NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

KINEMATIC ANALYSIS OF PLANAR LINKAGES BY COMPUTER AIDED GRAPHICAL METHODS

Irwin Puyun Ma

A Thesis
in
the Department
of
Mechanical Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering at
Concordia University
Montréal, Québec, Canada

March 1989

ISBN 0-315-49093-4

ABSTRACT

Kinematic Analysis of Planar Linkages by Computer Aided Graphical Methods

Irwin Puyun Ma

The objective of this thesis is to implement the typical graphical methods for the kinematic analysis of planar linkages by means of the modern computer aided graphics technology.

The concept of basic components of the modular approach is thoroughly studied. The programmability of a variety of popular graphical methods is verified. Individual unit of subroutines is prepared in the form of flow charts for the kinematic analysis of each of three popular basic components, namely the RRR II, RR-RR-RR III and RRR-RRR IV component. Simple analytic geometry is employed as the chief mathematical tool throughout the analysis. The simplicity and effectiveness of the graphical methods have been maintained whereas high efficiency and necessary accuracy have not been sacrificed. A VAX/VMS computer system loaded with the Norpak graphic package is employed to analyze a four-bar linkage based on one of the three subroutine units. The entire analysis procedure is displayed on a monitor screen for a quick understanding of the geometric and kinematic relationships of the linkage. The execution of the analysis is handily controlled with the help of the powerful interactive graphics. An important subroutine, named CROS, is proved to be more suitable to the nonlinear problems.

ACKNOWLEDGEMENT

The author wishes to express his sincere gratitude to Dr. V. I. Fabrikant and Dr. T. S. Sankar for their encouragement and guidance throughout this investigation. The financial support given by Natural Sciences and Engineering Research Council and Dr. T. S. Sankar is appreciated.

Thanks are also due, to the personnel of the computer center for their technical help. Special thanks to Miss Yan Yan Yueng and Mr. You Cai Chen for their immense help.

Finally, the author is indebted to his parents and other members of the family for their patience and understanding throughout this investigation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

$\vec{A}_A$	Linear acceleration of point A.
$\vec{A}_{BA}$	Linear acceleration of point B with respect to point A.
AB	Length of a link with node A and B at the two ends.
$\vec{V}_A$	Linear velocity of point A.
$\vec{V}_{BA}$	Velocity of point B with respect to A.
α	Angular acceleration of any link.
ϕ_{AB}	Angle between the positive x-axis and the direction of link AB measured in clockwise.
ω	Angular velocity of any link

Subscripts

A, B, C	Coordinates of position vectors
a, b, c	Coordinates of velocity vectors
a′,b′,c′	Coordinates of acceleration vectors
i, j, k	Variables corresponding to an increment of the input angle.

Superscripts

n	Normal component of a vector
t	Tangential component of a vector

CHAPTER 1

INTRODUCTION

Kinematic analysis of mechanisms is a vital part of any machine design process. It helps the designers to comprehend the motion of the existing as well as designed machines, to establish parameters for improvement and therefore to improve the machine efficiency. It is also a vital procedure for examining the mechanisms needed in the design of a new machine [1] [2]. Although this subject may be considered well developed, new technologies could always create openings for improvement. On one hand, the evolution of new knowledge leads to the new technologies. On the other hand, the creation of new facilities accelerates the emergence of new scientific concepts and design approaches. From a mechanism design point of view, such a phenomenon is characterized by the introduction of computer aided design (CAD) technology.

Prior to the 1950's, graphical methods were used almost exclusively for the kinematic analysis of mechanisms particularly for the displacement, velocity and acceleration analysis of planar mechanisms [3]. The emergence and hence the broad application of digital computers have greatly encouraged kinematicians to incorporate numerical methods to analyze planar as well as spatial mechanisms. Since then computer software packages making use of different numerical methods have been introduced successfully for efficient CAD approach using main-frame computers.

In recent years, technology of microcomputers has evolved dramatically and hence influenced the CAD technology. Consequently, machine design can be more and more easily accommodated with greater flexibility and lower cost particularly for industrial applications. Many small and special purpose oriented software packages have been created based on classical analytical methods and other conceptually new methods. The newest trend is the application of computer aided graphics (CAG). As a versatile tool, CAG is used to display results of analysis and to simulate all the mechanism movements as well. On the basis of the existing trend, it can be foreseen that new design approaches using CAG will dominate the mechanism analysis in design.

1.1 Survey of Previous Work

Traditionally, classical graphical methods offer a convenient way to solve linkage kinematic problems. They are simple, fast and easy to understand. With respect to their methodology, the positions, velocities and accelerations are determined in a way that the kinematic parameters associated with the driver are defined first, and quantities associated with other links are then defined in a sequential manner [4]. Their scheme involves a repetitive procedure of point by point analysis depending on the number of driver increments. Their accuracy depends on precision of line work, measurement and a choice of scales.

The analytical methods, as an alternative, also play a prominent role in kinematic analysis. Essentially they are based on the closed-form solutions of the general expressions for position, velocity and acceleration in terms of link geometry. Therefore once such general expressions are derived, accurate results can be obtained directly for all linkage positions. But not always can these expressions be found and very often elaborate algebraic manipulations are required to bring the expressions to explicit forms.

Ever since the increasingly wide application of digital computers coupled with the sophisticated numerical methods, design solutions towards the engineering problems have changed drastically. Uicker et al [5] developed an iterative method to analyze all planar mechanisms. Their scheme involves the solution derivation of a general loop-closure equation as well as its first and second order differential equations. Their methodology is based on the implementation of Newton-Ralphson iteration method. The application of Wengert's [6] computation procedure is later created so that their method can be programmed into a general package.

From a geometrical point of view, the schematics of a planar mechanism can be considered as a polygon. Any polygon can be divided into a number of triangles. Therefore the analysis of different kinds of linkages can be performed if a subroutine for each of the solutions of triangles has been created. Chace [7] expressed the sides of a triangle into complex polar vectors to establish its mathematical model. Solutions of related vector equations and the first and second order differential equations are hence programmed into different subroutines for general application.

The development of computer technology not only revolutionizes the computation methods in kinematics analysis, but also induces the radical changes in fundamental concepts. Traditionally, a linkage is viewed as a combination of links and attention is focused on links. Constraint methods, on the other hand, consider that a linkage is formed by a point system. These points are under certain constraints, thus move along certain paths. To implement these methods on computers, different kinds of constraints are programmed into individual subroutines to form a corresponding library. Because the kinds of constraints are limited, the number of subroutines is quite small. Jalon and Serna [8] developed a GAS method in which concepts of geometric constraints and kinematic constraints are defined. This method is applicable to planar as well as spatial mechanisms. Rooney [9] concentrated on planar mechanisms. He defined two kinds of constraints, the linear constraint and angular constraint. His scheme to solve the set of constraint equations is similar to that of the GAS method and has two parts. For simple mechanisms, the unknowns are sequentially solved. For more complicated ones, constraint equations are solved simultaneously. Concepts of other kinds of constraints such as the loop constraint and pair constraint are also defined in another research investigation [8]. Consequently many software packages, for example IMP, DYMAC, DAMN, MEDUSA and ADAMS, are created.

Suh and Radcliff [3] introduced the concept of basic component and hence proposed the modular approach. They showed that many planar mechanisms are assembled from one or more of the three basic combinations of rigid members: the two link dyad, the oscillating slider and the rotating guide. The analysis of these components are implemented in computer subroutines. Kinzel and Chang [10] extended the method by including three more basic components and incorporating the inversion technique. A relatively more complete description about the modular approach can be found in Liang and Yuen's text [2]. As many as sixteen basic components are solved in their work. A so called structural analysis is performed mainly for depicting a mechanism into a block diagram. Then the analytical methods coupled with numerical methods are extensively used throughout the kinematic and dynamic analysis. Subroutines are written to build the so called module for the complete analysis of a component. By creating a series of modules, a complete computer software package is developed for systematic analysis of planar mechanisms.

The dramatic development of computer graphic technology greatly encourages researchers to incorporate the available computing power and graphic capability with the kinematic analysis of linkages. ElMaraghy [11] used a PDP 11 minicomputer and other graphic facilities such as a graphic terminal and a light pen to analyze and synthesize a four-bar linkage. Barker [12] analyzed a four-bar linkage on an IBM PC with a color graphic monitor. Equipped with A Commodore-VIC 20 microcomputer system and a color graphic output, Smith [13] analyzed a slider-crank linkage, an inverted slider crank linkage and a four-bar linkage. Norton [14] concentrated on the analysis of a four-bar linkage and geared five-bar linkage and made use of an Apple IIe system. Since all these researches deal with specific and geometrically simple problems, the method employed is the closed-form solution. The sole difference is the mathematical approach to solve for positions, velocities and accelerations. The kinematic equations are solved simultaneously by either pure algebraic manipulation or Newton-Ralphson iteration method. Some other conceptually new methods have also been attempted recently. Funabashi [15] developed the incidence matrix method for the completely computer assisted analysis. Freudenstein and Beigi [16] established a general displacement equation based on the algebraic correspondence among the displacement equations of some basic linkages.

In many above mentioned researches, CAG techniques are employed for result displaying. Fabrikant and Sankar [17], on the other hand, made use of the CAG techniques to implement the classical graphical methods. A subroutine named CROS is created to locate the intersection(s) of curves explicitly displayed on a monitor screen with different pixel values for each curve. Such an essential graphical function is hence used to carry out the position analysis of a four-bar linkage[1]. This analysis is purely graphical unlike the other contemporary ones which are based on analytical and/or numerical methods.

1.2 Objectives of Research Work

The survey of literature indicates that analytical and numerical methods currently dominate the field of kinematic analysis of mechanisms. In most cases, the following pattern of procedure can be observed. At first, analytical methods are employed to derive the necessary mathematical expressions depicting the position, velocity and acceleration of mechanisms. Then a computer such as mainframes, minicomputers or microcomputers is used to solve these expressions numerically. Eventually a package is established to analyze either

1 In fact, like what the authors of the paper claimed, this is the first time that a graphical method is implemented on a computer for linkage kinematic analysis. For the sake of convenience a name of computer aided graphical method will be used to distinguish this kind of implementation.

general or specific mechanisms. The disadvantages of such an approach are that the mathematical analysis involved is typically complex, lengthy and often error prone. Moreover the mathematical methods are too theoretical to provide the quick insight a designer needs to understand and develop a practical mechanism.

Compared with analytical and numerical methods, graphical methods are simpler, faster, more flexible, more design oriented and easier to visualize. Kinematicians like Suh [3], Kinzel [10] and Liang [2] do recognize these advantages. They made use of the idea of the graphical methods which states that in most cases, it is possible to start the analysis at one end of a linkage and to analyze the linkage component by component until the desired output is determined. As a matter of fact, such an idea is the foundation of the modular approach. Nevertheless they apply the analytical and numerical methods to formulate and solve the problems of analysis instead. They all believe that it is very difficult if not impossible to automate the graphical methods on computers. Moreover graphical methods are thought to be suitable only for simple mechanisms and lack the accuracy required [2, 3, 10]. However this is not true.

Recently, the dramatic decrease of price/performance ratio of CAG facilities has made the powerful, versatile yet affordable graphic capability available to engineers and researchers. It is more appropriate than ever before to attempt the automation of the graphical methods. In fact, Fabrikant's [17] work could be considered as the first initiative for such an implementation. Therefore it is appropriate to conduct a relatively complete investigation to the practicality and hence the procedure to program the graphical methods into a package for linkage kinematic analysis. Thus the objectives of this thesis are

i) to verify the programmability of four typical graphical methods;

ii) to implement these graphical methods with CAG techniques to analyze three typical basic components and the corresponding linkages;

iii) to verify some of the key algorithms, which have been created during the implementation, by analyzing a planar four-bar linkage on a VAX/VMS minicomputer loaded with the Norpak graphic package.

1.3 A Brief Outline of the Thesis

In chapter 2, the distinct modular approach as well as a number of classical graphical methods such as the Chace's method, the instantaneous center method, the relative velocity and acceleration method and the velocity difference method are briefly reviewed. Then the programmability of these graphical methods and others being suitable for special kinds of problems are thoroughly investigated. Chapter 3 presents the procedure to implement the graphical methods with the CAG techniques. First of all, some secondary subroutines are created. Then subroutines to handle the position, velocity and acceleration analysis of three typical basic components, namely the RRR II, the RR-RR-RR III and the RRR-RRR IV components, are written in the form of detailed flow charts. These subroutines have been grouped into three units, one for each basic component. Finally linkages, each consisting of one of the three components, are analyzed to demonstrate the application of those subroutines. Chapter 4 discusses the implementation of the computer aided graphical methods on a VAX/VMS computer system loaded with a Norpak graphic package. After some utility subroutines are created, a four-bar linkage is analyzed by two approaches, one based on planar geometry and another based on analytic geometry.

CHAPTER 2

PRACTICALITY OF THE COMPUTER AIDED GRAPHICAL METHODS

2.1 Overview of the Modular Approach

Suh [3] included three component modules in the modular approach to the analysis of planar mechanism. The list is extended to eight by Kinzel [10]. But the most systematic as well as complete classification is proposed by Liang [2]. It is observed that one or more enclosed geometrical shapes form a basic component. The classification of the basic components is based on the number of pairs on the enclosed geometrical shape with the most sides. Refer to Fig. 2.1 (a). The simplest basic component, the dyad, consists of two links and three lower pairs. Because it has only two links, the enclosed geometrical shape is a straight line and there are two pairs at the ends of the line. Therefore it is classified as class II component. By the same token, the other two components in Fig. 2.1 are classified as class III and IV respectively. Different arrangements and types of pairs may appear for each class of components. Each class will consist of different types. For example, the component in Fig. 2.1 (b) is named as RR-RR-RR III component where the R's depict the revolute pairs and the way they are connected.

To deal with compound linkages with Kinzel's method [10], it is necessary to determine mentally a step by step approach to analyze the linkage before attempting to solve the entire assembly of the linkage. Liang [2] summarized such a mental procedure into a so called structural

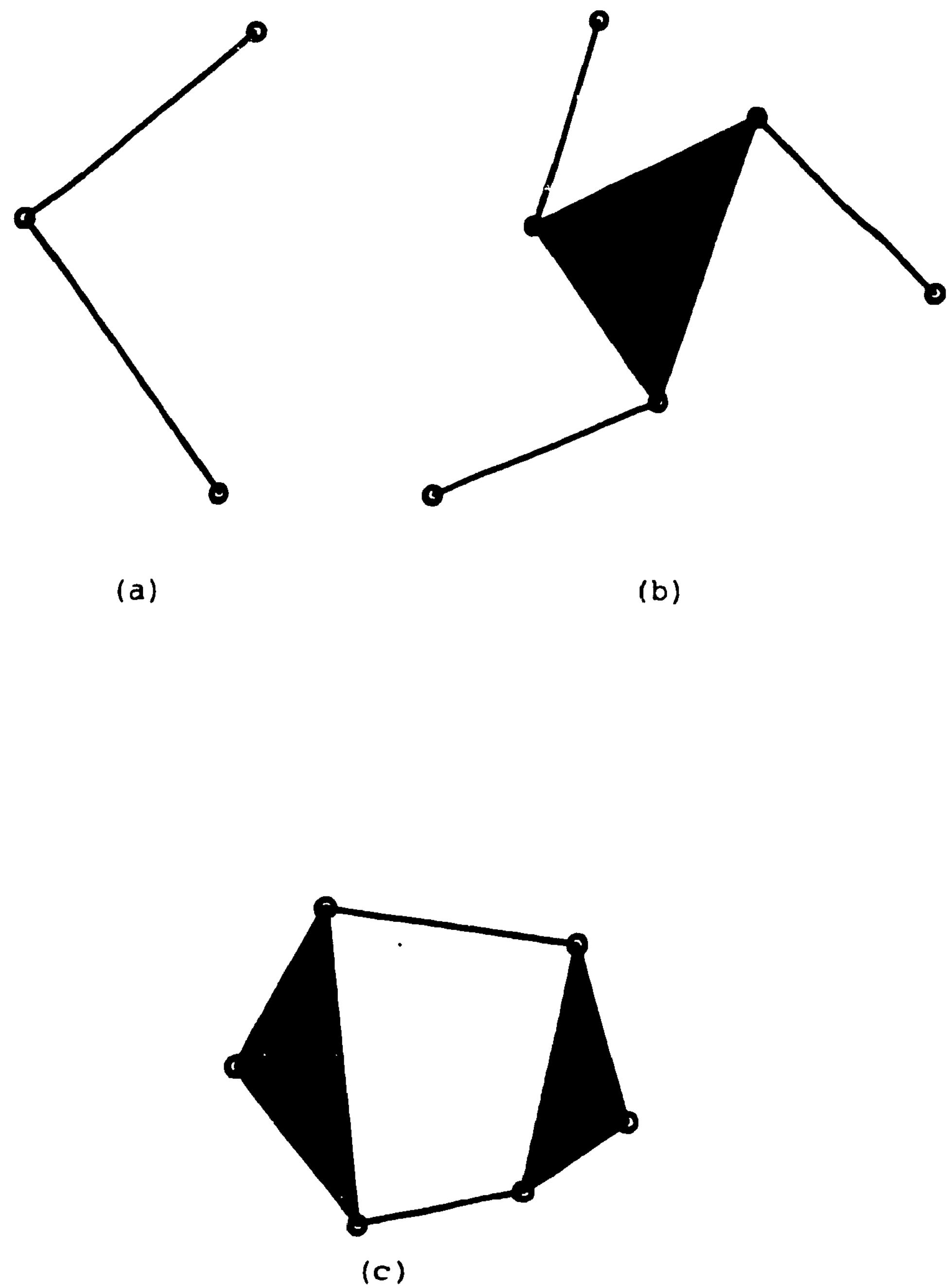


Fig. 2.1 Three typical basic components.

analysis of linkages. It has been proved that a linkage having one degree of freedom is composed of a frame, a driver and one or more basic components. For example, connecting an outer node of a dyad to a frame and another to a driver will form a four-bar linkage. Six steps have to be taken for the structural analysis.

i) Draw a schematics of the linkage.

ii) Designate a driver.

iii) Determine the degree of freedom.

iv) Remove the redundant constraints.

v) Decompose the linkage.

vi) Express the relationship amongst the driver, the frame and each of the basic components in a block diagram.

After the problem is clearly formulated, the kinematic analysis can be started. It includes the position analysis, the velocity analysis and acceleration analysis. The static and dynamic analysis, if it is necessary, could also be performed. In this research, however, concentration has been focused on the kinematic analysis rather than the structural, static and dynamic analysis.

Liang [2] based his position analysis on solving the vector polygons. The loop-closure vector equations are derived after projecting the position vectors to the Cartesian coordinate axes. Then first and second order differentiations with respect to time are taken to these equations to obtain the expressions for corresponding velocities and accelerations. For position analysis, equations of class II components are solved algebraically. For components of class III and IV or higher, because more equations are involved, more sophisticated mathematical manipulation is required. First of all, the highly nonlinear position equations are linearized by expanding them into Taylor's series. Then they are arranged into matrix form. The initial positions of the points on the component are then estimated by say graphical methods. Finally Newton-Ralphson iteration method is applied to solve the matrices. Obviously the mathematical manipulation involved is quite complicated. In this research, effort will be made to apply the graphical methods incorporated with CAG techniques to replace these analytical and numerical analysis.

2.2 Review of Classical Graphical Methods

2.2.1 Position analysis

The study of the position of linkages generally serves as a critical starting point to a complete kinematic analysis. The position analysis is usually the most difficult problem in kinematic analysis. It is a nonlinear algebraic problem when approached by analytical or numerical methods. For this reason, the graphical methods still remain attractive and are continuously enhanced as new techniques are created [1].

The problem of position analysis is to determine the values of all position variables, for example, the positions of joints and angular position of the links, given the dimension of each link and the values of the independent variables which are chosen to represent the degree(s) of freedom of the linkage such as the driver positions. To obtain the path(s) of the output motion of the desired points, the driver is allowed to move throughout its valid region. As well known, the graphical constructions involved in the analysis is based on planar geometry.

One of the significant approaches in the graphical methods is made use of by Chace [7]. In general, any single-loop planar linkages can be depicted as an n-vector polygon. If this polygon is solvable, its mathematical expression can always be simplified into

$$\vec{BD} = \vec{BC} + \vec{CD} \tag{2.1}$$

where $\vec{BD}$, $\vec{BC}$ and $\vec{CD}$ represent three vectors forming a triangle. And it is called the loop-closure equation. It can at most have two unknowns. Due to different combinations of the unknowns, four possible cases can arise, according to Chace.

case 1 Magnitude and direction of the same vector

case 2a Magnitudes of two different vectors

case 2b Magnitude of one vector and direction of another

case 2c Directions of two different vectors

Solutions of these four cases have been derived graphically [18]. It has been proved that planar linkages, consisting of only class II basic component(s) or lower ones, can be reduced to a form such that their position analyses can be conducted directly by applying these solutions.

This method[1] is most suitable for single loop mechanism analysis. When multi-loop mechanisms are involved, either rigorous manipulation of vector algebra may be necessary to obtain a vector equation with the same form as equation (2.1), or the related vector equations have to be

[1] Although the idea of this method has long existed, it is Chace who systematically implements it with the digital computer technology to perform the kinematic analysis to mechanisms. For the sake of convenience, in the rest of the thesis this method will be referred as the Chace's method.

solved simultaneously. In this research, some other specific methods are applied to reduce the multi-loop basic components into simple single loop ones so that the graphical methods that are suitable for single loop mechanisms can be used. Similar techniques will also be employed in the velocity and acceleration analysis of the same components. More details can be found in the later sections.

In the following chapter, a special graphical method is employed to analyze an RR-RR-RR III basic component and the method of assumed point [9] is applied to an RRR-RRR IV component.

2.2.2 Velocity analysis

Among the graphical methods for velocity analysis, two popular ones deserve more elaboration. They are the instantaneous center method and relative velocity and acceleration method.

The instantaneous center method is considered as one of the most effective methods for analyzing velocities of links in a linkage. In simple terms, an instantaneous center is defined as the point about which a body rotates relative to another body at a given instant. Applying this concept to a moving link of a linkage makes it convenient to describe its motion, at any given instant, in terms of pure rotation about an instantaneous center. With such an ability, one can greatly simplify the analysis by making it more convenient to determine the velocity of any point on a linkage.

Another method, the relative velocity method is probably the most common among the graphical methods. Compared to the others, it readily provides solutions for not only absolute velocities, but also relative velocities of points on a linkage. This singular feature makes it most desirable to use when determining relative velocities needed for acceleration analysis. The key procedure of this method is to construct velocity polygons based on the vector equation

$$\vec{V}_C = \vec{V}_B + \vec{V}_{CB} \tag{2.2}$$

which indicates that the absolute velocities of any two points in a linkage can be related by their relative velocities. With these concepts in mind, the velocity image method can easily be understood. If the velocities of any two points on a link are known, the velocity of the third point can be found by constructing a velocity image. Details about the construction of the velocity and acceleration image is given in the next chapter.

The so called auxiliary point method [19] has been developed to deal with relatively more complex linkages. The strategy of this method is to reduce the analysis of a complex linkage into that of simple single loop one, for example a four-bar linkage, by finding a special point based on the geometrical relation of the linkage and then starting the analysis from such a point. This method will be used for both the velocity and acceleration analysis of the RR-RR-RR III component and details will be found in the next chapter.

2.2.3 Acceleration analysis

The concept of relative acceleration method is similar to that of the relative velocity method, except that the acceleration of any point on a link will have two components, the normal and tangential components. The procedure to construct the acceleration polygons is therefore more complicated.

The velocity-difference method is probably the most straightforward of all the graphical methods used in kinematic analysis [1]. It does not rely on any sophisticated formulas, but instead employs the simple relationship

$$\vec{A} = \frac{\Delta\vec{V}}{\Delta t} \tag{2.3}$$

where

$\vec{A}$: the linear acceleration of a point on a linkage at any given position of the linkage;

$\Delta\vec{V}$: the change in velocity corresponding to a small change in position of the point caused by the rotation of the driver, namely $\Delta\theta$;

Δt: the time interval during which the change has taken place.

2.3 Programmability of the Classical Graphical Methods

It is traditionally believed that graphical methods are advantageous when analyzing a linkage at a single position, but become very laborious for multiple positions since each position must be started over as a completely new problem. Conversely, analytical and numerical methods do not suffer from this disadvantage. Once the mathematical expression of the solution is derived, it can be evaluated as often as desired at different positions of the linkage with very little effort. Therefore if graphical methods can by any way be applicable for the modern design, such a disadvantage must be overcome.

As implied by its name, modular approach includes a series modules. Each module analyzes a basic component and it consists of a few subroutines. Each subroutine tackles the position, velocity and acceleration problems individually. It is therefore possible to select the appropriate graphical method to solve a specific problem in each subroutine. Once the modules are established, they can be utilized whenever desirable to analyze the corresponding basic component. And the entire linkage could be analyzed component by component until the required output is found. So the problem here is to show that those selected graphical methods are programmable. And this will be the subject of the rest of the chapter.

The problem of programmability of the graphical methods could be discussed in two aspects, the graphical construction and mathematical analysis.

There are three types of instantaneous centers, the fixed type, the permanent type and the imaginary type. The first two types are those that can be readily located by inspection. It is the third type that needs to be located by graphical construction. An imaginary type center is a point, within or outside the mechanism, which is considered as the axis of rotation of a link at a specific instant. Kennedy's theorem is normally employed. It greatly simplifies the graphical construction for those planar mechanisms with lower pairs. A classical example involving a four-bar linkage pinpoints the typical procedure. Refer to Fig. 2.2. Points 12, 23, 34 and 14 are either fixed or permanent type centers. To determine the imaginary center 13, a

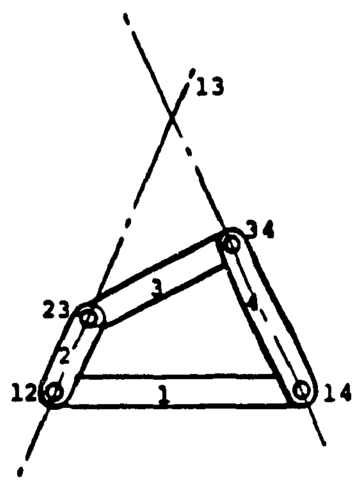


Fig. 2.2 Graphical construction for an imaginary type instantaneous center.

a straight line passing point, 12 and 23 are drawn to intersect another line passing point 34 and 14. Their intersection indicates the location of point 13. Obviously, it is a simple procedure in analytic geometry and therefore easy to be programmed into a subroutine. For more complicated linkages, the fundamental procedure would remain the same but the circle diagram method may be necessary. Essentially it is a method created to organize the searching process in a systematic manner. Details can be found in [1]. More effort is necessary to automate it.

Once the necessary instantaneous centers are obtained, the evaluation of the related velocities will involve only simple algebraic calculation. Refer to Fig. 2.3. Another instantaneous center point 24 is found in a similar way. Then velocity of point 34 can be found by either

$$V_{34} = V_{23} \frac{13\text{-}34}{13\text{-}23} \tag{2.3}$$

or

$$V_{34} = V_{23} \frac{12\text{-}24}{12\text{-}23} \frac{14\text{-}34}{14\text{-}24} \tag{2.4}$$

where 12-24 means the distance between point 12 and 24 and so as the others. The direction of V_{34} is apparently perpendicular to 13-14. By saving the calculation with regard to the coupler, equation (2.4) provides a direct way to evaluate the output velocity of the follower. As discussed in section 2.2.2, the general expression governing the relative velocity method is given as equation (2.2) having exactly the same form of equation (2.1) which is established to solve the position analysis problem. Similar phenomenon can be observed from the general

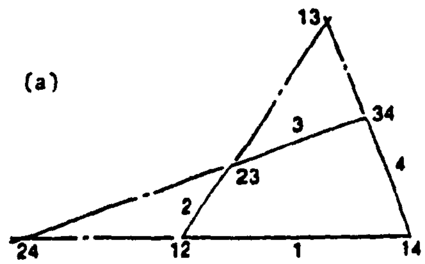


Fig. 2.3 Instantaneous centers of a four-bar linkage.

expression of the relative acceleration method. The accelerations of any two points on any rigid link under planar motion can be related by

$$\vec{A}_C^{\,t} + \vec{A}_C^{\,n} = \vec{A}_B^{\,n} + \vec{A}_B^{\,t} + \vec{A}_{CB}^{\,n} + \vec{A}_{CB}^{\,t} \tag{2.5}$$

It can be converted into

$$\vec{A}_C = \vec{A}_B + \vec{A}_{CB} \tag{2.5.1}$$

where

$$\vec{A}_C = \vec{A}_C^{\,t} + \vec{A}_C^{\,n} \tag{2.5.2}$$

$$\vec{A}_B = \vec{A}_B^{\,n} + \vec{A}_B^{\,t} \tag{2.5.3}$$

$$\vec{A}_{CB} = \vec{A}_{CB}^{\,n} + \vec{A}_{CB}^{\,t} \tag{2.5.4}$$

Recall that the key procedure to apply relative acceleration method is to construct the acceleration polygon, and any polygon can be viewed as being assembled from a number of triangles. If the acceleration polygon can by any way be defined, all the elemental triangles must be defined first. Therefore the problem of applying the relative acceleration method is reduced into a problem of solving a series triangles. This is exactly the same idea as the one implemented by Chace [7] to deal with the position analysis. Equation (2.5) depicts the accelerations of one of the links in a linkage and thus represents one of the elemental triangles in the acceleration polygon of the linkage. Such an elemental triangle is formed by other sub-elemental triangles expressed by equation (2.5.1) through (2.5.4). It has been shown by Chace [7] that solutions of the four cases for equation (2.1) cover all the possibilities in solving any triangle. If an acceleration polygon is solvable, the general procedure made use of by Chace should also be applicable for the relative acceleration method. Because the principle of the relative velocity method is the same as that of the relative acceleration method, there is no need to consider it separately.

In the case of an expanded link, the velocity image method is recommended to improve the efficiency in applying the relative velocity method. Fig. 2.4 (a) illustrates a link in plane motion. Velocities of point A and B are known. As indicated in Fig. 2.4 (b), velocity of point C can be found based on

$$\vec{V}_{AC} = \vec{V}_{AB} + \vec{V}_{BC} \tag{2.6}$$

which represents the shaded triangle. It is well known that the two shaded triangles are similar. By the same token, the acceleration image can also be constructed based on

$$\vec{A}_{AC} = \vec{A}_{AB} + \vec{A}_{BC} \tag{2.7}$$

Both equation (2.6) and (2.7) have the same form as equation (2.1). Therefore their solutions are also governed by the four cases for the equation (2.1). With respect to the graphical construction, a few typical procedures can be observed. While drawing a velocity polygon, the most frequent construction is to draw a line, originated from a specific point, to be perpendicular to a given line. While drawing an acceleration polygon, it is to draw lines parallel to given ones.

To calculate the acceleration of a point on a link at a particular position of the linkage, velocity difference method can be applied. Let t_0, t_1 and t_2 designate the instances bounding two equal time intervals, namely $\Delta t/2$. The acceleration of the desired point at t_1 is given by

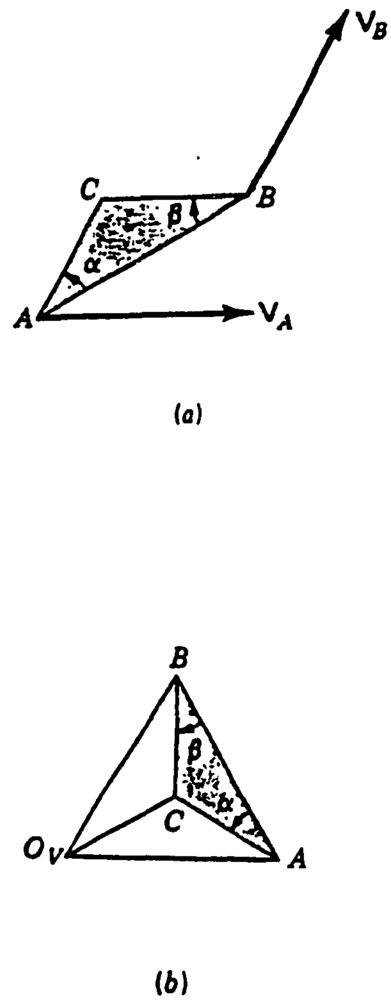


Fig. 2.4 Graphical construction of the velocity image method.

$$\vec{A}_1 = \frac{\vec{V}_2 - \vec{V}_0}{\Delta t} \tag{2.8}$$

This expression does not match equation (2.1) exactly due to a factor of Δt. But graphically it can be expressed as a triangle and the construction procedure is basically the same as the one in case 1 of the four cases for equation (2.1).

In summary, the graphical construction for all graphical methods discussed in this section is simple enough to be programmed by means of analytic geometry. Except the instantaneous center method in which unknown and known vectors are related by a numerical value, other graphical methods which have been discussed can be implemented by using the Chace's method.

2.4 General Discussion

The Modular approach is a systematic method created to analyze simple as well as complex mechanisms. It makes use of the concept of basic component and the idea of the classical graphical methods. But techniques of the graphical methods are abandoned. Instead, the analytical and numerical methods are applied to the kinematic analysis. The simplicity and effectiveness being the characteristics of the graphical methods are lost. In this chapter, a number of graphical methods are discussed. Their programmability is investigated. It is found that the Chace's method which deals with vector polygon is applicable to implement many typical graphical methods. And the graphical construction involved is quite straightforward.

CHAPTER 3

APPLICATION OF THE COMPUTER AIDED GRAPHICAL METHODS

In this chapter, three typical basic components are analyzed to demonstrate the implementation of the graphical methods discussed earlier by means of computer aided graphics. As illustrated in Fig. 2.1, the three selected components, the RRR II type (dyad), the RR-RR-RR III type and the RRR-RRR IV type, are amongst the most popular in each of their class.

The first part of the chapter is devoted to introduce some secondary subroutines needed later to analyze the components. They can be divided into two categories. The first category includes three elemental subroutines. Each of them is written to carry out one of the graphical method. In the second category, four subroutines are created to form an integrated package to assist the kinematic analysis of the basic components. They are also applicable when other linkages are analyzed. In the second part of the chapter, position, velocity and acceleration analysis of each of the three components is conducted individually in separate subroutines. In order to elucidate the procedure of applying these subroutines, one example for each component is attached at the end after the analysis of each component.

3.1 Preparation for the Analysis

Refer to equation (2.1) in section 2.2.1. Depending on the combination of the unknowns, four different cases may be encountered. In this thesis, only case 2c is needed for the related research. Although the graphical construction for each case is different, the general principle remains the same. Moreover, in view of analytic geometry, all the construction is fairly simple. Interested readers can refer to [18] where all four cases are solved graphically. In the mean time, subroutine TRICOS is created to solve case 2c. It determines the directions of two vectors by calculating their angles based on the screen coordinate system as shown in Fig. 3.1. As input parameters, X_B, Y_B, X_D and Y_D define the tip and tale of the given vector BD whereas BC and CD are the magnitudes of the other two vectors. It is important to specify the desirable configuration of the polygon. M is assigned to define the order of the denoting characters of ΔBCD. M equals 1 if B-C-D is clockwise. In case of ΔBC'D, M is -1. The output parameters include X_C, Y_C, ϕ_{BC}, and ϕ_{CD} which define the position of point C and angles of $\overrightarrow{BC}$ and $\overrightarrow{CD}$. A list of input and output parameters of TRICOS is given in Table 3.1. Similarly, parameters of other subroutines explained later in this section can be found either in Table 3.1 or Table 3.2 depending on which category they belong to.

In order to implement the velocity and acceleration image method, subroutine IMAGE is written. Refer to Fig. 3.1 again. Assume that ΔBCD is completely known. By knowing a vector $\overrightarrow{bc}$, it is necessary to construct Δbcd similar to ΔBCD as shown in Fig. 3.2. What IMAGE does is to locate the point c and find ϕ_{bc} and ϕ_{cd}, the angles of $\overrightarrow{bc}$ and $\overrightarrow{cd}$. As input parameters, X_B, Y_B, X_C, Y_C, X_D and Y_D define the vertices of ΔBCD. M takes a value of either -1 or +1 depending on the sequence of the denoting letters of ΔBCD. X_b, Y_b, X_c, Y_c indicate the position of the

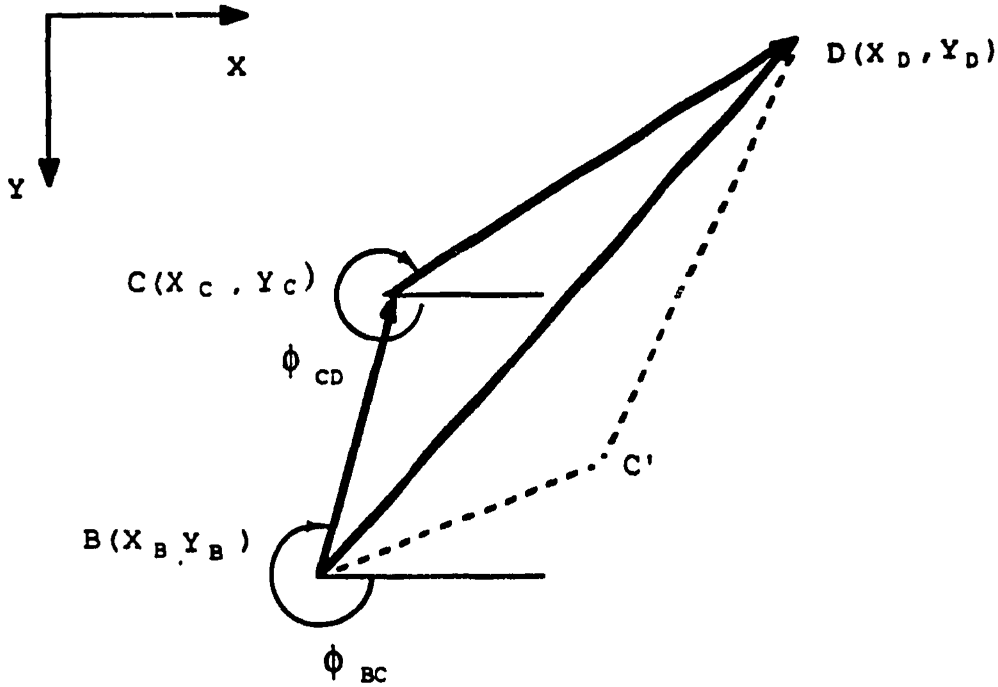


Fig.3.1 Solution of the loop-closure equation case 2c.

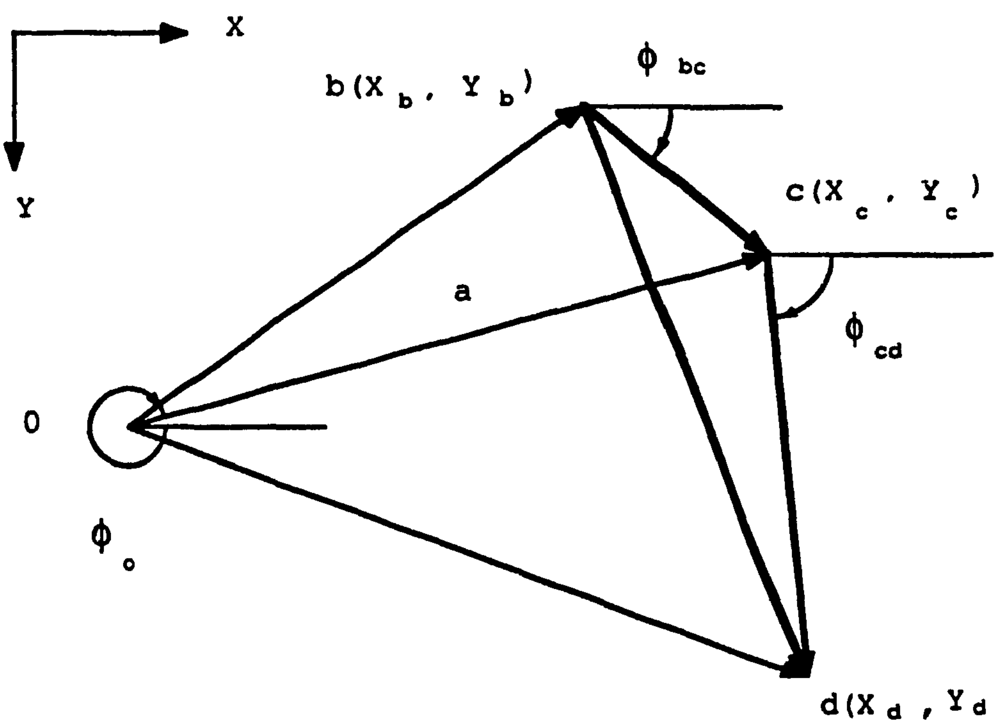


Fig. 3.2 Graphical construction of the velocity and acceleration image.

Table 3.1 Elemental subroutines of graphical methods

No	Name	Function	Input Parameter	Output Parameter
1	TRICOS	Solves case 2 of the loop - closure vector equation.	X_B, Y_B, X_D, Y_D, M BC, CD	$X_C, Y_C, \phi_{BC}, \phi_{CD}$
2	IMAGE	Implements the velocity and acceleration image method.	X_B, Y_B, X_C, Y_C, M X_D, Y_D, X_b, Y_b, X_c Y_c, X_o, Y_o	$X_d, Y_d, \phi_{bc}, \phi_{cd}$, a, ϕ_a
3	VELDF	Implements the velocity difference method.	$\Delta\phi, \omega, \alpha, X_a, Y_a$	X_a', Y_a'

Table 3.2 Utility subroutines for analysis

No	Name	Function	Input Parameter	Output Parameter
1	CRANK	Conducts position velocity and acceleration analysis with regard to a driver	$X_A, Y_A, \phi, \omega, \alpha,$ AB	X_B, Y_B, X_b, Y_b $X_{b'}, Y_{b'}$
2	INTE	Finds the intersection of two straight lines.	X_1, Y_1, S_1, X_2, Y_2	X_3, Y_3
3	AC	Determines the direction of an angular velocity and / or acceleration.	X_1, Y_1, X_2, Y_2, X_3 Y_3, X_4, Y_4, X_B, Y_B X_G, Y_G	AC
4	EX1	As part of DLIV, calculates the difference between the trial length and real length of a link.	BC, BD, CD, CE, EF EG, FG, DF, M_1, M_2 M_3, ϕ, N	D'F', ϕ_{CE}, ϕ_{EG}

given vector $\overline{bc}$. Finally the position of point O is given as X_0 and Y_0, where point O is the origin of a velocity or acceleration polygon such as the one shown in Fig. 3.2. As a result, position of point c is found once X_c, Y_c are defined. The direction and magnitude of vector $\overline{Oc}$ representing either the linear velocity or acceleration of point C is obtained as ϕ_a and a.

The last subroutine in this category is named VELDF which implements the velocity difference method. It evaluates the linear acceleration of a point on a link in any given time interval Δt providing that the linkage is not locked. When VELDF is called for the first time at any given time t, it memorizes the velocity of the point to be analyzed and returns. At the second call, with the velocity at t+Δt, VELDF calculates the average linear acceleration in Δt. The first three input parameters, $\Delta\phi$, ω and α, are the increment of the independent angle, the angular velocity and the acceleration of the driver. The other two input parameters, X_a and Y_a represent the linear velocity of the point to be analyzed. And the corresponding acceleration is delivered at the end as output parameters $X_{a'}$ and $Y_{a'}$.

The second category of subroutines includes three subroutines and one function program. They are useful when the three basic components are analyzed in the following sections of the chapter.

To use graphical methods for the linkage analysis, it is the first step to analyze the driver, because the kinematic parameters of the driver are usually given in an angular form and the variables of its moving end connected to the next basic component are unknown. Subroutine CRANK converts the given quantities into the necessary values defining the linear motion of the moving end.

Subroutine INTE is for searching the intersection of two straight lines. It is useful while constructing a velocity or acceleration polygon. Each of the two straight lines is defined by a point and the value of slope individually.

Function program AC identifies the direction of an angular velocity of a link. By supplying the positions and linear velocities of any two points on a link, one can obtain a value of AC, either 1 or -1, means clockwise or counterclockwise direction.

In writing subroutines for the displacement analysis of the RRR-RRR IV component, a subroutine EX1 is written to replace a repetitive computation procedure. Its information is also included in Table 3.2.

All subroutines or function program described so far will be employed sooner or later when the analysis of the three components and the corresponding linkages is carried out. All of their flow charts can be found in Appendix I. When any one of them is used for the first time, its flow chart will be attached at the end of the calling subroutine or main program. A table listing the related dummy and actual arguments is also provided.

3.2 Analysis of the RRR II Component and Its Application

Fig. 3.3 illustrates an RRR II component. Node B and D can be connected to other components or one of them to the frame. Usually

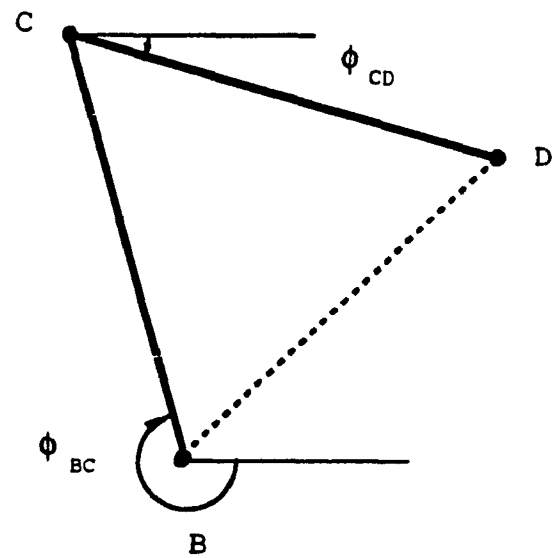


Fig. 3.3 RRR II basic component.

lengths of BC and CD as well as the kinematic parameters of the two outer node B and D are given. The problem is to derive the position, velocity and acceleration of the BC, CD and node C.

It can be observed that the position analysis about the inner node C is in fact a problem solved as case 2c of the loop-closure equation (2.1) and has been handled by subroutine TRICOS. In case that a coupler point on either BC or CD is to be located TRICOS is also applicable. Subroutine DLII is thus created.

The relative velocity method can be employed to obtain the linear velocity of node C. Subroutine INTE is useful while constructing a velocity polygon. By knowing the linear velocities of node B and D and lengths of BC and CD, the magnitudes of the angular velocities of BC and CD can be easily derived. Function program AC will provide them the directions. The linear velocity of a coupler point is obtained by calling IMAGE. The entire process about the velocity analysis has been programmed into subroutine VTII.

Similarly the problems in the acceleration analysis can be solved by relative acceleration method. And ACII is written for such a purpose.

So far three subroutines have been established to obtain the necessary kinematic values about an RRR II component. Table 3.3 systematically presents the information about these subroutines. Not only are the input and output parameters listed, but also the names of the employed secondary subroutines as well as the Norpak graphic subroutines are tabulated. Information about other subroutines created

Table 3.3 Subroutines for analysis of basic components

No	Component	Name	Function	Input Parameter	Output Parameter	Subroutine Called	Norpak Routine
1	RRRII	DLII	Position analysis	X_B, Y_B, X_D, Y_D, BC, CD, BE, CE, M, M'	X_C, Y_C, X_E, $\phi_{BC}, \phi_{CD}, \phi_{BE}$	TRICOS	
2	RRRII	VTII	Velocity analysis	$X_B, Y_B, X_C, Y_C, X_D, Y_D$, X_b, Y_b, X_d, Y_d, BC, CD	X_c, Y_c, ω_{BC}, ω_{DC}	INTE AC	
3	RRRII	ACII	Acceleration analysis	$X_B, Y_B, X_C, Y_C, X_D, Y_D$, $X_{b'}, Y_{b'}, X_{d'}, Y_{d'}$, BC, CD, ω_{BC}, ω_{DC}	$X_{c'}, Y_{c'}$, α_{BC}, α_{CD}	INTE AC	
4	RR-RR-RR III	DLIII	Position analysis	$X_B, Y_B, X_F, Y_F, X_G, Y_G$, $X_{E'}, Y_{E'}$, M, M', $\Delta\phi'$, $\Delta\phi$, BC, CD, DF, CE, DE, EG	$X_C, Y_C, X_D, Y_D, X_E, Y_E$, ϕ_{BC}, ϕ_{CD}, ϕ ϕ	CROS TRICOS	VDPINI SMASK SPIX LINE CIRCLE
5	RR-RR-RR III	VTIII	Velocity analysis	$X_B, Y_B, X_C, Y_C, X_D, Y_D$, $X_E, Y_E, X_F, Y_F, X_G, Y_G$, $X_S, Y_S, X_b, Y_b, X_f, Y_f$, X_g, Y_g, BC, CE, FD, DE, EG	$X_c, Y_c, X_d, Y_d, X_e, Y_e$, ω_{BC}, ω_{DC}, ω_{FD}, ω_{EG}	VTII	
6	RR-RR-RR III	ACIII	Acceleration analysis	$X_B, Y_B, X_C, Y_C, X_D, Y_D$, $X_E, Y_E, X_F, Y_F, X_G, Y_G$, $X_S, Y_S, X_{b'}, Y_{b'}, X_{f'}$, $Y_{f'}, X_{g'}, Y_{g'}, \omega_{BC}$, $\omega_{ED}, \omega_{FD}, \omega_{EG}$	$Y_{c'}, X_{c'}$, $X_{d'}, Y_{d'}$, $X_{e'}, Y_{e'}$, α_{BC}, α_{DC}, α_{EG}, α_{FD}	INTE ACII	
7	RRR-RRR IV	DLIV	Position analysis	X_B, Y_B, X_G, Y_G, BC, CD, BD, CE, EF, EG, FG, DF, c, $M_1, M_2, M_3, \phi_1, \Delta\phi$	X_C, Y_C, X_D, Y_D, X_E, Y_E, X_F, Y_F, ϕ_{CE}, ϕ_{DF}, ϕ_{EG}	EX1 CROS	VDPINI SMASK SPIX LINE RECT
8	RRR-RRR IV	VTIV	Velocity analysis	$X_B, Y_B, X_C, Y_C, X_D, Y_D$, $X_E, Y_E, X_F, Y_F, X_G, Y_G$, X_b, Y_b, X_g, Y_g, BC, CD, CE, DF	X_c, Y_c, X_d, Y_d, X_e, Y_e, X_f, Y_f, ω_{BC}, ω_{EF}, ω_{CE}, ω_{DF}	INTE VTII	
9	RRR-RRR IV	ACIV	Acceleration analysis	BC, CE, DF, EF, X_c, Y_c, $X_d, Y_d, X_e, Y_e, X_f, Y_f$, $X_{b'}, Y_{b'}$, $\Delta\phi, \omega, \alpha$	$X_{c'}, Y_{c'}$, $X_{d'}, Y_{d'}$, $X_{e'}, Y_{e'}$, $X_{f'}, Y_{f'}$, $\alpha_{DF}, \alpha_{EF}, \alpha_{BC}, \alpha_{CE}$	VELDF AC	

to analyze other components is also presented in Table 3.3.

After building a complete subroutine unit which analyzes a basic component, it is appropriate to test it by applying it to the analysis of a practical linkage. The simplest case would be a four-bar linkage. As shown in Fig. 3.4, the dimensions of the three moving links and the positions of the two base points are given. The driver AB rotates at a uniform speed ω. As it rotates to complete a full turn, it is necessary to evaluate the following variables: positions of node B, C and the coupler point E; angular velocities and accelerations of the coupler and follower; and finally the linear velocity and acceleration of the coupler point.

Obviously, any four-bar linkage consists of a frame, a driver and an RRR II component. The kinematic variables of node B can be derived by calling CRANK. Hence the necessary information about node B and D of the component is acquired. And TRICOS, VTII and ACII can therefore be called to perform the kinematic analysis. To deal with the coupler point, IMAGE is employed. The entire process is repeated as the driver proceeds increment by increment until it completes a full turn.

3.3 Analysis of RR-RR-RR III Component and Its Application

Fig. 3.5 illustrates an RR-RR-RR III component. Point C, D and E are the inner nodes whereas B, F and G are the outer nodes. Outer nodes can be connected to other components or to a frame but not more than two at the same time. The dimensions of the component and positions of the outer nodes are usually given. The positions of the inner nodes are to

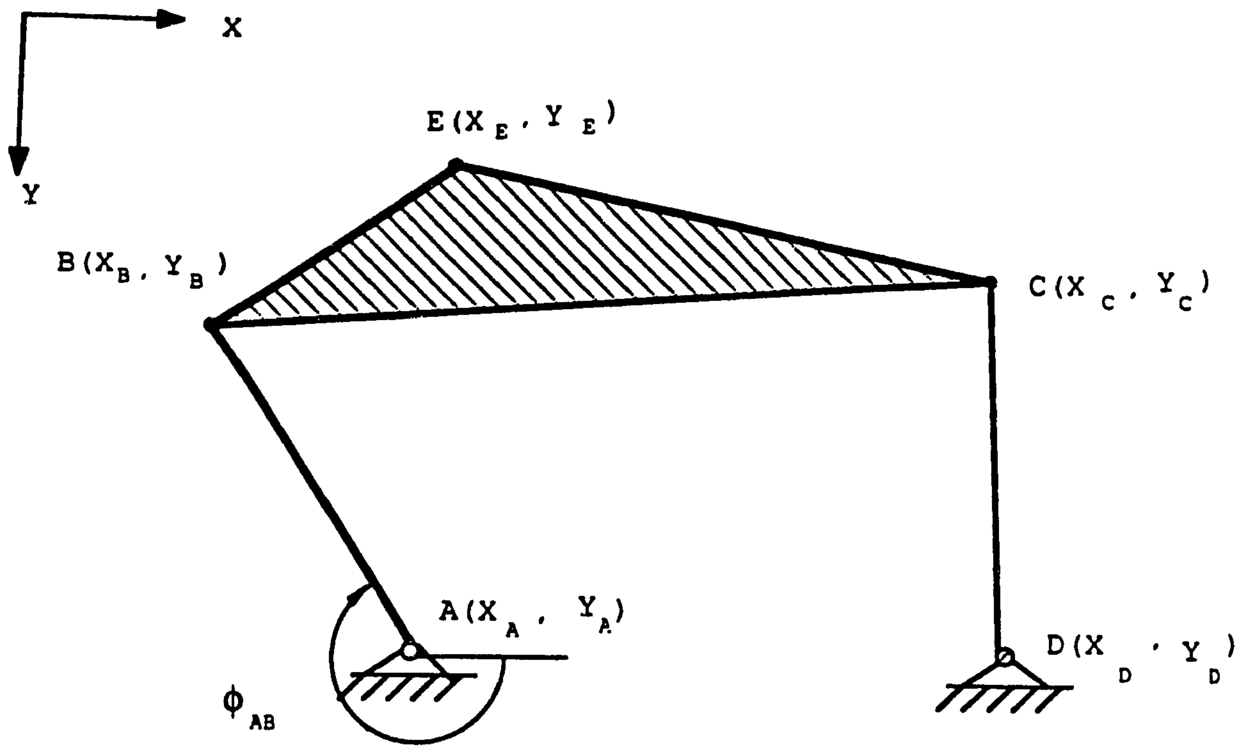


Fig. 3.4 A typical four-bar linkage.

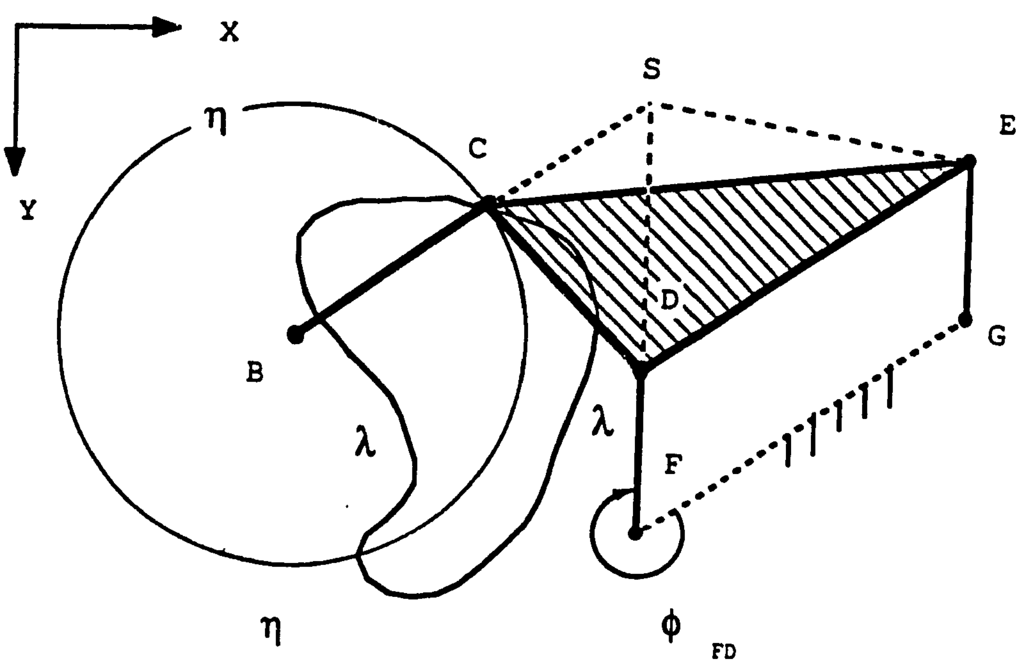


Fig. 3.5 Kinematic analysis of a RR-RR-RR III component.

be found. Furthermore, by providing the linear velocities and accelerations of the outer nodes, it is necessary to obtain the linear velocities and accelerations of the inner nodes and also the angular velocities and accelerations of all the links.

It is well known that the most difficult part in the kinematic analysis of planar mechanisms is the position analysis. For those relatively simpler ones such as the single loop mechanisms, the loop-closure equation method is handy. For multi-loop mechanisms, some special graphical methods have been developed. To analyze an RR-RR-RR III component, a method based on plane geometry has been employed in [20]. Refer to Fig. 3.5 again, assume that joint C is relaxed. Assume also that DF is the driver of a hypothetical four-bar linkage FDEG with ϕ_{FD} being the independent angle. Let link BC rotate around point B for 360^0 and form a circle $\eta\eta$. The main task in this position analysis is to locate the intersection(s) of the hypothetical coupler curve $\lambda\lambda$ and circle $\eta\eta$. Instead of drawing with pencil and paper, the method can be implemented on a monitor screen by applying the CAG techniques. Subroutine CROS is applicable in this case. Any point on $\lambda\lambda$ or $\eta\eta$ can be assigned to CROS to initiate the searching process. Consequently one or more intersection points can be found.

Subroutine DLIII has been written for implementing the method described above. As an input parameter, $\Delta\phi$ is taken as the increment of the independent angle. Smaller $\Delta\phi$ will result a smoother coupler curve whereby higher accuracy of the position of node C is attained. At the output DLIII delivers coordinates of the three inner nodes and angles of all four links. One point is worth to note. If subroutine DLIII is called consecutively, at the first call, more than one position of node C may be found. The user is asked to select the desirable one. With this information, DLIII will determine automatically the next position of C at the second call. The second position of C will serve to find the third one and so on. In fact, after the first call, DLIII will search in a region near the previous position of point C but not repeating the entire process all over again. Consequently significant saving in computation time can be achieved. Similar maneuver will be performed to the position analysis of the RRR-RRR IV component later.

The method of auxiliary point is used to the velocity and acceleration analysis [19]. The key concept involved is the so called auxiliary point S as shown in Fig. 3.5. It is found by drawing two intersecting lines, CS and DS which are the extensions of BC and FD respectively. It is an imaginary point in link CDE. Because of its special position, BSF can be viewed as an RRR dyad. Thus VTII and ACII are applicable for finding the velocity and acceleration of point S with the given linear velocities and accelerations of node B and F. Then consider SEG as another dyad to solve for node E and so as BCE for node C and finally FDE for node D. Therefore kinematic variables of all three inner nodes are obtained. The next step is to evaluate the angular velocities and accelerations of the four links. This analysis process has been programmed into subroutine VTIII and ACIII respectively. A few subroutines including those described in section 4.1 and those used in the analysis of the RRR dyad have been employed. Some basic graphic subroutines from the Norpak graphics package are also utilized [21].

A Stephenson type III linkage is to be analyzed as followed. As shown in Fig. 3.6, ϕ_{AB} is the independent angle. By knowing the input angular velocity ω and acceleration α, all kinematic variables of node B, C, D and E are to be determined.

By observation, the linkage consists of three elements, a driver AB, an RR-RR-RR III component and a frame. An outer node of the RR-RR-RR III component is connected to the driver and the other two to the frame. Therefore subroutine CRANK, DLIII, VTIII and ACIII can be called sequentially for a given ϕ_{AB} to determine the positions, velocities and accelerations of the desired nodes. Then repeat the entire procedure as ϕ_{AB} proceeds by an increment $\Delta\phi$ and so on until either finishing a complete turn or encountering a dead zone. In program SIXB this analysis is implemented. Three more input parameters are necessary. $\Delta\phi'$ is the increment of ϕ_{FD} needed to draw the hypothetical coupler curve. With this parameter, the user can specify the desirable accuracy for the position analysis. The other two parameters M and M' indicate the configuration of the hypothetical dyad DEG and DCE. Flow chart of SIXB is attached in Appendix I.

3.4 Analysis of RRR-RRR IV Component and Its Application

Fig. 3.7 illustrates an RRR-RRR IV component where C, D, F and G are the inner nodes whereas B and G are the outer nodes. Outer nodes can be connected to other components or one of them to the frame. By providing the dimensions of the links and kinematic parameters about the outer nodes, analysis is to be carried out with regard to the inner nodes. Angular positions, velocities and accelerations of all four

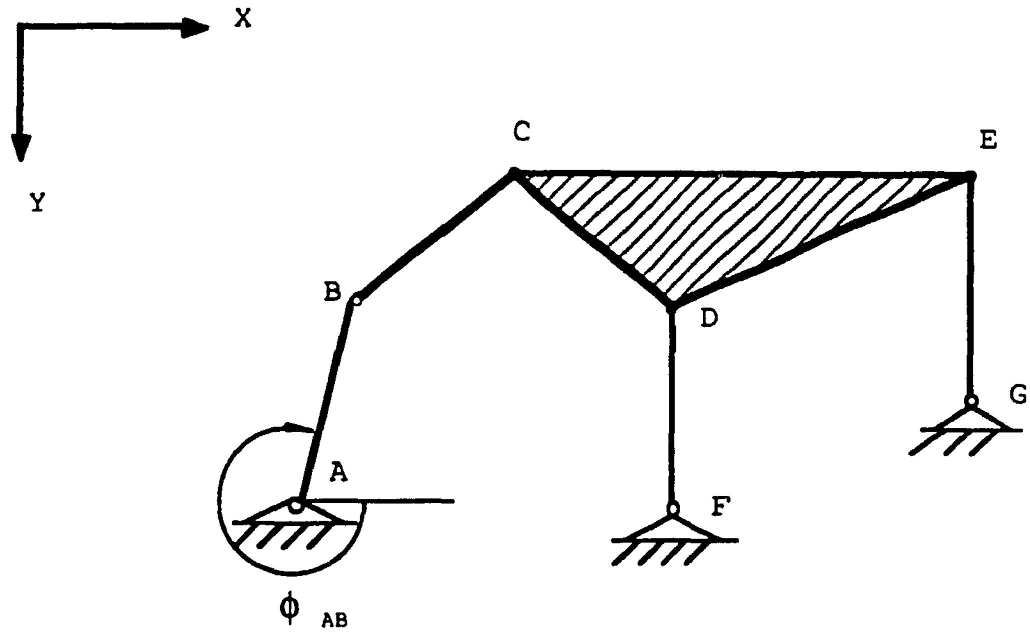


Fig. 3.6 Stephenson type III linkage.

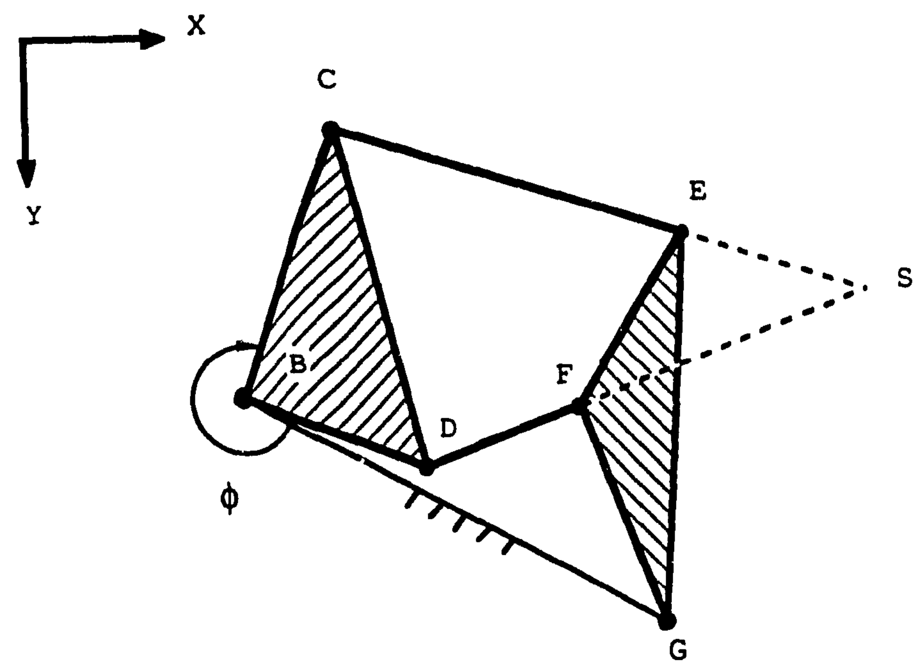


Fig. 3.7 Kinematic analysis of a RRR-RRR IV component.

links will also be found.

The method of assumed point, a relatively general technique developed for components with complex structure [2], can be used for the position analysis. Hypothetically, link DF is removed. And assumed that the line connecting node B and G is the frame so that a four-bar linkage BCEG is formed. Both BC and EG can be a driver. In this case, BC is selected. Give it an angular position ϕ. Position of node C is then defined. Consider CE and EG as an RRR dyad and apply subroutine TRICOS to search the position of node E. Apply TRICOS also to find the positions of node D and F. The distance between the two nodes is then calculated as

$$F_i = \sqrt{(X_F - X_D)^2 + (Y_F - Y_D)^2} \tag{3.1}$$

Usually this distance does not match the link length DF which has been given as the initial condition. Then let link BC proceed a small increment $\Delta\phi$ and find F_{i+1}. If this procedure is carried out for the entire valid range of ϕ, a series values of $F(\phi)$ will result. As shown in Fig. 3.8, a curve depicting these values are drawn together with a straight line

$$Y = DF \tag{3.2}$$

If the drawing is done on a monitor screen, CROS can be called to find the intersection(s) or the tangent point(s) of the two. With the value(s) of ϕ, the configuration(s) of the component is hence defined. Subroutine DLIV is created to perform the position analysis. As input parameter, ε dictates the allowable error in comparing the distance

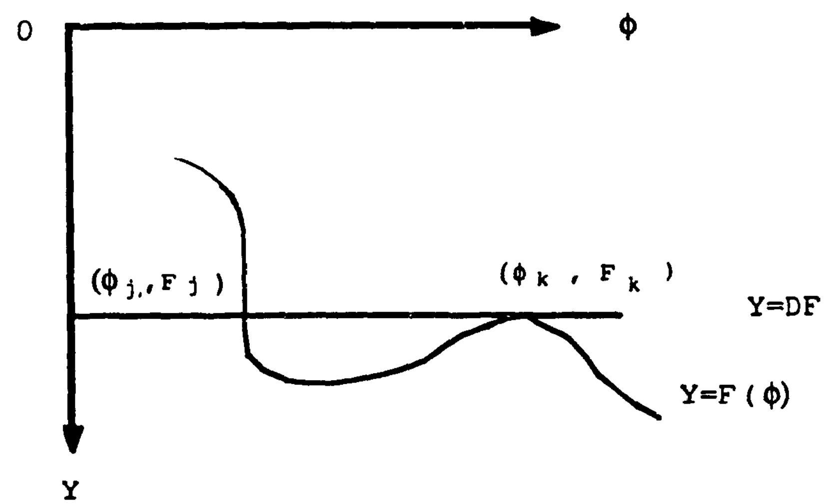


Fig. 3.8 Searching for Φ to satisfy DF.

between D and F with the true length of DF. If DLIV is needed to be called repetitively, at the first call, it may find more than one ϕ that satisfies the requirement. The user will be asked to select which ϕ, in other words, which configuration of the component is required. Based on the selected ϕ, DLIV can determine the subsequent ϕ's when it is called later.

The instantaneous center method is used to perform the velocity analysis here [19]. Refer to Fig. 3.7 again. Point S is found by extending CE and DF. And it is the imaginary center of the link BCD and EFG. At such an instant, BSG can be viewed as an RRR dyad. Then VTII is applicable to find the velocity of point S and the angular velocities of BS as well as SG. These two angular velocities are equal to those of link BCD and EFG respectively. Thus all other unknowns are easily derived. This velocity analysis has been implemented in subroutine VTIV.

ACIV is for the acceleration analysis. Here subroutine VELDF is used several times to find the accelerations of the four inner nodes. Due to the nature of the velocity difference method, an acceleration derived based on this method is the average value of a node in a position interval corresponding to the increment of the input angle such as $\Delta\phi$.

To demonstrate the application of the subroutine unit created in this section, a Stephenson type II linkage is to be solved. As shown in Fig. 3.9, all necessary dimensions and the driver position, velocity and acceleration are provided. It is necessary to find the kinematic variables of all five nodes, B, C, D, E and F. By observation, the

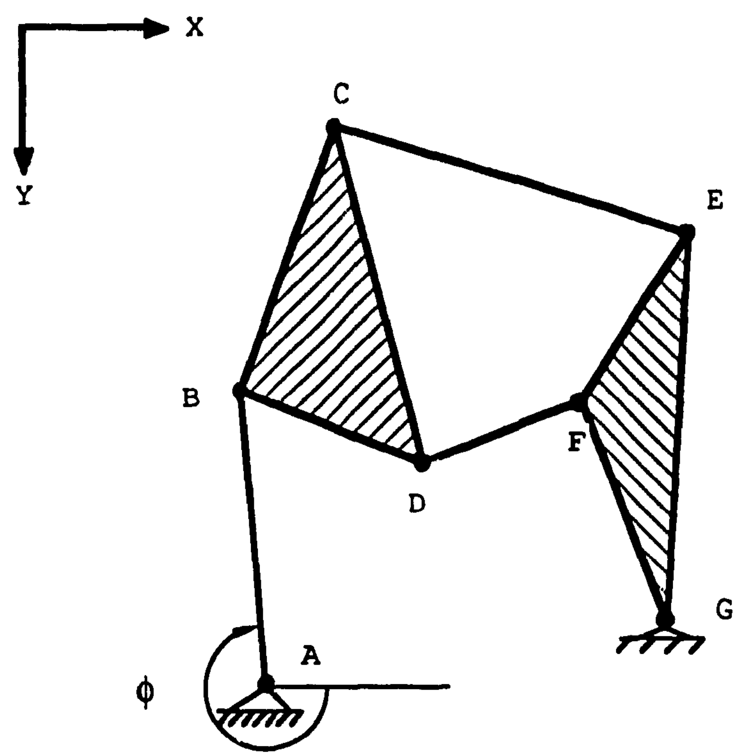


Fig. 3.9 Stephenson type II linkage.

linkage consists of a driver AB, an RRR-RRR IV component and a frame. Therefore in program STPN, subroutine CRANK, DLIV, VTIV and ACIV are called sequentially for the necessary values. As ϕ proceeds within its valid range, the entire calling process is repeated until it either completes a full turn or falls in a dead zone. The flow chart of this program as well as the list of the related dummy and actual arguments are given in Appendix I.

3.5 General Discussion

With CAG techniques a few typical graphical methods have been implemented to analyze three basic components. Two aspects with regard to such an implementation will be discussed, starting with the graphical methods used for the analysis.

Refer to Table 3.4 where methods employed in this chapter are summarized. The Chace's method is systematic and its graphical solutions of the general vector equation are simple and clear. The solution case 2c is used to create the subroutine TRICOS for the analysis of an RRR II component. It can be seen that TRICOS has been repetitively called later when position analysis of other two components are performed. It makes the analysis more organized and hence easier to understand. Similar phenomenon is observed from VTII and ACII where relative velocity and acceleration method is employed. Eventually significant saving in software composing is achieved. More details about the analysis of an RRR II component can be found in next chapter.

The special method used in the position analysis of an RR-RR-RR

Table 3.4 Graphical methods used to analyze the three components.

Names of components	Position analysis	Velocity analysis	Acceleration analysis
RRR II	loop-closure equation	relative velocity	relative acceleration
RR-RR-RR III	special	auxiliary point	auxiliary point
RRR-RRR IV	assumed point	instantaneous center	velocity difference

III component shares some common characteristics with the method of assumed point implemented to analyze the RRR-RRR IV component. With respect to the methodology, a link in the component is first relaxed so that the other parts of the component form a single loop mechanism, specifically a four-bar linkage. Then let the hypothetical driver rotate throughout the valid range and perform the position analysis in an exactly same way as performing the analysis of an independent four-bar linkage. The only difference of the two methods is the way to search the position(s) of the hypothetical coupler point(s) to satisfy the constraints of the component. For the analysis of the RR-RR-RR III component, the coupler curve is directly drawn on the screen. For the RRR-RRR IV component, on the other hand, the function values representing the distance between a point on the driver and another on the follower are plotted. But in both cases, CROS is handy in searching the intersections of curves. It can be shown that the method of assumed point is also applicable to the analysis of the RR-RR-RR III component. The sole purpose for selecting two different methods is to demonstrate the flexibility of the graphical methods which can be used in different ways in case it is necessary. Both ways are computationally efficient. Corresponding to each increment of the hypothetical driver, function value is evaluated only once. All the calculations are very simple. Compared with the analytical method used in [2] which needs Taylor's series expansion and Newton-Ralphson iteration method to solve quite large matrices, these methods are much more straightforward and efficient. The auxiliary point method is actually based on the principle of the relative velocity and acceleration method [20] [19]. The key concept involved is the special point S. The graphical construction needed is very simple. The subroutines written for the RRR dyad based on relative velocity and acceleration method have been used in many occasions.

With respect to the instantaneous center method applied to the RRR-RRR IV component, simplicity is the common characteristic in concept, graphical construction, computation and programming. Brevity is also obvious as the velocity difference method is incorporated in ACIV. Although the accuracy is restricted due to its nature, higher accuracy can be attained by choosing a smaller time interval. Furthermore the restriction that the input acceleration must be uniform usually does not jeopardize its application for the reason that most practical linkages run at either constant velocity or constant acceleration.

Obviously a variety of simple, efficient and programmable graphical methods are available for researchers to establish a sophisticated package for kinematic analysis.

Another aspect to discuss in regard to the implementation of the graphical method is about the mathematical analysis. Unlike many packages based on analytical methods in which concepts of the advanced mathematics such as tensors, screw matrices, partial differentiation equations and so on are necessary, throughout the software established thus far the mathematics employed is very simple. The extensive application of geometry, especially the analytic geometry, not only makes the implementation of the computer aided graphical methods more understandable, but also maintain the desirable accuracy. Moreover, the powerful CAG techniques make it possible to analyze the relatively complicated linkages with simple mathematics. Thus the advantages of the graphical methods such as simplicity and effectiveness are realized without sacrificing the efficiency and accuracy.

CHAPTER 4

IMPLEMENTING THE COMPUTER AIDED GRAPHICAL METHODS ON A VAX/VMS COMPUTER

The computer aided graphical methods are implemented on a VAX/VMS computer system to analyze a four-bar linkage in this chapter. Because of the importance of CROS, it is applied to construct the velocity and acceleration polygons. The approach developed in the previous chapter which is based on analytic geometry and characterized by the subroutine INTE, is also applied. Results of the analysis are generated from both approaches and then plotted. The effectiveness and efficiency of CROS will be discussed based on the comparison. In order to improve the friendliness of the software, interactive programming techniques are extensively applied.

In the first part of the chapter, some utility subroutines are formed. Then two programs utilizing either CROS or INTE to analyze a four-bar linkage are explained. Finally the acquired numerical results as well as visual effect are discussed.

4.1 Description of the Assistant Subroutines

To obtain properly displayed images and clearly denoted graphical results, two sets of utility subroutines are created.

In applying relative velocity and acceleration method, polygons of different sizes are to be constructed at different locations of the screen. Sometimes they may not fit the screen boundary. Thus scaling and/or shifting of the polygons will be necessary. Therefore subroutine MAXMIN, ARRANE, SCALE and SHIFT are created and related information is provided in Table 4.1. ARRANE is the main subroutine determining the amount to scale and/or shift. MAXMIN pre-processes data for ARRANE while SCALE and SHIFT are called inside ARRANE for the scaling and shifting.

Assume that point B and C are two pre-defined points on a rigid link in planar motion. $\vec{V}_B$, the velocity of point B, is completely known. Directions of $\vec{V}_{CB}$ and $\vec{V}_C$ are also known. Fig. 4.1 illustrates the procedure to construct the velocity polygon for finding the magnitudes of $\vec{V}_{CB}$ and $\vec{V}_C$, in other words location of point c. If CROS is used, origin of the polygon point $\grave{o}$ has to be defined mentally by estimating the size and location of the polygon. Otherwise the origins of the coordinate system and the polygon are coincident. Position of point b is hence found. With the values of slopes, line m_1 and m_2 can be drawn. One way to locate point c is to solve the algebraic equations of m_1 and m_2. As described in chapter 3, subroutine INTE is for such a purpose. This method will be used in program CH_TR. As mentioned before, another way is to search the intersection by calling CROS. In such a case, both lines must be clipped before displaying. It is therefore necessary to find the points where the lines cross the boundary of the window assigned to display the polygon. Subroutine CALY, CLIP and MODES are therefore written and they are listed in Table 4.2. Refer to Fig. 4.1 again. After point c is located, it is desirable to denote the polygon. By calling subroutine COLOR, ob, oc and bc will be displayed in different colors and the meaning of each

Table 4.1 Utility subroutines used in CH-TR

No	Name	Function	Input Parameter	Output Parameter
1	MAXMIN	Finds the rectangle confining the given polygon.	AXA, AYA, AXB, AYB, AXC, AYC, AXD, AYD, AXE, AYE	AXMA, AYMA AXMI, AYMI NA
2	ARRANE	Determines the amount to scale and / or shift a misplaced and / or oversized polygon.	AXMI, AYMI, AXMA, AYMA AXQ, AYQ, AXP, AYP, AXR, AYR, AXD, AYD, AXE, AYE	
3	SCALE	Scales a polygon to display it on the screen.	AD, AB, NA, AK AXA, AYA, AXB, AYB, AXC, AYC, AXD AYD, AXE, AYE	
4	SHIFT	Shifts a polygon to display it on the screen.	ASHIFX, ASHIFY NA AXA, AYA, AXB, AYB, AXC, AYC, AXD AYD, AXE, AYE	

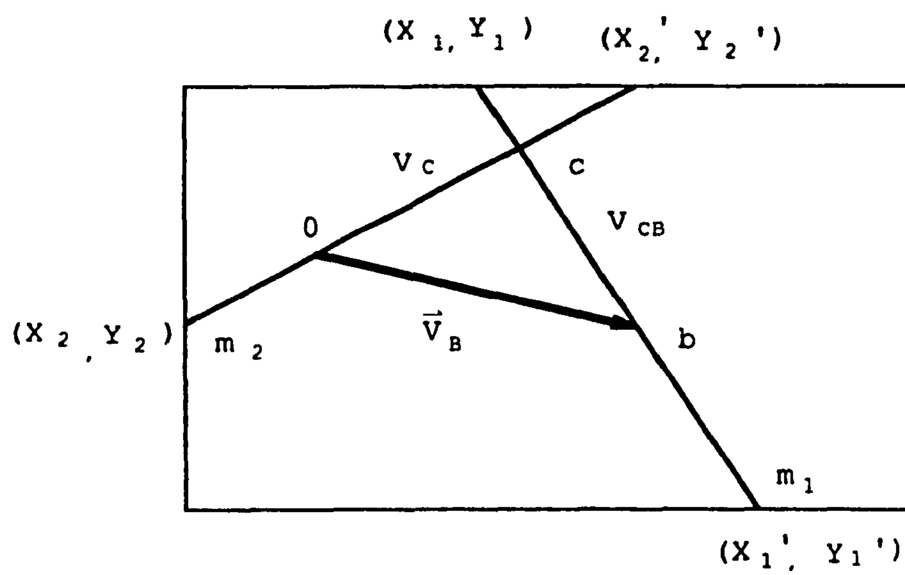


Fig. 4.1 Constructing a velocity polygon on a monitor screen.

Table 4.2 Utility subroutines used in CH-CR

<table>
<tr><th>No</th><th>Name</th><th>Function</th><th>Input Parameter</th><th>Output Parameter</th></tr>
<tr><td rowspan="2">1</td><td rowspan="2">CLIP</td><td rowspan="2">Clip's a straight line to display it on a monitor screen.</td><td>XQ, YQ, DX, DY</td><td></td></tr>
<tr><td colspan="2">X1D, Y1D, X2D, Y2D</td></tr>
<tr><td>2</td><td>MODES</td><td>As part of CLIP, finds the position of a line on a screen coordinate system.</td><td>X, Y, DX, DY</td><td>IX, IY</td></tr>
<tr><td>3</td><td>CALY</td><td>Evaluates the Y coordinates of the two endpoints of a line to be clipped.</td><td>XR, YR, SLOPE</td><td>X1D, Y1D
X2D, Y2D</td></tr>
<tr><td>4</td><td>COLOR</td><td>Draw a triangle with different colours on bank 1.</td><td>XT, YT, XU, YU
XV, YV, SX</td><td></td></tr>
<tr><td>5</td><td>ABD</td><td>Denotes three sides of a triangle.</td><td colspan="2">XE1, YE1, XE2, YE2
XE3, YE3, H, HA</td></tr>
</table>

color is listed on the lower left corner of the screen. In other occasions, proper notations can be written adjacent to the corresponding sides of the polygon regardless the shape and location of the polygon. Subroutine ABD is for such a purpose and its information is included in Table 4.2 also. All subroutines introduced thus far are also applicable for the construction of acceleration polygon.

4.2 Analysis of a Four-bar Linkage

4.2.1 Applying CROS

In Fabrikant's paper [17], position analysis of a four-bar linkage has been carried out by employing subroutine CROS. It is therefore appropriate to investigate what if CROS is applied to perform the velocity and acceleration analysis. A computer software is therefore developed. To initiate the investigation, a four-bar linkage is first simulated. The software written by Fabrikant [17] is thus applicable. It includes the main program CH_CR, subroutine RESTORE, MV, DRAW, CROS and SWITCH. Interactive programming techniques have been employed in the main program CH_CR. Once the linkage starts to move, the user will have four options:

i) observe the simulation till the crank completes a full turn

ii) punch key "A" to request the velocity analysis. The velocity polygon will be shown on the screen after giving the input angular speed followed by a cartridge return. The simulation can be continued by typing any key.

iii) hit key "B" to request an acceleration analysis. The acceleration polygon will appear on the screen after entering the desired ω.

The simulation can be resumed by hitting any key.

iv) stop the entire program by punching "S".

Procedure i) to iii) can be repeated randomly. No interference in image displaying will result.

The second part of the software consists of subroutine VSCH_CR, ASCH_CR, CALY, CLIP, MODES, ABD and COLOR. The first two handle the velocity and acceleration analysis and then display the required results whenever desirable. The last four have been discussed in the previous section.

4.2.2 Applying INTE

In program CH_TR, subroutine INTE replaces CROS to perform the kinematic analysis. Most of the contents in CH_TR is the same as those in CH_CR with two major differences. First of all, subroutines CALY, CLIP and MODES are unnecessary. As described earlier, all of them are for drawing line m_1 and m_2 as shown in Fig. 4.1. Here INTE directly calculates the position of point C instead of searching for it on the screen. Another difference is that automatic scaling and shifting are practical and therefore performed. When CROS is used, point C can be found only if m_1 and m_2 intersect inside the monitor screen. It is very difficult if not impossible to anticipate where they will intersect though. In case INTE is employed, point C is evaluated before the displaying process. It is quite easy to determine the size and location of the polygon. So appropriate scaling and/or shifting can be carried out if necessary. The lists of input and output parameters for VSCH_CR, ASCH_CR, VSCH_TR and ASCH_TR can be found in Table 4.3.

Table 4.3 Subroutines for analysis of a four-bar linkage

No.	Name	Function	Input Parameter	Output Parameter
1	VSCH_CR	Performs velocity analysis and display the procedure.	X_{A}, Y_{A}, X_{B}, Y_{B}, X_{C} Y_{C}, Y_{D}, [illegible], NN, N[illegible], I9	XQ, YQ, XP, YP, RRX, RRY
2	ASCH_CR	Performs acceleration analysis and display the procedure. (by using CROS)	X_{A}, Y_{A}, X_{B}, Y_{B}, X_{C} Y_{C}, X_{D}, Y_{D}, VEL1, VEL2, VEL3, DX, DY, HX, I9	LXI, LYI, LX1, LY1, LX2, LY2, LX3, LY3, RX2, RY2
3	VSCH_TR	Performs velocity analysis and display the procedure.	X_{A}, Y_{A}, X_{B}, Y_{B}, X_{C} Y_{C}, X_{D}, Y_{D}, NN2, NN1, I	AXQ, AYQ, AXP, AYP, AXR, AYR,
4	ASCH_TR	Performs acceleration analysis and display the procedure. (by using INTE)	X_{A}, Y_{A}, X_{B}, Y_{B}, X_{C} Y_{C}, X_{D}, Y_{D}, VEL1, VEL2, VEL3, NX	AXI, AYI, AX1, AY1, AX2, AY2, AX3, AY3, AXR, AYR

4.2.3 Data generation and results

Program CH_CR and CH_TR have been modified to eliminate the effect of the interactive programming techniques so that the velocity and acceleration analysis can be conducted for every position of the linkage when the crank proceeds increment by increment until completing a full cycle. The angular velocities of the coupler and follower, ω_2 and ω_3, as well as their accelerations, α_2 and α_3, are derived. The corresponding nominal values are hence calculated by dividing the velocity values with the input angular velocity ω_1 and by dividing the accelerations with ω_1^2 respectively. Velocities and accelerations of the coupler generated from CH_CR and those from CH_TR are plotted against the crank angle θ_1 in Fig. 4.2 for comparison. Curve 1 represents $\frac{\omega_2}{\omega_1}$ versus θ_1 and curve 2 depicts $\frac{\alpha_2}{\omega_1^2}$ versus θ_1. The same comparison for the follower is shown in Fig. 4.3. All these results are derived based on the following conditions:

$\omega_1 = 20$ rpm

$\alpha_1 = 0$

AB = 25 (in pixel units)

BC = 100

CD =50

AD = 110

where the dimensions represent the lengths of the driver, coupler, follower and the distance between the two bases. Fig. 4.4 and 4.5 are the similar plots with ω_1 changed to 10 rpm. Fig. 4.6 through Fig. 4.9 are for another setup:

$\omega_1 = 16$

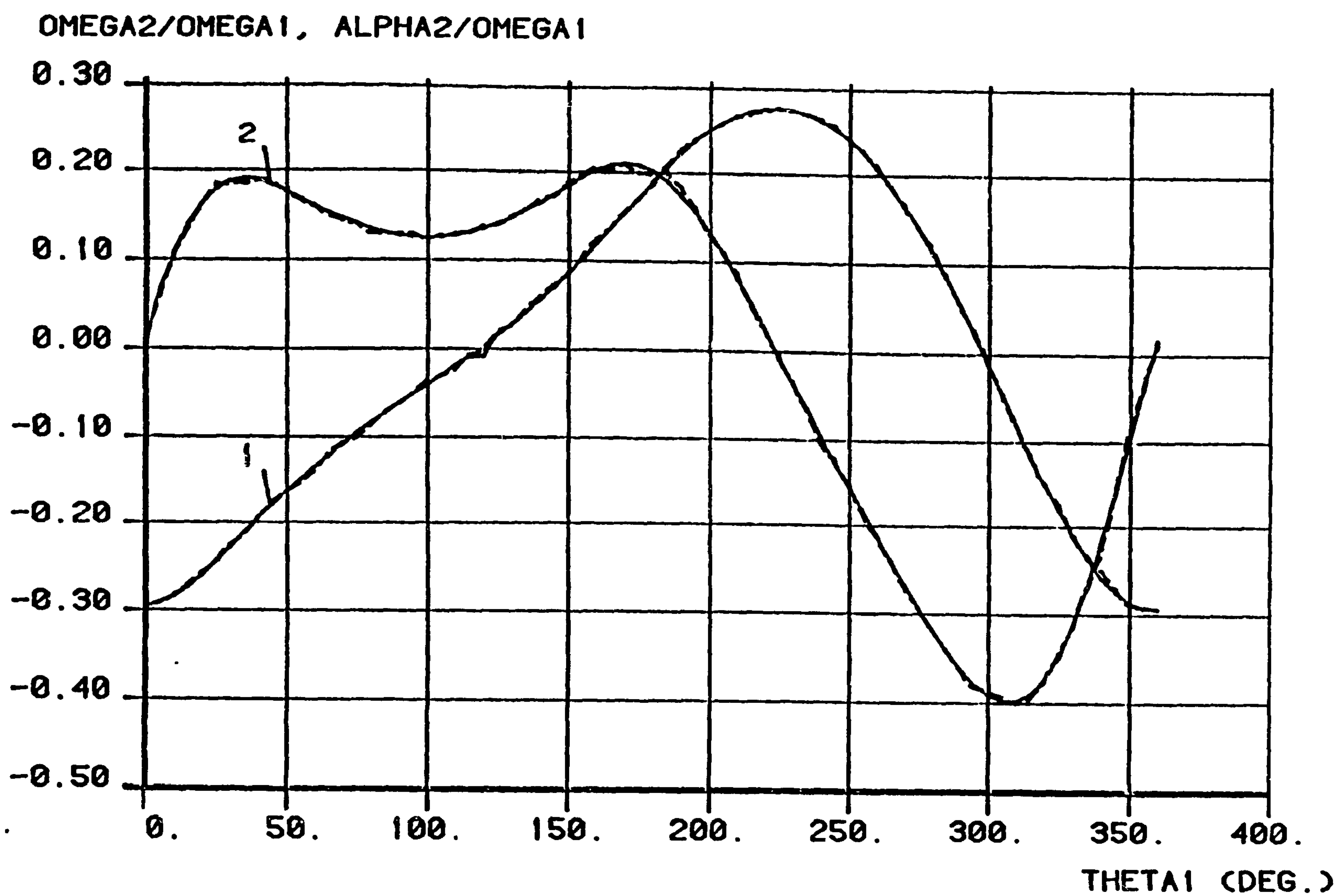


Fig. 4.2 KINEMATIC ANALYSIS OF THE COUPLER: TEST CASE (1)

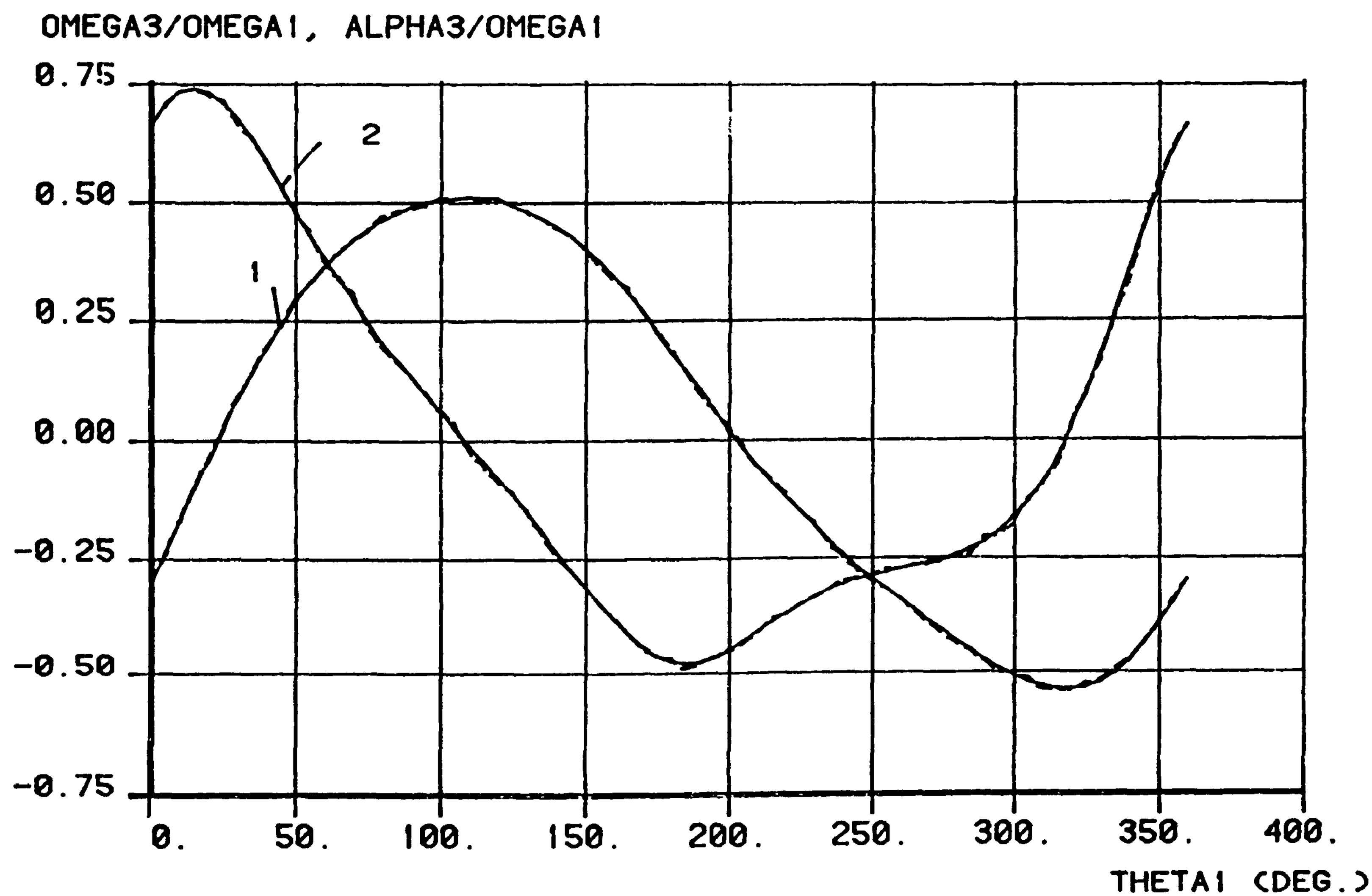


Fig. 4.3 KINEMATIC ANALYSIS OF THE FOLLOWER: TEST CASE (1)

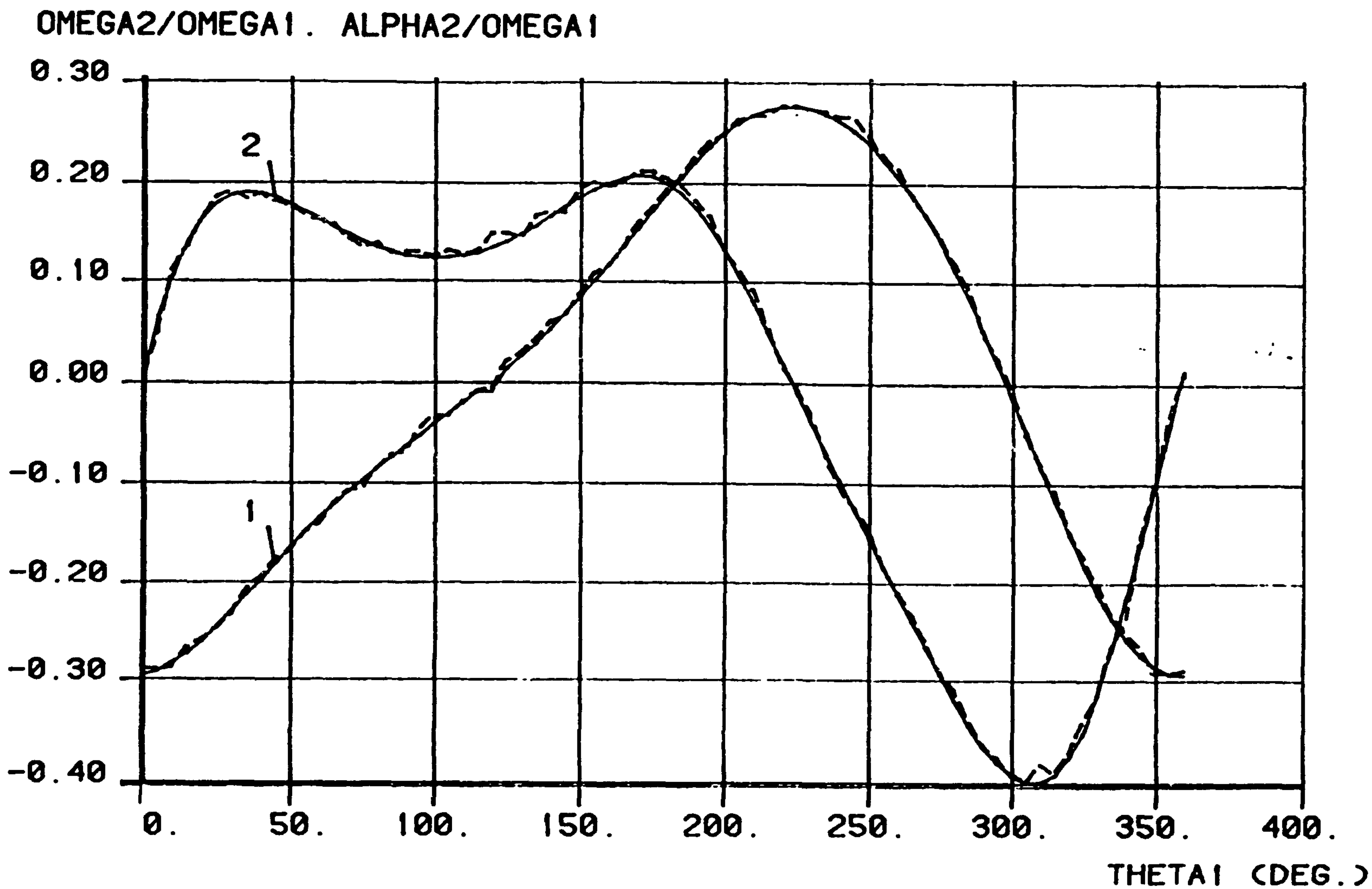


Fig. 4.4 KINEMATIC ANALYSIS OF THE COUPLER: TEST CASE (2)

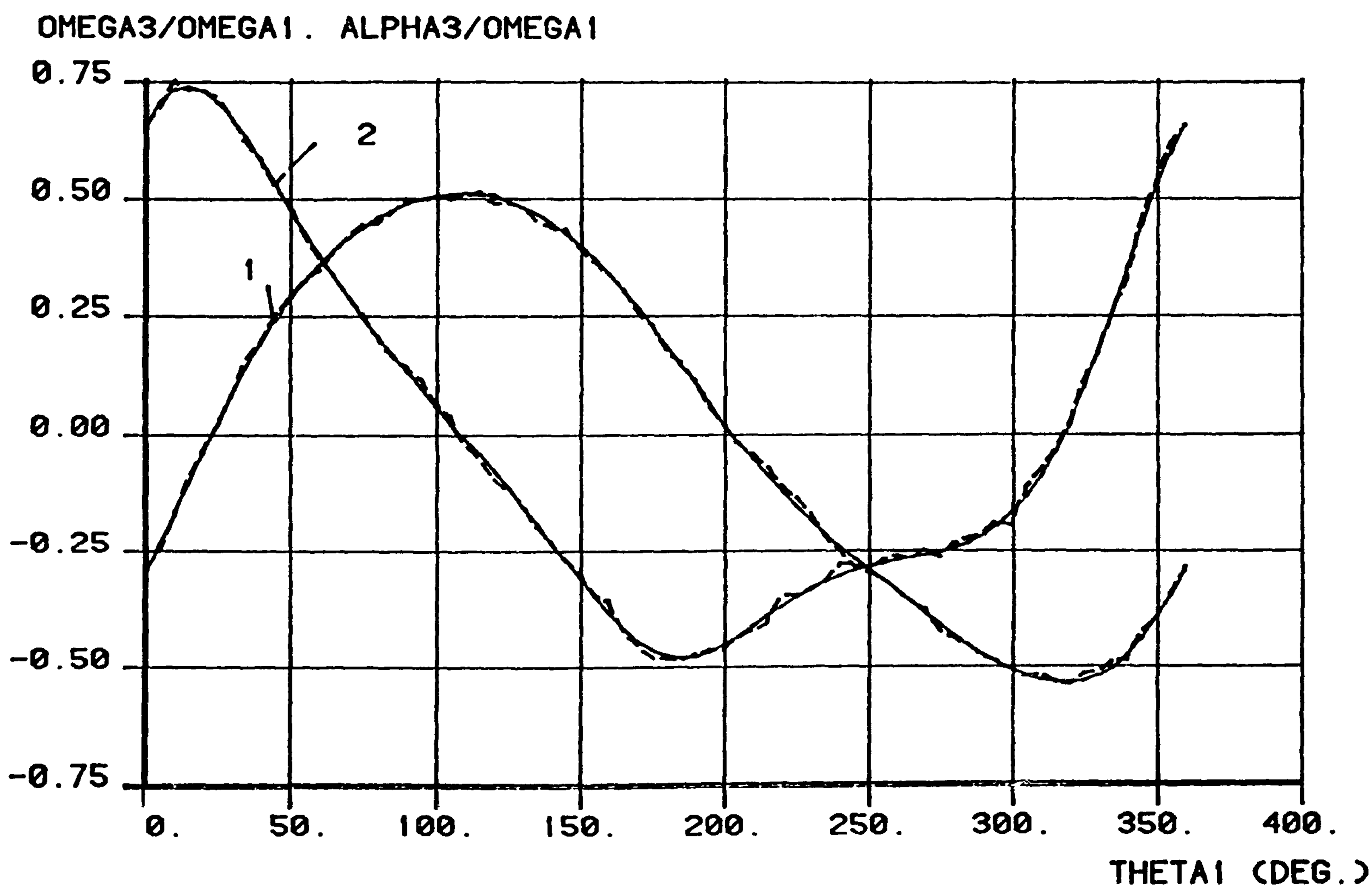


Fig. 4.5 KINEMATIC ANALYSIS OF THE FOLLOWER: TEST CASE (2)

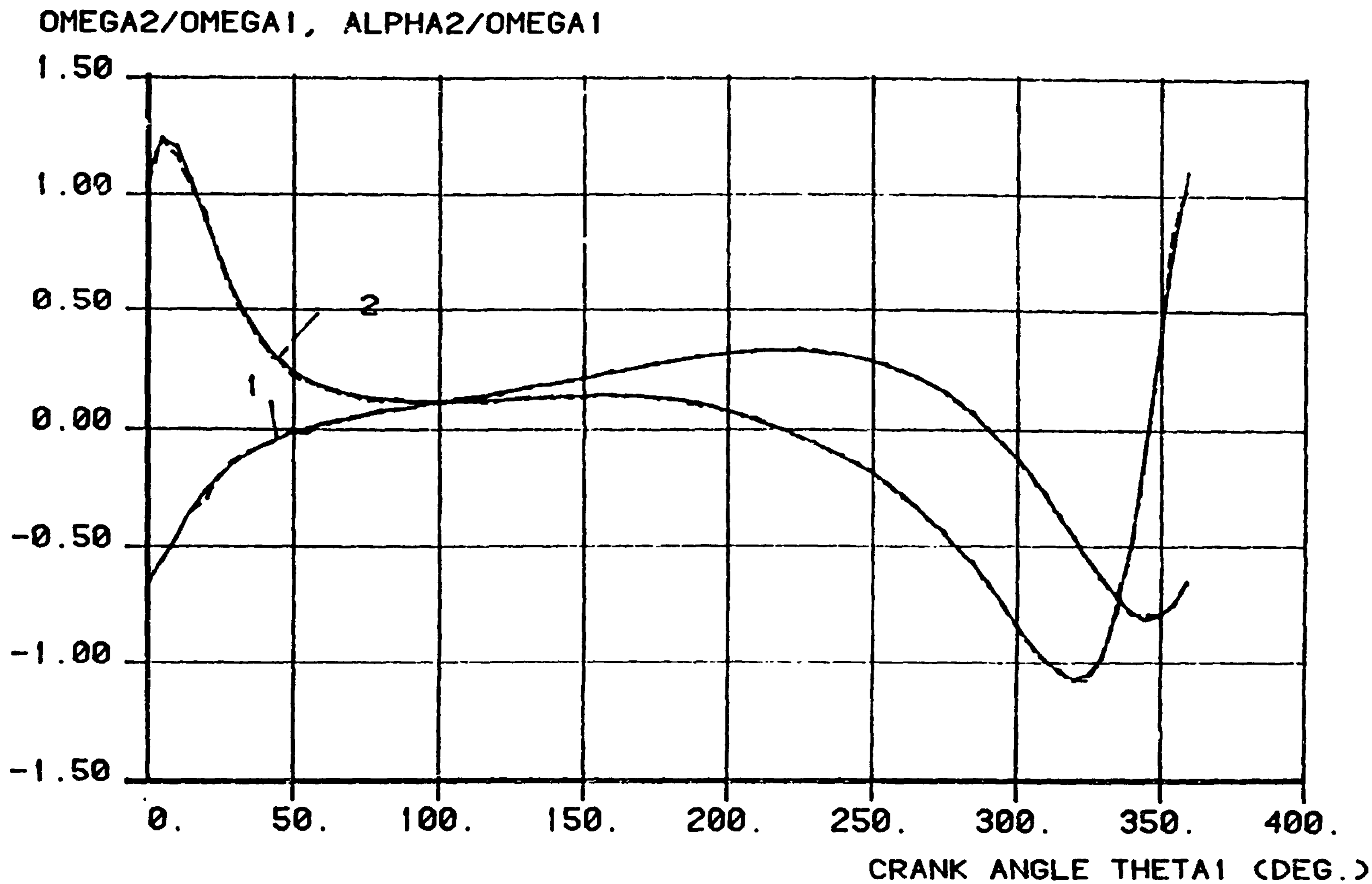


Fig. 4.6 KINEMATIC ANALYSIS OF THE COUPLER: TEST CASE (3)

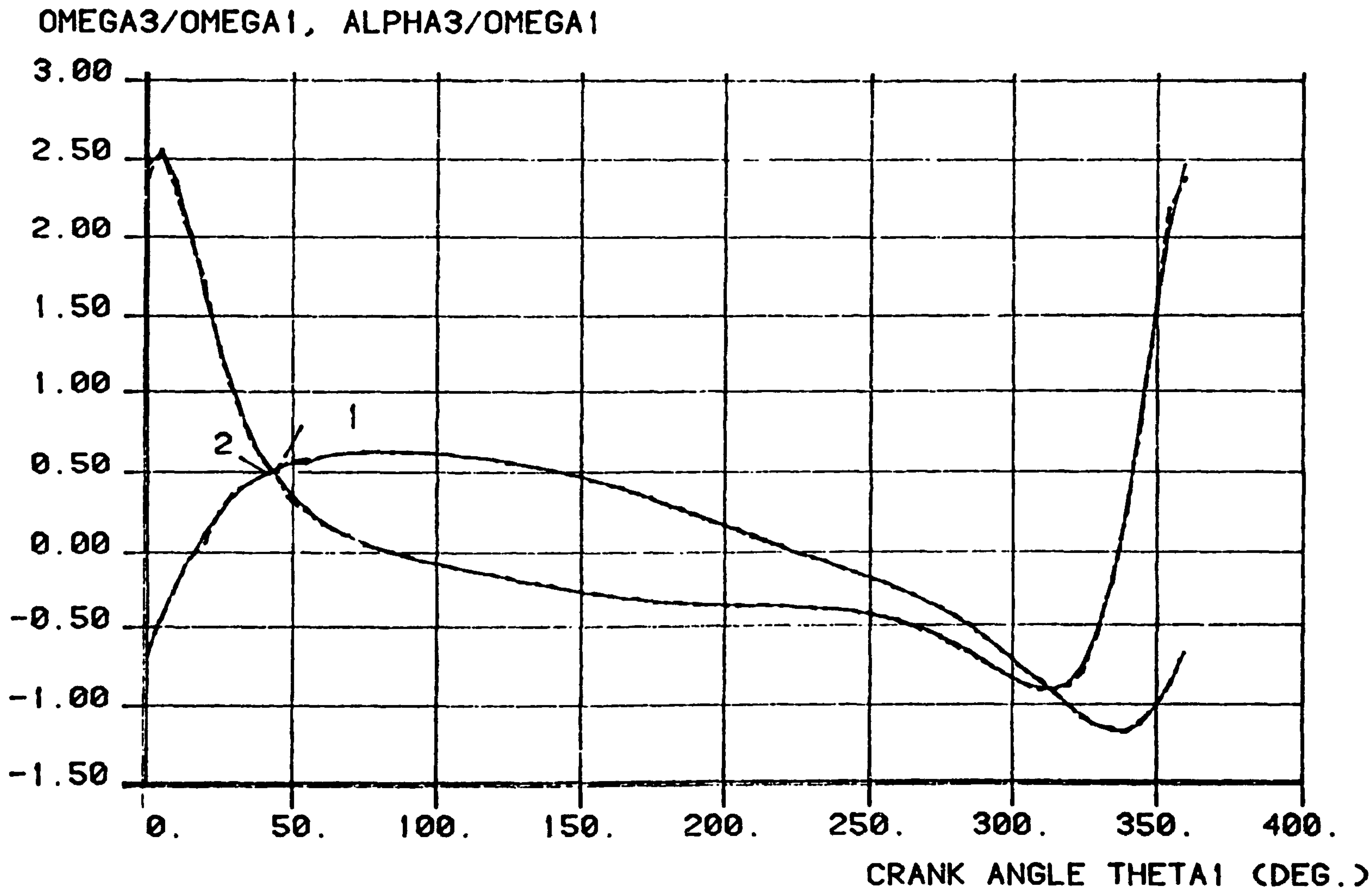


Fig. 4.7 KINEMATIC ANALYSIS OF THE FOLLOWER: TEST CASE (3)

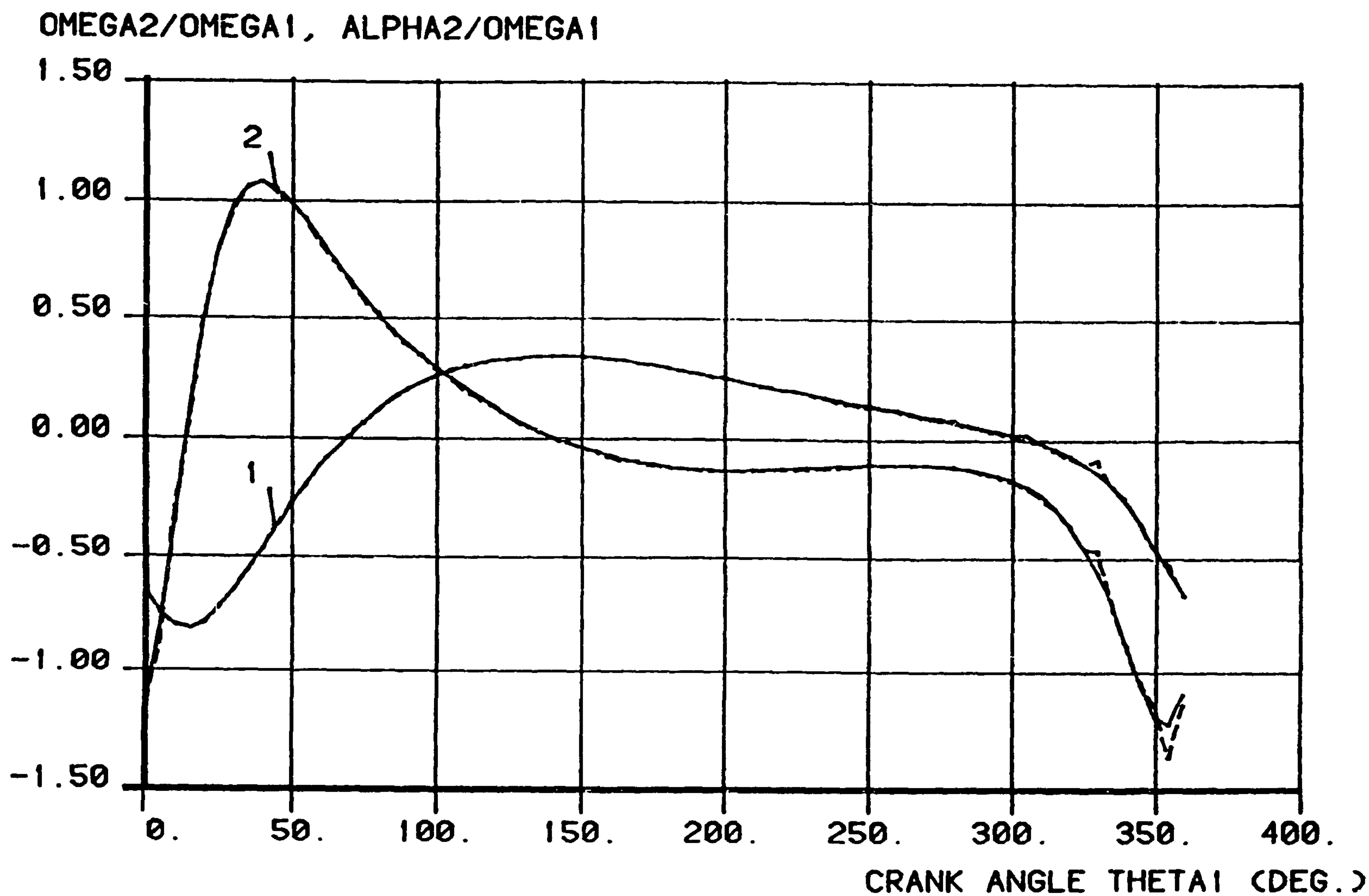


Fig. 4.8 KINEMATIC ANALYSIS OF THE COUPLER: TEST CASE (4)

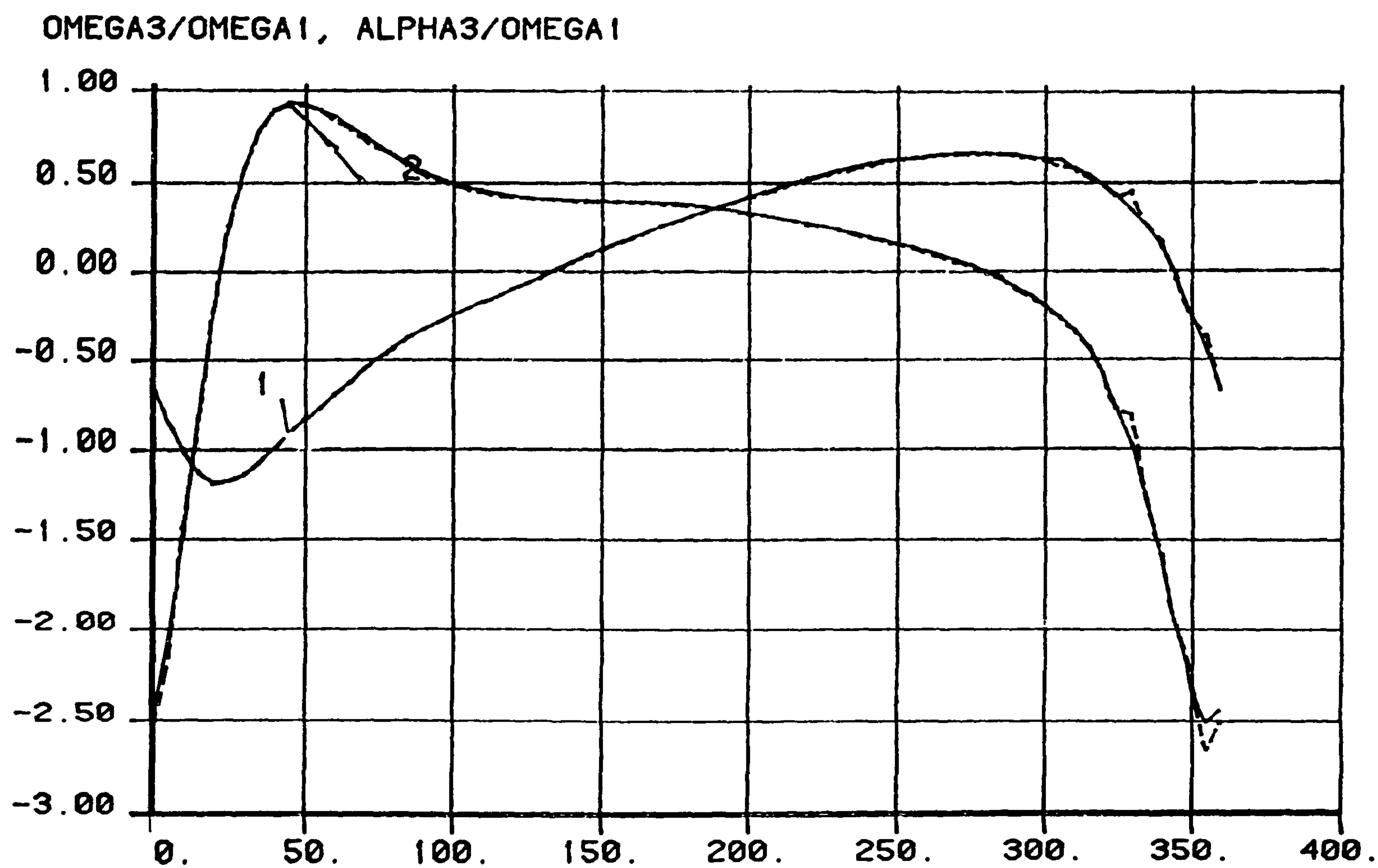


Fig. 4.9 KINEMATIC ANALYSIS OF THE FOLLOWER: TEST CASE (4)

$\alpha_1 = 0$

AB = 40

BC = 120

CD = 70

AD = 100.

The first two figures are for the uncrossed configuration of the linkage and the last two for the crossed configuration. Photos are taken on the images from the screen when CH_CR runs based on this set of given condition. Fig. 4.10 and Fig. 4.11 illustrate the velocity and acceleration polygons of the uncrossed configuration. Fig. 4.12 and Fig. 4.13 show the results at another position of the linkage. Results of the crossed configuration are given in Fig. 4.14 and 4.15.

4.3 General Discussion

4.3.1 Validity of the software

In this experiment, a crank-rocker linkage is tested. Based on the discussion in the previous chapter, it is obvious that other types of linkages can also be analyzed with a slight modification on the software composed in this chapter.

Refer to Fig. 4.1. It can be observed that if $\vec{V}_C$ is parallel to $\vec{V}_{CB}$, point c will be at infinity. The relative velocity method will fail. This is true when the follower aligns with the coupler, in other words, their slopes are equal. Fig. 4.16 shows three possible configurations of the linkages causing such kind of singularity. In case (a), the coupler is in the folded position with respect to the follower. In the other two cases, they are in extended position, in

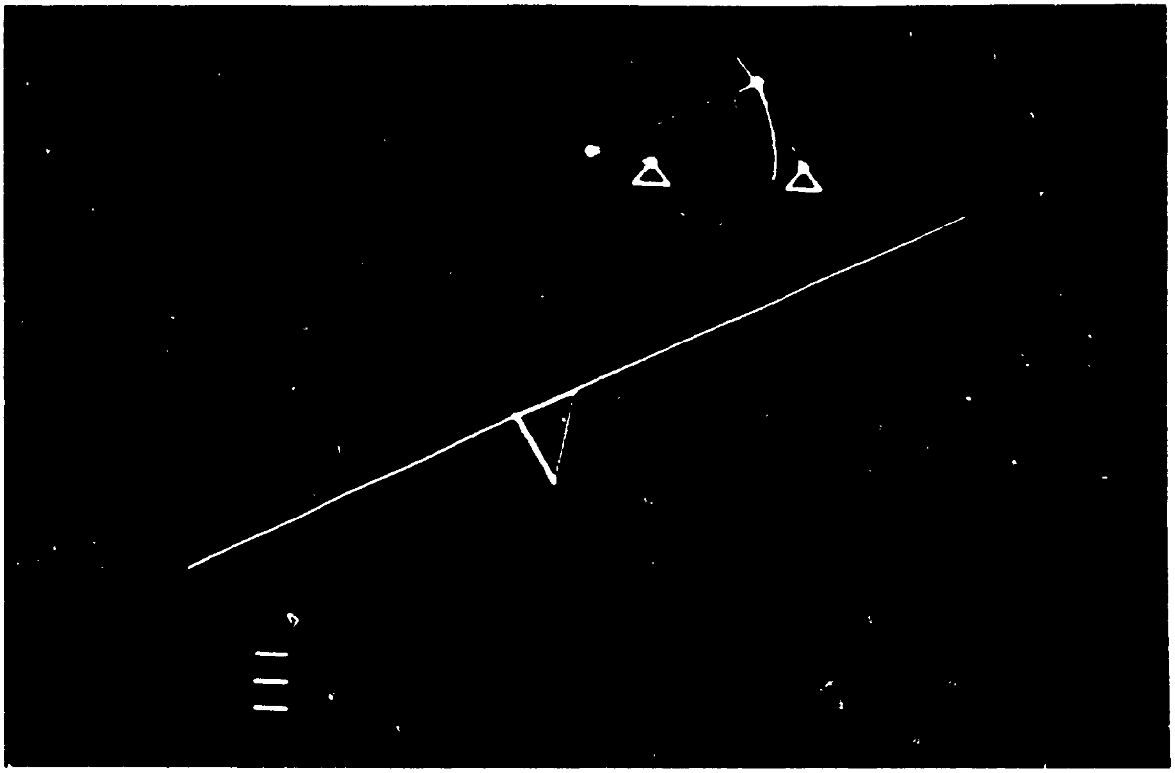


Fig. 4.10 Velocity polygon of an uncrossed four-bar linkage at position 1.

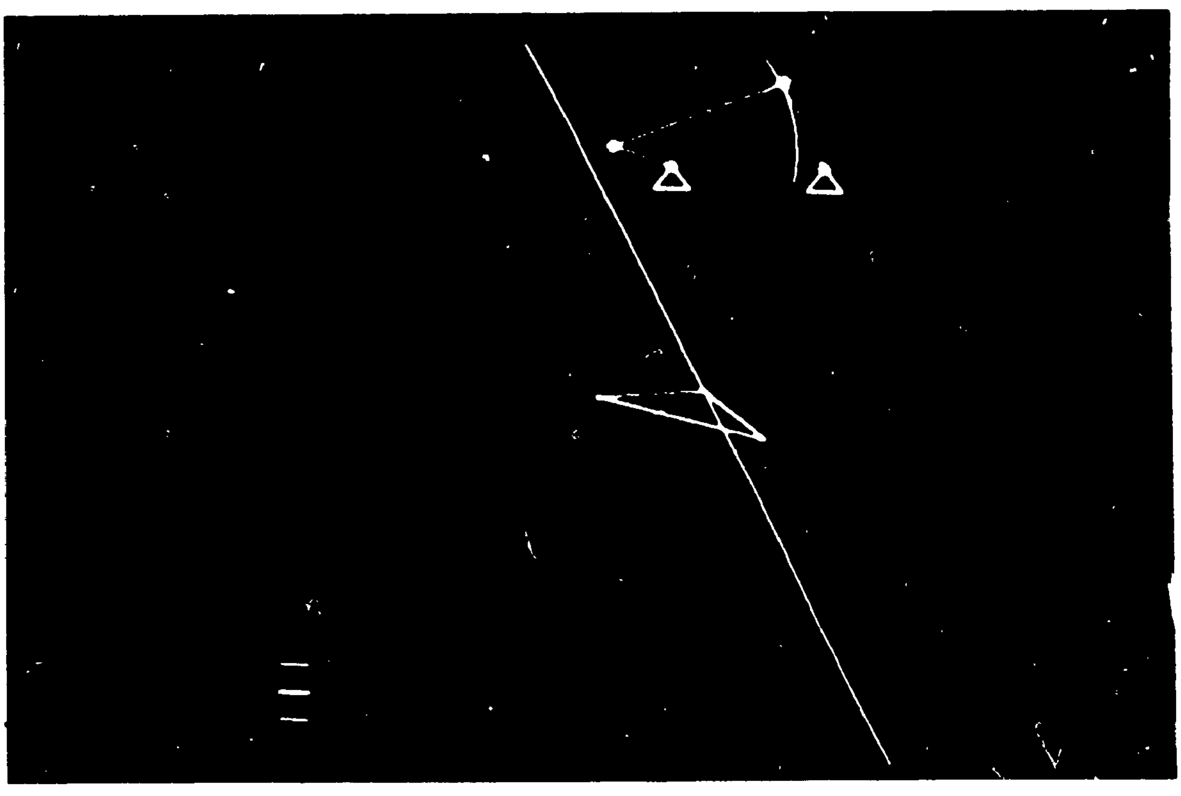


Fig. 4.11 Acceleration polygon of an uncrossed four-bar linkage at position 1.

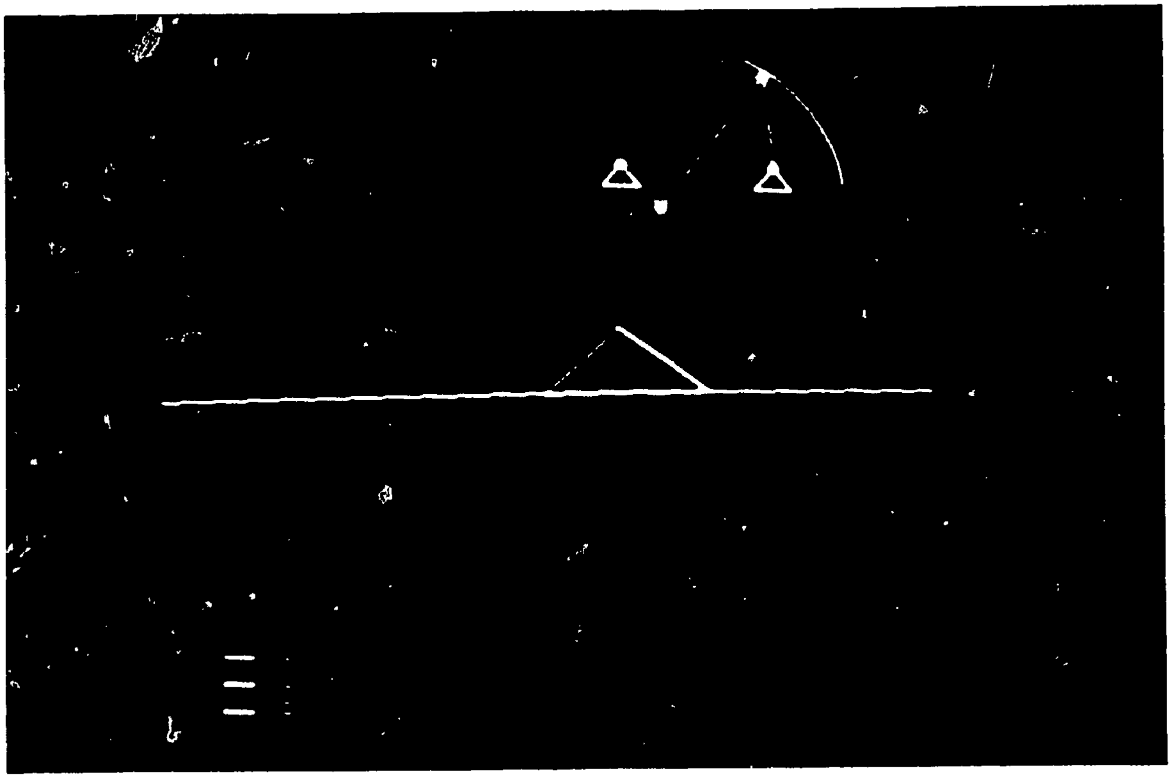


Fig. 4.12 Velocity polygon of an uncrossed four-bar linkage at position 2.

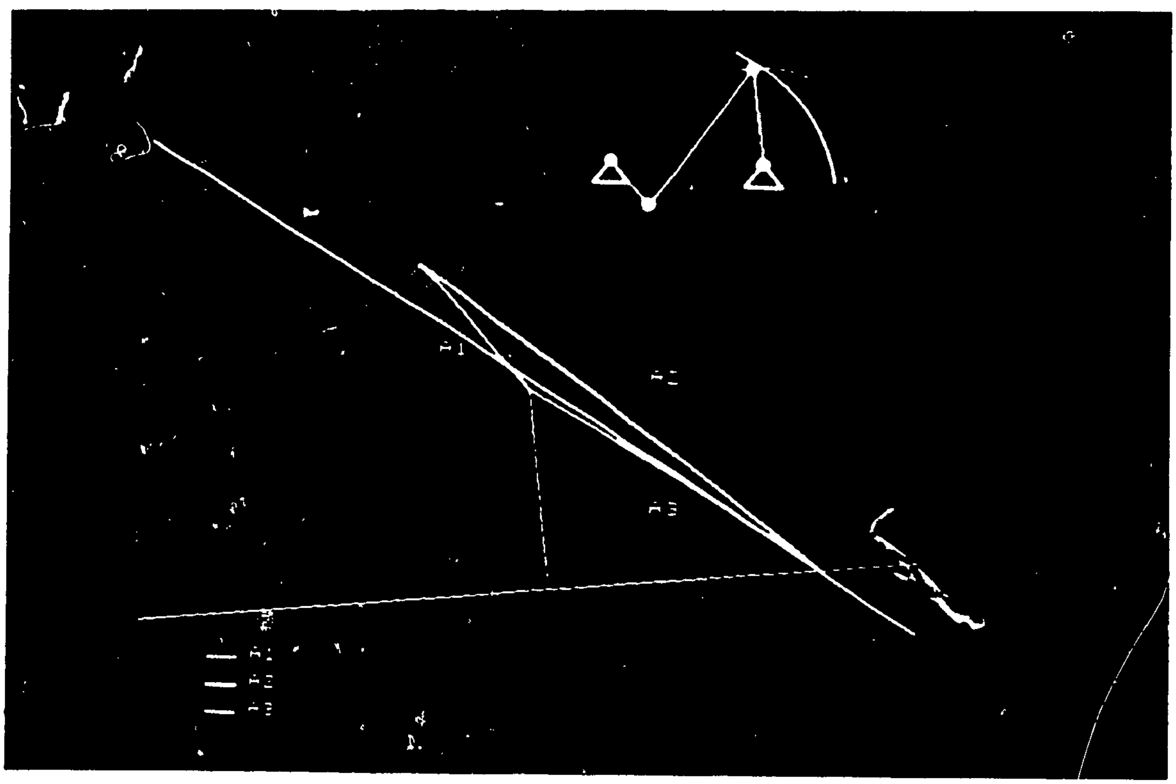


Fig. 4.13 Acceleration polygon of an uncrossed four-bar linkage at position 2.

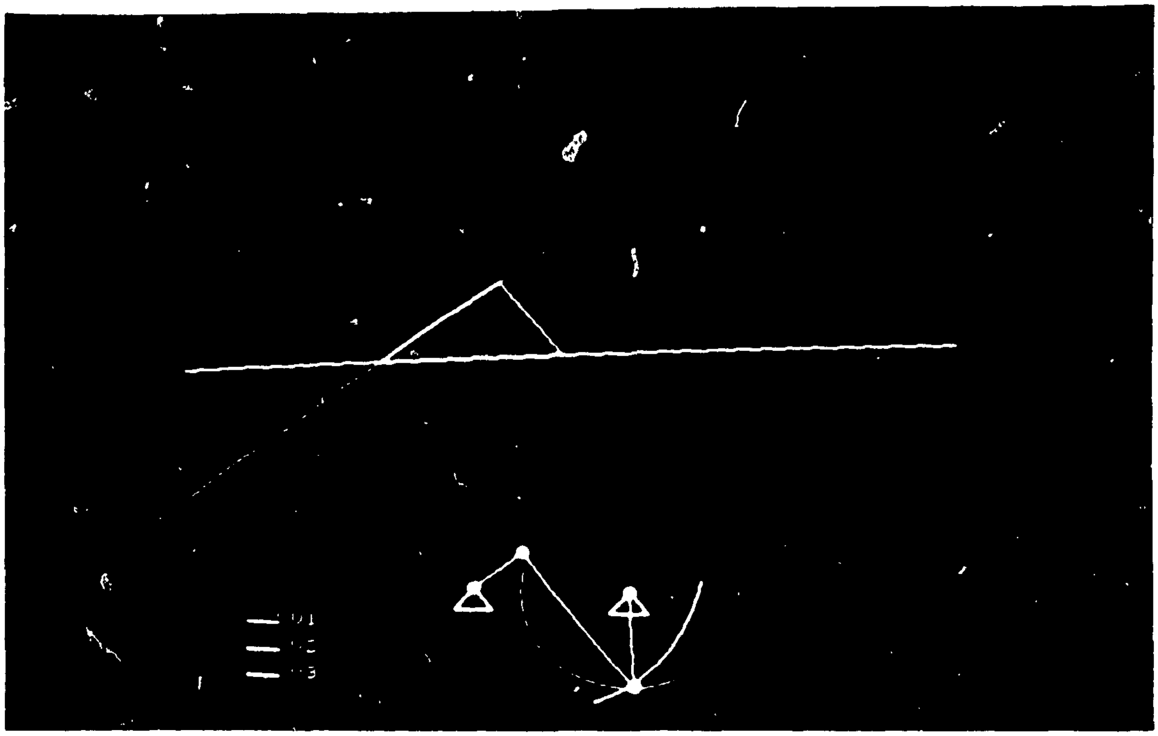


Fig. 4.14 Velocity polygon of a crossed four-bar linkage

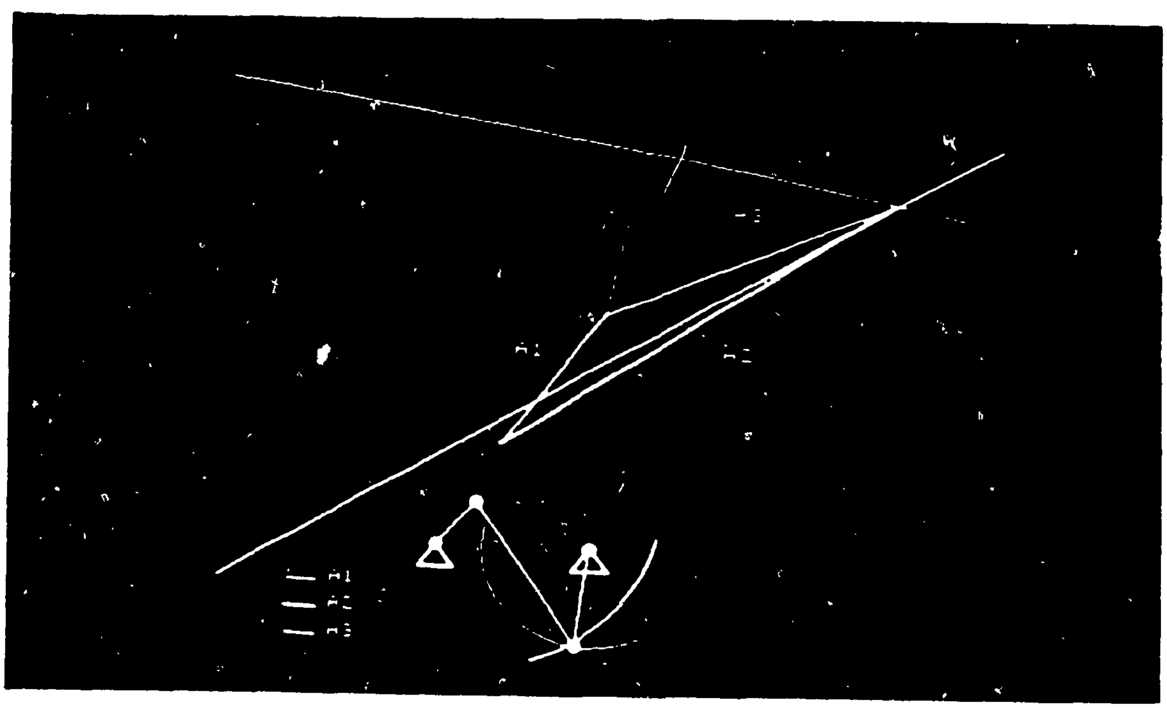
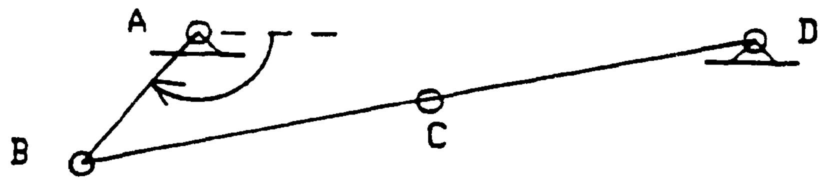
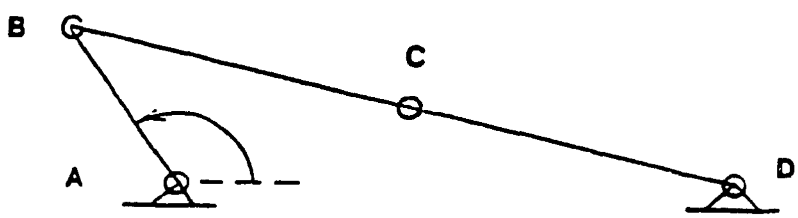


Fig. 4.15 Acceleration polygon of a crossed four-bar linkage

(a)



(b)



(c)

Fig. 4.16 Coupler aligns with the follower other words the driver reaches its extremums. Under all these circumstances, velocity of the follower is surely an infinity and even other methods will fail. Fig. 17 illustrates two configurations of a change-point linkage causing the singularity. In these cases, all links are aligned and have the same slope. The velocity of the follower is uncertain. Exactly the same phenomenon can be observed in case of acceleration analysis.

4.3.2 Results from the software

Refer to Fig. 4.2 to 4.9. As expected, curves depicting data from CH_TR are rather smooth, because no approximation has been made throughout the analysis. Obviously they represent the accurate results because the analysis is based upon presice analytic geometry of the mechanisms. On the other hand, curves drawn based on the data generated from CH_CR carry certain error due the application of CROS [17]. The maximum relative errors of both angular velocity and acceleration for all plotted data have been evaluated in program ERR and they are 4.7% and 6.0% respectively. Program ERR can be found in Appendix B. The error becomes quite severe when the input angular velocity ω_1 decreases. If all other conditions remain unchanged, smaller ω_1 means smaller velocity and acceleration polygons. Obviously smaller scale results higher error in graphical construction.

4.3.3 Effectiveness comparison between INTE and CROS

The central process unit (CPU) time for running CH_CR and CH_TR is recorded and compared. CH_TR runs approximately four times faster than

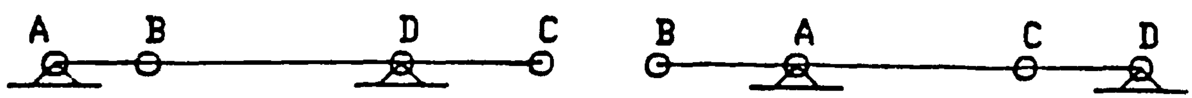


Fig. 4.17 Change-point four-bar linkage.

CH_CR if merely the simulation of the linkage movement is conducted. As the amount of computation for the velocity and acceleration analysis increases, the required time difference becomes greater. If complete kinematic analysis is performed at every increment of the independent angle, the difference is roughly fifteen times. The low speed of CH_CR is caused mainly by the usage of CROS. As indicated in section 4.1, CROS is called to search the intersection of m_1 and m_2. The two lines displayed on the screen are formed by pixels with pixel values. If the two lines are drawn in different pixel values, in other words, different colors, by searching pixel by pixel, CROS will be able to find the point where the pixel value changes which means the intersection of two lines. Very often, the point where CROS starts to search is a few hundreds of pixels away from the intersection point. Undoubtedly, large amount of computation time is needed.

As indicated in section 4.1, if m_1 and m_2 intersect outside the screen, CROS will have trouble to search the intersection. Unless some more calculation is carried out, it is impossible to anticipate where the intersection will fall on. Consequently, scaling and shifting are impossible. However, if INTE is used, the point is found by simple calculation. Therefore scaling and shifting can be done by knowing where the intersection is.

On the other hand, CROS is proved to be effective in dealing with nonlinear equations [17]. Instead of deriving expressions of the curves and solving them numerically, CROS can search their intersection(s) directly on a screen where they are displayed explicitly. In fact, the application of CROS in the position analysis of both the RR-RR-RR III and RRR-RRR IV components has substantially simplified the nonlinear problems and improved the efficiency as well. Therefore the application of CROS should be restricted to the nonlinear problems. And in the case of four-bar linkage, INTE should be used to solve the two linear equations representing m_1 and m_2. And it has been used in the analysis throughout chapter 3.

4.3.4 Advantages of the computer aided graphical method

With the CAG techniques, a number of graphical methods are implemented to analyze a four-bar linkage. Graphical methods such as the Chace's method and relative velocity and acceleration method are shown to be effective when incorporated with the Norpak graphics package. Graphic subroutines have been extensively applied to every part of the kinematic analysis including the simulation of the linkage movement and the velocity as well as acceleration polygon construction. Because the entire process of analysis can be displayed on the screen as the linkage is simulated, the relationships among the kinematic parameters and variables, such as the geometrical dimension of the linkage, the positions of the links and the linear velocities and accelerations of different points on the linkage, can be clearly visualized. The interactive programming techniques have further enhanced the controllability and efficiency of the software. Eventually a user can understand more easily the kinematic relationship of a linkage. And higher efficiency in linkage design can therefore be achieved.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

5.1 Conclusions

A variety of graphical methods have been proved to be suitable for automation of kinematic analysis of linkages. Based on the concept of basic component, typical graphical methods can well be incorporated with the modern CAG technology to form the computer aided graphical methods.

Three basic components, namely the RRR II, RR-RR-RR III and RRR-RRR IV component, are analyzed based on a wide variety of methods such as the Chace's method, method of assumed point, instantaneous center method, auxiliary point method, relative velocity and acceleration method and velocity difference method. With geometry especially analytic geometry as the chief mathematical tool, analysis is programmed into a series of computer subroutines to form independent units for the three basic components. Three linkages each consisting of one of the above mentioned components are also analyzed to demonstrate the application of the three units. The procedure of the implementation is shown to be straightforward and efficient. The Norpak graphic package loaded on a VAX/VMS system is employed to analyze a four-bar linkage. Two key subroutines, CROS and INTE, are applied to the position, velocity and acceleration analysis of the RRR II component. As a result, INTE which directly calculates the intersection of the two straight lines is shown to be more efficient in both programming and computation time. When INTE is used, no approximation is involved thus accurate results are obtained. In conclusion, INTE is more suitable for solving linear problems. But the application of CROS to the position analysis of two double-loop components indicates that it is powerful in dealing with nonlinear problems. Ti ghout the analysis of the four-bar linkage, visual contents are generated for a user to understand the analysis process and the corresponding geometrical relationships of the linkage.

In all the the analysis discussed thus far, the simplicity and effectiveness of the graphical methods are maintained without sacrificing the necessary accuracy. High efficiency has been attained in both software writing and computation time. The generated visual effect is versatile in mechanism design.

5.2 Recommendation for Future Work

Further future research may be necessary on the following aspects of graphical design of mechanisms:

i) It is necessary to establish the validity of the subroutines created in the form of detailed flow chart for the position, velocity and acceleration analysis of the RR-RR-RR III and RRR-RRR IV component, and then implement them on an appropriate computer system with adequate graphics capability.

ii) Also investigation must be carried out on the possibility and the procedure to develop a uniform and general computer aided graphical method for the kinematic analysis of different kinds of basic components instead of using several different methods.

iii) It is also necessary to analyze other required basic components in order to form a complete package for linkage analysis.

iv) An elastics and a dynamic analysis with graphical methods by means of CAG techniques to enhance the design package.

v) The software written for the four-bar linkage analysis is to be reorganized so that one program can both display the analysis process and generate necessary data as well.

vi) The visual effect may be enhanced by, for example, using windows to isolate different kinds of visual contents such as the linkage and polygons for better organization and less interference problems. Directions of all vectors in the polygons should also be illustrated.

REFERENCES

1. Barton, L. O., Mechanism Analysis, Marcel Dekker, 1984.
2. Liang, C. G. and P. S. Yuen, Computer Aided Linkage Design, Mechanical and Industrial Publishing, 1986.
3. Suh, C. H. and C. W. Radcliffe, Kinematics & Mechanisms Design, John Wiley and Son, 1978.
4. Shigley, J. E., Kinematic Analysis of Mechanisms, McGraw-Hill, 1969.
5. Uicker Jr., J. J., et al., 'An Iterative Method for the Displacement Analysis of Spatial Linkages', Trans. of ASME: Journal of Applied Mechanics, 31, 1964, p.309-314.
6. Wengert, R. E., 'A Simple Automatic Derivative Evaluation Program', Communication, ACM, 7 (8), 1964, p.463-464.
7. Chace, M. A., 'Vector Analysis of Linkages', Trans: of ASME: Journal of Engineering for Industry, 55 (3), August, 1963, p289-297.
8. Jalon, J. G. and M. A. Serna, 'Computer Method for Kinematic Analysis of Lower-Pair Mechanisms-I', MMT, 16 (5), 1981, p.543-550.
9. Rooney, G. T., 'A Constraint Approach to Displacement Analysis of Planar Linkages', Mechanism, September, 1972, p.60-63.
10. Kinzel, G. L. and C. Chang, 'The Analysis of Planar Linkages Using a Modular Approach', Mechanism and Machine Theory, 19 (1), 1984, p.165-172.
11. ElMaraghy, H. A. and W. R. Newcombe, 'Interactive Kinematic Analysis and Synthesis of Linkage Mechanisms', Proceeding of International Mini and Micro Computer, IEEE, Montreal, Canada,

November 11-18, 1977,p 179-182.

12. Barker, C. R., 'Analyzing Mechanisms with the IBM PC', Computers in Mechanical Engineering, July, 1983, p.37-44.

13. Smith, D. A., 'A Low Cost Graphics Simulation of Basic Kinematic Linkages', 1983 ASEE Annual Conference Proceedings, 1983, p.40-45.

14. Norton, R. L., 'Four-Bar and Geared Five-Bar Linkage Analysis Programs for the Apple Computer', Mechanism and Machine Theory, 20 (4), 1985, p.313-320.

15. Funabashi, H. 'A Study on Completely Computer-Assisted Kinematic Analysis of Planar link Mechanisms', Mechanism and Machine Theory, 21 (6), 1986, p.473-479.

16. Freudenstein F. and H. M. Beigi, 'On a Computationally Efficient Microcomputer Kinematic Analysis of the Basic Linkage Mechanisms', Mechanism and Machine Theory, 21 (6), 1986, p.467-472.

17. Fabrikant, V. I. and T. S. Sankar, 'An Efficient Graphical Method for CAD', Computer Aided Design, 17 (8), October 1985, p.369-373.

18. Shigley, J. E. and J. J. Uicker, Jr., Theory of Machines and Mechanisms, McGraw-Hill, 1980.

19. Paul, B., Kinematics and Dynamics of Planar Machinery, Prentice-Hall, January 1979.

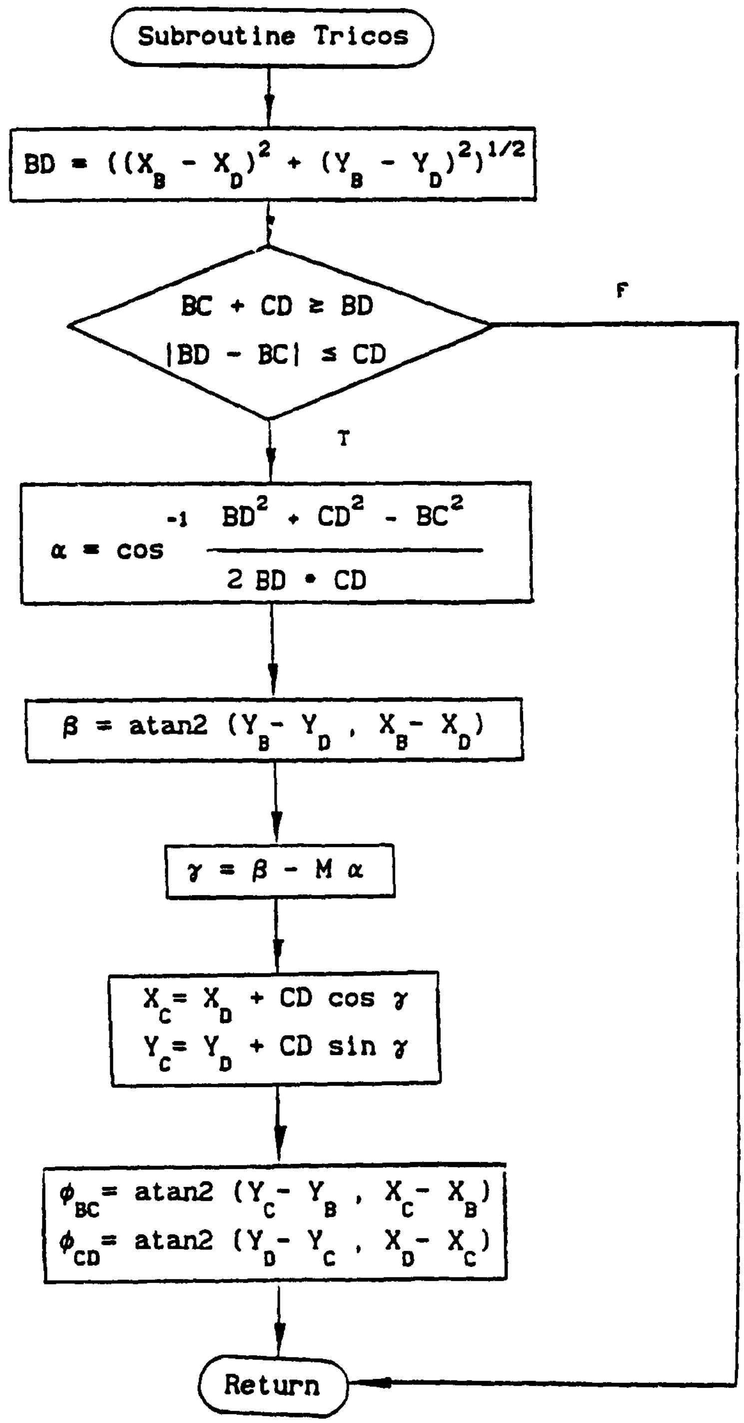
20. Huan, X. K. and W. W. Zheng, Principle of Machines, The People's Education, 1956.

21. VDP-2 Fortran Library, Docm: 85-04005-01, Norpak Ltd., Pakenham, Ontario, Canada.

APPENDIX A

FLOW CHARTS

A. 1 Elemental Subroutines of Graphical Methods



Subroutine Image

$$L_B = ((X_D - X_C)^2 + (Y_D - Y_C)^2)^{1/2}$$
$$L_C = ((X_D - X_B)^2 + (Y_D - Y_B)^2)^{1/2}$$
$$L_D = ((X_B - X_C)^2 + (Y_B - Y_C)^2)^{1/2}$$

$$\alpha_1 = \cos^{-1} ((L_C^2 + L_D^2 - L_B^2) / (2 L_C L_D))$$
$$\alpha_2 = \cos^{-1} ((L_B^2 + L_D^2 - L_C^2) / (2 L_C L_D))$$
$$\alpha_3 = 180^\circ - \alpha_1 - \alpha_2$$

$$L = ((X_b - X_d)^2 + (Y_b - Y_d)^2)^{1/2}$$

$$L' = (L \sin \alpha_2) / (\sin \alpha_1)$$
$$L'' = (L \sin \alpha_3) / (\sin \alpha_1)$$

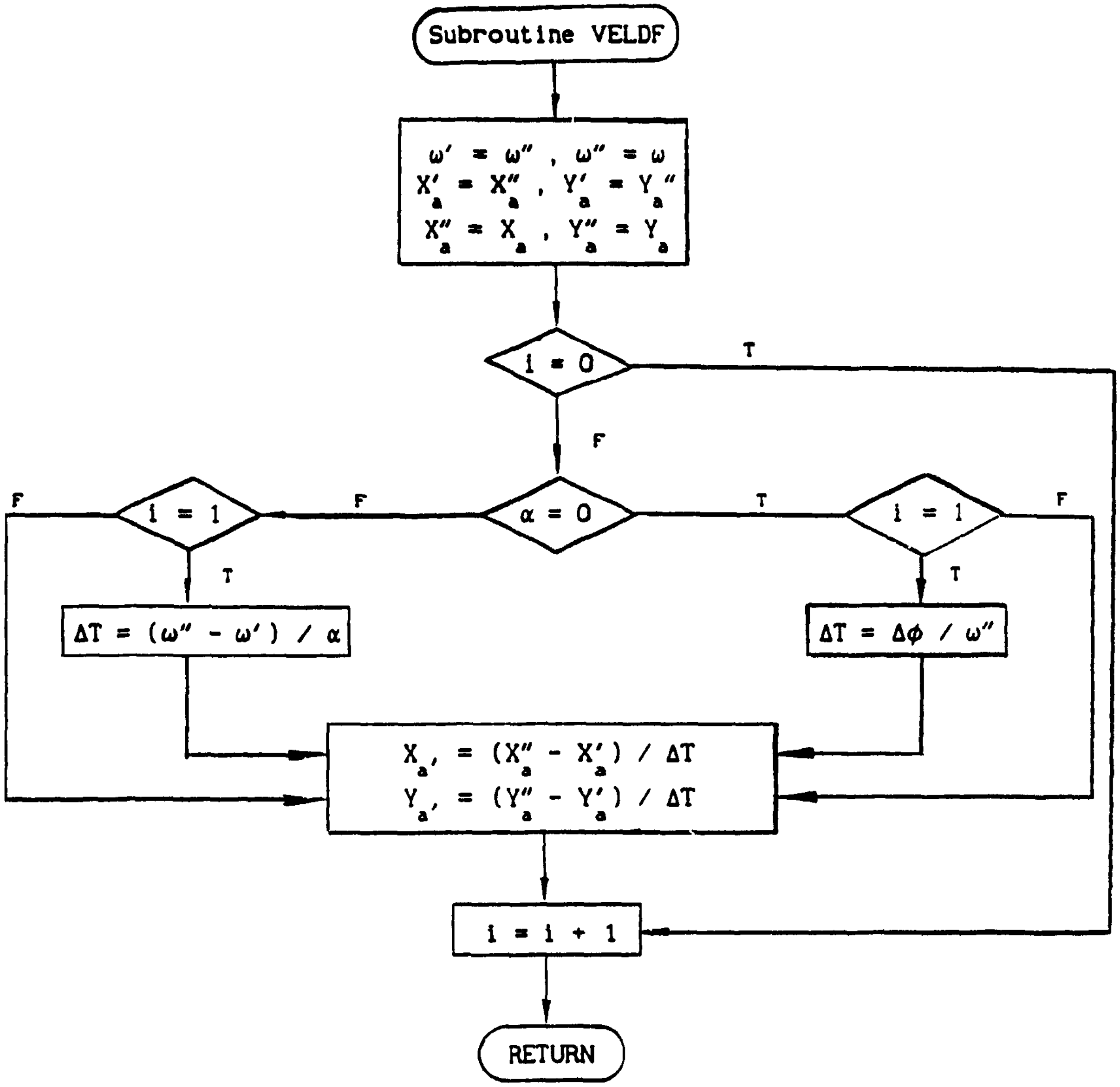
CALL TRICOS

$$a = ((X_c - X_0)^2 + (Y_c - Y_0)^2)^{1/2}$$
$$\phi_0 = \text{atan2}\,(Y_c - Y_0 , X_c - X_0)$$

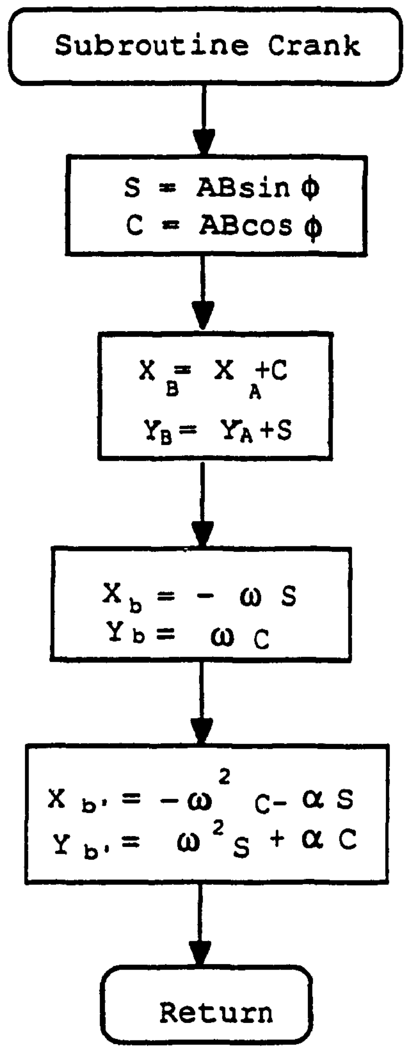
RETURN

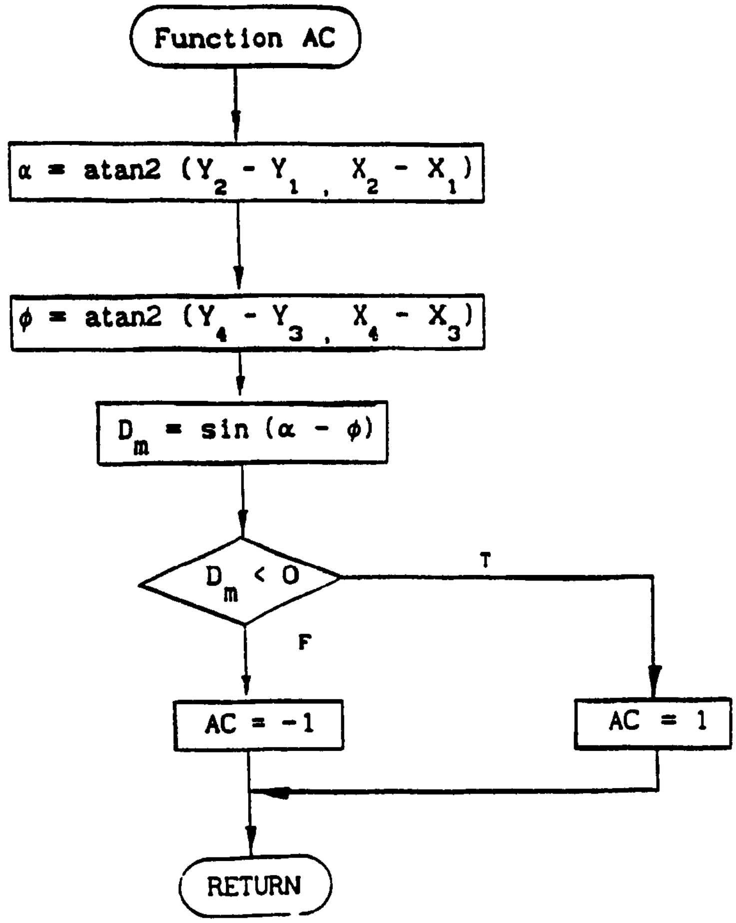
TRICOS

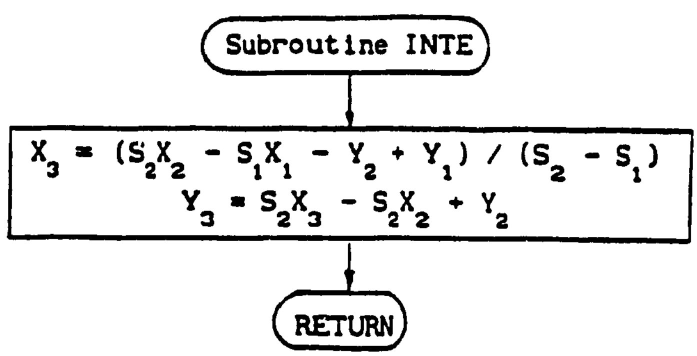
DUMMY ARGUMENT	X_B	Y_B	X_D	Y_D	M	BC	CD	X_C	Y_C	ϕ_{BC}	ϕ_{CD}
ACTUAL ARGUMENT	X_b	Y_b	X_d	Y_d	M	L'	L"	X_c	Y_c	ϕ_{bc}	ϕ_{cd}

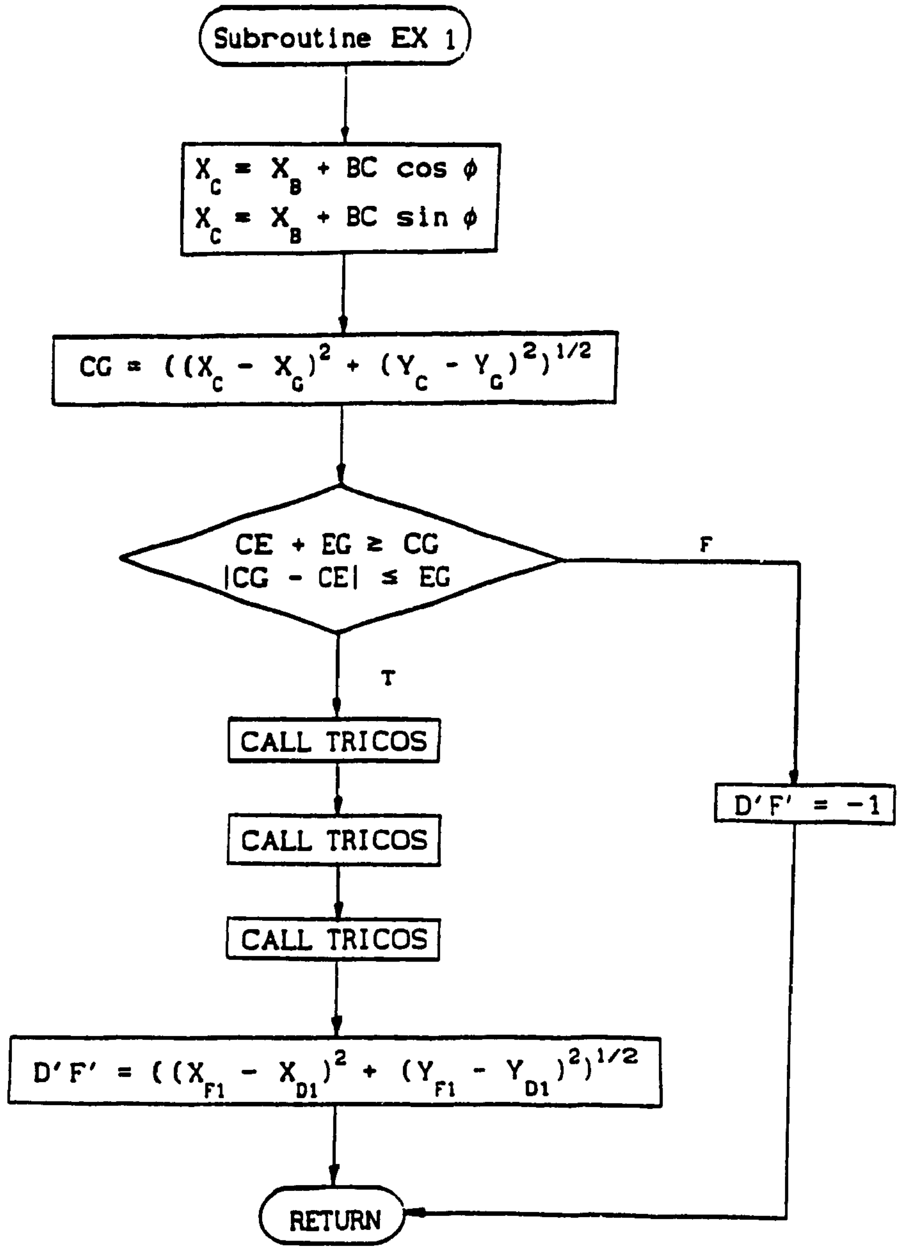

Subroutine VELDF
ω' = ω" , ω" = ω
X'a = X"a , Y'a = Y"a
X"a = Xa , Y"a = Ya
i = 0
T
F
α = 0
F
T
i = 1
i = 1
F
T
T
F
ΔT = (ω" - ω') / α
ΔT = Δφ / ω"
Xa' = (X"a - X'a) / ΔT
Ya' = (Y"a - Y'a) / ΔT
i = i + 1
RETURN

A. 2 Utility Subroutines for Analysis

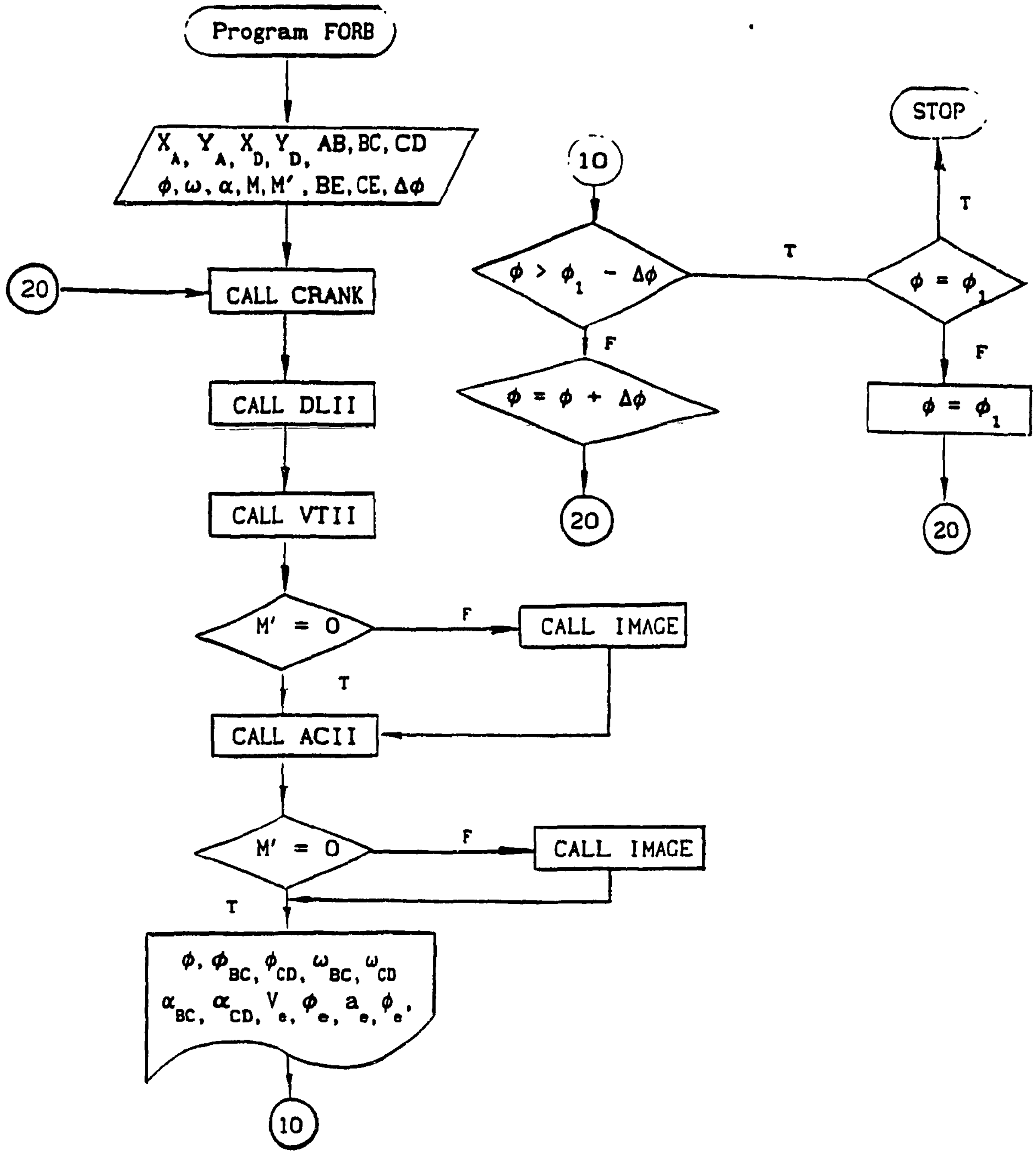



Function AC
α = atan2 (Y2 - Y1 , X2 - X1)
φ = atan2 (Y4 - Y3 , X4 - X3)
Dm = sin (α - φ)
Dm < 0
T
F
AC = -1
AC = 1
RETURN


Subroutine INTE
X3 = (S2X2 - S1X1 - Y2 + Y1) / (S2 - S1)
Y3 = S2X3 - S2X2 + Y2
RETURN


Subroutine EX 1
X_C = X_B + BC cos φ
X_C = X_B + BC sin φ
CG = ((X_C - X_G)^2 + (Y_C - Y_G)^2)^1/2
CE + EG ≥ CG
|CG - CE| ≤ EG
F
T
CALL TRICOS
CALL TRICOS
CALL TRICOS
D'F' = -1
D'F' = ((X_F1 - X_D1)^2 + (Y_F1 - Y_D1)^2)^1/2
RETURN

A. 3 Unit of RRR II Component



CRANK

DUMMY ARGUMENT	X_A	Y_A	ϕ	ω	α	AB	X_B	Y_B	X_b	Y_b	$X_{b'}$	$Y_{b'}$
ACTUAL ARGUMENT	X_A	Y_A	ϕ_O	ω	α	AB	X_B	Y_B	X_b	Y_b	$X_{b'}$	$Y_{b'}$

DLII

DUMMY ARGUMENT	X_B	Y_B	X_D	Y_D	BC	CD	BE	CE	M	M'	X_C	Y_C	X_E	Y_E	ϕ_{BC}	ϕ_{CD}	ϕ_{BE}
ACTUAL ARGUMENT	X_B	Y_B	X_D	Y_D	BC	CD	BE	CE	M	M'	X_C	Y_C	X_E	Y_E	ϕ_{BC}	ϕ_{CD}	ϕ_{BE}

IMAGE

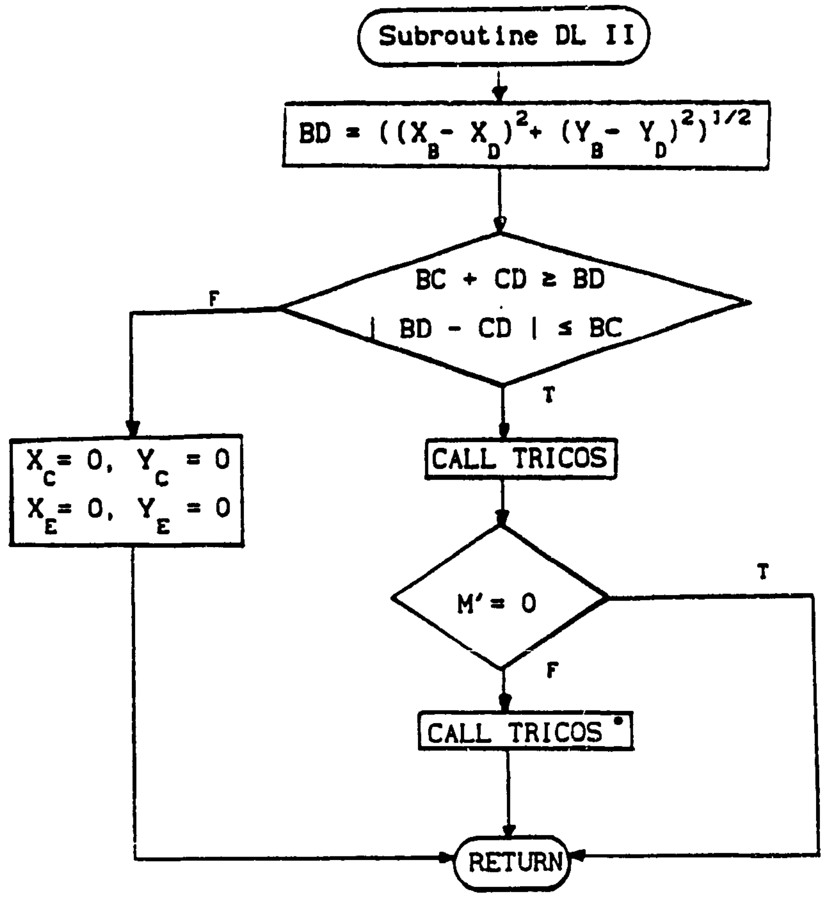
DUMMY ARGUMENT		X_B	Y_B	X_C	Y_C	X_D	Y_D	X_b	Y_b	X_d	Y_d
ACTUAL ARGUMENT	VELOCITY	X_B	Y_B	X_C	Y_C	X_E	Y_E	X_b	Y_b	X_c	Y_c
	ACCELERATION	X_B	Y_B	X_C	Y_C	X_E	Y_E	$X_{b'}$	$Y_{b'}$	$X_{c'}$	$Y_{c'}$
DUMMY ARGUMENT		M	X_c	Y_c	ϕ_{BC}	ϕ_{CD}	a	ϕ_a			
ACTUAL ARGUMENT	VELOCITY	M	X_e	Y_e	ϕ_1'	ϕ_2'	V_e	ϕ_e			
	ACCELERATION	M	$X_{e'}$	$Y_{e'}$	ϕ_1''	ϕ_2''	a_e	$\phi_{e'}$			

VTII

DUMMY ARGUMENT	X_B	Y_B	X_C	Y_C	X_D	Y_D	X_b	Y_b	X_d	Y_d	BC	CD	X_c	Y_c	ω_{BC}	ω_{DC}
ACTUAL ARGUMENT	X_B	Y_B	X_C	Y_C	X_D	Y_D	X_b	Y_b	X_d	Y_d	BC	CD	X_c	Y_c	ω_{BC}	ω_{DC}

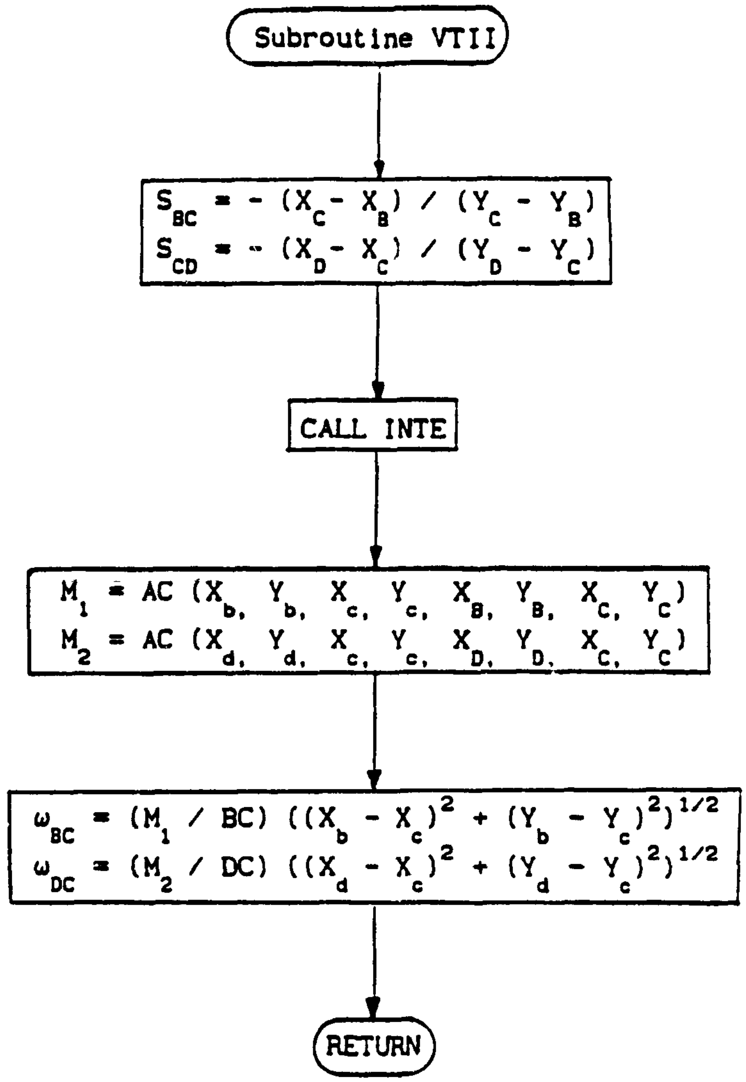
ACII

DUMMY ARGUMENT	X_B	Y_B	X_C	Y_C	X_D	Y_D	$X_{b'}$	$Y_{b'}$	$X_{d'}$	$Y_{d'}$	BC	CD	ω_{BC}	ω_{DC}
ACTUAL ARGUMENT	X_B	Y_B	X_C	Y_C	X_D	Y_D	$X_{b'}$	$Y_{b'}$	$X_{d'}$	$Y_{d'}$	BC	CD	ω_{BC}	ω_{DC}
DUMMY ARGUMENT	$X_{c'}$	$Y_{c'}$	α_{BC}	α_{CD}										
ACTUAL ARGUMENT	$X_{c'}$	$Y_{c'}$	α_{BC}	α_{CD}										



TRICOS

DUMMY ARGUMENT		X_B Y_B X_D Y_D M BC CD X_C Y_C $\emptyset_{BC}$ $\emptyset_{CD}$
ACTUAL ARGUMENT	WITHOUT *	X_B Y_B X_D Y_D M BC CD X_C Y_C $\emptyset_{BC}$ $\emptyset_{CD}$
	WITH *	X_B Y_B X_C Y_C M' BE CE X_E Y_E $\emptyset_{BE}$ $\emptyset_{CE}$



INTE

DUMMY ARGUMENT	X_1	Y_1	S_1	X_2	Y_2	S_2	X_3	Y_3
ACTUAL ARGUMENT	X_b	Y_b	S_{BC}	X_d	Y_d	S_{CD}	X_c	Y_c

Subroutine ALII

↓

$$a_{nCB} = \omega_{CB}^{2}\, CB$$
$$a_{nCD} = \omega_{CD}^{2}\, CD$$

↓

$$\phi_{CB} = \mathrm{atan2}\,(Y_B - Y_C\,,\; X_B - X_C)$$
$$\phi_{CD} = \mathrm{atan2}\,(Y_D - Y_C\,,\; X_D - X_C)$$

↓

$$X_{nB} = X_{b'} + a_{nCB} \cos \phi_{CB}$$
$$Y_{nB} = Y_{b'} + a_{nCB} \sin \phi_{CB}$$
$$X_{nD} = X_{d'} + a_{nCD} \cos \phi_{CD}$$
$$Y_{nD} = Y_{d'} + a_{nCD} \sin \phi_{CD}$$

↓

CALL INTE

↓

$$M_1 = AC\,(X_{nB}, Y_{nB}, X_{c'}, Y_{c'}, X_B, Y_B, X_C, Y_C)$$
$$M_2 = AC\,(X_{nD}, Y_{nD}, X_{c'}, Y_{c'}, X_D, Y_D, X_C, Y_C)$$

↓

$$\alpha_{BC} = (M_1 / CB)\,((X_{c'} - X_{nB})^2 + (Y_{c'} - Y_{nB})^2)^{1/2}$$
$$\alpha_{CD} = (M_2 / CD)\,((X_{c'} - X_{nD})^2 + (Y_{c'} - Y_{nD})^2)^{1/2}$$

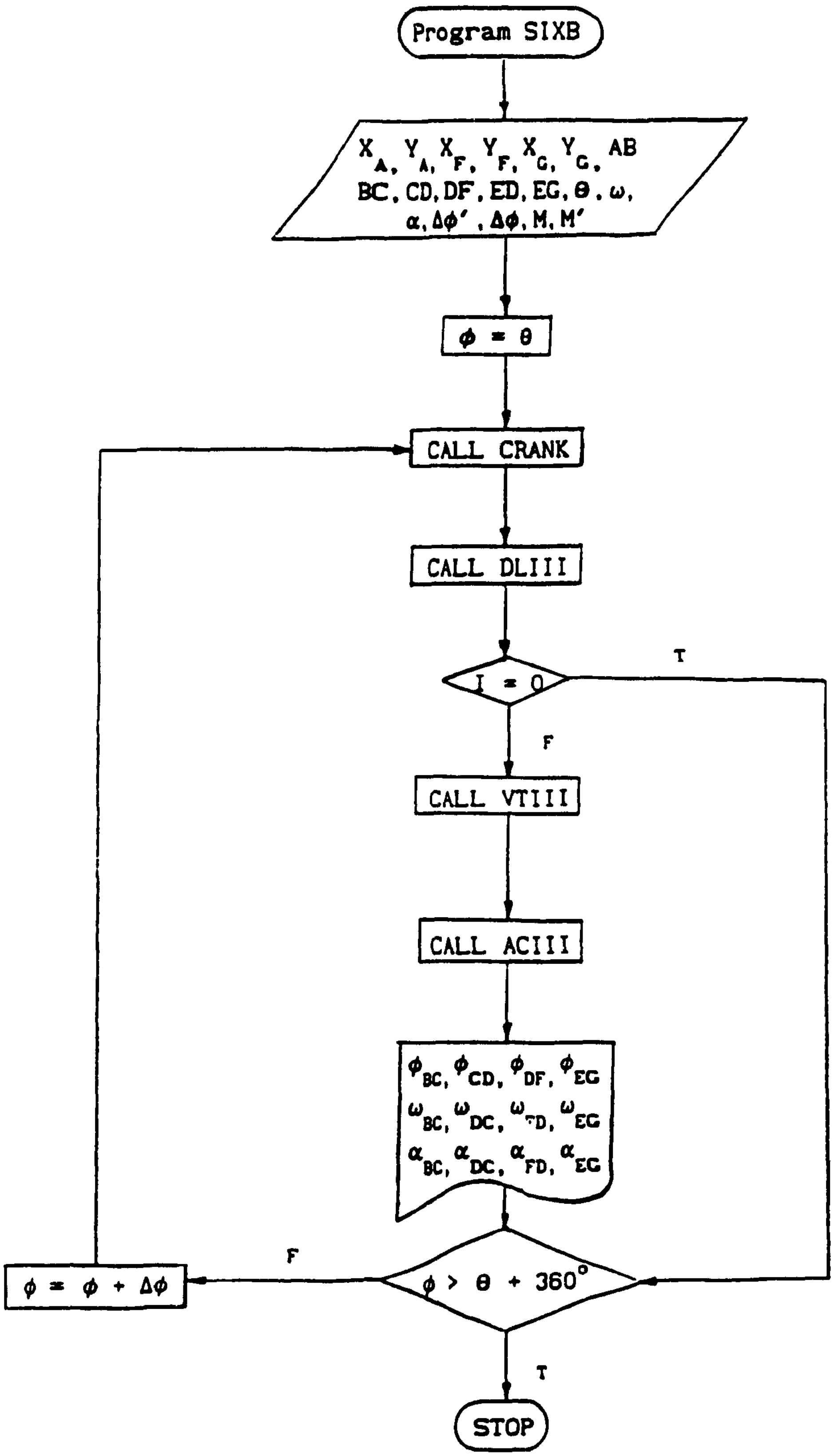
↓

RETURN

INTE

DUMMY ARGUMENT	X_1	Y_1	S_1	X_2	Y_2	S_2	X_3	Y_3
ACTUAL ARGUMENT	X_{nB}	Y_{nB}	$-\mathrm{ctan}\phi_{CB}$	X_{nD}	Y_{nD}	$-\mathrm{ctan}\phi_{CD}$	$X_{c'}$	$Y_{c'}$

A. 4 Unit of RR-RR-RR III Component



CRANK

DUMMY ARGUMENT	X_A Y_A ϕ ω α AB X_B Y_B X_b Y_b $X_{b'}$ $Y_{b'}$
ACTUAL ARGUMENT	X_A Y_A ϕ ω α AB X_B Y_B X_b Y_b $X_{b'}$ $Y_{b'}$

DLIII

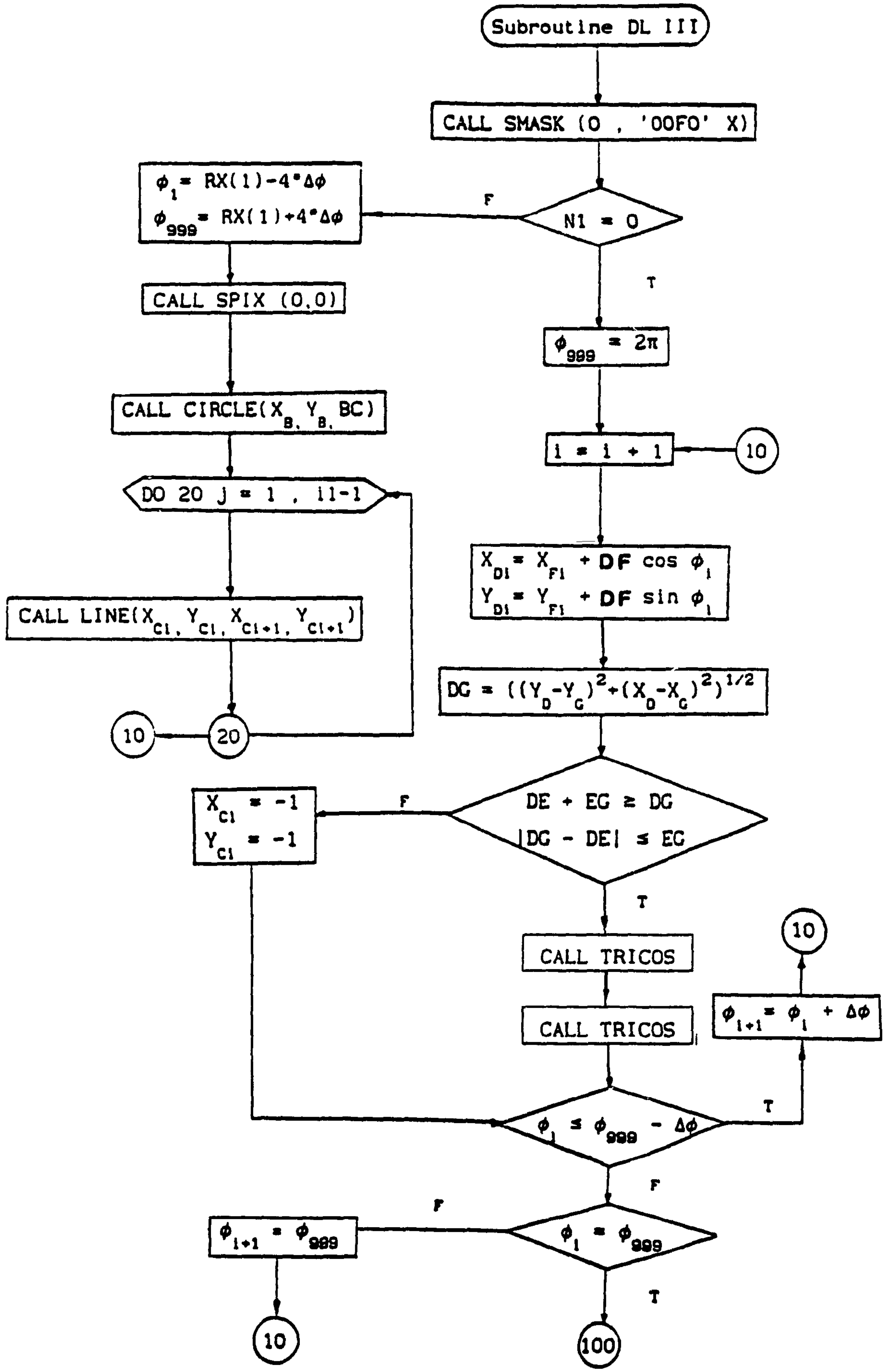
DUMMY ARGUMENT	X_B Y_B X_F Y_F X_G Y_G RX(1) RY(1) M M' $\Delta\phi$ BC CD
ACTUAL ARGUMENT	X_B Y_B X_F Y_F X_G Y_G RX(1) RY(1) M M' $\Delta\phi$ BC CD
DUMMY ARGUMENT	DF CE DE EG X_C Y_C X_D Y_D X_E Y_E ϕ_{BC} ϕ_{CD} ϕ_{DF} ϕ_{EG}
ACTUAL ARGUMENT	DF CE DE EG X_C Y_C X_D Y_D X_E Y_E ϕ_{BC} ϕ_{CD} ϕ_{DF} ϕ_{EG}

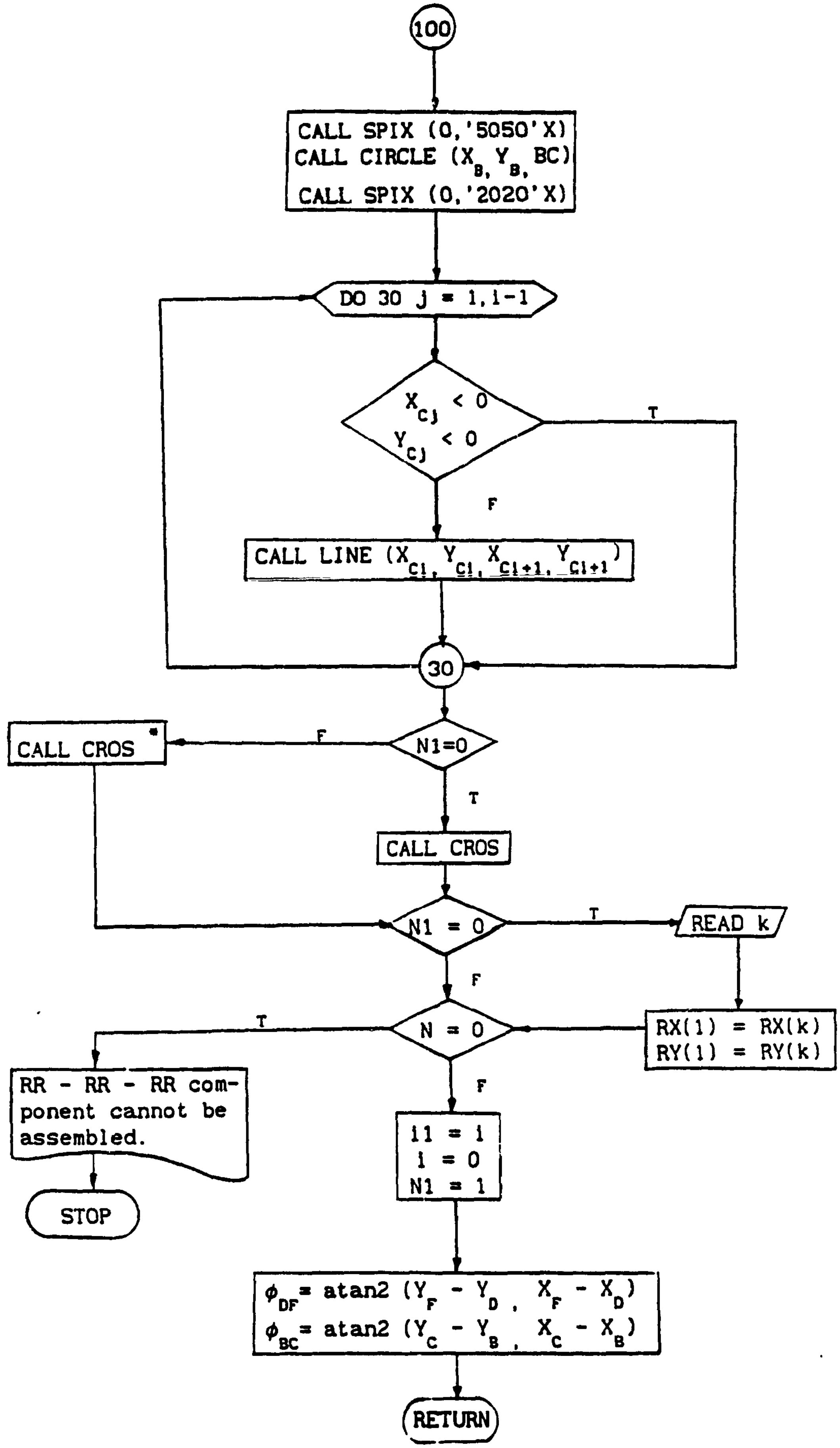
VTIII

DUMMY ARGUMENT	X_B Y_B X_C Y_C X_D Y_D X_E Y_E X_F Y_F X_G Y_G X_S Y_S X_b Y_b X_f Y_f
ACTUAL ARGUMENT	X_B Y_B X_C Y_C X_D Y_D X_E Y_E X_F Y_F X_G Y_G X_S Y_S X_b Y_b X_f Y_f
DUMMY ARGUMENT	X_g Y_g BC CE FD DE EG X_c Y_c X_d Y_d X_e Y_e ω_{BC} ω_{DC} ω_{FD} ω_{EG}
ACTUAL ARGUMENT	X_g Y_g BC CE FD DE EG X_c Y_c X_d Y_d X_e Y_e ω_{BC} ω_{DC} ω_{FD} ω_{EG}

ACIII

DUMMY ARGUMENT	X_B Y_B X_C Y_C X_D Y_D X_E Y_E X_F Y_F X_G Y_G X_S Y_S $X_{b'}$ $Y_{b'}$ $X_{f'}$
ACTUAL ARGUMENT	X_B Y_B X_C Y_C X_D Y_D X_E Y_E X_F Y_F X_G Y_G X_S Y_S $X_{b'}$ $Y_{b'}$ $X_{f'}$
DUMMY ARGUMENT	$Y_{f'}$ $X_{g'}$ $Y_{g'}$ ω_{BC} ω_{ED} ω_{FD} ω_{EG} $X_{c'}$ $Y_{c'}$ $X_{d'}$ $Y_{d'}$ $X_{e'}$ $Y_{e'}$
ACTUAL ARGUMENT	$Y_{f'}$ $X_{g'}$ $Y_{g'}$ ω_{BC} ω_{ED} ω_{FD} ω_{EG} $X_{c'}$ $Y_{c'}$ $X_{d'}$ $Y_{d'}$ $X_{e'}$ $Y_{e'}$
DUMMY ARGUMENT	α_{BC} α_{DC} α_{EG} α_{FD}
ACTUAL ARGUMENT	α_{BC} α_{DC} α_{EG} α_{FD}


Subroutine DL III
CALL SMASK (0 , '00F0' X)
N1 = 0
F
T
ϕ_1 = RX(1)-4*Δϕ
ϕ_{999} = RX(1)+4*Δϕ
CALL SPIX (0,0)
CALL CIRCLE(X_B, Y_B, BC)
DO 20 j = 1 , i1-1
CALL LINE(X_{Ci}, Y_{Ci}, X_{Ci+1}, Y_{Ci+1})
10
20
ϕ_{999} = 2π
i = i + 1
10
X_{Di} = X_{Fi} + DF cos ϕ_i
Y_{Di} = Y_{Fi} + DF sin ϕ_i
DG = ((Y_D-Y_G)^2+(X_D-X_G)^2)$^{1/2}$
DE + EG ≥ DG
|DG - DE| ≤ EG
F
T
X_{Ci} = -1
Y_{Ci} = -1
CALL TRICOS
CALL TRICOS
10
ϕ_{i+1} = ϕ_i + Δϕ
ϕ_i ≤ ϕ_{999} - Δϕ
T
F
ϕ_i = ϕ_{999}
F
T
ϕ_{i+1} = ϕ_{999}
10
100

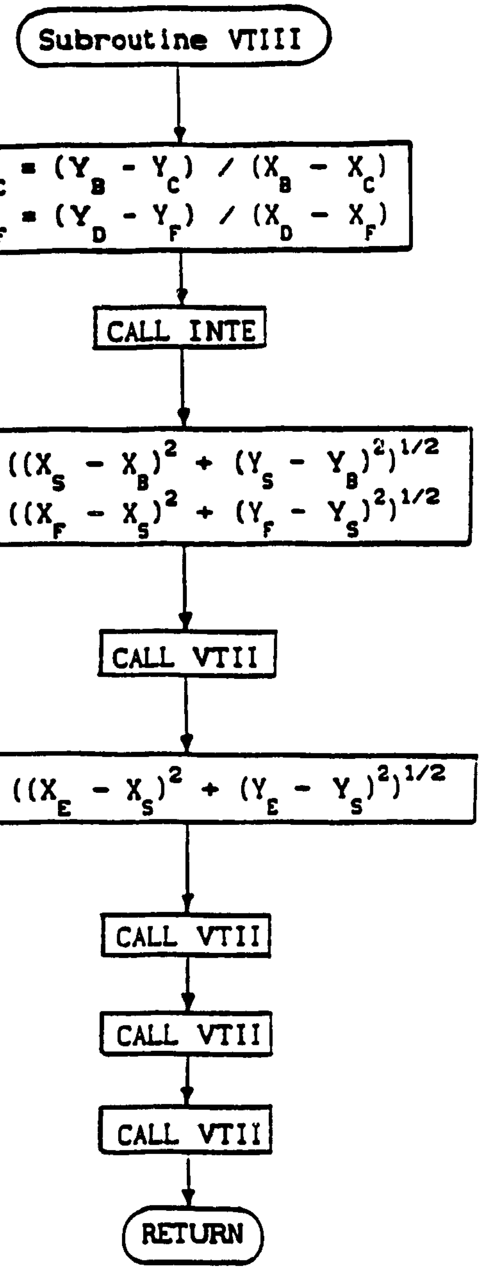

100
CALL SPIX (0,'5050'X)
CALL CIRCLE (X_B, Y_B, BC)
CALL SPIX (0,'2020'X)
DO 30 j = 1,i-1
X_Cj < 0
Y_Cj < 0
T
F
CALL LINE (X_Ci, Y_Ci, X_Ci+1, Y_Ci+1)
30
N1=0
F
CALL CROS*
T
CALL CROS
N1 = 0
T
READ k
F
N = 0
T
RX(1) = RX(k)
RY(1) = RY(k)
RR - RR - RR component cannot be assembled.
STOP
F
i1 = i
i = 0
N1 = 1
φ_DF = atan2 (Y_F - Y_D , X_F - X_D)
φ_BC = atan2 (Y_C - Y_B , X_C - X_B)
RETURN

CROS

DUMMY ARGUMENT		X	Y	RX	RY	N
ACTUAL ARGUMENT	WITH *	X_{C1}	Y_{C1}	RX	RY	N
	WITHOUT *	X_B+BC	Y_B+BC	RX	RY	N

TRICOS

DUMMY ARGUMENT		X_B	Y_B	X_D	Y_D	M	BC	CD	X_C	Y_C	ϕ_{BC}	ϕ_{CD}
ACTUAL ARGUMENT	POINT E	X_D	Y_D	X_G	Y_G	M	DE	EG	X_E	Y_E	ϕ_{DE}	ϕ_{EG}
	POINT C	X_D	Y_D	X_E	Y_E	M'	CD	EC	X_{C1}	Y_{C1}	ϕ_{CD}	ϕ_{EC}



VTII

DUMMY ARGUMENT		X_B	Y_B	X_C	Y_C	X_D	Y_D	X_b	Y_b	X_d	Y_d	BC	CD	X_c	Y_c	ω_{BC}	ω_{CD}
ACTUAL ARGUMENT	POINT S	X_B	Y_B	X_S	Y_S	X_F	Y_F	X_b	Y_b	X_f	Y_f	BS	SF	X_s	Y_s	0	0
	POINT E	X_S	Y_S	X_E	Y_E	X_G	Y_G	X_s	Y_s	X_g	Y_g	SE	EG	X_e	Y_e	0	ω_{EG}
	POINT C	X_B	Y_B	X_C	Y_C	X_E	Y_E	X_b	Y_b	X_e	Y_e	BC	CE	X_c	Y_c	ω_{BC}	ω_{CE}
	POINT D	X_F	Y_F	X_D	Y_D	X_E	Y_E	X_f	Y_f	X_e	Y_e	FD	DE	X_d	Y_d	ω_{FD}	ω_{DE}

INTE

DUMMY ARGUMENT	X_1	Y_1	S_1	X_2	Y_2	S_2	X_3	Y_3
ACTUAL ARGUMENT	X_B	Y_B	S_{BC}	X_F	Y_F	S_{DF}	X_S	Y_S

Subroutine ACIII

$$CS = ((X_C - X_S)^2 + (Y_C - Y_S)^2)^{1/2}$$
$$SD = ((X_S - X_D)^2 + (Y_S - Y_D)^2)^{1/2}$$
$$SE = ((X_S - X_E)^2 + (Y_S - Y_E)^2)^{1/2}$$

$$a_{nSB} = \omega_{BC}^2\, BC + \omega_{ED}^2\, CS$$
$$a_{nSF} = \omega_{FD}^2\, DF + \omega_{ED}^2\, SD$$

$$\psi_{SB} = \text{atan2}\,(Y_B - Y_S\ ,\ X_B - X_S)$$
$$\psi_{SF} = \text{atan2}\,(Y_F - Y_S\ ,\ X_F - X_S)$$

$$X_{nB} = X_{b'} + a_{nSB} \cos \psi_{SB}$$
$$X_{nB} = X_{b'} + a_{nSB} \sin \psi_{SB}$$
$$X_{nF} = X_{f'} + a_{nSF} \cos \psi_{SF}$$
$$X_{nF} = X_{f'} + a_{nSF} \sin \psi_{SF}$$

CALL INTE

CALL ACII

CALL ACII

CALL ACII

RETURN

INTE

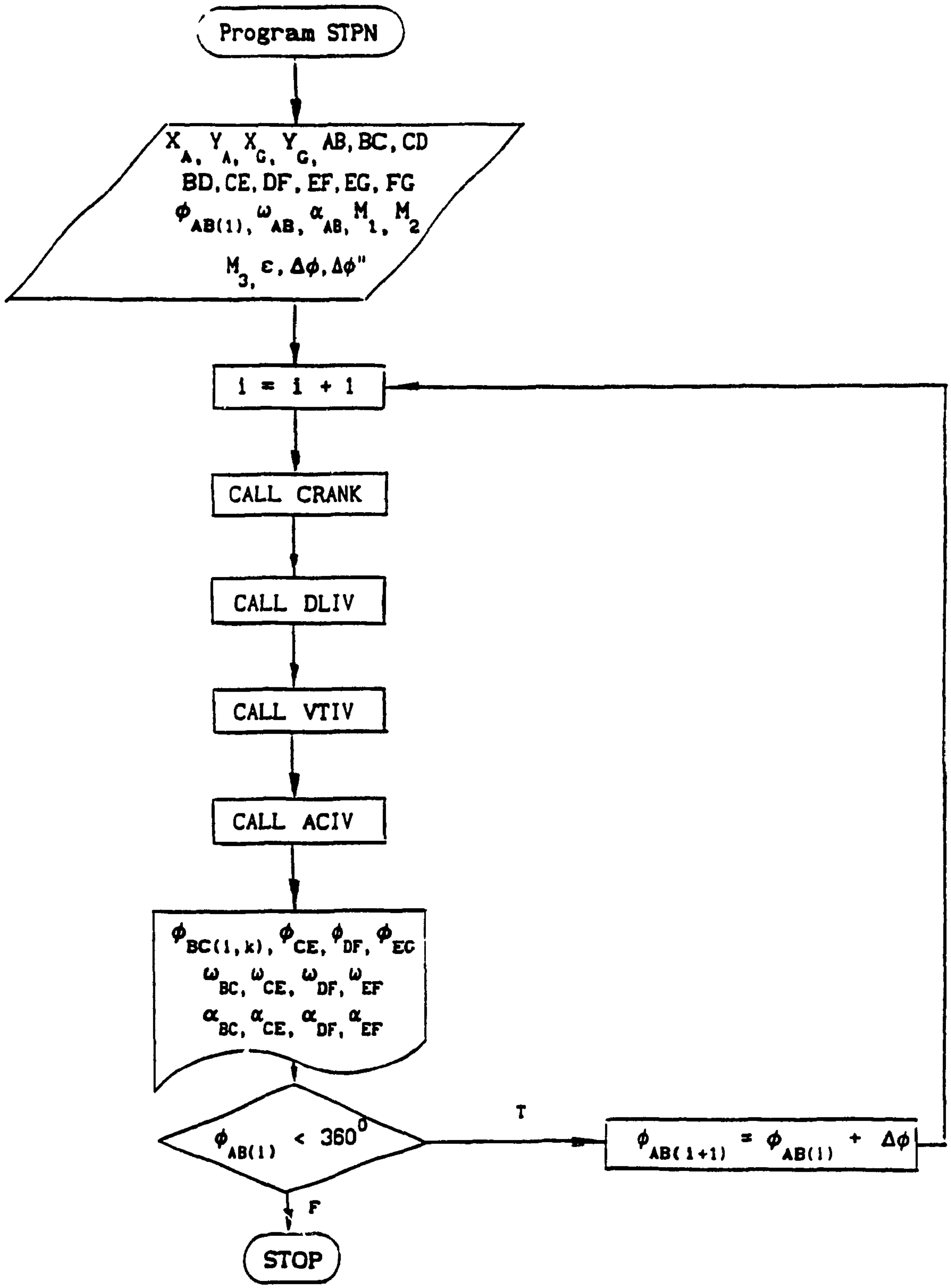
DUMMY ARGUMENT	X_1	Y_1	S_1	X_2	Y_2	S_2	X_3	Y_3
ACTUAL ARGUMENT	X_{nB}	Y_{nB}	$-ctan\ \psi_{SB}$	X_{nF}	Y_{nF}	$-ctan\ \psi_{SF}$	$X_{s'}$	$Y_{s'}$

ACII

DUMMY ARGUMENT		X_B	Y_B	X_C	Y_C	X_D	Y_D	$X_{b'}$	$Y_{b'}$	$X_{d'}$	$Y_{d'}$	BC	CD	ω_{BC}	ω_{DC}
ACTUAL	POINT E	X_S	Y_S	X_E	Y_E	X_G	Y_G	$X_{s'}$	$Y_{s'}$	$X_{g'}$	$Y_{g'}$	SE	EG	ω_{CE}	ω_{EG}
	POINT C	X_B	Y_B	X_C	Y_C	X_E	Y_E	$X_{b'}$	$Y_{b'}$	$X_{e'}$	$Y_{e'}$	BC	CE	ω_{BC}	ω_{CE}
ARGUMENT	POINT D	X_F	Y_F	X_D	Y_D	X_E	Y_E	$X_{f'}$	$Y_{f'}$	$X_{e'}$	$Y_{e'}$	FD	DE	ω_{FD}	ω_{DE}

DUMMY ARGUMENT		$X_{c'}$	$Y_{c'}$	α_{BC}	α_{CD}
ACTUAL	POINT E	$X_{e'}$	$Y_{e'}$	α_{CE}	α_{EG}
	POINT C	$X_{c'}$	$Y_{c'}$	α_{BC}	α_{CE}
ARGUMENT	POINT D	$X_{d'}$	$Y_{d'}$	α_{FD}	α_{DE}

A. 5 Unit of RRR-RRR IV Component



CRANK

DUMMY ARGUMENT	X_A Y_A ϕ ω α AB X_B Y_B X_b Y_b $X_{b'}$ $Y_{b'}$
ACTUAL ARGUMENT	X_A Y_A $\phi_{AB(1)}$ ω_{AB} α_{AB} AB X_B Y_B X_b Y_b $X_{b'}$ $Y_{b'}$

DLIV

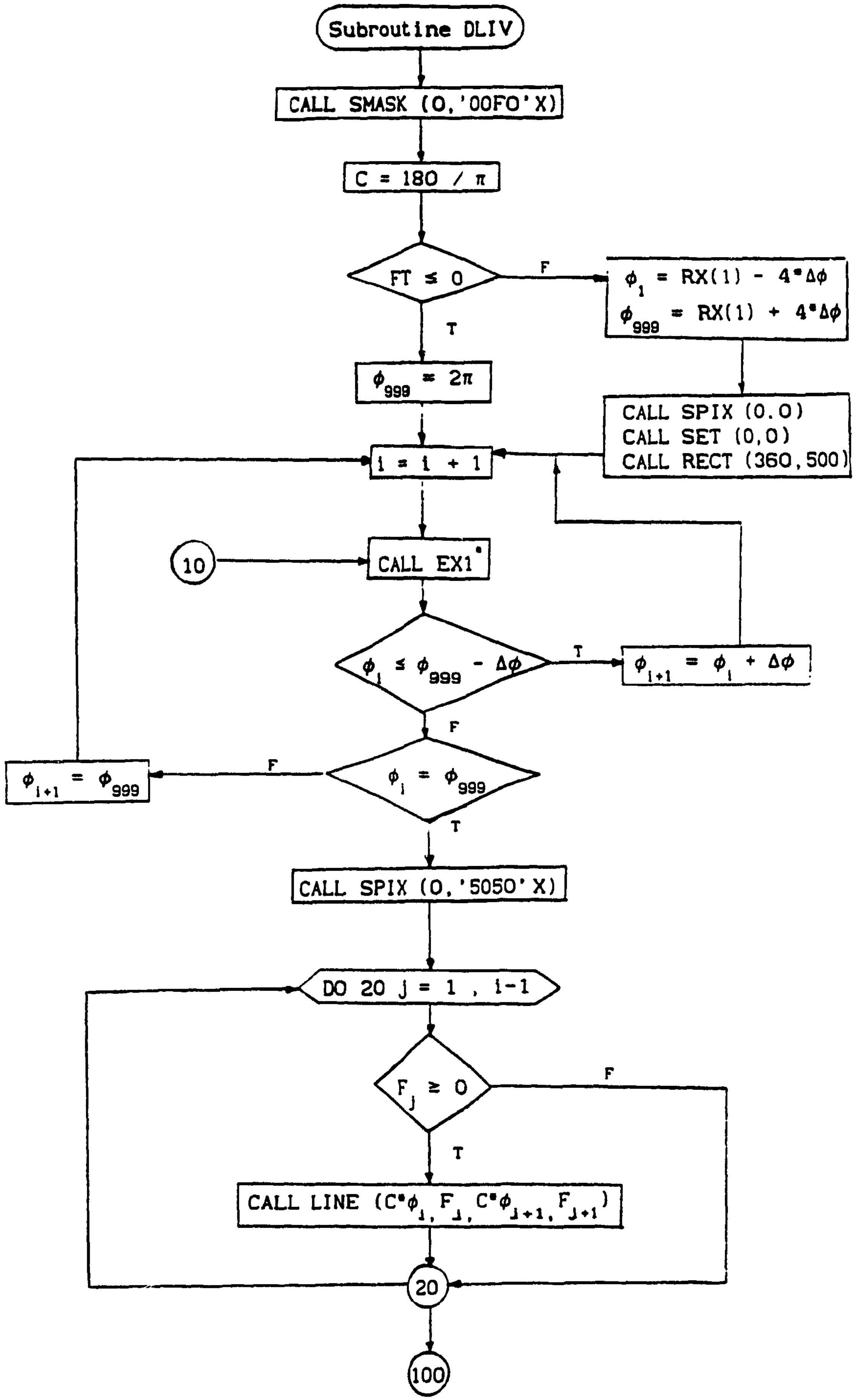
DUMMY ARGUMENT	X_B Y_B X_G Y_G BC CD BD CE EF EG FG DF ε M_1 M_2 M_3
ACTUAL ARGUMENT	X_B Y_B X_G Y_G BC CD BD CE EF EG FG DF ε M_1 M_2 M_3
DUMMY ARGUMENT	ϕ_1 $\Delta\phi$ X_C Y_C X_D Y_D X_E Y_E X_F Y_F ϕ_{CE} ϕ_{DF} ϕ_{EG}
ACTUAL ARGUMENT	ϕ_{AB} $\Delta\phi$ X_C Y_C X_D Y_D X_E Y_E X_F Y_F ϕ_{CE} ϕ_{DF} ϕ_{EG}

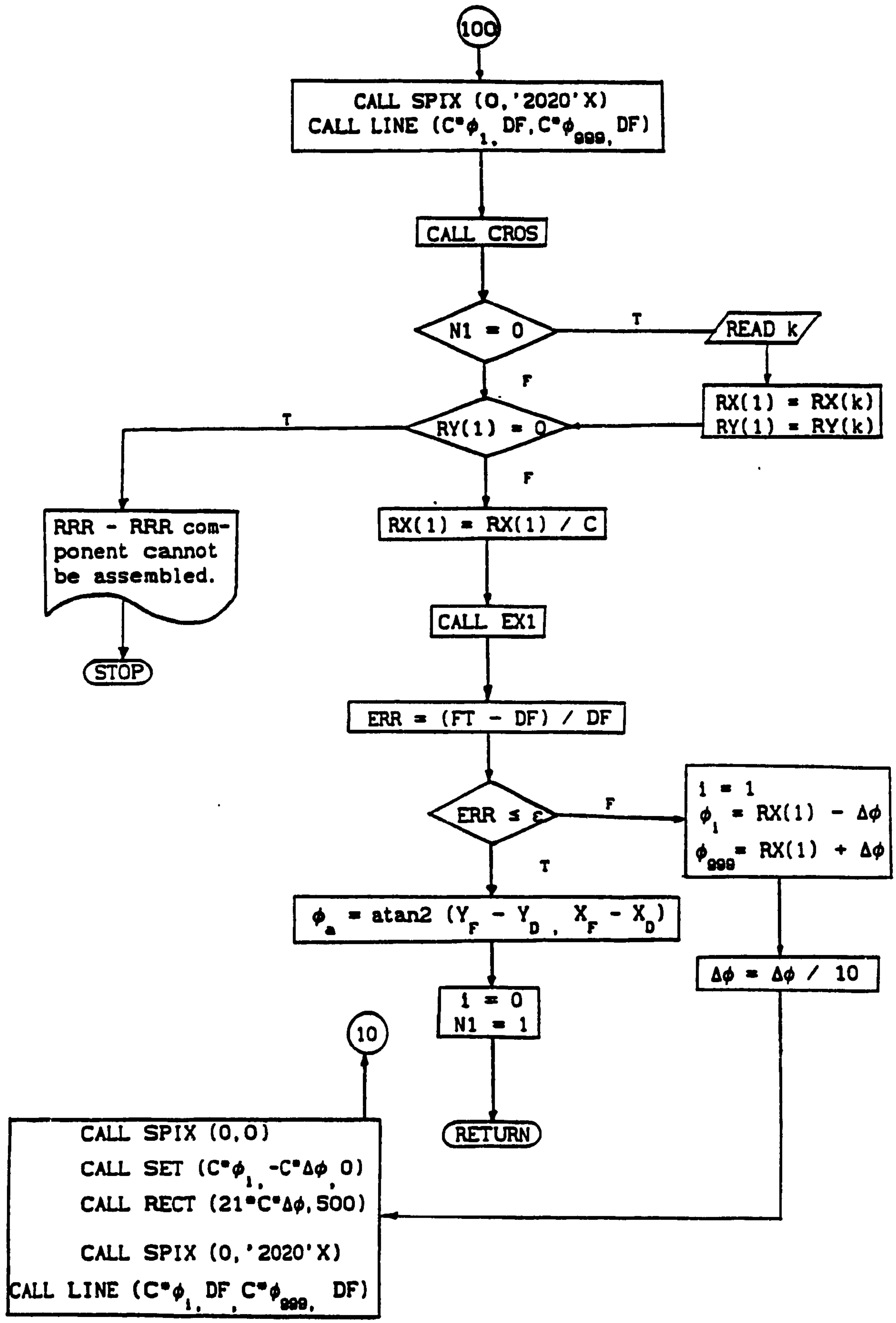
VTIV

DUMMY ARGUMENT	X_B Y_B X_C Y_C X_D Y_D X_E Y_E X_F Y_F X_G Y_G X_b Y_b X_g Y_g BC
ACTUAL ARGUMENT	X_B Y_B X_C Y_C X_D Y_D X_E Y_E X_F Y_F X_G Y_G X_b Y_b X_g Y_g BC
DUMMY ARGUMENT	CD CE DF X_c Y_c X_d Y_d X_e Y_e X_f Y_f ω_{BC} ω_{EF} ω_{CE} ω_{DF}
ACTUAL ARGUMENT	CD CE DF X_c Y_c X_d Y_d X_e Y_e X_f Y_f ω_{BC} ω_{EF} ω_{CE} ω_{DF}

ACIV

DUMMY ARGUMENT	BC CE DF EF X_c Y_c X_d Y_d X_e Y_e X_f Y_f $X_{b'}$ $Y_{b'}$ $\Delta\phi$ ω
ACTUAL ARGUMENT	BC CE DF EF X_c Y_c X_d Y_d X_e Y_e X_f Y_f $X_{b'}$ $Y_{b'}$ $\Delta\phi$ ω_{AB}
DUMMY ARGUMENT	α $X_{c'}$ $Y_{c'}$ $X_{d'}$ $Y_{d'}$ $X_{e'}$ $Y_{e'}$ $X_{f'}$ $Y_{f'}$ α_{BC} α_{CE} α_{DF} α_{EF}
ACTUAL ARGUMENT	α_{AB} $X_{c'}$ $Y_{c'}$ $X_{d'}$ $Y_{d'}$ $X_{e'}$ $Y_{e'}$ $X_{f'}$ $Y_{f'}$ α_{BC} α_{CE} α_{DF} α_{EF}


Subroutine DLIV
CALL SMASK (O,'OOFO'X)
C = 180 / π
FT ≤ 0
F
T
φ1 = RX(1) - 4*Δφ
φ999 = RX(1) + 4*Δφ
φ999 = 2π
CALL SPIX (0.0)
CALL SET (0,0)
CALL RECT (360,500)
i = i + 1
10
CALL EX1*
φi ≤ φ999 - Δφ
T
φi+1 = φi + Δφ
F
φi+1 = φ999
F
φi = φ999
T
CALL SPIX (0,'5050'X)
DO 20 j = 1 , i-1
Fj ≥ 0
F
T
CALL LINE (C*φj, Fj, C*φj+1, Fj+1)
20
100


100
CALL SPIX (0,'2020'X)
CALL LINE (C*φ1, DF,C*φ999, DF)
CALL CROS
N1 = 0
T
READ k
F
RX(1) = RX(k)
RY(1) = RY(k)
T
RY(1) = 0
F
RRR - RRR component cannot be assembled.
STOP
RX(1) = RX(1) / C
CALL EX1
ERR = (FT - DF) / DF
ERR ≤ ε
F
i = 1
φ1 = RX(1) - Δφ
φ999 = RX(1) + Δφ
T
φa = atan2 (YF - YD, XF - XD)
Δφ = Δφ / 10
i = 0
N1 = 1
10
RETURN
CALL SPIX (0,0)
CALL SET (C*φ1, -C*Δφ, 0)
CALL RECT (21*C*Δφ,500)
CALL SPIX (0,'2020'X)
CALL LINE (C*φ1, DF, C*φ999, DF)

TRICOS

DUMMY ARGUMENT		X_B	Y_B	X_D	Y_D	M	BC	CD	X_C	Y_C	ϕ_{BC}	ϕ_{CD}
ACTUAL	POINT E	X_C	Y_C	X_G	Y_G	M_2	CE	EG	X_{E1}	Y_{E1}	ϕ_{CE}	ϕ_{EG}
	POINT D	X_B	Y_B	X_C	Y_C	M_1	BD	CD	X_{D1}	Y_{D1}	ϕ_{BD}	ϕ_{CD}
ARGUMENT	POINT F	X_E	Y_E	X_G	Y_G	M_3	EF	FG	X_{F1}	Y_{F1}	ϕ_{EF}	ϕ_{FG}

EX I

DUMMY ARGUMENT		X_B	Y_B	X_G	Y_G	BC	BD	CD	CE	EF	EG	FG	DF	M_1	M_2	M_3	ϕ
ACTUAL	WITH *	X_B	Y_B	X_G	Y_G	BC	BD	CD	CE	EF	EG	FG	DF	M_1	M_2	M_3	ϕ_1
ARGUMENT	WITHOUT *	X_B	Y_B	X_G	Y_G	BC	BD	CD	CE	EF	EG	FG	DF	M_1	M_2	M_3	RX(1)

DUMMY ARGUMENT		D'F'	ϕ_{CE}	ϕ_{EG}	X_C	Y_C	X_{D1}	Y_{D1}	X_{E1}'	Y_{E1}'	X_{F1}	Y_{F1}
ACTUAL	WITH *	F_1	ϕ_{CE}	ϕ_{EG}	X_C	Y_C	X_D	Y_D	X_E	Y_E	X_F	Y_F
ARGUMENT	WITHOUT *	FT	ϕ_{CE}	ϕ_{EG}	X_C	Y_C	X_D	Y_D	X_E	Y_E	X_F	Y_F

CROS

DUMMY ARGUMENT	X	Y	RX	RY	N
ACTUAL ARGUMENT	$C^{*}\phi$	DF	RX	RY	N

Subroutine VTIV

$$S_{CE} = (Y_C - Y_E) / (X_C - X_E)$$
$$S_{DF} = (Y_D - Y_F) / (X_D - X_F)$$

↓

$$BS = ((X_S - X_B)^2 + (Y_S - Y_B)^2)^{1/2}$$
$$SG = ((X_S - X_G)^2 + (Y_S - Y_G)^2)^{1/2}$$

↓

CALL INTE

↓

CALL VTII

↓

$$X_c = -\omega_{BS} (Y_C - Y_B)$$
$$Y_c = \omega_{BS} (X_C - X_B)$$
$$X_d = -\omega_{BS} (Y_D - Y_B)$$
$$Y_d = \omega_{BS} (X_D - X_B)$$
$$X_e = -\omega_{SG} (Y_E - Y_G)$$
$$Y_e = \omega_{SG} (X_E - X_G)$$
$$X_f = -\omega_{SG} (Y_F - Y_G)$$
$$Y_f = \omega_{SG} (X_F - X_G)$$

↓

$$\omega_{BC} = \omega_{BS}$$
$$\omega_{EF} = \omega_{SG}$$

↓

$$M_1 = AC\ (X_c, Y_c, X_e, Y_e, X_C, Y_C, X_E, Y_E)$$
$$M_2 = AC\ (X_d, Y_d, X_f, Y_f, X_D, Y_D, X_F, Y_F)$$

↓

$$\omega_{CE} = (M_1 / CE)\ ((X_c - X_e)^2 + (Y_c - Y_e)^2)^{1/2}$$
$$\omega_{DF} = (M_2 / DF)\ ((X_d - X_f)^2 + (Y_d - Y_f)^2)^{1/2}$$

↓

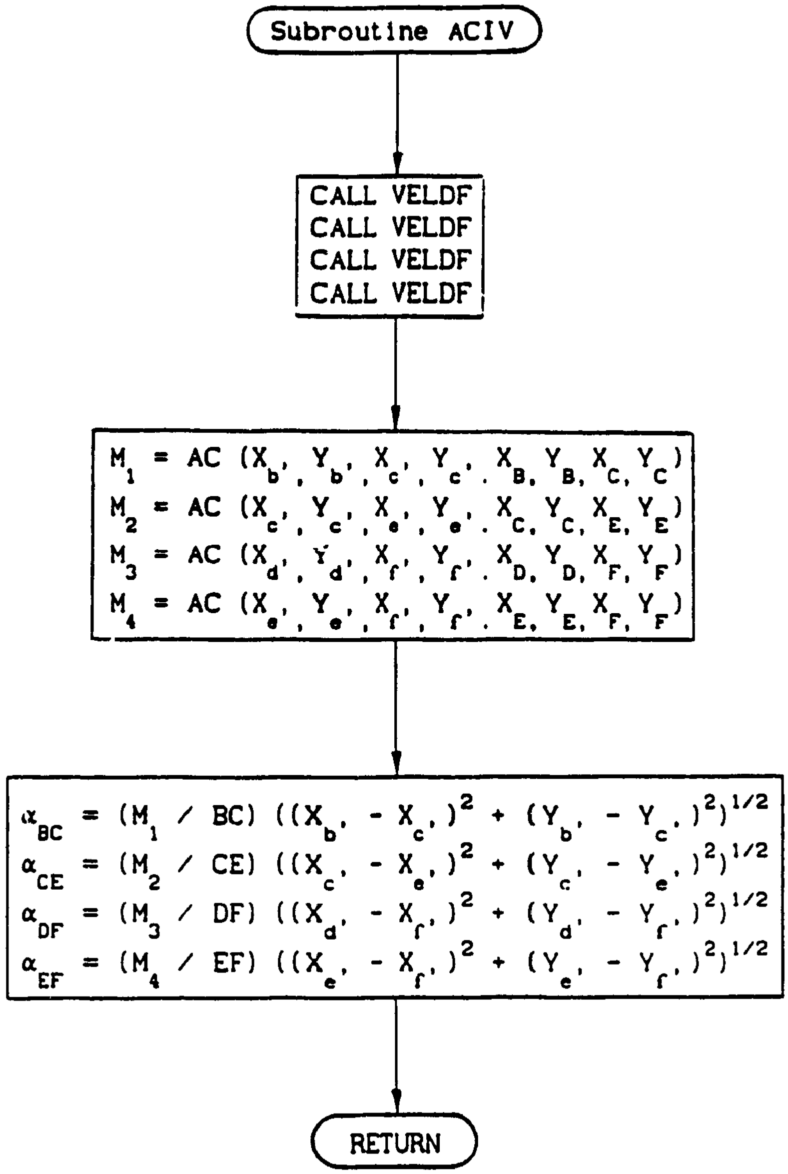
RETURN

INTE

DUMMY ARGUMENT	X_1	Y_1	S_1	X_2	Y_2	S_2	X_3	Y_3
ACTUAL ARGUMENT	X_C	Y_C	S_{CE}	X_D	Y_D	S_{DF}	X_S	Y_S

VTII

DUMMY ARGUMENT	X_B	Y_B	X_G	Y_G	X_D	Y_D	X_b	Y_b	X_d	Y_d	BC	CD	X_c	Y_c	ω_{BC}	ω_{DC}
ACTUAL ARGUMENT	X_B	Y_B	X_S	Y_S	X_G	Y_G	X_b	Y_b	X_g	Y_g	BS	SG	X_s	Y_s	ω_{BS}	ω_{SG}



VELDF

DUMMY ARGUMENT	$\Delta\phi$	ω	α	X_a	Y_a	$X_{a'}$	$Y_{a'}$
ACTUAL ARGUMENT POINT C	$\Delta\phi$	ω	α	X_c	Y_c	$X_{c'}$	$Y_{c'}$
ACTUAL ARGUMENT POINT D	$\Delta\phi$	ω	α	X_d	Y_d	$X_{d'}$	$Y_{d'}$
ACTUAL ARGUMENT POINT E	$\Delta\phi$	ω	α	X_e	Y_e	$X_{e'}$	$Y_{e'}$
ACTUAL ARGUMENT POINT F	$\Delta\phi$	ω	α	X_f	Y_f	$X_{f'}$	$Y_{f'}$

APPENDIX B

COMPUTER PROGRAMS

B. 1 Program CH_CR

```
      PROGRAM CH_CR
      IMPLICIT INTEGER*2 (B-Z)
      DIMENSION RX(72),RY(72),X1(100),Y1(100),X2(100),Y2(100)
     1,AAX1(100),AAY1(100),AAX2(100),AAY2(100),AAL(73)
      REAL PI,S,C
      COMMON AL,A3,BANK,X0,Y0,X3,Y3,R1,R2,R3
      COMMON LXI,LYI,LX1,LY1,RXN,RYN
      EXTERNAL cnio_OPEN,cnio_ready,cnio_READ,cnio_CLOSE
      BYTE CHAR,cnio_READ
      LOGICAL  cnio_ready
      CHARACTER*6,cnio
      INTEGER CHANNEL,EFN
      OPEN(UNIT=5,FILE='ST.DAT',STATUS='OLD')
      call cnio_open
C
C     IN THIS PROGRAM, THE "CH", THE MAIN BODY OF THE PROGRAM, IS EMPLOYED
C     TO RUN THE LINKAGE, AND INTERACTIVE GRAPHIC TECHNIQUE IS APPLIED TO
C     CONTROL THE LINKAGE MOVEMENT. HENCE, THE COORDINATES OF ALL THE
C     HINGE POINTS ARE TRANSMITTED TO THE SUBROUTINE "VSEARCH" TO DETERMINE
C     VELOCITIES ESPECIALLY THE ONE FOR THE INTERSECTING POINT OF THE COUPLER
C     AND THE FOLLOWER.
C
      PI=3.1415926
      CALL VDPINI
      TYPE*,'READ Y/N?'
      READ 5,I1
5     FORMAT(A1)
      IF(I1.NE.'Y') GO TO 4
      READ(9,*)X0,Y0,X3,Y3,UP
      READ(9,*)R1,R2,R3
      GO TO 6
4     TYPE*,'X0,Y0,X3,Y3,UP=?'
      READ*,X0,Y0,X3,Y3,UP
      TYPE*,'R1,R2,R3=?'
      READ*,R1,R2,R3
6     AR1=R1
      AR2=R2
      AR3=R3
      AX3=X3
      AY3=Y3
      DO I=1,73
      CALL COLLCT(2)
      AL=(I-1.)*PI/36.
      S=SIN(AL)
      C=COS(AL)
      X1(I)=X0+NINT(R1*C)
      Y1(I)=Y0-NINT(R1*S)
      AAX1(I)=X0+AR1*C
      AAY1(I)=Y0-AR1*S
      CALL SMASK(0,'00F0'X)
      IF(I.EQ.1) GO TO 1
      CALL SPIX(0,0)
      I1=X1(I-1)
      I2=Y1(I-1)
      IF(UP.EQ.1) THEN
              AYN=Y3-R3-10
              AYK=Y3+10
      ELSE
              AYN=Y3-10
```

```
                AYK=Y3+R3+10
        END IF
        AY1=I2
        XN=NINT(I1+SQRT(AR2**2-(AYN-AY1)**2))
        XK=NINT(I1+SQRT(AR2**2-(AYK-AY1)**2))
        YN=AYN
        YK=AYK
        CALL PRTCIR(I1,I2,XN,YN,XK,YK)
1       CALL SPIX(0,'0010'X)
        I1=X3
        ATHE=PI/15.
        II2=X3+NINT(R3*UP*COS(ATHE))
        II3=Y3-NINT(R3*SIN(ATHE))
        II4=X3-NINT(R3*UP*COS(ATHE))
        CALL PRTCIR(I1,Y3,II4,II3,II2,II3)
        CALL SPIX(0,'0020'X)
        I1=X1(I)
        I2=Y1(I)
        IF(UP.EQ.1) THEN
                AYN=Y3-R3-10
                AYK=Y3+10
        ELSE
                AYN=Y3-10
                AYK=Y3+R3+10
        END IF
        AY1=I2
        XN=NINT(I1+SQRT(AR2**2-(AYN-AY1)**2))
        XK=NINT(I1+SQRT(AR2**2-(AYK-AY1)**2))
        YN=AYN
        YK=AYK
        CALL PRTCIR(I1,I2,XN,YN,XK,YK)
        CALL CROS(XN,YN,RX,RY,N)
        X2(I)=RX(1)
        Y2(I)=RY(1)
C
C    TO BE MORE CONVENIENT IN CONTROLLING THE PROCESSING OF THE PROGRAM,
C      THE INTERECTIVE PROGRAMMING TECHNIQUE IS INTRODUCED HERE.
C
        IF (chio_ready(0)) THEN
          CHAR=chio_READ(0)
C
C    IF THE KEY 'S' IS PUSHED, THE ENTIRE PROGRAM WILL BE TERMINATED.
C
          IF (CHAR.EQ.'S') GO TO 8
          IF (CHAR.EQ.'A') NN1=0
          IF (CHAR.EQ.'B') NN1=1
C
C    IF THE KEY 'A' IS PUNCHED, THE VELOCITY SEARCHING SUBROUTINE
C      'VSEARCH' WILL BE ACTIVATED. OTHERWISE TYPING 'B' WILL ACTIVATE
C       SUBROUTINE 'ASEARCH' IN WHICH ACCELERATION WILL BE EVALUATED.
C
          CALL VSCH_CR(X0,Y0,X1(I),Y1(I),X2(I),Y2(I),X3,Y3,NN,NN1,I)
          IF (CHAR.EQ.'A') NX=0
          PAUSE
          PRINT*,'TO CONTINUE, TYPE ANY KEY'
        END IF
        CALL RESTORE(BANK)
        IF(I.EQ.1) THEN
          CALL DRAW(X1(I),Y1(I),X2(I),Y2(I),'B0B0'X)
          CALL DRAW(0,0,0,0,0)
        END IF
        CALL MV(X1(I),Y1(I),X2(I),Y2(I))
        END DO
8       I=73
        CALL MCLR
        STOP
```

```
      END
C
C     ***********************************************
      SUBROUTINE RESTORE(BANK)
      IMPLICIT INTEGER*2 (B-Z)
      IF(BANK.EQ.2) CALL SMASK('F000'X,0)
      IF(BANK.EQ.0) CALL SMASK('00F0'X,0)
      RETURN
      END
C     ***********************************************
      SUBROUTINE DRAW(X1,Y1,X2,Y2,SP)
      IMPLICIT INTEGER*2 (B-Z)
      DIMENSION RX(3),RY(3)
      COMMON AL,A3,BANK,X0,Y0,X3,Y3,R1,R2,R3
      COMMON LXI,LYI,LX1,LY1,RXN,RYN
      SP1='5050'X
      IF(SP.EQ.0) THEN
2       SP1=0
        XX1=X1SS
        YY1=Y1SS
        XX2=X2SS
        YY2=Y2SS
        X1SS=X1S
        X2SS=X2S
        Y1SS=Y1S
        Y2SS=Y2S
      ELSE
        X1S=X1
        X2S=X2
        Y1S=Y1
        Y2S=Y2
        XX1=X1
        YY1=Y1
        YY2=Y2
        XX2=X2
      END IF
      CALL SPIX('5050'X,0)
      CALL CIRCFL(X0,Y0,4)
      CALL CIRCFL(X3,Y3,4)
      CALL SPIX('2020'X,0)
      RX(1)=X0
      RY(1)=Y0
      RX(2)=X0+12
      RY(2)=Y0+16
      RX(3)=X0-12
      RY(3)=Y0+16
      CALL PLYFIL(3,RX,RY,0)
      RX(1)=X3
      RY(1)=Y3
      RX(2)=X3+12
      RY(2)=Y3+16
      RX(3)=X3-12
      RY(3)=Y3+16
      CALL PLYFIL(3,RX,RY,0)
      CALL SPIX(SP,0)
      CALL LINE(X0,Y0,XX1,YY1)
      CALL LINE(XX1,YY1,XX2,YY2)
      CALL LINE(XX2,YY2,X3,Y3)
      CALL SPIX(SP1,0)
      CALL CIRCFL(XX1,YY1,4)
      CALL CIRCFL(XX2,YY2,4)
1     RETURN
      END
C     ***********************************************
      SUBROUTINE MV(X1,Y1,X2,Y2)
      COMMON AL,A3,BANK,X0,Y0,X3,Y3,R1,R2,R3
```

```
          S2=0
          IF (N.GE.1) RETURN
        END IF
C
C   IN SPECIFIC CASES, SUCH AS APPLYING THE 'CROS' SUBROUTINE TO
C      THE VELOCITY OR ACCELERATION POLYGON APPROACH, USUALLY ONE
C      INTERSECTIONS WILL BE FOUND. TO SAVE COMPUTER TIME, THE
C      SUBROUTINE IS MODIFIED SUCH THAT IT RETURNS WHEN TWO CROSSING
C      POINTS ARE ENCOUNTERED.
C
5         IF(XS.EQ.O) GO TO 3
          IF(XS.EQ.-1) GO TO 6
          IF(XX(1).EQ.XS.AND.YY(1).EQ.YS) GO TO 3
6         XS=XN
          YS=YN
          XN=XX(1)
          YN=YY(1)
          GO TO 4
3         XS=XN
          YS=YN
          XN=XX(2)
          YN=YY(2)
4         IF(XN.EQ.X.AND.YN.EQ.Y) RETURN
          GO TO 100
        ELSE
          NN=NN+1
          S1=S1+XN
          S2=S2+YN
          GO TO 5
        END IF
        END
C   **************************************************************
        SUBROUTINE SWITCH(BANK)
        IMPLICIT INTEGER*2 (A-Z)
        CALL WVRET
        IF(BANK.EQ.O) THEN
          BANK=2
          CALL SMASK('F000'X,'0000'X)
          CALL WGBT('0920'X,O,1,1,8)
          CALL WGBT('0A20'X,O,1,1,0)
        ELSE ! IF(BANK.EQ.2) THEN
          BANK=0
          CALL SMASK('00F0'X,'0000'X)
          CALL WGBT('0920'X,O,1,1,'000A'X)
          CALL WGBT('0A20'X,O,1,1,'0012'X)
        END IF
        RETURN
        END
C
C *******************************************************
        SUBROUTINE VSCH_CR(XA,YA,XB,YB,XC,YC,XD,YD,NN,NN1,I9)
C -------------------------------------------------------------------
C   IN FACT, THE MORE ACCURATE WAY TO DETERMINE THE SLOPE OF R3 IS TO   |
C      USE THE COORDINATE OF POINT 3 IN REAL NUMBER INSTEAD OF INTEGER, |
C      WHICH WAS EXTRACTED FROM THE MAIN PROGRAM.                       |
C -------------------------------------------------------------------
        IMPLICIT INTEGER*2 (B-Z)
        COMMON AL,A3,BANK,XO,YO,X3,Y3,R1,R2,R3
        COMMON LXI,LYI,LX1,LY1,RXN,RYN
        REAL SLOPE1,SLOPE2,SLOPE3,THETA,OMEGA,VEL1,VEL2,VEL3,PI
        DIMENSION RRX(3),RRY(3),X(3),Y(3)
        CHARACTER*31 IFUN(6)
        OPEN(UNIT=6,FILE='CHARAC.DAT',STATUS='OLD')
        CALL COLLCT(1)
        CALL SMASK('00F0'X,O)
        SX=0
```

```
      CALL COLOR(XQ,YQ,XP,YP,RRX(1),RRY(1),SX)
      SX=80
      IF (NN.EQ.0) THEN
        DO I6=1,3
          READ (6,3),IFUN(I6)
3         FORMAT(A)
        END DO
      END IF
      CALL SMASK(0,'00F0'X)
      IF (NN.EQ.0) GO TO 8
      IF (NX.EQ.1) THEN
        CALL ASCH_CR(XA,YA,XB,YB,XC,YC,XD,YD,VEL1,VEL2,VEL3,DX,DY,NX,I9)
        GO TO 8
      END IF
      TYPE*,'PLEASE INDICATE THE CRANK ANGULAR VELOCITY, IN RPM'
      READ*,N1
C
C   THIS IS FOR ERRASING THE VELOCITY TRIANGLE.
C
7     CALL SPIX(0,0)
      CALL SET(X3D+8,Y3D+EXO)
      CALL TEXT(3,%REF(IFUN(2)))
      CALL SET(X5D+10,Y5D+EXO)
      CALL TEXT(3,%REF(IFUN(3)))
      CALL LINE(XQ,YQ,XP,YP)
      CALL LINE(X3D,Y3D,X4D,Y4D)
      CALL LINE(X5D,Y5D,X6D,Y6D)
      CALL PLYFIL(3,X,Y,-1)
2     IF (NN1.EQ.2) THEN
        NN1=3
        GO TO 4
      END IF
      GO TO 9
8     TYPE*,'PLEASE INDICATE THE CRANK ANGULAR VELOCITY IN RPM'
      READ*,N1
9     NN=NN+1
      PI=3.14159
C
C   TO EVALUATE THE SLOPES OF THE LINES PERPENDICULAR TO LINK 2 AND 3:
C
      SLOPE2=-FLOAT(XC-XB)/FLOAT(YC-YB)
      SLOPE3=-FLOAT(XD-XC)/FLOAT(YD-YC)
C
C   TO CALCULATE THE ANGULAR VELOCITY OF THE CRANK IN RADIANS:
C
      OMEGA=2.*N1*PI/60.
C
C   TO EVALUATE THE LINEAR VELOCITY OF POINT 2:
C
      VEL1=FLOAT(R1)*OMEGA
      THETA=AL
C
C   'THETA' HERE REPRESENTS THE CRANK ANGLE WITH RESPECT TO THE SCREEN
C     COORDINATE SYSTEM.
C
      XQ=255
      YQ=255
C
C   (XQ,YQ) IS THE ORIGIN OF THE VELOCITY POLYGON.
C
      XP=XQ-NINT(VEL1*SIN(THETA))
      YP=YQ-NINT(VEL1*COS(THETA))
C
C   (XP,YP) IS THE INTERSECTION OF V1 AND V2 AND IT IS THE TIP OF V1.
C
      CALL SPIX(0,'0010'X)
```

```
      DX=253
      DY=253
      X1D=XQ-DX
      X2D=XQ+DX
      CALL CALY(XP,YP,SLOPE2,X1D,Y1D,X2D,Y2D)
C
C    DEFINE THE BOUNDARIES OF THE WINDOW AS: DX=253,DY=253
C    WHICH ARE ONE HALF OF THE TOTAL WIDTH AND HEIGHT OF THE
C    WINDOW.
C
      CALL CLIP(XQ,YQ,DX,DY,X1D,Y1D,X2D,Y2D)
      CALL LINE(X1D,Y1D,X2D,Y2D)
      CALL SPIX(0,'0020'X)
C
C   TO RESTORE THE VALUES OF X1D AND X2D:
C
      X3D=X1D
      Y3D=Y1D
      X4D=X2D
      Y4D=Y2D
      X1D=XQ-DX
      X2D=XQ+DX
      CALL CALY(XQ,YQ,SLOPE3,X1D,Y1D,X2D,Y2D)
      CALL CLIP(XQ,YQ,DX,DY,X1D,Y1D,X2D,Y2D)
      CALL LINE(X1D,Y1D,X2D,Y2D)
      X5D=X1D
      Y5D=Y1D
      X6D=X2D
      Y6D=Y2D
      PRINT*,X6D,Y6D
      CALL CROS(X6D,Y6D,RRX,RRY,NM)
      CALL XMIT
      CALL CLEAR
C
C   THIS IS TO REPRESENT V1.
C
      IF (NM.EQ.0) THEN
         TYPE*,'LINK2 AND 3 ARE COINCIDENT'
         GO TO 12
      END IF
      AREP=(FLOAT(RRX(1)-XQ))**2+(FLOAT(RRY(1)-YQ))**2
      VEL3=SQRT(AREP)
      VEL2=SQRT((FLOAT(RRX(1)-XP))**2+(FLOAT(RRY(1)-YP))**2)
      IF (NN1.EQ.1) THEN
        NN1=2
        GO TO 7
      END IF
4     IF (NN1.EQ.3) THEN
        CALL ASCH_CR(XA,YA,XB,YB,XC,YC,XD,YD,VEL1,VEL2,VEL3,DX,DY,NX,I9)
        GO TO 12
      END IF
      CALL SMASK('00FO'X,0)
      CALL COLOR(XQ,YQ,XP,YP,RRX(1),RRY(1),SX)
      CALL XMIT
      CALL CLEAR
      SX=0
12    RETURN
      END
C
C   *********************************************************
      SUBROUTINE CLIP(XQ,YQ,DX,DY,X1D,Y1D,X2D,Y2D)
      IMPLICIT INTEGER*2 (B-Z)
C
C   ROUTINE TO FIND CLIPPED POINTS (X1D,Y1D) AND (X2D,Y2D) CORRESPOINDING
C     TO POINTS (X1,Y1) AND (X2,Y2).
C
```

```
      X1D=X1D-XQ
C
C AFTER TRANSLATING THE COORDINATE SYSTEM BY(XQ,YQ), INVERSE THE Y-AXIX
C
      Y1D=-Y1D+YQ
      X2D=X2D-XQ
      Y2D=-Y2D+YQ
C
C   FIND FRAME MODES OF (X1D,Y1D) AND (X2D,Y2D)
      CALL MODES(X1D,Y1D,DX,DY,IX1,IY1)
      CALL MODES(X2D,Y2D,DX,DY,IX2,IY2)
C
C   IF POINTS ARE IN THE SAME SECTOR OFF-SCREEN THEN RETURN.
C
      IF(IX1*IX2.EQ.1.OR.IY1*IY2.EQ.1) THEN
        X1D=0
        Y1D=0
        X2D=0
        Y2D=0
        TYPE*,'SCALING DOWN IS NECESSARY'
        GO TO 6
      END IF
      IF(IX1.EQ.0) GO TO 2
C
C   MOVE POINT 1 TO NEARER FRAME X-EDGE.
C
      XX=DX*IX1
      Y1D=Y1D+NINT(FLOAT(Y2D-Y1D)*FLOAT(XX-X1D)/FLOAT(X2D-X1D))
      X1D=XX
      CALL MODES(X1D,Y1D,DX,DY,IX1,IY1)
2     IF (IY1.EQ.0) GO TO 3
C
C   MOVE POINT 1 TO NEARER FRAME Y-EDGE.
C
      YY=DY*IY1
      X1D=X1D+NINT(FLOAT(X2D-X1D)*FLOAT(YY-Y1D)/FLOAT(Y2D-Y1D))
      Y1D=YY
3     IF (IX2.EQ.0)GO TO 4
C
C   MOVE POINT2 TO NEARER FRAME X-EDGE.
C
      XX=DX*IX2
      Y2D=Y1D+NINT(FLOAT(Y2D-Y1D)*FLOAT(XX-X1D)/FLOAT(X2D-X1D))
      X2D=XX
      CALL MODES(X2D,Y2D,DX,DY,IX2,IY2)
4     IF (IY2.EQ.0) GO TO 5
C
C   MOVE POINT 2 TO NEARER FRAME Y-EDGE.
C
      YY=DY*IY2
      X2D=X1D+NINT(FLOAT(X2D-X1D)*FLOAT(YY-Y1D)/FLOAT(Y2D-Y1D))
      Y2D=YY
C
C    NOW DRAW THE LINE BETWEEN THE NEW POINTS IF THEY ARE NOT
C      COINCIDENT
C
5     IF (ABS(X1D-X2D).EQ.0.AND.ABS(Y1D-Y2D).EQ.0) GO TO 6
      X1D=X1D+XQ
      Y1D=-Y1D+YQ
      X2D=X2D+XQ
      Y2D=-Y2D+YQ
6     RETURN
      END
C
C   ****************************************************************
      SUBROUTINE MODES(X,Y,DX,DY,IX,IY)
```

```
      IMPLICIT INTEGER*2 (B-Z)
C
C   SUBROUTINE TO FIND FRAME MODE OF POINT (X,Y)
C     DX*2 AND DY*2 IS THE SIZE OF THE FRAME CENTRED ON THE ORIGIN.
C
      IX=0
      IY=0
      IF(ABS(X).GT.DX) IX=X/ABS(X)
      IF(ABS(Y).GT.DY) IY=Y/ABS(Y)
      RETURN
      END
C
C   ****************************************************************
      SUBROUTINE CALY(XR,YR,SLOPE,X1D,Y1D,X2D,Y2D)
      IMPLICIT INTEGER*2 (B-Z)
      REAL SLOPE
C
C   THIS ROUTNE IS TO FIND THE Y-COORDINATES CORRESPONDING TO
C     X=2 AND X=508 WHICH ARE THE BOUNDARIES OF THE WINDOW FOR
C     XQ=YQ=255 AND DX=DY=253.
C
      Y1D=NINT(SLOPE*(X1D-XR)+YR)
      Y2D=NINT(SLOPE*(X2D-XR)+YR)
      RETURN
      END
C
C   ******************************************************************
      SUBROUTINE ASCH_CR(XA,YA,XB,YB,XC,YC,XD,YD,VEL1,VEL2,VEL3,DX,DY,NX,I9)
      IMPLICIT INTEGER*2 (B-Z)
      DIMENSION RX2(13),RY2(13),XZ(5),YZ(5)
      COMMON AL,A3,BANK,XO,YO,X3,Y3,R1,R2,R3
      COMMON LXI,LYI,LX1,LY1,RXN,RYN
      REAL VEL1,VEL2,VEL3,SLOPEB,SLOPEC,PI,VV1(73),VV2(73),VV3(73)
     *      ,AA2(73),AA3(73)
      CALL COLLCT(1)
      CALL SMASK(0,'00F0'X)
      PI=ACOS(-1.)
      IF (NX.EQ.0) GO TO 9
      CALL SPIX(0,0)
      CALL LINE(LXI,LYI,LX1,LY1)
      CALL LINE(LX1,LY1,LX2,LY2)
      CALL LINE(LXI,LYI,LX3,LY3)
      CALL LINE(X3C,Y3C,X4C,Y4C)
      CALL LINE(X5C,Y5C,X6C,Y6C)
      CALL PLYFIL(3,XZ,YZ,1)
      H=1
      CALL ABD(XZ(1),YZ(1),XZ(2),YZ(2),XZ(3),YZ(3),H)
      IF (NX.EQ.1) THEN
        NX=0
        GO TO 11
      END IF
9     NX=1
C
C   TO CALCULATE THE NORMAL COMPONENT OF THE CRANK ACCELERATION:
C     (ASSUME TANGENTIAL ONE IS ZERO FOR THE TIME BEING.)
C
      ACC1N=VEL1**2/R1
C
C   NORMAL COMPONENT OF THE COUPLER ACCELERATION:
C
      ACC2N=VEL2**2/R2
C
C   NORMAL COMPONENT OF THE ACCELERATION OF THE FOLLOWER:
C
      ACC3N=VEL3**2/R3
C
```

```
C    TO DETERMINE THE DIRECTION OF THESE NORMAL COMPONENTS OF ACCELERATION.
C      THEIR ANGLES RELATIVE TO THE HORIZONTAL X-AXIS ARE TO BE EVALUATED.
C      AND THE POSITIVE DIRECTION IS SET TO BE COUNTERCLOCKWISE.
C
        ALFA1=ATAN2(FLOAT(YA-YB),FLOAT(XA-XB))
        ALFA2=ATAN2(FLOAT(YB-YC),FLOAT(XB-XC))
        ALFA3=ATAN2(FLOAT(YD-YC),FLOAT(XD-XC))
C
C    SUPPOSE THE ORIGIN OF THE ACCELERATION POLYGON IS AT THE CENTRE OF
C      OF THE SCREEN FOR EXAMPLE (250,250).
C
        LXI=255
        LYI=255
C
C    TO REPRESENT A1|n
C
        LX1=LXI+ACC1N*COS(ALFA1)
        LY1=LYI+ACC1N*SIN(ALFA1)
C
C    TO EXPRESS A2|n
C
        LX2=LX1+ACC2N*COS(ALFA2)
        LY2=LY1+ACC2N*SIN(ALFA2)
C
C    TO DETERMINE A3|n
C
        LX3=LXI+ACC3N*COS(ALFA3)
        LY3=LYI+ACC3N*SIN(ALFA3)
        CALL SPIX(0,'0020'X)
C
C    TO EVALUATE THE SLOPES OF THE TWO TANGENTIAL COMPONENTS OF THE REST
C      ACCELERATIONS.
C
        SLOPEB=TAN(ALFA2+PI/2.)
        SLOPEC=TAN(ALFA3+PI/2.)
C
C    A SIMILAR CLIPPING PROCEDURE TO THE ONE APPLIED IN THE VELOCITY
C      EVALUATION PERFORMED PREVIOUSLY IS TO BE CARRIED OUT AGAIN TO
C      CLOSE THE POLYGON.
C
        X1C=LXI-DX
        X2C=LXI+DX
        AZ2=.5
        CALL CALY(LX2,LY2,SLOPEB,X1C,Y1C,X2C,Y2C)
        CALL CLIP(LXI,LYI,DX,DY,X1C,Y1C,X2C,Y2C)
        CALL LINE(X1C,Y1C,X2C,Y2C)
C
C    TO RESTORE VALUES OF X1C AND X2C AFTER THEIR CHANGE BY "CLIP"
C
        X3C=X1C
        Y3C=Y1C
        X4C=X2C
        Y4C=Y2C
        X1C=LXI-DX
        X2C=LXI+DX
        CALL SPIX(0,'0010'X)
        CALL CALY(LX3,LY3,SLOPEC,X1C,Y1C,X2C,Y2C)
        CALL CLIP(LXI,LYI,DX,DY,X1C,Y1C,X2C,Y2C)
        CALL LINE(X1C,Y1C,X2C,Y2C)
        X5C=X1C
        Y5C=Y1C
        X6C=X2C
        Y6C=Y2C
        PRINT*,'X=',X6C,'Y=',Y6C
        CALL CROS(X6C,Y6C,RX2,RY2,NM,IJ)
        CALL LINE(LXI,LYI,LX1,LY1)
```

```
      CALL LINE(LX1,LY1,LX2,LY2)
      CALL LINE(LXI,LYI,LX3,LY3)
C
C  TO CALCULATE THE RESULTANT ACCELERATION OF POINT C:
C
10    AEX8=(FLOAT(LXI-RX2(1)))**2+(FLOAT(LYI-RY2(1)))**2
      A3=SQRT(AEX8)
12    H=0
      RXN=RX2(1)
      RYN=RY2(1)
      CALL ABD(LXI,LYI,LX1,LY1,RX2(1),RY2(1),H)
      SX=80
      CALL SMASK('00F0'X,0)
      CALL COLOR(LXI,LYI,LX1,LY1,RX2(1),RY2(1),SX)
      CALL XMIT
      CALL CLEAR
11    RETURN
      END
C
C  THE FOLLOWING FUNCTION IS TO EVALUATE THE ANGLE IN A 360-DEGREE
C    RANGE.
C
C  *****************************************************************
      SUBROUTINE ABD(XE1,YE1,XE2,YE2,XE3,YE3,H)
      IMPLICIT INTEGER*2(B-Z)
      CHARACTER*31 IFUN(4)
      DIMENSION XD1(3),YD1(3),ALF(3),AXM(3),AYM(3),XS(3),YS(3),AFI(3)
      REAL X(3),Y(3),TAN2
      IF(H.EQ.1) GO TO 14
      OPEN(UNIT=6,FILE='CHARAC2.DAT',STATUS='OLD')
      DO K=1,3
        READ(6,3),IFUN(K)
3       FORMAT(A3)
      END DO
      API=ACOS(-1.)
      XD1(1)=XE1
      YD1(1)=YE1
      XD1(2)=XE2
      YD1(2)=YE2
      XD1(3)=XE3
      YD1(3)=YE3
11    DO 1 I=1,2
        X(I)=FLOAT(XD1(I+1)-XD1(1))
        Y(I)=FLOAT(YD1(I+1)-YD1(1))
        AFI(I)=ATAN2(Y(I),X(I))
1     CONTINUE
      AE=ABS(AFI(2)-AFI(1))
      IF(AE.GT.API.AND.AFI(1).LT.AFI(2)) GO TO 12
      IF(AE.GT.API.AND.AFI(1).GE.AFI(2)) GO TO 13
      IF(AE.LT.API.AND.AFI(1).LT.AFI(2)) GO TO 13
12      EX=XD1(2)
        EY=YD1(2)
        IFUN(4)=IFUN(1)
        XD1(2)=XD1(3)
        YD1(2)=YD1(3)
        IFUN(1)=IFUN(3)
        XD1(3)=EX
        YD1(3)=EY
        IFUN(3)=IFUN(4)
        AFI(1)=AFI(2)
13    X(2)=FLOAT(XD1(3)-XD1(2))
      Y(2)=FLOAT(YD1(3)-YD1(2))
      X(3)=FLOAT(XD1(1)-XD1(3))
      Y(3)=FLOAT(YD1(1)-YD1(3))
      AFI(2)=ATAN2(Y(2),X(2))
      AFI(3)=ATAN2(Y(3),X(3))
```

```
      DO 2 J=1,3
        ALF(J)=AFI(J)-API/2.
        IF (J.EQ.3) THEN
          AXM(J)=.5*FLOAT((XD1(J)+XD1(J-2)))
          AYM(J)=.5*FLOAT((YD1(J)+YD1(J-2)))
          GO TO 5
        END IF
        AXM(J)=.5*FLOAT((XD1(J)+XD1(J+1)))
        AYM(J)=.5*FLOAT((YD1(J)+YD1(J+1)))
5       XS(J)=NINT(AXM(J)+30.*COS(ALF(J)))
        YS(J)=NINT(AYM(J)+30.*SIN(ALF(J)))
2     CONTINUE
      CALL CSET(1)
      CALL CSIZE(0)
      CALL SPAC(3,0)
      CALL ROTC(0)
      GO TO 15
14    CALL SPIX(0,0)
15    DO 4 K=1,3
      CALL SET(XS(K),YS(K))
      CALL TEXT(3,%REF(IFUN(K)))
4     CONTINUE
      RETURN
      END
C   ****************************************************************
      SUBROUTINE COLOR(XT,YT,XU,YU,XV,YV,SX)
      IMPLICIT INTEGER*2 (B-Z)
      SX2=NINT(SX*2.25)
      SX3=NINT(SX*2.7)
      IF (SX.EQ.0) THEN
         XXT=XTS
         YYT=YTS
         XXU=XUS
         YYU=YUS
         XXV=XVS
         YYV=YVS
      ELSE
         XTS=XT
         YTS=YT
         XUS=XU
         YUS=YU
         XVS=XV
         YVS=YV
         XXT=XT
         YYT=YT
         XXU=XU
         YYU=YU
         XXV=XV
         YYV=YV
      END IF
      CALL SPIX(SX,SX)
      CALL LINE(XXT,YYT,XXU,YYU)
      CALL SPIX(SX2,SX2)
      CALL LINE(XXT,YYT,XXV,YYV)
      CALL SPIX(SX3,SX3)
      CALL LINE(XXU,YYU,XXV,YYV)
      RETURN
      END
```

B. 2 Program CH_TR

```
      PROGRAM CH_TR
      IMPLICIT INTEGER*2 (B-Z)
      DIMENSION RX(72),RY(72),X1(100),Y1(100),X2(100),Y2(100)
     1,AAX1(100),AAY1(100),AAX2(100),AAY2(100),AAL(73),ALE(73)
      REAL T,T1,T2,T3,PI,S,C,UP
      COMMON AL,ACCE1,ACCE2,ACCE3,BANK,X0,Y0,X3,Y3,R1,R2,R3
      EXTERNAL CHIO_OPEN,CHIO_READY,CHIO_READ,CHIO_CLOSE
      BYTE     CHAR,CHIO_READ
      LOGICAL  CHIO_READY
      CALL CHIO_OPEN
C
C   "CH_TR", THE MAIN BODY OF THE PROGRAM, SIMULATES THE LINKAGE MOVEMENT.
C    INTERACTIVE GRAPHIC TECHNIQUE IS APPLIED TO BE MORE USER FRIENDLY.
C    THE COORDINATES OF ALL HINGE POINTS ARE HENCE TRANSMITTED TO THE
C    SUBROUTINE "VSCH_TR" TO DETERMINE VELOCITIES. ATTENTION HAS BEEN
C    CONCENTRATED ON THE OUTPUT POINT, THE INTERSECTION OF THE COUPLER
C    AND THE FOLLOWER.
C
      PI=3.1415926
      CALL VDPINI
      CALL COLLCT(1)
      TYPE*,'READ Y/N?'
      READ 5,I1
5     FORMAT(A1)
      IF(I1.NE.'Y') GO TO 4
      READ(1,*)X0,Y0,X3,Y3,UP,RX,RY,R3
4     TYPE*,'X0,Y0,X3,Y3,UP=?'
      READ*,X0,Y0,X3,Y3,UP
      TYPE*,'R1,R2,R3=?'
      READ*,R1,R2,R3
      AR1=R1
      AR2=R2
      AR3=R3
      AX3=X3
      AY3=Y3
      DO I=1,73
      AL=(I-1.)*PI/36.
      S=SIN(AL)
      C=COS(AL)
      X1(I)=X0+NINT(R1*C)
      Y1(I)=Y0-NINT(R1*S)
      AAX1(I)=X0+AR1*C
      AAY1(I)=Y0-AR1*S
      CALL SMASK(0,'00F0'X)
      CALL TRICOS(AAX1(I),AAY1(I),AX3,AY3,AR2,AR3,UP,AXBB,AYBB)
      CALL RESTORE(BANK)
      X2(I)=NINT(AXBB)
      Y2(I)=NINT(AYBB)
      IF(I.EQ.1) THEN
        CALL DRAW(X1(I),Y1(I),X2(I),Y2(I),'B0B0'X)
        CALL DRAW(0,0,0,0,0)
      END IF
      CALL MV(X1(I),Y1(I),X2(I),Y2(I))
      CALL XMIT
      CALL CLEAR
C
C   THE INTERACTIVE PROGRAMMING TECHNIQUE IS INTRODUCED.
C
      IF (CHIO_READY(0)) THEN
```

```
          CHAR=CHIO_READ(0)
C
C   IF THE KEY 'S' IS PUSHED, THE ENTIRE PROGRAM WILL BE TERMINATED.
C
          IF (CHAR.EQ.'S') GO TO 8
C
C   IF THE KEY 'A' IS PUNCHED, THE VELOCITY FINDING SUBROUTINE
C     'VSCH_TR' WILL BE ACTIVATED OTHERWISE 'B' TRIGGERS 'ACCEL'
C     FOR ACCELERATION ANALYSIS.
C
          IF (CHAR.EQ.'A') NN1=5
          IF (CHAR.EQ.'B') NN1=1
          CALL VSCH_TR(X0,Y0,X1(I),Y1(I),X2(I),Y2(I),X3,Y3,NN2,NN1,I)
          PAUSE
          PRINT*,'TO CONTINUE, TYPE ANY KEY'
        END IF
        END DO
        CALL MCLR
8       STOP
        END
C   ****************************************************
        SUBROUTINE RESTORE(BANK)
        IMPLICIT INTEGER*2 (B-Z)
        IF(BANK.EQ.2) CALL SMASK('F000'X,0)
        IF(BANK.EQ.0) CALL SMASK('00F0'X,0)
        RETURN
        END
C   ****************************************************
        SUBROUTINE DRAW(X1,Y1,X2,Y2,SP)
        IMPLICIT INTEGER*2 (B-Z)
        DIMENSION RX(3),RY(3)
        COMMON AL,ACCE1,ACCE2,ACCE3,BANK,X0,Y0,X3,Y3,R1,R2,R3
        SP1='5050'X
        IF(SP.EQ.0) THEN
2         SP1=0
          XX1=X1SS
          YY1=Y1SS
          XX2=X2SS
          YY2=Y2SS
          X1SS=X1S
          X2SS=X2S
          Y1SS=Y1S
          Y2SS=Y2S
        ELSE
          X1S=X1
          X2S=X2
          Y1S=Y1
          Y2S=Y2
          XX1=X1
          YY1=Y1
          YY2=Y2
          XX2=X2
        END IF
        CALL SPIX('5050'X,0)
        CALL CIRCFL(X0,Y0,4)
        CALL CIRCFL(X3,Y3,4)
        CALL SPIX('2020'X,0)
        RX(1)=X0
        RY(1)=Y0
        RX(2)=X0+12
        RY(2)=Y0+16
        RX(3)=X0-12
        RY(3)=Y0+16
        CALL PLYFIL(3,RX,RY,0)
        RX(1)=X3
        RY(1)=Y3
```

```
        RX(2)=X3+12
        RY(2)=Y3+16
        RX(3)=X3-12
        RY(3)=Y3+16
        CALL PLYFIL(3,RX,RY,0)
        CALL SPIX(SP,0)
        CALL LINE(X0,Y0,XX1,YY1)
        CALL LINE(XX1,YY1,XX2,YY2)
        CALL LINE(XX2,YY2,X3,Y3)
        CALL SPIX(SP1,0)
        CALL CIRCFL(XX1,YY1,4)
        CALL CIRCFL(XX2,YY2,4)
1       RETURN
        END
C     ***********************************************
        SUBROUTINE MV(X1,Y1,X2,Y2)
        COMMON AL,ACCE1,ACCE2,ACCE3,BANK,X0,Y0,X3,Y3,R1,R2,R3
        SP=0
        CALL DRAW(0,0,0,0,SP)
        SP='8080'X
        CALL DRAW(X1,Y1,X2,Y2,SP)
        CALL SWITCH(BANK)
        RETURN
        END
C     ***************************************************
        SUBROUTINE SWITCH(BANK)
        IMPLICIT INTEGER*2 (A-Z)
        CALL WVRET
        IF(BANK.EQ.0) THEN
          BANK=2
          CALL SMASK('F000'X,'0000'X)
          CALL WGBT('0920'X,0,1,1,8)
          CALL WGBT('0A20'X,0,1,1,0)
        ELSE ! IF(BANK.EQ.2) THEN
          BANK=0
          CALL SMASK('00F0'X,'0000'X)
          CALL WGBT('0920'X,0,1,1,'000A'X)
          CALL WGBT('0A20'X,0,1,1,'0012'X)
        END IF
        RETURN
        END
C ************************************************
C
        SUBROUTINE VSCH_TR(XA,YA,XB,YB,XC,YC,XD,YD,NN2,NN1,I)
        IMPLICIT INTEGER*2 (B-Z)
        COMMON AL,ACCE1,ACCE2,ACCE3,BANK,X0,Y0,X3,Y3,R1,R2,R3
        REAL THETA,OMEGA,VEL1,VEL2,VEL3,PI
        DIMENSION EX(2),EY(2),X1D(3),Y1D(3)
        HA=1
        CALL SMASK(0,'00F0'X)
        IF (NN2.EQ.0) GO TO 8
        PRINT*,'NN1=',NN1
        IF (NX.EQ.1) THEN
          CALL ASCH_TR(XA,YA,XB,YB,XC,YC,XD,YD,VEL1,VEL2,VEL3,NX)
          GO TO 8
        END IF
C
C     TO ERRASE THE VELOCITY POLYGON
C
        TYPE*,'PLEASE INDICATE THE CRANK ANGULAR VELOCITY, IN RPM'
        READ*,N1
7       CALL SPIX(0,0)
        H=1
        CALL PLYFIL(3,X1D,Y1D,1)
        CALL ABD(NINT(AXQ),NINT(AYQ),NINT(AXP),NINT(AYP),NINT(AXR),
     *          NINT(AYR),H,HA)
```

```
2       IF (NN1.EQ.2) THEN
          NN1=3
          GO TO 4
        END IF
        GO TO 9
8       TYPE*,'PLEASE INDICATE THE CRANK ANGULAR VELOCITY, IN RPM'
        READ*,N1
9       NN2=NN2+1
        PI=3.14159
C
C   TO CALCULATE THE ANGULAR VELOCITY OF THE CRANK
C
        OMEGA=2.*N1*PI/60.
        VEL1=FLOAT(R1)*OMEGA
C
C   TO EVALUATE THE SLOPES OF THE PERPENDICULARS OF THE COUPLER AND
C     FOLLOWER
C
        AS1=ATAN(FLOAT(YO-YB)/FLOAT(XO-XB))
        AS2=ATAN(FLOAT(YC-YB)/FLOAT(XC-XB))
        AS3=ATAN(FLOAT(YD-YC)/FLOAT(XD-XC))
        IF (AS2.EQ.AS3) THEN
            IF (AS1.EQ.AS2) THEN
                PRINT*,'CHANGE POINT MECHANISM ENCOUNTERED'
                VEL2=VEL1
                VEL3=0.
                NN1=3
                GO TO 4
            END IF
            PRINT*,'DEAD ZONE ENCOUNTERED; TRY ANOTHER CONFIGURATION'
        END IF
        THETA=AL
C
C   'THETA' DEPICTS THE CRANK ANGLE WITH RESPECT TO THE SCREEN
C     COORDINATE SYSTEM.
C
        AXQ=250.
        AYQ=250.
C
C   (XQ,YQ) IS THE ORIGIN OF THE VELOCITY POLYGON.
C
        AXP=AXQ-VEL1*SIN(THETA)
        AYP=AYQ-VEL1*COS(THETA)
C
C   (XP,YP) IS THE INTERSECTION OF V1 AND V2 AND IS THE TIP OF V1.
C
        CALL INTE(AXR,AYR,AS2,AS3,AXP,AYP,AXQ,AYQ)
        VEL2=SQRT((AXR-AXP)**2+(AYR-AYP)**2)
        VEL3=SQRT((AXR-AXQ)**2+(AYR-AYQ)**2)
        IF (NN1.EQ.1) THEN
          IF ((NN2.EQ.0).OR.(NX.EQ.0)) THEN
              NN1=3
              GO TO 4
          END IF
          NN1=2
          GO TO 7
          END IF
4       IF (NN1.EQ.3) THEN
          CALL ASCH_TR(XA,YA,XB,YB,XC,YC,XD,YD,VEL1,VEL2,VEL3,NX)
          GO TO 11
        END IF
        NA=3
        CALL MAXMIN(AXQ,AYQ,AXP,AYP,AXR,AYR,AXD,AYD,AXE,AYE,AXMA,AYMA,
     *            AXMI,AYMI,NA)
        CALL ARRANE(AXMI,AYMI,AXMA,AYMA,AXQ,AYQ,AXP,AYP,AXR,AYR,AXD,AYD,
     *            AXE,AYE,NA)
```

```
      X1D(1)=NINT(AXQ)
      Y1D(1)=NINT(AYQ)
      X1D(2)=NINT(AXP)
      Y1D(2)=NINT(AYP)
      X1D(3)=NINT(AXR)
      Y1D(3)=NINT(AYR)
      CALL SPIX(0,'00F0'X)
      PRINT*,X1D(1),Y1D(1),X1D(2),Y1D(2),X1D(3),Y1D(3)
      IF (NX.EQ.0.AND.NN1.EQ.5) THEN
        CALL SMASK(0,'00F0'X)
      CALL PLYFIL(NA,X1D,Y1D,1)
      H=0
      CALL ABD(NINT(AXQ),NINT(AYQ),NINT(AXP),NINT(AYP),NINT(AXR),
     *         NINT(AYR),H,HA)
        GO TO 12
      END IF
      CALL PLYFIL(NA,X1D,Y1D,1)
      H=0
      CALL ABD(NINT(AXQ),NINT(AYQ),NINT(AXP),NINT(AYP),NINT(AXR),
     *         NINT(AYR),H,HA)
11    CALL XMIT
      CALL CLEAR
      CALL RESTORE(BANK)
12    RETURN
      END
C
C   ****************************************************************
      SUBROUTINE ASCH_TR(XA,YA,XB,YB,XC,YC,XD,YD,VEL1,VEL2,VEL3,NX)
      IMPLICIT INTEGER*2 (B-Z)
      DIMENSION XZ(5),YZ(5)
      COMMON AL,ACCE1,ACCE2,ACCE3,BANK,X0,Y0,X3,Y3,R1,R2,R3
      REAL VEL1,VEL2,VEL3,PI,AA2(73),AA3(73)
      CALL SMASK(0,'00F0'X)
      HA=0
      PI=ACOS(-1.)
      IF (NX.EQ.0) GO TO 9
      CALL SPIX(0,0)
      H=1
      CALL PLYFIL(3,XZ,YZ,1)
      CALL ABD(LXI,LYI,LX1,LY1,XR,YR,H,HA)
      CALL LINE(NINT(AX1),NINT(AY1),NINT(AX2),NINT(AY2))
      CALL LINE(NINT(AX2),NINT(AY2),NINT(AXR),NINT(AYR))
      CALL LINE(NINT(AXI),NINT(AYI),NINT(AX3),NINT(AY3))
      CALL LINE(NINT(AX3),NINT(AY3),NINT(AXR),NINT(AYR))
      IF (NX.EQ.1) THEN
        NX=0
        GO TO 11
      END IF
9     NX=1
C
C   TO CALCULATE THE NORMAL COMPONENT OF THE CRANK ACCELERATION, A1n
C     (ASSUME TANGENTIAL ONE IS ZERO FOR THE TIME BEING.)
C
      ACC1N=VEL1**2/FLOAT(R1)
C
C   TO FIND THE NORMAL COMPONENT OF THE COUPLER ACCELERATION, A2n
C
      ACC2N=VEL2**2/FLOAT(R2)
C
C   TO FIND THE NORMAL COMPONENT OF THE ACCELERATION OF THE FOLLOWER, A3n
C
      ACC3N=VEL3**2/FLOAT(R3)
C
C   TO DETERMINE THE DIRECTION OF THESE NORMAL COMPONENTS OF ACCELERATIONS,
C     THEIR ANGLES RELATIVE TO THE X-AXIS ARE TO BE EVALUATED.
C     THE POSITIVE DIRECTION IS SET TO BE COUNTERCLOCKWISE.
```

```
C
      ALFA1=AATAN2(FLOAT(YA-YB),FLOAT(XA-XB))
      ALFA2=AATAN2(FLOAT(YB-YC),FLOAT(XB-XC))
      ALFA3=AATAN2(FLOAT(YD-YC),FLOAT(XD-XC))
C
C   THE ORIGIN OF THE ACCELERATION POLYGON IS AT THE CENTRE
C     OF THE SCREEN, FOR EXAMPLE (250,250).
C
      AXI=250.
      AYI=250.
C
C   TO DETERMINE A1n
C
      AX1=AXI+ACC1N*COS(ALFA1)
      AY1=AYI+ACC1N*SIN(ALFA1)
C
C   TO DETERMINE A2n
C
      AX2=AX1+ACC2N*COS(ALFA2)
      AY2=AY1+ACC2N*SIN(ALFA2)
C
C   TO DETERMINE A3n
C
      AX3=AXI+ACC3N*COS(ALFA3)
      AY3=AYI+ACC3N*SIN(ALFA3)
C
C   TO EVALUATE THE SLOPES OF THE TWO TANGENTIAL COMPONENTS OF A2 and A3
C
      ASB=(ALFA2+PI/2.)
      ASC=(ALFA3+PI/2.)
      IF (ASB.EQ.ASC) THEN
         IF (ALFA1.EQ.ALFA2) THEN
            PRINT*,'CHANGE POINT MECHANISM ENCOUNTERED'
            GO TO 11
         END IF
         PRINT*,'DEAD ZONE ENCOUNTERED; TRY ANOTHER CONFIGURATION'
         STOP
      END IF
      CALL SPIX(0,'0020'X)
C
C   TO CALCULATE THE A3
C

      LXI=NINT(AXI)
      LYI=NINT(AYI)
      LX1=NINT(AX1)
      LY1=NINT(AY1)
      LX2=NINT(AX2)
      LY2=NINT(AY2)
      LX3=NINT(AX3)
      LY3=NINT(AY3)
      CALL INTE(AXR,AYR,ASB,ASC,AX2,AY2,AX3,AY3)
      PRINT*,'ASB,ASC,AXR,AYR=',ASB,ASC,AXR,AYR
      NA=5
      CALL MAXMIN(AXI,AYI,AX1,AY1,AX2,AY2,AX3,AY3,AXR,AYR,
     *     AXMA,AYMA,AXMI,AYMI,NA)
      CALL ARRANE(AXMI,AYMI,AXMA,AYMA,AXI,AYI,AX1,AY1,AX2,
     *                AY2,AX3,AY3,AXR,AYR,NA)
      ACCE2=SQRT((AXR-AX1)**2+(AYR-AY1)**2)
      ACCE3=SQRT((AX3-AXR)**2+(AY3-AYR)**2)
      XZ(1)=NINT(AXI)
      YZ(1)=NINT(AYI)
      XZ(2)=NINT(AX1)
      YZ(2)=NINT(AY1)
      XZ(3)=NINT(AXR)
      YZ(3)=NINT(AYR)
```

```
      CALL PLYFIL(3,XZ,YZ,1)
      CALL SPIX(0,'0010'X)
      H=0
      CALL ABD(NINT(AXI),NINT(AVI),NINT(AX1),NINT(AY1),NINT(AXR),
     *       NINT(AVR),H,HA)
      CALL LINE(NINT(AX1),NINT(AV1),NINT(AX2),NINT(AV2))
      CALL LINE(NINT(AX2),NINT(AV2),NINT(AXR),NINT(AVR))
      CALL LINE(NINT(AXI),NINT(AVI),NINT(AX3),NINT(AV3))
      CALL LINE(NINT(AX3),NINT(AV3),NINT(AXR),NINT(AVR))
11    CALL RESTORE(BANK)
      RETURN
      END
C
C    *****************************************************************
      SUBROUTINE MAXMIN(AXA,AYA,AXB,AYB,AXC,AYC,AXD,AYD,AXE,AYE,AXMA,
     *              AYMA,AXMI,AYMI,NA)
      IMPLICIT INTEGER*2(B-Z)
      REAL XT(5),YT(5)
C
C     THIS IS TO FIND THE RECTANGULAR FRAME CONTAINING THE POLYGON.
C
      AXMA=-100000.
      AYMA=-100000.
      AXMI=100000.
      AYMI=100000.
      XT(1)=AXA
      XT(2)=AXB
      XT(3)=AXC
      XT(4)=AXD
      XT(5)=AXE
      YT(1)=AYA
      YT(2)=AYB
      YT(3)=AYC
      YT(4)=AYD
      YT(5)=AYE
      DO 1 I=1,5
        IF (XT(I).LT.AXMI) AXMI=XT(I)
        IF (YT(I).LT.AYMI) AYMI=YT(I)
        IF (XT(I).GT.AXMA) AXMA=XT(I)
        IF (YT(I).GT.AYMA) AYMA=YT(I)
1     IF (I.EQ.NA) GO TO 2
C
C   WHEN NA=3 VELOCITY POLYGON IS GENERATED; AND NA=5 IS FOR
C     ACCELERATION POLYGON.
C
2     RETURN
      END
C
C    *****************************************************************
      SUBROUTINE ARRANE(AXMI,AYMI,AXMA,AYMA,AXA,AYA,AXB,AYB,AXC,
     *              AYC,AXD,AYD,AXE,AYE,NA)
      IMPLICIT INTEGER*2(B-Z)
      ACOMX=ABS(AXMA-AXMI)
      ACOMY=ABS(AYMA-AYMI)
      AK=1.
      IF (ACOMX.LE.420..AND.ACOMY.LE.420.) GO TO 4
      IF (ACOMX.LE.420..AND.ACOMY.GT.420.) GO TO 3
      IF (ACOMX.GE.ACOMY) THEN
        CALL SCALE(AXMI,AXMA,AXA,AYA,AXB,AYB,AXC,AYC,AXD,AYD,AXE,AYE,
     *          NA,AK)
      ELSE
3       CALL SCALE(AYMI,AYMA,AXA,AYA,AXB,AYB,AXC,AYC,AXD,AYD,AXE,AYE,
     *          NA,AK)
      END IF
4     AA1=AK*AXMA
      AA2=AK*AYMA
```

```
      AA3=AK*AXMI
      AA4=AK*AYMI
      IF (AA1.GT.420..OR.AA2.GT.420..OR.AA3.LT.0..OR.AA4.LT.0.) THEN
        ASHIFX=-AA3+20.
        ASHIFY=-AA4+20.
       CALL  SHIFT(ASHIFX,ASHIFY,AXA,AYA,AXB,AYB,AXC,AYC,
     *             AXD,AYD,AXE,AYE,NA)
      END IF
      RETURN
      END
C
C    *******************************************************************
      SUBROUTINE SCALE(AD,AB,AXA,AYA,AXB,AYB,AXC,AYC,AXD,AYD,AXE,AYE,
     *             NA,AK)
      IMPLICIT INTEGER*2 (B-Z)
      AW=420./(AB/AD-1.)
      AK=AW/AD
      AXA=AK*AXA
      AYA=AK*AYA
      AXB=AK*AXB
      AYB=AK*AYB
      AXC=AK*AXC
      AYC=AK*AYC
      IF (NA.EQ.3) GO TO 1
      AXD=AK*AXD
      AYD=AK*AYD
      AXE=AK*AXE
      AYE=AK*AYE
1     RETURN
      END
C
C    *******************************************************************
      SUBROUTINE SHIFT(ASHIFX,ASHIFY,AXA,AYA,AXB,AYB,AXC,AYC,AXD,AYD,AXE,AYE,
     *             NA)
      IMPLICIT INTEGER*2(B-Z)
      AXA=AXA+ASHIFX
      AYA=AYA+ASHIFY
      AXB=AXB+ASHIFX
      AYB=AYB+ASHIFY
      AXC=AXC+ASHIFX
      AYC=AYC+ASHIFY
      IF (NA.EQ.3) GO TO 1
      AXD=AXD+ASHIFX
      AYD=AYD+ASHIFY
      AXE=AXE+ASHIFX
      AYE=AYE+ASHIFY
1     RETURN
      END
C
C    *******************************************************************
      SUBROUTINE INTE(XR,YR,AS2,AS3,AXI,AYI,AXII,AYII)
      IMPLICIT INTEGER*2(B-Z)
      REAL XR,YR
C
C    THIS IS TO FIND THE INTERSECTION OF TWO STRAIGHT LINES
C
      XR=(AYI-AYII-AXI*TAN(AS2)+AXII*TAN(AS3))/(TAN(AS3)-TAN(AS2))
      YR=TAN(AS2)*(XR-AXI)+AYI
      RETURN
      END
C
C    *******************************************************************
      SUBROUTINE ABD(XE1,YE1,XE2,YE2,XE3,YE3,H,HA)
      IMPLICIT INTEGER*2(B-Z)
      CHARACTER*31 IFUN(4)
      DIMENSION XD1(3),YD1(3),ALF(3),AXM(3),AYM(3),XS(3),YS(3),AFI(3)
```

```
      REAL X(3),Y(3),TAN2
      IF(H.EQ.1) GO TO 14
      IF (HA.EQ.1) THEN
      OPEN(UNIT=6,FILE='CHARAC.DAT',STATUS='OLD')
      GO TO 22
      END IF
      OPEN(UNIT=6,FILE='CHARAC2.DAT',STATUS='OLD')
22    DO K=1,3
        READ(6,3),IFUN(K)
3       FORMAT(A2)
      END DO
      API=ACOS(-1.)
      XD1(1)=XE1
      YD1(1)=YE1
      XD1(2)=XE2
      YD1(2)=YE2
      XD1(3)=XE3
      YD1(3)=YE3
11    DO 1 I=1,2
        X(I)=FLOAT(XD1(I+1)-XD1(1))
        Y(I)=FLOAT(YD1(I+1)-YD1(1))
C       AFI(I)=TAN2
        IF (X(I).EQ.0..AND.Y(I).EQ.0.) GO TO 10
        AFI(I)=AATAN2(Y(I),X(I))
1     CONTINUE
      AE=ABS(AFI(2)-AFI(1))
      IF(AE.GT.API.AND.AFI(1).LT.AFI(2)) GO TO 12
      IF(AE.GT.API.AND.AFI(1).GE.AFI(2)) GO TO 13
      IF(AE.LT.API.AND.AFI(1).LT.AFI(2)) GO TO 13
12      EX=XD1(2)
        EY=YD1(2)
        IFUN(4)=IFUN(1)
        XD1(2)=XD1(3)
        YD1(2)=YD1(3)
        IFUN(1)=IFUN(3)
        XD1(3)=EX
        YD1(3)=EY
        IFUN(3)=IFUN(4)
        AFI(1)=AFI(2)
13    X(2)=FLOAT(XD1(3)-XD1(2))
      Y(2)=FLOAT(YD1(3)-YD1(2))
      X(3)=FLOAT(XD1(1)-XD1(3))
      Y(3)=FLOAT(YD1(1)-YD1(3))
      AFI(2)=AATAN2(Y(2),X(2))
      AFI(3)=AATAN2(Y(3),X(3))
      DO 2 J=1,3
        ALF(J)=AFI(J)-API/2.
        IF (J.EQ.3) THEN
          AXM(J)=.5*FLOAT((XD1(J)+XD1(J-2)))
          AYM(J)=.5*FLOAT((YD1(J)+YD1(J-2)))
          GO TO 5
        END IF
        AXM(J)=.5*FLOAT((XD1(J)+XD1(J+1)))
        AYM(J)=.5*FLOAT((YD1(J)+YD1(J+1)))
5       XS(J)=NINT(AXM(J)+15.*COS(ALF(J)))
        YS(J)=NINT(AYM(J)+15.*SIN(ALF(J)))
2     CONTINUE
      CALL SPIX(0,'0010'X)
      CALL CSET(1)
      CALL CSIZE(0)
      CALL SPAC(3,0)
      CALL ROTC(0)
      GO TO 15
14    CALL SPIX(0,0)
15    DO 4 K=1,3
      CALL SET(XS(K),YS(K))
```

```
      CALL TEXT(3,%REF(IFUN(K)))
4     CONTINUE
10    RETURN
      END
C
C     ******************************************************************
      SUBROUTINE TRICOS (AXAA,AYAA,AXCC,AYCC,AL2,AL3,M,AXBB,AYBB)
      IMPLICIT INTEGER*2 (B-Z)
      REAL ALFA,GAMA,BATA,M
      ALO=SQRT((AXAA-AXCC)**2+(AYAA-AYCC)**2)
      ALFA=ACOS((AL3*AL3+ALO*ALO-AL2*AL2)/(2.*ALO*AL3))
      BATA=AATAN2((AYAA-AYCC),(AXAA-AXCC))
      GAMA=BATA+M*ALFA
      AXBB=AXCC+AL3*COS(GAMA)
      AYBB=AYCC+AL3*SIN(GAMA)
      RETURN
      END
```

B. 3 Program ERR

```
      PROGRAM ERR
C
C     THIS PROGRAM PROCESSES ERRORS FOR THE ANGULAR VELOCITIES AND
C       ACCELERATIONS OF EITHER THE COUPLER OR FOLLOWER, ONE SET
C       AT A TIME. SO THE ONLY THING NEEDS TO DO IS TO CHANGE THE
C       NAMES OF THE FOUR DATA FILES FOR DATA ACQUISITION.
C
C     THE ERROR ANALYSIS IS BASED ON THE COMPARISON OF THE RESULTS
C       FROM GRAPHICAL METHOD AND ANALYTIC GEOMETRY METHOD.
C
      REAL THE1(146),W2C(146),A2C(146),EW(73),EA(73)
      REAL W2F(146),A2F(146),MA,MW,LA,LW
      EWM=.0000001
      EAM=.0000001
      MW=-10000.
      MA=-10000.
      LW=10000.
      LA=10000.
      OPEN (UNIT=7,FILE='C210M1.DAT',STATUS='OLD')
         READ(7,*),((THE1(J),W2C(J)),J=1,146)
      OPEN (UNIT=7,FILE='F210M1.DAT',STATUS='OLD')
         READ(7,*),((THE1(J),W2F(J)),J=1,146)
         DO 100 L=1,73
              A2C(L)=W2C(L+73)
              A2F(L)=W2F(L+73)
100      CONTINUE
      DO J=1,73
        EW(J)=ERROR(W2C(J),W2F(J))
        EA(J)=ERROR(A2C(J),A2F(J))
        IF (EW(J).GT.EWM) EWM=EW(J)
        IF (EA(J).GT.EAM) EAM=EA(J)
        IF (W2C(J).GT.MW) MW=W2C(J)
        IF (A2C(J).GT.MA) MA=A2C(J)
        IF (W2C(J).LT.LW) LW=W2C(J)
        IF (A2C(J).LT.LA) LA=A2C(J)
      END DO
      RW=2.*EWM/(ABS(MW)+ABS(LW))
      RA=2.*EAM/(ABS(MA)+ABS(LA))
      PRINT*,RW,RA
      STOP
      END
C   ******************************************************************
      FUNCTION ERROR(A,B)
      ERROR=ABS(A-B)
      RETURN
      END
```