Instant Texture Transmission using Bandwidth-optimized Progressive Interlacing Images

Michael Englert, Yvonne Jung, Marcel Klomann, Jonas Etzold, Paul Grimm

Fulda University of Applied Sciences, Germany (Email for all authors: firstname.lastname@cs.hs-fulda.de)

Abstract

We present an adaptive bandwidth-optimized approach for progressive image transmission to allow instant textured rendering in web-based 3D applications. Applications utilizing a lot of image data require an adaptive technology to build responsive user interfaces. This applies especially for the use in mobile networks. While several approaches provide a progressive geometry streaming, they do not focus on a fast and simple texture streaming for 3D web applications. However, standard 2D image transmission technologies are usually inappropriate within a 3D context. In 2D, image size as well as resolution are often set during the authoring phase, whereas in 3D applications size and displayed resolution of textured 3D objects depend on the virtual camera. An adaptive texture transmission mechanism has to consider this as part of its control function. Our approach thus combines progressive texture streaming with an adaptive number of refinement levels without any pixel retransmission. It enables a fast user feedback and reduces the transmitted data to a minimum without loss of visual quality. Moreover, our GPUII approach (GPU-based Image Interlacing) allows an easy integration into existing applications while also reducing the CPU load.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

Keywords: Progressive Image Transmission, WebGL, HTML5

1 GPU-based Image Interlacing

Our approach uses a simple encoding scheme, based on Adam7 interlacing [Costello 2003], and a fast decoding algorithm that benefits from hardware acceleration by a GPU – even with WebGL’s rather limited instruction set. During encoding we produce an image set consisting of a variable number of subimages that resembles the given image. Therefore, we can generate first previews of nearly identical size in bytes, independent of the source image size. This allows us to get a more flexible handling of source images with arbitrary resolutions. We further propose a texture download method that takes properties of the 3D scene, like camera position or screen space size, into account, to decide which preview has to be loaded next and – in case multiple textures are used – how to sort them for importance. Moreover, image interlacing schemes like Adam7 do not yet foresee pausing the download and carrying on if more data is required, though in our test cases this was obviously very helpful to efficiently reduce the transferred data amount.

The transmitted image data is decoded according to figure 2, where we used a GPU-based ping-pong rendering technique to produce a dynamically combined texture representation. Thus, our proposed method can be applied to stream textures along with their corresponding meshes for consistent rendering. In addition to the PNG format, all common image formats that are supported by browsers can be utilized as data transport containers for our GPUII textures.

To ease usage we have integrated the proposed technique as special texture node in X3DOM. For future work, we would like to combine our method with a hierarchical approach to stream large regularly organized meshes as described in [Herzig et al. 2013].

Acknowledgements

This work was carried out within the BMBF-funded project Flin.

References