Tile-Based Texture Mapping on Graphics Hardware

Li-Yi Wei
NVIDIA Corporation

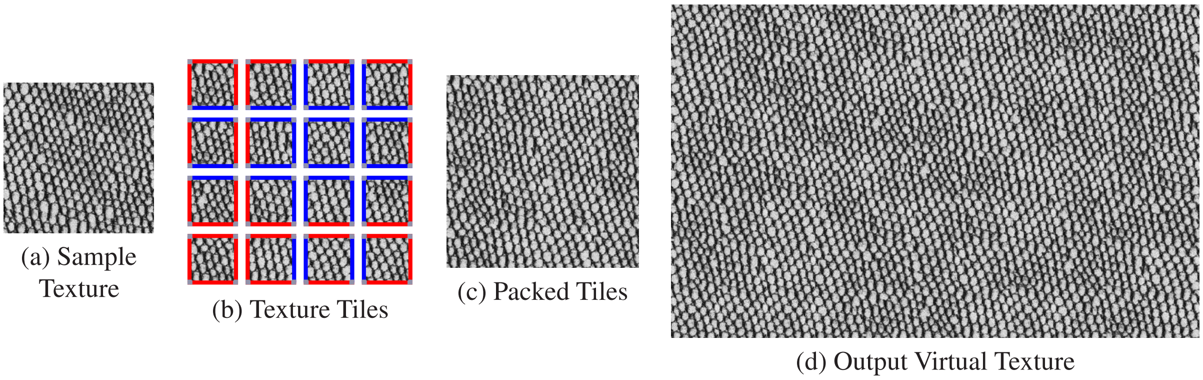

Figure 1: *Overview of our system. Given a sample texture (a), we first construct a set of texture tiles [Cohen et al. 2003] (b) so that adjacent tiles have continuous pattern across shared edges. We pack the tiles into a single texture map (c) and store it on a graphics processor. This texture map can then be sampled and filtered via a fragment program to support an arbitrarily large virtual texture (d) without physical storage.*

Abstract

Texture mapping has been a fundamental feature for commodity graphics hardware. However, a key challenge for texture mapping is how to store and manage large textures on graphics processors. In this paper, we present a tile-based texture mapping algorithm by which we only have to physically store a small set of texture tiles instead of a large texture. Our algorithm generates an arbitrarily large and non-periodic virtual texture map from the small set of stored texture tiles. Because we only have to store a small set of tiles, it minimizes the storage requirement to a small constant, regardless of the size of the virtual texture. In addition, the tiles are generated and packed into a single texture map, so that the hardware filtering of this packed texture map corresponds directly to the filtering of the virtual texture. We implement our algorithm as a fragment program, and demonstrate performance on latest graphics processors.

Summary

Texture mapping is a technique to represent surface details in computer rendered images without adding geometric complexity. Texture mapping has been a standard feature for recent consumer-level graphics hardware. Main cost of texture mapping on graphics hardware comes from the memory for texture storage, as well as the bandwidth to transfer and access those textures. For applications that use large amounts of textures, the storage or bandwidth requirements may prohibit real-time performance on graphics hardware.

One possible solution to address these problems is texture compression. However, texture compression techniques existing today are designed and optimized mainly for general images and may achieve sub-optimal compression ratio for textures that contain repetitive patterns. Even the most basic texture synthesis algorithms could beat the compression ratio compared to the texture compression method. Nevertheless, when it comes to real time applications on the graphics hardware, most texture synthesis algorithms are often too slow or too complex.

We adopt a different approach by creating a large virtual texture as a stochastic tiling of a small set of texture tiles. Specifically we use Wang Tiles [Cohen et al. 2003] as the basic texturing primitive in our system. Unlike [Cohen et al. 2003], which is only at the level of software implementation, ours allows the texture to be stored and accessed entirely within the graphics hardware.

Here is an overview of our system, with full details described in [Wei 2004]. At a preprocess step, we generate a set of Wang Tiles as described in [Cohen et al. 2003], and pack these tiles into a single texture map. We propose a tile-packing scheme that ensures correct mipmap filtering when a texel is fetched from this packed texture map. During the run time, for each texture request (s, t), we first determine which input tile it lands at based on the position of (s, t) within the output texture. We then compute the relative offset of (s, t) within that input tile, and fetch the corresponding texel from the packed texture map. Since out packing scheme supports correct mipmap filtering, the fetched input texel will be the same as a texel fetched from a large, physically stored texture map.

We have implemented our algorithm as a Cg fragment program. The performance measured on a 350 MHz Geforce FX 5600 graphics processor is 20 million trilinearly-filtered texture samples per second without any hand optimized assembly code. The speed and storage requirements of our algorithm remain roughly constant regardless of the size of the output virtual texture. In contrast, the performance of traditional texture map degrades with increasing texture sizes, and most graphics chips impose an upper limit on the available texture size. (The maximum texture size is 4K $\times$ 4K pixels on a Geforce FX 5600 GPU.)

Acknowledgements : I would like to thank Chung-Hui Chao for improving the writing style of this paper.

References

COHEN, M. F., SHADE, J., HILLER, S., AND DEUSSEN, O. 2003. Wang tiles for image and texture generation. *ACM Transactions on Graphics 22*, 3 (July), 287–294.

WEI, L.-Y., 2004. Tile-based texture mapping on graphics hardware. Submitted to *Graphics Hardware 2004*.