Behavior Research Methods, Instruments, & Computers
1984, 16 (5), 447-453

COMPUTER TECHNOLOGY

An Apple II-based slide-projector laboratory

GEOFFREY R. LOFTUS, STEVEN GILLISPIE, RALPH A. TIGRE, and WALTER W. NELSON
University of Washington, Seattle, Washington

We describe the software and hardware of a laboratory running under UCSD Pascal on an Apple II computer. The system includes: two random-access projectors and two standard projectors, all equipped with tachistoscopic shutters; two filter wheels; a voice key; a tone generator; and eight response boxes. The laboratory can be used for any experiment in which visual stimuli are to be presented and in which precise display times, reaction times, and luminances are required. It is particularly well suited to picture-perception and picture-memory experiments.

We have constructed a computer-controlled laboratory system designed for performing studies of visual perception and memory in which (1) 35-mm slides are to be used as stimuli and (2) precise stimulus display time and stimulus luminance is required, and/or (3) precise reaction time is required (see Loftus, 1982). The system allows: flexible, tachistoscopic stimulus presentation via four slide projectors; control of stimulus luminance via neutral-density filters; measurement of vocal and digital reaction times; and on-line response collection from up to eight subjects at once.

An overview of the system is followed by a description of system software, system hardware, and a procedure, SEQUENCE, which is the heart of the software timing system. Finally, costs, limitations, and implementation are described.

SYSTEM OVERVIEW

The system is controlled by an Apple II computer, running under UCSD Pascal,[1] and includes the following major components.

(1) Two Kodak random-access projectors.

(2) Two Kodak standard carousel projectors.

(3) Four Gerbrands tachistoscopic shutters with rise and fall times of approximately 1 msec.

(4) Two filter wheels, operated by Superior Electric M61 stepping motors.

(5) Eight Jameco JE 600 response keyboards, operating independently, and each capable of sending a 4-bit response.

(6) A sound-activated relay (voice key) for obtaining vocal reaction time.

(7) A standard Apple II speaker, which is software-integrated with the rest of the equipment.

The random-access projectors are numbered 1 and 2, and the standard projectors are numbered 3 and 4. Any piece of equipment assigned to a particular projector assumes the number of that projector. Thus, for example, Shutters 1-4 are assigned to Projectors 1-4, respectively. The two filter wheels are assigned to Projectors 1 and 3.

All projection equipment (projectors, shutters, and filter wheels) is enclosed in a soundproof box. All equipment can be accessed via statements in Pascal user programs.

SOFTWARE

The system components are controlled by procedures and functions that have been incorporated into a Pascal library unit. The procedures and functions that are typically used for running experiments are organized below primarily in terms of the hardware component to which they pertain.

Random-Access Projectors

Procedure PROJECTOR (PROJNUM, STATE, SLOTNUM) sends random-access projector PROJNUM (1 or 2) to slot SLOTNUM (0-80). STATE is either TRUE or FALSE, depending on whether the projector bulb is to be turned on or off. For normal operation, STATE is TRUE.

Before using the random-access projectors, procedure INITPROJECTORS must be called once. INITPROJECTORS initializes flags and relays that are used subsequently with procedure PROJECTOR.

Standard Projectors

Procedure PROJFOR (PROJNUM) sends standard projector PROJNUM (3 or 4) forward one slot.

Procedure PROJREV (PROJNUM) sends standard projector PROJNUM backward one slot.

The construction of the laboratory described herein was funded by National Science Foundation Grant BNS82-09617 to G. Loftus. The authors' mailing address is: Department of Psychology, University of Washington, Seattle, WA 98195.

Accessing the Shutters

Recall that there are four tachistoscopic shutters, one attached to each of the four projectors. The shutters can be turned on and off in any combination and in any timing sequence. Two procedures are used to control the shutters. SHUTTERS is used when timing is not important. SEQUENCE is used when timing is important.

Procedure SHUTTERS (CODE) effects a particular open/closed combination of the four shutters. The specific combination is determined by the value of CODE. When CODE is in the range 0-15, it is conceptualized as a 4-bit binary code, as shown in Table 1, with the lowest through highest order bits corresponding to the intended states of Shutters 1-4, respectively. Within the 4-bit code, 0 signals "closed" and 1 signals "open." Thus, for example, the statement SHUTTERS (0) would cause all four shutters to close (decimal 0 equals binary 0000), the statement SHUTTERS (10) would cause Shutters 1 and 3 to close and Shutters 2 and 4 to open (decimal 10 equals binary 1010), and so on.

Values of CODE greater than 15 will be described below.

Procedure SEQUENCE also allows the four shutters to be opened and closed, in any combination, but for precise times. The timing error within the SEQUENCE software is on the order of microseconds, and is considerably less than the mechanical timing error inherent within any mechanical shutter. Software timing error can thus be ignored. SEQUENCE also allows variable-frequency, computer-generated tones to be delivered at any time within a shutter sequence. The exact operation of SEQUENCE will be described more fully in a later section.

Tone Frequency

The function FREQUENCY (REALFREQ) causes the tone output with the SEQUENCE routine to be of frequency REALFREQ (REALFREQ in Herz).

Table 1
SEQUENCE Codes and Their Effects

Decimal Code	Binary Code	Action
00	0000	All shutters closed
01	0001	Shutter 1 open, rest closed
02	0010	Shutter 2 open, rest closed
03	0011	Shutters 1 and 2 open, rest closed
04	0100	Shutter 3 open, rest closed
05	0101	Shutters 1 and 3 open, rest closed
06	0110	Shutters 2 and 3 open, rest closed
07	0111	Shutters 1, 2, and 3 open, 4 closed
08	1000	Shutter 4 open, rest closed
09	1001	Shutters 1 and 4 open, rest closed
10	1010	Shutters 2 and 4 open, rest closed
11	1011	Shutters 1, 2, and 4 open, 3 closed
12	1100	Shutters 3 and 4 open, rest closed
13	1101	Shutters 1, 2, and 4 open, rest closed
14	1110	Shutters 2, 3, and 4 open, 1 closed
15	1111	All shutters open
16		Exit SEQUENCE
17		Start tone
18		Stop tone

Filter Wheels

Procedure WHEELS (PROJNUM, FILTERNUM) positions Filter FILTERNUM (1-6) in front of Projector PROJNUM (1 or 3). WHEELS works as follows.

Each filter wheel is attached to a rotating stepping motor. There are 400 stepping-motor steps and six evenly spaced filter slots; thus, the filters are centered at approximately every 400/6 = 66 stepping-motor step positions. A table relating filter-slot positions to stepping-motor step positions must be made accessible to any Pascal user program that manipulates the filter wheels. The relationship between stepping-motor positions and filter positions for each of the two filter wheels is determined only when the wheels are initially installed. A table containing these relationships is stored as a file on the Pascal system disk, from which it can be easily retrieved by user programs.

We have found it useful to accelerate and decelerate the wheels. If this is not done, the wheels must move very slowly; otherwise, their inertia prevents them from moving and/or their momentum causes them to overshoot. WHEELS automatically accelerates and decelerates the filter wheels, allowing much faster rotation speeds.

Prior to using WHEELS, procedure INITWHEELS must be called once. This procedure "zeros" each filter wheel by turning it until a small tab protruding from the wheel breaks an optical circuit. Each time Procedure WHEELS is called, the position of the wheel relative to this arbitrary zero point is recomputed.

Response Boxes

Prior to accessing the response boxes, the user program must specify whether each of the eight boxes will be active or inactive during a given experimental session. This is done by setting each location of a Boolean array, ACTBOXES [1..8], to TRUE or FALSE, depending on whether the corresponding box is active or inactive.

Procedure READBOXES (RESULTS) awaits responses from the active boxes and returns these responses in an integer array, RESULTS. The program "hangs" while in READBOXES; that is, no other computation can be performed by the main program while it is waiting for all boxes to respond.

The response boxes allow entry of the digits 0-9 and letters A-F. READBOXES is designed such that, from a subject's point of view, the F key acts like an enter or a return key: it is used to complete a response. Up to four digits[2] (which are transformed by READBOXES into a single integer, ranging from 0-9999) can be entered, followed by the F key. If the subject makes an error prior to pressing the F key, it can be erased by pressing the E key.

Procedure TIMEBOXES (RESULTS, TIMES) is used whenever reaction times are required. TIMEBOXES works much like READBOXES, except that reaction times, in milliseconds, are returned along with the responses.

Voice Key

Function VOICEKEY returns an integer that is the time, in milliseconds, elapsing between the calling of VOICEKEY and the activation of the voice-key microphone by a subject. Times are read from a Mountain Hardware clock.

Waiting

Procedure WAIT (TIME) causes a delay of TIME milliseconds. It can be used in simple timing situations that require no more than millisecond accuracy.

HARDWARE

Figure 1 illustrates the hardware system. The computer is an Apple II Plus, with 64K of memory. Input-output (I/O) is accomplished by four devices: two Interactive Systems DI-09 I/O cards, a California Computer serial port card, and the standard Apple II game controller input port.

DI-09 Cards

The two DI-09 cards are responsible for all I/O except control of the random-access projectors and the voice key. Briefly, each of these cards has 40 1-bit I/O lines. Thirty-two of these lines are referred to as data lines, and the other 8 are referred to as control lines. The difference between data and control lines is irrelevant for the present discussion. Each of the 40 lines can, via a software switch, be used as either an input or an output line. In addition to the I/O lines, each card includes two microsecond-accuracy timers, which we use principally in the SEQUENCE routine (see below).

All 32 data lines of the Slot 1 DI-09 card are used for coding the 4-bit responses from the eight response boxes. The Slot 4 DI-09 card is responsible for a variety of functions, which are as follows.

Response box strobes. Eight control lines—one from each response box—are used to signal that a button has been pushed on that box. The READBOXES and TIMEBOXES routines (see above) spend the majority of their time polling these strobe lines, waiting for all active boxes to respond. In TIMEBOXES, the detection of a response from a box is followed by recording the time via the Mountain Hardware clock.

Shutters. Four data lines control the four shutters. Each line acts as a switch to turn the shutter's power supply on or off, which causes the shutter itself to be opened or closed.

Filter wheels. Eight data lines control the two filter-wheel stepping motors. Four lines for each motor are used to specify one of the 400 stepping motor steps, to be arrived at by rotation in a particular direction.

Standard projectors. Four data lines control the forward/reverse functions for the two standard projectors. The two lines for each projector control two independent relays—one that directs the projector to go forward and one that directs it to reverse.

Filter-wheel zeroing. Two data lines signal the breaking of each of the two optical circuits ("optos") used for zeroing the two filter wheels.

Random-Access Projectors

The on-line random-access projector is controlled by a Mast 140-RS interface. This interface accepts as input a BCD code through a California Computer serial port card. The BCD code determines both the power status (on or off) of the projector and the desired slot number.

Voice Key

The voice key is interfaced via the standard Apple II game controller input port. When the voice-key microphone is activated, an impulse is sent to the game controller port, just as if the game-paddle button had been pushed. This arrangement requires no mechanical manipulation of the voice key during the running of an experiment.

When function VOICEKEY is called by the user program, the following occurs. First, the time at which the procedure is called is read from the Mountain Hardware clock. The time is then read again upon receipt of a signal through the game controller port. The difference between the two times is returned as the value of the function.

HOW SEQUENCE WORKS

SEQUENCE is used for executing precisely timed stimulus displays. Table 1 above provides the codes and corresponding operations that may be effected by SEQUENCE. As noted earlier, codes 0-15 correspond to the 16 possible open/shut combinations of the four shutters. Code 16 is used to escape from SEQUENCE and will be described momentarily. Codes 17 and 18 turn a computer-generated tone on and off, respectively. The frequency of the tone is under control of the user program.

We had two goals in the design of SEQUENCE. The first was to allow timing accuracy on the order of microseconds. To accomplish this, SEQUENCE ordinarily waits in an assembly language routine, waiting for an interrupt to be generated after a preset interval by one of the DI-09 timers. The second goal was to be able to easily exit from SEQUENCE in order to do such housekeeping chores as changing projectors, collecting and storing data, and so on, when time permits (e.g., during intertrial intervals).

SEQUENCE is based on two one-dimensional arrays, SEQTIMES and SEQEVNTS, along with SEQPTR, a pointer into the arrays. The user program must provide SEQTIMES with a sequence of cumulative times (in milliseconds) starting from some arbitrary time, 0. The corresponding array locations in SEQEVNTS must be filled with the codes of the events that are to take place at these times. SEQPTR designates the next location in SEQTIMES/SEQEVNTS that is to be processed.

Ordinarily, SEQUENCE works as follows.

(1) The program waits for an interrupt.

(2) When the interrupt occurs, the event indicated by the code at SEQEVNTS [SEQPTR] takes place.

(3) The next time interval is computed by subtracting

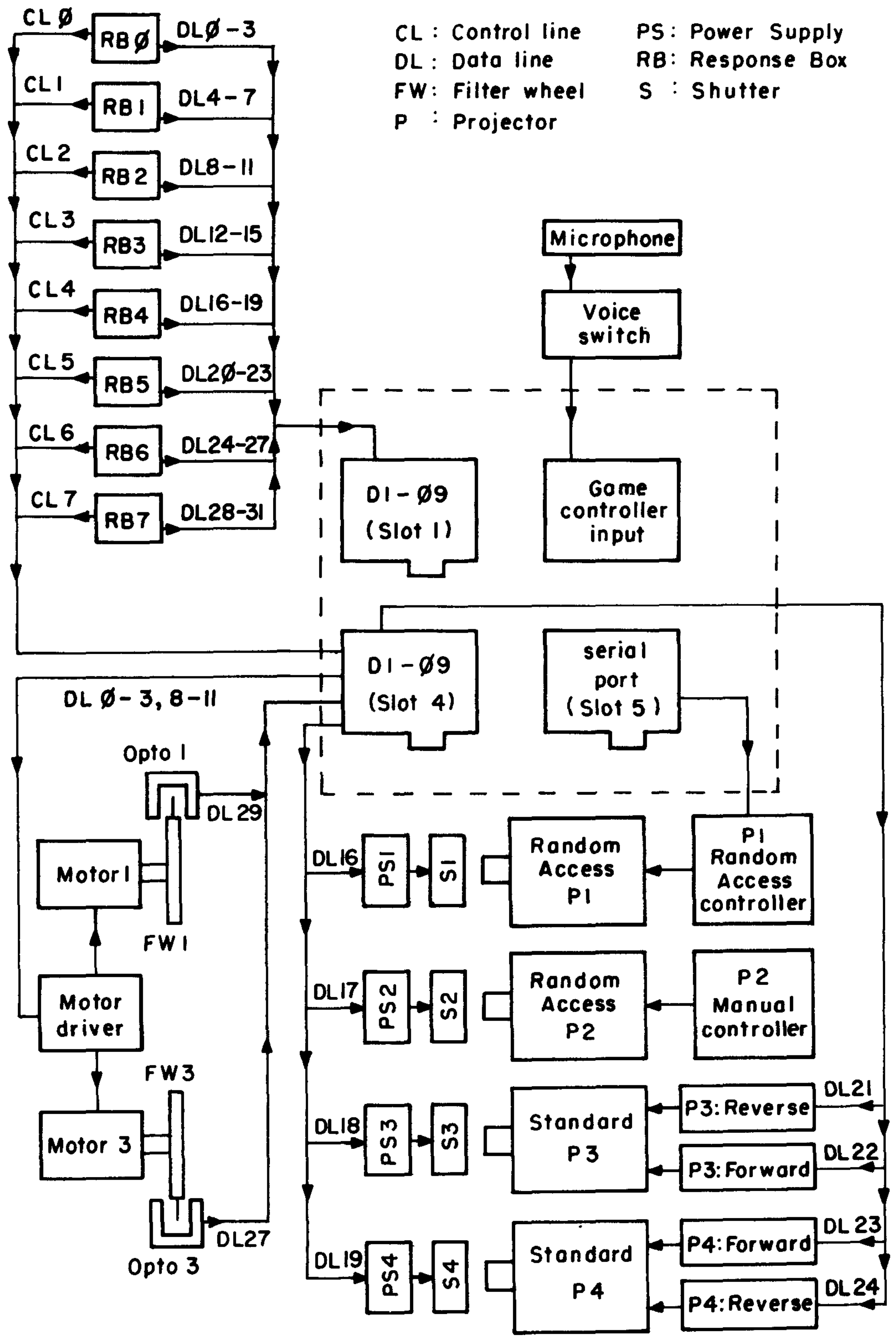

Figure 1. Schematic of the laboratory apparatus. Dashed lines enclose the Apple II. Display apparatus is below; response-collection apparatus is above. All projectors except Projector 2 are on line.

SEQTIMES SEQPTR from SEQTIMES [SEQPTR + 1].

(4) An interrupt is programmed to occur after that time interval has elapsed.

(5) The next code, SEQEVNTS [SEQPTR + 1], is noted.

(6) SEQPTR is incremented by one.

(7) If the next code (noted in Step 5) is anything but 16, the exit code, the program returns to its "hang" state within SEQUENCE. If the next code is 16, the program exits SEQUENCE and returns to the Pascal program.

Sometime prior to first executing SEQUENCE, procedure ENABLE (INITIALTIME) must be called. ENABLE rearranges the Pascal interrupt system and sets the timer to generate an initial interrupt INITIALTIME later. After all calls to SEQUENCE have been completed, procedure DISABLE must be called. DISABLE reestablishes the original Pascal interrupt system.

To appreciate the flexibility of this arrangement, suppose that a series of stimuli is to be presented, each stimulus for 50 msec. There are two ways in which this might be done.

First, stimulus onset asynchrony (SOA) between the onsets of stimulus n and stimulus n+1 might not be important. Table 2a shows Pascal code that would accomplish the task. SEQUENCE is used in a very simple way. Essentially, the experimenter controls when SEQUENCE is entered, that is, when the stimulus presentation occurs. The SEQUENCE exit code, 16, is associated with the same time as the offset of the stimulus. Thus, whenever SEQUENCE is entered, no interrupt is imminent, and the first code (1, or "open shutter 1") is executed immediately, followed by the remaining codes in SEQEVNTS at their appropriate times.

In the second case (Table 2b), a specific SOA, 10,000 msec, is desired. To accomplish this, the SEQUENCE exit code, 16, is associated with a time of 10000. The program will exit SEQUENCE immediately after closing all shutters, since, upon looking ahead in SEQEVNTS, it will have found the exit code. But before exiting, it will set an interrupt to occur at time 10000 (relative to time 0 when the shutter first opens, since all response times are cumulative.) Presumably, the house-

Table 2
Pascal Code for Displaying a Stimulus

(a) SOA between stimulus onsets is not important. Experimenter initiates each trial.

```
ENABLE (0); (* Set up interrupt sysem for SEQUENCE *)

(* Fill SEQUENCE *)
SEQTIMES [0] :=0; (* At time 0 . . . *)
SEQEVNTS [0] :=1; (* Shutter 1 opens *)

SEQTIMES [1] :=50; (* for 50 ms *)
SEQEVNTS [1] :=0;  (* and then closes *)

SEQTIMES [2] :=50; (* Immediately thereafter *)
SEQEVNTS [2] :=16; (* Exit SEQUENCE *)

FOR TRIAL :=1 TO TTRIAL DO (* Loop through TTRIAL trials *)
BEGIN
    PROJECTOR (1, ON, TRIAL); (* Move Projector 1 to slot TRIAL *)
    SEQPTR :=0; (* Reset SEQPTR *)
    WRITE ('Press anything to show stimulus . . .');
    READ (CHARACTER); (* Wait for keyboard character *)
    SEQUENCE; (* Present display *)
END; (* TRIAL loop *)

DISABLE; (* Return to standard interrupt system *)
```

(b) SOA is to be 10 sec (10,000 msec). Sequencing proceeds automatically

```
ENABLE (0); (* Set up interrupt system for SEQUENCE *)

(* Fill SEQUENCE *)
SEQTIMES [0] :=0; (* At time 0 . . . *)
SEQEVNTS [0] :=1; (* Shutter 1 opens *)

SEQTIMES [1] :=50; (* for 50 ms *)
SEQEVNTS [1] :=0;  (* and then closes *)

SEQTIMES [2] :=10000; (* 10 sec from shutter opening, next interrupt will occur *)
SEQEVNTS [2] :=16; (* Exit SEQUENCE *)

FOR TRIAL :=1 TO TTRIAL DO (* Loop through TTRIAL trials *)
BEGIN
    Projector (1, ON, TRIAL); (* Move Projector 1 to slot TRIAL *)
    SEQPTR :=0; (* Reset SEQPTR *)
    SEQUENCE; (* Present display *)
END; (* TRIAL loop *)

DISABLE; (* Return to standard interrupt system *)
```

keeping chores to be carried out during the intertrial interval (which, in this example, consist only of changing the projector and resetting SEQPTR) will be accomplished before this interrupt occurs. By the time the interrupt does occur, the program will have safely reentered SEQUENCE, will have discovered that an interrupt is imminent, and will be hanging within SEQUENCE, waiting for the interrupt to occur. When the interrupt does eventually occur, SEQPTR will have been reset to 0, and hence the sequence of events will begin, appropriately, at SEQTIMES [0] and SEQEVNTS [0].

This latter example illustrates the ability to change SEQPTR within the user program. Such ability permits a good deal of flexibility in that it allows any subset of events within SEQUENCE to be easily skipped, repeated, or modified.

IMPLEMENTATION

At present, the laboratory has been operational for 18 months. Approximately 40 experiments have been run in it, involving roughly 3,000 subject hours.

Costs

Table 3 lists all equipment costs (as of September 1982.) The voice key and the shutter power supplies[3] were the only items not purchased commercially. The overall hardware cost was approximately $14,400. Construction and debugging time was reasonable: Approximately 5 months elapsed between ordering the components and completing the first computer-controlled experiments.

Transportability

The system is immediately transportable to all Apple II, II Plus, and IIe computers. It is not transportable to the Apple IIc, since the latter does not have suitable output ports.

Transportability to other computer systems would necessitate several major modifications. First, of course, the assembly-language subroutines would have to be translated. Second, a substitute would be required for the DI-09 cards, which are designed for Apple computers. The basic software logic would remain unchanged.

Limitations

The system is designed expressly for experiments that use 35-mm slides as stimuli. The use of 35-mm slides in a computer-based laboratory has advantages and disadvantages, which are described in detail by Loftus (1982). The major timing and luminance limitations of the present system are as follows.

The nature of the shutters and the shutter power supplies is such that timing becomes inaccurate below about 20 msec. A further timing limitation results from the (approximately) 1 sec required to advance slides in the slide projectors. This means that, in a continuous-presentation paradigm such as those used by Potter (e.g., Potter & Levy, 1969) or Intraub (e.g., Intraub, 1980), the shortest possible average presentation time is roughly 250 msec. This limitation could be circumvented by using more than four slide projectors/shutters.

Within a given experimental session, luminance is quite precisely controllable via neutral-density filters. However,

Table 3
Costs of System Components

Computer	
Apple II, 2 disk drives, 64K of memory, Pascal software, monitor	$3,213
Interactive Structures DI-09 cards, 2 at $330 each	660
Hi-current kit for above	51
Mountain Computer Apple clock	276
California Computer Serial port card	160
Apple II Superfan	50
Interfaces	
Jameco JE-600 response boxes, 8 at $107 each	856
Anaheim Automation step motor driver DPA-05	740
Mast 140-RS Random Access Interface	662
Custom-made voice-switch ports	100
Custom-made shutter power supplies, 4 at $100 each	400
Cables, connectors, chassis boxes, electronic hardware	1,500
Optical Hardware	
Kodak RA-960 random-access slide projectors, 2 at $1,485 each	2,970
Kodak manual projectors, 2 at $250 each	500
Projector stands, 2 at $253 each	506
Gerbrands G-1166 shutters, 4 at $115 each	460
Superior Electric M61 stepping motors, 2 at $115 each	230
Neutral-density filters, 10 assorted	292
Soundproof Box Costs	
Soundproof box, supports, fans, hardware	650
Power strips, 2 at $50 each	100
Total	$14,376

the slide-projector bulbs undergo slight luminance changes over their lifetimes. This property imposes the major limit on luminance control.

Running Experiments

A typical picture-memory experiment might involve 20 groups of five to eight subjects per group. The total time required for such an experiment is approximately 25 h, broken down as follows.

(1) Four hours to modify previous Pascal programs—three programs are used for a given experiment: one to randomize presentation orders, one to run the experiment and collect the data, and one to compute means and standard deviations from the raw data.

(2) Twenty hours to actually run the subjects—data, collected via the response boxes, are analyzed at the end of each experimental session and are presented to subjects during debriefing, which occurs immediately after an experimental session.

(3) An hour at the end of the experiment for statistical analyses.

Thus, the laboratory permits an experiment to be run almost as speedily and efficiently as is possible. Of the total 25 h spent on this typical experiment, 20 h constitutes the irreducible time necessary to actually collect the data. In a typical experiment, each data point is based on 700-1,000 observations.

Experimental Paradigms

Examples of the experimental paradigms that have been used are the following.

(1) Visual-memory experiments have investigated acquisition of information from the iconic image (Loftus, Johnson, & Shimamura, 1985), picture luminance effects (Loftus, in press), and perceptual/conceptual masking effects (Loftus & Ginn, 1984). Performance measures, all obtainable via the response boxes, include old/new recognition, number of picture details recalled, confidence in subsequent recognition, and partial report of digit stimuli (Sperling, 1960).

(2) The effects of priming on object perception have been investigated in two ways. First, the probability of correctly naming a briefly presented, masked object has been measured. Second, vocal reaction time to name an object has been measured using the voice key.

(3) Response-contingent stimulus presentation has been used to investigate subjective duration of the icon in a synchrony-judgment paradigm (see Efron, 1970). In this paradigm, a target picture is displayed, followed by a variable-length blank interval, followed by a signal, such as a visual mask or an auditory beep. The subject's task is to adjust the length of the interval such that the post-interval signal seems to coincide with the disappearance of the icon. This is accomplished over a series of trials by increasing or decreasing the interval duration in accord with the subject's report of the temporal relation between signal and disappearance of the icon.

(4) Psychophysical brightness-judgment tasks (see Stevens, 1957) have been used to assess effects of stimulus complexity on perceived brightness.

(5) Segmented slides showing parts of a picture have been presented sequentially to simulate movement of the gaze from one part of the picture to another. This technique precisely controls how much information is available for each eye fixation. Several such visual-integration experiments that incorporate all features of the present system have been performed.

REFERENCES

EFRON, R. (1970). The relationship between the duration of a stimulus and the duration of a perception. *Neuropsychologia*, **8**, 37-55.

INTRAUB, H. (1980). Presentation rate and the representation of briefly glimpsed pictures in memory. *Journal of Experimental Psychology: Human Learning and Memory*, **6**, 1-12.

LOFTUS, G. R. (1982). Picture memory: Data and methodology. In C. R. Puff (Ed.), *Handbook of research methods in human memory and cognition*. New York: Academic Press.

LOFTUS, G. R. (in press). Picture perception: Effects of luminance on available information and information-extraction rate. *Journal of Experimental Psychology: General.*

LOFTUS, G. R., & GINN, M. (1984). Perceptual and conceptual masking of pictures. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, **10**, 435-441.

LOFTUS, G. R., JOHNSON, C. A., & SHIMAMURA, A. P. (1985). How much is an icon worth? *Journal of Experimental Psychology: Human Perception and Performance*, **11**, 1-13.

POTTER, M. C., & LEVY, E. I. (1969). Recognition memory for a rapid sequence of pictures. *Journal of Experimental Psychology*, **81**, 10-15.

SPERLING, G. (1960). The information available in brief visual presentations. *Psychological Monographs*, **74**, 1-29.

STEVENS, S. S. (1957). On the psychophysical law. *Psychological Review*, **64**, 153-181.

NOTES

1. A copy of the system software as well as a demonstration Pascal program may be obtained by sending a blank, 5.25-in. disk to the first author at the Department of Psychology, University of Washington, Seattle, WA 98195.

2. The value of four digits is arbitrary.

3. The standard Gerbrands power supply requires turning down the power with a manual switch if the shutter is to be open for more than a few seconds. This is necessary because the power required to rapidly open the shutter will burn out the shutter solonoid if left on too long. The power supplies that we have designed provide an initially high surge of power and then automatically decrease power to an amount just sufficient to keep the shutters open.

(Manuscript received, July 11, 1984;
revision accepted for publication November 14, 1984.)