VIEW-DEPENDENT TEXTURE CODING FOR TRANSMISSION OF VIRTUAL ENVIRONMENT

S. Horbelt, F. Jordan, T. Ebrahimi,

Swiss Federal Institute of Technology, Switzerland

Abstract: We propose a novel technique for coding of texture and three-dimensional data which takes the viewer position into account. This allows to transmit only the most visible parts of information that is needed to render a virtual scene. The rendering operation is modelled and studied using filtering and sampling theory. The technique is applied on real data and results are given for an operational encoding system.

Keywords: view-dependent scalability, anisotropic filtering, DCT filtering, very low bitrate, synthetic texture coding, back-channel, computer graphics.

1 INTRODUCTION

The development of digital communication has led to an astounding amount of research in the field of compression and transmission of video information. This research has developed several standards for the coding of digital video, which are now widely used. In parallel, digital networks like the Internet provide new means for exchange of visual information. This kind of network offers the ability for the user to interact directly with a remote database, which was not the case in TV broadcasting conditions, for instance. In parallel, we see the emergence of virtual reality applications. It turns out that the distributed and interactive access to synthetic worlds could open incredible breakthroughs for the development of visual communication. In order to allow such interactive access to remote data, the development of techniques for optimal coding of synthetic 2D/3D data become of utmost importance. This will lead to the definition of a new generation of standards [1, 6].

We propose a novel way to obtain an optimal coding of texture and 3D data. It consists of taking into account the position of the viewer in the 3D virtual world in order to transmit only the most *visible* information. This approach allows therefore to reduce greatly the transmitted information between a remote database and a user. At a given time, only the most important information is sent, depending on object geometry and viewpoint displacement.

For instance, if we consider a single plane on which a texture image is mapped, one can notice that the information displayed after rendering uses only a small amount of the data contained in the original texture. This depends on the distortions caused by the rendering process. They are due to perspective distortion, which depend upon orientation, distance or field-of-view as well as other criteria such as illumination, shadows, fog effect, motion blur, etc.

Little research has already been performed on this topic, nevertheless, several works in computer graphics have been made on model simplification in order to decrease the computing load of graphical engine [2]. Some of the ideas used in this field can be exploited for the optimal coding of data. In particular mip-mapping techniques [3] offer a solution to texture multi-resolution. Furthermore, Peter Lindstrom [4, 5] has remarked that texture resolution could be decreased by taking into account the viewer position.

2 RENDERING OPERATION

2.1 INTRODUCTION

The rendering operation is the mathematical transformation which projects on a 2D plane, the image of the 3D scene. Two kinds of projections may be used: orthogonal and perspective projections.

This section will show that perspective projection of an image, e.g. texture, can be approximated by a linear transformation.

2.2 PERSPECTIVE PROJECTION

As was shown by [5], the perspective projection of a line of length *L* is a line of length *L’* which can be written as:

(1) $$L' = L \cdot v \cdot \cos(\alpha) \cdot \frac{4D}{4D^2 - L^2 \cdot \sin^2(\alpha)}$$

with *D* the distance between viewpoint and line center, α the angle with the line and *v* the distance to the projection plane.

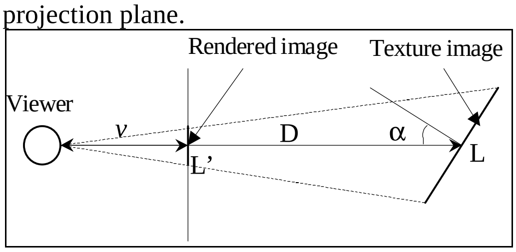

Figure 1

(2) In case, $D >> L$, $L' = L \cdot v \cdot \frac{\cos\alpha}{D} = k \cdot L$

Suppose a one-dimensional texture *f(x)* defined along the x-axis with $x \in [0, L[$. The perspective projection of this one-dimensional pattern *f(x)* is then a simple change of scale of this pattern:

(3) $f'(x) = f(k.x)$

More generally, [7] has shown that in the case of orthogonal projection, the projection of a two-dimensional texture *f(x, y)* is:

(4) $f'(x, y) = f(ax + by, cx + dy)$

In the following, we use (4) in the case of perspective projection, using the condition given in (2).

3 FILTERING OPERATION

3.1 INTRODUCTION

We will show in the following that the projection of a texture image can be considered as a filtering operation followed by a down-sampling.

3.2 DEPTH EFFECT

Let us consider the case of a texture *f(x,y)* which is mapped on a plane, normal to the viewing direction.

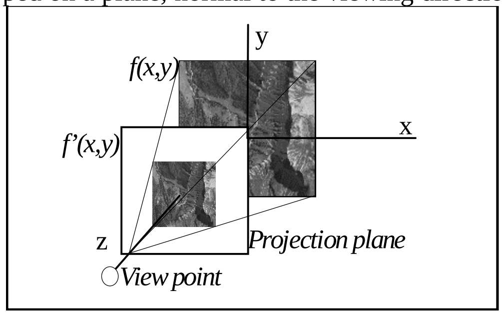

Figure 2: depth effect, k=2

This projection is a particular case of (4) with *b=c=0* and *a=d=k*. *k* is proportional to *D: k∝D.*

(5) $f'(x,y) = f(kx,ky)$

So, if *f(x,y)* is defined at L∗L discrete points (square texture), the number *N* of points which are rendered on the screen is:

(6) $N = \frac{L^2}{k^2}$

If we call *F(u ,v)* the Fourier transform of *f*, and FT the function which associates *F* to *f*, we then have:

(7) $\mathrm{FT}(f'(x,y)) = \frac{1}{k^2} F(\frac{u}{k},\frac{v}{k})$

Which means as noticed by [7] that the frequencies of the projected version is increased by a factor k in both dimensions. As a consequence, a low-passed version of *f* will have a projection similar to *f'*. This corresponds to a usual down-sampling operation.

Figure 2 shows an example where the distance has been chosen such that only a quarter of the frequency information is used, compared to the original texture.

3.3 TILTING EFFECT IN ORTHOGONAL PROJECTION

Let us consider the case of plane tilted around y-axis and projected on the x-y-plane.

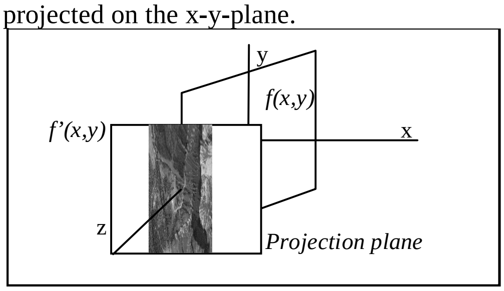

Figure 3: tilting effect, cos α= 0.5

The tilting of the square texture *f(x,y)* mapped on this plane is another particular case of (4) which can be written in orthogonal projection as:

(8) $f'(x,y) = f(kx,y)$

with *b=c=0*, *d=0* and *a=k* with $k \propto 1/\cos\alpha$.

3.4 ROTATION EFFECT IN ORTHOGONAL PROJECTION

Let us consider again the case of a plane normal to viewing direction (α=0). We rotate the plane by the angle β, then its FT is known by [9]:

(9) $\mathrm{FT}(f(x\cdot\cos\beta + y\cdot\sin\beta, -x\cdot\sin\beta + y\cdot\cos\beta)) = F(u\cdot\cos\beta + v\cdot\sin\beta, -u\cdot\sin\beta + v\cdot\cos\beta)$

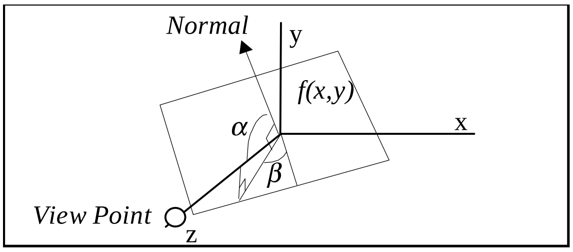

Figure 4: Orientation of the plane

The combination of this rotation with tilting and depth enables to describe the effect in the frequency domain of any orientation of the plane.

3.5 CONCLUSION

More generally, for a given viewpoint, we have shown that the projection of a texture is equivalent to an *anisotropic filtering* of the texture image. Consequently it is possible to decrease the number of bits necessary to encode a texture image by transmitting only a downscaled version of the texture.

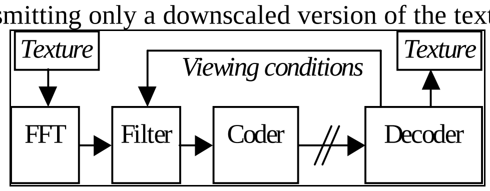

Figure 5: View-dependent coding and decoding

4 USE OF DCT FOR FILTERING

4.1 INTRODUCTION

In the following, we show how DCT can be used to approximate the above described filtering operation.

Let *H(u,v)* be the DFT of the transfer function of a filter and *h(x)* its impulse response. Let note *Fc(u,v)* the DCT of *f(x,y)* and *Wc(u,v)* the DCT of the filtered texture *w(x,y)*.

It has been shown by [11] that if:

(10) $Wc(u,v) = Fc(u,v)\cdot H(u,v)$

(where u,v are the coordinates in DCT and Fourier space respectively for *Fc* and *H*)

Let $\hat{f}(x,y)$ be the four-fold symmetric extension of the function *f(x,y)*. Then

(11) $w(x,y) = \hat{f}(x,y) * h(x,y)$

assuming that H is real and even:

(12) $H(u,v) = H(-u,-v) = H(-u,v) = H(u,-v)$

Thus, we can replace a filtering operation by using the DCT transform of a texture image *f(x,y)* as noticed by [8]. For coding efficiency reasons, we will use only binary filters H(u,v) in the following. These kinds of filters applied in DCT domain have already been studied by [12, 8, 10]. This filtering operation can be compared to a zonal coding [14, 15].

4.2 DEPTH EFFECT

As shown by [10], this operation can be considered as a simple change of scale. The change of scale using DCT transform has already been studied by [8, 10, 12]. It is possible to use a filter $Hc_k(u,v)$ applied on *Fc(u,v)* as in (10) with the shape described in Figure 6. In practice we simply compute [5]:

(13) $Wc\ (u,v) = Fc(u,v) \cdot Hc_k(u,v)$

with

$$Hc_k(u,v) = 0, \text{if } u > \frac{L-1}{k} \text{ or } v > \frac{L-1}{k}$$

(14) $Hc_k(u,v) = 1, \text{elsewhere}$

and $\{u,v\} \in \{[0,L-1],[0,L-1]\}$

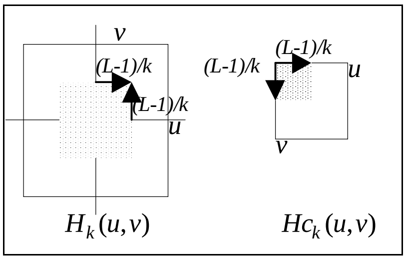

Figure 6: Filter design for depth effect

4.3 TILTING EFFECT IN ORTHOGONAL PROJECTION

As shown by (8) this effect can be seen as a simple change of scale in one direction. The corresponding H_k and Hc_k filters are defined in Figure 7.

$$Hc_k(u,v) = 0, \text{if } u > \frac{L-1}{k}$$

(15) $Hc_k(u,v) = 1, \text{else.}$

with $\{u,v\} \in \{[0,L-1],[0,L-1]\}$

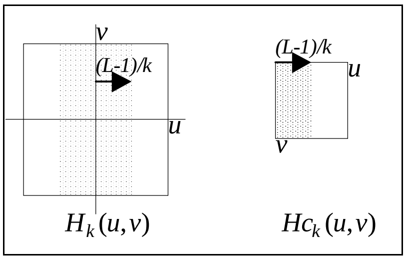

Figure 7: Filter design for tilting effect

4.4 ROTATION EFFECT IN ORTHOGONAL PROJECTION

The directional filtering in DCT domain has already been addressed by [13]. Taking into account equation (9), it follows that the optimal filter in Fourier domain should be as shown in Figure 8.

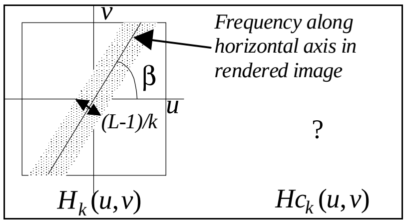

Figure 8: Optimal filter for rotation effect

Since this is a complex filter, there is no way to implement it by one operation in the DCT domain.

We have studied two real filters as alternatives to the optimal filter. These filters are symmetrical as shown by (12). These two filters are sub-optimal for different reasons:

- H1 is not removing some invisible frequencies.
- H2 is removing some visible frequencies.

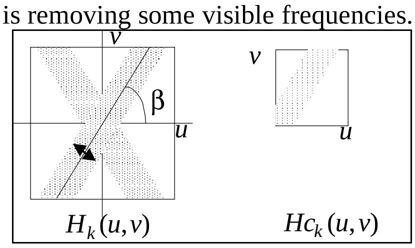

Figure 9: Real filter 1

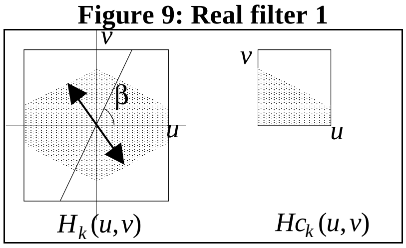

Figure 10: Real filter 2

Let us design the two filters to make a quantitative comparison between them (Figure 11). The plane is rotated by β=45° and titled by α=60°, which means that only half of the pixels of original texture are rendered on the screen. Thus, we design the filters such that their non-zero surface *S* covers half of the filter plane.

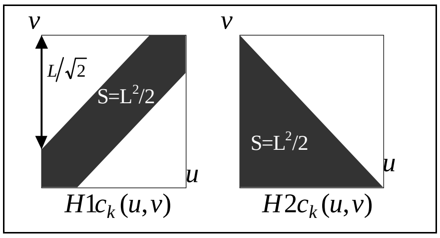

Figure 11: Shapes of compared filters.

For each of the filters we have made the following experiment:

1. DCT of the a texture f(x,y) with size L=128
2. Application of filter 1 or 2
3. IDCT of the texture
4. Texture mapping and rendering

Results are shown in Figure 12. We have compared these two images to the one we get with the original texture.

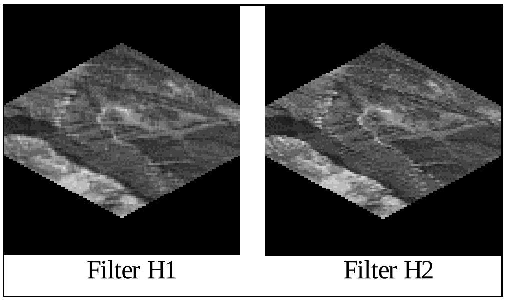

Figure 12: Results using filter H1 or H2

Technique	PSNR (dB)	Removed DCT
Filter H1	32.2811	8010
Filter H2	36.2233	8128

Table 1: PSNR using filter H1 or H2

Table 1 gives for each filter both the PSNR and the number of DCT coefficients that have been set to zero by the used filter.

The conclusion is that for this particular orientation filter H2 gives much better results. Although H2 is filtering frequencies which appear horizontal on the rendered image, this effect is less visible than the filtering performed by H1.

4.5 GENERALIZATION IN PERSPECTIVE PROJECTION

The assumption is used that perspective projection can be approximated locally with orthogonal projection. We use the rotation filter 2 combined with the depth filter to define the shrinking mask $[a_{uv}]$ using the depth D and the rotation angles α and β.

For all shrinking mask coefficients a_{uv} with

$v \in [0, L-1]$ and $u \in [0, L-1]$,

We define the normalized coordinates:

$V=(v+0.5)/L, \quad V \in [0,1]$

$U=(u+0.5)/L, \quad U \in [0,1]$

Let define the threshold value T as:

$T = (k/D) \cdot \cos \alpha;$

The shrinking angle β_S is defined as:

$\beta_S = | \arccos(\cos(90-\beta)) |$

Then for each a_{ij}:

if	$(U>k/D$ or $V>k/D)$	then $a_{uv} = 0$
if	$(\sin \beta_S = 0$ and $V> T)$	then $a_{uv} = 0$
else if	$(\sin \beta_S = 1$ and $U> T)$	then $a_{uv} = 0$
else if	$(U > T-(V-T)/ \tan \beta_S)$	then $a_{uv} = 0$

5 TEXTURE DATA REDUCTION PRINCIPLE

5.1 INTRODUCTION

We have been focusing on the coding of texture information that is used for generating photo-realistic landscapes [5]. This kind of data typically represents a large amount of space (several megabytes). However at a given time, only a small portion of it is really needed for visualization. The filtering described in the last chapter can be used for reducing the transmitted texture data mapped onto any kind of 3D object.

5.2 GENERALIZATION TO A REGULAR GRID MESH

The filtering discussed in the last chapter for one plane is now generalized for a 3D regular grid mesh.

We use, at the decoder side, the knowledge of the 3D regular mesh on which the texture is mapped. The texture is split into small 8×8 sized blocks, which are aligned to the cells of the 3D regular grid mesh. The DCT is computed for each texture block. Then each DCT block is filtered *independently* with the shrinking mask. It uses the local orientation and depth of the grid cell, which belongs to this DCT block.

Only the DCT coefficient with a corresponding none-zero shrinking mask coefficient a_{uv} are considered for the transmission and coding.

5.3 STREAMING TEXTURE DATA

Once the texture data needed for the first frame (we call the data *intra data*) has been loaded using the proposed method, it is then straightforward to send new DCT coefficients (we called them *inter data*) as the observer is moving in the scene. Therefore, it is possible to update the texture information according to the new viewer position using a very low bit rate channel. (Figure 13)

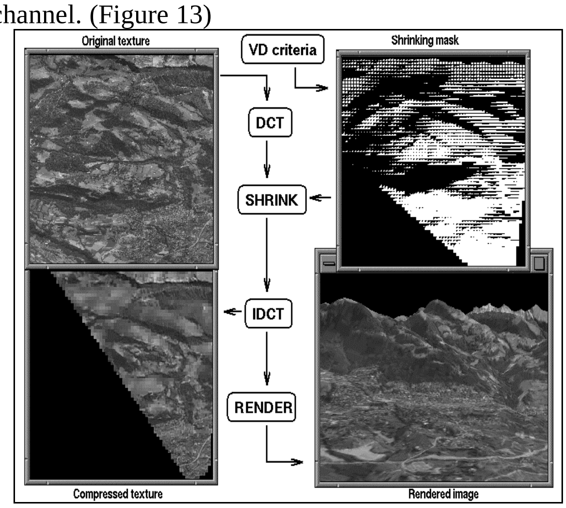

Figure 13: View-dependent compression scheme.

5.4 COLOR COMPONENTS

The colors are compressed by applying the same technique on the 1:4 down-sampled versions of the U and V texture components.

5.5 ENCODING THE BITSTREAM

The DCT coefficient remaining after the shrinking are quantized using the quantization tables of MPEG-2, zig-zag-scanned and variable length coded (VLC) using the VLC tables of MPEG-4, as detailed in [16].

Due to the shrinking operation and quantization the inter blocks often contain only zero valued DCT AC coefficients. So for each block a one bit entry in the coded pattern value (CPV) table denotes whether any none-zero DCT AC coefficients will be sent.

The DC coefficients are DPCM and VLC encoded. Its predictor uses the DC values of the previous blocks along the two texture axis.

6 RESULTS

Tests have been done with a flight over a landscape defined by a 128×128 sized elevation grid. The texture was an RGB image of size 1024×1024 pixels, which represents a size of 24Mbits. Figure 14 shows the bitrate obtained to generate each of the frames of the sequence. This bitrate does not include the transmission of the elevation data, which represent a raw data of 32768 bytes with 16 bits per elevation point (1 % of the texture data). The first frame requires transmitting 358242 bits, then the numbers of bits vary between 1000 and 20000 bits for each frame. For the first frame the intra compression ratio is 1:11, for the following frames the inter compression ratio is 1:460 and the overall compression ratio is 1:300.

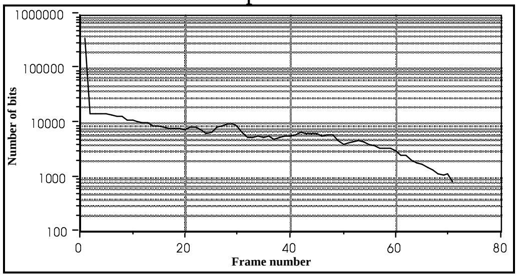

Figure 14: Number of bits transmitted to update the texture image, as the viewpoint is moving.

7 CONCLUSION

We have proposed a technique that uses a back-channel from the decoder to the coder. This takes into account the viewing conditions to specify the data required by the decoder. It enables to transmit and update texture data as the viewpoint is changing. Results show that it is possible to reach very low bit rate coding while keeping high quality of the rendered images. This technique has been proposed as a standard for coding of texture images in the framework of MPEG-4 standardization activity.

Finally, the technique can be easily extended to take into account other perceptive criteria, like shadows, speed or fog and therefore allow reducing even more the size of transmitted data.

This technique covers a wide range of applications: Streaming of flight simulation data at very low bit-rates, augmented reality for IFR conditions flying or exploration of planet surfaces. Streaming of data for virtual walks through buildings, eg. in architecture or virtual multi-player worlds.

We would like to thank Swissair for providing us with photo-realistic landscape textures and elevation data.

8 REFERENCES

1. Peter K. Doenges, Frederic Jordan, Igor S.Pandzic, 1996, "SNHC Application Objectives and Requirements", MPEG-4 technical report M1316, Tempere, Finland.
2. Funkhouser, T.A. and Sequin, Proceedings of SIGGRAPH 93, pp. 247-254.
3. Jackie Neider, Tom Davis, Mason Woo, OpenGL programming guide.
4. Peter Lindstrom, David Koller, Larry F.Hodges, William Ribarsky, Nick Faust, Proc. of SIGGRAPH 96, pp. 109-118.
5. Peter Lindstrom, David Koller, Larry F.Hodges, William Ribarsky, Nick Faust, “Level-of-detail management for real-time rendering of photo-textured terrain”, Internal report of Georgia Tech, http://www.cc.gatech.edu/gvu/research/techreports95.html
6. Fred Jordan, Touradj Ebrahimi, 1996, “View-dependent texture for transmission of virtual environment”, ISO&IEC JTC1, M1501.
7. Don Pearson, 1990, Signal Processing: Image communication, vol. 2, no. 4, pp. 377-395.
8. King N.Ngan, 1986, Signal Processing, vol. 11, pp. 249-263.
9. Kenneth R. Castleman, “Digital Image Processing”, Prentice-Hall.
10. Stephen A. Martucci, 1996, Proceedings of ICIP’95, vol. 2, pp. 244-247.
11. Chitprasert B., Rao K.R, 1990, Signal Processing, vol. 19, no. 3, pp. 233-245.
12. Adant J.M., and al., 1987, Signal Processing, vol. 13, pp. 285-397.
13. Toshio Tohne, Hiromi Koda, Hatsukazu Tanaka, 1996, Systems and Computers in Japan, vol. 27, no. 3, pp. 94-105.
14. Gonzalez Rafael, Woods Richard, “Digital Image Processing”, Addison –Wesley.
15. Stefan Horbelt, Jon Crowcroft, 1996, “A hybrid BTC/ ADCT video codex simulation bench", 7th Int. Workshop on Packet Video, Brisbane, Australia.
16. Stefan Horbelt, Fred Jordan, Touradj Ebrahimi, 1997, “View Dependent Texture Coding for Transmission of Virtual Environment”, ISO/IECJTC1/SC29/WG11, MPEG96/M1723.