Generating and Exploring Good Building Layouts

Fan Bao[1] Dong-Ming Yan[2] Niloy J. Mitra[3] Peter Wonka[1,2]

[1]Arizona State University [2]KAUST [3]University College London

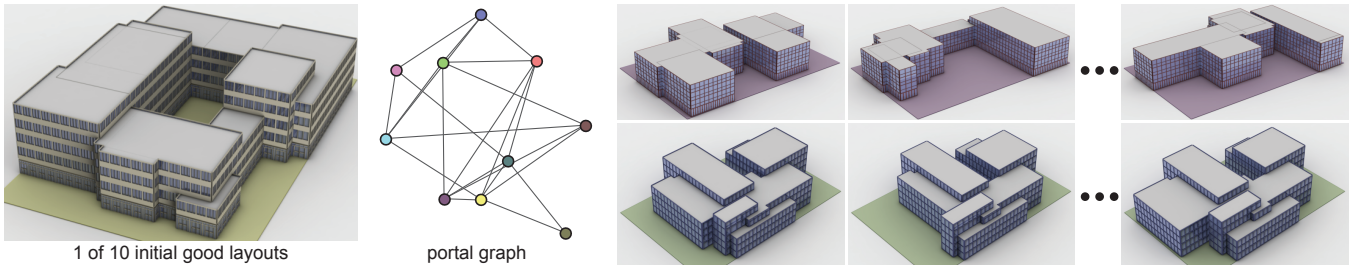

Figure 1: *Starting from a set of hard constraints (e.g., regulatory guidelines) and soft constraints (e.g., quality measures), we formulate a constrained optimization to characterize* **good** *building layouts. Then, starting from a discrete set of samples of good layouts, we analytically construct local shape spaces around each discrete layout, and link the local shape spaces via a portal graph. Exploration using the portal graph then reveals a family of layout solutions. Importantly, the user is explosed to only good layouts, simplifying layout exploration.*

Abstract

Good building layouts are required to conform to regulatory guidelines, while meeting certain quality measures. While different methods can sample the space of such good layouts, there exists little support for a user to understand and systematically explore the samples. Starting from a discrete set of good layouts, we analytically characterize the local shape space of good layouts around each initial layout, compactly encode these spaces, and link them to support transitions across the different local spaces. We represent such transitions in the form of a portal graph. The user can then use the portal graph, along with the family of local shape spaces, to globally and locally explore the space of good building layouts. We use our framework on a variety of different test scenarios to showcase an intuitive design, navigation, and exploration interface.

CR Categories: I.3.5 [Computer Graphics]: 3D Graphics and Realism—Computational Geometry and Object Modeling

Keywords: local variations, layout exploration, shape space, constrained optimization, computational design

Links: DL PDF WEB VIDEO DATA CODE

1 Introduction

Many layout generation problems can be formulated as global optimization or probabilistic sampling problems (e.g., furniture, rooms, buildings, cities, etc.). These approaches result in one or multiple (discrete) solution candidates for a user to choose. One significant challenge is to create a mathematical model to define what constitutes a *good* layout, e.g., using constraints, energy terms, or probability distributions. Such models typically have one or more of the following shortcomings: (i) too simplified for mathematical convenience; (ii) the parameters are crudely estimated due to high dimensions of the (embedded) solution space; and importantly, (iii) aesthetic or visual design factors, which are often very difficult to model, are ignored. Hence, it is desirable to allow the user to refine the solutions based on visual quality assessment. Further, in our discussions with architects and designers, we have learned that they often lament the absence of suitable (computational) guidance to facilitate design variations. Technically, the lack of appropriate characterization of the space of *good* solutions makes it difficult to refine the solutions without degrading the original layout qualities.

In this paper, we characterize the space of good *local* changes around any sampled configuration and further link multiple local characterizations corresponding to different sampled configurations to provide a *global* overview of the sampled solution space (see Figure 1). We study this problem in the context of individual buildings and parcel blocks of buildings. We first characterize the problem as an instance of constrained optimization by understanding what makes a building layout good, i.e., *valid* and *desirable*. Layouts are valid if they conform to regulations arising in the context of urban planning in the form of *hard constraints*, e.g., buildings should lie entirely inside the indicated parcel boundaries. Further, layouts are *desirable* if they improve quality measures specified as *soft constraints*, e.g., large courtyards are preferred.

Starting from an initial set of layouts (sampled or digitized from existing maps) along with associated constraints, we use constrained optimization to generate a (discrete) set of good layouts. Such layouts, however, have different parameterizations and hence cannot be directly combined (e.g., interpolated). We address this challenge in two stages. First, starting from any such good layout, we explicitly characterize the space of *local* layout variations. Specifically, we derive a low-dimensional space of variations, wherein layouts are guaranteed to remain good (i.e., valid and desirable). Second, based on an appropriate distance measure between pairs of layouts, we extract *portals* or good transition pathways linking the different local spaces thus providing a *global* overview of the extracted solution space. We then expose these variations, both local and global, via a simple and intuitive exploration interface. We eval-

ACM Reference Format
Bao, F., Yan, D., Mitra, N., Wonka, P. 2013. Generating and Exploring Good Building Layouts. ACM Trans. Graph. 32, 4, Article 122 (July 2013), 10 pages. DOI = 10.1145/2461912.2461997 http://doi.acm.org/10.1145/2461912.2461997.

0730-0301/13/07-ART122 $15.00.
DOI: http://dx.doi.org/10.1145/2461912.2461997

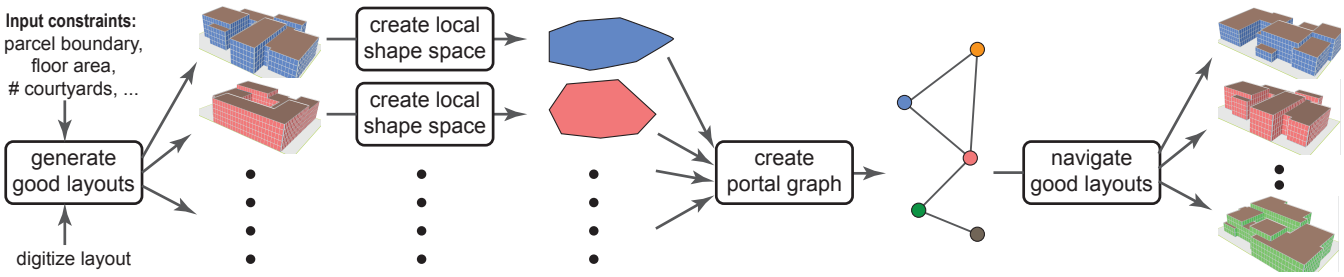

Figure 2: *Starting from an input set of hard and soft constraints, we generate good initial layouts, characterize the local shape space around each such initial layout, and then connect the local spaces using a portal graph. The portal graph along with the local shape spaces can then be interactively explored for local and global layout exploration. Note that the user sees only good (i.e., valid and desirable) layouts.*

uate our framework under different combinations of hard and soft constraints to generate good variations.

Contributions. In summary, our contributions include:

- formulating good (i.e., valid and desirable) building layout generation as an instance of constrained optimization;
- characterizing the space of local variations around any given layout that retains goodness of the original layout; and
- linking such local spaces of variations by building global connection pathways to facilitate intuitive exploration of the extracted space of good building layouts.

2 Related Work

Procedural modeling. Grammars have been successfully employed in the context of procedural building modeling [Parish and Müller 2001; Wonka et al. 2003; Müller et al. 2006]. All such methods, however, expect the user to encode the rules of the grammar, a task that is similar to implementing scripts in a professional modeling package. To simplify the rule creation process, several important ideas emerged over time: graphical user interfaces to simplify editing a grammar and the resulting models [Lipp et al. 2008]; or grammars augmented to interact with external constraints, such as guidance shapes, user input, or simulation [Měch and Prusinkiewicz 1996; Prusinkiewicz et al. 2001; Beneš et al. 2011; Talton et al. 2011]; or grammars derived from existing models, a process referred to as inverse procedural modeling [Aliaga et al. 2007; Müller et al. 2007; Bokeloh et al. 2010; Šťava et al. 2010].

Stochastic sampling. An alternate strategy for building modeling is to optimize a goal function and a set of constraints. A recent paper that embodied this approach is by Merrell et al. [2010], who computed room layouts using stochastic optimization and employed learning techniques to generate reasonable room sizes and connectivity graphs similar to existing examples. Extruding these rooms results in nice mass models. By modeling up to a few city blocks, the scale of our modeling approach falls somewhere in between residential building layouts (e.g., Merrell et al. [2010]) and city scale modeling (e.g., Vanegas et al. [2012]). We note that the data-driven method of Merrell et al. can possibly learn aesthetic design elements, which are difficult to capture using a purely analytic approach. Existing methods, however, do not support local refinements or exploration of the constrained solution space. Further, restricting sampling to only a few dimensions does not help because good local shape space variations are often distributed over the full representation (e.g., most of the boxes move in our framework even under local changes).

Retargeting existing designs. Retargeting and reshaping existing building models have been used to generate plausible variations. Cabral et al. [2009] used constrained optimization to deform architectural models while preserving angles and contact relationships; while Lin et al. [2011] broke down an architectural model into axis-aligned boxes retarget them by resizing and replicating boxes of the input model. Habbecke and Kobbelt [2012] proposed a constraint analysis and deformation framework for buildings.

Design exploration. An early inspiration for many modeling papers was the concept of design galleries [Marks et al. 1997] that is now commonplace. Shapira et al. [2009] proposed a great framework to explore different recolorings of input images; while in another interface, parametric design space of trees and human shapes [Talton et al. 2009] was proposed. In the context of isometric shape deformation, Kilian et al. [2007] proposed useful Riemannian metrics in the space of meshes to aid the user in design and modeling tasks; while for freeform architecture, Yang et al. [2011] introduced local shape space exploration that was an inspiration to our work. Large design spaces can also be sampled discretely using a probabilistic model: an idea that is complementary to our work. We are also inspired by recent efforts in deriving form from function, a design philosophy reinvented towards computational design, e.g., for modeling of furniture [Umetani et al. 2012], precast-based buildings [Liu et al. 2013], or land-use patterns [Vanegas et al. 2012]. In contrast to these efforts, we focus on exploring both local (continuous) and global (discrete) variations and allow the user to explore the choices via an interactive interface, while ensuring *goodness* of the generated layouts.

Building optimization. In civil engineering and architecture, optimization and simulation are used to find efficient ways to construct a building, e.g., [Rafiq et al. 2003; Coleman 2007; Hale and Long 2010]. While these methods mainly focus on optimizing construction details for buildings that have a fixed shape, the methods do not contribute to the shape modeling problem, as is our focus. The exploration of shape variations is limited to buildings that consist of one single box shape [Hale and Long 2010], or window arrangements on a given mass model [Gagne and Andersen 2010], or the layout of a housing blocks [Leblanc et al. 2011]. In another research thread, Whiting et al. [2009; 2012] explored structural feasibility in the context of modeling masonry buildings. They proposed a gradient-based nonlinear optimization approach to search the parameter space in procedural models to generate stable buildings, while modifying the geometry to achieve stability.

3 System Overview

At the beginning, the user has two options: (i) provide a parcel boundary, select from a set of available hard and soft constraints,

and generate initial layouts in a sampling stage; or, (ii) provide a set of digitized or modeled building layouts, select soft constraints, and extract (or select) relevant hard constraints consistent with the input layouts. Subsequently, local shape spaces of good layouts are generated, sampled, and then linked via portals (see Figure 2).

The layout space is now ready for exploration. The viewer has four windows (see Figure 3): (a) a *portal graph* showing the current shape space (around $\mathbf{\Gamma}^i$) and active connection pathways leading to other shape spaces; (b) a 2D *navigation polygon* showing the current shape space where neighboring points have visually similar layouts; (c) a set of *portal transitions* providing glimpses of neighboring portals, and (d) a *main window* showing the current layout. *Only* good layouts are made available to the user. The user can explore by selecting nodes in the portal graph, directly moving on the navigation polygon, or by interactively adjusting edges in the main window (see supplementary material). (We note that nodes may be disconnected based on the current portal jump threshold). Based on the selected building style (e.g., residential, warehouse, etc.) window and door elements are procedurally added to the current layout.

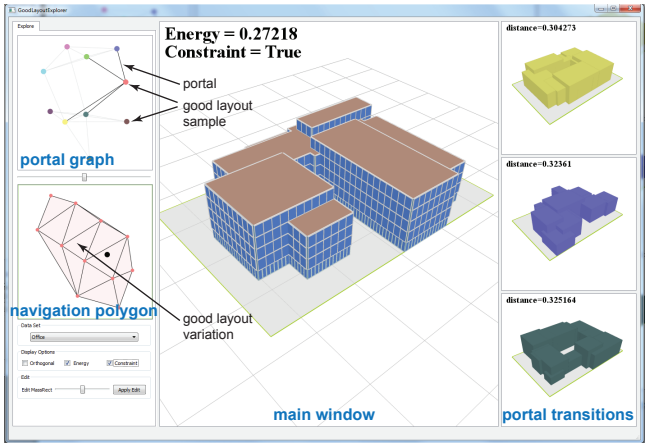

Figure 3: *Our system consists of four panels: a portal graph, a 2D navigation polygon, portal transitions, and a main window to show the current layout. Only good layouts are presented in the interface.*

4 Problem Formulation

We evaluate goodness of building layouts based on input sets of hard and soft constraints arising out of building regulations, construction efficiency, economic factors, or livability considerations. A layout is said to be *valid* if it satisfies all the *hard constraints*, and *desirable* if it has a low (acceptable) cumulative *soft constraint* energy. Layouts that are both valid and desirable are called *good*. In this section, we formulate the layout generation problem as an instance of constrained optimization. Later, aesthetic qualities are judged by the user only in the interactive exploration phase.

Representation. We represent each building layout as a union of a set of (overlapping) boxes, $\cup_i R_i$ (see Figure 4). We parameterize each box, R_i, using its size attributes (i.e., length l_i, width w_i, height h_i) and position attributes (i.e., center (x_i, y_i), rotation θ_i) and encode the attributes as a vector of six variables $[l_i, w_i, h_i, x_i, y_i, \theta_i]$. Thus, a building comprised of n boxes is represented as a single vector, $\mathbf{\Gamma} \in \mathbb{R}^d$ (e.g., $d = 6n$). We denote such parameterized buildings as $\mathbf{\Gamma}$.

Hard Constraints. Buildings have to conform to certain hard constraints, expressed as equality/inequality relations. Such constraints either arise from building guidelines (e.g., a building should lie completely inside its designated parcel or building plot, layout boundaries should maintain a minimum clearance from parcel boundaries, etc.), or from livability considerations (e.g., buildings should have a minimum thickness to facilitate access, or their boundary lines should have a minimum length to prevent unreachable corners, etc.). We abstract such conditions as hard constraints over the parameterized buildings as $\chi(\mathbf{\Gamma}) = 0$, indicating a valid layout and invalid otherwise. In our setup, we take as input a set of (linear) constraints (as equality/inequality constraint), say $\{\chi_0, \chi_1, \dots\}$.

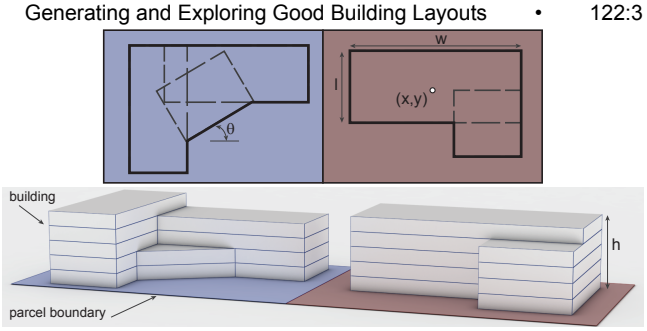

Figure 4: *We generate building layouts subject to hard constraints and soft constraints. Layouts are encoded as unions of boxes, where each box is parameterized by its position attributes, i.e., the center (x, y) and rotation θ; and its size attributes (l, w, h).*

Soft Constraints. Certain building layouts are preferred over others based on various preference and quality measures. For example, layouts with target floor areas, layouts with courtyards that provide privacy and light, or buildings that are less in shadow can be preferred. We abstract such quality measures as energy functions, $E(\mathbf{\Gamma})$, with lower energies indicating more desirable layouts. We take as input a set of (nonlinear) quality measures, say $\{E_0, E_1, \dots\}$.

Good Layout. A layout $\mathbf{\Gamma}$ is good, if $\chi_i(\mathbf{\Gamma}) = 0, \ \forall i$; and the cumulative energy $E := \sum_j E_j(\mathbf{\Gamma})$ is less than an input threshold.

5 Algorithm

Overview. Our system starts with input sets of hard and soft constraints. Then, it uses either stochastic sampling to generate or digitize existing layouts (e.g., extracted from city maps) to replicate a set of *valid* layouts. In either case, these valid layouts are optimized to generate a set of *good* layouts, say $\{\mathbf{\Gamma}^i\}$ (see Section 6). For each such layout $\mathbf{\Gamma}^i$, a local shape space of *good* layouts is then characterized and sampled, based on the modal analysis of the soft constraints (see Section 5.1). Any convex combinations of such samples produce *good* layouts, thus yielding a compact encoding of the local shape space around $\mathbf{\Gamma}^i$. However, configurations from different $\mathbf{\Gamma}^i$ and $\mathbf{\Gamma}^j$ have different representations and topologies, and cannot be directly combined. Hence, based on a layout-space similarity distance, we extract connection pathways or *portals* γ^{ij} linking local shape spaces around $\mathbf{\Gamma}^i$ and $\mathbf{\Gamma}^j$, such that visual discontinuities arising from such jumps are minimized (see Section 5.2). Note that γ^{ij} denotes a discrete jump and not a continuous transition. We expose the extracted layout space by allowing discrete transitions via the portals and thus linking the continuous local shape spaces built around each of the $\mathbf{\Gamma}^i$ (see Section 5.3).

5.1 Characterizing good local variations

Let $\mathbf{\Gamma}^i$ denote any good layout, i.e., both valid and desirable. Our goal is to characterize the local shape space to generate further layout variations $\mathbf{\Gamma}$, which are all good. In this stage, we keep the dimension of the representation, say d, fixed.

Any vector, $\mathbf{v} \in \mathbb{R}^d$ with $\|\mathbf{v}\| = 1$, defines a newly suggested layout, $\mathbf{\Gamma}_v \leftarrow \mathbf{\Gamma}^i + \mathbf{v}$. A random direction $\mathbf{v}$, however, can quickly degrade the desirability of a layout, i.e., we allow only small displacements without violating the constraints. In order to preserve the current soft constraints, we restrict movements to directions orthogonal to the gradient of the cumulative energy, i.e., $(\mathbf{\Gamma}_v - \mathbf{\Gamma}^i)^T \nabla E|_{\mathbf{\Gamma}^i} = 0$, where $\nabla E|_{\mathbf{\Gamma}^i}$ denotes the gradient evaluated at the starting layout, $\mathbf{\Gamma}^i$. Using Taylor expansion, the new energy at $\mathbf{\Gamma}_v$ is given by:

$$E(\mathbf{\Gamma}_v) \approx E(\mathbf{\Gamma}^i) + (\mathbf{\Gamma}_v - \mathbf{\Gamma}^i)^T \mathbb{H}|_{\mathbf{\Gamma}^i} (\mathbf{\Gamma}_v - \mathbf{\Gamma}^i)/2, \qquad (1)$$

where $\mathbb{H}$ denotes the Hessian of the energy function (see supplementary material) because the first-order term vanishes due to the previous condition. Note that, unlike Yang et al. [2011], we avoid building a full local osculant, which simplifies the subsequent formulation and exploration. Hence, if we restrict displacements to the lowest k eigenvectors, $\mathbf{e}_j$ for $j = 1 : k$ of the Hessian $\mathbb{H}$, i.e., $\mathbf{v} \in \sum_{j=1:k} \gamma_j \mathbf{e}_j$, the energy is best preserved as:

$$E(\mathbf{\Gamma}_v) \approx E(\mathbf{\Gamma}^i) + \sum_{j=1:k} \gamma_j^2 \lambda_j / 2 \qquad (2)$$

with λ_j denoting the corresponding eigenvalues. We decide on k based on the spectral plot of $\mathbb{H}$ (see Figure 5) using k to be the maximum $|\lambda_k| \leq 1.05|\lambda_0|$ with λ_0 being the eigenvalue with the smallest absolute value (If $k \leq 4$, we still use $k = 5$). Further, $\|\mathbf{v}\| = 1$ amounts to $\sum_{j=1:k} \gamma_j^2 = 1$. Thus, we have characterized the space of desirable variations using k dimensions.

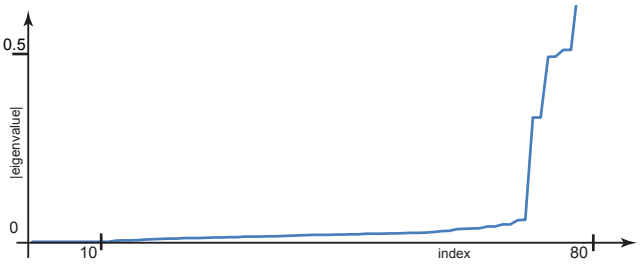

Figure 5: *A typical eigenvalue plot for Hessian matrix $\mathbb{H}$ of the soft constraints evaluated at initial good layout $\mathbf{\Gamma}^i$. We restrict sampling and navigation to the low eigen-modes (e.g., $k = 5$) to preserve layout desirability.*

Unfortunately, picking a direction this way can violate the hard constraints. We restore layout validity by correcting for violated hard constraints using quadratic programming (QP) as follows:

$$\begin{aligned} \mathbf{\Gamma}^\star := \underset{\mathbf{\Gamma}}{\operatorname{argmin}} \; & \frac{1}{2}(\mathbf{\Gamma} - \mathbf{\Gamma}_v)^T (\mathbf{\Gamma} - \mathbf{\Gamma}_v) \\ \text{s.t. } & (\mathbf{\Gamma} - \mathbf{\Gamma}^i)^T \nabla E|_{\mathbf{\Gamma}^i} = 0 \\ & \chi_j(\mathbf{\Gamma}) = 0 \quad \forall j. \end{aligned} \qquad (3)$$

Thus, we project the displacement vector, $\mathbf{v}$, to the good layout space to obtain $\mathbf{\Gamma}^\star$, giving an updated deformation, $\mathbf{v}^\star \leftarrow \mathbf{\Gamma}^\star - \mathbf{\Gamma}^i$. We invoke only the active linear constraints, i.e., those where the current layout $\mathbf{\Gamma}_v$ is close to being violated based on an allowed threshold. We solve the resultant QP in Equation 3 using Matlab's *quadprog* and line search along $\mathbf{v}^\star$ to refine the variation. Although this projection step can affect the energy, E, in practice, we found the effect to be negligible because the configurations $\mathbf{\Gamma}_v$ are close to the shape space. Note that the projected solution may deviate from the subspace spanned by the k eigenvectors of the Hessian.

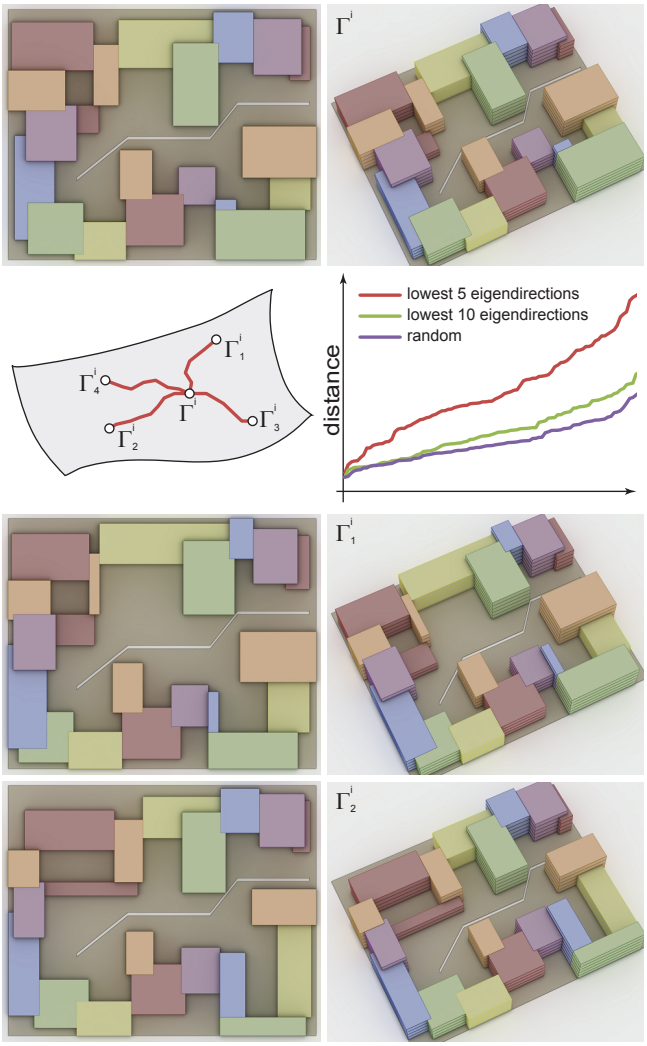

Figure 6: *Starting from a good layout, $\mathbf{\Gamma}^i$ (top), we use the lowest k ($k = 5$) eigenvectors of the Hessian of the soft constraints to explore the shape space. The solutions are then projected using a QP formulation to produce good layouts of the form $\mathbf{\Gamma}^i_j$. The graph shows that restricting navigation based on vectors from the lowest k eigen vectors (k=5 versus k=10) allows longer traversal (i.e., larger variations), while preserving goodness of the layouts. Note that as a hard constraint, the buildings cannot touch the white jagged line.*

For larger variations, we use multiple steps (10 in our examples). We reuse the original Hessian, $\mathbb{H}|_{\mathbf{\Gamma}^i}$, instead of recomputing it, to restrict movement in a consistent direction. Figure 6 shows a typical behavior where restricting navigation to the lowest few eigenmodes of the Hessian leads to larger movements (i.e., variations) in the shape space, while maintaining goodness of layouts.

For each random direction restricted to $\mathbf{v} \in \sum_{j=1:k} \gamma_j \mathbf{e}_j$, we obtain a (projected) good layout using shape space exploration as described above. Thus, in the end, we obtain a set of good layout variations $\{\mathbf{\Gamma}^i_1, \mathbf{\Gamma}^i_2, \dots\}$ all originating from $\mathbf{\Gamma}^i$. In our tests, we generated a fixed number of variations (set to 100). These variations are not necessarily all very distinct. One option is to cluster the variations based on their configuration parameters (c.f., [Vanegas et al. 2012]). But, configuration space similarity does not translate to visual similarity. Hence, we define a similarity distance directly in the space of the layouts by XOR of the volumes of any two layouts, $\mathbf{\Gamma}_1$

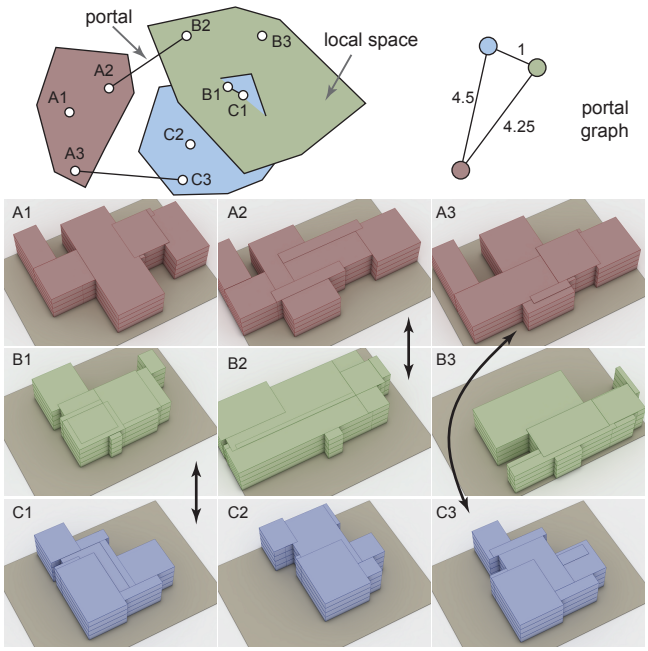

Figure 7: *Starting from a few good layout samples (A1, B3, C2), we characterize good local variations via shape space exploration to create local navigation polygons. Such local shape spaces are then linked via portal connections (e.g., A3↔C3) where shorter distances denote less visually noticeable transitions. The connectivity between the generated local spaces are encoded as a portal graph. Note that all the layouts have the same parcel boundary.*

and $\mathbf{\Gamma}_2$, i.e.,

$$d(\mathbf{\Gamma}_1, \mathbf{\Gamma}_2) := vol(\mathbf{\Gamma}_1) \oplus vol(\mathbf{\Gamma}_2). \quad (4)$$

Note that the configurations have the same parcel boundary and are hence aligned. Using this distance, we perform *farthest* sampling [Eldar et al. 1994] to retain only the top few (10 in our implementation) most visually different good layouts. Specifically, starting from a random sample from $\{\mathbf{\Gamma}^i_1, \mathbf{\Gamma}^i_2, \dots\}$, we pick the farthest sample (i.e., layout) from the remaining layouts; we progressively select additional layouts that are farthest from the currently selected layouts. Let, $\{\mathbf{\Gamma}^i_1, \dots \mathbf{\Gamma}^i_k\}$ denote these selected layouts.

By construction, all the layouts $\{\mathbf{\Gamma}^i_1, \dots \mathbf{\Gamma}^i_k\}$ are good. More importantly, since we work with linear hard constraints and sample from the space of desirable layouts (using Hessian information), any convex combination of the sampled configurations are also *good*. Thus, for any good layout, $\mathbf{\Gamma}^i$, we produce a continuous space of good layout variations as $\sum_{j=1:k} \alpha_j \mathbf{\Gamma}^i_j$ with $\sum_{j=1:k} \alpha_j = 1$ and $\alpha_j \geq 0$.

5.2 Extracting global portals

At this stage, we have a discrete set of shape spaces for good layouts respectively built around initial good layouts $\{\mathbf{\Gamma}^i\}$. These spaces, however, are disjointed, preventing users from navigating *across* these local shape spaces. Recall that each local space around $\mathbf{\Gamma}^i$ is characterized by a convex combination of $\{\mathbf{\Gamma}^i_j\}$, but the underlying dimensionality of different shape spaces can be different. This prevents any meaningful interpolation between two spaces, say $\mathbf{\Gamma}^i$ and $\mathbf{\Gamma}^j$. Instead, we look for good locations to jump between a pair of such local spaces — we call such pathways *portals*. In the context of layout exploration, we characterize good portals as those jumps that lead to small visual changes.

Ideally, given any pair of good layout spaces around $\mathbf{\Gamma}^i$ and $\mathbf{\Gamma}^j$, we want to extract parameters $\{\alpha_l\}$ and $\{\beta_m\}$ such that $d(\sum_l \alpha_l \mathbf{\Gamma}^i_l, \sum_m \beta_m \mathbf{\Gamma}^j_m)$ is minimized (subject to $\sum_l \alpha_l = 1$, $\alpha_l \geq 0$ and $\sum_m \beta_m = 1$ $\beta_m \geq 0$). Instead of directly solving this optimization, we coarsely evaluate the solution space.

We compute pairwise distances between elements from $\mathbf{\Gamma}^i$ and $\mathbf{\Gamma}^j$, and select the pair with $\gamma^{ij} := \text{argmin}_{l,m}\, d(\mathbf{\Gamma}^i_l, \mathbf{\Gamma}^j_m)$, where γ^{ij} denotes the portal. The information is encoded as a *portal graph*, where each local shape space around $\mathbf{\Gamma}^i$ becomes a node; and any two nodes are connected by an edge that represents the portal connecting them. For any edge, we take the corresponding volume-distance as its cost (i.e., $d(\gamma^{ij})$ with abuse of notation), with higher costs denoting larger visual disparity. (Note that for the distance computations, we assume that the parcel boundaries are consistent across different local spaces and hence the layouts can be directly compared.) Figure 7 shows an example.

For higher accuracy, it is possible to extract portals across the dense sampling of points first (100 in our setting) in shape spaces of $\mathbf{\Gamma}^i$ to find the portals and *then* use farthest point sampling by keeping the portal entry points as fixed landmark locations.

5.3 Navigating good layouts

At this stage, we have generated a set of local shape spaces of good layouts in the form $\{\{\mathbf{\Gamma}^1_1, \dots \mathbf{\Gamma}^1_k\}, \{\mathbf{\Gamma}^2_1, \dots, \mathbf{\Gamma}^2_k\}, \dots\}$ (for notational simplicity, we assume each local set has k good layouts). Inside each local set, any layout of the form $\sum_{j=1:k} \alpha_j \mathbf{\Gamma}^i_j$ with $\sum_{j=1:k} \alpha_j = 1, \alpha_j \geq 0$ is a good layout by construction. We now describe how to navigate and explore such good layouts, first locally and then globally across these local spaces.

Local exploration. We support two modes: (i) *handle-driven exploration*, where the user directly edits the current layout by dragging the layout edges, while the system restricts changes to the current local space; and (ii) *navigation polygon*-based exploration, where the user can move on a 2D embedded map of the local shape space where relative distances reflect the corresponding relative distances in terms of object space similarity (see Equation 4).

Handle-driven exploration. The user selects an edge of one of the boxes of the current layout, say $\mathbf{\Gamma}_0$, and then prescribes a target location. Note that the user selects an edge of $\mathbf{\Gamma}_0$ and not necessarily a full facade (see Figure 8). Assume that $\{\mathbf{\Gamma}^i_j\}$ denotes the current local set. Thus, the target layout is a new variation restricted to the local convex set such that it also satisfies certain new (linear) constraints $\chi_l(\mathbf{\Gamma}) = 0$, e.g., moving certain edge(s) to new position(s). In Figure 8, where the layout is axis-aligned, such a constraint is

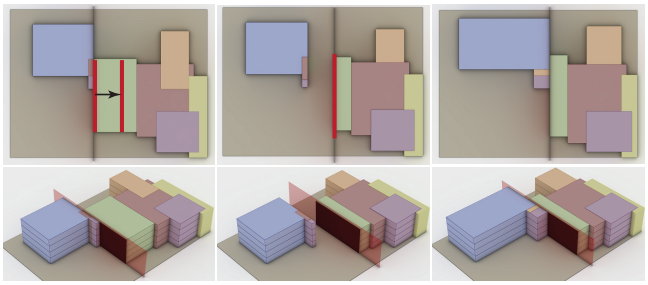

Figure 8: *The user can directly edit current layouts by dragging an edge to a new position (left); the corresponding box changes (here the green box is squeezed (middle)) and then we find a good solution by solving a QP by restricting the solution to the low-dimensional parameterization of the local shape space.*

simply a requirement that the corresponding edge ends up with a certain x (or y) coordinate. To obtain the new variation, we first try to solve the following QP problem:

$$\{\alpha_j^\star\} := \underset{\alpha_j}{\operatorname{argmin}} \frac{1}{2}\Big(\sum_{j=1:k} \alpha_j \mathbf{\Gamma}_j^i - \mathbf{\Gamma}_0\Big)^T \Big(\sum_{j=1:k} \alpha_j \mathbf{\Gamma}_j^i - \mathbf{\Gamma}_0\Big)$$
$$\text{s.t.} \sum_{j=1:k} \alpha_j = 1$$
$$\alpha_j \geq 0 \quad \forall j$$
$$\chi_l\Big(\sum_{j=1:k} \alpha_j \mathbf{\Gamma}_j^i\Big) = 0 \quad \forall l. \qquad (5)$$

Note that we do not require any additional conditions to ensure goodness since the parameterization (with k parameters) already ensures goodness. If no solution exists, then we satisfy the new constraints in a least-squares sense using:

$$\{\alpha_j^\star\} := \underset{\alpha_j}{\operatorname{argmin}} \frac{1}{2} \sum_l \chi_l\Big(\sum_{j=1:k} \alpha_j \mathbf{\Gamma}_j^i\Big)^2$$
$$\text{s.t.} \sum_{j=1:k} \alpha_j = 1$$
$$\alpha_j \geq 0 \quad \forall j. \qquad (6)$$

The new variation is $\mathbf{\Gamma}^\star = \sum_{j=1:k} \alpha_j^\star \mathbf{\Gamma}_j^i$. In our setting, this optimization runs at interactive rates, e.g., on the order of seconds for layouts involving 10-20 boxes (see supplementary material).

Navigation-polygon. Local layout spaces are parameterized in k (e.g., 10) dimensions, which are still large for the user to explore. Hence, we map the samples to 2D producing a navigation polygon (P^i for $\mathbf{\Gamma}^i$). Specifically, we compute all pairwise distances of the form $d(\mathbf{\Gamma}_j^i, \mathbf{\Gamma}_k^i)$ and map the points to 2D using multidimensional scaling (MDS) ([Borg and Groenen 2005], chapter 8). Nearby points in this navigation polygon denote comparable layouts. We mesh these 2D points $\{\mathbf{p}_j^i\}$ using Delaunay triangulation and allow the user to directly navigate inside the resulting triangulation of the polygon, P^i. Say, the user indicates point $\mathbf{x} \in P^i$. We locate the corresponding triangle and find the barycentric coordinates of $\mathbf{x}$ in this triangle. Navigation then amounts to using these coordinates to interpolate the layouts corresponding to the triangle vertices (i.e., three configurations of the form $\mathbf{\Gamma}_j^i$).

Global exploration. We provide a visual representation of the global space, via the *portal graph*, by drawing the navigation polygons, P^i, as nodes in a graph and linking them with edges. (The nodes are embedded in 2D using MDS with the minimal distance between pairs of local shape spaces.) The user can explore the space of good layouts directly in one of the following ways: (i) if the user is in the navigation polygon, P^i, she can select any of the outgoing edges (with the jump distance below the edge threshold) to use the respective portal, γ^{ij}, to transition to the navigation polygon, P^j. We animate this transition by first moving the current configuration to the portal entry point in P^i (i.e., interpolation using the navigation polygon) and then jumping to P^j; (ii) if the user selects a target local space by selecting a node in the portal graph,the shortest path in the portal graph to the node from the current location is computed and then the path similar to interaction type #i is animated, possibly via a sequence of alternating smooth interpolation and portal jumps; and (iii) if the user, while navigating, is close to one or more portals, the system suggests the current portal transition options for the user to select from or continues navigating using the current navigation polygon. We found the last mode of navigation particularly useful when the number of nodes/edges in the portal graph is large making direct graph-based navigation cumbersome. The supplementary demo presents these exploration options.

6 Generating Good Layouts

6.1 Initial Layout Generation

We provide two methods for the user to generate initial layouts, as described next.

Simulated annealing. We start by generating candidate layouts using simulated annealing (SA) (see also [Merrell et al. 2011; Yu et al. 2011] for similar applications in the context of furniture layout). While this stage can be also replaced by more advanced sampling strategies like [Talton et al. 2011; Yeh et al. 2012], we found SA to be sufficient for the initial sampling.

We start with a null layout, i.e., $\mathbf{\Gamma} \leftarrow \emptyset$, and set the initial energy to $E \leftarrow \infty$. We then iteratively apply one of the following steps, chosen at random (see Figure 9):
(i) add a box b, i.e., $\mathbf{\Gamma} \leftarrow \mathbf{\Gamma} \cup b$;
(ii) remove an existing box b at random, i.e., $\mathbf{\Gamma} \leftarrow \mathbf{\Gamma}/b$;
(iii) move an existing box b by randomly perturbing b_x, b_y;
(iv) resize an existing box b by randomly changing its size, b_l, b_w, b_h.
For numerical reasons, vertices/edges of any pair of (aligned) rectangles that are closer than a threshold (set to 3% of the parcel size) are snapped; similarly edges of b are snapped to parcel edges, if b is sufficiently close to any parcel boundary. We calculate the energy of the new layout as E_{new}.

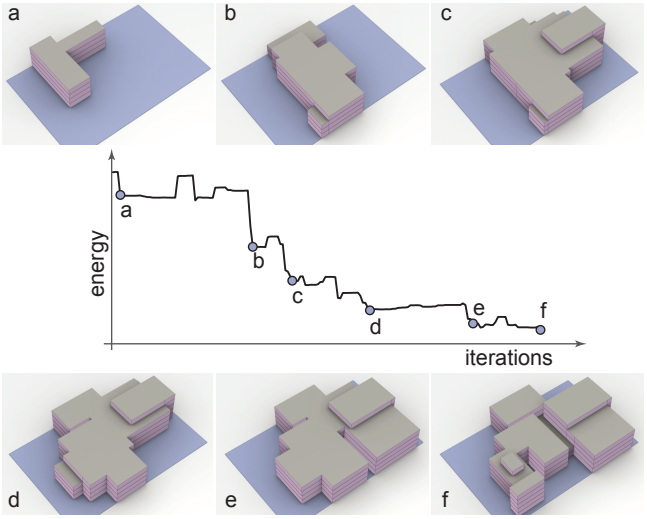

Figure 9: *(a–f) Evolution of a layout generated using simulated annealing (SA) based sampling. The layout has a hard constraint to lie entirely inside the parcel; while, the are soft constraints (e.g., target height distribution, desired floor area, number of courtyards, etc.) are grouped together as a cumulative energy to guide SA.*

If a sampled layout is not fully inside the input parcel, we simply reject the sample. Otherwise, if $E_{new} \leq E$, we accept the solution; else, in the annealing step, we accept the new solution with probability of $\exp(-(E_{new} - E)/t)$, where t is the temperature. If we accept the new solution, we set $E \leftarrow E_{new}$; otherwise the old layout and energy are retained. In the annealing schedule, we progressively reduce temperature t in each iteration. We stop if either the maximum number of steps (2000-5000 in our tests) has been reached, or when E falls below a prescribed threshold.

In this stage, we use the topological properties to measure configuration energy. For example, if a desirable attribute value is v' and the current value is $v(\mathbf{\Gamma})$, we set the normalized energy to

$E_{new}(\mathbf{\Gamma}) := ((v(\mathbf{\Gamma}) - v')/v')^2$. In the examples, we set target values for the number of boxes (3–30/parcel); the number of edges on the layout boundary (4–100); the number of holes (0–3); and the number of courtyards (0–5).

The height distribution is specified by a target histogram, i.e., number of floors as (f_i', n_i') sorted according to normalized number of floors. For example, $\{(2, 3/6), (4, 2/6), (7, 1/6)\}$, means 3 boxes have 2 floors, 2 boxes have 4 floors, and 1 box has 7 floors. Similarly, for the current layout we compute the (sorted) distribution $(f_i, n_i/\sum_i n_i)$. We define the corresponding energy as $\Pi_i(|f_i - f_i'|/f_{\max} + 1)(|n_i/\sum_i n_i - n_i'| + 1) - 1$.

We set the cumulative energy in the SA iterations as the weighted sum of the above energies.

Digitizing existing layouts. Initial layouts can also be generated by converting existing plans, modeling new layouts from scratch, or digitizing layouts from existing city maps (we show examples of the third option in Section 7). Given a set of layouts, constraints such as parcel boundaries, minimum width, covered floor area, or distance to the parcel boundary are automatically extracted and subsequently (optionally) adjusted by the user.

6.2 Constrained Optimization

In the initial layout generation stage, we obtain a set of candidate layouts. These layouts, however, may not yet be good as they may violate the hard and soft constraints (e.g., for a digitized layout). Let $\mathbf{\Gamma}$ be such a layout. If the current soft constraint energy is higher than the desired threshold level, we first improve the desirability of the current layout using gradient descent. Specifically, if $\|\nabla E\|$ is non-negligible (i.e., not already at a minima), we take a step of size β (set to 0.1 by default) and refine the current layout as $\mathbf{\Gamma}_g \leftarrow \mathbf{\Gamma} - \beta \nabla E/\|\nabla E\|$, where ∇E denotes the gradient evaluated at the current configuration. Next, we restore the violated hard constraints, if any, using a QP to project the layout to the good shape space as:

$$\mathbf{\Gamma}^* := \underset{\mathbf{\Gamma}_g}{\operatorname{argmin}} \frac{1}{2}(\mathbf{\Gamma}_g - \mathbf{\Gamma})^T(\mathbf{\Gamma}_g - \mathbf{\Gamma}) \quad \text{s.t. } \chi_i(\mathbf{\Gamma}_g) = 0 \quad \forall i. \tag{7}$$

We update the current layout as $\mathbf{\Gamma} \leftarrow \mathbf{\Gamma}^*$ and start a new iteration until the energy, E, falls below the input threshold, or a maximum number of iterations (50-100 in our tests) is reached. Figure 10 shows an example of this process.

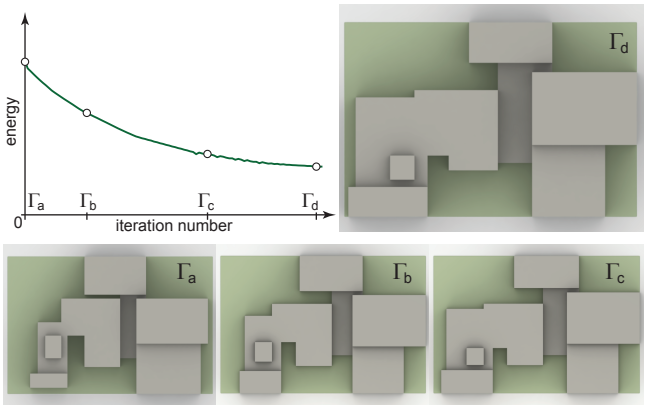

Figure 10: *Starting from a sampled layout $\mathbf{\Gamma}_a$ (bottom-left), we perform gradient descent using a QP formulation to produce valid and desirable layout $\mathbf{\Gamma}_d$. Here, we show a typical layout evolution using a combination of courtyard and covered area energies.*

6.3 Constraints

We now describe the different hard and soft constraints used in our framework (see Figure 11). While our choices were motivated by standard planning conventions and design guidelines, other constraints can similarly be formulated and handled.

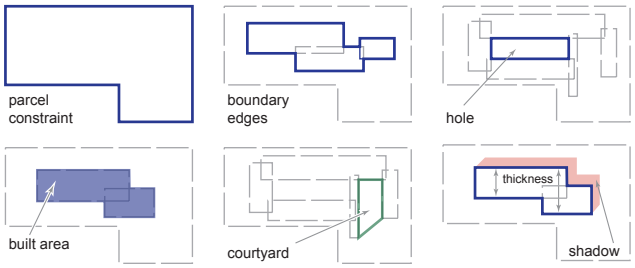

Figure 11: *Various hard constraints (e.g., parcel) and soft constraints used in our framework.*

Hard constraints. The following are the hard constraints, which are all linear equality/inequality constraints.

Parcel constraints. All the boxes in any configuration $\mathbf{\Gamma}$ are constrained to remain inside the specified parcel boundary. Further, edges on the parcel boundary can be constrained to remain on the boundary (e.g., on construction lines).

Topology constraints. In order to preserve the connectivity of the current union shape, we add appropriate (equality and inequality) linear constraints to maintain relative sidedness between pairs of edges (see supplementary demo).

Thickness constraints. It is desirable to avoid too narrow or too thick buildings. We ensure this as a thickness constraint where the distance t_i between pairs of candidate parallel lines is constrained as: $t_{min} \leq t_i \leq t_{max}$. We use an approximate medial axis of $\mathbf{\Gamma}$ to identify participating parallel edge pairs.

Soft constraints. We now present the soft constraints, with lower energies denoting more desirable configurations. The supplementary material includes detailed expressions and the associated gradient and Hessian terms.

Covered area energy. We add an energy to measure the deviation from the target occupied parcel area, A'_c, as $E_{cov}(\mathbf{\Gamma}) := |(A_1 - A'_c)/A'_c|$, where A'_c denotes the target covered floor area of the layout and A_1 denotes the ground floor area.

Courtyard energy. In order to encourage larger courtyards, we compute the area of the inner facades (i.e., facades inside a courtyard) receiving sunlight from a fixed directional light and normalize the area by the total area of the inner facades. We use this ratio as the courtyard energy (see supplementary material for details).

Shadow energy. A building can block light access of its neighboring buildings, which is of particular concern in dense cities (e.g., London). We define an energy (see supplementary for details) to penalize this shadowing effect onto adjacent buildings and onto the building itself. For simplicity, we take the direction of light to be given (e.g., the direction of sunlight at midday). Figure 12 shows the effect of this shadow energy.

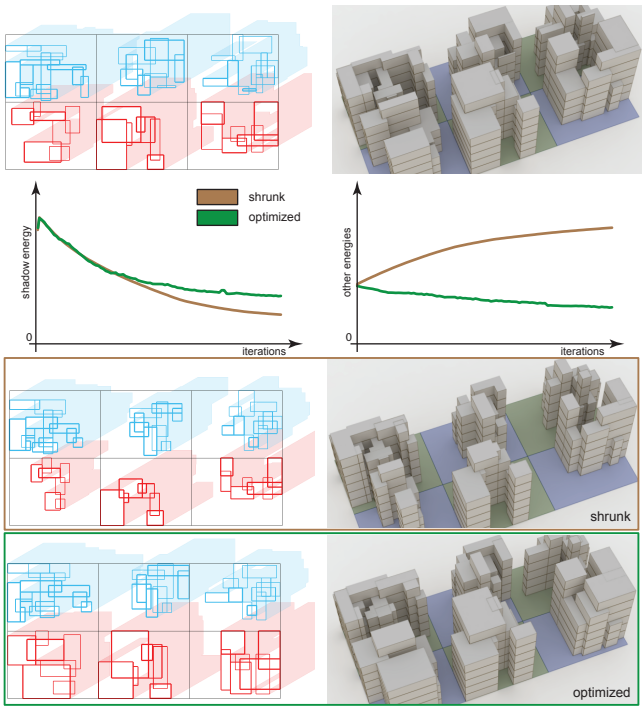

Figure 12: *(Top) High-rises in a dense neighborhood often block sunlight to the neighboring buildings. A naïve reshaping by shrinking buildings can reduce such shadowing effects, but at the cost of degradation of other desirable energies. (Bottom) Our coupled optimization reduces the shadow effect, while preserving the other energies. Shadow energies (shown in blue and red) are computed under a directional light assumption (see supplementary material).*

Heat energy. A building with a low exposed surface area (i.e., areas of the outer walls and roof area) to enclosed volume ratio is desirable as it better retains heat (see supplementary material for details).

7 Results

We implemented our layout generation framework in C# with Matlab as the backend for the QP optimization, while the shape space exploration interface was written in C++. The exploration interface supports procedural functions to add simple facade and roof elements to help better visualize the scale of the models (see supplementary demo).

We generated five test datasets to evaluate our method: (i) *offices:* commercial buildings generated by random sampling (see sample results in Figure 1); (ii) *villas:* very large single family houses, digitized from a map; (iii) *skyscrapers:* tall buildings digitized from around the world; (iv) *London:* buildings in an axis-aligned city block with initial layouts digitized from a London map; and

Table 1: *Performance statistics on a 3.4GHz i7, 4GB RAM laptop with time measured in seconds.*

Name	#nds	#boxes	QP	init.	opt.	loc. sp.	P.G.
offices	10	11.7	793.7	22.4	22.1	1.8	20.8
villas	10	10.3	496.6	-	14.0	0.6	5.9
skysc.	6	7.3	839.5	-	69.2	5.9	92.3
London	8	24.6	1189.9	-	40.5	5.1	23.3
Paris	6	14.3	785.5	31.8	46.8	2.8	11.6

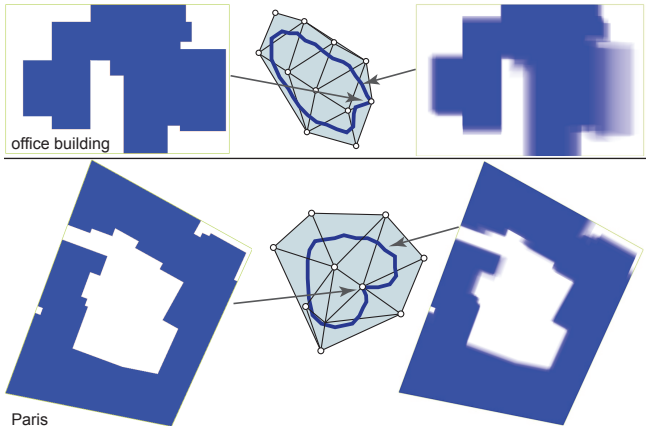

Figure 13: *Trails left during local shape space exploration via navigation polygons are visualized in the corresponding layout spaces. Such a visualization highlights which parts of the layouts are more constrained, i.e., sharp regions denote near-invariant locations, while fuzzy regions indicate where it is easier to change layouts while retaining their goodness. Such a preview is useful for deciding where to apply constrained editing, e.g., see Figure 8.*

(v) *Paris:* residential buildings generated by random sampling in a city block (city block stems of Paris).

We report statistics for the five datasets in Table 1 including generation times (in seconds) for each step. The generation times are already reasonably fast, but could be optimized using an integrated C++ implementation. In practice, a bottleneck for layout generation is the large number of constraints arising mainly due to layout topology, which can occasionally cause numerical problems. In future work, we plan to explore smart interfaces to support manual editing or adjusting the threshold used in the optimization.

The layout exploration runs at interactive rates and can be better judged by the supplementary demo. In Figure 11, we illustrate some of the constraints used in the test scenes. Figure 13 shows aggregated renderings of explorations paths in local shape spaces. By combining the layouts arising while traversing such a path, we can get a quick overview of the major variation modes in this local shape space. Further, by relaxing the allowed thresholds for the in-

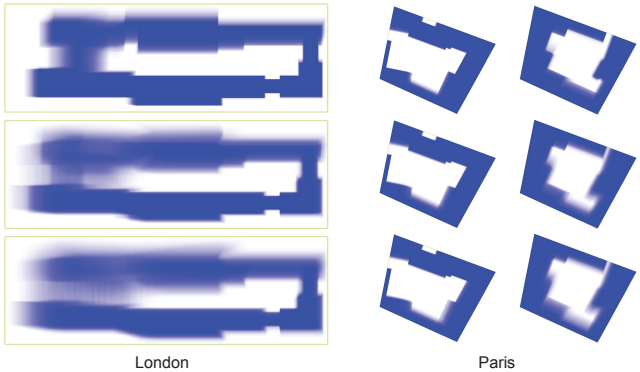

Figure 14: *The effect of varying the thresholds for the input set of constraints on the resulting layout variations. As looser thresholds are allowed (top-to-bottom), the corresponding local variations span larger spaces. This is visualized by aggregated renderings of many layouts from the local shape space. The supplementary demo shows these variations.*

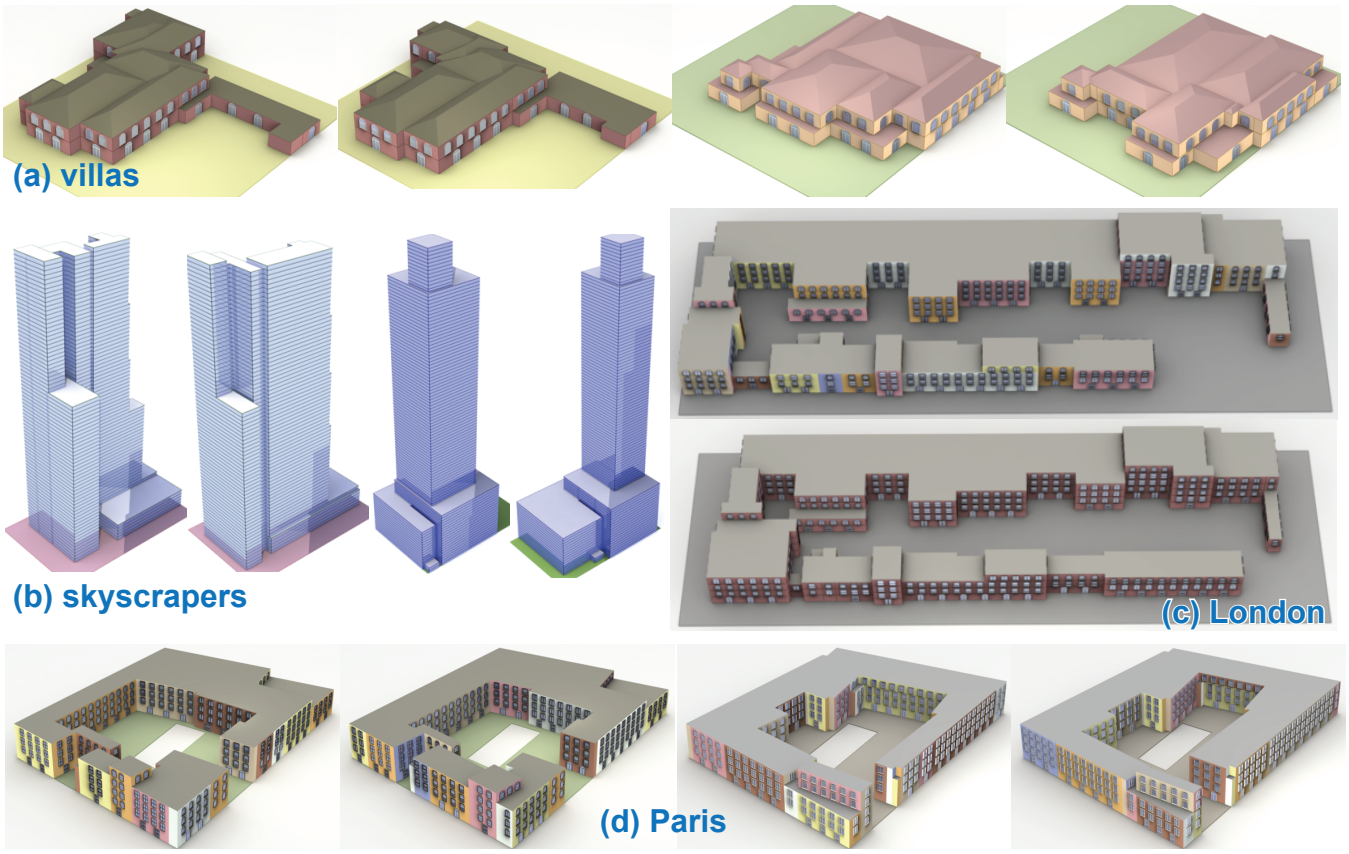

Figure 15: *Different layouts obtained using our shape space based local and global exploration of good building layouts.*

put constraints, we can increase the extent of allowed variations, as shown in Figure 14. Here, we show some representative results for each data set in Figures 1 and 15.

Limitations. Currently, our algorithm has several limitations. First, we are restricted to mass models consisting of extruded boxes. It seems conceivable to extend our framework to handle cylinders and curved footprints. In contrast, an extension to free-form architecture would require a technically different approach. Second, we only implemented a few example constraints and our list is by no means exhaustive. Adding additional constraints will require some programming efforts and cannot be done by an artist or architect. Third, our work is currently restricted to mass models. While our results can be enhanced by procedural methods, it would be interesting to see, for example, how lighting considerations can improve the layout of windows. Finally, in the current implementation, we do not model the effects of the terrain on the layouts.

8 Conclusions and Future Work

In this paper, we addressed the problem of generating and exploring *good* building layouts, i.e., layouts that fulfill certain hard and soft constraints. First, we formulated this modeling problem using regulatory guidelines and functional quality measures, such as lighting and heating. Second, we supported creation and exploration of local and global shape spaces by analytically characterizing the local shape space of good layouts around any given good layout, compactly encoding multiple local spaces, and linking them via a portal graph to support transitions across the different local spaces. Finally, we introduced a novel interface to explore the good layout space via both discrete and continuous variations.

In the future, on the application side, we would like to explore the applicability of our framework to other mixed continuous and discrete layout problems, such as street graph design, landscape design, architectural paneling, and room layouts. Further, a very interesting fundamental research question is how to combine exploration with online optimization. An interesting question in this context is how the user can specify areas of a shape space that should be explored by the optimization. In a similar note, we still do not have a systematic way to allow users to explore the full scope of any design space, specifically, how to assert that parts of the underlying shape space do not remain invisible. In absence of a global characterization, especially in high dimensions, this remains a fundamental problem and bottleneck. Finally, given the generality of the proposed approach, it would be interesting to explore its applicability in other areas.

Acknowledgements. We are grateful to Lars Hesselgren for insightful comments and to the anonymous reviewers for their constructive comments. The work was partially supported by a KAUST visiting scholarship, National Science Foundation, National Natural Science Foundation of China (No. 61271431 and 61172104), and a Marie Curie Career Integration Grant. We thank Yoshihiro Kobayashi and Christopher Grasso for their help with the renderings, and Charlotte Rakhit for video voiceover.

References

ALIAGA, D. G., ROSEN, P. A., AND BEKINS, D. R. 2007. Style grammars for interactive visualization of architecture. *IEEE*

TVCG 13, 4, 786–797.

BENEŠ, B., ŠT'AVA, O., MĚCH, R., AND MILLER, G. 2011. Guided procedural modeling. *CGF (Eurographics) 30*, 2, 325–334.

BOKELOH, M., WAND, M., AND SEIDEL, H.-P. 2010. A connection between partial symmetry and inverse procedural modeling. *ACM TOG (SIGGRAPH) 29*, 4, 104:1–104:10.

BORG, I., AND GROENEN, P. J. 2005. *Modern Multidimensional Scaling Theory and Applications*.

CABRAL, M., LEFEBVRE, S., DACHSBACHER, C., AND DRETTAKIS, G. 2009. Structure-preserving reshape for textured architectural scenes. *CGF (Eurographics) 28*, 2, 469–480.

COLEMAN, K., 2007. Building optimization: An integrated approach to the design of tall buildings. master thesis, MIT.

ELDAR, Y., LINDENBAUM, M., PORAT, M., AND ZEEVI, Y. 1994. The farthest point strategy for progressive image sampling. In *Pattern Recognition*, vol. 3, 93–97.

GAGNE, J., AND ANDERSEN, M. 2010. Multi-objective façade optimization for daylighting design using a genetic algorithm. In *SimBuild 2010*.

HABBECKE, M., AND KOBBELT, L. 2012. Linear analysis of nonlinear constraints for interactive geometric modeling. *CGF (Eurographics) 31*, 2, 641–650.

HALE, E. T., AND LONG, N. L. 2010. Enumerating a diverse set of building designs using discrete optimization. In *SimBuild 2010*.

KILIAN, M., MITRA, N. J., AND POTTMANN, H. 2007. Geometric modeling in shape space. *ACM TOG (SIGGRAPH) 26*, 3, 64:1–64:8.

LEBLANC, L., HOULE, J., AND POULIN, P. 2011. Component-based modeling of complete buildings. In *Graphics Interface 2011*, 87–94.

LIN, J., COHEN-OR, D., ZHANG, H., LIANG, C., SHARF, A., DEUSSEN, O., AND CHEN, B. 2011. Structure-preserving retargeting of irregular 3D architecture. *ACM TOG (SIGGRAPH Asia) 30*, 6, 183:1–183:10.

LIPP, M., WONKA, P., AND WIMMER, M. 2008. Interactive visual editing of grammars for procedural architecture. *ACM TOG (SIGGRAPH) 27*, 3, 102:1–102:10.

LIU, H., YANG, Y.-L., ALHALAWANI, S., AND MITRA, N. J. 2013. Constraint-aware interior layout exploration for precast concrete-based buildings. *The Visual Computer*.

MARKS, J., ANDALMAN, B., BEARDSLEY, P., FREEMAN, W., GIBSON, S., HODGINS, J., KANG, T., MIRTICH, B., PFISTER, H., RUML, W., RYALL, K., SEIMS, J., AND SHIEBER, S. 1997. Design galleries: a general approach to setting params. for computer graphics and animation. In *Proc. SIGGRAPH*, 389–400.

MERRELL, P., SCHKUFZA, E., AND KOLTUN, V. 2010. Computer-generated residential building layouts. *ACM TOG (SIGGRAPH Asia) 29*, 6, 181:1–181:12.

MERRELL, P., SCHKUFZA, E., LI, Z., AGRAWALA, M., AND KOLTUN, V. 2011. Interactive furniture layout using interior design guidelines. *ACM TOG (SIGGRAPH) 30*, 4, 87:1–87:9.

MÜLLER, P., WONKA, P., HAEGLER, S., ULMER, A., AND GOOL, L. V. 2006. Procedural modeling of buildings. *ACM TOG (SIGGRAPH) 25*, 3, 614–623.

MÜLLER, P., ZENG, G., WONKA, P., AND GOOL, L. V. 2007. Image-based procedural modeling of facades. *ACM TOG (SIGGRAPH) 26*, 3, 85:1–85:9.

MĚCH, R., AND PRUSINKIEWICZ, P. 1996. Visual models of plants interacting with their environment. In *Proc. SIGGRAPH*, 397–410.

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

PRUSINKIEWICZ, P., MÜNDERMANN, L., KARWOWSKI, R., AND LANE, B. 2001. The use of positional information in the modeling of plants. In *Proc. SIGGRAPH*, 289–300.

RAFIQ, M. Y., MATHEWS, J. D., AND BULLOCK, G. N. 2003. Conceptual building design – an evolutionary approach. *ASCE Journal of Computing in Civil Engineering 17*, 3, 150–158.

SHAPIRA, L., SHAMIR, A., AND COHEN-OR, D. 2009. Image appearance exploration by model-based navigation. *CGF (Eurographics) 28*, 2, 629–638.

ŠT'AVA, O., BENEŠ, B., MĚCH, R., ALIAGA, D. G., AND KRIŠTOF, P. 2010. Inverse procedural modeling by automatic generation of L-systems. *CGF (Eurographics) 29*, 2, 665–674.

TALTON, J. O., GIBSON, D., YANG, L., HANRAHAN, P., AND KOLTUN, V. 2009. Exploratory modeling with collaborative design spaces. *ACM TOG (SIGGRAPH Asia) 28*, 5, 167:1–167:10.

TALTON, J. O., LOU, Y., LESSER, S., DUKE, J., MĚCH, R., AND KOLTUN, V. 2011. Metropolis procedural modeling. *ACM TOG 30*, 2, 11:1–11:14.

UMETANI, N., IGARASHI, T., AND MITRA, N. J. 2012. Guided exploration of physically valid shapes for furniture design. *ACM TOG (SIGGRAPH) 31*, 4, 86:1–86:11.

VANEGAS, C. A., GARCIA-DORADO, I., ALIAGA, D., BENES, B., AND WADDELL, P. 2012. Inverse design of urban procedural models. *ACM TOG (SIGGRAPH Asia) 31*, 6, 168:1– 168:12.

WHITING, E., OCHSENDORF, J., AND DURAND, F. 2009. Procedural modeling of structurally-sound masonry buildings. *ACM TOG (SIGGRAPH Asia) 28*, 5, 112:1–112:9.

WHITING, E., SHIN, H., WANG, R., OCHSENDORF, J., AND DURAND, F. 2012. Structural optimization of 3D masonry buildings. *ACM TOG (SIGGRAPH Asia) 31*, 6, 159:1–159:11.

WONKA, P., WIMMER, M., SILLION, F. X., AND RIBARSKY, W. 2003. Instant architecture. *ACM TOG (SIGGRAPH) 22*, 3, 669–677.

YANG, Y.-L., YANG, Y.-J., POTTMANN, H., AND MITRA, N. J. 2011. Shape space exploration of constrained meshes. *ACM TOG (SIGGRAPH Asia) 30*, 6, 124:1–124:12.

YEH, Y.-T., YANG, L., WATSON, M., GOODMAN, N. D., AND HANRAHAN, P. 2012. Synthesizing open worlds with constraints using locally annealed reversible jump MCMC. *ACM TOG (SIGGRAPH) 31*, 4, 56:1–56:11.

YU, L.-F., YEUNG, S.-K., TANG, C.-K., TERZOPOULOS, D., CHAN, T. F., AND OSHER, S. 2011. Make it home: Automatic optimization of furniture arrangement. *ACM TOG (SIGGRAPH) 30*, 4, 86:1–86:11.