An image-based shading pipeline for 2D animation

Hedlena Bezerra, Bruno Feijo, Luiz Velho

An image-based shading pipeline for 2D animation

Hedlena Bezerra[1] Bruno Feijó[1] Luiz Velho[2]
[1]PUC-Rio – Pontifícia Universidade Católica do Rio de Janeiro
Caixa Postal 38097 Rio de Janeiro, RJ, Brazil
{hedlena, bruno}@inf.puc-rio.br
[2]IMPA – Instituto de Matemática Pura e Aplicada
Estrada Dona Castorina 110 22460-320 Rio de Janeiro, RJ, Brazil
lvelho@impa.br

Abstract

Shading for cel animation based on images is a recent research topic in computer-assisted animation. This paper proposes an image-based shading pipeline to give a 3D appearance to a 2D character by inspecting the hand-drawn image directly. The proposed method estimates normal vectors on the character's outline and interpolates them over the remaining image. The method does not limit the animator's creative process and requires minimal user intervention. The resulting shading pipeline can be easily applied to photorealistic and non-photorealistic 2D cel animation. In the proposed method, the animator can easily simulate environment reflections on the surface of 2D reflecting objects. As far as the authors are concerned, the proposed technique is the only one in the literature that is genuinely an image-based method for 2D animation.

Keywords: cel animation, non-photorealistic rendering, image-based shading, computer-assisted animation.

1. Introduction

Traditional animation [1], sometimes called cel animation, hand-drawn animation, cartoon animation, or 2D animation, is the most popular form of animation. The process of producing a 2D animation sequence can be roughly divided into two major stages: the drawing process and the ink and paint process. The drawing process can also be roughly broken into two sub-stages: (i) experienced animators draw the extreme poses of the character, which represent the main course of action; (ii) less experienced animators draw the inbetween drawings between the extreme poses. The main steps in the ink and paint process are: (i) drawings are traced onto acetate cels in ink; (ii) gouache or a similar type of paint is applied to the reverse side of the cels, producing paint regions, in which rims and tones are added to give the illusion of volume and illumination. A first major problem in 2D animation is to keep the frame-to-frame coherence of the drawing volumes (e.g. the 4 legs of a tiger) and the painted strokes (e.g. the stripes of a tiger) as would be in a 3D character. Another main problem is to illuminate the 2D character as it were a 3D object living in a 3D illuminated environment (called the shading problem). The origin of these problems is the fact that the

animator's drawings and paints are really two-dimensional projections of 3D characters as visualized in the animator's mind. These problems become even bigger when computer-assisted animation systems [2, 3] are used. Ed Catmull [4], in 1978, was among the first ones to discuss the frame-to-frame coherence in computer-assisted animation. The shading problem in computer-assisted 2D animation is a much recent concern [5, 6, 7, 8, 9], although some related works could be traced back to the beginning of the 90's [10]. S. Johnston [8] discusses the shading problem very clearly, indicating that the components involved in the illumination of a point on a surface (position and surface normal) are incomplete in hand-drawn artwork - the surface normal is unknown and the position information lacks depth.

Computer assistance for traditional animation can be considered part of the new area of Non-photorealistic Rendering - we recommend a visit to the Web page by Craig Reynolds [11]. Current research on computer assistance for traditional animation focuses on two main lines: geometry-based methods and image-based methods. In the first, geometric objects are created in order to support the animation process. The latter uses image-based techniques to render the 2D character without jumping to 3D geometry. The great advantage of the geometry-based methods is the possibility of supporting both the frame-to-frame coherence and the shading process. However, these methods cannot cope with more fluid animations, where few strokes would change the implied geometry drastically. On the other hand, image-based methods work directly with the vivid drawings made by the artist.

This paper proposes an image-based shading pipeline to give a 3D appearance to a 2D character by inspecting the hand-drawn image directly. The goal is to minimize the user intervention by extracting information based only on the 2D image. We show that the character's outline is the only information needed to infer a 3D aspect through an illumination model. The proposed method estimates normal vectors on the character's outline and interpolates them over the remaining image. As far as the authors are concerned, the proposed technique is the only one in the literature that is genuinely an image-based method. The proposed technique permits animators easily obtain the illusion of lighting in 2D animation that: does not require 3D input; avoids scene transformation to vectorial space; and automatically enhances depth and position perception.

This paper is organized as follows. Section 2 reviews the previous works. Section 3 presents a pipeline to process 2D images as a preparation for shading effects. Section 4 describes how rendering techniques can be adapted to use 2D position and approximate normals to illuminate a drawing. Section 5 describes how to simulate an environment reflection in a geometric 2D object. Section 6 concludes with final remarks and a discussion of ongoing work.

2 Previous work

Most of the prior work in this area relies on reconstructed geometry. Igarashi *et al.* [6] create a surface from 2D information, providing a gesture-based sketching interface to quickly and easily design freeform models. Correa *et al.* [5] take a geometry model and deform it to match a hand-drawn figure. Fiore & Reeth [9] generates approximate 3D models in order to help animators keep frame-to-frame coherence. Pretovic *et al.* [7] presents a scheme for "inflating" a 3D figure based on hand-drawn art to create shadows for cel animation.

The present research was inspired on the seminal work by Scott Johnston [8] that uses image-based techniques. He also estimates normal vectors and uses sparse interpolation to approximate them over the remaining image to illuminate 2D drawings. However, the present work has major conceptual and technical differences. The main drawbacks of the Johnston's method are the need of multi-channel images and the need of scene transformation to vectorial space. In our model, we work directly from the 2D image and we avoid transformations to vectorial space completely. This is the reason why we claim that our method is the only one in the literature that is genuinely an image-based method.

3. Guessing 3D surfaces from 2D images

This section describes a pipeline to process 2D images in order to infer a 3D surface from drawings outlines. Our pipeline is suitable to cel-based animation and is intended to minimize the amount of user intervention. Figure 1 depicts pipeline main stages.

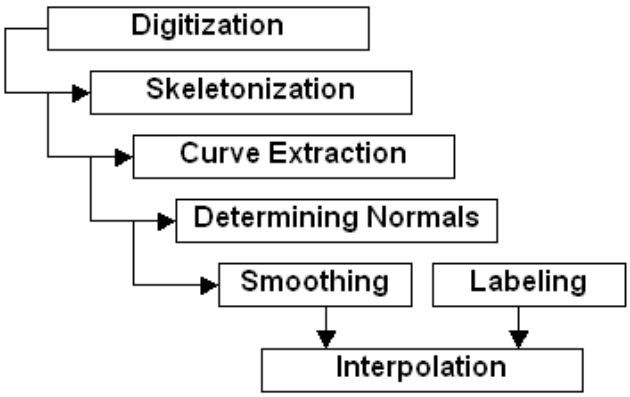

Figure 1. Pipeline main stages.

3.1. Digitization

In the traditional animation process, animators begin by drawing sequences of animation on a paper with pencils, one picture or ”frame” at a time. The pipeline’s input data are images made by the traditional cel animation production process. Those images are scanned from cel by a digital process. It is assumed that original drawings contain homogeneous, non-textured lines drawn onto a white background having black outlines. These requirements make the images cleaning process easier because, in the final step, it will also have black line drawings in a white background.

After the scanning process, each cel has registration holes. Those small holes allow the cel to be placed on pegs to ensure each cel aligns with the one before it. If the cells are not aligned in such a manner, the animation, when played at full speed, will look ”jittery.”

3.2. Skeletonization

In images consisting mainly of lines, it is the position and orientation of the lines and their relationship to one another that conveys information. Extra pixels are unwanted information that complicate the analysis, and removing them is desirable [12]. To facilitate the extraction of image features, a thinning algorithm erodes, therefore reducing objects without changing their topology. For example, consider the bear images in Figure 2, both of them have the shape of the same cute bear, but the image on the right has the extra pixels removed and can now be further analysed for shape.

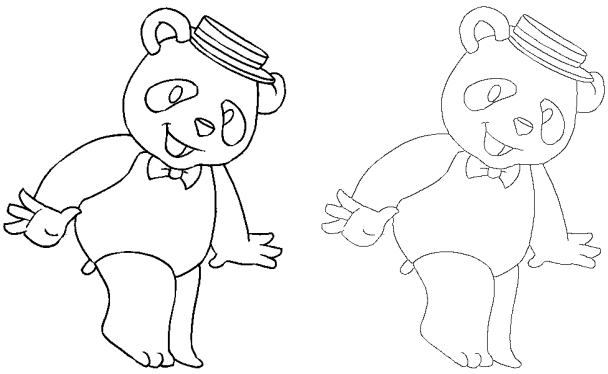

Figure 2. The skeletonization of the figure on left removes extra pixels and produces a simper image (right).

3.3. Curve extraction

This stage implements contour decoding. An arbitrary curve is composed from a sequence of unitary vectors with a limited set of possible directions. Once a skeleton has been extracted, the image curves can be well represented by storing only the direction to the next pixel for each of the connected pixels in the boundaries. A chain code method using an 8-connected grids models its layout and direction.

3.4. Determining Normals

Once the image curves were extracted, a 2D normal vector can be easily derived at any point in that representation, by taking the perpendicular direction to the curve orientation. The normals can be created in any space, but it is convenient to transform them to a screen space (X, Y) with depth Z. For an ortographic projection, the z-component of the normals along the edge of a curve surface must be zero. This is done to keep the normals perpendicular to the eye vector.

A point in the curve may have two normal vectors that share the same direction, but have opposite orientations. The direction of this vector determines the orientation of the surface. For example, if the normals of a circle point outwards its trace, interpolating those vectors across the circle interior area generates a field of normal vectors that determines a convex surface. On the other hand, if the inner orientation is chosen, the interpolation determines a concave surface. Our method guesses, for each segment, which is the best orientation by inspecting its dominant curvature, and by choosing the one that points outwards of the curve as shown in Figure 3.

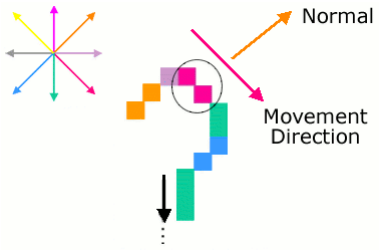

Figure 3. Derived normal vectors.

3.5. Smoothing

The chain code used models a limited range of vectors as there are just eight possible directions for neighbors. A smoothing stage low-pass filters the normal fields produced in the 3.4 section as follows: for each point, a normal vector is calculated by adding its previous and next neighbor’s vectors. Figure 4 depicts a normal vector field calculated over the bear’s outline image.

Figure 4. Normalized smoothed normal vectors in RGB space.

3.6. Labeling

Labeling, a common technique in computer vision, groups the pixels of an image into nonoverlaping regions based on some connectivity criteria [17]. In this paper, we present a labeling technique that partitions an image through a flood fill algorithm; in order to indicate which region of the image the interpolation must be applied on.

In outline images, each region is made up of pixels inside a boundary curve. Given a point in an image, a flood fill algorithm recursively colors all pixels which are adjacent to this point and have similar attributes. When applied to binary images, the algorithm finishes when it finds all pixels internal to a region's boundary. If we scan a binary image running this algorithm to non-colored pixels, we are able to extract all the image regions. As colored pixels are not considered in the scanning process, this approach has an efficient $O(n)$ cost, where n is the number of pixels. The Figure 5 depicts an image labeling process where each area is colored with a different color.

Figure 5. Image labeling through a flood fill algorithm.

3.7. Interpolation

Using estimated normals, an interpolation stage employs sparse interpolation to approximate normals over the remaining image. This phase is based on Johnston's work [8] which approximates normals vectors in multi-channel images. Rather than solving the complex problem of interpolating normals as multidimensional unit-length vectors, N_x and N_y are interpolated as independent components, and N_z is recomputed to maintain unit length. Thus, known pixel values remain fixed and unknown values are relaxed across the field using dampened springs between neighboring pixels. Given a field P to be interpolated, and a velocity field V initialized to zero, each iteration is computed by:

V'i,j = d.Vi,j + K.(Pi-1,j + Pi+1,j + Pi,j-1 + Pi,j+1 + 4P)
P'i,j = Pi,j + V'i,j
(1)

Johnston [8] shows that optimal values for the dampening and spring constant vary, but $d = 0.97$ and $k = 0.4375$ minimizes the convergence time for a circle example. Iterations are performed until the mean-squared velocity per pixel reaches an acceptable error tolerance. To adjust region's puffiness, he suggests to replace a normal with the normal of a sphere that has been scaled in z. 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:

$$\frac{(S.N_x, S.N_y, N_z)}{\|(S.N_x, S.N_y, N_z)\|} \tag{2}$$

Figure 6 depicts a normal field image in RGB space obtained from the outline drawing depicted in Figure 2 left. The normal field approximates a spherical shape for the bear image and provides it with a 3D appearance. Using image-based algorithms, it is possible to approximate the surface normals and directly avoid image transformation to 3D geometry. Our method is suitable for prototyping image-based shading techniques. For maximum control, standard 3D rendering works best.

4. Shading techniques

Shading is fundamental in the process of conveying a three-dimensional structure for an object in a two-dimensional image [13]. Researchers in non-photorealistic rendering (NPR) have investigated a variety of techniques to simulate the nonlinear shadings found in art works. Using 2D positions and approximated normals, most rendering techniques can be adapted to illuminate a drawing [8]. In this section, we demonstrate how shading techniques can be used to infuse hand-drawn artwork with not only photorealistic but also non-photorealistic vitality and beauty.

Figure 6. A normal image in RGB space.

Figure 7. Lake's cartoon shading approach applied to a 2D animation frame.

4.1. Cartoon shading

Cartoon characters are intentionally "two-dimensional". Animators deliberately reduce the amount of visual detail in their drawing in order to draw the audience into the story and to add humor and emotional appeal [14]. L. Willian [10] shows that one critical simplification which made traditional cel animation possible and practical is that objects and characters are represented by line drawings filled with areas of solid color. Details have to be used sparingly and consistently in order to generate hundred of thousand of drawings for an animation. In a cartoon shading, this economy leads the artist to shade part of a material that is in the shadow with a color that is a darkened version of the main material color.

Lake *et al.* [14] developed a cartoon shading algorithm similar to the cel animator's process of painting an already inked cel. Instead of calculating the color per vertex, Lake *et al.* generates a texture map of colors. In most cases, the number of colors is two: one for the illuminated color and one for the shadowed color. The boundary between lit and shadowed colors on the model depends on the cosine of the angle between the light and normal vectors. Disadvantage is the need to create 3D input models.

If the animator informs the light position, we demonstrate that using an approximated normal vector field, 3D approaches for cartoon shading techniques can be applied on 2D images. The Figure 7 depicts the result of applying Lake's *et al.* approach to a 2D character where its surface normal was obtained through our preprocessing 2D images pipeline.

Figure 8. Images illuminated by a Phong shading. On the right image, silhouette and outline details are added.

4.2. Phong shading

The Phong illumination model [15] is probably the most applied model in computer graphics nowadays. In this model, the light in any point is composed by three components: *ambient*, *diffuse* and *specular*. Those components are additive and give the color and illumination final appearance of a scene.

The primary components typically used to illuminate a point on a surface are its position and surface normal. With hand-made drawings, the surface normal is unknown, and the position lacks depth. However, we demonstrate that a Phong shading may be applied in 2D images to create a realistic style for animation films if approximated normal vectors are used. This model may give a strong tridimensional appeal by approximating the incident illumination on curved surfaces as showed in the Figure 8. If silhouette and outline details are added, the result looses the artificial appearance often found in computer generated imagery (Figure 8 right).

Figure 9. A sphere mapping technique applied to a 2D circle image(left). The image composed into an animation scene(right).

5. Sphere mapping

In recent years, the graphics community has made due progress in 3D animation which has given to the animator a variety of facilities to apply an image over a surface. This feature improves the production process and the final appearance of a drawing. Animators often use a texture to simulate environment reflection in a geometric object. In computer graphics, a sphere mapping technique enables any arbitrary pattern in the two-dimensional space to be reflected on a spherical surface. This technique approximates the effect of the environment reflecting on the surface of a highly reflective object.

The advantage of using a sphere mapping technique instead of a texture mapping in animated films is that the texture changes according to view point changes. In the case of texture mapping, the (u, v) coordinates are a function of the object only. In the case of sphere mapping, the (u, v) coordinates are a function of the objects position, surface normal and its orientation with respect to the eye.

There are several algorithms to apply sphere mapping techniques to a surface like [16]. Once again, disadvantages are the need of using 3D input models. However, we demonstrate that a sphere mapping technique can be applied to 2D images using approximated normal surfaces. Figure 9 depicts the result a circus environment reflexion into a circle where the surface normal was approximated through the pipleline described in this work. Composing the result imagem to the animation scene we get a realistic effect of a reflecting material. Using our technique, tradidional animators can easily simulate environment reflecting into the surface of reflecting objects.

To enhance properties of a reflecting material, such as chrome, illumination model with diffuse and specular components must be applied to each image pixel. The Figure 10 illustrates a circus evironment reflexion into a circle that simulates chrome material and reflexion properties.

Figure 10. A circle simulating a chrome material and reflexion properties.

6. Conclusions

Although the flat look of the 2D animations could be an intention of the artist, cel animators experience great difficulties in generating hundreds of thousands drawings with 3D appearance. Although computer assistance for traditional animation is gaining a lot of attention in recent years, the 2D shading problem is far from being completely solved. The goal is still to require minimal user intervention and do not limit the animator's creative process. Image-based methods seem to be a good road to achieve this goal. The present paper presents a method that is the only one in the literature that is genuinely an image-based method, because it works directly from the 2D image and requires no transformation to vectorial space. Animators do not like vectorization because drawing lines are usually modified during this process. The technique proposed by Johnston [8] has some drawbacks, because the animator needs a support system to build the multi-channels; moreover, the identification of each region of the image is made by the animator.

The following comparisons can also be done with Johnston's work [8]: the way our method stores each curve

(chain code) makes easy the task of finding normal vectors at each point (there is no need for using gradients as in [8]); the animator needs only to select the curve and set the orientation parameter in order to define the orientation of the normal vectors (there is no need for "light over dark" lines along each curve as in [8]); the segmentation process (labeling) is automatic, apart from the selection of the final color of each region (this process is not automatic in [8]). There is also a difference towards the objectives of the system: our method is for traditional cartoon animation, while Johnston [8] envisages the blending of 2D animation and live action as in "Who Framed Roger Rabbit" (1998) [18].

Further work will be carried out in building a framework to visual creation and edition of computer assisted animations. Through this framework, the animator will work in interactive high level programming environment which provides a fast adapted concept of: (i) shade engines (e.g. Perlin's Pixel Stream Editor [19]), (ii) shading languages (e.g. Cook's shade Trees [20] and Pixar Renderman [21]), (iii) real-time previews using normal look-up tables, (iv) real-time shadow effects by changing light position, (v) new styles application by learning analogies (e.g. Aaron Hertzmann *et al*. [22] Image Analogies) and (vi) creation of a styles library.

Acknowledgments. The authors thank César Coelho for all lovely bear outlines and his support as a renowned traditional animator. We also would like to thank Ives Macedo Júnior and Marcelo Vieira for many helpful discussions and comments regarding the present work. This work was developed at ICAD/VisionLab (PUC-Rio) - sponsored by CNPq and FINEP - and VISGRAF (IMPA) - sponsored by CNPq, FAPERJ, FINEP, and IBM Brasil. The first author is partially supported by scholarships from PUC-Rio and CAPES. The second author's research work is under research contracts CNPq Grant PQ No. 305982/2003-6, SEPIN-CNPQ-FINEP No. 01.02.0230.00 (Ref. 2425/02) and FINEP No. 01.04.0945.00 (Ref. 3110/04).

References

[1] P. Blair. Cartoon Animation. Walter Foster Pub. Inc., 1994.

[2] J. D. Fekete, E. Bizouarn, E. Cournarie, T. Galas and F. Taillefer. TicTacToon: A paperless system for professional 2D animation. In Proc. of SIGGRAPH'95, p 79–90, 1995.

[3] H. Griffin. The Animator's Guide to 2D Computer Animation. Focal Press, 2001.

[4] E. Catmull. The problems of computer-assisted animation. In Proc. of SIGGRAPH'78, p. 348–303, 1978.

[5] W. T. Corrêa, R. J. Jensen, C. E. Thayer and A. Finkelstein. Texture Mapping for Cel Animation. In Proc. of SIGGRAPH'98, p. 430–446, 1998.

[6] T. Igarashi, S. Matsuoka and H. Tanaka. Teddy: a sketching interface for 3D freeform design. In Proc. of SIGGRAPH'99, p 409–416, 1999.

[7] L. Petrovic, B. Fujito, L. Williams and A. Finkelstein. Shadows for cel animation. In Proc. of SIGGRAPH'00, p. 511–516, 2000.

[8] S. F. Johnston. Lumo: Illumination for cel animation. In Proc. of NPAR 2002, p. 45–52 and p. 156, 2002.

[9] F. Di Fiore and F. Van Reeth. Employing Approximate 3D Models to Enrich Traditional Computer Assisted Animation. In Proc. of Computer Animation, p. 183–190, 2002.

[10] L. Williams. Shading in two dimensions. In Proc. of Graphics Interface, p. 143–151, 1991.

[11] C. Reynolds. Stylized Depiction in Computer Graphics – Non-Photorealistic, Painterly and 'Tonn Rendering, www.red3d.com/cwr/npr [accessed in 12/May/2005].

[12] J. R. Parker. Pratical Computer Vision using C. John Wiley and Sons Inc., 1994.

[13] Bruce Gooch and Amy Gooch. Non-Photorealistic Animation Rendering. A K Peters Ltd., 2001.

[14] A. Lake, C. Marshall, M. Harris and M. Blackstein. Stylized Rendering techniques for scalable real-time 3D animation. In Proc. of NPAR 2000, p. 13–20, 2000.

[15] B. Phong. Illumination for computer-generated pictures. Communications of the ACM, V.18 #3 p. 311–317, 1975.

[16] L. Velho and J. Gomes. Sistemas Gráficos 3D. Instituto de Matemática Pura e Aplicada – IMPA, 2001.

[17] E. Davies. Machine Vision: Theory, Algorithms and Practicalities. Academic Press, 1990.

[18] C. Solomon. Enchanted Drawings: the History of Animation. Knopf, 1989.

[19] K. Perlin. Image analogies. In Proc. of SIGGRAPH'85, pp 287–296, 1985.

[20] Robert L. Cook. Illumination for computer-generated pictures. In Proc. of SIGGRAPH'84, p. 223–231, 1984.

[21] Pixar Animation Studios. Pixar's RenderMan Products, https://renderman.pixar.com/ [accessed in 15/May/2005]

[22] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless and D. H. Salesin. Image analogies. In Proc. of SIGGRAPH'01, p. 327–340, 2001.