Behavior Research Methods & Instrumentation
1980, Vol. 12 (6), 614-626

COMPUTER TECHNOLOGY

Visual psychophysics on the APPLE II: Getting started

PATRICK CAVANAGH
University of Montreal, Montreal, Quebec 101, Canada

and

STUART M. ANSTIS
York University, Downsview, Ontario M3J 1P3, Canada

Several routines are described for psychophysical procedures on the APPLE II. Method of adjustment and PEST routines are presented within short demonstration programs. Techniques for synchronized presentation of images in tachistoscope and motion displays are given, as well as methods for saving graphics images for later recall.

The APPLE is one of the better microcomputers for interactive psychophysics because of its graphics and games controls. Stimuli can be plotted in low resolution (40 by 48 squares) or high resolution (280 by 192 points), in color or black and white, either on a domestic TV set (cheaper) or on a video monitor (better picture quality). Durrett (1979) compares the color display abilities of the APPLE and its current competitors. The APPLE also comes with four analog to digital (A/D) converters, two of which are connected to games paddles (potentiometers). It can read the state of three pushbuttons or sense switches, and it can open and close four TTL-compatible outputs that can trigger various electronic devices. Methods to expand these I/O facilities are described by Carpenter (1979) and Thompson (1979).

This article offers some general-purpose programs and subroutines for interactive psychophysics, which the reader can adapt to his own purposes. Programs are given to draw two illusions in high-resolution graphics: the Mueller-Lyer arrows and the vertical-horizontal illusion. Each can be measured interactively, using either the method of adjustment or a random double staircase. Also, some subroutines are provided for putting up graphics in brief flashes synchronized to the TV frame rate. These are useful for tachistoscopic experiments and for simple moving displays. (Programs are written for ROM or language system APPLESOFT and are not compatible with RAM APPLESOFT memory allocations without modification.)

This research was supported by Grant A 8606 to P. Cavanagh and by Grant A 0260 to S. M. Anstis, both from the Natural Science and Engineering Research Council of Canada (NSERC). We are grateful to P. K. Kaiser for various helpful suggestions concerning psychophysical programs.

METHOD OF ADJUSTMENT

Program 1 measures the strength of the Mueller-Lyer arrow illusion as a function of fin angle, using the method of adjustment. The arrows are plotted end to end on the screen in high-resolution graphics. Using Paddle 1, the subject adjusts the position of the central fin until the two arrows look subjectively the same length. He presses a button, and the computer records his response and draws a new stimulus on the screen, with the central fin randomly offset and a new, randomly chosen fin angle. After 20 trials, the program plots a graph of the results on the screen. The results for most subjects are pleasingly linear, and this program has been successfully used in undergraduate laboratory classes.

The main loop is in Lines 100-250. Line 120 randomly sets the fins to be inclined at an angle of 10, 30, 50, or 70 deg to the left or right of the stalk, or 90 deg (perpendicular). The arrow stimuli are plotted in Lines 280-330. If the subject moves his paddle, Lines 190-250 detect hits, erase the old picture of the central fins (Lines 340-380), and replot them in their new position (Lines 280-330). When he is satisfied with his setting, the subject presses the pushbutton (Line 220). The setting Q(N) is stored, and a new stimulus is plotted (Lines 110-130). After 20 trials (Line 230), the arrows disappear from the screen and the subject's settings are plotted as a graph in Lines 390-570.

A word of warning: the length of all lines should be calibrated directly on the TV screen with a graticule or transparent ruler. The nonlinear picture geometry of a

0005-7878/80/060614-13$01.55/0

```
10  REM  ********************
20  REM      MULLER LYER
25  REM    ADJUSTMENT METHOD
30  REM  ********************
40  DIM A(100),Q(100),X(100),Y(100)
50 N = 0
60  TEXT : HOME : VTAB 22
70  PRINT " ADJUST PADDLE #1 UNTIL ARROWS LOOK THE SAME LENGTH. ";
80  PRINT "THEN PRESS PADDLE BUTTON."
90  HGR : HCOLOR= 7
100  REM   MAIN LOOP -----
110 N = N + 1
120 A(N) = 20 / 57.3 *  INT (9 *  RND (1) - 4): REM RANDOM FIN ANGLE -----
130 C = 20 *  RND (1) - 20 *  RND (1): REM  RANDOM OFFSET ---
140 R =  PDL (1)
150 Q(N) = R / 2 - 60 + C: REM   S'S SETTING -----
160 RR = 25: REM  FIN LENGTH -----
170 RC = RR *  COS (A(N)):RS = RR *  SIN (A(N))
180  GOSUB 280
190  REM    RE-PLOT ARROW IF SUBJECT ALTERS PADDLE ---
200  IF  ABS (R -  PDL (1)) <  = 2 THEN 210
205 F = 1: HCOLOR= 0: GOSUB 320: HCOLOR= 7: GOSUB 260
210 X =  PEEK ( - 16286)
220  IF F = 1 AND X > 127 THEN F = 0: GOSUB 340: GOTO 110
230  IF N > 10 THEN  GOTO 390: REM      END EXPERIMENT AFTER 10 TRIALS -----
240  IF X < 128 THEN  GOTO 200
250  GOTO 210
260 R =  PDL (1)
270 Q(N) = R / 2 - 60 + C: REM   S'S SETTING -----
280  REM  PLOT ARROWS -----
290  HPLOT 40,100 TO 240,100: REM  STALK -----
300  HPLOT 40 - RS,100 + RC TO 40,100 TO 40 - RS,100 - RC: REM   LEFT FIN ---
310  HPLOT 240 - RS,100 + RC TO 240,100 TO 240 - RS,100 - RC: REM   RIGHT FIN ---
315  REM  CENTRE FIN -----
320  HPLOT 140 + Q(N) + RS,100 + RC TO 140 + Q(N),100 TO 140 + Q(N) + RS,100 - RC
330  RETURN
340  REM   ERASE LAST PICTURE -----
350  CALL  - 1059: CALL  - 1059
360  HCOLOR= 0: GOSUB 300
370  HCOLOR= 7
380  RETURN
390  HGR : REM  PLOT GRAPH OF RESULTS -----
400  HCOLOR= 7
410  HPLOT 140,0 TO 140,160
420  HPLOT 60,80 TO 220,80
430  FOR K = 0 TO 160 STEP 20
440 XK = K + 60:YK = K
450  HPLOT XK,78 TO XK,82
460  HPLOT 138,YK TO 142,YK
470  NEXT K
480  HCOLOR= 5
490  FOR J = 1 TO N
500 X(J) = A(J) * 57.3 + 140
510 Y(J) = 4 * Q(J) + 80
520  IF Y(J) < 0 THEN Y(J) = 0
530  IF Y(J) > 159 THEN Y(J) = 159
540  HPLOT X(J),Y(J) TO X(J) + 1,Y(J) TO X(J) + 1,Y(J) + 1 TO X(J),Y(J) + 1
550  NEXT J
560  PRINT : PRINT "X=FIN ANGLE (ZERO=PERPENDICULAR)"
570  PRINT "Y=ILLUSION (ZERO=NO ILLUSION)"
580  END
```

Program 1. Method of adjustment for the Mueller-Lyer illusion.

domestic TV receiver has to be measured to be believed.

PEST WITH DOUBLE RANDOM STAIRCASE

The staircase method is a good compromise between the fast but inaccurate method of adjustment and the slow but sure method of constant stimuli. Program 2 measures the vertical-horizontal illusion; the subject reports via the keyboard whether a variable vertical line looks longer or shorter than a standard horizontal line. If it looks longer, then the program makes it shorter the next time. If it looks shorter, then the program makes it longer the next time. This interactive method makes the next stimulus depend upon the subject's history of previous responses. It is efficient because the negative feedback exerted by the last response upon the next stimulus will make the stimuli "home in" rapidly on the interesting threshold region, or point of subjective equality (PSE), that one wishes to measure.

The staircase method can be made still more efficient if the subject's previous responses determine the size, as well as the direction, of the next stimulus step. A large initial step size will make the stimulus step rapidly toward the region of the PSE. Each time the subject reverses his responses from "larger" to "smaller" or vice versa, the step size is halved. After a number of such reversals, the step size will fall to some predetermined criterion size, and the experiment is terminated. This is the principle of parameter estimation by sequential testing (PEST) (Taylor & Creelman, 1963).

The subject sometimes becomes too wise to his position on the staircase, and this can bias his responses. A good safeguard against this is the random double staircase (Cornsweet, 1962). Two staircases are run concurrently, one starting with a long variable line, the other with a short variable line. The sequence of trials switches at random times from one staircase to the other.

Program 2 contains a random double staircase with varying PEST step sizes. It measures the horizontal-vertical illusion. Two lines forming an inverted T are presented on the screen, and the subject reports whether the variable vertical line looks longer or shorter than the standard horizontal line by hitting "L" or "S" on the keyboard. If his response is "L" (or "S"), then the vertical line length is decreased (or increased) by Lines 240 and 250 and Lines 360 and 370. Staircases 1 and 2 are controlled by the variable P, which randomly switches between the values 1 and 2 in Lines 190 and 220. Line 280 compares the subject's latest "L" or "S" response, A$(P), with his previous response, OLDA$(P), on that staircase, and if they differ (i.e., if his response changed from "longer" to "shorter" or vice versa), the step size, D(P), is halved.

Most psychophysical procedures do not provide knowledge of results to the subject. For didactic purposes, this program prints out on the screen the trial number, step size, and staircase number (1 or 2) (Lines 430 and 440). When the step size D(P) falls to a criterion size, which is arbitrarily set to 1 in Lines 310 and 320, the experiment ends, and Lines 480-550 plot the two staircases in different colors on the screen. Lines 580-630 allow the subject to flip between the table of results and the graph of the staircases by hitting the space bar. This is useful for teaching students about the staircase method.

TACHISTOSCOPES AND MOVIES

Potential users of a microcomputer often ask whether it can simulate (1) a tachistoscope and (2) an animated movie. For the APPLE, the answers are (1) yes and (2) modestly.

For use as a tachistoscope, the APPLE has three separate graphics fields available. (Special methods described below can extend this to four or five fields.) If only three stimuli are needed, these can be plotted ahead of time and displayed in rapid sequence, up to a maximum rate of 1 picture/TV frame (16.6 msec). If more than three stimulus fields are needed, the cycling rate is limited by the time it takes to preplot an unseen field while another field is on display. For stimuli composed of a word or two of text, this replotting takes only a fraction of 1 sec: so, for example, one could expose a sequence of 1,000 words (read into memory from DATA statements) with an interstimulus interval (ISI) of only .1 sec or less between words. This would simulate an infinite-field tachistoscope. However, graphics take longer to plot. A single line or a few points can be plotted quickly, and sometimes small displays can be plotted rapidly in machine language (Anstis & Cavanagh, in press). But, as a general rule, plotting graphics is a slow business. As an extreme instance, a complete high-resolution field of random dots can take 5-10 min to plot. One handy expedient is to plot such pictures well in advance, SAVE them on disk, then LOAD them when you want to run an experiment. For even higher speeds, you may SAVE pictures in free areas of RAM memory and then recall them later (see below).

The APPLE has a limited capacity to display moving pictures. One approach is to construct a drawing using a shape table and, subsequently, to move the drawing around the screen in real-time using the DRAW and XDRAW commands. This can give quite convincing animation, and the shape table can be SAVEd to disk to be reused later. Note that the DRAWing time is a function of the total contour length, so bigger pictures will flicker perceptibly as they move. The method for building a shape table given in the APPLESOFT manual is laborious, to say the least. Figueras (1979) and Swenson (1979) have published labor-saving programs for building a set of up to 36 small shapes such as alphanumeric characters and a single large shape, respectively, in a single shape table.

For large displays that alternate between two positions, stimuli can be plotted slowly ahead of time on the

```
10  REM  ********************
20  REM          PEST
30  REM  ********************
40  TEXT : HOME : VTAB 10
50  PRINT "YOU WILL SEE A HORIZONTAL AND A VERTICAL LINE.  HIT S ON KEYBOARD ";
60  PRINT "IF THE VERTICAL LINE LOOKS SHORTER THAN THE HORIZONTAL. ";
70  PRINT "HIT L IF IT LOOKS LONGER. "
80  PRINT : PRINT "HIT SPACE BAR FOR FIRST TRIAL. "
90  GET N$: HOME : VTAB 22
110  REM  INITIALISE VARIABLES -----
120  DIM L(2): REM  VARIABLE LINE LENGTHS
130  DIM DD(2,150): REM   STORE D(P) INCREMENTS IN AN ARRAY
140  DIM D(2): REM   INCREMENTS IN LENGTH
150  DIM A$(2),OLDA$(2): REM  SUBJECT'S RESPONSES
160 D(1) = 5 + 10 *  RND (1):D(2) =  - 5 - 10 *  RND (1)
180 L(1) = 40:L(2) = 40: REM  STANDARD LINE LENGTH -----
190 P = 1: REM  CHOOSE WHICH STAIRCASE
200  GOSUB 340: REM  PLOTTING SUBROUTINE
210  REM  MAIN LOOP -----
220 R =  RND (1): IF R > 0.5 THEN P = 3 - P
230  GET A$(P)
240  IF A$(P) = "S" THEN D(P) =  ABS (D(P))
250  IF A$(P) = "L" THEN D(P) =  -  ABS (D(P))
260  IF A$(P) = "S" OR A$(P) = "L" THEN 270
265  PRINT "HIT S OR L, PLEASE": GOTO 200
270  REM : ON REVERSAL, INCREMENT IS HALVED -----
280  IF A$(P) <  > OLDA$(P) THEN D(P) =  INT (50 * D(P)) / 100
290 OLDA$(P) = A$(P)
300  GOSUB 400: CALL  - 1059: CALL  - 1059: GOSUB 340
310  IF  ABS (D(1)) < 1 AND  ABS (D(2)) < 1 THEN 460: REM END OF RUN ---
320  IF  ABS (D(P)) < 1 THEN P = 3 - P: GOTO 230: REM COMPLETE OTHER STAIRCASE ---
330  GOTO 220
340  REM  PLOTTING SUBROUTINE -----
350  HGR : HCOLOR= 7:L(P) = L(P) + D(P)
370  HPLOT 140,150 - 2 * L(P) TO 140,150: REM    VARIABLE LINE -----
380  HPLOT 80,150 TO 200,150: REM  STANDARD LINE -----
390  RETURN
400  REM  COUNT RESPONSES -----
410 N = N + 1:DD(P,N) = D(P)
430  PRINT "TRIAL # ";N;":INCR=";D(P);
440  PRINT  TAB( 25);A$(P);"   STAIR #";P
450  RETURN
460  REM   PLOT GRAPH OF RESULTS ----------
470  HGR
480  FOR P = 1 TO 2
490  HCOLOR= 4 + P
500  FOR Q = 1 TO N
510 HH = 10 * Q: IF HH > 279 THEN HH = 279
520 VV = 80 + 5 * DD(P,Q)
530  IF Q = 1 THEN  HPLOT HH,VV: GOTO 550
540  HPLOT  TO HH,VV
550  NEXT Q,P
560  HCOLOR= 7: HPLOT 0,80 TO 10,80: HPLOT 230,80 TO 279,80
570 K = 0
580  PRINT : PRINT "HIT SPACE BAR TO SWITCH BETWEEN GRAPH AND TABLE. "
590  PRINT "HIT RESET TO EXIT FROM PROGRAM. "
600  GET SW$: POKE  - 16303 - K,0
620 K = 1 - K: GOTO 600
640  END
```

Program 2. PEST procedure for vertical-horizontal illusion.

two memory pages of high-resolution graphics, known as HGR and HGR2. Getting the most out of the graphics capabilities requires a thorough knowledge of the graphics soft switches in the APPLESOFT reference manual. This knowledge can be obtained only by study and practice. For example, the command POKE -16300,0 displays Page 1 of graphics, and POKE -16299,0 displays Page 2. To display a slow sequence of more than two pictures, it is possible to display Page 1 and leave it on the screen while plotting (but not displaying) Page 2

with the command POKE 230,64. (Following this command all HPLOTs, DRAWs, and XDRAWs will be plotting on HIRES Page 2.) Then display Page 2, and, meanwhile, plot to Page 1 without displaying it with the command POKE 230,32. You must initialize the graphics, either with HGR and HGR2 or with the appropriate screen switches, before using these POKEs.

Program 3 uses the two high-resolution pages to demonstrate phi movement between a red square and a green triangle (Kolers, 1972). To alter the timing (ISI), adjust Paddle 0. To alter the spacing, adjust Paddle 1 and hold down the pushbutton on Paddle 1 until the display changes.

SYNCHRONIZING TO THE TV FRAME RATE

Rapid alternation between two graphics pictures will shred and tear the TV picture because the alternation is not synchronized to the TV frame rate. The same problem arises if text is flashed up tachistoscopically in Program 4.

As the sentence is flashed up repetitively, it will often be randomly fragmented, with a word or a few letters missing. Such problems can be cured by a small hardware modification (Reed, 1979), which can be used by a variety of control subroutines, three of which are described in the following sections.

On the APPLE, the vertical (frame) TV synch pulse is available at Pin 8 of the integrated circuit (IC) socket B11. This socket is almost directly below the "*" key of the keyboard, and Pin 8 is at the top left (northwest) corner, looking down at the socket from the keyboard. After verifying that the APPLE is turned off, connect this pin with a 22-gauge insulated wire to Pin 4 of Socket J14, which is the game I/O socket. To avoid pick-up, keep the wire away from the RF modulator. It is best to wire wrap the connecting wire to the appropriate pins of two wire-wrap sockets with top-mounted posts, if available. As a temporary expedient you can wrap a loop of wire around the appropriate IC pin without soldering and then reinsert the IC and games connector. This modification makes the TV synch pulse available to the digital sense switch SW3, which can be interrogated from BASIC by the command PEEK (-16285). The other two sense switches are devoted to the pushbuttons on Paddles 0 and 1 (for further details, see Reed, 1979).

With this hardware modification installed, the following three subroutines can be used to present synchronized sequences of TV images. The first subroutine (Program 5) constrains the changeover from Picture A to Picture B to coincide with the blanking interval between TV frames. The second (Program 6) does the same thing in rapid machine language. The third (Program 7) changes from Picture A to Picture B, holds Picture B for a preset number of TV frames, and then returns to

```
50  TEXT : HOME : VTAB 10
60  PRINT "APPARENT MOVEMENT (KOLERS, 1972)": PRINT
70  PRINT "TO VARY  TIMING,  TURN PADDLE 0. "
80  PRINT "TO VARY SPACING,  TURN PADDLE 1,  THEN HOLD DOWN"
81  PRINT "BUTTON ON PADDLE 1 UNTIL DISPLAY CHANGES"
82  INVERSE
85  PRINT : PRINT : PRINT "HIT ANY KEY TO CONTINUE. . . "
88  NORMAL
90  GET K$
100  HGR : HCOLOR= 1:K =  -  PDL (1) / 2.5
110  GOSUB 200: REM  PLOT TRIANGLE
120  HGR2 : HCOLOR= 5:K =  - K
130  GOSUB 300: REM  PLOT SQUARE
140 T1 = 2 *  PDL (0):T2 = 2 *  PDL (0): REM TIMING INTERVALS
150  GOSUB 400: REM  SHOW DISPLAYS
160  IF  PEEK ( - 16286) > 127 THEN  GOTO 100
170  GOTO 140
200  REM  PLOT TRIANGLE **********************
210  FOR J = 0 TO 20
220  HPLOT 140 + K - J,J + 70 TO 140 + K + J,J + 70
230  NEXT J: RETURN
300  REM  PLOT SQUARE ****************
310  FOR J = 130 TO 155
320  HPLOT J + K,70 TO J + K,90
330  NEXT J: RETURN
400  REM  PRESENT DISPLAYS ****************
410  POKE  - 16299,0: REM  DISPLAY PAGE 2
420  FOR J = 0 TO T1: NEXT J
430  POKE  - 16300,0: REM  DISPLAY PAGE 1
440  FOR J = 0 TO T2: NEXT J
450  RETURN
```

Program 3. Apparent motion.

```
100  HOME : GR : TEXT : VTAB 10: HTAB 10: INVERSE
110  PRINT "CAN YOU READ THIS?": NORMAL
120  HGR : REM  BLANK FIELD
130 T1 =  PDL (0):T2 = 300: REM  TIMING INTERVALS
140  GOSUB 400: REM  SHOW DISPLAY
150  GOTO 130
400  REM  PRESENT DISPLAYS **********
410  POKE  - 16303,0: REM  SHOW TEXT
420  FOR J = 0 TO T1: NEXT J
430  POKE  - 16304,0: REM  SHOW BLANK FIELD IN HIRES
440  FOR J = 0 TO T2: NEXT J
450  RETURN
```

Program 4. Tachistoscope for text.

```
400  REM  MODIFIED DISPLAY ROUTINE  ****
405  REM  FOR PROGS #3 & #4         ****
410 SWITCH =  - 16303: GOSUB 6000: REM  SHOW TEXT
420  FOR J = 0 TO T1: NEXT J
430 SWITCH =  - 16304: GOSUB 6000: REM  SHOW BLANK FIELD
440  FOR J = 0 TO T2: NEXT J
450  RETURN
6000  REM  BASIC SOFT GRAPHICS SYCHRONISED SWITCH *******
6010  WAIT  - 16285,128,128: REM  WAIT FOR PICTURE PORTION
6020  WAIT  - 16285,128: REM  THEN WAIT FOR SYNCH
6030  FOR J = 0 TO 7: NEXT J: REM  WAIT UNTIL NEXT SYNCH PULSE
6040  POKE SWITCH,0: REM  THEN SWITCH
6050  RETURN
```

Program 5. Synchronizing BASIC soft switches for graphics screens.

a

```
50  GOSUB 6000: REM  STORE SUBROUTINE IN MEMORY
400  REM  MODIFIED DISPLAY ROUTINE  ****
405  REM  FOR PROGS #3 & #4         ****
410  POKE 813,85: CALL 800: REM  PAGE 2
420  FOR J = 0 TO T1: NEXT J
430  POKE 813,84: CALL 800: REM  PAGE 1
440  FOR J = 0 TO T2: NEXT J
450  RETURN
6000  REM MACHINE LANGUAGE SYNCHRONISED GRAPHICS SWITCH
6010  DATA  169,127,205,99,192
6020  DATA  48,251,205,99,192
6030  DATA  16,251,141,0,192,96
6040  FOR I = 800 TO 815
6050  READ V: POKE I,V: NEXT I: RETURN
```

b

```
0320-  A9 7F     LDA  #$7F
0322-  CD 63 C0  CMP  $C063
0325-  30 FB     BMI  $0322
0327-  CD 63 C0  CMP  $C063
032A-  10 FB     BPL  $0327
032C-  8D 00 C0  STA  $C000
032F-  60        RTS
```

Program 6. (a) Synchronized machine language graphics switches. (b) Machine code representation of subroutine.

Picture A. For demonstration purposes, these subroutines replace the subroutine in Lines 400-450 of Program 4 or, with appropriate changes of switch values, the subroutine in Lines 400-450 of Program 3.

SYNCHRONIZING THE SOFTWARE GRAPHICS SWITCHES USING BASIC

The blanking interval between two picture frames is about 4 msec, much too short for BASIC to sense the synch pulse and POKE a graphics switch. It is possible, however, to synchronize BASIC to the onset of one synch pulse and then time out an appropriate interval, so that the occurrence of the POKE to the graphics switch coincides with a later synch pulse. Program 5 demonstrates this technique using BASIC WAIT statements. The first WAIT (Line 6000) waits until the video signal is in the picture portion (Bit 7 low; see the APPLESOFT manual for a description of WAIT); the second WAIT (Line 6010) waits for the onset of the synch pulse. (Note that if we had only used the second wait we would not have been assured of being synchronized to the onset of the synch pulse but might have found ourselves timed to any point within the 4-msec blanking interval.) If we were to POKE a graphics switch immediately after dropping out of the second WAIT, we would find the display page changing in the middle of the screen. A short delay (Line 6030) solves this problem by situating the POKE within the next blanking interval.

To demonstrate the subroutine, delete Lines 400-450 of Program 4 and add all the statements of Program 5. To demonstrate it using the graphics example of Program 3, delete Lines 400-450 of Program 3, add all the statements of Program 5, and change switch values in

Lines 410 and 430 to -16299 (display Page 2) and -16300 (display Page 1), respectively.

Only one software switch can be synchronized in this manner, as a second POKE to a graphics switch will not trigger the next change before the effects of the first POKE are seen. We are therefore limited to displays that can be alternated using a single switch: Page 1 to Page 2 (-16300, -16299), text to Page 1 high- or low-resolution graphics (-16304, -16303), and Page 1 low- to Page 1 high-resolution graphics (-16298, -16297). Any displays requiring two or more synchronized switches within a single blanking interval must use the short machine language routines described below.[1]

The maximum alternation rate using the BASIC statements is 20 pictures/sec (a minimum of three TV frames between successive switches). Again, for higher rates of switching, machine language subroutines must be used.

SYNCHRONIZING USING A MACHINE LANGUAGE SUBROUTINE

The machine language subroutine held in the DATA Statements 6010-6030 of Program 6a performs the same sequence of steps as does the BASIC subroutine just described, only much more rapidly: Wait for the picture portion, wait for the synch pulse, then toggle a graphics switch. To use this subroutine, three steps are essential: (1) As an initialization step before the main program begins, the subroutine must be POKEd into memory (Line 50 of Program 6a). (2) The switch to be toggled must be specified before the subroutine is called, and the methods for doing this are described below. (3) The subroutine is CALLed at the location at which it was POKEd into memory. If it was POKEd starting at Location 800, then a CALL 800 will wait for the next blanking interval, set the specified switch, and return to the BASIC program.

In Program 6a, the machine language routine is inserted into memory by Lines 6040-6050, starting at Memory Address 800. In general, Locations 768-975 are free for small machine language programs, unless the communications interface is installed (see the communications interface manual). The subroutine here can be POKEd to any location without necessitating a change in the program itself. A listing of the assembly language steps of this subroutine is given in Program 6b. Listings for other machine language routines can be obtained using the L command of the APPLE monitor once the subroutines have been POKEd to memory.

Once the program is installed, the graphics switch to be toggled must be specified. In BASIC, the switch addresses are from -16304 to -16297, but for machine language, the high- and low-order portion of the addresses must be specified separately. The decimal value of the high-order portion of all the special I/O switches is 192, and the low-order equivalents are given in Table 1.

Table 1
Software Switches for Graphics and Their Machine Language Equivalents (MLE)

Command	Soft Switches	MLE
(1) Graphics display, HI or LORES,	POKE-16304,0	80
or text display	POKE-16303,0	81
(2) Full-screen graphics	POKE-16302,0	82
or mixed graphics and text	POKE-16301-9	83
(3) Page 1 of text or graphics	POKE-16300,0	84
or Page 2 of text or graphics	POKE-16299,0	85
(4) LORES graphics	POKE-16298,0	86
or HIRES graphics	POKE-16297,0	87

In Program 6a, Line 410, the POKE 813,81 specifies that the switch to be toggled in the machine language program is the text mode switch equivalent to POKE -16303,0 (-16303 is memory address hex C051, and the decimal equivalent of the low-order portion, hex 51, is 81).

Changing only the low-order switch values in the subroutine gives access to all the various I/O and graphics switches from the same subroutine. Depending on the need for synchronized switching in the main program, the switch specification can be accomplished in a number of ways:

(1) If only a single switch needs to be synchronized, then the low-order value for that switch can be included in the original DATA statements (e.g., replacing the 0 in Line 6030 with 80). The switch is then toggled whenever a CALL to the start address is made (a CALL 800 will set graphics mode if 80 has replaced 0 in Line 6030).

(2) If more than one switch is to be synchronized but high speed is not essential, the appropriate low-order switch value can be POKEd to memory just preceding each CALL. The switch value must be POKEd at the start address +13, thus 813 in our example. A POKE 813,85:CALL 800 would then switch to Page 2, whereas POKE 813,84:CALL 800 would switch to Page 1.

(3) If the highest speed is required, then POKE the subroutine into memory once for each switch to be used and POKE the appropriate switch values into each replicate of the subroutine. For extra speed, the subroutine addresses can be represented by variables, saving the interpreter the task of decoding the decimal address values at each CALL. Image switching at the full TV frame rate of 60 images/sec is obtainable.

The switch values are not altered when the subroutine is called, so switches must only be respecified when a new switch is to be toggled. If multiple copies of the subroutine are placed in memory, make sure that they do not overlap.

Finally, if more than two switches must be toggled within the same blanking interval (e.g., to change from HIRES Page 2 to LORES Page 1), modifications are necessary within the machine language subroutine (these are outlined in Footnote 1).

To demonstrate the subroutine of Program 6, delete

Lines 400-450 of Program 4 and add all the statements of Program 6. Note that Line 50 is necessary before the main program in order to get the machine language subroutine POKEd into memory before it is CALLed. To demonstrate the subroutine using the graphics example of Program 3, add all the statements of Program 5 and change switch equivalents in Lines 410 and 430 to POKE 813,85 (show Page 2) and POKE 813,84 (show Page 1), respectively.

PRESENTING AN IMAGE FOR A PRESET NUMBER OF TV FRAMES

A machine language subroutine can be used with the hardware modification to present a stimulus for any desired number of television frames. The subroutine is in Lines 6010-6040. It is loaded at the beginning of the program, in Line 50, and called as often as required, as shown in Line 410. The number of frames to be presented is specified by POKE 801,N, where N is the presentation duration of frames. If the program always presents the display for the same number of frames, this value can be included in the original DATA statements, replacing the zero in Line 6010, Program 7.

The machine language program first conditions the graphics switch to move to the desired field (text, in this case), waits for the specified number of frames, and then conditions the next switch to return to the postfield (HIRES graphics). The first switch can be modified by POKEing in Location 815 and the second by POKEing 821 with the values shown in Table 1. The subroutine is invoked by a CALL 800. Again, if the routine is always to switch to and from the same fields, the switch specifications can be made in the original DATA statements, as is the case here. Remember that the switches toggled must be meaningful in the context of the display mode prior to the CALL 800. To demonstrate the subroutine of Program 7, delete Lines 400-450 of Program 4 and add all the statements of Program 7. The presence of Line 50 assures that the subroutine is stored before the main program is started. To demonstrate the subroutine using the graphics example of Program 3, delete lines 400-450 of Program 3 and add all the statements of Program 7 and change the switch equivalents that are now located in the DATA Statements 6030 and 6040. That is, the 81 of Line 6030 should be 85 (show Page 2), and the 80 of Line 6040 becomes 84 (show Page 1).

A series of N-frame presentations may be chained in order to simulate a multifield tachistoscope. The stimuli must be drawn on the various display pages prior to executing the chain.[2]

SAVING PICTURES TO DISK

When complex graphics are involved, it is often useful to SAVE the picture and reuse it later rather than to redraw it each time it is needed. Two options are available: (1) SAVEing to disk and (2) transferring to a free RAM memory area. SAVEing to disk is described in the DOS 3.2 manual. For example, to save Page 1, HIRES, under the name PIX2, type:

```
BSAVE PIX2,A$2000,L$2000
```

To SAVE Page 2, HIRES, under the name PIX2, type:

```
BSAVE PIX2, A$4000,L$2000
```

Reloading these pictures in their original pages is done by BLOAD PIX1 or BLOAD PIX2. The picture SAVEd from Page 1 can be loaded into Page 2 by BLOAD PIX1,A$4000, or that from Page 2 can be loaded to Page 1 by BLOAD PIX2,A$2000. All of these commands can be inserted into BASIC statements as described in the DOS 3.2 manual by composing a PRINT statement with a CHR$(4) code and the command in quotes, as follows:

```
PRINT CHR$(4); "BLOAD PIX1"
```

Reloading a picture from disk takes approximately 9.5 sec.

SAVING PICTURES TO MEMORY

In many cases, the time required for reloading an image from disk is far too long. It is quite easy, however, to save a graphics image in an unused portion of memory and then read it back into the display page when desired. In a 48K machine, there is enough space to store more than 30 low-resolution images (depending on program size) or one extra high-resolution image. Read-in time is approximately 25 and 200 msec for the low- and high-resolution images, respectively. The access time for reloading LORES pictures is sufficiently fast that the APPLE can be considered a 30-field tachistoscope for low-resolution images. The 30 available images can be read into the two low-resolution display pages in an alternating fashion to produce smoothly synchronized motion displays at a rate of 30 images/sec (Program 10).

Rowe and Grossman (1980) have presented a program for memory storage of graphics pictures, and Lines 7000-7020 of Program 8 give a slightly shorter version of their subroutine. The memory storage program moves four memory pages at a time. Each memory page contains 256 bytes, but the user should not confuse memory pages with the graphic display pages. The size of the low-resolution display page is 1,024 bytes, or four memory pages. Before calling the routine, the locations of the first page of the source and destination areas must be specified by POKEing in their values to locations 253 and 255. The steps for using the routine are then as follows: (1) POKE the routine into memory, (2) POKE the source and destination starting pages, respectively, and (3) CALL the program.

Page 1 of low-resolution graphics starts at Memory

```
50  GOSUB 6000: REM  STORE SUBROUTINE IN MEMORY
400  REM  DISPLAY ROUTINE FOR N-FRAME    ****
405  REM  PRESENTATION IN PROGS #3 & #4  ****
410  POKE 801,T1: CALL 800: REM T1 TEXT FRAMES THEN BLANK
420  FOR J = 0 TO T2: NEXT J
430  RETURN
6000  REM  MACHINE LANGUAGE N-FRAME PRESENTATION ROUTINE
6010  DATA  162,0,169,127,205,99
6020  DATA  192,48,251,205,99,192
6030  DATA  16,251,141,81,192,202
6040  DATA  16,238,141,80,192,96
6050  FOR I = 800 TO 823
6060  READ V: POKE I,V: NEXT I: RETURN
```

Program 7. Synchronized N-frame presentation.

```
7000  REM  MEMORY TRANSFER ROUTINE **********
7010  DATA  162,4,160,0,177,252,145
7020  DATA  254,200,208,249,230,253
7030  DATA  230,255,202,208,242,96
7040  FOR I = 860 TO 878
7050  READ V: POKE I,V: NEXT I
7060  POKE 252,0: POKE 254,0: REM  ZERO LOW ORDER ADDRESSES
7070  RETURN
```

Program 8. Transfer of a low-resolution graphics image to or from memory.

```
8000  REM  PAGE 2 AND 3 HIRES EXCHANGE **********
8010  REM  USES LOCATIONS 251 TO 255 IN MEMORY PAGE ZERO
8020  DATA  169,64,133,253,169,96
8030  DATA  133,255,162,32,160,0
8040  DATA  177,252,133,251,177,254
8050  DATA  145,252,165,251,145,254
8060  DATA  200,208,241,230,253,230
8070  DATA  255,202,208,234,96
8080  FOR I = 840 TO 874
8090  READ V: POKE I,V: NEXT I
8100  POKE 252,0: POKE 254,0: REM  ZERO LOW ORDER ADDRESSES
8110  RETURN
```

Program 9. Exchanging Pages 2 and 3 of high-resolution graphics.

Page 4, and the images to be stored can be sent anywhere from Memory Page 12 to Memory Page 146. The user should not overwrite the BASIC program and storage areas, which start at Memory Page 8, or the high-resolution pages (Memory Pages 32-63 and 64-95 for Pages 1 and 2 of HIRES, respectively), if they are being used. The safest area for a 48K machine starts at Memory Page 96. LOMEM and HIMEM can be set to avoid unexpected overlapping of BASIC variables and the graphics image storage.

For example, assume that an image has been drawn on low-resolution display Page 1. Once GOSUB 7000 is executed,

```
POKE 253,4: REM SPECIFY PAGE 1 LORES AS SOURCE

POKE 255,96: REM SPECIFY PAGE 96 AS DESTINATION

CALL 860
```

will store that image starting at Memory Page 96, just above HIRES Page 2. To recall the image,

```
POKE 253,96: REM SOURCE

POKE 255,4: REM PAGE 1 LORES IS DESTINATION

CALL 860
```

The source and destination values are altered by the subroutine each time it is executed; they end up pointing at the start of the next memory page following the four that were transferred. If the same memory pages are to be transferred each time, then the source and destination addresses must be respecified prior to each CALL. If a series of images are to be stored in sequential locations in memory, however, only the source need be respecified each time (e.g., POKE 253,4: REM SAVE LORES PAGE 1: CALL 860). The destination address, once set initially, will automatically point to the appropriate adjacent locations for each successive transfer. Similarly, when reading a series of images from memory to LORES page, the user need respecify only the destination; the source address, after being initially

```
5  TEXT : HOME : VTAB 10: PRINT "BEFORE RUNNING THIS PROGRAM": PRINT
7  PRINT : PRINT "ENSURE THAT A WIRE CONNECTS PIN 8 OF"
8  PRINT : PRINT "IC B.11 TO PIN 4 OF GAMES I/O SOCKET"
10  PRINT : PRINT "RESET: POKE 104,12:POKE 3072,0: RUN PSYCHO10"
15  PRINT : PRINT
20  PRINT "HIT ANY KEY TO CONTINUE": GET KEY$
30  REM  DEFINE MEMORY ADDRESSES AS VARIABLES TO SAVE EXECUTION TIME
40 SOURCE = 253:DEST = 255:TRANSFER = 860
50 FRSTPAGE = 800:SECNDPAGE = 820
60 P1 = 4:P2 = 8: REM  MEMORY PAGE ADDRESS FOR LORES1 AND LORES2
70  GR : HOME : GOSUB 6010: GOSUB 300: REM  INITIALIZE
80  REM  READ IMAGES TO ONE PAGE WHILE DISPLAYING THE OTHER
90  POKE SOURCE,20: REM  PAGE ADDRESS OF FIRST IMAGE TO BE READ IN
100  FOR I = 1 TO 16
110  POKE DEST,P1: CALL TRANSFER: CALL FRSTPAGE
120  POKE DEST,P2: CALL TRANSFER: CALL SECNDPAGE
130  NEXT I
140  GOTO 90: REM  SHOW IT AGAIN
150  END
300  REM  DRAW 64 PICTURES OF MOVING COLOURED SQUARES
310  REM  BUT ONLY SAVE LAST 32 AS MOVIE
320  POKE DEST,20
330 A = 2:B = 2:X = 8:Y = 18
340  FOR N = 0 TO 63
350  IF X > 35 OR X < 5 THEN A = A *  - 1
360  IF Y > 35 OR Y < 5 THEN B = B *  - 1
370 X = X + A:Y = Y + B
380  FOR J = X - 3 TO X + 3
390  COLOR= 15: HLIN 37 - Y,43 - Y AT 40 - J
400  COLOR= 11: HLIN Y - 3,Y + 3 AT J
410  COLOR= 12: VLIN 37 - Y,43 - Y AT J
420  COLOR= 2: VLIN Y - 3,Y + 3 AT 40 - J
430  NEXT J
440  IF N > 31 THEN  POKE SOURCE,P1: CALL TRANSFER
450  NEXT N
460  RETURN
6000  REM  MACHINE LANGUAGE SOFT GRAPHICS SYNCHRONISED SWITCH ********
6010  DATA  169,127,205,99,192
6020  DATA  48,251,205,99,192
6030  DATA  16,251,141,0,192,96
6040  REM  MAKE TWO COPIES OF THE SUBROUTINE, ONE TO
6050  REM  DISPLAY PAGE 1, THE OTHER TO DISPLAY PAGE 2
6060  FOR I = 800 TO 815: READ V
6070  POKE I,V: POKE I + 20,V: NEXT I
6080  POKE 813,84: POKE 833,85
7000  REM  MEMORY TRANSFER ROUTINE ******************
7010  DATA  162,4,160,0,177,252,145
7020  DATA  254,200,208,249,230,253
7030  DATA  230,255,202,208,242,96
7040  FOR I = 860 TO 878
7050  READ V: POKE I,V: NEXT I
7060  POKE 252,0: POKE 254,0: REM  ZERO LOW ORDER ADDRESSES
7070  RETURN
```

Program 10. Thirty-two image motion sequence.

set, will automatically point to the appropriate memory pages.

For high-resolution pages, there is only sufficient space in memory for a single additional image to be stored. Three HIRES pictures can be maintained without redrawing if reading the stored image back into HIRES Page 1 or 2 does not destroy the image already there. Program 9 is therefore an exchange routine that moves the picture in HIRES Page 2 into Memory Pages 96-127 and at the same time moves the content of Memory Pages 96-127 into HIRES Page 2. No parameters need to be set here. Once the GOSUB 8000 has been executed, a CALL 840 effects the exchange. This can be done, for example, while another display page is being presented. Switching back to HIRES Page 2 reveals the new picture.

A picture can be drawn in HIRES Page 2 and then moved to memory with a CALL 840, and a second

picture can then be drawn in HIRES Page 2. Alternatively, a picture can be drawn directly to Memory Pages 96-127, just as if it were a regular graphics page following a POKE 230,96. All HPLOTs and other HIRES graphics commands then plot to the third picture page. This page cannot be displayed until it is exchanged with HIRES Page 2. The exchange time is about 200 msec.

ADDITIONAL DISPLAY FIELDS: BLANK AND LORES PAGE 2

If two HIRES and one LORES displays are not sufficient for a particular paradigm, there are two additional options available. First (as described by Reed, 1979), a slight wiring modification permits the entire screen to be blanked when Annunciator 3 is set. Setting and resetting the annunciator can then be included along with the graphics switches in Programs 5-7 to blank and then to reveal the current display page in synchrony with the TV frame rate (low-order equivalent values for the machine language routines in Programs 6 and 7 are 94 for set, 95 for clear).

A second option, which poses some difficulties for disk-based APPLESOFT users, is the second page of the low-resolution display. It is difficult to access because it is used to store the current BASIC program in tokenized form. In order to clear the area, the pointers that indicate the start location of the program space must be modified, and the 1st byte of the program area must be set to zero. The memory page address of the program space is given in Location 104, and this must be set before loading or writing the program. POKE 104,12, for example, will move the start of the program area to just above Page 2 of low resolution (located in Memory Pages 8-11). The 1st byte of the program area must then be set to zero; for Memory Page 2, the first location is at 12 * 256, or 3072, and, therefore, POKE 3072,0. The program area can be moved anywhere in free RAM memory, but study the memory allocation diagrams in the APPLESOFT and DOS manuals to avoid surprises.

Once the LORES Page 2 is free, images can be moved to it using the memory transfer program (Program 8), and the two LORES images can then be used to present synchronized motion sequences as shown in Program 10. Thirty-two images are stored in memory and read in at a rate of 30 images/sec. One page is displayed while the other is being read in and the display page is then switched during each second blanking interval.

Following any program using the LORES Page 2 display, the program area should be reset to Memory Page 8, its usual position, with a POKE 104,8: POKE 2048,0. It is quite possible that some programs that normally run without problems will not run starting at Memory Page 12, as program areas and HIRES graphics may overlap.

INTEGER BASIC AND PASCAL

All of our programs have been presented in the APPLESOFT (floating-point BASIC) language because we feel it is the most appropriate language for the short, graphics-intensive programs typically used in psychophysical experiments. Integer BASIC, although more rapid in execution, is limited by its more cumbersome set of HIRES graphics routines. PASCAL allows sophisticated programming and rapid execution. It is fast enough to permit the drawing in real-time of sequences of small displays, including motion. However, PASCAL directly accesses only a single graphics screen, and this severely restricts the presentation of sequences of complex stimuli. It is possible to reserve memory areas with subroutines in UCSD assembler language and then to use these areas for storing and recalling images with transfer routines such as that in Program 8. This would be worth doing only for larger programs, in which PASCAL's other advantages over BASIC become substantial.

OTHER VISUAL DISPLAYS

The APPLE can draw pictures made up of straight lines or of little colored blocks. For more elaborate displays, it is often better to stay away from the APPLE graphics altogether and to use the computer as a glorified timer, to control electronic function generators, and so on. This can be done by modifying the existing I/O facilities of paddles, pushbuttons, and sense switches that come with the APPLE (Carpenter, 1979; Thompson, 1979), or else by purchasing A/D and D/A converters and digital I/O boards obtainable from Interactive Structures (P.O. Box 404, Bala Cynwyd, Pennsylvania 19004). A/D input devices are also sold by California Computer Systems (250 Caribbean, Sunnyvale, California 94086), Connecticut Microcomputer, Inc. (150 Pocono Road, Brookfield, Connecticut 06804), and Mountain Hardware (300 Harvey West Boulevard, Santa Cruz, California 95060), who sells a combined A/D and D/A board. For a list of APPLE accessories and their manufacturers, see Schmeltz (1980). For instance, to plot a human modulation transfer function, one can set up a grating on an oscilloscope using conventional techniques (Campbell & Green, 1965): Set the CRO's internal time base at about 1 kHz to the vertical (Y) input, to set up a raster of very fine vertical lines. Then feed a sine wave of about 5-10 kHz to the luminance (Z) input and also to the time base trigger input. This produces a stationary grating of vertical bars. To get the grating under computer control, generate output voltages from the D/A converter and feed them to different inputs on the Z generator: the gating input to turn the grating on and off, the VCF input (sometimes called the FM input) to vary the spatial frequency of the grating, and the AM input to

vary its contrast. The voltages need to vary only slowly, so the APPLE can generate these and still run the rest of the experiment. It is wasteful to tie up the computer in generating high-frequency sine waves, because it will then be too busy to do anything else.

REFERENCES

ANSTIS, S. M., & CAVANAGH, P. What goes up need not come down: Moving flicker edges give positive motion aftereffects. *Attention and performance* (Vol. 9), in press.

CAMPBELL, F. W., & GREEN, D. Optical and retinal factors affecting visual resolution. *Journal of Physiology* (London), **1965, 181**, 576-593.

CARPENTER, C. APPLE II: Easy I/O sensing and control. *Recreational Computing*, 1979, **7**, 4.

CORNSWEET, T. N. The staircase method in psychophysics. *American Journal of Psychology*, **1962, 75**, 481-485.

DURRETT, H. J. Color display systems: The state of the art. *Behavior Research Methods & Instrumentation*, 1979, **11**, 127-130.

FIGUERAS, J. How to do a shape table easily and correctly. *Micro*, December 1979, **19**, 11-22.

KOLERS, P. A. *Aspects of motion perception*. New York: Plenum, 1972.

REED, A. V. Microcomputer display timing: Problems and solutions. *Behavior Research Methods & Instrumentation*, 1979, **11**, 572-576.

ROWE, J., & GROSSMAN, C. Multiple page graphics for the Apple II. *Microcomputing*, March 1980, pp. 66-67.

SCHMELTZ, L. R. "Core" and more for your Apple. *Kilobaud Microcomputing*, January 1980, pp. 110-114.

SWENSON, C. Building a hi-res shape table for the Apple II. *Recreational Computing*, 1979, **7**, 26-28.

TAYLOR, M. M., & CREELMAN, C. D. PEST: Efficient estimates on probability functions. *Journal of the Acoustical Society of America*, 1963, **41**, 782-787.

THOMPSON, G. C. Behavioral programming with the APPLE II microcomputer. *Behavior Research Methods & Instrumentation*, 1979, **11**, 585-588.

NOTES

1. If two or more switches must be synchronized to the TV frame rate simultaneously, then Program 6 must be modified. Delete the ",96" from the end of Line 6030 and insert, between Lines 6030 and 6040, a DATA 141,0,192 for each additional switch. Terminate the last DATA statement with a ",96": for example, DATA 141,0,192,96. In Line 6040, increase the end address in the FOR loop by three for each additional switch, for example,

```
6040 FOR I = 800 TO 818
```

for two simultaneous switches. Once this is done, the number of switch specifications must be increased accordingly. The switches can be specified with any of the three methods outlined, but note that the addresses to be POKEd for the switch specification increase by three for each additional switch (e.g., 813 and 816 if two switches are used simultaneously in a routine starting at 800). The order of the switches is irrelevant, since both will be toggled during the same blanking interval. If the BASIC subroutine is changed as follows (after deleting ",96" from Line 6030):

```
6035 DATA 141,0,192,96

6040 FOR I = 800 TO 818
```

then following GOSUB 6000, the insertion of

```
POKE 813,84: POKE 816,86: CALL 800
```

anywhere in the program will switch to Page 1, LORES, whereas

```
POKE 813,85:POKE 816,87:CALL 800
```

will switch to Page 2, HIRES.

2. Here is an example of how to create a three-field tachistoscope by chaining copies of the N-frame subroutine in memory. The subroutine is first expanded to allow toggling of two graphics switches simultaneously. Three overlapping copies are then stored in memory in such a way that three display fields will be presented in succession, each for a specified number of TV frames. A fourth and final field is left on when the subroutine returns to the BASIC program. Assuming that the sequence of fields is always the same, the initial set up can be performed by dropping the ",202" from the end of Line 6030 of Program 7 and replacing Lines 6040-6060 with the following (see Table 1 for other switch values):

```
6040 DATA 141,0,192,202,16,235

6050 DATA 141,0,192,141,0,192,96

6060 FOR I = 800 TO 829: READ V: POKE I + 146,V

6070 IF I < 823 THEN POKE I,V:POKE I + 23,V

6075 NEXT I

6080 REM DEFINE FIRST FIELD SWITCH VALUES******

6090 POKE 815,87: REM HIRES

6100 POKE 818,84: REM PAGE 1

6110 REM DEFINE SECOND FIELD SWITCH VALUES******

6120 POKE 838,86: REM LORES

6130 POKE 841,84: REM PAGE 1

6140 REM DEFINE THIRD FIELD SWITCH VALUES******

6150 POKE 861,87: REM HIRES

6160 POKE 864,85: REM PAGE 2

6170 REM DEFINE TRAILING FIELD SWITCH
     VALUES******

6180 POKE 870,86: REM LORES

6190 POKE 873,84: REM PAGE 1

6200 RETURN
```

Following a GOSUB 6000 in the BASIC program, the presentation duration of each field must be specified (only once if they remain the same throughout the program), and then the routine can be CALLed (CALL 800). Note that because of the manner in which the programs are overlapped, the number of frames shown for presentation is actually one more than specified for all fields except the last (third field in this case). A short example in conjunction with the just-modified subroutine:

```
10 POKE-16304,0:POKE-16298,0:GOSUB 6000

20 POKE 801,9: REM FIRST FIELD TO BE SHOWN FOR 10
   FRAMES
```

```
30 POKE 824,19: REM SECOND FIELD TO BE SHOWN FOR
   20 FRAMES

40 POKE 847,5: REM THIRD FIELD TO BE SHOWN FOR 5
   FRAMES

50 CALL 800: REM DISPLAY SEQUENCE

60 END
```

This example simply displays, in the prescribed order, whatever you had previously on the graphics screens. The addresses used in this example for switch specification and field duration are no longer valid if any of the following are modified: the start address (800 here), the number of switches toggled within a single blanking interval (two here), or the number of fields in the sequence (three plus a trailer here).

(Received for publication June 17, 1980; revision accepted November 6, 1980.)