Scene Reconstruction Based on Constraints: Details on the Equation System Decomposition

Gilles Trombettoni[1] and Marta Wilczkowiak[2]

[1] COPRIN Project, I3S-INRIA-CERMICS, 2004 route des lucioles, 06902 Sophia Antipolis cedex, B.P. 93, France
trombe@sophia.inria.fr
[2] MOVI Project, INRIA Rhône-Alpes, 655 avenue de l'Europe, Montbonnot, 38334 Saint Ismier cedex, France
Marta.Wilczkowiak@inrialpes.fr

Abstract. We present a new approach to 3D scene modeling based on geometrical constraints. Contrary to most of the existing methods, we obtain 3D scene models that respect the given constraints *exactly*. Our tool can describe a large variety of linear and non-linear constraints in a flexible way.

Our approach is based on a dictionary of so-called *r-methods*, based on theorems in geometry, which can solve a subset of geometrical constraints in a very efficient way. Two fast and complete graph-based algorithms are proposed to find a reduced parameterization of a scene, and to decompose the equation system in a sequence of r-methods.

1 Introduction

Reconstruction of accurate and photorealistic 3D models is one of the most challenging tasks in Computer Vision. In this paper, we address the problem of image-based reconstruction of a scene respecting a set of geometrical constraints. Defining geometrical constraints between scene primitives and incorporating them into the reconstruction system helps to stabilize the calibration, improves the quality of the model and limits the number of required images.

A common approach consists in incorporating the constraints into the optimization process. These methods however are often costly. Furthermore, they guarantee neither the convergence nor the (exact) constraint satisfaction.

Our model acquisition approach is detailed in [7]. It is divided into three main phases: initialization, constraint planning and optimization.

Initialization. In addition to 2D images, geometric objects and constraints must be defined. The 3D model is represented by **points**, **lines** and **planes**. They are subject to linear and non–linear constraints such as **distance**, **incidence**, **parallelism** and **orthogonality**.

An initial reconstruction is provided by a quasi-linear approach exploiting projections and geometrical constraints [6]. After this phase, all the variables (camera and model parameters) have an initial value.

Constraint planning. Our model reconstruction system requires a set of **r-methods** which allows us to decompose the whole equation system into small subsystems. An r-method [5] is a predefined routine used to solve a subset of geometric constraints. An *r-method* computes the coordinates of *output objects* based on the current value of *input object* coordinates, and satisfies the underlying constraints between input and output objects. For example, an r-method computes the parameters of a line based on the current position of two points incident to this line.

Several r-method patterns have been incorporated in a dictionary used by our system. They correspond to standard theorems of geometry. The constraint planning is divided into two steps:

1. *R-method addition phase:* Add *automatically* in the equation graph all the r-methods corresponding to r-method patterns present in the dictionary.
2. *Planning phase:* Perform `GPDOF` [5][1] on the enriched equation graph. `GPDOF` produces a set of **input parameters** and a sequence of r-methods (called *plan*) to be executed one by one. Input parameters are a subset of the variables describing the scene such that, when a value is given to them, there exists a finite set of solutions for the system satisfying the constraints.

Model optimization. The optimization process[2] only adjusts the input parameters. Every time the cost function is computed (inside the numerical algorithm), the r-methods in the plan are executed, producing a new value for the other variables such that all the constraints are satisfied. The detailed process can be found in [7].

Contribution

Many works have focused on incorporating geometrical constraints for camera calibration and 3D reconstruction including [3, 2]. The reader will refer to [7] for more details on the existing approaches which often require costly computations or do not guarantee to provide a solution. The approach presented in this paper overcomes these drawbacks. It is complete, fast and can be used to model non-linear constraints like distances, angles and distance/angle ratios.

This paper focus on the constraint planning process (Section 2) and shows experimental results in Section 3.

2 Constraint Planning

This section details the algorithms necessary for the constraint planning.

2.1 Automatic addition of r-methods

The automatic addition of r-methods is essentially based on a simple subgraph isomorphism algorithm performed on the constraint graph. When a subgraph matches an entry in the dictionary, the corresponding r-methods are added to the equation graph. Two steps are performed:

[1] `GPDOF` stands for General Propagation of Degrees of Freedom.

[2] based on a standard numerical algorithm and minimizing the *reprojection errors*

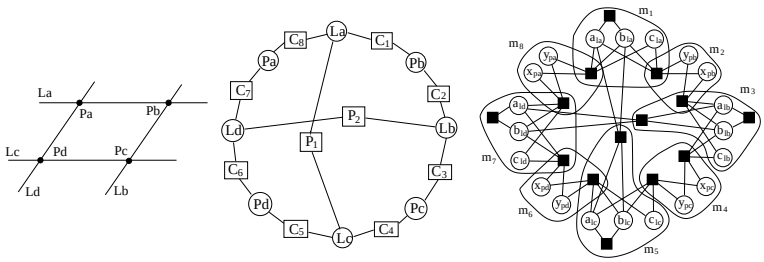


Fig. 1. Left: A didactic 2D scene describing a parallelogram in terms of lines, points, incidence constraints and parallelism constraints. **Center:** The corresponding constraint graph. It contains 4 points P_a,...,P_d, 4 lines L_a,...,L_d, 8 incidence constraints C_1,...,C_8 and 2 parallelism constraints P_1, P_2. **Right:** The enriched equation graph after automatic addition of r-methods. Equations are represented by rectangles and variables by circles. An r-method is represented by a hyper-arc including equations and output variables. Only 8 of the 16 r-methods are depicted for the sake of clarity. These r-methods match one of the three following patterns: line incident to two points (e.g., r-methods m_1 and m_7); point at the intersection of two known lines (m_2, m_4, m_6, m_8); line passing through a known point and parallel to another line (m_3, m_5).

-1- The first step explores all the connected subgraphs with size at most a small value k equal to the maximum number of nodes (objects+constraints) implied in any r-method of the dictionary, e.g., 7 in our tool. Starting from every single node, the subgraphs are built by incrementally adding a neighbor node to the current connected subgraph until the size k is reached. This depth first search algorithm is a simplification of the algorithmic scheme presented in [1]. The key idea allowing the algorithm to explore a *tree* of subgraphs is to consider at each step only a specific subset of selected neighbors, depending on a unique numbering of the nodes [1].

In practice, the time complexity of this algorithm is linear in the actual number of connected subgraphs of size less than k (which is $O(n^k)$). It is acceptable for small values of k and sparse graphs.

-2- For every found subgraph, a second procedure compares it with the subgraph patterns in our dictionary implemented as a hash table, which eliminates most of the subgraphs. A final comparison is made by a combinatorial process [3] inspired by the solving process of CSPs (BT). In short, objects in the subgraph are reordered to be matched with objects in a subgraph pattern. If the subgraph matches, the corresponding r-methods are added to the equation graph.

2.2 The GPDOF algorithm

GPDOF [5] works on an enriched equation graph. It computes a sequence of r-methods to be executed for satisfying all the equations. GPDOF solves this combinatorial problem in polynomial-time and is quasi-linear in practice. It performs the three following steps until no more equation remains in the equation graph G (success) or no more free r-method is available (failure)[4]:

[3] Deciding whether two graphs are *isomorphic* is still an open problem.

[4] In this case, one obtains an incomplete plan which solves only a subpart of the equations (geometric constraints) and more parameters are adjusted by optimization.

1. select a **free** r-method m [5],
2. remove from G the equations and the output variables of m,
3. create all the *submethods* of a r-method m_i that share equations or output variables with m.

A plan can be obtained by reversing the selection order: the first selected r-method will be executed last. The first two steps above define the standard PDOF local propagation algorithm [4] on which GPDOF is based (PDOF accepts only r-methods solving *one* equation.) Selecting iteratively free r-methods ensures that no loop is created in the plan.

It turns out that, when r-methods can solve several equations, there is no guarantee that PDOF finds a plan, even if one exists. This highlights the notion of *submethod* which renders GPDOF complete. In short, the notion of submethod explains that a partially removed r-method remains available for a future selection. The reader will refer to [5] to get a more detailed information.

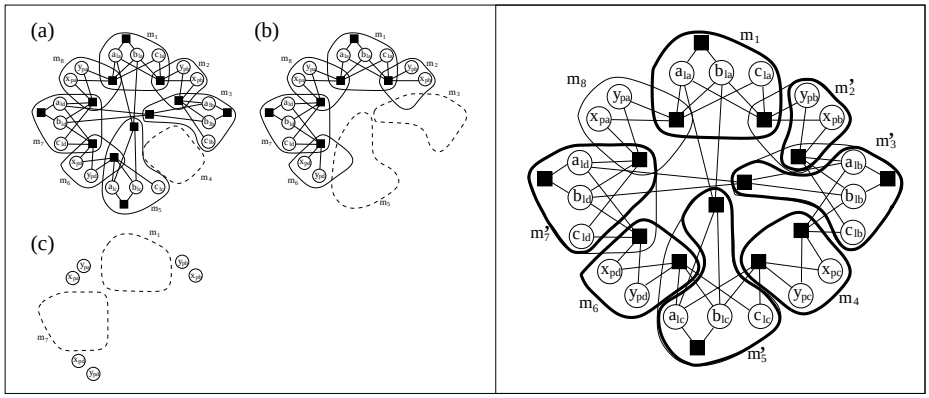


Fig. 2. Two possible planning phases performed by GPDOF on the didactic scene. **Left:** At the beginning, r-methods m_2, m_4, m_6, m_8 are free, so that one of them is selected, e.g., m_4. **(a)** This selection implies the removal of the equations and the output variables of m_4 from the equation graph. **(b)** This frees r-methods m_3 and m_5 which are selected and removed next in any order. **(c)** The r-methods m_1 and m_7 are then free and can be selected. The process ends since no more constraint remains in the equation graph. The obtained plan is the sequence (m_1, m_7, m_3, m_5, m_4). **Right:** GPDOF may also select first m_6 which is free. The third step of GPDOF then creates the submethod m'_5 of m_5 and the submethod m'_7 of m_7. The process continues and selects m_4, m'_5, m_1, m'_2, m'_3, and finally m'_7. Selected r-methods (m_1, m_4, m_6) and submethods (m'_2, m'_3, m'_5, m'_7) are represented by thick hyper-arcs.

2.3 Determining the input parameters

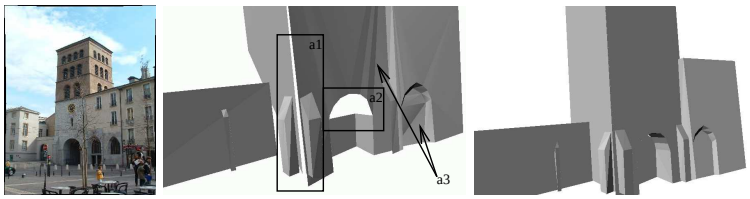
The input parameters modified by the numerical optimization simply consist of the variables which are output by no r-method in the plan. This yields the 6 coordinates of points P_a, P_b, P_d for the plan illustrated in Fig. 2-left- or the 2 coordinates of point P_a for the plan illustrated in Fig. 2-right-. Due to the selection of submethods, the values of variables in a second set of parameters are

[5] Output variables of a free r-method appear in no "external" equations.

read a first time (recall that every variable has an initial value) and computed later by r-methods, e.g., the coordinates of P_b are in this set (Fig. 2-right-). The other variables are only modified by r-method execution. This subtlety cannot be explained here due to a lack of space.

3 Results

We have used our approach to build a model of a church (see Figure 3). Five images architectural plans (distances) were used. The scene includes 137 constraints (including 10 distances), 251 equations, 119 objects, 427 variables. The time for the constraint planning (2 min. on a `Pentium IV 2GHz`) is dominated by the exploration of the connected subgraphs. 2213 r-methods have been added automatically. The execution time of `GPDOF` is negligible. The plan was built of 107 r-methods and is executed in 55 ms.



(a) (b) (c)

Fig. 3. (a)–One of the five photos used for the reconstruction; (b)–Some artifacts of the unconstrained model. (c)–The constrained model after optimization corrects the artifacts.

4 Acknowlegments

Thanks to D. Chancel for the architectural drawings. Thanks to E. Boyer, D. Daney, C. Jermann, B. Neveu and P. Sturm for useful discussions.

References

1. David Avis and Komei Fukuda. Reverse Search Enumeration. *Discrete Applied Mathematics*, 6:21–46, 1996.
2. Didier Bondyfalat, Bernard Mourrain, and Theodore Papadopoulo. An application of automatic theorem proving in computer vision. In *Automated Deduction in Geometry*, pages 207–231, 1998.
3. P.E. Debevec, C.J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: a hybrid geometry-and image-based approach. In SIGGRAPH '96, *New Orleans*, August 1996.
4. Ivan Sutherland. *Sketchpad: A Man-Machine Graphical Communication System.* PhD thesis, Department of Electrical Engineering, MIT, 1963.
5. Gilles Trombettoni. A Polynomial Time Local Propagation Algorithm for General Dataflow Constraint Problems. In *Proc. Constraint Programming CP'98, LNCS 1520 (Springer Verlag)*, pages 432–446, 1998.
6. M. Wilczkowiak, P. Sturm, and E. Boyer. The analysis of ambiguous solutions in linear systems and its application to computer vision. In *To Appear in Proceedings of the 14th British Machine Vision Conference, Norwich, England*, September 2003.
7. M. Wilczkowiak, G. Trombettoni, C. Jermann, P. Sturm, and E. Boyer. Scene reconstruction based on constraint decomposition techniques. In *Proceedings of the 9th International Conference on Computer Vision*, 2003.