Interactive Example-Based Urban Layout Synthesis

Daniel G. Aliaga Carlos A. Vanegas Bedřich Beneš
Purdue University

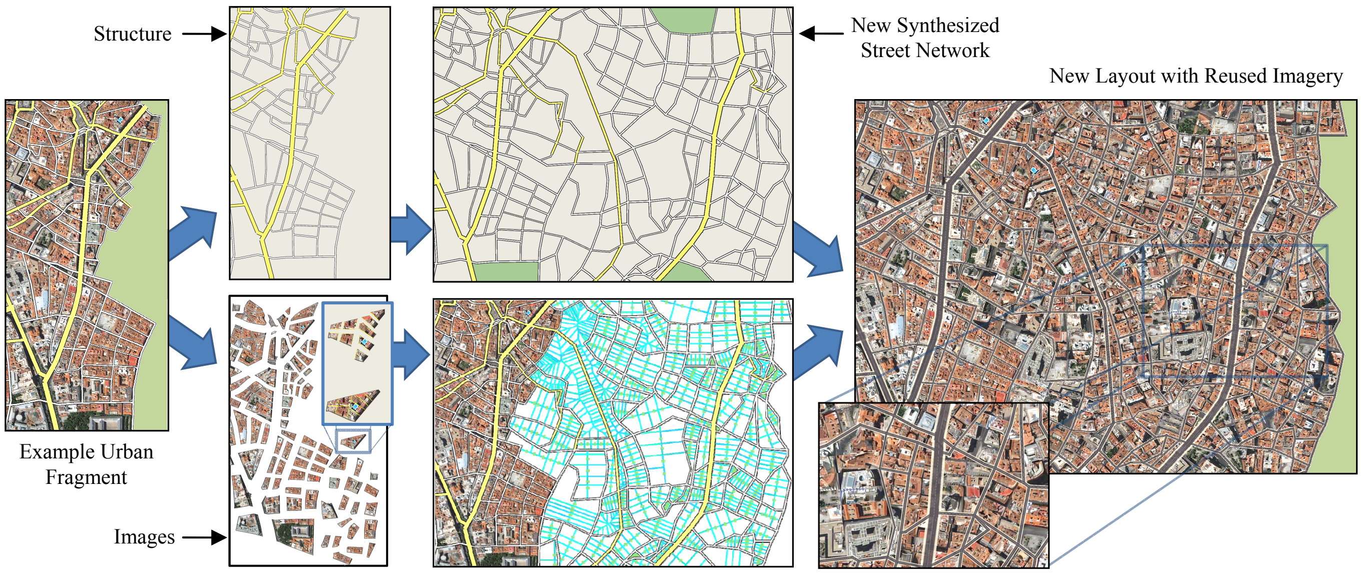

Structure
New Synthesized Street Network
New Layout with Reused Imagery
Example Urban Fragment
Images

Figure 1. **Urban Layout Synthesis**. First, our method extracts the street network and per-parcel aerial-view images from real-world urban layouts. Then, using an example-based approach new layouts are interactively created by synthesizing streets and images based on data from the example layout fragments.

Abstract

We present an interactive system for synthesizing urban layouts by example. Our method simultaneously performs both a structure-based synthesis and an image-based synthesis to generate a complete urban layout with a plausible street network and with aerial-view imagery. Our approach uses the structure and image data of real-world urban areas and a synthesis algorithm to provide several high-level operations to easily and interactively generate complex layouts by example. The user can create new urban layouts by a sequence of operations such as join, expand, and blend without being concerned about low-level structural details. Further, the ability to blend example urban layout fragments provides a powerful way to generate new synthetic content. We demonstrate our system by creating urban layouts using example fragments from several real-world cities, each ranging from hundreds to thousands of city blocks and parcels.

Keywords: example-based, texture and image synthesis, procedural modeling, content-aware image editing.
CR Categories: I.3 [Computer Graphics], I.3.3 [Picture/Image Generation], I.3.5 [Computational Geometry and Object Modeling], I.3.6 [Methodology and Techniques].

1. INTRODUCTION

We investigate the problem of synthesizing *urban layouts* by example. An urban layout refers to aerial-view images of urban areas combined with vector-based data describing the street and parcel network. Such an objective is pursued by numerous applications including the creation of new urban spaces in a similar style to an existing location [Mumford 1961], the design and extension of an urban area according to expected urban growth and development [Waddell et al. 2007], and the search for urban designs to accommodate evacuation routes, resources, and policing [Palma et al. 2007]. This paper describes an example-based framework (e.g., [Funkhouser et al. 2004, Hertzmann et al. 2001, Kwatra et al. 2005]) for intuitive interactive synthesis of urban layouts, using concepts from computer graphics, texture and image synthesis, and content-aware image editing.

In contrast with traditional image synthesis, an urban layout is a *structured image* which contains both vector and image data. While the imposed structure might facilitate some tasks (e.g., finding a route from one location to another), it makes other tasks more challenging. In particular, easy and intuitive modification and creation of new urban layouts are difficult because both image and vector data must be altered in a coordinated fashion. On the one hand, if we consider an urban layout to be an image without any structure, then we could use standard image operations to create new layouts by manipulating *image fragments,* defined as parts of an aerial image of an urban area with no structural information. The image fragments can be copied and pasted together using a mild amount of overlap and image blending (e.g., [Hertzmann2001]). However, this process provides no guarantees that the resulting street and parcel network is structurally sound. On the other hand, we could use the structure implied by the street and parcel network to partition an existing urban layout into rigid *structure fragments*. The creation task becomes either adding structure fragments to an existing configuration or creating a new tiling of structure fragments (e.g., [Cohen et al. 2003]). Unfortunately, geometrically tiling a plane is restricted to a small set of polygonal shapes. Moreover, we have the further challenge of making the streets of adjacent structure fragments form a plausible connected street network. While the tiling of non-perfectly fitting image fragments has been studied (e.g., [Kim02]), these methods require deforming and blending the boundaries which does not address making the streets of adjacent structure fragments connected. None of these approaches ensure a sensible urban layout is maintained or provide an intuitive and interactive mechanism to quickly synthesize an urban layout that is similar to, or explicitly different from, input examples.

Our key observation is that we can generate urban layouts by *simultaneously synthesizing vector and image data* from example data. In the context of urban layouts, synthesizing vector data is accomplished by capturing the properties of an existing arrangement of streets and producing similar ones. Synthesizing image data consists of generating new images similar to the original but able to populate the new street structure. Our strategy couples vector-based and image-based synthesis and uses a by-example methodology to enable intuitive high-level operations for easy urban layout modification and creation. While the objective of this paper is urban layout creation, our methodology can be applied to other types of structured images as well.

Our approach interactively synthesizes a new urban layout by creating and/or joining fragments of urban layouts based on a set of pre-existing examples. Each *urban layout fragment* consists of both structure and image data for a connected subset of the layout and can range from a small neighborhood to an entire city. The structure component of an urban layout fragment is represented by a hierarchical organization of the streets. The hierarchy follows a Gravelius ordering, traditionally used for plant modeling [Holton 1994], where the longest and most important streets (branches) are labeled level zero and the streets of each successive branching level are labeled as of the next level. With the added ability of supporting loops, this representation fits naturally with many street patterns. The image component of an urban layout fragment is represented by a collection of aerial-view images, which are registered with the street data. To perform synthesis, we use the aforementioned organization of structure and image data to enable easily (i) creating new urban layout fragments based on information from one or more examples, (ii) constraining the synthesis of a new urban layout to a pre-existing structure or to user-specified constraints such as a sketch of a city, and (iii) joining multiple urban layout fragments together to form a larger urban layout without the user having to be concerned about structure and image details. Altogether, our approach is able to succinctly represent a wide variety of urban layouts fragments and to intuitively and quickly create and/or modify complex structured images of urban layouts. We have used our approach to create and modify large layouts based on fragments from several real-world locations extracted from a Geographical Information System (GIS)-style database (Figure 1).

The main contributions of our work are

- *a structure synthesis algorithm* to create the streets and parcels of a new layout fragment based on input examples,
- *an image synthesis method* to generate aerial-view imagery for populating new layout fragments by reusing and warping image data from a set of input examples, and
- *a layout synthesis system* providing a set of interactive tools for assembling a new layout by seamlessly joining example layout fragments and for creating a new layout by blending between the styles of one or more example layout fragments.

System Overview

The system takes as input fragments from one or more existing urban layouts and interactively builds up a new layout. It also uses a parameterized method to synthesize a new fragment based on one or more examples and supports combining multiple new fragments in order to create a larger layout. Each example fragment is automatically computed from GIS vector data describing the street network and parcels and from registered aerial images.

Fragment synthesis consists of two parts: structure synthesis (Section 3.1) and image synthesis (Section 3.2). In structure synthesis, our street generation algorithm connects street intersections and generates a new street network that exhibits the attributes encoded in the intersection points. Parcels are then automatically created inside the city blocks defined by the new streets. In image synthesis, new parcels are filled with aerial imagery, warped from similar parcels of example layout fragments.

To generate a new layout, the user interactively copies and places groups of attributed intersection points in the region where a new layout is to be synthesized (Section 4). Three different high-level synthesis operations are defined between example layout fragments: join, expand and blend. To assist with the synthesis operations, the attributed street intersection points of each layout fragment are placed into an adaptive spatial partitioning data structure. Each leaf node of this data structure corresponds to a portion of a layout fragment that has similar spatial attributes and is considered an atomic unit.

The example layout fragments are automatically computed from GIS vector data and from aerial imagery (Section 5). For each example fragment, the system calculates the intersection points between the streets, and encodes a characterization of the geometry of the streets and parcels as attributes of the intersection points.

2. RELATED WORK

Our approach builds on concepts from example-based techniques and relates to methods in procedural modeling and image-based modeling. In the following sections, our work is described and compared to activities in these major areas.

2.1 Example-based Synthesis

Example-based approaches have a long history in graphics for they provide a fast way to generate complex results by borrowing from existing data. Texture synthesis exploits the regularity of patterns that appear in example texture fragments to create more texture (e.g., [Kwatra et al. 2005]) or terrain (e.g. [Zhou et al. 2007]) of a similar, potentially very complex, appearance. Wei et al. [2008] proposed inverse texture synthesis as a way to synthesize a compact representative texture from which a non-regular texture can be synthesized. Further, several content-aware image editing techniques propose algorithms to change existing textures or images. Avidan and Shamir [2007] describe approaches to resize and retarget images whereby the visually important information is maintained. Fang and Hart [2007] propose a technique to re-synthesize image texture when reshaping a textured portion of an image and to reduce unwanted deformations. Kim and Pellacini [2002] introduce an image mosaic algorithm whereby images of objects are composed together to form the final picture of an arbitrary shape. Structured data has also been generated by example. For instance, Funkhouser et al. [2004] and Sloan et al. [2001] developed systems to generate 3D models by using multiple example objects. Nevertheless, none of these methods are concerned with synthesizing both structure and image data.

Also relevant to our work are some systems that have edited images of urban layouts. In particular, Hertzmann et al. [2001] describe a framework for processing images by example. While they modify aerial photographs of cities they do not maintain or produce any underlying street and parcel information. Previously, we have described a constraint-based system for modifying urban layouts [Aliaga et al. 2008]. Parcels and streets are interconnected such that moving the street changes the parcel and vice versa. While this prior method is aware of the structure of an urban layout, it provides no tools to synthesize the structure (e.g., streets) of new layouts, no methods to synthesize the imagery for new layouts, and no mechanism to expand an existing layout.

This paper extends example-based methods to consider both structure and image data. Similarly to Ijiri et al. [2008], we enable growing a layout by pieces, but our approach deals with large geometrical structures and not only with an immediate neighborhood. We also generate content tuned to urban layouts and its associated imagery producing not only high quality visual imagery, but also a feasible network of streets and parcels.

2.2 Procedural Modeling

One of the first procedural techniques describing 3D city generation was introduced in [Parish and Mueller 2001]. It is based on both L-systems [Prusinkiewicz and Lindenmayer 1996] and Open L-systems [Měch and Prusinkiewicz 1996]. A similar approach for the template-based street network generation is that of [Sun et al. 2002] which also uses a rule-based rewriting system. Procedural methods have also been extended to the synthetic generation of buildings [Mueller et al. 2006, Wonka et al. 2003] and to the creation of buildings and façades imitating real-world structures [Mueller et al. 2007]. Nevertheless, these approaches require the explicit specification of a set of rewriting rules and/or the model is generated at once which limits editing.

Recently, Chen et al. [2008] proposed using tensor fields to guide the generation of street graphs. While the user controls editing via high-level parameters, it is challenging to steer the tensor fields to connect to and extend existing urban spaces and to blend between different existing street network styles. Further, no parcels or aerial-view images are produced. In contrast, our goal is to produce closed street networks, parcels, image content, and to provide higher-level example-based editing.

2.3 Image-based Modeling and Computer Vision

The efforts of image-based modeling and computer vision are focused primarily on the creation of geometric models, typically from photographs and/or laser-data, and combining the geometry with the images. Numerous reconstruction methods have been proposed using aerial imagery (e.g., [Vestri and Devernay 2001; Ribarsky et al. 2002]), ground-level imagery, or combinations of both (e.g., [Frueh and Zakhor 2003]). In addition to extracting the global urban structure, it is difficult for these methods to segment the data into all of streets, parcels, and building structures. Regardless, the approaches do not typically address the efficient modification of the captured geometry and of the urban configuration. In contrast, our focus is precisely on enabling the intuitive design of urban spaces and layouts in the style of provided exemplary real (or imaginary) urban spaces.

3. FRAGMENT SYNTHESIS

Fragment synthesis creates both the structure and image content of individual fragments which will later be combined to form new and larger layouts. Our approach to synthesizing individual

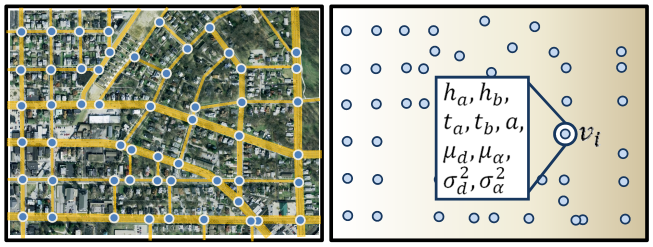


Figure 2. Streets and Intersection Points. Left: Set of streets (yellow) and street intersection points (blue) which form part of an example fragment. Right: The characteristics of the street network can be encoded as attributes of the intersection points.

fragments is inspired by the following observations: (i) a compact approximation of the geometrical structure of a layout can be represented by a spatial distribution of attributed street intersection points where the attributes include a characterization of the street geometry and parcel sizes in the vicinity of each intersection point (Figure 2), and (ii) the aerial-view images of an existing urban area are naturally partitioned by the street and parcels into small image fragments which can be warped and re-used as imagery of newly synthesized layout fragments. Based on these observations, we describe a parameterized synthesis algorithm to generate new geometrical structure from a spatial distribution of attributed intersection points and a method to generate new layout imagery from a set of aerial-view images of existing urban areas (Figure 3).

Layout fragment synthesis will create and populate several data structures that collectively define a new layout fragment $L = (S, B, I)$, where S is the street network, B is the set of city blocks and parcels, and I is the set of aerial images. *Structure synthesis* uses a set of 2D street intersection points $V^S = \{v_0, v_1, \ldots, v_{|V^S|}\}$, each with associated attributes, to generate two undirected attributed graphs: $S = (V^S, E^S)$ storing the synthesized street network, and $B = (V^B, E^B)$, storing the corresponding city block and parcel information. *Image synthesis* uses image fragments from a set of input aerial-view images to generate a new set of image fragments I for use by the new layout fragment. Our system, does not explicitly cut out image fragments from the original images, but rather stores the original images and uses texture coordinates to index the image data.

3.1 Structure-based Synthesis

Synthesizing the structure of a new urban layout fragment proceeds in two steps: First, the street graph $S = (V^S, E^S)$ is completed by joining the intersection points V^S with edges E^S created by a random walk based algorithm. Second, the city-block graph $B = (V^B, E^B)$ is derived from S (B is the dual graph of S) and captures the structure and adjacency of the city blocks. The city-block graph is then used to generate individual parcels (and later aerial-view images) within each city-block.

3.1.1 Street Generation

The street generation algorithm synthesizes a network of streets that attempts to follow the layout style implied by the attributed intersection points V^S (Figure 4). We define a street intersection point $v_i \in V^S$ to be between two crossing streets – crossings of more than two streets would have to be modeled by a collection of street intersection points. The algorithm performs directed random walks through the points in V^S. The initial steps connect intersection points so as to form streets of hierarchy h_0 (e.g., highways and avenues). In subsequent steps, the connectivity of the less important streets of levels $h_1, h_2, \ldots, h_m$ is produced. While there is no guarantee the hierarchy levels correspond one-to-one

```
$(S,B,I')$ = synthesize($V,I$)
/*Input: Attributed intersection points $V$, Imagery $I$.
  Output: Synthesized streets $S$, blocks and parcels $B$,
          and new imagery $I'$ */
$S$ = connectIntersectionPoints ($V$)
$B$ = computeBlocksAndParcels ($S$)
$I'$ = generateNewImagery ($I,B$)

connectIntersectionPoints ($V$)
REPEAT
    FOR each hierarchy level $h = h_0, h_2, \ldots, h_m$
        FOR each street $s$ of hierarchy level $h$
            randomWalk ($s,V$)
        ENDFOR
    ENDFOR
UNTIL no new street segments are added to any street

randomWalk ($s,V$)
FOR each of the two endpoints $v_e$ of street $s$
    Find the point $v^* \in V$ that maximizes $p(v_e,v)$
    IF $p(v_i,v^*) > 0$ THEN
        Add segment $(v_e,v^*)$ to $s$
        Update endpoints of $s$
    ENDIF
    /* if $p(v_e,v^*) = 0$ no feasible transition points
    have been found (see figure 5a)*/
ENDFOR

computeBlocksAndParcels ($S$)
Extract blocks as minimum cycles formed by streets $S$
FOR each block $b$
    Generate a Voronoi diagram from a set of points
       sampled along the main axis of the OBB of $b$
    Each Voronoi region enclosed in $b$ is a parcel of $b$
ENDFOR

generateNewImagery ($I,B$)
FOR each block $b$
    FOR each parcel $l$ of $b$
        Find in $I$ an image fragment $i$ with similar
            shape to the contour of $l$
        Generate a quadrilateral grid to produce a
            one-to-one mapping between the $i$ and $l$
        Warp $i$ into $l$ using the one-to-one mapping
    ENDFOR
ENDFOR
```

Figure 3. Pseudocode. Summary of synthesis algorithm.

with actual street importance, our process of using hierarchy levels has proved to be suitable for layout synthesis.

The attributes associated with each point include:

(i) the intersection hierarchy level $h_a(v_i)$ and $h_b(v_i)$ of the two streets that could be created through v_i (used for generating street connectivity),

(ii) the average tortuosity (e.g., ratio between the length of the curve segment and the distance between its endpoints) $t_a(v_i)$ and $t_b(v_i)$ of each of the streets incident to v_i (used for generating the geometry of a street segment),

(iii) the average area $a(v_i)$ of the parcels in the vicinity of v_i (used for generating city-blocks), and

(iv) the statistical characterization of the two streets that could pass though through v_i (used for generating street connectivity). This characterization consists of four values per street: the mean μ_d and variance σ_d^2 of the distance between two consecutive intersection points, and the mean μ_α and variance σ_α^2 of the angle between two consecutive street segments.

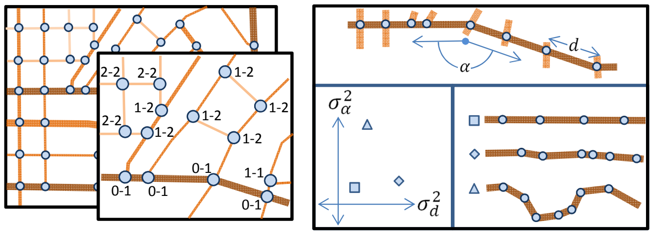


Figure 4. Attributes. Left: The hierarchy levels of the example streets are calculated and stored in each intersection point they generate. Right: A statistical characterization of each street is obtained by calculating the mean and variance of the distance between points and of the angle between segments.

A step in the random walk for street generation consists in moving from a base point v_i to one of the target points v_j, and connecting the points with a new street segment. The target point is randomly selected by using the transition probabilities $p(v_i,v_j)$ calculated from a bi-variate Gaussian distribution defined by the mean (μ_d, μ_α) and variance (σ_d^2, σ_α^2) stored in the base point v_i (Figure 5a). Intuitively, this stochastic process prefers a target point v_j with the following two properties: the distance from v_j to v_i is closest to the mean distance μ_d between intersection points and the angle formed between the proposed segment (v_i,v_j) and the last existing segment of the street is most similar to the mean angle μ_α between consecutive segments. A user-defined threshold is used to ignore segments of very low transition probability.

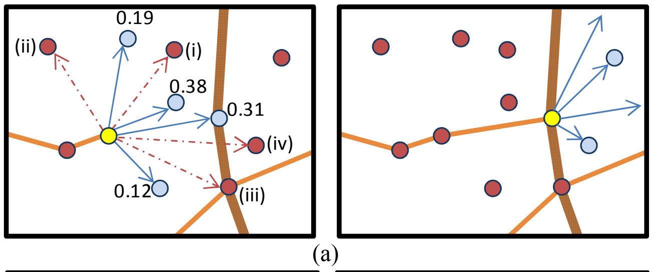
Performing a step to an intersection point is subject to constraints imposed by the hierarchy levels stored in the points, by the number of times an intersection point has been connected, and by previously generated street segments. A step of the random walk on a street of hierarchy level h_k can move to a point v_j if and only if $h_k = h_a(v_j)$ or $h_k = h_b(v_j)$. We assume that only two streets pass through an intersection point. Furthermore, we do not allow generation of new intersection points in the street generation phase. Consequently, a newly generated street segment cannot cross an existing street in an arbitrary position, but only in an intersection point. In all three constraint cases, we set to zero the transition probability from v_i to an invalid point v_j, i.e., $p(v_i,v_j) = 0$.

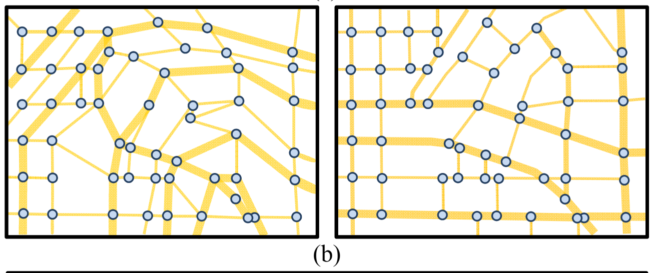
A random walk on a hierarchy level h_k and through v_i ends when for all v_j we have $p(v_i,v_j) = 0$. In this case, a new random walk is started from an available connection slot in a point v_i with $h_a(v_i) = h_k$ or $h_b(v_i) = h_k$. The process ends when no new random walks can be started.

Since the street segments between intersection points in the example layout fragments are not necessarily straight lines, we use the tortousity attributes of the points defining the segment e to generate a poly-line representing the geometry of the segment (Figure 5c). The poly-line mimics the style of the example streets and is stored as an attribute $g(e)$ in the corresponding edge.

3.1.2 City-Block Generation

Once the street generation is completed, we compute the blocks and parcels enclosed by the new streets yielding the city-block graph $B = (V^B, E^B)$. Each point $v \in V^B$ corresponds to a city block and forms a closed face of S (i.e., a closed cycle). Two city blocks b_a, b_b are adjacent if there is a corresponding street edge $e \in E^S$ in common for both faces. When this is the case, we add edge $f = (b_a, b_b)$ to E^B. Hence, the city block graph B is a dual graph to the street graph S as every face of S corresponds to a vertex in B. The geometry of each new block $b \in V^B$ is computed





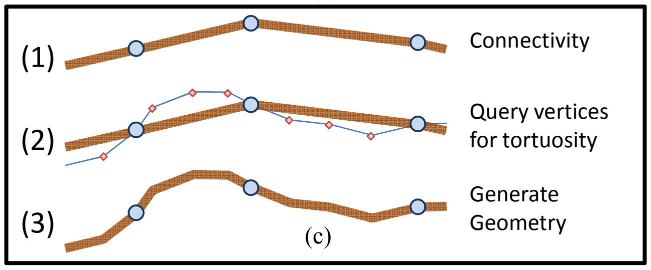


Figure 5. Street Generation. a) A transition probability for the random walk is associated to each intersection point based on its attributes. A point is not considered when (i) it cannot have the current hierarchy level pass through it, (ii) it is statistically too improbable for a street to pass through it, (iii) it has been traversed by a street the maximum number of times (i.e., 4), or (iv) it requires crossing an edge to reach it. b) A same point set with different attributes can generate two street networks with significantly different styles. c) Once the street segments are created, its street poly-line is computed using the tortuosity parameter stored in the intersection points.

by concatenating $g(e_i)$, for every edge e_i that forms the polygon enclosing b. An edge $e \in E^S$ is part of the polygon enclosing b if at least one of the edges from E^B that is incident to b intersects e.

Each block b is then subdivided into parcels $L_b = \{l_0, l_1, \dots, l_{n_b}\}$. The number of parcels per block n_b and the expected aspect ratio are determined from the average parcel area stored in the points of the edges enclosing b. Our block subdivision algorithm is guided by the following observations and assumptions:

- We assume all parcels have road access (egress rule).
- A block is subdivided into one or two rows of parcels, such that each parcel is touching the street in the front and touching a neighboring parcel in the back (and maybe on the sides); sometimes parcels have street access both in front and in back – this implies one row of parcels and often means very few parcels in that block; corner parcels, of course, have access on consecutive sides of the parcel.
- While parcels may be of any shape (and not necessarily convex), very often parcels can be well approximated by polygons of few edges and most often of four edges.

The general idea for subdividing a block b into n_b parcels is to sample the inside of the block with a set of points P_V, and then to

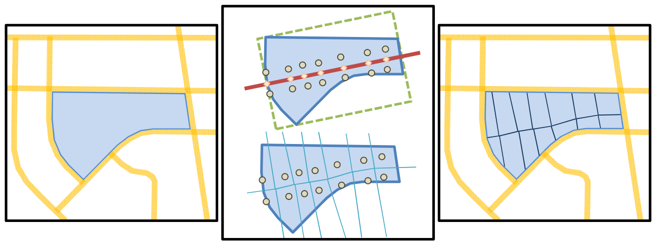


Figure 6. Parcel Generation. Left: A block is extracted from a synthetic street network. Middle: The main axis of the oriented bounding box of the block is calculated and stochastically sampled with points (up). Copies of the points are created on each side of the axis, and their Voronoi diagram is calculated (bottom). Right: The intersections of the Voronoi diagram with the contour of the block are used to help define the parcels.

calculate the Voronoi diagram of the points in P_V (Figure 6). Each resulting Voronoi region will constitute a parcel $l \in L_b$. Results have shown that if P_V is adequately selected, parcel geometries that closely resemble the parcels in the original city can be obtained.

Voronoi points P_V are generated along an axis that approximately splits the block into two halves. We have observed that for most blocks this axis corresponds closely to the central segments of the medial axis of the block. Since most blocks are nearly rectangular, the medial axis can be approximated by the longest axis of an oriented bounding-box (OBB) surrounding the block. The segment of the axis that lays inside b is then sampled with $(n_b - 1)/r_b$ points, separated by a distance $d + \epsilon$, where d is deduced from n_b and r_b, ϵ is a Gaussian noise (i.e., $\epsilon \sim N(0, \sigma_b^2)$) added to increase the diversity of the parcels inside b, and r_b is the number of rows in the block (i.e., 1 or 2). The variance σ_b^2 is computed from the standard deviation of the parcel area averages stored with the intersection points associated with b. For a single row of parcels inside the block, the Voronoi vertices are the $n_b - 1$ sampled points. In the case of two rows, the Voronoi vertices are given by creating two copies of the $(n_b - 1)/2$ sampled points, and translating them a distance $\pm\delta$ along a direction perpendicular to the central axis, where δ is a small positive number.

3.2 Image-based Synthesis

Synthesizing the image data for a new urban layout fragment uses a method for establishing polygon-similarity and an image-warping algorithm to generate plausible image content for each newly created parcel. Since our urban layouts are created by-example, it is natural to assume we can reuse some of the image content of the examples to populate the newly created layout fragment. Since both the input and created layout fragment are highly structured, we cannot resort to simply copying large blocks of pixels. Instead, for each new parcel l_N whose geometry is an a-sided polygon, we find a best parcel l_O of the example layout fragment whose geometry is a b-sided polygon, and map the polygon of l_O to the polygon of l_N. The selected example parcel image fragment is warped via texture-mapping onto the new parcel geometry and rendered together with the rest of the layout fragment.

To find the example parcel l_O most similar to the new parcel l_N we compare the OBB of all example parcels with the OBB of l_N. We let l_O be the parcel whose OBB is most similar in area and aspect ratio to the OBB of l_N and respects the same side facing the street.

Our image-warping algorithm generates a correspondence between two arbitrary concave or convex polygons. First, it defines a $s \times n$ quadrilateral grid inside the OBB of l_O and the OBB of l_N, such that a one-to-one mapping can be established between the grids of

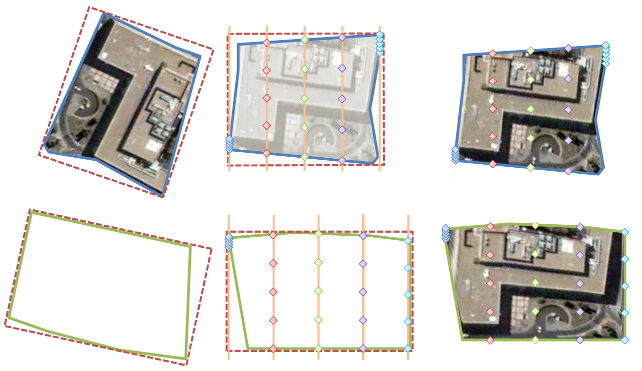


Figure 7. Image Warping. Given a blank, new synthetic parcel (bottom left), a similar parcel with image (source) is found (top left). A one-to-one correspondence is found between sample points on both parcels (middle), so that the image of the source parcel can be warped to the new parcel (bottom right).

l_O and of l_N. To do so, the algorithm defines for l_O and for l_N a set of s scan lines that are uniformly separated, and perpendicular to the main axis of the OBB of each parcel. Then, the method finds the intersection points p_a, p_b between each of the s scan lines and the contours of l_O and l_N, and generates n uniformly-separated sample points along each scanline and between p_a and p_b.

A total of $s \times n$ sample points are generated inside each parcel, and a one-to-one correspondence is therefore guaranteed. At display time, the new parcel l_N is rendered using the coordinates of the correspondence points to define a quadrilateral mesh, and the coordinates of the corresponding example parcel to obtain texture coordinates (Figure 7). This results in the example texture being warped onto the new parcel. The similarity metric and presence of many parcels of different shapes and sizes tends to keep the texture stretch small [Zhang et al. 2005]. This method will produce a C^0 continuous and non-self-intersecting warp if the projection of both parcel's contours onto the main axis of their OBBs is monotonic.

4. LAYOUT SYNTHESIS

Using the fragment synthesis algorithm, our approach enables a user to interactively assemble new layouts by combining example layout fragments. Choosing the location and attributes of each intersection point in the new layout is a difficult and time consuming process. Thus, we organize the point sets of each example layout fragment into an adaptive spatial partitioning that facilitates manipulation of the intersection points in a fast and intuitive way. Given the spatial data structure, a new layout is assembled by a sequence of layout synthesis operations. The user simply copies and pastes together the point sets of example fragments from which the system synthesizes a new and complete layout. The point sets can be placed next to each other with no overlap or with overlap. The former corresponds to the intuitive notion of assembling pieces of a puzzle. The latter is used to represent a blending of the styles of the corresponding original example fragments. In both cases, the user is sketching the new layout by only manipulating point sets and does not need to be concerned with the structure and imagery. Therefore, arbitrary layout fragments can be easily and interactively assembled together. The algorithms presented in Section 3 are used to connect the new intersection points with street segments based on their attributes, to populate the resulting street network with parcels and imagery, and finally to yield a new synthetic urban layout.

4.1 Organizing Intersection Points

The attributed intersection points V^s of each example fragment are organized into a spatial partitioning that divides the points into contiguous regions of approximately similar attributes. The intersection points of an arbitrary example fragment may have significantly varying attributes (e.g., varying spatial densities, different distributions of hierarchy levels, etc.). This may result in a strong dissimilarity between the intersection points of two layout fragments to be pieced together when creating a new layout by example. The dissimilarity hinders providing a smooth transition between the structures of the two fragments. To automatically synthesize a transition from one fragment to another, we partition each layout fragment into spatial nodes containing points of similar attributes. Intuitively, we seek to partition the layout fragment into areas that are each approximately self-similar; e.g., a downtown area containing a regular grid of intersection points, a housing subdivision containing a particular style of intersection points, etc. This in turn enables modeling each node using a simple set of stochastic parameters and enables parametrically producing attributed intersection points that can smoothly change from the attributes of one fragment to those of another fragment. Altogether, this facilitates defining a set of attributed intersection points that can either (i) exhibit a smooth spatial transition from the structure of one fragment to that of another adjoining one, or (ii) have attributes that are a weighted combination of the attributes from two or more layout fragments.

Instead of using a regular grid of boxes that would hardly each have similar attributes, our approach is to partition the set of attributed intersection points for each example layout into a quad-tree data structure, whose leaf nodes are rectangles containing a set of points of approximately uniform spatial distribution. Moreover, each leaf node also stores the relative probability of each hierarchy level within the confines of the associated quadrilateral. Thus, the quad-tree captures both the distribution of the locations of the intersection points and the spatial distribution of the hierarchy levels within the layout fragment. To construct the quad-tree, the OBB of the set of intersection points from an example layout is recursively subdivided until each leaf node has (i) a set of roughly uniformly-distributed intersection points, (ii) a minimum area, or (iii) a minimum number of intersection points. While other point distributions can be used as a model (e.g., 2D Gaussian distribution), an assumption of uniform density enables us to capture a grid-like configuration of intersections, which would be difficult with a Gaussian distribution.

4.2 Synthesis Operations

In subsequent by-example processing, the attributed intersection points of example fragments and their aforementioned spatial organization are used to enable several synthesis operations: *join, blend,* and *expand*. Without loss of generality we explain the operations for two example point sets L_a and L_b. The point sets can be joined by simply placing copies of their attributed intersection points next to each other and then the synthesis algorithm will connect the points to produce a new combined layout. In order to provide a smoother structural transition, the point sets should overlap. Then, we define a spatially varying blending operation within the overlap area. The points nearer to L_a will show a style more similar to that of the layout from which L_a was obtained; vice versa for L_b. An interesting variant of blending is that if the point sets overlap completely, we can define a constant value over the entire overlap region that serves to "interpolate" between the styles of the two sets. Finally, the same methodology is used to expand an existing layout. The user sketches the expansion by placing points (e.g., L_a or L_b) next to an existing or synthesized layout.

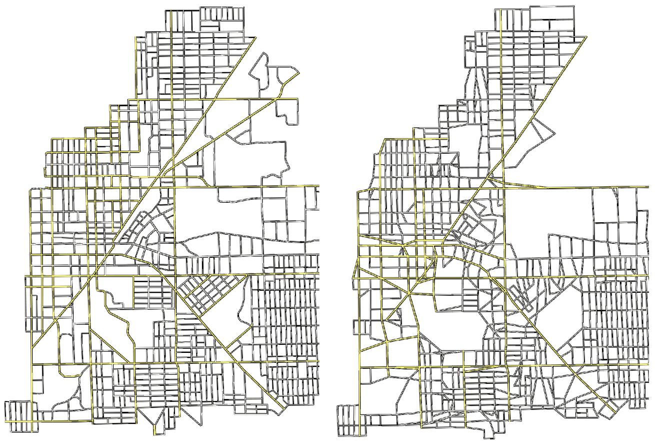


Figure 8. Abstraction into Attributed Points. The attributed intersection points of the street network of an example urban layout fragment (left) have been copied and dropped in an empty area (right). Our system is then used to connect intersection points. Notice the similarity between the reconstructed street network and the original one.

The synthesis algorithm connects the new intersection points to the existing street network. Combining these operations allows for significant expressivity in designing urban layouts by example.

5. OBTAINING EXAMPLE FRAGMENTS

Example fragments for use in our synthesis process are obtained by automatically calculating the attributes of a set of existing street intersection points. GIS databases are widely available for many cities and thus are a good source of data for computing example fragments. For our purpose of urban layout synthesis, we assume a GIS database contains a set of unorganized piecewise linear curves (poly-lines) describing street centerlines, a description of the individual parcels of land, and a set of geo-registered aerial-view images. The aerial-view images are directly used by the image synthesis steps of a new layout. Although GIS databases can be straightforwardly used for simple aerial flyovers or for querying for a particular street, city block, or route to follow (e.g., map directions), they are not adequately organized for the structure-based synthesis steps of urban layouts. Nevertheless, they do contain sufficient information to derive the intersection points and attributes necessary for our structure-based synthesis.

The set of intersection points V^S of a real-world layout fragment is constituted by all the pair-wise intersection points between street centerlines, and by all street centerline endpoints. The set of intersection points by themselves (i.e., with no additional attributes) does not capture the "style" of the layout; for instance, two layouts may have the same set of intersection points but very different street patterns (see Figure 5b). Thus, we calculate the set of attributes (as mentioned in Section 3.1.1) using the GIS database. To calculate the intersection point hierarchy levels, we select a small percentage of the longest and widest streets of the layout as the main branches and set their level to h_0. Any streets crossing them are defined to be of the next level of branches, e.g., $h_0 + 1$. This process is repeated until all streets are labeled, and it exploits the notion of first labeling the longest and widest streets and then recursively labeling the smaller roads. For the hierarchy levels $h_1, h_2, \ldots, h_m$, the streets of level h_i will be all those that intersect at least one street of level h_{i-1}, and that have not already been labeled with another hierarchy level. Since the maximum number of hierarchy levels m is not known in advance, h_0 represents the highest hierarchy level, and h_m the lowest. The remaining attributes of average tortousity, average parcel area, and the statistical characterization with the mean and variance of the distance and angle between two consecutive intersection points are easily computed from the GIS street centerlines and parcel data.

6. RESULTS AND DISCUSSION

We have used our approach to interactively create several synthetic urban layouts from fragments of real-world cities. We first demonstrate how our attributed point sets capture the structural characteristics of a real-world layout fragment. Then, we show the results of several operations used to compose layouts by example. All street networks and images are interactively generated by our system using the example fragments.

We have chosen parts of four cities from around the world, each of them exhibiting a different style of street network and from which multiple example layout fragments are extracted: Madrid, Istanbul, Buenos Aires, and Lafayette, IN. The sizes of the individual example layout fragments we extract from our set of cities vary from a few streets, blocks, and parcels to several hundred streets and blocks and several thousand parcels. The imagery was obtained from Google Maps. Although the needed street data can be obtained from GIS databases, the data of our examples was obtained using a simple interactive editor except for Lafayette which was provided by the town's urban planning division.

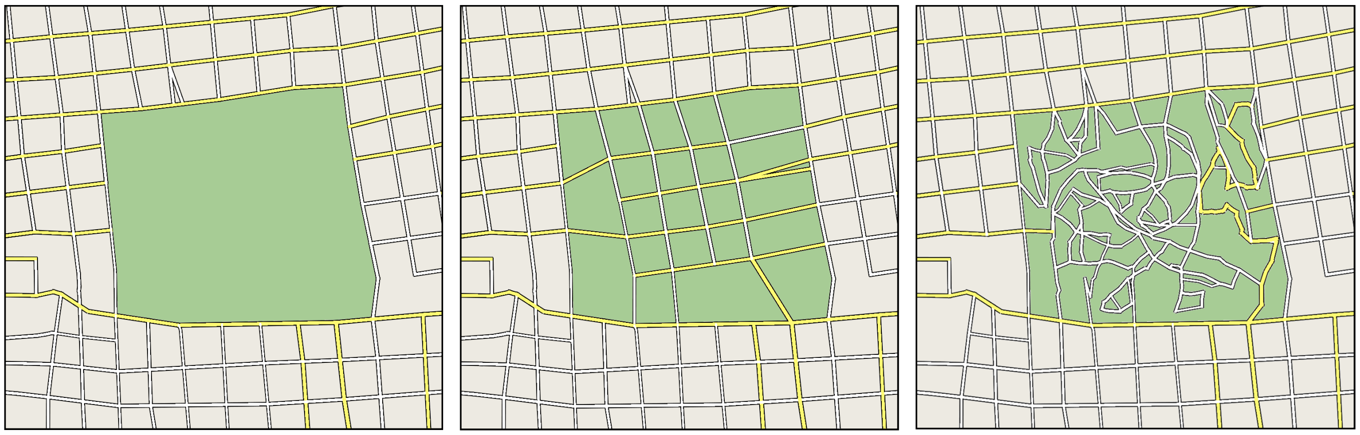


Figure 9. Expansion. An urban layout (left) with grid-pattern blocks grows into an empty area inside the layout that has been filled with attributed intersection points taken from a fragment of a grid city (middle), and attributed intersection points taken from the fragment of an irregular city (right). In both cases, new streets are generated which exhibit the style encoded in the intersection points.

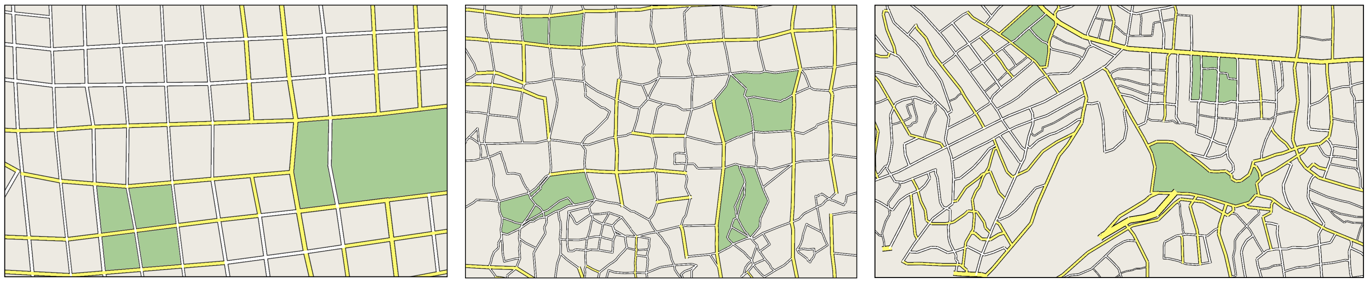


Figure 10. Blending. The user takes attributed intersection points of two fragments (left, right) from cities with different styles. Our quadtree-based approach modifies the points so that they encode a spatially varying (North-South) blend of both styles. A new street network is created by our street generation algorithm that intuitively resembles a merge of the example street networks (middle).

First, we show how a set of attributed intersection points captures the style of an existing urban layout (Figure 8). We used our program to create a copy of the attributed intersection points of Lafayette. This is interactively accomplished via a single mouse-dragging and pasting operation. The copied intersection points are automatically connected in a way that closely resembles the layout of the original city. Notice for instance how the main (longest and widest) streets in the synthetic city are almost identical to those of the example city. The errors in the recovered network are due to the presence of highly atypical streets and to the stochastic nature of our algorithm. While this particular result could also be accomplished by a simple copy and paste, it serves to show the effectiveness of our representation in capturing the style of a city using a mostly stochastic process.

To compose larger layouts, we demonstrate several types of synthesis operations. Figure 9 shows an urban layout (left) with grid-pattern city blocks (e.g., Buenos Aires) growing into an empty area inside the layout (e.g., a park). First, the user has placed a set of attributed intersection points taken from a similar grid-pattern city (middle). Street segments are automatically generated to connect the points in a way that closely resembles the style of the original city. In a second experiment (right), the user has placed a different set of attributed intersection points copied from a city with a more irregular street pattern (e.g., Istanbul -- see right side of Figure 10). Again, street segments exhibiting a style similar to that of the example city are created to connect the intersection points. In this latter case, there is a clear (but rough) transition in the style of the street network. Notice the higher tortuosity of the new streets and the small angles at many of the intersection points. In both cases, new streets are reasonably connected to the original street network.

A result of a spatially varying blending operation is shown in Figure 10. Example layout fragments are extracted from urban areas of different styles: one exhibiting a grid-pattern (left), and one exhibiting an irregular pattern (right). A new street network is generated as a combination of both patterns (middle). The blending value provided by the user defines the attributes of the upper points to be more similar to those of the intersection points in the grid pattern, and a different blending value defines the attributes of the lower points to be more similar to those of the points in the irregular pattern. On a large scale, the blending of both styles is visible from a view of the entire new layout. On a small scale, the blending is also seen in details such as the higher variance in the distance between intersection points at the bottom of the layout (similar to the irregular pattern), and the lower tortuosity of the street segments at top of the layout (similar to the grid pattern).

Figure 11 shows the result of a sequence of join and blend operations. The original street network of an existing irregular pattern city is loaded by the user. A copy of the set of attributed intersection points of a grid-pattern city is then placed to the northwest (left) of the original layout. Next, a copy of the intersection points of both the irregular pattern city and the grid pattern city are placed in the space between the original city and

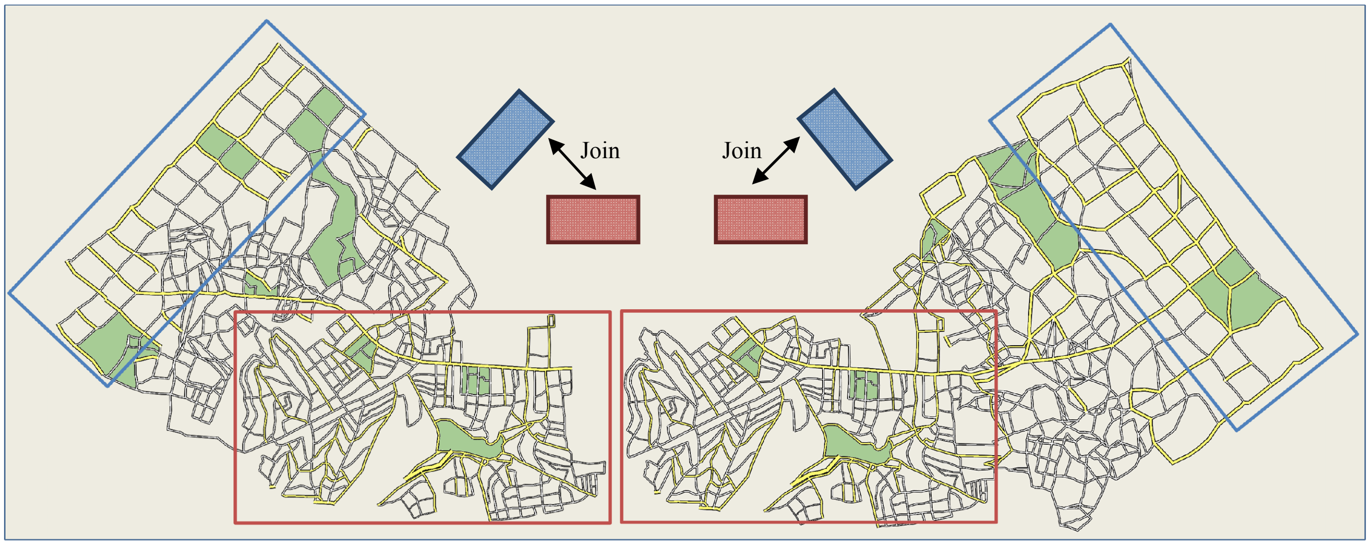


Figure 11. **Join and Blend**. Sequences of join and blend operations have been used to generate both street networks. In this case, the blue fragment (with grid pattern) and the red fragment (with irregular pattern) are connected in two different positions relative to each other.

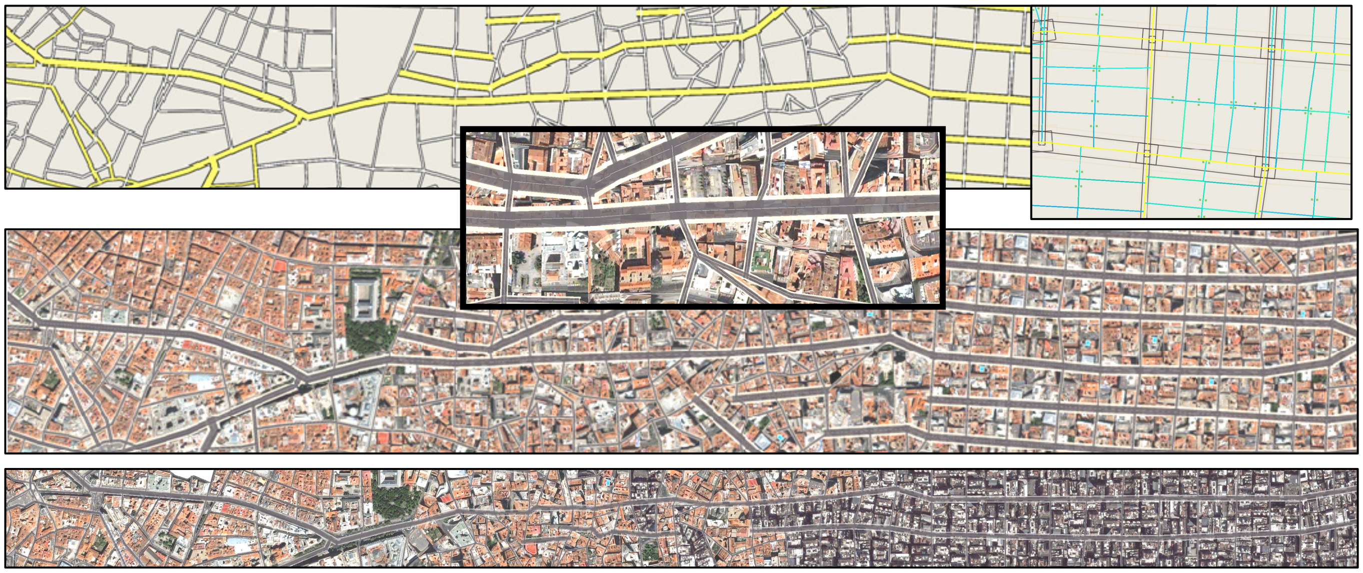


Figure 12. Larger example using multiple operations. (top) A fragment with irregular pattern (left) is joined together with a fragment with grid pattern (right) by a street network resulting from a spatially varying blending operation. (middle) Parcel imagery is generated by example from fragment on left side and (bottom) as an interpolation of imagery from both fragments.

the points dropped in the previous step. The system automatically calculates a new street network that joins the original city with the grid pattern points and that exhibits a blended style.

Figures 12 and 13 present additional examples. Figure 12 is an example of two layout fragments exhibiting irregular (left) and regular (right) patterns that have been joined together. The street network is the result from a spatially varying blending value, yielding a smooth transition between the two layout fragment styles. The parcel imagery is also interpolated between both cities, i.e., parcels towards the left side of the blend region used more textures from the irregular pattern city, while parcels on the right side used more textures from the regular pattern city. The blend parameter is used to modulate a simple probability value that determines which layout is searched for the best fitting parcel image. Such interpolations can be used to easily assemble a city from a variety of layout fragments and associated imagery. Figure 13 shows an expansion of a city producing both structure and images. Observe how a similar style of streets is produced and the seam between the original and new blocks is hard to see.

7. CONCLUSIONS

We have presented an interactive system to synthesize new layouts by example. The structure and image data of example layout fragments is separated. Then, both a structured-based synthesis and an image-based synthesis are performed resulting in a new synthetically created urban layout. Our core synthesis algorithm can be used to perform several high-level editing operations such as joining, expanding, and blending example layout fragments. The last option, blending, provides a powerful, yet intuitive, way to synthesize new layout fragments by blending between the styles of several examples. All operations are fully interactive except for parcel image generation which in our current implementation may take from seconds to several minutes. Nonetheless, new and complete layouts with plausible street networks and aerial imagery can be quickly generated without the user being concerned about low-level details. Our approach is useful to a variety of applications for virtual environments, entertainment, architecture, emergency management, and urban planning.

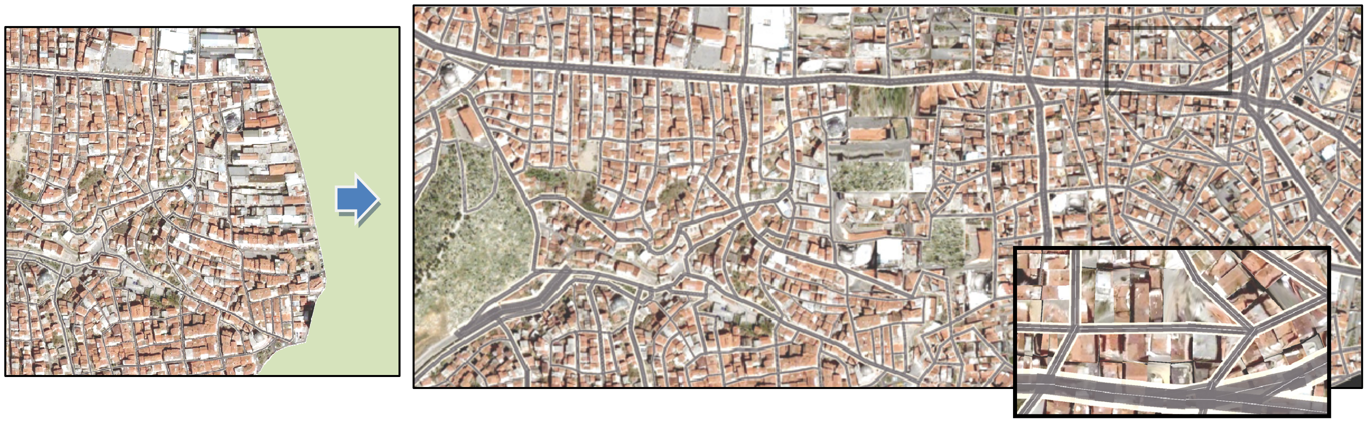


Figure 13. Expansion with Images. A layout fragment with an irregular pattern is automatically expanded to the East. New streets are connected to the example fragment and exhibit a similar style as the streets of the fragment. Imagery is also re-used to populate the new street network. Notice the difficult-to-see transition between the original fragment and the new streets and parcels.

Limitations. We have identified three limitations in our synthesis algorithm. First, the parcel image generation process assumes there is sufficient variety of parcel shapes and sizes available to fit in the synthetically created parcels. This results from our image-reuse strategy. If the provided input layout is too small or has a small variety of parcel shapes, it will be difficult to find good matches for parcel filling without severe image warps and high parcel image repetition. As our system is applied to larger areas, this issue becomes progressively less problematic. Second, while parcel images are selected from existing areas of the same general classification (e.g., zoning), contextual information is in general not used to select and warp parcel images. This is not strictly a limitation, but it may cause unwanted image warping. Third, without user editing, our method is not well suited for reproducing highly atypical streets of an example layout fragment. Since our system relies on street intersection points, an atypical street will have relatively few intersection points that capture its style and a street network different to the original one might be produced. An example of this case are the odd-looking streets synthesized from the duplicated intersection points of the streets with atypical orientation, near the center of the original urban layout of Lafayette (Figure 8). In these cases, the user would need to correct the generated street network.

Future Work. We are pursuing several items for future work. First, we would like to extend our approach to also consider the underlying terrain geometry. Thus in addition to characterizing the street network on a plane, changes in the terrain height should affect growth patterns. Second, a natural extension is to further produce synthetic content within each parcel. One option is to use a procedural modeling engine to generate 3D content within each parcel (e.g., [Aliaga et al. 2007, Mueller et al. 2006, Wonka et al. 2003]). A third direction of future work is to integrate our system with an urban simulation (e.g., [Waddell et al. 2007]) engine to provide automatic high-level control to guide the urban expansion. Furthermore, we could develop a feedback loop to generate visualizations of the urban spaces during a simulation.

8. ACKNOWLEDGEMENTS

This work is supported in part by the Purdue Research Foundation. We thank Adrian Lim for his help with the datasets. We are also very grateful to the reviewers for their valuable comments.

References

ALIAGA, D. G., BENEŠ, B., VANEGAS, C. A., ANDRYSCO, N. 2008. Interactive reconfiguration of urban layouts. *IEEE Computer Graphics & Applications* (May/June), 28, 3, 38-47.

ALIAGA, D. G., ROSEN, P., BEKINS, D. 2007. Style Grammars for Interactive Visualization of Architecture. *IEEE Trans. on Visualization and Computer Graphics*, 13, 4, 786-797.

AVIDAN, S., AND SHAMIR, A. 2007. Seam carving for content-aware image resizing. *ACM Trans. on Graphics*, *26*, 3.

CHEN, G., ESCH, G., WONKA, P., MUELLER, P., ZHANG E. 2008. Interactive Procedural Street Modeling. *ACM Trans. on Graphics, 27*, 3.

COHEN M., SHADE J., HILLER S., DEUSSEN O. 2003. Wang Tiles for Image and Texture Generation. *ACM Trans. on Graphics, 22,* 3.

FANG, H., AND HART, J. C. 2007. Detail preserving shape deformation in image editing. *ACM Trans. on Graphics, 26,* 3.

FRUEH, C., AND ZAKHOR, A. 2003. Constructing 3D city models by merging aerial and ground views. *IEEE Computer Graphics & Applications*, 23, 6, 52–61.

FUNKHOUSER, T., KAZHDAN, M., SHILANE, P., MIN, P., KIEFER, W., TAL, A., RUSINKIEWICZ, S., DOBKIN, D., 2004. Modeling by Example. *ACM Trans. on Graphics, 23,* 3, 652-663.

HERTZMANN, A., JACOBS, C., OLIVER, N., CURLESS, B., SALESIN, D. 2001. Image Analogies. In *Proc. of ACM SIGGRAPH 2001*.

HOLTON, M., 1994. Strands, Gravity, and Botanical Tree Imagery, *Computer Graphics Forum 13(1)* 57–67.

IJIRI, T., MECH, R., IGARASHI, T., MILLER, G. 2008. An Example-based Procedural System for Element Arrangement. *Proc. of Eurographics*, 429-436, Computer Graphics Forum, Vol. 27.

KIM, J., AND PELLACINI, F. 2002. Jigsaw image mosaics. *ACM Trans. on Graphics, 21*, 3, 657–664.

KWATRA, V., ESSA, I., BOBICK, A., KWATRA, N. 2005. Texture Optimization for Example-Based Synthesis. *ACM Trans. on Graphics, 24,* 3, 795-802.

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

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

MUELLER, P., ZENG, G., WONKA, P., GOOL, L. V. 2007. Image-based procedural modeling of façades. *ACM Trans. on Graphics, 26,* 3.

MUMFORD, L. 1961. The City In History, *Harcourt, Brace, & World (New York)*.

PALMA DE, A., PICARD, N., WADDELL, P. 2007. Discrete Choice Models with Capacity Constraints: An Empirical Analysis of the Housing Market of the Greater Paris Region, *Journal of Urban Economics*, 62, 204-230.

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

PRUSINKIEWICZ, P., AND LINDENMAYER, A. 1996. The algorithmic beauty of plants. *Springer-Verlag,* New York, Inc., NY, USA.

RIBARSKY, W., WASILEWSKI, T., FAUST, N. 2002. From urban terrain models to visible cities. *IEEE Computer Graphics & Applications,* 22, 4, 10–15.

SLOAN P., ROSE C., COHEN M. 2001. Shape by Example. *Proc. of ACM Symp. on Interactive 3D Graphics*, 135-143.

SUN, J., YU, X., BACIU, G., GREEN, M. 2002. Template-based generation of road networks for virtual city modeling. *Proc. of ACM Symp. on Virtual Reality Software and Technology*, 33–40.

VESTRI, C., DEVERNAY, F. 2001. Using Robust Methods for Automatic Extraction of Buildings. *IEEE Computer Vision and Pattern Recognition*, 133-141.

WADDELL, P., ULFARSSON, G., FRANKLIN, J., LOBB, J. 2007. Incorporating Land Use in Metropolitan Transportation Planning. *Transp. Res. Part A: Policy and Practice,* 41,382-410.

WEI, L., HAN, J., ZHOU, K., BAO, H., GUO, B., SHUM, H. 2008. *ACM Trans. on Graphics, 27,* 3.

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

ZHANG, E., MISCHAIKOW, K., TURK, G., 2005. Feature-Based Surface Parameterization and Texture Mapping, *ACM Trans. on Graphics*, *24,* 1, 1-27.

ZHOU, H., SUN, J., TURK, G., REHG, J. 2007. Terrain Synthesis from Digital Elevation Models. *IEEE Trans. on Visualization and Computer Graphics*, 13, 4, 834-848.