Gaze-Contingent Temporal Filtering of Video

Martin Böhme* Michael Dorr† Thomas Martinetz‡ Erhardt Barth§

Institute for Neuro- and Bioinformatics
University of Lübeck, Germany

Abstract

We describe an algorithm for manipulating the temporal resolution of a video in real time, contingent upon the viewer's direction of gaze. The purpose of this work is to study the effect that a controlled manipulation of the temporal frequency content in real-world scenes has on eye movements. We build on the work of Perry and Geisler [1998; 2002], who manipulate spatial resolution as a function of gaze direction, allowing them to mimic the resolution distribution of the human retina or to simulate the effect of various diseases (e.g. glaucoma).

Our temporal filtering algorithm is similar to that of Perry and Geisler in that we interpolate between the levels of a multiresolution pyramid. However, in our case, the pyramid is built along the temporal dimension, and this requires careful management of the buffering of video frames and of the order in which the filtering operations are performed. On a standard personal computer, the algorithm achieves real-time performance (30 frames per second) on high-resolution videos (960 by 540 pixels).

We present experimental results showing that the manipulation performed by the algorithm reduces the number of high-amplitude saccades and can remain unnoticed by the observer.

CR Categories: I.4.9 [Image Processing and Computer Vision]: Applications; I.4.3 [Image Processing and Computer Vision]: Enhancement—Filtering; J.4 [Social and Behavioral Sciences]: Psychology

Keywords: gaze-contingent display, foveation, temporal multiresolution pyramid

Figure 1: Top: Image from one of the video sequences. Bottom: Same image with gaze-contingent temporal filtering applied. The white square at the centre left (below the white sail) indicates the point of regard.

1 Introduction

Gaze-contingent displays manipulate some property of the (static or moving) image as a function of gaze direction (see Duchowski, Cournia and Murphy [2004] for a review). This type of display was first used in reading research [McConkie and Rayner 1975; Rayner 1975; Rayner 1998] and has since been used in many psychophysical and perceptual studies (e.g. [Loschky and McConkie 2000; Loschky et al. 2005; Cornelissen et al. 2005]). The image property that is most commonly manipulated in a gaze-contingent display is spatial resolution. A popular type of manipulation is to "foveate" an image or video, i.e. to simulate the effect of the variable resolution of the human retina, which is highest at the fovea and falls off towards the periphery. If the foveation is adjusted to match the resolution distribution of the retina, the effect is not noticeable for the observer, but the resulting images can be compressed more efficiently because they contain less high-frequency content [Kortum and Geisler 1996; Geisler and Perry 1998]. Another application is to visualize the effect of diseases of the eye, e.g. glaucoma [Perry and Geisler 2002]; these visualizations can be used to educate students or family members of patients about the effects of such diseases.

The current state-of-the-art algorithm for gaze-contingent spatial filtering of video is due to Perry and Geisler [2002]. Unlike previous algorithms, which introduced artifacts in the filtered images, their algorithm produces smooth, artifact-free results.

In this paper, we present a gaze-contingent display that manipulates not the spatial, but the temporal resolution of a video. The basic effect of temporal filtering is to blur the moving parts of an image while leaving the static parts unchanged (see Figure 1 for an example). Our motivation for performing this type of manipulation is that we want to examine the effect that it has on eye movements; movement or change in the periphery of the visual field is a strong cue for eye movements, and indeed we have been able to show that

*e-mail: boehme@inb.uni-luebeck.de
†e-mail: dorr@inb.uni-luebeck.de
‡e-mail: martinetz@inb.uni-luebeck.de
§e-mail: barth@inb.uni-luebeck.de

http://doi.acm.org/10.1145/1117309.1117353

gaze-contingent temporal filtering reduces the number of saccades with large amplitude [Dorr et al. 2005b] but can remain unnoticed by the observer [Dorr et al. 2005a]. We will present a summary of these results, but our main focus in this paper is on the temporal filtering algorithm, which has not previously been described. Our long-term goal is to find ways of guiding an observer's eye movements [Itap 2002]. This requires two components: We must prevent saccades to undesired locations (using techniques such as temporal filtering) and encourage saccades to desired locations (using suitable stimuli).

Our algorithm is similar to that of Perry and Geisler in that it is based on interpolating between the levels of a multiresolution pyramid. However, in our case, the pyramid is built along the temporal dimension. Because it is not feasible to hold the entire multiresolution representation of the video in memory, we shift a temporal window along the pyramid as the video plays and compute the contents of this window in real time. On a standard personal computer, our algorithm achieves real-time performance (30 frames per second) on high-resolution videos (960 by 540 pixels). Note that the computational requirements for temporal filtering are greater than for spatial filtering because the pyramid reduces resolution (and thus bandwidth) only along one dimension and because the interpolation between levels of the pyramid cannot be interleaved with the upsampling steps.

A limitation of temporal filtering compared to spatial filtering is that the temporal pyramid requires a certain amount of "lookahead" in the video stream because the filters used in the pyramid are non-causal. This means that the algorithm is only useful for pre-recorded video material; on a live video stream from a camera, the algorithm would introduce a latency of several seconds, making it impractical for applications such as head-mounted displays with video-see-through. Note, however, that this latency applies only to the video stream; the latency of the gaze-contingent effect, i.e. the time elapsed between an eye movement and the display update, depends only on the time required to process and display one frame.

2 Algorithm

The gaze-contingent temporal filtering algorithm takes a resolution map $R(x,y)$ that specifies the desired temporal resolution at each point in the visual field, relative to the point of regard. For each frame in the video, the resolution map is centred on the current gaze position (obtained from the eye tracker), and the amount of filtering specified by the map is applied at each pixel. This is achieved by interpolating between the levels of a temporal multiresolution pyramid.

For a spatial multiresolution pyramid, the complete data for all pyramid levels can be kept in memory at the same time. This means that the whole of pyramid level l can be computed before pyramid level $l+1$ (which depends on level l). If we wanted to take the same approach for the temporal multiresolution pyramid, then, because one level of the pyramid spans the entire image sequence, we would have to store this image sequence along with all of its multiresolution versions in main memory—which is clearly not feasible for all but the shortest of videos. Alternatively, we could precompute all of the pyramid levels and read them from disk, one video stream per pyramid level. However, we found that decoding these video streams actually takes more CPU time than computing the multiresolution pyramid on the fly. (Storing the video in uncompressed form is not an option because this would require excessive disk bandwidth.) For this reason, we compute the multiresolution pyramid as the video is being displayed, keeping only those frames of each pyramid level in memory that are needed at any given time.

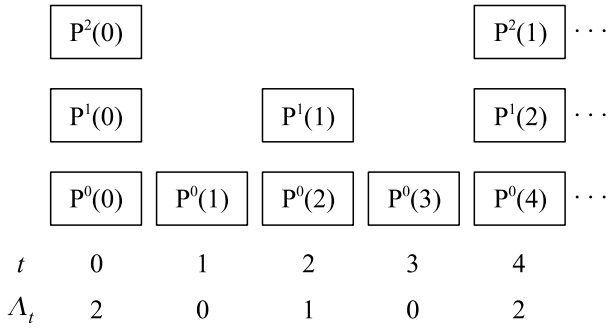

Figure 2: A temporal multiresolution pyramid with three levels. The temporal resolution (frames per second) is halved from level to level. Below the pyramid, t is the time step corresponding to each column, and Λ_t is the index of the highest pyramid level that contains an image for time step t.

2.1 Notational Conventions

The input video is given as a sequence of images $\mathrm{I}(0), \mathrm{I}(1), \ldots$ (the sequence is assumed to be infinite). The images contain a single channel (colour videos can be processed by filtering the colour channels individually); they have a width of w pixels and a height of h pixels. The pixel at position (x,y) of image $\mathrm{I}(t)$ is referred to as $\mathrm{I}(t)(x,y)$ ($x,y \in \mathbf{N}$, $0 \le x < w$, $0 \le y < h$). (The convention that is chosen for the direction of the image axes is irrelevant for our purposes.) Operations that refer to entire images, such as addition of images or multiplication by a constant factor, are to be applied pixelwise to all pixels in the image.

The individual levels of the multiresolution pyramid are referred to as P^0 to P^L (i.e. the pyramid contains $L+1$ levels). $\mathrm{P}^l(n)$ refers to the n-th image at level l. P^0 is identical to the input image sequence, and P^{l+1} is obtained by low-pass filtering P^l temporally and then subsampling temporally by a factor of 2, i.e. dropping every other frame. (See Burt and Adelson [1983] for a detailed description of multiresolution pyramids, albeit with spatial filtering.) Hence, $\mathrm{P}^{l+1}(n)$ corresponds to the same point in time as $\mathrm{P}^l(2n)$, and for an input image $\mathrm{I}(t)$ at time t, the corresponding image in pyramid level P^l is $\mathrm{P}^l(\frac{t}{2^l})$ (see Figure 2). Of course, such an image only exists if t is a multiple of 2^l. By Λ_t, we designate the index of the highest pyramid level that contains an image corresponding to $\mathrm{I}(t)$; Λ_t is thus the maximum integer value that fulfils $0 \le \Lambda_t \le L$ and $t \bmod 2^{\Lambda_t} = 0$.

Because each frame of the output video is generated by interpolating between all of the levels of the multiresolution pyramid, we need to upsample each level to the full video frame rate. We refer to these upsampled pyramid levels as Q^0 to Q^L and will discuss how to compute them in the next section.

2.2 Relationships Between Pyramid Levels

Figure 3 shows the relationships between different levels of the multiresolution pyramid and the way in which the upsampled versions $\mathrm{Q}^0, \ldots, \mathrm{Q}^L$ of the pyramid levels $\mathrm{P}^0, \ldots, \mathrm{P}^L$ are obtained.

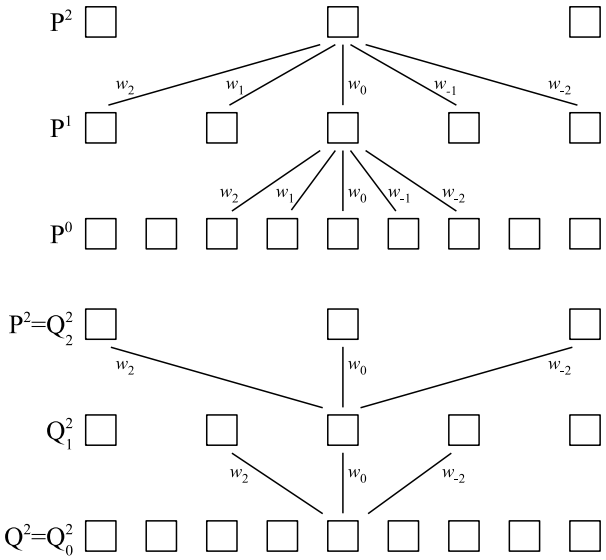

Figure 3: Top: Relationships between different levels of the multiresolution pyramid. Each level of the pyramid is computed from the level below by filtering with the kernel $w_{-c},\ldots,w_c$ (in this case, $c=2$) and subsampling by a factor of 2. Bottom: Scheme for computing upsampled versions Q^l of pyramid levels P^l. Q^l is computed from P^l by repeatedly inserting zeros and filtering with the kernel $w_{-c},\ldots,w_c$. In practice, only the non-zero frames are included in the convolution with the kernel.

$\mathrm{P}^{l+1}(t)$ is obtained by low-pass filtering from images in P^l:

$$\mathrm{P}^{l+1}(n) = \sum_{i=-c}^{c} w_i \cdot \mathrm{P}^l(2n-i) \Bigg/ \sum_{i=-c}^{c} w_i$$

The $w_{-c},\ldots,w_c$ are the kernel coefficients. We use a binomial filter with $c=2$ and $w_0=6$, $w_1=w_{-1}=4$ and $w_2=w_{-2}=1$.

Q^l is obtained from P^l by performing l upsampling steps. The intermediate results of these operations are denoted by Q^l_l to Q^l_0, where $\mathrm{Q}^l_l = \mathrm{P}^l$ and $\mathrm{Q}^l_0 = \mathrm{Q}^l$. Conceptually, Q^l_k is obtained from Q^l_{k+1} by inserting zeros to upsample by a factor of 2 and then performing a low-pass filtering operation. In practice, these two steps are combined into one, as expressed in the following formula:

$$\mathrm{Q}^l_k(n) = \sum_{\substack{i=-c \\ (n-i) \bmod 2 = 0}}^{c} w_i \cdot \mathrm{Q}^l_{k+1}\left(\frac{n-i}{2}\right) \Bigg/ \sum_{\substack{i=-c \\ (n-i) \bmod 2 = 0}}^{c} w_i$$

2.3 Sliding Window Boundaries

As described above, we wish to compute the P^l and Q^l_k on the fly, while the video is being displayed. To minimize memory requirements, we will keep only those frames of the P^l and Q^l_k in memory that are needed at any given time. This is achieved by using a circular buffer for each of the P^l and Q^l_k to implement a sliding window that contains the required frames. We will proceed in this section to derive the appropriate front and rear window boundaries, relative to the current frame. For example, a front and rear boundary for P^l of 4 and -2, respectively, would mean that, at time t, the sliding window would contain frames $\mathrm{P}^l(\frac{t}{2^l}-2)$ to $\mathrm{P}^l(\frac{t}{2^l}+4)$.

Because the filter used in the multiresolution pyramid is non-causal, the front boundaries have to extend into the future by different amounts (we also refer to this distance as the *lookahead*). For this reason, our algorithm is only suitable for pre-recorded video sequences, as noted in the introduction.

To derive the boundaries of the sliding windows, we start by noting that we need $\mathrm{Q}^0(t)$ to $\mathrm{Q}^L(t)$ in each time step to compute the final blended output image. We will work backwards from this to find the window sizes required in the preceding processing steps.

From the equation for Q^l (in Section 2.2), we see that to produce $\mathrm{Q}^l(t) = \mathrm{Q}^l_0(t)$, we need images $\mathrm{Q}^l_1(i)$ with $\frac{t-c}{2} \le i \le \frac{t+c}{2}$ and $i \in \mathbf{N}$. The front and rear boundaries for Q^l_1 are thus $\frac{c}{2}$ and $-\frac{c}{2}$, respectively, relative to the current image $\mathrm{Q}^l_1(\frac{t}{2})$. Because Q^l_1 thus needs a lookahead of $\frac{c}{2}$ images, we need to produce $\mathrm{Q}^l_1(\frac{t}{2}+\frac{c}{2})$ at time t (assuming that both t and c are even). We could now repeat this argument to compute the window boundaries required for all Q^l_k. The argument is simplified, however, by noting that we also obtain a valid result if we set a front boundary of c and a rear boundary of 0 for all k. These boundaries are sufficient for the following reason: At time t, we need to produce $\mathrm{Q}^l_k(\frac{t}{2^k}+c)$ (assuming that t is a multiple of 2^k). This requires the images $\mathrm{Q}^l_{k+1}(i)$ with $(\frac{t}{2^k}+c-c)/2 = \frac{t}{2^{k+1}} \le i \le (\frac{t}{2^k}+c+c)/2 = \frac{t}{2^{k+1}}+c$, i.e. a front boundary of c and a rear boundary of 0 is also sufficient for Q^l_{k+1} and by induction for all k.

Turning now to the downsampling side of the pyramid, we note that for all l, the sliding window for P^l needs to contain at least the image $\mathrm{P}^l(\frac{t}{2^l}+c)$, because this image is needed for Q^l, which has a lookahead of c. Additional requirements are imposed by the dependencies between the P^l. We start with P^L, which no other pyramid level depends on, and assign it a front and rear boundary of c. Hence, at time t, we need to compute $\mathrm{P}^L(\frac{t}{2^L}+c)$, for which we need the images $\mathrm{P}^{L-1}(\frac{t}{2^{L-1}}+2c-c),\ldots,\mathrm{P}^{L-1}(\frac{t}{2^{L-1}}+2c+c)$. This means that pyramid level P^{L-1} needs to have a rear boundary of c and a front boundary of $3c$. By continuing in this fashion, we find that level P^l requires a front boundary of $(2^{L-l+1}-1)\cdot c$. The rear boundary for all P^l is c because of the requirements imposed by the Q^l. We will refer to the front boundary or lookahead of P^l as $\lambda_l := (2^{L-l+1}-1)\cdot c$. Because the input images are fed into P^0, the total latency of the pyramid is λ_0.

2.4 Pyramid Update Algorithm

We are now ready to present the algorithm that updates the multiresolution pyramid in each time step:

Algorithm 2.1 (Temporal pyramid update step)

Input:	t	Time step to update the pyramid for
Globals:	$\mathrm{P}^0,\ldots,\mathrm{P}^L$	Pyramid levels
	$\mathrm{Q}^0,\ldots,\mathrm{Q}^L$	Upsampled pyramid levels
	Q^l_k	Intermediate results for upsampled pyramid levels $(0 \le l \le L, 0 \le k \le l)$

Downsampling phase

$\mathrm{P}^0(t+\lambda_0) = \mathrm{I}(t+\lambda_0)$

$\Lambda_t = \max(\{\Lambda \in \mathbf{N} \mid 0 \le \Lambda \le L \wedge t \bmod 2^\Lambda = 0\})$

for $l = 1,\ldots,\Lambda_t$ **do**

$$\mathrm{P}^l(\tfrac{t}{2^l}+\lambda_l) = \sum_{i=-c}^{c} w_i \cdot \mathrm{P}^{l-1}(\tfrac{t}{2^{l-1}}+2\lambda_l-i) \Bigg/ \sum_{i=-c}^{c} w_i$$

end for

Upsampling phase
for $l = 0,\ldots,L$ **do**
  if $\Lambda_t \geq l$ **then**
    $\mathrm{Q}_l^l(\frac{t}{2^l}+c) = \mathrm{P}^l(\frac{t}{2^l}+c)$
    $\hat{\Lambda} = l-1$
  else
    $\hat{\Lambda} = \Lambda_t$
  end if
  for $k = \hat{\Lambda}, \hat{\Lambda}-1,\ldots,0$ **do**

$$\mathrm{Q}_k^l(\tfrac{t}{2^k}+c) = \sum_{\substack{i=-c \\ (\frac{t}{2^k}+c-i) \bmod 2=0}}^{c} w_i \cdot \mathrm{Q}_{k+1}^l\left((\tfrac{t}{2^k}+c-i)/2\right) \Bigg/ \sum_{\substack{i=-c \\ (\frac{t}{2^k}+c-i) \bmod 2=0}}^{c} w_i$$

  end for
end for

This pyramid update is carried out in each time step before blending the pyramid levels together to obtain the final output image (see next section). Because of the latency in the pyramid, a certain number of images at the beginning of P^l and Q^l are never computed; specifically, these are the $\mathrm{P}^l(n)$ with $0 \leq n < \lambda_l$ and $\mathrm{Q}^l(n)$ with $0 \leq n < c$. These images are assumed to be initialized to some suitable value (e.g. all black or equal to the first image in the video).

2.5 Gaze-Contingent Temporal Filtering Algorithm

The desired temporal resolution at each pixel is specified by a resolution map $R(x,y)$, where (x,y) is measured relative to the point of regard, $-(w-1) \leq x \leq w-1$, $-(h-1) \leq y \leq h-1$, and $0 \leq R(x,y) \leq 1$. The values contained in the map specify the temporal resolution relative to the resolution of the original video, i.e. a value of 1 corresponds to pyramid level P^0, 0.5 corresponds to pyramid level P^1, and in general, 2^{-l} corresponds to pyramid level P^l. Intermediate values are handled by interpolating between the two pyramid levels whose resolutions bracket the desired resolution, as described below. Values less than 2^{-L} are treated as referring to pyramid level P^L, since no lower resolution versions of the image sequence are available.

Note that interpolating between two pyramid levels delivers only an approximation to the desired intermediate resolution. There are methods that deliver more accurate results (e.g. [Köthe 2004]), but they are computationally more expensive, and we believe that the approximation used here is sufficient for our purposes.

We compute blending functions that are used to blend between adjacent levels of the pyramid from the effective transfer functions for the pyramid levels (see Perry and Geisler [2002] for details). The transfer function for pyramid level P^l can be approximated by the Gaussian

$$T_l(r) = \mathrm{e}^{-r^2/(2\sigma_l^2)},$$

where $\sigma_l^2 = 1/(2^{2l+1}\ln 2)$, and r is the relative resolution. These transfer functions can now be used to define the blending functions. The blending function that is used to blend between levels P^l and P^{l+1} is given by

$$B_l(x,y) = \begin{cases} \frac{\frac{1}{2}-T_{l+1}(R(x,y))}{T_l(R(x,y))-T_{l+1}(R(x,y))} & 2^{-(l+1)} < R(x,y) < 2^{-l} \\ 0 & R(x,y) \leq 2^{-(l+1)} \\ 1 & R(x,y) \geq 2^{-l} \end{cases}$$

The algorithm for computing the output image $\mathrm{O}(t)$ at time t for a gaze position $(g_x(t), g_y(t))$ is now as follows:

Algorithm 2.2 (Pyramid level blending)

Input:	t	Current time step
	$(g_x(t), g_y(t))$	Gaze position
	$\mathrm{Q}^0(t),\ldots,\mathrm{Q}^L(t)$	Upsampled pyramid levels
Output:	$\mathrm{O}(t)$	Output image

  $\mathrm{O}(t) = \mathrm{Q}^L(t)$
  for $l = L-1, L-2,\ldots,0$ **do**
    for $x = 0,\ldots,w$, $y = 0,\ldots,h$ **do**
      $b = B_l(x-g_x(t), y-g_y(t))$
      $\mathrm{O}(t)(x,y) = (1-b)\,\mathrm{O}(t)(x,y) + b\,\mathrm{Q}^l(t)(x,y)$
    end for
  end for

To process an entire video, we now execute Algorithm 2.1 and Algorithm 2.2 for each video frame, as follows:

Algorithm 2.3 (Gaze-contingent temporal filtering)
  for $t = 0, 1, \ldots$ **do**
    Update pyramid for time step t (Algorithm 2.1)
    Get current gaze position $(g_x(t), g_y(t))$
    Compute output image $\mathrm{O}(t)$ (Algorithm 2.2)
    Display image $\mathrm{O}(t)$
  end for

Note that the gaze position is not needed for the pyramid update (Algorithm 2.1), so its measurement can be deferred until directly before the blending step (Algorithm 2.2).

2.6 Miscellaneous Considerations

As remarked above, colour images can be filtered simply by filtering the colour planes separately. In our implementation, we operate directly on the three colour planes of the YUV420 images read from the digital video files (MPEG-2 format). This has two advantages: First, we avoid having to transform the images to the RGB colour space before processing them; the processed images in YUV420 format can be output directly to the graphics card and are converted to RGB in hardware. Second, in the YUV420 image format, the chroma channels U and V are subsampled by a factor of two in horizontal and vertical direction. This reduces the number of pixels that have to be processed in these channels by a factor of four and halves the total number of pixels compared to the RGB format.

Another noteworthy point is that the processing time required by Algorithm 2.1 depends on Λ_t, the index of the highest pyramid level that is updated in time step t, and thus varies with each time step. Furthermore, the difference between the average and maximum processing time required is quite large. We omit the exact analysis here, but the average processing time is on the order of $\mathrm{O}(2L)$, while the maximum time is on the order of $\mathrm{O}(\frac{L^2}{2})$. Hence, if we choose the video resolution such that the maximum processing time will fit within the inter-frame interval, then, on most other frames, the CPU will be idle for a substantial proportion of the time. This can be avoided by introducing a variable-length buffer of several frames length to buffer the output of the multiresolution pyramid and compensate for the variation in processing time between frames.

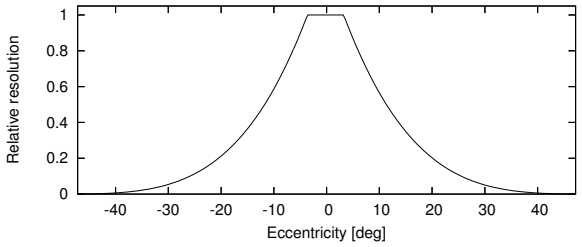

Figure 4: A slice through the radially symmetric resolution map used to generate the filtered image in Figure 1 and in the experiments on the effect of gaze-contingent temporal filtering on eye movements. The horizontal axis (radial eccentricity from the point of regard) has been scaled to give the visual angle in degrees, taking into account the monitor size and viewing distance used in the experiments.

3 Implementation and Results

We implemented the algorithms in C++, with performance-critical parts written in Intel x86 assembly language using the SSE2 vector instructions. Experiments were run under the Linux operating system on a PC with a 3.2 GHz Pentium 4 processor and 1 GB of RAM. For gaze tracking, we used a SensoMotoric Instruments iView X Hi-Speed eye tracker that provides gaze samples at a rate of 240 Hz. The eye-tracking software ran under Windows 2000 on a separate PC, and the gaze samples were sent to the Linux PC using a UDP network connection.

The video sequences used in the experiments were taken with a JVC JY-HD10 camera and had a resolution of 1280x720 pixels at 30 frames per second (progressive scan). These video sequences were then scaled down to a resolution of 960x540 pixels; at this resolution, with a multiresolution pyramid of six levels ($L = 5$ downsampled levels plus the original video sequence), the gaze-contingent filtering algorithm was able to process the video at the full rate of 30 frames per second.

Figure 1 shows a still frame from one of the video sequences (top) along with the image produced by the algorithm (bottom). The observer's point of regard (centre left of the image) is marked by a white square. The resolution map used in this example specified full resolution at the point of regard, with a smooth reduction towards the periphery (see Figure 4). Accordingly, the area around the point of regard is unchanged, and increasing amounts of temporal filtering (causing moving objects to blur or even vanish) are visible towards the periphery. Note for example that the two men walking at the left of the image are slightly blurred and that the person who is about to leave the image at the right edge has practically vanished completely.

We have performed experiments on the detectability of the gaze-contingent temporal filtering and of its effect on eye movements. These results have already been published [Dorr et al. 2005a; Dorr et al. 2005b], so we will only summarize them briefly.

In one study [Dorr et al. 2005a], we investigated the detection threshold of the gaze-contingent temporal filtering effect as a function of retinal eccentricity. We used a resolution map that retained the temporal resolution of the original video across the whole of the visual field except for a ring-shaped region at a certain eccentricity from the point of regard, which was set to a reduced resolution R_r. For various eccentricities, we then measured the threshold for R_r beyond which the manipulation was no longer detectable. These experiments were performed for four video sequences of twenty seconds duration each (a beach scene, a traffic scene at a roundabout and two scenes of pedestrian areas).

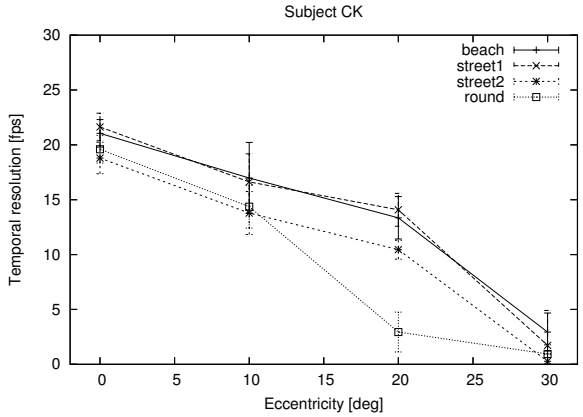

Figure 5: Detection threshold for temporal filtering as a function of eccentricity. Each line indicates results for one image sequence.

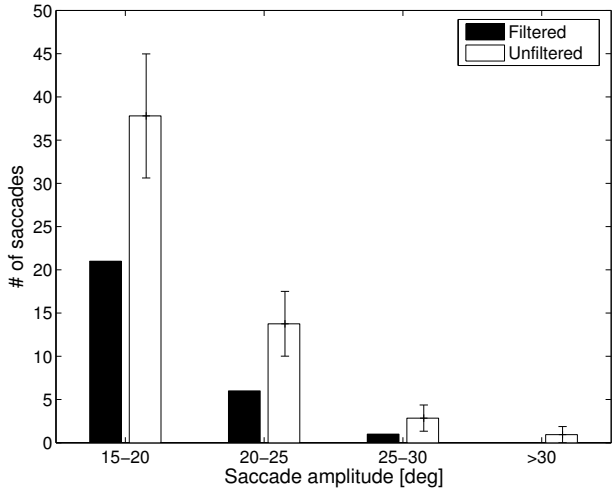

Figure 6: Histogram with error bars of saccade amplitudes with and without gaze-contingent temporal filtering. Only the histogram bins for saccades of 15 degrees and greater are shown.

Figure 5 shows the detection thresholds that were measured for one subject. It is apparent that the amount of temporal filtering that can be performed without being detected increases with eccentricity. Indeed, at high eccentricities, we can filter out almost all of the dynamic content of the video without the observer becoming aware of this manipulation. Note that we are not measuring whether the observer can detect certain temporal frequencies at all but whether the observer will notice that something is missing if we remove these temporal frequencies from the video. For example, in Figure 1, the person who is about to leave the scene at the right edge is clearly visible in the top image; but the absence of this person from the bottom image is not noticed if one does not know the original image.

In a second study [Dorr et al. 2005b], we investigated the effect that gaze-contingent temporal filtering has on eye movements. Here, we used the same resolution map that was used to generate the sample image in Figure 1, i.e. full resolution around the point of regard with a gradual falloff towards the periphery (see Figure 4).

We measured the amplitude distribution of saccades made by eight observers watching four temporally filtered video sequences (two of which were identical to ones used in the first study). The observers were instructed to watch the videos attentively, but no other specific task was given. We then compared the results to measurements made on the unmanipulated original videos (8 observers drawn randomly from a pool of 54 observers on which measurements were made, see [Dorr et al. 2005b] for details). Figure 6 shows the results. (Only saccades with amplitudes of 15 degrees or greater are shown, since this is where we obtain significant differences between the two conditions—for smaller amplitudes, no significant differences were observed. Note that the temporal filtering was strongest in the periphery of the visual field, so this is where we would expect to see the strongest effect. The error bars for the unfiltered condition show the range of variation that was obtained by drawing different random subsets of 8 observers from the pool of 54 observers; see [Dorr et al. 2005b] for details.) The plot shows that with gaze-contingent temporal filtering, the subjects made significantly fewer large saccades. This is a plausible result, since the temporal filtering removes high-frequency content in the periphery, which is usually a strong bottom-up cue for saccades.

4 Conclusions and Outlook

We have presented a new type of gaze-contingent display that manipulates the temporal resolution of an image sequence, and we have demonstrated that the temporal filtering effect can remain undetected by the observer if the cutoff frequency lies above an eccentricity-dependent threshold. We have also shown that gaze-contingent temporal filtering can reduce the number of saccades to the periphery of the visual field. We see this type of display as a valuable tool for psychophysical research on the spatio-temporal characteristics of the human visual system when presented with natural scenes.

In a wider context, our work is motivated by applications that involve the guidance of eye movements [Itap 2002]. In previous work, we showed that certain visual stimuli can trigger saccades to a chosen location [Dorr et al. 2004]. The effectiveness of this stimulation, however, may be reduced if the scene contains other salient locations that can themselves trigger a saccade. We have shown that a saliency map can be used to predict a small number of such locations that are likely candidates for saccade targets [Böhme et al. 2005]. Temporal filtering could be used to reduce the likelihood that the visual system will choose one of these locations as a saccade target; this should increase the probability that stimulation will then trigger a saccade to the intended location. We are currently carrying out experiments along these lines and plan to report on the results in a future paper.

As mentioned in the introduction, gaze-contingent spatial filtering (foveation) can be used to improve the compressibility of video. We would expect that temporal filtering also improves compressibility, but we have not investigated this since compression is not the motivating application for us.

The gaze-contingent temporal filtering algorithm presented in this paper can be combined with Perry and Geisler's algorithm [2002] for gaze-contingent spatial filtering. To do this, one would simply run Perry and Geisler's algorithm on the output image produced by the temporal filtering algorithm. This would allow full control over both the spatial and the temporal resolution at each pixel, allowing very specific experiments to be carried out on the spatiotemporal characteristics of the visual system.

Acknowledgements

Our research is supported by the German Ministry of Education and Research (BMBF) as part of the interdisciplinary project *ModKog* (grant number 01IBC01B) and by the European Commission within the COGAIN Network of Excellence. We thank SensoMotoric Instruments GmbH for their eye-tracking support; data were obtained using their iView X Hi-Speed system. We also thank the anonymous reviewers for their helpful comments.

References

BÖHME, M., DORR, M., KRAUSE, C., MARTINETZ, T., AND BARTH, E. 2005. Eye movement predictions on natural videos. *Neurocomputing*. (in press).

BURT, P. J., AND ADELSON, E. H. 1983. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications 31*, 4, 532–540.

CORNELISSEN, F. W., BRUIN, K. J., AND KOOIJMAN, A. C. 2005. The influence of artificial scotomas on eye movements during visual search. *Optometry and Vision Science 82*, 1, 27–35.

DORR, M., MARTINETZ, T., GEGENFURTNER, K., AND BARTH, E. 2004. Guidance of eye movements on a gaze-contingent display. In *Dynamic Perception Workshop of the GI Section "Computer Vision"*, U. J. Illg, H. H. Bülthoff, and H. A. Mallot, Eds., 89–94.

DORR, M., BÖHME, M., MARTINETZ, T., AND BARTH, E. 2005. Visibility of temporal blur on a gaze-contingent display. In *APGV 2005 ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization*, 33–36.

DORR, M., BÖHME, M., MARTINETZ, T., GEGENFURTNER, K. R., AND BARTH, E. 2005. Eye movements on a display with gaze-contingent temporal resolution. *Perception ECVP 2005 Supplement 34*, 50.

DUCHOWSKI, A. T., COURNIA, N., AND MURPHY, H. 2004. Gaze-contingent displays: A review. *CyberPsychology & Behavior 7*, 6, 621–634.

GEISLER, W. S., AND PERRY, J. S. 1998. A real-time foveated multiresolution system for low-bandwidth video communication. In *Human Vision and Electronic Imaging: SPIE Proceedings*, B. Rogowitz and T. Pappas, Eds. 294–305.

ITAP, 2002. Information technology for active perception website. `http://www.inb.uni-luebeck.de/Itap/`.

KORTUM, P., AND GEISLER, W. 1996. Implementation of a foveated image coding system for image bandwidth reduction. In *Human Vision and Electronic Imaging, SPIE Proceedings*, vol. 2657, 350–360.

KÖTHE, U. 2004. Accurate and Efficient Approximation of the Continuous Gaussian Scale-Space. In *26th DAGM-Symposium*, Springer, Heidelberg, C. E. Rasmussen, H. H. Buelthoff, M. Giese, and B. Schoelkopf, Eds., vol. 3175 of *LNCS*, 350–358.

LOSCHKY, L. C., AND MCCONKIE, G. W. 2000. User performance with gaze contingent multiresolutional displays. In *Proceedings of Eye Tracking Research & Applications*, 97–103.

LOSCHKY, L. C., MCCONKIE, G. W., YANG, J., AND MILLER, M. E. 2005. The limits of visual resolution in natural scene viewing. *Visual Cognition 12*, 6, 1057–1092.

MCCONKIE, G. W., AND RAYNER, K. 1975. The span of the effective stimulus during a fixation in reading. *Perception & Psychophysics 17*, 578–586.

PERRY, J. S., AND GEISLER, W. S. 2002. Gaze-contingent real-time simulation of arbitrary visual fields. In *Human Vision and Electronic Imaging: Proceedings of SPIE, San Jose, CA*, B. E. Rogowitz and T. N. Pappas, Eds., vol. 4662, 57–69.

RAYNER, K. 1975. The perceptual span and peripheral cues in reading. *Cognitive Psychology 7*, 65–81.

RAYNER, K. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin 124*, 3, 372–422.