SmartBoxes for Interactive Urban Reconstruction

Liangliang Nan[1] Andrei Sharf[1] Hao Zhang[2] Daniel Cohen-Or[3] Baoquan Chen[1]

[1] Shenzhen Institutes of Advanced Technology (SIAT), China [2] Simon Fraser Univ. [3] Tel Aviv Univ.

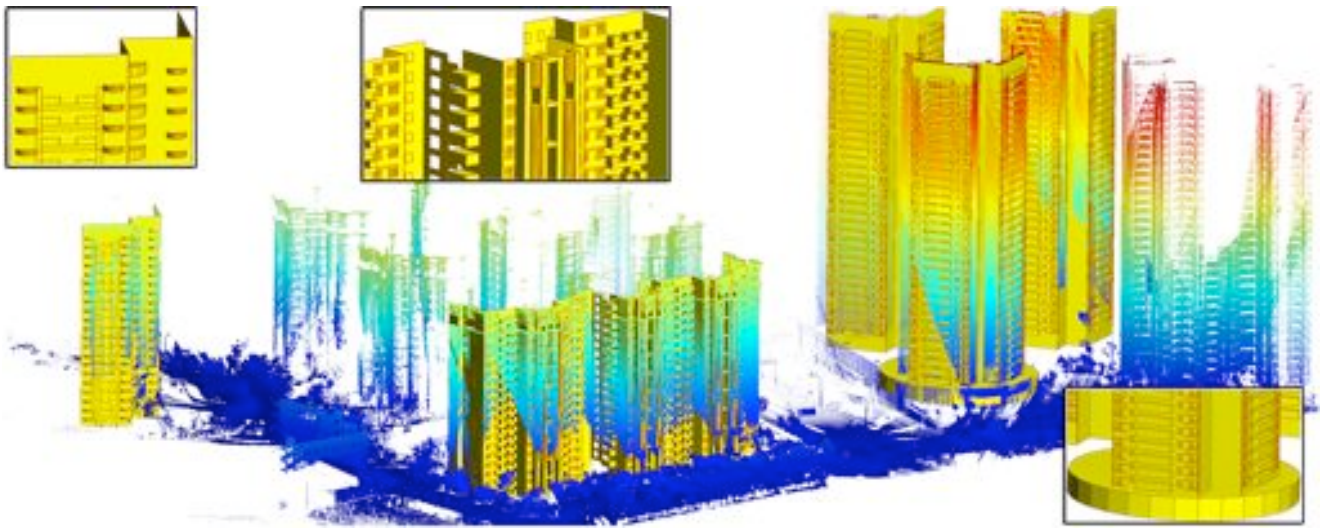

Figure 1: *A large-scale urban architectural scene is reconstructed in details from a noisy and sparse LiDAR scan using SmartBoxes interactively. Close-up views show the reconstructed details of 3D architectural structures.*

Abstract

We introduce an interactive tool which enables a user to quickly assemble an architectural model directly over a 3D point cloud acquired from large-scale scanning of an urban scene. The user loosely defines and manipulates simple building blocks, which we call SmartBoxes, over the point samples. These boxes quickly snap to their proper locations to conform to common architectural structures. The key idea is that the building blocks are smart in the sense that their locations and sizes are automatically adjusted on-the-fly to fit well to the point data, while at the same time respecting contextual relations with nearby similar blocks. SmartBoxes are assembled through a discrete optimization to balance between two snapping forces defined respectively by a data-fitting term and a contextual term, which together assist the user in reconstructing the architectural model from a sparse and noisy point cloud. We show that a combination of the user's interactive guidance and high-level knowledge about the semantics of the underlying model, together with the snapping forces, allows the reconstruction of structures which are partially or even completely missing from the input.

1 Introduction

In recent years, there has been an increasing interest in the modeling and reconstruction of digital urban scenes. Rapid advances in laser scanning technology and the recent proliferation of GIS services such as those offered by Microsoft Virtual Earth or Google Earth have been driving a strong trend towards 3D reconstruction of urban architectural models based on satellite and aerial photography combined with geometry capture enabled by street-level laser scanners. The state of the art on automatic reconstruction from such data allows modeling of the geometry of building layout and ground polygon extrusion with roofs, where the building facades are merely approximated with a small number of textured planes [Zebedin et al. 2008; Xiao et al. 2009]. Reconstructing detailed building structures including facades has remained a challenge.

The main difficulty with laser scans of large-scale urban environments is data quality (or lack thereof). Compared to photographs, acquired geometric data are already of much lower resolution. Large distances between the scanner and scanned objects also imply much reduced precision or higher level of noise. Furthermore, unlike the scanning of relatively small artifacts, such as those in the Michelangelo project [Levoy et al. 2000], where a scanner can be strategically positioned to achieve the necessary point density and coverage, similar controls are rather limited during large-scale urban scanning. As a result, the obtained point clouds typically exhibit significant missing data due to occlusion, as well as uneven point density. A combination of these issues often renders results from fully automatic and fully data-dependent reconstruction schemes less than satisfactory. In the not unlikely case where large portions of the data are completely missing, these schemes simply cannot succeed and no generic templates can be generally reliable. Therefore, some level of user intervention or the incorporation of domain knowledge becomes necessary.

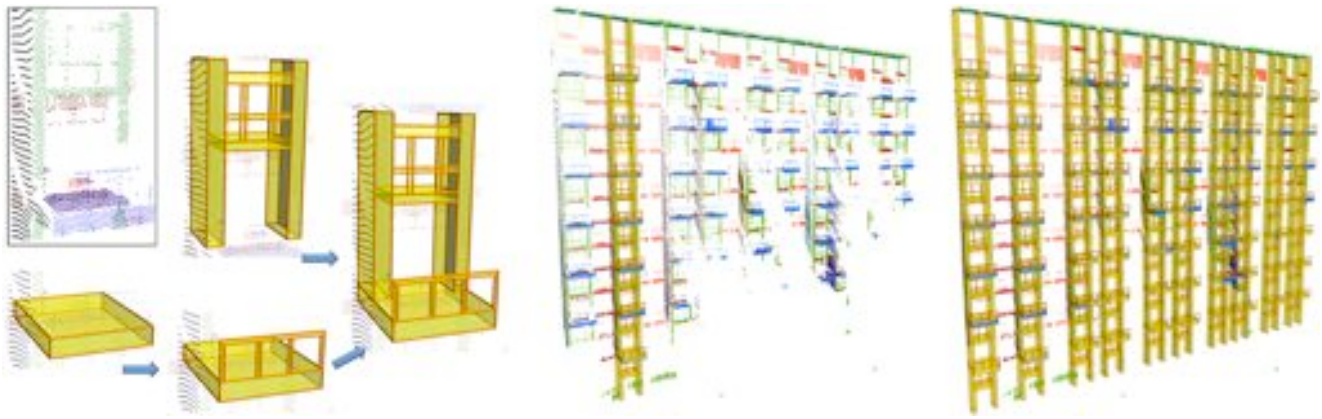

Figure 2: *From a typical poor scan, the user interactively reconstructs simple SmartBoxes to form window railings and balconies (left) through snapping and grouping. Next, one column (middle) is reconstructed through a drag-and-drop of the grouped compound balcony and window. Reconstruction of the whole facade (right) can then be achieved in a matter of seconds by dragging the whole column, grouped into a compound SmartBox, and snapping to sparse regions. Note the irregularities, e.g., in the scales of the columns and their spacing.*

In this paper, we introduce a new *interactive* tool called *SmartBoxes* for fast reconstruction of 3D buildings and facades through intelligent fitting of a set of building blocks directly over acquired point cloud data. An important feature of SmartBoxes is that it capitalizes on the special characteristics of urban models: (i) *orthogonality*: typically these model structures can be effectively approximated by an aggregate of axis-aligned boxes, our building blocks, since architectural scenes mostly consist of orthogonal axis-aligned planes, and (ii) *regularity*: at varying scales, the buildings exhibit a high degree of symmetry with their sub-structures often regularly repeated. While there exist methods to *detect* these symmetries and repeated patterns [Pauly et al. 2008], our task lies in adapting to these data regularities on-the-fly during *interactive reconstruction* of 3D architectural structures using a collection of boxes. Due to the generally poor quality of captured data, we argue that some level of user interaction is required to steer the solution search and achieve interactive speed. The user mainly imparts high-level decisions which involve understanding the semantics of the given model and assists SmartBoxes in recognition tasks which would otherwise take significantly longer time to carry out.

SmartBoxes allows fitting of boxes to acquired architectural data using the notion of *snapping*. The user is only required to loosely specify the extent of a box via region selection or drag-and-drop operations or define a rough logical grouping relation between a set of boxes. The tool automatically computes the sizes, locations, and arrangements of the boxes on-the-fly through an optimization, exploiting strong local relations among them as implied by the user knowledge and inherent data regularity. The boxes are said to be "smart" since unlike conventional snapping, their optimization considers both a fitting to the point data and the local contextual spatial relations among the boxes. The optimization is thus steered by two complementary forces defined respectively by a data-fitting term and a contextual term. By associating a confidence measure to data fitting which is aware of the data quality, we allow the contextual term to play a more dominate role in the optimization when the data-fitting confidence is low, achieving an adaptive balance between the two snapping forces.

In most less-than-perfect scenarios with architectural models and their laser scans, there is a combination of model irregularity and poor data capture quality, yet some level of data regularity and high confidence in data fitting can be detected. Our technique is designed to exploit these data characteristics so that in such cases, the assistance of the user is typically minimal to allow SmartBoxes to complete a quality reconstruction automatically. In cases where both the data-fitting and contextual forces are weak, e.g., over areas where there is no usable data due to large-scale occlusions, the user is always prompted to take control to more heavily influence the reconstruction. We demonstrate that by combining the interactive guidance and high-level domain knowledge from the user with a balanced and adaptive utilization of the snapping forces, our SmartBoxes tool allows the reconstruction of architectural structures which are partially or even completely missing in the input data. Figures 1 and 2 show a few such reconstructions of facades and full buildings, all achieved through short interactive sessions.

2 Related work

The existing body of work on architectural and urban space modeling has become quite large. The majority of the works have been on procedural modeling and more recently on automatic reconstruction from image or video collections. We refer the reader to the recent survey by Vanegas et al. [2009] for a comprehensive coverage. Here we only focus on previous works most closely related to ours, in particular those on the use of primitive fitting and interactive techniques for architectural model reconstruction.

Man-made objects are often composed by an assembly of basic primitive shapes exhibiting regular structures. This is particularly true for architectural models, which are predominantly made out of repetitions of box-like structures having axis-aligned planes. In the area of shape analysis and automatic repetition detection, [Pauly et al. 2008] and [Bokeloh et al. 2009] present approaches for detecting repetitive structures in 3D scans. Their algorithms find intra-shape symmetries by assuming an underlying guiding grid and a limited set of allowed transformations.

Few works in computer graphics exploit these repetitive patterns through priors fitting for shape reconstruction. Gal et al. [2007] fit a small set of basic shapes to local neighborhoods in an input scan at multiple scales via partial matching. The scan is augmented with noise-free samples, quality normals, and sharp features to aid the reconstruction. Schnabel et al. [2009] present a hole-filling algorithm that is guided by primitive detection in the input point cloud. The search for primitives that provide good local fits to data can be difficult due to noise and missing data. In our work, we make the process semi-automatic with the user giving coarse-level guidance

on the extent of the fitted primitives to avoid expensive search.

Procedural modeling of urban buildings and facades [Parish and Müller 2001; Wonka et al. 2003; Müller et al. 2006; Müller et al. 2007] focuses on generating synthetic rules or grammars while respecting architectural principles and exploiting predictabilities in the modeled structures. Works on automatic reconstruction of urban scenes have mostly been based on collections of photos [Werner and Zisserman 2002; Dick et al. 2004; Goesele et al. 2007; Sinha et al. 2008; Xiao et al. 2008; Furukawa et al. 2009b] or multi-view video [Pollefeys et al. 2008], relying on photogrammetric reconstruction and image-based modeling techniques.

Our approach to architectural reconstruction follows the seminal work of Debevec et al. [1996], in which the user interactively fits polyhedral primitives based on information from photographs. Specifically, the user manually matches edges in the images to edges in the model. Alignment of the model components to those in the photographs as well as the camera positions are computed by minimizing a non-linear photogrammetric objective function. In our work, we apply interactive modeling over a 3D point cloud.

The setting of our work is thus closer to that of Schindler and Bauer [2003]. They present a model-based method for reconstructing buildings from range images, where the input 3D points are obtained through image matching and bundle adjustment. The points are first segmented into main wall planes and their principal directions. Edge features are then detected and fitted using rectangular, circular and elliptic contour priors. For each contour, the orthogonal offset from the wall plane is computed by examining points inside the contour, yielding a 2.5D extrusion in the walls. In contrast, our data is fully 3D and the model fitting is more involved. We also explore regularities in the input point data and allow user guidance and grouping during reconstruction with 3D priors.

Recently, more research has been devoted to interactive modeling of architectural structures. Sinha et al. [2008] present an interactive image-based modeling system for reconstructing piecewise-planar 3D structures. To recover camera poses and a sparse 3D point cloud, they use automatic feature matching, structure from motion, and automatic and user-defined vanishing points. The user sketches 2D lines of planar sections over photographs which are automatically mapped onto 3D by aligning to vanishing points or existing point geometry. Xiao et al. [2008; 2009] present, respectively, automatic and semi-automatic image-based approaches for facade modeling that use images captured along streets. They assume planar rectangular facades where details are essentially 2.5D rectangular elements on top of them. A facade is decomposed into rectilinear patches, and each patch is then augmented with a depth value optimized using the structure-from-motion depth data. The system allows user interaction in 2D image space for controlling the decomposition and depth augmentation.

In the semi-automatic reconstruction work of Chen et al. [2008], the user provides free-hand 2D sketches as guidance. Their technique is based on maximum likelihood formulation to interpret the sketches and reconstruct 2.5D geometry by identifying junctions, edges, and faces, followed by reconstruction of geometric objects. The user can add detailed geometry and textures by matching user sketches to a database of detailed geometry and texture models. An elaborate procedure is performed to infer 2.5D from single-view sketches, where the user is required to specify camera parameters. However, limiting the interaction to 2D sketches is still prone to misinterpretation due to noise and ambiguous depth.

Jiang et al. [2009] present an interactive technique for 3D modeling of architectural objects from a single image based on available symmetries. First, they recover the camera model by fitting a frustum prior and non-linear optimization. Then a set of 3D points is reconstructed from stereo using the original and reflected camera and image. The user interactively marks planar components whose positions are automatically determined from the 3D points. Nevertheless, the algorithm assumes a simple reflective symmetry and interactive reconstruction is guided by a sparse set of 3D points.

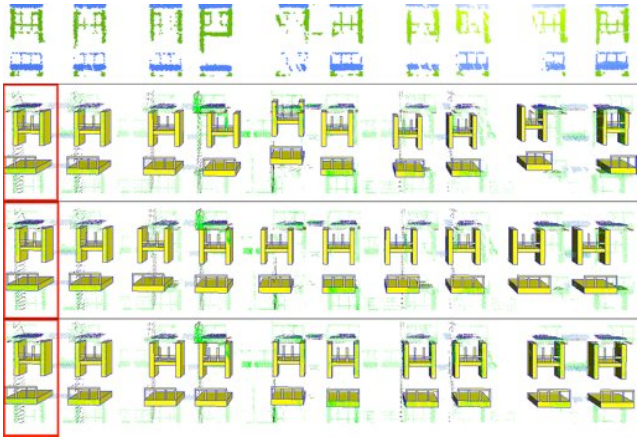

Figure 3: *Effects of the data-fitting and contextual forces on reconstruction, where a compound balcony-window SmartBox (shown in the red square) is dragged to snap to a noisy and sparse point cloud (top row). Second row: data-fitting force only. Third row: contextual force only. Last row: combining two forces so that inconsistencies in the data are disambiguated by the contextual force.*

Früh et al. [2005] present a pipeline for registering, merging and processing scanned rooftops and facades. Assuming nearly regularly sampled 2.5D depth images that are registered with airborne scanned rooftops, they detect objects, separate foreground from background, smoothly complete occlusions and triangulate neighboring points. The recent work of Hohmann et al. [2009] presents a work-flow for the automatic reconstruction of urban scenes by combining airborne LiDAR 3D point clouds, street-level 2D images and shape grammars [Wonka et al. 2003]. 2D facade images are segmented into facade elements manually and 3D point cloud is segmented into planes using RANSAC to retrieve the depth map for ortho-photos. Both segmentations are used to generate a grammar representation for a facade: the same z-value is a strong clue for identical grammar symbols. A shape grammar is computed bottom up and serves as the facade representation. Their process essentially extends the grammar generation of [Müller et al. 2006] with depth values. Nevertheless, since LiDAR 3D data is often too coarse in practice, e.g., as shown in Figure 1, they should merely serve as a guidance and user assistance becomes necessary.

3 Overview

SmartBoxes is an interactive tool for reconstructing architectural structures directly over a scanned point cloud. With the user providing loose guidance through several interactive operations, the tool automatically adjusts the sizes and positions of the building blocks on-the-fly to fit the given input data, respecting both data-fitting and contextual reconstruction criteria. Characteristics of architectural models including orthogonality and regularity have been accounted for in the design of SmartBoxes to take advantage of available data while allowing sensible reconstruction even over areas where the data is completely missing; see Figure 2 and also Figures 1 and Section 7 for some such results.

The basic primitive, the fundamental building block of our recon-

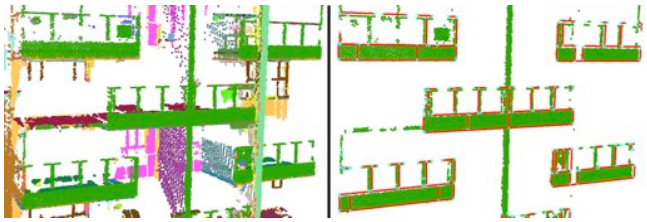

Figure 4: *Results of plane and edge detection. Left: a piecewise planar decomposition of the point cloud using RANSAC; all points on a plane are drawn by a unique random color. Right: edge segments (red) detected from one such plane using a line sweep procedure to identify gradient extrema.*

struction, is a *simple SmartBox*, an axis-aligned rectangular cuboid uniquely specified by its center and a diagonal vector. To allow the modeling of more complex repeatable architectural structures, we define a *compound SmartBox* as a set of simple SmartBoxes, not necessarily connected or adjacent, that are grouped and always act as an aggregate. While a compound SmartBox merely defines a logical relation, our system detects geometric and topological relations such as connectivity, intersections and alignment between the boxes in the compound. Throughout the paper, we refer to both a simple and a compound SmartBox as a SmartBox (or for brevity a box), unless a distinction is called for.

After preprocessing a raw input scan to detect the plane and edge primitives therein (Section 4), reconstruction by SmartBoxes is through a data- and context-driven optimization with initial guidance provided by user interaction. The interactive operations, as detailed in Section 6, consist of initializing a simple SmartBox by loosely defining its extent using a *2D rubber-band box*, *grouping* to specify a compound SmartBox, *drag-and-drop* of SmartBoxes to guide the generation of multiple SmartBox instances, and *automatic continuation* for SmartBox replication, even over regions completely void of input data. Figure 2 shows SmartBox grouping (into window railings and a balcony) and the drag-and-dropping of a compound SmartBox, a column, for the reconstruction of an entire building facade.

With interactive user guidance providing a rough initialization, our automatic reconstruction algorithm finds the best sizes and positions of the SmartBoxes via an optimization which balances between the data-fitting (to input points) and contextual forces in the current interactive operation. Figure 3 illustrates the effects of the two forces on SmartBox reconstruction and shows that the best result is achieved by combining the two forces. Whenever possible, the algorithm automatically detects repetitions and infers reconstruction of repetitive patterns that are already present. This in turn minimizes the required user interaction.

The data-fitting force aims at best fitting the boxes to the point cloud geometry. It measures how well the SmartBoxes explain the data, by measuring the distance between the boxes and the point cloud. The contextual force aims at regularizing the *local* inter- and intra-box relations. By assuming a "Manhattan world" [Furukawa et al. 2009a] (scenes consisting predominantly of axis-aligned piece-wise planar surfaces), the contextual force aligns SmartBoxes with common horizontal and vertical planes, forcing regularity in the repetition of patterns as well as topological consistency. We elaborate on these forces and our optimization procedure in Section 5.

4 Data acquisition and preprocessing

Before presenting the SmartBoxes tool in detail, we briefly describe the data acquisition process and preprocessing steps. The input scans in our work have been obtained from an Optec Lynx LiDAR scanner mounted on a street-level vehicle. The scanner captures a surrounding urban scene while the vehicle is moving at normal driving speed. Since the data is acquired from the street level, it is common that various occlusions would occur so that many parts of the buildings are far from being accurately captured. For example, in Figure 1 we see that the "shadow" of the trees appear as large data gaps over the point cloud of the facades. This occurs after one removes those points corresponding to the trees which occluded the actual facades. As well, the obtained scan data are typically quite noisy, with low resolutions, and outlier-ridden due to the presence of reflective objects such as windows and other glossy structures. To overcome such multitude of poor data qualities, user interaction becomes necessary. The user possesses certain domain knowledge about the modeled structures and can also be assisted by other cues such as those obtained from photometric images concurrently taken with the geometry scans.

Since architectural scenes are typically characterized by piece-wise planar surfaces with dominant directions, we assume that the building structures to be reconstructed are aligned with three dominant directions, corresponding to the ground plane and two additional orthogonal axes. As we will explain in Section 5, this assumption only serves to reduce the SmartBox search space to accelerate interaction. As our LiDAR scanner is coupled with a GPS and a motion tracker, the acquisition process yields a coherently registered point cloud which is aligned with the local ground plane; this allows us to identify the ground plane direction. The determination of the other two directions requires plane extraction.

During preprocessing of a raw input point cloud, we first extract all planar components from the scene using RANSAC. Points that have weak or no support by the extracted planes are considered as outliers and are discarded, while the remaining points are projected onto the nearest planes to reduce noise. Note that such a processing step is fairly standard for acquired architectural data, e.g., see [Schnabel et al. 2007]. After plane detection, we extract edge segments per plane in two steps. First, lines are extracted using a horizontal and vertical line sweeping algorithm in the plane; the gradient extrema in the point distribution along the sweeps give extracted horizontal and vertical lines, as done in [Schindler and Bauer 2003]. The end points of the edge segments are then determined by a 1D version of the line sweeping along the corresponding lines; see Figure 4 for some results.

Through plane detection, we can find the dominant horizontal and vertical directions for each plane. As most planes are orthogonal or nearly orthogonal to the ground plane, we infer the two dominant in-plane directions from orthogonality with the ground up-vector. If planes are not orthogonal to the ground plane, we detect in-plane lines using RANSAC followed by a line clustering step. The dominant horizontal and vertical directions in this case correspond to the two largest orthogonal line clusters.

5 SmartBox snapping

Regardless of the interactive operation, the fundamental computational task involves the determination of the parameters which define a SmartBox, either a simple or compound one, based on the given point cloud data and a subset of existing SmartBoxes. Through an optimization, the SmartBox is snapped by combining a data-fitting force and a contextual constraining force. These two forces constitute the terms of an objective function $M(B, \hat{B}, P)$

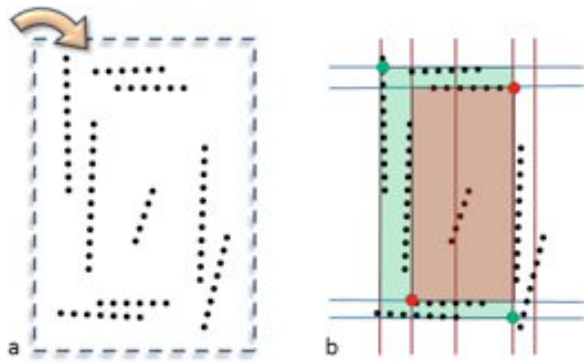

Figure 5: *Snapping a SmartBox via data fitting. The user loosely marks an area denoted by a dashed box (left). Candidate boxes are detected by permuting intersections between horizontal and vertical edges (right). Here only two such candidates are shown. The best fitting box (red) is computed using the data-fitting term alone.*

for defining a SmartBox instance B over the given point cloud P in the context of a reference or *contextual* SmartBox $\hat{B}$. Since a compound SmartBox is merely an addition of simple boxes, the associated optimization simply uses a sum of the objective functions for the simple boxes. For a 2D illustration showing the two terms at work, refer to Figures 5 and 6. In the following we first elaborate on the two terms, then we define the optimization problem and describe our solution scheme.

Data-fitting term Given a SmartBox B and the point cloud P, the data-fitting term measures the fitting of the facets and edges of B to points in P while accounting for an appropriate notion of *confidence*. The confidence value is intended to be correlated with data quality and it is defined at a point in the point cloud while considering the local neighborhood around the point.

- The facet-fitting term $F(B, P)$ measures a fitting of all facets f in B to points p in the data,

$$F(B,P) = \sum_{p,f|dist(p,f)<\varepsilon} \frac{conf_P(p) \cdot (1 - dist(p,f)/\varepsilon)}{area(f)}, \tag{1}$$

  The confidence term $conf_P(\cdot)$ measures the local uniformity of the point cloud data near point p; it is computed by examining the local covariance matrices defined at p, as in [Pauly et al. 2005]. Specifically, we compute for p the covariance matrices for its local neighborhood of points at three static scales. $conf_P(\cdot)$ is then an average of the ratios of the eigenvalues (λ_2/λ_1) of these matrices. Each eigenvalue ratio takes on a value between 0 to 1, corresponding to a perfect line and disc distribution correspondingly.

  The term $dist(\cdot)$ measures the Euclidean distance between a point and a facet. We consider only points that are ε-*close* to f, i.e., points p satisfying $dist(p, f) < \varepsilon$.

  The confidence and area terms in (1) favor fitting to densely sampled uniform data regions, while the distance term favors fitting facets that are close to the point cloud. A higher value of $F(B, P)$ signifies a better facet fitting. The main benefit offered by the confidence term arises when the data quality, in terms of point uniformity, is poor and the confidence value is low. In this case, the importance of face fitting is diminished so that other snapping forces, e.g., contextual or edge fitting, are allowed to play a more dominant role.

- The edge-fitting term $E(B, P)$ is defined similarly and measures a fitting of all edges e in B to a subset of points p' in the point cloud P which belong to detected edge segments,

$$E(B,P) = \sum_{p',e|dist(p',e)<\varepsilon} \frac{conf_E(p') \cdot (1 - dist(p',e)/\varepsilon)}{length(e)}, \tag{2}$$

  Recall from Section 4 that the data edge segments have been computed by observing sharp changes in point distribution along horizontal and vertical directions, yielding reasonably uniformly sampled edge segments. We designate the confidence term $conf_E(\cdot)$ in the edge-fitting case at a point p' belonging to a data edge segment as the gradient of the point distribution at p' [Schindler and Bauer 2003].

  The term $dist(\cdot)$ in the edge-fitting case (2) measures the Euclidean distance between a point p' residing on a detected edge segment and an edge e in the SmartBox. In the same manner, we consider only ε-close points to edge e. The confidence and length terms favor fitting to densely sampled uniform edges, while the distance term favors fitting edges that are close to the point cloud edges. Again, a higher value of $E(B, P)$ signifies a better edge fitting.

We normalize both facet- and edge-fitting terms independently to the unit interval $[0, 1]$ by recording the maximal and minimal values at each snapping operation. With an equally weighted combination for a minimization, we obtain the energy for data fitting as

$$D(B,P) = 1 - \frac{1}{2}(F(B,P) + E(B,P)).$$

Contextual term The contextual SmartBox $\hat{B}$ required to define the contextual force is a previously positioned SmartBox in a reconstruction sequence. The precise definition of context depends on the type of interaction performed, hence we explain this in detail in Section 6. Given $\hat{B}$, the contextual term for snapping the SmartBox B is defined as the sum of three terms (in the case of compound SmartBoxes, B and $\hat{B}$ refer to the bounding box of the SmartBox for the interval and size measurements below):

- The interval term $I(B, \hat{B})$ measures how well the interval length between B and $\hat{B}$ agrees with the expected interval length out of regularity constraints; see Figure 6(a). Let Δ be the expected interval length for a vertical or horizontal box sequence, then

$$I(B,\hat{B}) = |\|center(B) - center(\hat{B})\|_2 - \Delta|.$$

  Again, we defer the definition of the expected interval length to Section 6 since it depends on the context creation.

- The alignment term $A(B, \hat{B})$ measures how well corresponding edges of B and $\hat{B}$ align; see Figure 6(b). Let $l_{ext}(e)$ denote the line extension of an edge segment e, then the alignment term is a sum over the corresponding edge pairs,

$$A(B,\hat{B}) = \sum_{\hat{e}\in\hat{B} \text{ and } e\in B \text{ correspond}} \|l_{ext}(\hat{e}) - l_{ext}(e)\|_2.$$

- The size term $S(B, \hat{B})$ measures the size difference between the boxes; see Figure 6(c),

$$S(B,\hat{B}) = \max(diag(B)/diag(\hat{B}), diag(\hat{B})/diag(B)),$$

  where $diag(B)$ is the length of the box diagonal.

Here again, we normalize the three terms independently to $[0, 1]$ and arrive at a contextual energy term to be minimized,

$$C(B, \hat{B}) = \frac{1}{3}(I(B, \hat{B}) + A(B, \hat{B}) + S(B, \hat{B})).$$

SmartBox optimization The optimization problem can be posed as finding an optimal linear transformation T^*, consisting of translation and non-uniform scaling of the axis-aligned SmartBox B, to minimize a weighted sum of contextual and data-fitting terms,

$$\begin{aligned} T^* &= arg \min_{\langle T \rangle} M(B, \hat{B}, P) \\ &= arg \min_{\langle T \rangle} \omega \cdot D(T(B), P) + (1 - \omega) \cdot C(T(B), \hat{B}), \end{aligned}$$

where ω is a weight which balances between the data-fitting and contextual forces. The choice of the weight ω should correlate with an assessment of contextual regularity and data quality in the underlying architectural model to be reconstructed.

In our method, we include a mechanism which automatically adjusts the weight ω by extrapolating from the immediate past history of data-fitting and context-fitting errors. Such a mechanism is only enabled during a drag-and-drop sequence (see Section 6), where the user continuously snap SmartBoxes in succession. When the average data-fitting error in the sequence is high, attesting to poor quality of data in the vicinity of the operation, the fitting to that data is of less importance in the optimization so that the weight ω is decreased to allow the contextual force to play a more dominant role; see Figure 7 for an illustration. Similarly, when the user's drag-and-drop targets start to exhibit a high degree of irregularity, causing the contextual term to incur a large error, ω is increased to place more emphasis on data fitting, as shown in Figure 8. Typically at the start of a drag-and-drop reconstruction sequence, no prior information is available and we start with the two forces equally weighted. The system records the average error in data-fitting $\bar{D}(\cdot)$ and context $\bar{C}(\cdot)$ through a drag-and-drop sequence. In each step, if the ratio $\bar{D}/\bar{C}$ is higher than 2 (respectively, lower than 0.5), we increase (respectively, decrease) ω by 0.05, while keeping ω in the unit interval. Thus, the weight ω is automatically adjusted on-the-fly. Nevertheless, the user can always manually adjust it according to perceived contextual regularity and also in situations where data becomes too sparse or completely missing to allow a meaningful measure of data-fitting confidence to be computed.

Our formulation of SmartBox snapping above results in a continuous non-linear optimization problem with a six-dimensional search space. A direct search over the entire space is expensive and hinders real-time performance. It is also redundant in our work as we can exploit again the special characteristics of architectural structures. For one, it is rare for contextual criteria, e.g., alignment, to not be satisfied exactly. Also, despite the possible noise and potential high sparsity of the input point data, there are still a large number of detected planes and edges. Therefore the optimization problem we are facing is highly constrained by the data and contextual information. This motivates the use of a combinatorial search over a restricted set of candidate solutions constrained by the data and context.

Specifically, the search for the optimal SmartBox is over a discrete set consisting of data-driven candidates as well as data-aware candidates constrained by contextual forces. Data constraints are enforced by detected edges as well as 3D corner points given by intersections between triplets of detected planes. The data edges and planes must pass a confidence threshold as appearing in equations (1) and (2) to be considered. Three corner points which do not all lie in the same plane define a data-driven candidate box. Additional

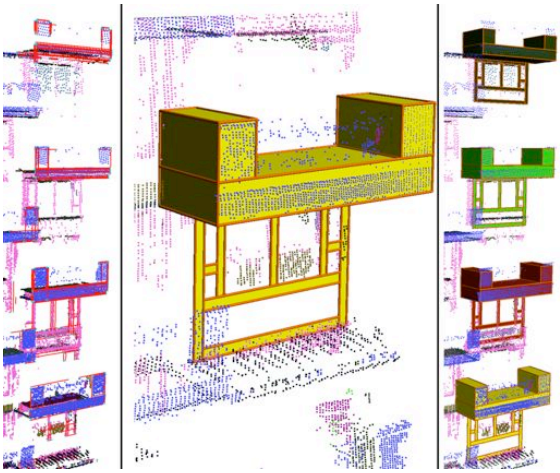

Figure 7: *Automatic adjustment of data-fitting and contextual forces during a drag-and-drop sequence. Starting with a given column of scanned balconies (left) and an initial SmartBox snapped to one instance (middle), the user drag-and-drops it from bottom to top (right). As data quality deteriorates (from bottom to top) the data-fitting force decreases (red: dark to light). In parallel, the contextual force starts to dominate (green: light to dark).*

candidates are generated by boxes whose edges or corners coincide with some data edges or corners and whose geometry also fulfills constraints provide by one or more sub-terms of the contextual force, including interval length, alignment, and box size. Figure 6 provides 2D illustrations of some of the candidate boxes generated this way. Note that only edges and corners confined by a *region of interest* (ROI) of the reconstruction are considered in the formation of the candidates. In the SmartBoxes tool, user interactions provide rough initialization for the desired box B, which defines the ROI, as we explain in Section 6.

Under typical scenarios, the number of candidates is not large, ranging from dozens to up to two hundred. This is expected since the user provides a rough initialization of the SmartBox sought through interaction and this, along with the data-fitting and contextual forces, is often sufficiently constraining. As a result, the combinatorial search has proven to be quite effective. Given the set of candidate boxes, we evaluate them using the objective function and take the one with the lowest value as our solution. In all of our reconstruction experiments, snapping of the SmartBoxes via such a discrete optimization can be computed in real time, as we demonstrate in the accompanying video.

6 Interaction with SmartBoxes

In this section, we detail the interactive operations we employ in the SmartBoxes tool. The first is the use of a 2D rubberband box in screen space to loosely define a simple 3D SmartBox. Next, is the core interactive operation in our tool of drag-and-drop of a simple or compound SmartBox to generate multiple instances. Then we present an automatic continuation operation, which extends SmartBox reconstruction as driven by the contextual force. We include it here since the operation can be regarded as one resulting from the generalization of a sequence of one or more drag-and-drop operations. Finally, we describe the grouping of simple SmartBoxes into a compound entity and the accompanying analysis and processing within the group. Also in this section, we discuss expanding our interactive reconstruction tool to include primitives other than axis-aligned cubes and their aggregates, such as sections of cylinders.

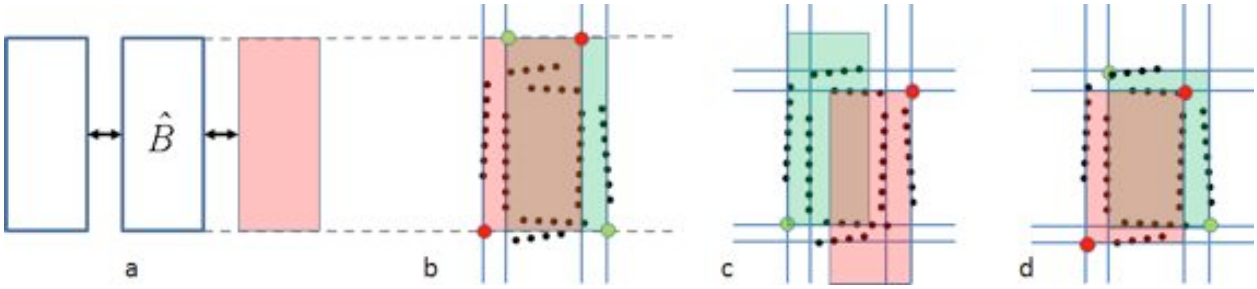

Figure 6: *2D illustration of candidate SmartBoxes B (pink, green). The combination of contextual and data-fitting terms generates a set of candidates (only a subset is shown) by considering a contextual box $\hat{B}$ to infer the interval length (a), alignment (b), and size (c) parameters for B, adjoined with detected edges in the data (blue lines). Additional data-driven candidates (d) are defined by edge intersections.*

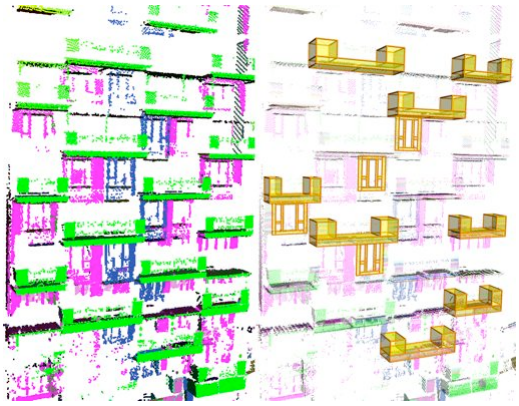

Figure 8: *With data exhibiting low contextual regularity (left), the SmartBoxes tool automatically detects the condition during a drag-and-drop sequence. The generated boxes are scaled and aligned primarily according to the data-fitting term (right).*

Interactive reconstruction sessions consisting of all types of user interactions can be viewed in the accompanying video.

SmartBox snapping initialization The primitive-building operation of our interaction framework is through a loose definition of the extent of a simple SmartBox over the given point cloud. The user specifies the extent by using a rubberband square which loosely fits the region over which the box is to be created. The marking of the square is performed in screen space which in turns is used to collect all visible points and corresponding planes that reside within this region in 3D space. To relieve the user from having to precisely draw a rubberband to contain the sought-after SmartBox, we apply analysis and optimization over point data which reside in a larger region than the perspective frustum defined by the rubberband square; this defines the region of interest (ROI) of the SmartBox initialization. In our current implementation, we use a ROI that is twice as large as, and centrally aligned with, the rubberband square specified by the user. Planes that belong to distant facades and project into the ROI are removed by sorting them in view direction and removing far planes. Given the ROI, we perform the discrete optimization as described in the preceding section to obtain the SmartBox solution; a 2D illustration is given in Figure 5. In this mode of interaction, only the data-fitting term is considered for the optimization and no contextual SmartBox needs to be specified.

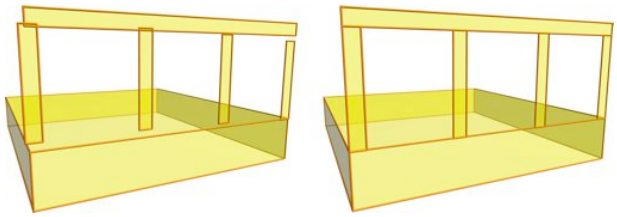

Figure 9: *Grouping several boxes (left) triggers an automatic repair of a few forms of inconsistencies, e.g., intersections, resulting in a more regularized and well-aligned compound box. (right).*

Drag-and-drop The essential interactive operation in the SmartBox tool is the drag-and-drop paradigm for creating complex structures through the propagation of a repeated structure, in our case, a SmartBox. The user selects a SmartBox and then drags-and-drops to a new target location in the vicinity. Similarly to the initialization of a simple box as described above, we apply the discrete optimization as described in the preceding section over the ROI defined by the drop operation. In this case, contextual information is utilized to define the candidate set since the drag-and-drop operation naturally implies a contextual relation between the original box and its copy. In particular, a sequence of drag-and-drop operations define an expected interval length Δ measuring the average of the interval lengths between consecutive SmartBox instances encountered along the sequence so far. In Figures 7 and 8, as well as Figure 2, we show some results of user drag-and-drop interactions.

Automatic continuation As the user performs drag-and-drop operations, contextual information is being formed and such information can assist in reconstruction even over areas where the data is highly sparse or completely missing. This is exploited in our design of the context-driven automatic continuation operation.

Specifically, we track and analyze every drag-and-drop operation performed by the user. Continuation essentially computes an amortized drag-and-drop average by averaging the current drag-and-drop interval, size and direction with all the previous ones. Thus, we automatically attempt to extend the reconstruction, following along the interval direction and distance, by copying the repetitive model to its next position. Once the next position is computed, we examine its neighborhood and find candidates as in the drag-and-drop step. Nevertheless, we give higher weight to the contextual term than for drag-and-drop.

SmartBox grouping Grouping allows the user to select several SmartBoxes to form a compound SmartBox, which is always ma-

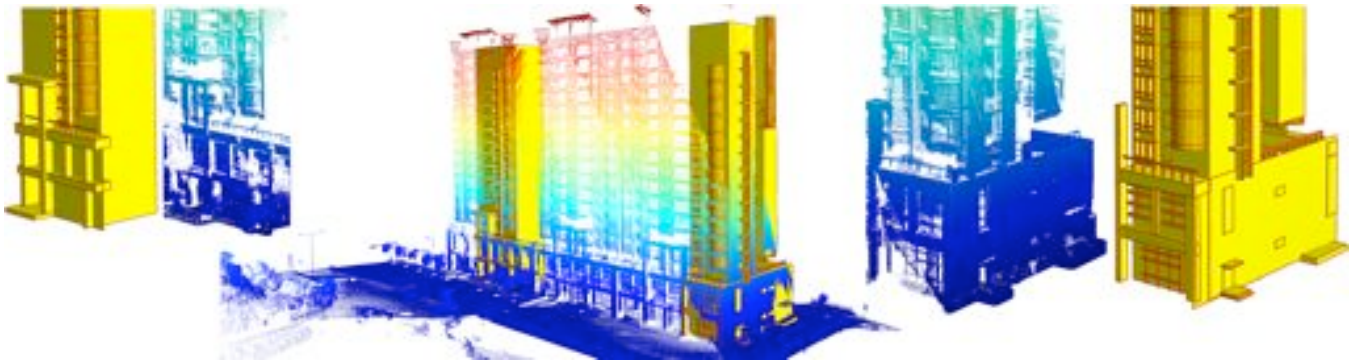

Figure 10: *Reconstruction of a building containing cylinders, along with other complex architectural structures.*

nipulated and transformed as a whole. The outcome of the operation is two-fold. First, it yields a *logical* grouping of the selected boxes allowing the creation of complex structures out of simple ones. By a logical grouping, we refer to the forming of only a symbolic link between the selected boxes — they are not physically connected. The second outcome is an automatic analysis of the grouped boxes to repair any inconsistencies among them.

By specifying that a set of boxes belong to a group, where the user intent is to have such a group represent a canonical architectural structure such as a balcony, a context is created. We respect such a context by enforcing alignment among the grouped boxes which would resolve the kind of inconsistencies we wish to repair, e.g., box intersections, small gaps, and other forms of misalignments. In Figure 9, we demonstrate the grouping of a set of boxes and rectangles, which are zero-depth boxes treated as regular boxes.

During grouping, we repair the inconsistencies by clustering and aligning close to co-linear edge segments. Specifically, given a grouped compound of boxes, we first cluster their edge segments by orientation. Due to the assumption of in-plane dominant horizontal and vertical directions (Section 4), all edge segments are axis-aligned and grouping by orientation is straightforward. Among edges in an orientation group, we need to further partition them into co-linear edge clusters. This is achieved by projecting centers of edges with same orientation onto a perpendicular plane and perform clustering based on Euclidean distances between the projections in that plane. The centroid of each cluster of projections, along with the respective orientation, uniquely determine a representative line onto which the corresponding edges are projected; these edges are assumed to be co-linear within the group.

Other types of building blocks The snapping technique and the interaction paradigm we have developed so far are not confined to axis-aligned boxes. In principle, we can incorporate any parametric primitive that can be detected using standard RANSAC. In our current implementation, we add cylinders and their sections, which are common structures in architectural models, to our modeling toolbox. A SmartCylinder is defined by an axis (assumed to be vertical), an interval over height, and a radius. These parameters are used to define the contextual term in the snapping force, in similar ways as the alignment, interval, and size parameters of SmartBoxes. The data-fitting term for SmartCylinders is also defined similarly to model fitting of the snapped cylinder to point data while using the same confidence measure; edge fitting is ignored.

To execute the combinatorial search for a SmartCylinder, we run RANSAC over the point data in the ROI of the user interaction to identify sections of cylinders. Each section defines an axis and a radius to anchor a candidate cylinder section; these constraints serve the same purpose as the 3D corners which define the candidate boxes. The remaining degrees of freedom are fixed by contextual constraints similarly as before. The optimal SmartCylinder is the candidate which gives the lowest combined energy. A reconstruction result containing SmartCylinders is shown in Figure 10.

7 SmartBoxes reconstruction results

The LiDAR data used in our experiments were collected using a commercially available mobile laser scanning systems Lynx from Optech Inc. and StreetMapper (3DLS). The system can acquire 3D points at close range (100-300m), high frequency (200K points per second), and high precision (within 5cm positional accuracy) with the scanner moving at normal driving speed. Compared with airborne laser scanning, ground-based scanning can acquire urban scenes from up-close with more meaningful geometric details.

Several reconstruction results using SmartBoxes have already been shown. We observe that the scan data is always partial and noisy. Nevertheless, it often contains sufficient hints to allow the user to use SmartBoxes to reconstruct the scene down to fine details, demonstrating the quality and versatility of the tool. Additional results can be found in Figures 11, where we show a photograph of the scanned scene, a visualization of the point cloud, the geometry reconstructed by SmartBoxes, and finally the results of a re-touching by an artist, who added textures, large side walls when data was completely missing and a stylistic shader. Note that in this figure as well as Figures 1 and 10, raw points are colored by height. In figures containing preprocessed point clouds, the colors correspond to different detected planes, unless otherwise specified.

In our work, we pay more attention to the reconstruction of significant 3D structures in a scene and less to flat facades. However, as can be seen in Figures 1 and 11(b), SmartBoxes can reconstruct well flat details such as windows as well as more volumetric structures such as balconies. The user can carefully select a region over a reliable data region to construct the geometry of a repeated structure, and then drag-and-drop it in other regions, where the data quality deteriorates or is completely missing. It is often effective to first drag-and-drop the SmartBoxes along a row or a column to form a larger compound, which is more powerful at completing missing data regions, as shown in Figure 2.

A central property of our SmartBoxes tool is its ability to exploit regularity in the data, through the contextual force. However, when the regularity is broken, as in Figure 11(a) (the upper floor contains a single balcony) or in Figure 11(b) (the center of the building is not a sub-grid), the user can still utilize the snapping power of the

Figure	#points	#polygons	Time
1	2,152,0913	13,500	45 m
10	7,610,426	4,390	20 m
11(hi)	419,422	4,984	8 m
11(mid)	2,865,358	3,885	20 m
11(low)	94,578	6,015	7 m

Table 1: *Point cloud size and SmartBoxes reconstruction times for the different results shown in the figures. The total time (in minutes) spans the whole reconstruction process, from raw point clouds to the final detailed reconstructed 3D models.*

SmartBoxes and their repetition through drag-and-drop. Observe that the balconies in Figure 11(a) are not of the same size; when the user drags one balcony over an area occupied by a larger instance, the data-fitting force allows it to adapt to the proper size on-the-fly.

As an interactive tool, the performance of SmartBoxes is best evaluated from viewing the reconstruction sessions in the accompanying video. User interactions are real-time, enabling the reconstruction of large-scale scenes which contain a complex and large variety of 3D structures. Total reconstruction times range from several minutes to an hour, depending on the scene complexity, data quality, and contextual regularity that can be exploited; see Table 1.

Our system was specifically designed to handle significant noise common in scanned urban scenes by enhancing data-fitting with contextual forces and user interactive guidance. Hence, our system's sensitivity to parameters is low, enabling us to tune the parameters to a consistent setting. Through all the examples, we normalized the scene bounding box diagonal to 1 and used the following parameter values: $\varepsilon = 0.0025$ for the ε-closeness threshold (Section 5); minimum number of points for RANSAC detection: 100; plane/edge confidence threshold: 0.001; line sweep edge detection interval: 0.0025; three neighborhood sizes for computing confidence value $conf_P(\cdot)$: ε, 2ε, 10ε; ω: the weight between data-fitting and contextual forces is a system/user controlled parameter ranging between 0.3 and 0.8 in our experiments.

8 Conclusions and discussion

We present SmartBoxes, a new interactive tool which enables fast reconstruction of buildings and facades. The tool possesses many characteristics common to a 3D shape editing system, including those designed for architectural models. However, it is unique in that here the construction is performed directly over point samples acquired as raw LiDAR scans. It is typical for such data to be low-resolution, be tempered with a high-level of noise, and above all, contain large areas with missing data due to occlusion and other scanner artifacts. The main strength of the SmartBoxes tool is that it allows sparse traces of the data to be sufficient for a quality reconstruction. In this sense, SmartBoxes enjoy and combine the two worlds of 3D editing and reconstruction.

Having the user in the reconstruction loop enables interactive reconstruction from partial or missing data and in possibly ambiguous scenes. The user is only required to impart high-level decisions based on domain knowledge and assist the reconstruction only where fully automatic recognition or interpretation of the data is too time-consuming or largely unreliable. Whenever the available data or the context allows it, Smartboxes would operate automatically and at interactive speed, even in adjusting a balance between data-fitting and contextual forces based on processing history.

We believe that our work is still only a first step in enriching the arsenal of interactive reconstructions techniques for urban scenes. Our current system can and should be extended in at least two ways. First, even more primitives can be included with snapping operations defined for them. Traditional 3D editors have so far been using the notion of snapping mainly on point primitives and only to grid junctions or other geometric primitives. SmartBoxes extends the concept of snapping to higher-level primitives and to higher dimensions by incorporating contexts. This is a strength of the technique to be built upon. A second direction is to extend the notion of context beyond our simple linear sequence definition and resort to analysis of different forms of symmetry and repeated patterns. Finally, we also plan to explore the use of SmartBoxes for the reconstruction of more varieties of architectural models, including those possessing less regularities.

Acknowledgements We thank the anonymous reviewers for their valuable suggestions. This work was supported in part by National Natural Science Foundation of China (60902104), National High-tech R&D Program of China (2009AA01Z302), CAS Visiting Professorship for Senior International Scientists, CAS Fellowship for Young International Scientists, Shenzhen Science and Technology Foundation (GJ200807210013A), and the Natural Sciences and Engineering Research Council of Canada (No. 611370).

References

BOKELOH, M., BERNER, A., WAND, M., SEIDEL, H.-P., AND SCHILLING, A. 2009. Symmetry detection using feature lines. *Computer Graphics Forum (Proceedings of Eurographics) 28*, 2, 697–706.

CHEN, X., KANG, S. B., XU, Y.-Q., DORSEY, J., AND SHUM, H.-Y. 2008. Sketching reality: Realistic interpretation of architectural designs. *ACM Trans. on Graphics 27*, 2, 1–15.

DEBEVEC, P. E., TAYLOR, C. J., AND MALIK, J. 1996. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *Proc. of ACM SIGGRAPH*, 11–20.

DICK, A. R., TORR, P. H. S., AND CIPOLLA, R. 2004. Modelling and interpretation of architecture from several images. *Int. J. Comput. Vision 60*, 2, 111–134.

FRÜH, C., JAIN, S., AND ZAKHOR, A. 2005. Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images. *Int. J. Comput. Vision 61*, 2.

FURUKAWA, Y., CURLESS, B., SEITZ, S. M., AND SZELISKI, R. 2009. Manhattan-world stereo. In *Proc. of IEEE Conf. on Comp. Vis. and Pat. Rec.*, 1422–1429.

FURUKAWA, Y., CURLESS, B., SEITZ, S. M., AND SZELISKI, R. 2009. Reconstructing building interiors from images.

GAL, R., SHAMIR, A., HASSNER, T., PAULY, M., AND COHEN-OR, D. 2007. Surface reconstruction using local shape priors. In *Proc. of Eurographics Symp. on Geometry Processing*, 253–262.

GOESELE, M., SNAVELY, N., CURLESS, B., HOPPE, H., AND SEITZ, S. 2007. Multi-view stereo for community photo collections. In *Proc. of Int. Conf. on Comp. Vis.*, 1–8.

HOHMANN, B., KRISPEL, U., HAVEMANN, S., AND FELLNER, D. W. 2009. Cityfit: High-quality urban recontrustion by fitting shape grammars to images and derived textured point clouds. In *Proceedings of the 3rd ISPRS Workshop*.

JIANG, N., TAN, P., AND CHEONG, L.-F. 2009. Symmetric architecture modeling with a single image. *ACM Trans. on Graphics 28*, 5, 1–8.

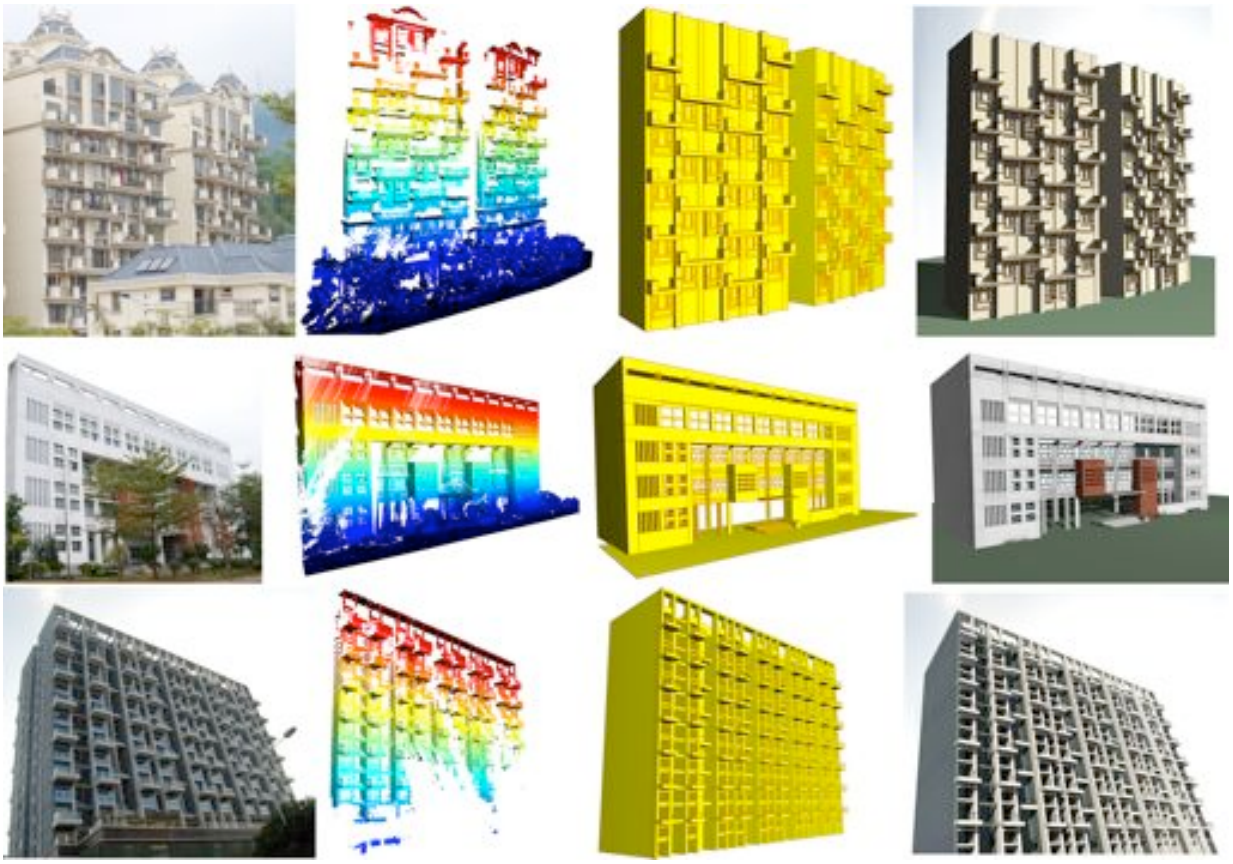

Figure 11: *Additional reconstruction results using SmartBoxes. From left to right: real photograph, LiDAR scan, 3D reconstruction, and its textured version for a visual comparison with the photograph. The examples show reconstruction of complex buildings with some irregularity.Grouping and contextual force during drag-and-drop allow the reconstruction to deal with large-scale missing data (bottom row).*

LEVOY, M., PULLI, K., CURLESS, B., RUSINKIEWICZ, S., KOLLER, D., PEREIRA, L., GINZTON, M., ANDERSON, S., DAVIS, J., GINSBERG, J., SHADE, J., AND FULK, D. 2000. The digital michelangelo project: 3D scanning of large statues. In *Proc. of ACM SIGGRAPH*, 131–144.

MÜLLER, P., WONKA, P., HAEGLER, S., ULMER, A., AND VAN GOOL, L. 2006. Procedural modeling of buildings. *ACM Trans. on Graphics 25*, 3, 614–623.

MÜLLER, P., ZENG, G., WONKA, P., AND GOOL, L. J. V. 2007. Image-based procedural modeling of facades. *ACM Trans. on Graphics 26*, 3, 85.

PARISH, Y. I. H., AND MÜLLER, P. 2001. Procedural modeling of cities. In *Proc. of ACM SIGGRAPH*, 301–308.

PAULY, M., MITRA, N. J., GIESEN, J., GROSS, M., AND GUIBAS, L. J. 2005. Example-based 3D scan completion. In *Proc. of Eurographics Symp. on Geometry Processing*, 23.

PAULY, M., MITRA, N. J., WALLNER, J., POTTMANN, H., AND GUIBAS, L. 2008. Discovering structural regularity in 3D geometry. *ACM Trans. on Graphics 27*, 3.

POLLEFEYS, M., NISTÉR, D., FRAHM, J. M., AKBARZADEH, A., MORDOHAI, P., CLIPP, B., ENGELS, C., GALLUP, D., KIM, S. J., MERRELL, P., SALMI, C., SINHA, S., TALTON, B., WANG, L., YANG, Q., STEWÉNIUS, H., YANG, R., WELCH, G., AND TOWLES, H. 2008. Detailed real-time urban 3D reconstruction from video. *Int. J. Comput. Vision 78*, 2-3, 143–167.

SCHINDLER, K., AND BAUER, J. 2003. A model-based method for building reconstruction. In *Proc. of IEEE Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis*, 74.

SCHNABEL, R., WAHL, R., AND KLEIN, R. 2007. Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum 26*, 2, 214–226.

SCHNABEL, R., DEGENER, P., AND KLEIN, R. 2009. Completion and reconstruction with primitive shapes. *Computer Graphics Forum (Proc. of Eurographics) 28*, 2, 503–512.

SINHA, S. N., STEEDLY, D., SZELISKI, R., AGRAWALA, M., AND POLLEFEYS, M. 2008. Interactive 3D architectural modeling from unordered photo collections. *ACM Trans. on Graphics 27*, 5, 1–10.

VANEGAS, C. A., ALIAGA, D. G., WONKA, P., MUELLER, P., WADDELL, P., AND WATSON, B. 2009. Modeling the appearance and behavior of urban spaces. In *Proc. of Eurographics State-of-the-Art Report*.

WERNER, T., AND ZISSERMAN, A. 2002. New techniques for automated architecture reconstruction from photographs. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, vol. 2, 541–555.

WONKA, P., WIMMER, M., SILLION, F., AND RIBARSKY, W. 2003. Instant architecture. *ACM Trans. on Graphics 22*, 3, 669–677.

XIAO, J., FANG, T., TAN, P., ZHAO, P., OFEK, E., AND QUAN, L. 2008. Image-based façade modeling. *ACM Trans. on Graphics 27*, 5, 1–10.

XIAO, J., FANG, T., ZHAO, P., MAXIME, L., AND QUAN, L. 2009. Image-based street-side city modeling. *ACM Trans. on Graphics 28*, 5, 1–12.

ZEBEDIN, L., BAUER, J., KARNER, K., AND BISCHOF, H. 2008. Fusion of feature- and area-based information for urban buildings modeling from aerial imagery. In *Proc. Euro. Conf. on Comp. Vis.*, 873–886.