Gaze-contingent real-time simulation of arbitrary visual fields

Jeffrey S. Perry* and Wilson S. Geisler**
Center for Perceptual Systems, University of Texas at Austin

ABSTRACT

We describe an algorithm and software for creating variable resolution displays in real time, contingent upon the direction of gaze. The algorithm takes as input a video sequence and an arbitrary, real-valued, two-dimensional map that specifies a desired amount of filtering (blur) at each pixel location relative to direction of gaze. For each input video image the follow operations are performed: (1) the image is coded as a multi-resolution pyramid, (2) the gaze direction is measured, (3) the resolution map is shifted to the gaze direction, (4) the desired filtering at each pixel location is achieved by interpolating between levels of the pyramid using the resolution map, and (5) the interpolated image is displayed. The transfer function associated with each level of the pyramid is calibrated beforehand so that the interpolation produces exactly the desired amount of filtering at each pixel. This algorithm produces precision, artifact-free displays in 8-bit grayscale or 24-bit color. The software can process live or prerecorded video at over 60 frames per second on ordinary personal computers without special hardware. Direction of gaze for each processed video frame may be taken from an eye-tracker, from a sequence of directions saved on disk, or from another pointing device (such as a mouse). The software is demonstrated by simulating the visual fields of normals and of patients with low vision. We are currently using the software to precisely control retinal stimulation during complex tasks such as extended visual search.

Keywords: variable resolution imaging, gaze contingent displays, real time image processing, real time simulation, low vision, eye disease, foveated imaging

1. INTRODUCTION

The human visual system implements an elegant comprise between the competing goals of maximizing field of view, maximizing spatial resolution, and minimizing neural resources: It encodes a large field of view using a retina with variable spatial resolution, and then when necessary, uses high-speed eye movements to direct the highest-resolution region of the retina (the fovea) at specific points in the visual scene. Although this is an elegant design, the combination of a foveated retina and frequent eye movements makes it difficult to precisely control the retinal stimulus during complex, ecologically relevant tasks such as visual search, visual navigation, and reading. And, without control over the retinal stimulus it is difficult to test hypotheses concerning the perceptual and neural mechanisms underlying performance in such tasks. Gaze contingent displays offer a powerful method for precisely controlling retinal stimulation during the performance of complex tasks[1-3]. In a gaze contingent display, the displayed image is updated continuously in real time, contingent on current gaze direction, which most typically would be measured with an eye-tracking device.

There are many possible dimensions along which stimulus information might be varied in a gaze contingent display. The dimension of greatest interest has been spatial resolution. Early real time variable resolution display systems[4-9] suffered from blocking artifacts, limited display sizes, or limited control of resolution (e.g., only a high resolution area of interest and a low resolution background area). Several years ago we described a multi-resolution pyramid algorithm and software for creating variable resolution displays in real time using general-purpose computers[10, 11]. This Windows 95/98/2000/NT compatible software (available at the website, www.svi.cps.utexas.edu) produces smooth nearly artifact-free images at high frame rates, but it is designed primarily for image compression, has some visible artifacts, and is limited to displays that mimic the fall-off in resolution of the human visual system.

Here, we describe an extension of the previous algorithm that produces substantially higher image quality and allows completely arbitrary variable resolution displays. The software implementing the new algorithm produces artifact-free

* jsp@mail.utexas.edu; phone 512-471-3054; Center for Perceptual Systems, Computer Science Department, University of Texas, Austin, TX 78712; ** geisler@psy.utexas.edu; phone 512-471-5380; Center for Perceptual Systems, Psychology Department, University of Texas, Austin, TX 78712

gaze-contingent video at high frame rates in either 8-bit gray scale or 24-bit color. In addition, the new algorithm is sufficiently general that minor software enhancements will allow the software to generate space-variant gaze-contingent displays for other dimensions, such as contrast, color, and noise level. The software is available at the web site: www.svi.cps.utexas.edu.

There are a number of potential applications for gaze contingent variable resolution displays. One area of application is research on the roles of central and peripheral vision in tasks such as search, navigation and reading. For example, with such displays it might be possible to determine, for any one of these tasks, which patterns of variation in spatial resolution across the visual field can be tolerated without decreasing performance. Another area of application is in education. For example, gaze-contingent variable resolution displays that mimic the visual field defects produced by eye disease could be used to educate students, physicians and patients' family members about the perceptual and performance consequences of specific forms of vision loss in tasks such as navigation, reading or watching television. Another obvious application is in low bandwidth video communications. The spatial resolution of the human visual system falls off precipitously from the direction of gaze and hence it is possible to reduce considerably the spatial resolution of video images away from the line of sight, and hence the number of bits that must be transmitted, with little effect on perceived image quality or task performance.

2. METHODS

The algorithm requires three inputs: a sequence of images, the direction of gaze for each image, and an arbitrary resolution map that specifies the spatial resolution across the visual field. The following steps summarize the algorithm:

1. Compute blending functions from the resolution map.
2. Obtain the input image.
3. Obtain the current gaze direction.
4. Compute the multi-resolution pyramid for the input image.
5. Shift the resolution map and blending functions to align with the current gaze direction.
6. Up-sample, interpolate and blend the multi-resolution pyramid using the shifted blending functions.
7. Display the output image.
8. Go to step 2.

The steps in this algorithm are the same as in our previous algorithm[10, 11]. The important enhancement is that the blending between levels of the pyramid is controlled by blending functions derived from the resolution map. These blending functions allow precise specification and control of the resolution at each pixel location.

2.1. Resolution map

The resolution map is a two-dimensional function $R(x,y)$ of values between 0.0 and 1.0 that specify the desired spatial resolution relative to the direction of gaze. By spatial resolution, we mean a level of blur applied to the image by low pass filtering. More specifically, the spatial resolution at a given pixel location is defined to be the half-height bandwidth of the transfer function associated with the low-pass filter applied at the given pixel location.

For each image frame, the resolution map is brought into correspondence with the image by shifting the map so that the pixel at the center of the map aligns with the image pixel located at the direction of gaze. In order for the resolution map to cover all of the image pixels, for all possible directions of gaze within the boundaries of the display screen, the dimensions of the resolution map are set to twice those of the image. The spatial resolution values specified in the resolution map are completely arbitrary.

Two example resolution maps are shown in Figure 1. We have used these maps to simulate the vision in two different individuals, one with normal vision and one with glaucoma. These resolution maps were obtained by interpolating "visual field" measurements obtained with a Goldmann perimeter.

The desired spatial resolution at each pixel location, specified by the resolution map, is implemented by blending between levels of a multi-resolution pyramid computed from each input image.

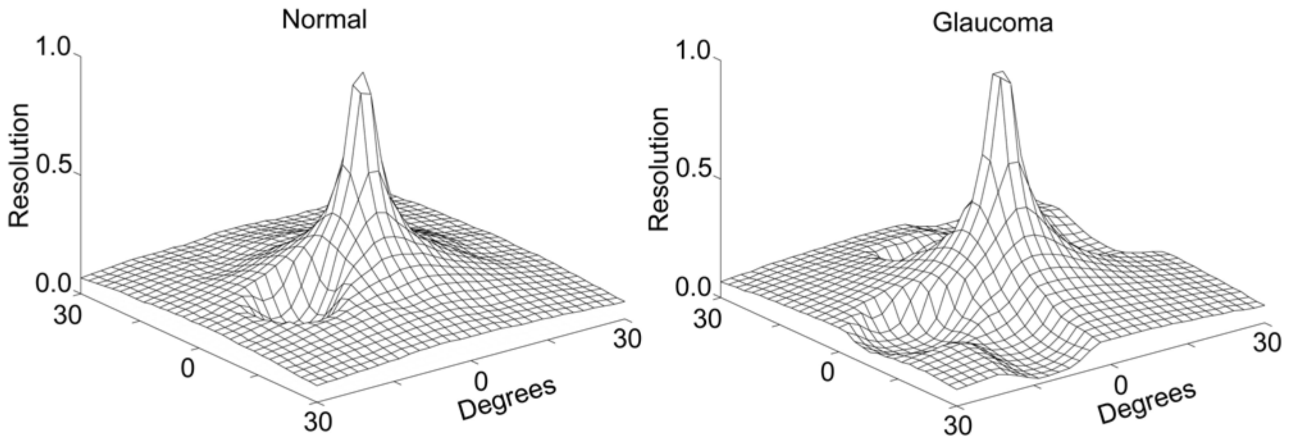

Figure 1. Example resolution maps representing relative spatial acuity as a function of retinal eccentricity for a normal individual and for a patient with glaucoma. These maps were estimated from the "visual fields" measured with a Goldmann perimeter. The dip in the normal map represents the "blind spot" where the optic nerve leaves the eye.

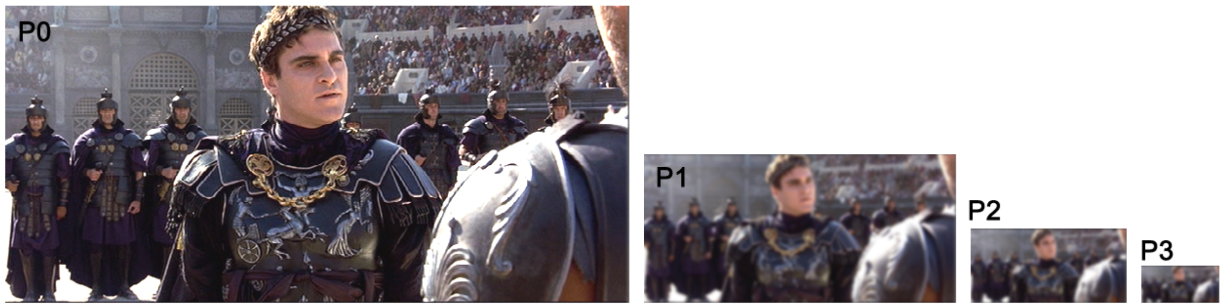

Figure 2. First four levels of a Gaussian multi-resolution pyramid.

2.2. Multi-resolution pyramid

The multi-resolution pyramid is illustrated in Figure 2. The first level of the pyramid, P0, is the original input image. The next level of the pyramid, P1, is computed by filtering the original image (i.e., convolving with a small weighting function) and then down-sampling this blurred image by a factor of two in both dimensions. This operation of blurring and down-sampling is applied recursively; P2 is computed by blurring and down-sampling P1, P3 is computed by blurring and down-sampling P2, and so on, up to the desired number of pyramid levels. To make this computation efficient, the simulation software uses the 3 by 3 "Gaussian" weighting function popular for computing the Laplacian pyramid[12]. The operation of blurring prior to down-sampling is necessary to prevent aliasing artifacts in the output images generated from the pyramid.

The number of levels of the pyramid determines the maximum amount of blur (the largest reduction in resolution) possible in the output image—the greater the number of levels, the greater the range of possible resolutions. In general, there is little cost in real-time performance (or in storage) of increasing the number of pyramid levels, because the number of pixels that must be processed to compute the next pyramid level decreases exponentially as a function of pyramid level. For most applications, six pyramid levels are adequate. In addition, we have also included a switch that can be used to make a resolution of zero correspond to the mean of the entire image (see later).

For image sequences that are stored on disk, the multi-resolution pyramid for each image may be pre-computed just once and stored in memory, in order to slightly increase real-time performance. However, the pyramid computation is so efficient that we typically perform the computation on the fly for each input image, just as we must do with live video.

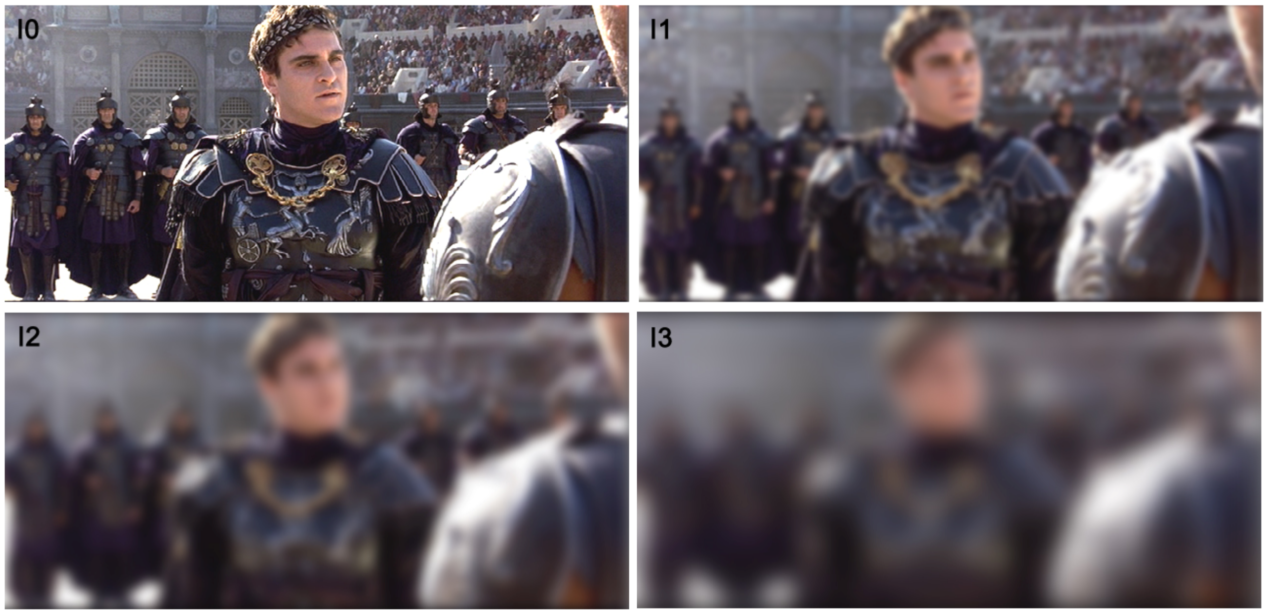

Figure 3. Up-sampled and interpolated images for the first four levels of the Gaussian multi-resolution pyramid shown in Figure 2.

Each level of the pyramid corresponds to a particular amount of blur, when the image at that level is expanded back to full size. Figure 3 shows the expanded image for each level of the pyramid in Figure 2. The expansion is accomplished by recursively up-sampling and interpolating by a factor of two in both dimensions until the full image size is reached[10, 12]. Although the operations of down-sampling and up-sampling are nonlinear, the entire process can be described by an effective transfer function, which shows the level of amplitude attenuation as a function of spatial frequency. For example, the curves labeled P0 to P3 in Figure 4 show the transfer function associated with the first four levels of the pyramid. We measured the transfer functions for all pyramid levels above P0 (including those above P3) by processing sine wave gratings of various spatial frequencies and then taking the ratio of the amplitude of the sine wave in the input image to the amplitude of the sine wave in the output (expanded) image. The transfer functions are closely approximated by Gaussian functions centered at the origin. For levels P1-P5 the standard deviations of the Gaussian functions are 0.124, 0.056, 0.0267, 0.0131, and 0.00654 cycles/pixel, respectively. A "transfer function" for level P0 is needed for the blending operation described below. For this level, we assumed that the standard deviation was 0.248 cycles/pixel, which is equivalent to assuming that the input image is an image in a larger pyramid. (Through out this paper, transfer functions are specified on a relative spatial frequency axis, where a value of 1.0 corresponds to 0.25 cycles/pixel.)

2.3. Blending functions

Each level of the multi-resolution pyramid after expansion (e.g., Figure 3) represents a certain fixed spatial resolution that is described by the transfer function for that level of the pyramid. We define the spatial resolution to be the spatial frequency where the transfer function is at half maximum (i.e., the spatial frequency where the horizontal line at 0.5 intersects the appropriate transfer function in Figure 4). Let R_i represent the fixed spatial resolution for the ith level of the pyramid. To render an arbitrary desired resolution, at any given pixel location, we first determine which two pyramid levels have resolutions that bracket the desired resolution, and then average the pixel values in the images from these two pyramid levels. The relative weights placed on the two pixels values in computing the average are given by a blending function. To define blending functions, we note that the spatial resolution associated with each level of the pyramid "slices" the resolution map at a particular resolution value (height) parallel to the horizontal plane (see Figure1).

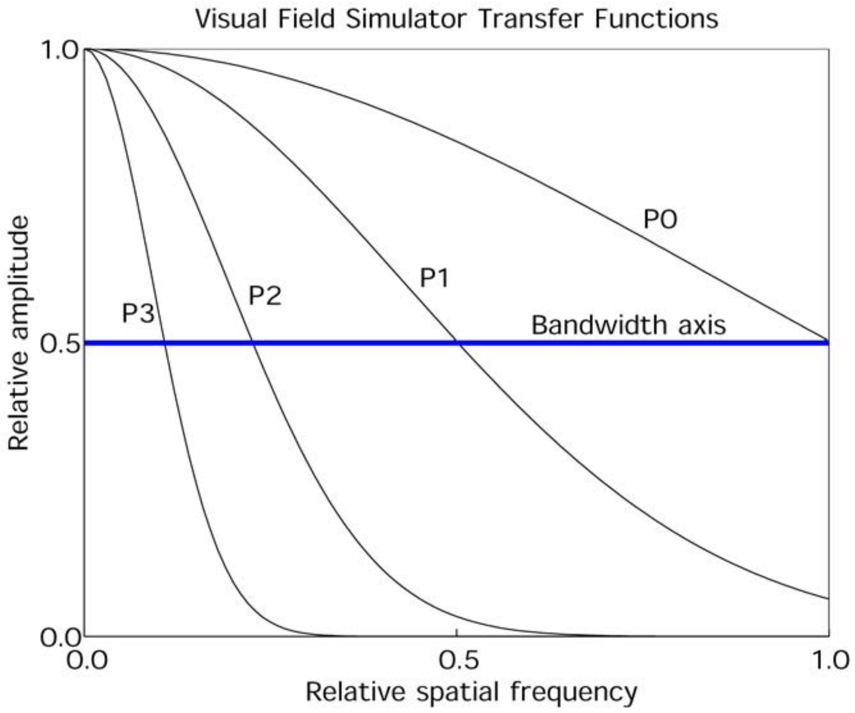

Figure 4. Transfer functions for the first four levels of a Gaussian multi-resolution pyramid.

The values of the resolution map that lie between adjacent slices are used to define the blending functions, one blending function for each adjacent pair of slices. Let $B_i(x,y)$ be the blending function defined by the slices at heights R_i and R_{i-1}. The blending function is defined by the following equation:

$$B_i(x,y)=\begin{cases} 0 & R(x,y)\le R_i \\ \dfrac{0.5-T_i\big(R(x,y)\big)}{T_{i-1}\big(R(x,y)\big)-T_i\big(R(x,y)\big)} & R_i < R(x,y) < R_{i-1} \\ 1 & R(x,y)\ge R_{i-1} \end{cases} \tag{1}$$

where T_i is the transfer function for the ith level of the pyramid. For pixels where the value of the resolution map is less than or equal to R_i the blending function is given a value of 0.0. For pixels where the value of resolution map is greater than or equal to R_{i-1}, the blending function is given a value of 1.0. For all other pixels, the blending function has a value between 0.0 and 1.0. Note that as $R(x,y)$ approaches R_{i-1}, $T_{i-1}\big(R(x,y)\big)$ approaches 0.5 and hence the value of the blending map approaches 1.0, and as $R(x,y)$ approaches R_i, $T_i(R(x,y))$ approaches 0.5 and the value of the blending map approaches 0.0.

The upper left panel in Figure 5 shows a gray scale plot of the resolution map of the glaucoma patient shown in Figure 1. The other panels in Figure 5 show gray scale plots of the blending functions: B1 is used to blend pixels between pyramid levels P0 and P1, B2 is used to blend pixels between pyramid levels P1 and P2, and B3 is used to blend pixels between pyramid levels P2 and P3.

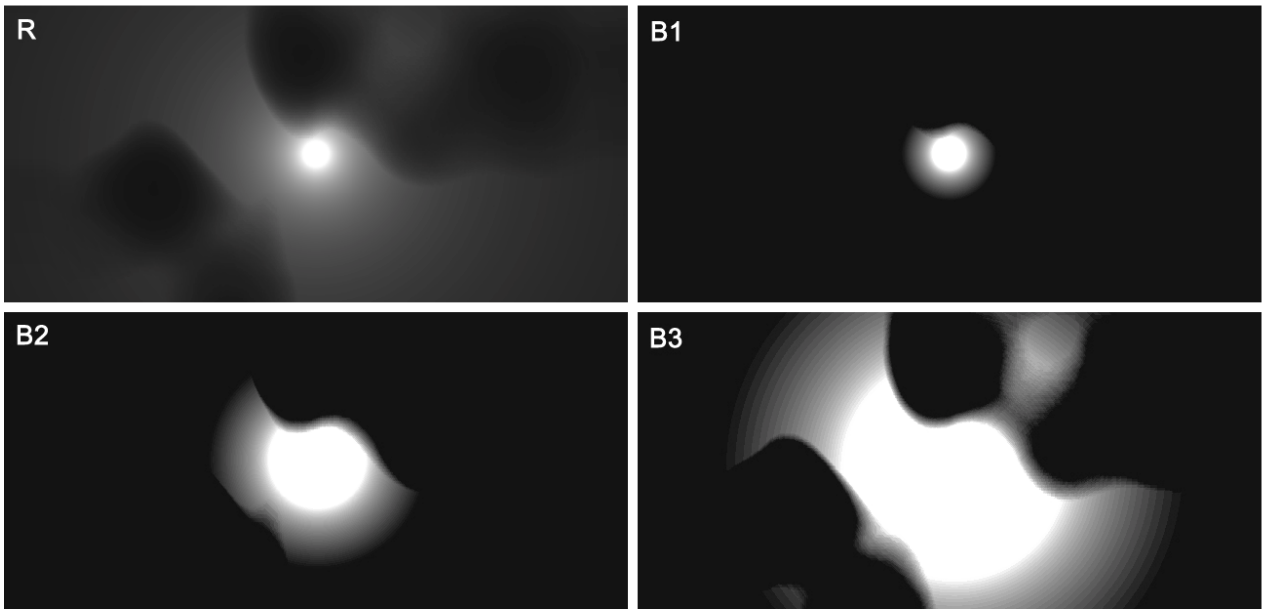

Figure 5. Glaucoma resolution map, R. Blending functions for the first four levels of the multi-resolution pyramid, B1-B3.

For pixels where the blending function is between 0.0 and 1.0 the output image is given by

$$O(x,y) = B_i(x,y)I_i(x,y) + (1 - B_i(x,y))I_{i-1}(x,y) \qquad (2)$$

where $I_i(x,y)$ and $I_{i-1}(x,y)$ are the expanded images for pyramid levels i and i-1. Equations (1) and (2) guarantee the desired resolution at each pixel location in the output image. Specifically, the effective transfer function at pixel location x,y has a half-height bandwidth of exactly $R(x,y)$. For example, the horizontal position of the diamond in Figure 6 represents a particular arbitrary value of the resolution map, and the dashed curve shows the effective transfer function that results by averaging pyramid images P2 and P3, using the blending function value from B3. As can be seen, the effective transfer function passes through the desired resolution (given by the diamond).

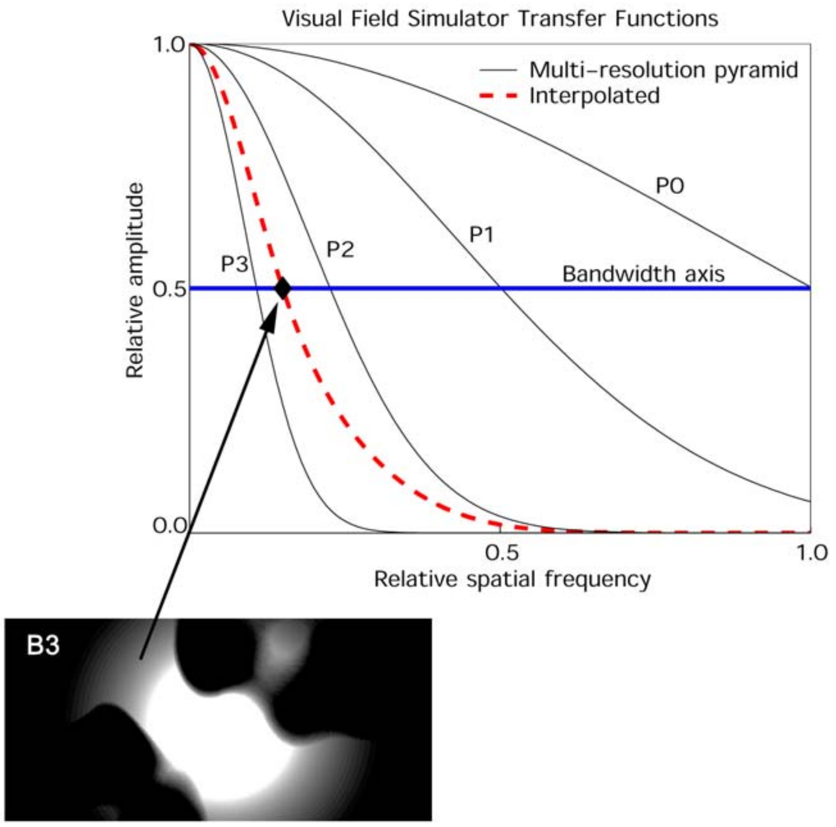

Figure 6. Effective transfer function (dashed curve) obtained by blending two levels of a Gaussian pyramid. The diamond indicates a desired resolution (half height bandwidth) that lies between the resolutions (bandwidths) associated with pyramid levels 2 and 3. The tail of the arrow indicates the blending-function value that produces the desired resolution (see equations 1 and 2).

2.4. Image processing sequence

The image processing components described above are combined in the following fashion. Prior to processing the video image sequence, all the blending functions are computed from the resolution map. Then, the follow steps are applied in real time to each input image frame:

First, the multi-resolution pyramid is computed (e.g., see Figure 2).

Second, the gaze direction is obtained, and the blending functions are aligned with the current gaze direction. For example, Figure 7B shows a blending function aligned with a particular gaze direction (the cross), where the white rectangle represents the display screen.

Third, the lowest resolution image is up-sampled and interpolated so that it has the same dimensions as the next highest resolution image in the pyramid. For example, pyramid level P3 in Figure 2 would be up-sampled and interpolated to the same size as pyramid level P2.

Fourth, image pixels from this next highest resolution image are then blended with the pixels in the up-sampled image using the appropriate blending function. For example, the blending function B3 would be used for blending the up-sampled pyramid level P3 with pyramid level P2. The blending only occurs in the area of the blending function that contains values strictly between 0.0 and 1.0. The pixels where the blending function is 0.0 do need not be blended because they are pixels that come from the lower resolution portions of the image; the resolutions in these portions have already been created. The pixels where the blending function is 1.0 represent pixels that belong to higher resolution portions of the image, and have yet to be processed. For example, Figure 7A shows the part of the image that is blended using the blending function in 7B. The resulting blended image is then up-sampled and interpolated, and then blended with the next higher resolution image in the pyramid, and so on, until all levels of the pyramid have been processed. For example, the result of blending pyramid levels P3 and P2 would be up-sampled and blended with pyramid level P1 using the blending function B2, then the result is up-sampled and blended with pyramid level P0 to obtain the final image. (Note that up-sampling and interpolation is applied to all pixels at each level of the pyramid, regardless of whether any blending is occurring.)

Fifth, the final image is displayed. For grayscale images, the software simply copies the image to the graphics raster buffer. For color images, the image processing is done in standard YUV color space[13], and is converted to RGB color space before being displayed. Figure 8B shows a final output for a pyramid with six levels, using the resolution map in Figure 5.

There are several other important details of the image processing. One is that during the final blending operation, any pixels where the blending function value is 1.0 must be dealt with separately—they are filled in by copying from the input image, which is equivalent to blending with a weight of 1.0.

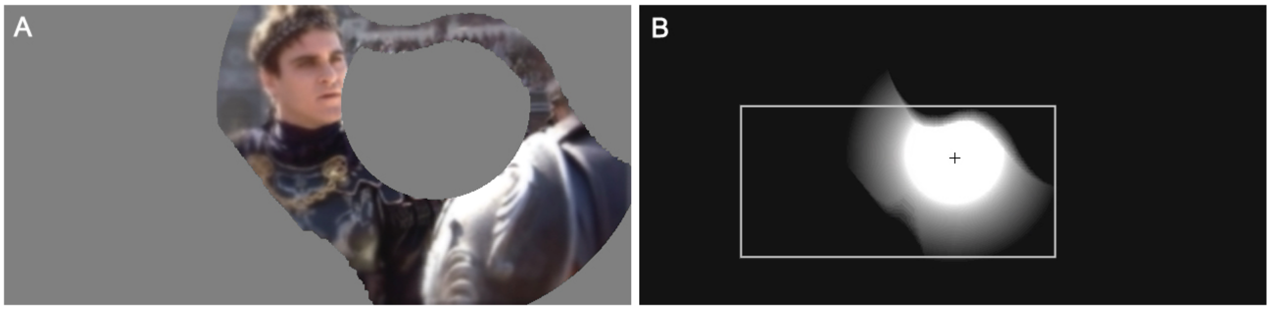

Figure 7. A. For each level of the pyramid, only the regions of the blending function with non-zero and non-1.0 values are blended (gray values represent parts of the image that require no blending). B. The blending function is shifted so that the center of the function aligns with the direction of gaze (cross).

Another important detail concerns the use of the blending functions. As illustrated in Figure 5, all of the blending functions are computed at full display size so they can all be aligned precisely with the direction of gaze. However, the blending functions are applied to the images in a multi-resolution pyramid (Figure 2), which is much more efficient than applying the blending functions to full size images at different resolutions (Figure 3).

A final detail concerns how the software interprets a value of 0.0 in the resolution map. We have implemented two different options. For one option, 0.0 is interpreted to be the lowest resolution of the multi-resolution pyramid. For the other option, 0.0 is interpreted as the average pixel value in the current video image (or the running average over some user-specified number of video images).

3. RESULTS

We have tested the software on a variety of image sequences using several different resolution maps. One example is shown in Figure 8, and two others are shown in Figure 9. Figure 9A shows the resolution map corresponding to a normal visual field and Figure 9B shows the resolution map corresponding to an abnormal visual field obtained from an individual with macular degeneration (note that a small island of functional retina exists within the large area of central vision loss). Figures 9C and 9D show the images created from the resolution maps in 9A and 9B, respectively. These examples illustrate that the algorithm produces quality images with precisely varying resolution. In general, we find that the outputs from video sequences contain no detectable artifacts such as blocking or ringing. Also, because the resolution varies continuously across the image, the transitions between resolution levels that were sometimes visible in our previous algorithm are no longer visible. Eliminating detectable artifacts is important in gaze-contingent experiments because of the high sensitivity of the human visual system to spatial and temporal discontinuities. The smooth transition between resolutions in the present algorithm is best appreciated for images like Figure 9C and 10B that are created with smooth resolution maps. We have found (under gaze contingent viewing conditions) that an original input video sequence is indistinguishable from a processed image sequence if the resolution map declines sufficiently slowly away from the direction of gaze.

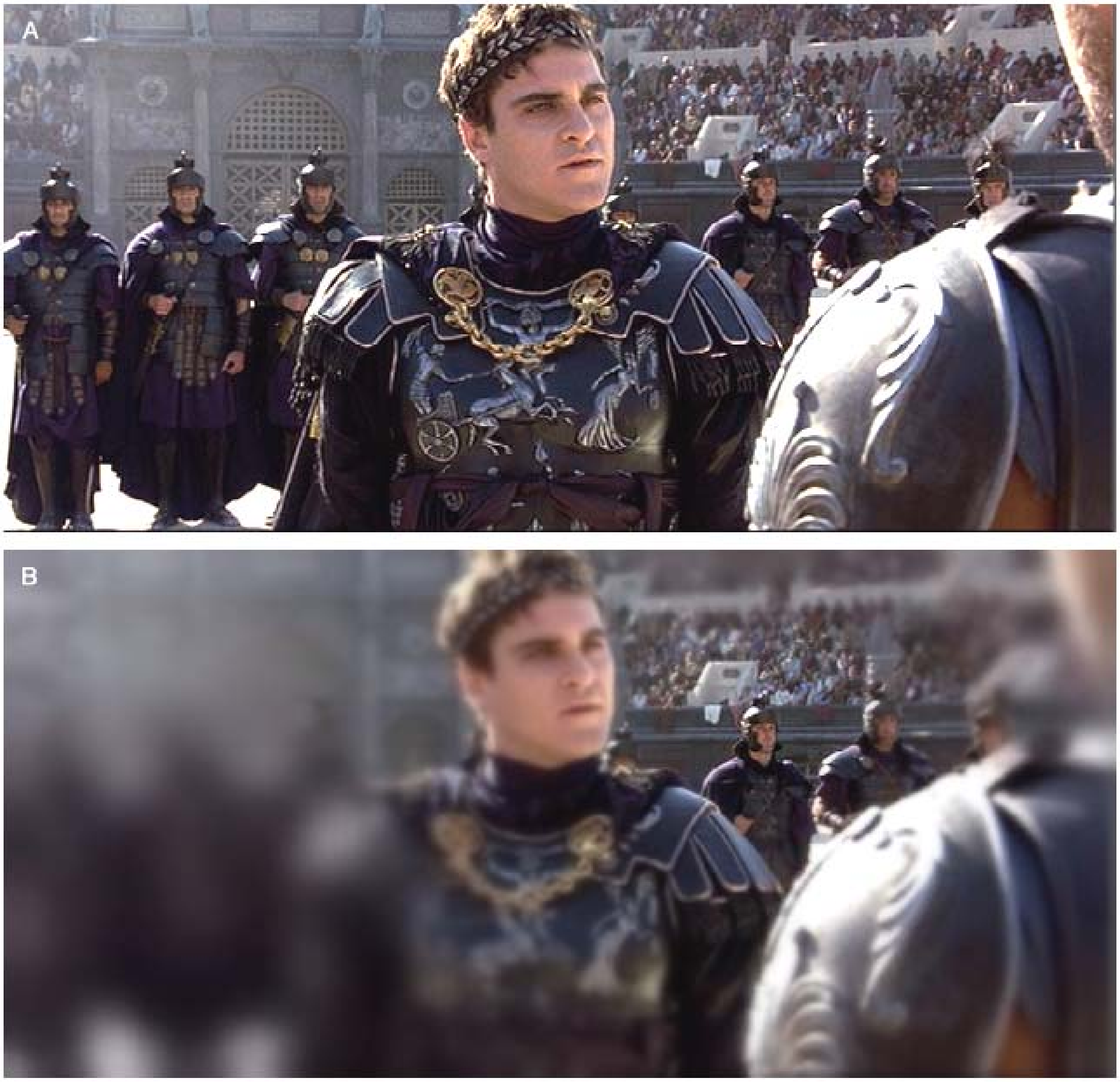

Figure 8. A. Original input image. B. Image encoded using a six level pyramid and the glaucoma resolution map from Figure 5.

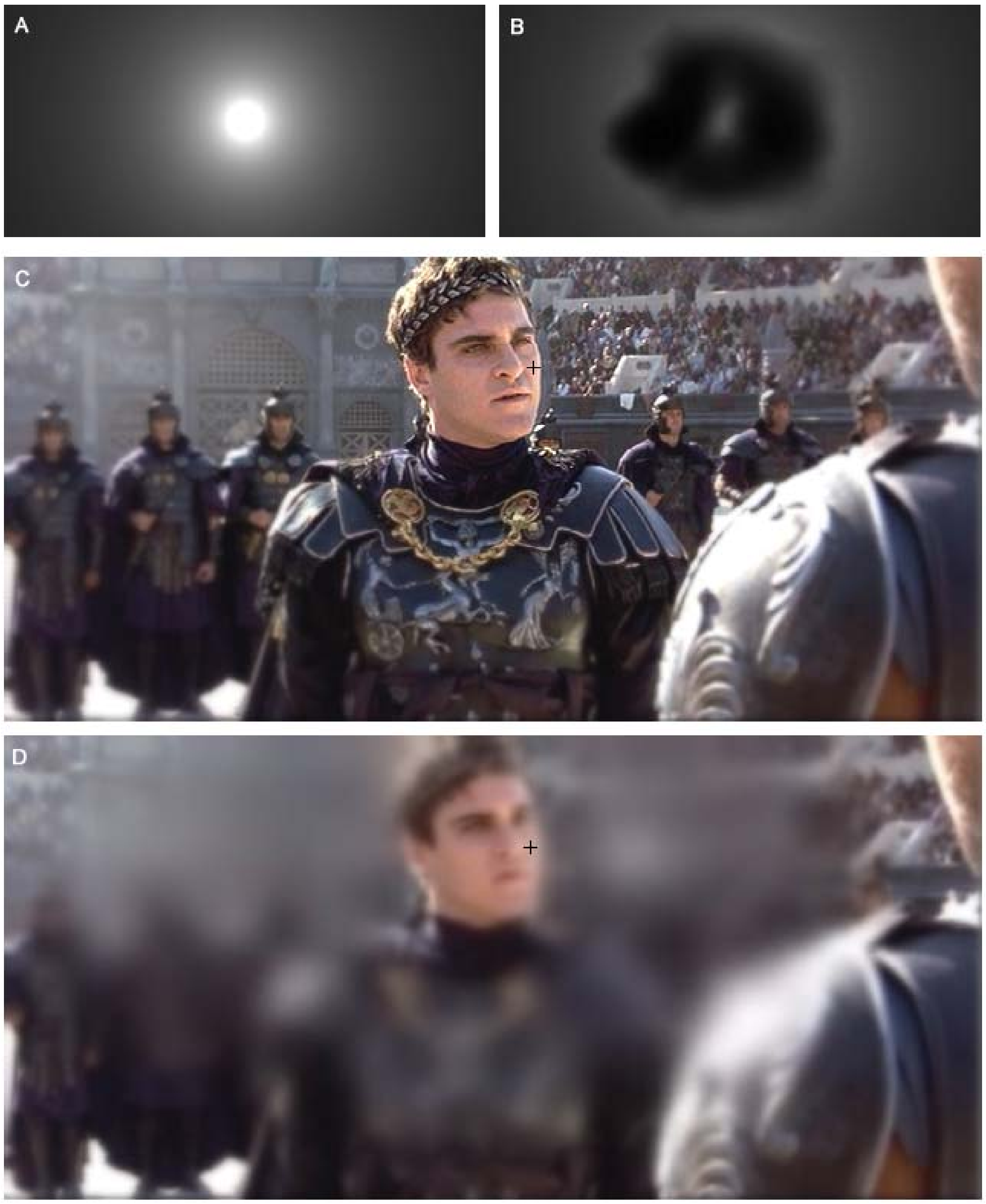

Figure 9. A. A normal resolution map with the "blind spot" omitted. B. A resolution map for a patient with macular degeneration. C. The output image for the resolution map in A. D. The output image for the resolution map in B.

Table 1 shows performance measurements for the software running on two separate personal computers with no special hardware. As the table indicates, the software runs at rates suitable for use in real-time gaze contingent experiments and demonstrations.

Intel® Processor	Grayscale	Color
800 MHz Pentium® III	60	20
1500 MHz Pentium® 4	56	28

Table 1. Performance on 640 x 304 video in frames per second.

Note that many factors affect the software's performance, including the software compiler's target CPU setting (Table 1 is based upon settings targeted for a Pentium III), the speed of the graphics controller, and the refresh rate of the video monitor. Also, note that many modern graphics controllers contain YUV-RGB conversion hardware, so it will be possible in future versions of the software to pass the output YUV image directly to the controller, instead of converting the image in software. We have found that eliminating software conversion can increase frame rates for color video by up to a factor of 2.

As mentioned in the Introduction, our previous algorithm was designed primarily for use in video compression and video telecommunication, while the present algorithm was designed primarily for simulation. In the previous algorithm, the blending function was an arbitrary linear ramp that was applied over smaller regions of spatial overlap between pyramid levels. The smaller regions of spatial overlap result in greater image compression. The present algorithm can be made more useful for compression by thresholding the blending functions to create less overlap between pyramid levels. This leads to superior image quality over the previous algorithm (for the same compression level) because of the better blending function. In addition, the present algorithm uses a superior method of up-sampling pyramid images. Figure 10 shows output images of the two algorithms for the same level of compression.

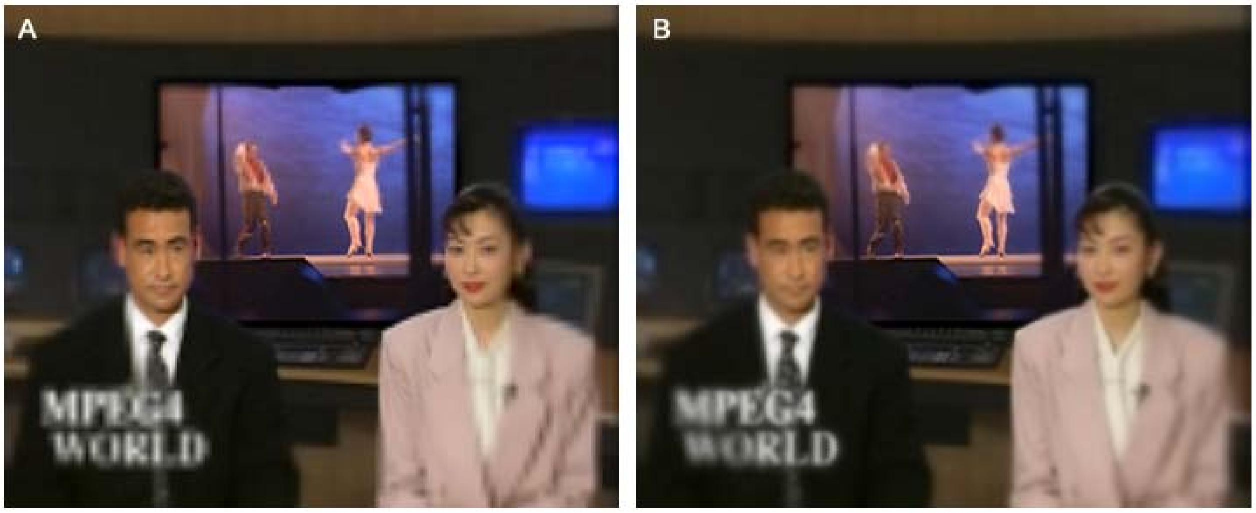

Figure 10. A. The previous algorithm produces artifacts that are especially visible in the text and on the woman's jacket; it also produces mildly visible boundaries between pyramid levels when the gaze direction is shifted. B. By thresholding blending function values below 0.375, the current algorithm creates a higher quality image with an equal number of bits.

4. DISCUSSION

In this paper we have described a method for generating gaze contingent variable resolution displays in real time on inexpensive general purpose computers. The current software allows the user to specify an arbitrary spatial resolution (a half-height filter bandwidth) for each pixel location with respect to the direction of gaze. This resolution map is then applied to each frame of an input video sequence, contingent on the current direction of gaze.

The ability to do this in a precise fashion opens up a number of possible research applications. For example, we are currently using the software to systematically vary the spatial information across the visual field in search tasks. An issue of particular interest is how steeply resolution can decline away from the gaze direction without having a significant affect on search accuracy and speed. Our aim is to understand how the spatial-frequency properties of the target and background affect visual search performance under those complex naturalistic conditions where the observer makes multiple fixations. Similar studies of other complex multi-fixation tasks such as visual navigation and reading are also possible with the current software.

The structure of the present algorithm and software will accommodate other forms of gaze contingent displaying. Specifically, instead of a spatial resolution map, one can use a spatial map that specifies (for each pixel location with respect to the direction of gaze) values along some other dimension. For example, we have analyzed the computational requirements of creating gaze contingent displays for the dimensions of local contrast, local mean luminance, color, noise level, and spatial distortion. For each of these dimensions, the computational requirements are very similar to those for spatial resolution, and thus there should be no problem attaining similar real-time performance.

The present algorithm and software also open up a number of possible practical applications. For example, we are currently using the software to create a collection of simulations of visual field defects produced by various forms of eye disease. This collection could be used to educate students, physicians, and patients' family members about the perceptual and performance consequences of specific forms of vision loss in tasks such as navigation, reading or watching television. We have found that such simulations are compelling and informative even when the observer controls the direction of gaze with a mouse, rather than with an eye-tracking device. The software could also be used to create simulations of human visual fields during the course of development, as well as simulations of the visual fields of other organisms.

Another practical application of the algorithm is in low-bandwidth video communications. These applications are discussed at length elsewhere[10, 11]. However, it is worth mentioning here that both the present and previous versions of the algorithm can serve as pre/post-processors to other video coding/decoding software or hardware, and hence both versions are compatible with existing standards.

ACKNOWLEDGMENTS

We thank Eli Peli for supplying us with Goldmann perimeter measurements and for encouraging us to consider simulations of the visual fields of low vision patients. Supported by NIH grant EY02688.

REFERENCES

1. Rayner, K., *Eye movements in reading and information processing: 20 years of research.* Psychoological Bulletin, 1998. **124**(3): p. 372-421.
2. McConkie, G.W., *Eye movement contingent display control: personal reflections and comments.* Scientific Studies of Reading, 1997. **4**(1): p. 303-316.
3. Kortum, P.T. and W.S. Geisler, *Search performance in natural scenes: the role of peripheral vision.* Investigative Ophthalmology & Visual Science Supplement, 1996. **37/3**(S297).
4. Weiman, C.F.R., *Video Compression Via Log Polar Mapping.* SPIE Proceedings : Real Time Image Processing II, 1990. **1295**: p. 266-277.
5. Juday, R.D. and T.E. Fisher, *Geometric transformations for video compression and human teleoperator display.* SPIE Proceedings: Optical Pattern Recognition, 1989. **1053**: p. 116-123.

6. Warner, H.D., G.L. Serfoss, and D.C. Hubbard, *Effects of Area-of-Interest Display Characteristics on Visual Search Performance and Head Movements in Simulated Low-Level Flight*. 1993, Armstrong Laboratory, Human Resources Directorate, Aircrew Training Division, Williams AFB, AZ.
7. Silsbee, P.L., A.C. Bovik, and D. Chen, *Visual pattern image sequence coding.* IEEE Transactions on Circuits and Systems for Video Technology, 1993. **3**(4): p. 291-301.
8. Barnett, B.S. and A.C. Bovik, *Motion compensated visual pattern image sequence coding for full motion multisession videoconferencing on multimedia workstation.* Journal of Electronic Imaging, 1996. **5**: p. 129-143.
9. Kortum, P.T. and W.S. Geisler, *Implementation of a foveated image-coding system for bandwidth reduction of video images.* SPIE Proceedings: Human Vision and Electronic Imaging, 1996. **2657**: p. 350-360.
10. Geisler, W.S. and J.S. Perry, *A real-time foveated multi-resolution system for low-bandwidth video communication.* Human Vision and Electronic Imaging, SPIE Proceedings, 1998. **3299**: p. 294-305.
11. Geisler, W.S. and J.S. Perry, *Variable-resolution displays for visual communication and simulation.* The Society for Information Display, 1999. **30**: p. 420-423.
12. Burt, P.J. and E.H. Adelson, *The Laplacian pyramid as a compact image code.* IEEE Transactions on Communications, 1983. **COM-31**(4): p. 532-540.
13. *Recommendations of the CCIR: Encoding parameters of digital television for studios.* International Telecommunication Union, 1990. **XI**(1): p. 95-104.