Eurographics Workshop on Urban Data Modelling and Visualization (2014)
V. Tourre and G. Besuievsky (Editors)

Hierarchical Radiosity for Procedural Urban Environments

Ferran Roure †, Gonzalo Besuievsky‡ and Gustavo Patow §

ViRVIG-UdG - Universitat de Girona - Spain

Abstract

In this paper we present a novel solution for the computation of diffuse global illumination in urban environments that takes advantage of the underlying structure of the procedural building models used for generating the city, using them to compute a realistic global illumination solution based on the well known hierarchical radiosity algorithm. As we generate the geometry procedurally, we take advantage of the generation hierarchy to be the base of the hierarchical radiosity algorithm, without using the classic quad-based subdivision scheme. This structure is used for low-frequency global illumination, being later combined with a shadow-map-like system for the high-frequency component, thus resulting in a complete global illumination solution for procedural urban environments.

1. Introduction

In the last decades we have witnessed enormous improvements both in urban modeling, mainly through procedural techniques, and in the interactive rendering of large scenes. However, the target of accurate photorealistic rendering of large urban scenes has been barely touched until now. Applications like GoogleMaps and Nokia's HERE (formerly OviMaps) are good examples of this, where only direct illumination is used for all kinds of visualizations. A more general solution, including inter-reflections and other indirect illumination problems should be considered in order to achieve an accurate visualization of urban environments. In this paper we present a novel interactive global illumination solution for large urban landscapes based on procedural modeling techniques.

Contributions Here we present a new global illumination technique for the interactive photorealistic rendering of urban landscapes. Basic ingredients of our approach include:

- a compact and efficient pipeline for both diffuse global illumination and high-frequency sun illumination for urban environments, minimizing the computations needed for each sun position.
- the illumination pipeline is smoothly blended onto the procedural modeling pipeline, taking advantage of all procedural features as exploring immediate results for changing building parameters.

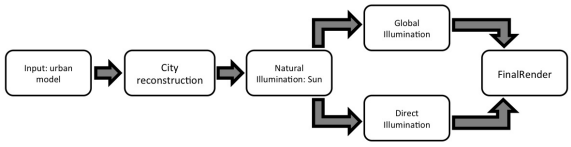

Figure 1: *The pipeline of our global illumination algorithm.*

2. Previous work

Real-time city rendering

In the last few years a number of algorithms for real-time rendering of urban environments have been proposed [CBG*07, DBCG*09, ABCN10, LBZ*10]. These approaches achieve interactive frame rates by adapting traditional acceleration strategies (level-of-detail, image-based rendering, visibility culling, and on-demand loading) to the particular properties of city models: 2.5D overall shape, plane-dominant geometry, regular structure, dense occlusion, and large texture datasets. However, they only consider local illumination, thus providing an incomplete result in terms of the final image quality.

† e-mail: froure@eia.udg.edu, Current Address: Computer Vision and Robotics Group (ViCOROB), Universitat de Girona, Spain
‡ e-mail:gonzalo@imae.udg.edu
§ e-mail:dagush@imae.udg.edu

DOI: 10.2312/udmv.20141071

On the other hand, several image-based representations have been proposed to accelerate the rendering of distant buildings, including planar impostors [MS95], textured depth meshes [SDB97, JW02], and point-based impostors [WWS01]. Some recent approaches represent the geometry of 2.5D cities as hierarchies of displacement maps which are rendered using relief mapping techniques [CBG*07, DBCG*09, ABCN10]. Ali et al. [AYRW09] also use displacement maps, although for representing facade details rather than complete 2.5D buildings.

General algorithms for view-frustum culling and occlusion culling have been also specialized for urban rendering [WWS00, PSC10]. State-of-the-art urban renderers also combine some form of on-demand loading [TMJ98, DB11], texture hierarchies [Buc05], and speculative prefetching [DBCG*09]. Also, the poor performance of traditional mesh simplification algorithms on urban models has motivated the development of algorithms specifically designed for buildings [HW10, DB05] and groups of buildings [CBZ*08, YZM*11]. Most of these algorithms generate discrete LOD representations, although some recent approaches rely on hardware tessellation to instantiate buildings at continuous levels of detail [LBZ*10].

Despite all the above techniques, most approaches for large-scale urban rendering only support direct illumination plus shadow mapping. Di Benedetto et al. [DBCG*09] proposed a system that computed indirect illumination effects in urban rendering, but limited to environments maps and precomputed ambient occlusion, thus severely limiting photorealistic appearance under changing lighting conditions. Argudo et al. [AAP12] proposed a photon mapping-based approach where photons where tracked and stored in specifically tailored cylindrical texture maps, one for each city block. Later, Muñoz et al. [MPAP13] used the same data structure to compute nocturnal illumination, by using a simple OpenGL rendering for distant illumination, which was progressively refined with photon mapping as the user visited different parts of the city. Both solutions required large data structures to store the photon hits, and were not able to converge to an artifact-free solution in real time. On the other hand, other global illumination approaches for urban models run at non-interactive rates [VAW*10].

Procedural Building Modeling

The current trend in procedural building modeling is to use grammar-based procedural techniques that have shown promising results, as shown by Wonka et al. [WWSR03] and later improved by Müller et al. [MWH*06].

The main concept of a shape grammar is based on a rulebase: starting from an initial axiom shape (e.g. a building outline), rules are iteratively applied, replacing shapes with other shapes. A rule has a labelled shape on the left hand side, called predecessor, and one or multiple shapes and commands on the right hand side, called successor. Commands are macros creating new shapes or commands. An example rule could be $predecessor \rightarrow CommandA : CommandB$. Traditionally, four commands were introduced in [MWH*06]: *Split* of the current shape into multiple shapes, *Repeat* of one shape multiple times, *Component Split* (called *Comp*) creating new shapes on components (e.g. faces or edges) of the current shape and *Insert* of pre-made assets replacing a current predecessor. Traditionally, during a rule application, a hierarchy of shapes is generated corresponding to a particular instance created by the grammar, by inserting rule successor shapes as children of the rule predecessor shape [MWH*06] [LWW08]. This production process is executed until only terminal shapes are left. From this rulebase, the instance and associated shape hierarchy in Figure 2 are automatically generated.

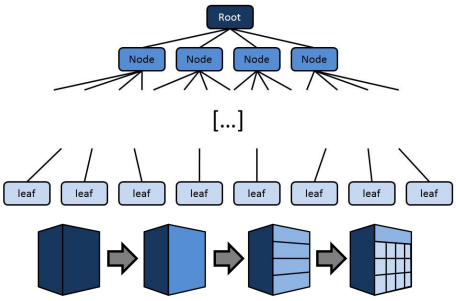

Figure 2: *The tree of primitives in a building construction, and the representation of different tree levels on a building.*

Hierarchical Radiosity

Hierarchical radiosity was introduced in the pioneering work by Hanrahan et al. [HSA91], and later on extended with different techniques. Here we will provide a brief overview, but a detailed description can be found in the book by Cohen and Wallace [CWH93]. In the classic algorithm, lighting calculations are limited to a user-specified level of accuracy, by means of hierarchically subdividing the polygons in the scene into a quadtree, and creating light-transfer "links" at the appropriate levels of the hierarchy. In the original hierarchical radiosity solution [HSA91], radiosity is considered constant over each quadtree element. The rectangular nature of the quadtree, and the constant reconstruction result in the need for very fine subdivision for high quality image generation (high quality shadows etc.). See Figure 3.

3. Overview

The base of our solution rely on a couple of observations. First of all, for daylight illumination, it can be noted that the diffuse component at a wall varies quite smoothly because

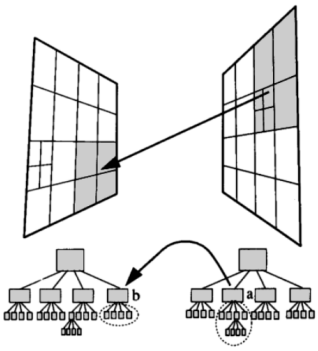

Figure 3: *Hierarchical radiosity algorithm: the energy is exchanged between two patches a and b at the same or different levels in the hierarchy of the two root primitives.*

Figure 4: *Computing global illumination at the asset level can be justified by observing the little variance in luminance values at two extreme points with a distance approximated to the asset size.*

it is composed of low frequency signals. Figure 4 shows an example where the parts of the wall that have no direct illumination have small luminance variations, even at large distances. The size of the window observed in the picture is roughly of the same size as the assets used in our models, so we can assume the same small variations for the diffuse illumination in our simulations.

The second observation comes from the simple fact that both hierarchical radiosity and procedural modeling of buildings generate trees of patches (often called primitives in procedural modeling). These trees, although different in the way they are generated (i.e., quad-tree like for hierarchical radiosity, as seen in Figure 3 and rule-based for procedural modeling, as seen in Figure 2), generate a hierarchy that is comparable in the sizes and shapes of the primitives used. Also, we can observe that the whole radiosity algorithm presented in Section 2 does not depend on the procedure used to refine the nodes, so the tree generation can be changed from the traditional quad-tree-based subdivision by any other one. Different strategies for subdividing the input

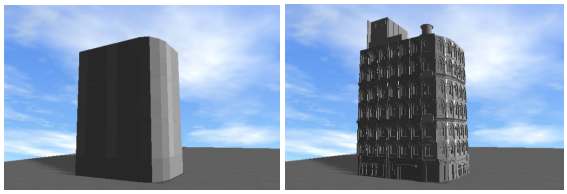

Figure 5: *After all primitives have been positioned (left), it is necessary to insert for each one the respective asset (right).*

patches can be used, and in this paper we propose to directly use the tree generated for the procedural modeling of the building. This completely avoids generating another surface subdivision, and allows us to re-use the originally created hierarchy, making our hierarchy completely independent of the sun position. This way, the illumination process is smoothly integrated into the procedural modeling of the urban landscape.

4. City Modeling and Reconstruction

To keep a link between modeling and the radiosity simulation, we decided to store the primitives as a tree (see Figure 2), storing for each leaf primitive the ID of the geometry asset to be inserted, e.g., a window or a door. This structure is directly derived from the procedural rules, where the *insert* commands are used as terminal shapes. In our case, these assets are not inserted in the modeling stage, but after the whole radiosity simulation is performed. We store these assets in an independent data structure, indexed by their IDs. See Figure 5.

5. Illumination computations

In this section we will describe the different stages of our algorithm. Firstly, the pre-process is described, presenting the actions taken for initializing the illumination computations. These computations are done once for every scene. Then, the run-time steps, i.e. those computed at every time the lighting conditions change, are presented and enumerated. These include the set-up, the low-frequency and the final high-frequency passes.

5.1. Pre-processing: Patch Links

The two observations mentioned in Section 3 lead to some conclusions that will shape the way we pose the hierarchical radiosity algorithm. First of all, our implementation will not need to *refine* any geometry, but descend the previously created hierarchy. On the other hand, in the bibliography there have been defined two main oracles [CWH93]: the first one generates the links based solely on geometric form factors, while the second also included an exchange-of-energy-based criterion. The main consequence of the first approach is that

the links between primitives can be created in pre-process time, as long as the geometry does not change, as is our case. As the hierarchies of all building facades are created in the pre-process procedural modeling stage, and the buildings are static, it is a sensible idea to also create the links in the pre-processing. As a consequence, our hierarchies are still valid for any sun position. We used this conservative approach that generates links that could result in little exchange of energy, but that is perfectly compatible with our objective of reducing to a minimum the computations that are needed for each sun position. This makes our pre-processing stage independent of any sun position, thus allowing faster runtime computations.

Algorithm 1 findLinks(Node n_1, Node n_2)

```
Node aux = oracle(n1, n2)
if aux != NULL then
  if aux = n1 then
    for all children from n1 do
      findLinks(child, n2)
    end for
  else if aux = n2 then
    for all children from n2 do
      findLinks(n1, child)
    end for
  end if
end if
```

Our final implementation, thus, generates the links in a pre-processing stage which is described in Algorithm 1. This algorithm should iteratively be called with all pairs of upper-level patches, and will generate the links between the respective hierarchies at the right levels. This algorithm is a recursive one that first calls the *oracle* function, shown in Algorithm 2. Then, depending on the result of that evaluation, it continues the recursion on the children of one node or the other. In the case *oracle* returned that no interaction is to be set, the algorithm finishes without further actions.

Algorithm 2 oracle(Node n_1, Node n_2) Return Node

```
float FF12 = formFactor(n1,n2)
float FF21 = formFactor(n2,n1)
Node res = NULL
if distance(n1,n2) < MIN_DISTANCE then
  if FF12 > MIN_FORM_FACTOR then
    res = n1
  else if FF21 > MIN_FORM_FACTOR then
    res = n2;
  else
    createLink(n1,n2)
  end if
end if
return res
```

The *oracle* function, described in Algorithm 2, starts by computing the respective form factors and determines whether one patch should be subdivided, the other one, both patches are all right to exchange energy at this level, or are too far away to be considered for energy exchange. This latter threshold was set to be 1.5 times the width of a standard, medium-sized street. The rationale behind this decision is to bound the interactions to a small scale that produces significant results but does not hinder computation speed. See Figure 6.

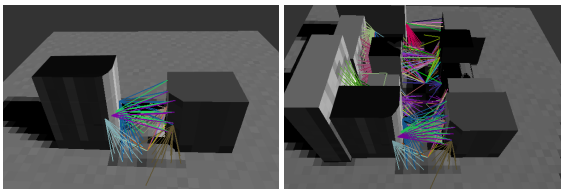

Figure 6: *Links between two buildings and for a street segment. In both cases we show only the links originating from the left.*

5.2. Set-up pass

The overall illumination pass starts by doing a rendering step computed from the sun position with an orthographic camera to emulate a directional light. This stage is computed entirely in an off-line buffer on the GPU, rendering all the building geometries at its maximum quality with all its respective assets (but no texturing information). For each asset instance, it is rendered with the ID of the corresponding primitive in one color channel. Thus, the final buffer will contain, for each asset, the ID of the corresponding primitive (plus depth information that will be used in the final stage). Then, we compute the number of times (i.e., pixels) each primitive ID appears on the buffer, and load its initial energy with this count times an energy-equivalence factor. Observe that in the previous stage we rendered assets with the IDs of the corresponding primitives, correctly resolving occlusion problems and providing a more accurate indicator of the exact amount of energy each asset/primitive receives. From this stage we also keep a depth buffer for computing the high-frequency direct illumination in the last stage.

5.3. Low-frequency pass

Once the direct illumination is computed in the previous stage, the energy exchange is calculated. Initially, all the primitives in the lowest hierarchy levels were initialized with their respective incoming radiosity B_s, and we set the outgoing radiosity B_g with this value. Then, for each link created in the first stage, we compute for both patches participating in the exchange their new values. For each one, the new outgoing radiosity is computed as the form factor times the outgoing radiosity of the other patch and the reflectance coefficient, as

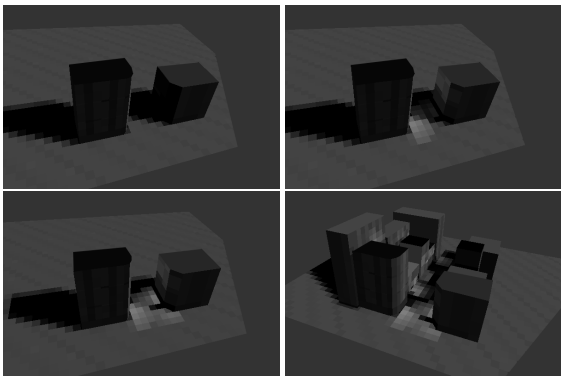

Figure 7: *Quality for a different number of iterations. The result for two buildings without iteration (top, left) and with 1 iteration (top, right) and 2 iterations (bottom, left). Finally, the same environment with a few buildings with with 3 iterations (bottom, right).*

$$n_1.B_s += FF_{12} * n_2.B_g * n_1.\rho$$

$$n_2.B_s += FF_{21} * n_2.B_g * n_2.\rho$$

This is iterated a number of times, which is chosen according to the convergence of the result, see Section 6.

5.4. High-frequency pass

The last stage combines the high-frequency information gathered in the first pass with the low-frequency illumination component computed in the second one. This is done with a fragment shader in the GPU which takes the depth information generated during the set-up pass, and enhances with the primitive-based soft global illumination. The last term is passed to the shader as fragment parameter by associating each primitive vertex with the respective intensity, and letting the rasterization pipeline to pass this information. If the per-vertex information is generated by averaging the indirect illumination at the primitives that share this vertex, then the vertex primitives can have different energies. In that case, the graphics hardware would automatically interpolate the illumination to get a smooth shading. However, for other kinds of simulations (e.g., solar radiation measures) interpolation would produce biased results that should be avoided.

6. Results and discussion

We can analyze the error related to the number of iterations in the hierarchical radiosity algorithm. The error is computed as the relative difference:

$$E_i = \sqrt{\sum_{patches} (B_g^i - B_g^{i-1})^2}$$

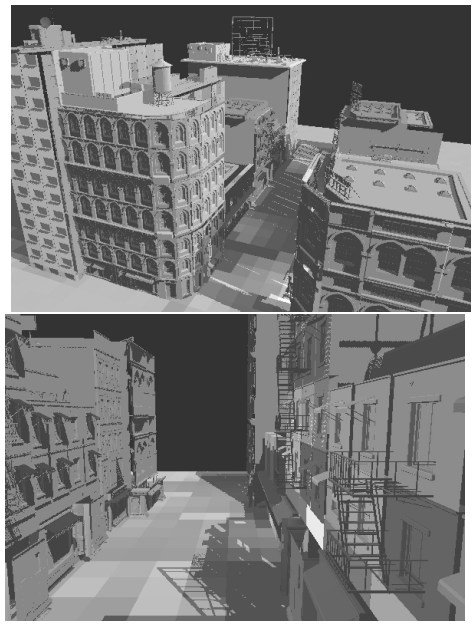

Figure 8: *An urban environment illuminated with global illumination resulting from our algorithm.*

where i is the iteration number starting from the second. We have observed that, after the third iteration, the error stabilizes from an initial value of 8.1 to a value of 3.7 and then decreases more slowly, which is acceptable for our purposes. As a result, we have decided to cut the number of iterations to three, which also agrees with the decision to have 1.5 times the average street width as threshold for the link creation.

At Figure 7 we can observe the evolution of the algorithm for a different number of iterations. As we can see, results improve with the number of iterations at the expense of increased computation times.

At Figure 8 we can see a converged image of our test scene, with the global illumination and high-frequency local illumination together. We have left the diffuse global illumination at the patch level to be able to appreciate the non-distorted results. For this scene, the simulation ran at 10 FPS.

7. Conclusion and Future Work

We have presented the first prototype for a hierarchical radiosity-based solution for urban environments. Contrary to traditional radiosity approaches, the one presented here reuses the hierarchy naturally produced by the modeling process to incorporate the illumination computations. This produces, for the first time, a smooth blend between the procedural modeling generation and the hierarchical radiosity computation.

A direct improvement to our method is the use of the power of current GPUs, so implementing the core parts in CUDA or OpenCL is an interesting line for study. We think this new blending between proven techniques opens the door to more complex calculations, like extending this approach at the full urban scale, and integrating these simulations into the more general framework of the study of the urban climate. These extensions to the domain of urban physics become now naturally integrated with procedural techniques, which is, in our humble opinion, a very promising avenue for further research.

Acknowledgements

This work was partially funded with grant TIN2010-20590-C02-02 from Ministerio de Educación y Ciencia, Spain.

References

[AAP12] ARGUDO O., ANDÚJAR C., PATOW G.: Interactive rendering of urban models with global illumination. In *Computer Graphics International* (Bournemouth University, United Kingdom, June 2012). 2

[ABCN10] ANDUJAR C., BRUNET P., CHICA A., NAVAZO I.: Visualization of large-scale urban models through multi-level relief impostors. *Computer Graphics Forum 29*, 8 (2010), 2456–2468. 1, 2

[AYRW09] ALI S., YE J., RAZDAN A., WONKA P.: Compressed facade displacement maps. *IEEE Transactions on Visualization and Computer Graphics 15*, 2 (2009). 2

[Buc05] BUCHHOLZ H.; DOLLNER J.: View-dependent rendering of multiresolution texture-atlases. In *IEEE Visualization* (2005), pp. 215–222. 2

[CBG*07] CIGNONI P., BENEDETTO M. D., GANOVELLI F., GOBBETTI E., MARTON F., SCOPIGNO R.: Ray-casted blockmaps for large urban models visualization. *Computer Graphics Forum 26*, 3 (2007), 405–413. 1, 2

[CBZ*08] CHANG R., BUTKIEWICZ T., ZIEMKIEWICZ C., WARTELL Z., RIBARSKY W., POLLARD N.: Legible simplification of textured urban models. *IEEE Computer Graphics and Applications 28*, 3 (2008), 27–36. 2

[CWH93] COHEN M. F., WALLACE J., HANRAHAN P.: *Radiosity and realistic image synthesis*. Academic Press Professional, Inc., San Diego, CA, USA, 1993. 2, 3

[DB05] DÖLLNER J., BUCHHOLZ H.: Continuous level-of-detail modeling of buildings in 3d city models. In *Proc. of the ACM international workshop on geographic information systems* (2005), GIS'05, pp. 173–181. 2

[DB11] DI BENEDETTO M.: *Multiresolution Techniques for Real-Time Visualization of Urban Environments and Terrains*. PhD thesis, Università degli Studi di Pisa, 2011. 2

[DBCG*09] DI BENEDETTO M., CIGNONI P., GANOVELLI F., GOBBETTI E., MARTON F., SCOPIGNO R.: Interactive remote exploration of massive cityscapes. In *Proc. of the International Symposium on Virtual Reality, Archaeology and Cultural Heritage* (2009). 1, 2

[HSA91] HANRAHAN P., SALZMAN D., AUPPERLE L.: A rapid hierarchical radiosity algorithm. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1991), SIGGRAPH '91, ACM, pp. 197–206. 2

[HW10] HAUNERT J.-H., WOLFF A.: Optimal and topologically safe simplification of building footprints. In *Proc. of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems* (2010), GIS '10, pp. 192–201. 2

[JW02] JESCHKE S., WIMMER M.: Textured depth meshes for real-time rendering of arbitrary scenes. In *Proc. of the Eurographics workshop on Rendering* (2002), EGRW '02, pp. 181–190. 2

[LBZ*10] LIANG C., BACIU G., ZHANG J., CHAN E. C. L., LI G.: Footprint-profile sweep surface: a flexible method for realtime generation and rendering of massive urban buildings. In *Proc. of the 17th ACM Symposium on Virtual Reality Software and Technology* (2010), VRST '10, pp. 151–158. 1, 2

[LWW08] LIPP M., WONKA P., WIMMER M.: Interactive visual editing of grammars for procedural architecture. *ACM Transactions on Graphics 27*, 3 (Aug. 2008), 102:1–10. 2

[MPAP13] MUNOZ-PANDIELLA I., ANDUJAR C., PATOW G.: NightLighting: a Nocturnal Urban Illumination Approach. In *Eurographics Workshop on Urban Data Modelling and Visualisation* (Girona, Spain, 2013), Tourre V., Besuievsky G., (Eds.), Eurographics Association, pp. 13–16. 2

[MS95] MACIEL P. W. C., SHIRLEY P.: Visual navigation of large environments using textured clusters. In *Proc. of the 1995 symposium on Interactive 3D graphics* (1995), I3D '95, pp. 95–ff. 2

[MWH*06] MÜLLER P., WONKA P., HAEGLER S., ULMER A., VAN GOOL L.: Procedural modeling of buildings. *ACM Trans. Graph. 25*, 3 (2006), 614–623. 2

[PSC10] PINA J., SERON F., CEREZO E.: Bqr-tree: A data structure for flights and walkthroughs in urban scenes with mobile elements. *Computer Graphics Forum 29*, 6 (2010), 1745–1755. 2

[SDB97] SILLION F. X., DRETTAKIS G., BODELET B.: Efficient impostor manipulation for real-time visualization of urban scenery. *Computer Graphics Forum 16*, 3 (1997), 207–218. 2

[TMJ98] TANNER C. C., MIGDAL C. J., JONES M. T.: The clipmap: a virtual mipmap. In *Proc. of SIGGRAPH '98* (New York, NY, USA, 1998), ACM, pp. 151–158. 2

[VAW*10] VANEGAS C. A., ALIAGA D. G., WONKA P., MÃIJLLER P., WADDELL P., WATSON B.: Modelling the appearance and behaviour of urban spaces. *Computer Graphics Forum 29*, 1 (2010), 25–42. 2

[WWS00] WONKA P., WIMMER M., SCHMALSTIEG D.: Visibility preprocessing with occluder fusion for urban walkthroughs. In *Proc. of the Eurographics Workshop on Rendering Techniques 2000* (2000), pp. 71–82. 2

[WWS01] WIMMER M., WONKA P., SILLION F.: Point-based impostors for real-time visualization. In *Proc. of the Eurographics Workshop on Rendering 2001: London, United Kingdom, June 25-27, 2001* (2001), Springer Verlag Wien, pp. 163–176. 2

[WWSR03] WONKA P., WIMMER M., SILLION F., RIBARSKY W.: Instant architecture. *ACM Transaction on Graphics 22*, 3 (July 2003), 669–677. Proceedings ACM SIGGRAPH 2003. 2

[YZM*11] YANG L., ZHANG L., MA J., XIE J., LIU L.: Interactive visualization of multi-resolution urban building models considering spatial cognition. *International Journal of Geographical Information Science 25* (February 2011), 5–24. 2