Animated Texels

Fabrice Neyret

INRIA, Syntim Project
B.P. 105, 78153 Le Chesnay Cedex, France
Fabrice.Neyret@inria.fr
http://www-rocq.inria.fr/syntim/research/neyret

Abstract.

Volumetric texturing is a method dedicated to modeling complex repetitive geometries, such as grass, fur or foliage, by storing a volumetric sample to be mapped on a surface. This representation is effective in a ray-tracing environment, giving images with low aliasing at low cost. We show here that it can be extended to get an easy way of animating complex repetitive geometries, like the effects produced by the wind in a wheat field, or fur motions.

1 Introduction

Complex repetitive geometries such as grass, hair, foliage, fur, forest and so on are a major component of the natural world, very important for the realism of synthetic images. Many representations exist to model such objects, such as particle systems [10, 11], L-systems [9], growing models [3], hypertextures [7], volumetric textures [4, 5].

Animating these objects is also a very important point for realism, and is also hard to model with classical geometric representations. One has to figure out the effects of the wind in a wheat field, or in a foliage, the motion of a moving animal's fur, etc. Representations dedicated to complex geometry have been generally studied for static scenes, excepted particle systems that are re-generated at each frame. But particle systems use an ad-hoc description based on thin trajectories, and an ad-hoc lightning incompatible with ray-tracing. Ray-tracing is however necessary for the realism, as shadows are important for the appearance of complex objects. Furthermore, ray-tracing can handle most of the other object representations.

In this article, we present a method for easily modeling animated complex repetitive geometries, in a ray-tracing context. The purpose is to handle a wide class of shapes, and to control the animation at a global scale. This method is based on volumetric texture representation (see section 3) and space animated deformations (see section 2). It is detailed in section 4.

2 Animating Complex Geometry

Two dedicated kinds of approach have been previously used to give realistic animation in the scope of the 'natural look': physical or pseudo-physical models (e.g. waves on the ocean), and particles systems (e.g. waterfalls). The results are impressive, but the methods cannot be reused so well: both are monolithic, as there is no separation between the model and its animation (and even its rendering, for particles systems). It is thus difficult to introduce and animate a given usual shape in this way. Moreover, both methods take the control of the 'simulation'. As a result, it is difficult for the user to specify precisely and interactively the motions.

Another point concerns the integration of a complex object in a whole scene: a rendering algorithm can produce an integrated image of the scene if it can handle the various kinds of objects that are present. The dedicated rendering used for particles systems prevents from integrating them in a classical 3D scene (this is generally done by image composition), thus forbidding interaction like shadows.

To model a wide class of animated complex geometries in an integrated context, one has thus to keep the generality and the interactivity of the usual tools and representations, and to make the model usable by a classical rendering algorithm (e.g. ray-tracing).

Many methods can be used to animate usual objects. Space deformation approaches [2, 1] have interesting properties in the scope of complex objects, especially if they deal with bounding volumes:

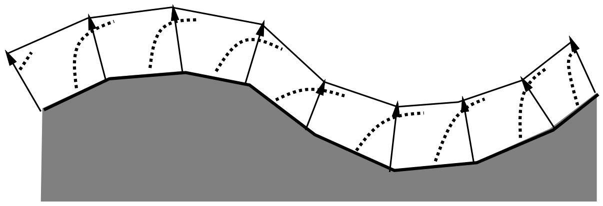
- no information is needed about the object structure (e.g. skeleton) so that the surface description is sufficient,
- collisions are considerably more easy to deal with as one has only to be aware of collisions between bounding volumes,
- self-collisions are avoided if the resulting space is not self-intersecting,
- animation specifications are easy to describe interactively, with few parameters.

The other animation approaches are not adapted so well: an explicit description of the motion of each part of the objects would be complex to specify for the user, physical models and particle systems are hard to control interactively and use ad-hoc representations dedicated to animation, using articulated models needs too much degrees of freedom in the scope of complex geometry.

Therefore, space deformation methods seem to be a good approach to animate easily complex repetitive geometry. But usually such methods operate on classical geometric data (e.g. facets, patches), which are seldom used to model complex geometry and may be costly for a large database, where a great amount of useless facets are transformed in any case (moreover, a complex object can have more facets than visible pixels). We adapt the approach in order to deal with a more efficient representation than classical geometry.

3 Modeling Complex Geometry

We propose to use the volumetric textures representation instead of classical geometry. Volumetric textures can represent geometry with every 3D effects and have interesting properties to model complex repetitive objects concerning efficiency and aliasing as described in [5], and work in a ray-tracing context. Volumetric texturing, first introduced by Kajiya et al [4], is based on the mapping of a volumetric 3D sample called *reference volume* over a surface (e.g. a lawn covering a hill, or fur on an animal). This allows to separate the specification of the local aspect of the geometry (e.g. weed blades) and the specification of its large scale shape (e.g. hill): one needs not to explicitly build each detail of the complex object everywhere it lays.



In our representation [5], the reference volume is encoded by an octree of voxels, in a multiscaling purpose. Each voxel contains a density of occupation, and a simplified reflectance model which simulates the piece of surface that is supposed to occupy this area of space. This model represents the normal distribution of the piece of surface (that otherwise would have been represented by several small facets), and can be filtered like density in order to obtain the rougher resolutions of the octree. This allows a correct and efficient multiscale representation of the object encoded in the reference volume, which can be rendered with few aliasing using only one ray per pixel (other level of detail approaches like facets decimation do not preserve the reflectance, e.g. a corrugated iron sheet will be filtered into a flat object). Usual representations with facets need quite more information than what can be really seen (e.g. leaves on the trees in a forest), which implies extra cost concerning ray intersection, storage, aliasing and modelization work.

The repeated and deformed copies of the reference volume mapped over the underlying surface are called *texels*, which are thus a space deformation of the reference volume. Unlike usual deformation methods, no transformations of the objects are effectively computed (which may be very costly for complex geometry): the deformation is formal, being achieved at rendering time by converting the rays crossing the deformed area into the reference volume, thus directly using the objet description without having to modify it. This approach saves computation as a complex object can have more details than visible pixels. It also allows the use of representations that cannot be explicitly space-deformed (e.g. implicit functions, CSG) since no transformations are effectively computed.

This also simplifies the building of a scene without intersections: instead of dealing with collisions between all the represented objects (e.g. weed blades on a hill), the user has just to build a correct sample. Then the texel mapping imposes the deformations so that the texels lay upon a surface and stick to one another without overlapping.

4 Animating Texels

As explained in the previous section our complex geometry model is volumetric textures, defined by the mapping with space-deformations of a reference volume onto a surface. Animation of the model is done on the mapping parameters, mainly by controlling the vertical edges of the texels (initially normal to the underlying surface), which determines the texels' deformation: pressure on the four vertical edges of a texel defines the deformation of the box bounding the texel, which causes the deformation of its content as for FFD [12].

- For a deformable surface like a cloth object, the texels' 'thick skin' covering it naturally follows its deformation, as a continuous material. We study this case in subsection 4.1.
- Otherwise, on a rigid surface, explicit motions can be generated by a force field acting on the vertical edges, thus deforming the texels. We deal with this in subsection 4.2.
- Another animation way can be used with texels: in a cartoon-like approach, successive steps of a simple motion (like oscillations of leaves in foliage) can be sampled in few separated volumes, successively used along time. We consider this case in subsection 4.3.

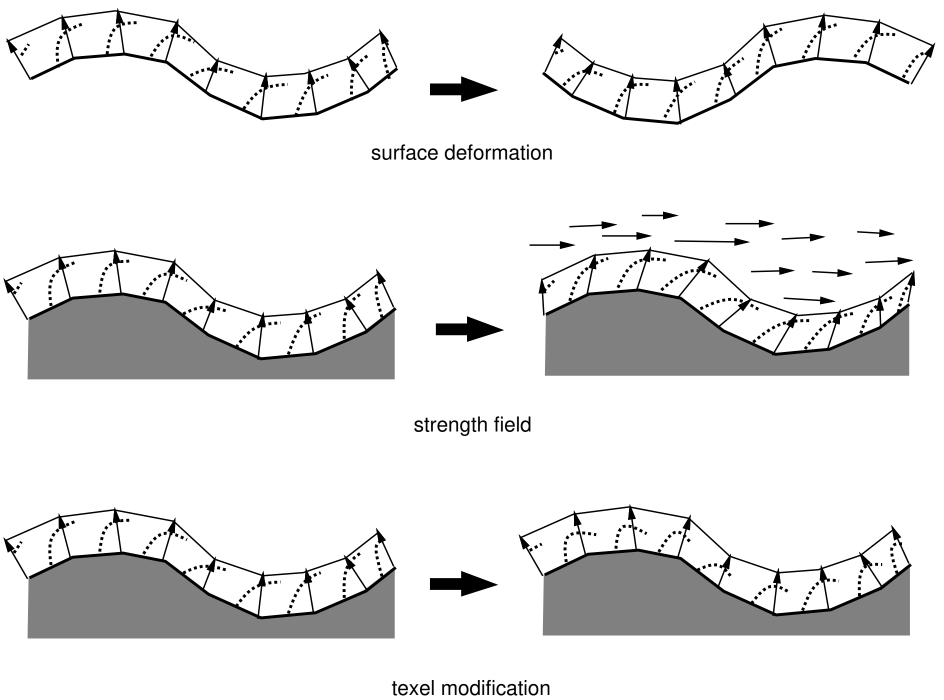


Figure 1: The three ways of animating texels.

For a realistic animation, the three methods can be blended to take into account the different scales of motion: the fur on the skin of a running animal follows the deformation of the animated body, it makes waves according to acceleration and inertia, and maybe the pils locally oscillate a bit inside the texels. Similarly for a tree in the wind, the surface of each main branch associated to its bough is geometrically deformed, foliage waves with the wind, and leaves locally oscillate inside the texels.

4.1 Texels on an Animated Surface

In usual implementations, each texel sticks to a surface element which is a bilinear patch. The four vertical edges of the texel follow a vector stored at the four vertices, that can be equal to the normal or 'combed' in a given direction. A texel is thus a trilinear deformation of the cubic reference volume, and texels are naturally sticking to each others. (More complicated deformations may be used, but one has to be aware of the ray intersection cost while rendering.)

Texel animation is simply achieved by taking into account the new surface and normals at each time step. One has just to take care of the maximum curvature of the surface, which can bring to degenerated deformations when the curvature radius is near or less than the thickness of texels.

In figure 4 (see Appendix), a volumetric pattern representing a piece of scaffolding has been mapped onto a flag animated by a non-linear mass-spring model[1]. The texels thicken the flag, and follow continuously its deformations.

4.2 Texels Animated by a Force Field

The deformation of the texels can be explicitly controlled by the motion of the vertical edges: this motion can be generated by a force field (e.g. Laplace field [15], stochastic flow [13]) acting on the vertical edges, or by a dynamic scheme (e.g. mass-spring, elasticity [14]) linking the top of the vertical edges.

In figure 3 (see Appendix), we present a lawn in the wind. A texel contains 16 weed blades. The motion is generated by an animated force field acting on the normals. This field $\vec{F}(M,t)$ models a gust of wind combined with a random jittering, that are encoded by two separated fields. Given the propagation direction $\vec{F}/||\vec{F}||$, The wind intensity at point M is obtained by the wave propagation expression $f(2\pi\frac{\vec{F}.\vec{M}}{\lambda} - \omega.t)$ (the origin is arbitrary).

For the pure wind component, $f()$ is a continuous periodic function for which the chosen pattern has a sudden attack and a slow falling (see figure 2). The jittering component is built in the orthogonal direction of the wind, using for the intensity a fractal solid noise function as $f()$ in the propagation expression. *Solid noise*, usually used for solid texturing [6], gives a signal which is at the same time continuous, derivable, and pseudo-random. The pseudo-frequency of the noise function can be controlled, and is used to define a fractal function called *turbulence*, more realistic for both texturing and random motion. The resulting lightning and shadow waves increase the realism of the motion.

[1]Thanks are due to Xavier Provot [8] for his flag model.

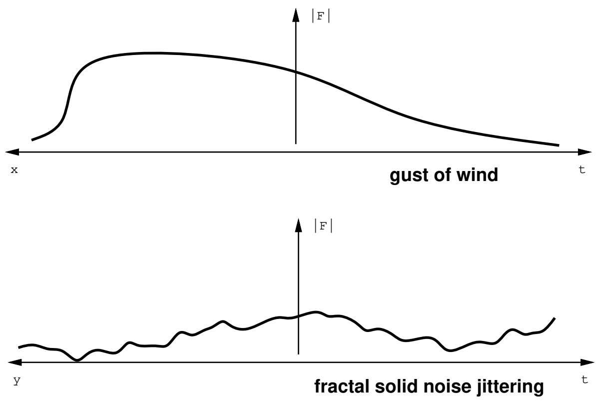


Figure 2: wind intensity in the x and y horizontal directions.

4.3 Animated Texels Content

Cartoon animation is based on switching sampled stages of a motion. This can also be used in 3D for simple or quick motions like oscillations of parts of the object encoded in the volumetric texture: some states are encoded in few separated volumes, successively used along the time (a single volume can also be modified after each step).

However, this increases the cost as additive storage or volume recomputation are needed. So it has to be used only for specific effects with very simple motions, like oscillations of leaves in foliage. This is also a way to break the regularity of the texel mapping: the phase of this loop animation can be randomly distributed along the surface.

5 Conclusion

Animated volumetric textures provide a tool adapted for animated complex repetitive geometries, which was not available before, and handle a wide class of repeated objects. This is done in a convenient way, at the scale adapted to the problem in the spirit of volumetric textures, and easily controlled in the spirit of space deformations.

The convenience is inherited from the different domains used to constitute the volumetric texture model:

- as a space deformation approach, animation can be easily interactively specified by the deformation of the bounding boxes, and intersection problems are limited to the intersection of the boxes.
- as a textural approach, the geometry encoded in texels sticks automatically to an underlying surface and follows its deformations.
- as a representation dedicated to complex repetitive geometry in a ray-tracing context, it can provide realistic results with low aliasing and low cost.

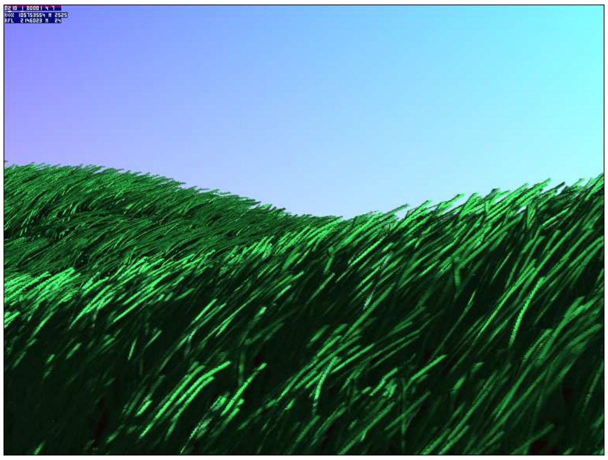


Figure 3: a lawn in the wind.

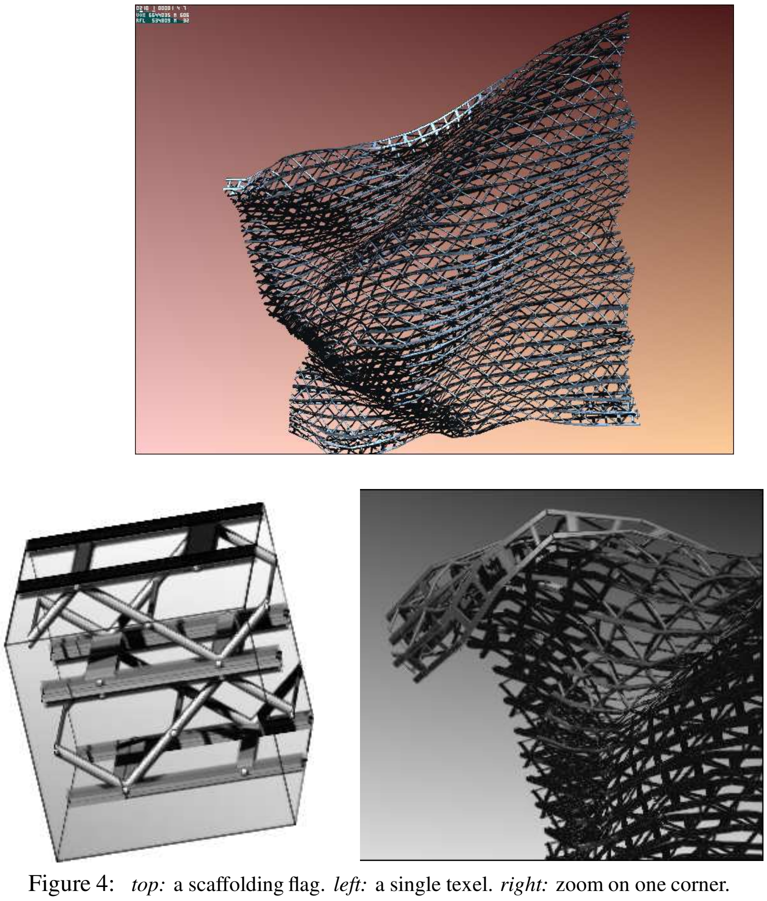


Figure 4: *top:* a scaffolding flag. *left:* a single texel. *right:* zoom on one corner.

References

1. D. Bechmann. Space deformation models survey. *Computer & Graphics*, 18(4):571–586, July–August 1994.
2. Sabine Coquillart and Pierre Jancène. Animated free-form deformation: An interactive animation technique. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 23–26, July 1991.
3. Phillippe de Reffye, Claude Edelin, Jean Françon, Marc Jaeger, and Claude Puech. Plant models faithful to botanical structure and development. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22(4), pages 151–158.
4. James T. Kajiya and Timothy L. Kay. Rendering fur with three dimensional textures. In Jeffrey Lane, editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23(3), pages 271–280, July 1989.
5. Fabrice Neyret. A general and multiscale method for volumetric textures. In *Graphics Interface'95 Proceedings*, May 1995.
6. Ken Perlin. An image synthesizer. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19(3), pages 287–296, July 1985.
7. Ken Perlin and Eric M. Hoffert. Hypertexture. In Jeffrey Lane, editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23(3), pages 253–262.
8. Xavier Provot. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Graphics Interface'95 Proceedings*, May 1995.
9. Przemyslaw Prusinkiewicz, Mark James, and Radomír Měch. Synthetic topiary. In Andrew Glassner, editor, *Computer Graphics (SIGGRAPH '94 Proceedings)*, pages 351–358, July 1994.
10. W. T. Reeves. Particle systems – a technique for modeling a class of fuzzy objects. *ACM Trans. Graphics*, 2:91–108, April 1983.
11. William T. Reeves and Ricki Blau. Approximate and probabilistic algorithms for shading and rendering structured particle systems. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19(3), pages 313–322, July 1985.
12. Thomas W. Sederberg and Scott R. Parry. Free-form deformation of solid geometric models. In David C. Evans and Russell J. Athay, editors, *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 151–160, August 1986.
13. Mikio Shinya and Alain Fournier. Stochastic motion - motion under the influence of wind. In A.C. Kilgour and L. Kjelldahl, editors, *Computer Graphics Forum (Eurographics '92)*, volume 11(3), pages 119–128, September 1992.
14. Demetri Terzopoulos and Kurt Fleischer. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22, pages 269–278, August 1988.
15. Jakub Wejchert and David Haumann. Animation aerodynamics. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 19–22, July 1991.