DOCUMENT RESUME

ED 067 810 EC 050 101

AUTHOR	Semmel, Melvyn I.; And Others
TITLE	An Information and Technical Manual for the Computer-Assisted Teacher Training System (CATTS).
INSTITUTION	Indiana Univ., Bloomington. Center for Innovation in Teaching the Handicapped.
SPONS AGENCY	Bureau of Education for the Handicapped (DHEW/OE), Washington, D.C.
PUB DATE	Jun 72
GRANT	OEG-0-242178-4149-032
NOTE	115p.; Working Paper 7.1
EDRS PRICE	MF-$0.65 HC-$6.58
DESCRIPTORS	Classroom Observation Techniques; *Computer Assisted Instruction; *Computer Programs; Educational Technology; Feedback; Guidelines; *Interaction Process Analysis; *Student Teacher Relationship; *Teacher Education

ABSTRACT

The manual presents technical information on the computer assisted teacher training system (CATTS) which aims at developing a versatile and economical computer based teacher training system with the capability of providing immediate analysis and feedback of data relevant to teacher pupil transactions in a classroom setting. The physical configuration of CATTS consisting of three interdependent stations (teaching, observation-coding, and analysis-encoding) is described. Hardware components (coding, computer, and output) of the prototype CATTS are graphed and discussed. It is explained that the basic control element in the CATTS system is a small digital computer which is programed to provide system timing, information collection, and feedback of results. Three variations of the basic system are considered including a consensus coding system, a simultaneous coding system, and CATTS with tally pattern recognition. Discussed are current hardware and software extensions of CATTS in the areas of data collection and input developments, extensions of the real-time multi-observer system, and feedback and output developments. Related research in the establishment of a general training laboratory is considered. (GW)

Center for Innovation in Teaching the Handicapped

School of Education, Indiana University, Bloomington

AN INFORMATION AND TECHNICAL MANUAL FOR THE COMPUTER ASSISTED TEACHER TRAINING SYSTEM (CATTS)

MELVYN I. SEMMEL, JERRY L. OLSON,
AND WILLIAM M. WEISKE, JR.

AN INFORMATION AND TECHNICAL MANUAL FOR THE COMPUTER-ASSISTED TEACHER TRAINING SYSTEM (CATTS)

Melvyn I. Semmel
Project Director

Jerry L. Olson
Supervisor of Technical Services

William M. Weiske, Jr.
Senior Systems Programmer

June, 1972

Working Paper 7.1

Teacher Education Laboratory

Center for Innovation in Teaching the Handicapped

Indiana University

Preliminary research and development of CATTS (Semmel, 1968) was supported by a grant to the Center for Research on Language and Language Behavior, contract OEC-3-6-061784-0508 with the U. S. Department of Health, Education and Welfare, Office of Education, at the University of Michigan, Ann Arbor, Michigan.

This manual was supported in part by a grant to the Center for Innovation in Teaching the Handicapped, contract OEG-0-242178-4149-032 with the U. S. Department of Health, Education and Welfare, Office of Education, Bureau for Education of Handicapped Children, at Indiana University, Bloomington, Indiana.

PREFACE

Within the last decade technological progress has extended the use of computers from the synthesis and analysis of data punched on cards to the development of "on-line" or "real-time" systems which communicate directly through remote terminals. Data can be instantaneously analyzed and sorted in random access files--and relevant messages are relayed quickly to remote terminals. Such systems are efficiently arranging international airline and hotel reservations, controlling the quality of parts in automobile factories, and monitoring and correcting manned space flights.

Computer technology makes possible modes of displaying output in forms which even the most naive among us may interpret. On the input side, advances have brought the language of the machine to where it approximates the natural language of man. The computer is rapidly becoming a servant to more and more men. It appears reasonable to predict that the "servant" will soon be performing a greater number and variety of tasks which man cannot do as efficiently or as rapidly--thus freeing man for more creative and humanistic work.

Educational institutions are beginning to profit from the advances in computer technology. Many large school systems currently utilize computers to facilitate scheduling, general accounting, grading, and other automatic functions which had previously demanded the laborious efforts of professional personnel. The advent of "real-time" systems and "shared time" arrangements has brought the capabilities of rapid

analysis and feedback directly into the learning situation through programmed instructional techniques and audio-visual approaches. Computer monitored instructional programs are currently being designed to reach large numbers of pupils with fewer numbers of teachers. Other computer-assisted instruction (CAI) programs seek to individualize instruction for specific children. Since automated programmed instruction has already been applied to handicapped pupils in our schools (Malpass, 1964; Stolurow, 1963) it is anticipated that computerized instructional techniques will also gain acceptance for this population.

When considering the applications of computer technology to remedying the learning problems of handicapped children, it would seem judicious to predict that these machines and relevant software will undoubtedly facilitate the teacher's task in the school. However, the use of computer technology will not reduce the need for adequately trained personnel who can teach handicapped children in classroom settings. Teachers may eventually need the ability to communicate with sophisticated man-machine systems, but these can be no substitute for their ability to understand and teach handicapped pupils. Hence, it appears equally appropriate to explore the potential of computers for preparing personnel to work with handicapped pupils in special education programs.

In this document, the authors outline an approach for applying computer technology to the understanding of the teaching process, and to the training of teachers. While much of the discussion projects

into the future, research and development activities are described which have already been conducted in the training laboratory at the University of Michigan (Semmel, 1968) and which are continuing at the Center for Innovation in Teaching the Handicapped at Indiana University.

This initial publication of a manual for the Computer-Assisted Teacher Training System (CATTS) is written in response to requests for more detailed documentation of the prototype CATTS system briefly described in the authors' earlier working papers (Semmel, 1968). An attempt has been made to explain fully the present status of all phases of the basic CATTS hardware and software systems, including expansion programs which were not reported in the initial working papers. A discussion of current system developments and related training laboratory developments has also been included. The system's documentation at this time is by no means complete; but through the expanded explanation of both the hardware configuration and main software operating program, this manual provides a more complete understanding of the basic operating system. Detailed supplementary reports will follow as new developments evolve and are tested at the Center for Innovation in Teaching the Handicapped.

Melvyn I. Semmel
June, 1972

Table of Contents

CHAPTER 1

INTRODUCTION

Although existing special education teacher training programs generally lack specificity about their objectives and procedures (Cruickshank, 1967), there appears to be an agreed hierarchy of emphasis on what is important in the training process. The amount and nature of practicum experiences is paramount in this hierarchy (Blatt, 1964, 1966). Direct contact with children is thought to be more valuable to trainees than vicarious exposure through lectures and discussions about the teaching process. However, simply providing an opportunity to observe or interact in a special or regular classroom setting does not assure the growth of trainees with regard to acquisition of specific teaching skills any more than do lectures or discussions in a university methods course.

Programs differ considerably in the nature and amount of structure offered to trainees in practicum environments. At one extreme the trainee is assigned a "master" teacher who is assumed to have the necessary skills for training the apprentice (Olson & Hahn, 1964). Usually the master teacher expects the trainee to teach as he does. Frequently, he assigns the trainee special tasks such as working with specific children or performing nonteaching assignments. At the other extreme we find the practice of periodic observation of trainees in situ, followed by supervisory conferences. In this case, impressions and observations are transmitted to the trainee, who is then expected to modify his behavior as a result of the feedback (Anderson & Junka, 1963).

The latter model appears superior to the former as a means of achieving the goals of a university training program. However, closer

analysis of the supervisory feedback process reveals that the trainee often derives little information about the specific behaviors (objectives) deemed important by the program. Further, the training supervisor often has no systematic technique for focusing on those teaching behaviors which are considered relevant. The supervisor too often relies on feeding back vague ad hoc impressions to the trainee, resulting in little relationship between one supervisory conference and another.

A detailed examination of the literature demonstrates that little attention has been given to the development and demonstration of methods of teacher training designed to eliminate the problems outlined above (Cain, 1964; Blatt, 1966; Guskin & Spicker, 1968). Clearly, there is a need for observational and feedback systems which focus on relevant training variables, and which assist teachers in skill development congruent with the philosophical orientation of the program that trains them. The need exists quite independently of the relationship of observable teaching behaviors to pupil growth criteria. Training programs first must operationalize those skills which they and/or their trainees define as most appropriate for teaching the handicapped child, then develop procedures to observe and modify trainee behavior toward these goals. Therefore, the task for university training programs is to demonstrate the ability to teach adults (i.e., train teachers) to develop specific teaching behaviors, patterns, and pedagogical environments in working with handicapped pupils (Gallagher, 1967).

In effect, it must be the assumption of the training program that these teaching strategies produce the best effects in classes with special children. An evaluation of the performance of these teaching

skills in the classroom serves as an empirical test of the hypothesis that the specific program objectives have a facilitating effect on the learning and behavior of handicapped pupils. The failure of previous research to demonstrate the efficacy of special classes for handicapped pupils may, in part, have been due to a failure to validate that different "special" teaching methods were, in fact, being used in such classes when compared to regular classes (Dunn, 1968).

Interaction Analysis Systems in Teacher Training

Many observation systems have been developed and tested by educators interested primarily in regular classroom pedagogy. The categories used in these systems are operational definitions of what the designers consider important classroom behavior. When teachers or trainees are encouraged to favor one subset of behavior from the total set subsumed by the system, the trainer may be said to have established specific behavioral objectives for the trainee.

Generally, existing systems attempt to classify interactions of teachers and pupils into different content categories. For example, Flanders (1964) has focused on categories of teacher and pupil talk (e.g., praise and encouragement, questions, lecturing, and student-initiated talk). Bellack, Hyman, Kliebard and Smith's (1966) system focuses on the nature of teacher's and pupil's interacting strategies, while Gallagher's (1965) system of analysis stresses cognitive behaviors modeled after Guilford's (1956) structure of the intellect paradigm. Still other systems have been presented which isolate different aspects of significant teacher-pupil behavior (Simon & Boyer, 1970; Medley & Mitzel, 1963).

A number of studies have been reported which attempt to use observation-coding systems to characterize the behavior of teachers and pupils in special education contexts (Minskoff, 1967; Cruickshank, 1967; Semmel & Kreider, 1971). However, relatively few attempts have been made to use systematically observation-coding systems as operational tools in special education teacher training programs.

The field of special education has produced relatively little research and development on systems of observation and techniques for feedback of specific classroom teacher-pupil interaction variables which are based on characteristics of the children or on special education methods (Guskin & Spicker, 1968).

Several new systems are currently being developed at the Center at Indiana University which focus on specific pedagogical considerations in working with educable mentally retarded (EMR) and emotionally disturbed (ED) pupils. The Indiana Behavior Management System (Fink & Semmel, 1971) is an evolving observation-coding system which focuses on deviant pupil classroom behavior and the behavior management skills used by teachers.

The Individual Cognitive Demand Schedule developed by Lynch and Ames (1971) attempts to assess the cognitive demands made by teachers of EMR pupils, to evaluate the responses of pupils, and to characterize the teacher's feedback behavior. The system permits tracking these triadic transactions between specific children in the classroom and their teacher.

Most teacher education programs generally use the same paradigm when applying observation-coding systems in their training programs. An

observer or supervisor sits in the classroom and either records the ongoing behavior using a prescribed code, or he records verbal interaction on magnetic or video tape and later transcribes and codes it to conform to the particular system adopted. In some methods the data obtained are coded at regular intervals and entries are subsequently summarized in a matrix reflecting the sum of double entry Markov chains, i.e., the frequency of behavior category X that followed category Y (Flanders, 1964). In others, simple proportions of the behavioral categories represented in the total corpus of material are coded. Interesting ratios and transformations of teacher and pupil categories can be calculated and related to characteristic patterns and outcomes in the classroom. The synthesis of data collection is subsequently shared with and interpreted to the teacher trainee, who is generally expected to alter his next teaching performance in an agreed upon direction. Hence, these systems are retrospective in that they are designed and used to summarize classroom transactions after they have taken place. When the results are used in training programs, they cannot be fed back to a teacher or trainee in real-time. Knowledge of results can have no immediate effect on the environment from which the data are drawn.

These observation-coding feedback systems are obviously subject to serious limitations as operational tools for teacher training programs. It is apparent that they involve extensive time commitments on the part of the trainer-coder who is required to observe, code, summarize, analyze, and feed back the results to the trainee. The cost of training of this type is very high indeed. Furthermore, segmenting and summarizing observed behavior produces an analogue of the teacher-pupil transactions

across a given period which distorts or loses much of the dynamic elements of what has taken place in the classroom. For example, while the Flanders system retains two-stage interaction chains, provision must be made for determining when such chains occur across the lesson. Sequential interactions are also obscured by the methods currently available for processing coded data and necessitate relatively long delays in feedback to trainees. Hence, it is questionable that the consequent feedback to trainees could have maximum effects on the modification of subsequent teaching behaviors.

The problems inherent in present observation-coding-feedback programs can be minimized, and systems adapted for operational special education training programs, by development of a man-machine system which provides for the following elements:

1. Rapid feedback of relevant information to the trainee, <u>while</u> he is teaching, through a feedback source located in the teaching environment.
2. The elimination of the tedium associated with coding, summarizing and analyzing observational data relevant to teacher-pupil interactions.
3. The development of analytic techniques for the rapid description and synthesis of teacher-pupil interactions while maintaining the essential interactive variables and their sequential temporal relationships.
4. The rapid cumulative storage and retrieval of pupil-teacher interaction data for the evaluation of growth of trainees participating in the program.

One promising direction toward meeting these criteria is through the development and utilization of a real-time computer-assisted teacher training system (CATTS).

Toward the Development of CATTS

Earlier pilot work at the University of Michigan on the analysis of pupil-teacher interaction in the classroom interested the senior author and his associates at the Center for Research on Language and Language Behavior (CRLLB) in the problem of the systematic real-time analysis and modification of teacher behavior. An extensive demonstration project was designed to determine the effects of feedback on teacher trainees who were systematically observed and evaluated during 15 half-hour practicum teaching lessons. Trainees were taught to use a modified version of the Bellack (1966) system of analysis to evaluate their performances from magnetic tape recordings of the sequence of lessons which they taught; supervisors were trained to feed back corrective information to individual trainees and to suggest specific teaching styles according to the amount and quality of teacher talk in the classroom. This pilot work served as the precipitant for the development of CATTS since it appeared to highlight the need for immediate feedback to trainees and also proved impractical as an operational procedure in light of the efforts which were necessary to derive the training data.

The goal for CATTS is to develop a versatile and economical computer-based teacher training system with the capability of providing immediate analysis and feedback of data relevant to teacher-pupil transactions in a classroom setting.

When CATTS is operational it should be applicable to any training situation in which:

1. The interaction of teachers and pupils is to be summarized or analyzed in terms of any system composed of behavioral categories.
2. The summarized and analyzed data are to be fed back immediately to the teacher in the classroom through a meaningful visual and/or auditory source.
3. The behavior, once coded, summarized, and analyzed by computer, is to be instantaneously stored for rapid subsequent retrieval.

Work on CATTS is presently directed toward practical application in university special education teacher training programs, in-service continuing education programs for special teachers in the school, and all programs that train personnel to direct and lead groups of children or adults.

The Prototype CATTS Configuration and Preliminary Research

In their work, Cybernetic Principles of Learning and Educational Design, Karl and Margaret Smith (1966) base their approach to human learning on the findings of early researchers in human engineering. The Smiths argue convincingly for a cybernetic interpretation of behavior--one quite different from conventional theories of learning. The cybernetic approach is a "general theory of behavior organization which . . . views the individual as a feedback system which generates its own activities in order to detect and control specific stimulus characteristics of the

environment [p. vii]. CATTS is currently conceptualized as a closed-loop cybernetic system which provides immediate feedback of relevant teacher-pupil interaction variables to the teacher trainee so that modification of trainee behavior can be realized through regulatory teaching moves in accordance with a predetermined strategy, thus creating the desired classroom environment (Semmel, 1968). The system enables a trainer to stipulate clearly those elements or patterns of teaching behavior which he wishes to develop as goals for training. Real-time feedback of performance is provided to the trainee so that regulatory behavior may be initiated toward establishing a desired classroom learning environment for the pupils. The trainees' progress toward achieving objectives can be systematically and cumulatively tracked and evaluated by the computer and its analytic and memory storage capabilities.

Preliminary Research on the Efficacy of CATTS

Actually, CATTS is just a kitten. Hence, many modifications are yet to be implemented through research and demonstration projects. Obviously, the most pertinent question is whether the system does in fact have the capability for developing and modifying specific teaching behaviors. It can be reported with confidence that the CATTS system eliminates the tedious, long hours of coding, summarizing, and analyzing interaction data which is associated with traditional approaches to classroom interaction analysis. However, the efficacy of CATTS is quite another matter. Four studies were completed in the teacher training laboratory at the University of Michigan during 1969--all of which

attempted to demonstrate the effects of the systems. Schmitt (1969) and Kreider (1969) demonstrated the impact of CATTS in training college juniors, who were majoring in mental retardation, to increase the use of specific categories of behavior in two content areas, as measured by the Flanders Interaction Analysis System (1964). Schmitt focused on increasing the trainees' uses of broad questioning behavior and reducing the frequency of binary questions in a class for EMR's. Kreider, on the other hand, attempted to increase the trainees'uses of pupil ideas in a class for EMR pupils. His results offered only limited support for CATTS training effects. However, Schmitt's results were very encouraging. As hypothesized, the results indicated that CATTS trainees spent significantly more time asking broad questions than did control trainees. Descriptive analysis also revealed a positive relationship between time teachers spent in asking broad questions and time spent by retarded pupils in producing broad responses. This study pointed to a number of complex interaction effects and problems of transfer which must be explored further.

Weaver (1969), a third member of the CATTS group, studied the effects of expectations about EMR children on the ability to modify trainees' use of pupil ideas under three feedback conditions--CATTS immediate feedback, delayed photographic presentations of the CATTS display functions, and verbal, impressionistic comments by a supervisor. Again, the results were moderately encouraging but did not unequivocally support the superiority of CATTS immediate feedback. Trainees who received CATTS feedback did demonstrate greater gains when compared to trainees who received delayed feedback.

Developments with CATTS have been so rapid that systematic, controlled research of each innovation appears to be an unrealistic goal if we hope to achieve an operational system within the next few years. The system's heuristic potential for significant contributions to many different areas in teacher education and educational research has already been demonstrated. For example, another member of the Michigan research group, Harolyn VanEvery (1971), demonstrated in a fourth thesis the feasibility of bringing CATTS out of the laboratory and into a practicum environment. In this study, VanEvery brought CATTS into a speech clinic through a remote terminal telephone line hook-up with the laboratory. Observations of therapists in training were being coded in the clinic and transmitted to our laboratory. Since a specific pattern of training deemed appropriate was to be emulated by the trainees, the nature of the feedback informed them when they were "in" or "out" of the pattern as prescribed by senior clinicians. The feedback in this study was provided by our computer which controlled an event recorder which, in turn, traced a pattern on a moving belt of paper within the clinical setting.

Results of the VanEvery study clearly support the adoption of CATTS into a training program. Analysis of the data revealed significant differences between CATTS and no CATTS trainees on the increased use of social reinforcement (SR) patterns while conducting therapy together with a significantly increased reinforcement response ratio. This rise in the use of SR modeling patterns increased for all trainees during the weeks of data collection, "but CATTS trainees increased sig-

nificantly more than no CATTS trainees."

The results of VanEvery's work demonstrated the feasibility of eventually moving CATTS into public school classrooms for in situ training opportunities. The present researchers have derived considerable knowledge from the aforementioned dissertations about how to improve CATTS so as to assure more direct results.

In still another dissertation, Diane Greenough Dolly (in preparation) is utilizing the CATTS system's versatility in high speed synthesis and analysis of chains of interaction data between trainable retarded (TMR) children and their mothers during teaching sessions. The unique elements of this extension of CATTS work have important implications for learning more about interaction patterns in teaching situations. Dolly has developed two systems of observation--one involving verbal and the other nonverbal categories. She records her mother-child teaching sessions on video tapes. Two observers seated at coding terminals in the laboratory then proceed to record simultaneously the interactions from the tape. Dolly has predicted that specific patterns of sequential behavior in the mother-child interactions will emerge from her analysis, and that patterns will differ when mothers teach their normal children. Using a program developed by Collet and Semmel (1970), the computer searches for these patterns and, if appropriate, verifies the predictions.

CHAPTER 2

SYSTEM CONFIGURATION

The physical configuration of CATTS consists of three interdependent stations: Teaching Station, Observation-Coding Station, and Analysis-Encoding Station. Figure 2.1 (p. 14) illustrates this configuration with a schematic diagram of the prototype CATTS installation which was developed by Semmel and his associates at the Center for Research on Language and Language Behavior at the University of Michigan. Figure 2.1 also illustrates the closed-loop concept of feedback previously discussed.

Teaching Station

The Teaching Station consists of a classroom or a similar room which can accommodate a feedback device, with space for visual observation and for machine-coding of observed classroom behavioral events. The feedback device is placed so that the teacher can use the information contained as required, with no interference in classroom control. The display device may be either visual or auditory in nature, controlled either directly by the computer or indirectly through external display hardware.

Current applications of visual displays for the Teaching Station vary from a closed-circuit televised image of a cathode-ray tube (CRT) display whose image is under direct computer control to an external device which displays feedback information by changing light patterns or x-y chart recordings. As an alternative system, feedback also may be provided to the teacher by the application of an auditory device which can receive auditory feedback messages from a remote location.

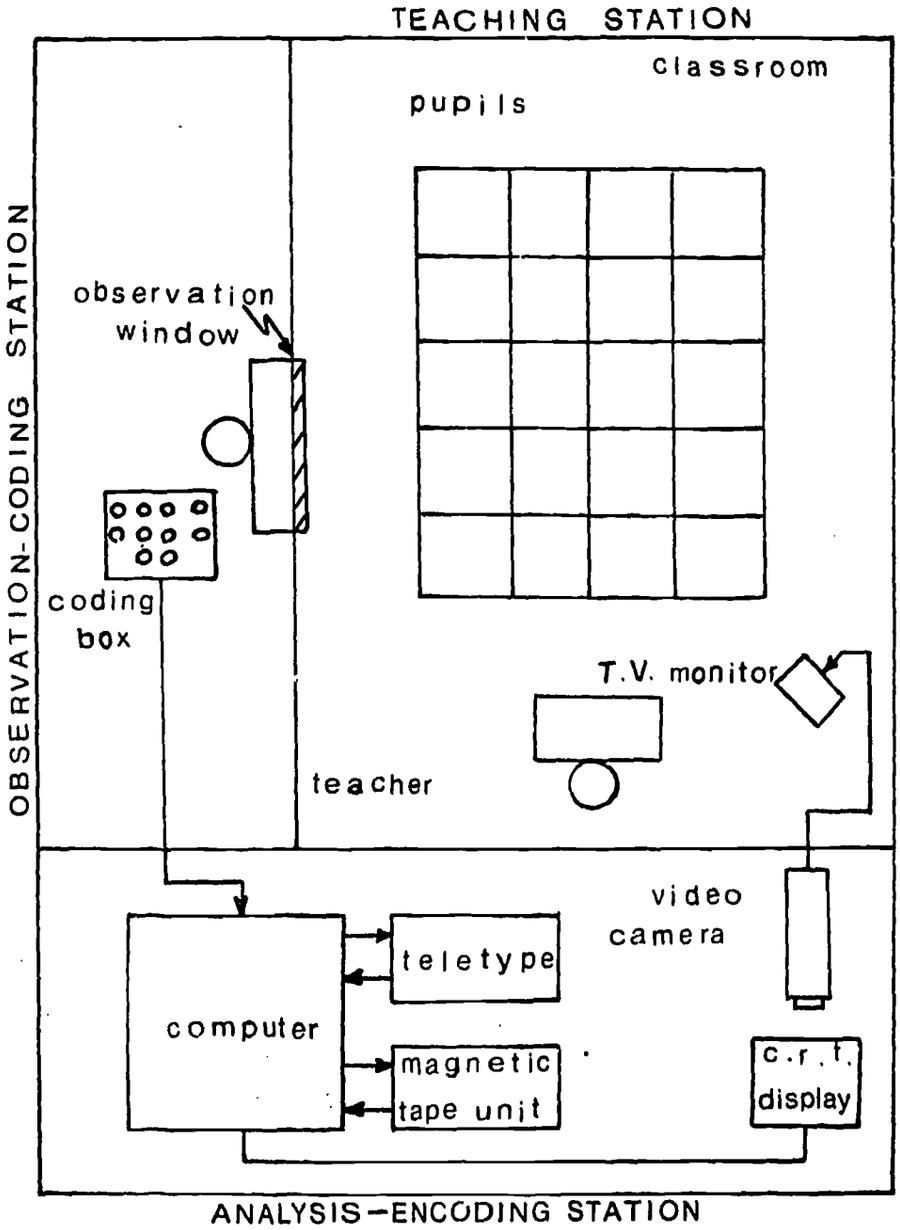


Fig. 2.1 Schematic diagram of present conceptual arrangement of CATTS stations.

Observation-Coding Station

The Observation-Coding Station provides the link between the events occurring in the classroom and the computer analysis of these events. The station must be constructed so that a trained observer can, by direct visual and auditory observation, code what is happening in the classroom according to an appropriate behavior classification observation coding system. Visual observation may take place within the classroom itself, within an observation booth adjoining the classroom, or by a closed-circuit television connection. At present, the mechanical interface with which the observer codes his observations consists of an input terminal on which ten mechanical pushbuttons are mounted. These buttons, in turn, interface directly with the computer.

The utilization of a touch-tone (TT) telephone will permit the observer-coder to input data which is transmitted over telephone lines into a TT data set and from there directly into the computer. The TT telephone interface allows direct real-time observation for remote observation coding in community classrooms. Projected system development will, in addition, permit "real-time" feedback to trainees in community practicum settings.

Analysis-Encoding Station

The Analysis-Encoding Station contains a small computer (e.g., PDP-4, PDP-12) and the associated computing hardware required for the on-line processing of the coded data which is gathered and transmitted from the Observation-Coding Station. In addition to processing the incoming data, the computer system controls the feedback display devices used in the

Teaching Station and also provides for hard copy print-out, storage, and transfer of the analyzed data. It is at this station that the trainer or experimenter initiates the computer program options available in CATTS. The teleprinter console, through software program control, allows the operator to select any specific CATTS program or option that will satisfy the objectives served by the system.

The selection of the mode and content of feedback to the teacher trainee at the Teaching Station is also initiated from the console. If a CRT display is chosen as the method for feedback, the operator determines the content of the display by assigning the incoming data, by code, to different computational functions for the computer to calculate and display as feedback. The nature of the display is also selected from the console, which allows feedback information to be presented either in alpha-numeric and/or graphic form. Paper and magnetic tape storage of collected and analyzed data is then summoned from the console for long- or short-term storage, with the option to recall data for a real-time replay for further analysis.

The associated teleprinter, which also serves as the communications link to the CATTS program, provides hard-copy data print-outs for inspection during the data gathering stage. These print-outs provide such information as the event times of the coded tallies together with the actual feedback functions being displayed to the teacher. At the end of a real-time session, a selection of various data summary printouts is available as a further option. Calling these options re-analyzes the raw data and prints out various descriptive statistical summaries. Some of these available options are: observer tally sums minute-by-minute,

cumulative minute-by-minute totals of recorded tallies, and matrix development of observed coded categories and subcategories.

In summary, it readily can be seen that the translation of the closed-loop cybernetic principle is achieved through a prototype CATTS by using a human observer-coder as the interface between the teachers and the computer. Behavior observed in the Observation-Coding Station is coded and transmitted, in real-time, directly to the computer in the Analysis-Encoding Station. The computer summarizes, analyzes, stores, and feeds back relevant information in real-time visual form to the teacher in the classroom. Simultaneously, the system can provide a comprehensive print-out of the analysis of all variables used in the observation of the classroom transactions.

CHAPTER 3

HARDWARE COMPONENTS OF THE PROTOTYPE CATTS

The physical hardware configuration of CATTS, as originally developed at the University of Michigan, evolves around a Digital Equipment Corporation model PDP-4 general purpose digital computer, which contains a 16-channel multiplexed analog-to-digital (A/D) converter input unit, three digital-to-analog (D/A) output channels, an 18-bit relay buffer connection, and a console teletype-teleprinter. In addition to the main computer configuration, outside support hardware such as a ten pushbutton input terminal and an oscilloscope CRT display are used as external input and output units.

The CATTS system, in its prototype physical form, depends upon certain hardware requirements for maximum application of its capabilities. Three general areas of hardware components are required: Input Coding Components, Computer Components, and Output Components. Figure 3.1 (p. 19) gives a graphic display of the interrelationship of these three component systems.

Input Coding Components

For input into the system, some type of mechanical coding interface is required for use by the observer in the Observation-Coding Station. The prototype system used a pushbutton device mounted in a box containing ten simple contact closure buttons used for coding input, with two small colored signal lights mounted on the box to indicate the coding sequence status (to the observer). These ten pushbutton switches are connected through a direct current voltage source and

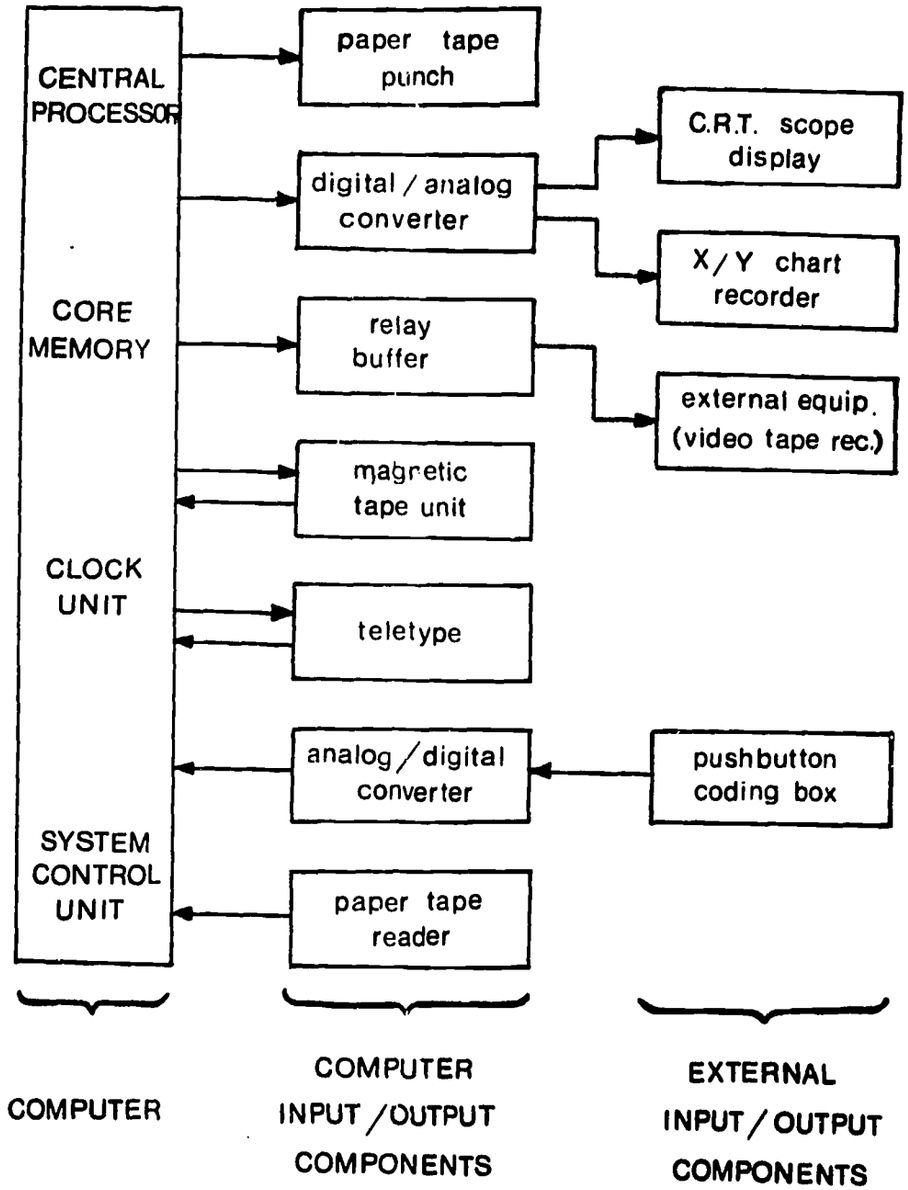


Fig. 3.1 Schematic diagram of Hardware component systems used in CATTS.

terminate directly into ten A/D converter inputs on the computer. The coding status lights on the button box are controlled by a relay buffer on the computer and an outside voltage source. The input portion of Figure 3.2 (p. 21) schematically represents the physical connection of the pushbutton and coder status lights on the computer.

Depending upon the coding system used, the signal lights, through program control, indicate to the observer the current status of the input acceptance system. For example, in the prototype operating program, if a two-level subscripted coding system is employed by the observer to record the classroom events, the first light is illuminated when the first button is selected and pressed, indicating to the observer that a main category has been selected and the button that is pressed will be accepted by the computer as a subcategory choice. When the second button is pushed for the subcategory choice, the status lights are extinguished indicating to the observer that the third subscript would now be ready to be accepted into the computer. Additional subcategories would be coded in an extension of the same sequence. Again, when the third button press is entered, both lights will go off, indicating that the system will now accept a new main category coded selection.

As an alternative to a direct connection of a button box into the computer, a direct distance dial TT telephone may be employed as an input device.

Computer Components

The computer hardware configuration for CATTS can be segmented into the following three groups: (a) input conversion hardware, (b) internal processor hardware, and (c) output conversion hardware. The schematic

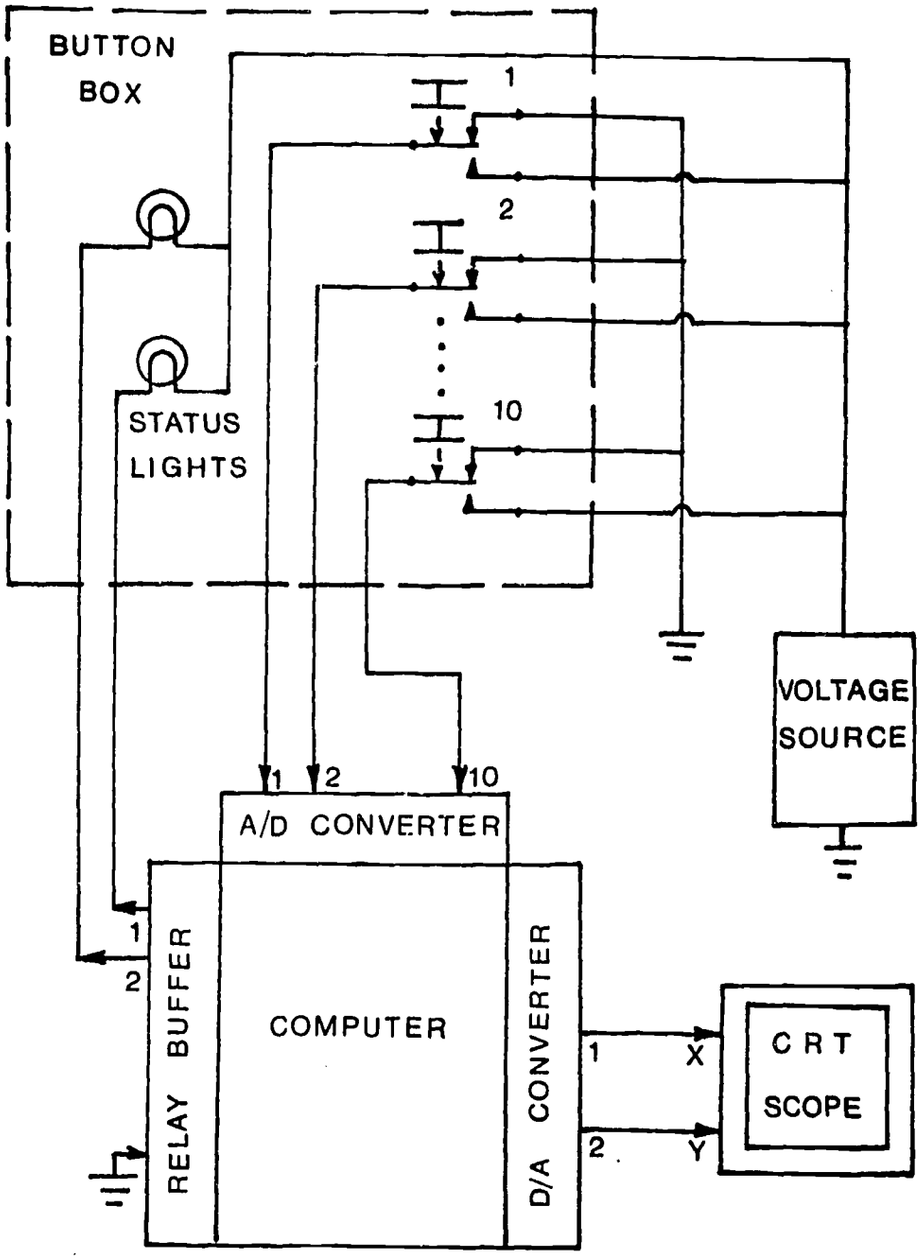


Fig. 3.2 Schematic diagram of physical connection of prototype CATTS input and output hardware.

interconnections between these components are illustrated within the center section of Figure 3.2.

Input conversion hardware. The input configuration for the CATTS prototype computing processor system, excluding the external coding device, is comprised of the following units: an A/D converter for the conversion of raw input data, a DATA-PHONE input buffer for direct distance telephone input, a punch-paper tape reader and a magnetic data type recorder for re-entry of previously coded and analyzed data for further analysis, and a teleprinter for program selection and control. These units, either independently or in conjunction with each other, accept and prepare the incoming data for further analysis by the computer's internal processor.

Internal processor hardware. The computer's computational unit contains basic internal hardware instructions which permit, through software programming, the control and execution of the many options that are available in CATTS. These options range from basic arithmetic computations for data reduction and feedback functions to controlling and monitoring the associated input/output (I/O) hardware. The computer's capacity to initiate and maintain CRT displays simultaneously with data input and computation functions is also an important requirement. An additional internal requirement of the central processor is the use of a real-time clock. This clock is required in order to provide timing functions essential to many of the basic CATTS programming options. Some of these options are: the calculation and storage of the time between incoming observer-coded tallies, the time base for the A/D converter sampling, and the time base for the CRT display control units.

Output conversion hardware. The output configuration consists of a CRT display control, a D/A converter for voltage control of external display devices such as chart recorders and/or x-y plotters, and a relay buffer for control of feedback status light signals.

A Tektronix model 503 x-y oscilloscope was employed as the original CRT display in the prototype CATTS configuration. In this connection the display beam is attached to two channels of the D/A output and controlled through software programming. The oscilloscope connection to the D/A converter is illustrated in Figure 3.2. An attachment of a CRT display unit, designed to accompany a specific computer, is more efficient in operation, in that many of the display functions previously served by software programming can be taken over by pre-wired hardware and instruction circuits. A built-in display unit, therefore, permits greater flexibility of CATTS display programming.

The data storage devices include a paper-tape punch and a magnetic tape data recorder which stores the data for either transfer to another computer system or re-entry into the same CATTS program for replay and analysis.

Output Components

Output component hardware for CATTS is selected to satisfy three methods of information transfer. A visual graphic display, such as produced by a CRT, chart recorder, and/or plotter, or any external hardware device which is required for the active and immediate feedback capabilities essential to CATTS. A teleprinter is necessary for print-

ing hard copy data which accumulates over the course of real-time data collection, and also serves as the main interface between the operator and programming options available to the CATTS system. Finally, external storage in the form of magnetic tape, paper tape, or some other storage device is necessary for transfer to larger computing systems for extensive data analysis and for providing available data for replay into the CATTS system.

CHAPTER 4

MAIN SYSTEM PROGRAM

As described in the previous section, the basic control element in the CATTS system is a small digital computer (i.e., PDP-4). This machine has been programmed to provide three basic services to the CATTS system: (a) system timing, (b) information collection, and (c) feedback of results. In the discussion which follows, the programming considerations of the system will be described without reference to the particular computer in use, except when necessary for clarity.

The structure of CATTS Software: The Monitor

The CATTS program is built in modular form. All functional parts are much like complete computer programs in themselves. In order to accomplish the required system objectives, these parts or modules are summoned by a supervisory program, or monitor, to perform their particular functions. The monitor is responsible for allocating the computer's resources in a manner which allows all system functions to be performed at the proper time. This process assures that the computer is exercising continuous control over the system.

To the observer, all operations (such as timing, data input, coder feedback, teacher feedback, hard-copy listing) appear to be occurring simultaneously. However, the computer can perform only one operation at any time. The monitor switches from one operation to another in response to external input conditions or internal program-generated conditions, thereby effectively time-sharing the system resources to satisfy the system requirements.

In describing the action of the monitor, it is instructive to view

the system operations as a group of tasks which must be performed at various times. Sets of foreground and background tasks can be distinguished from one another. Operations fall into one of the two groups of tasks, depending upon the speed required of the computer to respond to events occurring in the system (i,e., some tasks can afford to wait longer than others to be performed). An example of a foreground task is the reading of a signal from the coding terminal; the computer must not delay in reading the signal or else data may be lost. A background task might be the assembling of lines of characters to be printed on the teleprinter if the printing option is selected. If a line is being assembled and the button box terminal requires service, the monitor will determine that printing must wait and temporarily will suspend that task.

Within the background/foreground groups, the tasks are further arranged according to priority, i.e., based on the maximum allowable latency from request for service until response by the program. The monitor receives requests for computer response as they are generated by I/O equipment, the real-time clock, or internal program conditions. If only one request is received, and no higher-priority task is executing, the monitor initiates a particular task to service that request. If several requests are received simultaneously, their relative priority is determined and a task initiated to service the highest-priority request, while the others remain pending in what might be termed a job queue. If a new request has higher priority than a currently executing task, the current task is suspended and its identifying information placed on a task stack, while another task is initiated to service the

new request. The priority of a particular task may change dynamically in response to system conditions. The monitor is programmed to allow certain short lower-priority tasks to run to completion even if higher-priority requests are generated while these tasks are executing. The execution time of these tasks is so small that system performance is not impaired; and, in fact, system overhead ("extra" work by the monitor) is reduced because the short task does not have to enter the job stack. Foreground tasks may be considered as "monitor routines" and part of the monitor itself, since they exercise much control over the system and are responsible for initiating some lower priority tasks.

Figure 4.1 (p. 28) is a simplified block diagram of the software system as it is configured for real-time operation. Auxiliary functions such as program initialization before coding, and data output and analysis after the coding sessions are not shown, since these are not part of the real-time control function. The levels of the diagram may be thought of as priority levels of operation, since tasks executing on levels four and five may be interrupted by the occurrence of higher-level events within the system. Levels two and three contain important short tasks that take action on system requests for service and are not interruptable (i.e., once initiated, they run to completion, usually in 200 microseconds or less). Level one receives all requests for service (program interrupts from devices in the system) and dispatches control to the appropriate service program on level two. The discussion in the following paragraphs explains in general terms the actions taken in each of the levels.

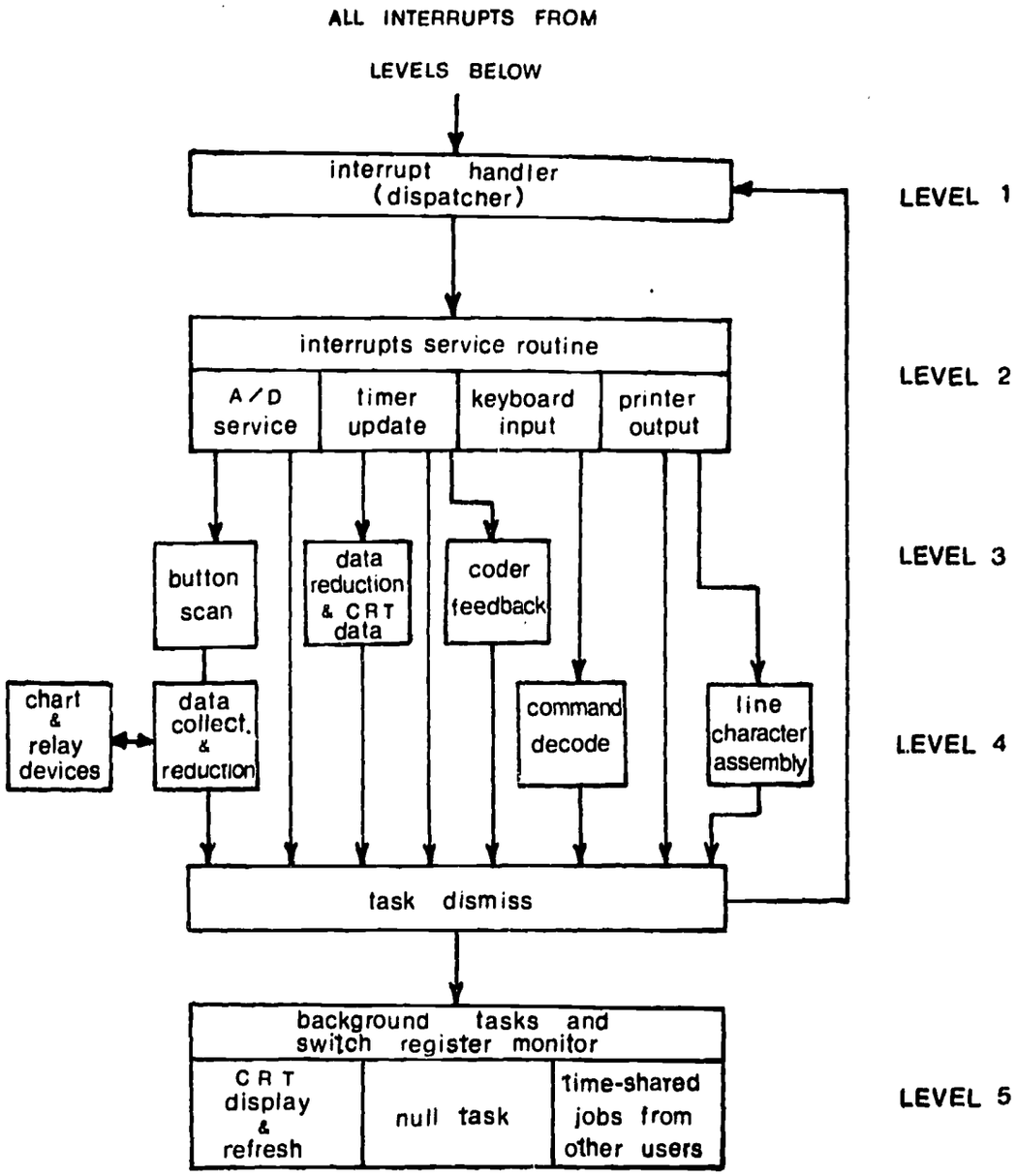


Fig. 4.1 Real-time software block diagrams.

Levels one, two, and three are considered as the monitor proper. Level one is the interrupt handler. When the interrupt system is enabled (i.e., when a task on level four or five is currently executing), and an interrupt request enters the computer hardware, execution of the current task is suspended and the central processor switches to the interrupt handling routine. The interrupt system is automatically disabled by this action. The interrupt handler saves the contents of the active processor registers and the link information (i.e., the address of the next instruction to be executed in the interrupted task) in a task stack (a last-in-first-out list in core storage). It then identifies the source of the interrupt by testing the status of all devices connected to the interrupt system in their respective orders of priority. As soon as the source of the interrupt is determined, control is dispatched to the appropriate service routine in level two. If several requests occur simultaneously, the handler always transfers control to the routine of the highest priority. (In the diagram, A/D service, timer update, keyboard input, and printer output is the priority order from high to low). The other requests remain pending.

Level two tasks are responsible for determining the conditions causing the particular interrupt and then taking appropriate action. If this action is simple and short, it is performed and the interrupt is dismissed as described below. If more lengthy processing is necessary, a task on level three or four is initiated.

An interrupt or a task is dismissed by restoring the active processor registers from the stack and then using the link information to return to the interrupted task. The interrupt dismissal routine is

entered at the completion of tasks on levels two, three, or four. If the interrupt system has been disabled at this time, it is turned back on. Control is dispatched to the most recently interrupted task unless interrupts on the A/D and timer have caused tasks servicing the keyboard and printer to propagate toward the bottom of the stack (this dynamic priority juggling is a complex process when the system is heavily loaded and a discussion of it is beyond the scope of this manual). Eventually, control is returned to the background tasks on level five when all interrupts have been serviced.

Tasks on level three carry out further service to a system request. These are not lengthy tasks, but they are usually longer than level two tasks. Although these tasks are not interruptable by the hardware, they may at times monitor system status and produce a "software interrupt." This is exactly equivalent to a hardware interrupt, except that certain tasks in the task stack may be rearranged according to special system conditions. This type of action almost never happens in the basic CATTS system but may occur in a more heavily loaded system when other users are sharing the computer.

Tasks on level four are initiated by request service tasks when more involved computation is required to complete a certain function but the results are not needed for immediate feedback purposes. These tasks are interruptable.

Tasks on level five run continuously unless interrupted. The program executes on levels one through four less than ten percent of the time, so a great amount of computer time is available in level five. In

CATTS, this time is used for refreshing displays and monitoring the console switch register (which cannot cause an interrupt on its own). Although displaying on the CRT is important because flicker is to be avoided, it should not be confused with high-priority tasks which need fast service. In actuality, time spent in higher levels is not noticeable on the CRT screen since these higher-level tasks are completed rapidly. If the display option is not selected, the "null task" is executed in level five. This consists only of monitoring the switch register. It is also possible to allocate some time available in level five for use by tasks not directly involved with real-time CATTS operations. Examples of such tasks are the transfer of data from paper tape to magnetic tape and reduction of such data with results written on devices other than the teleprinter (since the teletype is usually used by CATTS).

System Timing Functions

In order to provide a time base for the operation of the system, the monitor makes use of the computer's internal real-time clock. The clock is actually a peripheral device which can increment the contents of a memory location and then interrupt the currently executing program when the value in this location reaches zero. Included in the monitor is the timer update routine, which is initiated whenever the program is interrupted by the clock.

Figure 4.2 (p. 34) is the flow diagram of the timer update task. It is assumed that the initialization routine (described below) has initialized the clock location (in the PDP-4, this is core location 7)

and has turned on the clock. Basically, the routine works by re-initializing the clock location, restarting the clock, and then updating any cumulative and interval timers which are kept. In the PDP-4, the real-time clock increments location 7 at the rate of 60 Hertz. In the CATTS system, time is kept to the nearest tenth of a second. Therefore, the initializing value which is placed in location 7 is -6; this location must increment six times before a clock interrupt occurs. If re-initialized to this value each time the interrupt occurs, the interrupts will be spaced at one-tenth second intervals.

Several timers are maintained by the timing task and are updated whenever a clock interrupt occurs. These timers are simply core memory locations holding numbers representing minutes, seconds, or tenths of seconds.

The timers are:

1. Elapsed time since the beginning of the coding session, in minutes, seconds, and tenths.

2. Elapsed time since the initiation of the previous tally, in tenths.

3. Elapsed time since the end (completion of last button press) of the previous tally in tenths.

4. Elapsed time since the previous point was added to the feedback display vectors, in tenths.

All timers are incremented by one-tenth second on each pass through the task.

Timer 1 is read by the signal input task whenever a tally is completed; the time is stored with the tally as data. This puts all tallies in sequence on a time line. This data can be used to reconstruct

the series of classroom events for later analysis. In addition, each time this timer is incremented, it is compared with the time limit set for the coding session. When the two values are equal, the session termination task is initiated, ignoring all further data input, finishing any printing in progress, and allowing the subsequent output of data. Timer 2, after being incremented, is compared with a value entered as program initialization data. This value is the time, in tenths of seconds, which will be allowed before the coder is warned to initiate another tally. When the contents of timer 2 equal this value, the timing task must turn on the lights on the coding box and reset the timer to zero. In addition, if the "automatic button press" option is selected, the previous tally and the current value of timer 1 are entered as data. Timer 2 is also set to zero when the signal input routine detects the start of a tally. Timer 3 may be used similarly to timer 2; it is not presently being used in the basic CATTS. Timer 4, after being incremented, is compared with a number, n, described in the "feedback" section below. When equal, a data reduction task which accumulates various results is initiated, adding another point to each of the four feedback display vectors. (n is chosen so that the CRT display image will just fill the screen at the end of the coding session.)

Program Initialization

Before each training session, the operator is required to initialize the computer program, setting required parameters and selecting system options which will be used. Basically, four items require initialization: (a) feedback function specification, (b) time-line selection,

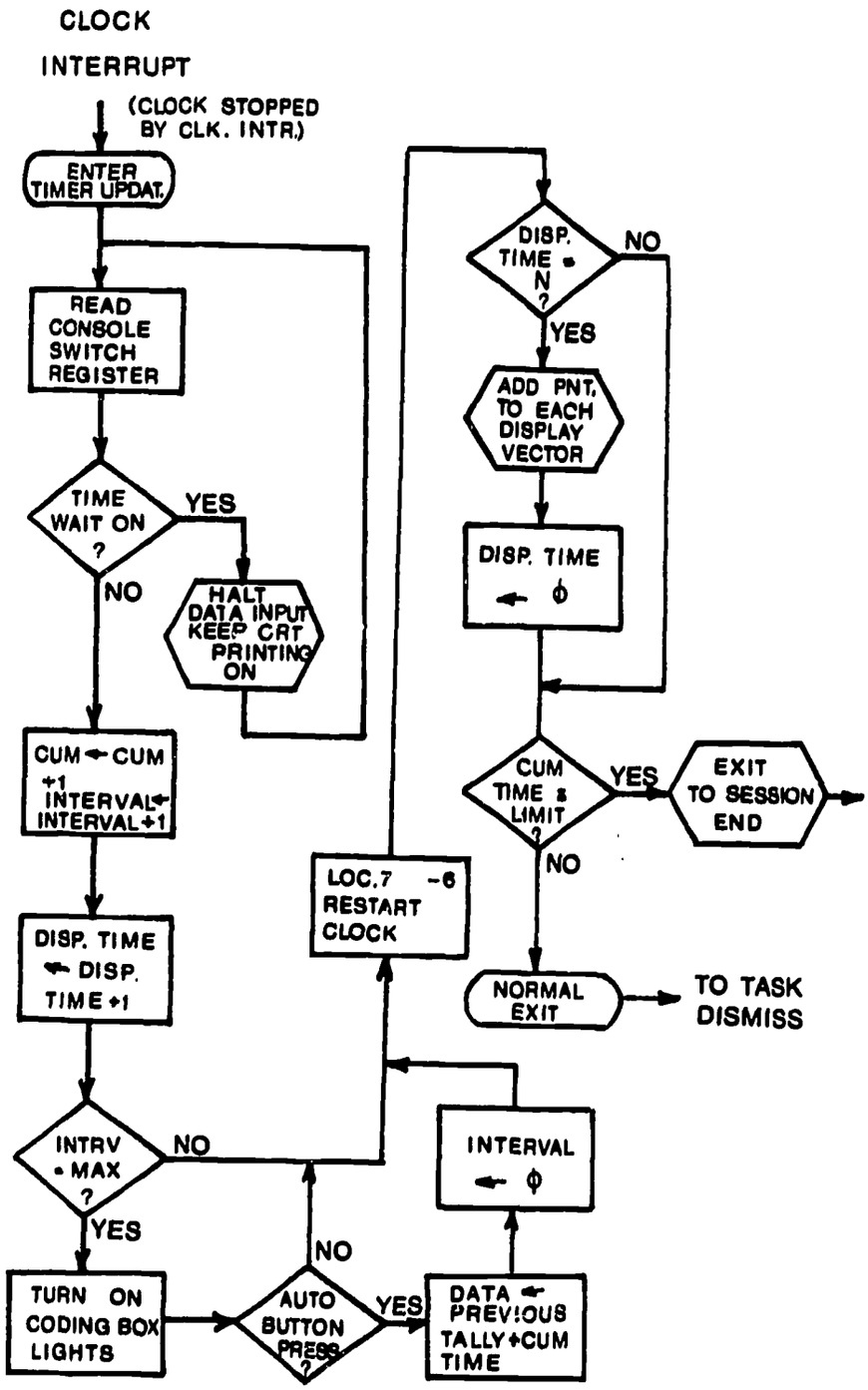


Fig. 4.2 Timer update flow diagram.

(c) feedback display selection, and (d) print-out summary selection. Initialization is accomplished by starting the program, setting various switches on the computer console switch register, and responding, via the teletype keyboard, to instructions and questions printed on the teleprinter.

The first section of the initialization routine instructs the operator to turn on all power supplies and I/O equipment. After this, the question "Standard Coding?" is printed. If the operator answers "yes," predefined functions, parameters, and options are supplied by the program. These have been written into the program as internal data and are provided merely for convenience. If the standard items are not desired, initialization proceeds as below.

First, the functions used in the feedback display subroutines are generated. Up to two functions may be used in the prototype system. Each one is expressed as a numerator and a denominator. The numerator or denominator consists of a sum of the counts of certain tallies and/or counts in certain main categories across all subcategories which come under that main category. At times during the course of the coding, the functions are evaluated by computing the indicated sums in the numerator and denominator (over all previous tallies, for the cumulative percentage function), dividing, and then converting the resulting fraction to percentage form. The numerators and denominators are specified by typing in the codes for the desired tallies, first all numerator codes and then all denominator codes for each function. This information is stored as data for the program and later appears in the header of the output-data tape for later reference.

The operator then specifies the number of minutes the coding session is to last. This number will be used by the program to determine how often the display functions will be evaluated. By adjusting the frequency of evaluation, the displayed graphs of the functions will just fill the CRT screen at the end of the session. The program also uses this number to signal the end of the coding session, after which no more coding input will be accepted.

Two types of feedback may be produced. Feedback to the coder is in the form of program-controlled lights on the coding box. The operator must furnish the number of seconds which will be allowed between tallies before the coder is signalled that another tally should be entered. This number also may be used in the "automatic button-press" option. If the coder does not initiate a tally in the specified time limit, he is signalled and the previous tally is also entered into the data buffer. This action is necessary to preserve the characteristics of the feedback display to the teacher. It relieves the coder of continuously coding even though classroom action is not changing.

The CRT feedback display is controlled from the computer console switch register. By setting some of these switches in certain configuration, either one or both function plots can be displayed with either cumulative or moving-window percentages, and with or without numerical labeling. Although the switches should be set as desired during initialization, they may be changed at any time, allowing changes in the display during the session.

Printing of data or results can be carried on simultaneously with

other system operations. One of three types of print-outs may be selected: (a) print-out raw data (tallies and associated time of occurrence), (b) minute cumulative totals of the number of tallies taken in each main category, or (c) the percentage values of the four display functions with associated times. This latter listing can be produced even though the display is not selected.

Signal Input Software

The signals from the observer coding box reach the computer (in this case, the prototype PDP-4 configuration at the University of Michigan) in the form of voltages on the ten input lines to the analog-to-digital (A/D) converter. A voltage on one of these lines signifies that the associated button is being pressed. If a button is not being pressed, the line is at ground potential (zero volts). Each line is sampled every .01 second (100 times per second) and the voltages processed as described below.

The heart of the button input system is the A/D line scanning service subroutine. This subroutine is responsible for servicing all program interrupts generated by the A/D converter. The A/D converter contains a 1KHz clock and a multiplex address register. The clock, when started by the program, initiates conversions on the line currently addressed by the multiplex address register. When the conversion is complete (the line sampled and the voltage digitized), the program is interrupted. The execution of the current program is stopped and the computer switches to the interrupt handler. The interrupt handler identifies the source of the interrupt and dispatches the appropriate service subroutine.

Both the interrupt handler and the service subroutines are considered as monitor, or executive, subroutines. A service subroutine runs as a task within the monitor, but it is of higher priority than tasks which are not general interrupt service subroutines.

The service routine reads the voltage and the line number from which the voltage was taken (this will always be line zero, from button #1, upon entry to the routine). A counter is incremented and tested. If the counter has reached 10 (to count the interval of one one-hundredth second using the 1KHz clock) the voltage is processed and the other button lines sampled and processed. If the counter has not reached 10, the routine is exited and the sampled voltage ignored. Control is returned to the interrupted program.

When voltages are to be processed, the action taken for each line depends upon the value (magnitude) of the voltage and the history of the signal based upon previous samples. The routine is responsible for (a) filtering the input signal to eliminate switch contact bounce, (b) registering the button press as tally data, (c) calling upon other routines to process the tally if the button press completed the tally, and (d) making an appropriate response to the coder (i.e., turning off a signal light to indicate acceptance of a button press). The routine must ignore an unwarranted button press, such as a tally subscript greater than five.

The button scanning routine is responsible for "filtering" the input signals in such a way that contact bounce of the switches does not cause multiple registration of a button press. This software filtering is achieved by setting a counter for each signal line to -3 when the in-

put voltage on that line is greater in magnitude than five volts. If, in a succeeding sampling of the line, the voltage is below five volts, the counter is incremented by one. Eventually, as the line is repeatedly sampled, the count will go to zero if the voltage stays below five volts. When the count drops to zero, the button press is registered. Note that contact bounce may allow the button signal to fluctuate above and below the five-volt threshold. The counter is actually used as a thirty millisecond timer. Experimentation has shown that fluctuations occur much more frequently than 30 milliseconds, so the counter will never "time-out" too soon. Further, all bounce dies out within 30 milliseconds of release of the button. A similar filtering technique could have been used on the leading edge of the signal also, but this was not found necessary. Figure 4.3 (p. 40) shows the relative timing involved.

The coder must be presented with information conveying the status of the tally on which he is presently working. This information should tell the coder (a) whether a button press was accepted as data, (b) how many more button presses must be entered to complete the tally, and (c) when a new tally should be initiated. This last function is performed by the timing routines and will not be discussed in this section.

Information is presented to the coder via the button-box lights. The button signal scanning routine is responsible for much of the control over these lights. When a button is pressed, the routine checks the level of the current tally (i.e., main category, subcategory). If the button press is to select a category, the routine lights two lights on

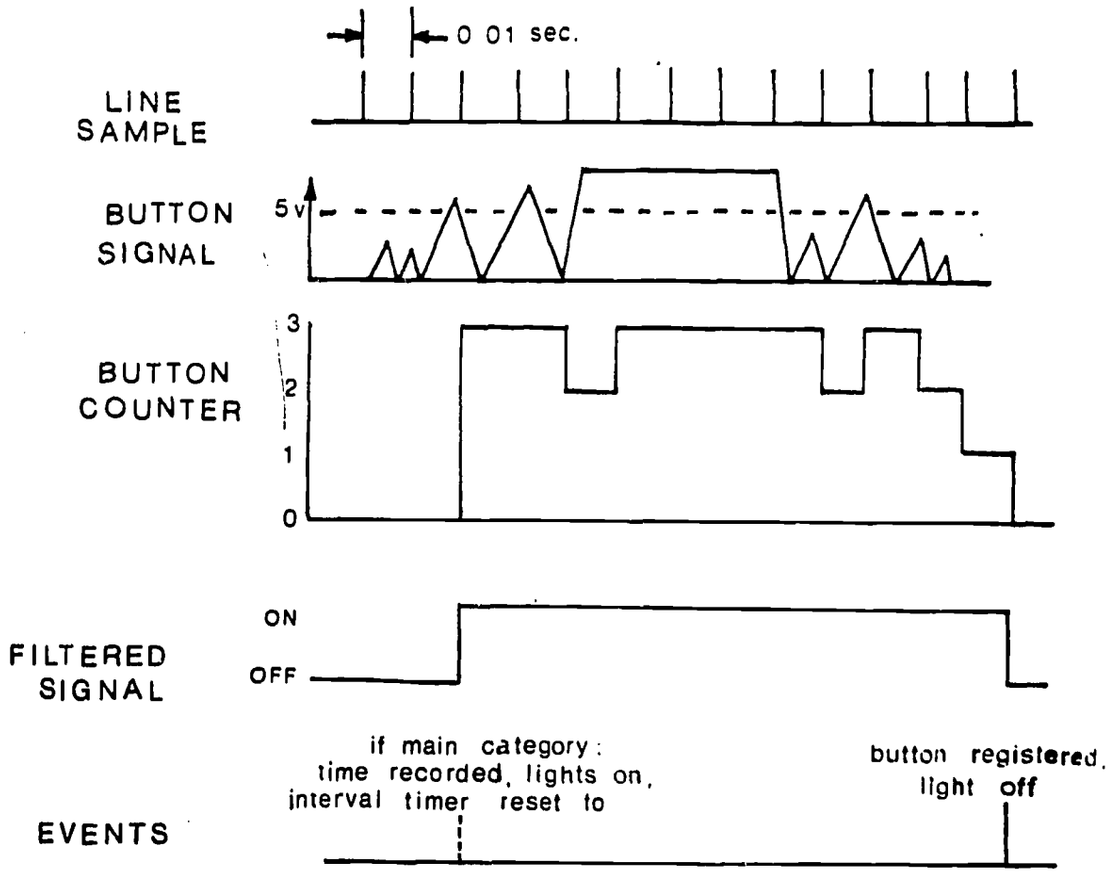


Fig. 4.3 Timing: One input signal line.

the box, reads the current elapsed time since the beginning of the coding session, and notes the button number. When the button is released, the counter for the button is counted down on successive scans of the line. When the counter reaches zero, one light is turned off. The remaining light tells the coder that the next button press will be taken as a subcategory. If the next button pressed is in the range of numbers one to five, the button number is read and when the button is released and counted down, the second light is turned off. The routine then records the category, subcategory, and time of initiation in the data buffer by calling on the subroutine which does this job. It may then call on data reduction subroutines. After this, all lights are out and the service routine is ready for another main category. Since buttons six to ten are invalid as subcategories, these lines are not scanned when the routine is looking for a subcategory, and are thus ignored.

Figures 4.4a and 4.4b (pp. 42 & 43) are flow diagrams of the button service subroutine.

Arithmetic Computations: Data Reduction and Display Points

Two main routines support the reduction of data for use by the CRT feedback display and the hard copy print-out subroutines. One of these routines, called at the completion of each tally, increments the count of the number of times that that tally was entered since the start of the coding session (generating the cumulative counts). The same routine also increments the count of the number of times the tally was entered in the last 60 tallies. These counts are kept in matrices of categories versus subcategories for easy access. The other routine is called by the timing

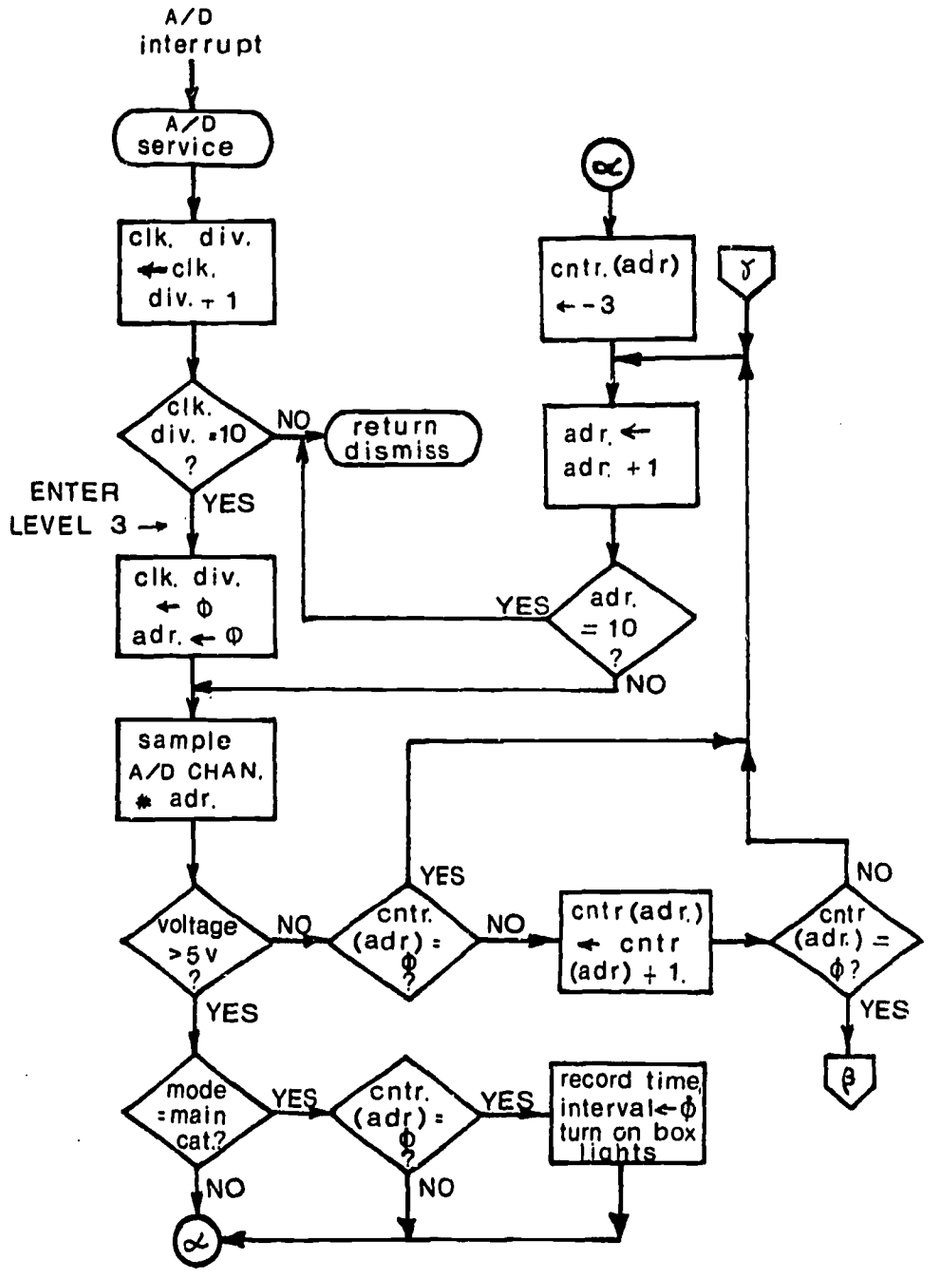


Fig 4.4a Signal input routine (Part A).

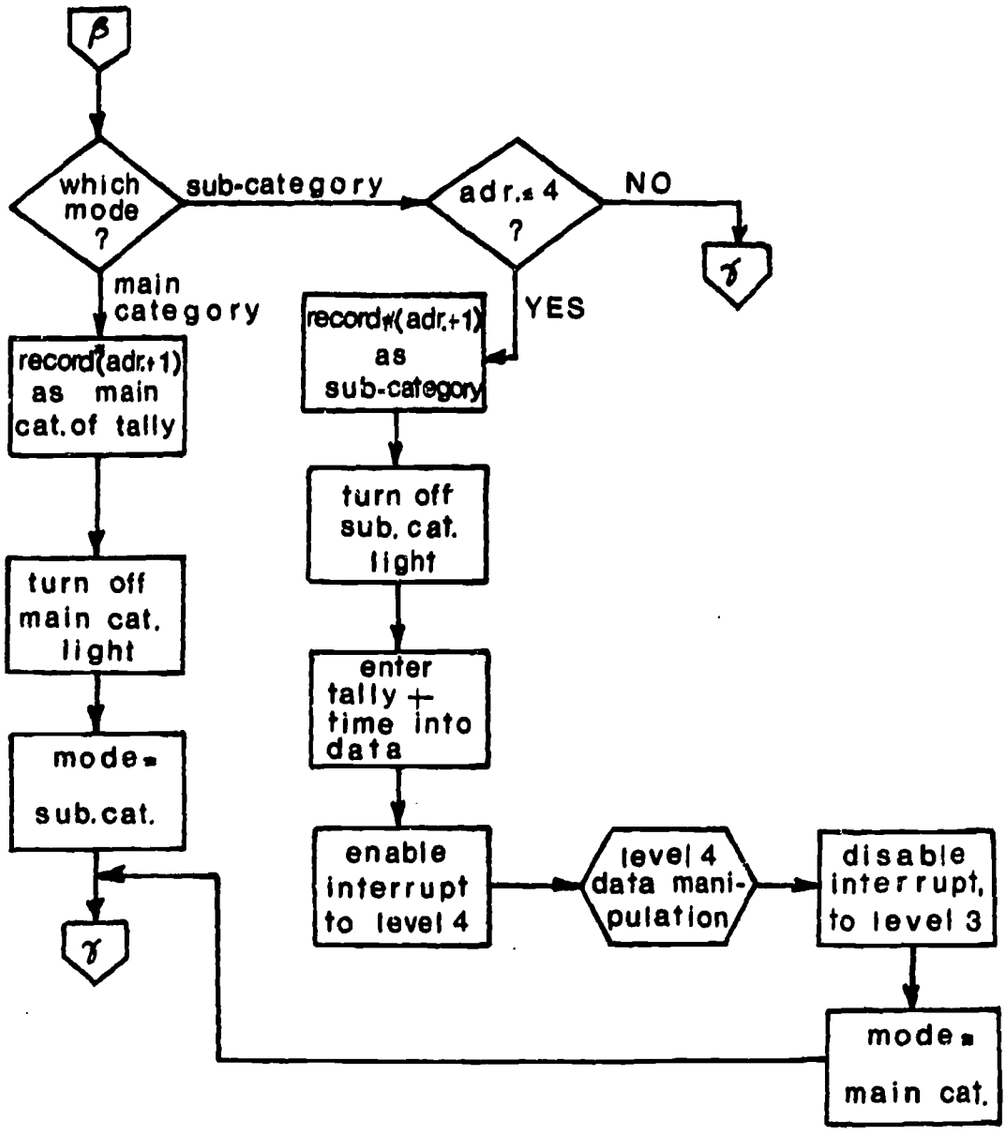


Fig. 4.4b Signal input routine (Part B).

routine; it operates upon the matrices to compute the values of the functions used for the CRT display and print-out.

Figure 4.5 (p. 45) illustrates the matrix format in which the tally counts are stored. Two such matrices are maintained, one for the cumulative counts and one for the moving-window (last 60 tallies) counts. The matrices are accessed by supplying the category number and the subcategory number of the cell desired. Note that the first cell of each row contains the number of counts summed across all subcategories. This call may be accessed as "subcategory zero."

To maintain the cumulative matrix, the subroutine accesses the cell determined by the category and subcategory of the current tally. It then increments this cell and the cell corresponding to the total main category count.

To maintain the moving-window matrix, the subroutine proceeds as it did for the cumulative matrix. However, the subroutine must then obtain the 60th previous tally from the data buffer and use this tally to access the moving-window matrix and to decrement the appropriate counts. Thus, the total number of counts in the moving-window matrix (excluding the first column, which holds redundant information) remains at 60 once the 60th tally is entered. Before the 60th tally, the cumulative and moving-window matrices are identical. Figure 4.6 (p. 46) is a flow diagram of the routine. It assumes that every cell in both matrices has been set to zero before entry of data is begun.

A "display vector" is a one-dimensional array which holds the percentage values of a particular function evaluated at previous points in time. Four of these vectors are maintained by the program during the

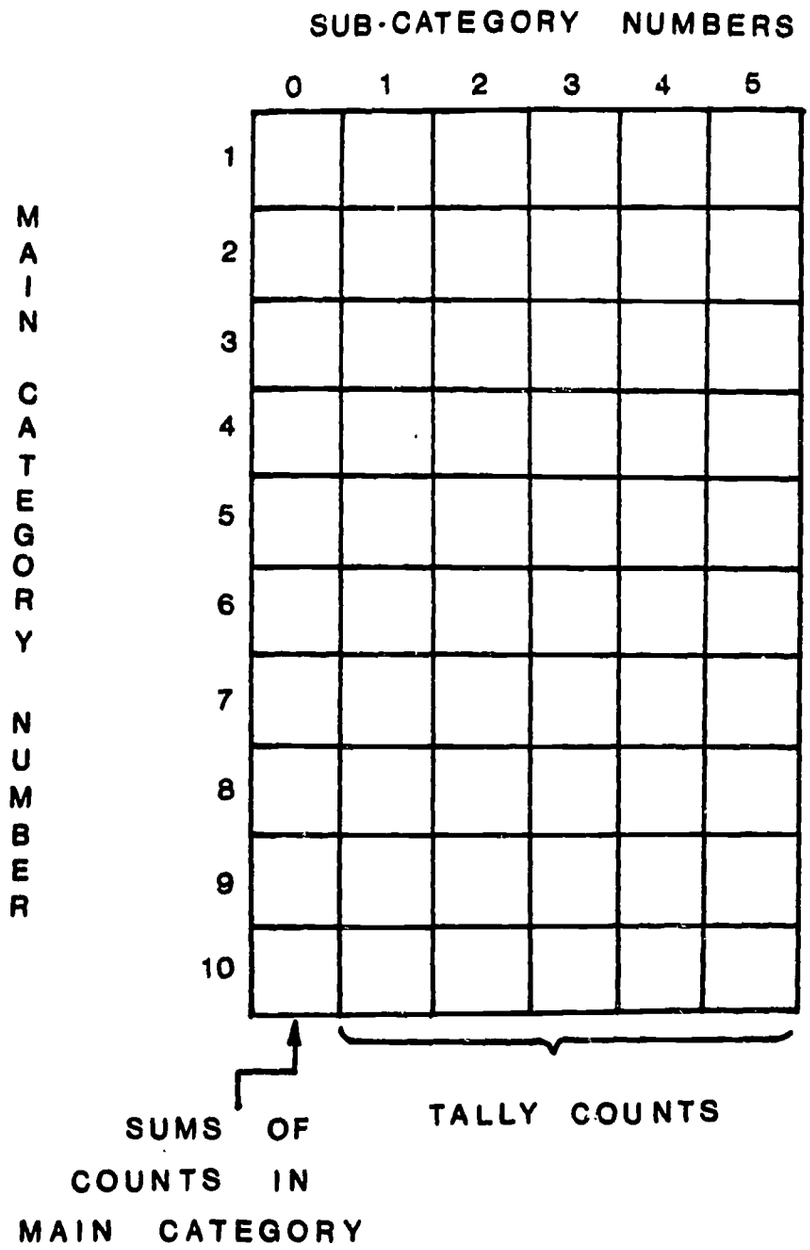
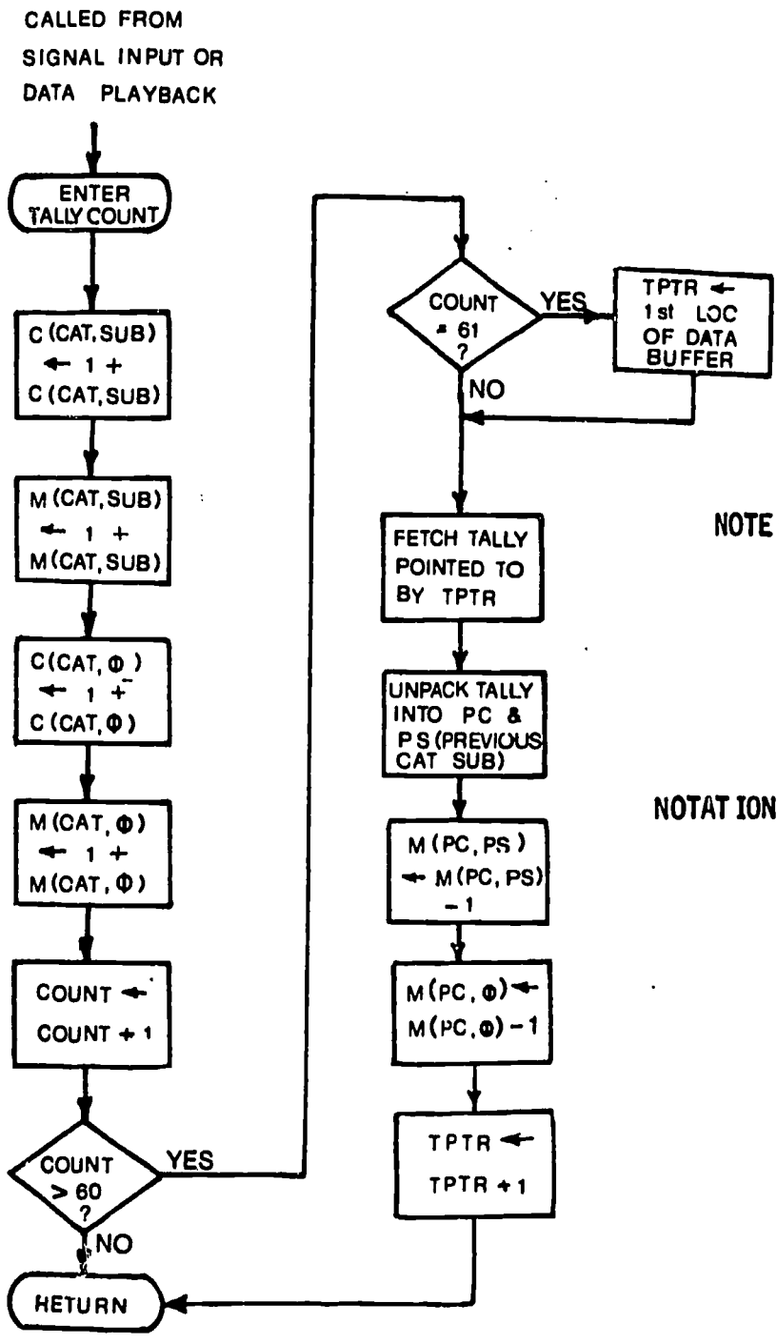


Fig. 4.5 Tally count matrix.



NOTE: C, M, and count must be zeroed by initialization routine.

NOTATION: C: cumulative matrix

M: moving-window matrix

CAT: current tally main category

SUB: current tally sub-category

PC: 60th previous tally main category

PS: 60th previous tally sub-category

TPTR: Pointer indicating location of 60th previous tally

Fig. 4.6 Tally count accumulation.

course of coding. Two of the vectors contain the cumulative and moving-window percentage values of one function; the other two vectors contain the cumulative and moving-window values of the second function.

The routine which maintains these vectors is called from the timing routine every n second, where n is determined by the formula:

n = total expected session time (seconds)/200

The display time line on the screen is two hundred units in length and should be completely filled by the end of the session. This is done by adding a point to the display every n seconds.

Once called, this routine determines the value of each function in three steps: (a) the numerator is determined by summing the counts of all cells (in the appropriate matrix) which are indicated by the numerator initialization data, (b) the denominator is determined in a similar manner, and (c) the numerator is divided by the denominator and the result multiplied by 100%. This value is then entered into the appropriate display vector. This procedure is carried out for both the cumulative and moving-window vectors for each function. Figure 4.7 (p. 48) shows the flow diagram of this process.

Feedback Software

The program must provide the means to output information directly to the teacher and the coder. This information, presented in response to previous data, provides the particular participant with a basis for future actions. Therefore, this information is termed "feedback" and should be presented in real-time (with minimum latency with respect to

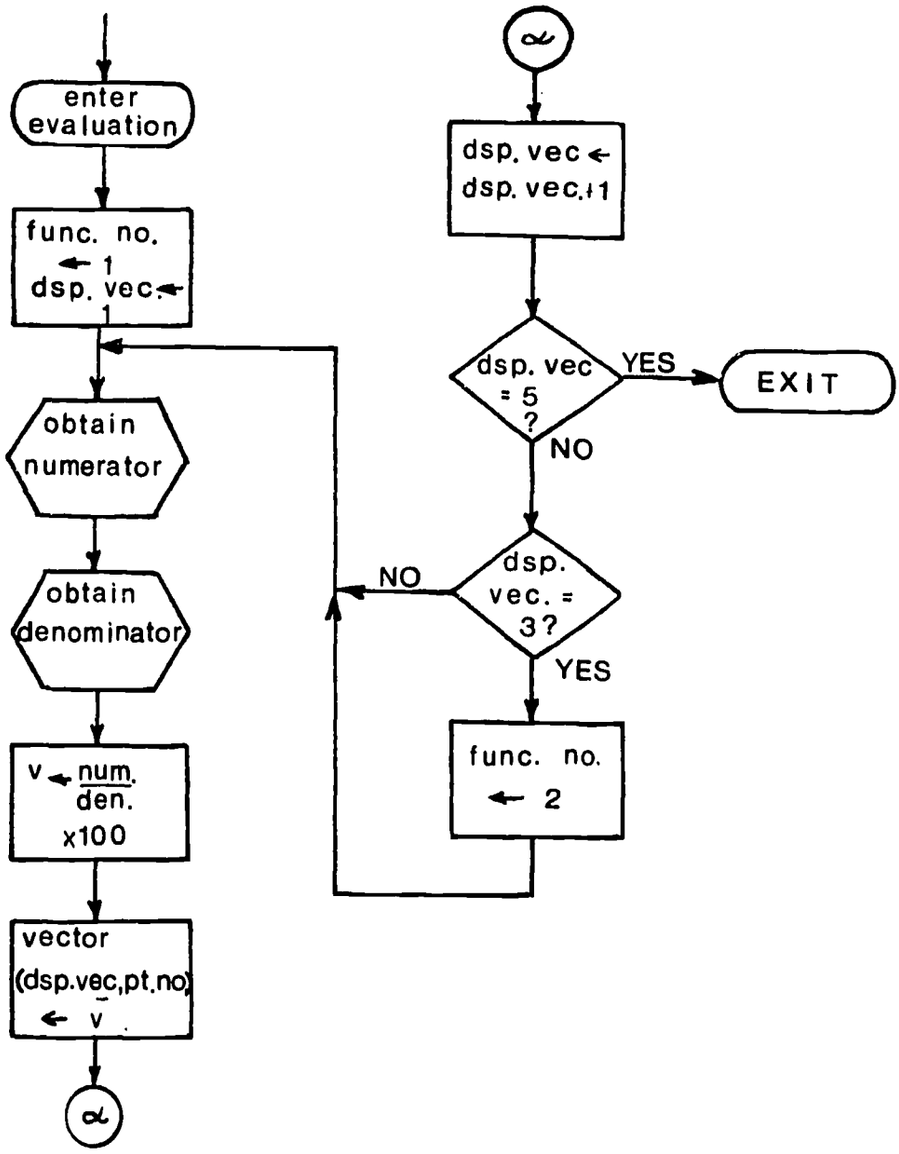


Fig. 4.7 Feedback functions evaluation flow diagram.

previous events). This section presents the software considerations for feedback routines. Hardcopy generation of data and results is also a form of feedback, but since this output need not be presented in real-time, it is not discussed here (see the next section).

Cathode-Ray Tube (CRT) display. A cathode-ray tube device, driven by the computer program, may be used to present information to the teacher. This device is ideal since it is silent, the information displayed may be readily changed, and virtually any type of characters and graphics may be displayed. An image on the screen is generated by allowing the computer to drive the horizontal and vertical deflection plates of the CRT. Images are made up of points placed on the screen by the program. In the following discussions, a point is said to be "plotted" when the program provides the x and y deflection voltages and moves the electron beam to the desired place on the screen. The image is constructed by plotting all of the points in the display vector.

Two types of CRT are available for this application: a storage CRT and a nonstorage CRT. Each has advantages and disadvantages. The storage tube, once an image is generated, is able to "store" or hold the image on the screen for several minutes until it fades or is erased. Therefore, the computer need plot a particular image only once. Changing the image requires erasing and replotting. The disadvantage here is that points may not be plotted at a high rate and that erasure of a previous image may take several seconds. The time of erasure depends upon the construction of the CRT tube.

The nonstorage tube displays a point only as it is plotted and the x and y deflection remains on that point. The point will remain visible for only about 30 milliseconds after the electron beam is moved, after which time it must be "refreshed" (replotted). However, points may be plotted at a very high rate and thus the image may be changed almost instantaneously. This type of display is very flexible but its refreshing requirement places a load on the computer and program. The prototype PDP-4 CATTS system employed this type of display. Since the computer is fast and can perform all other required tasks in very little time, refreshment is adequate and the image can be sustained.

The CRT feedback display routines are responsible for taking results generated elsewhere in the program and then positioning the electron beam on the screen to create the desired image. Because the image must be refreshed, these routines should be executing for as much of the time as possible to avoid flicker of the image.

Several different display routines are used by CATTS. One type of information which may be displayed is a graph of the percentage values obtained from previous evaluations of the feedback functions. To plot this graph, the display routine used a vector of the values plotted versus time. It first plots a horizontal time-line, then plots the percentage values on a 0-to-100 scale vertically. It plots as many values as are available. There are no differences in the plotting process for cumulative and for moving-window graphs. Any of the four available vectors may be used. Figure 4.8 (p. 51) is a flow diagram of the graph-plotting routine. As data, this routine needs the vector of values, the number of values to be plotted, the x and y coordinates of the origin of

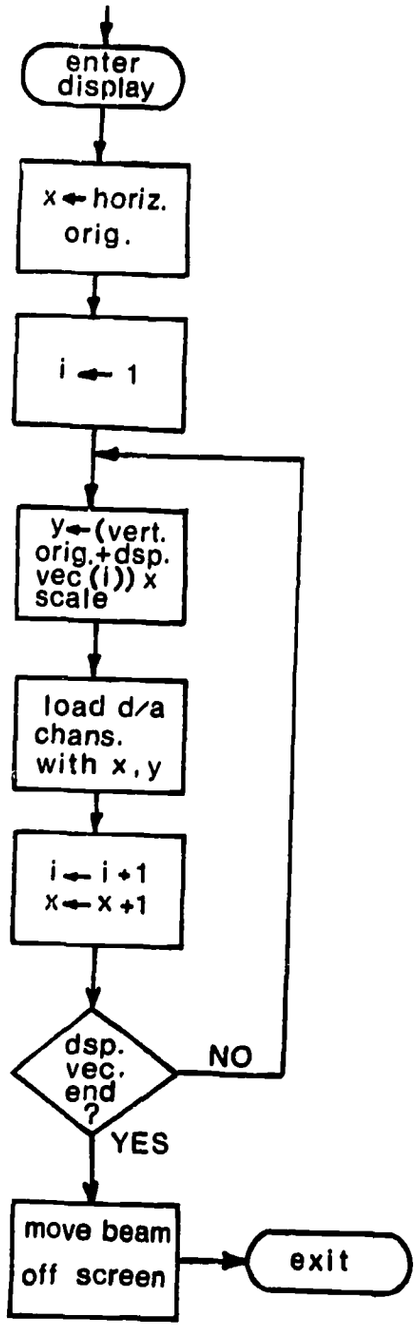


Fig. 4.8 CRT display plotting routine.

the graph, and the vertical size of the graph. The vertical size is determined by the number of graphs to be plotted for one image on the screen. If two graphs are to be shown, one occupies the top half of the screen and the other occupies the bottom half. If only one graph is needed, the entire screen is used and the vertical scale is multiplied by a factor of two, if the origin is at the bottom.

The performance of the display routine can be increased if the vector of functional values is "pre-processed." Pre-processing means transforming the vector of functional values into a similar vector, but with the actual vertical coordinates with respect to the screen inserted as values. These transformed values contain the proper offset for non-zero vertical origins and any multiplicative factors. These values are then directly transferred to the digital-to-analog converter register driving the vertical coordinate.

As an example of pre-processing, consider the displaying of two graphs on the screen. Figure 4.9 (p. 53) illustrates two such graphs, labeled A and B. The origin of B is (0, 0). The origin of A is (0, 120). The multiplicative factor is unity, since two graphs are displayed. The values of graph B are already of proper magnitude and need no transformation, since the vertical offset is zero. However, the vertical values of graph A need to be offset by +120. The pre-processor would simply take the original vector, a_1, a_2, ... a_i, and add the offset 120 to each value, producing the vector $a_1 + 120$, $a_2 + 120$, ... $a_i + 120$. These values are the actual vertical coordinates on the screen. If the origin and size do not change, and if no more values are added to the

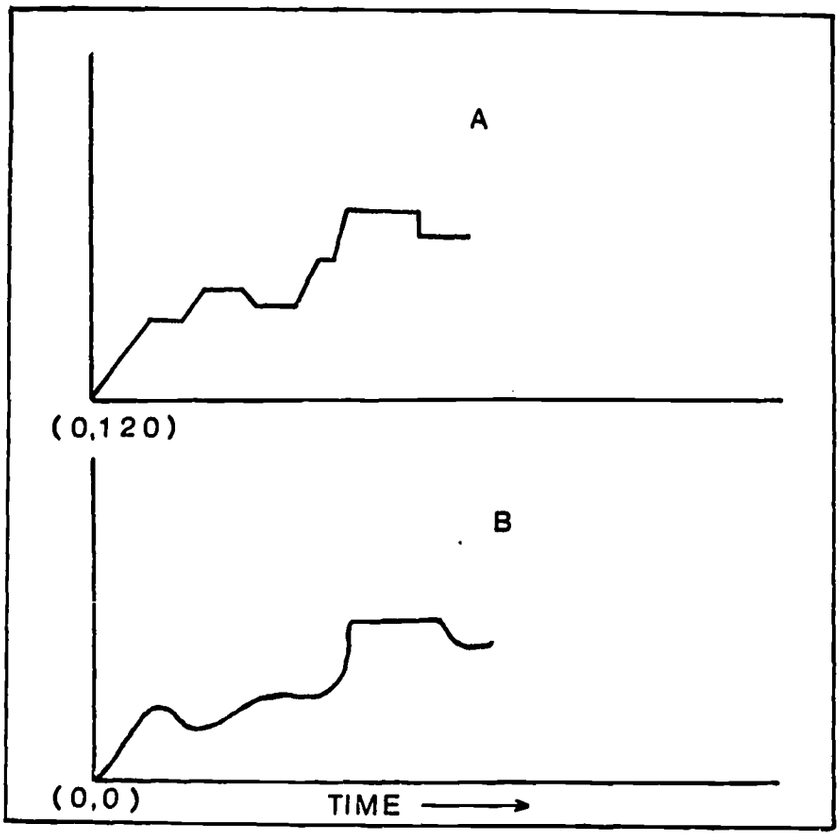


Fig. 4.9 Example of pre-processing graph origins for CRT plotting.

graph, this same transformed vector can be used to refresh the plot. If the offset does not change but it is necessary to add another point to the plot, only this new point must be processed and added to the transformed vector. In CATTS, the offset and size do not change throughout the coding session, so all pre-processing is done as each point in the original vector is generated.

Instead of displaying the actual percentage value graph, the values as an approximation to a predefined goal may be displayed. When this is done, the goal function must be defined with respect to time. The values of this goal function are then used as data (in addition to the functional percentage values, origin position, and multiplicative factor) in a pre-processing routine which computes a transformed factor. The image should be displayed as a horizontal line, representing the ideal goal across time, and a plot superimposed on this line showing plus and minus displacement values from the line.

The pre-processing routine computes the displacement of a display point from the horizontal goal line by subtracting the goal value from the functional value. The vertical ordinate on the screen is then computed by adding this displacement to the vertical ordinate of the corresponding point of the horizontal line. Figure 4.10 (p. 55) illustrates the relationships between goal function, percentage value function, and the plotted display.

The ability of the computer to display character information on the CRT allows numeric values to be presented. One option, selected by the console switches, will allow only a single numerical value to be displayed in place of the plotted graph. The value usually chosen is

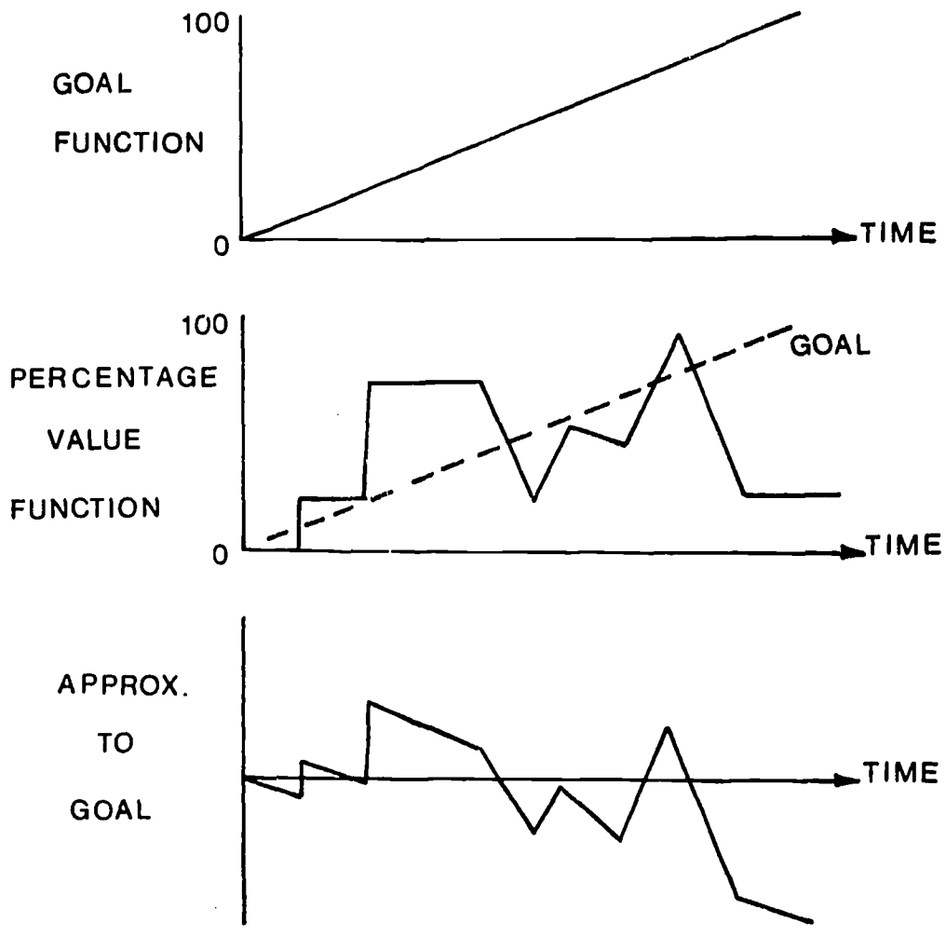


Fig. 4.10 Relationship between goal, value, and displacement from goal.

either the current (last) value of the percentage function or the current displacement from the goal function. Another option, also selected by console switches, allows display of the current numerical value beside the plotted graph.

Chart recorder display. The computer program can also drive a chart recorder pen. The pen movement can be controlled by the digital-to-analog (D/A) converter channels. If the recorder is an x-y recorder, two channels are necessary; they will supply voltages similar to the voltages supplied to the cathode-ray tube display, except that the image needs no refreshing. Another type of chart recorder allows pen control only in the y direction, with the x-coordinate continuously increasing with paper movement. Usually, with either type of recorder, the pen cannot be lifted from the paper without manual intervention. Therefore, the display is limited to a single graph with no numeric characters. The advantage of the chart recorder is that it is a very low-speed device and its controlling (D/A) signals can be transmitted over low-frequency, narrow-band width lines, with very little attention from the computer.

Relay-controlled devices. The PDP-4 has a relay buffer containing eighteen relays which may be individually opened and closed by the program. These relays may be connected to devices which can then be controlled by contact closures. A device such as a tape recorder is one method of providing feedback to the coder. For example, several tape players could be set up with prerecorded message tapes. The program could play sections of these tapes to the teacher by selecting a player, starting

it, and then stopping it after a timed interval.

A panel of lights also can be used to display coded information. The computer needs merely to selectively open or close relays controlling power to the bulbs. Relay-controlled devices require little attention from the program and should be considered if several CATTS systems are being driven from the same computer.

CHAPTER 5

SYSTEM EXPANSION SOFTWARE

The basic CATTS system has been expanded for use by other researchers. Three variations of the basic system are in existence: consensus coding (CONCODE), simultaneous coding (SIMCODE), and CATTS with tally pattern recognition. In general, all the above programs use the same programming monitor scheme, but are more sophisticated in operation.

Consensus Coding of Classroom Interaction (CONCODE)

The CONCODE system is based on the original CATTS program, but two additional coding box interfaces are added to the system. The objective in this system is to have three different coders observe the same classroom situation, tally using the same coding system, and arrive at identical tallies to describe the interaction. This system is used mainly for coder training purposes and for measuring the reliability of coders.

The program controls coding by forcing all coders to synchronize in entering their tallies: once one coder initiates a tally, all coders must initiate and complete a tally before another tally may be initiated by any coder. Upon receiving the final button press of this group of two or three tallies, the program compares the tallies entered by each coder. If all coders agree, the program enters both the single consensus tally and the time elapsed from initiation of the previous tally until initiation of the current tally into the data-storage buffer. It is then ready for another tally initiation. If the coders in a multicoder system disagree, a special error code is entered into the data buffer, then each individual tally is entered in the order of box number, then the box numbers by order of tally initiation, and finally the elapsed

time. At this point, the program waits for the coders to discuss the disagreement by stopping the timer and, if audio or video playback equipment is being controlled, stopping these devices. The coder on box #1 must enter the consensus tally before normal coding may resume. Because it must wait for consensus, this system is not particularly suitable to real-time coding of live classroom situations. Instead, it is usually used to code sessions recorded on audio or video tape. In this configuration, the computer controls the motion of the tape via its relay buffer. When disagreement occurs, the tape is stopped, error lights are lit on each coding box, and timing is suspended. One of the coders may then rewind the tape by remote control so that a particular event may be observed again. When the consensus tally is entered, timing and tape motion begin again. This same suspension of timing and action may also be initiated from a switch on each coding box called, not surprisingly, the "brake switch."

Several options, to be specified by the operator in advance, are available with the CONCODE program: (a) the number of coding boxes selected for use (unused boxes are ignored by the program; a single-coder system is roughly equivalent to the basic CATTS); (b) one, two, or three level coding sequences; (c) the length of the "time-out" interval (within which all coders must have begun to tally after one coder has initiated) before a forced disagreement is made; (d) the length of a pause interval (within which all boxes are ignored after a disagreement; this guards against an erroneous consensus being entered on box #1); and (e) the entering of the consensus may be bypassed for more ef-

ficient real-time operation (the disagreement is entered into the data, but action is not suspended).

CONCODE signal input. Signals from the boxes are input to the computer in exactly the same manner as in the basic CATTS. However, in the prototype PDP-4 system, a multiplexor, constructed from relays and driven by the computer's relay buffer, is used to connect sequentially each set of box signal lines to the analog-to-digital (A/D) converter inputs. Since the relays do not switch instantaneously (approximately 15 milliseconds must be allowed for the present CONCODE relays), the time must be noted when the relay was last commanded to switch. The inputs are not considered valid until at least the fifteen milliseconds have passed. Therefore, the A/D clock is used to "time out" the wait before sampling the lines. The signals are processed as in CATTS. The only additional data the scanning routine needs is the number of the box currently selected. It uses this number to maintain both separate counters and status data for each individual box.

Figure 5.1 (p. 61) is the flow diagram of the signal input subroutine. This routine is dispatched by the A/D clock interrupt service routine when the counter in this routine reaches a predefined number, n, where n is determined by the formula

$$n = \frac{3 \times 15}{\text{number of active coding boxes}}$$

The factor of 15 is the relay latency in milliseconds; the factor 3 is the maximum number of boxes possible in the system. The number of currently active boxes is defined as the number of button boxes currently being monitored for input. In a 3-coder system, this number can vary

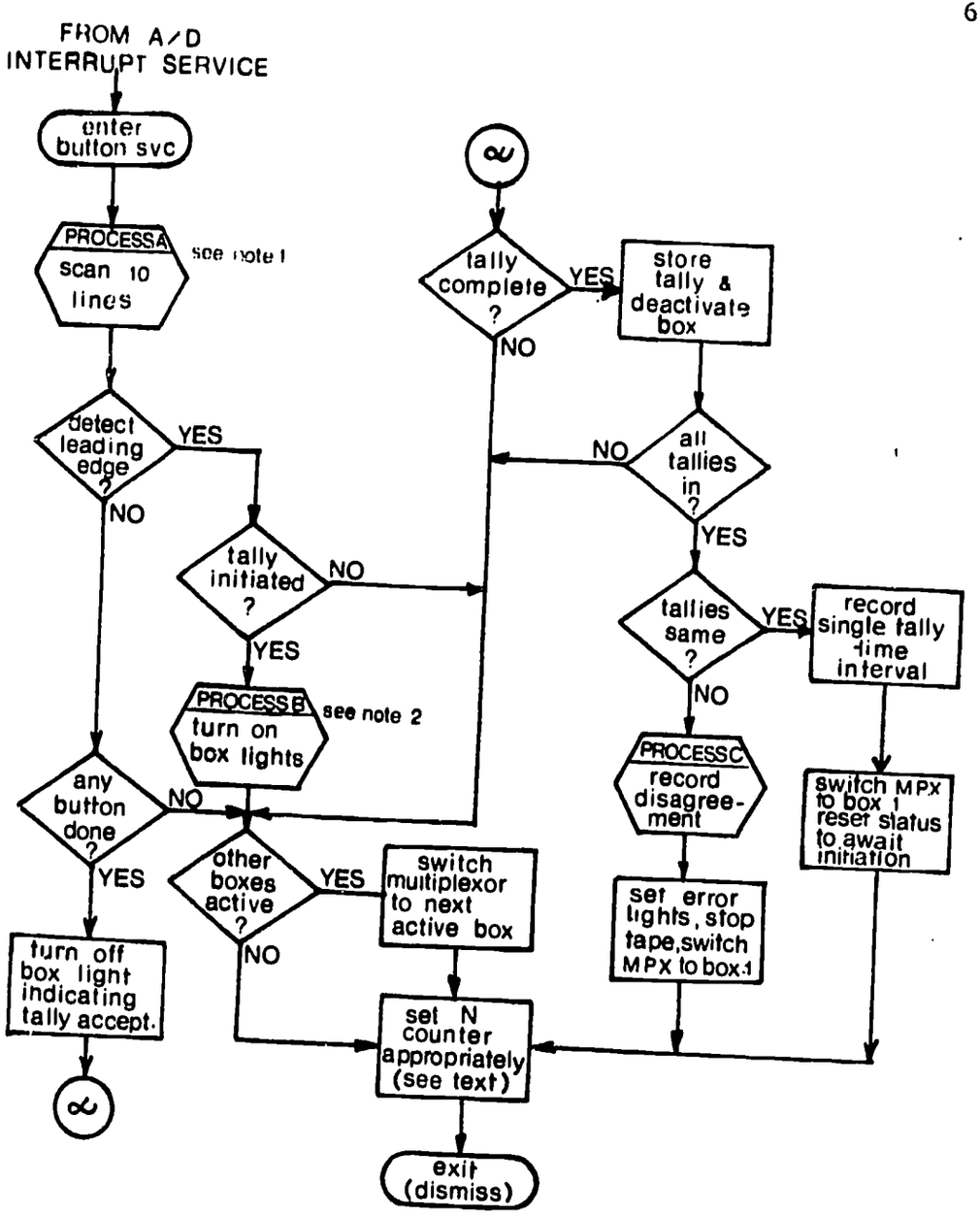


Fig. 5.1 CONCODE signal input routine.

NOTE 1: Process A samples all ten lines as in the basic CATTS. Upon exit from this process, flags are set indicating any detection of a leading edge of a button press, or any completion of a button press.

NOTE 2: Process B is entered when a leading edge was the first leading edge encountered while waiting for a tally initiation. This process performs three functions: (1) turns on one, two, or three lights on all boxes being used; (2) records the value of the interval timer (time since previous initiation); and (3) resets the timer to zero.

from one to three, since, as a box completes a tally, that box becomes inactive until all other tallies are completed. Therefore, n is always set so that any active box is monitored every 45 milliseconds, or 22.2 times per second. This sampling rate is about one-fourth the sampling rate in the basic CATTS, but no degradation in data input rate is noticeable.

Feedback to the coder is presented with three lights on each button box which are controlled by the program via the relay buffer. These signal lights indicate to each coder how many button presses are necessary to complete a tally. The light signals are individually controlled and are always valid for the particular coder. Thus, although complete tallies must be synchronous, individual button presses within tallies are completely asynchronous between coders.

The timer service routine is similar to the one in the basic CATTS. However, certain functions are different; for example, the main timer keeps the time elapsed between tally initiations, not the time elapsed since the beginning of the coding session. This facilitates the error time-out function described above and also decreases storage space for the time data. Each time a clock interrupt occurs (every 0.1 second), the interval timer is incremented. Its value is then compared with the value specified for time-out. If the timer equals or exceeds this value, and at least one coder has not begun a tally after another coder has initiated, the timing routine initiates a task to stop all action and set error lights on all boxes. The action taken by this task is identical to the action taken by the signal input routine when disagreement

occurs. When the consensus is entered, timing resumes.

The program allocates a location which stores status conditions for each box. Whenever conditions on a particular box change, its status work is updated. Information contained in the status work includes: box active, tally initiated, tally in progress, tally complete, level of current tally (i.e., category, subcategory, of sub-subcategory level), sequence number (the order in which the boxes began tallies). This status information is used by various parts of the program for coder feedback control, multiplexor control, and data recording.

A later version of CONCODE includes a CRT display which lists the previous five tallies, and, when a disagreement occurs, the current tally and status of each coder.

Data produced by CONCODE may be output on the teleprinter, paper tape, and/or magnetic tape. The magnetic tape output is the binary-coded decimal (BCD) records; the record format can be read by FORTRAN programs analyzing inter- and intra-observer performance and reliability, inter-term reliability, and statistics.

Simultaneous Coding of Classroom Interaction (SIMCODE)

The SIMCODE program uses the same multiple-observer hardware configuration as CONCODE. However, each observer coding with this program is allowed to use a different coding system, since tallies are not compared and tally levels may be specified differently for different boxes. Tallies must still be synchronous for ease of data-analysis. SIMCODE usually runs with only two coders; thus, a pair of tallies is always entered as data. If the two coding systems used are tailored to

describe different types of classroom events (i.e., verbal vs. nonverbal), each coder is freed from observation of too many widely varying types of details. Thus, more detailed data may be taken by each coder.

The signal input and timing routines are almost identical to those used in CONCODE. This is, if one observer initiates a tally, the other observer's lights come on. However, if the second observer does not begin a tally within a specified short period of time, the timing routine enters the second observer's previous tally as data to make a pair. This feature, an "automatic button press," allows one coder to refrain from continuous coding if action as defined by his particular coding system does not change.

All data output options noted previously are available with SIMCODE.

Code-Pattern Recognition

The basic CATTS system has been modified to provide an alternate method of data reduction and teacher feedback to that described previously. In this modified system, data is analyzed in groups of tallies. These groups are compared with predefined groups representing certain patterns of classroom behavior. Feedback information is derived from conditions indicating successful matches between input data and the predefined patterns. The following paragraphs describe the software unique to this system; the signal input and timing software is identical to that of the basic CATTS.

Pattern detection. When the signal-input routine detects the completion of a tally, it stores the tally and time of occurrence in the

data buffer, as before. However, instead of ending the task immediately and returning to a ready state, it initiates a level three task that compares this tally with those in a set of predefined tally sequences called patterns. The format and structure of a pattern are described below. Let us just say here that, for each pattern in the set, the current tally may (a) initiate the start of a pattern (the current tally state); (b) match the next sequential pattern tally to continue the pattern or to "complete" it; (c) match the previously matched tally of the pattern to continue the pattern; (d) not match any of the items mentioned above but match the first tally in a valid "subpattern" continuing the pattern; or (e) none of the above, terminating the pattern and setting its status to ready. If a pattern is completed by the tally, as in (b), the pattern is counted and feedback information may be generated.

Patterns are sequences of tallies, with each tally having certain properties, or attributes. Consider, for example, a two-column coding system. It is desired to count occurrences of the pattern (1_2, 3_1, 4_1, 6_3), which may represent a certain pattern of behavior. The tally 1_2 is designated the "head" of the pattern; 6_3 is designated the "tail." If the same sequence of tallies is received as input data, the pattern is said to be matched and it is counted. However, it may also be desired that the sequence 1_2, 3_1, 3_1, 4_1, 6_3 be counted as the same pattern, since it describes the same behavior. The repetition of tally 3_1 merely indicates continuation of the same event. Thus, 3_1 may be repeated any number of times (without differing intervening tallies) without destroying the pattern. Tally 4_1 may be a special type of event that should occur only once; therefore, if repeats of 4_1 occur, the pattern is not con-

tinued and is, therefore, not matched by the input sequence. The properties of "head," "tail," repeatable, nonrepeatable are all attributes of a particular tally, the tally when it occurs within certain patterns, or the position of the tally within a pattern.

It may be further desired to count as the same pattern an input sequence of 1_2, 3_1, 4_1, 3_1, 4_1, 6_3. Here, the subpattern 3_1, 4_1, may be repeated without destroying the pattern. In this case, 3_1 has the attribute of being the head of a subpattern and 4_1 has the attribute of being the tail of the same subpattern, a pattern conveniently written as $(1_2, (3_1, 4_1), 6_3)$.

To implement pattern detection on a computer in real-time, the patterns must be input as initialization data to the program. Each tally should carry with it its own attributes. In the PDP-4 CATTS system, a pattern is stored as if it were a stack of items in sequence. Thus, the pattern above may be viewed initially as the stack:

```
1
 2
3
 1
4
 1
6
 3
```

To test the input stream for a possible pattern match, the input tally is compared with the top tally of the stack. If the two items match, and the top item is the head, the input tally has "initiated" the pattern. The head remains at the top and is given the special but transient attribute of having initiated the pattern. Let us say that this has occurred by receiving the tally 1_2. If the next input tally is 1_2,

and if 1_2 is repeatable, the pattern is "continued," and the stack remains unchanged. If 1_2 is not repeatable, the pattern is discontinued and is terminated, losing thereby its initiated status. If the pattern has just been initiated and the next tally is 3_1, 3_1 is first compared with the top of the stack to test for repeats; the result is no match. 1_2 is then "popped" off the stack and 3_1 compared with the new top of the stack. This matches, so the pattern is continued. Repeats of 3_1 leave conditions unchanged. If 4_1 enters, 3_1 is popped off and the pattern continues. If neither 4_1 nor 3_1 had occurred, the sequence would have been broken and the entire stack restored to its original state, ready for a 1_2. When the final two items are left in the stack, it means that the next-to-last item has been matched already if the previous item was nonrepeatable. If the next-to-last item has been matched and is nonrepeatable, it will not be present. Then, if a tally matching the tail is input, the pattern is immediately counted and restored and any other desired action taken. Note that if another tail tally was input, the same pattern could be continued if the tail were repeatable. However, for real-time analysis, the pattern should be detected as soon as possible for feedback purposes. Thus, the tail is always nonrepeatable in real-time mode. (See text below for discussion for non-real-time analysis of the data.)

There are other features which can be included as attributes of tallies or subpatterns. Many times, the coding system being used carries information about who is speaking in the classroom. For example, the final subscript may be used for this purpose: 1 = teacher, 2 = teach-

er-student duet, and 3 through 10 = code numbers of individual students. Thus, an attribute of a tally within a pattern may denote that the tally should be matched for any of the subscripts 3 through 10 with a certain main category. This is written on paper as 3_{3-10}, for example. Another attribute allows the tally and the input tally to match only if certain columns match, with other columns "don't care." Any input tally of subscript 2 will match this tally, irrespective of the main category.

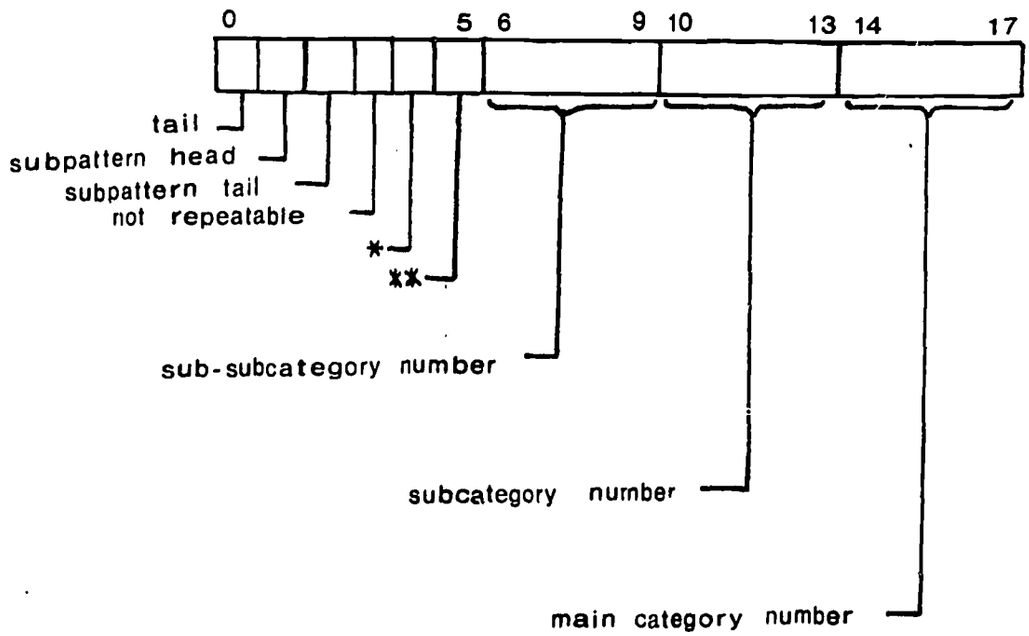
Certain rules for the construction of patterns must be observed if successful detection is to occur. If a pattern contains a subpattern, the head tally of the subpattern should not equal the tally following the tail of the subpattern. Consider the pattern $(2_1, (3_2, 4_1)\ 3_2, 6_4)$. If the previous tally has matched 4_1 and the current input tally is 3_2, an ambiguous condition arises. This action in this case would be that the 3_2 following the 4_1 would be matched; the subpattern would never be entered the second time. Consequently, the input pattern of 2_1, 3_2, 4_1, 3_2, 4_1, 3_2, 6_4 would not be detected as a pattern. This difficulty can be eliminated to some extent in non-real-time analysis if "backup" is allowed (more than one tally is used to determine which branch of pattern to take).

The stack operations are implemented by using a pointer to the top of the stack, with only one copy of the pattern in core memory as a vector. Popping the stack does not destroy a tally; it simply moves the pointer. Restoring the stack moves the pointer back to the beginning of the pattern vector.

In the prototype PDP-4 modified CATTS system, a maximum of three

levels to a tally is allowed (i.e., category, subcategory, sub-subcategory). Each button-press number may run from 1 through 10. These numbers are offset by -1 for storage purposes and are represented internally as four-bit numbers ranging from 0 through 9. Thus, to represent an entire tally, 12 bits are necessary. Pattern tallies are stored in on 18-bit computer work, leaving six bits to carry attribute information for the tally. Figure 5.2 (p. 70) shows the bit assignments in the pattern tally storage word. For the attribute bits, a binary one implies that the tally carries the particular attribute; a binary zero implies that it does not. Note that the head of the pattern attribute is not represented. This attribute is implied by the position of the tally in the vector.

Figure 5.2 is the flow diagram of the pattern detection subroutine. As data, this routine requires the tally pattern vectors, the number of such vectors, a vector of pointers indicating the storage address of the head of each pattern, and the input tally, with codes packed in the same format as bits 6 - 17 of the word shown in Figure 5.2. Bits 0 - 5 of the input tally word are zero and are ignored, as it carries no attributes of its own. The subroutine maintains three storage areas: one area is a vector of status indicators (one status word for each pattern); the second area is a vector of pointers indicating the address of the current top of the stack for each pattern; and the third area contains the results from scanning all patterns with one input tally (one word of this area contains the number of patterns completed by the current tally, followed by a vector which contains the numbers of the patterns completed). The status indicator may contain one of three numbers:



* Bit 4: A 1 in this position implies that the subcategory number given in bits 10-13 may range up to 10.

** Bit 5: Sub-subcategory may range from number given in bits 6-9 up to 10.

NOTE: A zero in any of the positions 6-9, 10-13, or 14-17 indicates that this position is don't care.

Fig. 5.2 Location of bit assignments for pattern tally storage word.

0 = top of stack must be matched; 1 = top of stack is repeatable (may be matched or next tally may be tried); 2 = try to match the tally following tail of subpattern. The following symbology is used in the flow diagram:

STATUS (n):	Status word of pattern n.
PRT (n):	Pointer to top of pattern stack n.
(6,17):	Bits 6 through 17 of associated data words.
INTAL:	Input tally word.
CVEC:	Vector of the numbers of the patterns completed with current pass of pattern scan.
t:	The number of patterns completed with the scan.

Feedback Generated from Pattern Detection. The results generated from the pattern scan are used in several ways. In the modified CATTS, the patterns described above are termed minor patterns and are used to delineate larger patterns of behavior, which are termed major patterns. Each minor pattern is placed into one of three groups:

Group 1:	Minor patterns in this group may initiate a major pattern.
Group 2:	Minor patterns in this group may end a major pattern.
Group 3:	Includes all patterns not in groups 1 or 2. These patterns are included for reference or analysis purposes.

Patterns in groups 1 and 2 simply delineate a major pattern which begins with a Group-1 pattern and must end with a Group-2 pattern, with any sequence of tallies or patterns in between (except a Group-2 pattern,

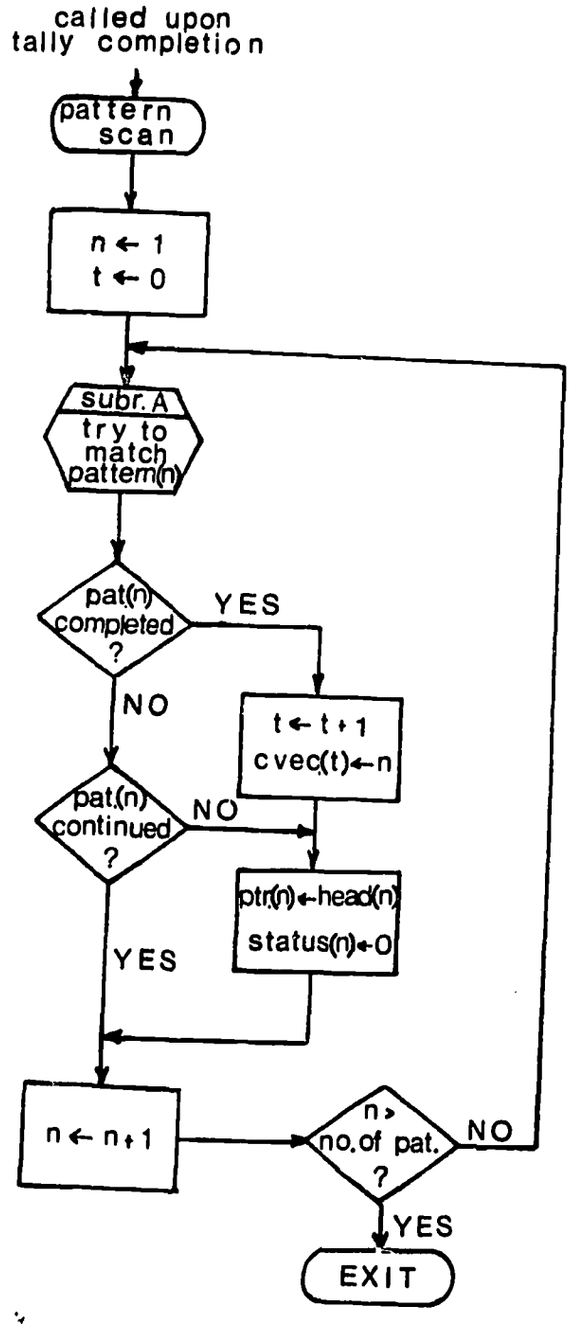


Fig. 5.3a Pattern detection subroutine (Part A).

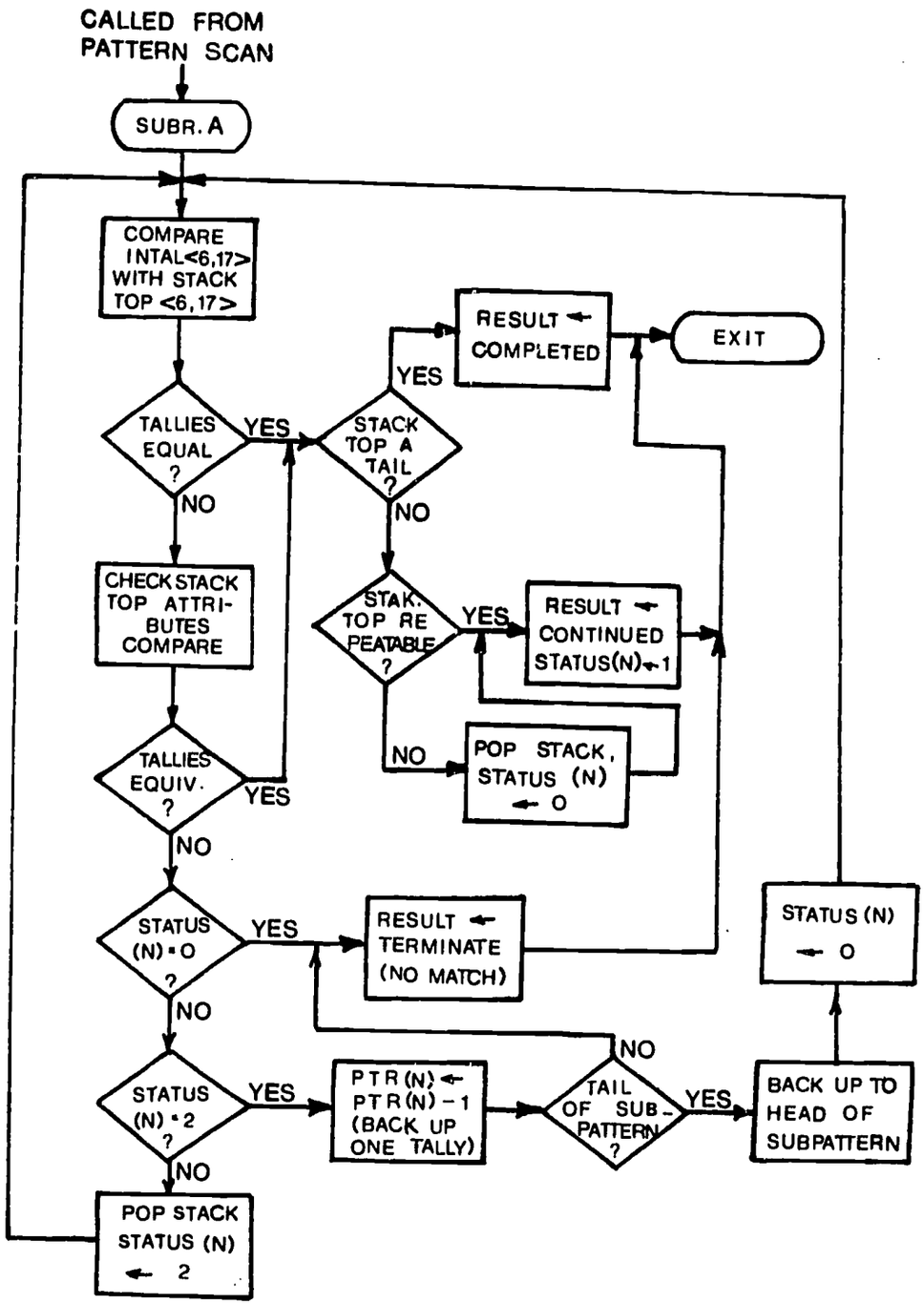


Fig. 5.3b Pattern detection subroutine (Part B).

which would end the major pattern).

In real-time operation, the program is running either in a major pattern or out of a major pattern. The "in" mode is entered from the "out" mode by the completion of a Group-1 pattern; no other out-in transition exists. The out code is entered at the start of the coding session or when a Group-2 pattern is completed while the program is running in the in mode. The transitions from out to in and vice-versa trigger changes in feedback information to the teacher, indicating whether he is in a major pattern or out of a major pattern. In addition, the timing routine provides information which indicates whether the teacher has been in or out of major patterns too long. These timing thresholds are specified at program initialization. Information is presented on a device capable of indicating one of four different states: (a) out too long; (b) out but not out too long; (c) in; (d) in too long. In CATTS, this device is a chart recorder driven by one of the computer's digital-to-analog converter channels. (A bank of four lights would be ideal for this application also.) The timing routine controls transition from (c) to (d) and (b) to (a). The results from the pattern scan are used to initiate transitions from (a) to (c), (b) to (c), (c) to (b), and (d) to (b).

In addition to the real-time feedback, a listing is produced which contains the following information for each tally: the time of the tally, the tally, the state of the feedback, a list of minor patterns completed by the tally, and time spent in each minor pattern just completed, the symbol X if the minor pattern began a major

pattern, with the time since the previous "in" state was left, or the symbol 0 if the tally ended a major pattern, with the time since the previous X occurred. At the end of the coding session, a print-out lists each minor pattern, the number of times it occurred, the total time spent in the pattern, the number of times it began a major pattern (if a member of Group 1), and the number of times it ended a major pattern (if a member of Group 2). Also listed is the total time spent in major patterns and the total time spent out of major patterns.

The pattern detection routine is partially responsible for maintaining the data representing the time spent in minor patterns. For timing purposes, a vector containing one location for each pattern is allocated. Each time the head of a pattern is matched for the first time (not a repeat), the corresponding timing location is set to zero. Each time a clock-interrupt occurs, the timing routine increments each pattern timing location. Thus, when a pattern is completed, the timing location contains the elapsed time (in tenths of seconds) since the pattern was begun. This number is eventually printed by the print out routine and is also added to the running total of time spent in the particular pattern.

Non-real-time pattern detection. Deficiencies exist in the real-time pattern detection scheme. For one, it is usually desirable for analysis of the data to count as the beginning of a major pattern the beginning of the Group-1 pattern which initiates this major pattern since this is the actual start of a certain sequence behavior. This point cannot be determined in real-time, since we cannot predict future

events. Further, a minor pattern is immediately completed in real-time which the tail is matched. However, there may actually be several succeeding tallies matching the tail, indicating continuation of the same pattern of behavior. This last process could be carried out in real-time, completing the pattern only when the tail is finally not repeated; but it was desired to trigger any feedback as soon as possible. To circumvent these deficiencies and provide a more accurate picture of events, a PDP-4 program was written to detect the actual start of major patterns and the actual end of minor patterns. The details of program operation are very complicated, but it uses a modified version of the pattern detection scheme described above and uses "look ahead" and "look back" schemes to complete major and minor patterns.

The result produced by the program is an "event vector" from which many different types of data and listings may be produced. Basically, the vector can be thought of as a series of events placed on a time line. An event can be one of several types of items. The types include:

1. Start of major pattern--coincident with a start of a group 1 minor pattern, which is also identified. Stored with this event is the tally immediately preceding the start of pattern and the time since the end of the previous major pattern.

2. Start of minor pattern. Pattern identified along with group number.

3. End of the major pattern--with tally or pattern ending the major pattern.

4. End of minor pattern.

5. Any tally which is specified as being of special interest.

The vector is scanned for particular events or types of events and various results are printed. This vector, or results obtained from it, may be written onto magnetic tape for further processing.

Markov Chaining Pattern Detection (CHAIN)

A parallel software development for extended batch-analysis of raw CATTS data has also been initiated. Work has begun on a new procedure for statistical analysis of teacher-pupil interaction data which promises to retain much of the information which is pooled or lost through traditional statistical methods. This procedure views the teacher-pupil interactions of a lesson as a single continuous sequential vector of behaviors. A computer program called CHAIN* searches the vector for Markov chains, "families" of category chains which appear too frequently to be chance sequences within the vector. The program can accommodate over 50 variables of teacher and/or pupil behavior and will uncover chains of events containing sequences of up to eight behavioral categories. In addition, the program yields the average time in each chain, the frequency of occurrence of each chain, and the proportion of total time consumed by each category of behavior. Chains can be analyzed by time periods across a lesson (e.g., beginning, middle, and end of lesson) to detect any changes in teacher-pupil interaction pattern as time progresses. Further, the program has an option which permits computer analysis of categories used most frequently across many obser-

*The program CHAIN was developed at the University of Michigan through the collaboration of Dr. Lee Collet and Dr. Melvyn I. Semmel. A more detailed rationale and application of the CHAIN program can be found in a paper presented at AERA in New York City, 1971.

vations. Unused categories or those accounting for relatively little variance in the total vector can be pooled and carried as a single throw-away category in the analysis of chains. Hence, in effect the computer can assist us in refining our observation systems, thereby enabling a clearer identification of significant teacher-pupil behaviors in classrooms.

CHAPTER 6

CURRENT SYSTEM DEVELOPMENTS

Because CATTS is considered to be an evolving system in a state of constant modification, ideas for expansion and application are now in various stages of development. Current hardware and software extensions of CATTS can be divided into three general areas: Data Collection and Input Developments; Extensions of the Real-Time Multi-Observer System; and Feedback and Output Developments.

Data Collection and Input Developments

Developments for increasing input and data collection techniques have concentrated on two different input methods. On-line collection for immediate feedback is moving from the original CATTS closed system method into the remote direct-distance dial touch tone (TT) telephone input method. By putting data directly into the computer through a telephone data set adaptor, pushbutton selections from a remote TT telephone can provide the flexible coding input required for the CATTS system. Coder status feedback may be employed by using the talkback system of two or three different tones available with a TT telephone data set. These tones then can be sent back over the incoming data lines either separately or in combination with each other to the originating coder input telephone. When the talkback system is put under computer program control, with electronic discrimination among these tones at the Observer-Coding Station, the position of the observer within the coding sequence may be signaled using the same method that is employed in the pushbutton box arrangement. These talkback tones may be distinguished at the cod-

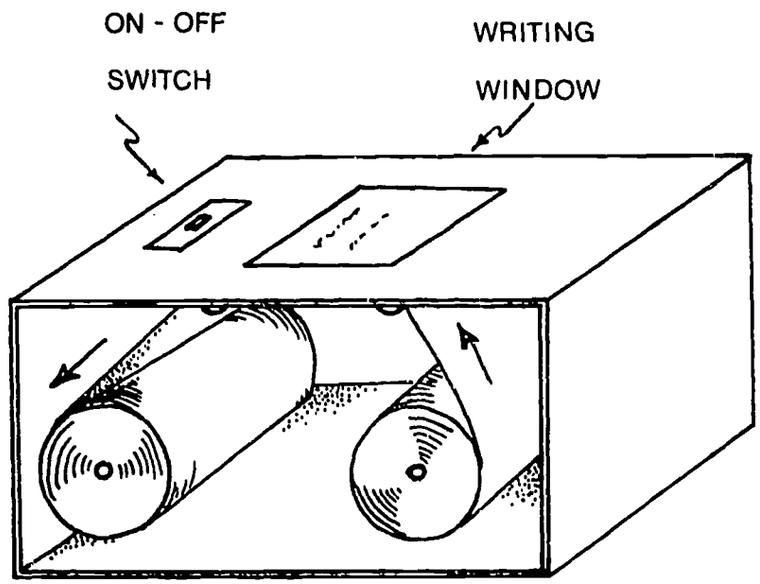
ing station by placing the receiver into a cradle containing a tuned filter network and electronic flip-flop switches set to control the two or more signal lights that are displayed in front of the observer. In addition, through software control, direct telephone inputs may be multiplexed into the computer and sorted out as the coded data arrives from different sources. The feedback signals to the observer-coders again can be presented through the individual telephone talkback systems. Since the data is entered by hand from the remote terminals, the ability of the system to multiplex input data is possible simultaneously with the sorting and storing of raw data for later processing. A system such as this allows selective feedback to various classrooms and general mass storage of observation data.

An alternative for the transfer of remote real-time coded observations to the central CATTS computer system by some method other than commercial telephone lines is also under consideration. Recent developments in possible educational uses of satellite communication systems for remote real-time connections to central computing facilities, which could provide interactive instruction (CAI, CMI) to outlying localities, lend themselves readily to widespread CATTS type applications. In a preliminary draft of a cost analysis of satellite versus commercial telephone communication systems, Jamison, Ball, and Potter of Stanford University's Institute for Mathematical Studies have suggested that use of satellite communications to remote areas in the United States will be the least expensive method for providing computer interaction instruction services. One of the basic foundations of the CATTS system

is its ability to gather classroom observation data simultaneously from multiple sources, and if required, feed back to the classroom the current status of these selected observations. A widespread application of a CATTS type configuration could use this interactive communication system very effectively.

Methods for gathering data by off-line methods are also under development. Portable paper-tape punches or portable magnetic tape data recorders which include pushbuttons and internal solid state timing and control circuits, may provide good methods for collection of baseline data from testable populations inaccessible to on-line methods.

One such device under development is a prototype manual unit for gathering off-line observation data. Application of this unit is meant only to assist in the collection of the raw data and not in the automatic transfer of the collected data to the computing facility. The unit illustrated in Figure 6.1 (p. 82) consists of a battery and/or AC powered motor which steadily moves a spool of simple adding machine paper tape across a writing platform, thus allowing an observer in a classroom to code observations with pencil upon the moving strip of paper. By moving the paper at a constant speed across the writing platform, the time between coded observations is estimated by simply measuring the distance the paper has traveled between codes. By retrieving the time measurement between sequential codes in this fashion, this unit allows the observer to concentrate on the important observable events taking place, and the coding becomes time independent. Other features include interchangeable window templates which provide the observer with different writing guides for the application of various coding



CASE CONTAINS PAPER ROLLS
CONSTANT SPEED MOTOR,
GUIDES, AND BATTERIES.

Fig. 6.1 Pictorial representation of manual device for off-line data recording.

systems on the same unit; if only one-half of the paper is used for coding, the paper may be reversed or turned over and the empty side used to code more observations.

Data transfer from this unit into the computing facility is done by hand, using a transparent template marked off in corresponding time units. The data is transferred onto cards or magnetic tape by entering each observed code entry, followed by the appropriate time hand-measured from the last observed code written on the adding machine paper tape.

An appropriate program is then called and an analysis of the data is performed, using the hand calculated time-line to provide the computer time-line for any requested graphic display of the data.

The basic design configuration of a portable paper-tape punch data collection device would consist of a paper-punch mechanism, a pushbutton coding device, and a timing and control unit. The paper-punch mechanism is an electro-mechanical,solenoid-operated,seven hole paper-tape punch which advances and punches the paper-tape only after a command from the control unit has been received. The coding device would consist of ten pushbuttons mounted into a box, and would be monitored by the control unit, similar to that used in the prototype CATTS on-line configuration. The control unit is conceived of being a solid state device that collects pushbutton coded information, stores and recodes this information until a punch command is given, sends the coded data to the punch mechanism in a recorded BCD structure along with a binary-coded time-frame to be punched after the data, and advances the

paper-tape into position to accept a new coding sequence. The time-frame, provided by an internal clock, must be reset and activated at the onset of each new coding sequence. The elapsed time could then be sent to the punch mechanism when requested by the punch control command. The cumulative total of the time elapsed during the data-gathering session could then be calculated when the punched tape is entered into the main CATTS computing facility. In addition, all three components could be mounted into a portable case and carried into any classroom.

Cassette or key-to-tape units are also commercially available for adaptation into a remote CATTS data-gathering system. A timing mechanism has been added to the cassette units to provide a time-frame similar to the portable paper-tape punch. Some of the available cassette tape units are battery powered, offering access to observable situations which lack 110 VAC power.

Another more likely unit is a continuous play/record tape recorder as illustrated in Figure 6.2 (p. 85). This unit basically would modulate a teletype code onto a moving audio tape recorder. Small solid state modules are available which could convert simple button presses into ASCII teletype code and modulate that code with a small oscillator. The time element in this case would be the time measured between the recorded data pulses on the tape recorder. Transmission to the computer could then be accomplished by inputting directly into the teletype input buffer found on most computers. By increasing the transmission rate within the input buffer, the speed on the tape recorder then could

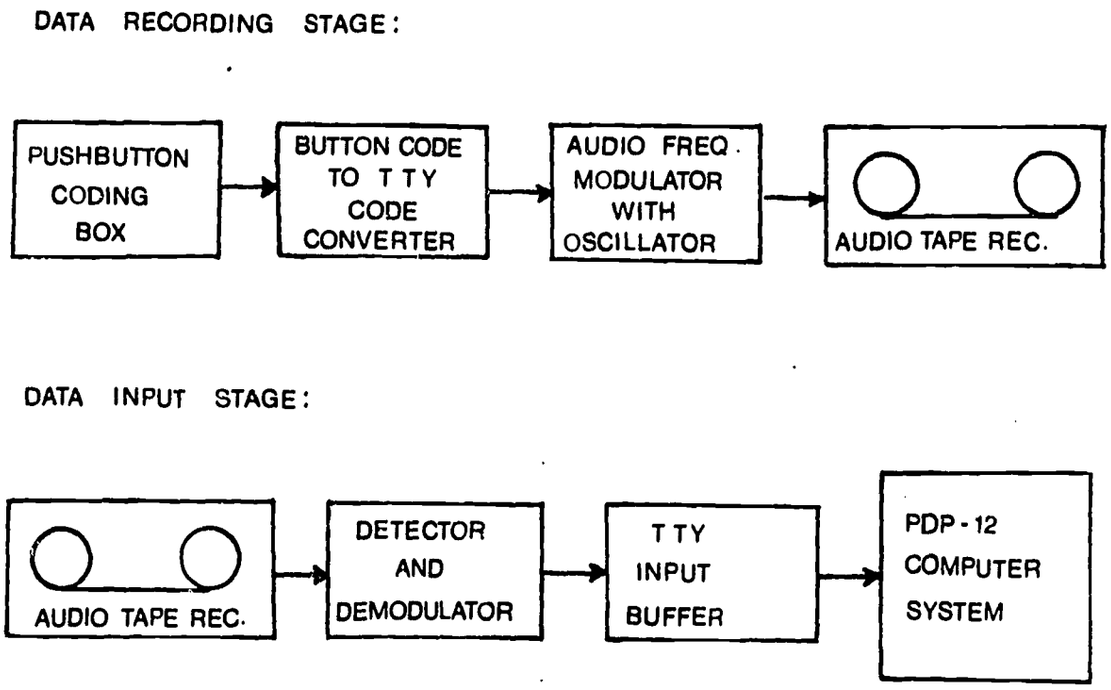


Fig. 6.2 Schematic of proposed off-line data recorder.

be increased and the transfer of data accomplished up to four times as fast. Time would be counted by the computer on a real-time clock and assigned to the transmitted code as it enters into a teletype buffer.

The recoded transfer of data from these portable units to the main processing computer may be handled by two methods. With a magnetic tape or cassette tape recording device, the tape or cassette cartridge itself may be mailed or played over an acoustic coupler by telephone into the main computing terminal. The portable paper-tape punch, while initially more expensive, will produce compatible fan-fold paper-tape which may be mailed to the CATTS computing facility and read into the computer directly with no alternative for storage and analysis.

A third and different development for input information into CATTS is based upon the extraction and evaluation of the prosodic features of human speech. The Speech Auto-Instructional Device (SAID), which was originally developed by Dr. Harlan Lane at the University of Michigan, is a system which can extract, in real-time, the prosodic features of human speech and transform features of fundamental pitch, amplitude, and tempo into corresponding analog signals. These signals then may be digitized by the computer to perform computations against data previously input, producing a correlation between (both) signals. Applied to the CATTS system, the incoming prosodic information may be classified by comparing the correlation figure to known prosodic parameters. Machine discrimination between statements and questions may be possible under this type of system. Therefore, this discrimination feature may be coupled with the standard form of CATTS observation coding. This combination may provide additional information to the on-line data collection

capabilities of CATTS.

Extensions of the Real-Time Multi-Observer System

The programming techniques used in CONCODE and SIMCODE may be extended to provide time-shared coding from many different observers. The coding boxes may be grouped to provide any combination of CATTS, CONCODE, or SIMCODE.

For example, in a system with ten coding boxes, three could be coding in one CONCODE system, two could be coding using SIMCODE, another two could be using CONCODE, and three could be coding independently in a CATTS-type system. To each observer, the system would appear only as his particular coding environment.

Given relay switching latency, the relay-multiplexed system described previously can support a maximum of four or five coding boxes. However, by using solid-state electronic switching, more than 20 boxes could be supported by a small computer, since switching could be done almost instantaneously.

The software for such a system is only slightly more complex than for a single CONCODE or SIMCODE system. The basic monitor remains the same. Only one signal input routine is required, since it can be written in re-entrant code and shared by all active coding boxes, independent of the particular coding system. The non-sharable portions of the program are the individual data and status buffers. Certain portions of, say, the CONCODE machine instructions will not be usable to SIMCODE or CATTS, but they may be shared by other CONCODE systems. The timing

routine still needs only one real-time clock. However, it will be keeping a timer for each currently active system, with each timer being either for a CONCODE, SIMCODE, or CATTS system.

Therefore, since all three programs can share in common major portions of the operating system software, only one program about ten percent larger than the largest single program is needed. Data storage is the only quantity of core storage which will increase in proportion to the number of currently active users. In such a system, a mass storage device like magnetic tape, or disk, could be used to increase available data-storage space. As small volumes of data are collected from each system, they could be stored on the device and the vacated space used for more incoming data. Then, when any user wishes output of his entire data buffer, these small sections could be recalled, assembled and output onto another device. Furthermore, the mass storage device could hold sections of program not currently needed, such as initialization routines. When a new user wishes to set up a coding system, the routine could be brought into core storage, initialization would take place, then this area of core could be used for a new data buffer. It is likely that on such a time-shared system 50% or more of the processor's time would still be available for other purposes. This amount of time is not enough to sustain flicker-free displays if they are driven by the D/A channels on a nonstorage CRT. However, by employing storage CRTs, many displays may be maintained with almost no attention from the program.

Feedback and Output Developments

Expansion of feedback methods are also in progress. Branching from

the CRT display as used in the original CATTS presentation, the development of flexible feedback devices has been of great importance. The use of closed-circuit television systems that now exist in some school systems can provide a built-in switchboard network for displaying to any observed classroom a feedback pattern. This system would continue to use the main CRT display and switch only the televised feedback image into the observed classroom.

One alternative to visual feedback is the further development of an auditory feedback source through which the teacher trainee would receive auditory feedback through a small earphone or some other similar device on his person. The method of transmission of the feedback material can either be through a direct connection to the teacher or transmitted from a corresponding FM transmitter to an FM receiver concealed on the teacher. The feedback information to be transmitted could originate either from a supervisor, who would be able to interpret and transmit, auditorily, the feedback information generated from the CATTS program, or from messages preprogrammed on an audio tape recorder, placed under computer control. These prerecorded feedback messages could be selected for classroom transmission by the computer as a result of the computer's detection of exceeded preprogrammed feedback parameters calculated during the on-line collection of CATTS data. The prerecorded auditory feedback method is closely related to the visual feedback method in that a supervisor is not required to interpret the computer's feedback data before it reaches the teacher. The advantage, in this case, is the protection from possible contamination of feedback information from a supervisor's misinterpretations.

A different approach to the observer-coding method found in the original CATTS is to supply the students or subjects with desk mounted pushbuttons, allowing the instructor to solicit responses to questions, attitudes, etc., during a class session. The result of these selections can then be displayed or combined with other observer data and fed back to the instructor to illustrate how the class is reacting to the material being presented.

Output developments in relation to hard-copy print-outs of descriptive statistics or data search-and-retrieval programs have been focused upon the direct connection of memory storage to a large computing facility. When print-outs of tabulated data are required, a high-speed line printer is more efficient than a slow speed teleprinter. The processing principle used here is that the raw observation data is first collapsed and simple percentage functions are extracted from the data for feedback display purposes; the raw data then is stored on magnetic tapes after the initial data collection. The main computing facility then is called and a data transfer onto a disk savefile is executed. A call to an existing tabulation-list-and-search program, also on savefile, to do work on the new data then is summoned and a printout of the data is retrieved at a later time. This external processing approach would take full advantage of the flexibility and on-line capabilities of a small computer and the large batch-processing advantages of a large computer.

CHAPTER 7

RELATED TRAINING LABORATORY DEVELOPMENTS

Improvements which stem from and parallel basic CATTS research and development are being formulated towards the establishment of a general training laboratory. A training laboratory facility would provide the participant, in addition to CATTS applications, with opportunities for experiences in developing observation and teaching skills through interactive simulations of classroom environments. An interactive training laboratory relies greatly upon the control of a real-time computing facility such as that used in CATTS applications, together with the compliment of large system processing found in shared-time remote computer systems. The key to the development of a related training laboratory is the intercommunication and common organization of the total system.

CONCODE Systems as a Training Device

Skill development in interaction analysis systems can be introduced, taught, and maintained by adapting the CONCODE program (explained in a previous section) into a simple consensus system which detects correct or incorrect coder tallies and regulates coder timing routines. By operating CONCODE as a simple control and detection system, coders are placed in a cooperative situation where a consensus of opinion about what is observed on video tape is necessary for the continuing of the video tape sequence. Figure 7.1 (p. 92) illustrates, in general, the sequences of operations involved with the consensus principle used as a training device. Precoded video tapes together with remote recorder playback control allow complete automation of the training sequence when using a computer regulated system.

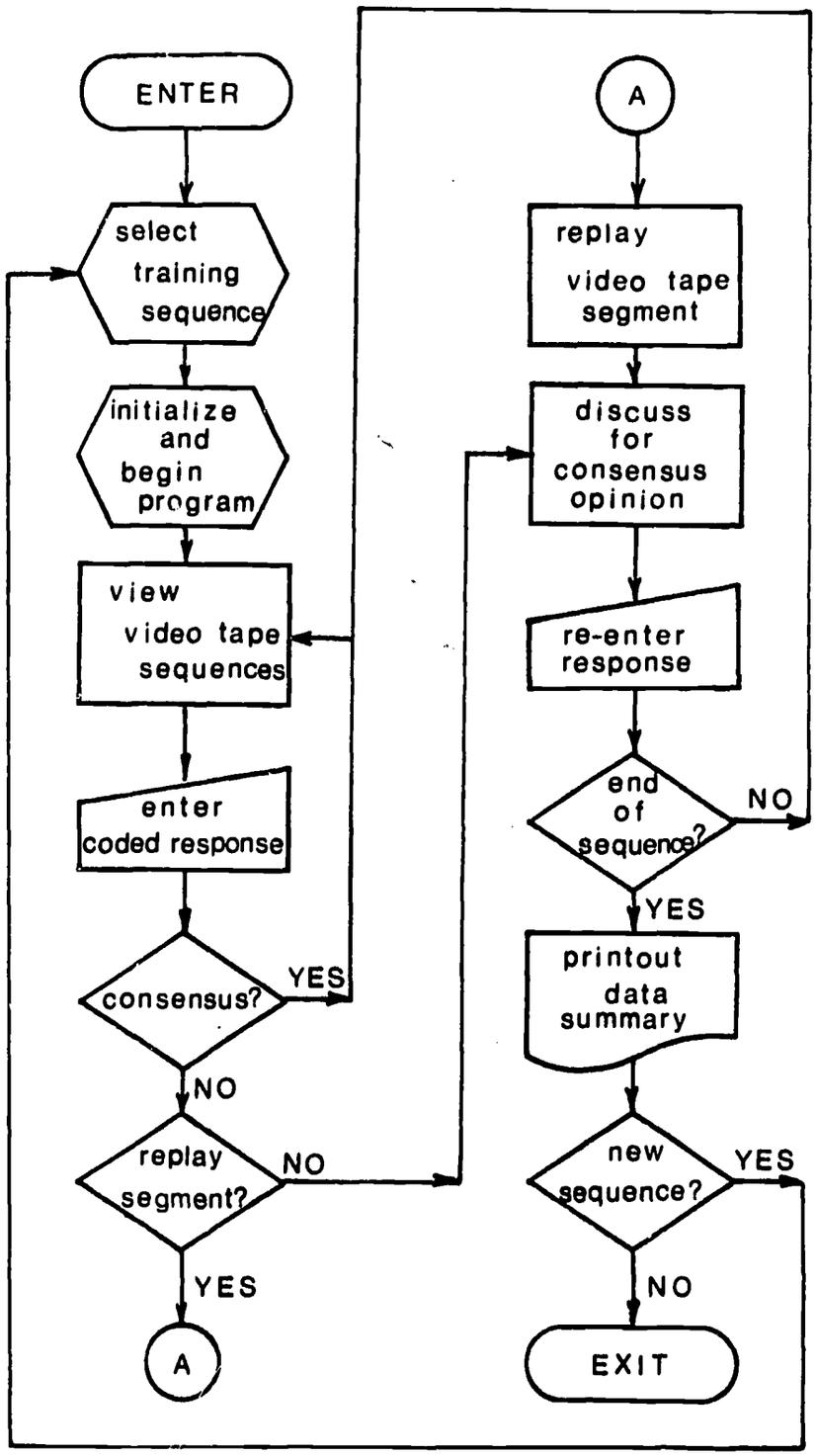


Fig. 7.1 System diagram of CONCODE as a training device.

A simple electro-mechanical relay CONCODE system is also feasible for limited applications in coder training. A scheme for a basic system together with a system flow chart can be found in Figures 7.2 and 7.3 (pp. 94 & 95). This relay controlled system is limited to detection of consensus between coders only and cannot provide the total control of the training situation as found in the computer based CONCODE system. The advantages of a simple relay system reside chiefly in its inexpensive cost and portability. This system could be used effectively as a remote training device for maintenance of observation skills.

Storage and Replay of Training Packages

During the initial development of CATTS, selected samples of interaction analysis data which had been collected during real-time classroom observations were stored for later retrieval on magnetic tape. Because portions of the main CATTS program system operate only on collected data, the ability to re-input previously collected data for reanalysis and replay provides a data for unlimited information retrieval applications. By systematically collecting previous interaction data and displays, together with video tapes of classrooms from which the data was collected, trainees may select for retrieval either real-time replays of the classroom coding situation or summary data printouts and displays. Figure 7.4 (p. 96) outlines the system flow chart which would be in an interactive program mode with the selected file material under the control of the main CATTS feedback subroutine. With the additions of video-taped sequences, actual classroom observation sessions can be replayed for further analysis or simulated training experiences.

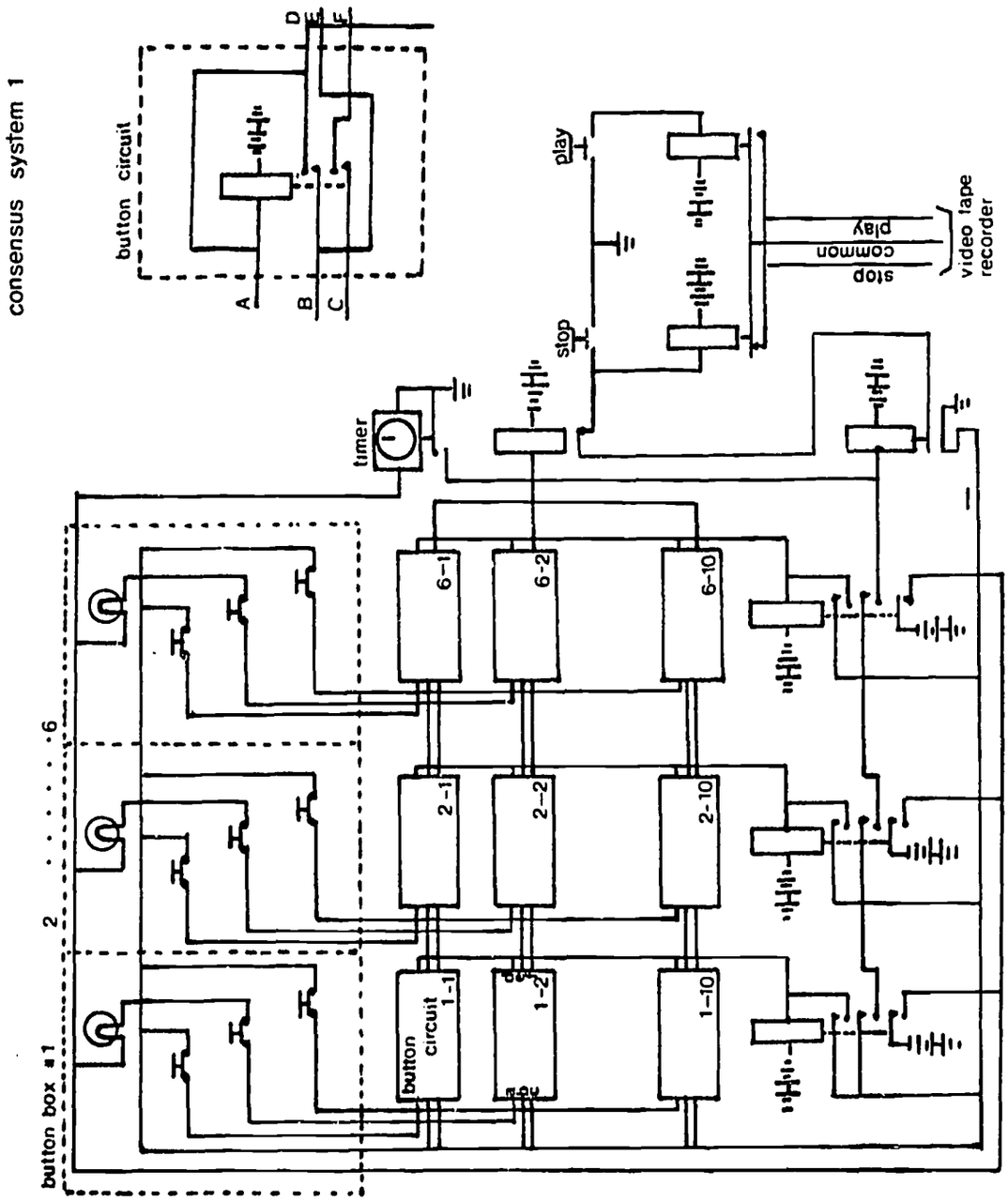


Fig. 7.2 Schematic diagram of an electro-mechanical CONCODE unit.

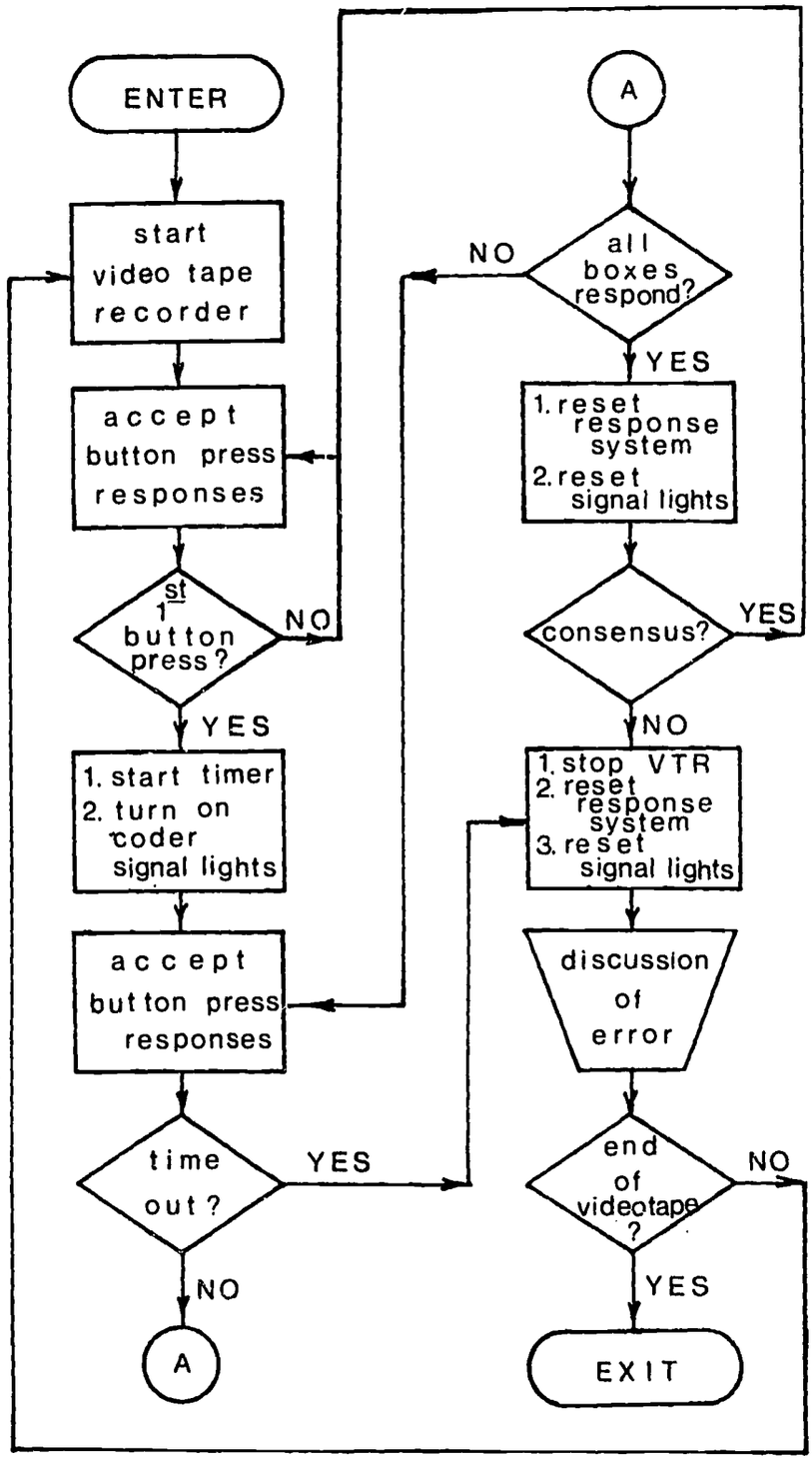


Fig. 7.3 Flow chart of electro-mechanical CONCODE unit.

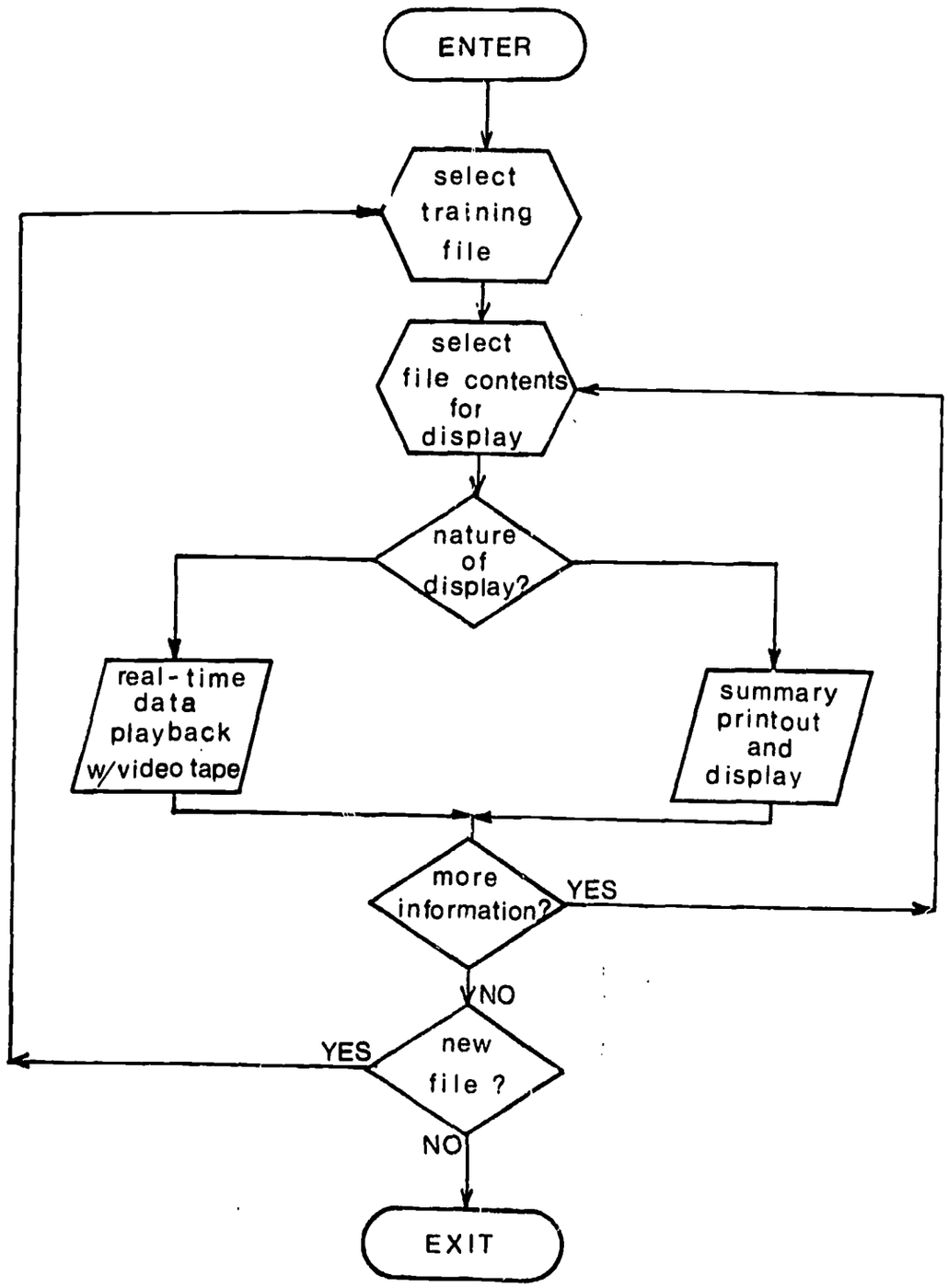


Fig. 7.4 Diagram of storage and retrieval system.

Interactive Training Modules

An attempt to employ video tape classroom segments coupled with computer regulated CAI type of interactive programming and control for the development of an interrelated series of training modules is also currently in process. A person or team of persons can view a segment of video tape; then they are requested to make different judgments about that segment according to a step-by-step sequence which increases in difficulty. These judgments can be evaluated later and assigned an evaluative score. A participant would be asked to react to selected content areas which must be detected from various educational environments presented on video tape. An example of the possible reactions which could be requested from participants in response to video tape situations is outlined in Table 7.1 (p. 98). The presented video tape segments may illustrate such situations as different social climates in the classroom, teacher behaviors, curriculum content,or motivational approaches. The system will request responses from participants according to different levels of difficulty. Requests in categories A and B require simple "yes" or "no" responses. Categories C, D, and E provide multiple choice responses which then can be evaluated for the best answer. Categories F and G have no actual correct answer but instead provide participants an opportunity to discuss fully all implications of their decisions. Figures 7.5a, b, and c (pp. 99, 100 & 101) are system diagrams providing a general outline for presentations of the training modules. By operating in an interactive mode, the system can provide access to information files and scorekeeping procedures. The on-line control capabilities of a small digital computer system permits

Table 7.1

Outline of Requested Categories for Use in Training Modules

	Request Category	Form of Request	Evaluation Level
Level I	A. Perception	A basic recognition of environment.	correct/incorrect
	B. Discrimination	A correct identification of environment.	correct/incorrect
Level II	C. Diagnosis	Diagnosis of the specific implications of the environment.	multiple choice
	D. Anticipation	What will happen next?	multiple choice
	E. Prescription	Prescribe the environment.	multiple choice
Level III	F. Generation	What would you do?	no correct answer
	G. Evaluation	How would you evaluate?	no correct answer

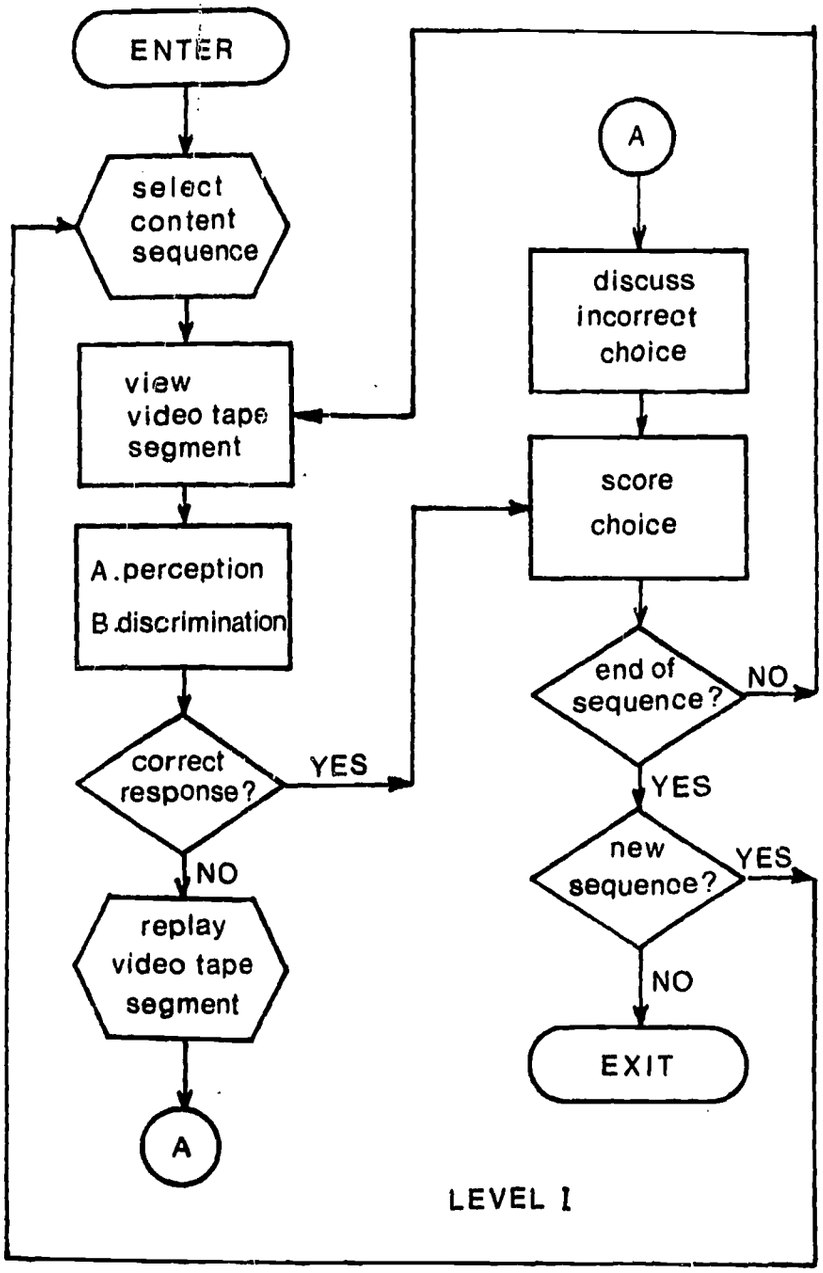


Fig. 7.5a Level I: Learning laboratory module system.

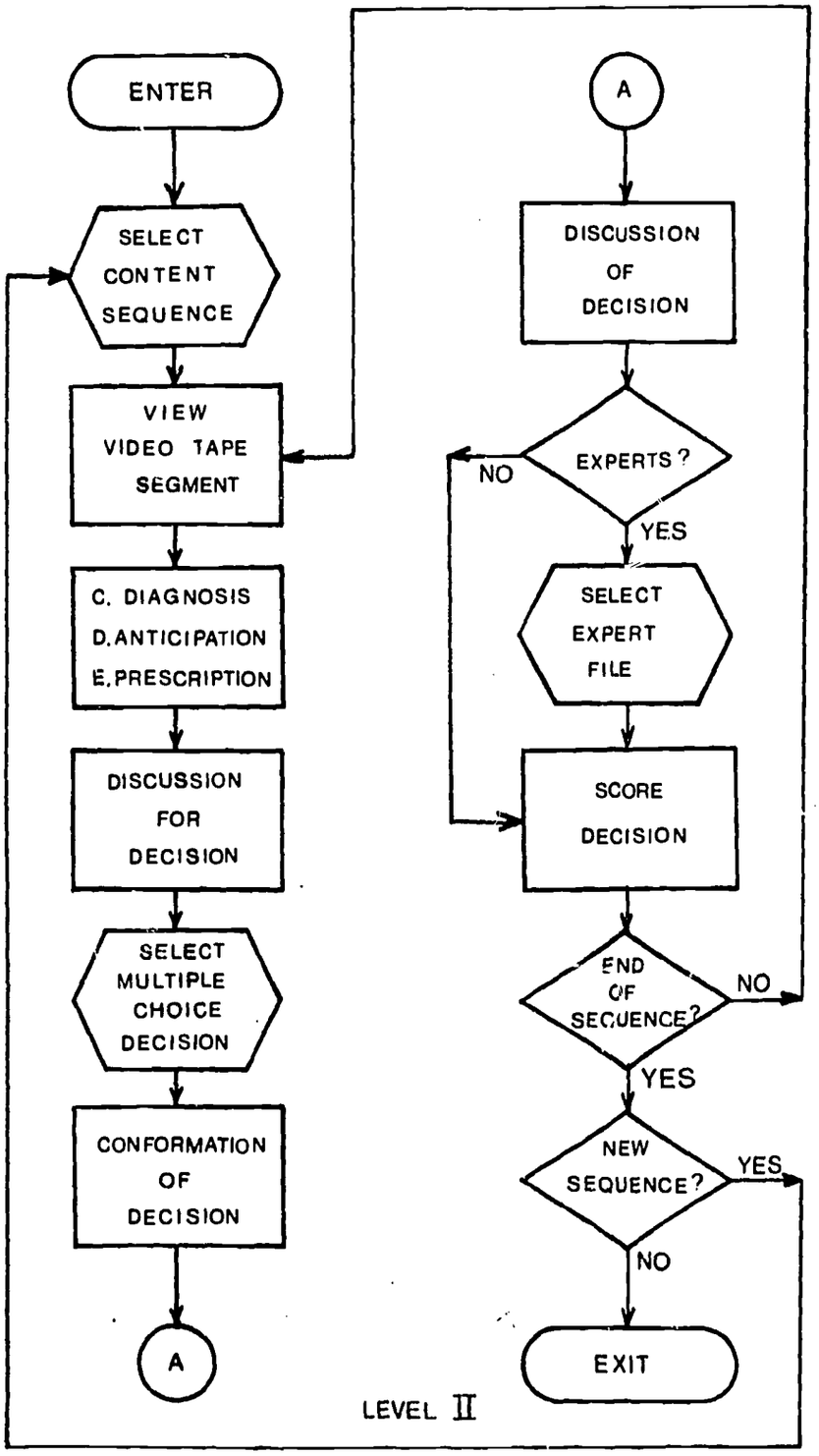


Fig. 7.5b Level II: Learning laboratory module system.

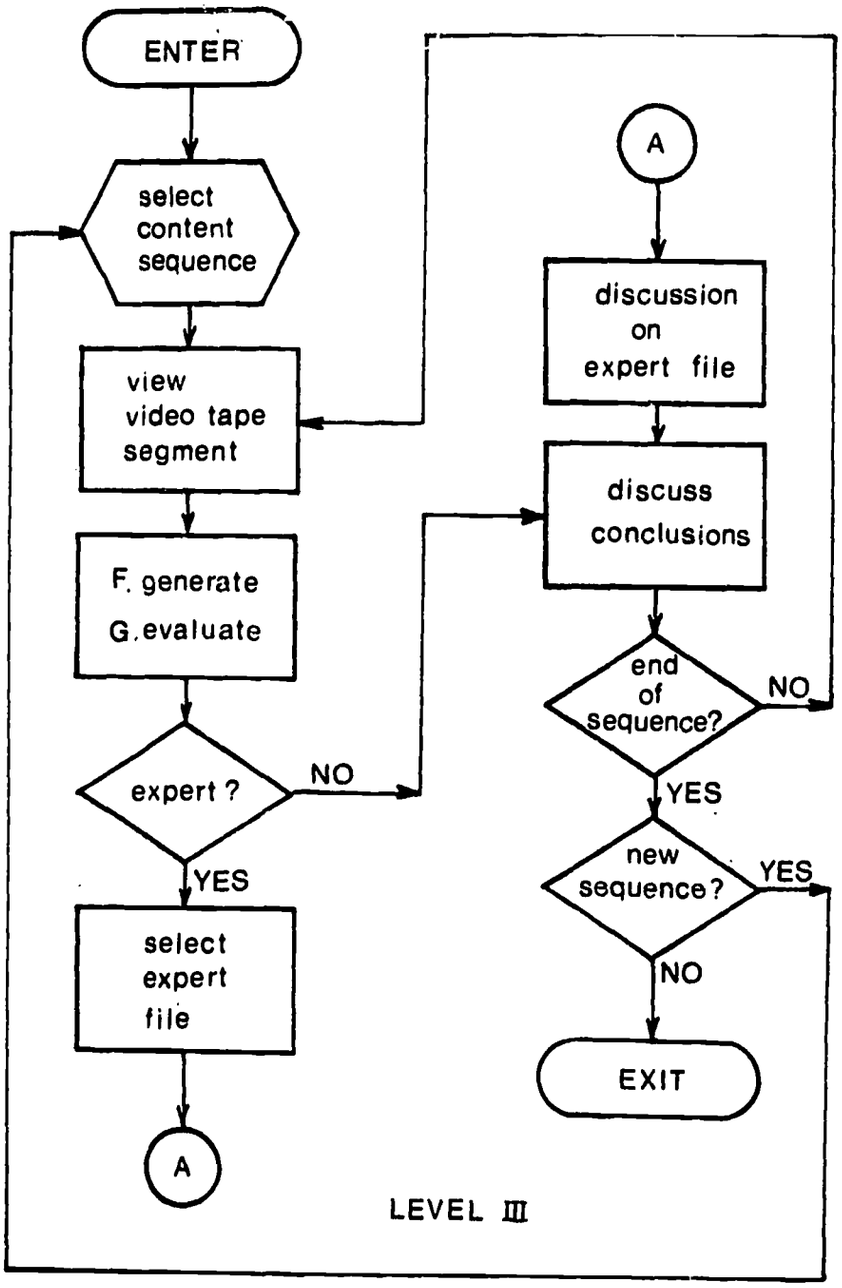


Fig. 7.5c Level III: Learning laboratory module system.

hardware manipulation for video playback control.

CATTS Simulation Game

A final step in the training module procedure could be the playing of a simulated CATTS game. Participants would be presented with parameters which describe a total educational environment. The system then questions the players' knowledge or reactions on different content areas contained in the selected environment. Responses are scored according to the complexity of the response. When criterion in any one content area is met, a new content area is chosen or the game is over. Scores can be calculated for the person who finished all content areas first; the person with the most points and/or the person with the highest single content score wins. An example of such a system can be found in Figure 7.6 (p. 103). The possibilities of such an application are endless; a next possible thrust is to apply simulated treatments upon an educational environment which, in turn, would generate the resulting effects of the applied treatments.

Affective State Feedback

Moving further ahead with possible technical applications to training techniques, technology currently exists for the computer to "sense" the emotional reactions of trainees through the monitoring of autonomic responses, and to feed back to them their affective states through real-time computer generated displays. A prototype system of this nature which had been initiated at the Center for Research on Language and Language Behavior in Ann Arbor, Michigan, by Dr. David Katz and Dr. Melvyn

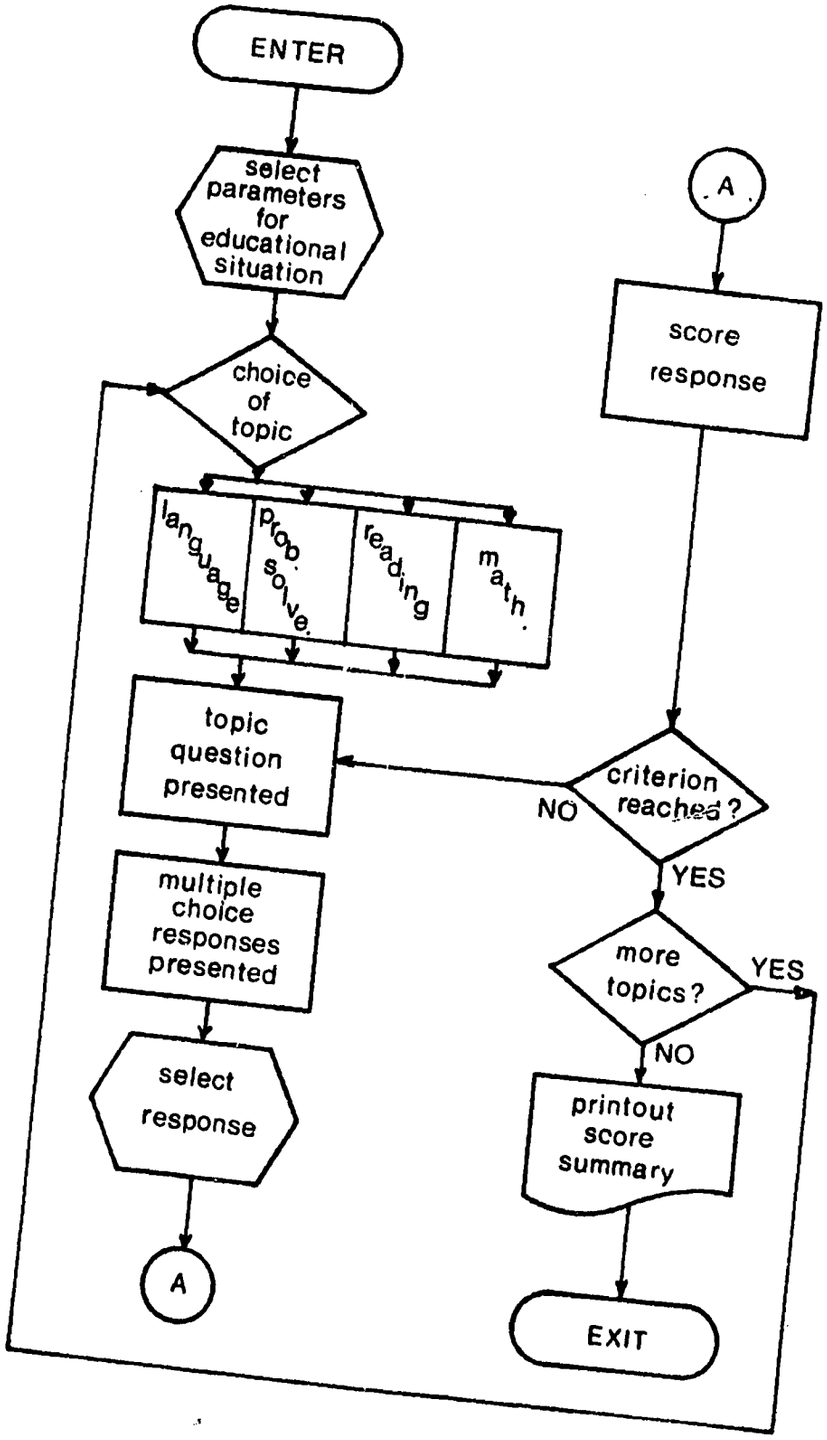


Fig. 7.6 System diagram of the CATTS game.

Semmel, explored the effects of a form of automatic relaxation therapy (desensitization therapy) through the subject's attempt to control his own affective states from visual feedback. The application of this principle in the near future may be to gather autonomic response information from teachers and/or trainees during the act of teaching children, and to provide them with feedback information relative to their arousal state.

REFERENCES

Anderson, C. C., & Junka, S. M. Teacher education: Some problems and a proposal. Harvard Education Review, 1963, 33, 1-22.

Bellack, A. A., Hyman, R. T., Kliegard, K. M., & Smith, F. L. The language of the classroom. Part II. New York: Columbia University, Teachers College, Cooperative Research Project No. 2023, 1966.

Blatt, B. The measurement and modification of the behavior of teachers. Mental Retardation, 1964, 2, 339-344.

Blatt, B. The preparation of special education personnel. Review of Educational Research, February, 1966, 151-159.

Cain, L. F. Special education moves ahead: A comment on the education of teachers. Exceptional Children, 1964, 30, 211-217.

Collet, L., & Semmel, M. I. The analysis of sequential classroom behavior. Unpublished paper, University of Michigan, Office of Research Services, School of Education, 1970.

Cruickshank, W. M. Current educational practices with exceptional children. In W. M. Cruickshank and G. O. Johnson (Eds.), Education of exceptional children and youth. Englewood Cliffs, New Jersey: Prentice Hall, 1967.

Dolly, D. G. Teaching patterns of mothers of trainable mentally retarded children. Unpublished thesis, University of Michigan, in preparation.

Dunn, L. M. Special education for the mildly retarded--Is much of it justifiable? Exceptional Children, September, 1968, 5-21.

Fink, A. H., & Semmel, M. I. Indiana behavior management system--II, observers' training manual. Bloomington, Indiana University, Center for Innovation in Teaching the Handicapped, 1971.

Flanders, N. Interaction analysis in the classroom: A manual for observers. Ann Arbor: University of Michigan, School of Education, 1964.

Gallagher, J. J. Productive thinking of gifted children. U. S. O. E., Cooperative Research Report No. 963, 1965.

Gallagher, J. J. New directions in special education. Exceptional Children, 1967, 33, 441-448.

Guilford, J. P. The structure of intellect. Psychological Bulletin, 1956, 53, 267-293.

Guskin, S. I., & Spicker, H. H. Educational research in mental retardation. In N. I. Ellis (Ed.), International review of research in mental retardation, Vol. III, New York: Academic Press, 1968, 217-273.

Jamison, D., Ball, J., & Potter, J. Personal communication--Preliminary draft of paper entitled, "Communication Economics of Interactive Instruction for Rural Areas: Satellite versus Commercial Telephone Systems", 1971.

Kreider, J. M. The effect of computer assisted teacher training system feedback on increasing teacher use of pupil ideas with EMR children. Unpublished doctoral thesis, University of Michigan, 1969.

Lynch, W. W., & Ames, C. Individual cognitive demand schedule, observers' training manual. Bloomington, Indiana University, Center for Innovation in Teaching the Handicapped, 1971.

Malpass, L. F. Automated instruction for retarded children. American Journal of Mental Deficiency, 1964, 69, 405-412.

Medley, D. M., & Mitzel, H. E. Measuring classroom behavior by systematic observation. In N. L. Gage (Ed.), Handbook of research on teaching. Chicago: Rand McNally, 1963, 247-328.

Minskoff, E. H. An analysis of the teacher-pupil verbal interaction in special classes for the mentally retarded. Unpublished doctoral dissertation, Yeshiva University, New York, 1967.

Olson, J. L., & Hahn, H. R. Our approach to preparing teachers of the mentally retarded. High School Journal, 1964, 48, 191-197.

Schmitt, J. S. Modifying questioning behavior of prospective teachers of mentally retarded children through a computer assisted teacher training system (CATTS). Unpublished doctoral thesis, University of Michigan, 1969.

Semmel, M. I. Project CATTS: A computer assisted teaching training system. In Van Teslaar (Ed.), Studies in language and language behavior, VII, Center for Research on Language and Language Behavior, University of Michigan, U. S. O. E. Report, 1968, 59.

Semmel, M. I., & Kreider, J. The relationship of pupil-teacher interactions in classrooms for the TMR to pupil gain in communication skills. Bloomington, Indiana University, Center for Innovation in Teaching the Handicapped, Technical Report 3.22, 1971.

Simon, A., & Boyer, E. G. (Eds.) Mirrors for behavior: An anthology of classroom observation instruments. Philadelphia: Research for Better Schools, 1970.

Smith, K. U., & Smith, M. F. Cybernetic principles of learning and educational design. New York: Holt, Rinehart, & Winston, 1966.

Stolurow, L. M. Principles for programming learning materials in self-instructional devices for mentally retarded children. Final Report, University of Illinois, Urbana, 1963.

VanEvery, H. Modifying therapy pattern used by aphasia therapist trainees by means of a computer assisted system providing in situ feedback. 1971.

Weaver, P. Effects of a computer assisted teacher training system and teacher expectancies on teacher-pupil verbal interactions with EMR children. Unpublished doctoral thesis, University of Michigan, 1969.