4th Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2007)
J. Dingliana, F. Ganovelli (Editors)

Occlusion-Based Snow Accumulation Simulation

David Foldes† and Bedřich Beneš‡

Department of Computer Graphics technology
The Envision Center for Data Perceptualization
Purdue University, USA

Abstract

We present a fast technique for the simulation of accumulated snow. Our technique is based on two phenomena; local occlusion of small holes and ditches, and the global influence of a skylight. We use ambient occlusion to predict the shape and location of snow accumulation and direct illumination from the skylight to simulate the melting and sublimation of snow as dissipation. The snow is simulated as a 3D layer that is added to the input scene. Our technique is a fast approximation and does not aim to be used for small and local features within a simulation. A scene with over 500k triangles can be calculated in about seven minutes on a standard computer and the major part of the calculation runs on the GPU. Results of our algorithm should be used for large distance views.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computational Geometry and Object Modeling]: Physically based modeling

1. Introduction

Snowfall is a common natural occurrence and should be incorporated into virtual scenes because it can play an important role in the visual aesthetics of a rendered scene. These factors make it a significant topic in computer graphics that has been addressed various times and from different points of view. Modeling techniques of snowfall have applications in screen media, games, simulations, terrain design, geographical information systems, etc.

Renderings of snowfall that employ the texture application of snow fail to properly visualize the geometry created by snow accumulation. Modeling fallen snow addresses this issue, however manual modeling of snow is time consuming, difficult, and scales with scene complexity. Methods that focus on simulation of falling snow and its accumulation produce good results for close-ups, but are not suitable for large-distance views. Techniques for approximate simulations of large-distance views would be useful in many applications, but we have not found any that provide efficient results at reasonable speeds. The procedural generation of snow geometry for large-distance views is therefore an important topic and it is a focus of our paper.

Figure 1: *Puget sound terrain (from the Large Geometric Model Archive of Georgia Institute of Technology) rendered from the North side with snow using the method presented in this paper*

We present a fast technique for simulation of accumulated snow. A scene with over 500k triangles in Figure 1 is calculated in less than seven minutes. Our algorithm is based on two phenomena; local occlusion of small holes and ditches, and global influence of sunlight (see an example in Figure 2).

† dfoldes@purdue.edu
‡ bbenes@purdue.edu

© The Eurographics Association 2007.

The Sun plays important roles in outdoor environments. It is responsible for the lighting of a scene, through both direct and indirect illumination. Direct illumination is when the Sun's rays reach a point without being occluded or reflected. Indirect illumination is when light has been reflected or refracted by other objects or particles before reaching its destination. These factors are important in generating the proper lighting of a scene, but should also be considered when simulating snow because they affect heated and cooled areas resulting in snow sublimation - the direct change from snow into vapor [PR95]. In addition to illumination, it is also responsible for heat in the form of solar energy. In the case of snow, this energy causes melting and sublimation. There are many studies on this effect. Even in climate temperatures that are below freezing, snow is still subject to melting and sublimation from direct sunlight. It is therefore important to consider solar energy when modeling snowfall and snow accumulation.

The main contributions of this paper are:

- a fast approximation of snow cover for distant views of large scenes,
- animations of snow changes (melting and accumulation),
- efficient usage of advanced GPUs (Vertex Buffer Objects, ARB Occlusion Extensions), and
- using irregular meshes allowing for concave features simulation.

2. Previous Work

One of the first methods that deals with simulation of accumulated material was introuced by Hsu and Wong [cHtW95]. Their work describes a simulation of dust particles using material properties such as stickiness, and surface geometry. Sloped surfaces determine the accumulation of dust particles by using a cosine function of the angle between the dust source and the surface normal of the slope. They also consider surface exposure to simulate dust removal from wind. Surface exposure is calculated by ray tracing local occlusion from the hemisphere above the surface. These parameters produce results that are good for detailed views of scenes.

Early methods of modeling snow accumulation involved the use of metaballs [NIDN97] to simulate snow density and snow covers. Snow is simulated on the surface as a layer and its optical properties are specified in order to display them correctly in the rendering phase. The results are suitable for close views of snow scenes but are hard to interpret in large or very large views.

Premože *et al.*, [PTS99] introduced a combined method that uses aerial photographs and physically-based methods to recreate snow and vegetation covered real-world scenes. Their method relies on removing shadows from source imagery to define the surface geometry. The snow is then described via physically-based snow melting and ablation and

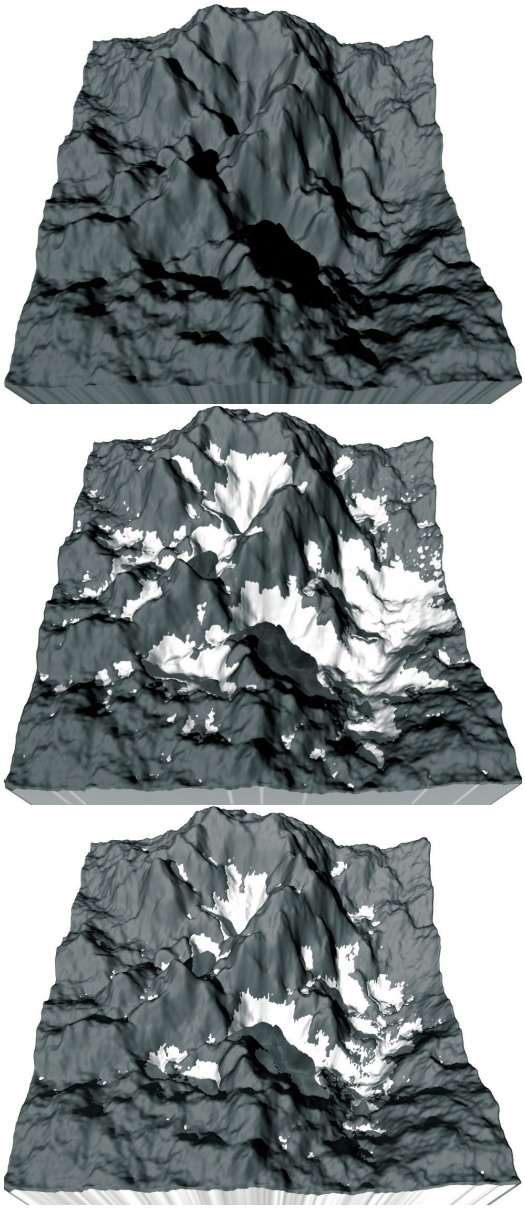

Figure 2: *Top: Model with no snow. Middle: Model with snow from only ambient occlusion step. Bottom: Model with snow from both ambient occlusion step as well as direct illumination step (Sun is on the left side)*

it is rendered by coloring the source image and draping it over the terrain as a texture. This can cause problems with rugged terrain.

A particle-based method by Muraoka and Chiba [MC00] used air fields to simulate snowfall and accounted for snow melting by heat propagation and sunlight by storing temperature values in a voxel space. Ground heat and its propaga-

© The Eurographics Association 2007.

tion was taken into account for simulation of snow melting. This method provides a time-consuming but very detailed simulation of snowfall in small scenes.

Fearing proposed a particle based simulation of snowfall in [Fea00]. His method is a two phase method. In the first step particles are used to determine accumulation of snow on the surfaces of objects in the modeled scene; and a local occlusion, similar to Hsu and Wong [cHtW95], to simulate obstruction of snowfall. The second phase simulates micro-avalanches in order to account for unstable and sloping surfaces. The results show properly modeled accumulation around steep surfaces and edges. This method is good for small scenes where obstruction of snowfall is common. It is suitable for close views of a scene, but a simulation of large-distance views would be expensive.

Shadow buffer techniques have recently been used to generate real-time snowfall [Tok06, OS04]. This method generates a shadow map using the Z-buffer. Snow is accumulated in areas that are not shadowed to simulate obstruction of snowfall. Each point of the snow is then stabilized using the micro-avalanche method introduced by Fearing [Fea00]. This method provides good results for small size scenes.

This paper describes a procedural modeling method for snow accumulation. This is a two step method that includes both calculations for direct and indirect illumination. Our method does not consider snow falling and its transportation in the air. We assume that there is a constant layer of snow on the surface and we deal with its ablation, melting, and sublimation by heat provided by lighting. To do so we calculate ambient occlusion culling to account for small holes and valleys. This first step calculates the local influence. The second step of our method considers direct illumination. We again use occlusion techniques, but in this step we consider the new snow that was accumulated. Snow that is not occluded from the Sun by an object is exposed to direct sunlight and is dissipated. We define dissipation as both the effect of melting as well as sublimation, however we do not consider the water result from melting because it would add unnecessary complications that do not significantly account for large distance views. Previous works used occlusion techniques for the purpose of determining obstruction of snowfall, our method uses occlusion to account for heat and dissipation. This illumination based method does not rely on complex particle simulations for snowfall and melting such as those described in [Fea00, MC00]. Our technique is intended to quickly generate snow cover over large distances and timescales.

The paper continues with Section 3 that describes the simulation steps in detail. Section 4 shows some results of our experiments and the Section 5 concludes the paper and discusses some future work.

3. Snow Accumulation

The method presented in this paper is a two step process. Both steps exploit some kind of occlusion. The first step uses ambient occlusion to determine local occlusion accounting for the indirect illumination where the shape and location of the snow is generated. The second step uses shadowing from the direct illumination of snow to determine global occlusion. Snow melting is a slow process, so we use an average of sunlight from the sky for this step. This is calculated by averaging the sunlight over a period of one day based on the sun's trajectory. Retention of snow is done by this calculation to ascertain the final shape.

The basic data structure used in our simulation is an irregular mesh of vertices. This is the data structure commonly used in terrain modeling and many level of detail algorithms can be seamlessly used for their efficient displaying.

The following subsections explain both steps in detail.

3.1. Ambient Occlusion

Ambient occlusion is commonly used as a lighting technique to simulate a subtle global illumination effect of diffuse-diffuse light reflection. Its main advantage in recent algorithms allow for real-time calculation for small and medium size models [PF05]. In real lighting, light rays are scattered and reflected by all objects, including the atmosphere and other participating media. Since it is virtually impossible for any point to have the complete absence of light the ambient light is commonly used to accommodate for this and it is a simple substitution of multiple light scattering. Traditionally the ambient value used in a lighting equation is a constant value, used to slightly increase illumination of a point on a surface. Ambient occlusion is a better approximation of the multiple scattering because it calculates influence of nearby surfaces. Various techniques can be used to calculate ambient occlusion and the most commonly used is by raycasting from each point to the nearby scene. An efficient technique for ambient occlusion culling was recently implemented on the GPU [Lef03] and we exploit it in the first step as an approximation of indirect light at each point.

The first step records an ambient occlusion value at each vertex of the input mesh. Our process begins by positioning the camera at a target vertex, with the focus in the direction of the vertex normal. The mesh is then rendered from this viewpoint using the OpenGL extension `ARB_OCCLUSION_QUERY` [Len03, WDS06]. This query returns the number of fragments that were successfully rendered (i.e., not occluded). The ambient occlusion at this vertex is then simply one minus the number of fragments rendered divided by the total number of pixels. This process is performed for each vertex of the mesh that provides a number between zero and one for each vertex. A value of one represents a point not occluded by any surfaces, a value of zero represents a point fully occluded. This technique is

© The Eurographics Association 2007.

commonly used as a preprocessing for small meshes in order to account for a local occlusion of a vertex. We use the occlusion value for simulation of the snow accumulation.

The static scene is rendered as many times as the number of vertices that are included in the scene in order to calculate each vertex's occlusion. The total occlusion is stored as a luminance texture which describes the occlusion value at any given vertex on the input mesh. This case is a perfect candidate for a GPU implementation, where the scene is hosted on the GPU and is not repeatedly loaded from the main memory of the computer where the data structure that hosts the scene is a Vertex Buffer Object (VBO). The combination of VBOs and occlusion culling hardware implementations make the algorithm very fast (see Section 4).

Our simulation of snow distribution is based on the ambient occlusion value of each vertex of the mesh denoted by $a_{i,j}$, where i and j are the indices of the vertex. The amount of snow $s_{i,j}$ added at each vertex is :

$$s_{i,j} = s_{max}(\tau - a_{i,j})cos\theta, \tag{1}$$

where τ is a user-defined threshold value and s_{max} is the height of the snow layer that is constant for the entire surface. The value of s_{max} is determined from the user defined proportion of the size of the model. The slope of the vertex is determined by the angle θ that is given by the angle between the normal vector $\vec{n_{i,j}}$ and the direction to the zenith. It is used as a slope stability function like those described in [cHtW95, OS04].

The equation (1) specifies the amount of snow that is located on each vertex $a_{i,j}$. This means that vertices whose value is at or above the threshold add zero snow and values below the threshold add a proportion of the snow layer. If the ambient occlusion is greater than the threshold, the snow is discarded, and no snow is accumulated at this point (see Figure 3).

After the first step we have determined the basic shape and location of the snow (see the second image in Figure 2). The second step determines where snow will be retained after direct sunlight is accounted for.

3.2. Dissipation by Sun Illumination and Sun Trajectory Sampling

Sunlight has a significant impact on the shape and the amount of snow cover. Snow that is exposed to direct sunlight is melted and sublimated, or simply dissipated.

The second step of our simulation is the dissipation simulation step. This is intended to increase the realism of the modeled snow geometry by simulating snow sublimation. As a byproduct of this step, the snow layer stability is also increased. The stability of snow refers to the state of rest. Snow that accumulates on an unstable surface such as those with edges or steep inclines is prone to instability. As the weight

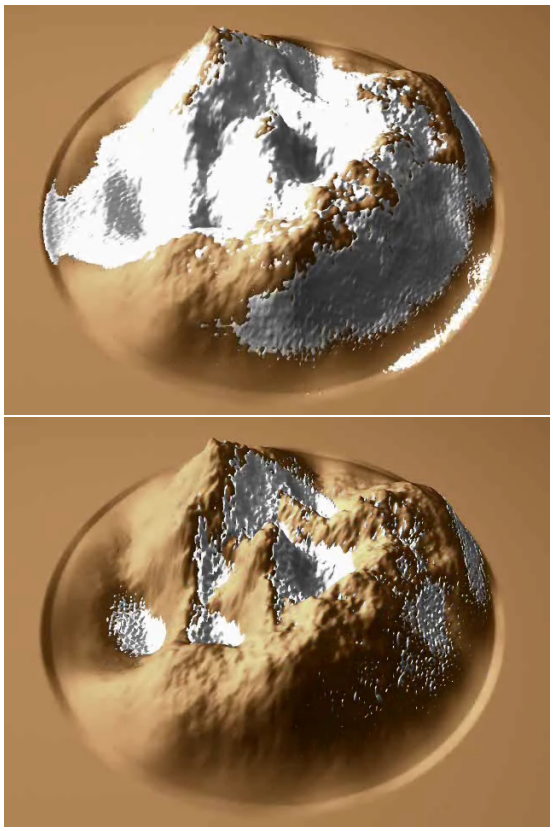

Figure 3: *The effect of the threshold value τ on the snow distribution*

of the snow layer increases it is likely to avalanche [Fea00] until it reaches a stable position. Vertices of the snow mesh that are not occluded and are located on unstable surfaces such as long and steep inclines will be exposed to the sun and dissipated with higher probability.

Skylight illumination [CW93] of a point P with coordinates $[\Theta, \gamma]$ of a clear sky with Sun is given

$$L(\Theta,\gamma) = L_z \frac{(0.91 + 10e^{-3\gamma} + 0.45\cos^2\gamma)(1 - e^{-0.32\sec\Theta})}{0.274(0.91 + 10e^{-3\gamma} + 0.45\cos^2 z_0)}, \tag{2}$$

where L_z is the luminance of zenith, γ is the angle between the Sun and the point P, Θ is the angle between the zenith and the point P, z_0 is the angle between the zenith and the Sun, and α is angle γ projected into the plane of horizon. The angle γ can be calculated from z_0, Θ, and α:

$$\gamma = \arccos(\cos z_0 \cos\Theta + \sin z_0 \sin\Theta \sin\alpha). \tag{3}$$

Luminance of each point on the sky is calculated from (2) and is averaged over the one day period. This value is used to illuminate each visible point on the snow cover. The occlusion has the same influence as the one described in (1).

© The Eurographics Association 2007.

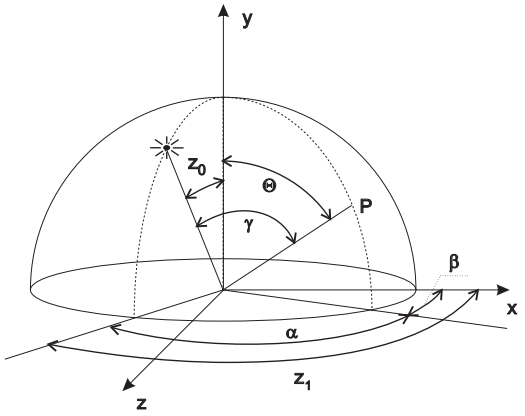

Figure 4: *Values used in the sky illumination*

The amount of snow melted is now the weighted summation of the occlusions of each sun position. Usually the Sun has stronger contribution and the τ_{Sun} should be smaller than the $\tau_{ambient}$. Both parameters are user-controlled.

Occlusion from the averaged Sun's illumination can be efficiently used to calculate the amount of dissipation for each vertex. To do so we use z-buffer to determine the visibility of each point from the position on the sky dome [Ben96]. The camera's viewpoint is first moved to the sun's position. The original mesh is then rendered, followed by each vertex of the new snow mesh as a point. The OpenGL extension ARB_OCCLUSION_QUERY [Len03, WDS06] is again used to efficiently calculate if each vertex is occluded from the sun. Using this extension we can determine the number of fragments that are occluded for the current object. Each vertex is rendered as a point (one fragment), therefore the result of each occlusion query will be either one or zero.

The amount of snow at each vertex is then reduced using (1). Each vertex of the snow mesh that is not occluded is lowered by a proportion of the snow layer, similar to the way in which it was originally added. The occlusion of a vertex being sampled from a single position on the sky is binary. Because of this hard boundaries are created when the mesh is dissipated by a single point sample.Using multiple sun positions accommodates for this as can be seen in Figure 5.

Using a Sun trajectory increases the realism of the snow mesh by averaging dissipation over time as well as smoothing hard boundaries in the mesh.

4. Implementation and Results

Our application is implemented in C++ and uses OpenGL [WDS06] and OpenGL 2.1 extensions [Len03] to calculate the snow accumulation and dissipation. The terrain models are either generated by a procedural technique [EMP*02] or are loaded as a Digital Elevation Model (DEM) file. Because this method requires multiple renderings we use VBOs and the GPU for efficient rendering. Final images are displayed using either OpenGL rendering or the scene is exported as an OBJ file to Maya and rendered using Mental Ray.

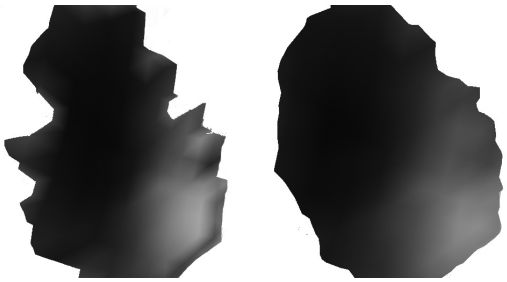

Figure 5: *The effect of sampling multiple Sun positions. Left: Snow dissipation from a single sample. Right: Snow dissipation from ten samples*

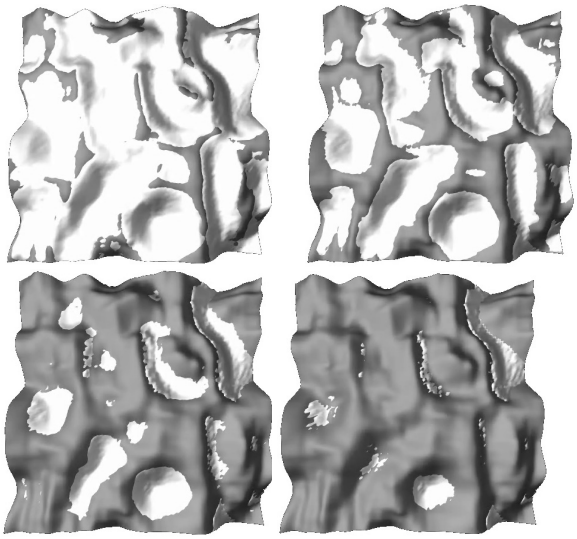

Figure 6: *The effect of sampling snow dissipation due to the Sun illumination and ambient occlusion in a series of images taken from the top*

The implementation was tested on a nVidia GeForce 8800 GTX with an Intel 2.2 Duo Processor. A majority of the calculation time is spent in the first step, calculating the ambient occlusion. The use of VBOs increased performance by up to 1500% compared to a naïve implementation. For low detailed models, calculation time is only few seconds, larger models take up a few minutes. Exact numbers can be seen in Figure 7. For the most complicated scene the calculation time was about seven minutes. It is also important to mention that the resolution of the window for the occlusion calculation has a very small effect on both the speed of calculation

© The Eurographics Association 2007.

and the quality of results. We used 800×600 pixels in all figures showed in this paper.

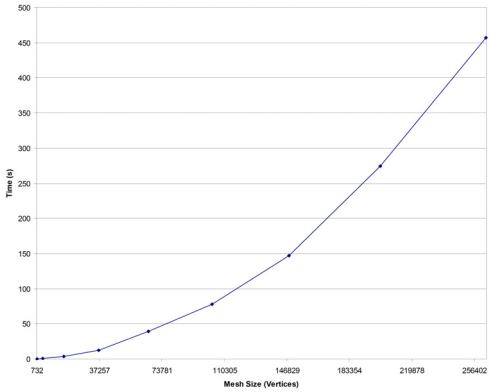

Figure 7: *Calculation time as the function of the scene complexity*

There are various parameters that user can control. The first is the effect of the ambient occlusion. The calculation of the ambient occlusion is the most time consuming and only needs to be precalculated a single time if the surface of the terrain does not change. It is stored as a luminance texture and is used to generate the snow layer and its size can be determined by a user specified parameter that multiplies the occlusion value for each vertex.

The second parameter is the skylight. The skylight sampling takes an insignificant amount of time (milliseconds) and the snow mesh can be modified in only a few seconds by varying parameters of the snow calculation. A user can define the orientation of the scene with respect to the sun trajectory and recalculate the occlusion map. Typically ten samples from the sun trajectory is sufficient to account for smooth edges. Animations of the snow by altering variables of our equations over a series of exports can be seen in the attached video.

The first series of images in Figure 2 shows the original mesh, snow accumulation by ambient occlusion only and the snow accumulation with the ambient occlusion and the Sun trajectory. As can be seen in the last image snow tends to stay on the "North side" of the mountain, i.e., on the side that is not directly exposed to the Sun. However, some snow stays in some holes because these areas are not easily reachable by the direct illumination. The ambient occlusion provides further protection to those areas.

Another series of images in Figure 6 shows a low frequency terrain with shallow mountains that is exposed to direct sun illumination and to the ambient occlusion. As can be seen the effect of the sunlight is not strongly expressed due to the fact that virtually all faces of the mountain are exposed to the sun to certain extent.

The last example in Figure 8 shows that the effect of Sun illumination has much stronger effects on sides of a mountain that are steep and therefor better protected or exposed to the sunlight. The model of the Puget sound terrain (from the Large Geometric Model Archive of Georgia Institute of Technology) has about 500k triangles and it takes about 7 minutes to preprocess it.

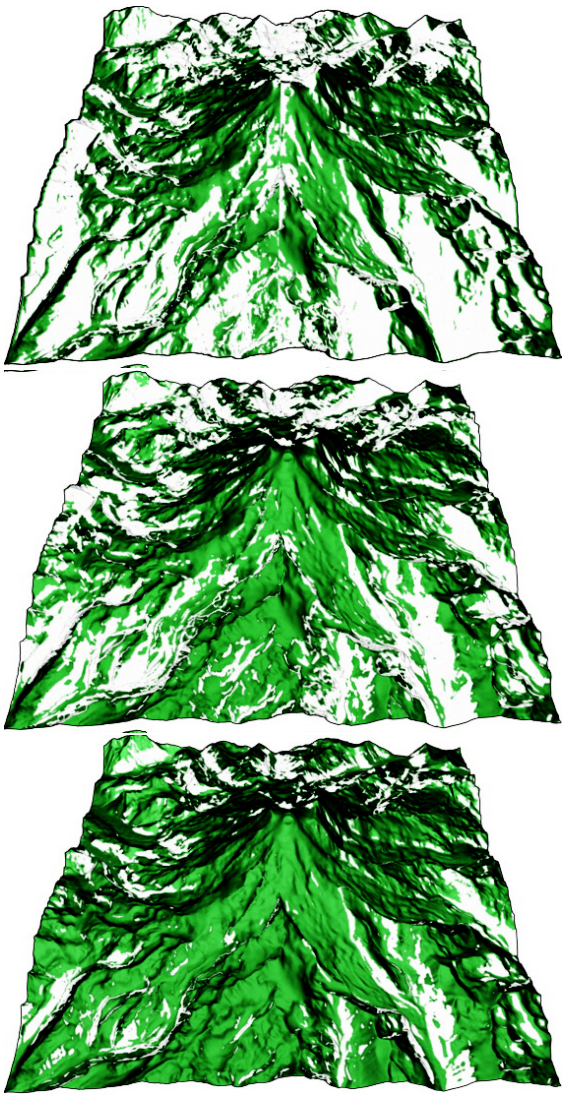

Figure 8: *The effect of snow dissipation is clearly expressed on a steep hills of a mountain of the Puget sound terrain*

© The Eurographics Association 2007.

5. Conclusions and Future Work

The procedural snow modeling technique described in this paper is a quick and effective way of generating snow-covered terrain used for large distance views. Using lighting equations for snow generation creates realistically effective results and allows for simulation of snow dissipation over time that can be applied to computer animations.

Future work on this method should include integration of the technique with some falling snow simulation algorithms, such as [Fea00, MC00] or [Tok06]. Other factors such as re-distribution from wind, erosion, and infrared transfer may prove to be useful in simulating snow in the future. The actual bottleneck of the implementation is the ambient occlusion calculation that is evaluated by the brute force GPU power. It would be interesting to compare speed of this calculation with some optimized CPU technique using space subdivision methods such as BSPs.

The limitation of the algorithm is the actual size of the video RAM of the GPU. If a model of the terrain needs more memory than is available the actual algorithm will not be able to use VBO and it will significantly hurt its performance. Another future work therefore includes some kind of out-of-GPU-core snow accumulation simulation.

References

[Ben96] BENEŠ B.: An Efficient Estimation of Light in Simulation of Plant Development. In *Computer Animation and Simulation'96* (1996), Springer Computer Science, Springer–Verlag Wien New York, pp. 153–165.

[cHtW95] CHI HSU S., TSIN WONG T.: Simulating dust accumulation. *IEEE Comput. Graph. Appl. 15*, 1 (1995), 18–22.

[CW93] COHEN M., WALLACE J.: *Radiosity and Realistic Image Synthesis*. Academic Press Professional, 1993.

[EMP*02] EBERT D. S., MUSGRAVE F. K., PEACHEY D., PERLIN K., WORLEY S.: *Texturing and Modeling: A Procedural Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[Fea00] FEARING P.: Computer modelling of fallen snow. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 37–46.

[Lef03] LEFEBVRE S.: *Shaderx2: Shader Programming Tips & Tricks*. Wordware Publishing, 2003, ch. Drops of water texture sprites, pp. 190–206.

[Len03] LENGYEL E.: *The OpenGL Extensions Guide*. Charles River Media; 1st edition, 2003.

[MC00] MURAOKA K., CHIBA N.: Visual simulation of snowfall, snow cover and snowmelt. In *ICPADS '00: Proceedings of the Seventh International Conference on Parallel and Distributed Systems: Workshops* (Washington, DC, USA, 2000), IEEE Computer Society, p. 187.

[NIDN97] NISHITA T., IWASAKI H., DOBASHI Y., NAKAMAE E.: A modeling and rendering method for snow by using metaballs. *Computer Graphics Forum 16*, 3 (1997), C357–C364.

[OS04] OHLSSON P., SEIPEL S.: Real-time rendering of accumulated snow. In *Sigrad Conference* (2004), pp. 25–32.

[PF05] PHARR M., FERNANDO R.: *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation (Gpu Gems)*. Addison-Wesley Professional, 2005.

[PR95] POMEROY J., RAY D.: *Snowcover Accumulation, Relocation and Management*. National Hydrology Research Institute, 1995.

[PTS99] PREMOZE S., THOMPSON W. B., SHIRLEY P.: Geospecific rendering of alpine terrain. In *Rendering Techniques '99* (1999), Lischinski D., Larson G. W., (Eds.), Eurographics, Springer-Verlag Wien New York, pp. 107–118. Proc. 10th Eurographics Rendering Workshop, Granada, Spain, June 21–23, 1999.

[Tok06] TOKOI K.: A shadow buffer technique for simulating snow-covered shapes. In *CGIV '06: Proceedings of the International Conference on Computer Graphics, Imaging and Visualisation* (Washington, DC, USA, 2006), IEEE Computer Society, pp. 310–316.

[WDS06] WOO M., DAVIS, SHERIDAN M. B.: *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 2.0*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006.

© The Eurographics Association 2007.