A generative computer tool to model shadings and openings that achieve sunlighting properties in architectural design

Daniel Siret

▶ **To cite this version:**

Daniel Siret. A generative computer tool to model shadings and openings that achieve sunlighting properties in architectural design. ENVIROSOFT'96, Sep 1996, Como, Italy. p. 695-704. halshs-02472137

HAL Id: halshs-02472137

https://shs.hal.science/halshs-02472137

Submitted on 10 Feb 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A generative computer tool to model shadings and openings that achieve sunlighting properties in architectural design

D. Siret
Laboratoire CERMA Ura CNRS, École d'Architecture de Nantes
Rue Massenet, 44300 Nantes, France

Abstract

We present a generative tool based on a new simulation method for making sunlighting an actual formgiver in architectural design. Our system reverses the common simulation process. It works with intuitive sunlighting properties such as “ this area must be sunless the afternon in summer ”. Given a property, it computes a complex geometrical volume figuring the sunlighting phenomenon in time and space. This volume provides a visualization of the sunlighting constraint and it enables the designer to model the different solutions — shadings or openings — that exactly check on the given property. We illustrate the system with two demonstrative examples in architectural design and we introduce some new developments under consideration at the present time.

1 Generative computer tools purpose

1.1 Generative vs evaluative tools

In architectural and urban design context, many simulation tools have been developped for appraising interactions between constructions and their environment. Most of these tools are based on an evaluative approach. They use a simplified or a physical model of a phenomenon to compute the resulting state of the phenomenon in the future construction : thermal response, lighting intensity, noise level, and so on.

These simulation tools generally need a full description of buildings (topology, geometry, materials). In architectural and urban design processes, they cannot be used before the main schemes are completed. As they work on definite plans, these evaluative tools do not suit design practice. They come at the latest stage of design, whereas key decisions are already taken. Whenever the computing results do not match the architect’s wishes or the program cons-

Development and Applications of Computer Techniques to Environmental Studies VI
Proceedings of ENVIROSOFT’96, Como, Italie
COMPUTATIONAL MECHANICS PUBLICATIONS, Sept. 1996, pp. 695-704

traints, the design has to be started again or, more often when possible, corrective solutions must be found. This is the well-known iterative process :

→ design → modeling → simulation → correction

As opposed to this evaluative process, generative computer tools emphasize a constructive point of view. A definition of these generative tools has been recently proposed by L. Khemlani [1] : « These are tools which are able to generate alternative solutions for various limited aspects of the design that satisfy some given well-definable specification. A generative process is essentially one of searching through all possible solutions to a given problem to find those that meet specific goals. The idea of generative tools is *not* to produce *the* solution to a problem (...). Instead, a generative tool aims to produce alternative solutions meeting *some* objective criteria, possibly in the form of ‘ tests ’. Using more subjective, non-computable criteria, the architect can then make a selection from among the solutions offered by the computer, with modifications and refinements if necessary. »

Following this definition, generative approach reverses the common simulation process in environmental design. Its purpose is not to compute the state of a phenomenon in a given construction, but rather to suggest ‘ all possible ’ constructions that achieve a given state of a phenomenon for a given environment. Thus, for generative computer tools, the question to answer is not : *what kind of properties can I observe for this construction in this environment ?* but rather : *what kind of construction(s) must I build in this environment to achieve these properties ?* In that way, generative tools suit design practice, as suggested by R. Woodbury [2] since 1991.

1.2 The reverse simulation approaches

Generative tool starts from result — the property to achieve — to produce alternative solutions that make this property real. Two ways of realizing this reverse simulation process can be found : the ‘ Generate and test ’ method which is used by Khemlani to design windows configurations from an energy counscious point of view, and the ‘ Inverse simulation ’ method that we advocate in this paper, following Schoeneman & al. [3]

1.2.1 Generate and test method

For a specified environmental property, this method consists of :

- firstly, defining and generating a set of solutions available to design,
- then, evaluating all these solutions to find those that meet the given property.

Khemlani used this method for window design. Different windows configurations available are first generated (within important constraints in order to reduce the combinatorial explosion and to enable the simulation) and then, they are tested with a simple simulation model. The subset of windows that achieve the desired property, from which the designer may choose one, is considered as the solution to the problem.

1.2.2 Inverse simulation method

The second method considers that generative tools reverse the classical process of simulation, that is in some way, they inverse the simulation model. Given a property to achieve, this method consists of :

- firstly, computing an inverse model of the simulation process for the given property,
- then, using this model to define alternative solutions.

All solutions that the inverse model provides achieve the property. As the set of solutions may be infinite, new kinds of properties can be brought in to explore them from different points of view. This method has been proposed by Schoeneman & al. to determine the light intensities that most closely match a desired lighting effect drawn in an architectural scene. It was there introduced within the larger field of ‘ inverse problems ’ studied in physical Sciences.

Inverse simulation method is opposed to direct one as shown on figure 1 below. The diagrams illustrate these opposite processes for a given phenomenon (the arrow) in a given context (the profile of a simple building). Direct simulation methodology consists of computing the state of the phenomenon (the gray shape) for the given context. On the contrary, inverse simulation starts from the result. Given a state of the phenomenon, it consists of computing the ‘ solutions ’ which enable the construction to generate this state. For instance, these solutions may describe the way of transforming the building in order to achieve the given state, if necessary.

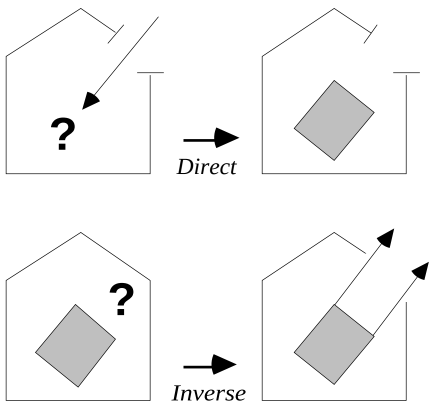

Figure 1. Direct and inverse simulation processes

2 A method for the inverse simulation of sunlighting

Direct sunlighting simulation is a well-known process based on geometrical methods. It consits of computing all sunless or sunlit shapes in a given geometrical construction, for a given environment and a given time period [4].

The inverse problem is less known. It aims at determining the solutions that achieve a given sunless or sunlit geometrical spot in a given construction, for a given time period. These solutions are members of two wide families of architectural objects : the openings (holes) that achieve ‘ sunlit ’ properties, and the shadings that achieve ‘ sunless ’ properties. It is clear that, at least for shadings, many alternative solutions can be found to check on a given sunlighting property (a characteristic of inverse problems).

2.1 The sunlighting property to achieve

Datas and results are reversed in both direct and inverse simulation processes. For the inverse sunlighting problem, datas are the parameters of the sunlighting property to achieve. Such a property can be formalized as a set of three parameters (P, S, T) where :

- P is the geometrical spot that receives sunlighting (a convex polygon),
- S is the sunlighting qualification (sunlit or sunless),
- T is the time period during which P must check on the qualification S.

Thus, a sunlighting property (P, S, T) is written : “ P must be S during T ”. It can be composed at any stage of the design process, to define the massings, to make windows, to draw sun visors, and so on.

2.2 The complex sunlighting volume associated with a property

The target is to make a given sunlighting property true : “ the spot P *must be* sunlit (or sunless) during the time period T ”. Our inverse simulation method is based on the computation of the complex sunlighting volume defined by P and T. We denote $\prod$(P, T) this volume (an artefact connecting time and space). Any sunlighting property (P, S, T) generates such a volume which embodies the exact set of points that may shade the polygon P for all the instants within the given time period T.

The three diagrams on figure 2 (next page) illustrate the method we use to build the sunlighting volume $\prod$ associated with a given property (P, S, T). Firstly, we compute the simple volume π defined for any point p of the plane of the polygon P, and for the time period T figured as a geometrical shape on the sky vault. The volume π embodies all the shading points for the point p during T. It has its vertex on p and its edges follow the solar directions p-t_i defined by the time interval T. As described in a previous paper [5], we use an original meshing of solar trajectories to make the shape T square with some intuitive lived time periods such as ‘ the end of the afternoon in spring ’, ‘ early morning in May ’, and so on.

Geometrically speaking, the complex volume $\prod$(P, T) results from the boolean union of all the simple volumes π set in each point of the polygon P. Assuming that P is convex, the result of this union squares with the displacement of π round the perimeter of P (figure 2, at the left bottom). This displacement produces the boundaries of the sunlighting volume $\prod$(P, T). It is computable using common boolean algorithms between polygons.

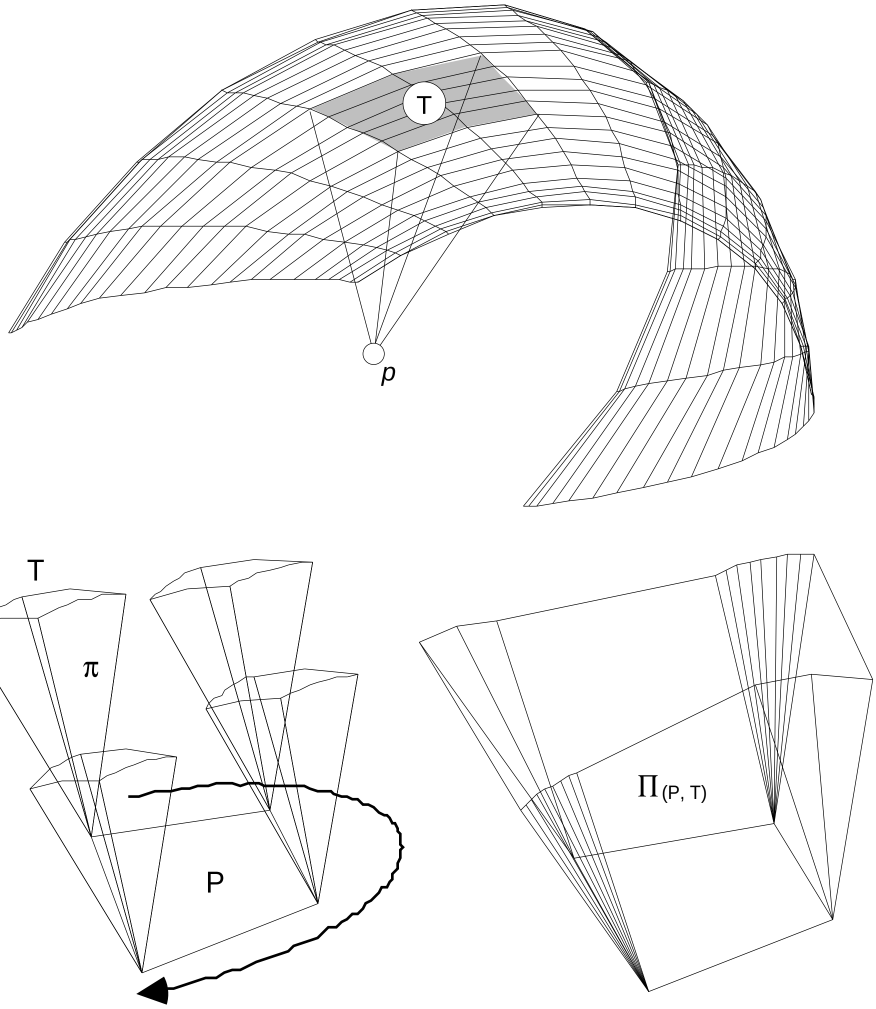

Figure 2. Three steps in the construction of ∏(P, T).

2.3 Shading and opening solutions

Given the volume ∏(P, T) associated with the property (P, S, T), one can define the solutions to the inverse sunlighting problem, that is : the openings or the shadings that enable the polygon P to be S during the time period T. In other words, these solutions make the property (P, S, T) become true.

An opening is any hole that lets the sunbeams reach the spot P during T. A shading is any object that makes a shadow on P during T. There always exists an opening or a set of openings (a shading or a set of shadings) that checks on a given sunlighting property.

Therefore, a sufficient condition for P to be sunless during T is that there exists at least one object intersecting the sunlighting volume ∏(P, T). Reciprocally, a necessary condition for P to be sunlit during T is that no object in the scene intersects, even partially, the volume ∏(P, T).

Development and Applications of Computer Techniques to Environmental Studies VI
Proceedings of ENVIROSOFT'96, Como, Italie
COMPUTATIONAL MECHANICS PUBLICATIONS, Sept. 1996, pp. 695-704

Given these simple rules, there are two ways of achieving a sunlighting property when necessary :

- to create the opening(s) resulting from the intersection between $\prod(P, T)$ and all the objects in the scene, for type (P, Sunlit, T) properties,
- to model at least one shading object that intersects the volume $\prod(P, T)$, for type (P, Sunless, T) properties.

Examples introduced in section 3.2 below illustrate both methods.

3 A generative tool for sunlighting design

3.1 Overview

We have developped a generative tool based on this inverse simulation method. This tool works in association with a common CAD system. It provides interactive implements to define a sunlighting property in any complex architectural or urban scene. Given such a property :

- a rectangle that figures the sunlighting zone P,
- the qualification *sunlit* or *sunless* S applied to this zone,
- an intuitive time period T,

… the system computes the associated sunlighting volume. Then, according to the qualification S, it generates new openings or shadings that achieve the property as explained below. In addition, our system provides direct simulation tools for appraising real sunlighting states before composing hypothetical properties.

3.1.1 Openings modeling

Openings that check on a ‘ sunlit ’ property are computed as exact geometrical intersections between the sunlighting volume associated with the property and all the objects of the scene. If an object cannot be drilled, the property cannot be achieved and the process fails. Otherwise, all the intersections are created. These openings must be then designed to get an architectural or urban relevance in the project context : windows, bays, places, etc. Facing the ‘ raw holes ’ computed by the system, a designer is free to give his own architectural interpretation. Of course, the initial spot P is slightly altered as exact openings are transformed into realistic windows.

3.1.2 Shadings modeling

Most of the time, the number of shadings that achieve a ‘ sunless ’ property is infinite. The system uses geometrical planes, that define polygons when intersecting the sunlighting volume, to figure shadings. It first suggests a default plane resulting in any polygon. Then, it enables the designer to graphically modify this plane and to explore different solutions. The shading polygon is automatically transformed as and when the plane is. Any kind of ‘ raw shading ’ can be outlined in that way : tree, façade, porche, sun visor, and so on. Such as openings, the exact shading polygons computed by the system have to be designed to get an architectural relevance in the project context.

3.2 Demonstrative examples in architectural design

Graphics on the next pages illustrate both methods. The project concerns a small building in France (latitude 47° N), which volumetry has been fisrt outlined following the program. Yet, two openings had to be made.

For the first one (figure 3 a, b, c, d), the architect decided to enable the area figured as the rectangle P to be sunlit during the time period T ‘ the middle of the morning in winter ’. Given the property (P, sunlit, T), the system computed the sunlighting volume shown on (b) and then, created the exact opening drawn on (c). The architect followed his own interpretation to design the window figured over the hole in graphic (d). As this window recovers the largest part of the exact opening, the resulting sunlighting spot for the time period T closely matches the rectangle P.

The form of the second opening (figure 4 a, b, c, d) has been decided under the shape of the rectangle P on the South-West façade. For such a large bay, the problem is to avoid summer-time overheating. Thus, the property (P, sunless, the beginning of afternoon in summer) has been first composed and the system computed the associated sunlighting volume shown on graphic (b). The process the designer used to explore alternative shading planes that achieve the property cannot be figured here. The result is an incline plane intersecting the sunlighting volume through the exact shading polygon shown on (c). Following his interpretation, the architect finally turned this polygon into an architectural sun-visor and designed the opening as shown on (d).

4 Future developments

These concern extensions to the existing tool which all converge to a best integration of the generative methodology into the design practice. Three main objectives are outlined below.

4.1 Management of numerous properties

Our system deals with properties by turns. A more suitable method would be to manage all properties together, that is to detect and to solve incompatible or redundant sunlighting effects during the design process. A solution to this problem consists of computing the boolean intersection and difference between the sunlighting volumes associated with the properties. Openings or shadings that achieve both properties must check on some rules related to the resulting partition : intersecting one part and not the others, for instance. Such rules can be formalized in a logical way. An intelligent tool working on these logic rules may infer relevant advices to manage properties together.

4.2 Achievement of fuzzy properties

Another limit of our system is the restriction to the binary qualification sunlit or sunless. A designer would prefer to deal with fuzzy sunlighting qualifications such as ‘ enough sunlit ’, ‘ not too sunless ’, and so on. That means he may accept that properties are roughly, rather than perfectly, achieved. Some of the fuzzy logic methods are under consideration to solve this problem.

4.3 Modeling of realistic openings and shadings

At the present stage of development, our system generates exact geometrical shapes figuring openings and shadings that achieve a given property. It is the designer's responsability to interprete and to transform these shapes into realistic shadings and openings that make sense according to the project context. However, an architect may prefer obtaining directly some relevant realistic objects, especially when these objects are difficult to model. An interesting example is the one of landscape design where ‘ tree-shadings ’ can be used for achieving sunlighting properties as well as visual ones. In that case, an architect would prefer working on alternative realistic trees, rather than on geometrical shapes. Our system could satisfy such a request by exploring a knowledge base of tree and plant features. For a given sunlighthing property, such an expert system would suggest alternative realistic tree-shadings solutions, according to their height, their volume, their leaves in winter or summer, and so on. Of course, this methodology could be extended to many kinds of relevant openings and shadings by using different architectural and urban knowledge bases.

5 Concluding remark

Advantages of generative approaches are numerous and quite still unexplored. Their scope could be extended to various environmental properties as well as architectural ones. The ‘ declarative methods ’ emphasized by Lucas & al.[6] and mainly applied to geometrical modeling, constitute an alternative way for such issue.

6 References

1. Khemlani. L., Genwin : A generative computer tool for window design in energy conscious architecture, *Building and Environment*, Vol. 30 n° 1, 1995
2. Woodbury R.F., Searching for designs : Paradigm and practice, *Building and Environment*, Vol. 26 n° 1, 1991
3. Schoeneman C., Dorsey J., Smits B., Arvo J., Greenberg D., Painting with Light, *Computer Graphics Proceedings*, Anaheim, California, 1993
4. Groleau D., Marenne C., Gadilhe A., Climatic simulation tools : an application for a building project in an urban space, in *Proc. of the 3th Inter. Conf. on Solar Energy in Architecture and Urban Planning*, Florence, Italy, 1993
5. Siret D., Declarative modeling methods related to ambient environment in architectural and urban design, *Revue internationale de CFAO et d'Informatique graphique*, Vol. 10 n° 5, 1995
6. Lucas M., Desmontils E., Declarative modelers, *Revue internationale de CFAO et d'Informatique graphique*, Vol. 10 n° 6, 1996

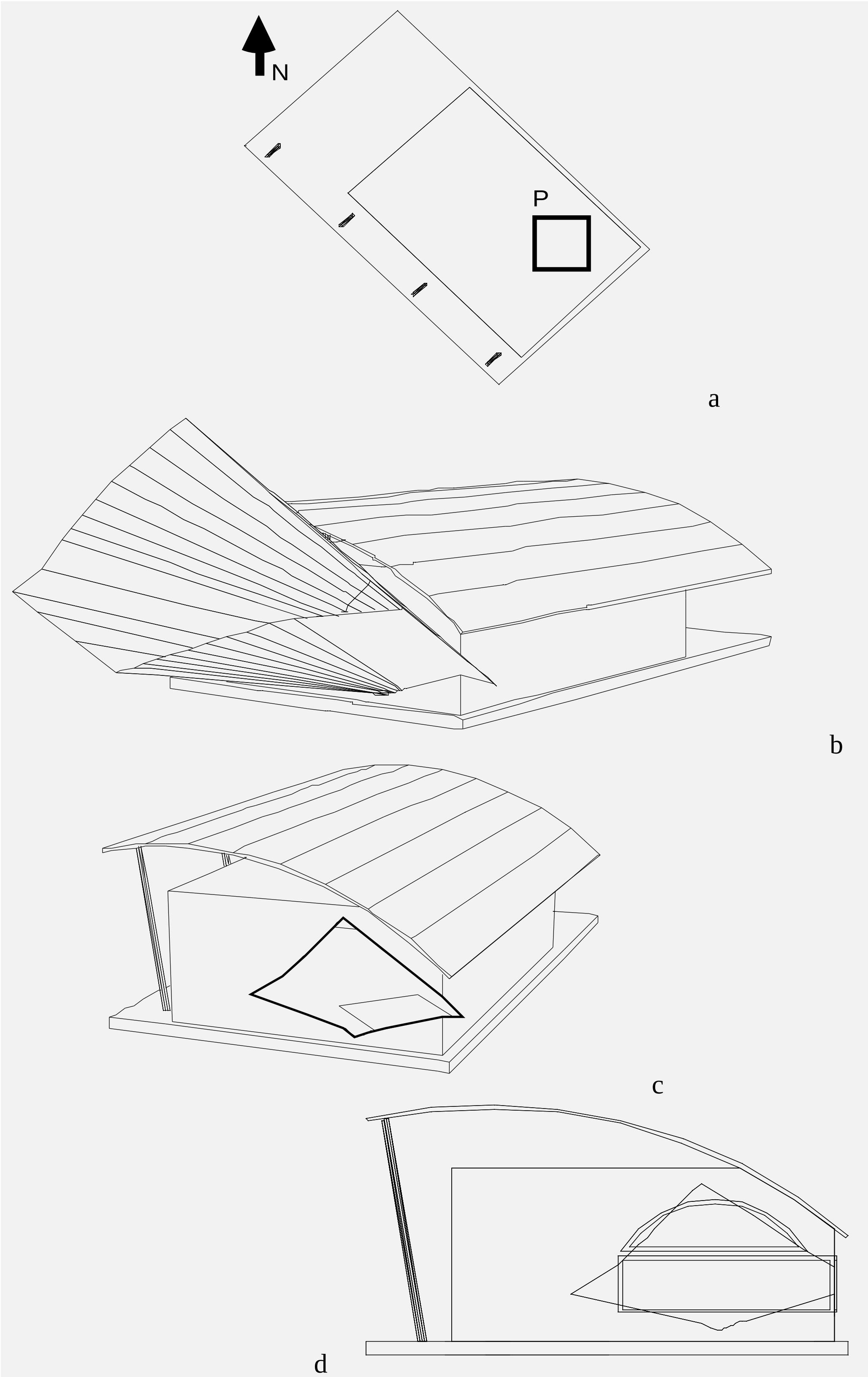

Figure 3. ‘ P, Sunlit, the Middle of the morning in winter ’

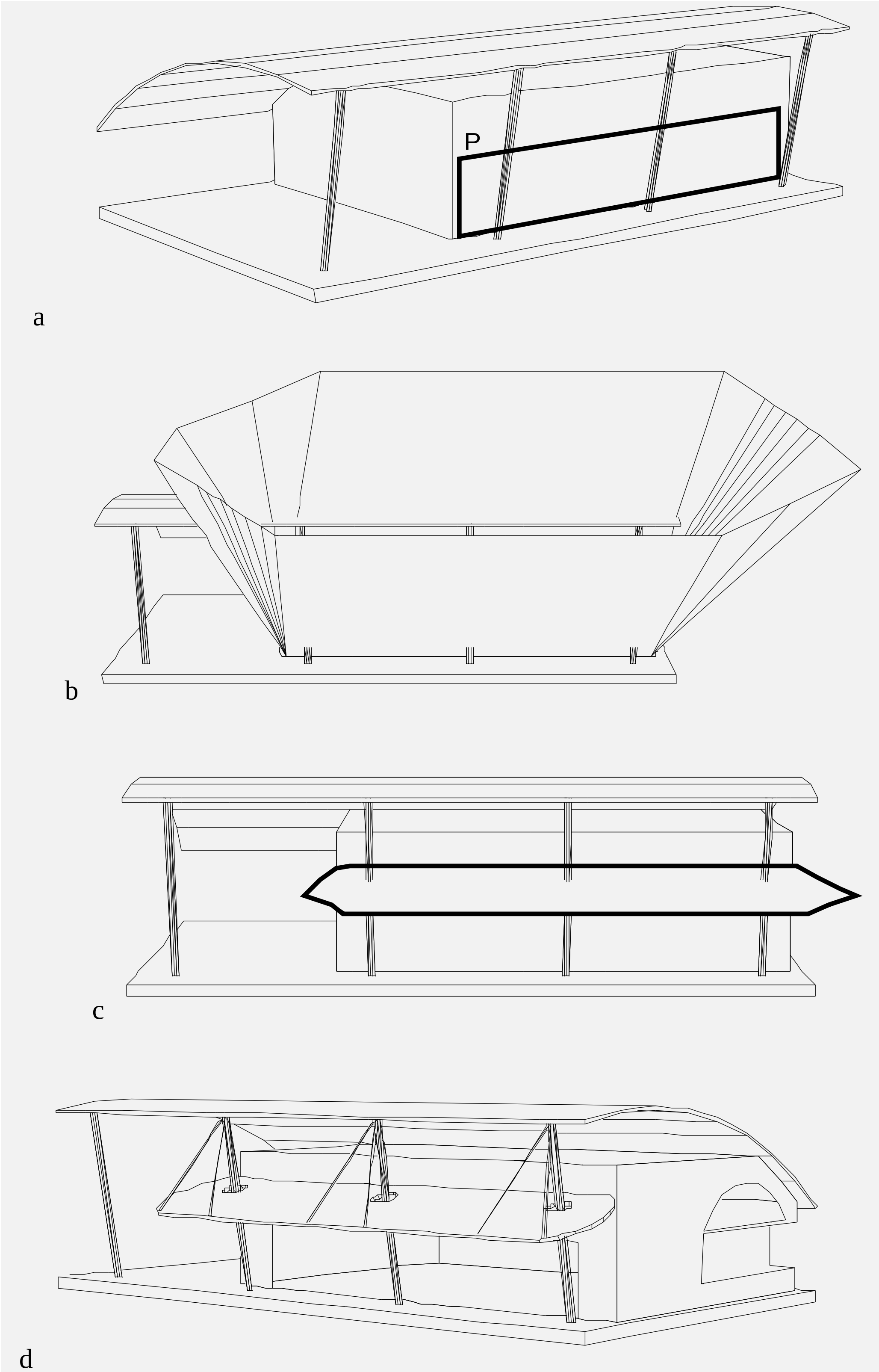

Figure 4. ‘ P, Sunless, the Beginning of afternoon in summer ’

Development and Applications of Computer Techniques to Environmental Studies VI
Proceedings of ENVIROSOFT’96, Como, Italie
COMPUTATIONAL MECHANICS PUBLICATIONS, Sept. 1996, pp. 695-704