Eurographics Symposium on Rendering 2003
Per Christensen and Daniel Cohen-Or (Editors)

Detail to Attention: Exploiting Visual Tasks for Selective Rendering

K. Cater,[1†] A. Chalmers[1] and G. Ward[2]

[1] Department of Computer Science, The University of Bristol, UK
[2] Anyhere Software, U.S.A

Abstract

*The perceived quality of computer graphics imagery depends on the accuracy of the rendered frames, as well as the capabilities of the human visual system. Fully detailed, high fidelity frames still take many minutes even hours to render on today's computers. The human eye is physically incapable of capturing a moving scene in full detail. We sense image detail only in a 2° foveal region, relying on rapid eye movements, or saccades, to jump between points of interest. Our brain then reassembles these glimpses into a coherent, but inevitably imperfect, visual percept of the environment. In the process, we literally lose sight of the unimportant details. In this paper, we demonstrate how properties of the human visual system, in particular **inattentional blindness**, can be exploited to accelerate the rendering of animated sequences by applying **a priori** knowledge of a viewer's task focus. We show in a controlled experimental setting how human subjects will consistently fail to notice degradations in the quality of image details unrelated to their assigned task, even when these details fall under the viewers' gaze. We then build on these observations to create a perceptual rendering framework that combines predetermined **task maps** with spatiotemporal contrast sensitivity to guide a progressive animation system which takes full advantage of image-based rendering techniques. We demonstrate this framework with a **Radiance** ray-tracing implementation that completes its work in a fraction of the normally required time, with few noticeable artifacts for viewers performing the task.*

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation - Viewing Algorithms I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism - Animation I.4.8 [Image Processing and Computer Vision]: Scene Analysis -Time-varying imagery

1. Introduction

One of the central goals in computer graphics is to produce the best *perceived* image in the least amount of time. Advanced rendering techniques such as ray-tracing and global illumination improve image quality, but at a commensurate cost. In many cases, we end up spending significant effort improving details the viewer will never notice. If we can find a way to apply our effort selectively to the small number of regions a viewer attends in a given scene, we can improve the perceived quality without paying the full computational price.

Most computer graphics serve some specific visual task

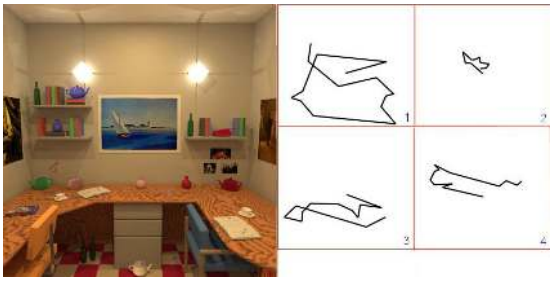


Figure 1: *Effects of a task on eye movements. Eye scans for observers examined with different task instructions; 1. Free viewing, 2. Remember the central painting, 3. Remember as many objects on the table as you can, 4. Count the number of books on the shelves.*

† cater@cs.bris.ac.uk

– telling a story, advertising a product, playing a game, or simulating an activity such as flying. In the majority of cases, objects relevant to the task can be identified in advance, and the human visual system focuses its attention on these objects at the expense of other details in the scene. Figure 1 shows a rendered image for which we instructed participants to perform a number of arbitrary tasks. The eye-tracking scans demonstrate that subjects focus on task-related objects and fail to attend other details in the scene. In this paper, we show experimentally that it is possible to render scene objects not related to the task at lower resolution without the viewer noticing any reduction in quality.

We take advantage of these findings in a computational framework that applies high-level task information to deduce error visibility in each frame of a progressively rendered animation. By this method, we are able to generate high quality animated sequences at constant frame rates in a fraction of the time normally required. A key advantage to this technique is that it only depends on the task, not on the viewer. Unlike the foveal detail rendering used in flight simulators, there is no need for eye-tracking or similar single-viewer hardware to enable this technology, since attentive viewers participating in the same task will employ similar visual processes.

We begin with a review of previous work in perceptually-based rendering, focusing on areas most closely related to our technique. We then present an experimental validation of selective rendering using a *task map* to control image detail. These results are followed by a description and demonstration of our perceptual rendering framework, which extends these ideas to incorporate a model of spatiotemporal contrast sensitivity, enabling us to predict local error visibility. In our implementation of this framework, we use ray-tracing and image-based rendering to compute an animated sequence in two minutes per frame.

2. Previous Work

Visual attention is a coordinated action involving conscious and unconscious processes in the brain, which allow us to find and focus on relevant information quickly and efficiently. If detailed information is needed from many different areas of the visual environment, the eye does not scan the scene in a raster-like fashion, but jumps so that the relevant objects fall sequentially on the fovea. These jumps are called *saccades* [30].

There are two general visual attention processes, labelled *bottom-up* and *top-down*, which determine where humans locate their visual attention [8]. The bottom-up process is purely stimulus driven, for example, a fire in the dark, a red apple in a green tree, or the lips and eyes of another person the most mobile and expressive elements of a face. In all these cases, the visual stimulus captures attention automatically without volitional control. This is evolutionary; the movement may be danger lurking behind a bush, or we may need to find ripe fruit for our meal. In contrast, the top-down process is under voluntary control, and focuses attention on one or more objects that are relevant to the observer's goal when studying a scene. Such goals might include looking for a lost child, searching for an exit, or counting the number of books on a shelf, as shown in Figure 1.

General knowledge of the human visual system has been used to improve the quality of the rendered image [4, 6, 15, 16, 21, 22]. Other research has investigated how complex model detail can be reduced without any reduction in the viewer's perception of the models [11, 19, 23, 28]. Along these lines, Maciel and Shirley's visual navigation system used texture mapped primitives to represent clusters of objects to maintain high and approximately constant frame rates [12]. The application of visual attention models in computer graphics has so far exploited only peripheral vision and the bottom-up visual attention process, as we discuss below.

2.1. Peripheral Vision

Due to the fact that the human eye only processes detailed information from a relatively small part of the visual field, it is possible to reduce detail in the periphery without upsetting visual processing. In numerous studies, Loschky and McConkie [10] used an eye-linked, multiple resolution display that produces high visual resolution only in the region to which the eyes are directed. They were able to show that photographic images filtered with a window radius of 4.1° produced results statistically indistinguishable from that of a full, high-resolution display. The display they propose does, however, encounter the problem of updating the multi-resolution image after an eye movement without disturbing the visual processing. Their work has shown that the image needs to be updated after an eye saccade within 5 milliseconds of a fixation, otherwise the observer will detect the change in resolution. These high update rates were only achievable using an extremely high temporal resolution eye tracker, and pre-storing all possible multi-resolution images that were to be used.

In another experiment, Watson et al. [27] evaluated the effectiveness of high detail insets in head-mounted displays. The high detail inset they used was rectangular and was always presented at the finest level of resolution. Three inset conditions were investigated: a large inset - half the complete display's height and width, a small inset size - 30 % of the complete display's height and width, and no inset at all. The level of peripheral resolution was varied at: fine resolution 320x240, medium resolution 192x144 and coarse resolution 64x48. Their results showed that although observers found their search targets faster and more accurately in a full high resolution environment, this condition was not significantly better than the high-resolution inset displays with either medium or low peripheral resolutions.

2.2. Saliency Models

Low-level saliency models determine what visual features will involuntarily attract our attention in a scene. Visual psychology researchers such as Yarbus [30], Itti and Koch [7] and Yantis [29] showed that the visual system is highly sensitive to features such as edges, abrupt changes in color, and sudden movements. This low-level visual processing has been exploited in computer graphics by Yee et al. [31] to accelerate animation renderings with global illumination, by applying a model of visual attention to identify conspicuous regions. Yee constructs a spatiotemporal error tolerance map, called the *Aleph map*, from spatiotemporal contrast sensitivity and a low-level *saliency map*, for each frame in an animation. The saliency map is obtained by combining the conspicuity maps of intensity, color, orientation and motion. The Aleph map is then used as a guide to indicate where more rendering effort should be spent in computing the lighting solution, significantly improving the computational efficiency during animation. Subsequent work by Marmitt and Duchowski [14] showed, however, that such bottom-up visual attention models do not always predict attention regions in a reliable manner.

Our rendering framework in Section 4 extends Yee's work, modeling task-level saliency rather than automatic visual processes, and deriving a map of *error conspicuity* in place of error tolerance. This permits us to finish a frame when errors have become invisible, or render the best possible frame in a fixed period of time optimizations Yee's method does not support.

2.3. Inattentional Blindness

In 1967, the Russian psychologist Yarbus recorded the fixations and saccades observers made while viewing natural objects and scenes. Observers were asked to answer a number of different questions concerning the depicted situation in Repin's picture "An Unexpected Visitor" [30]. This resulted in substantially different saccade patterns, each one being easily construable as a sampling of those picture objects that were most informative for the answering of the question, as shown in Figure 2.

Cater et al. [1] showed that conspicuous objects in a scene that would normally attract the viewer's attention are ignored if they are not relevant to the task at hand. In their experiments, viewers were presented with two animations. One was a full, high-quality rendering, while in the other, only the pixels in visual angle of the fovea (2°) centered around the location of a task within the environment were rendered at high quality. This high quality was blended to a much lower quality in the rest of the image. They showed that when observers were performing the task within an animation, their visual attention was fixed exclusively on the area of the task, and they consistently failed to notice the significant difference in rendering quality between the two animations.

We have extended the work in Cater et al. [1] to be able to distinguish between the effects of peripheral vision and *inattentional blindness*, which is the failure of an observer to see unattended items in a scene [13]. We present our results in the following section, where we substitute still images for the animation to ensure that the observed effect is not merely a result of resolution loss in the periphery, but a true exhibition of inattentional blindness.

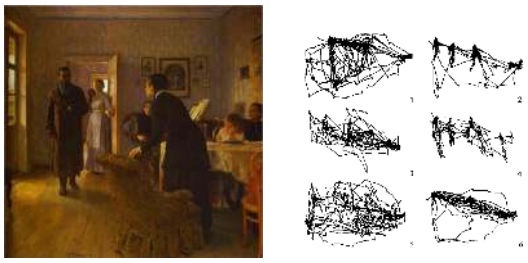


Figure 2: *Repin's picture was examined by subjects with different instructions; 1. Free viewing, 2. Judge their ages, 3. Guess what they had been doing before the unexpected visitor's arrival, 4. Remember the clothes worn by the people, 5. Remember the position of the people and objects in the room, 6. Estimate how long the visitor had been away* [30].

3. Task Maps: Experimental Validation

In this section, we demonstrate inattentional blindness experimentally in the presence of a high-level task focus. Our hypothesis was that viewers would not notice normally visible degradations in an image that did not affect the clarity of the objects we instructed them to seek. The experiments confirmed our hypothesis with a high level of certainty. An appropriate conjunctive search was selected as the task, with no pre-attentive cues, such as color, to differentiate the task objects from the other objects in the scene, this prevented any pop-out effects [24]. The task chosen for this experiment was to count the number of teapots in a computer generated scene. For ease of experimental setup a still image was used, however, previous work has proven that this method works just as well for animations [1].

A pre-study was run with 10 participants to find out how long subjects took to perform the task, this was found to be on average 2 seconds to count the five teapots in the image. A pilot study was then conducted to deduce the appropriate image resolution to use for the main experiment. 32 participants were shown 24 pairs of images at random, and asked if they could distinguish a change in resolution or quality between the two images. Each image was displayed for 2 seconds. One image was always the High Quality image rendered at a 3072x3072 sampling resolution, whilst the other image was one selected from images rendered at sampling resolutions of 256x256, 512x512, 768x768, 1024x1024, 1536x1536 and 2048x2048. In half of the pairs of images, there was no change in resolution; i.e., they saw two 3072x3072 resolution images. The results can be seen in Figure 3.

All the participants could easily detect a quality difference with the resolutions of 256x256 through to 1024x1024 in comparison to a resolution of 3072x3072. 72% still detected a quality difference between a resolution image of 1536x1536 and 3072x3072. However, it was decided that we would use a resolution of 1024x1024 in our main study as 100% of participants in the pilot study detected the difference.

The main study involved two models of an office scene, the only difference being the location of items in the scene, mainly teapots (Figure 4). Each scene was then rendered to three different levels of resolution quality, the entire scene at High Quality (HQ), a sampling resolution of 3072x3072 (Figure 5a), the entire scene at Low Quality (LQ), a sampling resolution of 1024x1024 (Figure 5b), and Selective Quality (SQ). The Selective Quality image was created by selectively rendering the majority of the scene in low quality (1024x1024) apart from the visual angle of the fovea (2°) centered on each teapot, shown by the black circles in Figure 4, which were rendered at the higher rate corresponding to 3072x3072 sampling. The high quality images took 8.6 hours to render with full global illumination in *Radiance* [26] on a 1 GHz Pentium processor, whilst the images for the low quality were rendered in half this time, and the Selective Quality in 5.4 hours.

In the study, a total of 96 participants were considered. Each subject saw two images, each displayed for 2 seconds. Table 1 describes the conditions tested with 32 subjects for the HQ/HQ condition and 16 subjects for the other conditions. We know from the pilot study that all participants should be able to detect the rendering quality difference if given no task; i.e., they are simply looking at the images for 2 seconds. The task chosen to demonstrate the effect of inattentional blindness had the subjects counting teapots located all around the scene. There were 5 teapots in both images. By placing the teapots all over the scene, we were able to see whether or not having to scan the whole image, and thus fixate on low quality as well as high quality regions, would mean that the viewers would indeed be able to detect the rendering quality difference. To minimize experimental bias, the choice of which condition to run was randomized, and for each 8 were run in the morning and 8 in the afternoon. Subjects had a variety of experience with computer graphics, and all exhibited normal or corrected vision in testing.

Before beginning the experiment, the subjects read a sheet of instructions on the procedure of the particular task they were to perform. After each participant had read the instructions, they were asked to clarify that they understood the task. They then placed their head on a chin rest that was located 45cm away from a 17-inch monitor. The chin rest was located so that their eye level was approximately level with the centre of the screen. The participants' eyes were allowed to adjust to the ambient lighting conditions before the experiment was begun. The first image was displayed for 2 seconds, then the participant stated out loud how many teapots they saw. Following this, the second image was displayed for 2 seconds, during which the task was repeated.

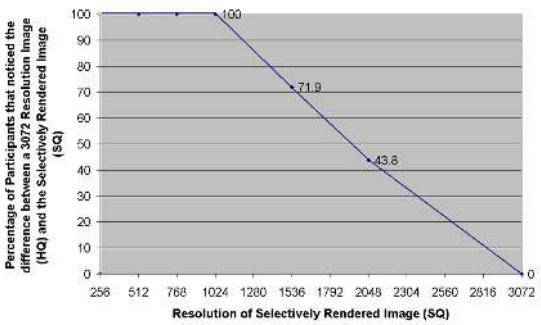


Figure 3: *Results from the pilot study: determining a consistently detectable rendering resolution difference.*

Figure 4: *Selective Quality (SQ) image showing the high quality rendered circles located over the teapots.*

On completion of the experiment, each participant was asked to fill out a detailed questionnaire. This questionnaire asked for some personal details including age, sex, and level of computer graphics knowledge. The participants were then asked detailed questions about the quality of the two images they had seen. Finally, the subjects were shown a high quality and a low quality image side-by-side and asked which one they saw for the first and second displayed images. This was to confirm that participants had not simply failed to remember that they had noticed a quality difference, but actually could not distinguish the correct image when shown it from a choice of two.

3.1. Results

Figure 6 shows the overall results of the experiment. Obviously, the participants did not notice any difference in the rendering quality between the two HQ images (they were

the same). Of interest is the fact that, apart from two cases in the HQ/SQ conditions, the viewers performing the task consistently failed to notice any difference between the HQ rendered image and the SQ image. Surprisingly, nearly 20% of the viewers in the HQ/LQ condition were so engaged in the task that they failed to notice any difference between these very different quality image.

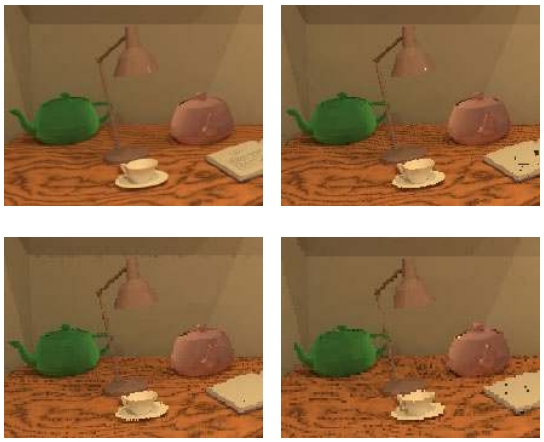


Figure 5: *Sampling resolutions: a(top left) 3072x3072 (HQ), b(top right) 1024x1024 (LQ), c(bottom left) 768x768 (LQ), d(bottom right) 512x512 (LQ)*

3.2. Statistical Analysis

Statistical analysis shows where our results are significant. The appropriate method of analysis is a "paired samples" *t-test for significance*, and since each subject had a different random selection of the images, an unrelated t-test was applied [2]. By performing comparisons of the other image pairings to the HQ/HQ data, we could determine whether the results were statistically significant.

When the observers were counting teapots, the difference between HQ/HQ and HQ/LQ counts were statistically very significant. For a two-tailed test with the $df = 62$ (df is related to the number of subjects), t must be greater than or equal to 2.0 for significance with $p < 0.05$ (less than 5% chance of random occurrence). The result for the pair-wise comparison of HQ/HQ and HQ/LQ was $t = 11.6$ with $p < 0.05$.

However, if we analyze statistics on the pair-wise comparison of HQ/HQ and HQ/SQ, the results are not statistically significant the *null hypothesis* is retained, as $t = 1.4$, $df = 62$, and $p > 0.1$. From this we can conclude that when observers were counting teapots, the HQ/HQ images and the HQ/SQ images produced the same result; i.e., the observers thought they were seeing the same pair twice, with no alteration in rendering quality. However, when the observers were simply looking at the images without searching for teapots in the pilot study, the result was significantly different; i.e., the observers could distinguish that they were shown two images rendered at different qualities.

An additional experiment was run to see at what value the results became significantly different from the HQ resolution of 3072x3072. At a sampling resolution of 768x768 (Figure 5c) the results were only just significant, $t = 2.9$, $df = 62$, and $p < 0.05$. I.e., only 7 participants, out of the 32 people studied, noticed the difference between the high quality image and a selectively rendered image whilst performing the teapot counting task. This only increased to 8 people out of 32 when the sampling resolution was dropped again to 512x512 (Figure 5d)!

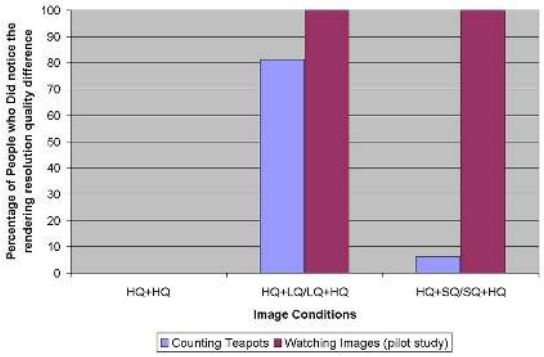


Figure 6: *Experimental results for the two tasks: counting the teapots vs. simply looking at the images.*

Acronym	Description
HQ	**High Quality:** Entire animation rendered at a sampling resolution of 3072x3072.
LQ	**Low Quality:** Entire animation rendered at a sampling resolution of 1024x1024.
SQ	**Selective Quality:** A sampling resolution of 1024x1024 all over the image apart from the visual angle of the fovea (2°) centered around each teapot, shown by the circles in Figure 4, which are rendered to a sampling resolution of 3072x2072.

Table 1: *The ordering image pairs shown in the experiment were: (1)HQ/HQ, (2)HQ/LQ, (3)LQ/HQ, (4)HQ/SQ, (5)SQ/HQ*

3.3. Verification with an Eye-tracker

To confirm that the attention of an observer was being fully captured by the task of counting teapots, the experiment was repeated using the Eyelink Eyetracking System developed by SR Research Ltd. and manufactured by SensoMotoric Instruments. Figure 7 shows an example of a scan path of an observer whilst performing the counting teapots task for 2

seconds. Whilst all the observers had slightly different scan paths across the images, they fixated both on the teapots and on other objects as well. The vases seemed to be the most commonly non-teapot object fixated upon, due to the fact they were the most similar looking item in the scene to a teapot. It could be deduced that the participants were making fixations on non-teapot objects in the image to make sure whether or not they were in fact a teapot, whatever the case these fixations were not enough for the observers to distinguish the different quality to which they were rendered.

Figure 8 shows the perceptual difference between the selective quality (SQ) and low quality (LQ) images computed using Daly's Visual Difference Predictor [?, 18]. The recorded eye-scan paths clearly cross, and indeed fixate, on areas of high perceptual difference. We can therefore conclude that the failure to distinguish the difference in rendering quality between the teapots, selectively rendered to high quality, and the other low quality objects, is *not* due purely to peripheral vision effects. The observers are fixating on low quality objects, but because they are not relevant to the given task of counting teapots, they fail to notice the reduction in rendering quality. This is inattentional blindness.

These results demonstrate that inattentional blindness, and not just peripheral vision, may be exploited to significantly reduce the rendered quality of a large portion of the scene without having any significant effect on the viewer's perception of the scene.

4. A Perceptual Rendering Framework

By our experiments, we know that selective rendering is cost effective for briefly viewed still images, and in fact task focus seems to override low-level visual attention when it comes to noticing artifacts. In the more general case of animated imagery, we can take even greater advantage of inattentional blindness, because we know the eye preferentially tracks salient objects at the expense of other details [1]. Using Daly's model of human contrast sensitivity for moving images [3], and Yee's insight to substitute saliency for movement-tracking efficacy [31], we can apply our *a priori* knowledge of task-level saliency to optimize the animation process.

The approach we describe has a number of key advantages over previous methods using low-level visual perception. First, task-level saliency is very quick to compute, as it is derived from a short list of important objects and their known whereabouts. Second, we have introduced a direct estimate of pixel error (or uncertainty), avoiding the need for expensive image comparisons and Gabor filters as required by other perceptually based methods [31, 16]. Third, we render animation frames progressively, enabling us to specify exactly how long we are willing to wait for each image, or stopping when the error has dropped below the visible threshold. Frames are still rendered in order, but the time spent refining the images is under our control. Our initial implementation of this framework is suitable for quick turnaround animations at about a minute per frame, but it is our eventual goal to apply these methods to interactive and real-time rendering [20, 25].

We have designed a general framework for progressive rendering that permits iterative frame refinement until a target accuracy or time allotment has been reached. A frame may be refined by any desired means, including improvements to resolution, anti-aliasing, level of detail, global illumination, and so forth. In our demonstration system, we focus primarily on resolution refinement (i.e., samples/pixel), but greater gains are possible by manipulating other rendering variables as well.

Figure 7: *An eye scan for an observer counting the teapots. The X's are fixation points and the lines are the saccades.*

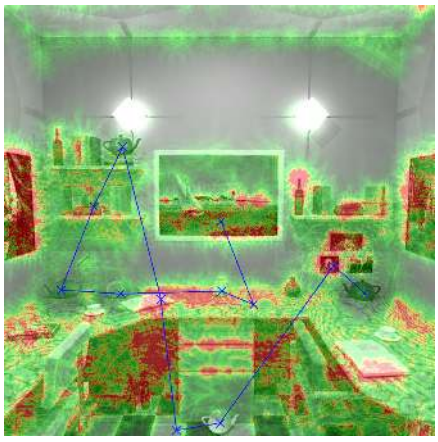


Figure 8: *Perceptual difference between SQ and LQ images using VDP [3]. Red denotes areas of high perceptual difference.*

4.1. Framework

The diagram shown in Figure 9 shows an overview of our system. The boxes represent data, and the ovals represent processes. The inputs to the system, shown in the upper left, are the viewer's known task, the scene geometry, lighting, and view, all of which are a function of time. The processes shown outside the "Iterate" box are carried out just once for each frame. The processes shown inside the box may be applied multiple times until the frame is considered "ready", by whatever criteria we set. In most cases, we call a frame ready when we have exhausted our time allocation, but we can also break from iteration when our *error conspicuity* (EC) drops below threshold over the entire image.

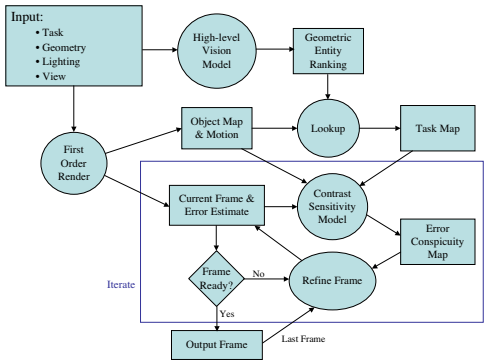


Figure 9: *A framework for progressive refinement of animation frames using task-level information.*

Our framework is designed to be general, and our implementation is just one realization. We start by explaining the basic methods that are applied once per frame, followed by the interactive methods for frame refinement. This overview pertains to any rendering algorithm one might use, from radiosity to ray-tracing to multi-pass hardware rendering. The Implementation section that follows details some of the specific techniques we used in our ray-tracing realization, and highlights our results.

Referring to Figure 9, our high-level vision model takes the task and geometry as input, and produces a table quantifying relative object importance for this frame. We call this the geometric entity ranking. Specifically, we derive a table of positive real numbers, where zero represents an object that will never be looked at, and 1 is the importance of scene objects unrelated to the task at hand. Normally, only task-relevant objects will be listed in this table, and their importance values will typically be between 1.5 and 3, where 3 is an object that must be followed very closely in order to complete the task.

For the first order rendering, we may use any method that is guaranteed to finish before our time is up. From this initial rendering, we will need an object map and depth value for each pixel. If subsampling is applied and some pixels are skipped, we must separately project scene objects onto a full resolution frame buffer to obtain this map. The pixel motion map, or image flow, is computed from the object map and our knowledge of object and camera movement relative to the previous frame. The object map is also logically combined with the geometric entity ranking to obtain the *task map*. This is usually accessed via a lookup into the ranking table, and does not require actual storage in a separate buffer.

Once we have a first order rendering of our frame and maps with the object ID, depth, motion, and task-level saliency at each pixel, we can proceed with image refinement. First, we compute the relative uncertainty in each pixel estimate. This may be derived from our knowledge of the underlying rendering algorithm, or from statistical measures of variance in the case of stochastic methods. We thought at first that this might pose a serious challenge, but it turns out to be a modest requirement, for the following reason. Since there is no point in quantifying errors that we cannot correct for in subsequent passes, we only need to estimate the difference between what we have and what we might get after further refinement of a pixel. For such improvements, we can usually obtain a reasonable bound on the error. For example, going from a calculation with a constant ambient term to one with global illumination, the change is generally less than the ambient value used in the first pass, times the diffuse material color. Taking half this product is a good estimate of the change we might see in either direction by moving to a global illumination result. Where the rendering method is stochastic, we can collect neighbor samples to obtain a reasonable estimate of the variance in each pixel neighborhood and use this as our error estimate [9]. In either case, error estimation is inexpensive as it only requires local information, plus our knowledge of the scene and the rendering algorithm being applied.

With our current frame and error estimate in hand, we can make a decision whether to further refine this frame, or finish it and start the next one. This "frame ready" decision may be based as we said on time limits or on some overall test of frame quality. In most cases, we will make at least one refinement pass before we move on, applying *image-based rendering* (IBR) to gather useful samples from the previous frame and add them to this one.

In an IBR refinement pass, we use our object motion map to correlate pixels from the previous frame with pixels from this frame. This improves our ability to decide when and where IBR is likely to be beneficial. We base our selection of replacement pixels on the following heuristics:

1. The pixel pair in the two frames corresponds to the same point on the same object, and does not lie on an object boundary.
2. The error estimate for the previous frame's pixel must be less than the error estimate for the current frame's pixel by some set amount. (We use 15%.)

3. The previous frame's pixel must agree with surrounding pixels in the new frame within some tolerance. (We use a 32% relative difference.)

The first criterion prevents us from using pixels from the wrong object or the wrong part of the same object. We test for position correspondence by comparing the transformed depth values, and for object boundaries by looking at neighboring pixels above, below, right, and left in our object map. The second criterion prevents us from degrading our current frame estimate with unworthy prior pixels. The third criterion reduces pollution in shadows and highlights that have moved between frames, though it also limits the number of IBR pixels we take in highly textured regions. If a pixel from the previous frame passes these three tests, we overwrite the current pixel estimate with the previous one, and reset the error to the previous value degraded by the amount used for the second criterion. In this way, IBR pixels are automatically retired as we move from one frame to the next.

Let us assume there is time for further refinement. Once we have transferred what samples we can using IBR, we determine which pixels have noticeable, or conspicuous, errors so we may select these for improvement. Here we combine the spatiotemporal *contrast sensitivity function* (CSF) defined by Daly [3] with our task-level saliency map. Daly's CSF model is a function of two variables, spatial frequency, ρ, and retinal velocity, v_R:

$$CSF(\rho, v_R) = k \cdot c_0 \cdot c_2 \cdot v_R \cdot (c_1 2\pi\rho)^2 exp(-\frac{c_1 4\pi\rho}{\rho_{max}}) \quad (1)$$

where:

$k = 6.1 + 7.3|\log(c_2 v_R/3)|^3$

$\rho_{max} = 45.9/(c_2 v_R + 2)$

$c_0 = 1.14, c_1 = 0.67, c_2 = 1.7$ for CRT at $100 cd/m^2$

Following Yee [31], we substitute saliency for movement-tracking efficacy, based on the assumption that the viewer pays proportionally more attention to task-relevant objects in their view. The equation for retinal image velocity (in$^\circ$/second) thus becomes:

$$v_R = |v_1 - \min(v_1 \cdot S/S_{max} + v_{min}, v_{max})| \quad (2)$$

where:

v_1 = local pixel velocity (from motion map)

S = task-level saliency for this region

S_{max} = max. saliency in this frame, but not less than 1/0.82

$v_{min} = 0.15^\circ$/sec (eye drift velocity)

$v_{max} = 80^\circ$/sec (movement-tracking limit)

The eye's movement tracking efficacy is computed as S/S_{max}, which assumes the viewer tracks the most salient object in view perfectly. Daly [3] recommends an overall value of 82% for the average efficacy when tracking all objects in a scene at once, so we do not allow S_{max} to drop below 1/0.82. This prevents us from predicting perfect tracking over the whole image when no task-related objects are in view.

Since peak contrast sensitivity shifts towards lower frequencies as retinal velocity increases, objects that the viewer is not tracking because they are not important will be visible at lower resolution than our task-relevant objects. However, if the entire image is still or moving at the same rate, the computed CSF will be unaffected by our task information. Because of this, we reintroduce our task map as an additional multiplier in the final error conspicuity map, which we define as:

$$EC = S \cdot \max(E \cdot CSF/ND - 1, 0) \quad (3)$$

where:

E =relative error estimate for this pixel

ND = noticeable difference threshold

Because the relative error multiplied by the CSF yields the normalized contrast, where 1.0 is just noticeable, we introduce a threshold difference value, ND, below which we deem errors to be insignificant. A value of 2 JNDs is the threshold where 94% of viewers are predicted to notice a difference, and this is the value commonly chosen for ND.

To compute the CSF, we also need an estimate of the peak stimulus spatial frequency, ρ. We obtain this by evaluating an image pyramid. Unlike previous applications of the CSF to rendering, we are not comparing two images, so we do not need to determine the relative spatial frequencies in a difference image. We only need to know the uncertainty in each frequency band to bound the visible difference between our current estimate and the correct image. This turns out to be a great time-saver, as it is the evaluation of Gabor filters that usually takes longest in other approaches. Because the CSF falls off rapidly below spatial frequencies corresponding to the foveal diameter of 2°, and statistical accuracy improves at lower frequencies as well, we need only compute our image pyramid up to a ρ of 0.5 cycles/degree.

Our procedure is as follows. We start by clearing our EC map, and subdividing our image into 2° square cells. Within each cell, we call a recursive function that descends a local image pyramid to the pixel level, computing EC values and summing them into our map on the return trip. At each pyramid level, the EC function is evaluated from the stimulus frequency (1/subcell radius in $^\circ$), the task-level saliency, the combined error estimate, and the average motion for pixels within that subcell. The task-level saliency for a subcell is determined as the maximum of all saliency values within a 2° neighborhood. This may be computed very quickly using a 4-neighbor check at the pixel level, where each pixel finds the maximum saliency of itself and its neighbors 1° up, down, left, and right. The saliency maximum and statistical error sums are then passed back up the call tree for the return evaluation. The entire EC map computation, including a statistical estimation of relative error, takes less than a second for a 640x480 image on a 1 GHz Pentium processor.

5. Implementation

In our implementation of the above framework, we modified the *Radiance* lighting simulation and rendering engine [26] to perform progressive animation. Figure 10 shows a frame from a 4-minute long animation we computed at 640x480 resolution using this software. Figure 11a shows our estimate of relative error at each pixel in the first order rendering, and Figure 11b shows the corresponding error conspicuity map. The viewer was assigned the task of counting certain objects in the scene related to fire safety. There are two task objects visible in this image, the fire extinguisher and the narrator's copter (the checkered ball), so the regions around these objects show strongly in the conspicuity map. Figure 12a shows the final number of samples taken at each pixel in the refined frame, which took two minutes to compute on a single 400 MHz G3 processor. We found this time sufficient to render details on the task-related objects, but too short to render the entire frame accurately. We wanted there to be artifacts in order to demonstrate the effect of task focus on viewer perception. About 50% of the pixels received IBR samples from the previous frame, and 20% received one or more high quality refinement samples.

For comparison, Figure 12b shows the scene rendered as a still image in the same amount of time. Both images contain artifacts, but the animation frame contains fewer sampling errors on the task-related objects. In particular, the fire extinguisher in the corner, which is one of the search objects, has better anti-aliasing than the traditionally rendered image. This is at the expense of some detail on other parts of the scene, such as the hatch door. Since the view is moving down the corridor, all objects will be in motion, and we assume the viewer will be tracking the task-related objects more than the others. Rendering the entire frame to the same detail as the task objects in Figure 10 takes *7 times longer* than our optimized method. Although direct comparisons are difficult due to differences in the rendering aims, Yee et al. demonstrated a 4-10 times speedup in [31] and Myszkowski et al. showed a speedup of roughly 3.5 times in [17]. This shows that we are able to achieve similar speedups controlling only rendered sampling resolution. If we were to refine the global illumination calculation also, similar to Yee, we could achieve even greater gains.

There are only a few aspects of our framework that we must tailor to a ray-tracing approach. Initially, we compute a low quality, first order rendering from a quincunx sampling of the image plane, where one out of every 16 pixels is sampled. (This sampling pattern is visible in unrefined regions of Figure 12a.) To obtain the object and depth maps at unsampled locations, we cast rays to determine the first intersected object at these pixels. We then estimate our rendering error by finding the 5 nearest samples to each pixel position, and computing their standard deviation. This is a very crude approximation, but it suited our purposes well. In cases where the high-quality samples in the refinement pass have an interreflection calculation that the initial samples do not, we use the method described earlier for estimating the error due to a constant ambient term.

Following the IBR refinement described in the previous section, and provided we are not out of time, we then compute the error conspicuity map, sorting our pixels from most to least conspicuous. For pixels whose EC value are equal (usually 0), we order from highest to lowest error, then from fewest to most samples. Going down this list, we add one high-quality ray sample to each pixel, until we have sampled them all or run out of time. If we manage to get through the whole list, we recompute the error conspicuity map and re-sort. This time, we only add samples to the top 1/8th of our list before sorting again. We find we get smoother animations by sampling each pixel at least once before honing in on the regions we deem to be conspicuous. We could insist on sampling every pixel in our first order rendering, but this is sometimes impossible due to time constraints. Therefore, we incorporate it in our refinement phase, instead.

Figure 10: *A frame from our task-based animation.*

Prior to frame output, we perform a final filtering stage to interpolate unsampled pixels and add motion blur. Pixels that did not receive samples in the first order rendering or subsequent refinements must be given a value prior to output. We apply a Gaussian filter kernel whose support corresponds to our initial sample density to arrive at a weighted average of the 4 closest neighbors. Once we have a value at each pixel, we multiply the object motion map by a user-specified blur parameter, corresponding to the fraction of a frame time the virtual camera's shutter is open. The blur vector at each pixel is then applied using an energy-preserving smear filter to arrive at the final output image. This technique is crude in the sense that it linearizes motion and does not discover obstructed geometry, but we have not found this to be objectionable in any of our tests. However, the lack of motion blur on shadows does show up as one of the few distracting artifacts in our implementation. This filtering operations take a small fraction of a CPU second per video resolution frame, and are inconsequential to the overall rendering time.

Of our two minute rendering time for the frame shown

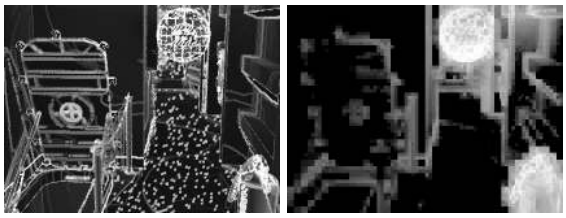


Figure 11: *a(left) Initial frame error, b(right) Initial error conspicuity.*

Figure 12: *a(left) Final frame samples, b(right) Standard rendering taking same time as Figure 10.*

in Figure 10, 1 second is spent updating the scene structures, 25 seconds is spent computing the 19,200 initial samples and the object map, 0.25 seconds is spent on IBR extrapolation, 0.9 seconds to compute the error map (times three evaluations), 1.25 seconds for the EC map, 0.4 seconds for filtering, and the remaining 90 seconds to compute about 110,000 high quality refinement samples. In this test, the Radiance rendering parameters were set so there was little computational difference between an initial sample and a high-quality refinement sample; we did not evaluate diffuse interreflections for either. Our method's combined overhead for a 640x480 frame is thus in the order of 14 seconds, 10 of which are spent computing the object map by ray casting. Intuitively and by our measurements, this overhead scales linearly with the number of pixels in a frame.

It is worth noting that IBR works particularly well in our progressive rendering framework, allowing us to achieve constant frame generation times over a wide range of motions. When motion is small, IBR extrapolation from the previous frame provides us with many low-error samples for our first refinement pass. When motion is great, and thus fewer extrapolated samples are available, the eye's inability to track objects and the associated blur means we do not need as many. This holds promise for realistic, real-time rendering using this approach with hardware support.

6. Conclusions and Future Work

As our experiments demonstrate, inattentional blindness may be exploited to accelerate rendering by reducing quality in regions that are unrelated to a given task. Extending this idea, we have designed a progressive animation framework that combines an indexed *task map* with a spatiotemporal *contrast sensitivity function* to determine which image areas need further refinement. Adding our knowledge of pixel uncertainty and movement between frames, we derive an *error conspicuity map*, which identifies noticeable artifacts in the presence of this task. We focus additional ray samples in these regions, and augment our results with IBR samples from the previous frame. We then apply the pixel movement map again to simulate motion blur in the final output.

Much work remains. Our current implementation performs poorly when subsequent refinement corrects for systematic errors in the initial estimate. This may result in noticeable discontinuities in the output, which makes it difficult to employ rendering methods that do not converge smoothly. Some intelligent blending or error dissipation is required if we wish to combine hardware rendering with ray-tracing, for example. At the level of the perceptual model, we would like to take advantage of masking effects to further reduce sampling in busy regions [5]. However, visual masking models have yet to be extended to the temporal domain, even though we know they are affected by movement. We would also like to find a sensible way to combine task-level information with low-level saliency. To apply them together, we need to know which visual processes dominate and under what conditions. Again, additional psychophysical research is required.

Human perception determines to a large extent what we do in computer graphics and indeed, why we do it. It seems fitting, therefore, that we should pay close attention to the attention graphics consumers pay to us. Exploiting task-level models of visual perception is one way to improve the viewing experience within a limited budget of time and resources.

Acknowledgements

This research is funded by the Engineering and Physical Sciences Research Council (Award No: 00301786). We are indebted to Tom Troscianko, Karol Myszkowski, Hector Yee and Scott Daly for all their help. We would also like thank everyone who attended the experiment; we would not have got any results if it were not for them.

References

1. K. Cater, A. Chalmers, and P. Ledda. Selective Quality Rendering by Exploiting Human Inattentional Blindness In *Symposium on VRST* 2002, ACM. pp. 17–24.

2. H. Coolican. *Research Methods and Statistics in Psychology*. Hodder & Stoughton Educational, U.K., 1999.

3. S. Daly. Engineering observations from spatiovelocity and spatiotemporal visual models. Chapter 9 in *Vision Models and Applications to Image and Video Processing*, 2001, Kluwer Academic Publishers.

4. J.A. Ferwerda, S.N. Pattanaik, P.S. Shirley, and D.P. Greenberg. A Model of Visual Adaptation for Realistic Image Synthesis. In *SIGGRAPH 1996*, ACM, pp. 249–258.

5. J.A. Ferwerda, S.N. Pattanaik, P.S. Shirley, and D.P. Greenberg. A Model of Visual Masking for Computer Graphics. In *SIGGRAPH 1997*, ACM, pp. 143–152.

6. D.P. Greenberg, K.E. Torrance, P.S. Shirley, J. Arvo, J.A. Ferwerda, S.N. Pattanaik, A.E. Lafortune, B. Walter S-C. Foo and B. Trumbore. A Framework for Realistic Image Synthesis. In *SIGGRAPH 1997*, ACM, pp. 477–494.

7. L. Itti and C. Koch. A saliency-based search mechanism for overt and covert shifts of visual attention. In *Vision Research*, 2000, Vol. 40, No. 10-12.

8. W. James A saliency-based search mechanism for overt and covert shifts of visual attention. *Principles of Psychology*, 1890, New York: Holt.

9. M. Lee, R. Redner and S. Uselton. Statistically Optimized Sampling for Distributed Ray Tracing. In *SIGGRAPH 1985*, ACM, Vol. 19, No. 3.

10. L.C. Loschky, G.W. McConkie, J. Yang and M.E. Miller. Perceptual Effects of a Gaze-Contingent Multi-Resolution Display Based on a Model of Visual Sensitivity. In *Advanced Displays and Interactive Displays Fifth Annual Symposium*, 2001, pp. 53–58.

11. D. Luebke and B. Hallen. Perceptually driven simplification for interactive rendering. In *12th Eurographics Workshop on Rendering*, 2001, pp. 221–223.

12. P.W.C Maciel and P. Shirley. Visual Navigation of Large Environments Using Textured Clusters. In *Symposium on Interactive 3D Graphics*, 1995, pp. 95–102.

13. A. Mack and I. Rock. Inattentional Blindness. MIT Press, 1998.

14. G. Marmitt and A.T. Duchowski Modeling Visual Attention in VR: Measuring the Accuracy of Predicted Scanpaths. *Eurographics 2002*, Short Presentations.

15. A. McNamara, A.G. Chalmers, T. Troscianko and I. Gilchrist. Comparing Real and Synthetic Scenes using Human Judgements of Lightness. In *12th Eurographics Workshop on Rendering 2000*, pp. 207–219.

16. K. Myszkowski, T. Tawara, H. Akamine and H-P. Seidel. Perception-Guided Global Illumination Solution for Animation Rendering. In *SIGGRAPH 2001*, ACM, pp. 221–230.

17. K. Myszkowski, R. Przemyslaw and T. Tawara. Perceptually-informed Accelerated Rendering of High Quality Walkthrough Sequences. In *Eurographics Workshop on Rendering 1999*, pp. 13–26.

18. K. Myszkowski. The Visible Differences Predictor: Applications to global illumination problems. In *Eurographics Workshop on Rendering 1998*, pp. 223–236.

19. C. O'Sullivan, J. Dingliana, G. Bradshaw, and A. McNamara. Eye-tracking for Interactive Computer Graphics. In *11th European Conference on Eye Movements (ECEM 11)*, 2001, Finland.

20. S. Parker, W. Martin, P-P. Sloan, P. Shirley, B. Smits, and C. Hansen. Interactive ray tracing. In *Interactive 3D Graphics*, 1999, ACM, pp. 119–126.

21. S.N. Pattanaik, J.A. Ferwerda, M.D. and D.P. Greenberg. A Multiscale Model of Adaptation and Spatial Vision for Realistic Image Display. In *SIGGRAPH 1998*, ACM, pp.287–298.

22. M. Ramasubramanian, S.N. Pattanaik, and D.P. Greenberg. A Perceptually Based Physical Error Metric for Realistic Image Synthesis. In *SIGGRAPH 1999*, ACM, pp. 73–82.

23. M. Reddy. Perceptually Modulated Level of Detail for Virtual Environments. *Ph.D. Thesis*, University of Edinburgh, 1997.

24. A. Treisman and J. Souther. Search asymmetry: A diagnostic for preattentive processing of separable features. In *Journal of Experimental Psychology: General*, 1985, 114 (3), pp. 285–310.

25. I. Wald, T. Kollig, C. Benthin, A. Keller and P. Slusallek. Interactive global illumination using fast ray tracing. In *13th Eurographics Workshop on Rendering 2002*, Springer-Verlag, pp. 9–19.

26. G. Ward. The RADIANCE Lighting Simulation and Rendering System. In *SIGGRAPH 1994*, ACM, pp. 459–472.

27. B. Watson, A. Friedman and A. McGaffey. An evaluation of Level of Detail Degradation in Head-Mounted Display Peripheries. In *Presence*, 6, 6, pp. 630–637.

28. B. Watson, A. Friedman and A. McGaffey. Measuring and Predicting Visual Fidelity. In *SIGGRAPH 2001*, ACM, pp. 213–220.

29. S. Yantis. Attentional capture in vision. In *Converging operations in the study of selective visual attention*, American Psychological Association, pp. 45–76.

30. A.L. Yarbus. Eye movements during perception of complex objects. In *Eye Movements and Vision*, 1967, Plenum Press, New York, Chapter VII, pp. 171–196.

31. H. Yee, S. Pattanaik and D.P. Greenberg. Spatiotemporal sensitivity and Visual Attention for efficient rendering of dynamic Environments. In *ACM Transactions on Computer Graphics*, 2001, Vol. 20, No. 1, pp. 39–65.

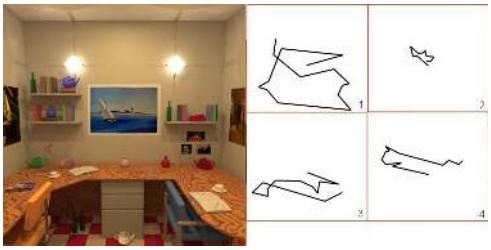
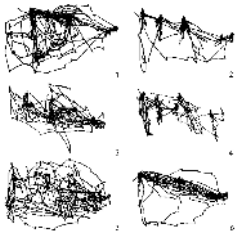


Figure 1: Effects of a task on eye movements. Eye scans for observers examined with different task instructions; 1. Free viewing, 2. Remember the central painting, 3. Remember as many objects on the table as you can, 4. Count the number of books on the shelves.

Figure 2: Repin's picture was examined by subjects with different instructions; 1. Free viewing, 2. Judge their ages, 3. Guess what they had been doing before the unexpected visitor's arrival,



4. Remember the clothes worn by the people, 5. Remember the position of the people and objects in the room, 6. Estimate how long the visitor had been away [30].

Figure 4: Selective Quality (SQ) image showing the high quality rendered circles located over the teapots.

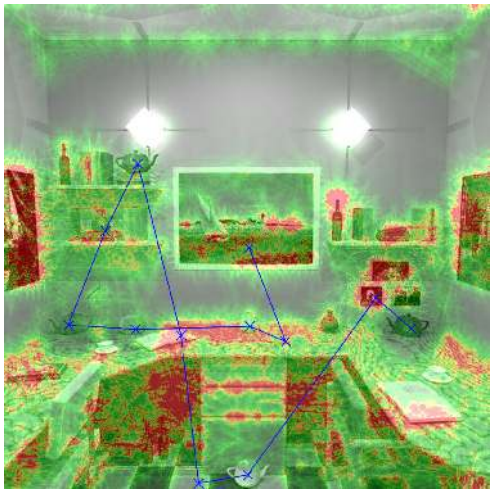


Figure 9: Perceptual difference between SQ and LQ images using VDP ?. Red denotes areas of high perceptual difference.

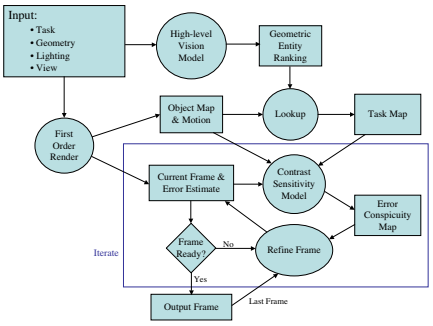


Figure 10: A framework for progressive refinement of animation frames using task-level information.

Figure 11: A frame from our task-based animation.