In Proceedings of the IEEE International Conference on Information Visualisation, July 19–21, 2000, London, England, pages 413–418, Los Alamitos, California, 2000. IEEE Computer Society, The Institute of Electrical and Electronics Engineers, Inc.

3D Illustrative Effects for Animating Line Drawings

Tobias Isenberg Maic Masuch Thomas Strothotte

Department of Simulation and Graphics
Otto-von-Guericke University of Magdeburg
{isenberg|masuch|tstr}@isg.cs.uni-magdeburg.de

Abstract

Most illustrative techniques used in non-photorealistic rendering to date apply a rendering style to objects or a scene and alter the appearance of this style by employing illumination or depth cueing. However, for generating high quality illustrations this approach relies almost entirely on smart placement of light sources.

In this paper we introduce the concept of illustrative effects *to describe the workings of illustrative techniques that rely on spatial location rather than just illumination or depth. Although the rendering style is still object-dependent, this technique allows us to visualize or emphasize parts of objects or the whole scene without being limited by the scene's object decomposition. The approach also enforces the decoupling of the model and its animation from the visualization task.*

We demonstrate our approach using a sample collection of illustrative effects applied to line drawings. A uniform set of tools to manipulate these effects is provided.

Keywords: Non-photorealistic rendering, non-photorealistic animation, spatial illustrative effects, computer-generated line drawings

1. Introduction

In non-photorealistic rendering numerous different rendering styles have been developed in recent years (for an overview see [1]). It can be said that, in general, non-photorealistic rendering systems to date have computed the appearance of objects as a function of:

(a) the drawing style,

(b) the light intensity at an object, and

(c) the distance from the viewer/camera to the point in space being visualized.

In this paper we argue that these considerations are not rich enough to achieve the necessary repertoire of illustrative techniques as used, for example, in hand-drawn illustrations. We introduce a new method of producing spatial effects in illustrations. Our method is based on specifying a general function $f_E(x,y,z)$ to describe the desired visualization effect to points (x,y,z) of the scene, combined with a function $f_S(object)$ which determines the visualization style in which an object in the scene is to be drawn. The combination of these two functions, $f_E \circ f_S$, is used for computing the final image.

Our approach has a number of advantages over previous methods. First, it enables users to directly specify effects they wish to see in their images, rather than, for example, placing light sources and relying on a model of reflection to achieve the desired results. Second, it enforces a clear separation between model and visualization in that the illustrative effects are treated strictly in the realm of the visualization rather than, as has been done in the past, by changing parameters of the model and then relying on the physics of light to produce these illustrative effects. Finally, and most important, it enables users to generate illustrative effects which go beyond those which can be realized by simulating physical phenomena, such as related to light.

The paper is organized as follows. Section 2 describes previous work done in the field of visualization using non-photorealistic rendering and animation. In Section 3 we introduce the concept of illustrative effects and explain its advantages for generating non-photorealistic visualizations. Following the discussion of the application of this technique using computer-generated line drawings with four exemplarily designed line-style-effects and a technique for the combination and animation of such effects in Section 4, a prototypical implementation of these techniques in an animations system for line drawings is discussed in Section 5. Finally, Section 6 gives a summary and conclusion of the methods and techniques presented in this paper.

2. Related Work

The ultimate goal of most of computer graphics is still to simulate the physics of illumination, reflection, and refraction in order to get as close to reality as possible. Thus, in such photorealistic graphics, to create appealing visualizations, various light sources have to be placed

within the scene to enable emphasizing certain parts of the model. This, however, can be a tedious task. POULIN and FOURNIER even try to calculate the positions of light sources from highlights and shadows in the picture as determined by the user [2].

In non-photorealistic rendering, however, there is no such ideal as reality. Instead, the communicative goal determines the selection of techniques. A non-photorealistic image can differ from a photo in shape, color, texture, light, and shadow. Such a rendition can be simplified by the abstraction of details—unnecessary or distorting detail is left out in order to focus the viewer's attention on the inherent message [3, 4]. In addition, certain parts of the depicted scene can be emphasized and de-emphasized by the variation of the drawing style, e. g. using bold lines for important objects [5].

This potential of abstraction in non-photorealistic rendering allows the creator of such images to concentrate on the visualization task at hand rather than on finding a smart placement of various light sources that result in an adequate rendering. In the past, this was typically done by assigning different rendering styles to different parts of the model to emphasize or de-emphasize certain elements.

A technique equivalent to depth cueing to be applied to non-photorealistic renderings was introduced by ELBER [6]. He uses depth information to indicate spatial relations in line drawing illustrations with techniques such as *haloed lines*.

An additional way of improving the illustrative character of non-photorealistic renderings is to add animation [7]. A number of different approaches to non-photorealistic animation have been introduced so far. However, all of them are either 2D-oriented [8, 9] or concentrate on the frame-coherent animation of one rendering style [10, 11]. Thus, they make no use of the available 3D information for improving the visualizational properties of the animation.

3. Spatial Manipulation

3.1. Illustrative Effects

Traditional photorealistic rendering generally relies on the simulation of physics to achieve realistic results. This means that the appearance of a point within the scene in the final image can be viewed as determined by a function $f_{PR}(x,y,z)$ of its location in space. This function takes into account the illumination of the point, texturing of the object it belongs to, and other things that make up the photorealistic model.

In contrast to photorealistic rendering, the appearance of the resulting image in non-photorealistic rendering is mainly determined by a rendering style that is assigned to the objects in the scene. The appearance of this style at a certain point is usually altered only by considering the illumination of the scene and the distance of objects from the viewer. Hence, the function to compute the appearance of a point can be described as a combination of depth, illumination, and style: $f_D(x,y,z) \circ f_I(x,y,z) \circ f_S(object)$.

However, this technique is very limiting since it only employs two ways of altering the rendering style. Furthermore, these two functions are directly derived from photorealistic rendering. We believe that these two functions are by far not powerful enough to produce high quality visualizations. Instead, we suggest using a general function $f_E(x,y,z)$ to describe the alteration of the appearance of the rendering style which we call *illustrative effect*. Since just one effect is often not enough to achieve the visualization goal we allow as many illustrative effects to be used as necessary. Thus, considering f_E to be a combination of many f_{E_i}, the resulting function to produce the final appearance of a certain point is: $f_E(x,y,z) \circ f_S(object)$.

The application of general illustrative effects allows us to employ any spatial alteration of the rendering style. Of course, with this method it is also possible to use illumination and depth cueing techniques, but it is not limited to techniques derived from photorealism anymore.

3.2. Object-Independent Manipulation

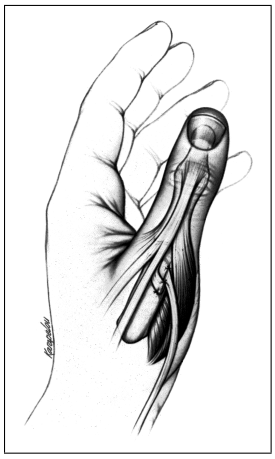


Figure 1: Hand-drawn scientific illustration [5].

Since in most cases two different rendering styles cannot be continuously blended into one another, highlighting in computer-generated illustrations has been limited to applying different rendering styles to whole objects or groups of objects only (see, for example, [11]) and thus adapting the function $f_S(object)$. Although this allows for creating illustrative images and animations, it is not very well suited for all kinds of models. In technical models, the decomposition into single parts makes it easy to assign effects to objects. In contrast, for instance in organic models, this kind of subdivision is usually not given. Generally, continuous transitions are much more common in reality than clear separations.

In [5], HODGES gives various examples for hand-made scientific illustrations, one example of which is shown in Figure 1. In the figure one can clearly see that the emphasis on the thumb does not end abruptly. Instead, the line width is smoothly reduced for lines further away from the thumb. Lines far away from the area of focus even almost disappear because they were drawn with a very small line saturation. Besides the line attributes, other parameters as well—like

the level of detail—were used in the example to increase the expressional power of the illustration.

This technique used by artists can now be simulated in computer-generated images by employing the technique of illustrative effects as introduced in Section 3.1. For emphasizing a part of a model such as the thumb in Figure 1, a corresponding effect function f_E has to be specified. Thus, by employing the object-independent approach of illustrative effects the subdivision of the underlying model into smaller objects gets obsolete.

3.3. Separation of model and visualization

In contrast to other illustration techniques, an illustration is not created by assigning different rendering styles to objects. Instead, a spatial approach is used to modify a rendering style. Because of their spatial character, the illustrative effects are independent from the model decomposition. Thus, the visualization process becomes more independent from the model structure compared to previous approaches. In particular, the adaptation of the model or its structure becomes unnecessary. Hence, the usage of illustrative effects enforces the clear separation between the model and its animation on one side and the visualization on the other and therefore enables the user to easily re-use existing models.

4. Line-Style-Effects

In this section we want to illustrate the concept introduced in Section 3 by designing a number of illustrative effects for the application in images generated in the style of line drawings. For generating the best possible quality in line drawings, we use an analytic rendering process to generate the basic illustration. This analytic process not only generates vector graphics as output, it also allows us to preserve the three-dimensional information derived from the 3D model during every step of the rendering process.

In particular, we can use the 3D data when computing the effect function f_E. These *line-style-effects* influence the line attributes—such as width or saturation—of the generated line drawings in an object-independent manner. However, instead of directly specifying an effect function $f_E(x,y,z)$, each individual line-style-effect with its parameters is equivalent to a function $t = f'_E(x,y,z)$. This parameter $t \in [0;1]$ is then used as an input for a parametric function $f_P(t)$ that can be specified by the user to further control the behavior of each effect: $f_E(x,y,z) = f_P(f'_E(x,y,z))$

In the following, four examples of line-style-effects will be presented and example renderings will be shown. The developed effects include a plane-sweep-effect, a volume-of-interest-effect, a camera-effect, and a lit-volume-effect.

4.1. Plane-Sweep-Effect

We introduce a plane-sweep-effect in analogy to the term *plane sweep* from analytic geometry. Similarly in our case, a plane sweeps through the three-dimensional space perpendicular to a certain direction that is defined by the *effect ray*. The two positions of the plane when it first and last touches the considered objects describe an *active segment* on the ray.

The function $t = f'_E(x,y,z)$ is calculated as the relative position on this active segment. In order to calculate the value of the influence at a certain point on an edge of an object, this point is first projected perpendicularly onto the ray and the relative position of the projected point on the active segment is calculated (t value). The value of $f_P(t)$ returns the resulting influence of the plane-sweep-effect.

An example of such a plane-sweep-effect is shown in Figure 2. Here, the effect ray is located along the axis of the staircase. The parametric function $f_P(t)$ is gradually changed during the animation. By shifting the point of this function where it falls from 1 to 0 over the course of the animation, it gives the illusion of a slowly appearing staircase, from bottom to top.

As demonstrated, a plane-sweep-effect imposes a one-dimensional influence on the line attributes. An effect that can produce a three-dimensional influence will be presented in the following.

4.2. Volume-of-Interest-Effect

Figures 3 and 4 show the application of a volume-of-interest-effect. To highlight certain areas in an illustration, this effect allows to specify a voluminous region around a certain point in which the lines can be accentuated.

In these particular cases, a sphere is used to describe the area of influence of the effect while the effect function is applied to the radius of this sphere. Therefore, to calculate the influence on an attribute at a specific point, its relative position on the radius of the sphere is computed which equals the value of t. The value of the function $f_P(t)$ is the resulting influence of the effect at the considered point.

Other primitives could be used in a volume-of-interest-effect as well, such as boxes. In this case, the function could be calculated by using a minimal distance function from the surface or by using a ray through the point being considered and the middle of the box as a reference.

4.3. Camera-Effect

Another approach to influencing effects is to use virtual objects like cameras that already exist in the 3D scene for visualization of the lines within the viewing volume of a certain camera (the *effect camera*). A different camera that is looking at the scene can show the visibility frustum of the effect camera, an effect that hardly can be achieved using photorealistic rendering. An application for such a visualization would be the planing of the camera movement in an animation.

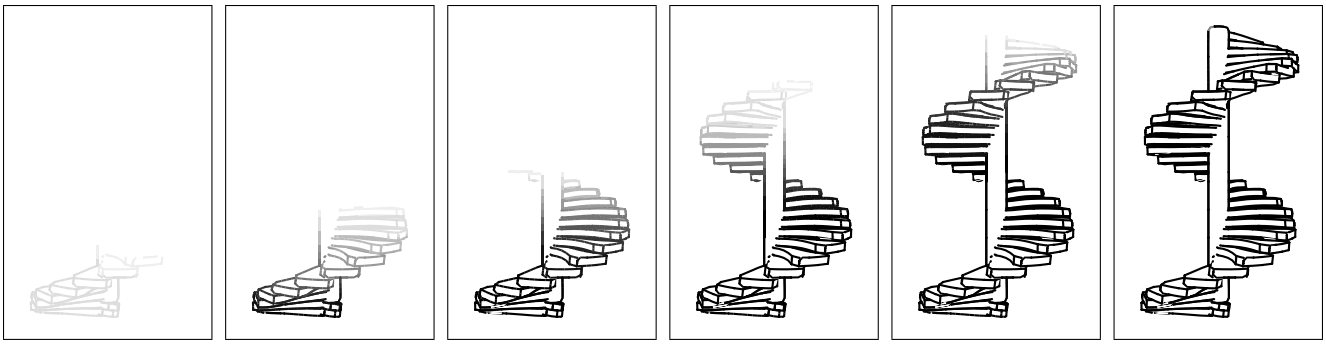
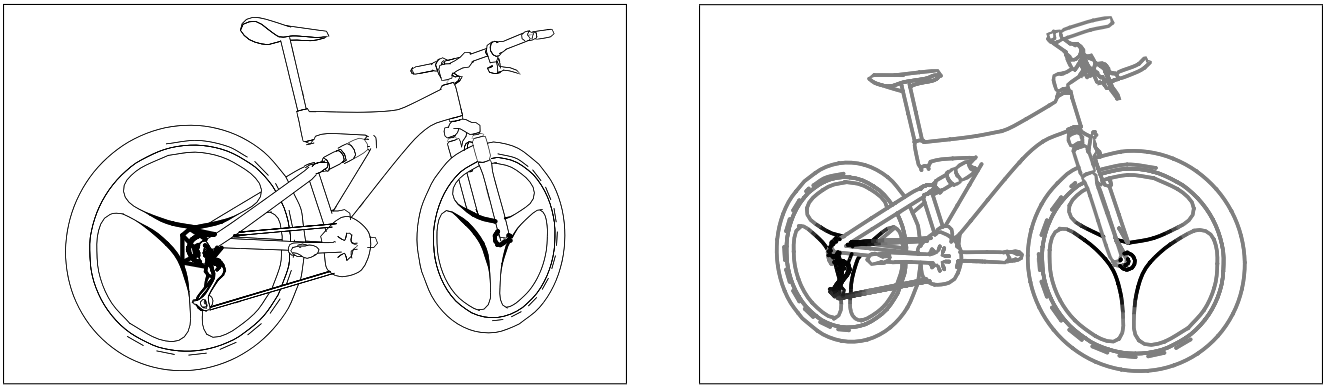


Figure 2: Visualization of a tower staircase using a single animated plane-sweep-effect with the active segment along the axis of the staircase.



(a) Influence on line width.

(b) Influence on line saturation.

Figure 3: Volume-of-interest-effect: emphasizing the stock and the chainring of a bicycle. Also, the same line-style-effect is applied to the same object using different line attributes. Note that the object-independent effect can be reduced to part of the object hierarchy only.

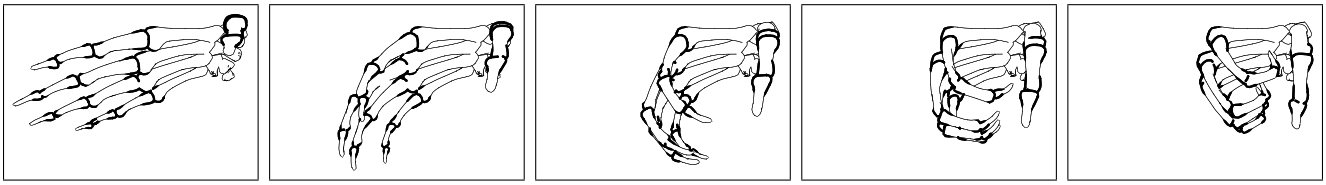


Figure 4: The volume-of-interest-effect applied in an animation of the bone model of a human hand being closed. The effect is used to emphasize the joint areas of the hand.

In Figure 5, an example for the application of the camera-effect is shown. The lines of the objects that are visible from a camera which is moving through the scene are shown by a different camera that is looking at the scene as a whole. The function $t = f'_E(x,y,z)$ is calculated as the relative position within the visible area as seen from the effect camera. Therefore, the camera-effect imposes a two-dimensional influence on the line attributes.

4.4. Lit-Volume-Effect

Similar to the camera-effect, it is also possible to visualize the influence of a light source by line drawing means, as it is common in traditional hand-drawn pictures (for an example see Figure 6). In contrast to the camera-effect, however, the lit-volume-effect has a three-dimensional influence on the attributes.

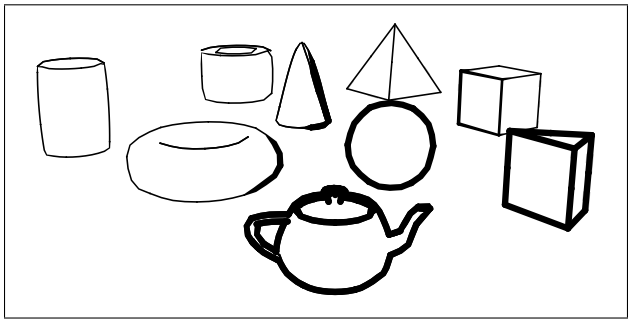


Figure 5: Visualizing the field of view of a camera using a camera-effect.

For calculating the influence of the effect at a certain point, the illumination according to the LAMBERT illumination model (i. e. diffuse illumination) is computed, then normalized for the interval $[0,1]$ and used as the input parameter t for the parametric function f_P. Alternatively, the angle between normal vector and the illumination vector can be used as a value for t.

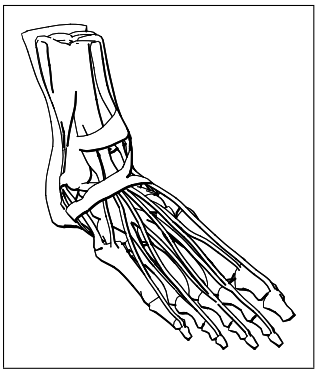


Figure 6: Impact of a lit-volume-effect on an illustration of a human foot with light coming from the top left.

Although the camera-effect described above seems to be similar to the lit-volume-effect, there are clear differences. Not only is the region different in which the attributes are influenced (pyramid vs. cone); the effect function of the former effect only imposes a two-dimensional influence, whereas it is three-dimensional for f_E of the latter. For example, using a lit-volume-effect, lines facing the light source are treated in a different way than lines facing the other direction, depending on f_P.

4.5. Combining and Animating Line-Style-Effects

For enabling the combination of more than one illustrative effect and providing means to intuitively animate these effects we use a combination of traditional keyframing (that uses hierarchically organized models) with an effect hierarchy as found, for example, in video editing systems. Besides a traditional keyframed animation of the model, this *hierarchical keyframing* (see Figure 7) uses a separate hierarchy of illustrative effects rather than assigning properties to objects within the object hierarchy.

Consequently, the keyframes only have local influence on the respective effect and cannot interfere with parameters from other effects. Although parameters from different effects can influence the same attribute, the single effects are independent and can be specified separately. Also, keyframes for the boundary (start and end) conditions of the effects are no longer necessary.

This approach enables users to specify as many effects as they want by arranging them into a hierarchy. It enables a prospective system to generate output with smoothly combined effects using different blending functions. These functions can be as simple as, for example, the maximum function. However, the maximum function works best to preserve the character of the individual effects that were combined in the final image. On the other hand, the remaining combination functions shown in the figure can nonetheless be useful to achieve artistic or illustrative results for the combination of single effects.

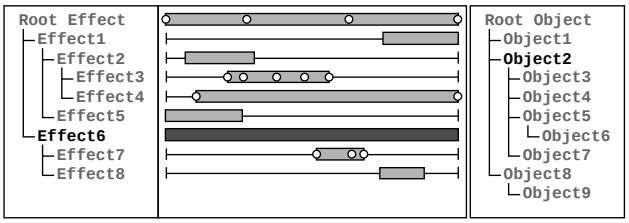


Figure 7: Hierarchical keyframing consisting of an object hierarchy as well as a separate effect hierarchy. Effects are animated by assigning local keyframes to effect parameters.

5. Line-Style-Effect Animation System

The line-style-effects presented above and the technique of hierarchical keyframing were implemented in a line-style-effect animation system using Visual Works Smalltalk. Since it is important to benefit from an existing modeler, the system uses predefined scene files that include the 3D model accompanied by the animation information modeled in 3D Studio MAX. Users can first enrich the model by constructing an effect hierarchy that is appropriate for the intended visualization goal before being able to generate an illustrative animation (see Figure 8).

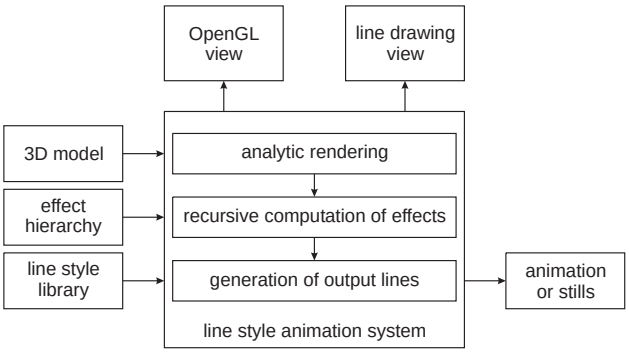


Figure 8: Schematic view of the line-style-effect animator.

In the animation system, users are presented two views: an OpenGL view and a line drawing view (see Figure 9). In the OpenGL view, users can adapt the view to the scene and can explore the scene in more detail while constructing the effect hierarchy and setting each effect's various variables. After finishing the process of constructing the effect hierarchy, the line rendering is started.

However, we noticed that there are several issues designers of illustration have to pay attention to when using the animation system. First, it pays off if background lines are assigned at least a small line width and line saturation. Otherwise, these background lines are not displayed in the resulting images which might confuse the viewer.

When creating illustrations for print media, images that only make use of line width rather than line saturation are

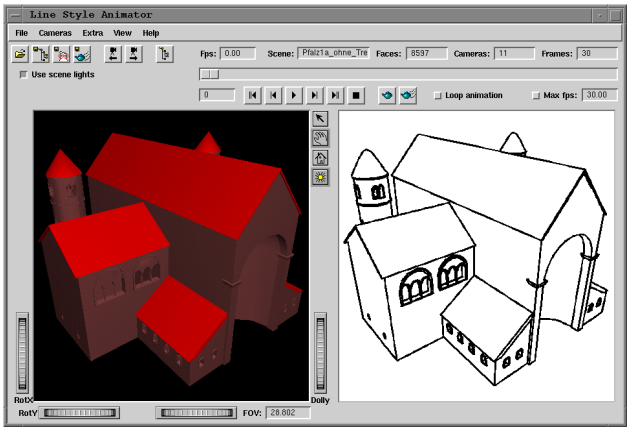


Figure 9: The line-style-effect animation system showing the two parts of the screen: the OpenGL view and the line drawing view.

better suited for rasterization reasons. A comparison of the application of the same effect on line width in one image and line saturation in another image is shown in Figure 3. For screen presentations, however, the situation is inverted. Here, illustrations that make use of different line saturations usually give better results.

A high potential of the system lies especially in the animation of the line-style-effects rather than in the animation of the objects, although the latter is possible, too. This is due to the high amount of rendering time needed for the creation of an animation using the analytic rendering. In Figure 2, the view on the staircase or the staircase itself is not animated at all. The illustrative effect is caused entirely by a continuous appearance of the steps from bottom to top, created by an animated plane-sweep-effect. For this animation the rendering had to be done only once to generate the output lines, for every subsequent frame only the changed influence on the line width had to be calculated.

6. Concluding Remarks

In this paper we argued that for creating high-quality non-photorealistic illustrations and animations the application of illumination or depth cueing to rendering techniques or the usage of different styles with different objects for emphasis is insufficient. Instead, an object-independent approach to visualization is necessary. We introduced *illustrative effects* which are based on spatial location in general rather than just illumination or depth.

In contrast to previous techniques, our approach separates the visualization from the model. Thus, users can alter the appearance of an illustration by applying spatial illustrative effects without changing the model. This not only allows users to create a variety of new effects but also promotes the easy re-use of existing models.

Our work opens up a variety of new questions. Besides adding other effects similar to fog or a level-of-detail-effect, one further direction of research is to adapt the technique of illustrative effects to other non-photorealistic techniques—especially by including surface shading in the process. It also remains to be investigated whether the information in the effect hierarchy in connection with the local keyframes could be used to generate image series that can convey the main intention of the animation.

It is interesting to note that many rendering techniques in non-photorealistic rendering stem originally from techniques designed for photorealism. Our methods make an explicit attempt to separate non-photorealism from photorealism by designing specific methods and tools which are not simply borrowed from photorealism and applied to non-photorealism. Indeed, it is now promising to investigate to which extent these new techniques can be applied to photorealism.

References

[1] John Lansdown and Simon Schofield, “Expressive Rendering: A Review of Nonphotorealistic Techniques”, *IEEE Computer Graphics and Applications*, vol. 15, no. 3, pp. 29–37, May 1995.

[2] Pierre Poulin and Alain Fournier, “Lights from Highlights and Shadows”, in *Proceedings of the 1992 Symposium on Interactive 3D Graphics*, 1992, pp. 31–38.

[3] Thomas Strothotte, *Computational Visualisation. Graphics, Abstraction and Interactivity*, Springer Verlag, Berlin · Heidelberg · New York, 1998.

[4] Thomas Strothotte, Bernhard Preim, Andreas Raab, Jutta Schumann, and David R. Forsey, “How to Render Frames and Influence People”, *Computer Graphics Forum*, vol. 13, no. 3, pp. 455–466, Sept. 1994.

[5] Elaine R. S. Hodges, Ed., *The Guild Handbook of Scientific Illustration*, Van Nostrad Reinhold, New York, 1989.

[6] Gershon Elber, “Line illustrations $\in$ computer graphics”, *The Visual Computer*, vol. 11, no. 6, pp. 290–296, 1995.

[7] Maic Masuch and Thomas Strothotte, “Visualizing Ancient Architecture using Animated Line Drawings”, in *Proceedings of the International Conference on Information Visualization ’98*, Los Alamitos, CA, 1998, IEEE Computer Society, pp. 261–266.

[8] Jean-Daniel Fekete, Érick Bizouarn, Éric Cournarie, Thierry Galas, and Frédéric Taillefer, “TicTacToon: A Paperless System for Professional 2D Animation”, in *Proceedings of SIGGRAPH 95*, New York, 1995, ACM SIGGRAPH, pp. 79–89.

[9] Siu Chi Hsu and Irene H. H. Lee, “Drawing and Animation Using Skeletal Strokes”, in *Proceedings of SIGGRAPH 94*, New York, 1994, ACM SIGGRAPH, pp. 109–118.

[10] Barbara J. Meier, “Painterly Rendering for Animation”, in *Proceedings of SIGGRAPH 96*, Reading, MA, Aug. 1996, ACM SIGGRAPH, pp. 477–484.

[11] Maic Masuch, Lars Schumann, and Stefan Schlechtweg, “Animating Frame-to-Frame-Coherent Line Drawings for Illustrative Purposes”, in *Simulation und Visualisierung ’98*, Delft, 1998, pp. 101–112, SCS.