Vision, Video and Graphics (2005)
E. Trucco, M. Chantler (Editors)

Rigorous Computing in Computer Vision

Michela Farenzena and Andrea Fusiello

Dipartimento di Informatica, Università di Verona
Strada Le Grazie 15, 37134 Verona, Italy
farenzena@sci.univr.it, andrea.fusiello@sci.univr.it

Abstract
In this paper we discuss how Interval Analysis can be used to solve some problems in Computer Vision, namely autocalibration and triangulation. The crucial property of Interval Analysis is its ability to rigorously bound the range of a function over a given domain. This allows to propagate input errors with guaranteed results (used in multi-views triangulation) and to search for solution in non-linear minimisation problems with provably correct branch-and-bound algorithms (used in autocalibration). Experiments with real calibrated images illustrate the interval approach.

Categories and Subject Descriptors (according to ACM CCS): I.4.8 [Image Processing and Computer Vision]: Scene Analysis I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling G.4 [Mathematical Software]: Reliability and robustness

1. Introduction

Interval analysis (IA) [Moo66] is an approach to the solution of numerical problems by performing computations on sets of reals rather than on floating point approximations to reals. IA defines methods for computing an interval that encloses the range of various elementary mathematical functions. Interval evaluations return a superset of the mathematically correct result, hence the interval approach is said to be *rigorous*.

There are two principal advantages of IA over classical numerical analysis. The first is that the input errors and the roundoff errors are automatically incorporated into the result interval. Thus, interval evaluation can be viewed as automatically performing both a calculation and an error analysis. The second is that IA allows one to compute provably correct upper and lower bounds on the range of a function over an interval, and this proves useful in solving global optimisation problems. Another important application of IA is the construction of verifiable constraint solvers, which return intervals that are guaranteed to contain all the real solutions.

In this paper we report our experience in applying IA tools to Computer Vision problems. IA is not a panacea, of course, but has been strangely overlooked by the Computer Vision community. To the best of our knowledge, only [Bro83, OF87, MKP96] in the past approached IA, but they did not push this research forward. More attention has been given to IA in the Computer Graphics community, where it has been applied to problems such as ray tracing, approximating implicit curves [Sny92], and others.

On the basis of our preliminary results, however, we maintain that this is a very interesting and promising paradigm, which might challenge the probabilistic maximum-likelihood one in problems involving real data and provide guarantee of global convergence to non-linear optimisation algorithms.

We applied IA techniques to the problem of autocalibration [FBFB04b], whose solution comes from a non-linear minimisation, and to the problem of triangulation [FBFB04a], which requires that error in the localisation of image points is suitably taken into account. In this paper we will give an overview of IA and outline its application to these two problems.

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 [MF92], which have been found to be very sensitive to noise [LF97]. Other formulations (see [Fus00] for a review) avoid this instability, but all are based on a non-linear minimisation and none of the existing methods is provably convergent. On the contrary, IA algorithms [Han92] solve the optimisation problem with *automatic result verification*, i.e. with the guarantee that the global minimisers have been found.

In the absence of errors, *triangulation* is a trivial problem, involving only finding the intersection of rays in the space corresponding to back-projections of the image points. If data are perturbed, however, the rays do not intersect in a common point, and obtaining the best estimate of the 3-D point is not a trivial task. In literature, the custom procedure is to find the "best" 3-D point in some sense [HS97, Zha98]. Thanks to IA, instead of selecting one best solution, one can enclose the set of *all* the possible solutions, given a bounded error affecting the image points.

Adhering to the IA paradigm, we do not model a probability distribution inside the intervals, therefore there is no preferred solution in the solution set.

2. Interval Analysis

Interval Analysis [Moo66] is an arithmetic defined on intervals, rather than on real numbers. It was firstly introduced for bounding the measurement errors of physical quantities for which no statistical distribution was known. In the sequel of this section we shall follow the notation used in [KNN*], where intervals are denoted by boldface. Underscores and overscores will represent respectively lower and upper bounds of intervals. The midpoint of an interval $\mathbf{x}$ is denoted by $\operatorname{mid}(\mathbf{x})$. $\mathbb{IR}$ and $\mathbb{IR}^n$ stand respectively for the set of real intervals and the set of real interval vectors of dimension n. If $f(x)$ is a function defined over an interval $\mathbf{x}$ then $\operatorname{range}(f,\mathbf{x})$ denotes the range of $f(x)$ over $\mathbf{x}$.

If $\mathbf{x} = [\underline{x}, \overline{x}]$ and $\mathbf{y} = [\underline{y}, \overline{y}]$, a binary operation between $\mathbf{x}$ and $\mathbf{y}$ is defined in interval arithmetic as:

$$\mathbf{x} \circ \mathbf{y} = \{x \circ y \mid x \in \mathbf{x} \wedge y \in \mathbf{y}\}, \forall \circ \in \{+, -, \times, \div\}.$$

Operationally, interval operations are defined by the min-max formula:

$$\mathbf{x} \circ \mathbf{y} = \left[\min\left\{\underline{x}\circ\underline{y}, \underline{x}\circ\overline{y}, \overline{x}\circ\underline{y}, \overline{x}\circ\overline{y}\right\},\right. \\ \left.\max\left\{\underline{x}\circ\underline{y}, \underline{x}\circ\overline{y}, \overline{x}\circ\underline{y}, \overline{x}\circ\overline{y}\right\}\right] \quad (1)$$

Here, interval division $\mathbf{x}/\mathbf{y}$ is undefined when $0 \in \mathbf{y}$. However, under certain circumstances it make sense to define such quotients in a *extended arithmetic* [Kea96], where the division by zero is included, resulting the interval $[-\infty, \infty]$ in the worst case.

It is to note that the ranges of the four elementary interval operations are exactly the ranges of the corresponding real operations, and the above definitions imply the ability to perform them with arbitrary precision. When implemented on a digital computer, however, truncation errors occur, and they may cause the resulting interval not to contain the true result. In order to preserve the guarantee that the true value always lies within the interval, end-points must be rounded *outward*, i.e., the lower endpoint of the interval must be rounded down and the upper endpoint must be rounded up.

2.1. Inclusion functions

In general, for arbitrary functions, interval computation cannot produce the exact range, but only approximate it.

Definition 1 (Interval extension) [Kea96] A function $\mathbf{f} : \mathbb{IR} \to \mathbb{IR}$ is said to be an *interval extension* of $f : \mathbb{R} \to \mathbb{R}$ provided

$$\operatorname{range}(f, \mathbf{x}) \subseteq \mathbf{f}(\mathbf{x})$$

for all intervals $\mathbf{x} \subset \mathbb{IR}$ within the domain of $\mathbf{f}$.

Such a function is also called an *inclusion function*. So, given a function f and a domain $\mathbf{x}$, the inclusion function yields a rigorous bound (or enclosure) on the exact range $\operatorname{range}(f,\mathbf{x})$. This property is particularly suited for error propagation: If $\mathbf{x}$ bounds the input error on the variable x, $\mathbf{f}(\mathbf{x})$ bounds the output error. Therefore, if the exact value is contained in interval data, the exact value will be contained in the interval result. This approach is different from the established techniques for error propagation [Fau93, Har96, Kan93], mainly based on statistical analysis: a statistical distribution of the error need not to be assumed, and the result is mathematically guaranteed to contain the exact value.

Definition 2 (Natural interval extension) Let us consider a function f computable as an arithmetic expression f, composed of a finite sequence of operations applied to constants, argument variables or intermediate results. A *natural interval extension* of such a function, denoted by $\mathsf{f}(\mathbf{x})$, is obtained by replacing variables with intervals and executing all arithmetic operations according to the rules above.

Similar definitions apply for interval vectors (or boxes) in $\mathbb{IR}^n$. Some points are worth noting:

- Different expressions for the same function yield different natural interval extensions. For instance, $\mathsf{f}_\mathbf{1}(\mathbf{x}) = \mathbf{x}^2 - \mathbf{x}$, and $\mathsf{f}_\mathbf{2}(\mathbf{x}) = \mathbf{x}(\mathbf{x}-1)$ are both natural interval extensions of the same function.
- Variable dependency: Evaluating the expression $\mathsf{f}(x) = x - x$ with the interval [1,2], the result is $\mathsf{f}([1,2]) = [1,2] - [1,2] = [-1,1]$, not 0, as expected, because the piece of information that the two intervals represent the same variable is lost.
- Overestimation: Although the ranges of interval arithmetic operations are exact, this is not so if operations are composed. For example, if $\mathbf{x} = [0,1]$ we have $\mathsf{f}_\mathbf{2}(\mathbf{x}) = [0,1]([0,1]-1) = [0,1][-1,0] = [-1,0]$, which strictly includes $\operatorname{range}(f,[0,1]) = [-1/4,0]$. This effect arises as a consequence of the previous two.
- Wrapping effect: This is a phenomenon intrinsic to interval computation in $\mathbb{R}^n$, namely the fact that the image of

a box $\mathbf{x}$ under a map $F : \mathbb{R}^n \to \mathbb{R}^n$ is not a box, in general. Interval computation can yield, at best, the *interval hull* of range$(F, \mathbf{x})$, i.e. the smallest box containing range$(F, \mathbf{x})$ (see Fig. 1).

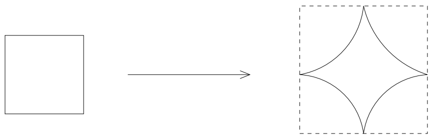

Figure 1: *The wrapping effect.*

The notion of *order* of an inclusion function characterises how sharply interval extensions enclose the range of a function: a higher order of inclusion means that the inclusion function gives sharper bounds. It can be shown [Kea96] that the natural interval extension is first order. Higher-order inclusion functions have been defined, for example the Taylor models (see [Neu02]):

Definition 3 (Taylor Model) Let $f : \mathbf{x} \subset \mathbb{R}^n \to \mathbb{R}$ be a function that is $(m+1)$ times continuously partially differentiable. Let x_0 be a point in $\mathbf{x}$ and $P_{m,f}$ the m-th order Taylor polynomial of f around x_0. Let $\mathbf{I}_{m,f}$ be an interval such that

$$f(x) \in P_{m,f}(x - x_0) + \mathbf{I}_{m,f} \quad \forall x \in \mathbf{x}. \qquad (2)$$

We call the pair $(P_{m,f}, \mathbf{I}_{m,f})$ an m-th order *Taylor model* of f [MB03] .

$P_{m,f} + \mathbf{I}_{m,f}$ encloses range$(f, \mathbf{x})$ between two hypersurfaces, as in Fig. 2.

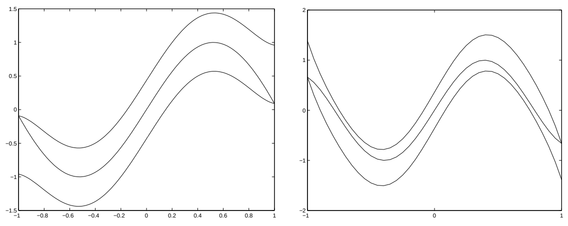

Figure 2: *Example of bounding a 7th order polynomial with a 3rd order Taylor model*

The sharpness of the bounds depends on the method used to obtain the inclusion function for $P_{m,f}$. 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 (see [NK02]). It can be shown that a Taylor-Bernstein form of degree m has order of inclusion $m+1$.

2.2. IA-based Optimisation

The ability of Interval Analysis to compute bounds to the range of functions has been most successful in global optimisation. The overall structure of the Moore-Skelboe or Hansen [Han92] branch-and-bound algorithm is:

1. store in a list $\mathcal{L}$ the initial interval $\mathbf{x}_0 \in \mathbb{IR}^n$ containing the sought minima;
2. pick an interval $\mathbf{x}$ from $\mathcal{L}$;
3. if $\mathbf{x}$ is guaranteed not to contain a global minimiser, then discard it, otherwise subdivide $\mathbf{x}$ and store the subintervals 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 minimiser is never deleted.

A problem of global optimisation algorithms based on this scheme is the so called *cluster effect* [KD94]: subintervals 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 [KD94], hence with Taylor-Bernstein form of degree ≥ 2 as inclusion functions the cluster effect is avoided.

We employed an algorithm inspired by a recently proposed global optimisation method [NK02], based on the Moore-Skelboe-Hansen branch-and-bound algorithm and Taylor-Bernstein forms for bounding the range of the objective function.

The complete optimisation scheme can be summarised as the pseudo-code reported in the next page. A combination of several tests has been used in our implementation:

1. The *cut-off* test uses an upper bound $\hat{f}$ of the global minimum of the objective function f to discard an interval $\mathbf{x}$ from $\mathcal{L}$ if $\mathbf{f}(\mathbf{x}) > \hat{f}$. 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.
2. The *monotonicity* test determines whether the function f has no stationary points in an entire sub-interval $\mathbf{x}$. Denote the interval extension of the gradient of f over $\mathbf{x}$ by $\nabla\mathbf{f}(\mathbf{x})$. If $0 \notin \nabla\mathbf{f}(\mathbf{x})$ then $\mathbf{x}$ can be deleted.
3. The *concavity* test examines the concavity of f, using its Hessian matrix H. Let $\mathbf{H}_{i,i}(\mathbf{x})$ denote the interval extension of the $i-$th diagonal entry of the Hessian over $\mathbf{x}$. An interval can be deleted if $\overline{H}_{i,i}(\mathbf{x}) < 0$ for some i.
4. The *Interval Newton step* applies one step of the interval Newton method [Kea96] to the non-linear system $\nabla f(x) = 0, x \in \mathbf{x}$. As a consequence we may validate that $\mathbf{x}$ contains no stationary points, in which case we discard $\mathbf{x}$, otherwise we may contract or subdivide $\mathbf{x}$.

3. Computer Vision background

Throughout this paper we will use the general projective camera model [HZ03]. Let $M = [x, y, z, 1]^{\mathsf{T}}$ be the homo-

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

geneous coordinates of a 3D point in the world reference frame. The homogeneous coordinates of the projected point m are given by

$$\kappa m = \mathrm{P}M \tag{3}$$

where κ is the depth of M wrt the camera, and $P = A[R|t]$ is the camera matrix, whose position and orientation are represented, respectively, by the translation vector t and the 3×3 rotation matrix R (*extrinsic parameters*). The matrix A contains the *intrinsic parameters*, and has the following form:

$$A = \begin{pmatrix} \alpha_u & \gamma & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{4}$$

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* [LF96]:

$$m'^{\top} F m = 0 \tag{5}$$

which is related to intrinsic and extrinsic parameters by

$$F \backsim A'^{-\top}([t]_{\times} R)A^{-1}. \tag{6}$$

where $\backsim$ 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.

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

$$E \backsim [t]_{\times} R. \tag{7}$$

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.

A counting argument implies that there must exist two linear independent constraints that characterise the essential matrix. Indeed, the essential matrix is characterised by the following Theorem [HF89, Har92]:

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.

4. Autocalibration: 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. In this hypothesis, called *weak calibration*, fundamental matrices can be obtained directly from conjugate points. *Autocalibration* consists in computing the intrinsic parameters, or –in general– recovering the Euclidean *stratum* from weak calibrated cameras.

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

$$\chi(A_i,\ i = 1 \ldots n) = \sum_{i=1}^{n} \sum_{j>i}^{n} w_{ij} \frac{{}^1\sigma_{ij} - {}^2\sigma_{ij}}{{}^1\sigma_{ij} + {}^2\sigma_{ij}}, \tag{8}$$

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

$$E_{ij} = A_i^{\top} F_{ij} A_j, \tag{9}$$

and w_{ij} are normalised weight factors.

4.1. The Huang-Faugeras cost function

The use of Eq. (8) as an optimisation 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 optimisation problem. Therefore, in the same spirit of the Mendonça-Cipolla algorithm, we minimise the following cost function,

$$\chi(A_i, i=1,\ldots,n) = \\ = \sum_{i=1}^{n}\sum_{j=i+1}^{n} w_{ij} \frac{2\,\mathrm{tr}(E_{ij}E_{ij}^{\top})^2 - \mathrm{tr}^2(E_{ij}E_{ij}^{\top})}{\mathrm{tr}^2(E_{ij}E_{ij}^{\top})}. \quad (10)$$

based on the Huang-Faugeras constraint:

$$\det(E)=0 \quad \wedge \quad 2\,\mathrm{tr}((EE^{\top})^2) - (\mathrm{tr}(EE^{\top}))^2 = 0. \quad (11)$$

which is equivalent to the constraint expressed by Theorem 1. Indeed, it is easy to see that

$$\mathrm{tr}(EE^{\top})^2 = \sum_{k=1}^{3} \sigma_k^4(E). \quad (12)$$

Hence, the second clause of (11) can be rewritten as

$$2\,\mathrm{tr}(EE^{\top})^2 - \mathrm{tr}^2(EE^{\top}) = \\ = (\sigma_1^2-\sigma_2^2)^2 + \sigma_3^2(\sigma_3^2 - 2(\sigma_1^2+\sigma_2^2)). \quad (13)$$

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 (8). 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.

The Jacobian and Hessian matrices of the cost function are derived in closed form in [FBFB03].

An enclosure **A** of the intrinsic parameters is obtained as the result of minimimizing (10) using the global optimization algorithm described in Sec. 2.2.

5. Triangulation: problem formulation

Let P_i, $i=1,\ldots,n$ be a sequence of n known cameras and m_i be the image of some unknown point M in 3-D space, both expressed in homogeneous coordinates. The problem of computing the point M given the camera matrices P_i and the image points m_i is known as the *triangulation* problem. In the absence of errors, this problem is trivial, involving only finding the intersection point of rays in the space. When data are perturbed by errors, however, the rays corresponding to back-projections of the image points do not intersect in a common point, therefore only an approximate solution can be defined. This approximation can be circumvented if one refrains from searching for *one* solution and compute instead a *set* of solutions that contains the error-free solution and can be defined precisely in terms of the error affecting the image points.

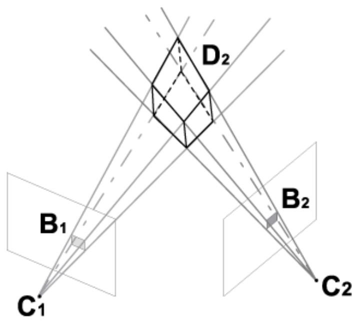

Figure 3: *Interval-based triangulation.*

In the case of two views, assuming that errors are bounded by rectangles B_1 and B_2 in the image, the *solution set* of triangulation is a polyhedron D_2 with a *diamond shape* as in Fig 3. Geometrically, D_2 is obtained by intersecting the two semi-infinite pyramids defined by the two rectangles B_1 and B_2 and the respective camera centres.

In the general case of n views, the solution set is defined as the polyhedron formed by the intersection of the n semi-infinite pyramids generated by the intervals $B_1, \ldots B_n$. Analytically, this region is defined as the set

$$D_n = \{M : \forall i = 1,\ldots,n\, \exists m_i \in B_i \text{ s.t. } m_i \simeq P_i M\}.$$

In the following we will show how the solution set can be enclosed with an axis-aligned box using Interval Analysis.

Given the camera matrices P_1 and P_2, let m_1 and m_2 be two corresponding points. It follows that m_2 lies on the epipolar line of m_1 and so the two rays back-projected from image points m_1 and m_2 lie in a common epipolar plane. As they lie in the same plane, they will intersect at some point. This point is the reconstructed 3-D scene point M.

The equation of the epipolar line can be derived from the equation describing the optical ray of m_1:

$$M = \begin{pmatrix} -P_{3\times3,1}^{-1} P_{\cdot\,4,1} \\ 1 \end{pmatrix} + \lambda \begin{pmatrix} P_{3\times3,1}^{-1} m_1 \\ 0 \end{pmatrix}, \quad \lambda \in \mathbb{R}, \quad (14)$$

where $P_{3\times3,1}$ is the matrix composed by the first three rows and first three columns of P_1, and $P_{\cdot\,4,1}$ is the fourth column of P_1. The epipolar line corresponding to m_1 represents the projection of the optical ray of m_1 onto the image plane 2:

$$\kappa m_2 = e_2 + \lambda m_1' \quad (15)$$

where

$$e_2 = P_2 \begin{pmatrix} -P_{3\times3,1}^{-1} P_{\cdot\,4,1} \\ 1 \end{pmatrix} \text{ and } m_1' = P_{3\times3,2} P_{3\times3,1}^{-1} m_1.$$

Analytically, the reconstructed 3-D point M can be found using Equation (15), by solving for parameters κ and λ, using the following closed form expressions:

$$\frac{1}{\kappa} = \frac{(m_2 \times m_1') \cdot (e_2 \times m_1')}{||e_2 \times m_1'||^2},$$
$$\frac{\lambda}{\kappa} = \frac{(m_2 \times e_2) \cdot (m_1' \times e_2)}{||m_1' \times e_2||^2}. \tag{16}$$

The coordinates of M are then obtained by inserting the value λ into Equation (15) (obviously, M can also be recovered with respect to the other camera, using the optical ray generated by m_2). After doing all the substitutions, we can write a closed form expression that relates the reconstructed point to the two conjugate image points:

$$M = \mathsf{f}(m_1, m_2) \tag{17}$$

We chose this formulation of the triangulation, introduced by [Fau92], precisely because it leads to this closed form expression, representing the geometric operation of intersecting rays in 3-D space. This will be a key feature for the application of Interval Analysis.

6. Interval-based triangulation

Let us consider the expression f defined in Eq. (17). If we let m_1 and m_2 vary in B_1 and B_2 respectively, then $\mathrm{range}(\mathsf{f}, B_1 \times B_2)$ describes the solution set D_2. Interval Analysis gives us a way to compute an axis-aligned bounding box containing D_2 by simply evaluating $\mathsf{f}(\mathbf{m}_1, \mathbf{m}_2)$, the natural interval extension of f, with $B_1 = \mathbf{m}_1$ and $B_2 = \mathbf{m}_2$.

IA guarantees that if the conjugate intervals $\mathbf{m}_1$ and $\mathbf{m}_2$ contain the exact point correspondences, then the interval result contains the exact (i.e. error-free) 3-D reconstructed point.

It may be worth noting that the result is not to be interpreted in a probabilistic or fuzzy way: no assumption is made on error statistical distribution, hence no point inside the resulting 3-D interval is more probable or more important than others.

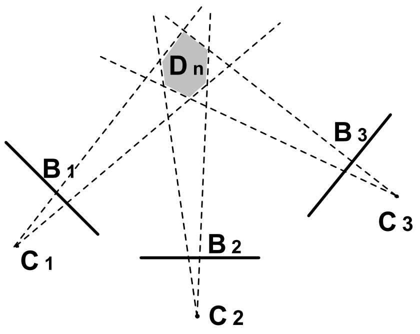

Figure 4: *Interval-based triangulation with n views.*

The approach is easily extendible to the general n-views case. As defined in Sec. 5, the solution set of triangulation is the 3-D polyhedron formed by the intersection of the semi-infinite pyramids generated by back-projecting in space the intervals $\mathbf{m}_1, \ldots, \mathbf{m}_n$ (Fig. 4). Thanks to the associativity of intersection, D_n can be obtained by first intersecting pairs of such pyramids and then intersecting the results. Let $D_2^{i,j}$ be the solution set of the triangulation between view i and view j. Then:

$$D_n = \bigcap_{\substack{i=1,\ldots,n \\ j=i+1,\ldots,n}} D_2^{i,j}. \tag{18}$$

An enclosure of the solution set D_n is obtained by intersecting the $n(n-1)/2$ enclosures of $D_2^{i,j}$ computed with the method described in Sec. 6. Since each enclosure contains the respective solution set $D_2^{i,j}$, their intersection will contain D_n. Similarly, as the error-free solution is contained in each $D_2^{i,j}$, then it must be contained in D_n as well.

7. Experimental results

Experimental validation of the algorithms described here and other results can be found in [FBFB04b] and on the Internet[†]. In this paper we only report one example of autocalibration and reconstruction.

We used the *Valbonne* sequence, consisting of five frames. The starting interval for the global minimisation is chosen as follows: the midpoint for $(\mathbf{u}_0, \mathbf{v}_0)$ is the image centre and the width is 20% of the image size; the interval for the focal lengths is $[300 \times 1700]$. The average width of the elements of the intrinsic parameters matrix obtained at the end of the minimisation is about one pixel. Table 1 compares our results with those published in [ZF96], obtained with a different autocalibration algorithm.

	α_u	α_v	u_0	v_0
Zeller & Faugeras	*681.3*	*679.3*	*258.8*	*383.2*
Our algorithm	618.5	699.2	234.1	372.8

Table 1: *Midpoint of intrinsic parameters computed with our algorithm versus the result found in [ZF96].*

Once intrinsic parameters are known, the motion can be factorised out from the essential matrices [Har92], and the projection matrices recovered as in [ZF96]. This part is executed using $\mathrm{mid}(\mathbf{A})$, but at the end of the process the interval nature of $\mathbf{A}$ is taken into account by computing normalised coordinates in interval arithmetic: $\mathbf{m} \leftarrow \mathbf{A}^{-1}\mathbf{m}$.

Normalised pointwise projection matrices $\mathrm{P}_i = [\mathrm{R} \mid t]$ are then used together with interval normalised coordinates to

[†] `http://www.sci.univr.it/~fusiello/demo`

reconstruct the Valbonne church (Figure 5) with our interval-based triangulation.

Given that image points are contained in 2-pixel wide intervals, the average side length of the 3-D boxes is about 50 cm. It is interesting to note that these boxes extend mainly along the z-axis.

8. Conclusions

In this paper we discussed how Interval Analysis can be used to solve some problems in Computer Vision, namely autocalibration and triangulation. Autocalibration consists in performing a non-linear minimisation, and triangulation requires that errors in the localisation of image points are suitably taken into account. IA allows to propagate input errors with guaranteed results and to obtain provably correct branch-and-bound algorithms.

On the basis of our preliminary results we maintain that IA is a very interesting and powerful paradigm, which might be applied to many other problems.

Work is in progress aimed at including geometrical constraints (e.g. known angles or lengths), which, together with the rigidity of the structure, will help to reduce further the width of the solution boxes.

Acknowledgements

Arrigo Benedetti co-authored some papers in the past and introduced the authors to IA.

References

[Bro83] BROOKS R.: Model-based three-dimensional interpretations of two-dimensional images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 5*, 2 (March 1983), 140–149.

[Fau92] FAUGERAS O.: What can be seen in three dimensions with an uncalibrated stereo rig? In *Proceedings of the European Conference on Computer Vision* (Santa Margherita L., 1992), pp. 563–578.

[Fau93] FAUGERAS O.: *Three-Dimensional Computer Vision: A Geometric Viewpoint*. The MIT Press, Cambridge, MA, 1993.

[FBFB03] FUSIELLO A., BENEDETTI A., FARENZENA M., BUSTI A.: *Globally Convergent Autocalibration using Interval Analysis*. Tech. rep., Dipartimento di Informatica, Università di Verona, 2003.

[FBFB04a] FARENZENA M., BUSTI A., FUSIELLO A., BENEDETTI A.: Rigorous accuracy bound for calibrated stereo reconstruction. In *Proceedings of the International Conference of Pattern Recognition* (Cambridge,UK, 2004), vol. IV, pp. 288–292.

[FBFB04b] FUSIELLO A., BENEDETTI A., FARENZENA M., BUSTI A.: Globally convergent autocalibration using interval analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 26*, 12 (December 2004), 1633–1638.

[Fus00] FUSIELLO A.: Uncalibrated Euclidean reconstruction: A review. *Image and Vision Computing 18*, 6-7 (May 2000), 555–563.

[Han92] HANSEN E. R.: *Global Optimization Using Interval Analysis.* Marcel Dekker, New York, 1992.

[Har92] HARTLEY R. I.: Estimation of relative camera position for uncalibrated cameras. In *Proceedings of the European Conference on Computer Vision* (Santa Margherita L., 1992), pp. 579–587.

[Har96] HARALICK R. M.: Propagating covariance in computer vision. In *Workshop on Performance Characteristics of Vision Algorithms* (Cambridge, UK, 1996), pp. 1–12.

[HF89] HUANG T. S., FAUGERAS O.: Some properties of the E matrix in two-view motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 11*, 12 (December 1989), 1310–1312.

[HS97] HARTLEY R. I., STURM P.: Triangulation. *Computer Vision and Image Understanding 68*, 2 (November 1997), 146–157.

[HZ03] HARTLEY R., ZISSERMAN A.: *Multiple view geometry in computer vision*, 2nd ed. Cambridge University Press, 2003.

[Kan93] KANATANI K.: *Geometric Computation for Machine Vision.* Oxford University Press, 1993.

[KD94] KEARFOTT B., DU: The cluster problem in multivariate global optimization. *Journal of Global Optimization 5* (1994), 253–365.

[Kea96] KEARFOTT B.: *Rigorous Global Search: Continuos Problems.* Kluwer, 1996.

[KNN*] KEARFOTT R. B., NAKAO M. T., NEUMAIER A., RUMP S. M., SHARY S. P., VAN HENTENRYCK P.: Standardized notation in interval analysis. Submitted to Reliable Computing.

[LF96] LUONG Q.-T., FAUGERAS O. D.: The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision 17* (1996), 43–75.

[LF97] LUONG Q.-T., FAUGERAS O.: Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of Computer Vision 22*, 3 (1997), 261–289.

[MB03] MAKINO K., BERZ M.: Taylor models and other validated functional inclusion methods. *International Journal of Pure and Applied Mathematics 4*, 4 (2003), 379–456.

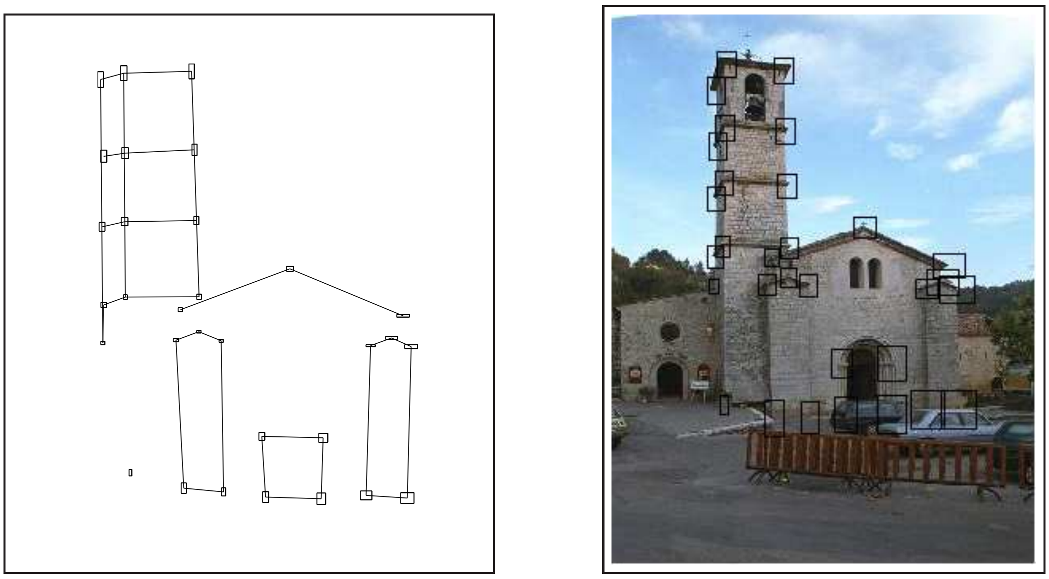

Figure 5: *Interval-based reconstruction of the Valbonne church (left). To better visualise the 3-D structure, segments joining the midpoints of the intervals have been drawn. On the right a frame of the sequence is shown with the projection of the 3-D intervals overlaid.*

[MF92] MAYBANK S. J., FAUGERAS O.: A theory of self-calibration of a moving camera. *International Journal of Computer Vision 8*, 2 (1992), 123–151.

[MKP96] MARIK R., KITTLER J., PETROU M.: Error sensitivity assessment ov vision algorithm based on direct error-propagation. In *Workshop on Performance Characteristics of Vision Algorithms* (Cambridge, UK, 1996).

[Moo66] MOORE R. E.: *Interval Analysis*. Prentice-Hall, 1966.

[Neu02] NEUMAIER A.: Taylor forms - use and limits. *Reliable Computing 9* (2002), 43 – 79.

[NK02] NATARAJ P. S. V., KOTECHA K.: An algorithm for global optimization using the taylor-bernstein form as an inclusion function. *International Journal of Global Optimization 24* (2002), 417–436.

[OF87] ORR M. J. L., FISHER R. B.: Geometric reasoning for computer vision. *Image Vision Computing 5*, 3 (1987), 233–238.

[Sny92] SNYDER J. M.: Interval analysis for computer graphics. In *Computer Graphics (SIGGRAPH '92 Proceedings)* (July 1992), vol. 26, pp. 121–130.

[ZF96] ZELLER C., FAUGERAS O.: *Camera Self-Calibration from Video Sequences: the Kruppa Equations Revisited*. Research Report 2793, INRIA, February 1996.

[Zha98] ZHANG Z.: Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision 27*, 2 (March/April 1998), 161–195.