Arctic Ice: Developing the Ice Look for How to Train Your Dragon 2

Feng Xie Mike Necci Jon Lanz Patrick O'Brien Paolo de Guzman Eduardo Bustillo
DreamWorks Animation

From the arctic far north to the [1]Bewilderbeast created icy mountains, ice is a dominant setting in How to Train Your Dragon 2. Production design called for a physically realistic look, as reflected in Figure 1. In this talk, we present the creative and technical solutions we developed to overcome the challenges in realizing the artistic vision for Dragon 2 ice.

Figure 1: *Dragon 2 Art Reference for the Ice Cathedral*

Developing the look of Glacier Ice

To achieve the look of Dragon 2's glacier like ice, elements that were massive in scale, contorted in shape and intersected by many complex objects, we relied heavily on translucency as the foundation of our ice look, but we also added raytraced glossy refraction and volumetric attenuation for close-up details.

Core Translucency The core translucency techniques we used were: a). Hierarchical point based translucency by Jensen and Bueler [2002] using dipole falloff presented in [Jensen et al. 2001], based on physically based dipole diffusion, this BSSRDF model effectively captures light scattering inside highly translucent material like ice; b). Noise free translucency by [Anders and Mertens 2007] which enables smooth looking translucent objects with a lower density point file; c). A ramp map that gives users the control for varying color falloff through a translucent material that can be used to illustrate how color deepens as the ice thickens; d). Area compensation: a technique we developed to compensate for energy loss in translucency for objects with sharp corners; e). Poisson point distribution: a new method for generating evenly distributed translucent points independent of object topology.

Ramp control for Translucency It is often the case that large ice formations show a deepening of the scattering color as illustrated in

Figure 2: *Reference photo of glacier ice*

Figure 2. To make it easier for artists to control how color deepens as the ice gets thicker, we added a ramp map that allowed the user to shift the color of the scattering source according to the distance between the scattering source and shading point, as illustrated by Figure 3.

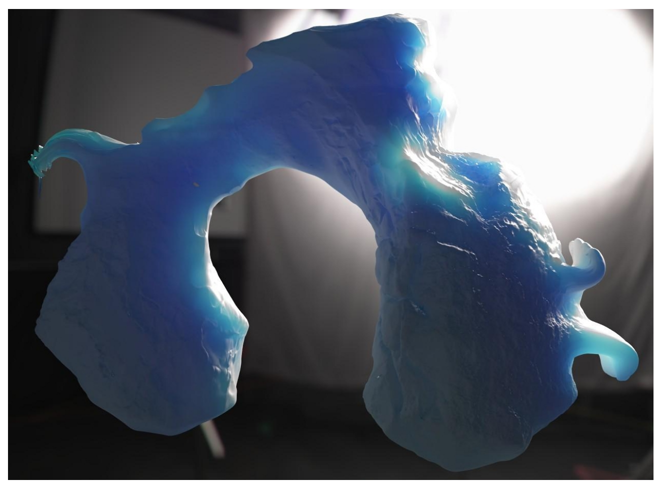

Figure 3: *Ramp control allows user to adjust how the blue deepens inside ice*

Area Compensation Reduces Translucency Energy Loss The dipole model assumes the area around the shading point is infinite relative to the scattering radius. This assumption is violated by geometric surfaces with many sharp corners, as in the case of the complex Bewilderbeast ice, which had many dark translucency artifacts due to energy loss in the sharp ice tips. The solution lies in computing a compensation factor for each shading point based on the ratio of the actual surface area with valid translucent points w.r.t the ideal surface area based on the scattering radius. [2]

$$ActualArea = \sum_{k=1}^{n} area_k * bssrdf(r_k);$$
$$IdealArea = \int_0^{scatteringRadius} bssrdf(r) * rdr;$$
$$areaCompensationFactor = 1 - \frac{ActualArea}{IdealArea}.$$

Assuming the ice material is mostly homogenous w.r.t subsurface

DigiPro 2014, August 09, 2014, Vancouver, British Columbia, Canada.

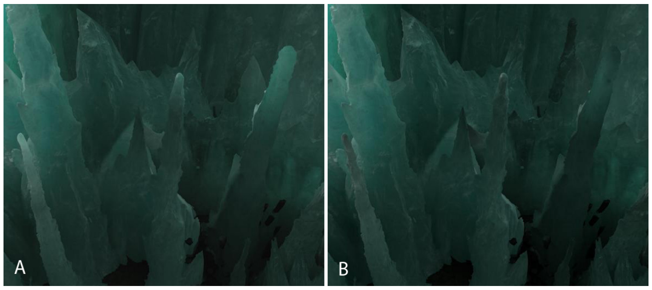

Figure 4: *Rendering comparison of ice tips (A) with and (B) without area compensation for recovering energy loss*

scattering, the solution works very well as illustrated in Figure 4. It is simple to implement and proves to be general and robust in production usage, as it was used for all the ice renders in Dragon 2 with no tweaking required. Even though we used point based translucency for Dragon 2, the area compensation technique for reducing energy loss in dipole based translucency can be easily added to raytracing based translucency solutions.

Poisson Points for Translucency One common technique for point generation in a scanline based renderer is to leverage the micropolygon tessellator to generate a point for each micropolygon. Even though the micropolygon tessellation uses a distance metric, the resulting points often do not form a uniform coverage of the surface area, as desired by the translucency computation. To improve the quality of translucency computation without increasing point file size, we developed an approximate Poisson point generator using a greedy algorithm that yields a point set where the minimum distance between any two points is guaranteed to be above the specified threshold. Figure 5 shows that our Poisson points forms a more even distribution over the surface compared to the regularly tessellated points, and Figure 6 is a close up view of evenly distributed Poisson points over Bewilderbeast Ice.

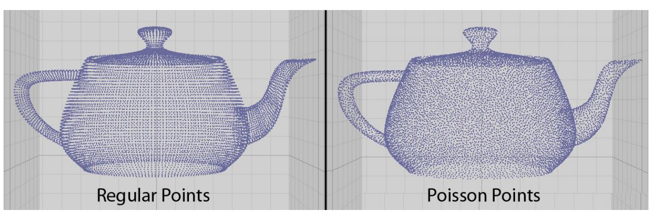

Figure 5: *Comparing point distribution over a teapot using regularly tessellated points vs. approximate Poisson points*

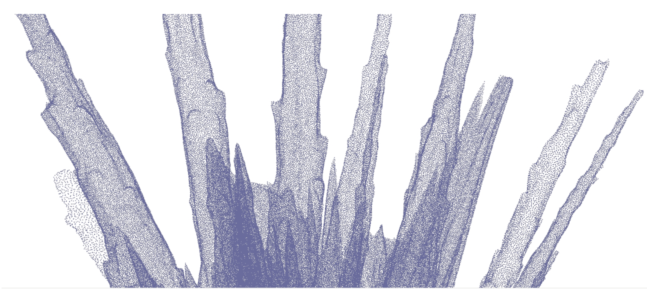

Figure 6: *Close up view of evenly distributed points on Bewilderbeast Ice*

Shading and Lighting The main shader network for ice consisted of a basic translucent ice material overlaid with a crack material in various ways. The overlays added the frosty lines and cracks on top of the translucent ice and were used to control how new or old the ice looks: the fresher the ice the less opaque the cracks. In terms of lighting, the layered translucent ice shader gave good results with a basic lighting rig consisting of environment lighting and small area lights. Quite often the same set of lights was used for both ice and non-ice assets since the goal was to achieve a consistent and physically realistic lighting look. Final finishing, however, sometimes required ice-specific light sources that could be targeted to affect specific components such as diffuse or translucency, as they could add or subtract locally to meet the look requirement.

Raytracing An ice look wouldn't be complete without ray tracing. We found seeing refractions through ice affected the sense of scale in that the ice appeared small if the refractions were too clear. At most we should only see blurry silhouettes in our glacier-like ice; as a result the ray trace passes were extremely simple, composed of mattes used to color correct the various render passes to hint at the layers of ice and embedded rocks below. For the destructive ice formed dynamically by Bewilderbeast's breath, we also added a raytraced volumetric pass to emulate single scattering of light through the ice which gave it more volume.

AOV Passes Final ice rendering was a multi-pass rendering process, as a result it was made significantly more efficient with the addition of AOV support in our proprietary renderer. Figure 7 shows a breakdown of the various render passes for ice environments.

Results

To summarize, we developed a combination of techniques to achieve the look of glacier ice in Dragon 2. Together they addressed these essential requirements of the production workflow in a feature animation:

Artistic control: Our layered ice shader was flexible enough to create many different ice looks ranging from aged arctic ice to dynamicly formed Bewilderbeat ice; the ramp color for translucency gave our artists an intuitive control on deep scattering; and AOVs allowed lighting artists to iterate over different rendering components without having to re-render the whole scene.

Performance: Evenly distributed Poisson points enabled us to create smooth transluency effects across massive ice formations without having to use extremely large point sets, and consequently yielded significant improvement in final rendering performance and reduced translucency and occlusion point generation time.

Quality: Area based compensation was a simple extension to the dipole translucency integrator that effectively removed dark artifacts in the translucency of sharp ice tips and geometric extremities. This simple translucency improvement saved production hundreds of hours of paint fix time as ice was featured in over half of the sequences in the movie.

By leveraging these techiniques developed on top of our proprietary lighting and rendering system, we were able to achieve the visual goals for ice in Dragon 2 and deliver them consistently across over 800 production shots [3] with heavy ice elements. Figure 8 illustrates how closely our final ice renders mirrored the painted lighting references.

References

ANDERS, L., AND MERTENS, T. 2007. Noise-free bssrdf rendering on the cheap. *SIGGRAPH'07 posters*.

JENSEN, H., AND BUELER, J. 2002. A rapid hierarchical rendering technique for translucent materials. *SIGGRAPH 2002*.

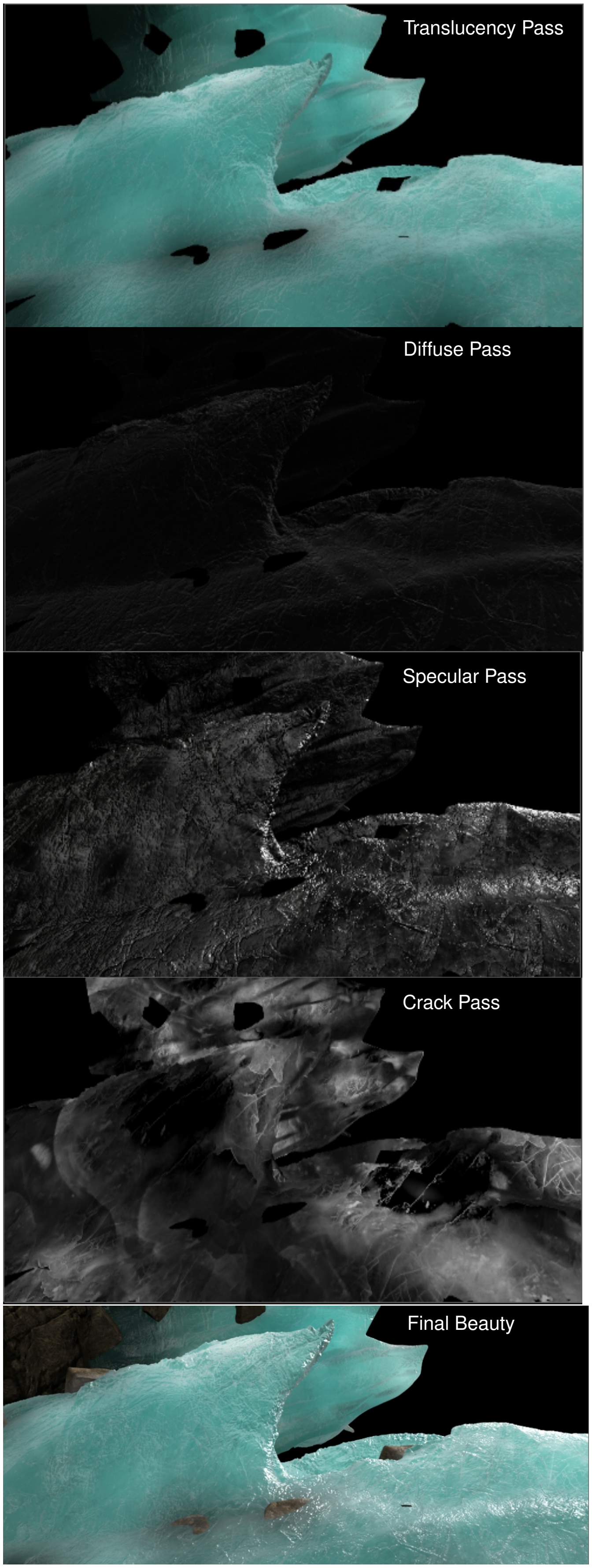

Figure 7: *AOV Passes for Ice*

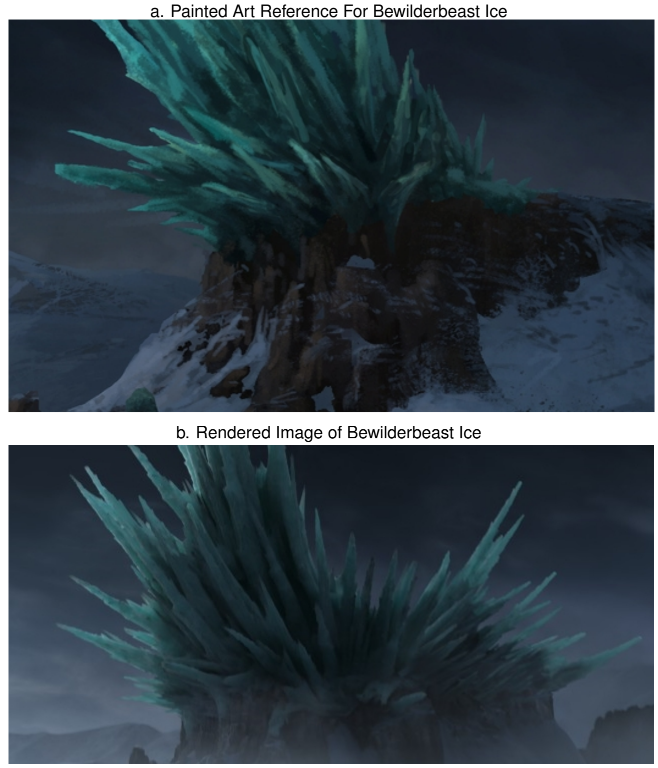

Figure 8: *Comparing painted reference and rendered image of ice mountain*

JENSEN, H., MARSCHNER, S. R., LEVOY, M., AND HARAHAN, P. 2001. A practical model for subsurface light transport. *SIGGRAPH 2001*.

Notes

[1]Bewilderbeast: a key character in Dragon 2 with the power to create massive ice formations.

[2]Both the actual integrated area and the ideal integrated area are scaled by the bssrdf kernel at each integration step.

[3]The complete film had 1615 shots.