# A CAD/CAM Interface for Computer-Aided Design of Cams

by

Ashit R. Gandhi

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Mechanical Engineering

APPROVED:

| | |
|---|---|
| Dr. Arvid Myklebust, Chairman | Dr. Charles F. Reinholtz |
| Dr. Hamilton H. Mabie | Dr. Said W. Zewari |

December, 1985

Blacksburg, Virginia

# A CAD/CAM Interface for Computer-Aided Design of Cams

by

Ashit R. Gandhi

Dr. Arvid Myklebust, Chairman

Mechanical Engineering

(ABSTRACT)

The purpose of this thesis is to provide a complete package for the design and three dimensional modeling display of cams.

The software produced as a part of this work will operate as a module of CADAM to produce cam designs and enter the resulting cam as a CAD model and produce the graphical display of the cam.

In addition to the introductory material, this thesis is divided into four sections. The section on the graphics packages used in this thesis includes a brief history and capabilities of each of the packages. The second section details the procedure to be adopted in order to design a cam. The next section details ANICAM, the program that has been developed to incorporate the design and display procedure. The fourth section of this thesis contains recommendations for further work in this area.

The theoretical work in this project is a combination of original derivations and applications of the theory in the design literature.

## ACKNOWLEDGEMENTS

My deepest gratitude is extended to the chairman of my supervisory committee, Dr. Arvid Myklebust, for his guidance, encouragement and support throughout my graduate studies.

I wish to express my sincere appreciation to the members of the supervisory committee, Dr. Charles Reinholtz, Dr. Hamilton Mabie and Dr. Said Zewari for their advice and support.

For their support, my thanks to the fellow students who provided technical help and moral support. In particular, I thank

and .

Finally, I am indebted to my parents and family members for their unfailing love and support which made my studies in United States possible.

TABLE OF CONTENTS

## LIST OF ILLUSTRATIONS

# CHAPTER 1 INTRODUCTION

Today's fast changing technology has put tremendous pressure on corporations for the development of new systems and also to improve the current designs. This has turned out to be a complex, time consuming and often costly process. In the past, engineers would run into errors from even the most detailed blueprints. That meant redesigns which would cost several thousand dollars or more. The computer helped eliminate many errors in recent years, giving engineers more flexibility to experiment with a design before it was actually manufactured. Still much of that early Computer-Aided engineering was executed in 2-D on the computer screen, like sketching on an electronic piece of paper.

Today however the so called 3-D systems are available. The addition of depth, on 19-inch full-color screens has led to significant advances in engineering speed and accuracy. Using portable workstations, design engineers are able to work independently, as well as away from the office. This means that an engineer can check the designs in progress to avoid missteps as the design takes shape. It accelerates the communication between various engineers on a project and eliminates what could turn out to be costly errors. The reduced errors will also improve productivity. The 3-D systems are able to store design information from the early stages of a project. Since engineers know how to retrieve it, they are free to spend more time on problem solving. Software packages developed can help engineers deduce where the moving parts will show early stress signs, which parts can be cut by the manufacturer from the same piece of material

(to be ground) and what illustrations might be best included in the user's manual.

The ability to model in 3-D is significant since the designer no longer has to tie up time in manufacturing a model. This has been especially true when many design iterations are required to satisfy motion constraints. While many programs have been written to handle design and analysis, almost nothing can be found to link the analysis step to a 3-D modeler. Transferring data between these two steps has been very laborious. If a method had been available to do this automatically, a great savings in time and accuracy would have resulted. The purpose of this study is to create a program which will make it possible to tie these design steps together and present a method for creating 3-D animated models automatically.

This thesis introduces a program, ANICAM which has been created to animate the motion of cams in a 3-D format. It has been written in FORTRAN 77 computer language using the IBM4341 running the VM/CMS system and has been designed to serve as an interface between cam design and three existing programs: GKS, a graphics package, CADAM, and MOVIE.BYU which are packages for generating 3-D graphics and CAD models.

# CHAPTER 2 LITERATURE REVIEW.

The history of cams can be traced back several centuries. However it is very difficult to say where and when cams got started. In spite of being in use for a long time, it was only in the last four decades that major developments in the area of cam design have taken place.

While the kinematics of the cam-follower system have been thoroughly investigated by many including Rothbart(1), Jensen(2) and Chen(3), the absence of literature on the dynamic response has made it very difficult to validate analytic advances in this field. In recent years, investigations into the dynamics of the cam-follower systems have been performed by Johnson(4), Donkin and Clark(5) and others. However little experimental data, either in the form of system parameters or dynamic quantities, is available. Pisano(6) has gathered data on the description and recording of such measurements. The dynamics of cam-follower systems remains an active field of research and development.

The last two decades have seen a lot of work in the fields of design synthesis, design optimization, and computer-aided design. Many mathematical techniques have been used to find the optimal or best solution to the design problem. These consist of two phases: first, to generate alternative solutions and second, to identify the optimal solution among the acceptable ones based on certain criteria. Work on optimal design of the cam-follower systems has been done by Berzak(7) who optimally synthesized the system by the polydyne cam design method. Sermon and Liniecki(8) have used pressure angle and maximum stress constraints to

design minimum volume cams. Sandler(9) uses the spectral theory of random processes for the synthesis while Chen and Shah(10) have used the sequential random vector technique for the optimal design of geometric parameters of the system. Chew, Freudenstein and Longman(11) have suggested a model using the optimal-control theory.

The need to improve manufacturing productivity has generated large scale interest in computer-aided design and computer-aided manufacturing. In recent years, several computer programs have been written to perform basic design functions.

A number of programs for designing cam and follower systems are available. Most of these packages use the well known aspects of kinematic and static force analysis of cam mechanisms. COMMEND(12) is a program which considers almost every design aspect for the cam. An integrated program constisting of CAM, KAVM, and NCCAM(13) allows the design and graphical display of the cam-follower system. Other programs for the analysis and design of cam mechanisms include CAMPAK(14), DYCAM and CAMCHK(15). All the above programs allow two dimensional display of the physical characteristics and geometry of the cam-follower systems.

If computer graphics is to be exploited fully, then the design and analysis of the cam should be supported by high resolution, three dimensional shaded color images. Coupled with animation, this would give a realistic picture of the cam and follower. Three dimensional modeling packages like GEOMOD, CATIA, PADL-1 and 2, UNISOLIDS, UNISCAD(16) and recently CADAM have been developed by private corporations for commercial purposes. Another package, MOVIE.BYU, has been developed and distributed by the Brigham Young University. The availability of 3-D graphics pack-

ages has helped reduce the time required in tedious verifications of the practicality of the design.

While this review is by no means complete, it forms the backbone of this project which aims at developing a computer program (to be named ANICAM) for design and three dimensional display of cams.

# CHAPTER 3 GRAPHICS SUPPORT

## 3.1 GRAPHICAL KERNEL SYSTEM

Though standards in programming languages have existed for 25 years it was not until June 1982 that the Graphical Kernel System (GKS) was adopted by the International Organization for Standardization (ISO) as a draft International Standard. It was in the late 60's that some standard approaches began to appear in computer graphics. Had developments continued in this manner a standard would have been evolved much earlier. However in the late 60's, interactive graphics required an expensive refresh display and a dedicated host computer. A total cost of $400,000 was not uncommon and thus such systems were only available to a few. The advent of time sharing systems along with the emergence of storage tube display had a dramatic impact. Interactive graphics was now available to a larger number of users. The rapid changes in both hardware and patterns of usage left little time for evolving standardization in computer graphics. As widely differing devices were put into different facilities the need for a standard received more attention. Standards would help guard against the unproductive diversity of systems.

The process of working towards a common specification of a core graphics system began in August 1974, at an IFIP WG5.2 meeting in Malmo, Sweden. In February 1977 a meeting of the programming language subcommittee ISO/TC97/WG5 was held in London. It was resolved that no existing graphics package could be a suitable candidate for a graphics standard.

It was decided that a working group of SC5 should be established to deal with the standardization process.

GKS was first reported in the first formal meeting of WG2 (the graphics working group) in Bologna in September 1978. It was decided that GKS be put forward to the parent body for registration as a Work Item with the aim of it becoming a Draft Proposal in a year. Technical meetings were held to resolve outstanding issues, incorporate changes into the GKS document and present it to the parent body. GKS was accepted as a standard in June 1982. GKS was revised and is now Draft International Standard (DIS) 7942(10).

### 3.1.1 CAPABILITIES OF GKS

GKS is a subroutine package that provides functions for managing graphical output and input. It is a general-purpose graphics package that provides facilities designed to address a wide range of applications. An application program uses it as a standard interface to graphic devices. The package is a large one with well over a hundred functions defined. Although its size gives GKS the appearance of complexity, the concepts embodied in the package are quite simple. The principal concepts of GKS are enumerated in the following list:

- Separate functions are provided for input and output. There is some interplay between the functions.

- Graphical primitives for output go either directly to the display or into a segment. Segments are used to manage interactive changes to a display.
- Primitive attributes are set to control the appearance of each graphical output primitive.
- Segment attributes affect all the primitives in a segment.
- The regeneration of the display from a set of segments can be controlled explicitly. GKS may also provide implicit or immediate display updating.
- Graphical input is provided by a set of logical devices, which are not really "devices", but interactive techniques for obtaining a particular kind of information.
- Inquiry functions allow the application program to obtain a great deal of information about the output and input devices available to GKS. The program can use this information to tailor its operation to the capabilities of the device.

GKS provides functions for 2-D graphics only; extensions to GKS now being designed for 3-D graphics. A complementary standard for 3-D grahics and animation, PHIGS, is under development. At present, 3-D objects can be displayed if the application program projects the 3-D information to 2-D before calling GKS. The most common approach is to make a perspective drawing of the object. Perspective, however, is not the only mechanism for communicating depth information to the viewer. Images that show solid objects with simulated lighting, shadows, texture and color can achieve

startling realism. Unlike making perspective drawings, generating these pictures, require a great deal of computations.

The GKS functions are a set of tools for making a display device present the picture or for implementing an interactive graphical dialog the user has designed. While understanding the functions is essential to undertake a programming project, a feeling for the strategy used to design an interactive program is also essential for the tactics that get the most out of GKS.

## 3.2 CADAM

The CADAM (Computer-graphics Augmented Design and Manufacturing) system is an integrated set of computer programs with a common data base that can be used to prepare mechanical drawings on a computer terminal, similar to using a drawing board. This system can be used for design and manufacturing tasks in a wide variety of industries. CADAM programs were first developed and marketed in the mid 60's by the Lockheed Aircraft Corporation. Today, the CADAM system is one of the most widely used packages throughout the world. Recently a new version of the CADAM system was released. The system now has 3-D capabilities, allowing the user to create wire frame models.

### 3.2.1 CAPABILITIES OF CADAM

The CADAM software is a set of interrelated computer programs that allow the user to perform a variety of design or NC (Numerical Control)

tasks. The system uses a computer and various peripheral computer devices to accomplish this task. The capabilities of the CADAM system are so widespread and versatile that it is impossible to dictate a mode of use. Detailed drawings from any model can be picked in a matter of seconds, making the system well suited for layout design. By combining user design skills with the power of the computer, it is possible to solve a variety of engineering problems much faster and with greater accuracy. The CADAM system does much of the drafting work, freeing the user from the repetitious job of drawing and redrawing the same objects over and over again.

An interesting and extremely useful feature is the Geometry Interface module which allows the user to access and modify the CADAM data base. This can be accomplished by the following methods:

CREATION of a collection of CADAM elements (a CADAM model) and adding it to the collection of CADAM models.

RETRIEVAL from a collection of CADAM models for use in the user application program.

Each geometry interface function is separate and distinct. There are 4 geometry interface programs. They are listed below with a brief description of the modules used in this thesis work.

LOFT takes curve and surface shape definitions and assembles them into a CADAM model which can be stored to the CADAM data base.

CADET is a collection of CALL statements which calls passive subroutines that are designed to receive disassembled CADAM elements.

CADCD provides a collection of subroutines which are driven by CALL statements to produce CADAM elements which can be added as a CADAM model to a drawing file. Using CADCD it is possible to put new drawings in the data base, make additions to existing drawings or to add details in a batch mode. All the user needs to do is to write an application program using the subroutines and link it together with the CADCD module.

CADMACGM is a collection of subroutines that are driven by CALL statements to produce CADAM elements. In fact it is an enhanced version of the CADCD module and contains all CADCD subroutine calls except data management.

All parameters in the calling sequence in CADCD and CADMACGM are either REAL*4 or INTEGER*4 and follow default FORTRAN naming conventions. It must be noted that CADCD and CADMACGM use blank common. Hence the user must be extremely careful while passing any information to these module subroutines. Since the blank common will override information passed, the user must not make use of blank common in his application program. An useful variation of using the CADCD subroutine is Macro Geometry.

Macro Geometry allows the user to build load modules from the CADCD subroutines to be executed from the CADAM scope. This capability is created by linking the CADMACGM module (instead of CADCD) with the user developed application program. The program is accessed via the /MACRO GEO/ menu option of the function key DETAIL for the interactive CADAM system. This displays a list of available application programs and the

required parameters whose program names, variable lists and/or menus were previously defined by an application programmer through the use of the CADPARMC assembler routine.

## 3.3 MOVIE.BYU

In 1966 the Computer Science department at the University of Utah, under the direction of David Evans, began to develop a hardware and software capability in graphics. The main thrust of the research work was in the area of continuous tone image generation. By 1968 Hank Christiansen had joined with Evans to explore engineering applications for shaded pictures. This led to the development of MOVIE.BYU, a general purpose computer graphics software system. The MOVIE.BYU system requires a time sharing digital computer with a word length of at least 32 bits. The system has been distributed by Brigham Young University since 1976.

The original display programs of MOVIE.BYU were written at the University of Utah and used to produce raster driven displays with the option of a visible surface processor to solve the hidden surface problem in hardware. This problem was solved with the availability of Watkin's algorithm (24) which was optionally called by the display program if the hardware processor was down. The development of new editions of the program led to significant improvement in the line drawing capabilities of the routines. This system was distributed under the name MOVIE.NUC. Research by Mike Stephenson and Christiansen at Utah produced an expanded capability for MOVIE as well as refinement of data editing routines, character generators, and the capability to process 3-D finite element

models. These developments were incorporated into the MOVIE.NUC program thus producing the 1976 edition of MOVIE.BYU. Since then revisions have been made each year.

### 3.3.1 CAPABILITIES OF MOVIE

The elements of the MOVIE system ( DISPLAY, UTILITY, SECTION, TITLE, MOSAIC and COMPOSE ) are FORTRAN programs for display and manipulation of data representing mathematical, topological and architectural models whose geometries may be described in terms of polygonal elements or contour line definitions. These six programs work in harmony to provide displays of the data in line drawing or continuous tone image format, to clip and cap 3-D systems to expose internal surfaces, to modify geometry, displacement, and/or scalar function files by way of correction, appendage, or symmetry operation, to generate new models or title representations, to convert complex contour line definitions into polygonal element mosaics, or to view multiple models simultaneously. A brief description of each program will help the user to understand their interrelationships.

DISPLAY is the heart of the system. It is the display program for the polygonal element models. The executable program consists of a minimum of six source files compiled and loaded together. The program allows the model to be translated in any direction and rotation about any of the global or local cartesian axes using rigid body commands. The capability to select color for the background and individual parts and the ability

to shade give the system an excellent capacity to produce a smooth surface simulation. Animated sequences involving the scene manipulation commands allow the specification of harmonic oscillations and incremental or smooth animation of the rigid body motion. The program also allows casting of shadows from multiple light sources, transparency, fog simulation and anti-aliasing.

UTILITY is the data generation and editing program which allows the user to produce and/or edit models of 2-D or 3-D polygonal systems. The program allows the generation of 2 and 3-D finite element models in shell or solid element format. The user may also use UTILITY to read data files for the purpose of modification, appendage to other files or to subject the model to symmetry operations.

SECTION is a special purpose program to modify solid data representations so that they are compatible with the display program. The algorithm also allows the repeated dividing of a solid model along a arbitrary set of planes and user defined curves in space, generation of elements on the cut surface and the deletion of all elements and corresponding coordinates, displacements and scaler functions that are interior to the model.

TITLE is a program to generate 2 and 3-D characters whose data format is the same as that used by other programs. The user enters a line of text which is converted into characters composed of polygons according to specifications given by the user.

MOSAIC is a program for processing complex contours to produce element mosaics which are suitable for line drawing and continuous tone display. MOSAIC includes node thinning and selective reduction of triangular pairs to quadrilaterals.

COMPOSE is a program which, in combination with a "Record" command in DISPLAY, allows the user to create multiple image line drawings. Several line drawings are saved on disk as named ASCII files before the user runs the COMPOSE program which retrieves the files and builds up multiple image displays in an automatic or manual mode.

# CHAPTER 4 DESIGN CONSIDERATION

## 4.1 INTRODUCTION

It is almost impossible to produce a cam which can meet all its functional requirements. Since present day requirements usually dictate that components move in a prescribed, exact path, it is important to have a cam which satisfies the displacements exactly. In most of the cases the designer is confronted with a set of requirements which must be satisfied. Generally, a cam must be optimized to meet most of its requirements. Thus the cam produced by the optimization technique would be the best one for a given set of conditions. A cam may be designed in two ways:

1. By assuming the required motion for the follower and designing the cam to give this motion.
2. By assuming the shape of the cam and determining what characteristics of displacement, velocity and acceleration this contour will give.

The first method is a good example of synthesis. In fact, designing a cam mechanism from the desired motion is an application of synthesis that can be solved every time. Hence, in any cam mechanism the first decision always relates to motion specifications. In order to select a simple, optimum cam profile it is necessary to develop a complete timing diagram showing the displacement of the mechanism and its proper spacing and place in the time cycle. A good cam profile is often one that has the lowest

maximum acceleration. If there are other considerations to be met, a compromise must be made with other optimizing parameters. On the other hand, if two or more mechanisms must be synchronized and have their motions closely coordinated, it may be necessary to use a less than optimum cam profile for a part of the total displacement. This brings about the need for careful boundary matching and the blending together of composite profiles.

Very little is available concerning methods of selecting the most suitable profile for a given application and of accurately producing the profile in cam manufacture. This difficulty is eliminated by making the cam symmetrical and by using shapes for cam contours that can be generated easily. Besides motion considerations, a primary concern of the designer must be the type and magnitude of the load between the cam and follower. For low speeds and low masses, the inertia load is low and will cause no appreciable deflection of components. Hence, almost any cam curve can be used. At high speeds (or masses), however, the inertia load increases and the cam mechanism becomes prone to deflection and creates vibrations which will produce vibratory stresses in the follower system. Now the cam profile becomes of utmost importance. Figure 1 displays the flow chart for the basic approach for cam design.

## 4.2 CAM DISPLACEMENT CURVES

There are infinitely many follower motions that can be used for the rise and return of the follower. There are a number of so called "basic curves" like Constant Velocity or Straight Line, Constant Acceleration

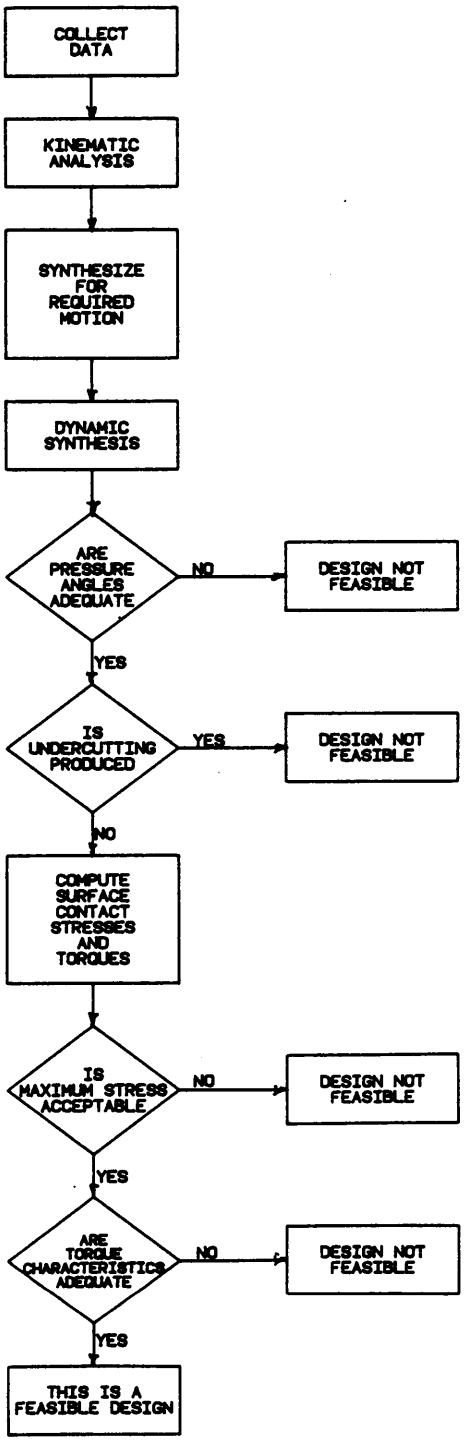


Figure 1. Flow chart for basic approach for Cam design

or Parabolic and trigonometric curves like Simple Harmonic, Cycloidal and Elliptical curves. These basic curves may be inadequate on some occasions, especially in high speed application. An alternate, versatile manner of motion specification is the use of polynomial function. Solutions for the basic curves and low-order power polynomials can be easily obtained. However, as the conditions of control become complex, the order of the polynomial increases and the designer runs into numerical difficulties. In an effort to incorporate the better features of the basic curves into one optimum curve, designers have tried many combinations of basic curves. Composite curves like Trapezoidal and Fourier Series curves have been formed to produce minimum acceleration of (and hence, minimum dynamic load on) the follower. Certain fundamental conditions must be satisfied while combining curves. It should be noted that any combination of the basic curves may be utilized to fulfill the requirements of the fundamental conditions. The following conditions must be satisfied:

1. The sum of the displacements during each of the various motions should be equal to the total stroke.
2. The sum of the angles turned through by the cam during each of the various motions should be equal to the total cam angle.
3. The velocities of all curves at the junction should be equal.
4. High-speed action requires that acceleration of all curves at the junction be equal.

Based on the fundamental conditions of combining cam curves, a customized cam can be developed with a predesigned building block of cam curve seg-

ments. This thesis considers 22 different motion curves. A brief description is provided for each type of curve.

### 4.2.1 SIMPLE HARMONIC MOTION

This motion is characterized by having its acceleration proportional and in an opposite direction to its displacement. The usefulness of these curves is further expanded by using the different forms of SHM as curve segments to form a single contour. If L is the total lift, $\beta$ the active cam angle and $\theta$ the angle of rotation of the cam then the motion characteristics of the curve segments are represented by equations 1 through 18.

H-1 Harmonic: As seen from Fig. 2 this is a lifting curve which has a constant velocity at the end of the stroke.

$$s = L\left\{1 - \cos\left(\frac{\pi\theta}{2\beta}\right)\right\} \tag{1}$$

$$v = \frac{\pi L}{2\beta} \sin\left(\frac{\pi\theta}{2\beta}\right) \tag{2}$$

$$a = \frac{\pi^2 L}{4\beta^2} \cos\left(\frac{\pi\theta}{2\beta}\right) \tag{3}$$

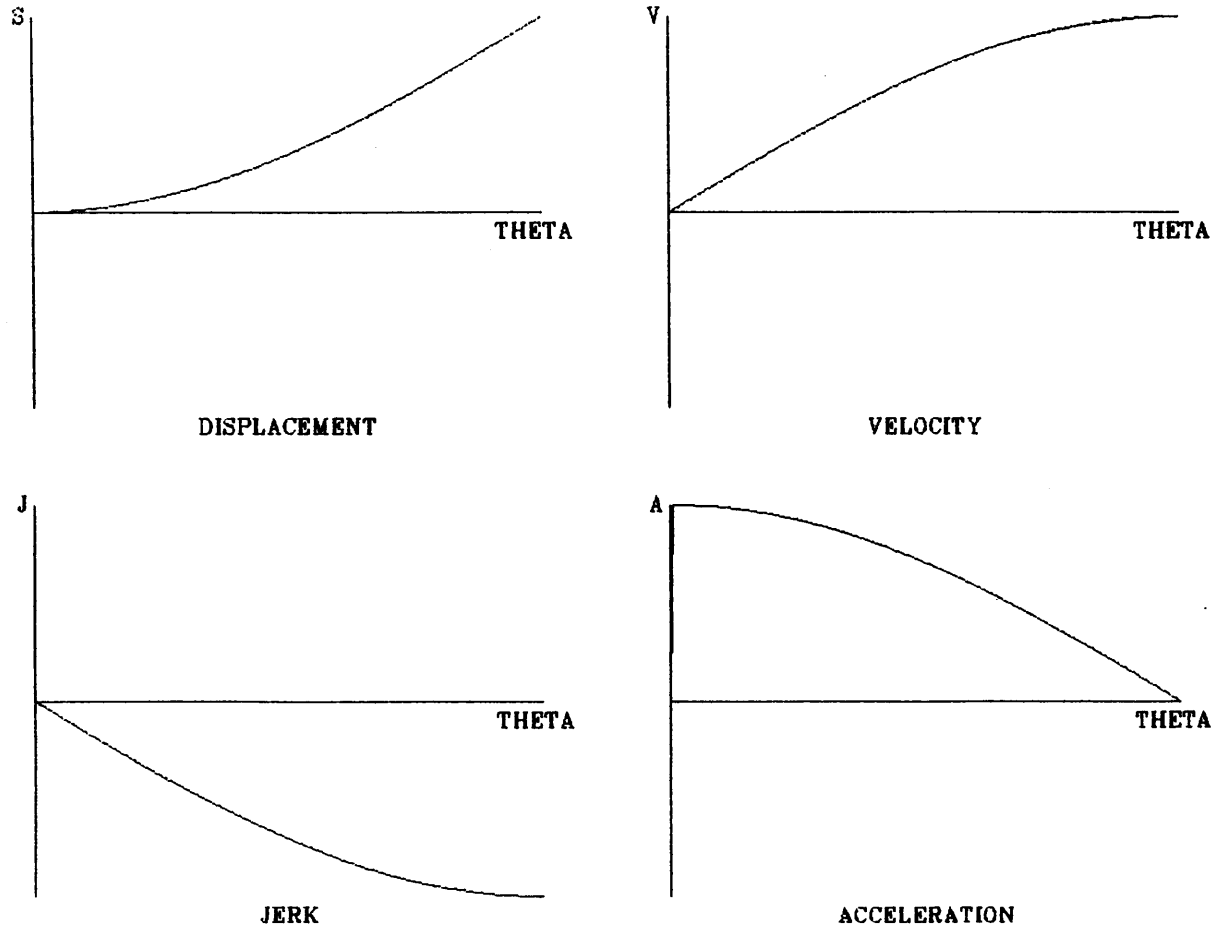


Figure 2. Motion Characteristics for H-1 curve

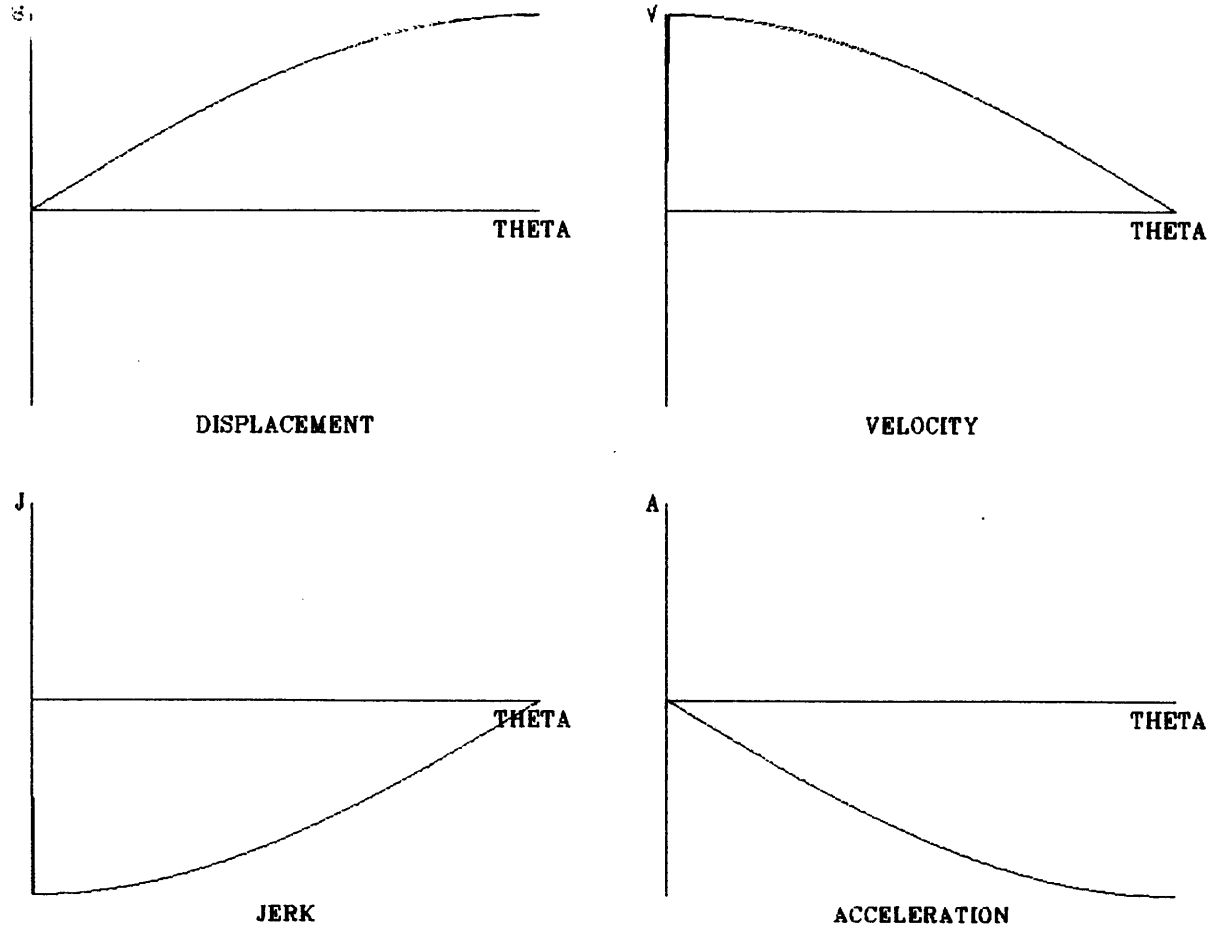


Figure 3. Motion Characteristics for H-2 curve

H-2 Harmonic: In the case of the H-2 lift curve the velocity goes to zero at the end of the stroke. However the acceleration at this point is not zero. The characteristics are as shown in Fig. 3.

$$s = L \sin\left(\frac{\pi\theta}{2\beta}\right) \tag{4}$$

$$v = \frac{\pi L}{2\beta} \cos\left(\frac{\pi\theta}{2\beta}\right) \tag{5}$$

$$a = -\frac{\pi^2 L}{4\beta^2} \sin\left(\frac{\pi\theta}{2\beta}\right) \tag{6}$$

H-3 Harmonic: H-3 is a descending curve and has properties symmetric to that of H-2. Hence it would be an ideal curve for combination with that curve. Figure 4 shows the motion characteristics for this curve segment.

$$s = L \cos\left(\frac{\pi\theta}{2\beta}\right) \tag{7}$$

$$v = -\frac{\pi L}{2\beta} \sin\left(\frac{\pi\theta}{2\beta}\right) \tag{8}$$

$$a = -\frac{\pi^2 L}{4\beta^2} \cos\left(\frac{\pi\theta}{2\beta}\right) \tag{9}$$

H-4 Harmonic: While this is a descending curve too, its characteristics, as seen from Fig. 5, are different from that of H-3.

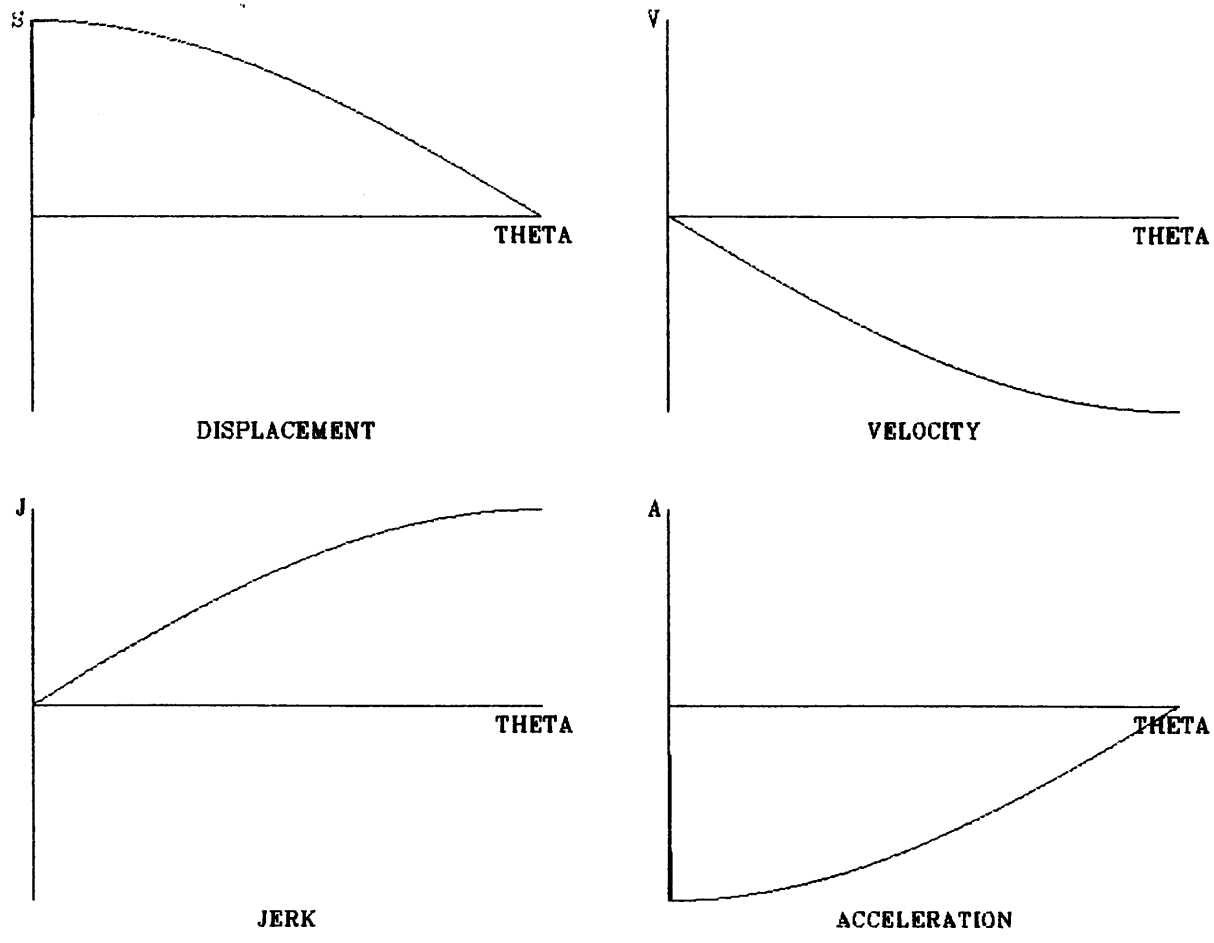


Figure 4. Motion Characteristics for H-3 curve

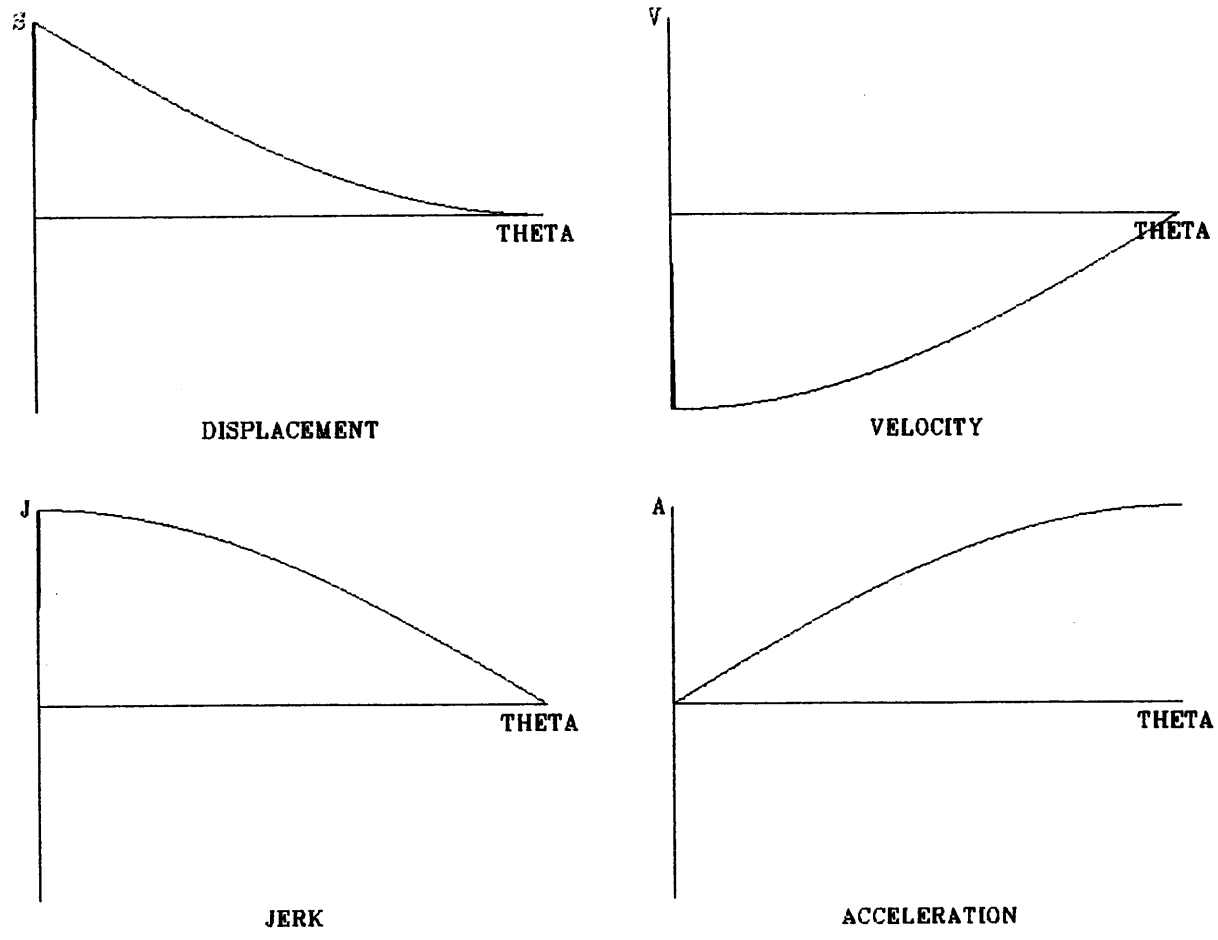


Figure 5. Motion Characteristics for H-4 curve

$$s = L\left(1 - \sin\left(\frac{\pi\theta}{2\beta}\right)\right) \tag{10}$$

$$v = -\frac{\pi L}{2\beta}\cos\left(\frac{\pi\theta}{2\beta}\right) \tag{11}$$

$$a = \frac{\pi^2 L}{4\beta^2}\sin\left(\frac{\pi\theta}{2\beta}\right) \tag{12}$$

H-5 Harmonic: H-5 curve is the most frequently used harmonic curve. This is because it produces a smooth transition point for the combining curve. H-5 characteristics are shown in Fig. 6.

$$s = \frac{L}{2}\left(1 - \cos\left(\pi\frac{\theta}{\beta}\right)\right) \tag{13}$$

$$v = \frac{\pi L}{2\beta}\sin\left(\pi\frac{\theta}{\beta}\right) \tag{14}$$

$$a = \frac{\pi^2 L}{2\beta^2}\cos\left(\pi\frac{\theta}{\beta}\right) \tag{15}$$

H-6 Harmonic: The H-6 curve is often used in conjunction with the H-5 curve and has properties symmetric to it. The motion characteristics of H-6 are as shown in Fig. 7.

$$s = \frac{L}{2}\left(1 + \cos\left(\pi\frac{\theta}{\beta}\right)\right) \tag{16}$$

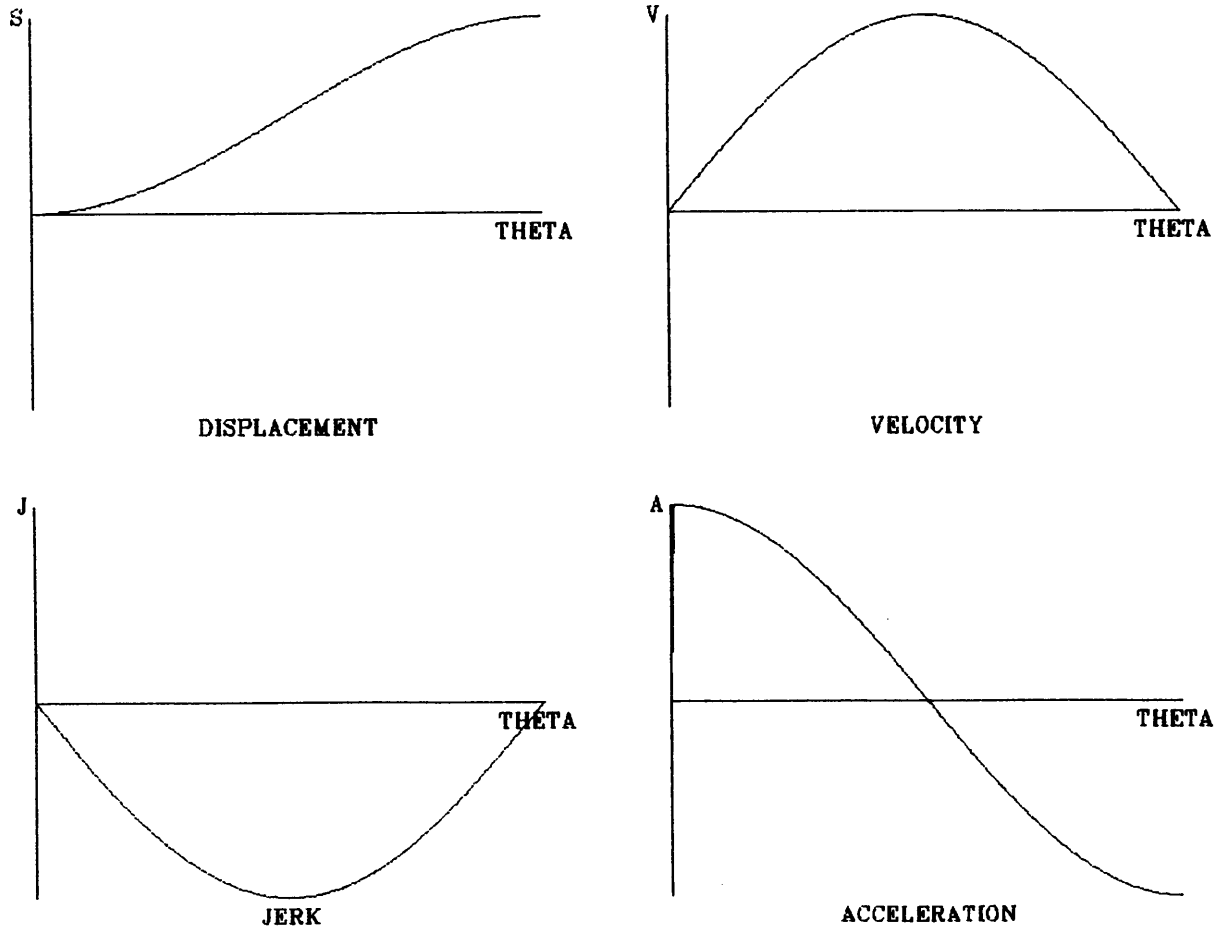


Figure 6. Motion Characteristics for H-5 curve

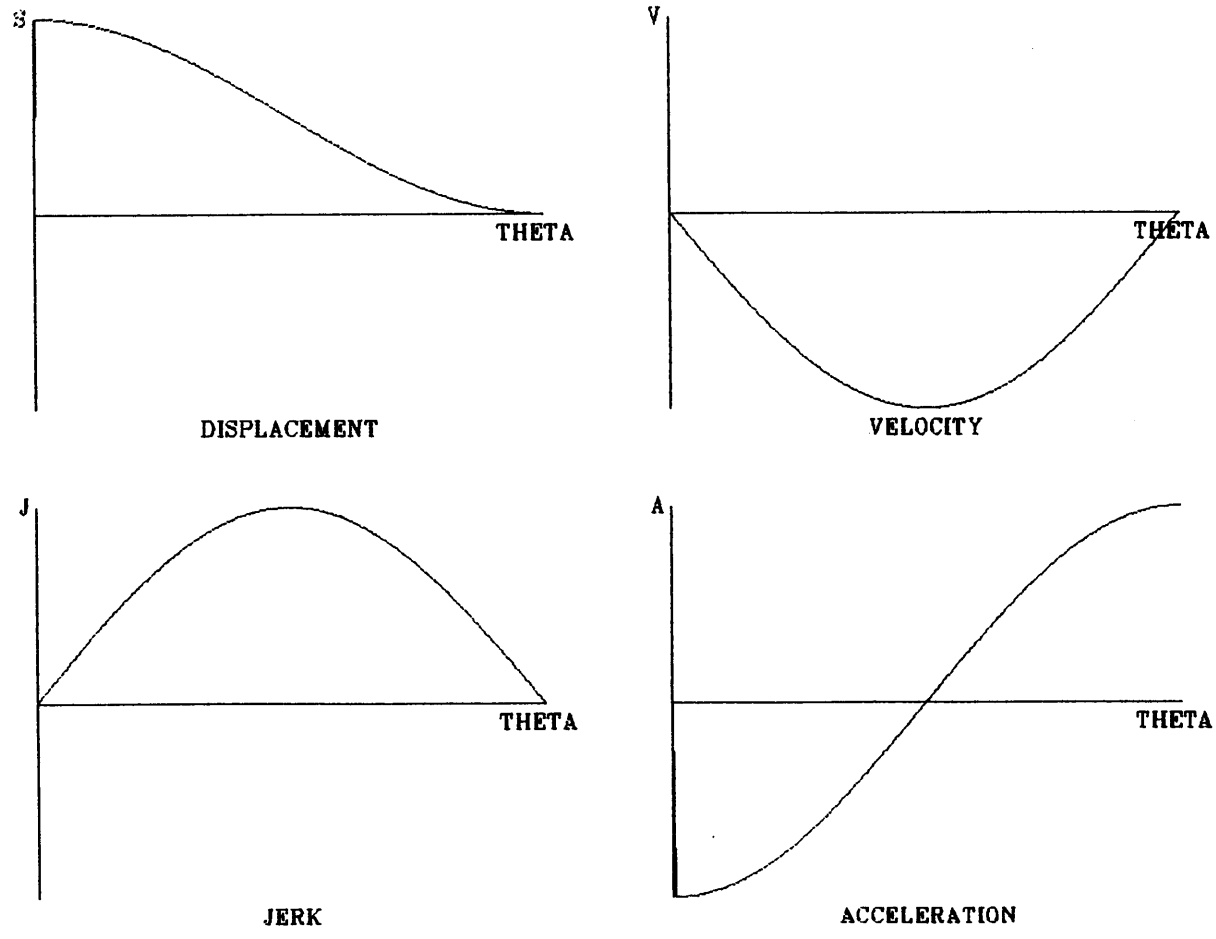


Figure 7. Motion Characteristics for H-6 curve

$$v = -\frac{\pi L}{2\beta} \sin\left(\pi\frac{\theta}{\beta}\right) \tag{17}$$

$$a = -\frac{\pi^2 L}{2\beta^2} \cos\left(\pi\frac{\theta}{\beta}\right) \tag{18}$$

### 4.2.2 CYCLOIDAL MOTION

This motion is obtained by rolling a circle on a straight line and generating the locus of a fixed point on the circumference of the circle. From the generating principle it is easy to write the motion equations for these curve segments (Equations 19 to 36).

C-1 Cycloidal: The motion characteristics of this curve are shown in Fig. 8. As seen from the figure, this curve has characteristics similar to the H-1 curve.

$$s = L\left(\frac{\theta}{\beta} - \frac{1}{\pi} \sin\left(\pi\frac{\theta}{\beta}\right)\right) \tag{19}$$

$$v = \frac{L}{\beta}\left(1 - \cos\left(\pi\frac{\theta}{\beta}\right)\right) \tag{20}$$

$$a = \frac{\pi L}{\beta^2} \sin\left(\pi\frac{\theta}{\beta}\right) \tag{21}$$

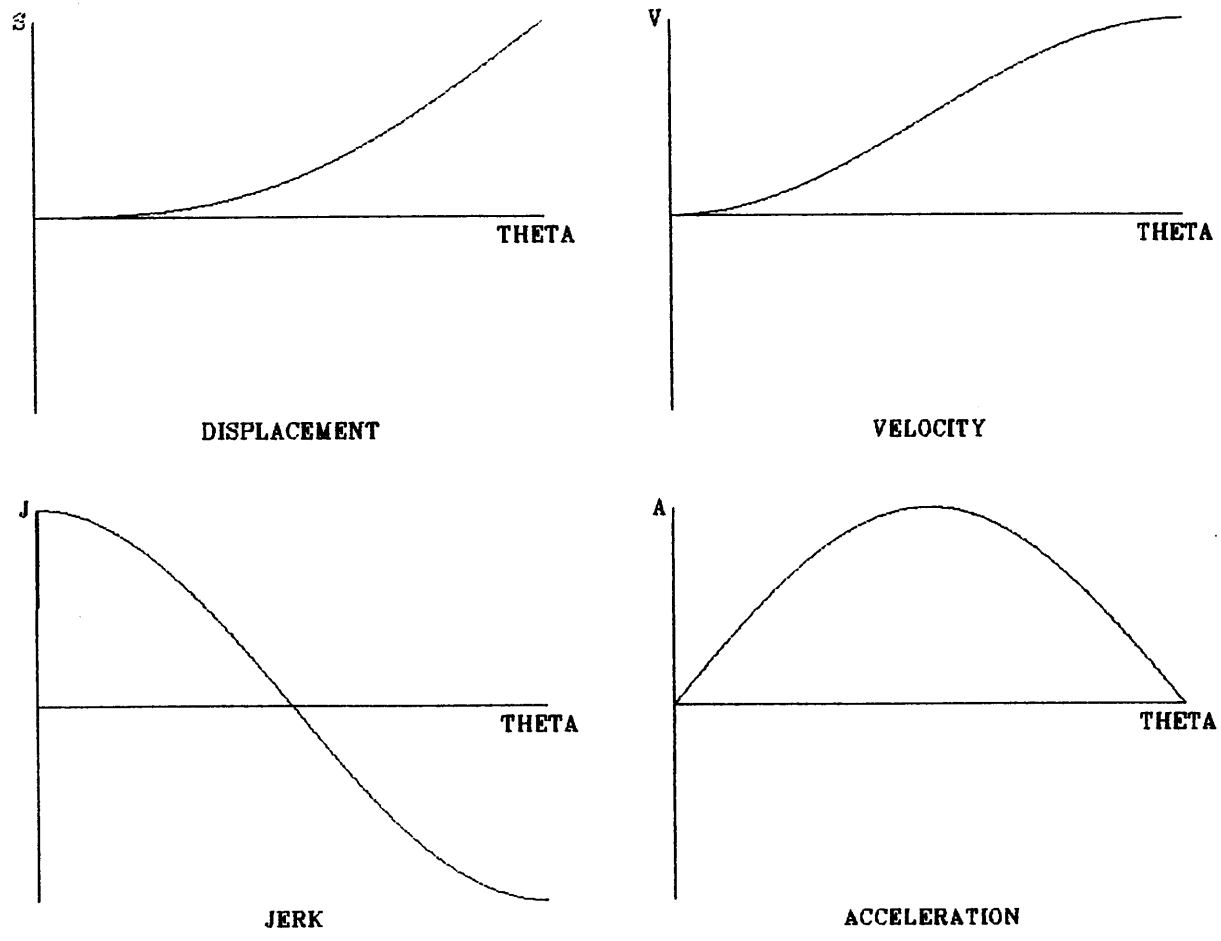


Figure 8. Motion Characteristics for C-1 curve

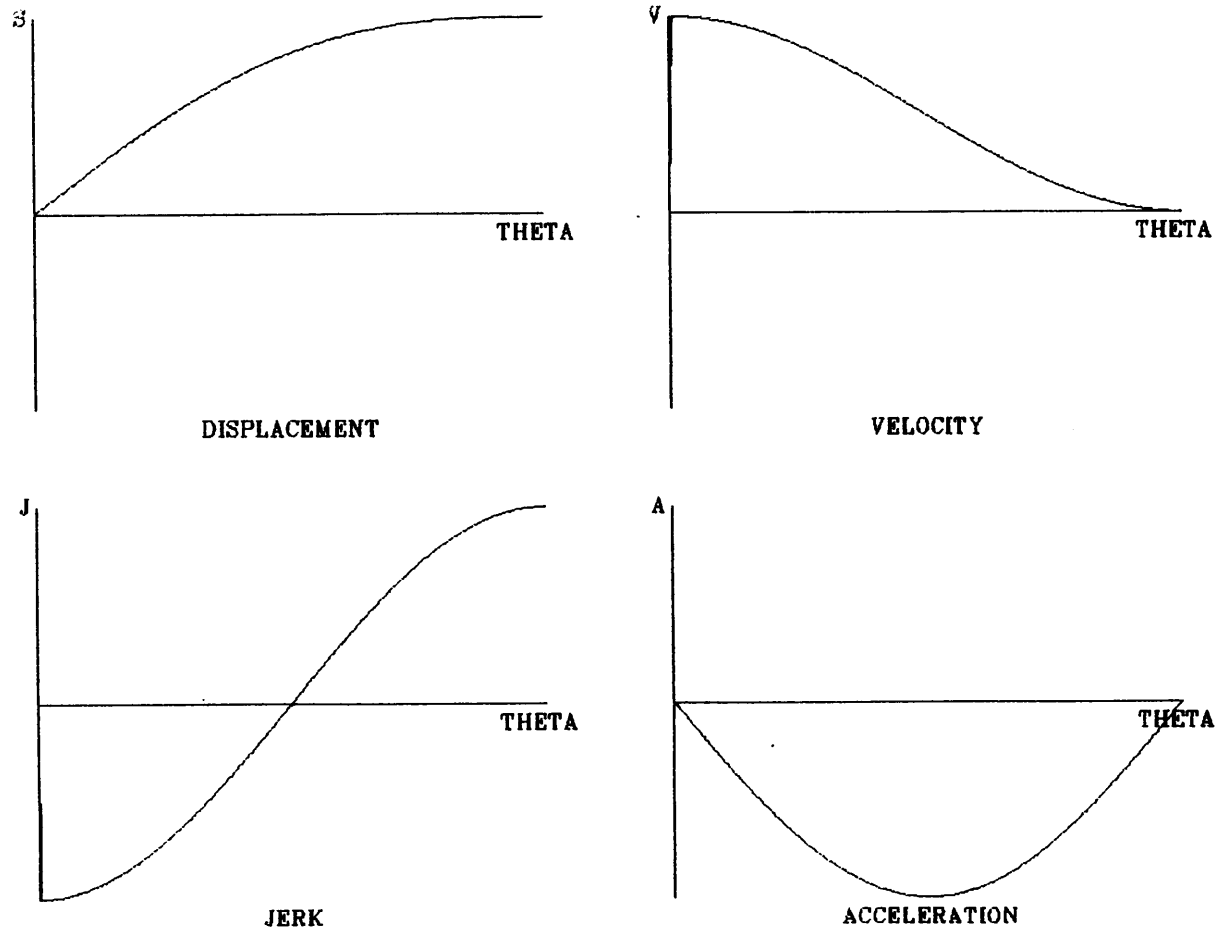


Figure 9. Motion Characteristics for C-2 curve

C-2 Cycloidal: Figure 9 shows the motion characteristics of the C-2 curve. Its velocity and acceleration are symmetric to that of the C-1 curve.

$$s = L\left(\frac{\theta}{\beta} + \frac{1}{\pi}\sin\left(\pi\frac{\theta}{\beta}\right)\right) \tag{22}$$

$$v = \frac{L}{\beta}\left(1 + \cos\left(\pi\frac{\theta}{\beta}\right)\right) \tag{23}$$

$$a = -\frac{\pi L}{\beta^2}\sin\left(\pi\frac{\theta}{\beta}\right) \tag{24}$$

C-3 Cycloidal: The motion characteristics of this curve are shown in Fig. 10.

$$s = L\left(1 - \frac{\theta}{\beta} + \frac{1}{\pi}\sin\left(\pi\frac{\theta}{\beta}\right)\right) \tag{25}$$

$$v = -\frac{L}{\beta}\left(1 - \cos\left(\pi\frac{\theta}{\beta}\right)\right) \tag{26}$$

$$a = -\frac{\pi L}{\beta^2}\sin\left(\pi\frac{\theta}{\beta}\right) \tag{27}$$

C-4 Cycloidal: As Fig. 11 shows, the C-4 curve has charecteristics which are symmetrically opposite to that of C-1 curve.

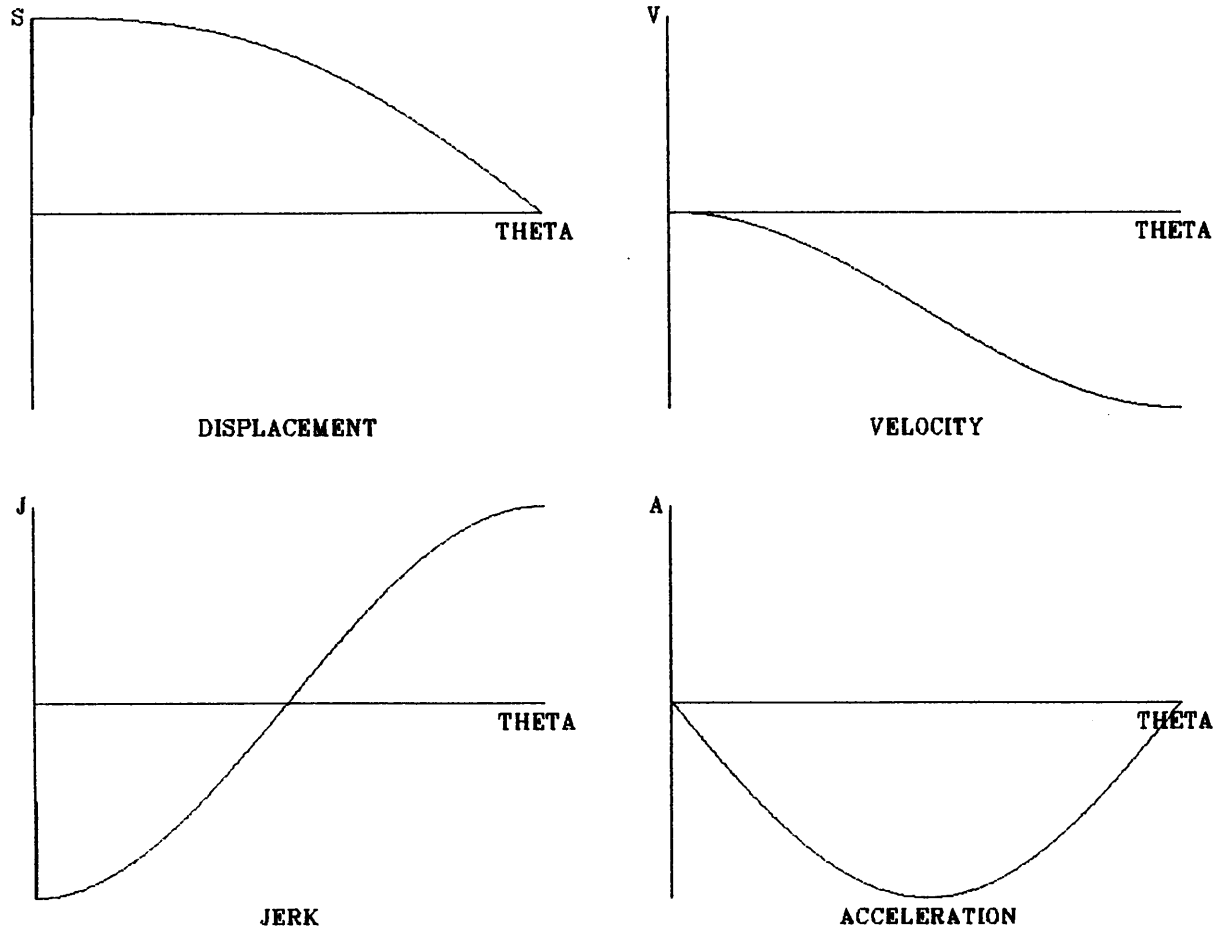


Figure 10. Motion Characteristics for C-3 curve

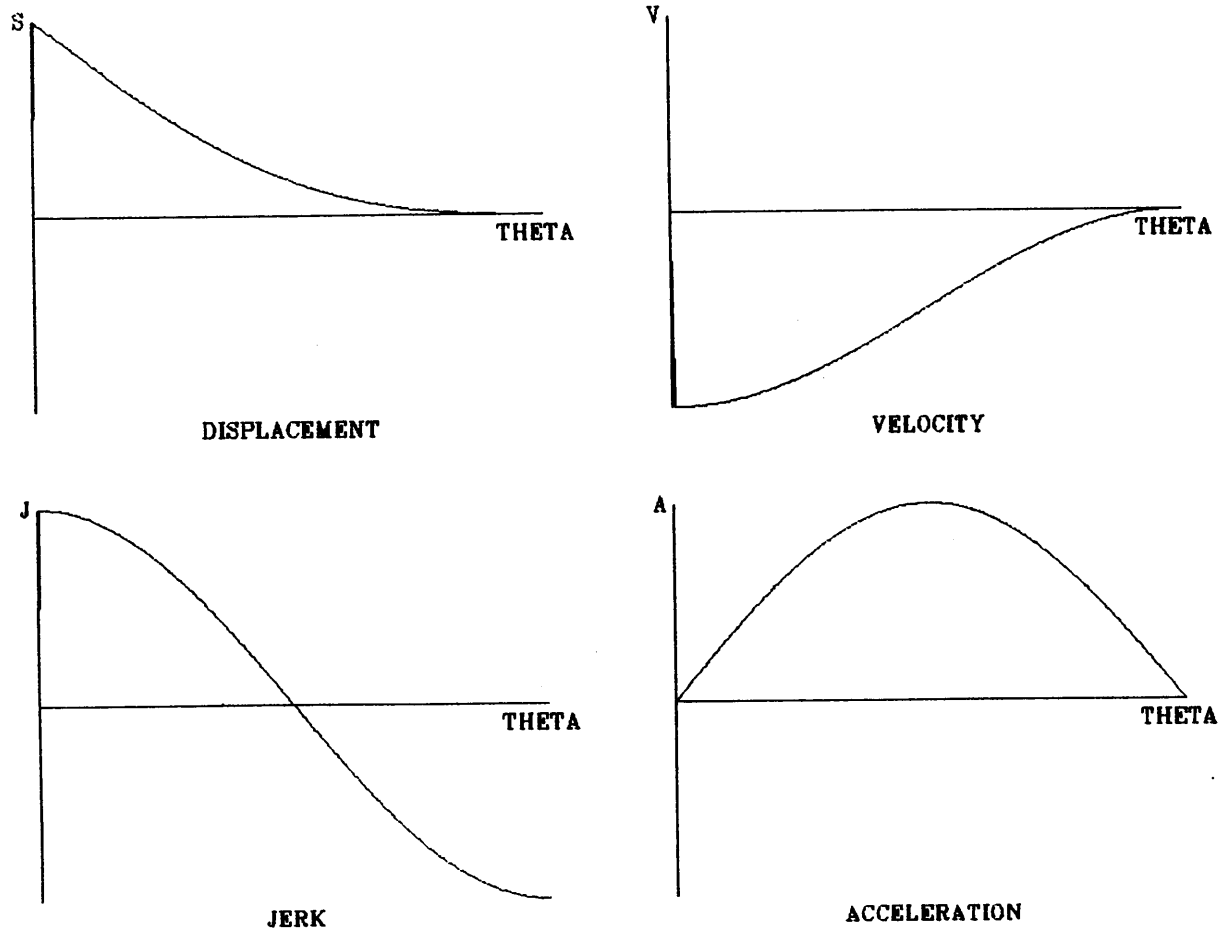


Figure 11. Motion Characteristics for C-4 curve

$$s = L\left(1 - \frac{\theta}{\beta} - \frac{1}{\pi}\sin\left(\pi\frac{\theta}{\beta}\right)\right) \tag{28}$$

$$v = -\frac{L}{\beta}\left(1 + \cos\left(\pi\frac{\theta}{\beta}\right)\right) \tag{29}$$

$$a = \frac{\pi L}{\beta^2}\sin\left(\pi\frac{\theta}{\beta}\right) \tag{30}$$

C-5 Cycloidal: The C-5 curve is used very often in conjunction with the C-6 curve. The combination of these curves gives continuous velocity and acceleration curves. Figure 12 shows the characteristics of the C-5 curve.

$$s = L\left(\frac{\theta}{\beta} - \frac{1}{2\pi}\sin\left(2\pi\frac{\theta}{\beta}\right)\right) \tag{31}$$

$$v = \frac{L}{\beta}\left(1 - \cos\left(2\pi\frac{\theta}{\beta}\right)\right) \tag{32}$$

$$a = \frac{2\pi L}{\beta^2}\sin\left(2\pi\frac{\theta}{\beta}\right) \tag{33}$$

C-6 Cycloidal: Figure 13 shows the characteristics of this curve.

$$s = L\left(1 - \frac{\theta}{\beta} + \frac{1}{2\pi}\sin\left(2\pi\frac{\theta}{\beta}\right)\right) \tag{34}$$

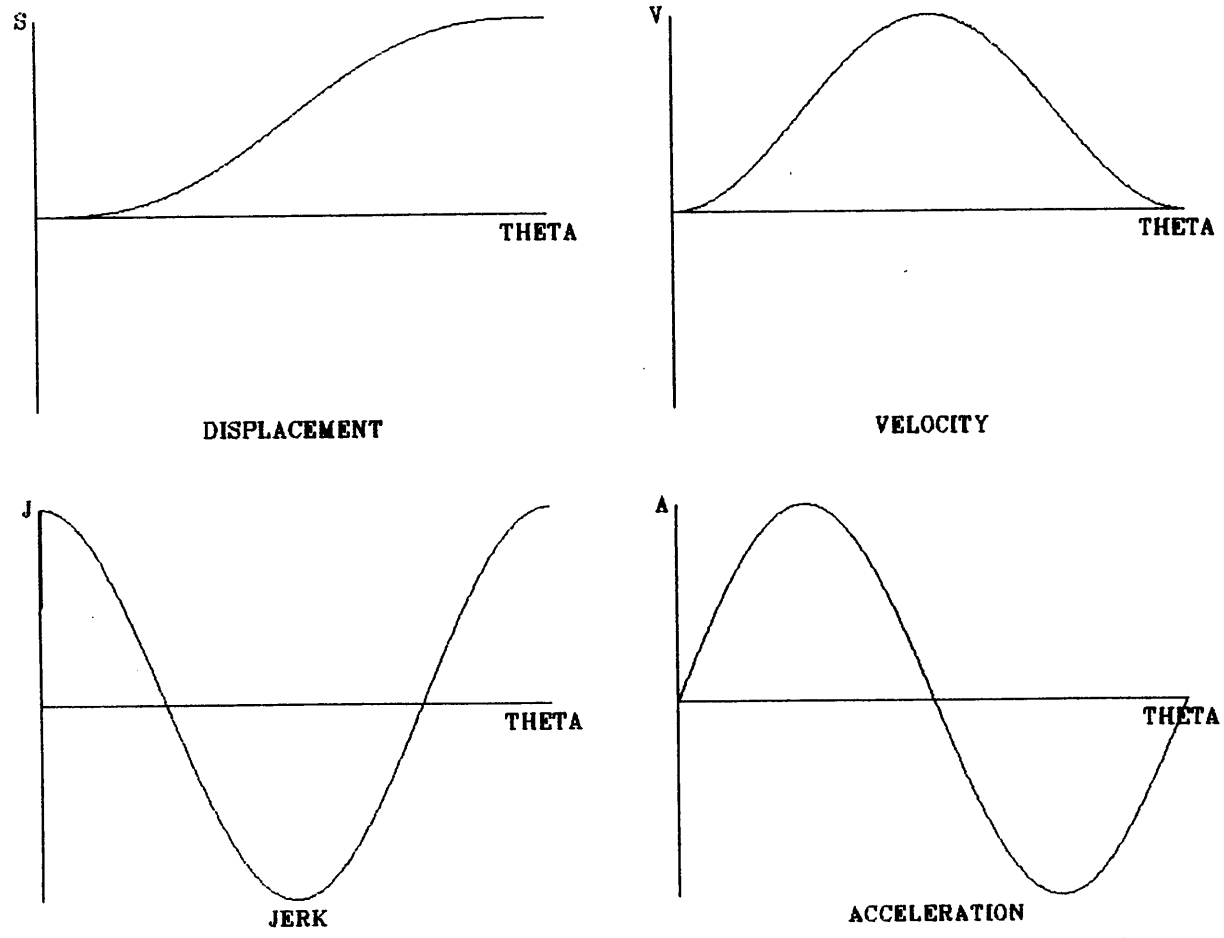


Figure 12. Motion Characteristics for C-5 curve

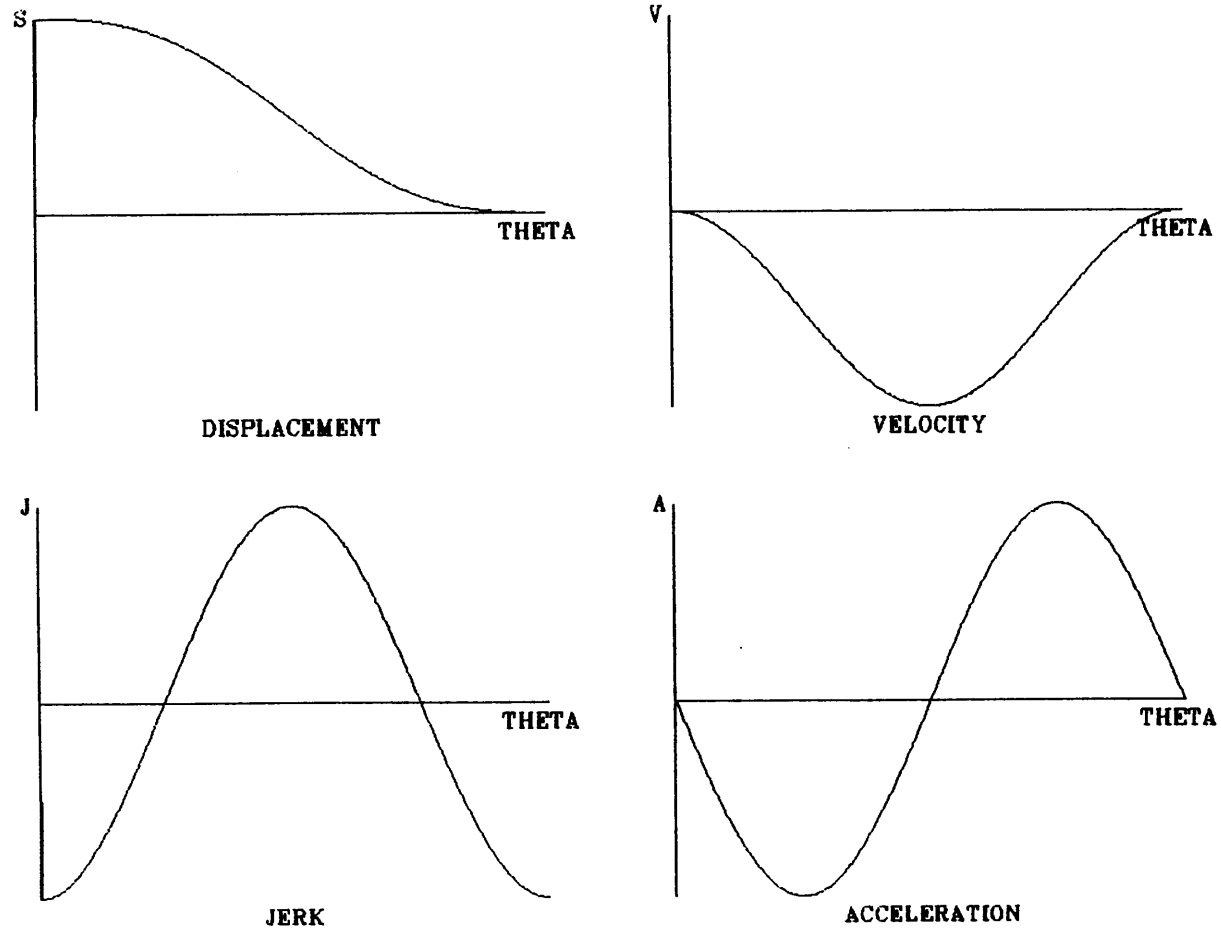


Figure 13. Motion Characteristics for C-6 curve

$$v = -\frac{L}{\beta}\left(1 - \cos\left(2\pi\frac{\theta}{\beta}\right)\right) \tag{35}$$

$$a = -\frac{2\pi L}{\beta^2}\sin\left(2\pi\frac{\theta}{\beta}\right) \tag{36}$$

As there are no sudden changes in acceleration at the dwell ends, the cycloidal cam is has the smoothest motion among all basic curves, and is therefore suitable for high-speed applications.

### 4.2.3 CONSTANT VELOCITY

This curve has the simplest form of the polynomial (or a straight-line) equation. If s is the follower displacement at a cam angle θ and $C_o$ and $C_1$ are constants then this form can be represented as,

$$s(\theta) = C_o + C_1\theta \tag{37}$$

Applying the end conditions to equation 37 we get,

$$s = \frac{h}{\beta}\theta \tag{38}$$

where, h is the total rise during cam rotation β. The equations for velocity, acceleration and jerk follow from this equation.

$$v = \frac{h}{\beta} \tag{39}$$

$$a = 0 \tag{40}$$

Motion characteristics for constant velocity are shown in Fig. 14.

### 4.2.4 CONSTANT ACCELERATION

This curve, also known as parabolic or gravity curve, has constant positive and negative accelerations. The curve may have different displacement functions on either side of the point of inflection. If the inflection point occurs midway through the stroke, then, the motion equations are

For $0 \leq \theta \leq \beta/2$

$$s = \frac{2h}{\beta^2}\theta^2 \tag{41}$$

$$v = \frac{4h}{\beta^2}\theta \tag{42}$$

$$a = \frac{4h}{\beta^2} \tag{43}$$

For $\beta/2 \leq \theta \leq \beta$

$$s = h - \frac{2h}{\beta^2}(\beta - \theta)^2 \tag{44}$$

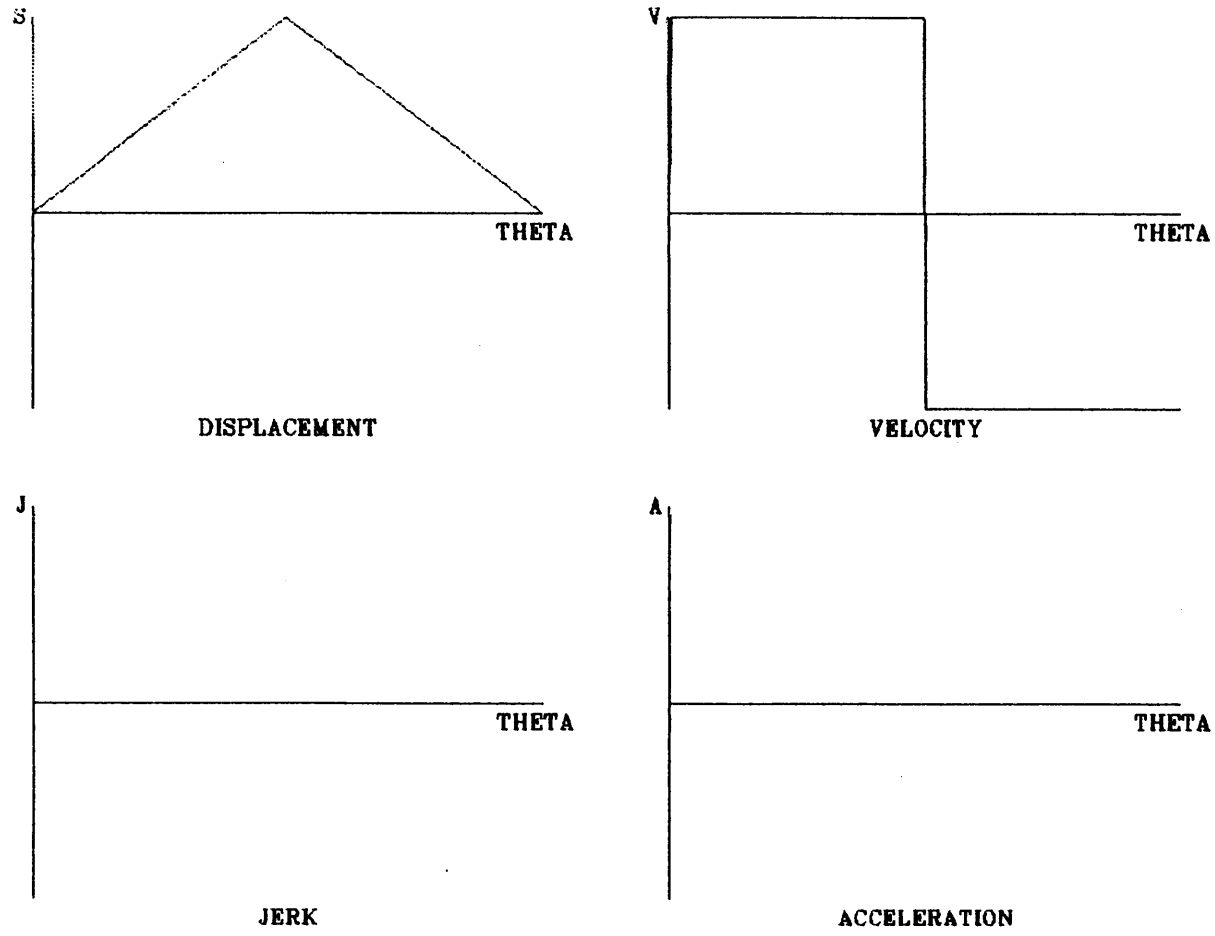


Figure 14. Motion Characteristics for Constant Velocity Curve

$$v = \frac{2h}{\beta}(\beta - \theta) \tag{45}$$

$$a = -\frac{2h}{\beta^2} \tag{46}$$

In both cases, the jerk is zero, except where changes in acceleration occur, in which case jerk is infinite. Figure 15 shows the motion characteristics for this type of motion.

### 4.2.5 POLYNOMIAL CURVES

As the boundary conditions of the motion increase it becomes necessary to use higher degree polynomial equations. Equation 47 shows an nth degree polynomial equation which can be used to satisfy any number of boundary conditions.

$$s = C_o + C_1\theta + C_2\theta^2 + ..... + C_n\theta^n \tag{47}$$

This technique can produce smooth and aesthetically acceptable profiles that fulfill design requirements, although the mathematics involved may sometimes be quite cumbersome. For the sake of convenience, we shall normalize both, the total rise (h) and the active cam angle ($\beta$) i.e. $h = 1$ and $\beta = 1$. The 2-3, 3-4-5, and 4-5-6-7 polynomial curves fall under this category. Figures 16, 17, and 18 represent the normalized motion characteristics of the three polynomials respectively. The normalized motion equations are represented by equations 48 through 56.

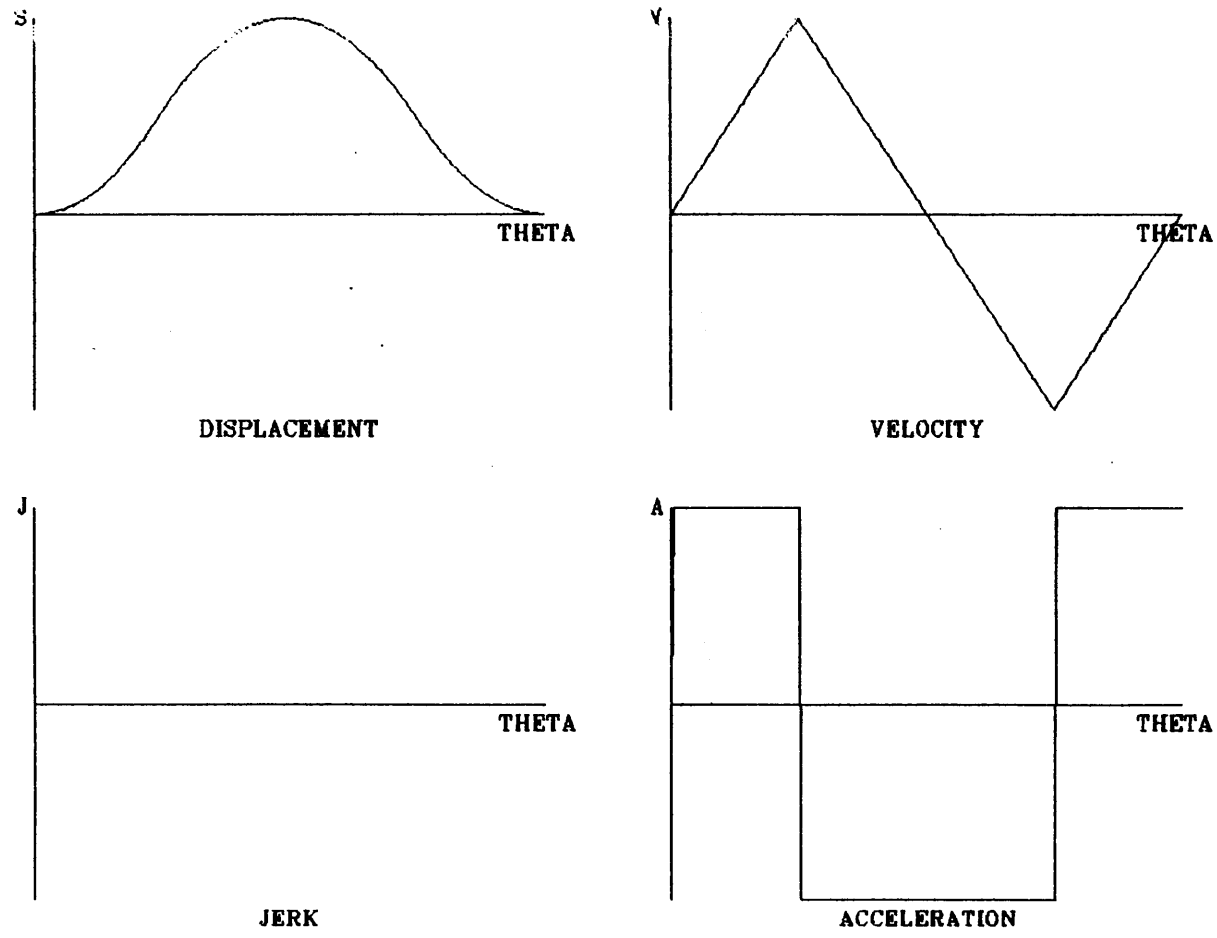


Figure 15. Motion Characteristics for Constant Acceleration Curve

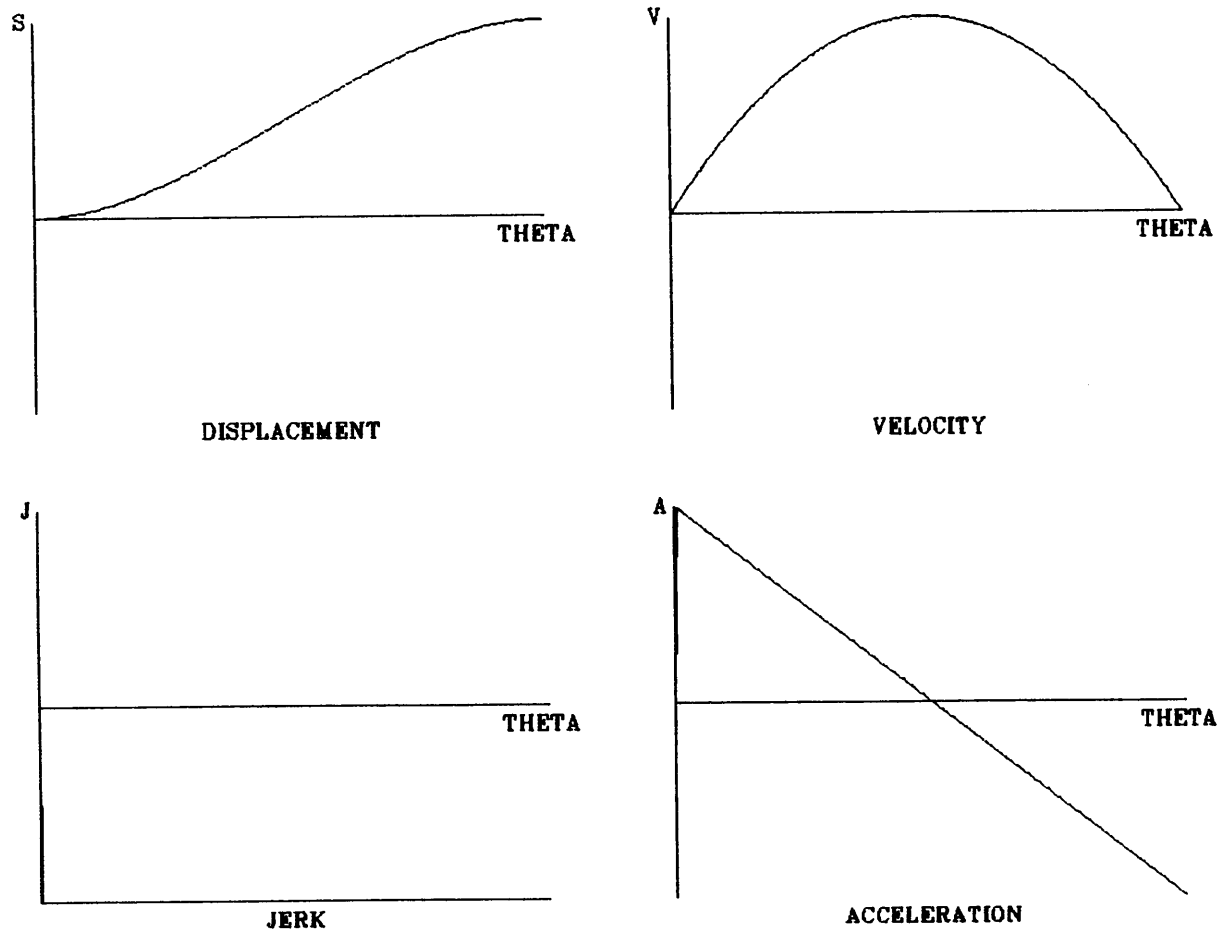


Figure 16. Motion Characteristics for normalized 2-3 Polynomial Curve

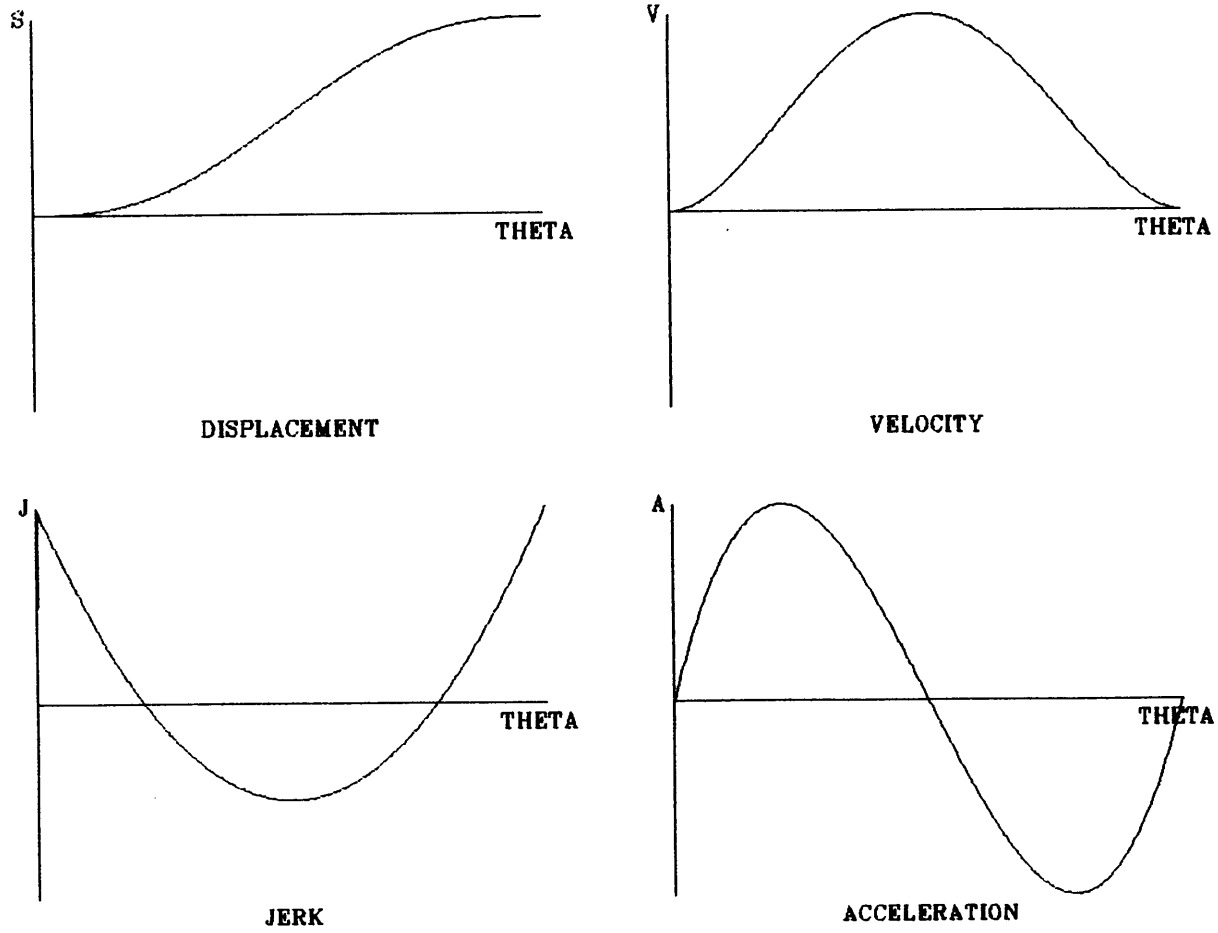


Figure 17. Motion Characteristics for normalized 3-4-5 Polynomial Curve

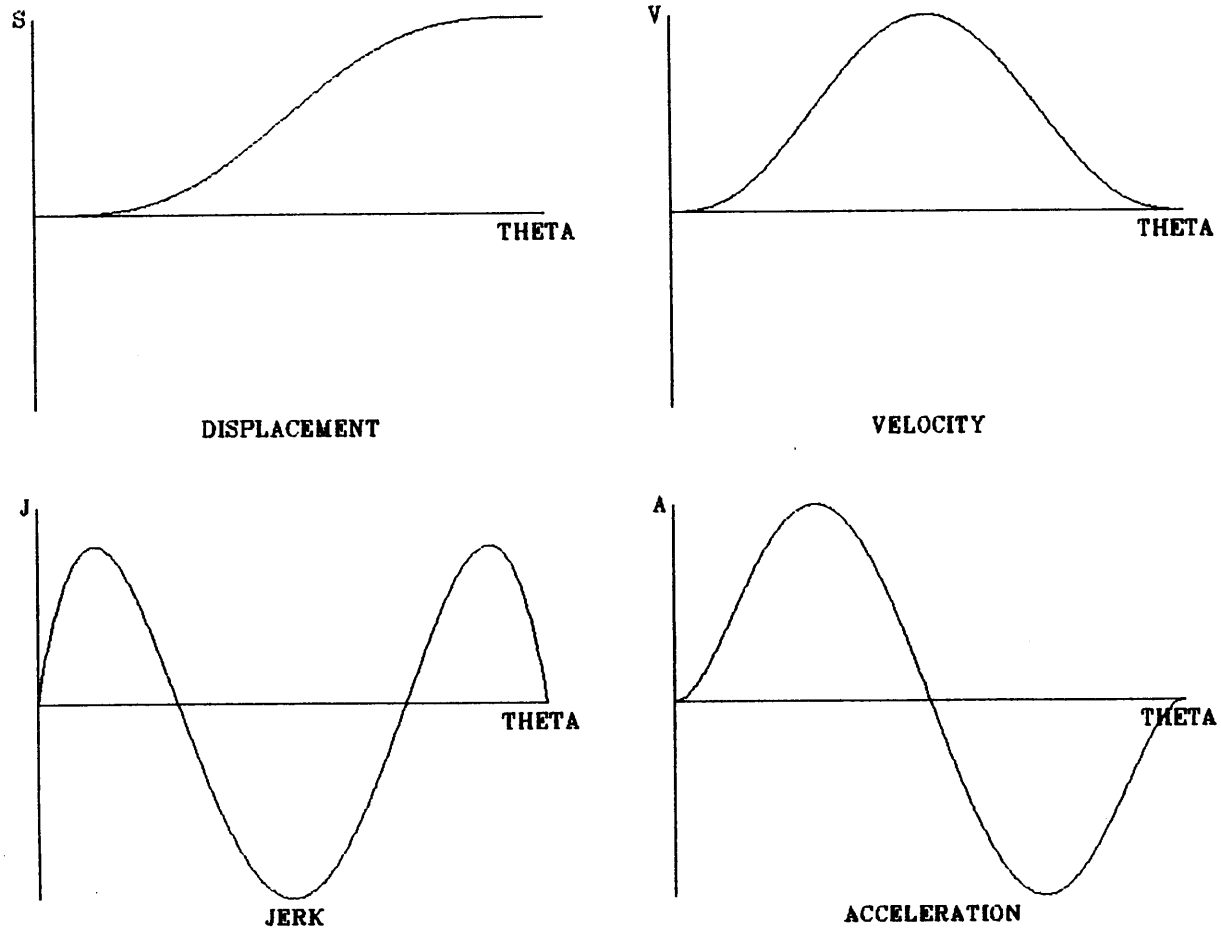


Figure 18. Motion Characteristics for normalized 4-5-6-7 Polynomial Curve

2-3 Polynomial

$$s = 3\theta^2 - 2\theta^3 \tag{48}$$

$$v = 6\theta - 6\theta^2 \tag{49}$$

$$a = 6 - 12\theta \tag{50}$$

3-4-5 Polynomial

$$s = 10\theta^3 - 15\theta^4 + 6\theta^5 \tag{51}$$

$$v = 30\theta^2 - 60\theta^3 + 30\theta^4 \tag{52}$$

$$a = 60\theta - 180\theta^2 + 120\theta^3 \tag{53}$$

4-5-6-7 Polynomial

$$s = 35\theta^4 - 84\theta^5 + 70\theta^6 - 20\theta^7 \tag{54}$$

$$v = 140\theta^3 - 420\theta^4 + 420\theta^5 - 140\theta^6 \tag{55}$$

$$a = 420\theta^2 - 1680\theta^3 + 2100\theta^4 - 840\theta^5 \tag{56}$$

### 4.2.6 FOURIER SERIES CURVES

There are certain disadvantages associated with the parabolic and cycloidal curves. They are likely to induce resonance vibrations or high acceleration. Any general purpose cam can be optimized with a curve-intermediate between the parabolic and the cycloidal, but with the fewest and the lowest possible harmonics. Based on the above several harmonic series have been proposed. The kinematic equations of three are given below:

Gutman 1-3 Harmonic(90): This curve is obtained from the Fourier series expansion of the parabolic curve by retaining the first two terms of the series. Using standard notations, the motion equations can be represented as follows:

$$s = h\left(\frac{\theta}{\beta} - \frac{15}{32\pi}\sin 2\pi\frac{\theta}{\beta} - \frac{1}{96\pi}\sin 6\pi\frac{\theta}{\beta}\right) \tag{57}$$

$$v = \frac{h}{\beta}\left(1 - \frac{15}{16\pi}\cos 2\pi\frac{\theta}{\beta} - \frac{1}{16\pi}\cos 6\pi\frac{\theta}{\beta}\right) \tag{58}$$

$$a = \frac{h\pi}{8\beta^2}\left(15\sin 2\pi\frac{\theta}{\beta} + 3\sin 6\pi\frac{\theta}{\beta}\right) \tag{59}$$

Figure 19 shows the motion characteristics for this curve.

Freudenstein 1-3 Harmonic(84): This curve has been developed to minimize the dynamic acceleration factor. The equations are

$$s = h\left(\frac{\theta}{\beta} - \frac{h}{2\pi}\ \frac{27}{28}\ \sin\ 2\pi\frac{\theta}{\beta} + \frac{1}{84}\ \sin\ 6\pi\frac{\theta}{\beta}\right) \tag{60}$$

$$v = \frac{h}{\beta}\left(1 - \frac{27}{28}\ \cos\ 2\pi\frac{\theta}{\beta} - \frac{1}{28}\ \cos\ 6\pi\frac{\theta}{\beta}\right) \tag{61}$$

$$a = \frac{2\pi h}{\beta^2}\left(\frac{27}{28}\ \sin\ 2\pi\frac{\theta}{\beta} + \frac{3}{28}\ \sin\ 6\pi\frac{\theta}{\beta}\right) \tag{62}$$

The motion functions are shown in Fig. 20.

Freudenstein 1-3-5 Harmonic(84): Corresponding equations for this curve are

$$s = \frac{h\theta}{\beta} - \frac{hm}{2\pi}\left(\sin\ 2\pi\frac{\theta}{\beta} + \frac{1}{54}\ \sin\ 6\pi\frac{\theta}{\beta} + \frac{1}{1250}\ \sin\ 10\pi\frac{\theta}{\beta}\right) \tag{63}$$

$$v = \frac{h}{\beta}\left(1 - m\left(\cos\ 2\pi\frac{\theta}{\beta} + \frac{1}{18}\ \cos\ 6\pi\frac{\theta}{\beta} + \frac{1}{250}\ \cos\ 10\pi\frac{\theta}{\beta}\right)\right) \tag{64}$$

$$a = \frac{2\pi h}{\beta^2}\left(m\left(\sin\ 2\pi\frac{\theta}{\beta} + \frac{1}{6}\ \sin\ 6\pi\frac{\theta}{\beta} + \frac{1}{50}\ \sin\ 10\pi\frac{\theta}{\beta}\right)\right) \tag{65}$$

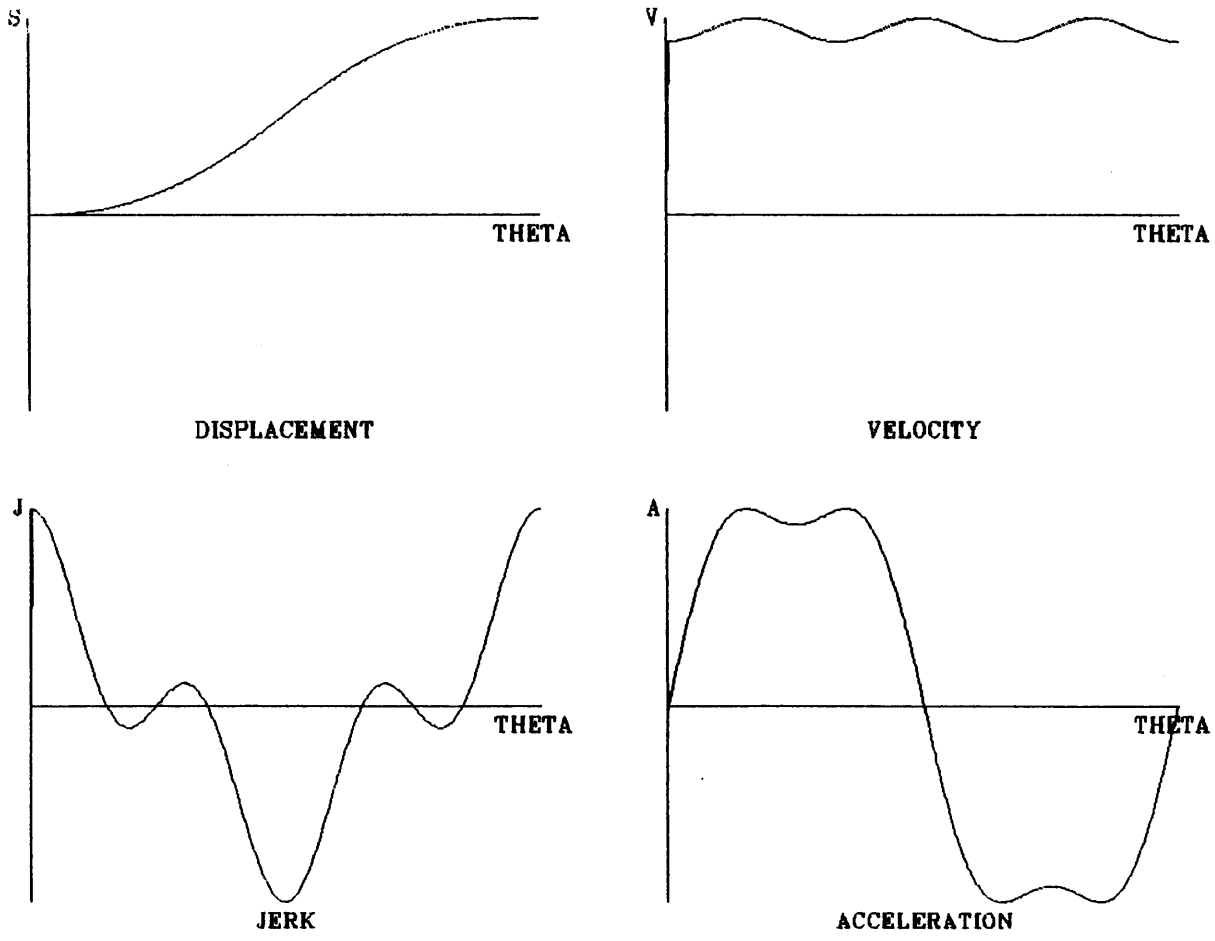


Figure 19. Motion Characteristics for Gutman 1-3 Harmonic Curve

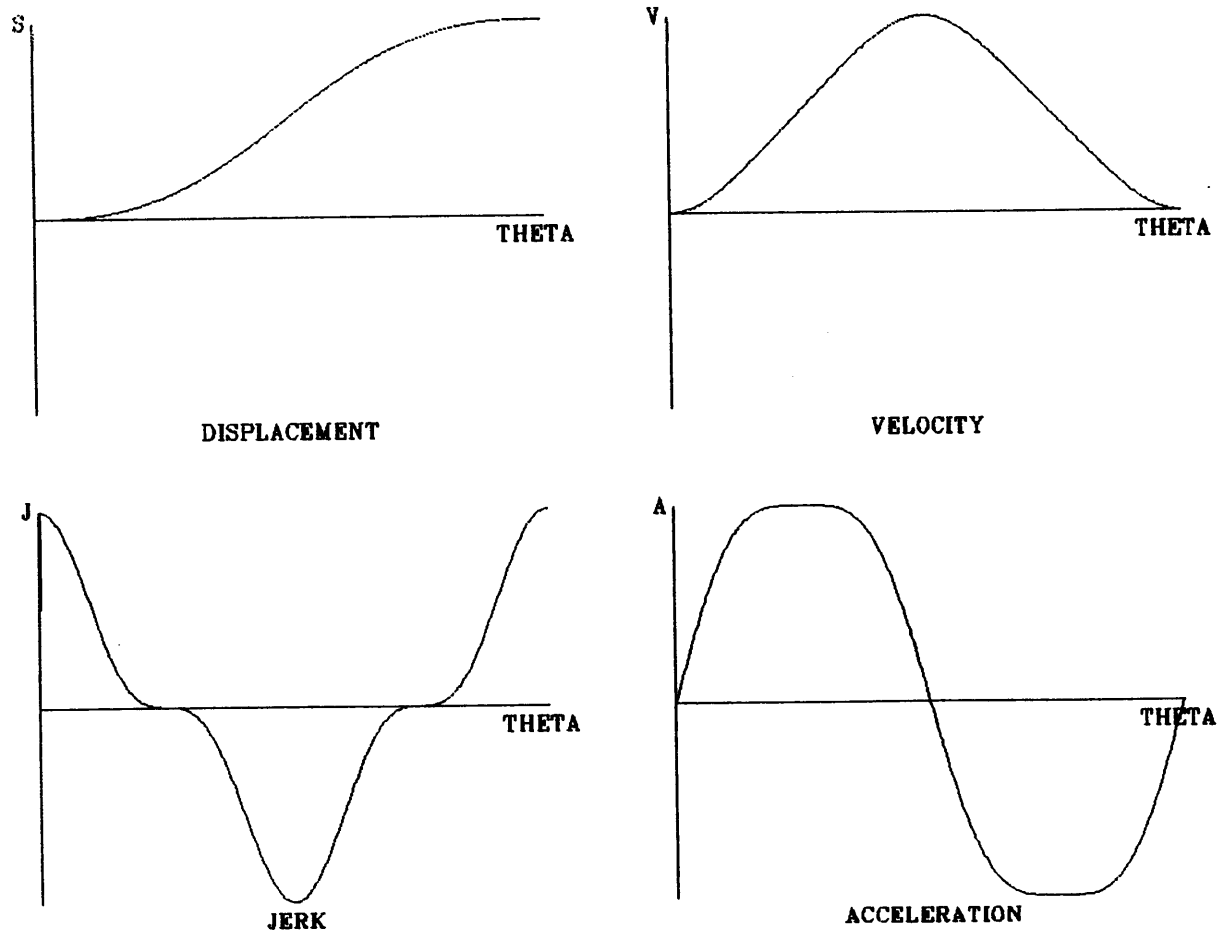


Figure 20. Motion Characteristics for Freudenstein 1-3 Harmonic Curve

where,

$$m = \frac{1125}{1192}$$

The motion characteristics are shown in Fig. 21.

### 4.2.7 MODIFIED CAM CURVES

Modified curves were developed to use the better features of the basic curves in designing cams with the lowest maximum acceleration and having bounded jerk. Two such curves are modified sine and modified trapezoidal curves. The characteristics of these curves are shown in Fig. 22 and Fig. 23.

Modified Sine: This composite curve is a combination of cycloidal and harmonic curves. It consists of 3 sections, each of which is governed by different displacement functions. Application of the boundary conditions, matching of velocities and accelerations, and sustituting $\gamma$ for $\theta/\beta$ lead to equations 66 to 74.

For $0 \leq \gamma \leq 1/8$

$$s = h(0.43989\gamma - 0.035014\sin(4\pi\gamma)) \tag{66}$$

$$v = 0.43989\frac{h}{\beta}(1 - \cos(4\pi\gamma)) \tag{67}$$

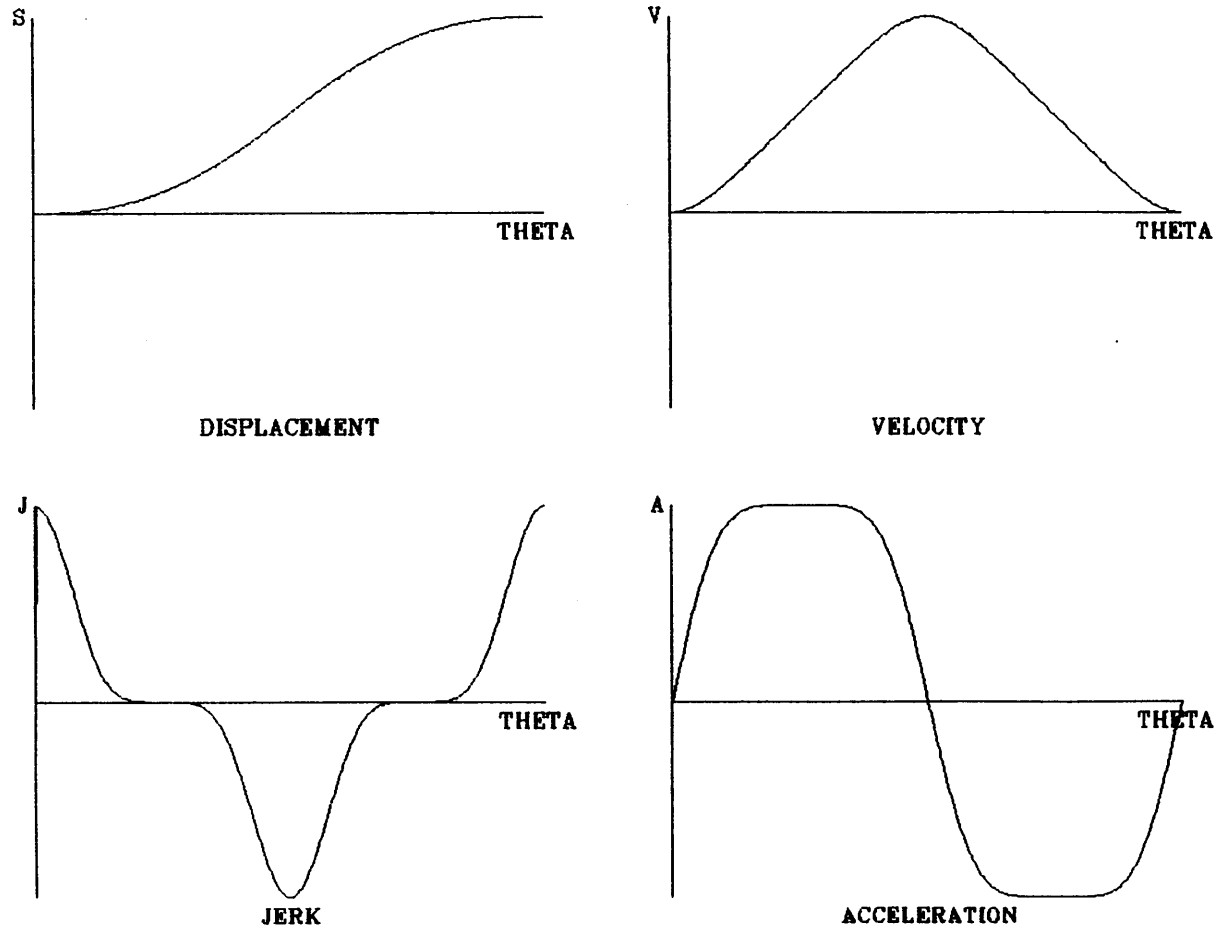


Figure 21. Motion Characteristics for Freudenstein 1-3-5 Harmonic Curve

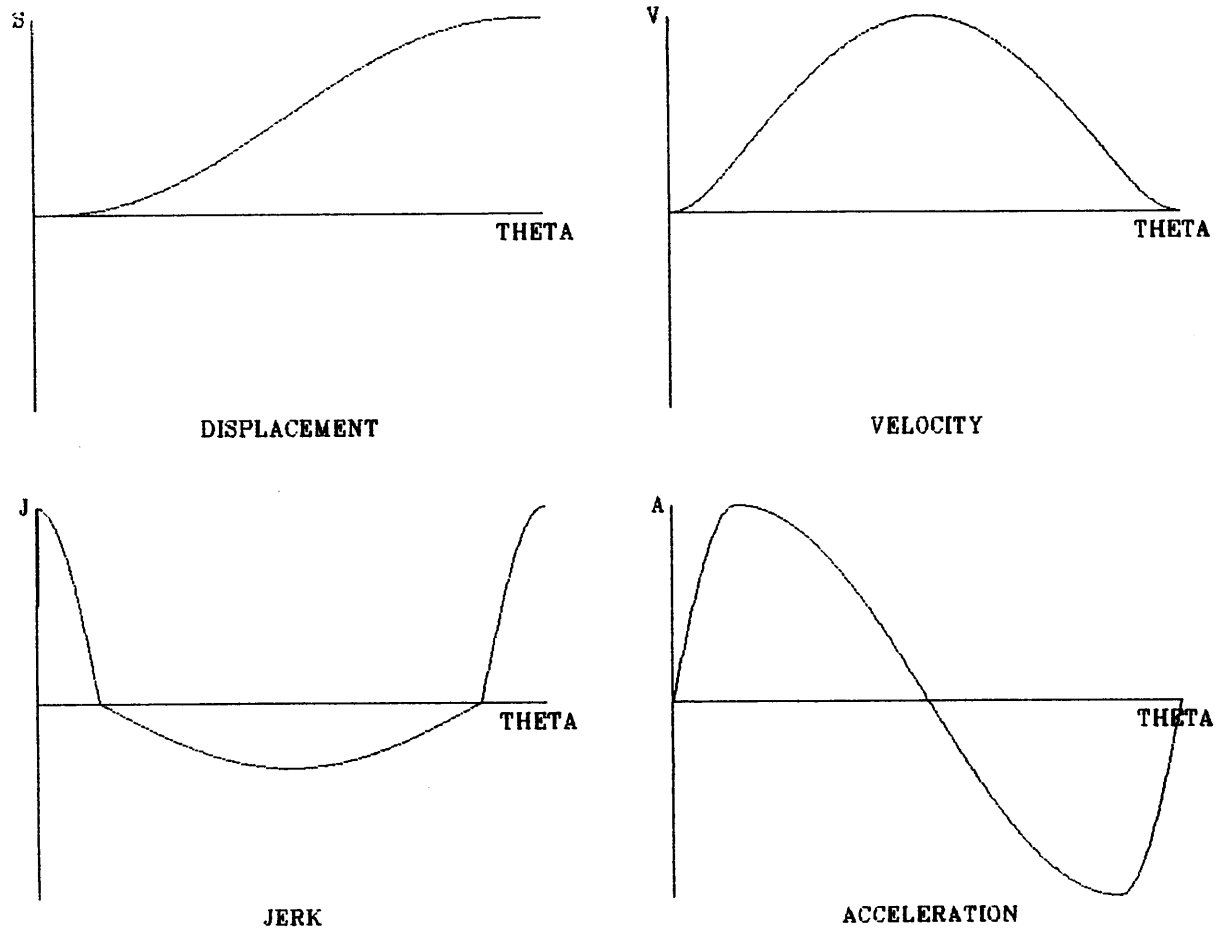


Figure 22. Motion Characteristics for Modified Sine Curve

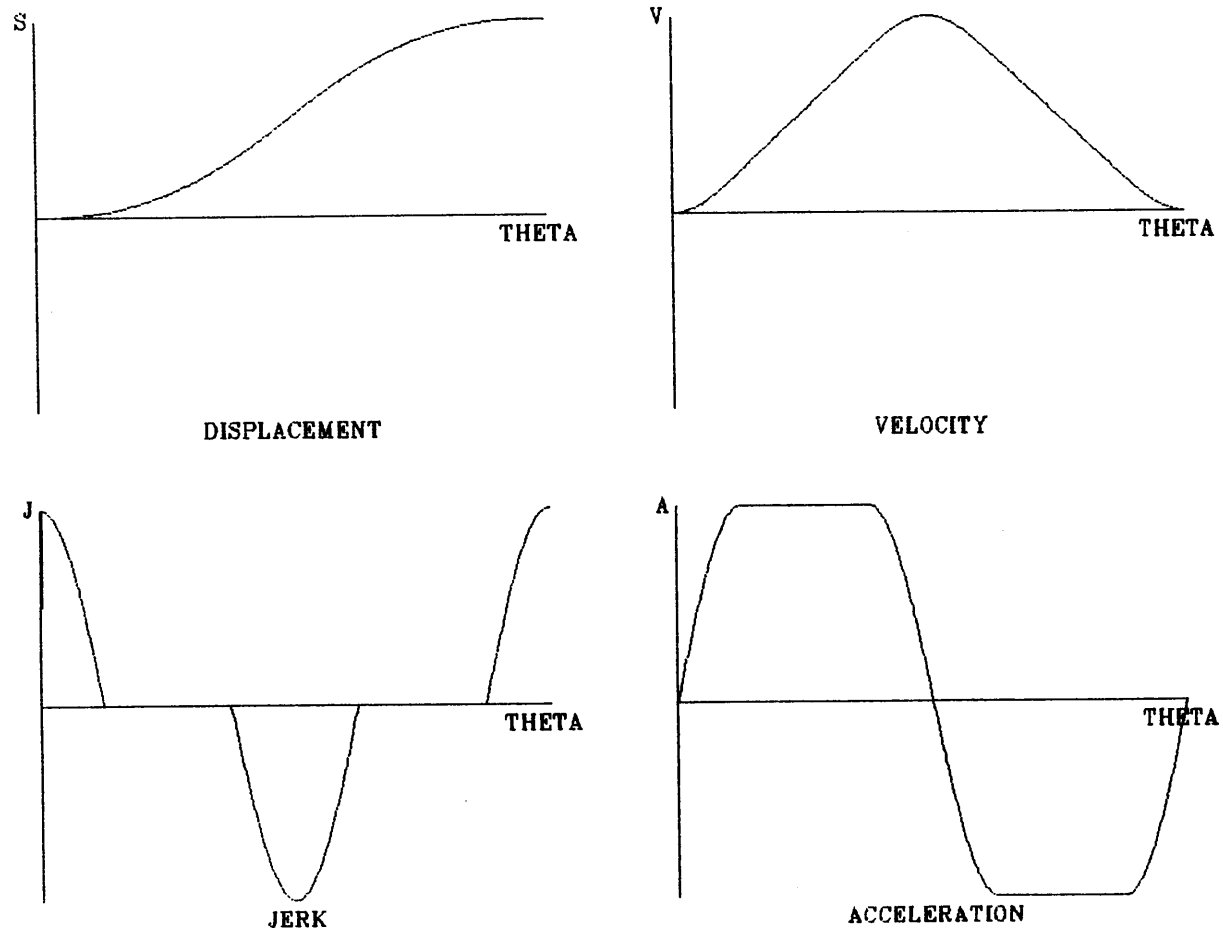


Figure 23. Motion Characteristics for Modified Trapezoidal Curve

$$a = 5.52794\frac{h}{\beta^2}\sin(4\pi\gamma) \tag{68}$$

For $1/8 \leq \gamma \leq 7/8$

$$s = h\left(0.28005 + 0.43989\gamma - 0.315055\cos\left(\frac{4\pi}{3}\gamma - \frac{\pi}{6}\right)\right) \tag{69}$$

$$v = \frac{h}{\beta}\left(0.43989 - 1.31967\sin\left(\frac{4\pi}{3}\gamma - \frac{\pi}{6}\right)\right) \tag{70}$$

$$a = 5.52794\frac{h}{\beta^2}\cos\left(\frac{4\pi}{3}\gamma - \frac{\pi}{6}\right) \tag{71}$$

For $7/8 \leq \gamma \leq 1$

$$s = h(0.56010 + 0.43989\gamma - 0.035014\sin 2\pi(2\gamma-1)) \tag{72}$$

$$v = 0.43989\frac{h}{\beta}(1 - \cos 2\pi(2\gamma-1)) \tag{73}$$

$$a = 5.52794\frac{h}{\beta^2}\sin 2\pi(2\gamma-1) \tag{74}$$

Modified Trapezoidal: This curve has been evolved by replacing the cubic equation by the cycloidal equation in the first, third, fourth and the sixth segments of the curve. Applying boundary conditions and denoting

the ratio $\theta/\beta$ by $\gamma$ the motion equations of the modified trapezoidal curve can be described as in equations 75 to 92.

For $0 \leq \gamma \leq 1/8$

$$s = 0.097246h(4\gamma - \frac{1}{\pi}\sin(4\pi\gamma)) \tag{75}$$

$$v = 0.388985\,\frac{h}{\beta}\,(1 - \cos(4\pi\gamma)) \tag{76}$$

$$a = 4.888124\,\frac{h}{\beta^2}\sin(4\pi\gamma) \tag{77}$$

For $1/8 \leq \gamma \leq 3/8$

$$s = h(2.4440618\gamma^2 - 0.22204\gamma + 0.0072341) \tag{78}$$

$$v = \frac{h}{\beta}\,(4.888124\gamma - 0.22204) \tag{79}$$

$$a = 4.888124\frac{h}{\beta^2} \tag{80}$$

For $3/8 \leq \gamma \leq 1/2$

$$s = h(1.61101154\gamma - 0.0309544\sin(4\pi\gamma - \pi) - 0.3055077) \tag{81}$$

$$v = \frac{h}{\beta}\,(1.6110154 - 0.388985\cos(4\pi\gamma - \pi)) \tag{82}$$

$$a = 4.888124\,\frac{h}{\beta^2}\sin(4\pi\gamma - \pi)) \tag{83}$$

For $1/2 \le \gamma \le 5/8$

$$s = h(1.6110154\gamma + 0.0309544\,\sin(4\pi\gamma - 2\pi) - 0.3055077) \tag{84}$$

$$v = \frac{h}{\beta}\,(1.6110154 + 0.388985\cos(4\pi\gamma - 2\pi) \tag{85}$$

$$a = -4.888124\,\frac{h}{\beta^2}\sin(4\pi\gamma - 2\pi) \tag{86}$$

For $5/8 \le \gamma \le 7/8$

$$s = h(4.6660917\gamma - 2.44406184\gamma^2 - 1.2292648) \tag{87}$$

$$v = \frac{h}{\beta}\,(4.6660917 - 4.888124\gamma) \tag{88}$$

$$a = -\,4.888124\,\frac{h}{\beta^2} \tag{89}$$

For $5/8 \le \gamma \le 7/8$

$$s = h(0.6110154 + 0.388985\gamma + 0.0309544\sin(4\pi\gamma - 3\pi)) \tag{90}$$

$$v = \frac{h}{\beta}\,(0.388985 + 0.3888985\cos(4\pi\gamma - 3\pi)) \tag{91}$$

$$a = -4.888124\,\frac{h}{\beta^2}\sin(4\pi\gamma - 3\pi) \tag{92}$$

## 4.3 CAM PROFILE DETERMINATION

Cam profiles can be determined graphically or analytically. Certain cam motion curves are easily laid out by graphical procedures. Due to inherent limitations in accuracy, graphical methods are used only for relatively low-speed cams. High-speed cams require that the coordinates be calculated by a computer.

Graphical methods employ an artificial device called an 'inversion'. This represents a mental concept which is very helpful in performing the graphical work. It requires the drawing of one position of the cam and many positions o fht e follower. Instead of rotating the cam, it is assumed that the follower rotates around the 'fixed' cam. As part of the inversion process, the direction of rotation is very important. In order to preserve the correct sequence of events, the artificial rotation of the follower must be the reverse of the cam's prescribed rotation.

Because of the voluminous calculations required in conjunction with its complicated geometry, the analytical methods of determining cam profiles coordinates were subordinated to graphical techniques. This thesis uses the 'Theory of Envelopes' (3) to determine certain cam profile coordinates equations for seven major types of cam followers. We shall consider them in sequence.

In the equations shown in the following pages, $r_b$ is the base circle radius, $r_f$ the roller radius, $r_a$ the distance between swinger pivot point and cam center, $r_r$ the length of follower arm, s the follower rise for angular displacement $\theta$, $\phi_o$ the intial angular postion of swinging follower, $\phi$ the change in angular position of swinging follower, $\alpha$ the pressure angle at angular displacement $\theta$, and e the eccentricity of the follower.

### 4.3.1 KNIFE EDGE FOLLOWER

Figure 24 shows the cam system with the knife edge follower. In this case the point of contact between the cam and the follower always lies on the radial line joining the cam center with the knife-edge. This fact makes it very easy to compute the coordinates of the cam profile. Using the standard notations it can be seen that the rectangular coordinates of the profile are:

$$x = r\cos\theta \tag{93}$$

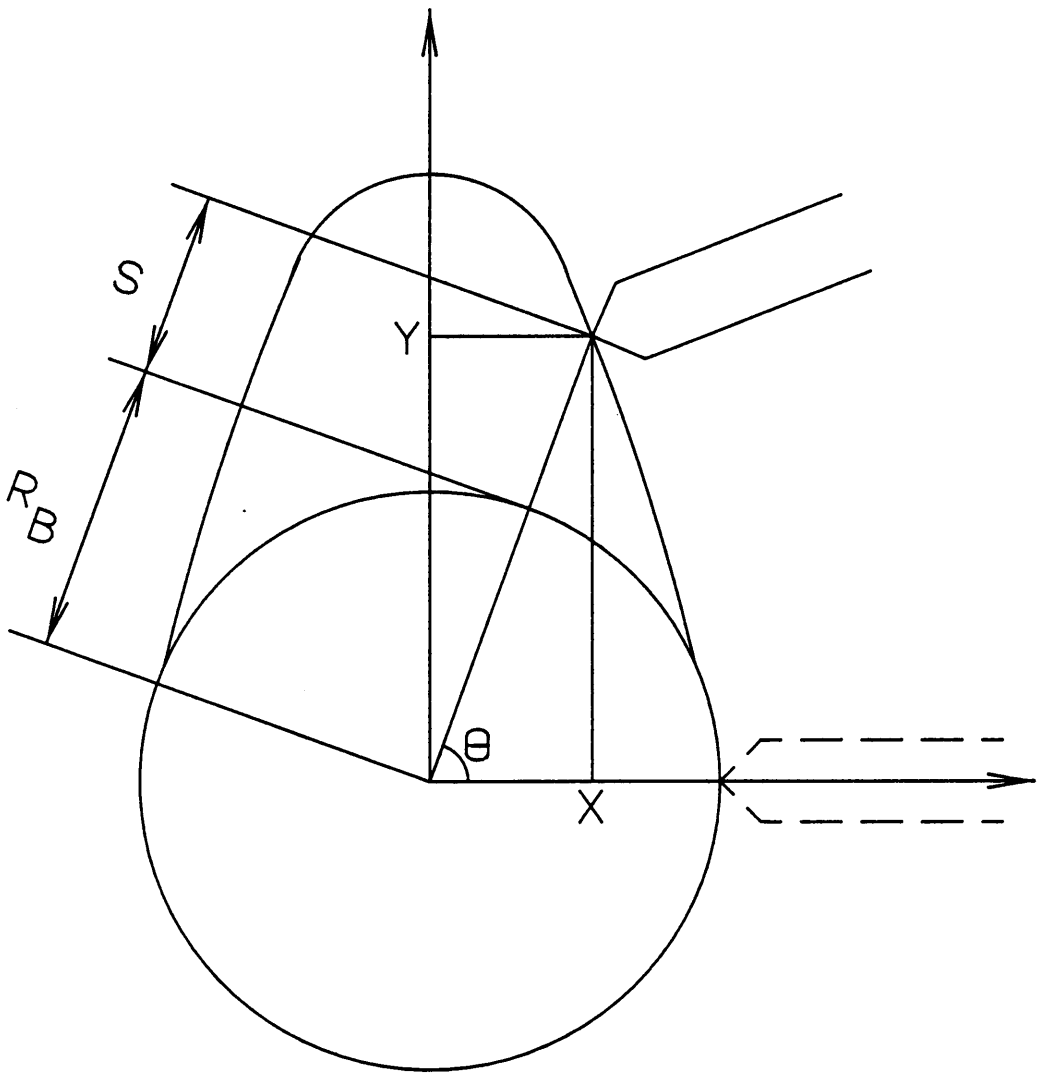


Figure 24. Cam with Knife-Edge follower

$$y = r\sin\theta \tag{94}$$

where,

$r = r_b + s$

### 4.3.2 RADIAL TRANSLATING ROLLER FOLLOWER

In this case the point of contact generally does not lie on the line joining the center of cam and the roller. The system is shown in Fig. 25. Again, the coordinates of the cam profile can be found by pure geometry. This indicates how the pressure angle $\alpha$ affects the system geometry. The x and y coordinates are as follows:

$$x = r\cos\theta - r_f\cos(\theta - \alpha) \tag{95}$$

$$y = r\sin\theta - r_f\sin(\theta - \alpha) \tag{96}$$

where,

$r = r_b + r_f + s$

$$\alpha = \tan^{-1}\left[\frac{\frac{ds}{d\theta}}{r}\right]$$

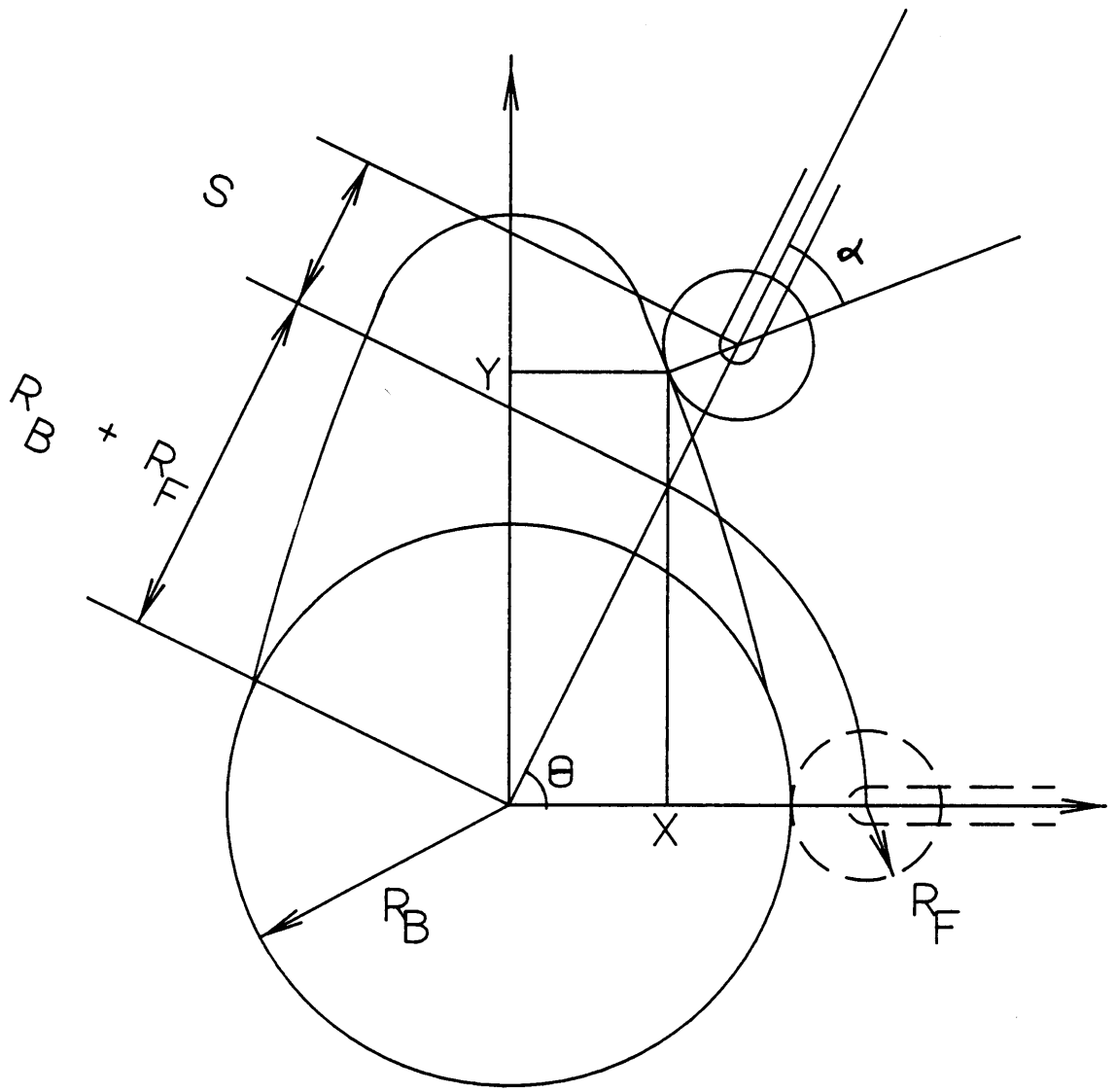


Figure 25. Cam with Roller follower

### 4.3.3 OFFSET TRANSLATING ROLLER FOLLOWER

This system is a variation of the one described above. The eccentric follower moves radially along a line that is offset from the cam center by an amount e. Equations 97 and 98 are derived from the geometry of the system shown in Fig. 26.

$$x = r\cos\theta - r_f\cos(\theta - \alpha) + e\sin\theta \tag{97}$$

$$y = r\sin\theta - r_f\sin(\theta - \alpha) - e\cos\theta \tag{98}$$

where,

$$r = \sqrt{(r_b + r_f)^2 - e^2 + s}$$

$$\alpha = \tan^{-1}\left[\frac{\frac{ds}{d\theta} - e}{r}\right]$$

It must be noted that if e = 0 then the above equations become the same as those for the radial follower.

### 4.3.4 SWINGING ROLLER FOLLOWER

As equations 99 and 100 show, the coordinates of the cam profile also depend on the geometry of the swing arm and the distance between the

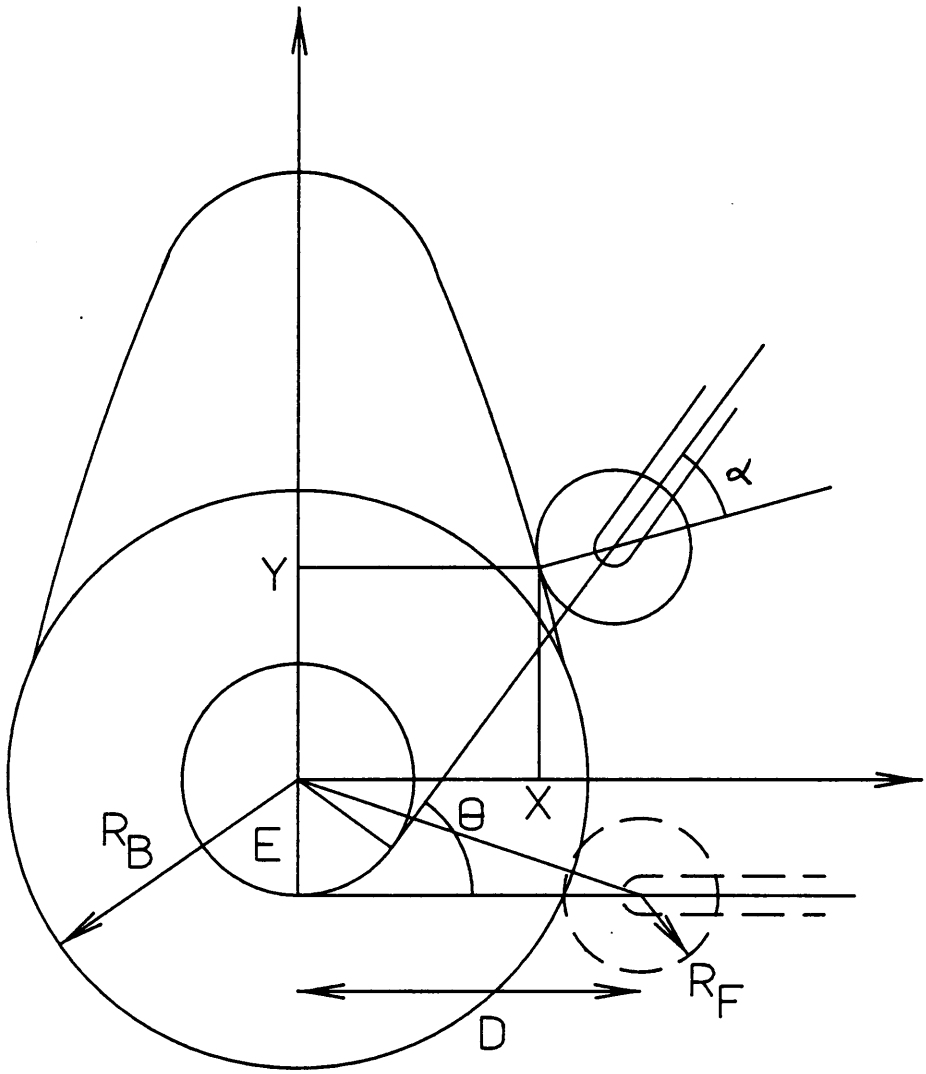


Figure 26. Cam with Offset Roller follower

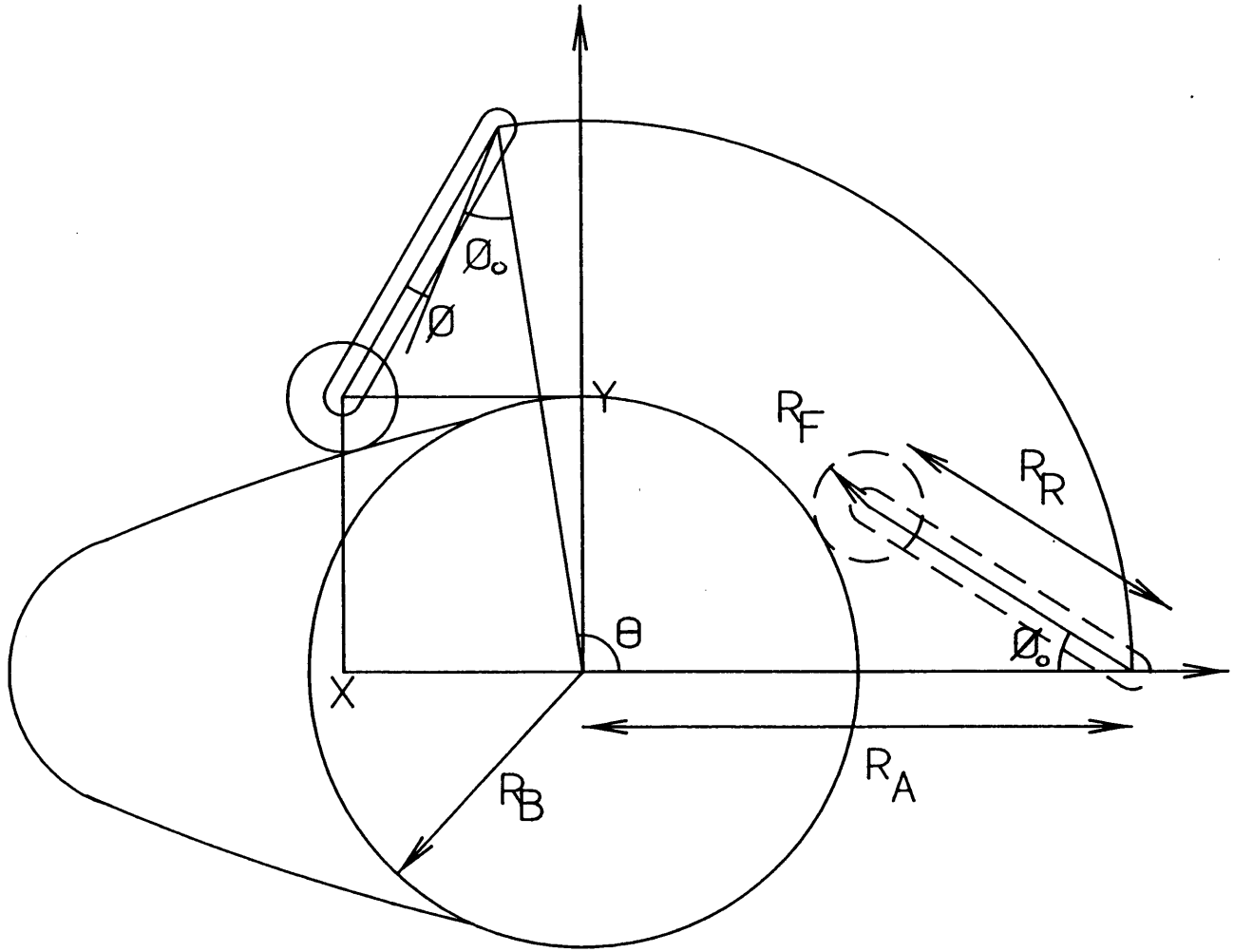


Figure 27. Cam with Swinging Roller follower

cam center and the swinger pivot point. Figure 27 shows a schematic diagram of this system.

$$x = r_a\cos\theta - r_r\cos(\theta - \phi - \phi_o) - r_f\sin(\alpha - \theta + \phi + \phi_o) \tag{99}$$

$$y = r_a\sin\theta - r_r\sin(\theta - \phi - \phi_o) - r_f\cos(\alpha - \theta + \phi + \phi_o) \tag{100}$$

where,

$$\phi_o = \cos^{-1}\left[\frac{r_r^2 + r_a^2 - (r_b + r_f)^2}{2r_a r_r}\right]$$

### 4.3.5 TRANSLATING FLAT FACED FOLLOWER

Figure 28 shows a cam with a translating flat faced follower. The general method of using geometry fails to help determine the cam profile. However combining this with the offset of the point of contact from the radial lines helps us get equations 101 and 102 for the coordinates of the cam profile.

$$x = (r_b + s)\cos\theta - \frac{ds}{d\theta}\sin\theta \tag{101}$$

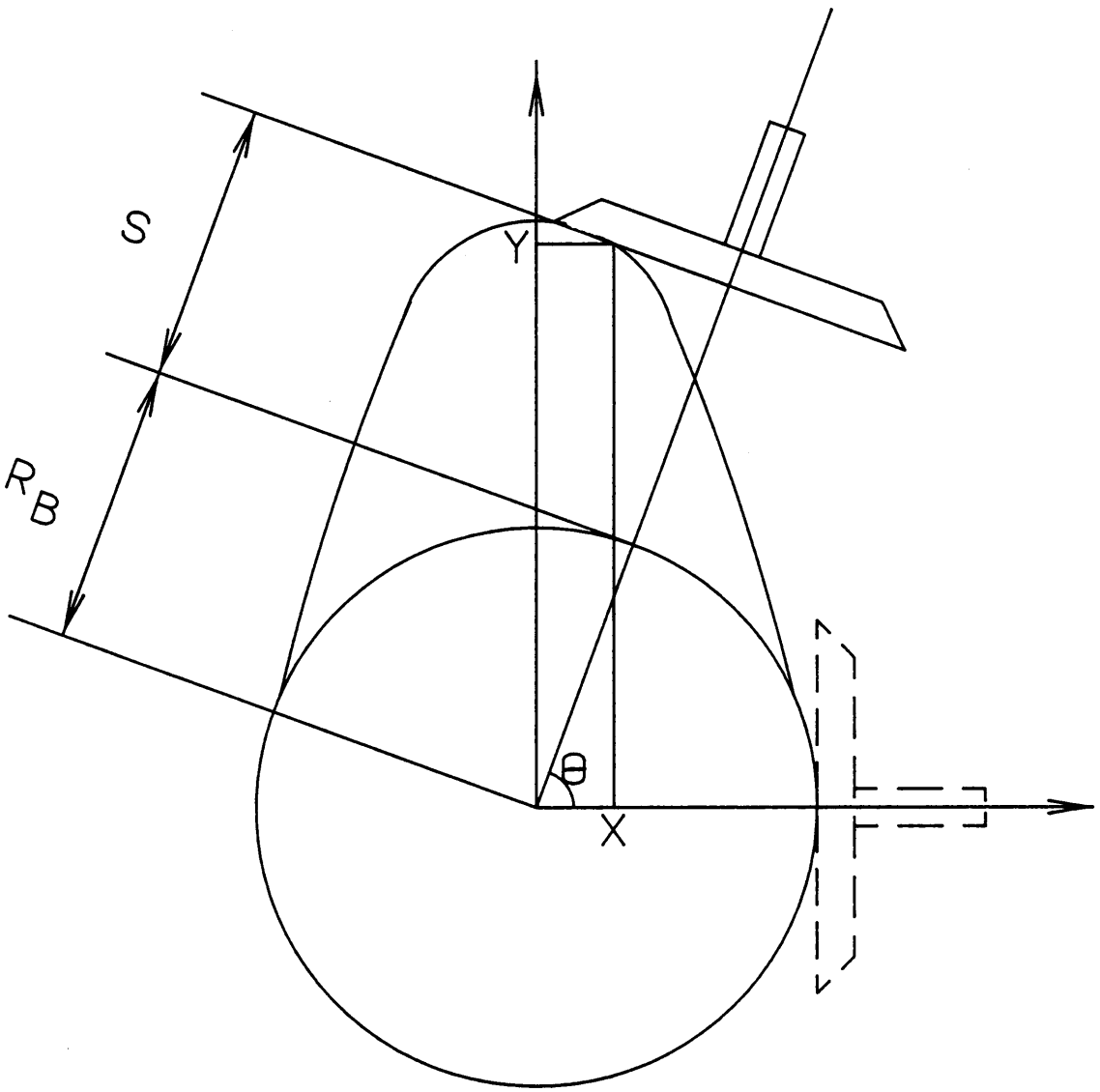


Figure 28. Cam with Flat-Faced follower

$$y = (r_b + s)\cos\theta + \frac{ds}{d\theta}\cos\theta \qquad (102)$$

### 4.3.6 SWINGING CENTRIC FLAT-FACED FOLLOWER

Figure 29 shows a swinging flat faced follower system. Using a method similar to the one used in the case of swinging roller follower, it is possible to derive equations 103 and 104 for the cam profile coordinates.

$$x = r_a(\cos\theta + A\cos\beta) \qquad (103)$$

$$y = r_a(\sin\theta - A\sin\beta) \qquad (104)$$

where,

$$A = \frac{\cos(\theta + \beta)}{\frac{d\phi}{d\theta} - 1}$$

$$\beta = \phi_o + \phi - \theta$$

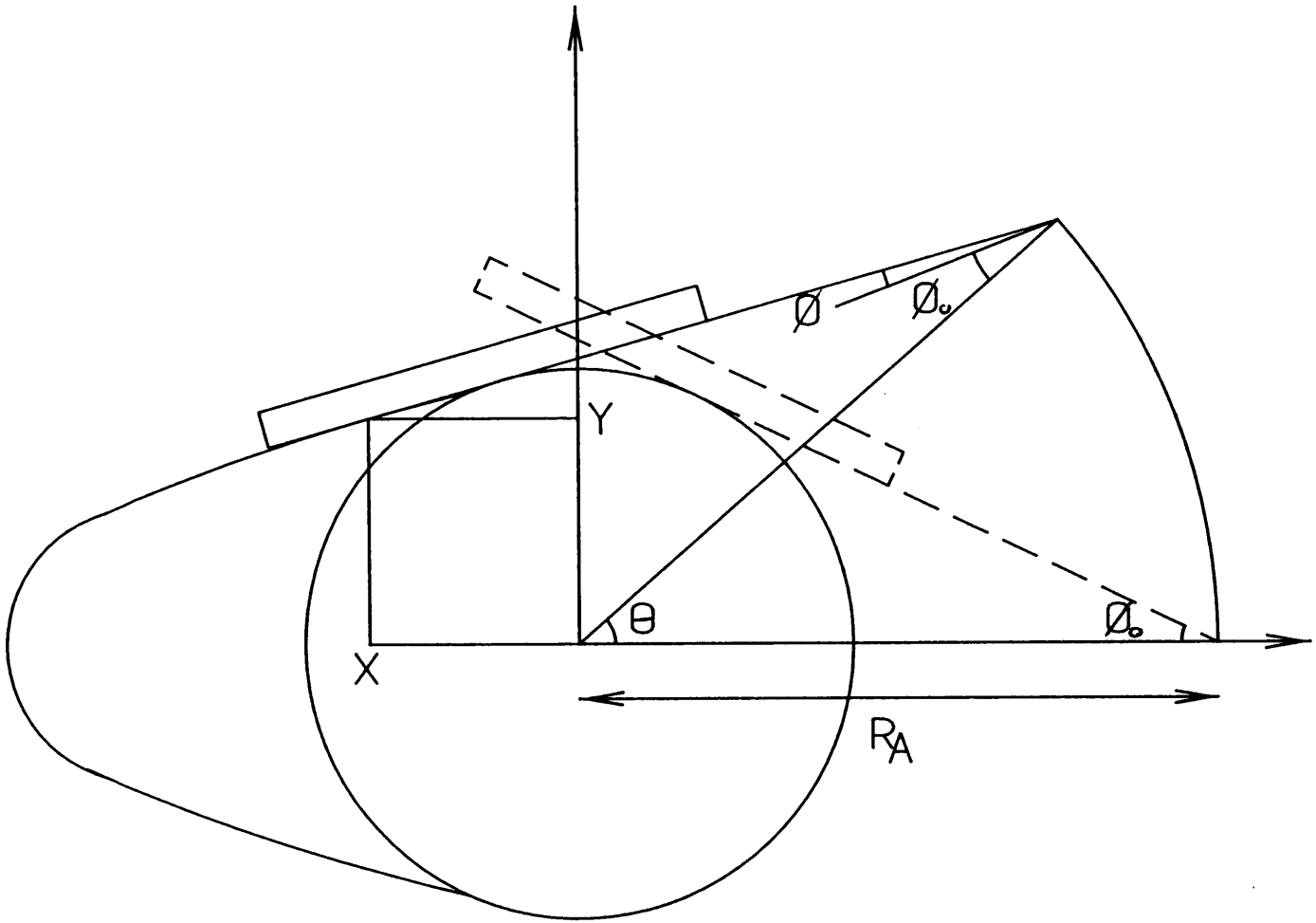


Figure 29. Cam with Swinging Centric Flat-Faced follower

### 4.3.7 SWINGING ECCENTRIC FLAT-FACED FOLLOWER

A swinging flat-faced follower of the eccentric type is shown in Fig. 30. The amount of eccentricity e is the distance between parallel lines through the cam pivot and the follower face. The profile coordinates are:

$$x = r_a(\cos\theta + A\cos\beta) + e\sin\beta \tag{105}$$

$$y = r_a(\sin\theta - A\sin\beta) + e\cos\beta \tag{106}$$

where,

$$A = \frac{\cos(\theta + \beta)}{\frac{d\phi}{d\theta} - 1}$$

$$\beta = \phi_o + \phi - \theta$$

If e = 0 then equations 105 and 106 reduce to 103 and 104 respectively.

## 4.4 FORCE CONSIDERATIONS

The cam-follower system is subject to a number of forces. It is necessary to determine these forces and use them in the design process. The cam-follower forces are:

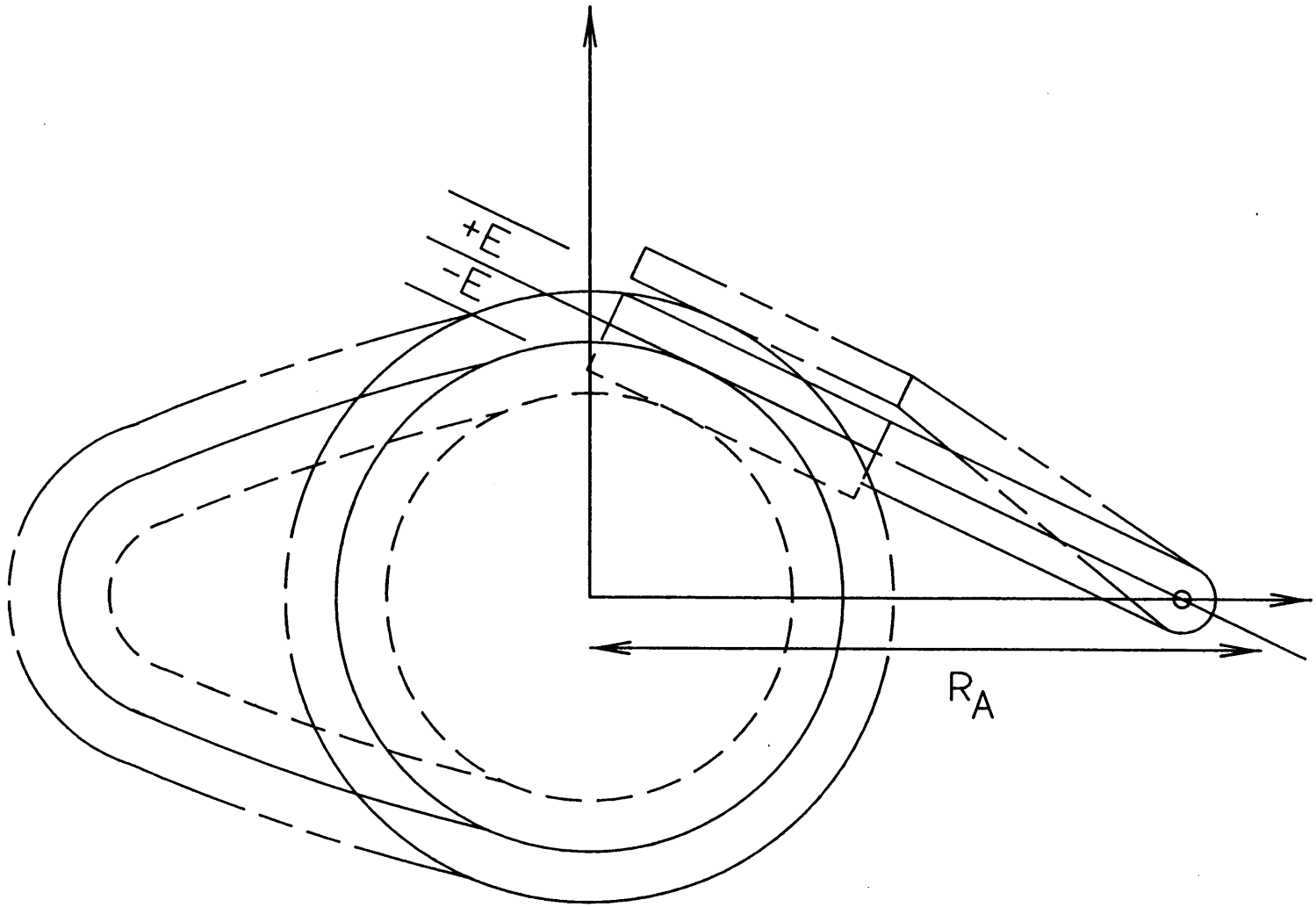


Figure 30. Cam with Swinging Eccentric Flat-Faced follower

1. The static force due to the external load on the cam.
2. The inertia force due to follower acceleration.
3. The acceleration forces which produce vibrations.
4. Frictional forces.

Static forces: These are gradually applied forces and are characteristic of slow moving cams. The equilibrium conditions applied in static force analysis are:

$$F = 0 \text{ (in the X, Y, and Z directions)}$$

$$M = 0 \text{ (about any point)}$$

Inertia Forces: If a rotating body has an external unbalanced torque, it will have an angular acceleration which will be resisted by a reaction torque. The direction of this torque will be opposite to the direction of rotation.

$$T = I\alpha$$

where $I$ = Moment of Inertia about the center of rotation.

$\alpha$ = angular acceleration.

Vibratory Forces: It is known that a large pulse (jerk) induces undesirable vibrations which occur at the natural frequency of the system. The forces produced by these vibrations add on to the inertia forces. Hence while designing cams for high speed applications, a dynamic magnification factor of 2 or more should be used (3).

Frictional Forces Relative movement of contacting surfaces is always opposed by frictional resistance. Friction can rarely be neglected. The best and most accurate method of including friction in the design is to use values from tests.

## 4.5 PRESSURE ANGLE CONSIDERATIONS

Pressure angle is important in cam design. Pressure angle is defined as the angle between the line of action and the center line of the follower. Pressure angles result in side thrusts on the follower and possible deflection and jamming of the follower stem. The maximum pressure angle affects the cam size, torque, loads, accelerations, wear life, and other factors. Thus the maximum pressure angle must be kept as small as possible. At present, the maximum recommended value this angle can achieve has been arbitrarily set at $30^{\circ}$. However, for light loads with accurate low-friction bearings, Rothbart(1) was successful in using pressure angles up to $47^{\circ}$.

Pressure angles can be easily measured from the graphical construction of a cam. However, analytical methods have been developed to calculate these angles for different follower types. Since solving equations for maximum pressure angles is very difficult, Mabie and Ocvirk(17) use a nomogram (Fig. 31) developed by E. C. Varnum to determine the maximum pressure angle. This chart can be used for three different types of motion. The nomogram uses $\beta_o$ and $L/R_o$ as parameters.

Pressure angles can be reduced by increasing the minimum radius of the cam so that the follower will have to travel a greater linear distance

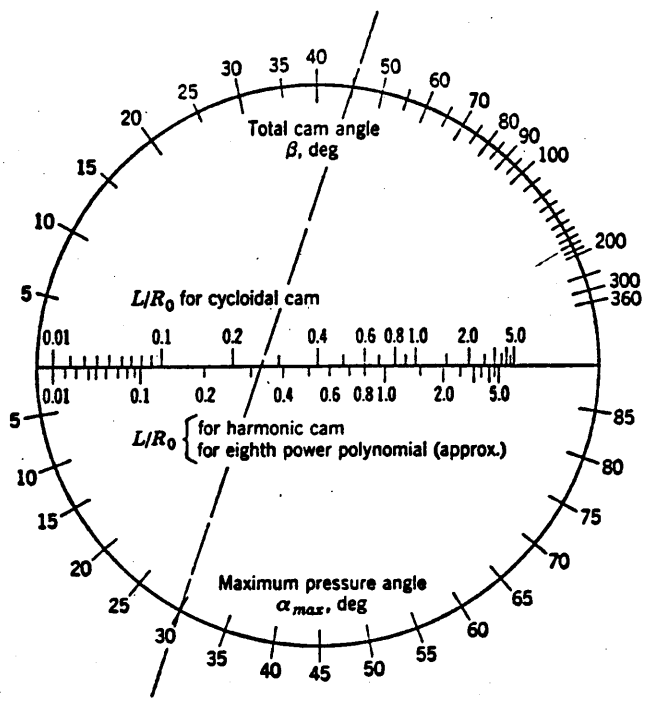


Figure 31. Nomogram to find maximum pressure angle. (Courtesy of E. C. Varnum, Barber-Colman Company.)

on the cam for a given rise. If the workspace does not permit such an increase, then the active cam angle $\beta$ can be increased along with the rotational speed to maintain a constant lift time.

## 4.6 CAM SIZE DETERMINATION.

Care should be taken to avoid cusps or undercutting. Cusp or point formation takes place when, both, $dx/d\theta$ and $dy/d\theta$ become zero simultaneously. Based on this principle, equations have been developed to check cusp formation for different types of followers. Two simple cases are listed below:

Flat Faced Follower: If $f(\theta)$ is the displacement function for the follower and $f''(\theta)$ the acceleration function, then the minimum radius of the cam (C) required, to avoid cusps or undercutting, can be easily determined by solving equation 107 for C.

$$C + f(\theta) + f''(\theta) > 0 \qquad (107)$$

Radial Follower: For this type of follower, a different parameter, the radius of curvature ($\rho$) of the pitch surface is considered. In order to prevent undercutting or point formation $\rho_{min}$ must be determined. Two sets of curves, that show the plot of $\rho_{min}/R_o$ versus $\beta$ for different values of $L/R_o$, have been developed and used in Mabie & Orvick (17). Here $\beta$ is the active cam angle, L is the total lift and $R_o$ the minimum pitch circle

radius. Figures 32 and 33 show the graphs for Cycloidal and Harmonic motions respectively.

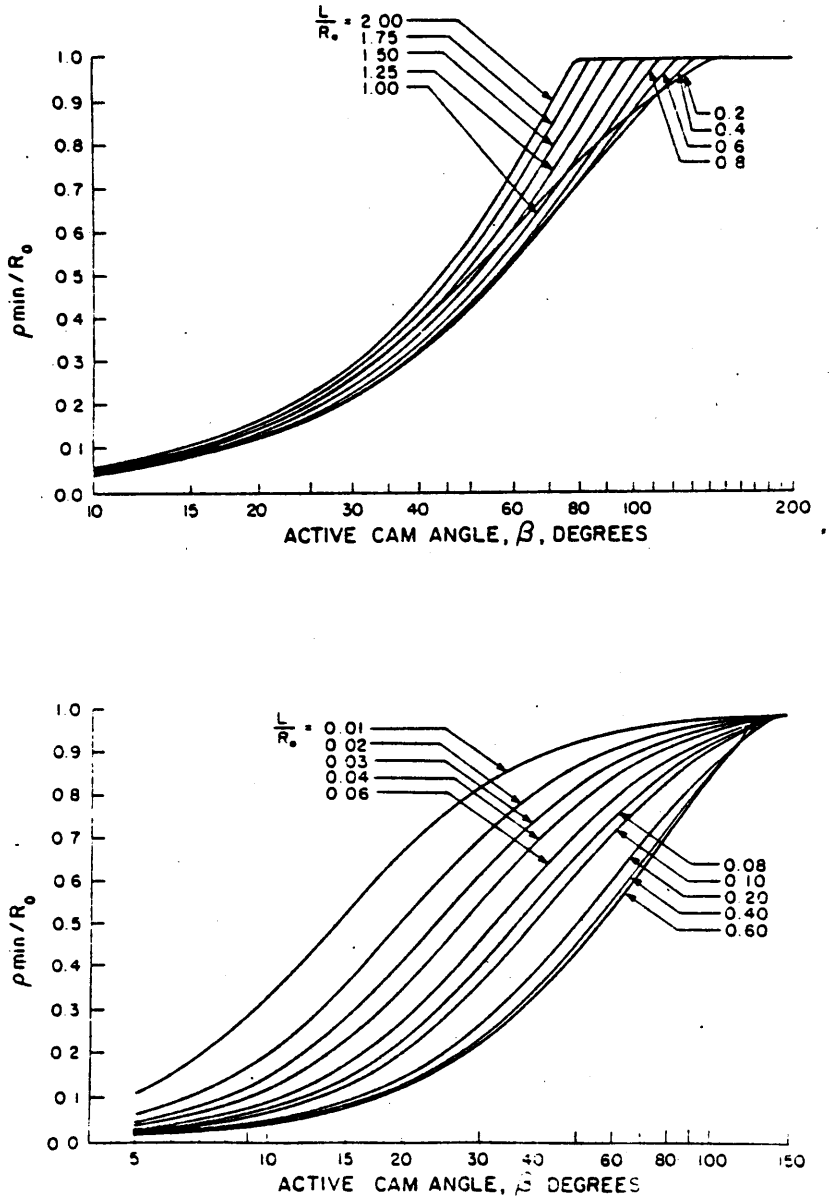


Figure 32. Curves for Cycloidal motion. (From M. Kloomok and R. V. Muffley,"Plate Cam Design-Radius of Curvature", Production Engineering, September 1955.)

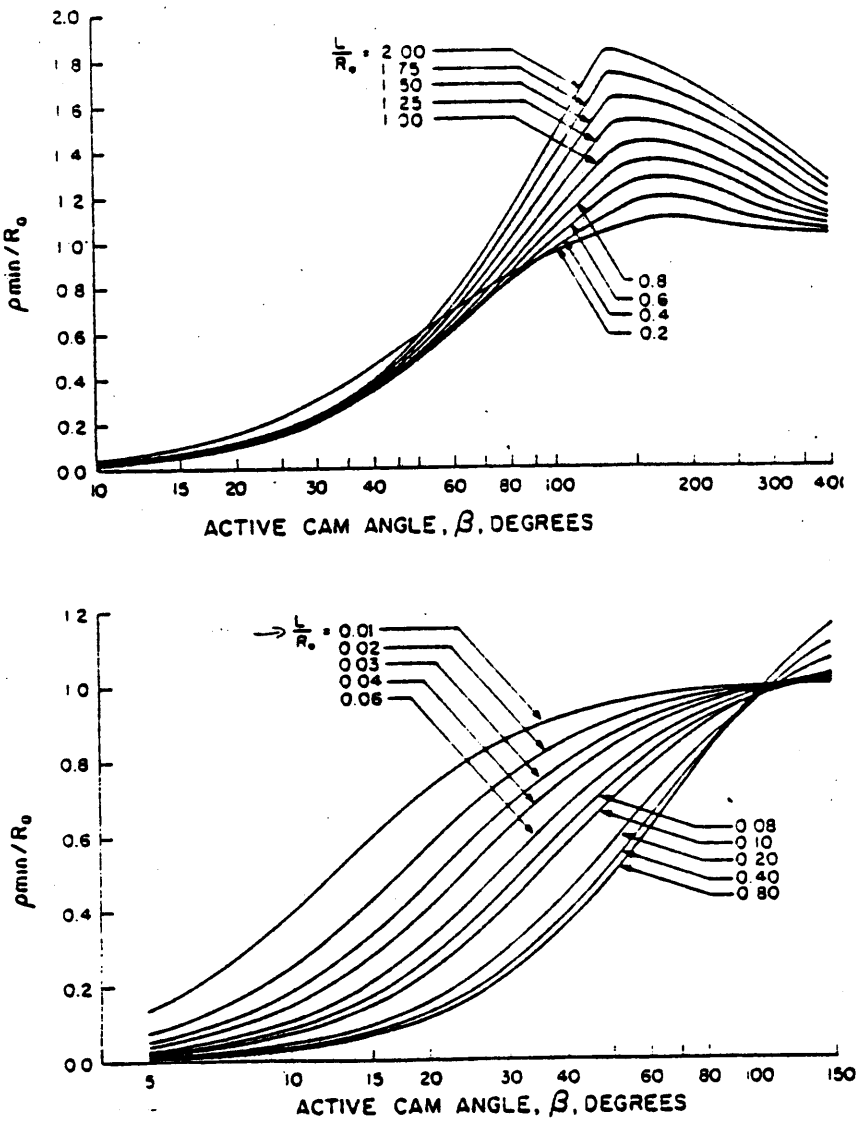


Figure 33. Curves for Harmonic motion. (From M. Kloomok and R. V. Muffley,"Plate Cam Design-Radius of Curvature", Production Engineering, September 1955.)

# CHAPTER 5 PROGRAM DESCRIPTION

## 5.1 OVERVIEW

ANICAM has been written to allow the user to create animated sequences for disk cams. Figure 34 illustrates the input and execution flow that a user would take to design and display the cam. Upon starting ANICAM, the user will be asked to select the system on which the cam model has to be displayed. Whether the user will be able to animate the cam, depends on the choice of the display system. The two systems which have animation capability are GKS and MOVIE.BYU. Based on the design requirements the user would sequentially select the follower type (there are 7) and the basic data of the cam and the motion (displacement) characteristics . The user has a choice of 22 different displacement functions and is expected to select compatible motions. The last input asked of the user is to enter the type of output required. The user may specify any of the 3 graphics options.

ANICAM is a fairly large program and will occupy a lot of memory space. By overlaying the program, it is possible to have only a part of the program in active memory. However one shortcoming of overlaying is that it can significantly slow the program execution. To facilitate possible overlays ANICAM is broken into four major parts with the functions of program control, motion data input, motion characteristics and profile generation and display. Since ANICAM is currently running on a system with virtual memory, overlaying is not needed.

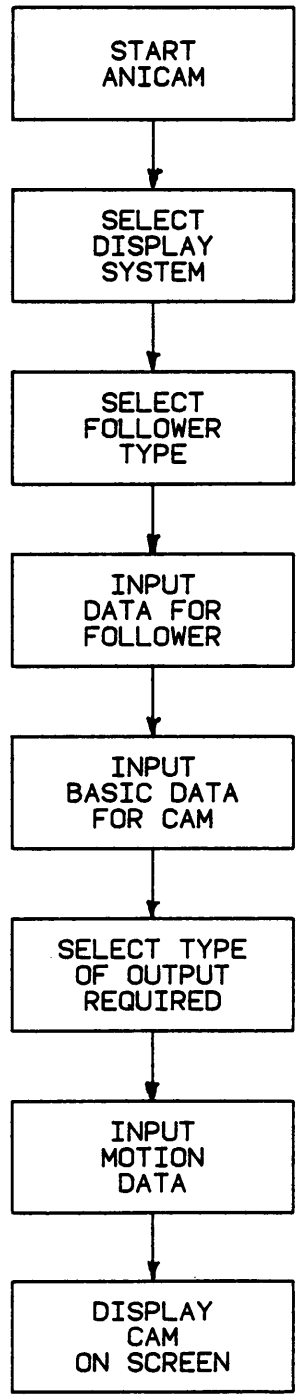


Figure 34. I/P and execution flow for ANICAM

## 5.2 PROGRAM CONTROL

The control portion of ANICAM is its heart. It calls the other program sections and passes the necessary data between them. This portion is kept small since it must always occupy active memory. The use of overlaying allows larger program segments to be called into memory above the root segment. Figure 35 illustrates a flow for the control section of the program. After starting the program the user is asked to key in the basic data for the cam and follower system. All this data is transferred through FORTRAN common blocks to the motion input program and continues on until the program has completed its operation. This process is repeated until the user indicates that he would like to stop.

## 5.3 MOTION DATA INPUT

The input section of the program takes the data regarding the displacement requirements and the functions governing the motion to be imparted to the follower. This continues until the user has specified these requirements for all the 360 degrees of the cam rotation. Once all data has been accepted, it is passed to the control section which in turn passes it to the generation section of the program. The flow chart for the input segment is shown in Fig. 36.

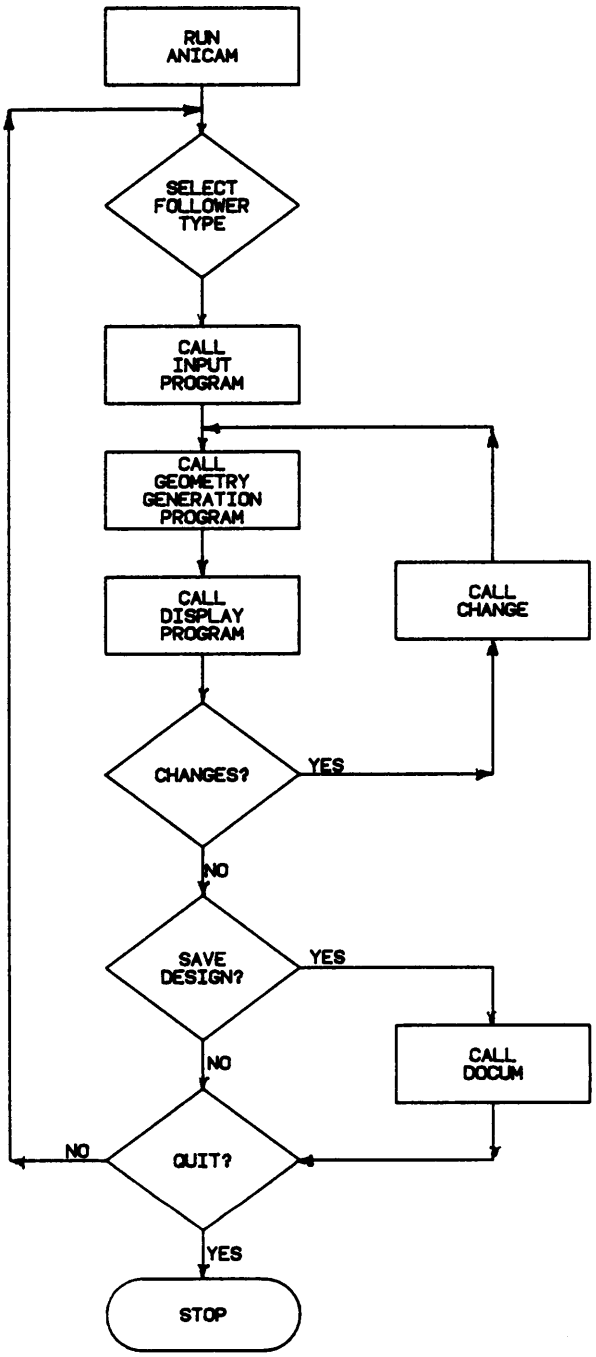


Figure 35. Flow of Control Section of ANICAM

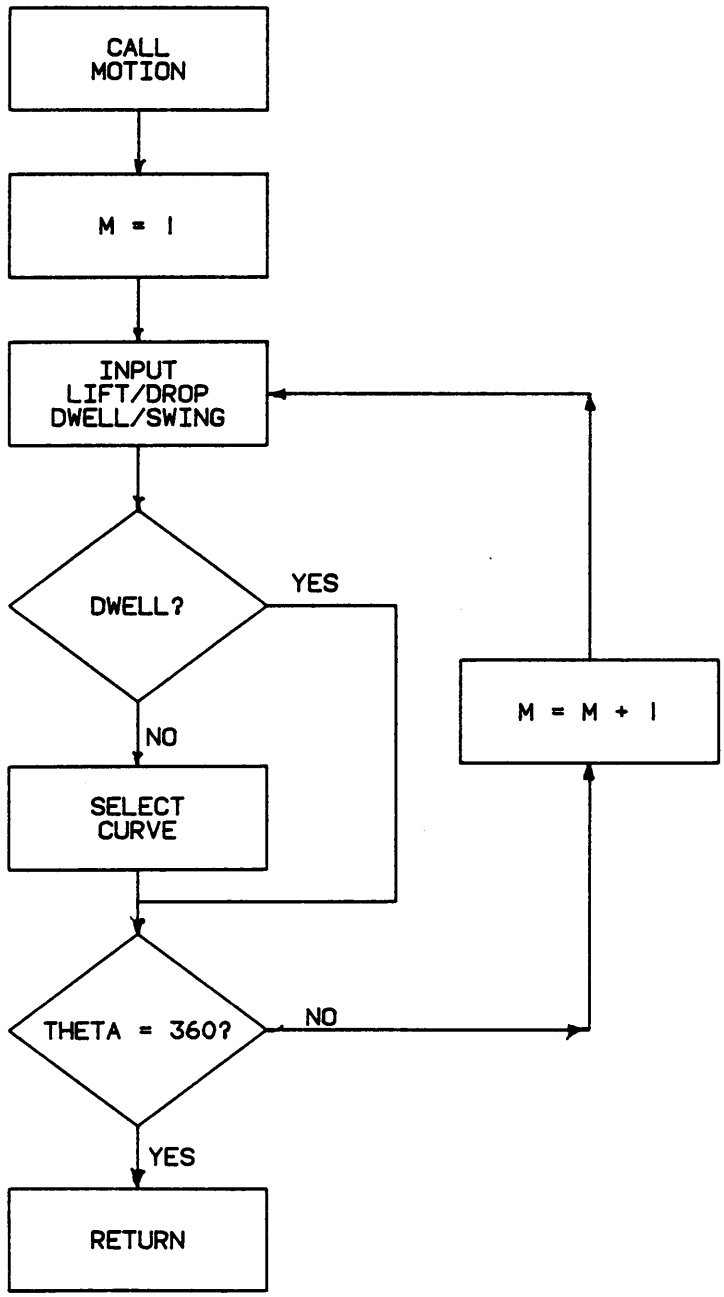


Figure 36. Flow for Input Section of ANICAM

## 5.4 CAM PROFILE AND MOTION CHARACTERISTICS GENERATION

This part of the program calculates the displacement, velocity, acceleration, jerk, and the cam profile coordinates at 1 degree intervals of angular displacement of the cam. It accepts the data regarding follower motion and jumps to the proper equations for the evaluation of the motion characteristics. Data regarding the follower type, along with data just entered is passed through to the subroutine for profile calculations. Here too, the segment goes to the correct equations for cam profile coordinate calculations. Once the coordinates are evaluated all data is passed to the display segment of the program. Figure 37 shows the flowchart for the generation segment.

## 5.5 DISPLAY

This portion takes all data and generated information and executes appropriate commands for the required display system. Each display system works independently and displays the designed model using different methods. By far, this is the most important part of the program. Displaying the designed cam helps the user to decide whether the design is feasible.

If the user has chosen GKS as the display system then he is asked to enter the type of TEKTRONIX terminal on which the model is to be displayed. GKS then proceeds to display the cam profile and the motion characteristics that would be imparted to the follower if that cam model is used. Various GKS subroutines are called in order to produce a color

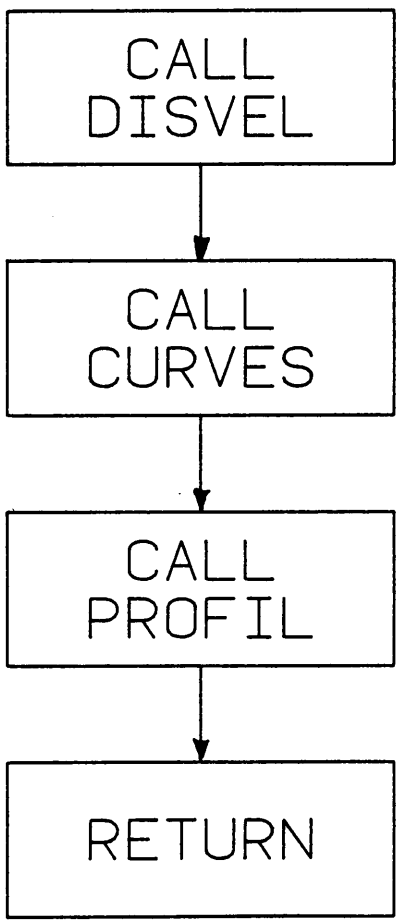


Figure 37. Flow for Geometry Generation Section of ANICAM

display of the cam model. A detailed list of all the subroutines called and the parameters used is given in Appendix A. GKS has the capability of performing animation. In the event that the user wishes to view the animation, GKS uses the transformation subroutine to transform the displayed segment. The illusion of animation is created by sequentially setting each transformed segment visible and invisible. The flow chart for the GKS display subroutine is shown in Fig. 38. Selected examples of the GKS display are shown in Figs. 39 and 40.

If the display system chosen is CADAM, then the user must logon to the CADAM system on the IBM 3250 or IBM 5080 before executing ANICAM. This allows the user to change the design of the cam without having to logon and run the program each time. ANICAM is linked to CADCD, a geometry interface module of the CADAM system. CADAM uses various CADCD subroutines to create and display the 3-D model of the cam and 2-D graphs of the motion characteristics. The user may perform various physical property analyses of the cam but does not have the capability to perform animation. Details of the CADCD subroutines used and the parameters used are listed in Appendix A. The system draws the cam profile by creating a 3-D spline between the calculated points. The flow chart for CADAM display is shown in Fig. 41. Selected examples of display obtained by this system are shown in Figs. 42 and 43.

CADAM creates a model which can be used for further design by interactively executing the CADMACGM program. This presents a new method to improve a design created by using CADAM.

If MOVIE is selected as the display system the user must leave ANICAM after creating the design. ANICAM writes the calculated data into

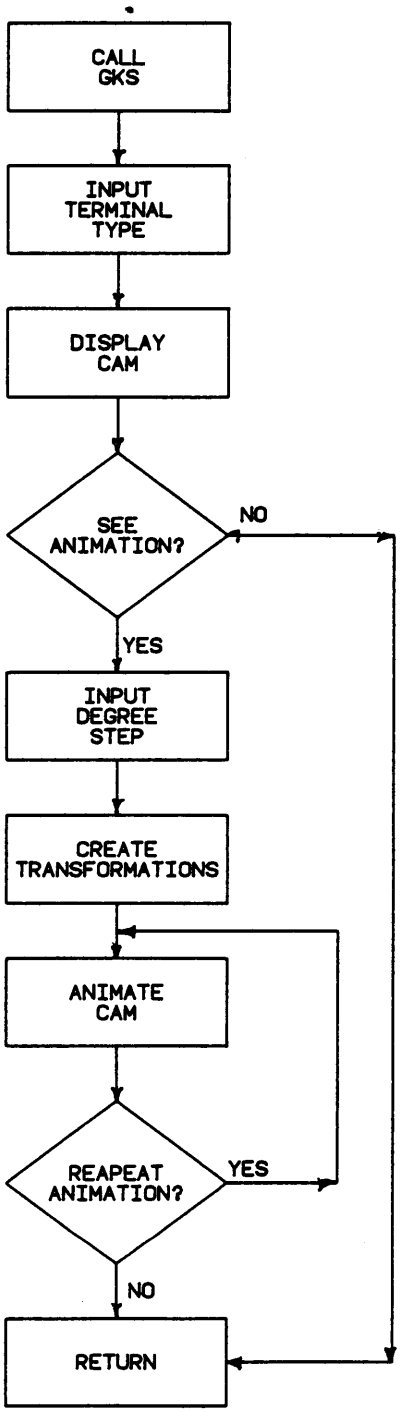


Figure 38. Flow chart for display using GKS.

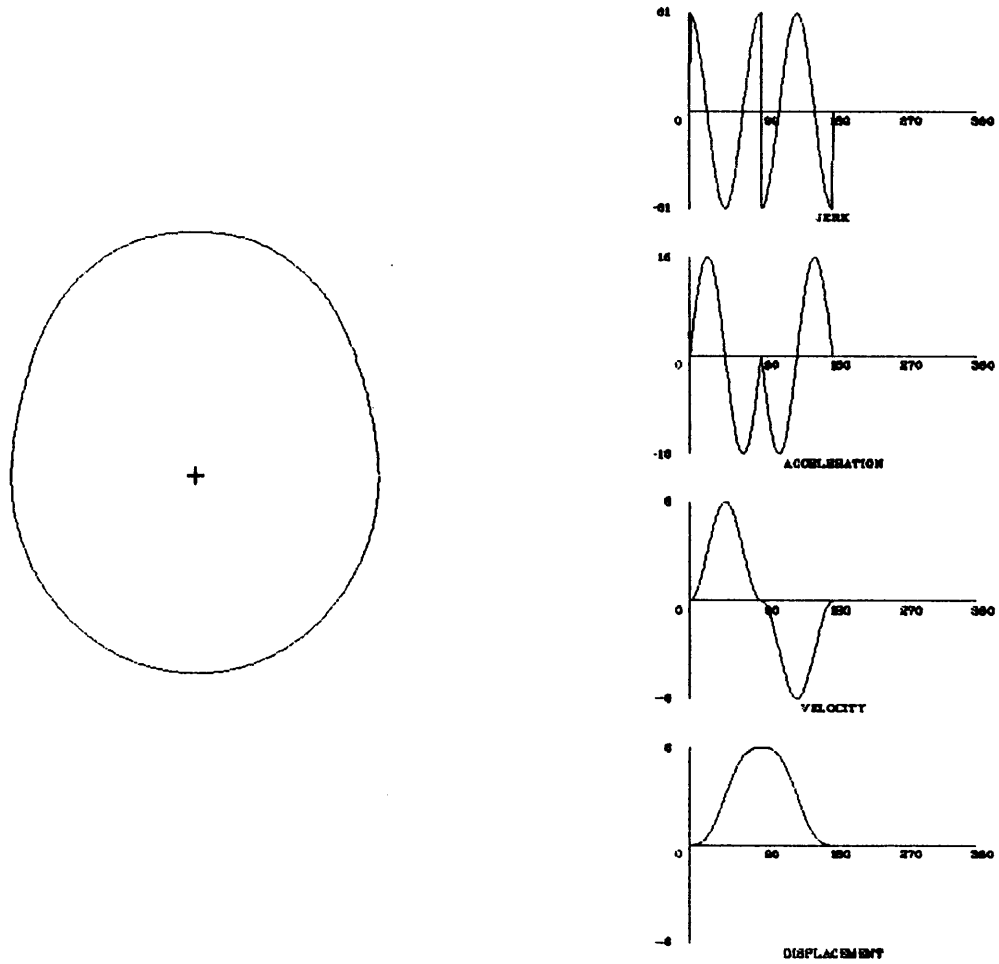


Figure 39. Example of GKS display.

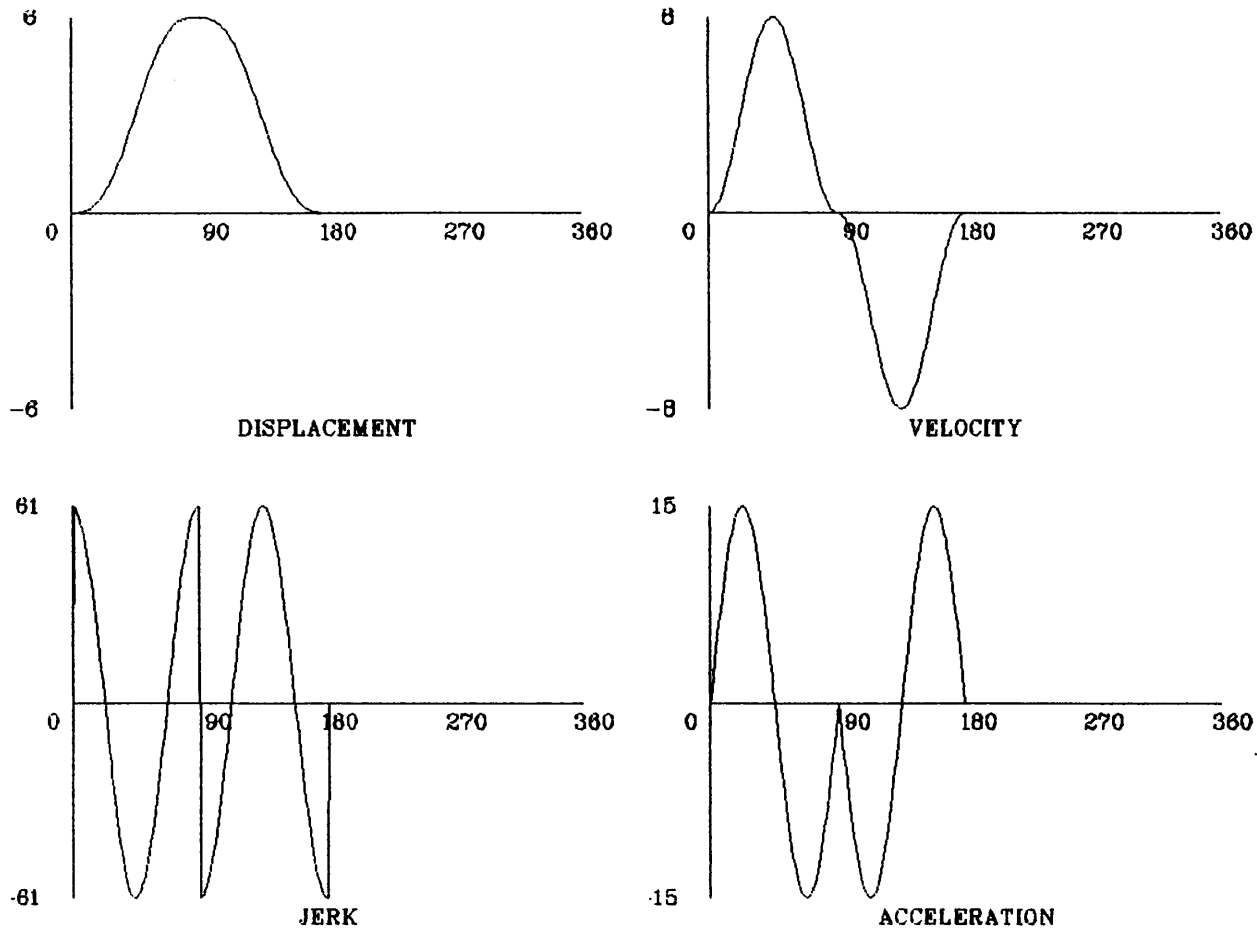


Figure 40. Example of GKS display.

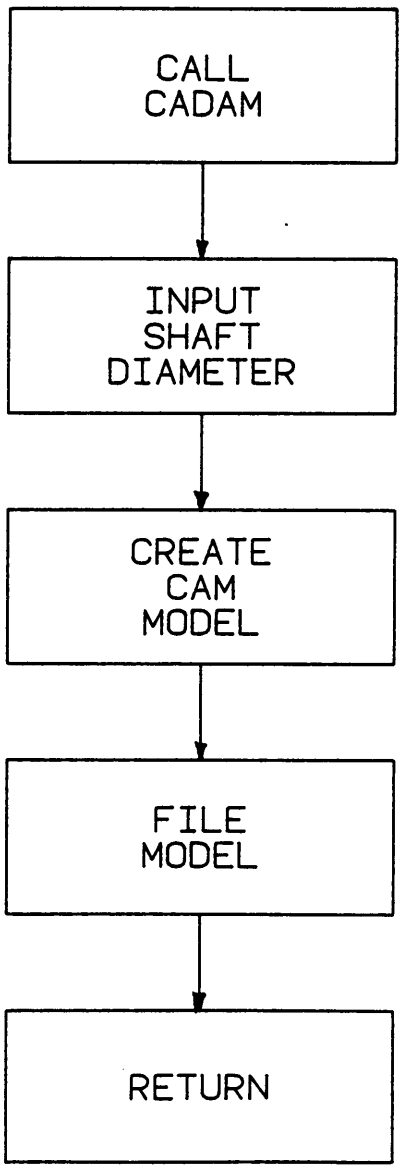


Figure 41. Flow chart for display by CADAM.

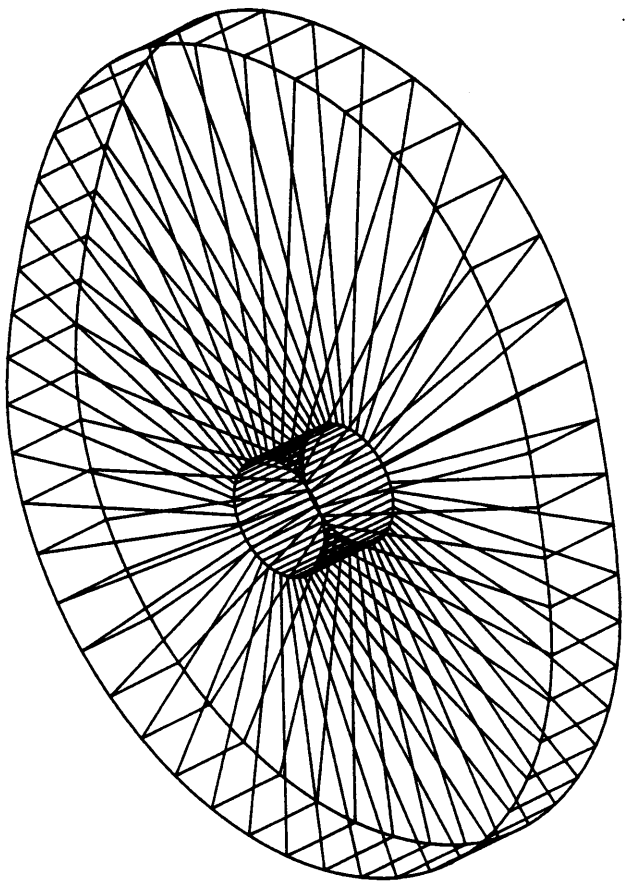

Figure 42. Example of CADAM display.

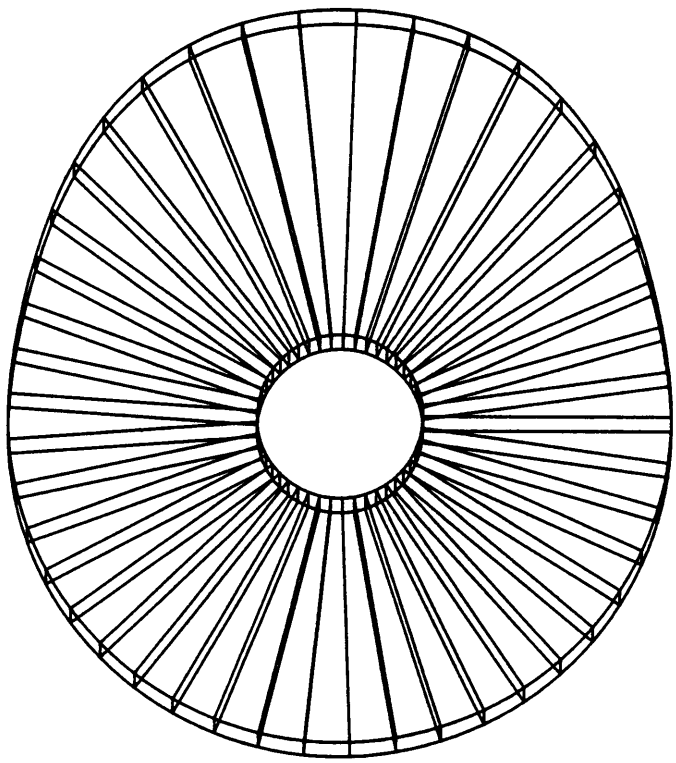

Figure 43. Example of CADAM display.

FILE.CAM, a file which has been formatted on lines of the geometry files created by the UTILITY subroutine of MOVIE.BYU. The creation of FILE.CAM enables the user to run DISPLAY without running UTILITY. The user will be asked for the name of the geometry file created using ANICAM. DISPLAY allows the model to be displayed in 3-D and be viewed at various angles. Hidden line elimination can be performed by giving a series of DISPLAY commands. If the user has chosen to animate this model while in ANICAM, he has created a command file, FILE.COM, which can be used by a modified version of DISPLAY to create and store all the frames of motion on a flexible disc. These frames can be animated by running ANIMATE (**). ANIMATE loads the data from the disc to the memory of the TEKTRONIX terminal, and sequentially displays each frame with refresh or raster graphics, giving an illusion of motion. A detailed list of all the DISPLAY commands and the format of the geometry file created are given in Appendix A. Figure 44 shows the flow chart for the MOVIE display subroutine. Selected examples of the output obtained by this display system are shown in Figs. 45 and 46.

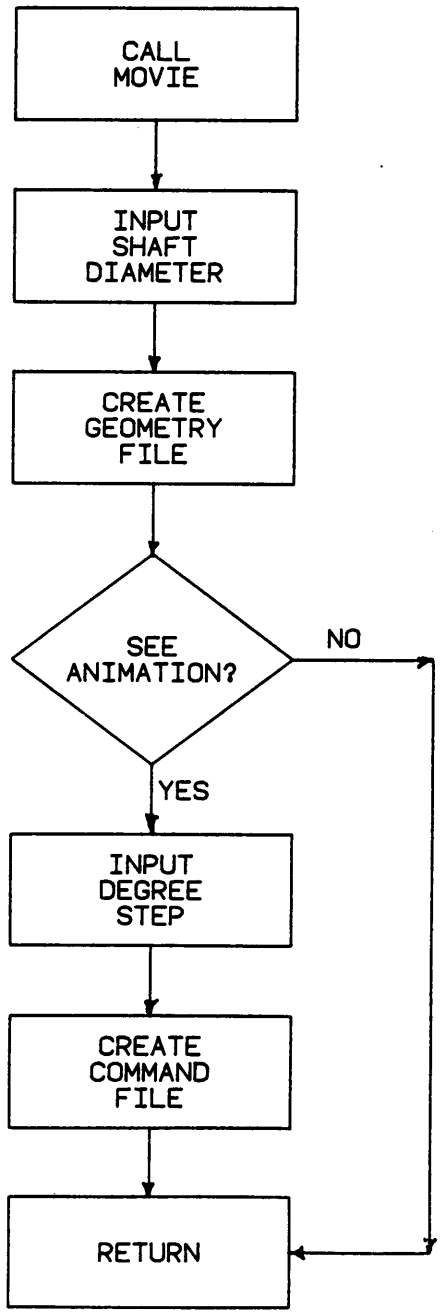


Figure 44. Flow chart for display using MOVIE.

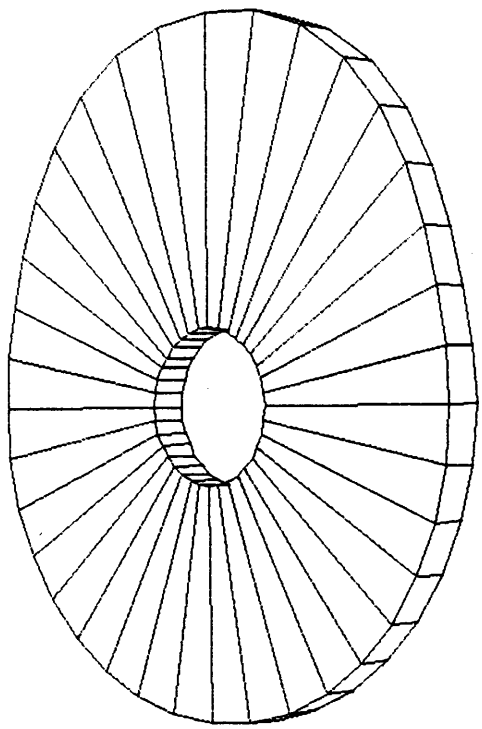

Figure 45. Example of MOVIE display.

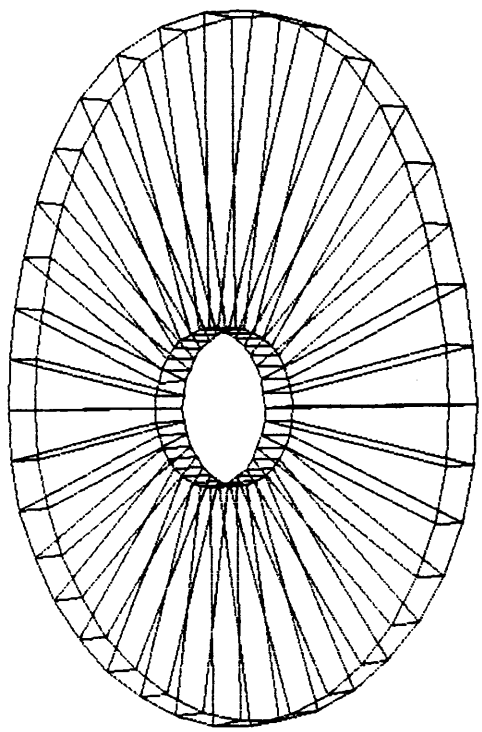

Figure 46. Example of MOVIE display.

# CHAPTER 6 RECOMMENDATIONS AND IMPLICATIONS

ANICAM is intended to be an interactive program which has the capability to design and display a cam model, and the motion characteristics which will be imparted to the follower. With full color 3-D capabilities, it can be used as a powerful tool to create wire-frame or solid models by linking it with a CAD/CAM package, and a widely used graphic display package viz. CADAM and MOVIE.BYU. While CADAM allows the analysis of the physical properties of the cam designed and futher use in design or manufacturing, the ANICAM link with MOVIE and GKS can be used as one of the most powerful tool for purpose of simulation. ANICAM can be seen having many industrial applications.

Though the system has fulfilled most of the requirements of designing principles it has a tremendous potential for increased capabilities. This design is based on kinematic considerations only, and does not consider the dynamic effects on the cam. While the cam must be developed to meet motion constraints, care must be taken to see that the kinematically sound design does not fail dynamically. Dynamic effects like the loads acting on the cam, vibrations, and the stresses caused by both may also lead to the cam not being able to perform the task it was created for. In order to make the design more comprehensive and reliable, it is recommended that dynamic analysis be incorporated in the program.

The ability of the program to use 3-D display systems brings the design of 3-D cams to the mind. 3-D cams are finding increasing applications in industries. The present ability of ANICAM to display 3-D

models would be highly enhanced by the capability of solid modelling. Solid modelling packages like CATIA can be easily incorporated into the program giving it the added ability to shade, giving the model a more realistic look. These abilities are expected to be available in CADAM in the near future.

The ability of the system to simulate the cam rotation would be put to better use if the follower is displayed and animated along with the cam. This would give the designer a better perspective of what to expect of the designed system and the actual design models which can be modified interactively.

One of the big advantages of the program structure is the separation of the design and display routines. By simply changing the design routine it would be possible to display other machine components like gears, linkages, pulleys, bearings, joints, chains, brakes, and clutches. It is also possible to add the design routines of the above components to ANICAM thus increasing the scope of the program.

## REFERENCES


1. Rothbart, H. H., Cams, John Wiley and Sons, 1956.

2. Jensen, P. W., Cam Design and Manufacture, The Industrial Press, 1965.

3. Chen, F. Y., Mechanisms and Design of Cam Mechanisms, Pergamon Press, 1982.

4. Johnson, R. C., "Force reduction by motion design in spring loaded cam mechanism", Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 106, pp 278, 1984.

5. Donkin, W. T. and Clark, H. H., "The Electric Telemeter and Valve Spring Surge", Transactions of SAE, Vol. 24, pp 687-700, 1979.

6. Pisano, A. P. and Freudenstein, F., "An experimental and analytical investigation of the dynamic response of the high speed cam follower system", Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 105, pp 692-704, 1984.

7. Berzak N., "Optimization of cam follower system in polydyne cam design", Doctoral thesis, Columbia University, 1979.

8. Sermon, C. F. and Liniecki, A., "Search for optimum solution of a single disk cam mechanism with an oscillating roller follower", ASME Pap. 72-MECH-61, 1972.

9. Sandler, B. Z., "Suboptimal dynamic synthesis of linearized mechanical system", ASME Pap. 74-DET-27, 1974.

10. Chen, F. Y. and Shah, A. M., "Optimal design of geometric parameters of a cam mechanism using sequential random vector technique", ASME Pap. 72-Mech-73, 1972.

11. Chew, M., Freudenstein, F., and Longman, R. W., "Application of optimal control theory to the synthesis of high speed cam follower systems", Journal of Mechanisms, Transmissions, and Automation in Design Vol. 105, pp 576-591, 1984.

12. Lafuente, J. M., "Interactive graphics in data processing cam design on graphics console", IBM Syst. J:3-4, pp 325-343, 1968.

13. De Fraine, J., "Integration of computer aided design and computer aided manufacturing for cam driving mechanisms", Proceedings of 5th world congress on theory of machines and mechanisms, pp 122-125, 1969.

14. Tesar, D., and Mathew, G. K., The dynamic synthesis, analysis and design of modeled cam systems, Lexington Books, Lexington, Massechusetts, 1976.

15. Vallard, H., "Computer programs for analysis of valve train dynamics and cam profile evaluation", The Institute of Internal Combustion Engines Bulletin No. IF-H5, Trondheim, Norway, 1969.

16. Keil, M. J., "A Method for Automatically Generating and Animating 3-D Models of Planar Linkages", Master's thesis, Florida Atlantic University, 1984.

17. Mabie, H. H., and Ocvirk F. W., Mechanisms and Dynamics of Machinery, John Wiley and Sons, 1978.

18. Ballaney, P. L., Theory of Machines, Khanna Publishers, New Delhi, 1980.

19. Tausworthe, R. C., Standardized Development of Computer Software (Part 1), Prentice-Hall Inc., 1977.

20. Foley, J. D., Vam Dam, A., Fundamentals of Interactive Computer Graphics, Addison-Wesley Publishing Company, 1984.

21. Sproull, R. F., Sutherland, W. R., Ullner, M. K., Device-Independent Graphics, McGraw-Hill Book Company, 1985.

22. Hopgood, F. R. A., Duce, D. A., Gallop, J. R., Sutcliffe, D. C., Introduction to the Graphical Kernel System, Academic Press, 1983.

23. Myklebust, A., "ANIMATE", Florida Atlantic University, 1983.

24. Brigham Young University, MOVIE.BYU.-- General Purpose Computer Graphics System, Depatment of Computer Science,1984.

25. CADAM Inc., Computer-Graphics Augmented Design and Manufacturing System - User Training Manual SH20-2035-3, IBM, 1982.

26. CADAM Inc., CADAM System - Geometry Interface Module Installation and Programmers Guide SH20-2099-6, IBM, 1983.

27. CADAM Inc., CADAM System - Geometry Interface Module Installation Guide SH20-6227-0, IBM, 1985.

# APPENDIX A.

## Subroutines used for GKS.

1. Activate workstation

   CALL GACWK(WKID)

   WKID = workstation identifier.

2. Close GKS

   CALL GCLKS

3. Close segment

   CALL GCLSG

4. Close workstation

   CALL GCLWK(WKID)

   WKID = workstation identifier.

5. Create segment

   CALL GCRSG(SGNA)

   SGNA = segment name

6. De-activate workstation

   CALL GDAWK(WKID)

   WKID = workstation identifier.

7. Evaluate transformation matrix

   CALL GEVTM(XO,YO,DX,DY,PHI,FX,FY,SW,MOUT)

   XO = x coordinate of the fixed point

   YO = y coordinate of the fixed point

   DX = shift vector x coordinate

DY = shift vector y coordinate

PHI = Rotation angle in radians

FX = x coordinate scale factor

FX = x coordinate scale factor

SW = Coordinate system

MOUT = Resulting transformation matrix

8. Draw fill area

CALL GFA(N,PX,PY)

N = Number of points

PX(N) = x coordinates of points in WC

PY(N) = y coordinates of points in WC

9. Open GKS

CALL GOPKS(ERRFIL)

ERRFIL = Error message file.

10. Open workstation

CALL GOPWK(WKID,CONID,WTYPE)

WKID = workstation identifier.

CONID = Connection identifier

WTYPE = Workstation type

11. Draw polyline

CALL GPL(N,PX,PY)

N = Number of points

PX(N) = x coordinates of points in WC

PY(N) = y coordinates of points in WC

12. Draw polymarker

CALL GPM(N,PX,PY)

N = Number of points

PX(N) = x coordinates of points in WC

PY(N) = y coordinates of points in WC

13. Set character height.

CALL GSCHH(CHH)

CHH = Character height in WC

14. Set character expansion factor.

CALL GSCHXP(CHXP)

CHXP = Character expansion factor

15. Set normalization transformation.

CALL GSELNT(TNR)

TNR = Transformation identifier

16. Set fill area color index.

CALL GSFACI(COLI)

COLI = Integer number corresponding to one color

17. Set fill area interior style.

CALL GSFAIS(INST)

INST = Integer number corresponding one style

18. Set line type.

CALL GSLN((LTYP)

LTYP = Integer number corresponding to one line type

19. Set marker type.

CALL GSMK(MTYPE)

MTYP = Integer number corresponding to one marker type

20. Set polyline color index.

CALL GSPLCI(COLI)

COLI = Integer number corresponding to one color

21. Set polymarker color index.

CALL GSPMCI(COLI)

COLI = Integer number corresponding to one color

22. Set segment transformation.

CALL GSSGT(SGNA,M)

SGNA = Segment name

M = Transformation matrix

23. Set text alignment.

CALL GSTXAL(TXALH,TXALV)

TXALH = Horizontal alignment of text

TXALV = Vertical alignment of text

24. Set text color index.

CALL GSTXCI(COLI)

COLI = Integer number corresponding to one color

25. Set text font and precision.

CALL GSTXFP(FONT,PREC)

FONT = Integer corresponding to one font

PREC = Integer corresponding to one precision value

26. Set segment visibility.

CALL GSVIS(SGNA,I)

SGNA = Segment name

I = 0 for invisible, 1 for visible

27. Set viewport.

CALL GSVP(TNR,XMIN,XMAX,YMIN,YMAX)

TNR = Transformation identifier

XMIN = Minimum value of x in NDC coordinates

XMAX = Maximum value of x in NDC coordinates

YMIN = Minimum value of y in NDC coordinates

YMIN = Maximum value of y in NDC coordinates

28. Set workstation viewport.

CALL GSWKVP(WKID,XMIN,XMAX,YMIN,YMAX)

WKID = Workstation identifier

XMIN = Minimum value of x in DC coordinates

XMAX = Maximum value of x in DC coordinates

YMIN = Minimum value of y in DC coordinates

YMIN = Maximum value of y in DC coordinates

29. Set workstation window.

CALL GSWKWN(WKID,XMIN,XMAX,YMIN,YMAX)

WKID = Workstation identifier

XMIN = Minimum value of x in NDC coordinates

XMAX = Maximum value of x in NDC coordinates

YMIN = Minimum value of y in NDC coordinates

YMIN = Maximum value of y in NDC coordinates

30. Set window.

CALL GSWN(TNR,XMIN,XMAX,YMIN,YMAX)

TNR = Transformation number

XMIN = Minimum value of x in WC coordinates

XMAX = Maximum value of x in WC coordinates

YMIN = Minimum value of y in WC coordinates

YMIN = Maximum value of y in WC coordinates

31. Draw text.

CALL GTX((PX,PY,CHARS)

PX = x coordinate of text postion in WC

PY = y coordinate of text postion in WC

CHARS = String of characters

32. Update workstation.

CALL GUWK(WKID,REGFL)

WKID = Workstation identifier

REGFL = Extent of update functions

## Subroutines used for CADAM.

1. Begin A Trap for View Geometry

CALL BEGVU(IOPT,IDVU,XYZ,XVCTR,YVCTR,ZVCTR,*ERROR)

IOPT = Option flag which must be set to one. This flag eventually determines the handling of input coordinates.

IDVU = 4 characters, of which the last two are used 'xxID'

XYZ = An array XYZ(3) representing the X, Y, and Z location of the view.

XVCTR = An array XVCTR(3) representing the direction cosines of the new X-axis.

YVCTR = An array YVCTR(3) representing the direction cosines of the new Y-axis.

ZVCTR = An array ZVCTR(3) representing the direction cosines of the new Z-axis.

*ERROR = ERROR. A view has already been started, or no more space for views.

2. File the Drawing

   CALL CADFIL(IOPT,NOGOOD,IDUMMY)

   IOPT = 0 File a drawing.

   1 Retrieve a drawing.

   2 Overfile an drawing.

   NOGOOD = An integer to be tested after filing. If not equal to one, the file call was no good.

   IDUMMY = Dummy varailble for later use.

3. Initialize a model.

   CALL CADST(DRAWID,EMPNO,GROUP)

   DRAWID = 5 word array containg drawid.

   EMPNO = 2 word array containing the employee ID.

   GROUP = One word array containing the group ID.

4. Draw a 3-D line.

   CALL LINE3D(XYZ1,XYZ2,*ERROR)

   XYZ1 = An array(3) containing the X, Y, Z absolute coordinates of one of the end points of the line.

   XYZ2 = An array(3) containing the X, Y, Z absolute coordinates of the other end point of the line.

   *ERROR = Error return indicating that the length of the line is less than the tolerance limit of 0.001.

5. Draw a 3-D Spline.

   CALL SPLINE(N,M,K,D,S,T1,T2,T3,*ERROR)

   N = Number of input points

M = Number of dimensions in space.(1,2 or 3)

D(M,N) = A floating point array containing spline points coordinate data.

V(M,N) = A floating point array of spline tangent vector components.

S(N) = A floating point array of chord lengths for bays defined by the input data.

T1 = Temporary floating point working area, must be of length N.

T2 = Temporary floating point working area, must be of length N.

T3 = Temporary floating point working area, must be of length N.

*ERROR = Denotes label in the program which subroutine returns in case of an error.

## Writing the Geometry file for MOVIE.BYU.

FILE.CAM generated by ANICAM is written in the same format as the geometry file generated by using the UTILITY program of MOVIE.BYU. The format used in writing the geometry file is as follows:

If the variables used are defined as

NP = Number of parts

NJ = Number of nodes or joints

NPT = Number of elements or polygons

NCON = Number of entries in the connectivity array

NPL = The parts list

X = The coordinates of the nodes

IP = The connectivity of the elements or polygons

then the geometry file is written using the following fortran statements:

```
      WRITE(10,100) NP,NJ,NPT,NCON
      WRITE(10,100) ((NPL(I,IJ),I=1,2),IJ=1,NP)
      WRITE(10,200) ((X(I,IJ),I=1,3),IJ=1,NJ)
      WRITE(10,100) (IP(I),I=1,NCON)
  100 FORMAT(16I5)
  200 FORMAT(6E12.5)
```

More detailed information can be got by referring to any of the manuals available on GKS, CADAM, and MOVIE.BYU.

# APPENDIX B.

This is the exec for ANICAM. This exec asks the user to choose the display system on which the cam model is to be displayed. It allows the user to QUIT from the program any time before the execution begins.

```
&TRACE OFF
FILEDEF 7 DISK CAM OUT A
FILEDEF 8 DISK FILE CAM A
FILEDEF 9 DISK FILE COM A
-5
CLEAR
&LOOP 1 12
&TYPE
&BEGTYPE 8
                              SELECT DISPLAY SYSTEM

                              1.  GKS
                              2.  CADAM
                              3.  MOVIE.BYU
                              4.  NO DISPLAY

                           Type QUIT to exit from program.
&READ VARS &MONI
CLEAR
&IF .&MONI EQ .1 &GOTO -10
&IF .&MONI EQ .2 &GOTO -20
&IF .&MONI EQ .3 &GOTO -30
&IF .&MONI EQ .4 &GOTO -40
&IF .&MONI EQ .QUIT &GOTO -50
&GOTO -5
-10
&LOOP 1 10
&TYPE
&BEGTYPE 7
                              Hit RETURN to load for GKS.
                              Execution will begin 30 seconds
                              after hitting RETURN key.
                              If selection is to be changed,
                              type INVALID to return to menu.

                              Type QUIT to exit from the program.
&READ VARS &DUMB
&IF .&DUMB EQ .INVALID &GOTO -5
&IF .&DUMB EQ .QUIT &GOTO -50
LOAD ANICAM1 (START
```

```
&GOTO -50
-20
&LOOP 1 10
&TYPE
&BEGTYPE 7
                              Hit RETURN to load for CADAM.
                              Execution will begin 45 seconds
                              after hitting RETURN key.
                              If selection is to be changed,
                              type INVALID to return to menu.

                              Type QUIT to exit from the program.
&READ VARS &DUMB
&IF .&DUMB EQ .INVALID &GOTO -5
&IF .&DUMB EQ .QUIT &GOTO -50
EXEC CADCDLNK ANICAM2
EXEC CADCD MECH ANICAM2
&GOTO -50
-30
&LOOP 1 10
&TYPE
&BEGTYPE 7
                              Hit RETURN to load for MOVIE.BYU.
                              Execution will begin 5 seconds
                              after hitting RETURN key.
                              If selection is to be changed,
                              type INVALID to return to menu.

                              Type QUIT to exit from the program.
&READ VARS &DUMB
&IF .&DUMB EQ .INVALID &GOTO -5
&IF .&DUMB EQ .QUIT &GOTO -50
LOAD ANICAM3 (START
&GOTO -50
-40
&LOOP 1 10
&TYPE
&BEGTYPE 7
                              Hit RETURN to load for SCREEN.
                              Execution will begin 5 seconds
                              after hitting RETURN key.
                              If selection is to be changed,
                              type INVALID to return to menu.

                              Type QUIT to exit from the program.
&READ VARS &DUMB
&IF .&DUMB EQ .INVALID &GOTO -5
&IF .&DUMB EQ .QUIT &GOTO -50
LOAD ANICAM4 (START
-50
CLEAR
&EXIT
```

# APPENDIX C.

```
**********************************************************************************
**********************************************************************************
**                                                                              **
**                                  ANICAM                                      **
**                                                                              **
**                                                                              **
**    THIS PROGRAM ALLOWS THE USER TO DESIGN A CAM BASED ON MOTION              **
**    CONSIDERATIONS.  THE USER HAS A CHOICE OF DISPLAYING THE CAM              **
**    ON 3 DIFFERENT GRAPHICS SYSTEMS.  AT THE SAME TIME HE CAN                 **
**    DOCUMENT THE OUTPUT PARAMETERS LIKE MOTION CHARACTERISTICS                **
**    AND CAM PROFILE COORDINATES.                                              **
**    TWO OF THE SYSTEMS, NAMELY, GKS AND MOVIE.BYU, ALLOW THE USER             **
**    TO ANIMATE (SIMULATE THE MOTION OF) THE CAM.                              **
**                                                                              **
**********************************************************************************
**********************************************************************************
```

This is the alphabetical list of the main variables that have been used in the program, ANICAM.

1. A: An array of 361 real numbers representing the acceleration of the follower at one degree angular displacement of the cam.
2. ALPHA: An array of 361 real numbers representing the pressure angle at one degree angular displacement of the cam.
3. B: Parameter with value 1125./1192.
4. BASE: This is a real number representing the base radius of the cam.
5. BETA: This is a real number representing the active cam angle for each cam segment. (radians)
6. C: Integer number representing the UPTO value of the M-1th segment.

7. DESI: This is an integer number representing the design technique to be adopted.
8. E: An array of N integer numbers representing the type of curve selected.
9. ECC: This is a real number representing the eccentricity of the follower with the cam center.
10. FROM: An array of N integer numbers representing the start of each cam segment. (degrees)
11. GAMMA: This is a real number representing the ratio THETA/BETA.
12. H: An array of N real numbers representing the amount of lift, drop, dwell or swing of the follower.
13. I: Counter
14. J: An array of 361 real numbers representing the jerk on the follower at one degree angular displacement of the cam.
15. L: Counter used for GKS.
16. M: Counter used for computing the motion characteristics.
17. MONI: Integer number used for selecting the type of display system.
18. N: Integer number used to represent number of segments in the cam.
19. OUT: This is an integer number representing the parameters to be displayed on the monitor.
20. PI: Parameter with value 3.1415926
21. R: Integer number representing selection of nature of design changes to be made.
22. RA: This is a real number representing the distance between the cam center and the swinging follower.

23. RF: This is a real number representing the length of the follower arm.
24. ROLL: This is a real number representing the radius of the roller follower.
25. RPM: This is an integer number representing the rotational speed of the cam in rotations per minute.
26. S: An array of 361 real numbers representing the displacement of the follower at one degree angular displacement of the cam.
27. THETA: This is a real number representing the angular dispalcement of the cam. (radians)
28. TYPE: This is an integer number representing the type of follower selected.
29. UPTO: An array of N integer numbers representing the start of each cam segment. (degrees)
30. V: An array of 361 real numbers representing the velocity of the follower at one degree angular displacement of the cam.
31. WIDTH: This is a real number representing the width of the cam.
32. X: An array of 361 real numbers representing the x-coordinate of the cam profile at one degree interval.
33. Y: An array of 361 real numbers representing the y-coordinate of the cam profile at one degree interval.
34. Z: An array of 361 real numbers representing the z-coordinate of the cam profile at one degree interval.

**********
** **
** MAIN PROGRAM **
** **

```
**********************************************************************************
**********************************************************************************
**                                                                              **
**    THE MAIN PORTION OF THE PROGRAM OPERATES AS THE NERVE CENTER               **
**    OF THE SYSTEM.  IT CALLS VARIOUS SUBROUTINES TO DESIGN THE                **
**    CAM.  IT ALLOWS REPEATED CHANGES IN THE DESIGN TILL THE USER              **
**    IS SATISFIED WITH THE DESIGN OR DECIDES TO QUIT.  THIS PART               **
**    OF THE PROGRAM ALSO ALLOWS THE USER TO SAVE THE OUTPUT IN                 **
**    FORM OF AN OUTPUT FILE CALLED CAM OUT.                                    **
**                                                                              **
**********************************************************************************
**********************************************************************************


      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

      CHARACTER CHAR

      PI = 3.1415926
      B = 1125./1192.
      L = 0

   10 FORMAT(A1)

  100 CALL FOLLOW
      CALL BASIC
      CALL OUTPUT
      CALL DESIGN
      CALL MOTION
      CALL DISPLY
  200 CALL DISVEL
      CALL CURVES
      CALL PROFIL

  300 WRITE(6,*) '              Do you want to change the design? (Y/N)'
      READ(5,10) CHAR
```

```
      IF (CHAR .EQ. 'Y') THEN
           CALL CHANGE
           GOTO 200
      ENDIF

      WRITE(6,*) '                Do you want to save this Design? (Y/N)'
      READ(5,10) CHAR

      IF (CHAR .EQ. 'Y') CALL DOCUM

      WRITE(6,*) '              Do you want to create a new Design? (Y/N)'
      READ(5,10) CHAR

      IF (CHAR .EQ. 'Y') GOTO 100

      WRITE(6,*) '              Are you ready to quit? (Y/N)'
      READ(5,10) CHAR

      IF (CHAR .NE. 'Y') GOTO 300

      STOP
      END
```

```
**********************************************************************************
**********************************************************************************
**                                                                              **
**                          SUBROUTINE FOLLOW                                   **
**                                                                              **
**********************************************************************************
**********************************************************************************
**                                                                              **
**    THIS SUBROUTINE ASKS THE USER TO SELECT THE FOLLOWER TYPE.                **
**    DEPENDING ON THE TYPE OF FOLLOWER CHOSEN THE USER IS ASKED                **
**    TO INPUT SECONDARY INFORMATION ABOUT THE FOLLOWER.  THIS                  **
**    INFORMATION IS PASSED TO THE MAIN PROGRAM WHICH IN TURN                   **
**    WILL PASS IT TO THE OTHER SUBROUTINES.                                    **
**                                                                              **
**********************************************************************************
**********************************************************************************
```

```
      SUBROUTINE FOLLOW

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
```

```
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

100   WRITE(6,*) '                        Please Enter Follower Type'
      WRITE(6,*) ' '
      WRITE(6,*) ' '
      WRITE(6,*) '                        1: Knife Edge'
      WRITE(6,*) '                        2: Radial Translating Roller'
      WRITE(6,*) '                        3: Offset Translating Roller'
      WRITE(6,*) '                        4: Swinging Roller'
      WRITE(6,*) '                        5: Translating Flat Faced'
      WRITE(6,*) '                        6: Swinging Centric Flat-Faced'
      WRITE(6,*) '                        7: Swinging Eccentric Flat-Faced'
      READ(5,*) TYPE

      IF (TYPE .LT. 1 .OR. TYPE .GT. 7) THEN
           WRITE(6,*) '                          ERROR!! INVALID SELECTION!'
           GOTO 100
      ENDIF

      IF (TYPE .GT. 1 .AND. TYPE .LT. 5) THEN
200        WRITE(6,*) '                          Please enter roller radius'
           READ(5,*) ROLL

           IF(ROLL .LT. 0) THEN
      WRITE(6,*) '                     ERROR!! RADIUS MUST BE POSITIVE'
                GOTO 200
           ENDIF

      ENDIF

      IF (TYPE .EQ. 3 .OR. TYPE .EQ. 7) THEN
           WRITE(6,*) '                          Please enter eccentricity'
           READ(5,*) ECC
      ENDIF

      IF (TYPE .EQ. 4 .OR. TYPE .EQ. 6 .OR. TYPE .EQ. 7) THEN
300        WRITE(6,*) '                          Enter center distance between'
           WRITE(6,*) '                          Cam Center and Follower Arm.'
           READ(5,*) RA

         IF(RA .LT. 0) THEN
      WRITE(6,*) '                     ERROR!! DISTANCE MUST BE POSITIVE'
             GOTO 300
         ENDIF
```

```
      ENDIF

      IF (TYPE .EQ. 4 .OR. TYPE .EQ. 6) THEN
  400 WRITE(6,*) '                    Enter length of the Follower Arm'
      READ(5,*) RF

           IF(RF .LT. 0) THEN
         WRITE(6,*) '                      ERROR!! LENGTH MUST BE POSITIVE'
                GOTO 400
           ENDIF

      ENDIF

      RETURN
      END
```

```
*****************************************************************************
*****************************************************************************
**                                                                         **
**                          SUBROUTINE BASIC                               **
**                                                                         **
*****************************************************************************
*****************************************************************************
**                                                                         **
**    THIS SUBROUTINE ASKS THE USER TO INPUT BASIC INFORMATION  OF         **
**    THE CAM THAT HAS TO BE DESIGNED.  THE USER IS ASKED TO INPUT         **
**    THE BASE RADIUS, ROTATATIONAL SPEED AND THE WIDTH OF THE CAM.        **
**    THIS INFORMATION IS PASSED TO THE MAIN PROGRAM WHICH IN TURN         **
**    WILL PASS IT TO THE OTHER SUBROUTINES.                               **
**                                                                         **
*****************************************************************************
*****************************************************************************
```

```
      SUBROUTINE BASIC

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
```

```
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

  100 WRITE(6,*) '                    Please enter base radius'
      READ(5,*) BASE

      IF(BASE .LT. 0) THEN
         WRITE(6,*) '                       ERROR!! RADIUS MUST BE POSITIVE'
         GOTO 100
      ENDIF

  200 WRITE(6,*) '                    Please Enter RPM'
      READ(5,*) RPM

      IF(RPM .LT. 0) THEN
           WRITE(6,*) '                        ERROR!! RPM MUST BE POSITIVE'
           GOTO 200
      ENDIF

  300 WRITE(6,*) '                    Please enter Width'
      READ(5,*) WIDTH

      IF(WIDTH .LT. 0) THEN
          WRITE(6,*) '                        ERROR!! WIDTH MUST BE POSITIVE'
          GOTO 300
      ENDIF

      RETURN
      END
```

```
**********************************************************************************
**********************************************************************************
**                                                                              **
**                          SUBROUTINE OUTPUT                                   **
**                                                                              **
**********************************************************************************
**********************************************************************************
**                                                                              **
**    THIS SUBROUTINE ASKS THE USER TO SELECT THE GRAPHICS OUTPUT               **
**    WHICH HE/SHE DESIRES.  THE USER HAS THREE CHOICES.  HE CAN                **
**    SELECT TO SEE THE PROFILE EITHER/OR/AND THE MOTION CURVES.                **
**    THIS INFORMATION IS PASSED TO THE MAIN PROGRAM WHICH IN TURN              **
**    WILL PASS IT TO THE OTHER SUBROUTINES.                                    **
**                                                                              **
**********************************************************************************
**********************************************************************************
```

```
      SUBROUTINE OUTPUT

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

  100 WRITE(6,*) '                    Please select type of Output'
      WRITE(6,*) ' '
      WRITE(6,*) ' '
      WRITE(6,*) '                    1: Profile only'
      WRITE(6,*) '                    2: Curves only'
      WRITE(6,*) '                    3: Profile and Curves'
      READ(5,*) OUT

      IF (OUT .LT. 1 .OR. OUT .GT. 3) THEN
           WRITE(6,*) '                        ERROR!! INVALID SELECTION!'
           GOTO 100
      ENDIF

      RETURN
      END
```

```
**********************************************************************************
**********************************************************************************
**                                                                              **
**                          SUBROUTINE DESIGN                                   **
**                                                                              **
**********************************************************************************
**********************************************************************************
**                                                                              **
**    THIS SUBROUTINE ASKS THE USER TO SELECT THE DESIGN PROCESS                **
**    TO BE ADOPTED IN ORDER TO DESIGN THE CAM.  ONLY THE FIRST                 **
**    OPTION IS OPERATING AT THIS MOMENT.  THIS SUBROUTINE WILL                 **
**    ALLOW THE OPTIMIZATION PROCESS TO BE INCORPORATED IN THE                  **
**    PROGRAM WITHOUT MANY COMPLICATIONS.                                       **
**                                                                              **
**********************************************************************************
**********************************************************************************
```

```
      SUBROUTINE DESIGN

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

  100 WRITE(6,*) '                    Please select Design Process'
      WRITE(6,*) ' '
      WRITE(6,*) ' '
      WRITE(6,*) '                    1: Conventional'
      WRITE(6,*) '                    2: Optimized'
      READ(5,*) DESI

      IF (DESI .LT. 1 .OR. DESI .GT. 2) THEN
           WRITE(6,*) '                        ERROR!! INVALID SELECTION!'
           GOTO 100
      ENDIF

      RETURN
      END
```

```
**********************************************************************************
**********************************************************************************
**                                                                              **
**                          SUBROUTINE MOTION                                   **
**                                                                              **
**********************************************************************************
**********************************************************************************
**                                                                              **
**    THIS SUBROUTINE ASKS THE USER TO DEFINE THE DISPLACEMENT                  **
**    CHARACTERISTICS EXPECTED OF THE CAM BEING DESIGNED.  IT IS                **
**    STRUCTURED SUCH THAT THE USER WILL BE PROMPTED TO GIVE THE                **
**    DISPLACEMENT REQUIREMENT FOR EACH SEGMENT OF THE CAM.  THIS               **
**    INFORMATION IS STORED AND PASSED TO THE CORRESPONDING                     **
**    SUBROUTINES THROUGH THE MAIN PROGRAM.                                     **
```

```
**                                                                              **
**********************************************************************************
**********************************************************************************

      SUBROUTINE MOTION

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

      I = 0
  100 I = I + 1
      WRITE(6,*) '                    Please enter Lift/Drop/Dwell/Swing'
      READ(5,*) H(I)

      IF (TYPE .EQ. 4 .OR. TYPE .EQ. 6 .OR. TYPE .EQ. 7) THEN
           H(I) = H(I)*PI/180.
      ENDIF

  200 WRITE(6,*) '                    Please Enter Duration'
      WRITE(6,*) '                    From and Upto'
      READ(5,*) FROM(I),UPTO(I)

      IF (UPTO(I) .GT. 360) THEN
         WRITE(6,*) '             ERROR!! DURATION MUST BE LESS THAN 360'
         GOTO 200
      ENDIF

      IF (H(I) .EQ. 0) THEN
         E(I) = 25
         GOTO 400
      ENDIF

  300 WRITE(6,*) '                          SELECT TYPE OF MOTION'
      WRITE(6,*) ' '
      WRITE(6,*) '                    1:            H-1 SHM'
      WRITE(6,*) '                    2:            H-2 SHM'
      WRITE(6,*) '                    3:            H-3 SHM'
```

```
      WRITE(6,*) '                    4:            H-4 SHM'
      WRITE(6,*) '                    5:            H-5 SHM'
      WRITE(6,*) '                    6:            H-6 SHM'
      WRITE(6,*) '                    7:         C-1 Cycloidal'
      WRITE(6,*) '                    8:         C-2 Cycloidal'
      WRITE(6,*) '                    9:         C-3 Cycloidal'
      WRITE(6,*) '                   10:         C-4 Cycloidal'
      WRITE(6,*) '                   11:         C-5 Cycloidal'
      WRITE(6,*) '                   12:         C-6 Cycloidal'
      WRITE(6,*) '                   13:      Constant  Velocity'
      WRITE(6,*) '                   14:      Constant Accl/Dccl'
      WRITE(6,*) '                   15:      2-3     Polynomial'
      WRITE(6,*) '                   16:      3-4-5   Polynomial'
      WRITE(6,*) '                   17:      4-5-6-7 Polynomial'
      WRITE(6,*) '                   18:    Gutman 1-3 Harmonic(90)'
      WRITE(6,*) '                   19: Freudenstein  1-3  Harmonic(84)'
      WRITE(6,*) '                   20: Freudenstein 1-3-5 Harmonic(84)'
      WRITE(6,*) '                   21:          Modified Sine'
      WRITE(6,*) '                   22:     Modified Trapezoidal'
      READ(5,*) E(I)

      IF (E(I) .LT. 1 .OR. E(I) .GT. 22) THEN
           WRITE(6,*) '                  ERROR!! INVALID SELECTION!'
           GOTO 300
      ENDIF

  400 IF (UPTO(I) .LT. 360) GOTO 100

      N = I

      RETURN
      END
```

```
**********************************************************************
**********************************************************************
**                                                                  **
**                      SUBROUTINE DISVEL                           **
**                                                                  **
**********************************************************************
**********************************************************************
**                                                                  **
**    THIS SUBROUTINE COMPUTES THE DISPLACEMENTS AND VELOCITIES AT  **
**    OF THE FOLLOWER AT ONE DEGREE INTERVALS.  THE SUBROUTINE IS   **
**    STRUCTURED SUCH THAT THE CALCULATIONS FOR EACH SEGMENT ARE    **
**    DONE INSIDE A DO LOOP.  THIS SUBROUTINE IS REQUIRED IN EACH   **
**    OF THE DISPLAY OPTIONS CHOSEN BY THE USER.  IT IS ALSO USED   **
**    IN THE COMPUTATIONS OF THE CAM PROFILE COORDINATES.           **
**                                                                  **
**********************************************************************
**********************************************************************
```

```
      SUBROUTINE DISVEL

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

      DO 4000 M= 1,N
      BETA = (UPTO(M) - UPTO(M-1))*PI/180.
      C = UPTO(M-1)
      THETA = 0.

      GO TO (100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,
     #       1400,1500,1600,1700,1800,1900,2000,2100,2200),E(M)

**************** If motion is DWELL then go to this section.***************

      DO 11 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C)
         V(I) = 0.
  11  CONTINUE

      GOTO 4000

**************** If motion is  H-1  then go to this section.***************

  100 DO 101 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C) + H(M)*(1 - COS(PI*GAMMA/2))
         V(I) = ((PI/2)*H(M)/BETA)*(SIN(PI*GAMMA/2))
  101 CONTINUE

      GOTO 4000

**************** If motion is  H-2  then go to this section.***************
```

```
  200 DO 201 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C) + H(M)*(SIN(PI*GAMMA/2))
         V(I) = ((PI/2)*H(M)/BETA)*(COS(PI*GAMMA/2))
  201 CONTINUE

      GOTO 4000

**************** If motion is  H-3  then go to this section.***************

  300 DO 301 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = - H(M)*(COS(PI*GAMMA/2))
         V(I) = ((PI/2)*H(M)/BETA)*(SIN(PI*GAMMA/2))
  301 CONTINUE

      GOTO 4000

**************** If motion is  H-4  then go to this section.***************

  400 DO 401 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = - H(M)*(1 - SIN(PI*GAMMA/2))
         V(I) = ((PI/2)*H(M)/BETA)*(COS(PI*GAMMA/2))
  401 CONTINUE

      GOTO 4000

**************** If motion is  H-5  then go to this section.***************

  500 DO 501 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C) + (H(M)/2)*(1 - COS(PI*GAMMA))
         V(I) = ((PI/2)*H(M)/BETA)*(SIN(PI*GAMMA))
  501 CONTINUE

      GOTO 4000

**************** If motion is  H-6  then go to this section.***************

  600 DO 601 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = - (H(M)/2)*(1 + COS(PI*GAMMA))
         V(I) = ((PI/2)*H(M)/BETA)*(SIN(PI*GAMMA))
  601 CONTINUE
```

```
      GOTO 4000

****************** If motion is  C-1  then go to this section.****************

  700 DO 701 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C) + H(M)*(GAMMA - (SIN(PI*GAMMA)/PI))
         V(I) = (H(M)/BETA)*( 1 - COS(PI*GAMMA))
  701 CONTINUE

      GOTO 4000

****************** If motion is  C-2  then go to this section.****************

  800 DO 801 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C) + H(M)*(GAMMA + (SIN(PI*GAMMA)/PI))
         V(I) = (H(M)/BETA)*( 1 + COS(PI*GAMMA))
  801 CONTINUE

      GOTO 4000

****************** If motion is  C-3  then go to this section.****************

  900 DO 901 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = - H(M)*( 1 - GAMMA + (SIN(PI*GAMMA)/PI))
         V(I) = (H(M)/BETA)*( 1 - COS(PI*GAMMA))
  901 CONTINUE

      GOTO 4000

****************** If motion is  C-4  then go to this section.****************

 1000 DO 1001 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = - H(M)*( 1 - GAMMA - (SIN(PI*GAMMA)/PI))
         V(I) = (H(M)/BETA)*( 1 + COS(PI*GAMMA))
 1001 CONTINUE

      GOTO 4000

****************** If motion is  C-5  then go to this section.****************

 1100 DO 1101 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C) + H(M)*(GAMMA - (SIN(2*PI*GAMMA)/(2*PI)))
```

```
        V(I) = (H(M)/BETA)*( 1 - COS(2*PI*GAMMA))
 1101 CONTINUE

      GOTO 4000

**************** If motion is  C-6  then go to this section.***************

 1200 DO 1201 I = FROM(M),UPTO(M)
        THETA = THETA + (PI/180.)
        GAMMA = THETA/BETA
        S(I) = - H(M)*( 1 - GAMMA + (SIN(2*PI*GAMMA)/(2*PI)))
        V(I) = (H(M)/BETA)*( 1 - COS(2*PI*GAMMA))
 1201 CONTINUE

      GOTO 4000

******** If motion is  Constant Velocity then go to this section.**********

 1300 DO 1301 I = FROM(M),UPTO(M)
        THETA = THETA + (PI/180.)
        S(I) = S(C) + (H(M)/BETA)*THETA
        V(I) = H(M)/BETA
 1301 CONTINUE
      GOTO 4000

******* If motion is Constant Acceleration then go to this section.********

 1400 DO 1401 I = FROM(M),UPTO(M)
        THETA = THETA + (PI/180.)
        GAMMA = THETA/BETA

        IF (GAMMA .LE. 0.5) THEN
          S(I) = S(C) + 2*H(M)*(GAMMA**2)
          V(I) = 4*H(M)*GAMMA/BETA
        ELSE
          S(I) = S(C) + H(M) - 2*H(M)*((1 - GAMMA)**2)
          V(I) = 4*H(M)*(1 - GAMMA)/BETA
        ENDIF

 1401 CONTINUE

      GOTO 4000

********** If motion is 2-3 Polynomial then go to this section.************

 1500 DO 1501 I = FROM(M),UPTO(M)
        THETA = THETA + (PI/180.)
        S(I) = S(C) + 3*(THETA**2) - 2*(THETA**3)
        V(I) = 6*THETA*(1 - THETA)
 1501 CONTINUE

      GOTO 4000
```

```
******** If motion is 3-4-5 Polynomial then go to this section. **********

 1600 DO 1601 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         S(I) = S(C) + 10*(THETA**3) - 15*(THETA**4) + 6*(THETA**5)
         V(I) = 30*(THETA**2) - 60*(THETA**3) + 30*(THETA**4)
 1601 CONTINUE

      GOTO 4000

******** If motion is 4-5-6-7 Polynomial then go to this section. ********

 1700 DO 1701 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         S(I) = S(C)+35*(THETA**4)-84*(THETA**5)+70*(THETA**6)
     #          -20*(THETA**7)
         V(I) = 140*(THETA**3)-420*(THETA**4)+420*(THETA**5)
     #          -140*(THETA**6)
 1701 CONTINUE

      GOTO 4000

******** If motion is Gutman 1-3 Harmonic then go to this section. ********

 1800 DO 1801 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C)+ H(M)*( GAMMA - 15*(SIN(2*PI*GAMMA))/(32*PI)
     #          -(SIN(6*PI*GAMMA))/(96*PI))
         V(I) = (H(M)/BETA)*(1-(15/16)*COS(2*PI*GAMMA)
     #          -COS(6*PI*GAMMA)/16)
 1801 CONTINUE

      GOTO 4000

**** If motion is Freudenstein 1-3 Harmonic then go to this section. ****

 1900 DO 1901 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C) + H(M)*GAMMA - (H(M)/(2*PI))*(27*SIN(2*PI*GAMMA)/28
     #           + SIN(6*PI*GAMMA)/84)
         V(I) = (H(M)/BETA)*(1-27*COS(2*PI*GAMMA)/28-COS(6*PI*GAMMA)/28)
 1901 CONTINUE

      GOTO 4000

** If motion is Freudenstein 1-3-5 Harmonic then go to this section. **

 2000 B = 1125./1192.
```

```
      DO 2001 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         S(I) = S(C) + H(M)*GAMMA - (H(M)*B/(2*PI))*(SIN(2*PI*GAMMA)
     #         + SIN(6*PI*GAMMA)/54 + SIN(10*PI*GAMMA)/1250)
         V(I) = (H(M)/BETA)*(1-B*(COS(2*PI*GAMMA)+COS(6*PI*GAMMA)/18
     #         + COS(10*PI*GAMMA)/250))
 2001 CONTINUE

      GOTO 4000

********** If motion is Modified Sine then go to this section. ************

 2100 DO 2101 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA

      IF (GAMMA .LT. 1/8) THEN
         S(I) = S(C) + H(M)*(0.43989*GAMMA - 0.035014*SIN(4*PI*GAMMA))
         V(I) = (0.43989*H(M)/BETA)*(1 - COS(4*PI*GAMMA))
      ELSE
        IF (GAMMA .LT. 7/8) THEN
           S(I) = S(C) + H(M)*(0.28005 + 0.43989*GAMMA
     #           - 0.315055*COS((4*PI/3)*GAMMA - PI/6))
           V(I) = (H(M)/BETA)*(0.43989 +1.31967*SIN((4*PI/3)*GAMMA-PI/6))
        ELSE
           S(I) = S(C) + H(M)*(0.56010 + 0.43989*GAMMA
     #           - 0.035014*SIN(2*PI*(2*GAMMA - 1)))
           V(I) = (H(M)/BETA)*(0.43989*(1 - COS(2*PI*(2*GAMMA - 1))))
        ENDIF

      ENDIF

 2101 CONTINUE

      GOTO 4000

******* If motion is Modified Trapezoidal then go to this section. ********

 2200 DO 2201 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA

         IF (GAMMA .LT. 1/8) GOTO 2210
         IF (GAMMA .LT. 3/8) GOTO 2220
         IF (GAMMA .LT. 1/2) GOTO 2230
         IF (GAMMA .LT. 5/8) GOTO 2240
         IF (GAMMA .LT. 7/8) GOTO 2250

      S(I) = S(C) + H(M)*(0.6110154 + 0.3889845*GAMMA
     #       + 0.0309544*SIN(PI*(4*GAMMA-3)))
      V(I) = (H(M)/BETA)*(0.3889845 + 0.3889845*COS(PI*(4*GAMMA-3)))
```

```
      GOTO 2201
2210  S(I) = S(C) + 0.09724612*H(M)*(4*GAMMA - SIN(4*PI*GAMMA)/PI)
      V(I) = (0.3889845*H(M)/BETA)*(1 - COS(4*PI*GAMMA))
      GOTO 2201
2220  S(I) = S(C) + H(M)*(2.44406184*(GAMMA**2) - 0.22203097*GAMMA
     #       + 0.00723407)
      V(I) = (H(M)/BETA)*(4.888124*GAMMA - 0.222031)
      GOTO 2201
2230  S(I) = S(C) + H(M)*(1.6110154*GAMMA-0.0309544*SIN(PI*(4*GAMMA-1))
     #       - 0.3055077)
      V(I) = (H(M)/BETA)*(1.6110154 - 0.3889845*COS(PI*(4*GAMMA - 1)))
      GOTO 2201
2240  S(I) = S(C) + H(M)*(1.6110154*GAMMA+0.0309544*SIN(PI*(4*GAMMA-2))
     #       - 0.3055077)
      V(I) = (H(M)/BETA)*(1.6110154 + 0.3889845*COS(PI*(4*GAMMA - 2)))
      GOTO 2201
2250  S(I) = S(C) + H(M)*(4.6660917*GAMMA - 2.44406184*(GAMMA**2)
     #       - 1.2292648)
      V(I) = (H(M)/BETA)*(4.6660917 - 4.888124*GAMMA)
2201  CONTINUE

4000  CONTINUE

      RETURN
      END
```

```
**********************************************************************************
**********************************************************************************
**                                                                              **
**                          SUBROUTINE CURVES                                   **
**                                                                              **
**********************************************************************************
**********************************************************************************
**                                                                              **
**    THIS SUBROUTINE COMPUTES THE ACCELERATION OF AND JERK ON                  **
**    THE FOLLOWER AT ONE DEGREE INTERVALS.  THE SUBROUTINE IS                  **
**    STRUCTURED SUCH THAT THE CALCULATIONS FOR EACH SEGMENT ARE                **
**    DONE INSIDE A DO LOOP.  THIS SUBROUTINE IS NOT REQUIRED IN                **
**    ALL THE DISPLAY OPTIONS CHOSEN BY THE USER.  HOWEVER IT MUST              **
**    BE CALLED IN CASE THE USER DECIDES TO DOCUMENT THE DATA.                  **
**                                                                              **
**********************************************************************************
**********************************************************************************
```

```
      SUBROUTINE CURVES

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
```

```
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

      DO 4000 M= 1,N

      BETA = (UPTO(M) - UPTO(M-1))*PI/180.
      C = UPTO(M-1)
      THETA = 0.

      GO TO (100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,
     #       1400,1500,1600,1700,1800,1900,2000,2100,2200),E(M)

***************** If motion is DWELL then go to this section.***************

      DO 11 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) = 0.
         J(I) = 0.
   11 CONTINUE
      GOTO 4000

***************** If motion is  H-1  then go to this section.***************

  100 DO 101 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) =  H(M)*((PI*0.5/BETA)**2)*(COS(PI*GAMMA/2))
         J(I) = -H(M)*((PI*0.5/BETA)**3)*(SIN(PI*GAMMA/2))
  101 CONTINUE
      GOTO 4000

***************** If motion is  H-2  then go to this section.***************

  200 DO  201 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) = -((PI/(2*BETA))**2)*H(M)*(SIN(PI*GAMMA/2))
         J(I) = -H(M)*((PI*0.5/BETA)**3)*(COS(PI*GAMMA/2))
  201 CONTINUE
      GOTO 4000
```

```
**************** If motion is  H-3  then go to this section.***************

  300 DO 301 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) = ((PI/(2*BETA))**2)*H(M)*(COS(PI*GAMMA/2))
         J(I) = -H(M)*((PI*0.5/BETA)**3)*(SIN(PI*GAMMA/2))
  301 CONTINUE
      GOTO 4000

**************** If motion is  H-4  then go to this section.***************

  400 DO 401 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) = -((PI/(2*BETA))**2)*H(M)*(SIN(PI*GAMMA/2))
         J(I) = - H(M)*((PI*0.5/BETA)**3)*(COS(PI*GAMMA/2))
  401 CONTINUE
      GOTO 4000

**************** If motion is  H-5  then go to this section.***************

  500 DO 501 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) =   2*((PI/(2*BETA))**2)*H(M)*(COS(PI*GAMMA))
         J(I) =  -4*((PI/(2*BETA))**3)*H(M)*(SIN(PI*GAMMA))
  501 CONTINUE
      GOTO 4000

**************** If motion is  H-6  then go to this section.***************

  600 DO 601 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) =  2*((PI/(2*BETA))**2)*H(M)*(COS(PI*GAMMA))
         J(I) = -4*((PI/(2*BETA))**3)*H(M)*(SIN(PI*GAMMA))
  601 CONTINUE
      GOTO 4000

**************** If motion is  C-1  then go to this section.***************

  700 DO 701 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) = (PI/BETA)*(H(M)/BETA)*(SIN(PI*GAMMA))
         J(I) = ((PI/BETA)**2)*(H(M)/BETA)*(COS(PI*GAMMA))
  701 CONTINUE
      GOTO 4000

**************** If motion is  C-2  then go to this section.***************
```

```
  800 DO 801 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) = -(PI/BETA)*(H(M)/BETA)*(SIN(PI*GAMMA))
         J(I) = -((PI/BETA)**2)*(H(M)/BETA)*(COS(PI*GAMMA))
  801 CONTINUE
      GOTO 4000

***************** If motion is  C-3  then go to this section.***************

  900 DO 901 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) = (PI/BETA)*(H(M)/BETA)*(SIN(PI*GAMMA))
         J(I) = ((PI/BETA)**2)*(H(M)/BETA)*(COS(PI*GAMMA))
  901 CONTINUE
      GOTO 4000

***************** If motion is  C-4  then go to this section.***************

 1000 DO 1001 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) = -(PI/BETA)*(H(M)/BETA)*(SIN(PI*GAMMA))
         J(I) = -((PI/BETA)**2)*(H(M)/BETA)*(COS(PI*GAMMA))
 1001 CONTINUE
      GOTO 4000

***************** If motion is  C-5  then go to this section.***************

 1100 DO 1101 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) = 2*(PI/BETA)*(H(M)/BETA)*(SIN(2*PI*GAMMA))
         J(I) = ((2*(PI/BETA))**2)*(H(M)/BETA)*(COS(2*PI*GAMMA))
 1101 CONTINUE
      GOTO 4000

***************** If motion is  C-6  then go to this section.***************

 1200 DO 1201 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         A(I) =  2*(PI/BETA)*(H(M)/BETA)*(SIN(2*PI*GAMMA))
         J(I) = ((2*(PI/BETA))**2)*(H(M)/BETA)*(COS(2*PI*GAMMA))
 1201 CONTINUE
      GOTO 4000

******** If motion is  Constant Velocity then go to this section.**********

 1300 DO 1301 I = FROM(M),UPTO(M)
```

```
         A(I) = 0
         J(I) = 0.
 1301 CONTINUE
      GOTO 4000

******* If motion is Constant Acceleration then go to this section.********

 1400 DO 1401 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA
         IF (GAMMA .LE. 0.5) THEN
            A(I) = 4*H(M)/(BETA**2)
         ELSE
            A(I) = -4*H(M)/(BETA**2)
         ENDIF
         J(I) = 0.
 1401 CONTINUE
      GOTO 4000

********** If motion is 2-3 Polynomial then go to this section.***********

 1500 DO 1501 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         A(I) = 6*(1 - 2*THETA)
         J(I) = -12.
 1501 CONTINUE
      GOTO 4000

********* If motion is 3-4-5 Polynomial then go to this section. **********

 1600 DO 1601 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         A(I) = 60*(THETA) - 180*(THETA**2) + 120*(THETA**3)
         J(I) = 60. - 360*(THETA) + 360*(THETA**2)
 1601 CONTINUE
      GOTO 4000

******** If motion is 4-5-6-7 Polynomial then go to this section. ********

 1700 DO 1701 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         A(I) = 420*(THETA**2) - 1680*(THETA**3) + 2100*(THETA**4)
     #           - 840*(THETA**5)
         J(I) = 840*(THETA) - 5040*(THETA**2) + 8400*(THETA**3)
     #           - 4200*(THETA**4)
 1701 CONTINUE
      GOTO 4000

******** If motion is Gutman 1-3 Harmonic then go to this section. ********

 1800 DO 1801 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
```

```
          GAMMA = THETA/BETA
          A(I)= (PI/8)*H(M)/(BETA**2)*(15*SIN(2*PI*GAMMA)
     #          +3*SIN(6*PI*GAMMA))
          J(I) = ((PI**2)/4)*(H(M)/(BETA**3))*( 15*COS(2*PI*GAMMA)
     #           + 9*COS(6*PI*GAMMA))
 1801 CONTINUE
      GOTO 4000

**** If motion is Freudenstein 1-3 Harmonic then go to this section. ****

 1900 DO 1901 I = FROM(M),UPTO(M)
          THETA = THETA + (PI/180.)
          GAMMA = THETA/BETA
          A(I) = (2*PI/BETA)*(H(M)/BETA)*(27*SIN(2*PI*GAMMA)/28
     #           + 3*SIN(6*PI*GAMMA)/28)
          J(I) = ((2*PI/BETA)**2)*(H(M)/BETA)*(27*COS(2*PI*GAMMA)/28
     #           + 9*COS(6*PI*GAMMA)/28)
 1901 CONTINUE
      GOTO 4000

** If motion is Freudenstein 1-3-5 Harmonic then go to this section. **

 2000 B = 1125./1192.
      DO 2001 I = FROM(M),UPTO(M)
          THETA = THETA + (PI/180.)
          GAMMA = THETA/BETA
          A(I) = (2*PI/BETA)*(H(M)/BETA)*B*(SIN(2*PI*GAMMA)
     #           + SIN(6*PI*GAMMA)/6 + SIN(10*PI*GAMMA)/50)
          J(I) = ((2*PI/BETA)**2)*(H(M)/BETA)*B*(COS(2*PI*GAMMA)
     #           + COS(6*PI*GAMMA)/2 + COS(10*PI*GAMMA)/10)
 2001 CONTINUE
      GOTO 4000

********** If motion is Modified Sine then go to this section. ************

 2100 DO 2101 I = FROM(M),UPTO(M)
          THETA = THETA + (PI/180.)
          GAMMA = THETA/BETA

          IF (GAMMA .LT. 1/8) THEN
             A(I) = 5.52794*(H(M)/(BETA**2))*SIN(4*PI*GAMMA)
             J(I) = 69.4659*(H(M)/(BETA**3))*COS(4*PI*GAMMA)
          ELSE
             IF (GAMMA .LT. 7/8) THEN
                A(I) = 5.52794*(H(M)/(BETA**2))*COS((4*PI/3)*GAMMA-PI/6)
                J(I) = -23.1553*(H(M)/(BETA**3))*SIN((4*PI/3)*GAMMA-PI/6)
             ELSE
                A(I) = 5.52794*(H(M)/(BETA**2))*SIN(2*PI*(2*GAMMA - 1))
                J(I) = 69.4659*(H(M)/(BETA**3))*COS(2*PI*(2*GAMMA - 1))
             ENDIF
          ENDIF
```

```
 2101 CONTINUE
      GOTO 4000

******* If motion is Modified Trapezoidal then go to this section. ********

 2200 DO 2201 I = FROM(M),UPTO(M)
         THETA = THETA + (PI/180.)
         GAMMA = THETA/BETA

         IF (GAMMA .LT. 1/8) GOTO 2210
         IF (GAMMA .LT. 3/8) GOTO 2220
         IF (GAMMA .LT. 1/2) GOTO 2230
         IF (GAMMA .LT. 5/8) GOTO 2240
         IF (GAMMA .LT. 7/8) GOTO 2250

         A(I) = -4.888124*(H(M)/(BETA**2))*SIN(PI*(4*GAMMA-3))
         J(I) = -61.425769*(H(M)/(BETA**3))*COS(PI*(4*GAMMA-3))
         GOTO 2201

 2210    A(I) = 4.888124*(H(M)/(BETA**2))*SIN(4*PI*GAMMA)
         J(I) = 61.425769*(H(M)/(BETA**3))*COS(4*PI*GAMMA)
         GOTO 2201

 2220    A(I) = (H(M)/(BETA**2))*(4.888124)
         J(I) = 0.
         GOTO 2201

 2230    A(I) = 4.888124*(H(M)/(BETA**2))*SIN(PI*(4*GAMMA - 1))
         J(I) = 61.425769*(H(M)/(BETA**3))*COS(PI*(4*GAMMA - 1))
         GOTO 2201

 2240    A(I) = -4.888124*(H(M)/(BETA**2))*SIN(PI*(4*GAMMA - 2))
         J(I) = -61.425769*(H(M)/(BETA**3))*COS(PI*(4*GAMMA - 2))
         GOTO 2201

 2250    A(I) = -4.888124*(H(M)/(BETA**2))
         J(I) = 0.

 2201 CONTINUE


 4000 CONTINUE

      RETURN
      END
```

```
**********************************************************************
**********************************************************************
**                                                                  **
**                      SUBROUTINE CHANGE                           **
```

```
**                                                                      **
**********************************************************************
**********************************************************************
**                                                                      **
**    THIS SUBROUTINE ALLOWS THE USER TO CHANGE ANY PARAMETER USED      **
**    IN THE DESIGN OF THE CAM.  THE USER DOES NOT HAVE TO PASS         **
**    THROUGH THE WHOLE PROCEDURE AGAIN.  BY SELECTING THE PROPER       **
**    OPTION THE USER MAY CALL ANY ONE OF THE ABOVE SUBROUTINES AND     **
**    INCORPORATE CHANGES IN THE DESIGN OF THE CAM.                     **
**                                                                      **
**********************************************************************
**********************************************************************


      SUBROUTINE CHANGE

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

      CHARACTER CHAR
   10 FORMAT(A1)
  100 WRITE(6,*) '                    Please select option'
      WRITE(6,*) ' '
      WRITE(6,*) ' '
      WRITE(6,*) '                    1: Follower Type'
      WRITE(6,*) '                    2: Basic Data'
      WRITE(6,*) '                    3: Output Type'
      WRITE(6,*) '                    4: Design Req.'
      WRITE(6,*) '                    5: Motion Data'
      WRITE(6,*) '                    6: NO MORE CHANGES'
      READ(5,*) R

      IF (R .LT. 1 .OR. R .GT. 6) THEN
         WRITE(6,*) '                       ERROR!! INVALID SELECTION!'
         GOTO 100
      ENDIF

      GO TO (200,300,400,500,600,700),R
```

```
200 CALL FOLLOW
    GOTO 100

300 CALL BASIC
    GOTO 100

400 CALL OUTPUT
    GOTO 100

500 CALL DESIGN
    GOTO 100

600 CALL MOTION
    GOTO 100

700 RETURN
    END
```

```
C*************************************************************************
C*************************************************************************
C**                                                                     **
C**                        SUBROUTINE PROFIL                            **
C**                                                                     **
C*************************************************************************
C*************************************************************************
C**                                                                     **
C**    THIS SUBROUTINE COMPUTES THE X AND Y COORDINATES OF THE CAM      **
C**    PROFILE.  IT USES THE DATA MADE AVAILABLE BY THE USER FOR        **
C**    THE CAM AND THE FOLLOWER.  THE DATA FOR MOTION CHARACTERISTICS   **
C**    MAY ALSO BE USED IN SOME CASES. THIS SUBROUTINE IS NOT REQUIRED  **
C**    FOR ALL DISPLAY OPTIONS CHOSEN BY THE USER.  HOWEVER IT MUST     **
C**    BE CALLED IN CASE THE USER DECIDES TO DOCUMENT THE DATA.         **
C**                                                                     **
C*************************************************************************
C*************************************************************************

      SUBROUTINE PROFIL

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R
```

```
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

      GO TO (100,200,300,400,500,600,700),TYPE


100   DO 101 I = 1, 360
         THETA = I*PI/180.
         X(I) = (BASE + S(I))*COS(THETA)
         Y(I) = (BASE + S(I))*SIN(THETA)
101   CONTINUE
      GO TO 1000

200   DO 201 I = 1, 360
         THETA = I*PI/180.
         ALPHA(I) = ATAN(V(I)/(BASE+ROLL+S(I)))
         X(I) = (BASE+ROLL+S(I))*COS(THETA) - ROLL*COS(THETA-ALPHA(I))
         Y(I) = (BASE+ROLL+S(I))*SIN(THETA) - ROLL*SIN(THETA-ALPHA(I))
201   CONTINUE
      GO TO 1000

300   DO 301 I = 1, 360
         THETA = I*PI/180.
         ALPHA(I) = ATAN((V(I)-ECC)/(BASE+ROLL+S(I)))
         X(I) = ((((BASE+ROLL)**2-(ECC)**2)**0.5)+S(I))*COS(THETA)
     #           - ROLL*COS(THETA-ALPHA(I)) + ECC*SIN(THETA)
         Y(I) = ((((BASE+ROLL)**2-(ECC)**2)**0.5)+S(I))*SIN(THETA)
     #           - ROLL*SIN(THETA-ALPHA(I)) - ECC*COS(THETA)
301   CONTINUE
      GO TO 1000


400   DO 401 I = 1, 360
        THETA = I*PI/180.
        PHIZ  = ACOS(((RF**2)+(RA**2)-((BASE+ROLL)**2))/(2*RA*RF))
        ALPHA(I)=ATAN(1/TAN(S(I)+PHIZ)-(RF*(1-V(I))/(RA*SIN(S(I)+PHIZ))))
        X(I) = RA*COS(THETA) - RF*COS(THETA - S(I) - PHIZ)
     #         - ROLL*SIN(ALPHA(I) - THETA + S(I) + PHIZ)
        Y(I) = RA*SIN(THETA) - RF*SIN(THETA - S(I) - PHIZ)
     #         - ROLL*COS(ALPHA(I) - THETA + S(I) + PHIZ)
401   CONTINUE
      GO TO 1000


500   DO 501 I = 1, 360
         THETA = I*PI/180.
         X(I) = (BASE+S(I))*COS(THETA) - V(I)*SIN(THETA)
         Y(I) = (BASE+S(I))*SIN(THETA) + V(I)*COS(THETA)
```

```
  501 CONTINUE
      GO TO 1000


  600 DO 601 I = 1, 360
         THETA = I*PI/180.
         PHIZ = ASIN(BASE/RA)
         DUMB = PHIZ + S(I) - THETA
         DUMA = (COS(THETA + DUMB))/(V(I) - 1)
         X(I) = RA*(COS(THETA) + DUMA*COS(DUMB))
         Y(I) = RA*(SIN(THETA) - DUMA*SIN(DUMB))
  601 CONTINUE
      GO TO 1000


  700 DO 701 I = 1, 360
         THETA = I*PI/180.
         PHIZ = ASIN((ECC+BASE)/RA)
         DUMB = PHIZ + S(I) - THETA
         DUMA = (COS(THETA + DUMB))/(V(I) - 1)
         X(I) = RA*(COS(THETA) + DUMA*COS(DUMB)) + ECC*SIN(DUMB)
         Y(I) = RA*(SIN(THETA) - DUMA*SIN(DUMB)) + ECC*COS(DUMB)
  701 CONTINUE

 1000 RETURN
      END
```

```
**********************************************************************************
**********************************************************************************
**                                                                              **
**                          SUBROUTINE GKS                                      **
**                                                                              **
**********************************************************************************
**********************************************************************************
**                                                                              **
**    THIS SUBROUTINE DISPLAYS THE OUTPUT PARAMETERS OF THE CAM TO              **
**    THE SCREEN IN FORM OF 2-D GRAPHICS.  IT ALLOWS THE USER TO                **
**    SIMULATE  THE MOTION OF THE CAM.  THE SIMULATION SMOOTHNESS               **
**    DEPENDS ON THE DEGREE STEPS SELECTED BY THE USER.                         **
**    THIS ROUTINE IS CALLED ONLY IF THE DISPLAY SYSTEM CHOSEN BY               **
**    THE USER AT THE BEGINING OF THE PROGRAM IS GKS.                           **
**                                                                              **
**********************************************************************************
**********************************************************************************
```

```
      SUBROUTINE GKS

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
```

```
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R, W
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

      REAL XFA(4), YFA(4)
      REAL NORMS(360), NORMV(360), NORMA(360), NORMJ(360), NORMSG(360)
      REAL XS(361), YS(361), XZ(361)
      REAL XA(361), YA(361), ZA(361)
      REAL XV(361), YV(361), ZV(361)
      REAL XJ(361), YJ(361), ZJ(361)
      REAL XAXIS(5), YAXIS(5)

      CHARACTER*1 CHAR
      CHARACTER*8 TYP
      CHARACTER*8 DIS1
      CHARACTER*8 VEL1
      CHARACTER*8 ACC1
      CHARACTER*8 JER1
      CHARACTER*8 DIS2
      CHARACTER*8 VEL2
      CHARACTER*8 ACC2
      CHARACTER*8 JER2

   10 FORMAT(A1)
      L = L + 1
      IF (L .LT. 2) THEN
      CALL UOPNSF(1,'ERROR DATA',1,JERROR)
         IF (JERROR .NE. 0) THEN
            WRITE(*,*)'CANNOT OPEN ERROR LOGGING FILE ',JERROR
            STOP
         END IF
      END IF

      KERFIL = 1

********** Open file for reading and writing **********

      OPEN (UNIT = 20, FILE = 'ERRFIL')

********** Open GKS **********
```

```
      CALL GOPKS (20)

*********This section asks the user to input the workstation type.*********

      IF (L .LT. 2) THEN
100   WRITE (5, *) 'PLEASE ENTER TERMINAL TYPE'
      READ (5, FMT=200) KWKTYP
200   FORMAT (I6)

**************** This section validates the workstation type.***************

      CALL GQEWK (0,KERROR,KNUM,KJUNK)
      TYP = 'INVALID'

      DO 300 I=1,KNUM
         W=I
         CALL GQEWK (W,KERROR,KJUNK,KTYPE)
         IF (KWKTYP .EQ. KTYPE) TYP = 'VALID'
300   CONTINUE

      IF (TYP .EQ. 'INVALID') GOTO 100
      END IF

***** Open and activate workstation 1, use LUN 5, and KWKTYP for type. ***

      CALL GOPWK (1,5,KWKTYP)
      CALL GACWK (1)

********************  Set values for dummy variables  ********************

      DUMMY1 = .75/4
      DUMMY2 = .75/2
      DUMMY3 = 3*.75/4

********************  Calculate maximum profile value  *********************
********************* and set dummy as double the value *********************

      SCALE1 = 0.001
      DO 350 I=1,360
      SCALE1 = MAX( ABS(X(I)), SCALE1)
  350 CONTINUE
      DO 375 I=1,360
      SCALE1 = MAX( ABS(Y(I)), SCALE1)
  375 CONTINUE
      SCALE1 = 2*SCALE1

********************  Calculate maximum displacement   *********************
********************* and normalize to get dummy value  *********************

      SCALE2 = 0.001
      DO 400 I=1,360
      SCALE2 = MAX( ABS(S(I)), SCALE2)
```

```
  400 CONTINUE
      DO 450 I=1,360
      NORMS(I)= S(I)/SCALE2
  450 CONTINUE

******************** Calculate maximum velocity.       **********************
******************** and normalize to get dummy value  **********************

      SCALE3 = 0.001
      DO 500 I=1,360
      SCALE3 = MAX( ABS(V(I)), SCALE3)
  500 CONTINUE
      DO 550 I=1,360
      NORMV(I) = V(I)/SCALE3
  550 CONTINUE

******************** Calculate maximum acceleration    **********************
******************** and normalize to get dummy value  **********************

      SCALE4 = 0.001
      DO 600 I=1,360
      SCALE4 = MAX( ABS(A(I)), SCALE4)
  600 CONTINUE
      DO 650 I=1,360
      NORMA(I) = A(I)/SCALE4
  650 CONTINUE

**********************       Calculate maximum jerk.       **********************
******************** and normalize to get dummy value  **********************

      SCALE5 = 0.001
      DO 700 I=1,360
      SCALE5 = MAX( ABS(J(I)), SCALE5)
  700 CONTINUE
      DO 650 I=1,360
      NORMJ(I) = J(I)/SCALE5
  750 CONTINUE

*********** Calculate nearest integer value for each of the motion ********
*********** characteristics and read them as text, to be used later********

      SCAL2 = SCALE2 + 0.5
      SCAL3 = SCALE3 + 0.5
      SCAL4 = SCALE4 + 0.5
      SCAL5 = SCALE5 + 0.5

      ISCAL2 = INT(SCAL2)
      ISCAL3 = INT(SCAL3)
      ISCAL4 = INT(SCAL4)
      ISCAL5 = INT(SCAL5)

      WRITE(DIS1,'(I6)') ISCAL2
```

```
      WRITE(VEL1,'(I6)') ISCAL3
      WRITE(ACC1,'(I6)') ISCAL4
      WRITE(JER1,'(I6)') ISCAL5

      WRITE(DIS2,'(I6)') -ISCAL2
      WRITE(VEL2,'(I6)') -ISCAL3
      WRITE(ACC2,'(I6)') -ISCAL4
      WRITE(JER2,'(I6)') -ISCAL5

********** Set windows and normalized transformations ******************

      CALL GSWN(1,0.,1.,0.,1.)

      CALL GSWN (2,-SCALE1,SCALE1,-SCALE1,SCALE1)
      CALL GSVP (2,0.,.75,0.,.75)

      CALL GSWN (3,0.,1.,0.,1.)
      CALL GSVP (3,.75,1.,0.,DUMMY1)

      CALL GSWN (4,0.,1.,0.,1.)
      CALL GSVP (4,.75,1.,DUMMY1,DUMMY2)

      CALL GSWN (5,0.,1.,0.,1.)
      CALL GSVP (5,.75,1.,DUMMY2,DUMMY3)

      CALL GSWN (6,0.,1.,0.,1.)
      CALL GSVP (6,.75,1.,DUMMY3,.75)

      CALL GSWN (7,0.,1.,0.,1.)
      CALL GSVP (7,0.,.5,.375,.75)

      CALL GSWN (8,0.,1.,0.,1.)
      CALL GSVP (8,.5,1.,.375,.75)

      CALL GSWN (9,0.,1.,0.,1.)
      CALL GSVP (9,.5,1.,0.,.375)

      CALL GSWN (10,0.,1.,0.,1.)
      CALL GSVP (10,0.,.5,0.,.375)


      CALL GSELNT (1)

*************** Inquire the maximum surface size. ***************************


      CALL GQMDS (KWKTYP, KERROR,XUNITS,XMETER,YMETER,XRAST,YRAST)
      IF (KERROR .NE. 0) WRITE (5, *) 'INQUIRE SIZE ERROR'

************** Define workstation transformation and set max.*************
**************          values to the correct units.          **************
```

```
      IF (XUNITS .EQ. 0) THEN
           XMAX=XMETER
           YMAX=YMETER

      ELSE

           XMAX=XRAST
           YMAX=YRAST

      ENDIF

      CALL GSWKWN (1,0.,1.,0.,.75)
      CALL GSWKVP (1,0.,XMAX,0.,YMAX)
      CALL GUWK (1,1)

********************** Set FILL AREA coordinates. ********************************

      CALL GSELNT (2)

      XFA(1) = -SCALE1
      XFA(2) =  SCALE1
      XFA(3) =  SCALE1
      XFA(4) = -SCALE1
      YFA(1) = -SCALE1
      YFA(2) = -SCALE1
      YFA(3) =  SCALE1
      YFA(4) =  SCALE1

      CALL GCRSG(1500)
      CALL GSFAIS(1)
      CALL GSFACI(4)
         IF (OUT .NE. 2) THEN
            CALL GFA(4,XFA,YFA)
         ENDIF
      CALL GCLSG

***************** Set text positions and set attributes. ******************

      TPOSX  = .125
      TPOSY  = .9
      TPOSY1 = .1

      CALL GSCHXP (1.25)
      CALL GSCHH (.03)
      CALL GSTXFP(1,2)
      CALL GSTXAL(3,3)
      CALL GSTXCI (0)

***************** Set Polyline and Polymarker attributes ******************
****************     and draw Polyline and Polymarker    ******************
```

```
      CALL GSPLCI(1)
      CALL GSLN(1)
      CALL GSMK(2)
      CALL GSPMCI(1)
      CALL GCRSG(2000)
         IF (OUT .NE. 2) THEN
            CALL GPL(360,X,Y)
            CALL GPM(1,0.,0.)
      ENDIF
      CALL GCLSG

********************* Select the right transformation **********************
********************* and give right value to FILL AREA *********************

      IF (OUT .NE. 1) THEN

      IF (OUT .EQ. 3) THEN
         CALL GSELNT (3)
      ELSEIF (OUT .EQ. 2) THEN
         CALL GSELNT (7)
      ENDIF


      XFA(1) = 0.
      XFA(2) = 1.
      XFA(3) = 1.
      XFA(4) = 0.
      YFA(1) = 0.
      YFA(2) = 0.
      YFA(3) = 1.
      YFA(4) = 1.

***************** Set axes positions and set attributes. *********************
***************** Draw the curves and write the text in  *********************
*****************              the right positions.      *********************


      XAXIS(1) = 0.1
      XAXIS(2) = 0.1
      XAXIS(3) = 0.9
      XAXIS(4) = 0.1
      XAXIS(5) = 0.1
      YAXIS(1) = 0.1
      YAXIS(2) = 0.5
      YAXIS(3) = 0.5
      YAXIS(4) = 0.5
      YAXIS(5) = 0.9

      CALL GCRSG(3000)
      CALL GSFAIS(1)
      CALL GSFACI(2)
      CALL GFA(4,XFA,YFA)
```

```
      XS(1) = 0.1
      YS(1) = 0.5

      DO 800 IS=1,360
      XS(IS+1) = 0.1 + 0.8*IS/360.
      YS(IS+1) = (0.5 + 0.4*NORMS(IS))
800   CONTINUE

      CALL GSPLCI(1)
      CALL GPL(5,XAXIS,YAXIS)
      CALL GSPLCI(0)
      CALL GPL(361,XS,YS)
      CALL GTX (TPOSX,TPOSY,DIS1)
      CALL GTX (TPOSX,TPOSY1,DIS2)
      CALL GTX (.08,.46,'0')
      CALL GTX (.35,.46,'90')
      CALL GTX (.55,.46,'180')
      CALL GTX (.75,.46,'270')
      CALL GTX (.95,.46,'360')

      CALL GSTXAL(2,3)
      CALL GTX (.5,.06,'DISPLACEMENT')
      CALL GSTXAL(3,3)

      CALL GCLSG

      IF (OUT .EQ. 3) THEN
         CALL GSELNT (4)
      ELSEIF (OUT .EQ. 2) THEN
         CALL GSELNT (8)
      ENDIF

      CALL GCRSG(4000)
      CALL GSFAIS(1)
      CALL GSFACI(6)
      CALL GFA(4,XFA,YFA)

      XV(1) = 0.1
      YV(1) = 0.5

      DO 900 IV=1,360
      XV(IV+1) = 0.1 + 0.8*IV/360.
      YV(IV+1) = (0.5 + 0.4*NORMV(IV))
900   CONTINUE

      CALL GSPLCI(1)
      CALL GPL(5,XAXIS,YAXIS)
      CALL GSPLCI(0)
      CALL GPL(361,XV,YV)
      CALL GTX (TPOSX,TPOSY,VEL1)
      CALL GTX (TPOSX,TPOSY1,VEL2)
```

```
      CALL GTX (.08,.46,'0')
      CALL GTX (.35,.46,'90')
      CALL GTX (.55,.46,'180')
      CALL GTX (.75,.46,'270')
      CALL GTX (.95,.46,'360')

      CALL GSTXAL(2,3)
      CALL GTX (.5,.06,'VELOCITY')
      CALL GSTXAL(3,3)

      CALL GCLSG

      IF (OUT .EQ. 3) THEN
         CALL GSELNT (5)
      ELSEIF (OUT .EQ. 2) THEN
         CALL GSELNT (9)
      ENDIF

      CALL GCRSG(5000)
      CALL GSFAIS(1)
      CALL GSFACI(2)
      CALL GFA(4,XFA,YFA)

      XA(1) = 0.1
      YA(1) = 0.5

      DO 1000 IA=1,360
      XA(IA+1) = 0.1 + 0.8*IA/360.
      YA(IA+1) = (0.5 + 0.4*NORMA(IA))
 1000 CONTINUE

      CALL GSPLCI(1)
      CALL GPL(5,XAXIS,YAXIS)
      CALL GSPLCI(0)
      CALL GPL(361,XA,YA)
      CALL GTX (TPOSX,TPOSY,ACC1)
      CALL GTX (TPOSX,TPOSY1,ACC2)
      CALL GTX (.08,.46,'0')
      CALL GTX (.35,.46,'90')
      CALL GTX (.55,.46,'180')
      CALL GTX (.75,.46,'270')
      CALL GTX (.95,.46,'360')

      CALL GSTXAL(2,3)
      CALL GTX (.5,.06,'ACCELERATION')
      CALL GSTXAL(3,3)

      CALL GCLSG

      IF (OUT .EQ. 3) THEN
         CALL GSELNT (6)
      ELSEIF (OUT .EQ. 2) THEN
```

```
        CALL GSELNT (10)
      ENDIF

      CALL GCRSG(6000)
      CALL GSFAIS(1)
      CALL GSFACI(6)
      CALL GFA(4,XFA,YFA)

      XJ(1) = 0.1
      YJ(1) = 0.5

      DO 1100 IJ=1,360
      XJ(IJ+1) = 0.1 + 0.8*IJ/360.
      YJ(IJ+1) = (0.5 + 0.4*NORMJ(IJ))
1100  CONTINUE

      CALL GSPLCI(1)
      CALL GPL(5,XAXIS,YAXIS)
      CALL GSPLCI(0)
      CALL GPL(361,XJ,YJ)
      CALL GTX (TPOSX,TPOSY,JER1)
      CALL GTX (TPOSX,TPOSY1,JER2)
      CALL GTX (.08,.46,'0')
      CALL GTX (.35,.46,'90')
      CALL GTX (.55,.46,'180')
      CALL GTX (.75,.46,'270')
      CALL GTX (.95,.46,'360')

      CALL GSTXAL(2,3)
      CALL GTX (.5,.06,'JERK')
      CALL GSTXAL(3,3)

      CALL GCLSG

      ENDIF

      WRITE(6,*) '      Do you want to animate this cam? (Y/N)'
      READ(5,10)CHAR

      IF (CHAR .EQ. 'Y') THEN
1200  WRITE(6,*) '      Enter the degree step you want'
      READ(5,*)ITE

      WRITE(6,*) '      Transformations being performed.'
      WRITE(6,*) '      Please wait.'
      CALL GSELNT (2)
      CALL GSPLCI(1)
      CALL GSMK(2)
      CALL GSPMCI(1)

      APHI = 0.
      NSTEP = 360/ITE
```

```
      DO 1300 IANI = 1,NSTEP
      APHI = APHI + ITE*PI/180.
      CALL GCRSG(IANI)
      CALL GSVIS(IANI,0)
      CALL GPL(360,X,Y)
      CALL GPM(1,0.,0.)
      CALL GEVTM(0.,0.,0.,0.,APHI,1.,1.,0,XFRM)
      CALL GSSGT(IANI,XFRM)
      CALL GCLSG
1300  CONTINUE

      WRITE(6,*) '       Transformations Completed.'
1400  WRITE(6,*) '       How many rotations do you want to see?'
      WRITE(6,*) '       MIN: 1    MAX: 30'
      READ(5,*)IRO

      IF (IRO .LT. 1 .OR. IRO .GT. 30) THEN
         WRITE(6,*)'   ERROR!! Input not valid. Reenter value'
         GOTO 1300
      ENDIF

1400  CALL GSVIS(2000,0)

      DO 1700 IAX = 1, IRO

        CALL GSVIS(2000,1)
        CALL GSVIS(2000,0)

        DO 1500 IANI = 1,NSTEP
           CALL GSVIS(IANI,1)
           CALL GSVIS(IANI,0)
1600    CONTINUE

1700  CONTINUE

      CALL GSVIS(1500,1)
      CALL GSVIS(2000,1)

      WRITE(6,*) '       Animation Complete.'
      WRITE(6,*) '       Do you want to see it again? (Y/N)'
      READ(5,10)CHAR

      IF (CHAR .EQ. 'Y') THEN
      WRITE(6,*) '       Do you want to change degree step.(Y/N)?'
      READ(5,10)CHAR

      IF (CHAR .EQ. 'Y') GOTO 1200
      WRITE(6,*) '    Do you want to change number of rotations? (Y/N)'
      READ(5,10)CHAR

      IF (CHAR .EQ. 'Y') GOTO 1400
      GOTO 1400
```

```
      ENDIF

      ENDIF


************************** Close up everything. ********************************

      CALL GDAWK (1)
      CALL GCLWK (1)
      CALL GCLKS
      CLOSE (20)

      RETURN
      END


*******************************************************************************
*******************************************************************************
**                                                                           **
**                          SUBROUTINE CADAM                                 **
**                                                                           **
*******************************************************************************
*******************************************************************************
**                                                                           **
**    THIS SUBROUTINE DISPLAYS THE OUTPUT PARAMETERS OF THE CAM TO           **
**    THE SCREEN IN FORM OF 3-D GRAPHICS.  IT LINKS THE PROGRAM TO           **
**    THE CADAM SYSTEM AND GENERATES A FILE (ANICAM    ,ARG) IN THE         **
**    EMPLOYEE ACCOUNT (MECH,CAM). THIS PROGRAM USES THE CADCD               **
**    GEOMETRY INTERFACE MODULE.                                             **
**    THIS ROUTINE IS CALLED ONLY IF THE DISPLAY SYSTEM CHOSEN BY            **
**    THE USER AT THE BEGINING OF THE PROGRAM IS CADAM.                      **
**                                                                           **
*******************************************************************************
*******************************************************************************


      SUBROUTINE CADAM

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R, W
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
```

```
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

      CHARACTER DRAWID*20, USERID*8, GROUP*4, IDVU*4, TEXT*12
      CHARACTER ID1*4, ID2*4, ID3*4

      REAL XVCTR(3), YVCTR(3), ZVCTR(3)
      REAL XVTTR(3), YVTTR(3), ZVTTR(3)
      REAL XYZ1(3), XYZ2(3), XYZ3(3), XYZ4(3)
      REAL D1(3,46), D(3,46), VCAD(3,46), SCAD(46)
      REAL T1(46), T2(46), T3(46)


      DATA DRAWID/'ANICAM           ARG'/
      DATA USERID/'CAM     '/, GROUP/'MECH'/
      DATA XVCTR/1.,0.,0./,YVCTR/0.,0.,1./, ZVCTR/0.,-1.,0./
      DATA XVTTR/0.866,-0.5,0.5/,YVTTR/0.5,0.866,0./
      DATA ZVTTR/0.5,-0.5,-0.866/
      DATA IDVU/'IDIS'/
      DATA ID1/'LIN1'/, ID2/'LIN2'/, ID3/'LIN3'/
      DATA NOUT/6/

      N = 46
      M = 3
      K = 0
      IOPT = 1

  100 WRITE(6,*)'                    Enter shaft radius'
      READ(5,*)SHAFT
      IF(SHAFT .LT. 0) THEN
      WRITE(6,*) '                   ERROR!! RADIUS MUST BE POSITIVE'
      GOTO 100
      ENDIF

***************************** Start a Drawing ***********************************

      CALL CADST(DRAWID,USERID,GROUP)

*****************************  Start a View  ************************************

      CALL BEGVU(IOPT,IDVU,XYZ,XVTTR,YVTTR,ZVTTR,*1000)

      EXTR = 0.
      Z(1) = 0.
      Z(2) = WIDTH

      DO 400 IZ = 1, 2

      DO 200 ICOOR = 1, 45
```

```
      ANGL =  8*ICOOR*PI/180.
      D1(1,ICOOR) = SHAFT*COS(ANGL)
      D1(3,ICOOR) = SHAFT*SIN(ANGL)
      D1(2,ICOOR) = Z(IZ)

  200 CONTINUE

      D1(1,46) = D1(1,1)
      D1(3,46) = D1(3,1)
      D1(2,46) = Z(IZ)

****************** Draw a spline representing the shaft **********************

      CALL SPLINE(N,M,K,D1,VCAD,SCAD,T1,T2,T3,*1001)

************************** Calculate the coordinates ****************************

      DO 300 ICOOR = 1, 45

      ICORD = 8*ICOOR
      D(1,ICOOR) = X(ICORD)
      D(3,ICOOR) = Y(ICORD)
      D(2,ICOOR) = Z(IZ)

  300 CONTINUE

      D(1,46) = X(8)
      D(3,46) = Y(8)
      D(2,46) = Z(IZ)

****************** Draw a spline representing the cam ************************

      CALL SPLINE(N,M,K,D,VCAD,SCAD,T1,T2,T3,*1001)

  400 CONTINUE

******** Draw lines between points on the corresponding splines. *********


      DO 500 I = 1,45

      XYZ1(1) = D(1,I)
      XYZ1(3) = D(3,I)
      XYZ1(2) = 0.

      XYZ2(1) = D(1,I)
      XYZ2(3) = D(3,I)
      XYZ2(2) = WIDTH

      XYZ3(1) = D1(1,I)
      XYZ3(3) = D1(3,I)
      XYZ3(2) = 0.
```

```
      XYZ4(1) = D1(1,I)
      XYZ4(3) = D1(3,I)
      XYZ4(2) = WIDTH

      CALL LINE3D (XYZ1,XYZ2,*1004)
      CALL LINE3D (XYZ3,XYZ4,*1004)
      CALL LINE3D (XYZ1,XYZ3,*1004)
      CALL LINE3D (XYZ2,XYZ4,*1004)

  500 CONTINUE

********************************** File the drawing ******************************

      IOPT=2
      NOGOOD=0

      CALL CADFIL(IOPT,NOGOOD,IDUMMY)

      IF(NOGOOD.NE.1)THEN
         WRITE(NOUT,*)'ERROR IN CADFIL'
      ENDIF

      GOTO 2000


1000  WRITE(NOUT,*)'ERROR EXISTS IN BEGVU'
1001  WRITE(NOUT,*)'ERROR EXISTS IN SPLINE'
1002  WRITE(NOUT,*)'ERROR EXISTS IN NAME'
1003  WRITE(NOUT,*)'ERROR EXISTS IN SRFRUL'
1004  WRITE(NOUT,*)'ERROR EXISTS IN LINE3D'

2000  RETURN
      END
```

```
**********************************************************************************
**********************************************************************************
**                                                                              **
**                          SUBROUTINE MOVIE                                    **
**                                                                              **
**********************************************************************************
**********************************************************************************
**                                                                              **
**    THIS SUBROUTINE DISPLAYS THE OUTPUT PARAMETERS OF THE CAM TO              **
**    THE SCREEN IN FORM OF 3-D GRAPHICS.  IT GENERATES THE GEOMETRY            **
**    FILE (FILE.CAM) FOR DISPLAY, USING MOVIE.BYU.  IF REQUIRED, IT            **
**    CREATES A COMMAND FILE (FILE.COM) TO BE USED IN CONJUNCTION               **
**    WITH THE ANIMATE SOFTWARE DEVELOPED BY A. MYKLEBUST.                      **
**    THIS ROUTINE IS CALLED ONLY IF THE DISPLAY SYSTEM CHOSEN BY               **
**    THE USER AT THE BEGINING OF THE PROGRAM IS MOVIE.                         **
```

```
**                                                                              **
**********************************************************************************
**********************************************************************************

      SUBROUTINE MOVIE

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R, W
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

      REAL X1(3,150), XS(361), YS(361)
      INTEGER IP(600), NPL(2,2)
      CHARACTER*8 FILE
      CHARACTER*1 CHAR

      IJ = 1
      NP = 1
      NJ = 148
      NPT= 144
      NCON = 576
      NPL(1,1) = 1
      NPL(2,1) = 144

   25 WRITE(6,*)'                          Enter shaft radius'
      READ(5,*)SHAFT
      IF(SHAFT .LT. 0) THEN
      WRITE(6,*) '                         ERROR!! RADIUS MUST BE POSITIVE'
      GOTO 25
      ENDIF

********** Calculate points on shaft circumference **********

      DO 50 I = 1, 360

      ANGL =  I*PI/180.
      XS(I) = SHAFT*COS(ANGL)
      YS(I) = SHAFT*SIN(ANGL)
```

```
 50 CONTINUE

      XS(361) = XS(1)
      YS(361) = YS(1)

      X(361) = X(1)
      Y(361) = Y(1)

********** Calculate the points on the cam and shaft **********
**********    to be used for the representation     **********

      DO 100 I = 1, 361, 10
      X1(1,IJ) = X(I)
      X1(2,IJ) = Y(I)
      X1(3,IJ) = 0.
      IJ = IJ + 1
100 CONTINUE

      DO 200 I = 1, 361, 10
      X1(1,IJ) = XS(I)
      X1(2,IJ) = YS(I)
      X1(3,IJ) = 0.
      IJ = IJ + 1
200 CONTINUE

      DO 300 I = 1, 74
      X1(1,I+74) = X1(1,I)
      X1(2,I+74) = X1(2,I)
      X1(3,I+74) = WIDTH
300 CONTINUE

      DO 400 I = 1,36

      IP(4*I-3) = I
      IP(4*I-2) = I+1
      IP(4*I-1) = I+38
      IP(4*I)   = -(I+37)

400 CONTINUE

      DO 500 I = 37,72

      IP(4*I-3) = I-36
      IP(4*I-2) = I+38
      IP(4*I-1) = I+39
      IP(4*I)   = -(I-35)

500 CONTINUE

      DO 600 I = 73,108

      IP(4*I-3) = I-35
```

```
      IP(4*I-2) = I+39
      IP(4*I-1) = I+40
      IP(4*I)   = -(I-34)

  600 CONTINUE

      DO 700 I = 109,144

      IP(4*I-3) = I-34
      IP(4*I-2) = I-33
      IP(4*I-1) = I+4
      IP(4*I)   = -(I+3)

  700 CONTINUE

********************** Create the geometry file ******************************

      REWIND(8)

      WRITE(8,800) NP,NJ,NPT,NCON
      WRITE(8,800) ((NPL(I,IJ),I=1,2),IJ=1,NP)
      WRITE(8,900) ((X1(I,IJ),I=1,3),IJ=1,NJ)
      WRITE(8,800) (IP(I),I=1,NCON)
  800 FORMAT(16I5)
  900 FORMAT(6E12.5)

      WRITE(6,*)'Do you want to animate this cam?'
      READ(5,1000)CHAR

      IF (CHAR .EQ. 'N') GOTO 2000

********************** Write command file if required. ***********************

      REWIND(10)

 1000 FORMAT(A1)
      WRITE(6,*)'GIVE NAME OF DISK FILE'
      READ(5,1100)FILE
 1100 FORMAT(A8)
      WRITE(6,*)'GIVE STEP ANGLE'
      READ(5,*) STEP
      NSTEP = 360/STEP

      WRITE(9,*)'FILE.CAM'
      WRITE(9,1200)
 1200 FORMAT(/)
      WRITE(9,*)'CLSG'
      DZ = 360/NSTEP
      ZANG = 0.

      DO 1500 IJ = 1,NSTEP
      WRITE(9,*)'ROTA'
```

```
      WRITE(9,*)'Z'
      WRITE(9,1300)ZANG
 1300 FORMAT(F10.4)
      WRITE(9,*)'OSEG'
      WRITE(9,1400)IJ
 1400 FORMAT(I2)
      WRITE(9,*)'VIEW'
      WRITE(9,*)'CSEG'
      ZANG = DZ
 1500 CONTINUE
      WRITE(9,*)'SAVT'
      WRITE(9,1600)FILE
 1600 FORMAT(A8,/,'EXIT')

 2000 RETURN
      END
```

```
**********************************************************************
**********************************************************************
**                                                                  **
**                        SUBROUTINE DOCUM                          **
**                                                                  **
**********************************************************************
**********************************************************************
**                                                                  **
**    THIS SUBROUTINE SAVES THE LAST CAM DESIGNED BY THE USER. THE  **
**    DOCUMENTATION IS DONE UNDER THE FILE CAM.OUT.  THIS HELPS THE **
**    USER TO MAINTAIN RECORDS OF THE CAMS DESIGNED AND THEN SELECT **
**    THE BEST OF ALL OF THEM.                                      **
**    THIS ROUTINE IS CALLED ONLY IF THE USER WANTS THE DESIGN TO   **
**    BE SAVED.                                                     **
**                                                                  **
**********************************************************************
**********************************************************************
```

```
      SUBROUTINE DOCUM

      COMMON/CAM1/ TYPE, BASE, RPM, WIDTH, DESI, ROLL, ECC, RA, RF
      COMMON/COUNT/ I, N, M, B, PI, C, R, L
      COMMON/ANGLE/ BETA, THETA, GAMMA
      COMMON/MTN/ E(25), H(20), FROM(20), UPTO(20)
      COMMON/CRV/ S(360), V(360), A(360), J(360)
      COMMON/COOD/ ALPHA(360), X(361), Y(361), Z(361)
      COMMON/DISP/ OUT, MONI

      INTEGER I, N, M, C, R, W
      INTEGER TYPE, OUT, DESI, MONI
      INTEGER E, FROM, UPTO
```

```
      REAL BETA, THETA, GAMMA, B, PI, BASE, RPM, WIDTH
      REAL ROLL, ECC, RA, RF
      REAL S, V, A, J
      REAL ALPHA, X, Y, Z, H

**************** Write all parameters in a file ******************************

      REWIND(8)
      DO 100 K = 10,360,10
      WRITE(8,10) K,S(K),V(K),A(K),X(K),Y(K)
   10 FORMAT(1X,I3,5(2X,F7.3))
  100 CONTINUE

      RETURN
      END
```

The vita has been removed from the scanned document