VARIATIONAL SYSTEMS

IN

COMPUTER AIDED DESIGN

by

Dale T Gallaher

B.S.M.E, Bucknell University, 1979

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN MECHANICAL ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1984

Signature of Author ______________________
Department of Mechanical Engineering
February 13, 1984

Certified by ______________________
Philip Meyfarth
Thesis Supervisor

Accepted by ______________________
Warren Rosenhow
Chairman, Departmental Graduate Committee

VARIATIONAL SYSTEMS

IN

COMPUTER AIDED DESIGN

by

Dale T Gallaher

Submitted to the Department of Mechanical Engineering on February 13, 1984 in partial fulfillment of the requirements for the degree of Master of Science.

ABSTRACT

Variational systems are systems which can be assembled in a large variety of arrangements. The design and analysis of this type of system represents a difficult problem in engineering design. The system components are normally few in type but can be used in great quantity and in an almost infinite number of arrangements. A typical example is an electrical system which is assembled from resistors, transistors and the like. In a variational system, there is a specific set of equations which applies to each component and connection type; however, each arrangement has a different set of system equations. A Computer Aided Design (CAD) program used to design this type of system must have a means for determining the appropriate set of equations for a specific system arrangement.

CAD software was developed to aid in the design of gearing systems. This software provides a graphical means of specifying the connections between the components of the system (the system topology) and assembles a set of equations implied by the specified topology. After the implied equations are assembled, a designer can enter additional equations which further constrain the system. The arrangement definition is the combination of the topology, the equations which are assembled by the program, and the equations entered by the designer. The design of a specific arrangement is completed by specifying the system variables which are to be held constant while the program calculates values for the remaining variables. Any set of variables can be specified as constant given that the number of unknowns is equal to the number of equations, and that the set of constants does not conflict with the equations.

The flexibility provided in both the arrangement definition and the calculation of the design variables allows for the design of many different gearing systems. The procedures used in the program were developed around systems design in general; therefore, it is possible to apply the same techniques to many systems other than gearing.

Thesis Supervisor: Dr. Philip Meyfarth

Title: Lecturer for the Department of Mechanical Engineering

ACKNOWLEDGEMENTS

There are many people both at MIT and GE whom I would like to thank for their help on this project. Without their help much of the work done would have not been possible:

To Dr. Philip Meyfarth for his guidance throughout the project.

To Bob Light at the MIT CADLAB for his help on many of the numerical and programming problems which were encountered.

To David Serrano for his help in interfacing his routines with my program.

To Ramona Garabedian, Tom Steele and many others for their help in preparing this document.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES

GLOSSARY OF SYSTEM TERMS

System Element -

This is the most fundamental component in the system. It cannot be subdivided further and still exist.

System Connection -

A connection is a set of relationships between a specific set of elements.

System Topology -

The topology is the specification of all the system connections.

Implicit Constraints -

These are the constraints which are implied by the connections and elements of a specific topology.

User-Defined Constraints -

These are additional constraints which can be added by a system designer.

System Constraint Equations -

This is a set of equations which represent all of the implicit and user-defined constraints.

Arrangement Definition -

The arrangement definition is the combination of the system topology and the system constraint equations.

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

This thesis describes a special method for using computers to aid in the design of systems. Before going into the detail of the method which has been developed, some of the terms and concepts of a system design will be introduced.

A system is any collection of elements which are connected such that they can perform a specific function. The term "elements" refers to the fundamental building blocks of the system. Elements are described by a set of physical properties and cannot be subdivided any further. Each of the elements of a system can be identified as a certain type based on the element's properties. For example, resistors, capacitors, and transistors would be some of the possible element types in electrical circuits. In many systems there are only a small number of element types, but each type can be used repeatedly throughout the system. With this kind of system there is a large variety of possible arrangements for the system elements.

In order to define the terms which are used for a system definition, it is necessary to describe how they are related. The diagram shown in Figure 1-1 shows the hierarchy of a system definition. An "element" is introduced into a system by defining its relationship to the other elements in the system. This relationship is called a "system connection". The "system topology" is the specification of all the system connections. Associated with each system connection and system element are a set of implied relationships which are called the "implied constraints". "User-defined constraints" are relationships which are specified in addition to the implied constraints. The difference between implied and user-defined constraints is explained in greater detail in Chapter 2. An "arrangement definition" describes a specific system by combining the topology definition and the system constraints.

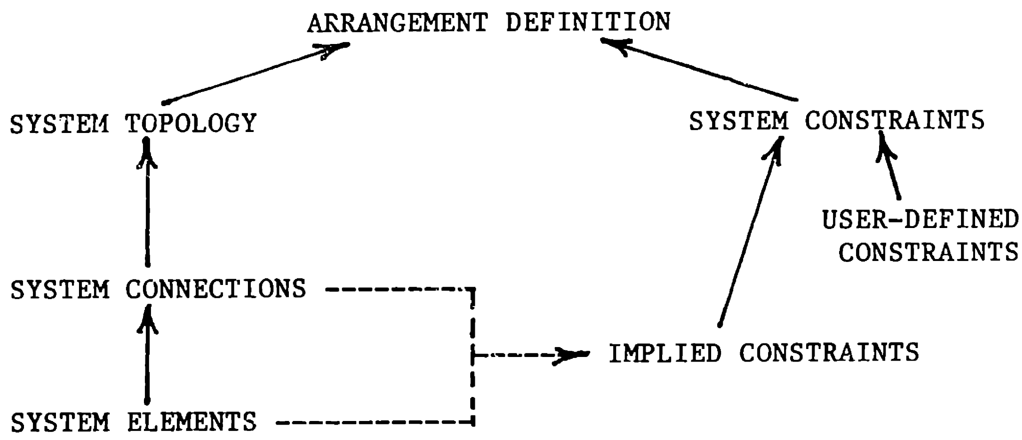


Figure 1-1 Hierarchy of the System Definitions

An example which demonstrates the interrelationships of the system definitions is a belt-pulley drive. The elements of this system would be pulleys and belts, and the connection would be the belt to pulley interface. The simplest useful topology would include two belt-pulley connections using one belt and two pulleys. Implied equations for the pulley elements might include relationships for the pulley geometry and any internal stresses. A belt might have equations relating the stresses in the belt to the belt tension. Each belt-pulley connection would have a set of equations relating the belt transverse speed to the pulley's rotational speed, and equations relating the belt tension to the forces imposed on the pulleys. The combination of all these equations is the set of implicit equations. An equation constraining the ratio of the pulley diameters could be added as a user-defined constraint. The combination of all the equations and the topology specification make up the arrangement definition for the belt-pulley drive.

The implied constraints for a system are represented by a set of equations. It is very difficult to design systems with many possible topologies since each system will have a different set of implied equations. An example of this type of system is an electrical circuit. Electrical circuits normally have a large number of each type of component (resistors, capacitors, etc.), and therefore an almost infinite number of possibilities exist for how they can be assembled. For each topology, the equations which govern the physical characteristics of the system will change. If a computer program is to be used to aid in the design of this kind of system, there must be a method for applying the correct set of equations for a specified topology. Traditionally, this has been done by writing all of the possible equations into the computer code, and then selecting the correct equations after a topology is specified. This method requires that the programmer anticipate all of the possible variations that a designer might require, which is difficult when the number of elements in the system is large. An alternate method is to not include the equations in the program, but based on the specified topology, select them from a data base of equations for the elements and element connections. This latter method is the basis for the software described in this thesis, and allows a designer and a programmer much more flexibility in handling the possible variations in the topology. An additional feature provided by the second method allows a designer to add equations which represent any user-defined constraints. This allows for the existence of many possible equation sets with the same topology. The complete set of equations is referred to as the system constraint equations.

1.2 OBJECTIVE

The objective of this research is to develop a computer method which will aid in the design of the systems described in the previous section. These systems are called "variational systems" due to the large variety of arrangement definitions which are possible. The type of system can be either mechanical, electrical, or any other engineering system. The steps of the design process for a variational system are shown in Figure 1-2.

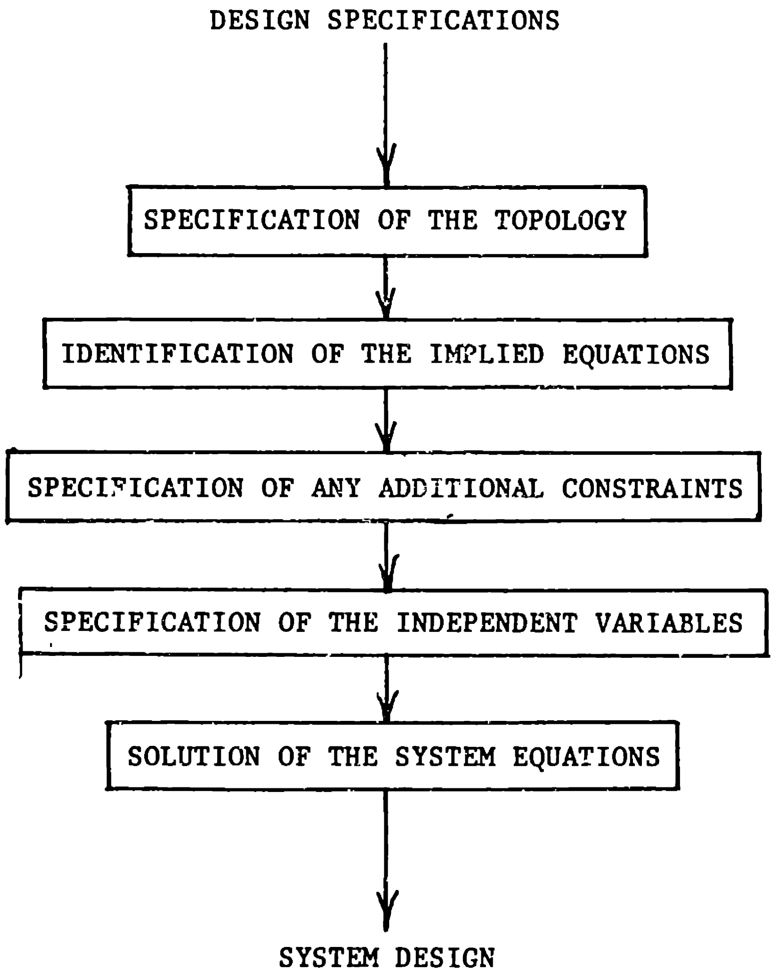


Figure 1-2 Variational System Design Process

The software being developed will aid all steps of the design process. When a set of design specifications is identified, a designer selects a topology and enters it into the program. For each connection and element in the topology specification, the program selects a set of equations from a data base. The element equations reflect the constitutive relationships within the element, and the connection equations reflect the relationships between the elements. User-defined constraints which are not implied by the topology can be added by a designer. These constraints are specified by equations which correspond to the desired relationship. An example of a user-defined constraint for an electrical circuit would be an equation which specifies the resistance of one resistor to be twice that of another. A constraint for a mechanical system might equate one of the size parameters of two different elements.

When all of the constraints have been specified, the independent variables and their values must be chosen by the designer. The independent variables are those variables in the constraint equations which are to be considered "constants" while the remaining variables are the "unknowns". The set of independent variables is entered into the program, which then calculates values for the remaining variables. After a design calculation is complete, the designer can choose a different set of independent variables and values for another calculation. This can be repeated until a satisfactory design is obtained. Also, any other step of the design process can be repeated in order to obtain a better design. The steps of this process and the computer method used are explained in detail in the following chapters.

A mechanical gearing system has been selected to demonstrate this solution technique. This type of solution scheme is particularly adaptable to gearing systems since they have only three types of elements (gear rotors, input devices, output devices), but they can have a large number of each with many possible topologies. The concepts that have been developed for a gearing system were chosen such that they apply to variational systems in general. Only solutions for gearing systems have been obtained as part of this thesis, but once the method has been successfully demonstrated, it will be possible to determine how it can be applied to many other similar systems.

1.3 RELATED WORK

Similar work has been done with the variational geometry methods for modifying a three-dimensional geometric model. These methods allow modification of a model by specifying values for the dimensions. The three-dimensional routines were developed by Lin [1], based on earlier work done by Light [2]. The concept used in variational systems analysis is very similar to the variational geometry methods, except it is applied to the variables which define a system rather than a geometric part. Equations for a variational system are based on a specified topology. These equations constrain the internal relationships of each element and the relationships between the elements. The equations for variational geometry are based on the dimensioning scheme which is applied. In this case, the equations constrain the relationships between the geometry parameters. A significant difference between the two techniques is that the variational system equations are loaded from a data base external to the program, but the variational geometry constraint equations are built into the program. Both methods use a modified Newton-Raphson method to obtain a solution for the equation set.

A preliminary design program developed by Serrano [3] allows a designer to sketch the geometry of a part to be designed and then allows the specification of design equations which apply to the part. The geometric dimensions of the sketch can be labeled and the same labels are used in the design equations. This allows a designer to specify a set of input parameters and lets the program calculate the remaining variables. The geometry of the part is updated using variational geometry. The concept of entering a set of symbolic design equations and then going on to solve the equation system is very similar to that which is used to solve variational systems; however, for the variational system design there is a set of implied equations that are stored in a data base in addition to the equations which are added by the designer. The preliminary design program could be used to design systems, but the equations for each element and each connection must be entered separately. For large systems this is time consuming and error prone.

A design scaling and optimization procedure was developed by Chang [4]. In this method, the design constraints or equations were represented by a set of penalty functions. These penalty functions are used by an optimization routine to calculate a new design for any change in the design parameters. The use of the penalty functions allows for inclusion of inequality constraints, but results in a limit as to the size and complexity of the system. The method used here for variational systems includes all of the constraints as equalities.

CHAPTER 2

DESIGN OF A VARIATIONAL SYSTEM

The process for designing a variational system is shown in the flow chart of Figure 1-2 on page 1-5. In this chapter the details of each step are defined. The computer method used to aid the process is described in chapter 3.

Based on the flow diagram, the steps which are required for the design process include:

1) Specification of the system topology

2) Identification of the system equations
   a) Implied equations
   b) Additional constraints

3) Specification of the independent variables

4) Solution of the system equations

The input to the process is a set of design specifications and the output is the system design. The system design includes a set of variables which can be used to describe all of the components and relationships in the system.

2.1 SYSTEM TOPOLOGY

The system topology identifies the connections between the elements of the system. In order to specify the topology for a system, it is first necessary to identify the possible element and connection types. These element and connection types will depend on the type of system which is being designed. For an electrical system the element types would be resistors, transistors, etc., and the connection types would be the electrical nodes or wires connecting the different element types. The relationship of elements and connections is explained in greater detail in the following sections.

2.1.1 SYSTEM ELEMENTS

System elements are the components of the system which can be described by a set of parameters and internal relationships between the parameters. An element can exist by itself, but it can also be related to other elements by a connection as described in the next section. Elements of a system are classified by types where each type will have associated with it a specific set of element parameters and relationships. For example, a gear rotor is described by a set of geometric parameters such as its diameter, length, the number of teeth on the rotor, etc.

The parameters and relationships that are associated with an element type are a function of the type of system being designed. An input device for a gearing system may have only a power and a rotational speed associated with it, but other parameters such as forces imposed by the device could also be included. When a set of element types is being defined for a specific system, a decision as to what parameters and relationships are to be included must be made.

Systems are hierarchical in nature. It is possible for an element of a system to be itself a system. For example, a steam turbine would be considered an input device in a gearing system, but if a steam turbine was being designed, the turbine would be the system, and the parts of the turbine would be the components. The turbine is only an element in the gearing system because there is not a subset of gearing related parameters for the turbine which can exist by themselves. In a steam turbine system there would be a set of elements such as turbine blades which are not found in gearing systems.

A topology specification will identify the quantity of each type of element. The variables and equations associated with each element can be assembled into a set of system equations after the topology has been specified.

2.1.2 SYSTEM CONNECTION

System connections are used to identify a specific relationship between a set of system elements. A good example is a mesh connection between two gear rotors. The mesh connection identifies the relationship between the geometric variables of the two rotors and also the forces which are imposed on the rotors as a result of the mesh connection. A system connection does not necessarily involve only two elements, but for a specific connection type, the number and type of elements is fixed.

System connections are all that are necessary to specify a unique topology since the type and number of elements are included as part of the connection specification. However, it is common for a designer to first specify an element to be added to the system, and then specify the connection. Once the connection has been identified, it is no longer necessary to retain a separate identification of the element.

2.2 SYSTEM GROUPS

System groups are not necessary for the definition of a system arrangement, but allow for a simplification of the total number of elements and relationships in the system. This category is used later in the implementation of the gearing system, but is introduced here due to its possible application to any type of system.

Groups are used to identify a collection of elements connected such that special constraints can be applied. For example, in a reduction gear when a pinion drives two gears, all elements must have the same tooth geometry and usually have the same axial face width. This set of elements would therefore be a group called a gear reduction.

The overall set of system equations and variables can be simplified by introducing a group variable and eliminating the corresponding element variable. In order for this to be possible there must be a constraint which specifies the element variables to be equal. If no groups were used for gearing systems, a face width variable would be included for each rotor, and a constraint equating the face widths of any meshing rotors would be required. When the reduction group is included, there is only one face width variable which applies to all of the rotors in a reduction, and the face width equation for mesh connections is not required. Again, it is not necessary for the reduction group, but it does simplify the final set of system constraints considerably. Also, the separation of a system into groups corresponds to the way a designer would actually look at the system.

After the topology definition has been completed, identification of the groups is possible. The elements and connections within the groups are also identified. Since there are many different ways groups can be defined, it may be possible for an element or connection to be part of several different groups.

2.3 SYSTEM CONSTRAINT EQUATIONS

Constraints are a set of equations which define the characteristics and relationships of the system. These equations apply to the elements and connections identified in the system topology. If groups are included in the system arrangement definition, there would be a set of constraint equations for each group. The two types of constraints are: implied constraints and user-defined constraints.

Implied constraints include the equations which must be applied to the topology that has been specified by a designer. These include relationships which must satisfy physical laws. For an element the equations will represent the constitutive or internal constraints of the element. An element constraint for an electrical resistor would be:

Voltage Drop = Current * Resistance

The equations for connections would represent the external constraints relating the connected elements. For two resistors connected in series:

Voltage Out (Resistor 1) = Voltage In (Resistor 2)

For each element and connection in the topology definition there will be a set of equations that must be added. The equation which corresponds to the above resistor example would be added for each resistor element.

In addition to the implied constraints, there may be additional constraints which can be identified by a designer. In an electrical circuit where there are two resistors, a designer might specify that the impedance of resistor 1 was twice that of resistor 2. Any number of additional constraints can be added, but since these additions are arbitrary, checks must be made to assure that the additional constraints do not conflict with any of the previous constraints.

The combination of the topology specification and the specification of the system constraint equations is called the "arrangement definition". Since it is possible to add different user-defined equations for the same topology, there can be several different arrangement definitions for the same topology.

2.4 SPECIFICATION OF THE INDEPENDENT VARIABLES

When all of the constraint equations have been assembled, it is necessary to identify the independent variables or constants. These variables are the constraint equation variables which are to be treated as constants while the remaining unknown variables are obtained using some solution process. For a set of N equations, there are M variables which are used to specify the equations. In order for the equation system to have a solution, (M-N) of the variables must be specified as constants. These constants are then assigned values. The number of unknown or dependent variables now equals the number of equations, and a solution for the system can be obtained.

For a given set of constraint equations, there are often many possible combinations of independent variables. In an equation as simple as:

$$A + B + C = 0$$

it is possible to have three different combinations of two independent variables each. Also, depending on the form of the equations, there may be certain variable combinations which are not valid because they conflict. With the equations:

$$A = B$$

$$A + B*C - D = 0$$

there are four variables and two equations; therefore, two variables must be specified as independent. If A and B are the two which are specified, the equation set would not be valid unless A and B were assigned the same value. With large numbers of equations, the number of possible invalid combinations increases dramatically; therefore, some method must be provided to assure that a valid set of independent variables is specified before attempting to obtain a solution.

2.5 SOLUTION OF A VARIATIONAL SYSTEM

Using the definitions from the foregoing steps, a solution can be obtained for all of the variables of the constraint equations. The solution method which is used to solve a set of constraint equations depends on the type of equations. If the equations are all linear then it is possible to use a direct method to solve them. However, there are normally coupled nonlinear equations which will make it necessary to use an iterative technique for the solution. Any step or steps of the variational system design process can be repeated as many times as necessary until a satisfactory set of design variables is obtained.

CHAPTER 3

IMPLEMENTATION OF A VARIATIONAL SYSTEM DESIGN PROCESS

The process which is used to design a variational system has been defined in the previous chapter. Since many of the steps in this process involve iteration and numerical analysis, the use of a computer to aid in the design of this type of system could be very helpful. Also, the use of computers will allow repeating the process many times with only minor variations in the system definition.

In order to demonstrate a computer implementation of the design process, a computer program has been developed which will aid in the design of reduction gear systems. Since a gearing system only has three types of elements (rotors, input devices, output devices), it is very suitable for the use of this type of solution technique. In gearing systems where there are large speed reduction ratios and several input and output connections, there can be a large variety of gearing arrangements. The following sections explain in detail the computer program which is used to define and the solve the equations of a gearing system.

3.1 GEAR SYSTEM DESCRIPTION

A gearing system is a set of toothed rotors which must be arranged such that they are connected by meshing with another rotor, and/or axially connected to either another rotor, an input device or output device. For a given number of input and output devices there are many possible arrangements which can be used to transmit the power. In the case of a single input device and single output device, some of the possible topologies are shown in Figure 3-1. Also, two possible arrangements for the topology shown as item (c) could be: 1) the speed of the upper rotor set being the same as the lower rotor set or 2) the speed of the rotor sets could be different. The first arrangement would require an additional equation which would equate the speed of the rotor sets.

The selection of the best arrangement may be a function of the speed difference between the input and output device, the amount of power being transmitted, the selection of the materials which are to be used or many other considerations. Normally, any one of the possible arrangements could be designed to carry a given combination of power and speed, but the resulting geometry could be either physically impossible or involve combinations which would be difficult or impossible to manufacture. For this reason selection of the optimum combination may involve choosing several gear combinations, calculating the design for each of these and then checking which combination gives the most desirable results. The most desirable arrangement may depend on several different factors such as minimum length, minimum height, minimum cost, etc.

After a designer selects an arrangement to be used, the topology and constraints must be specified. Once this is completed, the designer can select the initial input values for the independent variables and calculate the unknown or dependent variables. The solution for the system may then be repeated using different values for the independent variables, different sets of independent variables or a different arrangement definition. Many iterations of this process may be required before an acceptable arrangement and set of design variables is obtained.

The gearing domain implemented was limited to parallel offset gearing. Parallel offset gearing assumes all of the rotor centerlines are parallel to the z-axis of a standard x-y-z coordinate system. In addition, planetary gearing systems are excluded.

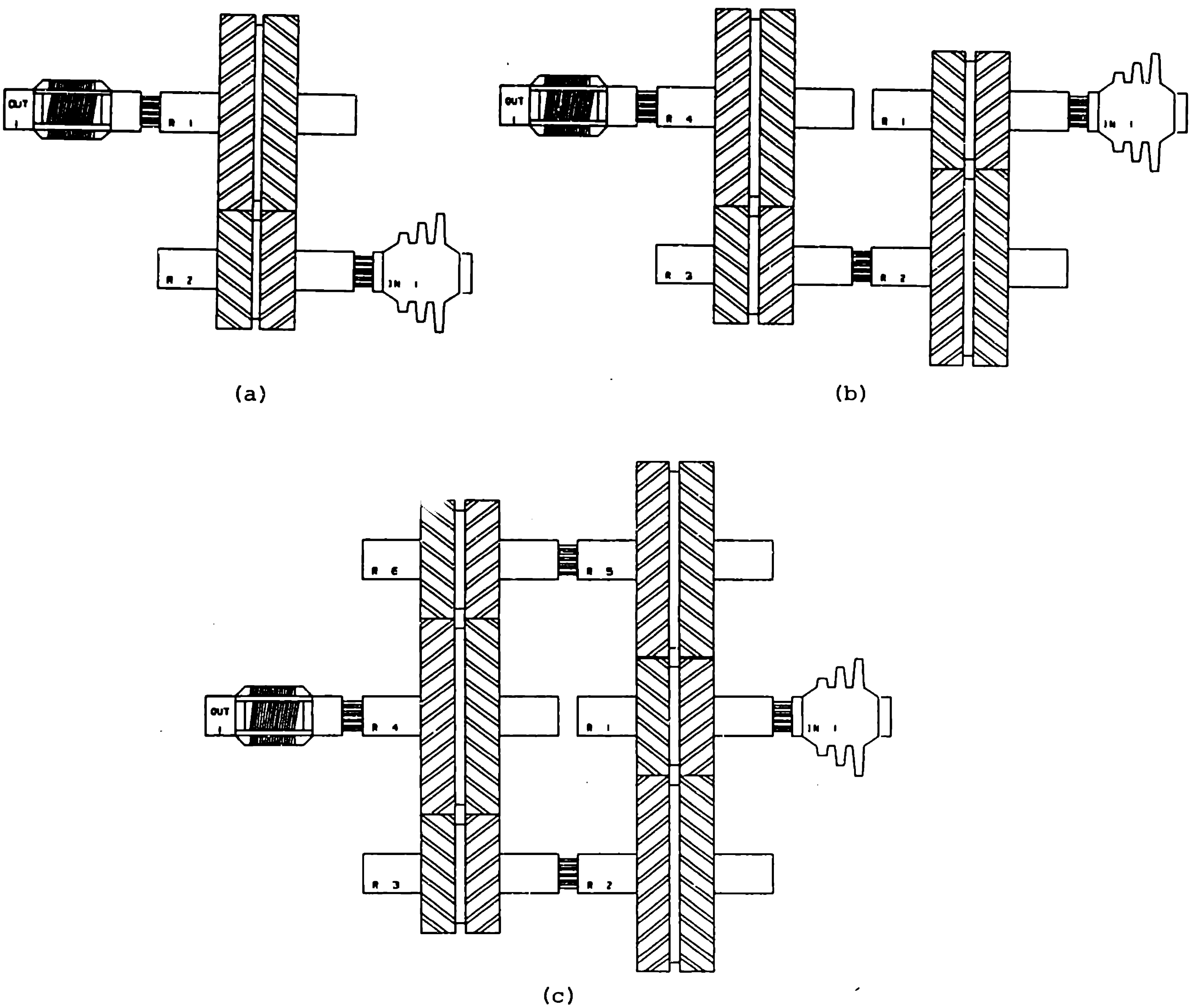


Figure 3-1 Alternate Topologies for One Input and One Output

3.2 DESCRIPTION OF THE COMPUTER IMPLEMENTATION

A computer program was developed to aid the designer with the process of designing a gearing system. The objective of the program was to provide a simple means for entering the complete arrangement definition for a system, and then to provide a flexible solution procedure which would allow for the many iterations required to obtain an acceptable design.

A combination of graphical routines and special entry forms are provided to enable to user to define the system topology and constraints. The graphical routines provide a means for specifying the arrangement topology using a digitizer and a data tablet. The location of the system elements and connections can be easily specified on a graphics display by using the digitizer. The program then assembles a numerical representation of the system topology which can be used for defining the system constraints. A set of special entry forms are used for entering labels for the topology elements, the system constraints, and the system variables. The combination of these forms and the graphical methods described above provide an easy and flexible interface between the program and the user.

After the system topology, constraints and independent variables have been specified, the remaining unknown variables are calculated by solving a set of nonlinear equations. A special routine has been included in the program which notifies the user when an invalid independent variable is selected. Normally the independent variables are identified by a set of design specifications which are imposed on a designer by a customer. There are many possible variations which could be imposed; therefore, allowing a designer to enter any valid set of input variables will accommodate all possible input specifications. Also, the user may choose many different combinations without altering the definition of the system topology and constraints.

The following sections describe the parts of the program and how they relate to the steps in the design process defined in Chapter 2. Since the discussion in Chapter 2 related to a general variational system design, the specific definitions which relate to gearing were not identified. Each of the items discussed in Chapter 2 has been repeated here, but applying specifically to gearing systems and the computer implementation.

3.2.1 TOPOLOGY DEFINITION

The gearing definitions which are used for each of the topology categories include:

A) Elements - 1) Gear rotors
2) Input device
3) Output device

B) Connections - 1) Mesh connections
2) Rotor to rotor shaft connections
3) Input shaft connections
4) Output shaft connections

C) Groups - 1) Gear reductions
2) Gear torque paths

Each topology definition category and the gearing definitions which are included in that category are described in the following sections. These definitions do not include all of the possible elements, connections, or groups of a gearing system but are the ones which are currently included in the program.

3.2.1.1 Definition Of The Gear Elements -

Gear rotors include any toothed gear element whether it is a pinion, a gear or an idler. A pinion is the element which has the smaller diameter of two meshing elements, and the gear is the larger element. The equations relating the variables of a rotor to each other are the same whether the element is a pinion or a gear; therefore, separate definitions are not required. The identification of whether a rotor is a pinion or a gear is necessary only when setting up the gear mesh connection equations.

An idler is a gear rotor which has no shaft connections, and therefore must have two rotors in mesh with it. The power is transmitted into the idler by one of the rotors and then is transmitted out to the other rotor. An example of a system with an idler is shown in Figure 3-2. An idler can be considered a pinion, gear or both depending on the size of the connecting elements. The definition applied depends on which of its two meshes is being considered. If the mating element for the mesh being considered is larger than the idler, the idler is a pinion; otherwise, the idler is considered to be a gear.

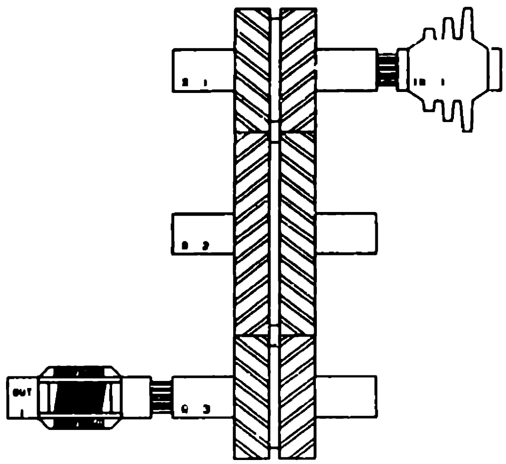


Figure 3-2 Gear Topology With an Idler

An input or output device is used to identify the power and speed which is associated with connections to the system. The same variables can be used to describe either an input or an output element, but they are separated so that the path of the power flow can be determined. For example, if an arrangement had three input/output connections as shown in Figure 3-3 the power transmitted by the two meshes would depend on which devices were inputs and which were outputs. If connections A and B were both input connections then the power level in the lower mesh would be the sum of the other two. However, if A was an input and B and C were outputs, the power level in the lower mesh would only be equal to the power level of the output connection C.

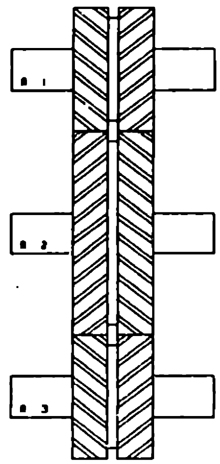


Figure 3-3 3 Input/Output Gear System

3.2.1.2 Definition Of The Gearing Connections -

A gear mesh connection is formed when two rotors are positioned such that the teeth of the rotors interconnect and can be used to transmit the torque. The parameters which are used to define a mesh connection involve geometric values, stresses, power levels and speeds. The interconnection between the teeth of the two rotors implies several relationships between the geometric parameters of the rotors. For mesh equations that are used in the program, only the geometric variables of the rotor which is the smaller of the two are used in the equation. A gear ratio is used to include the effects of the larger rotor. Any mesh connection in the program must be between a pinion and a gear. Mesh connections between two gears or two pinions are not considered to be valid. If a gear ratio of 1.0 is desired, either of two meshing elements can be specified as a pinion and then the gear ratio set to 1.0 when the input values are specified.

A shaft connection between two gear rotors means that the two rotors must have the same rotational speed and specifies the transfer of torque from one rotor to the other. Also, two rotors which are shaft-connected must be on the same axial centerline; therefore, their X and Y locations must be equal.

Input and output connections are used to transmit the power and speed of a connected device to a gear rotor. These also require that the input and output device be on the same centerline as the rotor.

3.2.2 GEARING SYSTEM GROUPS

One of the groups which is used in a gearing system is a gear reduction. A gear reduction is a set of gear rotors which are connected by a series of meshes. Since there is certain tooth geometry which must be the same for any two rotors that are meshing together, a series of rotors which are meshing together are all required to have the same tooth geometry. For example, if a pinion drives two gear rotors as shown in Figure 3-4, the axial face width of the gear teeth is assumed to be the same for each, and the pitch of the teeth and several other variables must be the same for all three rotors. These variables which are specified to be the same are defined as the gear reduction variables.

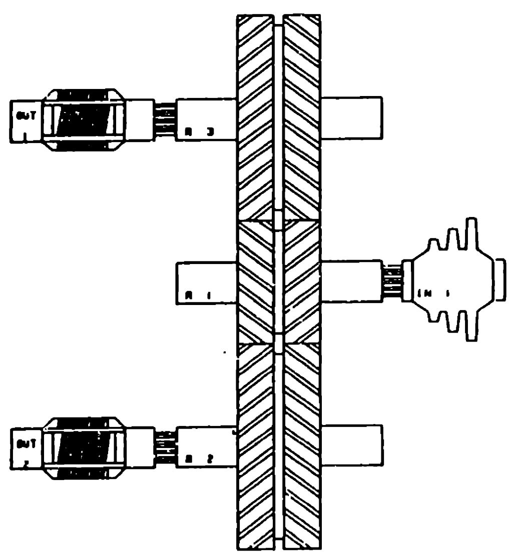


Figure 3-4 One Pinion Driving Two Gears

The torque path group is required in order to identify the torque which is transmitted by each connection in the gearing system. A rotor which has more than two connections will result in either a division or addition of the input torque levels. A typical multiple torque path condition is demonstrated by the arrangement shown in Figure 3-5. In this arrangement the power being applied to the system by the input device is divided or split into two torque paths. The power level in each torque path is a fraction of the input power level, but their sum must be equal to the total input power. The upper torque path includes the shaft connection and the two mesh connections on the upper half of the figure and the lower path includes the corresponding connections on the lower half of the figure.

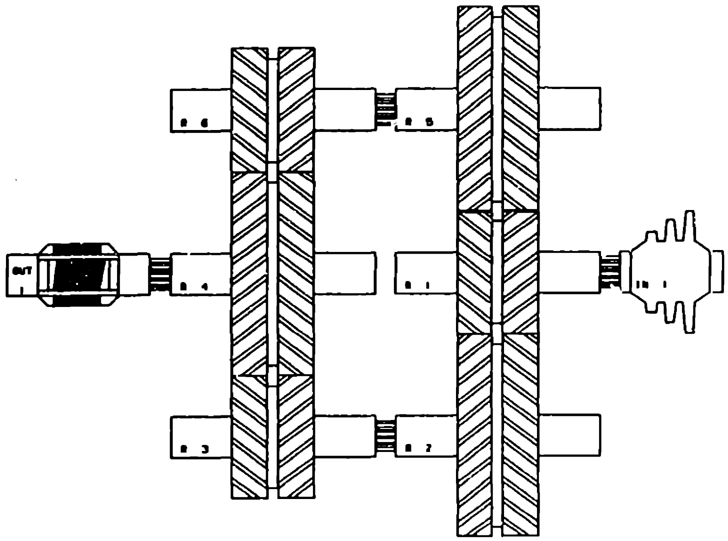


Figure 3-5 Double Torque Path Gearing Topology

3.2.3 GEARING SYSTEM EQUATIONS

The equations which are used to define the gearing system are divided into implicit equations and user-defined equations as explained in chapter 2. Both are described in detail in the following sections.

A set of symbolic equations are generated by the program for the implicit category, and the user-defined equations are also entered in symbolic form. Both types of constraint equations must be put into a format which can be used by the nonlinear solution method. A set of routines developed by Serrano [3] is used to put the equations into a tree structure as described in chapter 5. This tree structure can be used to calculate both equation residuals and partial derivatives for the solution of the equation system.

3.2.3.1 Implicit Equation Constraints -

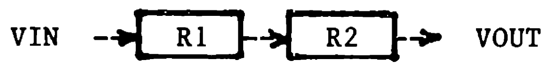
The implicit constraints are the equations which always apply to the gearing topology definition. A set of equations for each definition is stored in a general form and is loaded by the program for each element, group or connection. In order to explain this concept, it is easiest to use an example of an electrical circuit of two resistors connected in series. If a resistor element is defined as:

where VIi is the voltage on the input side of the resistor
VOi is the voltage on the output side of the resistor
Ri is the resistance of the resistor
and CURi is the current through the resistor.

The capital letters are used to identify what the variable represents and the lower case letter is used to identify the element number. The lower case letter will be replaced by the correct number in the equations final form. The resistor equation would be:

$$(VIi - VOi) = CURi * Ri$$

The two resistors are combined in the circuit:



The connection types for the above circuit from left to right are an input connection, resistor-resistor connection, and an output connection.

The connection equations would be:

1) Input: $VIN = VIi$

2) Resistor: $VOi = VIj$ and $CURi = CURj$

3) Output: $VOi = VOUT$

The subscript "i" refers to the resistor of the input and output connection equations. For the resistor-resistor connection, the "i" refers to the resistor with its OUTPUT connected to the node, and the "j" refers to the resistor with its INPUT connected to the node. Since this is defined as a two-resistor connection, it is not possible to have two inputs or two outputs connected. This is similar to the restriction against a pinion-to-pinion mesh connection. The identification of the numbers to be used for the subscripts is based on the system topology.

Making the substitutions into the resistor and the connection equations, the system of equations would be:

(VI1 - VO1) = CUR1 * R1

(VI2 - VO2) = CUR2 * R2

VIN = VI1

VO1 = VI2

CUR1 = CUR2

VO2 = VOUT

When the gearing system equations are loaded by the computer program, substitutions are made in order to relate the equation to the correct system elements. As for the equations in the example, the upper case letters are used to define the variable and the lower case letters are a subscript defining the element, connection or group number to which the variable belongs. For example, if the variable for the number of teeth on a pinion is NTe, when it is used for rotor 3, it becomes NT3. In the case of gear rotors, the "e" is normally used for the element number, but in some mesh connection equations it is necessary to know whether the element is a pinion or a gear and either a "p" or a "g" subscript is used. Since two rotors are involved for any mesh connection, the program must determine which one is the pinion before making any substitutions.

The equations which are used for each part of the gearing system are shown in the following tables. Each table corresponds to one of the definitions for the gearing topology. The equations are loaded for each of the system elements, connections and groups; therefore, when there is more than one of a particular item, the equations are loaded again with different values substituted for the subscripts. For items such as gear rotors and gear mesh connections, the same set of equations can be loaded many times. After the equations have been loaded with the proper subscripts, the next step is to specify any user defined constraints.

TABLE 3-1 Gear Rotor Variables

Pitch Diameter	PDIAe
Number of Teeth	NTe
X-Position	XLOCe
Y-Position	YLOCe
Z-Position	ZLOCe
Rotational Speed	RPMe

Note: The "e" indicates the number of the element.
The subscripts "p" and "g" are used for two meshing elements, and "i" and "j" are used for shaft connected elements.

TABLE 3-2 Input/Output Device Variables

Device Power	HPINi
	HPOUTi
Device Speed	RPMINi
	RPMOUTi

Note: The "i" indicates the device number.

TABLE 3-3 Mesh Connection Variables

Center distance	CDm
Gear ratio	RATIOm
K-factor	KFACm
Power	HPMESHm
Length/Diameter Ratio	RATLODm
Tangential force per mesh	TDFm
Angle to the Pinion	ANGMm

Note: The "m" indicates the mesh number.

TABLE 3-4 Gear Reduction Variables

Face Width - Active	FWr
Face Width - Total	TOTFWr
Gap Width	GAPr
Transverse diametral pitch	TDPr

Note: The "r" indicates the reduction number.

TABLE 3-5 Gear Rotor Equations

```
PDIAe =NTe /TDPr
```

TABLE 3-6 Gear Mesh Equations

```
RATIOm = PDIAg /PDIAp
CDm =(PDIAp +PDIAg )/2
RPMp =RATIOm *RPMg
KFACm =(RATIOm +1.)/RATIOm *(126050.*HPMESHm )/(RPMp *FWr *PDIAp **2)
LODRATm =(FWr +GAPr )/PDIAp
HPMESHm =HPTQRt
XLOCp =XLOCg +CDm *COS(ANGMm )
YLOCp =YLOCg +CDm *SIN(ANGMm )
```

TABLE 3-7 Input Shaft Connection Equations

```
RPMe = RPMINd
HPSFTs = HPINd
```

TABLE 3-8 Output Shaft Connection Equations

```
RPMe = RPMOUTd
HPSFTs = HPOUTd
```

TABLE 3-9 Rotor-Rotor Shaft Connection Equations

```
RPMp =RPMg
XLOCp =XLOCg
YLOCp =YLOCg
```

TABLE 3-10 Gear Reduction Equations

HELAr = arc cos(TDPr/NDPr)

TABLE 3-11 Gear Torque Path Equations

HPSFTs = HPTQRp *n
HPSFTs =HPTQRp + ...

Note: The "....." indicates repeated terms of the same form as the first term on the right side of the equals sign.

3.2.3.2 User-defined Equation Constraints -

The user-defined constraints are a set of equations which are added to the implicit equations and further constrain the possible solutions for the system. These constraints are entered by using the symbolic form of the equation and adding it to the set of system equations. The variables which are used in these equations must correspond to the variables in the implicit equations.

As each constraint is added, a check is made to assure that it does not conflict with any of the previously defined constraints. A constraint could be in conflict by either being a redundant constraint (stating the same thing as one of the previously defined constraints), or it could be a conflicting constraint (stating something which disagrees with one of the previous constraints). The method which is used to check the validity of a constraint is defined in more detail in Chapter 5.

3.2.4 DEFINITION OF THE INDEPENDENT VARIABLES

As each of the above constraints is added, a list of variables is assembled by the program. The user can assign each variable one of three possible types:

1. An always independent variable
2. An independent/dependent variable
3. An always dependent variable

"Always independent variables" are variables for which the designer must always specify a value, "independent/dependent" variables are variables which the designer has the option of either specifying a value or having the value calculated, and "always dependent" variables are always calculated by the program. After the designer has specified the variable types in the independent/dependent variable group, the total number of dependent variables must correspond to the total number of constraint equations. Also, as each variable of the independent/dependent variable groups is specified, a verification is made to assure that that specification does not conflict with all of the previously defined constraints and previously defined independent variables. This validity check is made using the same method which is used for checking the user-defined constraints equations.

The purpose of classifying the variables in this manner is to reduce the amount of user input required in the system solution routine. When the variable list is first assembled, all of the variables fall into the "independent/dependent" classification. This requires that all of the variables be presented to the user as possible inputs when a set of design values are being entered. However, by specifying variables as always dependent or always independent, the amount of effort required to enter a set of input data is reduced. The user enters the always independent values and then the user selects the variables which have an option (independent/dependent), and the always dependent variables are not seen until the system calculations are complete. If complete flexibility on the selection of the input variables is desired, then all of the variables are given the independent/dependent classification.

3.2.5 SYSTEM SOLUTION

The solution of the gearing system is obtained using a modified Newton-Raphson technique on the system constraint equations. A set of independent variables is assigned from the independent and independent/dependent classifications defined above. Due to the many combinations possible in the latter group, a routine is provided to check the validity of each of the independent variables as they are specified. This allows a designer to immediately select an alternate variable when the program finds a conflicting variable. The current number of equations and number of unknowns are displayed on each of the variable entry forms so that a designer knows when the number of independent variables required for a solution has been specified.

Associated with each variable is a set of values used to aid in the solution:

1. A lower limit
2. An upper limit
3. A nominal value

The upper and lower limits are used to control the stability of the convergence for the nonlinear solution method. The nominal value is used as a starting value for the iterative solution in the event that a previous solution has not been defined. Once an initial solution has been obtained for a specified arrangement then that solution is used as the starting value for any additional solutions.

Results can be displayed by several graphical procedures which plot the gear rotors to scale, or by a data printout. The graphical methods are necessary to show the relationships between the different element sizes; however, many of the loading and speed parameters cannot be shown graphically. For this reason a printout sheet is also required. The printout sheet is divided into sections for each reduction. All of the input and output variables are combined in one section.

CHAPTER 4

SAMPLE GEARING SYSTEM SOLUTION

A sample gearing system is described in this chapter to demonstrate the computer implementation. The arrangement is a double reduction gear reducer with one input and one output. In order to show the concept of the torque paths, the example unit has two torque paths. A schematic of the unit's topology is shown in Figure 4-1.

The computer program uses a series of menus to enable the user to select the different options in the design process. The design process in the program is divided into two routines:

1) ARRANGEMENT CREATION/EDITING

2) SYSTEM DESIGN CALCULATIONS

A data base management option is also provided in order to modify the standard data files the program uses.

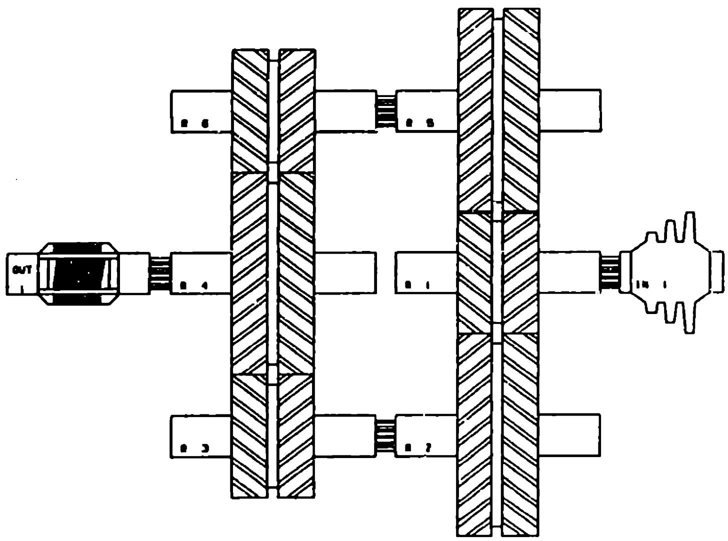


Figure 4-1 Sample Gearing Topology

A designer can create an arrangement definition using the arrangement creation/editing routine, save the definition, and then calculate the values for the design variables. In order to create a this definition, the ARRANGEMENT CREATION/EDITING option is selected. The initial entry menu which allows this selection is shown in Figure 4-2.

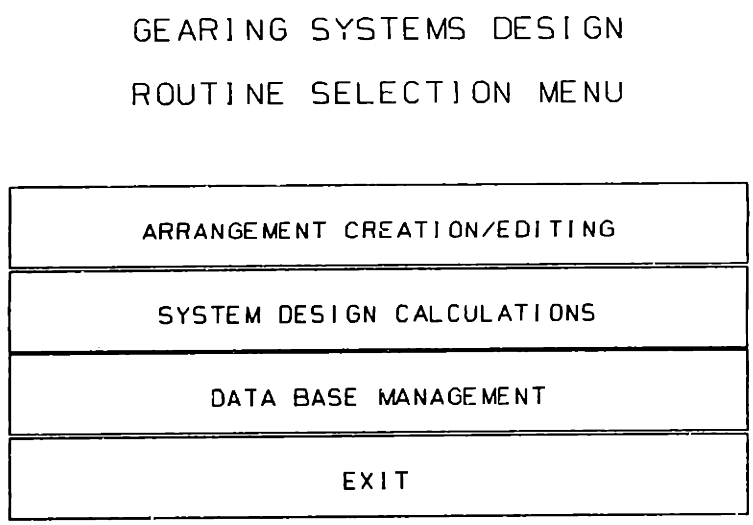


Figure 4-2 Main Entry Menu

The ARRANGEMENT CREATION/EDITING menu shown in Figure 4-3 includes the options necessary to specifiy the system topology, the constraint equations and the variable definitions and limits. It is possible to complete some of the steps in several different sequences while others must be always be done in a predefined order.

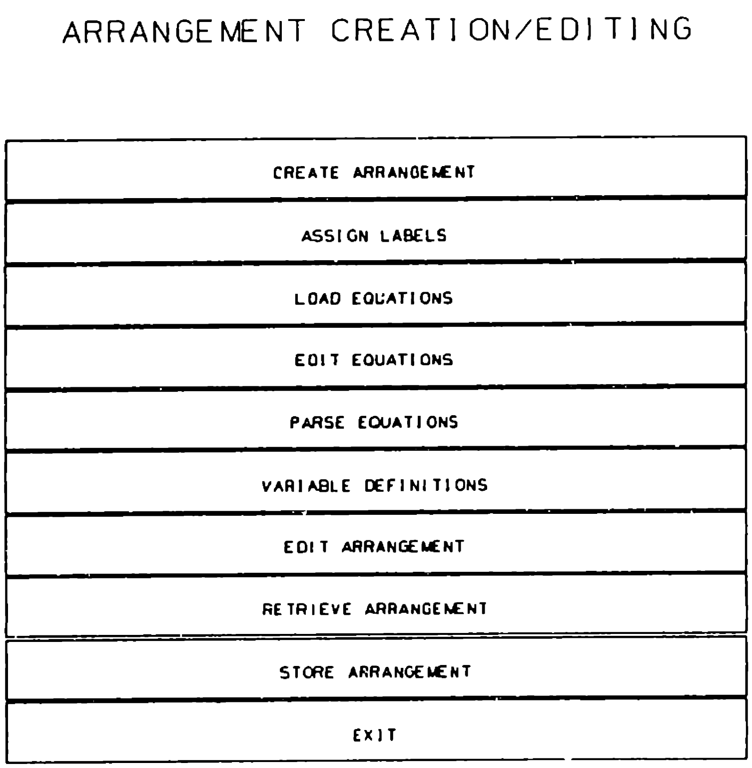


Figure 4-3 Arrangement Creation/Editing Menu

The steps which will be used for the example solution are:

1) Select the CREATE ARRANGEMENT option from menu

2) Enter the system topology

3) Select the ASSIGN LABELS option

4) Label the elements, connections and groups in the topology

5) Select the LOAD EQUATIONS option

6) Select the EDIT EQUATIONS option

7) Add a user-defined constraint equation

8) Select the PARSE EQUATIONS option

9) Select the VARIABLE DEFINITIONS option

   a) Load default limits and descriptions

   b) Make adjustments to limits and descriptions

   c) Modify the variable settings

10) Select the design calculation routine for a solucion

The CREATE ARRANGEMENT option requires the user to enter a file name for saving the arrangement data and a 60 character description of the arrangement. After these two items are entered, the program goes directly to the topology definition routine.

The topology definition is made by allowing the user to move the gear rotors, the input devices and output devices around on the screen of the graphics terminal with a digitizer and data tablet. The graphics screen shown in Figure 4-4 shows the arrangement with the first reduction pinion, one gear, one second reduction pinion, and the input device already added. Using the options shown on the menu the remaining items are added to complete the definition. When done, the EXIT option is selected and the program returns to the ARRANGEMENT CREATION/EDITING menu.

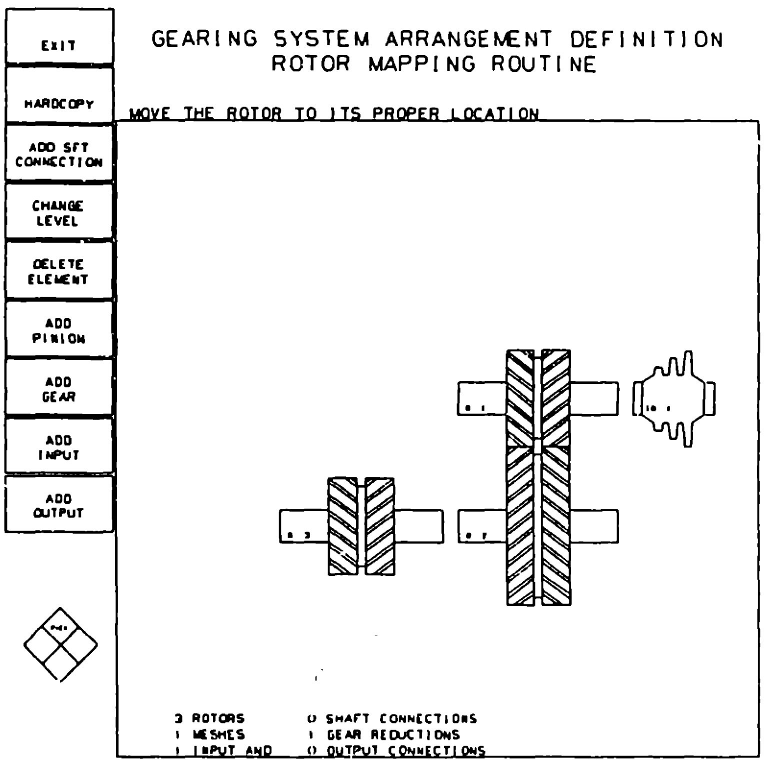


Figure 4-4 Topology Definition Screen

Labels can now be added to all of the components of the system by selecting the ASSIGN LABELS option. The labels are used to make it possible to distinguish between the variables which differ only in the number subscript. The label options are shown in Figure 4-5. By selecting the option and then the element that is to be labeled, a 12 character label can be entered for each of the elements, connections or groups.

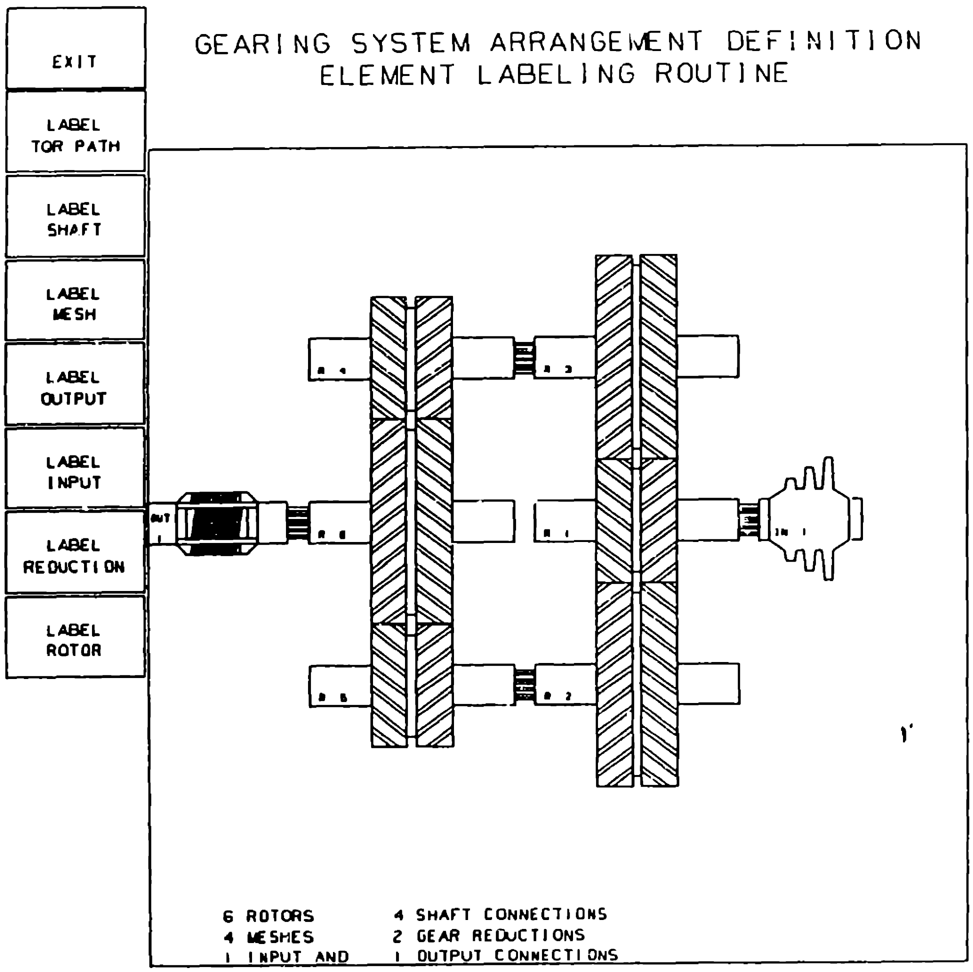


Figure 4-5 Label Assignment Screen

Some of the labels chosen for this example are:

HS PINION	for rotor number 1
HS/UP GEAR	for rotor number 2
HIGH SPEED	for reduction number 1

Now the set of implied equations can be loaded into the program by selecting the LOAD EQUATIONS option. This will use the topology which has been defined by the above steps to obtain the equations from the data base and load the correct subscripts. After loading the equations, the EDIT EQUATIONS options is used to add an additional constraint. The constraint added is:

NT02 = NT03

This requires the two first reduction gears to have the same number of teeth. Since all of the rotors in the first reduction have the same pitch, this also requires that the gear diameters be the same, resulting in identical high speed gear ratios. Both torque paths must terminate at the same output gear rotor; therefore, the overall ratio of both torque paths must be the same. Since the high speed gear ratios are the same for both torque paths, the low speed gear ratios must also be the same.

The VARIABLE DEFINITIONS option is subdivided into the loading of the default parameters and limits, the modification of the default parameters and limits, and the assignment of the variable independent/dependent settings. Each of these options is displayed on the menu shown in Figure 4-6. The default values must be selected first and then either of the other two options can be chosen.

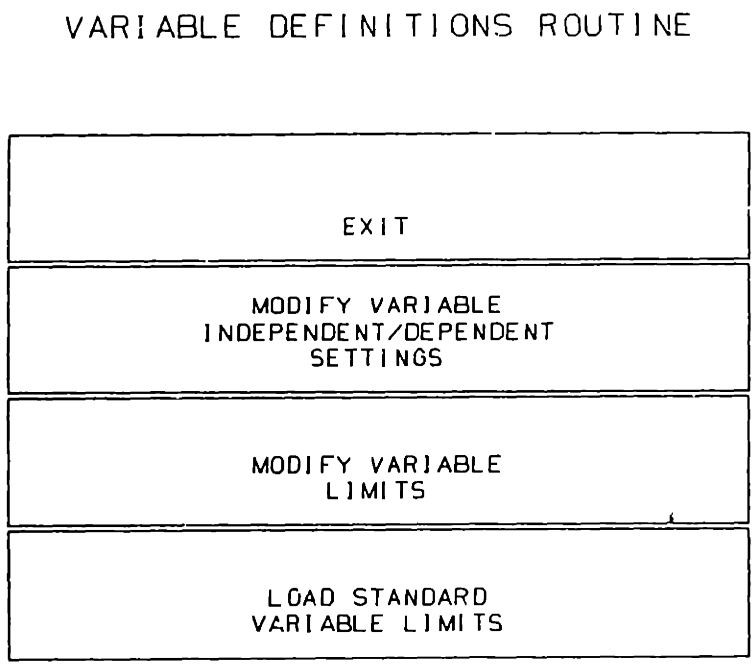


Figure 4-6 Variable Definition Menu

The variable limits are modified by using the scrolled form shown in Figure 4-7. The limits on the number of teeth for the high speed pinion are changed by moving the cursor to the line for that variable and hitting the RETURN key on the terminal. A second form, as shown in Figure 4-8, is now displayed and can be used to make the changes. For the variable which is displayed (number of teeth on the

HS pinion), the upper limit is decreased to 80. The HS pinion is always going to be the smallest rotor; therefore, the previous limit of 600 would be much higher than necessary. Similar changes could be made to the other variables.

V A R I A B L E S E L E C T I O N F O R M

Use the arrows to scroll to the variable to be modified and then use the ENTER to select the variable. Once the variable has been selected a form will be displayed which will allow modification of the variable description, lower limit or upper limit.

UPPER LIMIT 200.0 LOWER LIMIT 2.000

1	NT01	Number of teeth on the rotor	HS PINION	HS
2	TDP01	Transverse diametral pitch for reductio	HS	HS
3	PDIA01	Pitch diameter for the rotor	HS PINION	HS
4	NT02	Number of teeth on the rotor	HS/LW GEAR	HS
5	PDIA02	Pitch diameter for the rotor	HS/LW GEAR	HS
6	NT03	Number of teeth on the rotor	HS/UP GEAR	HS
7	PDIA03	Pitch diameter for the rotor	HS/UP GEAR	HS
8	NT04	Number of teeth on the rotor	LS/UP PIN	LS
9	TDP02	Transverse diametral pitch for reductio	LS	LS
10	PDIA04	Pitch diameter for the rotor	LS/UP PIN	LS

*** Hit the 0 key on the mini-keypad when done ***

Figure 4-7 Variable Limits Selection Form

V A R I B L E   L I S T   U P D A T E   F O R M

VARIABLE NAME -------------- NT01

VARIABLE DESCRIPTION ------- Number of teeth on the rotor

NOMINAL VALUE -------------- 36.0000

LOWER LIMIT ---------------- 25.0000

UPPER LIMIT ---------------- 60.0000

ELEMENT TYPE CODE ---------- 1

1 - Gear rotor 2 - Input Device 3 - Output Device
4 - Gear mesh 5 - Shaft connection 6 - Gear reduction
7 - Torque path

Figure 4-8 Limits Modification Form

The variable settings are changed using the scrolled form shown in Figure 4-9, and the return key is used to toggle between the three possible settings. The variable setting for the current variable is shown above the variable list as "CAN BE INPUT OR CALCULATED". A current variable is indicated by the location of the cursor. Using this form the parameters for the second gear in the first reduction and the second pinion in the second reduction are set to the always dependent (always calculated) group. These values do not need to be entered due to the additional constraint added above. Also, the input device speed and power are set as always independent (required input).

V A R I A B L E S E T T I N G S F O R M

Use the arrows to scroll to the variable to be modified and then use the ENTER to toggle the variable setting.

VARIABLE SETTING CAN BE INPUT OR CALCUALTED

1 NT01	**Number of teeth on the rotor**	**HS PINION**	**HS**
2 TDP01	**Transverse diametral pitch for reductio**	**HS**	**HS**
3 PDIA01	**Pitch diameter for the rotor**	**HS PINION**	**HS**
4 NT02	**Number of teeth on the rotor**	**HS/LW GEAR**	**HS**
5 PDIA02	**Pitch diameter for the rotor**	**HS/LW GEAR**	**HS**
6 NT03	**Number of teeth on the rotor**	**HS/UP GEAR**	**HS**
7 PDIA03	**Pitch diameter for the rotor**	**HS/UP GEAR**	**HS**
8 NT04	**Number of teeth on the rotor**	**LS/UP PIN**	**LS**
9 TDP02	**Transverse diametral pitch for reductio**	**LS**	**LS**
10 PDIA04	**Pitch diameter for the rotor**	**LS/UP PIN**	**LS**

*** Hit the 0 key on the mini-keypad when done ***

Figure 4-9 Variable Settings Selection Form

The final step is completed by going back to the main entry menu and selecting the SYSTEM DESIGN CALCULATIONS option. The DESIGN CALCULATIONS ROUTINE menu is shown in Figure 4-10. Shown at the bottom of the screen are the number of required inputs versus the number currently specified. Since the correct number have not been specified yet, the input variable option is selected. The variables are divided into a group for all of the input and output device parameters and a group for each reduction. For this case there are three groups. Each group is chosen and the values entered using a scrolled form such as the one shown in Figure 4-11. As the values are entered, the program checks if the inputs specified are valid. After completing the entry of the variables, the user returns to the DESIGN CALCULATION ROUTINE menu and selects the EXECUTE DESIGN CALCULATIONS option. When the calculation is complete, the results can be displayed using the graphics output shown in Figure 4-12 and Figure 4-13, or using the printout shown in Table 4-1.

SYSTEM DESIGN CALCULATIONS

EXIT
DISPLAY DESIGN RESULTS
PRINT DESIGN RESULTS
EXECUTE DESIGN CALCULATIONS
MODIFY INPUT VALUES
PLOT AXIAL VIEW
PLOT TORQUE PATH

Figure 4-10 System Design Calculations Menu

********* REDUCTION VARIABLE INPUT FORM **********

12 out of 12 required input values have been specified.

Current arrangement DT TYPE REDUCTION GEAR

Current reduction HS

#	DESCRIPTION	ELEMENT LABEL	TYPE	VALUE
1	Number of teeth on the rotor	HS PINION	I	36.00
3	Pitch diameter for the rotor	HS PINION	U	9.136
4	Number of teeth on the rotor	HS/LW GEAR	I	360.0
5	Pitch diameter for the rotor	HS/LW GEAR	U	91.36
15	Ratio for the meshing rotors.		U	10.00
16	Center distance for the mesh		I	50.25
19	Face width for the gear reduction	HS	U	14.09

Valid types are: I for an input or U for an unknown.

Use RETURN when done changing the variables.

Figure 4-11 Reduction Variables Entry Form

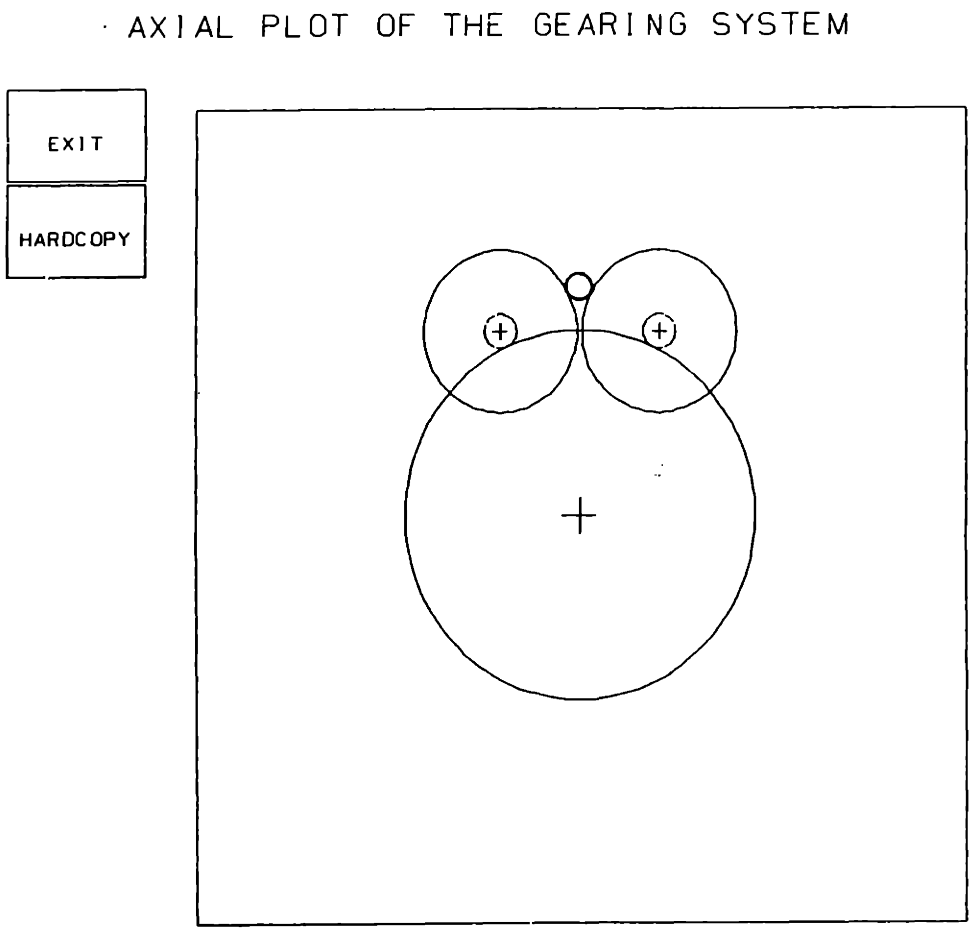


Figure 4-12 Axial Plot of the Final Design

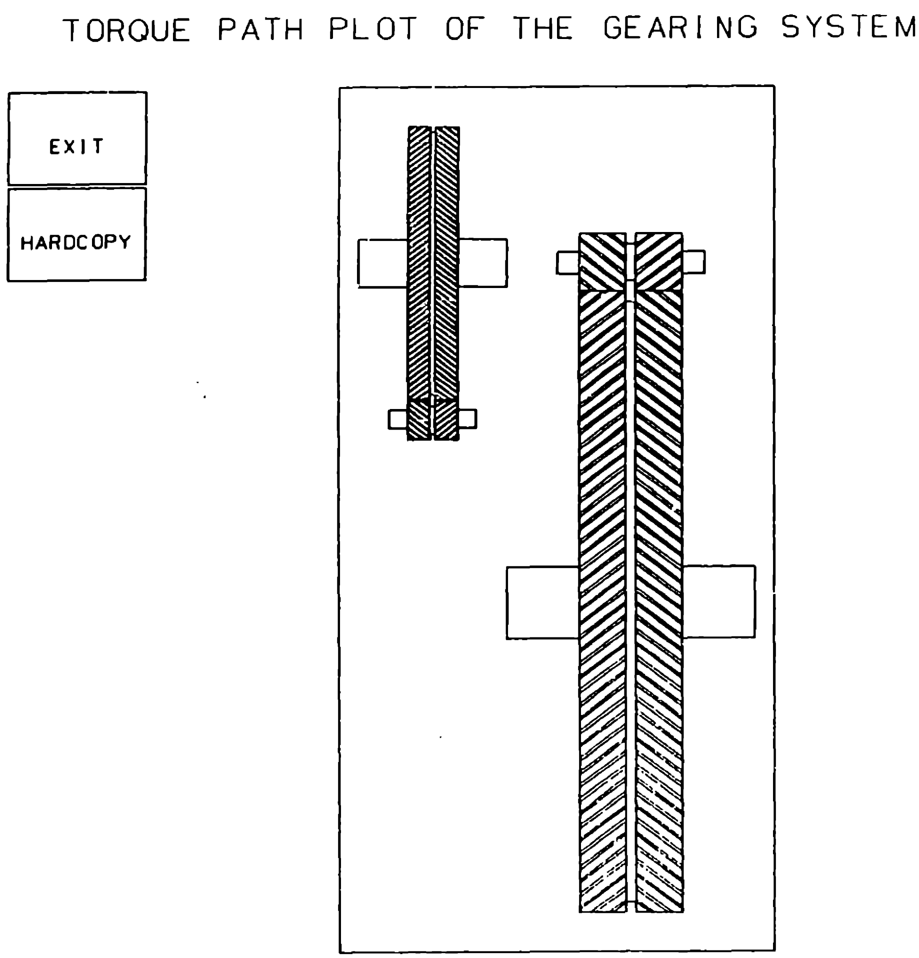


Figure 4-13 Scaled Section of a Torque Path

TABLE 4-1 Design Results Printout

Arrangement: DT Type Reduction Gear System

Data file: DTGEAR

Values for the input and output devices

1	Speed of the input device	TURBINE	6800.	INPUT
2	Horsepower of the input device	TURBINE	0.2000E+05	INPUT
3	Speed of the output device	PROPELLER	90.00	INPUT
4	Horsepower of the output device	PROPELLER	0.2000E+05	CALCULATED

TABLE 4-1 Design Results Printout (cont)

Values for HS Reduction

1	Number of teeth on the rotor	HS PINION	51.43	CALCULATED
2	Transverse diametral pitch for reduction	HS	4.430	CALCULATED
3	Pitch diameter for the rotor	HS PINION	11.61	CALCULATED
4	Number of teeth on the gear	HS/LW GEAR	360.0	INPUT
5	Pitch diameter for the rotor	HS/LW GEAR	81.26	CALCULATED
6	Number of teeth on the rotor	HS/UP GEAR	360.0	INPUT
7	Pitch diameter for the rotor	HS/UP GEAR	81.26	CALCULATED
8	Ratio for the meshing rotors.		7.000	INPUT
9	Center distance for the mesh		46.44	CALCULATED
10	Rotational speed of the rotor	HS/LW GEAR	971.4	CALCULATED
11	Rotational speed of the rotor	HS PINION	6800.	CALCULATED
12	Face width for the gear reduction	HS	12.09	CALCULATED
13	Horsepower transfered by the mesh		0.1000E+05	CALCULATED
14	K-factor for the mesh		130.0	INPUT
15	Gap for elements in this reduction	HS	3.000	INPUT
16	Length to pinion diameter ratio		1.300	INPUT
17	Mesh angle to the pinion		2.690	INPUT
18	X location of a rotor	HS/LW GEAR	41.78	CALCULATED
19	X location of a rotor	HS PINION	0.0000E+00	INPUT
20	Y location of a rotor	HS/LW GEAR	91.68	CALCULATED
21	Y location of a rotor	HS PINION	111.9	CALCULATED
22	Ratio for the meshing rotors.		7.000	CALCULATED
23	Center distance for the mesh		46.44	CALCULATED
24	Rotational speed of the rotor	HS/UP GEAR	971.4	CALCULATED
25	Horsepower transfered by the mesh		0.1000E+05	CALCULATED
26	K-factor for the mesh		130.0	CALCULATED
27	Length to pinion diameter ratio		1.300	CALCULATED
28	Angle of the mesh		0.4516	CALCULATED
29	X location of a rotor	HS/UP GEAR	-41.78	CALCULATED
30	Y location of a rotor	HS/UP GEAR	91.68	CALCULATED
31	Total face width of reduction	HS	15.09	CALCULATED

TABLE 4-1 Design Results Printout (cont.)

Values for LS Reduction

1	Number of teeth on the rotor	LS/UP PIN	46.32	CALCULATED
2	Transverse diametral pitch for reduction	LS	2.711	CALCULATED
3	Pitch diameter for the rotor	LS/UP PIN	17.09	CALCULATED
4	Number of teeth on the rotor	LS/LW PIN	46.32	CALCULATED
5	Pitch diameter for the rotor	LS/LW PIN	17.09	CALCULATED
6	Number of teeth on the rotor	LS GEAR	500.0	INPUT
7	Pitch diameter for the rotor	LS GEAR	184.4	CALCULATED
8	Ratio for the meshing rotors.		10.79	CALCULATED
9	Center distance for the mesh		100.7	CALCULATED
10	Rotational speed of the rotor	LS GEAR	90.00	CALCULATED
11	Rotational speed of the rotor	LS/UP PIN	971.4	CALCULATED
12	Face width for the gear reduction	LS	27.75	CALCULATED
13	Horsepower transfered by the mesh		0.1000E+05	CALCULATED
14	K-factor for the mesh		175.0	INPUT
15	Gap for elements in this reduction	LS	3.000	INPUT
16	Length to pinion diameter ratio		1.800	INPUT
17	Angle of the mesh		1.998	CALCULATED
18	X location of a rotor	LS GEAR	0.0000E+00	INPUT
19	X location of a rotor	LS/UP PIN	-41.78	CALCULATED
20	Y location of a rotor	LS GEAR	0.0000E+00	INPUT
21	Y location of a rotor	LS/UP PIN	91.68	CALCULATED
22	Ratio for the meshing rotors.		10.79	CALCULATED
23	Center distance for the mesh		100.7	CALCULATED
24	Rotational speed of the rotor	LS/LW PIN	971.4	CALCULATED
25	Horsepower transfered by the mesh		0.1000E+05	CALCULATED
26	K-factor for the mesh		175.0	CALCULATED
27	Length to pinion diameter ratio		1.800	CALCULATED
28	Angle of the mesh		1.143	CALCULATED
29	X location of a rotor	LS/LW PIN	41.78	CALCULATED
30	Y location of a rotor	LS/LW PIN	91.68	CALCULATED
31	Total face width of reduction	LS	30.75	CALCULATED

Values for shafts and torque paths.

1	Horsepower transmitted by torque path	0.1000E+05	CALCULATED
2	Horsepower transmitted by torque path	0.1000E+05	CALCULATED
3	Horsepower transmitted by shaft	0.2000E+05	CALCULATED
4	Horsepower transmitted by the shaft	0.2000E+05	CALCULATED

CHAPTER 5

NUMERICAL METHODS

Three of the steps in the solution technique that require special procedures are the definition of the tree structure for the equations, solution of the system of equations and the method which is used to check for conflicting constraints or conflicting variable definitions. Each of these is discussed in greater detail in the following sections.

5.1 EQUATION STRUCTURE

The symbolic equations are put into a tree structure so that the iterative solution process is more efficient. The tree structure is used to define each of the operations in the equation and the order in which they are performed. For the equation

$$A = B * C$$

the tree structure is shown in Figure 5-1.

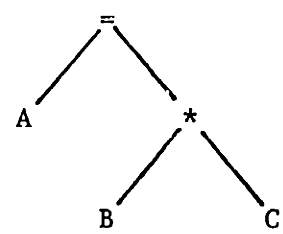


Figure 5-1 Equation Tree Representation

Each node on the tree would be an operation between two parameters, where a parameter would either be a variable, a constant or another node on the tree. Once the equation is in this form, the value of the equation and the partial derivative of the equation with respect to a certain variable can be calculated. The method used for breaking the equations up into a tree structure and the solution of the equation is described in detail by Serrano [3], but is summarized here for clarification.

In order to calculate the result, the value of the lowest node on the tree is calculated first and then the next lowest until the value of the very top node on the tree is calculated. For the equation

A = X * (Y * X)

the tree is shown in Figure 5-2.

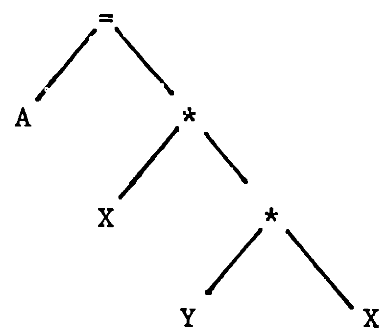


Figure 5-2 Tree for Sample Equation

In order to calculate the derivative of the equation we first must know the result of the operation at each node. After calculating the results for each node, we would then go through a derivative calculation for each node. Since any node only involves two parameters, a derivative of a node can be written as an equation which would involve the value of the two parameters and the derivative of the two parameters. For the right side of the equation tree shown in Figure 5-2, the result for the nodes would be

RESULT1 = Y * X

RESULT2 = X * RESULT1

The partial derivative d/dX for a multiplication is

DERV = RESULTi * DERVj + RESULTj * DERVi

and for the right side of the tree

DERV1 = Y * 1 + X * 0

DERV2 = X * DERV1 + RESULT1 * 1 = X*Y + X*Y = 2*X*Y

By replacing the equals sign with a subtraction operator the equation is converted to the form:

RIGHT SIDE - LEFT SIDE = 0

Carrying out the derivative calculation for the entire tree results in the partial derivative of the equation. Repeating the process for each of the unknown variables the Jacobian matrix is calculated for use in Newton's method. The exact form of the Jacobian matrix is described in section 5.2.1.

5.2 EQUATION SOLUTION

The solution of the equations is accomplished using a modification of Newton's method. In addition to the standard solution method several features have been added in order to control the stability of the solution. The Newton-Raphson method and the stability control are described below.

5.2.1 NEWTON-RAPHSON METHOD

The Newton-Raphson method uses a Jacobian matrix and a residual matrix to calculate a change in the solution vector dx. According to SHOUP [5], the Jacobian matrix (Rigidity matrix) and the residual matrix can be combined by the following equation.

$$
\begin{bmatrix}
dF1/dv1 & dF1/dv2 \ldots & dF1/dvn \\
dF2/dv1 & dF2/dv2 \ldots & dF2/dvn \\
 & \vdots & \\
dFn/dv1 & dFn/dv2 \ldots & dFn/dvn
\end{bmatrix}
\begin{bmatrix}
dx1 \\ dx2 \\ \vdots \\ dxn
\end{bmatrix}
=
\begin{bmatrix}
-f1 \\ -f2 \\ \vdots \\ -fn
\end{bmatrix}
$$

where dFi/dvj indicates the partial derivative of equation i with respect to variable j, dxi indicates the correction to be made to variable i and fi indicates the residual of equation i.

Using a set of initial values for the unknowns, the Jacobian matrix and the residuals are calculated. The above equation is used to calculate a correction dx for each of the unknowns. The values of the unknowns are then corrected by the dx array and the procedure is repeated with the new values for the unknowns. The iteration continues until the value for dx becomes very small relative to the actual values of the unknowns.

5.2.2 CONTROLLING THE STABILITY OF THE SOLUTION

Since there can be a large number of equations and many can be nonlinear, the solution of the equations can become unstable. Two methods are used in order to control the stability of the iteration. Also a set of initial default values are provided for each variable in order to have a good set of starting values.

A common method used for a Newton-Raphson solution involves a relation factor used to modify the correction dx. This relaxation factor is based on the relative magnitude of the actual value of the unknown and the value of the correction vector dx. When dx is large the relaxation factor is small and slows down the change in the unknown. As the variable values approach a solution, dx becomes smaller and the relaxation factor increases.

The second method involves checking the unknown value after it has been corrected by dx. If the new value is within the limits specified for the unknown, the new value is used. However, if the new value would exceed the limits on the unknown, the correction is decreased in order to keep the new value within the specified bounds. This method prevents the possibility of any mathematical overflow in the program. When this check was added to the program many of the larger equation sets which had been unstable now converged.

5.3 VERIFYING THE CONSTRAINTS

Since a designer can add any user-defined constraint or specify any variable as a constant, there must be a means for verifying that the constraint or constant is a valid one. To check both possibilities requires performing an upper triangularization on the Jacobian matrix. If the Jacobian becomes singular, there are conflicting constraints. By retaining a record of the operations performed on the Jacobian, it is also possible to determine which constraints conflict. The details of this method are described by Light and Gossard [8].

The matrix operations are initially performed on the implied set of equations, and then are performed on each additional equation or each additional assignment of an independent variable. Once the initial adjunct matrix and the upper triangularized Jacobian matrix have been developed, it is only necessary to work with the terms of any additional equations in order to get the new matrix into an upper triangularized form. The addition of an independent variable is treated as an equation. For example, if we added the variable A as an independent variable we would add the equation

A = [a constant]

to the system in order to check for whether this was a valid. This corresponds to adding a new row to the Jacobian matrix which has a 1.0 in the column corresponding to variable A and zeros for all other terms. If this causes the Jacobian matrix to become singular then A cannot be a constant. The user is informed that this variable conflicts and must go on to pick a different variable to be independent.

CHAPTER 6

COMPUTER METHODS

6.1 HARDWARE

The host computer on which the program was implemented is a Digital Equipment Corporation (DEC) VAX-11/750 which resides in the Computer Aided Design Laboratory at Massachusetts Institute of Technology. The terminals which were used to run the program were a DEC VT-100 series which allowed the use of DEC forms management software. The graphics device used was a monochromatic atek 7200 vector refresh display which included hardware for performing display transformations.

6.2 SOFTWARE

All of the programing was done in the VAX-11 Fortran utilizing fortran subroutine libraries for many of the graphical and numerical implementations. The subroutine libraries that were used included:

1) HARWELL ROUTINES

2) GRCORE ROUTINES

3) PARSING ROUTINES

4) SPARSE MATRIX ROUTINES

DEC's Forms Management software is used for many of the input and output interfaces; therefore, a VT-100 series terminal is required to take advantage of these features. An indicator is included in the program to identify the terminal which is being used. If the terminal is not a VT100 and no graphics device is available, most of the routines in the program cannot be used.

The graphics software which was used to support the megatek was the WAND subroutine package. The GRCORE routines discussed below call the necessary WAND routines when the Megatek 7200 is identified to the program as the graphics display which is to be used.

6.2.1 EQUATION SOLUTION ROUTINES

The subroutines that were used to solve the system of equations were developed by the Computer Science and Systems Division of AERE Harwell, Oxfordshire, England. These routines were set up to solve a set of linear equations with all of the storage and calculations being done in a sparse form; therefore, not requiring storage space for any of the zero elements of the Jacobian matrix. Even though the routines are set up for the solution of linear equations, it is possible to use them to solve a set of nonlinear equations by recalculating the Jacobian matrix between iterations.

The subroutines required were double precision routines MA28AD and MA28CD. The double precision routines were used in order to reduce errors due to roundoff. The MA28AC routine decomposes the matrix A from the equation

$$Ax=b$$

and creates a set of factors. The factors are used by the MA28CD routine in order to solve the equation for the x vector.

6.2.2 CORE GRAPHICS ROUTINES

The core graphics routines used were a set of subroutines developed at the MIT cadlab called GRCORE. These routines provided an interface with the WAND software package used to drive the Megatek 7200. Using GRCORE provided an interface with the current graphics hardware while still allowing the flexibility to expand to other devices. The GRCORE routines used included transformation and segmentation capabilities that may not be available on all graphics terminals.

The gearing design program allows for alternate methods of input/output depending on the available graphics device. It is possible to run the programs on a terminal with no graphics capabilities but many of the tasks which are very simple with the use of graphical input become very difficult and complex. Some of the more advanced graphics capabilities that are utilized include two dimensional transformations of screen segments, multi-button digitizer input and changing the visibility of and deleting segments. These capabilities made the process of mapping out a gear arrangement much easier than it would have otherwise been.

6.2.3 EQUATION PARSING ROUTINES

The equation parsing routines which were developed by Serrano [3] will decode a set of symbolic equations into the tree structure described in chapter 5. In order to decode the equations, the symbolic form of the equations was loaded into a fortran character array. By calling a parsing routine for the equation array, the tree structure is developed and loaded into another set of arrays (the equation stack). The equation stack can be used for calculating the residuals and derivatives. The routines which were used to perform the calculations on the equation stack were developed by the author.

6.2.4 SPARSE MATRIX ROUTINES

A set of sparse matrix routines available at the MIT CADLAB were used in order to perform operations on matrices in the sparse matrix format. These routines were developed by Robert Light. Since the validity checks on the system constraints and variable definitions required, a Gaussian elimination to be performed on the Jacobian matrix, these routines allowed performing the required operations in a sparse matrix format. Also, since the Jacobian was stored in double precision, the subroutines performed all of the operations with double precision arithmetic.

CHAPTER 7

RESULTS AND RECOMMENDATIONS

7.1 RESULTS

A method for modeling and designing a variational system has been developed and demonstrated. By storing the equations in a data base rather than incorporating them into the computer program, it is necessary to the use only the equations which apply to the system arrangement being designed rather than including all possible equations for all possible arrangements. Also, adding or modifying the equations can be done without making changes to the program. Use of the symbolic parsing routines allows the designer to input any type of design constraint by specifying the equation or equations which represent the constraint.

System component definitions have been broken up into two categories:

Element definitions

2) Connection definitions

After identifying the types of definitions which will apply to each category, it is possible to set up design variables and constraint equations for each. These constraint equations can be stored in a data base and assembled based on a specified topology. This definition scheme was demonstrated for the gearing system but applies to variational systems in general.

The program uses various graphical input procedures in order to allow the user to define the system topology. These graphical procedures were developed to correspond to the definitions which were used for the gearing system, but the method for storing the topology can be adapted to any system. After the system's topology has been defined it can be stored and retrieved for use on many different designs.

A computer solution is completed for the set of nonlinear equations that were assembled based on the system topology. The designer must specify a set of independent variables and values such that the number of dependent variables is equal to the number of equations. After the solution of the system of equations is obtained, a set of graphical and tabular output routines are available for displaying the results.

7.2 RECOMMENDATIONS FOR FURTHER STUDY

It has been demonstrated that the solution method developed works successfully for calculating the basic design parameters of a gearing system. In order to complete the gearing design process, there are additional features which must be included. These items might include fillet radii, chamfers, surface finishes, etc. If a computer program is to provide aid in designing these detailed features, a method must be included in the program to identify the order of calculations. By identifying the order of calculations, it will be possible to solve several sets of equations, one after the other, rather than solving them as a complete set. Separation of the equations into smaller groups should increase both the speed and stability of the solution.

There presently is no capability to use inequalities to constrain the system as there is in the procedure developed by Chang [4]. If the ability to define inequalities is included in this method it would allow for the optimization of the system. The optimization procedure could be added around the calculation for the system of equalities. By carefully selecting the inequalities, the total number of inequalities could be kept low, and avoid any convergence problems.

Many of the equations for the system have a direct solution; therefore do not require the iterative method. If each of the equations which could be solved directly were identified before the iteration begins, these values could be calculated exactly while the remaining equations were included in the iteration. No method for checking this condition was provided in the program which was developed; therefore, this potential time savings was not seen.

The inclusion of the preliminary design software developed by Serrano [3] would provide a means for adding new element types and new connection types to a system. Three of the major design functions which must be performed include:

1) Variational Geometry
2) Identification of Elements and Connections
3) Definition of a System

The software discussed in the previous chapters provides a method for defining a system based on predefined elements and connections. However, the addition of a new element would require a modification of the software. The program developed by Serrano aids a designer in the first two steps above. A combination of the two programs could provide the capability to create a new type of system, and then go on to create arrangement definitions. Since an assembly of components is a set of elements which are connected by a specific set of relationships, an assembly can also be treated as a variational system. Using a software combination of the preliminary design methods and the variational systems methods, it may be possible to develop a program which would allow the assembly of a set of components.

If the current variational systems design software was modified to include the preliminary design software, approximately 75 percent of the existing computer code could be used as is. Some additional interface routines would also have to be included. Also, if the variational systems software is to be used for a system other than gears, about 25 percent (the gear specific software) must be modified to apply to the new type of system.

REFERENCES

1) Lin on Variational Geometry
Master of Science Thesis, M.I.T., 1984.

2) Light,R. A., "Symbolic Dimensioning in Computer Aided Design", Master of Science Thesis, M.I.T., 1980.

3) Serrano, D.,"Mathpak: An Interactive Preliminary Design Package", Master of Science Thesis, M.I.T., 1984.

4) Chang,D., "Automatic Scaling of Assemblies for Computer-Aided Design",Master of Science Thesis, M.I.T., 1983.

5) Shoup, Terry E., "A practical guide to computer methods for Engineers", Prentice-Hall, Inc., New Jersey, 1979.

6) AERE Harwell, "MA28 - A Set of Fortran Subroutines for Sparse Unsymmetric Linear Equations",United Kingdom Energy Authority, 1980.

7) Digital Equipment Corporation, VAX-11 FMS Software Reference Guide, 1980.

8) Light, R.A., Gossard, D.C.,"Variational Geometry: A New Method for Modifying Part Geometry for Finite Element Analysis", Computers and Structures, Vol. 17, No. 5-6, June 1983.