INFORMATION TO USERS

This material was produced from a microfilm copy of the original document. While the most advanced technological means to photograph and reproduce this document have been used, the quality is heavily dependent upon the quality of the original submitted.

The following explanation of techniques is provided to help you understand markings or patterns which may appear on this reproduction.

1. The sign or “target” for pages apparently lacking from the document photographed is “Missing Page(s)”. If it was possible to obtain the missing page(s) or section, they are spliced into the film along with adjacent pages. This may have necessitated cutting thru an image and duplicating adjacent pages to insure you complete continuity.

2. When an image on the film is obliterated with a large round black mark, it is an indication that the photographer suspected that the copy may have moved during exposure and thus cause a blurred image. You will find a good image of the page in the adjacent frame.

3. When a map, drawing or chart, etc., was part of the material being photographed the photographer followed a definite method in “sectioning” the material. It is customary to begin photoing at the upper left hand corner of a large sheet and to continue photoing from left to right in equal sections with a small overlap. If necessary, sectioning is continued again — beginning below the first row and continuing on until complete.

4. The majority of users indicate that the textual content is of greatest value, however, a somewhat higher quality reproduction could be made from “photographs” if essential to the understanding of the dissertation. Silver prints of “photographs” may be ordered at additional charge by writing the Order Department, giving the catalog number, title, author and specific pages you wish reproduced.

5. PLEASE NOTE: Some pages may have indistinct print. Filmed as received.

University Microfilms International
300 North Zeeb Road
Ann Arbor, Michigan 48106 USA
St. John's Road, Tyler's Green
High Wycombe, Bucks, England HP10 8HR
BROWN, James Robert, 1951-
AN AUTOMATED SACCADe ANALYSIS SYSTEM.
The Ohio State University,
Ph.D., 1977
Engineering, biomedical

University Microfilms International, Ann Arbor, Michigan 48106
AN AUTOMATED SACCADIC ANALYSIS SYSTEM

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in the Graduate
School of The Ohio State University

By
James Robert Brown, B.E.E., M.S.

********

The Ohio State University
1977

Reading Committee:  
Professor R. B. Lackey  
Professor D. T. Davis  
Professor R. M. Campbell

Approved By
Richard M. Campbell
Advisor
Department of Electrical Engineering
May 9, 1951        Born - Columbus, Ohio
1973             B.E.E., Ohio State University, Columbus, Ohio
1973             M.S., Ohio State University, Columbus, Ohio
1973-1975        Research Assistant, Department of Biophysics, Ohio State University, Columbus, Ohio
1975-1977        Research Assistant, Institute for Research in Vision, The Ohio State University, Columbus, Ohio

FIELDS OF STUDY

Major Field:     Biomedical Engineering

   Studies in Instrumentation. Professor R. M. Campbell
   Studies in Biological Modelling. Professor H. R. Weed
TABLE OF CONTENTS

VITA .................................................. 11
LIST OF TABLES .......................................... v
LIST OF FIGURES ......................................... vi
INTRODUCTION ........................................... 1

Chapter

I. THE AUTOMATED SACCADe ANALYSIS SYSTEM ............ 5
   System Overview ...................................... 5
   Saccade Target Deflector .............................. 7
   Cornsweet-Crane Purkinje Image Eyetracker .......... 11
   Computer Hardware and Software ...................... 13
   Digital Plotter ..................................... 19

II. EXPERIMENTAL RESULTS ............................... 20
   Experimental Conditions ............................. 20
   Tests ................................................. 21
   Subjects ............................................. 22
   Results .............................................. 23
   Discussion .......................................... 24
   Summary ............................................. 26

APPENDIX

A: SACCADe TARGET DEFLECTOR ............................ 59
   Introduction ........................................ 59
   Specifications, Inputs, Outputs ..................... 61
   Controls and Indicators ............................. 65
   Operating Instructions ............................... 68
   Circuit Operation ................................... 72
   Calibration ......................................... 75
   Signal Interconnection List ......................... 77
<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>B: CORNSWET-CRANE PURKINJE IMAGE EYETRACKER.</td>
</tr>
<tr>
<td>90</td>
<td>Theory.</td>
</tr>
<tr>
<td>93</td>
<td>Eyetracker Operation.</td>
</tr>
<tr>
<td>95</td>
<td>Modifications.</td>
</tr>
<tr>
<td>97</td>
<td>Eyetracker Measured Specifications.</td>
</tr>
<tr>
<td>100</td>
<td>C: SYSTEM SOFTWARE.</td>
</tr>
<tr>
<td>100</td>
<td>Hardware.</td>
</tr>
<tr>
<td>104</td>
<td>Software Overview.</td>
</tr>
<tr>
<td>107</td>
<td>ADIS01: Analog dat input and storage program.</td>
</tr>
<tr>
<td>140</td>
<td>APLOTO: Data file plotting program.</td>
</tr>
<tr>
<td>163</td>
<td>SPROUT: Saccade processing routines.</td>
</tr>
<tr>
<td>194</td>
<td>SSFLOT: Saccade summary file plotting program.</td>
</tr>
<tr>
<td>213</td>
<td>LIST OF REFERENCES.</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table 1 - Signal interconnection list.
LIST OF FIGURES

Fig. 1 - Automated saccade analysis system. Hardware and hardware interconnections.

Fig. 2 - Automated saccade analysis system. Software and system data flow.

Fig. 3 - X and Y axis rotations as functions of time and Y axis rotation as a function of X axis rotation. Subject JRE response to a 120 degree target deflection.

Figs. 4-35 - Subject response data.

Fig. 36 - Selection timing diagram for radius.

Figs. 37-44 - Schematics and circuit board layouts for the saccade target deflector.

Fig. 45 - Schematic representation of the positions of the first and fourth Purkinje images during rotation and translation of the eye.

Fig. 46 - Automated saccade analysis system, hardware and major hardware interconnections.

Fig. 47 - Automated saccade analysis system, system software and data flows.
INTRODUCTION

The nature of human saccadic eye movements has been extensively studied in the last few decades. A number of basic features of these rapid eye movements may readily be observed. Saccadic eye movements, or saccades, are rapid step movements of rotation used to reposition the eyes to new fixation points. Their durations extend to tens of milliseconds and their magnitudes vary from a few minutes of arc to several tens of degrees. In most situations it is nearly impossible to redirect one’s gaze without saccadic movements; even a subjectively smooth slow movement between two fixed points takes place as a series of small saccades.

The relative importance of saccadic eye movements is well reflected in the literature. Hundreds of articles have appeared, presenting the results of studies of saccadic velocities, durations, muscle forces, curvatures, control system aspects, and relations to neurological disorders. Along the way, many ingenious methods of measuring saccadic and other eye movements have been devised. Most of these methods have severe limitations, testifying to the difficulty encountered in measuring eye rotations. These limitations result in difficult, time consuming experimental conditions, restrict the scopes of laboratory studies, and render nearly impossible quantitative clinical studies and tests.
Limitations are primarily instrumentation inadequacies characterized by compromises between accuracy and ease of use. This is especially evident in measurements of vertical eye rotation—to the extent that almost all of the saccadic studies to date have been restricted to horizontal eye movements. The highly accurate methods of measuring eye rotation usually involve touching the subject's eye, often measure movements of the non-viewing eye in monocular visual tasks, and may require short training periods for the subjects involved. These "contact lens" methods are, however, extremely accurate, and with careful use can provide resolution of a few seconds of arc. Other techniques have been devised which do not touch the eyeball but sacrifice accuracy. These, generally speaking, are variations on the theme of electronically monitoring the position of eyeball surface reflections using various illumination schemes and arrangements of photocells. Accuracy with these methods is limited to a few minutes of arc and there are often special problems for vertical rotations.

Problems do not end with the method of measurement. Having chosen or devised some way to measure eye rotation, the researcher must eventually process whatever measurements are recorded. More often than not, data is stored on chart-recorder paper or magnetic tape and analyzed by hand at a later time, an altogether tedious process. Computer analysis, however, is difficult due to the present day inadequacies of software pattern recognition techniques and the rarity of investigators facile (or even interested) in both eye research and computer programming. In addition, the previously
mentioned instrumentation difficulties often serve to restrict the quantity of data to the point that manual analysis requires less time than would be required to program a computer for the equivalent task.

All of the aforementioned problems have resulted in surprising gaps in the knowledge of human saccadic eye movement. Already mentioned is the fact that most knowledge has been gleaned from studies of horizontal saccades only; published data is not available, for instance, to verify whether or not response latencies to target steps are different for off-horizontal angles. In addition, although saccade durations have been studied, data about settling time to the target following a target step deflection is not available in any publications uncovered by this author.

The automated saccade analysis system described herein is a complex instrument which addresses itself to the resolution of many of the problems outlined above. The goal of this work was to design and construct a system which would allow the experimenter to easily generate, record, and analyze saccadic responses of untrained subjects to step target deflections at a multitude of angles and magnitudes. The system was to use a Cornsweet-Crane eyetracker to monitor eye rotation and a Digital Equipment Corporation PDP-8/E computer to store and process eye rotation data.

To create a complete operating system it was necessary to design and construct a target generation instrument, modify the eyetracker to reduce signal noise and to permit operation with
subjects wearing eyeglasses, and design and write the entire computer software package to manage data input, output, and processing for saccadic response data.

Present capabilities of the system are such that (1) subjects view the test target with normal binocular vision and the monitored eye is not touched in any way, (2) untrained subjects, even those wearing eyeglasses, may be tested with relative ease, (3) computer analysis of recorded responses to step target deflections will extract response latency and settling times, and (4) rapid testing may be done — total system efficiency, including the subject's time (with rest periods) and all processing time, is about one saccade per minute for large batches of deflection trials.

Preliminary testing of the saccade analysis system has been done, recording and analyzing nearly 1600 saccadic responses made by four subjects. The system performs well and is easy to use for both experimenter and subject. Results from these preliminary trials indicate that saccadic responses to target step deflections are not the same for different directions, nor are they the same for all subjects. These results are presented later. No attempt will be made to generalize the data presented to include the populace as a whole; the subject sample is much too small.
System Overview

The automated saccade analysis system is a rapid, highly accurate, easy to use, computer based system, providing the researcher with the means to evaluate human saccadic eye rotation responses to two-dimensional step deflections of a point target. Subjects may be untrained, and within broad limits, may, if necessary, wear their own eyeglasses during tests. The system measures two-dimensional angular rotation of the subject's right eye while the subject views the target, a very small, bright spot on an oscilloscope screen. Binocular vision is permitted. During testing, the subject is instructed to fixate on the target as well as possible and each saccade trial consists of a step target deflection from the center of the screen. Target deflections may be generated along any of twenty-four fixed angular directions and out to four fixed radii from the central position. Deflections may be preassigned by the experimenter or sets of deflections may be generated at random by the saccade target deflector circuitry. The subject's "step" response to each target deflection is monitored, encoded, and stored by the system computer for later analysis. Computer analysis of each response yields first saccade latency and response settling times to four different radii about the target position, as well as performing regularity checks to insure "normalcy" of each response.
analyzed. Special plotting programs allow the user to plot these results for batches of responses as either functions of deflection angle or deflection radius. Total system efficiency, from seating the subject to plotting results, is about one saccadic response per minute for large batches of trials.

Major hardware components of the saccade analysis system are (1) the saccade target deflector and oscilloscope, to provide visual stimuli to the subject, (2) a modified Cornsweet-Crane eyetracker, to monitor the angular rotations of the subject's eye movement response, (3) a FDP-8/E computer, to collect and analyze response data, (4) a dual flexible disk system to provide mass data storage, and (5) a digital plotter, to plot raw response data or the results of computer processing of many responses.

Software components developed specifically for the saccade analysis system include the four major programs ADISO1, APLCT0, SPROUT, and SSPLOT. All programs are FORTRAN II with intermixed SAER (Digital Equipment Corporation assembly language) programming, and all programs are called and controlled by teletype keyboard commands. ADISO1 programming manages computer collection and disk data file storage of eyetracker output data and APLCT0 is a plotter package used to plot the raw data stored in this data files. SPROUT processes batches of ADISO1-created data files of saccade responses to target deflections, storing the results as a summary data file on disk. SSPLOT is a special plotter program designed to plot the summarized data stored in a SPROUT-created saccade summary data file.
Hardware and software component operation with respect to the entire system is discussed in the following sections. System hardware and hardware interconnections are shown in block diagram form in Fig. 1; system software data flow is illustrated in Fig. 2.

**Saccade Target Deflector**

The saccade target deflector (STD) and a Tektronix 502A oscilloscope provide visual stimuli for the subject. The subject is instructed to fixate and follow as best as possible the target, a bright, sharply focused spot displayed on the oscilloscope screen. The STD provides X and Y axis oscilloscope drive voltages to produce either preset or pseudo-random step deflections of the target from its initial center position. Each deflection is initiated by the experimenter with an STD front panel switch. Deflections may be generated at any of 24 angles from 0 to 345 degrees in 15 degree increments. Zero degrees is referenced to the positive x-axis of the oscilloscope display; 90 degrees is along the positive y-axis. Deflection magnitudes may be any of four radii out from the center, the radial increments being equal. For all work here the radii and oscilloscope distance were adjusted to provide radii that subtended 1, 2, 3, and 4 degrees at the subject's eye. The STD can also provide a warning buzzer signal just prior to each target deflection and this feature was used for all testing done with subjects. Warning buzzer time and target deflection time are adjustable over convenient ranges.
FIG. 1: Automated saccade analysis system. Hardware and hardware interconnections.
FIG. 2: Automated saccade analysis system. Software and system data flow.
The STD accepts "blink" and "track" signals from the eyetracker, and provides trigger and data signals to the computer. The trigger pulse is generated at the instant of target deflection and is used to initiate data sampling by the computer of the eyetracker $X (H4)$ and $Y (\wedge 4)$ axis rotation signals. The subject's response to the target deflection is thus monitored from the onset of the deflection. The STD data outputs supply the computer with a "trial failure" bit and digitally encoded angle and radius information. This output data is valid during the target deflection time. The target angle and radius for a given trial can thus be input and stored with the subject's response to that deflection. The blink and track signals from the eyetracker enable the STD to monitor subject eyeblink or loss of track by the eyetracker. If either should occur during a deflection trial the STD registers the occurrence and sets the failure data line to the computer. The ADIS01 data input and storage program will detect this bit, resulting in all data for that trial being rejected and no corresponding data file output to disk.

For preliminary saccade testing as performed here, a single trial sequence proceeds as follows: (1) with the eyetracker monitoring the subject's right eye, the experimenter pushes the STD "GO" switch to initiate the trial; (2) the warning buzzer sounds for 1.5 seconds, alerting the subject to fixate on the target (still at its central position); (3) at the end of the warning, the target deflects and the computer is triggered to begin monitoring the subject's response; (4) the target remains deflected for 3 seconds and the computer samples
the subject's response for the first 2.5 seconds of the deflection time, (5) at the end of sampling, the computer reads the STD data outputs, checks the failure bit, and if no failure occurred proceeds to output the angle, radius, and response data to disk storage as a saccade data file.

The reader is referred to the appendix for a detailed description of the saccade target deflector. Included are operating instructions, explanations of capabilities and operating modes, output specifications, and schematic diagrams of the STD circuitry.

**Comsweet-Crane Purkinje Image Eyetracker**

The Purkinje image eyetracker was developed by T.N. Cornsweet and H.D. Crane and current models of the machine are produced on a custom basis by the Stanford Research Institute. The eyetracker directs a collimated infrared light beam into the eye and monitors the positions of two reflections, the first Purkinje image, reflected from the front surface of the cornea, and the fourth Purkinje image, reflected from the rear surface of the lens. These images move in concert as the eye translates but shift in opposite directions when the eye rotates. Their positions are optically tracked using servo driven mirrors and the servo signals, after suitable processing, provide electrical outputs $V_4$ and $V_4$; these voltages are directly proportional to the horizontal and vertical angular rotations of the eye.
Movement of the first Purkinje image is optically subtracted from the movement of both images to yield rotation signals free from translational artifacts. The eyetracker cannot, however, distinguish between an eye rotation that occurs because the target moved to the left and the corresponding rotation that would occur if the subject's head moved to the right while the target remained fixed. To reduce this source of translational artifact, subjects were held stationary by means of a dental wax biting bar. For all work done here the eyetracker was adjusted for each subject to yield $H_f$ and $V_f$ outputs of ±0.1 volt per ±1 degree of eye rotation from the center position, that is, when the subject was fixating the centered target.

Performance specifications for the eyetracker are listed below and the reader is referred to the appendix for a more detailed description of the eyetracker, its operating principles, modifications made by the author, and the methods used to measure the performance specifications.

RMS equivalent rotational noise, $H_f$ and $V_f$ outputs . . . . . 1.4 minutes of arc

Equivalent rotational slew rate, $H_f$ and $V_f$ outputs . . . . . 60 degrees/second

Response delay, $H_f$ and $V_f$ outputs . . . . . 2 milliseconds

Rotational linearity within a 4 degree radius circle, $H_f$ and $V_f$ outputs . . . . . error less than noise
Computer Hardware and Software

Various data is input to, processed by, and output from the PDP-8/E computer under the control of programs ADISO1, APLOTO, SPROUT, and SSFLOT, all developed specifically for this system. As shown in Fig. 1, the computer accepts analog eye rotation signals from the eyetracker and digitally coded angle, radius, and failure data from the saccade target deflector (STD). The $V_4$ and $V_4$ signals from the eyetracker are sampled and digitally encoded by the multiplexer and analog-to-digital converter. Sample timing is controlled by software via the computer real-time clock. The clock is triggered to begin analog sampling by the T2 TRIG OUT trigger pulse, generated by the STD at the onset of each target deflection. The STD also supplies the computer with encoded deflection angle, deflection radius, and trial failure data, all valid during target deflection time. The trial failure bit will be set true during a target deflection trial if the subject blinks or the eyetracker loses track lock, these conditions being signalled to the STD by the eyetracker blink and track output signals.

As shown in Fig. 2, computer input and storage of the above mentioned eyetracker and STD input data is controlled by the program ADISO1. This program provides two sampling modes: a single trial "one-shot" mode to input and store subject response data for one target deflection trial, and a recycle mode to automatically input and store data from sequences of trials. Data for each trial is stored
on disk FL2 as a data file whose name has been previously declared by the user. ADISOI sampling parameters are initialized to convenient values at startup but these may be changed, if desired, with user entered teletype commands. Teletype commands must also be used to declare filenames for disk storage of data and to initiate the one-shot or recycle sampling modes.

The one-shot mode sampling sequence for a target deflection is outlined below; timing values given are those used for all work here: (1) the computer begins one-shot mode by entering a clock trigger wait loop (2) the experimenter then initiates a target deflection with the STD "GO" switch, the warning buzzer sounds for 1.5 seconds, then the target deflects for 3 seconds (3) at the onset of target deflection the STD T2 TRIG OUT trigger pulse starts the computer real time clock running (4) the clock controls the multiplexer/converter and the H4 and V4 eye rotation signals (the subject's response) are each sampled every 5 millisecond sample period, each pair of samples being stored in an array in the computer core memory (5) sampling stops after 500 samples have been taken (2.5 seconds) (6) the computer then immediately reads the digital input data from the STD and checks the failure bit. (7) if the failure bit is set, no further action is taken on the data and control transfers back to the initial trigger pulse wait loop; a failure has occurred (8) for a normal successful trial the computer stores the angle and radius information at the start of the sample array along with coded sampling parameters, then outputs the entire array to
disk FL2 as a data file with the user entered filename (10) control is transferred back to the ADIS01 command mode, and the ADIS01 program is ready to accept new commands.

Operation in recycle mode begins as in one-shot mode. However, each time a data file is output to disk, the current filenumber part of the filename is incremented and control transfers to the trigger wait loop. In this way, up to 99 consecutively numbered data files may be stored, each data input and storage cycle initiated by a target deflection trial. Each ADIS01-generated data file contains a header with the following information about the data: multiplexer input channel numbers, sampling period, number of samples, and target deflection angle and radius. More detailed information about ADIS01 including a program listing may be found in the appendix.

Having generated a set of disk data files, each containing a subject response to one target deflection, the system user may plot this raw data using program AFLOTO. Normally these saccade data files are not plotted in raw form but are processed by SPROUT and the summarized results plotted using SSPlot. However, should the user desire raw data plots, AFLOTO may be used to plot any sampled channel as a function of any other channel or as a function of time, with full scaling capabilities, a "slow motion" feature, and other features discussed in detail in the appendix. To illustrate a typical subject response for one deflection trial, AFLOTO was used to generate the plots shown in Fig. 3. Shown are x and y axis rotations plotted as
Fig. 3 - X and Y axis rotations as functions of time and Y axis rotation as a function of X axis rotation. Subject JRB response to a 120 degree target deflection.
functions of time and a plot of y axis rotation as a function of x axis rotation, illustrating that this was a response to a 120 degree target deflection.

SPROUT is a program specifically designed to process ADIS01 data files containing two channel response data of the form illustrated by Fig. 3. SPROUT inputs these data files one at a time or may be instructed to automatically input consecutively numbered batches of data files, and for each file calculates latency and settling times to within 25, 20, 15, and 10 minutes of arc about the final position.

SPROUT calculates latency by starting at the beginning of the data record and calculating a velocity for each x-y sample pair. The latency period ends at that sample just preceding the sample for which the velocity exceeds a cutoff velocity, 20 degrees per second for all work here. To determine a settling time, SPROUT begins by calculating the "target" position, averaging the x and y data for the final 1 second of the 2.5 second data record. Then, starting at the beginning of the record, SPROUT calculates the distance from each sample pair to this final position. The settling time is determined as the time elapsing from the start of the record (target deflection) to the start of the first period of 0.1 second or longer for which the distance is within the settling radius. As an example, SPROUT calculations for the response shown in Fig. 3 yield a latency of 0.185 seconds, and settling times to 25°, 20°, 15°, and 10° are found to be 0.290, 0.430, 0.450, and 0.455 seconds respectively.
SPROUT also performs regularity checks on the data to insure that results from highly irregular data are not unknowingly included in the SPROUT output.

SPROUT output consists of a summary array which may be stored on disk FL2 as a saccade summary data file. Each entry in the summary array contains the following data about one saccade data file: the filename of the original saccade data file, the sampling parameters used to obtain the original data, the target deflection angle and radius used for that trial, and the calculated latency and settling times as discussed above. Thus the results from up to 63 individual trials may be stored on disk in compact form, saving storage space and making readily available results from many trials for later plotting by SSPLOT. SPROUT operation in controlled by keyboard commands that enable the user to process single data files or batches of data files and include the results in the summary array, delete entries from the array, and output the summary array to disk as a saccade summary data file. Processing consideration and methods are detailed in the appendix along with a more complete discussion of SPROUT operation and commands.

After using SPROUT to process and summarize saccade data files, the summary file data may be plotted using SSPLOT, a plotting routine designed to input SPROUT-created summary files and generate a variety of plots. The latencies and settling times in a summary file may be pointwise plotted as functions of angle or radius, with either radius or angle as a parameter, using any reasonable scale factors,
and with the data points plotted using any of five different symbols. Examples of SSPLOT output may be found in the next chapter, where data taken from four different subjects in plotted in detail. SSPLOT operation is discussed in more detail in the appendix.

**Digital Plotter**

Since the digital plotter is the output device of interest in this system, a few words are appropriate here. The plotter used is a model 130 Dataplotter, manufactured by Electronic Associates, Inc., West Long Beach, New Jersey. It is an x-y plotter with a single pen and has pen-up, pen-down, +X, -X, +Y, and -Y inputs cabled to a computer plotter interface. Under software control, this interface provides electrical pulses to raise and lower the pen and to step the pen along the axes. Step size on both axes is 0.01" and the maximum speed is 200 steps per second. Hardware and software was added by the author to allow the plotter to recover from attempted y-axis offscale drive. The plotter is also equipped with manual pen drive pushbuttons for initial pen positioning.
EXPERIMENTAL RESULTS

Experimental Conditions

During tests with the saccade analysis system, subjects were seated on a stenographer's chair and held in fixed position with respect to the eyetracker with a dental wax biting bar, clamped at a comfortable angle in a small three-axis machine tool bed. Subjects viewed the target with binocular vision and the eyetracker monitored the right eye; as such, a clear glass plate, part of the eyetracker, was located about 5 cm in front of this eye. The target in all tests was a sharply focused spot displayed on the screen of a Tektronix 502A oscilloscope with P2 phosphor. Spot brightness was adjusted to a comfortable viewing level with no halo. In its undeflected position the target was centered on the screen and centered in the visual fields of the eyetracker and the subject's right eye. The screen was positioned 56 cm from the cornea of the right eye. At this distance a 1 cm distance on the screen corresponds to 1 degree of eye rotation (4cm = 3.99 degrees) and the spot itself subtends about 2.5 minutes of arc. The oscilloscope front panel was covered with black cloth to eliminate afterimage distractions.

Tests were conducted in a dimly lighted room and subjects were permitted 15 minutes adaptation before each testing session. Tests for each subject were performed on two consecutive days, preceded
by a calibration session on a separate day. During this session
the eyetracker was alligned on the subject's eye and the horizontal
and vertical signal gains adjusted and offsets nulled while the subject
fixated selected points on an illuminated 1 cm oscilloscope screen
graticule; gains were adjusted to 0.1 volt per centimeter/degree
of eye rotation on both axes. The graticule was then removed and the
subject participated in a short set of practice trials.

Tests

Subjects were instructed to fixate the target when the warning
buzzer sounded and follow the target as best as possible during the
deflection period. Each target deflection trial was initiated by
the experimenter using the saccade target deflector. Each trial
consisted of a 1.5 second warning buzzer followed by a 3 second
target deflection; deflection was synchronous with the end of the
buzzer tone. After the 3 second deflection the target returned to
its original center position. A short interval then elapsed before
the next trial to allow the computer to transfer the subject response
data to disk storage, resulting in a trial repetition rate of about
15 seconds. Twenty-four target deflection angles, from 0 to 345
degrees in 15 degree increments, were used. The deflection radii used
were adjusted to subtend 1, 2, 3, and 4 degrees as seen by the
subject. Trials were conducted in sets of 24, corresponding to the
24 available deflection angles. For each set of 24 trials the saccade
target deflector was used in non-repeating random mode to generate
the deflection angles, i.e. the 24 angles were each used once but presented in a random sequence.

Trial sets were of two kinds: fixed radius and random radius. In a fixed radius set all 24 deflections were to a particular radius and in a random radius set the deflection radius for each trial was randomly selected by the saccade target deflector from among the four available choices. Eight sets of 24 trials were conducted for each subject on each of two consecutive days. On each day the eight sets consisted of four fixed radius sets, on at each radius, and four random radius sets. Fixed and random radius sets were alternated and the order of the fixed sets was reversed on the second day. In this manner data was obtained from each subject for two trials at all 96 deflection positions using random angle and fixed radius. In addition, an average of two trials for each position was obtained with both angle and radius randomly generated, thus providing a check on the effects, if any, of prior knowledge of radius. Intermixing fixed and random trial sets and reversing the trial set order on the second day was done to reduce the effects of boredom and fatigue.

Subjects

Four subjects, two males - JRB and DJS, and two females - LWG and KFW, all volunteers in their late twenties, were tested with the saccade analysis system. JRB and DJS had had prior experience with the system and LWG and KFW had not. Testing time for untrained subjects was approximately 90 minutes on each of the two days; a slightly
shorter time was required for the experienced subjects. Three
subjects, JRB, DJS, and LWG, were nearsighted and wore eyeglasses
during the tests; in addition DJS and KFW were slightly astigmatic
but subject KFW did not require or wear eyeglasses during the tests.

Results

Altogether, 384 responses for each subject were stored and
processed. Processing regularity checks forced a rejection of
approximately 2-5% of each subject's data, based on the criteria
mentioned with the SPROUT processing considerations in the appendix.
Eight SSFLOT-generated plots are shown for each of the four subjects
tested. These thirty-two plots are appended at the end of this chapter
as Figs. 4 through 35. Each plot shows both first saccade latency
(LTY) and settling time to 15° (ST15) for each response, all as
functions of deflection angle. ST15 is shown as it seems the most
useful of the four available settling times; ST25 and ST20 are for
larger radii and ST10 times are rendered nearly useless by small eye
movements after target acquisition. Note that as displayed for the
subject, 0 degree deflections are to the right, 90 degree deflections
are upwards, 180 degrees is to the left, and 270 degrees is downwards.
Each plot presents data for responses to a particular deflection
radius and separate plots are shown for each radius in both fixed and
random radius trials. Thus, data for each subject consists of four
plots, each showing all responses to 1, 2, 3, and 4 degree fixed radius
trials, respectively, and four similar plots, each showing, respectively,
all responses to 1, 2, 3, and 4 degree deflections compiled from all random radius trial sets.

All plots have x-axis scaling of 90 degrees per inch and y-axis scaling of 200 milliseconds per inch. The latency for each response is represented by a small square; settling time is represented as a small cross.

Discussion

Response latencies correspond, as expected, with those reported in the literature for horizontal saccades. Westheimer (1954), White and Eason (1962), and Saslow (1966a,b) have all observed latencies of 200-300 msec in a variety of random step-displacement experiments. White and Eason also reported a latency increase with radius for larger target displacements (10-40 degrees); this trend is not evident in the data presented here.

The striking increase in latencies for downward target deflections, seen in three subjects here, has not previously been reported in the literature and is quite a surprising result. This increase is quite evident in data from subjects DJS and KPW and appears to a lesser extent in subject JRB; subject LWG does not show the increased latency. No consistent change in the phenomenon is readily apparent with change in radius, nor are significant differences seen between the fixed and random radius trials. Indeed, as a check on fixed radius data, the random radius data seems only to show more scatter in the values obtained but no evident changes in other
Although these increased downward latencies might be accepted without question simply due to lack of published two-dimensional saccade studies, they are peculiar enough to warrant comparisons between raw response data and the plotted SPROUT-processed data. Such comparisons were made and no discrepancies were found; the phenomenon really exists and is not due to any hardware or software peculiarities.

Settling time to the displaced target has not been reported in the literature to date but no real surprises appear in this new data. The settling times tend to fall into bands of "immediate target acquisition" and "two-saccade target acquisition". Immediate acquisitions are those settling times occurring within a few tens of milliseconds after the latency and two-saccade acquisitions fall into a band 200-300 msec later. Immediate acquisitions are the result of a single, reasonably accurate saccade to the near vicinity of the target. As such the delays are quite compatible with saccade durations as reported by White and Eason (1962) and Robinson (1964). Two-saccade acquisitions occur as a result of a larger positional error in the first saccadic jump, corrected by the subject with another saccade after a latency-length delay has elapsed. Those subjects showing downward saccade latency increases naturally reflect this increase in immediate acquisition settling times but without further analysis no comment can be made about possible secondary effects of deflection angle on settling times.
Summary

An automated saccade analysis system has been developed to allow rapid evaluation of oblique saccadic response latencies and settling times to step target deflections. Naive subjects may be easily tested and subjects may wear eyeglasses and use normal binocular vision during testing. The experimenter may easily present target deflections at a multitude of angles and magnitudes and in random or pseudo-random sequences while the computer automatically collects and stores eye rotation response data for each trial. Computer software, in addition to managing the collection of data, has also been written to provide for plotting of raw response data, to process saccadic responses and extract latencies and settling times, and to generate pointwise plots of latencies and settling times for batches of saccadic responses. System speed and ease of use provide an overall efficiency, from subject to data plot, of about one response per minute for large batches of trials, easily an order of magnitude speed increase over other methods in use today.

Latencies and settling times for nearly 1600 total responses from four subjects have been computer calculated from raw response data and the results plotted and presented here to demonstrate the end result of system operation. A significant finding in this preliminary data is that the response latency increases for downward directed target deflections for some subjects.
Fig. 4 - Subject JRB, latencies (□) and settling times to 15' (+), for 1 degree fixed radius trials.
Fig. 5 - Subject JRB, latencies (□) and settling times to 15° (+), for 2 degree fixed radius trials.
Fig. 6 - Subject JRB, latencies (□) and settling times to 15' (+), for 3 degree fixed radius trials.
Fig. 7 - Subject JRB, latencies (o) and settling times to 15' (+), for 4 degree fixed radius trials.
Fig. 8 - Subject JRB, latencies (□) and settling times to 15' (+), for 1 degree random radius trials.
Fig. 9 - Subject JRB, latencies (○) and settling times to 15' (+), for 2 degree random radius trials.
Fig. 10 - Subject JRB, latencies (o) and settling times to 15' (+), for 3 degree random radius trials.
Fig. 11 - Subject JRB, latencies (□) and settling times to 15° (+), for 4 degree random radius trials.
Fig. 12 - Subject LWG, latencies (□) and settling times to 15° (+), for 1 degree fixed radius trials.
Fig. 13 - Subject LWG, latencies (o) and settling times to 15' (+), for 2 degree fixed radius trials.
Fig. 14 - Subject LWG, latencies (○) and settling times to 15° (+), for 3 degree fixed radius trials.
Fig. 15 - Subject LWG, latencies (○) and settling times to 15° (+), for 4 degree fixed radius trials.
Fig. 16 - Subject LWG, latencies (○) and settling times to 15° (+), for 1 degree random radius trials.
Fig. 17 - Subject LWG, latencies (□) and settling times to 15° (+), for 2 degree random radius trials.
Fig. 18 - Subject LWG, latencies (□) and settling times to 15° (+), for 3 degree random radius trials.
Fig. 19 - Subject LWG, latencies (○) and settling times to 15° (+), for 4 degree random radius trials.
Fig. 20 - Subject DJS, latencies (□) and settling times to 15° (+), for 1 degree fixed radius trials.
Fig. 21 - Subject DJS, latencies (□) and settling times to 15° (+), for 2 degree fixed radius trials.
Fig. 22 - Subject DJS, latencies (o) and settling times to 15° (+), for 3 degree fixed radius trials.
Fig. 23 - Subject DJS, latencies (□) and settling times to 15° (+), for 4 degree fixed radius trials.
Fig. 24 - Subject DJS, latencies (○) and settling times to 15° (+), for 1 degree random radius trials.
Fig. 25 - Subject DJS, latencies (□) and settling times to 15° (+), for 2 degree random radius trials.
Fig. 26 - Subject DJS, latencies (○) and settling times to 15° (+), for 3 degree random radius trials.
Fig. 27 - Subject DJS, latencies (○) and settling times to 15° (+), for 4 degree random radius trials.
Fig. 28 - Subject KPW, latencies (□) and settling times to 15° (+), for 1 degree fixed radius trials.
Fig. 29 - Subject KFW, latencies (□) and settling times to 15° (+), for 2 degree fixed radius trials.
Fig. 30 - Subject KFW, latencies (○) and settling times to 15' (+), for 3 degree fixed radius trials.
Fig. 31 - Subject KPW, latencies (o) and settling times to 15° (+), for 4 degree fixed radius trials.
Fig. 32 - Subject KFW, latencies (○) and settling times to 15° (+), for 1 degree random radius trials.
Fig. 33 - Subject KPW, latencies (□) and settling times to 15° (+), for 2 degree random radius trials.
Fig. 34 - Subject KFW, latencies (○) and settling times to 15° (+), for 3 degree random radius trials.
Fig. 35 - Subject KFW, latencies (□) and settling times to 15° (+), for 4 degree random radius trials.
APPENDIX A: SACCade TARGET DEFleCTOR

Introduction

The saccade target deflector (STD) is designed to generate pseudo-random deflections of the spot on an oscilloscope screen to provide visual stimuli to subjects used in the study of human eye saccades. Any oscilloscope may be used which is capable of operating in an X-Y display mode with equal x and y axis sensitivities on the order of 1 volt per centimeter.

In normal use it is assumed that the oscilloscope target spot is initially centered on the screen; the STD will generate target deflections to precisely controlled radii out from the center position and at selected angles relative to the display axes. Twenty four deflection angles from 0 to 345 degrees in 15 degree increments may be used and four radial distances are available, an adjustable maximum radius and precise 0.75, 0.50, and 0.25 fractions of the maximum radius. The experimenter may select a particular angle and radius for each target deflection trial or allow the STD to make random choices from a number of preselected allowed angles and/or radii. The STD may also be set to randomly choose from allowed angles and radii using each choice only once. This provides the experimenter with a way to present unpredictable target deflections in an efficient manner.
In addition, the STD has circuitry, inputs, and outputs to operate with a Stanford Research Institute Cornsweet-Crane Purkinje image eyetracker and a Digital Equipment Corporation PDP-8/E computer. As such, the STD will provide a trigger pulse at the onset of target deflection to initiate eyetracker data collection by the computer, and will present angle, radius, and failure (based on subject blink or eyetracker tracking loss) data to a computer input interface.
Specifications, Inputs, Outputs

Deflection radii: Four radial positions, 1.00, 0.75, 0.50, and 0.25 multiples of the maximum radius. Maximum radius adjustable from zero to four volts (x axis output for deflection angle of 0 degrees). Worst error in 0.75, 0.50, 0.25 fractions is ± 0.5% of maximum radius when maximum radius is set at 1 volt.

Deflection angles: 24 angles, 0 to 345 degrees in 15 degree increments. Worst x and y axis deflection voltage error: ± 0.3 degrees at 15, 25, 105, 165, 195, 255, 285, and 345 degrees when maximum radius is set at 1 volt.

Delay time T1 before deflection: Continuously adjustable from 0.05 to 2.5 seconds.

Target deflection time T2: Continuously adjustable from 0.05 to 5.0 seconds.

Random modes: Repeating or non-repeating for angle, repeating or non-repeating for radius. For repeating random mode the STD selects at random from non-inhibited angles or radii; for non-repeating mode each selection is inhibited when used and the STD selects the next deflection from those choices remaining.

Fail inputs: Six rear panel inputs at which TTL level signals indicating trial failures may be presented. Three inputs, FAIL L1, FAIL L2, and FAIL L3 at pins 2, 3, and 4 of J3 will cause failures to be registered for a low TTL signal level; the other three inputs, FAIL H1, FAIL H2, and FAIL H3 at pins 4, 5, and 6 of J3 register failures for a high level TTL signal. A fail signal occurring during times T1 or T2 will light the front panel FAIL indicator, set the COMP BIT 0 output high, and prevent the automatic choice inhibit from occurring in a non-repeating random mode.
X and Y axis outputs: Maximum output voltage available at either output is ±4 volts during deflection time T2. Output voltages are zero at other times as per a 10K resistor shunted to signal ground at each output. Output resistance during deflection time T2 is approximately 150 ohms.

T2 TRIG OUT: A rear panel BNC jack providing a 2-3 microsecond TTL high level pulse at the onset of deflection time T2. TTL fanout of two. Pulse may be inhibited (low) by placing the front panel T2 SWCHD switch in the off position. This pulse is compatible with and may be used to start a Digital Equipment Corporation PDP-8/E computer real time clock for subsequent data gathering by the computer A/D converter.

T1: TTL level output high during delay time T1. TTL fanout of ten. Appears at pin 10 of J3.

T1 SWCH: Same signal as T1, appears at pin 11 of J3. May be switched on and off (open circuit) with the front panel T1 SWCHD switch.

T2: TTL level output high during deflection time T2. TTL fanout of ten. Appears at pin 12 of J3.

T2 SWCH: Same signal as T2, appears at pin 13 of J3. May be switched on and off (open circuit) with the front panel T2 SWCHD switch.

LG GD: Logic Ground appears at pins 1 and 14 of J3, the shell of the T2 TRIG OUT jack, and one row of pins of the P2 data output connector. This is the ground reference for the fail inputs, the outputs T1, T1 SWCH, T2, T2 SWCH, T2 TRIG OUT, and the data outputs at connector P2.

SG GD: Signal Ground is present at the shells of the X and Y output jacks. This is the ground reference for these signals and should not be used for purposes other than the x and y axis connections to the oscilloscope.
P2 is a 40 pin Berg connector outputting deflection angle, radius, and failure data in a manner compatible for cabling directly to a Digital Equipment Corporation PDP-8/E computer M1703 omnibus input interface module. All outputs are TTL level high true with a fanout of 2 and are listed below:

<table>
<thead>
<tr>
<th>P2 pin</th>
<th>Signal Name</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>COMP BIT 0</td>
<td>Failure indicator bit</td>
</tr>
<tr>
<td>N</td>
<td>COMP BIT 5</td>
<td>Angle count E</td>
</tr>
<tr>
<td>R</td>
<td>COMP BIT 6</td>
<td>Angle count D</td>
</tr>
<tr>
<td>T</td>
<td>COMP BIT 7</td>
<td>Angle count C</td>
</tr>
<tr>
<td>V</td>
<td>COMP BIT 8</td>
<td>Angle count B</td>
</tr>
<tr>
<td>X</td>
<td>COMP BIT 9</td>
<td>Angle count A</td>
</tr>
<tr>
<td>Z</td>
<td>COMP BIT 10</td>
<td>Radius count B</td>
</tr>
<tr>
<td>BB</td>
<td>COMP BIT 11</td>
<td>Radius count A</td>
</tr>
<tr>
<td>DD</td>
<td>COMP READ REQ</td>
<td>Low pulse at T2 onset</td>
</tr>
<tr>
<td>A-UU</td>
<td>LG GD</td>
<td>Alternating pins from A to UU, logic ground.</td>
</tr>
</tbody>
</table>

COMP READ REQ is a 2-3 microsecond low pulse at the onset of deflection time T2 if the front panel T2 SWCHD switch is ON. This signal is used to set the M1703 interface flag, allowing the PDP-8/E to read the data under software control.

Radius count A and B correspond to the selected radius of deflection and are valid only during times T1 and T2. Coding is as shown:

<table>
<thead>
<tr>
<th>Deflection radius</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.25</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.50</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0.75</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1.00</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Angle count A, B, C, D, and E correspond to the selected angle of deflection and are valid only during times T1 and T2. Coding is as shown:

<table>
<thead>
<tr>
<th>Deflection angle</th>
<th>E</th>
<th>D</th>
<th>C</th>
<th>B</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>45</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>60</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>75</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Deflection angle</td>
<td>E</td>
<td>D</td>
<td>C</td>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>-----------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>90</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>105</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>120</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>135</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>150</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>165</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>180</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>195</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>210</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>225</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>240</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>255</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>270</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>285</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>300</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>315</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>330</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>345</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

*Failure indicator bit is normally low; set high by a fail input signal existing or occurring during times T1 or T2. The failure indicator bit is reset at the end of time T2.
Controls and Indicators

RAD ADJ: Front panel control, adjusts the maximum (RADIUS = 1.00) target deflection from zero to four volts. The maximum voltage is the x-axis voltage for ANGLE=0 or the y-axis voltage for ANGLE=90.

CLK: Front panel switch, controls the operation of the STD internal clock. In normal operation must be left in the ON position. This switch is placed in the off position to allow use of the CLK SING STEP switch.

CLK SING STEP: Front panel switch placed in the LO position for normal STD operation. The primary use for this switch is in debugging or testing. When the CLK switch is placed in the off position the STD internal clock may be switched HI and LO with the CLK SING STEP switch.

T1 ADJ: Front panel control. When a target deflection is initiated (GO switch) a delay T1 occurs before the target deflection. The T1 ADJ control adjusts this delay from 0.05 to 2.5 seconds. Rear panel outputs T1 and T1 SWCH are available for operating a warning buzzer or other device during this delay time T1.

T1 SWCHD: This front panel switch switches on or off (open circuit) the rear panel signal T1 SWCH.

T2 ADJ: Front panel control used to adjust the target deflection time T2. When a deflection is initiated (GO switch) a delay T1 occurs before the deflection. At the end of delay T1 the target is deflected for a time T2 before returning to its original position. T2 is adjustable from 0.05 to 5.0 seconds with T2 ADJ.

T2 SWCHD: Front panel switch, switches on and off (open circuit) the rear panel signal T2 SWCH. In addition, the T2 SWCHD switch switches on and off (TTL LO) the T2 TRIG OUT signal and the COMP READ REQ signal, both available at rear panel connectors.
ANG RAND MODE: Front panel switch, repeating (RPT) and non-repeating (NON RPT) positions. When a target deflection is initiated (GO switch) a deflection angle is selected at random by the STD from those angles which are not inhibited by the ANGLE INHIBIT switches; in RPT random mode this will continue to occur each time a deflection is initiated. In NON RPT random mode the STD will inhibit choices already used and select at random from those remaining each time a deflection is initiated. When all non-inhibited angles have been chosen no further deflection can occur until an angle reset is done (ANG RES).

RAD RAND MODE: Front panel switch, identical in function with respect to the four radii positions as the ANG RAND MODE switch function with respect to the angles.

ANGLE INHIBIT: 24 front panel switches provided to inhibit any of the available deflection angles. Inhibited angles will not be selected by the STD and target deflections will occur only at non-inhibited angles. Once the user has inhibited all angles desired, the ANGLE INHIBIT switch information must be loaded into the STD by depressing the ANG RES switch. This need be done only once if the user is operating in repeating ANG RAND MODE. When operating in non-repeating ANG RAND MODE a reset may be done at any time to restore STD inhibited angles which have been used for deflections; a reset must be done after all non-inhibited angles have been used or no further deflections can be initiated.

RADIUS INHIBIT: Four front panel switches provided to inhibit any of the four available radii. \( R = 1.00 \) is the maximum radius, corresponding to the voltage set by the RAD ADJ control. The 0.75, 0.50, and 0.25 radius positions correspond to precise fractions of the maximum radius. The RADIUS INHIBIT switches function with respect to the four radius positions as do the ANGLE INHIBIT switches with respect to the angles.

ANG RES: Front panel switch and indicator. Depressing this switch lights the indicator and loads the ANGLE INHIBIT switch settings into the STD. This must be done each time the ANGLE INHIBIT switch settings are changed. If operating in non-repeating ANG RAND MODE each non-inhibited angle will be internally inhibited when once selected for a target.
deflection; consequently an ANG RES must be done after all choices have been selected or no further target deflections will occur.

**RAD RES:**
Front panel switch and indicator. Loads RADIUS INHIBIT switch settings into the STD and in all ways performs with respect to the radius positions as does the ANG RES switch with respect to the angles.

**GO, ANG SLCTD, RAD SLCTD:**
The operation of the front panel GO switch and the two SLCTD indicators is as follows: each time the GO switch is depressed a target deflection is initiated. The STD selects at random from among the non-inhibited angles and radii; this is indicated to the user by the immediate lighting of the ANG SLCTD and RAD SLCTD indicators. Then, after the delay time T1, the target is deflected at the STD selected angle out to the selected radius and remains there for the deflection time T2. If all angle or radius choices are inhibited either by the user or by the STD operating in non-repeating mode(s), the appropriate indicator(s) will not light when the GO switch is depressed and no target deflection will occur.

**FAIL:**
This front panel indicator will light if a "failure" (fail inputs) occurs during either the T1 delay time or the T2 target deflection time. In practical use the rear panel fail inputs may be connected to the eyetracker "blink" and "track" outputs so that a trial failure consists of the subject blinking or the eyetracker losing track lock during a target deflection. The occurrence of a failure during a deflection will also set the COMP BIT 0 data output HI. This bit may then be used by computer software to reject data sampled for that trial. COMP BIT 0 and the FAIL indicator are both reset automatically at the end of deflection time T2.

One important feature of failure is the following: when operating in any non-repeating random modes the normal inhibition of the angle and/or radius will not occur. Consequently the STD will eventually again select that particular deflection for a retrial.
Operating Instructions

It is assumed that the reader has read and understood previous sections concerning controls, inputs, and outputs.

(1) Connect the STD power cord to a 120 VAC, 50-60 Hz outlet and turn on the STD and the oscilloscope to be used with the STD. Allow all equipment to warm up for at least 30 minutes before performing any adjustments of a precise nature.

(2) Set up the oscilloscope in X-Y display mode with each axis set for a sensitivity of 1 volt/cm. Note that although the sensitivity is not critical, if the sensitivities for both axes are not exactly equal, the angular deflections produced by the STD will be in error. Focus and carefully center the spot displayed on the oscilloscope screen and adjust the brightness to a comfortable viewing level.

(3) Connect the X AXIS OUT and Y AXIS OUT jacks on the STD rear panel to the oscilloscope x and y axis inputs.

(4) Maximum radius adjustment: Turn the T1 ADJ and T2 ADJ controls fully counterclockwise for minimum times. Make sure the CLK switch is ON and the CLK SING STEP switch is in the LO position. Set both RAND MODE switches to RPT. With the inhibit switches, inhibit all angles except 0 degrees and all radii except 1.00. Depress the ANG
RES and RAD RES switches, then depress the GO switch and hold it down. The oscilloscope spot should rapidly alternate between the center position and some position to the right of center (0 degrees). Adjust the RAD ADJ control for the desired maximum radius of deflection, corresponding to \( R = 1.00 \). When this has been done, release the GO switch and the spot will remain centered.

(5) T1 and T2 adjustments: Adjust the T1 ADJ control for the desired delay time between the pressing of the GO switch and the onset of spot deflection. Adjust the T2 ADJ control for the desired length of time the spot remains deflected before returning to center.

(6) Set the ANGLE INHIBIT and RADIUS INHIBIT switches to inhibit angles and radius positions not desired for target deflections. Press the ANG RES and RAD RES switches to load the inhibit switch settings into the STD.

(7) Select repeating or non-repeating ANG RAND NODE and RAD RAND MODE as desired and set the T1 SWCHD and T2 SWCHD switches on or off as required for generation of T2 TRIG OUT, COMP READ REQ, subject warning buzzer, or other user determined uses.

(8) The STD is now ready to generate target deflections. Each time the GO switch is depressed, the STD, in a pseudo-random fashion, selects an angle and radius from among the allowed choices, immediately lighting the ANG SLCTD and RAD SLCTD indicators (assuming that all angles and/or radii have not been preinhibited or inhibited by use
in a non-repeating random mode). The delay timer Ti will time out and the oscilloscope target spot will deflect to the selected angle and radius, remain deflected for time T2, then return to the center starting position. If all radii and/or all angles are inhibited by use of the inhibit switches or non-repeating random modes, the appropriate ANG SLCTD or RAD SLCTD indicator(s) will not light when the GO switch is depressed and no deflection will occur. An appropriate reset must be performed to load non-inhibited choices into the STD before further deflections may be initiated.

(9) As an example of the uses for various inputs and outputs, the following setup has been used successfully by the author:

The Ti SWCH output (and LG GD) was cabled to a five volt warning buzzer and the delay time Ti set at one second. This provided subjects viewing the target with an audible warning just prior to the target deflection. The buzzer could be inhibited at any time with the front panel Ti SWCHD switch.

The T2 TRIG OUT signal pulse was used to start a real time clock in a PDP-8/E computer which subsequently controlled a computer A/D converter. The A/D converter inputs were cabled to the eyetracker outputs; thus the subject's eye movements during the target deflection were monitored by the computer. Initiation of data collection by the computer could be inhibited at any time by switching off the front panel T2 SWCHD switch.

Target deflection time T2 was set at 3 seconds and the computer sampled the eyetracker outputs for 2.5 seconds, then read in angle,
radius, and failure data from the P2 data connector on the STD.

Failure inputs on the STD were cabled to TTL level eyetracker output signals indicating eyetracker tracking loss and subject blink. The computer was programmed to reject the 2.5 seconds of gathered data if the failure bit had been set; when no failure occurred, the computer was programmed to store the eye movement data on a mass storage device along with the angle and radius codes for the target deflection.
**Circuit Operation**

Selection of random choices for radius and angle is illustrated in the timing diagram for radius selection. Initially, when radius reset is performed, the radius position shift register, U4 on card 2, does a clocked parallel load of the four radius position settings (logic 0 is loaded for an allowed position). The radius reset also clears the radius control logic on card 1 and in particular resets the radius counter comprised of flip-flops U9B, U5A, and U5B. When radius reset is removed, the register begins clocked serial shifting of the radius switch data previously loaded and the radius counter counts the shifts in such a manner that the count corresponds to the switch data at the shift register output at any given time. Initially, RAD CNT A = RAD CNT B = 0 and the serial output corresponds to the switch data for the 0.25 radius inhibit switch. After two clock pulses, RAD CNT A = 1, RAD CNT B = 0, and the register output will be the previously loaded 0.50 radius switch data; after two more clock pulses the counter and shift register output correspond to radius switch 0.75, and so on.

The counter output drives the select inputs of analog multiplexer U2B on card 3, selecting the maximum radius voltage or the 0.75, 0.50, or 0.25 fractions of that voltage as the 1.00, 0.75, 0.50, and 0.25 radius switch data, respectively, appears at the shift register output.
When the GO switch is depressed, the first 0 shifted out of the radius shift register stops the clocking of the register and counter. The counter output is then selecting at the multiplexer whatever fractional voltage corresponds to this first non-inhibited switch data presented at the register serial output. The choice made by the STD is a pseudo-random choice as the state of the shift register cannot be known to the experimenter when the GO switch is depressed.

Selection of a non-inhibited angle by the STD takes place in a similar fashion to the radius selection process just outlined. The angle counters differ, consisting of a count to 5 counter clocking in turn a count to 3 counter. The five counter outputs drive analog multiplexers U4, U5, U8A, and U8B on card 3, selecting the proper fractions and polarities of the radius voltage to correspond to the angle data shifted to the output of register U3 on card 2.

When a non-inhibited angle and radius have both been selected, the RAD SLCT and ANG SLCT signals from the card 1 control logic combine to trigger the T1 timer, U12A on card 3. T1 eventually times out and triggers the T2 timer, U12B on card 3. When T2 goes high, the multiplexers U8A and U8B on card 3 are dis inhibited and the x and y axis voltages are presented at the x and y axis outputs of the STD. The target is thus deflected for time T2. At the end of time T2, the x and y axis outputs are again cut off, returning the outputs to 0 volts; the shift registers begin serially shifting once again.
In the case of a non-repeating random mode the serial input of the relevant shift register is held high for the first serial shift after T2, thus loading an "inhibit" for that switch position which had just been selected. In the event of a failure this will not be done and that switch position will remain enabled.
Calibration

Calibration of the various fractional voltages is done using internal STD adjustments. Calibration must be done using a digital voltmeter capable of resolving 1 millivolt with a 1 volt input.

Set up the STD for minimum T1 delay time and maximum T2 deflection time.

Radius fraction calibration:
(1) Set all angle switches for non-inhibit and inhibit all radius switches except the 1.00 switch.
(2) Set both random modes to RPT and perform angle and radius resets.
(3) Hold down the GO switch while monitoring the voltage at pin 10 of U1B on card 3, set the front panel RAD ADJ control for a reading of exactly 1,000 volts. Note that the reading will flicker every five seconds as the STD recycles for a new deflection.
(4) Release the GO switch, inhibit the 1.00 radius and disinhibit the 0.75 radius, do a radius reset, and depress and hold the GO switch again. Adjust R1 on card 3 for a reading of 0.750 volts at pin 10 of U1B.

Repeat this process for the 0.50 and 0.25 radius positions, adjusting R2 and R3 to set the voltage at pin 10 of U1B to be respectively 0.500 and 0.250 volts. This completes the radius fraction calibration.
Angle fraction calibration:

(1) Inhibit all radius positions except 1.00, inhibit all angles except 0 degrees, and perform angle and radius resets.

(2) Monitor the voltage at the X AXIS OUT jack while holding down the GO switch. Adjust the front panel RAD ADJ control, if necessary, for a reading of 1.000 volts.

(3) Inhibit all angles except 15 degrees, do an angle reset, and monitor the X AXIS OUT voltage while holding down the GO switch. Adjust R4 on card 3 for exactly 0.966 volts.

(4) Inhibit all angles except 30 degrees, do an angle reset, and monitor the X AXIS OUT voltage while holding down the GO switch. Adjust R5 on card 3 for a reading of exactly 0.866 volts.

(5) In similar fashion to the above, adjust R6 for a reading of 0.707 volts for 45 degrees, R7 for a reading of 0.500 volts for 60 degrees, R8 for a reading of 0.259 for 75 degrees, R9 for a reading of -0.966 for 165 degrees, and R10 for a reading of -0.966 for 195 degrees.

This completes the angle fraction calibration.
### Table 1 - Signal interconnection list.

<table>
<thead>
<tr>
<th>Signal name</th>
<th>From</th>
<th>To (card/pin)</th>
</tr>
</thead>
<tbody>
<tr>
<td>115 VAC</td>
<td>S38</td>
<td></td>
</tr>
<tr>
<td>115 VAC</td>
<td>S38</td>
<td></td>
</tr>
<tr>
<td>7.5</td>
<td>PWR SPLY</td>
<td>3/18  CR1</td>
</tr>
<tr>
<td>SG GD</td>
<td>PWR SPLY</td>
<td>3/19  CR1; J5; J6</td>
</tr>
<tr>
<td>-7.5</td>
<td>PWR SPLY</td>
<td>3/20</td>
</tr>
<tr>
<td>+5</td>
<td>PWR SPLY 1/21 2/21 3/21</td>
<td>S1-S30; CR2; CR3</td>
</tr>
<tr>
<td>LG GD</td>
<td>PWR SPLY 1/22 2/22 3/22</td>
<td>S33; S36; S37; L1-L5; J3/1,14; J4; P2</td>
</tr>
<tr>
<td>ANG INH 1</td>
<td>S1</td>
<td>2/A</td>
</tr>
<tr>
<td>2</td>
<td>S2</td>
<td>2/B</td>
</tr>
<tr>
<td>3</td>
<td>S3</td>
<td>2/C</td>
</tr>
<tr>
<td>4</td>
<td>S4</td>
<td>2/D</td>
</tr>
<tr>
<td>5</td>
<td>S5</td>
<td>2/E</td>
</tr>
<tr>
<td>6</td>
<td>S6</td>
<td>2/F</td>
</tr>
<tr>
<td>7</td>
<td>S7</td>
<td>2/H</td>
</tr>
<tr>
<td>8</td>
<td>S8</td>
<td>2/J</td>
</tr>
<tr>
<td>9</td>
<td>S9</td>
<td>2/K</td>
</tr>
<tr>
<td>10</td>
<td>S10</td>
<td>2/L</td>
</tr>
<tr>
<td>11</td>
<td>S11</td>
<td>2/M</td>
</tr>
<tr>
<td>12</td>
<td>S12</td>
<td>2/N</td>
</tr>
<tr>
<td>13</td>
<td>S13</td>
<td>2/P</td>
</tr>
<tr>
<td>14</td>
<td>S14</td>
<td>2/R</td>
</tr>
<tr>
<td>15</td>
<td>S15</td>
<td>2/S</td>
</tr>
<tr>
<td>16</td>
<td>S16</td>
<td>2/T</td>
</tr>
<tr>
<td>17</td>
<td>S17</td>
<td>2/U</td>
</tr>
<tr>
<td>18</td>
<td>S18</td>
<td>2/V</td>
</tr>
<tr>
<td>19</td>
<td>S19</td>
<td>2/W</td>
</tr>
<tr>
<td>20</td>
<td>S20</td>
<td>2/X</td>
</tr>
<tr>
<td>21</td>
<td>S21</td>
<td>2/Y</td>
</tr>
<tr>
<td>22</td>
<td>S22</td>
<td>2/Z</td>
</tr>
<tr>
<td>23</td>
<td>S23</td>
<td>2/1</td>
</tr>
<tr>
<td>24</td>
<td>S24</td>
<td>2/2</td>
</tr>
<tr>
<td>RAD INH 1</td>
<td>S25</td>
<td>2/10</td>
</tr>
<tr>
<td>2</td>
<td>S26</td>
<td>2/11</td>
</tr>
<tr>
<td>3</td>
<td>S27</td>
<td>2/12</td>
</tr>
<tr>
<td>4</td>
<td>S28</td>
<td>2/13</td>
</tr>
<tr>
<td>Signal name</td>
<td>From</td>
<td>To (card/pin)</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------</td>
<td>--------------</td>
</tr>
<tr>
<td>RAD RES 1</td>
<td>S29</td>
<td>1/16</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1/15</td>
</tr>
<tr>
<td>ANG RES 1</td>
<td>S30</td>
<td>1/T</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1/S</td>
</tr>
<tr>
<td>RAD RES LED</td>
<td>1/Y</td>
<td>L1</td>
</tr>
<tr>
<td>ANG RES LED</td>
<td>1/X</td>
<td>L2</td>
</tr>
<tr>
<td>RAD SLCT LED</td>
<td>1/18</td>
<td>L4</td>
</tr>
<tr>
<td>ANG SLCT LED</td>
<td>1/19</td>
<td>L5</td>
</tr>
<tr>
<td>FAIL LED</td>
<td>1/W</td>
<td>L3</td>
</tr>
<tr>
<td>RAD ADJ</td>
<td>CR1</td>
<td>3/1</td>
</tr>
<tr>
<td>T1 ADJ</td>
<td>CR2</td>
<td>3/3</td>
</tr>
<tr>
<td>T2 ADJ</td>
<td>CR3</td>
<td>3/2</td>
</tr>
<tr>
<td>T1</td>
<td>3/7</td>
<td>S31; J3/10</td>
</tr>
<tr>
<td>T1 SWCH</td>
<td>S31</td>
<td>J3/11</td>
</tr>
<tr>
<td>T2</td>
<td>3/8</td>
<td>S32; J3/12</td>
</tr>
<tr>
<td>T2 SWCH</td>
<td>S32</td>
<td>3/10 J3/13</td>
</tr>
<tr>
<td>GO</td>
<td>S33</td>
<td>1/U</td>
</tr>
<tr>
<td>RANDOM CON</td>
<td>1/20</td>
<td>S34; S35</td>
</tr>
<tr>
<td>RAD SI CON</td>
<td>S34</td>
<td>2/16</td>
</tr>
<tr>
<td>ANG SI CON</td>
<td>S35</td>
<td>2/5</td>
</tr>
<tr>
<td>CLK</td>
<td>3/14</td>
<td>1/14</td>
</tr>
<tr>
<td>CLK</td>
<td>3/13</td>
<td>1/13</td>
</tr>
<tr>
<td>CLK HI</td>
<td>S37</td>
<td>3/4</td>
</tr>
<tr>
<td>CLK LO</td>
<td>S37</td>
<td>3/5</td>
</tr>
<tr>
<td>CLK STOP</td>
<td>S36</td>
<td>3/6</td>
</tr>
<tr>
<td>X AXIS OUT</td>
<td>3/X</td>
<td>J5</td>
</tr>
<tr>
<td>Y AXIS OUT</td>
<td>3/Y</td>
<td>J6</td>
</tr>
<tr>
<td>COMP CLK TRIG</td>
<td>3/12</td>
<td>J4</td>
</tr>
<tr>
<td>COMP BIT 11</td>
<td>1/1</td>
<td>P2/BB</td>
</tr>
<tr>
<td></td>
<td>1/2</td>
<td>P2/Z</td>
</tr>
<tr>
<td></td>
<td>1/3</td>
<td>P2/X</td>
</tr>
<tr>
<td></td>
<td>1/4</td>
<td>P2/V</td>
</tr>
<tr>
<td></td>
<td>1/5</td>
<td>P2/T</td>
</tr>
<tr>
<td></td>
<td>1/6</td>
<td>P2/R</td>
</tr>
<tr>
<td></td>
<td>1/7</td>
<td>P2/N</td>
</tr>
<tr>
<td></td>
<td>1/12</td>
<td>P2/B</td>
</tr>
<tr>
<td>COMP READ REQ</td>
<td>3/11</td>
<td>P2/DD</td>
</tr>
<tr>
<td>Signal name</td>
<td>From</td>
<td>To (card/pin)</td>
</tr>
<tr>
<td>--------------</td>
<td>-------</td>
<td>--------------</td>
</tr>
<tr>
<td>FAIL L1</td>
<td>J3/2</td>
<td>3/J</td>
</tr>
<tr>
<td>FAIL L2</td>
<td>J3/3</td>
<td>3/K</td>
</tr>
<tr>
<td>FAIL L3</td>
<td>J3/4</td>
<td>3/L</td>
</tr>
<tr>
<td>FAIL H1</td>
<td>J3/5</td>
<td>3/M</td>
</tr>
<tr>
<td>FAIL H2</td>
<td>J3/6</td>
<td>3/N</td>
</tr>
<tr>
<td>FAIL H3</td>
<td>J3/7</td>
<td>3/P</td>
</tr>
<tr>
<td>RAD SER OUT</td>
<td>2/17</td>
<td>1/K</td>
</tr>
<tr>
<td>ANG SER OUT</td>
<td>2/6</td>
<td>1/J</td>
</tr>
<tr>
<td>FAIL</td>
<td>3/R</td>
<td>1/R</td>
</tr>
<tr>
<td>RAD CNT A</td>
<td>1/A</td>
<td>3/A</td>
</tr>
<tr>
<td>RAD CNT B</td>
<td>1/B</td>
<td>3/B</td>
</tr>
<tr>
<td>RAD RESET</td>
<td>1/L</td>
<td>2/15</td>
</tr>
<tr>
<td>RAD REG CLK</td>
<td>1/N</td>
<td>2/14</td>
</tr>
<tr>
<td>RAD SLCT</td>
<td>1/Y</td>
<td>3/S</td>
</tr>
<tr>
<td>ANG CNT A</td>
<td>1/C</td>
<td>3/C</td>
</tr>
<tr>
<td>ANG CNT B</td>
<td>1/D</td>
<td>3/D</td>
</tr>
<tr>
<td>ANG CNT C</td>
<td>1/E</td>
<td>3/E</td>
</tr>
<tr>
<td>ANG CNT D</td>
<td>1/F</td>
<td>3/F</td>
</tr>
<tr>
<td>ANG CNT E</td>
<td>1/H</td>
<td>3/H</td>
</tr>
<tr>
<td>ANG RESET</td>
<td>1/M</td>
<td>2/4</td>
</tr>
<tr>
<td>ANG REG CLK</td>
<td>1/P</td>
<td>2/3</td>
</tr>
<tr>
<td>ANG SLCT</td>
<td>1/Z</td>
<td>3/T</td>
</tr>
</tbody>
</table>
Selection Timing Diagram For Radius, Fig. 36

A: Shift register parallel loads during reset.

B: Rest removed, shift register cycles serially.

C: "GO" is initiated.

D: Serial shift continues until first allowed position (RAD SER OUT low).

E: First allowed position.

F: Next clock low, RAD SLCT goes high, RAD REG CLK inhibited, U4B will now detect fail inputs. If ANG SLCT is already high, the T1 timer is triggered.

G: During this time the T1 timer times out and triggers the T2 timer high.

H: FAIL goes low due to a fail input signal.

I: FAIL SYNC goes high as FAIL (L) is clocked into U4B by the first clock low.

J: T2 times out, clocks U3AQ high.

K: With U3AQ high, next clock low sets U3BQ high, resulting in reset of U3AQ and RAD SLCT. Since FAIL SYNC was set high, RANDOM CON stays low and this position will not be inhibited. Had no failure occurred to set FAIL SYNC, RANDOM CON would have been high from K to M and this position would have been inhibited.

L: First clock high to shift register and counters. RAD SER OUT (low, no inhibit) clocked to register serial input.

M: First clock low resets U3BQ, FAIL SYNC.

N: If "GO" and RAD WER OUT stay low this point corresponds to point F and the cycle repeats.
Fig. 36 - Selection timing diagram for radius.
Fig. 37 - Angle/radius control, card 1
Fig. 39 - Angle/radius shift registers, card 2
Fig. 41 - XY voltage decoder, fail circuit, clock, timers, card 3
Fig. 42 - XY voltage decoder, fail circuit, clock, timers, card 3, layout
Fig. 43 - Front and rear panel wiring
Fig. 44 - Power supply
APPENDIX B: CORMSWEET - CRANE
PURKINJE IMAGE EYETRACKER

Theory

One of the better methods yet developed to measure human eye rotation is the double-Purkinje image technique, implemented at Stanford Research Institute by T. N. Cornsweet and H. D. Crane. An early marketable version of a "Cornsweet-Crane" eyetracker was purchased for use at the Ohio State University Institute for Research in Vision and has been modified and used by this author as the core of the automated saccade analysis system.

The eyetracker measures eye rotation nearly uncontaminated by eye translation artifacts, and outputs electrical signals directly proportional to eye rotation along horizontal and vertical axes. The primary features of the machine are high accuracy coupled with the non-contacting measurement method; most other methods of measuring eye rotation must sacrifice one feature for the other.

The theory of operation is as such: a narrow collimated beam of infrared light is directed at the cornea of the eye, producing a number of real and virtual images. The reflection of the beam from the front surface of the cornea, a convex "mirror", produces a virtual point image (approximately), known as the first Purkinje image. Part of the collimated beam will also pass through the cornea, pupil, and lens, producing second, third, and fourth Purkinje
images from reflections at the rear surface of the cornea, and front and rear lens surfaces, respectively. The fourth Purkinje image, P4, resulting from beam reflection from the concave "mirror" formed by the rear surface of the lens, is a real image and it is this image, along with the virtual first image, P1, that is utilized in the double Purkinje image technique.

The positions of these images are illustrated schematically in Fig. 45. The rear surface of the lens, M4, and front surface of the cornea, M1, have been shown as sections of spherical mirrors. Point CR represents the center of rotation of the eye and points CM1 and CM4 represent the centers of curvature of the two mirrors. Fig. 45a depicts the positions of the two images P1 and P4 for the case where the eye is in its reference position with respect to both translation and rotation. The case of simple rotation is illustrated in Fig. 45b. As observed along the indicated viewing axis, P1 and P4 have moved with respect to each other. It is this relative movement of the images that is used as a measure of eye rotation; the movement is very nearly linear with respect to eye rotation for small angles. Translation of the eye, however, produces no relative movement of P1 and P4. As illustrated in Fig. 45c the images will both move when the eye translates, but by the same amount. Consequently, if one can measure only the relative movement of P1 and P4, rotation of the eye can be measured without translational errors.

A number of second order optical errors have been ignored in the foregoing discussion and the constancy of physiological parameters
Fig. 45 - Schematic representation of the positions of the first and fourth Purkinje images during rotation and translation of the eye.
such as lens curvature has been tacitly assumed. The performance specifications of the eyetracker itself, discussed later, are such that these effects are not significant. The reader may find more detailed treatments of the theory in relevant sources included in the reference listing.

Eyetracker Operation

The Cornsweet-Crane eyetracker provides a direct measure of eye rotation in the form of electrical signals directly proportional to the separation of the 1st and 4th Purkinje images. X and Y axis outputs are both provided to give independent measurements of rotation in the vertical and horizontal directions. When properly aligned for a particular subject, both outputs are at zero volts when the subject fixates the center of the visual field. Horizontal rotation to the right then produces a proportional positive x axis voltage while rotation to the left results in a corresponding negative voltage. Similarly, upward eye rotation produces a positive y axis voltage while downward movement yields a negative y axis output. These x and y axis outputs are respectively denoted H4 and V4. The eyetracker also provides two output signals, denoted H1 and V1, which are proportional to the horizontal and vertical position of the first Purkinje image. All four outputs have independent offset and gain adjustments.

H1 and V1, the signals proportional to first image position, are generated in the following way: light from the first Purkinje image (P1) is focused by eyetracker optics onto a four-quadrant
photodetector. Plus and minus x and y axis signals are derived from the position of the image on the photodetector. These signals are amplified and processed to control a servo-driven mirror in the optical path to the photodetector. The two servos used to drive the mirror are of the linear displacement type and each rotates the mirror a small amount in the horizontal or vertical plane. The net effect is such that the image of P1 remains centered at the photodetector while the real P1 moves about. Electrical signals corresponding to x and y axis P1 movements are generated by two differential transformer position sensors mounted to monitor the mirror movement. Thus, as the subject's eye moves, the resulting attempted movement of the conjugate P1 at the photodetector is detected and compensated by the servo-driven mirror in the optical path. The mirror position is proportional to P1 position and this mirror position in electrically monitored to provide outputs corresponding to x and y axis P1 movement.

H4 and V4, the signals proportional to eye rotation, are generated in a somewhat similar way by tracking the fourth Purkinje image with a second photodetector and servo-driven mirror. However, the optical arrangement is such that the light from the fourth image is split out of the first image optical path after the first image servo-mirror. Thus the image of the eye entering the fourth image optical path is made to be stationary with respect to the first Purkinje image, and the first image movement is "optically subtracted" from that of the fourth. The fourth Purkinje image is focused and centered on its four-quadrant photodetector and remains centered as the eye rotates.
due to the servo-mirror action. Two more differential transformer position sensors provide electrical signals corresponding to the fourth image mirror position and these are processed and output as the rotation signals $H^4$ and $V^4$.

**Modifications**

A number of modifications were made by the author to the Cornsweet-Crane eyetracker in order to improve performance. Changes are listed below.

(1) The power output stages of all four mirror servo driver circuits were replaced due to inadequate drive and heat sinking capacity.

(2) During an eyeblink by the subject, servo circuitry must be inhibited until the blink passes; "hold" circuitry as supplied, maintained the servo error, not the servo position, during the blink. The integrating servo circuitry would then invariably wander off during a blink. The circuitry was modified to hold servo position during a blink.

(3) Calibration circuits were added to enable step drive calibration of all four servo drive circuits to insure proper response with no overshoot or ringing. Calibration is done by setting up the eyetracker to "track" a small glass sphere, then supplying 10 Hz signals to the added circuitry; the circuitry switches signal gains at the appropriate quadrants of each four-quadrant photodetector.
to electronically mimic a 10 Hz square wave motion of the glass sphere. Servo calibration points may then be adjusted for optimum response to square wave displacements.

(4) Electronic filters were inserted to reduce noise in the fourth image output signals. Two-pole lowpass filters with a cutoff frequency of 250 Hz were added to both horizontal and vertical channels resulting in substantial reduction of random electrical noise and differential transformer carrier signal artifact.

(5) To eliminate slight beat frequency effects arising from interaction of independent carrier oscillators in the differential transformer position sensors, magnetic shielding was added to all four sensors.

(6) To enable eyetracker operation with subjects wearing eyeglasses, the gains of the photodetector signal amplifiers were increased. Eyeglasses attenuate the infrared beam by roughly a factor of 2. As such, control panel switches and internal circuitry were added to allow the user to set $X_1$ or $X_2$ normal gains for either or both of the first and fourth image photodetector amplifiers.
**Eyetracker Measured Specifications**

Specifications of prime interest to the eyetracker user are electrical noise expressed as equivalent eye rotation, eye tracking slew rate in degrees per second, tracking delay, and linearity over some specified field of view. These parameters were all measured by methods outline below, and are tabulated at the end of this section.

All instruments, in particular the eyetracker itself, had been recently calibrated. Measurements were made while tracking the right eye of subject JRB, or under conditions as closely approximating this as was possible. At the outset, \( H_4 \) and \( V_4 \) output gains were set at 0.1 volt/degree while tracking subject JR3; these gains were used throughout all tests of eyetracker performance. Linearity was measured while tracking subject JRB while noise, slew rate, and delay were measured while "tracking" a 1 cm glass sphere with all signals adjusted to normal levels as would be seen with a real eyeball.

Noise:

Electrical noise was measured with all servos operating, with the \( P_1 \) detector sensitivity set at \( X2 \) normal, and with the \( P_4 \) detector sensitivity set at \( X1 \) normal. Most subjects tested wore eyeglasses and these settings are most frequently used for such subjects. The computer was used to calculate noise by sampling the \( H_4 \) and \( V_4 \)
outputs every 5 milliseconds for 1 second, then calculating the standard deviation of the resulting 200 samples taken from each output. A/D converter resolution is 2 millivolts, corresponding to approximately 1.2 minutes of arc of eye rotation (at 0.1 volt/deg). H4 and V4 signal bandwidth is zero to 250 Hz.

Slew rate and delay:

Slew rate and delay were measured with all servos operating, with the P1 detector sensitivity set at X2 normal, and with the P4 detector sensitivity set at X1 normal. Most subjects tested wore eyeglasses and these settings are most frequently used for such subjects. The servo calibrator circuits were used to electronically mimic a 10 Hz square wave motion of the glass sphere "tracked" in this test.

Linearity:

Linearity was measured while tracking the right eye of subject JRB. The saccade target deflector was used to generate target deflections at angles of 0, 45, 90, 135, 180, 225, 270, and 315 degrees, and at 1, 2, 3, and 4 degree radii for each angle used. Two trials were made at each of the 32 positions. Eye movement data was taken for each trial using ADIS01 software sampling at a 5 msec rate for 2.5 seconds and all 64 trials were processed using a patched version of SPROUT. The modified SPROUT provided the calculated average x and y axis position and standard deviations for the first 0.1 and
final 1.0 second of each 2.5 second response record.

The purpose of using this seemingly elaborate method is to eliminate small offsets resulting from slight head movements between trials. For each trial then, the average initial position was subtracted from the average final position to yield a normalized final position for that trial. The assumption has been made that negligible shifts will occur during the duration of a trial.

Results:

Equivalent translational slew rate, $H_1$ and $V_1$ outputs: 2 inches/second

RMS equivalent rotational noise, $H_4$ and $V_4$ outputs: 0.4 minutes of arc

Equivalent rotational slew rate, $H_4$ and $V_4$ outputs: 60 degrees/second

Delay, $H_4$ and $V_4$ outputs: 2 milliseconds

Rotational linearity within a 4 degree radius circle, $H_4$ and $V_4$ outputs: any error less than noise
APPENDIX C: SYSTEM SOFTWARE

Hardware

The software included in this section was developed on and intended for use with a computer system consisting of the following:
(1) Digital Equipment Corporation (DEC) PDP-8/E computer equipped with the following DEC internal options:
   - KM8-E memory extension and time share
   - 8K total core memory
   - DK8-EP programmable real time clock
   - M865 teletype control
   - AD8-EA 10 bit analog to digital converter
   - AM8-EA eight channel analog preamplifier and multiplexer
   - XY8-E plotter control
   - M1703 omnibus input interface (two required)
(2) DEC OS/8, version 3C, software operating system.
(3) Teletype corporation model ASR-33 teletype.
(4) Sykes Datatronics, Inc. model 7250 dual buffered flexible disk system with PDP-8/E hardware interface and OS/8 software driver package.
(5) Electronic Associates Inc. model 130 dataplotter (digital x-y plotter).
Automated saccade analysis system interconnections, including the Cornsweet-Grane eyetracker and the saccade target deflector, are shown in Fig. 46 and outlined below:

1. M865 teletype control is the hardware interface between the computer and the ASR-33 teletype.
2. XY8-E plotter control is the hardware interface between the computer and the model 130 dataplotter.
3. One M1703 input interface (internal device address = octal 14) is cabled to the plotter and is used to transmit offscale signals to the computer to enable software management of the plotter when driven offscale.
4. One M1703 input interface (internal device address = octal 15) is cabled to the saccade target deflector and inputs target deflection angle and radius codes and one "failure" bit used by software to reject data. This bit is set if the subject blinks or the eyetracker suffers track loss during a target deflection trial.
5. Eyetracker H4 and V4 (eyeball x and y rotation) outputs are cabled to input channels 0 and 1 of the AM8-EA computer multiplexer.
6. Saccade target deflector T2 TRIG OUT signal is cabled to one of the DK8-EP clock trigger inputs (EVENT 1) and is used to trigger the clock which subsequently controls the A/D sampling of eyetracker eye rotation signals.
Fig. 46 - Automated saccade analysis system, hardware and major hardware interconnections.
of all hardware operation may be found in manufacturer's publications listed in the reference section.

The DEC OS/8 software system and the Sykes dual disk drive allow creation, manipulation, and storage of a variety of data and program files. OS/8 supports standard FORTRAN II programming which may be intermixed with DEC SABR assembly language instructions to give the user a highly versatile overall programming "language". The OS/8 system also includes a variety of text editing, file manipulation, debugging, and general utility programs best referenced by the user in the DEC publications mentioned in the reference listing.

In normal use with the automated saccade analysis system, OS/8 and author developed programs of frequent usage reside on disk drive FL1. This keeps disk drive FL2 free for data file storage and manipulation.
Software Overview

Programming for the automated saccade analysis system may be broken down into three functional areas: input, output, and processing. Program ADIS01 inputs analog data from the eyetracker, programs APLOTO and SSPLLOT respectively output raw and processed data to the plotter, and program SPROUT processes saccade data. All programs are structured to have teletype command decoder routines, allowing user control of, and interaction with, the programming via the teletype keyboard.

ADIS01 allows the user to control the sampling of analog output voltages from the eyetracker and to store the digitized data in data files on disk drive FL2. It is a general purpose analog-to-digital sampling and storage program with alterable sampling parameters and was written for general usage.

APLOTO is a general purpose x-y plotting program designed to operate with ADISO1-created data files as input. APLOTO allows the user to plot previously sampled analog data in a wide variety of ways; data previously sampled from any analog input channel may be plotted as a function of time or as a function of any other sampled channel. Scale factors are user selectable and a "slow-motion" plotting feature has been incorporated.

SPROUT programming processes saccade data files as created by ADISO1; it expects input data files from disk drive FL2 to each
consist of 2 channel (0 and 1) sampled data representing a
subject's eye movement response to a step target deflection lasting
about 3 seconds. SPROUT calculates first saccade latency and settling
times to four radii about the final target position for each of these
saccade data files. SPROUT will also input and process up to 63
data files with no user interaction and include the calculated results
in a single "saccade summary file", thus providing efficient storage
and access to data about many responses.

SSPLOT is a special purpose plotting program which inputs
saccade summary files previously generated and stored by SPROUT.
SSPLOT plots summary file data (latencies and settling times) in
a variety of convenient ways.

The user may thus input and store eye rotation data using
ADISO1, plot individual response data files using APLOTO, process
saccade data files and summarize the results with SPROUT, and generate
plots of summarized data for many saccades with SSPLOT. These four
programs are each discussed in detail in the following sections and
a program listing of each program is included with each discussion.
Fig. 47 illustrates data flows through the system software and indicates
FL2 file storage points.
Fig. 47 - Automated saccade analysis system, system software data flows.
ADIS01: Analog data input and storage program

Design considerations:

Reduced to simplicity, the purpose of ADIS01 is to store analog eye rotation data from the eyetracker in digitally coded form on flexible disk storage medium. Primary design considerations were (1) that ADIS01 be flexible and allow easy change of sampling parameters, (2) that ADIS01 be usable by non-technical personnel, and (3) that ADIS01 require little or no user interaction during experiments with subjects; data was to be collected and stored as automatically as possible to reduce experiment time with subjects and to reduce the work load on the experimenter.

To meet the above requirements ADIS01 has the following features: (1) all sampling parameters - number of channels, sample time, and number of samples - all have initial default values convenient to the work done here; all parameters are also alterable by use of teletype keyboard commands. A command is also provided to list all current parameters. (2) All parameter change commands cause a teletype request for specific information. (3) Error checks are performed to reject meaningless commands or entered specifications; pertinent error messages are printed and sampling cannot be started until user entered specifications are acceptable. (4) Once sampling has been triggered and started it may be stopped at any time with
an immediate return to command mode. (5) A sampling recycle mode is provided to allow the user to trigger one sampling set after another with disk data file storage and change of filename automatically occurring after each sample set. (6) At the end of a sampling set a single externally supplied 12 bit input word is read and inserted as a codeword in the data file. Thus information about the sample set (target deflection angle and radius for work here) may be included with the data. One bit of this input word may be used as a failure check to cause rejection of the entire sample set with no disk storage.

ADISOI thus allows the user to design an experiment such that many trials may be done quickly, data from each trial carrying codewords identifying the trial, with automatic rejection of "bad" trial data, automatic disk storage of "good" data, and automatic change of filename between sample sets to allow unattended sampling of many trials.
Specifications and features:

Analog multiplexer input voltage, any channel: ± 1.0 volts

A/D converter step size: 1.95 mv (1/512 volts)

Multiplexer input channels: 8 (0 to 7)

Sampling intervals available: 1, 2, 5, 10, 20, 50, 100, 200, 500 msec., 1, 2, 5, 10, 20, 50 seconds.

Maximum total samples (samples/channel X channels) per sample set (one disk data file): 1020

Sampling is started by any or all of 3 TTL level Schmitt trigger inputs (EVENT 1, 2, 3).

Starting modes:

Sampling sequence:

When a Schmitt trigger is fired, the specified starting channel is immediately sampled. The multiplexer then increments to the next highest channel number, samples that channel and so on until the specified number of channels has been sampled. After this, the program waits for the specified sample time, then repeats the sequence. Samples are stored in sequential order in an array.

User stop:

Sampling may be stopped at any time by pressing the teletype "S" key.

Output data filenames:

Must be six characters, first four alphanumeric, last two numbers, i.e. JRBD00 or HJ4$23. Data files are output to FL2.
Sampling modes: One shot: One set of samples is taken, the data is output to FL2 as a data file with the specified filename, the program returns to command mode.

Recycle mode: One set of samples is taken, the data is output to FL2 as a data file with the specified filename, the last two digits of the filename are incremented, and the program waits for another sample set to be triggered. Thus up to 100 (00 to 99) data sets may be sampled and stored with no user interaction other than a starting trigger pulse for each set.

Automatic data rejection: Regardless of sample mode, after the last sample in a set is taken, a failure input bit is checked. If set, the entire sample set is rejected, no disk storage occurs, and no filename increment is done in recycle mode.
ADIS01 Commands:

Commands to ADIS01 may be entered on the teletype whenever ADIS01 is in command mode, indicated when a colon is printed at the teletype left margin. Commands are AC, AI, AS, AT, AM, AE, AF, AP, AO, and AR. Command functions are outlined below.

AC: AC calls subroutine ACPRG which requests user entry of the number of channels to be sampled. Valid entry is an integer from 1 to 8, followed by a return to command mode.

AI: AI calls subroutine A1PRG which requests user entry of starting channel for sampling. Valid entry is an integer from 0 to 7. ADIS01 returns to command mode.

AS: AS calls subroutine ASFRG which requests user entry of the number of samples per channel. Valid entry is any integer from 1 to 1020. Program returns to command mode.

AT: AT calls subroutine A1PRG which requests user entry of a sampling timebase: 1, 2, or 5 milliseconds. Valid entry is 1, 2, or 5. Return to command mode follows.

AM: AM calls subroutine AMPRG which requests user entry of a timebase multiplier. Valid entry is an integer from 0 to 4. The sampling interval will be (timebase) X (10 to the timebase multiplier power). Return to command mode follows.
AE: AE calls subroutine AEPRG which requests user entry of "E" or "D" to enable or disable each of the three trigger inputs used to start sampling. Return to command mode follows.

AF: AF calls subroutine AFPRG which requests user entry of the output filename. Valid entry is AAAANN where each A is any alphanumeric and each N is an integer. Return to command mode follows.

AP: AP calls subroutine APFRG which prints a list of all sampling parameters as currently specified, then returns to command mode.

AO: AO calls subroutine AOFRG which manages sampling in one-shot mode. Sampling will start when an enabled Schmitt trigger is fired and continue until the user specified number of samples is taken and stored in a data array. The array is then normally output to FL2 under the current filename and the program returns to command mode.

AR: AR calls subroutine ARPRG which manages sampling in recycle mode. Exit from recycle mode back to command mode may be done by pressing the "S" key on the teletype during sampling, for an immediate exit, or between sample sets for an exit at the next starting trigger pulse.
ADIS01 data array format:

For purposes of FORTRAN II programming each analog voltage sampled by the multiplexer/ converter is coded as a decimal integer in the range ± 512, corresponding to input voltage ± 1 volt. The actual analog voltage may be later recovered for use in calculations by simply dividing the digitized form by 512.

Sampled data is stored in sequentially ascending locations in a FORTRAN integer array of size 1024. The first four words of the array are reserved for coded information about the sampling parameters used to take the data and one externally user supplied information word about the data (here, target deflection angle and radius). The maximum total number of samples (samples per channel times number of channels) that can be taken, stored, and output to disk FL2 in one sample set is thus 1020, the remainder of the array.

Codewords are formatted as such:

IDATA(1): Bits 0-5 contain the binary number of sequentially sampled channels, ICHNS. Bits 6-11 contain the binary number of the first channel used, ICHN1.

IDATA(2): Word 2 contains minus the number of samples per channel, ISPCT.

IDATA(3): Bits 0-5 contain the binary timebase multiplier, ITBM. Bits 6-11 contain the binary timebase, IMSEC. Sample period for the data was (IMSEC)(10**ITBM) milliseconds.

IDATA(4): Bits 5-9 contain the target deflection angle code, bits 10-11 contain the target deflection radius code. This codeword is externally supplied by the user and for the work here is supplied by the saccade target deflector.
Data storage in the array starts at word five (IDATA(5)) and is stored sequentially, one group of samples followed by the next in ascending order. For instance, if two channels are sampled, starting with channel 0, and 500 samples per channel are to be taken, the samples will be stored as such: IDATA(5) and IDATA(6) will contain, respectively, the channel 0 and 1 samples taken at the start of sampling, IDATA(7) and IDATA(8) will contain the channel 0 and channel 1 samples taken one sample period later, IDATA(9) and IDATA(10) will contain the next channel 0/1 sample pair, and so forth; the last sample pair will be stored at IDATA(1003) and IDATA(1004).

The data array, formatted as outlined above, is output to disk FL2 using FORTRAN 12A2 format; the disk file thus created may be read back into core using the same format to recover the data array.
ADIS01 program operation:

ADIS01 is comprised of a main FORTRAN II program ADIS1 which acts as a teletype command interpreter, and FORTRAN II/SAHR subroutines ACPRG, A1PRG, ASPRG, ATFRG, AMPRG, AEPRG, AFPRG, AOPRG, ARPtG, FLPFL, APCHK, FLINC, and SMPL1. All but the last four of these subroutines are called directly by ADIS1 in response to keyboard commands; FLPFL, APCHK, FLINC, and SMPL1 are called by other subroutines.

Major variables are "passed" among subroutines via FORTRAN II common storage locations. These are:

ICHNS - number of analog input channels (sampled sequentially)
ICHN1 - starting channel for sampling
IMSEC - sample period timebase, 1, 2, or 5 (msec)
ITBM - sample period timebase multiplier
ISAMP - number of samples per channel
IEN1, IEN2, IEN3 - ASCII codes for "E" or "D" for clock event enabling.
FNME - output filename, stored in ASCII A6 format.
ICHCT - equals -ICHNS
ISPCT - equals -ISAMP
ICKEN - clock enable word, instructs programmable clock
ITMBS - clock count word
ADIS1 and other subroutine operations are outlined below, followed by a program listing of ADIS01 in its entirety.

ADIS1:

ADIS1 interprets user entered keyboard commands. Appropriate subroutines are called to execute valid commands and an error message is printed if the command entered is invalid. Sampling parameters are initialized to values convenient for recording saccade eye movement responses to step target deflections and are as follows: number of channels to be sampled = 2, starting channel = 0, sample period = 5 msec, samples per channel = 500, and EVENT 1 (Schmitt trigger input) is enabled. With these initializations, then, ADIS01 will, when commanded (AO or AR), sample both channels 0 and 1 every 5 msec, beginning when triggered by EVENT 1 and continuing for 2.495 seconds, taking altogether 1000 samples.

ACPRG:

ACPRG is called by ADIS1 response to command AC and allows user entry of the number of analog channels to be sampled. ACPRG requests this information, then checks the user's entry to see that it is valid. If so, the entry is accepted, ICHCT is set equal to -ICHNS, and ACPRG returns to ADIS1; ACPRG repeats its request until the user entered number is acceptable.
AlFRG:

AlFRG is called by ADIS1 in response to command Al. AlFRG requests user entry of the starting channel for data sampling, checks that the entered number is between 0 and 7, and repeats the entry request until this is so. AlFRG returns to ADIS1 upon receiving a satisfactory entry.

ASFRG:

ASFRG is called by ADIS1 in response to command AS and requests user entry of the number of samples to be taken (per channel). A check is made to insure that the entry is a number between 1 and 1020. If so, ISAMP is set equal to -ISAMP and ASFRG returns to ADIS1; if not, the entry request is repeated.

ATFRG:

ATFRG is called by ADIS1 in response to command AT and requests user entry of the sample time timebase - 1, 2, or 5 (msec). ATFRG checks to insure that the entered number is 1, 2, or 5, and repeats the request until this is so. Entry of 1, 2, or 5 causes assignment of a value to the clock count word (ITMBS) and a return to ADIS1.

AMFRG:

AMFRG is called by ADIS1 in response to command AM and requests user entry of the timebase multiplier. A check is made to insure an entered number between 0 and 4 and the request is repeated until this
is so. Valid entry causes appropriate changes to be made in the clock enable word ICKEN and AMPRG returns to ADIS1.

AEPRG:

AEPRG is called by ADIS1 in response to command AE and requests user entry of "E" or "D" to enable or disable each of three separate Schmitt trigger inputs, denoted as EVENT 1, 2, and 3, which may be used to start the real time clock to initiate and control A/D sampling. Entries other than E or D cause a repeat of the entry request. Acceptable entries cause AEPRG to make appropriate changes in the clock enable word (ICKEN) to enable and disable the trigger inputs. AEPRG then returns to ADIS1.

AFPRG:

AFPRG is called by ADIS1 in response to command AF and requests user entry of the filename (FNMS) under which the yet-to-be sampled data array will be stored on disk drive FL2. A check is made to insure that the last two characters of the entered six-character filename are numbers, and the entry request is repeated until this is so. Acceptable entry causes a message to that effect and a return to ADIS1.

APPRG:

APPRG is called by ADIS1 in response to command AP and prints a list of all current sampling parameters, then returns to ADIS1.
AOFRG:

AOFRG is called by ADIS1 in response to command AO and permits one-shot sampling to occur, i.e. one sample set will be taken and output to FL2 and a return to ADIS1 will follow.

AOFRG first calls subroutine APCHK which checks all current sampling parameters to see if various criteria are met, then returns to AOFRG. If APCHK checks have been satisfied, AOFRG then calls the sampling subroutine SMPLR (version SMPL1) which waits for a trigger signal, does the analog sampling, and array storage, and returns to AOFRG. If the user interrupts sampling with the user stop feature SMPLR returns to AOFRG which immediately returns to ADIS1. If sampling is completed as usual by SMPLR, a return to AOFRG is followed by a call to subroutine FLPFL to output the array of sampled data. FLPFL then returns to AOFRG which returns to ADIS1.

ARFRG:

ARFRG is called by ADIS1 in response to command AR and manages sampling in recycle mode. ARFRG first calls subroutine APCHK to check all current sampling parameters; APCHK returns to ARFRG. Valid parameters result in ARFRG calling SMPLR (version SMPL1) which waits for a starting trigger signal, then samples the analog data channels as per the current sampling parameters. If the user interrupts sampling with the user stop feature, SMPLR returns to ARFRG which then immediately returns to ADIS1. A normal return from SMPLR to ARFRG causes a call to subroutine FLPFL, which outputs the sampled
data array to disk FL2 under the current filename. FLPFL then returns to ARFRG which then calls subroutine FLINC. FLINC increments the two-digit filenumber, the last two characters of the filename, and checks for overflow. If no overflow occurs, FLINC returns to ARFRG which then calls SMPLR and the cycle begins again when SMPLR receives another trigger pulse. Since FLINC incremented the filename, the next sample set taken will eventually be stored on disk with this new filename. The user may exit from this loop by pressing the teletype "S" key; an exit to ADISI will occur when SMPLR tries to take the next sample, regardless of when the "S" key is pressed.

APCHK:

APCHK is called by AOFRG and ARFRG to check sampling parameters for validity. The following checks are performed: (1) the number of channels requested must be from 1 to 8, (2) the starting channel must be between 0 and 7, (3) the number of channels plus the starting channel must be 8 or less, (4) the timebase must be 1, 2, or 5, (5) the timebase multiplier must be between 0 and 4, (6) the total number of samples (samples per channel times number of channels) must be 1020 or less, and (7) the last two characters (ASCII) of the filename must be numbers.

Failure of any check causes an appropriate error message and a return with the passed argument IFAIL set to 1. Successful checks result in a return with IFAIL = 0.
SMPLR (version SMPL1):

This subroutine is called by ACFRG and ARFRG and does all analog data sampling, data array packing and formatting, and various housekeeping and monitoring functions.

SMPLR passes one argument, KSTOP, which is used to implement the user stop feature. SMPLR begins its operation by setting KSTOP equal to zero. SMPLR then constructs and inserts the first three codewords in the data array, and instructs the programmable real time clock. The outer sample loop (samples per channel) is then entered, user stop is checked, the A/D converter and multiplexer are instructed and SMPLR then idles in a trigger wait loop. Once triggered, the clock runs continuously and generates an overflow signal once each sample period as per its instructions. Clock trigger or overflow causes entry of the inner sample loop (number of channels) where the specified number of channels are sampled as quickly as possible, starting with the specified starting channel. The multiplexer has previously been given this information along with an instruction to increment channels each time the A/D takes a sample. When all channels have been sampled, the inner loop exits to the start of the outer loop. User stop is checked, the converter and multiplexer instructions are refreshed and SMPLR then waits for the clock to overflow. Clock overflow after the sample period elapses starts the inner sample loop again.

When the outer loop has finished all cycles, SMPLR reads the 12 bit input word from the MI703 input interface and checks the failure bit contained in the word. If the failure bit is set, indicating that the
sample set just taken is "bad", control passes back to the start of SMPLR. For "good" data, the 12 bit input word, containing target deflection angle and radius information, is rearranged and stored as word 4 of the sampled data array. SMPLR then returns to its calling program. The normal return is with KSTOP = 0 as initialized. If user stop is detected by SMPLR, an immediate return is made with KSTOP = 1.

FLPFL:

FLPFL is called by AOFRG and ARFRG to output the sampled data array to disk FL2 under the current filename. Only that part of the data array containing sampled data (as per sampling parameters), and of course, the codewords, will be output.

FLINC:

FLINC is called by subroutine ARFRG to increment the filename number, the last two ASCII digits of the filename. Incrementing is done directly in ASCII with no conversion. If the file number to be incremented is 99 an overflow message is generated and an immediate return is made with a passed argument, KOVER, used to flag the overflow.

This completes the discussion of ADIS01 and a program listing is included on the following pages.
C PROGRAM ADISI INTERPRETS COMMANDS AND CALLS
C THE APPROPRIATE SUBROUTINES TO CHANGE
C SAMPLING PARAMETERS. ADISI IS A MODIFICATION
C OF ADIS AND INCLUDES INITIALIZATIONS OF
C SAMPLING PARAMETERS CONVENIENT FOR EYETRACKER
C DATA SAMPLING.

COMMON ICHNS, ICHNL, IMSEC, IT3, ISAMP, IEN1
COMMON IEN2, IEN3, F1ME, ICHCT, ISPCT, ICKEN
COMMON ITMBS

ICHNS=2
ICHCT=-2
ICHNL=3
IMSEC=5
ITMBS=-500
ITM3=0
ISAMP=500
ISPCT=-500
IEN1=352
IEN2=288
IEN3=288
ICKEN=-1167

IAC=67
IA1=113
IAT=84
IAM=77
IAS=83
IAE=69
IAO=79
IAR=82
IAF=70
IAP=30

WRITE(1, 5)
6 FORMAT('ADISI CMD ARE: AC, A1, AS, AT, AM, AE, '
C 'AF, AP, AO, AR')
CALL AFPAR
GO TO 290

READ (1, 16) ICMD
15 FORMAT(' ', 2)
16 FORMAT(' ', 2)
C CMD: AC USED TO SET NUMBER OF CHANNELS.
36 IF (ICMD-1-A) 35, 262, 39
C CMD: A1 USED TO SET STARTING CHANNEL.
39 IF (ICMD-IA1) 40, 210, 43
C CMD: AT USED TO SET TIME BASE.
40 IF (ICMD-1AT) 50, 220, 53
C CMD: AM USED TO SET TIME BASE MULTIPLIER.
IF (ICMD-IA1) 60, 230, 60
C CMD:AS USED TO SET SAMPLES PER CHANNEL.
63 IF (ICMD-IA5) 70, 240, 70
C CMD:IAE ALLOWS SCHMITT EVENT ENABLING.
73 IF (ICMD-IAE) 80, 250, 80
C CMD:AO STARTS SAMPLING IN ONE-SHOT MODE.
80 IF (ICMD-IAO) 90, 260, 90
C CMD:AR STARTS SAMPLING IN RECYCLE MODE.
90 IF (ICMD-IAAR) 100, 270, 100
C CMD:AF ALLOWS FILE NAME ENTRY.
100 IF (ICMD-IAF) 110, 280, 113
C CMD:AP LISTS CURRENT PARAMETERS.
110 IF (ICMD-IAAP) 120, 290, 130
120 WRITE (1, 131)
121 FORMAT('INVALID CMD')
131 GO TO 15
200 CALL ACPRG
210 GO TO 15
210 CALL AIPRG
220 GO TO 15
220 CALL ATPRG
230 GO TO 15
230 CALL AMPRG
240 GO TO 15
240 CALL ASPRG
250 GO TO 15
250 CALL AEPRG
260 GO TO 15
260 CALL AOPRG
270 GO TO 15
270 CALL ARPRG
280 GO TO 15
280 CALL AFPRG
290 GO TO 15
290 CALL APPRG
END
ACPRG ALLOWS NUMBER OF CHMLS TO BE SPEC'D.

SUBROUTINE ACPRG
COMMON ICNS, ICHM1, IMSEC, ITM1, ISAMP, ISAM
COMMON IEN2, IEN3, FM1E, ICHCT, ISPCT, ICKEN
COMMON ITM3S
20 READ(1,21)ICNS
21 FORMAT('ENTER NUMBER OF CHANNELS: ',15)
53 IF(ICNS) 100, 100, 50
100 WRITE(1,101)
101 FORMAT('INVALID')
GO TO 26
60 ICHCT=-ICNS
RETURN
END

AIPRG ALLOWS STARTING CHANNEL TO BE SPEC'D.

SUBROUTINE AIPRG
COMMON ICNS, ICHM1, IMSEC, ITM1, ISAMP, ISAM
COMMON IEN2, IEN3, FM1E, ICHCT, ISPCT, ICKEN
COMMON ITM3S
23 READ(1,21)ICNH1
21 FORMAT('ENTER START CHANNEL: ',15)
53 IF(ICHM1) 100, 50, 50
100 WRITE(1,101)
101 FORMAT('INVALID')
GO TO 26
60 CONTINUE
RETURN
END
ASPRG ALLOWS USER TO SPEC NUMBER OF SAMPLES PER CHANNEL, CHECKS TO INSURE SPECIFIED NUMBER BETWEEN 1 AND 1920. FURTHER STORAGE OVERFLOW CHECKS DONE ELSEWHERE.

SUBROUTINE ASPRG
COMMON ICHNS, ICHM1, IMS, IMSAMP, IEN1
COMMON IEN2, IEN3, FNME, ICHCT, ISPCT, ICKEM
COMMON ITM3S
20 READ(1,21)ISAMP
21 FORMAT("ENTER SAMPLES PER CHANNEL: ",15)
50 IF(ISAMP)133,130,53
100 WRITE(1,131)
131 FORMAT("INVALID")
GO TO 20
60 ISPCT=-ISAMP
RETURN
END

ATPRG ALLOWS TIMEBASE SPECIFICATION

SUBROUTINE ATPRG
COMMON ICHNS, ICHM1, IMS, IMSAMP, IEN1
COMMON IEN2, IEN3, FNME, ICHCT, ISPCT, ICKEM
COMMON ITM3S
20 READ(1,21) IMS
21 FORMAT("ENTER TIMEBASE: 1, 2, OR 5 (MSEC): ",15)
50 IF(IMSEC-1)50,100,53
100 WRITE(1,101)
101 FORMAT("INVALID")
GO TO 20
60 ITMBS=-100
RETURN
110 ITMBS=-200
RETURN
120 ITMBS=-500
RETURN
END
C AMPRG ALLOWS USER TO SPEC TIMEBASE MULTIPLIER, CHECKS FOR VALIDITY (0 TO 4), THEN CHANGES THE APPROPRIATE BITS (3-5) OF THE CLOCK ENABLE WORD.

SUBROUTINE AMPRG
COMMON ICHN5, ICHN1, IMSEC, ITBM, ISAMP, IEN1 COMMON IEN2, IEN3, FNAME, ICHCT, ISPCT, ICKEN COMMON ITM3S

20 READ(1, 21) ITBM
21 FORMAT('ENTER TIMEBASE MULTIPLIER: ', IS) IF(100-ITBM) 100, 52, 52
50 IF(4-ITBM) 100, 60, 60
100 WRITE(1, 101)
101 FORMAT('INVALID')
GO TO 20
60 JTBM = 5-ITBM
CLA CLL
S TAD \\JTBM
S RTL; RTL; RTL
S DCA \\JTBM
S TAD \\ICKEN
S AND (7377
S TAD \\JTBM
S DCA \\ICKEN
RETURN
END
SUBROUTINE AEPRG
COMMON ICNS, ICH1, IMSEC, ITP, ISAMP, IEN1
COMMON IEN2, IEN3, FNUM, ICHCT, ISPCT, ICKEM
COMMON ITMBS
READ(1, 21) IEN1, IEN2, IEN3,
C FORMAT(ENTER E OR D FOR EVENT 1: 'A1,' 2:')
WRITE(1, 41) IEN1, IEN2, IEN3
C ENABLE EVENT 1
CONTINUE
CLA
TAD ICKEM
AND (7776
IAC
DCA ICKEM
GO TO 60
C DISABLE EVENT 1
CONTINUE
CLA
TAD ICKEM
AND (7776
DCA ICKEM
GO TO 60
C ENABLE EVENT 2
CONTINUE
CLA
TAD ICKEM
AND (7775
TAD (0202
DCA ICKEM
GO TO 30
C    DISABLE EVENT 2

250 CONTINUE
S  CLA
S  TAD NICKEN
S  AND (7775
S  DCA NICKEN
GO TO 80

C    ENABLE EVENT 3

300 CONTINUE
S  CLA
S  TAD NICKEN
S  AND (7773
S  TAD 0004
S  DCA NICKEN
RETURN

C    DISABLE EVENT 3

350 CONTINUE
S  CLA
S  TAD NICKEN
S  AND (7773
S  DCA NICKEN
RETURN
END
C APFRG ALLOWS USER TO DECLARE FILENAMES IN THE
C FORMAT AAAANN, WHERE A'S ARE ALPHANUMERICS
C AND N'S ARE DIGITS.

SUBROUTINE APFRG
COMMON ICNS,ICNI,IMSEC,ITBM,ISAMP,IEN1
COMMON IEN2,IEN3,FNME,ICHCT,ISPCT,ICKEN
COMMON ITMB5
20 READ(1,21) FNME
21 FORMAT('ENTER FILENAME: ',A6)
IACL=0
SCALL 0,CLEAR
SCALL 1,FAD
SARG 
FNME
S CLA
S TAD ACL
S DCA \ IACL
IF(IACL+976)100,50,50
50 IF(IACL+391)60,60,100
100 WRITE(1,101)
101 FORMAT('INVALID FILE NUMBER')
GO TO 20
60 WRITE(1,61)FNME
61 FORMAT('FILES WILL START WITH: ',A6)
RETURN
END

C APFRG PRINTS A LIST OF SPEC'D PARAMETERS.

SUBROUTINE APFRG
COMMON ICNS,ICNI,IMSEC,ITBM,ISAMP,IEN1
COMMON IEN2,IEN3,FNME,ICHCT,ISPCT,ICKEN
COMMON ITMB5
20 WRITE(1,21)ICNS,ICNI
21 FORMAT(I2,' CHANNELS,STARTING WITH CHNL',I4)
WRITE(1,31)ISAMP
31 FORMAT(I5,' SAMPLES PER CHANNEL')
WRITE(1,41)IMSEC,ITBM
41 FORMAT('SAMPLE PERIOD:'I2,' MSEC TIMES 10''
'**'I2,'')
WRITE(1,51)IEN1,IEN2,IEN3
51 FORMAT('EVENT 1: ',A2,' 2: ',A2,' 3: ',A2)
WRITE(1,61)FNME
61 FORMAT('NEXT FILE: ',A6)
RETURN
END
AOPRG MANAGES SAMPLING IN ONE-SHOT MODE.

SUBROUTINE AOPRG
COMMON ICHNS, ICHM1, INSEC, IT3M, ISAMP, IEN1
COMMON IEN2, IEN3, FNAME, ICHGT, ISPCT, ICKEN
COMMON ITMBS
CALL APCHK(IFAIL)
IF(IFAIL)100,20,100
20 CALL SMPLR(ISTOP)
IF(ISTOP)100,30,100
30 CALL FLPFL
100 RETURN
END

ARPRG MANAGES SAMPLING IN RECYCLE MODE

SUBROUTINE ARPRG
COMMON ICHNS, ICHM1, INSEC, IT3M, ISAMP, IEN1
COMMON IEN2, IEN3, FNAME, ICHGT, ISPCT, ICKEN
COMMON ITMBS
CALL APCHK(IFAIL)
IF(IFAIL)100,20,100
20 CALL SMPLR(ISTOP)
IF(ISTOP)100,30,100
30 CALL FLPFL
CALL FLIMC(IOVER)
IF(IOVER)100,30,100
100 RETURN
END
APCHK checks specified sampling parameters. If OK, returns IFAIL=0; if not, prints error message and returns IFAIL=1.

SUBROUTINE APCHK(IFAIL)
COMMON ICHNS,ICHN1,IMSEC,ITEM,ISAMP,ITM3
COMMON ITM3S
COMMON ICHCT,ISPCT,ICKEN
COMMON ITM3S
IFAIL=0

C number of channels
IF(ICHNS)30,30,20
20 IF(8-ICHNS)30,40,40
30 WRITE(1,31)
31 FORMAT('BAD CHANNELS')
IFAIL=1

C start channel
40 IF(ICHN1)60,50,50
50 IF(7-ICHN1)60,70,70
60 WRITE(1,61)
61 FORMAT('BAD START CHANNEL')
IFAIL=1

C channels plus start channel
70 IF(ICHNS+ICHN1-8)130,103,90
90 WRITE(1,91)
91 FORMAT('BAD CHANNEL SUM')
IFAIL=1

C timebase
100 IF(IMSEC-1)110,140,110
110 IF(IMSEC-2)120,140,120
120 IF(IMSEC-5)133,140,130
133 WRITE(1,131)
131 FORMAT('BAD TIMEBASE')
IFAIL=1

C timebase multiplier
140 IF(ITM3)160,150,150
150 IF(4-ITM3)160,170,170
163 WRITE(1,161)
161 FORMAT('BAD TIMEBASE MULT')
IFAIL=1

C SAMPLES CHECKS

170 IF(ISAMP)193,190,180
180 IF(ISAMP+ICEMS-1020)230,200,190
190 WRITE(1,191)
191 FORMAT('BAD SAMPLES')
IFAIL=1

C FILENUMBER

200 KACL=0
SCALL 0,CLEAR
SCALL 1,FAD
SARG 'FNME
S CLA
S TAD ACL
S DCA \KACL
210 IF(KACL+976)220,210,210
210 IF(KACL+391)234,230,220
220 WRITE(1,221)
221 FORMAT('BAD FILENUMBER')
IFAIL=1

230 RETURN
END
SMPLI CONTROLS A/D SAMPLING AND DATA STORAGE
AS PER CLOCK AND A/D PARAMETERS SPEC'D BY
OTHER SUBROUTINES. A USER STOP FEATURE IS
INCLUDED: KEYBOARD "S" IS DETECTED BETWEEN
SAMPLES TO CAUSE AN IMMEDIATE RETURN WITH
KSTOP=1. SAMPLING STARTS IF AN ENABLED EVENT
(SCHMITT TRIGGER FIRES) OCCURS. IN ADDITION,
SMPLI READS AN M1703 INPUT INTERFACE AT THE
END OF SAMPLING. ACTION TAKEN ON THIS DATA
IS SPECIFIC TO OPERATION OF THE SACCAD
TARGET DEFLECTOR: SAMPLED DATA IS REJECTED
IF A FAILURE BIT IS PRESENT AND ANGLE AND
RADIUS DATA IS DECODED AND STORED AT WORD
4 OF THE DATA ARRAY.

SUBROUTINE SMPLRC(KSTOP)

COMMON ICHMS,ICINI,IMSEC,ITLM,ISAMP,IEN
COMMON IEN2,IEN3,FMME,ICHCT,ISPCT,ICKEN
COMMON ITMBS

SFDATA, COMMON 2003 / 1024 WORD DATA ARRAY.

S OPDEF CLZE 6133 /CLR CLK EVA PER AC.
S OPDEF CLOE 6132 /SET CLK EVA PER AC.
S OPDEF CLAB 6133 /AC TO CLK CNT REG.
S OPDEF CLSA 6135 /CLK STATUS TO AC.
S SKPDF CLSK 6131 /SKP ON CLK FLG.
S OPDEF ADCL 6530 /CLEAR A/D.
S OPDEF ADLM 6531 /AC2-11 TO A/D MUX.
S OPDEF ADST 6532 /CLR FLGS, START A/D.
S OPDEF ADRA 6533 /CLR FLG3, READ A/D.
S OPDEF ADLE 6536 /AC2-5 TO A/D EVA REG.
S SKPDF ADSK 6534 /SKP ON A/D DONE FLAG.
S OPDEF DCAI 3400 /GNGE TO DATA FLD 0.
S OPDEF CDF7 6221 /GNGE TO DATA FLD 1.
S OPDEF CDF1 6211 /BYTE SWAP.
S OPDEF CLAF 6152 /CLR M1703 FLGS.
S OPDEF RDDA 6154 /READ M1703, CLR FLG.
S SKPDF SDFS 6153 /SKP ON M1703 FLG.

S KSTOP=9
S START, CLA CLL
S TAD (7410) /SKP FOR START ROUTINE.
DCA Skp1

TAD \ICKEM
AND (07)7
TAD C5060
DCA \ICKEM

TAD FDAAD
DCA FDPNT
TAD \ISPCT
DCA FDCT

TAD \ICKEM
BSV
TAD \ICHNI
JMS FSTR

TAD \ISPCT
JMS FSTR

TAD \ITBH
BSV
TAD \INSEC
JMS FSTR

TAD \ISPCT
JMS FSTR

CLA CLL CMA
CLZE
CLA

TAD \ITM3S
CLAB
CLA
TAD \ICKEM
CLOE
CLA

SSTRT1, JMS SPCHK
ADCL
TAD ADENA
ADLE

TAD \ICHNI
ADLM

TAD \ICHCT
DCA CHCNT

Skp1, 0
JMP Al
Skp1, CLSK
JMP Ckfi
ADST

/SET UP CLK ENA WORD.
/SET DATA STORAGE PTR.
/SET SMPLS CNTR.
/SET #CHNLs CODE.
/ADD START CHNL CODE.
/PUT AT FDATA, WORD 1.
/PUT - (SMPLS PER CHNL)
/AT WORD 2.
/SET TMBASE 'MULT CODE.
/ADD TMBASE.
/PUT AT WORD 3.
/CLR WORD 4.
/CLEAR CLK.
/SET CLK TMBASE.
/INSTRUCT CLK.
/INSTRUCT A/D.
/SET A/D FIRST CHNL.
/SET CHNL CNTR.
/START ROUTINE. Skp OR NOP.
/SCHMITT FIRE?
/START A/D.
S TAD (?003) /CHANGE SKP TP NOP.
S DCA SKP1
S CLA CLL CML IAC RTL
S IAC /AC=7 CLEAR SCHMITT ENABLES.
S CLZ
S JIP A2
S A1, CLSK /SKP ON CLOCK OVERFLOW.
S JIP A1
S A2, CLSA /(CLEAR OVERFLOW FF).
SADFG1, ADSK /A/D DONE?
S JMP ADFG1
S ADRE /READ A/D.
S ADST /RESTART A/D.
S JMS FSTR /STORE DATA WORD.
S ISZ GHINT /ALL CHILS DONE?
S JMP ADFG1 /NO.
S ISZ FDCNT /YES, ALL 5XPLS DONE?
S JMP STRT1 /NO, RECYCLE.

C PICK UP M?703 DATA, DO FAILCHECK, PUT AT WORD 4.
S CLA CLL CML IAC RAL /+3 TO AC.
S TAD FDAOAD /SET WORD 4 POINTER.
S DCA FDPNT
SINSKP, SDSF5
S JMP INSKP
S RDAA /READ M?703.
S CLL RAL /FAILURE CHECK.
S SZL
S JMP START /FAILURE.
S RAR /NO FAILURE.
S DCA TEMPU /CHANGE ANGLE CODE SO
S TAD TEMPU /ANGLE=15*CODE.
S RAL
S BSW
S AND (3
S CLL RAL
S RTL
S CIA
S TAD TEMPU
S JMS FSTR
RETURN

PAGE
S FSTR, 0 /ROUTINE TO STORE DATA WORD
S CDF1 /AND INCR PTR.
S DCAI FDPNT
S CDF0
INC FDPNT
JMP I FSTR

SSPCHK, 0
CLA CLL
KSF
JMP I SSPCHK
KR3
TAD ("S"
SZA CLA
JMP I SSPCHK
KSTOP=1
RETURN

SFDPNT, 0
SFDCNT, 0
SCHCNT, 0
SADEMA, 0320
SFDAAAD, FDATA
STEMP!, 0

USER STOP ROUTINE.

KEYPRESS?

YES."S" ?

NO.

DATA STORAGE PTR.

SMPS CNTR.

CHNL CNTR.

A/D:CLK INH, AUTO INCR.

STORAGE START ADDRESS.

1M703 DATA HERE TEMPORARILY.
FLPFL OUTPUTS THE DATA ARRAY TO DEVICE FL2:
UNDER THE FILENAME FMME.

SUBROUTINE FLPFL

COMMON ICHNS, ICHN1, IMSEG, ITBY, ISAMP, IEN1
COMMON IEN2, IEN3, FMME, ICHCT, ISPCT, ICKEN
COMMON ITMBS

COMMON IDATA
DIMENSION IDATA(1024)

IURDS = ISAMP * ICHNS + 4

CALL OPEN('FL2', FMME)

WRITE(4,31)(IDATA(!!), M=1, IURDS)

CALL CLOSE
WRITE(1,41)FMME

RETURN
END
C FLINC INCREASES THE FILENUMBER "NN" OF FILENAME AAAAN. IF NN=99, DO INCREMENT, C AN ERROR MESSAGE, AND RETURN WITH COVER=1.

SUBROUTINE FLINC(KOVER)
COMMOM ICHMS,ICHM1,IMSEC,ITSM,ISAMP,IE11
COMMOM IE12,IE13,FNME,ICHCT,ISPCT,ICKEN
COMMOM ITM3S

COVER=0

SCALL 0,CLEAR /PUT FILENUMBER AT IL.
SCALL 1,FAD
SARG \FNME
S CLA
S TAD ACL
S DCA \IL
 IF(IL+391)52,49,53
43 WRITE(1,41)
41 FORMAT('FILENUMBER OVERFLOW')
COVER=1
RETURN

50 ILA=0 /INCREMENT FNME FILENUMBER.
S TAD \IL
S TAD (+7) /(UNITS+7), INCR TENS IF REQ'D.
S DCA \IL /SAVE NEW TENS.
S TAD \IL
S AND (77) /UNITS+7.
S SNA
S TAD (66) /FIXUP FOR ZERO.
S TAD (-6) /(UNITS+7)-6=UNITS+1.
S DCA \ILA /SAVE NEW UNITS.
S TAD \ILA /PUT NEW UNITS WITH TENS.
S AND (7700)
S TAD \ILA
S DCA \IL

SCALL 0,CLEAR /INSERT NEW NUMBER IN FNME.
SCALL 1,FAD
SARG \FNME
S TAD \IL
S DCA ACL
SCALL 1,STQ
SARG \FNME

RETURN
END
Design considerations:

The purpose of APLOTO is to plot sampled analog voltage data previously sampled, and stored as a data file on disk drive FL2 by ADIS01 programming. Design considerations were that APLOTO be easy to use by non-technical personnel and that APLOTO be capable of generating scalable x-y plots with any variable, including time, on any axis. In addition, it was desirable that APLOTO be capable of generating "slow motion" plots to allow the user to observe recorded eye movements plotted as though the recorded event were actually occurring at a much reduced speed.

To meet the above requirements APLOTO incorporates the following features: (1) all user interaction is by means of teletype commands, (2) each entered command causes a request for specific information pertinent to the command, (3) error checks are performed to inform the user of meaningless entries and to reject such information, (4) scale factors are entered by the user in "volts/inch" or "msec/inch" depending on the variable chosen for the axis, (5) data may be plotted in "slow motion", slowed down by as much as a factor of 2000, (6) "ticks" may be generated on the plot every specified number of samples, (7) ticks may be generated alone without plotting data between the
ticks, and (8) a user stop feature may be used to halt plotting and cause the pen to immediately return to the plot origin.

Commands and features:

Commands may be entered on the teletype keyboard whenever APLOTO is in command mode, indicated by the printing of a ">" at the teletype left margin. Commands are PF, PX, PY, FS, FT, PP, and PG and are explained below:

PF: PF calls subroutine PFPRG which requests user entry of a data file name. The user entered filename must be available as a data file on disk drive FL2 and the file must be a data file previously created by ADIS01. PFPRG gets the file and reads it into core, lists the sampling parameters coded at the beginning of the file, then requests x and y axis variables and scale factors (see commands PX and PY). Successful entry of these numbers results in a return to command mode.

PX: PX calls subroutine PVPRG(0) which requests user entry of x-axis variable and scale. Valid entries for variables are "T" for time or any channel number, 0 to 7, that was sampled for the data file currently in core. If "T" is given as the variable PVPRG requests the axis scale factor in milliseconds per inch; valid entries are real numbers from 2.0 to 10000.0. If an existing channel number is given as a variable the scale is requested in volts per inch; valid entries are real numbers.
from 0.01 to 1.0. After all PVPRG requests have been satisfied, PVPRG returns to APLOTO command mode.

PY: PY calls subroutine PVPRG(1) which requests user entry of y-axis variable and scale in exactly the same manner as outlined above for command PX.

PS: PS causes a request for user entry of a "slow-factor". Valid entry is any integer from -2047 to +2047. Numbers between -2047 and 1 disable the slow motion plotting feature. For an entry between 2 and 2047 the slow motion feature will be used on all subsequent plots as such: between each point plotted, the plotter will wait a delay time equal to the slow factor times the sampling time originally used to sample the data. Thus, the data may be plotted as though it were occurring in real time but at a rate reduced by the slow factor. Acceptance of a valid entry returns the user to command mode.

PT: PT causes a request for user entry of "samples per tick". Any integer from -2047 to +2047 may be entered. Entry of 0 disables the tick insertion feature. Entry of a positive integer N will cause the plotter to insert a tick, a 0.06" square, every N samples (data file samples) plotted. The tick is drawn centered directly on the data point being plotted. Entry of a negative integer -N causes the ticks to be plotted every N samples but disables plotting between ticks by raising the plotter pen at all other points along the plot.
PP: PP causes a call to subroutine PPPRG which lists all current plotting parameters: x and y axis variables and scales and the numbers entered for the slow factor and samples per tick options. PPPRG then returns to APLOTO command mode.

PG: PG causes a call to subroutine PGPRG which does the plot according to the parameters previously specified using other commands. PGPRG calls subroutine FLTLN to plot between data points, waits an appropriate time between points if a slow motion plot is being done, enters ticks (0.06" box) when called for, and at the completion of the plot draws axes extending to the limits of the plot, then returns the pen to the origin and returns the user to APLOTO command mode.
APLOTO program operation:

APLOTO is comprised of a main FORTRAN II program APLOT and FORTRAN II/SABR subroutines PFFRG, PVPRG, PPFRG, PGFRG, and PLTLN. APLOT calls all subroutines except PLTLN in response to various keyboard commands; PLTLN is called by PGFRG.

Major variables are "passed" among programs via FORTRAN common storage. The variables are:

ICHNS - number of analog input channels originally sampled for the data file. Coded in the data file header.

ICHNi - starting channel of the sampled data. Coded in the data file header.

IMSEC - data sample period timebase. Coded in the data file header.

ITEM - data sample period timebase multiplier. Coded in the data file header.

ISAMP - number of samples per channel in the data file. Coded in the data file header.

FNME - data file filename, stored in ASCII A6 format.

IXCHN - x-axis variable (channel number or "T").

IYCHN - y-axis variable (channel number or "T").

PXSCA - x-axis scale factor.

PYSCA - y-axis scale factor.

ITSCA - slow factor for slow motion plotting.

ITICS - samples per tick number.
APLOT and subroutine operations are outlined below, followed by a program listing of APLOTO.

APLOT:

Upon startup, APLOT performs initializations to disable the slow motion plotting feature and the tick insertion feature, prints an opening message, and calls PFPRG and PVPRG (twice) to request a data filename and the x and y axis variable and scales. APLOT then enters command mode and will accept and interpret entered keyboard commands, transferring control to appropriate places to execute valid commands. Commands PF, PX, PY, PP, and PG cause subroutine calls to be executed; commands PS and PT are managed within APLOT.

Note that whenever command PP is given to obtain a data file from FL2 that PFPRG is called to get the data file, then PVPRG is called twice. This forces the user to specify axes variables and scales which may then be checked against sampling parameters obtained when the file was retrieved.

PFPRG:

This subroutine is called at APLOTO startup and thereafter in response to command PF. PFPRG requests user entry of a data filename, then gets the file from disk drive FL2 and stores it in core in array IDATA(1024). Files specified must be ADISO1-created data files. PFPRG then unpacks the sampling parameters coded in the first three words of the array and prints a listing of these sampling parameters originally used by ADISO1 to obtain the data in the file.
PVPRG:

This subroutine is called by APLLOT in response to commands PF, PX, and PY, and requests user entry of an axis variable and scale. One variable is passed to determine which axis; a call to PVPRG(0) causes requests for x-axis information and a call to PVPRG(1) causes requests for y-axis information. The axis variable is first requested and the user entry is checked for validity, i.e. the specified variable must be time (T) or a channel number that was sampled for the data file currently in core. User entry of the axis scale factor is then requested in units relevant to the variable, either msec/inch or volts/inch. Unacceptable user responses to PVPRG requests cause a repeat of the request and when all requests are satisfied, PVPRG returns to APLLOT.

PPFRG:

This subroutine is called by APLLOT in response to command PP and prints a list of all plotting parameters: x and y axis variables and scales and the slow factor and samples per tick numbers. PPFRG then returns to APLLOT command mode.

PGPRG:

This subroutine is called by APLLOT in response to the command PG, PGPRG and a subroutine PLTN control plotter operation to generate the type of plot previously specified by the user by means of other commands.

PGPRG first performs various initializations, then issues instructions to the programmable real time clock in case a slow
notion plot is being done. Then, based on axis variable previously entered by the user, PGPRG sets up indices to either obtain data for the channels being plotted and/or calculate plotter movements based on time as an axis variable.

PGPRG then enters a plotting loop, which will be executed as many times as there are samples per channel in the data array; one sample is plotted in each pass through the loop. In one loop transit PGPRG first calculates the required number of plotter x and y axis steps and retains the largest movement yet calculated for drawing axes later. PGPRG then enters a delay loop if a slow motion plot is being done. After this, PGPRG checks to see if a tick need be drawn at this sample point and does so if required. Control then transfers back to the start of the loop to plot the next point.

PGPRG exits from the loop after all data for the specified channel(s) has been plotted; x and y axes are then drawn in one inch increments to extend just past all plotted data. The plotter pen is then returned to the plot origin, the clock is turned off, and PGPRG returns to APLOTO command mode.

PLTLIN:

This subroutine is a modified version of a Digital Equipment Corporation SAER program "DIGITAL S-12-U", copyrighted 1971. Changes made to the program enable PLTLIN to (1) be used directly as a FORTRAN II subroutine with 3 passed arguments, and (2) monitor an N1703 input interface cabled to the plotter for detection and compensation of offscale plotter movements.
The purpose of PLTLN is to drive a digital x-y plotter in such a manner that the best straight line in plotted between co-ordinates previously and currently passed into the subroutine.

PLTLN is called as PLTLN(KFG,KX,KY). KX and KY are integer x and y co-ordinates measured in plotter steps (100 per inch for the EAI model 130) and each number must be in the range from -2047 to +2047. For calls to PLTLN with KFG = 0 or KFG = 1, PLTLN examines the currently passed KX and KY and plots the best straight line from the old KX and KY, stored from the last call, to the new KX and KY. If KFG = 0 the pen is lowered before the movement; the pen is raised for KFG = 1. PLTLN is initialized by calling with KFG = -1; KX and KY are ignored in this call, the pen is raised, and the old KX and KY are zeroed, then PLTLN returns.

This completes the discussion of APLOTO and a program listing is included on the following pages.
C APLLOT DECODES PLOTTING COMMANDS AND CALLS
C APPROPRIATE SUBROUTINES.

COMMON ICM5, ICMM, I15EC, IT5M, I5AMP, F3M3E
COMMON IXCM1, IYCM1, PXCM2, PYCM2, ITSCA, ITICS

IPF=1030
IPX=1048
IPY=1049
IPS=1043
IPT=1344
IPP=1040
IPG=1331

ITICS=3
ITSCA=0

WRITE(1,21)

21 FORMAT('APLOT CMD ARE: PF, PX, PY, PS, PT, PP, 'PG.'/

25 CALL PFPRG
CALL PVPRG(2)
CALL PVPRG(1)

33 READ(1,31)ICMD

31 FORMAT('>',',A2')

50 IF(ICMD-IPF)53,25,53
53 IF(ICMD-IPX)63,210,63
60 IF(ICMD-IPY)73,223,73
70 IF(ICMD-IPS)83,230,83
80 IF(ICMD-IPT)93,243,93
90 IF(ICMD-IPP)103,250,100
100 IF(ICMD-IPG)120,260,120

120 WRITE(1,121)
121 FORMAT('INVALID CMD')
GO TO 30

213 CALL PVPRG(0)
GO TO 33
223 CALL PVPRG(1)
GO TO 32
233 READ(1,231)ITSCA
231 FORMAT('SLOW FACTOR: ',',I5')
GO TO 30
243 READ(1,241)ITICS
241 *FORMAT(*SAMPLES/TICK: ',15)*
   GO TO 30
250 CALL PPPRG
   GO TO 30
260 CALL PGPRG
   GO TO 30
END
SUBROUTINE PFPRG
COMMON ICINS, ICH11, IMSEC, ITBM, ISAMP, FMME
COMMON IXCIN, IYCH1, PXSCA, PYSCA, ITSCA, ITICS
COMMON IDATA

OPDEF BSN 7002

DIMENSION IDATA(1624)

READ(1,21)FMME
FORMAT('ENTER FILENAME: ',A6)

CALL IOPE1('FL2',FMME)
READ(4,31)IDATA
FORMAT(12A2)
ID=IDATA(1)

CALL CLL /GET # CHNL.
S TAD \ID 
S BSN 
S AND (77 
S DCA \ICIN1 
S TAD \ID /GET FIRST CHNL.
S AND (77 
S DCA \ICIN1 

ISAMP=-IDATA(2)
ID=IDATA(3)

S TAD \ID /GET TM3ASE 'ULT.
S BSN 
S AND (77 
S DCA \ITBM 
S TAD \ID /GET TM3ASE.
S AND (77 
S DCA \IMSEC 

WRITE(1,51)FMME, ICH11, ICH11, ISAMP, IMSEC, ITBM
FORMAT(/'FL2: ',A6=: '15,' CHNL, STARTING' 
C ' WITH:13/12X,15' SMP'S PER CHNL/13X, 
C 'SMP PERIOD: '12' MSE (TIMES 10**'12'')) 
RETURN 
END
PVPRG(2) ALLOWS DECLARATION OF PLOTTER
X-AXIS VARIABLE (CHNL) AND SCALE, PVPRG(1)
ALLOWS SAME FOR Y-AXIS.

SUBROUTINE PVPRG(IVA)
COMMON ICINS, ICHN1, ISEC, IT31, ISAMP, FME
COMMON IXCHN, IYCHN, PXSCA, PYSCA, IT3CA, ITICS

15 IF(IVA)25,20,25
20 READ(1,21)IVCHN
21 FORMAT('X VAR: ',A1)
GO TO 29
25 READ(1,26)IVCHN
26 FORMAT('Y VAR: ',A1)

29 IF(IVCHN-1312)30,120,30
30 IVCHN=(IVCHN+992)/64
IF(IVCHN-ICHN1)53,40,40
40 IF(IVCHN-ICINS+1)60,63,50
50 WRITE(1,51)
51 FORMAT('NO SUCH CHNL')
GO TO 15

60 READ(1,61)SCA
61 FORMAT('VOLTS/IN: ',E3.2)
IF(SCA-.01)80,73,70
73 IF(SCA-1.8)153,150,30
80 WRITE(1,81)
81 FORMAT('BAD SCALE')
GO TO 15

120 READ(1,131)SCA
131 FORMAT('MSEC/IN: ',E5.2)
IF(SCA-2.0)80,110,113
113 IF(SCA-10.000.)153,150,30

150 IF(IVA)170,160,170
160 IXCHN=IVCHN
PXSCA=SCA
RETURN
170 IYCHN=IVCHN
PYSCA=SCA
RETURN
END
SUBROUTINE PPRG
COMMON IXCHN, IYCHN, PXSCA, PYSCA, ITSCA, ITICS
COMMON ICCT, ICCT1, ICCT3, ICCT4, ISA1P, PNT

IF (IXCHN - 1312) .GT. 33, 20, 31
20 WRITE (1, 21) IXCHN, PXSCA
21 FORMAT ("X VAR: ', A1', ', F7.2', ' SEC/IN")
GO TO 40
30 WRITE (1, 31) IXCHN, PXSCA
31 FORMAT ("Y VAR: ', 12', ', F7.2', ' VOLS/IN")

40 IF (IXCHN - 1312) .GT. 60, 51, 58
50 WRITE (1, 51) IYCHN, PYSCA
51 FORMAT ("Y VAR: ', A1', ', F7.2', ' SEC/IN")
GO TO 70
60 WRITE (1, 61) IYCHN, PYSCA
61 FORMAT ("Y VAR: ', 12', ', F7.2', ' VOLS/IN")

70 WRITE (1, 71) ITSCA
71 FORMAT ("GAIN FACTOR: ', I5")

WRITE (1, 81) ITICS
81 FORMAT ("SAMPLES/TICK: ', I5")

RETURN
END
PGP2G PLOTS DATA ACCORDING TO SPECIFIED PLOTTING PARAMETERS.

SUBROUTINE PGP2G
COMMON ICONS, IGN1, INSEC, IT51, ISATP, FIME
COMMON IXCIN, IYCH1, PXSCA, PYSCA, ITSCA, ITICS
COMMON IDATA

DIMENSION IDATA(1324)

S OPDEF CLZE 6130
S OPDEF CLOE 6132
S OPDEF CLA3 6133
S OPDEF CLSA 6135
S SKPDF CLS1 6131
S OPDEF BSV 7062

IXI=0
IYI=0
IXI*XPI=0
IYI*YPI=0
IYI*XPI=2
IYI*XPI=8
PLFAC=162
SP=FLOAT(IMSEC)+10**FLOAT(ITJ1)

C CLOCK SETUP FOR SLOW PLOTS.

JT3M=5-ITB1
ITB1S=-103*IMSEC
S CLA CLL CMA
S CLZE
S CLSA
S CLA
S TAD \ITB1S
S CLA3
S CLA CLL
S TAD \JT3M
S BS1
S TAD (5010)
S CLOE
S CLA CLL

C INITIALIZE AND GET FIRST ARRAY INDICES.

CALL PLTLM(-1,0,0)
IFS=1
IF(IXCHI-1)=35,40,30
31 IXI=5+IXCHI-1CHI1
43 IF(IYCHI-1)=35,40,30
53 IYI=5+IYCHI-1CHI1

C DO THE PLOT.
67 DO 235 I=1,ISAMP
70 N=FLOAT(N-1)

C CALCULATE X VALUE.

IF(IYI)75,71,75
71 RIV=RJ*SP*PL FAC/PSCA
GO TO 79
75 RIV=(FLOAT(IDATA(IYI))/512.)*PL FAC/PSCA
IXI=IYI+ICHH5
79 IX=IFIX(RIV)

C CALCULATE Y VALUE.

IF(IYI)35,31,85
81 RIV=RJ*SP*PL FAC/PYSCA
GO TO 89
85 RIV=(FLOAT(IDATA(IYI))/512.)*PL FAC/PYSCA
IYI=IYI+ICHH5
89 IY=IFIX(RIV)

C RETAIN MAX EXCURSIONS.

IF(IX-IX'XP)125,105,100
101 IX'XP=IX
103 IF(IX-IX'XP)110,115,115
105 IX'XP=IX
107 IF(IY-IY'XP)125,125,125
112 IY'XP=IY
122 IF(IY-IY'XP)130,135,135
132 IY'XP=IY

C SLOW PLOT WAIT LOOP.

135 IF(ITSCA)150,153,140
143 DO 145 I=1,ITSCA
145 CONTINUE

C PLOT ONE POINT, CHECK FOR USER STOP.
PLTLN PLOTS THE BEST STRAIGHT LINE BETWEEN
THE OLD AND NEW CO-ORDINATES. X CO-ORDINATE
CORRESPONDS TO "KX", Y TO "KY". THESE ARE
STANDARD FORTRAN INTEGER VARIABLES AND MUST
BE >= -2347 AND <=2047. "KFG" CONTROLS
PLOTTING AS SUCH: KFG=-1, INITIALIZE;
KFG=1, PLOT WITH PEN UP; KFG=3, PLOT WITH
PEN DOWN.

SUBROUTINE PLTLN(KFG, KX, KY)

JFG = KFG
JX = KX
JY = KY

CLA CLL
TAD (4000
TAD \ JX
DCA \ JX
TAD (4000
TAD \ JY
DCA \ JY
CLA CLL
TAD \ JFG
SPA /MOVE PEN?
JMP PLOTA /NO: INITIALIZE.
TAD PLOTPH /ADD PEN STATUS.
CALL RTR
SPA CLA /ANY CHANGE?
JMP PLOT1 /NO: CONTINUE.
S'IL CLA
JMP A1 /GO LOWER THE PEN.
DCA PLOTPH /RAISE THE PEN.
PLPU
JMP A2
A1,133 PLOTPH /LOWER THE PEN.
PLPD
S A2, JMP PLOTUT /FLAGWAIT.
S JMP PLOT1 /CONTINUE.

S PLOT2, CLA
S PLPU /RAISE PEN.
S DCA PLOT2 /INITIALIZATIONS.
S TAD \\
S DCA PLOT2X /"S" TO X COORD.
S TAD \\
S DCA PLOT2Y /"U" TO Y COORD.
S DCA PLOT2Y
S JMP PLOT2
S PLOT2, CLA RETURN

S PLOT1, TAD PLOT1X /GET PREVIOUS X COORD.
S CIA CLL
S TAD \JX /FORM \JX=IPX.
S SNL /L=0: NPX<IX.
S CIA
S DCA PLOTDX /ABS VAL OF DIFFERENCE.
S RAL
S DCA PLOT1V /SAVE SIGN BIT.
S TAD \JX /UPDATE OLD X TO NEW X.
S DCA PLOT1X
S TAD PLOT1Y /GET PREVIOUS Y COORD.
S CIA CLL
S TAD \JY /FORM \JY=NPY.
S SNL /L=0: NPY<JY.
S CIA
S DCA PLOT1Y /ABS VAL OF DIFFERENCE.
S TAD PLOT1Y /SAVE SIGN BIT.
S RAL /BIT 10=1: DRUM DOWN, +X.
S DCA PLOT1V /BIT 11=1: PEN LEFT, +Y.
S TAD \JY /UPDATE OLD Y TO NEW Y.
S DCA PLOT1Y
S TAD PLOT1X
S CIA CLL
S TAD PLOT1Y
S SNL CIA
S JMP PLOT2 /L=0: DELTA X<DELTA Y.
S JMP PLOT2
S TAD PLOT2 /REVERSE NUMBERS.
S DCA PLOT2
S TAD PLOT2 /SET MAJOR MOTION INST.

S PLOT2, TAD PLOT2 /SET MAJOR MOTION INST.
S CALL BAR
S TAD PLOTT2
S A3, DCA PLOTM
S TAD PLOT3A
S DCA PLOT4A
S TAD PLOT4V /SET COMBINED MOTION.
S TAD PLOTT3
S DCA PLOTMV
S TAD I PLOTMV
S DCA PLDT3A
S TAD PLOTDX
S CLL BAR
S DCA PLOTNA
S TAD PLOTDX
S CMA
S DCA PLOTMV
S PLOT3, ISZ PLOTMV
S JMP A4
C ALL DONE WITH PLOT.
RETURN
S A4, TAD PLOTNA
S TAD PLOTDY
S DCA PLOTNA
S TAD PLOTNA
S CMA CLL
S TAD PLOTDX
S SZL CLA
S JMP PLOT4 /SINGLE MOTION.
S PLOTD3, TAD PLDT3A /COMBINED MOTION.
S JMP OFFSCA /OFFSCALE CHECK.
S TAD PLOTDX
S CIA
S TAD PLOTNA
S DCA PLOTNA
S A5, JMP PLOT3
S PLOT4, TAD PLOT4A /OFFSCALE CHECK.
S JMP A5
S PLOTT1, A6
S A6, 0340 /PEN RIGHT(-Y).
S 0320 /PEN LEFT(+Y).
S PLOTT2, A7
S A7, 0324 /DRUM UP(-X).
S 0310 /DRUM DOWN(+X).
S PLOTT3, A3
S A3, 0344 /UP-RIGHT.
/CHECK FOR OFFSCALE AND EXECUTE MOVEMENT.

S OFFSCA, 0
S DCA CMD /SAVE COMMAND.
S CLAF /CLR "OFFSCALE" FLG.
S TAD CMD /GET DIRECTION BIT.
S AND (0060) /CLH "OFFSCALE" FLG.
S SDFS /FLAG?
S SKP
S JMP B1 /NO, STORE BIT.
S DCA BITBUF
S DCA COUNT
S JMP EXEC
S B1, SNA /FLAG-RELATED MOVE?
S JMP EXEC /NO.
S AND BITBUF
S SNA CLA
S JMP B2
S ISZ COUNT /YES, UP COUNT.
S JMP EXEC
S B2, TAD COUNT
S SNA CLA /ALLOW ONSCALE MOVE?
S JMP EXEC /NO, DOWN COUNT.
S CLA
S TAD COUNT
S DCA COUNT
S TAD (0060) /DELETE ONSCALE MOVEMENT.
S CLA
S AND CMD
S DCA CMD
S EXEC, TAD CMD /GET ALLOWED CMD & DO IT.
S PLCF
S PLLR
S JMS PLOTWT
S JMP I OFFSCA

S PLOTWT, 0 /PLOTTER FLAG WAIT.
S Cl, PLSF
S JMP Cl
S PLCF
S CLA CLL
S JMP I PLOTWT
SPLTPN, 0
SPLTNX, 0
SPLTNY, 0
SPLTDX, 0
SPLTDY, 0
SPLTNA, 0
SPLTMV, 0
SPLTDHA, 0
SPLOTA, 0
S CMMD, 0
SBITBUF, 0
S COUNT, 0
END
SPROUT: Saccade processing routines

Processing considerations:

SPROUT was designed to automatically input and process sets of ADIS01-created data files, each data file containing raw data for a single, two-dimensional eye movement response to a step target deflection. The SPROUT analysis of each saccade data file yields latency and settling times to four different radii about the target. These results, for sets of up to 63 individual saccadic responses, are stored in a saccade summary array during processing. The summary array itself may then be stored as a data file on disk FL2, allowing efficient storage of saccade data and rapid access to data for many saccades for purposes of additional processing or plotting.

Design considerations for SPROUT include the following: (1) that SPROUT perform the above mentioned tasks, (2) that SPROUT be useable by non-technical personnel, (3) that a variety of utility features be available, such as input data scale factor changes and saccade deletions/additions to summary files, (4) that various checks be performed on saccade data during processing to avoid unknowingly including irregular data.
Input data:

Biological data is often fairly irregular in nature, corresponding to a pattern but with wide variations within the pattern. As far as was possible, the experimental setup used to generate saccadic responses has been designed to yield relatively regular data from the subjects. As such, it is possible to make quite a number of assumptions about the form of responses from normal subjects; this eases the processing task considerably.

Briefly, one experimental trial begins with the subject fixating on the target, a bright point of light in the center of an otherwise dark visual field. A warning buzzer is sounded for 1.5 seconds, then the target is deflected synchronously with the cessation of the sound. The target remains in the new position for 3 seconds and then returns to its original center position. The subject is instructed to follow the target as best as possible and eye rotation data is collected for 2.5 seconds by ADISOI, beginning at the instant of target deflection.

Subject eye movement response for one such trial is typically as follows: (1) an initial quiet period while the subject is still fixating the original target position, lasting 200-300 milliseconds (approximately). This is the latency period, followed by (2) a saccade, generally toward the new target position, followed by (3) an optional one or two saccades to correct positional inaccuracies, and finally (4) a relatively quiet period for the last second or so
while the subject fixates the target at its new position. Twitches, small saccades, and slow drifts of eye movement may occur during this final quiet period.

SPROUT is designed to process data consisting of x and y axis eye rotation records of typical responses to target deflections, as discussed above. SPROUT calculates the response latency and settling times to four different radii about the final target position.

Latency calculation:

Latency is defined as the time elapsing between the stimulus onset (target deflection) and the beginning of the subject response (first saccade). For processing purposes a velocity criterion has been used to determine the onset of response. A cutoff velocity was chosen such that (1) the velocity of normal eye drifts during the initial quiet period and the "velocity" due to electrical noise were both well below the cutoff velocity, and (2) the velocity of a typical small saccadic eye movement was well above the cutoff velocity. Fortunately, there exists great disparities in these velocities and a cutoff velocity of 20 degrees/second, corresponding roughly to the maximum velocity of a 1/3 degree saccade, was chosen.

The determination of latency is then done by SPROUT by calculating the total eye velocity at each data sample point, starting at the beginning of the record, and comparing the calculated velocity with the cutoff velocity. The latency period is then taken to end at that data sample just prior to the first sample at which the velocity exceeds
the cutoff velocity.

Settling time calculation:

Settling time to within a radius $r$ about the target positions is defined, for processing purposes, as the time elapsing between the target deflection and the start of the first period of 0.1 second or longer during which the subject's visual axis is directed within the given radius about the target in its final position.

Note that the settling time cannot be defined either as the time of the first entry into the given radius or the time of the last position outside the radius. The first criterion will yield settling times which may really be positional overshoots passing near the target while the second criterion will yield settling times which may be based on a twitch which occurs well after target acquisition has occurred. Thus the more complicated criterion must be used. The 0.1 second time is chosen as it is too long to falsely detect positional overshoots but short enough to miss most twitches or erroneous refixations that may occur after target acquisition.

Final target position must be known and referenced to the response data in order to calculate settling times. To avoid errors arising from temporary offsets resulting from small head movements, final target position is calculated directly from the response data by averaging the $x$ and $y$ axis data for the final 1 second of the 2.5 second response record. Error may be introduced here by eye twitches but this is detectable by regularity checks performed during processing.
Settling times are calculated in the above outlined manner for radii of 25, 20, 15, and 10 minutes of arc about the final position.

Regularity checks:

SPROUT performs a number of checks during processing to reduce the possibility of unknowingly including irregular responses as part of normal data. Initial assumptions about the form of a response include initial and final quiet periods of fixation with little or no eye movements. To check for this, SPROUT calculates standard deviations for the initial 0.1 second and final 1.0 second of the 2.5 second response record. Should these exceed set values (5 to 10 minutes of arc, depending on the subject) an error message is printed. SPROUT also generates an error message if the calculated latency is less than 0.1 second. This condition occasionally occurs due to subject inattention and failure to achieve steady target fixation during the warning buzzer time, and an infrequent "predictive" response at the cessation of the warning buzzer before the subject has determined the new target position. Both conditions occur rarely.

In actual use, SPROUT has been used to analyze over 2000 step target deflection responses from four different subjects and the regularity checks seem quite adequate to the task. Approximately 1 to 5 percent of responses must be rejected on the basis of these checks, depending on the subject tested.
SPROUT commands:

Commands to SPROUT may be entered on the teletype keyboard whenever SPROUT is in command mode, indicated by the printing of a "I" at the left margin. Valid commands are F, G, L, P, I, D, A, E, S, and V, and are explained below.

F:  Command F causes a request for user entry of the summary file name. This may be any 6 character alphanumeric string. The filename last entered using the F command (see command G) will be assigned to the saccade summary array when it is output as a data file to disk FL2 (see command E).

G:  Command G causes a request for user entry of the filename of a saccade summary data file already existing on disk FL2. SPROUT then gets the file and inputs it into the summary array in core, informing the user that it has done so. If desired, the summary file may then be altered using other commands and returned to disk storage with the E command. Use of the G command updates any filename previously entered with the F command to the name of the G-obtained file.

L:  Command L lists the original filenames of all saccade data files summarized in the current summary array.
P: Command P causes a request for user entry of the name of a saccade data file, previously processed and summarized in the current summary array. The summarized data for that entry is printed on the teletype or a "not found" message is printed.

I: Command I causes processing of an individual saccade data file and subsequent insertion of the results into the existing summary array. The command initially causes a request for user entry of a saccade data filename; the file named must already exist on disk FL2. SPROUT then inputs and processes the file, appends the results to the summary array, and returns to command mode. Note that if the requested saccade data file does not exist on FL2, a fatal error return to monitor (OS/8) will occur and the summary array in core will be lost.

D: Command D allows deletion of a saccade data entry from the summary array. The command causes an initial request for the filename of the entry, then searches the summary array for the entry with the user entered name, and deletes the entry, if found.

A: Command A allows automatic processing of sets of saccade data files. The 6-character filenames of files in the set must have identical first four characters and the last two characters, which are two-digit numbers, must be sequential with no missing numbers between highest and lowest. Command A causes an initial request for starting and ending filenames. As an example, "HKJX04" and
"HKJX27" might be valid entries. SPROUT would then input the 24 individual saccade data files, in sequence from HKJX04 to HKJX27, processing each and appending the data into the summary array. When finished with all files requested, SPROUT returns to command mode. Processing time for sequences of files is approximately 30 seconds for each file when each file has 500 samples per channel (1000 total).

S: Command 3 requests user entry of the scale factor, in volts/degree, to be used in processing saccade data files. The entered number is stored as variable VDEG, which is normally initialized to 0.1 volt/deg, the scale used in all work here.

V: Command V causes a request for user entry of the cutoff velocity in degrees/second to be used in determining latency. The entered number is stored as variable SVEL, which is normally initialized to 20, degrees/second, the value used in all work here.

E: Command E causes the following SPROUT summary file output sequence to occur: (1) the existing saccade summary array is compressed, eliminating any spaces resulting from entry deletions, (2) that part of the array with data is then output to disk FL2 under the existing filename (see F and G commands), (3) the summary array is then cleared and SPROUT returns to command mode.
SPROUT saccade summary array format:

After processing a saccade data file, SPROUT inserts the latency, the four settling times, the name of the saccade data file, the target deflection angle and radius for the trial, and the data file sampling parameters into the saccade summary array, IDATA(i024). The foregoing information comprises one entry in the summary array.

Up to 63 entries may be included in the summary array, and it is expected that the summary array eventually will be output and stored as a saccade summary file on disk FL2. Each entry in the array is a 16 word block (64 blocks total) and the first 16 word block of the array contains the filename of the summary array itself and a count of the entries in the array. This header block is structured as follows: words 1, 2, and 3 (the first 3 words of the array) contain the array filename, coded in A6 format and split up, two characters per array word. Word 4 has an integer number from 0 to 63, the count of the entries in the array. Note that this count is updated whenever the I, A, or D commands are used. Words 5 through 16 are not currently used.

Saccade data file entries are stored in ascending 16 word blocks after the header block, and are ordered in the sequence entered by the processing commands I or A. One saccade data block is structured as follows: words 1 through 3 of the block contain the name of the
original saccade data file, coded in A6 format and split, two characters per word. Word 4 contains the number of samples per channel in the original saccade data file, word 5 contains the sample time used to take the original data, coded as a timebase and timebase multiplier, word 6 contains the radius and angle of the target deflection used in the original trial, word 7 contains the saccade latency in units of sample time periods, and words 8 through 11 contain the settling times to 25, 20, 15, and 10 minutes of arc, respectively, in units of sample time periods. Words 12 through 16 are not presently used.
SPROUT program operation:

Due to computer core size limitations, all of SPROUT cannot reside in core at one time. SPROUT is thus comprised of three separate FORTRAN II/SABR programs which are swapped into core (chaining) as necessary: SPROUT, SACERO, and SACUSR with its subroutines SDCDE, FCOMP, and FNCHK. The user initially calls SPROUT, which does housekeeping, prints an opening message, and chains to SACUSR. SPROUT is not used after initial startup. SACUSR accepts and interprets user entered teletype commands and carries out manipulations on the saccade summary array. When commands A or I are given, SACUSR chains to SACERO which inputs and processes saccade data files, appends the results into the summary array, then chains back to SACUSR.

Major variables are "passed" among subroutines and chained programs SPROUT, SACUSR and SACERO via FORTRAN common storage, which remains intact during chaining. Variables are:

ICHNS - number of sampled channels of data in the currently processed saccade data file. Should = 2, always.

ICHN1 - starting channel of sampled data in the currently processed saccade data file. Should = 0, always.

IMSEC - sample period timebase of a saccade data file. 1, 2, or 5 (msec).

ITBM - sample period timebase multiplier of a saccade data file. An integer from 0 to 4.
ISAMP - number of samples per channel of a saccade data file.

IRAD - decoded radius number for target deflection, stored in a saccade data file header or a summary array entry. An integer from 1 to 4.

IANG - decoded angle of target deflection, stored in a saccade data file header or a summary array entry.

ICHID - integer equivalent of an ASCII keyboard command entry.

SIFLO - starting saccade data file name (see command A).

SIFHI - ending saccade data file name (see command A).

SOFLE - summary array output filename.

TSFN - temporary filename storage for utility purposes.

STIME - sample period in seconds for a saccade data file or summary array entry.

SLTCY - calculated saccade latency, in seconds.

ST25, ST20, ST15, ST10 - calculated response settling times to 25, 20, 15, and 10 minutes of arc.

VDEG - scale factor in volts/deg used in saccade data calculations.

SVEL - cutoff velocity in degrees/second used to calculate latency.

IDATA(1024) - integer array, used for the saccade summary array.

Operation of the programs SPROUT, SACUSR, and SACPRO are outlined on the following pages, followed by a listing of all three programs and all subroutines.
SPROUT:

SPROUT is an opening program used only once at startup. SPROUT clears the array IDATA(1024), prints an opening message, requests a summary output filename (as an error precaution), then chains to SACUSR.

SACUSR:

SACUSR interprets user entered keyboard commands and performs appropriate actions to carry out the commands. Summary file manipulation is done with all commands except I and A and these manipulations are carried out by SACUSR. Commands A and I require that saccade data file(s) be input, processed, and the results appended to the summary array; SACUSR chains to SACPRO for these functions.

Entry to SACUSR, either by a SACPRO or a SPROUT chain, causes initializations to be done (VDEG, SVEL) and SACUSR then enters command mode. User entry of an invalid command results in an error message and a return to command mode. Valid commands are executed as outlined below, followed by a return to command mode.

Command F initiates a request for user entry of the summary output filename, SOFLE. When command "E" is given to output the summary array, the disk data file will be given the name stored as SOFLE.

Command G causes an initial request for the filename of an existing saccade summary file stored on FL2. The entered name is
stored as variable SOFLE. The file requested must exist on FL2 or a fatal error return to OS/8 monitor results. SACUSR reads the first four words of the requested summary file, and checks the first three of these against the entered filename using subroutine FCOMP. An error message and return to command mode results if a match cannot be made. If the check is successful, SACUSR reads the entire data file into array IDATA(1024), then returns to command mode.

Command L causes a search of the summary array to be performed. Each 16 word block is checked and if the block contains saccade data the saccade file name for the entry is decoded (SDCDE) and printed. When all entries have been listed, SACUSR returns to command mode.

Command P causes a SACUSR request for user entry of a saccade data filename whose data is included in the summary array. A search and comparison (subroutines SDCDE, FCOMP) is made of the array entries and all summarized data for an entry is printed if a match is made. If the requested entry is not found, a message to that effect is printed; SACUSR returns to command mode when finished.

Command D causes SACUSR to request user entry of a saccade data filename whose data is included in the summary array. A search for the entry is made and if found, the entire data block is deleted (zeroed) and SACUSR decrements the summary file entry count. An error message results if the search fails. SACUSR returns to command mode when finished.

Command S causes a request for user entry of the data scale factor in volts/degree, stored as VDEG, and used in processing saccade data
files. VDEG is assigned a default value on entry into SACUSR.

Command V causes a request for user entry of the cutoff velocity (SVEL) in degrees/second, to be used in calculating saccade latencies. SVEL is assigned a default value on entry into SACUSR.

Command E is used to output the summary array to disk FL2 under whatever filename is stored at SOFLE. A set of nested DO loops first compresses the summary array to eliminate imbedded zero data blocks. The current value of SOFLE is then encoded and stored as the first three summary array words. SACUSR then outputs to FL2 as much of IDATA(1024) as is necessary, prints a message to that effect, then returns to command mode.

Command I causes a SACUSR request for a saccade data filename and the entered name is stored as SIFLO. SIFLO and SIFHI are then equated and subroutine FNCHK is called to check the form of the entered name. If the check is successful SACUSR chains to SACPRO for saccade data file input and processing. SACPRO chains back to SACUSR when finished with the specified file. The requested file must exist on disk FL2 or a fatal error return to OS/8 keyboard monitor will occur and any existing summary array will be lost.

Command A causes a request for user entry of a staring saccade data filename, stored as SIFLO, and an ending saccade data filename, stored as SIFHI. FNCHK is called to check the form of the filename and if the check is successful SACUSR chains to SACPRO. SACPRO will input and process all files sequentially, then chain back to SACUSR. All files must exist on disk FL2 as discussed with command A, or a
fatal error return to OS/8 keyboard monitor will result, causing loss of the existing summary array.

SDCDE(K):

This subroutine is called by SACUSR as necessary to decode summary information in one data block within the summary array. One argument K is passed into SDCDE; K is the starting location in the summary array for the data block to be decoded. With K as a pointer, then, SDCDE assembles the entry filename at TSFN, then unpacks and assigns values to ISAMP, ITBM, IMSEC, IANG, and IRAD. Sample period, latency, and settling times are then calculated in seconds and assigned to variables STIME, SLTCY, ST25, ST20, ST15, and ST10. SDCDE then returns.

FCOMP(A,B,I):

Subroutine FCOMP compares two passed filenames A and B. If the names match, FCOMP returns with argument I = 0, otherwise the return is with I = 1. ASCII filenames stored as FORTRAN real variables cannot be compared by simple subtraction with this operating system, hence the effective bit-by-bit comparison carried out by FCOMP.

FNCHK(I):

Subroutine FNCHK is called by SACUSR and checks whatever filename may be stored at TSFN to insure that the last two characters are ASCII coded numbers. Return is with I = 0 if this is so, I = 1 if not.
SACPRO:

SACUSR chains to SACPRO for processing of individual or sequential sets of saccade data files. SACPRO and SACUSR pass information back and forth by means of FORTRAN common storage locations which are undisturbed by program chaining operations.

SACPRO operation begins by extracting the two digit number from the ends of the upper and lower filenames SIFHI and SIFLO. These will be used later in the program. SACPRO then inputs the saccade data file SIFLO from FL2 into the array ITEMP(1024) and unpacks and decodes the sampling parameter information coded in the first three words of the array (see ADISOI programming). SACPRO then sets up a number of parameters (based on the data sampling parameters) in preparation for calculations of x and y channel (multiplexer channels 0 and 1) averages and standard deviations. These are calculated for the first 0.1 second of the data and for the final 1.0 second of the data.

X and Y averages and standard deviations are calculated using two SAER coded calls to an internal SAER/FORTRAN II subroutine. Due to core size restrictions these results are left in units of A/D converter counts, which may be scaled to voltage by dividing by 512. Results of these calculations are used later for data regularity checks and settling time calculations.

SACPRO then proceeds to calculate the latency, leaving the results in units of sample time periods, by computing the velocity at each x-y sample point of the data and comparing this against the cutoff
velocity. SACPRO starts at the beginning of the array and proceeds

to the end of the data or until the cutoff is exceeded, whichever

comes first.

SACPRO then calculates the settling times to 25, 20, 15, and 10

minutes of arc about the final x-y position, previously calculated

as the x and y averages for the final 1 second of data; settling
time results are left in units of sample time periods. Settling
times are calculated by first computing the distance from each x-y
sample pair to the final position point, beginning at the start of the
data array. Settling time for a given radius is defined as the
beginning of the first 0.1 second period (or longer) during which the
distance to the final position is within the given radius.

SACPRO then performs regularity checks. Standard deviations

of x and y for the first 0.1 and final 1.0 second of data are compared
against a maximum limit, and the calculated latency is checked to
see that it is less than 0.1 second. If any standard deviation is too
large or the latency too small, SACPRO indicates this by printing
out all calculated information: x and y initial and final averages
and standard deviations and the latency and settling times. These
numbers appear in units of A/D converter counts and numbers of sample
periods.

Regardless of regularity checks results SACPRO then searches
the summary array for the first empty data block and inserts the
following information into the block: the saccade data filename, the
saccade data file sampling parameters and target deflection angle and
radius information, and the latency and the four settling times (in units of sample periods). SACPRO then increments the summary array entry count. Should the user decide to reject this processed data based on the regularity checks failure and printout, the "D" command may be used when SACPRO chains back to SACUSR.

SACPRO then checks to see whether SIFLO = SIFHI and whether the SIFLO two digit filenumber = 99. If either is true, SACPRO chains back to SACUSR. If neither is true, SACPRO increments the SIFLO number, then transfers control back to the recycle point near the beginning of SACPRO. SACPRO begins the processing cycle again by inputting the saccade data file with the new SIFLO filename.

This completes the discussion of SFRCUT and its component programs; a complete program listing is included on the following pages.
SPROUT ZER0ES IDATA(1324), POINTS OPENING
MESSATE, REQUESTS SUMMARY FILENAME,CHAINS
TO SACUSK.

COMMON IG1M1, IG1M2, I1S3, IT01, IT1A, IT20, I2AS
COMMON I24, I25, I26
COMMON SAFL0, SAFHL, S0FLE, TSF1, ST1M, SLTCY
COMMON ST25, ST26, ST15, ST16, VDES, VVEL
COMMON IDATA

DIMENSION IDATA(1324)
DO 13 K=1,1324
13 IDATA(K)=3
WRITE(1,21)
21 FORMAT('SPROUT CHAIN: F,G,L,P,L,D,A,L,S,V')
READ(1,31)S0FLE
31 FORMAT('SUMMARY FILENAME FL2: 'A6)

CALL CHAIN('SACUSK')
STOP
END
SACCU3 PROCESSES INPUT CONTENTS FOR SACCARDE

SUMMARY FILE MANIPULATION AND SACCARDE FILE

PROCESSING. WHEN I or A IS GIVEN,
SACCU3 CHAINS TO SACP50 FOR SACCARDE FILE
DATA PROCESSING AND INSERTION OF RESULTS
IN THE SUMMARY FILE. IDATA(1324) CONTAINS
THE SACCARDE SUMMARY FILE.

COMMON ICH3S, ICHHL, IBS5, IT84, ISASP, IFRAD
COMMON IAML, ICMP
COMMON SIFLD, SIFHL, SIFLE, ISFL, STIM, SLTSCY
COMMON ST25, ST26, ST15, ST13, VDEG, SVEL
COMMON IDATA

ORDER 55 73 2

DIMENSION IDATA(1324)

ICF=416
ICG=433
ICL=999
ICP=1056
ICI=688
ICD=233
ICAL=96
ICG=352
ICS=1248
ICV=1448

VDEG=0.1
SVEL=29.0

READ(1,21)ICMID

FORMAT('! ',AI)

IF(ICMID-ICF)51,133,51
51 IF(ICMID-ICG)52,150,52
52 IF(ICMID-ICL)53,273,53
53 IF(ICAL-ICP)54,253,54
54 IF(ICMID-ICAL)55,343,55
55 IF(ICAL-ICG)56,353,56
56 IF(ICAL-ICAL)57,433,57
57 IF(ICAL-ICMID)58,453,58
58 IF(ICAL-ICAL)59,553,59
59 IF(ICAL-ICAL)60,573,60
60 WRITE(1,71)

71
C

71 FORMAT('BAD CMD')
GO TO 28

C

104 READ(1, 131) SOFLE
131 FORMAT('SUMMARY FILENAME FL2: 'A6')
GO TO 28

C

C'MD S: GET AN EXISTING SUMMARY FILE FROM FL2.

158 READ(1, 151) SOFLE
151 FORMAT('GET SUMMARY FILE FL2: 'A6')
CALL 10PE1('FL2', SOFLE)
READ(4, 161) IH, IM, IL, I16
161 FORMAT(12A2)

C

C CHECK FOR FILENAME AT START OF FILE.

5 CALL
5 SCALL 3, CLEAR /CLEAR F.P.AG.
5 TAD NIL
5 DCA ACM /F.P.AG HIGH WORD.
5 TAD NIL
5 DCA ACM /F.P.AG MIDDLE WORD.
5 TAD NIL
5 DCA ACL /F.P.AG LOW WORD.
5 SCALL 1, STO /F.P.AG TO TSF1.
5 AT3 CALL NTSFN

172 CALL FCOMP(TSF1, SOFLE, IANS)
175 WRITE(1, 171)
171 FORMAT('BAD SUMMARY FILE')
GO TO 28

175 LW=(ISUM+1)*16
CALL 10PE1('FL2', SOFLE)
READ(4, 161) (IDATA(K), K=1, LW)
GO TO 28

C

C'MD L: LIST ALL SACCAD E FILENAMES IN SUMMARY ARRAY.

234 WRITE(1, 281) SOFLE, IDATA(4)
281 FORMAT('SUMMARY FILENAME FL2: 'A6, 15
C ' SACCADExx

DO 233 K=17,1309,16
IF(IDATA(K))295,230,295
233 CALL SDGDECK(K)
WRITE(1,211)TSFN
211 FORMAT(A6)
230 CONTINUE
GO TO 20

C CMD P: PRINT SUMMARIZED DATA FOR THE
C SPECIFIED "ORIGINAL SACCADExx FILE".

250 READ(1,251)TSFN
251 FORMAT('05F:','A6')
DO 270 K=17,1309,16
CALL SDGDECK(K)
CALL FCMP(TSFM,TSFN,SIFLO,IA13)
IF(IA13)272,263,270
260 WRITE(1,251)TSFN,IA13,P15*19**IT31,
C FLOAT(IGAO)*.25,IA14,SLTGY,ST25,ST29,
C ST15,ST10
261 FORMAT('05F:','A6','S/CH','16' MSEC ',RAD='
C F4.2 ' ANG='14' DEG','/LTGY='F7.3' SEC'/
C 'ST25='F7.3' SEC'/ST29='F7.3' SEC'/ST15='
C F7.3' SEC'/ST10='F7.3' SEC')
GO TO 20
270 CONTINUE
GO TO 330

C CMD I: PROCESS THE SPEC'D SACCADExx FILE
C AND INCLUDE IT SUMMARY.

333 READ(1,301)SIFLO
301 FORMAT('FL2:','A6')
SIFLO=SIFLO
TSFN=SIFLO
CALL FICMUK(13AD)
IF(13AD)22,310,23
310 CALL CHAIN('SACCPR0')

C CMD D: DELETE THE SPEC'D SACCADExx FROM
C       THE SUMMARY FILE.

350      READ(1,351)SSFN
351      FORMAT('SOF: 'A6)

   DO 375 K=17,1339,16
   CALL SDCDSCM
   CALL FCMPCT(SFNM,SSFM,IAIS)
   IF(IAIS)375,362,375

362      K15=K+15
   DO 365 J=K,K15
   IDATA(J)=3
   WRITE(1,371)TSFN
   FORMAT('A6,' SUMMARY NOT FOUND'
   IDATA(4)=IDATA(4)-1
   GO TO 20
375      CONTINUE
   WRITE(1,381)SSFN
   FORMAT('A6,' SUMMARY DELETED'
   GO TO 23

C       CMD A: PROCESS ALL SACCADE FILES FROM
C       "SIFLO" TO "SIFHI" INCLUSIVE AND INSERT
C       DATA INTO SUMMARY FILE.

403      READ(1,431)SIFLO
431      FORMAT('START FL2: 'A6)
   TSFN=SIFLO
   CALL FICMK(IBAD)
   IF(IBAD)405,406
405      READ(1,406)SIFHI
406      FORMAT('END FL2: 'A6)
   TSFN=SIFHI
   CALL FICMK(IBAD)
   IF(IBAD)415,20
415      CALL CHAIK('SACPRO')

C       CMD B: COMPRESS SUMMARY DATA, INSERT
C FILENAME "SOFLE", OUTPUT FILE TO DISK
C (FL2:SOFLE), ZERO SUMMARY ARRAY.

433      DO 472 K=17,1339,16
   IF(IDATA(K))433,455,434
455      DO 472 L=K,1339,16
IF(IDATA(L)) 462, 473, 483

DO 465 I = J, 15
IDATA(J+1) = IDATA(L+1)
IDATA(L+1) = 0

465 CONTINUE
GO TO 463

473 CONTINUE

483 CONTINUE

S CALL 2 CLEAR /CLEAR F.P.AC.
SCALL 1, FAD /SOFLE TO F.P.AC.
SARG /SOFLE
S TAD AGH /SOFLE TO FIRST 3 FILEWORDS.
S DCA N1H
S TAD AGM
S DCA N1I
S TAD AGL
S DCA N1L
IDATA(1) = 1H
IDATA(2) = 1I
IDATA(3) = 1L

LM = (IDATA(4) + 1) * 16
CALL OPEMC('FL2', S0FLE)
WRITE(4, 491) (IDATA(K), K = 1, LM)

491 FORMAT (12A2)
CALL CLOSE
WRITE(1, 495) S0FLE, IDATA(4)

496 FORMAT ('SARCADE SUMMARY FILE FL2: 'A6', '
C 15' ENTRIES, FILED.'//)
DO 493 K = 1, 1224

493 IDATA(K) = 0
GO TO 20

C CMS S: ENTER VOLTS/DEG TO BE USED FOR
C SETTLING TIME COMPUTATION.

550 READ(1, 551) VDEG
551 FORMAT ('ENTER VOLTS/DEG: 'F6.4)
GO TO 20

C CMD V: ENTER DEG/SEC CUTOFF FOR LATENCY.

575 READ(1, 576) SVEL
576 FORMAT ('ENTER DEG/SEC LTGY CUTOFF: 'F4.1)
GO TO 26

END
SACPRO IS CHAINED TO BY SACUSR. SACPRO PROCESSES INDIVIDUAL SACCADE DATA FILES AND INSERTS THE DATA IN THE SUMMARY FILE. VARIOUS ERROR CHECKS PERFORMED ALONG THE WAY; DATA IS PRINTED IF CHECKS FAIL AND PROCESSING CONTINUES. SACPRO CHAINS BACK TO SACUSR WHEN FINISHED.

COMMON ICHNL,ICH1L,INSEG,IT31,ISAMP,IRAD COMMON I4MG,ICMD COMMON SIFL0,SIFNL,SIFEL,TSFL,STME,STLCY COMMON ST25,ST20,ST15,ST10,VSEG,SVEL COMMON IDATA

GET FILENAMES FROM FILENAMES.

ILH=0
IL=0
I'M=0
I'H=0

SCALL 2,CLEAR
SCALL 1,FAD
SARG \SIFL0
S TAD ACH
3 DCA NM
S TAD ACM
S DCA NM
3 TAD ACL
S DCA NIL

SCALL 2,CLEAR
SCALL 1,FAD
SARG \SIFNL
S TAD ACL
S DCA NIL

GET SACCADE FILE AND CODEWORDS.

CALL IOPZK('FL2',SIFL0) READ(4,5)ITEM
FORMAT(12A2)
ICHM3=ITEMP(1)
ICHM1=ICHM3
LSM1-ITEMP(2)
IT3=ITEMP(3)
INSEC=IT3

CLA /DECODE SACCADE FILE CODE WORDS.
TAD \ICHNS # CHANNELS.
BSV
AND (77
DCA \ICHNS
TAD \ICH31 /START CHANNEL.
AND (77
DCA \ICH11
TAD \ITB1 /TIMEBASE MULTIPLIER.
BSV
AND (77
DCA \ITB1
TAD \INSEC /TIMEBASE.
AND (77
DCA \INSEC

SACCADE DATA CALCULATIONS.
IMSEC=INSEC*10**ITB1
STIME=FLOA(NMSEC)*.001
MS3=133/MSEC
MF3=133/MSEC
SDIV=FLOA(MS3+1)
FDIV=FLOA(MF3+1)

X,Y AVERAGES AND STANDARD DEVIATIONS
FOR FIRST 3.1 SEC AND FINAL 1.9 SEC.
M1=6
M2=2*MS3+6
DIV=SDIV
J=5 CALC
XSAVG=XAVG
YSAVG=YAVG
X3DEV=XDEV
Y3DEV=YDEV

2=15A1P*2+4
M1=M2-2*MFS
DIV=DIV
J=5 CALC
XFAVG=XAVG
YFAVG=YAVG
XFDEV=XDEV
YFDEV=YDEV
GO TO 133

S CALC,

XAVG=3.
YAVG=3.
XDEV=2.
YDEV=0.

DO 140 KX=11,12,2
140 YAVG=YAVG+FLOAT(ITEMP(KX))
XAVG=XAVG/DIV
YAVG=YAVG/DIV

DO 160 KY=11,12,2
160 XDEV=XDEV+(FLOAT(ITEMP(KX))-XAVG)**2
YDEV=YDEV+(FLOAT(ITEMP(KY))-YAVG)**2
XDEV=SQRT(XDEV/DIV)
YDEV=SQRT(YDEV/DIV)

S JMP I CALC

C LATENCY CALCULATION.

C COUNTS/SAMPLE CUTOFF=DEG/SEC CUTOFF *
C VOLTS/DEG * COUNTS/VOLT * SEC3/SAMPLE:
C V0LT3/DEG * COUNTS/SEC/VOLT * SEC3/SAMPLE:
133 CSC0=SVEL*VDEG*512.*3TIME

ILTCY=0

DO 230 KY2=8,12,2
KY1=KY2-2
KX2=KY2-1
KX1=KX2-2
TCMT=SQRT((FLOAT(ITEMP(KX2)-ITEMP(KX1)))**2+
C (FLOAT(ITEMP(KY2)-ITEMP(KY1)))**2)
IF(CSC0-TCMT)>220, 223, 200
230 ILTCY=(KX1-3)/2

C SETTLING TIMES CALCULATIONS.

220 C:V1=VDEG*512.*60.
D1JM=13.*C"11'
D15M=15.*C"11'
D20M=24.*C"11'
D25'=25.*C"11'

C NUMBER OF SAMPLES FOR .1 SEC SETTLING
C TIME CRITERIA:
IT=130/4.1SEC

IT10=3
IT15=0
IT20=0
IT25=3

DO 250 KY=1,12,2
KX=KY-1
TD=3.1RT((FLOAT(IT20P(KY))-XFAVG)**2+ (FLOAT(IT20P(KY))-XFAVG)**2)
IF(TD-D25M)26A,26A,26A
262 IT25=(KY-4)/2
GO TO 270
264 IF((KY-4)/2-IT25-IT)270,256,256
266 D25'=18830.

270 IF(TD-D20M)274,274,272
272 IT20=(KY-4)/2
GO TO 280
274 IF((KY-4)/2-IT20-IT)283,276,276
276 D20'=18830.

233 IF(TD-D15M)234,234,232
232 IT15=(KY-4)/2
GO TO 290
234 IF((KY-4)/2-IT15-IT)293,236,236
236 D15'=10000.

299 IF(TD-D12M)294,294,292
292 IT12=(KY-4)/2
GO TO 250
294 IF((KY-4)/2-IT12-IT)258,296,296
296 D10'=18830.

253 CONTINUE

C DONE WITH CALCULATIONS, CHECK FOR SMALL
C STANDARD DEV'S, LTY<.1 SEC.
C
MAX ALLOWED SD=MAX SD(VOLTS)*COUNT/VOLT.
SD MAX=.013*512.
IF(1LT6-155)310,311,341
321 IF(YSDEV-SDMAX)332,313,314
322 IF(YSDEV-SDMAX)333,315,316
333 IF(XFDEV-SDMAX)334,317,318
334 IF(IFDEV-SDMAX)329,319,320
313 WRITE((1,311))5IFLO,ASAV,YSAVG,YSAVG,YSDEV,
C YSDEV,YFAVG,YFAVG,XFDEV,IFDEV,1LTCY,IT25,
C IT23,IT15,IT10
311 FORMAT('FILE FL2: A6,F7.3 SEC/
C 2('F7.2','F7.2') '<F7.2','F7.2'> ')S(15/))

C INSERT DATA IN FIRST EMPTY BLOCK IF
C SUMMARY ARRAY.
323 DO 343 K=17,149,16
IF(IDATA(K))340,325,340
325 IDATA(K)=11
IDATA(K+1)=11
IDATA(K+2)=IL
IDATA(K+3)=ITEMP(2)
IDATA(K+4)=ITEMP(3)
IDATA(K+5)=ITEMP(4)
IDATA(K+6)=I1TCY
IDATA(K+7)=IT25
IDATA(K+8)=IT23
IDATA(K+9)=IT15
IDATA(K+10)=IT10
IDATA(4)=IDATA(4)+1
GO TO 350
343 CONTINUE
C DONE WITH FILES OR FILENUM=99?
350 IF(IL-ILH)=351,539,351
351 IF(IL+391)360,533,360
360 ILA=3
S TAD \IL /INCREMENT FILENUMBER.
S TAD (7) /UNITS+7, INCR TENS IF REM.'L.
S DCA \IL /SAVE NEW TENS.
S TAD \IL
S AND (77) /UNITS+7.
S SNA
S TAD (66) /FIXUP FOR ZERO.
S TAD (-6) /UNITS+7-6=UNITS+1.
S DCA ILA /SAVE NEW UNITS.
S TAD \IL /PUT NEW UNITS WITH TENS.
S AND (77)00
S TAD \ILA
S DCA \IL
SCALL 3, CLEAR
SCALL 1, FAD
SARG \SIFLO
S TAD \IL
S DCA ACL
SCALL 1, STO
SARG \SIFLO
GO TO 40

500 CALL CHAIN('SAGUSR')
STOP
END
**SSPLOT: Saccade summary file plotting program**

**Design considerations:**

SSPLOT is a plotting program specifically designed to generate pointwise plots of saccade data as summarized in a saccade summary file (see SPROUT discussion). Design considerations were that:

1. SSPLOT be easily used by non-technical personnel,
2. summary file data should be plottable as functions of either target deflection angle or radius,
3. both axes of plots should have variable scales, and
4. parameterization should be available.

To meet the above requirements SSPLOT has the following features:

1. all user interaction is by means of teletype keyboard commands which generate requests for specific information, when necessary,
2. plotter x-axis variable may be either degrees or radius number; plotter y-axis is always time,
3. any of latency or the settling times may be plotted,
4. five different plotter symbols are available, and
5. the user may select a parameter value for radius or angle, whichever is not the x-axis variable, and only summary file entries with that particular target deflection parameter will be plotted.
SSPLOT commands:

Commands may be entered on the teletype keyboard whenever SSPLOT is in command mode, indicated by the printing of a "=" at the left margin. Commands are FN, XD, XR, YS, DA, PA, SY, PP, and GO and are explained below.

FN: Command FN causes a request for user entry of the filename of a saccade summary data file, as previously generated by SPROUT and currently stored on disk FL2. SSPLOT inputs the file and informs the user that it has done so by printing out the filename stored at the beginning of the saccade summary file, then printing the count of saccades included in the summary file.

XD: Command XD sets the x-axis variable to degrees and requests user entry of the x-axis scale factor in degrees/inch. Use of XD also sets the parameterization variable to radius number (0.25, 0.50, 0.75, or 1.00 fractions of the maximum radius of target deflection used). Note that angle and radius of target deflection are stored with each entry in the saccade summary array.

XR: Command XR set the x-axis variable to radius number, 0.25, 0.50, 0.75, or 1.00 fractions of the maximum saccade target deflector radius, and requests user entry of the x-axis scale in "full scale inches for R = 1.00". Use of XR also sets the parameterization variable to angle (0 to 345 in 15 degree increments).
YS: Command YS causes a request for user entry of the plotter y-axis scale factor in milliseconds per inch.

DA: Command DA causes a request for user entry of the data to be plotted. Choices are latency (LTOY) and settling times to 25, 20, 15, and 10 minutes of arc (ST25, ST20, ST15, ST10).

PA: Command PA requests user entry of a parameter, either radius number, if the x-axis variable is degrees (command XD), or degrees, if the x-axis variable is radius number (command XR). When an actual plot is done, only those entries in the summary file with matching parameter will be plotted. This feature is disabled by entering a negative number.

SY: Command SY requests user specification of the symbol to be used by the plotter. Available symbols are squares (S), crosses (C), diamonds (D), X's (X), and triangles (T). All symbols have maximum dimensions of 0.06".

PP: Command PP causes all current plotting parameters to be listed: summary filename, x-axis variable and scale, y-axis variable and scale, data specified to be plotted, parameter number, and symbol code.

GO: Command GO causes the plotter to generate a plot according to previously specified parameters. When the plotter has finished plotting the individual data points, axes are drawn in one inch increments to extend just past the maximum plot excursions and
SSPLOT program operation:

SSPLOT is comprised of a main FORTRAN II/SABR program SSPLOT and one FORTRAN II/SABR subroutine PLTLN. A discussion of PLTLN may be found in the section on APLOTO programming. Briefly, PLTLN(KP,KX,KY) drives the digital plotter in such a way as to plot the best straight line between its present pen position and the new co-ordinates KX and KY. KP controls origin initialization and pen up/down functions.

At startup, SSPLOT prints an opening message on the teletype and performs a variety of housekeeping functions and initializations. The x-axis variable is set for degrees with a scale factor of 60 degrees/inch, the y axis scale is set at 200 msec/inch, latency is declared as the data to be plotted, the parameterization feature is disabled, and squares are chosen as the plotter symbol. SSPLOT then enters command mode. Valid commands are FN, XD, XR, YS, DA, PA, SY, PP, and GO and SSPLOT returns to command mode after performing each commanded function.

Command FN causes a request for user entry of a saccade summary filename. After the name has been entered, SSPLOT reads in the first four words of the file from disk FL2. The first three words will contain the filename if the file is really a saccade summary file and
fourth word will contain the number of entries in the summary. SSPLLOT prints this information, then inputs the entire data file into array IDATA(1024) and returns to command mode.

Commands XD and XR set the x-axis variable to degrees or radius number, respectively, and request user entry of the desired scale factor in units of degrees/inch or full scale inches for R = 1.00. A switch variable, IXV, is set to 0 for XD, 1 for XR, and is used elsewhere in the program.

Command YS causes a request for user entry of the y-axis scale factor in milliseconds/inch.

Command DA requests user specification of the data to be plotted, LTCY, ST25, ST20, ST15, or ST10. An index variable IDEX is set in response to a valid entry or a "data not available" message is printed in response to an invalid entry. IDEX is used elsewhere in the program.

Command PA causes a check to be made of the switch variable IXV. If IXV = 0, a request for user entry of radius number as a parameter is issued; if IXV = 1 SSPLLOT requests user entry of a parameter in degrees.

Command SY causes a request for user specification of the plotter symbol and stores the entered response as variable INSYM to be used elsewhere in the program. Invalid symbol entries will eventually cause a default to the square as the symbol used.

Command GO causes the plotter to execute the plot as specified by the user using other commands. SSPLLOT first initializes the plotter
(call PLTN(-1,0,0)) and zeroes the variables IXMAX and IYMAX. These variables are used to store the maximum plotter excursions for later use in drawing axes. SSPLOT then enters a summary array search/plot loop, wherein each summary entry is dealt with, in sequence, as follows: (1) the sampling parameter information, along with target deflection angle and radius data, is extracted from the entry, (2) the x co-ordinate is calculated based on the x-axis variable (IXV) and satisfaction of a parameter, if specified, (3) the y-axis co-ordinate is calculated based on the y-axis scale factor and the requested data (IDX), (4) maximum x and y co-ordinates are retained, and (5) the pen is moved to the calculated co-ordinates and the specified symbol is drawn by means of an eight-pass DO loop and symbol array coding.

The search/plot loop recycles until all summary array entries have been examined. If a parameter requirement is not met (step 2 above), SSPLOT skips the entry and proceeds to the next. The drawing of a symbol is done by successive short pen movements based on coded entries in the array ISYMB(40). After all entries in the summary array have been examined and plotted, SSPLOT draws x and y axes in one inch increments extending just past the maximum limits of the plot, then returns to command mode.

All plotter symbols are drawn as an eight step sequence of short pen movements, some of which may be dummy moves, depending on the symbol. Moves and pen instructions are coded in eight word blocks in array ISYMB(40). ISYMB(40) entries were inserted by
using the OS/8 debugging program ODT after SSPLOT was compiled, assembled, and loaded into core. SSPLOT was then saved, along with the array ISYMB(40), by use of the OS/8 SAVE command.

This completes the discussion of SSPLOT and a program listing is included on the following pages.
C PROGRAM SSPLT PLOTS DATA FROM SAACADE
C SUMMARY DATA FILES.

5 OPDEF BSW 7002

DIMENSION IDATA(1024)
DIMENSION ISY(40)

WRITE(1,6)
6 FORMAT('SSPLT CMD ARE: FN, XD, XR, YS, DA,' 'PA, SY, PP, GO.')

ICFN=393
ICXD=1543
ICXR=1554
ICYS=1619
ICDA=257
ICPA=1025
ICSY=1241
ICPP=1049
ICGO=463

TSFM=0.
IXV=3
XSFAC=1.6667
YSFAC=0.5
IDEX=6
IPAR=-1
PIPAR=-1.
INSY'=1243
ID1=733
ID2=217
ISYS=1243
ISYC=224
ISYD=288
ISYX=1568
ISYT=1312

C READ AND INTERPRET COMMAND.
20 READ(1,21)ICMD
21 FORMAT('=', 'A2')

IF(ICMD-ICFN)41, 133, 41
41 IF(ICMD-ICXD)42, 159, 42
42 IF(ICMD-ICXR)43, 175, 43
43 IF(IC:ID-ICYS)44,283,44
44 IF(IC:ID-ICDA)45,225,45
45 IF(IC:ID-ICPA)46,380,46
46 IF(IC:ID-ICSY)47,354,47
47 IF(IC:ID-ICPP)48,375,48
48 IF(IC:ID-ICGO)55,530,55
55 WRITE(1,56)
56 FORMAT('3AD CMD')
GO TO 20

C CMD F1: DECLARE SUMMARY FILENAME, GET C FILE FROM FL2:

122 READ(1,121)SSFN
121 FORMAT('ENTER SACCADE SUMMARY FILENAME FL2: 'A6')

CALL IOPEN('FL2',SSFN)
READ(4,106)II,II,IL,ISNUM
106 FORMAT(12A2)

5 CLA CLL
SCALL 0,CLEAR /CLEAR F,P,AC.
5 TAD \IH /PUT FILENAME AT TSFN.
S DCA ACM
S TAD \IM
S DCA ACM
S TAD \IL
S DCA ACL
SCALL 1,STO
SARG \TSFN
WRITE(1,116)TSFN,ISNUM
116 FORMAT('SACCADE SUMMARY FILE FL2: 'A6,15 '
 'SACCADERS'/)

LW=(ISNUM+1)*16
CALL IOPEN('FL2',SSFN)
READ(4,106)(IDATA(K),K=1,LW)
GO TO 20

C CMD XD: X AXIS WILL BE DEGREES. ENTER C SCALE, DEG/INCH.

153 READ(1,151)XSFAC
151 FORMAT('DEG/INCH: 'F5.2)

XSFAC=153./XSFAC
IXV=0
GO TO 20
C  CMD XR:  X AXIS WILL BE RADIUS NUMBER.
C  ENTER FULL SCALE INCHES FOR R=1.20.

175  READ(1,176)XS,FAC
176  FORMAT(\"FULL SCALE (R=1.20) INCHES: \"F6.2\")

XS,FAC=25.*XS,FAC
IXV=1
GO TO 20

C  CMD YS:  ENTER Y AXIS TIME SCALE, 13SEC/INCH.

200  READ(1,201)YS,FAC
201  FORMAT(\"MSEC/INCH: \"F5.3\")

YS,FAC=100./YS,FAC
GO TO 20

C  CMD DA:  DECLARE DATA TO BE PLOTTED.

225  READ(1,226)ID1,ID2
226  FORMAT(\"LCY,ST25,ST20,ST15,OR ST10? ; \"2(A2)\")

IJLT=758
IJST=1236
IJ25=-843
IJ20=-848
IJ15=-907
IJ10=-912

IF(ID1-IJLT)231,250,231
231  IF(ID1-IJST)240,232,240
232  IF(ID2-IJ25)233,251,233
233  IF(ID2-IJ20)234,252,234
234  IF(ID2-IJ15)235,253,235
235  IF(ID2-IJ10)240,254,240

240  WRITE(1,241)
241  FORMAT(\"DATA NOT AVAILABLE\")
GO TO 20

250  IDEX=6
GO TO 22
251  IDEX=7
GO TO 23
252  IDEX=9
Go to 25
253 INDEX=9
Go to 25
254 INDEX=18
Go to 25

C CMD PA: DECLARE PARAMETER FOR DATA. ENTER
C RADIUS NUMBER OR ANGLE FOR PREVIOUS CMD
C XD OR XR. ENTER NEGATIVE NUMBER TO DISABLE.
330 IF(IXV)313,335,310
305 READ(1,336)IPAR
306 FORMAT('WHICH RADIUS? 1.00, 0.75, 0.50, 0.25:
C F4.2)
IPAR=IFIX(4.*IPAR)
Go to 25
313 READ(1,311)IPAR
311 FORMAT('WHICH ANGLE?: F4.0)
IPAR=IFIX(IPAR)
Go to 25

C CMD SY: PLOTTER SYMBOL SPECIFICATION.
350 READ(1,351)INSYM
351 FORMAT('S, C, D, X, OR T: A1)
Go to 25

C CMD PP: PRINT PLOT SPECIFICATIONS.
375 WRITE(1,376)TSFN
376 FORMAT('/FILENAME: A6)
333 WRITE(1,336)XSFAC
331 FORMAT('SCALE: F6.2 DEG/INCH')
Go to 390
385 WRITE(1,381)YSFAC/25.
381 FORMAT('SCALE: F6.2 INCHES FOR R=1.00')
390 WRITE(1,391)YSFAC, ID1, ID2, IPAR, INSYM
391 FORMAT('SCALE: F7.0 MSEC/INCH/'DATA:
C 5X,2(A2)'/PARAM: F8.2/'SYM: A8, A1/
Go to 20

C CMD GO: PLOT ACCORDING TO SPEC'D PARAMETERS.
CALL PLTLN(-1,3,3)

IX\:MAX=0
IY\:MAX=0
DO 730 K=17,LY,16

C GET SAMPLE TIME INFO,RADIUS,ANGLE.

IMSEC=IDATA(K+4)
IT8M=IMSEC
IANG=IDATA(K+5)
IRAD=IANG

S CLA CLL
S TAD \IMSEC  \GET TIMEBASE.
S AND (77
S DCA \IMSEC
S TAD \IT8M \GET TIMEBASE
S BM
S AND (77
S DCA \IT8M
S TAD \IANG \GET ANGLE/15.
S RTR
S AND (37
S DCA \IANG
S TAD \IRAD \GET RADIUS NUMBER.
S AND (3
S IAC
S DCA \IRAD
IANG=IANG*15

C PARAMETER CHECK, PLOTTER STEP CALCULATIONS.

IF(I\:XV)535,531,535

531 IXNUM=IFIX(FLOAT(IANG)*XSFAC)
IF(IRAD-IPAR)543,550,540
535 IXNUM=IFIX(FLOAT(IRAD)*XSFAC)
IF(IANG-IPAR)543,550,540
540 IF(IPAR)550,700,703
550 IYUW=IFIX(YSFAC*FLOAT(IMSEC)*IJ.*IT8M*
C FLOAT(IDATA(K+INDEX)))

C KEEP MAX EXCURSIONS.

IF(I\:NUM-I\:X\:MAX)565,565,561
561 IX\:MAX=IX\:NUM
565 IF(I\:Y\:UW-I\:Y\:MAX)573,573,566
566 IY\:MAX=IY\:NUM
C 'MOVE PEN, DRAW SYMBOL.'

578 CALL PLTLN(I,IXN1,IYV1)
ISY=1
IF(ISYM-1 SYC)585,581,535
581 ISY=9
585 IF(ISYM-1 SYD)590,586,593
586 ISY=17
590 IF(ISYM-1 SYX)595,591,593
591 ISY=25
595 IF(ISYM-1 SYT)631,596,621
596 ISY=33

631 DO 650 JS=0,7
IPEN=ISYM,JS(ISY+JS)
IXSY'=IPEN
IYSYM=IPEN

S CLA CLL
S TAD IIPFN
S RAL
S CLA
S RAL
S DCA IPEN
S TAD IYSYM
S AND C37
S RAR
S SZL
S CIA
S DCA IYSYM
S CLL
S TAD IXSYM
S BSW
S CLL
S AND C37
S RAR
S SZL
S CIA
S DCA IXSYM

IXSYM=IXSYM+IXN1
IYSYM=IYSYM+IYV1
CALL PLTLN(IPEN,IXSYM,IYSYM)

650 CONTINUE

703 CONTINUE

C DRAW AXES AND RETURN TO ORIGIN.
CALL PLTLM(1,0,3)
DO 750 L=0,IXMAX,100
I=L+100
CALL PLTLM(0,1,I)
CALL PLTLM(3,1,10)
CALL PLTLM(0,1,3)
750 CONTINUE

CALL PLTLM(1,0,3)
DO 760 L=0,1YMAX,100
I=L+100
CALL PLTLM(0,3,I)
CALL PLTLM(0,10,I)
CALL PLTLM(0,3,I)
760 CONTINUE

CALL PLTLM(1,0,3)
GO TO 20
END
SUBROUTINE PLTLN(KFG, XX, KY)

S OPDEF PLCF 6502
S OPDEF PLPU 6533
S OPDEF PLLR 6504
S OPDEF PLPD 6535
S SKPDF PLSF 6501
S OPDEF CLAF 6142
S SKPDF SDFS 6143

JFG = KFG
JX = XX
KY = KY

CLA CLL
TAD (4092
TAD \JX
DCA \JX
TAD (4090
TAD \JY
DCA \JY
CLA CLL
TAD \JFG
SPA
JMP PLTAL
TAD PLTPM
CALL RTPR
SPA CLA
JMP PLOT1
CALL CLA
JMP A1
DCA PLTPM
PLPU
JMP A2
A1.
ISZ PLTPM
PLPD

PLTLM PLOTS THE BEST STRAIGHT LINE BETWEEN
THE OLD AND NEW CO-ORDINATES. X CO-ORDINATE
CORRESPONDS TO "XX", Y TO "KY". THESE ARE
STANDARD FORTRAN INTEGER VARIABLES AND MUST
BE >=-2047 AND <=2047. "KFG" CONTROLS
PLOTTING AS SUCH: KFG=-1, INITIALIZE;
KFG=1, PLOT WITH PEN UP; KFG=0, PLOT WITH
PEN DOWN.

S "CLEAR PLTR FLG FF.
S "PEN UP.
S "LOAD DIR REG CLR
S "DIR REG, SET FLG.
S "PEN DOWN.
S "SKP ON PLTR FLG.
S "CLEAR M1703 FLAGS.
S "SKP ON M1703 FLAG.

MOVE PEN?
"NO: INITIALIZE.
"ADD PEN STATUS.
"CONTINUE.
"GO LOWER THE PEN.
"RAISE THE PEN.
"LOWER THE PEN.
S A2, JMS PLOTJT /FLAGWAIT.
S JMP PLOT1 /CONTINUE.

S PLOTA, CLA
S PLPU /RAISE PEN.
S DCA PLOTNP! /INITIALIZATIONS.
S TAD <4070 /"0" TO X COORD.
S DCA PLOTNX
S TAD <4330 /"0" TO Y COORD.
S DCA PLOTNY
S JMS PLOTWT
S RETURN

S PLOT1, TAD PLOTNX /GET PREVIOUS X COORD.
S CIA CLL
S TAD \JX /FORM NX-NPX.
S SML /L=0: NPX<NX.
S CIA
S DCA PLOTDX
S RAL
S DCA PLOTNY
S TAD PLOTNY
S CIA CLL
S TAD \JY /FORM NY-NPY.
S SML /L=0: NPY<NY.
S CIA
S DCA PLOTDY
S TAD PLOTNY
S CIA CLL
S TAD PLOTDY
S SML CLA /L=0: DELTA X<DELTA Y.
S JMP PLOT2
S TAD PLOTDX /REVERSE NUMBERS.
S DCA PLOTNA
S TAD PLOTDY
S DCA PLOTNX
S TAD PLOTNA
S DCA PLOTDY
S INC
S AMD PLOTNY
S TAD PLOT1
S JMP A3

S PLOT2, TAD PLOTNY
S CLL BAR
S A3, DCA PLOTMA
S TAD PLOTMA
S DCA PLOT4A
S TAD PLOT4V
S TAD PLOT3
S DCA PLOT4V
S TAD PLOT4V
S DCA PLOT4A
S TAD PLOTDX
S CLL RAR
S DCA PLOTMA
S TAD PLOTDX
S CIA
S DCA PLOT4V
S PLOT3, ISZ PLOT4V
S J'MP A4
C ALL DONE WITH PLOT.
RETURN
S A4, TAD PLOT'IA
S TAD PLOT'DY
S DCA PLOT'MA
S TAD PLOT'MA
S CIA CLL
S TAD PLOTDX
S S'IL CLA
S J'MP PLOT4 /SINGLE MOTION.
S PLOT'D3, TAD PLOT'D3A /COMBINED MOTION.
S J'MS OFFSCALE /OFFSCALE CHECK.
S TAD PLOTDX
S CIA
S TAD PLOT'MA
S DCA PLOT'MA
S A5, J'MP PLOT3
S PLOT4, TAD PLOT4A /OFFSCALE CHECK.
S J'MP A5
S PLOTT1, A6
S A6, 0344 /PEN RIGHT(-Y).
S 0020 /PEN LEFT(+Y).
S PLOTT2, A7
S A7, 0034 /DRUM UP(-X).
S 0018 /DRUM DOWN(+X).
S PLOTT3, A8
S A8, 0044 /UP-RIGHT.
S 0024 /UP-LEFT.
S 0030 /DOWN-RIGHT.
S 0033 /DOWN-LEFT.
S
S /CHECK FOR OFFSCALE AND EXECUTE MOVEMENT.
S OFFSCA, 0
S DCA CMMD /SAVE COMMAND.
S CLAF /CLR "OFFSCALE" FLG.
S TAD CMMD /GET DIRECTION BIT.
S AND (0060
S SDFS /FLAG?
S SKP
S JMP BI
S DCA BITBUF /NO, STORE BIT.
S DCA COUNT
S JMP EXEC
S 31, SNA /FLAG, RELEVANT MOVE?
S JMP EXEC /NO.
S AND BITBUF
S SNA CLA
S JMP 32
S ISZ COUNT /YES, UPCOUNT.
S JMP EXEC
S 32, TAD COUNT
S SNA CLA /ALLOW ONSCALE MOVE?
S JMP EXEC
S CMA /NO, DOWNCOUNT.
S TAD COUNT
S DCA COUNT
S TAD (0060 /DELETE ONSCALE MOVEMENT.
S CMA
S AND CMND
S DCA CMND
S EXEC, TAD CMND /GET ALLOWED CMD & DO IT.
S PLCF
S PLLR
S JMS PLOTUT
S JMP I OFFSCA

S PLOTUT, 0 /PLOTTER FLAG WAIT.
S 31, PLCF
S 31 JMP C1
S PLCF
S CLA CLL
S JMP I PLOTUT
END
References for saccadic eye movement:


References for the Cornsweet-Crane eyetracker:


References for the computer system:


