Lumo: Illumination for Cel Animation

Scott F. Johnston

Fleeting Image Animation, Inc.
www.fleetingimage.com

Abstract

A method is presented to approximate lighting on 2D drawings. The specific problem solved is the incorporation of 2D cel animation into live-action scenes, augmenting the existing method of drawn "rims and tones" with subtle environmental illumination. The image-based tools developed to solve the problem have both photorealistic and non-photorealistic applications.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—display algorithms; I.3.5 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, Shading, Shadowing, and Texture.

Keywords: cel animation, sparse interpolation, non-photorealistic rendering

1 Introduction

The primary components used to illuminate a point on a surface are its position and surface normal. In hand-drawn artwork, the surface normal is unknown, and the position information lacks depth. Previous work completes the data through geometric methods—creating a surface from 2D information [Igarashi et al. 1999], or fitting a geometric model to the data [Correa et al. 1998]. Their processes establish depth, giving the object geometry so that a surface normal can be computed. These methods provide useful results and can work well for integrated 2D/3D pipelines. However, in "cartoony" drawn animation, a few lines can alter an object's geometry; geometric methods work best when the objects are less fluid. *Lumo* utilizes image-based techniques to approximate a surface normal directly, avoiding the jump to 3D geometry, and to provide a generalized solution for lighting curved surfaces changing over time. As there is no need to see around the objects, accurate depth information is not a primary goal; convincing illumination is generated from 2D positions and approximate normals.

In implementation, this method estimates normals wherever possible and uses sparse interpolation to approximate them over the remaining image. Edge information is used to identify localized layering within an image. To improve the overall result, multiple approximations are blended together weighted by "confidence mattes" generated from the layering information. Rendering is performed with either 3D tools, or via compositing operations to light the normal image and apply it to the original artwork.

1.1 Previous Work

Most prior work in this area relies on reconstructed geometry. Systems like Teddy [Igarashi et al. 1999], and SKETCH [Zeleznik et al. 1996] provide interactive tools for building 3D models from 2D data, automating the "inflation" to 3D geometry. Petrovic's work expands these ideas and applies them to existing drawings to create shadows for cel animation through derived geometry [Petrovic et al. 2000]. Correa et al. [1998] take existing geometry and deform it to match cel animation for texturing and illumination.

While pursuing the method for painterly rendering described by Meier [1996], we researched the diffusion of parametric values across an arbitrary field of particles, allowing texture-like assignment of values where there was no underlying geometry. More recently, sparse interpolation has been used for image reconstruction in Lumigraphs [Gortler et al. 1996].

The process presented is also related to the problem of "toon" shading, as it partially inverts edge detection methods. The author recommends [Saito and Takahashi 1990], [Apodaca and Gritz 2000], and [Markosian et al. 1997] for background on image-based and analytic edge-detectors.

1.2 Background

For more than 80 years, artists and film makers have attempted to blend animation and live action seamlessly to create fantasies on the screen. High points in the integration of the two media include the Fleischer Studio's "Out of the Inkwell" series in the 1920's, MGM's "Anchors Aweigh" (1945), Disney's "Mary Poppins" (1964) and the Disney-Amblin co-production "Who Framed Roger Rabbit" (1988) [Solomon 1989]. In each case, the artists utilized and often invented the most sophisticated technologies available at the time: rotoscope, sodium/blue-screen, traveling mattes, optical printing, etc. While 3D CG tools can be used effectively to design and create new characters, using them to recreate drawn characters faithfully has proved expensive and often ineffectual.

1.2.1 Traditional ink and paint, rims and tones

One problem that has dogged attempts to integrate media is the inescapably flat look of the drawn artwork. In traditional ink and paint, the animators' drawings are traced onto the front surface of clear acetate cels in ink. Each cel is then flipped over and paint is applied to its back surface, producing flat areas of color with ink lines hiding the boundaries between paint regions. The process is an early example of an indexed color system: appropriate paint colors are chosen based on the region to be filled. Film makers have attempted to soften the flatness of traditional ink and paint since the mid-1930's, when artists applied real rouge to Snow White's cheeks.

Rims and tones are effects mattes drawn to add highlights and to distinguish between the lit and shadowed sides of an object. They add detail to the original drawing, help place it into a particular lighting situation and can intensify a mood. By heightening the perceived volume of an object, tones enhance its believability in its

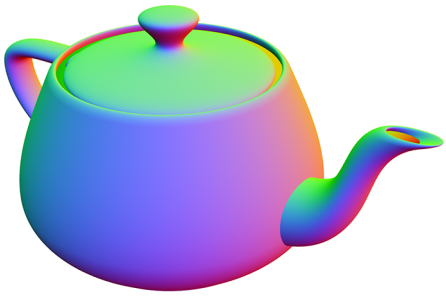


Figure 1: **Rendered RGB representation of Normals.**

environment. Generating tones by hand is time-consuming, but is commonly used to place 2D animation into a live-action scene.

Tone mattes are applied over the character in a camera setup to overexpose, underexpose, or filter portions of an image.

1.2.2 Digital ink and paint, rims and tones

Most digital ink and paint systems adopt the traditional strategy of keeping the ink and paint on separate layers and using indexed color. An indexed system has several advantages: It allows final color selection to be delayed in the production pipeline until other color decisions are finalized, and the same source material can be used multiple times with different look-up tables, commonly called palettes.

The ink and paint process is performed using a *painter's palette* where contrasting false-colors help the painters discern regions that may show only slight variations on-screen. *Base palettes* of standard or neutral colors for a character are the starting point for a sequence or location. The base palette is modified on a scene-by-scene basis to place the character in the environment, creating a *styled palette*.

Digital compositing offers more options than traditional ink and paint for applying tone mattes. A character is often color-styled twice, with separate palettes for shadow and light; the two versions are blended with a tone matte as a key. This gives the art director region-by-region control over the effect of the tone. Blurring a tone matte softens the boundary between the two sides. Multiple palettes and mattes can be combined to heighten the effect.

1.2.3 Ink and paint maps

A compact method for storing 2D cel animation is to encode each drawing in a multi-channel image [Apodaca and Gritz 2000]. Scanned grayscale drawings are stored in one channel. For any non-zero pixel in the scan—wherever there is line data—an index value for an ink color is placed in another channel. Regions in a third channel are filled with index values for paint colors. The indexed channels are aliased, but the boundaries between regions are hidden beneath opaque, anti-aliased ink lines, as on a traditional cel. To preserve valid index values, care must be taken when filtering these channels.

Look-up tables are used to turn an ink and paint map into a proper color image. The ink index channel is resolved with an ink palette, multiplied by the grayscale ink value to form an ink layer, and composed over the paint index channel resolved by a paint palette. Different look-up tables can be applied to the same source image for different results, or to pull mattes from specific areas. (E.g. a look-up table where all inks and paints are transparent except for the eyes can be used to pull an eye matte.)

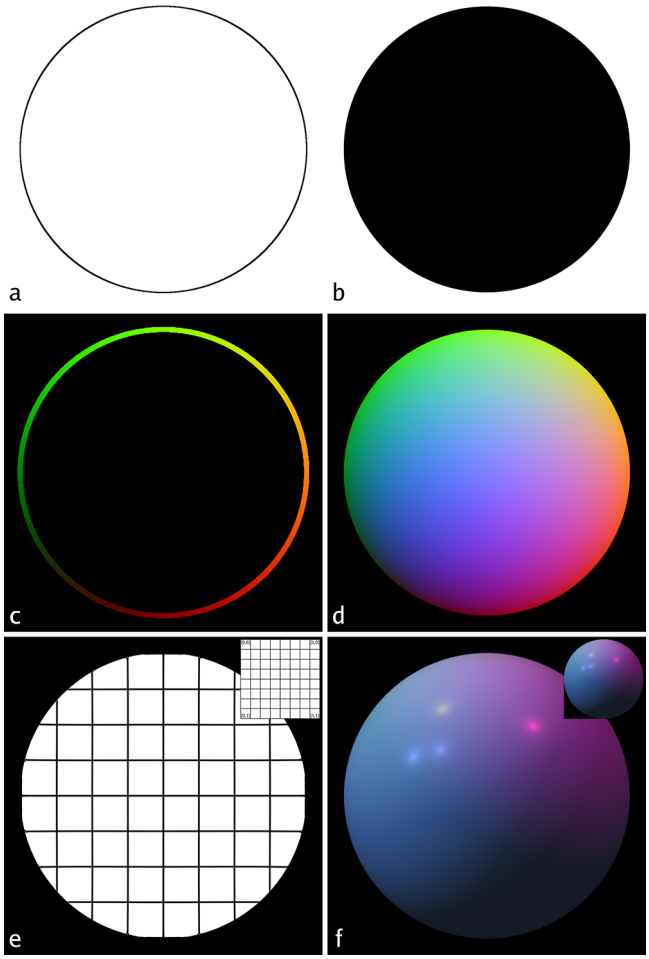


Figure 2: **Region-based normal approximation, interpolation and illumination.**

1.2.4 Normal Image

A normal image is an RGB color space representation of normal vector (N_x, N_y, N_z) information. Signed values are stored in a float image type, or the normal vector range $[-1.0, 1.0]$ is mapped to $[0, ONE]$, centered at $ONE/2$ for unsigned integer storage. The normals can be in any space, but it is convenient to transform them to screen space (X,Y) with depth Z. Figure 1 shows a normal image rendering of the classic teapot.

2 Normal Approximation

One of the principle methods for detecting visible edges in ink and paint rendering is to find points where the surface normal is perpendicular to the eye vector. For curved surfaces, these form the exterior silhouette and interior folds. By inverting this process, normal vectors are generated along these edges using the following methods.

2.1 Region-based normals: Blobbing

As a degenerate case, a drawn circle (Figure 2a) is presumed to be the 2D representation of a sphere. Taking the gradient across the circle's matte (Figure 2b) and normalizing the result produces

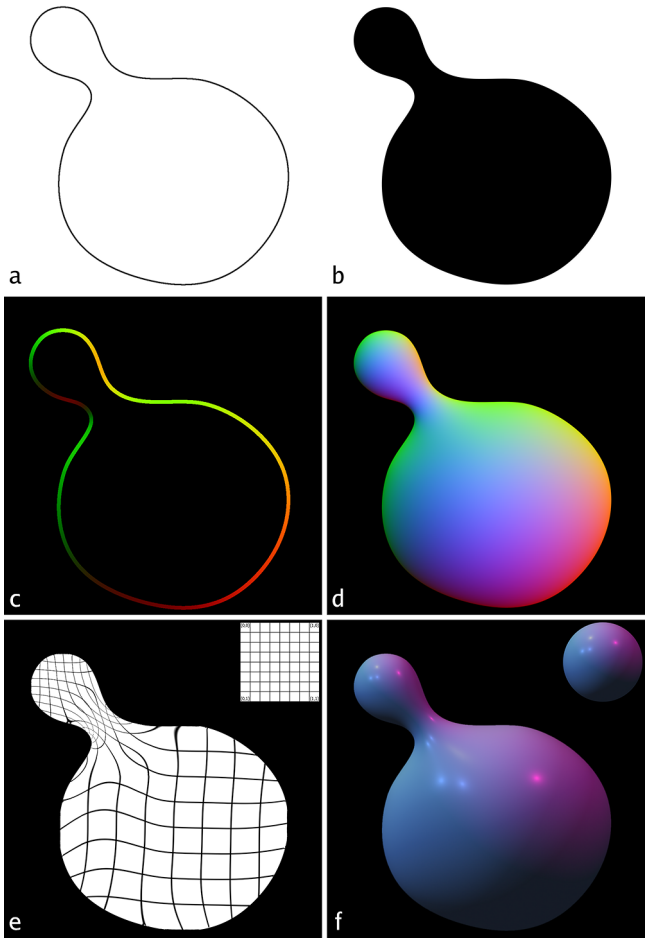


Figure 3: **Region-based normals on arbitrary shape.**

Figure 2c. For an orthographic projection, the z-component of the normals along the edge of a curved surface must be zero to keep the normal perpendicular to the eye vector, making normalized gradients an adequate approximation of the edge normals. (Scaling, introduced in section 2.6.2, allows the results to appear more like a perspective projection.) Interpolating the edge normals across the image generates a field of normal vectors (Figure 2d). The (N_x, N_y) components of the normal image linearly interpolate the field, as illustrated by using (N_x, N_y) as the parametric coordinates for texture mapping a grid image (Figure 2e, the texture source is inset). The (N_x, N_y) space can just as easily map a rendered sphere onto the circle (Figure 2f).

Applying the same process to another curve (Figure 3a-f), shows that the same sphere mapped onto a more arbitrary interpolated normal field provides a good illusion of shape and illumination.

The basic blobby shape created from the exterior boundaries of a more complex image (Figure 4a) has very little detail (Figure 4b). Using mattes to separate regions (Figure 4c), applying the technique on each region, and compositing the results creates a more accurate representation of the character's normals (Figure 4d). However, the result still lacks much of the detail contained in the original image.

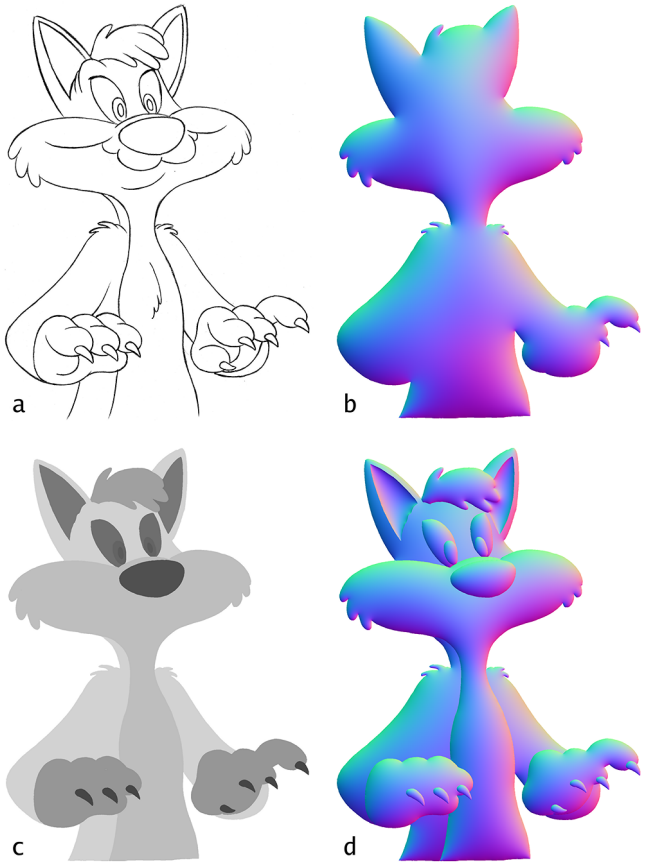


Figure 4: **Compound region-based normals.**

2.2 Line-based normals: Quilting

When the gradient is computed across ink lines, rather than matte edges, opposing normals are generated on each side of the lines (Figure 5a). When these normals are interpolated, much more detail is revealed, but the image appears quilted because the internal folds should only affect the normals on one side of the edge, not both (Figure 5b).

2.3 Over/Under assignment

The key contribution of *Lumo* is the identification and integration of the implied layering within a drawing to assist in generating credible normals.

A drawn line can be interpreted as a boundary separating two regions. In many circumstances, one of these regions overlaps the other; tagging edges with white on the "over" side and black on the "under" side produces an illustration that better defines the layering within the image (Figure 6a). For example, the lines for the cat's cheeks represent the boundary of an internal fold where the cheeks occlude part of his upper lip. The upper side of the cat's cheek lines are tagged white and the lower side black to indicate that his cheeks are over his lip. Similarly, his chin is over his neck, the fingers are layered over one another, etc.

In traditional ink and paint, light ink lines are used within regions of dark paint to reveal detail. These internal lines are called "light over dark" or L.O.D. lines. In *Lumo*, L.O.D. serves as a mnemonic for the painters when assigning layering information to the edges. The ink and paint artists perform "over" and "under" tagging as an additional step in the digital animation process. The tools to tag the

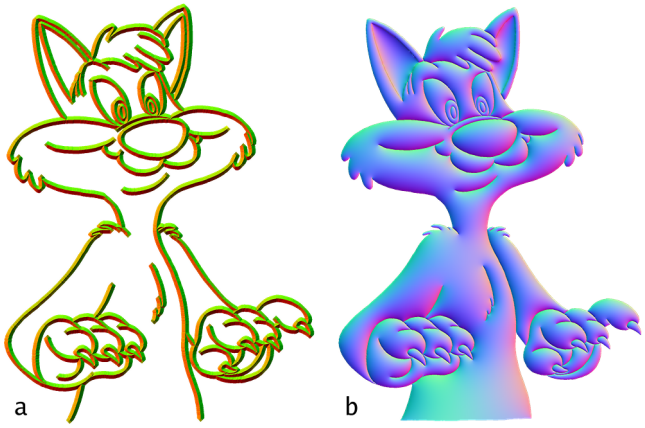


Figure 5: **Quilting: Line-based normal interpolation.**

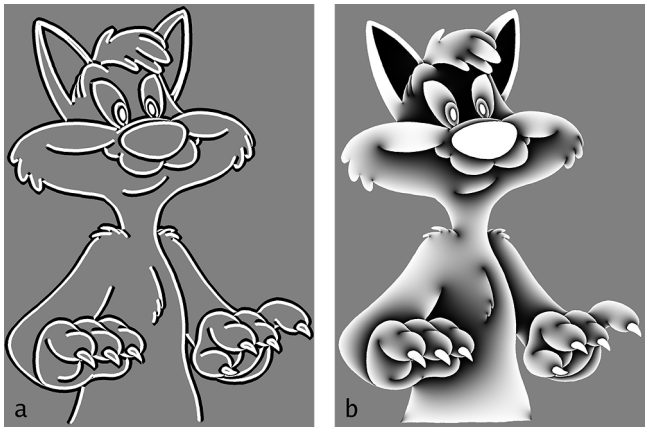


Figure 6: **Over/Under assignment and confidence matte.**

edges make an initial best-guess for each line segment by inspecting a relative-depth list of paint regions. The cat's nose is on top of his muzzle, so the "nose" side of a line bordering those regions is set to white and the "muzzle" side black. For line segments that don't border paint regions, the dominant curvature establishes the initial condition: the inside of a "C" curve is tagged "over" and the outside "under." Interactive tools allow painters to edit the edge assignments for a sequence of images.

2.4 Confidence Mattes

When the over/under edge illustration is interpolated, it forms a grayscale matte (Figure 6b). The quilted normal image is a more accurate representation of the normals on the "over" side of the edges where the matte is white; the normals on the darker "under" side of an edge are less certain. These mattes can be viewed as a measure of confidence in the known normals. The normal on the under side of an edge should interpolate the shape of the underlying object as if that edge didn't exist. In practice, this can be difficult to determine, as it may involve lines that are not part of the image. However, the blobby normal image found earlier is a gross approximation of removing *all* the internal lines.

2.5 Normal Blending

By blending the quilted and blobby normals using the over/under confidence matte as a key, a normal image is formed that resembles

Figure 7: **Blended Normals.** Blending quilted line-based normals with simple region-based normals using a confidence matte as a key generates a believable normal image.

a bas-relief of the object (Figure 7). Quick blending can be performed on the N_x and N_y components of the normal images using linear interpolation, with N_z recomputed from the result to maintain unit-length. This method is adequate when the source normals vary only slightly, but it is more accurate to use great-arc interpolation.

Blending multiple normal approximations, each weighted with its own confidence matte, can incrementally build more accuracy into the resulting image. It is also possible, though not always practical, to clean up the rough animation drawings on multiple levels to aid in the generation of more accurate normals. When the cat's paw passes in front of his face, information is lost for computing normals on his face. This isn't a problem where the face is obscured by the paw, but errors appear on the face at the boundary between it and the paw. The error would be reduced if they were drawn on separate levels.

2.6 Image-Based Manipulation

Normal images and confidence mattes can be manipulated using image-based tools to further improve results. Paint palette mattes are useful for localized control of these adjustments. Tracking an approximate 3D orientation for an object allows normal vector manipulations to be performed in either screen space or object space.

2.6.1 Tapered edges in confidence matte

Open-ended lines create dimples in the interpolated matte at the open end-points. Fading the ends on the "over" side of an open-ended line to black creates a softer, more natural transition. This tapers the effect of the "over" normal towards the tip of the line, as seen in the cat's cheeks in Figure 7.

Similarly, the effect of a normal can be reduced uniformly on a set of lines by scaling the edge confidence with an ink palette

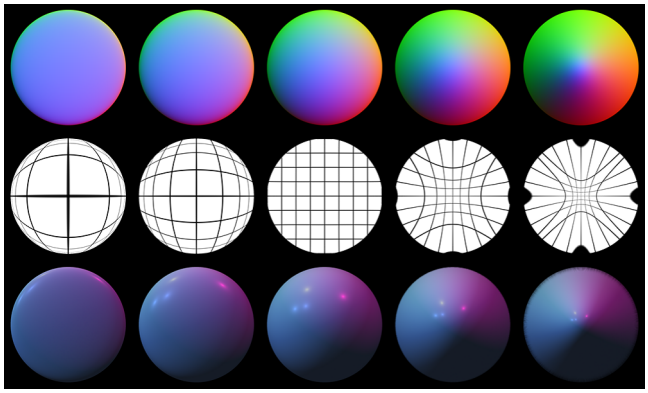


Figure 8: **Z-scaled sphere normals.** S=(0.25, 0.5, 1.0, 2.0, 4.0)

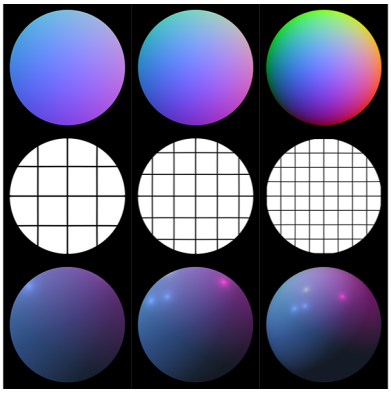


Figure 9: **Uniformly scaled sphere.** S=(0.125, 0.25, 1.0)

prior to sparse interpolation. The cat's chest line has been reduced slightly to make him less busty.

2.6.2 *Z*-Scaling Normal

Two methods of scaling are used to adjust a region's puffiness. The first replaces a normal with the normal of a sphere that has been scaled in *z*; the second replaces a normal with the normal of a portion of a sphere that has been scaled uniformly.

To map a field of normals to a sphere scaled in *z* by *S*, a pixel operation is applied replacing (N_x, N_y, N_z) with:

$$\frac{(S \cdot N_x, S \cdot N_y, N_z)}{\|(S \cdot N_x, S \cdot N_y, N_z)\|} \tag{1}$$

When the scale factor is less than one, the region becomes shallower, and when it is greater than one, the region appears deeper (Figure 8). Transforming the normals into a different space, applying the *z*-scale, and then transforming them back to screen space can achieve similar results in a direction other than *z*.

Rather than maintaining $N_z = 0$ along edges for a steep drop-off, it may be preferable to make the edge normals more subtle. Objects viewed in perspective have slightly non-zero *z*-components on their silhouette. For example, the visible portion of a sphere is a slice from the front of the sphere. To remap hemispherical normals to the normals of a visible unit-sphere slice of thickness *S*, N_x and N_y are linearly scaled by $\sqrt{S \cdot (2 - S)}$ and N_z is recomputed to maintain unit length (Figure 9).

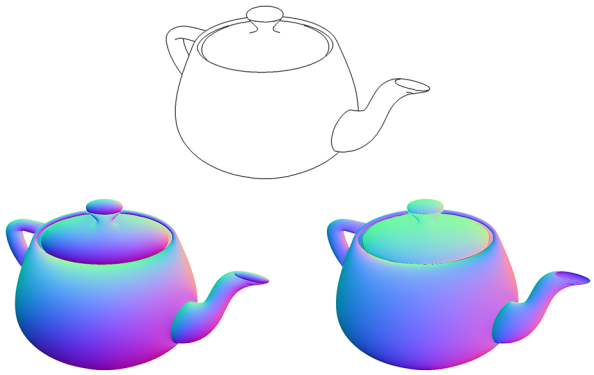


Figure 10: **Drawing of a teapot (top), overly rounded derived normals (bottom left) and color adjusted normals (bottom right).**

2.6.3 Color compositing operations

The normals derived from a drawing of a teapot create an overly rounded result. Scaling the normals of the body by (0,0.5,0), adding (0,1,0) to the normals of the lid, and renormalizing improves the appearance of the teapot (Figure 10). The use of conventional compositing tools to perform these manipulations as pixel operations is a legacy of Ken Perlin's Image Synthesizer [1985].

2.6.4 Drawn Tone Mattes

The silhouette edges of a curved surface mark the transition between the visible and hidden sides of an object from the vantage point of the camera. Similarly, a drawn tone matte defines a boundary between the lit and unlit sides of a character from the vantage point of a light source. Given a tone matte and the location of its light, a normal at a point along the edge of the tone should be perpendicular to the vector from the point to the light. Approximated normals can be refined near a tone matte edge by projecting them onto the plane perpendicular to the light vector and renormalizing.

3 Sparse Interpolation

The success of this technique demands accurate interpolation of values across a field when only a few are known in sparse locations.

Rather than solve the complex problem of interpolating normals as multi-dimensional unit-length vectors, the N_x and N_y components are interpolated independently, and N_z is recomputed to maintain unit length. As presented above, the normals on the edge of a circle should interpolate linearly in N_x and N_y to create the profile of a hemisphere.

Multi-resolution interpolation as described in Gortler's Lumigraph paper [Gortler et al. 1996] is quick, but requires accurate filtering. His interpolation was found to be mostly linear, but eased slightly toward known values, producing distortion when used for illumination. This may have been an implementation issue.

For experimental results, a simple iterative dampened-spring diffuser is used. Known pixel values remain fixed; unknown values are relaxed across the field using dampened springs between neighboring pixels. Given a field of values *P* to be interpolated, and a velocity field *V* initialized to zero, each iteration is computed by:

$$\begin{aligned} V'_{i,j} &= d \cdot V_{i,j} + k \cdot (P_{i-1,j} + P_{i+1,j} + P_{i,j-1} + P_{i,j+1} - 4 \cdot P_{i,j}) \\ P'_{i,j} &= P_{i,j} + V'_{i,j} \end{aligned} \tag{2}$$

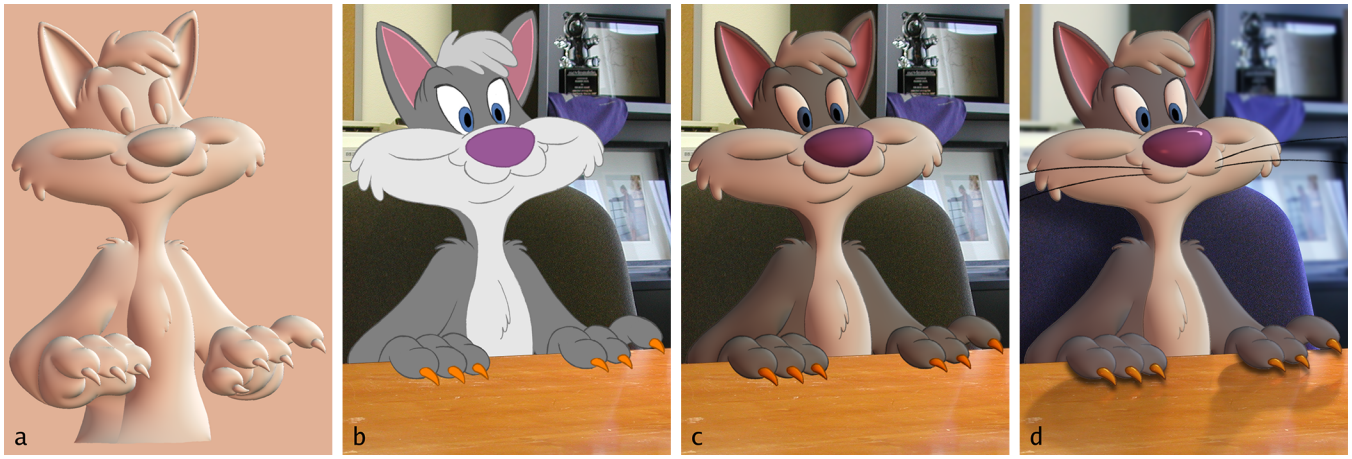


Figure 11: **Applied Illumination.** Normal image illuminated by a diffuse spheremap, base palette cat, cat scaled by illumination, final cat.

Optimal values for the dampening and spring constants vary, but $d = 0.97$ and $k = 0.4375$ minimized the time to convergence for the circle example. Iterations are run until the mean-squared velocity per pixel reaches an acceptable error tolerance.

This inefficient method takes a few minutes per frame for the examples presented, but proved fast enough for prototyping. The result is a smooth minimal surface that satisfies our requirements. As this is a form of the diffusion equation, many methods exist to reach the final steady-state more quickly [Press et al. 1992]. Further investigation into multigrid solvers is planned for a production system.

4 Rendering

Using 2D positions and approximated normals, most rendering techniques can be adapted to illuminate a drawing. For maximum control, standard 3D rendering works best. However, for deployment in a 2D compositing pipeline, it is useful to implement a subset of 3D algorithms as 2D compositing operations. It is also possible to use non-photorealistic (NPR) techniques, and, for integration into live action, captured environmental lighting and high dynamic range rendering.

Standard 3D Rendering To render using standard 3D tools, a rectangle is placed in a scene, and the normals used for illumination are overridden with a texture lookup into the normal image. The shader function can alter material properties for different regions by sampling the ink and paint map as an additional texture source.

Light source compositing operation To render with a 2D compositor given a normal image and the location and color of a light source, standard lighting models are written as pixel operators in the compositing software's scripting language. For example, a light vector image to a raster space light source L is formed by creating a four-corner gradient where the corner colors are set to: $(L_x, L_y, L_z), (L_x - width, L_y, L_z), (L_x, L_y - height, L_z), (L_x - width, L_y - height, L_z)$. Specular and diffuse functions are written independently, but can be combined into more complex procedures. Lighting nodes are applied multiple times and summed for multiple light sources. Additional material parameters such as surface color, specular color, and specular roughness, are stored in look-up tables for use within the lighting functions.

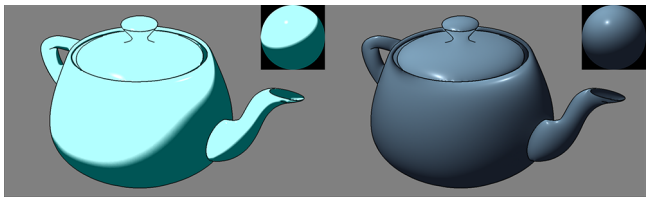


Figure 12: **Spheremap illumination on drawn teapot.** Rendered spheres (inset) provide illumination for arbitrary objects.

Sphere maps All the examples presented have been rendered with the fast lighting approximation introduced in section 2.1. Rather than light the object itself, a rendered sphere is used as a proxy for the lighting situation. This simplified "distant" lighting model ignores position and is based solely on normal information.

A unit sphere has the special property that any point on the sphere corresponds exactly to its surface normal at that point. Therefore, inspecting a lit sphere at a point shows the proper illumination for the corresponding surface normal. Lighting a point on an arbitrary object with a given normal can be approximated by the illumination of the point on a lit sphere with the same normal. (Figure 12) The main advantage of this method is speed: the illumination is independent of the complexity of the lighting on the sphere.

Diffuse and specular spheremaps of a white sphere are prepared using rendering, compositing or paint tools. When applied to a normal image, these maps produce diffuse and specular lighting images, which are then scaled by base-color and specular-color ink and paint images. Finally, the diffuse and specular components are summed to produce a lit image.

Figure 11 illustrates the application of illumination to a drawn character. In Figure 11a, the normal image is diffusely illuminated by five lights via a spheremap. Figure 11b shows the cat rendered with a standard base-palette. The product of the diffuse illumination and base-palette image is shown in Figure 11c. Adding specular illumination and further color-styling the elements into the scene results in Figure 11d.

NPR Non-photorealistic rendering techniques that use normal or gradient information from a surface can be adapted to use approximated normal images. Gradients can be taken from illumination information and used for deriving stroke directions [Haeberli 1990].

Three light sources contribute to the illustration of the teapot in

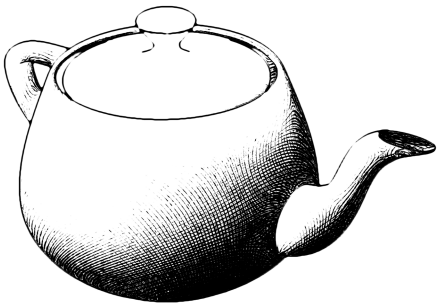


Figure 13: **Gradient-derived crosshatched illustration.**

Figure 14: **Captured Lighting, "Balloon Inflation."**

Figure 13: one provides illumination information, and crosshatch strokes are extracted from the gradients of the remaining two. The method for generating the hatching is an image-based version of the algorithm described in [Apodaca and Gritz 2000]. N.B.: when gradient information is taken from illumination data, rather than a parameterized surface, the results will slide on objects in motion.

Captured lighting Sphere mapped normals can sample an environment directly. The balloon photographed on the left in Figure 14 is transformed into the teapot on the right by using the balloon image as a spheremap. The natural light captured on the balloon transfers the material characteristics to the teapot. Advanced rendering techniques with light probes and irradiance maps can also be applied.

5 Results

The still artwork and animated examples demonstrate how the weighted blending of normal approximations drives the illumination of drawn artwork as dimensional curved surfaces. The resulting lighting transforms flat regions of paint into nuanced areas of color much more subtly than traditional tone mattes. The shading achieved with *Lumo* effectively integrates drawn character and effects elements into live-action. Characters appear to move within the live-action environment more convincingly than earlier efforts.

The technique may also be used within traditional animation to augment 2D methods for placing characters in painted or rendered environments. In recent features, the backgrounds and effects elements have visually outpaced the character treatment. *Lumo* can help bridge this gap to unify the art direction across all elements.

Animated television programs and commercials could be given a fuller look with only a minimal increase in cost. When clients request "CG," they are often really asking for a visual style, rather than a technique. For projects that demand the subtle performance of hand-drawn animation, *Lumo* can be used to make 2D look "more like CG."

Adapting 3D rendering techniques into compositing operations allows the lighting of objects to be performed in a 2D pipeline, in concert with other color design work. Outside of animation, this allows the *Lumo* technique to be used by graphic designers and illustrators to create illuminated artwork without resorting to 3D tools to build and light geometric models.

The technique is both useful and cost-effective. There is an incremental cost associated with the ink and paint department assuming an additional step, and illuminating while compositing requires more time than color-styling with multiple palettes. However, these additional costs are offset by the savings accrued from relieving the effects department of having to draw tone mattes.

6 Future Work

The assumption that surfaces are curved is the method's biggest limitation. Developing tools to aid in the assignment of normals to hard edges would expand the range of objects the system can handle.

A faster diffusion solver for sparse interpolation will hasten the adoption of the technique into production systems. The explicit iterative scheme works and is quick to implement, but for faster convergence, an implicit solution would be more productive.

It may be possible to construct geometry with more accurate depth information than the balloon inflation method in Petrovic's work by integrating the approximated normals and applying the result to both illumination and shadows.

Methods for assigning textures to implicit and subdivision surfaces might adapt to work on drawn objects. Sparse interpolation can derive pseudo-parametric coordinates across an object where "control points" have been drawn as part of the imagery and tagged in ink and paint. Reparameterizing the texture space based on the implied curvature of the normal approximation may reasonably stretch and compress a texture on the object.

A broad spectrum of rendering techniques for the application of distant illumination through acquired imagery, such as Ramamoorthi and Hanrahan's recent work [2001], is directly applicable to the rendering of normal images. Also, current generation video graphics hardware can be employed to accelerate the lighting process.

Ultimately, it is getting the tools into the hands of artists that will realize the potential of the technique.

7 Acknowledgments

I'd like to thank Charles Solomon for his support and animation expertise, Allison Abbate, Tony Cervone, Chris "Spike" Brandt, Dan Crane, Oliver Unter Ecker and Anthony Cianciolo at Warner Bros. Animation for their many contributions to this work, Tad Gielow and Tasso Lappas for reviewing early drafts of this paper, and especially Eric Goldberg for graciously providing artwork for the illustrations.

References

APODACA, A. A., AND GRITZ, L. 2000. *Advanced RenderMan: Creating CGI for Motion Pictures*. Morgan Kaufman, San Francisco.

CORREA, W. T., JENSEN, R. J., THAYER, C. E., AND FINKELSTEIN, A. 1998. Texture mapping for cel animation. *Computer Graphics 32*, Annual Conference Series (Aug.), 435–446.

GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. F. 1996. The lumigraph. *Proceedings of ACM SIGGRAPH 1996 30*, Annual Conference Series, 43–54.

Figure 15: **Cel animation integrated into live-action.**

HAEBERLI, P. 1990. Paint by numbers: Abstract image representations. *Computer Graphics 24*, 4 (Aug.), 207–214.

IGARASHI, T., MATSUOKA, S., AND TANAKA, H. 1999. Teddy: a sketching interface for 3D freeform design. *Computer Graphics 33*, Annual Conference Series, 409–416.

MARKOSIAN, L., KOWALSKI, M. A., TRYCHIN, S. J., BOURDEV, L. D., GOLDSTEIN, D., AND HUGHES, J. F. 1997. Real-time nonphotorealistic rendering. *Computer Graphics 31*, Annual Conference Series (Aug.), 415–420.

MEIER, B. J. 1996. Painterly rendering for animation. *Computer Graphics 30*, Annual Conference Series, 477–484.

PERLIN, K. 1985. An image synthesizer. *Computer Graphics 19*, 3 (July), 287–296.

PETROVIC, L., FUJITO, B., WILLIAMS, L., AND FINKELSTEIN, A. 2000. Shadows for cel animation. In *Proceedings of SIGGRAPH 2000*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 511–516.

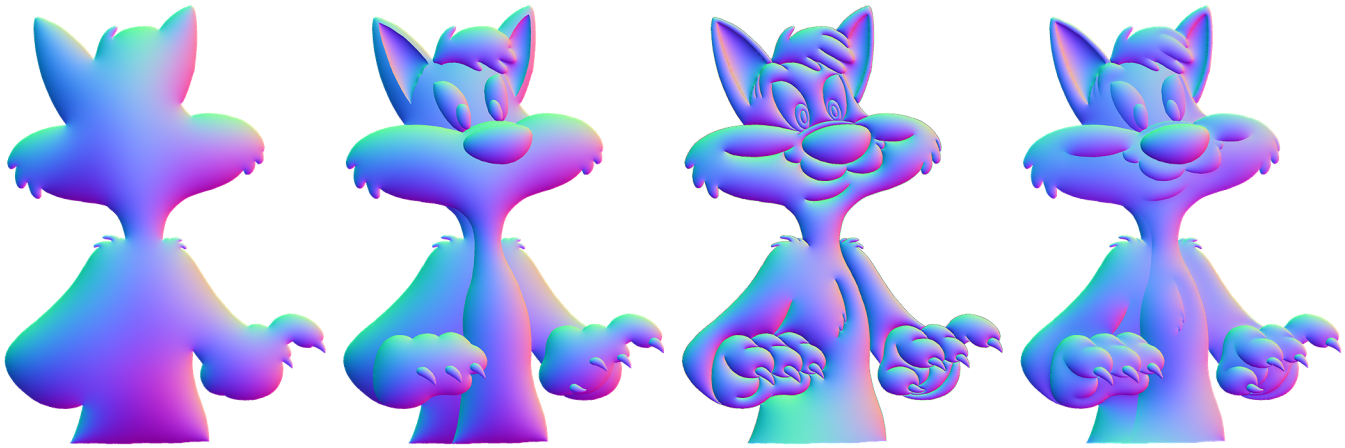
PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T., AND FLANNERY, B. P. 1992. *Numerical Recipes in C: The Art of Scientific Computing, Second Edition*. Cambridge University Press, Cambridge, United Kingdom.

RAMAMOORTHI, R., AND HANRAHAN, P. 2001. An efficient representation for irradiance environment maps. In *Proceedings of SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 497–500.

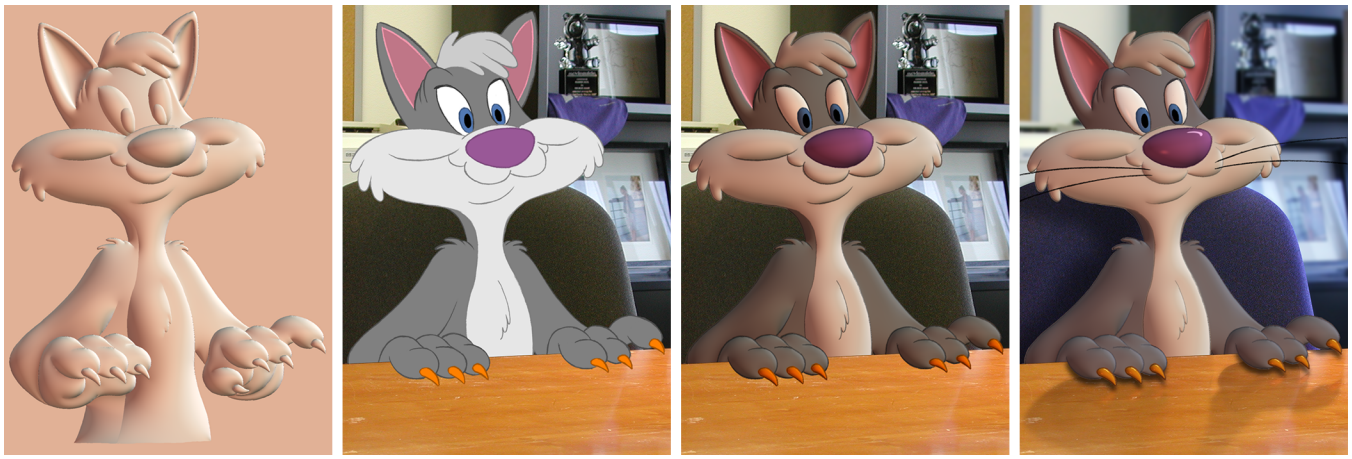
SAITO, T., AND TAKAHASHI, T. 1990. Comprehensible rendering of 3-D shapes. *Computer Graphics 24*, 4 (Aug.), 197–206.

SOLOMON, C. 1989. *Enchanted Drawings: The History of Animation*. Knopf, New York.

ZELEZNIK, R. C., HERNDON, K. P., AND HUGHES, J. F. 1996. SKETCH: an interface for sketching 3D scenes. *Computer Graphics 30*, Annual Conference Series, 163–170.



Normal Approximation. Simple blobby, compound blobby, quilted, and blended normals.



Applied Illumination. Normal image illuminated by a diffuse spheremap, base palette cat, cat scaled by illumination, final cat.

Cel animation integrated into live-action.