Image Filtering for Interactive Level-of-Abstraction Visualization of 3D Scenes

Amir Semmo Jürgen Döllner

Hasso Plattner Institute, Germany*

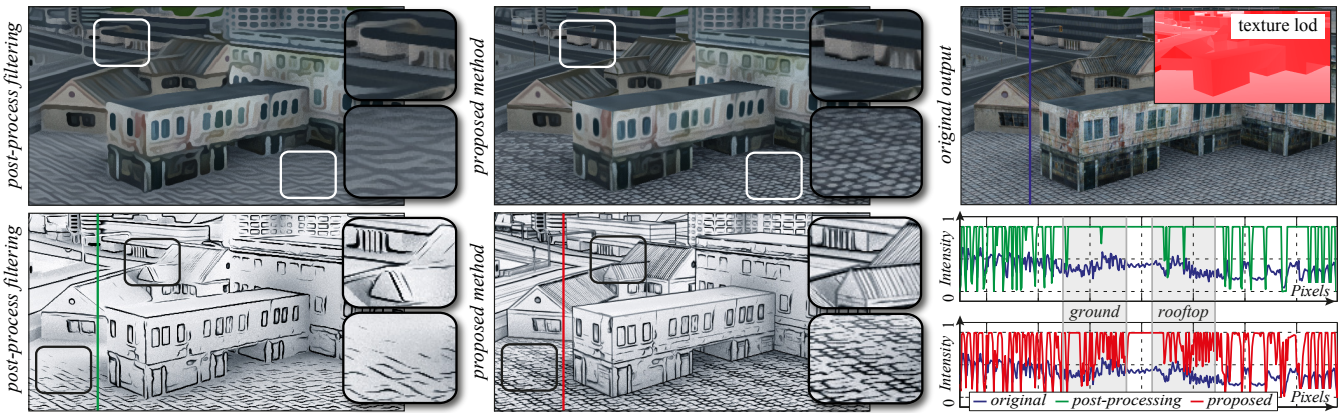

Figure 1: *Exemplary results of a coherence-enhancing filtering (top) and flow-based difference-of-Gaussians filtering (FDoG, bottom) for a textured 3D scene, rendered in real-time using our system. The closeups and scanline plots (FDoG) illustrate the high accuracy for texture gradients when using our methods instead of conventional filtering in a post-process stage on the rendered color image (top right).*

Abstract

Texture mapping is a key technology in computer graphics for visual design of rendered 3D scenes. An effective information transfer of surface properties, encoded by textures, however, depends significantly on how important information is highlighted and cognitively processed by the user in an application context. Edge-preserving image filtering is a promising approach to address this concern while preserving global salient structures. Much research has focused on applying image filters in a post-process stage to foster an artistically stylized rendering, but these approaches are generally not able to preserve depth cues important for 3D visualization (e.g., texture gradient). To this end, filtering that processes texture data coherently with respect to linear perspective and spatial relationships is required. In this work, we present a system that enables to process textured 3D scenes with perspective coherence by arbitrary image filters. We propose *decoupled deferred texturing* with (1) caching strategies to interactively perform image filtering prior to texture mapping, and (2) for each mipmap level separately to enable a progressive level of abstraction. We demonstrate the potentials of our methods on several applications, including illustrative visualization, focus+context visualization, geometric detail removal, and depth of field. Our system supports frame-to-frame coherence, order-independent transparency, multitexturing, and content-based filtering.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms I.4.3 [Computer Graphics]: Image Processing and Computer Vision—Enhancement—Filtering

Keywords: image filtering, visualization, level of abstraction

*http://www.hpi3d.de

http://dx.doi.org/10.1145/2630099.2630101.

1 Introduction

Virtual 3D scenes are often inherently complex with respect to appearance information. Texture mapping is a key technology for their visual design while facilitating a real-time image synthesis. Common texture maps encode diffuse, normal, or displacement information as *surface properties* to enrich shading and lighting effects (e.g., for the building façades in Figure 1). Rendering these properties with all details, however, is not the primary goal for an effective information transfer to a user [Ware 2004]. Instead, non-phototealistic rendering that takes into account a user's background, task, and perspective view, can facilitate how important or prioritized information is highlighted and cognitively processed in an application context [Santella and DeCarlo 2004; Cole et al. 2006; Winnemöller et al. 2007; Redmond and Dingliana 2009].

Highlighting important information while removing extraneous detail of textured 3D scenes is a challenging task, because feature contours and global salient structures must be preserved. A promising approach to address this problem is edge-preserving image filtering. Popular image filters serve as smoothing or enhancing operators, such as the bilateral filter [Tomasi and Manduchi 1998] and difference of Gaussians [Gooch et al. 2004]. Previous works have focused on applying these filters in a post-process stage on the rendered results to foster an artistically stylized rendering [Kyprianidis et al. 2013]. These approaches are able to smooth low-contrast regions and preserve high-contrast edges; however, they are generally not able to preserve depth cues important for perceiving model contents as three-dimensional (e.g., occlusion, texture gradient). For instance, the fine granular patterns on the ground, rooftops, and object boundaries of the 3D scene shown in Figure 1 are not preserved because spatial relationships and linear perspective are not considered. For effective visual information encoding, "good image filtering" needs to preserve these cues to help perceive relative positions, sizes, and shapes more clearly [Pfautz 2000; Goldstein 2010].

This work explores *level-of-abstraction* (LoA) visualization by means of image filtering, i.e., to adapt the spatial granularity at which 3D scene contents should be represented [Semmo et al. 2012]. We propose a system that is based on two key aspects:

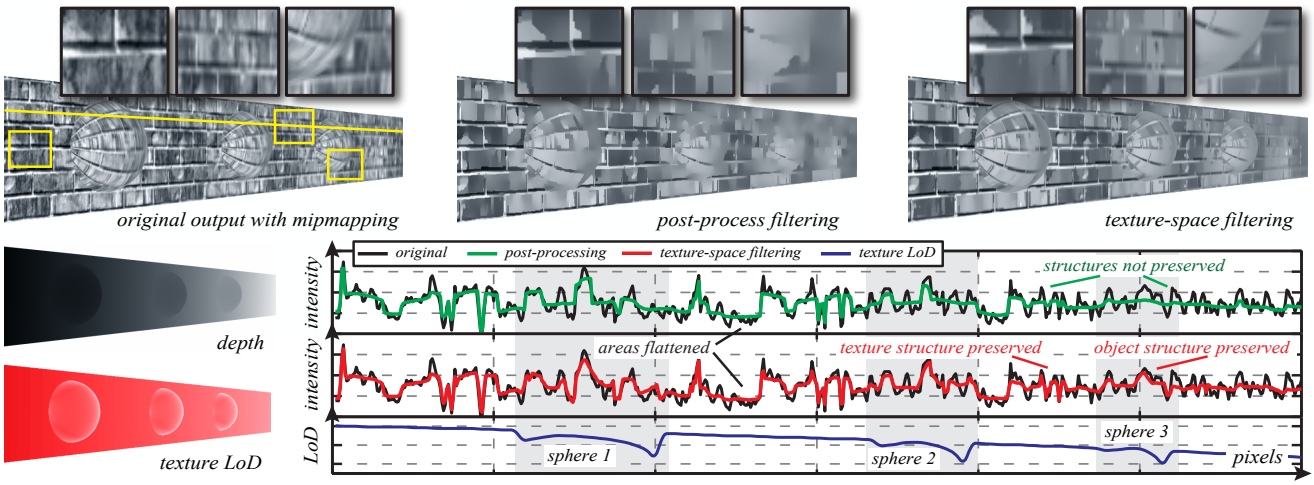

Figure 2: *Exemplary textured 3D scene for which the flow-based bilateral filter ($\rho = 2.0, n_e = 1, n_a = 4, \sigma_d = 4.0, \sigma_r = 5.0$) is applied (1) on the original output and (2) on each level of a mipmap pyramid prior to texture mapping. The scanline plots illustrate the differences in image regions of high texture LoD and occlusion, where the second approach clearly preserves texture gradients and object boundaries.*

(1) Filtering should be perspectively coherent. Linear perspective, occlusion, and texture gradient are effective cues for humans to infer depth [Wanger et al. 1992; Surdick et al. 1994]. Texture mapping considers these cues in perspective projections by foreshortening and scaling. Therefore, our system performs image filtering prior to texture mapping via decoupled deferred texturing. It is a simple yet effective method of preserving these cues without requiring modifications of the original filter algorithms.

(2) Filtering should enable interactive rendering. Viewing situations and regions of interest can be dynamically changed when exploring complex 3D scenes. Prominent examples are 3D geovirtual environments composed of virtual 3D city and landscape models (e.g., Google Earth). To this end, emphasizing important information while abstracting less important information requires selecting the LoA according to user interaction. We contribute per-fragment filtering and caching strategies to enable real-time frame rates for local image filters without requiring to pre-process texture data.

In image-based artistic rendering it is often desired to give the impression that texture features have been applied (painted) on a flat canvas [Bénard et al. 2011]. By contrast, this paper draws upon the potentials of image filtering for visualization purposes considering cognitive principles [Gooch et al. 2010]. We demonstrate benefits of preserving a perspective-coherent scale of texture features for focus+context visualization, illustrative visualization, geometric detail removal, and depth of field. Because filtering is performed in texture space, however, no explicit geometric abstraction is applied, for which specialized rendering techniques are required.

The remainder of this paper is structured as follows. Section 2 highlights the relevance of perspective coherence for image filtering in 3D perspective views. Section 3 reviews related works on image filtering, non-photorealistic rendering, and effective 3D information transfer. Section 4 presents our technical approach to deferred texture filtering, with applications presented and discussed in Section 5. Finally, Section 6 concludes this paper.

2 Why Perspective Coherence Matters

This background section highlights the relevance of *perspective coherence* for LoA visualization. We rendered the 3D scene shown in Figure 2 using a flow-based bilateral filter [Kyprianidis and Döllner 2008]. The input textures were converted to a low-pass filtered mipmap pyramid [Williams 1983] and used for antialiasing. The filtering was performed (1) in a post-process stage on the rendered results, and (2) in texture space prior to perspective projection. Given a texture $T : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ within the RGB color space as input, the first approach is described by an image filtering F performed *after* sampling the convolution of $T * P$ via mipmapping (approximating K), with P being a projector function that maps texture information into the domain of the rendered image $I : \mathbb{R}^2 \rightarrow \mathbb{R}^3$:

$$I(x, y) = F[K \cdot (T * P)](x, y). \quad (1)$$

While this approach performs in a similar manner as pre-filtering a texture in regions of high texture level of detail (LoD), it is not able to preserve structures in regions of low texture LoD because only compressed information serves as input for F (compare the plots of Figure 2). Instead, image filtering requires access to high-frequency texture information prior to perspective projection to preserve depth cues. To this end, mipmap levels are filtered separately prior to texture mapping to approximate the effect of F on $K(T * P)$:

$$I'(x, y) = K_F \cdot (T * P)(x, y). \quad (2)$$

The scanline plot in Figure 2 demonstrates that this approach naturally preserves texture gradients and object boundaries when tri-

Figure 3: *The painting "Paris Street, Rainy Day" (1877) by Gustave Caillebotte (source: Google Art Project). The artist carefully uses texture gradient with linear perspective to enhance depth sensation.*

linearly filtering the respective mipmap levels. While this is not a new approach in 3D computer graphics (e.g., used for pre-filtering shadow maps [Donnelly and Lauritzen 2006]), artists use similar principles in their works to enhance the sensation of depth by adapting the detail and size of texture features with the linear perspective (Figure 3).

We contribute an interactive system that implements Equation 2 to process texture data coherently with respect to linear perspective and occlusions. Further, we contribute optimization techniques to enable real-time performance for local image filters. Using this system with state-of-the-art edge-preserving image filters, we performed a comparison between the filtering approaches (1) and (2). Results are presented in the supplemental materials and video. They demonstrate how each filter's capability to preserve texture gradients and occlusions is improved, while the filters' qualities with respect to image smoothing or enhancing remain unaffected.

3 Related Work

Edge-preserving Image Filtering. Edge-preserving filtering emerged as an essential building block to reduce image details without loss of salient structures. Many filters have been proposed and explored using image abstraction and highlighting [Kyprianidis et al. 2013], and thus are of major interest for our work. Typical approaches operate in the spatial domain, use a kind of anisotropic diffusion [Weickert 1998], and are designed for parallel execution. We implemented a range of local filters in our system to demonstrate how they can be used for real-time filtering of 3D scene contents for progressive LoA visualization. A popular choice is the bilateral filter, which works by weight averaging pixel colors in a local neighborhood based on their distances in space and range [Tomasi and Manduchi 1998]. This method has been used for real-time image-based artistic rendering [Winnemöller et al. 2006] and enhanced by flow-based implementations adapted to the local image structure to provide smooth outputs at curved boundaries [Kyprianidis and Döllner 2008; Kang et al. 2009]. Because the bilateral filter may fail when used in high-contrast images, we also explored the usage of the anisotropic Kuwahara filter [Kyprianidis 2011] to maintain a uniform LoA due to local area flattening, and coherence-enhancing filtering based on directional shock filtering [Kyprianidis and Kang 2011] (see Figure 1). Because our system is designed for generic application without requiring modifications of the original algorithms, future extensions are easy to implement. Additional categories include morphological filtering based on dilation and erosion (e.g., for watercolor rendering [Bousseau et al. 2006]), geodesic filtering using distance transforms [Criminisi et al. 2010; Mould 2012], and filters that allow for detail removal by solving optimization problems in the gradient domain [Bhat et al. 2010].

Edge detection is based on finding zero-crossings or thresholding the gradient magnitude of an image. A popular choice is the difference-of-Gaussians (DoG) filter [Gooch et al. 2004] and its enhanced flow-based variants [Kang et al. 2007; Kyprianidis and Döllner 2008; Winnemöller et al. 2012] to create smooth coherent outputs for line and curve segments. Complemented by an image-space edge detection using geometry information [Nienhaus and Döllner 2003], we used the DoG filter to enhance important edges based on both geometry and texture information. Our method produces accurate filtering results with respect to linear perspective to preserve texture gradients (Figure 1), performs in real-time, and provides frame-to-frame coherence.

Recent filters focused on image decompositions by using optimization methods such as weighted least squares [Farbman et al. 2008], local extrema for edge-preserving smoothing [Subr et al. 2009], locally weighted histograms [Kass and Solomon 2010], domain transforms [Gastal and Oliveira 2011], L_0 gradient minimizations [Xu et al. 2011], and guided filtering [He et al. 2013]. Although most of them do not provide interactive frame rates, we implemented GPU versions to demonstrate how they can serve perspective-coherent filtering of diffuse, normal, and ambient occlusion maps for geometric detail removal and illustrative visualization.

Texture Mapping and Filtering for 3D Scenes. Using texture-based methods for coherent stylization is not a new approach. For an overview on this topic we refer to the survey by Bénard et al. [2011]. Relevant object-space methods reduced perspective distortions and scale variations via mipmapping (i.e., art maps [Klein et al. 2000] and tonal art maps [Praun et al. 2001]) and used infinite zoom mechanisms to maintain a quasi-constant size of texture elements for arbitrary viewing distances [Bénard et al. 2009]. By contrast, our work pursues different objectives: (1) filtering high-detail texture information (e.g., photorealistic imagery) without procedural modelling while (2) considering a perspective-coherent scale of texture features, and (3) maintaining an on-demand, interactive LoA visualization. Previous approaches used bilateral, DoG, and Kuwahara filtering with G-buffer information [Saito and Takahashi 1990] in a post-process stage to express uncertainty [Döllner and Kyprianidis 2010], direct a viewer's gaze to certain image regions [Redmond and Dingliana 2007; Redmond and Dingliana 2009], and convey different emotional and experiential representations [Magdics et al. 2013]. Our system also uses G-buffer information for deferred rendering, but decouples filtering from shading to preserve texture gradients and object boundaries when mapping the filtered results.

Previous works have supported the assertion that perspective is commonly used as an important source of depth information [Wanger et al. 1992; Surdick et al. 1994; Pfautz 2000]. In particular, surface textures are essential components of 3D scenes to judge distances, shapes, and spatial layouts [Gibson 1986; Ware 2004]. A large body of research is dedicated to the way human sensory systems process these pictorial depth cues [Howard and Rogers 2012]; however, there are only a few works that reflect the importance of preserving them when filtering information. A recent study showed that using the results of a DoG filter on diffuse maps significantly improved depth perception in thematic 3D visualization [Engel et al. 2013]. We demonstrate similar approaches to edge enhancement, but with an interactive filtering that can be parameterized at run-time.

To control the detail of 3D shapes via parameterized lighting, major related work is found in cartoon shading [Lake et al. 2000; Barla et al. 2006] that supports view-dependent effects (e.g., LoA, depth of field). By contrast, we propose LoA visualization of texture information (e.g., photorealistic diffuse maps) as an orthogonal approach to these works, and show how their salient structures can be preserved and used for stylized shading and lighting.

Focus+Context Visualization. Highlighting information in foci while maintaining a context for orientation guidance is subject to focus+context visualization. It has the potential to improve the perception of prioritized information [DeCarlo and Santella 2002; Santella and DeCarlo 2004] and direct a viewer's focus to certain image regions [Cole et al. 2006]. A common method is to parameterize image filters according to view metrics (e.g., view distance) [Redmond and Dingliana 2007] or by explicitly defined regions-of-interest [Cole et al. 2006; Cong et al. 2011] to select and seamlessly combine different LoA representations of 3D scene contents [Semmo et al. 2012]. We contribute an interactive system that is able to parameterize arbitrary image filters for view-dependent focus+context visualization. By incorporating texture information coherently with respect to linear perspective, our methods are able to enhance several applications for focus+context visualization, including *stylized focus pull* [Cole et al. 2006] and *semantic depth of field* [Kosara et al. 2001].

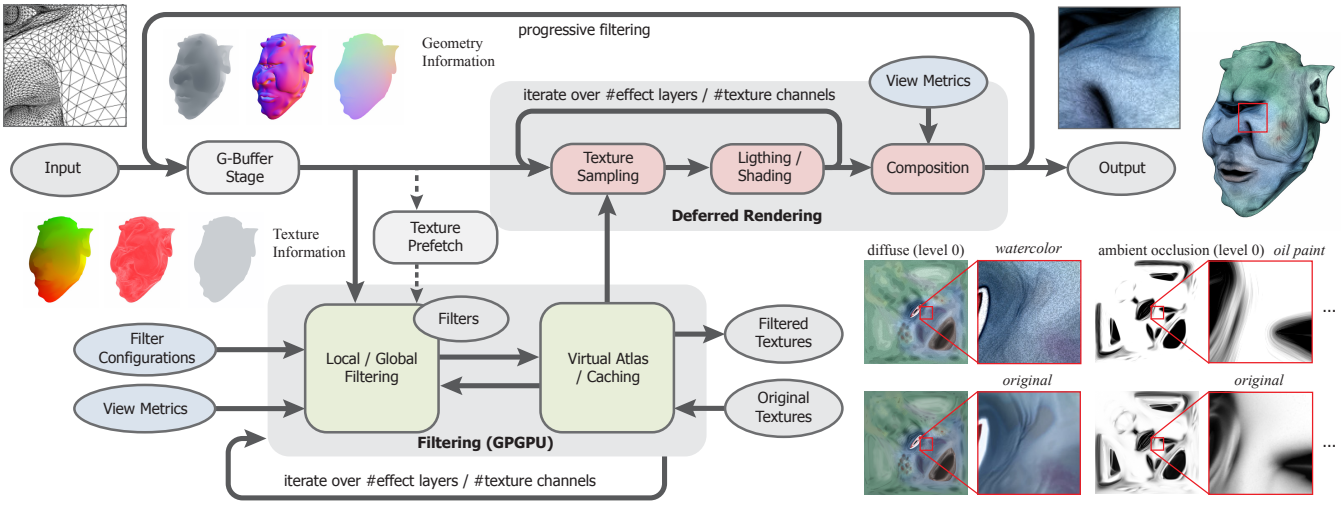

Figure 4: *Schematic overview of our system designed for LoA visualization. The system decouples image filtering from rendering and performs it prior to texture mapping. This way, texture data is filtered coherently with respect to linear perspective, and spatial relationships are preserved. (Jerry the Ogre © Keenan Crane. All rights reserved.)*

4 Method

An overview of our system is shown in Figure 4. It is designed for generic application, can be seamlessly integrated into existing rendering systems, is extensible for arbitrary 2D image filters, and has no particular requirements regarding the consistency of the input geometry (e.g., triangular irregular networks vs. point clouds).

Our system performs filtering prior to texture mapping, for which decoupled deferred texturing is proposed (Section 4.1). Filtering is performed using GPGPU separately on each mipmap level for LoA visualization (Section 4.2). A caching mechanism is used to ensure that image parts are only filtered once for a given configuration. Together with per-fragment and progressive filtering using visibility information, the system enables real-time performance for local image filters. Parameter sets of image filters can be defined per texture channel (e.g., diffuse vs. normal maps) and may be used to define layers with different levels of abstraction. These are blended according to view metrics or regions of interest for focus+context visualization. All geometry and texture buffers are represented as A-buffers to support an order-independent image blending of filtering effects (e.g., blueprint rendering styles).

Optional post-processing may be performed in screen space and combined with the filtering results (Section 4.3), such as for depth-of-field effects. The system can be explicitly parameterized at multiple stages to give users control over the filtering process, as well as implicitly by view metrics (e.g., viewing distance) (Section 4.4).

4.1 Decoupled Deferred Texturing

Our goal is to provide quality, interactive LoA visualization of textured 3D scenes by image filtering that complies with the key aspects named in Section 1. To this end, texture mapping is decoupled from the geometry pass and postponed by deferred texturing to transfer the filtering to texture space (Section 2).

First, in a pre-process stage, textures are assigned unique identifiers $T_{\text{ID}} \in \mathbb{N}$. After computation of the mipmap pyramids, a set of levels $(T_{m_0}, T_{m_1}, ..., T_{m_n})$ is defined per scene texture. The mipmap levels are transferred to GPU texture memory and referenced by their virtual address to enable *bindless texturing* with random read/write access. In addition, these virtual addresses are used for *virtual texturing* to enable a dynamic resolution of texture identifiers during processing. We used a page table with a memory layout shown in Figure 5. For each registered texture, this table references its dimensions, wrap modes required to handle filtering across texture borders (e.g., clamped vs. mirrored), and the virtual address of each mipmap level. The total number of mipmaps is adapted to the maximum texture dimension that can be processed by the GPU.

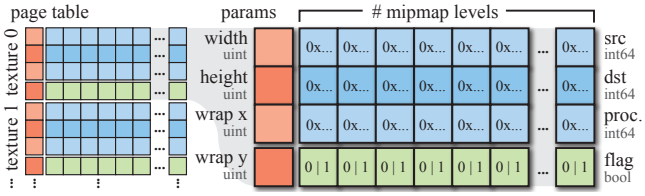

Figure 5: *Layout of the page table used for virtual texturing of original and filtered texture data on the GPU.*

Afterwards, rendering is performed in a series of three stages:

- ***Geometry stage:*** Conventional deferred rendering pipelines render all surface data in the G-buffer [Saito and Takahashi 1990], which includes texture sampling to synthesize diffuse, normal and thematic maps used for visualization. By contrast, our method computes additional buffers related to texture data: identifiers, coordinates, and mipmap LoD (Figure 6).

- ***Filtering Stage:*** Using the G-buffer information, texture parts required for rendering are sampled and filtered. The results are written to separate texture buffers and used together with the original texture data as input for shading.

- ***Shading Stage:*** The results of the geometry and filtering stages are used for *deferred texture mapping*. Optional filtering in screen space may be performed for post-processing effects, such as enhancement of contour lines or depth of field.

The following section describes the filtering stage in more detail.

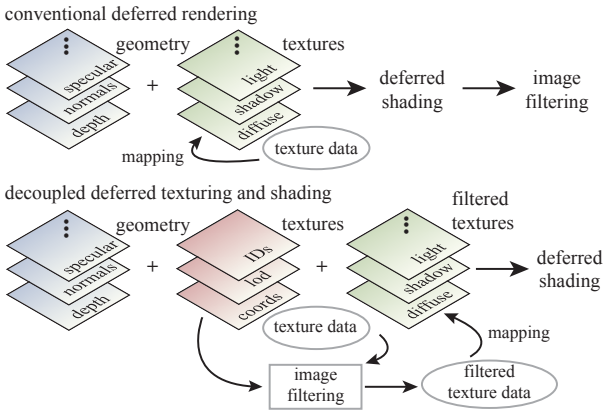

Figure 6: *Conventional deferred rendering performs texture mapping in the geometry stage (top). By contrast, we propose decoupled deferred texturing for perspective-coherent filtering (bottom).*

4.2 Image Filtering

Once the G-buffer has been computed, the following three basic stages are performed for each render pass:

- ***G-buffer mapping:*** The G-buffer is mapped as an arrayed 2D graphics resource, with each $(x, y) \rightarrow (T_{\text{ID}}, T_{\text{coord}}, T_{\text{lod}})$.
- ***Texture prefetch:*** A prefetch of relevant textures required for rendering is performed. For each fragment in the G-buffer, the correspondent mipmap levels $(T_{m_i}, T_{m_{i+1}}) \leftarrow (T_{\text{ID}}, T_{\text{lod}})$ are determined in parallel using atomic operations and stored in a global structure. To avoid redundant filtering, a processing flag is set for each mipmap level in the page table (Figure 5).
- ***Image filtering:*** Each unique T_{m_i} is filtered with the correspondent image filter and configuration. Additional borders are introduced according to the wrap modes $(T_{\text{wrapx}}, T_{\text{wrapy}})$ to avoid visible seams when texturing. The results are written to the destination buffer, and the processing flag is set.

Image filtering is performed separately on each mipmap level. Afterwards, the results are blended in the shading stage by trilinear filtering. Compared to filtering the highest mipmap level first and then downsampling the results, the approach enables a progressive LoA visualization (Figure 7). In this manner, visual clutter can be reduced significantly in areas of high perspective compression

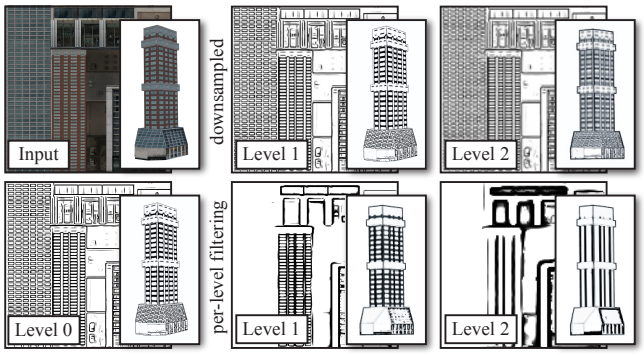

Figure 7: *Example of LoA visualization by means of trilinear and DoG filtering: (top) downsampled filtered input, (bottom) input downsampled and then filtered per mipmap level.*

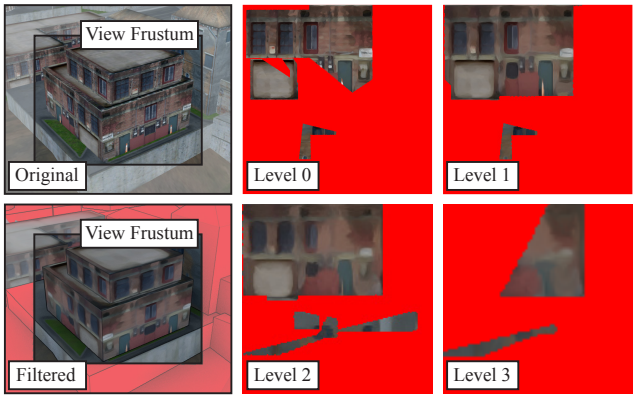

Figure 8: *Local filtering is performed only for visible texture parts as performance optimization. Unused texture parts are uninitialized for the given 3D view (red). Notice how geometry outside the view frustum (bottom left) is only textured if parts are reused.*

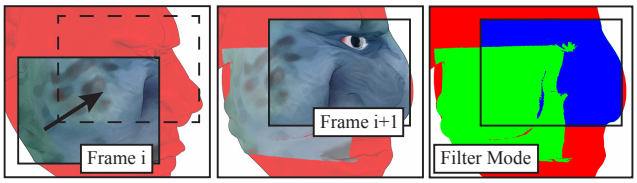

Figure 9: *Filtered texture parts are cached (green), reused for subsequent frames, and combined with new filtered parts (blue).*

without requiring the adaption of any filter parameters. Special care is required when mipmapped texture atlases are used for filtering to avoid bleeding across image tiles. Typically, this problem is addressed by introducing additional spaces between packed tiles, but it does not ultimately solve the problem with wide filter kernels or global operations. Virtual and bindless texturing alleviates this problem, but requires using non-packed textures as input. An alternative approach may use tile masks pre-defined per mipmap level; however, this method increases the memory footprint.

Dependent on the computational complexity of an image filter, the filtering stage could take significant processing time and stall the rendering stage. To this end, we introduce two enhancements:

1. Per-fragment Filtering and Caching: Because in most cases only small parts of textures are used for rendering and a lower texture LoD is selected in background regions of 3D perspective views, visibility-driven filtering allows for significant performance improvements. Therefore, per-fragment filtering is introduced by using the G-buffer to naturally process only the information required for rendering (Figure 8). The *texture prefetch* is coupled with the filtering process, for which the texels required for trilinear filtering are determined per fragment output by the rasterization (maximum eight), and correspondent filtered values are computed in parallel. We used a top-down filtering approach, i.e., information required for filtering is recursively resolved on-demand during processing. In addition, a caching mechanism is used so that texture parts are only filtered once for a given configuration and reused for subsequent render frames (Figure 9). The page table (Figure 5) references additional image masks $(T_{p_0}, T_{p_1}, ..., T_{p_n})$ per mipmap level that indicate whether a pixel has already been processed.

2. Progressive Filtering: Some local and many global filters that solve an optimization problem do not perform at interactive frame rates. Filtering the highest mipmap levels first and using them

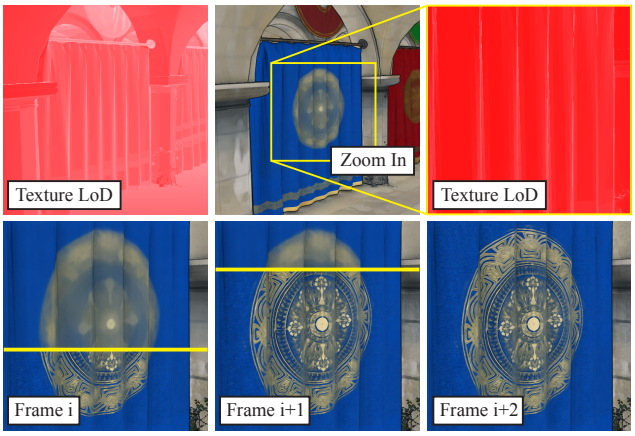

Figure 10: *Example of a textured 3D scene that is progressively filtered using a computational filtering budget per render pass to maintain interactive frame rates. (Sponza Atrium scene © Marko Dabrovic and Frank Meinl from Crytek. All rights reserved.)*

as fallback is used for progressive filtering. Our system uses a computational budget that is applied per rendered frame and can be interactively configured. Detail information is then progressively blended in subsequent render frames (Figure 10). This procedure also prepares an easy deployment on multi-GPU systems to decouple computationally expensive global filters from rendering, but remains subject to future work (Section 6).

The filtering kernel for both optimization techniques is summarized in the supplemental materials. Using these techniques, the computational cost of processing each texel over post-processing can be amortized: on the one hand, because pre-filtered texture information serves as input via mipmapping, and on the other hand because the caching strategies avoid reprocessing. Our performance evaluation in Section 5.2 demonstrates that this enables local image filters to process textured 3D scenes at real-time frame rates.

To enable filtering of multi-textured 3D scenes, the G-buffer is enhanced as follows (Figure 11). First, each fragment from the rasterization is assigned two sets of texture coordinates together with multiple texture identifiers defined per texture channel. Filter configurations can then be defined per texture channel to enable a content-based filtering that is adapted to 3D model semantics. Second, fragments are buffered in depth and a sorting is performed in a post-process stage to enable order-independent transparency effects. Third, different filters and configurations can be defined per input texture, for which the virtual texture table is extended by correspondent destination buffers. The filter results are then blended according to view metrics (e.g., viewing distance) or pre-defined regions of interest using image masks for focus+context visualization (e.g., stylized foci [Cole et al. 2006]). Examples that use these enhancements are presented in Section 5.

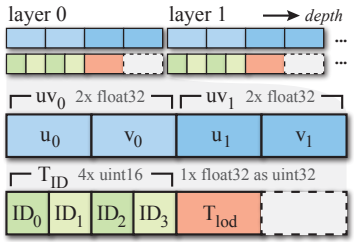

Figure 11: *Enhanced G-buffer.*

The number of texture coordinates and channels for the G-buffer is not ultimately defined, but should be limited to bound memory consumption. Because all processes are designed for generic application, the G-buffer can be extended easily by additional attributes.

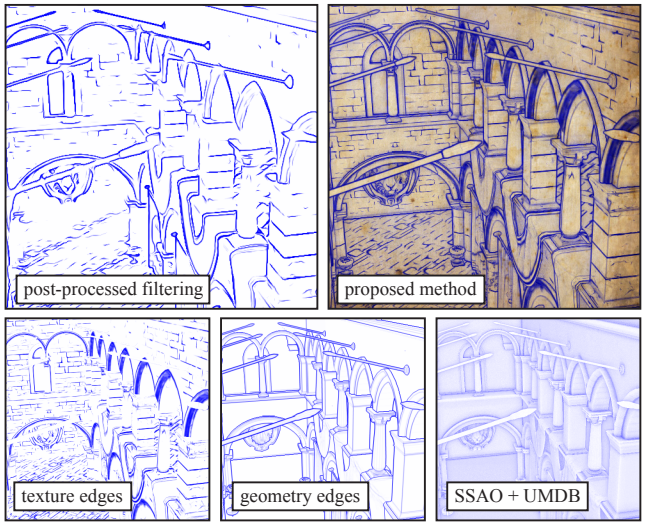

Figure 12: *Example of post-processing effects used in our system. Edge detection is decoupled into DoG filtering of texture and geometry information. The results are combined with screen-space ambient occlusion (SSAO), unsharp masking the depth buffer (UMDB), and a background texture to compose the final image (top right).*

4.3 Deferred Shading and Composition

Once the G-buffer is computed and filtering is performed, the results are used as input for texture sampling, which is independently performed from shading and lighting. Optionally, screen-space ambient occlusion and unsharp masking the depth buffer [Luft et al. 2006] is performed using normal and depth information of the G-buffer to improve depth perception further. Results that include DoG filtering are combined with an image-based edge enhancement technique [Nienhaus and Döllner 2003] to include silhouette, border, and crease edges based on depth, normal, and object identifier information [Saito and Takahashi 1990]. In contrast to traditional DoG filtering in screen space, our method is able to decouple edges based on texture and geometry information. Figure 12 and the accompanying video demonstrate that this approach produces much more accurate filtering results with respect to linear perspective while providing frame-to-frame coherence and real-time frame rates.

4.4 User Interaction

Our system gives users full control over the different parameters defined per image filter, including kernel size, quantization intensity, sensitivity of edge detection, adaptive smoothing in a post-process stage, and weights of flow fields. In particular, users are able to define parameters to control the composition of the filtering results:

- texture channel semantics with filters and configurations defined per channel for content-based LoA visualization;
- view metrics and region masks together with transition parameters used for blending layered filtering effects;
- colors for enhanced geometry edges, screen-space ambient occlusion and unsharp masking effects;
- multi-sampling and order-independent transparency parameters used for composition.

The capability to cache filtering results also enables users to export processed 3D scenes for future post-processing, or for using our system as a previsualization tool for print or digital production.

Figure 13: *Example of a stylized focus pull where the distance-based focal plane moves gradually from the foreground to the background. A watercolor and DoG filter were used to draw emphasis to regions of interest and preserve depth cues in context regions (e.g., building windows).*

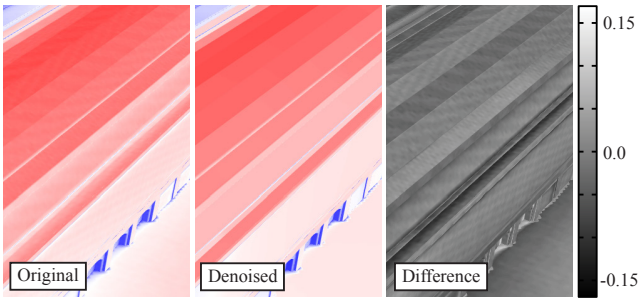

Figure 14: *L_0 gradient minimization ($\lambda = 0.02, \kappa = 2$) used with our system for edge-preserving denoising of texture-encoded solar potential data. The scene shows parts of a building roof.*

5 Results and Discussion

We have implemented our system using C++ and OpenGL/GLSL. OpenSceneGraph is used as the rendering engine to handle 3D data sets. We have implemented the filters used in this work on the GPU with CUDA. Filtering was performed in RGB color space or CIE-Lab. Optimizations were not heavily applied only where appropriate. We used texture and surface objects introduced in CUDA 5 for virtual texturing, along with the *OpenGL Interop* functionality for random read/write access of textures. Similar results should be achievable using OpenCL or compute shaders. The G-buffer is packed to RGBA or RGB texture channels. We used the OpenGL 4.4 extension `GL_ARB_bindless_texture` for bindless texturing and a stencil-routed A-buffer for order-independent image blending [Myers and Bavoil 2007]. In the following, we demonstrate the versatility of our methods on several applications.

5.1 Applications

The individual applications of the respective image filters – e.g., HDR tone mapping, detail exaggeration, edge adjustment, or colorization – can be maintained. The only difference is that they are transferred to the texture domain. Figure 14 shows a result of our system using L_0 gradient minimization [Xu et al. 2011] for perspective-coherent denoising of thematic data and contrast enhancement. Here, solar potential was computed by a radiation summed up over a whole year and encoded in texture maps.

Highlighting regions of interest while removing detail in context regions is a major goal in information visualization for directing a viewer's gaze to important or prioritized information [DeCarlo and Santella 2002; Santella and DeCarlo 2004]. Figure 13 presents a visualization using our methods to implement a *stylized focus pull* [Cole et al. 2006] based on the view distance. Emphasis is drawn to the respective image regions using the FDoG filter coupled with

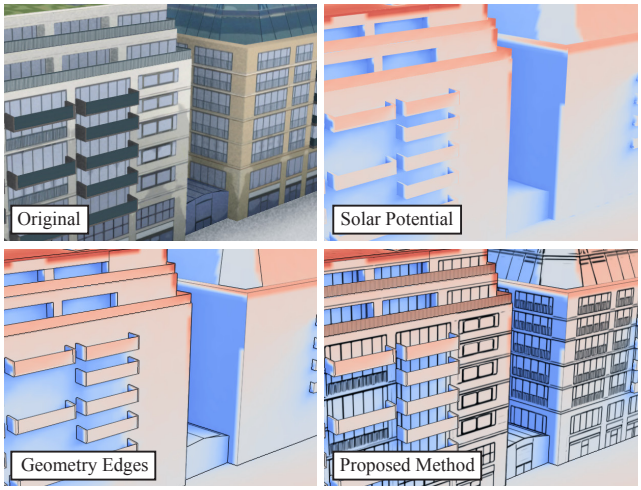

Figure 15: *A result of our system in which geometry edges and DoG filtered diffuse textures are combined to improve the visualization of color-encoded thematic data (solar potential) significantly.*

image-based enhancement of geometry edges for context regions. Our methods are able to preserve the overall structure of objects in the background and objects in the context regions (e.g., windows on the building façades to emphasize spatial relationships). In addition, they are generic with respect to filter combinations and extensible for further view metrics (e.g., region masks or view angles), because the image composition is performed in the deferred shading stage, and multiple layer effects may be defined. We used the multi-texturing support to enhance depth cues important for visualizing color-encoded thematic data in 3D scenes. Figure 15 presents a result in which the FDoG filter was used to detect edges in diffuse maps and blend the output with thematic information. Compared to an edge detection that is only based on geometry information, our approach also enhances structural information that is not explicitly modeled as geometry, but is captured by aerial or terrestrial imaging (e.g., building façades). This way, the correlation between thematic data and surface structures is much more plausible.

Depth of field is known to direct the pre-attentive cognition of prioritized information. We implemented a variant of the *semantic depth-of-field* effect [Kosara et al. 2001], in which scene objects are selectively highlighted when using image masks to direct a Gaussian smoothing in screen space. First, we used our system to filter diffuse maps in texture space, i.e., to control the LoA of textured surfaces. Afterwards, regular Gaussian smoothing is performed in screen space to blur geometry edges. As demonstrated in Figure 16, using this "dual" filtering approach enabled a clearer visualization of structures induced by geometry edges. By contrast, the regular screen-space approach requires wider filter kernels to achieve a sim-

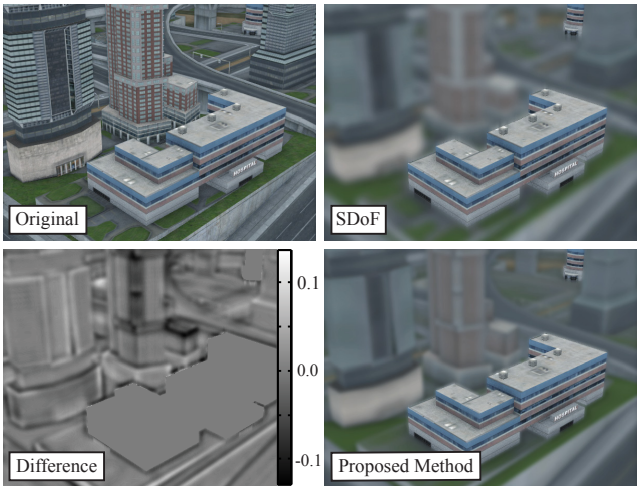

Figure 16: *Gaussian smoothing for semantic depth of field (SDoF). We used our system for perspective-coherent Gaussian smoothing of diffuse maps ($\sigma_t = 7.5$) prior to smoothing in screen space ($\sigma_s = 3.5$). Compared to the traditional approach, which only filters in screen space ($\sigma_s = 7.5$), our method preserves scene structures better at a similar LoA in context regions.*

ilar LoA effect, but at the cost of a considerably high degradation of scene structures.

Bump or displacement mapping is an essential process for enriching shading and lighting effects by geometric detail. We used our methods to perform edge-preserving smoothing of normal maps to coarse bump mapping with respect to linear perspective. Figure 17 shows the results of a domain transform [Gastal and Oliveira 2011] applied to normal maps, where mipmapping enables a smooth transition between the different levels of structural abstraction.

We also experimented with filtering normal and ambient occlusion maps to achieve stylized shading and lighting effects. The rich parameterization options of our system give artists creative control over this process. Figure 18 shows how an oil paint filter that is based on a smoothed structure tensor was used to apply Phong shading and ambient occlusion with a sketchy style. Similar directions were proposed in [DeCoro et al. 2007] for stylized shadows and toon shading for general LoA [Barla et al. 2006]; however, without the capability for flow-based image abstraction. By contrast, our methods are based on stylized variants of texture maps that include salient structures, which are blended by conventional shading. Hence, they are especially useful to interactively stylize photorealistic texture maps (e.g., captured by terrestrial photography).

Finally, the capability to layer filtering in depth was observed by a novel blueprint rendering approach. In contrast to image-based techniques that solely operate on G-buffer information [Nienhaus and Döllner 2004], we additionally utilized a DoG filter for perspective-coherent abstraction of diffuse maps that preserves texture gradients (Figure 19). The filtering results were colorized and blended by order-independent transparency. Similar results can also be achieved for volume visualization (i.e., texture slices).

The accompanying video demonstrates frame-to-frame coherence for all of our results. The coherence primarily comes from processing the respective mipmap levels according to perspective projection, *prior* to trilinearly filtering the results on the GPU for smooth interpolation (Equation 2). A similar approach has also been used for the art maps concept [Klein et al. 2000]. More results for state-of-the art image filters are presented in the supplemental materials.

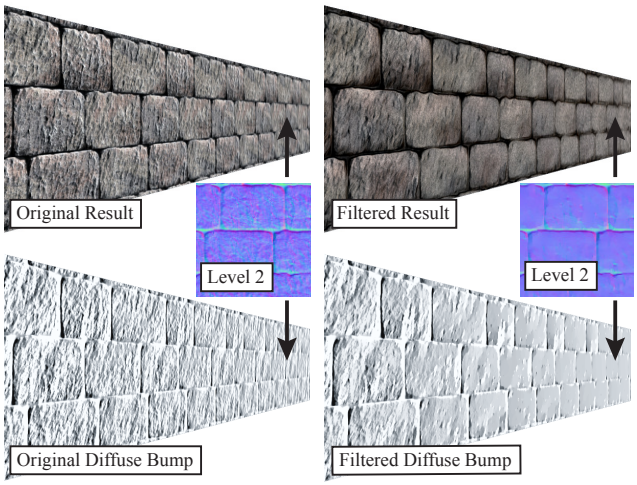

Figure 17: *Domain transform (recursive filter) applied to normal maps ($\sigma_s = 30.0, \sigma_r = 0.25, N = 3$) for a smoothed bump mapping, combined with a watercolor filter for the diffuse maps.*

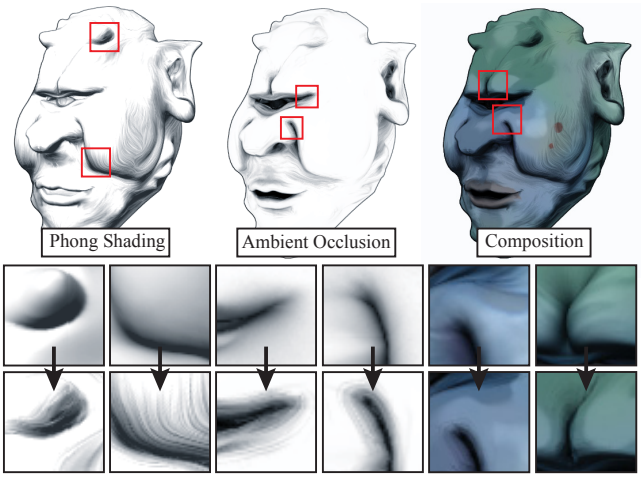

Figure 18: *Stylization of Phong shading, ambient occlusion, and diffuse maps. An oil paint filter was used for the normal and ambient occlusion maps prior to shading, combined with the abstraction filter presented in [Winnemöller et al. 2006] for diffuse maps.*

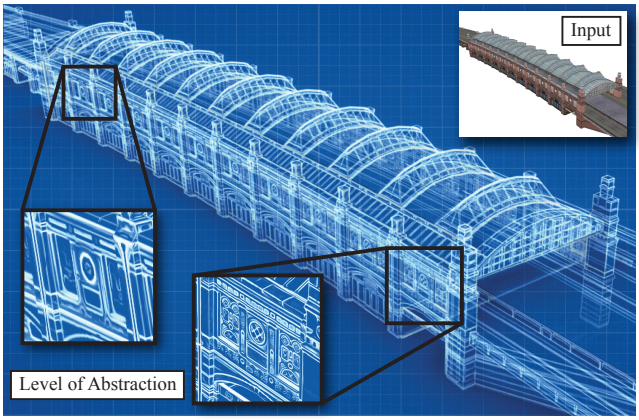

Figure 19: *A blueprint rendering based on geometry edges and FDoG filtered diffuse maps blended per depth layer. The close-ups (first layer) illustrate the LoA capabilities of our approach.*

5.2 Performance Evaluation

The performance tests of our system were conducted on an Intel® Xeon™ 4× 3.06 GHz with 6 GByte RAM and NVidia® GTX 760 GPU with 4 GByte VRAM. Because our system was designed for general-purpose application, its performance greatly depends on which image filters are used, which we identified as bottleneck, and how many effect layers are defined. We used the virtual 3D city scene depicted in Figures 13/16 and the multi-scale anisotropic Kuwahara filter ($N = 4$) [Kyprianidis 2011] as a test setup. The scene is composed of 540 unique 3D objects with 15 texture atlases (each 1024×1024 pixels). We defined a fly-through sequence that lasts 15 seconds with filter caches cleared prior to each iteration.

The results provided in Table 1 show that our system's performance scales with the screen resolution, reaching interactive frame rates in HD resolutions and real-time performance when using our caching methodologies. Notice how per-fragment filtering without caching is almost on par with conventional post-process filtering in HD screen resolutions, thus it could also be used for dynamic textures. The timings include all filtering and rendering cycles with potential memory transfers. The memory consumption is proportional to the screen resolution with respect to the G-buffer, and proportional to the number of filter layers with respect to cached textures targets. We believe that the memory consumption can be significantly decreased when using sparse textures (OpenGL 4.3).

5.3 Limitations

In contrast to image-based artistic stylization, our methods are not able to inherently control the LoA of the scene geometry. However, this also enables a more controlled geometric abstraction by specialized visualization techniques. Alternatively, our methods may be combined with filtering in a post-process stage to selectively filter across object boundaries. In addition, UV mapped textures are required as input, otherwise a spatial filtering is not practical.

There is also room for improvement of our system's performance. As a current limitation, interactive filtering cannot be maintained when solving sparse linear systems (e.g., for image decomposition) because local image filtering cannot be performed. This also applies to iterative approaches that are resistant to parallelization. Outsourcing computationally expensive filters to multi-GPU systems could alleviate this problem, which is supported by progressive and decoupled filtering, but it remains subject to future work.

6 Conclusions and Future Work

In this paper, we have presented image filtering for interactive level-of-abstraction visualization of textured 3D scenes. Our decoupled deferred texturing concept enables to process texture maps coherently with respect to linear perspective by arbitrary image filters to preserves depth cues. We fashioned a caching concept and optimization schemes by per-fragment and progressive filtering that enable interactive or real-time frame rates. In addition, we proposed a system that is extensible by custom image filters and provides interactive control over the filtering process. Several results demonstrate its manifold applicability to the fields of visualization.

We see multiple directions for future work. We plan to adapt the stylization functions to visual saliency and semantics, providing more context-sensitive visualization. In addition, sketch-based interfaces could be used to provide an interactive authoring tool for aesthetic renditions of 3D scenes. Finally, we plan to conduct a comprehensive user study to confirm that our methods are able to significantly improve visualization tasks.

Table 1: *Performance evaluation of our system using the multi-scale Anisotropic Kuwahara filter [Kyprianidis 2011] (average and minimum frames-per-second). Setups: (1) filtering entire mipmap levels with caching enabled, (2) per-fragment filtering with caching disabled and (3) enabled, (4) filtering in a post-process stage.*

Screen Res.	setup 1	setup 2	setup 3	setup 4
800×600	75.6 (12)	12.9 (9)	87.9 (44)	17.2 (16)
1280×720	51.9 (12)	11.5 (10)	67.3 (26)	14.7 (14)
1920×1080	31.5 (11)	11.3 (10)	37.3 (14)	11.8 (11)

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. We would also like to thank Marko Dabrovic and Frank Meinl from Crytek for the Sponza scene used in Figure 10/12 and the supplemental materials, and Keenan Crane for Jerry the Ogre used in Figure 4/9/18 and the supplemental materials. This work was funded by the Federal Ministry of Education and Research (BMBF), Germany, within the InnoProfile Transfer research group "4D*n*D-Vis" (`www.4dndvis.de`) and the Potsdam Research Cluster for Georisk Analysis, Environmental Change, and Sustainability (PROGRESS).

References

BARLA, P., THOLLOT, J., AND MARKOSIAN, L. 2006. X-toon: An Extended Toon Shader. In *Proc. NPAR*, 127–132.

BÉNARD, P., BOUSSEAU, A., AND THOLLOT, J. 2009. Dynamic Solid Textures for Real-time Coherent Stylization. In *Proc. ACM I3D*, 121–127.

BÉNARD, P., BOUSSEAU, A., AND THOLLOT, J. 2011. State-of-the-Art Report on Temporal Coherence for Stylized Animations. *Computer Graphics Forum 30*, 8, 2367–2386.

BHAT, P., ZITNICK, C. L., COHEN, M., AND CURLESS, B. 2010. GradientShop: A Gradient-domain Optimization Framework for Image and Video Filtering. *ACM Trans. Graph. 29*, 2, 10:1–10:14.

BOUSSEAU, A., KAPLAN, M., THOLLOT, J., AND SILLION, F. X. 2006. Interactive Watercolor Rendering with Temporal Coherence and Abstraction. In *Proc. NPAR*, 141–149.

COLE, F., DECARLO, D., FINKELSTEIN, A., KIN, K., MORLEY, K., AND SANTELLA, A. 2006. Directing Gaze in 3D Models with Stylized Focus. *Proc. EGSR*, 377–387.

CONG, L., TONG, R., AND DONG, J. 2011. Selective Image Abstraction. *Vis. Comput. 27*, 3, 187–198.

CRIMINISI, A., SHARP, T., ROTHER, C., AND PÉREZ, P. 2010. Geodesic Image and Video Editing. *ACM Trans. Graph. 29*, 5, 134:1–134:15.

DECARLO, D., AND SANTELLA, A. 2002. Stylization and Abstraction of Photographs. *ACM Trans. Graph. 21*, 3, 769–776.

DECORO, C., COLE, F., FINKELSTEIN, A., AND RUSINKIEWICZ, S. 2007. Stylized Shadows. In *Proc. NPAR*, 77–83.

DÖLLNER, J., AND KYPRIANIDIS, J. E. 2010. Approaches to Image Abstraction for Photorealistic Depictions of Virtual 3D Models. In *Cartography in Central and Eastern Europe*. Springer, 263–277.

DONNELLY, W., AND LAURITZEN, A. 2006. Variance Shadow Maps. In *Proc. ACM I3D*, 161–165.

ENGEL, J., SEMMO, A., TRAPP, M., AND DÖLLNER, J. 2013. Evaluating the Perceptual Impact of Rendering Techniques on

Thematic Color Mappings in 3D Virtual Environments. In *Proc. Vision, Modeling & Visualization*, 25–32.

FARBMAN, Z., FATTAL, R., LISCHINSKI, D., AND SZELISKI, R. 2008. Edge-Preserving Decompositions for Multi-Scale Tone and Detail Manipulation. *ACM Trans. Graph. 27*, 3, 67:1–67:10.

GASTAL, E. S. L., AND OLIVEIRA, M. M. 2011. Domain Transform for Edge-Aware Image and Video Processing. *ACM Trans. Graph. 30*, 4, 69:1–69:12.

GIBSON, J. J. 1986. *The ecological approach to visual perception*. Routledge.

GOLDSTEIN, E. B. 2010. *Sensation and perception*. Wadsworth Publishing Company.

GOOCH, B., REINHARD, E., AND GOOCH, A. 2004. Human Facial Illustrations: Creation and Psychophysical Evaluation. *ACM Trans. Graph. 23*, 1, 27–44.

GOOCH, A. A., LONG, J., JI, L., ESTEY, A., AND GOOCH, B. S. 2010. Viewing Progress in Non-photorealistic Rendering through Heinlein's Lens. In *Proc. NPAR*, 165–171.

HE, K., SUN, J., AND TANG, X. 2013. Guided Image Filtering. *IEEE Trans Pattern Anal Mach Intell. 35*, 6, 1397–1409.

HOWARD, I. P., AND ROGERS, B. J. 2012. *Perceiving in Depth, Volume 3: Other Mechanisms of Depth Perception*. No. 29. Oxford University Press.

KANG, H., LEE, S., AND CHUI, C. K. 2007. Coherent Line Drawing. In *Proc. NPAR*, 43–50.

KANG, H., LEE, S., AND CHUI, C. K. 2009. Flow-Based Image Abstraction. *IEEE Trans. Vis. Comput. Graphics 15*, 1, 62–76.

KASS, M., AND SOLOMON, J. 2010. Smoothed Local Histogram Filters. *ACM Trans. Graph. 29*, 4, 100:1–100:10.

KLEIN, A. W., LI, W., KAZHDAN, M. M., CORRÊA, W. T., FINKELSTEIN, A., AND FUNKHOUSER, T. A. 2000. Non-photorealistic Virtual Environments. In *Proc. ACM SIGGRAPH*, 527–534.

KOSARA, R., MIKSCH, S., AND HAUSER, H. 2001. Semantic Depth of Field. In *Proc. IEEE InfoVis*, 97–104.

KYPRIANIDIS, J. E., AND DÖLLNER, J. 2008. Image Abstraction by Structure Adaptive Filtering. In *Proc. EG UK TPCG*, 51–58.

KYPRIANIDIS, J. E., AND KANG, H. 2011. Image and Video Abstraction by Coherence-Enhancing Filtering. *Comput. Graph. Forum 30*, 2, 593–602.

KYPRIANIDIS, J. E., COLLOMOSSE, J., WANG, T., AND ISENBERG, T. 2013. State of the Art: A Taxonomy of Artistic Stylization Techniques for Images and Video. *IEEE Trans. Vis. Comput. Graphics 19*, 5, 866–885.

KYPRIANIDIS, J. E. 2011. Image and Video Abstraction by Multi-scale Anisotropic Kuwahara Filtering. In *Proc. NPAR*, 55–64.

LAKE, A., MARSHALL, C., HARRIS, M., AND BLACKSTEIN, M. 2000. Stylized Rendering Techniques for Scalable Real-time 3D Animation. In *Proc. NPAR*, 13–20.

LUFT, T., COLDITZ, C., AND DEUSSEN, O. 2006. Image Enhancement by Unsharp Masking the Depth Buffer. *ACM Trans. Graph. 25*, 3, 1206–1213.

MAGDICS, M., SAUVAGET, C., GARCÍA, R. J., AND SBERT, M. 2013. Post-processing NPR Effects for Video Games. In *Proc. ACM VRCAI*, 147–156.

MOULD, D. 2012. Texture-preserving Abstraction. In *Proc. NPAR*, 75–82.

MYERS, K., AND BAVOIL, L. 2007. Stencil routed A-Buffer. In *ACM SIGGRAPH Sketches*, 21.

NIENHAUS, M., AND DÖLLNER, J. 2003. Edge-enhancement - An algorithm for real-time non-photorealistic rendering. *Journal of WSCG 11*, 2, 346–353.

NIENHAUS, M., AND DÖLLNER, J. 2004. Blueprints: Illustrating Architecture and Technical Parts Using Hardware-accelerated Non-photorealistic Rendering. In *Proc. Graphics Interface*, 49–56.

PFAUTZ, J. D. 2000. *Depth perception in computer graphics*. PhD thesis, University of Cambridge.

PRAUN, E., HOPPE, H., WEBB, M., AND FINKELSTEIN, A. 2001. Real-time hatching. In *Proc. ACM SIGGRAPH*, 581–586.

REDMOND, N., AND DINGLIANA, J. 2007. Adaptive Abstraction of 3D Scenes in Real-Time. In *Eurographics Short Papers*, 77–80.

REDMOND, N., AND DINGLIANA, J. 2009. Investigating the Effect of Real-time Stylisation Techniques on User Task Performance. In *Proc. APGV*, 121–124.

SAITO, T., AND TAKAHASHI, T. 1990. Comprehensible Rendering of 3-D Shapes. In *Proc. ACM SIGGRAPH*, 197–206.

SANTELLA, A., AND DECARLO, D. 2004. Visual Interest and NPR: an Evaluation and Manifesto. In *Proc. NPAR*, 71–150.

SEMMO, A., TRAPP, M., KYPRIANIDIS, J. E., AND DÖLLNER, J. 2012. Interactive Visualization of Generalized Virtual 3D City Models using Level-of-Abstraction Transitions. *Comput. Graph. Forum 31*, 3, 885–894.

SUBR, K., SOLER, C., AND DURAND, F. 2009. Edge-preserving Multiscale Image Decomposition based on Local Extrema. *ACM Trans. Graph. 28*, 5, 147:1–147:9.

SURDICK, R. T., DAVIS, E. T., KING, R. A., CORSO, G. M., SHAPIRO, A., HODGES, L., AND ELLIOT, K. 1994. Relevant cues for the visual perception of depth: is where you see it where it is? In *Proc. Hum. Fact. Ergon. Soc. Annu. Meet.*, vol. 38, 1305–1309.

TOMASI, C., AND MANDUCHI, R. 1998. Bilateral Filtering for Gray and Color Images. In *Proc. ICCV*, 839–846.

WANGER, L., FERWERDA, J., AND GREENBERG, D. 1992. Perceiving spatial relationships in computer-generated images. *IEEE Computer Graphics and Applications 12*, 3, 44–58.

WARE, C. 2004. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers Inc., San Francisco.

WEICKERT, J. 1998. *Anisotropic diffusion in image processing*, vol. 1. Teubner Stuttgart.

WILLIAMS, L. 1983. Pyramidal parametrics. In *Proc. ACM SIGGRAPH*, vol. 17, 1–11.

WINNEMÖLLER, H., OLSEN, S. C., AND GOOCH, B. 2006. Real-Time Video Abstraction. *ACM Trans. Graph. 25*, 3, 1221–1226.

WINNEMÖLLER, H., FENG, D., GOOCH, B., AND SUZUKI, S. 2007. Using NPR to Evaluate Perceptual Shape Cues in Dynamic Environments. In *Proc. NPAR*, 85–92.

WINNEMÖLLER, H., KYPRIANIDIS, J. E., AND OLSEN, S. C. 2012. XDoG: an extended difference-of-Gaussians compendium including advanced image stylization. *Computers & Graphics 36*, 6, 740–753.

XU, L., LU, C., XU, Y., AND JIA, J. 2011. Image Smoothing via L_0 Gradient Minimization. *ACM Trans. Graph. 30*, 6, 174:1–174:12.