NAME OF AUTHOR/*NOM DE L'AUTEUR* ZELLA WOLOFSKY

TITLE OF THESIS/*TITRE DE LA THÈSE* Computer Interpretation of Selected Labanotation Commands

UNIVERSITY/*UNIVERSITÉ* Simon Fraser University

DEGREE FOR WHICH THESIS WAS PRESENTED/ *GRADE POUR LEQUEL CETTE THÈSE FUT PRÉSENTÉE* M. Sc (Kines.)

YEAR THIS DEGREE CONFERRED/*ANNÉE D'OBTENTION DE CE DEGRÉ* 1974

NAME OF SUPERVISOR/*NOM DU DIRECTEUR DE THÈSE* John M. Montgomery

DATED/*DATÉ* June 14, 1974 SIGNED/*SIGNÉ*

PERMANENT ADDRESS/*RÉSIDENCE FIXÉ* [redacted]

COMPUTER INTERPRETATION OF SELECTED LABANOTATION COMMANDS

by

Zella Wolofsky

B.Sc., McGill University, 1967

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
(KINESIOLOGY)

in the Department
of
Kinesiology

SIMON FRASER UNIVERSITY

April 1974

APPROVAL

NAME: ZELLA WOLOFSKY

DEGREE: Master of Science (Kinesiology)

TITLE OF THESIS: Computer Interpretation of Selected Labanotation Commands

EXAMINING COMMITTEE:

Chairman: N.M.G. Bhakthan

John M. Montgomery

Thomas W. Calvert

Iris Garland

Thomas C. Brown
External Examiner
Department of Mathematics

Date Approved: 3 June 1974

PARTIAL COPYRIGHT LICENSE

Title of Thesis/Dissertation:

Computer Interpretation of Selected Labanotation Commands

Author: ______________________

(signature)

ZELLA WOLOFSKY

(name)

June 13, 1974

(date)

ABSTRACT

A computer program which converts selected movement notation symbols into numerical representations of the body's positions and prints out the results in the form of graphic output was developed. The notation system chosen for conversion was Labanotation.

Gestural movement commands were analyzed and a program was written which interpreted these forms of commands. Both methods of describing gestural movements available in Labanotation were programmed. In addition, modifications of final positions through specific contractions and intermediate positions and knowledge about the amount of rotation within each limb were programmed. Once spherical coordinates for each bodypart were calculated for every unit of time, the program interpolated the movements for each interval ('frame') to be used in the dynamic displays of the computer outputs. These were plotted as stickfigure drawings for each 'frame.' These plots were photographed one at a time, using the standard animation technique of doubling to produce an animated film of the computer derived movements.

In order to examine the coherence and comprehensibility of the animated films produced from the computer outputs, assessors were asked to compare the computer derived drawings to similar drawings derived from cinematographic analysis of a dancer doing the same movements as were entered into the computer. The assessors were asked to rate the quality of the animation in general, the timing of the movements, the clarity of the drawings and the usefulness such films would have as cursory methods of visualizing movements.

The results of the assessments indicated that the interpolation formulae which calculated the timing of the movements in the conversion program seemed to be oversimplified, as the timing factor for the computer produced film was rated significantly poorer. In general, the assessment results seemed to indicate that both films showed the movements in a sufficiently clear and comprehensible manner for use by dancers.

"It need not surprise us that in the absence of a reliable method of recording dances the process of choreographic art has been checked: ballet, the highest dance form, has not followed the progressive path of other arts in spite of the genius of many illustrious 'maîtres de ballet.' It may be assumed that some of the finest manifestations of choreographic genius have been lost forever, because, being transmitted by memory from generation to generation, in time they pass away completely without a trace for posterity."

V.I. Stepanov (1892).

ACKNOWLEDGEMENT

I wish to express my thanks to John Montgomery who has been my research supervisor and an untiring source of encouragement. I would also like to thank him for helping with the draft of this manuscript and offering valuable criticisms. Tom Calvert and Iris Garland also deserve a special thanks for serving on my research committee and for helping me formulate this research. In addition, I would like to thank Ms. Garland for allowing me to film her dancing and to use her Kinesiology class as film assessors. John Dunn provided me with information about animation that I could never find in the literature, and Jaan Pill and Rudy Wrench, of the Simon Fraser University Film Workshop donated their time and expertise in helping me prepare the animated films. My thanks also goes to Mrs. Ellis who typed the manuscript.

Finally, the positive feedback that I received from David Lach made me realize that anything was possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Dance is an art form which concerns itself with space and time. For its development as a major form, it requires some method of recording and preserving the works of people engaged in making dances.

It required several centuries for music notation to become a uniform system. Because of its recent development dance notation has not yet reached the stage of having one acceptable form. However, dance is more complex than music for it deals with both space and time, and the human body has many potential simultaneous actions. Consequently, there are many more problems involved in the formulation of a notation system in dance than in music.

There are several criteria for a dance notation system to be useable. Firstly, a notation system for movement must be capable of recording movement from a variety of styles, such as ballet, modern dance, jazz, folk dance and tap dancing, as well as the movements of ordinary life, such as walking and running. Secondly, it must be economical. Since there are often several parts of the body to be notated, the system must be moderately quick to write. Thirdly, it must be accurate. Dancers and choreographers who have never seen a notated work should be able to read a score and get an adequate idea of the movements.

Three major systems of movement notation exist today [Turnbaugh, 1970]. One is based on a stickfigure notation, the other two, on analytical descriptions of body movements. Benesh notation, invented by Joan and Rudolph Benesh in 1958 [Causley, 1967], is used extensively by ballet companies but has not been successfully used in other dance forms. The notation is based on an abstract stick-figure drawing. The second notation system was developed by Noa Eshkol and Abraham Wachmann [Eshkol and Wachmann, 1958]. The movements are placed on a cumbersome and complicated staff, and are expressed in terms of degrees of planar, rotary, and conical movements, not in terms ordinarily used in dance. The third system is Labanotation, invented by Rudolph Laban in 1926 [Laban, 1956]. This system is capable of recording changes in angles of the limbs, dynamic changes, and spatial qualities. It too deals with movement in terms foreign to dancers, however, it allows the widest range of movement description of the three notation systems described [Hutchinson, 1972].

Because notation systems should be able to record any human movement, not just a very formalized subset called dance, the need for economy is extremely important. Labanotation combines the space-time elements into one symbol -- the length of a symbol in Labanotation is proportional to the time required to complete the movement.

As a means of preserving choreography, a notation system must be able to give to the reader information concerning spatial changes and the dynamics of the notated works. Video tape and film are useful for the preservation of dances but just as phonographic recordings are not enough for the serious study of a piece of music, then too, film is not enough for the serious reconstruction of a work by someone other than the choreographer.

Movement notation systems are difficult to learn and require several years of advanced study to develop proficiency in their use. Consequently, dancers and choreographers have been very reluctant to learn any notation, although most understand the need for a recording system in dance.

The use of computers in aiding the study of notation is feasible. There are two criteria for the selection of the notation system to be used in such a computer system. First, the notation system would have to be well known, in use and fully developed. Secondly, the notation system would have to be a general movement description system, as opposed to those that notate only movements of a specific style of dance.

The first criterion limits the selection to the notation systems listed above. The Benesh system is used extensively among ballet companies [Turnbaugh, 1970]. Despite its extensive use, this system of notation has several drawbacks. It is used primarily for ballet and is not an analytical system [Hutchinson, 1972] and would prove to be very difficult to computerize [Eshkol, _et al._, 1970]. The two remaining systems approach movement in a similar analytical fashion. Each has advantages and disadvantages.

Eshkol/Wachmann notation treats joint movement as spatial coordinates designating them in numbers. They stress the independence of their notation system from the "stylistic approach which is characteristic and special" [Eshkol and Wachmann, 1958, p. vii] and that it expresses pure movement, not just dance. However, to notate dance, the ability to notate a broader range of movement qualities is desirable.

The essence of the movement being notated must be captured if the notation is to be useful to anyone who has not seen the original dance. The reason for performing a certain movement and the manner in which it is performed are often stressed by the choreographer when setting dances; these subtleties must be written in the notation, otherwise the reconstructed dance would be only a series of positions. Labanotation has the potential of describing the process of movement as well as the spatial positions.

Eshkol *et al.* [1970] used computer technology to help interpret movements of one limb notated in Eshkol/Wachmann notation. There was no attempt at dynamic display of the movement nor was there any significant work on total body movement using their notation.

Works of several contemporary choreographers have been notated in Labanotation and this system is taught in many American universities. However, many dancers do not know, or feel qualified enough, to make use of the available notated scores. If a computer system were operational which read Labanotation symbols and produced a dynamic display of the movements, interpretations of scores could be simplified.

The use of Labanotation in a computer system such as the one suggested by the Eshkol study has never been attempted. Because of the analytical treatment of limb movements in Labanotation and because Laban's work formed the basis for Eshkol/Wachmann notation, the fundamental premises of the Eshkol work would be applicable to the development of a computer system using Labanotation as input.

Such a computer system would not develop into an art form itself but would be a tool of dance. It would be a rough tool and could never replace the use of film or video tape as a preserving and recording medium. Its purposes would be to aid in the reconstruction of dances and the visualization of notated work. Most important would be the usefulness of such a system in providing an overall view of a dance.

The basic problem when using notation is in getting 'the feel' of a dance. A computer system which interpreted notation symbols and produced dynamic displays of the notated movements would provide the cursory view needed when initially looking at a dance score.

Purpose of Research

It is proposed to develop a computer system which would interpret a subset of Labanotation symbols, and produce graphic outputs of the realized movements. The drawings would be in the form of a stickfigure, and by photographing the drawings produced by the computer, a conventional animated film of the movements would be produced. The comprehensibility and clarity of the film and the movements would be tested, and in this way, information regarding the validity of the computer program assumptions could be obtained.

CHAPTER II

RELATED LITERATURE

The desirability of establishing dance as a literary form has been noted by many authors [Morris, 1928; Tetley, 1948; Laban, 1956; Jay, 1957; Eshkol and Wachmann, 1958; McGraw, 1964; Hutchinson, 1970; Turnbaugh, 1970]. Choreographers, without the benefit of a movement notation system have had to rely on esoteric notes or good memories in order to shape a dance. The use of notation systems and the potential applications of computer technology in composition has been noted by Hutchinson [1967]

> . . . if the manipulations of . . . a machine could produce figures on a screen which could be recalled as needed and which accurately reflected a desired movement for the choreographer to see . . . the experimental value to the choreographer would be tremendous. (p. 82)

Recently Cunningham [1973], an American choreographer, also expressed the potential use he could see for computer help in the visualization of movement patterns and ideas. However, he was unclear as to how the man-machine interface could be accomplished. Notation as the language of communication between man and machine has the potential of solving the problem raised by Cunningham.

Preston-Dunlop[1969] provided a comprehensive listing of the current notation systems; however, the most publicized ones have proved to be difficult to master. Although notation is now part of many university dance programs [Dance Notation Bureau, 1973], there is still resistance to incorporate dance notation into dance training.

Work by Eshkol et al. [1970] has provided the basis for a method of reducing this problem by utilizing computer technology. In proving that kinematic descriptions of movement could be derived from coded notation, the need for educated guesses as to the meaning of a notated piece could be replaced by objective analysis by a computer.

Three areas of research are required to produce a computer system which could interpret notation. First, the movement notation must be translated into machine readable forms. Secondly, the computer program which would execute the movements by calculating the numerical representations of the movements would have to be formulated. Thirdly, the results of the program would have to be printed out in numerical or graphic form.

The first area was successfully accomplished by Eshkol's group [1970]. Input in the study was 100% keyboard entry. An attempt at entering Labanotation symbols into a computer was investigated by Fulkerton [no date, ca. 1967]. He called for the use of a light pen and cathode ray tube for entering Labanotation scores into a computer. Unfortunately, this project was never implemented. Implementation on a small scale was done through the manipulation of symbols on a cathode ray tube by Shapiro [1972]. She produced a short Labanotation score. From the point of view of simplicity, manipulation of Labanotation symbols on a screen, rather than remembering an alphameric code (needed with 100% card entry) for each symbol is a desirable attribute of a system such as the one proposed.

In the second area of research, the conversion of notation symbols into machine representations of body positions, virtually no work has been done prior to this time. The Eshkol work was limited to one limb and the STKMAN program of whole body movements did not use Eshkol/Wachmann notation. Eshkol/Wachmann notation uses an external coordinate system which is only effective for the moving limb; any limb attached to the moving limb would change its angles and orientation in the external fixed reference system. This created problems for the authors of the STKMAN program. Also, the rotational state of the limbs was not always uniquely defined. The authors suggested a solution to the latter problem. They felt that by specifying the normal by two polar angles or a rotational angle, a solution to the non-uniqueness of some rotational states could be overcome.

Initial work on developing a computer program which would use Labanotation has been done by the present author. Programming allowed for the description of any number of moving bodyparts. Due to notational features, the handling of locomotor descriptions (motion) was not attempted. The same problem was encountered by the Eshkol group, with no solution of the problem achieved.

Information on the joint movements [American Academy of Orthopaedic Surgeons, 1965] and general information on selected areas of analytical geometry [eg. McCrae, 1953] was obtained for solutions of specific problems in the Labanotation conversion program. However, this area is an entirely new area of research.

The third area of research relates to the display of the computer movements. Given the purpose of the present research project, a visual display of the movements is essential. Animated films, when properly produced, relay sufficient information to be satisfactory replications of movement [Kallis, 1969]. However film, whether real or animated, is not a totally satisfactory method for the observation and analysis of movement. Preston-Dunlop [1969] presented the drawbacks when using film to study movements. She stated that spatial changes were visible, but dynamics were not. However, in the system proposed the use of some sort of two-dimensional viewing medium is the only possible choice.

Summary

Notation systems exist today but are extremely difficult to master. A potential aid in improving this situation could be through the use of computer technology. Work using Eshkol/Wachmann notation as input into a computer system was undertaken, but the project was confined to motion of one limb.

Another notation system, Labanotation, is in extensive use in North America. This system has been used to notate much contemporary choreography, and was the basis for many of the premises of the Eshkol/Wachmann notation. As of yet, however, no work has been done to produce a computer compatible set of Labanotation. Because of the growing number of notated works using this notation system, computer assistance in the reconstruction of scores would be extremely useful. Literature on the conversion of Labanotation symbols into numerical representations of a body does not exist, but the work done in the Eshkol study has applications.

A shortcoming of the Eshkol project was the lack of dynamic displays of the graphic output produced by the computer. Animated film is a good choice for conveying the motions realized by the computer system, and can provide sufficient information to be useful. When cathode ray tube devices are not available, animated film is the only option which can display the movements in real-time.

CHAPTER III

THEORY OF LABANOTATION

Introduction

The problems involved in translating a three-dimensional art form into a two dimensional representation are many sided. In addition to the obvious loss of the third dimension, the interrelationship between time and space must be considered. Dance is primarily concerned with this interrelationship and keeping it intact is of fundamental importance. In film and video representations of dance, this interrelationship is visible although somewhat distorted because of the flatness of the view, but it is reproducible. When translating motion into an abstract notational form, the problem of the time-space relationship must be recognized as being the singularly most important aspect to convey to the viewer.

Nikolais [1948, p. 64] summarized the factors inherent in human movement which must be represented in any notation system. They are: (1) the nature of the body as a moving entity, (2) the space in which the body exists and moves, (3) the time in which the movement takes place, (4) the dynamic energies behind the movement. Others [Eshkol and Wachmann, 1958; Hutchinson, 1970] have also recognized these factors as being the major factors for consideration in movement notation systems.

Labananalysis is an umbrella term which describes the system of notations developed by Laban. There are three areas involved in Labananalysis. They are:

1. Labanotation: This is a description of movement using spatial orientations. Basically, Labanotation identifies the bodypart[1] moving, the timing of the movements, the direction and degree of movement and defines the system of reference from which is based the direction of the movement.
2. Motif Writing: This is the description of movement which identifies the underlying theme or most important feature of the movement being notated. When using Motif Writing the reader is concerned with keeping to a theme, not a particular movement pattern. One might think of Labanotation and Motif Writing as being at opposite ends of a continuum of movement specificity. Motif Writing is used when recording improvisational studies.
3. Effort/Shape: This is the term for the description of the qualitative aspects of movement. It concerns itself with the qualities of tension, body weight, spatial focus and time utilization, as well as the shaping of the body [Hackney, Manno, Topaz, 1970, Part X].

[1] Bodypart is a technical term used in Labanotation to represent a predefined section or segment of a body.

Description of Labanotation Commands Used in Computer Analysis

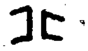
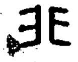
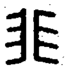
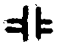
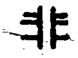
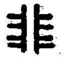
Laban, in the development of his movement notation system, clearly had a firm grasp of the need to represent the factors of space, time and bodypart identification. Each symbol in Labanotation can potentially describe these three factors of movement.

1. Space Factor

Bodypart is a general term used to indicate a limb or a part of a limb. Each bodypart has a bound end, or base, and a free end; these have been defined for reference by Hutchinson [1970, p. 227] and are represented in Table 1.

TABLE I

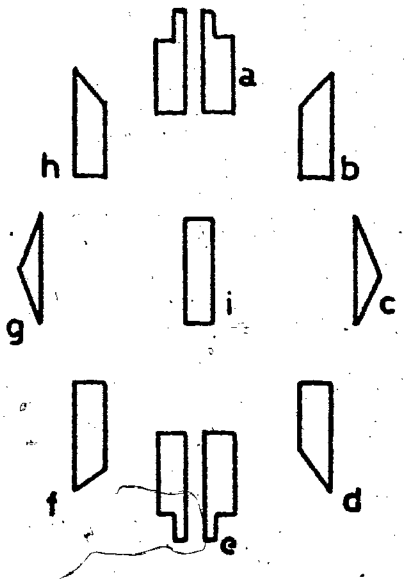
BODYPART COMPONENTS

Bodypart	Bound End	Free End	Labanotation Sign
upper arm	shoulder	elbow	
lower arm	elbow	wrist	
hand	wrist	finger tips	
upper leg	hip	knee	
lower leg	knee	ankle	
foot	ankle	tips of toes	

When calculating spatial directions, Labanotation has adopted a certain frame of reference for measurement.

(a) Direction

The directions are taken from the relationship of the bound to free ends of limbs with specific directional names given to specific relationships, and have been specially shaped to show the directions.



a. Front
b. Right Diagonal Front
c. Right Side
d. Right Diagonal Back
e. Back
f. Left Diagonal Back
g. Left Side
h. Left Diagonal Front
i. Place

Figure 1

Labanotation Symbols

When movement is described in terms of final position, the position is calculated from the relationship between the bound and free ends of the moving bodypart. This type of description is called destinational description in Labanotation. It is used mainly when dealing with gestural movements. A gestural movement (or gesture) is chiefly movement which involves parts of the body not supporting the weight of the body. When movement is described in terms of direction of movement, rather than final position, it is called motional description. Usually this type of movement description is used for locomotor activities and descriptions of the state of the weight-bearing bodyparts. Motional description is a less fundamental description than destinational description and can only be reduced to destinational description with great effort.

(b) Level

The description of direction must be a three-dimensional representation. To do this, shading of the symbol is used and is referred to, in Labanotation, as the 'level' of the movement. Level refers to the direction of movement in three dimensions. When the bound and free ends are in the same horizontal plane, this level is called 'Middle.' 'High' level is when the free end of a limb is 45° above the bound end; 'Low' level is when it is 45° below the bound end of the limb.

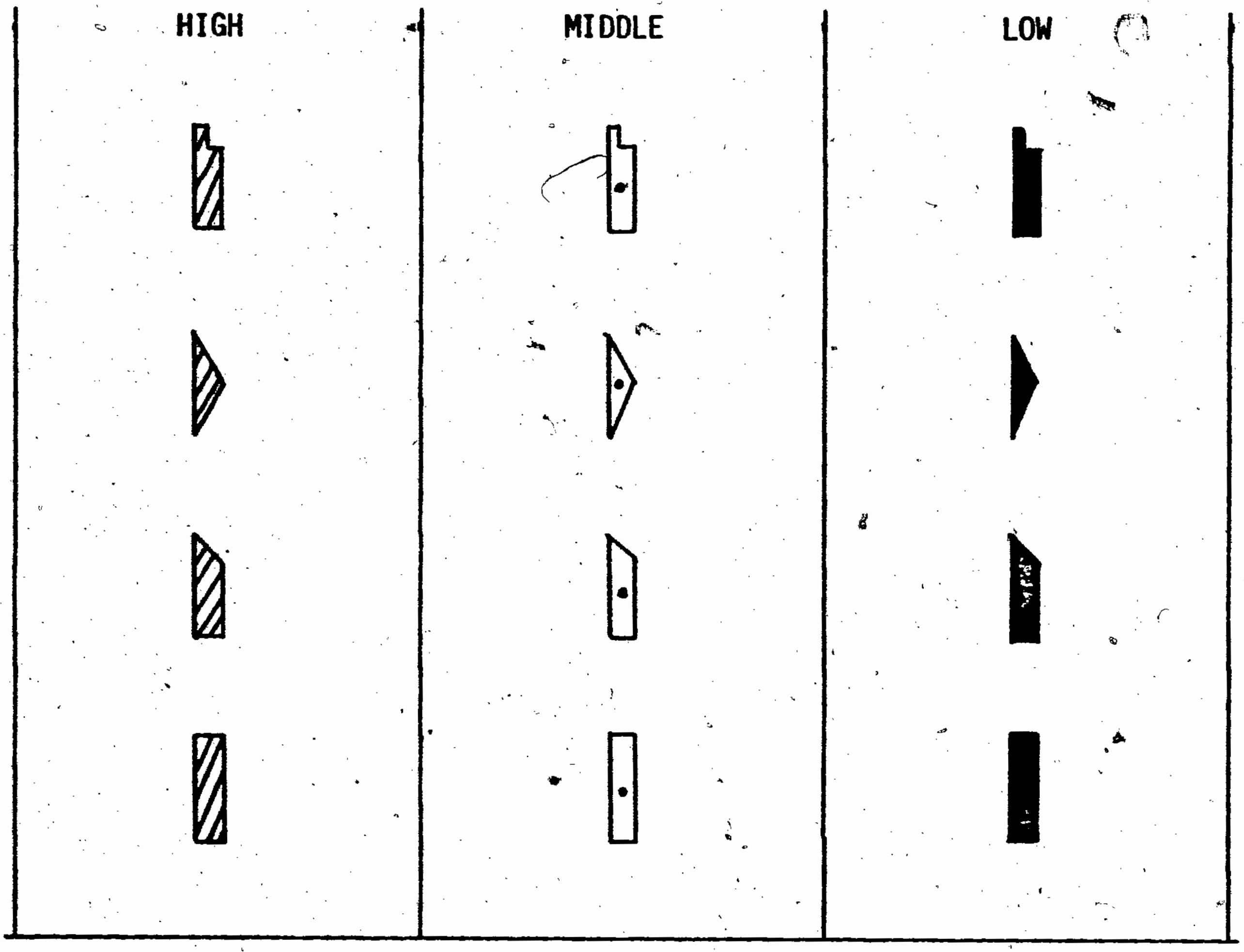


Figure 2

Directional Descriptions With Level Shadings

(c) Rotary Motion

The action of a limb or bodypart moving about its axis is called rotary motion. The axes of the limb and of the movement coincide so that no appreciable change in the position of the limb in space occurs [Eshkol and Wachmann, 1958; Hutchinson, 1970], unlike destinational movements, where the axis of the limb is at right angles to the axis of the movement.

(d) Contractions

Movement which shortens the distance between two points is referred to, in Labanotation, as either flexing or folding. A flexion is movement at two joints, while a fold is movement at one joint. The free end of a bodypart in flexion draws in towards the bound end on a straight path, while in a fold, it approaches the bound end on a curved path. In a flexion, the central joint of a limb is displaced while in a fold it is the extremity which moves. Figure 3 displays the differences between flexes and folds.

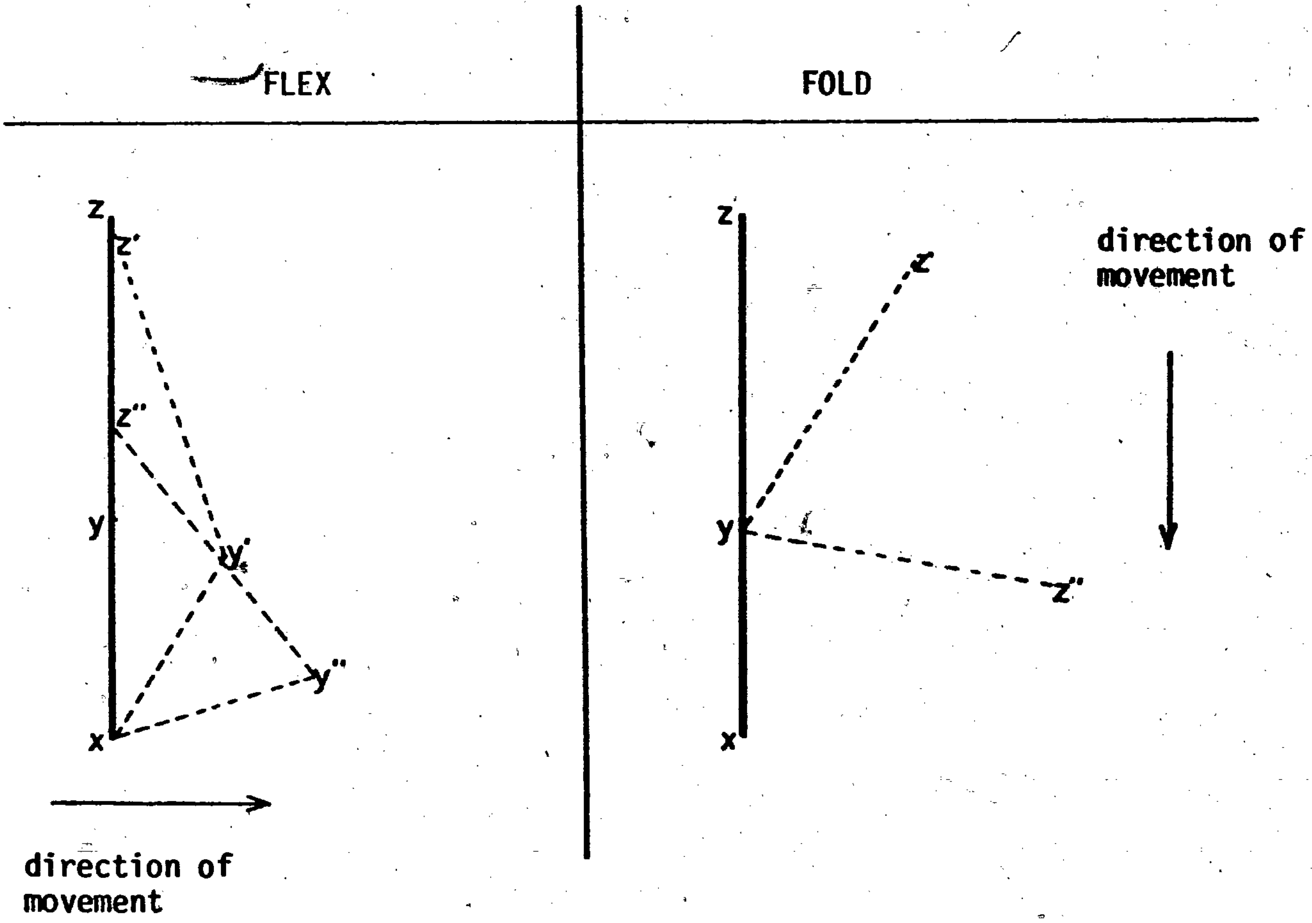


Figure 3

Specific Contractions

2. Bodypart Factor

As with music notation, Labanotation utilizes a staff on which to write the symbols. Unlike music, the staff runs vertically, and is read from the bottom upward. There can be up to fifteen columns in a staff, each column representing a bodypart, or group of bodyparts. Certain columns of the staff have definite bodyparts assigned, while other columns can have more liberal usage. When the latter case occurs, a 'presign' (a symbol used before a direction command on the staff to indicate which bodypart is moving) is used.

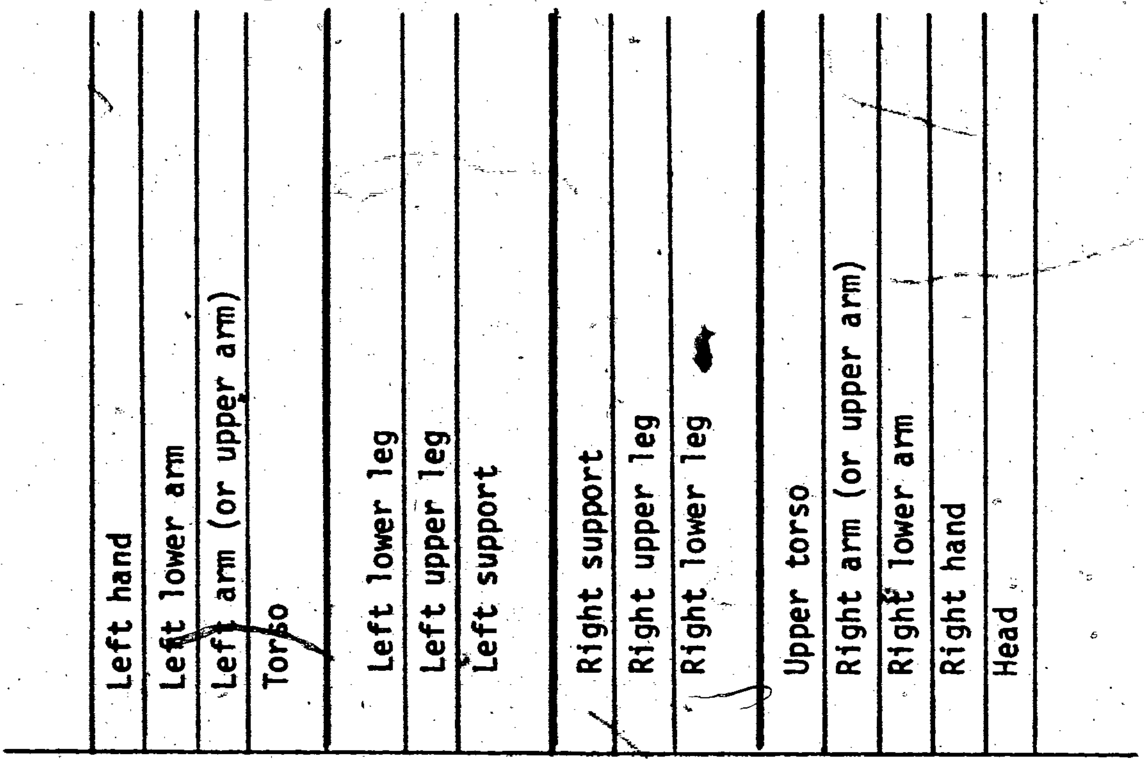


Figure 4
Labanotation Staff

The placement on a staff, or the use of a presign, therefore, indicates which bodypart is being referred to. It must be noted that this convention is used for destinational descriptions. The most often used presigns are shown in Figure 5. When the whole body moves (as in walking or running) the central two columns of the staff are used exclusively to indicate this.

3. Time Factor

Time is marked off on the central line of the staff, the space between the marks represents the duration of the unit of time (either in beats, measures or seconds), usually predetermined. The length of any symbol is proportional to the time required for the movement of the bodypart. Simultaneous actions are described in the same horizontal space on the staff. In Labanotation, the elements of time and space have been combined into one symbol, a factor not found in other notation systems [Hutchinson, 1972]. When both elements of space and time are combined into one symbol, there is an economy of the language, which is important when dealing with the complexities of human movement.

4. Summary

A description of the factors of movement as defined by Nikolais and others have been mentioned. Labanotation includes three of the four factors necessary in any movement notation system.

A destinational description symbol in Labanotation has three built-in factors:- (using Laban's terminology)

(a) identification of the bodypart moving.

(b) identification of the starting time and duration of the movement.

(c) direction and level of the movement.

Additional information to the above is required to describe rotary motion and contractions. In addition, a full explanation of all the Labanotation symbols has not been included in the above discussion, but can be found in Hutchinson [1970]. Only those symbols which have been used in the computer analysis were dealt with.

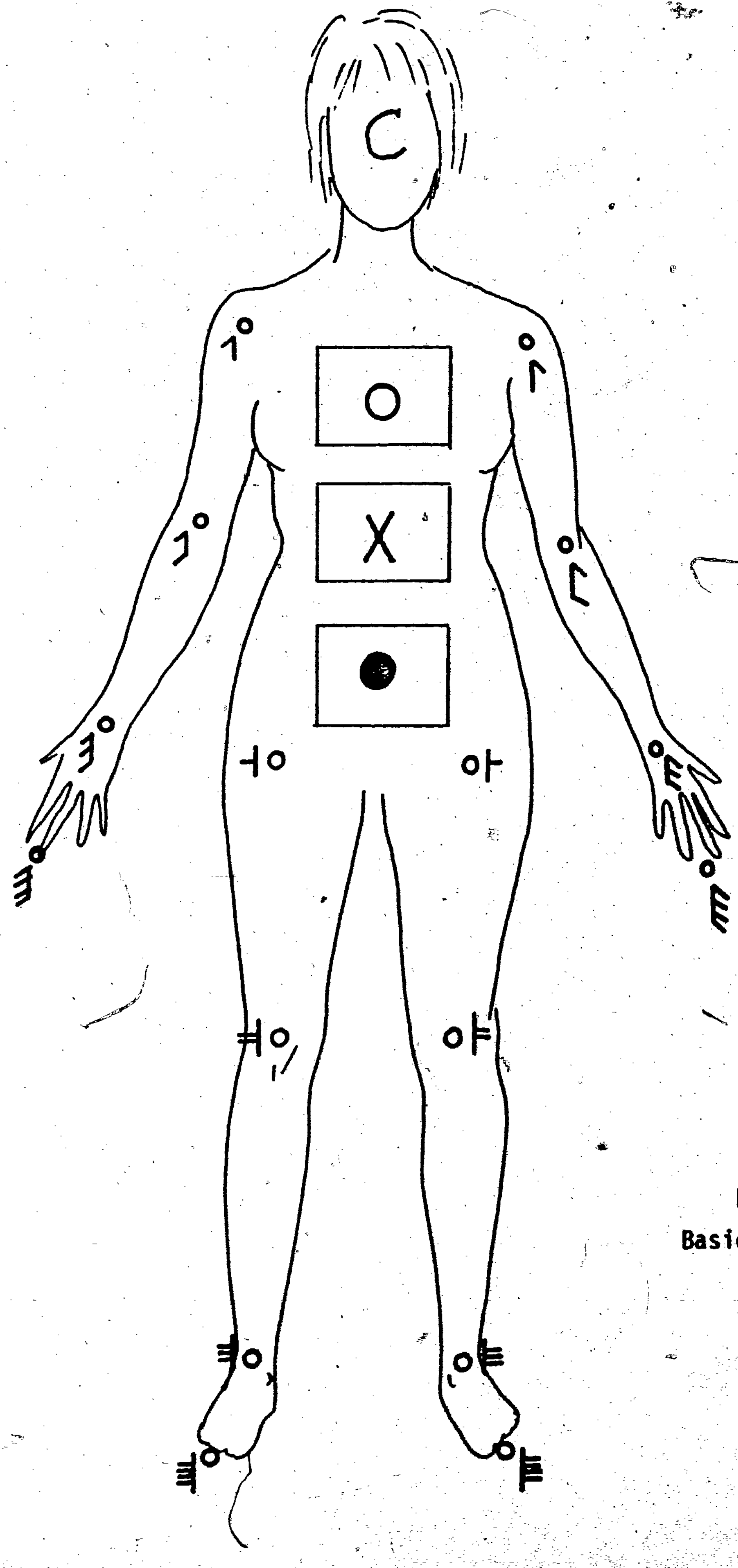


Figure 5

Basic Bodypart Signs

CHAPTER IV

MACHINE REPRESENTATION OF MOVEMENTS

Development of Coding Protocol

A computer program was written to interpret suitably coded information concerning movements and to produce drawings which, when made into an animated film, showed the realized movements in real-time. The optimum data format would allow the user to enter Labanotation symbols directly into the computer. Since the hardware which would allow this was not available, an alternate coding system was devised.

1. Coding Procedure

In Labanotation there are 'limbs' and 'joints,' these not necessarily coinciding with the conventions of anatomy, but representing some convenient point of reference. In the program, the Labanotation convention of bodypart identification was maintained as closely as possible.

The body was divided up into segments (limbs) which were either simple or complex. The simple limbs were defined as having no internal movement capabilities. For example, the arm is a complex limb and is composed of the upper arm, lower arm and hand, which are all considered simple limbs. In this research, the hand and foot was each treated as a unit -- there was no finger or toe movement. The trunk, or torso, was divided up according to the Labanotation convention: the chest region consisted of the region of the torso bounded by the thoracic vertebrae, the waist region bordered by the first and third lumbar vertebrae, and the pelvis, by the fourth vertebrae and the coccyx.

At the start of the program, this data was entered, as well as other relevant information about each bodypart. In addition to the limbs, the joints were numbered to accommodate the problem of hierarchy of movement.[1]

The lower end of the torso was set as '1' and the other joints of the torso (here defined as the free end of each torso section) were numbered consecutively. The right side of the body was numbered evenly, the left had odd numbers. Figure 6 shows the numbering of the bodyparts (both joints and limbs).

Two examples representing the same movements are shown in Figure 7. The first example will describe the movements using Labanotation symbols, the second uses the coding protocol developed for the program.

The movements -- The initial position is a standing position, facing forward, both feet together. On the first beat, the left leg

[1] Whenever a bodypart moves, it takes all the other bodyparts attached to its moving end, along with it. There is, then, a hierarchy of bodyparts, the closer the bodypart to the torso, the higher on the hierarchy. In the calculation of the position of each limb, at each time interval, a specific order had to be followed so that those limbs closer to the torso were calculated first.

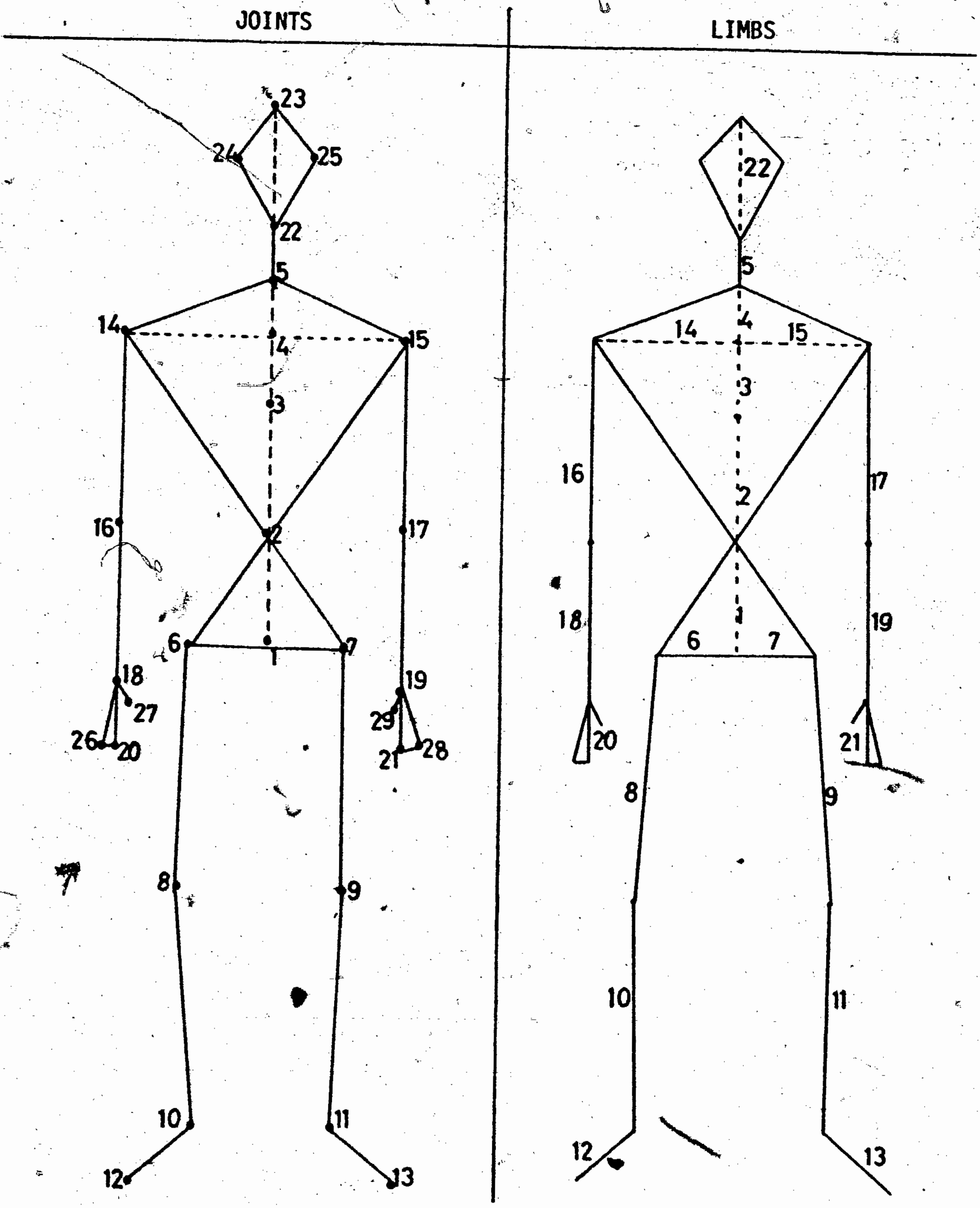


Figure 6

Numbering of Bodyparts in LABAN

goes to forward hip level, taking one entire beat to complete the movement. On the second beat, the left leg goes to the left side, also at hip level, taking one half beat. On the third beat, the left leg lowers by 45° and the movement requires an entire beat to complete.

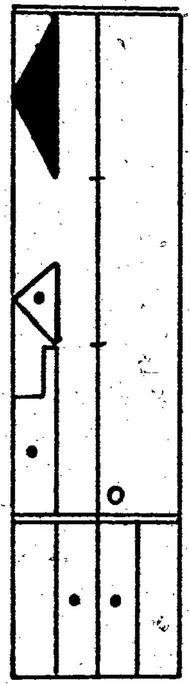
Labanotation representation -- The Labanotation representations of the above movements are shown in Figure 7.

Coded input of LABAN[2] program -- Bodypart, timing and spatial identifications are inherent in any Labanotation symbol by virtue of the fact that the symbol is placed on a staff (which defines the starting point and the bodypart moving), has a certain size (the duration of the movement), shape (direction of the movement) and shading (level of the movement). This information had to be explicitly given in the program. The following format was chosen.

The first column represents the type of movement, '2' for a support (a bodypart which bears weight), '1' for a gesture, '0' for the end of the input. The second column of the input data represents the starting time of a particular movement. The third column represents the bodypart (either a joint or a limb) moving or acting as a support. The fourth column represents the duration of the movement or support. The fifth column represents the direction and level of the movement. Often there can be more than one entry in the last column. This is how 'intermediate positions' (those directions not defined by a specific symbol) are entered. Appendix E shows the actual dance entered into LABAN.

[2] LABAN is the name given to the computer program developed for this research. The listing is given in Appendix A, and a flowchart provided in Appendix B.

LABANOTATION REPRESENTATION



INPUT INTO LABAN PROGRAM

Type of Movement	Starting Time	Bodypart	Duration	Direction/ Level
2	0.00	ANR	4.00	-
2	0.00	ANL	1.00	-
1	1.00	WLL	1.00	F
1	2.00	WLL	.50	LSM
1	3.00	WLL	1.00	LSL
0	4.00	-	-	-

Figure 7

Labanotation Description

2. Additional Data

Additional data needed for the final animation drawings had to be entered. The cine rate (desired frames per second) and timing (beats per minute) were required as data to calculate the number of frames per beat. In addition, information used to calculate the apparent angle of viewing of the final plotter drawings and the two-dimensional projections, was given at the outset of the program.

It was decided that each limb would be described in terms of spherical coordinates. Thus, translation of Labanotation direction and level symbols into angular measure had to be done. Front direction (in Labanotation) was considered to be zero radians, backwards, to be π radians, the right side (perpendicular to front) $\pi/2$ radians, the left side $3\pi/2$ radians. Middle level was chosen to be zero radians, high level, to be $3\pi/4$ radians and low level to be $\pi/4$ radians.

Each position in Labanotation could be described by using two angles, ϕ for direction measure and ψ for level measure. A problem arose when qualifying 'Place.' Because Place is the longitudinal axis of the body in gestural movements, certain changes had to be made to the values of ϕ (direction) when moving to or from this position. In order to keep the movements in one plane, when the origin (or destination) of a movement was Place, the ϕ value of the destination (or origin) had to be adopted instead of the value of 2π radians normally given to Place.[3] Thus Place has a ψ measure of $\pm\pi/2$

[3] This measure of 2π radians is an arbitrary measure; it could have been any other.

radians, but a non-unique ϕ.

Although the drawings produced by the computer were stick-figure drawings, the facings of palms, feet, and the rotation of each limb about its longitudinal axis had to be known and this was also part of the initial data. The angle of rotation was called θ.

LABAN Interpreter Program

1. General Overview

The program LABAN interprets: (1) coded bodyparts and commands to those bodyparts and computes the spherical coordinates describing the final position of each bodypart, (2) computes the duration left to complete each command at the end of each beat, (3) computes by interpolation, the rectangular coordinates of certain points, at specific intervals of time, (4) plots a stickfigure drawing of the moves, and prints the rectangular coordinates of the points at each time interval desired. The program can move as many bodyparts as must be moved at any given time. Any gestural movement, described in terms of destinational movements can be interpreted by LABAN.

The original plan was to have the program run on a Digital PDP 8e, a minicomputer, with initial debugging to be done on a larger machine, the IBM 370/50. The program, in order to fit into the minicomputer, was divided up into three sections, each section calling up the next when completed. However, the size of the program and the lack of adequate plotting facilities on the PDP 8e made the use of this computer quite impractical, and the final work was done exclusively on the IBM 370.

2. Translation of Labanotation into Machine Readable Forms

The first section of the program primarily read in and stored data concerning coded bodyparts, tempo of the dance, cine rate and plotting information. The smallest fraction of a beat used in any particular dance was referred to as the 'beat-step.' This value of the beat-step was constant, and all duration information was converted into beat step units. This allowed for the use of integer, rather than real values, thus providing economy of memory space requirements.

(a) Outline of LABAN1 (Section one of LABAN)

This routine read in the coded dance, decided if a bodypart was a limb or joint, and if the destinational command was modified by some sort of contraction or intermediate position. These modifications were stored in arrays to be used later in the program.

(i) Directional angle conversion - The direction and level commands were then translated into directional angles of ϕ and ψ respectively.[4] The ϕ and ψ values for each bodypart and the

[4] Labanotation commands have nine specific directions, intermediate positions are specially modified positions, and are given in the coded commands as a series of components of the desired position. The average of the components' directions is taken to give the correct ϕ and ψ. Differences of 7.5° in direction and level are possible in LABAN.

duration needed to complete the movement of each bodypart were stored in arrays at each beat-step.

If no new command was given for a particular bodypart at a specific time, the previous values of ϕ and ψ were retained. If no command was ever given for a particular bodypart, the initial default position value (the value read in before information about a dance is read) for the bodypart was stored in the arrays. If no direction and level commands were given, but a flex or fold command was, the previous ϕ and ψ values were assumed and stored.

(ii) Support bodyparts -- Arrays specifying which bodypart was supporting the body weight were also stored as well as arrays indicating the duration of these supports.

(iii) Complex to simple bodypart conversion -- Certain limbs were treated as 'complex,' composed of a series of bodyparts (e.g. arm). Movement of these complex limbs had to be described as movements of the component parts. The values of ϕ and ψ and the duration of the movement for each of the component parts of a complex limb were stored in arrays as if there had been movement commands for these component parts individually.

(iv) Calculations of angle of rotation of a bodypart -- Because of the nature of the human body, contractions normally occur in one direction, toward the inner surface of the limb. The 'facing' of the inner surface (i.e., the rotation of the limb) then had to be calculated before any flex or fold calculations could be done.

Theory used to calculate the angle of rotation: If the rotation of each limb could be defined as an angle of rotation about an axis, then it could be treated in the same way as the directional angles. An imaginary set of axes at each joint of the stickfigure is assumed to exist, with the positive X-axis to the right of the figure (viewers' left), the positive Y-axis coming out of the page, and the Z-axis having its positive direction upwards.

If the end of an element is described by the three dimensional vector $\underline{u}$, then a general rotation can be produced by an application of a transformation matrix A, so that

$$\underline{u}' = A\underline{u}$$

Rotation about the Z-axis in a clockwise direction, looking down the positive axis towards the origin, yields a positive ϕ and a matrix D which describes the rotation.

$$D = \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Rotation about the X-axis in a clockwise direction looking towards the origin along the positive axis, yields a positive ψ and a matrix C which describes the rotation.

$$C = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{pmatrix}$$

Rotation about the Y-axis in a counterclockwise direction, looking down the positive axis towards the origin, yields a positive θ and a matrix B which describes the rotation

$$B = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}$$

$$A = DCB \quad \cdots\cdots\cdots\cdots\cdots \quad (1)$$

A is a transformation matrix representing a general rotation through angles ϕ, ψ, and θ from an initial position with $\phi = \psi = \theta = 0$.[5] Because of the appropriate choice of angles in describing direction and level Labanotation commands, the ϕ and ψ are known. The θ is unknown in most cases except for that of a bent complex limb.

The degree of rotation of a complex limb can be determined from the coordinates of the normal to the plane of the complex limb. The

[5] This represents an impossible position, but was chosen to allow the normals of the head, torso, and arms to be off to the right and the normal of the legs, to the left.

coordinates of this normal can be found in two ways:

(i) by taking the cross-product of the vectors which represent the vectors of the positions of each of the component limbs.

(ii) by applying the transformation matrix A representing rotations of the normal through the angles ϕ, ψ, and θ.

By equating these two sets of coordinates, we can solve for θ, the angle of rotation.

Matrix A is

$$\begin{pmatrix} \cos\phi\cos\theta - \sin\phi\sin\psi\sin\theta & \sin\phi\cos\psi & \cos\phi\sin\theta + \sin\phi\sin\psi\cos\theta \\ -\sin\phi\cos\theta - \cos\phi\sin\psi\sin\theta & \cos\phi\cos\psi & -\sin\phi\sin\theta - \cos\phi\sin\psi\cos\theta \\ -\cos\psi\sin\theta & -\sin\psi & \cos\psi\cos\theta \end{pmatrix}$$

Multiplying matrix A by the vector X (the normal to the limb in its initial position of $\phi = \psi = \theta = 0$), where $X = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ results in the vector P, where L, M, and N represent the elements of the vector P.

$$P = \begin{pmatrix} L \\ M \\ N \end{pmatrix} = \begin{pmatrix} \cos\phi\cos\theta - \sin\phi\sin\psi\sin\theta \\ -\sin\phi\cos\theta - \cos\phi\sin\psi\sin\theta \\ -\cos\psi\sin\theta \end{pmatrix}$$

If there are two component limb parts, with directional angles ϕ and ψ and ϕ' and ψ' respectively, then the cross-product of the vectors which represent the component parts is

$$\begin{pmatrix} l \\ m \\ n \end{pmatrix} = \begin{pmatrix} \cos\phi'\cos\psi'\sin\psi - \sin\psi'\cos\phi\cos\psi \\ -\sin\phi'\cos\psi'\sin\psi + \sin\psi'\sin\phi\cos\psi \\ -\cos\psi'\cos\psi(\sin\phi'\cos\phi - \cos\phi'\sin\phi) \end{pmatrix}$$

These two vectors (l, m, n) and (L, M, N) are identical except for magnitude and therefore

$$l = kL \quad \ldots . (1)$$

$$m = kM \quad \ldots . (2)$$

$$n = kN \quad \ldots . (3),$$

where k is an unknown positive constant.

This unknown constant can be found by substituting k = n / N into the equations (2) or (1). The value of k is unique.

For computational sanity let

$$Mn = Nm \quad \ldots . (4)$$

$$Lm = Ml \quad \ldots . (5)$$

$$Ln = Nl \quad \ldots . (6)$$

Using (4) yields

$$\tan\theta = \frac{-n\cdot\cos\phi}{-m\cdot\cos\psi + n\cdot\cos\phi\sin\psi} \quad \ldots . (7)$$

Let θ be a solution to this equation, where $-\pi/2 < \theta_0 < \pi/2$. Then $\theta = \theta_0$ or $\theta = \theta_0 + \pi$. One of these two solutions yields a vector (L, M, N) which is identical (except for magnitude) with the vector (l, m, n); the other is parallel but in the opposite direction. To identify the correct value of θ, the vector (L, M, N) is calculated with $\theta = \theta_0$. Using the relations (1), (2), or (3) the sign of k is checked. If k is positive, then $\theta = \theta_0$; if k is calculated to be negative, then $\theta = \theta_0 + \pi$.

Method of calculating the rotation of limbs in LABAN: Rotations of the torso (bodyparts 1 - 10) were not allowed in this version of LABAN. Only the rotations for the arms and legs were calculated.

Each complex limb was considered to be composed of three simple bodyparts: upper, lower and extremity. If a limb was not bent the subroutines ARMS or LEGS were used to calculate the angle of rotation. These subroutines assigned rotations to certain directional angles and could calculate, using a basic formula, the angle of rotation for the other directional angles. If a limb was bent so that any part of the limb lay in a different plane, subroutine CHCK was used to calculate the angle of rotation. CHCK was based on the theory outlined in the section above (i.e. equating two sets of normal coordinates).

Admissible ranges of rotation between the lower limb and the extremity were set up and checked, changing any previous value for θ if necessary to bring its value within an admissible range. Once the angle of rotation of each bodypart for a particular beat-step was calculated and stored in arrays, the contraction modifications could be calculated, for the 'facings' of the limbs were then known, and the plane of movement of the contraction was known.

(v) Limb modification -- The basic problem involved when flexes or folds modified a position was that the ϕ and ψ of the bodyparts involved in the flex or fold would change.[6] Owing to the Labanotation convention that flexes and folds modify two adjacent bodyparts, detailed calculation had to be done if the bodyparts were the upper and lower arms (or legs). Any movement of the upper or lower arm (or leg) would move the hand (or foot). If the extremity had the same ϕ and ψ as the lower limb (same direction and level), the modification of the lower limb would affect the extremity as well.

Theory used to calculate limb contractions: A line of intersection (all points satisfying two linear equations simultaneously) of two planes has as its directional vector the cross-product of each of the two normals to the planes.

[6] Contractions occur in the plane described by the normal of the limb. The normal is known because the angle of rotation is known (θ).

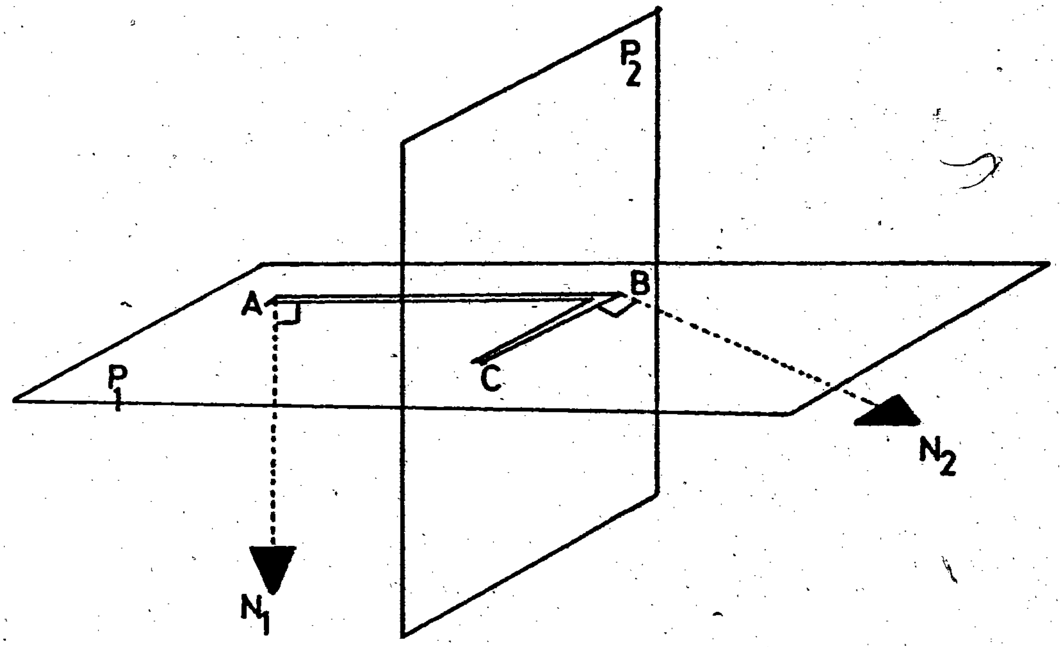


Figure 8

Intersection of Planes

In Figure 8, let AB and BC be two lines in a plane P_1, with the normal N_1. BC is also in the plane P_2 with normal N_2. Then the line of intersection of P_1 and P_2 is BC.

If AB and BC represent two adjacent simple limbs of a complex limb, ABC, their normals are known (θ for each limb has just been calculated). The cross-product of these normals yields the directional cosine of the limb BC. Since the length of BC is known, the coordinates of B can be calculated.

Knowing that a limb can be described by two angles ϕ, ψ and the length of a limb (L), the rectangular coordinates of the endpoint can be calculated, using the following formulae (see figure 9).

$$x = L \cdot \cos\psi\sin\phi \qquad \ldots. (8)$$

$$y = L \cdot \cos\psi\cos\phi \qquad \ldots. (9)$$

$$z = -L \cdot \sin\psi \qquad \ldots. (10)$$

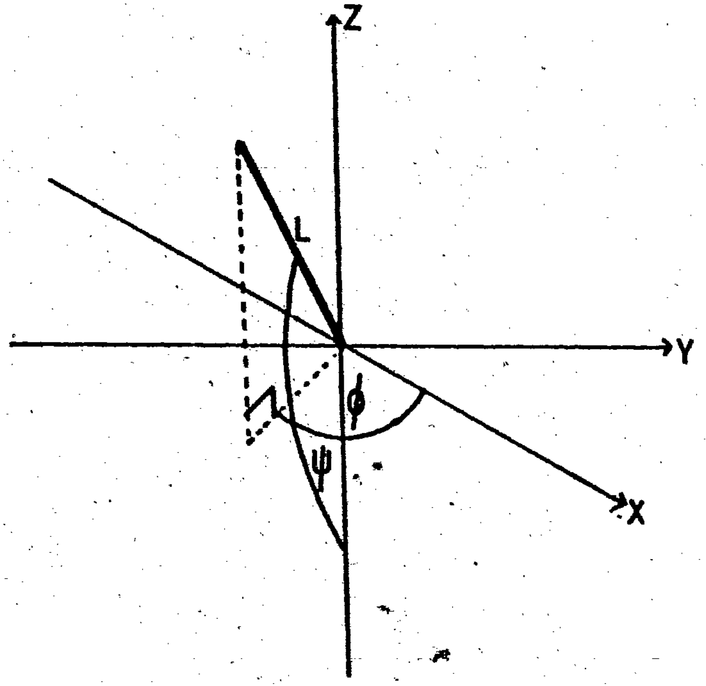


Figure 9

Defining Coordinates of a Line

Inversely, once the coordinates of a point are known, the angles ϕ and ψ can be calculated, from (10),

$\sin\psi = -L / z$, giving a value for ψ .

From (8) and (9),

$\sin\phi = x / L \cdot \cos\psi$

$\cos\phi = y / L \cdot \cos\psi$, giving a value for ϕ .

This results in the directional angles ϕ and ψ for a limb BC after a modification. The angle of rotation will not have changed, for the contraction has occurred within the plane of the limb.

Method of flexion calculation in LABAN1: The initial position of the whole limb before contraction is represented by PR in Figure 10. The upper limb (called MODB in the program) and the lower limb (MODB1) are PQ and QR respectively. Angle PQR is the angle of flexion derived from the degree of flexion specified in the initial Labanotation command to flex. (The amount of contraction is called 'degrees' of contraction in Labanotation)

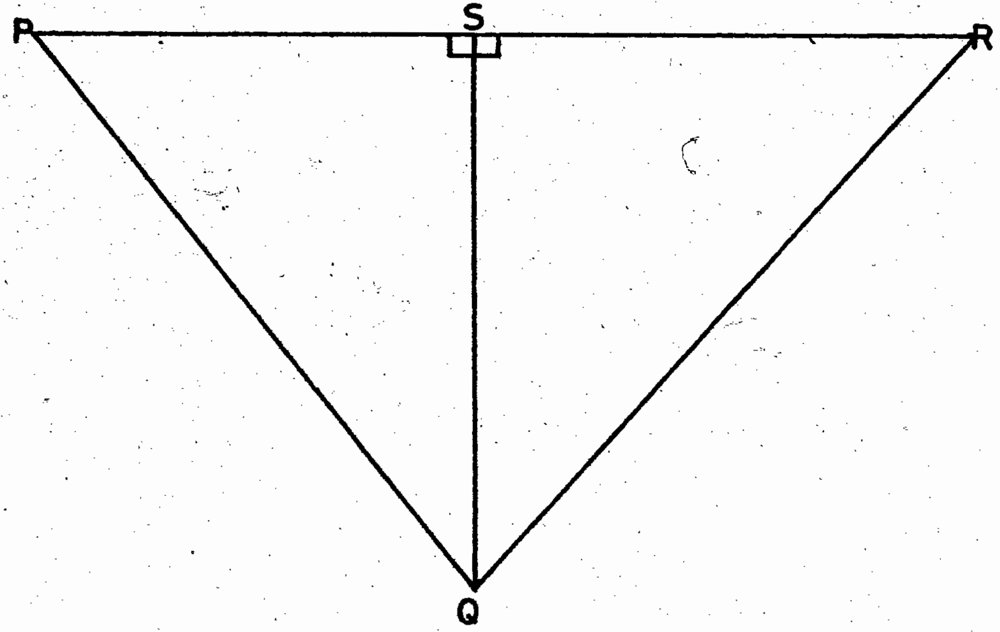


Figure 10

Flex Calculation

Using the sine rule,

$$PR = \frac{QR \cdot \sin P\hat{Q}R}{\sin Q\hat{P}R}$$

and PS and SQ can be calculated:

$$PS = PQ \cdot \cos Q\hat{P}R$$
$$SQ = PQ \cdot \sin Q\hat{P}R$$

The directional cosines of $\vec{PQ}$ and $\vec{QR}$ (= $-\vec{RQ}$) must be then found. In the program subroutine DCS calculated the directional cosines of $\vec{RQ}$ and $\vec{PQ}$, given the lengths PR, QR and the angle PRQ, in addition to the directional cosine of $\vec{PR}$ and the normal to the plane PRQ (the angle of rotation defines the normal). This subroutine took the cross-product of the normal to the plane PRQ and $\vec{PR}$, yielding the directional cosine of $\vec{SQ}$. Knowing the coordinates of S, the coordinates of Q were then calculated and the required directional cosines of $\vec{PR}$ and $\vec{QR}$ derived.

Once the directional cosines were known, the subroutine PHPSTA calculated the ϕ, ψ and θ for the limb, given the directional cosines of the normal to the limb.

<u>Method of fold calculation in LABAN1:</u> In a fold, the upper part of a limb remains stationary and the lower part of the limb moves. In Figure 11, PR represents the stationary part of the limb (MODB in the program), RQ represents the moving limb (MODB1). Angle PRQ is the angle of folding derived from the degree of folding specified in the initial Labanotation movement command.

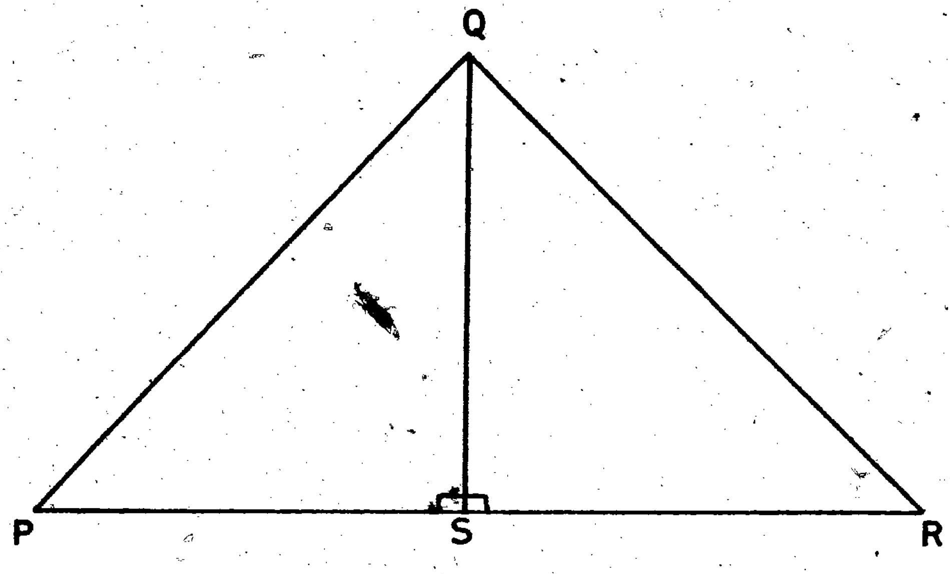


Figure 11

Fold Calculation

The fold modification is simpler to calculate than the flex modification, because the only directional cosine required is that of $\vec{RQ}$ (only one component part of a complex limb moves in a folding contraction).

$$PS = PR - QR \cdot \cos \widehat{PRQ}$$

$$SQ = QR \cdot \sin \widehat{PRQ}$$

The subroutine DCS in LABAN1 calculated the directional cosine of $\vec{RQ}$, by finding the directional cosine of $\vec{SQ}$, the coordinates of S and deriving the coordinates of Q. Once the coordinates of Q and directional cosine of $\vec{RQ}$ were known, the subroutine PHPSTA was called. This routine calculated the ϕ, ψ and θ given the normals to the planes of the limbs.

(vi) <u>Transfer of data onto file to be used in next section of LABAN</u> -- The previously stored values for the modified bodyparts were then changed to the newly calculated ones. Once all the modifications had been completed, the arrays storing the ϕ, ψ and θ values for each bodypart at a particular beat-step were written onto a file to be used by the next section of LABAN.

(vii) <u>Summary</u> - LABAN1 (1) converted final direction and level commands into directional angles ϕ, ψ; (2) stored information on supportive bodyparts; (3) converted complex limbs to simple component parts; (4) calculated the angle of rotation for each simple bodypart, before any contractions (flex or fold) had occurred; (5) changed the ϕ, ψ of each bodypart involved in a modification; (6) transferred the finalized ϕ, ψ and θ and durations of the movement of each simple bodypart at a particular beat-step onto file.

The six steps outlined above were repeated over again for every beat-step of the dance. When the signal for the end of the dance was reached, the program called the next section of LABAN (LABAN2) and proceeded with the new section.

3. Calculation of the Numerical Representations of Movements

(a) Introduction

Up to this point in the conversion program, LABAN, calculations have been based on final positions and the duration required to complete the movements. However, in order to have the necessary data to eventually produce a dynamic display of the realized movements, it is necessary to know the actual position of the limbs at specific points in time. This section of the program is concerned with calculating this type of information.

(b) Outline of LABAN2 (Section two of LABAN)

This section of the program first read in the final ϕ, ψ and θ and the time left to complete the final position for each bodypart for a particular beat-step. By interpolation, it then computed the ϕ, ψ and θ values of each bodypart at the end of each 'frame.'[7] The position at the end of each frame was called the 'current time' position, the directional angles have 'current time' values, as well. In this section of the program, the current time values for each bodypart's directional angles were calculated. Rectangular coordinates of each joint (extremity point of each limb, in this case)

[7] A frame indicates a drawing of a figure at the end of a small interval of time (e.g. 1/12th of a second). After photographing the frame with other frames in a series (using animation techniques) a conventional animated film would result.

were calculated, and then stored in arrays, later to be written onto files for use by the third section of LABAN.

(i) Calculation of current time ϕ, ψ and θ values - The information stored in the arrays passed on from LABAN1 represented the final position to be reached and the duration required to complete the movement. In order to know the exact position of the limb at any point in time (e.g. in a frame), the values of ϕ, ψ and θ at that moment, had to be known. These values of ϕ, ψ and θ were named 'current time' values for they represented the state of the limb at the desired time interval, not the ultimate position reached.

Theory of the current time values: The theory is based on the following assumptions. A movement occurs in a plane and each segment of the movement must remain in that plane. There exists an angle between the initial and final positions of the movement. At the end of a frame, there is an angle between the initial and current time positions, which is a fraction of the first angle. To calculate the current time position, the 'current time angle' is first calculated using linear interpolation. Since the normal of the plane is known, the current time values of ϕ, ψ and θ can be calculated. The mathematics involved are very similar to that used in fold calculations on LABAN1. For each calculation of current time values of ϕ and ψ, a current time value of θ has to be calculated as well.

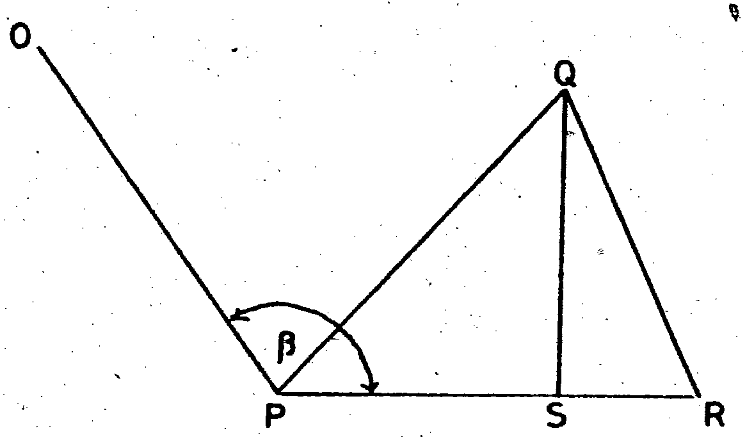


Figure 12

Current Time Calculations for Torso and/or Upper Sections of Limbs

If a bodypart is either part of the torso or the proximal section of the limbs, the following procedure is used. The initial (previous current time) position of the bodypart is represented in Figure 12 by PR, PO represents the final position (PR = PQ = PO). Angle OPR is the total angle of movement between initial and final positions (= β). Angle RPQ represents the amount moved in one frame (current time angle). The normal to the plane PRQ is obtained by taking the cross-product of $\vec{PR}$ and $\vec{PO}$. The lengths of PS and SQ can be obtained because angle RPQ is known. The directional cosines of $\vec{PQ}$ can be calculated and from there, the values of ϕ, ψ calculated for the position.

For lower sections of limbs, the following procedures are used to calculate the ϕ, ψ values.

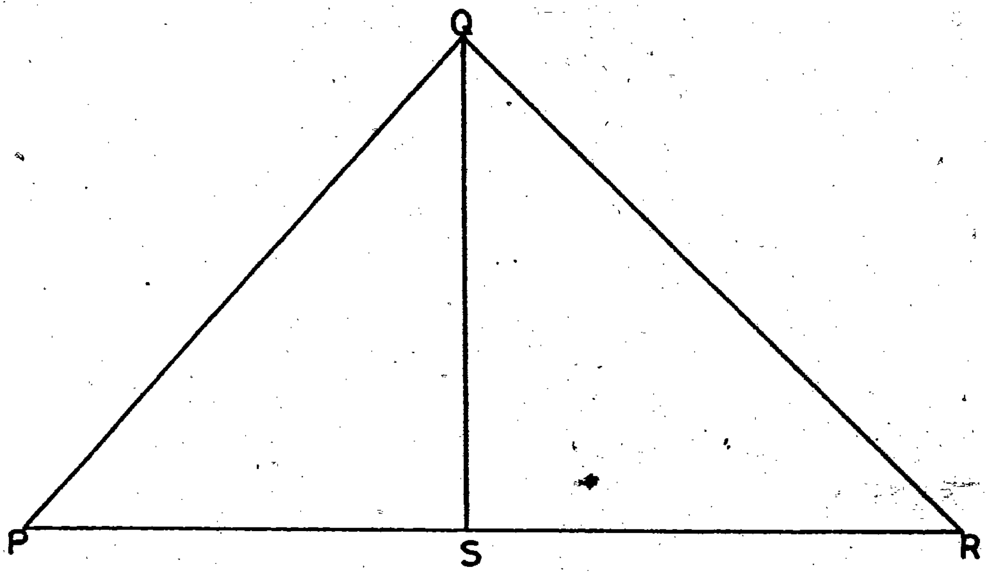


Figure 13

Current Time Calculations for Lower Sections of Limbs

PR is the current time position of the upper section of the limb (LMUP in the program). Angle PRQ is the interpolated value of the current time angle between the upper and lower sections. The normal to the plane is the normal to the current time position of the upper part of the limb, PR. The directional cosine of $\vec{RQ}$ can be calculated, and the current time ϕ and ψ values of the lower section of the limb worked out subsequently.

Method of calculating current time values in LABAN2: The initial angle and the final angle were calculated and the current time value for the limbs were calculated through linear interpolation. The routine NEWDA calculated these interpolations in a similar manner to the fold calculation used in LABAN1.

If a limb moved through Place High or Place Low, the θ value would change by π radians, and a flip in the limb would be noticeable. To accommodate this and to allow for a smooth transition, equivalent values which did not give rise to a flip were given to the ϕ and θ values of a limb, whenever the limb had a ψ value of $\pm \pi/2$ radians.

(ii) Rectangular coordinate generation -- Once the current time values of ϕ, ψ and θ were known for each bodypart for a particular frame, the rectangular coordinates of each joint were calculated. The computer program set the coordinates of the first joint in the hierarchy at (0, 0, 0) and looped through this section of LABAN2, calculating the coordinates relative to the first set of coordinates.

The program also determined if a support duration was stored for a particular joint. If there was a support duration stored, the z-coordinate of this joint was set to zero to represent the floor (whose z-coordinate is treated as being zero). Then all previously calculated coordinates were readjusted to accommodate this change. Since most cases calculated have the support on the feet, certain provisions had to be made to insure that a gesture of a leg would not bring the foot through the floor (i.e. have a z-coordinate which is a negative value).

(iii) Additional plotting points -- Additional points used for plotting purposes (right and left sides of the head, thumbs and palms of hands) were also calculated in LABAN2. The x-, y-, z- coordinates of each of the additional points were taken relative to the bound end of the limb in question (head or hand) and were part of the initial data read in by LABAN1 and passed on to LABAN2. By applying the transformation matrix A (same as in LABAN1)rotations through the angles ϕ, ψ and θ were obtained. Using the ϕ, ψ and θ values of the head or hand, the coordinates of the specific additional points were obtained (using subroutine TRDSP).

(iv) Transfer of data -- Once these coordinates were calculated, all the information needed to plot the particular frame was written onto a file for use by the next section of LABAN (LABANPL).

(v) Summary -- Section two of LABAN, LABAN2, (1) calculated the current time values of ϕ, ψ and θ values; (2) worked out rectangular coordinates of each joint for each frame of a beat-step; (3) calculated the coordinates of the additional points (used for plotting); and (4) stored the data of the previously calculated information as arrays, and wrote the current time values on file. Reiteration of LABAN2 continued for all the frames of any particular beat-step. Once the beat-step was completed, all the arrays concerned with the completed beat-step (ϕ, ψ and θ and the duration) were transferred to another set of arrays (called 'previous data' arrays) and the program was looped through again until the end of the data from LABAN1 was reached.

4. Graphic Output

(a) Introduction

This section read in the rectangular coordinates of each joint for a particular frame, calculated the projection of the x-, y-, z-coordinates of each joint onto a plane Y = 0, using previously supplied information so as to determine the height, distance and angle of the 'performer' (stickfigure) from the 'camera' (apparent viewpoint of viewers of the ultimate product -- an animated film), and plotted out a stickfigure drawing which diagrammed this information.

(b) Outline of LABANPL (Section three of LABAN)

(i) General theory of projections -- Projection of a point (X_p, Y_p, Z_p) onto a plane $Y_S = 0$ from a point (X_c, Y_c, Z_c) is the point (X_S, Y_S, Z_S), where

$$X_S = X_c + \lambda(X_p - X_c) \qquad \ldots\ldots (1)$$

$$Y_S = Y_c + \lambda(Y_p - Y_c) \qquad \ldots\ldots (2)$$

$$Z_S = Z_c + \lambda(Z_p - Z_c) \qquad \ldots\ldots (3)$$

Because $Y_S = 0$,

$$\lambda = -Y_c / (Y_p - Y_c) \qquad \ldots\ldots (4)$$

Then,

$$X_S = X_c - \frac{Y_c(X_p - X_c)}{(Y_p - Y_c)} \qquad \ldots\ldots (5)$$

$$Z_S = Z_c - \frac{Y_c(Z_p - Z_c)}{(Y_p - Y_c)} \qquad \ldots\ldots (6)$$

However (X_p, Y_p, Z_p) might not always be in the same set of axes as (X_s, Y_s, Z_s). In this case, (X_p, Y_p, Z_p) must be transformed.

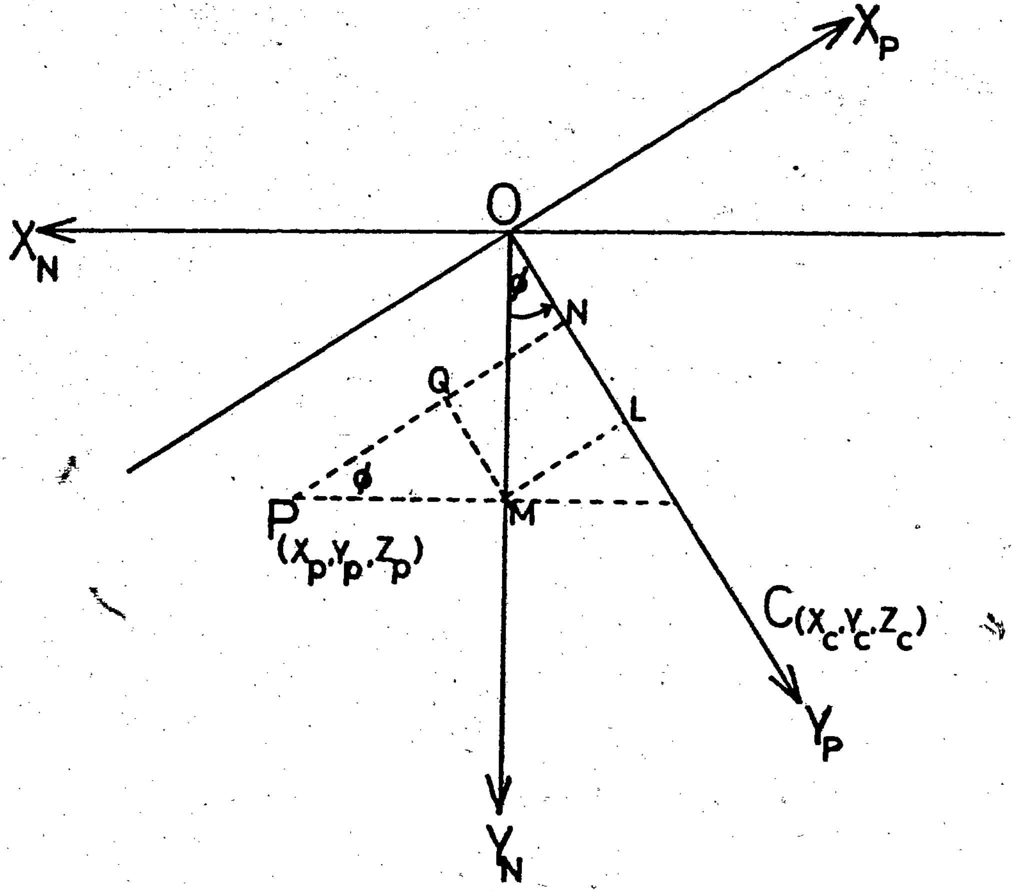


Figure 14

Projection of a Point onto a Plane

Using diagram 14,

$$X_p = -NP \qquad \ldots.(7)$$

$$Y_p = ON \qquad \ldots.(8)$$

$$Z_p = Z_N \qquad \ldots.(9)$$

Using (7),

$$X_p = -QP - QN = -X_N \cdot \cos\phi - Y_N \cdot \sin\phi \qquad . . . . . (10)$$

Using (8)

$$Y_p = OL - NL = Y_N \cdot \cos\phi - X_N \cdot \sin\phi \qquad . . . (11)$$

By substituting these values of X_p, Y_p and Z_p in equations (5) and (6), the projected point (X_S, Y_S, Z_S) can be found.

(ii) <u>Method of calculating projections of coordinates in LABANPL</u> -- The point (X_p, Y_p, Z_p) represented a point on the stickfigure calculated in LABAN2. Point (X_c, Y_c, Z_c) represented the 'camera' and had the coordinates (projected onto the plane $Y_S = 0$ of the plotter) of (0, r_c, h_c), where r_c represented the distance of the camera from the stickfigure and h_c represented the height of the camera above ground level. These were the names assigned to these parameters in the program LABANPL, and were initially entered into the program at the beginning of LABAN1 or at the start of LABANPL.

The subroutine PROJ calculated the projection of the points passed on from LABAN2, onto a plane $Y = 0$. The plotter's X-axis and the stickfigure's X-axis were in different directions, because when viewing the stickfigure from the front, its right became left and left became right.

A 'stage' (five points on the plane $Z = 0$) was also drawn, in order to provide additional visual cues.

(iii) Other features of LABANPL -- If no plotting was desired, LABANPL skipped the plotting routines and only listed the rectangular coordinates of each joint, as passed on from LABAN2, and each joint's projected coordinates as calculated in LABANPL. If only specific frame information was wanted, this was done by supplying special data cards which were read at the start of LABANPL.

Evaluation Procedures

The computer program LABAN was used to interpret movement commands given in specially coded Labanotation commands. A Calcomp plotter, available at the Simon Fraser University Computer Centre, was used to produce the individual frames of a film. Each frame was numbered and drawn in black ink on 12 inch wide white paper, using a pen size of 0.5 mm. In order to photograph the frames, conventional animation techniques were used. Each frame had to be made into a separate 9" x 12" sheet. Instructions to the plotter allowed for this, and a line was drawn separating each frame from the next, at a distance of 10 inches. Appendix D gives samples of the plots.

Animation stands have specially constructed bars at the top of the photographic area, so special paper had to be attached to the top of each frame drawing. The stage remained stationary throughout the dance, so its corners were used as reference points in aligning each frame drawing with the next. The stand in the Simon Fraser Film

Workshop produced the most aesthetic images using a size nine field. The camera used was a 16 mm Bolex reflex with single frame shutter capability using black and white film (TRI-X reversal, with an ASA of 160).

The conventional technique of photographing each frame twice was used, resulting in twelve image changes per second, with a cine rate of twenty-four per second. (More stable perceptual images arise in the case of 'doubling,' and work on the part of the animator is substantially reduced.)

The same movements, coded in Labanotation, were danced by a dancer and these were also photographed on 16 mm film. Specific coordinates (duplicating those points of the computer derived stick-figure drawings) were obtained from each frame of the dancer film, using a Vanguard Motion Analyzer. The coordinates were punched on cards and a plotting program was written to have the Motion Analyzer stickfigure drawings plotted on the Calcomp plotter. This was done to produce two sets of similar looking figures. The same animation techniques were used to produce a 16 mm black and white film of the Motion Analyzer derived films, using the same camera and film type.

Both films were made into film loops to allow viewing of the movements several times in quick succession. The two loops were shown to forty assessors. All but three had had at least one course in beginning dance at Simon Fraser University. The other three were professional dancers. A questionnaire was distributed before the showing of each loop. It asked assessors to rate the animation, timing of the movements, clarity, drawing and movements of the figure, using a five-point scale.

Questions requiring written answers were also included in the questionnaire. It was felt that these types of answers would provide valuable information on the acceptability of the films, even though they were not amenable to rigourous statistical analysis. Each film, being only eight seconds long, was shown fifteen times in rapid succession providing the assessors a chance to fully examine the contents of each film.

Analysis of Data

A sign test was applied to every assessor's responses. The direction of the differences, not the quantitative difference, was considered. The ratings of the Motion Analyzer derived film were compared with the computer derived film . If the rating of the former was higher, a positive sign was given; if the rating was lower, a negative sign was given. If both were rated equal, a zero score was assigned and the total number of pairs reduced when doing the statistical analysis for that question. [Seigel, 1956].

Table II shows the results of the questions for the Motion Analyzer film and the computer derived film, giving totals for each rating for each question.

TABLE II

TOTALS FOR COMPUTER DERIVED FILM ASSESSMENTS AND MOTION ANALYZER FILM ASSESSMENTS FOR EACH RATING[a]

Question[b]	Ratings									
	1		2		3		4		5	
1a			1		8	5	17	10	14	25
1b			1		2	3	18	14	19	23
2			1	2	2	1	15	5	22	32
3			4	2	12	3	14	20	10	15
4			3		9	5	16	20	12	15
5				1	4	4	16	12	20	23
6a			1	2	5	9	20	15	14	14
6b			3	2	5	7	25	21	7	10
6c				1	12	7	17	27	11	5
6d			1	3	12	9	16	23	11	5
6e			2	6	10	6	14	20	14	8
6f	1	1	1	3	9	6	24	20	5	10
6g			3	3	11	4	21	22	5	11
6h			2	2	14	7	20	24	4	7
6i			1	1	7	6	19	18	13	15
6j			1	1	10	8	20	18	9	13
6k				1	9	6	20	21	11	12
6l	1		2	5	6	5	16	20	15	10
6m	1		1	3	11	5	15	22	12	10
6n			1		14	5	14	20	11	15
6o				2	13	6	16	17	11	15
7a			2	2	7	6	12	16	19	16
7b			2	2	13	8	18	13	7	17
7c		1	2		7	1	12	14	19	24
8			2		10	6	12	16	16	18

[a] Computer derived film ratings are on the left hand side of each column; the Motion Analyzer film ratings on the right side of each column.

[b] See Appendix C for the text of the questionnaires.

Table III shows the total of positive and negative signs and the probability at the 5% level of significance. Table IV shows the results of the assessments as to how mechanical looking the movements appeared to each assessor. The results of the sign tests, as well as a sample of the questionnaire are listed in Appendix C.

TABLE III

DATA FROM THE SIGN TEST

(5% LEVEL OF SIGNIFICANCE)

Questions	No.of Positive Signs	No.of Negative Signs	Total of Signs (Non Zero)	Total of Fewer Signs	Probability
1a	17	2	19	2	.0005*
1b	17	5	22	5	.008*
2	11	4	15	4	.059
3	17	3	20	3	.001*
4	14	6	20	6	.058
5	9	5	14	5	.151
6a	7	5	12	5	.387
6b	9	6	17	6	.166
6c	10	7	17	7	.315
6d	12	6	18	6	.119
6e	12	6	18	6	.119
6f	12	8	20	8	.225
6g	14	4	18	4	.015*
6h	10	7	17	7	.315
6i	8	4	12	4	.119
6j	10	7	17	7	.315
6k	11	3	14	3	.029*
6l	6	10	16	6	.227
6m	11	9	20	9	.412
6n	11	2	13	2	.011*
6o	10	6	16	6	.227
7a	13	10	23	10	.339
7b	12	8	20	8	.225
7c	20	5	25	5	.002*
8	15	6	21	6	.039*

*Statistically significant   α = 0.05.

TABLE IV

ASSESSMENT OF QUESTION NINE[a] OF THE QUESTIONNAIRE

[a] The assessors were asked if they found the movements mechanical looking

Assessor	Answer to Question Nine	
	Motion Analyzer	Computer
1	N	N
2	N	Y
3	N	Y
4	N	Y
5	N	N
6	N	N
7	N	Y
8	N	Y
9	Y	N
10	Y	N
11	N	Y
12	N	N
13	Y	Y
14	N	N
15	Y	Y
16	Y	Y
17	N	N
18	N	N
19	N	N
20	N	N
21	Y	Y
22	N	N
23	N	N
24	Y	Y
25	N	N
26	N	N
27	N	N
28	N	N
29	Y	Y
30	Y	Y
31	N	N
32	N	N
33	N	N
34	N	Y
35	N	N
36	Y	Y
37	N	N
38	N	N
39	Y	Y
40	N	Y
Sum	Y: 11 N: 29	Y: 17 N: 23

CHAPTER V

CONCLUSIONS

Discussion of Results

There were several assumptions made when initially writing the LABAN interpreter program which were not anatomically or mechanically correct when describing human movement, but would perhaps be acceptable when looking at movement in animated films. In general, the assumptions made were simplifications; simplifying the stickfigure drawings to what might be the most basic acceptable form, and simplifying the velocity calculations for each movement.

1. Stickfigure Drawing

In order to produce as close an approximation to a human form as possible, curved, filled-in forms would be needed. Such drawings would: (a) be very costly to draw from the standpoint of computer time, (b) involve some type of solution of the hidden line problem.

It was decided to choose the simplest possible drawing which could show some three-dimensional characteristics. A stickfigure drawing with a torso, palms and head shapes, and straight lines for arms and legs was chosen. This figure could be drawn relatively simply using a series of vectors (each vector having eight points within it which were specified). The waist of the figure was a single point, obviously artificial. The head was a diamond shape, only a rough approximation of a general rounded shape (See figure 6, page 25).

2. Joint Articulation

To have anatomically and mechanically correct information would require knowledge of mass and volume of each bodypart. It was decided to eliminate all kinetic descriptions of movements and to use only kinematic descriptions. This allowed the limbs to be treated as straight rods having no mass. Therefore the only mathematics required were those which defined the positions of the limbs in space.

Limits to the range of movement of the elbow, knee, wrist and ankle were programmed and set at 165°. This was chosen from a purely visual point of view. These limitations were necessary in order to make the movements of the figure agree somewhat with what we would normally see when observing a person move. The articulations of the shoulders and hips were not limited in any way; it was felt that the Labanotation commands used would never provide visually impossible movements at these two sets of joints. In addition, movements of the arms and legs did not produce accompanying movements of the torso.

3. Timing of Movements

Human movement is not always accomplished with a constant velocity. Typically, there will be phases of acceleration, constant velocity and deceleration during a movement. However in LABAN, this was not observed. Each complete movement was divided up into equal time intervals and the following general formula was used in calculating the interval.

$$\begin{array}{c}\text{Movement Per}\\\text{Frame}\end{array} = \begin{array}{c}\text{Initial}\\\text{Starting}\\\text{Position}\end{array} + \frac{\text{(Final Position - Current Position)}}{\begin{array}{c}\text{Total Amount of Frames Needed to}\\\text{Complete the Movement}\end{array}}$$

Animators do not always follow this general formula [Dunn, 1973]. They sometimes use geometric, rather than arithmetic progressions to split up the total time, giving a much clearer acceleration and deceleration. For purposes of this research, it was felt that the arithmetic progressions used might be sufficient to give information about the realized movements.

The assumptions outlined above could potentially contribute to a loss of information concerning the movements demonstrated in the animated films. To what extent this would occur, was unknown. The questionnaires given to evaluate the films examined the drawing and timing of the movements separately, as well as the clarity of the animation and tried to get an overall impression of the assessors' feelings towards the usefulness of such types of films.

4. The Questionnaire

There were four sections to the questionnaire: animation techniques, timing of the movements, stickfigure drawing and general appreciations.

(a) Animation Techniques

In answering how well an experienced dancer might reproduce the movements shown in the film strip, twenty-five assessors gave the same response for both sets of films, and fifteen assessors gave different answers. The differences are not significant at the 5% level and judging from the total number of people who rated the films equally, this might indicate that the films may be sufficiently coherent to be used in learning a movement phrase.

There was a significant difference in the apparent smoothness of the two films. Most assessors found the Motion Analyzer derived film to be smoother. However, since both films were made using the same technique, the same camera, film and photographic techniques, the smoothness of the films might have been influenced by the contents of the films themselves. In general, however, judging from the results, most people viewing the films seemed to be able to accept animation as a method of portrayal of movement.

(b) Timing of the Movements

There was a significant difference in the ratings of impulses and accents as shown in both films. The Motion Analyzer derived film showed impulses and accents better than did the other film. But when asked how well an experienced dancer might be able to understand the timing, only fourteen assessors differed from their original answers and this difference was not significant. In the fourth section of the questionnaire, however, general written comments did concern themselves with the fact that the computer derived film showed mechanical looking movements, which might have partially influenced how assessors would understand the timing and rhythm of the movements of the film.

Because the LABAN program did have a simplified method of calculating velocity of the movements, the fact that people would find the movements somewhat mechanical was reasonable. Some comments mentioned that the computer derived film seemed to be very fast at the start. The lack of a more definite acceleration and deceleration would make the movements appear faster than they would normally appear if the acceleration and deceleration calculations had been closer to the manner in which human bodies are expected to move [Dunn, 1973].

(c) Stickfigure Drawings

Significant differences were found in the ratings of the torso size, arm shape and arm connection to the torso, as well as how clearly the leg movements could be observed. The torso in the computer derived film had broader shoulders and was 'squarer' looking (to use the words of one assessor).

The arms were straight lines and assessors felt that they could have been formed. The arm connection to the torso was also an area which had been simplified for programming ease. Assessors felt that the arm connections in the Motion Analyzer film did make a significant contribution to the naturalness of the image on the screen. The lack of associated shoulder girdle movement which should have accomplished some of the arm movements (particularly those which brought the arms overhead) was a shortcoming in the computer derived film.

There was a significant difference in clarity of arm movements between both films. The arm movements appeared to be more visible and comprehensible in the Motion Analyzer film.

(d) General Appreciations

This section of the questionnaire tried to gain information about how useful the films might be as a cursory method of viewing movement. Subjective responses to questions were encouraged. When asked if they found the movements mechanical looking, nine assessors answered affirmatively for both types of films. Several mentioned the angularity of the stickfigure as being a reason for the mechanical looking movements. Eight assessors felt that only the computer derived movements appeared mechanical, two found only the Motion Analyzer derived movements mechanical.

Since both films were approximately the same shape and proportions, factors other than the angularity of the stickfigure may have affected the assessments. One assessor noted that the stillness of the non-moving bodyparts in the computer derived film was very artificial. Unfortunately the LABAN program has no method of producing associated shoulder or hip movements when the arms or legs moved. Another assessor stated that the timing was 'mushy.' Impulses, often given at the start of movements in dance as an unconscious accent, were not programmed into LABAN, instead, the velocity was equal throughout a movement, a manner of moving not normally observed in dance. The differences between the ratings of the timing element of the films in the second section of the questionnaire were significant, and the method of velocity calculation in the program might have been a reason for this.

The dancer in the original film added embellishments to the movements that were part of her performing style and the film of her probably showed movements which were more complex than those produced by the computer. Rather than being more mechanical, the latter film could be thought of as being less complex, when compared to the Motion Analyzer derived film.

Thirty-one of the thirty five assessors who commented on the useability of these films as cursory methods of visualizing dance, felt that the films were extremely clear and an excellent method of showing the essentials of the movements.

Conclusions

There are three elements which could contribute to the loss of information in this research. The first is the use of film as a viewing medium which can destroy much of the dynamic qualities of movements. However, in this research, the use of film is completely unavoidable.

The second element lies in the use of stickfigure drawings.

The figure might be too simple to be of use. The results of the questionnaires given showed that stickfigure drawings are comprehensive and coherent. The assessors realized and accepted the abstract human form represented by the stickfigure drawing, for the drawings had enough detail to be easily identified as a human form. Reference was made to the 'mechanical man' feature of straight line drawings, suggesting that the body should be more rounded; however, it was also noted that the figure was still coherent.

Perhaps the element which contributed most to the loss of information was the method of velocity calculation in the LABAN program. The method used in the program may not have mirrored the usual movements of human bodies closely enough, giving rise to artificial looking movements in the computer derived stickfigure drawings. Animator's rule of thumb of geometric progressions for interpolation of time intervals are more acceptable to observers than the arithmetic progression used to split up the time required to go from position to position, as in the program.

In analyzing the 'direction' and 'level' commands the program did provide adequate interpretations of the Labanotation symbols, for the assessors felt that they could recreate the movements seen on both films equally well. In general, no one element proved to be detrimental to the overall comprehension of the movements of the figures in the films. Both types of films were judged as satisfactory by the majority of assessors.

In conclusion, it has been demonstrated that it is possible to write a program to interpret selected Labanotation commands and produce stickfigure drawings, which when viewed as an animated film, adequately represent the movements notated.

The development of a computer system which could interpret movement notation (specifically Labanotation) is feasible. Further work on the use of Labanotation symbols, rather than codes, as the method of entering commands into the system should be done. The way notation is handled in Labanotation is by symbols, and their use would make an interpreter computer system more inviting. Undoubtedly, the bulk of any further work would have to be done in the incorporation of a larger set of Labanotation commands. In order for the system to be useable, additional movement description possibilities are necessary. To provide a general mathematical analysis of motion description, however, would require much work, for in this type of description, the direction of movement, rather than the specific position, is stated. Real-time displays of the interpreted movements should also be part of a fully developed system. Some sort of cathode ray tube device would be required.

Making the reconstruction of notated movement scores simpler might encourage their use by dancers, choreographers and directors. Familiarity with Labanotation might encourage these same people to develop literacy, thus providing dance with a reliable, uniform notation system it still requires.

BIBLIOGRAPHY

BIBLIOGRAPHY

American Academy of Orthopaedic Surgeons (1965). Joint Motion - Method of Measuring and Recording. Chicago, Ill.: American Academy of Orthopaedic Surgeons.

Causley, Marguerite (1967). An Introduction to Benesh Movement Notation. London: Max Parrish.

Cunningham, Merce (1973). Personal communication.

Dance Notation Bureau (1973). Catalogue. New York: Dance Notation Bureau.

Dunn, John (1973). Personal communication.

Eshkol, Noa and Abraham Wachmann (1958). Movement Notation. London: Weidenfeld and Nicolson.

Eshkol, Noa; Peter Melvin; Jean Michl; Heinz Von Foerster and Abraham Wachmann (1970). Notation of Movement. Urbana, Ill.: University of Illinois, Department of Electrical Engineering, Biological Computer Laboratory, Report B.C.L. 10.0.

Fulkerton, Louis (no data, circa 1967). A Dance Notation Computer Program. Unpublished proposal submitted to the Dance Notation Bureau.

Hackney, Peggy; Sally Manno, and Muriel Topaz (1970). Correspondence Course in Labanotation. New York: Dance Notation Bureau.

Hackney, Peggy (1972). Personal communication.

Hutchinson, Ann (1967). "Choreography and Computers: A Reply," Dance Magazine. 41,(1), 45-46 and 81-82.

Hutchinson, Ann (1970). Labanotation: The System of Analyzing and Recording Movement. (second edition). New York: Theatre Arts Books.

Hutchinson, Ann (1972). Personal communication.

Jay, Leticia (1957). "A Stickman Notation," Dance Observer. (1), 7-8.

Kallis, S. (1969). "Motion Picture Animation by Computer," Computer and Automation. 18, (12), 30-34.

Laban, Rudolph. (1956). Principles of Dance and Movement Notation. London: Macdonald and Evans.

McCrea, William (1953). Analytical Geometry of Three Dimensions. Edinburgh: Oliver and Boyd.

McGraw, Charles (1964). Scoreography. Ann Arbor, Mich.: Edward Brothers (Lithographers).

Morris, Margaret (1928). The Notation of Movement. London: Kegan Paul, Trench Truber and Co.

Nikolais, Alwin (1948). "A New Approach to Dance Notation," Theatre Arts. 32, (2), 63-66.

Preston-Dunlop, Valerie (1969). "A Notation System for Recording Observable Motion," International Journal of Man-Machine Studies. 1, 361-386.

Seigel, S. (1956). Nonparametric Statistics for the Behavioral Sciences. London: McGraw-Hill Book Co.

Shapiro, Marla (1972). Personal communication.

Stepanov, V.I. (1892). Alphabet of Movement of the Human Body. Translated by Raymond Lister from the French Edition, Brooklyn: Dance Horizons, 1969.

Tetley, Glen (1948). "Choroscript," Dance Observer. 15, (9), 116-117.

Turnbaugh, Douglas B. (1970). "Dance Notation: Potential and Problems," Dance Scope. Spring 1970, 39-47.

APPENDIX A

PROGRAM LISTING OF LABAN

```
C                                 LABANOTATION RECONSTRUCTION
C
C
C THIS PROGRAMME READS AND INTERPRETS SPECIALLY CODED LABANOTATION AND PRODUCES
C THE STICKFIGURE REPRESENTATION OF THE COMMANDS
C
C
C DIMENSION STATEMENTS
      DIMENSION JPHIA(35),JPSIA(35),JDURA(35),JTTA(35),JMOV(35)
      DIMENSION LPRS1(35),LPRS2(35),LNTHL(35),JTDIS(35),JTPRX(35)
C     (DIMENSIONED BY NUMBER OF LIMBS: NLMS)
      DIMENSIONJPRS1(23),JPRS2(23),LIMBJ(23),ISDUR(23),JSDUR(23)
       DIMENSION IX(3,30), IEXCO(3,6), IEXTL(3)
C     (DIMENSIONED BY NUMBER OF JOINTS: NJTS)
      DIMENSION LIMNO(13),ISBAS(13),JBND(13)
C     (DIMENSIONED BY NUMBER OF COMPLEX LIMBS: NPRTS - NSIMP)
      DIMENSION IDNAM(9),IDANG(9),ILNAM(3),ILANG(3)
C     (DIMENSIONED BY NUMBER OF DIRECTIONS, LEVELS)
      DIMENSION IS(8),IXD(3),ISIMP(40),ICAM(9)
      DIMENSION IDR(6),ILVL(6)
      DIMENSION MOD1(10),MOD2(10),MODJBT(10),XI(3,3)
      DATA IBLANK/' '/
C     (WORKING STORAGE)
C
C ------------------------------------INCNT------------------------------------
C INITIALIZE BEAT AND FRAME COUNTERS
      API=3.14159
      JB=0
      JMOD=0
      FRAME=1.0
```

```
C
C READ IN PREDEFINED DATA
C NUMBER OF LIMBS; SIMPLE COMPONENTS OF LIMBS; JOINTS; REPRESENTED BY
C NLMS, NSIMP, NJTS; MAXIMUM DRAWING ERROR (MXERR), AND BOUND END OF
C TORSO (JTBND)
      READ(5,10)NSIMP,NLMS,NJTS,MXERR,JTBND
   10 FORMAT(12I5)
C
C READ IN LIMB CONSTANTS: PRESIGNS FOR LIMBS(LPRS1,LPRS2),LENGTH OF LIMB
C (LNTHL), DISTAL AND PROXIMAL JJINTS FOR LIMB (JTDIS,JTPRX),
C NUMBER OF COMPONENT PARTS (LN), LISTING OF COMPONENT PARTS(IS) ---
C ONLY FOR COMPLEX LIMBS.  FOR SIMPLE LIMBS, IS(1), IS(2), IS(3) ARE
C USED TO CONTAIN ASSUMED INITIAL DEFAULT VALUES OF PHI, PSI, THETA
C JTB IS THE NEW BOUND JOINT IN THE TORSO IF THIS IS CHANGED
   19 IBAS=0
      DO 45 LL=1,NLMS
      JCURA(LL)=0
      JMOV(LL)=0
      READ(5,20)L,LPRS1(L),LPRS2(L),LNTHL(L),JTDIS(L),JTPRX(L),LN,IS,JTB
      WRITE(6,20)L,LPRS1(L),LPRS2(L),LNTHL(L),JTDIS(L),JTPRX(L),LN,IS,JT
     B
   20 FORMAT(I5,2X,A2,A1,13I5)
      IF(LN-1)40,40,25
   25 MM=L-NSIMP
      LIMNO(MM)=LN
      DO 30 LS=1,LN
   30 ISIMP(IBAS+LS)=IS(LS)
      ISBAS(MM)=IBAS
      IBAS=IBAS+LN
      IF(JTB)35,33,35
   33 JTB=JTBND
   35 JEND(M3)=JTB
      GOTO 45
   40 JPHIA(L)=IS(1)
      JPSIA(L)=IS(2)
      JTTA(L)=IS(3)
   45 CONTINUE
      READ(5,50) IEXCO,IEXTL
50      FORMAT(13I4,3I2).
C
C
C
C READ IN RELATED LIMBCONSTANTS FOR USE WITH ROTATIONS
C READ IN JOINT CONSTANTS: PRESIGNS FOR JOINTS(JPRS1,JPRS2),CORRESPONDNG
C LIMB REFERRED TO BY PRESIGN(LIMBJ), DIRECTIONAL ANGLE(PHI) CHANGES
C DIRECTION(L/N) -IDANG,IDNAM, DIRECTIONAL ANGLE (PSI) CHANCES LEVEL
C (L/N) - ILANG, ILNAM
      DO 51 JJ=1,NJTS
      JSDUR(JJ)=0
      READ(5,20)J,JPRS1(J),JPRS2(J),LIMBJ(J)
      WRITE(6,20)J,JPRS1(J),JPRS2(J),LIMBJ(J)
   51 CONTINUE
      READ(5,60)IDNAM,ILNAM,IDANG,ILANG
      WRITE(3)NSIMP,NLMS,NJTS,JTBND,MXERR,
     *  (LNTHL(L),JTDIS(L),JTPRX(L),L=1,NSIMP) ,IEXCO,IEXTL
   60 FORMAT(9(3X,A2),3(4X,A1)/12I5)
C
C READ IN DANCE CONSTANTS AND CAMERA POSITIONS
```

```
C BEATSTEPS/BEAT (IBSBT), TIME SIGNATURE(KYTIM), FRAMES/SEC(ICINE),
C IPLOT IS THE NUMBER OF BEATS IN THE DANCE
C BEATSTEPS/SEC(BSPSC), FRAMES/BEATSTEP(FRATO) ARE COMPUTED
      READ(5,10)IBSBT,KYTIM,ICINE,IPLOT
      READ(5,10) ICAM
      BSPSC=FLOAT(IBSBT*KYTIM)/60.
      FRATO=FLOAT(ICINE)/BSPSC
      BSPBT=IBSBT
      WRITE(3) IBSBT,KYTIM,ICINE,IPLOT,ICAM
C
C
C----------------------------------INJ----------------------------------------
C READING IN THE COMMANDS
C NT IS KEY FOR SUPPORT OR GESTURE, AND IS 0 FOR END OF DATA
C IBEAT, BTFR ARE BEAT NUMBER AND PART OF BEAT; IBTS, DURFR ARE DURATION
C IN BEATS ~~AND~~ PART OF A BEAT
C IST IS THE STARTING BEATSTEP, IDUR IS THE DURATION IN BEATSTEPS
C IEND=1 SIGNALS END OF INPUT DATA
  100 READ(5,111)NT,IBEAT,BTFR,IPRS1,IPRS2,IBTS,DURFR, MODIF ,
       (IDR(I),ILVL(I),I=1,6)
 111  FORMAT(I2,I3,F3.2,1X,A2,A1,I3,F3.2, I4,6(1X,A2,A1))
      BE=FLOAT(IBEAT)+BTFR
      DE=FLOAT(IBTS)+DURFR
      IST=BE*BSPBT+0.001
      IDUR=DE*BSPBT+0.001
  124 IF(IST-JB)199,125,200
 125  WRITE(6,112)BE,IPRS1,IPRS2,DE,(IDR(I),ILVL(I),I=1,6),NT,MODIF
 112  FORMAT(F6.2,1X,A2,A1,F6.2,6(1X,A2,A1) ,2I4)
       IF(NT)130,131,131
 130  JB=-1
      WRITE(3)(JB,J=1,113)
      REWIND 3
      WRITE(6,129)
129    FORMAT(' LEAVING STAGE ONE')
      STOP
C
C
C =========================================CNVPS============================
C CONVERT PRESIGN INTO NUMBER OF BODYPART JBPT, WITH JTP=0 FOR A JOINT
C AND JTP=1 FOR A LIMB
C  SOME OF THE STEPS ARE ALSO SHOWN IN FORTRAN II
  131 JTP=0
      DO 135 JBPT=1,NJTS
      IF(IPRS1.NE.JPRS1(JBPT))GOTO 135
C     IF(IPRS1-JPRS1(JBPT))135,133,135
  133 IF(IPRS2.EQ.JPRS2(JBPT))GOTO 150
C 133 IF(IPRS2-JPRS2(JBPT))135,150,135
  135 CONTINUE
      JTP=1
      DO 145 JBPT=1,NLMS
      IF(IPRS1.NE.LPRS1(JBPT))GOTO 145
C     IF(IPRS1-LPRS1(JBPT))145,141,145
  141 IF(IPRS2.EQ.LPRS2(JBPT))GOTO 150
C 141 IF(IPRS2-LPRS2(JBPT))145,150,145
  145 CONTINUE
C ERROR: INVALID PRESIGN
      IER=1
      WRITE(6,147)IER,IPRS1,IPRS2
  147 FORMAT(I5,1X,A2,A1)
      STOP
```

```
C ===================================END OF CNVPS========================
C
150    IF(NT)199,180,155
C
C IF THE NT INDICATES A GESTURE
155    IF(JTP)161,160,161
160    JBPT=LIMBJ(JBPT)
161    IF(NT-2)170,162,162
C STORAGE OF MODIFICATIONS
C   NT:     CODE FOR MODIFICATIONS STORED IN ARRAY, MOD1
C MODIF : DEGREES OF MODIFICATION, STORED IN ARRAY MOD2
C MODJBT: ARRAY OF BODYPARTS BEING MODIFIED
C JMOD IS THE COUNTER TO INDICATE THE NUMBER OF MOCIFIATIONS IN EACH
C BEATSTEP.
162    JMOD=JMOD+1
       MOD1(JMOD)=NT
       MOD2(JMOD)=MODIF
C STORAGE OF BODYPART, DEPENDING ON WHETHER IT'S A LIMB OR JOINT
       IF(JTP)163,163,164
C JOINT AS INPUT
163    MODJBT(JMOD)=JBPT
       GOTO 170
164    IF(JBPT-NSIMP)167,167,165
C (LIMB IS A COMPLEX ONE)
165     IBAS=ISBAS(JBPT-NSIMP)
       JJC=ISIMP(IBAS+1)
       MODJBT(JMOD)=JJC
       GOTO 170
C ERROR:
167    WRITE(6,168)
168    FORMAT(' CONTRACTION OF SIMPLE LIMB')
       STOP
```

```
C ================================CNVOL================================
C CONVERT DIRECTION, LEVEL INTO DIRECTIONAL LEVELS  AND
C DETERMINATION OF INTERMEDIATE DIRECTIONS
C INTERMEDIATE DIRECTUONS ARE GIVEN AS A SERIES OF COMPONENTS OF THE
C DESIRED POSITION. TO FIND THE DESIRED POSITION, THE AVERAGE OF THE
C  COMPONENT DIRECTIONS IS TAKEN
C IF THERE IS MORE THAN ONE DIRECTION/LEVEL COMMAND, INTERMEDIATE
C  DIRECTIONS HAVE BEEN GIVEN.
C
C
C
170     JPHD=360
        JPSD=0
        DO 179 J=1,6
C  THERE WILL NOT BE MORE THAN SIX INTERMEDIATE COMPONENT DIRECTIONS
C THIS WOULD REPRESENT THE ABILITY TO SPECIFY UP TO 7  DEGREES CHANGE
        IDRJ=IDR(J)
        IF(IDRJ.EQ.IBLANK) GOTO 1736
C THE END OF THE COMPONENT DIRECTION COMMANDS HAS BEEN REACHED
C       IF(IDRJ-40) 1710,1736,1710
C  NOT BLANK
1710     DO 171 K=1,9
        IF(IDRJ.EQ.IDNAM(K)) GOTO 173
C       IF(IDRJ-IDNAM(K))171,173,171
171      CONTINUE
C ERROR STATEMENT: INVALID DIRECTION
        IER=2
        WRITE(6,172)IER,IDR
  172 FORMAT(I5,1X,A2)
        STOP
C TAKE AVGE OF DIRECTIONS READ SO FAR.  NOTE THAT PLACE DOES NOT
C AVERAGE -- OTHER DIRECTION REMAINS
```

```
173    IF(K-9)1731,1735,1731
1731     IF(JPHD-360)1733,1732,1733
1732    JPHD=IDANG(K)
       GOTO 1735
1733    JPHD=JPHD+IFIX(FLOAT(IDANG(K)-JPHD)/FLOAT(J))
1735    DO 174 N=1,3
       IF(ILVL(J).EQ.ILNAM(N)) GOTO 175
C      IF(ILVL(J)-ILNAM(N)) 174,175,174
174     CONTINUE
C ERROR STATEMENT: INVALID LEVEL
       IER=3
       WRITE(6,172)IER,ILVL
       STOP
175    IA=ILANG(N)
       IF(K-9)178,177,178
177    IA=IA+IA
C PLACE PSI IS DOUBLED
178     JPSD=JPSD+IFIX(FLOAT(IA-JPSD)/FLOAT(J))
179      CONTINUE
1736   IF(J-2)1740,1737,1737
C ********************************END OF CNVDL************************
C
C
1737   JMCV(JBPT)=1
1738    JPHIA(JBPT)=JPHD
       JPSIA(JBPT)=JPSD
1739   JDURA(JBPT)=IDUR
       GOTO 100
C NO DIRECTION LEVEL COMMANDS GIVEN AT TIME OF MODIFICATION
C THE J WILL BE EQUAL TO ONE
```

```
1740  IF(JBPT-NSIMP)1739,1739,1741
1741  JPHD=JPHIA(JJC)
       JPSD=JPSIA(JJC)
      GOTO 1733
C
C IF THE NT INDICATES A SUPPORT
  130 IF(JTP)134,132,134
C SUPPORT IS A JOINT (JTP=0)
  132 JSDUR(JBPT)=IDUR
      GOTO 100
C SUPPORT IS A LIMB(JTP=1)
  134 JSDUR(JTPRX(JBPT))=IDUR
      JSDUR(JTDIS(JBPT))=IDUR
      GOTO 100
C ERROR: TIME SEQUENCE CHECK
  190 WRITE(6,197)
  197 FORMAT(25X,'STOP1000')
      STOP
C
C-------------------------------------CONSIM--------------------------------------
C CHANGES THE COMPLEX LIMBS TO SIMPLE BODYPARTS FOR BEATSTEP J
C
  200 LL=NSIMP+1
      DO 240  M=LL,NLMS
      MP=JDURA(M)
      IF(MP)215,240,215
  215 MM=M-NSIMP
      JTBND=JBND(MM)
      NP=LIMNO(MM)
      IBAS=ISBAS(MM)
      JPH=JPHIA(M)
      JPS=JPSIA(M)
      DO 230 N=1,NP
      JSMPL=ISIMP(N+IBAS)
      JS=JDURA(JSMPL)
      IF(JS) 230, 220, 230
  220 JDURA(JSMPL)=NP
      JMOV(JSMPL)=1
      JPHIA(JSMPL)=JPH
      JPSIA(JSMPL)=JPS
  230  CONTINUE
  240  CONTINUE
C
C-------------------------SET UP DEFAULT VALUES FOR INITIAL POSITION------
C
  249 IF(JB) 280,250,280
  250  DO 270 KL=1,NSIMP
      IF (JDURA(KL)) 270,260,270
  260  JDURA(KL)=1
  270 CONTINUE
C
C ---------AMOUNT OF ROTATION ALONG AXIS OF LIMB WORKED OUT HERE----------------
C ************
C *** TORSO ***
C ************
C NO TORSO ROTAIONS ALLOWED FOR YET
 280    DO 287 K=1,10
      IF(JMOV(K))285,287,285
 285  JTTA(K)=0
 287  CONTINUE
```

```
C
C ***    ******
C*** LIMBS ***
C ************
      DO 400 K=11,20,3
      JMOVS=JMOV(K)+JMOV(K+1)+JMOV(K+2)
C IF NO MOVEMENT KEEP OLD THETAS, ELSE CALCULATE THEM IN 'ROTAT'
      IF(JMOVS)290,400,290
290    CALL ROTAT(JPHIA,JPSIA,JTTA,K)
400   CONTINUE
C----------------------------CONTRACTIONS-----------------------------------
      IF(JMOD) 333,333,500
500   DO 599 JM=1,JMOD
      MODFN=MODI(JM)
      IF(MODFN-3)501,501,600
501   MODB=MODJBT(JM)
C AB IS LENGTH OF FIRST LIMB
C B IS DISTAL END OF FIRST LIMB, A IS PROXIMAL END OF THIS LIMB
      AB= LNTHL(MODB)
      IB=JTDIS(MODB)
       IA=JTPRX(MODB)
C MODB1 IS THE ADJACENT LIMB DOWNWARDS. BC IS ITS LENGTH, C, ITS DISTAL END
      MODB1=MODB+1
      BC =LNTHL(MODB1)
      IC=JTDIS(MODB1)
C CHECK TO SEE THAT DISTAL OF FIRST IS EQUAL TO PROXIMAL OF SECOND LIMB
      IF(JTPRX(MODB1)-IB)620,510,620
510    JDURA(MODB1)=JDURA(MODB)
C MODB2 IS 'AUXILLARY LIMB' EG. LIKE HAND AND FOOT
      MODB2=MODB+2
      IF(JTPRX(MODB2)-IC)535,520,535
```

```
C THE AUXILLARY LIMB IS ATTACHED TO THE SECOND LIMB
520   IF(JPHIA(MODB1)-JPHIA(MODB2))530,525,530
525   IF(JPSIA(MODB1)-JPSIA(MODB2))530,527,530
530   IF(JDURA(MODB2))535,527,535
527     JDURA(MODB2)=JDURA(MODB)
C JMODT =1 REPRESENTS AN AUXILLRY LIMB THAT MUST BE CHANGED AFTER MODIFICATION.
C  JMODT=0 MEANS  THAT THERE IS NO CHANGE TO THE AUX LIMB AFTEF MODIFICATION
C OR THAT WE ARE DEALING WITH CONTRACTION AT ANKLE OR AT WRIST.
C CD IS LENGTH OF AUXILLIARY (3RD) LIMB IF IT IS INVOLVED IN CONTRACTION
      JMODT=1
      CD= LNTHL(MODB2)
      GOTO 540
535    JMODT=0
      CD=0.0
C
C CALCULATION OF DEGREES OF MODIFICATION
C THE DEGREES ARE GIVEN AS INTEGERS WHICH DEFINE THE ANGLE OF CONTRACTION BETA
540   DEG=MOD2(J1)
      IF(DEG-6.0)560,550,560
550   PIX=DEG*API*23./24.
      GOTO 565
560   PIX=DEG*API
565   BETA=API-PIX/6.
      CALL COSIN (BETA,CSBTA,SNBTA)
C
C CALC DIRECTION COSINES OF UNCONTRACTED LIMB, AND OF NORMAL TO LIMB,
C AND SIGN OF NORMAL (- FOR CONTRACTION AT KNEE, ELSE +)
      CALL VECNML(JPHIA(MODB),JPSIA(MODB),JTTA(MODB),DCLX,DCLY,DCLZ,
     * DCNX,DCNY,DCNZ)
```

```
      FNML=SNML(MODB)
      IF(MODFN-2)600,2570,3007
C=====================================FOLDS=============================
C CALC. DIRN COSINES DCRQ OF FOLDED LIMB MODB1
2570  PR=AB
      QR=BC
      PS=PR-QR*CSBTA
      SQ=QR*SNBTA
      CALL DCS(PR,QR,0.0,PS,SQ, DCLX,DCLY,DCLZ,   FNML,   DCNX,DCNY,DCNZ,
     1  DCRQX,DCRQY,DCRQZ,   DCPQX,DCPQY,DCPQZ)
C GET PHI,PSI,THETA OF MODB1 AND POSSIBLY MODB2 FROM D.C.'S DCRQ
     * CALL PHPSTA(DCRQX,DCRQY,DCRQZ,DCNX,DCNY,DCNZ,
     1                           JPHIA,JPSIA,JTTA,MODB1,JMODT)
      GOTO 599
C
C====================FLEXION=============================================
C CHECKING FOR SUPPORT IN LIMB
3007  IF(JSDUR(IB   ))3030,3030,3550
3030  IF(JSDUR( IC   ))3570,3570,3560
C IA IS THE PROX JT. OF MODB, IB THE PROX. JT. OF MODB1, IC THE PRX. JT. OF
C MODB2       THEY ARE SET AROUND STATEMENT 501 IN MAIN PROGRAMME
3560  JMODT=0
      CD=0.0
C IF IC IS A SUPPORT, THE AUXILLIARY LIMB CANNOT BE INVOLVED IN CONTRACTION
      GOTO 3570
C ERROR
3550    WRITE(6,3551)
3551  FORMAT(' CANNOT FLEX LIMB AT A SUPPORTED JOINT')
      STOP
```

```
3570  QR=BC + 0.5*CD
      PQ=AB
      XT=(API-BETA)/2.
      TANNE=(PQ-QR)*TAN(XT)/(PQ+QR)
      YT=ATAN(TANNE)
      P=XT-YT
      R=XT+YT
      CALL COSIN(P,CSP,SNP)
      PR=QR*SNBTA/SNP
      PS=PQ*CSP
      SQ=PQ*SNP
C CALC DIRN COSINES DCPQ OF IST LIMB MODB,
C AND NEGATIVES DCRQ OF DIRN COSINES OF 2ND LIMB MODB1
      CALL DCS(PR,QR,PQ,PS,SQ,DCLX,DCLY,DCLZ,  -FNML,  DCNX,DCNY,DCNZ,
     *  DCRQX,DCRQY,DCRQZ,  DCPQX,DCPQY,DCPQZ)
C GET PHI,PSI,THETA OF MODB FROM D.C.'S DCPQ
      CALL PHPSTA(DCPQX,DCPQY,DCPQZ,  DCNX,DCNY,DCNZ,  JPHIA,JPSIA,JTTA,
     *  MODB,0)
C GET PHI,PSI,THETA OF MODB1 AND POSSIBLY MODB2 FROM -VE OF DCRQ
      CALL PHPSTA(-DCRQX,-DCRQY,-DCRQZ,  DCNX,DCNY,DCNZ,  JPHIA,JPSIA,
        JTTA,MODB1,JMODT)
C
599   CONTINUE
      JMOD=0
       GOTO 338
C ERRORS
600    WRITE(6,610) MODFN
610   FORMAT(' INVALID NT ASSIGNMENT',I4)
      STOP
C ERROR
620    WRITE(6,621) IB
```

```
 621   FORMAT(' CONTRACTION AT JOINT',I4,' IMPOSSIBLE')
        STOP
 338   WRITE(3)JB,(JPHIA(L),JPSIA(L),JTTA(L),JDURA(L),L=1,NSIMP),JSDUR,J
       TBND
       WRITE(6,339)JB,(JPHIA(L),L=1,NSIMP),(JPSIA(L),L=1,NSIMP),
                     (JTTA(L), L=1,NSIMP),(JDURA(L),L=1,NSIMP)
 339   FORMAT(I5,'  PHI ',22I5/5X,'  PSI ',22I5/5X,'THETA  ',22I5/
         5X,'  DUR  ',22I5//)
       JB=JB+1
C*****************************************************************************
C*****************************************************************************
C ZERO ALL COMPLEX LIMBS
       DO 920 J=1,NLMS
       JMOV(J)=0
       IF(J-NSIMP) 905,905,903
 903    JDURA(J)=0
       GOTO 920
 905   JDM=JDURA(J)-1
       IF(JDM) 920,910,910
 910     JDURA(J)=JDM
 920    CONTINUE
        DO 940 J=1,NJTS
        JDM=JSDUR(J)-1
       IF(JDM) 940,930,930
 930    JSDUR(J)=JDM
 940   CONTINUE
       GOTO 124
       END
```

```
C -----SUBROUTINE ARMS(IPHI,IPSI,K,ITA)      ------------------------------
        SUBROUTINE ARMS(IPHI,IPSI,K,ITA)
      IF(IPHI)20,35,20
20    IF(IPHI-180)30,35,30
30    IF(IPHI-360)40,35,40
C FRONT,BACK AND PLACE HAVE THETA=90
35    ITA=90
      GOTO 60
C ALL OTHERS HAVE THETA DEPENDING ON PSI
40    ITA=IPSI+90
C COMPLEX LIMBS
60    IF(K-23)70,90,100
70    IF(K-27)80,90,90
80    IF(K-14) 90,100,100
C RIGHT SIDE OF THE BODY HAS DIFFERENT THETA
90    ITA=360-ITA
100   RETURN
      END
C                         LEGS
      SUBROUTINE LEGS(IPHI,K,ITA)
      IF(IPHI-90)10,20,20
10     ITA=0
      GOTO 50
20    IF(IPHI-270)40,10,10
40    ITA=540-2*IPHI
50    RETURN
      END
C                             ROTAT
      SUBROUTINE ROTAT(JPHI,JPSI,JTT,K)
      DIMENSION JPHI(22),JPSI(22),JTT(22)
      KPH=JPHI (K)
      KPPH=JPHI (K+1)
      KPPPH=JPHI (K+2)
      KPS=JPSI (K)
      KPPS=JPSI (K+1)
      KPPPS=JPSI (K+2)
C CHECK TO SEE IF FIRST PAIR SAME PHI,PSI
      IF(KPH-KPPH)295,292,295
292   IF(KPS-KPPS)295,370,295
C DIFFERENT UPPER,LOWER DIRECTIONS
295   FNML=SNML(K)
C ARMS--NORMAL POSITIVE X-DIRN FOR UPPER LOWER LIMB PAIR
C LEGS--NORMAL NEGATIVE X-DIRN FOR UPPER,LOWER LIMB PAIR
C CALCULATE THETA FOR UPPER LOWER
305   CALL CHCK(KPH,KPS,KPPH,KPPS,FNML,ITA,ITAP)
      JTT (K)=ITA
      JTT (K+1)=ITAP
C ARE LOWER,EXTREMITY SAME PHI,PSI?
      IF(KPPH-KPPPH)315,310,315
310   IF(KPPS-KPPPS)315,325,315
C DIFFERENT DIRECTIONS LOWER, EXTREMITY (DIFFERENT ALSO UPPER,LOWER)
315   CALL CHCK(KPPH,KPPS,KPPPH,KPPPS,+1.0,ITAP,ITAPP)
      JTT (K+1)=ITAP
320   JTT (K+2)=ITAPP
      GOTO 330
C SAME LOWER,EXTREMITY (DIFFERENT UPPER,LOWER)
325   ITAPP=ITAP
      GOTO 320
C CHECK IF ARMS
330   IF(K-16)335,335,399
C YES, ARMS, MUST CHECK ADMISSIBLE RANGE OF LOWER ARM THETA
335   IF(K-13)340,340,345
```

```
C RIGHT SIDE
340    LOW=ITA-180
       GOTO 350
C LEFT SIDE
345    LOW=ITA
350    IDELTA=ITAP-LOW
C      (0<ITAP<360; -180<LOW<360;  SO -360<IDELTA<540)
       IF(IDELTA.GE.360)IDELTA=IDELTA-360
       IF(IDELTA.LT.0)IDELTA=IDELTA+360
C      (0<IDELTA<360)
C CHECK RANGE
       IF(IDELTA.LE.180)GOTO 399
C OUT OF RANGE, SUBTRACT 180 TO GET A POSSIBLE POSITION
365    IF(ITAP.LT.180)ITAP=ITAP+360
       JTT (K+1)=ITAP-180
       IF(ITAPP.LT.180)ITAPP=ITAPP+360
       JTT (K+2)=ITAPP-180
       GOTO 399
C
C THE UPPER, LOWER ARE SAME, TRY LOWER, EXTREMITY
370    IF(KPPH-KPPPH)375,372,375
372    IF(KPPS-KPPPS) 375,380,375
C DIFFERENT LOWER, EXTREMITY (SAME UPPER, LOWER)
375    CALL CHCK(KPPH,KPPS,KPPPH,KPPPS,+1.0,ITAP,ITAPP)
       JTT (K)=ITAP
       JTT (K+1)=ITAP
       JTT (K+2)=ITAP P
       GOTO 399
C
C ALL THREE SAME PHIS, PSIS
```

```
C CHECK IF LEGS OR ARMS
380    IF(K-17)390,385,385
C LEGS
385    CALL LEGS(KPH,K,ITA)
       GOTO 395
C ARMS
390    CALL ARMS(KPH,KPS,K,ITA)
395    JTT (K)=ITA
       JTT (K+1)=ITA
       JTT (K+2)=ITA
399    RETURN
       END
C                 CHCK
       SUBROUTINE CHCK(IPHI,IPSI,IPHIPR,IPSIPR,FNML,ITA,ITAP)
C FINDS NORMAL TO FL, FM,FN TO LIMB PAIR (PHI,PSI) , AND (PHIPR,PSIPR)(EITH SIGN
C FNML) AND HENCE OBTAINS CORRESPONDING THETA AND THETAPR FROM 'CHECK' SUBROUTIN
C
       APHPR=RAD(IPHIPR)
       APSPR=RAD(IPSIPR)
       APH=RAD(IPHI)
       APS=RAD(IPSI)
       CALL  COSIN(APHPR,CSPHP,SNPHP)
        CALL COSIN(APSPR,CSPSP,SNPSP)
        CALL COSIN(APH,CSPH,SNPH)
        CALL COSIN(APS,CSPS,SNPS)
C
       VECX  =CSPS*SNPH
       VECY  =CSPS*CSPH
       VECZ  =-SNPS
```

```
      VECPX  =CSPSP*SNPHP*FNML
      VECPY  =CSPSP*CSPHP*FNML
      VECPZ  =-SNPSP*FNML
      CALL CROSP(VECX,VECY,VECZ,   VECPX,VECPY,VECPZ,   DCNX,DCNY,DCNZ)
      ITA=ITHETA(DCNX,DCNY,DCNZ,   CSPH,CSPS,SNPH,SNPS)
      ITAP=ITHETA(DCNX,DCNY,DCNZ,  CSPHP,CSPSP,SNPHP,SNPSP)
C (RESULTS ARE POSITIVE)
      RETURN
      END
C
      FUNCTION ITHETA(FL,FM,FN,CSPH,CSPS,SNPH,SNPS)
C THIS ROUTINE WAS ORIGINALLY SAME AS CHECK. THIS ROUTINE FINDS THE THETA WHICH,
C WHEN COMBINED WITH THE PHI, PSI YIELDS THE GIVEN NORMAL (FL,FM,FN)
      API=3.14159
       U=CSPH*SNPS
        V=SNPH*SNPS
C
C WORK OUT FIRST SET OF RATIOS
       P1=-(FM*CSPS)+FN*U
       P2=-FN*SNPH
      IF(ABS(P1)+ABS(P2)-0.001) 15,15,10
10    PONE=P1
      PTWO=P2
      GOTO 35
C
C IF THE FIRST RATIO ZERO, WORK OUT SECOND
C
15    P3=FL*CSPS-FN*V
      P4=-FN*CSPH
      IF(ABS(P3)+ABS(P4)-0.001) 25,25,20
20    PONE=P3
```

```
      PTWO=P4
       GOTO 35
C
C IF SECOND RATIO ZERO, WORK OUT THIRD
25    P5=SNPS*(FM*SNPH-FL*CSPH)
      P6=FM*CSPH+FL*SNPH
      IF(ABS(P5)+ABS(P6)-0.001) 33,33,30
30    PONE=P5
      PTWO=P6
CPONE=0,PTWO NOT 0
35    IF(PONE)120,110,120
110   IF(PTWO)115,199,115
115   TTAZO=API/2.
      GOTO 135
C
C FIRST = TO POSITIVE, PTWO=0
120   IF(PTWO)130,125,130
125   TTAZO=0.
      GOTO 135
C
C BOTH NOT ZERO
130   TANTH=PTWO/PONE
       TTAZO=ATAN(TANTH)
C THETA LIES BETWEEN -90 AND 90)
      IF(TTAZO)133,135,135
133   TTAZO=TTAZO+API
C
C USE N*N TO FIND TTAZO
135    CALL COSIN(TTAZO,CSTA,SNTA)
      IF(ABS(EN)-0.001) 160,160,140
```

```
140     CHCKN=-FN*CSPS*SNTA
         IF(ABS(CHCKN) -0.001) 160,160,142
142      IF(CHCKN)150,160,145
145     THETA=TTAZO
        GOTO 195
150     THETA=TTAZO+API
        GOTO 195
C
C USE M*M
160     IF(ABS(FM)-0.001)180,180,170
170       CHCKM= -(FM*SNPH*CSTA)-(FM*U*SNTA)
        IF(ABS(CHCKM)-0.001) 180,180,172
172     IF(CHCKM)150,180,145
C USE L*L
180     IF(ABS(FL)-0.001)  199,199,190
190     CHCKL=FL*CSPH*CSTA-(FL*V*SNTA)
        IF(ABS(CHCKL)-0.001)199,199,192
192     IF(CHCKL)150,199,145
195     ITHETA=IRND(THETA/0.017453)
        RETURN
C ERRORS
199     WRITE(6,200)
200       FORMAT('ALL NUMBERS ARE ZERO')
        STOP
33      WRITE(6,101)
101      FORMAT('ALL PSUMS ZERO')
        STOP
        END
C               PHPSTA
        SUBROUTINE PHPSTA(DCX,DCY,DCZ,  DCNX,DCNY,DCNZ,  JPHIA,JFSIA,JTTA,
      *  LIMB,JMODT)
         DIMENSION JPHIA(35),JPSIA(35),JTTA(35)
         SNPS=-DCZ
C        (-VE SIGN FOR PSI BECAUSE THAT CORRESPONDS TO +VE Z)
         CSPS=SQRT(1.0 - SNPS*SNPS)
         APSJ=ARCSN(SNPS)
C        (-PI/2 < APSJ < PI/2)
         IF(CSPS-0.001)30,30,10
 10      SNPH=DCX/CSPS
         CSPH=DCY/CSPS
         APHJ=ARCTN(SNPH,CSPH)
C        (0 < APHJ < 2*PI)
         ITA=ITHETA(DCNX,DCNY,DCNZ,   CSPH,CSPS,SNPH,SNPS)
         LAST=LIMB+JMODT
         DO 20 LIM=LIMB,LAST
         JPHIA(LIM)=IRND(APHJ/0.017453)
         JPSIA(LIM)=IRND(APSJ/0.017453)
         JTTA(LIM)=ITA
 20      CONTINUE
         RETURN
 30      WRITE(6,31)
 31      FORMAT(' ATTEMPT TO FIND PHI IN PHPSTA WITH PSI = +90 OR -90')
         STOP
         END
C
         FUNCTION ARCSN(Y)
         X=Y
         IF(ABS(X).LE.1.0)GOTO 20
         WRITE(6,11)X
 11      FORMAT(' ARCSN OF',F8.4)
         IF(X.GT.1.0)X=1.0
         IF(X.LT.-1.0)X=-1.0
 20      ARCSN=ARSIN(X)
         RETURN
         END
```

```
C
      FUNCTION ARCTN(X,Y)
      ARCTN=ATAN2(X,Y)
      IF(ARCTN)30,40,40
30    ARCTN=ARCTN+6.28318
40    RETURN
      END
C
      SUBROUTINE COSIN(A,CS,SN)
       CS=COS(A)
      SN=SIN (A)
       RETURN
       END
C
      FUNCTION IRND(X)
       RND=X+0.5
      IF(RND)10,20,20
10     RND =RND-1
   20 IRND=RND
      RETURN
      END
C
       FUNCTION RAD(I)
      IF(I-360) 10,20,10
   10 RAD=FLOAT(I)*0.017453
```

```
      RETURN
   20 RAD=0.0
      RETURN
      END
C
      FUNCTION SNML(MODB)
      IF(MODB-17)20,30,10
10     IF(MUDB-20)20,30,20
20     SNML=1.00
      RETURN
30    SNML=-1.00
      RETURN
       END
C           CROSP
      SUBROUTINE CROSP(X1,X2,X3,Y1,Y2,Y3,CROSS1,CROSS2,CROSS3)
C FINDS UNIT VECTOR NORMAL TO TWO GIVEN VECTORS
      CROS1=X2*Y3-X3*Y2
       CROS2=X3*Y1-X1*Y3
      CROS3=X1*Y2-X2*Y1
      SNTH=SQRT(CROS1*CROS1 + CROS2*CROS2 + CROS3*CROS3)
      CROSS1=CROS1/SNTH
      CROSS2=CROS2/SNTH
      CROSS3=CROS3/SNTH
      RETURN
      END
C                         DCS
      SUBROUTINE DCS(PR,QR,PQ,PS,SQ,DCPRX,DCPRY,DCPRZ,FNMULT,DCNX,DCNY,
     *  DCNZ,      DCRQX,DCRQY,DCRQZ,DCPQX,DCPQY,DCPQZ)
C AS INPUT TO THIS ROUTINE, PR,PS,SQ REQUIRED, IF QR>0.0,REQUIRE DCIS OF VECTOR
C RQ. IF PQ>0.0, REQUIRE DC'S OF PQ ((DC IS DIRECTIONAL COSINE)).
      DCNNX=DCNX*FNMULT
      DCNNY=DCNY*FNMULT
      DCNNZ=DCNZ*FNMULT
      CALL CROSP(DCNNX,DCNNY,DCNNZ,DCPRX,DCPRY,DCPRZ,DCSQX,DCSQY,DCSQZ)
      PQ1=PQ
      IF(PQ)10,10,20
10    PQ1=-1.0
20    DCPQX=(PS*DCPRX+SQ*DCSQX)/PQ1
      DCPQY=(PS*DCPRY+SQ*DCSQY)/PQ1
      DCPQZ=(PS*DCPRZ+SQ*DCSQZ)/PQ1
      IF(QR)40,40,30
30    DCRQX=(PQ1*DCPQX-PR*DCPRX)/QR
      DCRQY=(PQ1*DCPQY-PR*DCPRY)/QR
      DCPQZ=(PQ1*DCPQZ-PR*DCPRZ)/QR
40    RETURN
      END
C                  VECNML
      SUBROUTINE VECNML(IPH,IPS,ITA,DCVX,DCVY,DCVZ,DCNX,DCNY,DCNZ)
C THIS ROUTINE(VECNML) CALCULATES THE NORMAL FROM PHI,PSI,THETA
C IT FINDS THE DIRECTIONAL COSINES OF A LIMB AND OF THE NORMAL TO THE LIMB GIVEN
C THE THREE ANGLES, PHI,PSI,THETA.
       CALL COSIN(RAD(IPS),CSPS,SNPS)
       CALL COSIN(RAD(ITA),CSTA,SNTA)
      CALL COSIN(RAD(IPH),CSPH,SNPH)
      DCVX  =CSPS*SNPH
      DCVY  =CSPS*CSPH
      DCVZ   =-SNPS
      SS=SNPS* SNTA
      DCNX    =CSPH*CSTA-SNPH*SS
      DCNY    =-SNPH*CSTA-CSPH*SS
      DCNZ    =-CSPS*SNTA
      RETURN
      END
/*
//A198ZELL JOB (            ),'LABAN2    '
//STEP2  EXEC   FORTGCL,LDISP='OLD,KEEP',LOAD='SC.A1148.LBN(LABAN2)'
//FORT.SYSIN  DD  *
      DIMENSION IPHIA(22),IPSIA(22),IDURA(22),ITTA(22)
      DIMENSION JPHIA(22),JPSIA(22),JDURA(22),JTTA(22)
      DIMENSION JPHKT(22),JPSKT(22),JTTKT(22)
      DIMENSION                         LNTHL(22),JTDIS(22),JTPRX(22)
      DIMENSION                                    ISDUR(23),JSDUR(23),IC(23)
      DIMENSION  XD(3) , IXD(3)
       DIMENSION IX(3,35), IEXCO(3,6), IEXTL(3)
      DIMENSION ICAM(9)
      DATA DASH/'----'/
      WRITE(6,10)
 10   FORMAT(' ENTERING STAGE TWO')
      READ(3)NSIMP,NLMS,NJTS,JTBND,MXERR,
     *  (LNTHL(L),JTDIS(L),JTPRX(L),L=1,NSIMP) ,IEXCO,IEXTL
      WRITE(6,1984)
     *      NSIMP,NLMS,NJTS,JTBND,MXERR,
     *  (LNTHL(L),JTDIS(L),JTPRX(L),L=1,NSIMP) ,IEXCO,IEXTL
 1984 FORMAT(5I5/(3I5))
      READ(3) IBSBT,KYTIM,ICINE,IPLOT,ICAM
      WRITE(6,1985)IBSBT,KYTIM,ICINE,IPLOT,ICAM
 1985 FORMAT(13I5)
      BSPSC=FLOAT(IBSBT*KYTIM)/60.
      FRATO=(FLOAT(ICINE))/BSPSC
      BSPBT=IBSBT
      FRAME=1.0
      WRITE(4) IBSBT,KYTIM,ICINE,IPLOT,ICAM
100    READ(3)JB,(JPHIA(L),JPSIA(L),JTTA(L),JDURA(L),L=1,NSIMP),JSDUR,J
     TRND
      WRITE(6,101)JB,JPHIA,JPSIA,JTTA
101   FORMAT(I5  ,'  PHI',2215/5X,'  PSI  ' ,2215/5X,'THETA  '  ,2215//)
      IF(JB)130,280,280
130   KFR=-1
      JBEAT=JB/IBSBT
      WRITE(4)KFR,JB,JBEAT,IX
      REWIND 4
      WRITE(6,140)
140   FORMAT(' LEAVING STAGE TWO')
      STOP
  230 FJBF=FLOAT(JB)*FRATO
  290 TMFRN=FRAME-1.0-FJBF
C
C
C-----------------------------FRMKT------------------------------------------
C
C FRMKT FINDS THE CURRENT TIME DIRECTIONAL ANGLE FOR EACH FRAME AT EACH BEAT
C
  400 DO 499 IBPRT=1,NSIMP
      JDAB=JDURA(IBPRT)
      DJFR=FLOAT(JDAB)*FRATO
      IF(TMFRN) 410,450,450
C FRAME OVERLAPS TWO BEATS
  410 IDAB=IDURA(IBPRT)
      IF(IDAB+JDAB) 420,499,420
  420 IF(JDAB) 430,493,430
```

```
C SOMETHING IN SECOND
  430 IF(IDAB-1) 460,440,460
C SECOND IS NOT JUST CONTINUATION OF FIRST
  440 DURIF=DJFR/(1.0+TMFRN)
      IF(IDAB) 470,480,470
C NO OVERLAP
  450 IF(JDAB) 460,499,460
C SOMETHING IN SECOND AND NO OVERLAP, OR SOMETHING CONTINUING IN BOTH
  460 DURIF=DJFR-TMFRN
      IF(JB) 430,475,480
C OVERLAPS WITH SOMETHING IN BOTH WHICH IS NOT CONTINUOUS
  470 IPH=IPHIA(IBPRT)
      IPS=IPSIA(IBPRT)
      ITA=ITTA(IBPRT)
      GO TO 490
C REATSTEP 0
  475 IPH=JPHIA(IBPRT)
      IPS=JPSIA(IBPRT)
       ITA=JTTA(IBPRT)
      GO TO 490
C SOMETHING IN SECOND AND NO OVERLAP (NOT FRAME 1) OR CONTINUATION WITH
C OVERLP
  480 IPH=JPHKT(IBPRT)
      IPS=JPSKT(IBPRT)
      ITA=JTTKT(IBPRT)
 490  CALL NEWDA(IBPRT,IPH,IPS,ITA,JPHIA(IBPRT),JPSIA(IBPRT),
     *  JTTA(IBPRT),DURIF,JPHKT,JPSKT,JTTKT,LNTHL)
      GOTO 499
C OVERLP BUT NOTHING IN SECOND
 493  JPHKT(IBPRT)=IPHIA(IBPRT)
      JPSKT(IBPRT)=IPSIA(IBFRT)
      JTTKT(IBPRT)= ITTA(IBPRT)
  499 CONTINUE
```

```
C
C
C
C---------------------------COORDINATES-----------------------------------------
C CALCULATES THE COORDINATES OF EACH JOINT AT EACH FRAME
C
  600 DO 610 J=1,NJTS
  610 IC(J)=0
      IF(JSDUR(JTBND)) 620,620,630
C ARRANGEMENT OF TORSO HIERARCHY
  620 DO 625 K=1,3
  625 IX(K,JTBND)=0
  630 IC(JTBND)=1
      DO 780 L=1,NSIMP
      IF(L-4) 640,640,680
  640 IF(JTBND-3) 680,650,670
  650 IF(L-2) 660,660,680
 660  LL=3-L
      GO TO 690
  670 LL=5-L
      GO TO 690
  680 LL=L
  690 JBD=JTDIS(LL)
       JFR=JTPRX(LL)
      IF(IC(JBD)) 710,700,710
  700 JT=JBD
      JBD=JFR
      JFR=JT
C CALCULATION OF DISPLACEMENT IN ABSOLUTE TERMS OF THE PLOTTER
```

```
  710 CALL DISP(JPHKT(LL), JPSKT(LL), LNTHL(LL), X1,X2,X3)
      XD(1)=X1
       XD(2)=X2
      XD(3)=X3
      ISP=JSOUR(JFR)
      DO 760 K=1,3
      IXDF=IRND(XD(K))
      NEWX=IX(K,JBD)+IXDF
      IF(ISP) 715,750,715
C SUPPORT ENCOUNTERED: MODIFY PREVIOUSLY CALCULATED RELATIVE COORDS TO
C REAL COORDINATES USING KNOWN REAL COORDS OF SUPPORT
C ON BEATSTEP ZERO, ASSUME SUPPORTING JOINTS ARE LOCATED ON FLOOR: Z=0)
  715 IF(JB)720,716,720
  716 IF(K-3)750,718,750
  718 IX(K,JFR)=0
  720 IXM=IX(K,JFR)-NEWX
       IF(IABS(IXM)-MXERR) 760,760,730
  730 DO 745 J=1,NJTS
      IF(IC(J))740,745,740
  740 IX(K,J) =IX(K,J) +IXM
  745 CONTINUE
      GOTO 760
  750 IX(K,JFR)= NEWX
  760 CONTINUE
  770 IC(JFR)=1
  780 CONTINUE
C   *** ****** *************** * ****************    **********************
C CHANGE COORDINATES OF THE FOOT, IF IT GOES THRU THE FLOOR.
      DO 2001 IBPRT=19,22,3
      J=JTCIS(IBPRT)
      IF(IX(3,J))2005,2001,2001
```

```
C NEGATIVE Z COORD
2005  IF(JPHKT(IBPRT)-90)2020,2010,2020
C PLACE LOW
2010  ANG=-RAD(JTTKT(IBPRT)+JPHKT(IBPRT))
      GOTO 2030
C NON-PLACE LOW
2020  ANG=RAD(JPHKT(IBPRT))
2030  CALL COSIN (ANG,CSANG,SNANG)
      AT=LNTHL(IBPRT)
        ATX=AT*SNANG
       ATY=AT*CSANG
        JBT=J-2
      WRITE(6,2000) IX(1,J),IX(2,J),IX(3,J)
2000  FORMAT(' THE COORDINATES OF THE FOOT ARE',3I5)
      IX(1,J)=IFIX(ATX+ FLOAT(IX(1,JBT)))
      IX(2,J)=IFIX(ATY+FLOAT(IX(2,JBT)))
      IX(3,J)=0
2001   CONTINUE
C*****************************************************************
C*****************************************************************
      DO  795 IEXTR=1,3
      IEXLM=IEXTL(IEXTR)
      IEXTC=JTPPX(IEXLM)
C CALCULATE THREE-DIMENSIONAL COORDS OF HANDS AND HEAD
C IEXTL(1) IS THE HEAD, (2) IS THE RIGHT HAND, (3) IS THE LEFT HAND
C ORIGINAL JOINT FOR EACH IS WRIST OR NECK
      ITA=JTTKT(IEXLM)
      IFH=JPHKT(IEXLM)
       IPS=JPSKT(IEXLM)
      DO 792 ISIDE=1,2
      ISUB=2*IEXTR+ISIDE-2
C ISUB=1 RIGHT SIDE OF HEAD
C ISUB=2 LEFT  SIDE OF HEAD
C ISUB=3  RIGHT FINGERS
C ISUB=4 RIGHT THUMB
C ISUB=5 LEFT FINGERS
C ISUB=6 LEFT THUMB
      CALL TRDSP(ITA,IPH,IPS,IEXCO,ISUB,IXD)
      IEXJT=23+ISUB
      DO 790 K=1,3
      IX(K,IEXJT)=IX(K,IEXTO) + IXD(K)
790   CONTINUE
792    CONTINUE
 795  CONTINUE
      KFR=FRAME
      JBEAT=FLOAT(JB)/BSPBT
      WRITE(4)KFR,JB,JBEAT,IX
       WRITE(6,310) KFR,JB,JBEAT
310   FORMAT(' FRAME',I3,'; BEATSTEP',I3,'; BEAT',I3)
      WRITE(6,311) ((IX(K,J),J=6,29),K=1,3)
311   FORMAT(24I5/)
      WRITE(6,501)FRAME,JPHKT,JPSKT,JTTKT ,(DASH,I=1,30)
501   FORMAT(F5.0,'  PHI',22I5/ 5X,'  PSI',22I5/5X,'THETA',22I5/30A4//)
C -----------------FRAME AND BEATSTEP LOOP CONTROL--------------------
  800 FRAME=FRAME+1.0
      IF(FRAME-FJBF-FRATO)290,290,830
C     (DO NEXT FRAME IF IT TERMINATES WITHIN CURRENT BEAT)
 830  JB=JB+1
```

```
C -----------------------------------MOVJI-------------------------------
C TRANSFER DATA STORED IN AREA J INTO AREA I
900    DO 920 J=1,NSIMP
       IPHIA(J)=JPHIA(J)
       IPSIA(J)=JPSIA(J)
       ITTA(J)=JTTA(J)
        IDURA(J)=JDURA(J)
920     CONTINUE
       DO 940 J=1,NJTS
        ISDUR(J)=JSDUR(J)
940    CONTINUE
       GOTO 100
       END
C         NONEG
       SUBROUTINE NONEG(I)
C NONEG ALWAYS RETURNS A POSITIVE VALUE FOR ANGLE, X, WHERE 0<X<360
       IF(I)10,15,20
10     I=I+360
15     RETURN
20     IF(I-360)15,15,25
25     I=I-360
        GOTO 15
        END
C                NEWDA
       SUBROUTINE NEWDA(IB,INIPH,INIPS,ITA,LASPH,LASPS,LTA,DURI,
     *  JPHKT,JPSKT,JTTKT,LNTHL)
       DIMENSION JPHKT(22),JPSKT(22),JTTKT(22),LNTHL(22)
       API=3.14159
       IF(DURI-1.0)14,14,10
10      IF(INIPH-LASPH)20,12,20
12     IF(INIPS-LASPS)20,13,20
```

```
C THERE IS NO INTERPOLATION NECESSARY FOR PHI AND PSI, BUT FOR THETA, THERE IS
13     KPH=LASPH
       KPS=LASPS
       GOTO 200
14     KTA=LTA
15     KPH=LASPH
       KPS=LASPS
       GOTO 400
20     IF(IB-10)30,30,25
25     IREM=MOD(IB,3)
       IF(IREM-2)140,30,140
C
C  CALCULATE PHI,PSI FOR UPPER LIMBS AND FOR TORSO
C ------------
30     CALL DISP(INIPH,INIPS, 1,PX,PY,PZ)
       CALL DISP(LASPH,LASPS,1,OX,    OY,OZ)
       LSTLM=IB
C      (CALC. DIRN.COSINES OF INITIAL AND FINAL LIMB POSITIONS)
       CBTA=PX*OX + PY*OY + PZ*OZ
       IF(CBTA+0.99)130,130,40
C      (ERROR IF COS BETA = -1, IE BETA = 180)
40     BTA=ARCOS(CBTA)
       P=BTA/DURI
C (P IS ANGLE OF ROTATION THIS FRAME)
       CALL CROSP( PX,PY,PZ,OX,OY,OZ,FLP,FMP,FNP)
       PR=LNTHL(IB)
       PO=PR
       CALL COSIN(P,CSP,SNP)
       PS=CSP*PO
       SO=SNP*PO
```

```
      CALL DCS(PR, 0.0,PQ, PS,SQ,PX,PY,PZ,1.0,FLP,FMP,FNP,
         DCROX,DCROY,DCROZ,  DCPQX,DCPQY,DCPQZ)
C     (USING FOLD ROUTINE, CALC. D.C.'S OF CURRENT POSITION OF LIMB IB)
      CALL PHPS(DCPQX,DCPQY,DCPQZ,KPH,KPS,&120)
      GOTO 200
120   KPH=LASPH
C     (IF CURRENT PSI IS +90 OR -90, SET CURRENT PHI = DESTINATION PHI)
      GOTO 200
130   WRITE(6,131)IB
131   FORMAT(' ROTATION OF LIMB',I4,' THROUGH 180 DEGREES: ILLEGAL')
      STOP
C
C CALCULATE PHI,PSI FOR LOWER LIMBS
C ------------
140   LMUP=IB-1
C     (PREVIOUS, IE ADJACENT UPPER, LIMB)
      CALL VECNML(JPHKT(LMUP),JPSKT(LMUP),JTTKT(LMUP),
         DCLUPX,DCLUPY,DCLUPZ,  DCNUPX,DCNUPY,DCNUPZ)
C     (GET D.C.'S OF UPPER LIMB AND OF NORMAL TO UPPER LIMB)
      FNML=SNML(LMUP)
      IF(LMUP-LSTLM)150,160,150
C THE D.C.'S OF THE PREVIOUS (UPPER) LIMB WEREN'T CALCULATED IN
C THE PREVIOUS CALL TO NEWDA, SO THEY ARE EQUAL TO THE CURRENT D.C.'S
C OF LMUP, I.E. TO DCLUPX,Y,Z.
150   PX=DCLUPX
      PY=DCLUPY
      PZ=DCLUPZ
      QX=PX
      QY=PY
      QZ=PZ
```

```
160   CALL DISP(INIPH,INIPS,1,PPX,PPY,PPZ)
C     (D.C.'S OF INITIAL POSITION OF LIMB IB)
      CALL DISP(LASPH,LASPS,1,COX,COY,COZ)
C     (D.C.'S OF FINAL POSITION OF LIMB IB)
      CBTAIN=-(PX*PPX + PY*PPY + PZ*PPZ)
      CBTALS=-(OX*OOX + OY*OOY + OZ*OOZ)
C     (COSINES OF INITIAL AND FINAL ANGLES BETWEEN UPPER LIMB LMUP
C     AND THIS LIMB IB -- MINUS SIGN TO GET INCLUDED ANGLE)
      BTAIN=ARCOS(CBTAIN)
      BTALS=ARCOS(CBTALS)
      BTAKT=BTAIN + (BTALS-BTAIN) / DURI
C     (INTERPOLATE LINEARLY FOR CURRENT ANGLE BETWEEN LIMBS)
      IF(BTAKT-API+0.01)170,180,180
170   PR=LNTHL(LMUP)
      QR=LNTHL(IB)
      CALL COSIN(BTAKT,CSBTAK,SNBTAK)
      PS=PR - QR*CSBTAK
      SQ=QR*SNBTAK
      CALL DCS(PR,QR,0.0,PS,SQ,  DCLUPX,DCLUPY,DCLUPZ,   FNML,
        DCNUPX,DCNUPY,DCNUPZ,   DCRQX,DCRQY,DCRQZ,  DCPQX,DCPQY,DCPQZ)
C     (USING FOLD ROUTINE, CALC D.C.'S OF FOLDED LIMB IB)
      CALL PHPS(DCRQX,DCRQY,DCRQZ,KPH,KPS,&175)
      GOTO 190
175   KPH=LASPH
C     (IF CURRENT PSI IS +90 OR -90, SET CURRENT PHI = DEST PHI)
C     (USING D.C.'S OF LIMB IB, CALC PHI AND PSI)
      GOTO 190
180   KPH=JPHKT(LMUP)
      KPS=JPSKT(LMUP)
```

```
C       (IF ANGLE BETN LIMBS IS 180, PHI AND PSI OF LOWER LIMB ARE SAME
C       AS THOSE OF UPPER LIMB)
190     IF(IFEM)200,195,200
195     PX=PPX
        PY=PPY
        PZ=PPZ
        DX=DDX
        DY=DDY
        DZ=DDZ
        LSTLM=I3
C       (SAVE INITIAL AND FINAL D.C.'S OF LOWER LIMB FOR USE WHEN CALCULATING
C       EXTREMITY)
C
C CALCULATE THETAS
C --------------
200     IF(IB-10)210,210,220
C NO ROTATIONS OF TORSO ALLOWED
210     KTA=0
        GOTO 400
C ROTATIONS OF LIMBS
C MODIFY INITIAL THETA IF INIT PSI=+90 OR -90 TO CORR TO DEST PHI
220     ITAM=ITA
        IF(INIPH-360)270,232,270
232     IF(INIPS+90)240,250,240
240     IF(INIPS-90)270,260,270
C PLACE HIGH INITIALLY
250     ITAM=ITA+LASPH
        GOTO 265
C PLACE LOW INITIALLY
260     ITAM=ITA-LASPH
C IF INIT PSI=+90 OR -90, NO CHANGE TO DEST THETA REQUIRED
```

```
 265    LTAM=LTA
        GOTO 340
 C IF INITIAL PSI NOT +90 OR -90,
 C MODIFY DESTINATION THETA IF DEST PSI IS +90 OR -90 TO CORR TO INIT PHI
 270    LTAM=LTA
        IF(LASPH-360)310,272,310
 272    IF(LASPS+90)280,290,280
 280    IF(LASPS-90)310,300,310
 C PLACE HIGH FINALLY
 290    LTAM=LTA+INIPH
        GOTO 340
 C PLACE LOW FINALLY
 300    LTAM=LTA-INIPH
        GOTO 340
 310    IF (IABS(INIPH-LASPH)-180)340,320,340
 320    CALL MODDT(INIPH,LASPH,KPH,ITAM,LTAM)
 C      (MODIFY THETAS IF MOVE GOES THROUGH PLACE LOW OR PLACE HIGH FROM
 C      FRONT TO BACK OR VICE VERSA)
 340    CALL NEWDT(IB,ITAM,LTAM,DURI,KPH,KTA)
 400    JPHKT(IB)=KPH
        JPSKT(IB)=KPS
        JTTKT(IB)=KTA
        RETURN
        END
 C                    NEWDT
        SUBROUTINE NEWDT(IB,ITAM,LTAM,DURI,KPH,KTA)
 C INTERPOLATES THETAS
        DIMENSION IMPOSS (2,4)
        DATA IMPOSS/30,30,330,330,180,0,180,0/
```

```
C IMPOSS ORDER: RAF RAB LAF LAB RLF RLB LLF LLB
      KFRONT=1
      IF((KPH.LT.270).AND.(KPH.GT.90)) KFRONT=2
C KFRONT ID 1 IF CURRENT LIMB POSITION IS IN FRONT OF BODY
C KFRONT ID 2 IF CURRENT LIMB POSITION IS IN  BACK OF BODY
      LIMNO=(IB-2)/3-2
C  1=R ARM, 2= L ARM, 3=R LEG, 4= L LEG
      ITA=ITAM
      LTA=LTAM
      CALL NONEG(ITA)
      CALL NONEG(LTA)
      IMPOS=IMPOSS(KFRONT,LIMNO)
       IF((ITA-IMPOS)*(IMPOS-LTA)) 40,50,10
C A FORMULA WHICH FINDS IF A NUMBER LIFS BETWEEN TWO NUMBERS IF THE DIFFERENCE
C IN THE FIRST AND NUMBER IN QUESTION AND THE DIFFERENCE BWTEEN THE SECOND AND
C NUMBER IN QUESTION , MULTIPLIED BY EACH OTHER YIELD A POSITIVE ANSWER
C IMPOSSIBLE SITUATION (+VE)
C IT IS IMPOSSIBLE TO GC DIRECTLY TO LAST POSITION
10    IF(ITA-LTA) 20,40,30
20    ITA=ITA+360
C (INCREASE THE SMALLER BY 360)
      GOTO 40
30    LTA=LTA+360
40    INCR=FLOAT(LTA-ITA)/DURI   +0.5
      KTA=ITA+INCR
      CALL NONEG(KTA)
       RETURN
 50    WRITE(6,51) ITA,LTA, LIMNO,KFRONT
51.   FORMAT(' INIT OR FINAL THETA',2I5,' FOR LIMB',I4,'IN FONT/BACK',
     *I3,' IS AN IMPOSSIBLE VALUE')
      STOP
      END
C
      SUBROUTINE COSIN(A,CS,SN)
       CS=CCS(A)
      SN=SIN (A)
       RETURN
         END
C
      FUNCTION SNML(IB)
C     SIGN OF NML: +1 EXCEPT FOR THIGHS
      IF(IB-17)5,10,2
2     IF(IB-20)5,10,5
5     SNML=1.00
      RETURN
10    SNML=-1.00
      RETURN
      END
C
      FUNCTION ARCTN(X,Y)
      ARCTN=ATAN2(X,Y)
      IF(ARCTN)30,40,40
30    ARCTN=ARCTN+6.28318
40    RETURN
       END
C
      FUNCTION ARCSN(Y)
      X=Y
      IF(ABS(X).LE.1.0) GOTO 20
      IF(X.GT.1.0)X=1.0
      IF(X.LT.-1.0)X=-1.0
20    ARCSN=ARSIN(X)
      RETURN
      END
```

```
C         DISP
      SUBROUTINE DISP(IPH,IPS,LNT,X1,X2,X3)
      APH=RAD(IPH)
      APS=RAD(IPS)
      CALL COSIN (APH,CSPH,SNPH)
      CALL COSIN (APS, CSPS, SNPS)
      X1=CSPS*SNPH*LNT
      X2=CSPS*CSPH   *LNT
      X3=-SNPS *LNT
      RETURN
      END
C
       FUNCTION RAD(I)
   10  RAD=FLOAT(I)*0.017453
       RETURN
      END
C
      FUNCTION IRND(X)
       RND=X+0.5
      IF(RND)10,20,20
 10    RND =RND-1.
  20  IRND=RND
      RETURN
      END
C          MODDT
      SUBROUTINE MODDT(IN,LA,KP,ITS,LTS)
C   CHECKS IF LIMB MOVES IN FRONT TO BACK OR V.V.
```

```
12     IF((IN.LT.270).AND.(IN.GT.90))GOTO 30
C INITIAL IN FRONT-CHECK FINAL PHI
14      IF((LA.GE.270).OR.(LA.LE.90)) GOTO 40
C (INITIAL AND FINAL PHI BOTH FRONT. RETURN SAME THETA)
18      IF((KP.LT.270).AND.(KP.GT.90)) GOTO 24
C FROM LINE 18- INIT F: CURRENT F: FINAL B
C  FROM LINE 32 - INIT B; CURRENT  B  FINAL  F
20     LTS=LTS+180
        IF(LTS-360)40,21,21
21     LTS=LTS-360
        GOTO 40
C INIT B CHECK FINAL
30     IF((LA.LT.270).AND.(LA.GT.90)) GOTO 40
C INITIAL B, FINAL F
32     IF((KP.LT.270).AND.(KP.GT.90))GOTO20
C FROM LINE 18 INIT F  CURRENT B  FINAL B
C FROM LINE 32 INIT B, CURRENT F, FINAL F
24     ITS=ITS+180
       IF(ITS-360)40,25,25
25     ITS=ITS-360
40     RETURN
       END
C               PHPS
       SUBROUTINE PHPS(DCX,DCY,DCZ,KPH,KPS,*)
C PHPS  CALCULATES THE PHI,PSI, GIVEN DIRECTION COSINES OF A VECTOR)
       SNPS=-DCZ
C      (-VE SIGN FOR PSI BECAUSE THAT CORRESPONDS TO +VE Z)
       CSPS=SQRT(1.0 - SNPS*SNPS)
       APSJ=ARCSN(SNPS)
C      (-PI/2 < APSJ < PI/2)
               KPS=IRND(APSJ/0.017453)
```

```
      IF(CSPS-0.001)20,20,10
10    SNPH=DCX/CSPS
      CSPH=DCY/CSPS
      APHJ=ARCTN(SNPH,CSPH)
C     (0 < APHJ < 2*PI)
            KPH=IRND(APHJ/0.017453)
       RETURN
20    RETURN1
C     (IF PSI IS +90 OR -90, TAKE SPECIAL RETURN WITHOUT CALCULATING PHI)
       END
C           TRDSP
      SUBROUTINE TRDSP(ITA,IPHI,IPSI,IEXCO,ISUB,IXD)
      DIMENSION IEXCO(3,6),IXD(3)
      DIMENSION DXX(3),DXXO(3),A(3,3)
      DO 10 K=1,3
      DXXO(K)=IEXCO(K,ISUB)
10    CONTINUE
      THETA=RAD(ITA)
      PHI=RAD(IPHI)
      PSI=RAD(IPSI)
      CALL COSIN(THETA,CTA,STA)
      CALL COSIN(PHI,CPH,SPH)
      CALL COSIN(PSI,CPS,SPS)
      SS=SPS*STA
      A(1,1)=CPH*CTA - SPH*SS
      A(2,1)=-SPH*CTA - CPH*SS
      A(3,1)=-CPS*STA
      A(1,2)=SPH*CPS
      A(2,2)=CPH*CPS
      A(3,2)=-SPS
      SC=SPS*CTA
      A(1,3)=CPH*STA + SPH*SC
      A(2,3)=-SPH*STA + CPH*SC
      A(3,3)=CPS*CTA
      DO 100 I=1,3
      DXX(I)=0.
      DO 90 J=1,3
      DXX(I)=DXX(I) + A(I,J)*DXXO(J)
90    CONTINUE
100   CONTINUE
       DO 110 K=1,3
       IXO(K)=IRND(DXX(K))
110    CONTINUE
      RETURN
      END
C                CROSP
      SUBROUTINE CROSP(X1,X2,X3,Y1,Y2,Y3,CROSS1,CROSS2,CROSS3)
C FINDS UNIT VECTOP NORMAL TO TWO GIVEN VECTORS
      CROS1=X2*Y3-X3*Y2
       CROS2=X3*Y1-X1*Y3
      CROS3=X1*Y2-X2*Y1
      SNTH=SQRT(CROS1*CROS1 + CROS2*CROS2 + CROS3*CROS3)
      CROSS1=CROS1/SNTH
      CROSS2=CROS2/SNTH
      CROSS3=CROS3/SNTH
      RETURN
      END
```

```
C         DCS
      SUBROUTINE DCS(PR,QR,PQ,PS,SQ,DCPRX,DCPRY,DCPRZ,FNMULT,DCNX,DCNY,
     *  DCNZ,     DCRQX,DCRQY,DCRQZ,DCPQX,DCPQY,DCPQZ)
C AS INPUT TO THIS ROUTINE, PR,PS,SQ REQUIRED, IF QR>0.0,REQUIRE DC'S OF VECTOR
C RQ, IF PQ>0.0, REQUIRE DC'S OF PQ ((DC IS DIRECTIONAL COSINE)).
      DCNNX=DCNX*FNMULT
      DCNNY=DCNY*FNMULT
      DCNNZ=DCNZ*FNMULT
      CALL CRUSP(DCNNX,DCNNY,DCNNZ,DCPRX,DCPRY,DCPRZ,DCSQX,DCSQY,DCSQZ)
      PQ1=PQ
      IF(PQ)10,10,20
10    PQ1=-1.0
20    DCPQY=(PS*DCPRX+SQ*DCSQX)/PQ1
      DCPQY=(PS*DCPRY+SQ*DCSQY)/PQ1
      DCPQZ=(PS*DCPRZ+SQ*DCSQZ)/PQ1
      IF(QR)40,40,30
30    DCRQX=(PQ1*DCPQX-PR*DCPRX)/QR
      DCRQY=(PQ1*DCPQY-PR*DCPRY)/QR
      DCRQZ=(PQ1*DCPQZ-PR*DCPRZ)/QR
40    RETURN
      END
C           VECNML
      SUBROUTINE VECNML(IPH,IPS,ITA,DCVX,DCVY,DCVZ,DCNX,DCNY,DCNZ)
C THIS ROUTINE(VECNML) CALCULATES THE NORMAL FROM PHI,PSI,THETA
C IT FINDS THE DIRECTIONAL COSINES OF A LIMB AND OF THE NORMAL TO THE LIMB GIVEN
C THE THREE ANGLES, PHI,PSI,THETA.
       CALL COSIN(RAD(IPS),CSPS,SNPS)
       CALL COSIN(RAD(ITA),CSTA,SNTA)
      CALL COSIN(RAD(IPH),CSPH,SNPH)
      DCVX  =CSPS*SNPH
      DCVY  =CSPS*CSPH
      DCVZ   =-SNPS
      SS=SNPS* SNTA
      DCNX    =CSPH*CSTA-SNPH*SS
       DCNY   =-SNPH*CSTA-CSPH*SS
      DCNZ    =-CSPS*SNTA
      RETURN
      END
/*
//A198ZELL JOB (               ),'LABANPL   '
//STEP3  EXEC   FORTGCL,LDISP='OLD,KEEP',LOAD='SC.A1148.LBN(LABANPL)'
//FORT.SYSIN  DD  *
C  ======================LABOPLOT==============================
      DIMENSION IX(3,35), ICAM(9),IPX(2,35,3),IDATA(9),
     *  ISTGR(3,3),ISTGL(3,3)
      READ(4) IBSBT,KYTIM,ICINE,IPLOT,ICAM
      INIFR=1
      MAXFR=999999
      INCFR=1
1     READ(5,2,END=40)KODE,IDATA
2     FORMAT(10I5)
      GOTO(10,20,30),KODE
C KODE=1: STAGE DIMENSIONS: RIGHT WIDTH, LEFT WIDTH, FORWARD DEPTH,
C BACKWARD DEPTH, SCALING FACTOR.   REQUIRED INPUT.
10    ISTGR(1,1)=IDATA(1)
      ISTGR(1,2)=IDATA(1)
      ISTGR(1,3)=0
      ISTGL(1,1)=0
      ISTGL(1,2)=-IDATA(2)
      ISTGL(1,3)=-IDATA(2)
      ISTGR(2,1)=-IDATA(4)
      ISTGR(2,2)=IDATA(3)
      ISTGR(2,3)=IDATA(3)
      ISTGL(2,1)=IDATA(3)
      ISTGL(2,2)=IDATA(3)
      ISTGL(2,3)=-IDATA(4)
      SCALER=IDATA(5)
      DO 15 L=1,3
      ISTGR(3,L)=0
      ISTGL(3,L)=0
15    CONTINUE
      GOTO 1
C KODE=2: INITIAL AND LAST FRAMES.   OPTIONAL INPUT.
20    INIFR=IDATA(1)
      MAXFR=IDATA(2)
      INCFR=IDATA(3)
      IF(INCFR.LE.0)INCFR=1
      GOTO 1
C KODE=3:  CAMERA DATA OVERRIDING THAT SUPPLIED IN STAGE 1.   OPTIONAL.
30    DO 35 L=1,3
      ICAM(L)=IDATA(L)
35    CONTINUE
      GOTO 1
40    WRITE(6,41)INIFR,MAXFR
41    FORMAT(' DATA REQUESTED FOR FRAMES',I6,' THRU', I6)
      FRMBT=FLOAT(ICINE*60)/FLOAT(KYTIM)
      NFRBT=FRMBT
      IF(INIFR.GE.NFRBT)GOTO 45
      WRITE(6,42)NFRBT
42    FORMAT(' FIRST',I3,' FRAMES ARE IDENTICAL')
      INIFR=NFRBT
45    INIFRM=INIFR-1
```

```
      IF(INIFRM.LE.0)GOTO 30
C     BYPASS UNWANTED FRAMES
      DO 50 I=1,INIFRM
      READ(4)
50    CONTINUE
80    IF(IPLOT.EQ.0)GOTO 100
C INITIALIZE PLOTTER
      IF(MAXFR.EQ.999999)MAXFR=FLOAT(IPLOT)*FRMBT
      XMAX=10*(MAXFR-INIFRM) + 2
C     (NO. OF INCHES OF PLOT THAT WILL BE PRODUCED = 10 X NO. OF FRAMES)
      CALL PRFRM1(IX,IBSBT,ICAM,XMAX,ISTGR,ISTGL,SCALER)
100   DO 400 I=INIFR,MAXFR
      READ(4) KFR,JB,JBEAT,IX
      IF(KFR.LT.0)GOTO 500
C(END OF JOB)
      WRITE(6,310) KFR,JB,JBEAT
310   FORMAT(' FRAME',I3,'; BEATSTEP',I3,'; BEAT',I3)
      IF(MOD((I-INIFR),INCFR))400,101,400
C     (PRINT AND PLOT ONLY EVERY INCFR'TH FRAME)
101   IF(IPLOT)118,300,118
118   FRAME=KFR
119   CALL PRFRM(JB,FRAME)
      GOTO 400
300   WRITE(6,311)                    (J,(IX(K,J),K=1,3),J=1,30)
311   FORMAT(I5,5X,3I6)
400   CONTINUE
C END OF DANCE
500   WRITE(6,510)
510   FORMAT(' END OF JOB')
      STOP
      END
```

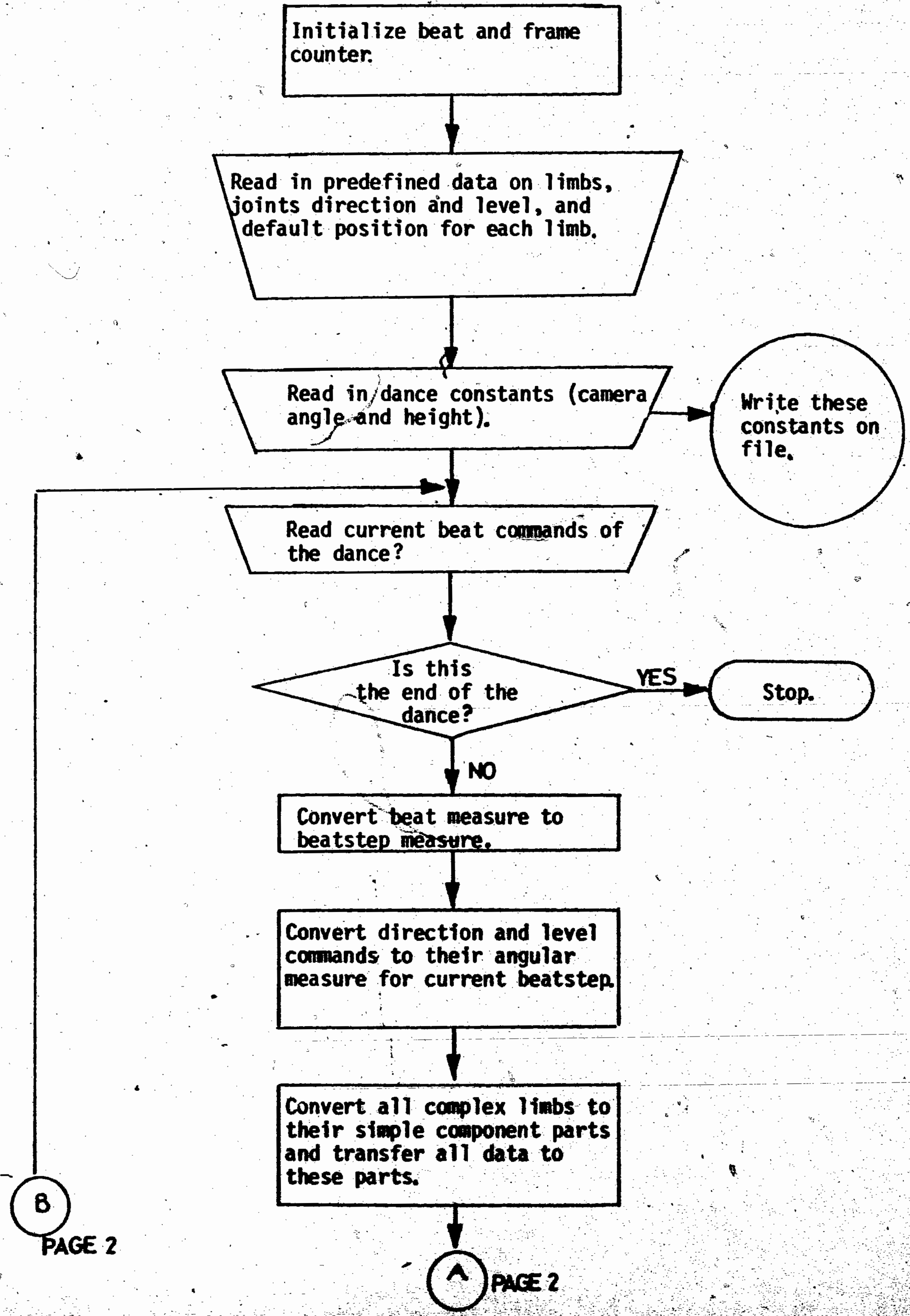
```
C          PRFRM1
      SUBROUTINE PRFRM1(IX,IBSBT,ICAM,XMAX,ISTGR,ISTGL,SCALFR)
        DIMENSION IX(3,35),ICAM(9),IPX(2,35,3), LIST(8,5),XARRAY(10)
      DIMENSION ZARRAY(10),ISTGR(3,3),ISTGL(3,3),XSTG(5),ZSTG(5)
       EQUIVALENCE(XSTG,XARRAY(6)),(ZSTG,ZARRAY(6))
      DATA LIST/12,10, 3,6,1,2,6,1,2,14,16,18,20,20,18,27,14,5,22,23,24,
     *22,25,23,5,15,17,19,21,28,19,29,15,2,7,1,7,9,11,13/
      XARRAY(9)=0.0
      ZARRAY(9)=0.0
      XARRAY(10)= SCALER
      ZARRAY(10)= SCALER
      CALL PLOTS(IBSBT,IBSBT)
      WRITE(6,1001)ICAM
1001   FORMAT(T11,'DISTANCE FROM',T32,'CAMERA LOCATIONS',T58,'ANGLE TO
     *RIGHT'                                     /T13,'PERFORMER',T30,'ELEVATION
     * ABOVE STAGE',T57,'OF STAGE FORWARD'
     *           /(T3,I10,T33,I10,T55,I10))
      NCAM=0
      DO 1003 I=1,3
      II=3*I - 2
      IF(ICAM(II).EQ.0)GOTO 1005
      NCAM=NCAM+1
1003    CONTINUE
1005  XMAX=XMAX*(FLOAT(NCAM))
      CALL XLIMIT(XMAX)
      RETURN
      ENTRY PRFRM(JB,FRAME)
      BEAT=JB/IBSBT
      PHIP=0.0
```

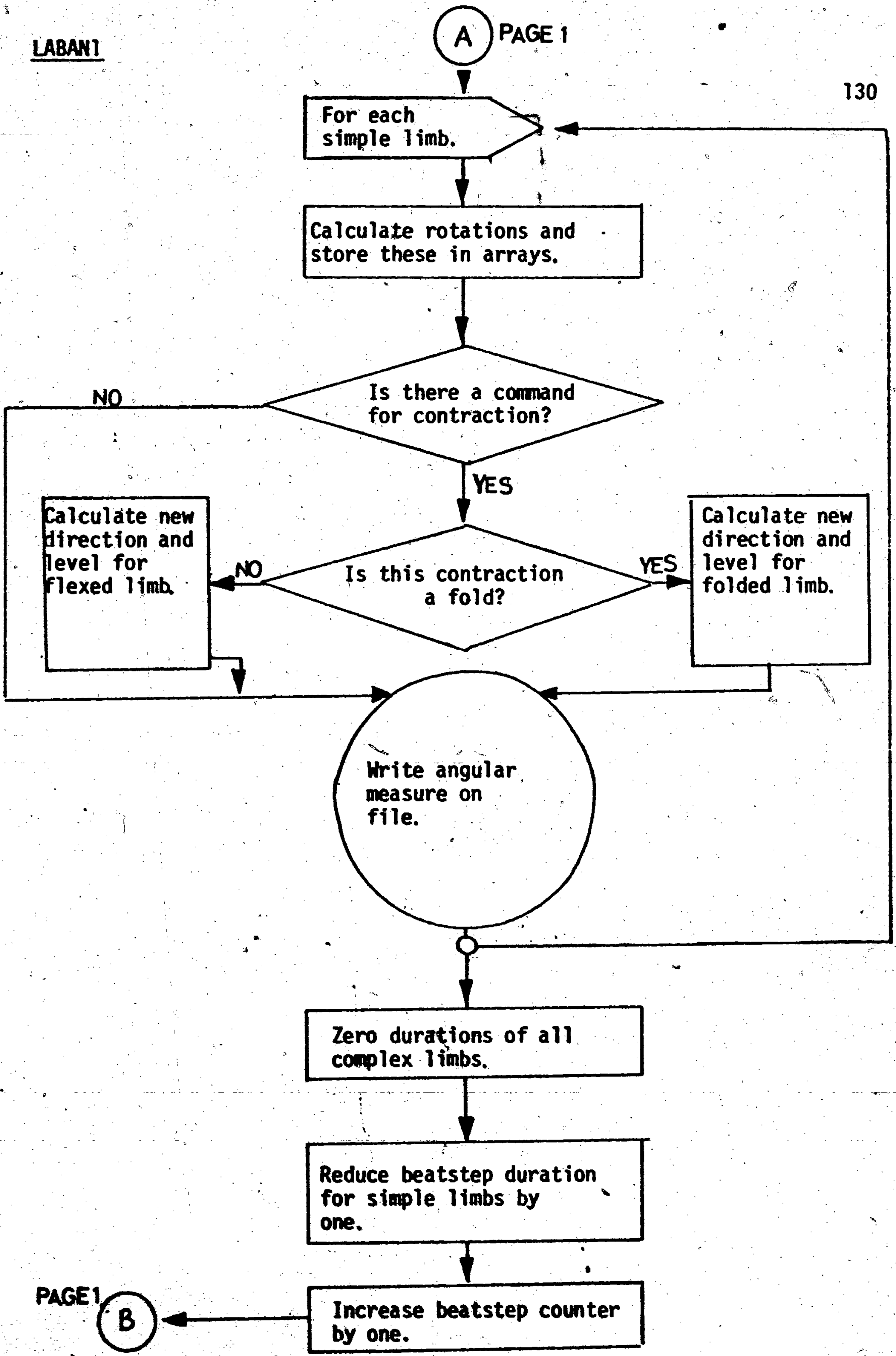
```
C       (ANGLE OF PERFORMER FORWARD FROM STAGE FORWARD)
        DO 1040 KAMERA=1,NCAM
        CAMERA=KAMERA
        KAMBAS=3*KAMERA
        RC=ICAM(KAMBAS-2)
C       (DISTANCE FROM PERFORMER TO CAMERA)
        HC=ICAM(KAMBAS-1)
C       (HEIGHT OF CAMERA ABOVE PERFORMER GROUND LEVEL)
        PHIC=RAD(ICAM(KAMBAS))
C       (ANGLE OF CAMERA FROM STAGE FORWARD)
        CALL NUMBER(0.0,1.0,0.14,FRAME,270.0,-1)
        CALL PLOT(0.0,0.0,-3)
C       (ORIGIN FOR DRAWING FIGURE)
        CALL PLOT(0.0,8.4,2)
C ( DRAWS A DIVIDING LINE BETWEEN FRAMES.)
        DO 1010 J=1,3
        CALL PROJ(RC,HC,PHIC,ISTGR,J,0.0 , XSTG(J),ZSTG(J),3,SCALER)
1010      CONTINUE
         CALL LINE(XSTG,ZSTG,3,1,0,2)
C       (DRAW STAGE)
          DO 1020 I=1,5
        DO 1030 J=1,8
        L=LIST(J,I)
        CALL PROJ(RC,HC,PHIC,IX,L,PHIP,X,Z,30,SCALER)
        XARRAY(J)=X
        ZARRAY(J)=Z
         IPX(1,L,KAMERA)=X
        IPX(2,L,KAMERA)=Z
1030      CONTINUE
        CALL LINE(XARRAY,ZARRAY,8,1,0,1)
1020      CONTINUE
```

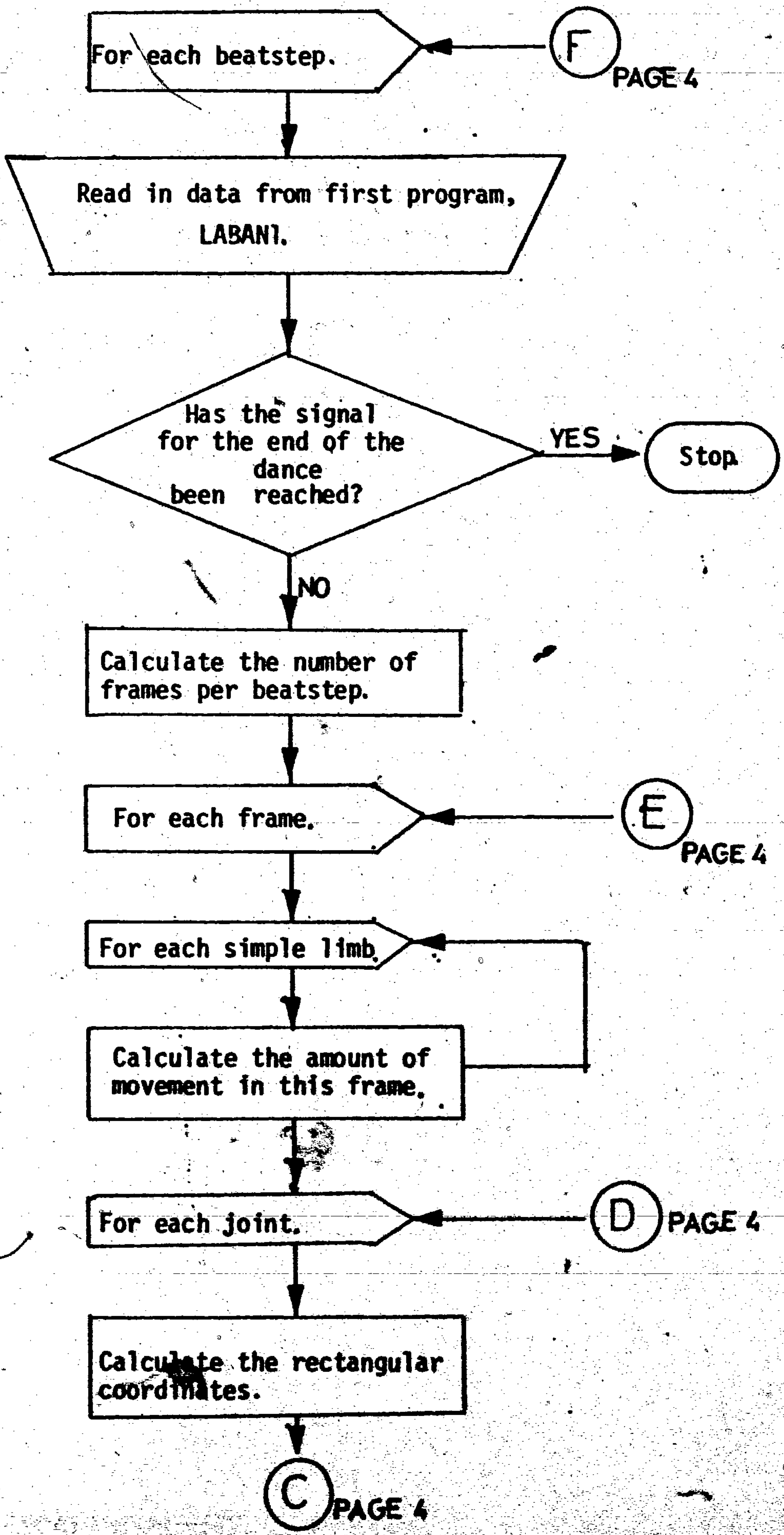
```
      DO 1035 J=1,3
      CALL PROJ(RC,HC,PHIC,ISTGL,J,0.0,XSTG(J),ZSTG(J),3,SCALER)
1035   CONTINUE
      CALL LINE(XSTG,ZSTG,3,1,0,2)
C     (DRAW STAGE)
      CALL PLOT(10.0, 0.0,999)
C     (PLOT STICK FIGURE, THEN SHIFT ORIGIN READY FOR NEXT FRAME OR
1040    CONTINUE
      DO 1050 J=1,29
C THE THIRD AND FOURTH POINTS COORDINATES ARE NOT CALCULATED FOR PLOTTING
      IF((J. EQ.3).OR.(J.EQ.4)) GOTO 1050
      WRITE(6,1031)J,(IX(K,J),K=1,3),((IPX(K,J,L),K=1,2),L=1,NCAM)
1031  FORMAT(I5,5X,3I6,5X,2I6,5X,2I6,5X,2I6)
1050  CONTINUE
      RETURN
      END
C                   PROJ
      SUBROUTINE PROJ(RC,ZC,PHIC,      IX,L,PHIP,X,Z,NSUB,SCALEF)
      DIMENSION IX(3,NSUB)
C THE SCALER IS REALLY THE NUMBER USED BY THE PLOTTER TO REDUCE THE SIZE
C IN THE PROJ ROUTINE, THE SCALER IS THE NUMBER NECESSARY TO MOVE THE
C    STAGE AND DRAWINGS OVER TO THE CORRECT PLACE ON THE PAPER.
      XP=IX(1,L)
      YP=IX(2,L)
      ZP=IX(3,L)
      PHID=PHIP-PHIC
      XTP=XP*COS(PHID) + YP*SIN(PHID)
      YTP=YP*COS(PHID) - XP*SIN(PHID)
C     (PERFORMR COORDS TRANSFORMED TO SCREEN COORDS)
C         (SCREEN IS PERPENDICULARTO LINE OF CAMERA TO PLACE)
          T = -RC / (YTP-RC)
          XO=-T*XTP
          ZO=T*(ZP-ZC)+ZC
          X= ZO+2.6*SCALER
C THE 2.6 IS THE AMOUNT TO THE RIGHT OF THE PLOT ORIGIN FROM WHICH TO DRAW
C THE PLOTS FOR ANIMATION
          Z = 4.95*SCALER - XO
C        4.95 IS THE NUMBER WHICH CONVERTS THE PLOTTER COORDINATES FROM THE
C MOTION ANALYZER COORIDNATES, BECAUSE THERE IS A INVERSION TAKING PLACE
C THE X OF FLOTTER IS Y OF MOTION ANAYZERS AND Y OF PLOTTER IS X OF ANALYZER.
C " CHANGE THE POINTS TO SUIT THE ANINATION NECESSITIES. NORMAL XO AND ZO WILL B
C    E THE X AND Z COORS PAST AND WILL BE SUFFICIENT FOR THE CORREST FIGURE)
          RETURN
          END
C
          FUNCTION RAD(I)
          IF(I-360) 10,20,10
   10     RAD=FLOAT(I)*0.017453
          RETURN
   20     RAD=0.0
          RETURN
          END
/*
```

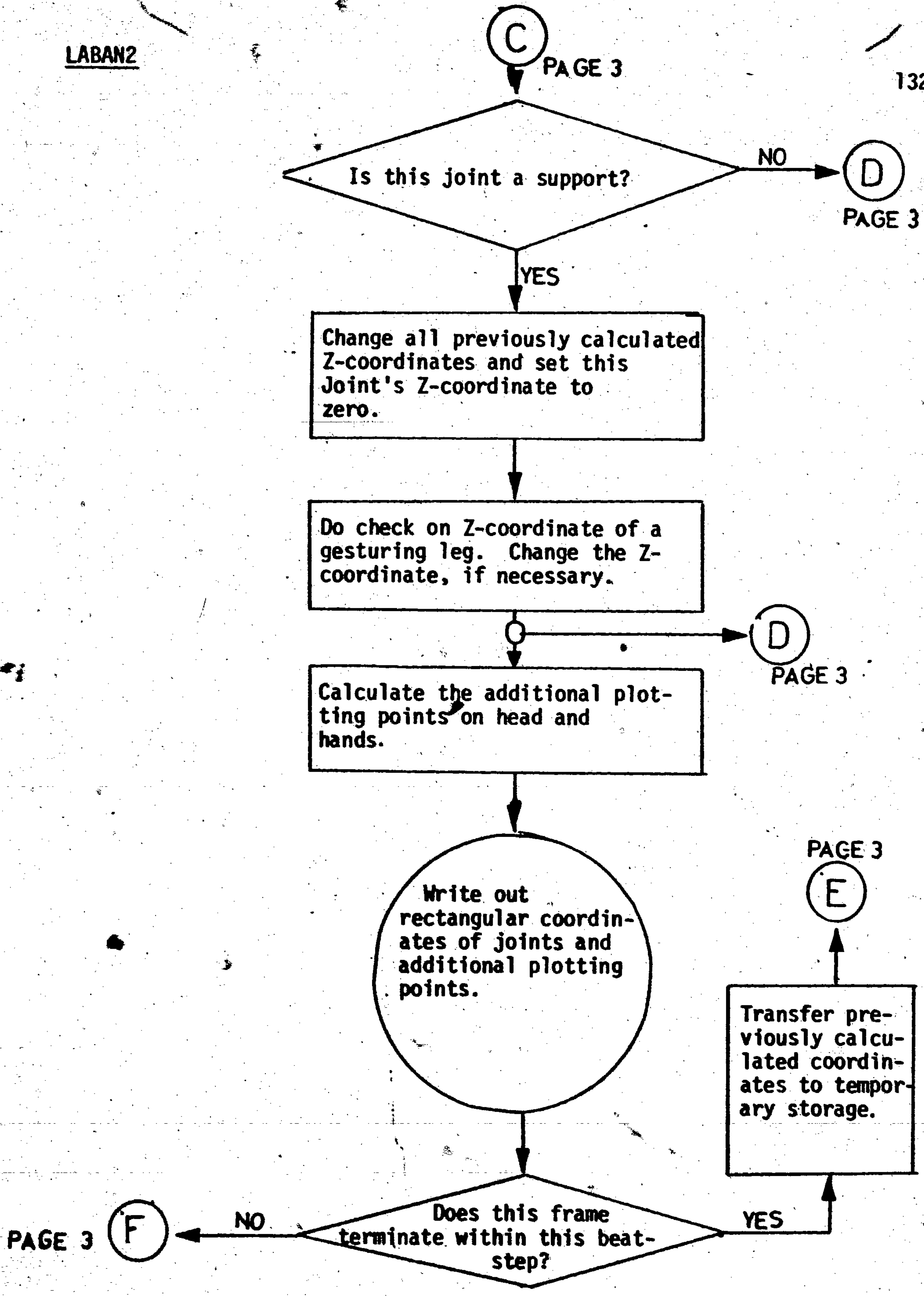
APPENDIX B

FLOWCHART OF LABAN

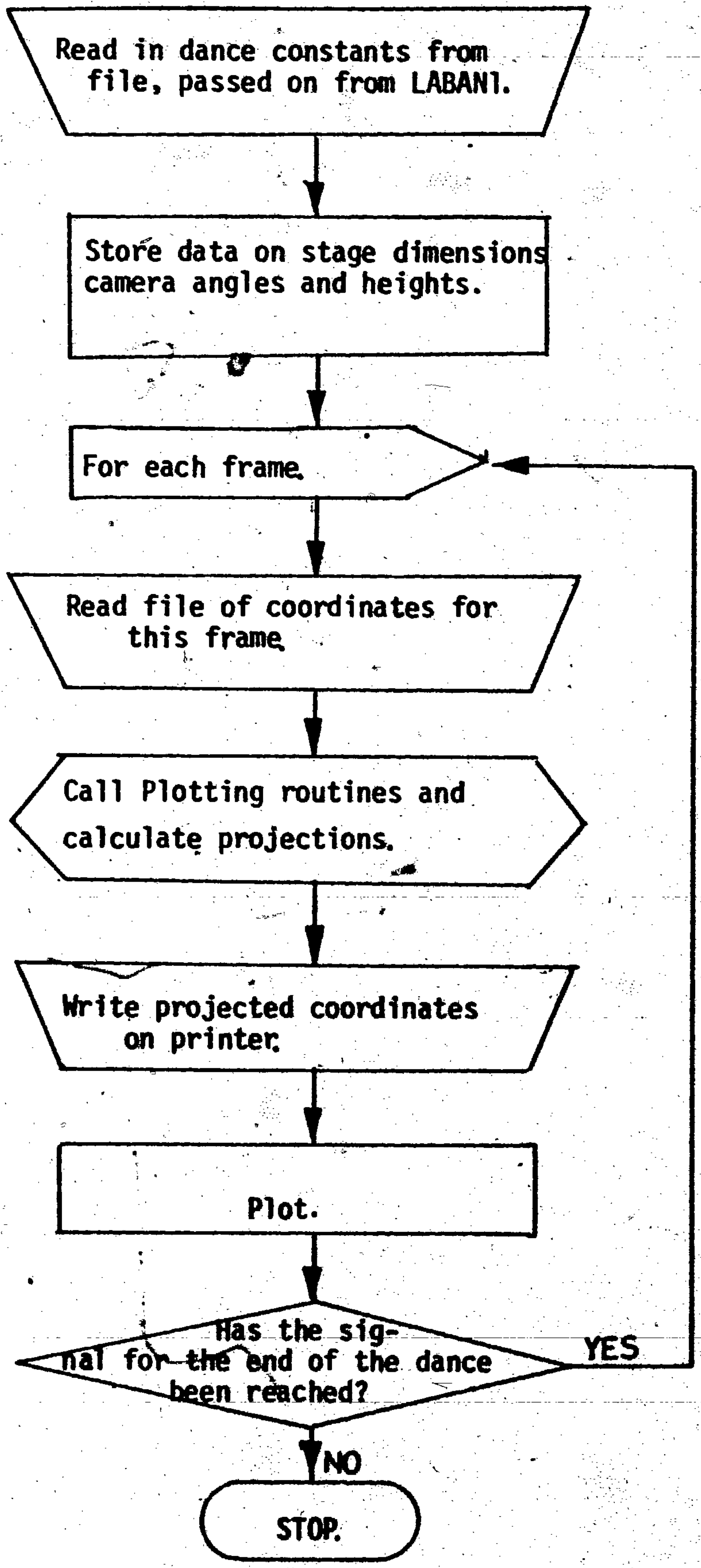

Initialize beat and frame counter.
Read in predefined data on limbs, joints direction and level, and default position for each limb.
Read in dance constants (camera angle and height).
Write these constants on file.
Read current beat commands of the dance?
Is this the end of the dance?
YES
Stop.
NO
Convert beat measure to beatstep measure.
Convert direction and level commands to their angular measure for current beatstep.
Convert all complex limbs to their simple component parts and transfer all data to these parts.
B
PAGE 2
A
PAGE 2


LABAN1
A PAGE 1
For each simple limb.
Calculate rotations and store these in arrays.
Is there a command for contraction?
NO
YES
Is this contraction a fold?
NO
YES
Calculate new direction and level for flexed limb.
Calculate new direction and level for folded limb.
Write angular measure on file.
Zero durations of all complex limbs.
Reduce beatstep duration for simple limbs by one.
Increase beatstep counter by one.
PAGE1 B


For each beatstep.
F
PAGE 4
Read in data from first program, LABAN1.
Has the signal for the end of the dance been reached?
YES
Stop
NO
Calculate the number of frames per beatstep.
For each frame.
E
PAGE 4
For each simple limb
Calculate the amount of movement in this frame.
For each joint.
D
PAGE 4
Calculate the rectangular coordinates.
C
PAGE 4


C
PAGE 3
Is this joint a support?
NO
D
PAGE 3
YES
Change all previously calculated Z-coordinates and set this Joint's Z-coordinate to zero.
Do check on Z-coordinate of a gesturing leg. Change the Z-coordinate, if necessary.
D
PAGE 3
Calculate the additional plotting points on head and hands.
Write out rectangular coordinates of joints and additional plotting points.
PAGE 3
E
Transfer previously calculated coordinates to temporary storage.
PAGE 3
F
NO
Does this frame terminate within this beat-step?
YES

LABANPL


Read in dance constants from file, passed on from LABAN1.
Store data on stage dimensions camera angles and heights.
For each frame.
Read file of coordinates for this frame.
Call Plotting routines and calculate projections.
Write projected coordinates on printer.
Plot.
Has the signal for the end of the dance been reached?
YES
NO
STOP.

APPENDIX C

EVALUATION QUESTIONNAIRE AND SIGN TEST SCORES

I. ANIMATION

The following questions will ask you to evaluate the animation of the film.

(1) Animation techniques sometimes produce jumpy or flickering images. Please rate the animation as to:

(a) how smooth, or unflickering, the film appears to you.
(b) how clearly you could see the details in the film.

(2) How well do you think that an experienced dancer could reproduce the movements just shown in the filmstrip?

II. TIMING AND RHYTHM

Much of learning dances involves understanding rhythm and timing. The following questions will try to assess how clearly timing and rhythm are shown in the film.

(3) How well do you feel that accents or impulses were shown?

(4) How close an approximation to human timing of similar movements is the film, in your opinion?

(5) Please evaluate how well an experienced dancer might understand the timing of the movements shown.

III. THE DRAWING

The stickfigure drawing shown was chosen because it was felt to be the simplest possible drawing to show motion. Please evaluate the following as to the satisfactoriness (or otherwise, if need be) from the point of view of a dancer using these types of films to recreate movements.

(6) Use the five point scale to evaluate each question below:

(a) the use of stickfigure drawings in general.
(b) this particular stickfigure.
(c) the shape of the figure (e.g. filled-in areas).
(d) the body proportions in general.
(e) the head size in relation to the rest of the body.
(f) the head shape.
(g) the torso size.
(h) the torso shape.
(i) the leg length.
(j) the arm length.
(k) the arm shape.
(l) the hands and thumb as to size.
(m) the hands and thumb as to shape.
(n) the arm connection to the torso.
(o) the leg connection to the torso.

(7) In looking at the arm movements:

(a) how well do the arm movements actually resemble human arm movements?
(b) how naturally does the torso respond to the arm movements?
(c) how clear are the directions of the arm movements?

(8) How clearly can you see what the legs are doing?

IV. <u>GENERAL APPRECIATIONS</u>

(9) Did you find the movements mechanical looking?

If YES, could you write some suggestions as to how you feel this could be avoided.

(10) Do you find that the film does not tell you enough about the movement for any specific thing, or for any reason. Please comment on your answer, if possible.

(11) Please comment on this type of film's useability as a cursory method of visualizing dance.

SIGN TEST SCORES

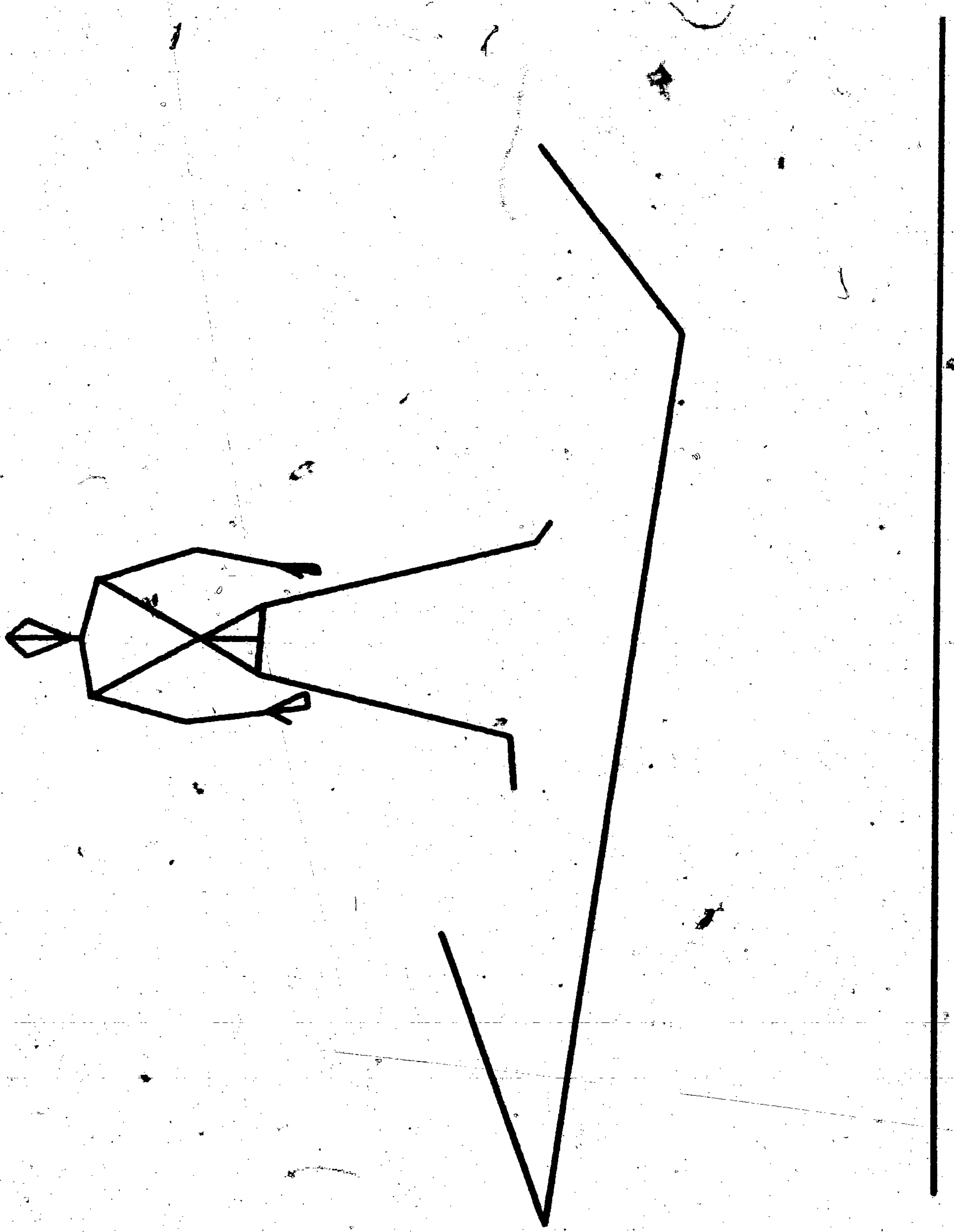
Assessor	Question																								
	1 a	1 b	2	3	4 a	5 b	6 a	6 b	6 c	6 d	6 e	6 f	6 g	6 h	6 i	6 j	6 k	6 l	6 m	6 n	6 o	7 a	7 b	7 c	8
1	+	+	0	0	0	0	0	0	-	0	0	0	0	0	0	0	+	0	0	0	0	0	0	+	0
2	+	0	+	+	+	0	0	0	0	+	0	0	0	+	0	+	0	0	0	0	0	0	0	0	+
3	+	0	0	+	+	+	0	0	0	-	0	0	0	0	+	+	+	-	-	0	0	+	0	0	+
4	+	+	0	+	0	0	-	0	+	+	+	0	0	-	0	-	0	0	+	+	+	+	+	+	+
5	0	+	+	0	-	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	+
6	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	+	0
7	+	0	+	+	+	+	0	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
8	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
9	-	0	0	0	-	-	0	-	-	0	+	-	0	-	0	-	-	-	-	0	0	-	-	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0
11	0	-	0	0	0	0	-	0	+	-	-	-	-	-	-	0	0	-	-	0	0	0	+	0	0
12	0	+	0	0	+	0	-	0	-	0	0	0	0	0	0	0	0	-	-	0	0	-	-	-	0
13	0	+	+	+	0	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
14	0	+	+	0	0	0	+	0	+	0	0	0	0	0	0	+	0	-	0	0	0	0	0	0	0
15	0	-	0	0	0	0	0	+	+	0	0	0	+	0	0	0	0	0	0	0	0	-	-	0	0
16	0	+	-	0	-	-	0	0	0	0	+	-	-	0	+	+	+	0	0	+	+	+	+	+	+
17	+	0	0	+	+	+	0	0	0	+	0	0	+	+	0	0	0	0	0	0	0	0	0	0	0
18	0	+	0	-	0	-	0	0	-	-	-	-	-	-	-	-	-	-	-	0	-	-	-	0	-
19	0	0	0	+	+	0	0	0	0	0	+	+	+	+	0	0	0	0	0	0	0	0	-	0	-
20	0	-	0	0	-	0	0	-	0	+	+	+	+	0	0	0	0	0	-	0	0	0	0	0	-
21	+	+	-	-	0	0	0	+	0	+	+	0	0	0	+	+	+	0	0	0	0	+	+	+	+
22	0	0	0	0	0	0	0	+	0	0	+	+	0	0	0	0	0	0	+	+	+	0	0	+	+
23	0	0	0	0	0	-	-	-	-	0	+	-	0	0	0	0	0	-	0	0	0	0	0	0	0
24	-	-	0	+	0	+	0	0	0	+	-	+	+	+	0	-	0	-	-	0	0	-	0	0	0
25	+	+	+	+	+	+	+	+	+	0	0	+	0	0	0	0	0	0	+	+	-	-	+	+	+
26	0	0	+	+	0	+	0	0	0	0	0	0	0	-	0	0	0	0	+	0	0	0	-	+	-
27	+	+	+	+	+	0	0	+	0	0	0	-	0	-	0	0	0	0	+	0	0	+	0	+	0
28	0	0	0	0	0	0	0	0	+	+	0	0	+	+	0	0	0	0	0	0	+	0	0	0	+
29	+	+	0	-	+	0	0	+	0	+	+	+	+	0	+	+	+	+	+	+	-	+	+	0	0
30	+	+	-	0	-	0	0	0	+	+	0	+	0	+	0	0	+	0	+	+	+	0	-	-	-

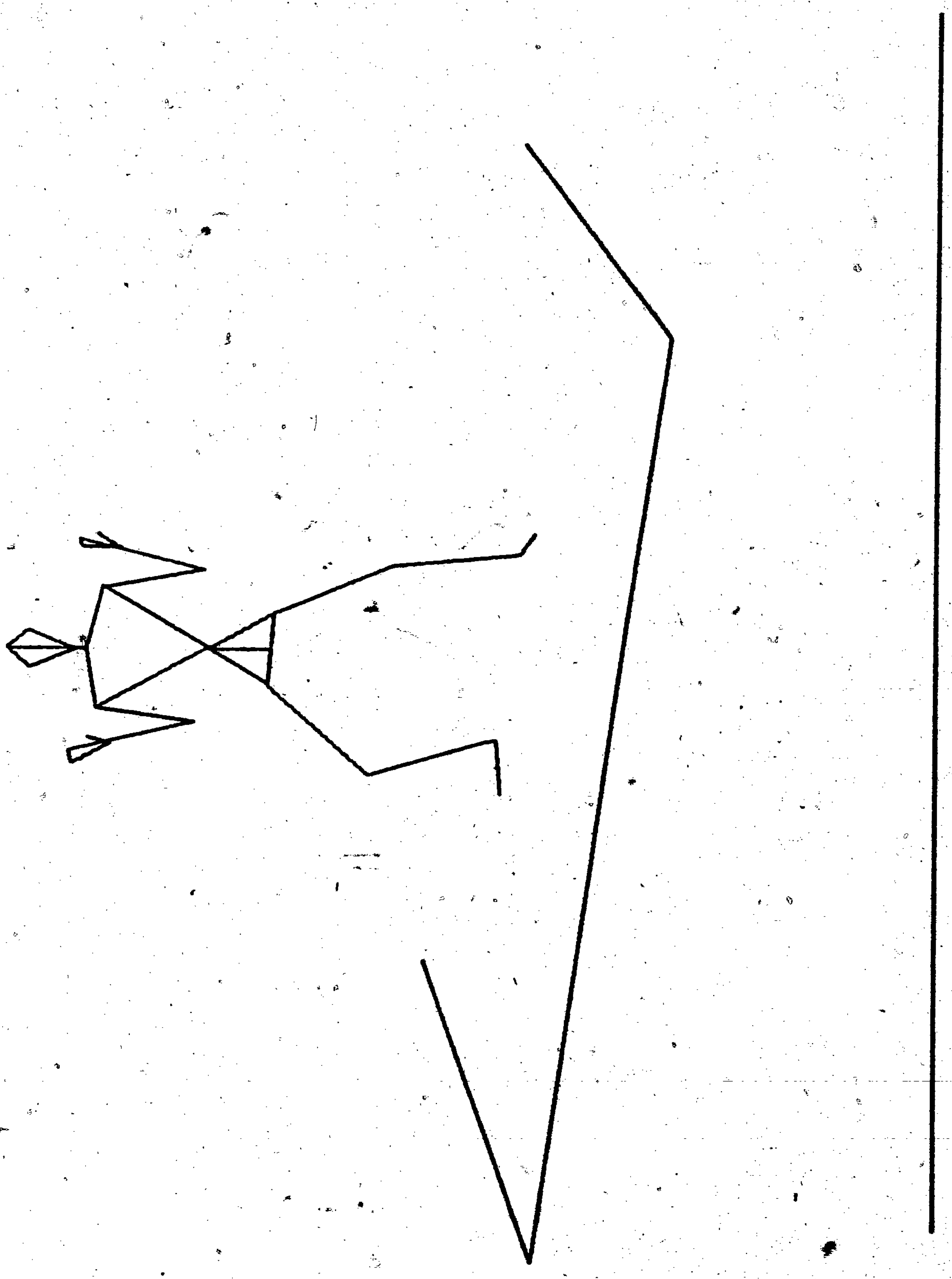
Sign Test Scores (continued)

Assessor	Question																							
	1	1	2	3	4	5	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	7	7	7	8
	a	b					a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	a	b	c	
31	+	0	-	+	0	0	0	0	0	0	0	0	+	+	0	0	0	-	0	0	0	-	0	-	+
32	0	0	0	0	+	0	+	0	0	0	0	0	+	+	+	+	0	0	0	0	0	0	+	0	0
33	0	0	0	0	-	0	0	0	0	0	+	+	+	0	-	-	+	0	0	+	+	0	0	-	0
34	+	0	0	+	+	0	+	-	-	0	0	0	0	0	0	0	0	+	+	-	-	+	+	0	0
35	0	+	0	0	+	-	0	0	+	0	+	0	0	0	0	0	0	0	-	+	+	-	0	0	0
36	0	-	0	0	0	0	0	0	0	0	0	+	+	0	0	0	0	0	0	0	0	0	0	+	0
37	0	+	+	+	0	0	+	+	0	0	0	0	0	0	0	0	0	0	0	0	0	+	+	+	0
38	+	0	0	+	+	+	0	+	0	+	0	+	+	0	0	0	+	+	+	0	-	+	0	+	+
39	+	+	0	0	0	0	0	+	0	-	0	0	0	0	0	0	0	0	0	0	0	+	0	0	+
40	+	0	+	+	0	0	+	-	0	-	-	-	0	0	+	+	+	+	+	+	+	+	+	+	+

APPENDIX D

SAMPLE DRAWINGS FROM THE PLOTTER





APPENDIX E

LABANOTATION PHRASE USED AS INPUT TO PROGRAM