COMPUTER TECHNOLOGY

EMDISP: A visual display system with digital and analogue sampling

MARTIN SHEPHERD

University of Durham, Durham, England

EMDISP provides general-purpose assembly language routines that can be called from higher level languages to produce specific application programs for experiments involving visual displays, manual responses, and analogue sampling (e.g., eye movement recording). The display system assumes the use of fast-phosphor oscilloscopes driven by 12-bit digital-to-analogue converters (DACs) to produce point-by-point displays. Within these limitations, EMDISP offers a flexible and efficient method of generating displays, in which elements of a complex display are defined as separate "pictures" and are controlled independently. High-level control of digital and analogue sampling are provided, enabling interactions between the contents of the display and analogue and digital inputs.

The described program was developed from a series of experiments on spatial attention and the relationship between spatial attention and eye movements. In those experiments, subjects were required to monitor rapidly changing visual displays and to make manual responses (via a keypad) to selected stimuli. One example of such a display is shown in Figure 1.

This display is from an experiment by Mueller (1984) that was designed to assess the effects of two types of spatial cues on the detectability of stimuli in peripheral vision. The sequence of events on each trial is represented by successive "frames" in Figure 1 (note that Figure 1 is not drawn to scale: the center of each of the peripheral boxes is 4.33 deg from the center of the display, and each box subtends a visual angle of 1 deg). In Frame I, the subject is fixating the central cross; fixation is maintained throughout the trial. In Frame II, the cross is replaced by a "T" in one of four orientations. The "T" remains visible for 1,500 msec. In Frame III, the "T" is replaced by the cross. One thousand milliseconds later, the cue appears. The cue is either a 50-msec brightening of one of the peripheral boxes (Frame IVa) or a 300-msec central arrow pointing to one of the peripheral boxes (Frame IVb). Both types of cue are informative: The cued position is three times more likely to contain the target than is any one of the uncued positions. The cue is followed after one of a variety of intervals by a target, which is a "T" in one of the peripheral boxes. There are two conditions of target presentation: the "single display," in which the other

three boxes are empty (Frame Va), and the "multiple display," in which the other three boxes contain distractor stimuli (crosses of the same bar size as the "T"; Frame Vb). In both types of display, the target and distractors are followed by contour masks (Frame VI). The subject's task is to press a key as quickly as possible to indicate whether the target "T" is in the same orientation as the original "T." The target duration is determined separately for each subject to give 75% correct performance, using a threshold-determination procedure devised by Findlay (1978). Target durations are typically 60 msec for the single display and 80 msec for the multiple display. The main variable of interest in this experiment is the way in which the effects of different types of spatial cue vary with the interval between cue onset and target onset (SOA).

An example of an application involving continuous monitoring of eye movements is a study on the effects of saccadic tracking movements on the detectability of peripheral stimuli (Shepherd, 1984). In this task, subjects track a spot that moves among three positions (center, 1.5 deg right, and 1.5 deg left), and make keypress responses to peripheral targets. Targets are presented at a variety of intervals following the arrival of the eye at the center position; thus, "center" has to be defined within certain limits, and eye position has to be monitored continuously to determine when the saccade to the center is made. The definition of the center position is fairly strict (about .5 deg in each direction) and is updated after each target presentation on the basis of the eye movement record for that trial in order to correct for slow "drift" in the eye movement recording system. Eye movements are monitored using a bifurcated fiber optic method (Findlay, 1974), although the programs described here are not restricted to any par-

I would like to thank Hermann Mueller for permission to use his displays as an example. My mailing address is: Department of Psychology, Science Laboratories, University of Durham, South Road, Durham DH1 3LE, England.

Figure 1. Visual display used in an experiment on spatial cuing by Mueller (1983). Successive frames (I to VI) show the possible sequence of events on a single trial.

ticular method of recording or, indeed, to eye movements at all: Their modularity ensures that the user can select whichever functions he or she requires, and trade-offs between different functions can be achieved. For example, EMDISP can be used to sample up to eight channels of EEG, as long as display requirements are minimal (the digital output routine DIGOUT can be used to control an independent display device such as a tachistoscope or slide projector). Alternatively, it can be used for word recognition experiments, in which display requirements are high and analogue sampling is not required.

SPECIFICATIONS

The programs had to include: (1) separate control of different parts of the display, different display elements having independent onsets and offsets, and levels of brightness; (2) the possibility of making changes in a display contingent on eye movements; (3) a user-controlled refresh rate (minimum of 50 Hz); (4) a user-controlled sampling rate (usually 100 Hz); and (5) a resolution of 1 msec for reaction times (manual responses).

The first of these requirements was met by defining display elements as "pictures," where a picture (a set of points) is refreshed at the specified interval independently of other pictures and that, once defined, can be switched on or off with a single subroutine call that refers to it by number. Brightening or dimming of a picture is achieved by changing its refresh rate. (The details of the picture-definition and refresh system are described elsewhere in this paper.) The methods used to meet the other requirements will be apparent in the next section, which describes the operation of EMDISP as seen by the user.

OPERATION OF EMDISP

The user program communicates with EMDISP via a common data area and via subroutine calls. The con-

Table 1
Common Block

Word	Label	Function/Description
1	KINT	Interrupt counter
2	IRESP	Digital input word
3	NSAMP	Number of samples
4	ISAMP	Most recent sample

Table 2
User-Callable Subroutines

Name	Arguments	Function
EMOPEN	9	Initialize all functions
EMCLOS	0	Shut down all functions
CLKON	0	Enable clock interrupts
CLKOFF	0	Disable clock interrupts
DIGON	0	Enable digital interrupts
DIGOFF	0	Disable digital interrupts
ADCON	0	Enable analogue sampling
ADCOFF	0	Disable analogue sampling
PICDEF	3	Define picture
PICON	1	Switch picture on
PICOFF	1	Switch picture off
CLWAIT	2	Wait a specified time
ERRCHK	1	Get error count
CLZERO	0	Set KINT to zero
CLTIME	1	Get current value of KINT
ADCNT	1	Get current value of NSAMP
ADSAMP	1	Get current value of ISAMP
DIGIN	1	Get current value of digital input
DIGOUT	1	Send value to digital output

tent and function of the common area are summarized in Table 1.

KINT counts interrupts; an operator may use it like a stopwatch, setting it to zero at the beginning of a timed period and waiting for it to reach the desired value. IRESP holds the value of the digital input word (if digital input has been enabled by a call to DIGON). If analogue input is enabled (by a call to ADCON), NSAMP holds the number of samples taken (since the last call to ADCOFF) and ISAMP holds the most recent sample. The names and functions of subroutines are summarized in Table 2.

A typical user program includes the following steps:

(1) Define the common data area. In FORTRAN, this is: `COMMON/INTCOM/KINT,IRESP,NSAMP,ISAMP`. Because there is no way to do this in BASIC, this step will have to be omitted; communication between a BASIC main program and EMDISP must take place entirely by subroutine calls (subroutines `CLZERO`, `CLTIME`, `ADSAMP`, `ADCNT`, and `DIGIN` are provided for this purpose; they will not need to be used unless the main program is in BASIC).

(2) Call `EMOPEN` to define the interrupt rate, refresh rate, analogue sampling rate, number of channels, first channel, DACs for x and y, and working arrays. `EMOPEN` also initializes various data areas (including the `COMMON` block), saves the interrupt vectors and priorities of the devices to which it requires access, and protects the interrupt vectors from interference by other programs in a foreground-background environment. `EMOPEN` allows access to all other routines and therefore must be called first. It has nine arguments (legal ranges are given in square brackets): (a) interrupt interval—interval (in microseconds) between interrupts [500-20000]; (b) refresh interval—interval (in interrupts) between refreshes [1-40]; (c) sampling interval—interval (in interrupts) between analogue samples [0-100]; (d) number of analogue channels [0-8]; (e) first channel number [0-7] (note—channels will be sampled in ascending order, starting at the first channel number, in a “group” (i.e., within one interrupt); (f) DAC channel number for x [0-4]; (g) DAC channel number for y [1-5]; (h) display array—the user must provide an array large enough to hold all points to be displayed plus the number of pictures defined; and (i) analogue samples array—the user must provide an array large enough to hold the maximum number of samples to be taken in the longest `ADCON-ADCOFF` program section.

(3) Call `PICDEF` to define pictures. `PICDEF` has three arguments: (a) picture number [1-20]; (b) address of input array—the array containing the xy pairs to be defined as a picture (`PICDEF` expects alternate x and y values in the input array); (c) number of points—number of xy pairs in (b).

Following this basic setup procedure, the remaining subroutine calls can be used freely to start and stop clock interrupts (`CLKON` and `CLKOFF`), to wait for a specified time (`CLWAIT`), to start and stop digital interrupts (`DIGON` and `DIGOFF`), to switch pictures on and off (`PICON` and `PICOFF`), and to enable and disable analogue sampling (`ADCON` and `ADCOFF`). The next section describes each of these functions in turn.

Timing Functions

Before any timing functions can be performed, the clock must be started and clock interrupts enabled with a call to `CLKON`. The interrupt counter `KINT` is then incremented on each interrupt (if the interrupt interval was set to 1000 in the call to `EMOPEN`, `KINT` will count in milliseconds). `KINT` is a single integer and will therefore overflow (go negative) if it reaches a value of

greater than 32767. If it is incremented every millisecond, then the longest period that can be timed is 32.767 sec. This should be adequate for most purposes, but the user must reset `KINT` before it reaches this value, because (due to interrupt overheads) this condition is not trapped as an error. The subroutine `CLWAIT` provides a tidy way of waiting for a specified time. It has two arguments: (1) time from which wait is to be calculated, and (2) duration of wait (in interrupts). The first argument will often be simply the current time (`KINT`), but can be any other time, allowing some processing to be included in the wait time; a typical example of this is an intertrial interval, where the duration of the trial itself is unknown but the time from the onset of one trial to the onset of the next is specified. In this case, save the value of `KINT` at the start of the trial; then the call to `CLWAIT` at the end of the trial will have this value as its first argument and the intertrial interval as its second argument. An attempt to wait for a time that has already passed will be trapped as an error (error S).

Two subroutines, `CLZERO` and `CLTIME`, are provided for users of BASIC. `CLZERO` has no arguments and has the effect of setting `KINT` to zero. `CLTIME` has one argument, which is returned as the current value of `KINT`.

The clock can be stopped and disconnected from interrupt by a call to `CLKOFF`.

Digital Sampling

Digital interrupts are enabled by a call to `DIGON` and disabled by a call to `DIGOFF`. While digital input is enabled, any input (e.g., a keypress) causes the value of the digital input word to be placed in `IRESP`. Digital input and output can also be accomplished without using interrupts, using routines `DIGIN` and `DIGOUT`. `DIGIN` has one argument, which is returned as the current value of the digital input word. `DIGOUT` has one argument, which is the value to be sent to the digital output word.

Picture Control

Once a picture has been defined and clock interrupts are running, the picture can be switched on by a call to `PICON` and switched off by a call to `PICOFF`. A picture can be brightened by switching it on more than once; switching it on twice doubles its refresh rate, switching it on three times triples its refresh rate, and so forth. It is important to note that a picture switched on *N* times counts as *N* pictures: The total number of pictures that can be displayed simultaneously is equal to the refresh interval (specified in the call to `EMOPEN`). Dimming can be achieved by, for example, switching on all pictures twice and then switching off the picture to be dimmed. When a picture is to be removed from the display, the user should be careful to switch it off the same number of times that it has been switched on. Switching a picture off more times than it has been

switched on is not regarded as an error (it has no effect), but switching a picture off fewer times than it has been switched on will leave one or more copies of it in the display.

Analogue Sampling

Once clock interrupts are running, analogue sampling can be enabled by a call to ADCON and disabled by a call to ADCOFF. While analogue input is enabled, the specified channels are sampled in groups at the specified interval (IADC), and values are stored consecutively in the user's array. For example, if the number of channels (NCHAN) was three and the starting channel (ICHAN) was two, elements 1, 2, and 3 of the user's array would contain the values obtained by sampling channels 2, 3, and 4, respectively. NSAMP holds the total number of samples taken, and ISAMP holds the most recent sample (in multichannel sampling, this will always be the sample for the highest channel number). Users of BASIC are provided with the subroutines ADCNT and ADSAMP to obtain these values; ADSAMP has one argument, which is returned as the current value of ISAMP, and ADCNT has one argument, which is returned as the current value of NSAMP. It is the responsibility of the user to check that there is enough array space to hold all of the samples.

Finishing Up

When the user program has finished with EMDISP, it must explicitly shut down the system with a call to EMCLOS; this ensures that all functions are disabled and that the interrupt vectors and priorities of the devices it has used are correctly restored. EMCLOS also checks that no timing errors have occurred on clock interrupts (see next section).

Errors

Errors are reported in a standard format: EMDISP NAME error A, where NAME is the name of the issuing routine and "A" is a single letter code that identifies the nature of the error. Error codes and their meanings are listed in Table 3. When an error is detected, the message is displayed on the console device and EMDISP is shut down in an orderly fashion: All interrupts are disabled, and EMCLOS is called to restore the system to its previous state.

One error condition is not reported immediately but is flagged internally and can be detected by a subroutine call (ERRCHK) from the user program. This error arises when a second interrupt occurs before the clock interrupt service routine (CLISR) has finished processing the previous interrupt. If this happens, control is passed straight back to the beginning of CLISR without a return to the main program, and an internal counter is incremented. Processing of interrupts will continue as normal, and the only effect will be a delay in the processing of the second interrupt corresponding to the mismatch between the time taken to finish processing the first interrupt and the time at which the second interrupt occurred (as long as interrupt servicing continues to take longer than the interrupt interval, processing of the user program will be suspended; some accuracy in timing will therefore be lost). Processing of any interrupt will always be complete, as CLISR is non-interruptible. The subroutine ERRCHK has one argument, which is the number of such errors detected since the last call to ERRCHK (ERRCHK sets the error count back to zero each time it is called). If the error count is nonzero when EMCLOS is called, an error will be reported in the usual way (error Z). It is recommended that user programs always call ERRCHK (and print the

Table 3
Error Codes

Code	Meaning	Issuing Routines
A	EMDISP not open	(All except EMOPEN)
B	EMDISP not closed	EMOPEN
C	Function already on	CLKON,DIGON,ADCON
D	Function already off	CLKOFF,DIGOFF,ADCOFF
E	wrong number of arguments	EMOPEN,PICDEF,PICON,PICOFF
F	illegal picture number	PICDEF,PICON,PICOFF
G	picture not defined	PICON
H	less than one point	PICDEF
J	illegal interrupt interval	EMOPEN
K	illegal refresh interval	EMOPEN
L	illegal sampling interval	EMOPEN
M	illegal number of channels	EMOPEN
N	illegal first channel	EMOPEN
P	illegal DAC channels	EMOPEN
R	illegal waiting time	CLWAIT
S	illegal finishing time	CLWAIT
Y	vector protect fail (clock)	EMOPEN
X	vector protect fail (dig)	EMOPEN
Z	ISR errors detected	EMCLOS

error count) immediately before the call to EMCLOS. If the error count is nonzero, the user should decrease the processing demands or increase the interrupt interval.

OPERATION OF THE PICTURE REFRESH SYSTEM

PICDEF has two functions: to copy the contents of the input array to the user's display array and to record the address of the picture in a table of picture addresses. PICDEF stores the number of xy pairs in the picture as the first word of the picture. Once interrupts are running, CLISR consults a list of the pictures currently being displayed. The display list is a circular buffer whose length is related to the refresh interval; a pointer to this list moves on to the next list element on each interrupt. CLISR inspects the element of the display list that corresponds to the current position of the pointer. If this list element is odd, there is no display on that interrupt; if it is even, it is interpreted as a picture address and is used to retrieve the picture contents from the display array. The picture can then be displayed and the pointer moved to the next position in the display list. Only one picture is displayed per interrupt: This ensures that the interval between refreshes of a picture is always correct, since the picture is always refreshed at the same point in the cycle of interrupts. This system also maximizes the accuracy of picture onsets: PICON inspects the current position of the display list pointer and tests the position two elements ahead of the pointer. If this position is free, PICON inserts the picture address; if this position is already occupied, PICON inspects the next position, and so forth, until a free position is found. This means that the picture will be displayed as soon as possible in the cycle. [The reason for looking two elements ahead of the pointer is that it is possible that an interrupt could occur between detection of a free position by PICON and the insertion of the picture address; if PICON looked only one element ahead, this (admittedly very unlikely) situation would result in the first display of the picture having to wait a whole cycle.] A formula for calculating the mean accuracy of picture onsets is given in the next section.

PERFORMANCE

Number of Pictures

The maximum number of pictures that can be displayed simultaneously is equal to the refresh interval. However, there is no theoretical limit to the number of pictures that can be defined; an arbitrary limit of 20 is specified for error trapping and should be adequate for most purposes.

Size of Pictures

The size of any given picture is limited by the interval between interrupts and the speed at which points can be

output by the interrupt service routine. On the PDP-11/03 (LSI/2), with an interrupt every millisecond, a practical maximum of 30 points per picture is recommended. Faster machines or slower interrupt rates will increase this figure, as will the use of a direct memory access (DMA) device (this option would require slight modification of the clock interrupt service routine). Errors resulting from a picture that is too large to be displayed within one interrupt interval are logged and can be inspected using the subroutine ERRCHK (see section on errors); this can be a useful diagnostic tool. If a stimulus requires more points than can comfortably be accommodated within a single picture, then it can be split into two or more parts, with each part being defined as a separate picture and the parts switched on and off "together" by successive calls to PICON and PICOFF (this will of course mean that different parts of the stimulus will be refreshed at different points in the cycle, but this process will be invisible).

Accuracy of Picture Onsets

Picture onset will be late by (on average) D interrupts where:

$$D = \frac{N! \cdot (R-K+1)! \cdot (R-N)}{(N-K+1)! \cdot R! \cdot (R-K+1)} \cdot K$$

for $K = 1$ to $N+1$, where R = refresh interval and N = number of pictures currently being displayed.

Refresh Rate

The minimum refresh interval is equal to the number of pictures to be displayed simultaneously. The maximum refresh interval is 20, because refresh rates slower than 50 Hz usually produce discernible flicker. Between these two extremes, any value can be chosen; fast rates give greater accuracy of picture offsets (not normally relevant, because perceived offset usually is about 20 msec after the last refresh), whereas slower rates give greater accuracy of picture onsets.

Sampling Rates

The maximum rate for analogue sampling is equal to the interrupt rate; the minimum rate for analogue sampling is unlimited, but an arbitrary maximum interval of 100 interrupts is specified for error-trapping purposes.

PORTABILITY

Processors

EMDISP will run without modification on any of the following PDP-11s: LSI/2, 11/23, 11/04, 11/34, 11/35, 11/40, 11/45, and all more recent processors. It will also run on 11/05, 11/10, 11/15, and 11/20 if the SOB instruction (not supported by these processors) is replaced with the appropriate decrement, test, and branch instructions; for example,

SOB R0, 1\$ should be replaced by DEC R0
 TST R0
 BGT 1\$

Neither EIS nor FIS processor options are required, because EMDISP does not use them.

Interfaces

EMDISP uses the CED 502 interface, with the universal clock (502-7), ADC (502-13 or 23), DAC (502-19), brightness control (on 502-8), and digital input and output (502-4, 5, 6) as minimal options. EMDISP can deal with as many as 8 analogue channels. Use of the DEC LPS-11 interface will require minimal modification, because it is mostly compatible with the CED 502.

Operating Systems

EMDISP will run under any RT-11 system (versions 2 to 5), under either FB or SJ monitors.

Languages

Any language that can run under RT-11 can use EMDISP. The languages most likely to be used are FORTRAN, BASIC, C, and MODULA. Subroutine linkage follows the RT-11 convention, in which argument addresses are stacked on R5. Users of BASIC, or of any other language in which it is not possible to define a shared data area, are provided with subroutine calls to perform the same functions.

MEMORY REQUIREMENTS

EMDISP occupies 2,242 words of memory. A smaller, faster version (with no error checking except that provided by ERRCHK), EMSMLL, occupies only 984 words. EMSMLL should be used only with programs

that have been fully debugged using EMDISP. All subroutine calls and functions are identical in both versions. Memory requirements of user programs depend on the application and the language used. Applications that require fast analogue sampling on several channels use up memory very quickly, but because EMDISP places no limit on the size of the analogue samples array, the only limit is the size of the machine.

AVAILABILITY

Copies of the programs, source code, and documentation are available in machine-readable form. These can be obtained from the author on receipt of a suitable disk and £5 (U.K.) or \$10 (U.S.) to cover reproduction and postal costs. Suitable disks are RX02 (soft-sectored) floppy disks (single- or double-sided, reversible, and formatted single- or double-density, as required). An early version of EMDISP for the CED Alpha (LSI 2/20G) computer is available on single-sided, single-density, hard-sectored disks. A variety of coordinate-generating routines are also available on request.

REFERENCES

- FINDLAY, J. M. (1974). A simple apparatus for recording micro-saccades during visual fixation. *Quarterly Journal of Experimental Psychology*, **26**, 167-170.
- FINDLAY, J. M. (1978). Estimates on probability functions: A more virulent PEST. *Perception & Psychophysics*, **23**, 181-185.
- MUELLER, H. J. (1983, January). *The effect of selective spatial attention on peripheral discrimination thresholds*. Paper presented at the meeting of the Experimental Psychology Society, London.
- SHEPHERD, M. (1984). *Attentional consequences of saccadic tracking*. Unpublished manuscript, University of Durham, Durham, England.

(Manuscript received December 22, 1983;
 revision accepted for publication March 13, 1984.)