ECOLE CENTRALE PARIS

P H D T H E S I S

to obtain the title of

Doctor of Ecole Centrale Paris
Specialty : APPLIED MATHEMATICS

Defended by

Loïc SIMON

Procedural Reconstruction of Buildings:

Towards Large Scale Automatic 3D Modeling of Urban Environments

prepared at Ecole Centrale de Paris, MAS laboratory

defended on July 25, 2011

Jury :

Chairman :	Luc VAN GOOL	-	ETHZ Zürich
Reviewers :	Frank DELLAERT	-	Georgia Tech
	Martial HEBERT	-	Carnegie Mellon University
	Jean-Michel MOREL	-	ENS Cachan
Advisor :	Nikos PARAGIOS	-	Ecole Centrale Paris
Examiners :	Edmond BOYER	-	INRIA Rhône-Alpes
	Nicolas PAPARODITIS	-	IGN France

Acknowledgments

I would like here to express all my gratitude to the persons who have helped me complete this PhD thesis.

First, I would like to thank Nikos Paragios who has supervised my work with much investment. Not only did he propose me an exciting topic for my thesis but he truly followed my progress and was always encouraging whenever difficulties arose. In addition to his great scientific support, I could also appreciate his valuable human qualities. He has indeed settled the most profitable conditions to carry out research in his team and it was always a pleasure discussing with him about diverse topics.

Then, I am grateful to Jean-Michel Morel, Frank Dellaert and Martial Hebert for their fruitful reviews on my manuscript. They have made it a point of honor to comment thoroughly on my work, pointing out the tiniest inconsistencies while being generous with enthusiastic remarks. I also address many thanks to Luc Van Gool, Edmond Boyer and Nicolas Paparoditis for participating to my thesis committee. It was a privilege to have such well-known researchers consider my work while theirs have been a considerable source of inspiration.

I had besides the opportunity to work closely with two amazing collaborators: Olivier Teboul and Panos Koutsourakis. We have shared so many things - passionate debates, laughter, and pressure - that it created deep bounds. They were sort of my PhD brothers. I had also the chance to collaborate with Chaohui Wang and Haithem Boussaid for a shorter period but with no less joy. I would also like to address special thanks to those who accepted the tedious task of proofreading my thesis before I dared present it to the referees. These brave fellows are Ahmed, Olivier, Fabrice and Mihai.

As many PhD students may confirm it, a thesis is not all about work. Fortunately, it comes with a lot of fun parts as well, even at the least expected moments (I am obviously talking about deadlines here). I am first thinking of those who shared my office: Olivier (with him everything was a matter of writing a python script), Panos (alias the navy guy), Aris (misunderstood supporter of Panathinaikos and of splendid theories about trees and civilization), Chaohui (aka Xiao Jiji, master of sandwiches and dear “nami” of mine), Salma (and her daughter), Ahmed (Boubes, expert on trousers), and Radhouene. Of course, I do not forget the rest of the team with whom it has been a real pleasure debating and laughing during our inescapable coffee/tea breaks: crazy BoBo (thanks for the tea, although it almost killed me once), Katerina, Haithem, Stavros^{TM} (life-saving audience in case of flops), Mickaël, Fabrice, Sarah, Nicolas, Pierre-Yves, Stavros II, Martin, Kostas, Fatima,

Georg, Charlotte, Olivier Juan, Regis and Daniel. Finally, I am thinking of Krishna, Sylvain, PingPing, Jose, Rola and Pascale. They only passed a short while in the lab, yet we have spent memorable moments together and I hope we will meet again soon.

On a larger scale, I have quite enjoyed my stay in the MAS laboratory, and that was granted thanks to many cheerful members. Sissi was of course one if not the most important of them. I always appreciated the numerous occasions of chatting with her and Annie as they always had nice stories to tell. Besides these two central figures of the lab, I would like to thank Eric Herbin, Marc Aiguier, Celine Hudelot, Iasonas Kokkinos and Pascal Laurent who offered me to teach very well organized classes. I should also express my gratitude to Dimitris Samaras who, as the most loyal guest in our team, has shown a lot of interest in my work and my well-being. The doctoral school secretaries, Geraldine and Catherine deserve also my gratitude, since they have always been supportive even when I was far behind schedule with my paperwork.

I would like to thank Luc Van Gool for kindly inviting me in his group in ETHZ for three months. It was a short stay but it counted significantly in my thesis. It was besides the occasion to know interesting people as Julien and my table soccer mates Severin, Valeria, Gabriele and Stefano. There in Zurich, I was hosted by Agnes and Chris who, as weird as it may sound, made me discovered a bit about Brazil.

Then, outside the scope of work, I enjoyed the support of many friends, especially Mihai Şi Consuela Elena Jitia Sardarescu, Yohann and Dess who were often around and others like Aurelien and Matthieu who were far but still present.

I dedicate my thesis to my parents and my brothers and sisters, who did not stay left either. Although I could sense their dubious expressions on the seldom occasions when I described my work, they have always pushed me forward - especially my mother, who was torn between her conviction that a thesis could not lead me to safe ground and her desire to let me do as it pleased me. I must not forget aunt Colette as she insisted on reviewing my entire manuscript, while she could not understand English.

Last, my thoughts are with Pierre and Odile, who are like a second family to me, and particularly with their wonderful daughter Aline and the child she bears as a promise of amazing adventures.

Abstract

This thesis is devoted to 2D and 3D modeling of urban environments using structured representations and grammars. Our approach introduces a semantic representation for buildings that encodes expected architectural constraints and is able to derive complex instances using fairly simple grammars.

Furthermore, we propose two novel inference algorithms to parse images using such grammars. To this end, a steepest ascent hill climbing concept is considered to derive the grammar and the corresponding parameters from a single facade view. It combines the grammar constraints with the expected visual properties of the different architectural elements. Towards addressing more complex scenarios and incorporating 3D information, a second inference strategy based on evolutionary computational algorithms is adopted to optimize a two-component objective function introducing depth cues.

The proposed framework was evaluated qualitatively and quantitatively on a benchmark of annotated facades, demonstrating robustness to challenging situations. Substantial improvement due to the strong grammatical context was shown in comparison to the performance of the same appearance models coupled with local priors. Therefore, our approach provides powerful techniques in response to increasing demand on large scale 3D modeling of real environments through compact, structured and semantic representations, while opening new perspectives for image understanding.

keywords: *architectural image-based modeling, shape grammars, inference, appearance models, multi-view reconstruction*

Résumé

La présente thèse est consacrée à la modélisation 2D et 3D d'environnements urbains à l'aide de représentations structurées et de grammaires de formes. Notre approche consiste à introduire une représentation sémantique de bâtiments, qui encode les contraintes architecturales attendues, et qui soit capable de traiter des exemples complexes en utilisant des grammaires très simples.

En outre, nous proposons deux nouveaux algorithmes d'inférence permettant l'analyse grammaticale d'images en utilisant ces grammaires. En premier lieu, un algorithme dit de *hill climbing* permet d'extraire les règles de grammaire et les paramètres correspondants à partir d'une vue unique d'une façade. Ce concept combine astucieusement les contraintes grammaticales et les propriétés visuelles attendues pour les différents éléments architecturaux. Cependant, afin de pouvoir traiter de cas plus complexes et également d'incorporer de l'information 3D, une deuxième stratégie d'inférence basée sur des algorithmes évolutionnaires a été adoptée pour optimiser un fonction à deux objectifs qui introduit notamment des notions de profondeur.

Le système proposé a été évalué tant qualitativement que quantitativement sur un panel de façades de référence toute munies d'annotations, démontrant ainsi sa robustesse face à des situations d'abords difficiles. Grâce à la force du contexte grammatical, des améliorations substantielles ont été démontrées par rapport aux performances des mêmes modèles couplés à des a priori uniquement locaux. Par conséquent, notre approche fournit des outils puissants permettant de faire face à la demande croissante en modélisation 3D d'environnements réels à large échelle, grâce à des représentations sémantiques compactes et structurées. Ce travail ouvre par ailleurs un vaste champ de perspectives pour le domaine de l'interprétation d'images.

mots clés: *modélisation 3D d'architecture, grammaire de formes, inférence, modèles d'apparence, reconstruction multi-vues*

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

This thesis aims at addressing 3D city modeling that is to say modeling of existing large-scale urban environments. Despite important advances from industrial and scientific view point which to a certain extent address this demand, it is still considered a challenging problem. The effort required to provide full large scale reconstruction of a cities like Paris is significant since one has to address compactness, scalability and modularity.

In this introductory chapter, we will motivate the rational of our work and position the thesis in the general context of 3D modeling. This part will be tackled from the industrial point of view in section 1.1. In particular, we will explain the current status of city modeling, and the challenges it poses. Then, in section 1.2, the principal ideas developed in this work are expressed. Finally, after clarifying the context of the thesis, we will identify its main objectives and discuss how these objectives were addressed on the developed approach.

1.1 Industrial Context

1.1.1 Importance of Computer Graphics Imagery

As the name implies, urban modeling relates to 3D modeling and more generally to Computer Graphics Imagery (**CGI**). This recently established activity started from the work of visionary researchers and industrials in the sixties, until becoming ubiquitous in the industry. It includes more specifically video games, advertising and movies. In point of fact, the importance of **CGI** in these industries has become overwhelming.

Considering for example the movie industry, the use of computer animations started with few experimental films back in the early seventies, then a series of landmark films have marked the history of movie making with groundbreaking effects. One may think of the first episode of Star Wars prequel trilogy (1977) introducing 3D wire-frame animated graphics. In Tron (1982), 15 minutes of the film were fully generated on computers. Terminator 2: Judgment Day (1991) introduced the first humanoïd **CGI** character with realistic movements. The dinosaurs of Jurassic Park (1993) have definitely left us a deep and long-run impression and they have proved the incomparable power of

CGI for virtual effects. In 1995, Walt Disney Pictures presented Toy Story, the first feature film entirely realized on computers. In 2009, James Cameron produced Avatar, the first movie with a fully computer generated photo-realistic 3D world: the planet Pandora.

In brief, we have witnessed computer animated sequences turn from experimental to common practice in nowadays movies. A similar statement holds for 3D modeling, which has become overwhelming in many sectors such as car design, urban planning or military simulations.

1.1.2 Current Status of City Modeling

City modeling was not left behind. For instance, it has created a real enthusiasm in the video game industry. Thanks to progress made on the modeling side, as well as increasing computational power, bigger and more complex urban scenes were created, immersing gamers into virtual worlds of extreme realism. This immersion was strengthen by the capability to perform actions that would be banned and dangerous in real life. One of the most impressive examples is certainly the Grand Theft Auto series, which combines almost completely unrestricted behaviors within virtual cities inspired from existing United States cities like New-York and Miami. One would expect the effect of evolving in a faithful reconstitution of a real city to have even deeper impact on the gamers. But all in all, if virtual cities have required sustainable modeling costs, the situation is more critical for real city modeling.

In any case, the modeling of existing cities concerns other applications, such as navigation systems. In this context, virtual cities make no sense. Yet, because it is an extremely challenging task or perhaps because it did not procure the same marketable guarantees in the first place, the relative technologies are still at their infancy. Although few entire city models were created worldwide, in a vast majority of cases the model consists of low-detailed buildings with lack of semantics. In clear, each building is represented by a coarse volume associated with texture mapping. This certainly gives a nice impression for fly-over kind of visit, but it is inadequate for an immersive walk-through, while requiring at the same time significant storage and memory capacities. On the other hand, some of the city models or at least some of their landmarks are highly detailed. But this is again achieved at the price of heavy human intervention.

1.1.3 The Time has Come for 3D Large Scale Modeling

Even though the previous statement sounds as a failure for city modeling, many aspects makes it ready for reaching maturity. Some reasons are purely hardware related. For instance, the computational power of PCs has improved significantly with the generalization of multi-core CPUs and high performance GPUs. This power is even more amplified by cloud computing techniques. What is even more fundamental for the success of city modeling, networks transmission bandwidths have literally exploded. This makes possible for very large scenes to be rendered on the server side while frames are simply streamed in real time to the client.

Again related to hardware, but from a media point of view, recent devices have emerged and provide greater sensations. Among them, Cave Automatic Virtual Environments (**CAVE**) use a

system of multiple projectors within a dedicated room to give breathtaking virtual visits. Even though less sensational, 3D televisions which have recently made their apparition, might reach soon a broad public.

New softwares are available as well. For instance, open and commercial Computer Assisted Design are now accessible and easy to use. Some of them as Sketchup implement novel ideas around the rapid sketching of 3D objects, making modeling within the grasp of non expert users. Besides, 3D content can be easily shared or distributed on 3D warehouses. As an example, it becomes easier to obtain models of street furniture or of architectural elements. Added to the success of collaborative project philosophy, this could give a serious boost to city modeling.

More importantly, many services have emerged which reserve an ideal context for urban modeling. On the one hand, data collection has greatly advanced. Indeed, with its Street View application, Google has proved that it is possible to capture data from an entire city. These facts might be balanced by the unordinary power of this company. In any case, another type of data collection has raised and it is accessible[1] to anyone at low cost. We refer to Internet user collections such as Flickr which consist of billions of photographs shared on-line. The fact that geo-localization meta-data are often embedded in the pictures, also facilitates the process. On the other hand, on-line services hosting Geographic Information Systems (**GIS**) provide a perfect frame for accessing digital city models. One can cite the Google Map / Google Earth pair or Microsoft Bing Maps. Nevertheless, these proprietary suppliers are challenged by open alternatives such as OpenStreetMap.

1.1.4 Future Challenges

If context is auspicious, one might wonder what challenges must be overcome for city modeling to take off. From a high level perspective the answer is merely scalability. But from a closer look, this apparently simple response is in fact multi-sided.

Data requirement

The process of modeling existing environments will necessary rely on some observations. Many different data can be useful. The most common are range map obtained from laser scans, ground-based photographs, aerial or satellite images and cadastrial maps. They do not share the same attributes and characteristics in terms of scalability. We have actually ranked them according to the acquisition cost. Nonetheless, it might be necessary to re-assess the respective ranks of ground-based and aerial images to account for the previously mentioned Web user collections.

Automation

Data being available, the process, or at least the most tedious part of the pipeline, must be automated. The difficulty of this goal rests on the aimed accuracy and faith of the model with respect to the true objects. For instance, coarse volumes corresponding to the main facades and the roof can

[1]— Although copyrights can pose delicate issues.

be extracted from cadastrial inputs along with aerial images thanks to rather simple photogrammetry techniques. The problem gets more complicated when aiming at recovering detailed structural elements such as windows and facade decorations.

Compactness

In addition to the cost due to the required human intervention, large-scale modeling requires that virtual walk-through can be processed efficiently. This becomes more challenging when one considers that eventually the services will be provided through a remote server. It will be therefore important to reduce loading times and to increase frame-rates.

As a result, compactness is a fundamental property. More precisely, it is crucial to find a balance between the accuracy of the model and its size. Going forward in this direction, the use of different Levels Of Detail (**LOD**) makes such sense that it will certainly be an inescapable key to success. In that case buildings and other urban objects will be available at different degrees of precision. During a walk-through, remote buildings can then be rendered with coarse representations while the ones close-by requires finer details.

Understanding

Looking at existing **GIS** applications, many additional services propose advanced interactions with the core data. For instance, it is possible to display real time traffic information, or consult consumer feedback about a restaurant. Likewise, one can expect that city model application evolve towards embedding high level information, authorizing finer interactions with the model. The model could integrate semantic descriptions of the parts making the model. If we know where the windows in the building are, we can easily improve night time rendering by adding light sources coming out of certain flats. Alternately, if the model is used as a video game scene, different physical behavior can be attached to the semantics. For example, windows can be broken.

1.2 Thesis Statement

As often, this new industrial interest has been mingled with active research. Two branches of computer science have dedicated substantial efforts in this direction: computer vision and computer graphics. The present doctoral work lies at their common boundaries.

1.2.1 Accurate Context

Various tasks must be tackled to reconstitute a satisfying 3D model of a city. Globally, these tasks can be decomposed into recovering the terrain geometry, the street networks, street furniture, and the individual building models. This work is directed entirely towards building reconstruction. To be more precise, we will deal with the modeling of buildings observed through photographs taken from the street level.

As such, this doctoral work tackles a problem at the core of computer vision. In that context, this problem has been solved using a variety of approaches related to photogrammetry and stereo vision. The strong particularity of this study with comparison to these latter relies on the objectives we pursue and on the inherent representation we will use in order to achieve them.

1.2.2 Goals and Consequences

General objectives

The broad purpose of this thesis is to provide a method solving image-based building modeling in a suitable way for large-scale urban modeling. This objective concerns not only technical constraints as imposed by hardware performance, but also quality expectations. In simple words, the building models should be realistic and good looking. It goes without saying as well that a person living in the neighborhood of a reconstructed building should be able to recognize it from the 3D model.

The aforementioned challenges of city modeling must be presented with much attention. In short, the reconstruction framework should ensure compact models, possibly organized in **LODs**, and integrating symbolic information.

An adapted representation for buildings

The question of the representation/parametrization is a fundamental step in all scientific domains. Very spectacular progress often rests on the capability of a representation to handle meaningful concepts. The most instructive example of such reliance is probably the number of breakthroughs opened by quantum mechanics during the past century.

Computer vision does not stand out as an exception. In their inspiring book, Dana H. Ballard and Christopher M. Brown [Ballard 1982] give a very sensible definition of computer vision which goes in this sense. They say, “Computer vision is the construction of explicit, meaningful descriptions of physical objects from images”. Their book dedicates largely to the advocacy of model based interpretation. So does this doctoral work.

If the problem under consideration is tied by history to computer vision, we have opted for a representation inspired by works done in computer graphics. It relies on the shape grammar paradigm and the procedural modeling ideas that have followed.

Given the visually perfect models provided by this type of approaches, the chosen representation shows great promises. Besides, it does not disregard the other objectives either. Indeed, grammars provide coarse-to-fine models naturally organized in a hierarchy. A sensitive control of the complexity of the model is by there ensured. This hierarchical decomposition warrants a certain level of understanding as well. This will be all the more strengthen as soon as the atomic parts carry strong architectural significance.

Affordable automation

Automation is an important aspect of our objectives as it is compulsory to reach scalability. However, in practice, an automated process will not be economically more relevant than an interactive one if it requires much time and computing resources. Although a general criteria can hardly be assessed about that, performance remains an important consideration in this work. And, in a matter of fact, inference within the scope of highly connected hierarchical structures often poses sophisticated optimization problems.

1.2.3 Organization of the Thesis

In the remainder of the thesis, we will first draw in chapter 2 a broad overview of works related to image-based reconstruction. A particularity of this study is that it places the representation at the center of the considerations. Then, in chapter 3, we will present a modular procedural modeling framework, we have developed. This chapter will also include a short historical review of shape grammars and their recent time of renown in computer graphics. The two last chapters are dedicated to the actual formulation of image-based procedural reconstruction approaches. In chapter 4, we consider the problem of recovering the underlying structure of a facade from a single image. It introduces the link between the semantic decomposition derived from procedural models and local appearance models obtained via supervised learning. The potentials of this association is demonstrated thanks to the use of a simple stochastic search optimization: the hill climbing algorithm. In the prospect of extending the previous approach to full 3D modeling, chapter 5 considers the generalization of hill climbing to other meta-heuristics. After showing substantial gains with respect to efficiency, the chapter tackles the case of multi-view procedural reconstruction. Eventually, chapter 6 concludes this manuscript by summarizing the contributions and opening with a discussion about yet undeveloped perspectives.

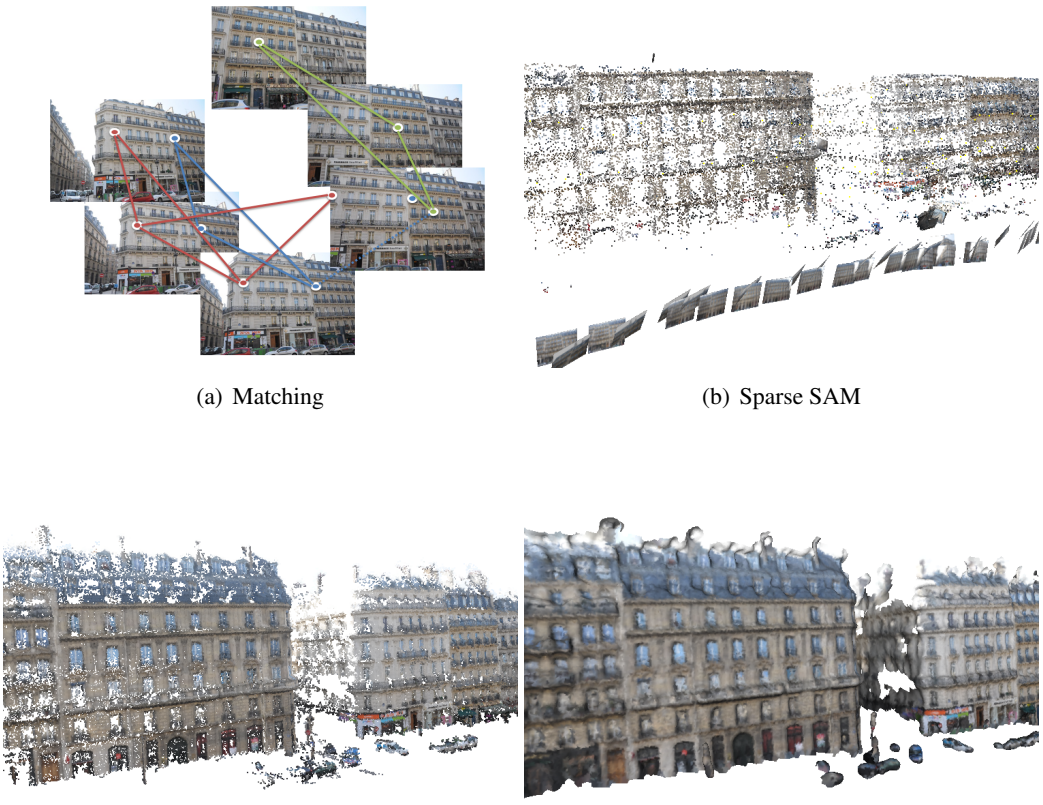
Chapter 2

A Survey of Model-Based Reconstruction: Towards Structure Discovering

2.1 Introduction

Image-based 3D modeling has been a fundamental branch of computer vision for the past three decades. It was initiated in the robotics field where a camera system mounted on a robot was supposed to grant the automaton with true depth perception over its environment and thus to facilitate interaction. Effort in this direction (that is still on going) gave also birth to a new field where the main considerations were not anymore related to direct interaction but rather to the accurate capture of the geometry of a scene. This field is commonly known as *shape from X*. Behind this generic denomination, a huge variety of techniques related to different areas from medical imagery to outdoor scene reconstruction do co-exist. The latter topic is the main interest of this thesis. More specifically, multi-view reconstruction towards digital 3D representations of urban environments will be considered. In this case, the implicit objective of challenging active scanning devices, such as LIDAR, is often mentioned. Stakes are both practical, and economical. Indeed, LIDAR scanning is costly and requires expert manipulations. This is all the more valid since the spreading of Internet photograph query systems like those provided by Flickr or Google, as virtually an image-based system could work without any need of capture session [Snavely 2007].

Many steps and advances were made in this domain related first to 3D projective geometry theory in general including diverse aspects as epipolar geometry [Faugeras 1992, Hartley 1992], self-calibration techniques [Pollefeys 1999] and bundle adjustment [Triggs 2000]. Then to more efficient methods were proposed like sparse bundle adjustment [Lourakis 2004, Lourakis 2010] or fast Approximate Nearest Neighbor (**ANN**) [Arya 1998]. From this whole material came out a general pattern allowing for fully automatic systems like [Snavely 2006a, Snavely 2006b] or [Vergauwen 2006a, Vergauwen 2006b]. The interested reader can refer to [Hartley 2004] for a



(a) Matching (b) Sparse SAM

(c) Dense SFM (d) Alternate representation

Figure 2.1: The general scheme of automatic reconstruction. First, (a) stable key-points are detected and filtered to form tracks. Then, (b) calibration is computed for each view as well as 3D location for each track. In (c) the point cloud is refined using SFM. Eventually a more structured representation can be recovered: in the example (d), a mesh.

recent book on this topic. The general scheme is described in Figure 2.1. First, stable key-points are extracted in each image of the sequence that are then fed to a matching step providing sparse 2D correspondences. The set of correspondences can be further filtered to dismiss outliers by considering geometric (epipolar) constraints. Based on the output of this step, a non-linear system of

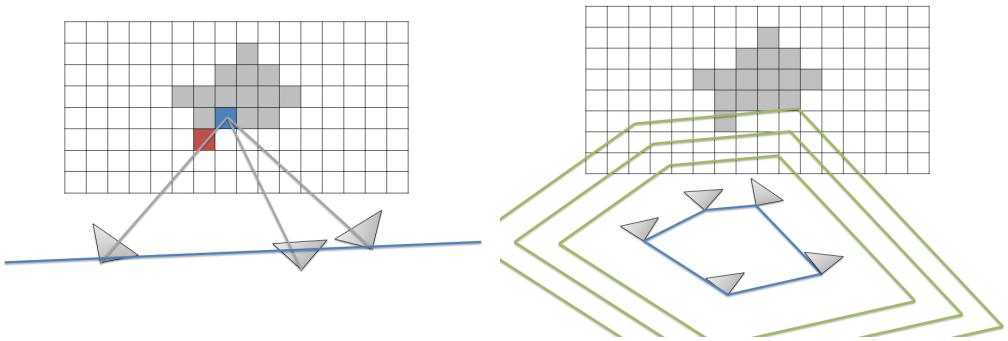
equations involving both camera parameters and 3D locations of the tracked points must be solved. Then, a refinement of the previously mentioned parameters, as well as the camera model (in order to take into account radial distortion) is possible through bundle adjustment. Such a process leads to the estimation of the camera intrinsic and extrinsic parameters along with a sparse 3D point cloud. The procedure is often called sparse Structure And Motion (**SAM**), where structure refers to the point cloud and motion to the position of the cameras. Ultimately, a complementary process is necessary to reach the desired density of the digital representation. In what follows, this step will be referred to as dense Structure From Motion (**SFM**) because, given the the sparse data constraints, it aims at getting the scene representation denser. The final output is therefore a denser point cloud or, as discussed later on, a more structured representation.

The literature of 3D reconstruction is rich and constantly evolving. Existing methods differ on several crucial aspects. It is therefore quite complicated to draw an exhaustive overview of this major branch of Computer Vision. Nonetheless few such attempts [Dyer 2001, Slabaugh 2001, Seitz 2006] were successful enough to be acknowledged. Among them, [Seitz 2006], both targeted to establish a taxonomy of multi-view reconstruction and to propose a benchmark for evaluating reconstruction approaches. While the latter objective turned out of rare practical value to emancipate quantitative comparisons between state of the art methodologies, the former brought a clear and structured overview of existing approaches, still of relevance today. The separation between the considered works is based on diverse criteria, namely the photo-consistency measure, visibility considerations, shape prior, reconstruction algorithm and also the scene representation.

The current chapter is mainly inspired from the last angle of differentiation. Indeed the work in this thesis is strongly related to the structure and the semantics embedded in the representation. It is therefore important to have a clear overview of the past and more recent efforts made in this context. The considered representations are clustered starting from the less to the more structured ones. Although the present chapter focuses on structured representations, to better explain their importance, scattered representations such as point clouds and meshes will be also presented (section 2.2). Then representations based on uncorrelated primitives will be discussed in section 2.3 and followed, in section 2.4, by methods introducing higher order relations between the primitives.

2.2 Scattered Representations

Herein, the word scattered is used mainly to refer to the disorganized aspect of the concerned representations. This lack of structure is sometimes stressed in the literature by using the term soups (of points or of triangles in the case of meshes). Albeit in the current discussion we advocate strongly in favor of structure, the reader should bear in mind that several advantages come with these simpler representations. As a matter of fact, less order implies no or few interactions between parts of the shape. Inference is therefore simplified due to independence properties. This is certainly an advantage when efficiency and accuracy are at stake. Besides it should be noted that this type of representations rely on fewer assumptions about the expected nature of the scene. Therefore, low-structured representations could play a complementary role (such as a robust initialization)



(a) Ordinal visibility constraint (b) Topological order

Figure 2.2: Voxel coloring. (a) The blue voxel is occluded by the red in the left image. This image should not be taken into account when deciding upon photo-consistency of the blue voxel. (b) If the ordinal visibility constraint is checked, a topological order can be established.

with respect to the more sophisticated and specialized ones.

2.2.1 Volumetric Representations

Unlike upcoming classes of representations, volumetric approaches represent a shape as a closed volume and not directly as a surface. This distinction is arguable for Level Sets but they are classified here because of their inherent representation which can be interpreted as a volumetric distribution of probabilities to belong to the inner side of the object. Moreover, tightly related to this volumetric way of thinking, all the methods described here rely on a regular 3D discretization of the space. As noted in [Slabaugh 2001], the term volumetric in this context, which comes from the computer graphics nomenclature, is not generalized in computer vision.

Such methods have introduced the concept of photo-consistency as an answer to shape from silhouette. While the later were used to determine the so called visual-hull of the objects, volumetric approaches lead to the concept of photo-hull, defined as the largest photo-consistent reconstruction. Theoretically, the photo-hull was expected to resolve the visual-hull's inability to model concave regions. Although, as shown in [Dyer 2001], this often proves wrong as soon as one deals with textureless regions.

The most naive representation is based on occupancy functions. In the simplest expression, they attached to each voxel of the space a binary value. This value is merely set to one if the voxel is believed to be occupied by the object. They were introduced by [Seitz 1997], in the voxel coloring framework which adopts a simple principle. Every voxel is considered sequentially and checked for photo-consistency. This boils down to compute a similarity measure between the

different colors obtained by projecting the considered voxel onto the input images. A voxel is labeled as occupied if the output is above a given threshold. As often with multi-view approaches, the problem of visibility reasoning arises. Within the scope of the previous work, an elegant answer was proposed, under the assumption that the whole bounding volume of the scene stands outside the convex hull of the camera centers. This hypothesis is known as the ordinal visibility constraint. It enforces that voxels are visited in such an order that every potential occluder to a given voxel has been treated earlier. In theory, voxels should be considered in layers of increasing distance to the convex hull of camera centers (see Figure 2.2 (b)). In practice, the ordinal visibility constraint is often simplified by requiring that the cameras and the scene stand on either side of a plane (as illustrated in Figure 2.2 (a)), and an elementary plane sweeping strategy guarantees visibility to be determined exactly at each location in a single pass.

In spite of its ingeniousness, practically speaking, the ordinal visibility constraint forbids the scene to be captured from every angle. Consequently, voxel coloring is bound to provide partial reconstructions. This issue has been however tackled later by relaxing the constraint on the camera placement at the cost of multiple plane sweeping passes. The resulting algorithm was defined in [Kutulakos 1998, Kutulakos 2000] under the name of space carving.

Compared to shape from silhouette approaches, the use of photo-consistency has been a promising step forward. Reconstruction accuracy was improved drastically while eliminating the need for silhouette extraction. Nonetheless, thresholding produces false positives and false negatives and it can often lead to flattening effects or unreal bumps and cavities. For this very reason, the reconstruction task could benefit from a more global formulation driven from an energy minimization. Using similar occupancy functions, but representing them as a Markov Random Field (**MRF**), [Vogiatzis 2005] demonstrated the strength of such approach. Global optimization can address the shortcoming inherited from local decisions (like thresholding) and allows to embed a regularization term, improving the visual aspects of the output. Besides, in a **MRF** with binary variables, the energy can be efficiently and globally optimized using the min-cut algorithm. The downside of this approach, however, is that only pairwise interaction between voxels can be modeled, which is not compatible with proper visibility reasoning. In [Vogiatzis 2005], visibility is only approximated using a visual-hull which was considered an inefficient approximation and replaced in [Vogiatzis 2007] with an implicit model. Inspired by [Hernandez Esteban 2004], this was achieved by modifying the photo-consistency measure in order to account for possible outliers.

An alternative way to formulate the reconstruction in an optimization framework was suggested by [Faugeras 1998]. The authors put forward the idea that **SFM** can be described in terms of variational principles. By deriving the associated Euler-Lagrange equations, an initial surface can be deformed through a combination of internal and external forces, towards the true boundary of the object. In practice the resolution of the Partial Derivative Equations (**PDE**'s) was carried out thanks to the level sets method. The clear advantage over **MRF** solutions is that visibility can be tackled explicitly without restrictions. The impressive flexibility of level sets approach has been demonstrated in many works. In particular, in [Pons 2007b], a global measure of image similarity is defined which alleviates many common simplifications as the Lambertian surface assumption. On the other hand, level sets are often criticized for their computational complexity. Nonetheless,

the most important ideas are related to the variational principles, and as discussed later on, found their path through meshes [Pons 2007a], hence eliminating the computational burden.

Limitations

Despite substantial evidence of the robustness demonstrated by volumetric representations, they suffer from some serious limitations when it comes to outdoor scenes. The main reasons are that they require knowledge of the scene bounding box and the definition of a discretization beforehand. Their use should therefore remain limited to reconstruction of isolated objects.

2.2.2 Point Clouds

Both of the aforementioned criticisms can be addressed by directly modeling the scene as a collection of points without resorting to a regular grid. Point clouds are the simplest and less structured among such representations. One should maybe not classify them amongst model-based representations, as in the vast majority of works they derive from a direct image-based pass. The most common cases are the multiple depth maps representations [Kolmogorov 2002, Gargallo 2005, Vergauwen 2006b, Gallup 2007, Gallup 2008] and quasi dense feature matching [Lhuillier 2005, Furukawa 2007].

Principles used in two-view stereo reconstruction, were amended to dense multi-view reconstruction by computing multiple depth maps. The more important difficulty lies in the necessity to maintain consistency between the different scene interpretations provided by each depth map. This task can be performed by merging the separate depth maps in a post-processing step. [Narayanan 1998] applied a simple sub-optimal strategy where the different depth maps are considered in order to fix holes detected in the current reconstruction.

One can improve performance by integrating constraints during the depth maps calculation. For example in [Gargallo 2005], both depths maps and color maps are inferred using a global Bayesian framework, where consistency between different depth maps is enforced using a multiple depth map prior. This prior is actually formulated as a Markov network over the point cloud. Similarly, in [Kolmogorov 2002], a discrete **MRF** formulation ensures the consistency of the recovered 3D points through visibility reasoning. Both methods adopt reduced connectivity to achieve practical performance. This is ensured by defining a neighborhood of interaction, which even though successful is an arguable choice.

In [Gallup 2008] the authors tackle an inherent issue of multi-view stereo vision, that is to say the compromise between wide and narrow baselines. It is commonly admitted that narrow baselines (*e.g.* consecutive frames of a video) suit better to the matching task since smaller search ranges are involved and therefore ambiguity is decreased. However, in such a context, one should address a fundamental issue of matching, that is endowed with projective distortions which discard similarity measures based on rectangular window. There again narrow baselines minimize the difficulty. On the contrary, when triangulation is concerned, wide baselines introduce less uncertainty. The mentioned article pleads for an automatic selection of adapted baseline depending on the depth

under consideration. This choice is theoretically justified if one seeks a uniform accuracy within the reconstructed scene. Another pertinent suggestion to improve matching performance in the wide baseline configuration consists in using more advanced descriptors than rectangular patches. Until recently, this option was limited to sparse matching because of the computational load. However, the development of efficient and compact descriptors [Strecha 2010a, Tola 2010] made their use possible for dense depth maps as well.

The class of methods relies on feature tracking and is, as a result, somehow related to what is done in sparse **SAM** system. The main difference with depth maps approaches lies in the fact that matching is not sought for every pixel systematically but only for some detected feature points and eventually propagated to the remaining ones.

The strongest argument in favor of this kind of methods was presented by [Lhuillier 2002]. The authors noted that given a sparse set of high confidence correspondences, the task of finding good correspondences for pixels in the vicinity of already matched ones is easier. In fact, the two principal sources of ambiguities in these **SFM** are the presence of repetitive patterns and of low textured areas. They are both alleviated by the gradual diffusion of matching because the search is performed locally around the seed match. This statement leans however on the presumably robustness of the seed correspondences and on the assumption of piecewise smoothness of natural scenes. In [Lhuillier 2005], improvement not only with respect to the reconstruction but also to the camera motion estimator, was demonstrated based on the same idea. A similar approach was considered in [Furukawa 2007], where the initial correspondences are enforced to be evenly spaced. This was achieved by detecting a given number of features in each cell of a regular grid in each image. Besides, the cloud is composed of small patches instead of mere points. In that way, a surface element is defined at each vertex, which is a nice feature not only for optional post-processing but also during the reconstruction process itself as it facilitates visibility reasoning. Eventually, the expanding step is followed by a filtering procedure focusing on removing patches lying outside the actual surface. Expansion and filtering are performed multiple times in order to improve the occlusion clues. As noted by the authors, the combined use of local expansion and visibility filtering enforces smoothness in the reconstruction step to circumvent the need of further regularization. Associated to this work, an open source software is delivered on the web [Furukawa 2008] demonstrating the effectiveness of the approach.

With respect to scalability aspects again, in their famous article entitled “Building Rome in a Day” [Agarwal 2009], Agarwal *et al.* have designed a system taking advantage of parallelism at each stage of the reconstruction pipeline that was capable of reconstructing point clouds of entire cities within 24 hours. The computations are driven on a cluster of 500 core processors and involve up to 150K images.

Limitations

Given the impressive results reported by [Seitz 2006] and [Strecha 2008] both in terms of completeness and accuracy, point clouds representations can solely be criticized with respect to qualitative aspects. The most critical of which resides obviously in the lack of connexion between the recov-

ered points. As such, the model is not a true surface and, to some extent, suffers from the same drawbacks as the outcome of a sparse **SAM**. Nonetheless, we will see in the next section that dense point clouds can be turned into connected representations fairly easily.

2.2.3 A Natural Turn towards Meshes

Being the prevailing representation for rendering, meshes constitute a major candidate for **SFM** too. They were used in other connected fields as 3D segmentation [Slabaugh 2005]. The fundamental ideas remain very close to those presented in the level set approaches. A surface is evolving under the action of different forces towards satisfying a variational principle. The only difference is that the representation is here an explicit triangulated surface rather than implicitly specified as the zero level-set of a function. The benefits are three-fold. First, the outcome of the algorithm can be easily visualized. Besides, by integrating forces computation over entire faces, and ensuring a minimum area for each face, a certain robustness to noise can be reached. Last, computational load is greatly reduced thanks to a good behavior under larger time steps. This speed-up can also be strengthen since it is possible to perform massively parallel computations on graphics hardware, thanks to the use of shaders. We will narrow our analysis to few contributions that have sensible connection with the current discussion.

The forces involved in the deformation of the surface divide into two classes, internal forces (for regularization), and the external ones, tying the model to the observations. In order to compute them, one needs to be able to determine the gradient of the associated energy. A commonly considered external energy is the reprojection error. It evaluates the difference between a color predicted by the model at a specified 3D location and its projections in the observed images. However, occlusions make the exact gradient computation challenging and were long ignored [De La Gorce 2008]. To the best of our knowledge, a closed form expression of the gradient was first provided in [Delaunoy 2008]. In addition to the sound theoretical resolution of the problem, the correct handling of visibility plays a crucial role as it enforces the appearance contours of the mesh to project at high gradient location in the observed images. It therefore has an analogous role as silhouettes in the visual-hull reconstruction except that no explicit silhouette extraction is required. Furthermore, it eliminates the need to introduce artificial shrinking forces that were commonly adopted. It is worth mentioning that, the authors of [Delaunoy 2008] additionally propose a simple alternative to the commonly consented Lambertian surface assumption. To this end, they enrich the constant color model defined at the level of the mesh with corrective components defined in the images respective domains.

If computationally speaking, meshes are generally acknowledged to outperform implicit implementations, the later have a clear advantage over the former when it comes to topological evolution. In all likelihood, depending on the initialization the evolving mesh will have to adapt its connectivity in order to closely fit the observations. In the absence of necessary topology adaptations, the mesh would certainly be driven by the external forces towards non admissible configurations, notably when faces are intersecting each other.

A first way to handle this issue, is to detect explicitly degenerate configurations and fix them by

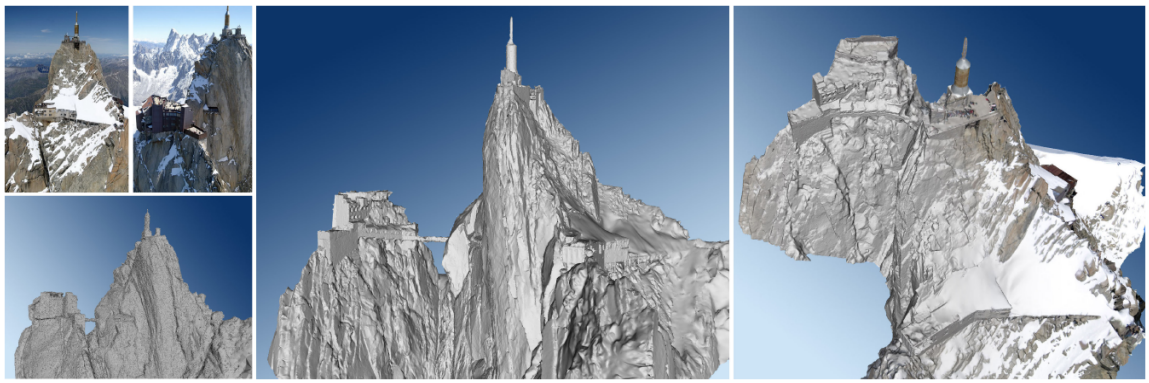


Figure 2.3: The combination of a point-cloud reconstruction and variational refinement of an extracted mesh can lead to extremely impressive reconstructions of large-scale outdoor scene. Image courtesy of [Hiep 2009].

adapting the mesh connectivity. This was first explored in [Duan 2004] where both a merging and a splitting procedure could be triggered whenever two vertices came into a distance shorter than a threshold. Similar ideas have been expressed in the later literature, notably in [Zaharescu 2010], where a complete set of topological operators is proposed. The framework called TransforMesh uses merges, splits, hole formations, and hole losses in order to cope with self-intersections that might appear during the optimization.

An alternative approach consists in handling the topology issues implicitly without detecting undesirable arrangements. For instance, in [Pons 2007a], restricted 3D Delaunay triangulations are used successfully in this purpose.

Initialization is a critical component of variational approaches. This type of methods are not designed to reach a global optimum. Quite often, the expected solution is not itself the global minimum of the energy but a particular local one. Most works cited previously adopt a visual-hull initialization. Visual-hulls unfortunately cannot be computed for outdoor scenes. A preferable option is to use a point cloud reconstruction and turn it into a mesh. For instance, in [Furukawa 2007] a process to turn a patch-based reconstruction into a mesh is presented. This process consists of a classical Poisson reconstruction which is also a variational method, where external forces push the vertices of the mesh towards the patches. Then the mesh is refined through another variational principle where external forces drop the influence of the patch-based reconstruction in favor of a pure photo-consistency energy. The same scheme was also followed in [Hiep 2009], with a different approach for the initialization. In this work, the Delaunay triangulation of the point cloud is extracted, and each tetrahedron is allowed to be labeled as belonging either to the interior or to the exterior of the scene. An energy is defined to guarantee a visibility consistency and a mesh quality criteria. The labeling problem can be solved globally using the min-cut algorithm. The potential of this combination is illustrated in Figure 2.3.

2.2.4 Summary

In this section, three classes of scattered representations have been reviewed. Although volumetric representations brought new ideas, they are doomed by the space discretization requirement. This is a major concern when aiming at outdoors reconstructions. Among them some methods (**MRF**) are particularly robust and others (level sets) are flexible and particularly powerful when well initialized. Point clouds overcome the principal default of volumetric representations and appear both robust and accurate. Their inherent lack of connectivity is a critical downside especially with respect to rendering. Eventually, meshes provide an interesting alternative to point clouds. By adapting variational ideas, efficient reconstruction frameworks can tackle large-scale scene modeling. Nonetheless, they are sensitive to the initial surface guess. For that reason, a point cloud initialization with a mesh variational refinement provides a very powerful combination.

One can conclude that scattered representations contributed to the success of automatic 3D reconstruction. Such statement might be moderated, as there is yet room for improvement as regards accuracy, especially if one seeks to really catch up with LIDAR performance. However in this thesis, the imperfections to be discussed are of a very different kind. In fact, the actual concerns debated thereafter are, on the one hand, compactness and, on the other hand, semantics and structure understanding. They will be partly covered in the next section where we investigate the benefits of involving primitive shapes.

2.3 Compositional Modeling: Primitives in the Loop

The interest of introducing primitives in a representation relates to the pursuit of compactness. Primitives can be represented with few parameters (*e.g.* the radius and the height for a cylinder), and their rendering can be achieved with minimal tessellations towards a reliable reproduction of the primitive geometry. Besides, primitives are sometimes embedded with a symbolic description, in such way that the reconstruction explains the scene at both the geometric and semantic level.

2.3.1 Piecewise Planar Reconstructions

Aspiring for compression, it is straightforward to notice that human-made objects, especially buildings, are often composed of planar regions. As such, modeling objects as collections of planes is a natural direction. Such modeling is very different in nature from meshes, as first the planar regions are expected to span large areas, while methods based on meshes often assume a high density of triangles. Secondly, the constraints (verticality, pairwise perpendicularity) over the planes, if any, are of higher level than those involved between the faces of a mesh. Consequently, planes can be considered as elementary primitives to be composed towards modeling a complete scene. Therefore the distinction between this class of representations and meshes is rather conceptual. Nevertheless, when considering the resulting meshes (cf. Figure 2.4), the benefit of plane-based modeling is very concrete. It indeed logically contributes to a somehow logical removal of unrealistic cluttered regions, leading to neater models.

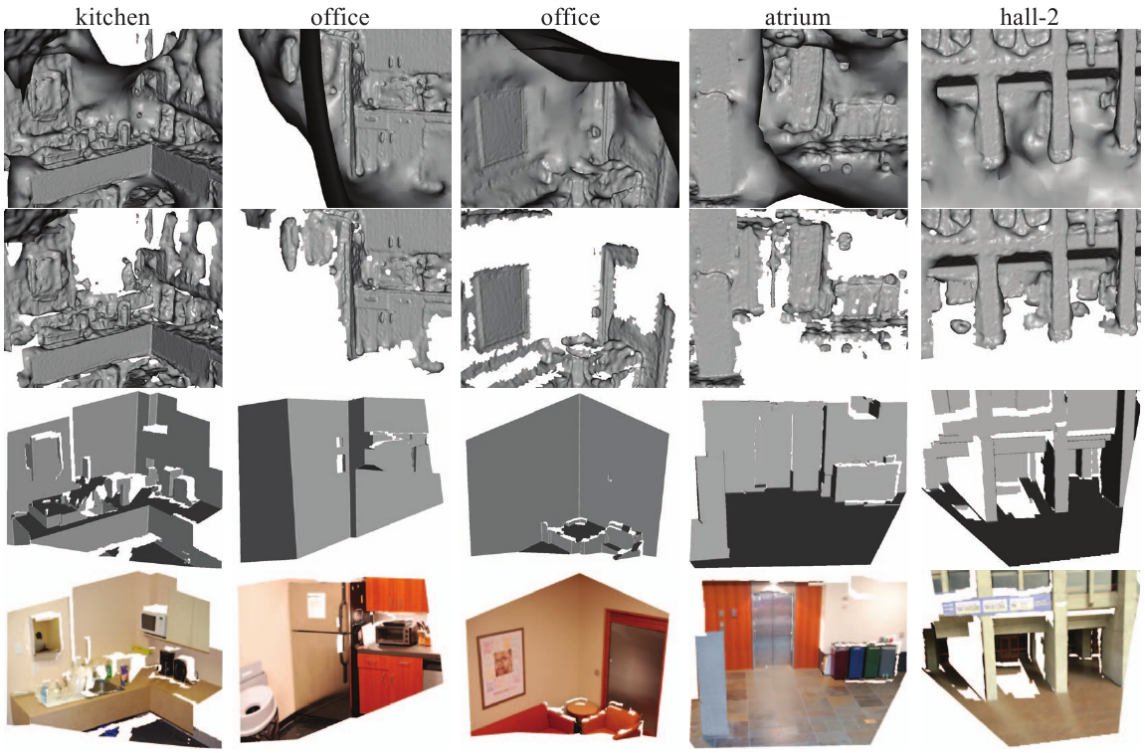


Figure 2.4: Examples of man-made interior scenes. The first two columns correspond to two approaches derived from [Furukawa 2007]. The last two columns correspond to a Manhattan world model [Furukawa 2009]. Although results of [Furukawa 2007] were acknowledged as very flexible and accurate, the under-constrained aspect inherent to scattered representation makes the output mesh cluttered while the scene can be recovered quite accurately using a combination of large and pairwise perpendicular planes. Image courtesy of [Furukawa 2009].

In 1999, Baillard and Zisserman [Baillard 1999], proposed the following steps to reconstruct the piecewise planar model. First, 3D lines are reconstructed thanks to a line matching procedure. Each line generates a single parameter family of hypothetical half-planes (obtained by rotating around the line). Each half-plane induces in turn an homographic mapping between every input image retinal plane and its own coordinate system. A half-plane can therefore be tested for validity by mapping every image onto the plane and computing a similarity measure based on the different textures obtained in the neighborhood of the original 3D line. Subsequent steps permitted to delineate each half-plane in order to obtain clear transitions between the parts of the reconstructed model. This approach demonstrated promising results on sequences of aerial images. Indeed the coarse volume of the building, including an accurate model of its roof, could be recovered without any constraint on the shape of the footprint.

Werner *et al.* [Werner 2002], proposed an alternative to recover the main planes of a scene when ground view images are available. Two concurrent ways to discover planes were tested. First, based on a 3D line and a 3D point, a plane hypothesis can be formulated. Using RANSAC, valid planes are detected. The alternative is based on a plane sweep strategy. The results show reconstruction of the main vertical planes, of the roof and also the ground plane.

Very recent works including [Sinha 2009, Gallup 2010, Furukawa 2009] presented a common contribution. In these three works, an extensive collection of plane hypotheses is produced and the task of filtering wrong hypotheses is cast as a labeling problem expressed in an **MRF**. Efficient ways of solving the optimization problem being available, the decisions concerning certain planes are taken globally. For this reason, these three approaches produced very convincing reconstructions for man-made outdoor scenes. Besides this shared contribution, each work came with its own personal features. In particular, Furukawa *et al.* [Furukawa 2009], restricted the planes hypotheses to those aligned with three dominant perpendicular orientations, hence leading to a so-called Manhattan world. Gallup *et al.* [Gallup 2010] allowed an extra label corresponding to non planar regions, so that they could use an alternative reconstruction method. This extension will be discussed when dealing with hybrid representations.

A last work [Delong 2010] related to the preceding ones, introduced a Minimum Description Length (**MDL**) principle. Though this work targets model fitting in general, it could be successfully applied in the case of piecewise planar modeling. The **MDL** principle is concretely realized thanks to a label cost, penalizing the number of retained hypotheses. It turns out to induce a very strong prior on the part-based configuration. The authors even claim that it suffices to enforce regularity within the reconstruction, hence moderating the need for regularization terms.

Limitations

Natural failures are encountered whenever non planar (but possibly smooth) surfaces are present in the scene to reconstruct. A possible extension consists in adding other volumetric primitives, such as spheres and cylinders, to the modeling vocabulary. In addition, given the expected compression rate, it is not surprising that such reconstructions oversimplify the actual scenes. In consequence, it is frequently appropriate to break down the model locally. To account for these two obstacles, collections of more general primitives and Lego-kits will be introduced in the next section.

2.3.2 Collections and Lego-Kits

The introduction of new types of primitives was motivated by curved surfaces. Sphere and cylinders, for instance, are frequent in architectural structures. It can also be desirable to introduce primitives that can be decomposed into a set of planes for the mere reason that they correspond to high frequency configurations (Figure 2.5). This the case of parallelepipeds and prisms for example. In fact, extracting common configurations is certainly a key to achieve more compact representations. It is also vital to better discern which atomic (semantically speaking) shape constituents compose a scene. Such an approach could have a positive impact on both compression

Figure 2.5: Because of their high repetition rate in architecture, some common configurations of planes should be taken as primitives by themselves. The example (a photograph of the Campanile, Berkeley's clock tower) shows many instances of prisms and parallelepipeds. Image courtesy of [Debevec 1996].

and understanding.

The Facade project [Debevec 1996] somewhat pioneered research in this direction. This many-faceted work introduced the idea of modeling the scene as collection of such volumetric atomic shapes. The inference was guided through user interaction. More precisely, the interaction consists in marking edges on the images, as well as defining the list of primitive instances expected in the scene along with their spatial relationships. This provides a parametric template for the scene and image cues for where edges of the model are expected to project. Then, a minimization procedure adapts the model to the observation. Although the user is required to interact much more than in previously discussed systems, the high quality of the inferred models testified the potential of the approach. As a side note, the authors also show that systems based on constrained volumetric shapes, can be put to good use in order to upgrade a projective reconstruction to metric. This benefit was also claimed in [Wilczkowiak 2005], where only parallelepipeds were used.

The Lego-kit approach was developed towards addressing the second limitation of piecewise

planar representations. Generally, the main planes of a building correspond to its facades and roof sections. They usually exhibit a set of structuring elements, such as windows or dormers. Because these elements come out back and forth the corresponding plane, they create perceptible texture distortion, whenever they are not modeled properly. On that account, to advance further towards better realism, one should aim at recovering such details. Building upon this note, models can be defined as Lego-Kits where a coarse volumetric model is used as a support to plug or carve primitives. This was attempted for instance in [Werner 2002]. The primitives considered in this work correspond to windows and dormers which were modeled as template meshes. Candidate positions of a primitive can be detected using the plane sweep information. Indeed if an element bulges out of the facade, the similarity score used to fit the median plane will be locally better when applying a small deviation from this median position. Depending on the complexity of the primitive, line matching can also be used to fit the model of the primitive. In a series of articles, published in the early 2000's, Dick *et al.* [Dick 2001, Dick 2004] formulate a Bayesian model discriminating between wall and different kinds of primitives (doors, columns, windows, *etc.*). The discriminative power is obtained thanks to a learning phase concentrating on the appearance of the various involved primitives. Additionally, the primitives are designed in a parametric fashion and prior distributions are expressed over the involved parameters. One serious difficulty raises when fitting this generative model to a set of observed images, as indeed the dimensionality of the model is directly linked to the number of primitives in the instance. In such situation, classical approaches to inference do not apply. However, the reversible jump Markov Chain Monte Carlo (**rjMCMC**) allows to jump between different dimensions following a proposal distribution. This technique actually creates a sequence of candidate models asymptotically sampled from the posterior distribution. These samples can be sorted based on their posterior score. Instead of merely keeping the best of them, the authors propose to leave to a user the responsibility of choosing the best model among the top N models.

Limitations

Though aiming at filling the flaws of the piecewise planar representation, the modeling presented in this section still inherits the inability to accurately capture all scenes. To truly tackle this issue it might be necessary to resort to a combination of compact and scattered representation.

The lack of structure in the way that the primitives are arranged constitutes a major concern. This eminently difficult task was assigned to a human operator in [Debevec 1996]. The user had actually to enforce constraints (symmetry, repetition, continuity) between primitives. Nan *et al.* [Nan 2010], rely on the same strategy. In this work, the structure is imposed interactively by roughly initializing positions, and also defining groups of elements called smart boxes. Then an automatic optimization loop, refines the position of each smart box. This step is based on two forces imposing respectively the consistency with the input point cloud and a contextual prior which favors alignments and regular interleave for instance. Nonetheless various attempts were made to automate structure inference to some extents. It paved the way to a very rich area of research dealing with structural models. This topic will be covered in section 2.4.

2.3.3 Hybrid Representations

Prior art addressing compactness and accuracy in the same time is limited. This observation surely derives from the strong antithesis between both purposes. The existing attempts naturally mix a representation based on primitives for their great potential in summarizing scenes, and a scattered representation expected to increase accuracy where needed. The main question with no obvious answer is where the compact representation would be advantageously refined.

The plane plus parallax is an example of approaches that are in the line of refining a coarse model. The scene is modeled as a first approximation as a piecewise planar model, and then refined by allowing small deformations of the plane in the direction of its normal. As such, the scene is assumed to be composed of almost planar parts. The displacements are usually defined on regular grids, each of which is overlaid onto a given plane. Because the direction of displacement is orthogonal to the plane, it is referred to as a parallax. It was proposed in the case of a single plane reconstruction by Collins [Collins 1992] and was applied to the reconstruction of elevation map from aerial images. However, few scenes can be approximated efficiently with a unique plane. Therefore the disparity to the median plane must not be restricted in too small a range. Henceforth, the hybrid representation gets less appealing. Debevec *et al.* [Debevec 1996], however demonstrated the benefit of the parallax component when multiple planes are involved. The authors combined their user guided piecewise planar reconstruction with an automatic parallax estimation, arguing that such allocation of tasks was in line with the efficiency of each operator. Since then, above mentioned works have proved that the piecewise planar decomposition could be handled automatically with robustness.

It is important to note that the parallax component can be efficiently stored as a texture. This option turns out all the more handy when it comes to rendering the reconstructed scene. This representation, referred to as bump mapping or parallax mapping, depending on subtle distinctions, has drawn much interest in the graphics community. Its success relies on the availability of programmable functionality in the rendering pipeline. Technically, a shader allows to deform the texture encoding the appearance of each plane depending on the parallax field as well as the view point.

One can notice that the plane and parallax pattern literally skips the analysis of which parts need finer modeling. On the contrary, Gallup *et al.* [Gallup 2010], conceive an approach resting upon this exact angle. In their **MRF** formulation, an additional label is dedicated to point out regions believed to be non planar. Such regions are modeled using a disparity map, but this particular scattered representation could be replaced by earlier discussed representations.

Eventually, if one considers strength and limitations of existing methods, meshes and primitives lead to a powerful combination. Meshes can be considered as the highlight of scattered representations while primitive collections are more advanced in terms of semantics as well as compression than the piecewise planar representation.

In the context of 3D reconstruction, this type of mixtures has been studied in [Lafarge 2009, Lafarge 2010]. Their general idea is to start from a mesh, generated by one of the previously described variational approaches, and to try and segment it into clusters corresponding to the fol-

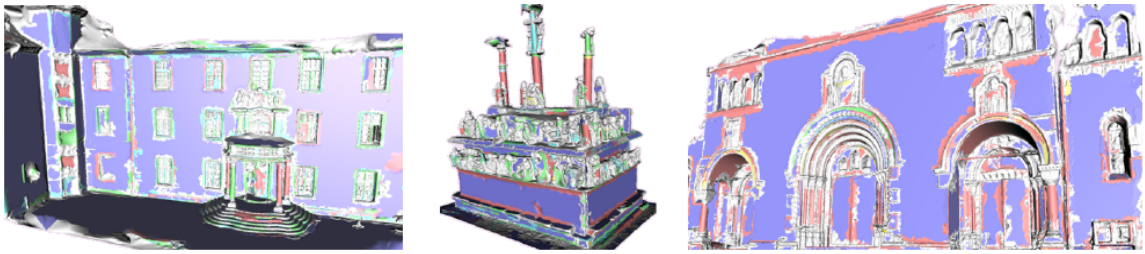


Figure 2.6: Reconstruction of different challenging scenes, based on the hybrid representation developed in [Lafarge 2010] (The following color code was used: purple=plane, pink=cylinder, blue=cone, yellow=sphere, green=torus, gray=mesh). Despite undeniable false positive and false negative cases with respect to the primitive detection, the results are quite promising. Image courtesy of [Lafarge 2010].

lowing primitives: plane, cylinder, cones, sphere and torus. Additionally, parts corresponding to none of these primitives were allowed. In that case, the corresponding region shall remain modeled as a mesh. Several contributions proposed in these works are worth discussing. In [Lafarge 2009], a two-step approach is concerned, where first each vertex has to be labeled as either part of a type of primitive or as scattered. This problem is cast as an **MRF** where the singleton potential is based on an estimate of the curvature in two major directions. From this step connected components with a constant label are extracted. Those with a label corresponding to a certain primitive must then be processed in order to fit an instance of such primitive. This second step inspired by [Marshall 2001], is performed robustly by testing multiple hypotheses. In [Lafarge 2010], the segmentation is refined using a jump-diffusion process [Grenander 1994] which in essence alternates a jumping decision based on Metropolis-Hasting central idea, and a diffusion process. The goal lies in the minimization of an energy composed of three terms dealing respectively with photo-consistency, regularization and primitive layout. The layout prior favors perpendicular and parallel configurations between primitives.

Reconstructions obtained using the latter framework are depicted in Figure 2.6. Results demonstrate the expecting behavior where cluttered regions are preferably modeled thanks to meshes while others are represented by adequate primitives. More careful consideration results on two observations. On one side, irregular parts are sometimes recovered as a set of many small primitives and on the other side, primitive types can be swapped. The latter case concerns more specifically cylinders and tori.

2.3.4 Summary

Having discussed various approaches based on primitives, one can summarize their strengths and limitations. Piecewise planar representations are a promising effort towards compressed reconstruction of man-made scenes. Their success relies on the simplicity of the proposed hypotheses,

and the generality of such descriptions. Meanwhile, substantial research effort was devoted towards more advanced than planar primitives for two main reasons. Geometric considerations were considered first, since architectural landscapes are not compound of planes exclusively but display also a variety of curved shape such as spheres, cylinders or cones. The second reason, which is fundamental to computer vision in general, concerns symbolic understanding. It is common that some particular arrangements of planes are more recurrent than others. These arrangements often bear strong semantic information about the scene. Eventually, because accuracy and compactness are so contradictory, faithful reconstructions based on primitives only can hardly be conceived. In this context, few works merge primitives and less structured description in the same framework. Although impressive capabilities were demonstrated, it is clear that there is much room for improvement in this direction.

Despite important investment on scene representation through primitives, prior art has approached the problem from a questionable angle. The essential question that one has to answer might be *"How is organized the scene?"* rather than simply *"What is it composed of?"*. This is a central difference between compositional modeling and structural modeling to be described in the next section.

2.4 Structural Modeling

Tackling the problem of reconstruction by considering structure is appealing and sound. It is appealing because the obtained reconstruction might well be much richer than the output of usual representations. It is besides reasonable to think that detecting structure can greatly reduce ambiguities that are inherent to noisy observations. Even if the task of imposing structure can be tackled efficiently through user interaction [Debevec 1996, Nan 2010], we focus in this section on automatic derivation of the scene structure. In this context, we will widen the scope of the survey to scene interpretation in general, because many interesting papers do not deal strictly with 3D reconstruction.

Parsing vs model inference

Two complementary tasks might be targeted when dealing with structure. In many works, a generic structural model is assumed, and one naturally seeks to generate an instance which best explains available observations. This major component of structural modeling, called parsing, will be discussed thereafter.

It is also desirable to extract the structural principles from examples. This branch can be thought as model inference or model learning. It would mean that a program would be fed with a set of objects sharing common features, and would output a structural model - for instance a set of associative principles believed to be universally followed by the class under consideration. However appealing the previous formulation, it has remained so far an utopia. A more realistic way to reformulate model inference consists in assuming that a generic model is known for an exceedingly broad class of objects, and one shall narrow the model to the class corresponding

to some observed objects. Another way to think of inference is as pruning the space of shapes spanned by the generic model. Despite being interesting, model learning has drawn little attention [Merrell 2007, Becker 2009, Šťava 2010, Bokeloh 2010].

Model inference and parsing are certainly not exclusive. It is possible to consider structural inference by first parsing various examples using the generic model, and then summarize the common configurations revealed in this stage (for instance, in a statistical manner). In [Schnier 1996] a similar approach, where the search is progressively biased to account for configurations learned from already parsed examples, was considered . Additionally, as proposed in [Becker 2009], one may gather knowledge thanks to structural inference and use it to improve the individual parsing, in particular for regions that are poorly/not visible in the data.

Parsing strategies

Structural modeling aims at bringing expert knowledge into the process of inference. This domain specific knowledge often accounts for constraints between meaningful parts of the scene. These constraints can be either local or global. When it comes to parsing specifically, this distinction is of strong relevance. If local constraints are considered, the model invariably aggregates them in a hierarchy from which the global structure emerges. The hierarchy can often be seen as a grammar parse tree where these constraints are encoded through production rules. In general, two diametrically opposite ways to get to grips with the parsing are possible. The first type of approaches, referred to as bottom-up, starts from the detection of terminal elements (primitives) of the hierarchy and fuse them in order to recover the structure. The fusion steps must be done in accordance with the structural principles encoded in the model. On the contrary, top-down approaches sample hypothetical models, evaluate their strengths and eventually perturb them in order to come closer to the solution. The top-down denomination mirrors this coarse to fine way of thinking. Otherwise, when a global constraint is settled at once, models are typically flat, so that the top-down/bottom-up distinction makes little sense. We will qualify methods falling into this class as bottom-up, because results are usually obtained from an image pass.

Symbolic primitives

This axis has been discussed in the case of compositional modeling and is a central aspect as structure is concerned. Therefore, prior art can be classified into two groups. In the absence of semantics, parsing often relies on the assumption that multiple instances of a primitive should have similar appearance [Müller 2007, Musialski 2009, Musialski 2010, Wu 2010]. Such a valid assumption bears a considerable ambiguity in the boundary of atomic elements composing the structured object than can be dealt using heuristics. For instance, authors often favor boundaries occurring at strong gradient locations [Müller 2007]. However, terminal elements of the decomposition do not inherit symbolic meaning, which leads to a less tangible interpretation. On the contrary, attaching semantics to the primitives improves the degree of scene understanding and it allows to explicitly encode appearance [Alegre 2004]. It results in a substantial decrease in the

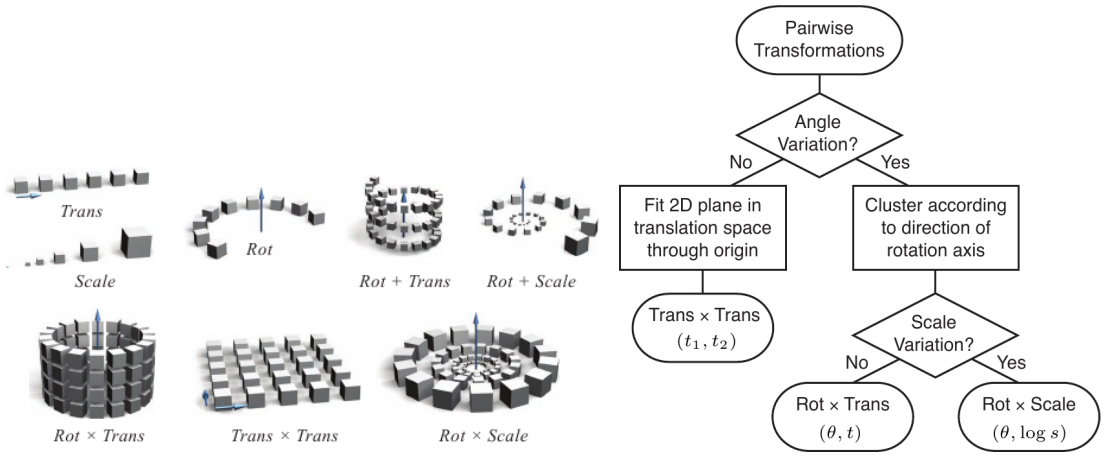
aforementioned ambiguity. One may argue however, that the expert effort is greater. Such objection becomes less annoying when the class of objects under consideration is of broad range. For instance, as far as buildings are concerned, semantics like windows, walls and balconies are so frequent that involving them is not a matter of expert but rather of common knowledge.

2.4.1 Bottom-up Analysis

Let us now proceed to an overview of works following a bottom-up scheme. An important number of methods investigate the problem under the angle of discovering repetitive pattern. Semantic interpretations are usually absent from such processes. In [Müller 2007], a 2D regular grid referring to the unknown parameters of the model was fitted to an ortho-rectified and cropped facade image. The two steps δ_x and δ_y of the grid corresponded to the unknown variables that were recovered by optimizing the mutual information [Wells 1996] between regions. Subsequent refinement steps were considered where the aim was to explore edge presence towards making the process more robust to noise while imposing geometric consistency on the position of tiles. This constitutes a nice illustration of the power of structure with respect to ambiguity. Taking into account the post-estimation steps, the approach would fairly rank among the top-down ones. Nevertheless, it is presented here since the repetitive patterns scheme was borrowed by many articles implying flat representations. For instance, [Musialski 2009, Musialski 2010] complete the translational symmetry with a reflective one, while in [Wu 2010] the same idea is explored for multiple repetitive patterns. Opposite to the previous works, the latter makes no assumption about ortho-rectification nor cropping and assumes no prior knowledge about the domain where patterns are to be found. The boundary of the elementary tiles are determined based on local symmetries in the direction of repetition and on similarity score drops in the other direction. At the end, the model is weaker from a structural point of view, but suits to more general configurations.

In [Van Gool 2007] the approach presented in [Müller 2007] was extended for facades captured from non frontal view points and with small focal lengths. In this case perspective distortion effects are significant enough so that depth reasoning can be conducted based on a single image and structure knowledge. This quite remarkable perspective, once again exemplify the power of structural modeling. An energy formulation is derived in the form of an **MRF** where each node involves an unknown parallax. Then the parallax field is corrected thanks to a shape prior imposing axis aligned rectangular elements. Practical results display reconstructions of facades with correctly excavated windows and protruded balconies. Later, [Xiao 2008] proposed a semi-interactive method, based on precomputed ortho-rectified depth maps using multi-view **SFM**. Despite demanding user interaction effort, the resulting reconstructions are highly detailed and much cleaner than the raw **SFM** models.

The aforementioned works tackle the task of recovering symmetries. They rely on models where no or little hierarchy is present. It would be interesting to study works investigating bottom-up parsing in the case of non flat models. Unfortunately, to the best of our knowledge, in computer vision, existing approaches choose either a totally top-down scheme or a bidirectional analysis. The rationale behind this observation is not clear. Yet, compared to other domains such as string



(a) Crystallographic group patterns (b) Parsing Algorithm

Figure 2.7: (a) Very regular patterns can be generated though a crystallographic group derived from translation, rotation and scaling parameters. (b) Parsing algorithm proposed by [Pauly 2008]. Image courtesy of [Pauly 2008].

grammar parsing, where bottom-up approaches can even be preferred to top-down, computer vision tasks are doomed by overwhelming ambiguities as tokenization is concerned. Therefore, information derived at low level of the hierarchy can hardly be consistently transferred to upper levels, unless validated by a top-down loop.

2.4.2 Top-down Analysis

In [Pauly 2008], an approach that is compelling with top-down analysis was presented. Their goal was to parse 3D models in order to recognize parts that involve repetitive patterns. Few 2-parameter patterns that can be derived from a discrete group of translations, rotations, and scaling transformations are considered (see Figure 2.7 (a)). Strictly speaking the patterns are regular, but more general than 2D grids. Besides, given that many combinations involving a subset of the available transformations are possible, it is beneficial to let a top-down algorithm decide which is present. The proposed algorithm is depicted in Figure 2.7 (b). Then each case is dealt within an adapted way. Experimental results are described on both synthetic and real data. The validation tends to prove that the algorithm is robust to noise and to missing data.

In [Gupta 2010], the problem of image parsing as the recovery of a structured set of physical blocks was considered. Strong importance was paid to the physical consistency of the blocks interpretation. Consistency was enforced by a set of constraints involving the law of statics and internal energies based on the density of the blocks. These constraints affect multiple blocks at a time, and therefore classical inference techniques such as graphical models were not possible. The

authors proposed therefore an iterative process starting from the most confident regions and adding one block at a time. At each step, a set of candidate blocks is generated accounting for the current configuration, and the best one is chosen following the score of the new global configuration.

Another work, following a strict top-down approach was proposed [Xiao 2009], where the goal is to reconstruct an urban area using different views. The scene is modeled using a hierarchical decomposition. A the top level, it is decomposed into sky, buildings and ground regions. Then the buildings are decomposed in facades and eventually the facades are represented as collections of structural rectangular elements. Like in their previous work [Xiao 2008], the method use a classical **SFM** approach to compute ortho-rectified texture and parallax maps. Once again the success of the whole pipeline rests on the one of each individual steps.

In [Vanegas 2010], a Manhattan buildings are models as successions of floors all of which are assumed to be made of facades aligned with one of three orthogonal planes. Floors are represented as strings using a turtle graphics convention allowing right turns only. The authors consider few typical transitions between successive floors and show that they can be interpreted as generalized rewriting rules acting on the aforementioned string representation. More precisely, each transition corresponds to a given rule controlled by three parameters. The reconstruction consists in recovering for each transition, the correct rule and the corresponding parameters. The score used to evaluate the quality of a rule is based on a correlation measure between the appearance profile of the floor and the position of the facade transitions. The optimization is driven progressively from the lower to the top floors[1]. In order to make the decision more robust, all the floors following the current one within a given range are considered to evaluate the score of the decision about to be made.

Such approaches rely on greedy decisions which once taken are not revisited. This has the non negligible advantage that the space of configurations is rapidly pruned as soon as decisions are based on strong evidence. In general though, ambiguities can possibly mislead the decision process. It is therefore often necessary to use a less direct path: once a hierarchical decomposition is found, it must be iteratively improved. Sampling methods provide a generic way to do so, like the **rjMCMC** extension of classic Metropolis-Hasting scheme, introduced in 1995 [Green 1995]. This method provides a convenient way to optimize models of varying dimension. It is well adapted to hierarchical model of unknown number of parts. A series of articles have used it to parse scenes encoded via a grammar. It started with [Alegre 2004], where a facade segmentation is interpreted as a parse tree of a split grammar. The facade decomposition is composed of locally aligned rectangular regions. The model assumes that each terminal element of the decomposition is of constant color to which is added a Gaussian perturbation. This leads to a Bayesian formulation accounting as well for priors on the parse tree. As usual, the **rjMCMC** sampling is used in order to approximate the posterior probability. This seminal work has initiated two research directions.

The proposed Bayesian formulation was too simple to be suited to common architectures. In this direction, in [Ripperda 2006], a more complex likelihood model involving both image and range data was considered. It accounts for repetitive behaviors and domain specific knowledge

[1]— Considering the hierarchical structure, the scheme remains top-down

such that windows lie behind the walls and appear darker too. Besides, a more powerful prior for the facade structure is proposed which favors symmetrical and well aligned configuration of the structural elements. In one of the article related to this thesis [Teboul 2010], we proposed to learn the appearance model of each terminal primitives. This lead to a very generic framework where no ad hoc priors such as constant color or darker windows are used.

The second direction is related to the design of the proposal distribution. Alegre and Dellaert [Alegre 2004], considered a simple way to propose new derivation trees, by pruning a randomly chosen sub-tree and re-deriving it. The re-derivation is performed by choosing a random applicable rule which parameters are optimized by exhaustive search. Such a strategy can in theory produce satisfactory results but in practice random choices and exhaustive search lack efficiency. Following [Zhu 2000], better ways to drive the proposal must be explored. This is where data cues are of practical interest, leading to mixed top-down and bottom-up analysis.

2.4.3 Mixed Approaches

As suggested previously, a bottom-up analysis can be set as a complement to another top-down in order to speed up to concentrate decisions on promising options. This argument was introduced in [Zhu 2000] within a Data Driven Markov Chain Monte Carlo (**DDMCMC**) paradigm. A toy example, called Ψ world, is used to demonstrate the principle of **DDMCMC**. The goal is to parse binary images composed of lines, circles and Ψ characters. The jumps corresponds to birth, death, composition and split. An element can emerge or disappear or split into two distinct ones (*e.g.* a Ψ can be decomposed into a line and a circle) and vice versa. The proposal for the parameters of each type of element is computed using a Hough transform. The main difference with respect to previously published **rjMCMC** techniques, is that it uses data cues to increase chances of finding jumps leading to high likelihood. As a result, the acceptance rates tend to be higher than with proposal based on mere priors.

The previous study case has been proposed to illustrate the generic algorithm. Since then other more interesting ones have been introduced where hierarchy is an inherent feature. Han and Zhu, proposed two approaches concerning architectural parsing. In [Han 2004], single view reconstruction of building is tackled. In this work the scene is decomposed into a background plane and a building. The building is encoded as set of 3D primitives tied together by a binary relation describing physical connections between them. The structure is further imposed in a smooth way through a prior enforcing row and column alignments of windows as well as beauty and regularity criteria. Besides an appearance model is tied to each face appearing in the primitives. The jumps correspond either to death and birth of 3D primitives, appearance model switching and addition and removal of binary relations, The model can be chosen among three types corresponding respectively to constant color, smooth color variation and cluttered appearance. The proposal sampling strategy for primitive moves is based on an aspect hierarchy going from line group, through faces and aspects (corresponding to combination of connected faces) and finally up to the level of primitives. Using this hierarchy, primitives likely to have generated the edge cues extracted from the image can be inferred and used to sample new proposals. The second approach [Han 2005] deals

with image parsing without explicit 3D representation. The grammar enforces different natural configurations between potential 2D projections of 3D rectangles. The approach assumes nested and aligned rectangles, cubical configuration, and is therefore appropriate for man-made object analysis. In [Zhu 2006] a generic framework based on a and/or graph representation is presented. Three different applications are proposed including the one we have just described. At last, in the same vein, [Ripperda 2007] followed the DDMCMC approach and extended the previously descried work [Ripperda 2006] in order to include bottom-up proposals.

Towards a quite different objective than **DDMCMC**, [Becker 2009] proposed an alternative approach which has interesting potentials. In this work, range data are first processed in a bottom-up fashion in order to fit grammar rules. The extracted rules are aggregated in a facade grammar that can be used in a top-down way to reconstruct poorly observed regions or even parts which include no sensor data at all.

Last but not least, [Toshev 2010] tackles piecewise planar reconstruction based on grammar parsing. The terminal nodes of the grammar are planar patches automatically extracted from a point cloud. Non terminal nodes are either roof parts, volumes or two unique super nodes used to discriminate between elements attached to a building or to various cluttering objects such as trees. The grammar is composed of five rules where two are used to gather the atomic patches into larger planes and roof parts. These rules are parsed in an automatic manner by using respectively a coplanarity and a vicinity predicate. A third rule is parsed deterministically to infer the core volumes from the previous planar parts. The two remaining rules encode the hierarchy of the volumes as well as the classification between building and clutter. The remaining parsing task is cast as a maximum spanning tree problem. Therefore, the approach uses an entirely bottom-up approach to aggregate low-level information (based on predicates and other features) and validates its consistency in a top-down process introducing a strong structure.

2.5 Conclusion

Recall that our aim was to conduct a broad survey of 3D reconstruction methodologies. Our objective was not to perform an exhaustive review of the state of the art but rather to extract the logical path leading to the current trends that have mostly inspired our work. From that perspective, approaches were clustered into three major categories based on the characteristics of the inherent representations. Scattered representations were historically the first considered, because of their simplicity. In these circumstances, advances have permitted to define consistent energies and to reach accuracy in reconstruction of small objects or even lately of outdoor environments. Arguing that man-made objects have very peculiar properties, compositional approaches tackle the problem by looking for combinations of regular geometric primitives. However such collections of primitives are not entirely representatives of manufactured designs, because of their lack of structure. This last argument explains the recent drift towards positioning structure at the heart of considerations. It led to a class of approaches labeled as structural modeling in general.

We have also defended the idea that understanding structure and semantic decomposition of

object is sound and adds value to the outcome of reconstruction. This literature study has shown that approaches based on grammars have contributed in that direction and bear great promises. Besides, the next chapter will show that grammars were considered for shape modeling with quite some success. Architectural designs are no exception to this statement.

Chapter 3

Procedural Modeling

In the previous chapter, we have discussed various representations useful to 3D reconstruction. In particular, a recent trend towards introducing structure as a central aspect led to the development of shape grammars in this context. Such representations have been studied extensively in urban planning, computer graphics and computer vision communities. They were first considered for formal design analysis, then for random generation of peculiar designs and more recently for automatic image-based reconstruction of existing environments. In this chapter, we discuss the strengths of these representations from the computer graphics point of view, towards producing models following a generic formal rules. This angle is of particular interest in digital image synthesis while the opposite (inverse) relates to computer vision and will be discussed in the following chapters.

The remaining of this chapter is organized as follows. In section 3.1, we present the different modeling policies and their historical foundations leading to the procedural approaches. A brief overview of applications to urban modeling is presented in 3.1.4. Section 3.2 is dedicated to the procedural framework developed during this thesis while various examples of shape grammars defined within this scope are presented in section 3.3 with emphasis on grammars adapted to Haussmannian architecture.

3.1 State of the Art

Procedural modeling is generic and embraces a variety of techniques. It is therefore difficult to give a formal definition of it. Its main characteristic stands on the constructive vision of modeling that does not correspond to a static description but rather to a construction made of elementary steps as presented in the seminal work of [Newell 1975].

In this thesis however, the term procedural modeling will be associated with design techniques that are based on the notions of production rules and derivation process. These rules were first defined in the context of formal grammars. Such procedural modeling is closely related to shape grammars that the terms are often switched in the literature. This confusion will be avoided in this section but tolerated in the remaining parts of the thesis. Our presentation will follow the historical

path that led to procedural modeling, going from Chomsky formal grammars to shape grammars, and eventually ending with the peculiarities of procedural modeling itself.

3.1.1 String Grammars

Introduced in [Chomsky 1957], formal grammars have been used in the theory of languages to determine generative models of grammatical sentences. Linguistics and more recently of Natural Language Processing (**NLP**) are the domains being associated with them. However, other domains have benefited from their development, like programming languages which could be described in generative terms by a grammar. Surely the study of natural language involves concerns unrelated to this thesis. However, shared notions make it a relevant example to illustrate formal definitions. Furthermore, this domain was the driving force of the theory of grammars, and one can observe strong analogies with shape grammars. It will therefore occupy a central illustrative position in this section.

Following [Chomsky 1963], a grammar is "a set of rules that give a recursive enumeration of the strings belonging to the language". More precisely, a formal grammar is described by the following elements:

- a vocabulary $\mathcal{V} = \mathcal{N} \cup \mathcal{T}$,
- a finite set of production rules $\mathcal{P}$,
- an initial symbol $\omega \in \mathcal{N}$ called axiom.

The vocabulary is a finite set of symbols that comprises terminal ($\mathcal{T}$) and non terminal nodes. In general, the production rules are mappings from the set of finite strings of symbols V^* onto itself ($*$ being the Kleene star). A rule can be represented as follows:

$$\boldsymbol{\alpha} = \alpha_1 \dots \alpha_n \in V^* \rightarrow \boldsymbol{\beta} = \beta_1 \dots \beta_m \in V^* \tag{3.1}$$

In practice, the partition of the vocabulary into terminal and non terminal symbols is not explicit and is based on the fact that a given symbol appears at the left-hand-side of a rule or not.

In order to illustrate this concept, let us expose a pedagogical grammar example which expresses linguistics considerations. The vocabulary is composed of the following non terminal symbols $\mathcal{N} = \{\text{phrase}, \text{subject}, \text{verb}, \text{object}, \text{noun}, \text{pronoun}\}$ and the following terminal symbols $\mathcal{T} = \{\textbf{Peter}, \textbf{Mary}, \textbf{he}, \textbf{she}, \textbf{him}, \textbf{her}, \textbf{likes}\}$. It is associated with the rules presented in Table 3.1.

Derivation

From a generative perspective, a formal grammar can be used to generate sentences. To do so, a process called the derivation is defined (see Table 3.2). The starting configuration consists of an axiom symbol. The process adopts a rule which left-hand-side is exactly composed of this symbol

phrase	→ subject verb object
subject	→ noun
subject	→ pronoun
object	→ noun
object	→ pronoun
noun	→ **Peter**
noun	→ **Mary**
pronoun	→ **he**
pronoun	→ **she**
pronoun	→ **him**
pronoun	→ **her**
verb	→ **likes**

Table 3.1: A didactic formal grammar.

and replaces the current sentence by the right-hand-side string of the rule. Therefore, one needs to locate a sub-string matching the left-hand-side of a rule and replace this sub-string once again with the new right-hand-side. The process should continue until no rule can be applied anymore. If the grammar is well designed, the derivation will stop when only terminal symbols appear in the outcome string, but it can also be that no sub-string matches any of the production rule left-hand-sides.

$s \leftarrow \omega$
while $\exists r(\alpha \rightarrow \beta) \in \mathcal{P}$ s.t. $s = s_l \alpha s_r$
    $s \leftarrow s_l \beta s_r$
end

Table 3.2: Derivation process for string grammars.

Let us consider the previous grammar example (Table 3.1). Using the derivation process, several different sentences can be generated. An step-by-step example is depicted hereafter.

init:		$s \leftarrow$ phrase
phrase	→ subject verb object	$s \leftarrow$ subject verb object
subject	→ noun	$s \leftarrow$ noun verb object
noun	→ **Peter**	$s \leftarrow$ **Peter** verb object
verb	→ **likes**	$s \leftarrow$ **Peter likes** object
object	→ pronoun	$s \leftarrow$ **Peter likes** pronoun
pronoun	→ **her**	$s \leftarrow$ **Peter likes her**

Table 3.3: Example of derivation of a string grammar.

Grammar	Language	authorized forms of productions
Type-0	Recursively enumerable	$\boldsymbol{\alpha} \rightarrow \boldsymbol{\beta}$, $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathcal{V}^*$
Type-1	Context-sensitive	$\boldsymbol{\gamma} A \boldsymbol{\delta} \rightarrow \boldsymbol{\gamma}\boldsymbol{\beta}\boldsymbol{\delta}$, $A \in \mathcal{N}$, and $\boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta} \in \mathcal{V}^*$
Type-2	Context-free	$A \rightarrow \boldsymbol{\beta}$, $A \in \mathcal{N}$, and $\boldsymbol{\beta} \in \mathcal{V}^*$
Type-3	Regular	$A \rightarrow bC$, $A, C \in \mathcal{N}$ and $b \in \mathcal{T}$ $A \rightarrow b$, $A \in \mathcal{N}$ and $b \in \mathcal{T}$

Table 3.4: The Chomsky hierarchy.

Chomsky hierarchy

Such a simplistic grammar can generate sensible English language sentences. However, we did not explore all the possible derivations, which could produce erroneous sentences such as "Peter likes she". It is not the point of this thesis to thoroughly discuss the issues arising in **NLP**. However this issue is important to modeling as well, since it concerns the expressive power of a formal grammar.

As soon as 1956, in [Chomsky 1956] proposed grammars were classified according to their expressive power. This hierarchy is still considered to be the most representative.

As depicted in Table 3.4, this hierarchy suggests four classes of grammars, ranked by decreasing order of expressive power. Each class in this hierarchy is characterized by constraints on the form of the production rules, which restrict the generality of the language generated by the grammar. The type-0 class corresponds to unrestricted grammars where a string is replaced by another string. Within the remaining three classes, only a single non-terminal symbol (denoted by A) is replaced at a time. Therefore, the derivation process is tied to a *parse tree* where each internal node contains a non-terminal symbol corresponding to the left-hand-side of a rule applied at some point. Such a node is expanded with children accounting for the symbols emerging from the application of the rule. In context-sensitive grammars, the replacement can depend on the context of the symbol to be replaced. In the corresponding production form, $\boldsymbol{\gamma}$ and $\boldsymbol{\delta}$ stand respectively for the left and right context. In context-free grammar, both contexts are always empty. Grammars of this type are not powerful enough to describe natural languages, but are well adapted for most programming languages. Regular grammars, introduce an additional constraint on the right-hand-side production, which is composed of at most one terminal and one non terminal symbol. It is interesting to note that all context-free grammars can be converted into a form close to regular grammars. This form - called the Chomsky Normal Form (**CNF**) - enables the use of right-hand-side with two non-terminal symbols as well as those of the regular form. In such a form, any derivation produces a binary parse tree, which makes the whole process more systematic.

Extensions

Many extensions of the previous formal grammar framework were proposed for various applications. We present a subset of them, the most relevant ones with interest for subsequent developments in the context of this thesis. Affix grammars, for instance are improving the symbols of the

subject	→ pronoun.t=subject
object	→ pronoun.t=object
pronoun.t=subject	→ **he**
pronoun.t=subject	→ **she**
pronoun.t=object	→ **him**
pronoun.t=object	→ **her**

Table 3.5: Affix use for better grammatical agreement.

vocabulary with attributes called affixes and with values in a finite set. They were useful in **NLP** due to the fact that they imply more discriminative derivations. For instance in the example of Table 3.1, in order to avoid sentences like "Peter likes she". One can add an affix (denoted by t in Table 3.5) expressing whether a pronoun derives from an object or a subject.

Similar affix rules can be introduced in **NLP** to handle other grammatical agreements with respect to gender and number. Having said that, the most interesting property of these grammars refers to their generalization to other types of attributes (as integers, or real valued numbers). Recent versions of shape grammars definitely fall into this scope, if we consider the geometric description of a shape as an attribute. However, given that shape grammars are strongly focused towards geometry, we will discuss them separately.

Other extensions worth mentioning, introduced conditions and probabilities and consequently have greatly influenced shape grammars. The scope of stochastic conditional context-free grammars consists of rules of the following form:

$$p, c : A \in \mathcal{N} \rightarrow \boldsymbol{\beta} \in V^* \tag{3.2}$$

where p is the probability of applying the rule during the derivation, and c a necessary condition for its consideration. The condition can involve Boolean expressions that can be evaluated given the node of the parse tree under consideration. As a result, even if the grammar itself is context-free, the context of a rule can still be checked before applying it.

3.1.2 Lindenmayer-Systems

L-systems were introduced by Lindenmayer [Lindenmayer 1968] for the mathematical study of the structure and the development of filamentous organisms. Without entering into subtle nuances, L-systems are considered to be a first attempt to use formal grammars into producing geometry. The approach is quite different from shape grammars but worth mentioning. A shape is processed in two separate steps. In the first stage, a derivation process is performed. This process differs slightly from the one previously described due to the fact that the derivation of all non terminal symbols of the current string is performed in one shot. Such a subtle difference results that L-systems are designed as parallel rewriting and grammars as sequential rewriting systems. In the second stage, the string can be interpreted geometrically using a LOGO turtle procedure described bellow. This

F	Draw a unit line segment forward
f	Move one unit step forward (without drawing)
+	Rotate right of an angle α
-	Rotate left of an angle α
[	Push current position in the location stack
]	Go to the position recorded at the top of the stack and pop it.

Table 3.6: A limited number of LOGO turtle symbols along with the associated geometric command. The angle α is a parameter of the turtle.

interpretation can be performed at any given iteration of the derivation, leading to a multi-scale description.

LOGO turtle

The geometric interpretation is obtained by interpreting each symbol as a drawing or move command (amusingly thought to be executed by a turtle). Few common symbols and their interpretations are presented in Table 3.6.

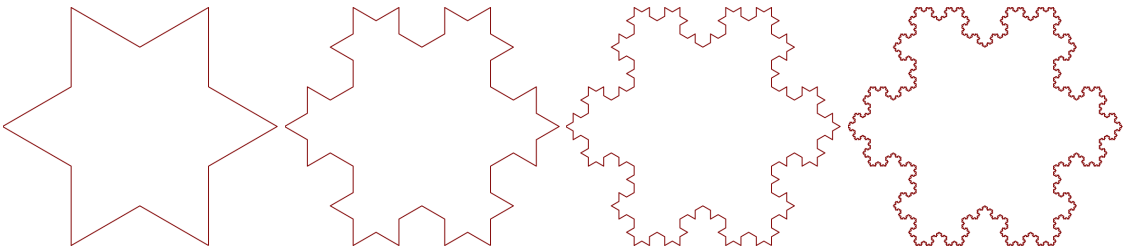
Historically, F, $+$, $-$, where the first symbols introduced to model fractal curves such as the Von Koch snowflake (Figure 3.1). Then f allows to create separate connected components. A typical example is the Cantor dust. In order to model branching structures such as plants an trees, the two brackets symbols were introduced in [Prusinkiewicz 1986] respectively to record the current position of the turtle in a stack and recover it. Note that other non terminal symbols can be added without the necessity of being associated with turtle commands. Later on, many extensions were proposed for example to produce stochastic models. In graphics, an extension of the turtle to 3D commands was presented in [Abelson 1981] which introduced two additional symbols for pitch and roll rotation. Other commands corresponding to color changes, or other stroke parameters are also possible.

Von Koch snowflake

This curve is a typical example of fractal curved obtained thanks to an L-system. In this case the axiom is $\omega = F{+}{+}F{+}{+}F$, and the grammar is composed of a single rule $F \rightarrow F{-}F{+}{+}F{-}F$. Each rotation correspond to an angle $\alpha = 60$. Curves generated at different scales are depicted in Figure 3.1.

Algae and plant generation

Let us illustrate the use of the bracket symbols with an algae system. The axiom is $\omega = F$, and a unique rule $F \rightarrow F[+F]F[-F][F]$ creates two branchings. The different scale representations are



(a) $n = 1$ (b) $n = 2$ (c) $n = 3$ (d) $n = 5$

Figure 3.1: Generation of a Von Koch snowflake fractal model with an L-system. The geometric representation is shown for different values n of the derivation iteration.

shown in Figure 3.2 for an angle value of $\alpha = 25.7$. We refer to [Prunsinkiewicz 1996] as a very comprehensive textbook regarding the modeling of plants thanks to L-systems.

3.1.3 Shape Grammars

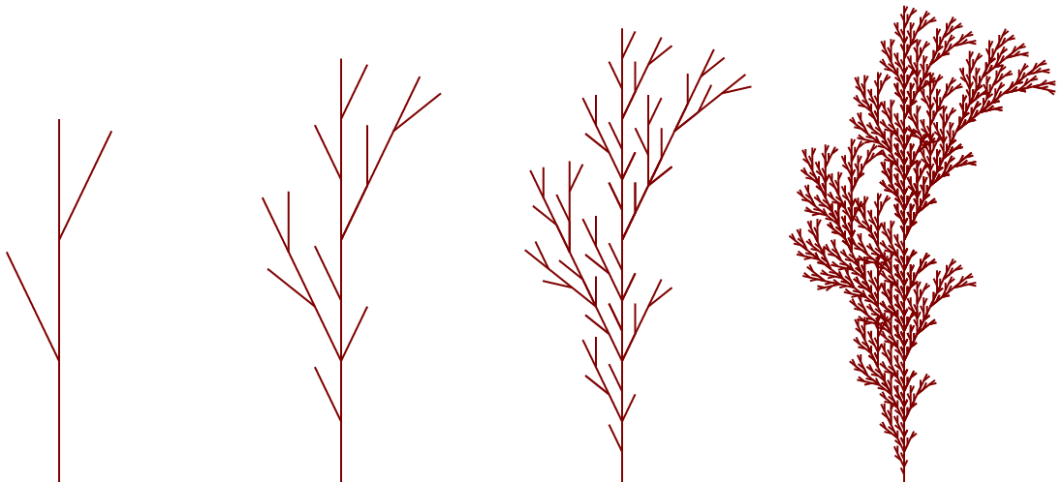
L-systems were successful towards modeling various structures undergoing a growth process. However, as a legitimate criticism, one can point out that the two-step procedure (*i.e.* a formal grammar derivation followed by an interpretation) makes it a non natural tool for design. This is an important obstacle to a general use of L-systems in modeling.

Stiny shape grammars

In 1972, the seminal work of [Stiny 1972] introduced a new mathematical framework, called *shape grammars*. It bears strong analogies with string grammars apart from the vocabulary definition. Instead of being composed of formal symbols, the vocabulary is a finite set of shapes. In this framework, shapes were defined as configurations of a finite number of points and line segments . Then, starting from an axiom shape chosen among the vocabulary, a derivation process (similar to the one considered for string grammars) allows to create new configurations.

It is worth mentioning that such shape grammars can produce similar designs similar to L-systems. For instance, the Von Koch curve can be generated by the grammar depicted in Figure 3.3.

This example suggests that shape grammars do inherit a more convenient visual representation and therefore have the potential to become an intuitive tool for modeling. However, shape grammars obeying such definition suffer a serious limitation with respect to automatic derivation. In fact, the complication comes from the fact that the derivation process involves a sub-shape matching problem which is much more intricate than the sub-string equivalent relative to formal-grammars. The problem is further enhanced in practice due to the fact that sub-shape matching is defined up to a similarity transform. Defining a programming language for shape grammars is also challenging due to the generality implied by the definition of shapes. As a result, such a framework



(a) $n = 1$ (b) $n = 2$ (c) $n = 3$ (d) $n = 5$

Figure 3.2: Generation of an algae model with an L-system. The geometric representation is shown for different values n of the derivation iteration.

is not adequate for computer aided design nor for specifying template models in view of reconstruction. As a matter of fact, it was mainly used in design analysis [Stiny 1978, Flemming 1987], where particular architectural styles were manually inspected in order to extract specific grammars.

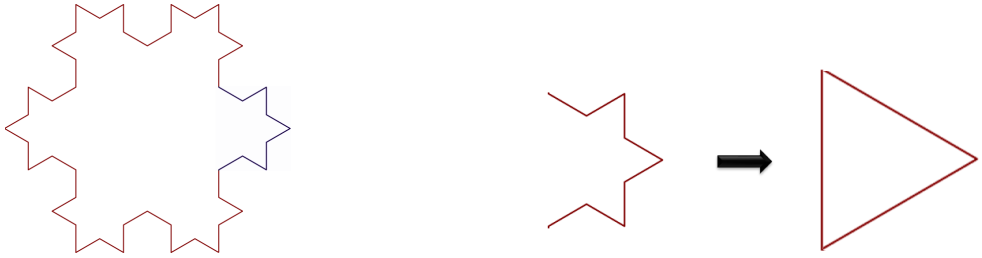
One very intriguing property of such grammars, relates to emergence. Derivation of simple grammars (even with a single rule) can reveal highly complex and detailed patterns. This is due to the fact that the emerging combinations of lines can be reinterpreted as various elements of the vocabulary. For instance, in the snowflake grammar, every time a single line is replaced by the right-hand-side of the rule, the emerging combination can be considered as four segments which can be derived independently. But the combinations of lines resulting from separate derivation steps also produced combination of lines which can be interpreted as the left-hand-side of a rule (see Figure 3.4). This reinterpretation is linked with the aforementioned sub-shape problem. Indeed at any step of the derivation, the current shape can be reinterpreted in many different ways leading to completely different derivations. The situation is even more complicated than the one associated with unrestricted formal grammars. It is hardly possible to control the derivation process in a general way and, in this context, derivation trees do not make sense anymore.

From set grammars to procedural modeling

Despite superior expressive power, the emergence property was dropped in favor of simpler and more efficient computer implementations of grammar interpreters, that is to say programs capable of reading a grammar specification and efficiently perform the derivation process. Such an inter-

(a) Axiom (b) Rule

Figure 3.3: A shape grammar for generating Van Koch snowflake.



(a) Emergent shape (b) Additional rule

Figure 3.4: After applying multiple times the rule of the snowflake grammar, new combination of lines emerge (a). They can incidentally form the left-hand-side of another rule (b). Assuming that such an additional rule was present in the grammar, it leads to different derivation paths.

preter should be able to answer two questions. How can we specify shapes and rules in a suitable form for a textual description? And how a standard automatic derivation process can be defined?

Both questions were partly answered in [Stiny 1982] where a new framework more tightly related to formal grammars was introduced. The framework was called *set grammars*, because shapes are seen as collections (or sets) of elementary primitives. A set grammar is specified as follows:

- a vocabulary $\mathcal{V}$ of primitives,
- spatial relations between the primitives
- a finite set of production rules $\mathcal{P}$,
- an initial primitive $\omega \in \mathcal{V}$ called axiom.

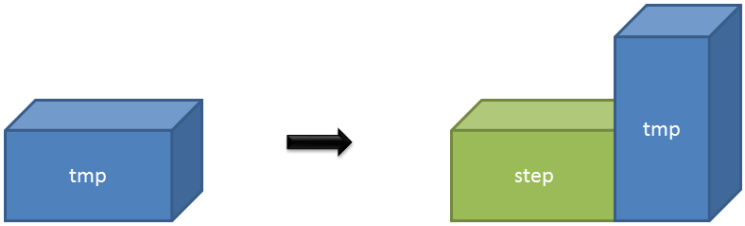


Figure 3.5: A rule defined in the set grammar framework. The right-hand-side can be described using a binary relation between the two primitives denoted by tmp and $step$.

A primitive involves the definition of a symbol and the associated geometry. As in formal grammars, symbols are represented by strings. The geometry can be two- or three-dimensional, depending on the application. Practically, the geometry can be expressed using universal shapes (squares, cubes, cylinders, *etc.*), or in terms of data structure interpretable from a computer (meshes, B-rep, *etc.*). Therefore it becomes possible to define the vocabulary in a text file. for instance the left-hand-side of the rule represented in Figure 3.5, can be described, using a self-explanatory notation, as $primitive(symbol = \text{'tmp'}, geometry = \text{'cube'})$. A rule can be represented textually too, as $lhs \rightarrow rhs$, where lhs is a single primitive symbol and rhs is a combination of primitives defined in terms of the spatial relations cited above. For instance, assuming that a relation called 'ortho' has been defined accordingly, the rule depicted in Figure 3.5, can be described as: $tmp \rightarrow \text{ortho}(step, tmp)$.

Primitive definitions are two-fold *i.e.* symbolic and geometric and therefore set grammars might seem halfway between formal and shape grammars. In practice, the symbolic representation conceals the geometric one with many respects, and therefore the situation concerning the derivation complexity is comparable to that of formal grammars. In that way, the set grammar framework brings practical answers to the second question. The derivation process is described in Table 3.7 using classical set operations. The shape s is initialized with the axiom, and, at each stage of the process, a non terminal primitive p is replaced by using a rule with a consistent left-hand-side. Similarly to formal grammars, a parse tree can be defined, and is actually used as a basis for grammar interpreters. Starting from a single node containing the axiom, the tree is iteratively expanded by applying rules to leaf nodes containing non terminal primitives. The process stops when the leaves of the tree contain only terminal primitives. Different derivation paths are possible, given that the leaves can be visited in arbitrary order. For example a Depth First Search (**DFS**) scheme can be implemented. Please note as well that Breadth First Search (**BFS**) corresponds exactly to parallel rewriting systems as L-system. In any case, as long as no context is introduced, the order has no impact on the final shape.

Building a functional interpreter requires one more step which aims at encoding spatial relations. The seminal work of [Stiny 1972] provides a first answer to that. It consists in placing markers on the primitives and as consequence, encoding spatial relations boils down to the posi-

$s \leftarrow \{\omega\}$
while $\exists p \in s$ and $r : lhs \rightarrow rhs \in \mathcal{P}$ s.t. $p.symbol = r.lhs$
    $s \leftarrow s \backslash p \cup r(p)$
end

Table 3.7: Derivation of a set grammar. The generated shape, denoted by s, is a set of primitives. Notation-wise, $r(p)$ stands for the application of the rule r to the primitive instance p.

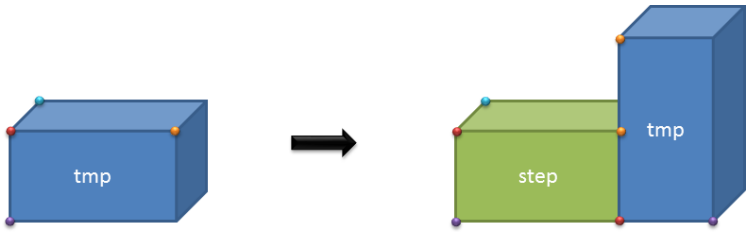


Figure 3.6: Spatial relations specified with markers (colored points).

tions of the markers of the primitives involved in the right-hand-side of a rule relatively to the left-hand-side markers (cf. Figure 3.6). The required number of markers depends on the symmetries of the primitive, and four markers are always sufficient (three when considering 2D grammars). In that case, it is equivalent to specify a generalized similarity transform between the location of the left-hand-side primitive and each of the right-hand-side ones. Despite the convenient visualization implied by markers, the latter encoding is better suited to computer representations and text file specifications. The rule in Figure 3.6 can be specified as:

$$tmp \rightarrow \text{identity}(step) + \text{translate}[dx, 0, 0](\text{rotate}[0, 90, 0](tmp))$$

Interestingly enough, leaving scaling transformation aside, this approach relates to the turtle moves as each transform specifies where the next primitive should be located. The idea of replacing turtle commands by transformations was first elaborated in [Hart 1992] where an object instancing paradigm corresponding exactly to set grammars was presented. This work was inspirational to the graphics community [Parish 2001, Marvie 2005] in its research on procedural techniques.

Procedural modeling goes further in handling spatial relations by introducing advanced procedures as alternative ways to position the right-hand-side primitives into generic arrangements. One can think of alignment configurations such as the one induced by the split procedure proposed in [Wonka 2003], the regular splits proposed in [Müller 2006b], or symmetric configurations. Although, these specific configurations can be described in terms of translation, rotation and scaling, the benefits of considering them are two-fold. First, in the perspective of coding a template model, the designer's programming task is simplified because a command can be introduced for each special arrangement. In that respect, the benefit is analogous to the use of macros in programming languages. Handling specific arrangements with appropriate procedures is also important because

(a) Straight windows

(b) Arched windows

Figure 3.7: Two nearby Parisian buildings. Even if they share the same architectural style (Haussmannian), some of their elements as the windows show different appearance. The building on the right presents two distinct iron patterns (see the running balconies of the second and last floors).

it explicitly accounts for the expert's knowledge. If on the contrary, translations, rotations and scaling were preferred to create a special configuration, then the knowledge would be concealed by the generality of the arrangements possibly modeled in that way.

The benefits inherited from the use of procedural modeling go beyond this line. Rather than considering that primitives are described geometrically from the beginning, these methods enable on-the-fly creation of geometry. In this case, the vocabulary is defined in terms of symbols associated with polymorphic appearances. The obtained grammar framework can be truly considered as an attributed formal grammar, where the geometry is a generalized attribute on which the rules impact. Such a property is of extreme value in architecture since elements with the same symbolic function can have different geometry depending on the building under consideration. For instance, windows can be arched or straight (cf. Figure 3.7), a variability that can also be observed within the same building. We refer to the example depicted in Figure 3.7 (b), where the patterns of the running balconies vary.

In such a context, additional procedures are to be considered to generate the geometry, as one for creating basic geometric primitive, $geometry[type](.)$, where $type \in \{cube, sphere, etc.\}$ and

a procedure to insert a model (mesh or else) stored in a file, $insert[filename](.)$. It is even more interesting that other procedures can be invoked to create adaptive geometry, which means that the outcome depends on the left-hand-side primitive instance. A typical example would be the extrude operator, which produces a generalized cylinder with its basis corresponding to the left-hand-side shape. In section 3.2, we will introduce more of these procedures.

3.1.4 Urban Procedural Modeling

Shape grammars have been used to analyze and generate different types of objects ranging from coffee makers [Agarwal 1998] to Hartley Davidson motorbikes [Pugliese 2002]. These concepts have emerged to large scale urban modeling at the beginning of the last decade.

Cities

In [Parish 2001], an approach was investigated aiming at procedural modeling of city layouts. Their first contribution was a general modeling scheme, suitable for large-scale environment production with limited user interaction. In that respect, large-scale environmental input data were expected such as under-water regions, and population density maps. Starting from these data, the modeling program should create a consistent street network, which can be used to produce building lots. Eventually, the lots are decomposed into individual buildings to be modeled separately in 3D. They implemented this general scheme in a software called CityEngine, where L-systems were used to produce street networks composed of highways, avenues and streets. Three frequent network patterns (organic, radial and rectangular) were considered. In addition, the L-system derivation produced a generic template which was truly instantiated as a post process guided by a set of global and local constraints driven from the input data.

Procedural generation of street networks was further studied in [Chen 2008], where the derivation is following a tensor field input. In [Coelho 2007], a different extension called Geo-spatial L-systems leveraged the need of external post-processing by integrating geo-spatial context in the production rules themselves. Besides the authors conclude their paper on the perspective of simulating spatio-temporal evolution of urban environments. This option was considered in [Weber 2009], where at each expansion step, the major street network is first expanded, followed by a completion of the minor network.

An interesting prospect when modeling urban environments procedurally is the potentiality of creating needed geometries at run-time (for example for close-by buildings). Building on this idea, [Greuter 2003] proposes a pipeline for the rendering of pseudo-infinite cities. The key idea is to populate the view frustum with buildings procedurally generated in real-time. As always in virtual environment model, the variability is obtained thanks to randomness. But, in order to keep the virtual environment invariant during the visit, the random seed used to generate a building at a specified location must be always the same along the simulation. This requirement is achieved by using a hashing strategy.

Buildings

In the previous list of works, the common thread is the underlying city layout. In order to create 3D geometry, it is necessary to generate a model per building in the layout. It is possible to generate nice visual renderings for fly-through visits with simple models (created by extruding footprints and mapping textures on the volumes obtained). We will consider procedural approaches to create highly detailed building models.

Following a general scheme proposed in [Marvie 2005] and then in [Müller 2006b] in a slightly more general form, building modeling can be decomposed in two successive stages. The first models the coarse volume of the building and the second places finely detailed structural elements on the surface of this mass model. Concerning mass models, a simple strategy starts from the footprint, extrudes it and optionally creates a non-flat roof model [Larive 2006]. Note that it is a sound strategy for two reasons. First, footprints can be generated easily and are also easily available for existing cities on the web services such as OpenStreetMap. The second reason stands on the mere fact that until recently, most buildings of widespread architectures could be faithfully represented in this way.

In [Parish 2001], coarse building models were generated using an adapted L-system associated with transformation operators as in the paradigm described in [Hart 1992]. Thanks to the iterations in the derivation of the L-system, the model was progressively refined. Therefore, an automatic generation of **LODs** was straightforward. The pattern used in this article to generate mass models is also interesting. The whole model is a sequence of volumes of decreasing basis area and increasing vertical position. In the initial L-system, step-forward volumes were obtained from the current ones by applying a random shrinking similarity transformation. This pattern was reconsidered in [Greuter 2003], with a generation driven from the upper toward the lower volume. At each step, a random polygon was added to the current footprint. Last but not least, it was considered in [Vanegas 2010], using other type of transitions described in the previous chapter.

Towards modeling of facades, in [Wonka 2003], the notion of split procedures was introduced to define the so-called instant architectures. Surprisingly realistic facade layouts were composed in that way using cube primitives only. Additionally a complementary grammar controls the derivation of the shape grammar in order to follow certain high level design objectives. Then the split idea was borrowed and completed with other procedures first in the FL-system [Marvie 2005] and then in the CGA grammars [Müller 2006b]. Inn [Marvie 2005] the split operator was integrated in the object instancing paradigm developed in [Hart 1992]. In [Müller 2006b] contributions include the notion of relative length specification for the split attributes and a repeat operator used to split a placeholder into chunks of identical size. Both components introduce adaptive behaviors either with respect to the size of the emerging primitive or to their number. Such additional features greatly improved the modularity of the framework. Furthermore, an operator called component-split, allowed to decompose a volume into its constituting faces and edges. This operator is of particular interest when aiming a link between mass models and surface modeling. Last but not least, other contributions relative to the control of the derivations were proposed. First, priority attributes are allocated to the rules and used to define a modified **BFS** derivation scheme. This

contribution makes sense when combine with two additional elements introducing context sensitivity. The first assumes that rule derivation is conditional to the outcome of occlusion queries. This allows for instance to avoid windows to be created at the boundary shared by two volumes. The second adopts numerical attributes for the splits which can be automatically adjusted to align patterns with the major lines of the overall structure. This concept is closed to snapping utilities of classical modeling tools and is therefore referred as snap lines.

Previously reviewed state-of-the-art methods consider the geometry of structural elements to be either simplistic or defined externally thanks to models stored in various file formats. Further enhancement of these models could allow to create each structural element procedurally. In this case, the previously mentioned procedures are not adapted because they are bound to moving and scaling primitives or creating simple geometry. In this direction, [Havemann 2001, Havemann 2005] developed a stack language called GML (standing for Generative Modeling Language), inspired by PostScript. In this framework, the representation of shapes relies on combined B-reps which couple meshes (used for global geometry control) and free-form patches modeled as subdivision surfaces. Based on this representation, and a closed set of low-level procedures corresponding notably to the Euler operators for the mesh parts, and to operators acting on the edge sharpness property for the free-form patches, complex modeling procedures can be designed and applied in a controlled way. This language allows to describe shapes of any complexity resulting on an excellent candidate for modeling structural elements. For instance, in [Havemann 2004], Gothic window models of great visual appearance were successfully encoded with GML. In fact, GML implements also classic operators on basic numerical variable and this makes it Turing complete. It is also fairly easy to define the split and other specific procedures within this scope [Hohmann 2009]. On the other hand, GML lacks higher level notion of structure and semantic context, which is introduced by rewriting systems. As such, a sensible option consists in using GML as an underlying language to define the procedures involved in a grammar-based modeling tool which could be extensible at will.

Given the countless publications relative to procedural modeling during the last decade, it is out of the scope of this thesis to draw an exhaustive survey. Yet, few other contributions, either technical or application related, deserve being mentioned. In [Lipp 2008], extended CGA grammars with persistent interactive editing were proposed while [Ilcik 2010] introduced kinematic behaviors based on the procedural modeling of a skeletal system. It also worth presenting the work of [Whiting 2009] where the modeling of self-adaptive attributed grammars was considered to enforce that the produced edifices are compliant with mechanical constraints. Besides, procedural modeling has been used for inferring plausible and detailed reconstructions of archaeological sites such as Mayan city [Müller 2006a] or ancient Rome [Dylla 2008].

3.2 An extended Procedural Modeling Framework

We will now describe in details the actual procedural framework developed in this thesis. Many components are not novel, and procedural modeling was not by itself the objective of the thesis. As pointed out before, our purpose is rather related to its adoption in the reconstruction of existing

environments from images. Consequently, it is questionable whether designing our own framework was a true necessity. The answer to this question was not clear at the beginning, but this option appears now to be justified. Indeed, developing it has provided us with a better understanding of the underlying concepts. Furthermore, relying upon a third-party modeler would have been cumbersome when considering reconstruction, as available modelers are thought for generation only. And access to internal representations, such as the derivation tree, might well be compulsory for the optimization. Last but not least, at the beginning of this project no procedural toolkit was available neither for commercial nor for research activities. The situation has recently changed with this respect.

The modeler that has been developed is called Centrale Procedural Architect (CPA). It was implemented in C++. The main modules and data structures of this software are provided in Figure 3.8. The global work-flow starts from an XML specification of the grammar. An XML parser is responsible for reading this specification file and instancing a C++ Grammar object. Then a driver module handles the derivation process and produces a Parse Tree. At last, the parse tree is transposed into a scene graph representation used as input for the rendering pass.

The reader should bear in mind that we will consider two different use-cases. 2D interpretation of facades requires the generation of layouts of 2D primitives (mainly rectangles) decomposing the complete scope of the facade. On the other hand, 3D reconstruction relies on 3D models of buildings. Depending on the use-case, the relevant definitions can slightly differ, but in all generality, the 3D case encompasses the 2D one.

3.2.1 How to Represent Shapes?

In CPA, shapes are made of collections of primitives. But primitive so far was used as a rather general qualifier. We must therefore provide formal definitions.

Primitives

In the CPA framework, primitives are described by different properties useful at different level of analysis. In simple word a primitive is described as a tuple $p = (s, c, d, a)$, where:

- s is a symbol (a string). It is used as an identifier of the primitive class.
- c, d are two versatile properties, that is to say that they can vary among primitives of identical symbol. They are respectively called consistency tag and differentiation tag, and are used to achieve a fine control of the derivation process.
- a is the appearance of the primitive.

Appearance

A primitive is described by its geometry and the reflectance properties (encoded thanks to color, textures or shaders). It is also versatile, because operators may produce a different geometry or

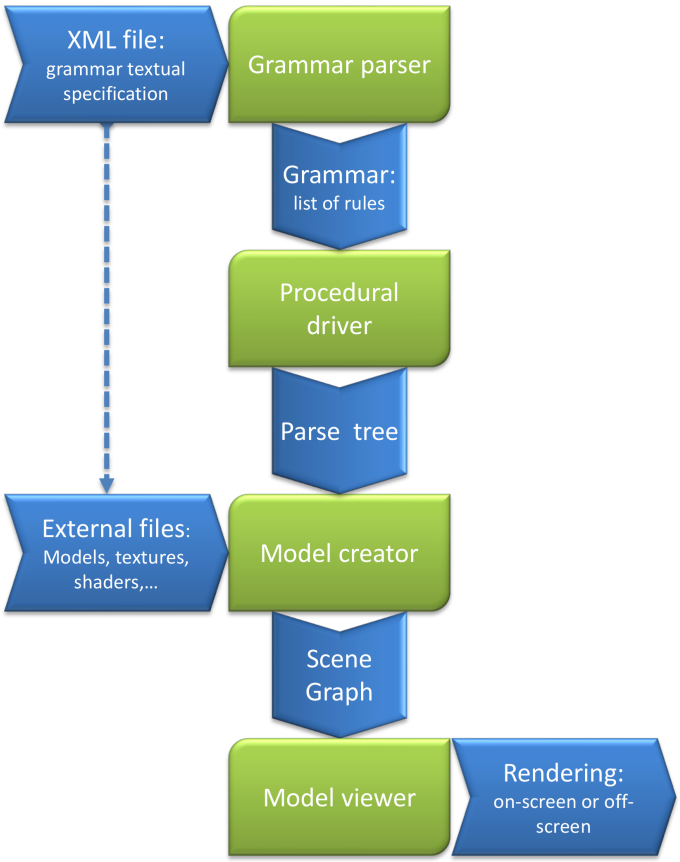
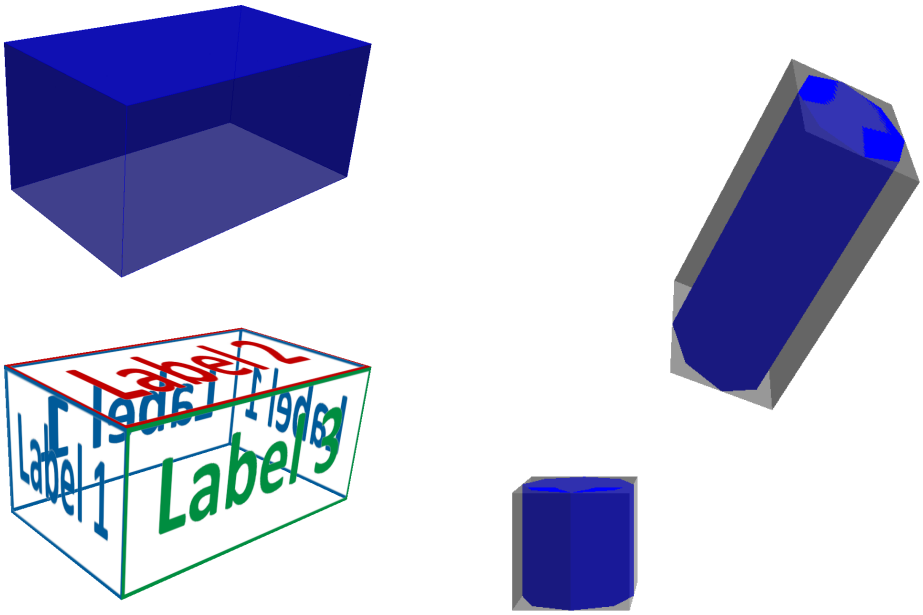


Figure 3.8: CPA internal structure. The flowchart represents the main modules (green blocks), and the data structures exchanged between them (blue).

color depending on the context in which they are applied. As for the geometry, it is represented as a mesh for a 3D application or a 2D polygon otherwise. Note that our definition of meshes encodes to certain extent symbolic information which may be useful to some operators. This information consists of a string label associated with each face of the mesh (as illustrated in Figure 3.9 (a)).

Additionally, following [Müller 2006b], we associate with the geometry a transformation called scope (see Figure 3.9 (b)), specifying the location, orientation and scaling of the primitive. Rather than manipulating the mesh of a primitive, an operator can act on its scope if its only purpose is to change its position or scale. In illustrations the scope will be viewed as 3D box which is the image of the axis aligned unit cube by the scope transformation. In CPA, the actual definition of scope involves an additional transformation, corresponding to reflections along the three main axes. This supplementary property allows to ease the definition of operator performing mirror configurations.



(a) Labeled Mesh (b) Scope

Figure 3.9: Appearance of a 3D primitive. (a) Upper: a primitive is defined geometrically as a mesh - Lower: each face is associated with a label. (b) The scope of a primitive (gray box around the blue mesh) delineates the absolute location, orientation and scaling of the primitive. A cylindrical blue mesh is shown in its natural disposition (lower left) and after its scope has been modified (upper-right).

Axiom

Among the primitives, the axiom plays a key role. We need to specify all of its properties beforehand, while for other the properties are obtained automatically when deriving the grammar. This specification is also done via an XML file, describing the geometry of the axiom. We found general enough (for building applications) to start with a volume obtained by extruding an horizontal polygon. The upper and lower bases are labeled respectively as “top” and “bottom”. Besides, the lateral faces of this volumes are labeled, accordingly to cadastrial information, as “street”, “courtyard”, or “neighbor”. This allows in particular to model street and courtyard facades differently, while facades adjoining two buildings need not be modeled. In practice, the XML file defines the polygonal basis, and the label of each edge of the polygon. After extrusion, this label will be associated to the lateral face swept by the considered edge.

Note that we have decoupled on purpose the specification of the axiom from this of the grammar. This was done on purpose in order to be able to reuse the same grammar on different axioms. For instance a complete cadastrial map can be processed with the same grammar to produce a street

or a district; the axioms are then derived from each of the footprints.

Compound Shape Models

In Figure 3.8, two compound representations are depicted. First, the parse tree itself is the inherent representation of shapes during a derivation. A parse tree is made of nodes containing a primitive. In this tree, the parent to child relationship reflects the left-hand-side to right-hand-side one encoded in the rule applied to the parent primitive. The final model can be obtained by instancing each terminal primitive located in a leaf.

Besides, the parse tree can be turned into an optimized structure for rendering called scene graph. We have used the OpenSceneGraph toolkit which provides an efficient implementation of it.

3.2.2 How to Interact with Shapes?

Rules provide a direct way to acts on shapes. With respect to notations, a rule will be represented as follows:

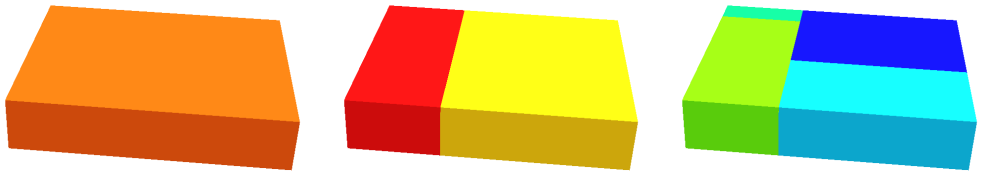
$$rid : condition, lhs \rightarrow rhs[\mathbf{x}|s_1, \ldots, s_m](.),$$

where rid is an identifier, $condition$ a necessary condition, lhs is a primitive symbol referred as the left-hand-side of the rule, and $rhs[](.)$ is a parametric operator referred as its right-hand-side. A rule can be applied at any leaf node of a parse tree under certain conditions that will be developed later. If all constraints are satisfied then we will say that the rule is applicable to the node.

It is actually the right-hand-side operator that is applied to the primitive. An operator (or procedure) can be thought as a function taking a primitive argument p, a list of primitive symbols $s_1, \ldots, s_m$ and a set of numerical parameters $\mathbf{x}$ and producing a set of primitives $p_1, \ldots p_n = rhs[\mathbf{x}|s_1, \ldots, s_m](p)$. Virtually, the operator creates a set of geometric primitives which are associated in a specific way with the m symbols. In most cases, the number of primitives matches the number of symbols (*i.e.* $n = m$) and the primitive p_i is associated with the symbol s_i. In some cases however, the association of an emerging geometric primitive with one of the symbols is less trivial.

In order to allow for compact grammar definitions, operators can be composed (cf. Figure 3.10). This aggregation is achieved by replacing some of the symbolic arguments by other operators. An operator is then of the form $rhs[\mathbf{x}|op_1, \ldots, op_m]$, where the op_is can be operators or symbols. We will refer to this composition as operator nesting. It leads to a recursive definition of operators. Each geometric primitive p_i emerging from the application of rhs will not be associated with a symbol but further treated by the associated nested operator op_i. The application of an operator is therefore a recursive process that stops when reaching a state without nested operators. At the end of the process, the temporary geometric primitives are discarded and only the ones appearing at the end of the recursion are returned.

The global process of shape creation follows the same scheme as this presented in Table 3.7 with minor modifications (described in pseudo-code in Table 3.8). It starts with a parsing tree com-



(a) primitive p (b) parent operator (c) nested operators

Figure 3.10: Starting from the primitive p, the rightmost configuration (c) can be realized by applying a first rule creating the center layout (b), and then applying a rule on each of the red and yellow primitives. Otherwise, it can be achieved via a single rule with nested operators.

posed of a single node containing the axiom primitive. At each step, the current node is processed by choosing an applicable rule and adding the emerging primitives in children nodes.

The numerical parameters can be of three types: literals, variables, and expressions. Literals are deterministic parameters, variables are randomly generated at run-time and constitute one way of creating variability, while expressions are used to introduce complex dependency on the context. The latter were implemented by embedding a python interpreter in the CPA software. They can therefore involve any valid python expression (including random number generators). Besides, the expressions can refer to the current node of the parse tree and proceed to topological operations such as accessing its parent. When a node is considered, access is granted to all primitive properties except for the appearance which is accessible via the scope. For example the following valid expression returns the size in the X direction of the scope of the parent node of the current node:

$$\text{return } lhsNode.parent.scope.scaling.x \tag{3.3}$$

We should note that CPA handles only rules with a single symbol at the left-hand-side. Such an approach corresponds to context-free grammars in the Chomsky hierarchy. Nevertheless, within our formulation, context is available in a much more extended way than in context-sensitive grammars thanks to the topological operations allowed in expressions.

3.2.3 Control of the Derivation

The proposed derivation algorithm mainly differs from set grammars because of additional control in the way rules are chosen and applied. Normally, a rule could be applied to any node with a primitive sharing the same symbol as the left-hand-side of the rule. In CPA, the rule application is also dependent on its *condition* attribute. Conditions are expressions, and therefore usually depend on the current context.

Another peculiarity of the derivation, is that the rule to apply to a node is not always selected at random. Indeed, if the primitive currently under consideration holds symbol s and consistency tag $c \geq 0$, and if a primitive with identical settings has been formerly processed then both the rule and

$p \leftarrow \omega$
$n \leftarrow Node(p)$
$pt.tree \leftarrow Tree(n)$
$pt.history \leftarrow HashTable()$
repeat
    if $p.c \geq 0$ and $pt.history.haskey((p.s, p.c))$:
        $(r, \mathbf{x}) \leftarrow pt.history[(p.s, p.c)]$
    else:
        $(r, \mathbf{x}) \leftarrow$ (random rule applicable to n, random parameters)
        $pt.history[(p.s, p.c)] \leftarrow (r, \mathbf{x})$
    add $r.rhs[\mathbf{x}|s_1, \ldots, s_m](p)$ as children to n
    $n \leftarrow$ next node in $pt.tree$
end

Table 3.8: Derivation process in CPA. The parsing tree pt is initialized with a single node containing the axiom primitive. Then it is progressively expanded with new nodes. At each iteration, a rule is either chosen randomly among those applicable to p or searched in a hash-table $pt.history$.

its parameters are chosen equally to the previous application. The benefit of this will be illustrated in later examples. This requires the use of a hash table $pt.history$ storing the rule identifier and the parameters value for each pair of symbol and consistency tag encountered so far. Note that if c is negative, then the rule and parameters are always randomly selected as in the usual derivation scheme.

DFS vs BFS

In Table 3.8, the last operation of each iteration selects the next leaf node to process in the parse tree. Which node should come next? Such a question does not have always a unique answer. We have pointed out earlier that when context is considered, the answer to this question makes a real difference on the shapes generated with the grammar. In general, two major iteration schemes are possible over the tree: **DFS** and **BFS**. Their practical relevance depends on the actual application. And one thing is sure, the designer of the grammar must know which scheme will be used to foresee the consequence of context sensitivity. One solution consists in letting the designer specify the scheme within the specification of the grammar. In our case, we found **DFS** more natural for context reasoning so we only implemented this option.

3.2.4 Operators

In order to complete the technical description of CPA, we will list the available operators and describe their impact in terms of geometry and semantics. Such a description concludes the technical specification of CPA, even though the importance of some operators might be difficult to grasp

without specific use-cases. The following section will address this issue by presenting the complete modeling of different designs.

In what follows, operators are globally classified according to the involved properties. Note that as a general rule, properties left unspecified by the computations of an operator are inherited from the primitive p on which it is applied. The only exception concerns consistency tag which must be specified explicitly (using an ad hoc operator) and is otherwise always assumed to be negative (set to -1 by default).

Operators acting on scope

As a first class, we can consider the operators performing only computation on scopes. This include two separate groups: the transformation operators and the splitting ones. All the transformations applicable on scopes are considered, that is to say translations, rotations and scalings along each main direction of the scope of the primitive. This leads to the following list of operators that require a single numerical parameter, a unique symbolic argument and produce one primitive.

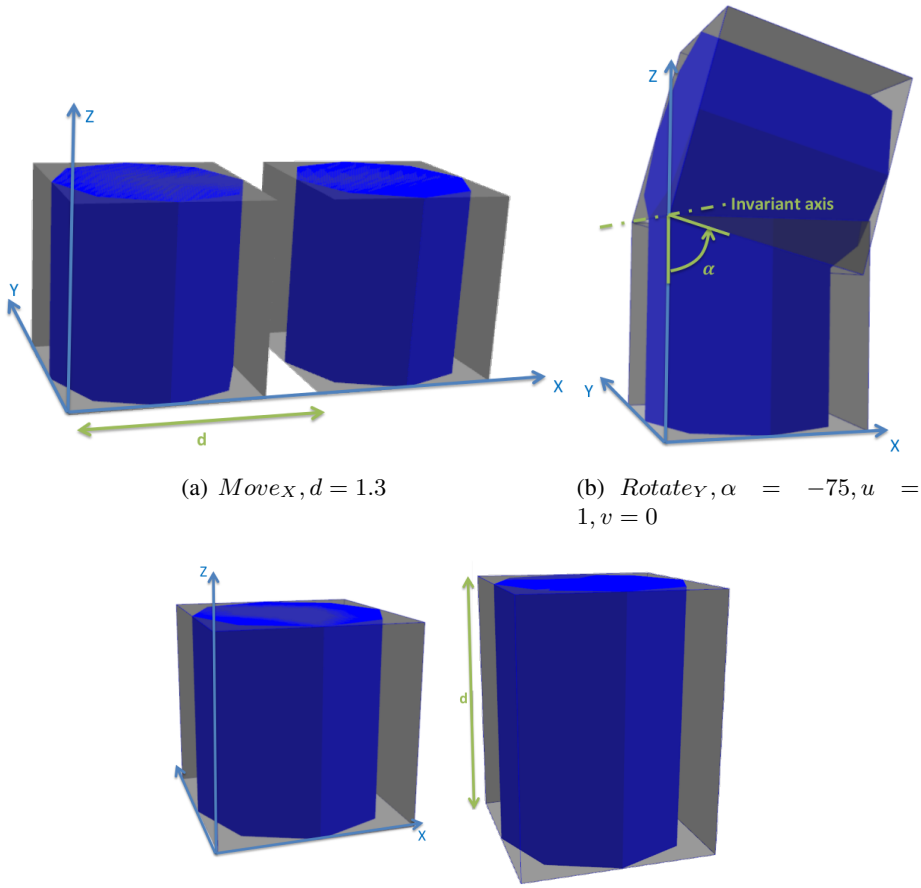
- $Move_{axis}[d|s](p)$ for translations,
- $Rotate_{axis}[\theta, u, v|s](p)$ for rotations,
- $Scale_{axis}[d, u|s](p)$ for scale transformations.

In each case, $axis \in \{X, Y, Z\}$, d and θ are numerical parameters standing for a distance and an angle (cf. Figure 3.11). Eventually u and v are utility parameters specifying anchors (*i.e.* invariant locations) for rotations and scalings. Through operator nesting, any generalized similarity transformation can be performed. Note that the role of u and v is redundant with the ability to compose translations with rotations or scalings.

Splitting operators cut the scope of the primitive on which they operate into chunks along a specified direction. Different modalities are possible:

- $Split_{axis}[d_1, \ldots, d_m|s_1, \ldots, s_m](p)$ is the most general form of splitting operator. It produces m scopes , aligned along one principal direction and filling the input scope.
- $Repeat_{axis}[d|s](p)$ implements a regular split.
- $Mirror_{axis}[d|s](p)$ and $Mirror_{axis}[d, d'|s, s']$ implement splits creating two symmetric scopes (and possibly a third scope in between).

The behaviors of the previous operators are illustrated in Figure 3.12.



(a) $Move_X, d = 1.3$

(b) $Rotate_Y, \alpha = -75, u = 1, v = 0$

(c) $Scale_Z, d = 1.4, u = 0.5$

Figure 3.11: Examples of transformation operators. The input and output primitives are depicted with their scope. Note that in (c) the input and output primitives have been separated for better visualization.

Operators acting on meshes

The proposed framework also allows to act on geometry. It is enriched with operators able to switch between volumetric and planar representations. The operator $Facetize[(l_1, s_1), \ldots, (l_m, s_m)]$, simply transforms the primitive p into separate meshes corresponding to each of its faces. Note that the number of emerging primitives is indefinite as it depends on the number of faces of p. Therefore the correspondences between the generated geometric primitives and the symbolic attributes are not trivial. This difficulty is addressed by specifying a correspondences between the face labels l_i and the symbols s_i. Note that $Facetize$ does not rely on any numerical parameter.

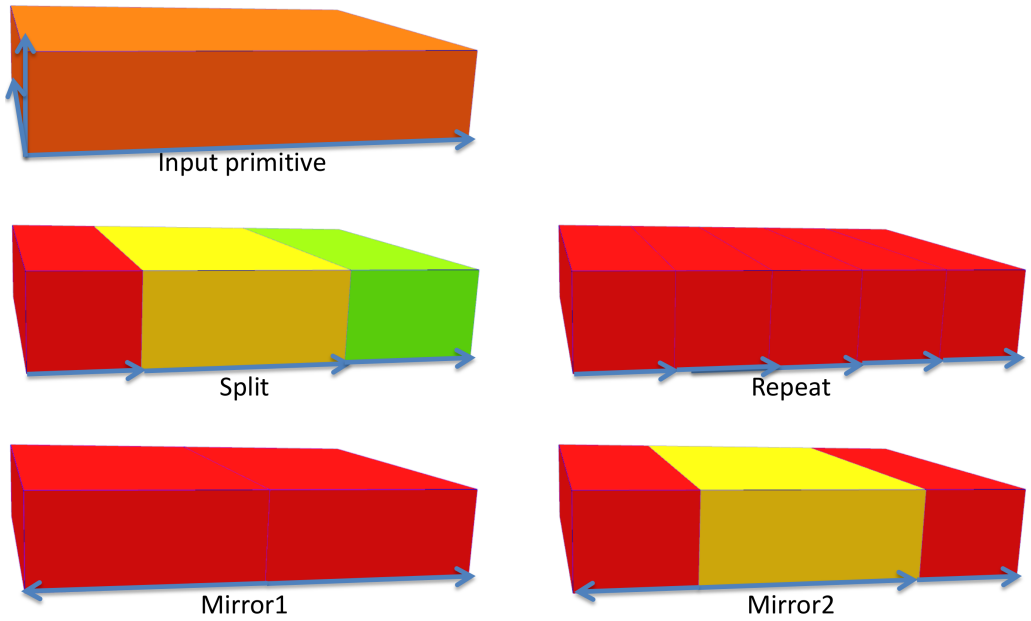
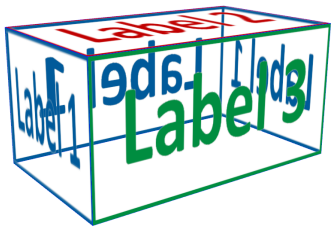


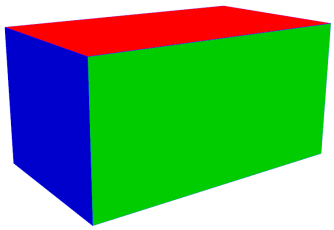
Figure 3.12: The different modalities of splits in the case where $axis$ is X. For visualization purpose, output primitives with different symbols are represented with different colors. Notice how the mirrors results in axis inversions.

A complementary operator, called $Extrude[d|s]$ turns any planar primitive into a volume. The applications of both operators are demonstrated in Figure 3.13.

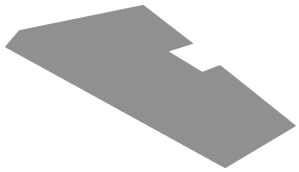
In addition, operators based upon the notion of straight skeleton allow to produce more complex geometry. The straight skeleton can be viewed as an efficient tool to shrink (or expand) a polygon. A polygon is shrunk by sweeping all of its edges along their normals (possibly with a given velocity per edge if the straight skeleton is weighted). This kinematic process, illustrated in Figure 3.14, defines a one-parameter family of polygons. The main difficulty lies on the efficient computation of the polygon associated to a given shrinking factor. Indeed, progressive shrinking implies different types of events (vertex collisions and edge splits) which require the polygon topology to be updated. Remarkably enough, the positions of these events can be computed formally by reasoning on generalized bisectors. The straight skeleton, described in the case of constant velocity in [Aichholzer 1995], is a mere polygon that records the succession of events. This work studied the mathematical properties of the straight skeleton and proposed to derive roofs model from it. It was also used as a basis to roof modeling in [Eppstein 1999], where constant slope is assumed. In order to generate roofs with different slopes, a slight generalization of the straight skeleton, where varying velocity can be associated to edges. For example, the two roofs depicted in Figure 3.14 present vertical sloped parts. This is a typical configuration when dealing with sequences



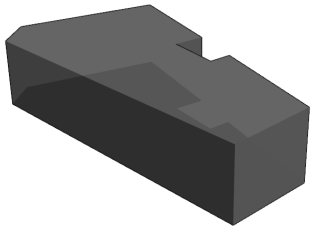
(a) Input labeled mesh



(b) $Facetize$ output



(c) Input planar mesh



(d) $Extrude$ output

Figure 3.13: First row: (a) the input primitive has labels attached to its faces, (b) each face is turned into a primitive which symbol (depicted via a color) depends on the label of the face. Second row: an polygonal planar face (c) is extruded along its normal (d).

of adjacent buildings.

Other appearance operators

In addition to the definition of the scope and the geometry, appearance is characterized by reflectance properties. In CPA, an operator ($Colorize[h|s](.)$) permits to specify the hue component of a primitive. It was mainly intended to enhance illustrations. Nevertheless more advanced procedural techniques relative to textures could be implemented. For instance, the cellular texturing techniques presented in [Legakis 2001] would definitely integrate nicely here.

At last, the $Insert[](.)$ impacts both geometry and reflectance properties in a static way. This operator uses string parameters to specify these properties. The geometry can be one of the following types: square, cube, cylinder, sphere. Other basic types could enhance this limited set. Both geometry and reflectance can refer to external files as well. For instance, to integrate a door model designed externally, the operator should be called as: $Insert[geom =$ 'file:path/door.obj'$|door]$. Alternatively, the door can be approximated by a brown cube. In this case, the operator becomes $Insert[geom =$ 'cube', $color =$ 'brown'$|door]$.

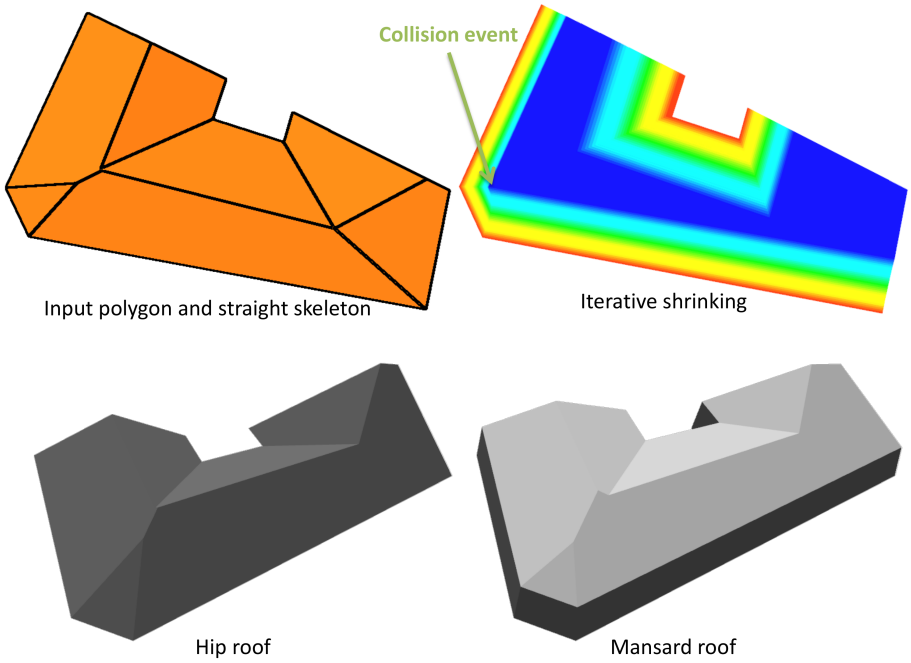


Figure 3.14: First row: left, an input polygon with a weighted straight skeleton - right the edge sweeping process is depicted by showing the shrunk polygons on top of each others with color coding corresponding to the shrinking factor. Second Row: a hip roof and a mansard models obtained thanks to operators based on the straight skeleton. Note how different velocities induce slopes of different angles (in particular a null velocity creates a vertical slope).

Boolean operators

Boolean operations are very common in modeling. In our framework, we consider two operators of such type: a union operator denoted by $Union[s_1, \ldots, s_m]$ and a conditional jump operator denoted by $Switch[cond_1, \ldots, cond_m | s_1, \ldots, s_m]$. The union operator creates m primitives while the switch one produces only the one corresponding to the first positive condition. These two operators contribute to the compactness of grammar specifications. In both operators, the output primitives share the exact same appearance as the input, unless other operators are nested.

Tagging operators

Last, the tag properties have never been considered in any operator. Besides, while the use of consistency tags has been explained, this of differentiation tags has not. The reason for that is merely that the latter is intended to provide an additional property to distinguish primitives of same

Operator signature	modified properties
$Move_{X\|Y\|Z}[d\|s]$ $Rotate_{X\|Y\|Z}[\alpha\|s]$ $Scale_{X\|Y\|Z}[d\|s]$ $Split_{X\|Y\|Z}[d_1,\ldots,d_m\|s_1,\ldots,s_m](p)$ $Repeat_{X\|Y\|Z}[d\|s](p)$ $Mirror_{X\|Y\|Z}[d\|s](p)$ $Mirror_{X\|Y\|Z}[d,d'\|s,s']$	scope
$Facetize[(l_1,s_1),\ldots,(l_m,s_m)]$ $Extrude[d\|s]$ $Shrink[d\|s]$ $Hip[\|s]$ $Mansard[d,\alpha\|s_{loft},s_{top}]$	geometry
$Colorize[h\|s]$	reflectance (hue)
$Insert[params\|s]$	geometry and reflectance but not scope
$Ctag[c\|s]$	consistency tag
$Dtag[d\|s]$	differentiation tag
$Union[s_1,\ldots,s_m]$ $Switch[cond_1,\ldots,cond_m\|s_1,\ldots,s_m]$	None

Table 3.9: Summary of the operators defined in CPA. Note that the parameters d and α in the mansard operator stand for the height and slope angle of the loft (*i.e.* the lower part of the roof).

symbols in the condition of a rule or in the switch operator. Contrary to c, d is not reset to a default value every time it is not specified, but inherited. This allows us to impact on the course of the derivation in the long run. A typical use case will be demonstrated in Figure 3.19.

Despite the fundamental difference between the two tags, similar operators are needed to process their values. They are denoted by $Ctag[c|s]$ and $Dtag[d|s]$, where c and d are numerical parameters defining the value of the consistency and differentiation tags.

3.3 Grammar Examples

We have described the procedural framework implemented in CPA. We will now demonstrate how to use it in order to model buildings of different architectural styles. In particular, a grammar explaining Haussmannian typology of architecture will be of central interest in the thesis.

3.3.1 HLM Towers

HLM stands for "Habitation à Loyer Modéré" and refers to housing with subsidized rent developed in France in response to the post-war crisis. The minister of urban development at the time

r_1	$\omega =$ **FP**	$\rightarrow$	$Scale_Z$[**Vol**]
r_2	**Vol**	$\rightarrow$	$Facetize$[**Fac**]
r_3	**Fac**	$\rightarrow$	$Union[opFence, opFacade]$
			$opFacade = Split_Y$[**Gf**, $Split_X$[**A**, **B**, **C**], **Cornice**]
			$opFence = \ldots$
r_4	**A**	$\rightarrow$	$Extrude$[**Wall**]
r_5	**B**	$\rightarrow$	$Repeat_Y[Repeat_X$[**Btile**]]
r_6	**C**	$\rightarrow$	$Repeat_Y[Split_X$[**Wall**, **Ctile**, **Wall**]]
r_7	**Btile**	$\rightarrow$	$Union[Scale_Y$[**Balcony**], $Extrude$[**Bwin**]]
r_8	**Ctile**	$\rightarrow$	$Split_Y[Extrude$[**Cwin**], **Wall**, $Extrude$[**Cwin**]]
r_9	**Bwin**	$\rightarrow$	$Facetize[\ldots[opGlass]\ldots]$
			$opGlass = Insert[vshader =$ 'file:reflect.vp',
			$fshader =$ 'file:reflect.fp'|**Glass**]
r_{10}	**Balcony**	$\rightarrow$	$Split_Y$[**Hbar**, $Repeat_X[\ldots$[**Vbar**]$\ldots]$, **Hbar**]

Table 3.10: Symbolic specification of a HLM tower grammar. Note that to get concise notations, numerical parameters were dropped and dots (. . .) sometimes replace cumbersome operators. Also for better readability, symbols are shown in bold fonts.

was Eugène Claudius-Petit. He encouraged the development of massive constructions with public financial support. Such low-rent buildings are now widespread in suburban area in France.

Such type of architecture was driven from cheap and massive construction flows and therefore, HLM are particularly suitable to procedural modeling. Ideally, apartments being identical to one another, and buildings are replicated with small variations over large residences where reusability is obviously a desirable feature. Internal repetition of flats has resulted in strong repetitive patterns at the facade level. Table 3.10 presents a simple grammar composed of 10 rules, modeling a HLM-like building template. Figure 3.15 shows few models at progressing stages of the derivation. Note that the final model is composed of cubes only, which means that we did not make use of geometric primitives modeled externally.

3.3.2 Haussmannian Architecture

The previous example illustrates the capability of procedural modeling with respect to buildings. However, HLM is a fairly simple type and will not be the architecture of interest in this thesis. Therefore, limited effort was invested in studying various examples and in establishing a grammar capable at very fine level to generate HLM buildings. On the contrary, the Haussmannian architecture, that is more complex and more frequent in Paris center, has drawn our attention.

Architectural-wise, modern Paris finds its roots in the Second Empire that is to say under the reign of Napoleon III. Rebuilding of Paris was initiated immediately after French revolution and in particular after the cataclysmic cholera epidemic in 1832. Paris districts were insanitary due to the lack of air circulation, of water provision and efficient waste evacuation. It was argued that the consequences of cholera would have been diminished if the above issues were addressed which could have been feasible if large avenues replaced the current narrow street network. This would also

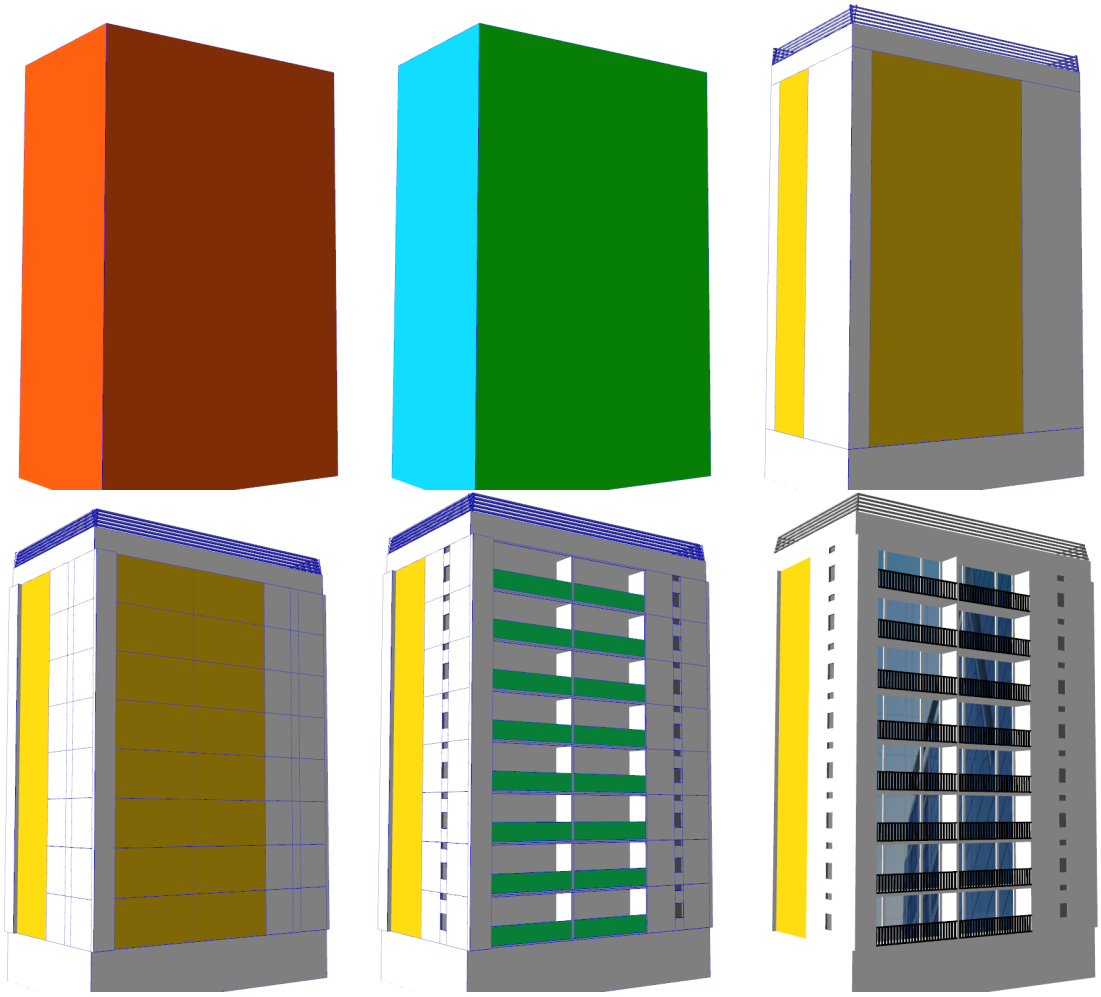


Figure 3.15: Few selected derivation steps of a grammar for HLM towers.

improve traffic in the city. Prefect Rambuteau created a 13 meters wide street in the center of Paris. But due to the limited power granted to the administration, no true urban program was developed. On the contrary, once self-declared Emperor of France, Napoleon III had all latitude to settle down a huge expropriation program allowing a much faster evolution of Paris landscape. Napoleon III's purpose was beyond sanitary issues and better circulation of troops was definitely at stake. Baron Georges Eugène Haussmann was named prefect of the Seine to execute a visionary renovation plan from 1852 to 1870. Beside the practical aspect of expropriation, public regulations enforced new aesthetic criteria. One can cite limits on the height of the building depending on the width of the street, specifications about roof angles. Systematic alignments were also part of Haussmann's

Figure 3.16: A typical building from the early Haussmannian period.

principles. As such, neighboring buildings were to share common facade planes and to present aligned floors. Last, the use of quarry stone was compulsory. This urban plan was quite general as it included a redefinition of Paris districts, a sewer system, and green spaces management[1]. We refer to [Pinkney 1958] for an in-depth description of Haussmann's contributions.

The principles introduced by Haussmann were often criticized by the next generations of architects and town planners. Having said that, such an architectural type is of great interest for procedural modeling due to an amazing combination of its repetitive nature, complexity and facade variation. Haussmannian buildings are also widespread in Paris and in other cities of France. Consequently, we will define two grammars that are the basis of image-based reconstructions presented in the two last chapters. The first grammars works with 2D primitives (rectangles) while the other use the full 3D power of CPA. Apart from this difference, they both allow to generate random layouts of semantic primitives. The obtained layouts are controlled by the choices made on rules applied during the derivation and the corresponding numerical parameters. It is crucial to

[1]— One can think of the generalization of tree alignments along the avenues as one aspect (a particularly annoying one when collecting data).

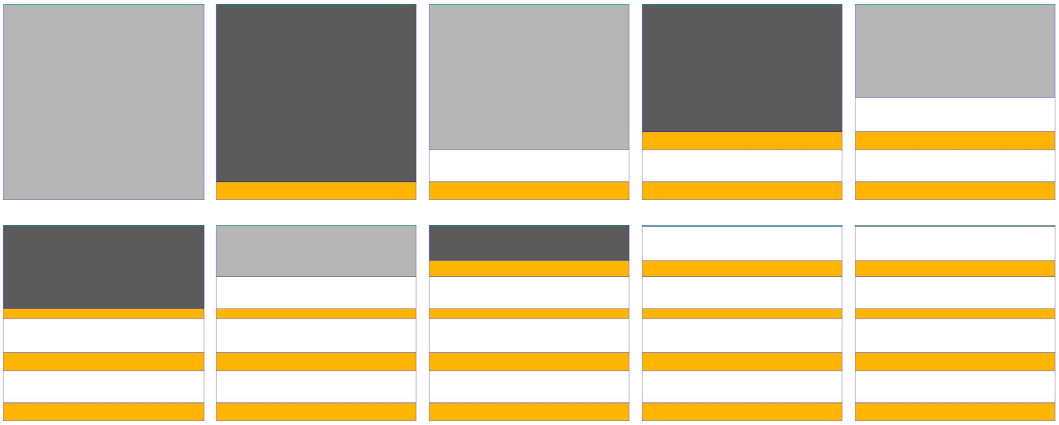


Figure 3.17: Alternating patterns obtained thanks to mutually recursive splits. The figure shows the different stages of the recursion.

understand that these two sources of variability will be later the object of inference.

2D modeling of facade layouts

Let us first introduce the 2D Haussmannian grammar in order to facilitate the reader's understanding. It inherits simplicity (compared to the 3D one) and allows natural illustration of many aspects mentioned earlier such as the consistency and differentiation tags. The grammar is presented in Table 3.11. It is made of 12 rules, 9 non terminal symbols and 7 terminal ones. There are 3 situations where a choice must be taken about which rule to apply, namely when the parse node is among the following symbols: **`Fac,Gf,Fl`**. Besides most of the rules imply one or two variable parameters (denoted as $h_{\texttt{symbol}}$ and $w_{\texttt{symbol}}$ in the table). Let us review few specific techniques that we use in this grammar.

Binary recursive split is a technique that combines two mutually recursive rules in order to generate alternating patterns obtained with a classical split, but without forecasting any upper bound of the number of expected elements. This technique is performed in the following pairs of rules: $\texttt{r}_6$, $\texttt{r}_7$ and $\texttt{r}_{10}$, $\texttt{r}_{11}$. The successive steps of this pattern creation are depicted in Figure 3.17 in the case of the rules $\texttt{r}_6$, $\texttt{r}_7$. Starting from a primitive of symbol $\textbf{Fls}_1$ and applying in turn $\texttt{r}_6$ and $\texttt{r}_7$, a succession of primitives **`Wa`** (orange) and **`Fl`** (white) is generated. Note that a recursion is created because each rule generates the left-hand-side of the other, respectively $\textbf{Fls}_1$ (light gray) and $\textbf{Fls}_2$ (dark gray).

Consistency tags use is demonstrated in rules $\texttt{r}_8$ to $\texttt{r}_{10}$. The goal is to avoid situations depicted in Figure 3.18 (b), where windows are vertically misaligned. A simple solution could impose align-

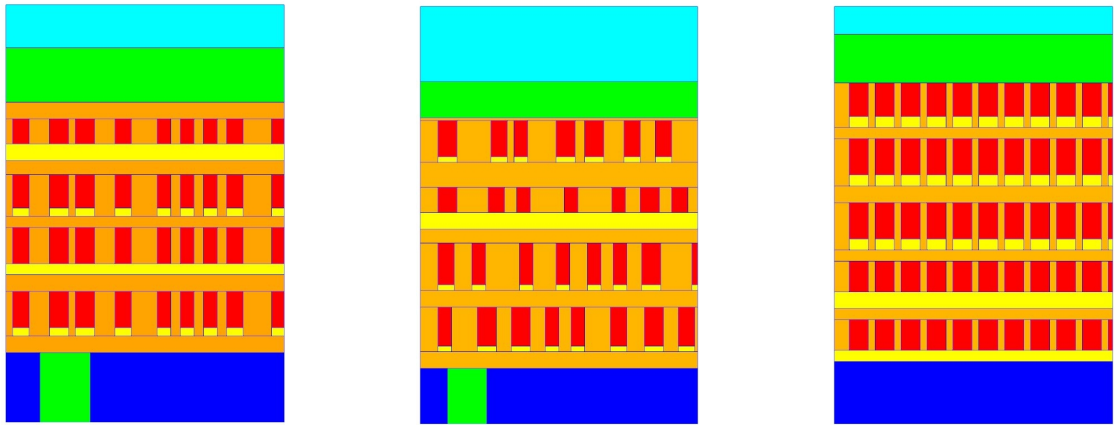
$\mathtt{r}_1$	$\omega = \mathbf{Im}$	$\rightarrow$	$Split_Y[h_{\mathtt{Fac}}, h_{\mathtt{Rf}} \vert \mathbf{Fac}, \mathbf{Rf}, \mathbf{Sky}]$
$\mathtt{r}_2$	$\mathbf{Fac}$	$\rightarrow$	$Split_Y[h_{\mathtt{Gf}} \vert \mathbf{Gf}, \mathbf{Fls1}]$
$\mathtt{r}_3$	$\mathbf{Fac}$	$\rightarrow$	$Split_Y[h_{\mathtt{Gf}} \vert \mathbf{Gf}, \mathbf{Fls2}]$
$\mathtt{r}_4$	$\mathbf{Gf}$	$\rightarrow$	$Split_X[w_{\mathtt{Sh}}, w_{\mathtt{Do}} \vert \mathbf{Sh}, \mathbf{Do}, \mathbf{Sh}]$
$\mathtt{r}_5$	$\mathbf{Gf}$	$\rightarrow$	$Insert[\mathbf{Sh}]$
$\mathtt{r}_6$	$\mathbf{Fls}_1$	$\rightarrow$	$Split_Y[h_{\mathtt{Wa}} \vert \mathbf{Wa}, \mathbf{Fls}_2]$
$\mathtt{r}_7$	$\mathbf{Fls}_2$	$\rightarrow$	$Split_Y[h_{\mathtt{Fl}_1} \vert \mathbf{Fl}, \mathbf{Fls}_1]$
$\mathtt{r}_8$	$\mathbf{Fl}$	$\rightarrow$	$Split_Y[h_{\mathtt{Bc}} \vert \mathbf{Bc}, Dtag[0 \vert Ctag[0 \vert \mathbf{Fl}_1]]]$
$\mathtt{r}_9$	$\mathbf{Fl}$	$\rightarrow$	$Ctag[0 \vert \mathbf{Fl}_1]$
$\mathtt{r}_{10}$	$\mathbf{Fl}_1$	$\rightarrow$	$Split_X[w_{\mathtt{Wa}} \vert \mathbf{Wa}, Ctag[(lhs.c+1) \vert \mathbf{Fl}_2]]$
$\mathtt{r}_{11}$	$\mathbf{Fl}_2$	$\rightarrow$	$Split_X[w_{\mathtt{Tl}} \vert Ctag[0 \vert \mathbf{Tl}], Ctag[lhs.c \vert \mathbf{Fl}_1]]$
$\mathtt{r}_{12}$	$\mathbf{Tl}$	$\rightarrow$	`if` $lhs.d \neq 0$`:` $Split_Y[h_{\mathtt{Bc}} \vert \mathbf{Bc}, \mathbf{Wi}]$ `else:` $Insert[\mathbf{Wi}]$

Table 3.11: A 2D grammar for Haussmannian facade layouts. The value of the variable parameters (h's and w's) will be generated at random during the derivation. The symbols are again in bold font, with a distinctive color when the symbol is terminal. We will keep the same semantic color code in the remainder of the manuscript.

ment by applying the same rule and parameters every time we have to derive a given non terminal symbol. This would result in configurations as shown in Figure 3.18 (c) and in lack of realism since it imposes too much regularity in the layout. Thanks to the combination of numerical expression in the $Ctag$ operator, a tag counter initialized to 0 at every floor (in $\mathtt{r}_8$ or $\mathtt{r}_9$) is incremented every time a $\mathbf{Fl}_1$ primitive is derived. Therefore the width parameters are correlated only vertically, which leads to configurations such as Figure 3.18 (a).

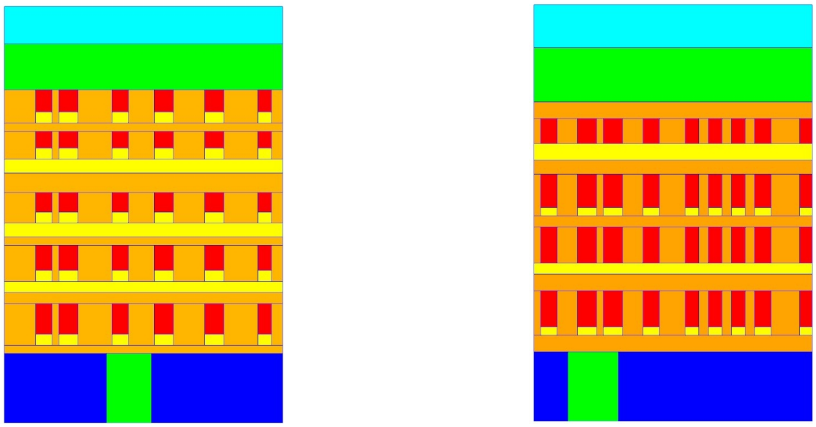
Global variable parameters can be useful when a parameter must be fixed in all different contexts. For instance, in order to enforce that all the windows have the same width, we can define $w_{\mathtt{Tl}}$ as such, a constraint that is not always true in practice. If a global parameter is defined, it will be notified with a g as exponent, as for instance, $w^g_{\mathtt{Tl}}$.

Differentiation tags role is illustrated in Figure 3.19. Their role is to adjust the derivation of tiles ($\mathbf{Tl}$) depending on whether they belong to a floor with a running balcony or not. This information can be propagated along the derivation process since rule $\mathtt{r}_8$ applies the $Dtag$ operator to the generated floor ($\mathbf{FL}_1$). Then, as it was explained earlier, the differentiation tag is inherited in all the primitives deriving from a floor with a running balcony, and when reaching the level of the tiles, rule $\mathtt{r}_{12}$ will replace the tile by a simple window with an optional balcony guard. The option is addressed by a $Switch$ operator, represented in the table by a "`if:  else:`" statement for better readability.



(a) (b) (c)

Figure 3.18: (a) A consistent layout where the windows are vertically well aligned. (b) A non consistent one. (c) A case where the same rules and parameters are kept identical for every node of given symbol.

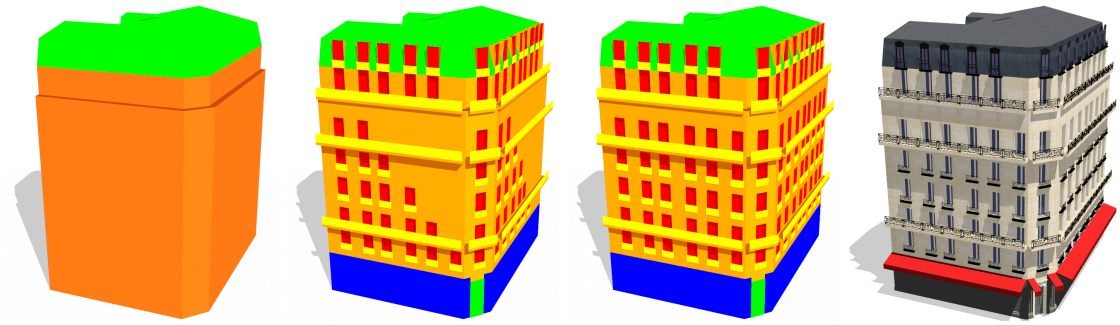


(a) (b)

Figure 3.19: Differentiation as a way to enforce better context control. (a) Without the use of the differentiation tag, every window would be associated with a balcony guard. This does not make sense on floors presenting a running balcony. (b) The situation can be disambiguated by using differentiation tags.

3D modeling of building layouts

The presented 2D grammar which sets the context of Haussmannian facade analysis, is now used as inspiration to derive an extended grammar for 3D analysis. Basically, the 2D grammar com-



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

Figure 3.20: (a)-(c): snapshot of selected steps in the derivation of the 3D Haussmannian grammar as defined in Table 3.12. In (a), a mass model is generated. Then, in (b), a partial derivation of the facade layout is obtained. A complete 3D layout is depicted in (c). The last picture (d) corresponds to a realistic model obtained by extending the grammar with additional rules, replacing the semantic primitives with adapted textured models defined in external files.

prises almost all the necessary components to 3D modeling except two. The two missing elements correspond to the mass model generation and to the depth of the structural elements composing the facades.

The complete 3D grammar is presented in Table 3.12 and we one can observe that it is more complex than the previous one. In order to better demonstrate the process and the principles of the 3D grammar, let us first comment on the snapshots presented in Figure 3.20. The first example shows the derivation stage corresponding to the mass model. It is composed of a volume extruded from the footprint, on top of which a shrunk volume classically called a setback in architectural jargon and a mansard roof. The setback usually contains a single floor and is typical of many architectures including the Haussmannian one. The remaining of the grammar follows almost the same strategy as in the 2D one, with three notable distinctions. Structural elements are extruded in or out to create realistic depth profiles. Then, rules $\mathtt{r}_{15}$ and $\mathtt{r}_{16}$ allow to produce symmetrical horizontal floor configurations. And last, geometric consistency is addressed in a more thorough manner. Two differences arise when comparing with the 2D case, since consistency must be ensured in the facade vertical decompositions, while the consistency tag counter used to enforce window alignments must be reset to a null value only when deriving the first facade. The first condition will guarantee that floors are aligned along the possibly many facades composing the building. On the other hand, the second condition will allow differentiation between the horizontal decomposition of different facades.

r_1	$\omega = \mathbf{Fp}$	$\rightarrow$	$Scale_Z[h^g_{Vol}\|\mathbf{Vol}] + Move_Z[h^g_{Vol}\|\mathbf{At}]$
r_2	$\mathbf{At}$	$\rightarrow$	$Mansard[\alpha^g_{\mathtt{Rf}}, h^g_{\mathtt{Rf}}\|\mathbf{Rf}, \mathbf{Rf}]$
r_3	$\mathbf{Vol}$	$\rightarrow$	$Split_Z[h^g_{SV}\|\mathbf{LV}, Shrink[d^g_{UV}\|\mathbf{UV}]]$
r_4	$\mathbf{LV}$	$\rightarrow$	$Facetize[Ctag[0\|\mathbf{Fac}]]$
r_5	$\mathbf{UV}$	$\rightarrow$	$Facetize[$ $Split_Y[h^g_{\mathtt{Co}}\|Ctag[cnt_{Fac}\|\mathbf{Fl}], \mathbf{Wa}]+$ $Move_Y[Dtag[1\|Ctag[cnt_{Fac}\|\mathbf{Fl}]]$ $]$
r_6	$\mathbf{Fac}$	$\rightarrow$	$Split_Y[h^g_{\mathtt{Gf}}\|\mathbf{Gf}, Ctag[0, \mathbf{Fls1}]]$
r_7	$\mathbf{Fac}$	$\rightarrow$	$Split_Y[h^g_{\mathtt{Gf}}\|\mathbf{Gf}, Ctag[0\|\mathbf{Fls2}]]$
r_8	$\mathbf{Gf}$	$\rightarrow$	$Split_X[w_{\mathtt{Sh}}, w_{\mathtt{Do}}\|\mathbf{Sh}, \mathbf{Do}, \mathbf{Sh}]$
r_9	$\mathbf{Gf}$	$\rightarrow$	$Insert[\mathbf{Sh}]$
r_{10}	$\mathbf{Fls}_1$	$\rightarrow$	$Split_Y[h_{\mathtt{Wa}}\|\mathbf{Wa}, Ctag[(lhs.c+1)\|\mathbf{Fls}_2]]$
r_{11}	$\mathbf{Fls}_2$	$\rightarrow$	$Split_Y[h_{\mathtt{Fl}_1}\|Ctag[cnt_{Fac}\|\mathbf{Flo}], Ctag[lhs.c\|\mathbf{Fls}_1]]$
r_{12}	$\mathbf{Flo}$	$\rightarrow$	$Scale_Y[h_{\mathtt{Bc}}\|Extrude[d_{\mathtt{BC}}\|\mathbf{Bc}]] + Dtag[0\|Ctag[lhs.c\|\mathbf{Fl}]]$
r_{13}	$\mathbf{Flo}$	$\rightarrow$	$Ctag[lhs.c\|\mathbf{Fl}]$
r_{14}	$\mathbf{Fl}$	$\rightarrow$	$Ctag[cnt_{Fl}\|\mathbf{Fl}_1]$
r_{15}	$\mathbf{Fl}$	$\rightarrow$	$Mirror[Ctag[cnt_{Fl}\|\mathbf{Fl}_1]]$
r_{16}	$\mathbf{Fl}$	$\rightarrow$	$Mirror[w^g_{\mathtt{Tl}}\|Ctag[cnt_{Fl}\|\mathbf{Fl}_1], \mathbf{Tl}]$
r_{17}	$\mathbf{Fl}_1$	$\rightarrow$	$Split_X[w_{\mathtt{Wa}}\|\mathbf{Wa}, Ctag[(lhs.c+1)\|\mathbf{Fl}_2]]$
r_{18}	$\mathbf{Fl}_2$	$\rightarrow$	$Split_X[w^g_{\mathtt{Tl}}\|Ctag[0\|\mathbf{Tl}], Ctag[lhs.c\|\mathbf{Fl}_1]]$
r_{19}	$\mathbf{Tl}$	$\rightarrow$	`if` $lhs.d \neq 0$`:` $Scale_Y[h^g_{\mathtt{Bc}}\|\mathbf{Bc}] + Extrude[d^g_{Wi}\|\mathbf{Wi}]$ `else:` $Extrude[d^g_{Wi}\|\mathbf{Wi}]$

Table 3.12: A 3D grammar for Haussmannian architecture. It can be viewed as an extension of the 2D version.

3D modeling of buildings

The introduced grammars intentionally stop their generations to semantic primitives of simple geometries. Such a choice is constrained from the inference process since retrieving automatically the structure of buildings from images with high geometric details is not feasible given standard acquisition means and state-of-the-art inference techniques. However, given a semantic decomposition, it is possible to extrapolate a plausible fine-grain model by replacing the simple primitives with suitable rich exemplars (with respect to geometry and appearance) stored in an external library. Such a task can be achieved by adding supplementary rules to the grammar as illustrated in the last example of Figure 3.20. Note however that from an inference point of view, the additional information would not rely on observed evidence.

Going even further in the seek of realism, one can add rules to complement the model with typical primitives such as carved decorations, cornices, or chimneys. These elements are not present in the initial grammar, but nevertheless, their locations are correlated with the positions of other structural elements. For instance, carved ornaments are always placed on top of windows. This correlation can be encoded in grammar rules. Figure 3.21 shows a district generated in this way.

Figure 3.21: A district generated with a completely developed 3D Haussmannian grammar.

3.4 Conclusion

Procedural modeling was a driving force in computer graphics and was aimed at low-cost generation of random but convincing virtual urban environments in the large-scale. In this chapter, we have shown that procedural modeling shares concepts, principles and practices with shape grammars. Furthermore, we have reviewed its use in computer graphics towards large-scale urban modeling. This overview was followed by a presentation of a procedural modeling framework developed in the context of the thesis. This development was compulsory for practical reasons and necessary to comprehend the subtleties of grammars and procedural modeling. The ability to eliminate third-party dependencies has offered the opportunity to come up with novel extensions such as the consistent derivation scheme, the differentiation property and some novel operators.

From the point of view of the expressive power of this framework, many extensions are possible. For instance, having a representation based on combined B-reps and allowing operators to be defined based on a stack language (as for instance GML) would open new horizons in term of the geometric complexity accessible procedurally. Additionally, we have considered in a rather preliminary state the procedural generation of textures. Many works have been done in this direction and their integration in the current version of the procedural engine is of reasonable difficulty. These limitations can be circumvented by the possibility through the introduction of external models or textures. We are convinced that in the context of our study, the procedural engine was largely sufficient to satisfy our expectations. We have indeed shown models of HLM towers, towards demonstrating the essence architecture modeling using split grammars and their extensions. Then several grammars were described in details to account for different levels of analysis of the central architecture in this work, namely the Haussmannian style. Two grammars were designed specifically in the purpose of deriving structured semantic 2D and 3D layouts. Last but not least, we have presented how these grammars could be extended to generate eyecandy models representing realistic Haussmannian buildings.

Chapter 4

Single View Procedural Reconstruction

In this chapter, we study single view reconstruction using procedural modeling. The objective is - given a grammar and a rectified facade image - to determine the derivation of the grammar and the corresponding parameters towards semantic reproduction of observed facade. The interest of such an approach is to demonstrate the two most valuable characteristics of grammar based techniques with respect to modeling. First, as semantics are attached to the elements generated by the derivation, it is possible to link modeling to symbolic understanding. This is achieved by defining an energy framework which combines the grammar with appearance priors. The resulting optimization problem recasts the grammar derivation as a classification task. The reciprocal aspect concerns the benefit of using grammars to drive this exact same classification task. It lies on the strong context imposed by the grammar with respect to the absolute and relative locations of the various semantic elements in the scene.

The reminder of this chapter is organized as follows. In section 4.1, the general framework will be introduced, where simplifying assumptions are made on the grammar. We will then describe a generic probabilistic energy formulation and the corresponding optimization scheme in section 4.2. The energy will be designed under the hypothesis that distribution models are known which statistically describe the appearance of different semantic classes. The way to obtain these models is discussed of section 4.3. Lastly, we will present quantitative and qualitative experimental results validating the interest of procedural modeling and therefore justifying further exploration along this path.

4.1 General Settings

The modeling pipeline assumes as input a single ortho-rectified and cropped view of a facade (rectification can be achieved using standard automatic techniques [Hartley 2004]). An example of such view is presented in Figure 4.1 (a). Then a simplified version of the 2D grammar presented in the previous chapter is used to generate candidate segmentations of the image domain. Each candidate can be evaluated thanks to an energy described later. A local search algorithm is used to

(a) (b) (c)

Figure 4.1: Input (a) and output (b) of the procedural inference. The 3D model in (c) is obtained by extrapolation of the 2D facade parsing.

explore the procedural space towards minimizing the segmentation energy.

The outcome of the process consists of an optimal derivation of the 2D grammar (see Figure 4.1 (b)). It can be used as a basis for the derivation of a 3D grammar as explained in the previous chapter. New rules are used to create an artificial mass model and to impose credible depths for the elements of the facades. Even though this additional information is based on prior knowledge only, the fact that the created depth discontinuities match the visual information contributes to very realistic models (as the one in Figure 4.1 (c)).

4.1.1 A Simplified Procedural Space

In order to keep the optimization task of reasonable difficulty for automatic inference, we introduce certain modifications to the 2D Haussmannian grammar. Our purpose is to represent the degrees of freedom ruling the derivation as a fixed dimensional real valued vector. We recall that in general space of inference consists of all derivation trees, which are of dynamic size and involve both discrete and continuous parameters.

The adopted modifications to achieve such a simplification are the following.

- The grammar will include a unique rule for each non terminal symbol. In this way, discrete parameters are no longer part of the problem.

- We replace the binary recursive splits with a single split creating many primitives. In order to preserve the expression power of the grammar, we adopt an over-estimate with respect to the

number of primitives. The correct number of primitives is determined through the inference process by allowing zero size primitive placeholders.

- We enforce the derivation to be always consistent. Therefore, consistency tags are always set to the same value such that primitives associated with a given non terminal symbol are always derived with the exact same parameter values.

Such constraints impact the layouts that can be generated with the grammar. Given the rule choices available in the general form of the 2D grammar, the single rule constraint implies that running balconies should occur at certain floors only. This is often the case for Haussmannian buildings where running balconies appear at the second and fifth floors. The replacement of recursive splits assumes an estimate of the maximal number of floors (known in Haussmannian buildings) and windows in the facade. Let us point out also that the last constraint does not enforce regular decomposition of the facade because the horizontal splits are now performed in one step while all the corresponding degrees of freedom (the widths between the windows, and their common width) are fixed independently at the first floor derivation and kept constant for the remaining floors.

Concerning the procedural space, it turns out that facade layouts can be now described with a fixed number of rules. We denote $\pi = (r_1, r_2, \ldots, r_M)$ the sequence of rules or *policy*. Applied on the axiom ω (corresponding to the image domain), the grammar generates a specific building instance. Thus, a facade layout and a policy are two representations of the same object.

Given the axiom ω, we can now define the procedural space of ω given a grammar $\mathcal{G}$ as the set of layouts $\mathcal{L}$ that can be generated with $\mathcal{G}$ starting from ω:

$$\mathcal{L}(\omega, \mathcal{G}) = \{\pi = (r_1, r_2, \ldots, r_M) | M \in \mathbb{N}, r_i \in \mathcal{G}, \forall i\} \tag{4.1}$$

The dimension of the procedural space is:

$$d(\mathcal{L}(\omega, \mathcal{G})) = \sum_{i=1}^{M} DoF(r_i) \tag{4.2}$$

In the case of the simplified 2D grammar, the dimension is 30[1]. If we assume that each parameter lays in a discrete space of cardinality 10, then the cardinal number of $d(\mathcal{L}(\omega, \mathcal{G}))$ is 10^{30}. Note that without the consistency derivation constraint, we can expect different splits for each of the 6 possible floors, and a different small balcony guard in front of each window. Thus the number of admissible layouts grows to $10^{18} \times (10^{11})^6 \times 10^{60} = 10^{144}$, and if we do not force all windows to have the same size, then it becomes 10^{174}. In simple words, without consistency and unless an intelligent pruning process is introduced, the dimension of the procedural space grows exponentially with the depth of the parse trees.

[1]— This value is obtained when assuming 6 floors and 10 windows. Note that the main vertical split introduces 14 unknowns (12 for the wall and floor placeholders and 2 for the running balconies) and the horizontal split 11 (and not 20 since all windows share the same width). The remaining unknowns account for the sizes of other elements (ground floor, shop, door, *etc.*).

Nevertheless, it is observable that the inconsistent layouts represented the great majority of the procedural space before factorization while corresponding to a very small proportion of real buildings. The proposed factorization reduces the dimensionality of the procedural space and imposes to a certain extent architectural logic. The outcome of this process can be seen as an optimal dimensionality reduction of the search space which is able to account for realistic and complex buildings. The grammar still encompasses a huge amount of shapes.

4.2 Exploring the Procedural Space

The previous simplifications yield a facade layout that is equivalent to a specific sequence of M rules, or policy (in our case $M = 8$). The inference task is equivalent to choose the M rules with an optimal fitness to the observed image cues. Once defined an energy function $E(\pi)$ mirroring these cues (see section 4.2.2) the grammar-driven image-based modeling problem can be seen as the following optimization problem:

$$\pi^* = \underset{\pi \in \mathcal{L}(\omega, \mathcal{G})}{\operatorname{argmin}} E(\pi) \tag{4.3}$$

4.2.1 Image Support

In this section, we discuss the process of associating a cost for a given facade layout with an observed image. This is done using conventional classification tools that will be presented in section 4.3. We assume that those classifiers provide the probability for any pixel x to it belong to a given class c, or in other terms to represent a part of one of the terminal shapes of the dictionary (*i.e.* window, wall, balcony, door, roof, sky, shop). Therefore, through those classifiers we have access a probability map for each semantics c:

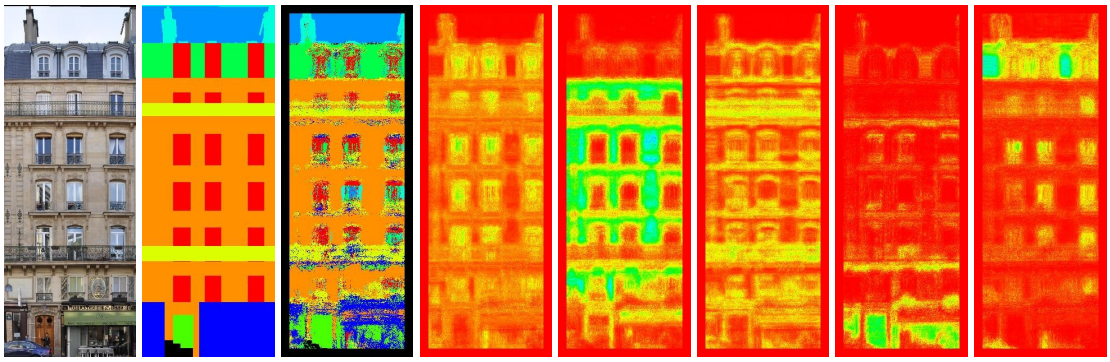
$$p(c|f_x) \tag{4.4}$$

where f_x is a feature vector at location x.

Since the classification process is done locally at the pixel level regardless of any global information, the obtained probability maps are noisy and corrupted by occlusions. Figure 4.2 depicts an example of probability maps obtained using Randomized Forest classifiers (see section 4.3 for details).

4.2.2 Image-based Facade Energy

We are discussing now how to build a meaningful energy that will combine the weak image support with the flexible shape grammar priors. The energy or score function should quantify how appropriate a given policy π is with respect to the image. To compare a policy π to an image, the



(a) image (b) truth (c) result (d) window (e) wall (f) balcony (g) door (h) roof

Figure 4.2: (a) is the original data. (b) is the image of the label defined by hand. Each color corresponds to a label. (c) shows for each pixel the most probable label, in the same color space as in (b). Images (e),(f),(g),(h),(i) shows the probabilities for each pixel to belong to one of the classes *wall, window, balcony, door* and *roof*. The bluer the pixel, the higher the probability, the redder, the lower.

semantic layout defined by the policy is overlaid on top of the image plane. As a result a policy induces a specific segmentation of the image into regions R_s, where s is a semantics such as window, door, balcony, wall, *etc.* (see Figure 4.1).

Given such segmentation, the energy can be defined from the posterior segmentation likelihood. This measure is estimated using the probability maps provided by the classifier. For a region R and a candidate label c , we can compute the probability $p(c|f_x)$ of each pixel x to belong to the class c. The probability of the whole region R to belong to class c can be computed as the joint probability over all its pixels. Assuming conditional independence, the joint probability can be factorized as follows:

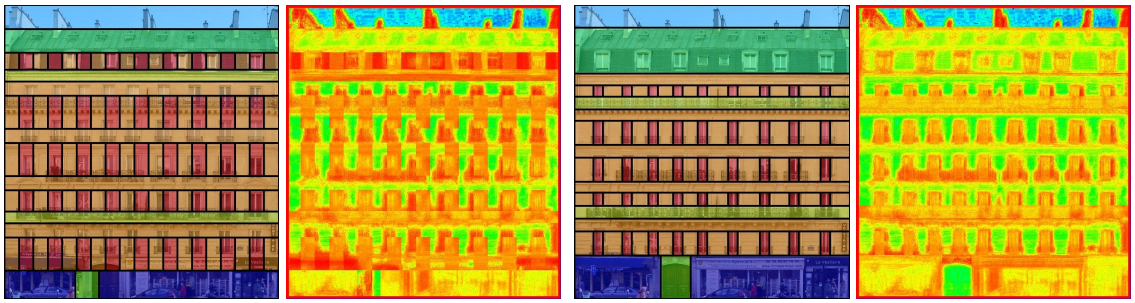
$$p(c|R) = \prod_{x \in R} p(c|f_x) \tag{4.5}$$

We can turn this into an energy through Boltzmann's transformation:

$$E_c(R) = -\sum_{x \in R} \log p(c|f_x) \tag{4.6}$$

Then, the energy of a whole policy π is computed as the sum of the energies of all the regions $\{R_i\}$ (with label c_i) in the segmentation provided by π, adding in the segmentation the sky as what lies above the top of the roof in the image.

$$E(\pi) = \sum_i E(c_i|R_i) = -\sum_i \sum_{x \in R_i} \log p(c_i|f_c) \tag{4.7}$$



(a) High energy building (b) Low energy building

Figure 4.3: Graphical representation of the energy on two examples of buildings over the same image. The first and third images are the projections of the buildings on the image facade. Colors correspond to semantics. We can notice that the building in (b) better fits the image. The second and last images visually sum up the energy of a given configuration. It is a patchwork of probabilities built using the probabilities of each semantics (see Figure 4.2) and the segmentation provided by the current building. One should notice that the low energy building displays colors that tend more towards yellow and green tints while the high energy one is redder (low probability).

For a given image, comparing the energies of two different buildings π_1 and π_2 is meaningful since they are both defined as a sum of positive numbers ($-\log p$) over the same image (see Figure 4.3).

Moreover, the use of integral images [Viola 2001] is an efficient computational tool for such likelihood estimates. Terminal primitives overlaid onto the rectified image are rectangular, and therefore the energy can be estimated from the values of the integral image at their corners. Once the image energy component has been defined, the next task consists in adopting an appropriate search strategy with respect to the feasible facade layouts (*i.e.* those possibly generated using the grammar).

4.2.3 Sampling Grammar Rules

The central idea of a hill climbing approach approach is to generate new samples on-the-fly that are random perturbations of the current state. In our case, we have to deal with grammar rules which are viewed as vectors of ${\mathbb{R}_+}^d$, where d is the number of degrees of freedom of the rule. In order to sample a new rule r around the rule r_0, we basically follow the sampling Equation 4.8, using centered Gaussian laws with a given standard deviation σ.

$$r = r_0 + \sum_j x_j \delta_{ij} \quad where \quad x_j \sim \mathcal{N}(0, \sigma), \forall j \tag{4.8}$$

where i is a given component and $\delta_{ij} \in \mathbb{R}^d$ is the vector which elements are all 0 excerpt for the ith that is 1.

We can re-sample a policy by perturbing all its rules. Since a policy is equivalent to a single building in the grammar, we are able to generate on-the-fly buildings geometrically close to the current one.

4.2.4 The Hill Climbing Algorithm

With the knowledge of how to score a policy, and how to generate new ones, we still have to find out the optimal policy. Since we know the dimension of the problem (see section 4.1.1), and that we allow the parameters to have continuous values, we decide to adopt a random walk approach in the procedural space. This optimization strategy is commonly known as the **steepest ascent hill climbing** algorithm. The approach suggests a rather simple strategy where starting an initial seed, we consider random instances of buildings in the neighborhood of the seed that could better fit the input image. If so, the seed is updated, and we go on exploring the neighborhood of the new one, as specified by the following algorithm:

Algorithm 4.1 The hill climbing optimization algorithm

Require: $n_{max} \sim 100$, $k \sim 1^{e4}$
  Initial seed: $\pi_0 \leftarrow (r_1^0, \ldots, r_7^0)$
  for $n = 1$ to n_{max} **do**
    for $i = 1$ to k **do**
      $\pi_n^i \leftarrow$ Perturbation(π_n)
      compute $E(\pi_n^i), \quad \forall i$
    end for
    $\pi_{n+1} \leftarrow \text{argmin}_i \, E(\pi_n^i)$
  end for
  return $\pi_{n_{max}}$

The initial seed is chosen by using customized average values for the parameters of the rules. These values are obtained from expert assessments and might depend on the considered architectural style. This expert knowledge can be much precise when urban planning regulations exist[2] but these regulations are often subject to evolution in time which moderates their practicality. In any case, this initialization needs not be too accurate in practice. A totally random initialization is possible as well but we found it less robust. At the beginning of the process, we allow the optimizer to search for buildings far from the initial seed. Then, according to the status of the optimization, the search is restricted to smaller neighborhoods, which in practice is achieved by decreasing the standard deviation of the Gaussian law (see Equation 4.8). This is known as the exploitation-exploration trade-off. The more we explore the procedural space, the more knowledge

[2]For instance, for a typical Haussmannian building, the height of the cornice can be initialized to 18 meters.

we have about it and the more we want to use it. However, we still have to explore the procedural space from time to time, in order to avoid falling into local minima.

After about 50 iterations, the algorithm usually converges towards a reasonable minimum (see Figure 4.4]). Section 4.4 shows a large number of results obtained with the presented method.

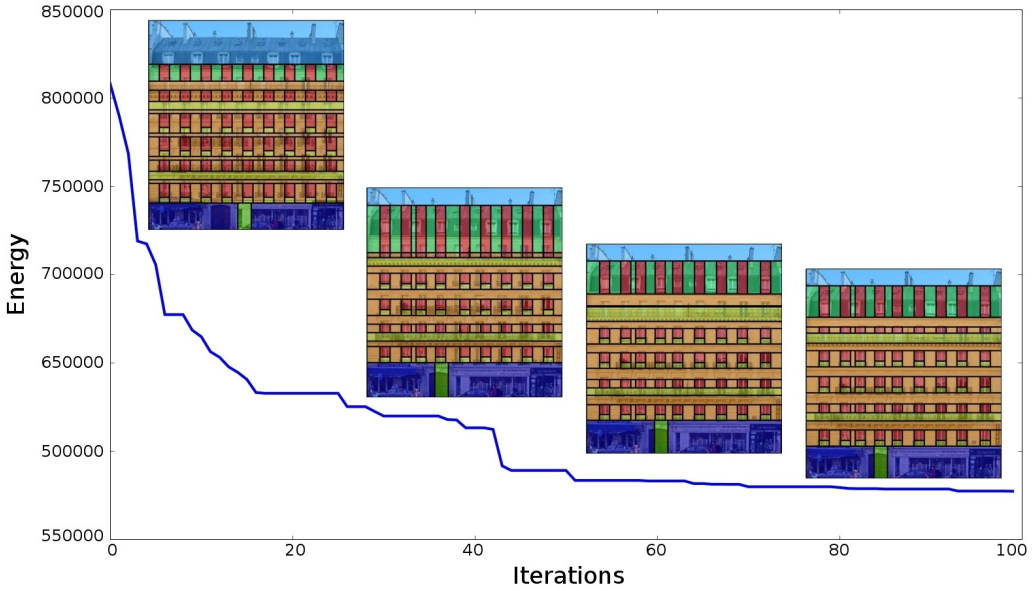


Figure 4.4: Evolution of the energy with the iterations. The energy is decreasing quickly in the first few iterations and then reaches a flat region characterizing convergence.

4.3 Learning the Shape Dictionary

In this section, we discuss two alternatives towards weak image classification as required by the energy formulation presented in section 4.2.2: a supervised method based on randomized forests [Lepetit 2006] and a user-interactive approach [Blake 2004].

4.3.1 Randomized Forests

Randomized Forests [Breiman 2001] are powerful classifiers. They have been applied in a number of problems in computer vision like object recognition [Lepetit 2006], object classification coupled with bags of words techniques [Shotton 2008] or with graph cuts [Winn 2006]. We adopt this machine learning technique towards systematic identification of the facade semantic elements and the image outliers. A randomized forest is used for supervised classification among C classes and

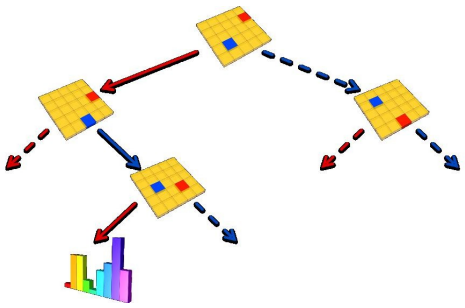


Figure 4.5: Principle of a Randomized Tree. A patch is dropped in the forest. At each node, two random elements are compared(the blue one and the red one). If the value of the red element is greater than the value of the blue one, the patch is sent to the left child otherwise to the right one, where it is further processed. The tree leaves store the probabilities of a patch to belong to the different classes.

is made of a set of T random decision trees. Leaf nodes keep track of the visits of input feature vectors (see Figure 4.5). Internal nodes consist of a simple random test on a feature vector.

Learning a randomized forest classifier

Feature vectors associated to a label are dropped into the tree. Their paths are determined according to the tests being applied at the different internal nodes. After d tests the vector falls into a leaf where the number of visits per label is updated. Thus, each leaf holds a histogram $h = (h_1, \ldots, h_C)$ containing the number of feature vectors associated to each class in the training set which have fallen in there.

One shall notice that in the process exposed above, some of the paths of the tree won't be explored by any of the vectors in the training set. Therefore, in practice, we dynamically create the node of the tree as needed.

Classifying pixels with randomized forests

During the testing phase, an unlabeled feature vector is dropped in each tree of the forest. In a given tree τ, the feature vector falls in the leaf l_τ in which a histogram of labels is stored. Once normalized, this histogram actually provides an estimation of the posterior probability for the feature vector to belong to each class c, given the leaf l_τ in which the patch has ended up:

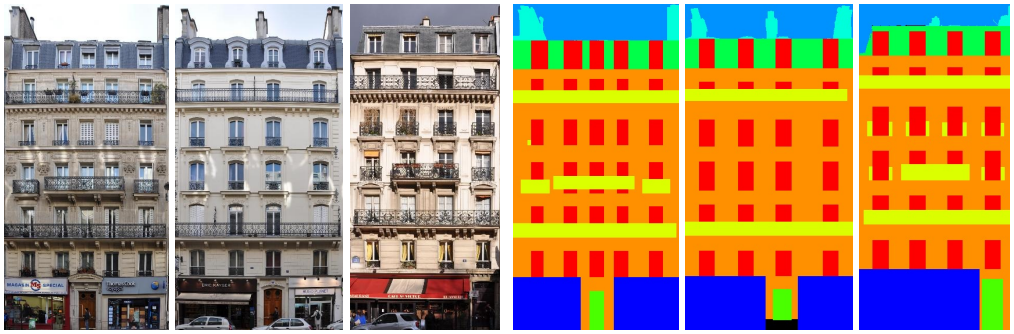
$$P(c|l_\tau) = \frac{h_c}{\sum_i h_i} \tag{4.9}$$

Then, the probability over the forest is obtained by averaging the probabilities of all the trees.

$$P(c|f_x) = P(c|l_1, \ldots, l_T) = \frac{1}{T} \sum_{\tau} P(c|l_\tau) \tag{4.10}$$

Typically, a randomized forest is made of around 10 trees. In our case, we want to classify all the pixels of an image. The feature vectors we consider are symmetric patches of a certain size. This size and the depth of the decision trees are discussed in section 4.3.1. Ultimately, the decision tests can be of two kinds: comparing the values of two pixels of the patch, or comparing the value of a pixel with a random threshold.

Training Set for Architecture



(a) Rectified training data (b) Ground Truth labels

Figure 4.6: Three training samples from the Ecole Centrale Paris Facades Database. Original data and preprocessing of the data: labeling of the basic shapes. Each color corresponds either to a basic shape of the dictionary or to another part of the image, such as the sky for instance.

Once a tool for learning the appearance of terminal elements has been set up, one has to train these classifiers with respect to a given architecture. Indeed, since we feed the randomized forests classifiers with visual examples of grammar elements (*basic shapes*), it is important to carefully consider the training data itself. Obviously, it is unrealistic to seek universal classifiers since the appearance of architectural elements can vary a lot from a style to another. More wisely, we restrict ourselves to train classifiers with examples from a specific architecture or a group of similar ones. Indeed, some different styles may still share basic elements such as windows, balconies or the stones used to make the walls. In these cases, only the way these elements are combined will differ from one style to another (heights of floors, rhythm of windows).

Given that the target architecture is the Haussmannian, we have collected and annotated a dataset that we named the Ecole Centrale Paris Facades Database. It is made of about 150 facade pictures taken in the 5^{th} district of Paris, mainly in Monge street and Soufflot street. Among them

we keep 20 pictures that are used as training set. The pictures are rectified in a preprocessing step (see Figure 4.6). For each image, we label by hand the various semantic elements of the facade : windows, walls, balconies, doors, roof, shop, sky, chimneys and outliers. Images were taken at different lighting and weather conditions.

Empirical adjustment of randomized forest performances

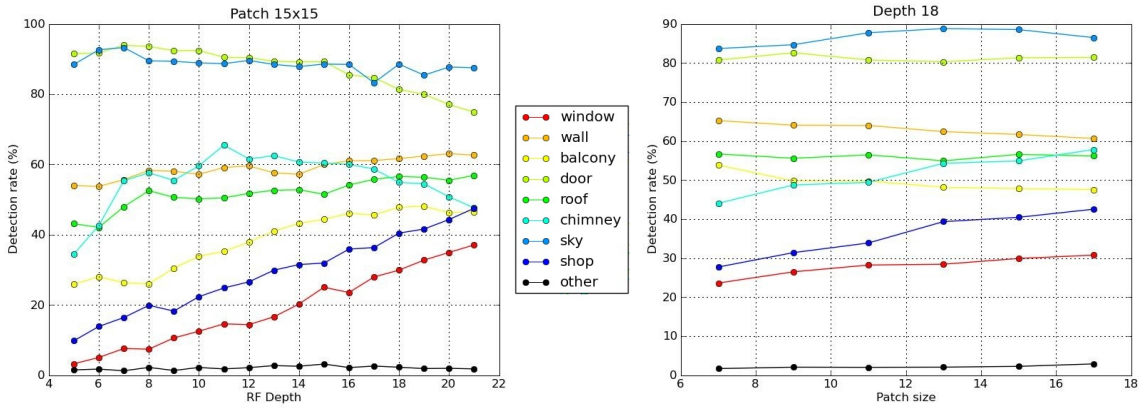


Figure 4.7: Evolution of the detection rate with respect to the depth of the randomized trees, and the size of the patches. The detection rates are mainly sensitive to the depth of the forest.

A randomized forest has three degrees of freedom : the depth of the trees, the number of trees and the size of the feature vectors. According to [Lepetit 2006], 10 trees ensure a good robustness. In order to choose the two other parameters, we basically train forests with depths ranging from 5 to 21, and with patch sizes ranging from 7 to 17. Then we test the trained forest on a data set of 10 new images with known ground truth. For each pixel we attribute the most probable label according to the obtained posterior distribution. Then we compare the classified label with the one given by the ground truth. Figure 4.7 sums up the detection rates for each class. We notice that the size of the patches has almost no impact on the classification, while the detection rates are quite sensitive to the depth of the forest. Especially difficult classes such as windows or shops require more decision tests to be well classified, whereas easy classes such as sky tend to have good results on small depth forests.

As we can see in the first graph of Figure 4.7, the curves are first increasing and then, for some of them, decreasing. This second trend is mostly due to *over-fitting*. A very deep forest will tend to over-fit the training data, making impossible any generalization. The depth for which a forest over-fits depends on the visual complexity of each class. Therefore, deciding a fixed depth is necessarily a trade-off. Since windows are key elements in the building structure, we give more credit to a

decent window detection rate, and therefore decide to use a forest made of 10 trees of depth 18 and with patches of size 15×15.

Finally, we focus on the classification performances for the given forest. For that matter, Figure 4.2 shows the probabilities obtained for each class on a single image. We then build a confusion matrix obtained after applying the classification on the testing set of 10 images mentioned above. On the line i and column j of the confusion matrix is the percentage of pixels[3] in this data set, that belong to the class i according to the ground truth, and that have been classified as j by the Randomized Forest classifier. In the ideal case, the confusion matrix is diagonal.

$$\begin{pmatrix} \mathbf{30} & 11 & 13 & 7 & 12 & 14 & 3 & 9 & 2 \\ 3 & \mathbf{62} & 12 & 4 & 1 & 6 & 1 & 10 & 1 \\ 11 & 9 & \mathbf{48} & 7 & 7 & 4 & 0 & 13 & 2 \\ 1 & 2 & 1 & \mathbf{81} & 0 & 0 & 0 & 14 & 0 \\ 5 & 9 & 6 & 0 & \mathbf{57} & 12 & 9 & 0 & 1 \\ 9 & 14 & 8 & 0 & 12 & \mathbf{55} & 2 & 0 & 1 \\ 1 & 0 & 0 & 0 & 3 & 6 & \mathbf{89} & 0 & 0 \\ 6 & 7 & 9 & 28 & 6 & 1 & 1 & \mathbf{40} & 2 \\ 9 & 10 & 15 & 6 & 14 & 18 & 11 & 14 & \mathbf{2} \end{pmatrix} \begin{matrix} window \\ wall \\ balcony \\ door \\ roof \\ chimney \\ sky \\ shop \\ other \end{matrix}$$

Let's first notice that unsurprisingly the windows are poorly detected (30%), but that the classification is still better than random (11%). Indeed, windows are not visually invariant, and in many cases, reflections or transparency mingle their appearance with other objects. If we consider the columns of the confusion matrix, we can conclude that when an object is classified as *windows*, it is a window 40% of the time in this data set. Conducting finer calculation with the probabilities of equation(4), one can show that pixels that should be labeled as windows according to the ground truth, have an average probability of 24.5% to be a window, whereas pixels from other classes have an average probability of 10.9% to belong to the window class (approximately random). Therefore, the randomized forest are still separating windows from the other classes, even though it does not detect them very well.

4.3.2 Gaussian Mixture Model

Completely supervised methods such as Randomized Forest are very powerful, but require the construction of a proper training set in order to achieve generalization. The creation of such training set can be problematic for two reasons. First, it requires intensive user interaction. In the absence of mechanical Turk, manual segmentation of a large number of images is to be performed. Not only does this process take a lot of time, but it is also user-sensitive. Secondly, the segmentation of unique buildings remains an issue. Indeed, some architectural styles such as the Victorian in United Kingdom or the Haussmannian in France are widely spread and therefore building a relevant training set might be feasible. However, modern buildings do not fit into any style, and the

[3] We insist on the fact that detection rates are computed pixel-wise and not based on whole rectangles as this would be oversensitive to negligible errors.

appearance of the architectural elements is unique. For those buildings, the only way to specify the appearance of the elements is to give some examples of them directly in the considered image.

To address this issue, we use here a simple model of color intensity inspired from [Blake 2004]. We consider that a user has chosen some regions of the image to which he has associated a label: wall, window, etc.

Thus, we have a small training set of N examples in the 3-dimensional RGB color space: $(f_{x_i} = r_i, g_i, b_i)_{i<N}$. We consider that each class c can be explained by a Gaussian Mixture Model (GMM) made of K Gaussian function of $\mathbb{R}^3$. Each component is completely defined by its mean μ and covariance matrix Σ. The number K of component is fixed. In practice, $K = 3$ is a fair choice. Therefore the probability of a feature vector f_x knowing that it belongs to class c_i is given by:

$$p(f_x|c_i) = \sum_{k=1}^{K} \pi_k^i \mathcal{N}(f_x|\mu_k^i, \Sigma_k^i) \tag{4.11}$$

Considering that the parameters π, μ, Σ of the GMM are known for each class, we can then estimate the posterior probability of the class knowing the data in a softmax trend using Bayes' rule:

$$p(c_i|f_x) = \frac{p(f_x|c_i)p(c_i)}{\sum_j p(f_x|c_j)p(c_j)} \tag{4.12}$$

Considering that all the classes are equally probable, and injecting Equation 4.11 in Equation 4.12 we obtain:

$$p(c_i|f_x) = \frac{\sum_{k=1}^{K} \pi_k^i \mathcal{N}(f_x|\mu_k^i, \Sigma_k^i)}{\sum_j \sum_{k=1}^{K} \pi_k^j \mathcal{N}(f_x|\mu_k^j, \Sigma_k^j)} \tag{4.13}$$

The mixture components π_i^j, the means μ_i and covariance matrices Σ_i for each class are estimated using Expectation-Maximization (EM) algorithm to compute the maximum likelihood [Bishop 2006]. To this end, training samples (*i.e.* hand-labeled pixels) are extracted interactively from the image under consideration. Figure Figure 4.3.2 shows some examples of GMM softmax classification as well as the paint strokes used to define the training samples.

For evaluation, we will show results with randomized forest classification when training data are available and Gaussian mixture models otherwise. Of course, these two classifiers are merely given as examples of image support, but one could propose any other method, as long as it provides probability maps better than random. The better these estimations are, and the easier the optimization will be. However, even very noisy and poorly discriminative probability estimations can lead to a very accurate segmentation when combined with procedural shape priors. Next section illustrates qualitatively and quantitatively this strong assertion.

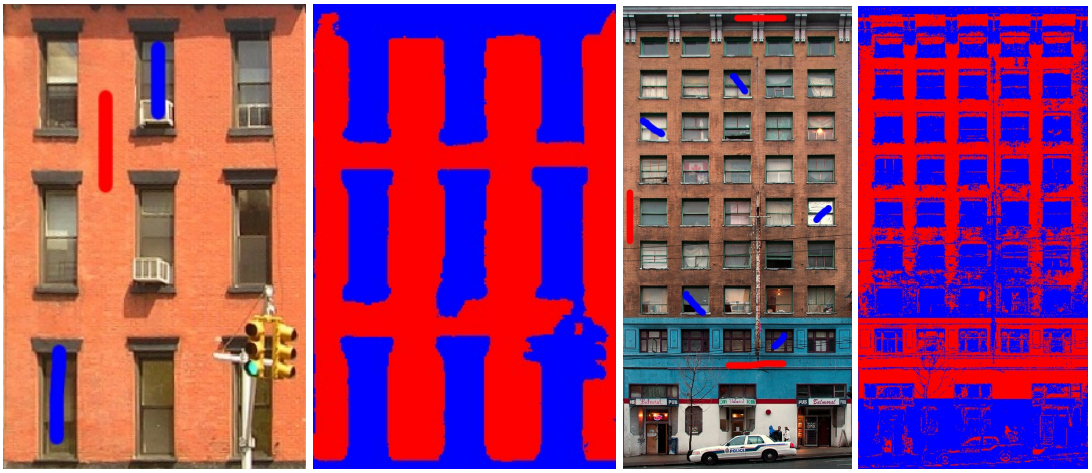


Figure 4.8: Example of GMM softmax classification. The user selects some representative part of the image corresponding to architectural elements (here walls and windows). The parameters of the Gaussian mixture models are learned through EM, and then each pixel is hardly classified using softmax.

4.4 Experimental Results

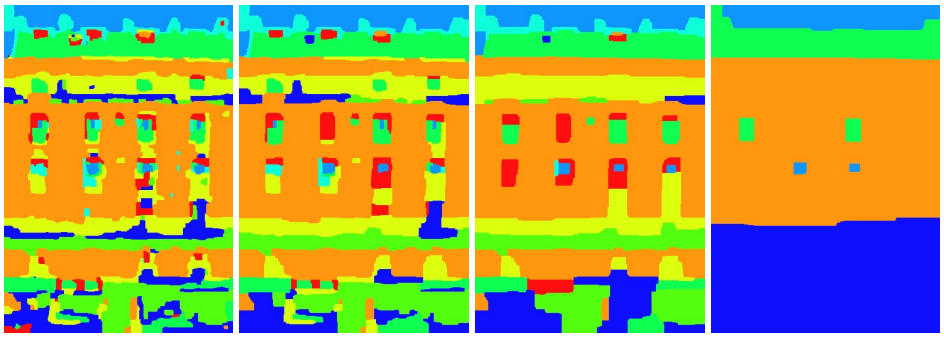
4.4.1 Quantitative Validation

Comparison with RF segmentation

To assess the performance of the procedural layout inference we compare the obtained 2D segmentations with the expected ground truth for a set of 10 test facades belonging to the Haussmannian architecture. The quality of the segmentation is evaluated by computing the confusion matrix over the testing set.

$$\begin{pmatrix} \mathbf{81} & 9 & 6 & 0 & 4 & 0 & 0 & 0 & 0 \\ 5 & \mathbf{83} & 8 & 1 & 0 & 0 & 0 & 3 & 0 \\ 13 & 13 & \mathbf{72} & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & \mathbf{71} & 0 & 0 & 0 & 29 & 0 \\ 8 & 12 & 0 & 0 & \mathbf{80} & 0 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 19 & \mathbf{0} & 75 & 0 & 0 \\ 2 & 0 & 0 & 0 & 4 & 0 & \mathbf{94} & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & \mathbf{95} & 1 \\ 23 & 8 & 3 & 14 & 16 & 0 & 10 & 25 & \mathbf{0} \end{pmatrix} \begin{matrix} \mathbf{+51} & window \\ \mathbf{+21} & wall \\ \mathbf{+24} & balcony \\ \mathbf{-10} & door \\ \mathbf{+23} & roof \\ \mathbf{-55} & chimney \\ \mathbf{+5} & sky \\ \mathbf{+55} & shop \\ \mathbf{-2} & other \end{matrix}$$

In order to evaluate quantitatively the contribution of the grammar and the optimization in the global outcome, we can compare the confusion matrix obtained by the final 2D segmentation with



(a) $\lambda = 0.5$ (b) $\lambda = 1$ (c) $\lambda = 2$ (d) $\lambda = 10$

Figure 4.9: MRF segmentation using Potts Model and Randomized Forest classifiers

the one obtained using the Randomized Forest only. Obviously, we used the same test set as before so that the comparison is fair.

First of all, the confusion matrix is much less noisy than the one obtained in section 4.3.1. This is directly due to the grammar: a region is entirely labeled with the same class. We also notice a great overall improvement compared to the Randomized Forest based classification. Changes are depicted in the right column in green when beneficial and in red otherwise. Since the grammar imposes to the labeling map to be piecewise constant, isolated outliers are easily corrected with their actual labels.

Let us now focus on the most specific classes. First, one cannot fail to notice the significant raise in the window detection rate, reaching a steady 81% detection from a poor level of 30%. However, this fact was to expect as the grammar constrains the windows to be surrounded by walls which are pretty well detected by the Randomized Forest (62%). Hence wall positions are easily caught by the optimization based on the Randomized Forest information, and windows are bound to be well placed by the grammar. On the contrary, the door detection rate decreases of 10% compared to the Randomized Forest classification. Once again the same phenomenon is at stake. Having a closer look at the confusion matrix of the Randomized Forest, we can see, that most of the misclassification of the door are reported on the shops and vice versa. Moreover, the grammar places the doors and the shop as direct challengers. As a consequence, situations occur where after optimizing, the door is entirely placed on a shop spot (see Figure 4.13). Eventually, as the designed grammar does not deal with chimneys, their "detection rate" is obviously zero.

Comparison with Markov Random Field segmentation

One could claim that the discussion above is problematic, since we do not compare ourselves to state-of-the-art segmentation algorithms. Indeed, segmentations obtained with Randomized Forest classifiers are very noisy, since pixels are labeled independently. In order to perform a fair

comparison, we built an Markov Random Field (MRF) segmentation based on the classifiers (see Equation 4.14). The graph is derived from the image grid (4-connectivity), the single potentials are directly obtained from the classifiers ($-log(p)$), and the pairwise potentials follow the Potts model, enforcing labeling smoothness on the optimal segmentation. This model constitutes a fair alternative to the grammar-driven approach since the label of a given pixel depends both on its classification value and on its local neighborhood. The segmentation minimizes the MRF energy given in Equation 4.14, in which we denote by c_i the label associated to the pixel x_i in the image $\mathcal{I}$. The expression $x_i \sim x_j$ means that pixel x_i is linked to pixel x_j through an edge of the graph. In the case where λ is equal to 0, we retrieve the previous RF-based independent classification. Therefore, the Potts model generalizes the RF model.

$$E(c_1, \ldots, c_n) = \sum_{x_i \in \mathcal{I}} -\log p(c_i | f_{x_i}) + \lambda \sum_{x_i \in \mathcal{I}} \sum_{x_j \sim x_i} \mathbb{1}(c_i \neq c_j) \tag{4.14}$$

For different values of the parameter λ, we get different segmentations, becoming smoother when λ increases. Figure 4.9 shows different segmentations of the same image for different values of λ. Here are the confusion matrices for different values of the smooth parameter λ.

$$\underbrace{\begin{pmatrix}
\mathbf{20} & 12 & 14 & 1 & 33 & 7 & 6 & 3 & 0 \\
0 & \mathbf{73} & 9 & 9 & 0 & 0 & 0 & 6 & 0 \\
1 & 2 & \mathbf{65} & 2 & 11 & 1 & 0 & 16 & 0 \\
0 & 0 & 0 & \mathbf{99} & 0 & 0 & 0 & 0 & 0 \\
1 & 2 & 3 & 0 & \mathbf{75} & 8 & 7 & 0 & 0 \\
2 & 0 & 0 & 0 & 2 & \mathbf{94} & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 9 & \mathbf{90} & 0 & 0 \\
4 & 2 & 6 & 30 & 20 & 0 & 0 & \mathbf{35} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0}
\end{pmatrix}}_{\lambda=0.5}
\begin{matrix}
\mathbf{-61} & \textit{window} \\
\mathbf{-10} & \textit{wall} \\
\mathbf{-7} & \textit{balcony} \\
\mathbf{+28} & \textit{door} \\
\mathbf{-5} & \textit{roof} \\
\mathbf{+94} & \textit{chimney} \\
\mathbf{-4} & \textit{sky} \\
\mathbf{-60} & \textit{shop} \\
\mathbf{+0} & \textit{other}
\end{matrix}$$

$$\underbrace{\begin{pmatrix}
\mathbf{23} & 14 & 13 & 0 & 32 & 5 & 5 & 2 & 0 \\
0 & \mathbf{75} & 8 & 10 & 0 & 0 & 0 & 3 & 0 \\
3 & 2 & \mathbf{72} & 3 & 10 & 0 & 0 & 7 & 0 \\
0 & 0 & 0 & \mathbf{99} & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 2 & 0 & \mathbf{77} & 8 & 6 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & \mathbf{99} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 11 & \mathbf{88} & 0 & 0 \\
1 & 2 & 6 & 34 & 21 & 0 & 0 & \mathbf{33} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0}
\end{pmatrix}}_{\lambda=1.0}
\begin{matrix}
\mathbf{-58} & \textit{window} \\
\mathbf{-8} & \textit{wall} \\
\mathbf{+0} & \textit{balcony} \\
\mathbf{+28} & \textit{door} \\
\mathbf{-3} & \textit{roof} \\
\mathbf{+99} & \textit{chimney} \\
\mathbf{-6} & \textit{sky} \\
\mathbf{-62} & \textit{shop} \\
\mathbf{+0} & \textit{other}
\end{matrix}$$

Even if the segmentation is smoother than the direct RF-based segmentation, it is still inaccurate. Numerical comparisons with the procedural segmentation are shown on the right column of the confusion matrices obtained with $\lambda = 0.5$ and $\lambda = 1$. Apart from the identical exceptions

as earlier, the figures demonstrate a strong advantage in favor of the procedural context. Besides this approach lacks in recovering the underlying structure of the facades which was completely expectable. The MRF model enforces local constraints in the neighborhood of a given pixel, whereas the procedural model enforces higher order structural constraints, between pixels of the images that are not necessarily neighbors through procedural shape priors. Such high order constraints cannot be modeled by classic Markov Random Field with single and pairwise potentials unless we concatenate the label space which will lead to the curse of dimensionality.

4.4.2 Qualitative Validation

In this section, we show the diversity of results obtained using the proposed method. The combination of an efficient parametric grammar with a fixed derivation scheme with simple classifiers enabled us to handle many challenging situations.

First of all, the proposed method performs well on the Haussmannian architecture on which the classifiers have been trained. Figure 4.10, Figure 4.11, Figure 4.12, Figure 4.13 show some segmentation and modeling results on buildings in the testing set. This well-spread and typically Parisian style presents a lot of intra-class variations: materials, ornaments, window rhythms, presence/absence of balconies.

Besides, as stressed in Figure 4.14 the method is able to deal with occlusions and illumination variations. Since the geometry is constrained by the shape grammar, if some part of the information is missing, the intrinsic repetitions lying in the grammar rules make the recovery possible. In 4.14, the first result shows a Parisian facade in which the first floor is covered in vegetation. Even if the windows are not appearing on this floor, based on the other floors, the grammar optimization recovers them. The second image shows an example in which another building casts a large shadow on the facade. As stressed in the second column of Figure 4.14, the Randomized Forest classification is very sensitive to shadows and occlusions, yet the grammar-based segmentation remains robust to them.

Furthermore, although the image support has been trained mainly on one architectural style, the proposed method still performs well on other architectures for which the basic elements (the dictionary) are close to the one used for training. This is the case for a large number of architectures. Figure 4.15 shows some results on other Parisian architectures such that Louis XIV(1680) or Restauration(1830). Other styles even further from the one used for training are also handled quite well with a simplified grammar and the same training set. Figure 4.16 shows different results on Greek neoclassic buildings from Heraklio, Greece and Barcelona, Spain. These examples demonstrate the great ability of the grammar to express numerous topologies with a small set of parametric rules and a few primitives.

Architectural styles that are far away from the one encountered in the training set, are dealt with GMM classifiers. Once more, with rather limited user interaction, the proposed algorithm converges towards the optimal sequence of rules. Results of segmentation and image-based modeling are shown in Figure 4.17. In these results, the regions selected by the user are used as a training set to learn a Gaussian Mixture model (GMM) for each semantic element (walls, windows, etc.).

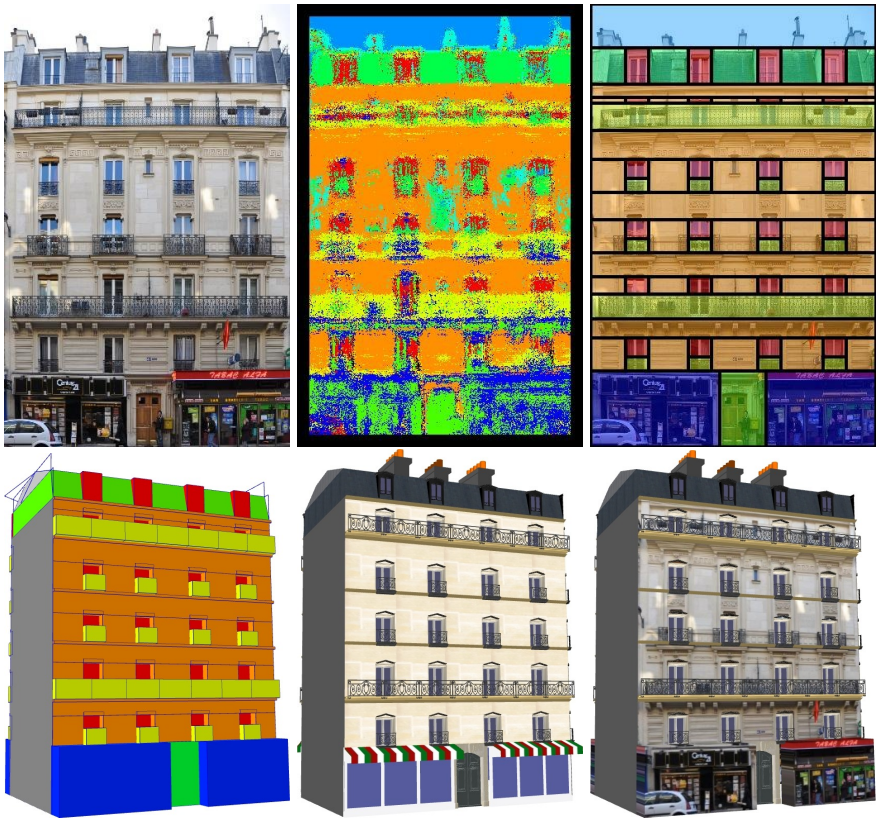


Figure 4.10: Image-based modeling of an Haussmannian building. The first row shows from left to right: the input data, the randomized forest-based segmentation and the projection of the optimized grammar on the input data. The second row gives 3 levels of modeling. On the left one, each terminal shape is represented by a colored cube. On the middle one, the basic elements have a pre-defined geometry. Finally the right one combines the input image as a texture for the facade with some pre-defined 3D models for each terminal basic shape.

One could also imagine to *bootstrap* by recomputing a new GMM per class after a first segmentation, and use it to re-segment the facade again in an iterative loop. In any case, this ability to switch to other techniques of supervised learning makes the method very generic and very flexible. To illustrate this property, we show very different buildings from Paris, Vienna, or New York City.

Last but not least, the proposed methodology lends itself to large scale reconstruction. In that matter, we present in Figure 4.12, the 3D models of a small district composed of ten buildings. The proposed approach is computationally efficient (five minutes per facade) and leads to a model fitted for real time navigation. Indeed, the output is described by a shape grammar and can be converted into a static 3D model of variable complexity (playing on the terminal shape models).

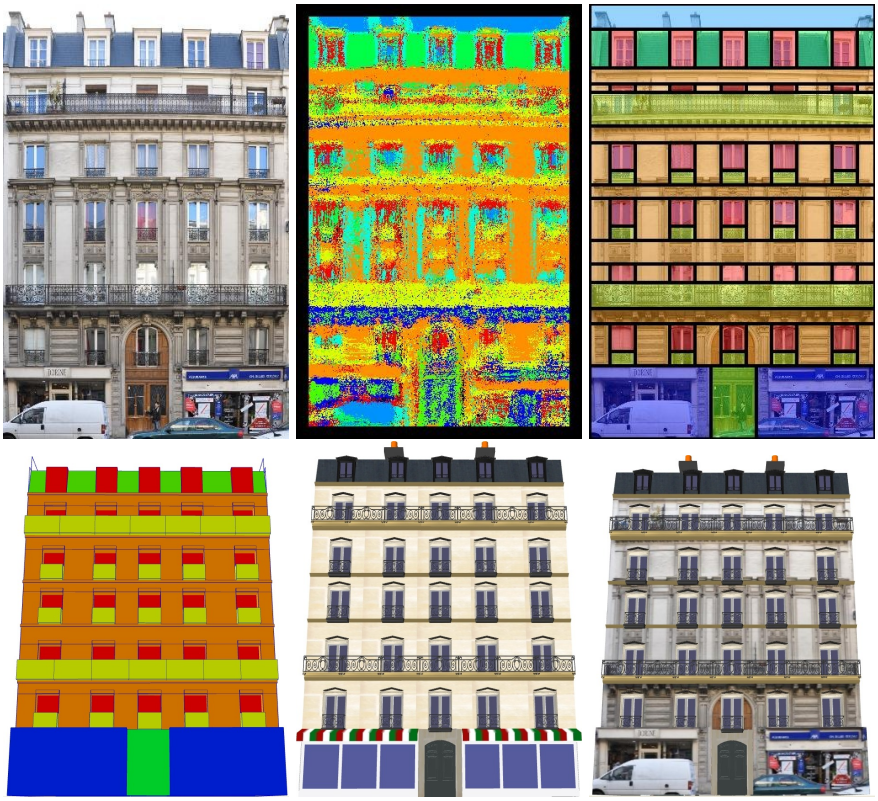


Figure 4.11: Another image-based modeling result on the complex Haussmannian architectural style.

4.5 Conclusion

In this chapter, an approach to procedural building modeling was presented. The input refers to a single ortho-rectified image of a facade which will be segmented according to a 2D split grammar. Based on this 2D parsing a realistic 3D model can be extrapolated.

Such an approach demonstrates that well designed images cues and strong grammar context constitute a very efficient facade parsing tool. This can be achieved by establishing a connection between the terminal semantics of the grammar and images using machine learning techniques. State-of-the art results from qualitative and quantitative view point demonstrated the performance obtained with the procedural approach. In particular, these results have been compared with the ones obtained without any prior and with a Potts smoothness prior. In all cases, the procedural method outperforms the competing method.

Additionally, inference was conducted by following a local search technique. It demonstrated good convergence behavior in this specific context. Qualitative results validates the inference

Figure 4.12: Modeling of a small district made of 10 buildings in rue Monge, Paris.

scheme capability to handle general 2D layout conforming with the grammar. Robustness to challenging illumination conditions and occlusions was considered too, justifying the consistent derivation idea.

Nevertheless, the inference algorithm is not satisfying in all respects. First it requires strong simplifying assumptions on the grammar. Therefore the initial problem, even though it remains intricate, becomes of lower complexity. In particular, attention is to be paid on the simplifications such that the grammar maintains its expressive power. Such a demand becomes almost intractable when considering 3D grammars. And it is even more disputable since to some extents, it cancels the dynamic property of grammars.

An other criticism of the current local search scheme relates to computational efficiency. Indeed, the number of candidate layouts to generate before reaching convergence typically amounts to 1^{e6}. This estimate is expected to increase exponentially if the grammar is less constrained. Besides, speed was not much an issue in this application because of the rectification constraint. Thanks to this constraint, integral images make the computation of the energy very fast.

In conclusion, our first attempt to procedural modeling has demonstrated promising capabilities but in a somewhat simple use case. It opened a new path and highlighted two particular points for improvement. The first one concerns the grammar which must include more general operators so that inference can lead the true 3D structure of the considered building. The other point concerns the inference algorithm, which should be changed for a more efficient one, capable to handle dynamic derivation trees. Both points are considered in the next chapter.

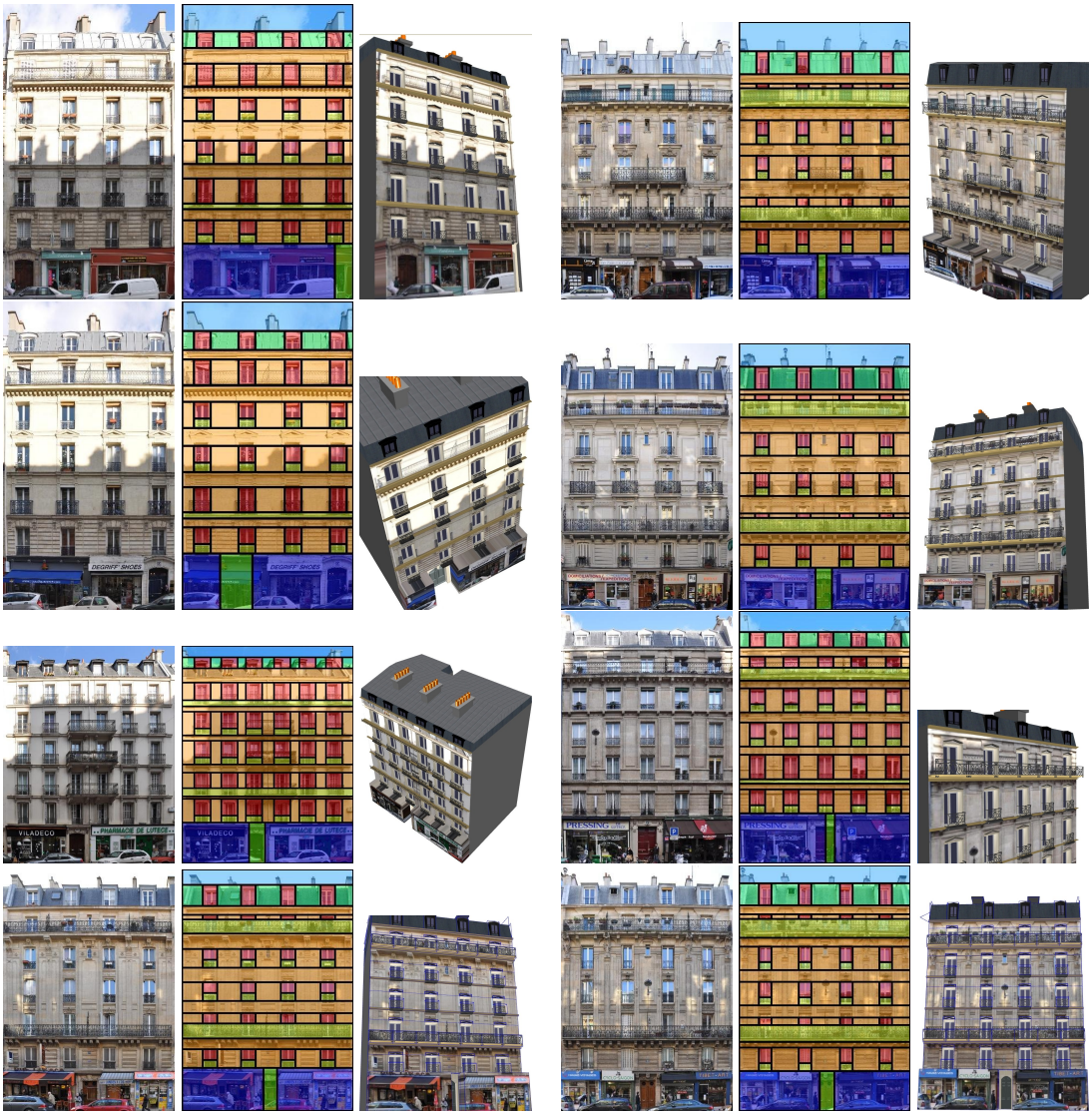


Figure 4.13: More results obtained with the proposed method on various Haussmannian buildings.

Figure 4.14: Accurate building modeling under challenging conditions: illumination variations and occlusions. As shown in the second column, some impressive amount of information is missing. No wall or window are detected by the classifier on the first floor and part of the second one in the second example. However, the proposed method takes advantage of the intrinsic repetitions and regularities of the grammar to recover the missing windows. The information of the upper floor is spread to the lower one thanks to the consistent derivation of the grammar: the same split rule should be applied on all the floors.

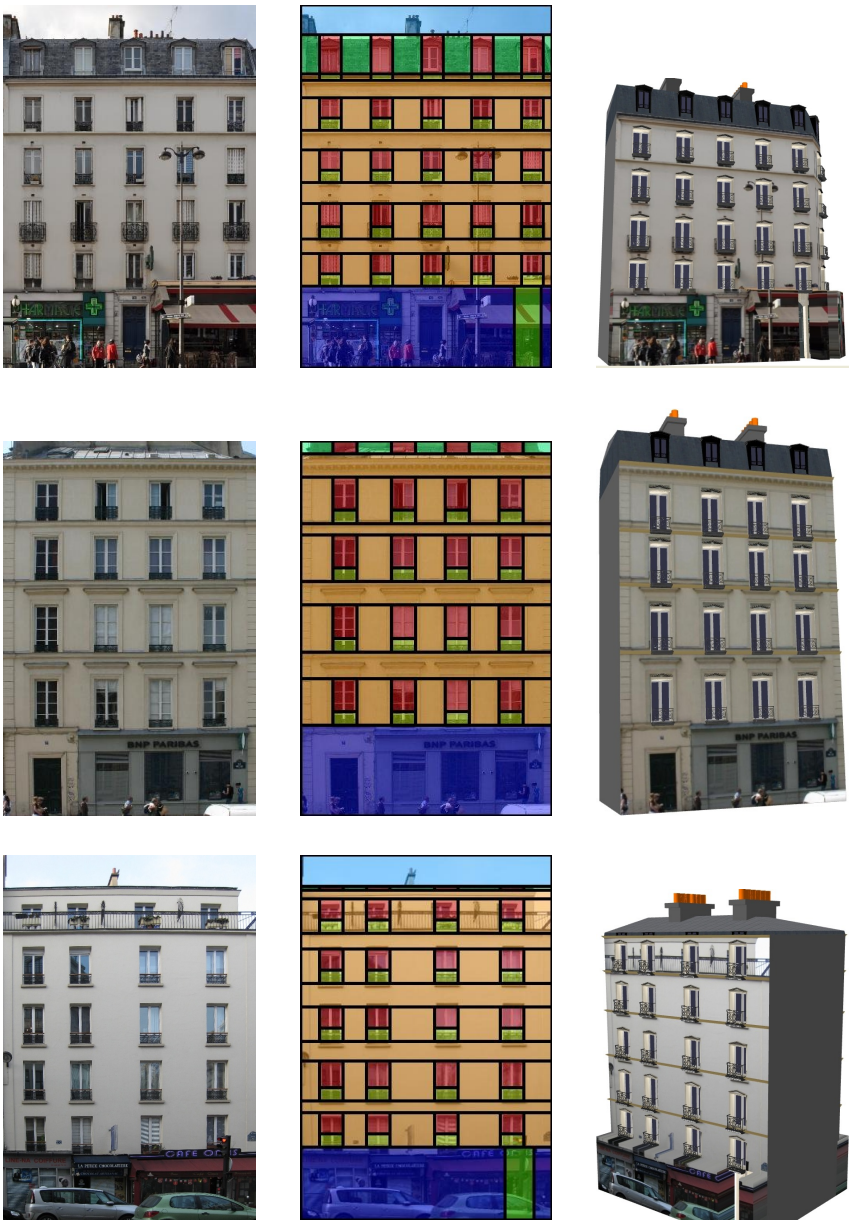


Figure 4.15: The proposed method still performs well on different architectures as long as the visual properties of the basic elements of these new styles are similar with the ones the classifier has been trained with. Here are example of Louis XIV (1680) building (first row), and Restauration (1815) ones (second and third rows).

Figure 4.16: The method is still able to cope with even more different architectural styles, such as Barcelona architecture (first row), or Greek Neoclassic buildings in Heraklio, Greece (second row).

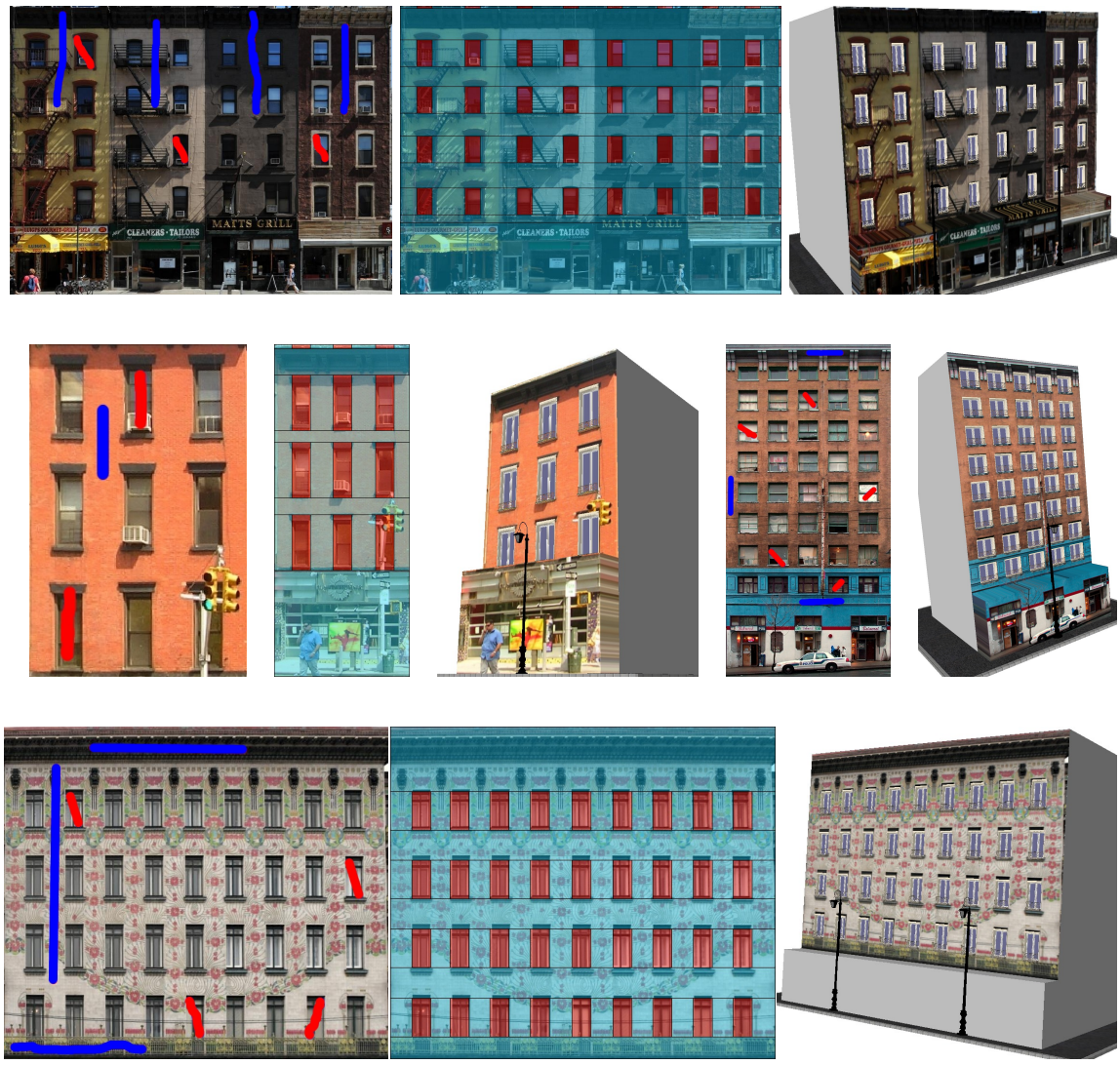


Figure 4.17: Segmentation obtained with a simpler grammar and using the GMM classifier.

Chapter 5

Grammar-based Multi-view 3D Reconstruction

In the previous chapter, we have introduced an approach for 2D grammatical facade analysis. The method explored the appearance of terminal semantics in the grammar. In theory, the same idea can be applied to 3D grammar parsing provided that we know how each primitive projects onto the observed image(s). However, such a task is more challenging because the parse tree gets deeper and includes more parameters. In particular, the behavior of the previous energy would be less discriminative with respect to the depth parameters. Additionally, the generation and the evaluation of hypotheses are costly in 3D. This can be explained by the higher complexity of the 3D rules and the inadequacy of integral images. Eventually, the simplifying assumptions made on the grammar in the previous chapter are not fundamentally necessary.

Based on the above observations, and in order to be able to cope with complex 3D structures and multiple images, we have to improve the computational efficiency of the method and the visual cues which cannot determine real depth. To this end, additional cues will be necessary to infer the depth of the structural elements. In our approach, this will be achieved by using depth maps extracted from the images thanks to standard multi-view techniques. At the same time, previously imposed grammar simplifications are very restrictive and have to be abandoned. Therefore, a novel optimization framework is to be considered which truly accounts for the dynamic behavior of the parse tree while being to reasonable extent computationally efficient.

The remaining of this chapter is organized as follows. In section 5.1, a global overview of the developed 3D grammar-based reconstruction pipeline is presented. Then, section 5.2 will discuss grammar inference, including a review of evolutionary algorithms and their practical use in procedural modeling. Section 5.3 is dedicated to the proper definition of two competitive energy components related to appearance and depth. Experimental results are discussed in section 5.4, while the last section concludes the chapter.

5.1 General Settings

Through out the entire chapter, we shall consider two settings, associated respectively with 2D and 3D analysis.

Single view configuration

Let us begin with the 2D case that shares similarities with the settings of the previous chapter. The main noticeable difference lies on the fact that no constraint will be imposed on the grammar anymore. As a result, we will be able to consider the 2D Haussmannian grammar in its original form (as presented in chapter 3). This will also allow us to to assess the gain provided by the new inference method. Such an assumption that relaxes the finite dimension of the search space will result on a more complex inference. Indeed, the inherent structure to optimize will now be the parse tree itself. The impact of such a choice is not negligible, as in this dynamic structure, a correlation exists between the topology of the tree and the unknowns variables of its nodes. Simply speaking, the number of unknown parameters depends on the values of these parameters.

Besides, the complete optimization problem involves parameters of different nature. The continuous parameters controlling the size of the final elements were already at stake in the previous formulation. However, in the present methodology, we will need to choose which rule must be applied at certain nodes, as for instance to decide whether a running balcony is present or not at a given floor. This new type of variables has a discrete nature. As such, the inference problem involves both continuous and combinatorial optimization aspects. In order to give a glimpse of the complexity, a parse tree for the 2D case is depicted in Figure 5.1. In fact, some branches of this tree have been pruned in order to fit in the page. The complete tree is approximately two times deeper than the one in the figure.

Multi-view configuration

When dealing with 3D procedural reconstruction, two other aspects will change. In order to compute depth cues, the input consists of a sequence of N images $I = I_1, \ldots, I_N$ instead of a single one. The associated cameras are automatically calibrated using the structure-and-motion tool Bundler [Snavely 2006a], and then a dense point cloud is generated using the PMVS tool [Furukawa 2008]. In the rest, we will denote $\pi_1, \ldots, \pi_N$ the N projective transforms.

Moreover, while in the 2D case, the derivation axiom was merely a rectangle filling the whole image domain, here, it is derived from the footprint of the building. In the experiments presented later, all footprints were retrieved from OpenStreetMap, using the address of the targeted building. The footprint must be expressed in the same euclidean coordinate system as the calibrated cameras. We have met these requirements by providing few correspondences between the cadastrial map and the structure-from-motion point cloud. Note that this step could be automated as well [Strecha 2010b]. Such a registration allows the output building models to be seamlessly embedded in a complete **GIS** environment.

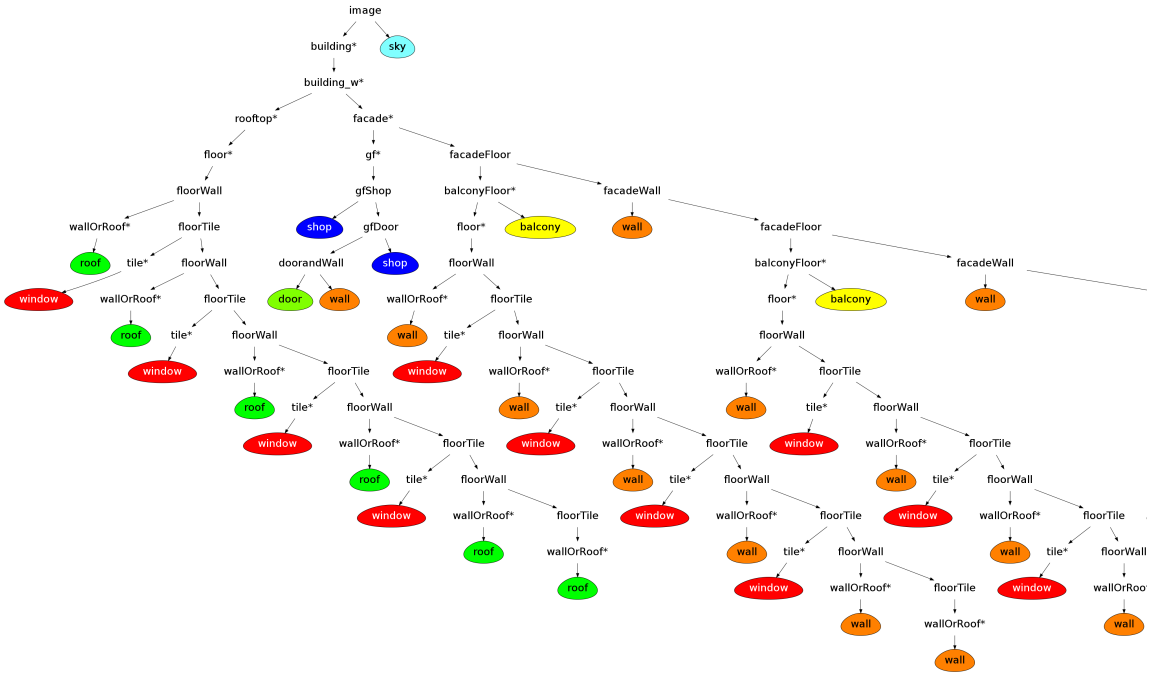


Figure 5.1: An example of parse tree associated with the 2D grammar. Each colored node simply holds a terminal symbol. On the contrary, internal nodes (in white) contain the unknown rules and parameters.

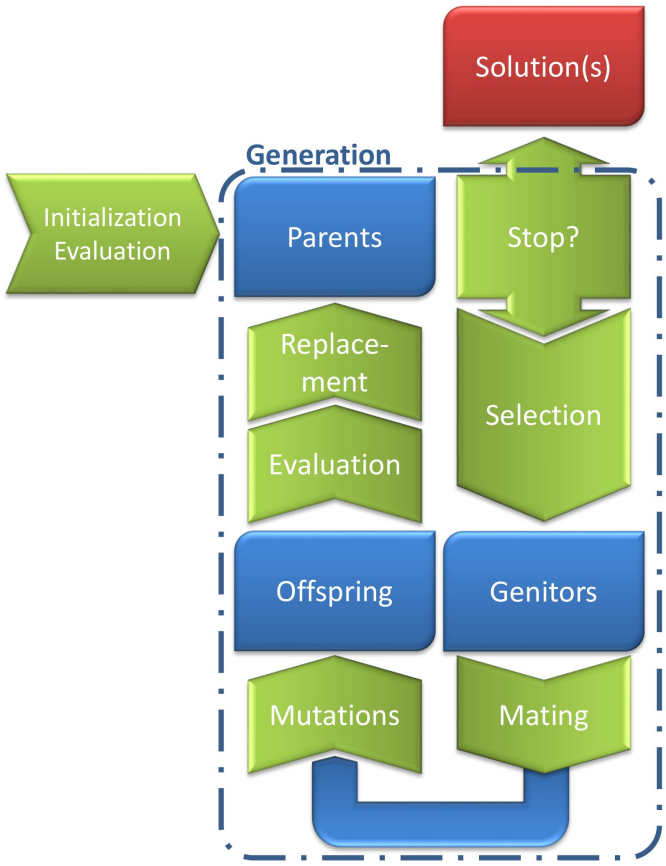
5.2 Better Exploration Strategy through Evolutionary Algorithms

To tackle the aforementioned challenges, we introduce a more elegant and appropriate inference framework based on evolutionary algorithms. In some sense, it can be viewed as an extension of the hill climbing approach developed earlier. We advise the reader to seek in [Coello 2007] for a recent and extensive survey of this field. We also recommend the introductory book [Luke 2009] which is available on-line.

5.2.1 Evolutionary Algorithms

Evolutionary algorithms are part of meta-heuristics which are stochastic search optimization methods. They are inspired by the biological evolution theory. The main idea behind this class of tools consists in using Darwin's principles to guide a population of candidate solutions towards the minimum of an objective function. The population evolves through a combination of variation and selection operators mimicking biological systems. A fundamental notion in this strategy refers to elitism which allows to favor promising individuals and discards the other ones.

The underlying concept of these algorithms is presented in Figure 5.2.1. Each iteration aims at



Algorithm 5.1 Evolutionary Algorithm in pseudo code

$P \leftarrow$ random population
evaluate fitness P
loop
  $P^* \leftarrow$ select P
  $O \leftarrow$ recombine P^*
  $O \leftarrow$ mutate O
  evaluate fitness O
  $P \leftarrow$ replace O, P
end loop

Figure 5.2: Typical pipeline of evolutionary algorithms. A population of individuals evolves to optimize a fitness function. At each generation, the parents P go through a elitist sampling from which the next generation genitors P^* are selected. The offspring is generated by re-combinations and mutations of the genitors and evaluated (*i.e.* the fitness of each individual is computed). In the last step, a subset the parent population is replaced by some children.

improving a generation P made of previously evaluated individuals. Evaluation of an individual consists in computing its objective function value(s). Then promising individuals P^* are selected for mating (recombination) leading to O as offspring. After a subsequent mutation step, the offspring O is evaluated, and reinserted in the current population towards replacing part of the previous generation (or the entire generation). Once a new generation has been formed, this evolutionary process is repeated, keeping track of the best individuals along the iterations. It continues until a stopping criterion is met.

One thing worth mentioning is that evolutionary algorithms can be applied without restriction on the objective function. This includes the case of multiple objective functions. Because of

this broad applicability, the fine theoretical study of convergence guarantees is very complex. Such guarantees depend on the data structure, the objective function to optimize and the concrete form of the operators involved in the loop. In general, the only legitimate claim about convergence merely states that if the mutation operator is "global", then the optimal individual will be surely found under the assumption of an algorithm running forever. In the previous statement, "global" refers to the capability to visit any point of the admissible space from any other point with a non-null probability.

In practice, we cannot afford to run the optimization for an infinite time interval. But, evolutionary algorithms behave more adequately when showing such a global behavior. In fact, the most important practical issue lies on the well-known exploration/exploitation trade-off. It consists in preventing the population from converging very slowly, while eliminating the risk of premature convergence to a local minimum. In the case of evolutionary algorithms, this trade-off is ruled by different parameters recapped hereafter.

- **The population size:** large populations ensure diversity for a longer time and therefore improve robustness to local minima. Note that a single individual population boils down to hill climbing and therefore relates to the previous methodology.
- **The number of offspring:** if many children are generated, the algorithm is less explorative since substantial effort is spent to visit a local neighborhood of each parent.
- **The selective pressure:** if unfit individuals are unlikely to be selected as genitors, the algorithm is fairly exploitative. When the best individual is the only genitor selected, then the algorithm behaves again as hill climbing.
- **The mutation amplitude:** a mutation can be thought as a jump in the space of solutions. If the mutation operator strongly favors "small jumps", then exploitation is preferred over exploration.
- **The replacement mode:** there are two major replacement schemes called (μ,λ) and $(\mu+\lambda)$[1]. Within the first case, the new generation is selected among the offspring, while the latter uses a pool made of the previous parents and the newborns. The second replacement scheme pushes good individuals to spread quickly and therefore favors exploitation over exploration.

The task adjusting the above mentioned settings is tedious. Furthermore, in practice towards truly optimal convergence, some of the settings require dynamic adjustment and should be updated during the run. For instance, following a naive idea, one could expect a progressive decrease of the mutation amplitude to improve convergence. But if the decrease rate is not very carefully chosen, premature convergence is likely to happen. Other heuristic rules for changing adaptively the amplitude have been proposed as the "one-fifth rule"[2].

[1]— They are pronounced respectively "mu comma lambda" and "mu plus lambda".

[2]— This rule increases the amplitude if more than one-fifth of the offspring are fitter than their parents, and decrease it otherwise

The use of concurrent populations (evolving with different parameters) is a common practice which provides an efficient way of dealing with the latter. Such a scheme, known as the island model [Skolicki 2008], is presented in Algorithm 5.2 where different sub-populations evolve almost independently. From time to time (after m generations), individuals are selected to migrate from a population to another. Moreover, the resources allocated to a population is related to its performance: the better it is, the bigger it grows, while the global population size remains constant. The selection process is done at the individual and the population level. In practice, this multi-population approach allows to perform a model selection that speeds up the convergence.

Algorithm 5.2 Concurrent populations: the island model

$\{P_1, \ldots, P_m\} \leftarrow$ random populations
loop
  for $i = 1$ to m **do**
    $n_i \leftarrow$ allocate resource (P_i)
    $P_i \leftarrow$ new generation (P_i, n_i)
  end for
  migrate $\{P_1, \ldots, P_m\}$
end loop

5.2.2 Optimization of the Grammar Derivation

To apply the previous methodology to our particular problem we need to specify the operators driving the search. Two issues are to be addressed: the kind of individuals we are studying and the evolution processes applied to them.

Two representations usually make sense in evolutionary algorithms: the genotype and the phenotype. Generally speaking, the genotype is a convenient representation to operate on the individuals (*i.e.* perform mutations and recombinations). On the other hand, the phenotype is a "physical" representation which is natural to evaluate the quality or fitness of an individual. It might happen that in some applications a unique representation satisfies both requirements. It is not the case here, as the natural genotypic representation is the parse tree while the phenotype should be the semantic layout implied by performing the derivation. We will first consider the genotype and the associated operators (mutation and recombinations) and then the properties related to the phenotype.

Genotype

A parse tree is made of nodes involving three components $n = (r, \mathbf{x}, p)$. r and $\mathbf{x}$ are respectively the rule identifier and the internal parameters of this rule. They correspond to the unknown variables of the optimization. The last component, p, is the semantic primitive created in the scope of the node

during derivation. It stores both a symbol s and tag c used to control the derivation. One could argue that such component is redundant, given that it could be obtained by deriving in accordance to the parse tree. Nevertheless, it greatly helps with explanations and implementation.

For a given node, the rule can be chosen among the finite set $\mathcal{R}_p$ of rules applicable to p. On the contrary, the parameters correspond to a real-valued vector. Changing a rule impacts the topology of the derivation tree and the symbols of its nodes. Changing a parameter mainly impacts the geometry of the layout, but may also remove some nodes or make new nodes emerge in the tree. Pay attention that modifying a single node can affect the validity of the sub-tree below and sometimes of other sub-trees because of the non-local bounds induced by consistency tags.

Therefore, after applying an evolutionary operator to an individual, repairing steps must be performed to turn the modified individual into a valid derivation tree. In fact two predicates must be verified. First, the children of a node must have exactly the symbols emerging from the application of the rule of the parent. Moreover, all nodes sharing the same symbol and same tag must have the exact same rule and parameter values.

Mutation

A mutation refers to a random perturbation of an individual. After a node $n = (r, \mathbf{x}, p)$ is chosen randomly, three different mutations can be performed, one acting on the rule and two on the parameters.

1. Rule mutations consist in replacing r by a random rule of $\mathcal{R}_p$, if $\#\mathcal{R}_p > 1$.

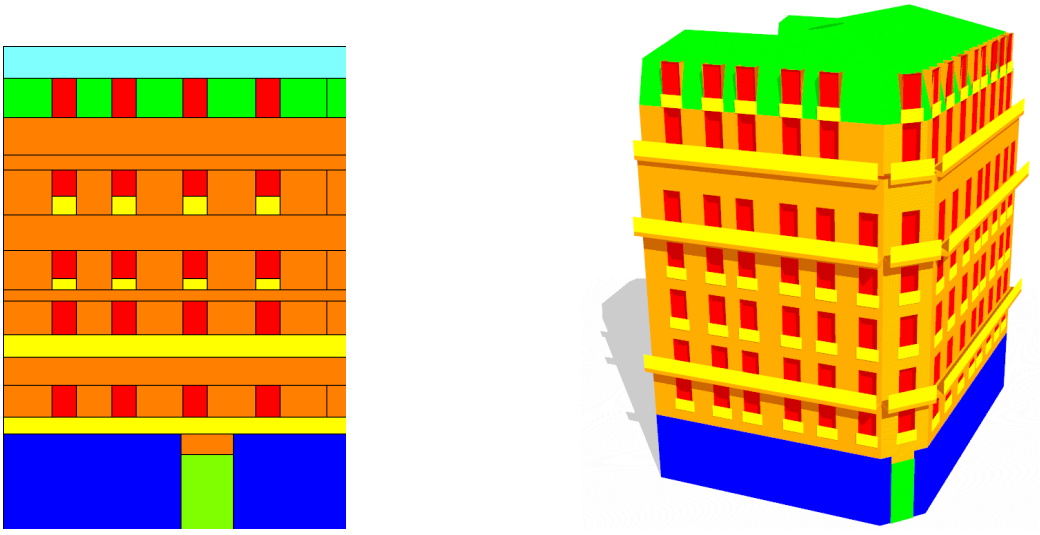
2. Parameter mutations can modify $\mathbf{x} \in \mathbb{R}^n$ by adding a perturbation to a randomly chosen element $x_i \in \mathbf{x}$:

$$x_i \leftarrow x_i + u \tag{5.1}$$

   Alternatively, parameter mutations randomly select two elements x_i and x_j either from the same node or from successive ones, add a random perturbation to x_i and decrease x_j accordingly. These concurrent mutations present the advantage of modifying the position of a given architectural element without impacting the rest of the layout.

$$\begin{aligned} x_i &\leftarrow x_i + u \\ x_j &\leftarrow x_j - u \end{aligned} \tag{5.2}$$

   In Equation 5.1 and Equation 5.2, $u \sim \mathcal{U}([-\delta, \delta])$ is sampled from a uniform law. The value of δ depends on the rule in which $\mathbf{x}$ is involved and the mutation policy. Such a policy can be coarse or fine, and actually modifies δ accordingly.



(a) (b)

Figure 5.3: A parse tree implies a semantic layout model providing a hypothetical explanation of the observations. Such layouts are shown in 2D (a) and in 3D (b). They constitute the phenotype of an individual.

Recombination

Mating two individuals simply swaps a part of their contents. In our case, we first choose one node n in the first tree and then a node n' in the second one, such that they share the same class. Once again, two recombination modes are possible.

1. Rule recombination can be performed by swapping the rule components of the two nodes as well as their parameters.

2. Parameter recombination between $\mathbf{x}$ and $\mathbf{x}'$ is done by swapping the values of two randomly chosen components x_i and x'_i.

Phenotype and evaluation

A parse tree stores all the necessary information to derive a semantic layout M describing a facade (Figure 5.3 (a)) or a building (Figure 5.3 (b)). This layout refers to the physical expression of the individual, in other words, its phenotype. We have already seen in the single-view case, that the individual relevance with respect to the observations can be evaluated thanks to an appearance energy $E_a(M)$. In the multiple-view case, an individual will be also evaluated thanks to a depth energy $E_d(M)$. The precise definitions of both energies will be detailed in a subsequent section.

The evaluation step consists in computing for each individual its fitness function comprising all energy components.

Selection

Selection is an elitist process allowing to favor "strong" individuals to survive and evolve during the generations. When only an appearance energy is considered, individuals are selected by running tournaments. To choose one genitor, few candidates (2 in our case) are randomly picked. Then, only the fittest candidate among them is chosen as genitor. The selected candidate remains in the pool so that it can be selected several times.

Otherwise, since $E_{total}(M) = (E_a(M), E_d(M)) \in \mathbb{R}^2$ and since no preference is made about the two fitness components, one can not establish a ranking of the individual based on a total ordering. However, we can compare individuals using Pareto partial ordering. Formally, we will say that an individual M_1 dominates another M_2 and note $M_1 >_p M_2$, if M_1 has all energies smaller than M_2 (one being strictly smaller). With a partial ordering, there is no guarantee that an individual will be better than any of the others. However, we can find a set of individuals that are dominated by none of the admissible solutions. This set called the **Pareto frontier** is depicted in green in the explanatory example of Figure 5.4.

Several selection policies relying on Pareto ordering have been defined in the literature. We have chosen to follow the modified Strength Pareto Evolutionary Algorithm (**SPEA-II**) scheme [Zitzler 2001] that is considered among the state of the art. Other recent schemes would possibly perform equivalently. In any case, a selection operator should at least ensure two properties. Of course if $M_1 >_p M_2$, then M_1 should be preferably selected over M_2. Additionally, selection should ensure a certain level of diversity with respect to the fitness values among the selected individuals. We will see that this second goal offers important practical benefits.

Algorithm 5.3 The modified strength-pareto evolutionary algorithm

Require: P: current population
Require: A: archive of non dominated solutions
Require: a: maximum size of archive
  $A \leftarrow$ non-dominated solutions of $P \cup A$
  if $\#A < a$ **then**
    extend A with $(a - \#A)$ fittest dominated solutions from P
  else
    remove from A its $(\#A - a)$ individuals at dense locations if A
  end if
  $P^* \leftarrow$ tournament selection on A
  return P^*, A

SPEA-II is presented in pseudo-code in Algorithm 5.3. The algorithm performs both the pareto front estimation and the selection of genitors. Selection is performed by using a classical tournament on the current estimate of the front. Then, the front estimation step implements both of the previously stated requirements. The front is approximated by a fixed size archive A, which is initialized with all non dominated solutions found so far and is then pruned or completed (if it is respectively too big or too small). Note that diversity is promoted by the fact that that the pruning is performed preferably on solutions located at dense position in the front.

Concurrent populations

As mentioned earlier, different strategies are put in competitions thanks to a sub-population model. In our case, one population evolves through rule mutation and recombination, and two populations through parameters mutation and recombination, respectively with coarse and fine mutation policies. The first population aims at optimizing the topology through the rules while the two others focus on the geometry through the parameters. Ideally, the sub-population which acts finely on parameters should perform best when approaching a local minimum while the other two should help escaping from non global optima.

5.2.3 Optimal solution in the multi-objective case

Successive iterations of the multi-objective algorithm do not produce a specific solution but a whole set of candidate solutions approximating the Pareto frontier. An illustrative example of a Pareto frontier and its approximation is depicted in Figure 5.4. In practice, we are looking for a single optimal solution. Thus, we linearly combine the two energies into a single one:

$$E(M) = \alpha E_a(M) + (1 - \alpha) E_d(M) \tag{5.3}$$

The level sets of E correspond to straight lines in the energy diagram, represented in black dashed style in Figure 5.4. The optimal solution with respect to this energy is easily determined from the complete set of Pareto solutions. Note that among the approximate Pareto solutions, only those belonging to the Pareto convex hull can be optimal for a linear combination of the objectives.

Better exploration behavior

One may think that we could have used a combined energy from the beginning. This assumption is strongly objected in the theory of evolutionary optimization where the consensus state that keeping multiple objectives reduces the risk of convergence to a local minimum. This is particularly relevant when the Pareto frontier shows concave parts, as in the illustrative example proposed in Figure 5.4. In such cases, a local minimum of the scalar energy often appears far from the true optimum. If little attention is paid to this fact, a strong risk exists that the population converges to the local minimum. It would then be vain to hope that the population could escape from there quickly.

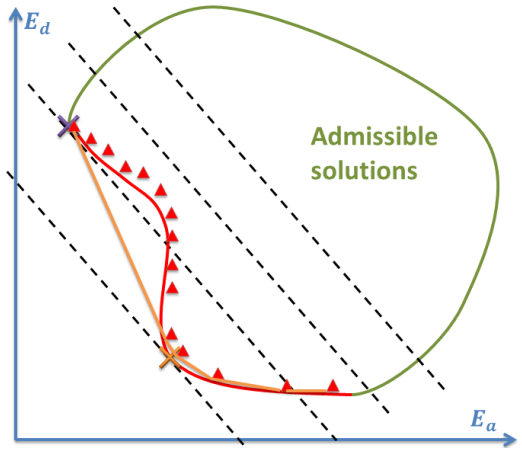


Figure 5.4: A Pareto frontier (red line) estimated as a set of solutions (red triangles). The orange poly-line is the convex hull of this approximate Pareto front. Few level sets of a linearized scalar fitness are depicted (dashed black). Observe the local minimum (purple cross) located far away from the global one (orange cross).

Single objective selection operator are largely affected by this phenomenon. On the contrary, **SPEA-II** selection produces a population with a diverse range of fitness values. Therefore, when converging, the population should ideally spread over the entire Pareto frontier.

We illustrate this assertion with an experiment made on a real case. Figure 5.5 presents in a shared energy graph, the space spread by two populations, conducted respectively by **SPEA-II** selection and a tournament selection based on a linearized objective function. The latter population is therefore guided by the scalar energy. Surprisingly, the population produced with the consideration of the two individual components presents better candidates with respect to this scalar energy. In order to support our previous analysis, the figure depicts the last generations of both populations. As expected, the single-objective population falls in a narrow region of the admissible space. Considering now the average behavior of each population, the **SPEA-II** appears to drive the population towards more promising regions.

Automatic weight selection

It is also important to note that the choice of α plays a key role on the final outcome of the optimization process. Defining such a value beforehand can only be done in a heuristic manner. This task becomes less difficult when the Pareto set is known. Indeed, its bounding box could produce valuable insights regarding the appropriate balance between the two components of the objective function. A practical way to automatically extract a sensible α is explained in Figure 5.6. It consists in choosing a weight such that the iso-lines of the scalar energy to be parallel to the diagonal of

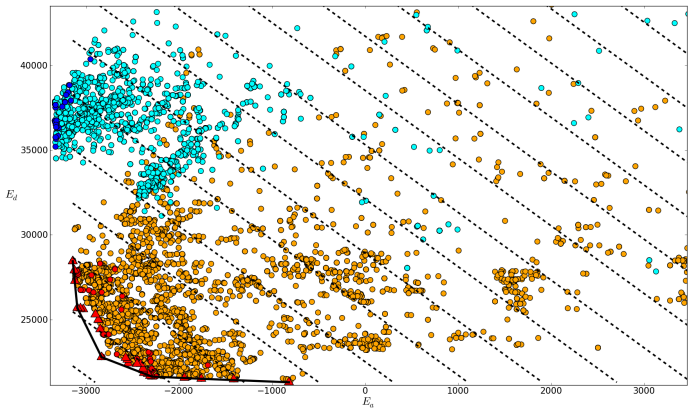


Figure 5.5: Convergence comparison of **SPEA-II** and tournament selection using a linearized energy. A population driven by **SPEA-II** is shown in orange dots. Red dots and triangles correspond respectively to the last generation and the estimated Pareto set. A population obtained with the tournament selection is depicted in cyan, the corresponding last generation being in blue.

the Pareto set bounding box. As a result, the optimal solution is the one maximizing the euclidean distance to the diagonal of the bounding box. This automatic procedure is implemented in our approach.

Optimal solution uncertainty

Eventually, it is possible to assess qualitatively the selected solution. Indeed, as pointed out earlier, the potential challengers to the optimal solution belong to the convex hull of the Pareto set. This is verified as soon as we restrict ourselves to select the solution based on a weighted sum criterion. Then the strength of the solution chosen based on α_0 can be reasonably defined as the range of α values producing the same optimal solution. A very small range means that using an alternative value only slightly different from α_0 would lead to a different solution. In such case, the intervention of a human operator can be encouraged and can be easily considered given that we can produce a subset of well chosen challengers. This subset is constrained from the convex hull and within it, challengers can be ranked by increasing distance between their associated range and α_0.

5.3 Energies in the Multi-view Context

In this section, we will propose two different energy models, one extending the appearance model to multiple views and another evaluating the quality of the model with respect to depth profiles. Note that all the developments made here are specific to the multi-view settings.

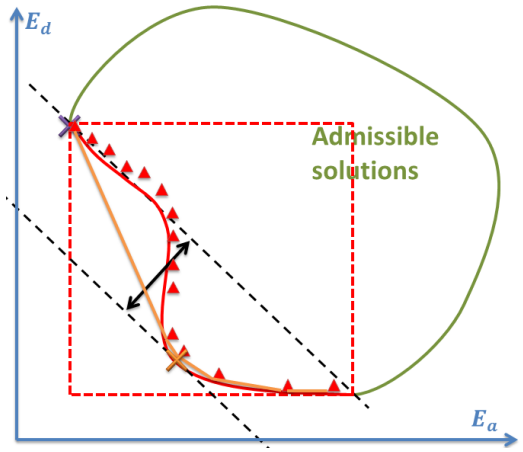


Figure 5.6: Automatic selection of α. The weight can be chosen so that the iso-lines of the linearized scalar energy are parallel to the diagonal of the bounding box of the Pareto set.

5.3.1 Appearance Model

The first energy is largely inspired from the one derived in the single-view approach. We adopt a Bayesian formulation where we aim at defining the posterior probability of the 3D model M knowing the set of views I: $P(M|I)$. Since M is procedurally generated, it can be viewed as a set of voxels $\mathcal{X}$ associated with semantic classes (or terminal symbols) c: $M = (\mathcal{X}, c)$.

We first express the posterior with the likelihood and the prior using the Bayes rule: $P(M|I) \propto P(I|M)P(M)$. Given that the grammar already expresses a very strong prior on the model, we do not consider additional ones, an assumption that boils down to a direct use of the likelihood. Then, assuming independence between the different views and among the pixels of each image, the likelihood is factorized as:

$$P(I|M) = \prod_{k=1}^{N} P(I_k|M) = \prod_{k=1}^{N} \prod_{x} P(I_k(x)|M) \tag{5.4}$$

In the factorization, we now have to express only the pixel likelihood $P(I_k(x)|M)$. We make the natural assumption that the distribution of $I_k(x)$ given the model M only depends on the voxel $\pi_k^{-1}(x)$ which projects on the pixel x in I_k. Then $P(I_k(x)|M)$ can be rewritten as $P(I_k(x)|\pi_k^{-1}(x))$.

Of course this latter expression is only valid when $\pi_k^{-1}(x)$ exists, that is to say when the line of sight passing through x intersects M. We concisely denote this case by $\pi_k^{-1}(x) \in \mathcal{X}$. In the opposite case, when $I_k(x)$ cannot be explained by M, then $\pi_k^{-1}(x)$ should be considered as a hidden variable, and $P(I_k(x)|M)$ can be evaluated by averaging the previous expression over the set of possible value of $\pi_k^{-1}(x)$. As a result, we obtain the following expression for the pixel

likelihood:

$$P(I_k(x)|M) = P(I_k(x)|\pi_k^{-1}(x)) \quad \text{if } \pi_k^{-1}(x) \in \mathcal{X} \tag{5.5}$$
$$= \mathbb{E}[P(I_k(x)|\pi_k^{-1}(x))] \quad \text{otherwise} \tag{5.6}$$

Let us motivate the importance of dealing with the likelihood at pixels which are not explained by the model. To that end, we need to consider that the goal is to maximize the likelihood value with respect to the degrees of freedom of the model. This is equivalent to the minimization of the negative log-likelihood energy:

$$E(I, M) = \sum_k \sum_x -\log\left(P(I_k(x)|M)\right) \tag{5.7}$$

In the inner sum, some of the pixels are explained by M while others are not. Evaluating the likelihood in both cases makes the energies of all candidate models comparable. Practically, this prevents the model from progressively shrinking during the optimization so as to diminish the number of observations to account for. One can note that this difficulty does not arise in the 2D settings as the complete image domain is classified.

Following the previous chapter, we can use an appearance model to derive the pixel likelihood. In that case, it is estimated as $P(I_k(x)|\pi_k^{-1}(x)) = P(I_k(x)|c(\pi_k^{-1}(x)))$. Thus for any class c the probability distribution $P(I_k(x)|c)$ is learned using a Gaussian Mixtures Model on the RGB values of pixels in a training set. In practice, the training set is created by a user who paints some brush strokes on one of the input image for all the terminal classes of the grammar.

Consequently, the expectation in Equation 5.6 takes place over the set $\mathcal{C}$ of all possible classes. Pixel contributions which do not correspond to valid projections of voxels are replaced with the average cost:

$$P(I_k(x)|M) = \frac{1}{\#\mathcal{C}} \sum_{c \in \mathcal{C}} P(I_k(x)|c) \tag{5.8}$$

Therefore, the appearance energy E_a is obtained by taking the negative log-likelihood as in Equation 5.4. It uses appearance as a way to distinguish the different primitives of the model. However, it fails to account for real 3D evidence with respect to the voxels belonging to different or even to the same class when located at different depths. This can be addressed through a depth energy.

5.3.2 Depth Model

As an alternative, a depth energy is obtained by comparing the model M with the reference 3D point cloud $\mathcal{P}_{ref}$. Basically, from any existing or artificial camera with projection π, we build two depth maps: the depth map of the model D_M and the depth map of the point cloud D_{ref}. In each

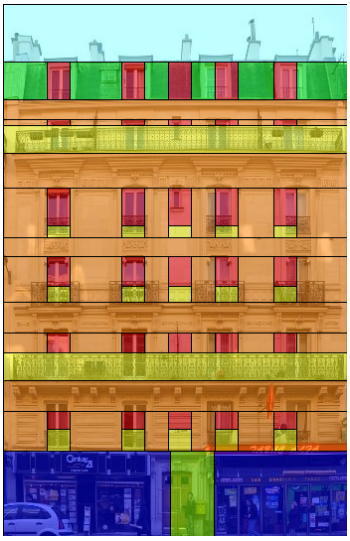


Figure 5.7: Example of undesired segmentation corresponding to a local minimum of the energy.

pixel x of a depth map D, $D(x)$ represents the distance between the optic center of the camera and the voxel $\pi^{-1}(x)$ that is projected onto the depth map. Then the energy of a model is:

$$E_d(M) = \sum_x ||D_M(x) - D_{ref}(x) - \bar{d}||^2 \tag{5.9}$$

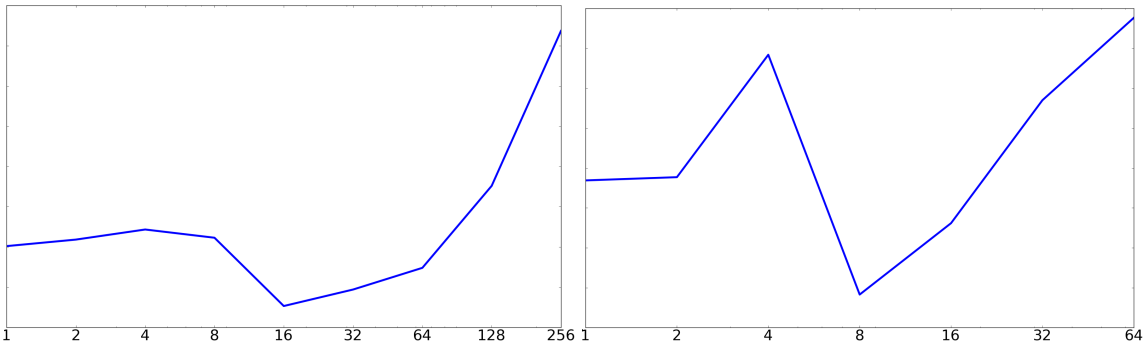
where $\bar{d}$ is the mean distance between the depth map of the model and the depth map of the 3D point cloud. Note that the goal expressed by this energy is not to have a model with the exact same depth profiles as the reference but simply that they match up to a constant gap. This subtle difference makes the depth cues relatively robust to slight inaccuracy introduced by the registration of the structure from motion frame with the cadastrial map.

In practice, all depth maps are extracted from the Z-buffer of virtual orthographic cameras. For each visible facade of the building, a camera of this sort faces the facade.

5.4 Experimental Validation

5.4.1 Single-View

In the following set of experiments, we will evaluate the performance of the evolutionary algorithm in the single view context. In particular, a comparison with the hill climbing approach will be presented.



(a) $\#O = 8$ (b) $\#P = 64$

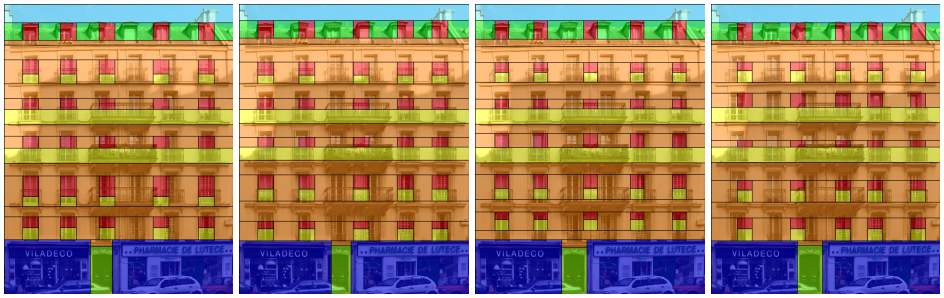
Figure 5.8: Evolution of the best individual energy obtained with varying population size (a) and with varying offspring size (b). For each size parameter, the optimization was launched 100 times on the same facade with different random seeds. The blue curves represent the average optimal energies obtained for the 100 runs. As usual, a lower energy indicates better performance.

Empirical calibration of the evolutionary settings

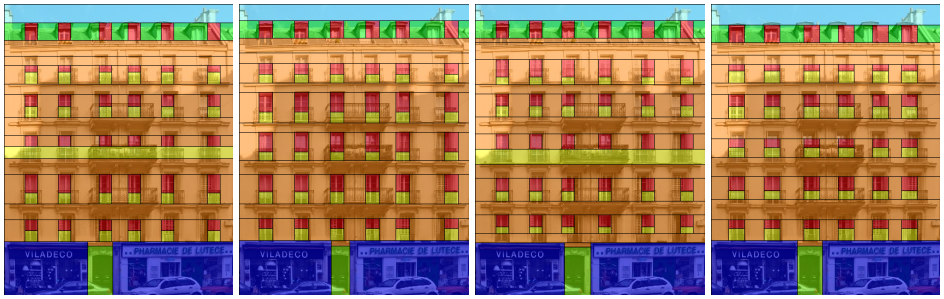
Some of the settings of the evolutionary algorithm deserve attention. In fact, the energy we want to optimize is non convex. It presents many local modes where the optimization can easily get stuck. An illustration of the reason behind the multi-modal behavior of the energy is provided in Figure 5.7. In the depicted segmentation, all columns of windows except the central one are aligned with actual windows in the facade image. To remove the erroneous column, all the right ones need to be pushed side-wise which will inescapably increase the global energy. In the hill climbing approach, in order to escape from undesired solutions, interesting parts of the parent neighborhood were thoroughly exploited by sampling extensively (1^{e4} children) at every iteration. It is probably not the optimal solution in terms of speed and convergence guarantees.

In order to determine an acceptable consensus between the population size and the offspring one, we have studied the performance of the algorithm with respect to these two settings. The results are presented in Figure 5.8. For each sub-figure, one parameter varies while the other is kept constant. In order to eliminate the effect of unpredictable fluctuations due to the stochastic facets of the optimization, many runs were considered per value of each parameter, resulting on an average performance estimate.

Note that if we want to compare the performance obtained in different settings on a fair basis, we need to impose an invariant number of tested hypotheses (*i.e.* the total number of individuals created during a run). This limit was chosen based on another study [Teboul 2011], where we have tackled the same problem using Reinforcement Learning. In this paper, we have demonstrated that it is possible to obtain similar performance as in the hill climbing approach with only 3000 to 5000 derivations. Therefore the results shown in Figure 5.8 were obtained with a number of derivations bounded to 3000.



(a) $\#P = 1$



(b) $\#P = 16$

Figure 5.9: Some local modes expressed with populations of size 1 and 16. In each case, the images represent final segmentations obtained at randomly selected runs.

Both curves shown in Figure 5.8 present the same global behavior. Keep in mind that better performance is achieved when the curve reaches a lower position. In both cases, the left region corresponds to exploitative settings and the right one to explorative regions. As expected, the optimal settings are always located in an intermediary region. One can note a special artifact in the performance evolution, in the left region of each graph, where unexpectedly the performance suffers a small drop. This artifact is due to the fact if exploration is weak, then the population is quickly uniform. As a result, convergence towards local modes cannot be avoided and occurs at a lower speed.

The natural conclusion is that a population made of a single individual is not optimal because it can not escape efficiently from the numerous local minima. This statement is supported by the final segmentations obtained during the runs. Figure 5.9 shows a random selection of such segmentations for the single individual case (upper row) and for a population size of 16. With a single individual, the segmentation often misses a column of windows. Similarly, a large offspring is incompatible with a limited number of derivations, because too much time is spent exploring the same regions of the space.

Note that the optimal settings depend on various elements. This includes especially the complexity of the facade to be parsed and the allocated time to perform the optimization. About,

the first point, the same experiment was conducted on other facade, leading to varying optimal configurations. When using a simple facade, the performance profiles are rather flat, with better performance on exploitative variants. On the contrary, with complex facades implying many local minima, the profiles present a sharper optimum. After a careful observation, a population size of 32 and an offspring of 16 were found to provide a decent consensus.

Performance on the Ecole Central Paris Facade Database

The following tests were launched on the same data-set as in chapter 4. 10000 hypotheses are tested for each facade. The island model was not used and the unique population evolves under rule mutations. Motivated by the previous experiments, the population and offspring sizes are set respectively to 32 and 16.

To compare with the results obtained with hill climbing, we computed the confusion matrix. The differences with respect to the detection rate are shown in green or red depending on whether better or worst detection is performed. Globally the performance has slightly improved. The only exception concerns the windows which are subject to an approximately 10% loss. This is mainly due to the restriction to global mutations in these tests. As a result, few rows of pixels of the windows are often missed. Because windows are small structural elements, even a slight misdetection of their boundary has an noticeable impact on the detection rate.

This experiment shows that evolutionary algorithms provide equivalent performance as the hill climbing alternative, with fewer hypotheses tested. This is very promising for the 3D multi-view extension.

$$\begin{pmatrix} 70 & 23 & 5 & 0 & 2 & 0 & 0 \\ 2 & 91 & 6 & 0 & 0 & 0 & 1 \\ 7 & 19 & 73 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 76 & 0 & 0 & 22 \\ 6 & 4 & 0 & 0 & 88 & 2 & 0 \\ 1 & 0 & 0 & 0 & 3 & 96 & 0 \\ 0 & 2 & 0 & 4 & 0 & 0 & 93 \end{pmatrix} \begin{matrix} -11 \\ +8 \\ +1 \\ +5 \\ +8 \\ +2 \\ -1 \end{matrix} \begin{matrix} \text{window} \\ \text{wall} \\ \text{balcony} \\ \text{door} \\ \text{roof} \\ \text{sky} \\ \text{shop} \end{matrix}$$

5.4.2 Multi-View Experiments on Artificial Data

We first study the convergence of our algorithm on artificial data, for which we have a noise-free ground-truth. To do so, we randomly generate a building and visualize it from 3 cameras. Here, the classification cues are perfect: the probability is either 1 or 0 depending directly on whether the color predicted by the candidate solution matches the one of the synthetic model. As for depth, D_{ref} is here the depth map of the ground-truth model. The evolution of the populations is given in Figure 5.11. For clarity sake, we plot the evolution of the 3 populations for the appearance energy only in Figure 5.10 (the evolution of the depth energy is indeed very similar).

We can observe on Figure 5.10, that the energies E_a (or similarly E_d) of the 3 sub-populations are in average decreasing with time, and tend to converge towards a constant energy after approximately 150 iterations. It is interesting to observe that, upon convergence, P_f has a lower mean

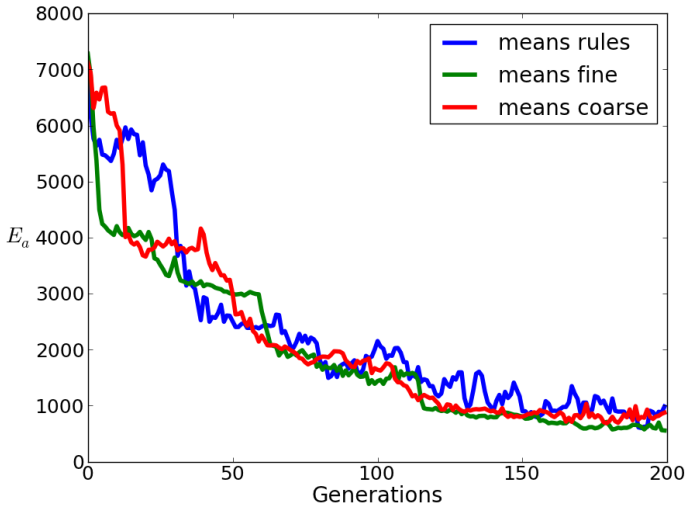


Figure 5.10: Convergence of the 3 sub-populations, on the appearance objective. The 3 concurrent populations are in average decreasing toward a minimum.

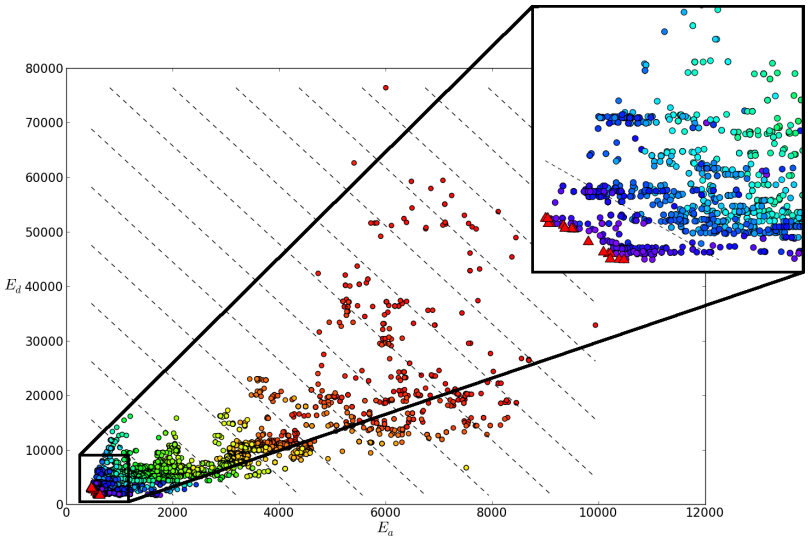


Figure 5.11: Convergence of the 3 sub-populations on both objectives. Each point represents an individual, with one color per generation. The individuals start at the upper right corner and converge toward the lower left one. The Pareto frontier approximation is depicted in red triangles. It shows a noticeable concavity.

energy than P_c, that is itself better that P_r. The *rules* population only allows important changes, since its mutations are completely changing the topology of a building. The *coarse* population also enforces important changes, but mainly geometrically, whereas the *fine* population acts on a finer scale.

On Figure 5.11, we plot the evolution of the whole population without distinction of sub-population. Each color is an iteration, ranging from red (1) to blue (200). At the beginning of the evolution, the individuals are scattered out. As iterations goes they all move towards the known optimal point $(0, 0)$. After 200 generations, the best set of individuals is the Pareto frontier of the population, here represented in red triangles. Note that this set spreads a very limited region which is consistent with the fact that the real Pareto frontier is made of a single point.

5.4.3 Quantitative Validation on Real Multi-view Data

Performance criteria

Given that the model incorporates both geometric and semantic information, it is natural to seek for the evaluation of the relevance of both aspects. We have already adopted a criterion regarding the semantic aspect of the obtained solution which is based on confusion matrices. We will keep the same line in this set of experiments.

As for the geometric aspect, the task is more difficult. In the presence of a ground-truth model - obtained with a LIDAR scanner for instance - the comparison of the reconstruction with the ground-truth is not straightforward. In particular, classical statistics based on point-to-surface distances do not make sense. The main reason for that is that the two models encapsulate very different level of details about the geometry of the building under consideration. More specifically, the LIDAR cloud reflects faithfully small details on non reflective surfaces, as for example facade ornaments. On the contrary, reflective surfaces such as windows are poorly captured by LIDAR techniques. This applies as well to thin structures like ironworks. Unfortunately, these failure cases correspond precisely to the major structural elements recovered by our approach.

If we want to assess geometric accuracy, we need a ground truth model M_{gt} comparable to the one inferred from the grammar. Then, we can compute the average point-to-surface distance between the inferred model and the reference one.

$$d(M, M_{gt}) = \frac{1}{\#\mathcal{X}} \sum_{x \in \mathcal{X}} \min_{x_{gt} \in \mathcal{X}_{gt}} d(x, x_{gt}) \tag{5.10}$$

where $\mathcal{X}$ and $\mathcal{X}_{gt}$ are fine point cloud derived from both models.

To go further in our analysis, we propose a variation of the previous criterion, where we consider alternately the different semantics and extract the same statistics from the 3D points matching the semantic class under consideration.

$$d_c(M, M_{gt}) = \frac{1}{\#\{x \in \mathcal{X} s.t.\ c(x) = c\}} \sum_{x \in \mathcal{X} s.t.\ c(x)=c} \min_{x_{gt} \in \mathcal{X}_{gt}} d(x, x_{gt}) \tag{5.11}$$

Ground-truth

One way to obtain the required inputs for both criteria consists in performing a manual derivation of the grammar. In that perspective, we have designed a GUI allowing to progressively tune the derivation tree with respect to both rules and their parameters.

Practically, one needs to adapt alternately the position and sizes of each element. Against all odds, this task is not simply tedious but also extremely hard to tackle with regard to the depth accuracy. Because of this difficulty, our set of ground truth models is restricted to 5 Haussmannian buildings.

Results

We first consider the classification criterion presented in the following confusion matrix. The form of the matrix is very similar to the one of the single-view configuration. This shows that the grammar context still plays its role. Besides, the numerical values are approximately equivalent to these obtained in the single-view experiments. We do not provide a detailed comparison with the previous experiments as the data set is different.

$$\begin{pmatrix} \mathbf{70} & 24 & 5 & 0 & 1 & 0 \\ 3 & \mathbf{83} & 13 & 0 & 0 & 0 \\ 10 & 7 & \mathbf{82} & 0 & 1 & 0 \\ 0 & 2 & 0 & \mathbf{84} & 0 & 14 \\ 8 & 6 & 7 & 0 & \mathbf{79} & 0 \\ 0 & 4 & 0 & 2 & 0 & \mathbf{94} \end{pmatrix} \begin{matrix} \text{window} \\ \text{wall} \\ \text{balcony} \\ \text{door} \\ \text{roof} \\ \text{shop} \end{matrix}$$

In Table 5.1, we show the statistics obtained with Equation 5.11 for all the semantic classes. Overall, the deviation is bounded by 30 centimeters. Large gaps between the ground truth and the inferred model concern mainly the roof and the shops. But in such cases, this difference has little impact on the visual quality of the model since it concerns large structures. On the contrary, the error for windows averages to 11 centimeters and is less satisfactory given that the depth of a window amounts to approximately 50 centimeters. Nevertheless when considering the final model, we will see that this is hardly noticeable.

c	$d_c(M, M_{gt})$
window	11
wall	4
balcony	13
door	1
roof	31
shop	27

Table 5.1: Geometric accuracy - distances are expressed in centimeters.

As a basis for computational efficiency comparison, we have allowed $20,000$ hypotheses for buildings with only one facade visible from the street. Otherwise, the number of hypotheses was multiplied by the number of facades. This shows that despite a greater complexity in the inference task, the number of candidate derivations remains much lower compared to the one required by the hill climbing methodology. However, even though the algorithmic complexity is substantially reduced, the necessary time to reconstruct a single building remains significant. Indeed, the inference typically requires around one hour. This duration is nonetheless subject to the complexity of the building.

Qualitative Results

We present here some results on few different Haussmannian buildings from Paris. The optimal solution was chosen following the automatic α selection scheme presented in section 5.2.3.

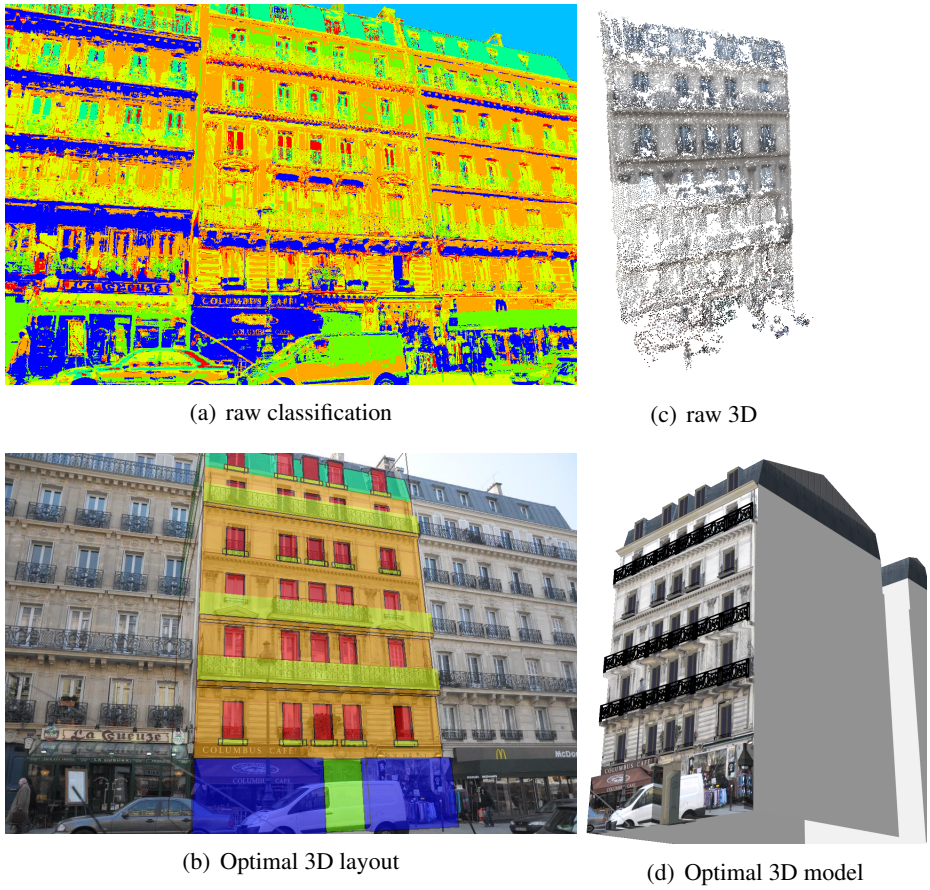
For each of the buildings, we show the classification induced by the 3D semantic layout and the textured 3D model in which structural elements are replaced by highly detailed models. In particular, Figure 5.12 draws a parallel between these results and the raw classification and depth cues. The gain is obvious with respect to both aspects. In spite of the complexity of the problem and the very noisy level of both appearance and depth information, the models are very accurate, both in terms of geometry and topology. Different results obtained on other examples are depicted in Figure 5.13. Last, Figure 5.14 shows the reconstruction obtained with a building presenting many street facades. This example illustrates the benefit of handling the building as a whole instead of dealing with each facade separately.

Although the results are very satisfying, few failure cases are to be deplored. First, it happens as in Figure 5.12, that an additional running balcony appears. This case occurs typically when balcony guards are large and there is a partially running balcony at the level of the concerned floor. In this situation, the grammar is at fault as it does not allow to accurately model the observed building. Therefore the reconstruction is a consensus between the language of the grammar and the observations. Besides, the registration of the point cloud and the camera is sometimes not as accurate as expected, and it can therefore introduce some bias in the final model. An example of this drawback can be observed on the second row of Figure 5.13, where windows are well fitted on the bottom floor and step progressively aside their expected position in the upper floors.

5.5 Conclusion

In this chapter, we have extended 2D grammar-based procedural reconstruction to 3D while introducing additional depth cues. The main aspect lies in the ability to provide the real 3D geometry of the building. To do so, the approach combines two likelihood functions based on appearance and depth defined on a multi-view basis. The reconstruction task was tackled as an inference of an optimal grammar derivation.

To solve the induced multi-objective optimization problem, an adapted evolutionary algorithm was introduced. The relative performance of this algorithm was considered with respect to the



(a) raw classification (c) raw 3D

(b) Optimal 3D layout (d) Optimal 3D model

Figure 5.12: Optimal procedural reconstruction compared to raw inputs. First row: raw classification and point cloud. Second row: the optimal 3D layout and the derived 3D model obtained by adding detailed 3D models and back-projecting a texture.

previous approach by addressing the 2D parsing problem where a single objective is available. It was then studied with respect to the multi-view reconstruction problem, first in a synthetic case and then on real data. In this latter case, the demonstrated performance relies on the efficiency of evolutionary algorithms to estimate the Pareto front and a well suited automatic selection scheme among all the Pareto candidates.

The explicit modeling of occlusions is a natural extension of our approach. The estimation of the camera parameters on the fly, as well as the footprint of the building as part of the evolutionary algorithm could lead to a fully automatic approach to 3D procedural modeling. Looking at efficient means of accelerating the convergence process through intelligent mutations driven from bottom-up cues is also a very promising direction.

Figure 5.13: 3D reconstruction of different buildings.

Figure 5.14: 3D reconstruction of a building with multiple facades visible from the street. Note that floors align seamlessly along the different facades.

Chapter 6

Conclusion

We conclude this thesis by summarizing the majors contributions and open the discussion on some potential directions of research that will be investigated.

6.1 Contributions

In the introductory chapter, we have presented the perimeter and the objectives of the thesis towards image-based building modeling of large-scale environments. Globally, our response to this challenge integrates procedural representations within image-based inference paradigms. Therefore, our work is an attempt at conciliating graphics modeling approaches with techniques typically related to computer vision. Although our purpose was a well defined and expected objective, satisfaction of such an attempt was not straightforward as it can be testified by the numerous attempts made in both communities.

The first aspect we had to deal with concerned procedural modeling on its own. Shape grammars frameworks developed in the last decade have provided a strong basis for our research as they introduced adapted techniques for architecture modeling. Building upon these ideas, we have come up with an extended approach contributing with a novel derivation scheme helping to finely control the designs described by a grammar. This control was demonstrated as an efficient solution to enforce the architectural consistency of procedurally generated buildings. Within this framework we have also demonstrated the power of binary recursive split rules to generate alternative patterns of unlimited cardinality. These contributions have led to the definition of grammars specifically fitted to the generation of 2D and 3D semantic layouts for buildings of Haussmannian type. The capability to create hypothetical semantic layouts has granted a sound basis for image-based analysis.

Such a modular procedural generation engine was combined with the inference of a plausible layout for a building observed in a single or in multiple photographs. One major idea that stands out of our work ties the semantics involved in the grammar with the visual appearance of the associated architectural elements. This idea was first investigated in the context of 2D analysis from a

single ortho-rectified view of a facade. Disregarding technical considerations by using simplifying assumptions, a hill climbing inference method revealed the potentials of the formulation. Then, to handle more complex cases where notably 3D modeling is at stake, we have extended our approach both in terms of visual cues as well as in terms of inference. Due to the fact that appearance models are not suitable for accurate depth recovery, we have introduced a concurrent likelihood model based on depth profiles derived from a classical multi-view reconstruction. To cope with the multi-objective problem implied by this extension, an evolutionary algorithm was considered aiming at the estimation of the so-called Pareto frontier. Besides, we presented an automatic scheme to select a single optimal layout among the non-dominated solutions. It appeared that this two-step process was successful in combining the cues induced by both objective functions.

Our approach has been tested on both single-view 2D facade parsing and 3D building reconstruction. In the first case, we provided a benchmark of annotated facade views. We recommend the use of this data-set for researchers willing to compete in the same type of structured segmentation problems since it involves facades of high complexity and implies challenging situations (such as large cast shadows and color saturation). In the second case, the approach was evaluated with respect to classification and to reconstruction on a smaller data-set.

6.2 Future Directions

The philosophy sustained in this work was natural given previous research but it has raised many difficulties. We have provided promising responses, yet various extensions are under consideration to push forward the scope of our work.

The first extension concerns the definition of more complex grammars to encompass a broader class of buildings. The Haussmannian grammar could be easily simplified to embrace a generic class of buildings encountered in various places. Yet, more complex styles would require additional features in the grammar for instance to model bay windows, rounded corners, *etc.* . As a consequence, even the procedural framework would need to be extended. We have mentioned that the use of geometric languages such as GML would be of a great value in this regard.

As a second direction, we strongly support the automatic design and inference of grammars from sample data. It is certainly the counterpart of this work that will make grammatical descriptions an obvious choice for large scale modeling. Indeed, so far a non negligible part of the modeling pipeline consists in creating the grammar describing the class of buildings to be reconstructed. Automation in this regard would make the approach even more appealing. The issues raised in this context are of very high complexity. We believe that such inference would necessarily imply stronger reliance in bottom-up techniques so as to extract local relations between elementary parts and aggregate this knowledge in the inferred grammar.

Then various improvements could be sought from the parsing point of view as well. Introducing bottom-up cues would certainly speed-up convergence. One way to do so in our evolutionary algorithm is to introduce data-driven mutations. Moreover, with respect to the derivation of better appearance models, features based on recent descriptors could be used to enhance the classifiers

performance. Other type of energies could be considered as well, for instance to enforce visual similarity between the instances of the same architectural element. Another very appealing variation of our work would account explicitly for physical constraints between the elements of the grammar.

Given the important industrial context attached to this work, much attention should be paid to pre/post-processing steps. Some of these steps as the data capture protocol are purely practical. On the contrary, 3D model enhancement, which has been partly discussed in this work, raises further research topics. Various tasks can be undertaken in this respect. One aspect concerns geometry refinement which is necessary to transform 3D layouts into detailed models. The solution proposed in our approach could be extended to integrate more variability among the atomic architectural elements and to include hybrid reconstructions. In addition, textures plays a key role in the visual aspect of the reconstructed model. Instead of using raw inputs, an optimal texture could be derived from a complementary inference process. The purpose of such process would be to remove visual artifacts induced by occlusions and cast shadows.

On a more general note, grammars can be used for other applications in computer vision and medical images where structure is of central interest. This has already been the case in the past with respect to image understanding. Beyond computer vision, we hope that, in their seeking for hierarchical representations of physical objects and for efficient ways to infer their internal degrees of freedom, our work will inspire other researchers.

Publications of the Author

International Journals

- Loic Simon, Olivier Teboul, Panagiotis Koutsourakis and Nikos Paragios. *Random Exploration of the Procedural Space for Single-View 3D Modeling of Buildings*. International Journal of Computer Vision (IJCV), 2010.

International Conferences

- P Koutsourakis, L Simon, O Teboul, G Tziritas and N Paragios. *Single view reconstruction using shape grammars for urban environments*. In IEEE International Conference on Computer Vision (ICCV). IEEE, 2009.

- O Teboul, L Simon, P Koutsourakis and N Paragios. *Segmentation of building facades using procedural shape priors*. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2010. **Oral presentation.**

- Olivier Teboul, Iasonas Kokkinos, Panagiotis Koutsourakis, Loic Simon and Nikos Paragios. *Shape Grammar Parsing via Reinforcement Learning*. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2011.

Bibliography

[Abelson 1981] Harold Abelson and Andrea A DiSessa. Turtle Geometry. The MIT Press, 1981. 54

[Agarwal 1998] M Agarwal and J Cagan. *A blend of different tastes: the language of coffeemakers*. Environment and Planning B Planning and Design, vol. 25, no. 2, pages 205–226, 1998. 61

[Agarwal 2009] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M Seitz and Richard Szeliski. *Building Rome in a day*. 2009 IEEE 12th International Conference on Computer Vision, no. Iccv, pages 72–79, 2009. 31

[Aichholzer 1995] Oswin Aichholzer, Franz Aurenhammer, David Alberts and Bernd Gartner. *A novel type of skeleton for polygons*. Journal Of Universal Computer Science, vol. 1, no. 12, pages 752–761, 1995. 72

[Alegre 2004] Fernando Alegre and Frank Dellaert. *A probabilistic approach to the semantic interpretation of building facades*. In International Workshop on Vision Techniques Applied to the Rehabilitation of City Centres, volume 2, page 3. Citeseer, Citeseer, 2004. 42, 45, 46

[Arya 1998] Sunil Arya, David M Mount, Nathan S Netanyahu, Ruth Silverman and Angela Y Wu. *An optimal algorithm for approximate nearest neighbor searching fixed dimensions*. Journal of the ACM, vol. 45, no. 6, pages 891–923, 1998. 25

[Baillard 1999] Caroline Baillard and Andrew Zisserman. *Automatic reconstruction of piecewise planar models from multiple views*. In IEEE Conference on Computer Vision and Pattern Recognition, volume 2, pages 559–565, Fort Collins, USA, 1999. IEEE Comput. Soc. 35

[Ballard 1982] Dana H Ballard and Christopher M Brown. Computer Vision, volume 1982. Prentice Hall, 1982. 23

[Becker 2009] Susanne Becker and Norbert Haala. *Grammar supported facade reconstruction from mobile lidar mapping*. ISPRS Workshop CMRT09 City Models Roads and Traffic, vol. XXXVIII, no. 2007, pages 229–234, 2009. 42, 47

[Bishop 2006] Christopher M Bishop. Pattern Recognition and Machine Learning, volume 4 of *Information science and statistics*. Springer, 2006. 97

[Blake 2004] Andrew Blake, Carsten Rother, M Brown, Patrick Perez and Philip Torr. *Interactive image segmentation using an adaptative GMMRF model*. In European Conference on Computer Vision ECCV, pages 428–441, 2004. 92, 97

[Bokeloh 2010] Martin Bokeloh, Michael Wand and Hans-Peter Seidel. *A Connection between Partial Symmetry and Inverse Procedural Modeling*. ACM Transactions on Graphics, vol. 29, no. 4, page 1, 2010. 42

[Breiman 2001] L Breiman. *Random Forests*. Machine Learning, vol. 45, no. 1, pages 5–32, 2001. 92

[Chen 2008] Guoning Chen, Gregory Esch, Peter Wonka, Pascal Müller and Eugene Zhang. *Interactive procedural street modeling*. ACM Transactions on Graphics, vol. 27, no. 3, page 1, 2008. 61

[Chomsky 1956] Noam Chomsky. *Three models for the description of language*. Ire Transactions On Information Theory, vol. 2, no. 3, pages 113–124, 1956. 52

[Chomsky 1957] Noam Chomsky. Syntactic Structures, volume 33 of *Janua linguarum. Studia memoriae ; nr. 4.* Mouton, 1957. 50

[Chomsky 1963] N Chomsky and Marcel P Schutzenberger. The algebraic theory of context-free languages, pages 118–161. Elsevier, 1963. 50

[Coelho 2007] A Coelho, M Bessa, A A Sousa and F N Ferreira. *Expeditious Modelling of Virtual Urban Environments with Geospatial L-systems*. Computer Graphics Forum, vol. 26, no. 4, pages 769–782, 2007. 61

[Coello 2007] C A C Coello, G B Lamont and D A Van Veldhuizen. Evolutionary algorithms for solving multi-objective problems. Springer-Verlag New York Inc, 2007. 113

[Collins 1992] Robert T Collins. *Projective reconstruction of approximately planar scenes*. In Proc. SPIE, pages 174–185. Citeseer, 1992. 39

[De La Gorce 2008] Martin De La Gorce, Nikos Paragios and David J Fleet. *Model-based hand tracking with texture, shading and self-occlusions*. IEEE Conference on Computer Vision and Pattern Recognition (2008), no. June, pages 1–8, 2008. 32

[Debevec 1996] Paul Debevec, Camillo J Taylor and Jitendra Malik. *Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach*. In Annual Conference on Computer Graphics and Interactive Techniques, page 20. ACM, 1996. 37, 38, 39, 41

[Delaunoy 2008] A Delaunoy, E Prados, P Gargallo, J P Pons and P Sturm. *Minimizing the Multi-view Stereo Reprojection Error for Triangular Surface Meshes*. British Machine Vision Conference, 2008. 32

[Delong 2010] Andrew Delong, A. Osokin, H.N. Isack and Y. Boykov. *Fast approximate energy minimization with label costs*. International Journal of Computer Vision, no. 1, 2010. 36

[Dick 2001] A R Dick, P H S Torr, S J Ruffle and R Cipolla. *Combining Single View Recognition and Multiple View Stereo For Architectural Scenes*. In IEEE International Conference on Computer Vision, pages 268–280, Vancouver, Canada, 2001. IEEE Computer Society. 38

[Dick 2004] Anthony R Dick, Philip H S Torr and Roberto Cipolla. *Modelling and Interpretation of Architecture from Several Images*. International Journal of Computer Vision, vol. 60, no. 2, pages 111–134, 2004. 38

[Duan 2004] Ye Duan, Liu Yang, Hong Qin and Ddimitris Samaras. *Shape reconstruction from 3D and 2D data using PDE-based deformable surfaces*. In European Conference on Computer Vision, volume 3, pages 238–251, Prague, Czech Republic, 2004. 33

[Dyer 2001] Charles R Dyer. *Volumetric Scene Reconstruction from Multiple Views*. In L.∼S. Davis, editeur, Foundations of Image Understanding, pages 469–489. Kluwer, 2001. 27, 28

[Dylla 2008] Kimberly Dylla, Bernard Frischer, P. Mueller, Andreas Ulmer and Simon Haegler. *Rome Reborn 2.0: A Case Study of Virtual City Reconstruction Using Procedural Modeling Techniques*. Computer Graphics World, vol. 16, page 25, 2008. 63

[Eppstein 1999] D Eppstein and J Erickson. *Raising roofs, crashing cycles, and playing pool: Applications of a data structure for finding pairwise interactions*. Discrete and Computational Geometry, vol. 22, no. 4, pages 569–592, 1999. 72

[Faugeras 1992] Olivier D Faugeras. *What can be seen in three dimensions with an uncalibrated stereo rig?* In European Conference on Computer Vision, volume 588 of *LNCS 588*, pages 563–578, Santa Margherita Ligure, Italy, 1992. 25

[Faugeras 1998] Olivier D Faugeras and Renaud Keriven. *Variational principles, surface evolution, PDE's, level set methods, and the stereo problem.* In IEEE Transactions on Image Processing, volume 7, pages 336–344, Bombay, India, 1998. Inst. Nat. de Recherche en Inf. et Autom., Sophia-Antipolis, France. Olivier.Faugeras@sophia.inria.fr. 29

[Flemming 1987] U Flemming. *More than the sum of parts: the grammar of Queen Anne houses*. Environment and Planning B Planning and Design, vol. 14, no. 3, pages 323–350, 1987. 56

[Furukawa 2007] Yasutaka Furukawa and Jean Ponce. *Accurate, Dense, and Robust Multi-View Stereopsis*. In IEEE Conference on Computer Vision and Pattern Recognition, volume 1, pages 1–8, Minneapolis, USA, 2007. Ieee. 30, 31, 33, 35

[Furukawa 2008] Yasutaka Furukawa and Jean Ponce. *Patch-based Multi-view Stereo Software (PMVS - Version 2)*. http://grail.cs.washington.edu/software/pmvs/, 2008. 31, 112

[Furukawa 2009] Yasutaka Furukawa, Brian Curless, Steven M Seitz and Richard Szeliski. *Manhattan-world stereo*. In IEEE Conference on Computer Vision and Pattern Recognition, volume 0, pages 1422–1429, Miami, USA, 2009. Ieee. 35, 36

[Gallup 2007] David Gallup, J.M. Frahm, Philippos Mordohai, Qingxiong Yang and Marc Pollefeys. *Real-time plane-sweeping stereo with multiple sweeping directions*. In IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8, Minneapolis, USA, 2007. IEEE. 30

[Gallup 2008] David Gallup, Jan-Michael Frahm, Philippos Mordohai, Marc Pollefeys and Chapel Hill. *Variable Baseline / Resolution Stereo*. In IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8, Anchorage, USA, 2008. Ieee. 30

[Gallup 2010] David Gallup, Jan-Michael Frahm and Marc Pollefeys. *Piecewise planar and non-planar stereo for urban scene reconstruction*. In IEEE Conference on Computer Vision and Pattern Recognition, pages 1418–1425, San Francisco, USA, 2010. IEEE. 36, 39

[Gargallo 2005] Paul Gargallo and Peter Sturm. *Bayesian 3D Modeling from Images using Multiple Depth Maps*. In IEEE Conference on Computer Vision and Pattern Recognition, volume 2, pages 885–891, San Diego, USA, 2005. Ieee. 30

[Green 1995] P J Green. *Reversible jump Markov chain Monte Carlo computation and Bayesian model determination*. Biometrika, vol. 82, no. 4, pages 711–732, 1995. 45

[Grenander 1994] U Grenander and M I Miller. *Representations of Knowledge in Complex Systems*. Journal of the Royal Statistical Society Series B Methodological, vol. 56, no. 4, pages 549–603, 1994. 40

[Greuter 2003] Stefan Greuter, Jeremy Parker, Nigel Stewart and Geoff Leach. *Real-time procedural generation of 'pseudo infinite' cities*. In International conference on Computer graphics and interactive techniques in Australasia and South East Asia, pages 87–95, New York, New York, USA, 2003. Citeseer. 61, 62

[Gupta 2010] Abhinav Gupta, Alexei A Efros and Martial Hebert. *Blocks World Revisited : Image Understanding Using Qualitative Geometry and Mechanics*. Robotics, vol. 6314, pages 482–496, 2010. 44

[Han 2004] Feng Han and Song-Chun Zhu. *Automatic Single View Building Reconstruction by Integrating Segmentation*. In IEEE Conference on Computer Vision and Pattern Recognition, volume 00, pages 53–53, Washington, USA, 2004. Ieee. 46

[Han 2005] Feng Han and Song-Chun Zhu. *Bottom-up/Top-Down Image Parsing by Attribute Graph Grammar*. In IEEE International Conference on Computer Vision, volume 2, pages 1778–1785, Beijing, China, 2005. Ieee. 46

[Hart 1992] John C Hart. *The object instancing paradigm for linear fractal modeling*. In Proceedings of Graphics Interface 92, pages 224–231, 1992. 59, 62

[Hartley 1992] Richard I Hartley, Rajiv Gupta and Tom Chang. *Stereo from uncalibrated cameras*. In IEEE Conference on Computer Vision and Pattern Recognition, pages 761–764, Champaign, USA, 1992. Citeseer. 25

[Hartley 2004] Richard I Hartley and Andrew Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, 2004. 25, 85

[Havemann 2001] Sven Havemann and Dieter Fellner. *A versatile 3D model representation for cultural reconstruction*. International Symposium on Virtual Reality, Archeology and Cultural Heritage (VAST), page 205, 2001. 63

[Havemann 2004] Sven Havemann and Dieter Fellner. *Generative parametric design of Gothic window tracery*. Proceedings Shape Modeling Applications, pages 350–353, 2004. 63

[Havemann 2005] Sven Havemann. *Generative Mesh Modeling*. PhD thesis, Technische Universität Braunschweig, 2005. 63

[Hernandez Esteban 2004] C Hernandez Esteban and F Schmitt. *Silhouette and stereo fusion for 3D object modeling*. Computer Vision and Image Understanding, vol. 96, no. 3, pages 367–392, 2004. 29

[Hiep 2009] Vu Hoang Hiep, Renaud Keriven, Patrick Labatut and Jean-Philippe Pons. *Towards high-resolution large-scale multi-view stereo*. In IEEE Conference on Computer Vision and Pattern Recognition, pages 1430–1437, Miami, USA, 2009. Ieee. 33

[Hohmann 2009] B Hohmann, U Krispel, S Havemann and D Fellner. *CITYFIT : HIGH-QUALITY URBAN RECONSTRUCTIONS BY FITTING SHAPE GRAMMARS TO IMAGES AND DERIVED TEXTURED POINT CLOUDS*. Proceedings of the International Workshop 3DARCH 2009, vol. XXXVIII, no. Part 5/W1, pages 2007–2007, 2009. 63

[Ilcik 2010] Martin Ilcik, Stefan Fiedler, Werner Purgathofer and Michael Wimmer. *Procedural Skeletons: Kinematic Extensions to CGA-Shape Grammars*. Proceedings of the Spring Conference on Computer Graphics 2010, pages 177–184, 2010. 63

[Kolmogorov 2002] V Kolmogorov and R Zabih. *Multi-camera scene reconstruction via graph cuts*. Lecture Notes in Computer Science, vol. vol, pages 82–96, 2002. 30

[Koutsourakis 2009] P Koutsourakis, L Simon, O Teboul, G Tziritas and N Paragios. *Single view reconstruction using shape grammars for urban environments*. In IEEE International Conference on Computer Vision (ICCV). IEEE, 2009.

[Kutulakos 1998] K.N. Kutulakos and S.M. Seitz. *What do N photographs tell us about 3d shape?* University of Rochester, TR680, 1998. 29

[Kutulakos 2000] Kiriakos N Kutulakos and Steven M Seitz. *A Therory of Shape by Space Carving*. International Journal of Computer Vision, vol. 3, no. 38, pages 197–216, 2000. 29

[Lafarge 2009] Florent Lafarge, Renaud Keriven and Mathieu Brédif. *Combining meshes and geometric primitives for accurate and semantic modeling*. In Proceedings of British Machine Vision Conference, pages 1–11, 2009. 39, 40

[Lafarge 2010] Florent Lafarge, Renaud Keriven, Mathieu Brédif and Vu Hoang Hiep. *Hybrid multi-view reconstruction by Jump-Diffusion*. In IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, USA, 2010. IEEE. 39, 40

[Larive 2006] Mathieu Larive and Veronique Gaildrat. *Wall grammar for building generation*. In Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia, pages 429–437. ACM, 2006. 62

[Legakis 2001] Justin Legakis, Julie Dorsey and Steven Gortler. *Feature-based cellular texturing for architectural models*. ACM Transactions on Graphics, pages 309–316, 2001. 73

[Lepetit 2006] Vincent Lepetit and Pascal Fua. *Keypoint Recognition Using Randomized Trees*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 9, pages 1465–1479, 2006. 92, 95

[Lhuillier 2002] M Lhuillier and L Quan. *Match propagation for image-based modeling and rendering*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 8, pages 1140–1146, 2002. 31

[Lhuillier 2005] Maxime Lhuillier and Long Quan. *A quasi-dense approach to surface reconstruction from uncalibrated images*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 3, pages 418–433, 2005. 30, 31

[Lindenmayer 1968] Aristid Lindenmayer. *Mathematical models for cellular interaction in development*. Journal of Theoretical Biology, 1968. 53

[Lipp 2008] Markus Lipp, Peter Wonka and Michael Wimmer. *Interactive Visual Editing of Grammars for Procedural Architecture*. ACM Transactions on Graphics, vol. 27, no. 3, pages 102:1—-10, 2008. 63

[Lourakis 2004] Manolis I. A. Lourakis and Abtonis A. Argyros. *The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm*. ICSFORTH Technical Report TR, vol. 340, no. 340, 2004. 25

[Lourakis 2010] Manolis I. A. Lourakis. *Sparse non-linear least squares optimization for geometric vision*. European Conference on Computer Vision, pages 43–56, 2010. 25

[Luke 2009] Sean Luke. Essentials of Metaheuristics. Lulu, 2009. 113

[Marshall 2001] Andrew David Marshall, Ga'bor Lukacs and Ralph Robert Martin. *Robust Segmentation of Primitives from Range Data in the Presence of Geometric Degeneracy*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 3, pages 304–314, 2001. 40

[Marvie 2005] Jean-Eudes Marvie, Julien Perret and Kadi Bouatouch. *The FL-system: a functional L-system for procedural geometric modeling*. The Visual Computer, vol. 21, no. 5, pages 329–339, May 2005. 59, 62

[Merrell 2007] Paul Merrell. *Example-based model synthesis*. In Symposium on Interactive 3D graphics and games, volume 2008, pages 105–112. ACM, 2007. 42

[Müller 2006a] Pascal Müller, Tijl Vereenooghe, Peter Wonka, Iken Paap and Luc Van Gool. *Procedural 3D Reconstruction of Puuc Buildings in Xkipché*. International Symposium on Virtual Reality, Archeology and Cultural Heritage (VAST), pages 139–146, 2006. 63

[Müller 2006b] Pascal Müller, Peter Wonka, Simon Haegler, Andreas Ulmer and Luc Van Gool. *Procedural modeling of buildings*. ACM Transactions on Graphics, vol. 25, no. 3, page 614, July 2006. 59, 62, 65

[Müller 2007] Pascal Müller, Gang Zeng, Peter Wonka and Luc Van Gool. *Image-based procedural modeling of facades*. ACM Transactions on Graphics, vol. 26, no. 3, page 85, 2007. 42, 43

[Musialski 2009] Przemyslaw Musialski, Peter Wonka, Meinrad Recheis, Stefan Maierhofer and Werner Purgathofer. *Symmetry-based facade repair*. In Marcus A Magnor, Bodo Rosenhahn and Holger Theisel, editeurs, Vision, Modeling, and Visualization Workshop 2009 in Braunschweig, Germany, pages 3–10. Citeseer, 2009. 42, 43

[Musialski 2010] Przemyslaw Musialski, Meinrad Recheis, Stefan Maierhofer, Peter Wonka and Werner Purgathofer. *Tiling of Ortho-Rectified Façade Images*. In Spring Conference on Computer Graphics SCCG10, 2010. 42, 43

[Nan 2010] Liangliang Nan, Andrei Sharf, Hao Zhang, Daniel Cohen-Or and Baoquan Chen. *SmartBoxes for interactive urban reconstruction*. ACM Transactions on Graphics, vol. 29, no. 4, page 1, 2010. 38, 41

[Narayanan 1998] P J Narayanan, P W Rander and T Kanade. *Constructing Virtual Worlds using Dense Stereo*. In IEEE International Conference on Computer Vision, pages 3–10, Bombay, India, 1998. IEEE Computer Society, Narosa Publishing House. 30

[Newell 1975] M Newell. *The Utilization of Procedure Models in Digital Image Synthesis*, 1975. 49

[Parish 2001] Yoav I H Parish and Pascal Müller. *Procedural modeling of cities*. In ACM Transactions on Graphics, pages 301–308, 2001. 59, 61, 62

[Pauly 2008] Mark Pauly, Niloy J Mitra, Johannes Wallner, Helmut Pottmann and Leonidas J Guibas. *Discovering structural regularity in 3D geometry*. ACM Transactions on Graphics, vol. 27, no. 3, page 1, 2008. 44

[Pinkney 1958] David H Pinkney. Napoleon III and the Rebuilding of Paris. Princeton University Press, 1958. 78

[Pollefeys 1999] Marc Pollefeys. *Self-calibration and metric 3D reconstruction from uncalibrated image sequences*. International Journal of Computer Vision, 1999. 25

[Pons 2007a] Jean-Philippe Pons and Jean-Daniel Boissonnat. *Delaunay Deformable Models: Topology-Adaptive Meshes Based on the Restricted Delaunay Triangulation*. In IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8, Minneapolis, USA, 2007. Ieee. 30, 33

[Pons 2007b] Jean-Philippe Pons, Renaud Keriven and Olivier Faugeras. *Multi-View Stereo Reconstruction and Scene Flow Estimation with a Global Image-Based Matching Score*. International Journal of Computer Vision, vol. 72, no. 2, pages 179–193, 2007. 29

[Prunsinkiewicz 1996] Przemyslaw Prunsinkiewicz and Aristid Lindenmayer. The Algorithmic Beauty of Plants. Springer-Verlag, 1996. 55

[Prusinkiewicz 1986] P Prusinkiewicz. *Graphical applications of L-systems*. In Proceedings of Graphics Interface 86 Vision Interface 86, volume terface'86, pages 247–253, 1986. 54

[Pugliese 2002] Michael J Pugliese and Jonathan Cagan. *Capturing a rebel : modeling the Harley-Davidson brand through a motorcycle shape grammar*. Research in Engineering Design, vol. 13, no. April, pages 139–156, 2002. 61

[Ripperda 2006] Nora Ripperda and Claus Brenner. *Reconstruction of Façade Structures Using a Formal Grammar and RjMCMC*. In Katrin Franke, Klaus-Robert Müller, Bertram Nickolay and Ralf Schäfer, editeurs, Annual Symposium of the German Association for Pattern Recognition DAGM, volume 4174 of *Lecture Notes in Computer Science*, pages 750–759. Springer Berlin / Heidelberg, 2006. 45, 47

[Ripperda 2007] Nora Ripperda and Claus Brenner. *Data Driven Rule Proposal for Grammar Based Facade Reconstruction*. In PIA07, page 1, 2007. 47

[Schnier 1996] T Schnier and John S Gero. *Learning genetic representations as alternative to hand-coded shape grammars*. Artificial Intelligence in Design, 1996. 42

[Seitz 1997] Steven M Seitz and Charles R Dyer. *Photorealistic scene reconstruction by voxel coloring*. In IEEE Conference on Computer Vision and Pattern Recognition, volume 35, pages 1067–1073, San Juan, USA, 1997. 28

[Seitz 2006] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein and Richard Szeliski. *A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms*. In IEEE Conference on Computer Vision and Pattern Recognition, volume 1, pages 519–528, New York City, USA, 2006. IEEE, Ieee. 27, 31

[Shotton 2008] Jamie Shotton, Matthew Johnson and Roberto Cipolla. *Semantic texton forests for image categorization and segmentation*. IEEE Conference on Computer Vision and Pattern Recognition (2008), pages 1–8, 2008. 92

[Simon 2010] Loic Simon, Olivier Teboul, Panagiotis Koutsourakis and Nikos Paragios. *Random Exploration of the Procedural Space for Single-View 3D Modeling of Buildings*. International Journal of Computer Vision (IJCV), 2010.

[Sinha 2009] S.N. Sinha, Drew Steedly and Richard Szeliski. *Piecewise planar stereo for image-based rendering*. In IEEE International Conference on Computer Vision, pages 1881–1888, Kyoto, Japan, 2009. 36

[Skolicki 2008] Z Skolicki and K De Jong. *The importance of a two-level perspective for island model design*. In Evolutionary Computation, 2007. CEC 2007. IEEE Congress on, pages 4623–4630. IEEE, 2008. 116

[Slabaugh 2001] Greg Slabaugh, Bruce Culbertson, Tom Malzbender and Ron Schafer. *A survey of methods for volumetric scene reconstruction from photographs*. International Workshop on Volume Graphics, vol. 2, no. 7, pages 81–100, 2001. 27, 28

[Slabaugh 2005] Greg Slabaugh and Gozde Unal. *Active Polyhedron: Surface Evolution Theory Applied to Deformable Meshes*. In IEEE Conference on Computer Vision and Pattern Recognition, volume 2, pages 84–91, San Diego, USA, 2005. Ieee. 32

[Snavely 2006a] Noah Snavely. *Bundler: Structure from Motion (SfM) for Unordered Image Collections*. http://phototour.cs.washington.edu/bundler/, 2006. 25, 112

[Snavely 2006b] Noah Snavely, Steven M Seitz and Richard Szeliski. *Photo tourism: exploring photo collections in 3D*. ACM SIGGRAPH 2006 Papers, vol. 25, no. 3, pages 835–846, 2006. 25

[Snavely 2007] Noah Snavely, Steven M Seitz and Richard Szeliski. *Modeling the World from Internet Photo Collections*. International Journal of Computer Vision, vol. 80, no. 2, pages 189–210, 2007. 25

[Stiny 1972] George Stiny and James Gips. *Shape Grammars and the Generative Specification of Painting and Sculpture*. In C V Freiman, editeur, Information Processing 71, volume 71, pages 1460–1465. North-Holland, 1972. 55, 58

[Stiny 1978] George Stiny and William J Mitchell. *The Palladian grammar*. Environment And Planning B, vol. 5, no. 1, pages 5–18, 1978. 56

[Stiny 1982] George Stiny. *Spatial Relations and grammars*. Environment and Planning B Planning and Design, vol. 9, no. 1, pages 113–114, 1982. 57

[Strecha 2008] C Strecha, W Von Hansen, L Van Gool, P Fua and U Thoennessen. *On benchmarking camera calibration and multi-view stereo for high resolution imagery*. In IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8, Anchorage, USA, 2008. Ieee. 31

[Strecha 2010a] Christoph Strecha, Alexander M Bronstein, Michael M Bronstein and Pascal Fua. *Technical Report LDAHash : Improved matching with smaller descriptors*. Technical report, 2010. 31

[Strecha 2010b] Christoph Strecha and Pascal Fua. *Dynamic and Scalable Large Scale Image Reconstruction*. In IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, USA, 2010. IEEE. 112

[Teboul 2010] O Teboul, L Simon, P Koutsourakis and N Paragios. *Segmentation of building facades using procedural shape priors*. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2010. 46

[Teboul 2011] Olivier Teboul, Iasonas Kokkinos, Panagiotis Koutsourakis, Loic Simon and Nikos Paragios. *Shape Grammar Parsing via Reinforcement Learning*. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2011. 126

[Tola 2010] Engin Tola, Vincent Lepetit and Pascal Fua. *DAISY: an efficient dense descriptor applied to wide-baseline stereo.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 5, pages 815–830, 2010. 31

[Toshev 2010] A Toshev, P Mordohai and B Taskar. *Detecting and parsing architecture at city scale from range data*. In IEEE Conference on Computer Vision and Pattern Recognition, pages 398–405, San Francisco, USA, 2010. 47

[Triggs 2000] Bill Triggs, Philip F Mclauchlan, Richard I Hartley and Andrew W Fitzgibbon. *Bundle Adjustment - A Modern Synthesis*. Synthesis, vol. 34099, pages 298–372, 2000. 25

[Van Gool 2007] Luc Van Gool, Gang Zeng, Filip Van Den Borre and Pascal Müller. *Towards Mass-produced Building Models*. In U Stilla, H Mayer, F Rottensteiner, C Heipke and S Hinz, editeurs, PIA07 International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences, pages 209–220. Institute of Photogrammetry and Cartography, Technische Universitaet Muenchen, 2007. 43

[Vanegas 2010] Carlos A Vanegas, Daniel G Aliaga and Bedrich Benes. *Building Reconstruction using Manhattan-World Grammars*. In IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, USA, 2010. 45, 62

[Vergauwen 2006a] Maarten Vergauwen. *ARC 3D Webservice: A Family of Web Tools for Remote 3D Reconstruction*. http://homes.esat.kuleuven.be/∼visit3d/webservice/v2/links.php, 2006. 25

[Vergauwen 2006b] Maarten Vergauwen and Luc Van Gool. *Web-based 3D Reconstruction Service*. Machine Vision and Applications, vol. 17, no. 6, pages 411–426, 2006. 25, 30

[Viola 2001] Paul Viola and Michael Jones. *Robust Real-time Object Detection*. International Journal of Computer Vision, vol. 57, no. 2, pages 137–154, 2001. 90

[Vogiatzis 2005] George Vogiatzis, Philip Torr and Roberto Cipolla. *Multi-view stereo via volumetric graph cuts*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 12, pages 2241–6, 2005. 29

[Vogiatzis 2007] George Vogiatzis, Carlos Hernández Esteban, Philip H S Torr and Roberto Cipolla. *Multiview stereo via volumetric Graph-Cuts and occlusion robust photo-consistency.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 12, pages 2241–2246, 2007. 29

[Šťava 2010] O Šťava, B Beneš, R Měch, Daniel G Aliaga and P Krištof. *Inverse Procedural Modeling by Automatic Generation of L-systems*. Computer Graphics Forum, vol. 29, no. 2, pages 665–674, 2010. 42

[Weber 2009] Basil Weber, Pascal Müller, Peter Wonka and Markus Gross. *Interactive Geometric Simulation of 4D Cities*. Computer Graphics Forum, vol. 28, no. 2, pages 481–492, 2009. 61

[Wells 1996] W M Wells, P Viola, H Atsumi, S Nakajima and R Kikinis. *Multi-modal volume registration by maximization of mutual information.*, 1996. 43

[Werner 2002] T Werner and Andrew Zisserman. *New Techniques for Automated Architecture Reconstruction from Photographs*. In European Conference on Computer Vision, volume 2, pages 541–555, Copenhagen, Denmark, 2002. 36, 38

[Whiting 2009] Emily Whiting, John Ochsendorf and Frédo Durand. *Procedural modeling of structurally-sound masonry buildings*. ACM Transactions on Graphics, vol. 28, no. 5, page 1, December 2009. 63

[Wilczkowiak 2005] Marta Wilczkowiak, Peter Sturm and Edmond Boyer. *Using geometric constraints through parallelepipeds for calibration and 3D modeling.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 2, pages 194–207, 2005. 37

[Winn 2006] J Winn and Jamie Shotton. *The Layout Consistent Random Field for Recognizing and Segmenting Partially Occluded Objects*. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Volume 1 CVPR06, vol. 1, no. c, pages 37–44, 2006. 92

[Wonka 2003] Peter Wonka, Michael Wimmer, François Sillion and William Ribarsky. *Instant architecture*. ACM Transactions on Graphics, vol. 22, no. 3, page 669, 2003. 59, 62

[Wu 2010] Changchang Wu, Jan-Michael Frahm and Marc Pollefeys. *Detecting Large Repetitive Structures with Salient Boundaries*. In European Conference on Computer Vision, volume 6312 of *Lecture Notes in Computer Science*, pages 142–155, Heraklion, Crete, Greece, 2010. 42, 43

[Xiao 2008] Jianxiong Xiao, Tian Fang, Ping Tan, Peng Zhao, Eyal Ofek and Long Quan. *Image-based façade modeling*. ACM Transactions on Graphics, vol. 27, no. 5, page 1, December 2008. 43, 45

[Xiao 2009] Jianxiong Xiao, Tian Fang, Peng Zhao, Maxime Lhuillier and Long Quan. *Image-based street-side city modeling*. ACM Transactions on Graphics, vol. 28, no. 5, page 1, 2009. 45

[Zaharescu 2010] Andrei Zaharescu, Edmond Boyer and Radu Horaud. *Topology-Adaptive Mesh Deformation for Surface Evolution, Morphing and Multi-View Reconstruction.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 4, pages 823–837, 2010. 33

[Zhu 2000] Song-Chun Zhu, R Zhang and Z Tu. *Integrating bottom-up/top-down for object recognition by data driven Markov chain Monte Carlo*. In IEEE Conference on Computer Vision and Pattern Recognition, numéro October, pages 738–745, Hilton Head, USA, 2000. 46

[Zhu 2006] Song-Chun Zhu and D Mumford. *A Stochastic Grammar of Images*. Foundations and Trends® in Computer Graphics and Vision, vol. 2, no. 4, pages 259–362, 2006. 47

[Zitzler 2001] E Zitzler, M Laumanns, L Thiele and Others. *SPEA2: Improving the strength Pareto evolutionary algorithm*. In Eurogen, volume 3242. Citeseer, 2001. 119