Euclidean Shape and Motion from Multiple Perspective Views by Affine Iterations [1]

Stéphane Christy and Radu Horaud
GRAVIR–IMAG & INRIA Rhône-Alpes
655, avenue de l'Europe
38330 Montbonnot Saint Martin, FRANCE

Long version of a paper which appeared in
IEEE Trans. on Pattern Analysis and Machine Intelligence, volume 18, NO. 11, pages 1098–1104

Abstract – *In this paper we describe a method for solving the Euclidean reconstruction problem with a perspective camera model by incrementally performing Euclidean reconstruction with either a weak or a paraperspective camera model. With respect to other methods that compute shape and motion from a sequence of images with a calibrated perspective camera, this method converges in a few iterations, is computationally efficient, and does not suffer from the non linear nature of the problem. With respect to methods that use an affine camera model (such as factorization and/or affine-invariant methods) the method described below solves for the sign (reversal) ambiguity in a very simple way and provides much more accurate reconstructions results. We give a detailed account of the method, analyse its convergence based on numerical and experimental considerations, and test its efficiency with both synthetic and real data.*

Keywords – *perspective, weak perspective, and paraperspective camera models, 3-D Euclidean and affine reconstruction, shape and motion from multiple views.*

[1] This work has been supported by "Société Aérospatiale".

Contents

1 Introduction and background

The problem of computing 3-D shape and motion from a long sequence of images has received a lot of attention for the last few years. Previous approaches attempting to solve this problem fall into several categories, whether the camera is calibrated or not, and/or whether a projective or an affine model is being used. With a calibrated camera one may compute Euclidean shape up to a scale factor using either a projective model [5], [21], [20], [3], or an affine model [6], [22], [23], [27], [14], [18], [15]. With an uncalibrated camera the recovered shape is defined up to a projective transformation [8], [2], [10], [26], or up to an affine transformation [8], [10]. One can therefore address the problem of either Euclidean, affine, or projective shape reconstruction.

In this paper we are interested in Euclidean shape reconstruction with a calibrated camera. In that case, one may use either a perspective camera model or one of its affine approximations – orthographic projection, weak perspective, or paraperspective.

The perspective model has associated with it, in general, non linear reconstruction techniques. This naturally leads to non-linear minimization methods which require some form of initialization [5], [21], [20], [3], [19], [2], [10], [26]. If the initial "guess" is too faraway from the true solution then the minimization process is either very slow or it converges to a wrong solution. Affine camera models lead, in general, to linear resolution methods [6], [23], [27], [14], [15], but the solution is defined only up to a sign (reversal) ambiguity and, both these two solutions are just an approximation of the true solution. For example, Figure 1 shows four Euclidean 3-D shapes that have orthographically been projected onto an image plane. The first one (top-left) is the theoretical shape. The second one (top-right) is the reconstructed shape using a perspective camera model and the method outlined in this paper. The third and fourth ones (bottom-left and bottom-right) are the reconstructed mirror-symmetric shapes using a weak perspective camera model.

One way to combine the perspective and affine models could be to use the affine solution in order to initialize the non-linear minimization process associated with perspective. However, there are several drawbacks with such an approach. First, such a resolution technique does not take into account the simple link that exists between the perspective model and its linear approximations. Second, there is no mathematical evidence that a non-linear least-squares minimization method is "well" initialized by a solution that is obtained linearly. Third, there are two solutions associated with the affine model and it is not clear which one to choose.

The perspective projection can be modelled by a projective transformation from the 3-D projective space to the 2-D projective plane. Weak perspective and paraperspective are the most common affine approximations of perspective. Weak perspective may well be viewed as a zero-order approximation: $1/(1+\varepsilon) \approx 1$. Paraperspective [1] is a first order approximation of full perspective: $1/(1+\varepsilon) \approx 1-\varepsilon$. Recently, in [7] a method has been proposed for determining the pose of a 3-D shape with respect to a single view by iteratively improving the pose computed with a weak perspective camera model to converge, to the limit, to a pose estimation using a perspective camera model. To our knowledge, the method cited above, i.e., [7] is among one of the first computational paradigms that link linear techniques (associated with affine camera models) with a perspective model. In [12] and [13] an extension of this paradigm to paraperspective is proposed. The authors show that the *iterative paraperspective* pose algorithm has better convergence properties than the *iterative weak perspective* one.

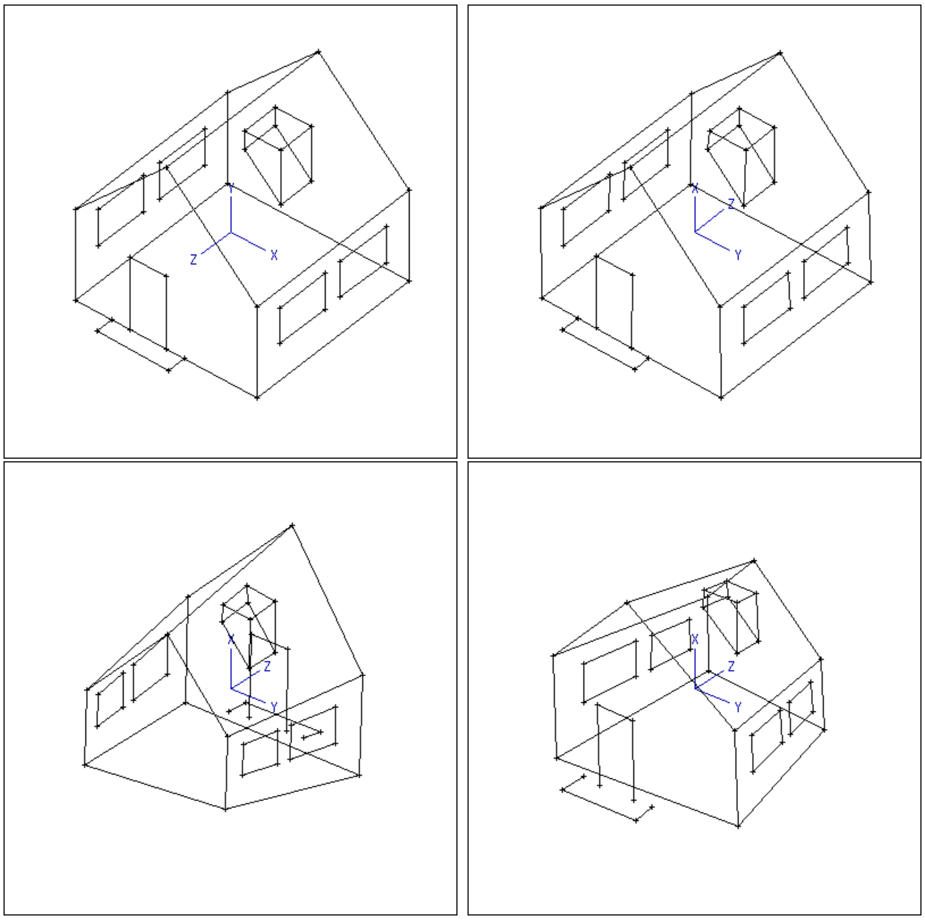

Figure 1: This figure shows a theoretical 3-D shape (top-left) and three reconstructions of this shape from 10 views. The first reconstruction (top-right) was obtained using a perspective camera model and the method described in this paper. The second reconstruction (bottom-left) and its reversal (bottom-right) were obtained using a weak perspective camera model and the factorization method of Tomasi & Kanade.

In this paper we describe a new Euclidean reconstruction method that makes use of affine reconstruction in an iterative manner such that this iterative process converges, to the limit, to a set of 3-D Euclidean shape and motion parameters that are consistent with a perspective model. The novelty of the method that we propose is twofold: (i) it extends the iterative pose determination algorithms described in [7] and in [13] to deal with the problem of shape and motion from multiple views and (ii) it is a generalization to perspective of the factorization methods [23], [15] and of the affine-invariant methods [27], [28]. More precisely, the *affine-iterative reconstruction* methods that we propose here have a number of interesting features:

- They solve the sign (or reversal) ambiguity that is inherent with affine reconstruction;
- They are fast because they converge in a few iterations (typically 3 to 5 iterations), each iteration involving simple linear algebra computations. In particular there is no matrix inversion being involved as is the case with any iterative non-linear optimization technique;

- We show that the quality of the Euclidean reconstructions obtained with our methods is only weakly influenced by camera calibration errors. The only intrinsic camera parameter that has a crucial effect on the quality of the reconstruction is the ratio between the horizontal pixel size and vertical pixel size — ratio which is provided by camera manufacturers and which is known to be very stable parameter [24];
- They allow the use of either weak or paraperspective camera model approximations which are used iteratively, and
- They can be combined with almost any affine shape and motion algorithm. In particular we show how our method can be combined with factorization methods [23], [15].

1.1 Paper organization

The remainder of this paper is organized as follows. Section 2 describes the relationship between full perspective, paraperspective, and weak perspective and establishes the relationships between the perspective projection of a 3-D point and its weak and paraperspective projections. Section 3 describes how to perform reconstruction with a perspective camera model by iterating either a weak perspective reconstruction or a paraperspective reconstruction algorithm. Section 4 outlines the wellknown factorization algorithm and describes a method for converting affine reconstruction into Euclidean reconstruction using paraperspective. Section 5 describes how to solve for the reversal ambiguity associated, in general, with an affine camera. Section 6 explains why the method described in this paper is not too sensitive to camera calibration errors, provides a simple numerical analysis of the convergence properties of the affine iterative algorithms, and provides a theoretical comparison of the complexity associated with affine iterative methods and with non-linear minimization methods. Finally, section 7 provides a practical evaluation of the method using simulated data and various motions and Section 8 describes results obtained with real imagery.

2 Camera models

Let us consider a pin hole camera model. We denote by P_i a 3-D point lying onto a 3-D object with Euclidean coordinates X_i, Y_i, and Z_i in a frame that is attached to the object – the object frame. The origin of this frame may well be the object point P_0. An object point P_i projects onto the image in p_i with image coordinates u_i and v_i and we have ($\mathbf{P}_i$ is the vector $\overrightarrow{P_0P_i}$ from point P_0 to point P_i):

$$\begin{pmatrix} su_i \\ sv_i \\ s \end{pmatrix} = \begin{pmatrix} \alpha_u & 0 & u_c \\ 0 & \alpha_v & v_c \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{i}^T & t_x \\ \mathbf{j}^T & t_y \\ \mathbf{k}^T & t_z \\ 0\,0\,0 & 1 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix} \tag{1}$$

The first 3×3 matrix describes the affine transformation between the camera coordinates and the images coordinates. The second 3×4 matrix describes the projective transformation between the 3-D projective space and the 2-D projective image plane. The third 4×4 matrix describes the rigid transformation (rotation and translation) between the object frame and the camera frame. **i**, **j**, and **k** are the three row vectors associated with the rotation matrix.

From now on we will be assuming that the intrinsic camera parameters are known and therefore we consider the relationship between image points expressed in camera coordinates and in image coordinates:

$$\begin{aligned} u_i &= \alpha_u x_i + u_c & (2) \\ v_i &= \alpha_v y_i + v_c & (3) \end{aligned}$$

In these equations α_u and α_v are the vertical and horizontal scale factors and u_c and v_c are the image coordinates of the intersection of the optical axis with the image plane. It will be shown that the quality of the reconstruction method described below depends strongly only on the ratio α_u/α_v and that the reconstruction obtained with our method is not sensitive to errors in u_c and v_c. The relationship between object points and image points expressed in camera coordinates can be written as:

$$\begin{aligned} x_i &= \frac{\mathbf{i} \cdot \mathbf{P}_i + t_x}{\mathbf{k} \cdot \mathbf{P}_i + t_z} & (4) \\ y_i &= \frac{\mathbf{j} \cdot \mathbf{P}_i + t_y}{\mathbf{k} \cdot \mathbf{P}_i + t_z} & (5) \end{aligned}$$

We divide both the numerator and the denominator of eqs. (4) and (5) by t_z and we introduce the following notations:

- $\mathbf{I} = \mathbf{i}/t_z$ is the first row of the rotation matrix scaled by the z-component of the translation vector;
- $\mathbf{J} = \mathbf{j}/t_z$ is the second row of the rotation matrix scaled by the z-component of the translation vector;
- $x_0 = t_x/t_z$ and $y_0 = t_y/t_z$ are the camera coordinates of p_0 which is the projection of P_0 – the origin of the object frame, and
- We denote by ε_i the following ratio:

$$\varepsilon_i = \frac{\mathbf{k} \cdot \mathbf{P}_i}{t_z} \qquad (6)$$

We may now rewrite the perspective equations as:

$$\begin{aligned} x_i &= \frac{\mathbf{I} \cdot \mathbf{P}_i + x_0}{1 + \varepsilon_i} & (7) \\ y_i &= \frac{\mathbf{J} \cdot \mathbf{P}_i + y_0}{1 + \varepsilon_i} & (8) \end{aligned}$$

Whenever the object is at some distance from the camera, the ε_i are small compared to 1. We may therefore introduce two approximations of the perspective equations: weak and paraperspective.

2.1 Weak perspective

Weak perspective assumes that the object points lie in a plane parallel to the image plane passing through the origin of the object frame, i.e., P_0. This is equivalent to a zero-order approximation:

$$\frac{1}{1+\varepsilon_i} \approx 1 \;\; \forall i, \; i \in \{1...n\}$$

With this approximation, eqs. (7) and (8) become:

$$x_i^w - x_0 = \mathbf{I} \cdot \mathbf{P}_i \tag{9}$$

$$y_i^w - y_0 = \mathbf{J} \cdot \mathbf{P}_i \tag{10}$$

In these two equations x_i^w and y_i^w are the camera coordinates of the weak perspective projection of the point P_i. By identification with eqs. (7) and (8) we obtain the relationship between the weak perspective and the perspective projections of P_i:

$$x_i^w = x_i(1+\varepsilon_i) \tag{11}$$

$$y_i^w = y_i(1+\varepsilon_i) \tag{12}$$

These equations allow us to determine the quality of the weak perspective approximation with respect to the perspective projection. Indeed the error between the weak perspective projection and the "true" projection is:

$$\Delta x^w = |x_i^w - x_i| = |x_i \varepsilon_i| \tag{13}$$

$$\Delta y^w = |y_i^w - y_i| = |y_i \varepsilon_i| \tag{14}$$

Hence the quality of the approximation depends both on the value of ε_i AND on the position of the point in the image. We consider for example a 512×512 image. Approximate values for the intrinsic parameters are: $\alpha_u = \alpha_v = 1000$ and $u_c = v_c = 256$. Therefore using eq. (2) and (3) we have:

$$0 \leq x_i \; , \; y_i \leq 0.25$$

The lower bound corresponds to the point of intersection of the optical axis with the image plane ($x_i = y_i = 0$). The upper bound corresponds to the image borders:

$$x_i = \frac{u_i - u_c}{\alpha_u} = \frac{512 - 256}{1000} \approx 0.25$$

We conclude that for small distance/size ratios, i.e., large values for ε_i, the weak perspective approximation is still valid *provided that the object lies in the neighborhood of the optical axis.*

2.2 Paraperspective

Paraperspective may be viewed as a first-order approximation of perspective. Indeed, with the approximation:

$$\frac{1}{1+\varepsilon_i} \approx 1 - \varepsilon_i \;\; \forall i, \; i \in \{1...n\}$$

we obtain the paraperspective projection of P_i:

$$\begin{aligned} x_i &\approx (\mathbf{I} \cdot \mathbf{P}_i + x_0)(1 - \varepsilon_i) \\ &\approx \mathbf{I} \cdot \mathbf{P}_i + x_0 - x_0 \varepsilon_i \\ &= \frac{\mathbf{i} \cdot \mathbf{P}_i}{t_z} + x_0 - x_0 \frac{\mathbf{k} \cdot \mathbf{P}_i}{t_z} \\ &= x_i^p \end{aligned}$$

where the term in $1/t_z^2$ was neglected. There is a similar expression for y_i^p.

Finally, the paraperspective equations are:

$$x_i^p - x_0 = \frac{\mathbf{i} - x_0\,\mathbf{k}}{t_z} \cdot \mathbf{P}_i \tag{15}$$

$$y_i^p - y_0 = \frac{\mathbf{j} - y_0\,\mathbf{k}}{t_z} \cdot \mathbf{P}_i \tag{16}$$

In order to obtain the relationship between the paraperspective and the perspective projections of P_i we can write these equations as follows:

$$\begin{aligned} x_i^p &= x_0 + \mathbf{I} \cdot \mathbf{P}_i - x_0 \varepsilon_i \\ y_i^p &= y_0 + \mathbf{J} \cdot \mathbf{P}_i - y_0 \varepsilon_i \end{aligned}$$

Again, by identification with eqs. (7) and (8) we obtain the relationship between the paraperspective and the perspective projections of P_i:

$$x_i^p = x_i(1 + \varepsilon_i) - x_0 \varepsilon_i \tag{17}$$

$$y_i^p = y_i(1 + \varepsilon_i) - y_0 \varepsilon_i \tag{18}$$

As in the previous section, we can easily estimate the error between the paraperspective and perspective projections:

$$\Delta x^p = |x_i^p - x_i| = |(x_i - x_0)\varepsilon_i| \tag{19}$$

$$\Delta y^p = |y_i^p - y_i| = |(y_i - y_0)\varepsilon_i| \tag{20}$$

Whenever an object point P_i is far from the optical axis the weak perspective model is a poor approximation. However, a proper choice of the origin, i.e, P_0, and the use of the paraperspective model can compensate and provide a good approximation even if ε_i is not small.

3 Reconstruction with a perspective camera

Let us consider again the perspective equations (7) and (8). These equations may also be written as:

$$x_i(1 + \varepsilon_i) - x_0 = \mathbf{I} \cdot \mathbf{P}_i \tag{21}$$

$$y_i(1 + \varepsilon_i) - y_0 = \mathbf{J} \cdot \mathbf{P}_i \tag{22}$$

Let us subtract the *paraperspective term* from both the left and right sides of equations (21) and (22). We obtain:

$$
\begin{aligned}
x_i(1+\varepsilon_i) - x_0 - x_0 \underbrace{\frac{1}{t_z}\mathbf{k}\cdot\mathbf{P}_i}_{\varepsilon_i} &= \frac{1}{t_z}\mathbf{i}\cdot\mathbf{P}_i - x_0\frac{1}{t_z}\mathbf{k}\cdot\mathbf{P}_i \\
y_i(1+\varepsilon_i) - y_0 - y_0 \underbrace{\frac{1}{t_z}\mathbf{k}\cdot\mathbf{P}_i}_{\varepsilon_i} &= \frac{1}{t_z}\mathbf{i}\cdot\mathbf{P}_i - y_0\frac{1}{t_z}\mathbf{k}\cdot\mathbf{P}_i
\end{aligned}
$$

These equations can be written more compactly as:

$$
\begin{aligned}
(x_i - x_0)(1+\varepsilon_i) &= \mathbf{I}_p \cdot \mathbf{P}_i & (23)\\
(y_i - y_0)(1+\varepsilon_i) &= \mathbf{J}_p \cdot \mathbf{P}_i & (24)
\end{aligned}
$$

with:

$$
\begin{aligned}
\mathbf{I}_p &= \frac{\mathbf{i} - x_0\,\mathbf{k}}{t_z} & (25)\\
\mathbf{J}_p &= \frac{\mathbf{j} - y_0\,\mathbf{k}}{t_z} & (26)
\end{aligned}
$$

To summarize, we have two different sets of equations that describe the *same* perspective camera model:

- The first set, i.e., equations (21) and (22) establish the link between perspective and weak perspective and
- The second set, i.e., equations (23) and (24) establish the link between perspective and paraperspective.

Furthermore, equations (21) and (22) or equations (23) and (24) may be interpreted in two different ways:

- We can consider x_i and y_i as the perspective projection of P_i or
- We can consider $x_i(1+\varepsilon_i)$ and $y_i(1+\varepsilon_i)$ or $x_i(1+\varepsilon_i) - x_0\varepsilon_i$ and $y_i(1+\varepsilon_i) - y_0\varepsilon_i$ as the weak or paraperspective projections of P_i.

The basic idea of our method is to estimate values for ε_i *incrementally* such that one can compute either the weak or the paraperspective projections of the 3-D points from the perspective projections which are the *true image measurements*. Therefore, the perspective reconstruction problem is reduced to the problem of iterative weak perspective or iterative paraperspective reconstruction.

Let us consider now k views of the same scene points. We assume that image-to-image correspondences have already been established. Both equations (21) and (22) or equations (23) and (24) can be written as:

$$
\underbrace{\mathbf{s}_{ij}}_{2\times 1} = \underbrace{A_j}_{2\times 3}\underbrace{\mathbf{P}_i}_{3\times 1} \qquad (27)
$$

In this formula the subscript i stands for the i^{th} point and the subscript j for the j^{th} image. The 2-vector $\mathbf{s}_{ij}$ is equal to (weak perspective):

$$\mathbf{s}_{ij} = \begin{pmatrix} x_{ij}(1+\varepsilon_{ij}) - x_{0j} \\ y_{ij}(1+\varepsilon_{ij}) - y_{0j} \end{pmatrix} \tag{28}$$

or to (paraperspective):

$$\mathbf{s}_{ij} = \begin{pmatrix} (x_{ij} - x_{0j})(1+\varepsilon_{ij}) \\ (y_{ij} - y_{0j})(1+\varepsilon_{ij}) \end{pmatrix} \tag{29}$$

In these equations ε_{ij}, i.e., eq. (6) is defined for each point and for each image:

$$\varepsilon_{ij} = \frac{\mathbf{k}_j \cdot \mathbf{P}_i}{t_{z_j}} \tag{30}$$

The reconstruction problem is now the problem of simultaneously solving $2 \times n \times k$ equations of the form of eq. (27). The unknowns of these equations are:

- The coordinates of the 3-D points P_i ($3 \times n$ variables);
- The terms of the projection matrices $\mathbf{A}_j$ ($2 \times 3 \times k$ variables), and
- The *perspective corrections* ε_{ij} ($k \times n$ variables).

We introduce a method that solves these equations by linear iterations. More precisely, this method can be summarized by the following algorithm:

1. $\forall i,\ i \in \{1...n\}$ and $\forall j,\ j \in \{1...k\}$ set: $\varepsilon_{ij} = 0$ (initialisation);
2. Update the values of $\mathbf{s}_{ij}$ according with eq. (29) and using the newly computed values for ε_{ij};
3. Perform an Euclidean reconstruction with a paraperspective camera;
4. $\forall i,\ i \in \{1...n\}$ and $\forall j,\ j \in \{1...k\}$ estimate new values for ε_{ij} according with eq.(30);
5. Check the values of ε_{ij}:

   if $\forall (i,j)$ the values of ε_{ij} just estimated at this iteration are identical with the values estimated at the previous iteration,
   then stop;

   else go to step 2.

The most important step of this algorithm is step 4: estimate new values for ε_{ij}. This computation can be made explicit if one considers in some more detail step 3 of the algorithm which can be further decomposed into: (i) Affine reconstruction and (ii) Euclidean reconstruction.

The problem of affine reconstruction is the problem of determining both $\mathbf{A}_j$ and $\mathbf{P}_i$, for all j and for all i, in eq. (27), when some estimates of $\mathbf{s}_{ij}$ are provided. Such a reconstruction determines shape and motion up to a 3-D affine transformation. Indeed, for any 3×3 invertible matrix $\mathbf{T}$ we have:

$$\mathbf{A}_j\mathbf{P}_i = \mathbf{A}_j\mathbf{T}\mathbf{T}^{-1}\mathbf{P}_i \tag{31}$$

In order to convert affine shape and motion into Euclidean shape and motion, one needs to consider some Euclidean constraints associated either with the motion of the camera or with the shape being viewed by the camera. Since we deal here with a calibrated camera, we may well use rigid motion constraints in conjunction with weak or paraperspective [23], [15]. Therefore, step 3 of the algorithm provides both Euclidean shape ($\mathbf{P}_1 ... \mathbf{P}_n$) and Euclidean motion, i.e., k matrices of the form:

$$\begin{pmatrix} \mathbf{i}_j^T & t_{x_j} \\ \mathbf{j}_j^T & t_{y_j} \\ \mathbf{k}_j^T & t_{z_j} \\ 0\;0\;0 & 1 \end{pmatrix}$$

Various methods are available for estimating Euclidean structure from affine structure either with a calibrated camera [23], [28] (weak perspective), [15] (paraperspective) or with an uncalibrated camera [16, 17]. Based on the parameters of the Euclidean shape and motion thus computed one can estimate ε_{ij} for all i and for all j using eq. (30) – step 4.

The first iteration of the algorithm performs a 3-D reconstruction using the initial image measurements and a weak or para perspective camera model. This first reconstruction allows an estimation of values for the ε_{ij}'s which in turn allow the image vectors $\mathbf{s}_{ij}$ to be *modified* (step 2 of the algorithm). The $\mathbf{s}_{ij}$'s are modified according to eq. (28) (for weak perspective) or according to eq. (29) (for para perspective) such that they better fit the approximated camera model being used.

The next iterations of the algorithm perform a 3-D reconstruction using (i) image vectors that are incrementally modified and (ii) a weak (or para) perspective camera model.

At convergence, the equations (27) are equivalent with the perspective equations (21), (22) or (23), (24). In other terms, this algorithm solves for Euclidean reconstruction with a perspective camera by iterations of an Euclidean reconstruction method with an affine camera. Therefore, before we proceed further in order to understand some important features of this iterative algorithm, it is necessary to have insights into the problem of Euclidean reconstruction with an affine camera model.

The iterative algorithm outlined in this paper is best illustrated in Figure 2. At the first iteration, the algorithm considers the *true perspective projections* of P_i and attempts to reconstruct the 3-D points as if they were projected in the image using weak perspective. At the second iteration the algorithm considers modified image point positions. At the last iteration, the image point positions were modified such that they fit the *weak perspective projections*. The relative positions of the perspective projections and of the weak perspective projections verify the theoretical relationship given by eqs. (11) and (12). Notice that the perspective projection of point P_0 is identical to its weak (and para) perspective projection and hence the projection of this point is not modified. The same strategy can be used for paraperspective.

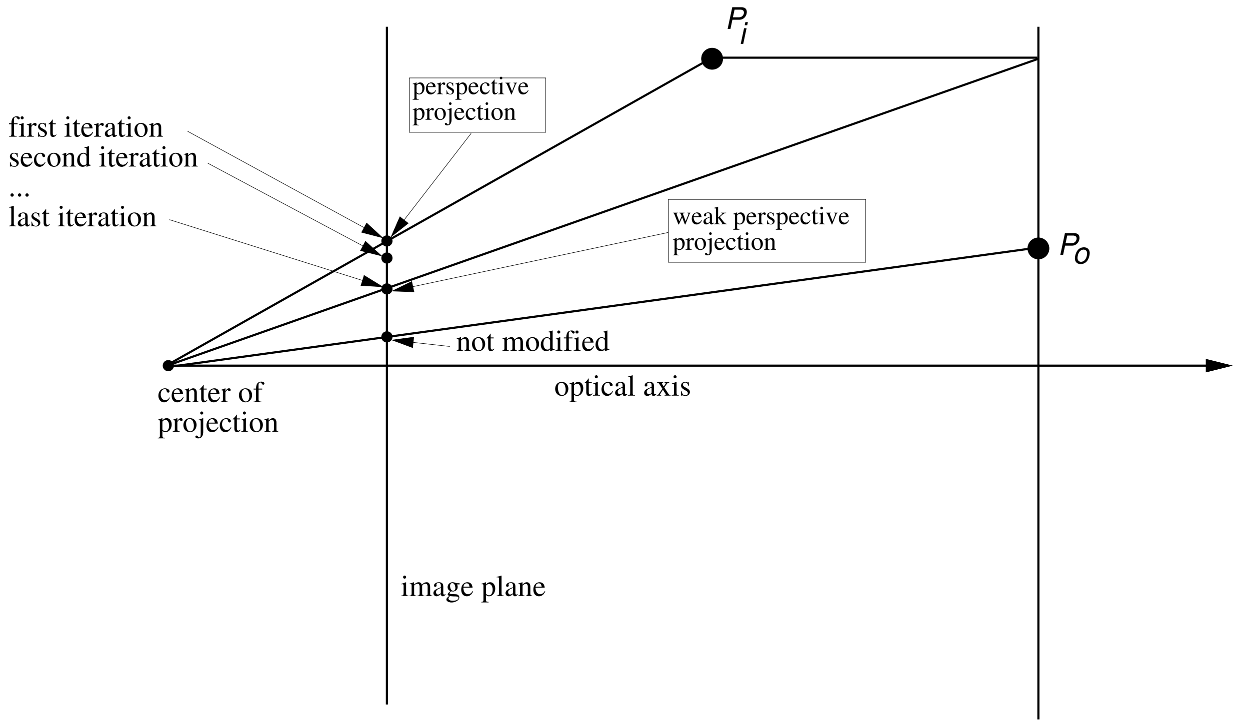

Figure 2: The iterative algorithm described in this section modifies the projection of a 3-D point from true perspective to weak perspective (see text).

4 Reconstruction with a weak-perspective or a paraperspective camera

In this section we develop step 3 of the algorithm outlined in the previous section. Methods that use a linear camera model provide a 3-D affine reconstruction if at least 2 views of 4 non-coplanar points are available and if the motion is not a pure translation. However, 3 views are necessary in order to convert this affine reconstruction into an Euclidean one. While the affine-invariant method allows a more direct analysis of the problem, [14], [27], [4] the factorization method is more convenient from a practical point of view.

The factorization method, [23] computes shape and motion simultaneously by performing a singular value decomposition of the $2k \times n$ matrix $\boldsymbol{\sigma}$ which is formed by concatenating eq. (27) for all i and j:

$$\boldsymbol{\sigma} = \mathbf{A}\mathbf{S} \tag{32}$$

We refer to this formula as the *affine shape and motion equation*, or:

$$\begin{pmatrix} \mathbf{s}_{11} & \ldots & \mathbf{s}_{n1} \\ \vdots & & \vdots \\ \mathbf{s}_{1k} & \ldots & \mathbf{s}_{nk} \end{pmatrix} = \begin{pmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_k \end{pmatrix} \begin{pmatrix} \mathbf{S}_1 & \ldots & \mathbf{S}_n \end{pmatrix}$$

Tomasi & Kanade [23] noticed that A and S in eq. (32) may be computed simultaneously by performing a singular value decomposition of the $2k \times n$ matrix $\boldsymbol{\sigma}$:

$$\boldsymbol{\sigma} = \mathbf{O}_1 \boldsymbol{\Sigma} \mathbf{O}_2$$

Affine shape and motion, i.e., the $3 \times n$ matrix S and the $2k \times 3$ matrix A can be computed only if the rank of the *measurement* matrix $\boldsymbol{\sigma}$ is equal to 3. The rank of $\boldsymbol{\sigma}$ is provided by the rank of the $n \times n$ diagonal matrix $\boldsymbol{\Sigma}$. Even if the rank condition stated above is satisfied, the rank of $\boldsymbol{\Sigma}$ may be greater than 3 due to numerical instability. Tomasi & Kanade suggested to solve the rank problem by truncating the matrix $\boldsymbol{\Sigma}$ such that only the 3 largest diagonal values are considered. They claim that this truncation amounts to removing noise present in the measurement matrix. Therefore, one can write the singular value decomposition of the measurement matrix as:

$$\boldsymbol{\sigma} = \mathbf{O}_1' \boldsymbol{\Sigma}' \mathbf{O}_2' + \mathbf{O}_1'' \boldsymbol{\Sigma}'' \mathbf{O}_2''$$

where $\boldsymbol{\Sigma}'$ is a 3×3 diagonal matrix containing the 3 largest diagonal terms of $\boldsymbol{\Sigma}$.

Finally, affine shape and affine motion may be given by:

$$\mathbf{S} = (\boldsymbol{\Sigma}')^{1/2} \mathbf{O}_2'$$

$$\mathbf{A} = \mathbf{O}_1' (\boldsymbol{\Sigma}')^{1/2}$$

It is worthwhile to notice that Tomasi & Kanade were not the first to propose such a factorization with truncation technique. Hartley [11] recalls that the method is in fact due to Tsai & Huang [25] who devised a factorization method in order to extract translation and rotation from the essential matrix – a 3×3 matrix with its rank equal to 2. In [11] Hartley recalls that the truncation methods (enforcing a number of singular values to be null) minimizes the Frobenius norm.

4.1 From affine to Euclidean

Obviously, the factorization method briefly outlined above does not provide a unique decomposition of the measurement matrix $\boldsymbol{\sigma}$. Tomasi & Kanade [23] and Weinshall and Tomasi [28] provide solutions for the case of a weak perspective camera. Poelman and Kanade [15] provide a solution for the case of a paraperspective camera. The method that we describe below for recovering Euclidean shape and motion with a paraperspective camera is an alternative approach to the method described in [15]. One may wonder why an alternative solution is ever necessary. The solution suggested by Poelman and Kanade assumes that the 3-vectors $\mathbf{i}_j$ and $\mathbf{j}_j$ (see below) are perpendicular to each other, while our method doesn't. Indeed, there is no mathematical reason to assume that these two vectors are perpendicular because there is no explicit constraint enforcing this perpendicularity. Poelman and Kanade determine the motion parameters by solving an overcostrained set of 6 linear equations in 4 unknowns. Below we show that there is a simpler solution which solves for 3 linear equations with 3 unknowns and we prove that a unique solution always exists.

One has to determine Euclidean shape and motion by combining the affine reconstruction method just described and the Euclidean constraints available with the camera model being used. As already mentioned, one has to determine a 3×3 invertible matrix $\mathbf{T}$ such that the affine shape $\mathbf{S}$ becomes Euclidean:

$$\begin{pmatrix} \mathbf{P}_1 & \ldots & \mathbf{P}_n \end{pmatrix} = \mathbf{T}^{-1} \begin{pmatrix} \mathbf{S}_1 & \ldots & \mathbf{S}_n \end{pmatrix}$$

and the affine motion becomes rigid:

$$\begin{pmatrix} R_1 \\ \vdots \\ R_k \end{pmatrix} = \begin{pmatrix} A_1 \\ \vdots \\ A_k \end{pmatrix} \mathbf{T}$$

Indeed, in order to avoid confusion we denote by S and A affine shape and affine motion and by P and R Euclidean shape and rigid motion. The matrices R_j are given by:

$$R_j = \begin{pmatrix} \mathbf{I}_{p_j} \\ \mathbf{J}_{p_j} \end{pmatrix}$$

The row vectors of these matrices must have an expression identical with eq. (25) and (26). The Euclidean constraints allowing the computation of $\mathbf{T}$ are therefore the following [15]:

$$\frac{\|\mathbf{I}_{p_j}\|^2}{1+x_{0_j}^2} = \frac{\|\mathbf{J}_{p_j}\|^2}{1+y_{0_j}^2}$$

and

$$\mathbf{I}_{p_j} \cdot \mathbf{J}_{p_j} = \frac{x_{0_j} y_{0_j}}{2} \left(\frac{\|\mathbf{I}_{p_j}\|^2}{1+x_{0_j}^2} + \frac{\|\mathbf{J}_{p_j}\|^2}{1+y_{0_j}^2} \right)$$

We denote by $\mathbf{a}_j$ and $\mathbf{b}_j$ the row vectors of matrix $\mathbf{A}_j$. Using the constraints above, for k images one obtains $2k$ constraints for the matrix $\mathbf{T}$:

$$\frac{\mathbf{a}_j^T \mathbf{T}\mathbf{T}^T \mathbf{a}_j}{1+x_{0_j}^2} - \frac{\mathbf{b}_j^T \mathbf{T}\mathbf{T}^T \mathbf{b}_j}{1+y_{0_j}^2} = 0 \quad (33)$$

$$\mathbf{a}_j^T \mathbf{T}\mathbf{T}^T \mathbf{b}_j = \frac{x_{0_j} y_{0_j}}{2} \left(\frac{\mathbf{a}_j^T \mathbf{T}\mathbf{T}^T \mathbf{a}_j}{1+x_{0_j}^2} + \frac{\mathbf{b}_j^T \mathbf{T}\mathbf{T}^T \mathbf{b}_j}{1+y_{0_j}^2} \right) \quad (34)$$

These constraints are homogeneous and non linear in the coefficients of $\mathbf{T}$. In order to avoid the trivial null solution the scale factor must be fixed in advance. For example, one may choose:

$$\|\mathbf{I}_{p_1}\|^2 = 1 + x_{0_1}^2$$

or:

$$\|\mathbf{J}_{p_1}\|^2 = 1 + y_{0_1}^2$$

Hence we obtain one additional constraint:

$$\mathbf{a}_1^T \mathbf{T}\mathbf{T}^T \mathbf{a}_1 = 1 + x_{0_1}^2 \quad (35)$$

or:

$$\mathbf{b}_1^T \mathbf{T}\mathbf{T}^T \mathbf{b}_1 = 1 + y_{0_1}^2 \quad (36)$$

These constraints are non linear in the coefficients of $\mathbf{T}$. With the substitution:

$$\mathbf{Q} = \mathbf{T}\mathbf{T}^T$$

equations (33), (34), and (35) (or (36)) become linear and there are 6 unknowns because, by definition, $\mathbf{Q}$ is a 3×3 symmetric positive matrix. Since we have $2k+1$ independent equations and 6 unknowns, at least 3 views are necessary to estimate $\mathbf{Q}$. Finally $\mathbf{T}$ can be derived from $\mathbf{Q}$ using a factorization of $\mathbf{Q}$. As it will be explained later in section 5 there is an ambiguity associated with the factorization of the symmetric semi-definite positive matrix $\mathbf{Q}$ and this ambiguity is the origin of the reversal ambiguity associated with reconstruction with an affine camera model.

Once the matrix $\mathbf{T}$ is determined, one can easily estimate Euclidean shape, $P_i = \mathbf{T}^{-1} S_i$ and rigid motion, $R_j = \mathbf{A}_j \mathbf{T}$. We are therefore left with the estimation of the rigid transformations between the scene frame and each camera frame.

Next we determine the parameters of the Euclidean motion by taking explicitly into account the paraperspective camera model. The method presented below is an alternative to the method proposed in [15] and it is equivalent to the problem of computing pose with a paraperspective camera [12].

First we determine the translation vector. From the formulae above we have:

$$t_{z_j} = \frac{1}{2}\left(\frac{\sqrt{1+x_{0_j}^2}}{\|\mathbf{I}_{p_j}\|} + \frac{\sqrt{1+y_{0_j}^2}}{\|\mathbf{J}_{p_j}\|}\right)$$

and:

$$t_{x_j} = x_{0_j} t_{z_j}$$
$$t_{y_j} = y_{0_j} t_{z_j}$$

Second, we derive the three unit vectors $\mathbf{i}_j$, $\mathbf{j}_j$, and $\mathbf{k}_j$ as follows. In theory these three vectors are mutually orthogonal but in practice there is no guarantee that they are. Our method does not assume that $\mathbf{i}_j$ and $\mathbf{j}_j$ are orthogonal but insures that the third vector is orthogonal to the first two. Equations (25) and (26) may be written as:

$$\mathbf{i}_j = t_{z_j}\,\mathbf{I}_{p_j} + x_{0_j}\,\mathbf{k}_j \tag{37}$$
$$\mathbf{j}_j = t_{z_j}\,\mathbf{J}_{p_j} + y_{0_j}\,\mathbf{k}_j \tag{38}$$

The third vector, $\mathbf{k}_j$ is the cross-product of these two vectors:

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

Let's, for convenience, drop the subscript j. We obtain for $\mathbf{k}$:

$$\mathbf{k} = t_z^2\,\mathbf{I}_p \times \mathbf{J}_p + t_z y_0\,\mathbf{I}_p \times \mathbf{k} - t_z x_0\,\mathbf{J}_p \times \mathbf{k}$$

Let $S(\mathbf{u})$ be the skew-symmetric matrix associated with a 3-vector $\mathbf{u}$, and $I_{3\times3}$ be the identity matrix. It is well known that the cross product $\mathbf{u}\times\mathbf{v}$ can be written as $S(\mathbf{u})\mathbf{v}$. The previous expression can now be written as follows:

$$\underbrace{\left(I_{3\times3} - t_z y_0\,S(\mathbf{I}_p) + t_z x_0\,S(\mathbf{J}_p)\right)}_{\mathbf{B}}\,\mathbf{k} = t_z^2\,\mathbf{I}_p \times \mathbf{J}_p \tag{39}$$

This equation allows us to compute $\mathbf{k}$, provided that the linear system above has full rank. Indeed, one may notice that the 3×3 matrix $\mathbf{B}$ is of the form:

$$\mathbf{B} = \begin{pmatrix} 1 & c & -b \\ -c & 1 & a \\ b & -a & 1 \end{pmatrix}$$

Its determinant is strictly positive:

$$det(B) = 1 + a^2 + b^2 + c^2$$

Therefore, B has full rank and one can easily determine $\mathbf{k}_j$ using eq. (39) and $\mathbf{i}_j$ and $\mathbf{j}_j$ using eqs. (37) and (38). As a consequence, it is always possible to compute the rigid motion between each camera position and the 3-D scene, i.e., $\mathbf{i}_j$, $\mathbf{j}_j$, $\mathbf{k}_j$, t_{x_j}, t_{y_j}, and t_{z_j} and to estimate ε_{ij} for each image and for each point (eq. (30)).

5 Solving the reversal ambiguity

The algorithm outlined in Section 3 solves for Euclidean reconstruction with a perspective camera by iterations of an Euclidean reconstruction method with either a weak perspective or a paraperspective camera. In this section we show how this iterative algorithm has to be modified in order to solve the reversal ambiguity problem which is inherent with any affine camera model. Indeed, let us consider again the affine shape and motion recovery method outlined in the previous section. A key step of this method consists of computing a transformation T that converts affine structure into Euclidean structure. This transformation must be computed by decomposition of a symmetric semi-definite positive matrix $\mathbf{Q}$:

$$\mathbf{Q} = \mathbf{T}\mathbf{T}^T$$

There are at least two ways to determine $\mathbf{T}$:

1. $\mathbf{Q}$ can be written as:

   $$\mathbf{Q} = \mathbf{O}\mathbf{D}\mathbf{O}^T$$

   where O is an orthogonal matrix containing the eigenvectors of $\mathbf{Q}$ and D is a diagonal matrix containing the eigenvalues of $\mathbf{Q}$. Since the eigenvalues of a symmetric semi-definite positive matrix are all real and positive, one may write $\mathbf{Q}$ as:

   $$\mathbf{Q} = (\mathbf{O}\mathbf{D}^{1/2})(\mathbf{O}\mathbf{D}^{1/2})^T = \mathbf{K}\mathbf{K}^T$$

2. Alternatively one may use the Cholesky decomposition of $\mathbf{Q}$:

   $$\mathbf{Q} = \mathbf{L}\mathbf{L}^T$$

   where $\mathbf{L}$ is a lower triangular matrix.

Let $\mathbf{H}$ be a non singular matrix such that:

$$\mathbf{L} = \mathbf{K}\mathbf{H}$$

and we have:

$$\begin{aligned} \mathbf{Q} &= \mathbf{L}\mathbf{L}^T \\ &= \mathbf{K}\mathbf{H}\mathbf{H}^T\mathbf{K}^T \\ &= \mathbf{K}\mathbf{K}^T \end{aligned}$$

We conclude that $\mathbf{H}$ is necessarily an orthogonal matrix. The orthogonality of $\mathbf{H}$ is also claimed in [28] but without any formal proof. Therefore $\mathbf{H}$ represents either a rotation or a mirror transformation (its determinant is either $+1$ or -1) and there are two classes of shapes that are possible:

- a *direct* shape which is defined up to a rotation and
- a *reverse* shape which is obtained from the direct shape by appling a mirror transformation.

Since shape is defined up to rotation and without loss of generality we choose the mirror transformation to be $-\mathbf{I}$ where $\mathbf{I}$ is the identity matrix. Therefore the affine shape and motion equation can be written as:

$$\boldsymbol{\sigma} = \mathbf{A}\mathbf{S} = (-\mathbf{A})(-\mathbf{S})$$

Because of this reversal ambiguity, there are two solutions for the ε_{ij}'s at each iteration of the reconstruction algorithm described above.

We consider the case of paraperspective. A similar treatment is possible for weak perspective. The vectors $\mathbf{k}_j$ are computed using eq. (39). This equation may use either $\mathbf{I}_p$ and $\mathbf{J}_p$ (the first solution) or $-\mathbf{I}_p$ and $-\mathbf{J}_p$ (the second solution). Therefore we obtain two distinct solutions, that is, $\mathbf{k}_j^1$ and $\mathbf{k}_j^2$. The two solutions for ε_{ij} correspond to $\mathbf{k}_j^1$ and $\mathbf{P}_i$ and to $\mathbf{k}_j^2$ and $-\mathbf{P}_i$:

$$\varepsilon_{ij}^{1,2} = \pm \frac{\mathbf{k}_j^{1,2} \cdot \mathbf{P}_i}{t_{z_j}}$$

At each iteration of the perspective reconstruction algorithm two values for ε_{ij} are thus estimated. Therefore, after N iterations there will be 2^N possible solutions. All these solutions are not, however, necessarily consistent with the image data and a simple verification technique allows to check this consistency and to avoid the explosion of the number of solutions. Finally, a unique solution is obtained.

The first iteration of the algorithm makes available two solutions – a "positive" shape S and a "negative" shape $(-S)$ – that are both considered. At the next iterations of the algorithm two shapes are maintained: one shape consistent with S and another shape consistent with $-S$. Therefore, at convergence, one obtains two solutions, each one of these solutions being consistent with one or the other of the initial shapes. Finally, the solution that best fits the image data is selected as the unique solution. This solution selection process is best illustrated in Figure 3.

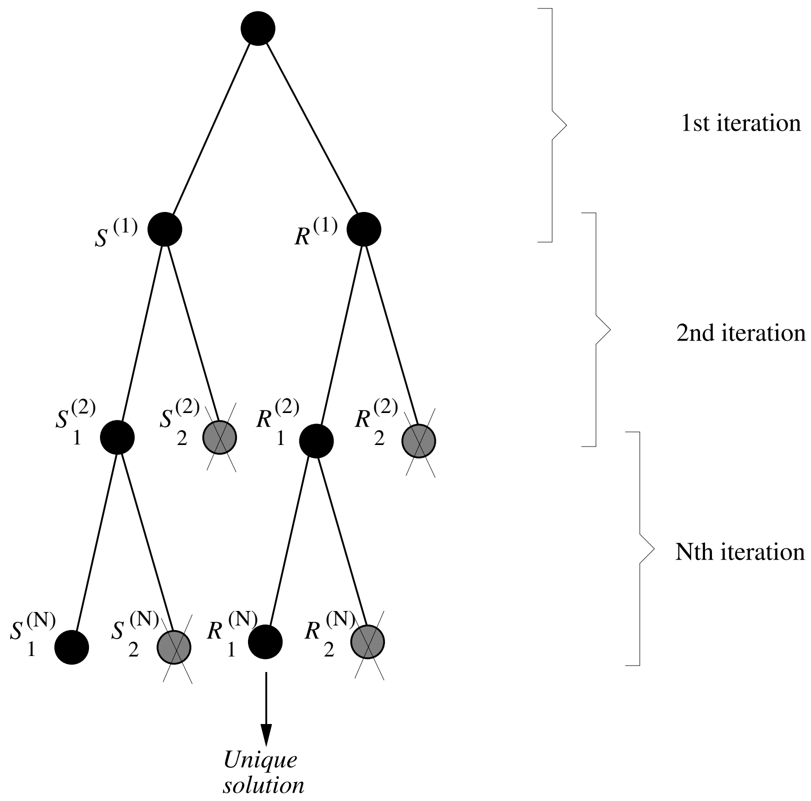

Figure 3: A strategy for selecting a unique solution (see text).

More formally, let $S^{(1)}$ be the positive shape computed at the first iteration of the algorithm and $R^{(1)}$ be the negative shape ($R^{(1)} = -S^{(1)}$). At each one of the next iterations one has to deal with four shapes: $S_1^{(i)}$ and $S_2^{(i)}$ that are issued from the positive solution and $R_1^{(i)}$ and $R_2^{(i)}$ that are issued from the negative solution. The S-shape and the R-shape the most consistent with the shapes selected at the previous iterations are selected. Finally, a unique solution is selected on the basis of consistency with the image data.

The solution selection process just described is illustrated in Figure 1-bottom and on Figure 4. The algorithm first computes two affine reconstructions (Figure 1-bottom): a "positive" shape S (bottom-left) and a "negative" shape R (bottom-right). At convergence, the algorithm comes up with two shapes, one that is issued from S (Figure 4-left) and one that is issued from R (Figure 4-right). Eventually, the solution issued from the R family of shapes (right) is the one that is the most consistent with the image data and selected as the unique solution of the perspective reconstruction algorithm.

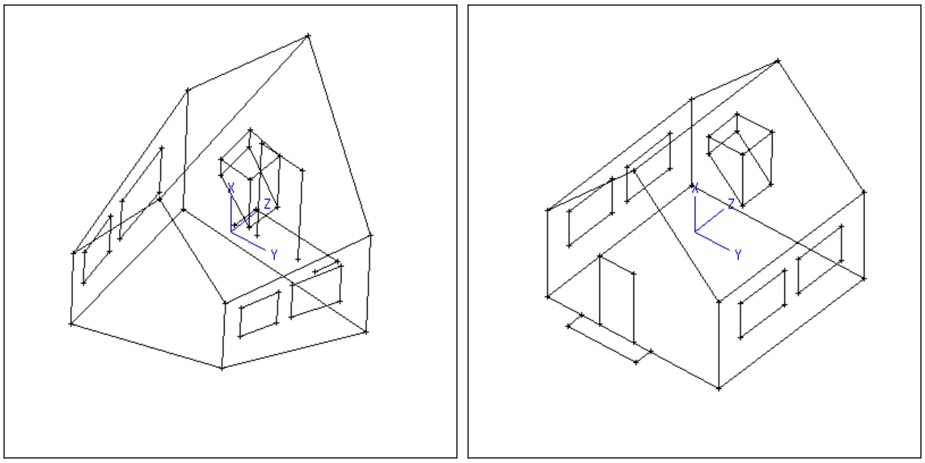

Figure 4: Solving for the reversal ambiguity: the initial affine reconstructions become two perspective reconstructions but only one among them is consistent with the image data.

6 Method analysis

6.1 Sensitivity to camera calibration

So far we assumed that the camera is calibrated, which means that the intrinsic camera parameters are known, i.e., α_u, α_v, u_c, and v_c in eqs. (2) and (3). It is well known that camera calibration is difficult and that the camera parameters are not stable over time and space. It is also known that the only stable intrinsic camera parameter is the ratio between the horizontal and vertical pixel size:

$$\gamma = \frac{\alpha_u}{\alpha_v}$$

We consider again the perspective equations (21) and (22). By combining them with equations (2) and (3) we obtain:

$$u_i(1+\varepsilon_i) - u_0 - u_c\varepsilon_i = \gamma\alpha_v \mathbf{I}\cdot\mathbf{P}_i \tag{40}$$
$$v_i(1+\varepsilon_i) - v_0 - v_c\varepsilon_i = \alpha_v \mathbf{J}\cdot\mathbf{P}_i \tag{41}$$

By inspecting the above equations it is straightforward to notice that the intrinsic parameter α_v acts as a scale factor on the 3-D shape. Since shape can be recovered only up to a similarity transformation, exact knowledge of the value of α_v does not affect the solution.

Moreover, the remaining intrinsic parameters, u_c and v_c, are weighted by ε_i which is the ratio between the size of the object measured along the optical axis divided by the distance between the object and the optical center of the camera. The absolute value of this ratio (ε_i) varies between 0 (the object is very far) and a positive value that insures that the object doesn't "bump" into the lens of the camera. In practice a maximum value of ε_i may be 0.5 but more realistic values are in the range [0.1, 0.2]. Therefore, the calibration errors committed on u_c and v_c are scaled down by the same factor.

To conclude, errors in camera calibration have only a weak effect on the perspective reconstruction algorithm outlined in this paper. The only intrinsic camera parameter for which an exact value is needed, is the ratio between the horizontal pixel size and vertical pixel size. The exact pixel size is not required because the 3-D shape is recovered up to a scale factor and the image coordinates of the center of projection are "scaled down" by ε_i.

6.2 An analysis of convergence

In order to analyse the convergence of the iterative reconstruction algorithm outlined in Section 3 we consider separately the equations associated with a weak perspective camera model and with a paraperspective camera model. It is quite difficult to analyse the convergence of such algorithms from a theoretical point of view. Therefore we base our analysis on numerical considerations by analysing the size of the neglected terms relative to those that are retained.

Consider the equations (21), (22) and (23), (24). Both these sets of equations describe the *full* perspective projection with a calibrated camera. The first ones allow to express the perspective reconstruction problem in terms of an *iterative weak perspective* algorithm while the second ones allow to express the same problem in terms of an *iterative paraperspective* algorithm. If good estimates for the values of ε_i are available, then the perspective reconstruction problem is reduced to an affine reconstruction problem. Of course, in practice it seems difficult to provide such estimates. In this section we show that initializing ε_{ij} to zero provides sensible initial estimates even if the object is quite close to the camera. This explains why the iterative algorithms converge in a few iterations. We start by analysing the weak perspective case and then we extend our analysis to paraperspective.

Let, for n points and k views, $\varepsilon^{\star}_{ij}$ be the true values that the iterative algorithm is supposed to eventually estimate. At the first iteration, the algorithm performs a standard reconstruction using a weak perspective camera model, i.e., it computes shape and motion using the affine shape and motion equation (32) followed by Euclidean normalization.

Therefore, the "reconstruction errors" are proportional to the "weak perspective errors" $|x_{ij}\varepsilon^{\star}_{ij}|$ and $|y_{ij}\varepsilon^{\star}_{ij}|$, i.e., eqs. (13) and (14). If these errors are large, the weak perspective solution estimated at the first iteration of the algorithm will be rather different than the solution being sought and convergence cannot be guaranteed. In their work on pose estimation, Dementhon & Davis [7] noticed that the iterative weak perspective algorithm converges even if the expected values for ε_{ij} are as large as 1, *provided that the scene points lie in the neighborhood of the optical axis.* Indeed, when the scene points are close to the optical axis, the camera coordinates of their projections, x_{ij} and y_{ij}, are small (the origin of the camera frame lies onto the optical axis) and they compensate for the large values of $\varepsilon^{\star}_{ij}$ (see Section 2.1). Moreover, as it has been discussed in the previous Section, the most realistic maximum value for $\varepsilon^{\star}_{ij}$ is 0.5.

The iterative paraperspective algorithm is able to deal with configurations where the weak perspective algorithm diverges. Indeed, in the case of paraperspective, the initial errors are $|(x_{ij} - x_{0j})\varepsilon^{\star}_{ij}|$ and $|(x_{ij} - x_{0j})\varepsilon^{\star}_{ij}|$, i.e., eqs. (19) and (20). Whenever the point p_0 is properly chosen (typically, it should be the center of mass of the set of image points), then the differences $(x_{ij} - x_{0j})$ and $(y_{ij} - y_{0j})$ are small and they compensate for large values of $\varepsilon^{\star}_{ij}$. Therefore the iterative paraperspective reconstruction algorithm is likely to converge over a wider range of configurations than the weak perspective one.

6.3 A comparison with non linear minimization methods

In the past, a number of authors have tried to solve the Euclidean structure and motion problem using non linear optimization techniques. It its most general form the problem is to minimize the following error function [5], [20], [3], [19]:

$$f(X) = \sum_{ij} \left((x_{ij} - \hat{x}_{ij})^2 + (y_{ij} - \hat{y}_{ij})^2 \right)$$

where x_{ij} and y_{ij} are the coordinates of an image point and $\hat{x}_{ij}$ and $\hat{y}_{ij}$ are the coordinates of the projected 3-D point using a perspective camera model: the latter coordinates are therefore given by eqs. (4) and (5).

For n points and k images, the error function above has $2 \times n \times k$ positive squared terms. The vector X encapsulates the unknowns of the problem: $3 \times n$ coordinates and $6 \times k$ motion parameters. We seek a value for X which minimizes the error function. The Jacobian of $f(X)$ is a $m \times p$ matrix and whe have:

$$\begin{aligned} m &= 2 \times n \times k \\ p &= 3 \times n + 6 \times k \end{aligned}$$

Any non-linear minimization method searches the minimum incrementally and at each iteration the following linear system must be solved in order to compute dX and replace X by $X + dX$:

$$\mathbf{J}^T(X)\mathbf{J}(X)\, dX = \mathbf{b}$$

Therefore, the complexity of each iteration is dominated by the complexity of inverting a symmetric definite positive matrix – the Hessian. The size of the Hessian matrix is $p \times p$ and therefore it is a function of n and k. Furthermore, if one takes into account the fact that the Hessian is a banded matrix, the complexity of inverting the Hessian is of $p^3 + 8\,p^2 + p$ flops [9]. By replacing p with its expression we obtain the following complexity:

$$27\,n^3 + 162\,n^2k + 324\,nk^2 + 216\,k^3 + 72\,n^2 + 288\,nk + 288\,k^2 + 3\,n + 6\,k$$

If we retain the third-order terms we obtain:

$$27\,n^3 + 162\,n^2k + 324\,nk^2 + 216\,k^3 \tag{42}$$

In order to compare our iterative method with such a non linear method, let's compute the complexity of one iteration. The most time-consuming part of the algorithm is the singular value decomposition of the measurement matrix $\boldsymbol{\sigma}$ of size $2k \times n$. Therefore, the complexity of singular value decomposition is [9]:

$$22\,n^3 + 8\,nk^2 \tag{43}$$

Method	$k \approx n/10$	$k \approx n$	$k \approx 10n$
Factorization	$22n^3$	$30n^3$	$822n^3$
Non-linear (one iteration)	$46n^3$	$729n^3$	$250000n^3$

Table 1: This table shows the number of float operations as a function of the number of points (n) and of the number of images (k). If the number of images is small compared with the number of points then the two methods have comparable complexity. Nevertheless, if the number of images increases, the non-linear minimization methods requires much more computation.

The complexities of the factorization method and of the non-linear minimization methods are shown in Table 1 for three cases: the number of images (k) is much smaller than the number of points (n), the number of images is approximatively equal to the number of points, and the number of images is much larger than the number of points.

The above comparison holds for one iteration. We can conclude that the method proposed in this paper is *intrinsically* more efficient than a non linear minimization method. Notice the dramatic increase in complexity of the non linear method when the number of images is larger than the number of points.

7 Simulation experiments

In this section we study the performance of the affine iterative algorithms and we compare them with the factorization method. Two types of performances are studied: (i) the accuracy of 3-D reconstruction as a function of various types of motions and in the presence of image (pixel) noise and (ii) the convergence of the affine iterative algorithms as a function of various types of motion.

In all the experiments described in this section we used the following common features:

- The intrinsic camera parameters are: $u_c = v_c = 256$, $\alpha_u = \alpha_v = 1000$.
- There are 15 images and 42 points (the synthetic data being used is shown on Figure 1).
- For all the considered motions the angular variation between each view is of 2^0 and therefore, the total angular variation is of 30^0,
- Gaussian noise with maximum standard deviation $\sigma = 1$ has been added to the image measurements and there are 200 trials for each experiment, i.e., each experiment has been run 200 times with various noise levels.
- One important parameter for each experiment is the average distance between the 3-D points and the camera. Let D be the distance from the center of gravity of the set of 3-D points to the center of projection, *divided* by the diameter of the 3-D point set – D is therefore unitless and we call it a *relative distance*. Notice that D is approximatively equal to $1/\overline{\varepsilon_j}$ where $\overline{\varepsilon_j}$ is the average value for all the points.

- For those experiments for which D varies, the maximum variation is limited to 5.
- The quality of a reconstruction result is evaluated by the mean and maximum of the Euclidean distance between the theoretical 3-D points and the estimated 3-D points (the scale factor being fixed accordingly) and the mean and maximum of the difference between the theoretical angle values between pairs of 3-D edges and the angle values between estimated pairs of 3-D edges.

The experiments below compare the iterative paraperspective algorithm with the paraperspective factorization algorithm, the latter being simply the first iteration of the former. Figures 5 through 10 show the quality of the 3-D Euclidean reconstruction for various motion types (motion parallel to the image plane and motions parallel to the optical axis). The behaviour of the reconstruction algorithm does not seem to depend on the direction of motion.

Figures 11 and 12 study the behaviour of the two iterative algorithms (weak perspective and paraperspective) when the motion is parallel to the optical axis and when the center of gravity of the 3-D point set is at a fixed offset away from the optical axis. On an avarage, the paraperspective iterative algorithm requires less iterations than the weak perspective iterative algorithm. The convergence rate of both algorithms is close to 100% for a relative distance greater than 4. When the relative distance is equal to 3, the convergence rate of both algorithms drops to 75%. It is worthwhile to notice that relative distances smaller than 3 are not realistic in practice, because, in this case, partial occlusion of the point set becomes predominant.

Finally, Figures 13 and 14 compare the accuracy obtained with the two iterative algorithms.

8 Real imagery experiments

In this section we consider several examples obtained with real images and with the paraperspective iterative algorithm:

- A sequence of 5 images of a cube where 38 points were tracked over the image sequence (Figure 15);
- A sequence of 5 images of a house with 46 tracked points (Figure 16), and
- A sequence of 10 images of a wood piece with 10 tracked points (Figure 17);

In all these experiemnts the camera center was fixed to $u_c = v_c = 256.$ and the horizontal and vertical scale factors were fixed to $\alpha_u = 1500$ and $\alpha_v = 1000$. The camera motion was a general motion. One may easily notice the large discrepancy between the reconstruction results obtained with the factorization method and with the affine iterative method – this discrepancy is visible especially when the data is shown from above such that the perspective effect associated with visualization is almost null.

Table 2 summarizes the performances of the paraperspective iterative algorithm obtained both with synthetic and real imagery.

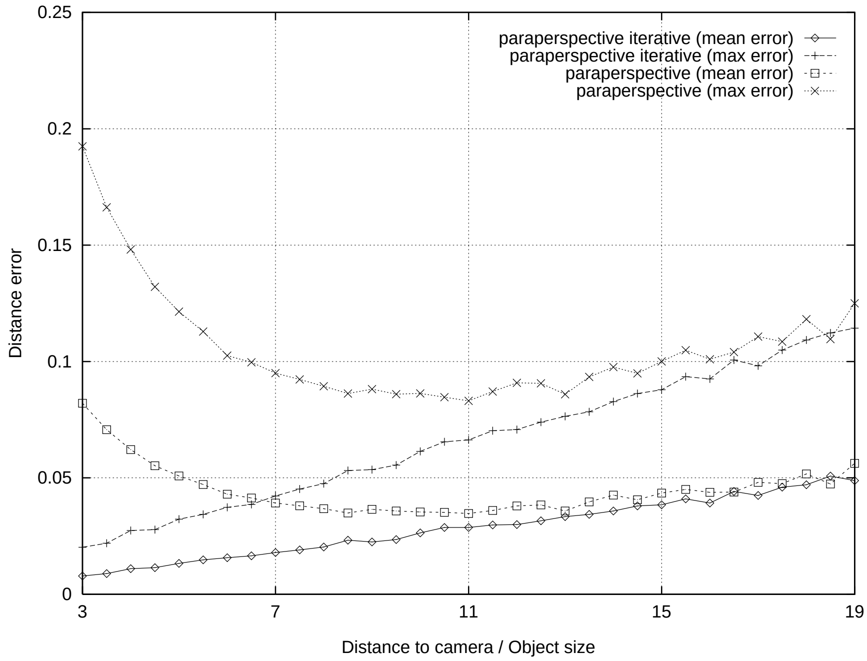

Figure 5: The quality of reconstruction as a function of D (see text) when the motion direction is parallel to the image plane.

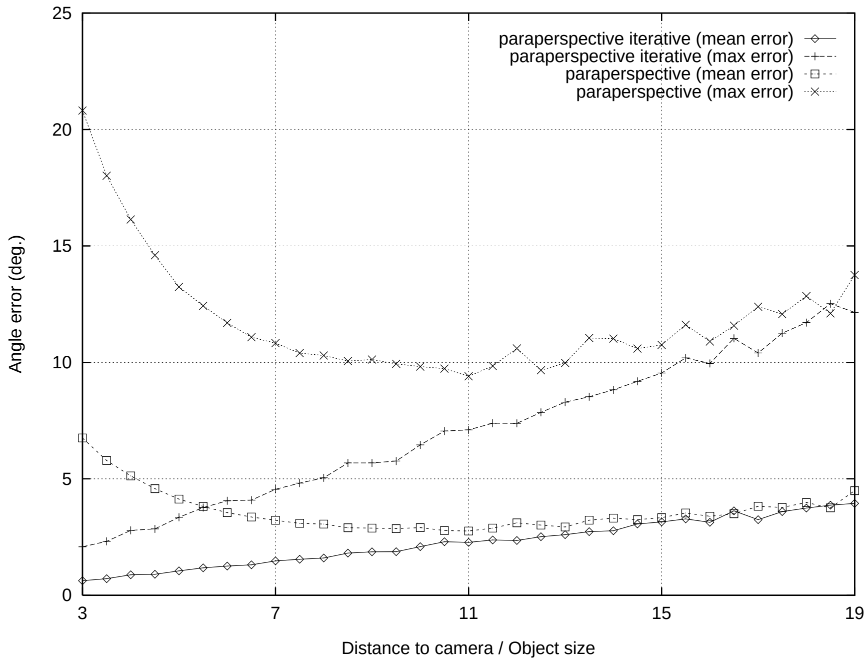

Figure 6: The quality of reconstruction (angle measurements) as a function of D (see text) when the motion direction is parallel to the image plane.

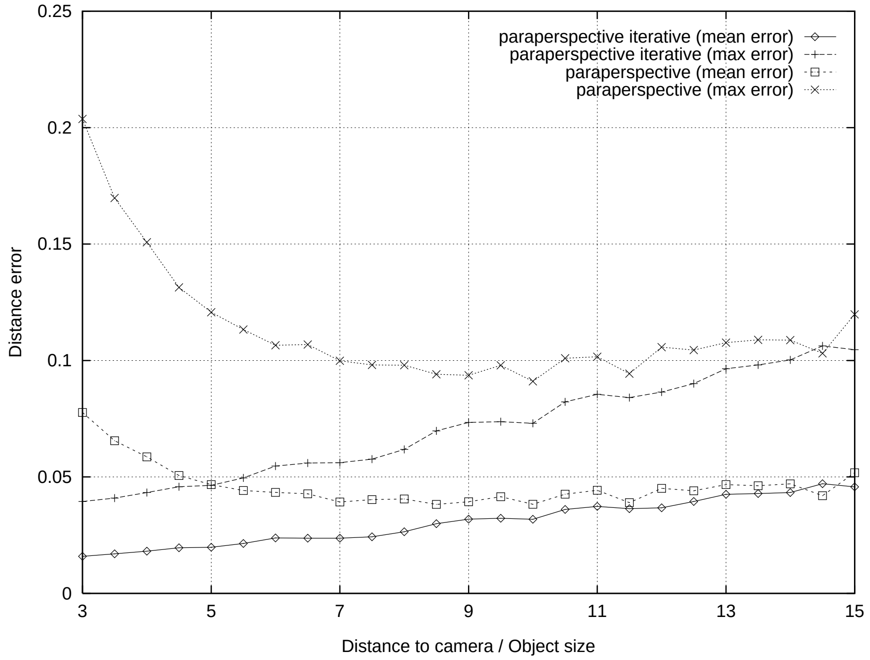

Figure 7: The quality of reconstruction as a function of D when the motion is towards the camera (roughly parallel to the optical axis), the depth variation being equal to 5.

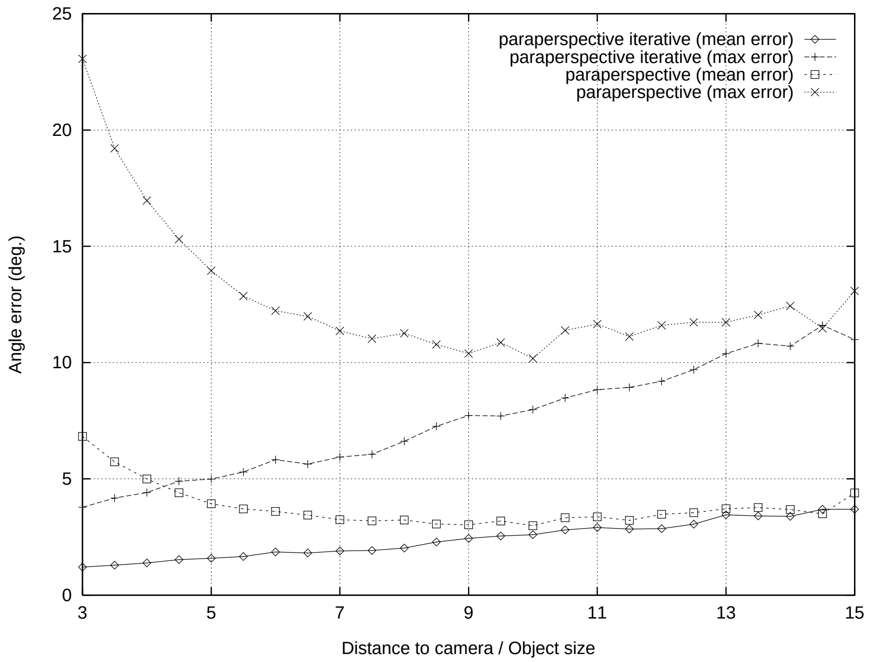

Figure 8: The quality of reconstruction (angle measurements) as a function of D when the motion is towards the camera (roughly parallel to the optical axis), the depth variation being equal to 5.

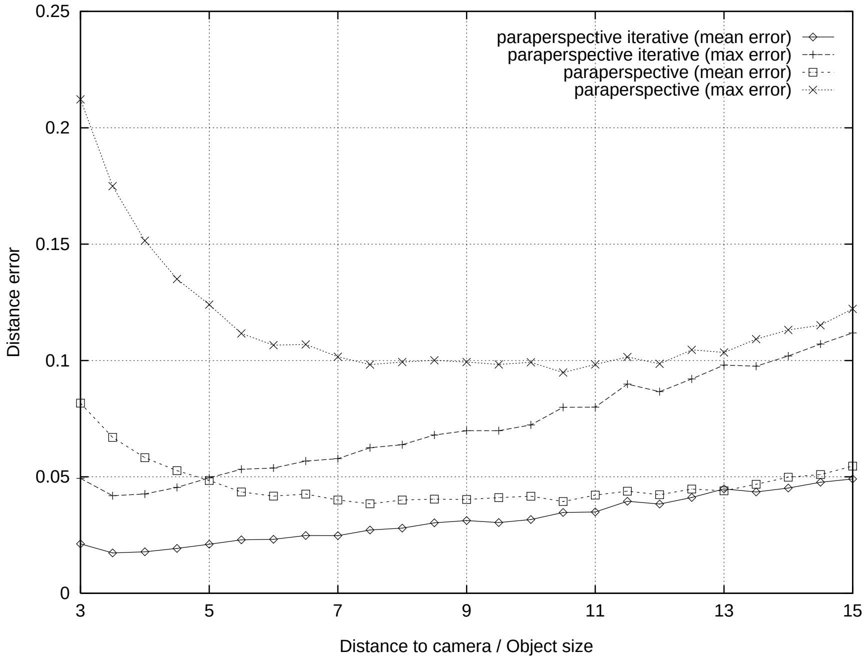

Figure 9: The quality of reconstruction as a function of D when the motion is towards the camera, roughly parallel to the optical axis, and the center of gravity of the point set is at a fixed offset away from the optical axis.

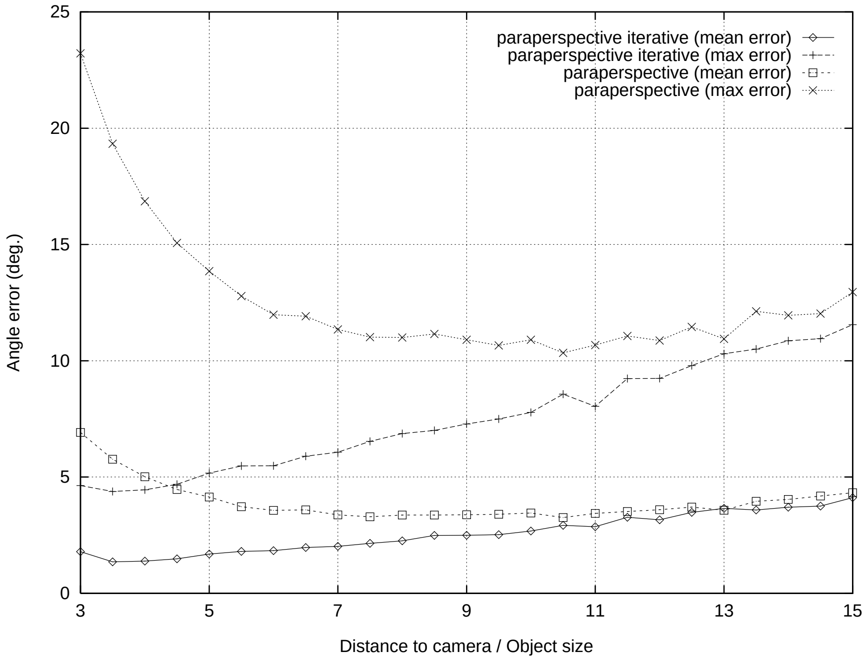

Figure 10: Same as above but for angle measurements.

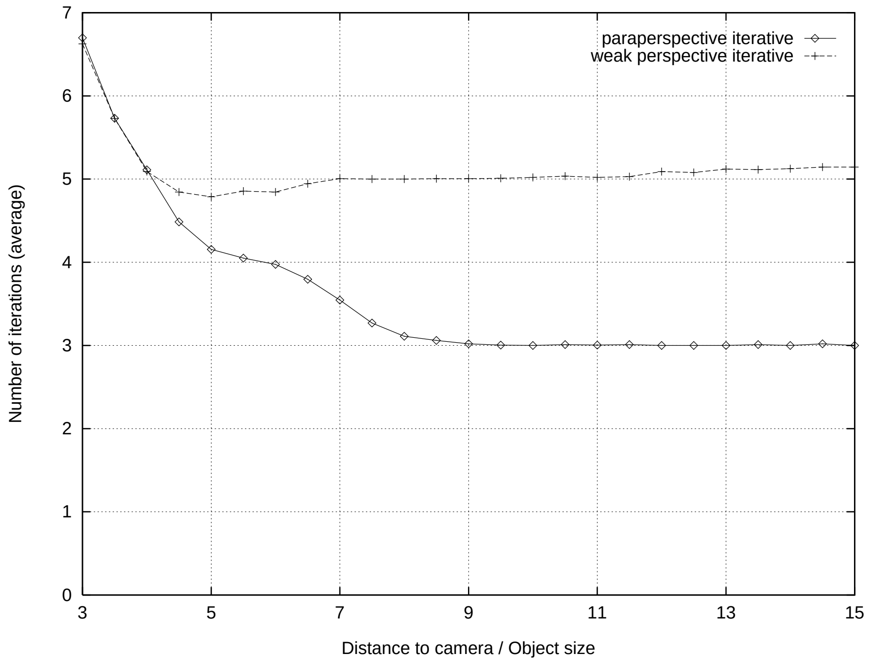

Figure 11: Number of iterations (weak perspective and paraperspective) as a function of D where the motion is towards the camera and the center of gravity of the point set is at a fixed offset away from the optical axis.

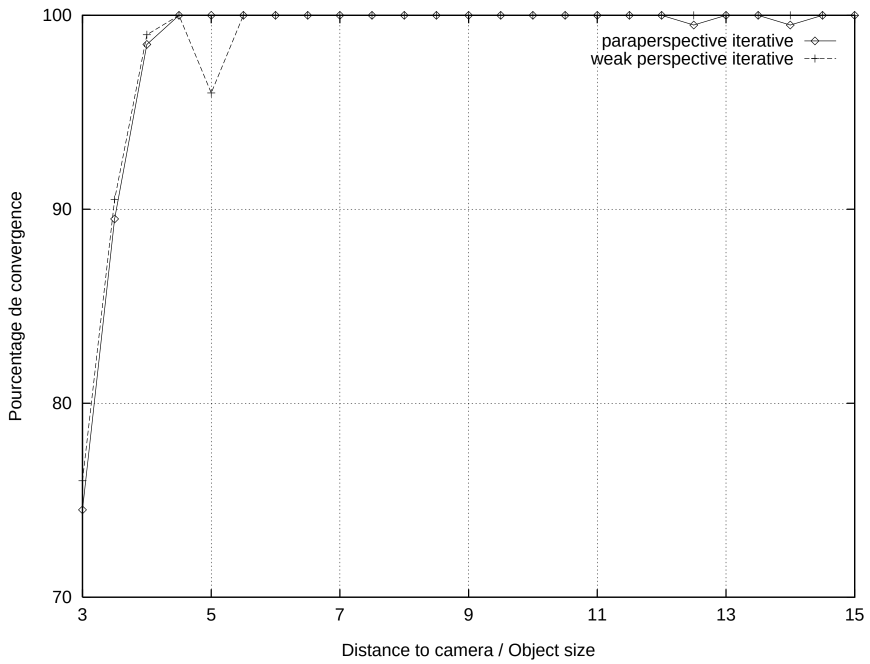

Figure 12: The rate of convergence of the two iterative algorithms as a function of D where the motion is towards the camera and the center of gravity of the point set is at a fixed offset away from the optical axis.

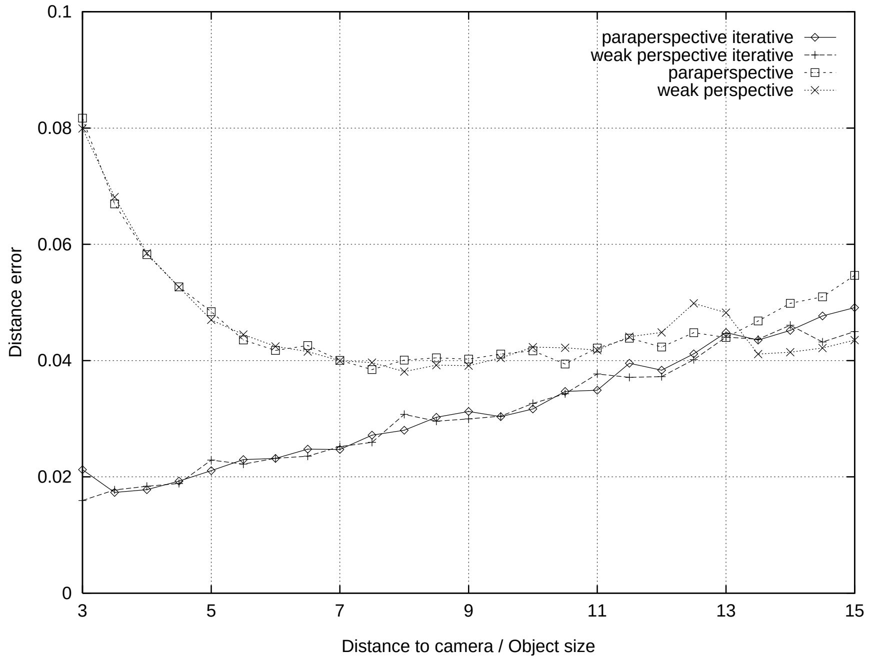

Figure 13: Comparison between the accuracy associated with the two iterative algorithms, the motion being parallel and at a fixed offset from the optical axis.

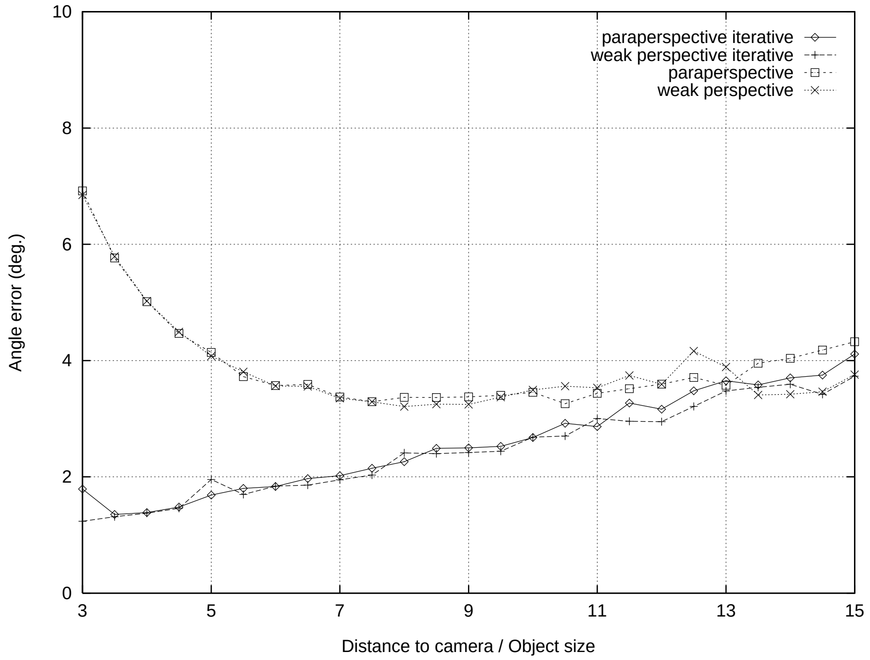

Figure 14: Comparison between the accuracy associated with the two iterative algorithms (angle measurements).

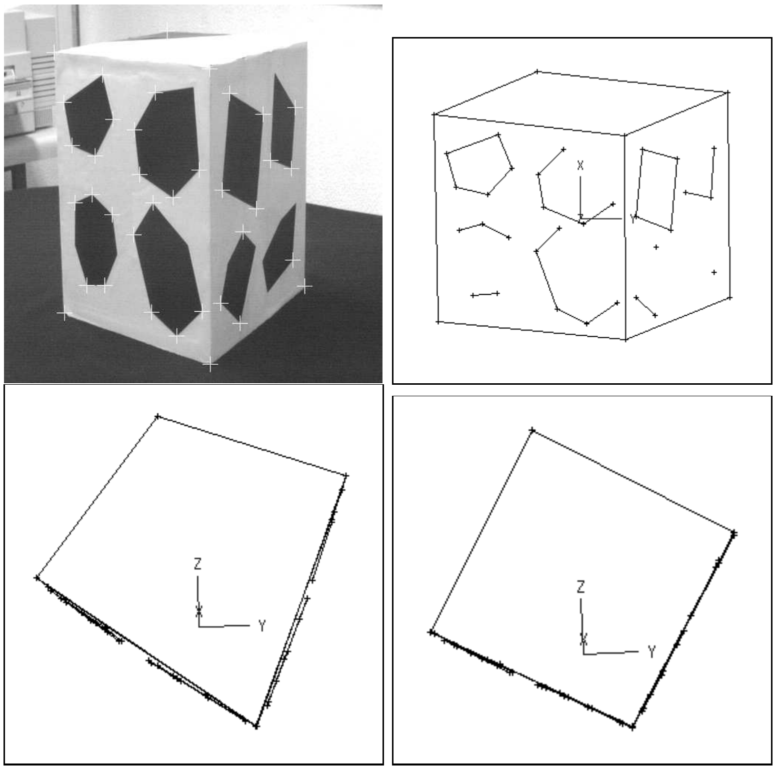

Figure 15: This figure shows an image (top-left) out of a sequence of 5 images grabbed with a moving camera and the result of reconstructing 38 points with the iterative method (top-right). Top views of the reconstructed scene allow to compare more quantitatively the result of the factorization method (bottom-left) with the result of the iterative method (bottom-right) described in this paper.

N. of points	N. of images	CPU time/iteration	N. of iterations	image sequence
38	5	0.31	5	"cube"
10	10	0.37	4	"wood piece"
42	15	1.50	4	synthetic

Table 2: Summary of the performances of the paraperspective iterative algorithm. The CPU time is in seconds and it was obtained on a Sun/Sparc10/SunOs platform.

9 Discussion

In this paper we described a method for solving the Euclidean reconstruction problem with a perspective camera by incrementally performing an Euclidean reconstruction with either a weak or a paraperspective camera model. The method converges – on an average – in 5 iterations, is computationally more efficient than non-linear minimization methods, and it produces accurate results even in the presence of image noise and/or camera calibration errors. The method may well be viewed as a generalization to perspective of shape and motion computation using factorization and/or affine-invariant methods. It is well known that with a linear camera model, shape and motion can be recovered only up to a sign (reversal) ambiguity. The method that we propose in this paper solves for this ambiguity and produces a uniqe solution even if the camera is at some distance from the scene.

Although the experimental results show that there are few convergence problems, we have been unable to study the convergence of the algorithm from a theoretical point of view. We studied its convergence based on some numerical and practical considerations which allow one to determine in advance the optimal experimental setup under which convergence can be guaranteed. Indeed the experiments that we carried out and which are described in detail above (section 7) show that convergence does not depend upon the motion that the camera undergoes with respect to the scene. The algorithm fails to converge when there are scene points very close to the camera. However such configurations are not desirable because they lead to occlusions. The paraperspective model has better convergence properties than the weak perspective one, mainly because it requires fewer iterations.

Although we have not performed such a comparison, it is clear that non-linear minimization algorithms provide more accurate results than our algorithm, simply because non linear methods are designed to minimize a least-squares error function at the cost, nevertheless, of a larger number of float operations (see Table 1). As already mentioned, our algorithm converges after 5 iterations (on an average) which compares favourably with the number of iterations associated with non linear minimization methods as it was reported by a number of authors [3], [2], [20], [10], [19].

Therefore, the class of iterative algorithms described in this paper are an excellent compromise between linear resolution techniques (affine camera) and non-linear minimization techniques (perspective camera), both in terms of quality of reconstruction and speed.

References

[1] Y. Aloimonos. Perspective approximations. *Image and Vision Computing*, 8(3):177–192, August 1990.

[2] B. Boufama, R. Mohr, and F. Veillon. Euclidean constraints for uncalibrated reconstruction. In *Proceedings Fourth International Conference on Computer Vision*, pages 466–470, Berlin, Germany, May 1993. IEEE Computer Society Press, Los Alamitos, Ca.

[3] T. J. Broida and R. Chellappa. Estimating the kinematics and structure of a rigid object from a sequence of monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):497–513, June 1991.

[4] S. Christy and R. Horaud. Euclidean shape and motion from multiple perspective views by affine iterations. Technical Report RR-2421, INRIA, December 1994. Available by anonymous ftp on imag.fr, directory pub/MOVI, file RR-2421.ps.gz.

[5] N. Cui, J. Weng, and P. Cohen. Extended structure and motion analysis from monocular image sequences. In *Proc. Third International Conference on Computer Vision*, pages 222–229, Osaka, Japan, December 1990.

[6] C. Debrunner and N. Ahuja. Motion and structure factorization and segmentation of long multiple motion image sequences. In *Proc. Second European Conference on Computer Vision*, pages 217–221, Santa Margherita Ligure, Italy, May 1992.

[7] D. F. DeMenthon and L. S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1/2):123–141, 1995.

[8] O. D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig. In G. Sandini, editor, *Computer Vision – ECCV 92, Proceedings Second European Conference on Computer Vision, Santa Margherita Ligure, May 1992*, pages 563–578. Springer Verlag, May 1992.

[9] G.H. Golub and C.F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, 1989.

[10] R. I. Hartley. Euclidean reconstruction from uncalibrated views. In Mundy Zisserman Forsyth, editor, *Applications of Invariance in Computer Vision*, pages 237–256. Springer Verlag, Berlin Heidelberg, 1994.

[11] R. I. Hartley. In defence of the 8-point algorithm. In *Proceedings Fifth International Conference on Computer Vision*, pages 1064–1070, Cambridge, Mass., June 1995. IEEE Computer Society Press, Los Alamitos, Ca.

[12] R. Horaud, S. Christy, F. Dornaika, and B. Lamiroy. Object pose: Links between paraperspective and perspective. In *Proceedings Fifth International Conference on Computer Vision*, pages 426–433, Cambridge, Mass., June 1995. IEEE Computer Society Press, Los Alamitos, Ca.

[13] R. Horaud, F. Dornaika, B. Lamiroy, and S. Christy. Object pose: The link between weak perspective, paraperspective, and full perspective. Technical Report RR-2356, INRIA, September 1994.

[14] J. Koenderink and A. van Doorn. Affine structure from motion. *J. Opt. Soc. Amer. A*, 8(2):377–385, 1991.

[15] C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. In Jan-Olof Eklundh, editor, *Computer Vision – ECCV 94, Proceedings Third European Conference on Computer Vision*, volume 2, pages 97–108. Springer Verlag, Stockholm, Sweden, May 1994.

[16] L. Quan. Self-calibration of an affine camera from multiple views. Technical Report RT 125, IMAG–LIFIA, December 1994.

[17] L. Quan. Self-calibration of an affine camera. In *Proc. of Sixth International Conference on Computer Analysis of Images and Patterns*, pages 448–455, Prague, Czech Republic, September 1995.

[18] L. S. Shapiro, A. Zisserman, and J. M. Brady. Motion from point matches using affine epipolar geometry. In Jan-Olof Eklundh, editor, *Computer Vision – ECCV 94, Proceedings Third European Conference on Computer Vision, Stockholm, Sweden, May 1994*, volume 2, pages 73–84. Springer Verlag, May 1994.

[19] R. Szelinski and S. B. Kang. Recovering 3-D shape and motion from image streams using nonlinear least squares. *Journal of Visual Communication and Image Representation*, 5(1):10–28, March 1994.

[20] C. J. Taylor and D. J. Kriegman. Structure and motion from line segments in multiple images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(11):1021–1032, November 1995.

[21] C. J. Taylor, D. J. Kriegman, and P. Anandan. Structure and motion in two dimensions from multiple images: A least squares approach. In *IEEE Workshop on Visual Motion*, pages 242–248, Princeton, New Jersey, USA, October 1991.

[22] C. Tomasi. *Shape and Motion from Image Streams: a Factorization Method.* PhD thesis, Carnegie Mellon University, Pittsburgh, Penn, USA, 1991.

[23] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[24] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, RA-3(4):323–344, August 1987.

[25] R. Y. Tsai and T. S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(1):13–27, January 1984.

[26] T. Viéville and Q-T. Luong. Computing motion and stucture in image sequences without calibration. In *Proc. of the Twelfth International Conference on Pattern Recognition*, pages 420–425, Jerusalem, Israel, October 1994. IEEE Computer Society Press.

[27] D. Weinshall. Model-based invariants for 3-d vision. *International Journal of Computer Vision*, 10(1):27–42, February 1993.

[28] D. Weinshall and C. Tomasi. Linear and incremental acquisition of invariant shape models from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):512–517, May 1995.

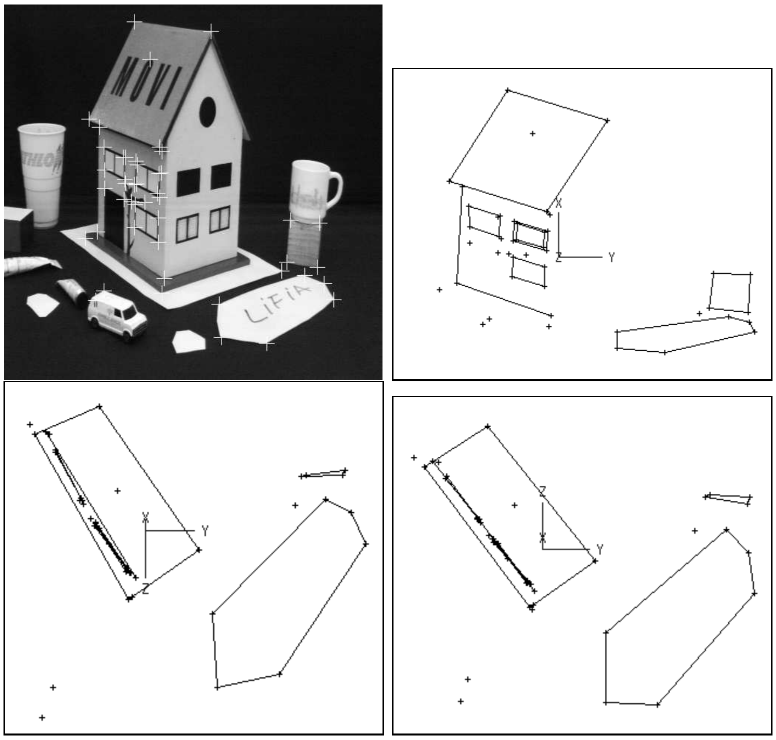

Figure 16: Same as the previous figure for another sequence of 5 images and 46 points.

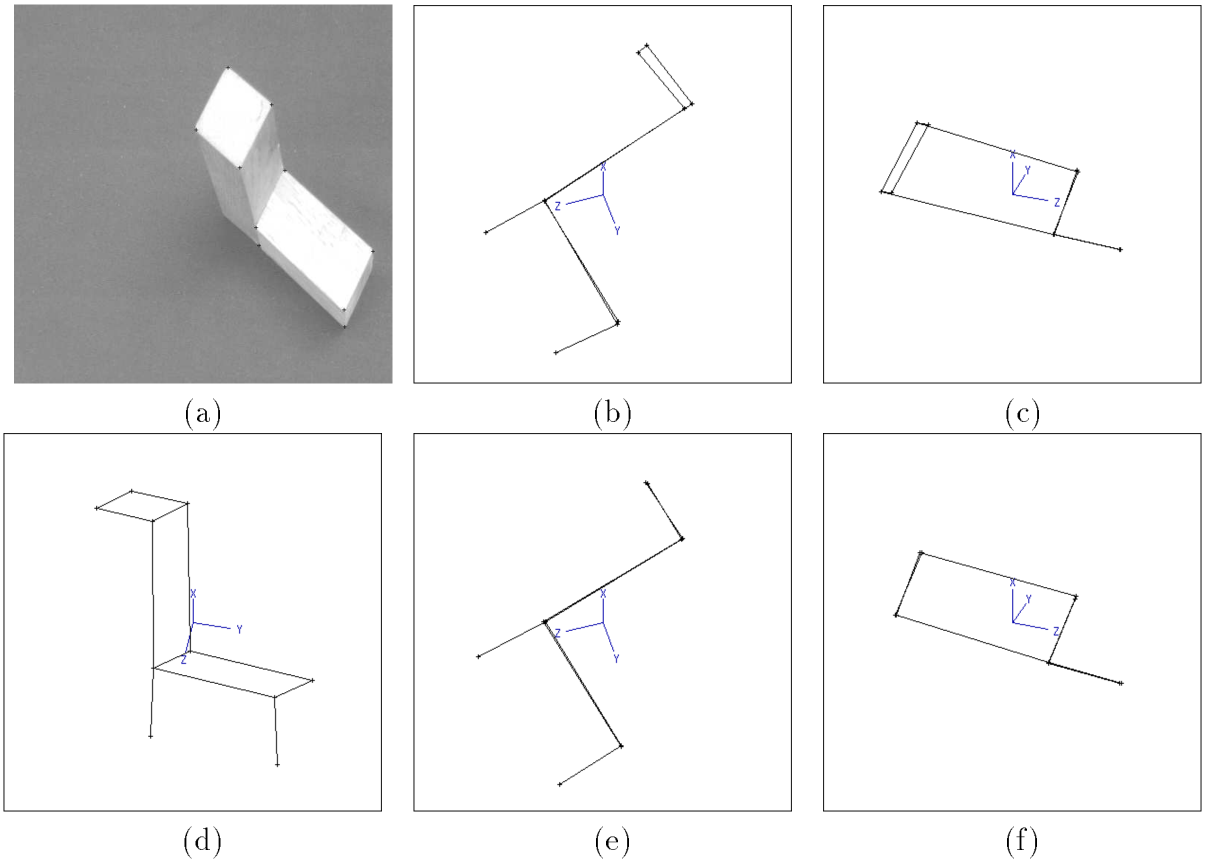

Figure 17: This figure shows one image (a) out of a sequence of 10 images where only 10 points were tracked and reconstructed. The first row (b) and (c) shows the result of reconstruction using the factorization method with a paraperspective model, while the second row (d), (e), and (f) shows the result of reconstruction with the iterative method and a perspective model. In this example the iterative algorithm converged in 4 iterations.