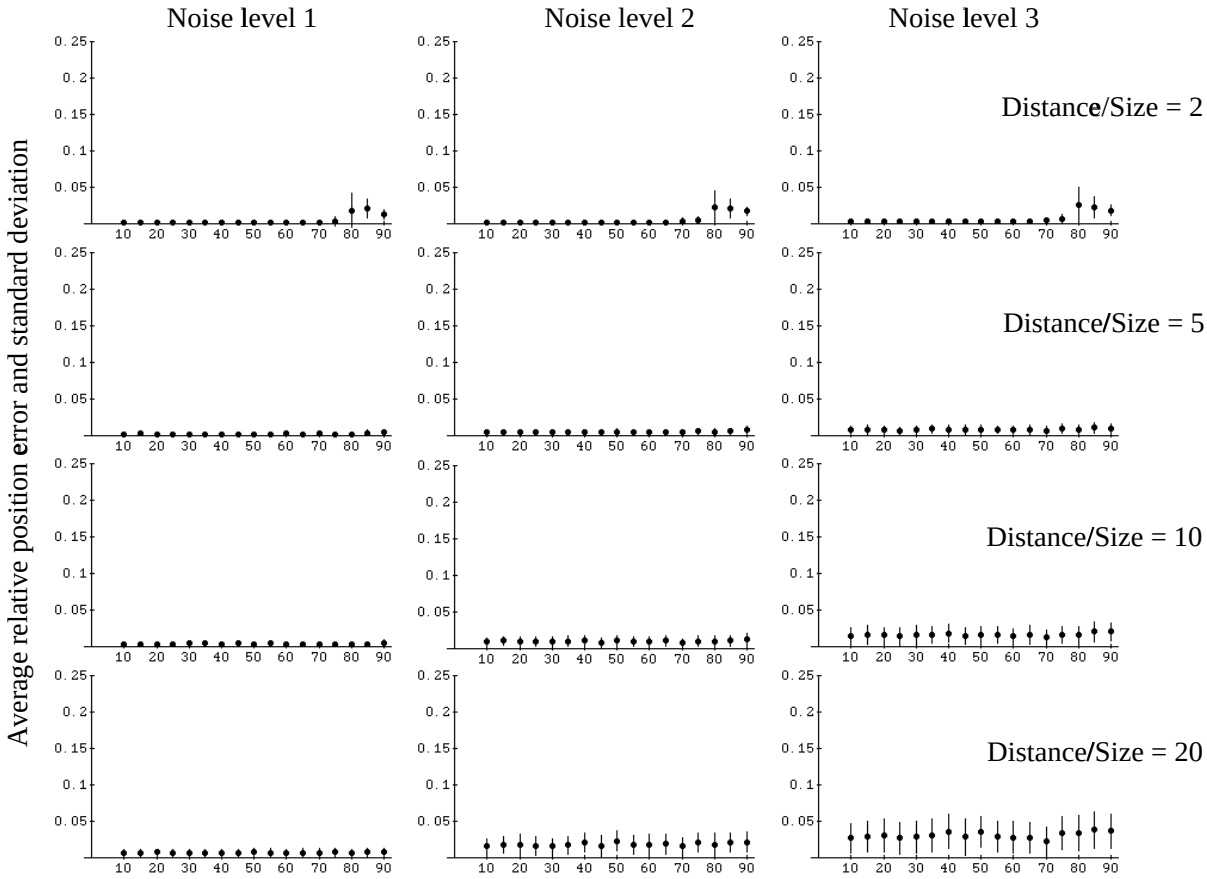


Figure 12: Average position errors and standard deviations against elevation angles α for ten coplanar feature points. These results are obtained from 72 azimuth angles β.

of errors from 72 images, and the chances that small shifts in the image points be interpreted by large shifts in the pose angle are large. This does not happen for grazing views of the object, where most rotations displace the feature points along directions normal to their lines of sight.

Fig. 12 shows that the position results are less sensitive to image noise at nadir than the orientation results, with position errors always under 6%, even at nadir at a distance ratio of 20 with the largest image noise level. We can explain this from the fact that only translation in the direction of the optical axis is difficult to detect from the image for this pose. Then the chances that small shifts in image points be interpreted by large shifts in pose angle are small.

11.7 Other simulations

The map used as a model often does not provide altitude information; then the feature points may be modeled as coplanar, whereas the actual geometry may not be planar. Also, with aerial imagery, the spread of feature points may be large compared with the elevations of the points. Even if the map shows that the ground is not planar, these feature points should be considered coplanar if the matrix describing the geometry of feature points can be considered of rank 2 instead of 3. Experiments were performed to estimate the effect of this type of modelling approximation. Both average position and orientation error increase slightly with the default of planarity, but the simulations show that the algorithm is not very sensitive to this type of input error. For example, if we consider a 100×100m scene, with actual elevations up to 15m (top of houses) for the scene points, the use of a street map with no indications of altitudes will typically generate a position error of 10m when the camera is 500m away from the scene (a relative error of 2%).

We also studied the influence of uncertainty in image center position. Indeed, this position is not always available; printed images may be peripheral parts of original photographs. The simulations show that the computation of the camera orientation is almost insensitive to the shifts. For the camera position, a shift of the image center by (20,20) pixels generates an average relative camera position error of 4% (i.e. 20m for a scene of 100m seen at 500m). A lateral image translation is interpreted as a small lateral translation of the object. Details and diagrams for these simulations are provided in [10].

12 Experiments on Real Images

We present results using the aerial photograph shown in Fig. 13. This picture is a view of the Mall in Washington, DC, taken from the top of the Washington Monument. For verification of pose results, the monument position is available on a topographic map of known scale (Fig. 14). The camera elevation with respect to the assumed plane of the area was estimated from the known altitude of the monument and the elevation of the ground at this location. With respect to a scene coordinate frame (O, u, v, w) of our choice, the position of the camera

was

$$U = -2276 \text{ m};\ V = -31 \text{ m};\ W = 159 \text{ m}$$

Note that this position is approximate because of the low precision of the map and the uncertainty on the elevation. The image center was assumed to be the center of the picture. (C, x, y) is an image coordinate frame centered on this estimated image center. The orientation parameters of the camera were not recorded during the snapshot, but they can be computed by observing the position of the intersection of the optical axis with the assumed plane of the ground.

Ten feature points, numbered 1 to 10 in Fig. 13, were chosen in the image and located on the map. Their coordinates were measured in both frames. The threshold of the convergence test (see Fig. 7) was adjusted for each experiment in order to obtain single pose estimations, as refined as possible.

The algorithm, applied to the first four points, generated a single solution with the error measure $E_1 = 0.13\%$. This means that for the estimated pose, the reconstructed image points would be at an average distance of 0.2 mm from the actual image points shown in Fig. 13. This is quite good considering the uncertainty of the feature points' locations in the original image. The corresponding computed position of the camera is

$$U = -2279 \text{ m};\ V = -27 \text{ m};\ W = 161 \text{ m}$$

Comparing this computed position with the position estimated from the map and the monument height we find

$$\Delta U = 3 \text{ m};\ \Delta V = 4 \text{ m};\ \Delta W = 2 \text{ m}$$

which is surprisingly good considering the low precision of the map.

Note, however, that the control points were chosen in a favorable configuration (they cover a large portion of the scene). For example, the results degrade if we use points 3, 4, 6 and 8. We obtain $E_1 = 0.14\%$ and

$$\begin{aligned} U &= -2277 \text{ m}, & \Delta U &= 1 \text{ m} \\ V &= 9 \text{ m}, & \Delta V &= 40 \text{ m} \\ W &= 163 \text{ m}, & \Delta W &= 4 \text{ m} \end{aligned}$$

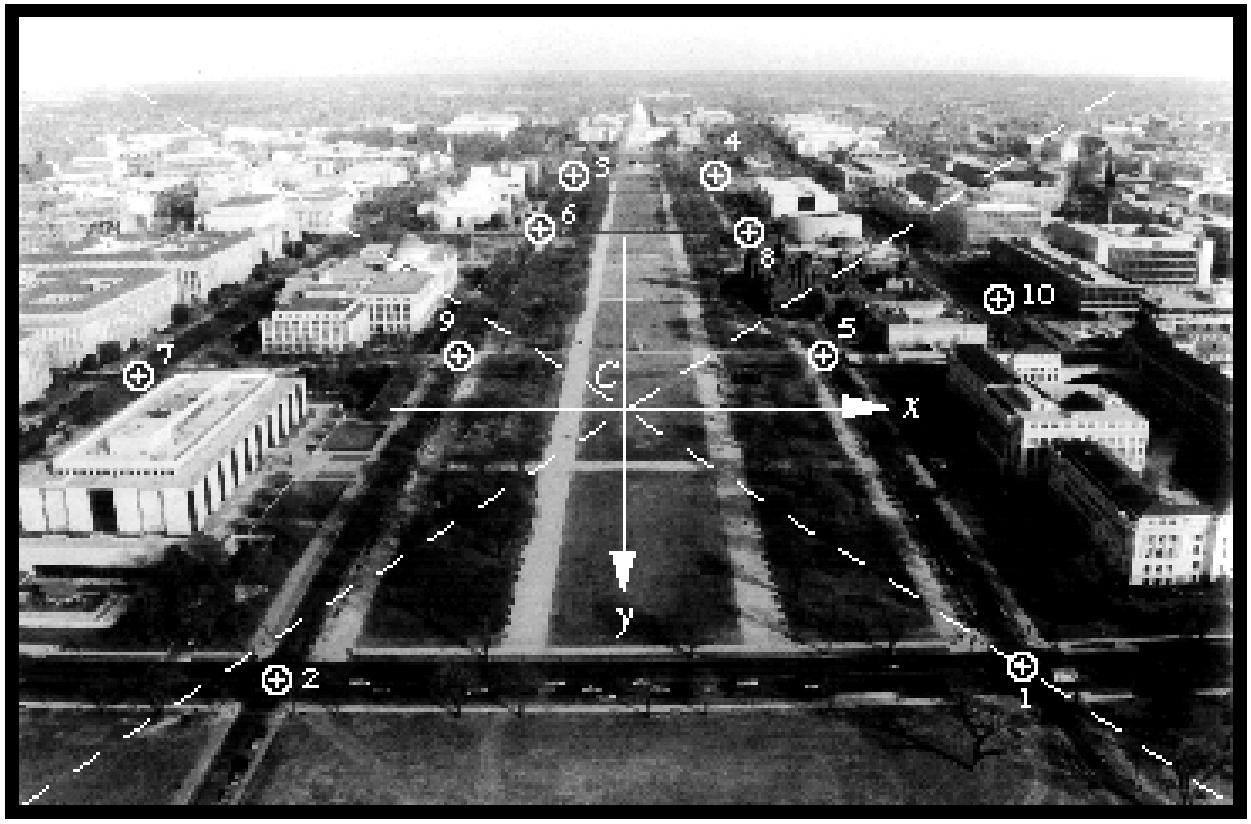


Figure 13: The Mall in Washington, DC, from the top of the Washington Monument, with feature points and image coordinate system.

It is interesting to note that with points 3, 4, 6 and 8 (more distant from the camera), we can obtain a second solution. But the corresponding error measure for this solution is quite high: $E_2 = 2.31\%$. Furthermore, this pose must be rejected because object points 1 and 2 would be behind the camera.

An advantage of the algorithm is that it can deal with more than four control points. With all ten points, we find $E\,(= E_1) = 0.58\%$ and

$$\begin{aligned} U &= -2266 \text{ m}, \quad & \Delta U &= 10 \text{ m} \\ V &= -24 \text{ m}, & \Delta V &= 7 \text{ m} \\ W &= 152 \text{ m}, & \Delta W &= 7 \text{ m} \end{aligned}$$

Using a number of points larger than the minimum required (four points) provides a least square estimate of the pose, based on all the available information.

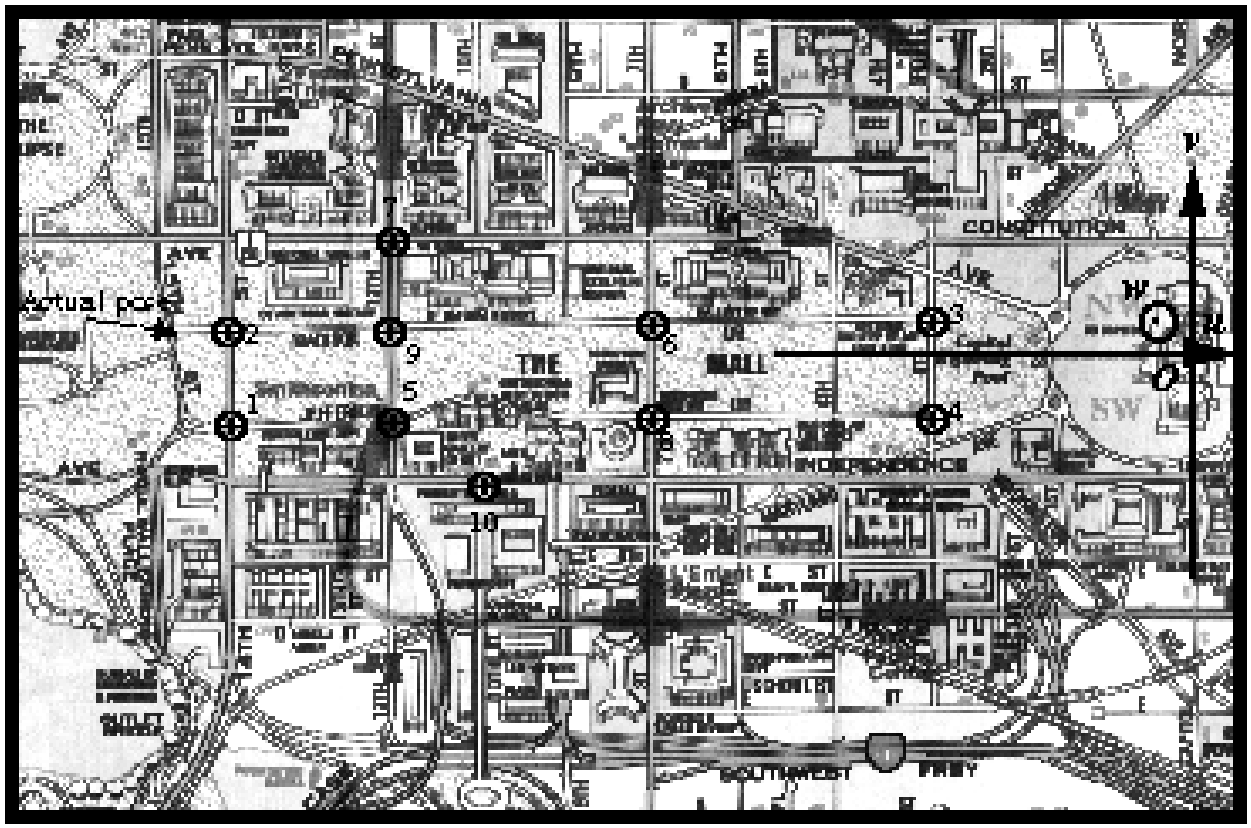


Figure 14: Topographic map of the area, with feature points and object coordinate system.

13 Conclusion

We have presented an iterative method for the computation of the pose of a camera with respect to an object. This method is flexible in the sense that it can be used with four or more coplanar feature points. Unlike techniques based on closed form solutions for four coplanar points, this method is able to determine two poses with quality measures. Applications range from aerial imagery interpretation to robot navigation.

Whereas there is no upper limit to the number of points the algorithm can use, there is a lower limit. We performed experiments with only three feature points, with images for which there was clearly four possible poses. For this purpose we used the example provided by Fischler and Bolles [3] for an equilateral triangle. However, our algorithm always converged toward the pose in which the triangle is parallel to the image plane. For this reason we advise that the algorithm be used only when more than three feature points are available.

Acknowledgements

The support of the first author by the Thomson-CSF company and of all authors by the Defense Advanced Research Projects Agency (ARPA Order No. 8459) and the U.S. Army Topographic Engineering Center under Contract DACA76-92-C-0009, is gratefully acknowledged. We also wish to thank R. de Peufeilhoux for his insightful guidance.

Appendix: Example of calculation with four coplanar points

- **Effective focal length** = 760 pixels;
- **Object points**: $(-15,0,0),(15,0,0),(15,500,0),(-15,500,0)$ (m) (Fig. 15)
  Object reference frame $F_o = (M_0, u, v, w)$;
- **Position and orientation** of F_o with respect to the camera reference frame
  $F_c = (O, x, y, z)$:

$$\mathbf{T} = \begin{bmatrix} 250 \\ 100 \\ 2000 \end{bmatrix} \text{ (m)} \quad \text{and} \quad \mathbf{R} = \begin{bmatrix} 0.5 & -0.866 & 0 \\ -0.557 & -0.321 & -0.766 \\ 0.663 & 0.383 & -0.643 \end{bmatrix} \Leftrightarrow \begin{bmatrix} \Theta_x & = & 130^\circ \\ \Theta_y & = & 0^\circ \\ \Theta_z & = & 60^\circ \end{bmatrix}$$

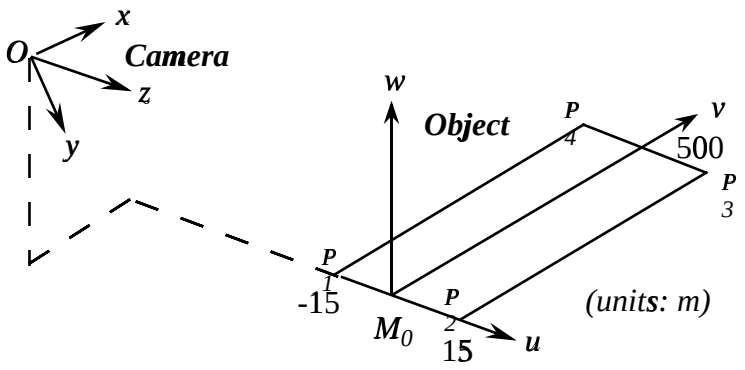


Figure 15: Object and camera defined in Appendix.

The synthetic perspective image for this object and this pose is represented on top of Fig. 16. The coordinates of the image points are

$$(92.6, 41.38), (97.37, 34.65), (-60.59, -23.84), (-66.37, -18.24)$$

The diagram of the computations and results is provided in Fig. 16.

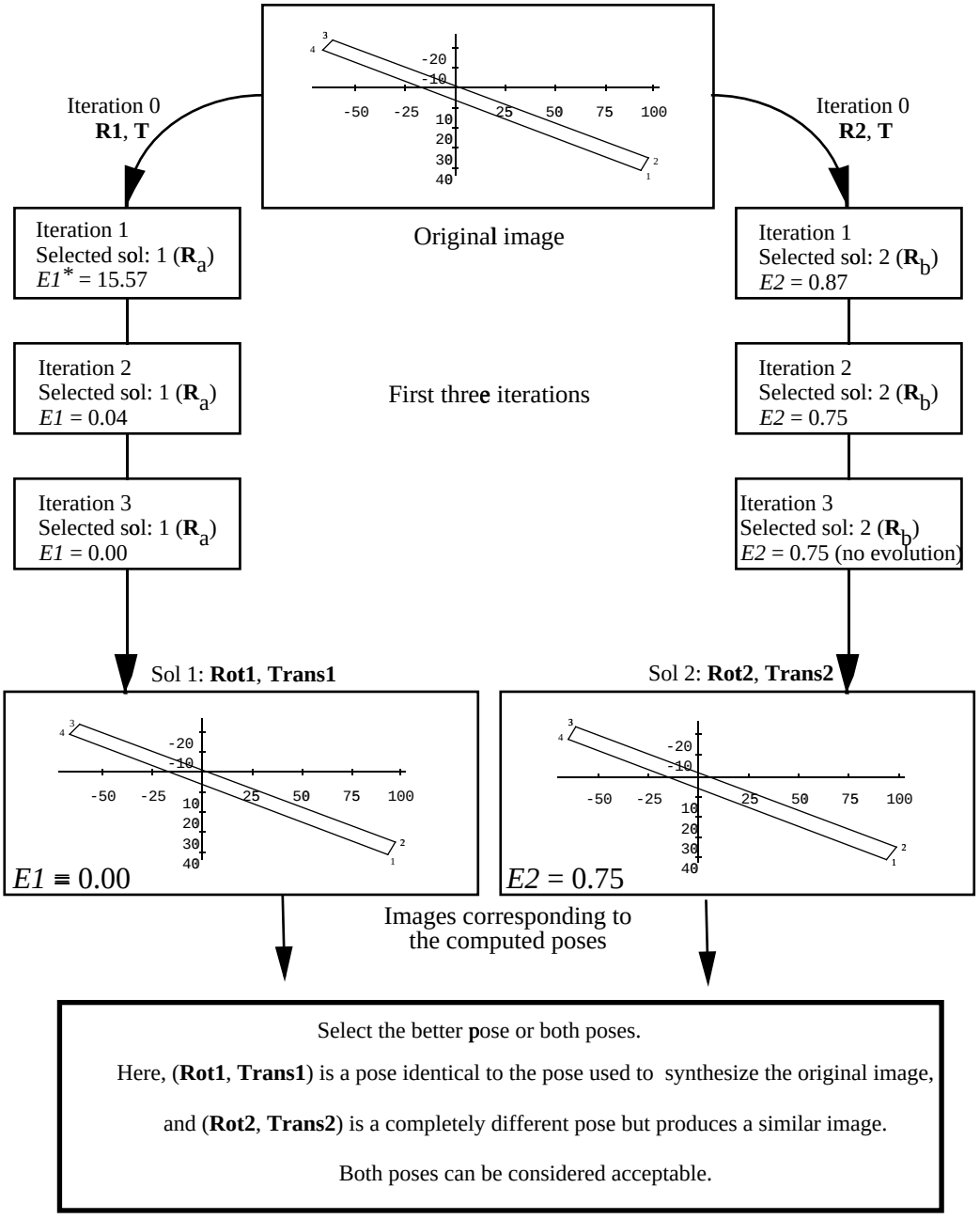
References

[1] M.A. Abidi, T. Chandra, "A New Efficient and Direct Solution for Pose Estimation Using Quadrangular Targets: Algorithm and Evaluation", Department of Electrical and Computer Engineering, The University of Tennessee, July 1991, to be published in *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

[2] D.F. DeMenthon, L.S. Davis, "Model-Based Object Pose in 25 Lines of Code", Second European Conf. Computer Vision, May 1992, Lecture Notes in Computer Science, G. Sandini (Ed.), Springer-Verlag; extended version to appear in *International Journal of Computer Vision*, Summer 1995.

[3] M.A. Fischler, R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", *Communications of the ACM*, Vol. 24, No. 6, 1981, pp. 381–395.

[4] R.M. Haralick, C. Lee, K. Ottenberg, M. Nölle, "Analysis and Solutions of the Three Point Perspective Pose Estimation Problem", Proceedings of the IEEE Computer So-



* The length of the image diagonal is 100 pixels. The error measure is the average Euclidean distance between actual image points and corresponding image points for the computed poses.

Figure 16: Diagram of calculations and results for example of Appendix.

ciety Conference on Computer Vision and Pattern Recognition (CVPR), June 1991, Maui, HI, pp. 592–598.

[5] R.M. Haralick, "Determining Camera Parameters from the Perspective Projection of a Rectangle", *Pattern Recognition*, Vol. 22, No. 3, 1989, pp. 225–230.

[6] R.J. Holt, A.N. Netravali, "Camera Calibration: Some New Results", *CVGIP: Image Understanding*, Vol. 54, No. 3, 1991, pp. 368–383.

[7] R. Horaud, B. Conio, O. Leboulleux, B. Lacolle, "An Analytic Solution for the Perspective 4-Point Problem", *Computer Vision, Graphics, and Image Processing*, Vol. 47, 1989, pp. 33–44.

[8] D. Huttenlocher, S. Ullman, "Recognizing Solid Objects by Alinement", Proc. DARPA Image Understanding Workshop, 1988, pp. 1114–1122.

[9] W.H. Press, B.P. Flannery, S.A. Teukolsky,

[10] D. Oberkampf, D.F. DeMenthon, and L.S. Davis, "Iterative Pose Estimation using Coplanar Feature Points", Center for Automation Research Technical Report CAR-TR-677, University of Maryland, July 1993. W.T. Veterling, *Numerical Recipes in C*, Cambridge University Press, Cambridge, UK, 1988.

[11] W.J. Wolfe, D. Mathis, C.W. Sklair, M. Magee, "The Perspective View of Three Points", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 1, 1991, pp. 66–73.

[12] J.S.C. Yuan, "A General Photogrammetric Method for Determining Object Position and Orientation", *IEEE Transactions on Robotics and Automation*, Vol. 5, No. 2, 1989, pp. 129–142.