International Journal of Multimedia and Ubiquitous Engineering
Vol.10, No.12 (2015), pp.395-404
http://dx.doi.org/10.14257/ijmue.2015.10.12.37

Real-Time Rendering of Snow Accumulation and Melt Under Wind and Light

Jae-Khun Chang[1] and Seung-Taek Ryoo[1*]

[1]*School of Computer Engineering, Hanshin University, Osan-si, Gyeonggi-do, 447-791, South Korea*
jchang@hs.ac.kr, stryoo@hs.ac.kr

Abstract

Snow is a common natural phenomenon that is frequently observed in winter. To simulate the effect of snow accumulation and melt, we generate exposure map from calculating occluding and shadow area using depth map. For simulating snow accumulation, we generate snow accumulated map using dispersed wind direction and occluder slope. We use light map calculating attenuation with distance on light source for simulating snow melt. To render snow in real-time, we use vertex displacement according to snow accumulation and melt. Also, we represent snow diffuse effect using perlin noise and snow boundary using stochastic sampling.

Keywords: *Real-time Rendering, Snow Accumulation, Snow Melt*

1. Introduction

Snow is a common natural phenomenon that is frequently observed in winter. Snow has the ability to completely change the appearance and atmosphere of a scene by placing a white blanket over the landscape, and filling the air with falling and fluttering snowflakes. With the continually increasing speeds of computers and graphics hardware, interest in real-time rendering of phenomena such as snow simulation has increased.

In this paper, we represent snow accumulation under the influence of wind using snow accumulated map which created by depth map based shadow mapping method and snow melt under the influence of the strength of light source using light map. In Section 2, we review previous works related to snow rendering and we explain our snow accumulation and melt method in Section 3 and show the results in Section 4. Lastly, we conclude the paper and discuss future work in Section 5.

2. Related Work

Nishita [1] describes a calculation method for light scattering due to snow particles taking into account both multiple scattering and sky light, and the modeling of snow using metaball. Fearing [2] describes an algorithm for the creation of snow covered models, using a novel particle location scheme that allows surfaces to independently control sampling effort needed to determine accumulation. Haglund [3] present a method for real-time simulation of accumulation of snow on surfaces. They simulate the different stages, starting with a snow free environment and ending with a totally snow covered scene. Felman [4] presents a method for modeling the appearance of snow drifts formed by the accumulation of wind-blown snow near buildings and other obstacles. Ohlsson [5] presents a method of computing snow accumulation as a per pixel effect while rendering the scene. The method is similar to the shadow mapping method for shadow calculations. Saltvik [6] present a novel model for real-time visualization of snow which was developed by integrating several components from previous work related to snow visualization,

ISSN: 1975-0080 IJMUE

and consists of models for falling snow, wind simulation, and accumulating snow. Eidissen [7] present a realistic snow simulation, utilizing modern GPUs to achieve real-time performance. This simulation of snowfall is a computationally expensive problem since each snowflake is simulated interacting with a dynamic wind field. Alexey Stomakhin *et al* [8] presents a novel snow simulation method utilizing a user controllable elasto-plastic constitutive model integrated with a hybrid Eulerian/ Lagrangian material point method. They demonstrate the power of their method with a variety of snow phenomena including complex character interactions.

In this paper, we suggest the real-time rendering system to represent snow accumulation and melt. To show the effect of snow accumulation, we use snow accumulated map which created by depth map based shadow mapping method. Also, to represent snow melt, we use light map which created by light sources such as point, spot and directional lights.

3. Real-Time Snow Rendering

The components for simulating snow consist of snowflake, environment properties (temperature, wind, light, dust, gravity, *etc.*) and surface properties (occluder, surface slope, surface material, *etc.*). In this paper, we use wind, light, occluder position and slope for representing snow accumulation and melt. As shown in Figure 1, suggested system consists of preprocessing, snow accumulation, snow melt and snow rendering phase. First of all, we generate exposure map from calculating occluding and shadow area using depth map. For simulating snow accumulation, we generate snow accumulated map using dispersed wind direction and occluder slope. Also, we use light map calculating attenuation with distance on light source for simulating snow melt. To render snow in real-time, we use vertex displacement according to snow accumulation and melt, represent snow diffuse effect using perlin noise and snow boundary using stochastic sampling.

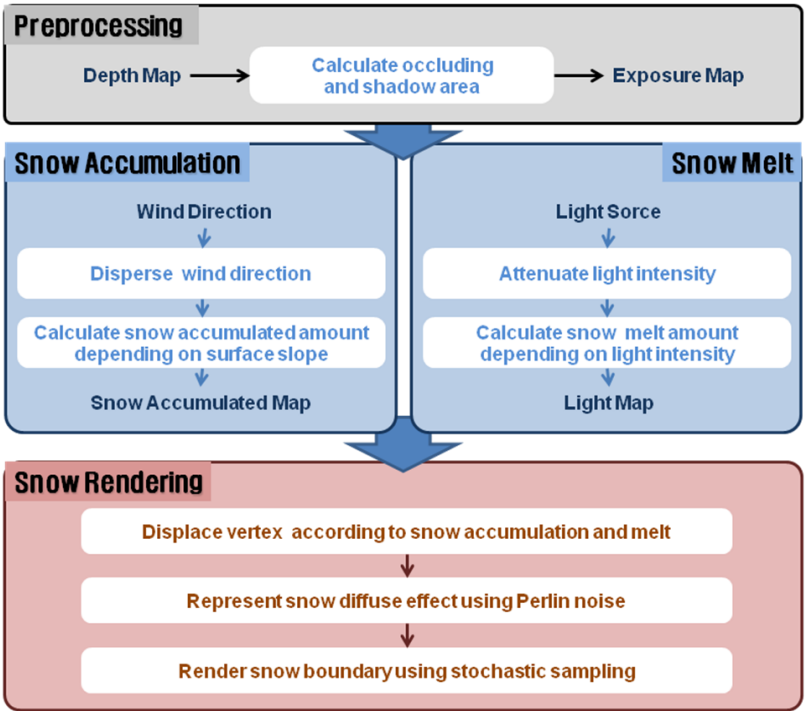


Figure 1. The Overview of Suggested Snow Accumulating and Melting System

3.1. Preprocessing Step

In preprocessing step, we generate exposure map using the difference between depth from orthogonal depth map and transformed depth into projection space. Equation 1 shows how to generate exposure map (E).

$$E(u, v, z_v) = \begin{cases} 1 & if \quad z_v < d + bias \\ 0 & otherwise \end{cases} \tag{1}$$

If orthogonal depth is greater than projected depth then exposure value is 1, otherwise 0. As shown in Figure 2, d means depth value of (u, v) coordinate in orthogonal depth map and zv is transformed depth value into projection space. Also, we add shadow information according to light source into exposure map to simulate snow melt.

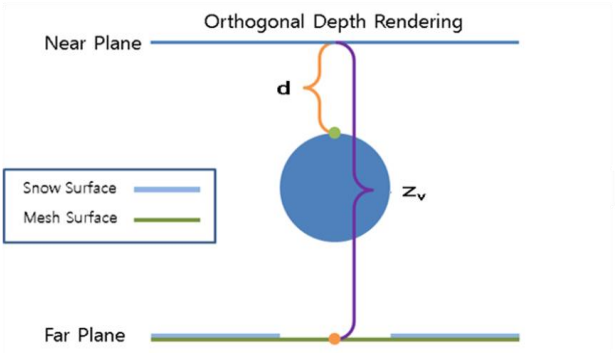
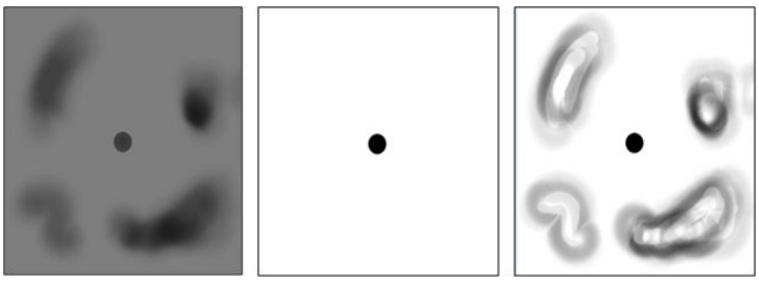


Figure 2. Comparison Orthogonal Depth with Projected Depth

3.2. Snow Accumulation

To calculate snow amount and snow accumulated location, we use occluders, surface slope and wind direction. The locations of snow accumulation can be extracted from occluder position and wind direction, and snow amount can be calculated from surface slope. Equation 2 shows how to generate snow accumulated map (A) using exposure value (E), surface normal (N) and Up Vector.

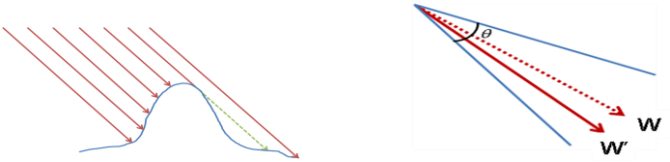
$$A = E \times (N \bullet Up) \tag{2}$$



(a) Depth Map (b) Exposure Map (c) Accumulated Map

Figure 3. The Generation of Snow Accumulated Map (No Wind)

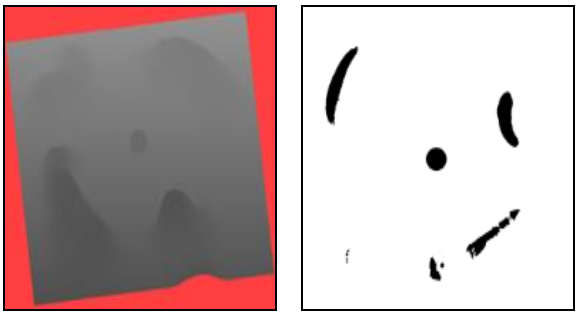
Also, we represent snow accumulation under the influence of wind. To generate wind-driven snow accumulated map, we use depth map from the viewpoint of wind direction. As shown in Figure 4-a, the location of snow accumulation according to wind direction presents red arrow line and the occluded location is a green dotted line.



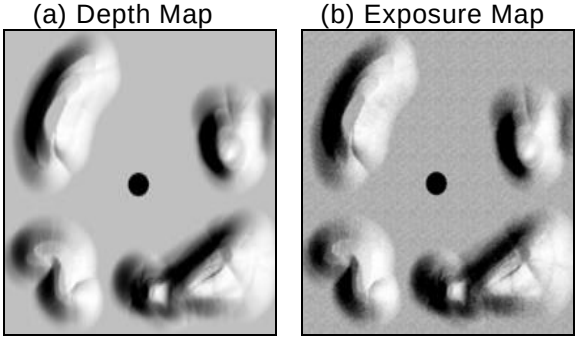
(a) Accumulation With Wind (b) Dispersion of Wind Direction

Figure 4. Snow Accumulation According to Wind Direction

Wind blows not from fixed direction but dispersed direction. As show in Figure 4-b, we implement dispersion of wind direction using perlin noise in specified cone angle (θ). W is wind direction and W' is dispersed wind direction. Figure 5 shows the generation of snow accumulated map according to wind direction and wind dispersion. we generate depth map from the viewpoint of wind direction (Figure 5-a) and exposure map from extracted depth map (Figure 5-b). Figure 5-c shows snow accumulated map with fixed wind direction (left) and dispersed wind direction (right).



(a) Depth Map (b) Exposure Map



(c) Snow Accumulated Map

Figure 5. The Generation of Snow Accumulated Map According to Wind Direction and Dispersion

3.3. Snow Melt

To represent snow melt, we use light map which created by light sources such as point, spot and directional lights. Equation 3 shows how to calculate light map using Phong illumination model. I_L is the quantity of light intensity according to the type of light source (point, spot, directional) and t is control variable on snow accumulated amount according to time. As show in Figure 6, we can be created light map according to directional (left), point (middle) and spot light (right).

$$L = t \times (K_a I_a + K_d (N \bullet L) I_L + K_s (R \bullet V)^n I_L) \quad (3)$$

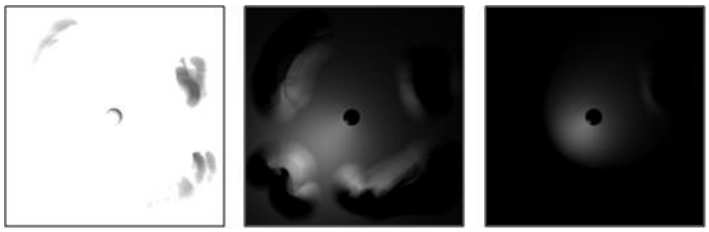


Figure 6. The Generation of Light Map According to Directional, Point and Spot Light

If snow melts, it transfers snowflake into water particle. To represent snow melting phenomenon, we increase the effect of specular reflection according to the intensity of light map. To do this, we change specular reflection constant of surface material (K_s). Figure 7 shows the snow melting effect according to altering specular reflection.

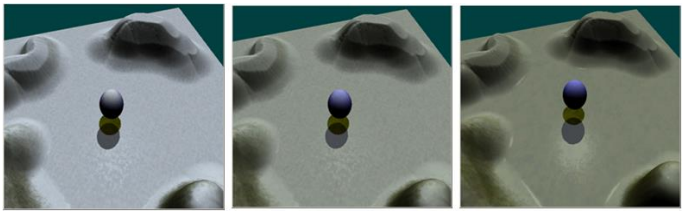


Figure 7. Snow Melting Effect (Left: Accumulated Snow, Middle: Melted Snow, Right: Melted Snow according to Increasing Specular Reflection)

3.4. Snow Rendering

To represent the effect of snow accumulation and melt, we displace surface vertex according to snow accumulation map and light map. Equation 4 shows how to calculate surface vertex displacement. Displaced vertex(P') can be calculated from original surface vertex(P), perlin noise (F), snow accumulated value(A), light intensity from light map(L) and surface normal(N).

$$P' = P + t \times F \times A \times (1 - L) \times N \tag{4}$$

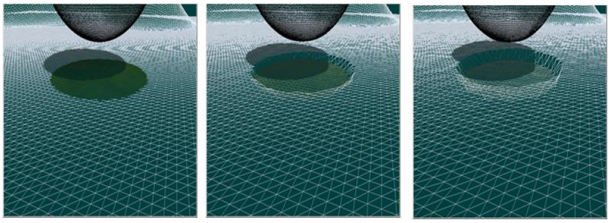


Figure 8. Vertex Displacement

Also, we represent diffuse effect of snow using dispersion of surface position and normal. To do this, we displace surface position using perlin noise function (F) as shown in Equation 5. A and B is control constant of amplitude and frequency according to texture value (x) from three dimensional volume noise texture. Figure 9 compares simple snow rendering (left) with snow rendering using Perlin noise function (right).

$$F = \sum_{i=0}^{n} A^{i} noise\ (B^{i} x) \quad (0 < A \leq 1, 1 < B) \tag{5}$$

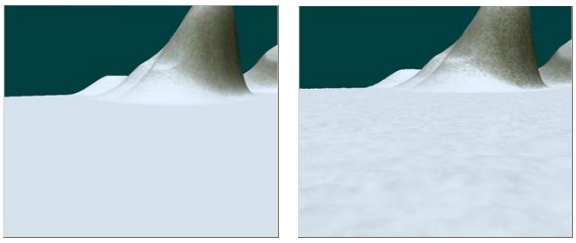


Figure 9. Snow Rendering using Perlin Noise Function

To render the boundary of snow accumulated location, we use stochastic super sampling such as grid, random, poison disk and jitter. Figure 10 shows snow rendering using super sampling algorithms (left: no sampling, middle: grid sampling, right: super sampling with jitter).

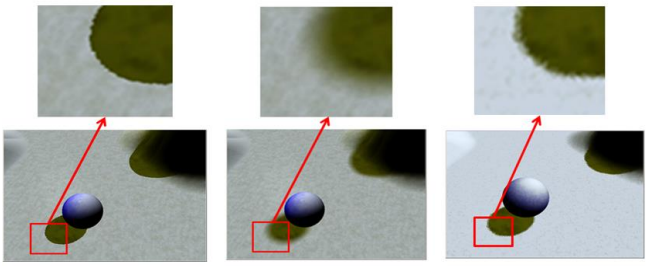


Figure 10. Snow Rendering Using Stochastic Super Sampling

4. Results

In this paper, we implement HLSL based rendering system using nvidia geforce 9800GT. As shown in Figure 11, we simulate snow accumulated rendering according to surface slope. To test snow accumulated amount, we change surface slope from 0 to 63 degree. We simulate wind-driven snow accumulation. To control wind direction, we use user interface (bottom-left circle) as shown in Figure 12. Figure 13 shows the result of snow accumulated rendering over time.

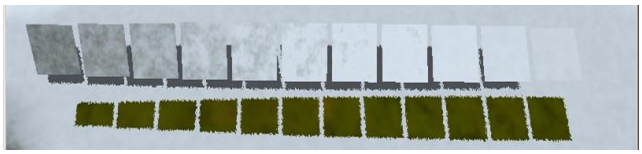


Figure 11. Snow Accumulated Rendering According to Surface Slope

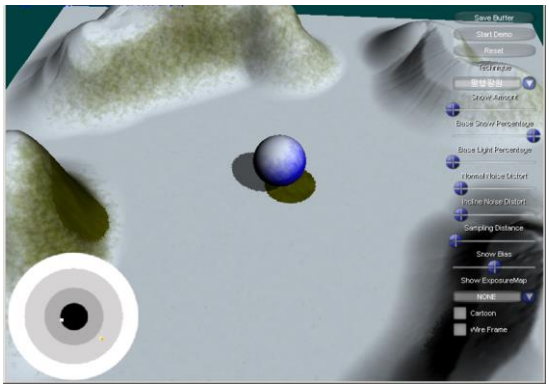


Figure 12. User Interface to Control Wind-driven Snow Accumulation

To render snow melting effect, we apply light intensity according to point light as shown in Figure 14. We use light intensity from light map such as 0, 0.33, 0.66 and 1. Figure 15 shows the result of snow rendering according to point and spot light. As shown in Figure 13 and 15, we simulate snow accumulation and melt according to altering wind and light source.

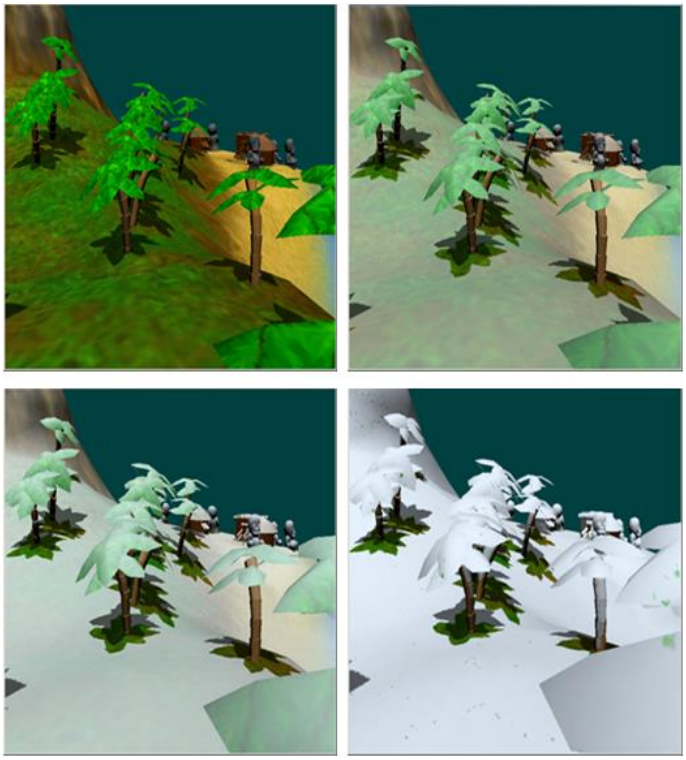


Figure 13. A Result of Snow Accumulated Rendering

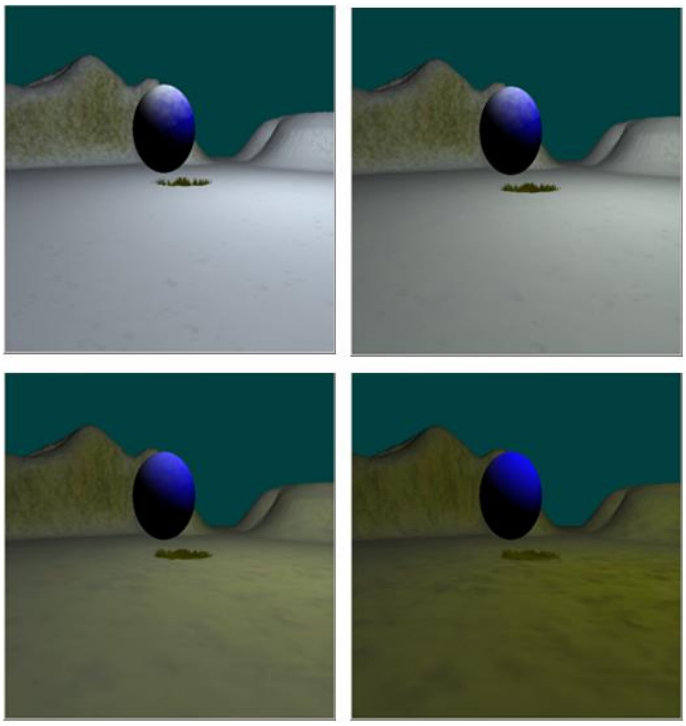


Figure 14. Snow Melting Effect According to Light Intensity from Light Map

Figure 15. Snow Rendering according to Point and Spot Light

5. Conclusions

In this paper, we simulate the effect of snow accumulation and melt. To accumulate snow according to wind direction, we use snow accumulated map which is created by depth map based shadow mapping method. To melt snow according to illumination, we use light map which is calculated from the influence of the strength of light source.

We suggest the effect of snow accumulation and melt using depth based vertex displacement. But, this suggested method cannot represent snow accumulation and melt in detail. To overcome this problem, we'll expand our system to use particle based snow accumulation and melt in future work.

Acknowledgement

This research was supported by Hanshin University Research Grant.

References

[1] T. Nishita, H. Iwasaki, Y. Dobashi and F. Nakamae, "A modeling and rendering method for snow by using metaballs", Computer Graphics Forum, vol. 16, no. 3, **(1997)**, pp. C357-C364.
[2] P. Fearing, "Computer Modeling of Fallen Snow", Proceedings of the 27th annual conference on Computer graphics and interactive techniques, **(2000)**, pp. 37-46.
[3] H. Haglund, M. Andersson and A. Hast, "Snow accumulation in real-time", Proceeding of the SIGRAD 2002, **(2002)**, pp. 11-15.
[4] B. E. Feldman and J. F. O'Brien, "Modeling the accumulation of wind-driven snow", ACM SIGGRAPH 2002 Conference Abstracts and Applications, **(2002)**, pp. 218-218.
[5] P. Ohlsson and S. Seipel, "Real-time rendering of accumulated snow", Proceeding of the SIGRAD 2004, **(2004)**, pp. 25-32.

[6] I. Saltvik, A. C. Elster and H. R. Nagel, "Parallel Methods for Real-Time Visualization of Snow", Lecture Notes in Computer Science, vol. 4699, **(2006)**, pp. 218-227.
[7] R. Eidissen, "Utilizing GPUs for real-time visualization of snow", Master Thesis, IDI, NTNU, **(2009)**.
[8] A. Stomakhin, C. Schroeder, L. Chai, J. Teran and A. Selle, "Material-Point Method for Snow Simulation", ACM Transactions on Graphics, vol. 32, no. 4, **(2013)**, pp. 102:1-102:10.

Authors

Seung-Taek Ryoo, received his B.S. and M.S. degrees in Computer Science Engineering from Chung Ang University in 1996 and 1998, respectively. He had received his Ph.D. in Department of Image Engineering, Graduate School of Advanced Imaging Science, Multimedia and Film from Chung Ang University in 2002. He is currently a professor in School of Computer Engineering, Hanshin University, Korea. His interests are computer graphics, image-based rendering, real-time rendering, non-photorealistic rendering.

Jae-Khun Chang, received his B.S. degree from Hanyang University in 1985. He had received his M.S. degree from Department of Computer and Information Science in New Jersey Institute of Technology and Ph.D. from Department of Computer Science in University of South Carolina in 1989 and 1997 respectively. He is currently in School of Computer Engineering, Hanshin University, Korea. His interests are computer vision, ITS, image processing, pattern recognition, and motion tracking.