Fractal Image compression for efficient texture mapping

Jerzy Stachera
Institute of Computer Science
Warsaw University of Technology
Ul. Nowowiejska 15/19
00-665 Warszawa, POLAND
e-mail: J.Stachera@elka.pw.edu.pl

Sławomir Nikiel
Institute of Control and Computation Engineering
University of Zielona Góra
Ul. Podgórna 50
65-246 Zielona Góra, POLAND
e-mail: S.Nikiel@issi.uz.zgora.pl

ABSTRACT

Texture mapping has traditionally been used to add visual realism to computer graphics images. In the paper we propose an alternative technique for image texture handling. We use fractal image compression scheme to compress and decompress large and complex textures. We identify special properties of the method to apply in the very process of texture mapping. The main idea is to exploit high degree of local self-similarity in natural textures. The method allows for very high compression of complex textures, provides real-time decompression and perfectly suits Level of Detail. Properties of the method are compared to those of classical DCT and wavelet compression schemes.

Keywords

Image generation, Texture mapping, Fractal image compression, Multi-resolution.

1. Introduction

Texture mapping is a powerful technique for adding visual complexity to a computer generated scene and in its basic form it lays an image map onto polygon objects [Hec86a]. When mapped onto an object the appearance of the object is modified by a corresponding data from the image, these can be patterns, bumps or others [Gar85a][Pal03a]. The image is typically a sampled array so a continuous seamless texture must first be reconstructed from the samples. During the mapping the image must be warped to match perspective distortions. Additionally the warped texture is filtered to avoid aliasing artefacts. The required filtering is approximated by one of several methods. The most popular is mip mapping [Wil83a]. Other filtering techniques may also be used [Cro84a]. Over past years most of texture mapping has moved from software domain to high performance graphics hardware [Dee88a].

Current applications of three dimensional computer graphics, however, require much larger texture images, usually stored in PNG, TIFF and JPEG format, to achieve high degree of visual realism. This is particularly visible in the humanoids and the outdoor scenes where repeating patterns create unnatural feel of the image. Among most popular applications are 3D image editors for movie industry and 3D gaming. Considering film industry large image textures are used. They are often scanned from nature or even hand painted. For hand painted scenes the amount of texture map data is increasing rapidly with the number of layers used and the depth of undo buffer. Such textures can be easily stored on DVD and disk matrices off-line. In the case of internet based 3D MMORPG (Massive Multi-player Online Role Playing Games) the situation is different. Some general textures and geometry are stored and accessed locally but all changes in the scene require heavy mirroring and downloads of upgraded data. Geometry is not usually the problem but textures can be too big to download via internet connection. There is a constant need for efficient texture transfer method for such kinds of applications.

***WSCG Posters* proceedings**
WSCG'2004, February 2-6, 2004, Plzen, Czech Republic.

2. Problem definition

Most image compression schemes are focused on storage. Choosing a compression scheme appropriate for texture mapping several restrictions need to be considered [Bee96a].

Decoding speed. Textures are compressed once and decompressed many times, it is an asynchronous process. Thus, decoding speed it is a main factor

which states if given method is suitable for texture compression.

Compression Rate and Visual Quality. High compression rate offers better storage for more textures. It simplifies texture memory handling and texture downloads. While lossless compression methods preserve original texture data they do not achieve high compression rates in comparison to lossy methods. Moreover, texture compression method artefacts introduced by lossy methods are alleviated by texture filtering.

Random access. Random access to compressed texture data allows for rendering directly from compressed textures. Thus, texture compression scheme should provide fast random access to a single element of the texture. Otherwise, the process of decompression may limit performance of the rendering pipeline.

3. Fractal texture compression

3.1 Fractal image compression PIFS

Fractal image compression utilizes an extension of IFS called a partitioned iterated function system (PIFS). A PIFS consists of a complete metric space (X, d) a set of domains $D \in X$ and a set of contractive transformations $W : D \to X$. By the Contraction Mapping Fixed-Point Theorem W has a unique fixed point $f_w \in X$ satisfying $f_w = W(f_w)$. Thus to solve the image encoding problem we have to find a PIFS such that its fixed point f_w is as close as possible to the encoded image $I \in X$. Therefore it is required that W minimises the distance between its fixed point f_w and the encoded image I.

3.2 Image coding

Let I be an encoded image. A set R of non-overlapping range cells $R = \{r_1, r_2, ..., r_n\}$, $\forall_{i \neq j} r_i \cap r_j = \varnothing$ that tile I, $I = \bigcup r_i$. This set is called a range pool. A set D of overlapping domain cells $D = \{d_1, d_2, ..., d_m\}$ is called a domain pool $d_i \in I$. A set W consists of contractive transformations which for each range map on its corresponding domain cell $w_i : d_i \to r_i$. For a compact representation W is restricted to a class of transformations:

$$r_i = w_i(d_i) = s_i \varphi(d_i) + o_i$$

where:

φ - is a spatial contraction function which contracts domain cells to the size of range cells,

s_i - is a scaling factor, $s_i \in R$, $|s_i| < 1$,

o_i - is an offset value, $o_i \in R$.

The triplets (φ, s_i, o_i) are called transform parameters. The encoding problem of the image I is stated as follows:

- Partition the image I into non-overlapping range cells $r_i \in R$ that tile I,
- for each range cell r_i find a domain cell $d_i \in D$ and transform parameters such that $d(r_i, s_i \varphi(d_i) + o_i)$ is minimised,
- save transformation parameters (φ, s_i, o_i).

The process of image coding results in a set of contractive transformations W such that its fixed point f_w is an approximation to the image I. Thus W defines a lossy code for the image I [Bach93a].

3.3 Texture compression

The compression scheme for textures is based on the block oriented fractal compression scheme for images. To compress textures we define additional assumptions to a basic fractal encoding scheme:

1. $N \times N$ - size of the texture ($N = 2^l$),
2. $B \times B$ - size of a range block ($B = 2^n$),
3. $D \times D$ - size of a domain block is twice the size of a range block ($D = 2B$),
4. $\varphi(\cdot)$ - the spatial contraction function that averages four adjacent texture elements and then maps the averaged value onto the range block applying one of eight isometries.

The resulting range block values are scaled by s_i and added to o_i: $r_i = w_i(d_i) = s_i \varphi(d_i) + o_i$

In most cases textures are colour images in the RGB colour space. Because, the human visual system is not particularly sensitive to colour information we can take advantage of this insensitivity and convert texture to the YUV colour space. The YUV colour space consists of three channels: luminance channel Y and two chrominance channels U (hue), V (saturation). The chrominance channels store information about colour. It is possible to compress it with a little to no visible degradation, and additionally reduce the compression time.

Fractal encoding is a complex process because of the searching part of the algorithm. Attempts to improve the algorithm focused on domain classification and feature extraction [Wel99a]. In the proposed fractal texture compression scheme we exploit inherent local self-similarity presented in majority of textures by utilising nearest neighbour search strategy [Sau95a] and domain classification. To achieve a high compression rate we use quadtree partitioning. The transform parameters corresponding to the range blocks on different quadtree levels are quantised and coded with variable length codes.

The overall scheme for fractal texture compression consists of the following steps:

- convert texture colour space to the YUV colour space,
- average the chrominance channels (U, V) to one-half,
- for each component: Y, U', V' use quadtree partitioning method and the nearest neighbour search strategy,
- save header: quadtree depth, number of transformations, texture resolution, …
- save quantised transform parameters with quadtree information using variable length code.

3.4 Texture decompression

In decompression scheme we utilise hierarchical decoding introduced by Baharav Z. [Bah93a]. He stated that PIFS can be decoded in different spaces, yielding different fixed points in each space. The collection of fixed points is called a pyramid of the PIFS fixed points, where $f^{1/2^p}$ is the p level of the pyramid. Thus, $f^{1/2^p}$ has $N^2/2^p$ elements, where N^2 - size of texture ($N=2^l$), and f^1 corresponds to $p=0$ and is of length N^2. The level with the coarse resolution is called a top level. A relation between two different fixed points exists when the size of the range block is halved:

- in order to go from level p to level $p+1$, the operation is as follows:
  $$f^{1/2^{p+1}}(x,y) = 1/4\{f^{1/2^p}(x,y) + f^{1/2^p}(x+1,y) + f^{1/2^p}(x,y+1) + f^{1/2^p}(x+1,y+1)\}$$
  where: $x,y = 1,2,...,N^2/2^{p+1}$
- in order to go from level $p+1$ to level p, the operation is as follows:
  $f^{1/2^p}(x,y) = s_i f^{1/2^{p+1}}(x/2, y/2) + o_i$, where $x,y = 1,2,...,N^2/2^p$

The number of levels in a pyramid of the PIFS fixed points which satisfies above relation is equal to $\log_2(B)+1$, where B^2 range block size ($B=2^l$) and it is assumed that $D=2B$. The top level length is equal to $N^2/2^l$. A hierarchical decoding algorithm consists of the following steps:

- compute the top level fixed point, using iterative method or doing it directly [Lep93]
- advance to higher resolution, until the required size of the texture is achieved

The hierarchical decoding method is one order of magnitude less computational expensive than the iterative method (assuming $B^2 >> it$, where it is equal to number of iteration in iterative method) [Bah93a]. Moreover, we can decompress texture in a finite predetermined number of steps which depends on the partitioning of the texture and not on the texture image itself. The hierarchical method was introduced only for a fixed size range blocks. In our scheme we use modification of this method extended to the case of quadtree partitioning by Malah D. [Mal99a].

The proposed scheme utilises the hierarchical decompression method. In the first step the transformations (range block and domain block sizes) are scaled by a factor $1/2^{\max}$ (where $\max$ is equal to the maximum quadtree depth) in order to approximate the top level fixed point of the PIFS pyramid. At the top level we apply the transformation only once to level grey texture in order to approximate the fixed point $f^{1/2^{\max}}$. Then, we double the resolution by multiplying the transformation by a factor equal to two, and then we apply the transformation. The process is repeated until we get the required texture size.

The overall decompression process may be written in the following steps, for each YUV component:

1. read the quadtree representation and create an array of transformations,
2. multiply the transformation by a factor of $1/2^{\max}$
3. apply transformation to the top level
4. multiply transformation by 2
5. apply transformation to the next level
6. repeat 4), 5) until the required resolution is achieved
7. for U, V components - multiply transformation by 2, apply transformation (assuming, that U, V components are averaged to one-half).

4. Experimental results.

We have compared the proposed fractal compression scheme with JPEG algorithms on a set of test textures. The result are presented for textures saved in the 512×512 resolution (Table 1). For the fractal compression scheme, the minimum quadtree depth is 4 and maximum is 6. We used a bit allocation scheme for transform coefficients, where s_i (5-bit), o_i (7-bit), φ (3-bit for one of eight isometries) and variable length codeword for domain location. We use domain classification proposed by Y. Fisher [Fis95a]. Real-time decompression and texturing of 3D objects is one of the main purposes of our research. Thus, the experiments were carried out mainly to achieve high compression rates.

Image	FCI		JPEG		JPEG2000	
	CR	PSNR	CR	PSNR	CR	PSNR
Brick	89.4:1	28,9	81.7:1	29,26	253.1:1	29,5
Leaf	187.7:1	33,8	39.7:1	34,3	191.3:1	34,4
Sky	139.8:1	32,4	33.6:1	33,5	181.5:1	33,6
Marble	146.7:1	32,3	33.7:1	33.8	191.7:1	33,2
Lenna	74.1:1	26,7	24.1:1	27,8	77.9:1	28,58

*CR – compression ratio, PSNR – peak signal to noise ratio

Table 1. Compression method comparison.

Figure1. Bricks

Figure 2. A Leaf

Figure 3. Sky

Figure 4. Marble

Figure 5. Lenna

5. Discussion

Most image compression algorithms based on transform methods such as JPEG and JPEG2000 are characterized by multi-staged decompression algorithm and expensive computation of inverse transform. The computation complexity of the DCT transform or the wavelet functions for a texture element is $O(n^2)$, whereas for the fractal method is linear $O(n)$. Moreover the direct representation of compressed texture as a set of transformation simplifies the process of decompression.

The results of the fractal compression show that it achieves higher compression rates and image quality than the DCT (JPEG) and is comparable to the wavelet functions (JPEG2000). Further compression can be achieved by entropy coding of transform coefficients [Cis96a][Cho01a].

In most cases the aliasing problem in texture mapping is solved by the interpolations method. These methods tend to smooth the texture, resulting in a blurred texture image. The proposed scheme solves the problem by utilising fractal properties, namely: super-resolution and simple decoding algorithm. The super resolution property is used in proposed scheme to decode the chrominance channels at a twice than encoded resolution. The approach preserves details in colour information at higher resolutions. Moreover, the simple decoding algorithm makes it possible to implement it in a dedicated hardware for real-time texturing.

6. Conclusions and future work

Texture mapping is a highly efficient technique adding complex visual detail to three dimensional geometry. Popular PC-based systems offer each year a new generation of accelerated graphics boards providing more realistic computer generated images. Much of this visual realism is achieved thanks to advanced texture mapping based on natural images rather than on procedural textures. Concerning internet based applications, like MMORPG, there is still need for efficient image texture handling. High efficient compression scheme proposed in the paper works well with natural textures. Our method utilises local self-similarity in natural images allowing for multi-resolution decompression. Fractal texture handling offers significant savings both in storage and in time of decompression which is close to meet real-time constraints. Our future work will focus on investigation on fractal compression of volumetric textures and on development of highly decentralised codec for very large image textures e.g. satellite image textures for terrain models.

7. References

[Bah93a] Baharav Z., Malah D., Karnin E., Hierarchical Interpretation of Fractal Image Coding and Its Application, Proceedings of the IEEE International Conference on Digital Signal Processing, pp. 190-195, Nicosia, Cyprus, July 1993.

[Bee96a] Beers A., Agrawala M., Rendering from compressed textures, Siggraph 1996, July 1996.

[Cho01a] Fast Fractal Image Encoding Based on Adaptive Search, IEEE Transactions on Image processing, Vol. 10, No. 9, 2001

[Cis96a] Cisar G., On entropy coding Fisher's fractal quadtree code, Technical report, Institut für Informatik, University of Freiburg, June 1996.

[Cro84a] Crow F. , Summed-Area Tables for Texture Mapping, Computer Graphics, Siggraph'84 Proceedings, July, 1984, pp. 207-212

[Dee88a] Deering M. Et al., The Triangle Processor and normal Vector Shader: A VLSI system for High Performance Graphics, Computer Graphics, Siggraph'88 Proceedings, August, 1988, pp.21-30

[Fis95a] Fisher Y., Fractal Image compression: theory and application, Springer-Verlag, 1995

[Gar85a] Gardner G., Visual simulation of Clouds, Computer Graphics, Siggraph'85 Proceedings, July,1985, pp. 297-303

[Hec86a] Heckbert P., Survey of Texture Mapping, IEEE Computer Graphics and Applications, November 1986, pp. 56-67

[Lep93] Attractor image compression with a fast non-iterative decoding algorithm, In. Proc. ICASSP, pp. 5:337-340,1993

[Mal99a] Malah D., Hierarchical fast decoding of fractal image representation using quadtree partitioning, Technion - I.I.T, Israel

[Pal03a] Palewski M., Fractal shading model, MSc thesis, Institute of Control and Computation Engineering, University of Zielona Góra, Zielona Góra, 2. 2003 (in Polish)

[Sau95a]Saupe D., Fractal Image Compression via Nearest Neighbor Search, NATO Advance Study Institute Fractal Image Encoding and Analysis,Trondheim, 1995

[Wel99a] Welstead S., Fractal and Wavelet Image Compression Techniques, SPIE, 1999

[Wil83a] Williams L., Pyramidal Parametrics, Computer Graphics, Siggraph'83 Proceedings, July 1983, pp. 1-1