IOWA STATE UNIVERSITY
Digital Repository

Retrospective Theses and Dissertations

Iowa State University Capstones, Theses and Dissertations

1996

Optimal synthesis of a planar four-bar mechanism with prescribed timing using generalized reduced gradient, simulated annealing and genetic algorithms

Gloria Kay Starns
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

 Part of the Mechanical Engineering Commons

Recommended Citation
Starns, Gloria Kay, "Optimal synthesis of a planar four-bar mechanism with prescribed timing using generalized reduced gradient, simulated annealing and genetic algorithms " (1996). *Retrospective Theses and Dissertations*. 11128.
https://lib.dr.iastate.edu/rtd/11128

This Dissertation is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

www.manaraa.com

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road. Ann Arbor. MI 48106-1346 USA
313/761-4700 800/521-0600

Optimal synthesis of a planar four-bar mechanism with prescribed timing using generalized reduced gradient, simulated annealing and genetic algorithms

by

Gloria Kay Starns

A Dissertation Submitted to the

Graduate Faculty in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Department: Mechanical Engineering

Major: Mechanical Engineering

Approved:

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa

1996

Copyright © Gloria Kay Starns, 1996. All rights reserved.

UMI Number: 9620987

Copyright 1996 by
Starns, Gloria Kay

All rights reserved.

UMI Microform 9620987
Copyright 1996, by UMI Company. All rights reserved.

This microform edition is protected against unauthorized
copying under Title 17, United States Code.

UMI
300 North Zeeb Road
Ann Arbor, MI 48103

DEDICATION

This work is dedicated to my best friend, Virginia Blackburn, for her support and love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

My thanks to Mr. Steve McGrew and the professionals of New Light Industries for the production and support of Generator™; their willingness to answer my questions and support their product contributed enormously to the completion of my work.

I would like to thank Mr. James D. Wong for his assistance in the fabrication of the models used for my defense presentation. I am, however, more grateful for the blessing of his and Cheryl's friendship.

My gratitude also goes to professors Raj Dahiya, Jim Hoekstra, Patrick Kavanagh, Ken McConnell, and Thomas Rudolphi. Their gifts as teachers have been immense.

Finally, I would like to thank my major professor, Dr. Donald R. Flugrad for his contributions to my personal and professional growth. His mentorship has been one of patience, enthusiasm, and friendship. I am a better person for having known him.

I. INTRODUCTION

Interest in straight line mechanisms has warranted full texts dedicated to the topic [1,2]; their uses encompass implementations as disparate as replacement of telescoping parts on cameras to mechanisms transmitting power in cylinders. Kurt Hain points out in Applied Kinematics that Burmester's contributions to modern mechanism synthesis came in part from his studies of straight line motion [3]. Watt's straight line linkage, a double-rocker, is used in automobile suspension systems to guide the rear axle up and down in a straight line [4]. Other straight line mechanisms include the Chebyschev straight-line linkage, the Robert' s straight line linkage, and the Hoeken's linkage. The Hoeken's linkage and the Chebyschev linkage are cognates of one another [4]. The tasks of these mechanisms is exclusively one of producing approximate straight line motion; the coupler point producing a prescribed path as the crank drives. The objective of producing large oscillations of the crank has been relegated, for the most part, to multiloop mechanisms like the Watt II six-bar. Hain points out that the task of producing large oscillations of a crank without running into poor transmission angles would be very difficult [3,5].

This dissertation will demonstrate the synthesis of a four-bar mechanism possessing a coupler point capable of following a straight path to conform to the motion of a linear actuator attached to the mechanism's coupler point. The crank, which normally acts as the driver, will in this case be the driven member of the mechanism. The actuator may be assumed to be a hydraulic cylinder confined to an enclosure that limits the actuator's lateral

motion. The actuator acts along a line while pushing the coupler point. This configuration of the coupler as the input or driving member, and the crank being the output or driven member is atypical of the normal arrangement, where the crank is driving the coupler. The force applied by the actuator to the coupler point must be capable of producing torque on the crank that yields no less than 180° of angular output of the crank. The sense of direction of the torque must remain consistent throughout the crank's motion. The synthesis of this mechanism will be accomplished by exploiting the features of *path generation with prescribed timing* and three specific optimization procedures. Figure 1.1 identifies and defines the problem and its associated parameters.

Mechanical optimization has included optimum synthesis of planar linkage coupler curves specified by position coordinates [6], selective precision synthesis [7], a general method of optimization that utilizes arbitrary limits of accuracy at various positions, and optimization of mechanisms using simulated annealing [8,9]. Optimization, it has been pointed out [10], is finally reaching wide recognition as evidenced by significant application in industry.

A discussion of various optimization procedures applied to mechanisms by Gabriele, Angeles, Liu, and Faik [11] indicates that designers of mechanisms have implemented numerous approaches to mechanism optimization and the authors point out various strengths and weaknesses of those methods. Among the procedures discussed by Gabriele, et al., is the generalized reduced gradient method, where robustness and accessibility are cited among its strengths. Also noted is that in the case of general constrained optimization, significant

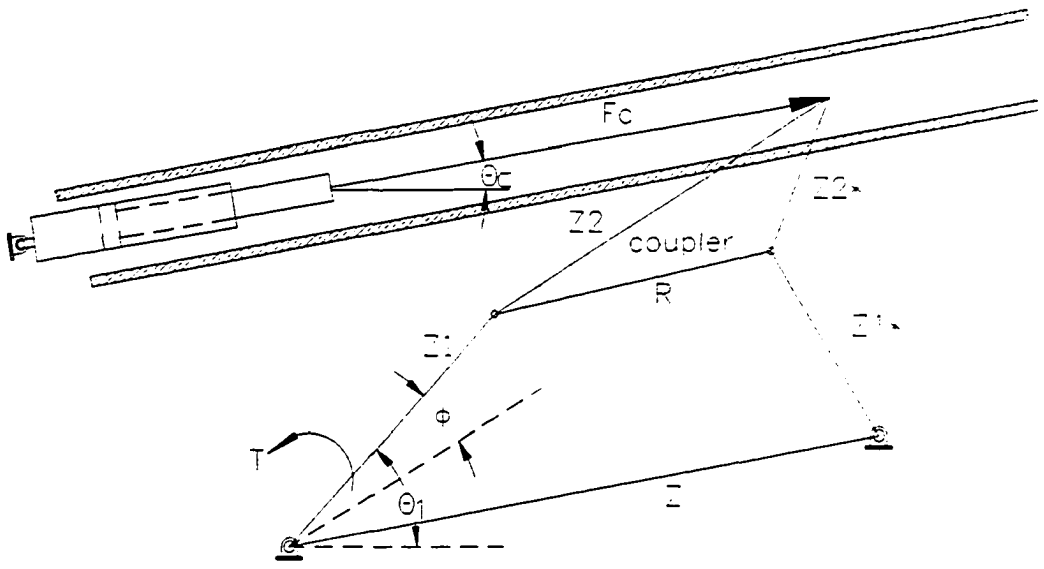

Figure 1.1 - Mechanism parameters and configuration

Fc = Actuating force

θc = Actuator's Orientation

Z_1 = Crank, Output Member

θ_1 = Crank Angle

Z_1^* = Follower

$R\text{-}Z_2\text{-}Z_2^*$ = Coupler, Input Member

Z = Ground

ϕ = Angular displacement of crank from θ_1

T = Output torque

development in gradient base methods has not occurred since the development of the Sequential Quadratic Programming (SQP) method. The gradient based methods tend to find local minima and as a result, the preponderance of recent research in mechanism optimization has been with respect to identifying global optima. Simulated annealing and genetic algorithms are each noted by the authors as strategies employing global sampling strategies. In addition to the generalized reduced gradient method, both simulated annealing and genetic algorithms are applied in this research and their respective strengths and weaknesses are assessed.

With respect to the combination of kinematic and static synthesis, Huang and Roth produced results for three planar linkages using a virtual work approach [12] and Raghavan's dissertation examined analytical methods for designing linkages to match force [13]. Force matching and kinematic synthesis have more recently been extended to the design of spatial closed-loop linkages [14].

The research presented within this dissertation may be distinguished from previous research in the following respects:

1. Static equations of equilibrium and virtual work terms are considered in the synthesis of the mechanism and its optimization. Synthesis of straight line mechanisms has been accomplished in the past with the task of path generation being the principal design objective with a crank acting as the driving member. The mechanisms synthesized within this work define the crank as the output member; the mechanism being driven by a linear actuating force applied to the coupler.

2. Prescribed timing requirements include a final angular displacement of the crank of no less than 180° while producing output torque in a consistent direction throughout the motion.
3. The transmission angle is defined for the specific application. It is non-traditional in that the applied load is on the coupler and the output member is the crank.
4. Genetic algorithms, simulated annealing and generalized reduced gradient methods are implemented to synthesize a mechanism with optimal characteristics that include minimization of the deviation in transmission angle from 90°.

Discussion and demonstration of the topics noted below follow:

- Derivation of constraint equations implemented in the generalized reduced gradient, simulated annealing, and genetic algorithm methods
- The generalized reduced gradient method
- Simulated annealing
- Genetic algorithms
- Sample mechanisms produced with hybrid approaches to optimization
- Conclusions and Summary

Chapter II, Model Development, presents the derivation of the equations of constraint used in the three optimization procedures and introduces terminology and notation used throughout the dissertation. The objective functions utilized within each optimization procedure are likewise developed and penalty functions are formulated.

Chapter III, The Generalized Reduced Gradient, briefly discusses the generalized reduced gradient method and the reasoning followed in its implementation. Some terminology associated with optimization in general is presented and the strengths and weaknesses in this application are enumerated. A sample mechanism synthesized with the

Generalized Reduced Gradient is included in the chapter accompanied by relevant demonstration of the mechanism's characteristics.

Chapter IV, Simulated Annealing, acquaints the reader with the simulated annealing algorithm and a discussion of its use is presented. Several mechanisms are synthesized using simulated annealing and each mechanism's behavior and characteristics are demonstrated.

Chapter V, Genetic Algorithms, begins with an introduction to genetic algorithms and includes a presentation and discussion of the formulation of the constraints and objective function used for mechanism synthesis and optimization. Synthesized mechanisms found with Genetic Algorithms are included.

Chapter VI, Hybrid Approaches, identifies optimized mechanisms found using hybrid approaches to optimization. Specifically, a mechanism found using a combination of the Generalized Reduced Gradient method, Genetic Algorithms and Simulated Annealing is presented. Also included is a mechanism synthesized with a hybrid of Genetic Algorithms and Simulated Annealing.

Chapter VII, Summary and Conclusions, discusses a methodology for identifying mechanisms requiring the features delineated in this work and makes some recommendations regarding pertinent issues involved with the various methods. Additionally, the efficacy and insufficiencies of each optimization method are discussed.

II. MODEL DEVELOPMENT

The generic mechanism shown in Figure 2.1 will be synthesized such that the following criteria are met:

1. The output torque on link member Z_1, the crank, is to remain in a consistent direction throughout the mechanism's motion. This requirement will ensure that the motion along the coupler point path does not change direction prior to the crank reaching at least 180° of rotation output.
2. The mechanism must remain "closed" in all positions.
3. The coupler path points must not deviate by more than a predetermined amount from straight line motion (the amount is not to exceed 3° of the slope of the path). A linear actuator is assumed to be attached to the coupler point and is required to push the coupler point along a path possessing a slope that is approximately equal to the actuator's angular orientation.
4. The transmission angle at any position must not deviate from 90° by more than a specified amount (the greatest deviation allowed in any instance is 50°).
5. The mechanism's pin joints are assumed to be frictionless.
6. Inertial effects are assumed to be negligible.

To ensure that the output torque maintains a consistent sense of direction, the virtual work done by the actuation force on virtual displacements is examined at each precision point. The virtual work term is determined as follows and as shown in the work by Huang and Roth [15]. Referring to Figure 2.2 the notation used in this development is as follows:

θ_i generalized coordinates, i = 1,3
Fc force applied by the actuator at the coupler point
T output torque produced on member Z_1
r vector from ground pivot A to the applied force
$\hat{i}$ unit vector along the horizontal axis
$\hat{j}$ unit vector along the vertical axis

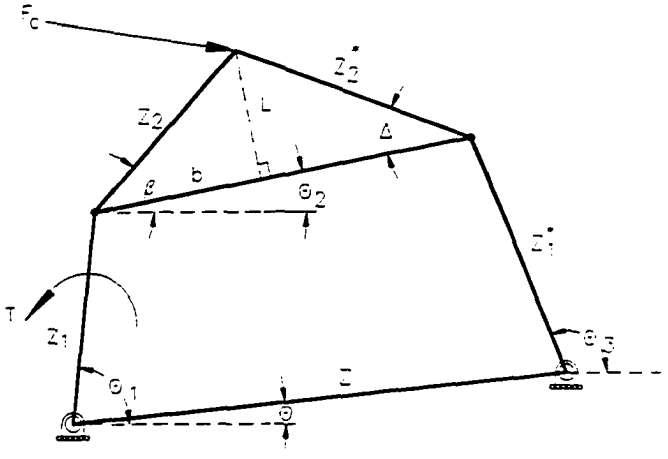

Figure 2.1: Generic mechanism under consideration

The virtual work in a given position is:

$$\delta W = T\,\delta\theta_1 + F_c\,\delta r \qquad [2.1]$$

The vector r is expressed as :

$$r = \left[Z_1\cos(\theta_1) + Z_2\cos(\theta_2+\beta)\right]\hat{i} + \left[Z_1\sin(\theta_1) + Z_2\sin(\theta_2+\beta)\right]\hat{j} \qquad [2.2]$$

The differential of the vector is

$$\delta r = \left[-Z_1\sin(\theta_1)\delta\theta_1 - Z_2\sin(\theta_2+\beta)\delta\theta_2\right]\hat{i} + \left[Z_1\cos(\theta_1)\delta\theta_1 + Z_2\cos(\theta_2+\beta)\delta\theta_2\right]\hat{j} \qquad [2.3]$$

Prior to taking the dot product of the differential term with the force vector, the virtual

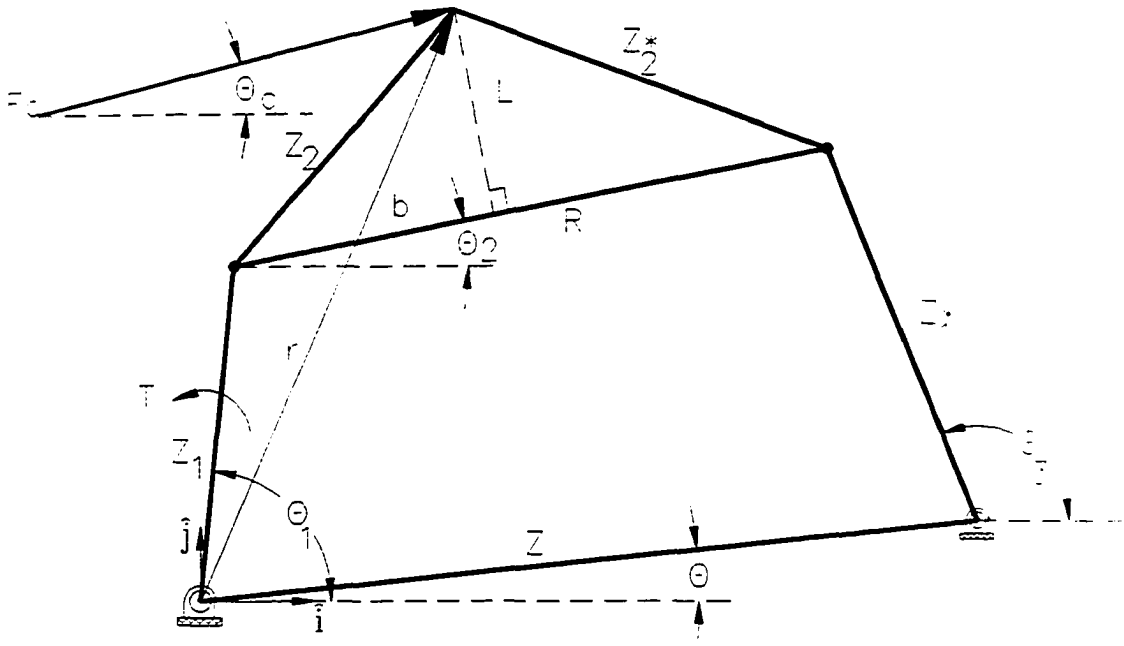

Figure 2.2: Virtual work parameters

displacement, $\delta\theta_2$ must be expressed in terms of $\delta\theta_1$. Traversing the loop $Z_1 + R = Z + Z_1{}^*$ and separating the loop equation into real and imaginary components gives,

$$Z_1 \cos(\theta_1) + R\cos(\theta_2) = Z\cos(\theta) + Z^*{}_1 \cos(\theta_3) \qquad [2.4]$$

$$Z_1 \sin(\theta_1) + R\sin(\theta_2) = Z\sin(\theta) + Z^*{}_1 \sin(\theta_3) \qquad [2.5]$$

The differentials of these terms are

$$Z_1 \sin(\theta_1)\delta\theta_1 + R\sin(\theta_2)\delta\theta_2 = Z^*{}_1 \sin(\theta_3)\delta\theta_3 \qquad [2.6]$$

$$Z_1 \cos(\theta_1)\delta\theta_1 + R\cos(\theta_2)\delta\theta_2 = Z^*{}_1 \cos(\theta_3)\delta\theta_3 \qquad [2.7]$$

Division of Eq. 2.6 by Eq. 2.7 gives

$$\frac{\sin(\theta_3)}{\cos(\theta_3)} = \frac{Z_1 \sin(\theta_1)\delta\theta_1 + R\sin(\theta_2)\delta\theta_2}{Z_1 \cos(\theta_1)\delta\theta_1 + R\cos(\theta_2)\delta\theta_2} \qquad [2.8]$$

Equation 2.8 may be rearranged to give $\delta\theta_2$ as

$$\delta\theta_2 = \frac{Z_1\delta\theta_1\left[\cos(\theta_1)\sin(\theta_3) - \sin(\theta_1)\cos(\theta_3)\right]}{R\left[\sin(\theta_2)\cos(\theta_3) - \cos(\theta_2)\sin(\theta_3)\right]} \qquad [2.9]$$

In evaluating Eq. 2.1 and eliminating $\delta\theta_2$ in favor of $\delta\theta_1$, the virtual work expression becomes

$$\begin{aligned}\delta W = T\delta\theta_1 &- F_{c_x}\left[Z_1\sin(\theta_1)\delta\theta_1 + Z_2\sin(\theta_2+\beta)\frac{Z_1\delta\theta_1\left[\cos(\theta_1)\sin(\theta_3)-\sin(\theta_1)\cos(\theta_3)\right]}{R\left[\sin(\theta_2)\cos(\theta_3)-\cos(\theta_2)\sin(\theta_3)\right]}\right] \\ &+ F_{c_y}\left[Z_1\cos(\theta_1)\delta\theta_1 + Z_2\cos(\theta_2+\beta)\frac{Z_1\delta\theta_1\left[\cos(\theta_1)\sin(\theta_3)-\sin(\theta_1)\cos(\theta_3)\right]}{R\left[\sin(\theta_2)\cos(\theta_3)-\cos(\theta_2)\sin(\theta_3)\right]}\right]\end{aligned} \qquad [2.10]$$

The applied force components, F_{cx} and F_{cy}, may be expressed with the force magnitude, F_c, and direction, θ_c, in the preceding expression to give

$$\begin{aligned}0 = T &- F_c\cos(\theta_c)\left[Z_1\sin(\theta_1) + Z_2\sin(\theta_2+\beta)\frac{Z_1}{R}\frac{\sin(\theta_3-\theta_1)}{\sin(\theta_2-\theta_3)}\right] \\ &+ F_c\sin(\theta_c)\left[Z_1\cos(\theta_1) + Z_2\cos(\theta_2+\beta)\frac{Z_1}{R}\frac{\sin(\theta_3-\theta_1)}{\sin(\theta_2-\theta_3)}\right]\end{aligned} \qquad [2.11]$$

Further simplification yields

$$0 = T + F_c\left[Z_1\sin(\theta_c-\theta_1) + Z_2\sin(\theta_c-(\theta_2+\beta))\frac{Z_1}{R}\frac{\sin(\theta_3-\theta_1)}{\sin(\theta_2-\theta_3)}\right] \qquad [2.12]$$

The bracketed term in Eq. 2.12, , should be of constant sign and opposite that of T throughout the motion of the mechanism; if it is not, the sense of T is inconsistent and the mechanism's path changes direction prior to reaching a 180° output of the crank.

To ensure closure of the mechanism in all positions and to establish an expression for the precision points, the following development is used. The precision points are the discrete points that the coupler's apex passes through; in this case, they are also the points that define the line of action of the actuation force. Figure 2.3 indicates the parameters used. A circuit around loop $Z_1 - Z - l'$ gives the real term:

$$l'\cos(\upsilon) = -Z_1\cos(\theta_1) + Z\cos(\theta) \qquad [2.13]$$

The imaginary term around the same loop is:

$$l'\sin(\upsilon) = -Z_1\sin(\theta_1) + Z\sin(\theta) \qquad [2.14]$$

The value for l' may now be found as follows;

$$l' = \sqrt{\left(Z\cos(\theta) - Z_1\cos(\theta_1)\right)^2 + \left(Z\sin(\theta) - Z_1\sin(\theta_1)\right)^2} \qquad [2.15]$$

The angular orientation of l' is given by:

$$\upsilon = \tan_2^{-1}\left(\frac{Z\sin(\theta) - Z_1\sin(\theta_1)}{Z\cos(\theta) - Z_1\cos(\theta_1)}\right) \qquad [2.16]$$

Link member R, determined from loop $l' - Z^*_1 - R$, is

$$R = \sqrt{\left(-l'\cos(\upsilon) + Z^*_1\cos(\theta_3)\right)^2 + \left(-l'\sin(\upsilon) + Z^*_1\sin(\theta_3)\right)^2} \qquad [2.17]$$

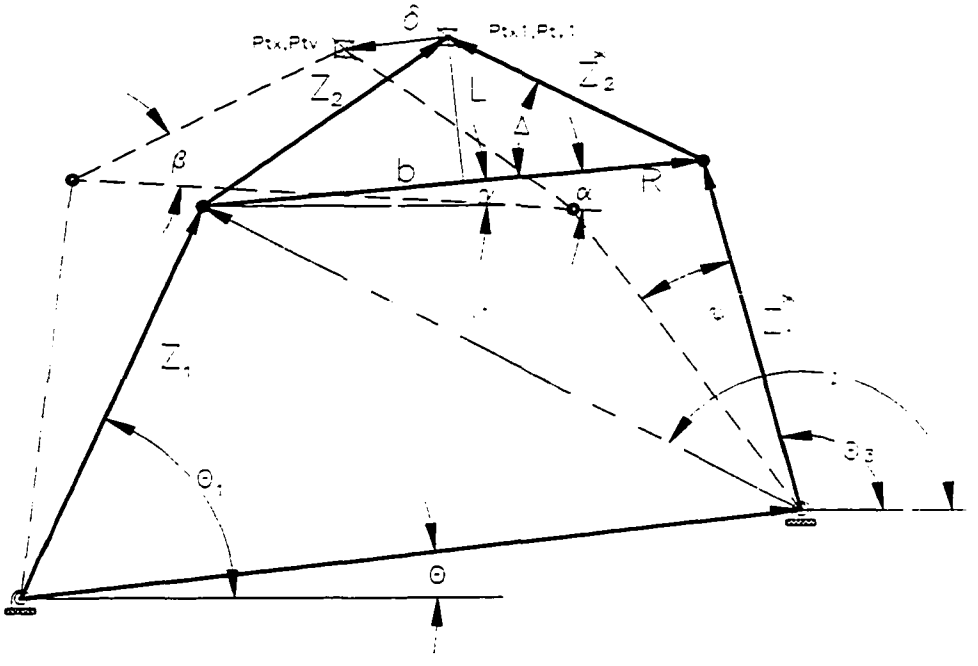

Figure 2.3: Parameters used in loop closure

The angular orientation of the coupler is determined from loop $l' - R - Z_1^*$; components of R are:

$$\begin{aligned} R_x &= -l'\cos(\upsilon) + Z^*_1 \cos(\theta_3) \\ R_y &= -l'\sin(\upsilon) + Z^*_1 \sin(\theta_3) \end{aligned} \qquad [2.18]$$

From which the angular orientation of R, γ, is

$$\gamma = \tan_2^{-1}\left(\frac{-l'\sin(\upsilon) + Z_1^* \sin(\theta_3)}{-l'\cos(\upsilon) + Z_1^* \cos(\theta_3)}\right) \qquad [2.19]$$

To determine the value of θ_3, the law of cosines and loop closure around $l' - Z^*_1 - R$ gives:

$$\theta_3 = \upsilon \pm \cos^{-1}\left(\frac{R^2 - Z_1^{*2} - l'^2}{2Z_1^* l'}\right) \qquad [2.20]$$

Now the x and y components of the precision points may be determined.

$$Pt_x = Z_1\cos(\theta_1+\phi)+b\cos(\gamma+\alpha)+L\cos\left(\gamma+\alpha+\frac{\pi}{2}\right)$$
$$Pt_y = Z_1\sin(\theta_1+\phi)+b\sin(\gamma+\alpha)+L\sin\left(\gamma+\alpha+\frac{\pi}{2}\right) \qquad [2.21]$$

The value of b gives the distance from the pin joining the crank and the coupler to the perpendicular joining the coupler apex, or precision point, and member R. L is the perpendicular's magnitude. Angles ϕ and α are the displacements of links Z_1 and R. respectively, from their initial angular orientations(see Figure 2.3).

The slope of the path, which ideally is also the slope of the line of action of the actuator force, given by θc, may now be expressed as a function of the precision points:

$$\theta_c = \tan_2^{-1}\left(\frac{Pt_y - Pt_{y_1}}{Pt_x - Pt_{x_1}}\right) \qquad [2.22]$$

Pt_{x_1} and Pt_{y_1}, are coordinates of the precision point in the mechanism's initial position. The measure of deviation from straightness is determined by comparing θ_c with θ_{c1}; the actuator orientation in a given position of the mechanism is compared with the orientation of the actuator in the mechanism's initial position. In the case of Simulated Annealing the value of θ_{c1} is determined by including it as an unknown parameter in the optimization of the mechanism. The value of θ_{c1} in the Genetic Algorithm is the average of the θ_c values determined by Eq. 2.22 for each position of the mechanism. The Generalized Reduced Gradient procedure finds values of the actuator orientation for four positions of the mechanism, i.e., $\theta_{c1}, \theta_{c2}, \theta_{c3}$, and θ_{c4} using equations of static equilibrium as constraints.

Also generated within the Generalized Reduced Gradient procedure are the values of the actuator's orientation, or ideal coupler point path as determined by Eq. 2.22. The unknown parameters, $\theta_{c1}, \theta_{c2}, \theta_{c3}$, and θ_{c4}, are compared with the values determined by Eq. 2.22. In each optimization method, if $|\theta_c - \theta_{c1}|$ is less than or equal to a predetermined allowable deviation, say 3°, the value of the link parameters making this relationship true are accepted as feasible.

The transmission angle of four bar mechanisms typically occurs between member R. of the coupler, and the follower, link Z^*_1. Ordinarily the input link is the crank, Z_1, and the force applied to the follower is transmitted through the coupler. The transmission angle occurs between the line of action of the force transmitted along member R and the line of action of the force acting on the follower. The torque produced on the follower is optimal when the transmission angle is 90°. If the transmission angle is not 90°, components of the force acting along the coupler are imposed on the pin joint and as a result may decrease the effectiveness with which the mechanism executes its motion. In this study, the input, or driving link is the coupler and the driven link is the crank. The three forces acting on coupler are

1) the actuation force, F_c, at an orientation of θ_c
2) the force acting on the pin joining the coupler and follower, F_{43}, at an orientation of θ_3. The point at which the lines of action of F_c and F_{43} intersect is the *concurrency point.*

3) the force tending to drive the crank, F_{23}, whose line of action is defined by a line with one end point being the concurrency point and the second endpoint occurring at the pin joint of the crank and coupler.

The transmission angle in this case is located between the line of action of the force tending to drive the crank, F_{23}, and the crank itself. Again, the nearer to 90° the transmission angle is, the greater the output torque on the crank.

Figure 2.4 indicates the relationship between the applied actuation force at the coupler point and the transmission angle. The analytical expression for the transmission angle may be determined as noted below. From the law of sines:

$$A = \frac{Z_2^{*} \sin(\gamma + \pi - \Delta - \theta_3)}{\sin(\theta_3 - \theta_c)} \qquad [2.23]$$

A's magnitude defines the distance between the coupler's apex and the point of concurrency for the actuation force and the force acting along the follower.

Now the direction of the force occurring at the pin joint of the crank, member Z_1, and member R of the coupler is given by:

$$\lambda = \tan_2^{-1}\left(\frac{Z_2 \sin(\gamma + \beta) + A\sin(\theta_c)}{Z_2 \cos(\gamma + \beta) + A\cos(\theta_c)}\right) \qquad [2.24]$$

The transmission angle is thus:

$$\mu_T = \lambda + \pi - \theta_1 \qquad [2.25]$$

Optimal synthesis of the mechanism in this study requires an objective function

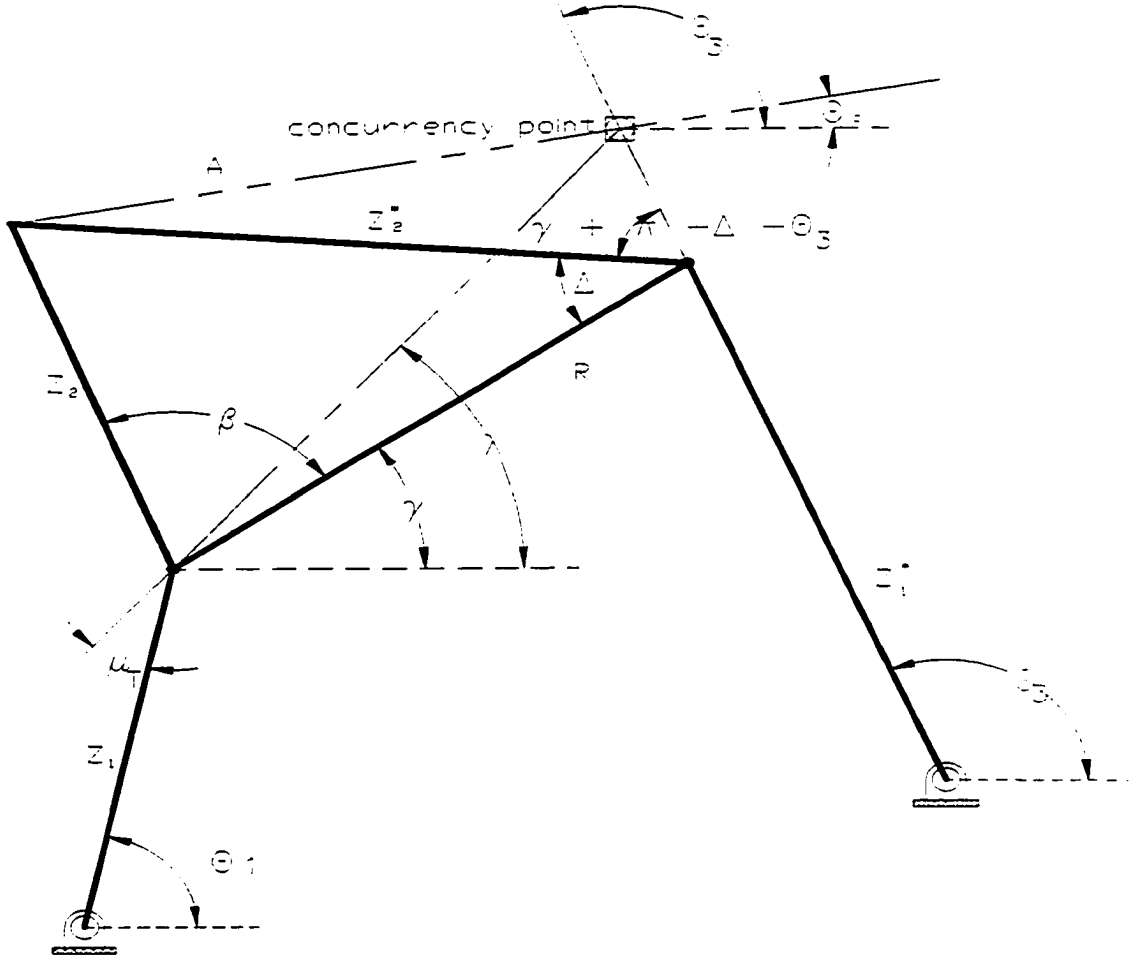

Figure 2.4: Transmission angle's determination

combining the goals of a straight path, transmission angle near 90°, and penalties associated with a change in the coupler point path direction prior to a crank output of at least 180°. In addition to an objective function, constraint equations are also required.

The constraint equations used for the Generalized Reduced Gradient method are divided into two categories:

1. Equations of static equilibrium
2. Loop closure around the mechanism

The equations of static equilibrium include summation of forces in the x and y directions and moment equations. Although the virtual work approach would have been equally acceptable for use with the generalized reduced gradient method, the course of the research began with use of the equations of static equilibrium. It was observed that the non-gradient based optimization methods, simulated annealing and genetic algorithms specifically, did not perform well with the domain defined in terms of Equations 2.26 - 2.28.

Figure 2.5 represents the mechanism in terms of its static analysis. Link Z_1 is treated as a free body, and the coupler combined with link Z^*_1 is treated as a second free body. For each position of the mechanism, the following equations are included in the static analysis.

Link Z_1 - Body 2

$$\begin{aligned} F_{12_x} + F_{32_x} &= 0 \\ F_{12_y} + F_{32_y} &= 0 \\ F_{12_y} Z_1 \cos(\theta_1) + F_{12_x} Z_1 \sin(\theta_1) + T &= 0 \end{aligned} \qquad [2.26]$$

Coupler Link and Link Z^*_1 - Body 3 and Body 4

$$\begin{aligned} F_{23_x} + F_c \cos(\theta_c) + F_{14} \cos(\theta_3) &= 0 \\ F_{23_y} + F_c \sin(\theta_c) + F_{14} \sin(\theta_3) &= 0 \\ F_{14} R \sin(\theta_3 - \gamma) + F_c\left(b \sin(\theta_c - \gamma) - L \cos(\theta_c - \gamma)\right) &= 0 \end{aligned} \qquad [2.27]$$

Composite Body - Bodies 2, 3 and 4
Mechanism as a Free Body

$$\begin{aligned} F_{12_x} + F_c \cos(\theta_c) + F_{14} \cos(\theta_3) &= 0 \\ F_{12_y} + F_c \sin(\theta_c) + F_{14_j} \sin(\theta_3) &= 0 \end{aligned} \qquad [2.28]$$

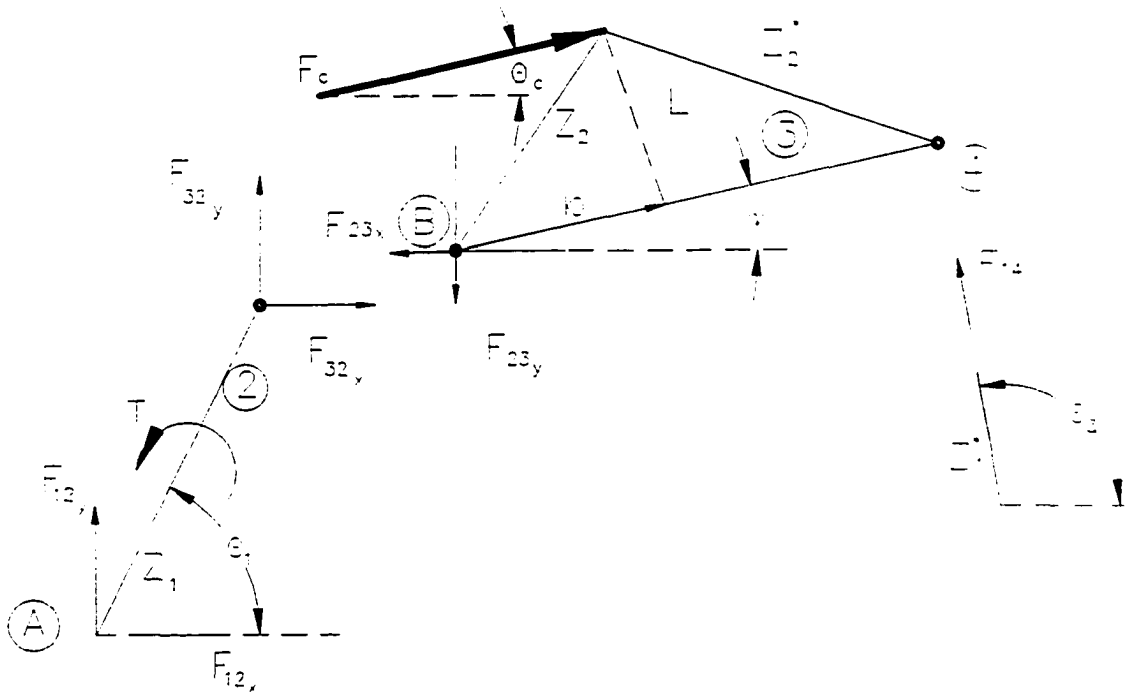

Figure 2.5: Static equilibrium analysis

Equations 2.28 contain constraints used in the Generalized Reduced Gradient method of Chapter III. Other constraints include the moment equations of Eqs. 2.26 and 2.27. (Simulated Annealing and Genetic Algorithms implement exterior penalty functions which examine the signs on virtual work terms. Further explanation and development follow in Chapters IV and V). To guarantee closure of the mechanism, the real and imaginary parts of a loop around the four bar mechanism are included within the constraint set (see Figure 2.1):

$$\begin{aligned} Z_1\cos(\theta_1)+R\cos(\gamma) &= Z\cos(\theta)+Z_1^*\cos(\theta_3) \\ Z_1\sin(\theta_1)+R\sin(\gamma) &= Z\sin(\theta)+Z_1^*\sin(\theta_3) \end{aligned} \qquad [2.29]$$

The transmission angle, in terms of forces, and as an alternative to Eq. 2.25, may be expressed as

$$\mu_T = \tan_2^{-1}\left(\frac{-F_{12_y}}{-F_{12_x}}\right) + \pi - \theta_1 \qquad [2.30]$$

The Generalized Reduced Gradient method is used to minimize a composite objective function comprised of the mechanism's approximate straight line behavior, minimization of the transmission angle deviation from 90° and minimization of the perpendicular distance for twenty discrete points of an initial mechanism's coupler path from an optimal mechanism's path. The details of the Generalized Reduced Gradient method and its implementation for solution to this synthesis problem follow in Chapter III.

III. GENERALIZED REDUCED GRADIENT

The method of optimization selected for solution to the nonlinear, constrained problem of path generation with prescribed timing is known as the Generalized Reduced Gradient method, or GRG. The following terms will be used interchangeably throughout this discussion: *state, basic and dependent* variables are the same; *decision, non-basic and independent variables* are the same.

Gabriele and Ragsdell [16] present the flow chart of Fig. 3.1 with their discussion of GRG. The generalized reduced gradient method is capable of handling nonlinear constraints, both equality and inequality constraints, as well as nonlinear objective functions. Abadie [17] first detailed a methodology in which nonlinearities in the objective function and the constraint equations could be accommodated with a generalization of Wolfe's [18] method. These approaches essentially render the problem unconstrained by using the constraint equations to solve for as many variables as there are equations in terms of the remaining unknowns.

The variables of the problem are divided into two sets: the first set is comprised of the *decision* or in some literature, the *non-basic* variables, and the second set is made up of the *state*, or *basic* variables. The decision variables are *independent* and the state variables are *dependent*. Assuming there are L constraint equations and N total variables, the idea is to solve the constraint equations for L of the N total variables, leaving N-L variables to be determined. Once the state variables are found in this way, the problem becomes

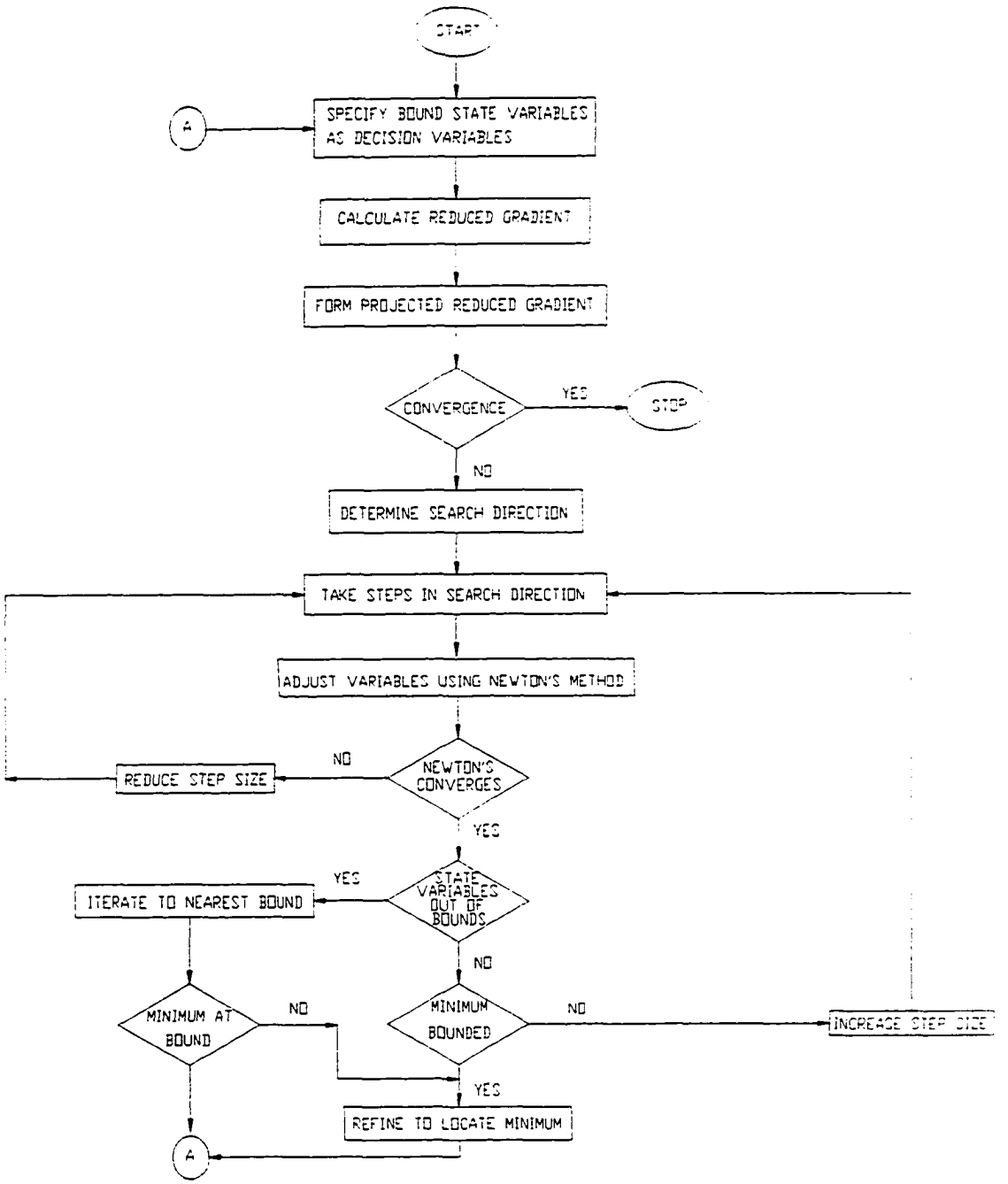

Figure 3.1: Flowchart for Generalized Reduced Gradient method

unconstrained. Following determination of the decision variables, they are substituted into expressions for the state variables from the simultaneous solution of the nonlinear constraints. Although closed form expressions for the basic variables are sometimes possible with the advent of symbolic manipulators such as Maple© and Mathematica©, they are still difficult to acquire if more than a few variables appear in a highly nonlinear context. Consequently, the following discussion will outline the handling of nonlinear constraints using the numerical approach of dividing variables into the two groups used by the GRG method.

Using the same terminology as that used by Gabriele and Ragsdell, the basic variables are represented by $\bar{y}$ and the non-basic, or independent variables, are represented as $\bar{z}$. The M equality constraint equations are represented by the vector ψ. The total number of unknowns is N so that the number of non-basic variables is N-M=Q. The authors provide the following guidelines for selecting state and decision variables:

1. Selection of state (basic) variables should be made such that the matrix $\left[\frac{\partial \psi}{\partial \bar{y}}\right]$ is non-singular
2. State variables may be arbitrarily adjusted to maintain feasibility. For example, if any variable becomes equal to an upper or lower bound, that variable should become a decision (independent, non-basic) variable.
3. All *slack* variables should be designated as state variables.

Slack variables are introduced into the set of unknowns in order to change inequality constraints into equality constraints. For example, consider the inequality $x_1 \leq 5$. By introducing a new variable, s_1, and adding it to x_1, the inequality equation becomes an equality equation.

$$x_1 + s_1 = 5 \qquad [3.0]$$

s_1 may be thought of as the "slack" between the left and right hand sides of the equation. If the inequality is $\geq$, then a *surplus* variable is subtracted from the left hand side, indicating the amount by which the left hand side exceeds the right hand side. Both slack and surplus variables are non-negative.

The *reduced gradient*, g_r, is the rate of change of the objective function with respect to the decision variables, the state variables being adjusted to maintain feasibility [16]. The expression for the reduced gradient is:

$$[g_r(\bar{x})]^T = [g_r(\bar{z})]^T - [g_r(\bar{y})]^T \left[\frac{\partial \psi^{-1}}{\partial \bar{y}}\right]\left[\frac{\partial \psi}{\partial \bar{z}}\right] \qquad [3.1]$$

where, $\left[\frac{\partial \psi^{-1}}{\partial \bar{y}}\right]$ is the inverse of the matrix formed by determining the Jacobian of the constraint equations with respect to the basic variables. The matrix $\left[\frac{\partial \psi}{\partial \bar{z}}\right]$ is the Jacobian of the constraint equations with respect to the non-basic variables.

The numerical implementation of GRG is accomplished by first calculating the matrix, $\left[\frac{\partial \psi^{-1}}{\partial \bar{y}}\frac{\partial \psi}{\partial \bar{z}}\right]$. Next the objective function, F, is calculated using the current values of $\bar{y}$ and $\bar{z}$. The ith element of the decision vector is perturbed by some small amount, δ.

$$\bar{z} = \bar{z}_i + \delta \qquad [3.2]$$

The M state variables are changed simultaneously, using

$$\bar{y} = \bar{y}_m - \left[\delta \frac{\partial \psi^{-1}}{\partial \bar{y}} \frac{\partial \psi}{\partial \bar{z}} \right]_m \quad m = 1,2,3...M \qquad [3.3]$$

where m is the column number in the matrix $\left[\frac{\partial \psi^{-1}}{\partial \bar{y}} \frac{\partial \psi}{\partial \bar{z}} \right]$.

The ith element of the reduced gradient is

$$g_r(\bar{x})_i = \frac{F\left(\bar{z}_1, \bar{z}_2, ..., \bar{z}_i, ..., \bar{z}_Q, \bar{y}_1, \bar{y}_2, ..., \bar{y}_M\right) - F_1}{\delta} \qquad [3.4]$$

where F_1 is the objective function at the current point, $(\bar{z}, \bar{y})$ and $\bar{x}$ is the set of design variables, or unknowns.

The projected reduced gradient is established by checking to see if there are any out of bounds occurrences of the decision variables. If the ith element of the reduced gradient is negative and the non-basic variable associated with that index is equal to the upper limit set for that variable, that element of the reduced gradient is set to zero. If the ith element of the reduced gradient is positive and the associated non-basic variable is equal to the lower limit set for the variable, that element of the reduced gradient is set to zero. Otherwise all elements of the reduced gradient retain their values. Convergence is checked by comparing the L_2 norm of the projected reduced gradient with some previously specified criterion, where

$$\|L\|_2 = \sqrt{\sum_{i=1}^{N} g_r\left(\bar{x}_i\right)^2} \qquad [3.5]$$

If the criterion is met, a constrained relative minimum has been reached, otherwise the algorithm proceeds.

Continuing the algorithm requires that a search direction be established. Any gradient based unconstrained technique may be used such as the conjugate gradient method [16]. The *projected reduced gradient* is used in this search.

A search direction for the state variables is also established using the following expression for the differential displacements of the state variables:

$$d\bar{y} = -\left[\frac{\partial \psi^{-1}}{\partial \bar{y}} \frac{\partial \psi}{\partial \bar{z}}\right]\{d\bar{z}\} \qquad [3.6]$$

Gabriele and Ragsdell [16] point out that expression 3.6 yields only an approximation for the search direction of the state (basic) variables and that further adjustment is required when the constraints are nonlinear. A line search is performed to locate a local minimum along the vector P(z). (P(z) is a line in the N-M space of the decision variables containing a local minimum of the objective function; it is given by the negative of the reduced gradient, Eq. 3.1). This is accomplished by first identifying a bracket within which a minimum is known to occur. Next the bracket is continuously narrowed by some scheme until an acceptable tolerance is reached. P(z) is used to update the decision variables, and P(y) is used to update the state variables. P(y) is given by Equation 3.6. The updates occur as noted below.

$$\bar{z}_i^{k+1} = \begin{Bmatrix} b_i \text{ if } \bar{z}_i^k + \alpha P(\bar{z}_i) \geq b_i \\ a_i \text{ if } \bar{z}_i^k + \alpha P(\bar{z}_i) \leq a_i \\ \bar{z}_i^k + \alpha P(\bar{z}_i) \end{Bmatrix} \qquad [3.7]$$

$$\bar{y}^{k+1}{}_m = \left\{ \bar{y}_m{}^k + \alpha P(\bar{y}_m) \right\} \qquad [3.8]$$

The parameter, α, in Eqs. 3.7 and 3.8 is a step length parameter and a_i and b_i are the lower and upper bounds, respectively, on the decision variables. It is unlikely that the new point will be feasible and, as a result, a method for the solution of nonlinear equations such as Newton's method is required to adjust the state variables, $\bar{y}^{k+1}$, while holding the decision variables, $\bar{z}_i^{k+1}$, constant. Now the bounds on the variables are checked: if all of the state variables are within bounds, the objective function is evaluated and the procedure continues. If the point $[\bar{z}_i^{k+1}, \bar{y}^{k+1}]$ is not feasible, linear interpolation is performed until the nearest bound becomes active. If convergence for a minimum fails to occur, the step length parameter, α, is reduced and a new trial point established.

If a state variable is at its upper or lower bound following the preceding activities, it must be exchanged with a decision variable before the next iteration. The decision variable selected to exchange with the bounded state variable, y_s, is determined by *maximizing* the following expression [16]:

$$\min\left\{ |D_{si}|(b_i - z_i), |D_{si}|(z_i - a_i) \right\} \; i = 1,2,\ldots,Q$$
$$|D_{si}| = \text{element of } \left[\frac{\partial \psi^{-1}}{\partial \bar{y}} \frac{\partial \psi}{\partial \bar{z}} \right] \qquad [3.9]$$

where,

b_i is the upper bound on the ith decision variable
a_i is the lower bound on the ith decision variable
Q is the total number of decision variables
si is the sth row, ith column in the matrix $\left[\frac{\partial \psi^{-1}}{\partial \bar{y}} \frac{\partial \psi}{\partial \bar{z}}\right]$

Vanderplaats [19] describes a scheme for picking the dependent variables and avoiding singularities by performing Gaussian elimination on the Jacobian of the constraints with respect to all variables, that is by solving the system [Q]{x} = [I] where [I] is the identity matrix and when the elimination is complete, the inverse of the Jacobian of the state variables remains, $\left[\frac{\partial \psi^{-1}}{\partial \bar{y}}\right]$. The matrix [Q] is the Jacobian of the constraint equations with respect to *all* unknowns and {x} is the vector of unknowns.

Gabriele and Ragsdell continue their discussion of the Generalized Reduced Gradient method with recommendations that reduce computation time while retaining numerical integrity of the procedure. They also present findings of their comparative analysis of GRG with other optimization procedures.

Loop closure equations and equations of static equilibrium comprise the constraint equations of the optimization problem to be solved by the Generalized Reduced Gradient method. The four position problem results in 24 constraint equations. Loop closure equations are included in the constraint set to ensure that the mechanism remains closed in all positions and statics equations to: a) monitor the actuator force's direction and b) to minimize the deviation from 90° of the transmission angles. The angular displacement of

the crank in its jth position is represented by ϕ_j. The follower's angular displacement, in the jth position, is ψ_j and the angular displacement of the coupler is α_j.

$$\begin{aligned} Z_1\cos(\theta_1)+R\cos(\gamma)-Z^*_1\cos(\theta_3)-Z\cos(\theta)=0 \\ Z_1\sin(\theta_1)+R\sin(\gamma)-Z^*_1\sin(\theta_3)-Z\sin(\theta)=0 \end{aligned} \qquad [3.10]$$

$$\begin{aligned} Z_1\cos(\theta_1+\phi_j)+R\cos(\gamma+\alpha_j)-Z^*_1\cos(\theta_3+\psi_j)-Z\cos(\theta)=0 \\ Z_1\sin(\theta_1+\phi_j)+R\sin(\gamma+\alpha_j)-Z^*_1\sin(\theta_3+\psi_j)-Z\sin(\theta)=0 \end{aligned} \qquad j=2,4 \quad [3.11]$$

$$\begin{aligned} F_{12_{x1}}+F_{c_1}\cos(\theta_{c_1})+F_{14_1}\cos(\theta_3)=0 \\ F_{12_{y1}}+F_{c_1}\sin(\theta_{c_1})+F_{14_1}\sin(\theta_3)=0 \end{aligned} \qquad [3.12]$$

$$\begin{aligned} F_{12_{xj}}+F_{cj}\cos(\theta_{c_j})+F_{14_j}\cos(\theta_3+\psi_j)=0 \\ F_{12_{yj}}+F_{cj}\sin(\theta_{c_j})+F_{14_j}\sin(\theta_3+\psi_j)=0 \end{aligned} \qquad j=2,4 \quad [3.13]$$

$$\begin{aligned} -Z_1\cos(\theta_1)F_{12_{1_y}}+Z_1\sin(\theta_1)F_{12_{1_x}}+T_1=0 \\ F_{14_1}R\sin(\theta_3-\gamma)+F_{c_1}\left(b\sin(\theta_{c_1}-\gamma)-L\cos(\theta_{c_1}-\gamma)\right)=0 \end{aligned} \qquad [3.14]$$

$$\begin{aligned} -Z_1\cos(\theta_1+\phi_j)F_{12_{j_y}}+Z_1\sin(\theta_1+\phi_j)F_{12_{j_x}}+T_j=0 \\ F_{14}R\sin(\theta_3+\psi_j-(\gamma+\alpha_j))+ \\ F_{c_j}\left(b\sin(\theta_{c_j}-(\gamma+\alpha_j))-L\cos(\theta_{c_j}-(\gamma+\alpha_j))\right)=0 \end{aligned} \qquad j=2,4 \quad [3.15]$$

Inspection of the 24 equations represented by Eqs. 3.10 - 3.15, indicates that some choices of variables as "free" choices in the solution of this set of equations will result in numerical problems such as a singular Jacobian or linear dependence and rank deficiency of the coefficient matrix, where $[A]\{x\} = \{b\}$ is the system of equations and [A] is the coefficient matrix. Identification of such free choices is much like a decision prescription for the determination of state and decision variables. Table 3.1 indicates which of the 43 unknowns are associated with each position of the mechanism.

Table 3.1: Unknowns in Eqs. 3.10 - 3.15

Position	**Kinematic Unknowns**	**Static Equilibrium Unknowns**
1	$Z_1,\theta_1,Z^*_1,\theta_3,Z,\theta,R,\gamma,L,b$	$F_{12x1},F_{12y1},F_{c1},\ \theta_{c1},F_{141},T_1$
2	$\alpha_2,\ \phi_2,\ \psi_2$	$F_{12x2},F_{12y2},F_{c2},\ \theta_{c2},F_{142},T_2$
3	$\alpha_3,\ \phi_3,\ \psi_3$	$F_{12x3},F_{12y3},F_{c3},\ \theta_{c1},F_{143},T_3$
4	$\alpha_4,\ \phi_4,\ \psi_4$	$F_{12x4},F_{12y4},F_{c4},\ \theta_{c4},F_{144},T_4$
total	19	24

Since the problem involves the solution of a path generation problem with prescribed timing, the angular displacements of the crank, ϕ_j, are known. Also, the output torque, T_j, in each position can be specified. The number of unknowns is now reduced to 36, requiring specification of 12 parameters in order to solve the 24 nonlinear equations represented by Equations 3.10 - 3.15. The free choices may be selected from among the remaining

parameters. However, Equations 3.12 and 3.13 contain only the kinematic parameters, θ_2, ψ_2, ψ_3, and ψ_4. This leads to the observation that θ_3 must be considered an unknown variable and not a free choice; the Jacobian of the constraints becomes singular when θ_3 is not among the unknowns. On the other hand, parameters Z, θ, R, γ, L, b, *should be among the free choices* since they do not appear in Equations 3.12 and 3.13. Further examination of the equations indicates that if α_j, and ψ_j are selected as free choices, linear dependence in Equations 3.10 and 3.11 will occur and the coefficient matrix of the nonlinear problem will be rank degenerate. If any other kinematic parameter is selected from those remaining, Z_1, θ_1, Z^*_1, the mechanism is fully defined. The equations represented by Eq. 3.10 could be used to solve for the remaining two unknowns; consequently, these parameters should appear in the set of unknowns. Table 3.2 summarizes this discussion and delineates the parameters remaining from which the free choices and unknowns may be formed.

Table 3.2: Free choices for solution to Eq. 3.10- Eq. 3.15

free choices	**unknowns**	**remaining free choices**
Z,θ,R,γ,L,b	α_j, ψ_j, θ_3, Z_1,θ_1,Z^*_1	F_{12xj},F_{12yj},F_{cj}, θ_{cj},F_{14j}

Six more parameters must be selected to solve the kinematics problem with simultaneous solution of the equations of static equilibrium. Since the mechanism will be constrained to follow approximate straight line motion dictated by the direction of the actuator, θ_{cj}, the choice of this parameter represents four free choices, since $\theta_{c1}=\theta_{c2}=\theta_{c3}=\theta_{c4}$. Now 10

parameters (Z_1, θ_1, R, γ, L,b, θ_{ci}) have been selected as free choices: two more must be selected from among the remaining parameters, all of which are forces.

The point of the preceding discussion is to note that the appropriateness of free choices is not always apparent and that the parameters included in the set of free choices are, in some cases, determined by the nature of the problem. The fact that there are more variables than equations suggests that an optimization approach to the problem may be possible.

The procedure used to accomplish the numerical tasks of the path generation problem with prescribed timing and imposed actuating force was accomplished by the GRG routine supplied with Microsoft Excel© ver. 5.0 [20,21]. The spreadsheet solver requires the constraint equations, (both inequality as well as equality), an objective function, and identification of the cells containing the variables sought. Any determination of state and decision variables is transparent to the user but the sensitivity and limits reports generated by Excel's solver are indicative of the final configuration of the variables.

The equations given previously as Equations 3.10 - 3.15 were entered into Excel© as the constraint equations. The objective function used is given below:

$$F = w_1 \sum_{j=2}^{4} \left| \tan_2^{-1}\left(\frac{P_{y_j} - P_{y_1}}{P_{x_j} - P_{x_1}}\right) - \tan_2^{-1}\left(\frac{P_{y_{j-1}} - P_{y_1}}{P_{x_{j-1}} - P_{x_1}}\right) \right|$$

$$+ w_2 \sum_{j=1}^{4} \left| \frac{\pi}{2} - \left[\tan_2^{-1}\left(\frac{-F_{12_{y_j}}}{-F_{12_{x_j}}}\right) + \pi - (\theta_1 + \phi_j) \right] \right| + w_3 \sum_{k=1}^{20} \Delta h_k \qquad [3.16]$$

$$= w_1 \sum_{j=2}^{4} \left| \theta_{c_j} - \theta_{c_{j-1}} \right| + w_2 \sum_{j=1}^{4} \left| \frac{\pi}{2} - \mu_{T_j} \right| + w_3 \sum_{k=1}^{20} \Delta h_k$$

P_{yj} = y coordinate of precision point in jth position
P_{xj} = x coordinate of precision point in jth position
F_{12xj} = horizontal component of force exerted on the crank by the ground in the j^{th} position
F_{12yj} = vertical component of force exerted on the crank by the ground in the j^{th} position
θ_1 = crank (link Z_1) angle
ϕ_j = angular displacement of the crank from an initial orientation of θ_1
μ_{Tj} = transmission angle in the jth position
Δh = perpendicular distance between ideal coupler point path and start mechanism's coupler point path
θ_{cj} = coupler point path angle in the jth position

$w_1 \sum_{j=2}^{3} \left| \theta_{c_j} - \theta_{c_{j-1}} \right|$ = objective of minimizing the difference between the initial orientation of the actuator, θ_{c_1}, and subsequent orientations, θ_{c_j}.

$w_2 \sum_{j=1}^{4} \left| \frac{\pi}{2} - \mu_{T_j} \right|$ = objective of minimizing the difference between the transmission angle at position j and 90°.

$w_3 \sum_{k=1}^{20} \Delta h_k$ = objective of minimizing the perpendicular distance between an ideal mechanism's path and a mechanism with parameters satisfying the constraint equations

The weights used for the components of the objective function were $w_1 = 15$, $w_2 = 5$, and $w_3 = 2$. The weight values were determined through trial and error.

The j^{th} element of $\sum_{j=1}^{4} \left| \frac{\pi}{2} - \mu_{T_j} \right|$ was set to zero if the transmission angle fell between 140° and 50°. The term Δh_k is the perpendicular distance between the precision points of the start mechanism and the precision points of the theoretically ideal mechanism. The slope of the ideal coupler point path was taken to be the slope of the line joining the first and last precision points of the start mechanism's coupler point path. Since Generalized Reduced Gradient requires active constraints at the start of the optimization, a feasible solution was

identified by first solving Eqs. 3-10 through 3.15 in addition to an equation that includes a loop relating the left dyad of the mechanism and the actuator's ground point. The equation. broken into its component parts follows. W_x is the horizontal distance from the left dyad's ground pivot to the actuator ground and Δy is the vertical distance from the left dyad's ground pivot to the actuator ground. D_j is distance from the ground of the actuator to the coupler point, in the jth position of the mechanism.

$$\begin{aligned} D_j \sin(\theta_{c_j}) &= Z_1 \sin(\theta_1 + \phi_j) + Z_2 \sin(\gamma + \beta + \alpha_j) - \Delta y \\ D_j \cos(\theta_{c_j}) &= Z_1 \cos(\theta_1 + \phi_j) + Z_2 \cos(\gamma + \beta + \alpha_j) - W_x \end{aligned} \qquad [3.17]$$

In dividing the vertical component of Eq. 3.17 by the horizontal component, an expression for the arc tangent of θ_{cj} is established. The value of the actuator's angular orientation is now expressed as

$$\theta_{c_j} = \tan_2^{-1}\left(\frac{Z_1 \sin(\theta_1 + \phi_j) + Z_2 \sin(\gamma + \beta + \alpha_j) - \Delta y}{Z_1 \cos(\theta_1 + \phi_j) + Z_2 \cos(\gamma + \beta + \alpha_j) - W_x} \right) \qquad [3.18]$$

Figure 3.2 indicates the relationship of the mechanism's left dyad and a loop including the actuator.

Twenty (20) discrete points of the mechanisms were used for minimization of the perpendicular distances between the coupler paths. The points were taken at 9° intervals over the full range of motion of the crank. Figure 3.3 shows the relationship between these precision points and Eqs. 3.18. Table 3.3 indicates the angular displacements of the crank and Table 3.4 summarizes the parameters found by the GRG solver in Excel© and Figure 3.4

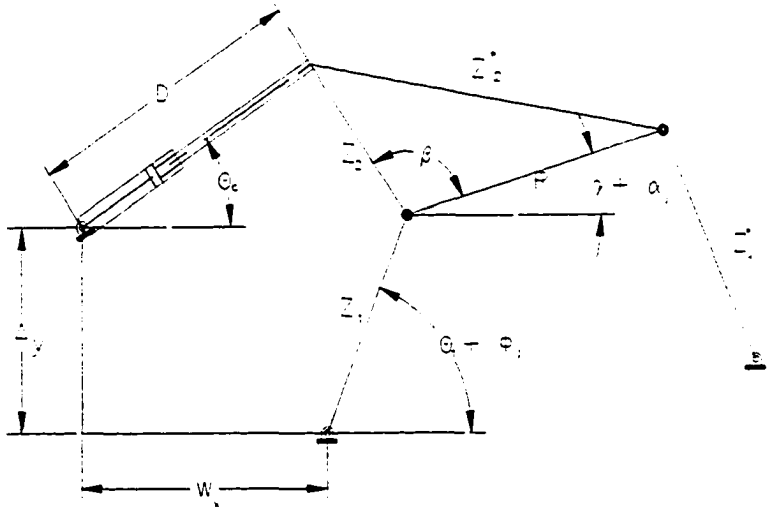

Figure 3.2: Actuator loop and left dyad

depicts the mechanism in its initial position. Also noted in Figure 3.4 are the four precision points relative and their associated crank displacements. Figures 3.5 and 3.6 depict the variation in actuator angle with crank angle and the variation of the transmission angle with crank angle respectively. Figure 3.7 compares the initial guess mechanism's coupler path, which represent a feasible solution to the constraint equations, with the optimized mechanism's coupler path. The optimized path is nearly coincident with the ideal path.

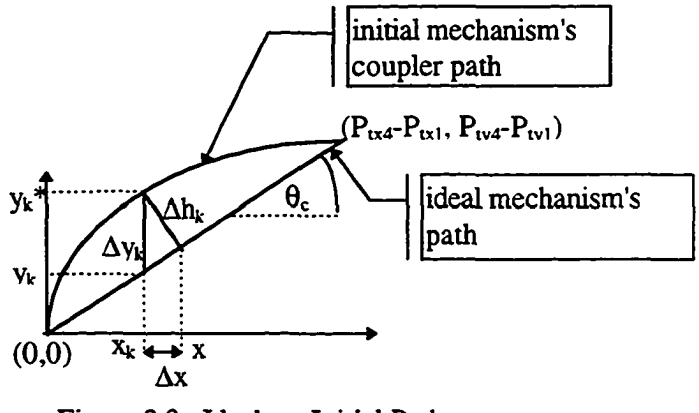

Figure 3.3: Ideal vs. Initial Path

$$
\begin{aligned}
&\Delta y_k = y_k^{\bullet} - y_k \\
&y_k^{\bullet} = Z_1 \sin(\theta_1 + \phi_k) + b\sin(\gamma + \alpha_k) + L\cos(\gamma + \alpha_k) - \left[Z_1 \sin(\theta_1) + b\sin(\gamma) + L\cos(\gamma)\right] \\
&y_k = \tan(\theta_c) * x_k \\
&\Delta x_k = \Delta y_k \cos(\frac{\pi}{2} - \theta_c) \\
&x_k^{\bullet} = Z_1 \cos(\theta_1 + \phi_k) + b\cos(\gamma + \alpha_k) - L\sin(\gamma + \alpha_k) - \left[Z_1 \cos(\theta_1) + b\cos(\gamma) - L\sin(\gamma)\right] \\
&\Delta h_k = \Delta x_k \tan\left(\frac{\pi}{2} - \theta_c\right) \\
&\theta_c = \tan_2^{-1}\left(\frac{P_{t_{y4}} - P_{t_{y1}}}{P_{t_{x4}} - P_{t_{x1}}}\right)
\end{aligned}
\qquad [3.19]
$$

The index k varies in Eq. 3.19 from 1 to 20, the number of discrete points taken from the start mechanism's path.

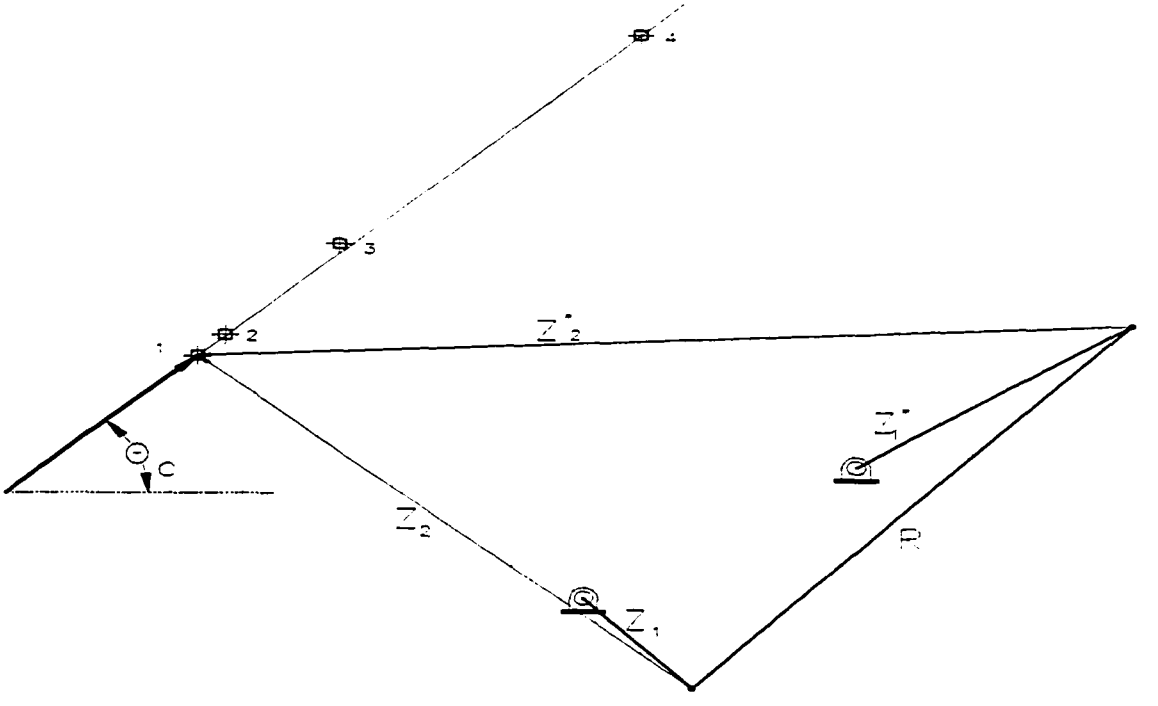

Figure 3.4: GRG mechanism in its initial position

Table 3.3: Crank angular displacements

Position, j	**Displacement ϕ_j, (Degrees)**
2	-5°
3	-60°
4	-181°

Table 3.4: GRG solution to synthesis problem

Parameter	**Value (inches)**	**VALUE / z_1 (inches)**
Z_1	11.89	1.00
θ_1	314.7°	
Z_1*	22.19	1.87
θ_3	45.1°	
R	49.55	4.17
γ	52.2°	
Z	27.52	2.31
θ	32.9°	
θc	48.3°	

The sensitivity analysis produced by Excel© is useful for predicting the effect of a small change in a constraint's right hand side on the value of the objective function. Since the GRG solver of Excel© provides the reduced gradient elements as well as the Lagrangian multipliers, the effect of making a small change in the right hand side of the constraints may

be determined. The Lagrangian multipliers give some insight into the effect that the changing of a particular constraint would have on the objective function. It can be shown [22], that a change in the objective function due to small changes in a constraint can be predicted using the values of the Lagrangian multipliers at the optimum, according to

$$\Delta F = F(b_i, e_j) - F(0,0) = -v^*_i b_i - u^*_j e_j \qquad [3.20]$$

where b_i is a value near to zero of an equality constraint

e_j is a value near to zero of an inequality constraint

F(0,0) is the value of the objective function at the optimum

$F(b_i,e_j)$ is the value of the objective with perturbations of b_i and e_j

v^*_i is the Lagrangian multiplier of the ith equality constraint

u^*_j is the Lagrangian multiplier of the jth inequality constraint

The constraints for the prescribed timing path generation problem were all equality constraints so u_j^* in Eq. 3.20 are 0. The largest Lagrangian multiplier, whose value was 66.1, was determined to be associated with the summation of the horizontal components of forces in the first position; the *magnitude* of the second largest Lagrangian multiplier was 31.4, associated with the summation of vertical components of the forces in the first position. Common to both constraints are the parameters F_{c1}, θ_{c1}, and θ_3. Equation 3.20 suggests that the objective function's value would tend to increase with small positive changes in the right hand sides of the constraint associated with the multiplier -31.4. The objective function would decrease with a small change in the right hand side of the constraint associated with the multiplier 66.1.

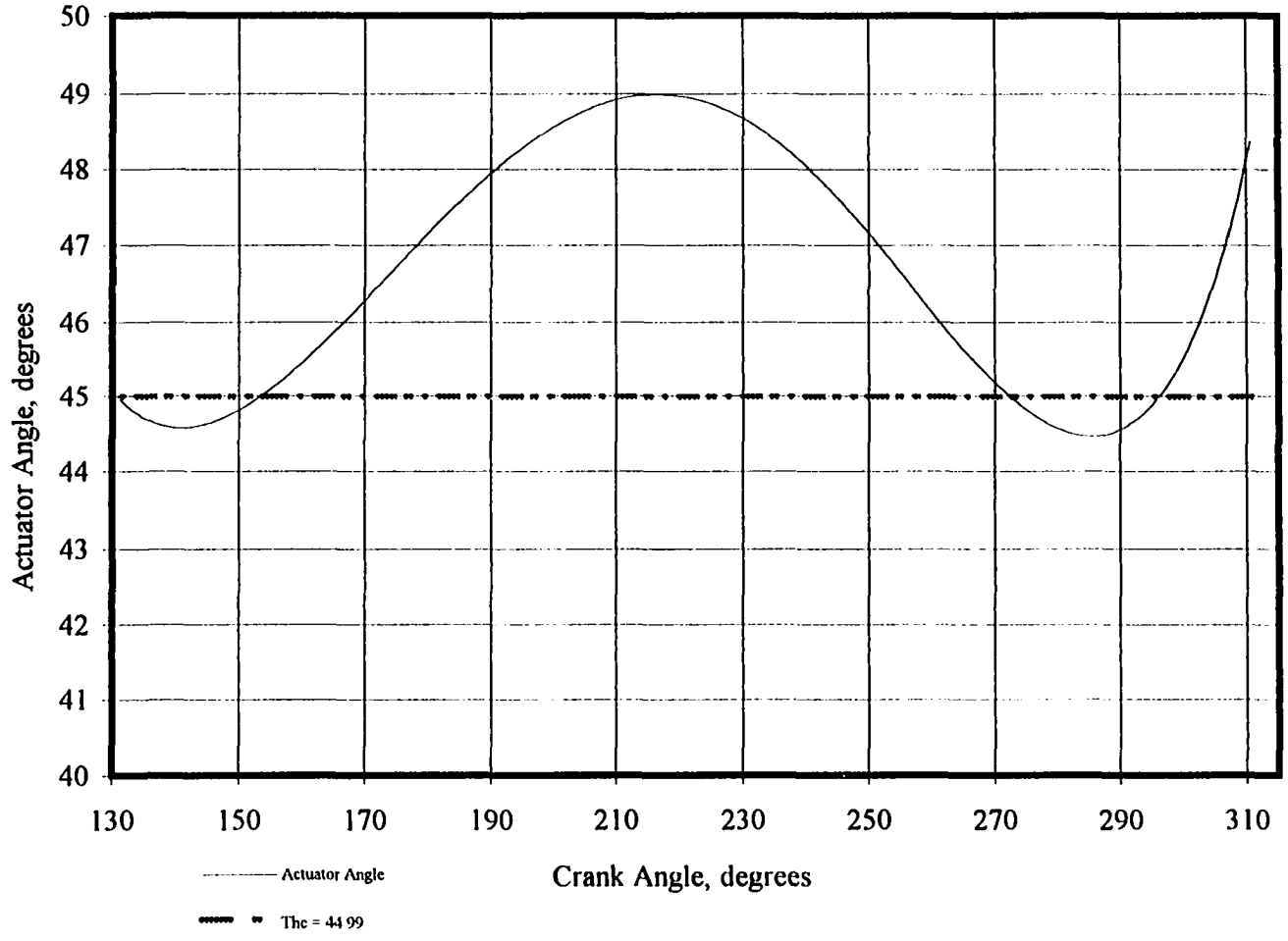

Figure 3.5: Variation in actuator angle with changing crank angle

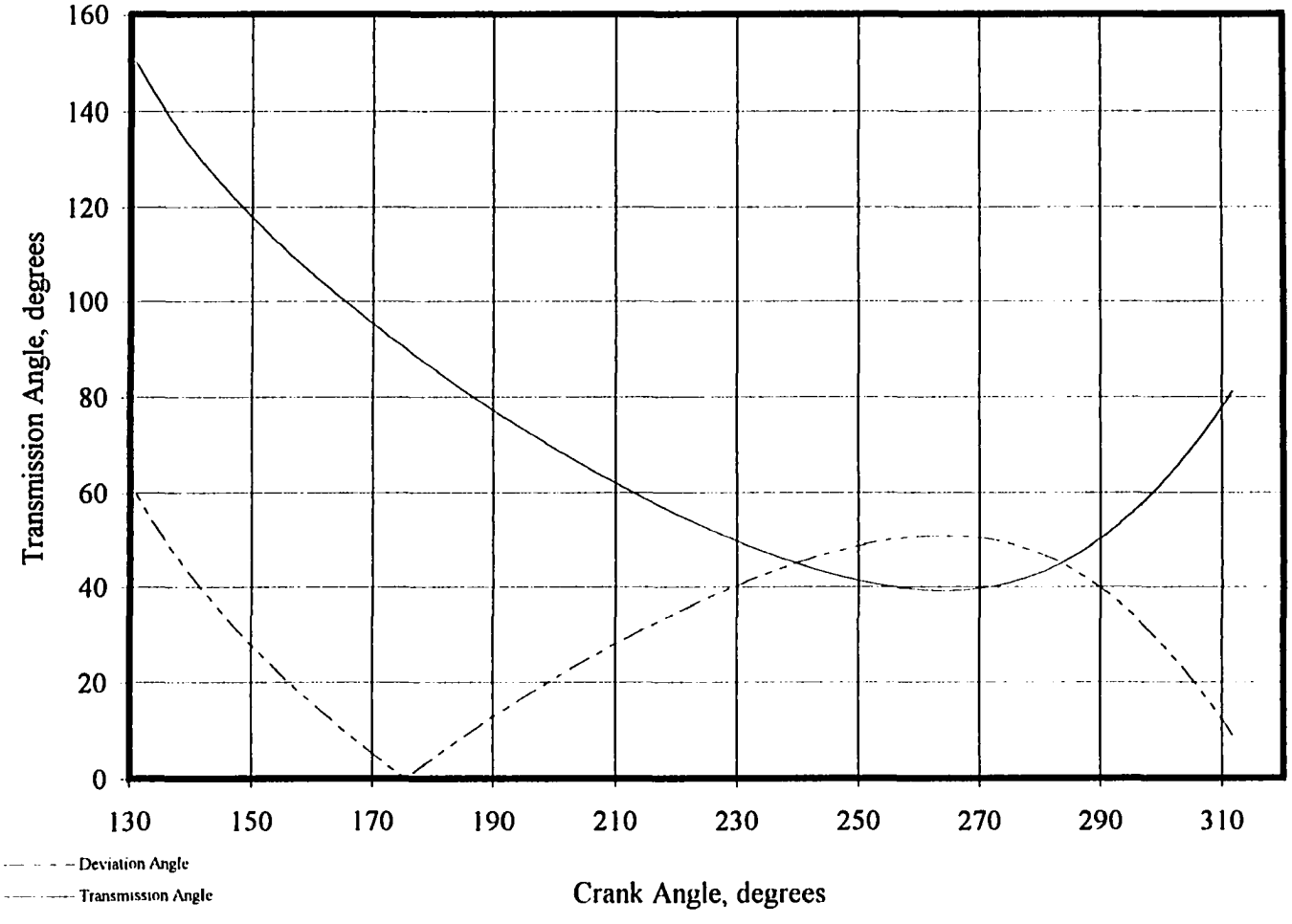

Figure 3.6: Variation in transmission angle with crank angle

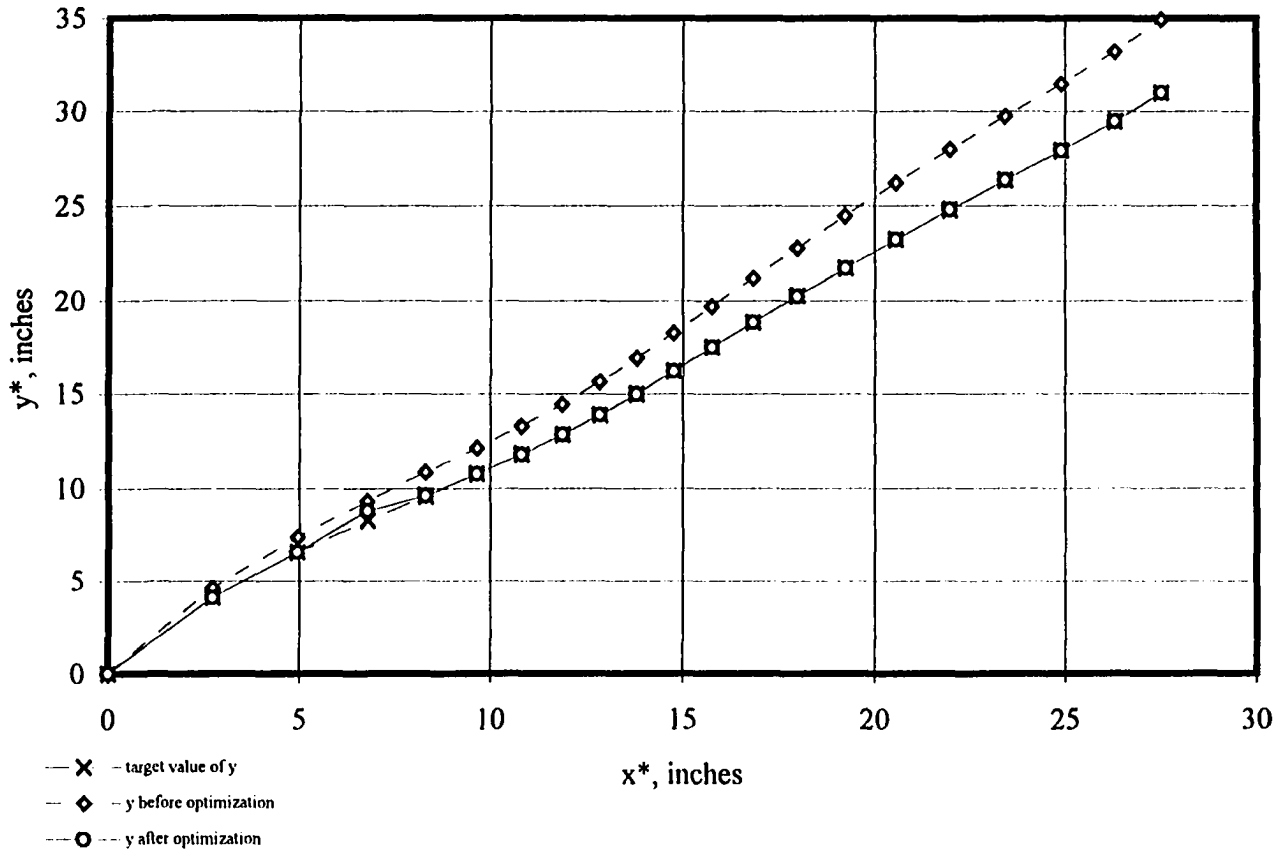

Figure 3.7: Minimization of perpendicular distances between ideal and start path

The mechanism found with the GRG method of optimization yields approximate straight line motion but with deviations of more than (+/-)3° from the actuator's initial orientation of 48.3° (see Figure 3.5). Also transmission angles, although within 50° of 90° for most of the mechanism's motion, become poor (150°) in the last 5° of the crank's angular output. Since the output on the crank is required to exceed 180° and this mechanism outputs exactly 180° the poor transmission angles in the last 5° make the mechanism depicted in Figure 3.4 unacceptable.

Simulated annealing will be implemented in the next chapter to identify mechanism`s meeting all design criteria.

IV. SIMULATED ANNEALING

Simulated annealing is a computational technique taken from statistical mechanics that emulates the naturally occurring phenomenon of annealing. The annealing process is one in which quasi-equilibrium is established at decreasing temperatures to ensure that materials cool without forming defective or metastable crystalline structures. The numerical analogy of annealing is accomplished by simulating the temperature decreases as each quasi-equilibrium state is realized. This state is reached by providing sufficiently slow cooling, or decrementation of an outer loop controlling the temperature variable. Additionally, within each of these quasi-equilibrium states, enough sampling of the domain must occur to ensure that any local optima are identified. If the algorithm is allowed to progress at a rate that provides sufficient exploration of the domain and the temperature decrement rate is slow enough to avoid quenching, a global optimum (or minimal energy state) will be found. The use of simulated annealing has become prolific [23, 24, 25, 26] and with its ease of implementation and minimal coding requirements, this numerical procedure possesses a number of compelling features upon which to capitalize.

Simulated annealing is among the most appropriate numerical procedures to provide solutions for the problems addressed within this work; it can process cost functions with arbitrary nonlinearities, discontinuities and arbitrary boundary conditions and constraints [27]. The equations given in Chapter 2 as Equations 2.27 - 2.29, when established as a set of constraints, consistently demonstrate evidence that the solution domain is highly convoluted and multimodal. As a result, a robust and non-gradient dependent procedure is

desirable for finding satisfactory solutions to the problem. The simulated annealing algorithm is summarized below:

f (x) is the objective function

g(x) is the penalty function

x is the vector of unknowns

x* is a value in the neighborhood of x

f* is the value of the objective function at x*

g* is the value of the penalty function at x*

Δf is f*-f

r is a random number on the interval [0..1]

- Set T(i): set the initial temperature
- Set length of inner loop (length of Markov chain): provide enough states
- evaluate objective function and penalties for constraint violations f+g(i)
- perturb unknown vector to a state in the "neighborhood"; x→ x*
- evaluate f*+g*
- if f*-f<0 accept x* as the current optimal solution
- if f*-f>0 generate random number, r, on the interval [0..1]
- if exp(Δf/T)>r accept this state with probability r
- check for convergence and keep best solution so far
- decrement T

The numerical procedure used for this work was written by Geoffe, Ferrier and Rogers [28] and was based on the paper by Corona, et al [29]. The code used for this application may be found in Appendix A. Two subroutines were supplied to the code. Subroutine FCN is called by the driver routine to acquire values of the objective function. Subroutine RDETER is called to acquire values of the penalty coefficient, 1/r.

The start temperature is among the most critical of the parameters supplied by the user. A temperature that is too low may induce the algorithm to find a local minimum, or to "quench"; a temperature too high results in unnecessary function evaluations. For this algorithm in particular the step size in the neighborhood search is dependent upon the value of temperature and the number of accepted states at a given temperature. To find a satisfactory start temperature, the temperature was gradually increased until the initial step size of each parameter was deemed adequate to sample an appropriate region of the domain. As the temperature decreases, the step size for the neighborhood search does as well, ultimately yielding a search in the most promising areas of the domain for the occurrence of an optimum [31]. The relationship between number of states accepted and the step size is given below. The step size is adjusted so that approximately half of all function evaluations are accepted after ns*n function evaluations, where

ratio = number of accepted states/number of cycles

ci = constant coefficient used to scale step size, vm

vm = vm*(1+ci*(ratio-.6)/.4) if ratio > .6 [4.1]

vm = vm/[(1+ci*(.4-ratio)/.4]if ratio < .4

ns = number of cycles in inner loop

n = number of variables

nt = the number of iterations before the temperature variable is decremented

The temperature is decremented by 0.85*T after nt iterations and ns*n inner loops. Consequently, the total number of inner loops is nt*ns*n. For the problem at hand this

yields, nt = 15, ns = 20 and n = 9 or 15*20*9 (2700) states visited at each temperature. The number of states visited within the inner loop must be sufficient enough to ensure that a low energy configuration will be reached. With decreasing temperatures, and especially toward the freezing temperature, this requirement becomes critical [31].

The objective function consists of the summation of the differences between the initial actuator orientation, θ_{c_1} and orientations at subsequent positions, i.e., $|\theta_{c_1}-\theta_{c_j}|$. The penalty function is given as the sum of the squares of the *violated* constraints times a penalty coefficient that approaches infinity as the constraints become *active*. An *active* constraint is one where the equality constraints have zero as their right hand side and a *non-binding* constraint has a non-zero right hand side. The constraint equations consist of expressions for the virtual work at each of the precision points. Assuming a clockwise displacement of the driven link, member Z_1, the required torque for static equilibrium would be in a counterclockwise, or positive direction (see Figure 4.1). If the term involving F_c in Eq. 4.2 is negative, the constraint will become binding, or equal to zero, with the appropriate addition of a positive, or counterclockwise, torque.

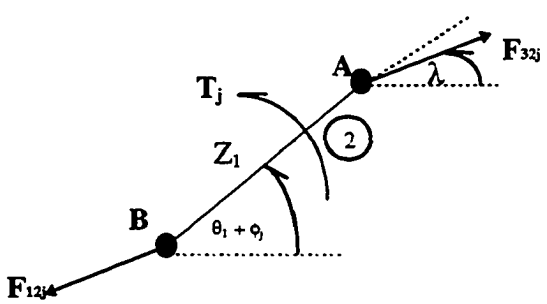

Figure 4.1: Parameters used for determination of moments

F_{cj} = Force of actuator, position j
Z_1 = Crank length
θ_{1j} = Orientation of crank, position j
R = Length of coupler link joining link members Z_1 and Z_1*
γ_j = Orientation of link R, position j
Z_1* = Length of follower link
θ_{3j} = Orientation of link Z_1*, position j
L = perpendicular distance from link R to precision point
B = distance from joint of links Z_1 and R along R to perpendicular L
$\beta = \tan^{-1}\left(\frac{L}{B}\right)$

$$F_{c_j}\left\{Z_1 \sin\left(\theta_{c_j} - \theta_{1_j}\right) + Z_2 \sin\left(\theta_{c_j} - \left(\gamma_j + \beta\right)\right) * \frac{Z_1}{R}\left[\frac{-\sin(\theta_{3_j} - \theta_{1_j})}{\sin(\theta_{3_j} - \gamma_j)}\right]\right\} + T_j = 0 \quad [4.2]$$

Equation 4.3 represents the penalty function used within this procedure. The penalty coefficient, 1/r, is determined by examining the magnitudes of the constraint violations and is decreased with decreasing distance from the constraint boundary. This observation leads to the penalty definition in the simulated annealing procedure. The function being minimized is penalized when the first portion of the virtual work expression is positive (all terms collectively in Eq. 4.2, except T_j). Each of the constraint equations that do not yield positive values (a constraint violation) are treated as follows:

$$g\left(\vec{x}\right) = \left[F_{c_j} Z_1 \sin\left(\theta_{c_j} - \theta_{1_j}\right) + F_{c_j} Z_2 \sin\left(\theta_{c_j} - \left(\gamma_j + \beta\right)\right) * \frac{Z_1}{R}\left[\frac{-\sin(\theta_{3_j} - \theta_{1_j})}{\sin(\theta_{3_j} - \gamma_j)}\right]\right]^2 \quad [4.3]$$

$$penalty = \sum_j^{\text{inactive constraints}} g\left(\vec{x}\right) \quad [4.4]$$

To maintain transmission angles that are acceptable, an objective function was established such that if the transmission angle's deviation from 90° exceeds 20°, the contribution to the objective function from the minimization of transmission angles is not reduced. If the link parameters place the transmission angle between 110° and 70° the contribution to the value of the objective function is set to zero. Any value of the actuator orientation that is within a predetermined tolerance of the simulation's value for θc was considered acceptable for approximate straight line motion. The composite objective function used to achieve optimal synthesis is given below. The index j varies from 1 to the number of precision points, n.

$$\min_{f}\left(\vec{x}\right) = w_1\left[\sum_{j=1}^{n}\left(\left|\theta_{c_{j-1}} - \theta_{c_1}\right|\right)\right] + \frac{1}{r}\left[\sum_{k}^{\text{inactive constraints}} g_k\left(\vec{x}\right)\right] + w_2\left(\sum_{j=1}^{n}\left|90^\circ - \mu_{T_j}\right|\right) \qquad [4.5]$$

where $\vec{x}$ is the vector of unknowns.

The weights for the two objectives are 1.5 for straight line motion, w_1, and 1 for the deviation of the transmission angle from 90°, w_2.

The equation given below indicates the relationships between the moment equation about point B of link member Z_1. The sense of the output torque is determined by the value of the difference in $\theta_1 + \phi_j$ and the direction of the force imposed by the coupler on the crank, angle λ in Figure 4.1. To maintain a positive, counterclockwise sense of torque for example, the composite angle $\theta_1 + \phi_j - \lambda_j$ must remain between 0° and 180°. Likewise, a clockwise sense would require a composite angle between 180° and 360°.

$$
\begin{aligned}
&-F_{32_{x_j}} Z_1 \sin(\theta_1+\phi_j)+F_{32_{y_j}} Z_1 \cos(\theta_1+\phi_j)+T_j=0\\
&-F_{32}\cos(\lambda_j)Z_1\sin(\theta_1+\phi_j)+F_{32}\sin(\lambda_j)Z_1\cos(\theta_1+\phi_j)+T_j=0\\
&\frac{-T_j}{Z_1F_{32}}=-\cos(\lambda_j)\sin(\theta_1+\phi_j)+\sin(\lambda_j)\cos(\theta_1+\phi_j) \qquad [4.6]\\
&\frac{T_j}{Z_1F_{32}}=\sin(\theta_1+\phi_j-\lambda_j)
\end{aligned}
$$

Figure 4.2 represents a mechanism synthesized with simulated annealing with the minimization of the function represented by Eq. 4.5. The values for link Z_1's angular displacements were specified for 8 displacements, resulting in a prescribed timing problem with 9 precision points. Objectives of the minimization were to:

a). produce straight line motion
b). produce a consistent sense of torque (counterclockwise in this case)
c). produce acceptable deviations from 90° of the transmission angle
d). produce no less than 180° of angular displacement by the crank

The mechanism's parameters are summarized in Table 4.1. The simulated annealing parameters were as follow:

Temperature = 2000; Temperature Decrement = 0.85;
Coefficient for step size, 0.25;
Tolerance for convergence =1.E-6

Table 4.1: Four-bar straight line mechanism found with Simulated Annealing

Z_1 Z_1/Z_1	θ_1	Z^*_1 Z^*_1/Z_1	θ_3	Z Z/Z_1	θ	b b/Z_1	L L/Z_1	θc
6.16 1.00	202.0°	35.58 5.76	223.8°	6.88 1.17	291.4°	6.67 1.08	20.31 3.30	161.1°

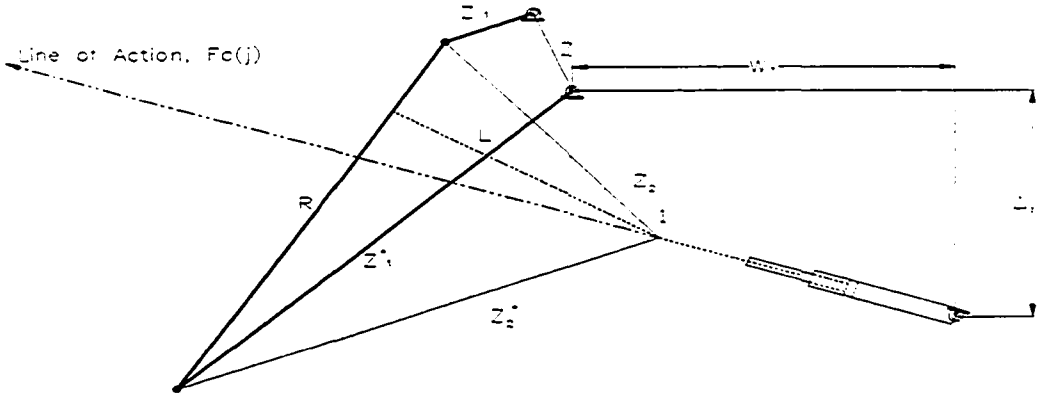

Figure 4.2 - Mechanism 1 in its initial position

The simulation required 375 301 function evaluations and reached a final temperature of 3.6378E-7 before convergence criteria were met. The parameters Wx and Δy are determined by closing a loop around the actuator and the left dyad of the mechanism. Figure 4.3 indicates the arrangement of the actuator relative to the left dyad. The ground of the actuator may occur at any point along the line of action of the actuating force, Fc as long as its placement does not interfere with the mechanism.

Figure 4.4 depicts the mechanism in four of the sixteen positions for which optimization was performed. Also shown is the mechanism in relation to the actuator ground, Wx and Δy. Figure 4.5 indicates the path of the coupler point and is immediately

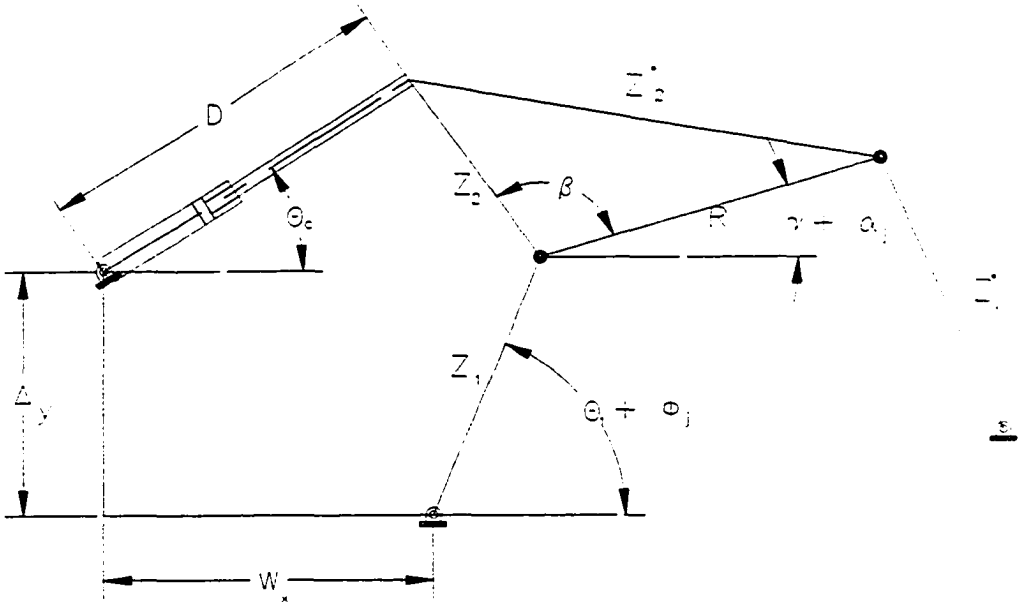

Figure 4.3: Left Dyad and Wx and Δy

followed with a plot of the actuator orientation over the full range of motion of the crank(Fig. 4.6). The variation in the angle is at most (+/-) 2.5°. Also depicted is the transmission angle and its deviation from 90° (Fig. 4.7). The mechanism is shown at four precision points in relation to the point of concurrency of the actuator force, F_c, the line of action of the force at the pin joint connecting link Z_1 to the coupler link and the line of action of link Z^*_1 (Figures 4.8-4.9). Figure 4.10 represents the behavior of the forces imposed on the mechanism as a function of the crank's angular displacements. Forces on the mechanism were determined by solving the linear system of equations given by Eq. 4.8. The value of torque, T_j, used for each position was 10 in-lbs.

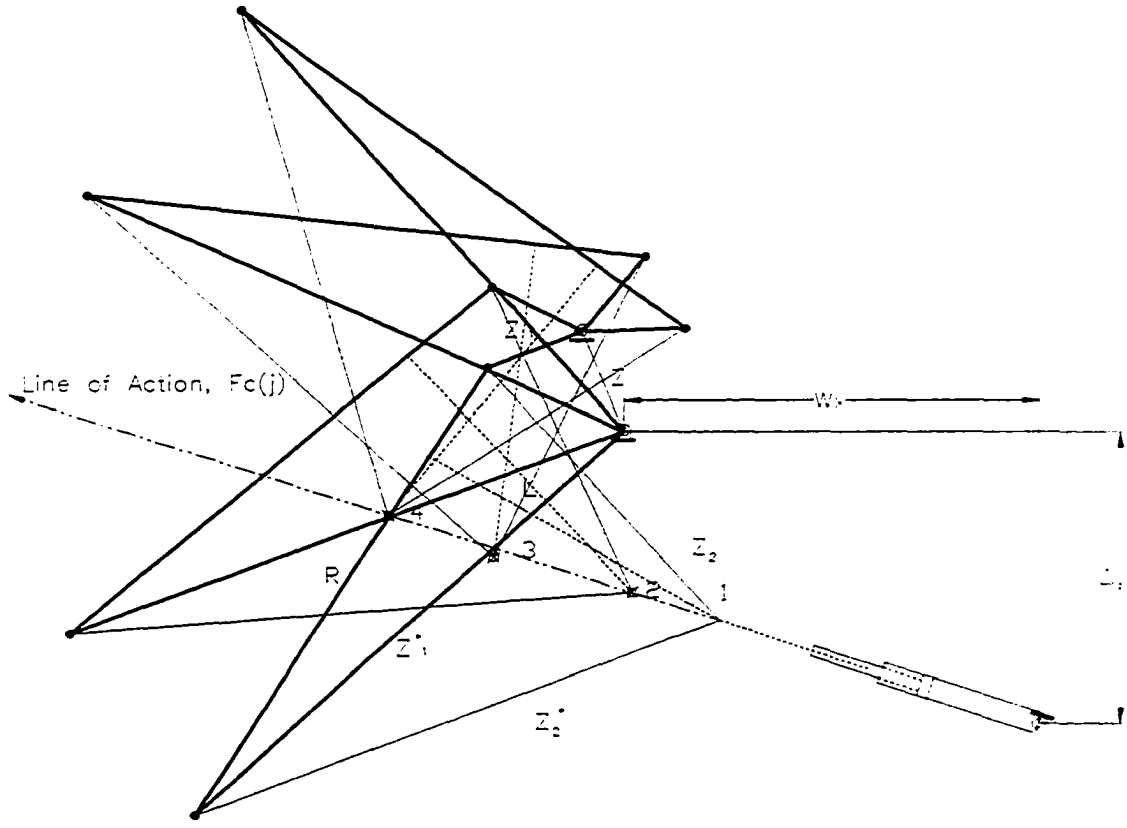

Figure 4.4: Mechanism 1 in four positions

$$\begin{bmatrix} \cos(\theta_{c_j}) & \cos(\theta_{3_j}) & 1 & 0 \\ \sin(\theta_{c_j}) & \sin(\theta_{3_j}) & 0 & 1 \\ 0 & 0 & -Z_1\sin(\theta_1+\phi_j) & Z_1\cos(\theta_1+\phi_j) \\ b\sin(\theta_{c_j}-\gamma_j)-L\cos(\theta_{c_j}-\gamma_j) & R\sin(\theta_{3_j}-\gamma_j) & 0 & 0 \end{bmatrix} \begin{Bmatrix} F_{c_j} \\ F_{14_j} \\ F_{12_{x_j}} \\ F_{12_{y_j}} \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ -T_j \\ 0 \end{Bmatrix} \quad [4.8]$$

The second mechanism found with the use of simulated annealing is represented in its initial position in Figure 4.11. The crank angle's angular orientation was varied by eight equally spaced increments of 22.625° in a clockwise direction. The transmission angles for

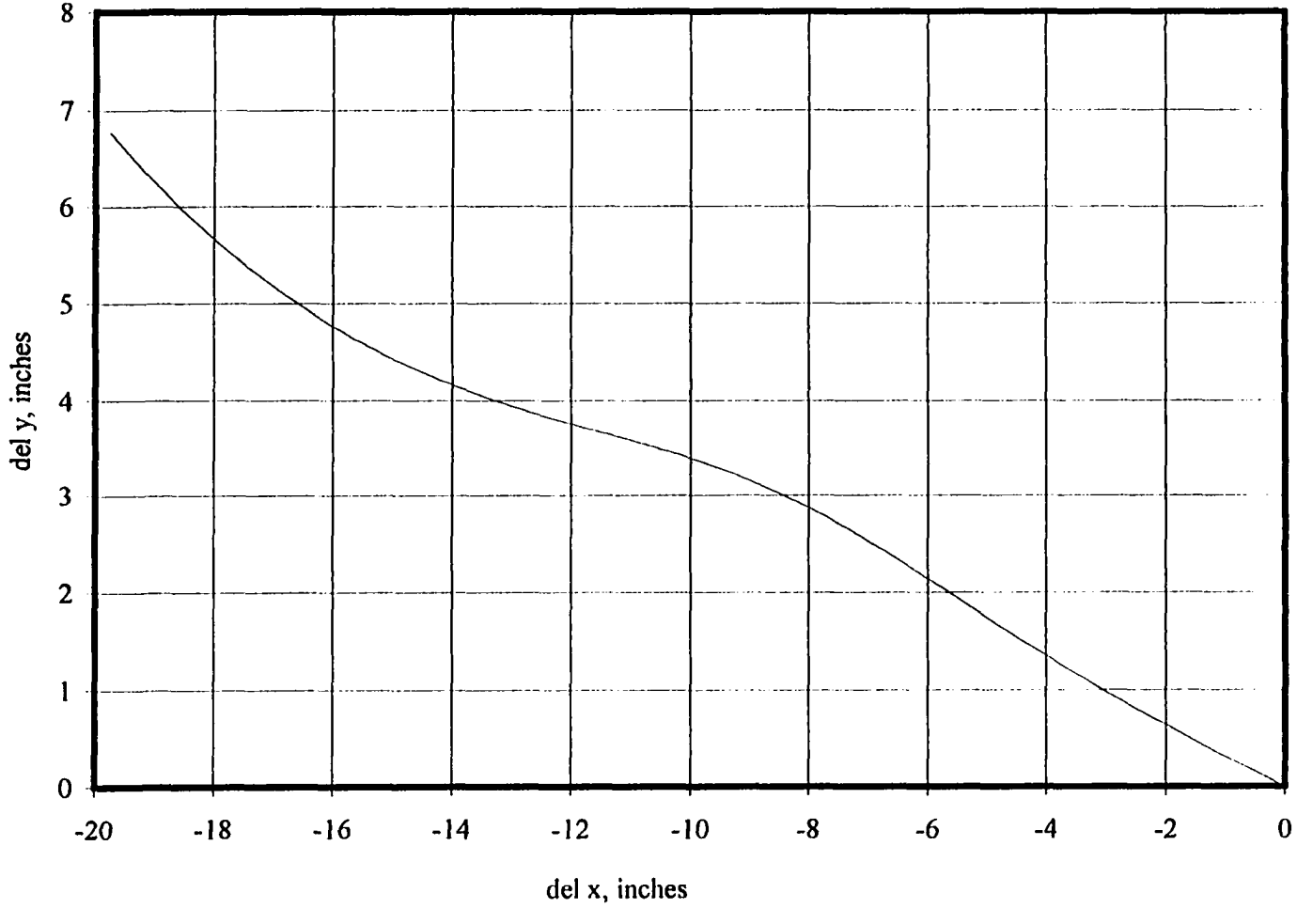

Figure 4.5: Coupler path mechanism 1

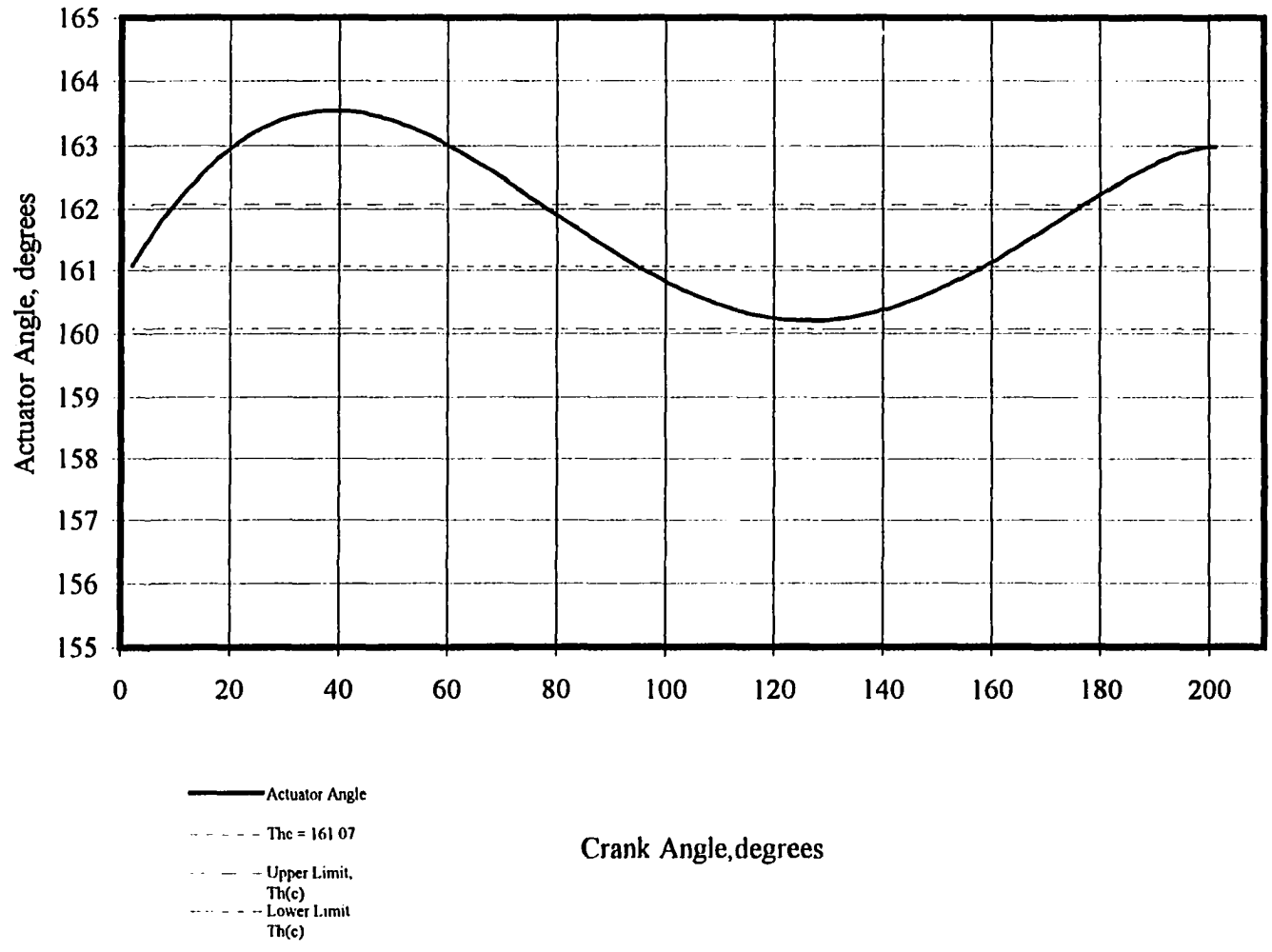

Figure 4.6: Actuator orientation, mechanism 1

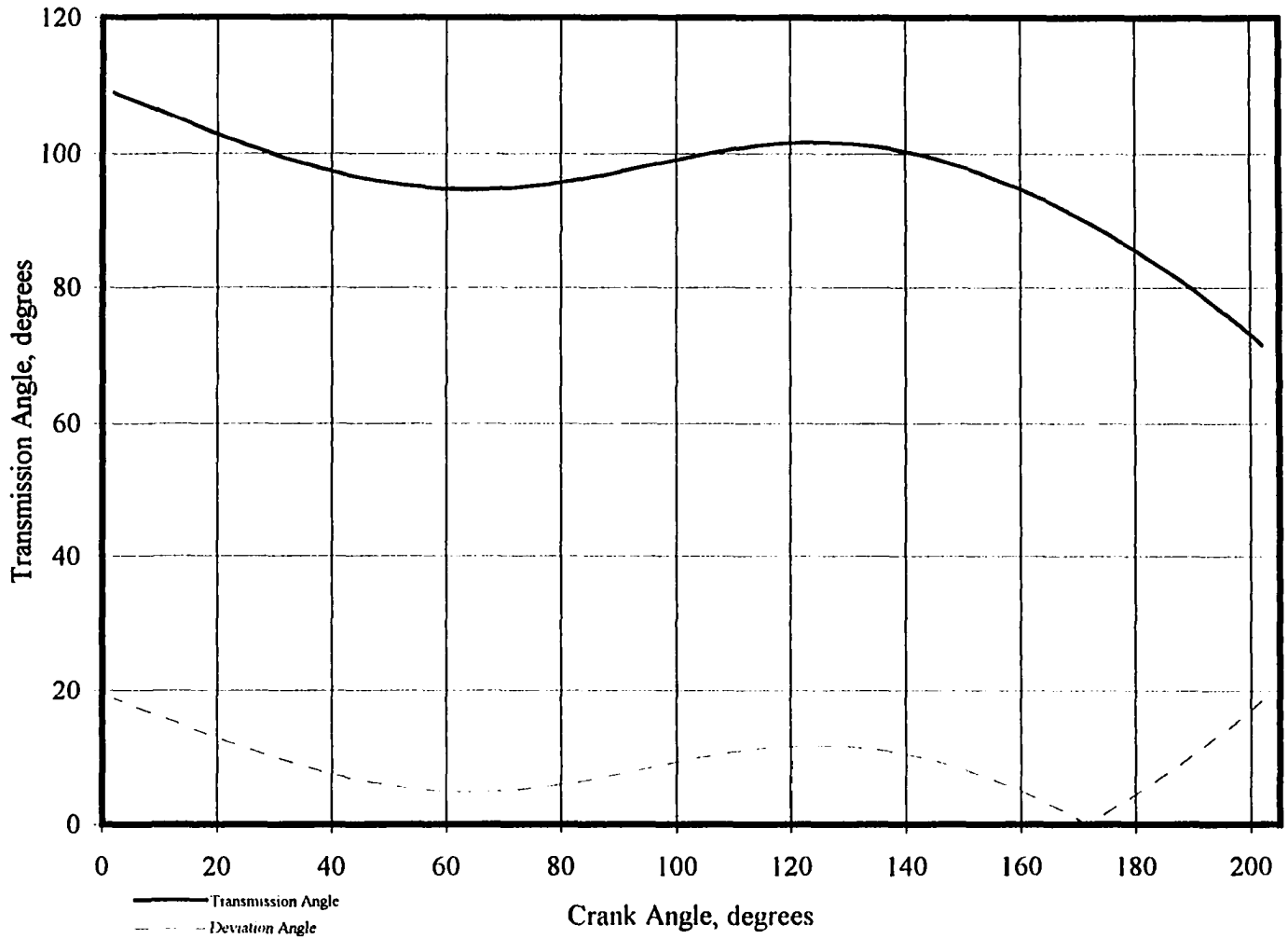

Figure 4.7: Transmission angles, mechanism 1

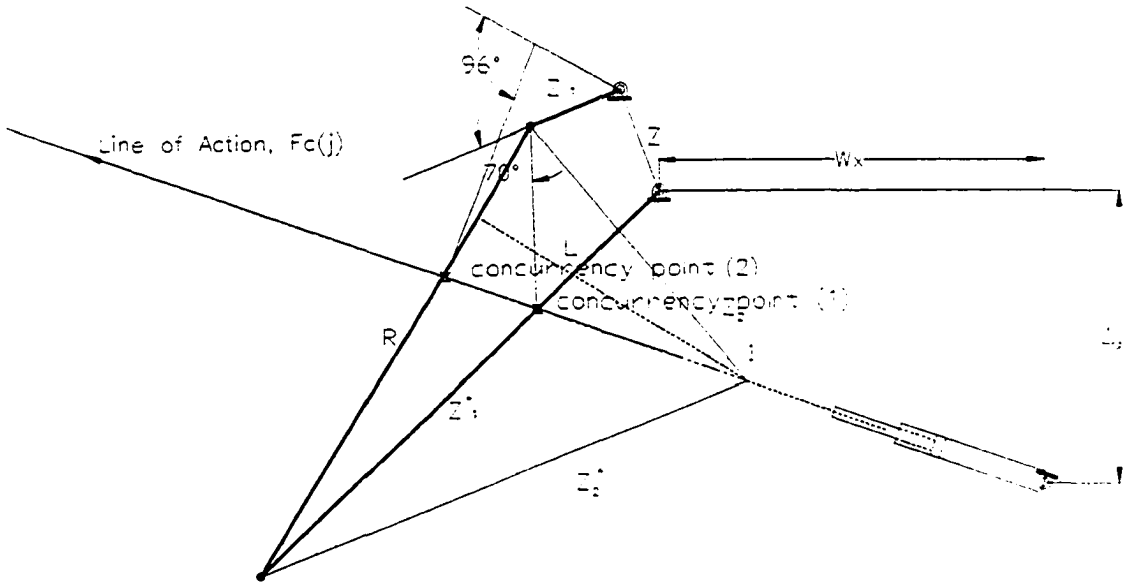

Figure 4.8: Forces acting on mechanism 1, positions 1 and 2

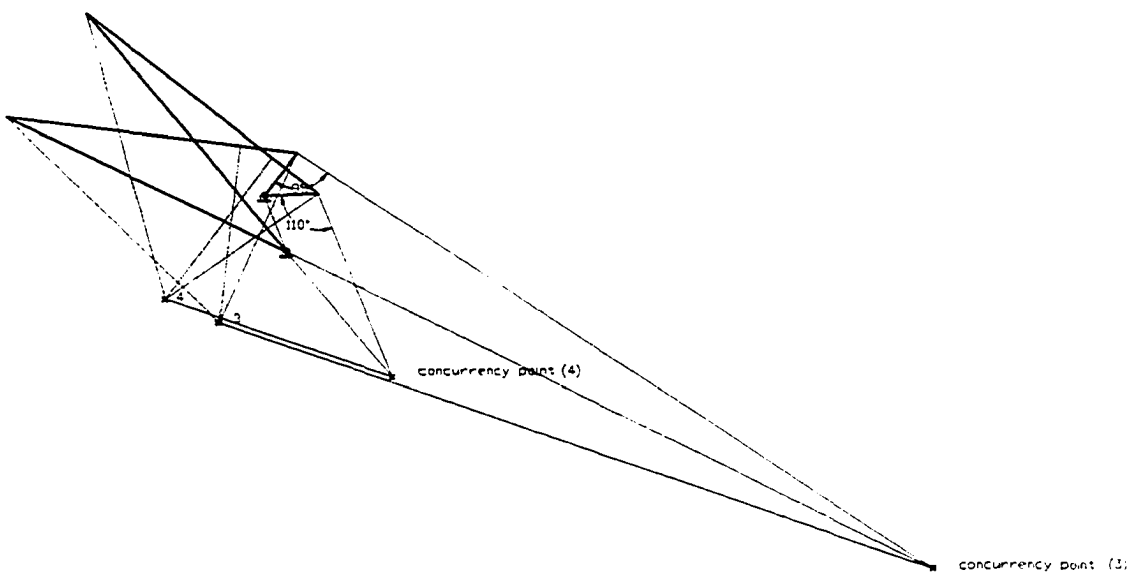

Figure 4.9: Forces acting on mechanism 1, positions 3 and 4

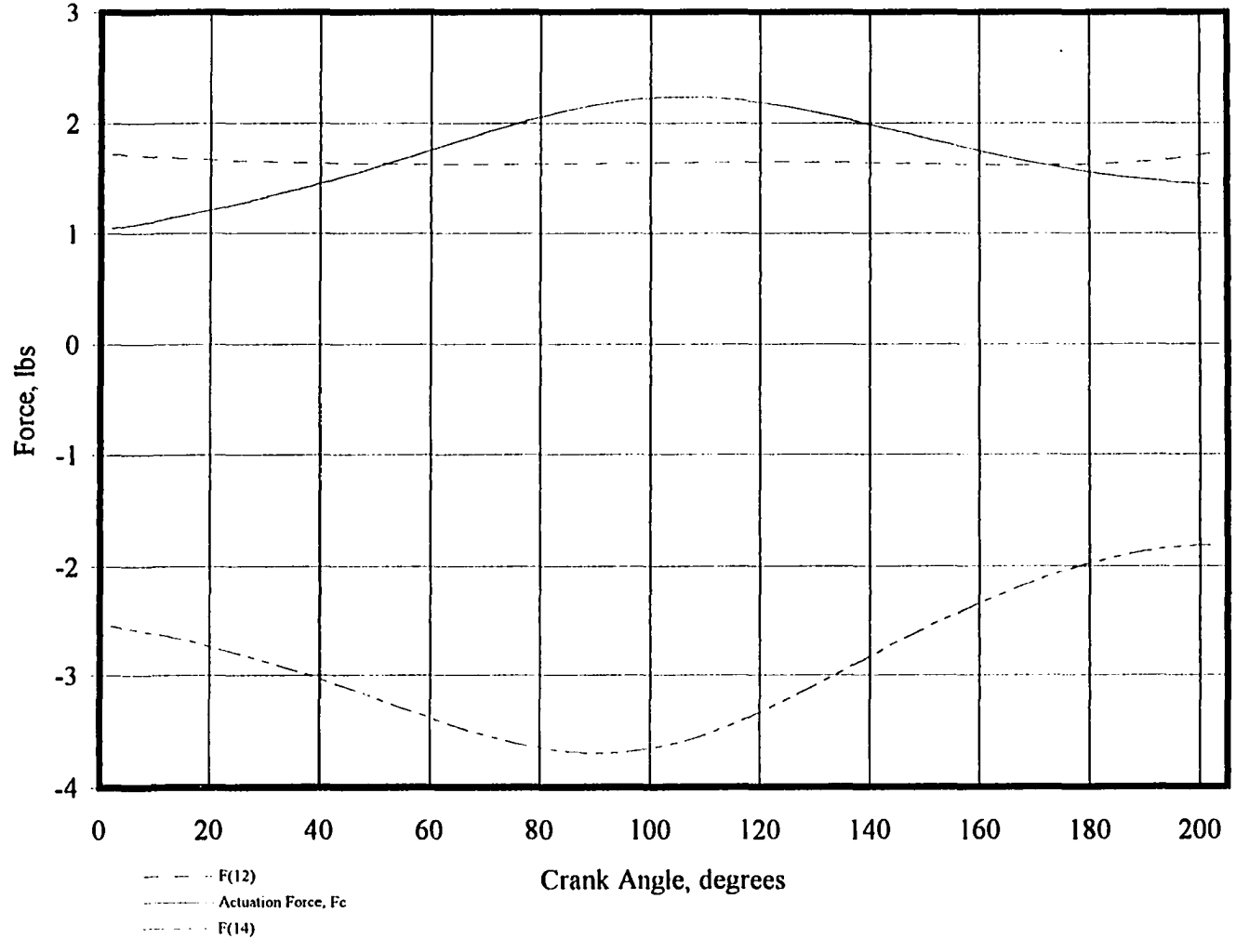

Figure 4.10: Force variation with crank angle, mechanism 1

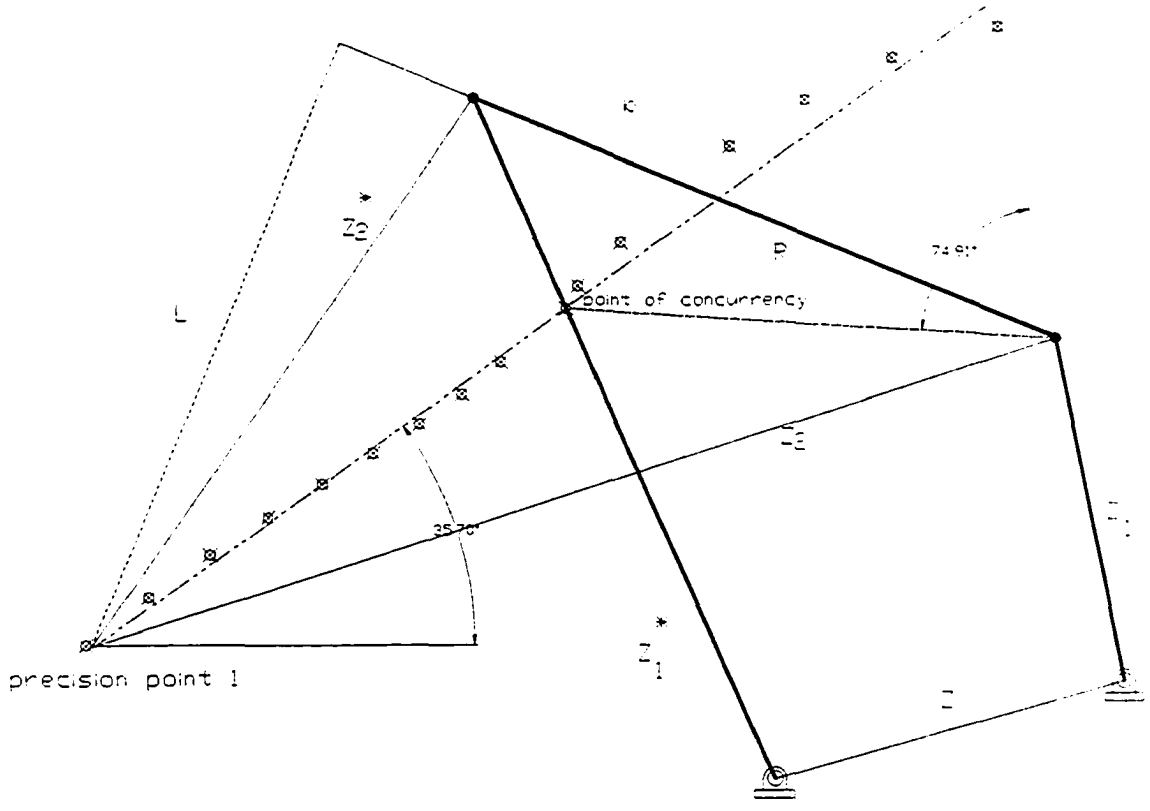

Figure 4.11: Mechanism 2 in initial position

this mechanism were held to within 20° of 90°, and the deviation band of the actuator was held to within 2° of the value of θ_{c1}. The mechanism's parameters are noted in Table 4.2 and the behavior of the coupler path and the transmission angles with respect to the crank angle are shown in Figures 4.12 and 4.13 respectively. The actuator's angular variance with increasing output crank angle is noted in Figure 4.14. Force behavior follows in Figure 4.15.

The parameters used for the simulated annealing procedure were as follows:

Temperature = 2000; Temperature Decrement = 0.85;
Coefficient for step size, 0.25;
Tolerance for convergence =1.E-6

Table 4.2: Mechanism 2 - Final kinematic parameters

Z_1 Z_1/Z_1	θ_1	Z^*_1 Z^*_1/Z_1	θ_3	Z Z/Z_1	θ	b b/Z_1	L L/Z_1	θc
17.56 1.00	101.4°	37.32 2.13	113.6°	18.22 1.04	195.2°	38.84 2.21	32.82 1.87	35.7°

The simulation required 172 801 function evaluations and reached a final temperature of 7.15e-2 before convergence criteria were met.

The third mechanism produced was found by using 20 evenly spaced angular displacements of the crank over a range of 220°. The deviation angle of the transmission angle was to be 20° or less. The variance from the path's slope was considered acceptable if less than 2° of the simulation's value for the actuator orientation was encountered. The mechanism identified meets the requirement of the straightness over the entire interval. All positions of the crank keep the path that the mechanism is being pushed along in a 2° envelope. The quality of the mechanism with respect to transmission angles is likewise good. The maximum value of the transmission angle over the 220° range of the crank reaches 110° and the minimum value approaches 70°. Figure 4.16 shows this mechanism at an initial crank position of 236.98°. Figures 4.17 and 4.18 depict the coupler path generated by mechanism 3 and the variation in the path from linear behavior, respectively. Transmission angles are shown in Fig. 4.19 and force behavior is reflected in Fig. 4.20. Table 4.3 lists the kinematics parameters of mechanism 3.

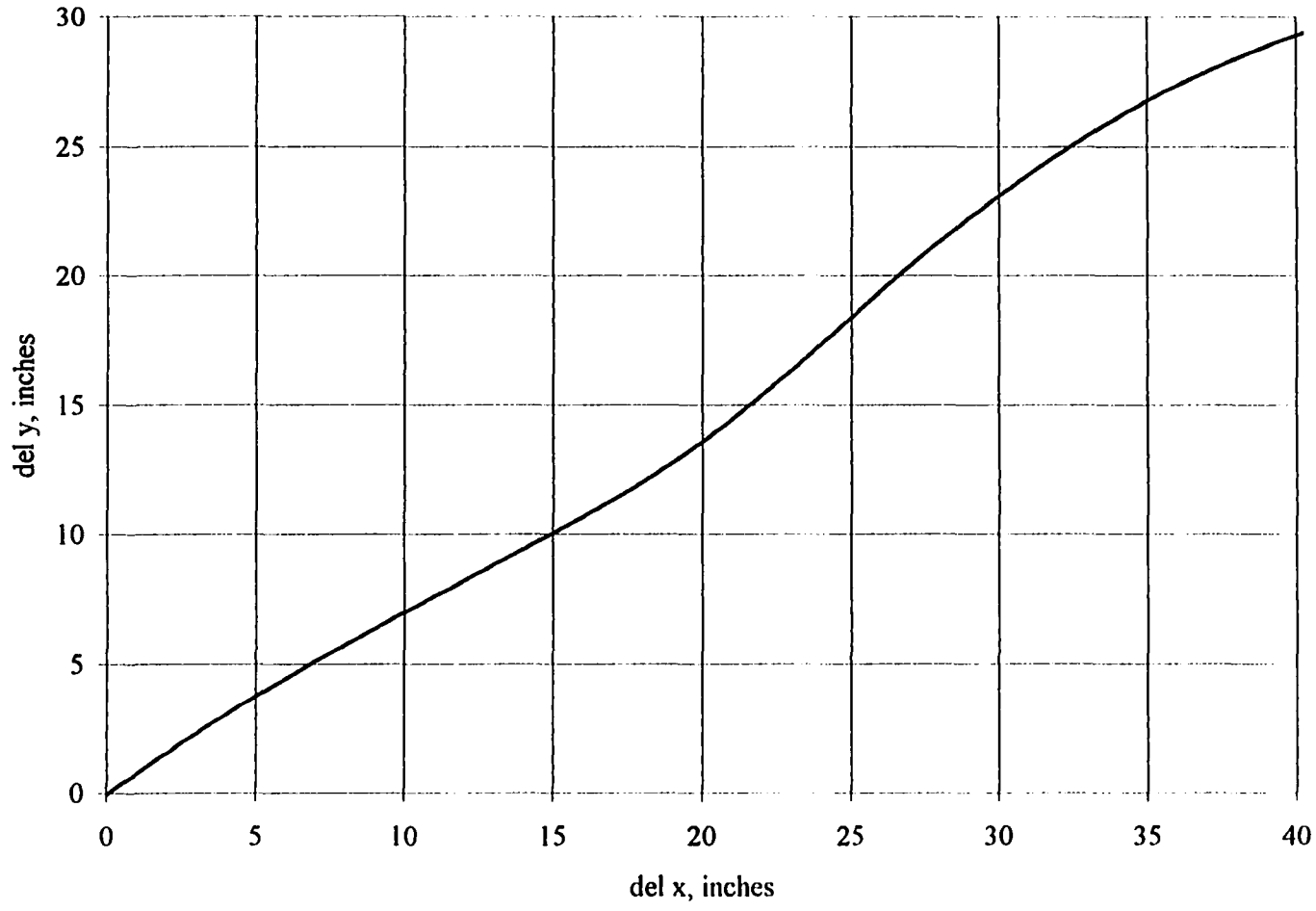

Figure 4.12: Mechanism 2 coupler path

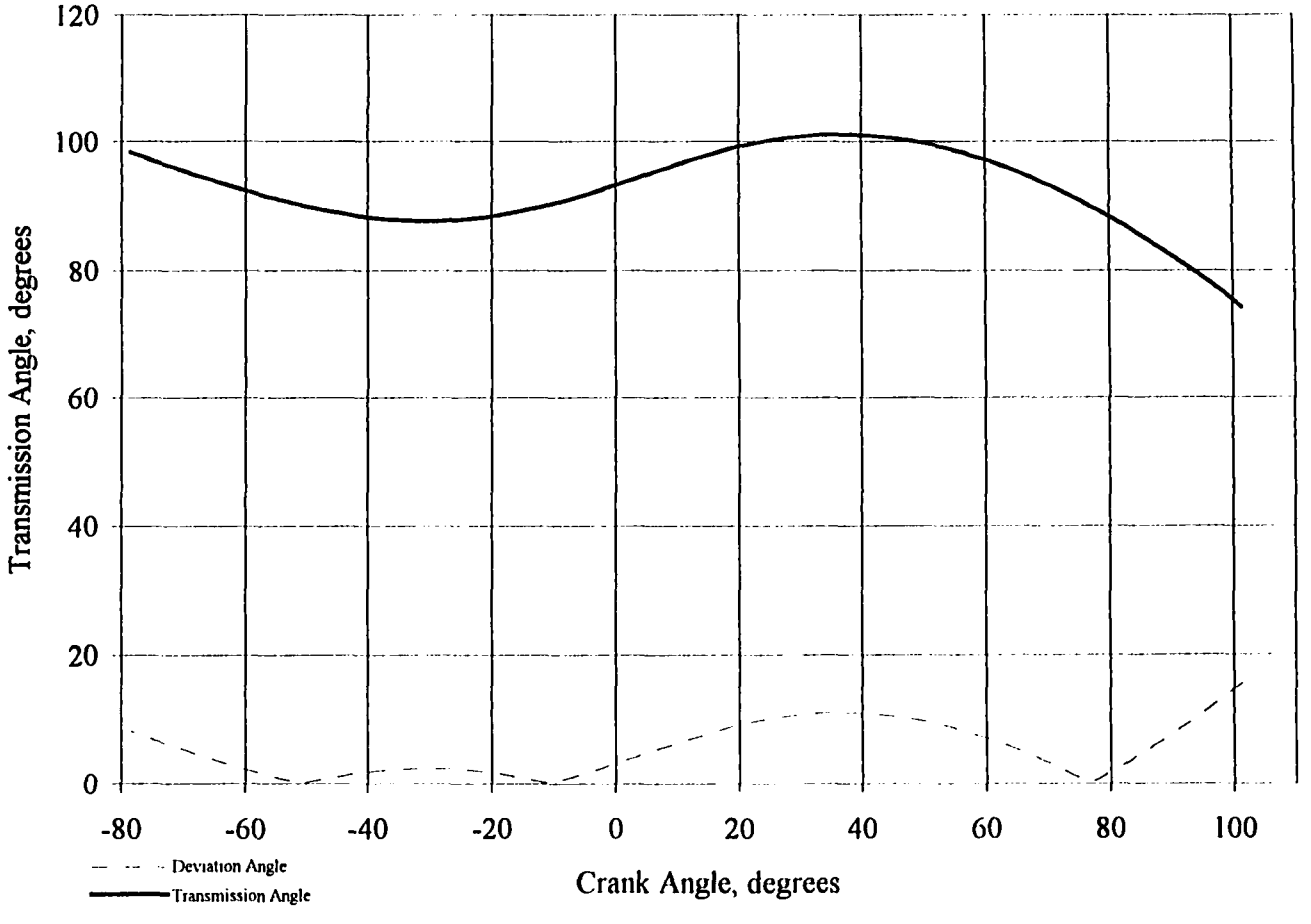

Figure 4.13: Variation in transmission angle, mechansim 2

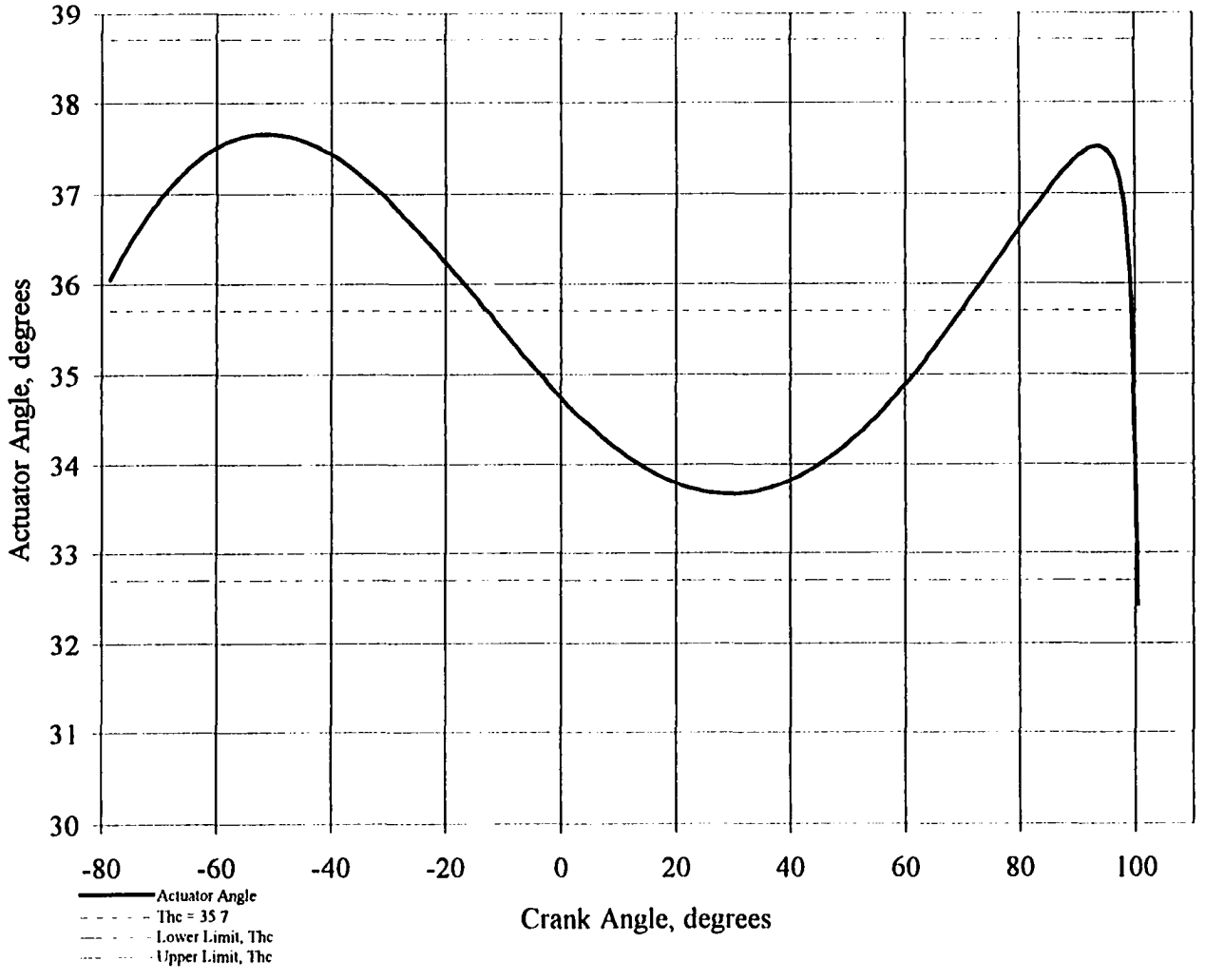

Figure 4.14: Variation in actuator angle, mechanism 2

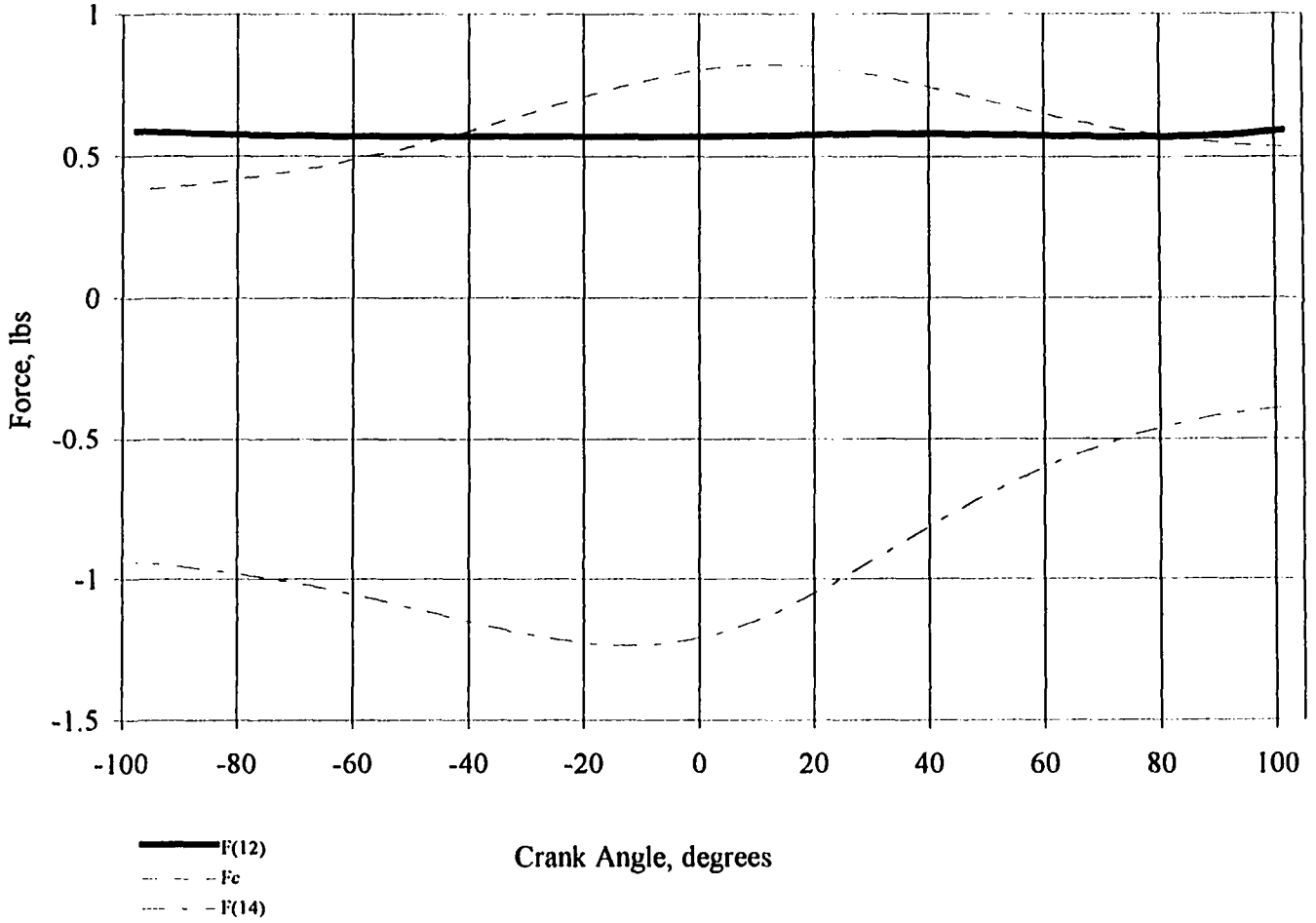

Figure 4.15: Forces acting on mechanism 2

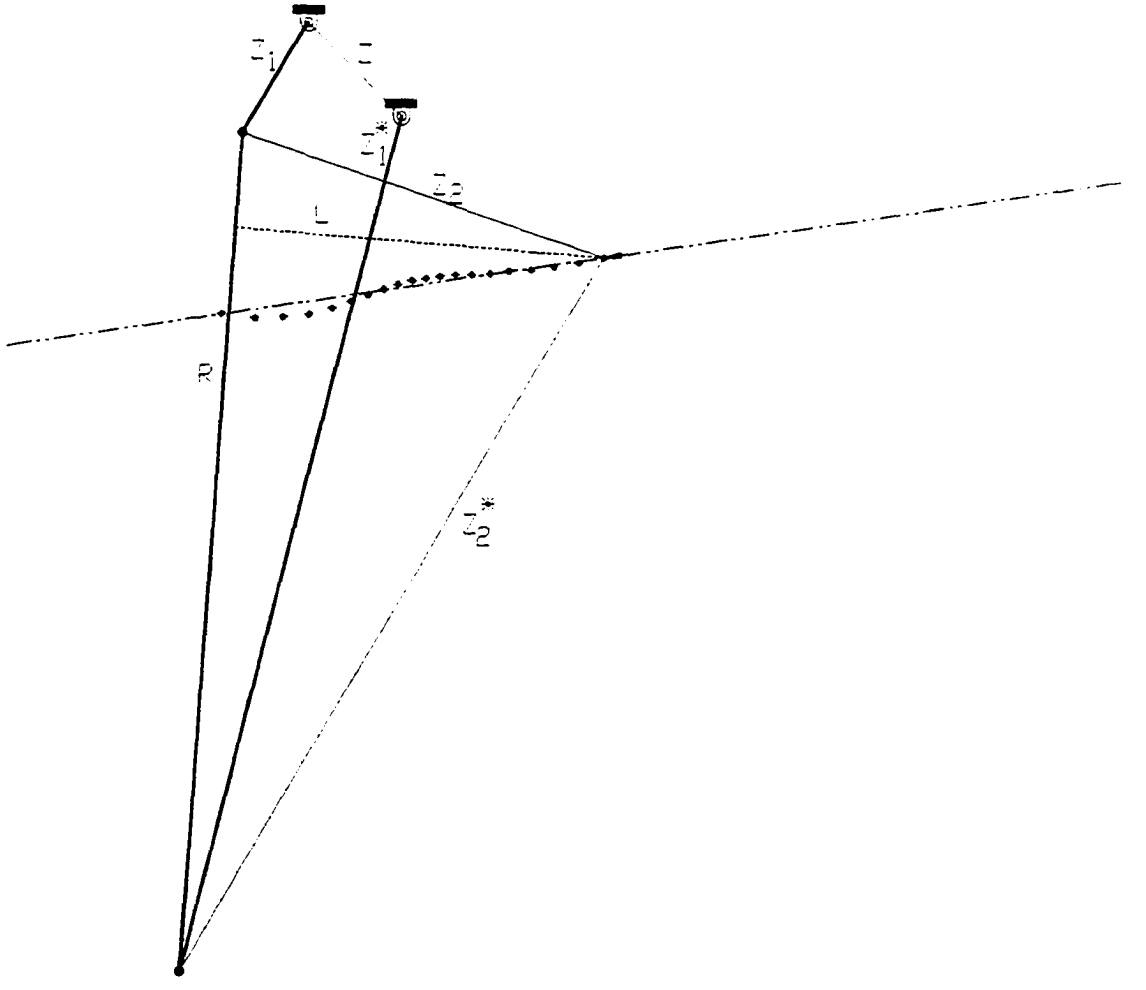

Figure 4.16: Mechanism 3 in initial position

Table 4.3: Mechanism 3 - Final kinematic parameters

Z_1 Z_1/Z_1	θ_1	Z^*_1 Z^*_1/Z_1	θ_3	Z Z/Z_1	θ	b b/Z_1	L L/Z_1	θc
6.96 1.00	236.9°	47.04 6.76	254.33°	7.31 1.05	316.5°	5.03 0.72	47.04 6.76	187.5°

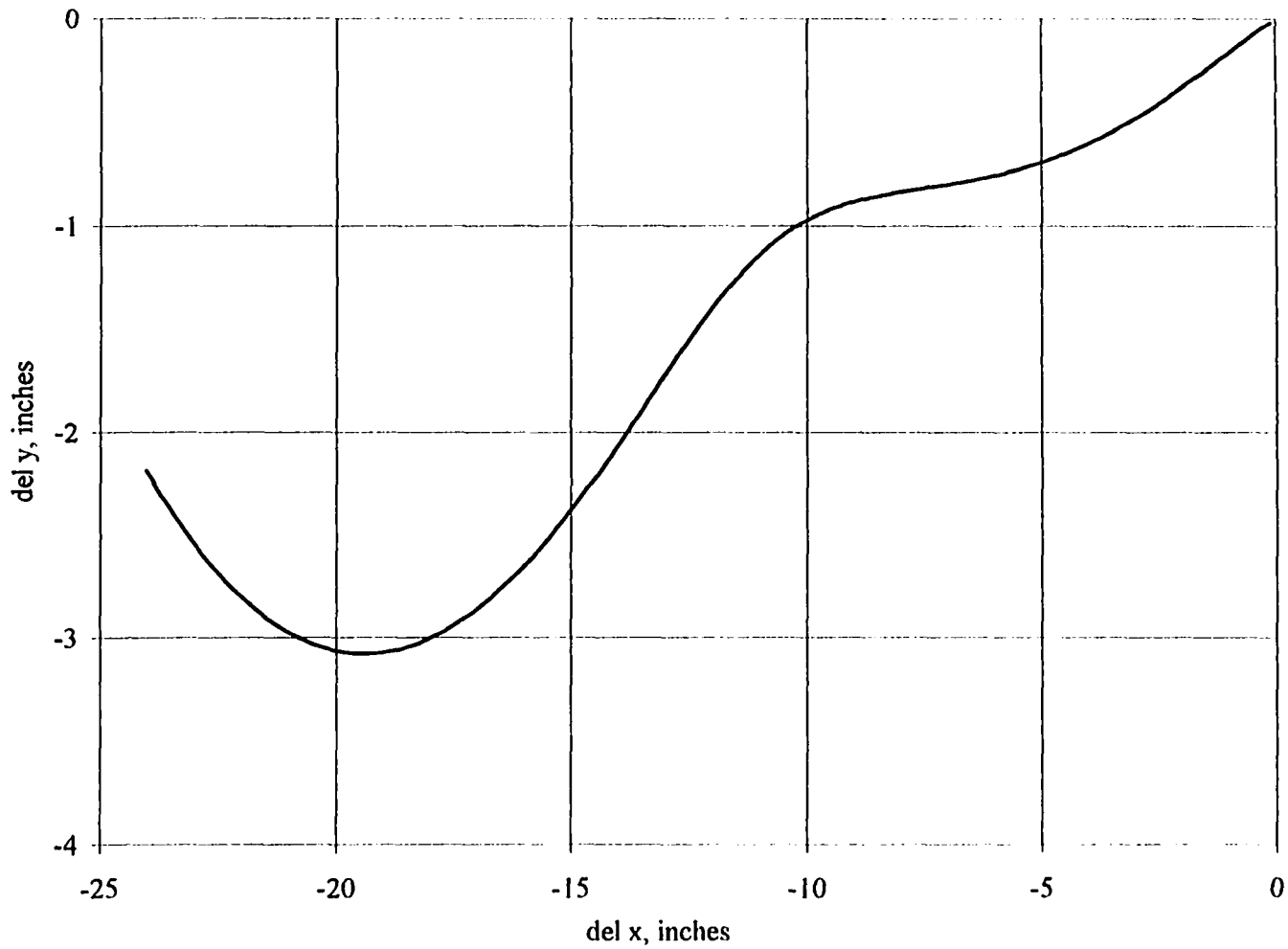

Figure 4.17: Coupler path mechanism 3

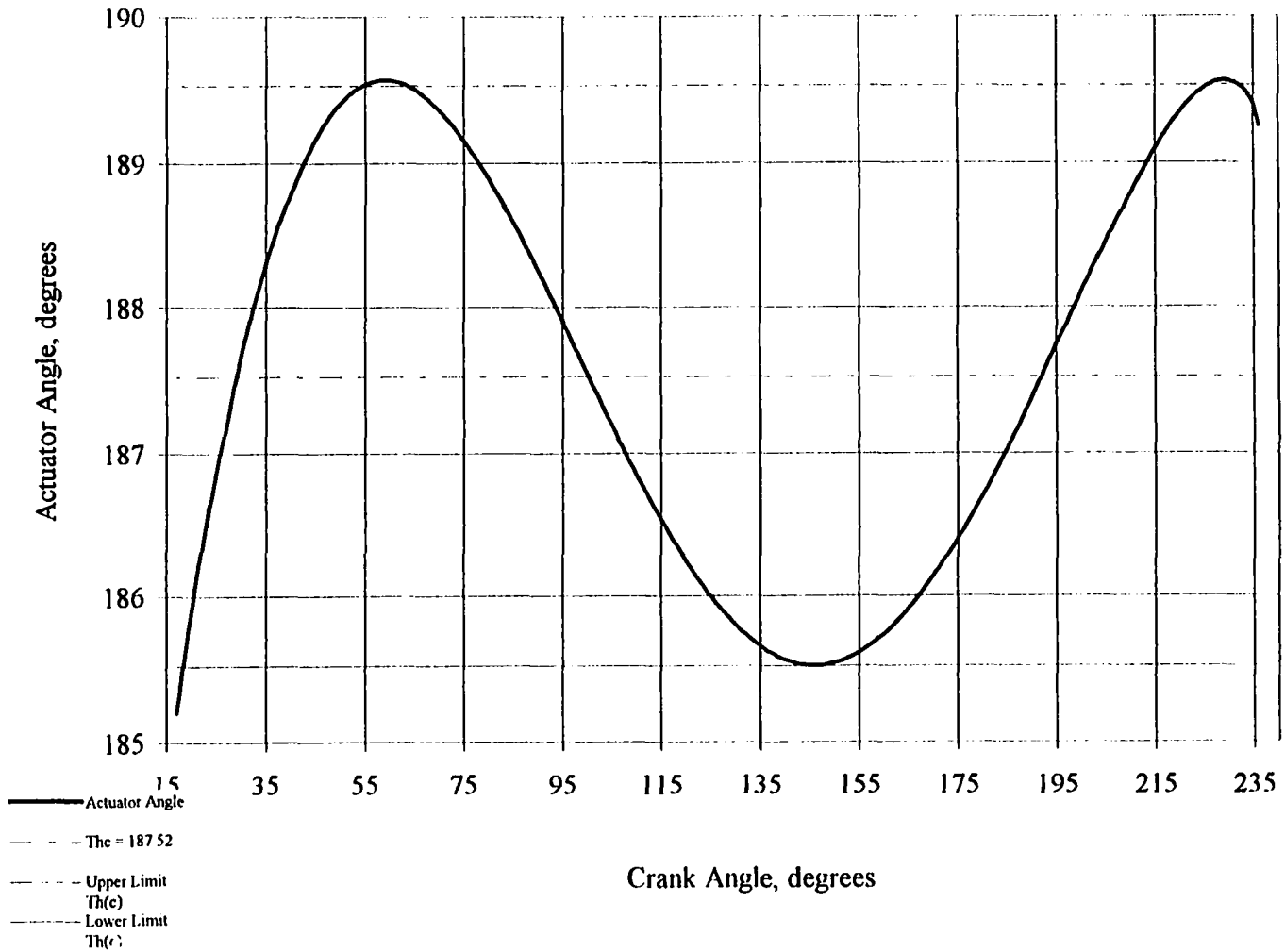

Figure 4.18: Variation in actuator orientation, mechanism 3

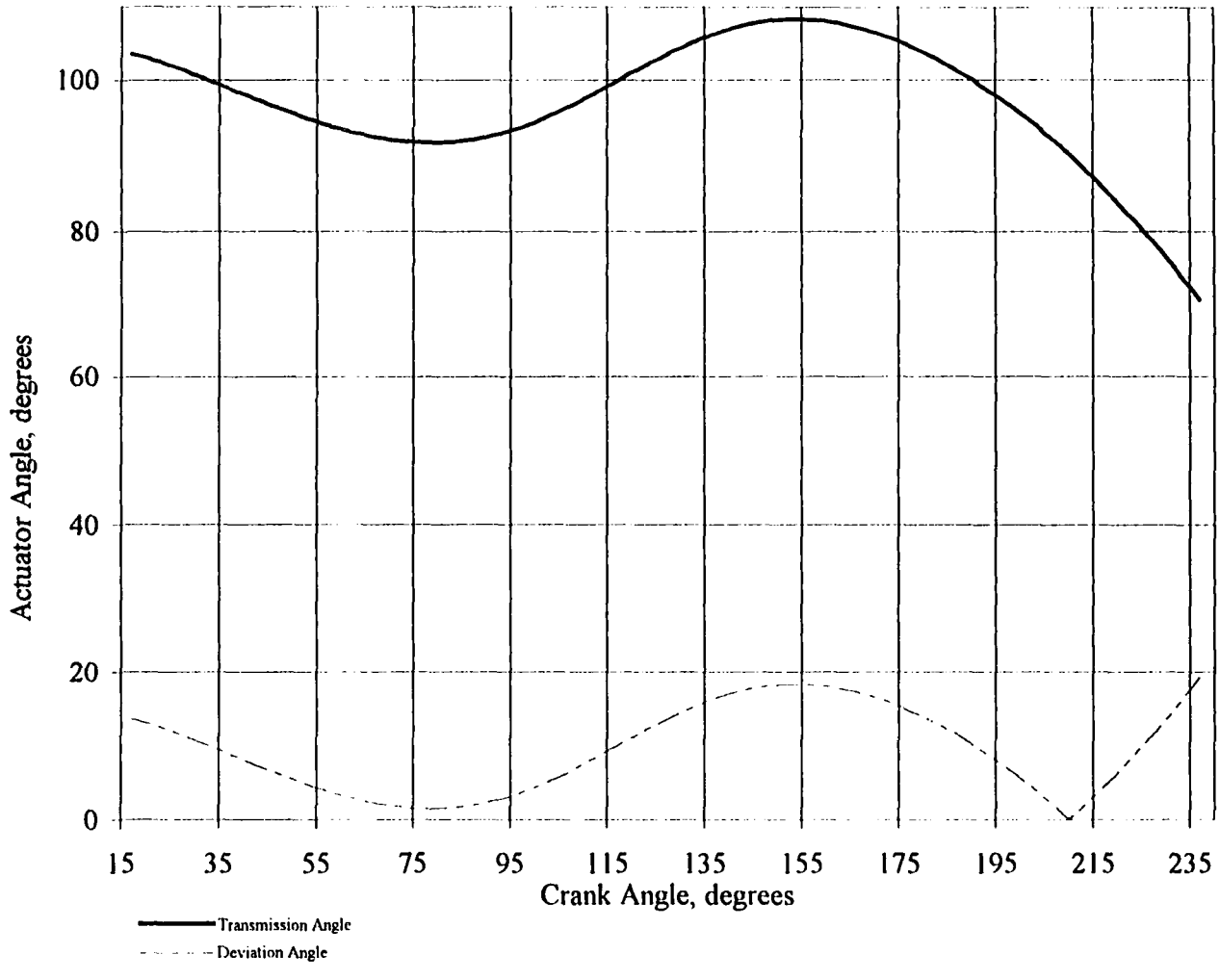

Figure 4.19: Transmission angle variation, mechanism 3

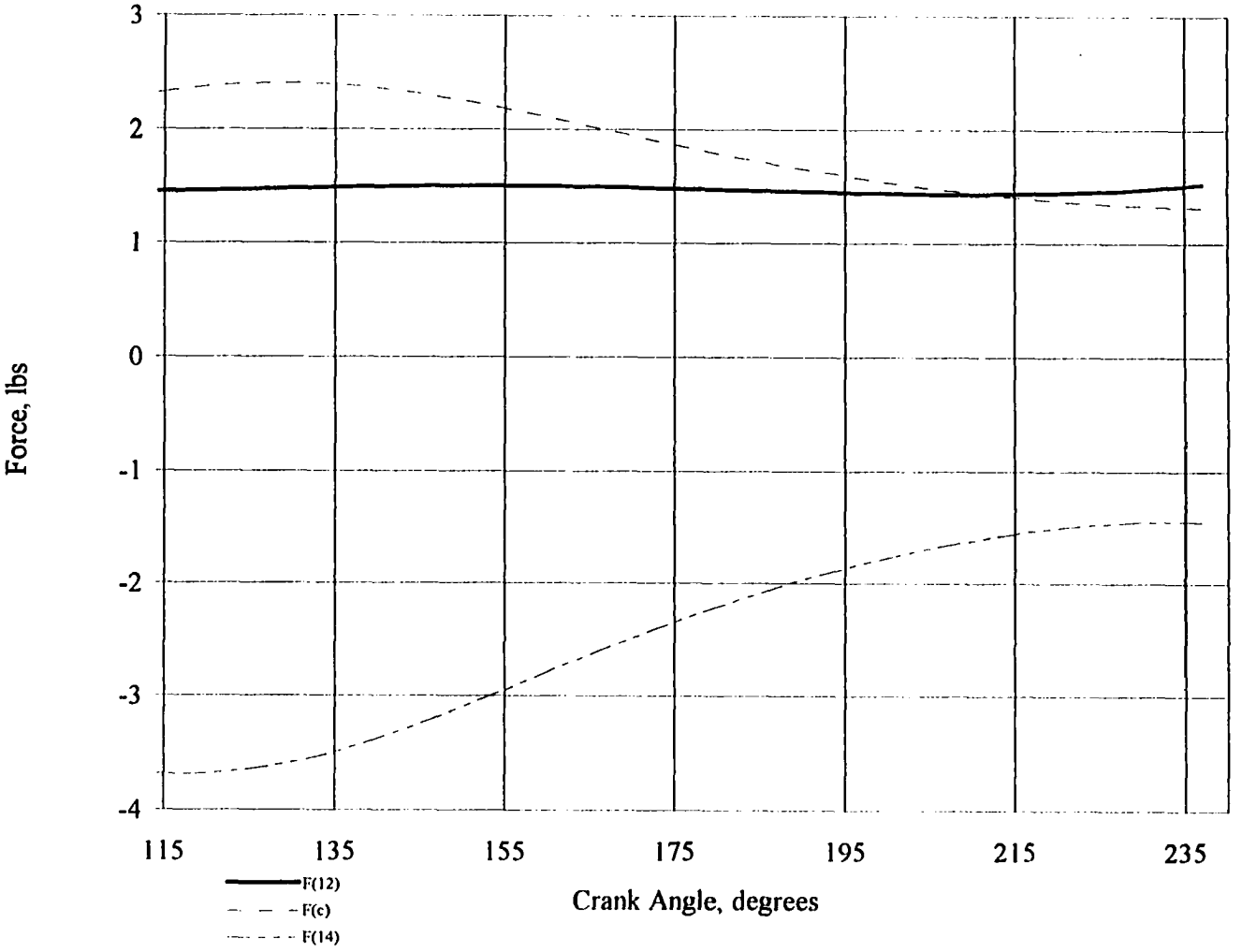

Figure 4.20: Forces acting on mechanism 3

Simulated annealing required 380 701 function evaluations for the 20 precision point problem and converged at a temperature value of 2.62E-7. The parameters required by the procedure were identical to those used for mechanisms 1 and 2. The objective function's value at convergence was 6.09E-2.

V. GENETIC ALGORITHMS

Genetic algorithms are numerical search procedures that emulate the evolutionary process. The development of genetic algorithms is widely attributed to John Holland at the University of Michigan. Holland's book, Adaptation in Natural and Artificial Systems [32] is considered a seminal work in the area of evolutionary programming, particularly with respect to genetic algorithms and their use. Holland points out that genetic algorithms are best used to perform preprocessing of a domain to search for optima occurring in that domain. However, genetic algorithms have been used successfully in optimization across a disparate spectra of fields. Wang and Chen [33] selected the optimal locations of elastic and rigid beams in order to maximize their fundamental natural frequencies using genetic algorithms. Champman and Jakiela [34] applied genetic algorithms to structural topology design problems as did Sandgren and Jensen [35], and General Electric put genetic algorithms to use in their design of the Boeing 777 jet engine [36]. Numerous other successful applications of Genetic Algorithms exist in fields as contrasting as biology and image processing [37].

David E. Goldberg points out in Genetic Algorithms in Search, Optimization and Machine Learning [37] that

> Theorists interested in optimization have been too willing to accept the legacy of the great eighteenth and nineteenth century mathematicians who painted a clean world of quadratic objective functions, ideal constraints and ever present derivatives. The real world of search is fraught with discontinuities, and vast multimodal, noisy search spaces. . . .

Genetic algorithms are used in this work as a result of experiencing some of the numerical proclivities Goldberg discusses to minimize a composite objective function made up of two parts; one part of the function enforces straight line motion on the mechanism while the second part minimizes deviation of the transmission angle from 90°. The driven member is required to produce an angular displacement of no less than 180° while maintaining a constant sense of output torque.

The terminology of genetic algorithms is taken directly from the study of genetics and evolution. A *population* is made up of binary representations of the various parameters of the optimization problem; each parameter is referred to as a *gene*. The *alleles* are the values or *states* of the genes. *Parents* from within the population of *genotypes* (the collective group of genes) are selected for reproduction based on their *fitness* relative to other members of the population. Following selection, the most fit members of the population exchange "genetic material" to produce a "child" that is more fit than either parent. Fitness is determined based on the effect each genotype has on the objective function. To preclude the event of becoming trapped in the domain near a local minimum, the process of *mutation* is also allowed to occur. Pseudocode for the process in its entirety is given below [38]:

```
BEGIN /* genetic algorithm */
        generate random population (generation 1)
        compute each population member's fitness
    WHILE NOT converged DO
    BEGIN /* produce offspring that form next generation */

        FOR population_size
        BEGIN /*reproduction cycle */
```

```
        select individuals from old generation for mating
        produce "children" from these individuals
        compute fitness of children
        insert offspring in new generation
        END

        IF convergence = TRUE
        converged = TRUE
    END

END
```

The population of genotypes is initialized randomly with each parameter represented as a binary string. The length of the string is determined by the level of precision required in the parameter. For example if the level of precision is 3 decimal places and the upper and lower bounds on the variable are 0.0 and 50.0 respectively, then to divide the length of [0..50] into 50*1000 equal parts the required number of bits is 16 since $2^{16} < 50*1000 < 2^{15}$. The remaining parameters, or genes, are converted to binary in a similar manner. (Issues associated with binary representation of problem parameters are discussed in Chapter 7, Summary and Conclusions). After the population is initialized, each member is ranked according to its fitness. The fitness is considered improved if the value of the objective function decreases in the case of minimization and increases in the case of maximization: penalties diminishing in either case. The population's fitness is calculated by summing individual fitness numbers. Each population member's probability of selection is determined as the ratio of individual fitness to population fitness as given by

$$p_j = \frac{f_j}{\sum_{i=1}^{population} f_i} \qquad [5.1]$$

Following the determination of the probability of selection of each member of the population, the cumulative probability, q_i, for selection is calculated. A random number on the interval [0..1] is generated and used to select the chromosome lying between $q_{i-1} < r < q_i$. Once this "roulette wheel" selection is complete, the recombination operator, or *crossover* is applied. Another random number on [0..1] is generated and compared to the probability of crossover, p_c, established by the user at the start of the algorithm. The expected number of population members to undergo crossover is the probability of crossover multiplied by the population size. Once the *mates*, or *parents*, are selected, another random number between 1 and m-1, where m is the number of bits on the gene group, is generated. This number will determine where the chromosome's chain is broken for recombination with another chromosome. Considering the 16 bit chromosomes below, recombination would occur as follows if the random number 12 were generated for a place on the chromosome for recombination to occur.

1 0 0 1 1 1 0 0 0 1 1 1 **0 0 0 1** to be recombined with 1 0 0 0 0 0 1 1 1 1 1 **1 0 0 1 1**

Crossover of these *parents* yields the *children* given below.

1 0 0 1 1 1 0 0 0 1 1 1 **0 0 1 1**

1 0 0 0 0 0 1 1 1 1 1 1 **0 0 0 1**

These new population members are now subjected to the probability of mutation, p_m. The probability of mutation when multiplied by the number of bits, m, and the population size, indicates the expected number of bits to be "toggled" during mutation. If selected, again based on a probability of mutation p_m, and comparison with a random number, a bit's

complement is exchanged for the bit if it is selected for mutation. Roulette wheel selection, crossover and mutation represent the activities of a single *generation*. Once evaluated for fitness, the process is repeated until convergence criteria are met.

The following example is taken from Genetic Algorithms + Data Structures = Evolution Programs by Michalewicz [39]. The function given below is mapped in Figure 5.1.

$$f(x) = x*\sin(10\pi*x)+1 \qquad [5.2]$$

It is observed this function contains many local optima on the interval x = [-1.00...2.00]. Michalewicz selects six places after the decimal as the precision required in x: this implies the interval [-1.00...2.00] should be divided into 300 000, or $[2-(-1)]*10^6$, equal parts. The smallest binary number required to capture six decimals of precision in x is 2^{22} (2^{22} = 4 194 304). The number of bits in the x gene is therefore 22. To convert x from binary to decimal the following formulation will be used.

$$\begin{aligned} x &= -1.0 + x'\left(\frac{3}{2^{22}-1}\right) \\ &= a + x'\left(\frac{b-a}{2^m-1}\right) \end{aligned} \qquad [5.3]$$

x' is the binary representation of x
a is the lower bound on the interval
b is the upper bound on the interval
m is the number of bits making up the gene

The population size for this problem will be set at eight; there are eight chromosomes, each with one gene, x. The first step will be to generate a random population. A random number

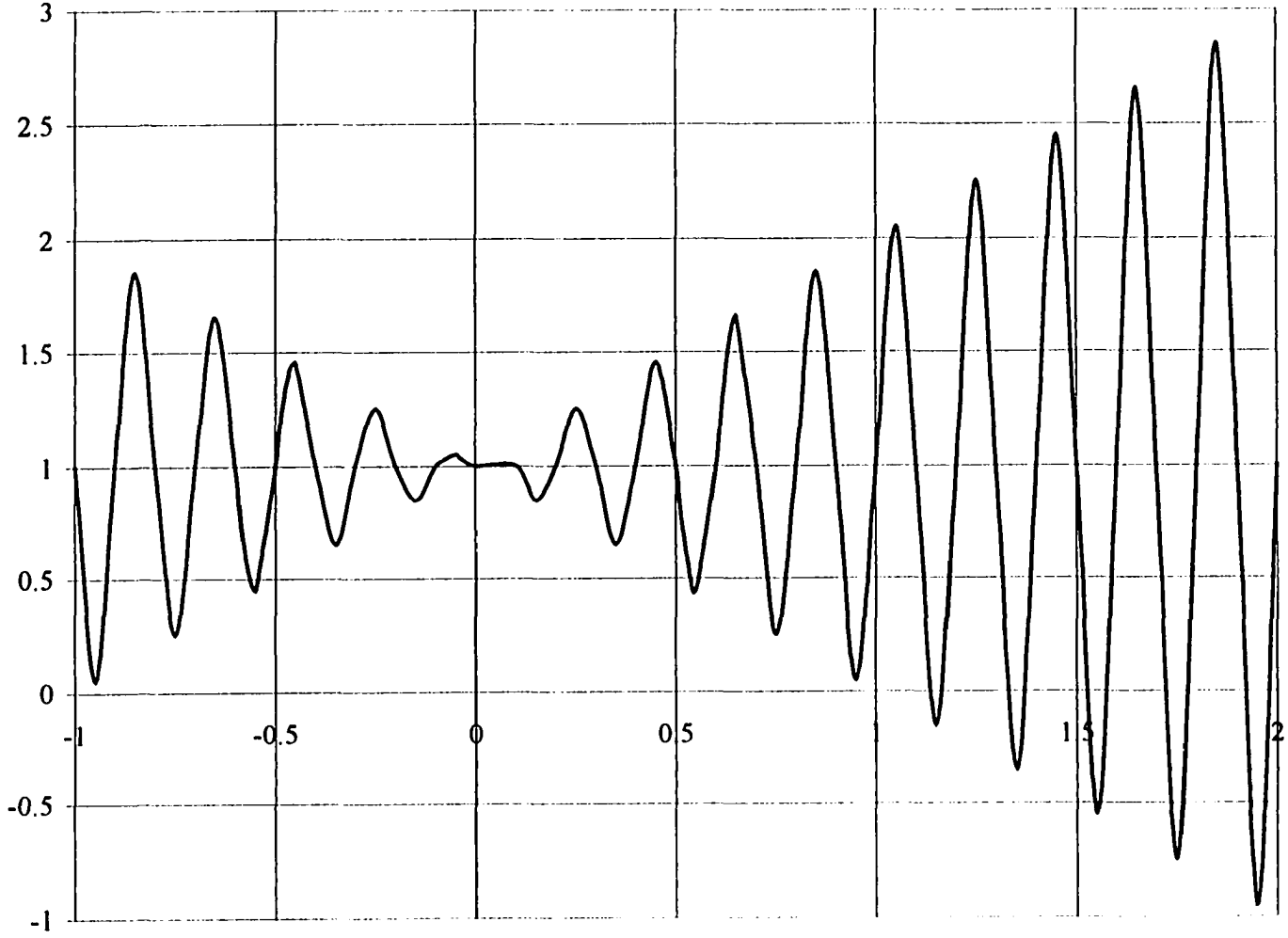

Figure 5.1: Example function

on the interval [0..1] is generated for each of the eight population members and they are treated as the normalized values of x. For example if r is 0.75, then the actual value of x is:

$$\begin{aligned} x &= -1+.75*[2-(-1)] \\ &= 1.25 \end{aligned} \qquad [5.4]$$

After determining the actual value of x, the fitness, f(x) is determined for each member of the population. Table 5.1 indicates the random, or normalized values of x, the respective binary representation, as determined by solving Eq. 5.3 for x', the actual value of x and each population member's fitness value, computed from Eq. 5.2.

Table 5.1: Random population and associated fitness - Generation 1

Random normalized population member (decimal)	**Random population member (binary)**	**x value**	**f(x), fitness**
.1317098	01100000100100101001O1	-.604871	1.092193
.4057039	01110111111101000001O0	+.217112	1.111174
.4100634	01111000010100110100l1	+.230190	1.187033
.4472208	01111011011111111000001	+.341662	0.669991
.9452150	10100101111111011101l1	1.835645	2.652121
.4668567	01111101001010111111l0	+.400570	1.007174
.7607986	10010110010000001001l0	1.282396	1.673626
.0867630	01011100101111001011O0	-.739711	0.298597

The fitness of the population is the sum of each population member's fitness in Table 5.1; the population fitness is therefore, 9.691909. Now the probability for selection of each member is determined by taking the ratio of the individual's fitness to the population's fitness. Equation 5.1 is used to calculate the probability of selection for each population member and Table 5.2 summarizes results.

Table 5.2: Probability of selection

Individual fitness	**Probability of selection**
1.092193	.112691
1.111174	.114650
1.187033	.122477
0.669991	.069129
2.652121	.273643
1.007174	.103919
1.673626	.172682
0.298597	.030809

The cumulative probability is now determined: it is the cumulative sum of the individual fitness values. Table 5.3 indicates the cumulative probabilities for each of the eight population members. Immediately following Table 5.3 is a demonstration of the use of these probabilities for selection of mating members. A random number on the interval [0..1] is again generated. The first value of the cumulative probability that is greater than the random number will determine which members of the population are selected for reproduction. Assume the following eight random numbers are generated:

.481903 .927673
.143312 .683109
.976353 .568225
.687414 .804817

Table 5.3: Cumulative probabilities

Population member, i	**Cumulative probability, q_i**
1	.112691
2	.227341
3	.349818
4	.418947
5	.692590
6	.796509
7	.969191
8	1.00000

The random number .481903 falls between cumulative probabilities q_4 and q_5: .418947<.481903<.692590. This indicates that population member five should be selected for possible recombination. The second random number .927673 falls between cumulative probabilities q_6 and q_7, so member seven is likewise selected. Continuing in this manner it is determined that the members selected for the new population are:

member 5, member 7, member 2, member 5, member 8, member 5, member 5, member 7

It is worth noting that member 5 is selected a total of four times and member 7 twice; this is expected as a result of the Schema Theorem which essentially ensures that the best chromosomes will be duplicated, the worst will die off and the average ones will stay approximately the same. The new population in binary is given in Table 5.4

Table 5.4: Population members considered for recombination

population member	binary representation
5	10100101111111101110111
7	10010110010000001001110
2	01110111111010000000100
5	10100101111111101110111
8	01011100101111001011100
5	10100101111111101110111
5	10100101111111101110111
7	10010110010000001001110

If the probability of crossover, pc, is assigned a value of 0.25, then it is expected that .25*8, or 2 chromosomes will undergo crossover. Likewise, if the probability of mutation is set to .01 and the total number of bits in the 8 member population is 176 (8 x 22), then approximately 2 bits will be toggled due to the mutation operator. Now eight more random numbers are generated and if any random number is < 0.25, the respective member of the population is selected for crossover.

.354000	.767835
.170838	.084101
.968773	.925236
.827876	.915969

From the numbers generated above, it is determined that population members 3 and 4 should be selected for crossover, or members 2 and 5 from the first generation. Now a random number on [1..21] is generated. This number will indicate the place on the chromosomes that is to be broken for combination with the mate. Assuming the random number 9 is generated on this interval the mates before recombination are shown below.

	Generation 1	Generation 2
01110111111110100000100	member 2	member 3
10100101111111101110111	member 5	member 4

The offspring of these parents are:

01110111111111101110111

10100101111110100000100

Now 176 random numbers, one for each bit in the entire population are generated. It was found that the 173 [rd] bit should be toggled by the mutation operator (the 173[rd] random number was .008534 and was the only random number generated less than .01). The new population is noted in Table 5.5

Table 5.5: Population members - Generation 2

population member	**binary representation**	**fitness values**
5	10100101111111101110111	2.652121
7	10010110010000001001110	1.673626
2''	**01110111111111101110111**	1.078014
5''	**10100101111110100000100**	0.679743
8	01011100101111100101100	0.298597
5	10100101111111101110111	2.652121
5	10100101111111101110111	2.652121
7''	10010110010000001001010	0.282541

The new population members, 2'', 5'' are the result of crossover and population member 7'' was created through the mutation operator on bit 173. Now the population's fitness is 11.968884 whereas in the first generation the population fitness was 9.691909. The process when continued as demonstrated would continually yield increased population fitness and ultimately identify the global optima at a value of x* =1.85, and the fitness, f(x*) =2.85.

Synthesis of mechanisms in this work is achieved using the commercial package Generator™ [40]. Minimization is accomplished using the following fitness function:

$$fitness = w_1\left[\sum_{i=2}^{4}\left|\Delta\theta_{c_i} - \Delta\theta_{c_{i-1}}\right|\right] + w_2\left[\sum_{j=1}^{4}\left|\frac{\pi}{2} - \mu_{T_j}\right|\right] + penalties \qquad [5.5]$$

Where: θ_{ci} is the slope angle of the coupler point path

$\Delta\theta_c$ is the *change* in the slope angle of the coupler point path

μ_{Tj} is the transmission angle

penalty is the penalty function used to determine direction of torque, T, on the driven crank

if $T_j > 0$ then if $Fc_j\left[Z_1 \sin\left(\theta_{c_j} - \theta_{1_j}\right) + Z_2 \sin\left(\theta_{c_j} - \theta_{2_j}\right)\right]\left[\frac{Z_1}{R}\right]dq_j$

$$penalty = \sqrt{\sum_{k=1}^{inactive\ constraints}\left\{F_{c_j}\left[Z_1 \sin\left(\theta_{c_j} - \theta_{1_j}\right) + Z_2 \sin\left(\theta_{c_j} - \theta_{2_j}\right)\right]\left[\frac{Z_1}{R}\right]dq_j\right\}^2}$$

w_1 is the weighting parameter placed on the objective of straight line motion

w_2 is the weighting parameter placed on the deviation of the transmission angle from 90°.

The population size is set at 20; the best 7 members of each population are kept in each generation; the probability of mutation is set at 5% of the population for 5% of the range of

each parameter's domain. Two point crossover is implemented in Generator™ and is widely acknowledged as being superior to single point crossover [37, 41]. The weightings on the composite objective function are 1.5 for w_1, and 1.0 for w_2. Coefficients of the two objectives figure prominently in the outcome of the mechanism's behavior. For instance, when the objectives are weighted equally, transmission angle deviations from 90° are minimized effectively, but the quality of straight line motion is poor. If the weighting on straight line motion is given an inordinately heavy weighting, say 100:1 , transmission angles exceed values acceptable to the figure of merit. Two gene groups, as shown in Fig. 5.2, are used in the synthesis of the first mechanism.

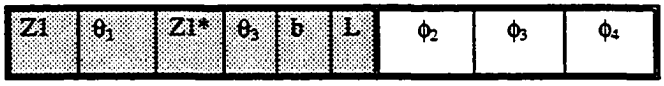

Figure 5.2: Gene groups used for kinematic synthesis

Each gene was normalized on the interval [0..1]. For example, the angular displacements of the crank, ϕ_j, were arranged in such a way that an order defect in the mechanism's path did not occur; i.e., the mechanism proceeds through the precision points 1, 2, and 3, not 1, 3, 2. If ϕ_1 were assigned a range between 0.00 and -60*π/180.00 radians, then ϕ_2's range might be assigned a range between -61*π/180.00 and -179*π/180.00 radians and ϕ_3's range from -181*π/180.00 to -190*π/180.00 radians. Normalizing on the interval [0..1] for each of the three genes makes their lower limits map to 0.00 and each of their upper limits to 1.00.

The penalty function is incorporated into the fitness function to ensure that the load torque is always in a direction opposite that of the driven member's angular displacement. The fitness function is penalized when the sign on the term $F_{c_j}\left[Z_1 \sin(\theta_{c_j} - \theta_{1_j}) + Z_2 \sin(\theta_{c_j} - \theta_{2_j})\right]\left(\frac{Z_1}{R}\right)dq_j$ becomes opposite that of the direction of motion, indicated by ϕj. The amount of penalty applied is a function of the distance from the constraint boundary. This type of penalty was found to be more effective in achieving minimization of the objectives in this application. Previous work in exploring genetic algorithms with penalty functions [42] supports this observation. Other research using penalty functions with genetic algorithms suggest schemes using traditional parabolic penalty functions [43,44] and some good results have been demonstrated using a stepwise approach to varying penalty function coefficients [45]. Equation 5.6 indicates the terms involved in the determination of the virtual work, previously derived in Chapter 2, Eqs. 2.1 - 2.9.

$$dq_j = \frac{-\sin(\theta_{3_j} - \theta_{1_j})}{\sin(\theta_{3_j} - \theta_{2_j})};$$

$$F_{c_j}\left[Z_1 \sin(\theta_{c_j} - \theta_{1_j}) + Z_2 \sin(\theta_{c_j} - \theta'_{2_j})\right]\left(\frac{Z_1}{R}\right)dq_j + T_j = 0 \quad [5.6]$$

$$\theta'_{2_j} = \gamma + \alpha_j.$$

Figure 5.3 represents a mechanism synthesized with Generator™ . Plots of the mechanism's behavior with respect to straight line behavior and resulting transmission angles and their deviation from 90° are also presented. Figures 5.4, 5.5, and 5.6 indicate the

mechanism's behavior with respect to the coupler path it generates, its path's variation from straightness, and the transmission angles at each position, respectively. Table 5.6 indicates the resulting forces on this mechanism, the crank's angular displacement, and the transmission angle at the 4 positions used in synthesis. A fitness value of 2.29 was achieved in 79 generations of Generator's™ run.

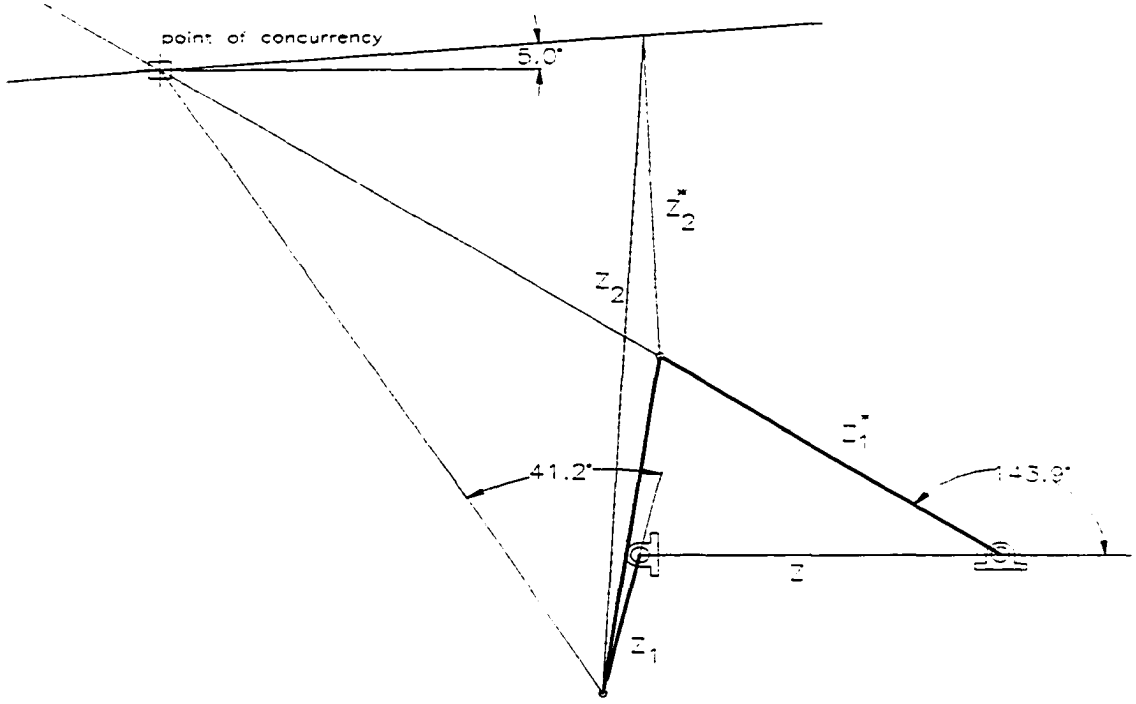

Figure 5.3: Mechanism 1 synthesized with Generator™

Table 5.6: Static Equilibrium Analysis; Output Torque = (+) 10.00 in-lbs

POSITION	ϕ_j	F_{12xj}	F_{12yj}	F_{14j}	F_{cj}	μ_j
j = 1	0.00°	0.7585	-1.3528	2.1504	0.9832	41.22°
j = 2	- 52.57°	0.2021	-1.0358	1.3992	0.8150	75.55°
j = 3	- 82.35°	-0.0695	-1.0196	1.1994	0.8064	90.39°
j = 4	-180.75°	-1.5461	-2.2152	2.0915	1.4409	157.77°

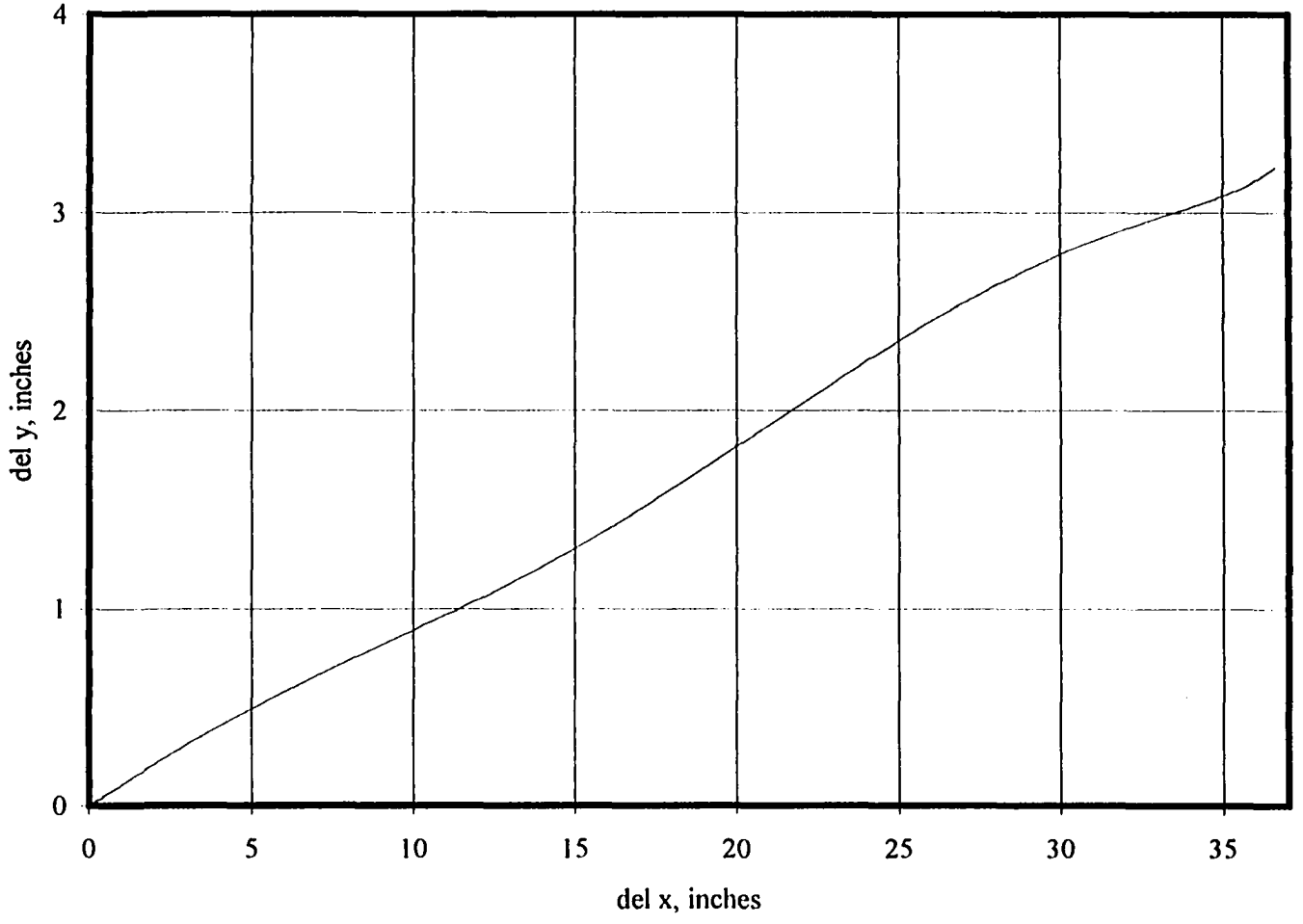

Figure 5.4: Coupler path mechanism 1

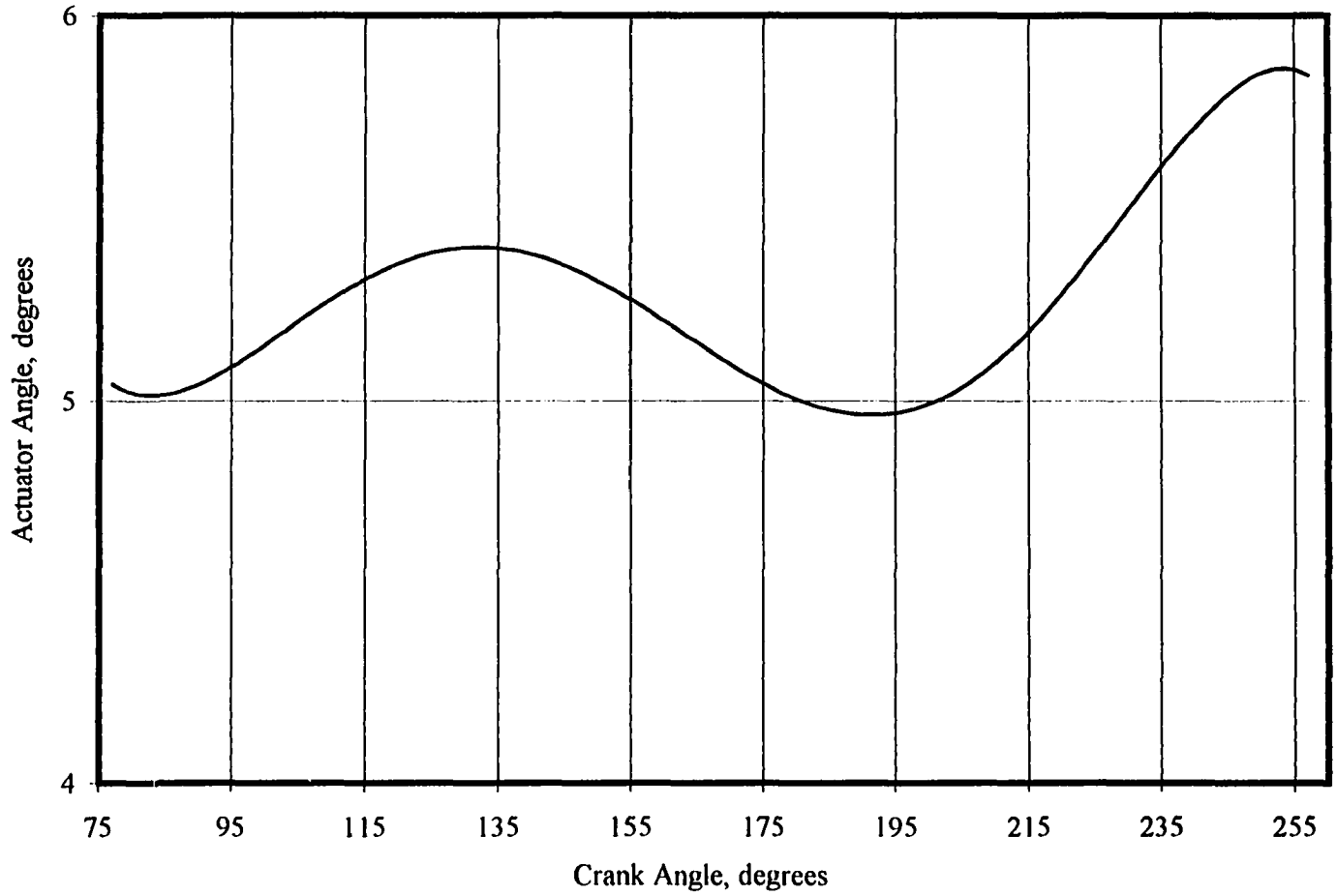

Figure 5.5: Mechanism 1 variation in actuator angle

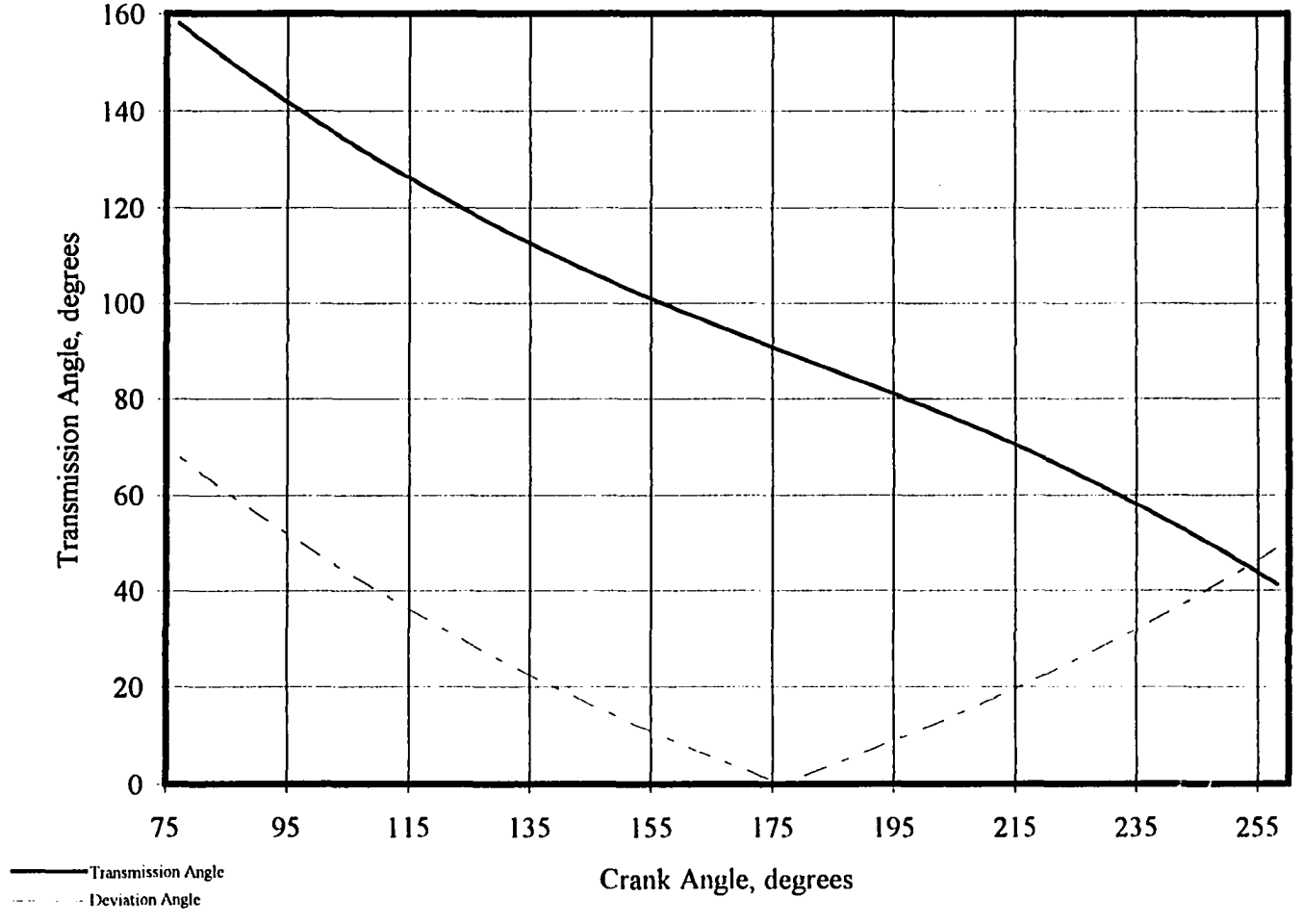

Figure 5.6: Mechanism 1 transmission angles

The actuating force, Fc_j, behaves as expected throughout the mechanism's motion: the magnitude increases with increasing deviation of the transmission angle from 90°. The objective of producing output torque in a consistent direction throughout the 180° angular motion of the driven link is likewise met with this mechanism.

Since the transmission angle in the above mechanism's final position is within 22.23° of 180°, another minimum was sought. Figure 5.7 indicates the modified gene group for the second approach to minimization. It should be noted that this gene group differs from the one previously identified in the following respects: a). the ground link, Z, and its orientation, θ, are included as genes and b). link Z_1's displacements, ϕ_j, are prescribed.

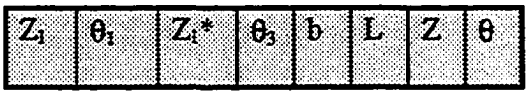

Figure 5.7: Modified gene group

Angular displacements of link member Z_1 are specified for eight points, yielding a 9 precision point problem. Transmission angles are limited to a range between 120° and 60°. Values of subsequent positions of the coupler point are accepted if less than a 3° deviation from a straight path occurred between precision points. The following equation indicates the relationship between the slope of the path generated by the mechanism and its precision points.

$$\theta_{c_i} = \tan^{-1}\left(\frac{Pt_{y_i} - Pt_{y1}}{Pt_{x_i} - Pt_{x1}}\right) \qquad [5.7]$$

The coupler points themselves, Ptx and Pty, are determined as noted in Eq. 5.8. Figure 2.4, Chapter 2, shows a generic linkage and its precision points.

$$\begin{aligned} Pt_{x_i} &= Z_1 \cos(\theta_1 + \phi_i) + b\cos(\gamma + \alpha_i) - L\sin(\gamma + \alpha_i) \\ Pt_{y_i} &= Z_1 \sin(\theta_1 + \phi_i) + b\sin(\gamma + \alpha_i) + L\cos(\gamma + \alpha_i) \end{aligned} \qquad [5.8]$$

Transmission angles are handled in much the same way; if the deviation of the transmission angle from 90° was less than or equal to 30°, parameters are accepted as more fit individuals of the population. Equation 5.9 expresses the formulation of the terms included in the fitness function under these conditions.

$$\begin{aligned} &if \left|\theta_{c_{i-1}} - \theta_{c_i}\right| \le \frac{3\pi}{180} \; then \left|\theta_{c_{i-1}} - \theta_{c_i}\right| = 0 \\ &if \left|\mu_{T_i} - \frac{\pi}{2}\right| \le \frac{30\pi}{180} \; then \left|\mu_{T_i} - \frac{\pi}{2}\right| = 0 \end{aligned} \qquad [5.9]$$

The fitness function itself is given below.

$$f(x) = w_1\left[\sum_{i=1}^{n-1}\left|\theta_{c_{i-1}} - \theta_{c_i}\right| + \left|\max(\theta_{c_1} \ldots \theta_{c_n}) - \min(\theta_{c_1} \ldots \theta_{c_n})\right|\right] + w_2 \sum_{i=1}^{n}\left|\mu_{T_i} - \frac{\pi}{2}\right| + penalties \qquad [5.10]$$

The penalty function, a function related to virtual work, remains the same as indicated by Eq. 5.6 and weightings are 1.5 for straight line motion, w1, and 1 for transmission angles, w2. With a population of 20 members, keeping the best seven, a mutation rate of 5% of the genes over 5% of the range, Generator™ found a minimum in 13 minutes, 34 seconds on a 486DX, 33 MHZ, personal computer with 8 MB of RAM. Table 5.7 summarizes the mechanism, and Figure 5.8 depicts the mechanism in its initial position.

Table 5.7: Kinematic parameters for mechanism 2

Z_1 Z_1/Z_1	θ_1	Z_1^* Z_1^*/Z_1	θ_3	b b/Z_1	L L/Z_1	Z Z/Z_1	θ	ϕ_n	θ_c
20.65 1.00	87.6°	44.44 2.15	106.2°	47.73 2.31	42.72 2.07	22.12 1.07	187.5°	-200°	36.1°

Figures 5.9 and 5.10 depict the coupler path and transmission angle variation with crank angle, respectively. The deviation from straight line behavior is depicted in Figure 5.11 and the forces required for static equilibrium at each position are noted in Figure 5.12. The mechanism exceeds the 3° band between crank angles of -71° and -51°. However the maximum value beyond the average of 36.1° was 39.3°. If the average value of 36.1° were to be lowered only slightly, say to 36°, the mechanism's path would fall between the allowable deviation and transmission angles would not be significantly affected.

The final mechanism identified using genetic algorithms, mechanism 3, differs from mechanism 2 in that the transmission angle's deviation from 90° is restricted to occur within a 20° variance from 90°. Figure 5.13 shows the mechanism in its initial position. Table 5.8 summarizes the synthesized mechanism and Figures 5.14 and 5.15 indicate the behavior of the mechanism's coupler path and transmission angles with changing crank angle. Figure 5.16 represents mechanism 3's variation from straightness throughout its motion. The static forces imposed at each position are depicted in Figure 5.17.

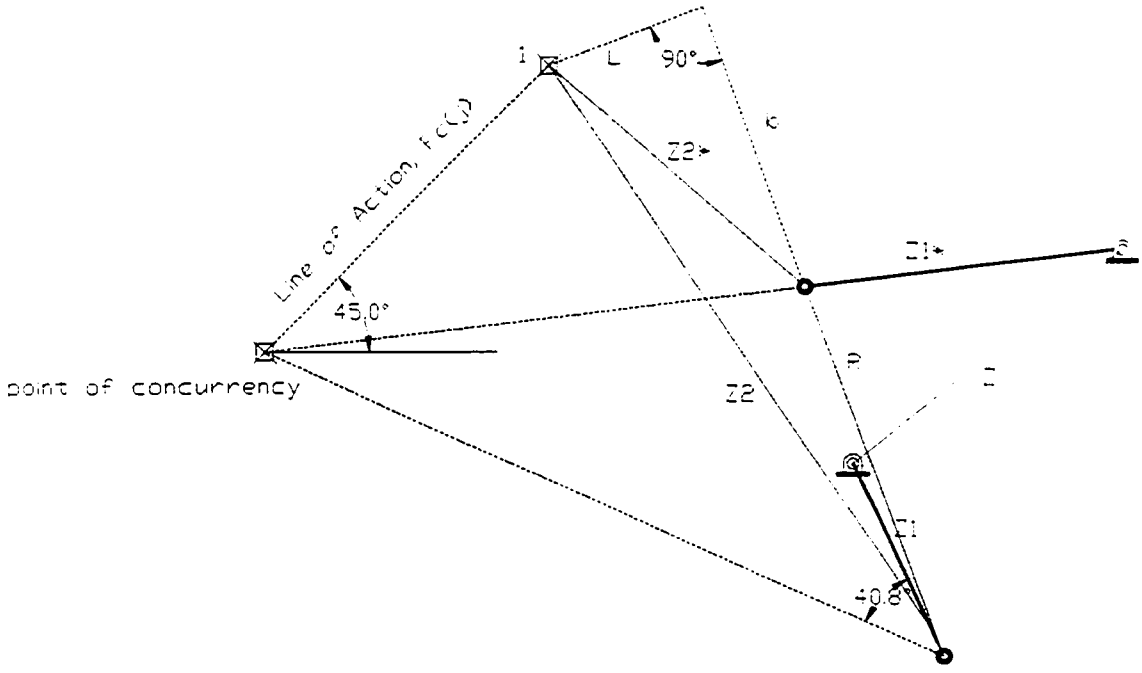

Figure 5.8: Mechanism 2 using new gene group, initial position

Table 5.8 : Kinematic parameters for mechanism 3

Z_1 Z_1/Z_1	θ_1	Z_1* Z_1*/ Z_1	θ_3	b b/ Z_1	L L/ Z_1	Z Z/ Z_1	θ	ϕ_n	θ_c
17.87 1.00	149.2°	39.60 2.22	181.2°	37.88 2.12	42.12 2.36	18.09 1.01	252.9°	-200°	122.6°

The three mechanisms produced by the genetic algorithm of Generator™ exhibit varying degrees of quality of straight line motion and minimization of transmission angles values from 90°. Mechanism 1, although its deviation from straight line motion is slight, produces the poorest transmission angles as well as the least amount of angular output of the crank. Mechanism 2 produces acceptable transmission angles--the largest deviation from

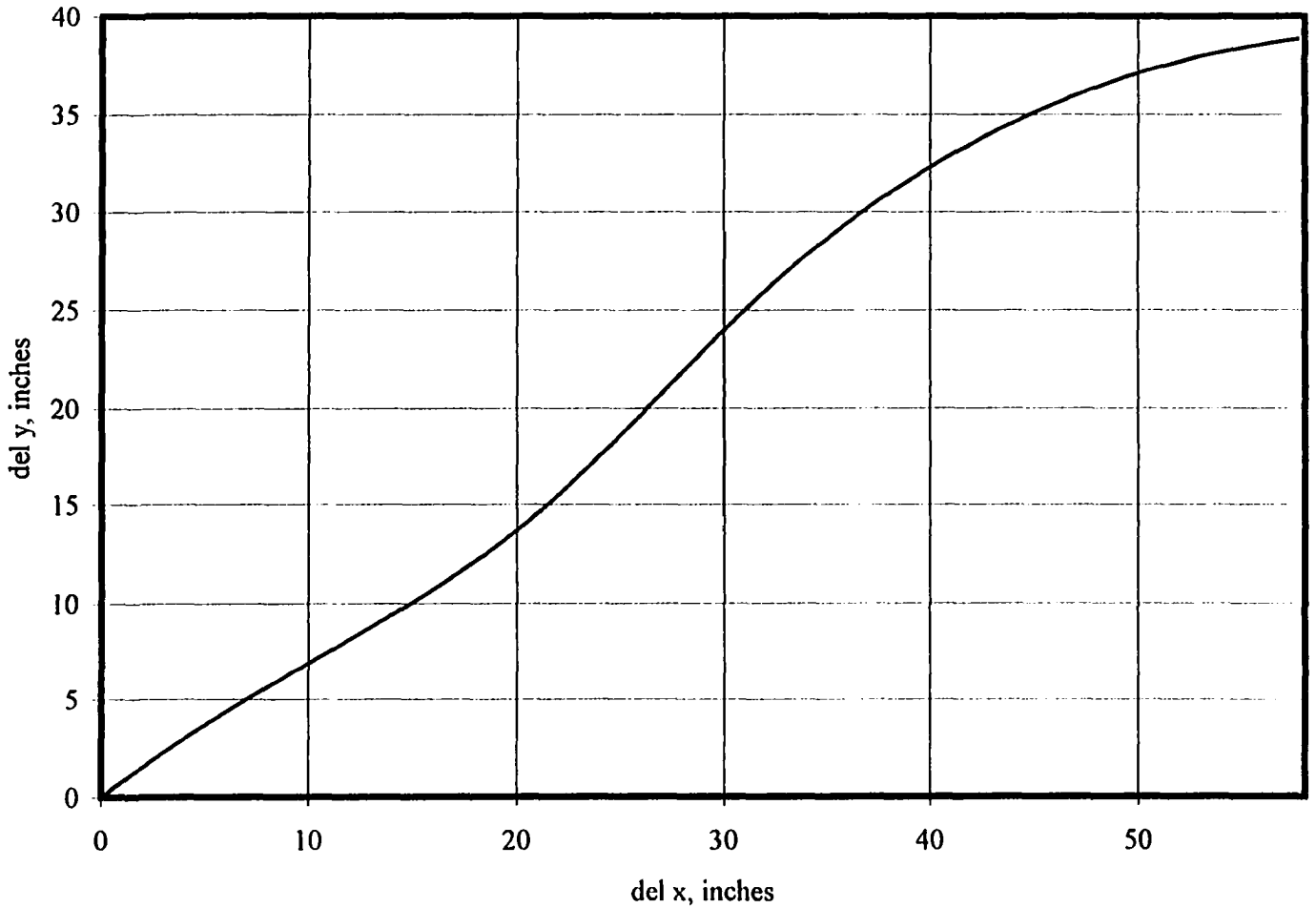

Figure 5.9: Mechanism 2 coupler path

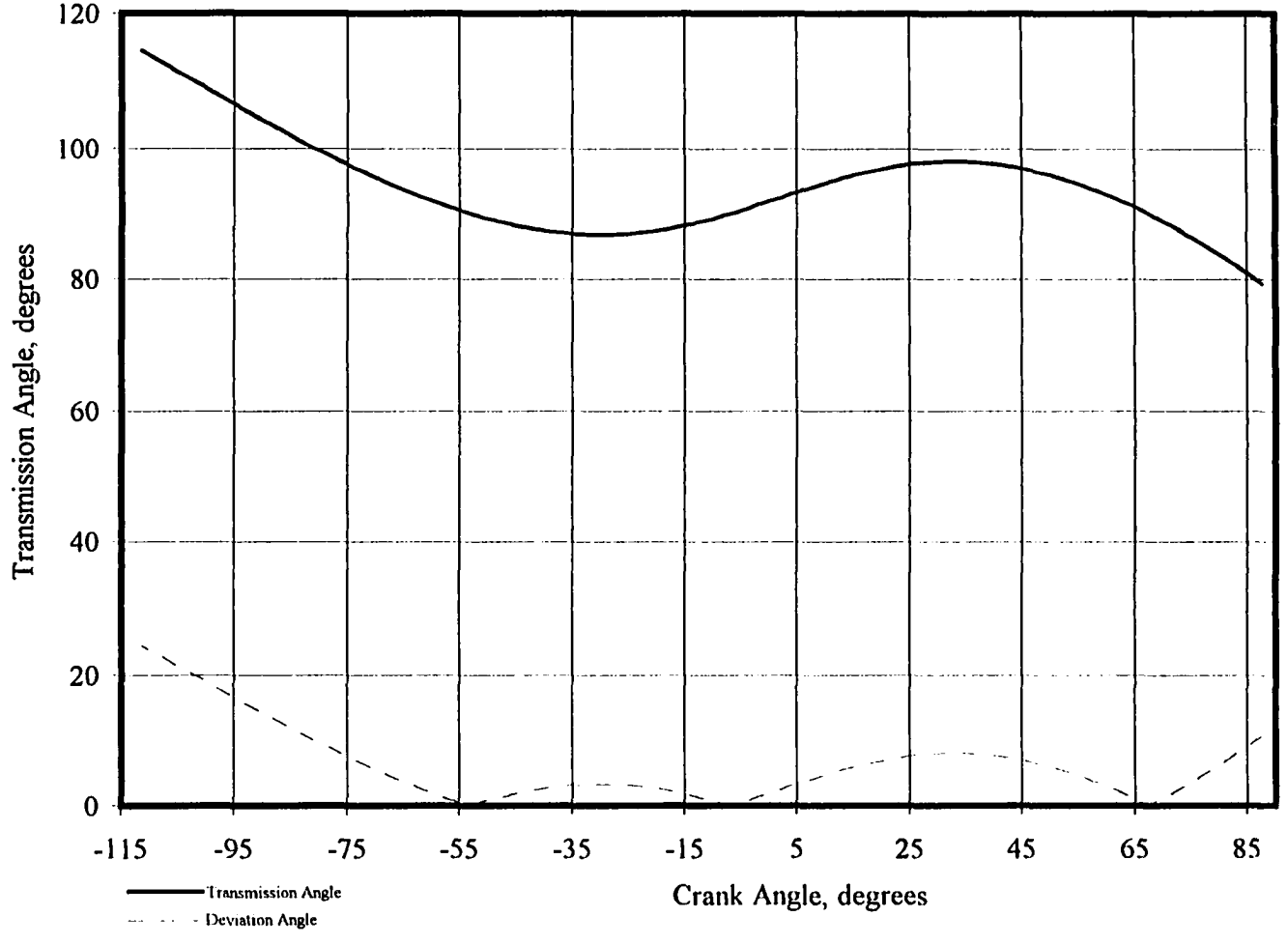

Figure 5.10: Mechanism 2 transmission angles

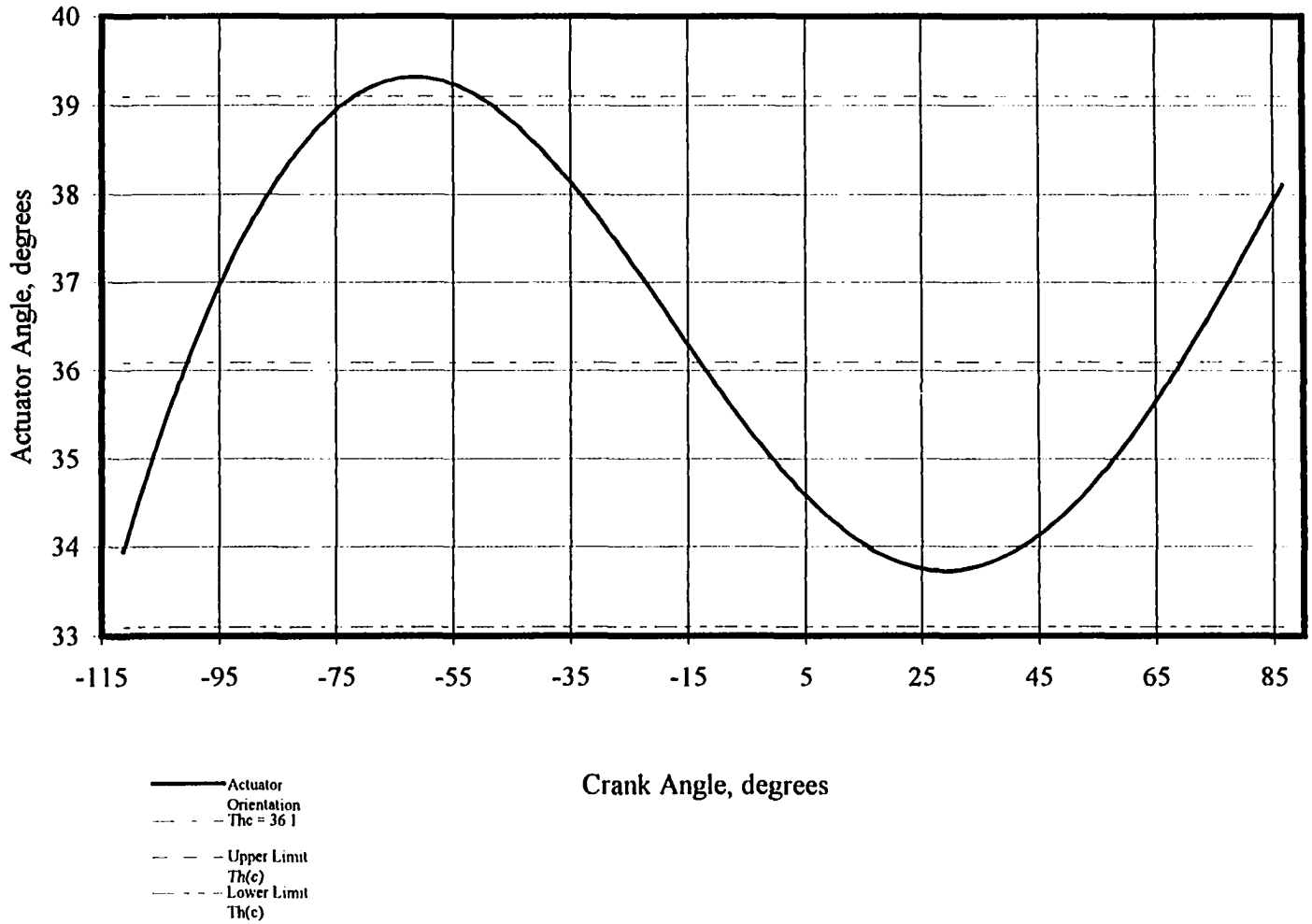

Figure 5.11: Mechanism 2 variation in actuator angle

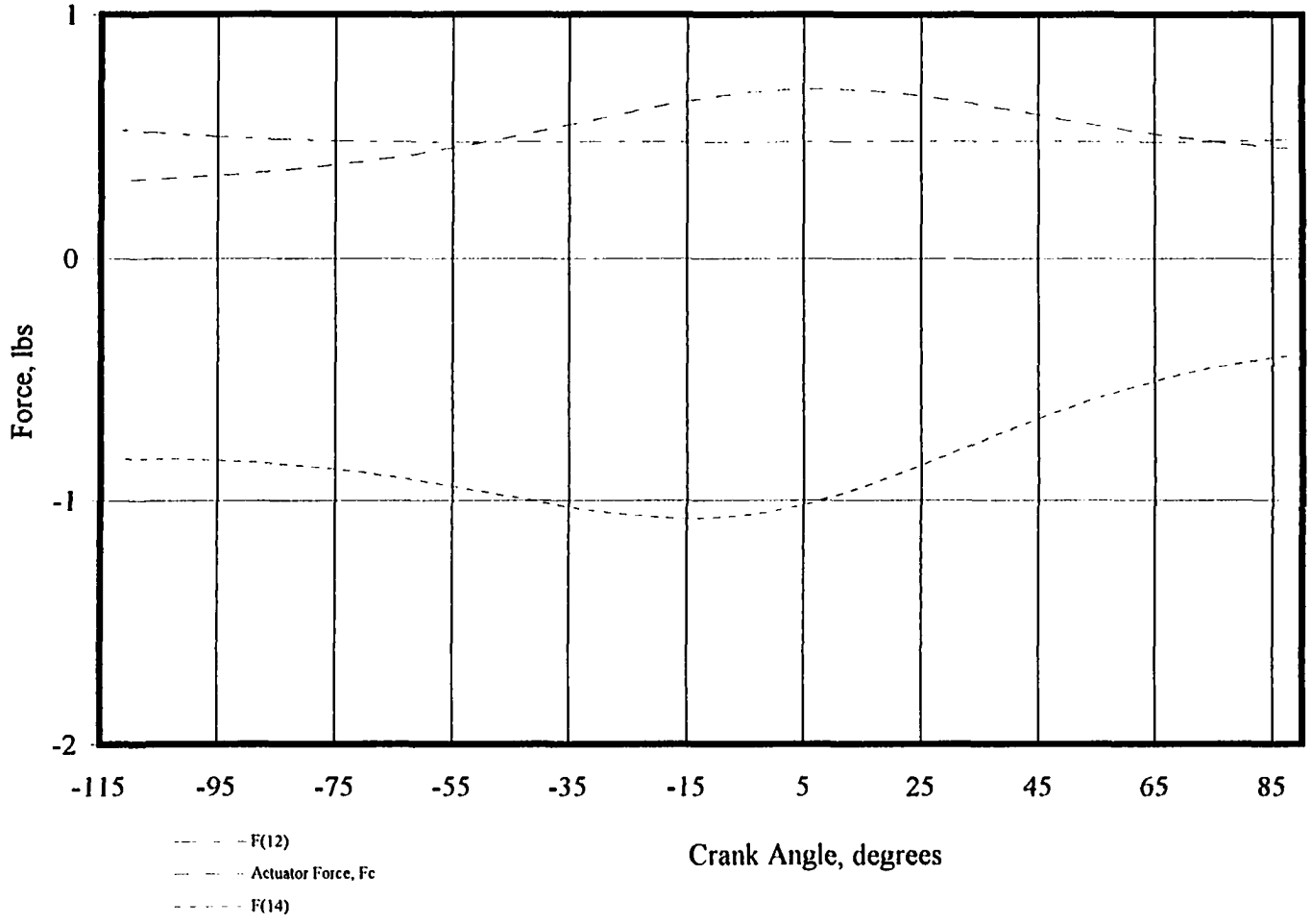

Figure 5.12: Forces imposed on mechanism 2

90° occurred at the last precision point and was 24.4 °. The characteristics of Mechanism 2's coupler path are relatively good with the exception of a departure of 3.2° from the average actuator angle. This departure from the specified 3° tolerance bands occurs over a 20° displacement of the crank. Mechanism 2 is the most effective of the three mechanisms synthesized with respect to force transmission, since the maximum deviation of the transmission angles from 90° is 15°. The coupler path of mechanism 3 meets the criterion of straightness; all points of the path remain within 3° of the average actuator angle. Although the transmission angles exceed the 20° band by at most 12° , this mechanism's overall quality is acceptable. The determination of acceptability is based on the fact that the criteria of approximate straight line motion is meet, the transmission angles, although exceeding the 20° band, are still relatively good (the highest value is 122°) and the crank oscillates a total of 200°.

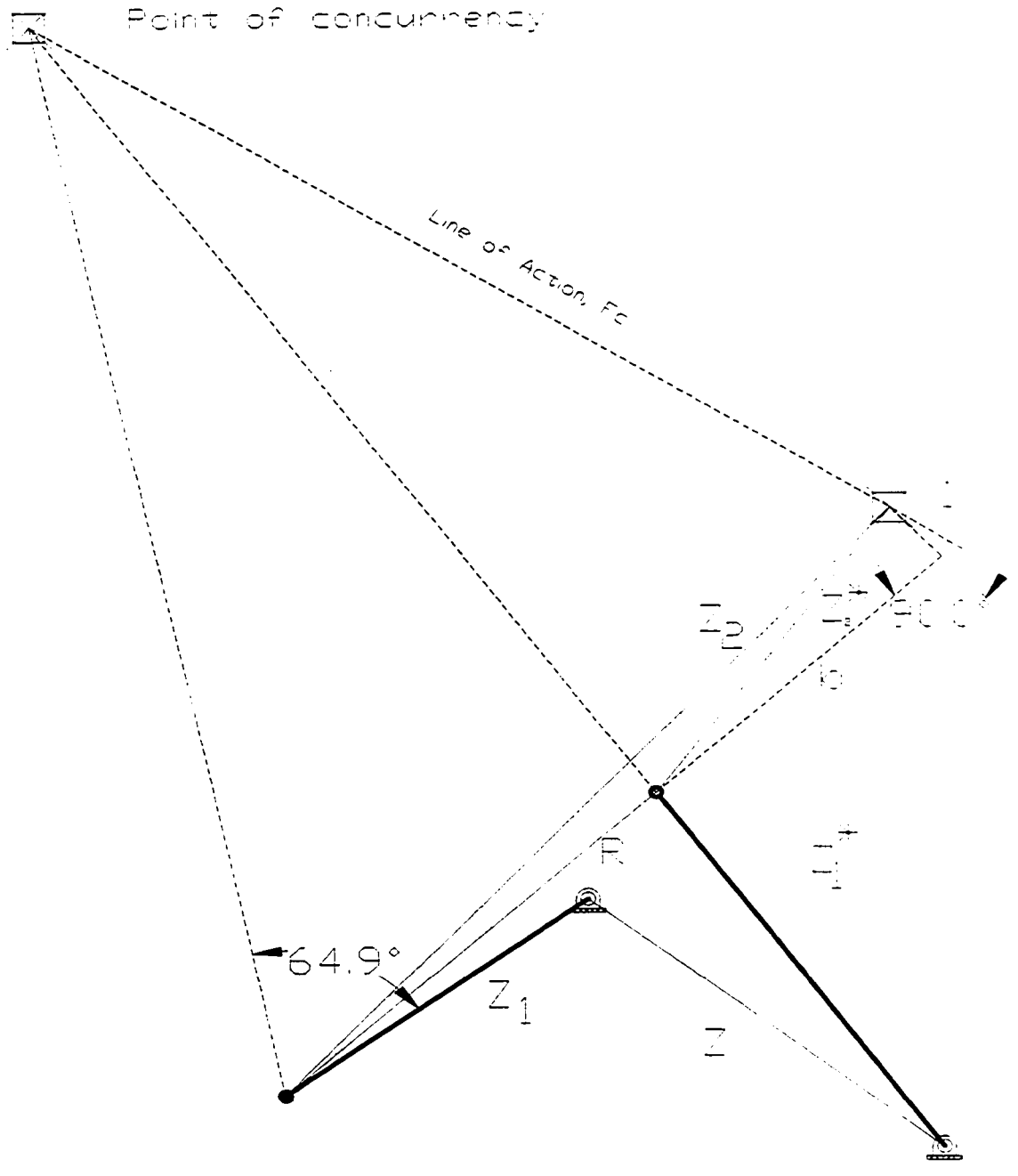

Figure 5.13: Mechanism 3, initial position

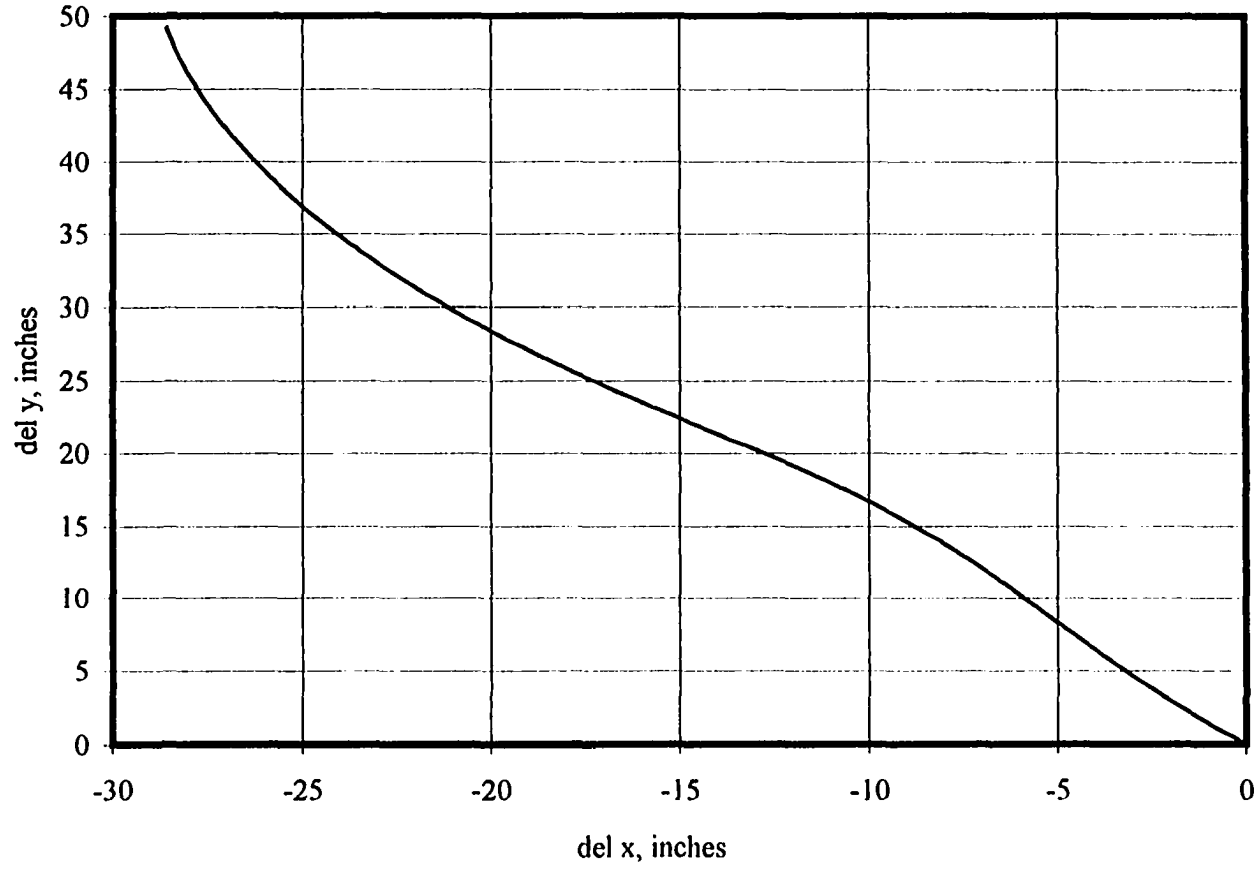

Figure 5.14: Mechanism 3 coupler path

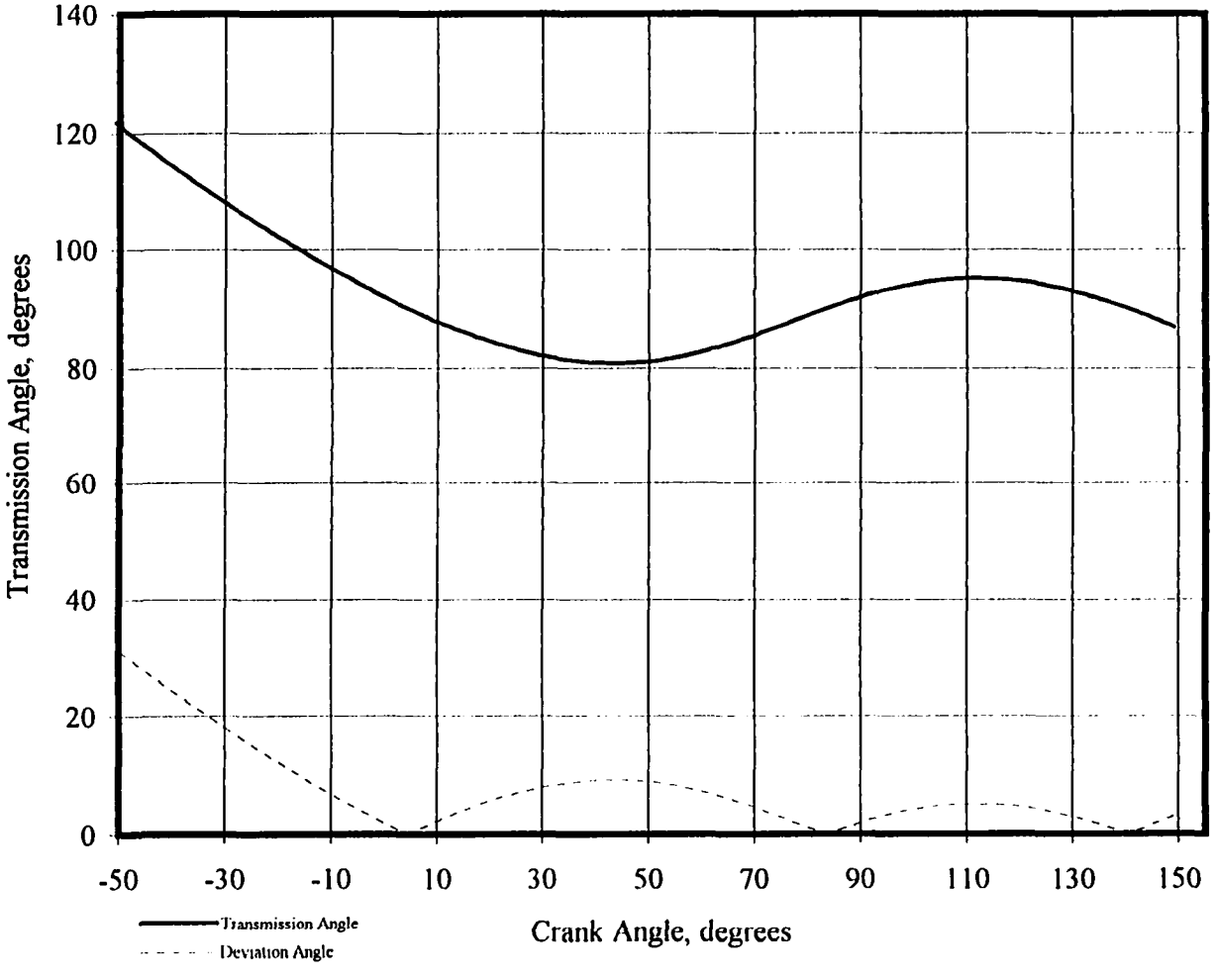

Figure 5.15: Mechanism 3 transmission angles

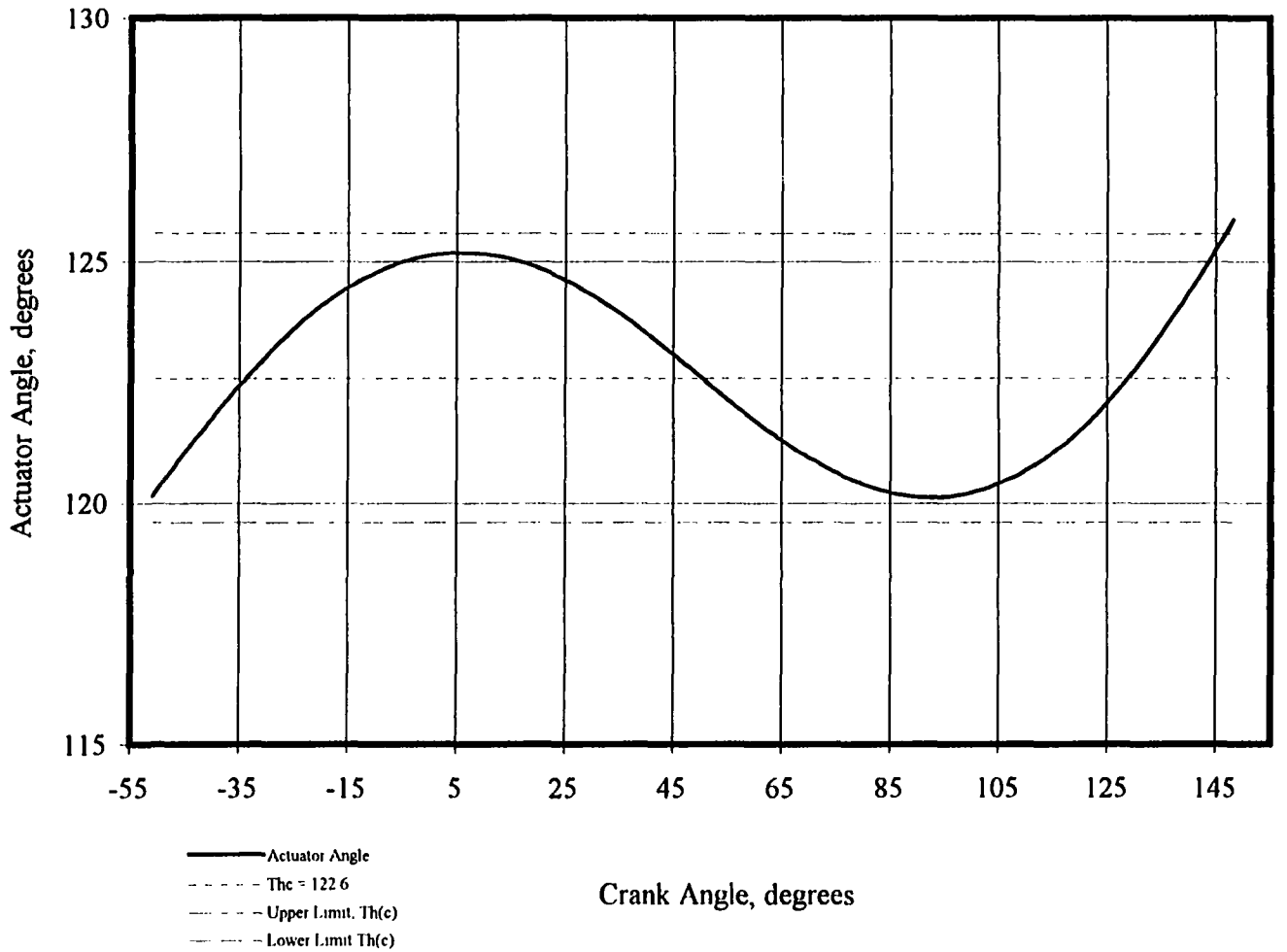

Figure 5.16: Mechanism 3 actuator angles

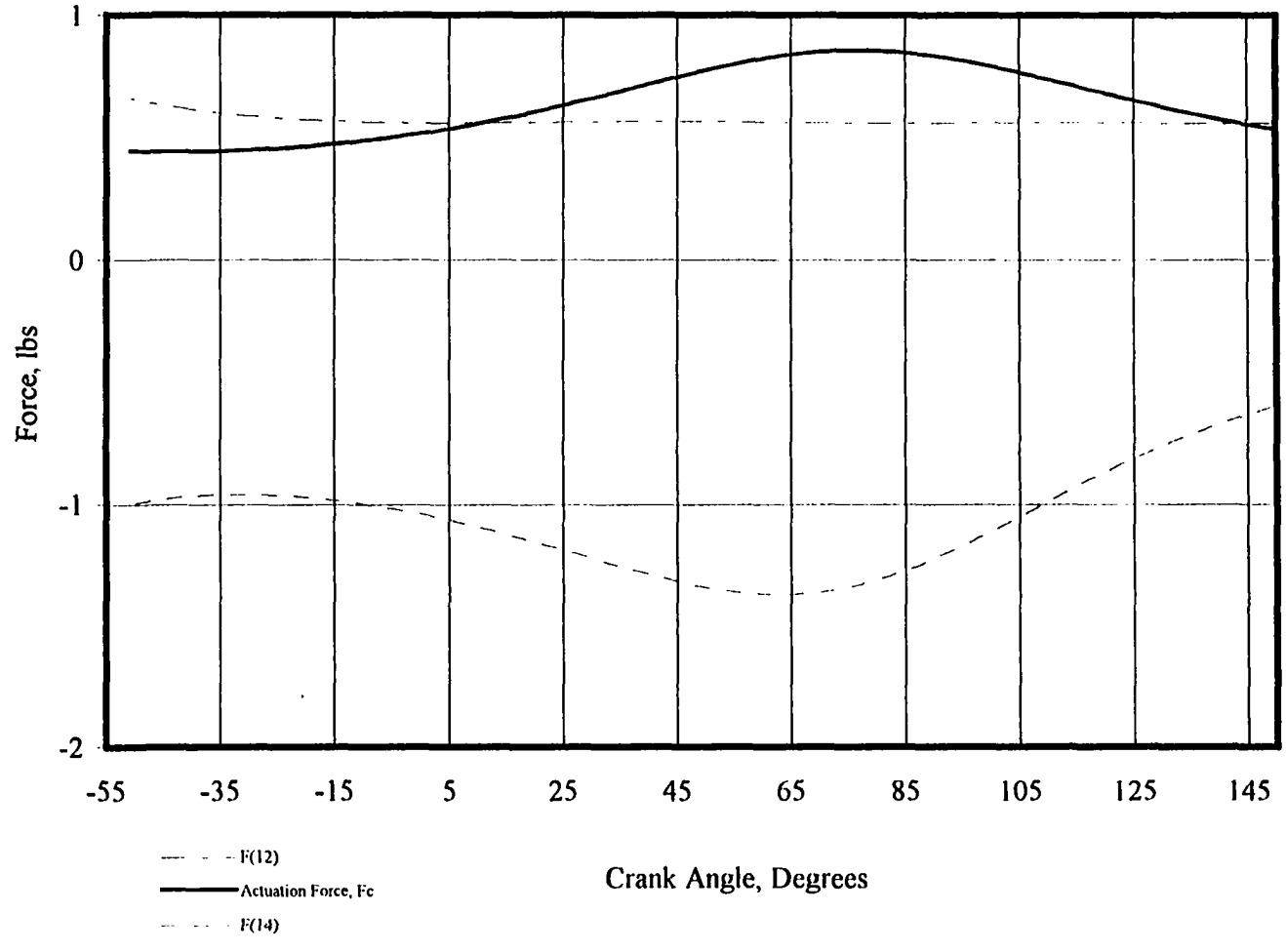

Figure 5.17: Forces imposed on mechanism 3

VI. A HYBRID APPROACH TO SYNTHESIS

This chapter examines results acquired by combining optimization procedures to synthesize four bar path generators with prescribed timing. The criteria sought are:

a) acceptable deviation of transmission angles from 90 ° throughout the mechanism's motion

b) a consistent direction for output torque throughout motion

c) at least 180° of angular displacement for the driven member (link Z_1)

d) minimal deviation from straight line motion

Although the comparison of genetic algorithms and simulated annealing should be made within the context of the problem they are being used to solve, it has been observed that in some cases, genetic algorithms are more effective in the early stages of a search procedure than simulated annealing [46]. Sirag and Weisser [47] proposed a unification of simulated annealing and genetic algorithms that involves expressing the reproduction and mutation schemes of genetic algorithms in thermodynamic terms. The authors use the fact that the success of genetic algorithms largely depends upon the diversity of a population in finding good solutions. The use of thermodynamic operators in their hybrid scheme was demonstrated to be efficient in providing such increased diversity. Murata and Ishibuchi [48] used a genetic algorithm-simulated annealing hybrid for flow shop scheduling problems and presented findings that supported their supposition that better solutions could be found using such an approach. Ingber [49] uses parameter sensitivities in a procedure known as Very Fast Simulated Reannealing and demonstrates (on a multimodal domain with

approximately 10^{20} local minima) that it vastly outperforms genetic algorithms. Ingber acknowledges that genetic algorithms are, however, competitive with the standard annealing procedure. The simulated annealing procedure implemented in this work is of the standard type, where the annealing schedule is based solely on a constant determined by observing its effect on the step sizes of the parameters.

Two hybrid approaches are explored. The first approach is one in which the genetic algorithm is used as a pre-processor of the domain and the simulated annealing algorithm is used to refine and further minimize the objective function. The genetic algorithm was initiated with a random population and allowed to proceed until the fitness did not improve. The parameter values of the last generation of the genetic algorithm were then pipelined to the simulated annealing procedure. (Nine precision points were used in each algorithm and the objective functions were the same in each case). In the simulated annealing procedure the value of the actuator's orientation, θc_1, was included among the unknowns. In the genetic algorithm, θc_1 was taken as the average of the values over the number of precision points. The average value of θc_1 was used in the genetic algorithm since convergence rates increased when θc_1 was not included among the unknowns. The deviation band around the actuator's orientation was limited to within 1° of the linear path the mechanism generates. The transmission angles were limited to values between 110° and 70°. Two executions of the simulated annealing program were made; the first with random initial guesses. The second execution of simulated annealing was made using the parameter values first determined by the genetic algorithm. The genetic algorithm was initiated with random

guesses. Objective functions were the same in each case. However, the penalty functions differed in that the simulated annealing algorithm used a parabolic penalty, premultiplied by a coefficient approaching infinity as the optimization progressed while the penalty function in the case of the genetic algorithm used distances from the constraint boundary of the inactive constraints.

The penalty function for the genetic algorithm follows.

$$\begin{aligned} &\text{if } T_j > 0 \text{ then if } F_{c_i}\left[Z_1\sin(\theta_{c_i}-\theta_{1_j})+Z_2\sin(\theta_{c_i}-\theta_{2_j})\right]\left(\frac{Z_1}{R}\right)dq_j>0 \\ &\text{penalty}=\sqrt{\sum_{k=1}^{\text{inactive constraints}}\left\{F_{c_i}\left[Z_1\sin(\theta_{c_i}-\theta_{1_j})+Z_2\sin(\theta_{c_i}-\theta_{2_j})\right]\left(\frac{Z_1}{R}\right)dq_j\right\}^2} \\ &\text{else} \\ &\text{penalty}=0 \end{aligned} \qquad [6.1]$$

In the case of simulated annealing the penalty function is:

$$\begin{aligned} &\text{if } T_j > 0 \text{ then if } F_{c_i}\left[Z_1\sin(\theta_{c_i}-\theta_{1_j})+Z_2\sin(\theta_{c_i}-\theta_{2_j})\right]\left(\frac{Z_1}{R}\right)dq_j>0 \\ &\text{penalty}=\sum_{k=1}^{\text{inactive constraints}}\left\{F_{c_i}\left[Z_1\sin(\theta_{c_i}-\theta_{1_j})+Z_2\sin(\theta_{c_i}-\theta_{2_j})\right]\left(\frac{Z_1}{R}\right)dq_j\right\}^2 \\ &\text{else} \\ &\text{penalty}=0 \end{aligned} \qquad [6.2]$$

The annealing routine, using initial random guesses, converged in 394 201 function evaluations and yielded an objective function value of 9.9049E-2. The simulated annealing procedure using a seed mechanism from the genetic algorithm, Hybrid 1, converged in 388 801 function evaluations and yielded an objective function value of 3.9742859E-2. A 149.4% reduction in the objective function's minimal value occurred with 5400 fewer function evaluations using the hybrid approach. Hybrid 1's values for the kinematic parameters follow in Table 6.1.

Table 6.1: Hybrid mechanism 1 using GA as a precursor to SA

Z_1 Z_1/Z_1	θ_1	Z_1* Z_1*/ Z_1*	θ_3	b b/ Z_1	L L/ Z_1	Z Z / Z_1	θ	θc
7.03 1.00	117.6°	43.91 6.25	134.6°	8.04 1.14	21.85 3.11	8.07 1.15	209.2°	64.9°

Table 6.2: Mechanism found with SA using a random guess

Z_1 Z_1/Z_1	θ_1	Z_1* Z_1*/ Z_1	θ_3	b b/ Z_1	L L/ Z_1	Z Z/ Z_1	θ	θc
16.68 1.00	113.2°	47.89 2.87	121.6°	32.47 1.95	36.99 2.22	19.13 1.15	206.3°	40.8°

Table 6.2 summarizes the mechanism found by using a random guess to initiate simulated annealing.

A comparison of the resulting mechanisms indicates that with respect to minimization of transmission angles, each of the procedures yielded mechanisms that performed within the 20° range around 90°. By meeting the objective of falling in the appropriate range of transmission angles, the contribution to the objective function's value at convergence of these two cases is exclusively from falling outside the 1° limit of the actuator angle. Various measures of the severity of this violation for each mechanism are summarized in Table 6.3. The hybrid mechanism is pictured in its initial position in Fig. 6.1 followed by the mechanism shown in four different positions (Figure 6.2). The variation in actuator angle for the Hybrid 1 mechanism using a GA-SA combination follows in Fig. 6.3. Figure 6.4 depicts the variation in actuator angle of the mechanism found using simulated annealing with a random

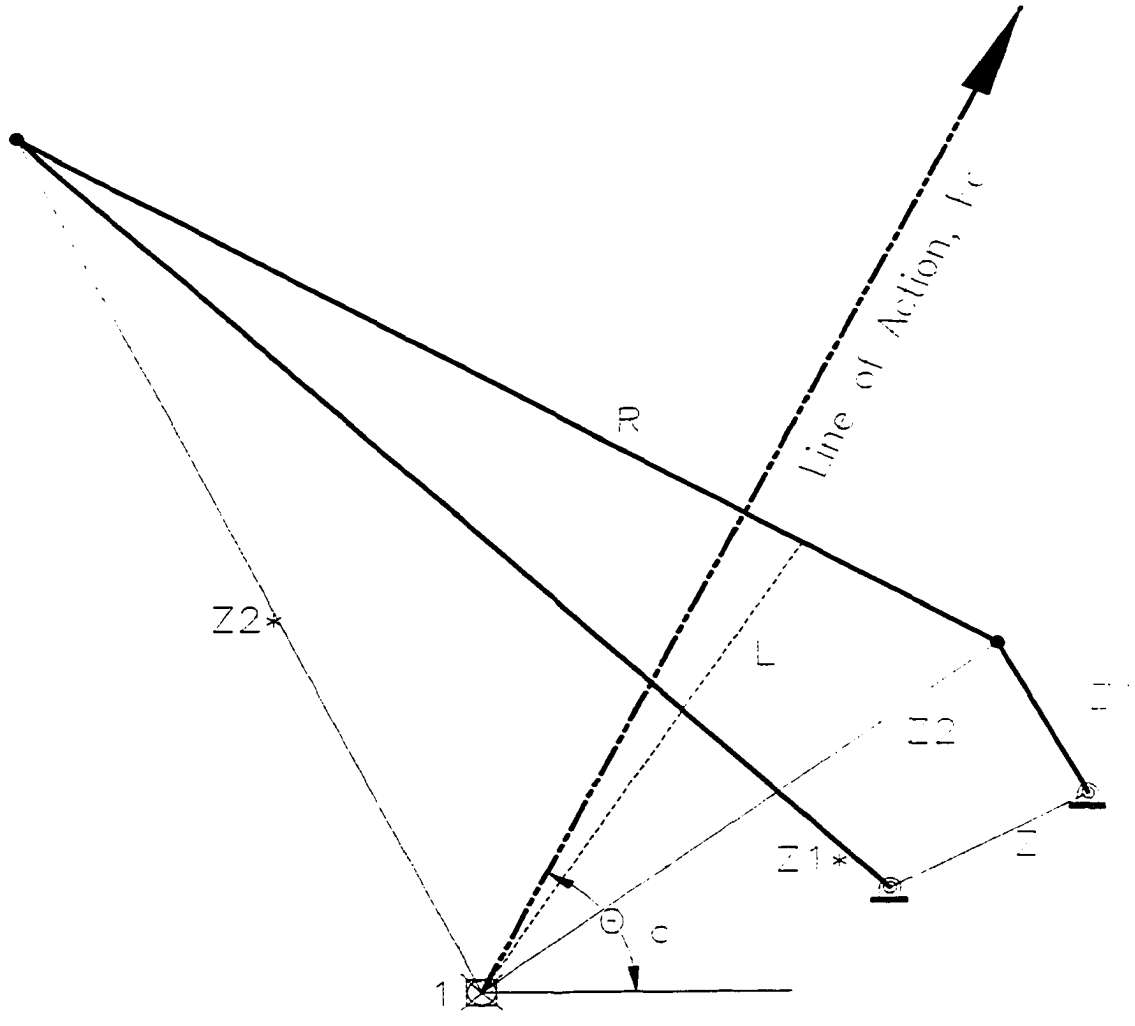

Figure 6.1: Hybrid Mechanism in initial position

start . The hybrid's coupler path is shown (Fig. 6.5) followed by the transmission angles resulting from this mechanism (Fig. 6.6).

Although it is not evident that the hybrid approach yielded a global minimum, it is clear that neither algorithm alone found better solutions than when coupled.

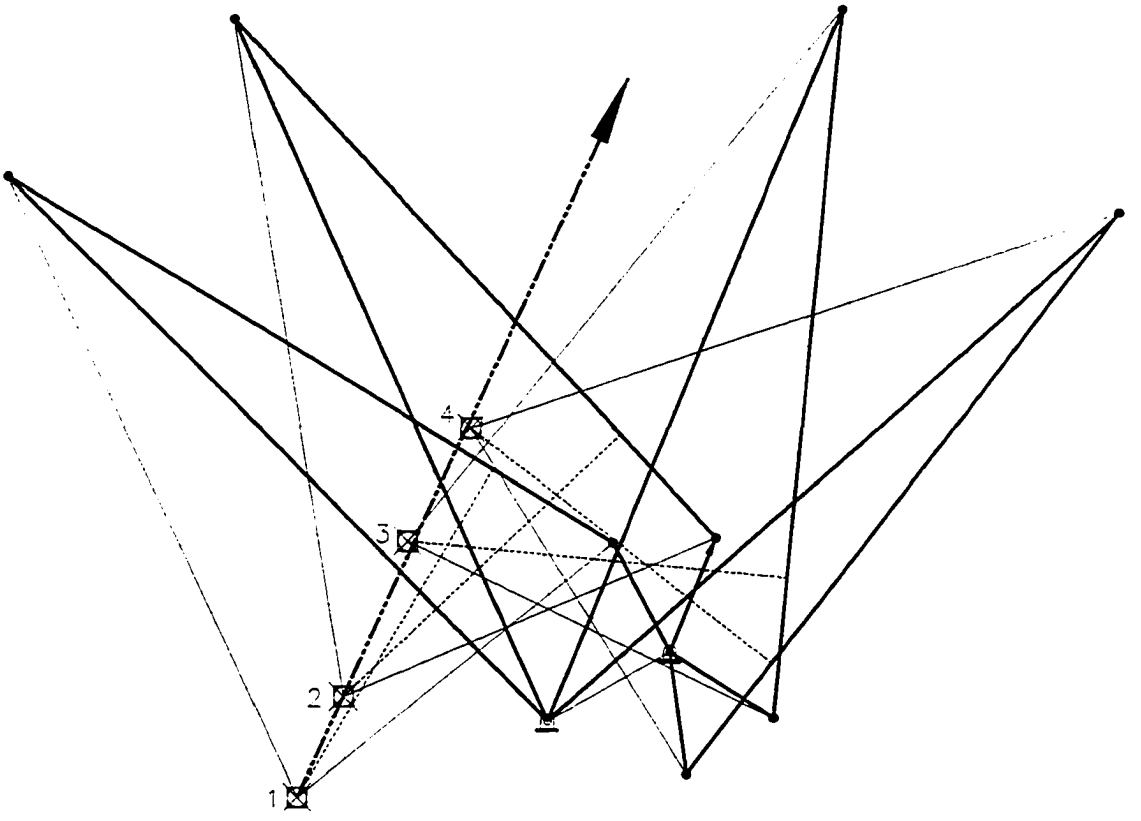

Figure 6.2: Four positions of first hybrid mechanism

Table 6.3: Measures of mechanism performance

Approach	θc_1	**Maximum over range**	**Minimum over range**	**Largest deviation**
Hybrid	64.9°	66.4° (+1.5°)	63.7° (-1.2°)	=max-min = 2.7°
S.A.	40.8°	43.2° (+2.4°)	39.7° (-1.1°)	= max-min = 3.5°

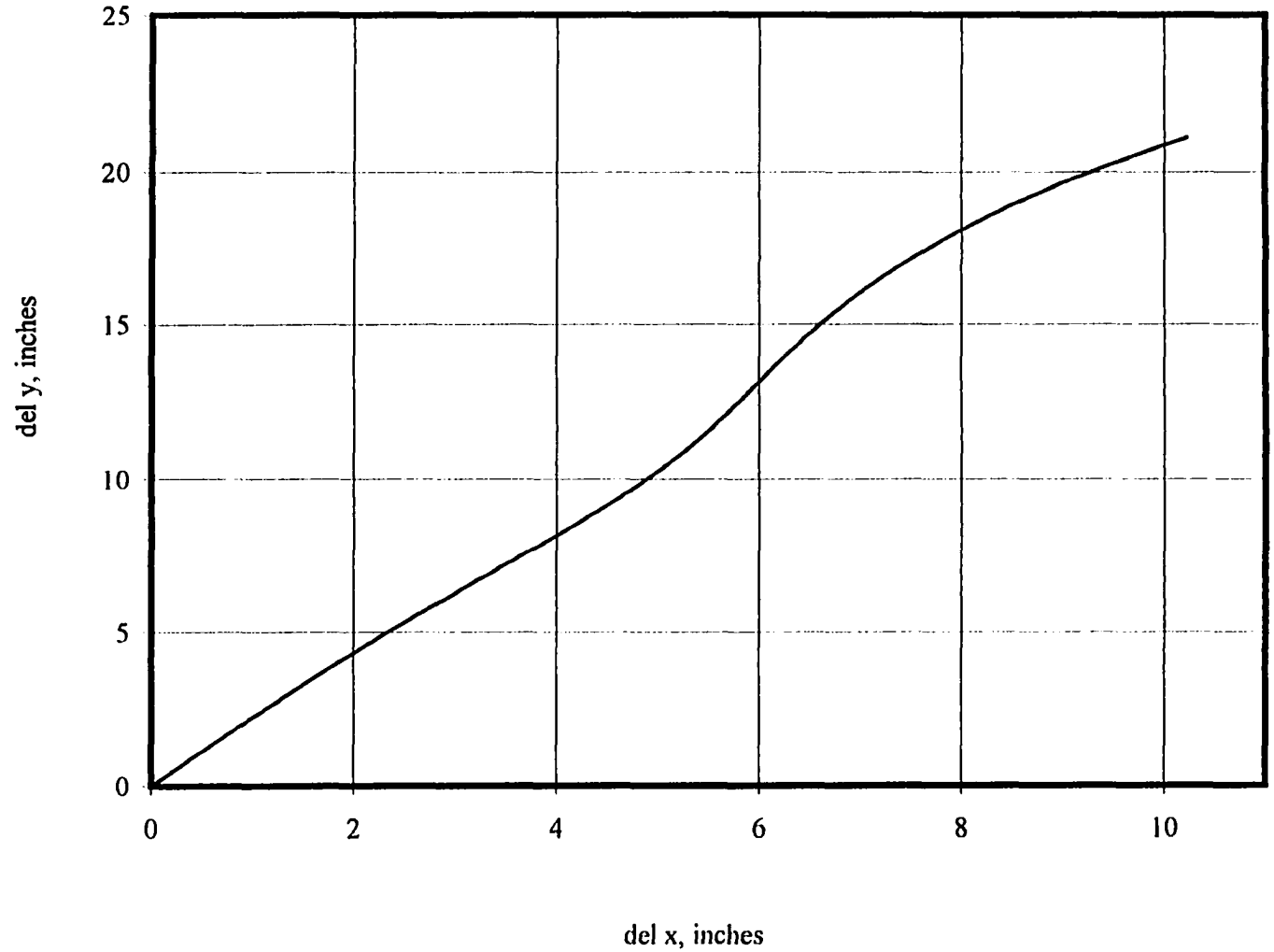

Figure 6.3: Coupler path hybrid 1

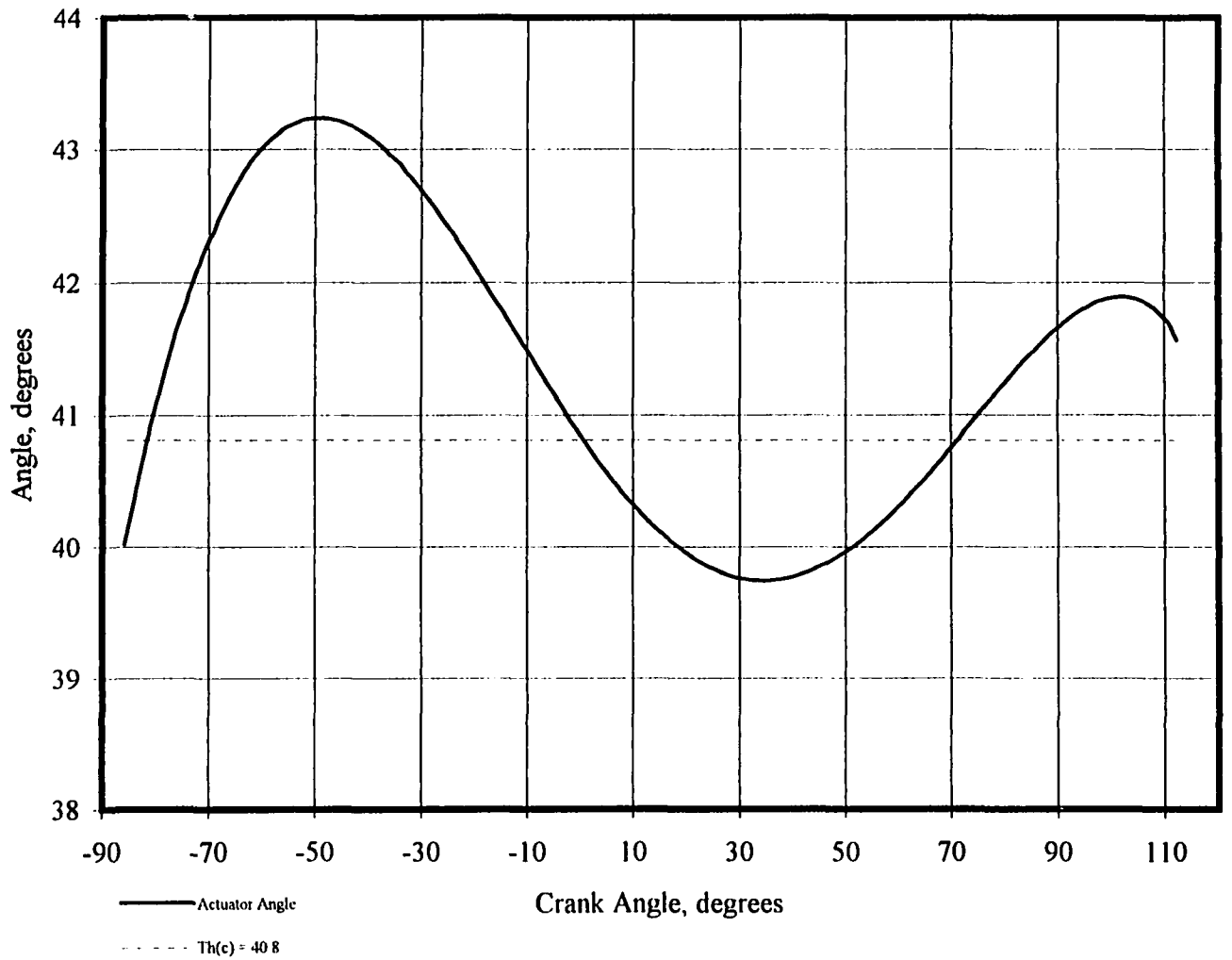

Figure 6.4: Random start, simulated annealing actuator variation

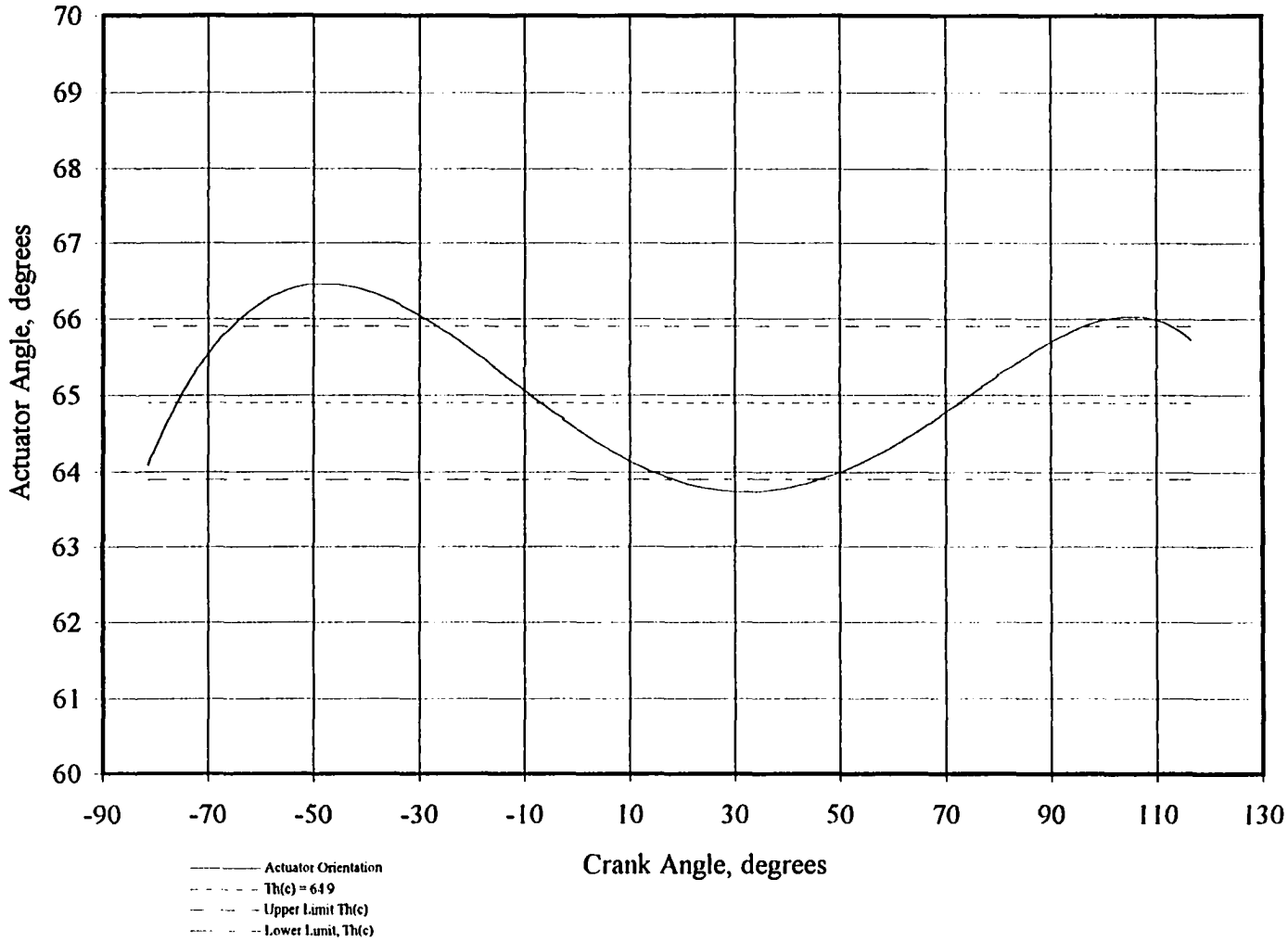

Figure 6.5: Hybrid 1 actuator variation

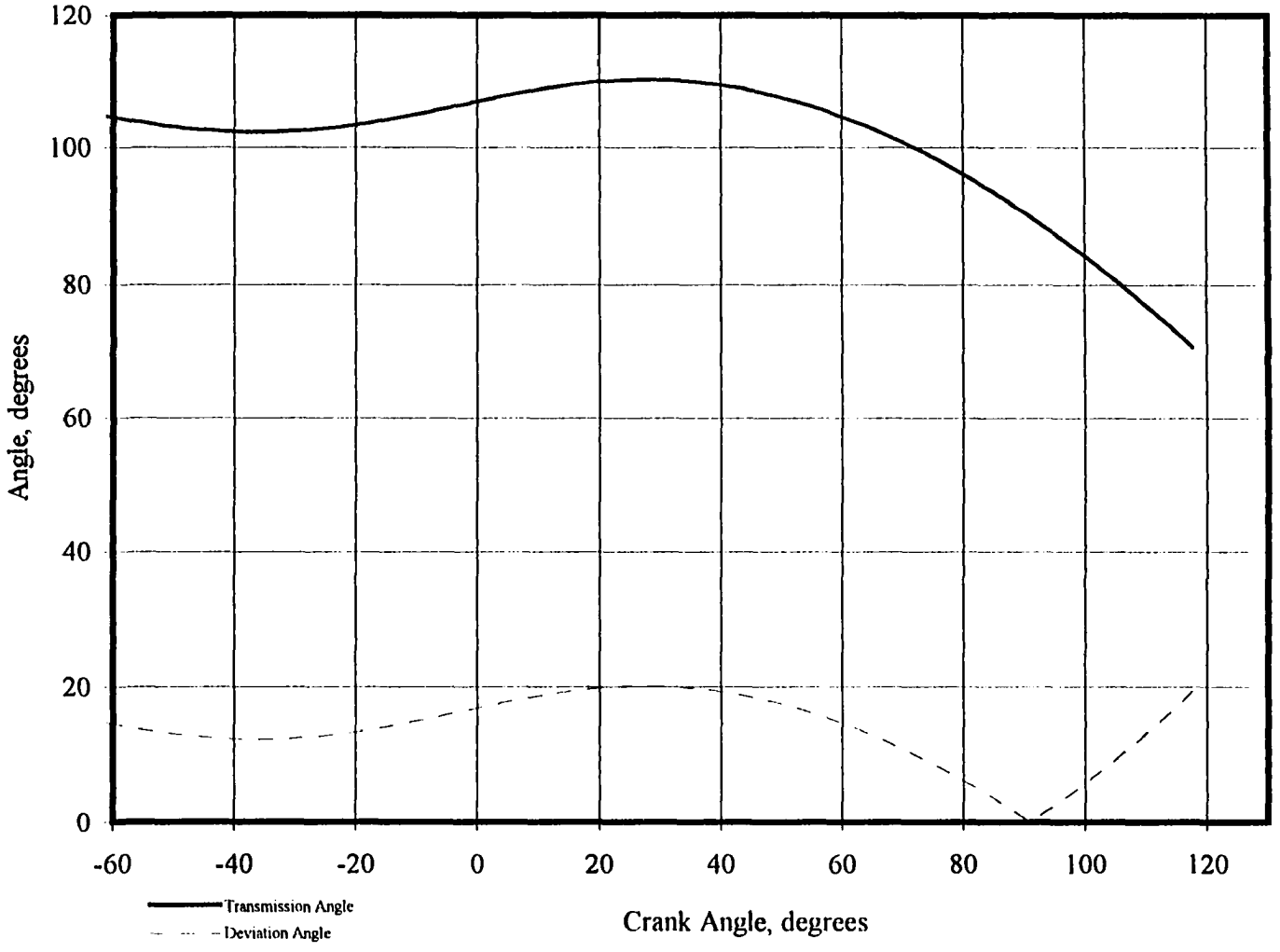

Figure 6.6: Hybrid 1 transmission angles

The second hybrid approach combined the GRG method with GA followed by SA. The mechanism identified in Chapter III using GRG was used as a start mechanism for both GA and SA simultaneously. The mechanism identified using GRG required that the transmission angles fall between 140° and 50° ; the exercise was not to find a better minimum for this criteria, but to improve upon the quality of the mechanism. Mechanism quality includes small deviations from straight line behavior, transmission angles near to 90° and optimal angular output of the crank. The designer is required to weight these criteria as deemed appropriate for the application. Simulated Annealing, starting with the mechanism found with GRG, required 170 101 function evaluations to produce a mechanism with transmission angles between 120° and 60° and a deviation of less than 3° around the actuator's path with an angular output on the crank of 181°. Generator™, the genetic algorithm, required 80 generations, starting with the same GRG mechanism, to reach a fitness value that did not change in 10 generations. The criteria for each of the simulated annealing and genetic algorithms with respect to actuator orientation, acceptable transmission angles, and crank output were the same. The objective function was the same for each algorithm with penalties being different as noted previously. (It was pointed out in Chapter 5 that in the case of genetic algorithms, penalties that are expressed as a function of the distance from the constraint boundary appear to perform more effectively than those penalties that are simply a function of the number of violated constraints [50]. Since GAs exploit information about the domain as the evolution progresses, even non-feasible solutions provide information that is useful). The actuator's behavior with respect to linear

behavior and the transmission angles for each mechanism follow. Figure 6.7 indicates the variation of the actuator angle with crank angle for the mechanism found with a combination of GRG and GA. Figure 6.8 shows the same mechanism's transmission angles and the deviation from 90°. Figures 6.9 and 6.10 are the variation with crank angle of the actuator orientation and transmission angles, respectively, of the approach using a GRG and SA hybrid. Finally, Figures 6.11 and 6.12 show the hybrid mechanism behavior using a GRG-GA-SA pipeline; these figures are the actuator's orientation and transmission angles over the full range of motion. Figure 6.13 shows the mechanism in its first position and Figure 6.14 reflects its behavior for four different values of the crank's angular displacement. Hybrid 2's synthesis consisted of combining the various optimization procedures by first synthesizing a mechanism using a combination of GRG and GA. The result of the Genetic Algorithm served as the initial guesses for the simulated annealing procedure. The simulated annealing procedure required 164 701 function evaluations to identify a mechanism that yields transmission angles between 80° and 120° although transmission angles were allowed between 60° and 120°. The behavior of the actuator is within the 3° allowable deviation from the simulation's optimal value of actuator orientation.

Even though it is impossible to draw general conclusions from a single presentation of this approach, it is not surprising that genetic algorithms or simulated annealing , global search algorithms, would improve upon results acquired by a local search routine like GRG. The constraint boundary of the domain searched by GRG was defined by static equations of equilibrium that included summation of horizontal and vertical forces imposed on the

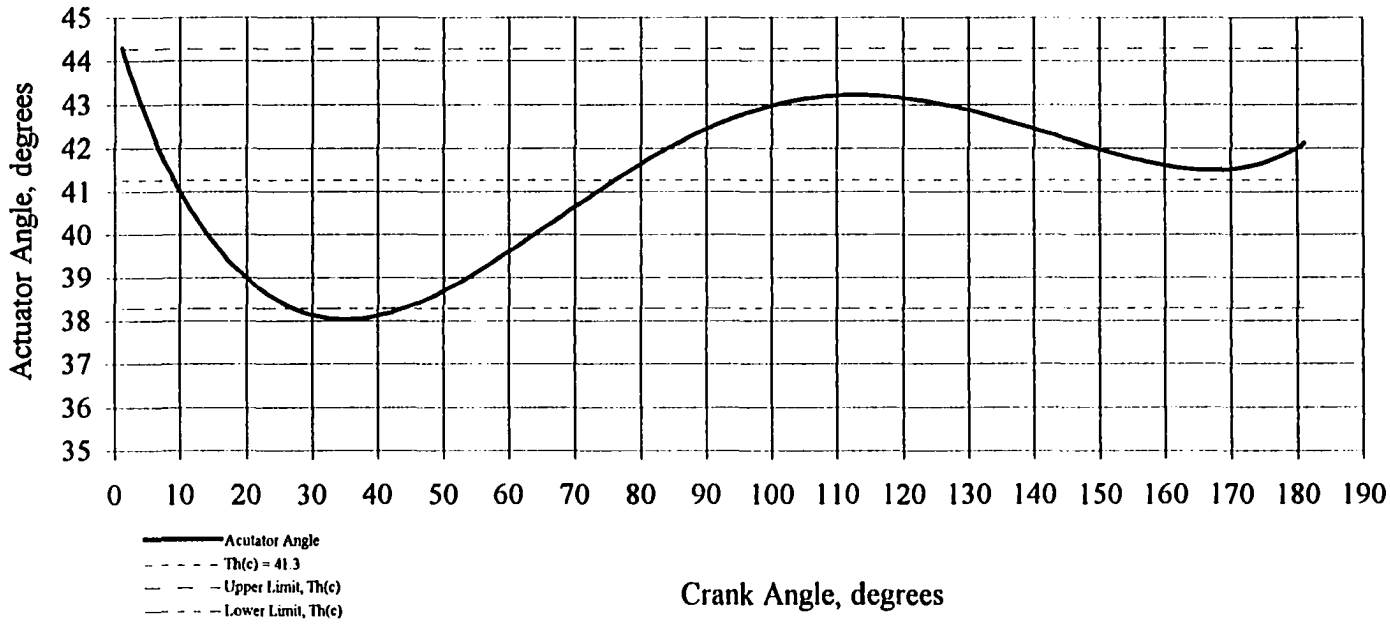

Figure 6.7: GRG-GA variation in actuator angle

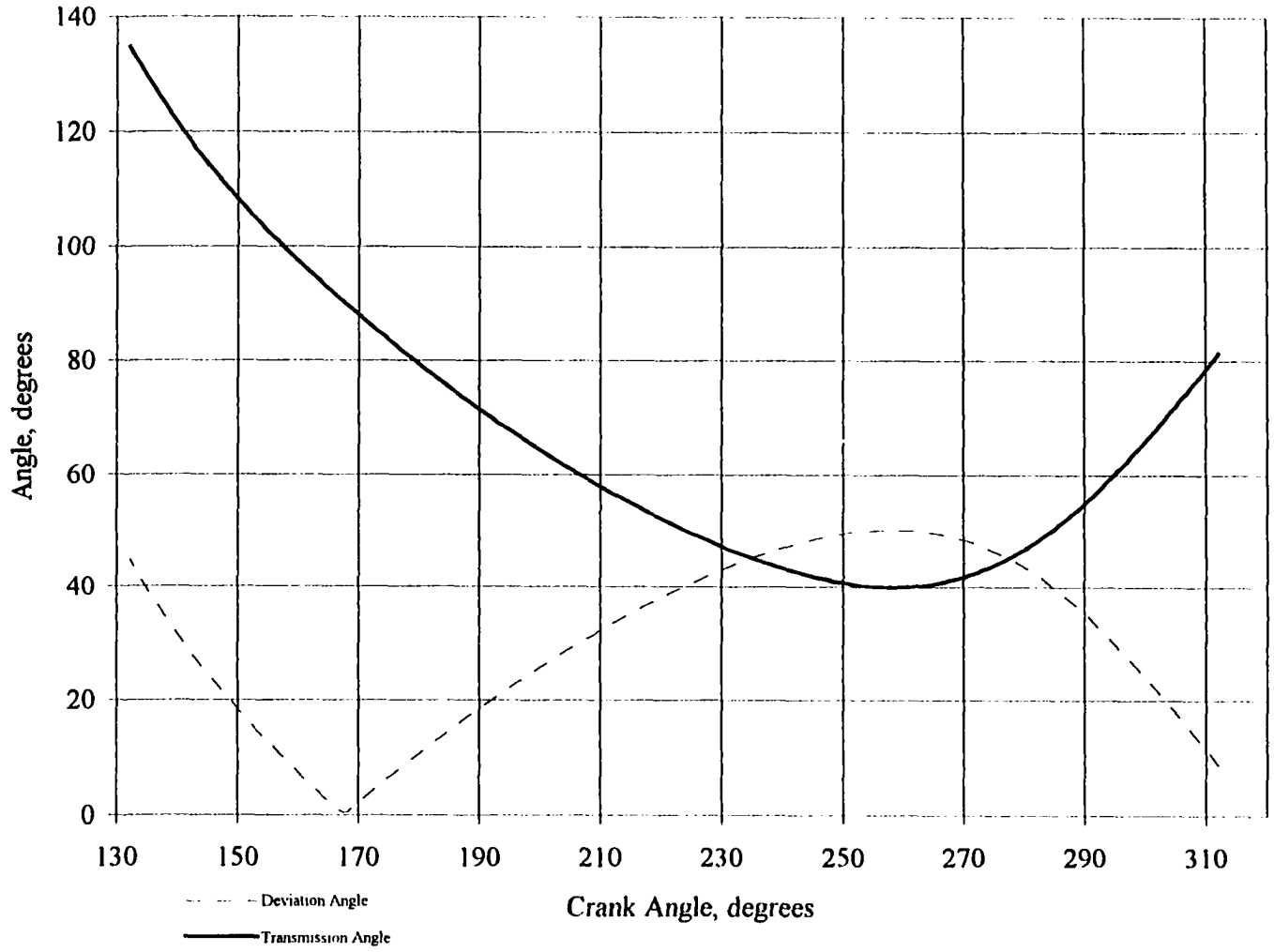

Figure 6.8: GRG-GA transmission angle variation with crank angle

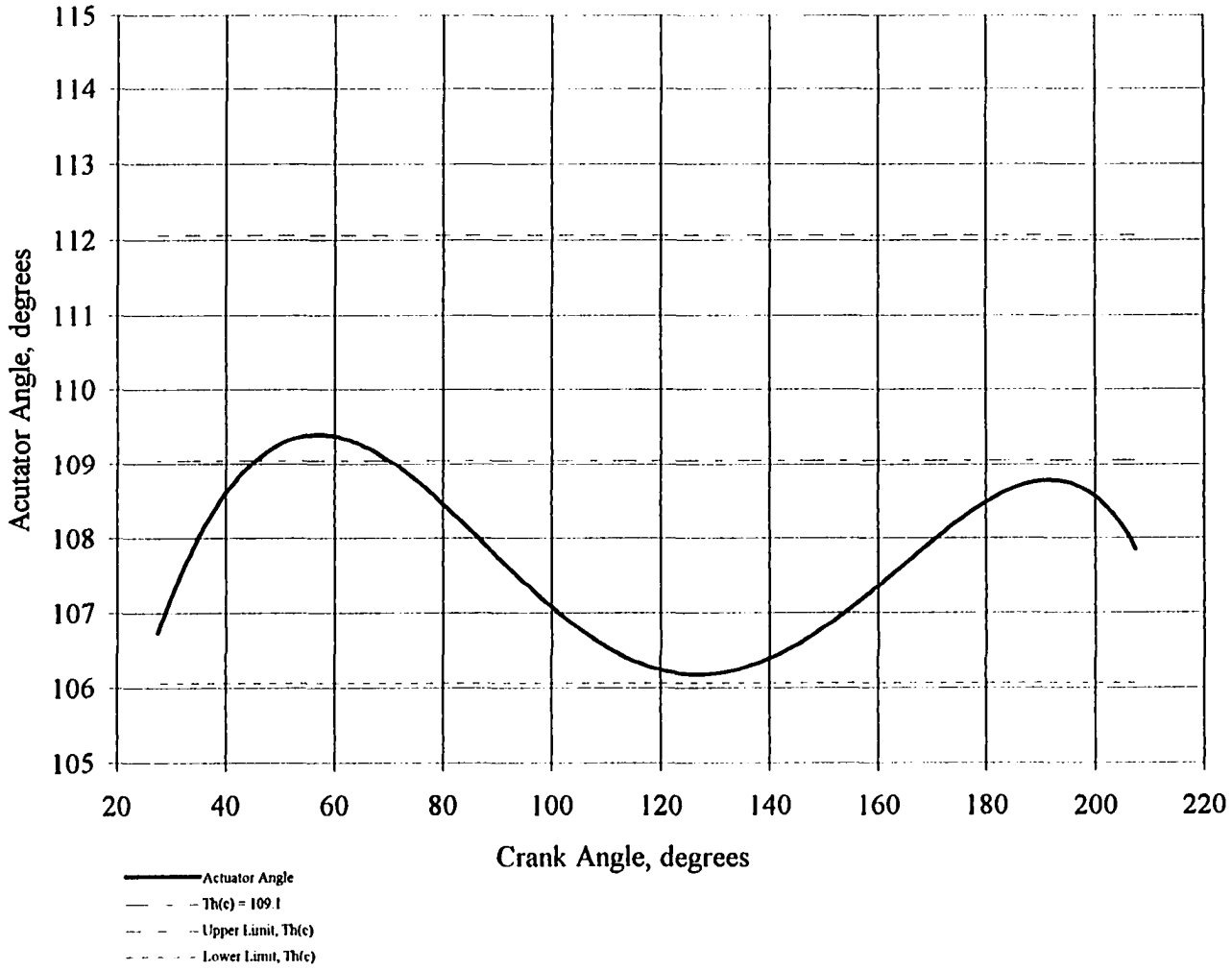

Figure 6.9: GRG-SA hybrid actuator angle variation with crank angle

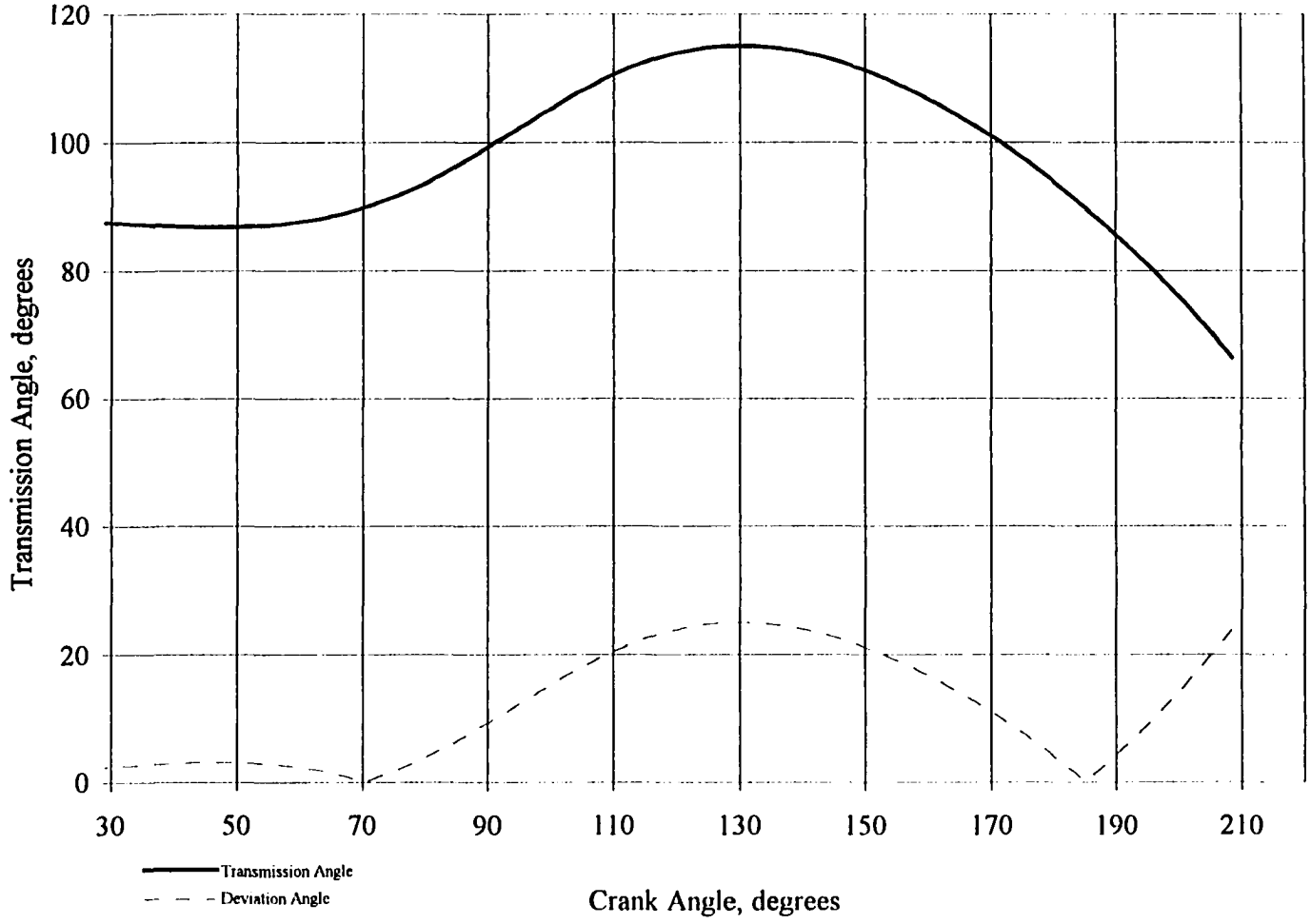

Figure 6.10: GRG-SA variation in transmission angle with crank angle

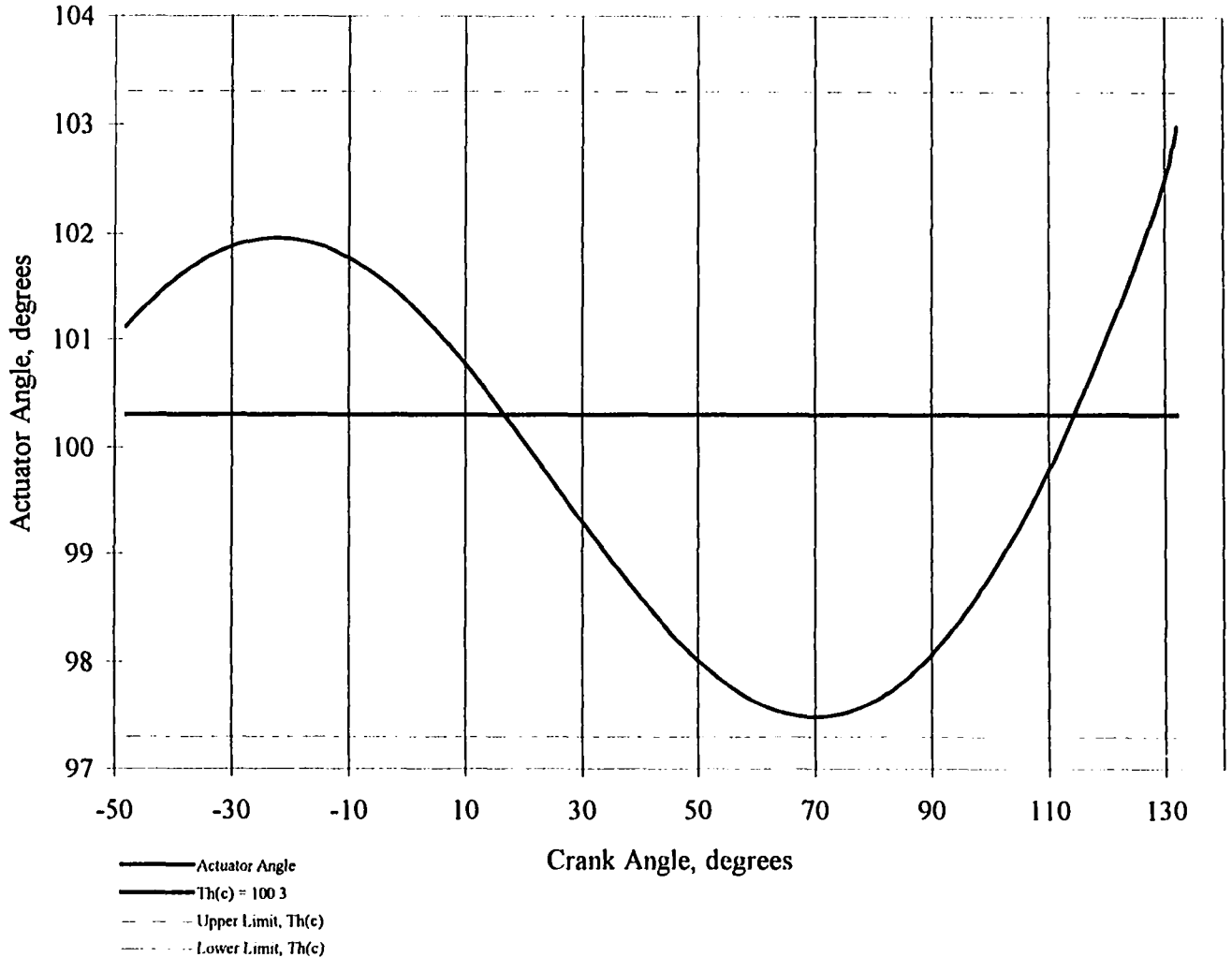

Figure 6.11: Hybrid 2 GRG-GA-SA actuator angle behavior

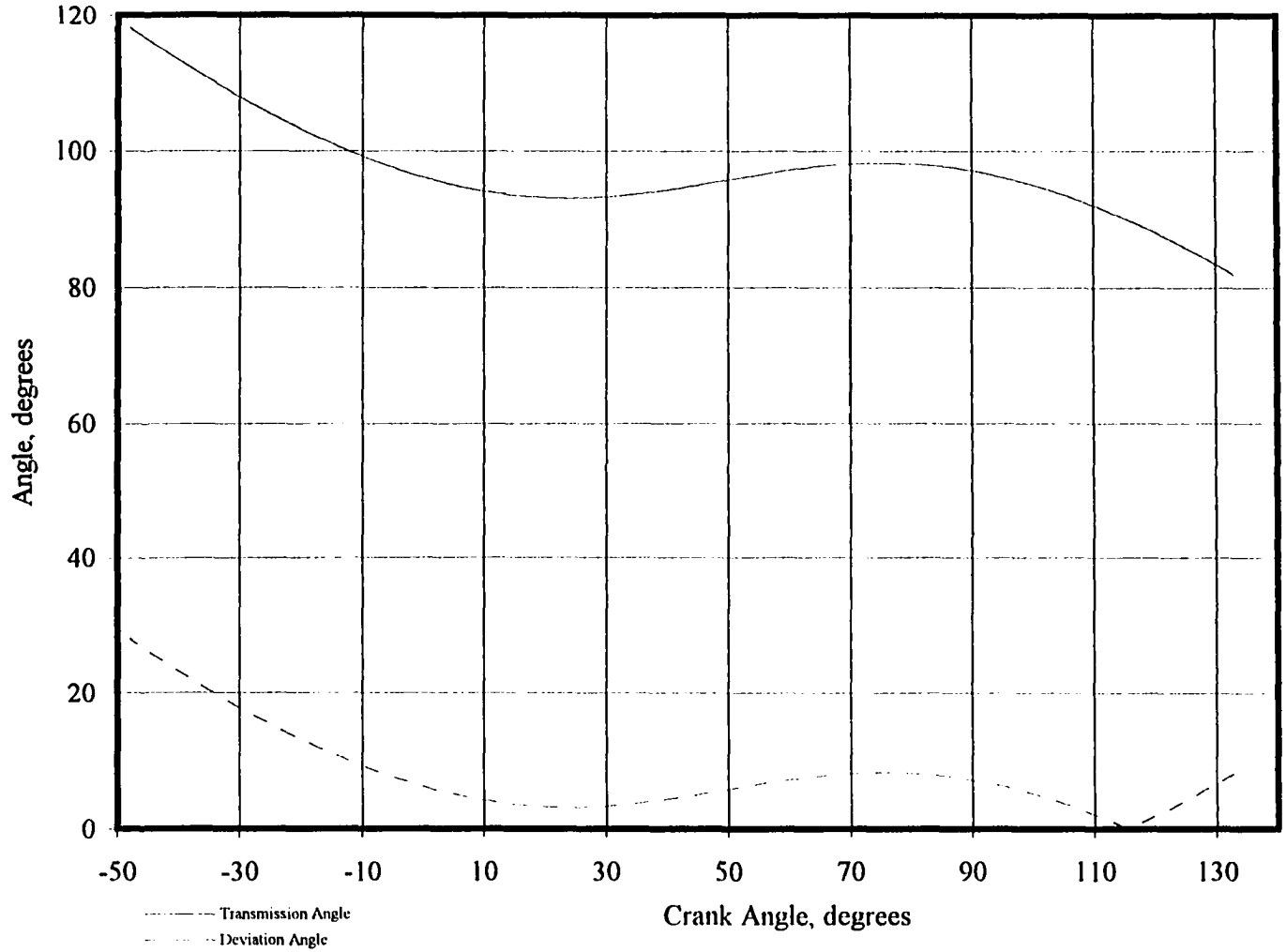

Figure 6.12: Hybrid 2 GRG-GA-SA transmission angles

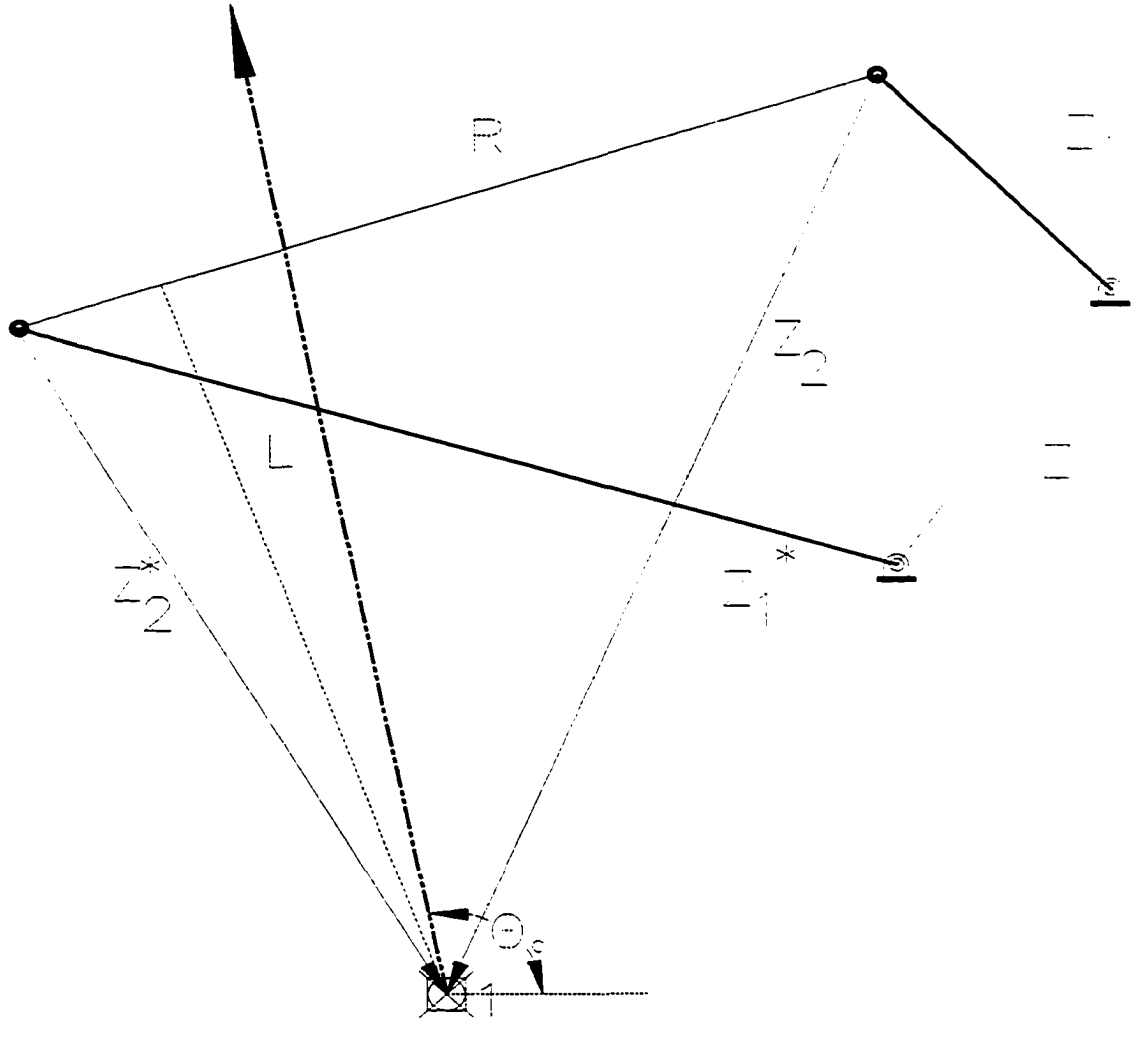

Figure 6.13: GRG-GA-SA hybrid in its first position

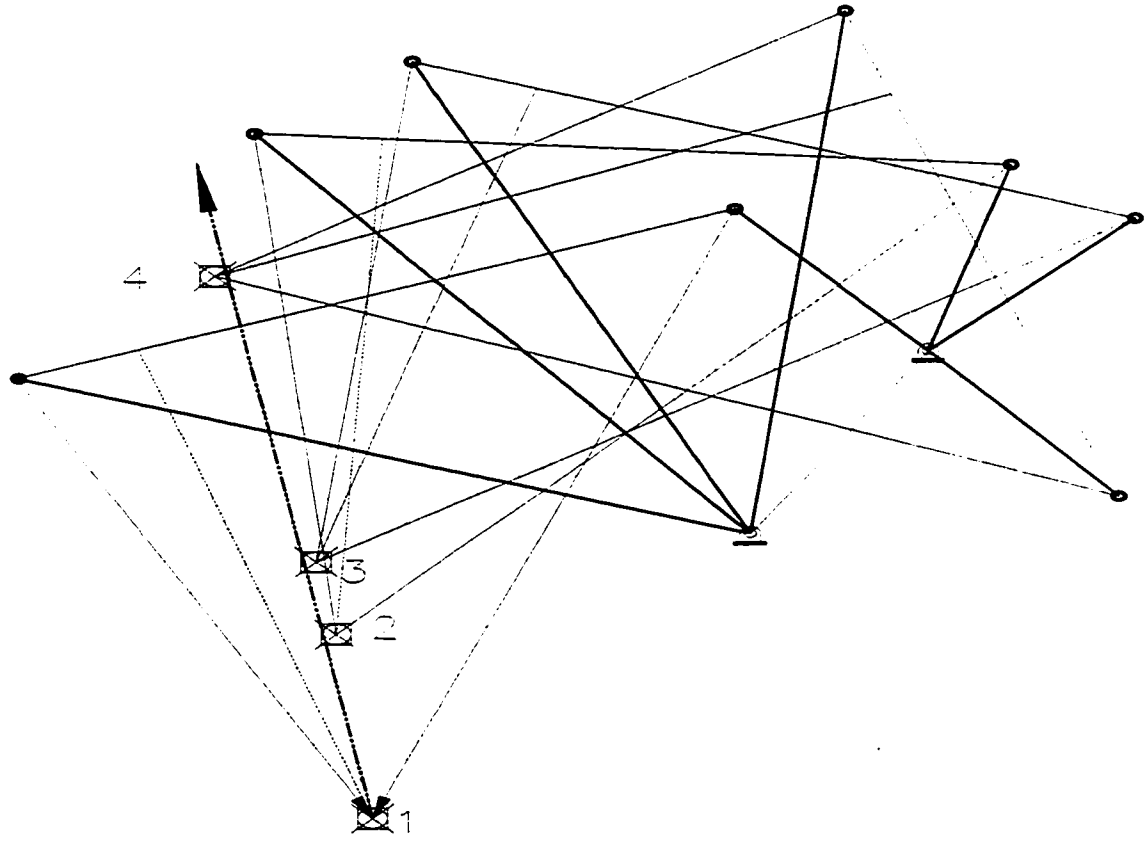

Figure 6.14: GRG-SA-GA hybrid in four positions

mechanism as well as the summation of moments about various points. The genetic algorithm's fitness function was however penalized when the output torque on the crank changed its sense during the mechanism's motion. Using the GRG final results to seed the Genetic Algorithm may therefore introduce schemata templates that are well defined and likely to survive crossover and mutation. The mechanism exhibiting the best characteristics with respect to linear behavior and low transmission angles resulted from the GRG-GA-SA pipeline. Results may be reiterated by observing that the transmission angles for the

mechanism found with GRG varied from a low of 40° to a high of 150°. Giving this mechanism to the Genetic Algorithm yielded a mechanism which produced transmission angles between 40° and 130°. Also, the deviation from straight line behavior was improved with the hybrid. For example, the original mechanism's coupler path slope deviates from the (+/-)3° band for 40° of the crank's motion (see Figure 3.4). The hybrid however, violates the band for approximately 20° (see Figure 6.7). In the case of the mechanism produced with a combination of the GRG and SA, again, transmission angles are observed to be improved. The values ranged, for the hybrid from a low of 65° to a maximum of 115°: this is a significant improvement from the original mechanism's values. Straight line behavior was likewise improved; the (+/-)3° was not violated anywhere throughout the mechanism's motion (see Figure 6.9). The final hybrid, a GRG-GA-SA model showed continued improvement. The minimum transmission angle for this mechanism is 81° and the maximum is 115°. Approximate straight line behavior occurs without any violation of the (+/-)3° band around the numerically determined value of θ_{c1}. Again it is not evident that the mechanism shown in Figure 6.14 represents a global optimum, but it is better than any mechanism found using a single approach. Table 6.4 indicates the mechanism's dimensions.

Table 6.4: Mechanism found using GRG-GA-SA

Z_1 Z_1/Z_1	θ_1	Z_1* Z_1*/Z_1	θ_3	b b/Z_1	L L/Z_1	Z Z/Z_1	θ	θc
17.46 1.00	132.2°	47.14 2.70	162.5°	38.62 2.21	44.56 2.55	19.70 1.13	235.9°	100.3°

VII. SUMMARY AND CONCLUSIONS

This thesis set out to achieve the synthesis of four-bar mechanisms meeting the following criteria:

1) the mechanism's coupler point is to be forced along an approximate straight path by a linear actuator that may not deviate in its lateral motion by more than a specified amount (no more than 3° in the worst case). The driving link is the coupler and the output link is the crank. This differs from the traditional synthesis in that the crank is generally treated as the driving member.

2) the transmission angles throughout the mechanism's motion must maintain values that do not deviate from 90° by more than a specified amount (a smaller band around 90° is preferable, but not more than 50° in the worst case).

3) the crank member of the mechanism must produce an angular output of no less than 180° while the load torque's sense remains opposite that of the crank's rotation throughout the cycle. This requirement is imposed to ensure that displacement along the coupler point path does not change direction before a 180° oscillation of the crank is realized.

The solution to this problem involved two approaches using three different methods of optimization. The first approach used equations of static equilibrium and sought not only kinematic parameters but statics parameters as well that would minimize deviation of

transmission angles from 90° and deviation of the coupler point path from straightness. The generalized reduced gradient solver of Microsoft Excel© was used to solve for dimensions of the four-bar mechanism and is configuration over four precision points. As the problem was one of prescribed timing, the crank's angular displacements were specified. The statics parameters sought in the optimization included the bearing reactions on the crank and the follower links as well as the actuation force on the coupler. The constraint equations were comprised of the component loop closure equations around the mechanism's crank, coupler, follower and ground links and equations for static equilibrium. The objective function for the Generalized Reduced Gradient method was a composite objective that included: minimization of transmission angle deviation from 90°, minimization of perpendicular distances from a feasible start mechanism's coupler path and an ideal mechanism's path, as well as minimization of the deviation of the coupler point path from approximate straight line motion. The Generalized Reduced Gradient solver was successful in producing a mechanism that satisfied the constraints as well as the objectives of the synthesis criteria. The advantages of the generalized reduced gradient method include its robustness and availability [50]. The disadvantages of the method include its difficulty in implementation and the identification of only a local optimum.

Genetic Algorithms were used in an attempt to find a global optimum. Although it cannot be stated that a global optimum was found using this method, it can be said, based on results acquired in this study, that genetic algorithms were very effective in leading to improvements in mechanism quality, i.e., improved transmission angles and more linear

behavior of the coupler point path. The domain was defined somewhat differently for the use of Genetic Algorithms. Statics parameters were not explicitly sought, but the sign on virtual work terms were examined to enforce constraints involving the sense of torque. Genetic Algorithms, in the context of this research, seemed most effective in "sweeping" the domain for areas of minima. Attempts to use the same constraints that were used in the Generalized Reduced Gradient approach were ineffective and did not yield viable mechanisms. An attempt was made to place kinematics parameters on one gene group and statics parameters on a different gene group. Considering each of the four positions, a total of five gene groups would result (one gene group for the kinematics terms and a gene group for the force terms at each position). However, this approach was effective for only one position. Genetic Algorithms showed significant improvement when the domain was restricted to gene groups defined by kinematic parameters coupled with a penalty examining the value of virtual work at every precision point. The objective function involved minimization of deviations from 90° for the transmission angles and minimization of the coupler point path from a straight line. The fitness values were penalized if the sense of torque on the crank changed during the progression of the optimization. Since Genetic Algorithms are exploitive of the domain, these less fit population members, or members that indicated the path was changing direction, were allowed, but at a price to fitness. In other words, the Genetic Algorithm can implicitly avoid future poor members of the population by acquiring information about regions of the domain where they exist.

The representation of parameters in binary form may present problems for a genetic algorithm if many parameters are involved and high precision is required [39]. Binary representation may also present problems in fine tuning near a solution. The chief advantage of Genetic Algorithms is the ability to identify global optima and to preprocess a domain for a global optimum search.

Simulated Annealing used the same objective function as that formed for the genetic algorithm. However, the penalty function employed was an exterior, parabolic penalty formed from inactive constraints. A coefficient of the penalty function, r, was increased as the penalty function's value became smaller and smaller; eventually r became large enough to render 1/r, the multiplier of the penalty, essentially zero.

The chief advantages of Simulated Annealing are its minimal coding requirements and clever emulation of a naturally occurring phenomenon. Many applications of Simulated Annealing have shown it to be effective in identifying global optima [51, 52, 53,5 4]. Among the disadvantages of Simulated Annealing and its inherent slowness [55]. Additionally, algorithm parameters, such as starting temperature and length of the Markov chain, which assures sufficient sampling of the solution space, are highly problem dependent. Analytical expressions have been developed for these parameters [56, 57] but often require "artful" fine tuning.

Hybrids of the methods were effective in providing continued improvement to mechanisms found with Generalized Reduced Gradient and Genetic Algorithms and pipelined to Simulated Annealing. A reduction in the number of function evaluations by

Simulated Annealing was observed when the domain was preprocessed by Genetic Algorithms. That does not imply increased numerical efficiency since the work of GA and SA were not considered collectively. In other words, the cumulative numerical work of the algorithms is not considered--only that of Simulated Annealing preceded by application of GA.

A generic approach to solving the kinematics problem might best begin with the solution to a set of nonlinear equations made up of the same constraints used in the Generalized Reduced Gradient solver. The solution would result in a feasible mechanism for a four position problem. After identifying such a mechanism, the mechanism could serve as the initial guess values for a local search method like Generalized Reduced Gradient. Since the generalized reduced gradient method requires active constraints at the start of its search, solution the non-linear problem is required. If a global solution is sought, Genetic Algorithms could be used as a preprocessor of the domain of interest, followed by application of Simulated Annealing. Although a global minimum may be found first with Genetic Algorithms, subsequent application of Simulated Annealing could serve to confirm such a claim. Post processing of GA results with Simulated Annealing did result in improved mechanism quality in this study.

The following observations may be made regarding the mechanisms produced within this work:

1. All mechanisms, with the exception of the GRG-GA-SA hybrid are non-Grashof double-rockers (no link is capable of a complete revolution).

2. Link length ratios, longest link/shortest link, varied between a minimum of 2.15 to a maximum ratio of 6.25.
3. The crank is typically the shortest of the links in the mechanisms; the follower, the longest link.

Table 7.1 summarizes the characteristics of the mechanisms identified in this research.

The objective function used for synthesis of mechanism GA-1, the first mechanism discussed in the chapter on Genetic Algorithms, consisted of a minimization of the differences in slopes at subsequent positions of the mechanism (or, orientations of the actuator). Consequently, a tolerance band was not specified.

With respect to mechanism quality as determined by transmission angles, Table 7.1 indicates that mechanisms produced with Simulated Annealing were consistently favorable. Mechanism GA-2 exhibits good transmission angles, but only fair behavior with respect to crank output. This observation is particularly significant when mechanism GA-2's crank output is compared to that of the three mechanisms produced with Simulated Annealing. Mechanism GA-3 is among the best of the mechanisms produced with Genetic Algorithms when considering all criteria collectively; i.e., good behavior with respect to linear motion, acceptable transmission angles, and good angular output of the crank.

The Hybrid-1 mechanism will produce a coupler curve that is well behaved with respect to linear behavior and minimal deviation from the actuator's orientation. The transmission angles at each position of this mechanism are very good and remain very close

Table 7.1: Mechanism Summary

Solution Method	Allowed Deviation θ_c	θ_c MAX.	θ_c MIN.	Allowed Deviation μ_T	μ_T MAX.	μ_T MIN.	Crank Output
GRG	+/- 3° θ_C=44.9°	48.9° +4.0°	44.5° -0.4°	+/- 50°	157.7°	40.0°	181°
SA - 1	+/- 1° θ_C=161.1°	163.5° +2.4°	160.2° -0.9	+/- 20°	108.8°	90.2°	200°
SA - 2	+/- 2° θ_C=35.7°	37.6° +1.9°	32.4° -3.3°	+/- 20°	105.7°	90.1°	200°
SA - 3	+/- 2° θ_C=187.5°	189.6° +2.1°	185.4° -2.1°	+/- 20°	109.0°	90.1°	220°
GA - 1	NOT SPECIFIED θ_C=5.0°	5.8° +0.8°	4.95° -0.05°	NOT SPECIFIED	157.0°	40.0°	181°
GA - 2	+/- 3° θ_C=36.1°	39.3° +3.2	33.7° -2.4	+/- 30°	105.4°	90.0°	181°
GA - 3	+/- 3° θ_C=122.6°	125.9° +3.3	120.1° -2.5	+/- 20°	121.2°	90.1°	200°
SA - RANDOM	+/- 1° θ_C=40.8°	43.2° +2.4	39.7° -2.9	+/- 20°	120.0°	90.0°	200°
GA-SA HYBR-1	+/- 1° θ_C= 64.9°	66.4° +1.5	63.7° -1.2°	+/- 20°	110.3°	90.2°	181°
GRG-SA	+/- 3° θ_C=109.1°	109.4° +0.3°	106.2° -2.9°	+/- 30°	114.9°	90.1°	181°
GRG-GA	+/- 3° θ_C= 41.3°	44.3° +3.0°	38.0° -3.3	+/- 30°	140.1°	90.4°	181°
GRG-GA-SA HYBR-2	+/- 3° θ_C=100.3°	103.0° +2.7°	97.5° -2.8°	+/- 30°	118.1°	90.2°	181°

to the 20° band allowed at all positions. This mechanism was produced by starting a genetic algorithm with the same random numbers used for mechanism SA-Random. After optimization with the genetic algorithm, the resulting mechanism parameters were given to Simulated Annealing as initial values, optimization was again performed and Hybrid-1

resulted. Comparison of Hybird-1 and SA-Random indicate that when start values of the procedures are random, a combination approach yields improved mechanism behavior with respect to linear behavior.

Mechanism GRG-SA (this mechanism was synthesized by starting the simulated procedure with mechanism GRG) exhibits better transmission angles and linear path behavior than mechanism GRG-GA (produced by starting the genetic algorithm with mechanism GRG). Continued refinement with Simulated Annealing provides further improvement in mechanism quality (GRG-GA-SA).

BIBLIOGRAPHY

[1] Kempe, A.B., 1877 *How to Draw a Straight Line*, Nature Bd. 16, London.

[2] Kraus, R.,1955, *Die Geradführung durch das Gelenkviereck*, Verlag, Düsseldorf.

[3] Hain, K., 1967, *Applied Kinematics, 2nd edition*, Mc-Graw Hill Inc., New York.

[4] Norton, D., 1992, *Design of Machinery*, Mc-Graw Hill Inc., New York.

[5] Sandor, G. and Erdman, A., 1984, *Advanced Mechanism Design: Analysis and Synthesis, Volume 2*, Prentice-Hall International, Inc., Englewood Cliffs, N.J.

[6] Nolle, H. and Hunt, K.H., 1971, Optimum synthesis of planar linkages to generate coupler curves, *Journal of Mechanisms*, Vol.6, pp. 267-287.

[7] Kramer, S.N. and Sandor, G.N., 1975, Selective Precision synthesis-a general method of optimization for planar mechanisms, *ASME Journal of Engineering For Industry*, May, pp. 689-701.

[8] Jain, P. and Agogino, A., 1988, Optimal design of mechanisms using simulated annealing: theory and applications, *ASME Advances in Design Automation*, DE-Vol. 14, pp.155-164.

[9] Martinez-Alfaro, H., 1993, Collision-free path planning for robots using B-splines and simulated annealing. Ph.D. Dissertation, Iowa State University.

[10] Papalambros, P.Y., 1995, Optimal design of mechanical engineering systems. *ASME 50th Anniversary of the Design Engineering Division*, DE-Vol. 117, June, pp. 55-62.

[11] Gabriele, G., Angeles, J., Liu,Z., and Faik, S., 1990 Optimization in mechanisms. *The First Forty Years of Modern Kinematics - A Tribute to Ferdinand Freudenstein*, pp. 11:1-11:24.

[12] Huang, C. and Roth, B., 1993, Dimensional synthesis of closed-loop linkages to match force and position specifications, *ASME Journal of Mechanical Design*, DE-Vol. 115, June, pp. 194-198.

[13] Raghavan, M., 1988, Analytical methods for designing linkages to match force specifications, Ph.D. Dissertation, Stanford University.

[14] Huang, C., and Roth, B., 1994, Position-force synthesis of closed-loop linkages. *ASME Journal of Mechanical Design*, DE-Vol. 116., March, pp. 155-162.

[15] Huang, C. and B. Roth, 1993,Dimensional synthesis of closed-loop linkages to match force and position specifications, *ASME Journal of Mechanical Design*, DE-Vol. 115, 1993, pp. 194-198.

[16] Gabriele, G.A., and Ragsdel, K.M., 1977, The generalized reduced gradient method: a reliable tool for optimal design, *ASME Journal of Engineering for Industry*, May, pp. 394-400.

[17] Abadie, J., and Carpentier, J.M, 1969, Generalization of the Wolfe reduced gradient method to the case of nonlinear constraints. *Optimization*, R. Fletcher, ed., Academic Press, New York, pp. 37-47.

[18] Wolfe, P., 1963, Methods of nonlinear programming. *Recent Advances in Mathematical Programming*, R.L. Graves, and P. Wolfe, eds., McGraw-Hill, New York, pp. 76-77.

[19] Vanderplaats, G.N., 1984, *Numerical Optimization Techniques for Engineering Design: With Applications*, Mc-Graw Hill, Inc., New York.

[20] Walkeback, J. and Maguiness, D., 1992, *Excel 4 for Windows*, IDG Books, Inc., San Mateo, California.

[21] Lasdon, L.S., Waren, A.D., and Ratner, M., 1978, Generalized reduced gradient code for nonlinear programming, *ACM Transactions on Mathematical Software*, Vol. 4, No.1, March, pp. 34-50.

[22] Arora, J.S., 1989, *Introduction to Optimum Design*, Mc-Graw Hill, Inc., New York.

[23] Parks, G.T., 1990, An intelligent stochastic optimization routine for nuclear fuel cycle design, *Nuclear Technology*, Vol. 89, February, pp. 223-246.

[24] Malhotra, A., Oliver, J. and Tu, W., 1991, Synthesis of spatially and intrinsically constrained curves using simulated annealing. *ASME Advances in Design Automation*, DE-Vol. 1, pp. 145-155.

[25] Devadas, S. and Newton, R., 1987, Topological optimization of multiple-level array logic, *IEEE Transactions on Computer-Aided Design*, Vol. CAD No. 6, November, pp. 915-941.

[26] Friesez, T.L., Cho, H.J., and Mehta, N.J., 1992, A simulated annealing approach to the network design problem with variational inequality constraints, *Transportation Science*, Vol. 26, February, pp. 18-26.

[27] Ingber, A.L., 1993, Simulated annealing: practice versus theory, *Journal of Mathematical Computer Modeling*, Vol. 18, No. 11, pp. 29-57.

[28] Geoffe, Ferrier, and Rogers, 1994, Global optimization of statistical functions with simulated annealing, *Journal of Econometrics*, Vol. 60, No. ½, January/February, pp. 65-100.

[29] Corona, A., Marchesi, M., Martini, C., and Ridella, S., 1987, Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm. *ACM Transactions on Mathematical Software*, Vol. 13, No. 3, September, pp. 262-280.

[30] Faigle, U. and Kern, W., 1991, Note on the convergence of simulated annealing algorithms, *Math. Oper. Res.*, Vol. 29, January, pp. 153-159.

[31] Davis, L., 1987, *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann Publishers, Inc., Los Altos, California.

[32] Holland, J.H., 1975, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.

[33] Wang, B.P. and Chen, J.L., 1994, Application of genetic algorithm for the support location optimization of beams, *ASME Advances in Design Automation*, DE-Vol. 69-2, pp. 323-327.

[34] Chapman, C.D. and Jakiela, M.J., 1994, Genetic algorithm-base structural topology design with compliance and manufacturability considerations, *ASME Advances in Design Automation*, DE-Vol. 69-2, pp. 309-322.

[35] Sandgren, E., Jensen, E., and Welton, J.W., 1990, Topological design of structural components using genetic optimization methods, *Sensitivity Analysis and Optimization with Numerical Methods*, AMD-Vol. 115, Proceedings of the Winter Annual Meeting of ASME, Dallas. pp. 31-43.

[36] Ashley, S., 1992, Engineous explores the design space, *Mechanical Engineering*, February, pp. 49-52.

[37] Goldberg, D.E., 1989, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing, Inc., Reading, Mass.

[38] Beasley, D., Bull, D. and Martin, R., 1993, *An Overview of genetic algorithms: part 1, foundations*, University Computing, Vol. 15, No. 2, pp. 58-69.

[39] Michalewicz, Z., 1994, *Genetic Algorithms + Data Structures = Evolution Programs, 2nd extended edition* , Springer-Verlag, New York.

[40] McGrew, S., (algorithm architecture), Wickes, L., (software engineer) and Devries, D., (documentation), 1995, *Generator Release 1.0* by New Light Industries, Spokane.

[41] Beasley, D., Bull, D., and Martin, R.,1993, *An overview of genetic algorithms: part 2, research topics*, University Computing, Vol. 15, No. 4, pp. 170-181.

[42] Richardson, J.T., Palmer, M.R., Liepins, G., and Hilliard, M., 1989, Some guidelines for genetic algorithms with penalty functions, *Proceedings of the Third International Conference on Genetic Algorithms*, Morgan Kaufman Publishers, Los Altos, California, pp. 191-197.

[43] DeJong, K., 1965, Genetic algorithms: A 10 year perspective, in Grefenstette (ed.), *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pp. 168-177.

[44] Goldberg, D.E., 1987, Simple genetic algorithms and the minimal deceptive problem, in Davis (ed.), *Genetic Algorithms and Simulated Annealing*, Morgan Kaufman Publishers, Inc., Los Altos, California.

[45] Homaifar, A., Qi, C.X., and Lai, S.H., 1994, Constrained optimization via genetic algorithms, *Simulation*, Vol. 62, No. 4, pp. 242-254.

[46] Poon, P.W. and Parks, G.T., 1993, Application of genetic algorithms to in-core nuclear fuel management optimization, *Proceedings of the Joint International Conference on Mathematical Methods and Supercomputing in Nuclear Applications*, I-777, 19-23, April, Karlsruhe, Germany.

[47] Sirag, D.and Weisser, P.T., 1987, Toward a unified thermodynamic genetic operator, *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, Grefenstette (ed.), July, pp. 116-122.

[48] Murata, T. and Ishibuchi, H., 1994, Performance evaluation of genetic algorithms for flowshop scheduling problems, *Proceedings of the 1st IEEE Conference on Evolutionary Computation*, Vol. 2, pp. 812-817.

[49] Ingber, L. and Rosen, B., 1992, Genetic algorithms and very fast simulated reannealing: a comparison, *Mathematical and Computer Modeling*, Vol. 16, No. 11, pp. 87-100.

[50] Reklaitis, G.V., Ravindran, A. and Ragsdell, K.M., 1983, *Engineering Optimization Methods and Applications*, John Wiley and Sons,

[51] Kropaczek, D.J., Parks, G.T., Maldonado, G.I. and Turinsky, P.J., 1991, Application of simulated annealing to in-core nuclear fuel management optimization, *Advances in Mathematics, Compuatations, and Reactor Physics*, Vol. 5, International Topical Meeting, Pittsburgh, April, pp. 22.1 1:1-22.1 1:12.

[52] Jwayman, J.L., 1990, Optimization of signal distribution networks using simulated annealing, *IEEE Transactions on Communications*, Vol. 40, July, pp. 465-471.

[53] Wong, D.F., Leong, H.W. and Liu, C.L., 1988, *Simulated annealing for VLSI design*, Kluwer Academic Publishers, Boston.

[54] Wilson, S.S., 1991, Teaching network connectivity using simulated annealing on a massively parallel processor, *Proceedings of the IEEE*, Vol. 79, April, pp. 559-566.

[55] Davis, L. and Steenstrup, M., 1987, Genetic algorithms and simulated annealing: an overview, *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann Publishers, Los Altos, California, pp. 1- 11.

[56] Romeijn, H.E. and Smith, R.L., 1995, Simulated annealing for constrained global optimization, *Journal of Global Optimization*, Vol.5, pp. 101-126.

[57] Jones, A.E.W., and Forbes, G.W., 1995, An adaptive simulated annealing algorithm for global optimization over continuous variables, *Journal of Global Optimization*, Vol. 6, pp. 1-37.

APPENDIX - SIMULATED ANNEALING CODE

```
      program simann

      parameter (n=9,neps=4)
      implicit double precision (a-h,o-z)
      double precision lb(n),ub(n),x(n),xopt(n),c(n),vm(n),
     .                 fstar(neps),xp(n),t,eps,rt,fopt

      integer nacp(n),ns,nt,nfcnev,ier,iseed1,iseed2,maxevl,
     .        iprint,nacc,nobds


      logical max

      external fcn,constr,rdeter
      pi = 4.d0*datan(1.d0)
      max = .false.
      eps = 1.d-6
      rt = .85
      iseed1 = 1
      iseed2 = 2
      ns = 20
      nt = 15
      maxevl = 750000
      iprint = 1
      do 10 i = 1,n
c       lb(i) = -1.d25
c       ub(i) = 1.d25
        c(i) = .25d0
 10   continue

      lb(1) = 1.d0
      ub(1) = 50.d0
      lb(2) = 0.d0
      ub(2) = 2.d0*pi
      lb(3) = 1.d0
      ub(3) = 50.d0
      lb(4) = 0.d0
```

```
      ub(4) = 2.d0*pi
      lb(5) = 1.d0
      ub(5) = 50.d0
      lb(6) = 1.d0
      ub(6) = 50.d0
      ub(7) = 50.d0
      lb(7) = 0.d0
      ub(8) = 2.d0*pi
      lb(8) = 0.d0
      lb(9) = 0.d0
      ub(9) = 2.d0*pi
```

INITIAL GUESSES

```
      x(1) = 25.d0
      x(2) = pi/4.d0
      x(3) = 25.d0
      x(4) = pi/2.d0
      x(5) = 25.d0
      x(6) = 25.d0
      x(7) = 25.d0
      x(8) = pi/3.d0
      x(9) = 45.d0*pi/180.d0
```

INITIAL TEMPERATURE

```
      t =2000.d0
      do 20 i = 1,n
       vm(i) = 1.0
 20   continue

      write(*,1000)n,max,t,rt,eps,ns,nt,neps,maxevl,iprint,
     .           iseed1,iseed2

      call prtvec(x,n,'starting values')
      call prtvec(vm,n,'initial step length')
      call prtvec(lb,n,'lower bound')
      call prtvec(ub,n,'upper bound')
      call prtvec(c,n,'c vector')

      write(*,'(/," **** end of driver routine output ***"
     .        /," **** before call to SA.          ***")')
```

```
      call sa(n,x,max,rt,eps,ns,nt,neps,maxevl,lb,ub,c,iprint,iseed1,
     .     iseed2,t,vm,xopt,fopt,nacc,nfcnev,nobds,ier,fstar,xp,
     .     nacp)
      open(unit=8,file='simu.out',status='unknown')
      write(8,'(/" **** results after sa   **** ")')

      call prtvec(xopt,n,'solution')
      call prtvec(vm,n,'final step length')
      write(8,*)'solution is-->',(xopt(i),i=1,n)
      write(8,*)'step length-->',(vm(i),i=1,n)
      write(8,1001) fopt,nfcnev,nacc,nobds,t,ier
      rx =-xopt(1)*dcos(xopt(2))+xopt(7)*dcos(xopt(8))
     . + xopt(3)*dcos(xopt(4))
      ry =-xopt(1)*dsin(xopt(2))+xopt(7)*dsin(xopt(8))
     . + xopt(3)*dsin(xopt(4))
      r = dsqrt(rx**2+ry**2)
      gamma = datan2(ry,rx)*180.d0/pi
      write(8,610)r,gamma
  610 format(1x,'r = ',f10.4,t20,'gamma = ',f10.3)
 1000 format(/,' simulated annealing example ',/,
     .     /,' number of parameters: ',i3,'  maximization: ',l5,
     .     /,' initial temp: ',g8.2,' rt: ',g8.2,' eps: ', g8.2,
     .     /,' ns: ',i3,'  nt ',i2,' neps: ',i2,
     .     /,' maxevl: ',i10,' iprint: ',i1,'  ised1: ', i4,
     .     ' iseed: ',i4)
 1001 format(/,' optimal function value: ',g20.13,
     .     /,' number of function evaluations:  ',i10,
     .     /,' number of accepted evaluations:  ',i10,
     .     /,' number of out of bounds evaluations: ',i10,
     .     /,' final temp: ',g20.13,' ier: ',i3)

      stop
      end
```

SUBROUTINE FCN SUPPLIES THE OBJECTIVE FUNCTION

```
      subroutine fcn(n,x,f)

      implicit double precision (a-h,o-z)
      double precision x(n),l,f,fvec(9),lprmx(9),lprmy(9),lprm(9)
        double precision nuprm(9),term1(9),th3n(9),psi(8),gamm(9),
     . trans(9),ambd(9),by(9),bx(9),phi(8),ptx(9),pty(9),
```

```
     . delx(8),dely(8),thc(9),a(9)

      save fold
        nn = 9
        nm1 = 8
      pi = 4.d0*datan(1.d0)
      z1  =  x(1)
      th1  =  x(2)
      z1s  =  x(3)
      th3  =  x(4)
      b =     x(5)
      l =     x(6)
      z =     x(7)
      th =    x(8)
      thc(1) = x(9)

c       do 1 k = 1,nm1
c
c       phi(k) = -22.625*dfloat(k)*pi/180.d0
c 1     continue

      phi(1) = -5.d0*pi/180.d0
      phi(2) = -20.d0*pi/180.d0
      phi(3) = -40.d0*pi/180.d0
      phi(4) = -60.d0*pi/180.d0
      phi(5) = -100.d0*pi/180.d0
      phi(6) = -150.d0*pi/180.d0
      phi(7) = -181.d0*pi/180.d0
      phi(8) = -200.d0*pi/180.d0

      tsum = 0.d0
      usum = 0.d0
      vsum = 0.d0
      wsum = 0.d0

      z2 = dsqrt(b**2+l**2)

      beta = datan2(l,b)

      lprmx(1) = z1*dcos(th1)-z*dcos(th)
      lprmy(1) = z1*dsin(th1)-z*dsin(th)
```

```
      do 6 i = 2,nn
      lprmx(i) = z1*dcos(th1+phi(i-1))-z*dcos(th)
      lprmy(i) = z1*dsin(th1+phi(i-1))-z*dsin(th)
 6    continue



      do 100 i = 1,nn
      nuprm(i) = datan2(lprmy(i),lprmx(i))
      lprm(i) = dsqrt(lprmx(i)**2+lprmy(i)**2)
 100 continue

      rx11 = -lprmx(1)+z1s*dcos(th3)
      ry11 = -lprmy(1)+z1s*dsin(th3)
      r = dsqrt(rx11**2+ry11**2)

      do 200 i = 1,nn
      term1(i) = (r**2-lprm(i)**2-z1s**2)/(-2.d0*lprm(i)*z1s)
      if(dabs(term1(i)).gt.1.d0)then
c      print*,'in acos loop'

        f = 1000000.d0
      return
        endif
 200  continue

      do 150 i = 1,nn
      th3n(i) = nuprm(i)+acos(term1(i))
 150  continue


      do 160 i = 1,nm1
      psi(i) = th3n(i+1)-th3
 160   continue

      gamm(1) = datan2(z1s*dsin(th3)-lprmy(1),
     .           z1s*dcos(th3)-lprmx(1))
      do 7 i = 2,nn
      gamm(i) = datan2(z1s*dsin(th3+psi(i-1))-lprmy(i),
     .           z1s*dcos(th3+psi(i-1))-lprmx(i))
 7    continue
```

```
      z2s = dsqrt(l**2+(r-b)**2)

      ptx(1) = z1*dcos(th1)+z2*dcos(gamm(1)+beta)
      pty(1) = z1*dsin(th1)+z2*dsin(gamm(1)+beta)

      do 8 i = 2,nn
      ptx(i) = z1*dcos(th1+phi(i-1))+z2*dcos(gamm(i)+beta)
      pty(i) = z1*dsin(th1+phi(i-1))+z2*dsin(gamm(i)+beta)
8     continue
      do 9 i = 1,nm1
      delx(i) = ptx(i+1)-ptx(1)

      dely(i) = pty(i+1)-pty(1)
      thc(i+1)= datan2(dely(i),delx(i))
      if(thc(i+1).lt.0.0d0)thc(i+1)=thc(i+1)+2.d0*pi
9     continue

      delta=datan2(l,r-b)
      a(1) = z2s*sin(gamm(1)+pi-delta-th3)/(sin(th3-thc(1)))
        do 10 i= 2,nn

      a(i) = z2s*sin(gamm(i)+pi-delta-th3-psi(i-1))/
     .                    (sin(th3+psi(i-1)-thc(i)))
10    continue
      do 11 i =1,nn
      bx(i) = z2*cos(gamm(i)+beta)+a(i)*cos(thc(i))

      by(i) = z2*sin(gamm(i)+beta)+a(i)*sin(thc(i))


      ambd(i) = datan2(by(i),bx(i))
11      continue
      trans(1) = ambd(1)+pi-th1
      do 12 i = 2,nn
      trans(i) = ambd(i)+pi-(th1+phi(i-1))
12    continue
      do 13 i = 1,nn

      if(trans(i).lt.0.d0)trans(i)=trans(i)+2.d0*pi
      if(trans(i).gt.2.d0*pi)trans(i) = 2.d0*pi-trans(i)
      if(trans(i).gt.pi)trans(i)=trans(i)-pi
13    continue
```

```
      iflag = 0
      pentra = 0.d0
      do 14 i = 1,nn
      if((trans(i).gt.110.d0*pi/180.d0).or.
     .                  (trans(i).lt.70.d0*pi/180.d0))then
      pentra=dabs(pi/2.d0-trans(i))+pentra
      endif
      tsum = tsum+dabs(pi/2.d0-trans(i))

   14 continue



      obj7 = tsum
      tsum = 0.d0
        penthc = 0.d0
      do 15 i =1,nm1
       if(dabs(thc(i+1)-thc(1)).gt.2.d0*pi/180.d0)then
        penthc = (dabs(thc(i+1)-thc(1)))+penthc
        endif

        usum = usum+dabs(thc(i+1)-thc(1))

   15  continue
      obj9 = usum
      usum = 0.d0
      do 16 i = 1,nn
      vsum =vsum+thc(i)
   16  continue
      avg = vsum/dfloat(nn)
      do 17 i=1,nn
      wsum = wsum+dabs(thc(i)-avg)
   17  continue
      obj8 = wsum
      wsum = 0.d0

      call constr(n,x,fvec,th3n,gamm,r,beta,thc,psi,pen)
      call rdeter(pen,rpen)
      f =rpen*pen + pentra + 1.5d0*penthc
      fold = f
      return
      end
```

SUBROUTINE RDETER FINDS THE PENALTY COEFFICIENT

```
      subroutine rdeter(pen,r)
      implicit double precision (a-h,o-z)
      if(pen.ge.100)r=1000.d0
      if((pen.lt.100).and.(pen.ge.10))r=100.d0
      if((pen.lt.10).and.(pen.ge.1))r=50.d0
      if((pen.lt.1).and.(pen.ge..1))r= 10.d0
      if((pen.lt..1).and.(pen.ge..01))r= 1.d0
      if((pen.le..01).and.(pen.ge..001))r=0.1d0
      if((pen.le..001).and.(pen.ge..0001))r=.01d0
      if((pen.le..0001).and.(pen.ge..00001))r=.001d0
      if(pen.lt..00001)r=0.
      return
      end
```

SUBROUTINE SA IS THE SIMULATED ANNEALING PROCEDURE

```
      subroutine sa(n,x,max,rt,eps,ns,nt,neps,maxevl,lb,ub,c,iprint,
     .            iseed1,iseed2,t,vm,xopt,fopt,nacc,nfcnev,nobds,ier,
     .            fstar,xp,nacp)


        double precision x(n),lb(n),ub(n),c(n),vm(n),fstar(neps),
     .              xopt(n),xp(n),t,eps,rt,fopt
        integer nacp(n),ns,nt,nacc,maxevl,iprint,nobds,ier,
     .      nfcnev,iseed1,iseed2
      logical max

      double precision f,fp,p,pp,ratio
        integer nup,ndown,nrej,nnew,lnobds,i,h,m
      logical quit
      double precision exprep
      real ranmar

      call rmarin(iseed1,iseed2)

c     set initial values
      nacc = 0
      nobds = 0
      nfcnev = 0
```

```
      ier = 99

      do 10 i = 1,n
      xopt(i) = x(i)
      nacp(I) = 0
   10 continue

      do 20 i = 1,neps
      fstar(i) = 1.d+20

   20 continue



c     if the initial temperature is not positive notify the user
c     and return to the calling routine

      if(t.le.0.0)then
         write(*,'(/," the initial temperature is not positive. "
     .         /," reset the variable t. "/)')
      ier = 3
      return
      endif

c     if the initial value is out of bounds notify the user and return
c     to the calling routine

        do 30 i = 1,n
      if( (x(I).gt.ub(i)).or.(x(i).lt.lb(i)))then
      call prt1
      ier = 2
      return
      endif
   30 continue

c     evaluate the function with input x and return value as F

      call fcn(n,x,f)


c     if the function is to be minimized, switch the sign of the function
c     note that all intermediate and final output switches the sign
```

```
c     back to eliminate any possible confusion to the user

      if(.not.max)f=-f
      nfcnev = nfcnev+1
      fopt = f
      fstar(1) = f
      if(iprint.ge.1)call prt2(max,n,x,f)

c     start the main loop.  Note that it terminates if (i) the
c     algorithm succesfully optimizes the function or (ii) there are
c     too many function evaluations (more than maxevl)

 100  nup = 0
      nrej = 0
      nnew = 0
      ndown = 0
      lnobds = 0

      do 400 m = 1,nt
        do 300 j = 1,ns
          do 200 h = 1,n

c     generate xp, the trial value of x. Note use of Vm to choose xp
          do 110 i = 1,n
            if(i.eq.h)then
              xp(i) = x(i) +(ranmar()*2-1.)*vm(i)
            else
              xp(i) = x(i)
            endif

c     if xp is out of bounds selecta point in bounds for the trial

          if((xp(i).lt.lb(i)).or.(xp(I).gt.ub(I)))then
            xp(i) = lb(i) +(ub(I)-lb(I))*ranmar()
            lnobds = lnobds + 1
            nobds  = nobds + 1
            if(iprint.ge.3)call prt3(max,n,xp,x,fp,f)
            endif
 110        continue

c     evaluate the function with the trial point xp and return as fp
          call fcn(n,xp,fp)
```

```
      if(.not.max)fp=-fp
      nfcnev = nfcnev+1
      if(iprint.ge.3)call prt4(max,n,xp,x,fp,f)

c     if too many function evaluations occur, terminate the algorithm
      if(nfcnev.ge.maxevl)then
      call prt5
      if(.not.max)fopt = -fopt
      ier =1
      return
      endif

c     accept the new point if the function value increases
      if(fp.ge.f)then
        if(iprint.ge.3)then
        write(*,'(" point accepted")')
        endif
        do 120 i = 1,n
         x(i) = xp(i)
 120    continue
        f = fp
        nacc = nacc+1
        nacp(h) = nacp(h) + 1
        nup= nup+1

c       if greater than any other point, record as new optimum
        if(fp.gt.fopt)then
        if(iprint.ge.3)then
         write(*,'(" new optimum")')
         endif

         do 130 i = 1,n
          xopt(i) = xp(i)
 130     continue
         fopt = fp
         nnew = nnew + 1
         endif


c     if the point is lower, use the Metropolis criteria to decide
c     on acceptance or rejection
```

```
        else
         p = exprep((fp-f)/t)
         pp = ranmar()
         if(pp.lt.p)then
         if(iprint.ge.3)call prt6(max)
         do 140 i = 1,n
          x(i) = xp(i)
140       continue
         f = fp
         nacc = nacc + 1
         nacp(h) = nacp(h) + 1
         ndown = ndown + 1
         else
         nrej = nrej + 1
         if(iprint.ge.3)call prt7(max)
         endif
        endif
200     continue
300     continue

c    adjust vm so that approximately half of all evaluations are
c    accepted

     do 310 i = 1,n
       ratio = dfloat(nacp(I))/dfloat(ns)
       if(ratio.gt..6)then
       vm(i) = vm(i)*(1.+c(i)*(ratio-.6)/.4)
       elseif(ratio.lt..4)then
       vm(i) = vm(i)/(1.+c(i)*((.4-ratio)/.4))
       endif
       if(vm(i).gt.(ub(I)-lb(I)))then
        vm(I)=ub(i)-lb(I)
        endif
310  continue

     if(iprint.ge.2)then
     call prt8(n,vm,xopt,x)
     endif

     do 320 i = 1,n
     nacp(i) = 0
320  continue
```

```
400  continue

     if(iprint.ge.1)then
     call prt9(max,n,t,xopt,vm,fopt,nup,ndown,nrej,lnobds,nnew)
     endif

c    check termination criteria

     quit = .false.
     fstar(1) = f
     if((fopt-fstar(1)).le.eps)quit = .true.
     do 410 i = 1,neps
     if(abs(f-fstar(i)).gt.eps)quit=.false.
410  continue

c    terminate sa if appropriate
     if(f.eq.0.0)quit=.true.
     if(quit)then
     do 420 i =1,n
     x(i) = xopt(i)

420  continue

     ier = 0
     if(.not.max)fopt = -fopt
     if(iprint.ge.1)call prt10
     return
     endif

c    if termination criteria is not met, prepare for another loop
     t = rt*t
     do 430 i = neps,2,-1
     fstar(i) = fstar(i-1)
430  continue

     f = fopt
     do 440 i = 1,n
     x(I) = xopt(I)
440  continue

c    loop again
```

```
      goto 100
      end


      function exprep(rdum)

c     this function replaces expto avoid under and overflows
c     and is designed to ibm 370 type machines. It may be necessary
c     to modify it for other machines. Note that the maximum and
c     minimum values of exprep are such that they has no effect
c     on the algorithm

        double precision rdum,exprep
      if(rdum.gt.174.)then
      exprep = 3.69d+75
      elseif(rdum.lt.-180.)then
      exprep = 0.0
      else
      exprep = exp(rdum)
      endif
      return
      end

      subroutine rmarin(ij,kl)

c     this subroutine and the next function generate random numbers
c     see the comments for SA for more information.  the only changes
c     from the original code is that the test to make sure that rmarin
c     runs first was taken out since SA assures that this is done

c     note:  the seed variables can have values between 0<=ij<=31328
c                                      0<=kl<=30081


      real u(97),c,cd,cm
      integer i97,j97
      common /raset1/ u,c,cd,cm,i97,j97
      if(ij.lt.0 .or. ij.gt.31328 .or.
     .  kl.lt.0 .or. kl.gt. 30081)then
      print '(A)','the first random number seed must have a value
     .          between 0 and 30081'
```

```
      stop
      endif

      i = mod(ij/177,177)+2
      j = mod(ij,177)+2
      k = mod(kl/169,178) + 1
      l = mod(kl,169)
      do 2 ii = 1,97
      s= 0.0
      t = 0.5
      do 3 jj = 1,24
      m = mod(mod(i*j,179)*k,179)
      i = j
      j = k
      k = m
      l = mod(53*l+1,169)
      if(mod(l*m,64).ge.32)then
      s = s+t
      endif
      t = 0.5*t
3     continue
      u(ii) = s
2     continue
      c = 362436.0/16777216.0
      cd = 7654321.0/16777216.0
      cm = 16777213.0/16777216.0

      i97 = 97
      j97 = 33
      return
      end

      function ranmar()
      real u(97),c,cd,cm
      integer i97,j97
      common /raset1/ u,c,cd,cm,i97,j97
      uni = u(i97)-u(j97)
      if(uni.lt.0.0)uni=uni+1.0
      u(i97)=uni
      i97=i97-1
      if(i97.eq.0)i97=97
      j97=j97-1
```

```
if(j97.eq.0)j97=97
c = c-cd
if(c.lt.0.0)c=c+cm
uni = uni-c
if(uni.lt.0.0)uni=uni+1.0
ranmar=uni
return
end

subroutine prt1

write(*,'(/," the starting value (x) is outside the bounds"
.       /," (lb and ub). execution is terminated without any"
.       /," optimization. respecify x,ub or lb so that "
.       /," lb(i).lt.x(i).lt.ub(i), i=1,n "/)')

return
end

  subroutine prt2(max,n,x,f)
  double precision x(*),f

logical max
write(*,'(" ")')
call prtvec(x,n,'initial x')
if(max)then
  write(*,'(" initial f: ",/, g25.18)') f
else
  write(*,'(" initial f: ",/, g25.18)') -f
endif
return
end

subroutine prt3(max,n,xp,x,fp,f)
double precision xp(*),x(*),fp,f
integer n
logical max
write(*,'(" ")')
call prtvec(x,n,'current x')
if(max)then
write(*,'(" current f: ",g25.18)')f
else
```

```
write(*,'(" current f: ",g25.18)')-f
endif
call prtvec(xp,n,'trail x')
write(*,'(" point rejected since out of bounds ")')
return
end

subroutine prt4(max,n,xp,x,fp,f)
double precision xp(*),x(*),fp,f
integer n
logical max

write(*,'(" ")')
call prtvec(x,n,'current x')
if(max) then
  write(*,'(" current f: ",g25.18)') f
  call prtvec(xp,n,'trial x')
  write(*,'(" resulting f: ",g25.18)')fp
  else
  write(*,'(" current f: ",g25.18)')-f
  call prtvec(xp,n,'trial x')
  write(*,'(" resulting f: ",g25.18)')-fp
  endif
  return
  end


  subroutine prt5
  write(*,'(/," too many function evaluations:  consider "
.       /," increasing maxevl or eps, or decreasing "
.       /," nt or rt these results are likely to b "
.       /," poor. ",/)')

return
end

subroutine prt6(max)
logical max
if(max) then
 write(*,'(" though lower, point accepted ")')
 else
   write(*,'(" though higher, point accepted ")')
```

```
endif
return
end

subroutine prt7(max)

logical max
if(max)then
write(*,'(" lower point rejected")')
else
write(*,'(" higher point rejected")')
endif
return
end

subroutine prt8(n,vm,xopt,x)
double precision vm(*),xopt(*),x(*)
integer n
write(*,'(/,
.   "intermediate results after step length adjustment",/)')
call prtvec(vm,n,'new step length (vm)')
call prtvec(xopt,n,'current optimal x')
call prtvec(x,n,'current x')
write(*,'(" ")')
return
end

subroutine prt9(max,n,t,xopt,vm,fopt,nup,ndown,nrej,lnobds,nnew)
double precision xopt(*), vm(*),t,fopt
integer n,nup,ndown,nrej,lnobds,nnew,totmov
logical max
totmov = nup+ndown+nrej
write(*,'(/,
. "intermediate results before next temperature reduction",/)')
write(*,'(" current temperature:      ",g12.5)')t
if(max)then
write(*,'(" max function value so far:    ",g25.18)')fopt
write(*,'(" total moves:                  ",i8)')totmov
write(*,'(" uphill:                       ",i8)')nup
write(*,'(" accepted downhill:            ",i8)')ndown
write(*,'(" rejected downhill:            ",i8)')nrej
write(*,'(" out of bounds trials:         ",i8)')lnobds
```

```
      write(*,'(" new maxima this temperature  ",i8)')nnew
      else
      write(*,'(" min function value so far    ",g25.18)')-fopt
      write(*,'(" total moves:                 ",i8)')totmov
      write(*,'(" downhill:                    ",i8)')nup
      write(*,'(" accepted uphill:             ",i8)')ndown
      write(*,'(" rejected uphill:             ",i8)')nrej
      write(*,'(" trials out of bounds:        ",i8)')lnobds
      write(*,'("new minima this temperature   ",i8)')nnew
      endif
      call prtvec(xopt,n,'current optimal x')
      call prtvec(vm,n,'step length (vm)')
      write(*,'(" ")')
      return
      end

      subroutine prt10
      write(*,'(/," sa achieved termination criteria ier = 0. ",/)')
      return
      end

      subroutine prtvec(vector,ncols,name)
      integer ncols
      double precision vector(ncols)
      character *(*) name
      write(*,1001)name
      if(ncols.gt.10)then
      lines = int(ncols/10)
      do 100 i = 1,lines
      ll = 10*(i-1)
      write(*,1000)(vector(j),j=1+ll,10+ll)
  100 continue

      write(*,1000)(vector(j),j=11+ll,ncols)
      else
      write(*,1000)(vector(j),j=1,ncols)
      endif

 1000 format(10(g12.5,1x))
 1001 format(/,25x,a)
      return
      end
```

SUBROUTINE CONSTR FINDS THE VALUE OF THE CONSTRAINT EQUATIONS

```
      subroutine constr(n,x,fvec,th3n,gamm,r,beta,thc,psi,pen)
      implicit double precision (a-h,o-z)

      double precision x(n),fvec(9),l,dq(9),th3n(9),psi(8)
      double precision gamm(9),thc(9),xsum(9),ysum(9),phi(8),
     . th2p(9)

      nn = 9
      nm1= 8
      pi = 4.d0*datan(1.d0)



      z1  =  x(1)
      th1 =  x(2)
      z1s =  x(3)
      th3 =  x(4)
      b =    x(5)
      l =    x(6)
      z =    x(7)
      th=    x(8)
      thc(1) = x(9)


c     do 1 k = 1,nm1
c     phi(k) = -22.625d0*dfloat(k)*pi/180.d0
c 1   continue
      phi(1) = -5.d0*pi/180.d0
      phi(2) = -20.d0*pi/180.d0
      phi(3) = -40.d0*pi/180.d0
      phi(4) = -60.d0*pi/180.d0
      phi(5) = -100.d0*pi/180.d0
      phi(6) = -150.d0*pi/180.d0
      phi(7) = -181.d0*pi/180.d0
      phi(8) = -200.d0*pi/180.d0
```

```
      wxsum = 0.d0
      wysum = 0.d0

      xsum(1)=z1*dcos(th1)+r*dcos(gamm(1))-z1s*dcos(th3n(1))-z*dcos(th)
      ysum(1)=z1*dsin(th1)+r*dsin(gamm(1))-z1s*dsin(th3n(1))-z*dsin(th)

      do 6 j = 2,nn
      xsum(j) = z1*dcos(th1+phi(j-1))+r*dcos(gamm(j))-z1s*dcos(th3n(j))
     .  -z*dcos(th)
      ysum(j) = z1*dsin(th1+phi(j-1))+r*dsin(gamm(j))-z1s*dsin(th3n(j))
     .  -z*dsin(th)

    6 continue


      do 11 k = 1,nn
      wxsum = wxsum+xsum(k)
      wysum = wysum+ysum(k)
   11 continue
      z2 = dsqrt(b**2+l**2)

      dq(1) = -(dsin(th3-th1))/(dsin(th3-gamm(1)))
        do 7 j = 2,nn
      dq(j) = -(dsin(th3+psi(j-1)-th1-phi(j-1)))/
     .        (dsin(th3+psi(j-1)-gamm(j)))
    7 continue
      do 8 k = 1,nn
      th2p(k) = gamm(k)+beta
    8 continue
      fvec(1) = z1*dsin(thc(1)-th1)+z2*dsin(thc(1)-th2p(1))*(z1/r)*dq(1)

      do 9 k = 2,nn
      fvec(k) = z1*dsin(thc(k)-th1-phi(k-1))+z2*dsin(thc(k)-th2p(k))
     .    *(z1/r)*dq(k)
    9 continue

      pen1 = 0.d0
      do 600 i = 1,nn
      if(fvec(i).gt.0.d0)then
```

```
      PEN1 = fvec(i)**2+pen1

    endif
600 continue
      pen = wxsum+wysum+(pen1)


    return
    end
```