Integration of Geomorphing into Level of Detail Management for Realtime Rendering

Christopher Zach[*]

VRVis Research Center

Abstract

Realtime rendering of scenes with discrete levels of detail (LOD) often suffers from noticeable visual changes between succesive frames. We propose geomorphing to obtain smoother animations while retaining guaranteed frame rates. Our level of detail management calculates a set of representations, that are well suited for some future time interval according to the predicted motion of the user. Thus, the rendering system has enough time to change the representations smoothly to the desired level of detail. Further on uniprocessor systems, the time used by the LOD selection is amortized between several frames.

Keywords: Flyover setting, Realtime rendering, Geomorphing

1 Introduction

Our department at the VRVis Reseach Center aims on the photorealistic realtime visualization of virtual cities and surrounding landscapes that are reconstructed from aerial and terrestrial images and data from laser scans. Currently the acquisition of a digital model of Graz (Austria) is an ongoing task. A view on the already gathered data is shown in Figure 1. The currently available dataset for Graz consists of 3300 buildings, that were automatically generated from roof and eave lines. The total polygon count for the architectural data is more than 200 thousand, but we expect the final model containing 40 thousand buildings and 40 million triangles at least.

We focus on realtime rendering of urban areas with guaranteed frame rates to ensure smooth navigation through the virtual environment, even if the complexity of the scene is increasing. Therefore the rendering system is allowed to sacrifice image quality to minimize lags in the displayed images. We strive for at least 20 frames per second on contemporary hardware.

The demands on the rendering system range from terrain flyover scenarios to close inspection of facade detail during a walkthrough. The following topics are adressed in our work:

- Level of detail generation for urban data sets
- Level of detail management and selection at run-time
- Retrieval and caching of necessary parts of the scene database
- Improved photorealism through rendering of artificial generated objects (vegetation)

The third topic was already discussed in [12]. In this paper we treat the run-time selection of appropriate levels of detail for each frame.

2 Related Work

2.1 Basic Frame Rate Control

Most scene graph libraries use distance based or screen size based LOD switching to accelerate rendering of complex scenes. The render time for each frame depends on the scene content, the viewing parameters and the LOD switching threshold, but it can be arbitrary large. In terms of optimization these methods maximize the frame rate subject to controlled image quality. Whenever an guaranteed interactive frame rate is needed, the role of frame rate and image quality must be exchanged: maximize the image quality that is achievable subject to controlled render time.

Funkhouser and Sequin [2] formulated this optimization task as a multiple choice knapsack problem (MCKP), which is known to be NP-hard, and used an approximation method to select the appropriate LOD for each object to be rendered. The importance of every object and the accuracy of every LOD depend on the viewing parameters, thus the MCKP must be solved for every frame.

Solving the MCKP needs a certain amount of time, which may affect the rendering performance on a uniprocessor computer. Funkhouser and Sequin used a dual processor solution for rendering: one processor selects the representations for the next frame and the other processor feeds the graphics pipeline.

It is important to mention that the solutions of the MCKP for succesive frames may be very different: changing the viewing parameters slightly may yield to sudden changes in the representation for visible objects. These discontinuities between frames are often called “popping artefacts.” Funkhouser and Sequin considered utilizing a hysteresis factor to penalize switching between levels of detail, but they did not incorporate it into their implementation.

[*]zach@vrvis.at

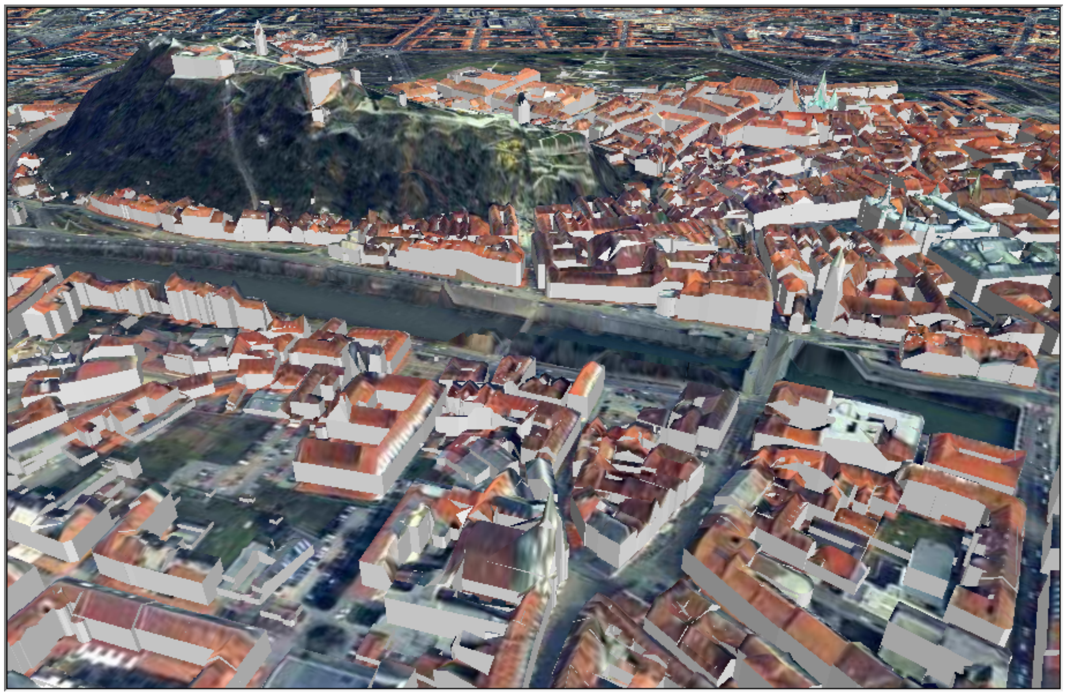


Figure 1: A view on the virtual center of Graz

2.2 Realtime Rendering of Hierarchies

One major weakness of the approach presented above is, that visible objects may be completely missing in the rendered image, if the allowed render time is not large enough to render at least the coarsest representation of every visible object. In our application this could yield to missing buildings or missing terrain patches. Mason and Blake [8] utilized a hierarchical level of detail approach and used a variation of the MCKP to select appropriate representations for every frame. Their approximation method of this extended MCKP can be seen as a greedy traversal of the LOD hierarchy. Further they observed that their approximation scheme (and the one proposed by Funkhouser and Sequin) is correct, if the value of higher LOD representations is diminishing. This means that rendering a representation with twice the number of polygons adds less than twice to the total image quality.

2.3 Geomorphing

If a smooth transition between different discrete levels of detail is desired, two solutions are possible: Blending between the two levels using the alpha channel or geometrically morphing between the two models [7]. The first approach requires both levels to be rendered during the blending phase, therefore reducing the available render time. Further, blending of two levels may result in visually not well defined images. Geomorphing interpolates the vertices of the two levels at every frame during the transition, therefore it requires a mapping of corresponding vertices in the source and destination representation. The complexity of the geomorph is equal to the complexity of the higher level of detail.

2.4 Multiresolution Models

Multiresolution models [6], [3] provide an almost continuous sequence of levels of detail, therefore switching between different levels can be naturally made smooth. Additionally the mapping of vertices between representations is often very straightforward and geomorphing is well supported.

Although multiresolution modeling is a very promising technique, there are still several reasons for conventional discrete LODs. Although multiresolution analysis is very successful on reconstructed natural surfaces with a dense mesh, they may be not appropriate for a specific visualization task. In such cases a specific LOD generator that produces high quality discrete levels is often preferable. We observed, that architectural objects yield to multiresolution models with low quality. Additionaly, discrete LODs can be converted offline into triangle strips and fans for faster rendering. For these reasons we decided to build our visualization system on a discrete and hierarchical level of detail approach.

An interesting approach for realtime rendering of multiresolution models in presented in [5]. The authors employ an active set method (an extension of the simplex method for nonlinear problems) to select suitable representations from a multiresolution model.

3 Our Approach

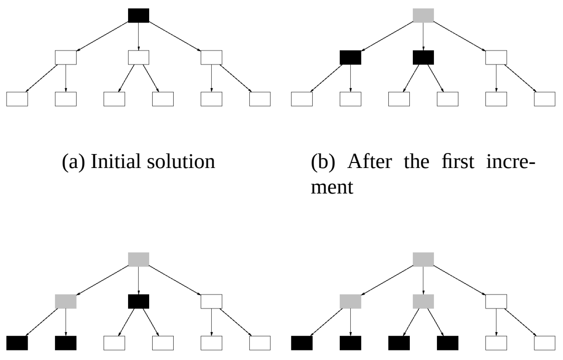
3.1 Terminology

Our terminlogy follows mostly [10]. Let $\{o_i\}$ be a set of distinguished objects (buildings, for example). Every object o_i can have multiple representations r_{ij} in different level of detail or using different rendering methods. One representation may cover several objects; the root representation, for example, is a very coarse representation for all objects. Since in our application different representations correspond to various levels of detail, we use these terms interchangeably.

For given viewing parameters every representation r_{ij} has some positive impact on the rendered image. This quantity is called the benefit of r_{ij} and denoted by $b_{ij}(\cdot)$. The argument given to each $b_{ij}(\cdot)$ consist of the viewing parameters. For convenience we will write $b_{ij}(t)$, if we refer to the benefit according to the (possibly predicted) viewing parameters at time t. It is assumed, that the total image quality can be approximated by the sum of benefits for the rendered representations. The benefit of invisible objects is 0. Rendering r_{ij} has some cost measured by c_{ij}, which is usually the (estimated) render time for r_{ij}. For simplicity we assume, that these costs are independent of the viewing parameters. The ratio $b_{ij}(\cdot)/c_{ij}$ is called the value of r_{ij}. The goal of the basic LOD selection process is to choose a set of r_{ij}, that maximizes the total benefit, such that every visible object is represented exactly once and the available render time for this frame is not exceeded.

Basically an approximate solution to this optimization problem can be obtained by sorting the r_{ij} according to their values in decreasing order and to select the most valuable representations until the cost limit is reached. Some modifications are necessary to handle the additional constraint, that every visible object must be represented exactly once, but the basic idea remains the same.

We give a brief overview of the approximation method for this case (see [8] for details). The representations r_{ij} are organized as a tree, where the leave nodes correspond to objects in full level of detail. The method performs a sequence of so called increment operations, until the next increment would exceed the available render time. The increment operation replaces a node in the current solution with its children, that are most valuable. The root node is the initial solution. An example for a sequence of increment operations is given in Figure 2.


(a) Initial solution

(b) After the first increment

(c) After the second increment

(d) After the third increment

Figure 2: A sequence of possible increments. Black nodes comprise the current solution and grey nodes mark already improved (and replaced) representations. The rightmost subtree is invisible and therefore not rendered.

This procedure returns a 2-approximation (ie. the value of the objective function is at least half the true optimal value), if the value of children (better representations) is less than the value of the parent node.

3.2 Multiple Hierarchies

We extended the framework described in Section 2.2 slightly to incorporate multiple hierarchies. For our purpose we need at least two separate hierarchies: one represents the buildings and one consists of a quadtree based terrain model.

Instead of performing a greedy traversal of one tree, our LOD selection scheme traverses several trees in parallel. The initial solution consists of the root nodes of all trees (if the root node is visible at all). Thus, it is assumed this initial solution is feasible (ie. this solution does not exceed the available render time). The increment operation is applied for the tree that results in the best improvement. This increment steps are repeated until the available render time is consumed. This method gives more render time to buildings, if the user navigates within the city and assigns more render time to the terrain model, when the landscape is visually more important.

3.3 LOD Selection for Several Frames

For complex scenes with a large number of possible representations available the LOD selection process may require a significant amount of time, which heavily affects the rendering performance on uniprocessor systems. Therefore it is preferable to distribute the LOD selection time to several frames and reuse the last solution until the new solution is available. This idea is somewhat similar to the *instant visibility pipeline* presented in [11].

Since the current selection process will take effect in the future and the generated solution will be used for a certain number of frames, the selection process should know the future viewing parameters. Since in applications with unconstrained user motion this is not possible, we employ path prediction to obtain probable future view points and directions.

Rendering and computing the appropriate LOD is done asynchronously. Let t resp. τ be the starting time resp. the planning horizon of the LOD selection process. During the time interval $[t, t+\tau)$ the recently calculated LODs are rendered and the set of representations valid for the time interval $[t+\tau, t+2\tau)$ is computed. This temporal dependence is illustrated in Figure 4.

The predicted path at time $[t+\tau, t+2\tau)$ is sampled and the benefit of a representation is the average of the benefits computed for the sampled viewing parameters:

$$\hat{b}_i = \sum_{t_k} b_i(t_k).$$

Since the future viewing parameters can only be estimated, we need to compensate for erroneous prediction. At least we want that every potentially visible object in the future time interval is rendered, even if the choosen representation is too coarse for the actual viewing parameters. To limit the set of potentially visible objects we assume that the maximal translational velocity and the maximal angular velocity is restricted by θ resp. ω. An additional camera with a field of view of $fov + 4\tau\omega$ and projection center

$$eyePos_t - \frac{2\tau\theta}{\sin(fov/2 + 2\tau\omega)}\, eyeDir_t$$

is introduced, where fov denotes the field of view of the original camera. This view volume is a conservative approximation of the union of all view volumes, that can be reached by the user in time 2τ. The derivation of this view volume is shown in Figure 3.

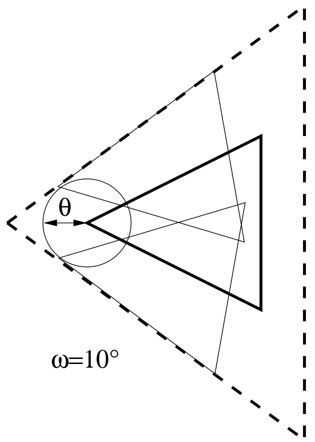


Figure 3: The derivation of the conservative viewing pyramid, if the user changes his position about θ resp. his viewing direction about ω at most. The current view volume is the thick and solid outlined pyramid, whereas the conservative volume has a dashed border. The two view volumes with the thin and solid border correspond to extreme viewing parameters.

Let b'_i be the benefit of every representation according to this camera. The total estimated benefit $\bar{b}_i$ is computed as $\hat{b}_i + \lambda b'_i$, where λ is a constant, that encodes the confidence about the path prediction. This augmented benefit function is used to calculate the appropriate levels of detail using the method described in Section 3.1.

Since the render time per representation may depend on the viewing parameters (as it is the case in [2]), the estimated render cost is the maximum of the sampled costs.

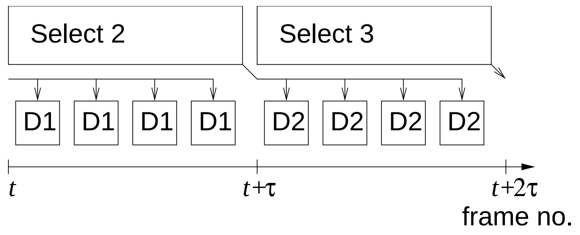


Figure 4: Asynchronous LOD selection without morphing. Select i represents the i-th LOD selection process and Di corresponds to drawing a frame based on the i-th selection.

3.4 Integration of Geomorphing

Additionaly we integrated smooth transitions between different representations into our framework. A two–step pipeline as described in the previous section is not sufficient, since the currently running LOD selection procedure may finish at any time within the interval $[t, t+\tau]$ and hence the available time for morphing different levels might by arbitrarily short. To compensate for this we utilize a three step pipeline. We assume, that τ is the duration for morphing between two levels, too. During the interval $[t, t+\tau)$ the representations morph to the (already computed) levels valid for the interval $[t+\tau, t+2\tau)$. In parallel the probably appropriate LODs for the interval $[t+2\tau, t+3\tau)$ are calculated. From time $t+\tau$ until $t+2\tau$ the rendered representations will finally morph to the obtained levels. Now the interval for predicting the samples

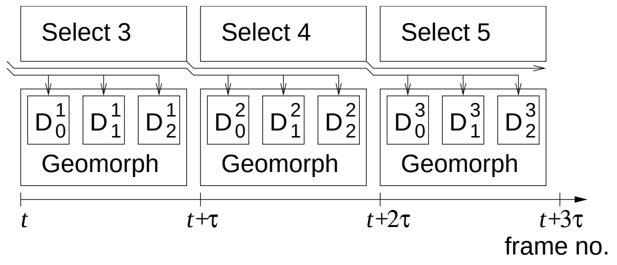


Figure 5: Asynchronous LOD selection with morphing. D^i_j draws the levels selected in the corresponding phase Select i with the weights according to the morphing step j.

is $[t+2\tau, t+3\tau)$. Further, the view volume of the conservative camera must be enlarged: the field of view is now

$fov + 6\tau\omega$ and the position is

$$eyePos_t - \frac{3\tau\theta}{\sin(fov/2 + 3\tau\omega)}\, eyeDir_t.$$

For convenience we introduce two terms: geomorphing a coarser representation to a more accurate one is called upmorphing, while the reverse operation is called downmorphing.

Upmorphing and downmorphing are not symmetric in time, because the complexity of the morphed representation is always the complexity of the more detailed one. If the representation valid for $[t, t + \tau]$ is more accurate than the representation used for $[t + \tau, t + 2\tau]$, the downmorphing occurs during $[t, t + \tau]$, such that the coarse level is reached at the beginning of the desired interval $[t+2\tau, t+3\tau]$. This ensures, that the coarser and less time consuming representation is used during this interval.

On the other hand, if the selected level in the later interval $[t+2\tau, t+3\tau]$ is higher than the choosen representation for the previous interval, the better representation cannot be established at the beginning of the target interval, since this would violate the cost limit in the preceeding interval. Thus, the upmorphing must take place during the later interval $[t + 2\tau, t + 3\tau]$, such that the desired resolution is obtained at the end of this interval. Figure 6 illustrates this situation.

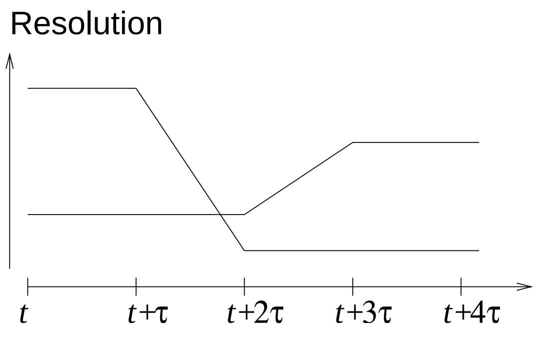


Figure 6: Upmorphing and downmorphing. The interval, for which the representations are computed, is $[t + 2\tau, t + 3\tau]$. The coarse models are available at the beginning of the target interval, while the more accurate model has finished the morphing not until the end of the interval.

Some complications arise because of this unsymmetry: Assume, that the first LOD selection procedure implies the upmorphing of an object, but the consecutive LOD selection process requests the downmorphing of the same object. Since the upmorphing is delayed, but the downmorphing is not, both morphing operations refer to the same time interval. In this case, the requested two transitions collapse into one, such that the geomorphing takes place from the level valid at the beginning of the conflicting interval to the level required at the end of the interval. This conflict does not appear for a sequence of monotone geomorphing or downmorphing followed by upmorphing.

Figure 7 shows the possible combinations of morphing directions.

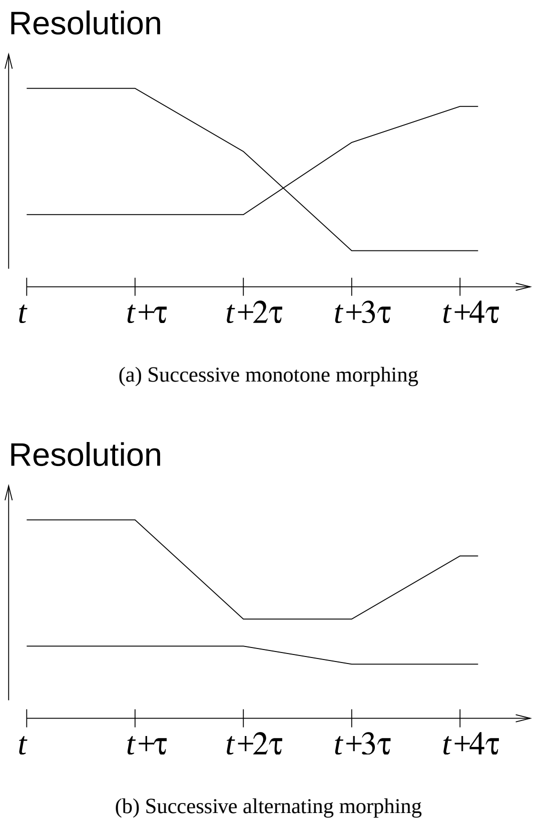


Figure 7: A sequence of 2 upmorphing and downmorphing operations. (a) shows the application of successive monotone transitions and (b) illustrates alternating morphing directions. This situation may arise, when the user moves initially forwards and subsequently backwards. The upmorphing for the coarser representation is overridden by the downmorphing requested by the sequent LOD selection.

4 Implementation

4.1 LOD generation

Initially we used quadric error metrics [4] as a general purpose LOD generator, which is not especially suited for architectural models. Recently a volumetric level of detail generator based on octree decomposition [1] is available, that is targeted at simplifying man made structures.

4.2 Benefit and Cost Computation

The benefit of a representation is computed similar to the method described in [10]. The benefit depends on the viewing parameters $(eyePos, eyeDir)$ according to the following rule:

$$b_{ij} = accuracy \times screenSize(o_i),$$

where

$$accuracy = \frac{\max(1, \frac{screenSize(r_{ij})}{\#orginalfaces})}{\max(1, \frac{screenSize(r_{ij})}{\#faces(r_{ij})})}.$$

Here $screenSize(\cdot)$ is the estimated screen size of the bounding box according to the given viewing parameters. This benefit rule assigns diminishing values to more accurate representations.

The rendering cost for each representation is not computed entirely predictive as in [2]. Since our render primitives consist of mostly long triangle strips, we assume, that the render time per representation is proportional to the number of vertices sent to the graphics pipeline, ie. $renderTime = c_{vertex} \times \#vertices$. Since this estimated cost is view independent and the total number of state changes is not known during LOD selection, we compensate these error sources using a feedback strategy. After rendering a frame, the actual render time is measured and c_{vertex} is updated accordingly (using a weighted average scheme).

4.3 Path Prediction

We employ a linear path prediction method, that estimates the future view point, the viewing direction and the up vector of the future projection. The attempts to utilize low order polynomials were not successful, since even quadratic polynomials oscillate too much to be useful in extrapolating regions far outside the sampled interval.

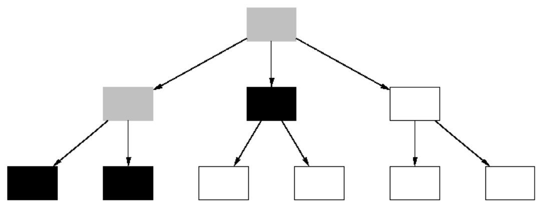
The path prediction method must be adapted for the degrees of freedom provided by the user interface. The flyover viewer of OpenInventor is rather easy to use, but it allows abrupt changes of the path. For an application in a public place (e.g. museum or exhibition) an interface with fewer degrees of freedom is preferable to reduce the probability of getting lost in the virtual environment.

4.4 Geomorphing

We computed vertex correspondences for successive levels of detail using the nearest vertex rule: a vertex in the finer representation r_{ij} is mapped on the closest (according to the euclidian distance) vertex in r_{ij-1}. Initially we allowed geomorphing between arbitrary levels of detail, which caused visually apparent problems, since these vertex correspondences are not topology preserving. Whenever a transition between nonadjacent levels is requested, a sequence of geomorphs is generated. For instance, if r_{ij} should be finally replaced by r_{ij+d}, d successive geomorphing operations within the time interval of length τ are performed: From r_{ij} to r_{ij+1}, then from r_{ij+1} to r_{ij+2} and so on.

The set of necessary transitions is computed as follows: the set R_1 of representations rendered from $[t+\tau, t+2\tau]$ is compared with the set R_2, that is valid for $[t+2\tau, t+3\tau]$. If a representation in R_2 has a (direct or indirect) parent in R_1, a transition to a higher level is registered. On the contrary, if a representation in R_2 is a coarse version of an item in R_1, a morphed transition to a coarser representation is registered. Other levels in R_2 are not affected by morphing.

This relationship between the recent and new solution can be efficiently computed during the increment operation. Those nodes in the hierarchy, that were part of the previous solution, are marked black. Therefore it is easy to discover, whether these nodes are replaced by a better representation during the current LOD selection process. Nodes in the current solution, that are coarser levels of previously selected nodes, are determined similarly. To achieve this, all predecessors of black nodes are labeled grey. If a node in the current solution was marked grey during the previous selection procedure, its grey subnodes are followed, until black nodes are reached. Every other node in the current solution needs no morphing. The relationship between sequent solutions is illustrated in Figure 8.


(a) Previous solution

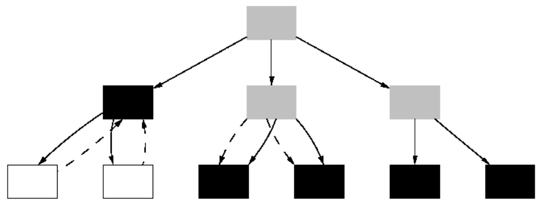

(b) Current solution

Figure 8: An example of two consecutive solutions produced by the LOD selection procedure. Black nodes are members of the solution and gray nodes were visited. Dashed edges represent geomorphs. The rightmost subtree needs no geomorphing, because it was not member of the previous solution (and therefore not rendered at all).

A sequence of successive frames are shown in Figure 9.

4.5 Choosing τ

Choosing the right value for τ implies a tradeoff between two competing objectives: τ should be large enough to compensate for the higher complexity of the selection process (because of sampling several benefits). We evaluate the benefit for three probable future cameras and for the conservative camera, therefore the LOD selection process will run approximately four times longer than the LOD selection for individual frames and τ should at least cover 4 frames. On the other hand, τ should be as small as possible to to improve path prediction. If the target frame rate is 20 frames per second, choosing $\tau = 0.5s$ yields to a LOD selection procedure every 10 frames and the predicted interval ends 1.5 seconds in the future.

5 Conclusions and Future Work

We presented a realtime rendering framework, that integrates geomorphing to enhance the visual quality of image sequences in walkthrough and flyover applications. Our approach guarantees desired frame rates and can be integrated into existing scene graph libraries.

Further work is needed to enhance the realism of the rendered scenes. This includes objects that are not acquired by the reconstruction process, e.g. trees. We consider point based rendering [9] to represent vegetation.

Additionally our method of finding corresponding vertices in successive levels of detail needs enhancements, because the mesh obtained from the better representation with shifted vertices is not necessary topologically equivalent to the mesh of the coarser representation. In such cases visual artefacts are the result.

On uniprocessor systems the priority of several threads must be assigned according to the current demands (load balancing). The LOD selection thread and the render thread are intrinsically synchronized, but our application uses another thread for the communication with the scene database to retrieve the requested parts on demand. Whenever the retrieval from the scene database is delayed, the threads concerning rendering should decrease their priority to allocate more resources to fulfill the retrieval requests. On the other hand more resources should be assigned to the render thread, if the retrieved scene is already of high complexity.

References

[1] C. Andujar, D. Ayala, and P. Brunet. Validity preserving simplification of very complex polyhedral models. In *Proceedings of 5th Eurographics Workshop on Virtual Environments, EGVE'99*, 1999.

[2] Thomas A. Funkhouser and Carlo H. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. *Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 247–254, 1993.

[3] M. Garland. Multiresolution modeling: Survey & future opportunities. In *Eurographics '99 – State of the Art Reports*, pages 111–131, 1999.

[4] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. *Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 209–216, 1997.

[5] Enrico Gobbetti and Eric Bouvier. Time-critical multiresolution rendering of large complex models. *Journal of Computer-Aided Design*, 32(13):785–803, 2000.

[6] Hugues Hoppe. Progressive meshes. *Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 99–108, 1996.

[7] Hugues H. Hoppe. Smooth view-dependent level-of-detail control and its application to terrain rendering. In *IEEE Visualization '98*, pages 35–42, 1998.

[8] A. E. W. Mason and E. H. Blake. Automatic hierarchical level of detail optimization in computer animation. *Computer Graphics Forum*, 16(3):191–200, 1997.

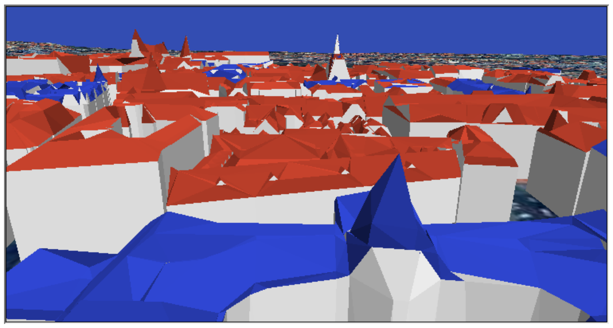
[9] Szymon Rusinkiewicz and Marc Levoy. QSplat: A multiresolution point rendering system for large meshes. In *Computer Graphics (SIGGRAPH 2000 Proceedings)*, pages 343–352, 2000.

[10] Eyal Teler and Dani Lischinski. Streaming of complex 3D scenes for remote walkthroughs. *Computer Graphics Forum*, 20(3):17–25, 2001.

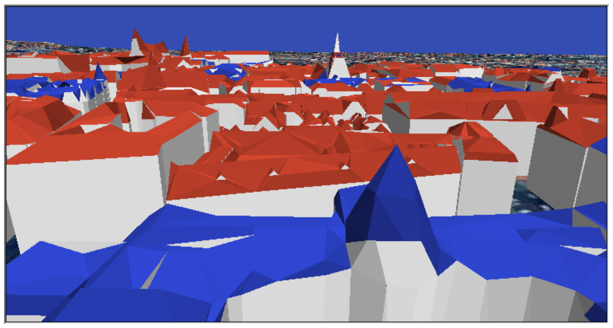
[11] Peter Wonka, Michael Wimmer, and François X. Sillion. Instant visibility. *Computer Graphics Forum*, 20(3):411–421, 2001.

[12] Christopher Zach and Konrad Karner. Prefetching policies for remote walkthroughs. In *Proceedings of the WSCG*, 2002.

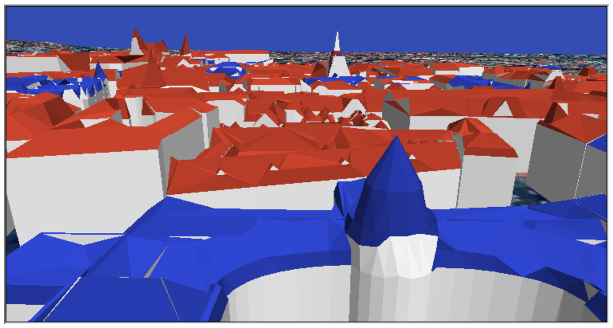
This work has been done in the VRVis research center, Graz and Vienna/Austria (http://www.vrvis.at), which is partly funded by the Austrian government research program Kplus.



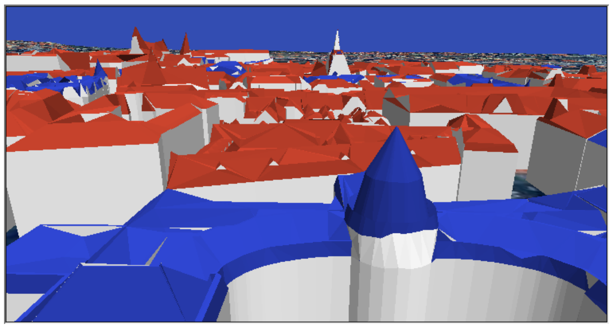
(a) Frame 0



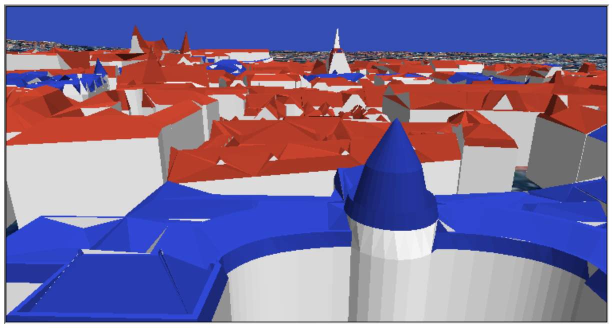
(b) Frame 1



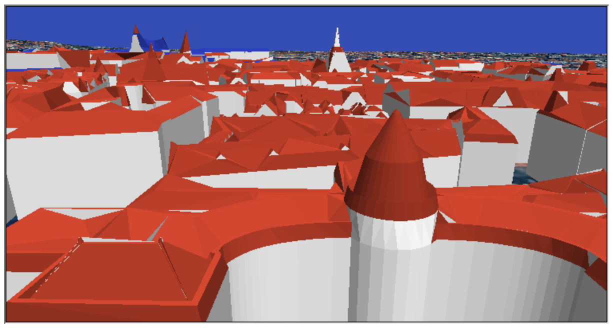
(c) Frame 2



(d) Frame 3



(e) Frame 4



(f) Frame 5

Figure 9: A sequence of frames. Buildings, that are geomorphs, have blue roofs.