To the Graduate Council:

I am submitting herewith a thesis written by Robert Joseph Bryant entitled "Computer Simulation of Interpolating Techniques for Generating a Realtime EEG Potential Contour Display." I recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

J. M. Moshell, Major Professor

We have read this thesis and recommend its acceptance:

Accepted for the Council:

Vice Chancellor
Graduate Studies and Research

# COMPUTER SIMULATION OF INTERPOLATING TECHNIQUES FOR GENERATING A REALTIME EEG POTENTIAL CONTOUR DISPLAY

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Robert Joseph Bryant

March 1978

ACKNOWLEDGMENTS

I would like to express my appreciation to Dr. J. M. Moshell for his encouragement and direction in the preparation of this paper. I would also like to thank Mr. Bill Baird and West High School for the use of their photographic facilities.

Thank you also goes out to my family and all my friends for providing the much needed moral support.

## ABSTRACT

This study is concerned with evaluating the feasibility of using hardware adaptable interpolation techniques for approximating the EEG potential field extending over the human scalp. The criteria require that the technique be realizable by analog components and that the approximations be as reliable as that achieved from Bilinear interpolation. A successful interpolation routine would allow the construction of an analog device capable of receiving voltage signals from electrodes and producing a realtime gray-scale contour map of the EEG potential field on a video display screen. A device of this type would be particularly useful in the area of biofeedback research.

A computer program was written to simulate the activity of several interpolation techniques. Voltage data presented to the program consisted of specific test pattern data and real EEG data records. A TV monitor was used to display the resulting contour maps which were photographed to permit direct image comparisons among the several techniques.

The results show that the Inverse-square radius weighting technique produces images similar to that of the Bilinear, but it contains some undesirable features. The use of a smearing technique substantially reduced these features. The success of the combined Inverse-square and smearing technique and their adaptability to analog components suggest their use in constructing a realtime EEG potential contour display device.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE PAGE

# CHAPTER I

## INTRODUCTION

Research in the area of electroencephalography is aimed at obtaining a better understanding of how the brain functions. Standard clinical EEG measurements are made using electrodes placed directly on the scalp surface. These measurements provide two types of information about the electrical fields generated by the human brain, frequency of the voltage variations and differences in the voltage variations between two points on the scalp surface (Daube and Lake, 1972). The data collected can then be subjected to a variety of analysis techniques.

Data collected in this manner poses the problem of data redundancy. To illustrate this, consider the amount of data collected during a standard clinical EEG recording session. Using six to eight channels at 3600 cycles per second over a span of 20 minutes can result in excess of one-half billion numbers. However, the activity measured may, quite often, be explained in just a few sentences (Walter and Shipton, 1951).

An approach to the problem of redundant data was demonstrated as early as 1947 at the First International EEG Congress. The device is known as the Toposcope. It was constructed using a single cathode-ray tube connected to electrodes placed on the subject's scalp. The electron beam was modulated at different positions on the screen to reflect the relative intensity of the voltage level recorded by correspondingly located electrodes. The Toposcope was a simple device and quite limited in resolution, but it did provide a pleasant

demonstration of topographic detail. Unfortunately, the limitation of then current technology prevented significant improvements in the display technique (Walter and Shipton, 1951).

This technique did, however, spark sufficient interest to encourage others to pursue research in the area of spatio-temporal display of EEG data. This is indicated by work in the early 1950's by Cohen, Marko and Petsche, and W. G. Walter and Shipton and in the 1960's by Shipton, Remond and D. O. Walter (Shaw, 1970). Until recently, many of the efforts were centered around devices similar to the original toposcope or generation of potential contour maps that were either hand-drawn or produced on computer printout forms.

More recently, improvements have been made possible due to technological advances, notably the EEG Computerized Display by T. Estrin and R. Uzgalis, 1969. The technique utilized improvements in analog-digital conversion, high-speed computers and graphic display systems. The recorded electrical activity was stored on magnetic tape. It was later processed on the computer to generate a separate potential contour map on a cathode-ray tube for each time instant recorded. The cathode-ray tube was then photographed under computer control. The film produced could then be viewed later, primarily for study of alpha wave bursts (Estrin and Uzgalis, 1969).

Another research effort reported by Daube and Lake in 1972 resulted in a technique which produced a computer generated, three dimensional display of potential fields for single time instants. Two separate electrode arrangements were involved, from which data was recorded and

stored on magnetic tape. The three dimensional display requires selection of the time instant and one of several analytical options. It also allows orientation and size alteration along with suppression or enhancement of certain characteristics.

These two efforts, and others not mentioned, indicate the vast improvements achieved since the introduction of the first toposcope. However, none have been able to display the potential field distribution information in real time, such as would be required for standard clinical electroencephalography.

This presentation investigates the feasibility of constructing a special purpose hardware system to display, in real time, a two dimensional contour map of the potential fields on the scalp surface on a video display screen. This means that a visual evaluation of the potential field over the scalp would be possible while the EEG data is recorded.

This type of display would aid the electroencephalographer in increasing his awareness of the oscillating potential continuum that extends across the scalp. Visualizing such an image has been a problem for electroencephalographers, since most tend to think in terms of one dimensional traces, displaying potential difference as amplitude against time as abscissa (Petsche, 1970). An area in which this would be of special interest is biofeedback research.

Harris, 1967, placed sixteen electrodes along a great circle over the scalp surface to determine the minimum number of electrodes required to approximate the voltage curve along these sixteen electrodes. He

selected n successive electrodes and estimated the curve by an $n - 1$ degree polynomial. The criteria for a successful approximation require that all the minimal and maximal points be detected. His results indicate that an average of five and a maximum of nine electrodes are required. From this, it is reasonable to assume that the $5 \times 5$ electrode grid used in the research will, on the average, reliably sample the electrical activity.

The Bilinear interpolation scheme, explained in Chapter III, is the basic standard against which the interpolation techniques present here will be compared. Adoption of this technique is supported by its wide use for interpolating between random points on a surface (McLain, 1976) and Remond's discussion of its utility in 1968.

## CHAPTER II

## THE SIMULATION APPROACH

The EEG potential display system under investigation, as shown in Figure 1, is composed of three main components: EEG data collection, hardware interpolation, and video display. The hardware interpolation device is the portion of consequence, as the remaining parts are currently available.

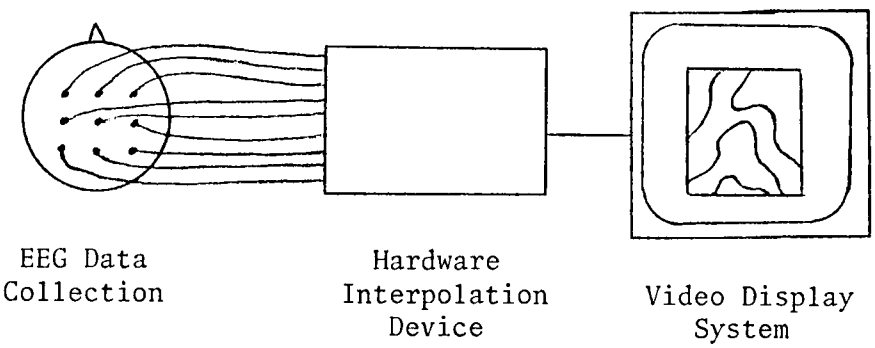


Figure 1. Schematic of the EEG display system.

The interpolation device calculates an approximation of the potential fields existing between the electrode positions which are then used as input to the video display system. Use of a general purpose mini-computer to generate the data for display on a TV screen would not be possible in a real time environment. To illustrate this, let us consider the number of operations that would be required per second. A standard TV screen is approximately a 500 by 500 picture element array, or has $2.5 \times 10^5$ positions, and it requires 30 new images per second.

Therefore, the computer must calculate $7.5 \times 10^6$ picture elements per second. Now, using one of the most simple interpolation routines, in terms of computations per picture element, there are five computations required per picture element. This totals to approximately 37.5 million operations per second without regard to any logical operations, such as determining which electrodes are to be used in the current computation. Even with current technology, the computation rate required to perform this many operations per second is beyond the capability of general purpose mini-computers. Therefore, a special purpose hardware device would be required to process the input signals from the electrodes and to produce the video brightness signals for the display system.

However, a special purpose device could not be easily modified should the interpolation algorithm implemented be found to be unsatisfactory. Therefore, a computer program has been written to simulate the activity of this hardware component.

The software simulation program has been used to evaluate several interpolation techniques. After initial testing of each routine, modifications were made to compensate for characteristics found to be undesirable and to enhance those that were desirable. Changes at this stage of the investigation are much easier and more quickly completed than after the device has been built.

This software package consists of several interpolation routines to be evaluated. Each routine, in turn, has input parameters that adjust its operation such that there is a large number of possible interpolation routines. Evaluation of these routines included the use of

artificial test data, whose potential contour maps were known, and the use of actual EEG recordings. To complete this simulation and to provide a visual evaluation of the success of the interpolation routines, a video display system has been provided.

The evaluation of the success of each interpolation routine is based upon intuitive knowledge of what the image should look like. This approach has been taken due to the lack of knowledge concerning the actual potential fields (Lehmann, 1971). A contributing factor to this lack of knowledge is the nonhomogeneous nature of the human head, that is, the convoluted brain and the inconsistent thickness of the skull. This is complicated by the lack of certainty concerning the number and location of EEG generators (Lehmann, 1971). However, since the scalp is also known to be an averager of the potentials generated by the cortex (Daube and Lake, 1972), it is believed that the continual display of the electrical patterns that are observed will provide an insight to the actual potential activity.

# CHAPTER III

## DISCUSSION OF INTERPOLATION PROCEDURES

### Bilinear Interpolation

The concern of interpolation algorithms is the computation of "close" approximations to continuous functions using points for which the actual values are known. In this study, the value of the points sampled are voltage signals which are collected using physiological electrodes placed on the surface of the subject's scalp in regular array patterns. Figure 2 represents such an arrangement using four electrodes labelled A, B, C and D, with corresponding instantaneous voltages $V_a$, $V_b$, $V_c$ and $V_d$.

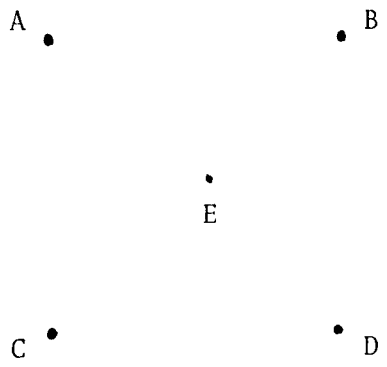


Figure 2. Sample arrangement of four electrodes.

A fundamental technique for approximating the voltage of point E that is equally distant from all four electrodes is

$$V_e = \frac{V_a + V_b + V_c + V_d}{4}$$

which is the average. However, as point E moves closer to some electrodes than others, the use of weighting factors is required to reflect the degree of contributation of each electrode to $V_e$. The Bilinear interpolation technique accomplishes this weighting function by computing point values relative to the distance between each point and the surrounding electrodes.

Initially, two Bilinear routines were considered, termed "X-first" and "Y-first." The difference between these two routines lie with the order in which the interpolation is performed and not in the general algorithm. An evaluation of these two routines shows them to be equivalent; they can be reduced to a common set of equations which were used in the research. Further details pertaining to the "X-first" and "Y-first" routines are found in Appendix A.

The Bilinear technique incorporated in the software program consists of two cases. The first case (see Figure 3) involves points which occur on line segments between two electrodes. Points P and Q lie on vertical line segments, whereas points R and S lie on horizontal line segments. The corresponding voltages are computed by

$$V_p = V_a + (V_c - V_a)\ (y_1/y_2)$$

$$V_q = V_b + (V_d - V_b)\ (y_1'/y_2)$$

$$V_r = V_a + (V_b - V_a)\ (x_1/x_2)$$

$$V_s = V_c + (V_c - V_d)\ (x_1'/x_2)$$

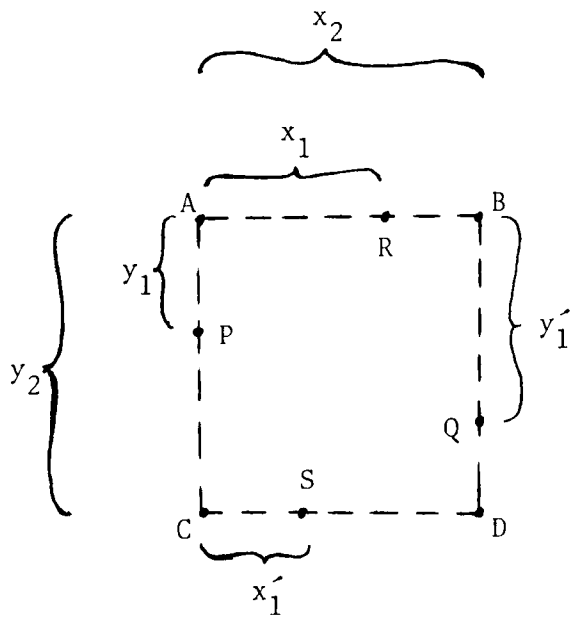


Figure 3. Location of points for case 1 of Bilinear interpolation.

The second case (see Figure 4) involves points, such as T, that lie on the interior of the region described by the four electrodes and whose voltages are computed by

$$V_t = V_a \cdot (1 - x_1/x_2 - y_1/y_2 + (x_1/x_2 \cdot y_1/y_2)) +$$
$$V_b \cdot (y_1/y_2 - (x_1/x_2 \cdot y_1/y_2)) +$$
$$V_c \cdot (x_1/x_2 - x_1/x_2 \cdot y_1/y_2)) +$$
$$V_d \cdot (x_1/x_2 \cdot y_1/y_2)$$

As indicated, the Bilinear technique considers only four electrodes at any one time, but the arrangement typically involves more than four. Interpolation of an arrangement with more than four electrodes is handled by segmenting it into several regions. Each region is the area bounded by four electrodes joined by straight line segments. For example, an arrangement of 25 electrodes would be segmented into 16 interpolation regions (see Figure 5). The effect of this is twofold: interpolation over the entire electrode net is continuous, but its first derivative is not (McLain, 1974); and noise generated by region switching, in a hardware system, would interfere with the brightness data generating the image.

Processing regions separately results in an image with "creases" (discontinuities of the first derivative) along region boundaries. This is most apparent when adjacent electrodes have readings that vary significantly. Hardware implementation of the Bilinear method would require that electronic circuits be provided to perform the region

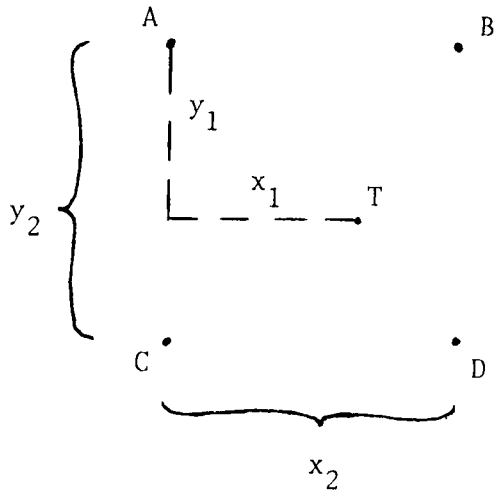


Figure 4. Location of points for case 2 of Bilinear interpolation.

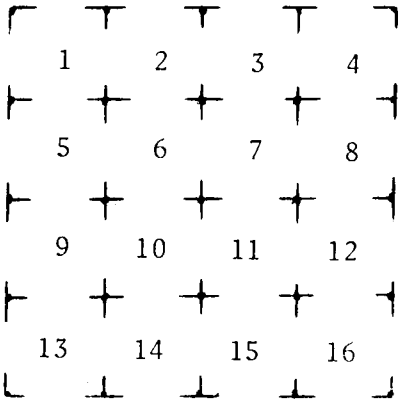


Figure 5. Identification of regions for Bilinear interpolation.

switching function. This switching event occurs with the transition from region to region and is accompanied by a certain amount of noise, which when injected into the video brightness signal would form a straight line or checkerboard pattern on the TV screen and would obscure the image. If the noise generated could be made regular then there would be possibilities for using it as a coordinate system to locate the electrode sites. However, as it is, the result of electronic noise, its shape and appearance on the screen, would be difficult to control. This switching problem is the major obstacle to hardware construction of Bilinear interpolation.

Continuous Interpolation

A desirable alternative to the Bilinear method would be the use of a continuous interpolation function that utilizes every electrode in the interpolation of each point. This would eliminate the concern over which region the current point is located and the associated discontinuities along region boundaries.

The question now is what weighting factor can be applied which will best emphasize the contribution of electrodes located close to the current point and minimize the contribution of more distant electrodes. For the moment, let $w_i$ represent the weighting factor. The interpolated voltage, $V_e$, of some point E would then be calculated by

$$V_e = \sum_{i=1}^{n} w_i v_i$$

where n is the number of electrodes.

Two promising candidates, as weighting factors, are the Inverse radius and the Inverse-square radius. Both will be discussed in further detail in Chapter IV.

Physiological Justification

One might ask: Do we not take a risk in arbitrarily "guessing" what activity is actually occurring between the electrodes? The answer is obviously yes, but then so does the Bilinear method, which will be shown in Chapter V, to contain characteristic features that do not correspond to actual activity. Let us consider for a moment two basic questions: What type of information are we really able to produce and how can it be applied?

As discussed in Chapter II, there are several physiological problems associated with the reconstruction of bioelectrical fields that are far from being solved. Keeping these problems in mind, the best that can be realistically hoped for is a surface mapping of potential fields that provide a basic indication of where the predominant activity occurs.

An area in which this type of potential field display is of immediate interest is biofeedback research. Here, the exact numerical magnitude of the underlying phenomenon is only of secondary interest. What the biofeedback trainer desires is that the images presented exhibit a range of distinguishable patterns corresponding to different electrical states of the brain. When the trainer detects a desirable state, by traditional methods, the subject will be able to recognize that state, as well as similar states. The recognition is not the

result of a highly accurate model of underlying brain activity, but is the result that similar brain activities will yield similar visual images. This is the key to the utility of such a display device for biofeedback training.

# CHAPTER IV

## NONLINEAR INTERPOLATION

### Inverse and Inverse-Square Radius

Consider the arrangement of electrodes as being represented by an $N \times M$ array, where N and M may or may not be equal. Let $K = N \times M$ be the total number of electrodes and let the position of the $i^{th}$ electrode in some Cartesian coordinate system be represented by $e_i = (x_i, y_i)$. Also, let $p = (p_x, p_y)$ represent any point within the convex polygon enclosing the K electrode sites. The radius, $R_i$, of p from any electrode $e_i$, is then defined as

$$R_i = \sqrt{(x_i - p_x)^2 + (y_i - p_y)^2}$$

For any weighting function of the radius, $W(R_i)$ and some assignment of voltage values, $v_i$, to $e_i$, the voltage at p is defined to be

$$V = \sum_{i=1}^{K} W(R_i) \cdot v_i$$

The value of V is then merely the sum of each electrode weighted in accordance to its distance from p.

The two main weighting functions used in this research are the Inverse radius and the Inverse-square radius, where $W(R_i) = 1/R_i$ and $W(R_i) = 1/R_i^2$, respectively. Using the Inverse radius as an example, the voltage at p would then be

$$V = \sum_{i=1}^{K} \left( \frac{1}{R_i} \right) \cdot v_i$$

Normalization

As point p approaches the location of any electrode, the radius decreases, causing the weighting factor, whether it be $1/R_i^2$ or $1/R_i$, to approach infinity. Subsequently, the voltage at point p will also approach infinity. This problem is alleviated by using the normalized value of p, whereby the voltage at point p becomes

$$V_N = \frac{v}{\sum_{i=1}^{K} W(R_i)} = \frac{\sum_{i=1}^{K} W(R_i) \cdot v_i}{\sum_{i=1}^{K} W(R_i)}$$

The following illustrates the effect of normalization. When point p is relatively distant from any electrode, $e_i$, the radius is relatively large resulting in a normalizing factor that is small, close to one, and has no significant effect upon the resulting interpolated value. However, as point p approaches an electrode site, the normalizing factor approaches infinity as does the weighting factor and effectively limits the voltage at p to $v_i$. In other words, as the radius approaches zero, both the weighting factor and the normalizing factor approach infinity. They then cancel each other leaving a maximum interpolation value equal to $v_i$.

Problems with Inverse Normalized Displays

During initial testing, contrived electrode voltages were presented to the software package to identify undesirable image characteristics

attributable to the Inverse-normalized interpolation routines. Two such characteristics were exhibited by both the Inverse and Inverse-square routines. Due to a lack of better terminology, these image flaws were called "knobs" and "ripples."

Knobs tend to occur around electrode sites which lie near brightness boundaries. Figure 6 illustrates the occurrences of knobs in an image that ideally would have been represented by a set of concentric circles. Ripples were noticed in images where each horizontal row of electrodes were assigned the same voltage (see Figure 7). Preferably, the image would have shown various brightness levels separated by nearly straight horizontal lines.

Two possible solutions were implemented in an effort to reduce these incongruous features and to provide a smoother image.

Face-centered point generation. In an effort to minimize the knobs and ripples, the Face-centered point generation routine was used to compute additional voltages for "pseudo-electrodes" which were interspersed in a regular fashion among the original electrodes. By doing this, the distance over which each interpolation is performed is decreased and the interaction of adjacent electrodes and "pseudo-electrodes" is increased, leading to smoother contour lines. The effect of this technique can be seen by comparing Figures 7 and 8. Note that ripples still occur, but that, overall, the image has generally straighter lines.

An array that is originally $n \times m$ is expanded by a single generation to an $n(n-1) \times m(m-1)$ array; the expansion is performed in

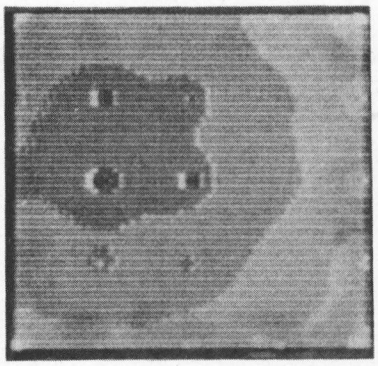

Figure 6. Occurrence of knobs.

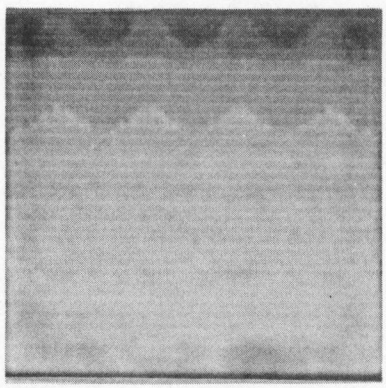

Figure 7. Occurrence of ripples.

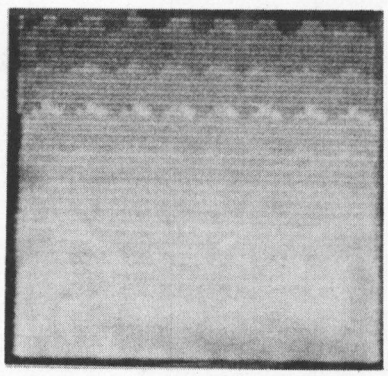

Figure 8. Application of Face-centered point generation to reduce occurrence of ripples.

two steps. The first step computes a value for each set of four commonly adjacent electrodes (see Figure 9). This "pseudo-electrode" is equally distant from the four surrounding electrodes; its voltage is computed by

$$V = \frac{V_1 + V_2 + V_3 + V_4}{4}$$

which is the numerical average. The second step computes the remaining points, involving two cases where the electrodes contributing to the value of the centered point are located above, below, and to either side. In the first case, the centered points are along the perimeter as in Figure 10. There are only three values that may be used since the fourth point is outside the sampling area. The value is computed by

$$V = \frac{V_1 + V_2 + V_3}{3}$$

In case two, all the remaining points are computed (see Figure 11) and again the four points are averaged using

$$V = \frac{V_1 + V_2 + V_3 + V_4}{4}$$

Any number of generations may be performed, resulting in an increasingly dense array of voltage points.

Originally, the addition of "pseudo-electrodes" via the Face-centered point generation routine were to be included as the result of both calculation steps. However, as development progressed it was

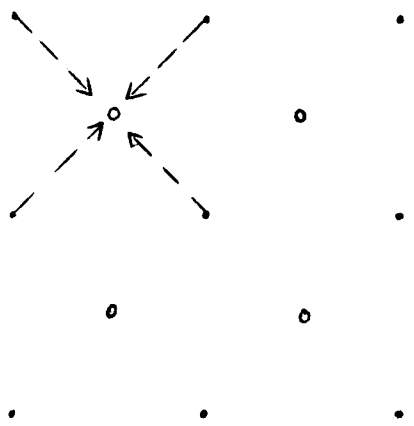

Figure 9. Case 1 example: electrode sites contributing to centered "pseudo-electrode."

• represents original electrode sites

o represents "pseudo-electrode" sites

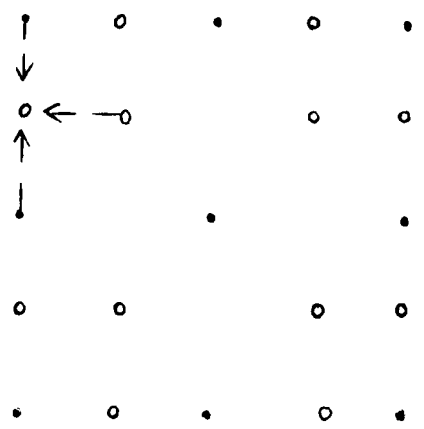

Figure 10. Case 2 example: electrode sites contributing to perimeter "pseudo-electrode."

• represents original electrode site

o represents "pseudo-electrode" site

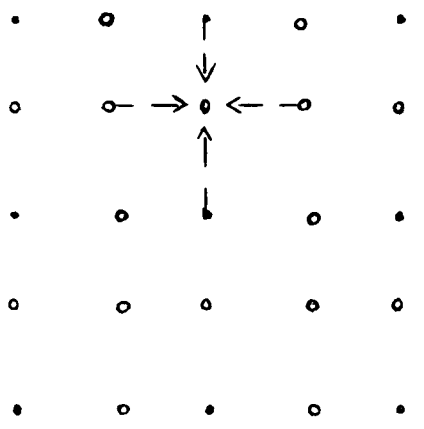

Figure 11. Case 2 example: electrode sites contributing to centered "pseudo-electrode."

• represents original electrode site

o represents "pseudo-electrode" site

decided to allow "pseudo-electrodes" to be included as the result of the first calculation step and become known as a "half-generation." Therefore, the Face-centered point generation routine can be applied in terms of one-half generations, which allow the effect of "pseudo-electrodes" to be studied in greater detail.

"Smear" technique. A second approach taken to reduce the knobs and ripples is the "smear" technique. As the name suggests, this technique is aimed at extending the influence of each electrode slightly to effect a contour smoothing operation. This might also be thought of as "blurring" the image where some of the fine details are lost, yet retaining the areas showing extreme potentials and rapidly changing potentials.

In essence, this technique says that all points lying within a specified "smear" radius of any electrode will be evaluated equally with respect to that electrode. The smear radius has as a maximum value of one-half the distance between any two adjacent electrodes.

Figure 12 illustrates how two points, A and B, would be evaluated with respect to electrode $e_i$ having a smear radius of $r_s$. Point B clearly has a radius, $r_B$, that is less than $r_s$. Therefore, the radius for point B with respect to $e_i$ would be $r_s$ instead of $r_B$. Point A, on the other hand, lies outside the smear radius and so its radius relative to $e_i$ would be $r_A$ and not $r_s$.

The effect of the Smear technique can be seen in Figures 13, 14, and 15. As the minimum radius value increases, indicated by the increasing "smear factor," the ripples become less noticeable. In the

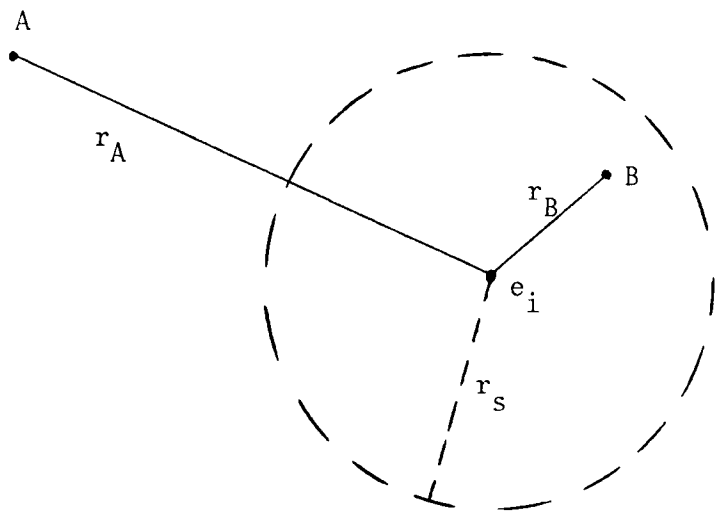


Figure 12. Illustration of "smear radius."

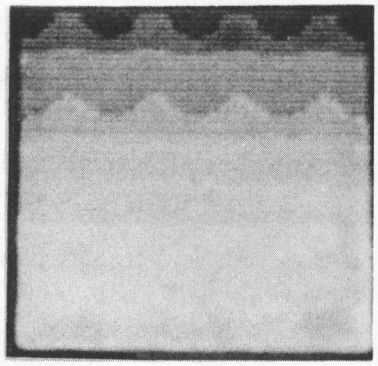

Figure 13. Smear technique with smear factor = .1, straight line data.

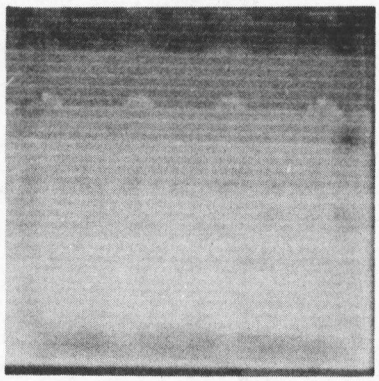

Figure 14. Smear technique with smear factor = .3, straight line data.

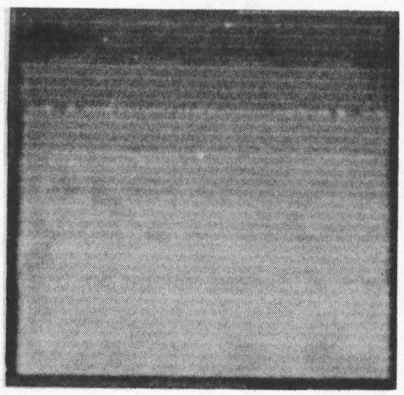

Figure 15. Smear technique with smear factor = .4, straight line data.

case of knobs, Figures 16 and 17 show that, as the minimum radius increases, considerable smoothing takes place.

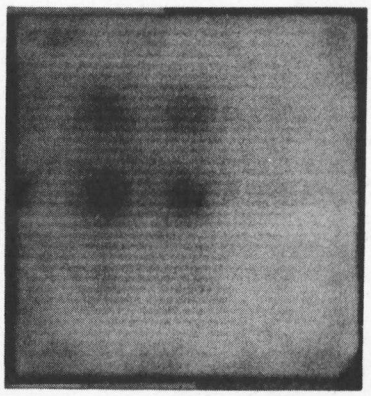

Figure 16. Smear technique with smear factor = .1, conical data.

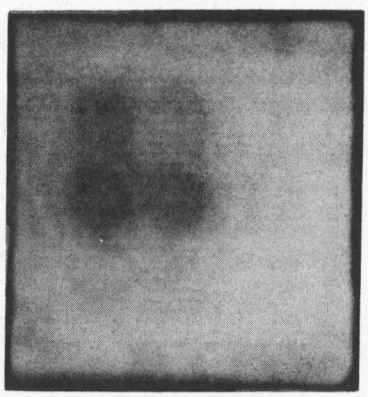

Figure 17. Smear technique with smear factor = .3, conical data.

# CHAPTER V

# THE INTERPOLATION INVESTIGATION

## The Computer Program

The computer program used to simulate the proposed hardware system is presented in three segments: specification of the desired interpolation routine and other required program parameters, performance of the interpolation subroutines, and production of the video image. The interpolation routine to be performed is determined by responses that the operator makes to prompting messages issued by the main program. The requested interpolation routine is performed by one or more subroutines. The video image is produced by another subroutine which converts the data, computed by the interpolation routine, into a form that is acceptable for display on the television screen.

The function of the main computer program is to assist the operator in correctly specifying the current simulation conditions, including the arrangement of the electrodes and their value assignments, as well as the desired interpolation technique and the value of any associated parameters. This is achieved in two ways. First, prompting messages are sent to the operator and contain, whenever possible, a list of allowable responses and a short explanation of each selection. Second, the specific prompting messages that are sent to the operator depend upon selections made to previous requests. Therefore, the operator knows that any requested input is required to assure proper operation

of the simulation program. This frees him from having to screen out messages irrelevant to current needs.

Description of the physical properties of the electrode arrangement is completed with the assignment of the electrode values. This may be accomplished by one of two different methods. The first method is by direct specification from the terminal. The operator is requested to input the value for each electrode position separately. The electrode positions are numbered by row and column according to standard array ordering.

The second method employs real EEG recordings that were stored on a disk pack and were input under program control. Standard operation is sequential input of consecutive time frame recordings. However, an option is available that allows selection of every $n^{th}$ time frame recording. Each record was recorded at standard time intervals, and using this option, one may sample a series of recordings at equal time intervals.

The size of the image produced on the video screen is determined by the number of individual points to be interpolated between any pair of horizontally or vertically spaced electrodes. This value, known as the "interelectrode distance," divides the actual distance between electrodes into a series of discrete points for which values are computed and then displayed. If a large image is desired, then a larger number of points between electrodes should be specified.

The maximum number of gray levels that can be displayed by the current program is sixteen. Any number of levels up to and including

sixteen may be specified by the operator in order to tailor the output image to specific needs.

Before the video image may be generated, the interpolation values must be scaled and assigned a corresponding gray level value. There are two methods for scaling the interpolation values and both may be applied to either terminal or disk input data. The first consists of segmenting the range from the minimum to the maximum values into equal intervals. This is termed equipotential gray level mapping and has been the standard practice in this research. The second method allows one to specify the individual minimum and maximum values for each gray level. This provides the ability to display several levels over a small range of computed values for a more detailed image within that range. A disadvantage of both techniques is that they require advance knowledge of the expected minimum and maximum electrode values. Advance knowledge is required since interpolation, scaling and display activities are performed one line at a time. However, this is not a problem when providing data from the terminal and is only a small inconvenience when using the actual EEG data from disk. The effect of an incorrect specification of the expected minimum and maximum values is the appearance of dark areas where the computed values fall outside the allowable range.

The basic interpolation routines examined in this presentation are the Bilinear, the Inverse-square radius and the Inverse radius. A fourth routine, known as Face-centered point generation, may be used in combination with any of the three basic interpolation algorithms.

The three interpolation routines compute values for each row of picture elements one at a time, beginning with the top row. As processing for each row is completed, the display routine is initiated to include that row in the video image. Control is then returned to the interpolation routine for additional row computations. This sequence continues until the last row in the image is completed, at which time control is returned to the main program. This interpolation and display procedure is used because the memory of the mini-computer is not sufficiently large to contain the entire video image at once.

The EEG data supplied to each interpolation routine is a two dimensional array containing the potential values recorded at each electrode site at a single time instant. The position where each value is found in the array reflects the location of the corresponding electrode on the scalp. The first row in the array represents the electrode row nearest the forehead of the subject with the last row representing the row farthest from the forehead. The resulting video image is oriented in the same manner. The observer is given a top view of the scalp with the nose of the subject centered at the top of the image.

There is one more common feature among the three interpolation routines that needs to be mentioned. The value of points that lie on electrode sites reflect the actual value recorded by the electrode, not some computed value.

The electrode arrangement for the Bilinear algorithm, INTERP, is divided into a set of regions. Each region is defined by four electrodes,

such that each lie in one corner of the region. As the current interpolation point moves from one region to the next, the associated electrode values to be used in the calculation must be updated. To simplify this process, a 2 × 2 array is maintained which always contains the electrode values associated with the region in which the current interpolation point occurs. It is the elements of this array that are used for calculating the interpolation value, regardless of which one of the three equations is required.

The Inverse radius and Inverse-square radius are two nonlinear techniques that utilize every electrode reading in computing each interpolation value. These two subroutines are operationally identical and, for this reason, the following discussion is applicable to both.

Unlike the Bilinear subroutine which requires knowledge of in which electrode region the interpolation point is located, the Inverse routines have no such requirement. The same electrode values are used to compute interpolation point values, regardless of their location with respect to the current interpolation point.

Due to the above feature of the Inverse subroutines, use of the Face-centered point generation subroutine may include half-generations. When half-generations are performed, extremely high values are assigned to the missing pseudo-electrode sites. These are detected by the Inverse subroutines and calculations that would normally incorporate them are bypassed.

Use of the Smear technique discussed earlier requires that the operator specify the value of the "smear factor." The input parameter

should have a value less than .5 and greater than zero. In the event that the effect of the Smear technique is not desired, a value of .001 should be entered. The subroutine will perform its calculations in the same manner regardless of the value of the "smear factor," but a value of .001 will effectively cause the Smear technique to be nullified.

The Face-centered point generation subroutine, FACE, computes "pseudo-electrode" values that are used, in addition to the original electrode values, as input to the requested interpolation subroutine. The number and location of the values computed depend upon a value provided by the operator, which indicates the number of generations. This value consists of two digits separated by a period. The digit to the left of the period indicates the number of complete generations, and the digit to the right indicates whether the first half of the next generation is to be computed. Using the number of generations requested, the subroutine generates an array of the proper dimensions to contain all the original electrode values plus the ones to be computed. If a half-generation is requested, then the array values for the second half of the incomplete generation are set to an extremely high value so that they will not be used in computations by the interpolation routines. At the completion of FACE, control is returned to the main program which immediately executes the requested interpolation.

If the Face-centered point generation subroutine is used in combination with the Bilinear, then no half-generations may be allowed. This is due to the fact that the Bilinear assumes that all elements in the electrode array are present. If any array elements are missing due to a half-generation, then the resulting video image will be invalid.

Two routines, written in assembler language for the PDP 11/40, are used to interface the FORTRAN IV routines with the video display unit. One subroutine, CLEAR, erases whatever image may have previously been on the video screen and prepares the system to accept the incoming data. The second routine, DISPLY, transforms the gray level numbers generated by the interpolation subroutines into a binary string acceptable to the video system.

The technique employed to produce the video image requires the use of two of the four available output channels. One channel is associated with the lower intensity eight gray levels while the other channel is associated with the remaining eight higher intensity gray levels. The binary string generated reflects this separation and incorporates video system instructions to display both channels on the same image line resulting in sixteen gray levels. The length of each image line and the number of lines in the image are independent of this subroutine and are only limited by the size of the screen.

## Photographic and Computer Hardware Description

The computer system consists of a PDP 11/40 with 32K words of memory and runs under the single user operating system RT-11. The peripheral equipment used includes one RK05 disk pack, a DEC VT05 alphanumeric terminal, a desk top line printer, and a Data Disc 6600 Series video display system. The simulation program and EEG test data are stored on the disk. The desk top line printer is available for printing the results of computations performed by the program. Figure 18 illustrates the equipment configuration.

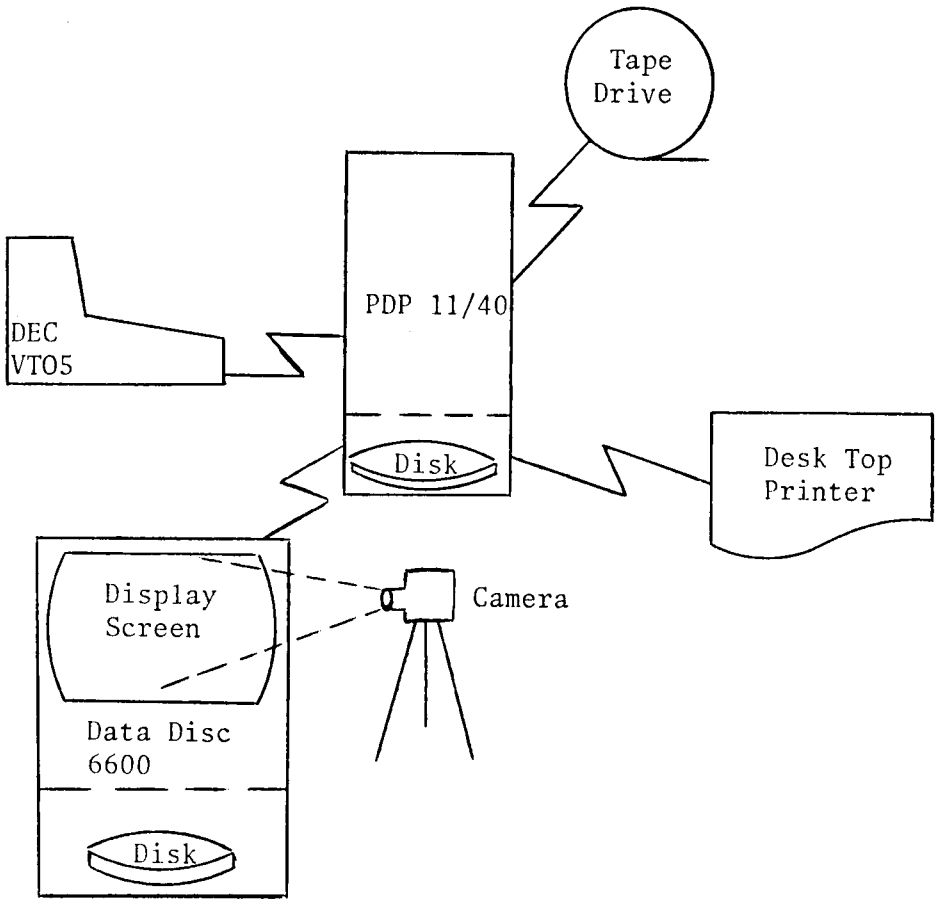


Figure 18. Schematic of computer and photographic equipment.

The video display system consists of two video screens or TV monitors, one color and one black and white, and a video data storage disk. The following discussion of the TV monitor will refer to only the black and white monitor since the color monitor was not used during the experiment. Appendix C addresses the conversion to a color display.

Data is stored on the video disk relative to the order in which it will be displayed. The TV monitors use an interlaced scan, meaning that all the even-numbered lines are displayed and then all the odd-numbered lines are displayed, and so on in an alternating fashion. This is achieved by storing data for the even-numbered display lines or even field on one half of the disk and the odd-numbered data lines or odd field on the other half.

TV monitors have no provisions for storing images on the screen and therefore require continual refreshing. The refreshing is performed by a constant transfer of display data from the disk to the video screen and is accomplished once during each complete revolution of the disk. The disk revolves at a rate of 30 times per second thus the entire TV image is refreshed 30 times each second. This refresh rate is fast enough so that before the brightness level of any point decays to a level perceptive by the human eye, the point has already been refreshed. This is why the image appears to be fixed and free of any distracting flicker.

Several types of data may be displayed on the video screen. The generation of images requires that instructions be presented to the video system prior to the data to be displayed. Alphanumeric characters

of varying sizes may be displayed by specifying the character and the screen location. Straight lines and right rectangles may also be drawn by indicating the correct location parameters. The fourth type of data that may be displayed is called graphic byte. It permits addressing of individual picture elements or pixels, which are singular display points whose screen positions are indicated by row and column numbers. There are 480 rows and 640 columns of addressable points. The rows are numbered 0 to 479 from top to bottom and the columns are numbered 0 to 639 from left to right. The last method was chosen primarily because of the flexibility it allows in generating an image and because of its simplicity. Data written to the screen may be in the form of either black and white or gray scale. Use of the gray scale option allows pixels to be assigned various brightness or light intensity levels ranging from black to white.

Data is transferred from the storage disk to the screen through what are called I/O channels. There are four channels, each of which may individually display up to four different gray levels. The gray level is determined by the value of a two digit binary number. The four possible combinations are 00, 01, 10 and 11. In addition, each channel has a different base line intensity. Thus, the four binary combinations will each display a different gray level when used with different channels, resulting in sixteen different levels. To increase the number of possible gray levels, multiple channels may be used simultaneously to display one pixel. If two channels are used, then a four digit binary number is used to describe one of sixteen possible gray level combinations. The same is true for the use of three and four channels

simultaneously. Three channels yield a six digit binary number and 64 possible gray levels, while four channels yield an eight digit binary number and 256 possible gray levels.

The equipment used to photograph the image on the video screen consists of a Minolta SRT-100 camera fitted with a No. 2 close-up lens. The lens was positioned 9.75 inches from the screen with an aperture of 5.6f and a shutter speed of 1/8 second. These settings were standard during each picture taking session to insure consistency in the quality of the pictures taken between different sessions. An entire picture taking session, using a series of images, was devoted to determining the settings best suited for photographing the TV monitor.

The shutter speed of 1/8 second allows for approximately four complete traces of the image during each exposure. At other shutter speeds the last image retrace was not completed due to the rotational speed of the video disk. This resulted in the portion of the image with the additional retrace appearing darker than the portion of the image without the additional retrace. Minor adjustments of the brightness and contrast controls on the video screen were made from time to time to enhance desired characteristics of the image.

## The Experiment

Determination of the interpolation routine to be suggested for hardware implementation is, at this point, based upon the quality of the image produced. Data presented to the simulation program consists of specific patterns of contrived electrode voltages and a full second of real EEG potential recordings. Unless otherwise specified, all data

patterns consist of 25 electrodes arranged in a regular 5 × 5 array pattern and are equally spaced, both horizontally and vertically. The contrived voltage patterns were specifically designed to test the flexibility of the Inverse and Inverse-square routines and the images produced were compared to the image generated by the Bilinear routine.

The two test patterns are referred to as "straight line" and "conical." Every electrode in each row of the straight line pattern is assigned the same voltage. The voltage assigned to the top row was 10 millivolts and was increased by 10 for each successive row, so that the bottom row had a value of 50 millivolts. The expected gray scale image would be a series of straight or nearly straight horizontal lines. The electrode sites for the conical pattern were assigned values to simulate the occurrence of a single voltage source which would generate a series of concentric circles. As seen in Figures 19 and 20, the Bilinear interpolation method approximates both these regions quite well. However, Figure 19, in particular, shows the straight lines that are characteristic of the Bilinear technique when it is used to approximate continuous regions.

The EEG data was supplied by Dr. Lehmann, Neurologische Universitatsklinisk und Poliklinik, Kantonsspital Zurich, Switzerland. The data used during the testing phases was real time spontaneous alpha EEG recorded using 48 channels, some of which were not always connected or became unusable during the recording session. The readings were made at constant time intervals of 4 milliseconds, allowing 250 samples per second. The data supplied was on a magnetic tape, and was not directly

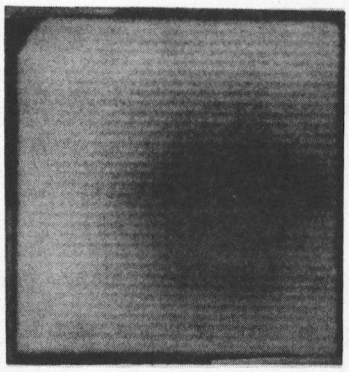

Figure 19. Bilinear interpolation of conical data.

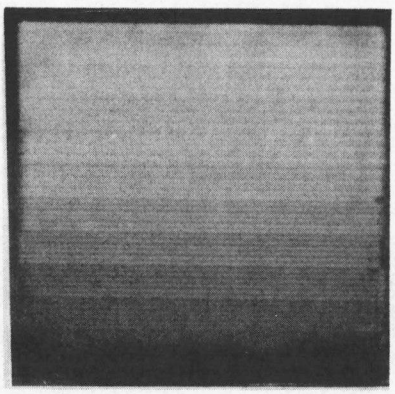

Figure 20. Bilinear interpolation of straight line data.

usable. A special program was written to read the data from the tape and write it to the mini-computer RK05 disk in a format acceptable to the simulation program. During this research, only 25 of the original 48 channels were used. Of the 250 time instants available, every tenth one was selected for investigation providing 25 equally spaced samples.

The experiment consisted of three phases: a pilot study, the main investigation, and a final concentrated study. The goal for Phase I was the determination of undesirable characteristics in the images generated by the Inverse radius and the Inverse-square radius interpolation routines and the selection of possible techniques for correcting these deficiencies. As discussed in Chapter IV, the characteristics observed were knobs and ripples and the contour smoothing techniques were the Face-centered point generation and Smear techniques.

An additional feature that became evident during Phase I is called a singularity. It appears as a small dot located at electrode sites and is completely surrounded by a different gray level, as in Figure 21. Each singularity is composed of the electrode site and one or more interpolated points around it. The occurrence of singularities indicate that the potential value of an electrode is not sufficiently large to affect many, if any, of the interpolated values of nearby points. In fact, the values of the points contained in the singularity may be quite close to the values of the points around it, but due to the particular range of values associated with each gray level, they are assigned different brightness values. Generally, the size of the singular region directly reflects the degree to which it differs with the surrounding

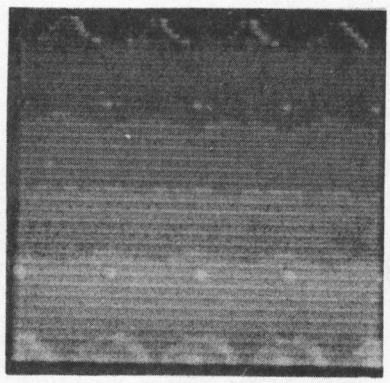

Figure 21. Illustration of singularities.

region. The smaller the singularity, the smaller the difference between the actual electrode voltage and the value of the interpolated points outside the singularity.

During Phase II, the smoothing techniques were implemented. To evaluate these techniques, another series of images using the test data were photographed for both the Inverse radius and Inverse-square radius routines.

When the Face-centered point generation routine was used with the conical data, additional singularities were observed toward the center of the image instead of a general smoothing of the contours (see Figure 22). The number of generations ranged from one-half to one and one-half with similar results in all cases. However, the Inverse-square image did have smoother contours than the Inverse, but that was due to the difference in the routines, not as a result of the Face-centered point generation. When straight line data was used, the results were somewhat better for both inverse routines in that the amplitude of the waves were noticeably reduced.

The Smear technique, when applied to the conical data, was quite successful considering the severe nature of the data. Figures 23 and 24 show that adjacent gray levels tend to form common gray level regions as the smear factor increases.

The Smear technique was also successful against the straight line data in terms of decreasing the amplitude of the wave form. In fact, for a smear factor of .4, the Inverse-square image, Figure 25, was very similar to that generated by the Bilinear, Figure 20 (page 41).

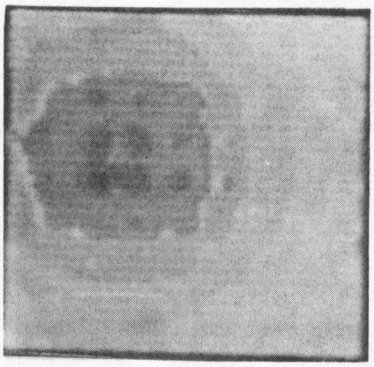

Figure 22. Inverse-square radius with one generation of Face-centered technique, conical data.

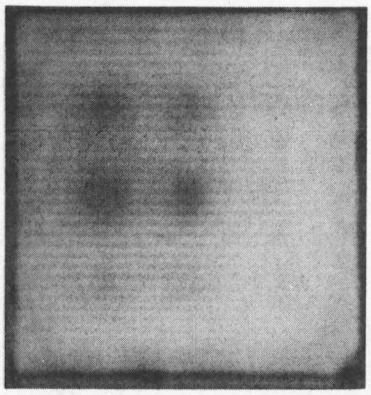

Figure 23. Inverse-square radius with a smear factor = .1, conical data.

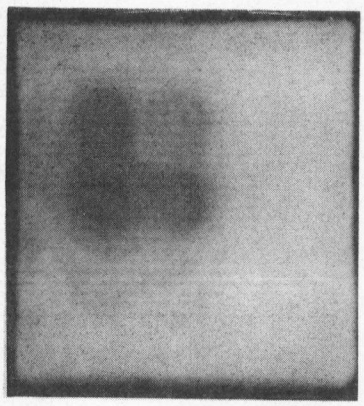

Figure 24. Inverse-square radius with smear factor = .3, conical data.

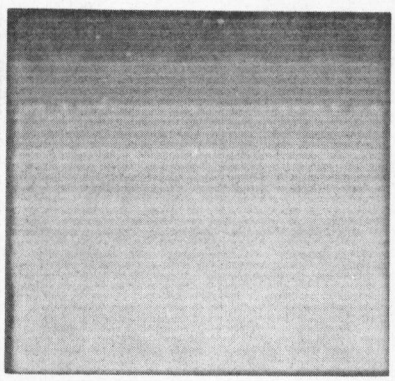

Figure 25. Inverse-square radius with smear factor = .4, straight line data.

Noting the singularities in Phase I, a more intensive study was conducted regarding their occurrence and effect. Singularities were found in both the Inverse radius and the Inverse-square radius images of real EEG data. There were no noticeable deficiences in the Inverse-square images, but the number of visible gray levels in the Inverse image were markedly less than in the same image generated using the Bilinear routine.

Application of the Face-centered point generation routine to reduce the occurrence of singularities was totally ineffective. The result was not fewer singularities, but more, for both the inverse routines. On the other hand, the Smear technique was quite effective in decreasing the size of the singularities and in some cases removing them completely. In either case, the basic problem is not with the occurrence of singularities, but rather the decreased number of visible gray levels exhibited by the Inverse radius images.

At this point, it became evident that the Inverse-square radius is a more promising technique than the Inverse radius and that additional investigation of the combination of the Inverse-square with both the Face-centered and Smear techniques should be pursued. During this phase, a large number of different images were used, most of which exhibited similar features. Therefore, a representative few were selected for use in Phase III.

The scope of the third and final phase is limited to the Inverse-square routine and images selected following Phase II. Considering the success of the Face-centered point generation in the

straight line test data and the increased amount of data it provides the interpolation routine, plus the singularity removal success of the Smear technique, the two were combined with the Inverse-square radius. The combination of these two techniques with the Inverse-square produced several different routines resulting from the various degrees to which the Face-centered point generation and Smear techniques were applied.

As expected, singularities were generated and were quite noticeable for low smear factor values, but as the value of the smear factor increased nearly all the singularities were removed. This was the case regardless how many generations of Face-centered point generations were performed. A comparison of the Inverse-square in combination with Face-centered point gneration and the Smear technique against the Inverse-square with only Smear does not show any great differences. The only point of conflict is the occasional singularities caused by the Face-centered point generation. Therefore, of the two routines, the Inverse-square with Smear would be preferable.

To compare the Bilinear, the Inverse-square and the Inverse-square with various smear factor values, three time instants have been selected (Figures 26 through 34). It is evident from this series of images that the three routines produce very similar images. However, the Inverse-square with a smear factor in the range of .2 to .3 exhibits smoother contours than the Inverse-square without the Smear technique.

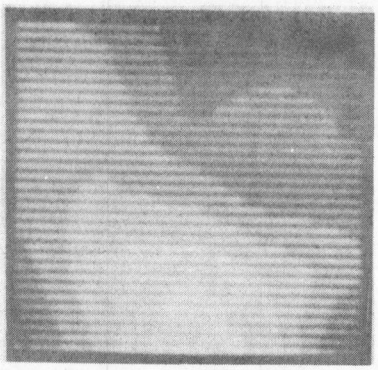

Figure 26

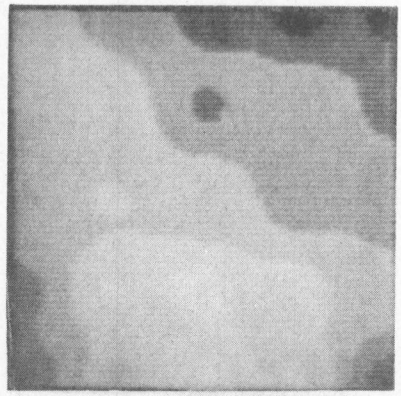

Figure 27

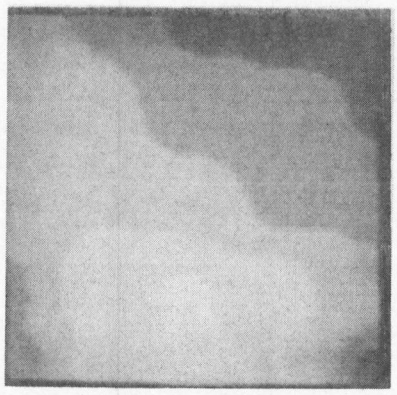

Figure 28

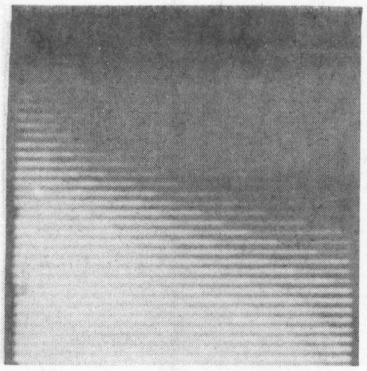

Figure 29

Figure 30

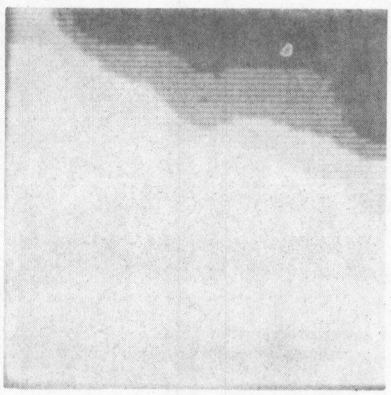

Figure 31

Figure 32. EEG data time frame 230, Bilinear interpolation.

Figure 33. EEG data time frame 230, Inverse-square radius interpolation.

Figure 34. EEG data time frame 230, Inverse-square radius with a smear factor = .3.

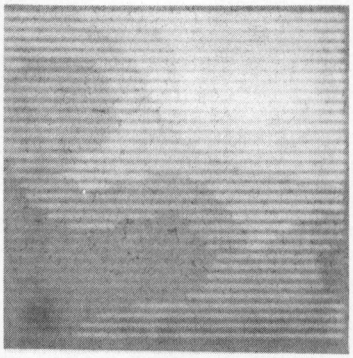

Figure 32

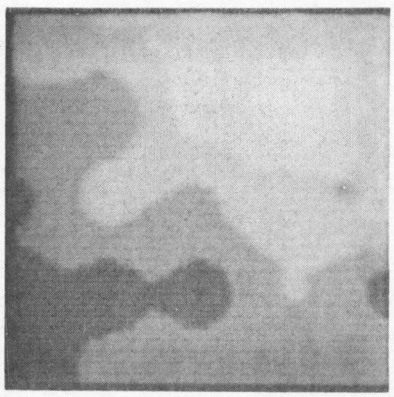

Figure 33

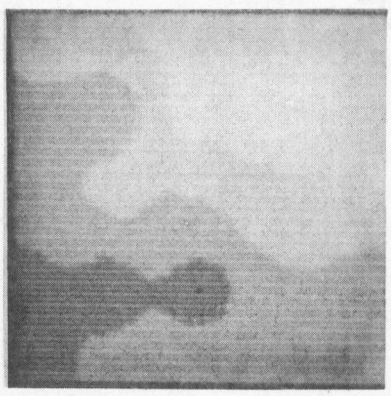

Figure 34

# CHAPTER VI

## CONCLUSION

The feasibility of constructing a hardware system capable of producing a real time two-dimensional contour map of the potential field over the human scalp on a TV screen is dependent upon three factors: the availability of an acceptable interpolation routine, the adaptability of that interpolation routine to electronic components, and the availability of the components.

Results from the experiment show that the Inverse-square radius interpolation when combined with the Smear technique is capable of producing an image that is very similar to that generated by the Bilinear interpolation routine. The simplicity of the calculations performed by the Inverse-square radius routine make it adaptable to hardware components which are available due to continuing technological advancements in electronic circuitry. Having fulfilled these requirements, the Inverse-square radius interpolation with the Smear capability should be considered as a viable option for hardware construction.

# LIST OF REFERENCES

## LIST OF REFERENCES


Daube, J. R., and Lake, R. M. "System for Computer-Generation of Three-Dimensional Display of Electroencephalogram," Proc. of the San Diego Biomedical Symposium, vol. II, 1972.

Estrin, T., and Uzgalis, R. "Computerized Display of Spatio-Temporal EEG Patterns," IEEE, Transactions on Bio-Medical Engineering, vol. BME-16, no. 3, July 1969.

Harris, J. A. "A Spatial Interpolation Procedure for Electroencephalography," M.S. thesis, University of Minnesota, Minneapolis, 1967.

Lehmann, D. "Multidimensional Topography of Human Alpha EEG Fields," Electroencephalogr. Clin. Neurophysiol., vol. 31, 1971.

McLain, D. H. "Drawing Contours from Arbitrary Data Points," Computer Journal, vol. 17, 1974.

_____. "Two Dimensional Interpolation from Random Data," Computer Journal, vol. 19, 1976.

Petsche, H. "The Quantitative Analysis of EEG Data," ed. J. P. Shade, J. Smith, Progress in Brain Research, vol. 33, 1970.

Remond, A. "The Importance of Topographic Data in EEG Phenomena and an Electrical Model to Produce Them," Electroencephalogr. Clin. Neurophysiol., vol. 27 (Supplement), 1968.

Shaw, J. C. "A Method for Continuously Recording Characteristics of EEG Topography," Electroencephalogr. Clin. Neurophysiol., vol. 29, 1970.

Walter, W. G., and Shipton, H. W. "A New Toposcopic Display System," Electroencephalogr. Clin. Neurophysiol., vol. 3, 1951.

# APPENDIXES

## APPENDIX A

## BILINEAR INTERPOLATION EQUATIONS

Equations for the "X-first" and "Y-first" Bilinear algorithm follow. The voltage for any point G is denoted f(G). Electrode sites are A, B, C and D.

X-first:

$$f_x(P) = f(A) + (f(B) - f(A))\ (x_1/x_2)$$

$$f_x(R) = f(C) + (f(D) - f(C))\ (x_1/x_2)$$

$$f_x(Q) = f(P) + (f(R) - f(P))\ (y_1/y_2)$$

Y-first:

$$f_y(P_1) = f(A) + (f(C) - f(A))\ (y_1/y_2)$$

$$f_y(R_1) = f(A) + (f(D) - f(B))\ (y_1/y_2)$$

$$f_y(Q) = f(P_1) + (f(R_1) - f(P_1))\ (x_1/x_2)$$

The interpolated values for points on the perimeter are computed by taking the difference between the two electrodes on either side and reducing the difference by a factor of the distance between them. The result is then added to the voltage of the electrode from which the point is measured. The point in question is Q and it will be shown in the following that $f_x(Q)$ is equivalent to $f_y(Q)$.

Expand $f_x(Q)$:

$$f_x(Q) = f(P) + (f(R) - f(P))\ (y_1/y_2)$$

$$f_x(Q) = [f(A) + (f(B) - f(A))\ (x_1/x_2)] +$$
$$[(f(C) + (f(D) - f(C))\ (x_1/x_2)) -$$
$$(f(A) + (f(B) - f(A))\ (x_1/x_2))]\ (y_1/y_2)$$

$$f_x(Q) = f(A) + f(B)\ (x_1/x_2) - f(A)\ (x_1/x_2) +$$
$$[(f(C) + f(D)\ (x_1/x_2) - f(C)\ (x_1/x_2)) -$$
$$(f(A) + f(B)\ (x_1/x_2) - f(A)\ (x_1/x_2))]\ (y_1/y_2)$$

$$f_x(Q) = f(A) + f(B)\ (x_1/x_2) - f(A)\ (x_1/x_2) +$$
$$f(C)\ (y_1/y_2) + f(D)\ (x_1/x_2)\ (y_1/y_2) -$$
$$f(C)\ (x_1/x_2)\ (y_1/y_2) - f(A)\ (y_1/y_2) -$$
$$f(B)\ (x_1/x_2)\ (y_1/y_2) + f(A)\ (x_1/x_2)\ (y_1/y_2)$$

$$f_x(Q) = f(A)\ [1 - (x_1/x_2) - (y_1/y_2) + (x_1/x_2)\ (y_1/y_2)] +$$
$$f(B)\ [(x_1/x_2) - (x_1/x_2)\ (y_1/y_2)] +$$
$$f(C)\ [(y_1/y_2) - (x_1/x_2)\ (y_1/y_2)] +$$
$$f(D)\ [(x_1/x_2)\ (y_1/y_2)] \qquad \text{(A-1)}$$

Expand $f_y(Q)$:

$$f_y(Q) = f(P_1) + (f(R_1) - f(P_1))\ (x_1/x_2)$$

$$f_y(Q) = [f(A) + (f(C) - f(A))\ (y_1/y_2)] +$$
$$[(f(B) + ((f(D) - f(B))\ (y_1/y_2)) -$$
$$(f(A) + (f(C) - f(A)\ (y_1/y_2)]\ (x_1/x_2)$$

$$f_y(Q) = f(A) + f(C)\ (y_1/y_2) - f(A)\ (y_1/y_2) +$$
$$[f(B) + f(D)\ (y_1/y_2) - f(B)\ (y_1/y_2) -$$
$$f(A) - f(C)\ (y_1/y_2) + f(A)\ (y_1/y_2)]\ (x_1/x_2)$$

$$f_y(Q) = f(A) + f(C)\ (y_1/y_2) - f(A)\ (y_1/y_2) +$$
$$f(B)\ (x_1/x_2) + f(D)\ (y_1/y_2)\ (x_1/x_2) -$$
$$f(B)\ (y_1/y_2)\ (x_1/x_2) - f(A)\ (x_1/x_2) -$$
$$f(C)\ (y_1/y_2)\ (x_1/x_2) + f(A)\ (y_1/y_2)\ (x_1/x_2)$$
$$f_y(Q) = f(A)\ [1 - (y_1/y_2) - (x_1/x_2) + (y_1/y_2)\ (x_1/x_2)] +$$
$$f(B)\ [(x_1/x_2) - (Y_1/y_2)\ (x_1/x_2)] +$$
$$f(C)\ [(y_1/y_2) - (y_1/y_2)\ (x_1/x_2)] +$$
$$f(D)\ [(y_1/y_2)\ (x_1/x_2)] \qquad \text{(A-2)}$$

Comparing equations (A-1) and (A-2), it is clear that they are equivalent.

# APPENDIX B

## COMPUTER PROGRAM LISTING

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:02              PAGE 001

0001          COMMON ERANGE(17), ELECTR(21,21), ISPACE, IROW, ICOL,
             *  MAXCOL, NUMLEV, IDEBUG, MAXROW, INTVAL(400)
      C ** COMMON BLOCK FOR SUBROUTINE FACE
0002          COMMON    /BLOCK1/EXPAND(21,21), ISETS, SETS, ISIZE, JSIZE, JDEBUG
      C ** COMMON BLOCK FOR SUBROUTINE ISQRAD AND SUBROUTINE INVRAD
0003          COMMON   /BLOCK3/CVAL(400), SMEAR
0004          DIMENSION LOCATE(48,3), EDATA(48)
      C
      C ****************  VARIABLE DEFINITIONS  ******************
      C
      C     IROW - # OF ROWS IN ELECTRODE NET
      C     ICOL - # OF COLUMNS IN ELECTRODE NET
      C     NUMLEV - # OF POTENTIAL LEVELS (GRAY LEVELS) - CURRENTLY EQUAL SPACING
      C     ISPACE - # OF POINTS BETW/ EACH ELECTRODE - CURRENTLY EQUAL  SPACING
      C     ELECTR(21,21) - VALUES FOR EACH ELECTRODE
      C     ERANGE(17) - BOUNDARY VALUES FOR POTENTIAL LEVELS
      C     EMAX - MAXIMUM POTENTIAL LEVEL VALUE
      C     EMIN - MINIMUM POTENTIAL LEVEL VALUE
      C     STEP - ABSOLUTE RANGE OF EACH LEVEL
      C     MAXROW - TOTAL # OF ROWS IN THE DISPLAY
      C     MAXCOL - TOTAL # OF COLUMNS IN THE DISPLAY
      C     NUMROW - # OF CURRENT ROW
      C
      C ***********************************************************
      C
      C
      C THE CALL TO SUBROUTINE CLEAR RESULTS IN BLANKING THE
      C VIDEO SCREEN PRIOR TO EACH DISPLAY
      C
0005  1       CALL CLEAR
0006          ICHNGE = 10
0007          NPASS = 0
      C
      C ----> INITIALIZE MAX & MIN ELECTRODE POTENTIALS
0008          EMAX = -1000.
0009          EMIN = 1000.
0010          WRITE(7,50)
0011  50      FORMAT(' DO YOU WANT TO SEE DEBUG MESSAGES ? '/
             *5X,' 0 = NO'/5X,' 1 = TERMINAL MESSAGES'/5X,' 2 = PRINTER',
             *' MESSAGES'/5X,' 3 = ALL MESSAGES')
0012          READ(5,200) IDEBUG
      C
      C --> INPUT SIZE OF ELECTRODE NET (IROW, ISIZE)
0013          WRITE(7,100)
0014  100     FORMAT(' INPUT # OF ROWS IN ELECTRODE NET')
0015          READ(5,200) IROW
0016  200     FORMAT(I3)
0017          WRITE(7,300)
0018  300     FORMAT(' INPUT # OF COLUMNS IN ELECTRODE NET')
0019          READ(5,200) ICOL
      C
      C --> KROW & KCOL ARE USED LATER TO DETERMINE WHICH ELECTRODE
      C     READINGS ARE USABLE
0020          KROW = IROW
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:02              PAGE 002

0021          KCOL = ICOL
      C
      C --> SPECIFY # OF INTERPOLATION POINTS BETWEEN EACH PAIR OF ELECTRODES
0022          WRITE(7,400)
0023  400     FORMAT(' INPUT # OF POINTS BETW/ EACH ELECTRODE')
0024          READ(5,200) ISPACE
      C
      C --> SELECT ELECTRODE DATA SOURCE
0025          WRITE(7,404)
0026  404     FORMAT(' INDICATE SOURCE OF EEG DATA'/
     *  5X,' 0 = TERMINAL'/5X,' 1 = DISK FILE')
0027          READ(7,200)  INPUT
0028          IF (INPUT .EQ. 1)  GO TO 970
      C
      C --> INPUT ELECTRODE VALUES FROM TERMINAL
0030          DO 500  I = 1, IROW
0031          DO 500  J = 1, ICOL
0032          WRITE(7,600) I, J
0033  600     FORMAT(' INPUT ELECTRODE POTENTIAL FOR POSITION (',I2,',',I2,')')
0034          READ(5,700) ELECTR(I,J)
0035  700     FORMAT(F4.2)
0036          IF (ELECTR(I,J) .GT. EMAX)  EMAX = ELECTR(I,J)
0038          IF (ELECTR(I,J) .LT. EMIN)  EMIN = ELECTR(I,J)
0040  500     CONTINUE
0041          GO TO 971
      C
      C ---> THE FOLLOWING CODE READS THE ELECTRODE POSITION VALUES FROM DISK
      C
      C       1.   INPUT ELECTRODE POSITIONS AND SET INDICATOR WHICH
      C         SPECIFIES WHETHER OF NOT THE SPECIFIC
      C         RECORDING POSITION IS USABLE.
      C
      C ---> ASSIGN 1 AS THE LOGICAL UNIT # FOR THE DISK FILE
0042  970     CALL  ASSIGN(1,'TEST1.DAT')
      C
      C --> SPECIFY THE NTH RECORD WHICH IS TO BE SELECTED FROM THE
      C     AVAILABLE DATA BY INDICATING N AT THE TERMINAL
      C     (ONLY USED WITH DISK DATA)
0043          WRITE(7,931)
0044  931     FORMAT(' SELECT EVERY NTH EEG RECORD FROM AVAILABLE',
     *  ' TIME FRAME - SPECIFY N  (I3)')
0045          READ(7,200)  NTHREC
0046          DO 979  K = 1, 48
0047          READ(1,980)  (LOCATE(K,J),J=1,2)
      C
      C   LOCATE(K,3) IS THE USABLE INDICATOR FOR EACH PAIR OF ELECTRODE COORDINATES
0048          LOCATE(K,3) = 1
0049          IF (LOCATE(K,1) .EQ.  0 .AND.  LOCATE(K,2) .EQ.  0) LOCATE(K,3)=0
0051  979     CONTINUE
0052  980     FORMAT(30X,2(8X,I2))
      C
      C --> OUTPUT LOCATE VALUES ONLY WHEN DEBUGGING ERRORS
0053          IF (IDEBUG .NE.  2) GO TO 1980
0055          DO 909  K = 1, 48
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:02              PAGE 003

0056          WRITE(6,976) K, (LOCATE(K,J),J=1,3)
0057  909     CONTINUE
0058  976     FORMAT(' LOCATE VALUES (K =',I2,',  J = 1,  2,  3)',3(5X,I2))
      C
      C       READ REMAINDER OF CURRENT DATA BLOCK (SKIP OVER NEXT 2 RECORDS)
0059  1980    READ(1,981)  I, J
0060  981     FORMAT(I2/I2)
      C
      C ---> INDICATE WHETHER OR NOT A PAUSE FOLLOWING EACH COMPLETED DISPLAY
      C       OUTPUT IS TO OCCUR (ONLY WHEN READING EEG DATA FROM DISK FILE)
0061          WRITE(7,1984)
0062  1984    FORMAT(' SHOULD I PAUSE AFTER EACH COMPLETED DISPLAY ?'/
             * 5X,' 0 = NO'/5X,' 1 = YES')
0063          READ(7,200)  IPAUSE
      C
      C ---> INPUT # OF GRAY LEVELS TO BE DISPLAYED
0064  971     WRITE(7,800)
0065  800     FORMAT(' INPUT # OF POTENTIAL LEVELS')
0066          READ(5,200) NUMLEV
      C
      C ---> SPECIFY METHOD FOR LEVEL RANGE SETTING
      C         (IRANGE = 0  CANNOT BE USED W/ DISK DATA)
0067          WRITE(7,1825)
0068  1825    FORMAT(' INDICATE METHOD FOR SETTING POTENTIAL',
             * ' LEVEL RANGE VALUES'/5X,' 0 = EQUI-POTENTIAL',
             * ' RANGES BETWEEN MIN & MAX INPUT VALUES'/5X,' 1 = INPUT',
             * ' SPECIFIC RANGES (CONTINUOUS)'/5X,' 2 = SPECIFY MIN & MAX',
             * ' VALUES (EQUI-POTENTIAL)')
0069          READ(7,200)  IRANGE
0070          IF (IRANGE .EQ. 0)  GO TO 1800
0072          IF (IRANGE .EQ. 1)  GO TO 1826
      C
      C ---> INPUT SPECIFIC MIN & MAX VALUES TO BE DISPLAYED
0074          WRITE(7,1801)
0075  1801    FORMAT(' INPUT MIN POTENTIAL VALUE TO BE DISPLAYED (F12.8)')
0076          READ(7,1831)  EMIN
0077          WRITE(7,1803)
0078  1803    FORMAT(' INPUT MAX POTENTIAL VALUE TO BE DISPLAYED (F12.8)')
0079          READ(7,1831) EMAX
      C
      C -----> CALCULATE EQUI-POTENTIAL RANGE VALUES
      C         STEP - CONSTANT INCREMENT BETWEEN EACH LEVEL
      C         ERANGE(1) = MINIMUM POTENTIAL LEVEL VALUE (MIN
      C               ELECTRODE VALUE)
      C         ERANGE(NUMLEV+1) = MAXIMUM POTENTIAL LEVEL VALUE
0080  1800    STEP = (EMAX - EMIN) / NUMLEV
0081          ERANGE(1) = EMIN
0082          ERANGE(NUMLEV+1) = EMAX
0083          WRITE(7,78) STEP,EMAX,EMIN
0084  78      FORMAT(' STEP=',F8.3,' EMAX=',F8.3,' EMIN=',F8.3)
0085          DO 900  I = 2, NUMLEV
0086          K = I - 1
0087          ERANGE(I) = ERANGE(K) + STEP
0088  900     CONTINUE
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:02              PAGE 004

0089          GO TO 72
      C
      C --> SPECIFY MIN & MAX VALUE FOR EACH LEVEL (CONTINUOUS)
0090  1826    WRITE(7,1827)
0091  1827    FORMAT(' INPUT MIN DATA VALUE TO BE ANALYZED (F12.8)')
0092          READ(7,1828)  EMIN
0093  1828    FORMAT(F12.7)
0094          ERANGE(1) = EMIN
0095          DO 1829  K = 1, NUMLEV
0096           WRITE(7,1830)  K
0097          READ(7,1831)  EMAX
0098          ERANGE(K+1) = EMAX
0099  1829    CONTINUE
0100  1830    FORMAT(' INPUT MAX VALUE (F12.8) FOR LEVEL ',I2)
0101  1831    FORMAT(F12.8)
0102          WRITE(6,556)  EMIN, EMAX
0103  556     FORMAT(' EMIN = ',E20.6,'  ,  EMAX = ',E20.6)
      C
      C --> OUTPUT TO TERMINAL THE RESULTING LEVEL RANGES
0104  72      DO 73  K = 1, NUMLEV
0105          WRITE(7,74)  K, ERANGE(K)
0106  73      CONTINUE
0107          WRITE(7,74)  K, ERANGE(K)
0108  74      FORMAT(' ERANGE(',I2,') = ',F12.7)
      C
      C --> READ EEG DATA FROM DISK FILE
      C       (IF INPUT = 0  THEN DATA HAS ALREADY BEEN INPUT FROM TERMINAL)
0109          IF (INPUT .NE. 1) GO TO 545
      C
      C --> DETERMINE ROW & COLUMN INCREMENT VALUES W. R. T. ELECTRODE NET SIZE
      C       (INCRJ = COLUMN INCREMENT)
      C       (INCRI = ROW INCREMENT)
0111          INCRJ = -1
0112          IF (KCOL .EQ. 2) INCRJ = 4
0114          IF (KCOL .EQ. 3) INCRJ = 2
0116          IF (KCOL .EQ. 5) INCRJ = 1
0118          INCRI = -1
0119          IF (KROW .EQ. 2) INCRI = 4
0121          IF (KROW .EQ. 3) INCRI = 2
0123          IF (KROW .EQ. 5) INCRI = 1
0125          IF (INCRI .GT. 0 .AND. INCRJ .GT. 0)  GO TO 1832
0127          IF (INCRI .LT. 0)  WRITE(6,3210)  KROW
0129          IF (INCRJ .LT. 0)  WRITE(6,3110)  KCOL
0131  3110    FORMAT(' INVALID # OF COLUMNS IN ELECTRODE NET ',I2,
             *   '; PGM. TERMINATING')
0132  3210    FORMAT(' INVALID # OF ROWS IN ELECTRODE NET ',I2,
             *   '; PGM. TERMINATING')
0133          STOP
      C
      C --> INPUT 1 SET OF EEG READINGS FROM CURRENT TIME FRAME
      C
      C --> SELECT EVERY NTH EEGRECORD FROM THE DATA SOURCE
0134  1832    DO 9888  IR = 1, NTHREC
0135          READ(1,982,END=1557)  (EDATA(I),I=1,43)
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:02              PAGE 005

0136  9888  CONTINUE
0137  982   FORMAT(4E20.6)
      C     WRITE(6,2788)  (EDATA(I),I=1,48)
      C2788 FORMAT(' ////'THE FOLLOWING IS EEG DATA DIRECTLY FROM DISK'
      C     * //(12(5X,4E20.6/))/////)
      C
      C --> INITIALIZE ARRAY ELECTR TO 100.
0138        DO 8210  IR = 1, 5
0139        DO 8220 IC = 1, 5
0140         ELECTR(IR,IC) = 100.
0141  8220  CONTINUE
0142  8210  CONTINUE
      C
      C ---> INITIALIZE EEG DATA ARRAY FOR PROCESSING OF CURRENT DATA
0143        DO 983  K = 1, 48
0144        IC =  -2
0145        IR =   2
0146        DO 984  I = 1, 5, INCRI
0147         IF (LOCATE(K,1) .EQ. IR)  GO TO 987
0148         IR = IR + INCRI
0150  984    CONTINUE
0151        GO TO 983
0152  987   DO 986  J = 1, 5, INCRJ
0153         IF (LOCATE(K,2) .EQ. IC)  GO TO 988
0155         IC = IC + INCRJ
0156  986   CONTINUE
0157        GO TO 983
0158  988   I = LOCATE(K,1) - 1
0159        J = LOCATE(K,2) + 3
0160        ELECTR(I,J) = EDATA(K)
      C     WRITE(6,992) K, LOCATE(K,1), K, LOCATE(K,2), I, J, ELECTR(I,J)
      C992  FORMAT(' LOCATE(',I2,',1) = ',I2,'  LOCATE(',I2,',2) = ',
      C     * I2,'  INITIALLY POSITIONED AT ELECTR(',I2,',',I2,') = ',E20.7)
0161        IF (ELECTR(I,J) .GT. EMAX)  EMAX = ELECTR(I,J)
0163        IF (ELECTR(I,J) .LT. EMIN)  EMIN = ELECTR(I,J)
0165  983   CONTINUE
      C
      C --> COMPRESS THE SELECTED ELECTRODE POSITIONS TO ACTUAL SIZE OF ELECTR(*,*)
0166        IREAL = IROW + (IROW - 1) * (INCRI - 1)
0167        JREAL = ICOL + (ICOL - 1) * (INCRJ - 1)
0168        I = 1
0169        DO 8230  IR = 1, IREAL, INCRI
0170        J = 1
0171        DO 8240  IC = 1, JREAL, INCRJ
0172         ELECTR(I,J) = ELECTR(IR,IC)
0173         J = J + 1
0174  8240  CONTINUE
0175        I = I + 1
0176  8230  CONTINUE
      C --> FOLLOWING CODE OUTPUTS ELECTR(*,*) TO PRINTER FOR
      C     DEBUGGING PURPOSES
0177  545   CONTINUE
0178        DO 552  IR = 1, IROW
0179         DO 553  IC = 1, ICOL
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:02              PAGE 006

      C       WRITE(6,554)  IR, IC, ELECTR(IR, IC)
0180  553    CONTINUE
0181  552   CONTINUE
0182  554   FORMAT('   ELECTR(',I2,',',I2,') = ',E20.6)
      C
      C -----> INDICATE WHETHER OR NOT FACE-CENTERED POINT
      C       GENERATION IS TO BE PERFORMED
      C            BEFORE FIRST IMAGE PRODUCED:
      C             ICHNGE = 10 (INITIAL VALUE)
      C            AFTER FIRST IMAGE PRODUCED:
      C             ICHNGE = 0 (NO INTERPOLATION ROUTINE CHANGE ON CURRENT DATA)
      C             ICHNGE = 1 (CHANGE INTERPOLATION ROUTINE FROM PREVIOUS)
0183  991    IF (INPUT .GT. 0 .AND. NPASS .GT. 0)  GO TO 2000
0185         IF (ICHNGE .LT. 1) GO TO 2000
0187         WRITE(7,1000)
0188  1000   FORMAT(' PERFORM FACE-CENTERED POINT GENERATION ROUTINE ?'
            * /,5X,' 0 = NO'/,5X,' 1 = YES')
0189         READ(5,200) IFACE
0190  2000   IF (IFACE .NE. 1) GO TO 1005
0192         IF (INPUT .GT. 0 .AND. NPASS .GT. 0) GO TO 2001
0194         WRITE(7,1010)
0195  1010   FORMAT(' INPUT # OF COMPLETE GENERATIONS - FORMAT IS F3.1')
0196         READ(5,201)  SETS
0197  201    FORMAT(F3.1)
      C
      C --> EXECUTE FACE-CENTERED POINT GENERATION ROUTINE
0198  2001   CALL FACE
      C
      C -----> SET ELECTRODE ARRAY EQUAL TO FACE-CENTERED POINT
      C       GENERATION EXPAND ARRAY
0199         DO 1200  I = 1, ISIZE
0200         DO 1300 J = 1, JSIZE
0201          ELECTR(I,J) = EXPAND(I,J)
0202  1300   CONTINUE
0203  1200   CONTINUE
      C
      C --->  ADJUST  # OF POINTS BETW EACH GENERATED ROW & COLUMN POSITION
      C       RESULTING FROM FACE-CENTERED POINT GENERATION ROUTINE
      C       (APPROXIMATION ONLY)
0204         ISPACE = (ISPACE * (IROW - 1) / (ISIZE - 1))
      C
      C ---> SAVE THE ROW & CLUMN VALUES OF THE ORIGINAL ELECTRODE NET
0205         ITROW = IROW
0206         ITCOL = ICOL
      C
      C --->  RESET # OF ROWS AND COLUMNS IN ELECTRODE ARRAY TO # OF ROWS
      C       AND COLUMNS IN EXPANDED ARRAY
0207         IROW = ISIZE
0208         ICOL = JSIZE
      C
      C ----> INDICATE WHICH INTERPOLATION ROUTINE IS TO BE PERFORMED
0209  1005   IF (INPUT .GT. 0 .AND. NPASS .GT. 0)  GO TO 2100
0211         IF (ICHNGE .LT. 1) GO TO 2100
0213  1006   WRITE(7,1020)
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:02                PAGE 007

0214  1020  FORMAT(' INDICATE INTERPOLATION ROUTINE'/5X,' 1 = TERMINATE',
     *  'CURRENT PGM'/5X,' 2 = BILINEAR'/5X,' 3 = INVERSE-SQUARE',
     * ' RADIUS'/5X,' 4 = INVERSE RADIUS')
0215        READ(7,200) INTER
0216        IF (INTER .GE. 1 .AND. INTER .LE. 4) GO TO 7170
0218        WRITE(7,1023)
0219  1023  FORMAT(' INVALID INTERPOLATION SPECIFIED - TRY AGAIN?')
0220        GO TO 1006
      C --> SPECIFY MIN. RADIUS FACTOR FOR INVERSE & INVERSE SQUARE RADIUS
      C       INTERPOLATION ROUTINES
0221  7170  IF (INTER .LT. 3) GO TO 1022
      C - SMEAR = FRACTION BETWEEN 0 & .5, USED TO CALC. MIN.
      C                  RADIUS THAT ANY INTERPOLATION POINT IS TO ASSUME
      C                  FROM ANY ELECTRODE POSITION
0223        WRITE(7,7171)
0224  7171  FORMAT(' PLEASE INPUT MIN-NONZERO RADIUS FACTOR (F2.1)')
0225        READ(7,7172)  SMEAR
0226  7172  FORMAT(F6.4)
0227        WRITE(7,7173) SMEAR
0228  7173  FORMAT(' SMEAR = ',F6.3)
      C
      C -----> DETERMINE MAX ROW & COLUMN DISPLAY VALUES
0229  1022  MAXROW = (ISPACE * (IROW - 1)) + IROW
0230        MAXCOL = (ISPACE * (ICOL - 1)) + ICOL
      C
      C -----> CHECK ON DIMENSIONS OF DISPLAY ARRAY TO BE WITHIN LIMITS
0231        IF (MAXROW .LE. 449) GO TO 1415
0233        WRITE (6,1410) MAXROW
0234  1410  FORMAT(' ROWS TO BE DISPLAYED = ',I3,',  IS TOO LARGE AND',
     *  ' WILL BE RESET TO 449')
0235        MAXROW = 449
0236  1415  IF (MAXCOL .LE. 609) GO TO 1400
0238        WRITE(6,1420) MAXCOL
0239  1420  FORMAT(' COLUMNS TO BE DISPLAYED = ',I3,',  IS TOO LARGE AND',
     *  ' WILL BE RESET TO 609')
0240        MAXCOL = 609
      C
      C ----> CALCULATE BEGINNING ROW AND COLUMN DISPLAY POSITIONS --> CAUSE
      C       DISPLAY TO BE CENTERED ON SCREEN
0241  1400  IROWB = (479 - MAXROW) / 2
0242        ICOLB = (639 - MAXCOL) / 2
      C
      C -----> CALL REQUESTED INTERPOLATION ROUTINES
0243  2100  IF (INTER .EQ. 1) STOP
0245        IF (INTER .EQ. 2)   CALL INTERP(IROWB,ICOLB)
0247        IF (INTER .EQ. 3) CALL ISQRAD(IROWB,ICOLB)
0249        IF (INTER .EQ. 4) CALL INVRAD(IROWB,ICOLB)
      C
      C --> COUNT # OF IMAGES DISPLAYED
0251        NPASS = NPASS + 1
      C
      C --> RESET ROW & COLUMN VARIABLES TO THE ORIGINAL VALUES
0252        IF (ISPACE .LT. 1) GO TO 2101
0254        IROW = ITROW
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:02                PAGE 008

0255          ICOL = ITCOL
      C
      C --> BEGIN PROCEDURE TO MAKE CHANGES IN CURRENT DATA
      C       INPUT = 0       MANUAL OPERATION
      C       INPUT = 1       SEMI-AUTOMATIC OPERATION (DISK EEG DATA)
0256  2101    IF (INPUT .LE. 0)  GO TO 1609
      C
      C --> IPAUSE = 0      CONTINUOUS PGM. EXECUTION AFTER DISPLAY OUTPUT
      C     IPAUSE = 1      PAUSE UNTIL INSTRUCTED TO CONTINUE
0258          IF (IPAUSE .LE. 0)  GO TO 1832
0260  1611    WRITE(7,1608)
0261  1608    FORMAT(' OPERATOR, MAY I CONTINUE ?'/4X,' -1 = TERMINATE ME'/
             # 5X,' 0 = NO'/5X,' 1 = YES')
0262          READ(7,200)  IGO
0263          IF (IGO .GT. 0)  GO TO 1832
0265          IF (IGO .GE. 0)  GO TO 1611
0267          WRITE(7,1607)
0268  1607    FORMAT(' THE EEG POTENTIAL DISPLAY PGM. (       )',
             # ' HAS FINISHED')
0269          STOP
0270  1557    WRITE(7,1556)
0271  1556    FORMAT(' **** END OF FILE ****')
0272  1609    WRITE(7,1610)
0273  1610    FORMAT(' DO YOU WANT TO CONTINUE WITH THE CURRENT DISPLAY ?'/
             # ,5X,' 0 = NO'/,5X,' 1 = YES')
0274          READ(7,200) ICONT
0275          IF (ICONT .GT. 0)  GO TO 1612
0277          WRITE(7,1613)
0278  1613    FORMAT(' DO YOU WANT TO RESTART THE ENTIRE PROGRAM'/
             # 5X,' 0 = NO'/5X,' 1 = YES')
0279          READ(7,200)  ISTART
0280          IF (ISTART .GT. 0)  GO TO 1
0282          WRITE(7,1607)
0283          STOP
0284  1612    WRITE(7,1600)
0285  1600    FORMAT(' ANY POTENTIALS TO BE CHANGED ?'/,5X,' 0 = NO'
             #  /,5X,' 1 = YES')
0286          READ(7,200) ICHNGE
0287          IF (ICHNGE .EQ. 0)  GO TO 2550
0289          WRITE(7,1650)
0290  1650    FORMAT(' INPUT FORMAT FOR I, J, CHANGE FOLLOWS'/' I2,I2,F7.3')
0291  1900    READ(7,1700) I, J, CHANGE
0292  1700    FORMAT(I2,I2,F7.3)
0293          IF (I .LT. 1 .OR. J .LT. 1) GO TO 2400
0295          IF (I .GT. IROW .OR. J .GT. ICOL) GO TO 2200
0297          ELECTR(I,J) = CHANGE
0298          GO TO 1900
0299  2200    WRITE(7,2300)
0300  2300    FORMAT(' ERROR: # ROWS = ',I2,' # COLUMNS = ',I2/,5X,
             #  'EITHER I = ',I2,' OR J = ',I2,
             #  ' ARE TOO LARGE - IGNORING CHANGE, TRY AGAIN')
0301          GO TO 1900
0302  2400    WRITE(7,2500)
0303  2500    FORMAT(' POTENTIAL CHANGES COMPLETE')
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:02                PAGE 009

0304  2550  WRITE(7,2600)
0305  2600  FORMAT(' DO YOU WANT TO CHANGE THE INTERPOLATION ROUTINE ?'/
     *  ,5X,' 0 = NO'/,5X,' 1 = YES')
0306        READ(7,200) ICHNGE
0307        GO TO 1800
0308        STOP
0309        END
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:13              PAGE 001

0001          SUBROUTINE  INTERP(IROWB,ICOLB)
0002          COMMON  ERANGE(17), ELECTR(21,21), ISPACE, IROW,ICOL, MAXCOL,
             *  NUMLEV, IDEBUG, MAXROW, INTVAL(400)
0003          DIMENSION  TELECT(2,2)
0004          IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
             *   WRITE(6,123)
0006  123     FORMAT(' SUBROUTINE INTERP ENTERED')
0007          NSPACE = ISPACE + 1
0008          IBEGIN = 1
0009          ILAST = IROW - 1
0010          JLAST = ICOL - 1
      C ****** SET THE ROW POSITION VALUES - ISTART, IEND
0011          ISTART = 0
0012          IEND = NSPACE
      C
      C  LOOP THRU THE SUBROUTINE ONE TIME FOR EACH ROW
      C     KROW = CURRENT ROW BEING PROCESSED
      C
0013          DO 99999  NUMROW = 1, MAXROW
0014          KROW = NUMROW - 1
0015          IRESET = 1
      C
      C --> DETERMINE WHICH SET OF 4 ADJACENT ELECTRODES CONTAIN THE CURRENT POINT
      C ****** SET THE COLUMN POSITION VALUES - JSTART, JEND
0016          JSTART = 0
0017          JEND = NSPACE
0018          DO 100  I = IBEGIN, ILAST
0019          IF (KROW .GE.  ISTART .AND.  KROW .LE.  IEND) GO TO 200
0021          ISTART = IEND
0022          IEND = IEND + NSPACE
0023          IRESET = 1
0024  100     CONTINUE
0025          WRITE(7,1) IROW, ISTART, IEND, I
0026  1       FORMAT(' UNABLE TO DETERMINE CORRECT ROW INTERVAL:',/,'IROW=',
             *  I4,4X,'ISTART=',I4,4X,'IEND=',I4,4X,'I=',I2,4X,'PGM END')
0027          STOP
      C
      C --> BEGIN LOOP CONTROLLED BY KTEMP - CURRENT COLUMN
      C
0028  200     JBEGIN = 1
0029          IBEGIN = I
0030          DO 300  KTEMP = 1, MAXCOL
0031          KCOL = KTEMP - 1
0032          DO 400  J = JBEGIN, JLAST
0033           IF (KCOL .GE.  JSTART .AND.  KCOL .LE.  JEND) GO TO 500
0035           IRESET = 1
0036           JSTART = JEND
0037           JEND = JEND + NSPACE
0038  400     CONTINUE
0039          WRITE(7,2) KCOL, JSTART, JEND, J
0040  2       FORMAT(' UNABLE TO DETERMINE CORRECT COLUMN INTERVAL:',/,
             *    ' KCOL=',I4,4X,'JSTART=',I4,4X,'JEND=',I4,4X,'J=',I2,4X,
             *    'PGM END')
0041          STOP
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:13                PAGE 002

0042  500    IF (IRESET .NE. 1) GO TO 600
0044          JBEGIN = J
      C
      C --> SET THE VALUES OF THE TEMPORARY ELECTRODE ARRAY
      C
0045           J1 = J + 1
0046           I1 = I + 1
0047           TELECT(1,1) = ELECTR(I,J)
0048           TELECT(1,2) = ELECTR(I,J1)
0049           TELECT(2,1) = ELECTR(I1,J)
0050           TELECT(2,2) = ELECTR(I1,J1)
0051           IRESET = 0
0052          IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
             *WRITE(6,66) TELECT(1,1),TELECT(1,2),TELECT(2,1),TELECT(2,2)
0054  66      FORMAT(' TELECT(1,1)=',F7.3,'  TELECT(1,2)=',F7.3,
             *   '  TELECT(2,1)=',F7.3,'  TELECT(2,2)=',F7.3)
      C
      C --> DETERMINE THE RELATIVE POSITION OF THE CURRENT POINT AND THE
      C     ASSOCIATED POTENTIAL CALCULATION SCHEME
      C
      C ****** DETERMINE IF CURRENT POINT IS STRICTLY WITHIN PERIMETER
      C         OF TEMPORARY ARRAY - BRANCH TO 1200 IF YES
      C
0055  600    IF  (KROW .GT. ISTART .AND. KROW .LT. IEND .AND.
            *  KCOL .GT. JSTART .AND. KCOL .LT. JEND)  GO TO 1010
0057           IVAL = 1
      C ****** DETERMINE WHERE THE CURRENT POINT IS LOCATED ON
      C         PERIMETER OF TEMPORARY ARRAY - IF NOT DETERMINED, THEN
      C         OUTPUT ERROR MESSAGE AND TERMINATE PGM.
      C
0058          ITEMPS = ISTART
0059          JTEMPS = JSTART
0060          ITEMPE = IEND
0061          JTEMPE = JEND
0062          JTEMP = JSTART
0063          DO 700  II = 1, 2
0064           IF (KROW .EQ. ITEMPS) IVAL = 2
0066           IF (KCOL .EQ. JTEMP)  IVAL = 3
0068           DO 800  IJ = 1, 2
0069            IF (KROW .EQ. ITEMPS .AND. KCOL .EQ. JTEMPS) GO TO 4000
0071            JTEMPS = JTEMPE
0072  800      CONTINUE
0073          JTEMPS = JTEMP
0074          JTEMP = JTEMPE
0075          ITEMPS = ITEMPE
0076         GO TO  (700, 1000, 2000) ,  IVAL
0077  700    CONTINUE
0078          WRITE(6,7) KROW, KCOL, ITEMPS, ITEMPE, JTEMPS,JTEMPE
0079  7       FORMAT(' ERROR:  CANNOT DETERMINE LOCATION OF CURRENT POINT'/
             *    ' KROW = ',I3,',  KCOL = ',I3,',  ITEMPS = ',I3,',  ITEMPE = ',
             *    I3,',  JTEMPS = ',I3,',  JTEMPE = ',I3)
0080           STOP
      C
      C --> THE FOLLOWING IS EXECUTED WHEN THE CURRENT POINT IS ON THE
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:13                PAGE 003

      C      INTERIOR OF THE TEMPORARY ARRAY
      C
0081  1010    X1 = KROW
0082  1100    IF (X1 .LE. NSPACE) GO TO 900
0084          X1 = X1 - NSPACE
0085          GO TO 1100
0086  900     Y1 = KCOL
0087  1300    IF (Y1 .LE. NSPACE) GO TO 1200
0089          Y1 = Y1 - NSPACE
0090          GO TO 1300
0091  1200    X1DX2 = X1 / NSPACE
0092          Y1DY2 = Y1 / NSPACE
0093          XMULTY = X1DX2 * Y1DY2
0094          EPOTEN = TELECT(1,1) * (1 - X1DX2 - Y1DY2 + XMULTY)
     *                 + TELECT(1,2) * (Y1DY2 - XMULTY)
     *                 + TELECT(2,1) * (X1DX2 - XMULTY)
     *                 + TELECT(2,2) * XMULTY
0095          GO TO 9000
      C
      C --> THE FOLLOWING CODE IS EXECUTED WHEN THE CURRENT POINT IS ON THE
      C   FIRST OR LAST ROW OF THE TEMPORARY ARRAY
      C
0096  1000    Y1 = KCOL
0097  1500    IF (Y1 .LT. NSPACE) GO TO 1400
0099          Y1 = Y1 - NSPACE
0100          GO TO 1500
0101  1400    EPOTEN = TELECT(II,1)
     *                 + ((TELECT(II,2) - TELECT(II,1)) * (Y1 / NSPACE))
0102          GO TO 9000
      C
      C --> THE FOLLOWING CODE IS EXECUTED WHEN THE CURRENT POINT IS ON THE
      C     FIRST OR LAST COLUMN OF THE TEMPORARY ARRAY
      C
0103  2000    X1 = KROW
0104  1700    IF (X1 .LT. NSPACE) GO TO 1600
0106          X1 = X1 - NSPACE
0107          GO TO 1700
0108  1600    EPOTEN = TELECT(1,II)
     *                 + ((TELECT(2,II) - TELECT(1,II)) * (X1 / NSPACE))
0109          GO TO 9000
      C
      C --> THE FOLLOWING CODE IS EXECUTED ONLY WHEN THE CURRENT POINT IS
      C     LOCATED AT ONE OF THE TEMPORARY ELECTRODE POSITIONS
      C
0110  4000    EPOTEN = TELECT(II,IJ)
      C
      C --> THIS CODE EQUATES THE CURRENT POINT POTENTIAL VALUE TO THE
      C     CORRESPONDING GRAY LEVEL VALUE
      C
0111  9000    DO 1800  IM = 1, NUMLEV
0112            IN = IM + 1
0113            IF (EPOTEN .GE. ERANGE(IM) .AND. EPOTEN .LT. ERANGE(IN))
     *                                                          GO TO 1900
0115  1800    CONTINUE
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:13              PAGE 004

0116          IM = NUMLEV
0117          IF (EPOTEN .GE. ERANGE(IN) .AND. EPOTEN .LT.
             * ERANGE(IN) + .1)  GO TO 1900
0119          IM = 0
0120          IF (IDEBUG .EQ. 1 .OR. IDEBUG .EQ. 3)
             *  WRITE(7,3) EPOTEN, IVAL
0122  3        FORMAT(' UNABLE TO DETERMINE A VALID GRAY LEVEL FOR EPOTEN = ',
             #     ,E20.7,',  IVAL = ',I3,' ; PGM END')
0123  1900    INTVAL(KTEMP) = IM
0124          IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
             #WRITE(6,60) KTEMP,INTVAL(KTEMP), EPOTEN, KCOL, KROW
0126  60      FORMAT(' INTVAL(',I4,') = ',I4,  5X,'EPOTEN = ',F7.3,
             # 5X,' KCOL=',I3,5X,' KROW=',I3)
0127  300     CONTINUE
      C
      C --> INSERT 99 AT END OF CALCULATED GRAY LEVELS TO INDICATE END OF DATA
0128          INTVAL(KTEMP) = 99
0129          IF (IDEBUG .EQ. 2  .OR.  IDEBUG .EQ. 3)
             *   WRITE(6,61) KROW, MAXCOL, IROWB, ICOLB
0131  61      FORMAT(' KROW = ',I3,5X,'MAXCOL = ',I3,5X,'IROWB = ',I3,
             #    5X,'ICOLB = ',I3)
0132          CALL DISPLY(KROW,MAXCOL,IROWB,ICOLB,INTVAL)
0133  99999 CONTINUE
0134          RETURN
0135          END
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:20              PAGE 001

0001          SUBROUTINE ISQRAD(IROWD,ICOLB)
0002          COMMON  ERANGE(17), ELECTR(21,21),ISPACE, IROW, ICOL, MAXCOL,
             * NUMLEV, IDEBUG, MAXROW, INTVAL(400)
0003          COMMON   /BLOCK3/EVAL(400), SMEAR
      C       WRITE(7,123)
      C123    FORMAT(' SUBROUTINE ISQRAD  ENTERED')
      C
      C -----> INVERSE-SQUARE RADIUS INTERPOLATION ROUTINE
      C
      C       EACH EXECUTION OF THIS ROUTINE COMPUTES THE INTERPOLATION
      C       VALUE FOR EVERY POINT FOR AN ENTIRE LINE.   THE DISPLAY IS
      C       CREATED ONE LINE AT A TIME AFTER EACH IS COMPUTED.
      C
      C ---> DISTANCE BETWEEN POINTS IS RELATIVE TO INPUT SCALE VALUE 'SCALE'
      C
      C *** VARIABLES:
      C       YDIST - VERTICLE DISTANCE COMPONENT
      C       XDIST - HORIZONTAL DISTANCE COMPONENT
      C       IPOINT - CURRENT ROW NUMBER
      C       JPOINT - CURRENT COLUMN NUMBER
      C       MAXCOL - MAX. NUMBER ELEMENTS / ROW
      C       EVAL(*) - INTERPOLATED VALUES
      C       SCALE - DISTANCE SCALE FACTOR FROM POINT TO POINT
      C
0004          IREPLC = 1
0005          IREROW = 1
0006          INCREM = ISPACE + 1
0007          RSMINV = 1 / ((INCREM * SMEAR) ** 2)
0008          DO 100  IPOINT = 1, MAXROW
      C
      C  --->  ZERO FILL POTENTIAL ARRAY  EVAL(400)
0009          DO 10  I = 1, MAXCOL
0010           EVAL(I) = 0.
0011           INTVAL(I) = 0
0012    10    CONTINUE
0013          DO 200  JPOINT = 1, MAXCOL
0014           IPOS = 1
0015           WT = 0.0
0016           DO 300  I = 1, MAXROW, INCREM
0017            JPOS = 1
      C
      C --> CALCULATE X, Y DISTANCES
0018            DO 400  J = 1, MAXCOL, INCREM
      C
      C ---> TEST FOR 'ELECTRODE' NOT TO BE INCLUDED IN CALCULATIONS -
      C       RESULTING FROM FACE SUBROUTINE
0019              IF (ELECTR(IPOS,JPOS) .GT. 999.) GO TO 350
      C
      C - > COMPUTE DISTANCE FROM INTERPOLATION POINT TO CURRENT
      C       ELECTRODE
0021              YDIST = FLOAT(IPOINT - I)
0022              XDIST = FLOAT(JPOINT - J)
0023              IF (YDIST .EQ. 0.   .AND.   XDIST .EQ. 0.)   YDIST = 1
      C
```

```
FORTRAN IV       V01C-03A   WED 22-JUN-77 07:17:20                PAGE 002

      C --> COMPUTE INTERPOLATION VALUE
0025            RSQINV = 1 / (YDIST ** 2 + XDIST ** 2)
0026            IF (RSMINV .LT. RSQINV) RSQINV = RSMINV
0028            WT = WT + RSQINV
0029            EVAL(JPOINT) = EVAL(JPOINT) + ELECTR(IPOS,JPOS) *
     *       RSQINV
0030          IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
     *  WRITE(6,1001) RSQINV, IPOS, JPOS, MAXROW, MAXCOL, INCREM,
     *    IPOINT, JPOINT, EVAL(JPOINT), I, J, ELECTR(IPOS,JPOS), WT
0032  350       JPOS = JPOS + 1
0033  400      CONTINUE
0034           IPOS = IPOS + 1
0035  300     CONTINUE
0036          EVAL(JPOINT) = EVAL(JPOINT) / WT
0037  1001    FORMAT(' RSQINV = ',F7.2,'  IPOS = ',I2,'  JPOS = ',I2,
     *'    MAXROW = ', I3,'  MAXCOL = ',I3,' INCREM = ',I3,
     *  '   IPOINT = ',I3,'  JPOINT = ',I3,'  EVAL(JPOINT) = ',F7.3/
     *  '   I = ',I3,'  J = ',I3,'  ELECTR(IPOS,JPOS) = ',F7.3,
     *  '   WT = ',F7.5)
0038  200     CONTINUE
0039          IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
     *  WRITE(6,1010)  JPOINT, IPOINT, (EVAL(I),I=1,MAXCOL)
0041  1010    FORMAT(' JPOINT = ',I3,5X,'EVAL(*) FOR ROW #',I3/23(' ',18F7.3/))
      C
      C -----> AFTER COMPUTING VALUES FOR EACH ROW, CHECK TO SEE IF ORIGINAL
      C      ELECTRODE VALUES SHOULD BE INSERTED - IF SO, THEN DO AND
      C      INCREMENT IREPLC
0042          IF (IPOINT .NE. IREPLC)  GO TO 450
0044          JK = 1
0045          DO 475  J = 1, MAXCOL, INCREM
0046           EVAL(J) = ELECTR(IREROW,JK)
0047           JK = JK + 1
0048  475     CONTINUE
0049          IREPLC = IREPLC + INCREM
0050          IREROW = IREROW + 1
      C
      C -----> AFTER EACH ROW IS COMPLETED - SET EACH CORRESPONDING POSITION IN
      C      INTVAL EQUAL TO THE ASSOCIATED GRAY LEVEL VALUE
0051  450     DO 500  I = 1, MAXCOL
0052           DO 600  IM = 1, NUMLEV
0053            IN = IM + 1
0054            IF (EVAL(I) .GE. ERANGE(IM) .AND. EVAL(I) .LT.
     *          ERANGE(IN))  GO TO 700
0056  600      CONTINUE
0057           IF (EVAL(I) .GE. ERANGE(IN) .AND. EVAL(I) .LE. (ERANGE(IN) + 1.))
     *         GO TO 700
0059          IF (EVAL(I) .LT. ERANGE(1))  INTVAL(I) = 1
0061          IF (EVAL(I) .GT. ERANGE(IN))  INTVAL(I) = NUMLEV
0063          IM = 0
0064          IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
     *     WRITE(6,800) I, EVAL(I), IPOINT
0066  800     FORMAT(' UNABLE TO DETERMINE EVAL(',I3,') = ',F7.3,
     *       ' IN ROW ',I3)
0067           GO TO 500
```

```
FORTRAN IV       V01C-03A   WED 22-JUN-77 07:17:20                PAGE 003

0068  700     INTVAL(1) = 1M
0069  500    CONTINUE
0070           INTVAL(JPOINT) = 99
0071          IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
             *   WRITE(6,1000) IPOINT, (EVAL(K), INTVAL(K), K=1,MAXCOL)
0073  1000   FORMAT(' EVAL(*), INTVAL(*) FOR ROW #',I3/
             *   250(' ',F7.3,5X,I2/))
      C       WRITE(7,124)
      C124    FORMAT(' 1500RAD EXIT, ENTERING DISPLY')
0074          CALL DISPLY(IPOINT,MAXCOL,IROWB,ICOLB,INTVAL)
0075  100     CONTINUE
0076          RETURN
0077          END
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:26              PAGE 001

0001          SUBROUTINE FACE
0002          COMMON ERANGE(17), ELECTR(21,21), ISPACE,IROW, ICOL,
             *MAXCOL, NUMLEV, IDEBUG, MAXROW, INTVAL(400)
0003          COMMON   /BLOCK1/EXPAND(21,21), ISETS, SETS, ISIZE, JSIZE, JDEBUG
0004          LOGICAL  BETWEN
0005          WRITE(7,123)
0006   123    FORMAT(' SUBROUTINE FACE  ENTERED')
       C
       C  SUBROUTINE FACE PERFORMS THE FACE-CENTERED POINT GENERATION, CONSISTS OF
       C  2 OPERATIONS:   1-THE VALUE AT THE CENTER OF EACH SET OF 4 ADJACENT
       C   ORIGINAL POINTS IS CALCULATED - SUM VALUE OF EACH POINT
       C   & DIVIDE BY 4.   2-THE POINTS ON BOTH, ABOVE & BELOW OF EACH NEW POINT
       C   ARE ALSO CACULATED.  THE RESULT IS AN ARRAY WITH (N-1) ADDITIONAL
       C   ROWS  & COLUMNS.   THE PROCEDURE COULD BE PERFORMED AS MANY TIMES
       C   AS DESIRED - UNTIL THE REQUIRED DENSITY IS ACHIEVED.
       C
       C     INPUT TO THE ROUTINE:
       C     ELECTR(21,21) - ARRAY CONTAINING THE ORIGINAL ELECTRODE POINT VALUES
       C     IROW - # OF ROWS IN THE ORIGINAL ELECTRODE ARRAY
       C     ICOL - # OF COLUMNS IN THE ORIGINAL ELECTRODE ARRAY
       C     ISETS - # OF TIMES THE PROCEDURE IS TO BE PERFORMED
       C
       C     OUTPUT FROM THE ROUTINE:
       C     EXPAND(*,*) - THE EXPANDED (HIGHER DENSITY) ARRAY
       C                   CONTAINING ALL THE ORIGINAL AND CALCULATED POINT VALUES
       C
       C     OTHER IMPORTANT VARIABLES:
       C     ISKIP - THE # OF POINTS INSERTED BETWEEN EACH 2 ROW OR COLUMN
       C               POINTS
       C     NTIMES - # OF TIMES THE PROCEDURE IS EXECUTED
       C
       C ---->  REMAINING VARIABLES ARE DEFINED IN THE SECTIONS THEY APPEAR
       C
       C
       C ****** CURRENT MAXIMUM # OF GENERATIONS ALLOWED IS 2 FOR AN ORIGINAL 6X6
       C        ELECTRODE ARRAY.   IF RESULTING EXPANDED ARRAY IS LARGER THAN ALLOWED
       C        A WARNING IS OUTPUT AND A REDUCTION OF ISETS IS PERFORMED
       C        UNTIL AN ACCEPTABLE VALUE IS FOUND.   THE OPERATOR IS
       C        INFORMED OF THE NEW VALUE OF ISETS.
       C
       C
       C --> SET DEBUG FLAG
0007          ISETS = SETS + .5
0008          GO TO 400
0009   300    WRITE(7,500) SETS
0010   500    FORMAT(' THE # OF GENERATIONS, ', F3.1,', IS TOO LARGE',
              *' AND WILL BE REDUCED BY 1')
0011          ISETS = ISETS - 1
0012          SETS = SETS - 1.
0013   400    ISKIP = 0
       C -----> COMPUTE TOTAL # OF ELEMENTS INSERTED BTWN POINTS IN ORIGINAL ARRAY
0014          DO 600  K = 1, ISETS
0015          ISKIP = (2 * ISKIP) + 1
0016   600    CONTINUE
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:26              PAGE 002

0017          IPLUS = ISKIP + 1
      C ------> DETERMINE IF EXPANSION ARRAY IS LARGER THAN ALLOWED
      C           ISIZE - # OF RESULTING ROWS IN EXPANDED ARRAY
      C           JSIZE - # OF RESULTING COLUMNS IN EXPANDED ARRAY
      C
0018          ISIZE = ((IROW - 1) * ISKIP) + IROW
0019          JSIZE = ((ICOL - 1) * ISKIP) + ICOL
0020          IF (ISIZE .GT. 21 .OR. JSIZE .GT. 21) GO TO 300
      C ------> DIMENSIONS OF EXPANDED ARRAY ARE ACCEPTABLE
0022          WRITE(7,610) SETS
0023  610     FORMAT(' THE # OF GENERATIONS PERFORMED IS ',F3.1)
      C
      C ------> ZERO FILL EXPAND(*,*)
      C
0024          DO 606  I = 1, 21
0025          DO 607  J = 1, 21
0026           EXPAND(I,J) = 999.99
0027  607     CONTINUE
0028  606     CONTINUE
      C
      C ------> POSITION ORIGINAL ARRAY IN THE EXPANDED ARRAY
      C       IK,JK - ROW, COLUMN (RESPECTIVELY) POSITIONS IN THE EXPANDED ARRAY
      C       I, J - ORIGINAL ARRAY ROW, COL VALUES PLACED AT IK,JK IN
      C                 EXPANDED ARRAY
      C       IPLUS - INCREMENT FROM POINT TO POINT IN EXPANDED ARRAY
0029          IK = 1
0030          DO 700  I = 1, IROW
0031          JK = 1
0032           DO 800  J = 1, ICOL
0033            EXPAND(IK,JK) = ELECTR(I,J)
0034            JK = JK + IPLUS
0035  800        CONTINUE
0036          IK = IK + IPLUS
0037  700     CONTINUE
0038          IF (IDEBUG .NE. 2 .AND. IDEBUG .NE. 3) GO TO 720
0040          WRITE(6,703)
0041  703     FORMAT(' CURRENT  EXPAND ARRAY')
0042          DO 701  ITEST = 1, 21
0043          WRITE(6,702) (EXPAND(ITEST,JTEST),JTEST=1,21)
0044  702     FORMAT(' ',21(F5.2,1X))
0045  701     CONTINUE
      C
      C -----> COMPUTE AND POSITION THE CENTER OF EACH SET OF 4 POINTS
      C          INEW - ROW POSITION OF CENTER POINT
      C          JNEW - COLUMN POSITION OF CENTER POINT
      C        IEND - # OF ROW CENTER POINTS TO BE CALCULATED (1 LESS
      C               THAN # CALCULATED IN PREVIOUS GENERATION)
      C        JEND - # OF COLUMN CENTER POINTS TO BE
      C               CALCULATED (1 LESS THAN # CALCULATED IN
      C               PREVIOUS GENERATION)
0046  720     IEND = ISIZE - IPLUS
0047          JEND = JSIZE - IPLUS
0048          DO 99999  NTIMES = 1, ISETS
0049          IK = 1
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:26                PAGE 003

0050          DO 900  I = 1, IEND, IPLUS
0051           JK = 1
0052           IKLAST = IK + IPLUS
0053            DO 1000  J = 1, JEND, IPLUS
0054             JKLAST = JK + IPLUS
0055             INEW = (IK + IKLAST) / 2
0056             JNEW = (JK + JKLAST) / 2
0057             EXPAND(INEW,JNEW) = (EXPAND(IK,JK) + EXPAND(IK,JKLAST)
     *           + EXPAND(IKLAST,JK) + EXPAND(IKLAST,JKLAST)) / 4
0058            JK = JKLAST
0059  1000      CONTINUE
0060          IK = IKLAST
0061  900     CONTINUE
0062          IF (IDEBUG .NE. 2 .AND. IDEBUG .NE. 3) GO TO 920
0064          WRITE(6,903)
0065  903     FORMAT(' CENTER POINTS  EXPAND ARRAY')
0066          DO 901  ITEST = 1, 21
0067          WRITE(6,902) (EXPAND(ITEST,JTEST),JTEST=1,21)
0068  902     FORMAT(' ',21(F5.2,1X))
0069  901     CONTINUE
      C
      C ---> TEST FOR LAST GENERATION TO BE ONLY FOR CENTER OF EACH SET OF 4
      C       ADJACENT 'ELECTRODES'
0070          SETS = SETS - 1.0
0071          IF (SETS .LT. 0.)  RETURN
      C
      C **** COMPUTE THE REMAINING ELEMENTS OF EXPAND IN ORDER TO COMPLETELY FILL
      C       THE ARRAY
      C
      C -----> NEWINC = INCREMENT REQUIRED TO GET FROM CURRENT CALCULATED ELEMENT
      C                 TO THE NEXT
0073  920     NEWINC = IK - INEW
0074          IF (NEWINC .GT. 0) GO TO 1100
0076          IF (IDEBUG .EQ. 1 .OR. IDEBUG .EQ. 3) WRITE(7,1200) NEWINC
0078  1200    FORMAT(' NEWINC=',I2,' AND PGM IS TRYING TO EXECUTE -- STOP')
0079          STOP
      C -----> SECTION 1:  COMPUTE THE PERIMETER VALUES - ONLY 3 NEIGHBORING
      C         VALUES ARE USED INSTEAD OF THE USUAL 4
0080  1100    INCREM = 2 * NEWINC
      C -----> COMPUTE FIRST AND LAST ROW PERIMETER VALUES
0081          INEXT = I + NEWINC
0082          IPREV = ISIZE - NEWINC
0083          JBEGIN = 1 + NEWINC
0084          IEND = ISIZE - NEWINC
0085          JEND = JSIZE - NEWINC
0086          DO 1600 JVAL = JBEGIN, JEND, INCREM
0087          JPREV = JVAL - NEWINC
0088          JNEXT = JVAL + NEWINC
      C **** FIRST ROW
0089          EXPAND(1,JVAL) = (EXPAND(1,JPREV) + EXPAND(1,JNEXT) +
     *                          EXPAND(INEXT,JVAL)) / 3
      C **** LAST ROW
0090          EXPAND(ISIZE,JVAL) = (EXPAND(ISIZE,JPREV) + EXPAND(ISIZE,JNEXT)
     *                            + EXPAND(IPREV,JVAL)) /3
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:26                PAGE 004

0091  1600  CONTINUE
      C
      C ----> COMPUTE FIRST AND LAST COLUMN PERIMETER VALUES
0092        JNEXT = 1 + NEWINC
0093        JPREV = JSIZE - NEWINC
0094        IBEGIN = 1 + NEWINC
0095        DO 1500 IVAL = IBEGIN, IEND, INCREM
0096        IPREV = IVAL - NEWINC
0097        INEXT = IVAL + NEWINC
      C **** FIRST COLUMN
0098        EXPAND(IVAL,1) = (EXPAND(IPREV,1) + EXPAND(INEXT,1) +
           *                              EXPAND(IVAL,JNEXT)) / 3
      C **** LAST COLUMN
0099        EXPAND(IVAL,JSIZE) = (EXPAND(IPREV,JSIZE) + EXPAND(INEXT,JSIZE)
           *                           + EXPAND(IVAL,JPREV)) / 3
0100  1500  CONTINUE
0101        IF (IDEBUG .NE. 2 .AND. IDEBUG .NE. 3) GO TO 1550
0103        WRITE(6,1303)
0104  1303  FORMAT(' PERIMETER VALUES  EXPAND ARRAY')
0105        DO 1301  ITEST = 1, 21
0106        WRITE(6,1302) (EXPAND(ITEST,JTEST),JTEST=1,21)
0107  1302  FORMAT(' ',21(F5.2,1X))
0108  1301  CONTINUE
      C -----> SECTION 2:  COMPUTE ALL THE SECOND LEVEL INTERNAL POINTS
      C **** NO INTERNAL POINTS TO CALCULATE WHEN ORIGINAL ARRAY
      C        HAS ONLY 2 COLUMNS AND NTIMES = 1
0109  1550  IF (ICOL .EQ. 2 .AND. NTIMES .EQ. 1) GO TO 2200
0111        BETWEN = .TRUE.
0112        IVALE = IK - NEWINC
0113        JVALE = JK - NEWINC
0114        IPOSIT = NEWINC + 1
      C ----> EVERY TIME THROUGH THE ROUTINE THE BEGINNING COLUMN VALUE
      C        IS RESET, DEPENDING ON THE VALUE OF BETWEN (TRUE OR FALSE)
0115        DO 1800  IVAL = IPOSIT, IVALE, NEWINC
0116        INEXT = IVAL + NEWINC
0117        IPREV = IVAL - NEWINC
      C ----> SET VALUE OF JSTART DEPENDING UPON BETWEN (TRUE OR FALSE)
0118        IF (.NOT. BETWEN) GO TO 1900
0120        JSTART = (2 * NEWINC) + 1
0121        BETWEN = .FALSE.
0122        GO TO 2000
0123  1900  JSTART = NEWINC + 1
0124        BETWEN = .TRUE.
0125  2000   DO 2100 JVAL = JSTART, JVALE, INCREM
0126          JPREV = JVAL - NEWINC
0127          JNEXT = JVAL + NEWINC
0128          EXPAND(IVAL,JVAL) = (EXPAND(IVAL,JPREV) + EXPAND(IVAL,JNEXT)
           *     + EXPAND(IPREV,JVAL) + EXPAND(INEXT,JVAL)) / 4.
0129  2100   CONTINUE
0130  1800     CONTINUE
0131  2200  IPLUS = IPLUS /2
0132        IF (IDEBUG .NE. 2 .AND. IDEBUG .NE. 3) GO TO 1820
0134        WRITE(6,1803)  NTIMES
0135  1803  FORMAT(' GENERATION ',I2,'  COMPLETE   EXPAND ARRAY')
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:26                PAGE 005

0136          DO 1801  ITEST = 1, 21
0137          WRITE(6,1802)  (EXPAND(ITEST,JTEST),JTEST=1,21)
0138    1802  FORMAT(' ',21(F5.2,1X))
0139    1801  CONTINUE
0140    1820  IEND = ISIZE - IPLUS
0141          JEND = JSIZE - IPLUS
0142   99999 CONTINUE
0143          RETURN
0144          END
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:34                PAGE 001

0001          SUBROUTINE INVRAD(IROWB,ICOLB)
0002          COMMON  CRANGE(17), ELECTR(21,21),ISPACE, IROW, ICOL, MAXCOL,
             * NUMLEV, IDEBUG, MAXROW, INTVAL(400)
0003          COMMON   /BLOCK3/EVAL(400), SMEAR
      C       WRITE(7,123)
      C123    FORMAT(' SUBROUTINE INVRAD  ENTERED')
      C
      C -----> INVERSE-SQUARE RADIUS INTERPOLATION ROUTINE
      C
      C       EACH EXECUTION OF THIS ROUTINE COMPUTES THE INTERPOLATION
      C       VALUE FOR EVERY POINT FOR AN ENTIRE LINE.  THE DISPLAY IS
      C       CREATED ONE LINE AT A TIME AFTER EACH IS COMPUTED.
      C
      C --> DISTANCE BETWEEN POINTS IS RELATIVE TO INPUT SCALE VALUE 'SCALE'
      C
      C *** VARIABLES:
      C       YDIST - VERTICLE DISTANCE COMPONENT
      C       XDIST - HORIZONTAL DISTANCE COMPONENT
      C       IPOINT - CURRENT ROW NUMBER
      C       JPOINT - CURRENT COLUMN NUMBER
      C       MAXCOL - MAX.  NUMBER ELEMENTS / ROW
      C       EVAL(*) - INTERPOLATED VALUES
      C       SCALE - DISTANCE SCALE FACTOR FROM POINT TO POINT
      C
0004          IREPLC = 1
0005          IREROW = 1
0006          INCREM = ISPACE + 1
0007          RINVMX = 1 / (INCREM * SMEAR)
0008          DO 100  IPOINT = 1, MAXROW
      C
      C ---->  ZERO FILL POTENTIAL ARRAY  EVAL(400)
0009          DO 10  I = 1, MAXCOL
0010           EVAL(I) = 0.
0011           INTVAL(I) = 0
0012    10    CONTINUE
0013          DO 200  JPOINT = 1, MAXCOL
0014           IPOS = 1
0015           WT = 0.0
0016           DO 300  I = 1, MAXROW, INCREM
0017            JPOS = 1
      C
      C --> CALCULATE X, Y DISTANCES
0018            DO 400  J = 1, MAXCOL, INCREM
      C
      C ----> TEST FOR CURRENT 'ELECTRODE' NOT TO BE INCLUDED IN CALCULATION -
      C       RESULTING FROM FACE SUBROUTINE
0019          IF (ELECTR(IPOS,JPOS) .GT. 999.)  GO TO 350
      C
      C  -> COMPUTE DISTANCE FROM INTERPOLATION POINT TO CURRENT
      C       ELECTRODE
0021            YDIST = FLOAT(IPOINT - I)
0022            XDIST = FLOAT(JPOINT - J)
0023            IF (YDIST .EQ. 0.   .AND.   XDIST .EQ. 0.)   YDIST = 1
      C
```

```
FORTRAN IV       V01C-03A   WED 22-JUN-77 07:17:34                PAGE 002

      C --> COMPUTE INTERPOLATION VALUE
0025              RADINV = 1 / SQRT(YDIST ** 2 + XDIST ** 2)
0026              IF (RINVMX .LT. RADINV) RADINV = RINVMX
0028              WT = WT + RADINV
0029              EVAL(JPOINT) = EVAL(JPOINT) + ELECTR(IPOS,JPOS) *
     *       RADINV
0030           IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
     *  WRITE(6,1001) RADINV, IPOS, JPOS, MAXROW, MAXCOL, INCREM,
     *   IPOINT, JPOINT, EVAL(JPOINT), I, J, ELECTR(IPOS,JPOS), WT
0032  350         JPOS = JPOS + 1
0033  400       CONTINUE
0034            IPOS = IPOS + 1
0035  300       CONTINUE
0036           EVAL(JPOINT) = EVAL(JPOINT) / WT
0037  1001     FORMAT (' RADINV = ',F7.2,'  IPOS = ',I2,'  JPOS = ',I2,
     *'   MAXROW = ', I3,'  MAXCOL = ',I3,' INCREM = ',I3,
     *  '   IPOINT = ',I3,'  JPOINT = ',I3,'  EVAL(JPOINT) = ',F7.3/
     *  '   I = ',I3,'  J = ',I3,'  ELECTR(IPOS,JPOS) = ',F7.3,
     *  '   WT = ',F7.5)
0038  200     CONTINUE
0039           IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
     *  WRITE(6,1010)  JPOINT, IPOINT, (EVAL(I),I=1,MAXCOL)
0041  1010    FORMAT(' JPOINT = ',I3,5X,'EVAL(*) FOR ROW #',I3/23(' ',18F7.3/))
      C
      C ----> AFTER COMPUTING VALUES FOR EACH ROW, CHECK TO SEE IF ORIGINAL
      C     ELECTRODE VALUES SHOULD BE INSERTED - IF SO, THEN DO AND
      C     INCREMENT IREPLC
0042           IF (IPOINT .NE. IREPLC)  GO TO 450
0044           JK = 1
0045           DO 475  J = 1, MAXCOL, INCREM
0046            EVAL(J) = ELECTR(IREROW,JK)
0047            JK = JK + 1
0048  475     CONTINUE
0049           IREPLC = IREPLC + INCREM
0050           IREROW = IREROW + 1
      C
      C ----> AFTER EACH ROW IS COMPLETED - SET EACH CORRESPONDING POSITION IN
      C     INTVAL EQUAL TO THE ASSOCIATED GRAY LEVEL VALUE
0051  450     DO 500  I = 1, MAXCOL
0052           DO 600  IM = 1, NUMLEV
0053            IN = IM + 1
0054            IF (EVAL(I) .GE. ERANGE(IM) .AND. EVAL(I) .LT.
     *          ERANGE(IN))  GO TO 700
0056  600      CONTINUE
0057            IF (EVAL(I) .GE. ERANGE(IN) .AND. EVAL(I) .LE. (ERANGE(IN) + 1.))
     *         GO TO 700
0059           IF (EVAL(I) .LT. ERANGE(1))  INTVAL(I) = 1
0061           IF (EVAL(I) .GT. ERANGE(IN))  INTVAL(I) = NUMLEV
0063           IM = 0
0064           IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
     *     WRITE(6,800) I, EVAL(I), IPOINT
0066  800      FORMAT(' UNABLE TO DETERMINE EVAL(',I3,') = ',F7.3,
     *       ' IN ROW ',I3)
0067            GO TO 500
```

```
FORTRAN IV      V01C-03A   WED 22-JUN-77 07:17:34                PAGE 003

0068  700    INTVAL(I) = 1M
0069  500   CONTINUE
0070         INTVAL(JPOINT) = 99
0071        IF (IDEBUG .EQ. 2 .OR. IDEBUG .EQ. 3)
           #   WRITE(6,1000) IPOINT, (EVAL(K), INTVAL(K), K=1,MAXCOL)
0073  1000  FORMAT(' EVAL(*), INTVAL(*) FOR ROW #',I3/
           #   250(' ',F7.3,5X,I2/))
      C     WRITE(7,124)
      C124  FORMAT(' INVRAD EXIT, ENTERING DISPLY')
0074        CALL DISPLY(IPOINT,MAXCOL,IROWB,ICOLB,INTVAL)
0075  100   CONTINUE
0076        RETURN
0077        END
```

```
.MAIN   RT-11 MACRO VM02-12   14-MAY-77 00:07:22 PAGE 1


                        .MCALL  ..V2..,.PRINT,.REGDEF
                        .GLOBL  DISPLY
                ;  *************** DEFINE DISPLAYINSTRUCTIONS ***************
                ;
        164000  ICS     = 164000                ; CONTROL & STATUS FOR VIDEO DISPLAY
        164002  IWC     = 164002                ; WORD COUNT IN INSTRUCTION BUFFER
        164004  ICA     = 164004                ; CORE ADDRESS OF DISPLAY INSTRUCTIONS
        024000  LX1     = 24000                 ; LOAD FIRST X VALUE
        004000  LY1     = 4000                  ; LOAD FIRST Y VALUE
        044000  LCA     = 44000                 ; LOAD CHANNEL ADDRESS
        016000  BXF     = 16000                 ; BLOCK TRANSFER OF DISPLAY DATA
        000100  BLACK   = 100                   ; BLACK BACKGROUND
        002000  LCR     = 2000                  ; LOAD CONTROL REGISTER
        076000  CLS     = 76000                 ; CLEAR SYSTEM
        000020  GRAY    = 20                    ; GRAY SCALE INDICATOR
        001400  DSPLYD  = 1400                  ; DISPLAY DATA MODE
        000010  ADDITV  = 10                    ; ADDITIVE PICTURE ELEMENT
        000040  GRAY16  = 40                    ; 16 LEVELS OF GRAY
        000020  GRAY4   = 20                    ; 4 LEVELS OF GRAY
        000000  R0=%0
        000001  R1=%1
        000002  R2=%2
        000003  R3=%3
        000004  R4=%4
        000005  R5=%5
        000006  SP=%6
        000007  PC=%7
00000           DISPLY:
00000 005725            TST     (R5)+                   ;SKIP OVER # OF ARGS.
00002 013567            MOV     @(R5)+,CURROW           ;CURRENT ROW #
      000474
00006 013567            MOV     @(R5)+,NUMCOL           ;NUMBER OF PIXELS TO BE
      000472
                                                        ;DISPLAYED
00912 013567            MOV     @(R5)+,ROW              ;ROW # OFFSET FROM TOP OF
      000470
                                                        ;SCREEN
00016 013567            MOV     @(R5)+,COL              ;COLUMN # OFFSET FROM LEFT
      000466
                                                        ;LEFT OF SCREEN
00022 011501            MOV     (R5),R1
00024 012700            MOV     #LCAINS,R0
      000516'
00030 012737            MOV     #-1.,@#IWC              ;INITIALIZE DISPLAY WORD COUNT
      177777
      164002
                                                        ;NOW R0 -> LY1 INSRTUCT.
00036 012703            MOV     #8.,R3                  ;8 SHIFT VALUES
      000010
00042 012704            MOV     #SHIFTO,R4              ;PLACE ADDR. OF BIT SHIFT
      000364'
                                                        ;PATTERN (CHANNEL 0) IN R4
00046 012705            MOV     #STACK-2.,R5
      000472'
00052 012445            MOV     (R4)+,-(R5)             ;PUT BIT SHIFT ON STACK
00054 077302            SOB     R3,.-2                  ;BRANCH UNTIL ALL BIT SHIFTS
                                                        ;ARE ON STACK
```

```
.MAIN.  RT-11 MACRO VM02-12   14-MAY-77 00:07:22 PAGE 1+

48 00056 010145          MOV     R1,-(R5)              ;PLACE ADDR. OF INTEGER
49                                                     ;ARRAY ON STACK
50 00060 012703          MOV     #8.,R3                ;8 SHIFT VALUES
         000010
51 00064 012704          MOV     #SHIFT1,R4            ;PLACE ADDR. OF BIT SHIFT
         000404
52                                                     ;PATTERN (CHANNEL 1) IN R4
53 00070 012445          MOV     (R4)+,-(R5)           ;PUT BIT SHIFT VALUE ON STACK
54 00072 077302          SOB     R3,.-2                ;BRANCH UNTIL ALL BIT SHIFTS
55                                                     ;ON STACK
56 00074 010145          MOV     R1,-(R5)              ;PLACE ADDR.OF INTEGER ARRAY
57                                                     ;ON STACK
58 00076 012767          MOV     #2.,CHANL             ;SET DISPLAY CHANNEL TO 1
         000002
         000374
59 00104 012767          MOV     #3.,CLRBIT            ;CLEAR BIT PATTERN (0011)
         000003
         000364
60 00112           NXCHNL:
61                 ; BUILD LCA INSTRUCTION
62 00112 012710          MOV     #LCA,(R0)
         044000
63 00116 062710          ADD     #GRAY4,(R0)
         000020
64 00122 066720          ADD     CHANL,(R0)+
         000352
65 00126 062737          ADD     #-1.,@#IWC            ;INCREMENT IWC
         177777
         164002
66                 ; BUILD LX1 INSTRUCTION
67 00134 012710          MOV     #LX1,(R0)
         024000
68 00140 066720          ADD     COL,(R0)+
         000344
69 00144 062737          ADD     #-1.,@#IWC            ;INCREMENT IWC
         177777
         164002
70                 ; BUILD LY1 INSTRUCTION
71 00152 012710          MOV     #LY1,(R0)             ;
         004000
72 00156 066710          ADD     ROW,(R0)              ;ADD ROW DISPLACEMENT TO LY1
         000324
73                                                     ;INSTRUCTION
74 00162 066720          ADD     CURROW,(R0)+          ;ADD ARRAY ROW #, R0 -> BXF
         000314
75                                                     ;INSTRUCTION
76 00166 062737          ADD     #-1.,@#IWC            ;INCREMENT IWC
         177777
         164002
77 00174 010067          MOV     R0,BLKTRN             ;BLKTRN = ADDR. OF CURRENT
         000312
78                                                     ;BXF DISPLAY INSTRUCT.
79 00200 012720          MOV     #BXF,(R0)+            ;INITIALIZE BXF INSTRUCT.
         016000
80 00204 062737          ADD     #-1.,@#IWC            ;INCREMENT IWC
         177777
         164002
```

```
.MAIN.  RT-11 MACRO VM02-12   14-MAY-77 00:07:22 PAGE 1+

81 00212 012501          MOV     (R5)+,R1                ;GET ADDR OF INTEGER ARRAY
82                                                       ;SP -> BIT SHIFT LIST
83 00214 010567          MOV     R5,STACK                ;SAVE CURRENT STACK POINTER
         000254
84                ;
85                ; BEGIN CONSTRUCTION OF A NEW DISPLAY DATA WORD
86                ;
87 00220 005004 NXTWRD:  CLR     R4                      ;R4 = TEMP. LOC. OF DISPLAY
88                                                       ;DATA WORD
89 00222 012703          MOV     #8.,R3                  ;R3 = OF PIXELS/DISPLAY WORD
         000010
90 00226 016705          MOV     STACK,R5                ;RESET SP TO FIRST BIT SHIFT VALUE
         000242
91                ;
92                ;
93 00232 012102 GET:     MOV     (R1)+,R2                ;GET AN INTEGER ARRAY ELEMENT
94 00234 022702          CMP     #99.,R2                 ;HAS LAST PIXEL BEEN PROCESSED
         000143
95 00240 001405          BEQ     OUT                     ;BRANCH IF IT HAS
96 00242 046702          BIC     CLRBIT,R2                       ;CLEAR BITS
         000230
97 00246 072225          ASH     (R5)+,R2                ;SHIFT R2 AS REQUIRED
98 00250 050204          BIS     R2,R4                   ;COMBINE CURRENT & PREVIOUS
99                                                       ;DISPLAY WORD
100 0252 077311          SOB     R3,GET                  ;BRANCH UNTIL 8 PIXELS INPUT
101               ;
102               ; ADD COMPLETE DISPLAY DATA WORD TO DISPLAY DATA BUFFER
103               ;
104 0254 010420 OUT:     MOV     R4,(R0)+                ;PLACE DISPLAY WORD
105                                                      ;DISPLAY BUFFER
106 0256 062737          ADD     #-1.,@#IWC              ;INCREMENT DISPLAY WORD COUNT
         177777
         164002
107 0264 062777          ADD     #2.,@BLKTRN             ;INCREMENT BXF BYTE COUNT
         000002
         000220
108 0272 022761          CMP     #99.,-2(R1)             ;HAS LAST PIXEL BEEN
         000143
         177776
109                                                      ;PROCESSED ?
110 0300 001347          BNE     NXTWRD                  ;BRANCH TO CONSTRUCT NEXT
111                                                      ;WORD IF NOT
112 0302 022767          CMP     #1.,CHANL               ;IS CURRENT CHANNEL # = 1
         000001
         000170
113 0310 001414          BEQ     INITD                   ;IF YES, BRANCH TO INITIATE
114                                                      ;DISPLAY
115 0312 012767          MOV     #12.,CLRBIT                     ;NOT FINISHED - RESET BIT
         000014
         000156
116                                                      ;CLEAR PATTERN TO (0011)
117 0320 012767          MOV     #1,CHANL                ;CHANGE CHANNEL TO 1
         000001
         000152
118 0326 062767          ADD     #16.,STACK              ;INCREMENT CURRENT STACK ADDR.
         000020
         000140
```

```
 MAIN.   RT-11 MACRO VM02-12   14-MAY-77 00:07:22 PAGE 1+


119 0334 016705           MOV     STACK,R5                ;SET STACK POINTER TO NEXT
         000134
120                                                       ;BIT SHIFT LIST
121 0340 000664           BR      NXCHNL                  ;GO PROCESS INTEGER ARRAY
122                                                       ;FOR NEW CHANNEL
123 0342 012737 IN1TD:    MOV     #CBUFF,@#1CA            ;
         000514'
         164004
124 0350 105767           TSTB    ICS                     ;
         164000'
125 0354 100375           BPL     .-4                     ;
126 0356 005267           INC     ICS                     ;
         164000'
127 0362 000207           RTS     PC                      ;
128 0364 000000 SHIFTO:   .WORD   0.,2.,4.,6.,8.,10.,12.,14.
    0366 000002
    0370 000004
    0372 000006
    0374 000010
    0376 000012
    0400 000014
    0402 000016
129 0404 177776 SHIFT1:   .WORD   -2.,0.,2.,4.,6.,8.,10.,12.
    0406 000000
    0410 000002
    0412 000004
    0414 000006
    0416 000010
    0420 000012
    0422 000014
130 0424        BLOCK:    .BLKW   20.
131 0474 000000 STACK:    .WORD   0
132 0476 000000 CLRBIT:   .WORD   0
133 0500 000000 CHANL:    .WORD   0
134 0502 000000 CURROW:   .WORD   0
135 0504 000000 NUMCOL:   .WORD   0
136 0506 000000 ROW:      .WORD   0
137 0510 000000 COL:      .WORD   0
138 0512 000000 BLKTRN:   .WORD   0
139                       .EVEN
140 0514 003520 CBUFF:    .WORD   LCR+DSPLYD+GRAY+BLACK   ;LOAD CONTROL REG.
141 0516        LCAINS:   .BLKW   150.                    ;DISPLAY DATA INSTRUCTIONS
142      000000'          .END    DISPLY
```

```
MAIN.   RT-11 MACRO VM02-12   14-MAR-77 21:49:59 PAGE 1


                        .MCALL
                        .GLOBL  CLEAR
                ; ********** DEFINE VIDEO DISPLAY INSTRUCTIONS **********
        000100 BLACK    = 100           ; BLACK BACKGROUND
        004000 LY1      = 4000          ; LOAD FIRST Y VALUE
        010000 LY2      = 10000         ; LOAD SECOND Y VALUE
        024000 LX1      = 24000         ; LOAD FIRST X VALUE
        030000 LX2      = 30000         ; LOAD SECOND X VALUE
        076000 CLS      = 76000         ; CLEAR SYSTEM
        044000 LCA      = 44000         ; LOAD CHANNEL ADDRESS
        002000 LCR      = 2000          ; LOAD CONTROL REGISTER
        000040 GRAY16   = 40            ; 16 LEVELS OF GRAY
        000200 ERASE    = 200           ; ERASE THIS CHANNEL
        001200 GRLINE   = 1200          ; GRAPHIC LINE
        164004 ICA      = 164004        ; CORE ADDRESS OF DISPLAY INSTRUCTIONS
        164002 IWC      = 164002        ; WORD COUNT IN INSTRUCTION BUFFER
        164000 ICS      = 164000        ;CONTROL & STATUS REG.  FOR VIDEO DISPLAY
        001400 DSPLYD   = 1400
        000020 GRAY     = 20
        000010 OUTLIN   = 10
                ;
                ; ***** DEFINE REGISTERS *****
                ;
        000007 PC       = %7
                ;
                ; FOLLOWING CLEARS THE VIDEO SCREEN PRIOR TO EACH DISPLAY
                ;
00000 105767 CLEAR:   TSTB    ICS
        164000'
00004 100375          BPL     .-4
00006 012767          MOV     #CBLOCK,ICA
        000030'
        164004'
00014 012767          MOV     #-5.,IWC
        177773
        164002'
00022 005267          INC     ICS
        164000'
00026 000207          RTS     PC
00030 076000 CBLOCK:  .WORD   CLS
00032 003300          .WORD   LCR+GRLINE+BLACK
00034 044240          .WORD   LCA+GRAY16+ERASE+0
00036 044241          .WORD   LCA+GRAY16+ERASE+1
00040 044242          .WORD   LCA+GRAY16+ERASE+2
        000000'       .END    CLEAR
```

```
FORTRAN IV      V01C-03A    THU 16-JUN-77 00:58:03                PAGE 001

0001          DIMENSION  IBUFF(1000), ILINE(40), ICHAR(40)
      C
      C ---> PURPOSE OF THIS PROGRAM IS TO (FOR SPECIFICIED FILE #)
      C       1 - READ BLOCK #'S 121 & 122 AND OUTPUT TO A DISK FILE
      C       2 - REWIND INPUT TAPE AND OUTPUT SPECIFIED # OF BLOCKS OF EEG DATA
      C         TO SPECIFIED DISK FILENAME
      C
      C
      C ---> INPUT THE FOLLOWING VARIABLE VALUES FROM KEYBOARD (PROMTPED)
      C          NAMFIL - NAME OF OUTPUT DISK FILE (MAX OF 12 CHARACTERS)
      C         KFILE - TAPE FILE # (1-4) FROM WHICH DATA IS TO BE READ
      C          KBLK - # OF DATA BLOCKS ON TAPE TO BE WRITTEN TO DISK
0002          WRITE(7,23)
0003  23      FORMAT(' INPUT # OF BLOCKS TO BE WRITTEN TO DISK')
0004          READ(7,3) KBLK
0005  3       FORMAT(I3)
0006          WRITE(7,24)
0007  24      FORMAT(' INPUT FILE # TO BE WRITTEN TO DISK')
0008          READ(7,3) KFILE
0009          WRITE(6,4)
0010  4       FORMAT(' BEGIN PROCESSING EEG TAPE PGM.  NAME -- TSTAPE.FOR')
0011          WRITE(6,45)  KBLK, KFILE
0012  45      FORMAT(' COPY ',I3,' BLOCKS FROM FILE #',I2,' TO ',
     *   'DISK')
0013          CALL ASSIGN(2,'MT')
0014          CALL  ASSIGN(1,'TEST1.DAT')
      C
      C
0015          IFILE = 1
0016          ITRIP = 0
      C FOLLOWING CALL TO MTREAD ALLOWS PASSING OVER TAPE HEADER ONLY
0017  5       CALL MTREAD(IBUFF, IEOF)
0018  90      IBLOCK = 1
      C
      C
0019          WRITE(6,40)  IFILE
0020  40      FORMAT('0BEGIN FILE #',I2)
      C
      C
0021  70      CALL  MTREAD(IBUFF, IEOF)
0022          IF (IEOF .NE. 0)  GO TO 1000
      C
      C
0024          IF (IFILE .EQ.KFILE)
     *     WRITE(6,50)  IBLOCK
0026  50      FORMAT('  BEGIN BLOCK #',I3)
      C       WRITE(6,52) IEOF
      C52     FORMAT('    END OF FILE FLAG (IEOF) = ',I2)
      C
      C
0027          DO 10  K = 1, 25
      C  WANT TO OUTPUT BLOCK #'S 121 & 122 DURING FIRST READ OF TAPE
      C ---> BRANCH TO STMT. LABEL UNTIL BLOCK #121 IS REACHED DURING FIRST
      C       READ OF TAPE(ITRIP = 0)
```

```
FORTRAN IV      V01C-03A    THU 16-JUN-77 00:58:03                PAGE 002

      C     INCREMENT ITRIP BY 1 IMMEDIATELY AFTER REACHING BLOCK #121 FOR FIRST TIME
      C
0028          IF (IFILE .NE. KFILE)  GO TO 65
0030          IF (ITRIP .EQ. 0  .AND. IBLOCK .LT. 121)
     *        GO TO 65
0032          IF (ITRIP .EQ. 0)  ITRIP = ITRIP + 1
0034          J = (K - 1) * 40
0035          DO 20  N = 1, 40
0036           ILINE(N) = IBUFF(J + N)
0037    20    CONTINUE
0038          DECODE(80,30, ILINE, ERR=3000)  ICHAR
0039    30    FORMAT(40A2)
0040    140   IF (K .EQ. 1)  WRITE(6,130)  K,(ICHAR(INDEX),INDEX=1,40)
0042    130   FORMAT(' K = ',I4,' :',40A2)
0043          WRITE(1,30, ERR=2000) (ICHAR(INDEX),INDEX=1,40)
      C     WRITE(6,30)  (ICHAR(INDEX),INDEX=1,40)
0044    10    CONTINUE
      C
      C
0045          WRITE(6,60)  IBLOCK
0046    60    FORMAT('  FINISH BLOCK #',I3/)
0047    65    IBLOCK = IBLOCK + 1
0048          IF (ITRIP .EQ. 0 .AND. IBLOCK .GT. 123)  GO TO 1000
      C
      C --> ITRIP = 1 AFTER BLOCK #121 IS READ FOR FIRST TIME
      C
0050          IF (ITRIP .EQ. 1 .AND. IBLOCK .GT. 122 .AND. IFILE .EQ. KFILE)
     *   GO TO 200
      C
      C --> ITRIP = 2 AT BEGINNING OF SECOND READ OF TAPE
      C
0052          IF (ITRIP .EQ. 2 .AND. IBLOCK .GT. 123) GO TO 1000
0054          IF (ITRIP .EQ. 2 .AND. IBLOCK .GT. KBLK .AND. IFILE .EQ. KFILE)
     *   GO TO 4000
0056          GO TO 70
0057    1000  WRITE(6,80)  IFILE, IEOF
0058    80    FORMAT(' FINISH FILE #',I2,',  IEOF =',I2)
0059          IF (IFILE .GE. 4)  GO TO 4000
0061          IFILE = IFILE + 1
0062          GO TO 5
      C
      C
0063    2000  WRITE(6,100)  (ICHAR(INDEX),INDEX=1,40)
0064    100   FORMAT(' ???? WRITE ERROR, OUTPUT AS FOLLOWS'/40A2)
0065          GO TO 10
0066    3000  WRITE(6,110) (ICHAR(INDEX),INDEX=1,40)
0067    110   FORMAT(' ???? DECODE ERROR, INPUT AS FOLLOWS'/40A2)
0068          GO TO 140
      C
      C --> REWIND TAPE IN PREP. FOR SECOND TAPE READ
      C
0069    200   WRITE(6,210)  IBLOCK
0070    210   FORMAT(' HAVE FINISHED READING BLOCK #',I4/' START '
     *'PROCESSING EEG DATA')
```

# APPENDIX C

## CONVERSION OF GRAY SCALE DISPLAY TO COLOR DISPLAY

Conversion of the current gray scale display to a color display involves several changes in the subroutine DISPLA. Instead of two display channels, three must be used—one for each of the primary colors: red, blue and green. Each eight bit data byte will carry six mask bits, two for each channel, in the low order positions, allowing the display of one image point. Note that the two high order bit positions are not used and that the data bytes are arranged in descending order. The use of three display channels provides for up to 64 different colors. Mapping of the 16 gray levels into the 64 colors should be done so as to allow for a maximum of differentiation. This might well be done by assigning each gray level to every fourth color combination. Simplification of the assembler program may be achieved by performing this mapping prior to execution of the display program.

# VITA

Robert Joseph Bryant was born in Royston, Georgia, in 1951. His parents moved to Greeneville, Tennessee, the following year. He attended elementary schools in that city and was graduated from Greeneville High School in June 1969. The following September he entered Tennessee Technological University. He then transferred to the University of Tennessee the next September, where he received a Bachelor of Science degree in mathematics in December 1973.

He entered graduate school at the University of Tennessee in January 1974. He was employed by Tennessee Valley Authority the preceding September. He received his Master of Science degree in computer science in March 1978.