An Image-based and Knowledge-based System for Efficient Architectural and Urban Modeling

D. Boucard[1,2], S. Huot[1], Ch. Colin[1], G. Hégron[2], D. Siret[2]

[1]CERMA, Ecole d'Architecture de Nantes
[2]Ecole des Mines de Nantes

Abstract

In this paper, we present two user-centered systems aiming at making easier the modeling of architectural and urban scenes by using two different but complementary approaches. The first one MArINa, an image-based modeler, allows the user to reconstruct urban scenes from one or more graphical documents. This method focuses more on reconstructing models and is more dedicated to the production of 3D sketches. The second modeler, MArCo is a knowledge-based modeler containing the know-from and know-how on classical architecture. It allows the user to model classical architectural scenes verifying automatically all the domain rules. Finally, we show how MArINa and MArCo can cooperate providing the user a tool combining efficiently their respective capabilities.

1 Introduction

Virtual worlds play an increasing role in design, communication, and research. But, surprisingly, even if computer graphics techniques produce very realistic models of architectural and urban environments, modeling a complex scene still remains a challenge for the designer. With classical modelers, she has to compose low level primitives producing objects without associated semantic. This characteristic is a real problem for architects or historians since this bunch of geometric primitives has no direct meaning in their mental universe and so, no domain rules are associated with them. Another problem with classic modelers when applied to architecture is the nature of data the designer owns. This information is partial and present in different media like drawings, texts, engravings, or even photographs. Current modeling tools are not adequate since they do not allow the user to directly use her documents. Before starting the modeling, she still needs a precise idea of the scene, and converts it into a set of primitives and operators known by the used modeler. Moreover, the user can not test different hypothesis since with classic modelers, it means remodeling an important part of the scene for each hypothesis.

To help the user reconstruct architectural and urban scenes in an efficient way, we propose two complementary tools: MArINa and MArCo. On the one hand, MArINa is an image-based modeler using projective geometry and allowing the user to rapidly reconstruct buildings from uncalibrated graphical documents (photographs, engravings or sketches). Using this method, the user does not need to model every detail since scenes, which are textured with excerpts of the given graphical documents, provide her a visually satisfactory model. On the other hand, MArCo is a knowledge-based modeler focusing on classical architecture. It allows the construction of scenes respecting the domain rules thanks to its high level knowledge on objects and their composition rules.

These two modelers are user-centered because we think that the system can be faster using the user knowledge. In fact, the user can give quickly more pertinent information than an automatic method. The system can not guess lacking information from initial data without the help from the user. Moreover, the user wants to manage the design process herself.

In this paper, we first present works related to 3D reconstruction, then we present our image-based modeler MArINa and our knowledge-based modeler MArCo. Finally, we show how MArINa and MArCo can work together providing a tool combining the advantages of both modelers, letting the user mix appearance and details approaches in the same project.

2 Related works

One kind of reconstruction tool in the architectural field is based on existing documents (often, photographs), but they rarely contain all the information. In order to take advantages of those aspects, our work focuses on the one hand on image-based reconstruction and on the other hand, on knowledge-based modelers. While the first approach takes advantages of graphical documents, the second one aims at completing missing information by using domain rules.

2.1 Image-based reconstruction

Methods involving static images belong to two families: the first one contains methods based on stereo algorithms, the second one on constraints.

- Stereo-based methods, like photogrammetry (American Society of Photogrammetry, 1966) or computer vision (Faugeras, 1993), are used to produce automatic systems. Since they are based on stereo correspondence methods they are sensitive to noise, light variations and deformation of perspective. Furthermore, most of them require calibrated images and methods like photogrammetry still have heavy needs in both computing time and hardware. In this article, we focus on interactive tools centered on the user, but not on taking its place, so automatic methods are not suitable in our case.
- Methods based on constraints allow the use of only one photograph on which the user has to specify geometric constraints (van den Heuvel, 1998) or projective ones (Liebowitzand, Criminisi and Zisserman, 1999). Here images do not need to be calibrated, but in both methods, reconstruction is done by using essentially expensive numerical approaches. As explained in part 3, MArINa is based on formal projective geometry, which allows fast reconstruction without additional loss of precision due to numerical approximations.

Finally, there exists an approach (Debevec, Taylor and Malik, 1996) mixing the previous ones in FACADE. Here, the user has to place predefined 3D primitives on calibrated images to make them match the buildings. With constraints contained in the primitives and photogrammetry, it renders 3D model based on those primitives. The drawbacks of this approach are that it requires calibrated images, that there is a limited set of primitives and finally that it is based on numerical approaches. Note that FACADE was a source of inspiration for MetaCreation's product Canoma (MetaCreation, 1999) which uses the principle of 3D primitives to place, but this time on uncalibrated images.

2.2 Knowledge-based architectural CAD

In our knowledge, there are few knowledge-based architectural modelers. Among them, we can cite the work of A. Chassagnou (Chassagnou et al., 1998), which aims at reconstructing gothic vaults from a map of the vault and a system using classification. From a set of parameters deduced from the map for objects, it is able to deduce from which class it belongs. When this deduction done, each reasoning applying to the class objects apply to it.

Another system, developed by O. Grau (Grau, 1998) combines a knowledge base with a photogrammetry system. The knowledge on objects allows the system to add new constraints on them. For instance, if a polygon is said to be a wall, the system is able to add it a verticality constraint.

The previous systems use a knowledge base to analyze objects from their characteristics in order to model them. In MArCo, the approach is quite different since it is a constructive one allowing the user to build scenes from scratch.

In an urban modeling approach, we can cite the works realized at Gamsau (School of Architecture of Marseille (France)) (Faure, 1988; Maltret and Zoller, 1996). Those works, for example, the Remus project uses a knowledge base to simulate and complete information about architectural and urban shapes. Information deduced is about parcels and constructions but also about data given by stereo-photogrammetrical retrieval methods. All information is encoded using two relations: "is-a" and "part-of". In MArCo, we use a more powerful representation of objects allowing a larger set of relations, which allows a more precise description.

Finally, MultiCAD (Miaoulis, Plemenos and Skourkas, 2000) uses a knowledge base to manage descriptions of objects and also the properties and relations the system can apply to them. The knowledge base here is exclusively hierarchical. In MArCo, the knowledge base is structured the same way the domain is, providing the domain expert a representation close to the concepts he usually uses.

3 MArINa

MArINa, stands for "Modeling Architecture Interactively and Naturally", is a system that assists users in obtaining 3D architectural models from uncalibrated views (photographs, paints or sketches). The user starts from images displayed in the background. She draws over them the shape of the parts that she wants to reconstruct in 3D and specifies constraints on them. The reconstruction is split into two worlds: the human world made of simple human tasks and the machine world made of simple computation tasks. The constraints and the parts of interests are known by the user but cannot be inferred by the machine yet. The elevation is tedious to perform manually, such as with a traditional modeling system, but is easy to perform with our formalism. Finally, textures are extracted from the images and the scene is rendered with textures mapped.

Our user-centered 3D reconstruction method is based on an original approach relying on projective geometry and Grassman-Cayley algebra. With this formal approach, computing 3D reconstruction out of 2D images is simple and fast. User drawing and constraints are translated into equations using Grassman-Cayley algebra and are sent to a special solver that reconstructs 3D objects out of these 2D shapes and constraints. The primary goal of this system is not to produce geometrically accurate models, but to allow people with limited computer graphics background to quickly reconstruct visually satisfactory scenes.

3.1 The user interface of MArINa

We designed the user interface of our system as a classical vector based drawing program with drawing and selection tools used to position basic objects such as points and lines on the images. This choice is different than Facade or Canoma where users must choose from a set of existing shapes and position them on the images. This is due to our ability to express constraints between basic objects whereas the other tools have the constraints hidden inside the predefined shapes. With MArINa, the user can draw any polyhedral shape using direct manipulation techniques, rubber-banding and dynamic cues for picking objects. This interface is closest to drawing than 3D modeling. It's well adapted to non-specialist users (even a child is able to redraw a shape with tracing paper).

Most constraints are specified directly by clicking on the primitives. Other properties are given using dialog boxes. As specifying a lot of constraints with interactive tools or dialog boxes can burden the user with a tedious task and to speed up the specification of constraints on well-known shapes, we have developed special tools. For example, one of the predefined shapes is the *rectangle*. This shape is the projection in 2D of a 3D rectangle. Therefore, in 3D, the four points are coplanar, lines are parallel and orthogonal. These constraints are automatically added to the scene database when this tool is used. However, these special tools are close to drawing macros since users can still specify new constraints on all the primitives or remove already existing constraints. They are predefined drawing sequences, so the user drawing task is not interrupted. Several shapes are predefined (like blocks or pyramids) but it is very easy for the user to define new ones for domain dependent editors.

3.2 Reconstruction steps

We now explain the method we use to construct a 3D model from one image. The user's task is incremental and can be divided into three steps:

1. Drawing and setting constraints,

2. Reconstructing the 3D model,

3. Displaying the result.

The user can repeat these steps - adding elements or constraints - to refine the model.

Drawing and setting constraints

The user draws on the image using drawing tools described in the previous section. She needs to specify the hidden parts of the drawing she wants to compute in 3D. She has to roughly estimate their position and declare them as hidden in the constraints manager. The kernel will then know that these parts are hidden for this point of view and will not use their projected 2D coordinates to compute their 3D coordinates. Of course, hidden parts can be computed if there are enough constraints on them.

3D model reconstruction

The kernel (Sosnov, Macé and Hégron, 2002) operates in two steps: 2D drawing correction, to eliminate perspective drawing errors due to possible camera distortions and 3D model computation. 3D reconstruction with the kernel could end up in failure in two particular cases. First, when a drawing can not be corrected to respect the constraints, the kernel is unable to correct the contradiction. This error is impossible if reconstruction is made from photographs that are not extremely distorted. The second error is due to missing constraints on objects. If the user doesn't specify enough constraints on the desired 3D model, the kernel will not be able to raise it. In this case, the kernel asks for more constraints on the model.

Displaying the result

3D scenes are displayed with a classical 3D rendering engine, by mapping reconstruction images on the recovered 3D model. We extract textures from the visible faces of the scenes in the image used for the reconstruction. For texture extraction, we use the drawing and information given by the user. Non-visible faces of the model in images are filled with a default color. The user can manipulate the view interactively.

Multiple images reconstruction

MArINa also allows the user to obtain 3D models from multiple images. We have designed a simple manual correspondence method. The idea is to perform local reconstructions in each image space and to merge resulting 3D models using correspondence points given by the user (four non-coplanar points for each pair of models, as explained in (Huang and Netravali, 1994)). To perform this task, the user identifies which part of the image is relevant for the complete 3D model and sets constraints in order to reconstruct it. Since the drawing is done on a little part of the image, it can be minimal. In this way, the user reconstruct the whole model little part by little part avoiding the repetition of constraints on every image. When accomplished, she has to make some correspondences between images to allow the system to reconstruct the whole model. When done, if enough information is given, MArINa reconstructs the model, else it gives an error message and lets the user modify the set of constraints.

3.3 Results

First, modeling a 3D scene from images with MArINa is based on drawing on images and on setting geometric constraints on objects. Thanks to our interactive tools and drawing primitives these task becomes simple. For example, the model of Figure 1 has been obtained in 6 minutes. This time takes into account all the steps of the process: drawing and setting constraints, 3D reconstruction and texture extraction. This model is made of 10 drawing primitives for a total of 20 points; there are 13 constraints specified manually (not including implicit constraints of primitives), for a total of 206 constraints. The drawing and constraining phase took 5 minutes, 3D reconstruction less than one second, and texture extraction 12 seconds for 8 textured faces. For these tests, we used a C++ implementation of the kernel, on a standard PC (600 MHz CPU with 128 Mb of RAM).

Figure 2(a) represents the only picture we have of a building. We don't have any information on its back face. Using MArINa, the user draws on the picture the visible parts and also the hidden ones like she imagines them. In this case, she obtains the 3D model shown in Figure 2(b). The visible parts are textured with excerpts of the picture and the hidden ones are colored with a default color. Figure 2(c) shows a sketch representing (maybe speculatively) the back of this building. Figure 2(d) highlights the merging of two types of documents used for the reconstruction and illustrates the bridge between 3D modeling and design. Thanks to the speed of the process and its simplicity, MArINa allows architects to explore several ideas and see them in context.

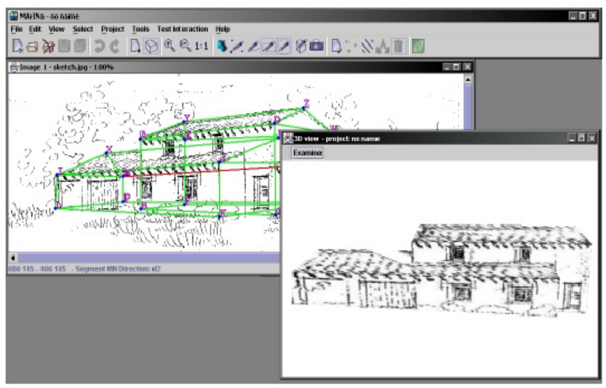
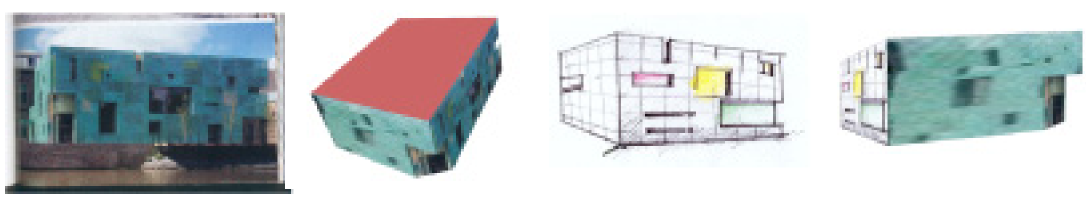


Figure 1. 3D reconstruction (right side) of a house from a sketch (left side).



a b c d

Figure 2. Different ways to retrieve a 3D model with MArINa: (a) An image of a building. (b) This image is a 3D reconstruction from the single image (a). Hidden parts have been estimated by the user. (c) A drawing of the back of this building. (d) This 3D model was obtaining by combining two type of documents: the image of (a) and the sketch of (c).

3.4 Limits

MArINa allows the user to easily reconstruct scenes from photographs. However, it has some limitations. The first one is its inability to guess which texture to map on faces that are hidden on all images, or that are not drawn as visible faces. In this case, the system has no information on which texture to use, so no texture is applied. The second limitation comes when there is an obstacle in front of a visible face. In this case, the system takes both face and obstacle on the texture to be mapped. So, if, on a photograph, there is a tree in front of a building, the texture will take the tree. The last limitation of MArINa is that it only uses textures to render the appearance of a building. Most of the time, it is sufficient to produce a visually satisfactory scene, but when the viewer is near than a building, they are not realistic enough. In fact, objects on textures appear to have no relief to give the appearance of a volume.

4 MArCo

The main drawback with classic approaches is that they do not have any semantic on manipulated objects. The user has to compose simple primitives provided by the modeler in order to get complex objects usually found in classical architecture. Thus, the user has to explicitly provide information that is implicit for her. Even, with MArINa, modeling details is time consuming since we fall in the same drawbacks than classical modelers with primitives. To allow the user to model easily details usually found in architecture in a simple way, the modeler need knowledge on objects. In order to manage domain knowledge, we have developed MArCo (Cognitive Architectural Modeler), a knowledge-based system specialized in architecture. However, to be efficient, this kind of systems requires a coherent corpus. With its regularity and its limited set of canonical objects, the well-described classical architecture is a good candidate for the first release of MArCo. Since this domain remains large, we have begun our study with a subset limited to the columns of the five orders defined by Vignola (Vignola, 1563; Pérouse de Montclos, 2000) before enlarging the knowledge.

4.1 Levels of abstraction

Formalizing this corpus led us to consider three different levels of abstraction to represent objects.

- The first one, the knowledge contains descriptions of architectural objects and the know-how on them. This representation is done using conceptual graphs, a knowledge representation formalism using two kinds of nodes (relations and concepts). Concepts can represent every entity used in the domain. It can be a physical object like a column, a mathematical one like a variable or even an abstract one like an axis. Concept nodes can also contain a conceptual graph representing its description (Figure 3). Relation nodes represent relations between objects represented by concept nodes. For example, it can express that a window is centered in a bay. In our case, the description of an object is done by using a decomposition in architect's terms. Thus, a column is decomposed into a shaft, a base and a capital. This formalism using labels to distinguish its nodes (relations or concepts), it is possible to use architectural terms directly in the knowledge base, ensuring an easier communication with the architect. For instance, in the case of a column (figure 3), the concept representing the shaft is simply named shaft.

- The second one, the constraints permit to express intra- and inter-objects constraints of the domain. They are mathematical relations between objects parameters. They can represent an esthetical or physical aspect. For example, as an esthetic constraint, Victor Louis (French architect of the Age of Enlightenment) (Taillard, 1991) expressed that the radius of a place can not exceed three times the height of buildings around it. An example of physical constraint may be that the distance between two objects supporting a beam made of stone can not be too long elsewhere the beam will break under its own weight.

- Finally, the third one is the geometry, which is used to make calculus or display objects.

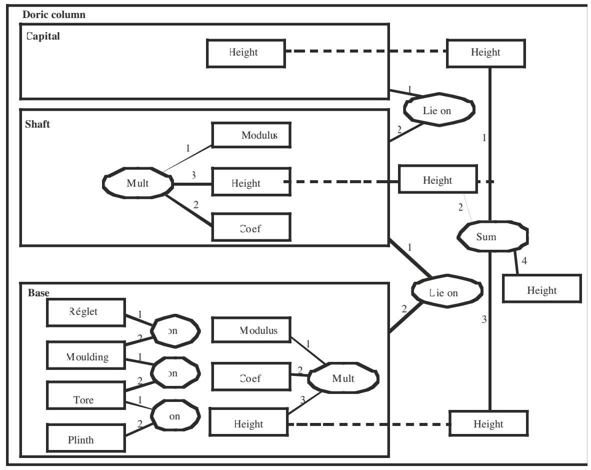


Figure 3. Doric column represented with conceptual graphs.

4.2 Managing objects at different levels

Using the levels of abstraction described in the previous section, it is possible to translate information from one level to a lower level, but it induces a loss of semantics. For instance, if we translate the knowledge "a is on top of b", we get a constraint between the coordinates of a and b. It is not possible to invert this process because one can not know if it is a casual case or not. To avoid problems, we always use the highest representation possible for a given object. This characteristic allows, on the one hand, the manipulation of incompletely described models. For example, it is possible to manipulate a colonnade without knowing the number of columns it contains. On the other hand, computations are done only when absolutely required, for example, when the systems need to generate the geometry of the scene. This characteristic has a strong impact on performances. Since the highest level of representation is used, scene manipulation is done in a more compact way, avoiding unrequired computations, and so, making it faster. For example, it avoids the performance problems expressed in (AlSayyad, Elliott and Kalay, 1996).

4.3 Customized visualization

Since conceptual graphs allow a multi-level representation, it can be extended to generate a customized geometric representation. This can be useful since classical architecture objects are often very detailed, making a scene difficult to handle in real-time even with recent computers. However, the designer often focuses on a little part of the scene at once, so having a fully detailed scene at every moment is not always necessary. In this way, the user can define its own level of precision by specifying its center of interest. To handle that, the knowledge base associates a drawing function (that can produce an approximation object) to each relevant architectural concept. When drawing the scene, the system looks for every object in the scene. If this object matches the precision required by the user, it uses the approximation object associated, otherwise, it recursively draws the concepts present in its description using the same method. Since the geometry of the scene is always generated from its representation in knowledge form, there is no loss of information and even rendered in an approximated way; objects keep the semantics and behavior of the architectural object they are an approximation. Thus, even rendered as a block (like in Figure 6), a Tuscan column keeps its behavior, so every rule applying to the column is respected by the block. In this way, since manipulating simpler geometry requires less resource, scene manipulation is faster.

4.4 Reasoning and constraints verification

Since the knowledge base contains domain constraints, created models are correct from an architectural point of view. Numerical constraints ensure that proportions are respected, while symbolic constraints (not included yet) will ensure the correctness in a symbolic way. An example of symbolic constraint is the rule forbidding an order (example: Tuscan) to be placed on top of a higher-level order (example: ionic).

Knowledge allows also reasoning using the domain rules. This reasoning is done using rules provided by conceptual graphs: given a graph hypothesis, it permits to add new knowledge allowed by this hypothesis. This mechanism allows the completion of models.

4.5 Results

MArCo is suited to model scenes with canonical objects, like columns (Figure 4.) and pilasters, that the user can modify. Figure 6 represents a scene containing 59 columns. The user gives a brief description: the endings of the colonnades, the radius and angle of circular paths, the number and type of columns. Then, she specifies that spaces between columns in all the colonnades are the same. The system produces the scene in less than one minute on a Pentium 3 733Mhz. To put into perspective this result, this time must be compared to the time the user would have spent if she used a classical modeler. In fact, she would have to create the models and then to place them respecting all the domain rules. Figure 6 (higher-left) shows an example of multi-level representation. Here, the user's center of interest is the central column, while peripheral ones are only used to give the general aspect of the place. That is why the central column is highly detailed, while others are automatically simplified. So, the user can work on the main object while seeing its integration into the scene, and manipulate easily the scene because most of the objects are represented with a simple approximation. By using the short description given by the user (see above) to produce the scene shown in figure 6, the system has generated 710 concepts, 42 relations and 185 constraints to produce it.

MArCo allows also managing the know-how of the domain. So in (Boucard and Colin, 2002), one can find how MArCo manages alignments of architectural objects. For example, Figure 5 was created by specifying a rhythm, the kind of present objects and their position within the rhythm, the number of objects and the path (begin, end, radius). The system has generated 341 concepts, 15 relations and 81 constraints and the geometry creation has been done in approximately 25 seconds.

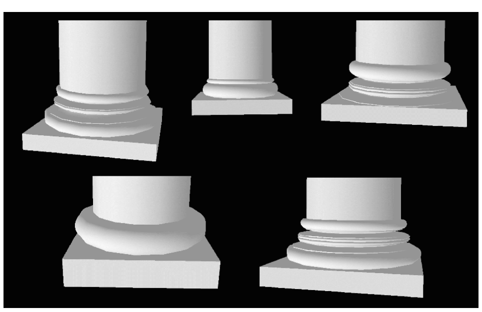


Figure 4. Columns base of 5 orders

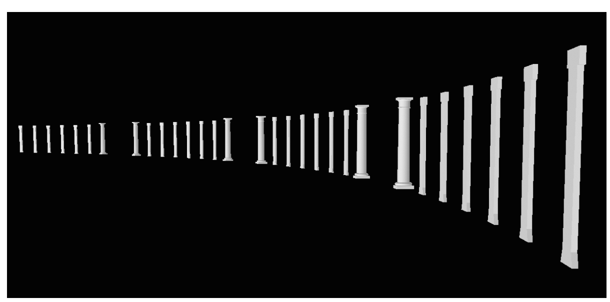


Figure 5. Alignment of columns & pilasters

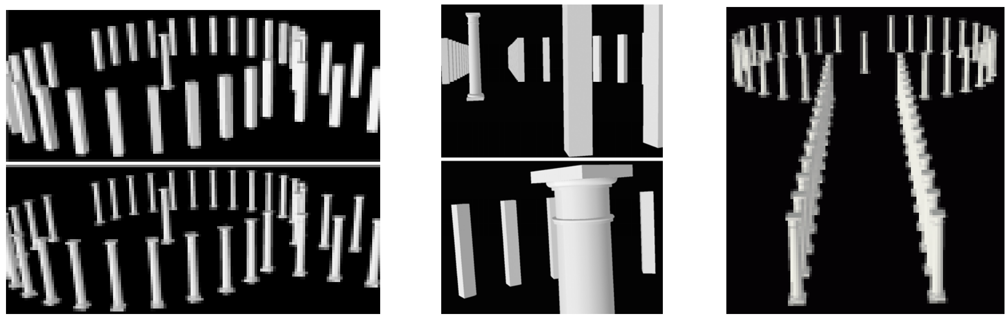


Figure 6. Multi-level representation of a scene with 59 columns

5 Perspectives on merging MArCo with MArINa

As announced, MArINa and MArCo have different but complementary approaches. So merging them in one modeler would help to reduce their limitations. Our goal here is to provide the user a tool combining both approaches. Depending on her whishes, she can either privilege the image-based modeling side, if details can be approximated by a texture, or privilege the knowledge-based one to model every detail. She can also mix both approaches on the same objects to model some parts with a high detail level and the others with textures. In this section we present the benefits of merging both approaches, from the MArCo point of view and from the MArINa one.

5.1 Benefits for MArCo

The main problem of MArCo and knowledge-based systems in general is their inability to manage objects not found in their knowledge base. In our case, since the knowledge base contains the classical architecture, it is quite limiting to reconstruct old buildings and their environment because those areas are rarely composed exclusively with classical architecture objects. So, it is necessary to MArCo to manage external objects as if they were in the knowledge base. To accomplish that, we shall use a minimal set of knowledge composed with information given by the user and information common to every geometrical object like its position, orientation, and size. Here MArINa can provide objects modeled by image-based reconstruction. MArCo can also benefit from MArINa by using its modules of texture extraction and manipulation, so objects present in the knowledge base can be textured using real world images. Finally, since knowledge base objects are managed from descriptions, MArCo can use MArINa's interaction to allow the user to place them in the scene. For example, it can be helpful to draw a block on the scene to compute its position and dimensions by using the MArINa approach. After reconstruction, the system can use it as a bounding box and will adapt automatically object dimensions to the drawn bounding box.

5.2 Benefits for MArINa

MArINa's main limitation is its inability to handle with occultation. For that, a knowledge base can be useful to bring semantics on objects. This semantics can be useful in order to complete missing information on objects. For example, if we know how much bays are present on a building, it is possible to guess using constraints where they are and then to duplicate texture of a non-hidden bay to map on the place of a hidden one or to replace it by a 3D object coming from the knowledge base. Knowledge on buildings can also help guessing what kind of texture is needed for parts that are hidden on all graphical documents used for the reconstruction. Finally, it is possible to increase the realism of scenes by adding 3D detailed objects to the textured model produced by MArINa. Actually, textures are visually satisfactory when the model is far from the viewer, but near, shapes are too flat to be realistic. In this case, including relief shapes increases the realism of the rendering even when the user is near to the object.

6 Conclusion

In this paper, we have presented two tools, based on different approaches, whose goal is to make easier 3D scene reconstruction and design. The first one, MArINa allows fast 3D reconstruction of urban scenes from one or more graphical documents. It favors more the appearance of the model (production of a 3D sketch) than its precise representation. The second one, MArCo contains domain knowledge and, thus, allows the user to get more detailed scenes while verifying all the domain rules. In essence, it is restricted to a specific domain. Thus, it can not generate objects missing from its knowledge base because they do not belong to this domain. Since both tools have limits, but one's limit is the strength of the other, mixing them will produce a modeler combining efficiently both approaches. Since the user manages the design process, she can produce customized scenes with textures and details where it is really needed.

References

AlSayyad N., Elliott A., Kalay Y. (1996), Narrative models: a database approach to modeling medieval Cairo, *ACADIA'96*, Tucson, Arizona, 1996.

American Society Of Photogrammetry (1966), *Manual of Photogrammetry*, 1966.

Boucard D., Colin Ch. (2002), MArCo, a knowledge-based architectural modeler and its application to the alignment problem, *3IA'2002*, May 2002, Limoges, France.

Chassagnou A., Guéna F., Maille M., Untersteller L.P. (1998), Modelisation and calculation of gothic or ogive vaults, *First European congress on restoration of gothic cathedrals*, Europa Congress Hall, Vitoria - Gasteiz, May 1998.

Debevec P.E., Taylor C.J., Malik J. (1996), Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach, *SIGGRAPH'96*, New Orleans, Louisiana, 1996.

Faugeras O. (1993), *Three-dimensional computer vision, a geometric viewpoint*, MIT press, 1993.

Faure S. (1998), *Simulation urbaine, modèle général de représentation de la connaissance urbaine*, Ecole d'architecture de Marseille (France), June 1988.

Grau O. (1998), 3D modelling of buildings using high-level knowledge, *CGI'98*, Hannover, Germany, June 1998.

van den Heuvel F.A. (1998), 3D reconstruction from a single image using geometric constraints, *ISPRS journal of Photogrammetry & Remote sensing*, 53, 1998.

Huang T.S. and Netravali A.N. (1994), Motion and structure from feature correspondences: A review, *Proceedings of the IEEE*, 82(2):252-268, February 1994.

Liebowitz D., Criminisi A., Zisserman A. (1999), Creating architectural models from images, *EUROGRAPHICS'99*.

Maltret J-L., Zoller J. (1996), Simulation of architectural and urban morphology, *OEEPE Workshop on 3D-city models*, Bonn, October 1996.

MetaCreations International Ltd. (1999), Wilson House, Fenian Street, Dublin 2, Ireland, *Canoma's users guide*, 1999.

Miaoulis G., Plemenos D., Skourkas C. (2000), MultiCAD Database: toward an unified data and knowledge representation for declarative scene modelling, *3IA'2000*, Limoges, France, 2000.

Pérouse de Montclos J.M. (2000), *Architecture : méthode et vocabulaire*, Editions du patrimoine, ISBN 2-85822-593-1, 2000.

Sosnov A., Macé P., Hégron G. (2002), Semi-metric formal 3D reconstruction from perspective sketches, In *Proceedings of International Conference on Computational Science (ICCS)*, p285-294, Amsterdam, The Netherlands, 2002.

Taillard C. (1991), *Victor Louis 1731-1800*, Thèse de Doctorat d'État, Université Paris 4, 1991.

Vignola, Le (1563), *Regola delli cinque ordini d'architettura di M. Iacomo Barozzio da Vignola.*