Journal of Advances and Scholarly Researches in Allied Education
Vol. 16, Issue No. 9, June-2019, ISSN 2230-7540

Study on Design Textures on Large Format 3D Modals

Dr. Bipin Sinha*

Assistant Professor of Physics, Barsi College Pandygangout Nawada, Magadh University

Abstract – To create a bright 3D solid texture, a simple 2D texture pattern, an effective model-independent 3D texture synthesis algorithm is presented based on texture turbulence and texture turbulence. Due to the 2D texture pattern of some increasing material, our technique can produce a 3D volumetric cube anisotropic texture to simulate 3D material development. An effective tiling system is designed to save computer and storage costs. Target objects are immersed directly in the synthesised volume of 3D textures in order to generate creative, sculptural models that can be visualised interactively. Compared to other solid texture methods, our method is conceptually intuitive, quick computational and efficient storage. Our approach is able to seamlessly decorate 3D point rendering systems, contrary to conventional 2D texture maps on polygonal surfaces. Our combination of texture turbulence and texture-growing techniques also offers an attractive way of synthesising, tiling and creating natural 2D textures or simply interesting movement textures.

Keywords: 3d Models, Texture Design, Mapping Texture.

----------X----------

INTRODUCTION

3D MODELS

Surfaces represent 3D models in the most common geometrical format. They are used extensively because they are compact and easy to build, transmit and render. However, the application of surface representation is limited because internal information is lacking. For example, the model cannot be cut and the detailed internal structures inspected.

The additional advantages and limitations of volumetric representations. The user can cut the model and observe internal structures as the volumetric representation stores internal information. However, the amount of data required generally exceeds the amount of surface representation, which makes it considerably harder to store, transmit, model and render. Modeling is particularly problematic, although more memory, faster processors and networks can eventually mitigate further issues. Since modelling problems are very closely related to human perception and manipulation limitations, it is important for the design of appropriate user interfaces to address them. Capturing real-world objects and the design of procedures are the main sources of volume data. These are not suitable for the speedy drawing for communication purposes of volumetric illustrations. Some interactive methods use 3D input devices (e.g. Phantom), but do not work for detailed inner textures.

Texture analysis and synthesis have attracted a lot of interest in both 2-D and 3-D in recent years. Computer graphics apps often use textures to decorate virtual objects without modelling geometric details. This is frequently done by first synthesising the texture pattern in order to simulate the characteristics of the sample texture image and then attach it to the volume or surface of the decoded virtual object. The most important issue is to resolve the link between distortion or discontinuity between adjacent patches. The mapping of a specific texture pattern to conventional polygon objects was investigated over the past decade. There was, however, little work on the texture mapping for the point rendering system.

Point rendering is an attractive approach because of its speed and simplicity to render visually complex objects. To date , the main focus of the point rendering work has been on issues like the point data structure, the selection of shape points, the bending hole and texture changes. Now, most point rendering systems only involve the transmission of original textures predefined in point or splat data structures. The way to generate new textures from either any given texture pattern or using a methodical

approach to point-rendered objects is not very considered. However, many applications require a certain texture to be simulated using a point rendering technique. In this paper we propose a method to simulate the volumes of 2D texture re-available. This method is called evolution of texture.

To simulate the developments of specified 2D texture and produce a tileable 3D solid texture cube, the concept of texture turbulence and texture growth is used. The model-independent 3D texture mapping with these synthesised solid texture structures is a trivial and effective job because all you need to do is dip the object into a 3D texture volume. This method extends the work previously carried out to simulate solid textures from 2D patterns and takes the main advantages of solid textures such as model-independent, resolution scalable and non-continuity. The creation of vivid virtual works of art such as wood carvings and stone sculptures is particularly well suited for simulating natural solid texture, such as marbles and woods. Tiling is used to reduce the computing and storage burden. The resulting solid textures can easily be transferred to point-rendering systems, which are not suitable for the most polygonal surface texture processes. Furthermore, the combined use of texture growth and texture turbulence leads to interesting solutions for the production of 2D texture synthesising and temporal texture. The rest of the paper is arranged accordingly. In Sect, related work is introduced briefly. 2. Section 3 describes our texture-synthesis method. The test results are presented in Sect. 4. Finally, the sect. And finally. 5 The paper concludes and talks about future work

Our objective is to develop an interactive design and browsing system that enables users to add textures to the surface mesh manually using existing 2D reference pictures (see figure 1).

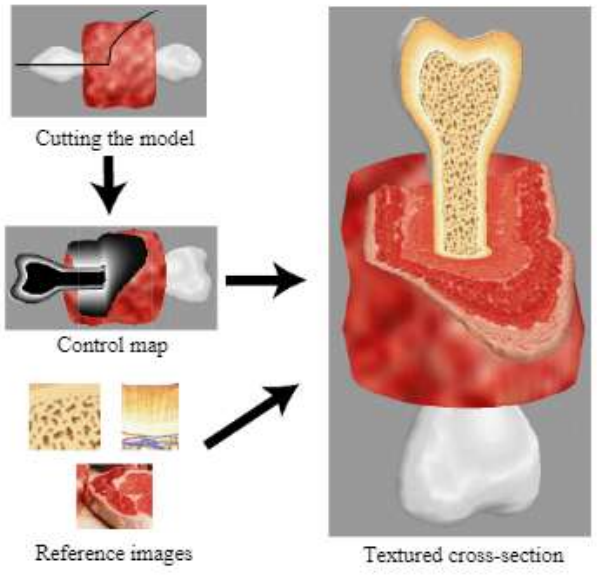


Figure 1 Reference image Textured cross-section

All 3D volumetric data cannot be seen at once; we see only one 2D cross-section at a time because of an occlusion. For instance, biology textbooks or scientific journals often show cross-sections of an internal structure of a bulk object (Figure 2). It is also important to note, instead of literally simulating reality, that the illustrations are the product of careful design processes. For example, the nucleus appears in the illustration in all cells, whereas the actual cross-section contained cells which did not appear in the nucleus.

Building on this observation, we suggest a new representation of 3d models with internal textures, one whereby, instead of keeping all 3D volumetrical data, the system synthesises internal textures with 2D reference images for cross sections (Figure 1). The designer assigns internal textures to the model by providing guidance information such as the orientation of the stream between the geometric cross-section and the 2D reference image. This approach reduces the number of data required by the model significantly. Designers can also add the internal 3D textures to the model, without specifying each voxel manually. The interfaces used to allocate internal texture to a particular surface mesh and algorithms that synthesise textures to a cross section, are our technical contribution. Our contribution to a larger scale is a modelling structure where simple volumetric textured models are easily and conveniently defined and viewed, so that non-experts can quickly produce volumetric illustrations.

Texture Mapping Background

In general, Catmull [1975] is attributed to the groundbreaking work on texture mapping. He was probably the first person to show an arbitrary flisher and cylindry surfaces drawing of a (brick) texture pattern. The idea was then taken one step further by Blinn and Newell [1976] by mapping photographic textures on a bicubic patch of now famous teapot. The technique of making synthetic textures on the faces of 3D-House models used independently by Aoki and Levine[1978] and by Feibush et al.[1980]. A structure mapping system for architectural design has also been designed by feibush and greenberg[1980]. The concept of computer "cell" textures in the field of visual flight simulators was later introduced by Economy and Bunker [1984] and Bunker et al. [1984], while Dungan et al. used actual photographic texturing patterns in the field. The latter was taken from low-flying planes with vertical pictures of representative trees, and photographs of grass texture were taken from model boards. The following was taken. In their simulated scenes, these textures have been mapped to 3D polygonal terrain height models. These authors were the first to provide examples of multi-level resolution textures as a method of anti-aliasing. Then, General Electric scientists [Economy and Bunker 1984; Bunker et al. The texture mapping apps of

individual pictures on flat billboards, multifaceted 3D tree models developed by Honeywell and Honeywell [Scott 1983; Erickson etcoll. 1984]. In order to make the texture regions around trees and between leaves invisible, transparency effects were introduced. Similar translucent techniques for inserting texture patterns of cloud layers in simulated scenes have been described by Bunker and others.

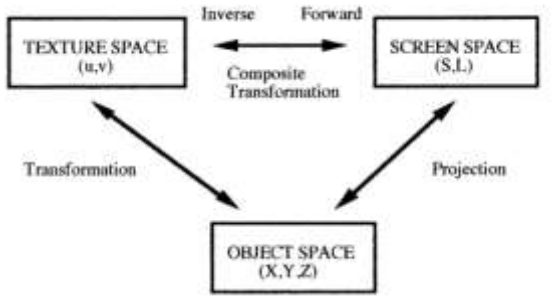


Figure 2. Texture-mapping geometry

Figure 2 illustrates the geometry of the structure mapping. It involves two transformations. One transformation is from space for objects, and space for texture (input) to space for screens. The object's space is distinguished by the equation of the 3D model surfaces. When the texture of the object is orthogonally mapped to a flat quadrilateral it can be as simple as an affine or bilinear transformation. Alternatively, when non-Cartesian co-operations, like the cylindrical or spherical ones, can represent the texture co-ordinates and textures "wrapped up" around a model 3D of the same symmetry, parametric transformation can take place.

Texture Mapping Approaches

Using most graphics systems, the fundamental, non-textured rendering technique generally projects the vertices of the primitive (usually flat polygon) from world coorders to screen coorders using a projection equating perspective as shown in Table 2. The colour (and depth of perspective) at each edge of the projected primitive and inside pixels are then linearly interpolated. The interplay of colour and perspective-depth values for each vertex is typically based on an algorithm such as the Digital differencial analyser (DDA) [Swanson and Thayer 1986; Pineda 1988] or the forward difference technique [Lien et al. 1987; Shantz and Lien 1987]. Such algorithms are very efficient because each scanning line of the planned primitive uses incremental calculations. An example of this sort of interpolation is the popular Gouraud Shading Technique, in which the information about shading associated with the vertex is determined by a lighting model together with the ordinary vertex. Phong shading technique is similar, but the normal vertex interpolates before the value of each inner pixel is defined. For instance, see for details on both of these specific shading techniques Rogers [1985] and Foley et al [1993].

REVIEW OF LITERATURE

Bowyer et al. (2012) Concentrate on a technology for facial recognition using 3D data individually or in conjunction with texture. The author discusses the restriction of 3D facial recognition and the future needs to overcome problems in the same paper.

Sheenstra et al.(2013) Provide a short review focusing on facial recognition techniques and Kitter et al. Present a brief survey of 3D image sensors, 3D face models, 3D imagery and 3D face recognition.

Wen et al. (2014) It also provides an overview of various 3D face treatment topics, including a 3D image reconstruction section. In this study , we focus on 3D reconstruction technology based on 2D face or video frames. This is the first study of our literature on the subject of 3D face reconstruction, to the best of our knowledge.

Blanz et al (2015) Use a non-stiff alignment method based on optical flow. The two faces to be aligned are parameters for cylinder coordinates like the angle and height where for each point of the face the radius and the colour are stored. The algorithm created a flow field which calculates pairs of points that take the texture and form into account. This method is used when the texture of ex one façade and bread is highly different and the other face is less contrasting. Therefore, various restrictions and smooth interpolation are imposed in order to overcome this problem so as to avoid false results. Optical flow is performed at various resolution levels.

Hutton et al. (2016) Generate a PCA model for a dense 3D surface using 400 scans from the age of 0 to 50. Based on the nine manual sites at each workout surface, all samples were aligned and a dense surface pattern was produced for each training form.

Hyun chul choi et.al.(2015) It introduces a pose-invariant facial recognising method using a 3D morph and the neural network with a training image and a query image. The system uses the 3D morph to obtain a rebuilt 3D face from the training picture and to obtain a 2D imagery patch of the 3D face component under various head positions. These patches are for training the neural network with the ability to be robust in query image 32. With the BJUT 3D scanning database, the position invariant facial recognition is over 98% accurately recognised.

Victor-Emil et al (2016) Investigate the identification of 3D faces by proposing a processing stage (a) calculation of the 3D image depth threshold maps; (b) normalisation and alignment; (c) the extraction characteristic of the

gabbord wavelet, which includes the extracting of the philter; (d) the analysis of the principal component(PCA)(e). Compared to 83.60% obtained by k-mean and 88.52% obtained by NN, som 's best experimental results lead to a recognition rate of 95.2%.

Turk et al (2013) The correct recognition rate reported for the FERET database was 95 percent with approximately 3,000 different faces. The image of the face tested appears to be taken with little difference in vision and light, although the facial expressions differ significantly. The main drawback of the Eigenface approach is that the dispersion is maximised not only because of interclass dispersion, which is useful for classification but also because of the 'interclass dispersion' used for classifying purposes. For a comparative purpose, researchers set up their own faces.

Belhumeur et al(2016) The method for projecting a face image into a three-dominated linear subspace was employed by the fisher face. The projection is based on a Linear Diiscriminant fisherman to maximise class dispersion and reduce the class dispersion to a minimum. In particular, the variable illumination approach was shown to be more efficient than a face-to - face approach. The test was performed on 150 sides with 15 objects in the ORL database. The results show that the face in itself is sturdy, but sensitive to light and pose when it comes to glasses and facial expression. The right rate of recognition is 95%. We use LDA extraction in our experiment with NN supervised and unattended learning regulations. It shows a rise in the detection rate, but with back propagation it is good with SVM.

Peijiang liu et al. (2013), Proposed a new 3D face form, a major step in the mining and acknowledgment of facial characteristics. The input of this study is the unstructured point cloud, which determines the widespread application of the representation proposed. The contribution comprises the spherical depth map and facial alignment based on the SDM. In many applications, the spherical deep-map is used as a special 33-type image that uses previous human face anatomy. It makes the face more efficient to align. The results of different methods are not comparable because the works present in the literature are performed on different sizes of databases of a different nature under different conditions. One technique, for example, was tested in the front picture with a high rate of recognition while the other was tested on broken faces with a low fault rate. Some researchers have combined techniques that are more successful than a separate method. The error rate and the computational costs are important as well as the recognition rate. If the error rate decreased substantially while the recognition rate slightly increased, the method of combination is still preferable. When computing costs increased considerably, it has much to do with practise. The key element of this review is studying, testing and comparing the results of different methods.

OBJECTIVES OF THE STUDY

1. To Study on Develop a 3D face recognition algorithm from blurred 3D faces using an artificial neural network.

2. To examine In addition, our combination of texture turbulence and texture-growing techniques provides an attractive way to synthesise and tile natural 2D texture patterns.

RESEARCH METHODOLOGY

SIMULATING 3D TEXTURE BY TEXTURE GROWING AND TEXTURE TURBULENCE

Texture turbulence

The image disturbance technique can be used to warp texture. The first attempt is to produce each frame by giving the previous frame a corresponding disturbance. The problem can be converted to a time or dynamic textures synthesization by considering the "z" dimension as a time dimension and each new image generated as a frame. We treat variables z and t equally for convenience. For the creation of a visible feeling of the movement texture the space-time self-regressive model (STAR), which is widely employed in time texture and flow visualisation. If I(x, y, t) is the intensity of pixels (x , y) at the moment t the STAR model is the same.

$$I(x, y, t) = \sum_{i=1}^{p} a_i I(x + \delta x_i, y + \delta y_i, t + \delta t_i) + a(x, y, t) ,$$

When the signal I(x, y, t) is modelled as a linear combination of several of its own lagged values plus a Gaussian white noise process a(x, y, t) and is worse than a number of frames of a calculation reference (in general, p=1). A (or partial) line order is imposed so that I(x,y,t) depends on the previous frames' pixels only. This model can be considered as the extension, by adding a time dimension, of the causal random Gaussian Markov model used in texture modelling.

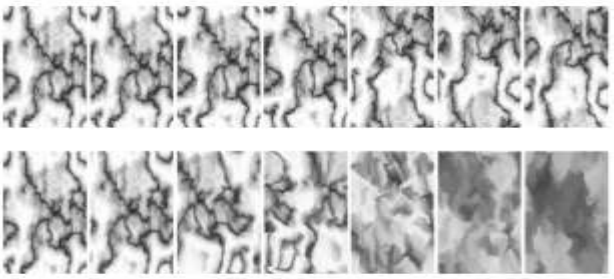


Fig. 3. Comparison of the approaches based on texture turbulence and the STAR model. Upper: texture frames generated by texture turbulence (from left to right: frame 1, 3, 6, 10, 30, 60, 100); Lower: texture frames generated by the STAR model (from left to right: frame 1, 3, 6, 10, 30, 60, 100

Texture growing

While the texture turbulence gives the original pattern a certain degree of creativity, the turbulence is limited only to local regions, as the central position of each frame in a z-direction remains unchanged. This is not the way many natural materials are grown. Although several procedural approaches exist to simulate the development of natural scenes such as the particulate system and the L-system, they do all suffer and do not satisfy these requirements with similar inconveniences to the STAR model. We define a unique texture path for the target 3D texture cube to simulate a naturally developing process. A texture path is generated along the Z direction for each pixel on the particular texture pattern to denote the path along which the pixel travels on the front of the 3D cube from its point of departure to its end on the back. The same path applies to all pixels so that the global texture movement remains consistent. The texture path should be a zaxis-defined 2D vector function that meets the following equation.

$$V(z_i) = V(z_0) + F(z_i)\ ,$$

Where V(zi)is a 2D position vector that indicates the pixel position in the zdirection on the 3-D texture cube, and F(z) is a 2D motion vector that describes the pixel motion through all 3-D cube frames, defined along the Zaxis.

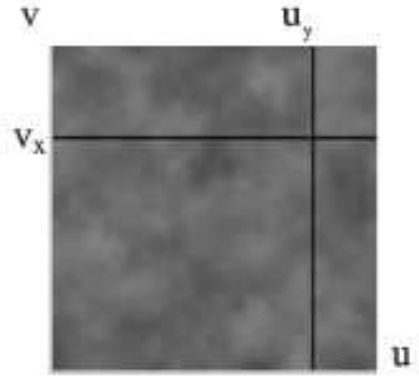


Fig. 4. Illustration of the turbulence-based texture pathfunction

DATA ANALYSIS

Though the algorithm of texture development is covered in the Sect. The concept of texture turbulence and texture growth was originally designed for a point-return-system, and is well-suited for many other applications related to texture. We give several experimental findings in this section. On a single PC workstation, equipped with the 1.6 GHz Pen-tium 4 processor, 256 MB RAM and NVIDIAGeForce4 display adapter, all experiments were carried out via C++.

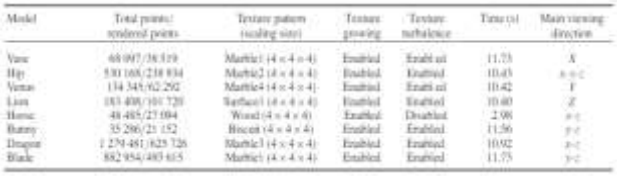
Texturing point-rendering system

In order to test the effectiveness of our solid texture method, we borrowed the QSplat system designed by Rusin-Kiewicz et al., because it is a point-rendering system for complex subjects which contain up to 100 million point samples. The 3D texture cube is made in a pre-production routine and it takes insignificant time to map the texture during rendering. Therefore, the good interactive rendering performance of QSplat remains unchanged. In our experiments certain 2D texture patterns show the results of our application to several complex models. A cube of texture 128 bis128 bis128 3D is synthesised in each case by a certain pattern of 128 bis128 texture. This routine takes only approximately ten seconds. Then the severe 3D cubes are tiled in order to create a bigger space in 3D texture. Due to the little computer requirement of our rendering approach, all models can be interactively navigated with our restricted hardware in 512 versus 512 windows with a frame rate of approximately 8 frames per second.

Table 1 shows a number of representative results. It demonstrates that the algorithm is fast enough to execute. The 128 bis128 bis128 3D texture cube may be calculated in approximately 10 seconds using our unoptimized programme

and hardware if the options are enabled both for texture-growing and texture-turbulence. Two-thirds of the total time for computing textures, texture turbulence takes up the remainder. The 5-007 value in the Eq. 3 can accelerate the algorithm by a few seconds or slow it down. After producing the cube, the point rendering system can be tiled and decorated in an item-independent way. Comparable with conventional solid texture techniques, tiling saves considerably on storage life, and object-independent texture saves computation considerably if different objects are lost. While the final texture is built by tiling several ex-actly the same copies, the complexity of the scene is successfully masked and becomes imperceptible in practise. The approach is consequently very efficient as compared with the previous solid texture method in terms of time and space.

Tabl e 1. Details of the models and the texture patterns



CONCLUSION

We have introduced a method to create vibrant 3D textures based on the principles of texture turbulence and growth. The resulting texture development algorithm is able to produce 3D textures from 2D texture patterns, modell-independent and discontinuity-free. This property allows the decoration of complex objects scanned from range. The method also gives the 2D texture and motion texture synthesis a promising solution. For future work there are a number of guidelines. One option is to design stronger texture evo-lution techniques to simulate the development of natural materials more realistically. For example, we can achieve more flexible texture growth through morphogenesis and developmental processes. Another option would be the combination, in order to produce more impressive simulations of nature, of the idea of texture growth and texture turbulence together with other dynamic texture and flow visualisation techniques. The relationship between our 3D texturing method and other stochastic methods lies in a more attractive direction for future work. It's very interesting to note the different roles of pseudo-white noise playing in fields like solid texture, visualisation of flow and modelling of the natural scene. Can we find a theoretical framework that uniformly and harmoniously encompasses all these approaches? This is a very interesting matter and is undoubtedly worth investigating.

REFERENCES

1. Bowyer et al. (2012) Synthesizing natural textures. 2001 ACM Symposium on Interactive 3D Graphics, pp 217–226

2. Peijiang liu et al. (2013), Octree textures. In Proceed-ings of ACM SIGGRAPH 2002, ACM Press /ACM SI G-GRAPH, Computer Graphics Proceedings, Annual Confer-ence Series, pp 101–106

3. Belhumeur et al(2016) Multiresolution sampling procedure for analysis and synthesis of texture images. In Whitted T (ed) SIGGRAPH 1997 Conference Proceedings, Annual Con-ference Series, ACM SIGGRAPH, pp 361–368

4. Turk et al (2013) Imaging vector fields usingline integral convolution. In: Proceedings of ACM SIG-GRAPH 93, Computer Graphics Proceedings, Annual Con-ference Series, pp 263–272

5. Victor-Emil et al (2016) Painting andrendering textures on unparameterized models. Proceedings of SIGGRAPH 02, pp 763–768

6. Hyun chul choi et.al.(2015) Anisotropicsolid texture synthesis using orthogonal 2D views. Comput Graph Forum 17(3): pp. 87–95

7. Sheenstra et al.(2013) A survey of 3D texturing. Comput Graph 25(1): pp. 135–151

8. Blanz et al (2015) Texturing and modelling: a procedural approach. Morgan Kaufmann, San Diego, CA, USA

9. Peijiang liu et al. (2011), Texture synthesis by non-parametric sampling. In: International Conference on Com-puter Vision, vol 2, pp 1033–1038

10. Hutton et al. (2016)Image quilting for textured synthesis and transfer. In: Proceedings of ACM SIG-GRAPH 2001, pp 341–346

11. Bowyer et al. (2013)Point sample rendering. In: Rendering Techniques'98. Springer, Berlin Heidelberg New York, pp 181–192

12. Hutton et al. (2016) Pyramid-based texture analysis/synthesis. In: Cook R (ed) SIGGRAPH 95 Conference Proceedings, Annual Conference Series.

ACMSIGGRAPH, Addison-Wesley, pp 229–23

Corresponding Author

Dr. Bipin Sinha*

Assistant Professor of Physics, Barsi College Pandygangout Nawada, Magadh University