Eurographics Symposium on Rendering (2006)
Tomas Akenine-Möller and Wolfgang Heidrich (Editors)

Directing Gaze in 3D Models with Stylized Focus

F. Cole[1], D. DeCarlo[2], A. Finkelstein[1], K. Kin[1], K. Morley[1], and A. Santella[2]

[1]Princeton University [2]Rutgers University

Abstract

We present an interactive system for placing emphasis in stylized renderings of 3D models. The artist chooses a camera position, an area of interest, and a rendering style for the scene. The system then automatically renders the scene with emphasis in the area of interest, an effect we call "stylized focus." Stylized focus draws the viewer's gaze to the emphasized area, through local variations in shading effects such as color saturation and contrast as well as line qualities such as texture and density. We introduce a novel algorithm for local control of line density that exhibits a degree of temporal coherence suitable for animation. Animating the area of emphasis produces an effect we call the "stylized focus pull." Finally, an eye-tracking experiment verifies that the emphasis does indeed draw the viewer's gaze to the area of interest.

1. Introduction

Artists have over the centuries developed a set of principles by which they adjust rendering qualities such as line density and contrast in order to emphasize some areas of an illustration and de-emphasize other areas. This ability is one of the cornerstones of abstraction, allowing the artist to focus the viewer's attention on what is important while eliding unnecessary detail. Researchers advocating non-photorealistic rendering (NPR) techniques often cite the ability to direct the viewer's attention as a benefit of NPR [GG01,SS02]. Few systems, however, have offered this ability automatically and with local control. DeCarlo and Santella [DS02, SD04] developed a system to automatically create abstract renderings from photographs, based on where people tend to look in the photos. However, to date no equivalent work has been performed for 3D rendering. We present an interactive system for directing a viewer's gaze in stylized imagery rendered from 3D models (Figure 1).

There are several benefits to basing such a system on 3D models. First, 3D is a natural domain for many applications that can benefit from our method, such as rendering of architectural models. Second, a 3D system can adjust rendering qualities in ways that might be difficult or impossible using purely 2D methods. Indeed, we experiment with such effects (Section 3) and evaluate their effectiveness (Section 4). Finally, it is easier to create certain kinds of dynamic or animated imagery from 3D models than from 2D sources. One novel form of dynamic imagery presented in this paper is the *stylized focus pull*. In live action cinematography, a focus pull is used to subtly or overtly draw the attention of the audience from one part the scene to another, by dynamically adjusting which parts of the scene are in focus. Our system provides a similar effect using the tools of abstraction.

One challenge in creating abstractions from 3D models is the need for local control of level of detail (LOD). Controlling LOD has two main purposes: first, to reduce visual artifacts that occur when many features of the model project to a small area in the image; second, to aid in directing the viewer's gaze by reducing detail in de-emphasized regions of the image. In both cases, it is vital to maintain temporal coherence for dynamic imagery. Coherence is a thorny challenge for LOD because details that suddenly appear or disappear will visually "pop."

This paper offers several contributions to the literature. First, our system is the first to provide temporally coherent control of emphasis for interactive 3D NPR applications. Second, in support of such applications we describe a new algorithm for local control of line LOD. Third, we present the concept of the *stylized focus pull* wherein emphasis is animated between two areas using the tools of abstraction. Finally, we present the results of several user studies demonstrating that the visual effects for 3D renderings described herein do indeed cause the viewer to

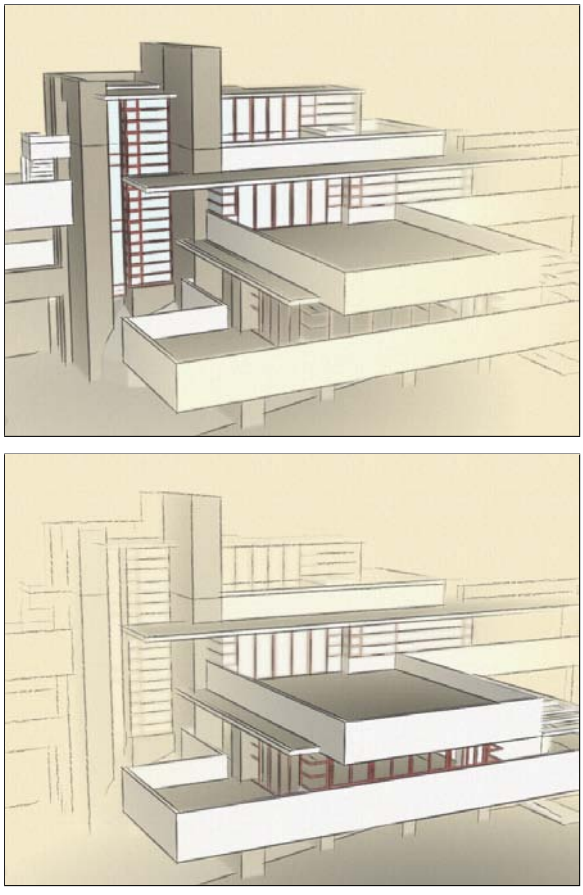

Figure 1: *Placing emphasis. In the upper image the viewer's gaze is drawn to the upper-left part of the structure, whereas in the lower image (with the same camera pose) emphasis is placed in the lower-front balcony. A combination of visual cues give rise to this effect, including line quality and density, color saturation, and contrast.*

look at the emphasized regions. Our examples are largely motivated by architectural rendering, however the potential applications for this research are much broader, including medical and technical illustration, games, interactive story books, and education.

2. Related work

DeCarlo and Santella [DS02] used an eye tracker to identify regions in photographs that viewers tended to focus on, and generated abstract renderings of these photographs with the other, less interesting regions reduced in detail. They then used eye tracking to confirm [SD04] that this abstraction was effective at guiding viewers' gazes. We apply this idea to 3D models, using similar abstraction techniques to focus the viewer on particular regions of the model.

Kosara et al. [KMH01] also explored abstraction and emphasis for 3D models. They extended the conventional depth of field blur effect by blurring semantically irrelevant objects, regardless of their distance to the camera. The viewer was expected to direct attention to the remaining sharp objects in the scene. The authors noted, however, that the blurriness looks unnatural and can be irritating to the eye. Isenberg et al. [IMS00] proposed a more general formulation for assigning emphasis, but used a simple rendering scheme based only on line weight. In contrast to both these papers, our system integrates control of many rendering parameters for both lines and color that work in concert to provide an effective emphasis effect, as shown in our study.

One crucial aspect of our method is control over level of detail (LOD). Most LOD work has focused on reducing detail for efficiency while affecting perception as little as possible [LWC*02]. In contrast, the goal in NPR is to change the impression of the image by reducing unnecessary detail. In NPR, unnecessary detail often takes the form of overly dense or cluttered lines. Most line density control schemes are specialized to the type of lines being drawn. For example, Deussen and Strothotte [DS00] proposed a method to control the drawing of tree leaves and branches. While specialized, their method is effective and provides good temporal coherence.

For rendering architectural models in a pen-and-ink style, the method of Winkenbach and Salesin [WS94] reduces line clutter using "indication": rather than drawing a complicated texture over an entire surface, only a few patches marked by the user are drawn with high detail. Their method also provides local control over line density, primarily for the purpose of controlling tone. However, such hand-crafted line textures are not amenable to rendering with temporal coherence. The system of Strothotte et al. [SPR*94] offers a similar interface – with similar benefits and limitations – to that of Winkenbach and Salesin with the explicit goal of directing the attention of the viewer. Praun et al. [PHWF01] introduced a temporally-coherent hatching method based on blended textures that provides control of line density for reproducing tone. These methods are intended to deal with clutter and shading, not abstraction; they are effective at simplifying repetitive or stochastic textures, but not at abstracting larger structures.

Jeong et al. [JNLM05, NJLM05] developed a method for abstraction based on a series of representations of a 3D model, with varying complexity (created by simplifying the original model). Their algorithm rendered the model with varying LOD, depending on importance or distance. However, such methods do not provide explicit *local* control of line density; rather it emerges as a by-product of mesh simplification. Furthermore, the scheme based on simplification does not perform well on "boxy" meshes such as architectural models. A method introduced by Wilson et al. [WM04] does offer *local* control of line LOD

based on line priority, like the method described herein. The method of Grabli et al. [GDS04] operates similarly to that of Wilson et al., but adds a more sophisticated measure of line density. These two methods only drop strokes in areas of high density. In contrast, the simplification work of Barla et al. [BTS05] simplifies strokes by replacing groups of strokes with fitted, simplified versions. However, none of these methods addressed temporal coherence, inhibiting their use in an interactive setting. One of the contributions of our paper is the introduction of a prioritization algorithm designed for local control of line LOD. This algorithm works with boxy models and enjoys temporal coherence appropriate for rendering dynamic scenes.

There are several strategies to assess the effectiveness of NPR algorithms. One approach is to survey users to gather their subjective impressions. Schumann et al. [SSRL96], for example, demonstrate that architects prefer sketchy renderings to depict the preliminary nature of a design. A second approach measures performance in some task. For example, Gooch et al. [GRG04] compare response times for recognition of faces in photographs and artistic renderings. Heiser et al. [HPA*04] use a concrete task, and evaluate automatically generated assembly instructions [APH*03] by recording construction times for assembly of a piece of furniture. A third approach is available when we are concerned with how the images guide a viewer's attention: examining recordings of eye movements of viewers [SD04]. A viewer's overt attention is measured from eye movements, which are closely coupled with cognitive processes [HH98]. This is the path we take in evaluating our method.

3. Rendering

This section describes how our system renders stylized emphasis. We present several models for calculating "focus," and describe how the level of focus impacts colors and lines in the image. Many of the rendering policies described here are motivated by principles developed by artists and codified in numerous texts, for example [Gup76, Lor85, MA85]. Briefly, we control four qualities to emphasize or de-emphasize part of an illustration: contrast, color saturation, line density, and line sharpness. These strategies are of course interrelated; for example, sharper lines can yield higher contrast.

Our system supports a range of styles composed of lines and shading (suggestive of pen and ink combined with watercolor) but many of the mechanisms we describe should be applicable in a spectrum of media ranging from cel animation to colored pencil.

3.1. Focus Models

In order to specify the degree of emphasis at every part of the image, we introduce the normalized scalar value *abstract focus* (or simply "focus") $f(p)$, which indicates how much emphasis to place at every point p. Following the general framework for expressing focus described by Isenberg et al. [IMS00], our system provides four intuitive focus models that allow an artist to easily express $f(p)$ for a range of useful effects.

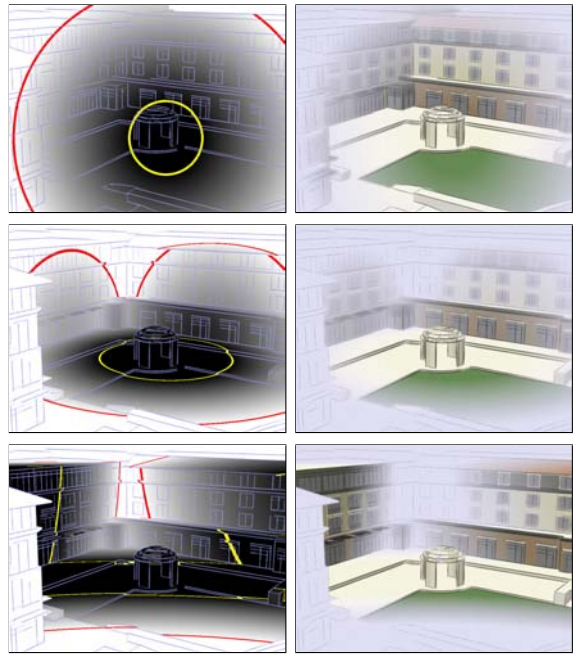

Figure 2: *Comparing focus models. Top to bottom:* 2D focal point*,* 3D focal point*, and* focal plane*. Left: shading level indicates focus; inside yellow is 100%; outside red is 0%. Right: resulting imagery.*

Segmentation. In the simplest model, we assign focus values f_i to a labeled set of discrete components, such as individual organs in medical data, or parts of a mechanical assembly (for example, the different parts of the milling table shown in Figure 6c). Segmentation is appropriate for such applications, but in many other cases we prefer a continuous variation of focus.

Focal plane. Inspired by a real-world camera lens, this model expresses focus $f(p)$ as the distance from p to a "focal plane." The artist specifies this plane by clicking on any point p_f in the model. The focal plane then is taken to be the plane perpendicular to the view direction that contains p_f. This model is useful for expressing a fog-like falloff in emphasis simply by choosing a focal plane close to the camera. Note that its physical analog appears extensively in photography and cinema (because it is embodied by camera lenses), and was used Kosara et al. [KMH01] for directing attention. While this model is familiar to photographers, we have found that the two focal models that follow often feel more natural for abstract rendering of a variety of scenes.

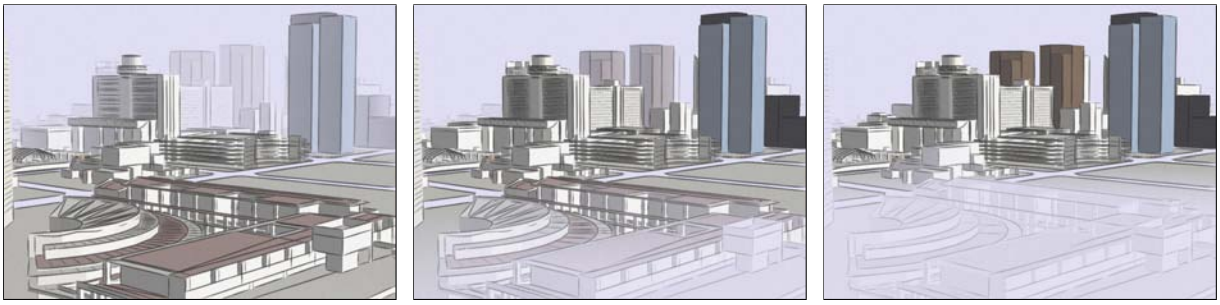

Figure 3: *Stylized focus pull. Focal plane moves gradually from foreground to background, yielding a change in emphasis.*

2D Focal point. In this model, the artist picks a 2D point p_f and the focus $f(p)$ is taken to be the 2D distance $||p - p_f||$. This model produces a foveal effect, which can feel quite natural, especially when used in concert with a "focus pull" as described below. Note that when working from still imagery such a photographs (as were DeCarlo and Santella [DS02]) this may be the only viable option. However, for some compositions this effect appears unnaturally symmetrical, and we find the next model to be more pleasing.

3D Focal point. In this model, the artist chooses a 3D point p_f from which the focus falls off radially in 3D: $f(p) = ||p - p_f||$. This focal model is perhaps most intuitive for 3D scenes, and distinguishes our work from the bulk of previous methods for placing emphasis in illustrations, as they generally did not have access to 3D information.

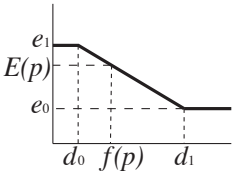

Figure 2 illustrates the effects of the three continuous camera models. In all three cases, the artist chose a focal point (or plane) by clicking at the rear of the base of the cupola. A transfer function (right) provides further control for adjusting focus, based on four artist-specified parameters e_0, e_1, d_0 and d_1, whose effects may be seen in Figure 2. The parameters e_0 and e_1 indicate the degree of emphasis in the most or least emphasized regions (fully black and white, respectively). The parameters d_0 and d_1 express the locations of the yellow or red bands, controlling the falloff region. This transfer function takes the form:

$$T(p) = clamp((f(p) - d_0)/(d_1 - d_0))$$

where d_0 and d_1 are expressed relative to the image diameter for the 2D focal point model (or the model diameter for the other two cases) in order to provide controls ranging between 0 and 1. Finally the emphasis at every point p is taken to be

$$E(p) = T(p)e_0 + (1 - T(p))e_1$$

a linear combination of e_0 and e_1, which express the strongest and weakest emphasis. We have found these controls to be quite intuitive, and given a scene the artist can compose a shot using only: (1) choice of focus model, (2) the click of a mouse to choose the focal point, (3) the adjustment of the four parameters.

Focus pull. A common visual device in live action film is the "focus pull" wherein the camera operator *pulls* the focus lever of the camera in order to shift the focus of the lens, and thereby *pulls* the attention of the viewer from one part of the scene to another. Inspired by this effect, we introduce the notion of a "stylized focus pull" where we animate the focal point (or plane), causing a temporal shift in the emphasis of the resulting rendering (Figure 3). In many cases the results appear quite natural, and in other cases they can be visually distracting.

3.2. Rendering Effects

We implement eight effects that respond locally to emphasis $E(p)$ in the image. Three are color effects (desaturation, fade, and blur) while five adjust line qualities (texture, width, opacity, density and overshoot), some of which are shown in Figure 4. These effects are generally used in combination, though their relative impact are under artistic control. Each effect has its own transfer function analogous to that of $E(p)$, with its own set of four artist-specified parameters, and is applied to the result of the emphasis transfer function. For example, there are four parameters for adjusting the shape of the desaturation transfer function D, and the color at point p is desaturated by the amount $D(E(p))$. This two-stage transfer function has the benefit of offering both global control of the shape of the emphasis in the image as well as fine control over the individual effects. Aside from the obvious benefit of aesthetic flexibility, this fine control also tends to "break up" the image so that the potential derivative discontinities that occur at the yellow and red bands in Figure 2 are not noticeable.

While it might seem cumbersome to adjust 36 parameters (four for each of eight individual effects and overall emphasis) in order to set up a scene, we have found the process to be quite straightforward for two reasons. First, the controls themselves behave quite intuitively, and with very little

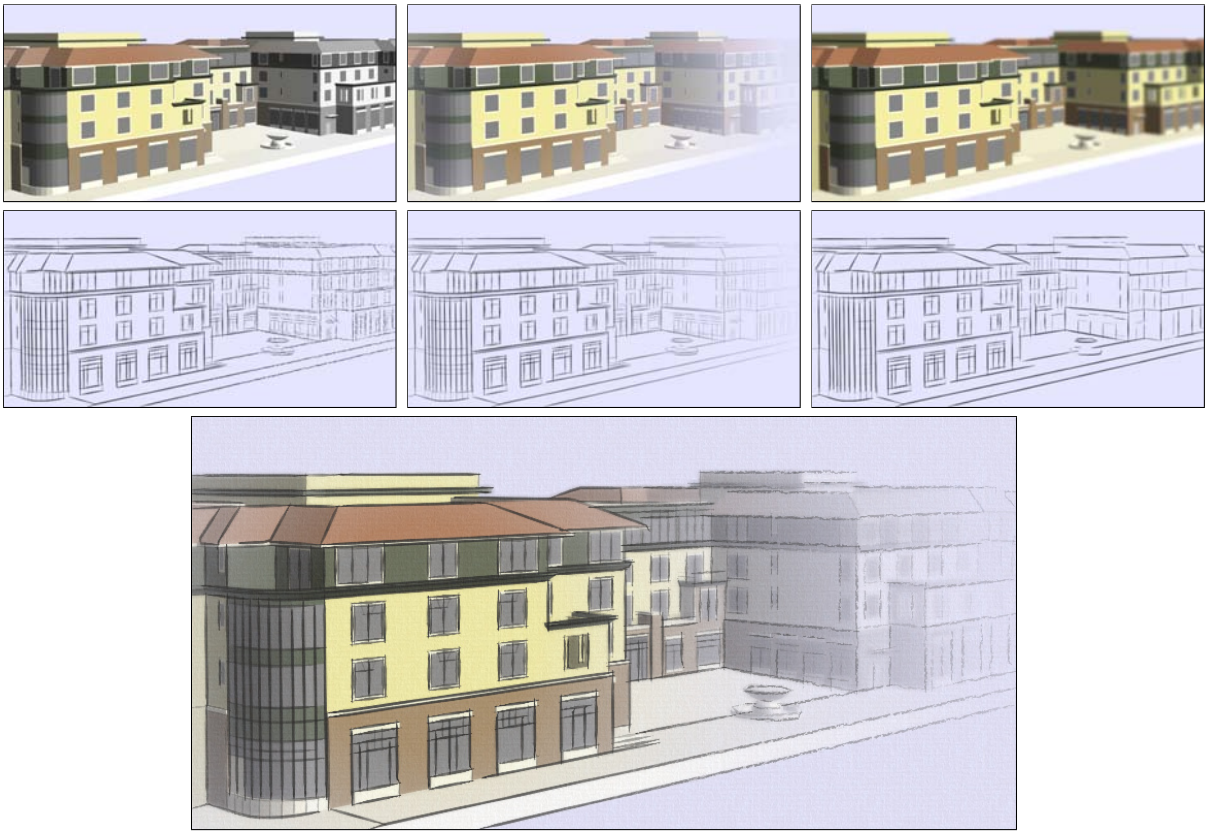

Figure 4: *From top to bottom: color effects (desaturation, fade, blur); line effects (opacity, texture, density); all six combined.*

practice it is easy to converge on a desired effect. Second, our system can save and restore these parameters (together with a few others) collectively in a "style." Having saved a style, for example the "blueprint" style shown in Figure 6, it is then easy to create new imagery using that style with slight or no modification. Other aspects of the style include background color, the choice of textures used for lines and paper, as well as the paper opacity and scale.

Given a model, camera, point of emphasis, and style, the overall rendering process is as follows. First, visible line paths are determined using a process described in the next section. Next, colors are rendered into the frame buffer using a pixel shader. Finally, the lines are drawn over the resulting color image as textured triangle strips, using a method similar to those of Northrup and Markosian [NM00] and Kalnins et al. [KDMF03]. In order to provide emphasis cues, the width, opacity and textures are modulated along the lengths of these triangle strips. The line textures are interpolated among two or more 2D textures τ_i, based on an emphasis calculation, using a simple trick. The series τ_i is loaded into a 3D texture wherein the emphasis indexes the third dimension of the texture. The texture is then rendered using trilinear interpolation.

The pixel shader for color rendering first renders the color of the model at every pixel. During the same rendering pass, the shader also computes pixel-accurate values for emphasis using one of the four focus models described in Section 3.1. Note that in an early implementation we computed emphasis and color effects (on the CPU) at vertices of the model, and found that large triangles could give rise to color discontinuities and also miss significant color transitions (the equivalent of highlight artifacts in Gouraud shading). The shader first desaturates the color, and then fades the color by interpolating towards the background color, in both cases by an amount appropriate to the style and emphasis. Finally, the shader blurs the image based on local emphasis. The ideal way to do this would be to use a variable-size filter kernel. However, in order to achieve interactive performance, our system blends between a series of four texture images of varying blur, ranging from no blur to an approximation of a gaussian blur with a kernel radius of 8 pixels. Each final pixel is a linear combination of the two nearest blur textures. Note this process requires rendering the blur textures at every frame, which is feasible using separable filters. When applied in combination, these color effects provide a natural cue for emphasis.

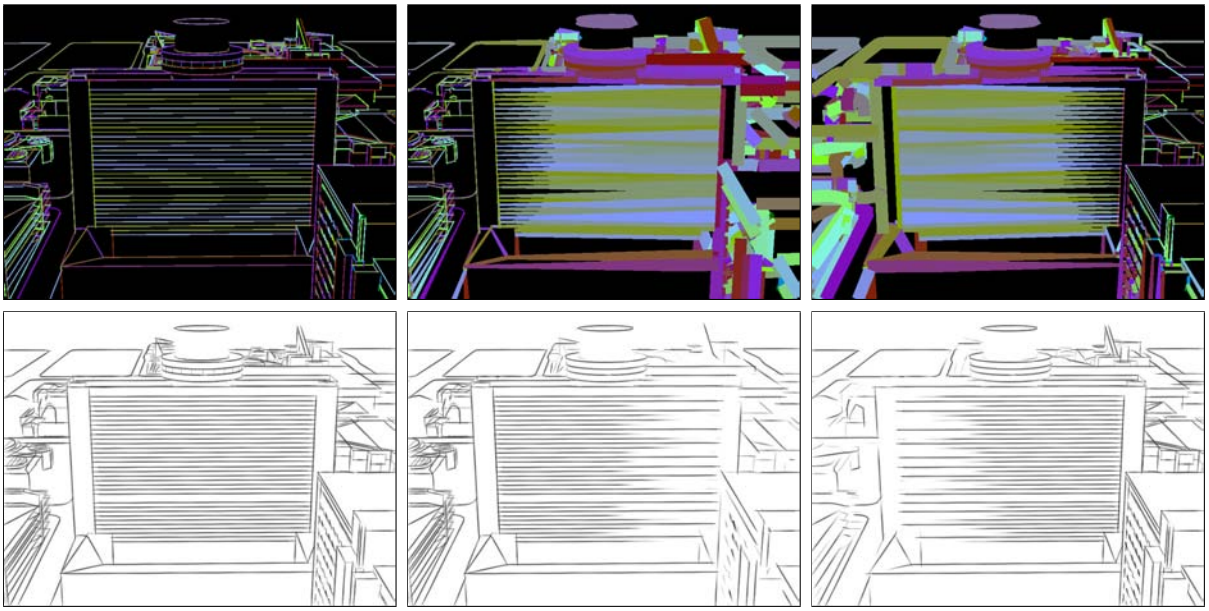

Figure 5: *Item and priority buffer. The item buffer (top left) determines line visibility (bottom left). The priority buffer (top middle and top right) determines line density (bottom middle and bottom right). Lines are drawn in unique colors in order from low- to high-priority, so the highest priority line in any region will prevail. To emphasize the left side of the image (middle), narrower lines are drawn on the left side of the priority buffer, leading to higher line density. The same approach is used to emphasize the right side of the image (right).*

3.3. Controlling Line Density

We have observed that one's initial impression for many of our illustrations is that the strongest emphasis cues come from color effects; perhaps the line effects are superfluous. However, our experiments described in Section 4 indicate that lines also play a substantial role. Furthermore, in styles such as the "blueprint" and "chalk" styles shown in Figure 6, color effects are minimal and thus lines play the central role in suggesting emphasis. While line opacity and width provide important controls, we have found that they are generally best used sparingly because the opaque lines combat the aforementioned illusion of fog. Therefore we have found that control over line *density* plays a crucial role in placing emphasis.

Based on desired emphasis, our algorithm sets a maximum line density in every area of the scene. One limitation of this approach is that we have no mechanism for adding lines to increase density in a given region where no lines existed in the model. Nevertheless, the approach seems to work well for models with moderate to high complexity, and nicely complements the other color and line effects.

Our density control method is inspired by the *item buffer* data structure. This data structure was introduced by Weghorst et al. to accelerate ray tracing [WHG84]. Northrup and Markosian [NM00] and later Kalnins et al. [KMM*02, KDMF03] adapted the item buffer for computing visibility in line drawings, calling it an "ID reference image." We also use this approach for line visibility because of its simplicity and efficiency. The item buffer is an image that contains at each pixel p the identity (ID) of the object from the scene that covers p. To use the item buffer for line visibility, we render the lines and polygons of the model together in a separate visibility pass. Each line is identified by a unique color ID (Figure 5), while each polygon is drawn in black. We then step along each line at a small, fixed screen space interval. At each step, we test to see if the line ID is present in the item buffer at its projected location p. If so, that section of the line is visible in the final scene. If not, that section is obscured by a polygon or other line segment.

The item buffer already gives minimal control of line density, because no more than one line can ever cover p. However, there is no local control; the item buffer essentially sets a global maximum line density. We would like the ability to control this maximum line density at a local level. For example, we wish to stipulate that for a particular region in the illustration, no two lines will be closer than 10 pixels. Furthermore, since we must remove lines to reduce line density, we want to select the least important lines (in some sense) for removal. To achieve this goal, we introduce a new algorithm for locally controlling line density. The method relies on a data structure inspired by the item buffer, which we call the *priority buffer*.

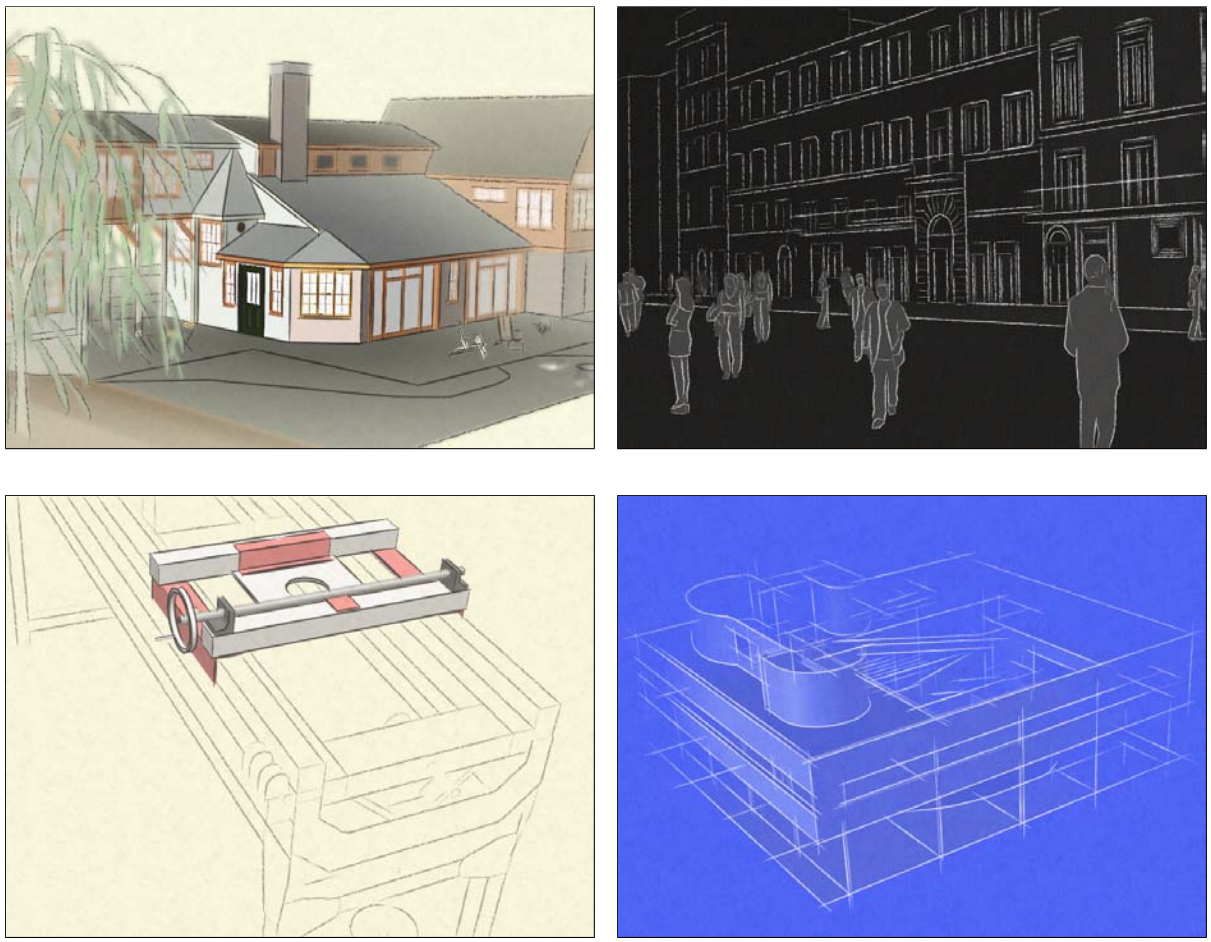

Figure 6: *Results, left to right and top to bottom: (a) a tree gives context but does not distract from the house; (b) a style with chalk lines over dark paper; (c) a segmented milling table emphasizes only the upper tray; (d) line overshoot gives this blueprint style a schematic quality.*

The priority buffer shares its general appearance with the item buffer (Figure 5): it is an offscreen buffer, consisting of lines colored by ID. The priority buffer departs from the item buffer in two major respects: first, the lines are sorted in the priority buffer by an arbitrary "priority" value, not by depth. Higher priority lines will be drawn on top of lower priority lines, even if the lower priority lines are closer in 3D to the camera. Second, lines in the priority buffer may vary dramatically in width. The width of the priority buffer line is inversely proportional to the desired line density in the region. Thus, in areas of low density, lines will be drawn wider; in high density areas, lines will be drawn narrower. Wide, high priority lines will carve a broad swath through the image, overwriting any other lines in their neighborhood. The exact width of the priority buffer lines is controlled by the user through a transfer function (Section 3.1).

We implement the priority buffer algorithm as follows. After we use the item buffer to compute the visible portions of each line, we sort the visible lines by *priority*. The priority quantity can be any measure of how important the individual lines are. In our experiments we have used a simple heuristic to assign priority: the length of the original line in the 3D model, before visibility testing has been performed. We assume, in other words, that long lines correspond to important features. This heuristic seems to work well for the architectural models in our experiments, but one can imagine more sophisticated methods that might consider, for example, exterior silhouettes to be of high importance. In our tests the $O(n \log n)$ time required to sort the lines is negligible; however it would be easy to remove this overhead by assigning depths based on priority and using the z-buffer to perform the sort while rendering to the priority buffer.

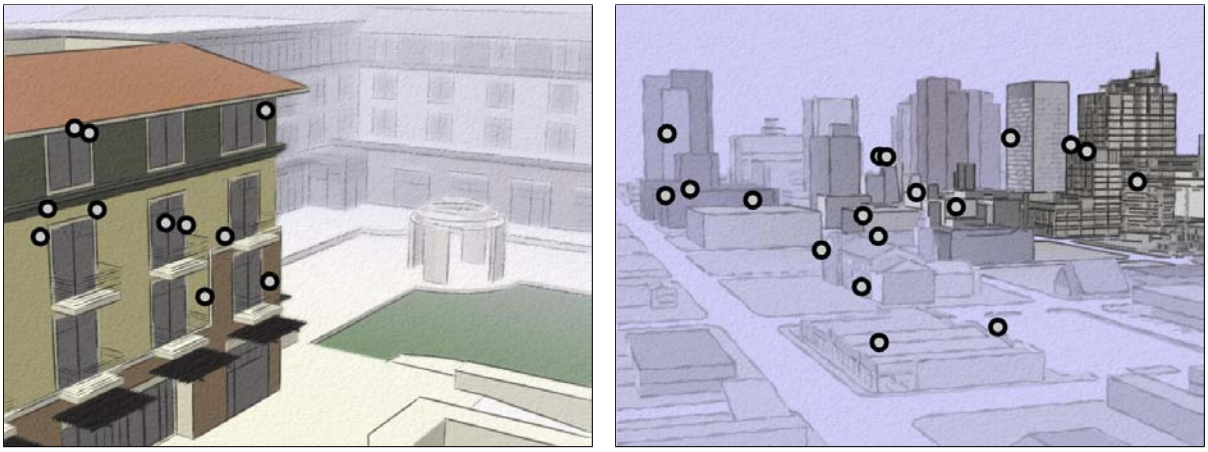

Figure 7: *Evaluation. Tracked eye-movements in the hotel scene (left) follow the emphasis. Eye-movements in the city-scape (right) only loosely respond to emphasis.*

Once we have rendered the priority buffer, we can use it to decide which visible lines to render and which lines to omit. Similarly to the item buffer test, we walk along the length of each line, checking at each step whether the line is visible in the priority buffer. The item buffer test makes a binary decision about line visibility: either the line is visible, or it is not. If we use a similar test with the priority buffer, we produce effective imagery, but with one caveat – weak temporal coherence. Lines tend to "pop" in and out, particularly when groups of parallel lines alias with rows and columns of pixels in the buffer. Our solution to this problem is straightforward. When walking a line in the priority buffer, we test a $w \times w$ region (we use $w = 5$) centered around the projected sample point. We count the number of pixels in this region with the appropriate ID, and divide by the number we would expect to see if the line were fully visible. For example, if the line were three pixels wide, we would expect 3×5 or 15 appropriately colored pixels to appear in our window. The ratio of visible to expected pixels gives us a *degree* of visibility v between 0 and 1. We then multiply the opacity of the line by v when rendering, creating a continuous falloff of visibility. This strategy provides reasonable temporal coherence, while not forfeiting the interactive frame rates available with the priority buffer.

4. Results

We have found the techniques described herein to be effective for drawing the eye to targeted areas within an illustration, based on both our own informal experience using this system and also the results of a formal study described in Section 4.1. The system provides a natural and intuitive interface for the artist to compose a shot with emphasis. It offers controls with a fair degree of stylistic flexibility, but without so many parameters as to be cumbersome. Figure 6 shows some example illustrations created in our system, further demonstrating the range of styles available as well as the variety of types of scenes with which we have experimented.

The system runs at interactive frame rates for models of some complexity, where the performance is generally bound by the visual complexity of the rendering. For example, the model of downtown Phoenix shown in Figure 7 (right) contains 240K faces and 125K line paths, and typically achieves 3-5 FPS on a 3.0 Ghz Pentium 4 with a nVidia GeForce 6800 graphics card. For typical views in such a model, the application is CPU-bound, spending the bulk of its time sampling line paths for visibility in the item and priority buffers. However, where the view is such that only a few lines are visible (for example a close-up of the side of a building) the frame rate is dominated by the time to send lines and faces over the bus to the graphics card. For much smaller scenes, or when only parameters such as the degree of focus are adjusted, the application refreshes at video rates and is bound by the time to read back the item and priority buffers.

4.1. Evaluation

The goal of this project is to construct imagery that implicitly guides the attention of the viewer to specific places in a scene. A natural question to ask is: how effective are these renderings at achieving this objective? We measure the overt visual attention given to emphasized regions using eye tracking, comparing viewings for emphasized and uniform images. Results of this evaluation indicate that viewers examine emphasized regions more than they examine the same location in a uniformly rendered image of the same scene (p-value < 0.001).

Our experimental design follows that of Santella and DeCarlo [SD04]. Thirteen (student) viewers examined a series of 18 rendered scenes in a variety of styles, with and without emphasis. The viewers were asked to rate how much they liked each image. This task served to motivate viewers to look at the images, but resulting scores were too sparse and noisy to analyze quantitatively. Each image was displayed for eight seconds on a 19-inch LCD display. The screen was viewed at a distance of approximately 86 cm, subtending a visual angle of approximately 25 degrees horizontally x 19 degrees vertically. Eye movements were monitored using an ISCAN ETL-500 tabletop eye-tracker (with a RK-464 pan/tilt camera). The pan/tilt unit was not active during the experiment. Instead, subjects placed their heads in an optometric chin rest to minimize head movements. Viewers saw renderings with:

1. color and lines with uniform level of focus in the scene
2. uniform color only, no lines
3. uniform lines only, no color
4. emphasis of color and lines, based on a single focal point
5. emphasis in color only, with uniform lines
6. emphasis in lines only, with uniform color
7. emphasis in color only, no lines
8. emphasis in lines only, no color

We chose two different focal points in each scene, leading to two versions of each type of emphasized image (types 4-8). This resulted in a total of 13 versions of each of the 18 scenes. Use of two focal points for each scene provides evidence that the effect of emphasis is not limited to the single most natural focal point in the scene. Each viewer saw only one (randomly chosen) version of each scene.

Recall that we have an emphasis value $E(p)$ at every point p in a rendering. When a viewer fixates at p, the value $E(p)$ tells us how emphasized that region is. We can use this value, sampled under fixations, to measure how much the viewer looked at the emphasized regions. Suppose the viewer's fixations in an emphasized image followed a path $p(t)$ from time t_0 to time t_1. Then we can measure the average emphasis E_f under those fixations by:

$$E_f = \frac{1}{(t_1 - t_0)} \int_{t_0}^{t_1} E(p(t))dt$$

Of course we cannot simply claim success if E_f is greater than the average $E(p)$ in the scene. The viewer might have looked in places of high $E(p)$ independently of the emphasis, perhaps because they were interesting parts of the image. However, for a given focal point we can control for such cases by evaluating E_f using the same $E(p)$ measured over the fixation path from the corresponding uniformly-emphasized image (types 1-3). This provides a (control) measure of how much those emphasized areas are examined *even without emphasis*. If emphasis increases attention on an area, E_f will be greater for the fixation path in the emphasized (test) image than for the fixation path of the uniformly emphasized (control) image. On the other hand, if the viewer looks in un-emphasized areas in a test image, then E_f will not be higher than that of the control. Likewise, if the viewer examines emphasized regions in the test, but also looks in the same regions in the control, then E_f will be similar for both images.

For each type of emphasized image (test images, types 4-8), values of E_f were compared with those in a corresponding unemphasized image (controls, types 1-3), collapsing over all scenes and both emphasis points for each scene. A two way condition cross scene ANOVA for all conditions was conducted followed by multiple comparisons between each test and control condition. Emphasized locations were more heavily examined in *all* styles of emphasis (types 4-8) (p-value < 0.001). As we hypothesized, emphasis of color and lines (stimuli type 5 and 6) each had a significant effect individually. These effects were individually weaker (p-value < 0.001) than their combined effect (type 4). Finally, matching our subjective impression, the effect of emphasizing color with uniform lines (type 5) was stronger (p-value < 0.01) than that of emphasizing lines with uniform color (type 6).

We conclude from this experiment that our method is effective. In all rendering types it shifts viewer attention to the emphasized point. This is a general effect, found in multiple styles and for multiple points of emphasis in the scene. Emphasis works in styles that contain just color or lines alone. It is also effective when images contain both color and lines, but only one is emphasized, though it is most effective when emphasis of both are combined.

5. Conclusion

We demonstrate an interactive system for directing the attention of the viewer to areas of interest in architectural renderings from 3D models. The system combines many interactive rendering effects to place emphasis in illustrations. We introduce a new method for line density control suitable for animation. Using this system we present a new idiom called an *stylized focus pull* that is the artistic analog of the cinematic focus pull used with live action cameras. Finally, we show experimental evidence that the effects described herein actually work.

5.1. Limitations and Future Work

Our method has three primary limitations, each of which prompts areas for future research:

No feedback. While the system attempts to emphasize some areas of the illustration and de-emphasize others, it does not evaluate whether or not such effects were achieved. For example, when a striking feature exists outside the emphasis in the scene, it may require stronger de-emphasis than would automatically take place. Our system has no mechanism to notice such situations, which may explain

the weak result in Figure 7 (right). Computational models of visual salience [IKN98] can estimate the prominence of features in an emphasized rendering, and could allow the system to iteratively adjust emphasis until the desired result is achieved.

The model. Our method depends on the model to offer features that can be emphasized, so it is difficult to place emphasis in areas of very low detail. In particular, the method relies on lines and colors to create stylized focus, but has no mechanism to "invent" lines and colors where none exists. Two possible strategies for addressing this concern in the future are:

- *Texture abstractions.* Artists often apply textures to surfaces in order to provide fine detail. Abstracting such textures for use in NPR, perhaps in the spirit of Praun et al. [PHWF01] might allow a broader range of emphasis in simple models.
- *Shadow hatching.* Shading effects such as shadows might offer a tone source for adding hatching similar to that of Winkenbach and Salesin [WS94] could provide further detail.

Fogginess. As shown in Figure 8, some combinations of effects can suggest fog in de-emphasized areas. This impression can be either enhanced or combatted through manipulation of the line rendering qualities, but the cue seems to come most strongly where the colors fade into the background. We believe that this illusion can be ameliorated by instead simplifying the colors, as mocked up in the right figure, in the spirit of Barla et al. [BTM06].

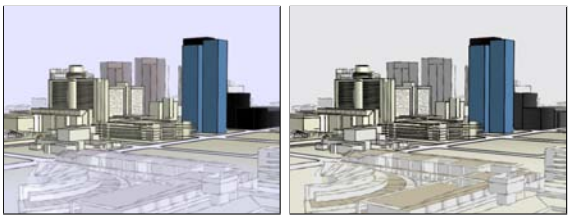

Figure 8: *Fog. Left: some combinations of effects can suggest fog in de-emphasized areas. Right: this illusion can be ameliorated using a quantized color simplification technique similar to that of Barla et al. [BTM06] (mocked up here).*

Acknowledgments

We would like to thank John Bacus, Tyler Miller, and the rest of the team at @Last Software for their advice and encouragement on this project, as well as the use of most of the models shown in this paper. We also thank Matthew Potts for the use of the house model with trees. This work is supported in part by NSF grant IIS-0511965.

References

[APH*03] AGRAWALA M., PHAN D., HEISER J., HAYMAKER J., KLINGNER J., HANRAHAN P., TVERSKY B.: Designing effective step-by-step assembly instructions. *ACM Transactions on Graphics 22*, 3 (July 2003), 828–837.

[BTM06] BARLA P., THOLLOT J., MARKOSIAN L.: X-toon: An extended toon shader. In *International Symposium on Non-Photorealistic Animation and Rendering (NPAR)* (2006).

[BTS05] BARLA P., THOLLOT J., SILLION F.: Geometric clustering for line drawing simplification. In *Proceedings of the Eurographics Symposium on Rendering* (2005).

[DS00] DEUSSEN O., STROTHOTTE T.: Computer-generated pen-and-ink illustration of trees. In *Proceedings of ACM SIGGRAPH 2000* (2000), Computer Graphics Proceedings, Annual Conference Series, pp. 13–18.

[DS02] DECARLO D., SANTELLA A.: Stylization and abstraction of photographs. *ACM Transactions on Graphics 21*, 3 (July 2002), 769–776.

[GDS04] GRABLI S., DURAND F., SILLION F.: Density measure for line-drawing simplification. In *Proceedings of Pacific Graphics* (2004).

[GG01] GOOCH B., GOOCH A.: *Non-Photorealistic Rendering*. A K Peters, 2001.

[GRG04] GOOCH B., REINHARD E., GOOCH A.: Human facial illustrations: Creation and psychophysical evaluation. *ACM Trans. Graph. 23*, 1 (2004), 27–44.

[Gup76] GUPTILL A. L.: *Rendering in Pen and Ink*. Watson-Guptill Publications, New York, 1976.

[HH98] HENDERSON J. M., HOLLINGWORTH A.: Eye movements during scene viewing: An overview. In *Eye Guidance in Reading and Scene Perception*, Underwood G., (Ed.). Elsevier, 1998, pp. 269–293.

[HPA*04] HEISER J., PHAN D., AGRAWALA M., TVERSKY B., HANRAHAN P.: Identification and validation of cognitive design principles for automated generation of assembly instructions. In *AVI '04: Proceedings of the working conference on Advanced visual interfaces* (New York, NY, USA, 2004), ACM Press, pp. 311–319.

[IKN98] ITTI L., KOCH C., NIEBUR E.: A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20*, 11 (1998), 1254–1259.

[IMS00] ISENBERG T., MASUCH M., STROTHOTTE T.: 3D Illustrative Effects for Animating Line Drawings. In *Proceedings of the IEEE International Conference on Information Visualisation, July 19–21, 2000, London, England* (2000), pp. 413–418.

[JNLM05] JEONG K., NI A., LEE S., MARKOSIAN L.: Detail control in line drawings of 3D meshes. *The Visual*

Computer 21, 8-10 (September 2005), 698–706. Special Issue of Pacific Graphics 2005.

[KDMF03] KALNINS R. D., DAVIDSON P. L., MARKOSIAN L., FINKELSTEIN A.: Coherent stylized silhouettes. *ACM Transactions on Graphics 22*, 3 (July 2003), 856–861.

[KMH01] KOSARA R., MIKSCH S., HAUSER H.: Semantic depth of field. In *IEEE Symposium on Information Visualization 2001 (InfoVis 2001)* (22–23 2001).

[KMM*02] KALNINS R. D., MARKOSIAN L., MEIER B. J., KOWALSKI M. A., LEE J. C., DAVIDSON P. L., WEBB M., HUGHES J. F., FINKELSTEIN A.: WYSIWYG NPR: drawing strokes directly on 3d models. In *Proceedings of SIGGRAPH 2002* (2002), pp. 755–762.

[Lor85] LORENZ A.: *Illustrating Architecture*. Van Nostrand Reinhold, 1985.

[LWC*02] LUEBKE D., WATSON B., COHEN J. D., REDDY M., VARSHNEY A.: *Level of Detail for 3D Graphics*. Elsevier Science Inc., 2002.

[MA85] MEYER S. E., AVILLEZ M.: *How to Draw in Pen and Ink*. Roundtable Press, 1985.

[NJLM05] NI A., JEONG K., LEE S., MARKOSIAN L.: *Multi-scale Line Drawings from 3D Meshes*. Tech. Rep. CSE-TR-510-05, Department of EECS, University of Michigan, July 2005.

[NM00] NORTHRUP J. D., MARKOSIAN L.: Artistic silhouettes: a hybrid approach. In *NPAR '00: Proceedings of the 1st international symposium on Non-photorealistic animation and rendering* (2000), pp. 31–37.

[PHWF01] PRAUN E., HOPPE H., WEBB M., FINKELSTEIN A.: Real-time hatching. In *Proceedings of SIGGRAPH 2001* (2001), p. 581.

[SD04] SANTELLA A., DECARLO D.: Visual interest and NPR: an evaluation and manifesto. In *NPAR 2004* (June 2004), pp. 71–78.

[SPR*94] STROTHOTTE T., PREIM B., RAAB A., SCHUMANN J., FORSEY D. R.: How to render frames and influence people. *Computer Graphics Forum, Proceedings of EuroGraphics 1994 13*, 3 (1994), 455–466.

[SS02] STROTHOTTE T., SCHLECHTWEG S.: *Non-Photorealistic Computer Graphics: Modeling, Rendering, and Animation*. Morgan Kaufmann, 2002.

[SSRL96] SCHUMANN J., STROTHOTTE T., RAAB A., LASER S.: Assessing the effect of non-photorealistic rendered images in CAD. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Common Ground* (1996), pp. 35–41.

[WHG84] WEGHORST H., HOOPER G., GREENBERG D. P.: Improved computational methods for ray tracing. *ACM Transactions on Graphics 3*, 1 (Jan. 1984), 52–69.

[WM04] WILSON B., MA K.-L.: Rendering complexity in computer-generated pen-and-ink illustrations. In *NPAR '04: Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering* (New York, NY, USA, 2004), ACM Press, pp. 129–137.

[WS94] WINKENBACH G., SALESIN D. H.: Computer-generated pen-and-ink illustration. In *Proceedings of SIGGRAPH 1994* (1994), pp. 91–100.