TOWARDS MASS-PRODUCED BUILDING MODELS

Luc Van Gool[a,b,*] **Gang Zeng**[a] **Filip Van den Borre**[b] **Pascal Müller**[a]

[a]ETH Zürich, (vangool, zengg, pmueller)@vision.ee.ethz.ch
[b]KU Leuven, (Luc.VanGool, Filip.VandenBorre)@esat.kuleuven.be

Commission III/5

KEY WORDS: Facade 3D Reconstruction, Procedural Modeling, Shape Grammar, Repetition Detection

ABSTRACT:

Interest in the automatic production of 3D building models has increased over the last years. The reconstruction of buildings, particularly their facades, is a hard subproblem, given the large variety in their appearances and structures. This paper discusses building facade reconstruction algorithms that process single images and exploit expectations about facade composition. In particular, we make heavy use of the repetitions that tend to occur, e.g. in windows and balconies. But this is only an example of the kind of rules found in recent architectural shape grammars. We distinguish between cases without and with substantial perspective effects in the input image. The focus is on the latter case, where also some depth layering in the facade can be performed automatically. We give several examples of real building reconstructions.

1 INTRODUCTION

The photogrammetry, vision, and graphics communities have already invested enormous efforts into the creation of 3D models from images. As a result, much progress has been made already. And the body of literature is still growing at a fast pace. Yet, we have to ask ourselves how far pure bottom-up approaches - and these constitute the vast majority so far - can bring us. Even when presented with a single photograph, people can often make stronger statements about the 3D structure of the objects in it than our best 3D modeling systems can generate from multiple views of the same scene. Obviously, people have strong expectations about the world, and an exquisite capacity to recognize the objects that populate it. Such knowledge is not brought to bear in most of our 3D acquisition systems. Probably the 3D modeling of faces is the one and foremost example where researchers have actually drawn heavily on expectations about that particular object class (Blanz and Vetter, 1999), and very successfully so.

Hence, there is a case for a wider object class specific extraction of 3D information. In this paper the focus is on the use of such strategies for the important class of buildings. Also here, the benefits of using prior knowledge about this class has been demonstrated already, *e.g.* in (Debevec et al., 1996, Dick et al., 2001). With the rampant growth in geo-applications like Google Earth or Microsoft Virtual Earth and the fast evaluation towards 3D GPS navigation systems, buildings form a class of objects that does indeed deserve special attention. The creation of 3D city models still is a very interactive procedure. Any advance in productivity for the creation of such models would be extremely timely. Here we propose methods for the mass production of 3D facade models, exploiting knowledge about their typical composition.

1.1 Overview of the work

In this paper we propose to rely on architecture-oriented shape grammars. The first steps of this approach have been laid in a 2001 paper by Parish and Müller (Parish and Müller, 2001). A more full-fledged grammar for buildings was proposed in our later work (Müller et al., 2006). There it was shown that these shape grammars can be used to efficiently generate models of existing buildings, or of virtual buildings of a particular style. In that work we have for instance used the building footprints at Pompeii to generate extensive 3D models of the site. This could be achieved largely automatically, once the footprints had been delineated manually from archaeological maps. The modeling of existing buildings required more intensive interaction from the user though. In more recent work (Müller et al., 2007), we started to use photographs to automatically derive grammatical rules that could be used to re-create the facades of buildings. In that work, we mainly detected repetitive structures in the facade, and automatically delineated its structural entities like windows or doors. Moreover, templates were fitted to such entities. The result was a far more compact 3D representation of the facade, yet with higher visual quality. A tool was presented to manually displace selected structures as a group with respect to the main plane of the facade. As an example, the tool allows the user to jointly select all windows, and to simultaneously put them a bit deeper than the facade. The size of the shift is interactively estimated by comparing its result with what can be seen in the photographs. In the work presented here, we exploit the repetition in facades to actually measure the depth difference for the repeated elements.

Although it would be useful to look into multi-view reconstructions, as most contributions in this area did (see refs in section 1.2), we stick to single-view analysis as in (Müller et al., 2007), because such data is much easier to come by as yet. This said, the nature of the images we used in (Müller et al., 2007) and in this paper are different in nature. Our previous work mainly focused on the use of oblique aerial imagery, i.e. the type most often used to create large city models for the moment, and ground-level imagery without strong perspective effects. Here we fully focus on close-range photogrammetry type of data, where images of facades have strong perspective distortions. The latter are becoming available at a high pace.

This leads to a two-legged strategy. If there is sufficient perspective in the image, then we propose the fully automatic strategy as laid out in this paper. If perspective effects are too weak, we resort to the earlier, slightly interactive strategy of (Müller et al., 2007). Camera focal length would tell what to do, or so could the position of vanishing points. The latter are extracted for both strategies anyway. Obviously, the vanishing point criterion is more directly related to the appearance of the building in the image.

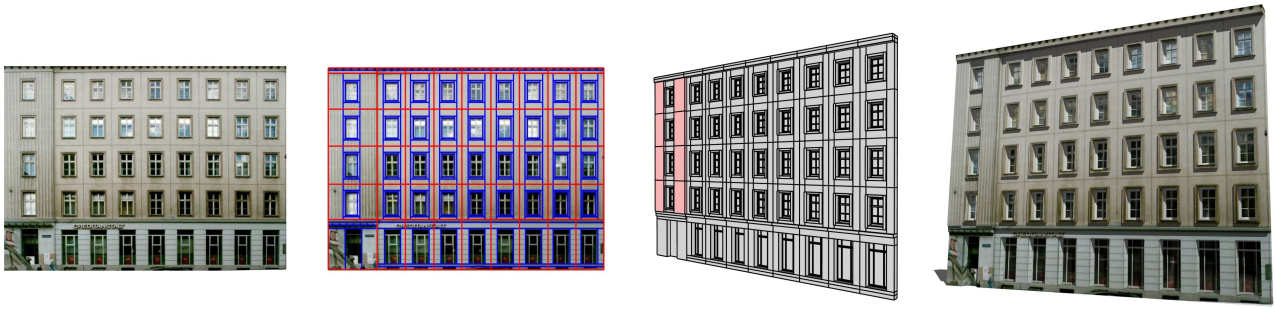

Figure 1: *The grammar based reconstruction can convert single facade textures of arbitrary resolution to semantic 3D models of high visual quality. Left: rectified facade image as input. Middle left: facade automatically subdivided and encoded as shape tree. Middle right: resulting polygonal model. Right: rendering of final reconstruction including shadows and reflections enabled by semantic information.*

The paper is organised as follows. In the remainder of this section, we give an overview of related work, to put these contributions into context. Then, section 2 recapitulates on the first strategy with weak perspective imagery, where the reader is referred to (Müller et al., 2007) for a more extensive account. Section 3 continues with the second strategy, where images have strong perspective effects. Section 4 concludes the paper.

1.2 Related Work

Shape Grammar Shape grammars were introduced by Stiny (Stiny, 1975) in the 70's as a formal approach to architectural design. They were successfully used for the construction and analysis of architectural design (Stiny and Mitchell, 1978, Koning and Eizenberg, 1981, Flemming, 1987, Duarte, 2002). The strategy of recent work was to simplify the geometric rules (Stiny, 1982), but to extend the derivation mechanisms (Parish and Müller, 2001, Wonka et al., 2003, Marvie et al., 2005, Müller et al., 2006). These shape grammars could be complemented by cellular textures (Legakis et al., 2001) to generate brick layouts and generative mesh modeling (Havemann, 2005) to generate facade ornaments. Many aspects and concepts of procedural architectural modeling are inspired by by L-systems (Prusinkiewicz and Lindenmayer, 1991), such as geometry sensitive rules (Prusinkiewicz et al., 1994), the incorporation of computer simulation (Mech and Prusinkiewicz, 1996) and artistic high-level control (Prusinkiewicz et al., 2001).

Building Facade Analysis Though limited, there is already some literature on the topic of building facade analysis. In practice, several systems still resort to semi-automatic methods (*e.g.* (Lee and Nevatia, 2003, Takase et al., 2003)). Generally, in these systems, a user is assisted by computer vision methods (Debevec et al., 1996) during modeling. This said, some automated processes have been proposed. Some of these make simplifying assumptions to get started. For example, Alegre and Dellaert (Alegre and Dellaert, 2004) as well as Brenner and Ripperda (Brenner and Ripperda, 2006) assume that windows basically correspond to dark rectangles. Others try to fit a limited set of rather complicated, parametrical models (Dick et al., 2001), or use detectors pre-trained for particular elements like windows (Mayer and Reznik, 2003). Finally, Lee and Nevatia (Lee and Nevatia, 2004) use a single ground-based image but their goal is restricted to windows.

Urban Reconstruction Urban reconstruction algorithms make use of a wide variety of input data, for example: ground-based facade images (Jepson et al., 1996, Debevec et al., 1996, REALVIZ, 2007, Lee et al., 2002, Dick et al., 2001, Wang et al., 2002), interactive editing using aerial images (Ribarsky et al., 2002), aerial images combined with ground-based panorama images (Wang et al., 2006), ground-based laser scans combined with aerial images (Früh and Zakhor, 2001), ground-based and airborne laser scans (Früh and Zakhor, 2003), ground-based laser scans combined with facade images (Karner et al., 2001), and laser scans, aerial images, and ground-based images (Hu et al., 2006). The problem is simplified if 3D data is available as depth displacements between elements (e.g. windows vs. walls) yield a strong, additional cue for their segmentation (Dick et al., 2004, Brenner and Ripperda, 2006, Schindler and Bauer, 2003).

2 STRATEGY 1: WEAK PERSPECTIVE

In this section we describe our strategy in case a facade image with small perspective effects is provided. Fig. 1 shows an overview of the procedure, from the input image on the left to the 3D result on the right. The procedure consists of four parts organized as *stages* in a pipeline. This pipeline transforms a single image into a textured 3D model including the semantic structure as a shape tree. We use a top-down hierarchical subdivision analogous to splitting rules in procedural facade modeling (Wonka et al., 2003, Bekins and Aliaga, 2005, Müller et al., 2006) (see Fig. 2). The following sections describe each of the four stages in this pipeline. Readers are referred to our Siggraph paper (Müller et al., 2007) for more technical details.

It is important to note that the facade image is rectified to a fronto-parallel view as a preprocessing step. We used an automatic rectification tool of our own implementation, which is a variant of the vanishing point based algorithm by Liebowitz and Zisserman (Liebowitz and Zisserman, 1998).

2.1 Determination of Facade Structure

The goal of this first stage is to detect the general structure in a facade and to subdivide it accordingly. The input is a single image and the output is a subdivision into floors and tiles. Additionally, we compute symmetry information so that we know

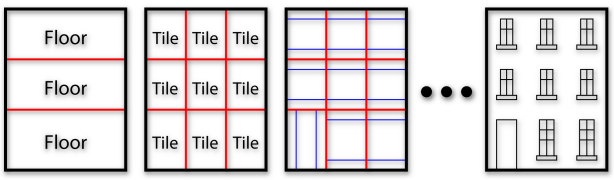

Figure 2: *Our system computes a hierarchical subdivision of facades. This subdivision scheme was successfully employed in the procedural modeling literature by various authors.*

Figure 3: *Left: the facade from Fig. 1 after removing the vertical symmetry. Right: further removing the horizontal symmetry yields the* Irreducible Facade. *Please note that we use the average pixel color for display purposes.*

for each pixel the location of corresponding pixels in symmetric tiles.

Firstly, we detect similar image regions using mutual information (MI). Secondly, based on the extracted repetitions, we create a data structure called *Irreducible Facade* (IF). An IF example is shown in the right image of Fig. 3). Its construction entails the determination of the splitting lines shown in Figs. 1(b) and 2 (only the thicker lines delineating the 'tiles' are meant here). The IF reduces the facade image to its essence, taking out all the repetitions. Although no longer visible, the IF encodes information about these symmetries that govern the floors and tiles (see Fig. 2). The aforementioned splitting lines are found through a global optimisation across all floors and tiles. This not only improves the robustness of the algorithm, but also guarantees that similar elements are split at corresponding positions.

2.2 Subdivision of Facade Tiles

At this stage we want to subdivide the detected tiles into smaller regions. We propose an algorithm which recursively selects the best splitting line in the region under consideration. See Fig. 4 for an example. This structure subdivision is a concept used in procedural modeling and will automatically create a hierarchy of elements. Such a hierarchy will be essential for further analysis, such as the generation of rules for a shape grammar.

Because individual tiles are noisy, the splitting algorithm exploits the knowledge about repetitions which is embedded in the IF. Fig. 5 left illustrates how noise makes the subdivision of individual tiles very unreliable. Therefore, the algorithm analyzes similar structures in other tiles to synchronize the derivation and in so doing, significantly improves the result (see Fig. 5 right).

2.3 Matching 3D Elements

Subdivision of facade tiles leads to a set of rectangular regions clustered into groups of similar regions. At this stage we want to match some of the architectural elements with 3D objects in a library. This is useful for the generation of high-quality geometric information and can provide some semantic interpretation. The solution has to fit the computer graphics modeling pipeline leading to two constraints: We need fast computation times and a general solution working for 3D models in a library.

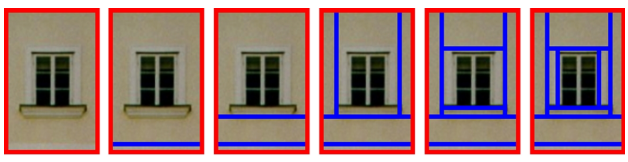

Figure 4: *In the second stage of the process, the tiles are hierarchically subdivided (illustrated as incrementally added lines). Each image represents one step of the subdivision.*

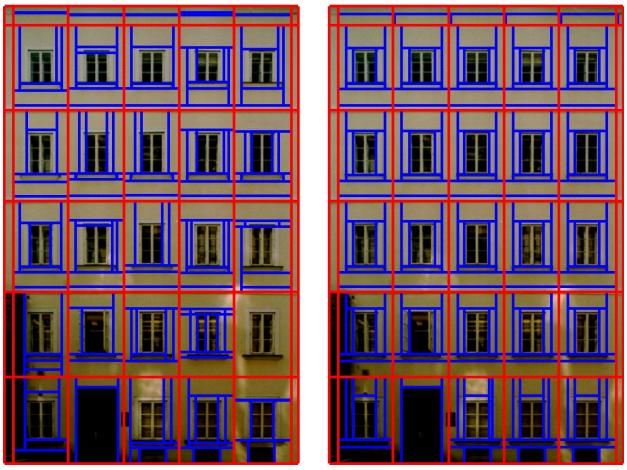

Figure 5: *To make the splitting process more stable, we make use of the previously detected tile repetitions. Left: subdivided tiles based on per-tile local split detection. Right: result if global split synchronization is added.*

2.4 Editing and Rule Extraction

At this stage of the pipeline, the resulting facade interpretation is encoded as a shape tree including fitted templates, but does not contain depth information for the relative positions of the different layers in which the facade and these templates ly. Therefore, simple editing operations are required to set the depth of the facade elements. The user can select clusters of elements and adjust their depth interactively. The added depth information is stored in the shape tree.

In the final step, we can encode the computed subdivision (i.e. the shape tree) as shape grammar rules (Bekins and Aliaga, 2005). The generated rules contain the hierarchical information and correct dimensions. As example, we present the rule set for the facade encoded as *CGA Shape* (Müller et al., 2006) in Fig. 6.

2.5 Discussion

A strength of this method is that it works well even for low resolution facade images, a challenge that has not been tackled previously. Even though the approach is robust in general, there are smaller and larger errors depending on the quality of the input image and input image complexity. Fig. 7 illustrates typical failure cases. The main problems for the fully automatic processing are heavy image noise or small irregular elements (e.g. several irregularly placed air conditioners outside of the window boundaries). In these difficult cases MI might be unable to detect repetitions (see stage 1). Also ground floors of commercial buildings are often problematic for MI due to their non-repetitive structure. As a consequence, vertical symmetries may be left undetected (even if the floors above consist of the same tiles). Another problem is posed by windows with prominent, thick frames. Furthermore, our approach assumes an orthorectified image as input. Strongly

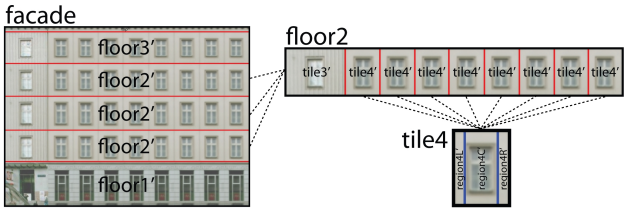

Figure 6: *The extracted shape tree can be automatically converted into a CGA shape grammar rule set.*

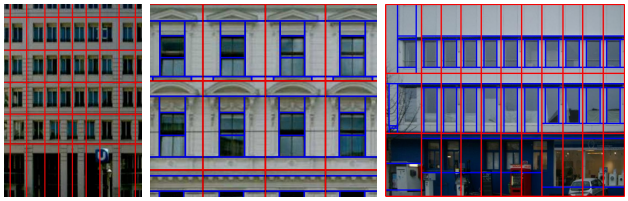

Figure 7: *Failure cases. Left: The facade structure detection cannot handle asymmetric patterns like mezzanines or non-aligned tiles. Middle: Thick window frames are wrongly interpreted as a split and the user has to reverse the split manually. Right: Worst case scenario consisting of a blurry texture with low contrast, a chaotic ground floor disturbing the MI-based repetition detection, and image noise caused by vegetation (left).*

protruding elements, such as balconies, violate this assumption and lead to incorrect tile subdivisions. To summarize, if our approach is applied on less repetitive architectural facades, we lose the structural support and run into the classic difficulties of edge detection i.e. the operations in section 2.2 will be less stable. Hence, we suggest using our technique only in urban areas with buildings of multiple storeys. Also, strong perspective effects complicate rather than help matters, as repetition detection will suffer more from differences due to multiple depth layers. Our second strategy, described next, exploits those very differences to automate the depth layering in stage 4 of the first strategy.

3 STRATEGY 2: STRONG PERSPECTIVE

Similar to the first strategy, the second one uses a single uncalibrated image and exploits the repetitions in typical facade structures. Different from the first strategy, the image here is supposed to show sufficient perspective effects and instead of interactively depth layering structures like windows, this is done automatically. First we summarize the main ideas and contributions behind the strategy. Then, we discuss them in more detail.

Relying on perfectly repeated elements would render the system fragile, especially in the presence of strong perspective effects. Nevertheless, we can hope that traces of repeated elements are found at some feature locations, if the spatial extent of these features is limited. Our method is based on a chain-wise similarity measure to robustly group these feature points. Each group provides evidence for potential repetitions. A group of feature points is also assumed to lie on a plane parallel to the facade to be reconstructed.

A new formulation is proposed to encode the interplay between repetition detection and shape recovery, *i.e.*the former provides clues for the latter, while the latter in turn produces 3D information (occlusion and depth differences) for the former. An energy functional captures the consistency between shape and image, the quality of repetition, and the smoothness. A graph-cut minimization globally optimizes the solutions for both the repetition detection and 3D shape recovery problems.

In contrast to the prior art, we are capable of reconstructing both windows and balconies, and we try to avoid using strong models for them, in order to keep the method sufficiently generic. The goal is also to deal with larger variations in appearance than what has been demonstrated so far.

3.1 Formulation and Overview

Given a single uncalibrated ground-based image $I(\mathbf{x})$ of a building facade, our goal is to reconstruct a three-dimensional shape

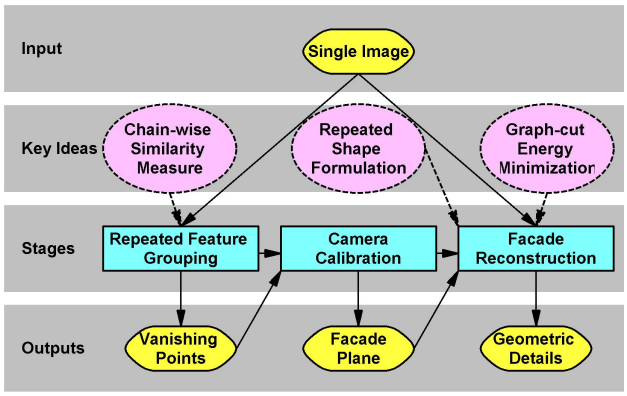

Figure 8: *Overview of the three stages of the proposed algorithm.*

$z(\mathbf{x})$ that is consistent with this input image. We assume that the facade contains multiple elements of the same type (*e.g.*similar windows or balconies) and that their appearances repeat in the horizontal and/or vertical directions along the facade plane. These two assumptions hold for most buildings. The exceptions are beyond the scope of this paper. Please note that we make no assumption on repeated element appearance or frequency.

Let $P = K[R|t]$ be the 3×4 camera matrix, where K and $[R|t]$ are the internal and external parameters. We can choose the world coordinate system such that $[R|t]$ is equal to $[I|0]$, thus we have $P = [K|0]$. Let $\mathbf{p}$ denote a unit vector representing the orientation of the facade plane, then the fact that repeated elements share the same depth layer parallel to the facade can be expressed in terms of $z(\mathbf{x})$, K and $\mathbf{p}$ as the following implicit function:

$$\mathbf{p}^T K^{-1} \left(\begin{pmatrix} \mathbf{x}_l \\ 1 \end{pmatrix} z(\mathbf{x}_l) - \begin{pmatrix} \mathbf{x}_r \\ 1 \end{pmatrix} z(\mathbf{x}_r) \right) = 0, \qquad (1)$$

where the pair $(\mathbf{x}_l, \mathbf{x}_r)$ are two arbitrary corresponding image points of the repeated elements.

In general, Eq. (1) is not easy to solve for those depths, as $\mathbf{p}$, K, $z(\mathbf{x})$ and point correspondences are all unknown. However, the following facts simplify the computation of this equation:

- Repetition is ubiquitous in facades and finding at least part of the repetitions ought to be feasible. We propose a robust matching method to provide several reliable corresponding pairs for Eq. (1);
- Considering the lines that link all pairs in the same group, two main vanishing points can be obtained corresponding to the vertical and horizontal directions of the facade plane.
- K can be determined by orthogonal vanishing points assuming that $K = diag(f, f, 1)$. This is acceptable under conditions specified later.
- $\mathbf{p}$ can be determined by a vanishing line and K;
- Once K and $\mathbf{p}$ are known, Eq. (1) becomes a linear equation in terms of $z(\mathbf{x})$, which can be optimized via a graph-cut minimization technique.

Based on these clues, we divide the whole system into the following three steps: In the first step (Sec. 3.2), repeated feature points are robustly detected and matched in groups; In the second step (Sec. 3.3), these groups are used to ease the computation of $\mathbf{p}$ and

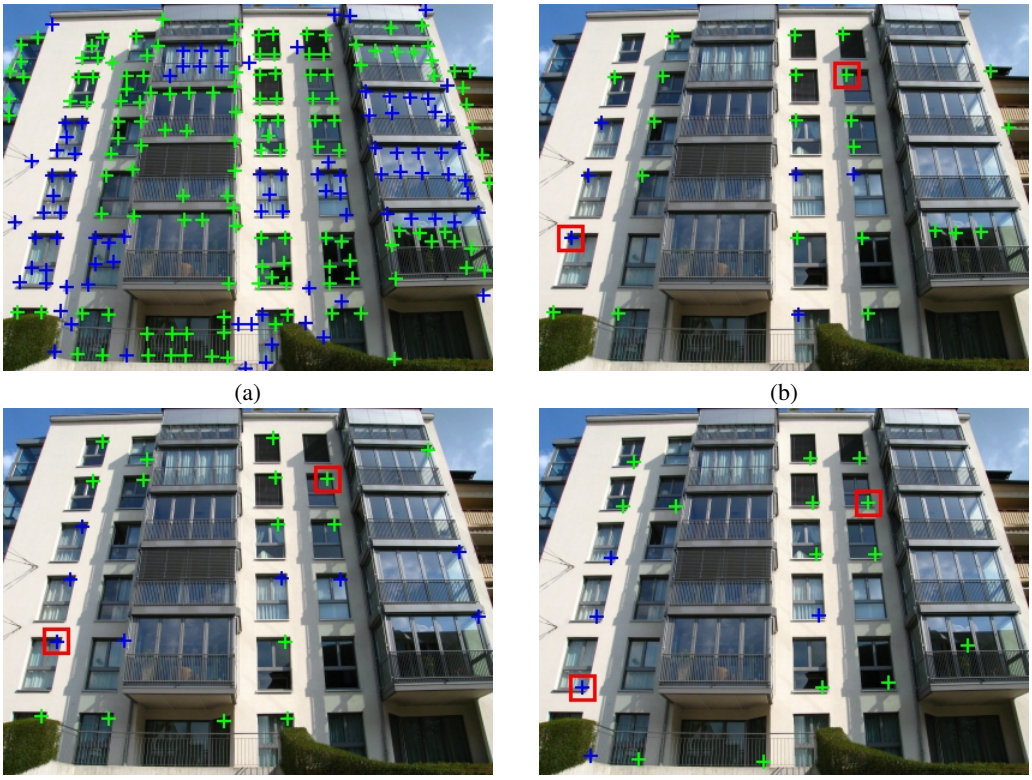

Figure 9: *Groups of feature points: (a) All detected feature points (Crosses with different intensities are used for visibility); (b)-(d) Three biggest groups by the chain-wise similarity measure. Although some feature points (*e.g.*see squares in (b)-(d)) are hard to match due to the different photometric or geometric transformations, they can be linked through additional evidence from intermediate patterns.*

K; In the last step (Sec. 3.4), an energy minimization scheme is designed to optimize both repetition $\{(\mathbf{x}_l, \mathbf{x}_r)\}$ and shape $z(\mathbf{x})$ densely. An overview of the whole system can be found in Fig. 8.

3.2 Repeated Feature Grouping

The first step is to define a feature detector and a robust similarity measure $C(\mathbf{x}_l, \mathbf{x}_r)$. We choose corners and a small square region around them (11×11 in the experiments) as our features. These regions are smaller than the areas matched with mutual information in our first strategy. The reason to keep the regions smaller is to make the matching more robust against the perspective effects dealt with here. Zero-mean normalized cross-correlation (ZNCC) can deal with some intensity changes (e.g. due to shadows), but is more stringent than MI, leading to fewer false matches between small regions. Thus we take $C(\mathbf{x}_l, \mathbf{x}_r) = ZNCC(\mathbf{x}_l, \mathbf{x}_r)$.

However, in order to group feature points into several types, the use of a robust pair-wise measure like ZNCC does not suffice. For effective grouping, a similarity measure is required to be an *equivalence relation* satisfying the following requirements:

- Reflexivity: $\mathbf{x}_i$ is similar to itself;
- Symmetry: If $\mathbf{x}_i$ is similar to $\mathbf{x}_j$, $\mathbf{x}_j$ is similar to $\mathbf{x}_i$;
- Transitivity: If $\mathbf{x}_i$ is similar to $\mathbf{x}_j$, and if $\mathbf{x}_j$ is similar to $\mathbf{x}_k$, $\mathbf{x}_i$ is similar to $\mathbf{x}_k$.

Unfortunately, the third requirement is not guaranteed to hold for the pair-wise similarity $C(\mathbf{x}_l, \mathbf{x}_r)$.

When comparing two feature points $\mathbf{x}_l$ and $\mathbf{x}_r$ of the same type, $C(\mathbf{x}_l, \mathbf{x}_r)$ may fail to achieve a high score due to the different photometric or geometric transformations. But important, additional evidence may come from intermediate patterns that are found, i.e. there exists a chain $\overrightarrow{\mathbf{x}_0\mathbf{x}_1\cdots\mathbf{x}_n}$ ($\mathbf{x}_0 = \mathbf{x}_l$ and $\mathbf{x}_n = \mathbf{x}_r$) in which subsequent elements have high similarity scores, even if the end nodes do not. This should encourage the system to group $\mathbf{x}_l$ and $\mathbf{x}_r$ into the same type.

3.2.1 Chain-wise Similarity The above observations have motivated us to introduce a chain-wise similarity $\hat{C}(\mathbf{x}_l, \mathbf{x}_r)$. The basic idea is to link up two feature points with the most gradually changing chain of elements, such that the pair-wise similarities between adjacent elements are high. This chain-wise similarity is then expressed as:

$$\hat{C}(\mathbf{x}_l, \mathbf{x}_r) = \max_{\overrightarrow{\mathbf{x}_0\cdots\mathbf{x}_n}} \{\min_i \{C(\mathbf{x}_i, \mathbf{x}_{i+1})\}\} \quad \text{with} \quad \mathbf{x}_0 = \mathbf{x}_l, \mathbf{x}_n = \mathbf{x}_r. \tag{2}$$

In order to compute such a similarity measure, we embed the problem into a complete graph with the nodes being the feature points $\mathbf{x}_i$ and the edges among them having $C(\cdot)$ as weights. We then consider the spanning tree (ST), which is a graph containing all the nodes, but having no loops. With the maximal sum of its edge weights, the "maximum spanning tree" (MST) leads to an efficient computation of the chain-wise similarity $\hat{C}(\mathbf{x}_l, \mathbf{x}_r)$ for all pairs of nodes $\mathbf{x}_l$ and $\mathbf{x}_r$, *i.e.* to the path that leads to the maximum chain-wise similarity as just defined. Please note that this maximum spanning tree is similar to the usual minimum spanning tree but aimed at high edge weights.

With the chain-wise similarity $\hat{C}(\mathbf{x}_l, \mathbf{x}_r)$, the transitivity prop-

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

Figure 10: *Detected vanishing points: (a) With pair-wise similarity measure $C(\mathbf{x}_l, \mathbf{x}_r)$ and all feature points in Fig. 9(a), vanishing points are wrongly detected; (b)-(d) With chain-wise similarity measure $\hat{C}(\mathbf{x}_l, \mathbf{x}_r)$ and the three biggest groups in Fig. 9(b)-(d), vanishing points are correctly detected. The pairs consistent with vanishing points are linked with the superimposed lines.*

erty is satisfied as $\hat{C}(\mathbf{x}_l, \mathbf{x}_r) \geq \min\{\hat{C}(\mathbf{x}_l, \mathbf{x}_i), \hat{C}(\mathbf{x}_i, \mathbf{x}_r)\}$, and thus the grouping process of elements into types can be based on it. Given a threshold τ, if $\hat{C}(\mathbf{x}_l, \mathbf{x}_r) \geq \tau$, $\mathbf{x}_l$ and $\mathbf{x}_r$ are supposed to be of the same type, otherwise of different types. In the graph, this is equivalent to breaking certain branches of the MST, resulting in a subtree per element type.

Fig. 9 shows an example with its three biggest groups as detected by the chain-wise similarity measure. In general τ can be chosen more conservatively than the threshold used with a pair-wise measure. In all the experiments, we set $\tau = 0.9$.

3.3 Camera Calibration

Given a set of feature groups, the next step is to compute K and $\mathbf{p}$. Observing Eq. (1), it is interesting to examine the pair set,

$$S_{\mathbf{t}} = \{(\mathbf{x}_l, \mathbf{x}_r) : \begin{pmatrix} \mathbf{x}_l \\ 1 \end{pmatrix} z(\mathbf{x}_l) - \begin{pmatrix} \mathbf{x}_r \\ 1 \end{pmatrix} z(\mathbf{x}_r) = kK\mathbf{t}\}, \tag{3}$$

where k is a scaling factor and varies for different pairs and where, in the world coordinate system we use $S_{\mathbf{t}}$ to describe a set of vector pairs $(K^{-1} \begin{pmatrix} \mathbf{x}_l \\ 1 \end{pmatrix} z(\mathbf{x}_l), K^{-1} \begin{pmatrix} \mathbf{x}_r \\ 1 \end{pmatrix} z(\mathbf{x}_r))$ that share a common direction $\mathbf{t}$ for their difference. Once $S_{\mathbf{t}}$ is obtained, Eq. (1) can be converted into the form $\mathbf{p}^T\mathbf{t} = 0$. Two such equations are sufficient to solve $\mathbf{p}$.

On the other hand, in the image plane $S_{\mathbf{t}}$ manifests itself as a set of pairs $(\mathbf{x}_l, \mathbf{x}_r)$ that share a common vanishing point $K\mathbf{t}$. Vanishing points in turn can be used to solve for K: Two vanishing points $\mathbf{v}_1$ and $\mathbf{v}_2$ with perpendicular directions satisfy

$$\mathbf{v}_1^T \omega \mathbf{v}_2 = 0, \tag{4}$$

where $\omega = (KK^T)^{-1}$ is the absolute conic in the image.

In order to obtain vanishing points with perpendicular directions, we consider a group of feature points detected in Sec. 3.2. The feature points are corresponding points on the repeated elements. Searching these is based on our assumption facade structures repeat in the horizontal and vertical directions along the facade plane. This however, also implies repetitions in diagonal directions, which are of no further import to our analysis. For the detection of the horizontal and vertical vanishing points, we prefer using the repetition groups with the highest number of matched features. Within such group, we look for the two vanishing points supported by largest number of feature pairs. Fig. 10 shows an example of vanishing point detection with (b-d) and without (a) grouping information. The chain-wise similarity measure links the feature points in groups even if their appearances differ, and it produces more consistent pairs than the pair-wise ZNCC measure does. Hence, vanishing point detections with groups are more reliable than those without groups. Fig. 10(b-d) show the dominant vanishing points for the corresponding feature groups in Fig. 9(b-d).

In general, three couples of vanishing points with perpendicular directions are sufficient to solve K with the assumptions of no skew and square pixels. However, for a building facade image, the vanishing point for the third, perpendicular direction – the depth direction of the building – is often very difficult to extract, if possible at all. As a result, we simplify the internal camera model further to $K = diag(f, f, 1)$. This model assumes square pixels, as before, but adds the assumption that the principal point is known. Based on Eq. (4), the only unknown parameter f can then be solved.

Once K is obtained, $\mathbf{p}$ can be easily computed by the vanishing line that connects the two vanishing points, $\mathbf{p}^T = (\mathbf{v}_1 \times \mathbf{v}_2)^T K$, where $\times$ represents the vector or cross product.

3.4 Facade Reconstruction

Given K and $\mathbf{p}$, Eq. (1) becomes a linear equation in terms of $z(\mathbf{x})$. The last step is to design an energy minimization scheme to optimize $z(\mathbf{x})$. In order to achieve the goal, we first define a consistency measure to describe "how good $z(\mathbf{x})$ is".

We consider the group with the most feature points. Based on our assumption, the points in this group lie on a plane parallel to the facade. To fix the scale, we let $(\mathbf{p}^T, 1)$ denote this plane. Note that the resulting 3D construction will therefore come at a certain scale, which probably is not the correct one. Our final result is only defined up to an unknown scale. Thus, for each feature point $\mathbf{x}_i$ in this group, its depth value Z_i can be estimated. Please note that these $\{Z_i\}$ will not act as hard constraints when optimizing $z(\mathbf{x})$ (*i.e.*$Z_i = z(\mathbf{x}_i)$ does not always hold). $\{Z_i\}$ are only used to estimate the 3D *transformation vectors* introduced in the following paragraphs.

Suppose we are given a pair of corresponding feature points $(\mathbf{x}_l, \mathbf{x}_r)$. Corresponding points close to these two corresponding feature points can be joined by identical 3D transformation vectors:

$$\begin{aligned} \begin{pmatrix} \mathbf{x}_L \\ 1 \end{pmatrix} z(\mathbf{x}_L) \quad &- \quad \begin{pmatrix} \mathbf{x}_R \\ 1 \end{pmatrix} z(\mathbf{x}_R) \\ \equiv \begin{pmatrix} \mathbf{x}_l \\ 1 \end{pmatrix} Z_l \quad &- \quad \begin{pmatrix} \mathbf{x}_r \\ 1 \end{pmatrix} Z_r. \end{aligned} \tag{5}$$

Considering the inverse problem, the corresponding point $\mathbf{c}_{l,r}(\mathbf{x})$ of $\mathbf{x}$ and its depth value $z(\mathbf{c}_{l,r}(\mathbf{x}))$ can be determined from this 3D transformation vector, *i.e.*

$$\begin{aligned} \mathbf{c}_{l,r}(\mathbf{x}) &= \frac{\mathbf{x}z(\mathbf{x}) - \mathbf{x}_l Z_l + \mathbf{x}_r Z_r}{z(\mathbf{x}) - Z_l + Z_r}, \\ z(\mathbf{c}_{l,r}(\mathbf{x})) &= z(\mathbf{x}) - Z_l + Z_r. \end{aligned} \tag{6}$$

Based on these equations, we define two measures to describe the consistency with the input image and the quality of repetition, respectively:

$$\begin{aligned} e_{image,l,r}(\mathbf{x}) &= 1 - |C(\mathbf{x}, \mathbf{c}_{l,r}(\mathbf{x}))|, \\ e_{repeat,l,r}(\mathbf{x}) &= |z(\mathbf{x}) - z(\mathbf{c}_{l,r}(\mathbf{x})) - Z_l + Z_r|. \end{aligned} \tag{7}$$

3.4.1 Energy Minimization We minimize an energy functional of the form:

$$E_{total}(z) = E_{image}(z) + \beta E_{repeat}(z) + \gamma E_{smooth}(z). \tag{8}$$

The first term enforces the consistency between the observed image and the synthesized shape

$$E_{image}(z) = \sum_{\mathbf{x}} \min_{l,r} e_{image,l,r}(\mathbf{x}), \tag{9}$$

where $\min_{l,r}$ takes the minimum value from all potential matching points, i.e. all points found at a displacement corresponding with one of the 3D transformations coming out of repetition detection.

The second term assesses the quality of the repetition

$$E_{repeat}(z) = \sum_{\mathbf{x}} \sum_{l,r} |C(\mathbf{x}, \mathbf{c}_{l,r}(\mathbf{x}))| e_{repeat,l,r}(\mathbf{x}), \tag{10}$$

where $|C(\mathbf{x}, \mathbf{c}_{l,r}(\mathbf{x}))|$ gives more weight when the repetition quality is high.

The third term imposes smoothness. Since $\mathbf{p}$ is known, an intuitive idea is to measure the variation along $\mathbf{p}$. We define $z_{\mathbf{p}}(\mathbf{x})$ as

$$z_{\mathbf{p}}(\mathbf{x}) = \mathbf{p}^T K^{-1} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} z(\mathbf{x}), \tag{11}$$

which is the distance to the plane $(\mathbf{p}^T, 0)$ in the world coordinate system. Thus the third smoothness term can be defined as

$$E_{smooth}(z) = \sum_{\mathbf{x}} ||\nabla z_{\mathbf{p}}(\mathbf{x})|| \cdot (1 - ||\nabla \bar{I}(\mathbf{x})||), \tag{12}$$

where $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ is the gradient operator and $\bar{\nabla}$ represents normalised gradient magnitude, i.e. the maximum gradient value in the image is put to one and the other values are scaled accordingly. The effect of multiplying with $1 - ||\nabla I(\mathbf{x})||$ is to make the smoothing edge-preserving. Smoothing is stronger in homogeneous regions than near intensity boundaries.

3.4.2 Graph Cuts The success of graph-cut optimizations in similar domains has motivated us to embed our energy minimization problem (in Eq. (8)) into a graph, and use the classic max-flow/min-cut algorithm to obtain the optimal solution. Kolmogorov and Zabih (Kolmogorov and Zabih, 2004) give a characterization of what energy functions can be minimized using graph-cuts, and they also provide a graph-construction method. Readers are referred to their paper for more detailed information.

In the following paragraphs, we follow their approach and focus on the proof that validates our energy minimization problem, *i.e.* we convert the energy functional in Eq. (8) into a binary form which is graph-representable, *i.e.* each term $E^{i,j}$ satisfies the following condition

$$E^{i,j}(0,0) + E^{i,j}(1,1) \leq E^{i,j}(0,1) + E^{i,j}(1,0). \tag{13}$$

$\alpha-$expansion Although $z(\mathbf{x})$ is a continuous function and cannot be represented by binary variables, we can convert it for the $\alpha-$expansion operation: Any configuration $z_\alpha(\mathbf{x})$ within a single $\alpha-$expansion of the initial configuration $z(\mathbf{x})$ can be encoded by a binary function

$$\triangle z(\mathbf{x}) = \begin{cases} 0, & \text{if } z_\alpha(\mathbf{x}) = z(\mathbf{x}); \\ 1, & \text{if } z_\alpha(\mathbf{x}) = z(\mathbf{x}) \frac{\alpha}{z_{\mathbf{p}}(\mathbf{x})}. \end{cases} \tag{14}$$

Given Eq. (11) the α label defines a plane

$$\mathbf{p}^T K^{-1} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} z_\alpha(\mathbf{x}) = \alpha \tag{15}$$

with orientation $\mathbf{p}$ and distance to the origin α. Let $z_\triangle(\mathbf{x})$ denote a configuration defined by $\triangle z(\mathbf{x})$. Then, we have the energy of binary variables,

$$\begin{aligned} \triangle E_{total}(\triangle z) &= \triangle E_{image}(\triangle z) + \alpha \triangle E_{repeat}(\triangle z) \\ &+ \beta \triangle E_{smooth}(\triangle z), \end{aligned} \tag{16}$$

where

$$\begin{aligned} \triangle E_{image}(\triangle z) &= E_{image}(z_\triangle) \\ \triangle E_{repeat}(\triangle z) &= E_{repeat}(z_\triangle) \\ \triangle E_{smooth}(\triangle z) &= E_{smooth}(z_\triangle). \end{aligned} \tag{17}$$

The first term $\triangle E_{image}(\triangle z)$ depends on only one variable, and

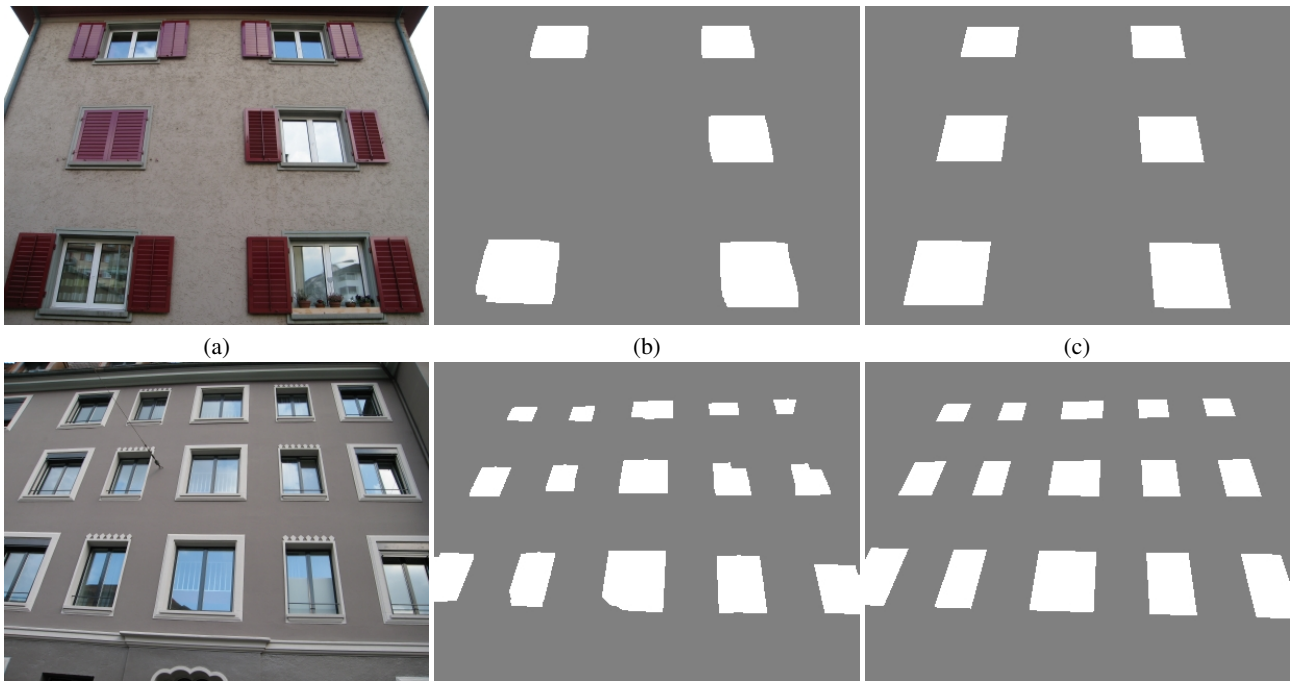

Figure 11: *Experimental result: (a) Input image; (b) Optimal shape by the graph-cut minimization; (c) Final shape after the refinement step.*

thus it is graph-representable.

For the second term $\triangle E_{repeat}(\triangle z)$, let's consider a single term $|C(\mathbf{x}, \mathbf{c}_{l,r}(\mathbf{x}))|e_{repeat,l,r}(\mathbf{x})$ in equation (10). Based on the fact that $e_{repeat,l,r}(\mathbf{x}) \equiv 0$ when (x) and $\mathbf{c}_{l,r}(\mathbf{x})$ have same label, we have $\triangle E^{\mathbf{x},\mathbf{c}(\mathbf{x})}_{repeat}(1,1) = 0$, and it can be proven that $\triangle E^{\mathbf{x},\mathbf{c}(\mathbf{x})}_{repeat}(0,0) \leq \triangle E^{\mathbf{x},\mathbf{c}(\mathbf{x})}_{repeat}(0,1) + \triangle E^{\mathbf{x},\mathbf{c}(\mathbf{x})}_{repeat}(1,0)$. Therefore, condition (13) holds.

For the third term $\triangle E_{smooth}(\triangle z)$, we have $\triangle E^{\mathbf{x},\mathbf{c}(\mathbf{x})}_{smooth}(1,1) = 0$ and $\triangle E^{\mathbf{x},\mathbf{c}(\mathbf{x})}_{smooth}(0,0) \leq \triangle E^{\mathbf{x},\mathbf{c}(\mathbf{x})}_{smooth}(0,1) + \triangle E^{\mathbf{x},\mathbf{c}(\mathbf{x})}_{smooth}(1,0)$. It is also graph-representable.

3.4.3 Shape Prior Windows and balconies often have rectangular shapes. It is not straightforward to directly add such prior constraints into the graph-cut minimization, since we have no information on the element locations before the minimization process starts. Therefore, we enforce the shape prior in a second refinement step. The element locations can then be based on the optimal shape produced with graph-cuts. The goal is also to align vertical and horizontal boundaries. Moreover, since the facade orientation $\mathbf{p}$ is known, we can add connecting planar patches orthogonal to the facade at steep transitions between different depths.

In practice, this refinement can be easily done by first summing up $z_{\mathbf{p}}(\mathbf{x})$ (in Eq. (11)) along the vertical and horizontal directions and then taking the positions of maximal variation of the sum as the element boundary positions. Figs. 12(b)-(c)&13(b)-(c) compare the element shapes before and after the refinement. Please note the connection between the balconies (in black) and wall (in gray).

3.5 Results

Implementation Details Feature points have been selected with the Harris corner detector. The threshold, τ, for classifying the feature points in Sec. 3.2 was fixed to 0.9. The β and γ parameters in Eq. (8) are set to 0.125 and 0.25, resp. The other parameters are all determined automatically by the system.

Experimental Results We show four experimental results to demonstrate the quality of the repetition detection and facade reconstruction. The first two are shown in Fig. 11. They contain windows with reflections. The first example has both open and closed windows of the same type, while the second example has two window types with different width as shown in Fig. 11(a). Such variations make the repetition detection harder. By combining repetition detection and shape recovery into the same framework and performing joint optimization via the graph-cut minimization, the proposed algorithm robustly detects the window regions by repeated feature points and depth differences as shown in Fig. 11(b). In the first example a window is missing due to the lack of depth difference. Although the window blind is a clue for a human, it can be regarded as wall texture and thus is hard to detect. The second example demonstrates the ability of our algorithm to handle different ratios between width and height. Please note the two partially open windows. By adding the prior knowledge of element shapes and layout, the boundaries of the final results are more accurate as shown in Fig. 11(c). The running time of the whole process for these two experiments are about 300 seconds on a Pentium4 3.2GHz machine.

Fig. 12 shows a third experimental result. Again, the windows vary greatly in their appearance and their detection is far from trivial. Moreover, balconies present another kind of building elements and often occlude other elements (*e.g.*windows or doors). Fig. 12(b) shows the optimal depth by the graph-cut minimization. Almost all of the windows are detected, except the top-left and top-right ones, due to the strong occlusions in both cases. There are some noticeable errors on window frames, *i.e.*the window frames are sometimes wrongly detected due to their thin shapes. Fig. 12(c) shows the final depth by adding shape priors. Please note the correction of the two missing windows and the added connections between the balcony fronts and the wall.

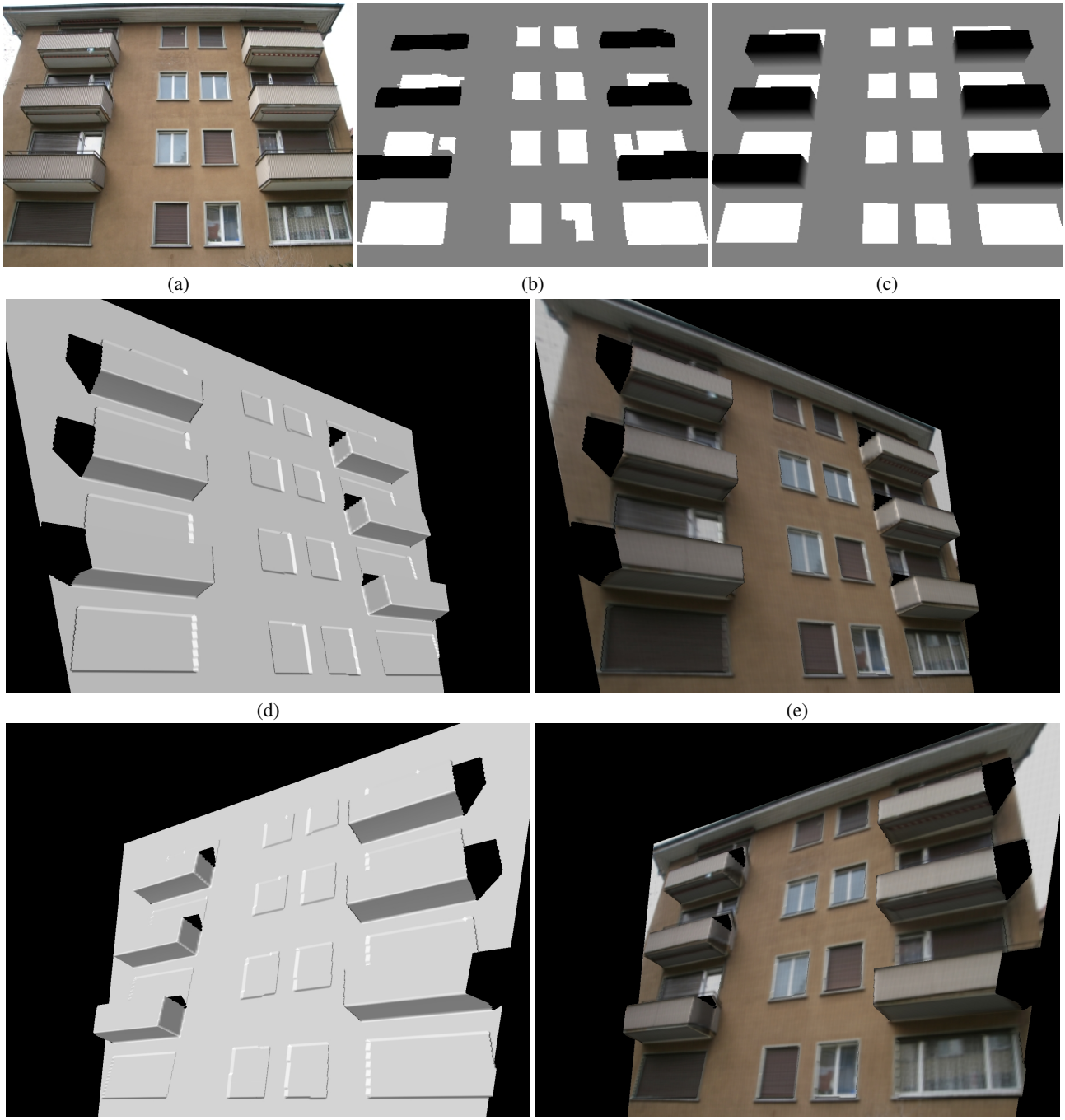

Figure 12: *Reconstruction result: (a) Input image; (b) Optimal shape $z_{\mathbf{p}}(\mathbf{x})$ obtained by the graph-cut minimization; (c) Final shape after the refinement with rectangular shape priors; (d) 3D surfaces in new viewpoints; (e) 3D surfaces with texture in the same viewpoints.*

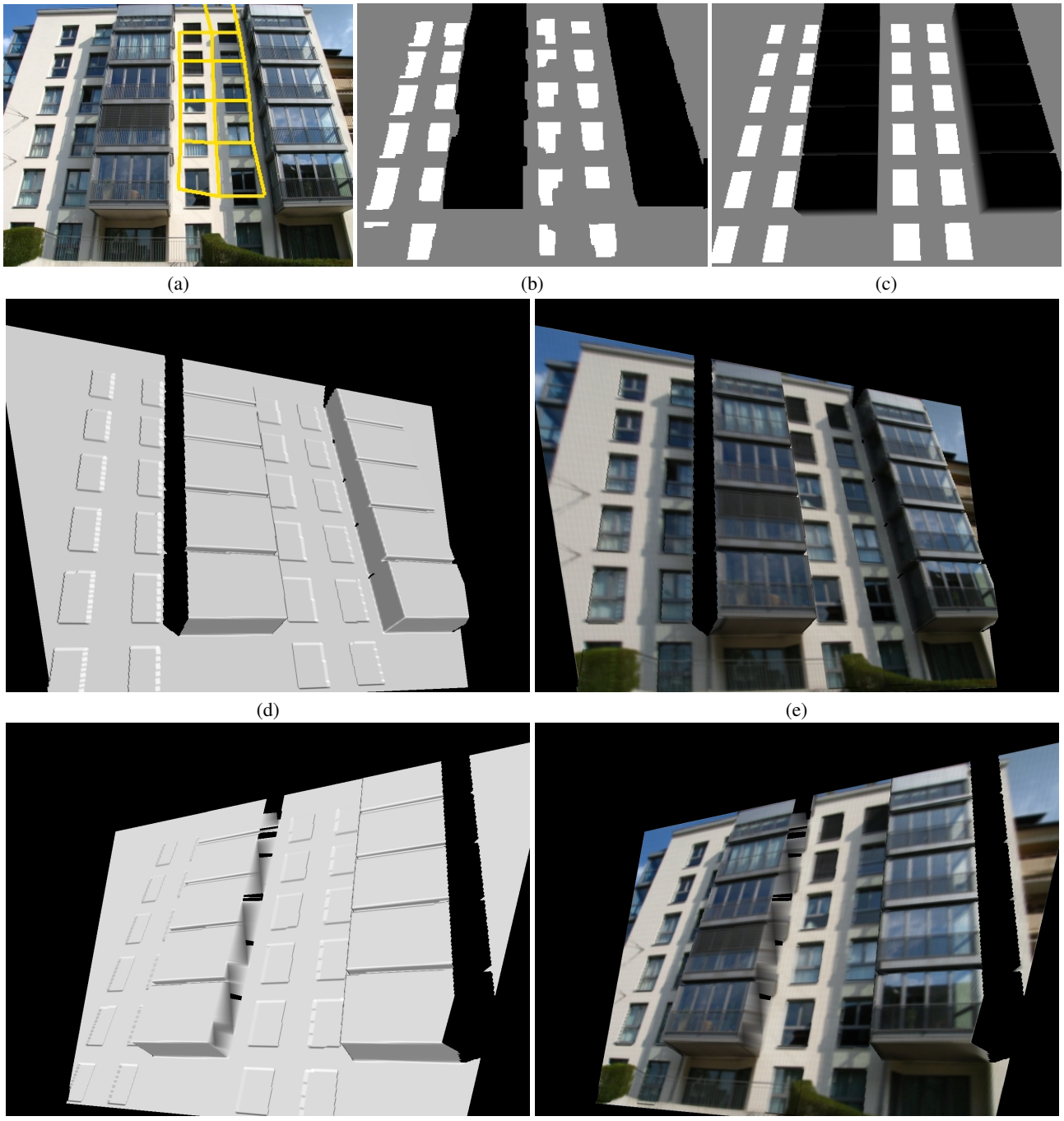

Figure 13: *Reconstruction result: (a) Input image; (b) Optimal shape $z_{\mathbf{p}}(\mathbf{x})$ obtained by the graph-cut minimization; (c) Final shape after the refinement with rectangular shape priors; (d) 3D surface in a new viewpoint; (e) 3D surface with texture in the same viewpoint; As comparison, the yellow lines in (a) shows the results of the state of the art repetition detection by Hays* et al. *(Hays et al., 2006).*

Fig. 12(d)&(e) show the 3D shape without and with texture from two new viewpoints.

Fig. 13 shows a fourth experimental result with many windows. The balconies are vertically connected. It contains several lighting effects, such as shadows, highlights, transparency and so on. Plants, window curtains, and blinds let the windows appear quite different from each other. Fig. 13(b) shows the optimal depth by the graph-cut minimization. There are some small errors on windows although the main parts are robustly detected. Fig. 13(c) shows the final depth by adding shape priors. The errors are all corrected, and occluded windows are detected. Fig. 13(d)&(e) shows the 3D shape without and with texture from a new viewpoint. The running times of the whole process for the two latter experiments are about 700 seconds.

Finally, we show a comparison with the state of the art in repetition detection. One of the detected groups of Hays *et al*. (Hays et al., 2006) is shown in Fig. 13(a). Their approach exploits the regular distribution of repeated elements (or texture). It is a generic method and has been designed for general purposes. The detected result contains almost all windows on the right hand side, but no connection is established with the similar windows on the left, as elements are supposed to be contiguous. Also, since neither 3D information nor shape knowledge is encoded, a meaningful element is often separated into parts, which belong to different repeated patterns. Compared with this work, our method is designed for the special purpose of repetition detection of building elements. Repetition detection provides information for shape recovery, while the latter in turn produces additional information for the former. Our detected results in Fig. 13(b)&(c) contain both sides of windows although they are not connected. The boundaries and depths are determined, and occlusions are also handled.

3.5.1 Discussion The proposed approach has been tested with various images of different qualities and conditions. We summarize the issues raised from these experiments as follows: Firstly, in order to speed up the whole process, we need to resize the image into 640×480 for all our experiments. A high resolution image requires too long a time for optimization. On the other hand, too small a resolution cannot provide sufficient information to distinguish different depth layers.

Secondly, readers may have noticed some errors in feature point grouping in Fig. 9. The situation is a bit like with RANSAC. Too many outliers or too many missing inliers - difficult to quantify in general terms - may cause failure to recover from such flaws.

Finally, the images should be taken with short focal lengths, so that there are strong perspective effects, conveying good depth information. This said, keeping a complete building in the field of view often imposes such choice.

4 CONCLUSION

In this paper, we have proposed two strategies for the efficient 3D modeling of facades. Both relied on prior knowledge of architectural structures and used a single image as input. The first could deal with single images, even if they show little perspective. But the price to pay is a need for limited interaction. In the second strategy, the full process is automatic, but it requires sufficiently strong perspective effects to be present in the image.
The automatic selection between the strategies still remains to be implemented. In the future, we would also like to extend the strategy further, and deal with multiple views in case they are available. Again, such system would then have to automatically adapt its strategy to the nature of the input.
For the moment, we have mainly considered simple repeat rules of the CGA Grammar. In the future we hope to extract more sophisticated rules from the imagery, and to describe the result of the image analysis as a set of CGA grammatical rules. This will lead to very compact building representations.

REFERENCES

Alegre, F. and Dellaert, F., 2004. A probabilistic approach to the semantic interpretation of building facades. In: International Workshop on Vision Techniques Applied to the Rehabilitation of City Centres.

Bekins, D. R. and Aliaga, D. G., 2005. Build-by-number: Rearranging the real world to visualize novel architectural spaces. In: IEEE Visualization, IEEE Computer Society, p. 19.

Blanz, V. and Vetter, T., 1999. A morphable model for the synthesis of 3d faces. In: SIGGRAPH, pp. 187–194.

Brenner, C. and Ripperda, N., 2006. Extraction of facades using rjMCMC and constraint equations. In: Photogrammetric Computer Vision, pp. 155–160.

Debevec, P. E., Taylor, C. J. and Malik, J., 1996. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In: SIGGRAPH, pp. 11–20.

Dick, A. R., Torr, P. H. S., Ruffle, S. J. and Cipolla, R., 2001. Combining single view recognition and multiple view stereo for architectural scenes. In: ICCV, pp. 268–274.

Dick, A. R., Torr, P. H. S. and Cipolla, R., 2004. Modelling and interpretation of architecture from several images. International Journal of Computer Vision 60(2), pp. 111–134.

Duarte, J., 2002. Malagueira Grammar – towards a tool for customizing Alvaro Siza's mass houses at Malagueira. PhD thesis, MIT School of Architecture and Planning.

Flemming, U., 1987. More than the sum of its parts: the grammar of queen anne houses. Environment and Planning B 14, pp. 323–350.

Früh, C. and Zakhor, A., 2001. 3d model generation for cities using aerial photographs and ground level laser scans. In: CVPR (2), IEEE Computer Society, pp. 31–38.

Früh, C. and Zakhor, A., 2003. Constructing 3d city models by merging aerial and ground views. IEEE Computer Graphics and Applications 23(6), pp. 52–61.

Havemann, S., 2005. Generative Mesh Modeling. PhD thesis, TU Braunschweig.

Hays, J., Leordeanu, M., Efros, A. A. and Liu, Y., 2006. Discovering texture regularity as a higher-order correspondence problem. In: A. Leonardis, H. Bischof and A. Pinz (eds), ECCV (2), Lecture Notes in Computer Science, Vol. 3952, Springer, pp. 522–535.

Hu, J., You, S. and Neumann, U., 2006. Integrating lidar, aerial image and ground images for complete urban building modeling. In: 3DPVT, IEEE Computer Society, pp. 184–191.

Jepson, W., Ligget, R. and Friedman, S., 1996. Virtual modeling of urban environments. Presence 5(1), pp. 72–86.

Karner, K., Bauer, J., Klaus, A., Leberl, F. and Grabner, M., 2001. Virtual habitat: Models of the urban outdoors. In: Third International Workshop on Automatic Extraction of Man-Made Objects from Aerial and Space Imaging, Ascona, pp. 393–402.

Kolmogorov, V. and Zabih, R., 2004. What energy functions can be minimized via graph cuts? IEEE Trans. Pattern Anal. Mach. Intell. 26(2), pp. 147–159.

Koning, H. and Eizenberg, J., 1981. The language of the prairie: Frank lloyd wrights prairie houses. Environment and Planning B 8, pp. 295–323.

Lee, S. C. and Nevatia, R., 2003. Interactive 3d building modeling using a hierarchical representation. In: HLK, IEEE Computer Society, pp. 58–65.

Lee, S. C. and Nevatia, R., 2004. Extraction and integration of window in a 3d building model from ground view image. In: CVPR (2), pp. 113–120.

Lee, S. C., Jung, S. K. and Nevatia, R., 2002. Automatic integration of facade textures into 3d building models with a projective geometry based line clustering. Comput. Graph. Forum.

Legakis, J., Dorsey, J. and Gortler, S. J., 2001. Feature-based cellular texturing for architectural models. In: SIGGRAPH, pp. 309–316.

Liebowitz, D. and Zisserman, A., 1998. Metric rectification for perspective images of planes. In: CVPR, IEEE Computer Society, pp. 482–488.

Marvie, J.-E., Perret, J. and Bouatouch, K., 2005. The fl-system: a functional l-system for procedural geometric modeling. The Visual Computer 21(5), pp. 329–339.

Mayer, H. and Reznik, S., 2003. Building facade interpretation from image sequences. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 36(3/W24), pp. 55–60.

Mech, R. and Prusinkiewicz, P., 1996. Visual models of plants interacting with their environment. In: SIGGRAPH, pp. 397–410.

Müller, P., Wonka, P., Haegler, S., Ulmer, A. and Gool, L. J. V., 2006. Procedural modeling of buildings. ACM Trans. Graph. 25(3), pp. 614–623.

Müller, P., Zeng, G., Wonka, P. and Gool, L. V., 2007. Image-based procedural modeling of facades. Proceedings of ACM SIGGRAPH 2007 / ACM Transactions on Graphics.

Parish, Y. I. H. and Müller, P., 2001. Procedural modeling of cities. In: SIGGRAPH, pp. 301–308.

Prusinkiewicz, P. and Lindenmayer, A., 1991. The Algorithmic Beauty of Plants. Springer Verlag.

Prusinkiewicz, P., James, M. and Mech, R., 1994. Synthetic topiary. In: SIGGRAPH, ACM, pp. 351–358.

Prusinkiewicz, P., Mündermann, L., Karwowski, R. and Lane, B., 2001. The use of positional information in the modeling of plants. In: SIGGRAPH, pp. 289–300.

REALVIZ, 2007. Realviz ImageModeler V4.0 product information. http://www.realviz.com.

Ribarsky, W., Wasilewski, T. and Faust, N., 2002. From urban terrain models to visible cities. IEEE Computer Graphics and Applications 22(4), pp. 10–15.

Schindler, K. and Bauer, J., 2003. A model-based method for building reconstruction. In: HLK, IEEE Computer Society, pp. 74–82.

Stiny, G., 1975. Pictorial and Formal Aspects of Shape and Shape Grammars. Birkhauser Verlag, Basel.

Stiny, G., 1982. Spatial relations and grammars. Environment and Planning B 9, pp. 313–314.

Stiny, G. and Mitchell, W. J., 1978. The palladian grammar. Environment and Planning B 5, pp. 5–18.

Takase, Y., Sho, N., Sone, A. and Shimiya, K., 2003. Automatic generation of 3d city models and related applications. In: International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, pp. 113–120.

Wang, L., You, S. and Neumann, U., 2006. Large-scale urban modeling by combining ground level panoramic and aerial imagery. In: 3DPVT, IEEE Computer Society, pp. 806–813.

Wang, X., Totaro, S., Taillandier, F., Hanson, A. and Teller, S., 2002. Recovering facade texture and microstructure from real-world images. In: Proc. ISPRS Commission III Symposium on Photogrammetric Computer Vision, pp. 381–386.

Wonka, P., Wimmer, M., Sillion, F. X. and Ribarsky, W., 2003. Instant architecture. ACM Trans. Graph. 22(3), pp. 669–677.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge support by the European Commission, through IST project Cyberwalk, and the Flemish Fund for Scientific Research, FWO. The authors also thank Peter Wonka, Simon Haegler, and Andreas Ulmer for their contributions to the weak perspective case.