Fitting Constrained 3D Models in Multiple Aerial Images

Bruno Vallet, Franck Taillandier
Laboratoire MATIS, Institut Géographique National
2,4 av Pasteur, 94165 Saint-Mandé, France
`firstname.lastname@m4x.org`

Abstract

This paper presents a method for fitting constrained models in multiple aerial images for building reconstruction applications. Compared to previous works, geometrical constraints are automatically inferred and enforced on the initial approximate model of building and the final constrained model is inherently compliant with the detected constraints through an implicit parameterization. Fitting models in images is performed through automatic matching between model edges and segments detected in the images. An iterative minimization enables to search for the *constrained* model that minimizes the distance with this image information. Results show that both the introduction of constraints on the model and the use of image information allow a significant gain in the precision of the reconstruction.

1 Introduction

1.1 Context

In the context of urban environments reconstruction from images, taking into account geometrical properties such as planarity, orthogonality, symmetry or horizontal lines is a necessary step to improve the quality of reconstruction [15], to ensure topological and geometrical coherence and to allow visual realism. The application described in this article is part of a global project [19] aiming at automatic building reconstruction from multiple *calibrated* aerial images (calibration will be assumed to be known in this article). Integration of constraints in this scheme has already shown advantages to select the best representation among a set of possible models of buildings. This article focuses on fitting the constrained reconstruction in images once topology is known by integrating constraints directly inferred on the model and performing matching between *model* edges and *image* segments.

1.2 Related Work

For man-made environments reconstruction applications, various approaches try to take benefit of geometrical constraints to fit 3D models in images. Some authors use model-based techniques where objects are decomposed into known base primitives grouped together to build more complex shapes [5, 17, 21, 8, 13, 14]. As models become more complex, constraints enforcing is difficult because of redundant constraints and over-constraints that are not handled in the geometrical definition of the model. More general approaches use constraints-based techniques [15, 16, 11, 4, 20, 1] but, due to the complexity of the problem, they all rely on user intervention for topological *and* geometrical description of the structures, which is very time-consuming and often redhibitory for important areas reconstruction.

In this article, we present an automated solution that avoids user intervention to fit an approximate 3D model in images with constraints enforcing. Whatever the strategy is, constraints such as planarity, orthogonality, parallelism, symmetry, horizontality of some edges can be inferred from the building, either *a priori* using a basis of models synthesizing the knowledge, or *a posteriori* by looking for approximative constraints verified by the sets of primitives. On the contrary to systems where constraints are embedded in the Least Square minimization process [20, 1, 15], we use an implicit and automatic modeling of the constrained scene before performing minimization as in [4, 11]. Therefore, we ensure that reconstruction is inherently compliant with the constraints. The method for matching a 3D model with image data is derived from the local search strategy described by [2]. More bibliographic references describing other strategies such as key features search, geometric hashing, or tree search transform can be found in this article. However local search strategy reveals to be well adapted to our context since images are already calibrated and an approximate position of the 3D model is known.

1.3 General Scheme

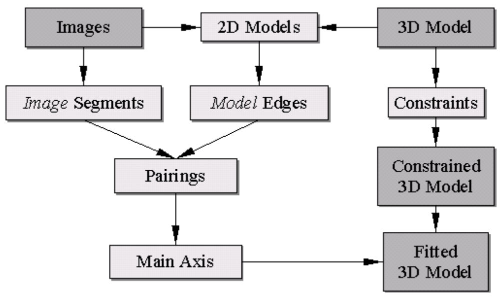
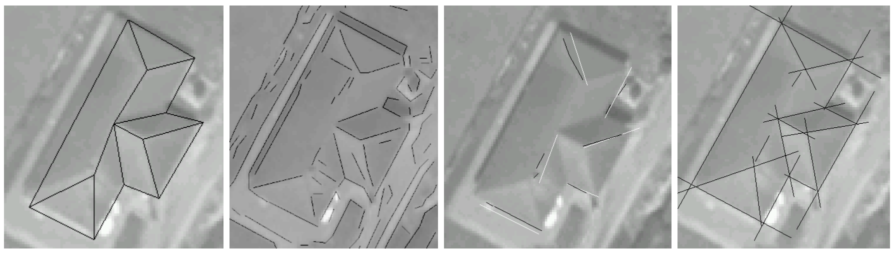


Figure 1: General scheme for the fitting of a constrained 3D model

The application presented is dedicated to automatic building reconstruction. Therefore we focus on constraints recurrent in these environment: parallelisms, orthogonality, horizontal edges, vertical symmetry. The scheme of the resolution of the problem is summarized in figure 1. First, edges of the 3D model are projected in each image. Then, segments are extracted from images. These *image* segments are matched with the projected edges (*model* edges). These matched segments are eventually used to compute a set of main axes. These steps are described in section 2.

At the same time, constraints inferring is performed through a procedure recalled in section 3 for readability and detailed in [18]. Enforcing constraints on the model is done through implicit parameterization as described in [11]. Finally, fitting this constrained model on the axes extracted from images is performed as described in section 4.

2 Model-Image Matching



(a) Model projection (b) Segments detection (c) Matching (d) Main axes

Figure 2: Image information extraction. For clarity, only a few matches are visualized in (c) (*model* edges are in white, matched *image* segments in black)

The edges M_j of the initial 3D model are projected in each image i, leading to the sets of 2D edges $\{M_j^i\}$ (figure 2.a). Conversely, using classical gradient filters and polygonization methods [6, 9], a set of segments $\{S_k^i\}$ are detected in each image (figure 2.b). For each projected edge M_j^i the goal is to find a set of matching segments $\mathscr{A}_j^i$ in $\{S_k^i\}$. This is achieved through a procedure derived from Beveridge's approach [3], which consists in minimizing a matching energy characteristic of the distance between edges and matched segments.

For each matching hypothesis between M_j^i and S_k^i, $\mathbf{m}_j^i$ (resp $\mathbf{s}_k^i$) denotes the vector represented by M_j^i (resp S_k^i), $\tilde{d}_j^i(\mathbf{x})$ defines the distance of a point $\mathbf{x}$ with regard to the line supporting M_j^i and $\mathscr{R}(M_j^i, S_k^i)$ defines the fraction of M_j^i overlapped by S_k^i when projected on M_j^i. For each *model* edge M_j^i, we define three energy terms:

$$E_G(S_k^i,\delta) = \int_{\mathbf{x}\in S_k^i} \left(\frac{\tilde{d}_j^i(\mathbf{x})}{\delta}\right)^2 d\mathbf{x}, \quad E_T(S_k^i,\alpha) = \frac{\tan(\mathbf{m}_j^i,\mathbf{s}_k^i)}{\tan(\alpha)}, \quad E_R(S_k^i) = \mathscr{R}(M_j^i,S_k^i) \quad (1)$$

The first term is related to the geometrical distance between M_j^i and S_k^i, the second term defines their direction discrepancy and the third term is related to the overlap of M_j^i by S_k^i. These energies are normalized to be 1 when the edges are at an average squared distance of δ for E_G, the edges directions form an angle of α for E_T, and M_j^i is totally overlapped by S_k^i for E_R. The best matching set $\mathscr{A}_j^i$ of S_k^i is then defined as the set of segments that minimizes the global energy:

$$E(\mathscr{A}_j^i,\delta,\alpha) = \sum_{S_k^i\in\mathscr{A}_j^i} E_G(S_k^i,\delta) + E_T(S_k^i,\alpha) - E_R(S_k^i) \quad (2)$$

Since, with the provisions taken, this energy is separable in S_k^i, its minimization is trivial: a segment S_k^i is added to $\mathscr{A}_j^i$ only if this add makes the global energy decrease thus if the corresponding energy term is negative. In case two *image* segments overlap when projected on M_j^i, only the one that has the lowest matching energy is kept. In this case, we may indeed assume that one of the two segments is not a correct matching hypothesis. Some examples of matches obtained through this method are shown in figure 2.c

In the following, we will try to minimize the squared distance between each *model* edge M_j^i and the matched *image* segments in the set $\mathscr{A}_j^i$. It can be easily proved that this problem reduces to minimizing the distance between M_j^i and the principal axis $\mathscr{D}_j^i$ of $\mathscr{A}_j^i$ (figure 2.d). $\mathscr{D}_j^i$ is defined as the line supplying the best approximation for a set of given segments. Using a canonical representation of a line as $d_j^i(\mathbf{x}) = -x\sin(\theta_j^i) + y\cos(\theta_j^i) - \rho_j^i = 0$, it boils down to the search of parameters θ_j^i and ρ_j^i minimizing the criteria:

$$\begin{aligned}\sum_{S_k^i \in \mathscr{A}_j^i} \int_{\mathbf{x}\in S_k^i} \left(d_j^i(\mathbf{x})\right)^2 d\mathbf{x} &= \sum_{S_k^i \in \mathscr{A}_j^i} \int_0^1 \left(t d_j^i(\mathbf{a}_k^i) + (1-t) d_j^i(\mathbf{b}_k^i)\right)^2 \|\mathbf{s}_k^i\| dt \\ &= \sum_{S_k^i \in \mathscr{A}_j^i} \frac{\|\mathbf{s}_k^i\|}{6} \left(d_j^i(\mathbf{a}_k^i)^2 + d_j^i(\mathbf{b}_k^i)^2 + 4 d_j^i \left(\frac{\mathbf{a}_k^i + \mathbf{b}_k^i}{2}\right)^2 \right)\end{aligned} \tag{3}$$

Where $\mathbf{a}_k^i$ and $\mathbf{b}_k^i$ are extremities of $\mathbf{s}_k^i$. We refer the reader to [7] for the solution to this classical problem.

Because the energies are separable, this method is much faster and simpler to implement than a simulated annealing or a genetic algorithm running on all possible matchings as Beveridge proposes in [3]. In practice, the results of the matching process are very satisfying, except when the model is too far from the reference, or when there is a very important image segment close to the correct image segment (shadows can so easily disturb the matching steps).

3 Constraints Inferring and Enforcing

3.1 Constraints Inferring

This section briefly recalls the procedures used to infer the constraints from an approximate 3D model. The reader can refer to [18] for more details. As stated previously, the algorithm assumes an initial approximate 3D model that may come from automatic algorithms [19] and *which topological relations* are known. In a first step, approximate plane normals of the different facets are clustered, leading to a restricted set of directions $\mathbf{v}$. The vertical normal is added to these directions because of its obvious importance in man-made scene. Each facet of the model is then linked to a direction, which will be useful to build the constrained system.

Constraints between plane normals have to be determined on this reduced set of directions. To achieve this, a so called constraints graph is deduced from these clusters. In this unoriented graph, each direction is a node and constraints are represented as arcs. Several types of valued arcs are indeed added between two nodes $\mathbf{v}_i$ and $\mathbf{v}_j$ whenever a condition of *orthogonality*, *vertical symmetry* or *horizontal line* is approximatively verified according to an angular threshold. Whereas the clustering step was performed to handle parallelism properties and horizontal planes, the constraints graph integrates all the other geometrical constraints that will be applied on the model.

One of the difficulties of methods performing an implicit parameterization of the model as done in section 3 [4, 11] is that directions must be computed sequentially so that they can be deduced from some previous ones in the sequence and some constraint rules. As shown in [4], it is equivalent to orient the constraints graph *while ensuring that no over-constraint occurs.* The algorithm described in detail in [18] explicitly builds this sequence of directions by iteratively integrating the directions (including the vertical direction) in the sequence and orienting the constraints of the arcs graph. During this process, some constraints may disappear because of some possible cases of over-constraints but heuristic procedures try to maximize the number of constraints kept in the final model. The dependences graph is the result of the orientation of the constraints graph.

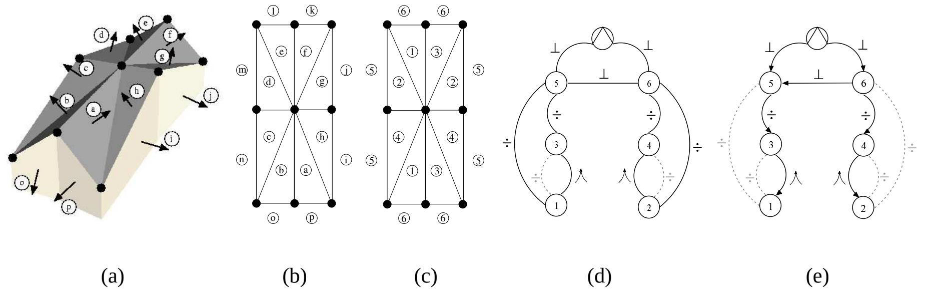


Figure 3: Approximate 3D building (a) and its schematic representation with directions (letters) annotated on facets (b) where black circles are initial approximate points that do not verify constraints and vertical planes making up the border are inferred from them. (c) shows the result of clustering. (d) and (e) depict constraints graph and dependences graph respectively. Each arc represents a constraint between two directions. $\textcircled{\scriptsize\wedge}$ defines the vertical direction, $\perp$, $\div$ and $\curlywedge$ symbolize orthogonality, horizontal edge and vertical symmetry constraints respectively. Grey arcs are constraints suppressed from the graph (see text for details). The sequence of directions of this dependences graph is $(\textcircled{\scriptsize\wedge}, 5, 4, 2, 6, 3, 1)$ and renumbering can be performed.

From the previous procedure, each direction may thus be numbered so that they can be computed sequentially, each one being defined by the previous ones and some constraints applied on it given by edges of the dependences graph. Whenever a direction $\mathbf{v}_i$ is not completely constrained by previous directions in the sequence (for instance perpendicular directions that depend on only one other direction), one assumes that it depends on some parameters $\boldsymbol{\theta}_i$. Then rules recalled in [18] enable to sequentially compute each direction from these parameters and directions of dependences so that *geometrical constraints imposed to them are verified*. It can also be shown that derivatives $\partial \mathbf{v}_i / \partial \boldsymbol{\theta}_k$ can easily be deduced. They will be of use in the iterative minimization.

3.2 Geometrical Parameterization

This section mostly recalls principles of the method depicted in [10] and [18] As explained above, each direction is computed from some parameters $\boldsymbol{\theta} = [\boldsymbol{\theta}_1^T, \ldots, \boldsymbol{\theta}_D^T]^T$. Each pair of points of one facet belong to the same plane which normal is linked to direction $\mathbf{v}_i$ if their coordinates $\mathbf{X}_m$, $\mathbf{X}_n$ verify the planarity constraint:

$$\mathbf{v}^T(\mathbf{X}_m - \mathbf{X}_n) = 0 \tag{4}$$

Concatenating all the equations for all facets leads to $\mathbf{B}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_D)\mathbf{X} = \mathbb{0}$ where $\mathbf{X} = [\mathbf{X}_1^T, \ldots, \mathbf{X}_N^T]^T$ holds all the points coordinates and $\mathbf{B}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_D)$ a $P \times N$ matrix, holds the geometrical constraints. [10] shows that an *implicit parameterization* of the points $\mathbf{X}$ is then given by:

$$\mathbf{X} = \mathbf{U}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_D)\mathbf{V} \tag{5}$$

with $\mathbf{V} \in \mathbb{R}^M$ and where $\mathbf{U}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_D)$ is a $3N \times M$ matrix which columns form an orthonormal basis of the nullspace of $\mathbf{B}$. Thus $\mathbf{X}$ *implicitly verifies all the geometrical constraints and* $\mathbf{V}$ *holds the implicit parameters of the shape*. By sequential construction, the plane normals verify the geometrical constraints and thus *any value* for $\mathbf{V}$ gives a solution for the constrained system. Consequently, M is the number of degrees of freedom of $\mathbf{X}$.
The unknowns, $\boldsymbol{\theta}$ and $\mathbf{V}$ are collected in a single vector $\boldsymbol{\Theta} = [\boldsymbol{\theta}, \mathbf{V}]$. [10] shows also procedures to ensure that this expression is differentiable and that $\nabla\mathbf{X}(\boldsymbol{\Theta})$ can be computed using the partial derivatives of directions and chain rule.

Some over-constraints may occur during this geometrical parameterization. However, these situations are easily detected and can be dealt with by iteratively removing constraints until no more degeneracies or over-constraints occur [18].

4 Fitting the Constrained Model

4.1 Problem Statement

For each *model* edge M_j^i, section 2 explains how to extract a main axis $\mathscr{D}_j^i$ from the image i. $\mathscr{D}_j^i$ is defined by its unit normal vector $\mathbf{n}_j^i$ and its distance to the origin ρ_j^i . Moreover, the parameterization done in the previous section provides a function $\mathbf{X}(\boldsymbol{\Theta})$ giving the 3D coordinates of all the vertices of the model according to a set of variables $\boldsymbol{\Theta} = [\boldsymbol{\theta}, \mathbf{V}]$. We are now going to explain how we fit this constrained model on the main axis $\mathscr{D}_j^i$.

The goal is to minimize the distance between the projections M_j^i of each edge M_j of the constrained model and the corresponding axis $\mathscr{D}_j^i$ in each image i:

$$Q_E(\boldsymbol{\Theta}) = \sum_{i,j} \int_{\mathbf{x} \in \{M_j^i\}} \left(d_j^i(\mathbf{x})\right)^2 d\mathbf{x} \tag{6}$$

where:

$$d_j^i(\mathbf{x}) = -x\sin(\theta_j^i) + y\cos(\theta_j^i) - \rho_j^i = \mathbf{n}_j^i.\mathbf{x} - \rho_i \tag{7}$$

is the distance from point $\mathbf{x}$ to the axis $\mathscr{D}_j^i$. From the computations made in identity 3, it can be shown that Q_E depends only on the extremities of M_j^i, which are the vertices of the 3D model projected in each image. The projection matrices are assumed to be known for all images and the 3D coordinates of the vertices are contained in $\mathbf{X}(\boldsymbol{\Theta})$. It proves thus that Q_E depends only on $\boldsymbol{\Theta}$. Let us remark that if $\mathscr{A}_j^i$ is empty (no *image* segment matches the edge), $\mathscr{D}_j^i$ is not defined and the corresponding term is simply omitted in the sum.

4.2 Numerical Scheme

As in [10] a Levenberg Marquardt (LVM) method is used to minimize the equation (6). To have a well conditioned numerical scheme, normalization of points coordinates is performed in 2D and 3D as advised in [12]. For simplicity, notations are not changed in the following.

From the identities (3) and (7), equation (6) can be rewritten with matrix notations:

$$Q_E(\boldsymbol{\Theta}) = \sum_i \frac{1}{2} ||\mathbf{N}^i \mathbf{x}^i - \mathbf{b}^i||^2 \tag{8}$$

where $\mathbf{x}^i$ contains the coordinates of edge extremities of M^i, which are the coordinates of the model vertices projected in image i, $\mathbf{N}^i$ depends on the normals $\mathbf{n}^i_j$ to the axis $\mathscr{D}^i_j$, and $\mathbf{b}^i$ on ρ^i_j.

The minimization of (8) by LVM will require to compute its derivatives:

$$\nabla Q_E(\boldsymbol{\Theta}) = \sum_i \nabla \frac{1}{2} ||\mathbf{N}^i \mathbf{x}^i(\mathbf{X}^i(\Theta)) - \mathbf{b}^i||^2 = \sum_i (\mathbf{N}^i \mathbf{x}^i - \mathbf{b}^i)^T \mathbf{N}^i \nabla \mathbf{x}^i \nabla \mathbf{X}(\Theta) \tag{9}$$

$\nabla\mathbf{X}$ can be computed as explained in section 3. $\nabla\mathbf{x}^i$ is deduced from the classical projection formula linking $\mathbf{X}$ and its projection $\mathbf{x}^i$ in image i:

$$\begin{bmatrix} s\mathbf{x}^i \\ s \end{bmatrix} = \mathscr{P}^i \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbb{A}^i & \mathbb{B}^i \\ \mathbb{C}^i & \mathbb{D}^i \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix} \tag{10}$$

where $\mathscr{P}^i$ is the projection matrix (assumed to be known) of image i. The derivation yields:

$$\nabla \mathbf{x}^i(\mathbf{X}) = \frac{\mathbb{A}^i}{s} - \frac{(\mathbb{A}^i\mathbf{X} + \mathbb{B}^i)\mathbb{C}^i}{s^2} \qquad s = \mathbb{C}^i\mathbf{X} + \mathbb{D}^i \tag{11}$$

LVM can now be run to fit the constrained model on the axes, therefore find the best $\boldsymbol{\Theta}$ to minimize (8). Results obtained are shown and commented in the next section.

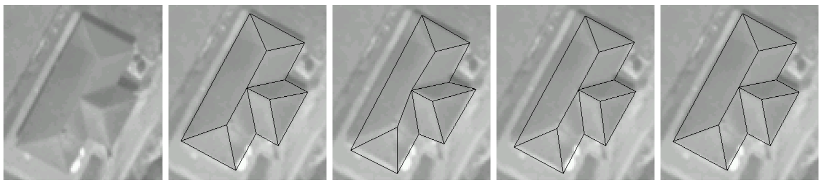
5 Results

5.1 General Protocol

An example of results is presented in figure 4. In the following benchmarks performed for simulation purpose, we use a series of images such as (a) and a hand input reference (b) assumed to be perfect. An initial *model* (c), imperfect by definition is supposed to come from a general but imprecise reconstruction process. In our case, it is derived from the reference (b) by adding noise. Finally, (d) shows the application of constraints on the building without image fitting [18], and (e) the fitting of the constrained model using image information.

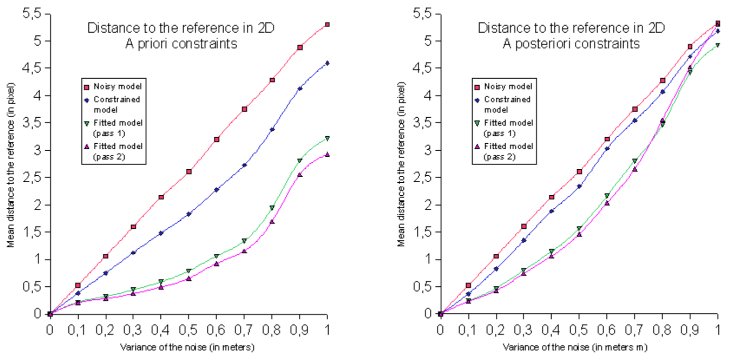
5.2 Evaluation

The validity of our approach has been assessed though several benchmarks. The efficiency is measured by comparing the result of the fitting with the reference. The distance between



(a) Image of a building (b) Reference (c) Model (d) Constrained model (e) Fitted constrained model

Figure 4: Result of a fitting process

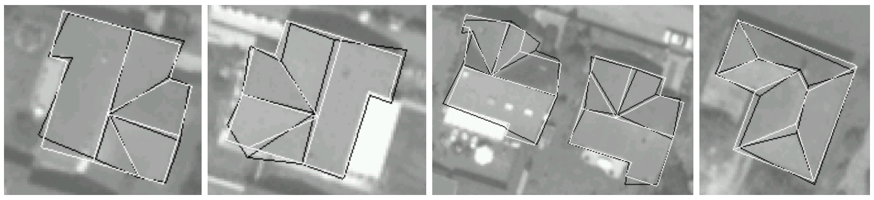


(a) a priori constraints (b) a posteriori constraints

Figure 5: Distance to the reference

the models is computed by averaging the distances between the projection of the vertices of the model and those of the reference. The benchmark was done on 25 buildings, with a variance for the noise varying from 0 to 1 meter. Moreover, we ran 15 tests for each building and each value for the noise variance. Finally, the tests consisted in simply applying the constraints to the model, then applying the fitting, then making a second pass. The idea underlying this second pass is that the matching process will be more efficient and have less errors if the model has already been fitted once, so that a second iteration, relying on a more precise matching will enhance the quality.

In the case of a priori constraints, the figure 5 (a) shows a significant improvement in comparison to the mere application of constraints, in particular for moderate values for the noise. In the case of a posteriori constraints, the figure 5 (b) still shows a improvement through our method, even if less significant. This comes from the fact that the non-detection of constraints, and especially the detection of wrong constraints penalizes a lot the process, as this will prevent the constrained building from conforming to the image.



(a) (b) (c) (d)

Figure 6: Other examples of fitting results: model is in black, result of the fitting in white

6 Conclusion

The method presented in this paper automatically fits a constrained 3D model in images. Constraints are directly inferred from an initial approximate model and the constrained result is compliant with the provided constraints through implicit parameterization. In the algorithm depicted here, the model is fitted using images information, and results show the significant gain in precision obtained compared with the sole application of constraints on the model.

Several important theoretical problems remain unsolved, especially in the construction of constrained models. A better formalization of the constraints seems necessary to avoid the arbitrary suppression of constraints during the constraint graph orientation as it is done for the moment. This could also lead to a prediction of the degenerescency of the constrained model during its construction, and to ensure the creation of non-degenerated constrained models without heuristic procedures.

References

[1] B. Ameri and D. Fritsch. Automatic 3D building reconstruction using plane-roof structures. In *ASPRS*, Washington DC, 2000.

[2] J.R. Beveridge and E.M. Riseman. Optimal geometric matching under full 3D perspective. *CVIU*, 61(3):351–364, 1995.

[3] J.R. Beveridge and E.M. Riseman. How easy is matching 2D line models using local search. *IEEE Trans. on PAMI*, 19(6):564–579, June 1997.

[4] D. Bondyfalat. *Interaction entre Symbolique et Numérique; Application à la Vision Artificielle.* Thèse de doctorat, Univ de Nice Sophia-Antipolis, 2000.

[5] P.E. Debevec, C.J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: a hybrid geometry-and image-based approach. In *Proc. of SIGGRAPH 96*, pages 11–20, August 1996.

[6] R. Deriche. Using canny's criteria to derive a recursively implemented optimal edge detector. *IJCV*, 1(2):167–187, May 1987.

[7] R. Deriche, R. Vaillant, and O. Faugeras. From noisy edges points to 3d reconstruction of a scene : a robust approach and its uncertainty analysis. In *Theory and Applications of Image Analysis*, volume 2 of *Series in Machine Perception and Artificial Intelligence*, pages 71–79. World Scientific, 1992.

[8] A. Fischer, T. Kolbe, F. Lang, A. Cremers, W. Förstner, L. Plümer, and V. Steinhage. Extracting buildings from aerial images using hierarchical aggregation in 2D and 3D. *CVIU*, 72(2):163–185, November 1998.

[9] P. Garnesson and G. Giraudon. Polygonal approximation overview and perspectives. Technical Report RR-1621, INRIA, jun 1991.

[10] E. Grossmann. *Maximum Likelihood 3D Reconstruction From One or More Uncalibrated Views Under Geometric Constraints*. PhD thesis, IST, 2002.

[11] E. Grossmann and J. Santos-Victor. Maximum likelihood 3D reconstruction from one or more images under geometric constraints. In *Proc. of BMVC*, 2002.

[12] R. Hartley. A linear method for reconstruction from lines and points. In *Proc. of ICCV*, pages 882–887, 1995.

[13] C. Jaynes, E. Riseman, and A. Hanson. Recognition and reconstruction of buldings from multiple aerial images. *CVIU*, 90(1):68–98, April 2003.

[14] S. Noronha and R. Nevatia. Detection and modeling of buildings from multiple aerial images. *IEEE Trans. on PAMI*, 23(5):501–518, May 2001.

[15] H-Y. Shum, M. Han, and R. Szeliski. Interactive construction of 3d models from panoramic mosaics. In *Proc. of CVPR*, pages 427–433, 1998.

[16] P.F. Sturm and Maybank S.J. A method for interactive 3d reconstruction of piecewise planar objects from single views. In *Proc. BMVC*, pages 265–274, 1999.

[17] I. Suveg and G. Vosselman. 3D reconstruction of building models. In *Proc. of XIXth ISPRS Congress*, pages B3:538–545, 2000.

[18] F. Taillandier. Inferring and enforcing geometrical constraints on a 3d model for building reconstruction. In *Proc. of SCIA*, LNCS, pages 699–708. Springer, June 2005.

[19] F. Taillandier and R. Deriche. Automatic buildings reconstruction from aerial images: a generic bayesian framework. In *Proc. of the XXth ISPRS Congress*, 2004.

[20] F-A. van den Heuvel. A line-photogrammetric mathematical model for the reconstruction of polyhedral objects. In *Videometrics VI, SPIE*, volume 3641, pages 60–71, 1999.

[21] F.A van den Heuvel and M.G. Vosselman. Efficient 3D-modelling of buildings using a priori geometric object information. In *SPIE*, volume 3174, pages 38–49, 1997.