Sergei I. Vyatkin,
Sergei A. Romanyuk,
Alexander V. Melnik

TEXTURE MAPPING ON CURVILINEAR SURFACES

Abstract

The problem of displaying a two-dimensional array of texture on arbitrarily oriented plane in the three-dimensional space is considered. The method of texture mapping on curvilinear surfaces is proposed.

Additional Keywords: free forms, texture mapping, curvilinear surfaces.

Introduction

Texture mapping on the surface is effective method of raising realism in computer graphics. Using the textures allows, in the first place, prototype a color drawing on surfaces, as well as transparency, sharp borders, moving objects and many other effects. Texture vastly raises a visual difficulty of expressing under comparatively small amount of calculations.

Term "texture" is defined in the computer graphics as a two-dimensional scene, displayed on the three-dimensional plane or three-dimensional curvilinear surface, or three-dimensional scene displayed on the three-dimensional space.

When displaying a texture happens to decide two problems: geometric image of array of texture on the surface and filtration of expressing, which avoids on expressing step-like lines, moirés and etc - all that in the literature on the computer graphics is marked by the term Aliasing. Aliasing can result when a signal has unreproducible high frequencies.

Process of displaying the texture maps on flat surfaces includes two stages. First is a perspective transformation, calculation of texture coordinates (u,v), corresponding coordinates (Xs,Ys) of pixel on the screen. Plane parameterized, if assign on her single texture vector U and V and origin coordinates. The task consists in that to, knowing coordinates of screen (Xs,Ys), get texture coordinate U and V corresponding spots of plane in the coordinate system of viewer.

Transformation can be written in the matrix type [1]:

$$\begin{bmatrix} UW, & VW, & W \end{bmatrix} = \begin{bmatrix} Xs, & Ys, & 1 \end{bmatrix} \begin{vmatrix} Au & Av & Az \\ Bu & Bv & Bz \\ Cu & Cv & Cz \end{vmatrix}$$

We compute texture coordinates from screen coordinates as follows:

$$u = \frac{Au * Xs + Bu * Ys + Cu}{Az * Xs + Bz * Ys + Cz},$$

$$v = \frac{Av * Xs + Bv * Ys + Cv}{Az * Xs + Bz * Ys + Cz},$$

where (Xs,Ys) are a screen coordinates a pixel; Ai, Bi, Ci are an elements of matrix of transformation of coordinates.

The second stage of displaying a texture is a filtration required for preventing aliasing. They the most are often used so named by MIP-map (pyramidal) texture map, the most suiting for hardware realization [2].

Way to preliminary filtrations get a kit of square texture maps with the different permit for each object. Each texture to the map put in the correspondence fixed point value so named LOD-level of detail.

Depending on distances before faces and its orientation are chosen for functioning two texture maps with nearby level of detail.

Criterion of choice is a linear size to projections a pixel on the verge. If projection a pixel covers less two texels (element of texture map), is observed aliasing. Beside verges, with angles of slopping (between planes faces and screen), close direct, texture drawing powerfully blurs. This effect carries a name blur [1].

Then in accordance with texture coordinates from each texture map are read on four texels. Trilinear interpolation is a process of filtration terminates on these eight values. Factors of tri-linear interpolation are fractional parts of texture coordinates and LOD. By mutating a process, filtrations get not-which

special effects. For instance, possible get texture drawing with the constant width a contour.

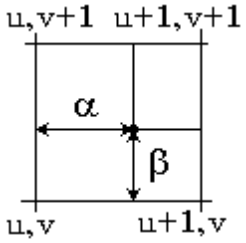

Fig. 1 Coefficients of bilinear interpolation

Calculation of level of detail

Consider a choice of LOD of texture map. In general event a projection "round pixel» on the plane be an ellipse [1, 3], which greater axis will take as a diameter ∅ filter. Size of ellipse depends on:

Interval of sample δ =max (δx, δy);

Distances from the watcher before the cross point of ray of vision with the plane P (D);

Angle of slopping of plane to the ray of vision.

Big axis of ellipse rests upon lines of crossing two planes. One of them - texturing plane, but other - such that in she lies normal to planes and ray of vision. It follows that ∅ = 2*δ*D/ cosθ, where θ - is an angle between normal to planes and ray of vision, cosθ = d/D, where d is a distance from the origin of coordinate system before the plane. D is a distance from the watcher before the cross point of ray of vision with the surface of object. So ∅ = 2δD2/d ≈ 2δZ2/d.

Trilinear Interpolation

For getting a final result, i. e. color and transparency in each pixel is required to produce trilinear interpolation [2]:

$$result = F\_level + \gamma * (C\_level - F\_level),$$

$$F\_level = F\_bottom + f\beta * (F\_top - F\_bottom),$$

$$F\_bottom = f(fu, fv) + f\alpha * (f(fu+1, fv) - f(fu, fv)),$$

$$F\_top = f(fu, fv+1) + f\alpha * (f(fu+1, fv+1) - f(fu, fv+1)),$$

$$C\_level = C\_bottom + c\beta * (C\_top - C\_bottom),$$

$$C\_bottom = c(cu, cv) + c\alpha * (c(cu + 1, cv) - c(cu, cv)),$$

$$C\_top = c(cu, cv + 1) + c\alpha * (c(cu + 1, cv + 1) - c(cu, cv + 1)),$$

where

fa, fb is a fractional part of texture coordinates of best level of detail;

ca, cb is a fractional part of texture coordinates of worst level of detail;

c[cu,cv], f[fu,fv] are a values color component in the spot [u,v] for each of level of detail.

Raw data is an 8-bit value a component of color (transparencies). Coefficients of bilinear interpolation a and b for each of chosen level of detail are got from U and V addresses. Coefficient linear interpolation between different levels of detail g is chosen depending on the most diameters of projections a pixel (ellipse) on texture plane.

Contour Texture

Contour Texture must ensure a possibility of creation and displaying the sidebars of the free form within the framework of texture map, not blur when approaching. Application of such texture - an image of letters, symbols, borders of section and etc.

Enumerate main characteristics of contour texture map are:

- width of sidebar forms 1-2 pixels;

- when drawing near (real value LOD is a negative)

- sidebar does not blur, width of sidebar is saved;

- in the field of with positive values LOD contour texture move over to usual, RGBT - textures.

- a form of the sidebar sufficiently free, i.e. for the task of sidebars is required more one bit on texel.

For processing the contour texture maps required transparency channel. This allows getting full color expressing of inwardly sidebars. However, for getting a full-fledged picture will take a double lay texturing verges: one - for expressing inwardly sidebar, second - outside of its.

After bilinear interpolation, such, either as in rest channels, for the separation of sidebar are produced following calculations:

$T = \frac{t - 0.5}{Ps} + 0.5$, hereinafter follows a correction:

T= 0, if T<0; T, if 0 <T<1; 1, if T>1

, where

T is the final result,

t is the result of bilinear interpolation,

Ps is the size of projections a pixel.

Required addition operation:

$T = (t - 0.5) << Log_2 Ps + \gamma' * (t - 0.5) << Log_2 Ps + 0.5$,

where γ' is a fractional part of the size of projections a pixel.

Texture Address

For the calculation of texture address necessary texture coordinate, level of detail of pixel, value of general order of calculations from texture coordinate evaluator and offset of local and global texture coordinates a friend for the friend.

Hereinafter each of texture coordinates is divided into four numbers (Fig. 2).

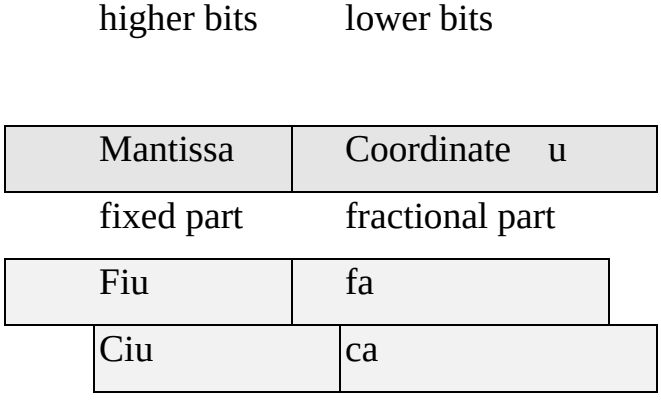

Fig. 2 Coefficients of bilinear interpolation.

fu = ((fiu - du) & 0111) + (du & 011),

cu = ((ciu-du/2) & 011) + ((du/2) & 011),

where (fiu, fiv) and (ciu, civ) are the texture coordinate of current pixel, scaled in accordance with the level of detail. du (dv) is an offset of net of local texture coordinates comparatively global, makes sense only for compressed texture maps; are a zero for non-compressed textures.

For fv, cv calculations are similar. Texture address is formed from values fu, fv, cu, cv as follows (Fig. 3).

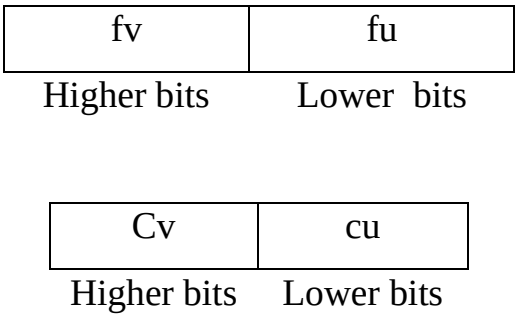

Fig. 3 Forming a texture address.

Texture mapping on free forms and volumes

A paper [4] shows that texture mapping onto planar quadric and super quadric surfaces, and bicubic and biquadrate image warps are two-pass transformable. The article [5] presents an extension to texture mapping for representing three-dimensional surface details.

Give a determination of texture in "three-dimensional" sense, coming from its initial determination in "flat" variant, which we have described above. Besides, will uniquely define main notions:

Texture – detailed elaboration of surfaces or volume of displayed object. Here greatly that is not specified concrete construction of texture, the texture in general event not without fall presents itself two-dimensional an array of texels.

Displayed object is the whole three-dimensional scene or its element, carrying information on its form [6].

For instance, scene, presented by two ellipsoids, on the one hand can be considered as a scene of two independent objects, on the other hand, these ellipsoids possible to consider as an object-association of two ellipsoids, and these dieresis of vision will not contradict each other, but will be alternative.

Then, in terms of objects, texture -is an element a scene, changing characteristics of other object and itself not displaying under rasterization. For instance, painted ball will be presented by two objects: form in the manner of the ellipsoid and object by the texture, containing reference to the array of colors, some determined by the image superimposed on the ellipsoid, using linear, cylindrical, and spherical or some other parameterization.

Characteristic of displayed object - any parameter, nominated this object, which is used at the calculation of resulting color a pixel on the screen. For instance, this can be a color of object, factors of diffuse and reflecting forming models of Phong illuminating, transparency and etc. Texture, thereby, can influence upon any parameter of displayed object.

Except objects, given analytically, - quadrics exists a possibility to display objects, presenting itself scalar arrays given [7]. This vastly increases possibilities and system applications in following events:

Visualization of three-dimensional arrays, tinned as a result scans of real objects in the real space (tomography and etc.).

Visualization of scientific data, tinned as a result calculations or measurements.

Visualization of meteorological data (sharing a temperature, moisture, and etc.).

Visualization of two-dimensional net of heights without preliminary triangulations.

Visualization of scalar arrays though and requires significant expenses of memory (order N,2 N3 for two-dimensional and three-dimensional arrays), but in contrast with equivalent their presentation in the manner of ensembles of triangles, arrays all more compact and suitable in use and manipulating.

Let there is an object, representing itself some scalar array in some format, known only the most object.

Define this object as an object-array. In total, in the system main primitives will be presented by two classes of objects. One class is analytically described primitives (quadrics), other - arrays of scalar data. And that and other classes have more compact description in contrast with triangles.

Object-array can be marketed several ways. It can conclude in itself univariate, two-dimensional or three-dimensional array, as well as can be a net of heights [6], [7].

On given in this work determination, texture – is an object (not displayed), changing characteristics of other object. Particularity of texture is that texture can be any object. For this was offered small changing a data processing scheme and buildings of tree a scene not touching of the most algorithm of rasterization [8].

On the most current level of recursion of a faking, rasterizer finds a nearest visible voxel and requires beside displayed object (scene) parameters of this space element for their use in the calculation of color a pixel.

Accompaniment in the structure given at the account of texture is that each object can contain a reference to other object, being texture for first. If this references no, the object on the request for the value of their own characteristics gives a value, determined for this object by default.

Otherwise, parameters of object in current voxel are subjected change on the part of object-textures, to which is exiled our object.

Important at the imposition of texture on the object is a transformation of coordinates from texture space (U, V, W) in the space of object (X, Y, Z). In our event parameterizing needed for objects-arrays only. To have a possibility to superimpose a texture on the surface and volume of displayed object, objects-arrays are complemented by the characteristic of transformation of coordinates from the model space M in texture space T.

At the request beside the object-array of value of some characteristic, it will convert coordinates current voxel from M in T: (X, Y, Z)→(U, V, W) and uses these coordinates as an address in the array. If, for instance, array two-dimensional, one coordinate is not used.

Marketed three types of transformation of object coordinates in texture coordinates:

Linear:

$U=(1 + X)/2,$

$V=(1 + Y)/2,$

$W=(1 + Z)/2,$

Transformation from binary cube into unit.

Cylindrical:

$U=\text{Arctan}(Y/X)/2\pi,$

$V=(1 + Y)/2,$

$W=2\sqrt{(X\bullet X + Y\bullet Y)}$

Spherical:

$U=\text{Arctan}(Y/X)/2\pi,$

$V=\text{Arctan}(2\sqrt{(X\bullet X + Y\bullet Y)}/Z)/2\pi,$

$W=2\sqrt{(X\bullet X + Y\bullet Y + Z\bullet Z)}$

Conclusion

Ways of displaying a texture, stated in the article, were marketed software. The proposed method of texture mapping on surfaces of free forms offers a wide scope of application.

References

1. Paul S. Heckbert, Survey of Texture Mapping, IEEE Computer Graphics and Applications, Pages 56-67, November, 1986.

2. Lance Williams, Pyramidal Parametric, Computer Graphics, pages 1-11, July 1983.

3. A.M. Kovalev, Yu.V. Tarasov, Texture on arbitrarily oriented flat surfaces/ Aerometry, №6, 1986.

4. Smith, Alvy Ray. Planar 2-Pass Texture Mapping and Warping. Proc. SIGGRAPH 87 (Anaheim, CA, July 27-31, 1987), pp.263-272.

5. Manuel M. Oliveira, Gary Bishop, David McAllister. Relief Texture Mapping. Proc. SIGGRAPH 2000 (New Orleans, Louisiana, July 23-28, 2000).

6. Sergei I. Vyatkin, Boris S. Dolgovesov, Valerie V. Ovechkin, et al. “Photorealistic imaging of digital terrains, free forms and thematic textures in real-time visualization system Voxel-Volumes”, GraphiCon ’97, Moscow.

7. S.I. Vyatkin, B.S Dolgovesov, A.V. Yesin, et al. Voxel Volumes volume-oriented visualization system, International Conference on Shape Modeling and Applications (March 1-4, 1999, Aizu-Wakamatsu, Japan) IEEE Computer Society, Los Alamitos, California, 1999, P. 234.

8. S.I. Vyatkin, B.S. Dolgovesov, S.E. Chizhick, “Synthesis of virtual environment with object-space recursive subdivision” - Graphicon’98 Proceedings, S. Klimenko et al. (Eds). 1998.