Behavior Research Methods, Instruments, & Computers
2001, 33 (4), 524-531

A multichannel PC tachistoscope with high resolution and fast display change capability

JIE-LI TSAI
National Yang-Ming University, Taipei, Taiwan

The number of channels that PC tachistoscopes can have has increased recently (Bokhorst, 1995; Myors, 1998); however, neither the quality of display nor speed of image switching has been improved. This article shows the capability of VESA's VBE 3.0 standard (1998) for increasing the number of channels of high-quality images. And the refresh rate can be set to the fastest cathode ray tube (CRT) scanning rate the monitor can tolerate in order to reduce the timing delay of changing display. A PCTSCOPE library was written in C to provide these capabilities, which is compatible with conventional DOS real mode. The PC tachistoscope can have numbers of channels with various resolutions and colors and different refresh rates. For example, 25 images with the resolution of 640 × 480 pixels and 256 colors can be loaded to video memory, and vertical refresh rate can be set to 180 Hz. It takes less than 6 msec to change the display among 25 channels in synchronizing with the start of the video scanning frame. In this library, the number of channels, the resolution of the images, and the speed of changing display all are improved. The multichannel PC tachistoscope with this technique is especially suitable for research requiring high-quality images and rapid successive presentation of stimuli.

The purpose of the tachistoscope, which was invented by Volkmann in 1859, was to present stimuli for brief durations. Tachistoscopes have been useful in psychological studies when the precision of stimulus timing was critical, especially when the stimulus exposure time is very brief and when a set of stimuli is presented sequentially for short durations. An example is the visual masking paradigm used to study subliminal or unconscious processing (Marcel, 1983). During experiment trials, a word is presented for a variable duration and is followed by a pattern mask. The stimulus onset asynchrony (SOA) between word and mask at which the subject has difficulty in deciding whether or not a word had appeared is identified. The SOAs are quite short in order to measure the perceptual threshold for detecting a word. The precision of display timing provided by a tachistoscope is especially critical to this kind of experiment.

Today, computers have become popular for displaying stimuli in psychological experiments. Using the traditional tachistoscope to prepare stimuli for display is time-consuming and does not meet all the needs of many researchers. The merits of using computers to control tachistoscopic displays are the efficiency of stimulus preparation, of carrying out the experiments, and of collecting data for analysis, as well as the flexibility in sequencing stimulus presentations. However, there are certain limitations that must be noted when using computers for stimuli display. One of these is the limited number of channels available for creating dynamic display change, and another one is the error of display timing caused by the delay of frame refreshing.

Since much psychological research uses computers to display stimuli, it is necessary to develop the techniques that keep the virtue of the display timing precision of the tachistoscope but that take advantage of the computer's positive characteristics. The reason for a tachistoscope having three or more fields is that it allows researchers to put all the stimuli in the fields before the start of each trial; the tachistoscope can then serially light up the fields to show the stimuli and change the display rapidly. By analogy with the tachistoscope, multichannel displays for a PC tachistoscope can be produced by preloading all the images for a trial into video memory and quickly switching among the channels as needed. Recent papers have shown how to increase the number of display pages that can be stored in video memory on the PC (Bokhorst, 1995; Myors, 1998); however, the restriction is that the proposed methods can only be used in text mode or low-resolution graphic modes when 32 pages are needed (Myors, 1998), or in VGA monochrome graphic mode with 8 pages (Bokhorst, 1995).

Another limitation in using computers for tachistoscopic displays is the refresh rate of the monitors. The refresh rate is the number of times that the cathode ray tube (CRT) updates the screen from the top to the bottom in one second. While the CRT is refreshing one frame, any change made to the video memory will not be updated on the actual display until the current frame is finished and the next

This article was supported by grants from the National Science Council and Academia Sinica, Taiwan, to Ovid J.-L. Tzeng and Daisy L. Hung. I thank Jonathan Vaughan and two anonymous reviewers for their helpful comments. Special thanks to George W. McConkie of the University of Illinois, Urbana-Champaign, for the discussion of this idea and the great help on prior versions of the article, and to Gary S. Wolverton of the Beckman Institute at UIUC for helpful suggestions. Also thanks to Ovid Tzeng, Daisy Hung, and Chia-Ying Lee of our lab for fully supporting the work. The PCTSCOPE library is for research uses and can be obtained by e-mail to J.-L.T. Correspondence should be addressed to J.-L. Tsai, Laboratory for Cognitive Neuropsychology, National Yang-Ming University, Pei-Tou, Taipei 112, Taiwan (email: jerry@daisy.ym.edu.tw).

frame begins. Bridgeman (1998) showed clearly the problems caused by refreshing displays when using brief presentation times of visual stimuli on computers (also see Krantz, 2000, for a review of timing a stimulus of an image). An algorithm was provided to have a more accurate estimation for actual stimulus durations (Bridgeman, 1998). Because of the updating process for computer displays, it is necessary to synchronize the timing of the display change with the start of the frame cycle in order to eliminate error caused by the delay of updating the changed display. And the speed of screen updating rate determines the time resolution of both changing the display and the duration of displayed stimuli. With a higher refresh rate, the control for display changes can be faster and display durations can be shorter. The default vertical refresh rate of 60 Hz in standard VGA graphic modes, 17 msec for updating the screen area once, is apparently too slow to run experiments that require brief displays of visual stimuli.

The present paper describes a library of C functions called the PCTSCOPE library, which is able to resolve the problems just described. Its capability includes preloading many high-resolution images in video memory, changing the display among these images quickly, and increasing the refresh rate of the monitor. This library is appropriate for computerized experiments that demand high-quality images and precise display timing control. This article first describes the required hardware. It then shows the capabilities and the ease of using the routines in the PCTSCOPE library. To explain how these simple routines work, the necessary technical information is introduced in the following sections. These sections describe how to preload multiple high-quality images and how to specify one of them to display. It also shows a method for synchronizing the display change to the refreshing frame and for setting the refresh rate to accelerate the display change. The discussion shows an application to an eye movement study requiring fast display changes in response to subjects' own eye movements (McConkie, Wolverton, & Zola, 1984).

METHOD

The PCTSCOPE library has a set of routines written in C, compiled with Borland C++ 3.1 in large memory mode and developed in a DOS real-mode environment that is compatible with most software for the real-time control of experiments. All the routines were developed for the VESA's graphic modes and it follows the VESA VBE (VESA Bios Extension) 3.0 standard.

The Hardware Requirements

The required hardware for the PCTSCOPE library includes a multifrequency color monitor and a VGA card that supports VESA's graphic modes. The higher the frequencies that the monitor can tolerate and that the VGA card can support, the higher the refresh rate can be set. The more memory the VGA card has, the more images can be stored. There are some hardware restrictions that should be noted. First, the maximum memory that can be currently accessed is 8 MB. This is caused by a limitation of memory addressing in the DOS real mode. Second, the VGA card must support VESA VBE 3.0. Some of the functions, especially the refresh rate control, work only for VBE 3.0 standard and not for earlier versions. If VBE 3.0 is not available on the VGA card, VESA drivers supporting VBE 3.0 (e.g., UniVBE by SciTech company) can be used but some of the graphic modes may not be available. As an example, our work is carried out using a 19-in. monitor, ViewSonic PT795. The vertical frequency ranges from 50 to 180 Hz and the horizontal frequency ranges from 30 to 110 kHz. The display controller card is an ASUS AGP-V3400TNT with 16 MB SDRAM. It supports VESA DDC2B+ and VBE 2.0/3.0.

Showing and Changing Images With the PCTSCOPE Library

The usage of the PCTSCOPE library is very easy and clear. A program only needs to include the head file "pctscope.h" and declare two public variables, ActivePage and VisualPage; then all the routines in this library can be called. The Appendix shows a sample program that demonstrates how easy it is to use the relevant display routines to load the images and change between them. The general procedure for doing tachistoscopic displays is as follows. First, to initialize the VESA graphic mode, assign the values of the graphic mode number and the refresh rate by calling initVbeMode(*ModeNo*, *RefreshRate*). The resolution and colors for all indexed numbers of VESA graphic modes are listed in the head file "pctscope.h." Both the resolution and the maximal refresh rate of a graphic mode are constrained by the capabilities of the display card and the monitor. Also the number of pages available depends on the resolution of the desired mode. For example, the VESA mode 0 × 101 is defined for the mode with the resolution of 640 × 480 pixels and 256 colors. In this mode, 25 images can be preloaded and the refresh rate can be set to 180 Hz, with the hardware described in the prior section (Table 1).

Second, the value of the variable ActivePage specifies which page will be activated for drawing images. There are routines for drawing on the page assigned by ActivePage. For graphic modes with 16 colors or 256 colors, vbe_putpixel(*X, Y, ColorIndex*) draws a pixel at the specified location. And an image file with PCX format can be loaded by using load_PCX_file(*X, Y, Filename*). For graphic modes with 64K or more colors, vbe_putpixel_ rgb(*X, Y, RedValue, GreenValue, BlueValue*) is used for drawing a pixel and load_JPG_file(*X, Y, Filename*) can load an image file with JPEG format.

Once all images have been drawn on their different pages, the routine visualPage(*PageNo*) is called to switch to the page that is wanted on the screen. To avoid the problem of the display tearing apart while changing among pages, caused by switching pages while refresh scanning is in progress, the vsync() routine should be added before visualPage().

The routines in the PCTSCOPE library cover all the actions for tachistoscopic displays (Table 2). They include those needed to store images, to switch between images,

Table 1
The Available Pages and Refresh Rates on Different Resolutions of VESA Modes

VESA Mode No.	Resolution	No. Colors	Pages Available With 8 MB Memory	Maximum Refresh Rate (Hz)	Time for One Scanning Frame (Milliseconds)
101h/111h/112h	640 × 480	256/64K/16.8M	25/12/6	180	5.56
102h/103h/114h/115h	800 × 600	16/256/64K/16.8M	32/16/8/4	167	5.99
104h/105h/117h/118h	1024 × 768	16/256/64K/16.8M	16/10/5/2	133	7.52
106h/107h/11Ah	1280 × 1024	16/256/64K	10/6/3	102	9.80

Note—The capabilities of the PCTSCOPE library were tested on a ViewSonic PT795 monitor and an ASUS AGP-V3400TNT display card. M, million.

to synchronize the display change to the refreshing frame, and to increase the refresh rate. Actually there are many basic elements that make these actions possible. The following sections introduce the procedures and methods to do tachistoscopic displays on computers, showing how these simple routines are built on numerous parameter settings and calculations.

Storing High-Quality Images in Video Memory

There are several ways that the display on a PC tachistoscope can be quickly changed. A new image can be computed from a file and drawn at the moment a change is desired; however, this is time-consuming if much of the image must be recomputed. Alternatively, the images can be prepared in advance and stored in the computer's semiconductor memory, being moved to video memory when a stimulus change is desired. This method is more efficient but it works only for those graphic modes with lower resolution. The architecture of the DOS environment allows allocating only 64K (65,536 bytes) of memory as a video buffer for a VGA display. The video buffer cannot handle any image with higher resolution and more than 16 colors. It cannot set graphic modes that require more than 64K of video buffer. For example, an image with 640 × 480 resolution and 256 colors requires 307,200 bytes for the display information. The required memory for such a graphic mode is about five times the size of the video buffer.

SuperVGA provides a bank-switching method for displaying an image that needs higher resolution and more colors. The memory on the display card can be divided into several banks of 64K bytes, and these banks can map to the single DOS video buffer one at a time to compose the SuperVGA image for the display (Figure 1). Any action for reading or writing a pixel requires switching to the corresponding bank in order to map to the video buffer and then being able to change the content of the memory address.

Therefore, if the required memory for an image is larger than the video buffer, the image has to be separated into parts in the banks of video memory. Even if the image is loaded in the computer's semiconductor memory, displaying it requires a series of bank switches, transferring part of the image into the video buffer with each switch and sending that part to the screen. This bank-mapping approach to changing the display needs to split the image and transfer the parts one at a time, which is often still not fast enough.

The fastest method of showing a high-quality image is to compute all needed banks and then to store them in video memory ahead of time so they are ready for immediate access. Mapping the banks of the image to the video buffer can be accomplished by internal commands in the VGA adapter. Only one thing is required for showing an image: assigning the beginning address of the video memory where the image is located. Switching among the images can then

Table 2
Description of Main Functions in PCTSCOPE Library

Function	Argument	Description
initVbeMode(mode, rfrate)	int mode	The number for a specified VESA mode
	float rfrate	The refresh rate for the assigned mode
setVbeMode(mode, rfrate)	int mode	The number for a specified VESA mode with the default refresh rate
visualPage(pageno)	int pageno	The page number shown on the display
vsync(void)		Synchronize with the scan frame
vbe_putpixel(x,y,color)*	int x,y	The *x* and *y* coordinates for a pixel
	int color	The color index
vbe_putpixel_rgb(x,y,r,g,b)†	int x,y	The *x* and *y* coordinates for a pixel
	int r,g,b	The red, green, blue values
load_PCX_file(x,y,filename)*	int x,y	The *x* and *y* coordinates for a PCX file
	char *filename	Load an image file with PCX format
load_JPG_file(x,y,filename)†	int x,y	The *x* and *y* coordinates for a JPEG file
	char *filename	Load an image file with JPEG format
clearPage(pageno,color)	int pageno	The page number to be cleared
	int color	The color number for the background

*For graphic modes with 16 or 256 colors. †For graphic modes with 64K or 16.8M colors.

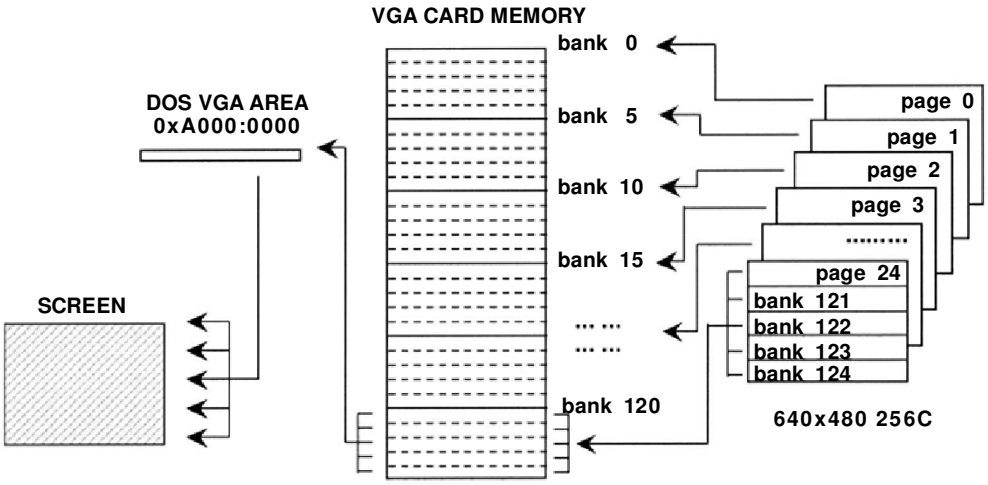

Figure 1. A simple diagram shows the mapping between pages of images and banks of video memory.

simply be like opening and closing shutters when stimulus changes are needed.

Most of today's display cards have sufficient RAM on board to store many images. To utilize all of the video RAM, it is convenient to store the images under Windows 9x or DOS protected mode because all the memory can be addressed directly by the linear frame buffer method. In DOS real mode, with its limited memory addressing ability, it is necessary to use the banking method to access all of the video memory. The banking method is slower than direct addressing because it is necessary to calculate where to switch to a different memory bank when drawing pixels. However, if the bank-switching method can be used to store all the images in the video memory before starting a trial, then any of the images can be quickly displayed by just specifying its starting address in the video memory. This approach shifts image creation and loading time to the noncritical time prior to the trial, eliminating delays for loading images during critical times.

Using the VESA's function call 4F01h, the computer can acquire the necessary information to compute the number of banks of memory for an image in the current graphic mode. The area of memory required to hold one image will be referred to as a page. In the equation given below, one more bank needs to be added for a page if the memory size is not evenly divisible by 64K.

```
BanksPerPage = ((Height*BytesPerLine +
Width*BytePerPixel)/(64*1024))
```

The memory on the display card can be accessed to show a high-quality image by the bank-switching method. Therefore all of the video memory on the display card would be accessible by the same method. With a display card having 8 MB of memory, a total of 128 banks of memory can be accessed. One image with the resolution of 640 × 480 pixels and 256 colors needs roughly five banks of memory. Thus, 25 images of this size can be stored in 8 MB of memory. Actually there is enough memory to store 27 such images (8 MB divided by 307,200 bytes). However, the work is reduced if images are loaded in such a way that the beginning memory addresses of images are always at the beginning of a bank. It is then simpler to calculate the offset memory address of a pixel location on a given page. The number of pages that can be stored varies with different resolutions and depends on how much memory is needed for a display.

The PCTSCOPE library defines a variable called ActivePage that indicates which page on the display card (with pages numbered sequentially from 0) is active for drawing the image, but this page is not necessarily the one showing on the display. With the information of the assigned VESA graphic mode, the bank number and the offset from start address of any pixel in the active page can be calculated. For example, given a pixel at position (x, y) on page N, the corresponding bank number is N multiplied by banks per page. And the offset memory address from the beginning of the corresponding bank can also be obtained. Given

```
ActivePage = N,
BankNumber = N*BanksPerPage + ((y*Bytes-
PerScanLine + x*BytesPerPixel)/64K)
OffsetAddress = (y*BytesPerScanLine +
x*BytesPerPixel) & 0xffff
```

With the value of BankNumber, the setbank() routine will switch to the corresponding bank. And the OffsetAddress decides the memory address offset from that bank. The drawing routine then uses this information to assign the appropriate bank and put a pixel with its color on any position of the requested page. The drawing function calls write directly to the video memory; therefore, it is different in calculating the corresponding memory address for each pixel for different color depths of graphic modes. The PCTSCOPE library provides two sets of routines to draw a pixel or load an image file to any page in different graphic modes. The vbe_putpixel() and load_PCX_file() can be

used for graphic modes with 16 or 256 colors, and vbe_putpixel_rgb() and load_JPG_file() for graphic modes with 64K or more colors. Drawing pixel by pixel takes longer to load images, especially with higher resolution and more colors; however, the image loading time typically occurs between trials during an experiment, and the loading time is acceptable when one is using a computer with a Pentium CPU.

In summary, to store an image, this library acquires the information by using VESA's function call 4F01h to get the number of banks for each page in the current graphic mode. To put a pixel at any location of the assigned page using the drawing routines, the corresponding bank is identified by the information of ActivePage, banks per page, and the location being drawn. Also the offset from the beginning of that bank can be calculated automatically. The drawing routines use all the necessary information to assign the appropriate bank using setbank() and put pixels at the corresponding memory address.

Specify an Image to Display

Once the images are stored in the memory of the display card, they can be switched on and off rapidly for the tachistoscopic display. One method for showing an image involves copying the content of the first bank for one page to the video buffer, switching to the next bank, and copying again, and repeating this process until all banks for the page have copied. However, the bank switching and copying process are too slow for many purposes. A faster method is to use VESA's function 4f07h, which provides an efficient method for display switching. It can change the start address of the video buffer to any offset address of the memory on the display card; the contents of the following memory locations are then displayed on the screen. For example, the offset address is 0 for the beginning of the first bank 0, 5*64K for bank number 5, and so on. With five banks as a single page, the offset address can be obtained by PageNumber*BanksPerPage*64K. Assigning the offset address for any page to VESA's function 4f07h, the display card will switch to display the specified page very quickly. The visualPage() function in this library is used for assigning any one of the pages in memory to be shown on the display. Passing the page number to visualPage(), it calculates the offset address of the assigned page and passes it to VESA's function 4f07h, causing the display to change immediately.

The library keeps the page number of the current display in the VisualPage variable and the value of VisualPage is assigned through the argument of visualPage() function. The drawing routines will not change the display on the screen unless ActivePage and VisualPage have the same page number. By default, both the ActivePage and VisualPage are set to 0, which is the first of all pages and also the page shown on the screen when the graphic mode is being initialized. If there are several images to be loaded into different pages, but these are not to be displayed as they are being loaded, then it is important that the value of the ActivePage not be the same as the value of the VisualPage.

Synchronizing the Display Change to Frame Refreshing

For standard VGA graphic modes with a resolution of 640×480 pixels, the vertical refresh rate is 60 Hz. At this speed, the CRT takes about 17 msec to scan and update the whole display from the top line to the bottom line of the screen. While the CRT is refreshing one frame, any change to the video buffer will have no effect on the displayed arrays until the current frame is finished. At that point, the computer begins displaying the newly specified image. Thus, even though a new image is specified, it may be several milliseconds before the computer begins to display it. Thus, the time between the moment that image A is requested until the moment that image B is requested is not necessarily the amount of time that image A is displayed. For example, if image A was requested just after a scan of the previous image begins, and image B is requested 14 msec later, just before that scan is ended, image A would in fact never appear on the screen, even though the program might report that it had been displayed for 14 msec. In general, the actual display time can differ from the time between image requests by the time of a single frame (less 14 msec) or, under certain conditions, even more for specific parts of the image. The results of the experiment can thus be contaminated if the time between image requests is very brief or the changing timing is crucial. Thus it is important to synchronize the timing of the display change with the start of the frame cycle in order to eliminate the error caused by the delay of updating the changed display.

There is an input status register, 0x3da, on the VGA control, which a program can check to see whether vertical scanning is in progress or not. By continuously checking this register, a program can wait until a refresh scan is ended before initiating a display change. With the function vsync() shown below, the display change can be delayed in order to synchronize with the start of a refresh scan.

```
void vsync(void)
{
while(!(inp(0x3da) & 0x08));    /* wait
for end of retrace */
while(inp(0x3da) & 0x08));      /* wait
for start of retrace */
}
```

The PCTSCOPE library has the tools necessary for storing multiple high-quality images and quickly switching among them. Furthermore, synchronizing the display change to frame refreshing can reduce the delay error of presentation times. When using the library in practical programming, it is only necessary to set a page number to ActivePage and then use the drawing routines to put pixels on the assigned page. Once all images are loaded, the display can switch rapidly to show any page by simply assigning the page number to the visualPage() function. The vsync() function can be called before the visualPage() in order to synchronize the page switching to the frame refreshing. However, if the display change has to wait until the beginning of the next frame before it can begin, then refresh rate will directly affect how quickly the page can be

changed, following the request for a new image. The following section shows how to accelerate the speed of switching by increasing the refresh rate.

Refresh Rate Control

With the core functions of VBE 3.0, it is possible to control the refresh rate for the graphic modes being set in the real-mode DOS environment. Control of refresh rate is accomplished by passing a set of CRT controller (CRTC) parameters and pixel clock values to function 4F02h. The computing of CRTC timings is quite complicated. Briefly, initializing a VESA graphic mode with a specified refresh rate requires the following procedures. First, the information for the speed limitations of the monitor must be obtained by using VBE/DDC, if available. Second, generalized timing formula (GTF) is used to compute the necessary parameters for the specified refresh rate and the normalized pixel clock. Finally the resulting CRTC timings are passed to function 4F02h to initialize the graphic mode with the acquired refresh rate. The VBE 3.0 standard itself does not provide any means to compute these parameters; however, the MGL library distributed by the SciTech Company provides the source of functions to compute the desired values for controlling the refresh rate. The MGL library with the source code is available on the SciTech's website (http://www.scitechsoft.com/). The function described here for initiating the VESA graphic modes and setting the refresh rate is from part of the source code in the MGL library.

The VESA's function 4f02h can initiate various SVGA graphic modes and can control the refresh rate in the VBE 3.0 standard. There are two parameters involved in setting the refresh rate in the VESA's function 4f02h. The BX value holds a specific number for the assigned VESA graphic mode in the first 8 bits, and the 11th bit of BX can enable or disable the refresh rate control. For example, to enable refresh rate control for a VESA graphic mode 0x102, it can be done with the OR operation of 0x0102 and 0x0800. The second step is to point the structure of CRTC values to ES:DI when the CRTC values have been obtained by GTF computations.

In the PCTSCOPE library, the initVbeMode() routine includes the procedures to initiate the graphic mode and set the refresh rate. It is important to know the capability of the display card and the limitations of the screen under the desired resolution. For example, the ASUS V3400TNT display card can support up to 250 Hz of vertical refresh frequency with the resolution of 640 × 480. But the ViewSonic PT795 monitor is limited to 180 Hz at that resolution. Thus, the maximum refresh rate should be set to 180 Hz. Moreover, the vertical frequency is correlated with horizontal frequency as well as with the bandwidth. So all these limitations must be taken into consideration. If the refresh rate settings exceed any of the limits, the display will be distorted or the CRT may be damaged.

DISCUSSION

The PCTSCOPE library provides a solution for using high-quality images in a PC tachistoscope with precise timing control in the limited DOS environment. The required monitor and display card are available on the commercial market and they are not expensive. It is easy to create new C programs or to add to existing C programs with the routines in this library. The computer can be programmed to act as a multichannel tachistoscope with an increased number of channels without sacrificing resolution and colors, and the time required for a display change can be reduced by increasing the refresh rate. Research using eye movement contingent display control methods, in which the stimuli on the display change at times indicated by where and when the subject's eyes move, is particularly demanding.

For example, the moving window method used to study the perceptual span in reading (McConkie & Rayner, 1975; McConkie, Zola, Wolverton, & Burns, 1978) requires that the display be changed in response to the subject's eye movements on a line of text, creating a high-resolution window where the subject's gaze is directed, with other stimuli or lower resolution outside this area. The display change must be made within the duration of a saccade or at the very beginning of a fixation while saccadic suppression is high in order to minimize interference while the subject reads the text. Some saccades are less than 20 msec in duration, thus putting strong constraints on the time available to detect the onset of the saccade, estimate where the eyes will stop and actually change the image to place the high-resolution window at the desired location. Those experiments using eye movement contingent display demand a more advanced technique for tachistoscopic display than the methods described by Bokhorst (1995) and Myors (1998).

As an example of this method, a study was conducted in which a computerized display of this type was used to investigate the perceptual span in Chinese reading. For this type of work, a PC tachistoscope needs several capabilities to create the moving windows. First, it needs a high-resolution image in order to show Chinese writing clearly on the display. Chinese must be shown in graphic modes and each character is usually composed of a bitmap of 24 × 24 pixels in experiments. A resolution of 320 × 200 pixels is insufficient to show the number of characters typically needed for a sentence. Also the shape of the character strokes on the display is too coarse. Displays of Chinese text demand a resolution of 640 × 480 pixels or higher in order to keep the size of a character to about one degree of visual angle. Second, a Chinese language tachistoscope needs a number of pages in order to create the dynamic changing display. In a standard moving window experiment, only a certain amount of correct text is shown, that lying within a specified region relative to the reader's gaze location, and the remainder is masked on every eye fixation. The text window must change according to where the eyes stop on each eye fixation. Redrawing part of the text at the beginning of each eye fixation is too slow. One way to accomplish the rapid display change is to preload a number of versions of the text including all the possible text window locations and then switch among these text images dynamically in response to their gaze location at the beginning of each fixation. Finally, the display change of

the moving windows has to be as fast as possible, requiring a fast refresh rate. There is only 15 msec or less in which to change the display for the eye movement contingent display. The refresh rate of a standard VGA graphic mode, 60 Hz, is not fast enough. It is necessary to increase the refresh rate in order to shorten the time of waiting for the synchronization point. This time can be further shortened by disabling the frame synching feature, causing the image to change as soon as it is requested, rather than waiting until the end of the frame. Since any tearing of the image will occur during or shortly after the saccade, at the time when saccadic suppression reduces visual sensitivity, such stimulus aberrations are not likely to be detected.

One study that implemented this technique investigated the perceptual span of Chinese passage reading (Tsai, Tzeng, Hung, & Yen, 2000). The eye movements are recorded via the EYELINK eyetracker system, which has a sampling rate of 250 times per second. Two Pentium II 333 computers are interconnected via an ethernet cable. One computer is for monitoring the eye movements and the other one is for stimulus display. The computer for stimulus display is equipped with a ViewSonic PT795 monitor and an ASUS V3400TNT-AGP display control. The program for the experiment combines the PCTSCOPE library and the functions for eye tracking supplied by SR Inc., developer of the EYELINK eyetracker. During the experiment, one line of a passage was shown at a time and changed to the next when the subject had read that line. Twenty-five Chinese characters in 24 × 24 pixel bitmaps were shown horizontally in the graphic mode with a resolution of 800 × 600 pixels and 16 colors. This mode was initialized by the initVbeMode() routine with a 167-Hz refresh rate, which is the limitation for the ViewSonic PT795 monitor.

As a subject read each line, the display was changed at the beginning of each eye fixation to create a window for restricting the area of reading. In the extreme case, only one character (the character to which the eyes were directed) was from the original sentence and the rest of the characters were replaced with characters of very low cultural frequency. In this situation, there are 25 possible line configurations, one for each character position to which the gaze could be directed during a fixation. All these 25 images, each having a different window location, were loaded into display memory before beginning the sentence presentation. Then with each eye movement, the display was changed to the image that corresponded to the location at which the eyes landed. In this way, the window showing normal text moved to wherever the eyes went. To reduce visual interference caused by the changing display, the image changed to all low-frequency characters when the subject's eyes began a saccade. Once the eyes moved and were slowing to a stop for the next location, the visualPage() function was called to change the display, specifying the image that would put the window at the location of the new eye fixation.

To validate the timing of the changing display, any of several methods can be applied. First, the refresh rate of the current status is available by user controls in most of the monitors; otherwise, an oscilloscope can be used to get this information. Second, a timer program can be used to measure the time spent waiting for synchronization. For example, the eye movement recording program included in the EYELINK software package can insert text as markers in the data file, thus providing time stamps to indicate the moment at which a call was made to change the display and also when the display change had been completed after vsync(). When the refresh rate was set to 180 Hz, the result showed that the time from synchronizing with the CRT scan to when the change was completed was within 6 msec, the duration of one refresh cycle. Third, a similar method showed that the time for visualPage() to switch images was less than 1 msec. A program using the same procedure as Myors (1998) was run to estimate the time for switching to an image by visualPage(). It took 40.5 μsec for visualPage() to switch images on a Pentium II 333 computer with an ASUS V3400TNT-AGP display card. Of course, additional time is required for the display controller to reach the end of the current frame and then paint the new image on the screen.

With the technique of refresh rate control and rapid switching of high-quality images described here, both the precision and speed of image switching are improved for PC tachistoscopic presentations. Also, the accessibility of the memory banks on the display card makes it possible to increase the channels of graphic display, though further tests must be done to determine how to gain memory accessibility beyond 8 MB. The stimuli for a trial can be prepared by loading them into different pages before initiation. The concerns about the time required for allocating system memory or moving an image buffer into display memory are eliminated. And the time for drawing images is also eliminated by having the operation occur prior to the beginning of the trial. It is versatile and can be broadly applied to many areas of research, including eye movement contingent display control studies, experiments using graphic stimuli, and animated display of motions.

This article presents a way to advance applications of PC tachistoscopes for psychological experiments. The techniques described are especially suitable for research demanding high-quality images and rapid, successive presentation of stimuli. The routines are included in the PCTSCOPE library and should be compatible with other regular function calls. They are easy to implement and combine with existing programs. It is hoped that this technique will improve on existing PC tachistoscopic presentation techniques and extend the possibilities of research.

REFERENCES

Bokhorst, F. D. (1995). Bit-plane layering for high-resolution EGA and VGA graphics on the IBM PC/XT/AT. *Behavior Research Methods, Instruments, & Computers*, **27**, 496-501.

Bridgeman, B. (1998). Durations of stimuli displayed on video display terminals: $(n - 1)/f +$ Persistence. *Psychological Science*, **9**, 232-233.

Krantz, J. H. (2000). Tell me, what did you see? The stimulus on computers. *Behavior Research Methods, Instruments, & Computers*, **32**, 221-229.

Marcel, A. J. (1983). Conscious and unconscious perception: Experi-

ments on visual masking and word recognition. *Cognitive Psychology*, **15**, 197-237.

McConkie, G. W., & Rayner, K. (1975). The span of the effective stimulus during a fixation in reading. *Perception & Psychophysics*, **17**, 578-586.

McConkie, G. W., Wolverton, G. S., & Zola, D. (1984). Instrumentation considerations in research involving eye-movement contingent stimulus control. In A. G. Gale & F. Johnson (Eds.), *Theoretical and applied aspects of eye movement research* (pp. 39-47). Amsterdam: North-Holland.

McConkie, G. W., Zola, D., Wolverton, G. S., & Burns, D. D. (1978). Eye movement contingent display control in studying reading. *Behavior Research Methods, Instruments, & Computers*, **10**, 154-166.

Myors, B. (1998). The PC tachistoscope has 32 pages. *Behavior Research Methods, Instruments, & Computers*, **30**, 457-461.

Tsai, J. L., Tzeng, O. J. L., Hung, D. L., & Yen, N. S. (2000). *The perceptual span in reading Chinese passages: A moving window study of eye movement contingent display.* Paper presented at the annual meeting of the Chinese Psychology Association, Taipei, Taiwan.

VESA Bios Extension (VBE) core functions standard Version 3.0. (1998). San Jose:Video Electronics Standards Association. Available: http://www.vesa.org/vbe3.pdf

APPENDIX
A Sample Program to Load and Switch Between Images With the PCTSCOPE Library

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include " pctscope.h"
int ActivePage=0,VisualPage=0;
void main(void)
{
int p=1;
initVbeMode(0x112,180.0);           /* Set the mode to 640x480 16.8M colors, */
                                    /* and the refresh rate to 180.0 Hz */
visualPage(0);                      /* show page 0, it is the default page. */
ActivePage=1;                       /* Select page 1 for drawing pixels */
load_JPG_file(0, 0, "pic1.jpg"); /* Load and draw a JPEG image on page 1 */
ActivePage=2;                       /* Select page 2 for drawing the image */
load_JPG_file(0, 0, "pic2.jpg"); /* Load and draw a JPEG image on page 2*/
getch();                            /* Press any key to switch between images continuously */
while(!kbhit()) {
vsync();                            /* Synchronize with the retrace cycle  */
visualPage(p); /* Switch between page 1 and page 2 on the display */
p = 3 - p;
}
getch();
setVbeMode(0x03);                   /* Reset to the text mode */
}
```

(Manuscript received September 27, 2000;
revision accepted for publication June 13, 2001.)