Globally Convergent Autocalibration using Interval Analysis

A. Fusiello, A. Benedetti, M. Farenzena, A. Busti.

A.Fusiello, and M Farenzena are with the Dipartimento di Informatica, Università degli studi di Verona, I-37134 Verona, Italy. Email: fusiello,farenzen@sci.univr.it

A. Benedetti is Currently affiliated with KLA-Tencor, One Technology Dr., Milpitas, CA 95035. Email: Arrigo.Benedetti@kla-tencor.com.

Abstract

We address the problem of autocalibration of a moving camera with unknown constant intrinsic parameters. Existing autocalibration techniques use numerical optimization algorithms whose convergence to the correct result cannot be guaranteed, in general. To address this problem, we have developed a method where an interval branch-and-bound method is employed for numerical minimization. Thanks to the properties of Interval Analysis this method converges to the global solution with mathematical certainty and arbitrary accuracy, and the only input information it requires from the user are a set of point correspondences and a search interval. The cost function is based on the Huang-Faugeras constraint of the essential matrix. A recently proposed interval extension based on Bernstein polynomial forms has been investigated to speed up the search for the solution. Finally, experimental results are presented.

Index Terms

Image Processing and Computer Vision, Camera calibration, Modeling from video, Interval arithmetic, 3D/stereo scene analysis, Self-calibration.

I. Introduction

One of the goals of Computer Vision is to compute properties (mainly geometric) of the three-dimensional world from images. A challenging problem is to *reconstruct* a three-dimensional model of the scene from a moving camera. Most of the earlier studies in the field assume that the intrinsic parameters of the camera (focal length, image center and aspect ratio) are known. Computing camera motion in this case is a well known problem for which several methods are available (see [1] for a review). Given all the parameters of the camera, reconstruction is straightforward.

However, there are situations where the intrinsic parameters are unknown and the camera is not accessible (e.g. when using stock footage). In these cases the only information one can exploit are contained in the video sequence itself.

The classical approach to *autocalibration* (or *self-calibration*), in the case of a single moving camera with constant but unknown intrinsic parameters, is based on the Kruppa equations [2], which have been found to be very sensitive to noise [3], possibly due to the instability in the computation of the epipole [4]. Indeed, formulations which avoid the epipole seems to be more stable [5], [4].

Other methods [6], [7], [8], based on the *stratification* approach, upgrade a projective reconstruction to an Euclidean one without solving explicitly for the intrinsic parameters (see [9] for a review). The constant intrinsic parameters hypothesis has been relaxed in [10], [11], by assuming that some other parameters are known.

Recently, Mendonça and Cipolla [12] presented an algorithm which directly recovers the intrinsic parameters from fundamental matrices, like the Kruppa equations, but it is simpler and copes with varying parameters.

Under the assumption that only the (varying) focal length is unknown, closed form and linear solutions can be obtained [13], [14], [5], [15]. In all the other cases the parameters come from the solution of a system of polynomial equations or from the minimization of a non-linear function. In principle, continuation (homotopy) techniques could be applied to the former case, though—in practice—iterative minimization techniques must be used [3], as homotopy algorithms are applicable only in the case of few displacements, and can give rise to bifurcation phenomena. When minimizing a non-linear function by gradient descent methods, convergence to the global minimum is not guaranteed: it depends on the initialization—for deterministic algorithms,—or it is guaranteed only in probability—for stochastic algorithms [16]. Quasi-linear approaches reduce the sensitivity to the initial guess [17], [8], [18], but they do not solve the problem. The solutions of a simpler problem (only focal lenght is unknown) have been used to initialize the minimization in [11], [19]. In [20], a stratified approach has been proposed, based on the direct evaluation of a dense sampling of the search space. Albeit some of these techniques are effective, none of the existing methods is provably convergent.

In this paper we introduce a method for autocalibration that is *guaranteed* to converge to the global minimum, regardless of the starting point. In the same spirit of [12], [4], [16], we compute directly the intrinsic parameters from fundamental matrices. We assume constant intrinsic parameters, but the technique is flexible and can be adapted to varying parameters as well.

The minimization algorithm is based on Interval Analysis (IA) [21], a branch of numerical analysis that has received increasing attention during the last decade and has been strangely overlooked by the computer vision community.

Classical numerical optimization methods for the multidimensional case start from some approximate trial points and sample the objective function at only a finite number of points.

There is no way to guarantee that the function does not have some unexpectedly small values between these trial points, without making specific assumptions. On the contrary, IA optimization algorithms [22] can be seen as if they could evaluate the objective function over a continuum of points, including those points that are not finitely representable on the computer. They solve the optimization problem with *automatic result verification*, i.e. with the guarantee that the global minimizers have been found.

The rest of the paper is structured as follows. The next section introduces notation and some background notions of Computer Vision. The autocalibration problem that we address is formulated in Sec. III. In Sec. IV the reader is first introduced to Interval Analysis, and the specific optimization algorithm is described. Results are reported in Sec. V, and conclusions are drawn in Sec. VI.

II. Background

Throughout this paper we will use the general projective camera model [23]. Let $w = [x, y, z, 1]^\mathsf{T}$ be the homogeneous coordinates of a 3D point in the world reference frame. The homogeneous coordinates of the projected point are given by[1]

$$m \simeq P w, \tag{1}$$

where $P \triangleq A\,[R|t]$ is the camera matrix, whose position and orientation are represented, respectively, by the translation vector t and the 3×3 rotation matrix R. The matrix A contains the *intrinsic parameters*, and has the following form:

$$A = \begin{bmatrix} \alpha_u & \gamma & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2}$$

where α_u, α_v are the *focal lengths* in horizontal and vertical pixels, respectively, (u_0, v_0) are the coordinates of the *principal point*, given by the intersection of the optical axis with the retinal plane, and γ is the *skew* factor, that models non-rectangular pixels.

Two conjugate points m and m' are related by the *fundamental matrix* F [24]:

$$m'^\mathsf{T} F m = 0 \tag{3}$$

[1] $\simeq$ denotes equality up to a scale factor.

The rank of F is two and, being defined up to a scale factor, it depends upon seven parameters. Its computation requires a minimum of eight conjugate points to obtain a unique solution [24]. F depends on the intrinsic and extrinsic parameters according[2] to

$$F \simeq A'^{-\mathsf{T}}([t]_\times R)A^{-1}. \tag{4}$$

When conjugate points are in normalized coordinates ($A^{-1}m$), i.e., intrinsic parameters are known, one obtains the *essential matrix*:

$$E \simeq [t]_\times R. \tag{5}$$

The essential matrix encodes the rigid transformation between the two cameras, and it depends upon five independent parameters: three for the rotation and two for the translation up to a scale factor.

III. Problem Formulation

In many practical cases, the intrinsic parameters are unknown and point correspondences are the only information that can be extracted from a sequence of images. *Autocalibration* consists in computing the intrinsic parameters, or—in general—recovering the Euclidean *stratum*, starting from point correspondences. In this section we will see which constraints are available for autocalibration.

As we saw in Sec. II, the epipolar geometry of two views is described by the fundamental matrix, which depends on seven parameters. Since the five parameters of the essential matrix are needed to describe the rigid displacement, two independent constraints are available for the computation of the intrinsic parameters from the fundamental matrix. Indeed, the essential matrix is characterized by the following Theorem [25], [13]:

Theorem 1: A real 3×3 matrix E can be factored as the product of a non-zero skew-symmetric matrix and a rotation matrix if and only if E has two identical singular values and one zero singular value.

[2] $[t]_\times = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}.$

By exploiting this constraint, Hartley [13] derived two quadratic equations in the two values of the focal length. He also pointed out that no more information could be extracted from the fundamental matrix without making additional assumptions (e.g. constant intrinsic parameters).

It can be shown (see Sec. III-A) that the conditions on the singular values are equivalent to:

$$\det(E) = 0 \quad \wedge \quad 2\,\mathrm{tr}((EE^{\mathsf{T}})^2) - (\mathrm{tr}(EE^{\mathsf{T}}))^2 = 0, \tag{6}$$

which in turn is equivalent to the Kruppa equations [3]. The second clause of (6) can be decomposed [3] in two independent polynomial constraints.

All these constraints are algebraic interpretations of the so-called *rigidity constraint*, namely the fact that for any fundamental matrix F there exist two intrinsic parameters matrix A and A' and a rigid motion represented by t and R such that (4) is satisfied.

The autocalibration method by Mendonça and Cipolla is based on Theorem 1. They designed a cost function which takes the intrinsic parameters as arguments, and the fundamental matrices as parameters, and returns a positive value proportional to the difference between the two non-zero singular value of the essential matrix. Let F_{ij} be the fundamental matrix relating views i and j (computed from point correspondences), and let A_i and A_j be the respective (unknown) intrinsic parameter matrices. The cost function is

$$\chi(A_i,\ i = 1 \dots n) \triangleq \sum_{i=1}^{n} \sum_{j>i}^{n} w_{ij} \frac{{}^1\sigma_{ij} - {}^2\sigma_{ij}}{{}^1\sigma_{ij} + {}^2\sigma_{ij}}, \tag{7}$$

where ${}^1\sigma_{ij} \geq {}^2\sigma_{ij}$ are the non zero singular values of

$$E_{ij} = A_i^{\mathsf{T}} F_{ij} A_j, \tag{8}$$

and w_{ij} are normalized weight factors. In the general case of n views, the $n(n-1)/2$ fundamental matrices are not independent, neither are the $n(n-1)/2$ constraints that can be derived from them [26]. It can be shown [11] that, if n_k parameters are known and n_c parameters are constant, the unknown intrinsic parameters can be computed provided that

$$n(n_k + n_c) \geq 8 + n_c. \tag{9}$$

For example, if the intrinsic parameters are constant, three views are sufficient to recover them. If the skew is zero and the other parameters are varying, at least eight views are needed.

A. *The Huang-Faugeras cost function*

The use of (7) as an optimization criterion has been considered, however bounding the ranges of the singular values of an interval matrix is not trivial, since it requires the solution of a min-max optimization problem. Therefore, in the same spirit of the Mendonça-Cipolla algorithm, we minimize the following cost function, based on the Huang-Faugeras constraint, given by (6):

$$\chi(A_i, i=1,\dots,n) \triangleq \sum_{i=1}^{n}\sum_{j=i+1}^{n} w_{ij} \frac{2\,\mathrm{tr}(E_{ij}{E_{ij}}^{\mathsf{T}})^2 - \mathrm{tr}^2(E_{ij}{E_{ij}}^{\mathsf{T}})}{\mathrm{tr}^2(E_{ij}{E_{ij}}^{\mathsf{T}})}. \tag{10}$$

It is easy to see that

$$\mathrm{tr}(EE^{\mathsf{T}})^2 = \sum_{k=1}^{3} \sigma_k^4(E). \tag{11}$$

Hence, the second clause of (6) can be rewritten as

$$\begin{aligned} 2\,\mathrm{tr}(EE^{\mathsf{T}})^2 - \mathrm{tr}^2(EE^{\mathsf{T}}) &= \\ 2(\sigma_1^4+\sigma_2^4+\sigma_3^4) - (\sigma_1^2+\sigma_2^2+\sigma_3^2)^2 &= \\ (\sigma_1^2-\sigma_2^2)^2 + \sigma_3^2(\sigma_3^2 - 2(\sigma_1^2+\sigma_2^2)). & \end{aligned} \tag{12}$$

Therefore, provided that $\sigma_3 = 0$, each term of the cost function expressed by (10) vanishes for $\sigma_1^2 = \sigma_2^2$, as does the corresponding term of the Mendonça-Cipolla function (7). Moreover, as the terms are always positive, we do not need to take their square, as it would be required in a generic least squares problem, thereby reducing the order of the numerator and the denominator of the cost function from sixteen to eight.

If the essential matrix E is derived from F via (8), then $\det(F) = 0$ implies $\sigma_3 = 0$.

IV. Interval Analysis

Interval Arithmetic [27] is an arithmetic defined on intervals, rather than on real numbers. In the beginning, Interval Arithmetic was mainly employed for bounding the measurement errors of physical quantities for which no statistical distribution was known. Later on it was leveraged to a broad new field of applied mathematics, aptly named Interval Analysis, where rigorous proofs are the consequence of numerical computations.

A. *Notation and useful results*

In the sequel of this section we shall follow the notation used in [28], where intervals are denoted by boldface, scalar quantities are denoted by lower case letters and vectors and matrices are denoted by upper case. Brackets "$[\cdot]$" will delimit intervals, while parentheses "$(\cdot)$" will delimit vectors and matrices. Underscores and overscores will represent respectively lower and upper bounds of intervals. An interval $\boldsymbol{x}$ is called *degenerate* when $\underline{\boldsymbol{x}} = \overline{\boldsymbol{x}} = x$. $\mathbb{IR}$ and $\mathbb{IR}^n$ stand respectively for the set of real intervals and the set of real interval vectors of dimension n. The midpoint of an interval $\boldsymbol{x}$ is denoted by $\mathrm{m}(\boldsymbol{x})$, and the vector whose entries are midpoints of the entries of $\boldsymbol{X} \in \mathbb{IR}^n$ is denoted by $\mathrm{m}(\boldsymbol{X})$. The *width* of $\boldsymbol{x}$ is defined as $\mathrm{w}(\boldsymbol{x}) = \overline{\boldsymbol{x}} - \underline{\boldsymbol{x}}$. If $\boldsymbol{X} \in \mathbb{IR}^n$ then $\mathrm{w}(\boldsymbol{X}) = \max\{\mathrm{w}(\boldsymbol{x}_i),\ i = 1, \ldots, n\}$. If $f(x)$ is a function defined over an interval $\boldsymbol{x}$ then $\boldsymbol{f}^u(\boldsymbol{x})$ denotes the range of $f(x)$ over $\boldsymbol{x}$. Similarly, the range of $F : \mathbb{R}^n \to \mathbb{R}$ over $\boldsymbol{X}$ is denoted by $\boldsymbol{F}^u(\boldsymbol{X})$.

Interval arithmetic is an arithmetic defined on sets of intervals. If $\boldsymbol{x} = [\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}]$ and $\boldsymbol{y} = [\underline{\boldsymbol{y}}, \overline{\boldsymbol{y}}]$, a binary operation in the *ideal interval arithmetic* between $\boldsymbol{x}$ and $\boldsymbol{y}$ is defined as:

$$\boldsymbol{x} \,\mathrm{op}\, \boldsymbol{y} \triangleq \{x \,\mathrm{op}\, y \mid x \in \boldsymbol{x} \text{ and } y \in \boldsymbol{y}\},$$
$$\text{for op} \in \{+, -, \times, \div\}.$$

Thus, the ranges of the four elementary interval operations are exactly the ranges of the corresponding real operations. The operational definitions for the four elementary interval arithmetic operations are

$$\begin{aligned}
\boldsymbol{x} + \boldsymbol{y} &\triangleq \left[\underline{\boldsymbol{x}} + \underline{\boldsymbol{y}}, \overline{\boldsymbol{x}} + \overline{\boldsymbol{y}}\right], \\
\boldsymbol{x} - \boldsymbol{y} &\triangleq \left[\underline{\boldsymbol{x}} - \overline{\boldsymbol{y}}, \overline{\boldsymbol{x}} - \underline{\boldsymbol{y}}\right], \\
\boldsymbol{x} \times \boldsymbol{y} &\triangleq \Big[\min\left\{\underline{\boldsymbol{x}}\ \underline{\boldsymbol{y}}, \underline{\boldsymbol{x}}\ \overline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\ \underline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\ \overline{\boldsymbol{y}}\ \right\}, \\
&\qquad \max\left\{\underline{\boldsymbol{x}}\ \underline{\boldsymbol{y}}, \underline{\boldsymbol{x}}\ \overline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\ \underline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\ \overline{\boldsymbol{y}}\right\}\Big], \\
\frac{1}{\boldsymbol{x}} &\triangleq \begin{cases} [1/\overline{\boldsymbol{x}}, 1/\underline{\boldsymbol{x}}] & \text{if } \underline{\boldsymbol{x}} > 0 \\ [1/\underline{\boldsymbol{x}}, 1/\overline{\boldsymbol{x}}] & \text{if } \overline{\boldsymbol{x}} < 0 \end{cases} \quad (0 \notin [\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}]), \\
\boldsymbol{x} \div \boldsymbol{y} &\triangleq \boldsymbol{x} \times 1/\boldsymbol{y}.
\end{aligned}$$

The above definitions imply the ability to perform the four elementary operations with arbitrary precision. When implemented on a digital computer, however, truncation errors occur that may cause the resulting interval not to contain the result that would be obtained with ideal interval arithmetic. In order to avoid this effect, the lower endpoint of the interval must be rounded down to the nearest machine number less than the mathematically correct result, and the upper endpoint must be rounded up to the nearest machine number greater than the mathematically correct result. This mode of operation, called *direct rounding*, is available on any machine supporting the IEEE floating point standard.

Our use of IA is motivated by the need to obtain bounds on the range of mathematical functions.

Definition 1 (Interval extension): A function $\boldsymbol{F} : \mathbb{IR}^n \to \mathbb{IR}$ is said to be an *interval extension* of $F : \mathbb{R}^n \to \mathbb{R}$ provided

$$\boldsymbol{F}^u(\boldsymbol{X}) \subseteq \boldsymbol{F}(\boldsymbol{X})$$

for all intervals $\boldsymbol{X} \subset \mathbb{IR}^n$ within the domain of $\boldsymbol{F}$ [28].

The *natural* interval extension of a function is obtained by replacing variables with intervals and executing all operations according to the rule above. For instance, $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{x}(\boldsymbol{x} - 1)$ is an interval extensions of $f(x) = x^2 - x = x(x-1)$. By setting $\boldsymbol{x} = [0, 1]$ we have

$$\boldsymbol{f}(\boldsymbol{x}) = [0,1]\,([0,1] - 1) = [0,1]\,[-1,0] = [-1,0]\,,$$

which necessarily includes the exact range $\boldsymbol{f}^u([0,1]) = [-1/4, 0]$.

However, the bounds provided by natural interval extensions are usually too wide or pessimistic to be of value. The following definition characterizes how sharply interval extensions enclose the range of a function.

Definition 2 (Order α inclusion function): Let $\boldsymbol{F}(\boldsymbol{X})$ be an interval extension of $F : \mathbb{R}^n \to \mathbb{R}$ evaluated over an interval $\boldsymbol{X}$. We say that $\boldsymbol{F}$ is an *order α inclusion function* for F if there is a constant K, independent of the interval $\boldsymbol{X}$, such that

$$\mathrm{w}(\boldsymbol{F}(\boldsymbol{X})) - \mathrm{w}(\boldsymbol{F}^u(\boldsymbol{X})) \leq K \mathrm{w}(\boldsymbol{X})^\alpha \tag{13}$$

for all intervals $\boldsymbol{X}$ with $\mathrm{w}(\boldsymbol{X})$ sufficiently small.

It can be shown [28] that natural interval extensions are first order. Higher-order inclusion functions are key to the design of efficient global optimization algorithms, as we shall see in the next section.

A powerful method in Interval Analysis is *Interval Newton method*, that combines the classical Newton method and interval analysis. The result is an iterative method that can be used both to refine enclosures to solutions of nonlinear systems of equation, and to prove existence and uniqueness of such solutions, including tight and rigorous bounds on critical points of constrained optimization problems. Suppose now that $F : \mathbb{R}^n \to \mathbb{R}^n$, $\boldsymbol{X} \in \mathbb{IR}^n$, and $\check{X} \in \boldsymbol{X}$. Then a general form for the (multivariate) interval Newton operator is

$$\boldsymbol{N}(F; \boldsymbol{X}, \check{X}) \triangleq \check{X} + \boldsymbol{V}, \tag{14}$$

where $\boldsymbol{V}$ is the solution to the interval system[3] $\boldsymbol{J}(\boldsymbol{X})V = -F(\check{X})$ and $\boldsymbol{J}(\boldsymbol{X})$ is an interval extension of the Jacobian matrix of F over $\boldsymbol{X}$. Under certain natural smoothness conditions, interesting results can be shown [28]:

- $\boldsymbol{N}(F; \boldsymbol{X}, \check{X})$ must contain all points $X^* \in \boldsymbol{X}$ such that $F(X^*) = 0$. Consequently, if $\boldsymbol{N}(F; \boldsymbol{X}, \check{X}) \cap \boldsymbol{X} = \emptyset$, then there are no solutions of $F(X) = 0$ in $\boldsymbol{X}$.
- If $\boldsymbol{X}$ contains a solution of $F(X) = 0$ and $\mathrm{w}(\boldsymbol{X})$ is sufficiently small, the width of $\boldsymbol{N}(F; \boldsymbol{X}, \check{X})$ is roughly proportional to $\mathrm{w}(\boldsymbol{X})^2$.

The Newton method consists of the the following iteration

$$\boldsymbol{X}^{(k+1)} = \boldsymbol{N}(F; \boldsymbol{X}^{(k)}, \check{X}) \cap \boldsymbol{X}^{(k)}. \tag{15}$$

The interval $\boldsymbol{V}$ in (14) can be computed using the interval Gauss-Seidel method [28], which is the interval adaptation of the classical numerical method. If $0 \in \boldsymbol{J}(\boldsymbol{X})$ the quotients must be computed using the rule of *extended interval division* defined in [22]. The outcome will be, in general, a union of disjoint intervals. For details and further references, see [28].

B. IA based Global Optimization

The ability of Interval Analysis to compute bounds to the range of functions has been most successful in global optimization. The overall structure of the Moore-Skelboe or Hansen [22] branch-and-bound algorithm is:

1) store in a list $\mathcal{L}$ the initial interval $\boldsymbol{X}_0 \in \mathbb{IR}^n$ containing the sought minima;
2) pick an interval $\boldsymbol{X}$ from $\mathcal{L}$;

[3]The *solution set* of the interval linear system $\boldsymbol{A}X = \boldsymbol{B}$, is defined as the set $\{X : \exists A \in \boldsymbol{A} \text{ and } \exists B \in \boldsymbol{B} \text{ t.c. } AX = B\}$. Numerical methods provides an hyperrectangle containing the solution set.

3) if $\boldsymbol{X}$ is guaranteed not to contain a global minimizer, then discard it, otherwise subdivide $\boldsymbol{X}$ and store the sub-intervals in $\mathcal{L}$;
4) repeat from step 2) until the width of the intervals in $\mathcal{L}$ are below the desired accuracy.

The criteria used to delete intervals are based on rigorous bounds, therefore the interval containing the global minimizer is never deleted even in the presence of rounding errors.

We employed an algorithm inspired by a recently proposed global optimization method [29], based on the Moore-Skelboe-Hansen branch-and-bound algorithm and Bernstein polynomials for bounding the range of the objective function.

A combination of several test have been used in our implementation.

The *cut-off* test uses an upper bound $\hat{F}$ of the global minimum of the objective function F to discards an interval $\boldsymbol{X}$ from $\mathcal{L}$ if $\boldsymbol{F}(\boldsymbol{X}) > \hat{F}$. Any value taken by F is an upper bound for its global minimum, but the tighter is the bound, the more effective is the cut-off test. In Section IV-B.1 we describe the method that we used to determine and update $\hat{F}$.

The *monotonicity* test determines whether the function F has no stationary points in an entire sub-interval $\boldsymbol{X}$. Denote the interval extension of the gradient of F over $\boldsymbol{X}$ by $\boldsymbol{\nabla F}(\boldsymbol{X})$. If $0 \notin \boldsymbol{\nabla F}(\boldsymbol{X})$ then $\boldsymbol{X}$ can be deleted.

The *concavity* test examines the concavity of F, using its Hessian matrix H. Let $\boldsymbol{H}_{i,i}(\boldsymbol{X})$ denote the interval extension of the $i-$th diagonal entry of Hessian over $\boldsymbol{X}$. An interval can be deleted if $\overline{\boldsymbol{H}}_{i,i}(\boldsymbol{X}) < 0$ for some i.

The *Interval Newton step* applies one step of the interval Newton method (14) to the non-linear system $\nabla F(X) = 0$, $X \in \boldsymbol{X}$. As a consequence we may validate that $\boldsymbol{X}$ contains no stationary points, in which case we discard $\boldsymbol{X}$, otherwise we may contract or subdivide $\boldsymbol{X}$.

The complete optimization scheme can be summarized as the following pseudocode:

GLOBAL-OPTIMIZATION ALGORITHM

```
$\mathcal{U} \leftarrow \emptyset$
$\mathcal{L} \leftarrow \{\boldsymbol{X}_0\}$ list of intervals sorted in order of increasing $\underline{\boldsymbol{F}}(\boldsymbol{X})$
**while** $\mathcal{L} \neq \emptyset$ **do**
    remove the first interval $\boldsymbol{X}$ from $\mathcal{L}$
    **if** stop criterion **then** $\mathcal{U} \leftarrow \mathcal{U} \cup \{\boldsymbol{X}\}$
    **else if** (cut-off test: $\boldsymbol{F}(\boldsymbol{X}) > \hat{F}$   or
            monotonicity test: $0 \notin \boldsymbol{\nabla F}(\boldsymbol{X})$   or
            concavity test: $\overline{\boldsymbol{H}}_{i,i}(\boldsymbol{X}) < 0$ for some $i$)   **then** $\boldsymbol{Y} \leftarrow \emptyset$
        **else**  interval Newton step: $\boldsymbol{Y} \leftarrow \boldsymbol{X} \cap \boldsymbol{N}(\nabla F; \boldsymbol{X}, \mathrm{m}(\boldsymbol{X}))$
    bisect $\boldsymbol{Y}$ and insert the resulting intervals in $\mathcal{L}$
    update $\hat{F}$
**end**
**return** $\mathcal{U}$
```

A problem of global optimization algorithms based on IA is the so called *cluster effect*: as observed in [30], sub-intervals containing no solutions cannot be easily eliminated if there is a local minimum nearby. As a consequence of over-estimation in range bounding, many small intervals are created by repeated splitting, whose processing may dominate the total work spent on global search. This phenomenon occurs when the order of the inclusion function is less than three [30], hence we shall look for sharper inclusion functions.

1) Taylor-Bernstein forms: An interesting extension of IA that reduces the over-estimation is based on Taylor polynomials.

Definition 3 (Taylor Model): Let $F : \boldsymbol{X} \subset \mathbb{R}^n \to \mathbb{R}$ be a function that is $(m+1)$ times continuously partially differentiable. Let X_0 be a point in $\boldsymbol{X}$ and $P_{m,F}$ the m-th order Taylor polynomial of F around X_0. Let $\boldsymbol{I}_{m,F}$ be an interval such that

$$F(X) \in P_{m,F}(X - X_0) + \boldsymbol{I}_{m,F} \quad \forall X \in \boldsymbol{X}. \tag{16}$$

We call the pair $(P_{m,F}, \boldsymbol{I}_{m,F})$ an m-th order *Taylor model* of F [31] .
Hence $P_{m,F} + \boldsymbol{I}_{m,F}$ encloses F between two hypersurfaces on $\boldsymbol{X}$ (Fig. 1).

Taylor models of any computable function can be obtained recursively using the *Taylor Model Arithmetic* described in [31]. In order to bound the range of a function F over a domain $\boldsymbol{X}$,

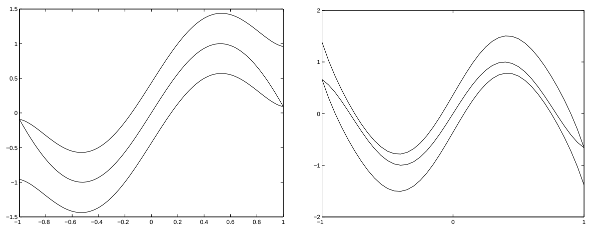

Fig. 1. Example of bounding a 7th order polynomial with a 3rd order Taylor model

it is sufficient to compute an interval extension $\boldsymbol{P}_{m,F}(\boldsymbol{X})$ for the polynomial $P_{m,F}$, since from Definition 3 (with $X_0 = 0$) it follows that

$$\boldsymbol{F}^u(\boldsymbol{X}) \subseteq \boldsymbol{P}_{m,F}(\boldsymbol{X}) + \boldsymbol{I}_{m,F}.$$

The sharpness of the bounds depends on the method used to obtain the inclusion function for $P_{m,F}$. More precisely, if $\boldsymbol{P}^u_{m,F}(\boldsymbol{X})$ denotes the exact range of $P_{m,F}$, then $\boldsymbol{P}^u_{m,F}(\boldsymbol{X}) + \boldsymbol{I}_{m,F}$ is an $m+1$ order inclusion function for F over $\boldsymbol{X}$, where m is the degree of the Taylor polynomial [29].

A *Taylor-Bernstein form* is a Taylor model where the polynomial is expressed in the Bernstein basis rather than in the canonical power basis. The advantage is that the Taylor-Bernstein form allows to compute the exact range of the polynomial part. Hence, with $m \geq 2$, the cluster effect is avoided. A Bernstein polynomial has the form (in one dimension):

$$p(x) = \sum_{i=0}^{m} a_i \begin{pmatrix} m \\ i \end{pmatrix} x^i (1-x)^{m-i}. \tag{17}$$

An important property of these polynomials is that $p(x)$ on $\boldsymbol{x}$ is a convex combination of a_i's, so that the coefficients of the Bernstein form provide lower and upper bounds to the range:

$$\boldsymbol{p}^u(\boldsymbol{x}) \subseteq [\min\{a_i\}, \max\{a_i\}].$$

If the polynomial is monotone over a domain $\boldsymbol{x}$ then the Bernstein form gives the exact range since the minimum and maximum occurs respectively at a_1 and a_m, $a_1 = p(\underline{\boldsymbol{x}})$ and $a_m = p(\overline{\boldsymbol{x}})$. This suggests that the exact range of a polynomial p on $\boldsymbol{x}$ can be obtained by transforming the polynomial into Bernstein form and then repeatedly subdividing it until the bounds of all sub-intervals are exact. The subdivision can be easily done with De Casteljau algorithm, well known in Computer Graphics [32]. Bernstein polynomials can be easily extended to the multivariate case, where analogous properties hold (see [29]).

The knowledge of the exact range of $P_{m,F}$ helps to make the cut-off test more effective. Indeed, if $\boldsymbol{P}^u_{m,F}(\boldsymbol{X})$ is the exact range, then $\underline{\boldsymbol{P}}^u_{m,F}(\boldsymbol{X})$ = $\min\{P_{m,F}\}$ and the minimum of F over $\boldsymbol{X}$ is contained in $\underline{\boldsymbol{P}}^u_{m,F}(\boldsymbol{X}) + \boldsymbol{I}_{m,F}$. Then $\underline{\boldsymbol{P}}^u_{m,F}(\boldsymbol{X}) + \overline{\boldsymbol{I}}_{m,F}$ is an upper bound of the minimum of F over $\boldsymbol{X}$. The cut-off value $\hat{F}$ is the smallest upper bound for all the intervals in the list.

The advantages and limits of Taylor models are widely discussed in [33], where the author also points out that the Taylor-Bernstein form is well suited to low dimension problems.

The Jacobian and Hessian matrices of the cost function are derived in closed form in [34].

V. Experimental results

A. Autocalibration

In our experiment we assume that the intrinsic parameters of the camera are constant. Fundamental matrices were computed using the linear 8-point algorithm with data normalization as described by Hartley in [35]. The weight w_{ij} has been defined as the residual of the estimation of F_{ij} [12]. We used Taylor models of degree four. As a stop criterion in the global optimization algorithm we required $\mathrm{w}(\boldsymbol{I}_{m,F}) \leq 10^{-10}$; using this value we typically get solution interval 2.5 pixels wide. Time figures refers to our implementation in MATLAB and C++, on a Pentium III 900 MHz processor.

In order to test our technique, we run a synthetic experiment in which data consisted of 50 points randomly scattered in a sphere of unit radius, centered at the origin. Views were generated by placing cameras at random positions, at a mean distance from the center of 2.5 units with a standard deviation of 0.25. The orientations of the cameras were chosen randomly with the constraint that the optical axis should point toward the center. The intrinsic parameters were given a known value: $\alpha_u = \alpha_v = 800, u_0 = v_0 = 256$ pixels. As customary it was assumed $\gamma = 0$.

The accuracy (see [34] for detailed results) is in agreement with the figures reported in [12], [16], as we basically use the same cost function.

In order to justify the use of our global optimization algorithm, we run a standard gradient method[4], initialized by randomly choosing a point in the domain $[300, 1700] \times [300, 1700] \times [156, 356] \times [156, 356]$. After performing 100 trials we recorded how many times the algorithm converged to the correct solution, which was assumed to be the one to which it converged when initialized with the true intrinsic parameters (within a 10% tolerance). The quasi-Newton method converged in the 86% of cases, with 5 views and 1.0 pixel noise. Average running time was 0.9 sec. Our algorithm spent 23.2 min. on the same problem, but convergence is 100% guaranteed.

We tested our autocalibration on the same real sequences used in other papers [16], [36], [4], [37]; all the sequences consists of five frames. The starting interval was chosen as follows: the midpoint for (u_0, v_0) is the image center and the width is 20% of the image size; the interval for the focal lengths is always $[300 \times 1700]$. Point correspondences were obtained manually. In all the real experiments the final interval width was around one pixel.

Table I compares our results with those previously published, when available. Please note that the values reported by other articles are the result of different autocalibration algorithms, and must not be taken as ground truth. Values in brackets were guessed, not computed.

TABLE I

MIDPOINTS OF INTRINSIC PARAMETERS COMPUTED WITH OUR ALGORITHM VERSUS PREVIOUS RESULTS.

	Our algorithm				Previous results			
Sequence	α_u	α_v	u_0	v_0	α_u	α_v	u_0	v_0
Valbonne [36]	619	699	234	372	681	679	259	383
ETL [37]	800	831	405	352	837	837	(378)	(252)
Nekt [4]	720	600	410	191	713	605	378	314

In order to make a more meaningful assessment we compared the results of our algorithm with those obtained by a standard calibration technique [38]. Table II reports the result for four sequences taken in Verona, each consisting of five frames[5]. Computation times are shown in Table III. It is interesting to note that in “ETL” and “Piazza Dante” the motion of the camera was close to a degenerate configuration, and indeed the computation time is significantly larger than the average.

[4]We used the quasi-Newton method implemented by the `fminunc` function in the MATLAB Optimization Toolbox.

[5]Test sequences can be found on the World Wide Web at
`http://www.sci.univr.it/~fusiello/demo/autocal`.

TABLE II

MIDPOINT OF INTRINSIC PARAMETERS COMPUTED WITH OUR ALGORITHM VERSUS CALIBRATION.

Sequence	α_u	α_v	u_0	v_0
Calibration	*1341*	*1343*	*521*	*382*
Castel Vecchio	1328	1319	582	328
S.Zeno	1359	1405	436	345
Piazza Erbe	1368	1289	450	402
Piazza Dante	1405	1358	460	410

TABLE III

COMPUTATION TIMES FOR REAL SEQUENCES.

Sequences	Time [min]
Valbonne	77
ETL	97
Nekt	65
S.Zeno	47
Castel Vecchio	20
Piazza Dante	90
Piazza Erbe	32

B. 3-D Reconstruction

Using the midpoint of intrinsic parameters computed by autocalibration, and the fundamental matrices, structure was recovered by first factorizing out the motion from the essential matrices [13], then recovering the projection matrices [36] and finally computing 3-D structure by triangulation [39]. As customary, results are refined by bundle adjustment, in order to obtain a maximum likelihood solution with respect to the underlying measures. More details can be found in [34].

As shown in Fig. 2, the projection of the reconstructed points coincides with the original image points.

In order to assess quantitatively the metric accuracy of the reconstruction, as the absolute dimensions of the objects are unknown, we computed the angles between 3D segments that are known to be parallel or orthogonal in the real scene (Table IV).

TABLE IV

ANGLES BETWEEN SEGMENTS OF THE RECONSTRUCTION SHOWN IN FIG. 2.

	Valbonne				Castel Vecchio				Piazza Erbe			
Segm	20-22	20-18	17-19	21-18	1-6	2-3	1-4	4-5	3-4	1-2	6-7	5-8
Computed	1.52°	0.82°	88.6°	88.4°	89.7°	88.9°	2.7°	89.8°	0.8°	89.3°	88.8°	0.8°
True	0°	0°	90°	90°	90°	90°	0°	90°	0°	90°	90°	0°

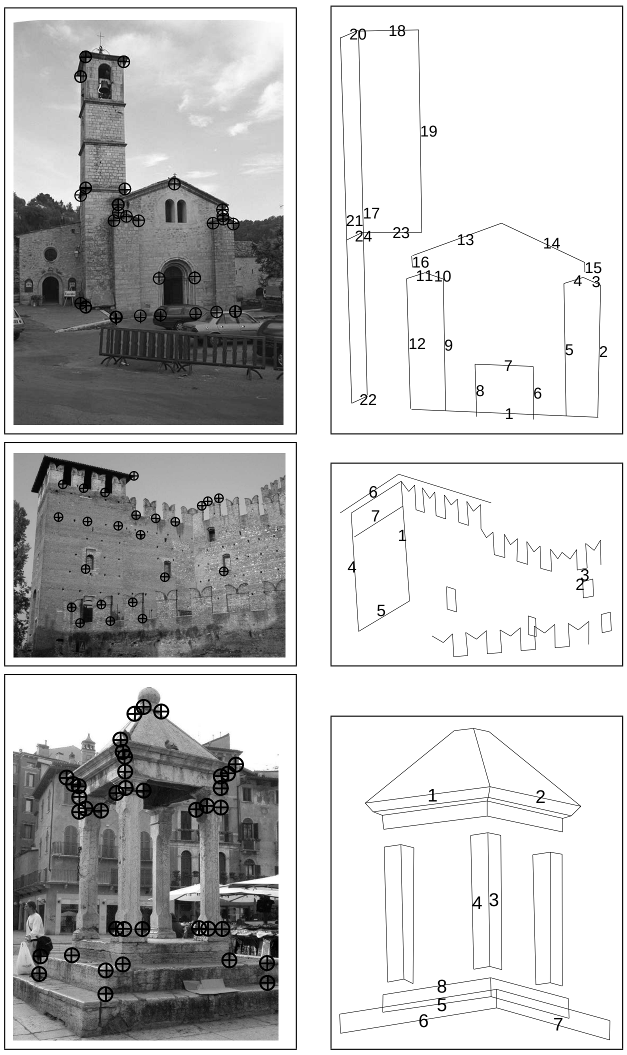

Fig. 2. Valbonne (top row), Castel Vecchio (middle row) and Piazza Erbe (bottom row). Left: the projection of reconstructed points (○ mark) are shown superimposed onto the original feature points (+ mark). Right: a view of the 3D reconstruction.

VI. Conclusions and future work

Global optimization based on Interval Analysis is a general method that in this paper has been applied to the autocalibration problem, obtaining a technique that is guaranteed to converge to the global solution with mathematical certainty and arbitrary accuracy. The choice of the initial interval is not critical for the successful termination of the algorithm – provided that it contains the global minimizer – because it only influences the computation time.

The experiments show that our method achieves results comparable to standard methods. The computation time confines this technique to off-line applications, but future work will aim at reducing it by introducing several variations to the present model.

We also plan to explore the use of IA tools to automatically detect degenerate configurations [40], [17].

Acknowledgments

The authors are grateful to R. P. Mendonça for introducing them to matrix differential calculus. A special thanks goes to Arnold Neumaier for many helpful discussion. G. Roth kindly provided the image sequences he used in his paper [16]. A. Whitwhead pointed out an error in the conference paper. This work was partially supported by the NSF CNSE at the California Institute of Technology (EEC-9402726). A preliminary version of this paper appeared on ICCV'03.

References

[1] T. S. Huang and A. N. Netravali, "Motion and structure from feature correspondences: A review," *Proceedings of IEEE*, vol. 82, no. 2, pp. 252–267, 1994.

[2] S. J. Maybank and O. Faugeras, "A theory of self-calibration of a moving camera," *International Journal of Computer Vision*, vol. 8, no. 2, pp. 123–151, 1992.

[3] Q.-T. Luong and O. Faugeras, "Self-calibration of a moving camera from point correspondences and fundamental matrices," *International Journal of Computer Vision*, vol. 22, no. 3, pp. 261–289, 1997.

[4] M. I. Lourakis and R. Deriche, "Camera self-calibration using the singular value decomposition of the fundamental matrix," in *Proc. of the 4th Asian Conference on Computer Vision*, vol. I, January 2000, pp. 403–408.

[5] R. I. Hartley, "Kruppa's equations derived from the fundamental matrix," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 133–135, February 1997.

[6] M. Pollefeys and L. Van Gool, "A stratified approach to metric self-calibration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Puerto Rico, 1997, pp. 407–412.

[7] A. Heyden and K. Åström, "Euclidean reconstruction from constant intrinsic parameters," in *Proceedings of the International Conference on Pattern Recognition*, Vienna, 1996, pp. 339–343.

[8] B. Triggs, “Autocalibration and the absolute quadric,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Puerto Rico, 1997, pp. 609–614.

[9] A. Fusiello, “Uncalibrated Euclidean reconstruction: A review,” *Image and Vision Computing*, vol. 18, no. 6-7, pp. 555–563, May 2000.

[10] A. Heyden and K. Åström, “Euclidean reconstruction from image sequences with varying and unknown focal length and principal point,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Puerto Rico, 1997, pp. 438–443.

[11] M. Pollefeys, R. Koch, and L. Van Gool, “Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters,” in *Proceedings of the IEEE International Conference on Computer Vision*, Bombay, 1998, pp. 90–95.

[12] P. Mendonça and R. Cipolla, “A simple techinique for self-calibration,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1999, pp. I:500–505.

[13] R. I. Hartley, “Estimation of relative camera position for uncalibrated cameras,” in *Proceedings of the European Conference on Computer Vision*, Santa Margherita L., 1992, pp. 579–587.

[14] P. Sturm, “On focal length calibration fro two views,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. II, Kauai, USA, 2001, pp. 145–150.

[15] S. Bougnoux, “From projective to Euclidean space under any practical situation, a criticism of self-calibration,” in *Proceedings of the IEEE International Conference on Computer Vision*, Bombay, 1998, pp. 790–796.

[16] G. Roth and A. Whitehead, “Some improvements on two autcalibration algorithms based on the fundamental matrix,” in *Proceedings of the International Conference on Pattern Recognition*, vol. 2, Quebec City, Alberta, 2002, pp. 312–315.

[17] J. Ponce, “On computing metric upgrades of projective reconstructions under the rectangular pixel assumption,” in *Proc. of the SMILE 2000 Workshop on 3D Structure from Multiple Images of Large-Scale Environments,*, ser. LNCS, no. 2018. Springer-Verlag, 2000, pp. 52–67.

[18] J. Oliensis, “Fast and accurate self-calibration,” in *Proceedings of the IEEE International Conference on Computer Vision*, 1999.

[19] A. Heyden and K. Åström, “Minimal conditions on intrinsic parameters for Euclidean reconstruction,” in *Proceedings of the Asian Conference on Computer Vision*, Hong Kong, 1998.

[20] R. Hartley, E. Hayman, L. de Agapito, and I. Reid, “Camera calibration and the search for infinity,” in *Proceedings of the IEEE International Conference on Computer Vision*, 1999.

[21] A. Neumaier, *Introduction to Numerical Analysis*. Cambridge: Cambridge Univ. Press, 2001.

[22] E. R. Hansen, *Global Optimization Using Interval Analysis*. Marcel Dekker, New York, 1992.

[23] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[24] Q.-T. Luong and O. D. Faugeras, “The fundamental matrix: Theory, algorithms, and stability analysis,” *International Journal of Computer Vision*, vol. 17, pp. 43–75, 1996.

[25] T. Huang and O. Faugeras, “Some properties of the E matrix in two-view motion estimation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 12, pp. 1310–1312, December 1989.

[26] Q.-T. Luong and T. Viéville, “Canonical representations for the geometries of multiple projective views,” *Computer Vision and Image Understanding*, vol. 64, no. 2, pp. 193–229, 1996.

[27] R. E. Moore, *Interval Analysis*. Prentice-Hall, 1966.

[28] R. B. Kearfott, *Rigorous Global Search: Continuous Problems*. Kluwer, 1996.

[29] P.S.V.Nataraj and K.Kotecha, "An algorithm for global optimization using the taylor-bernstein form as an inclusion function," *International Journal of Global Optimization*, vol. 24, pp. 417–436, 2002.

[30] B. Kearfott and Du, "The cluster problem in multivariate global optimization," *Journal of Global Optimization*, vol. 5, pp. 253–365, 1994.

[31] K. Makino and M. Berz, "Taylor models and other validated functional inclusion methods," *International Journal of Pure and Applied Mathematics*, vol. 4, no. 4, pp. 379–456, 2003.

[32] D. Rogers and J.A.Adams, *Mathematical Elements for Computer Graphics*, 2nd ed. McGraw-Hill, 1990.

[33] A. Neumaier, "Taylor forms - use and limits," *Reliable Computing*, vol. 9, pp. 43 – 79, 2002.

[34] A. Fusiello, A. Benedetti, M. Farenzena, and A. Busti, "Globally convergent autocalibration using interval analysis," Dipartimento di Informatica, Università di Verona, Tech. Rep. RR 09/2003, 2003.

[35] R. I. Hartley, "In defence of the 8-point algorithm," in *Proceedings of the IEEE International Conference on Computer Vision*, 1995.

[36] C. Zeller and O. Faugeras, "Camera self-calibration from video sequences: the Kruppa equations revisited," INRIA, Research Report 2793, February 1996.

[37] T. Ueshiba and F. Tomita, "A factorization method for projective and Euclidean reconstruction from multiple perspective views via iterative depth estimation," in *Proceedings of the European Conference on Computer Vision*, University of Freiburg, Germany, 1998, pp. 296–310.

[38] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[39] P. Beardsley, A. Zisserman, and D. Murray, "Sequential update of projective and affine structure from motion," *International Journal of Computer Vision*, vol. 23, no. 3, pp. 235–259, 1997.

[40] P. Sturm, "A case against Kruppa's equations for camera self-calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1199–1204, Sep 2000.