DOI: 10.1111/cgf.12788

COMPUTER GRAPHICS *forum*
*Volume 00 (2016), number 0 pp. 1–12*

# Discovering Structured Variations Via Template Matching

Duygu Ceylan[1,*], Minh Dang[2,*], Niloy J. Mitra[3], Boris Neubert[4] and Mark Pauly[2]

[1]Adobe Research, San Jose, CA, USA
duygu.ceylan@gmail.com
[2]EPFL, Lausanne, Switzerland
{minh.dang, mark.pauly}@epfl.ch
[3]UCL, London, UK
niloym@gmail.com
[4]KIT, Karlsruhe, Germany
boris.neubert@gmail.com

**Abstract**

*Understanding patterns of variation from raw measurement data remains a central goal of shape analysis. Such an understanding reveals which elements are repeated, or how elements can be derived as structured variations from a common base element. We investigate this problem in the context of 3D acquisitions of buildings. Utilizing a set of template models, we discover geometric similarities across a set of building elements. Each template is equipped with a deformation model that defines variations of a base geometry. Central to our algorithm is a simultaneous template matching and deformation analysis that detects patterns across building elements by extracting similarities in the deformation modes of their matching templates. We demonstrate that such an analysis can successfully detect structured variations even for noisy and incomplete data.*

**Keywords:** shape analysis, template matching

**ACM CCS:** I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling

## 1. Introduction

Many applications involving digital cities, such as mapping and navigation, heavily depend on three-dimensional (3D) models of buildings. One way to create such content is to digitize real-world scenes using different 3D acquisition technologies such as multi-view stereo reconstruction (MVS) or 3D scanning. However, many challenges arising from lighting variations, occlusions, specular surfaces still remain unsolved, and often result in noisy and partial reconstructions.

A possible approach to provide effective priors to augment reconstruction algorithms is to explore similarity patterns among building elements. However, it is extremely challenging to recover such patterns from noisy and incomplete raw acquisitions such as MVS or 3D scan data. In this paper, we present an algorithm to discover element similarities by directly analysing the given raw acquisitions.

We focus on detecting two types of element similarities that are common in urban scenes. Given a common base geometry and a structure-preserving deformation model, some elements are derived via identical deformation parameters and thus exhibit *full similarity*. Some elements, on the other hand, share only a subset of the deformation parameters and thus exhibit *partial similarity*. Such elements demonstrate structured variations of the base geometry, for example, windows with identical arch but varying height. In order to capture this general notion of similarity, we propose to utilize a set of template models of common element types. Each template is equipped with a deformation model that defines a rich set of variations of its base geometry. Given such a set of templates, we abstract element similarities by similarities in template deformations.

Robust template matching and fitting is necessary to reliably detect element similarities. However, noise and outliers in raw reconstructions unfortunately prohibit this. One way to improve template fitting is to propagate information across similar elements. However, neither the template deformations nor element similarities are known upfront. Hence, we propose an iterative algorithm to

*Both the authors contributed equally.

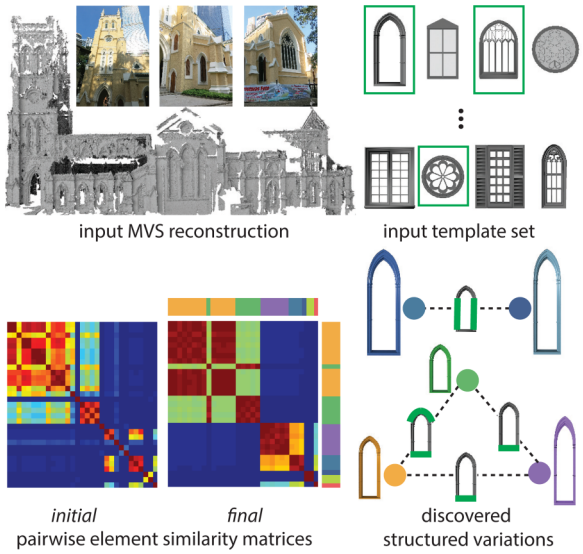


**Figure 1:** *(Dataset 1) Given a 3D building acquisition, we detect element similarities using a set of deformable templates. From an initial pairwise element similarity matrix, we optimize to reveal element clusters as shown in the final similarity matrix. We show the selected templates (in blue) and the similarities detected across the instances of these templates matching to each element cluster (in green in the graph). For example, the blue window instances have equal height but varying width and arch.*

simultaneously perform template matching and fitting across multiple elements. For each template, we discover the subset of its parameters that are similar across multiple elements matching to this template. We repeat the template deformation across these elements by enforcing such parameters to stay similar. This way of coupling the template deformation across multiple elements progressively improves the robustness of template fitting. In addition, such coupled parameters reveal both full and partial element similarities.

### 1.1. Contributions

Our key contribution is to extract similarity patterns among a set of input elements by utilizing deformable templates. Such patterns not only reveal replicated elements, but also expose structured variations among elements that are derived from the same base geometry. Central to our analysis is the coupling of template fitting across multiple elements via the extracted deformation parameters. In contrast to prior work which focuses on individual template fitting, this coupling enables our algorithm to handle challenging datasets with significant amount of noise and missing data.

## 2. Related Work

### 2.1. Urban reconstruction

A vast body of research exists to digitize urban scenes. Procedural modelling is an efficient way to create high-quality models [VAW*10], but obtaining a suitable generative procedure to create a target shape is challenging. Image-based modelling [Qua10, WKM15] and LiDAR scanning [MWA*13] are among the most common approaches for reconstruction of existing architecture. However, the output of such approaches is often incomplete and noisy, and rarely exposes the structure of the original models. We propose to analyse such raw output to reveal high-level structures by utilizing deformable templates.

### 2.2. Template-based reconstruction

Buildings are often constructed from similar elements due to economic and style considerations. In an early attempt, Dick *et al.* [DTRC01] use a set of parameterized part templates to create 3D models from single images. Schindler *et al.* [Sch03] fit segmented image measurements to a set of pre-defined shape templates to create Computer-aided-design (CAD)-like 3D facade reconstruction; Pauly *et al.* [PMG*05] warp selected models from a database of 3D shapes and combine suitable parts towards object completion; Chen *et al.* [CKX*08] propose an interactive setup to lift freehand sketches to 2.5D using a database of architectural elements. The GlobFit system [LWC*11] proposes a primitive-based analyse and reconstruct setting, where arrangements among primitives are discovered. Lafarge *et al.* [LKBH13] has demonstrated a primitive-based hybrid MVS reconstruction approach for large scale models. Typical man-made objects have also been used for indoor scene understanding [KMYG12, NXS12]. Bao *et al.* [BCLS13] use anchor points to deform and fit shape templates to poor quality scans. Kurz *et al.* [KWW*14] propose a template deformation approach that preserves symmetry properties of templates while fitting a scanned object.

All such template-based approaches either assume that exact shape templates are available, or only allow limited deviations from the templates. Furthermore, multiple templates are fitted *independently* across the scene making it challenging to ensure robustness for partial inputs or a limited template database. Our key contribution is to perform template fitting across multiple elements simultaneously while coupling the deformation parameters of templates detected as similar. This analysis not only enables to handle noisy scans but also helps to understand structured variations based on the correlation among the deformation parameters.

### 2.3. Symmetry analysis

Symmetry is ubiquitous in man-made environments and finding symmetries in geometric data has received significant attention. Transformation-space voting [MGP06, PMW*08] and spectral analysis [LCDF10] are among the common approaches proposed (see the survey by Mitra *et al.* [MPWC13]). The method of Pauly *et al.* [PMW*08] is only applicable for detecting regular repetitions whereas our method does not make any assumption about the spatial arrangement of repeated structures. The method of Mitra *et al.* [MGP06] and Lipman *et al.* [LCDF10], on the other hand, focus on detection of exact/approximate symmetries. In contrast, our goal is to detect structured variations between input elements by discovering partial similarities across deformed templates. This is a problem that has not been addressed by any of the previous methods.

Symmetry results in redundant measurements and thus has been effectively exploited in the context of urban modelling to consolidate and improve noisy reconstructions [ZSW*10, JTC11, LZS*11, WFP11, CML*12]. Most of the proposed approaches, however, focus on detection of replicated elements, often arranged as regular grids. In contrast, we focus on detection of full and partial element similarities with no assumption on their spatial arrangement.

### 2.4. Pattern detection

A common practice for detecting patterns is to employ the input with a set of descriptors. Leifman *et al.* [LT13] segment a given surface into pattern and non-pattern vertices using a combination of *point feature* and *curvature* histograms. Liu *et al.* [LMLR07] detect periodic reliefs on triangle meshes based on the auto-correlation of curvatures of the boundary points. Shechtman *et al.* [SI07] present *local self-similarity* descriptors to match images based on self-similarity of colour, edges and repetition patterns. In case of noisy and incomplete data, however, it is challenging to detect reliable descriptors. Hence, we propose an iterative approach where element similarities are abstracted by patterns in template deformations.

### 2.5. Co-analysis

Reconciling observations from multiple instances of data to extract reliable information is common in the literature. Learned-Miller [LM06] jointly aligns a set of images in a process called *congealing*. An affine transformation for each image in a stack is computed to minimize the variance for each pixel location in the stack. Faktor *et al.* [FI13] co-segment an object of interest in a given set of images by aggregating information from corresponding image patches. In the context of visual element discovery, Doersch *et al.* [DSG*12] use a discriminative clustering approach to detect distinctive image patches from a large set of geotagged imagery. Similarly, we present a simultaneous analysis to detect deformation patterns among a set of elements. We aggregate observations from multiple deformable templates to extract reliable similarity patterns.

## 3. Overview

We present a template matching and deformation algorithm to understand structured variations between building elements. We use a set of template models of common element types such as windows. Each such template is characterized by a set of deformation parameters that define its structured variations (see Figure 2). Our algorithm operates on a 3D reconstruction (MVS or scan) of a building where an *element*, for example, a window, constitutes the subset of the input reconstruction falling inside its bounding box. For each of these elements, we identify the best matching template and compute the best fitting deformation of this template, which we call a *template instance*. A key feature of the proposed algorithm is to reveal geometric similarities among the elements by detecting patterns in the deformation parameters of their matching template instances.

Typically, we observe two types of relations between given elements. First, elements that are replicas of the same geometry should be matched to the same instance of the same template. Second, elements that exhibit variations of a base geometry, for example, windows with the same arch but varying height, should be matched to different instances of the same template. In this case, deformation parameters defining the instances will be partially the same. An intuitive approach for detecting such relations is to compute the matching template instance for each element independently and then detect patterns across the resulting deformation parameters. Assume a template is parameterized with $k$ deformation parameters. Each instance of this template is represented as a point in a $k$-dimensional deformation space with nearby points representing instances with similar deformation parameters. In the ideal case, elements with the same geometry will map to a single point in this space. Elements that are variations of a base geometry, however, will map to nearby points (see Figure 3a). This is revealed in similarity matrices computed based on pairwise element distances in

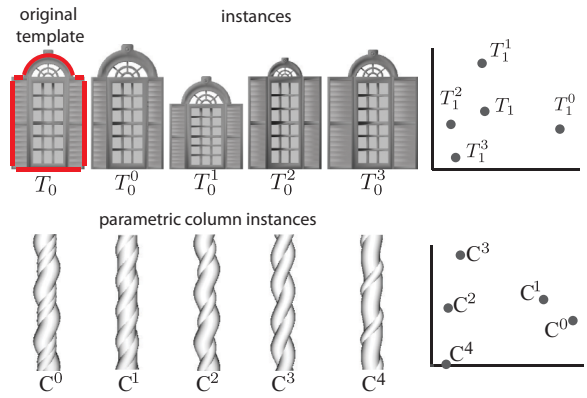


**Figure 2:** *For the template $T$ equipped with the i-Wires deformation model, we illustrate various instances ($T^0, \ldots, T^3$) with different parameters of the detected feature wires (shown in red). We also show several column instances generated by a parametric model. Each instance is visualized as a point in the corresponding deformation space using multi-dimensional scaling projection.*

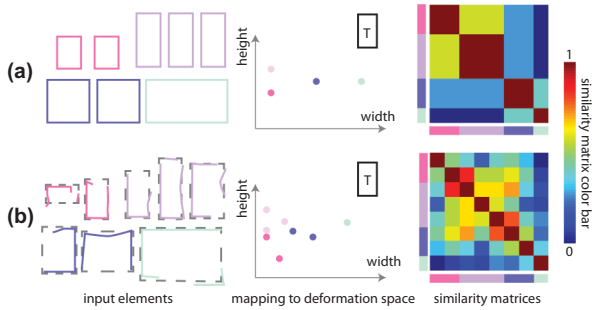


**Figure 3:** *(a) In case of perfect input data, elements with the same geometry are mapped to a single point in the 2D deformation space of a rectangular template $T$. Elements with partial similarities, that is, same width or height, form loose clusters (shown as dotted ellipses). (b) The presence of noise and missing data avoids observing clear clusters in the deformation space. Similarity matrices computed using the pairwise element distances in the deformation space reveal this behaviour. The bars on the left and bottom of the matrices identify the elements.*

the deformation space (see Figure 3, right). Elements that are exact replicas are represented as red blocks whereas elements with partial similarities are represented by colours closer to red.

In case of real data, however, due to noise and partial data even elements that are exact replicas are often matched to different instances of the same template (see Figure 3b) or, even worse, different templates. Thus, similar elements map to scattered points in the template deformation space, avoiding to observe clear clusters (see Figure 9). To address this issue, we propose to analyse the input elements simultaneously (see Figure 4). We begin by deforming a set of templates to fit the given elements. We combine observations from template deformations to map each element to a common subspace representation. Intuitively, similar elements are expected to map to nearby points in this subspace resulting in small pairwise element distances. Using these distances as constraints, we consistently label each element with a deformed template instance. For each template, we discover the subset of its parameters that are similar across elements matching to different instances of the template. We repeat template deformation by coupling these parameters, that is, enforcing them to stay similar. Iterating between these steps progressively brings similar elements closer in the common subspace and reveals clear clusters (see Figure 9). Being independent of the specific choice of the deformation model, this iterative analysis robustly identifies which elements are replicas and which share partial similarities.

## 4. Simultaneous Template Matching and Deformation

We propose to discover similarity patterns among a set of building elements by abstracting the similarities via template deformations. Our analysis is independent of the chosen deformation model (see Section 5 for a collection of the deformation models we used in our experiments). We now describe each stage of our analysis in detail.

### 4.1. Subspace analysis

Given a set of elements $\mathcal{S} := \{s_i\}$ and a set of templates $\mathcal{T} := \{T_j\}$, our goal is to label each $s_i$ with the tuple $(T^i, \mathbf{d}^i)$ where $T^i$ is the best matching template and $\mathbf{d}^i$ is the deformation parameters of the best fitting instance of this template. (In the rest of the paper, a template instance is specified by a subscript denoting the template index and a superscript denoting the fitted element index.) To achieve this goal, we first deform each template $T_j$ to fit all the elements in $\mathcal{S}$:

$$\min_{\mathbf{d}_j} \sum_{s_i \in \mathcal{S}} E_{fit}(T_j, \mathbf{d}_j^i, s_i) + w_{sim} E_{sim}. \tag{1}$$

$\mathbf{d}_j^i$ represents the deformation parameters of $T_j^i$, that is, the instance of $T_j$ that best fits $s_i$. $\mathbf{d}_j$ is a vector of deformation parameters constructed by concatenating $\mathbf{d}_j^i$ for each element $s_i$. The first term measures how well the template fits each element individually while the second term minimizes the difference between the deformation parameters of $T_j$ detected as being similar across multiple elements. $E_{fit}(T_j, \mathbf{d}_j^i, s_i)$ is defined based on the chosen deformation model and we refer the reader to the supplementary material for details. $E_{sim}$ initially evaluates to 0 since we assume no prior knowledge about element similarities.

The computed template deformations provide observations about the element geometries. We represent such observations in a multi-layer graph $M$. Each individual layer is a fully connected graph $G_j = (S, E_j, W_j)$ and encodes the deformation parameters obtained by fitting the template $T_j$ to each element (see Figure 4). Elements are represented as nodes and the edges $E_j$ between the nodes are weighted. An edge $e_{ik} \in E_j$ connecting the nodes $s_i$ and $s_k$ is weighted by

$$w_{ik}(\in W_j) = e^{-D(\mathbf{d}_j^i, \mathbf{d}_j^k)/m_D}. \tag{2}$$

$D(\mathbf{d}_j^i, \mathbf{d}_j^k)$ measures the Euclidean distance between the deformation parameters of the instances $T_j^i$ and $T_j^k$. $m_D$ is the maximum of such distances and is used for normalization.

Each graph layer in $M$ captures different observations of the element geometries obtained by the corresponding template deformations. Our goal is to combine the information from each layer of $M$ to extract a set of consistent relations among the elements. We achieve this goal by adopting the subspace analysis approach of Dong *et al.* [DFVN14]. This method first computes a subspace representation of each graph layer using the corresponding graph Laplacian. Multiple subspaces are then combined into a single representative subspace $U$ by constructing a common graph Laplacian (see the supplementary material). $U$ summarizes the information captured in each graph layer by mapping each element to a low dimensional form. In a subsequent step, we solve a labelling problem by incorporating the relations captured in $U$ as constraints.

### 4.2. Labelling problem

Each element $s_i$ is mapped to the common low-dimensional space $U$ as $s_i'$. Elements with geometric similarities map to nearby points in $U$. Thus, pairwise element distances in this space (see Figure 4) provide an indication of their degree of similarity. Our goal is to label elements that have small pairwise distances with similar instances of a template.

Recall that we want to label each element $s_i$ with a tuple $(T^i, \mathbf{d}^i)$ consisting of both the matching template and the deformation parameters defining the matching instance. Even though we have a set of discrete labels with respect to the template type, the deformation parameters are defined in a continuous deformation space. It is possible to discretize each deformation space by sparsely sampling it. However, it is challenging to obtain a good coverage of the large deformation space with a sparse set of samples while ensuring the samples are sufficiently close to the instances matching the given elements. Yet, the deformation parameters obtained by fitting templates to the input elements provide a good set of initial labels. Therefore, we formulate this labelling problem as a Markov Random Field (MRF) optimization [DOIB10] where the label set $\mathcal{L} = \{(T_j, \mathbf{d}_j^i)\}$ consists of the current set of template instances obtained by fitting each template $T_j$ to each element $s_i$. (We choose the best three template instance for each element in our experiments.) The MRF optimization consists of data, smoothness and label costs:

$$\sum_{s_i} E_d(s_i, L_i) + w_s \sum_{s_i, s_k} \alpha_{ik} E_s(s_i, s_k, L_i, L_k) + \sum_{T_j} \lambda_{T_j} E_L. \tag{3}$$

The data term, $E_d(s_i, L_i)$, measures the cost of fitting the template instance defined by the label $L_i$ to the element $s_i$. It is defined as the average distance between the closest point correspondences

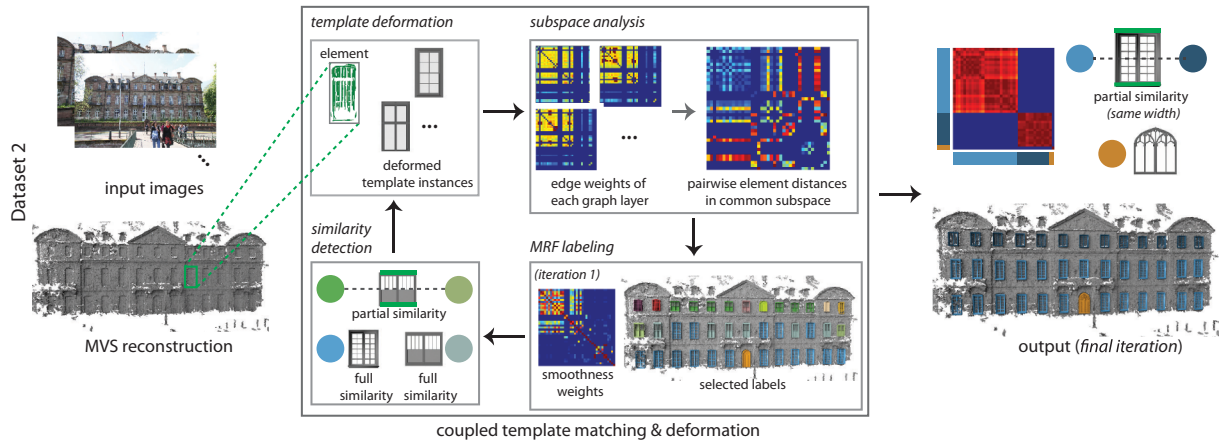


**Figure 4:** *Given a 3D acquisition (e.g. MVS) of a building, we utilize a set of deformable templates to match its elements, that is, windows. We combine observations from multiple template deformations via a subspace analysis to extract relations among the elements. Using these relations as constraints, we label each element with a deformed template instance (same instances are denoted in same colour). We repeat template deformation by consolidating observations across elements matched to similar template instances. Performing this analysis iteratively reveals which elements are replicas of the same geometry (represented as red blocks in smoothness weight matrices) or share partial similarities (highlighted in green on the matching templates).*

established between the deformed template instance and $s_i$. The smoothness term, $E_s(s_i, s_k, L_i, L_k)$, evaluates the consistency of the labelling of each pair of elements. If two labels belong to the same template, this term measures the Euclidean distance between the deformation parameters of the two instances of the template. Otherwise, a fixed smoothness cost (two times the maximum distance between any two instances of the same template in our experiments) is assigned. The smoothness weights

$$\alpha_{ik} = e^{-\|s'_i - s'_k\|^2/\sigma^2} \tag{4}$$

are determined from the distances between the elements mapped to $U$. Intuitively, similar elements have small pairwise distance in $U$ resulting in high smoothness weights (revealed as colours closer to red in pairwise smoothness weight matrices shown in Figures 4 and 11). These elements are expected to be assigned to similar labels, that is, instances of the same template. The constant $\sigma > 0$ determines how rapidly the smoothness weight drops with increasing pairwise element distances and is set to 0.1 in our experiments. The last term in Equation (3) penalizes each unique template that appears in the final labelling. Specifically, we group the candidate labels coming from the same templates into subsets and a fixed label cost $E_L$ (equal to $1/5^{th}$ of the average data cost in our experiments) is induced if at least one label is used from such a subset. The indicator variable $\lambda_{T_j}$ is set to 1 if an instance of the template $T_j$ appears in the final labelling.

Due to noise and partial data, replicated elements often map to scattered points in the representative subspace $U$. The smoothness term progressively enforces these elements to be assigned to the same label and thus brings them closer. This leads to the formation of red blocks in the pairwise smoothness matrices in the final iterations of our algorithm (see Figure 11). $w_s$ determines the relative weight of this term can be adapted based on the input data quality (see Section 6). If there are similar templates, similar elements might get assigned to instances of different templates. The label cost favours the use of as few unique templates as possible and thus enables a consistent labelling. Both smoothness and label costs enforce the selection of fewer templates. Hence, elements that exhibit variations of a base geometry are matched to different instances of the same template (see Table 1).

**Table 1:** *The table shows the number of input images ($N_i$), the number of user selected elements ($N_s$), the total number of detected elements ($N_e$), the numbers of templates selected by the independent analysis ($T_d$) and with the coupled analysis ($T_c$) and the total number of template instances discovered ($T_i$). Note that for Dataset 8 we use a parametric model considered as a single template.*

| | $N_i$ | $N_s$ | $N_e$ | $T_d$ | $T_c$ | $T_i$ |
|---|---|---|---|---|---|---|
| Dataset 1 | 120 | 7 | 32 | 8 | 3 | 7 |
| Dataset 2 | 60 | 3 | 39 | 10 | 2 | 3 |
| Dataset 3 | 160 | 8 | 32 | 12 | 4 | 10 |
| Dataset 4 | 299 | 10 | 99 | 22 | 7 | 9 |
| Dataset 5 | 70 | 13 | 25 | 7 | 5 | 9 |
| Dataset 6 | 129 | 6 | 57 | 13 | 3 | 4 |
| Dataset 7 | 126 | 7 | 17 | 7 | 4 | 8 |
| Dataset 8 | – | 36 | 36 | – | – | 10 |

### 4.3. Similarity detection

Once each element is labelled with a matching template instance, we evaluate the labels to extract a set of similarity relations $R = \{r\}$ among the elements. In particular, if elements $s_i$ and $s_k$ are matched to two instances of template $T_j$, we define the relation $r = (s_i, s_k, \mathbf{c}_j, T_j)$. $\mathbf{c}_j$ is a binary vector of size equal to the number of deformation parameters of $T_j$. It contains a 1 for the deformation parameters of the matching template instances that have a

Euclidean distance below a certain threshold (0.5% of the length of the diagonal of the bounding box of the input reconstruction). Such *coupled parameters* indicate partial similarities between $s_i$ and $s_k$ with respect to template $T_j$. In subsequent iterations of our algorithm, we enforce the coupled parameters to stay similar during template deformation. A vector $\mathbf{c}_j$ consisting of all ones ($\mathbf{c}_j = \mathbf{1}$) indicates that $s_i$ and $s_k$ are exact replicas. In this case, we define an additional relation for these elements for all other templates with $\mathbf{c}_j = \mathbf{1}$.

Once we extract a set of relations, $R$, we repeat template deformation using these relations as additional constraints. When deforming a template $T_j$ to fit the elements in $\mathcal{S}$, we minimize the energy given in Equation (1) with

$$E_{sim} = \sum_{(r)\in R} E_{dist}(r). \tag{5}$$

$E_{dist}(r) = \mathbf{c}_j^T(\mathbf{d}_j^i - \mathbf{d}_j^k)^2$ for $r = (s_i, s_k, \mathbf{c}_j, T_j)$ measures the difference between the coupled deformation parameters of the instances of $T_j$ that fit $s_i$ and $s_k$, $w_{sim}$ (set to 10.0 in all our evaluations) defines the weight of this term. With the new deformation parameters, we update the multi-layer graph $M$ and repeat the subspace analysis and the MRF optimization.

We iterate between these steps until no change is observed in the final labelling (typically five to six iterations). Intuitively, enforcing the coupled parameters of a template to stay equal when deforming the template to fit multiple elements can be considered as consolidating observations across these elements. This consolidation of observations improves the candidate label set for the subsequent MRF optimization step of our algorithm. Also, potential element clusters in template deformation spaces become more pronounced as similar elements are progressively pulled closer (see Figure 9). In our evaluations, we demonstrate the benefits of such a simultaneous template fitting approach over fitting templates to each element individually (see the supplementary material).

## 5. Template-Based Deformation

Given a deformation model, we have presented a template matching algorithm. We evaluate this algorithm by adopting two deformation models suitable for architectural data (see Figure 2). For element types with dominant feature lines such as windows we adopt the structure-aware i-Wires deformation model [GSMCO09]. We also demonstrate an example of a parametric model on curved columns. We provide a brief description of these deformation models and refer the reader to the supplementary material for more details.

### 5.1. i-Wires deformation model

For each template model, in a pre-processing stage, we extract feature lines, called wires, based on the dihedral angles between the edges (see [GSMCO09]). Each feature wire is a collection of atomic wires that can be one of the *straight line, circular* or *elliptic arc* types. Atomic wires are defined by a set of parameters such as the length and the direction of a straight line, or the centre, radius and the opening angle of a circular arc. Each template is parameterized with the union of the parameters of its atomic wires.

The goal of template deformation is to compute the parameters of the atomic wires of a template that best fit an element. We identify the inter- and intra-wire relations of the template, that is, equal length, orthogonality, planarity and symmetry, and preserve them during deformation [GSMCO09].

### 5.2. Parametric deformation model

We use a parametric deformation model to demonstrate our analysis on element types such as curved columns. Our parametric model is based on a *helical structure*, that is, a circle swept along a 3D helix curve. A variety of columns can be generated by applying CSG operations, that is, union or difference, to a set of helixes. Each helical structure is characterized by the *pitch*, *radius* and *start angle* of its helix and the *radius* of its swept circle.

We consider this parametric model as a single template where each column generated by a different set of parameters is an instance of this template. We first generate a set of candidate column instances by fitting helical structures to each input element (see the supplementary material). Our algorithm then deforms these instances to match the input elements and detects patterns in the resulting deformation parameters.

## 6. Evaluation

**Datasets.** We demonstrate our algorithm both on MVS data (we compute camera parameters using the VisualSFM tool [Wu13] and the dense reconstruction by PMVS [FP09]) and scans acquired with Microsoft Kinect (using the software Skanect [ska]).

**Element selection.** Our algorithm detects similarity patterns among a set of building elements. Even though there exist automatic facade parsing methods exploiting the presence of horizontal and vertical splitting lines and regular grids [MWA*13], we observe that such methods fail to identify the elements of more complex architectural scenes. Instead, we utilize a semi-automatic approach. For MVS reconstructions, we adopt the method of Ceylan *et al.* [CMZP14] which requires the user to roughly mark an element of interest, for example, a window frame, in only *one* of the input images and automatically detects its repetitions. We revert to user input to mark any missing element in case of strong variation or large occlusions. For scan data, we require the user to mark the elements in 3D by defining their bounding boxes. Table 1 provides the number of user marked elements for each of our examples. Please note that although additional user input to cluster similar elements may improve the results, this is not sufficient to detect partial similarities across the clusters. Such similarities are difficult to manually specify as they may not be obvious by visual inspection and require the user to mentally solve the template selection and deformation problems simultaneously. Therefore, once elements are identified, we revert to our automatic analysis with no use of prior information to detect both full and partial element similarities.

**Template set.** In our evaluations, we mainly focus on window elements as they often exhibit full and partial similarities. We use a template set consisting of 60 window models downloaded from the Digimation Model Bank and Trimble 3D Warehouse. For computational efficiency, we deform each pair of templates to fit the

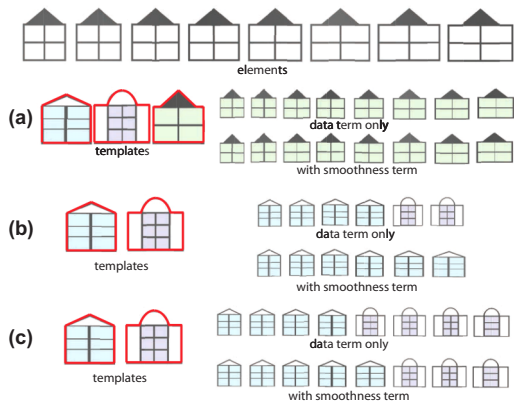


**Figure 5:** *The amount of variation among the elements affects the final choice of template instances. For different sets of templates (with feature wires shown in red) and elements, we show the selected template instances based on data term only and additionally considering the smoothness term.*

other and group them based on their deformation capabilities: *arch, rectangle, triangle-top* and *circular* windows. Given a grouping of the templates, for each element we first identify the best matching group with respect to a similarity transformation and consider the templates only in its matching group in the rest of the analysis.

### 6.1. Performance on synthetic data

We evaluate our algorithm by changing the amount of variation across input elements, the number of utilized templates and the data quality. In order to assess each factor independently, we perform evaluations on synthetic data using a fixed set of parameters.

#### 6.1.1. *Effect of element deformations*

Our algorithm labels each input element with a matching template instance by incorporating data and smoothness terms. While the data term evaluates the individual label assignments, the smoothness term favours similar labels for similar elements. Due to this coupling, the amount of variation among the elements affects the final choice of labels. We illustrate this effect on a set of synthetic elements created by gradually increasing the variation among them (see Figure 5). When the template set includes a template capable of capturing all of these variations, all elements are labelled with different instances of this template (Figure 5a), that is, the smoothness term has no effect. When we remove this template, however, none of the remaining templates is capable of perfectly capturing the element variations. We first consider the first six elements, where four of them prefer the first template based on the data term only. Even though the fifth and sixth elements prefer the second template based on the data term, the smoothness term enforces them to pick labels from the first template (Figure 5b). We then add two more elements that also prefer the second template individually. This is perceived as a strong indication that the second template is also a likely assignment. Thus the last three elements are now assigned to labels from the second template (Figure 5c).

#### 6.1.2. *Effect of number of templates*

Since templates deform similarly to fit similar elements, each template contributes to detection of element similarities. We illustrate this on a synthetic house model with two types of windows showing variation in height and width (see Figure 6). We run our algorithm using an increasing template set size of 1, 10 (selected templates are shown in the supplementary material) and 60 (organized as four groups). In each case, we show the pairwise element smoothness weights in colour-coded matrices. For each block of identical elements, the side colour bars denote the colour of the corresponding matching template instance. We show the partial similarities detected between such instances in a graph by highlighting the coupled parts of templates in green.

We observe that even a single template is capable of distinguishing the variation among the elements resulting in the selection of four distinct template instances (Figure 6a). With additional templates, the two window types are identified resulting in the selection of two instances of each template (Figure 6b). When using a grouping

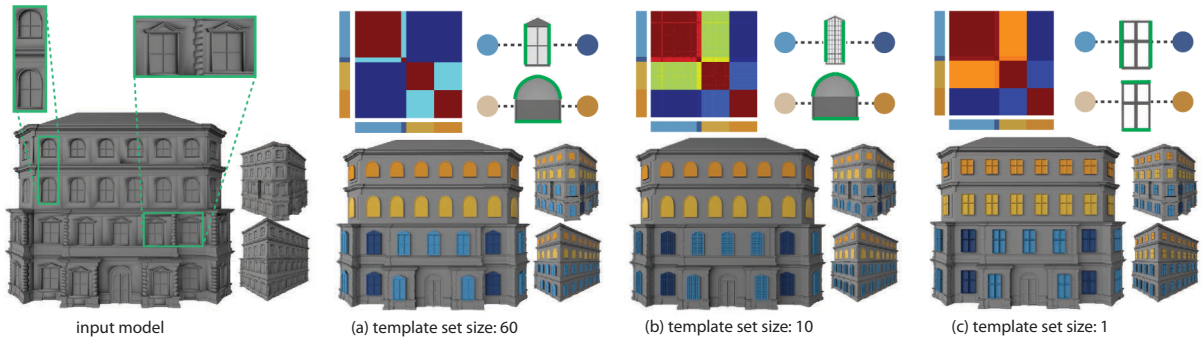


**Figure 6:** *We show the selected template instances for a synthetic house model (consisting of 38 narrow triangle-top, 4 wide triangle-top, 23 long arch and 23 short arch windows) when using different number of templates. For each case, we also show the colour-coded smoothness matrices and the partial similarities detected between the elements (highlighted in green). Note that the removal of the triangle-top template selected in (c) results in a selection of another triangle-top window in (b).*

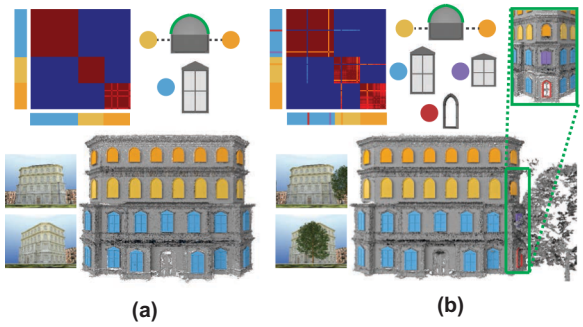


**Figure 7:** *We evaluate our algorithm on MVS reconstructions obtained from rendered images of a synthetic model. Due to loss of fine details, we cannot recover the subtle variation in width of the triangle-top windows in blue (a) and the occlusion by a large tree results in wrong template assignments for some elements (b).*

among the templates (Figure 6c), the two type of window elements are initially matched to different template groups resulting in no smoothness relation between them, that is, blue blocks in the corresponding smoothness matrix. With a single rectangular template, however, the similarity between the height of the triangle-top and long arch windows is reflected as a reasonably high smoothness weight, that is, orange block in the corresponding matrix.

#### 6.1.3. *Effect of data quality*

The input data quality has a direct impact on template deformations. We compare the performance of our algorithm on synthetic data (Figure 6c) and a MVS reconstruction obtained from the rendered images of the model (Figure 7). We observe two main sources of error that potentially influence our results. First, due to the challenges in correspondence search or limited sensor resolution, 3D reconstructions exhibit a *general* degradation in data quality which might lead to failure in capturing fine details. Even though our algorithm selects the same templates as in the ground truth case, it fails to capture the subtle variation in the width of the two instances of triangle-top windows (Figure 7a). Second, factors such as large occlusions result in *local* degradation in data quality. Thus, when we place a large tree model in front of the house, we cannot recover the correct template assignments for the windows occluded by this tree (Figure 7b).

### 6.2. Performance on real data

We evaluate our algorithm on various real datasets with different style and varying complexity (Figures 1, 4, 11 and 8). Table 1 shows the statistics of our algorithm on each dataset. We summarize our main findings and refer the reader to the supplementary material for a more complete set of results.

#### 6.2.1. *Performance on MVS output*

In our evaluations, we mainly focus on challenging MVS output that suffers from significant amount of noise and missing data.

Our algorithm discovers both replicated elements and partial element similarities. Figures 1 and 11 illustrate many such similarities detected which would have been difficult to capture otherwise. For example, for *Dataset 1*, our algorithm discovers five instances of the same template for 30 window elements (Figure 1) whereas template fitting for each element individually results in the selection of five different templates (Figure 9).

Our analysis makes no assumption about the presence of any specific type of spatial arrangement such as 2D grids. Yet, we can successfully detect similarities between elements that are rotationally symmetric (Figure 11, *Dataset* 7), arranged as 1D grids (Figure 11, *Dataset* 6) and located across different facades of a building (Figure 11, *Dataset* 5). Even though noise in MVS reconstructions makes it difficult to initialize transform domain grid fitting as proposed by Pauly *et al.* [PMW*08], template instances detected by our algorithm enables the discovery of grid-like spatial arrangements between the elements. We use such relations to spatially snap the template instances in our results.

#### 6.2.2. *Performance on scan data*

We also evaluate our algorithm on a scan of an indoor scene containing curved columns (see Figure 8) using a parametric deformation model. This deformation model captures the properties of the individual helical structures each column is composed of. Starting from a candidate set of columns fitted individually to each element, we discover similarities across these individual helical structures by our simultaneous template deformation framework: identical, reflected (i.e. with opposite rotation direction) and sharing the same pitch angle only. We also apply our algorithm to analyse the window elements in the MVS reconstruction of the same scene (obtained from 400 images) and detect that they are identical.

To evaluate the robustness of our algorithm, we have synthetically added noise to this dataset and re-performed our analysis (see the supplementary material). Even though some of the fine details are not captured due to noise, our analysis is stable and recovers the expected similarity patterns.

#### 6.2.3. *Comparison to naive clustering*

Due to noise and partial data, an independent analysis of each element often results in the selection of different templates for elements that are derived from the same base geometry (Figure 9a, left). Even if we annotate the correct template selection, replicated elements are mapped to scattered points in the template deformation space. Thus, standard clustering algorithms such as *k-means* fail to identify the correct element clusters (Figure 9b, left). Our iterative analysis, however, progressively consolidates observations across similar elements and reveals distinct element clusters (Figure 9b, right). For this dataset, compared to ground truth clustering by visual inspection, the clustering of our algorithm achieves a mutual information score [VEB09] of 0.876 with one mislabelled element whereas the clustering based on independent analysis has a score of 0.468.

#### 6.2.4. *Comparison to Kurz et al. [KWW*14]*

For the dataset shown in Figure 9, we also evaluate the method of Kurz *et al.* [KWW*14]. This method presents a template

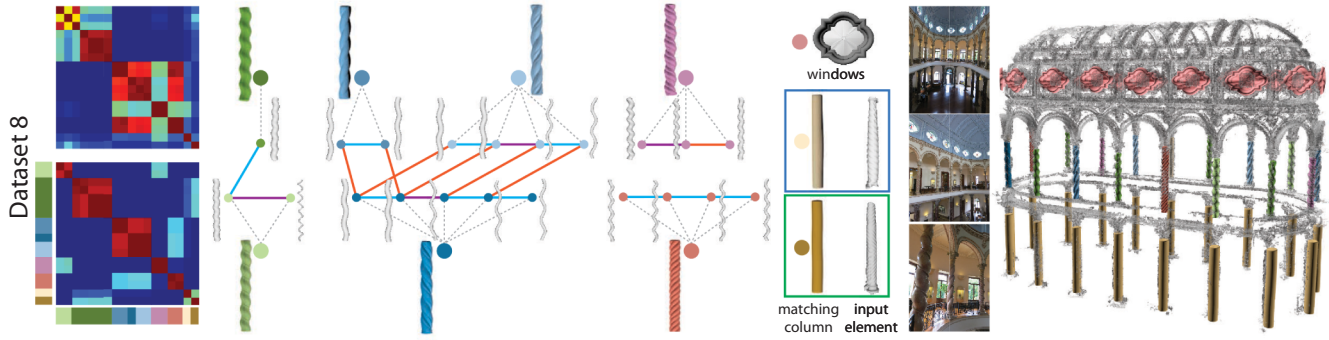


**Figure 8:** *We use a parametric deformation model to match the helical columns in a 3D scan of a museum. We show the smoothness matrices in the first (top-left) and final (bottom-left) iterations of our algorithm. We omit the elements in the bottom floor, which have been detected as identical, from these matrices for visualization purposes. The side colour bars denote the colour of the matching column instance of the corresponding block of identical elements. Each column instance is composed of a number of individual helical structures that we show in gray (e.g. the red instance is composed of four helical structures). We show the similarities detected across these substructures in solid edges: identical (blue), reflected (orange), same pitch only (purple).*

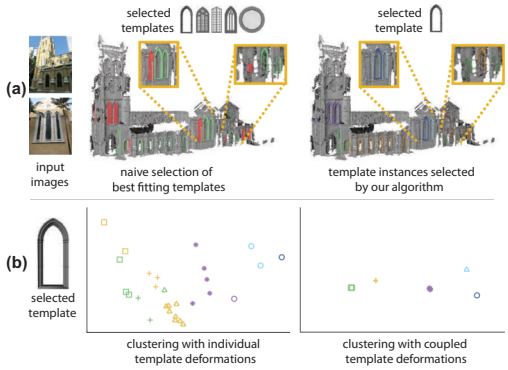


**Figure 9:** *(a) Individual template fitting for a set of elements results in the selection of five different templates whereas our algorithm assigns the elements to five different instances of the same template. (b) Given a template selection, we visualize each element in the low-dimensional deformation space of the template (replicated elements are shown in same colour) using the deformation parameters obtained by individual fitting versus our algorithm. The clusters generated by the k-means ($k = 5$) algorithm are indicated by different symbols. Note how clusters on the left span different template instances and lead to misclassification.*

deformation model that explicitly considers the symmetric features of the templates. Even with such an advanced deformation model, independent template fitting for each element maps the replicated elements to scattered points (Figure 10, left) whereas our analysis successfully produces tight clusters (Figure 10, right). Please note that the method of Kurz *et al.* is complementary to our analysis since it can be used as the input template deformation model.

### 6.2.5. *Effect of parameters*

Our analysis involves a small set of parameters that are listed in Table 2. For most of these parameters, we use the default values introduced in Section 4 in all of our evaluations. The only parameter that requires adjustment is $w_s$, the weight of the smoothness term involved in the MRF optimization. An inherent challenge in analysing raw data measurements as obtained from MVS or scanning is to distinguish noise from fine details. Our algorithm solves a labelling problem consisting of data and smoothness terms to reflect this tradeoff. A lower $w_s$ helps to capture high frequency details in case of reliable data. In case of noisy and partial data, for example, MVS data, however, the data term becomes unreliable and a higher $w_s$ allows consolidating observations across multiple elements. In

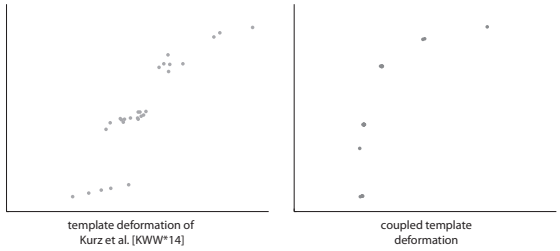


**Figure 10:** *Given the correct template selection, the template deformation model of Kurz et al. [KWW*14] maps replicated elements to scattered points as a result of individual fitting. Our simultaneous analysis, on the other hand, forms tight clusters. The method of Kurz et al. provides a free-form deformation. Thus, we parameterize the deformed templates by the width and height of their bounding boxes.*

**Table 2:** *The table shows the parameters involved in our analysis and their values used in our evaluations.*

| | MVS data | Scan data |
|---|---|---|
| $w_{sim}$ (Equation 1) | 10 | 10 |
| $E_L$ (Equation 3) | 1/5th of avg. data cost | 1/5th of avg. data cost |
| $w_s$ (Equation 3) | 0.1 | 0 |
| $\sigma$ (Equation 4) | 0.1 | 0.1 |

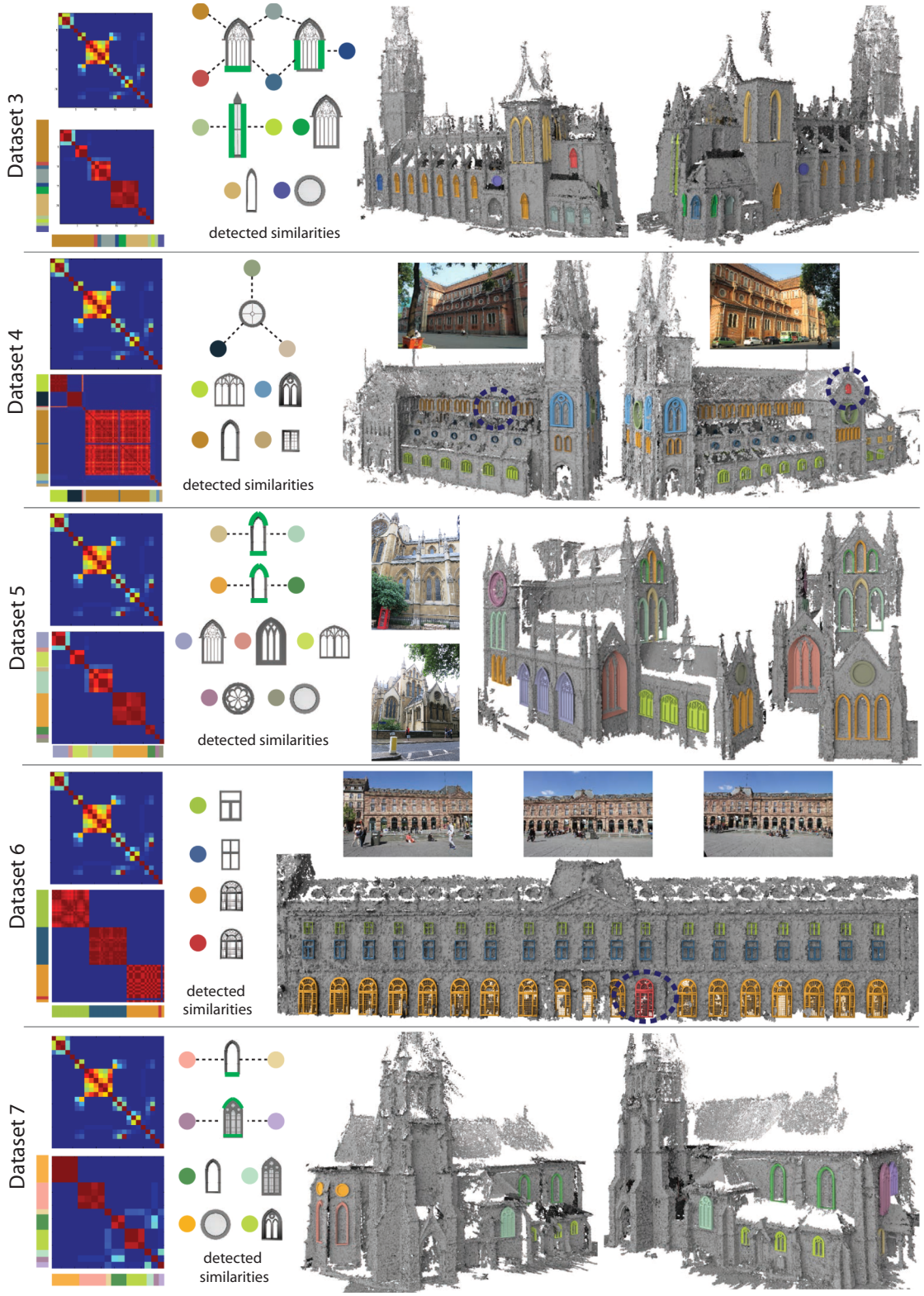


**Figure 11:** *For each data set, we show the smoothness matrices in the first (top left) and final (bottom left) iterations of our algorithm. Colour bars at the sides of the matrices denote the colour of the matching template instances of the corresponding block of identical elements. Partial similarities detected between different element blocks are shown on the corresponding templates in green. We denote the elements matched to wrong template instances with dotted circles. Please refer to Table 1 and supplementary material for details.*

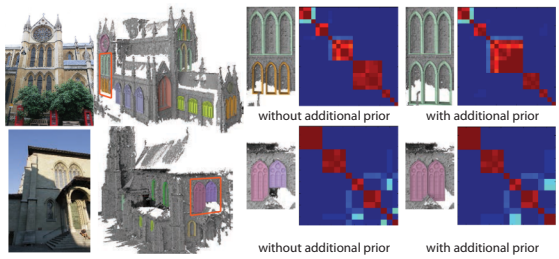


**Figure 12:** *Our algorithm fails to match the windows indicated in orange to the correct template instances due to large occlusions. It is possible to augment our analysis with additional priors, for example, incorporating smoothness constraints among elements arranged in a grid, to resolve such failures. We show the element smoothness matrices with and without use of such priors.*

our evaluations, we set $w_s = 0.1$ for all the MVS examples. We disable the smoothness term for the scan data since the data quality is reliable and there are subtle variations across the elements we would like to capture (e.g. the dark and light green columns in Figure 8).

#### 6.2.6. *Performance evaluation*

We run our experiments on a 2.8 GHz Intel Core i7 machine. Our analysis is iterative where each iteration begins with the *template deformation* stage. We group the template models based on their deformation capabilities and identify the best matching group of each input element. For each element, we deform the templates only in its matching group. Given $n$ input elements and $k$ templates in each template group in average (15 in our evaluations), we perform $O(nk)$ template deformations. We then map the observations from the deformations to a common space using the *subspace analysis*. This step is almost instantaneous and has negligible time complexity (700 ms in average). We then perform the *labelling optimization* to label each input element with a deformed template instance. We choose the best three fitting template instances for each element to construct a label set of $3n$ labels. Given $n$ elements and $3n$ labels, this step takes 80 s in average. Finally, the *similarity detection* step extracts the coupled template parameters across the selected template instances. This step also has negligible time complexity (80 ms in average). For all of our datasets, our analysis converges in five to six iterations. The computational complexity is dominated by the template deformation step where the time spent for each deformation depends on the choice of the deformation model. Please note that, this step could easily be parallelized by deforming each template to the input elements in parallel.

#### 6.2.7. *Limitations*

Due to limited sensor resolution, 3D reconstructions often fail to capture fine details, for example, in the substructures of the elements. Such missing details or the lack of a more suitable template might result in selection of a template different than a user-intended one. For *Dataset 4*, the closest template has been selected to capture the two-arch structure of the windows shown in green (see Figure 11). For *Dataset 8*, we have failed to capture the subtle details in the column shown by the blue rectangle and our parametric model is not capable of generating the details on the column shown by the green rectangle (see Figure 8).

Severe local degradations in data quality, for example, due to large occlusions, prevent reliable template deformation and is another source of failure for our algorithm. In our examples, we indicate such failures by dotted ellipses (Figures 1 and 11). In Figure 12, we demonstrate two challenging cases, where almost half of the indicated windows are occluded. Even though our algorithm identifies the correct template, it discovers the wrong (shorter) instance.

## 7. Conclusion

We presented an algorithm to discover similarity patterns among a set of elements by using deformable template models. Our template matching and deformation analysis identifies the best fitting template instance of each element and detects patterns in the deformation modes of these instances. Even though it is possible to incorporate additional priors, we assumed no additional information to demonstrate the effectiveness of the approach. Our approach is independent of the choice of the deformation model. Thus, it can be adopted to other problem settings by defining other context-specific templates.

In our evaluations, we utilized a simple template grouping strategy. Considering a more sophisticated organization, for example, a hierarchical grouping, is an interesting future direction. Exploring additional priors, for example, style-annotated templates, both in template organization and deformation can aid tasks such as discovering similarities across different buildings.

Accompanying our algorithm with a deformation model that supports discrete parameters will enable to capture discrete changes among elements, for example, in their substructures.

We have adopted a semi-automatic approach to identify the elements of a building. Learning common element deformations to enable automatic detection of building elements is an interesting future direction.

## Acknowledgements

We thank the anonymous reviewers for their valuable comments. We also thank Michael Wand and Xiaokun Wu for helping us with comparisons to their previous work. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013)/ERC Grant Agreement n. 257453: COSYM and ERC Starting Grant SmartGeometry (StG-2013-335373).

## References

[BCLS13] Bao S. Y., Chandraker M., Lin Y., Savarese S.: Dense object reconstruction with semantic priors. In *IEEE CVPR* (Portland, OR, USA, 2013), IEEE Computer Society, Los Alamitos, CA, USA.

[CKX*08] Chen X., Kang S. B., Xu Y.-Q., Dorsey J., Shum H.-Y.: Sketching reality: Realistic interpretation of architectural

designs. *ACM Transactions on Graphics 27* (2008), 11:1–11:15.

[CML*12] Ceylan D., Mitra N. J., Li H., Weise T., Pauly M.: Factored facade acquisition using symmetric line arrangements. *Computer Graphics Forum (Eurographics) 31* (May 2012), 671–680.

[CMZP14] Ceylan D., Mitra N. J., Zheng Y., Pauly M.: Coupled structure-from-motion and 3d symmetry detection for urban facades. *ACM Transactions on Graphics 33*, 1(Feb. 2014), 2:1–2:15.

[DFVN14] Dong X., Frossard P., Vandergheynst P., Nefedov N.: Clustering on multi-layer graphs via subspace analysis on Gassmann manifolds. *IEEE Transactions on Signal Processing 62* (Feb. 2014), 905–918.

[DOIB10] Delong A., Osokin A., Isack H., Boykov Y.: Fast approximate energy minimization with label costs. In *IEEE CVPR* (San Francisco, CA, USA, June 2010), IEEE Computer Society, Los Alamitos, CA, USA, pp. 2173–2180.

[DSG*12] Doersch C., Singh S., Gupta A., Sivic J., Efros A. A.: What makes paris look like paris? *ACM Transactions on Graphics (SIGGRAPH) 31*, 4(2012), 103–110.

[DTRC01] Dick A. R., Torr P. H. S., Ruffle S. J., Cipolla R.: Combining single view recognition and multiple view stereo for architectural scenes. In *IEEE International Conference on Computer Vision* (Vancouver, British Columbia, Canada, 2001), IEEE, IEEE Computer Society, Los Alamitos, CA, USA.

[FI13] Faktor A., Irani M.: Co-segmentation by composition. *IEEE International Conference on Computer Vision* (Sydney, Australia, 2013), IEEE, IEEE Computer Society, Los Alamitos, CA, USA, pp. 1297–1304.

[FP09] Furukawa Y., Ponce J., : Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32* (2009), 1362–1376.

[GSMCO09] Gal R., Sorkine O., Mitra N. J., Cohen-Or D.: iWIRES: An analyze-and-edit approach to shape manipulation. *ACM Transactions on Graphics (SIGGRAPH) 28* (2009), 33:1–33:10.

[JTC11] Jiang N., Tan P., Cheong L.-F.: Multi-view repetitive structure detection. In *IEEE International Conference on Computer Vision* (Barcelona, Spain, 2011), IEEE, IEEE Computer Society, Los Alamitos, CA, USA, pp. 535–542.

[KMYG12] Kim Y. M., Mitra N. J., Yan D.-M., Guibas L.: Acquiring 3d indoor environments with variability and repetition. *ACM Transactions on Graphics (SIGGRAPH Asia) 31* (Nov. 2012), 138:1–138:11.

[KWW*14] Kurz C., Wu X., Wand M., ThormÃ¤hlen T., Kohli P., Seidel H.-P.: Symmetry-aware template deformation and fitting. *Computer Graphics Forum 33* (2014), 205–219.

[LCDF10] Lipman Y., Chen X., Daubechies I., Funkhouser T.: Symmetry factored embedding and distance. *ACM Transactions on Graphics (SIGGRAPH) 29*, 4(July 2010), 103:1–103:12.

[LKBH13] Lafarge F., Keriven R., Brédif M., Hiep V.: A hybrid multi-view stereo algorithm for modeling urban scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35*, 1(Jan. 2013), 5–17.

[LM06] Learned-Miller E.: Data driven image models through continuous joint alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence 28* (Feb. 2006), 236–250.

[LMLR07] Liu S., Martin R. R., Langbein F. C., Rosin P. L.: Segmenting periodic reliefs on triangle meshes. In *Proceedings of the IMA International Conference on Mathematics of Surfaces* (2007) R. Martin, M. Sabin, J. Winkler (Eds.), Springer-Verlag, Sheffield, UK, pp. 290–306.

[LT13] Leifman G., Tal A.: Pattern-driven colorization of 3d surfaces. In *IEEE CVPR* (Portland, OR, USA, June 2013), IEEE, IEEE Computer Society, Los Alamitos, CA, USA, pp. 241–248.

[LWC*11] Li Y., Wu X., Chrysanthou Y., Sharf A., Cohen-Or D., Mitra N. J.: Globfit: Consistently fitting primitives by discovering global relations. *ACM Transactions on Graphics (SIGGRAPH) 30*, 4(2011), 52:1–52:12.

[LZS*11] Li Y., Zheng Q., Sharf A., Cohen-Or D., Chen B., Mitra N. J.: 2d-3d fusion for layer decomposition of urban facades. In *IEEE ICCV* (Barcelona, Spain, November 2011), IEEE, IEEE Computer Society, Los Alamitos, CA, USA.

[MGP06] Mitra N. J., Guibas L. J., Pauly M.: Partial and approximate symmetry detection for 3d geometry. *ACM Transactions on Graphics (SIGGRAPH) 25* (2006), 560–568.

[MPWC13] Mitra N. J., Pauly M., Wand M., Ceylan D.: Symmetry in 3d geometry: Extraction and applications. *Computer Graphics Forum 32*, 6(2013), 1–23.

[MWA*13] Musialski P., Wonka P., Aliaga D. G., Wimmer M., van Gool L., Purgathofer, W.: A survey of urban reconstruction. *Computer Graphics Forum 32*, 6(2013), 146–177.

[NXS12] Nan L., Xie K., Sharf A.: A search-classify approach for cluttered indoor scene understanding. *ACM Transactions on Graphics 31* (Nov. 2012), 137:1–137:10.

[PMG*05] Pauly M., Mitra N. J., Giesen J., Gross M., Guibas L. J.: Example-based 3D scan completion. *Computer Graphics Forum (SGP)* (2005), 23–32.

[PMW*08] Pauly M., Mitra N. J., Wallner J., Pottmann H., Guibas L., : Discovering structural regularity in 3D geometry. *ACM Transactions on Graphics (SIGGRAPH) 43* (2008), 1–11.

[Qua10] Quan L.: *Image-Based Modeling*, 1st ed. Springer, New York, USA, 2010.

[Sch03] SCHINDLER K.: A model-based method for building reconstruction. In *Proceedings of the International Conference on Computer Vision Working on Higher-Level Knowledge in 3D Modeling & Motion* (Nice, France, 2003), IEEE, IEEE Computer Society, Los Alamitos, CA, USA, pp. 74–82.

[SI07] SHECHTMAN E., IRANI M.: Matching local self-similarities across images and videos. In *IEEE CVPR* (Mineapolis, USA, June 2007), IEEE, IEEE Computer Society, Los Alamitos, CA, USA, pp. 1–8.

[ska] Skanect 3d scanning software. http://skanect.occipital.com/. Accessed: 2015-01-10.

[VAW*10] VANEGAS C. A., ALIAGA D. G., WONKA P., MÜLLER P., WADDELL P., WATSON B.: Modelling the appearance and behaviour of urban spaces. *Computer Graphics Forum 29*, 1 (2010), 25–42.

[VEB09] VINH N. X., EPPS J., BAILEY J.: Information theoretic measures for clusterings comparison: Is a correction for chance necessary? In *ICML* (Montreal, Quebec, Canada, 2009), ACM, New York, USA, pp. 1073–1080.

[WFP11]WU C., FRAHM J.-M., POLLEFEYS M.: Repetition-based dense single-view reconstruction. In *IEEE CVPR* (Providence, RI, USA, 2011), IEEE, IEEE Computer Society, Los Alamitos, CA, USA.

[WKM15] WANG T. Y., KOHLI P., MITRA N. J.: Dynamic SFM: Detecting scene changes from image pairs, *Computer Graphics Forum 34* (2015), 177–189.

[Wu13] WU C.: Towards linear-time incremental structure from motion. In *3DV-Conference* (Seattle, WA, USA, 2013), IEEE, IEEE Computer Society, Los Alamitos, CA, USA, pp. 127–134.

[ZSW*10] ZHENG Q., SHARF A., WAN G., LI Y., MITRA N. J., COHEN-OR D., CHEN B.: Non-local scan consolidation for 3D urban scenes. *ACM Transactions on Graphics (SIGGRAPH) 29* (2010), 94:1–94:9.

## Supporting Information

Additional Supporting Information may be found in the online version of this article at the publisher's web site:

**Movie**

**Template Deformation**

**Subspace Analysis**

**Detailed Results**