Urban Dictionaries: Modeling and Simplifying City Environments

Daniel G. Aliaga Chris Hoffmann
Department of Computer Science at Purdue University

ABSTRACT

Urban environments are challenging to model because they are both very large and very diverse. The size of the environment makes capturing the details of every structure prohibitive, leaving us with only sparse information. Photogrammetric reconstruction attempts to build 3D structures but is often not robust and requires manual input, thus making it difficult to scale to large areas. Procedural modeling of urban environments provides a means for quickly creating architecture, but it does not address populating the database of urban features and it does not use real-world information to mimic actual urban areas.

In this paper, we use sparse aerial-based information about an urban environment that is readily available in many towns and cities. From this information, we isolate a small number of canonical urban structures that can be assembled algorithmically using only a few simple operations. The result is an approximation of the entire urban environment to within a prespecified error threshold of the original data. Unlike synthetic approaches that prescribe a grammar, our task is to discover a small dictionary of words and associated application rules for an actual urban environment. Once the dictionary and rules are known, an urban environment in the style of the original can be instantiated. The canonical set of words can be used to prioritize acquisition efforts, to reduce image data, to compress geometry of the model, to find urban patterns, and to visualize properties of the urban environment.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling, I.3.7 [Computer Graphics]: Three-dimensional Graphics and Realism, I.6.3 [Simulation and Modeling]: Applications.

1. Introduction

Modeling and visualizing large urban environments is a great challenge for computer graphics. Cities are a complex collection of man-made structures arranged in parcels, city blocks, and neighborhoods. A digital model of such a large-scale urban environment enables applications such as urban interactive walkthroughs, web-based navigation (e.g., MapQuest, Google Earth), emergency response training and simulation, urban planning and modeling, and content creation for the entertainment market.

Acquiring large urban spaces, on the whole, is a significant challenge, involving considerable manual effort and computational tasks. Dense urban environments are particularly challenging to model because they are both very large and very widespread, spanning from a few square kilometers to hundreds of square kilometers. The size of the environment makes obtaining detailed information of every structure prohibitive, leaving us with only sparse information. Satellite and aerial photographs provide overhead views of large urban areas, but they do not provide accurate 3D structural data. However, cities and counties do maintain limited records of parcel boundaries, basic building information, and other metadata, such as water pipes, sewage systems, etc., that is in digital form and can be used as a starting point. Extracting streets and building contours fully automatically is an option, albeit a challenging one.

The 3D modeling of large urban environments has a significant history in graphics. Procedural modeling specifies a set of terminals and rules for synthesizing urban-like models [Parish01]. While procedural modeling provides a means for quickly creating architecture, it does not address populating the database of urban features and does not use real-world information to mimic actual urban areas. Geometric models have been obtained from photographs using 3D reconstruction and photogrammetric modeling. While it is a stated goal to recover the model automatically, the most robust systems require user input and thus do not scale to entire cities [Debevec96, Debevec98]. In general, the large size of urban environments and the difficulty in obtaining detailed information limit the above modeling techniques.

In this paper, we attack this immense problem by providing an automatic way to focus resources and obtain information about the most unique and valuable urban structures. For a given urban area, we discover a compact hierarchical dictionary of approximate urban structures and associated application rules that can simulate the entire urban environment. Once these rules and the dictionary are known, a model of the urban environment can be instantiated from this small number of urban structures. As input to this automated discovery, we are provided with aerial pho-

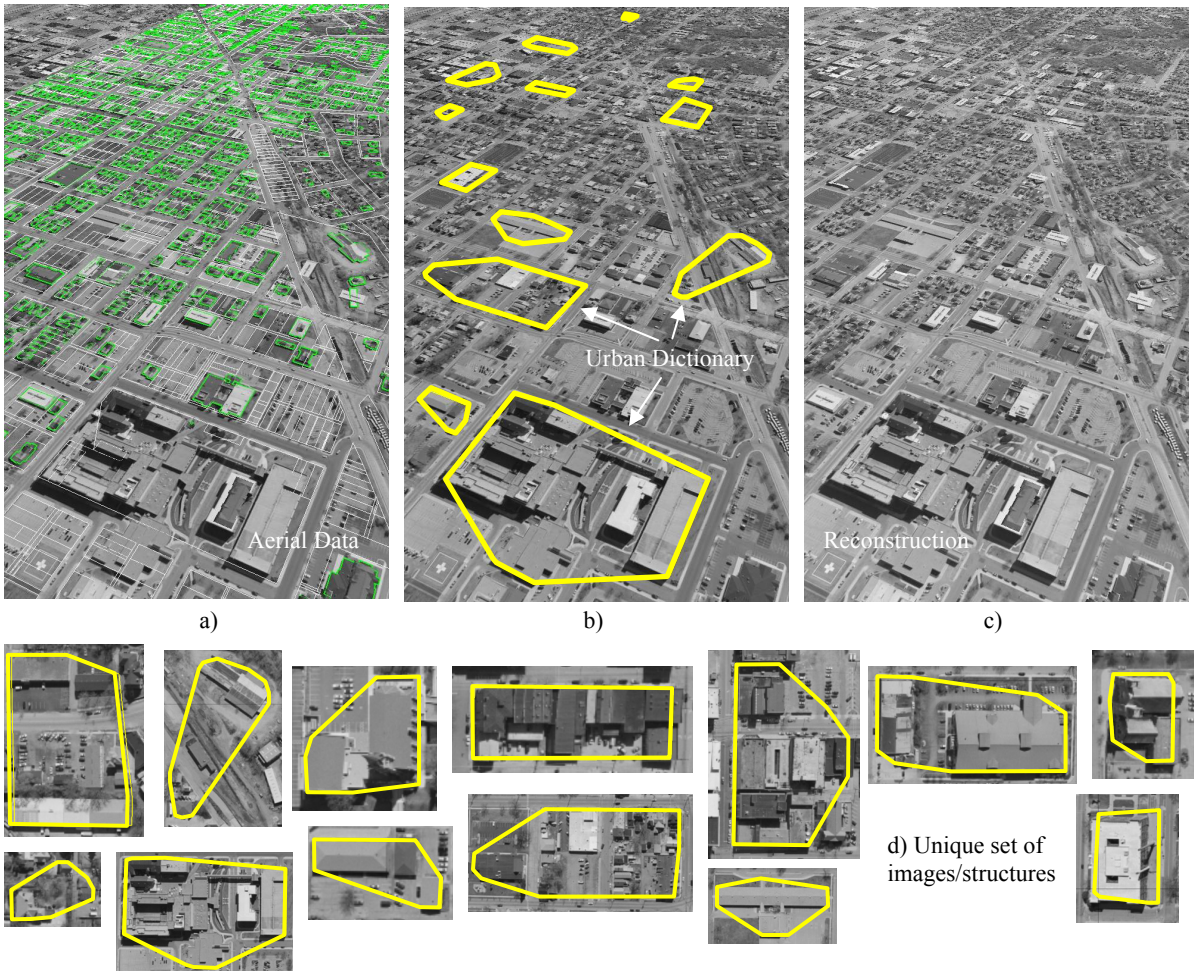

Figure 1. Urban Dictionary*. (a) Contains initial aerial data (photographs, contours, parcels, etc.). (b) Identifies compact clusters of urban structures that best represent all parcels with buildings. (c) Discovered dictionary and its instantiations are used to replace/reconstruct all parcels with building contours. (d) Illustrates the most unique building/parcel pairs, the words of the dictionary. This dictionary of words can also be used to seed procedural modeling, ease detailed acquisition efforts, and compress data.*

tographs and parcel boundaries, building contours, and building heights overlaid on the photographs of a real city. (Figure 1). The hierarchical dictionary so discovered can then be used to establish the terminals and rules for procedural modeling; or to rank photogrammetric reconstruction efforts; or to compress aerial photographs and urban geometry; or to find urban patterns and classifications.

Our key observation is that large urban environments exhibit a significant amount of *local* and *global* redundancy despite the individuality in detail. General geometry simplification exploits local redundancy by collapsing adjacent elements of geometry [Luebke01]. However, urban models exhibit both local redundancy (e.g., adjacent houses can be very similar) and global redundancy (e.g., a house in one part of a city can be very similar to a house in another distant part). As we consider even larger urban areas, we could find even more redundancy. Similar to image quantization methods and epitomes [Jojic03], we seek to efficiently exploit this redundancy in order to find a compact hierarchy of urban structures for modeling the entire urban environment.

Our contributions in this paper include:

- developing a flexible framework to automatically approximate real cities and generate new ones "in the style of" other cities;
- articulating new, but natural, operations for identifying and simplifying structures that are measurably similar;
- proposing a novel metric by which to quantify the similarity of an approximating city model and to quantify the data reduction accomplished by the algorithm.

2. Related work

The modeling and rendering of large urban of environments has been addressed in several ways in computer graphics. Procedural modeling focuses on building synthetic models of urban-like environments from a set of principles defined by a grammar. This approach is most useful for creating models of objects or systems that have a high degree of redundancy or self-similarity. Most notably, L-systems have been successful in the modeling of plants [Prusinkiewicz91], and have been used for automatic city and building generation [Parish01]. Shape grammars [Stiny75],

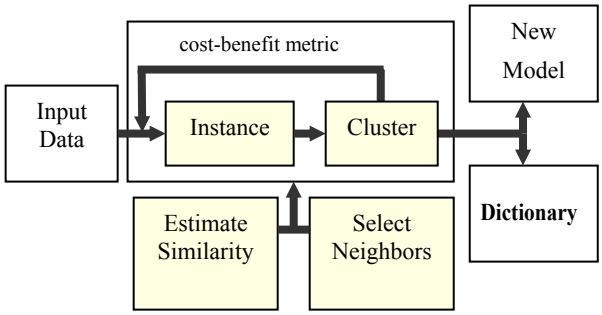

Figure 2. Processing Pipeline. *We use an iterative method that applies instancing and clustering operations, guided by cost-benefit metrics, to create the dictionary.*

which define rules for the specification and transformation of 2D and 3D shapes, have also been used to model architecture. Wonka et al. [Wonka03] employ a split grammar to automatically generate architecture from a database of rules and attributes. The split grammar divides a geometric object into feature regions, which are then provided with detailed geometry and material from a database to produce a final rendering. The user can specify parameters that guide the choices made by the generation program, allowing for very fast creation of many variations of a particular building shape, type, and style. While procedural modeling provides a means for quickly creating architecture from a small number of terminals and rules, the data and procedures are not based on an actual real-world city and do not attempt to mimic a real world environment.

Photogrammetric reconstruction and image-based modeling and rendering (IBMR) focus on building a representation of a real model from photographs. Photogrammetric reconstruction recovers the dimensions of a 3D geometric model and can map the image data to it [Pollefeys00]. Facade [Debevec96, Debevec98] has served as the prototype for several commercial packages [MetaCreations02, Eos05, RealViz05] that combine photogrammetric reconstruction with user-assistance. Image-based modeling and rendering [Max95, McMillan95] is a partner of capture techniques like photogrammetric modeling, but unlike photogrammetric modeling, an IBMR system directly re-samples photographs of a static scene to create novel views of the acquired object or small environment [Aliaga03, Buehler01, Gortler96, Levoy96, Shum99]. Some 3D reconstruction efforts have focused on large urban environments [Coorg98]. For instance, Teller et al. [Teller01] use global positioning satellites in conjunction with an image-based calibration approach tuned to typical building shapes to capture images spanning a campus-size environment. Unfortunately, these techniques are difficult to extend and scale up to large environments and the hardware platforms used by some make their deployment cumbersome in large urban environments. Their usability would be significantly improved if we could direct our reconstruction efforts to a suggested subset of the entire urban environment.

In our work, we identify and address a common shortcoming of these two extremes to urban modeling. By discovering an estimate of the local and global redundancy of an urban environment, we can create a compact hierarchy of canonical urban structures. These structures can serve to provide terminals and rules for procedural modeling efforts, to guide photogrammetric reconstruction efforts, and to assist in compressing the urban model data.

The inspiration for our work comes from several quantization and compression methods. In particular, we draw from the concept of epitomes of images [Jojic03]. The epitome of an image contains the essence of the textural and shape properties of the image. The size of the epitome is considerably smaller than the size of the image it represents. However, it still contains most of the constitutive elements needed to reconstruct the image and supports tasks such as image segmentation, motion estimation, and image editing.

3. Urban Dictionaries

We devise an iterative algorithm that efficiently navigates through the solution space in order to find a hierarchy of representative urban structures (e.g., parcels, buildings, neighborhoods, etc.) from which to approximate a large urban environment. Starting with initial data given by aerial photographs and digital information outlining parcels and basic building shapes, our method establishes a set of equivalence classes and spatial clusters of representative structures. By simultaneously merging equivalence classes and clustering nearby structures, we arrive at a small number of equivalence classes and at a few clusters that are compact in size, staying within a user-specified error tolerance. A small set of equivalence classes reduces the number of typical structures necessary to represent and reconstruct the environment. Spatially-compact clusters of representative structures facilitate 3D reconstruction efforts and compression methods. The resulting compact hierarchical dictionary can be used to reconstruct the urban model by instantiating copies of these words, yielding a representation that is more efficient to capture, render, compress and analyze.

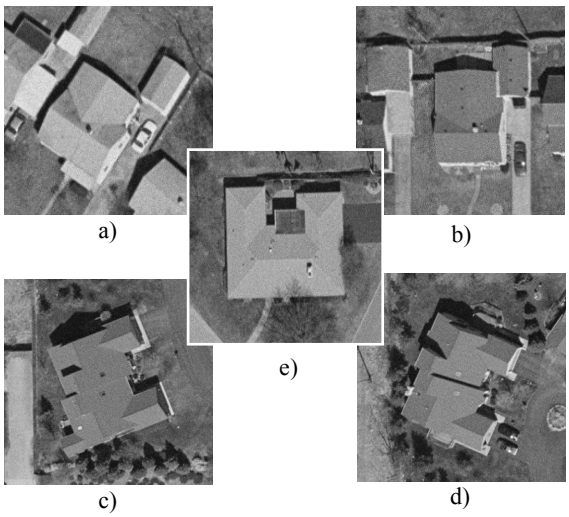

Figure 3. Instancing and Similarity. *Global redundancy is exploited by growing bottom-up a hierarchy of equivalence classes and subsequently instancing from the tree. In this example, houses (a) and (b) as well as (c) and (d) are considered very similar to each other according to our similarity metric. These similarities are used to form a tree, where (e) is chosen as the best the overall representative.*

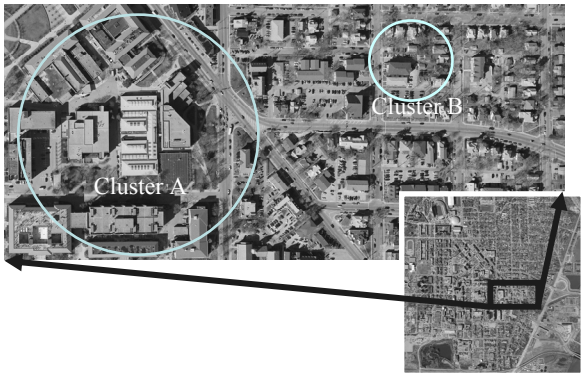

***Figure 4. Hierarchical Clustering**. For large urban areas (see inset), we seek compact sets of urban representatives. By combining spatial clustering with instantiation, our method finds spatial clusters, such as A, containing a set of dissimilar structures but of frequent use elsewhere (e.g., in the area surrounding B). In addition, the same algorithm must find clusters such as A, which contain unique structures, but are needed to maintain the same error measure.*

Figure 2 shows our processing pipeline. First, the initial model is input to the simplification algorithm. Then, cost-benefit metrics guide class-merging and clustering operations to converge to a hierarchical representation. This process also uses metrics that define similarities between structural elements and methods that establish the neighborhood of each structural element. Finally, a dictionary of words is output as well as a reconstruction of the urban environment.

3.1 Instancing and Equivalence Classes

We establish equivalence classes of buildings and of groupings of buildings. Intuitively, buildings and houses in one area of a city might be very similar to structures in another, possibly distant, part. So, each equivalence class combines similar structures and allows representing the elements in the class with a single representative structure. In the tree, the similarity between members of a class increases as you travel from the root to the leaves, because of compounding similarity errors. Thus, we can choose a small coarse set of representative structures near the root or a larger detailed set of structures near the leaves to approximate the urban environment.

We construct the equivalence class tree bottom-up by merging classes. Initially, each equivalence class is a singleton, becoming a leaf node in the tree., Each merging operation joins two similar equivalence classes and decreases the total number of classes. Given N initial classes, there are at most $O(N^2)$ possible mergers in each iteration. To choose a near-optimal set of equivalence classes, we take a greedy approach that maintains a priority queue of operations and merges the next most similar equivalence classes subject to a constraint described later.

Figure 3 shows example equivalences. Using our similarity metric, we obtain a scalar value that measures the similarity of all pairs. Tree construction merges the most similar classes first (e.g., (a) and (b), and (c) and (d)) and eventually approximates all structures using a single representative (e). More details on our similarity metric are provided in Section 3.3.

3.2 Spatial Clustering

The objective of clustering is to combine dissimilar structures that frequently occur with minor changes elsewhere (Figure 4). The hierarchy of spatial clusters is also built bottom-up with each parcel or building initially placed in a cluster of its own. Given an average neighborhood size of $k_1 << N$ structures, clustering chooses from an overall $O(k_1N)$ possible pairings of neighbors during each iteration. In an urban environment, we generally group two types of neighbors: adjacent (e.g., abutting parcels) and nearby (e.g., city blocks and houses separated by roads or empty space). Clustering first joins all immediate neighbors and then proceeds to group nearby neighbors. Eventually, either all city blocks are joined or the algorithm terminates when the error threshold has been exceeded.

To perform clustering, an adjacency graph is used to first join immediate neighbors and then a spatial data structure (e.g., quadtree) is used to group city blocks. Eventually, the entire area is contained within a single cluster. Unlike equivalencing, where spatially distant, but similar structures are joined, clustering requires the combined structures to be spatially adjacent. Instances of cluster may, of course, be distant.

3.3 Similarity Metric

We define a similarity metric between two urban structures. The metric should be invariant to both translation and rotation. Thus, a parcel or house observed from above as facing north in one area of a city should be considered identical to the same house in another part of the city facing east.

To estimate the similarity between buildings, parcels, or parcels containing buildings, we use a weighted combination of geometry-based and image-based similarity. For geometry, our solution is to compare a polar-coordinate representation of the two structures. For images, we use a normalized image-difference metric [Hartley04]. First, the centroids of the structures are aligned, so determining a translational offset between the structures. Second, a dense sampling of points along the boundaries of the parcels and buildings are transformed to a regular sampling in polar coordinates. Third, the best correlation between the two arrays of radial distances and the two image regions is found yielding the rotational offset between the structures. The correlation value is an indication of dissimilarity. We use the negative of the correlation value as similarity measure. Thus, zero indicates equality and a large negative number indicates a significant difference.

Given two equivalence classes of the same configuration (e.g., parcel-to-parcel, building-to-building, or parcel-with-

```
0. Initialize equivalence tree and spatial
   tree with one node per structural element
1. Find neighbors and sort structures by size
2. Enqueue class mergers and collapses above
   quality threshold
3. While queue not empty
4.   Execute next best operation
5.   If collapse, inherit neighbors
6.   If class merger, collapse empty clusters
7.   Remove affected nodes and operations
8.   Enqueue new operations
9. End
```

***Figure 5. Pseudo-code summary of simplification algorithm**.*

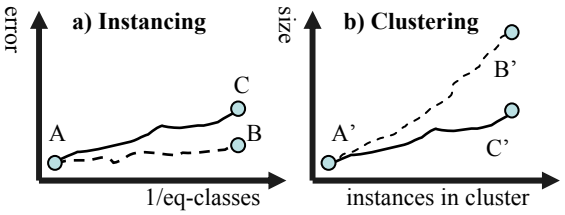

Figure 6. Cost-Benefit Metrics*. We use cost-benefit metrics to steer our simplification algorithm. In this didactic example, the initial solutions starts at A/A' and naïve instancing leads to B/B'. By simultaneously considering both instancing and clustering benefits and costs, we are able to reach a better overall answer C/C'.*

building pairs), similarity is estimated by comparing representatives from each class. The representative of a class is chosen to be the member most similar to other members (i.e., the one with the largest sum of similarity values). The representative is recomputed only when members are added or removed from the class.

4. Simplification Algorithm

Our algorithm traverses the solution space by using a similarity metric and simultaneously performing a combination of equivalence class merging and spatial clustering. Candidates for both types of operations are placed into a single priority queue and the next best candidate at the top of the queue is iteratively executed. Figure 5 provides a pseudo-code summary of the simplification algorithm.

The order of the operations affects the compactness of the final dictionary. Thus, we employ cost-benefit considerations to steer our algorithm. For equivalence class merging, the horizontal axis in Figure 6a indicates benefit of the current solution as the inverse of the number of equivalence classes. The vertical axis represents cost as the current similarity threshold (e.g., larger error means less similarity). For clustering, the horizontal axis in Figure 6b shows the benefit as the average number of elements that can be instantiated using the members of a cluster and the vertical axis depicts cost as the average size of the clusters (e.g., radius, number of members, etc.). Initially, we have one equivalence class and one cluster per structural element (point *A* in Figure 6a and point *A'* in Figure 6b). Ideally, we seek a solution at the same height (i.e., same cost) and all the way to the right in both graphs (i.e., maximum benefit).

Each cost-benefit graph determines an ordering for each type of operation. However, the next best operation of one type might adversely affect the benefit of the other type. For instance, consider an environment with significant homogeneity amongst the structural elements (e.g., an urban area of tract houses). A significant number of class mergers can be performed at low cost (point *B* in Figure 6a). However, if we do not consider spatial clustering, our solution might appear near point *B'* in Figure 6b.

To prevent this situation, our algorithm chooses the next best operation (either merge or cluster) that enhances the solution in both graphs and yields an overall improved solution (such as point *C* and *C'* in Figures 6a and 6b, respectively). That is, we can (1) enforce a maximum cost for merging, (2) enforce a maximum cost for clustering, or (3) use a weighted combination of both metrics. The third option is useful if the user can provide information about the environment. Roughly speaking, urban environments exhibiting homogeneity benefit, on average, from more instancing while environments with large variations profit more from clustering structures.

5. Implementation Details

Our software is implemented in C++ on a Pentium IV PC using standard OpenGL, GLUI, and GLUT libraries. To accelerate rendering, we cache the similarity values and the translational and rotational offsets between the representatives of each equivalence class and its members.

Our urban data was extracted from an ArcGIS database in active use by <removed-for-anonymous-submission> city. It was exported to a DXF file format and converted to a custom file format. We assigned building contours to parcels that contained them and joined parcels contained within the same building. A data filter was also included to remove incomplete parcels and building specifications.

To accelerate comparisons for equivalence class merging, we reduce the total number of pairs to $O(k_2N)$ and the number of operations to update to $O(k_2)$ for some constant $k_2 << N$. To obtain only $O(k_2N)$ operations, we sort representatives from each equivalence class based on their physical size (e.g., radius) and only pair them with equivalence classes within a distance k_2 in this sorted array. After each operation is performed, we only need to update $O(k_2)$ operations. We remove the k_2 operations involving the just-merged equivalence classes and compute k_2 operations for the newly formed class.

6. Results

We have applied our algorithm to the discovery of urban dictionaries using information from a large real-world area. Our dataset consists of publicly-available high-resolution images and metadata of the central portion of a city with a population of over 250,000 people (<website-removed>). Aerial photographs were taken at a resolution of approximately 15 centimeters per pixel and spanning over 32 square kilometers. Metadata includes a network of roads, parcel boundaries, building contours, and partial building height information. Both images and contours are geo-registered. Other information, such as water systems and sewage is also available, although we do not use it.

With our algorithm, we can obtain a variety of solutions for the same area. The solution sets in this paper can be computed in 1 to 15 minutes, approximately. The solutions vary in the size, number, and distribution of representative structures. Figure 7 reports various solutions. From left to right, each row demonstrates three solution sets representing a high-quality, medium quality, and low-quality solution. The top row uses only instancing and equivalence class merging (Figures 7a-c). The set of representative buildings, in this case, are generally distributed through the urban area. On the other hand, the bottom row demonstrates a solution using both instancing and clustering (Figures 7d-f). The cost-benefit metric helps steer the algorithm towards these apparently conflicting goals. It arrives at solutions that are spatially more compact, yet of similar instantiation

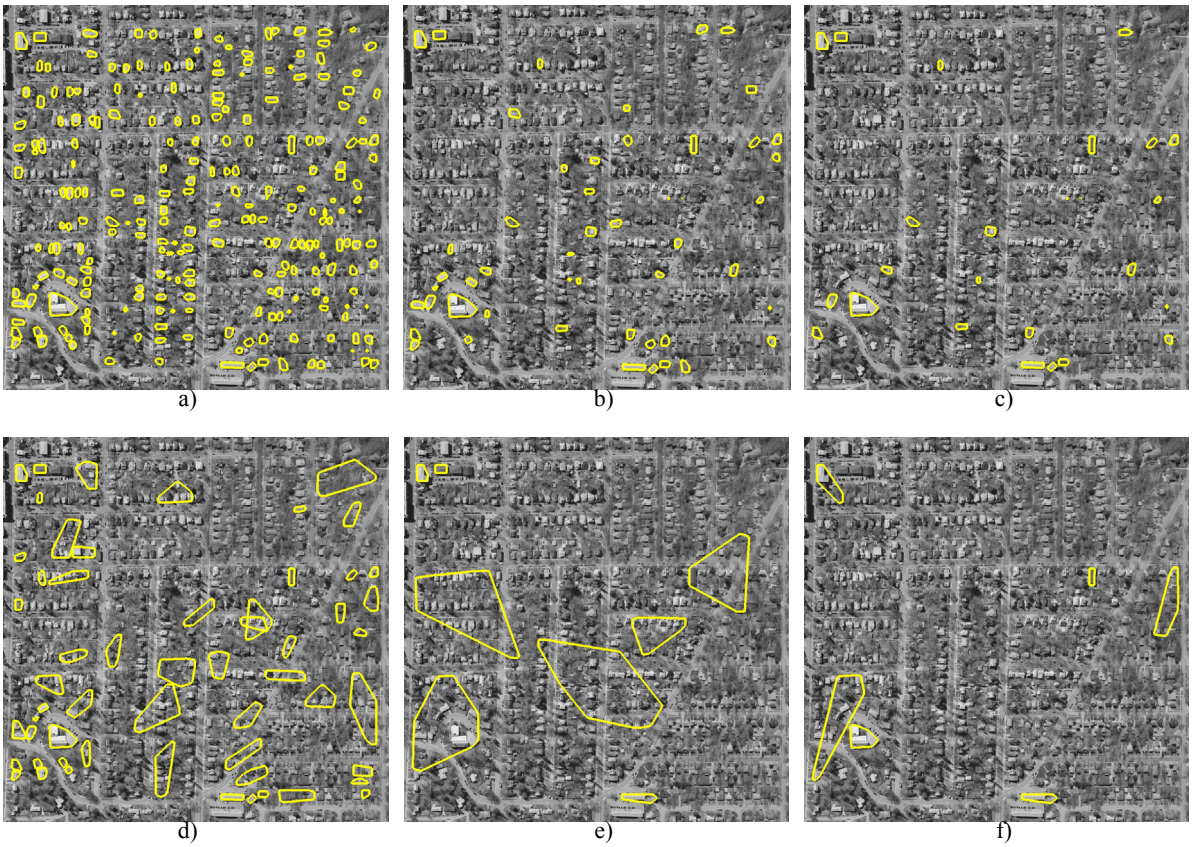

Figure 7. Urban Dictionary Comparisons*. From left to right, both rows show a progression of solution sets (high, medium, and low quality) during the simplification process. The top row (a-c) uses only equivalence class merging and instantiation yielding solutions that typically have a widespread distribution of the found representative structures. On the other hand, the bottom row (d-f) demonstrates how simultaneously considering clustering and instancing can help steer the solution to produce a more spatially compact dictionary of representative structures.*

quality as the corresponding solution from the top row. We believe this control and behavior will help to focus and localize detailed 3D acquisition efforts. For example, visiting only these sparse set of locations, will significantly simplify the immense acquisition task for an entire city.

To better visualize the performance of the cost-benefit metric, we show in Figure 8 a graph of the quality and cost values during a typical computation as well as some example images. The graphs show that during simplification, cost slowly increases, with most of the penalty being at the end, and quality (or, benefit over cost) progressively decreases. In this graph, both clustering and instancing values have been normalized. During execution, estimated weights are used to be able to compare them. In this particular solution set, clustering did not produce significant benefit at the being and thus initially decreased rapidly. Figure 8 also contains close-ups of several building contours drawn on top of the original aerial images. As the simplification proceeds, the original contours are progressively replaced with similar structures from other parts of the city.

Figure 9 demonstrates dictionaries for a large and diverse urban area. Figure 9a shows the original aerial view and Figure 9b demonstrates the parcel and building information available. Using this, our algorithm selected the highlighted areas (yellow) in Figure 9c as being most similar to the surrounding areas for a given error threshold, based on a combination of image similarity, parcel similarity, and building contour similarity. Many of the large (yellow) areas correspond to parks and unique structures and thus cannot be replaced by other structures. Figure 9d contains a reconstructed view in the style of the original environment but using the subset of parcels and buildings from Figure 9c.

The representative structures can also be fed into a procedural modeling program. Figures 9e and 9f demonstrate an overview and close-up view of a synthetic rendering of the urban area generated by a simple in-house procedural modeling system. While it is not our focus, we look forward to more sophisticated procedural modeling systems to generate detailed cities in the style of the captured environments (e.g., Instant Architecture [Wonka03], Build-by-Numbers [Bekins05], etc).

In our current implementation, we do not explicitly handle roads or enforce a functionally correct urban area. Moreover, we rely on the similarity metric to yield plausible reconstructions and thus currently our simplification scheme does not handle very coarse reconstructions, including causing highly repetitive instantiations (see bottom right of Figure 9d).

7. Conclusions

We have presented an automated algorithm that uses sparse urban information obtained from aerial views and city re-

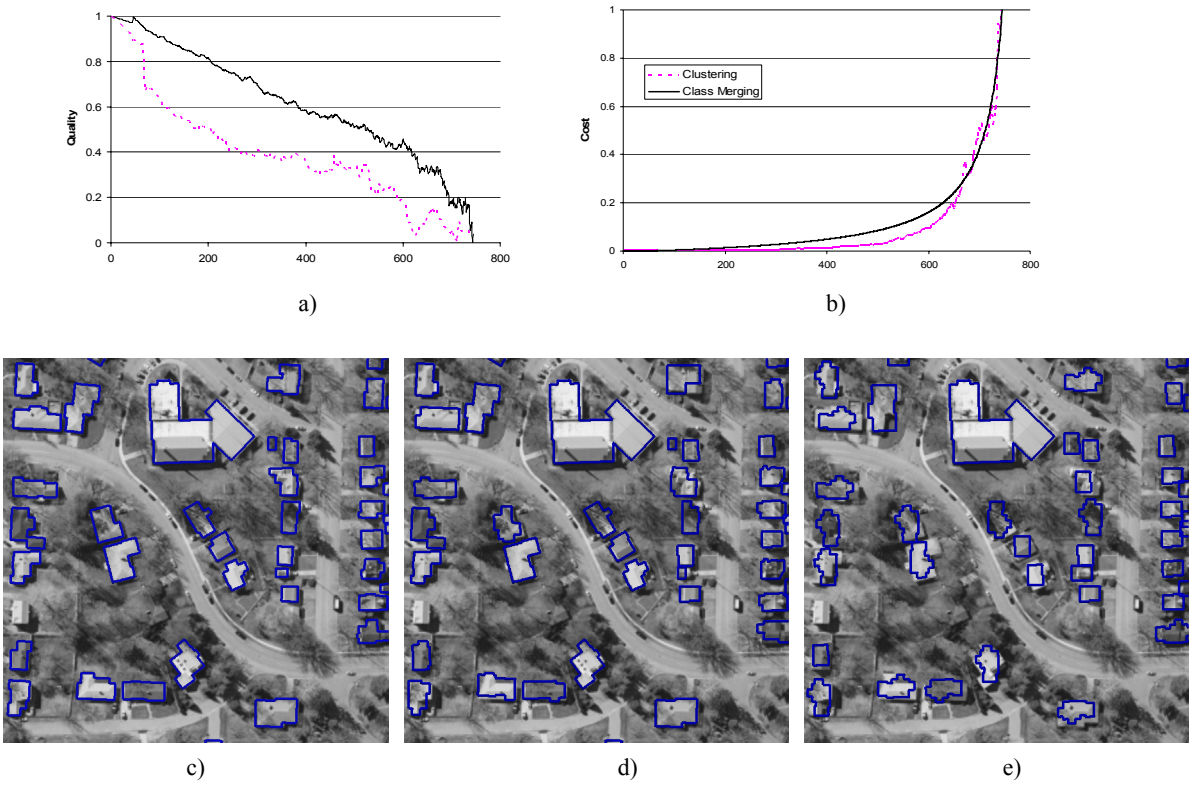

Figure 8. Steering the Simplification Algorithm*. We show the quality values (a) and cost values (b) for both operation types during the simplification of an example urban area. Figures (c-e) show close-ups of a small neighborhood and how building contours are gradually replaced with representatives from elsewhere in the environment. The leftmost image contains the original information and the remaining two images are from a midpoint and the endpoint of the curves in the graph. The simplification stopped when a maximum error threshold was reached.*

cords to discover a compact dictionary of representative structures of the urban environment. Using a real-world city, we demonstrated how representative structures can be efficiently identified. Our approach implements a solution that exploits global redundancy as well as curtails to the benefits of locality. These structures can then be used to establish the terminals and rules for procedural modeling, to rank photogrammetric reconstruction efforts, to compress aerial photographs and urban geometry, and to find urban patterns and classifications.

As future work, we are interested in several areas of improvement. First, we would like to extend our algorithm with a recursive look-ahead – this would enable us to traverse the search space better and to converge on more efficient solutions. Second, we would like to enforce the solution set to account for functionality. This implies that we must ensure the road network, parcel layout, and building placement is feasible. Third, akin to image epitomes and vector quantization, we are investigating the compression aspects of a dictionary of image regions.

At this time, the field is far from capturing explicit details of every structure in every major city. However, by combining aerial views with metadata and other information available for cities, we can discover significantly more information than before and make large strides towards capturing, representing and reasoning about detailed models of large urban spaces.

References

[Aliaga03] Aliaga D., Funkhouser T., Yanovsky D., Carlbom I., "Sea of Images: A Dense Sampling Approach to Capturing Large Indoor Environments", *Computer Graphics & Applications*, pp. 22-30, Nov/Dec, 2003.

[Bekins05] Bekins D., Aliaga D., "Build-by-Numbers: Rearranging the Real World to Visualize Novel Architectural Spaces", *IEEE Visualization*, 2005.

[Buehler01] Buehler C., Boose M., McMillan L., Gortler S., Cohen M., "Unstructured Lumigraph Rendering", *ACM SIGGRAPH 2001*, pp. 425-432, 2001.

[Coorg98] Coorg S., "Pose Imagery and Automated Three-Dimensional Modeling of Urban Environments.", Ph.D. Thesis, Massachusetts Institute of Technology, 1998.

[Debevec96] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. "Modeling and Rendering Architecture from Photographs." *ACM SIGGRAPH*, pp. 11-20, August, 1996.

[Debevec98] Debevec P., Borshukov G., Yu Y., "Efficient View-Dependent Image-Based Rendering with Projective Texture Mapping", *9th Eurographics Rendering Workshop*, June, 1998.

[Eos05] Eos Systems, Inc. PhotoModeler. www.photomodeler.com, 2005.

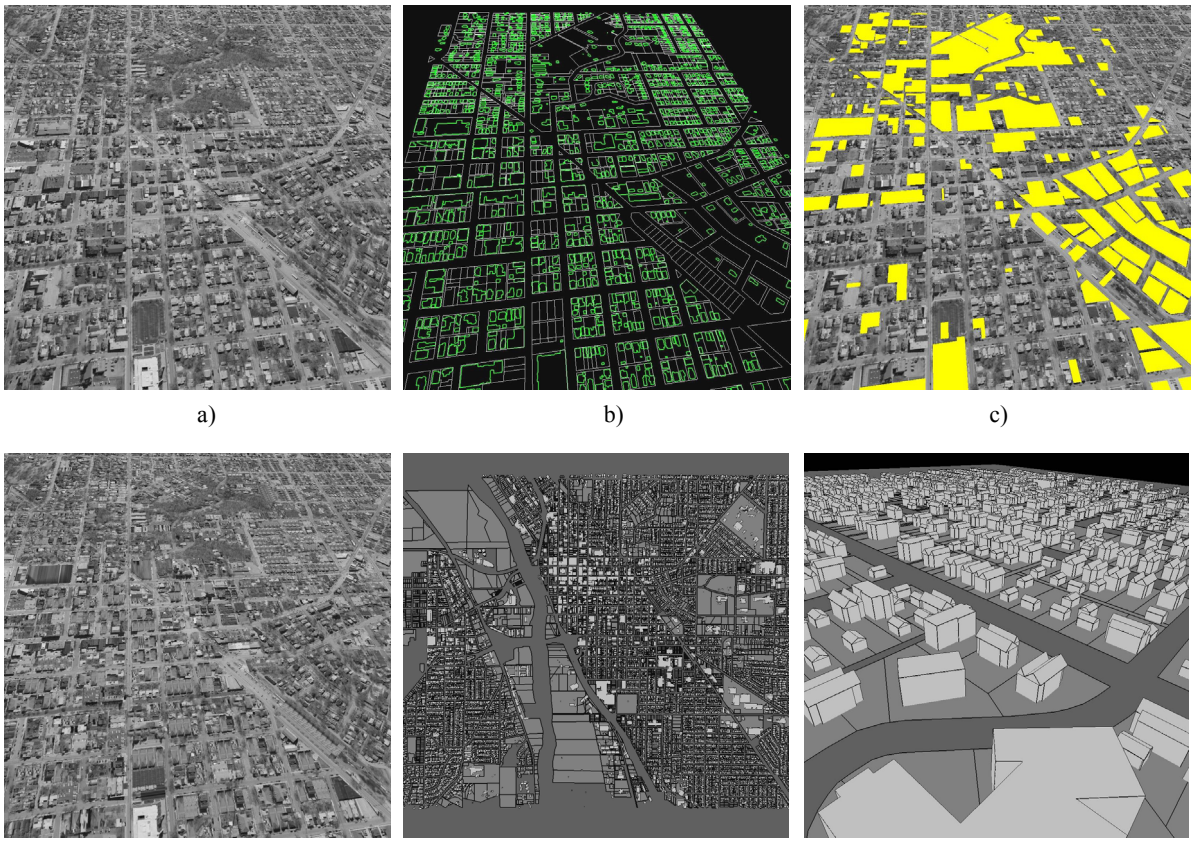

Figure 9. Representing Urban Spaces*. (a) Demonstrates original aerial views of a large and diverse urban space. (b) Contains a rendering of the parcel boundaries and building contours used by our algorithm. (c) A computed dictionary of representative structures. The large yellow areas correspond mostly to uniquely structured parks. (d) An image where parcels with buildings have been replaced with instantiations from the computed dictionary. (e, f) An overview and close-up of a synthetic model created using a simple in-house procedural modeling program and the terminals we computed.*

[Hartley04] Hartley R., and Zisserman A., Multiple view geometry in computer vision, *Cambridge University Press*, 2004.

[Jojic03] Jojic N., Frey B., Kannan A., "Epitomic Analysis of Appearance and Shape", *IEEE International Conference on Computer Vision (ICCV),* 2003.

[Luebke01] Luebke D., "A Developer's Survey of Polygonal Simplification Algorithms", *IEEE Computer Graphics & Applications*, 2001.

[Max95] Max N. and Ohsaki K., "Rendering Trees from Precomputed Z-Buffer Views", *Rendering Techniques '95*: *Proceedings of the 6th Eurographics Workshop on Rendering*, pp. 45-54, 1995.

[McMillan95] McMillan L. and Bishop G., "Plenoptic Modeling: An Image-Based Rendering System", *ACM SIGGRAPH 95*, pp. 39-46, 1995.

[MetaCreations02] MetaCreations Corporation, Canoma, www.canoma.com, 2002.

[Parish01] Parish, Y. I. H., and Muller, P. "Procedural modeling of cities." *ACM SIGGRAPH 2001*, ACM Press, E. Fiume, Ed., 301.308, 2001.

[Pollefeys00] Pollefeys M., Koch R., Vergauwen M., Van Gool L., "Automated reconstruction of 3D scenes from sequences of images", *ISPRS Journal Of Photogrammetry And Remote Sensing* (55)4, 251-267, 2000.

[Prusinkiewicz91] Prusinkiewicz, P., and Lindenmayer, A. "The Algorithmic Beauty of Plants." Springer-Verlag, 1991.

[RealViz05] RealViz, S.A. ImageModeler. www.realviz.com, 2005.

[Shum99] Shum H., He L., "Concentric Mosaics", *ACM SIGGRAPH 99*, pp. 299-306, 1999.

[Stiny75] Stiny, G. "Pictorial and Formal Aspects of Shape and Shape Grammars." Birkhauser Verlag, Basel, 1975.

[Teller01] Teller S., Antone M., Bodnar Z., Bosse M., Coorg S., Jethwa M., Master N., "Calibrated, Registered Images of an Extended Urban Area", IEEE Computer Vision and Pattern Recognition (CVPR), 2001.

[Wonka03] Wonka P., Wimmer M., Sillion F., Ribarsky B., "Instant Architecture", ACM SIGGRAPH, 2003, 669-677.