INTERACTIVE COMPUTER AIDED

DIGITAL CONTROL DESIGN

by

HUSEYIN HAKAN YAKALI

A Thesis Submitted to the Faculty of the

College of Engineering

in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Engineering

Florida Atlantic University

Boca Raton, Florida

December 1988

INTERACTIVE COMPUTER AIDED

DIGITAL CONTROL DESIGN

by

HUSEYIN HAKAN YAKALI

This thesis was prepared under the direction of the Candidate's thesis advisor, Dr. Zvi S. Roth, Department of Electrical and Computer Engineering and has been approved by the members of his supervisory committee. It was submitted to the faculty of the College of Engineering and was accepted in partial fulfillment of the requirements for the degree of Master of Science in Engineering.

SUPERVISORY COMMITTEE :

Thesis Advisor
Dr. Zvi S. Roth

Chairperson, Department of
Electrical Engineering

Dr. Ahmet Duyar

Dean College of Engineering

Dr. Ali Zilouchian

Dean of Graduate Studies

12/9/88
Date

ACKNOWLEDGEMENTS

I would like to give my deepest gratidude to my thesis advisor, Dr.Zvi S. Roth, for his effort in this study. His unwavering patience, dedication, and good humor made it a pleasure to complete this work under his guidance. Special thanks also go to my thesis committee members, Dr.Ahmet Duyar and Dr.Ali Zilouchian.

Huseyin Hakan YAKALI
November , 1988

ABSTRACT

Author : Huseyin Hakan Yakali

Title : Interactive Computer Aided Digital Control Design.

Institution : Florida Atlantic University

Degree : Master of Science in Engineering

Year : 1988

This thesis outlines the design philosophy and implementation aspects of a new interactive CAD tool implemented in BASIC language on an IBM PC/AT computer for single input single output (SISO) digital control systems. The direct Digital Control design method presented is classical in nature. The program main features are:

(1) The use of Modified z-transform to model the effects of transport delay due to control computation time.
(2) The use of windows on a split screen to allow the designer observation of the closed-loop step response while systematically shaping a root locus or synthesizing closed-loop pole/zero patterns.
(3) Display of system response in between sampling instants.

TABLE of CONTENTS

LIST OF FIGURES

1.. INTRODUCTION

1.1.. Introduction and Motivation.

With a few design parameters and not very demanding specifications a controller can often be designed by hand using techniques like root locus, Bode plots, etc. This is the case with many continuous-time industrial single-loop PID controllers. On the other hand, even the most simple looking single-loop digital control systems may occasionally exhibit subtleties and intricacies that make systematic control design a very computation intensive process.

Referring to the basic single-loop digital control set up of Fig. 1.1 the first obvious difficulty is the transport delay due to the computation time of the control command within the **C(z)** block. The time t_c from the instance of reading the output until the computation of **u(kT)** is done may be significant portion of the system sampling period **T**. Furthermore t_c varies depending on the complexity of the control algorithm **C(z)**.

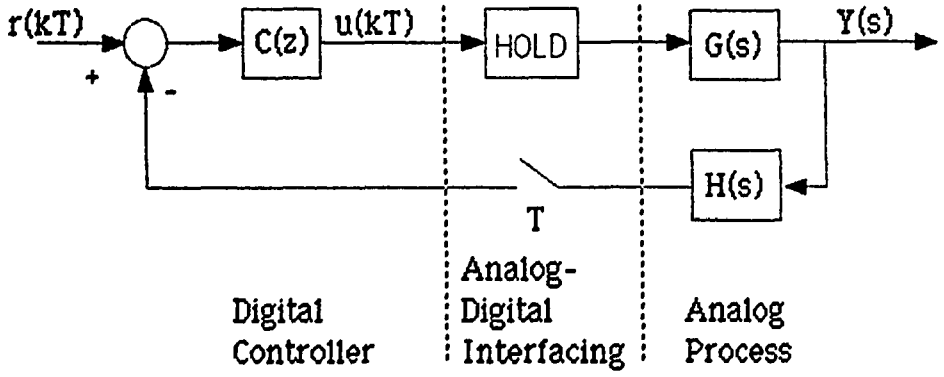


Fig. 1.1 A single-loop digital control system

Direct Digital Control Design requires the development of an overall discrete time model of the system (Fig. 1.2). The design of **C(z)** then proceeds in **z**-domain by selecting the sampling period **T** as well as an upper bound on the computation time t_c (as $\exp(-s\,t_c)$ may be appended to G(s)) [1-4].

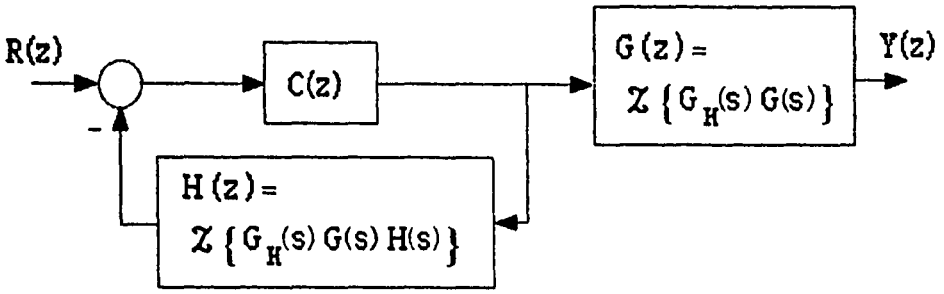


Fig. 1.2 A block diagram of an overall discrete-time model of a single-loop digital control system.

In many digital control implementations the system sampling period **T** is determined by an external timer and it is a variable design parameter. The selection of **T** is a non-trivial task [5]. One of the compicating factors in selecting **T** is the realization that one should not sample "too fast" [6]. Pulse-transfer functions **G(z)**, **H(z)** may contain zeros outside the unity circle even for an originally minimum phase analog processes **G(s)** and **H(s)**, where such a phenomenon aggravates the smaller **T** gets. Oversampling may then significantly complicate the controller design task. Taking too large values of **T** on the other hand has destabilizing effects such as "hidden oscillations" [7] that are not visible from the discretize-time model. Using either modified **z**-transform of fictitious samplers technique [1,2] for detection of oscillations in the response in between sampling instants is again demanding computationally.

A control design may start by guessing an initial reasonable value for the sampling period **T**. An upper bound t_c on the computation time needs to be guessed too as the complexity of the control algorithm in terms of number of additions and multiplications is apriori unknown. Fixing **T** and t_c enables the discretizing of the closed-loop system then to be followed by a systematic construction of **C(z)**. The process of selecting **T** and t_c is interactive. Upon finding a candidate **C(z)**, which appers to satisfy all design specifications, it is necessarry to estimate the computation time to verify that it is still consistent with the predetermined bound t_c. Inability to find **C(z)** on the other hand suggests modifying **T**. The problem of developing a systematic iteration procedure for **T** selection is still open.

For given **T** and t_o a classical direct digital control design may follow two general philosophies. One may synthesize a desired closed-loop transfer function **M(z)** where,

$$M(z) = \frac{Y(z)}{R(z)} = \frac{C(z)\,G(z)}{1 + C(z)\,H(z)} \tag{1.1}$$

and from the known transfer functions **G(z)** and **H(z)** solve for **C(z)**. The choice of **M(z)** is subject to design constraints imposed by realizability of **C(z)**, closed-loop internal stability and steady state error requirements [1,2,8].

Alternatively, one may modify the open-loop transfer function as in the Root Locus design method. While the latter method allows the designer a tight control over the complexity of **C(z)** by predetermining the number of its poles and zeros, in the closed-loop synthesis method it is easier to impose a closed-loop behaviour according to dominant poles, thus making it easier to satisfy the transient response design specifications.

Both design approaches complement each other and should be integrated.

1.2.. Systematic Controller Design Method

1.2.1.. Closed-loop Synthesis Method

In the Synthesis Design Method there are constraints that **M(z)** should satisfy. Some of these constraints come from the unstable zeros or poles of **G(z)** and **H(z)**, and others come from asymptotic tracking and disturbance attenuation requirements as applying the Internal Model Principle [9]. These constraints are explained in Ch. 3 in great detail.

Given **G(z)** and **H(z)** the synthesis of **M(z)** starts with the evaluation of transient response specifications such as peak overshoot, damping coefficient and settling time for the sake of determining possible location for the dominant poles and zeros of **M(z)**. The design may aim initially at establishing one or two closed-loop dominant poles by keeping all the rest of the closed-loop poles at or very near the origin in the **z**-plane. Facilitating tools commonly available in the literature are constant-ζ

(damping coefficient) curves and constant-ω_n (natural frequency curves [1-4]. These are valid for the case of two dominant poles and no dominant zeros. Peak overshoot and peak time nomograms [1] that are valid for the two poles and one zero case are also available. Very often though there are more than one dominant zero as in the case of including unstable zeros of **G(z)** and **H(z)** and in such a case the above curves and nomograms while being invalid may still be used as a first guess. This first guess then can be extended to more complicated transfer function by taking the constraints into consideration. Since **G(z)**, **H(z)** and **M(z)** are known, **C(z)** can be solved from Eq. (1.1).

Evidently the synthesis design method may lead to over comlicated controller functions **C(z)** due to unnecessary cancellations of original system poles and zeros. The method can be complemented by applying Root Locus design to simplify the result of the synthesis design.

1.2.2.. Root Locus Design

A Root Locus design philosophy is to modify the open-loop transfer function by moving, adding and deleting open-loop poles and zeros while monitoring the closed-loop poles location through root locus plots.

In the case of an unstable process, stabilization needs to be done first. Alternatively Root Locus design and closed-loop Synthesis design may be integrated. The design may start by using the synthesis method option. The resulting controller **C(z)** may then be simplified iteratively, thus providing candidates **C(z)** functions to be tested using Root Locus design.

1.2.3.. Features of the CAD Program

The program is interactive providing the user with menus at decision nodes along the design tree. The design starts by inputting **G(s)** and **H(s)** numerators and denominators in either a polynomial or factored format.

The next step is to choose a hold device from a menu of three entries: Zero-Order Hold, First-Order Hold or Exponential Hold. Future version of the program will allow a user defined hold function. The program then requests the user to choose the sampling period **T** and a modified **z**-transform parameter **m** (defined as $[1-t_c/T]$).

Upon specification of **T** and m the program evaluates and displays the discretized blocks **G(z)** and **H(z)**. Discretized model of the system can also be input and from this stage the design can be continued

At this point the user may specify an arbitrary controller **C(z)**. The program follows by computing the closed-loop poles and zeros and displaying the closed-loop discrete-time step response. Furthermore, the step response in between sampling instances may be computed and displayed to detect possible oscillations. The user needs to specify how many points in between samples are to be taken.

Rather than guessing **C(z)** as above the user may choose at this point one of two options: Closed-loop Synthesis design and a Root Locus design. As it was indicated previously the two methods are compatible and may be integrated in the sense that one may branch out in the midst of one design option using partial results within the other design option.

Upon selecting the Synthesis design method, program automatically satisfies all the constraints due to unstable zeros (and in future versions of the program of poles as well) of the system. A list of all the stable zeros of overall closed-loop transfer function is then displayed and the user is asked to select those zeros that will also be added to **M(z)**. Next the user is asked whether new zeros of **M(z)** should be added and which one zero, if any, should be considered "dominant" for the sake of calculating recommended locations for the dominant poles and approximate overshoot and peak-times.

In the next interactive step the user is asked to choose a desired closed-loop damping coefficient ζ, a desired peak-time expressed in terms of multiples of **T** and desired K_v (constant ramp-input error). Upon specifying these numbers, the program displays on the screen the respective ζ-curve and calculates and displayes the peak overshoot and peak-time based on a two pole and one zero model.

It should be pointed out that the calculated values of the overshoot and peak-time are not accurate since there may be more than one significant closed-loop zero. These are merely useful quantitative guidelines.

The program then announces the necessery number of "negligible"

poles that have to be added to **M(z)**. The user needs to determine where near the origin to place these poles needed to make **C(z)** realizable. Upon receiving this input the program automatically adjusts the gain of **M(z)** to nullify the steady state error to step inputs. The user now can introduce changes in the design such as adding lag compansation to improve on K_v. The step response is displayed together with the actual rise time, overshoot and K_v value. This response and its parameters may be saved in memory. Up to three design results may be saved and displayed simultaneously. The user can also change location of the dominant poles and observes the effect of the changes on the closed-loop step response.

For each one of the saved designs the program can display both the final closed-loop transfer function **M(z)** together with the computed controller **C(z)** in both polynomial and factored formats.

The user can also ask for viewing the step response in between sampling instants. The number of points between samples has been specified by user upon inputting **T** and **m** at the beginning of the program.

Many of the options in the Root Locus design are standard and similar to many other commercially available Root Locus programs. These include the mandatory specification of initial and final gain values and gain step size, automatic or user defined scaling and an optional movable cursor. The latter can be moved along a root locus branch providing upon request its exact location, the gain value at this point and list of all closed-loop poles that correspond to such a gain. The cursor moves from one computed root locus point to the next. At any time the user may increase the gain resolution providing for more points in between given old root locus points.

Special features of the program are the ability to observe the closed-loop step response at each root locus point and the ability to freely move, add or delete open loop poles and zeros. The program automatically monitors that at all times during this iterative process the controller **C(z)** remains realizable.

1.3.. Limitations of The Program.

Limitations of the current version of the program are explained in great detail in the last chapter.

2 . DISCRETIZATION OF SYSTEMS

2.1.1.. Digital Control Systems With Zero Order Hold (Z.O.H.) Device

From the impulse response shown in Fig. 2.1 ,the transfer function of the ZOH device is,

$$G_H(s) = \frac{1 - e^{-Ts}}{s} \qquad (2.1)$$

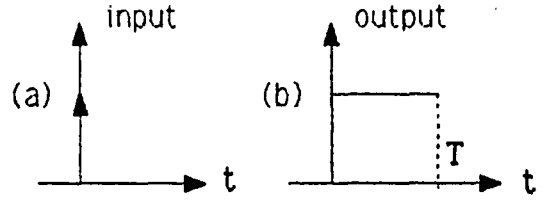


Fig 2.1 Unity impulse (a), and impulse response of ZOH (b).

As will be shown, the ZOH is a particular case of the Exponential Hold (EH) Device, that is covered in detail in the next section.

2.1.2.. Digital Control Systems With Exponential Hold (E.H.) Device

Transfer function and impulse response, (Fig. 2.2) , of an EH device is as follows ;

$$G_H(s) = \frac{1 - e^{-Ts} \cdot e^{-aT}}{s + a} \qquad (2.2)$$

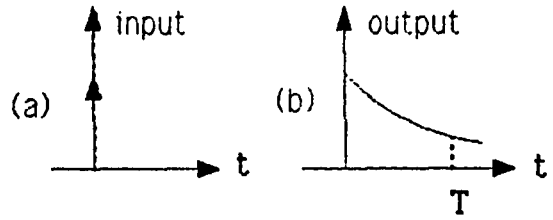


Fig 2.2 Unity impulse (a) , and impulse response of EH device (b).

In the following derivations substitution of a=0 provides the formulas for systems with ZOH.

Consider the single loop digital control system (Fig. 2.3). By standard Digital Control Theory [1-4], the block diagram of the discretized model is given as Fig. 2.4.

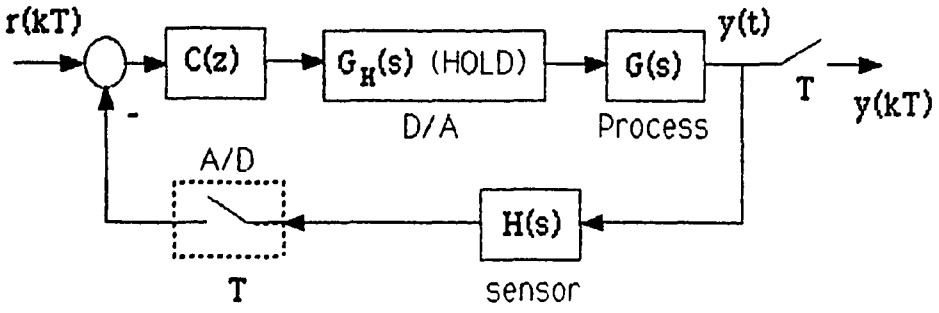


Fig 2.3 A single loop SISO digital control system.

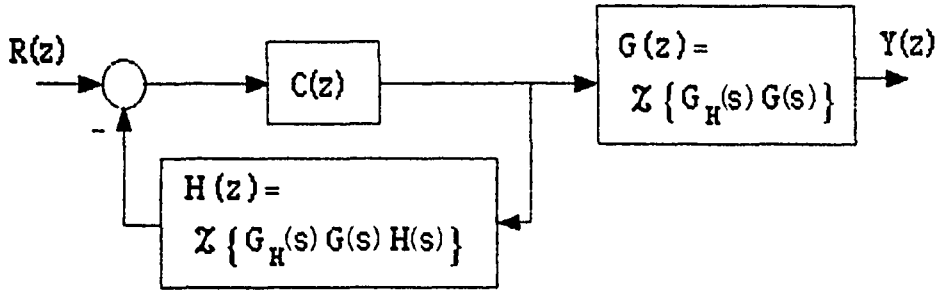


Fig 2.4 Discretized model of a SISO digital control system.

Both **G(s)** and **(G(s) H(s))** are assumed to be rational strictly proper functions of **s**. The computation of **G(z)** is shown next. The computation of **H(z)** follows exactly the same procedure and will not be shown here.

$$G(z) \triangleq (1 - z^{-1} e^{-aT})\, \mathcal{Z}\left\{ \frac{G(s)}{(s+a)} \right\} \triangleq \frac{N(z)}{D(z)} \tag{2.3}$$

Let the polynomials **N(s)** and **D(s)** be the numerator and denominator, respectively, of **G(s)**. For simplicity, one can assume that all the roots of **D(s)** are real, this requirement will be waived later. One can write **G(s)** in the following form:

$$G(s) = \frac{N(s)}{D(s)} = \frac{N(s)}{\prod_{i=1}^{n}(s + b_i)} \qquad (2.4)$$

By using partial fraction expansion method and taking the hold device into account,

$$\frac{G(s)}{(s+a)} = \frac{N(s)}{(s+a)D(s)} = \sum_{i=1}^{n+1} \frac{a'_i}{s + b_i}$$

where $b_{n+1} = a$ ... (2.5)

Then by taking the inverse Laplace transform,

$$g(t) \triangleq \mathfrak{L}^{-1}\left\{ \frac{G(s)}{(s+a)} \right\} \qquad (2.6)$$

$$g(t) \triangleq \mathfrak{L}^{-1}\left\{ \sum_{i=1}^{n+1} \left(\frac{a'_i}{s + b_i} \right) \right\} = \sum_{i=1}^{n+1} \left(\mathfrak{L}^{-1}\left\{ \frac{a'_i}{s + b_i} \right\} \right) \qquad (2.7)$$

$$g(t) = \sum_{i=1}^{n+1} a'_i e^{-tb_i} \qquad (2.8)$$

Sampling of $g(t)$ at the sampling period T gives the following sampled values,

$$g_i \triangleq g(iT)$$

$i = 0, 1, \ldots$ (2.9)

Again, by standard Digital Control Theory, the poles $s = b_i$ of $G(s)$ are mapped into poles $z = \exp(-b_i T)$ in $G(z)$. Therefore ;

$$D(z) = \prod_{i=1}^{n} (z - e^{-Tb_i}) \qquad (2.10)$$

By using long division ,

$$\mathcal{Z}\left\{\frac{G(s)}{(s+a)}\right\} = g_0 + g_1 z^{-1} + g_2 z^{-2} + \ldots = \sum_{i=0}^{\infty} g_i z^{-i} \tag{2.11}$$

where g_0=0 if **G(s)** is not improper. Thus from Eq.(2.3) and Eq.(2.10)

$$G(z) = \frac{N(z)}{D(z)} = (1 - z^{-1} e^{-Ta})(g_0 + g_1 z^{-1} + g_2 z^{-2} + \ldots) \tag{2.12}$$

The unknown polynomial **N(z)** can now be found using the known polynomial **D(z)**.

$$N(z) = D(z)(1 - z^{-1} e^{-Ta})(g_0 + g_1 z^{-1} + g_2 z^{-2} + \ldots) \tag{2.13}$$

As is well known, **G(z)** has the same number of poles, **n**, as in **G(s)** and the number of zeros of **G(z)** is in general **n-1** for strictly proper systems **G(s)**. Thus ;

$$\begin{aligned} \deg [\, N(z) \,] &= n-1 \\ \deg [\, D(z) \,] &= n \end{aligned} \tag{2.14}$$

The set $\{ g_0, g_1, \ldots \}$ contains highly redundant information. If suffices to take the first **n** samples to be able to solve for the unknown coefficients of **N(z)**. The proof will be shown together with the proof of similar result for modified **z**-transforms.

Modified **z**-transform are found in a similar manner. Define the modification coefficient **m** as :

$$m \triangleq 1 - \Delta \triangleq 1 - \frac{d}{T} \tag{2.15}$$

where **d** is a delay (smaller than **T**, the sampling period);

$$d = T\,(1-m) \tag{2.16}$$

Obviously ,

$$\mathfrak{L}^{-1}\left\{\frac{G(s)\,e^{-sd}}{(s+a)}\right\} = g(t-d) = \sum_{i=1}^{n+1} a'_i\, e^{-(t-d)b_i} \tag{2.17}$$

This time, the samples g_i are (by using Eq. (2.16)) ;

$$g_i \triangleq g(iT-d) \triangleq g((i+m-1)T)$$

$$i=1,2,\ldots \tag{2.18}$$

The modified z-transform is a rational function of z. Hence ,

$$\mathcal{Z}_m\left\{\frac{G(s)}{(s+a)}\right\} = \mathcal{Z}\left\{\frac{G(s)\,e^{-sd}}{(s+a)}\right\} = g_1 z^{-1} + g_2 z^{-2} + \ldots \tag{2.19}$$

Studying the modified z-transform more closely reveals ,

$$G(z,m) = (1-z^{-1}e^{-Ta})\,\mathcal{Z}_m\left\{\frac{G(s)}{(s+a)}\right\} \tag{2.20}$$

$$G(z,m) = (1-z^{-1}e^{-Ta})\,\mathcal{Z}_m\left\{\frac{N(s)}{(s+a)D(s)}\right\} \tag{2.21}$$

$$G(z,m) = \frac{z-e^{-Ta}}{z}\cdot\frac{N'(z)}{(z-e^{-Ta})\,D(z)} \tag{2.22}$$

$$G(z,m) = \frac{N'(z)}{z\,D(z)} \tag{2.23}$$

where **D(z)** is the same as in an ordinary z-transform, but **N'(z)** is new and unknown.

$$\frac{N'(z)}{z\,D(z)} = (1-z^{-1}e^{-Ta})(g_0 + g_1 z^{-1} + g_2 z^{-2} + \ldots) \tag{2.24}$$

$$D'(z) \triangleq z\,D(z) \tag{2.25}$$

$$N'(z) = D'(z)(1 - z^{-1}e^{-Ts})(g_1 z^{-1} + g_2 z^{-2} + \dots) \tag{2.26}$$

or

$$N'(z) = D(z)(1 - z^{-1}e^{-Ts})(g_1 + g_2 z^{-1} + \dots) \tag{2.27}$$

where ,

$$\begin{aligned} \deg [\, N'(z) \,] &= n \\ \deg [\, D'(z) \,] &= n+1 \end{aligned} \tag{2.28}$$

This time n samples $[\, g_0 , g_1 , \dots , g_{n-1} \,]$, are needed for the solution of $N(z)$.

For the ordinary and modified z-transform, n and $n+1$ samples, respectively, are needed to calculate $N(z)$. The proof is simple and shown for the ordinary z-transformation. In Eq.(2.3), the numerator and denominator polynomials in z, are represented in term of positive powers of z. Thus from Eq. (2.13) ,

$$N(z) = D(z)\,(1 - z^{-1}e^{-Ts})(g_0 + g_1 z^{-1} + g_2 z^{-2} + \dots)$$

$$N(z) = D(z)\,(g_0 + g'_1 z^{-1} + g'_2 z^{-2} + \dots)$$

where ,

$$g'_i = g_i - g_{i-1}\,e^{-Ts} \qquad , i = 1,2,\dots \tag{2.29}$$

$$D(z) = z^n + d_{n-1} z^{n-1} + \dots + d_1 z + d_0 \tag{2.30}$$

Thus ,

$$\begin{aligned} N(z) = {} & g_0 z^n + (g'_1 + d_{n-1} g_0) z^{n-1} + (g'_2 + d_{n-1} g'_1 + d_{n-2} g_0) z^{n-2} + \\ & (g'_n + d_{n-1} g'_{n-1} + d_{n-2} g'_{n-2} + \dots + d_1 g'_1 + d_0 g_0) z^0 + \dots \end{aligned} \tag{2.31}$$

Since $N(z)$ contains only positive powers of z then, obviously, only n samples

are needed. One can use the same derivation to show the result for modified z-transforms.

As is well known, the modified z-transform of G(z) for m=0 is ;

$$G(z,0) = \frac{N'(z)}{D'(z)} \tag{2.32}$$

and if the ordinary z-transform of G(z) is ;

$$G(z) = \frac{N(z)}{D(z)} \tag{2.33}$$

then ,

$$\begin{aligned} N'(z) &= N(z) \\ D'(z) &= z\,D(z) \end{aligned} \tag{2.34}$$

$$\begin{aligned} \deg[N'(z)] &= n-1 \\ \deg[D'(z)] &= n+1 \end{aligned} \tag{2.35}$$

2.1.3. Digital Control Systems With First Order Hold (F.O.H.) Device

The FOH impulse response is as shown in Fig.2.5.

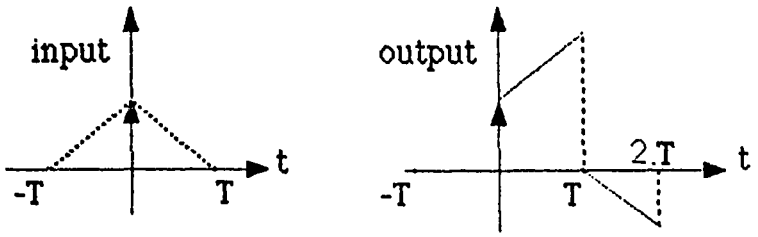


Fig.2.5 Unit impulse (a), and impulse response of FOH device (b).

Output of the FOH device, $g_h(t)$, may be expressed mathematically as follows:

$$g_h(t) = \left(\frac{t}{T} + 1\right)[u(t) - u(t-T)] + \left(\frac{-t}{T} + 1\right)[u(t-T) - u(t-2T)] \tag{2.36}$$

$$g_h(t) = \left(\frac{t}{T} + 1\right)u(t) - \left(\frac{2t}{T}\right)u(t-T) + \left(\frac{t-T}{T}\right)u(t-2T) \tag{2.37}$$

$$g_h(t) = \left(\frac{t}{T} + 1\right)u(t) - \left(\frac{2(t-T)}{T} + 2\right)u(t-T) + \left(\frac{t-2T}{T} + 1\right)u(t-2T) \tag{2.38}$$

Taking the Laplace transform, results in:

$$G_H(s) = \frac{1}{Ts^2} + \frac{1}{s} - \frac{2e^{-Ts}}{Ts^2} - \frac{2e^{-Ts}}{s} + \frac{e^{-2Ts}}{Ts^2} + \frac{e^{-2Ts}}{s} \tag{2.39}$$

$$G_H(s) = \frac{1}{Ts^2}\left(1 - 2e^{-Ts} + e^{-2Ts}\right) + \frac{1}{s}\left(1 - 2e^{-Ts} + e^{-2Ts}\right) \tag{2.40}$$

$$G_H(s) = \frac{1+Ts}{s^2}\left(1 - e^{-Ts}\right)^2 = \frac{s + 1/T}{s^2}\left(1 - e^{-Ts}\right)^2 \tag{2.41}$$

$$G_H(s) = \frac{1+Ts}{s^2}\left(1 - e^{-Ts}\right)^2 = \frac{s + 1/T}{s^2}\left(1 - e^{-Ts}\right)^2 \tag{2.42}$$

The discretized transfer function for a system with a FOH is then;

$$G(z) = \mathcal{Z}\left\{G_H(z)\,G(s)\right\} = \mathcal{Z}\left\{\frac{s + 1/T}{s^2}\left(1 - e^{-Ts}\right)^2 G(s)\right\} \tag{2.43}$$

$$G(z) = (1 - z^{-1})^2\,\mathcal{Z}\left\{\frac{s + 1/T}{s^2}\,G(s)\right\} \tag{2.44}$$

$$G(z) = \frac{N(z)}{z\,D(z)} = \frac{N(z)}{D'(z)} \tag{2.45}$$

or we can rewrite $D'(z)$ by using Eq.(2.9).

$$D'(z) = z\,D(z) = z\prod_{i=1}^{n}(z - e^{-Tb_i}) \tag{2.46}$$

and also taking the inverse Laplace transform of;

$$\frac{s + 1/T}{s^2}\,G(s) = \frac{s + 1/T}{s^2}\,\frac{N(s)}{D(s)} = \sum_{i=1}^{n+1}\frac{a'_i}{s + b_i} + \frac{a'_i}{s^2} \tag{2.47}$$

As it was done by using Eq.(2.6) and Eq.(2.9);

$$G(z) = \frac{N(z)}{z\,D(z)} = (1 - z^{-1})^2\,(g_0 + g_1 z^{-1} + \ldots + g_n z^{-n}) \tag{2.48}$$

and finally

$$N(z) = D'(z)\,(1 - z^{-1})^2\,(g_0 + g_1 z^{-1} + \ldots + g_n z^{-n}) \tag{2.49}$$

where ;

$$\begin{aligned} \deg[\,N(z)\,] &= n \\ \deg[\,D'(z)\,] &= n+1 \end{aligned} \tag{2.50}$$

Again **n** samples of **g(t)** are needed. Taking modified **z**-transform;

$$G(z,m) = (1 - z^{-1})^2\,\mathcal{Z}_m\left\{\frac{s + 1/T}{s^2}\,G(z)\right\} \tag{2.51}$$

$$G(z,m) = (1 - z^{-1})^2\,\frac{1}{(z-1)^2}\,\frac{N'(z)}{D(z)} \tag{2.52}$$

$$G(z,m) = \frac{N'(z)}{z^2\,D(z)} = \frac{N'(z)}{D''(z)} \tag{2.53}$$

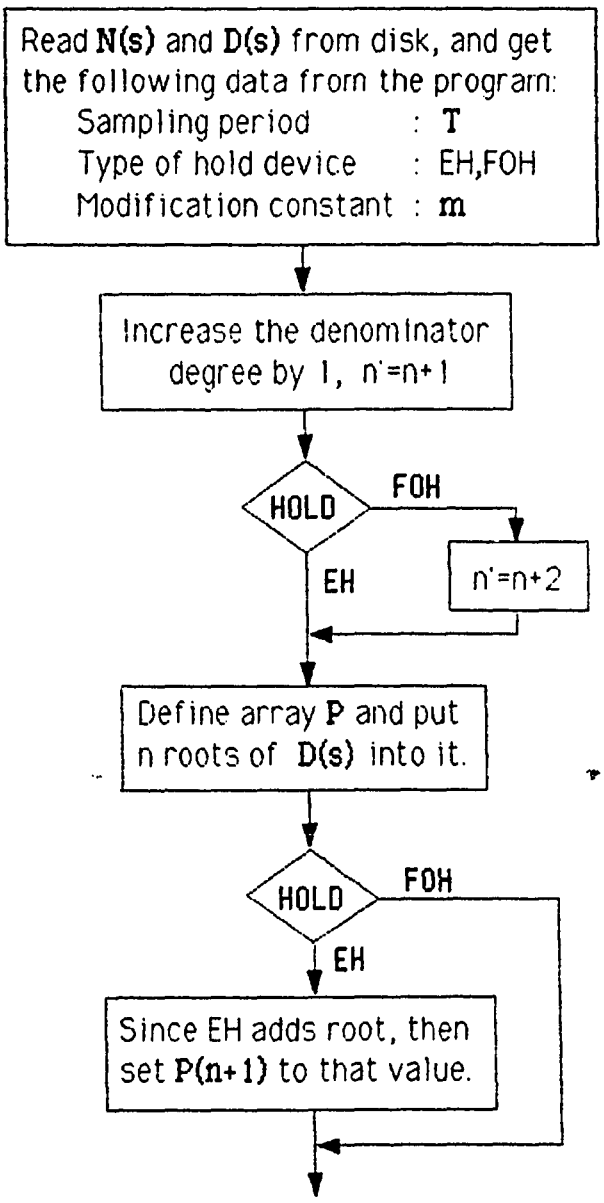
where ;

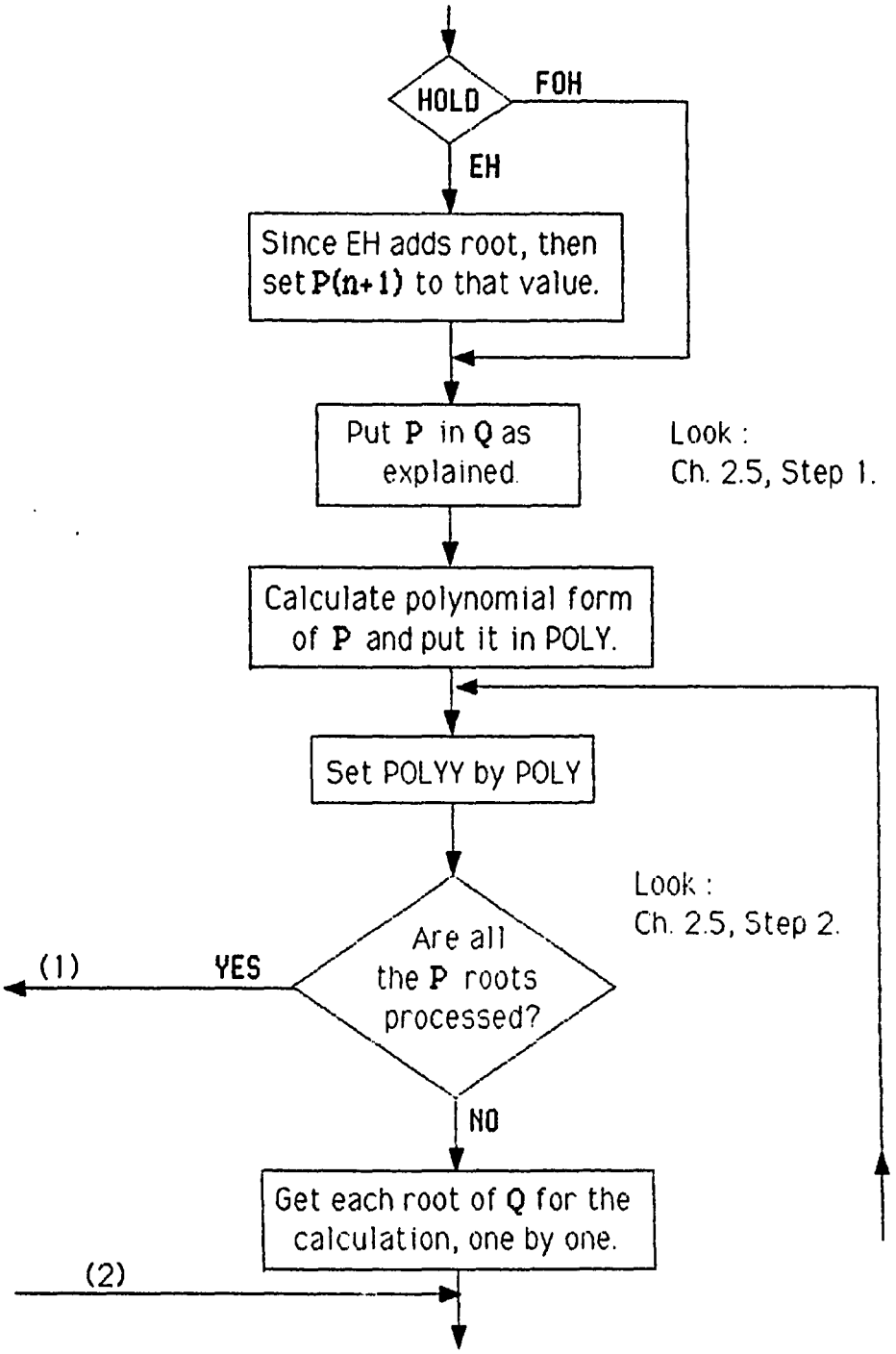
$$\begin{aligned} \deg[\,N'(z)\,] &= n+1 \\ \deg[\,D''(z)\,] &= n+2 \end{aligned} \tag{2.54}$$

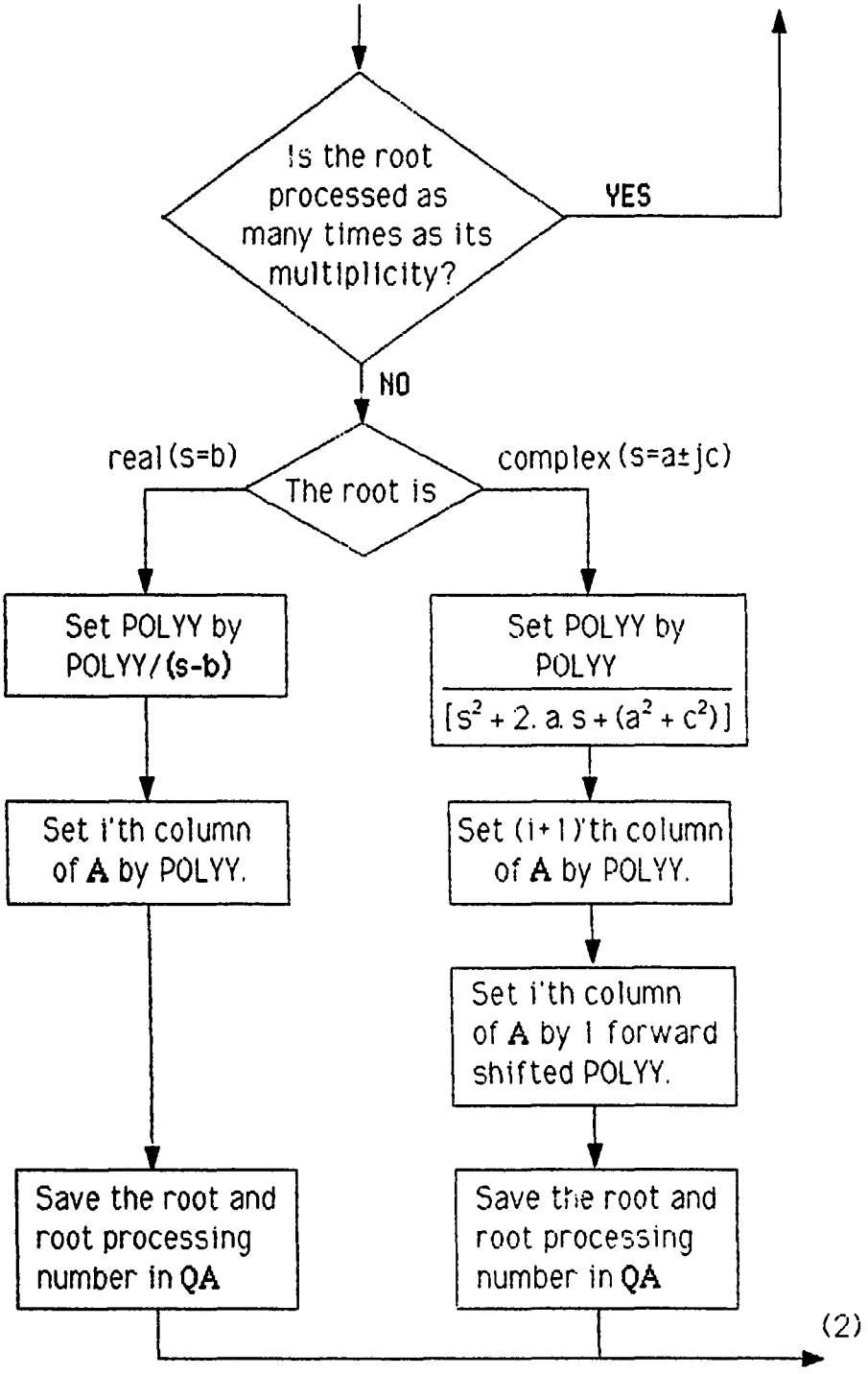
In this case **n+1** samples of **g(t)** are needed.

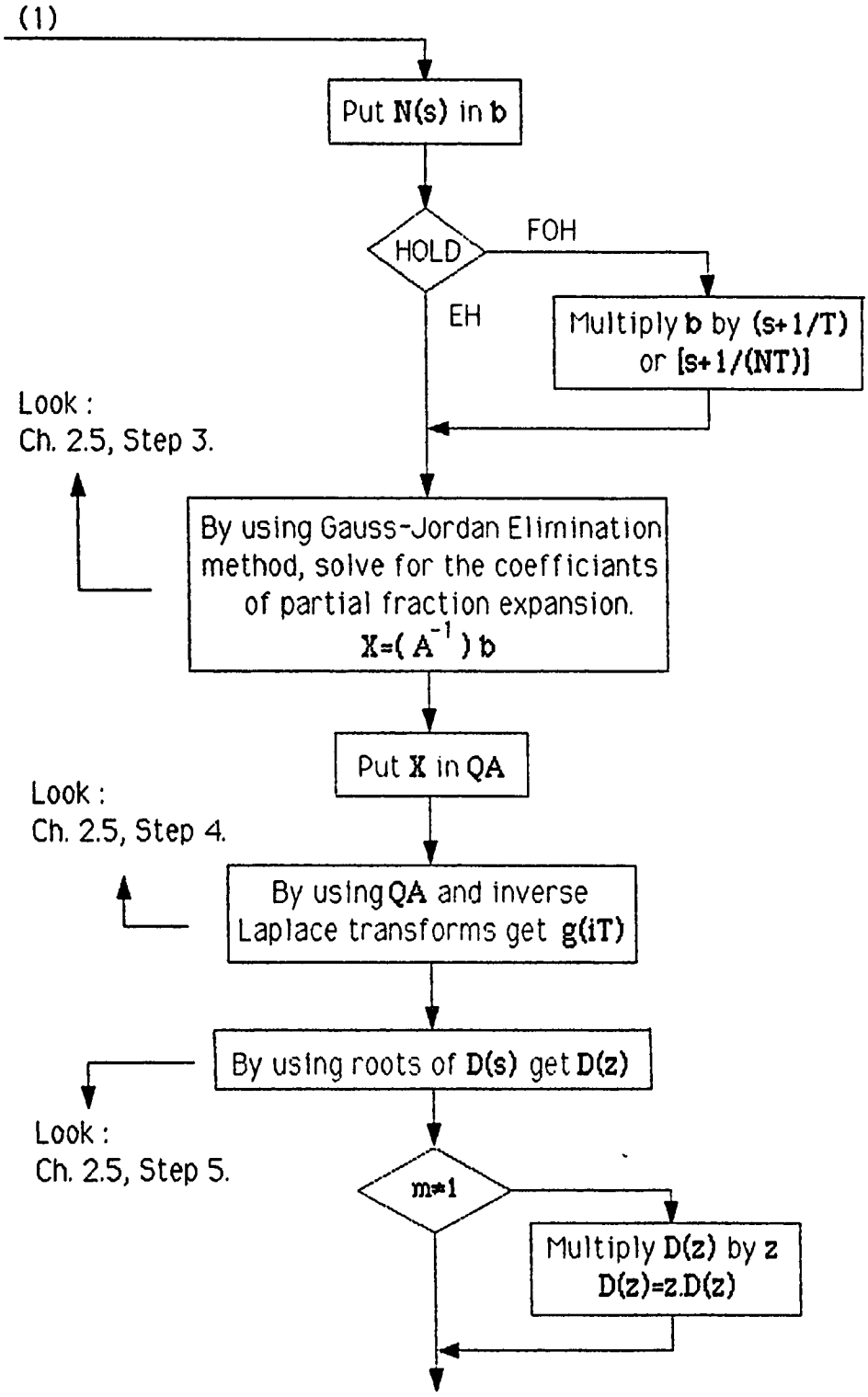
2.1.4.. Flowchart of the z-transform Calculation

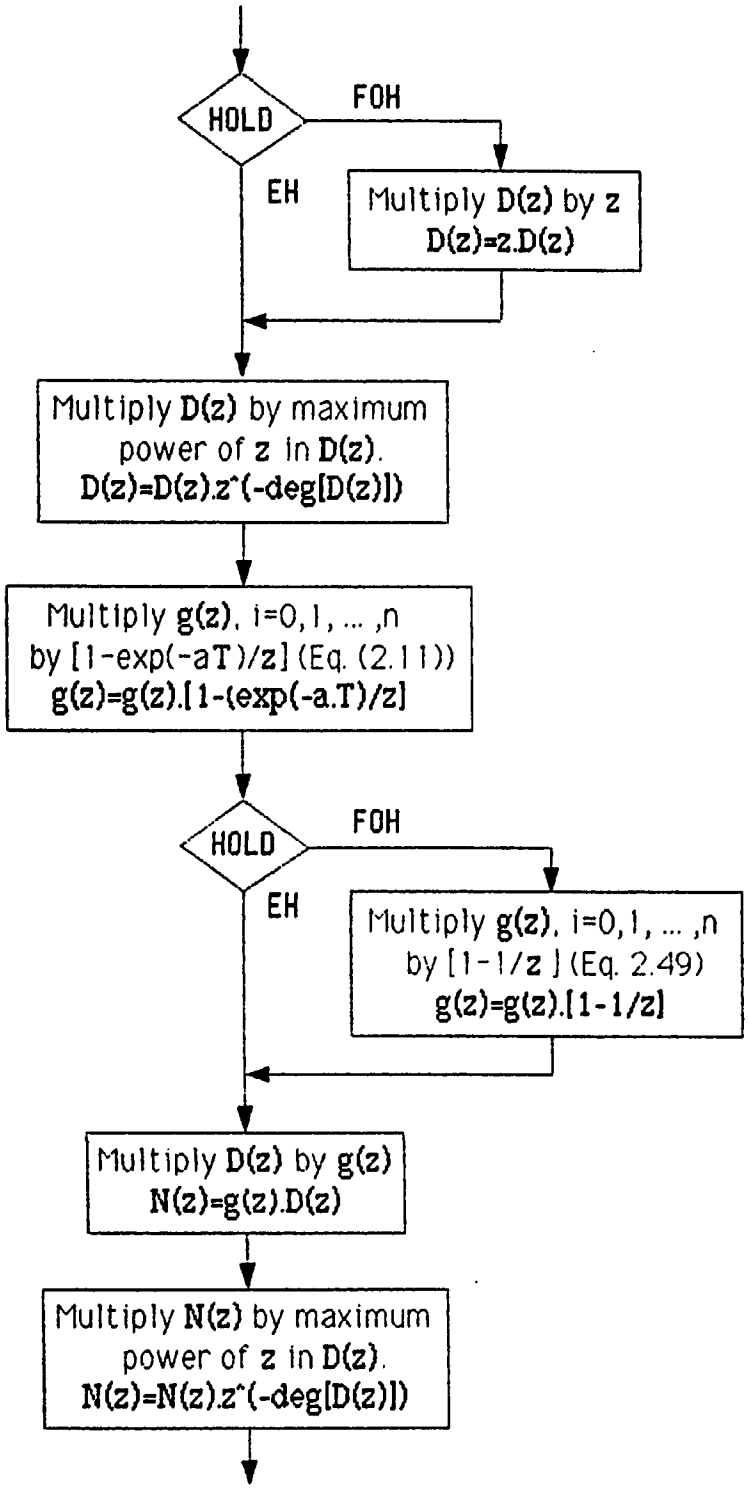
Let **G(s)** be as defined in Eq.(2.4), where **G(s)** denominator degree is **n**, and **G(s)** numerator degree is less than **n**. Following is the flow chart of **z**-transform of **G(s)** whose numerator and denominator are **N(s)** and **D(s)**, respectively.

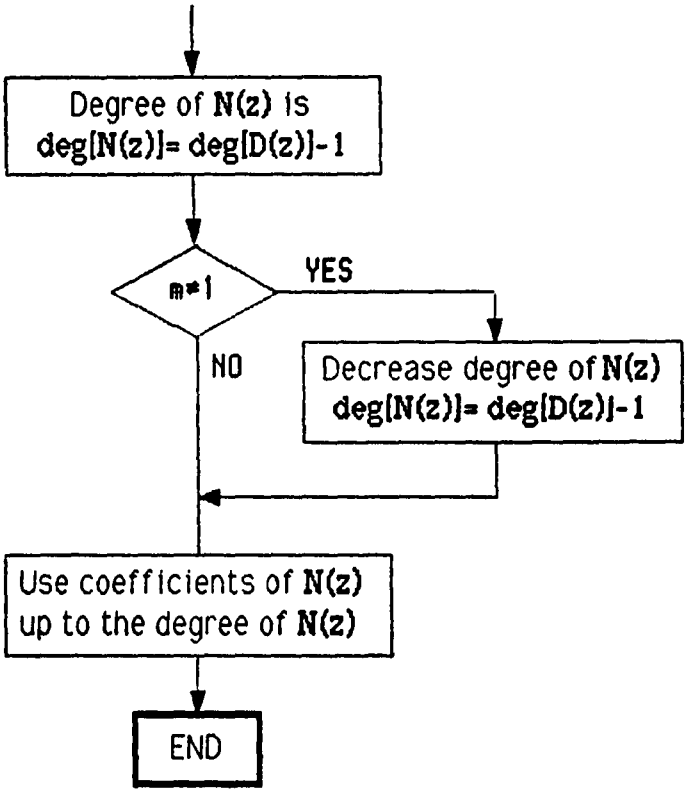



HOLD
FOH
EH
Since EH adds root, then set P(n+1) to that value.
Put P in Q as explained.
Look : Ch. 2.5, Step 1.
Calculate polynomial form of P and put it in POLY.
Set POLYY by POLY
Are all the P roots processed?
YES
(1)
NO
Look : Ch. 2.5, Step 2.
Get each root of Q for the calculation, one by one.
(2)


Is the root processed as many times as its multiplicity?
YES
NO
The root is
real (s=b)
complex (s=a±jc)
Set POLYY by POLYY/(s-b)
Set POLYY by POLYY / [s² + 2. a. s + (a² + c²)]
Set i'th column of A by POLYY.
Set (i+1)'th column of A by POLYY.
Set i'th column of A by 1 forward shifted POLYY.
Save the root and root processing number in QA
Save the root and root processing number in QA
(2)


(1)
Put N(s) in b
HOLD
FOH
EH
Multiply b by (s+1/T) or [s+1/(NT)]
Look : Ch. 2.5, Step 3.
By using Gauss-Jordan Elimination method, solve for the coefficiants of partial fraction expansion.
X=(A^-1) b
Put X in QA
Look : Ch. 2.5, Step 4.
By using QA and inverse Laplace transforms get g(iT)
By using roots of D(s) get D(z)
Look : Ch. 2.5, Step 5.
m≠1
Multiply D(z) by z
D(z)=z.D(z)


HOLD
FOH
EH
Multiply D(z) by z
D(z)=z.D(z)
Multiply D(z) by maximum power of z in D(z).
D(z)=D(z).z^(-deg[D(z)])
Multiply g(z), i=0,1, ... ,n by [1-exp(-aT)/z] (Eq. (2.11))
g(z)=g(z).[1-(exp(-a.T)/z]
HOLD
FOH
EH
Multiply g(z), i=0,1, ... ,n by [1-1/z] (Eq. 2.49)
g(z)=g(z).[1-1/z]
Multiply D(z) by g(z)
N(z)=g(z).D(z)
Multiply N(z) by maximum power of z in D(z).
N(z)=N(z).z^(-deg[D(z)])



2.1.5. Examples

The following examples illustrate all the steps in the above procedure:

Step 1: For the sake of demonstration, let ,

$$G(s) = \frac{s^2 + 4s + 1}{s^2(s+1)^2(s^2+2s+2)}$$

if there is a ZOH device in the loop, then ,

$$\frac{G(s)}{s} = \frac{s^2 + 4s + 1}{s^3(s+1)^2(s^2+2s+2)}$$

Then the array **P** consists of the following pole values of **G(s)/s** (roots of **D(s)** with root of ZOH, s).

P(1)= 0
P(2)= 0
P(3)= -1
P(4)= -1
P(5)= -1 -j
P(6)= -1 +j
P(7)= 0

Rearranging P gives us the array Q which is ,

Q(1) = 0 , 3
Q(2) = -1 , 2
Q(3) = -1 ± j , 1

The right most column gives the root multiplicity (i.e., there are 3 poles at z=0).

Step 2: Continuing with G(s)/s ,

$$N(s) = s^2 + 4s + 1$$
$$sD(s) = s^3(s+1)^2(s^2+2s+2)$$

one needs to apply partial fraction expansion method on G(s)/s;

$$\frac{G(s)}{(s)} = \frac{N(s)}{sD(s)} = \frac{X_1}{(s)} + \frac{X_2}{(s)^2} + \frac{X_3}{(s)^3} + \frac{X_4}{(s+1)} + \frac{X_5}{(s+1)^2} + \frac{X_6 s + X_7}{s^2+2s+2}$$

or one can show that ,

$$\frac{G(s)}{s} = \frac{X_1 f_1 + X_2 f_2 + X_3 f_3 + X_4 f_4 + X_5 f_5 + (X_6 s + X_7) f_6}{sD(s)}$$

where ,

$$f_1 = \frac{s\,D(s)}{s} = s^6 + 4s^5 + 7s^4 + 6s^3 + 2s^2$$

$$f_2 = \frac{s\,D(s)}{s^2} = s^5 + 4s^4 + 7s^3 + 6s^2 + 2s$$

$$f_3 = \frac{s\,D(s)}{s^3} = s^4 + 4s^3 + 7s^2 + 6s + 2$$

$$f_4 = \frac{s\,D(s)}{(s+1)} = s^6 + 3s^5 + 4s^4 + 2s^3$$

$$f_5 = \frac{s\,D(s)}{(s+1)^2} = s^5 + 2s^4 + 2s^3$$

$$f_6 = \frac{s\,D(s)}{s^2 + 2s + 1} = s^6 + 2s^5 + s^4$$

substituting these in the previous equation and rearrenging ,

$$\begin{aligned}
N(s) = s^6 &(X_1 + X_4 + X_6) + \\
s^5 &(4X_1 + X_2 + 3X_4 + X_5 + 2X_6) + \\
s^4 &(7X_1 + 4X_2 + X_3 + 4X_4 + 2X_5 + X_6 + 2X_7) + \\
s^3 &(6X_1 + 7X_2 + 4X_3 + 2X_4 + 2X_5 + X_7) + \\
s^2 &(4X_1 + 6X_2 + 6X_3) + \\
s &(2X_2 + 7X_3) + \\
&(2X_3) \\
N(s) = s^2 + 4s + 1 &
\end{aligned}$$

so,

$$\begin{bmatrix}
1 & & & 1 & & 1 & & \\
4 & 1 & & 3 & 1 & 2 & 1 & \\
7 & 4 & 1 & 4 & 2 & 1 & 1 & 2 \\
6 & 7 & 4 & 2 & 2 & & 1 & \\
2 & 6 & 7 & & & & & \\
 & 2 & 6 & & & & & \\
 & & 2 & & & & &
\end{bmatrix}
\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \end{bmatrix}
=
\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 4 \\ 1 \end{bmatrix}$$

One can see that there are 3 roots at s=0 and if **s D(s)** is divided by **s** 3 times, the result of each division gives the first three columns of **A**. All there is needed is the total number of roots and where they are located. This is the information that array **Q** has. **Q** has a total of 3 variables depending on the variety of root locations. But, now, a new array, **QA**, is needed with 7 variables or rows.

QA ..	1	:	1	0	$\mathbf{X}_1$
	2	:	2	0	$\mathbf{X}_2$
	3	:	3	0	$\mathbf{X}_3$
	4	:	1	-1	$\mathbf{X}_4$
	5	:	2	-1	$\mathbf{X}_5$
	6	:	1	$-1 \pm j$	$\mathbf{X}_6$
	7	:			$\mathbf{X}_7$
			I	II	III

Column I , gives multiplicity number of the root ,
Column II , gives location of the root ,
Column III , gives corresponding coefficient for partial fraction expansion.

Step 3: In Step 2 , calculation of **X** is skipped. Gauss-Jordan Elimination [9,10] method is used for this calculation.

This method is based on nullifying all the non-diagonal matrix coefficients.

Let $\mathbf{L}_i$, i=1,2, .. ,7 ,denote each line of **A** and b. Then from 2nd to 5th line, applying the method, which is ,

$$\mathbf{L}_{2,NEW} = \mathbf{L}_2 - 4\mathbf{L}_1$$
$$\mathbf{L}_{3,NEW} = \mathbf{L}_3 - 7\mathbf{L}_1$$
$$\mathbf{L}_{4,NEW} = \mathbf{L}_4 - 6\mathbf{L}_1$$
$$\mathbf{L}_{5,NEW} = \mathbf{L}_5 - 2\mathbf{L}_1$$

then ,

1			1		1		0
	1		-1	1	-2	1	0
	4	1	-3	2	-6	2	0
	7	4	-4	2	-7		0
	6	7	-2		-2		1
	2	6					4
		2					1

repeating the same process will give the **x** vector. The exact value of **X** does not have any particular importance in the example.

Step 4: As one can see from Laplace transform tables, for real roots,

$$\mathfrak{L}^{-1}\left\{\frac{k}{(s+a)^n}\right\} = \left(\frac{k}{(n-1)!}\right) t^{n-1} e^{-ta} \triangleq 1_1(k, a, n, t)$$

and for complex roots ,

$$\mathfrak{L}^{-1}\left\{\frac{k_1 s + k_2}{\left[(s+a)^2 + b^2\right]}\right\} = e^{-ta}\left\{k_1 \cos(bt) + \left[\frac{k_2 - k_1 a}{b}\right]\sin(bt)\right\} \triangleq 1_2(k_1, k_2, a, b, t)$$

$$\mathfrak{L}^{-1}\left\{\frac{k_1 s + k_2}{\left[(s+a)^2 + b^2\right]^2}\right\} = \frac{e^{-ta}}{2b}\left\{\begin{array}{l} k_1 t \sin(bt) + \\ \left[\frac{k_2 - k_1 a}{b^2}\right](\sin(bt) - bt\cos(bt)) \end{array}\right\} \triangleq 1_3(k_1, k_2, a, b, t)$$

Using these three formulas one can get **g(t)** or **g(iT)**. Because of the above equations, there are some program restrictions that apply in this particular case,

- not more than 10 multiple roots,
- not more than 2 multiple complex roots.

First restriction comes from computation of **(n-1)!**. One may change the upper bound of factorial calculations. To improve on the second restriction, one needs to calculate inverse Laplace transforms for higher order systems and substitute in the program.

These restrictions may be waived in later versions of the program.

Using these three equations with the information that is stored in **QA**, one can calculate the samples $[g_0, g_1, .., g_n]$, as follows,

QA ..

		$g_{0,k}$	$g_{1,k}$	$g_{2,k}$ ...
k=1 : 1, 0	, X_1 ...	$1_1(X_1,0,1,0)$	$1_1(X_1,0,1,T)$	$1_1(X_1,0,1,2T)$
k=2 : 2, 0	, X_2 ...	$1_1(X_2,0,2,0)$	$1_1(X_2,0,2,T)$	$1_1(X_2,0,2,2T)$
k=3 : 3, 0	, X_3 ...	$1_1(X_3,0,3,0)$	$1_1(X_3,0,3,T)$	$1_1(X_3,0,3,2T)$
k=4 : 1, -1	, X_4 ...	$1_1(X_4,1,1,0)$	$1_1(X_4,1,1,T)$	$1_1(X_4,1,1,2T)$
k=5 : 2, -1	, X_5 ...	$1_1(X_5,1,2,0)$	$1_1(X_5,1,2,T)$	$1_1(X_5,1,2,2T)$
k=6 : 1, -1 ± j	, X_6 ...	$1_2(X_6,X_7,1,1,0)$	$1_2(X_6,X_7,1,1,T)$	$1_2(X_6,X_7,1,1,2T)$
k=7 :	, X_7	0	0	0

Thus ,

$$g(iT) = \sum_{k=1}^{7} g_{i,k} \quad \dots \quad i = 0, 1, 2, \dots, 7$$

Step 5: System denominator **D(s)** is,

$$D(s) = s^2(s+1)^2(s^2+2s+2)$$

or ,

$$D(s) = s^2(s+1)^2[(s+1)^2+1]$$

then applying the following two transformations,

- for real roots ,

  $s+a \quad \dots \quad z - e^{-aT}$

- for complex roots ,

  $(s+a)^2 + b^2 \quad \dots \quad z^2 - ze^{-aT}\cos(bT) + e^{-2aT}$

using these two equation, **D(z)** can be written as,

$$D(z) = (z-1)^2 (z-e^{-T})^2 \left(z^2 - ze^{-T}\cos(T) + e^{-2T}\right)$$

2.2., Closed Loop Discretized Transfer Function

Redrawing the closed-loop block diagram, (Fig 2.6)

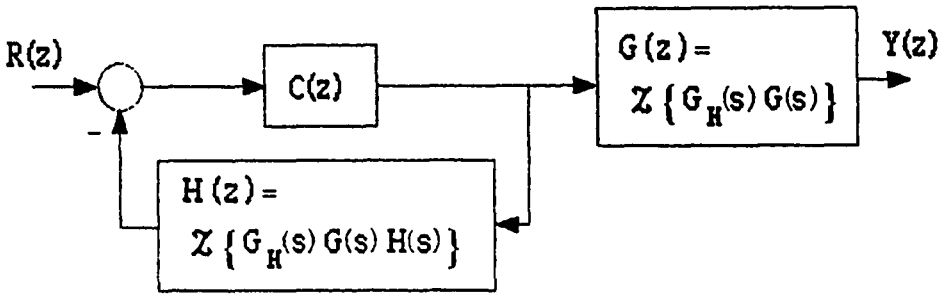


Fig. 2.6 Dicsretized block diagram

where,

$$G(z) = Z\{G_H(s)\, G(s)\}$$
$$H(z) = Z\{G_H(s)\, G(s)\, H(s)\} \qquad (2.55)$$

($G_H(s)$ is the Laplace transform of the response of the hold device).

From the block diagram, the closed loop transfer function **Y(z)/R(z)** of the system is as follows.

$$M(z) = \frac{Y(z)}{R(z)} = \frac{C(z)\,G(z)}{1 + C(z)\,H(z)} \qquad (2.56)$$

G(z) denominator roots are the transformed roots of **G(s)** denominator, and **H(z)** denominator roots are the transformed roots of both **G(s)** and **H(s)** denominator roots. Let **HD(s)** be the denominator of **H(s)**, then one can easily write **HD(z)** from **HD(s)**. It is assumed that non of **HD(s)** roots are cancelled by **GN(s)** roots, where **GN(s)** denotes the numerator of **G(s)**. Rewriting the above equation by using these notations,

$$M(z) = \frac{CN(z)\,GN(z)\,[[HD(s)](z)]}{CD(z)\,HD(z) + CN(z)\,HN(z)} \qquad (2.57)$$

As it is seen, transformed roots of **H(s)** and the denominator and numerator of **G(z)** and **H(z)** provide the closed-loop zeros and poles, respectively.

2.3.. System Step Response in Between Sampling Instants

The transfer functions **T(z), H(z),** and **G(z)** describe system response at the sampling instants only. Several techniques have been developed [2] to find the system response in between sampling instances, to detect possible hidden oscillations. One such technique is based on modified **z**-transform theory. A more efficient method for computer aided analysis and design is based on introducing virtual fast samplers as will be described in this section.

Suppose that **(N-1)** points are desired in between samples. For that, a "dummy" fast sampler is introduced as shown in Fig 2.7 .

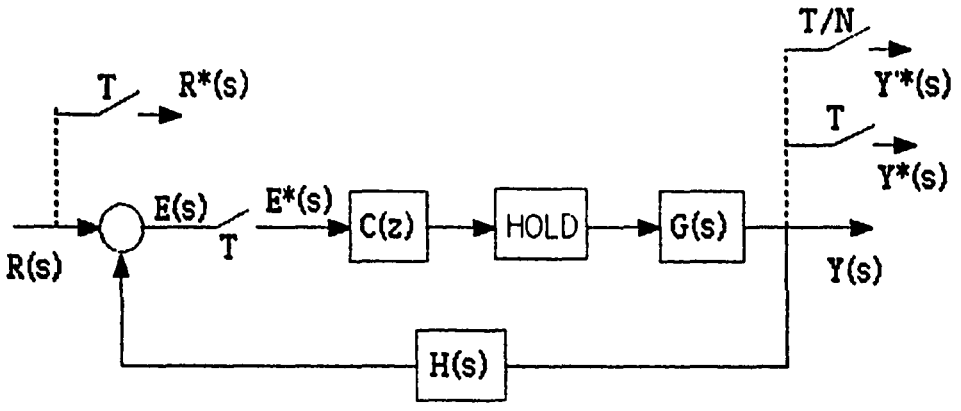


Fig. 2.7 Block diagram with fast sampler

Where the **E*(s)** notation denotes the **z**-transform of the samples of the inverse Laplace transform of **E(s)** [1-4].

Then ,

$$
\begin{aligned}
Y(s) &= E^*(s)\, C(z)\, G_H(s)\, G(s) \\
Y^*(s) &= E^*(s)\, C(z) \left[G_H(s)\, G(s) \right]^* \\
E(s) &= R(s) - E^*(s)\, C(z)\, G_H(s)\, G(s)\, H(s) \\
E^*(s) &= R^*(s) - E^*(s)\, C(z) \left[G_H(s)\, G(s)\, H(s) \right]^* \\
Y'(s) &= E^*(s)\, C(z^N)\, G_H(s)\, G(s) \\
Y'^*(s) &= E^*(s)\, C(z^N) \left[G_H(s)\, G(s) \right]^*
\end{aligned}
\tag{2.58}
$$

Taking z-transform,

$$
\begin{aligned}
Y(z) &= E(z)\, C(z)\, \mathcal{Z}\left\{ G_H(s)\, G(s) \right\} \\
&= E(z)\, C(z)\, G(z) \\
E(z) &= R(z) - E(z)\, C(z)\, \mathcal{Z}\left\{ G_H(s).\, G(s).\, H(s) \right\} \\
&= R(z) - E(z)\, C(z)\, H(z) \\
Y'(z) &= E(z^N)\, C(z^N)\, \mathcal{Z}\left\{ G_H(s)\, G(s) \right\} \\
&= E(z^N)\, C(z^N)\, G'(z)
\end{aligned}
\tag{2.59}
$$

assuming a EH device is used, and rewriting $G'(z)$,

$$
G'(z) = (1 - z^{-N} e^{-aT})\, \mathcal{Z}\left\{ \frac{G(s)}{s+a} \right\}_{T/N} = \frac{1 - z^{-N} e^{-aT}}{1 - z^{-1} e^{-a(T/N)}}\, \mathcal{Z}\left\{ G_H(s) G(s) \right\}_{T/N}
\tag{2.60}
$$

note that sampling period is T/N for $G'(z)$.

Hold device in the block diagram works for T seconds. Therefore, $-N^{th}$ power shows up in the above equation, instead of -1. As is well known, z^{-1} represents a pure time delay by T. However since the sampler works every T/N seconds, it causes z to be replaced by z^N for the part of system that works every T seconds. Thus $Y'(z)$,

$$
Y'(z) = R(z^N)\, \frac{C(z^N)}{1 + C(z^N)\, H(z^N)}\, \mathcal{Z}\left\{ G_H(s)\; G(s) \right\}
\tag{2.61}
$$

where z-transform is taken with respect to T/N.

$$Y^*(z) = R(z^N)\,\frac{C(z^N)\,G(z^N)}{1 + C(z^N)\,H(z^N)}\,\frac{\mathcal{Z}\{G_H(s)\,G(s)\}}{G(z^N)} \tag{2.62}$$

substituting $T(z^N)$ in above equation, and rewriting it,

$$Y^*(z) = R(z^N)\,M(z^N)\,\frac{\mathcal{Z}\{G_H(s)\,G(s)\}}{G(z^N)} \tag{2.63}$$

defining new transfer function, P(z),

$$P(z) \triangleq \frac{M(z)}{G(z)} \tag{2.64}$$

then ,

$$Y^*(z) = R(z^N)\,P(z^N)\;\mathcal{Z}\{G_H(s)\,G(s)\} \tag{2.65}$$

in order to substitute $G^*(z)$ in the above equation,

$$Y^*(z) = R(z^N)\,P(z^N)\,(1 - z^N e^{-aT})\;\mathcal{Z}\left\{\frac{G(s)}{s+a}\right\} \tag{2.66}$$

and finally ,

$$Y^*(z) = R(z^N)\,P(z^N)\,\frac{1 - z^{-N} e^{-aT}}{1 - z^{-1} e^{-a(T/N)}}\,G^*(z) \tag{2.67}$$

or,

$$Y^*(z) = V(z)\,P(z^N)\,G^*(z) \tag{2.68}$$

where,

$$V(z) \triangleq \frac{1 - z^{-N} e^{-aT}}{1 - z^{-1} e^{-a(T/N)}}\,\frac{z^N}{z^N - 1} \tag{2.69}$$

$$V(z) = \frac{z^N - e^{-aT}}{z^N - 1} \frac{z}{z - e^{-a(T/N)}} \tag{2.70}$$

Using the Eq. (2.65) for the system where FOH is used, **V(z)** can also be calculated. **V(z)** with the FOH systems is ,

$$Y'(z) = R(z^N)\, P(z^N)\, \mathcal{Z}\{ G_H(s)\, G(s) \} \tag{2.71}$$

$$Y'(z) = R(z^N)\, P(z^N) \frac{(1 - z^{-N})^2}{(1 - z^{-1})^2} G'(z) \tag{2.72}$$

where **G'(z)** is calculated from the following equation. Note that sampling frequency in that equation is **T/N**, but the time dependence is coming from the FOH device. That is why **T** is multiplied by **N**.

$$\begin{aligned} G'(z) &\triangleq (1 - z^{-N})^2\, \mathcal{Z}\left\{ \frac{s + 1/(N\,T)}{s^2} G(s) \right\} \\ &= \frac{(1 - z^{-N})^2}{(1 - z^{-1})^2}\, \mathcal{Z}\left\{ \left[\frac{s + 1/(N\,T)}{s^2} \left(1 - e^{-(T/N)}\right)^2 \right] G(s) \right\} \end{aligned} \tag{2.73}$$

thus the following can be writen ,

$$Y'(z) = V(z)\, P(z^N)\, G'(z) \tag{2.74}$$

where, for the FOH device **V(z)** is ,

$$V(z) = \frac{(1 - z^{-N})^2}{(1 - z^{-1})^2} \cdot \frac{z^N}{1 - z^N} = \frac{1}{z^{N-2}} \cdot \frac{(z^N - 1)}{(z - 1)^2} \tag{2.75}$$

For plants with delay, modified **z**-transform should be redefined for the fast sampler. This process is as follows:

If the delay is **d** for sampling period **T**, this gives **D** pure time delay which is ;

$$D \triangleq INT\left(\frac{N.d}{T}\right) \tag{2.76}$$

So new delay for this system is ;

$$d' = d - D \cdot \frac{T}{N} \tag{2.77}$$

and new modification coefficient is ;

$$m' \triangleq 1 - \frac{d'}{T/N} \tag{2.78}$$

Following picture shows these variables on a delayed step response.

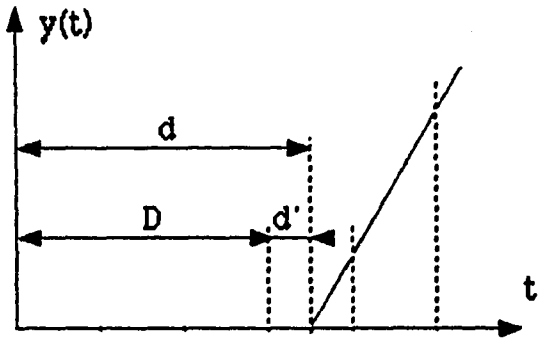


Fig. 2.8 d , d' , D are shown on very beginning of a step response figure.

The modified z-transform of fast sampling transfer function changes also ;

for EH device :

$$Y'(z) = R(z^N)\,\frac{1}{z^D}\,\frac{C(z^N)}{1 + C(z^N)\,H(z^N, m)}\,\frac{1 - z^{-N}\,e^{-sT}}{1 - z^{-1}\,e^{-s(T/N)}}\,G'(z, m') \tag{2.79}$$

for FOH device :

$$Y'(z) = R(z^N)\,\frac{1}{z^D}\,\frac{C(z^N)}{1 + C(z^N)\,H(z^N, m)}\,\frac{(1 - z^{-N})^2}{(1 - z^{-1})^2}\,G'(z, m') \tag{2.80}$$

Using above equations modified z-transform can also be handled.

Example :

Let

$$G(s) = \frac{25}{s(s+5)}$$

be the open loop transfer function. Suppose that one needs to discretize the system using ZOH adevice. Sampling frequency, T, is .3 second. However to see whether there are hidden oscilations, it is desired to see the step response at every .1 seconds, for instance. First one needs to discretize the system. Let us find both discretized models, both for T=.1 and T=.3 seconds.

$$\begin{aligned} G(z) &= (1-z^{-1})\ \mathcal{Z}\left\{\frac{a^2}{s^2(s+a)}\right\} \\ &= (1-z^{-1})\left[\frac{aTz}{(z-1)^2} - \frac{1-e^{-aT}}{(z-1)(z-e^{-aT})}\right] \\ &= (z-1)\left[\frac{aT(z-e^{-aT}) - (1-e^{-aT})(z-1)}{(z-1)^2(z-e^{-aT})}\right] \\ &= \frac{(aT-1+e^{-aT})z + 1 - e^{-aT}(1+aT)}{(z-1)(z-e^{-aT})} \end{aligned}$$

For T=.3 seconds the discretized transfer function is as follows :

$$\begin{aligned} G(z) &= \frac{(.72313)z + (.44217)}{z^2 + (-1.22313)z + (.22313)} \\ &= (.72313)\ \frac{z + (.61147)}{z^2 + (-1.22313)z + (.22313)} \end{aligned}$$

For T=.1 seconds

$$\begin{aligned} G'(z) &= \frac{(.10653)z + (.0902)}{z^2 + (-1.60653)z + (.60653)} \\ &= (.10653)\frac{z + (.84674)}{z^2 + (-1.60653)z + (.60653)} \end{aligned}$$

Using **G(z)** and applying step input to the system, step response can be calculated by long division

$$Y(z) = (.72313)\frac{z+(.61147)}{z^2+(-.5)z+(.6653)}\frac{z}{z-1}$$
$$= 0z^0 + (.72313)z^{-1} + (1.52686)z^{-2} + (1.44764)z^{-3} + \ldots$$

If one wants to see the step response at every .1 seconds, then using the equation which is given in the previous chapter can be done as follows:

$$Y'(z) = \frac{(.10653)[z+(.84674)]}{z^2+(1.60653)z+(.60653)}\;\frac{z^6+(-1.22313)z^3+(.22313)}{z^6+(-.5)z^3+(.6653)}\;\frac{z}{z-1}$$

$$= 0\,z^0 + (.10653)\,z^{-1} + (.36787)\,z^{-2} + (.72313)z^{-3} +$$
$$+ (1.0583)z^{-4} + (1.31065)z^{-5} + (1.52686)z^{-6} +$$
$$+ (1.6235)z^{-7} + (1.31065)z^{-8} + (1.52686)\,z^{-9} + \ldots$$

The reader is invited to compare results. Now, let us try the same example with modified **z**-transform.

$$G(z,m) = (1-z^{-1})\;\mathcal{Z}_m\left\{\frac{a^2}{s^2(s+a)}\right\}$$
$$= (1-z^{-1})\left[\frac{aT}{(z-1)^2} + \frac{maT-1}{(z-1)} + \frac{e^{-maT}}{z-e^{-aT}}\right]$$
$$= (1-z^{-1})\left[\frac{aT+(maT-1)(z-1)}{(z-1)^2} + \frac{e^{-maT}}{z-e^{-aT}}\right]$$
$$= (1-z^{-1})\left[\frac{(maT-1)z+(aT-maT+1)}{(z-1)^2} + \frac{e^{-maT}}{z-e^{-aT}}\right]$$
$$= \frac{1}{z}\;\frac{[(maT-1)z+(aT-maT+1)](z-e^{-aT}) + e^{-maT}(z-1)^2}{(z-1)(z-e^{-aT})}$$

$$G(z,m) = \frac{(maT - 1 + e^{-maT})z^2 + [-e^{-aT}(maT - 1) + aT(1-m) + 1 - 2e^{-aT}]z + e^{-aT}[aT(m-1) - 1]}{z(z-1)(z-e^{-aT})}$$

If m=.2, then the delay in the system is,

$$d = T(1-m) = .24 \text{ sec.}$$

For T=.1 sec. one needs to define new m, d and D. Using the definitions of these variables from the previous chapter ,

$$D = 2 \ , \ d = .04 \text{ sec.}$$

That means that there are two pure time delays. Again using these information new modification coefficient is found. Denote new modification coefficient as m'.

$$m' = 1 - \frac{.04}{.1} = .6$$

Then ,

$$G(z,.2) = \frac{(.0408)z^2 + (.8745)z + (.2499)}{z[z^2 + (-1.2231)z + (.2231)]}$$

and the step response of this system,

$$Y(z) = \frac{(.0408)z^2 + (.8745)z + (.2499)}{z^3 + (-1.1823)z^2 + (1.0976)z + (.2499)} \frac{z}{z-1}$$
$$= 0z^0 + (.0408)z^{-1} + (.9636)z^{-2} + (2.2598)z^{-3} + \ \ldots$$

The same way as it was done before, but this time for T=.1 seconds.

$$G'(z,.6) = \frac{(.0408)z^2 + (.1429)z + (.0129)}{z[z^2 + (-1.6065)z + (.6063)]}$$

and the step response ,

$$Y(z) = \frac{z}{z-1}\,\frac{1}{z^2}\,G(z,.6)\,\frac{z^3[z^6-(1.2231)z^3+(.2231)]}{z^9+(-1.1823)z^6+(1.0976)z^3+(.2499)}$$

$$= 0\,z^0 + 0\,z^{-1} + 0\,z^{-2} + (.0408)z^{-3} +$$
$$+(.2493)\,z^{-4} + (.5725)z^{-5} + (.9636)z^{-6} +$$
$$+(1.39\ )\,z^{-7} + (1.8374)z^{-8} + (2.2598)z^{-9} + \ \ldots$$

2.4.. Roots of a Polynomial

It is obvious that for control problems, one definetly needs a polynomial root solving program. The standard Bartistow Method was tried first [9]. In Bartistow Method, the initial conditions are very important. If not chosen properly, the algorithm may not converge or give accurate results. In the standard Bartistow Method, initial conditions are well defined and it gives fairly accurate result, but the accuracy is not good in case of multiple roots (in the computer implementation, of course). As an example, for $(s+1)^4$,

$$(s+1)^4 = s^4 + 4s^3 + 6s^2 + 4s + 1$$

solving this by standard Barlstow Method on a regular IBM PC, one gets 4 complex roots.

Another method is Lagguerre's iteration method [10] which is as follows.

For given polynomial $P(z)$,

$$P(z) = a_0 z^n + a_1 z^{n-1} + \ldots + a_n \qquad \ldots \qquad a_0 \neq 0$$

$$z_{k+1} = z_k - \frac{n\,P(z_k)}{P'(z_k) \pm \sqrt{H(z_k)}}$$

where,

$$H(z) = (n-1)\left[(n-1)(P'(z))^2 - n\,P(z)\,P''(z)\right]$$

and n is the degree of the polynomial. $P'(z)$ and $P''(z)$ are first and second derivative of $P(z)$, respectively. Finally z_k is k^{th} iteration of the root. So, for $z_0=0$, z_1 is ,

$$z_1 = -\frac{n a_n}{a_{n-1} \pm \sqrt{H(z_0)}}$$

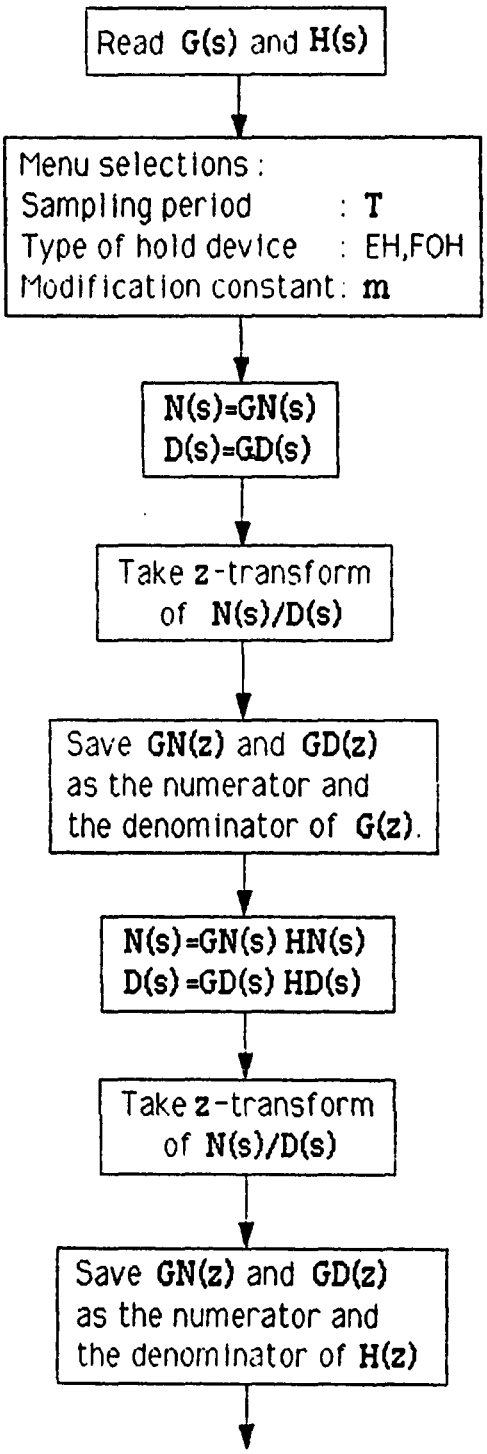
$$H(z_0) = (n-1)^2 a_{n-1}^2 - 2n(n-1) a_n a_{n-2}$$

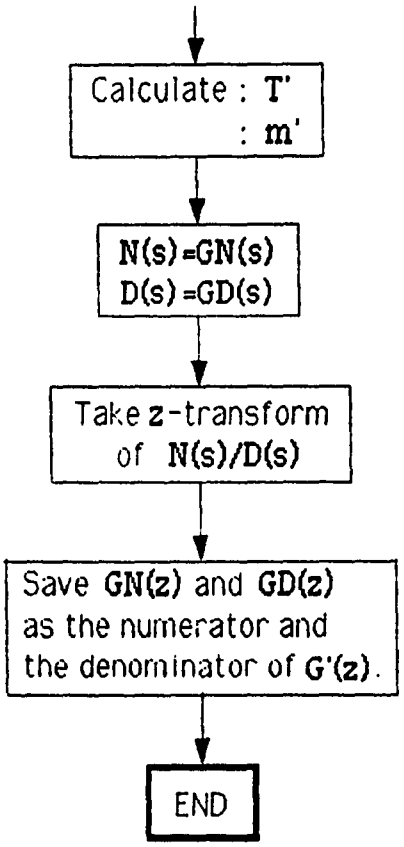
This is not easy to implement because BASIC Programming Language does not have complex number operations, but it's first iteration is very simple to implement. To compensate for weakness of Baristow Method, due to initial conditions, z_1 is tried in Baristow as the initial condition6. Using this modified Baristow Method improved the precision of program in multiple root cases.

2.5.. Computation and Flow Chart of System Discretizetion

So far, it has been shown how the z-transform is computed. However the explicit computation of $G(z)$, $H(z)$ and $G'(z)$ have not yet been shown.

Sampling period, type of hold device and modification coefficient are the same for the calculation of $G(z)$ and $H(z)$. For $G'(z)$ however, sampling period and modification coefficient have to be modified. Taking this information into account, one can find the following flow chart for the calculation of blocks.


Read G(s) and H(s)
Menu selections :
Sampling period : T
Type of hold device : EH,FOH
Modification constant: m
N(s)=GN(s)
D(s)=GD(s)
Take z-transform of N(s)/D(s)
Save GN(z) and GD(z) as the numerator and the denominator of G(z).
N(s)=GN(s) HN(s)
D(s)=GD(s) HD(s)
Take z-transform of N(s)/D(s)
Save GN(z) and GD(z) as the numerator and the denominator of H(z)



Z-transform computation is shown only for the strictly proper transfer functions. Any proper transfer fuction can be written as a summation of a constant and a strictly proper transfer function. Even though the current version of the program can not handle proper transfer functions, future versions will definetly do.

3. SYNTHESIS DESIGN METHOD

3.1. Theoretical Aspects of the Synthesis Design Method

Referring again to the discretized block diagram, assuming the use of a ZOH device,

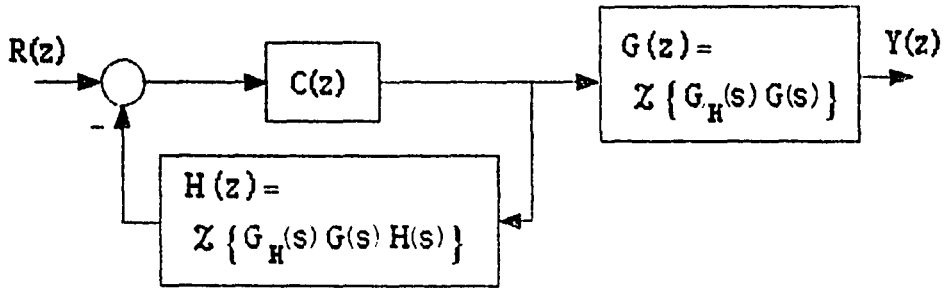


Fig 3.1 Discretized system block diagram.

where, **G(z)** and **H(z)** have been defined in Ch.2.

The closed-loop transfer function is ,

$$M(z) = \frac{Y(z)}{R(z)} = \frac{C(z)\,G(z)}{1 + C(z)\,H(z)} \tag{3.1}$$

Given a desired closed-loop transfer function **M(z)**, one may solve for the controller **C(z)** as follows;

$$C(z) = \frac{M(z)}{G(z) - M(z)\,H(z)} \tag{3.2}$$

For the sake of simplicity let **H(s)=1** Then ,

$$H(z) = G(z)$$

thus Eq.(3.1) becomes ,

$$C(z) = \frac{1}{G(z)} \cdot \frac{M(z)}{1 - M(z)} \tag{3.3}$$

The following design method is a discrete-time version of the well-known Guillemin-Truxal Synthesis method.

For the internal stability of the closed-loop system, there must be no closed-loop unstable poles and furthermore there must be no cancellation of open-loop unstable poles and zeros.
This imposes two constraints on **M(z)** :

-Unstable zeros of **G(z)** must be zeros of **M(z)**.

-Unstable poles of **G(z)** must be zeros of 1-**M(z)**.

Current version of the program handles the first requirement. The second constraint will be included in future versions of the program. So presently the synthesis design method is limited to the control design of open-loop stable plants.

Extending these ideas to systems with non-unity feedback transfer function **H(s)** is possible as follows;

$$P(z) \triangleq \frac{M(z)}{G(z)} \tag{3.4}$$

then,

$$C(z) = \frac{P(z)}{1 - P(z)\,H(z)} \tag{3.5}$$

In this case, internal stability imposes three constraints on **M(z)** :

- Unstable zeros of **G(z)** must be zeros of **M(z)**.
- Unstable poles of **H(z)** must be zeros of 1-**M(z)**.
- Unstable poles of **H(z)** (not included in **G(z)**) must be zeros of **M(z)**.

Current version of the program handles the first and last requirements only.

Next problem is how to specify **M(z)**. This will be shown for the case where **H(s)** is equal to 1.

The location of the dominant poles and zero of **M(z)** can be calculated from the transient response design specifications (i.e., overshoot, peak time, damping coefficient, etc.). The nomograms [2] given in the next page prove to be very useful for systems with two poles and one zero.

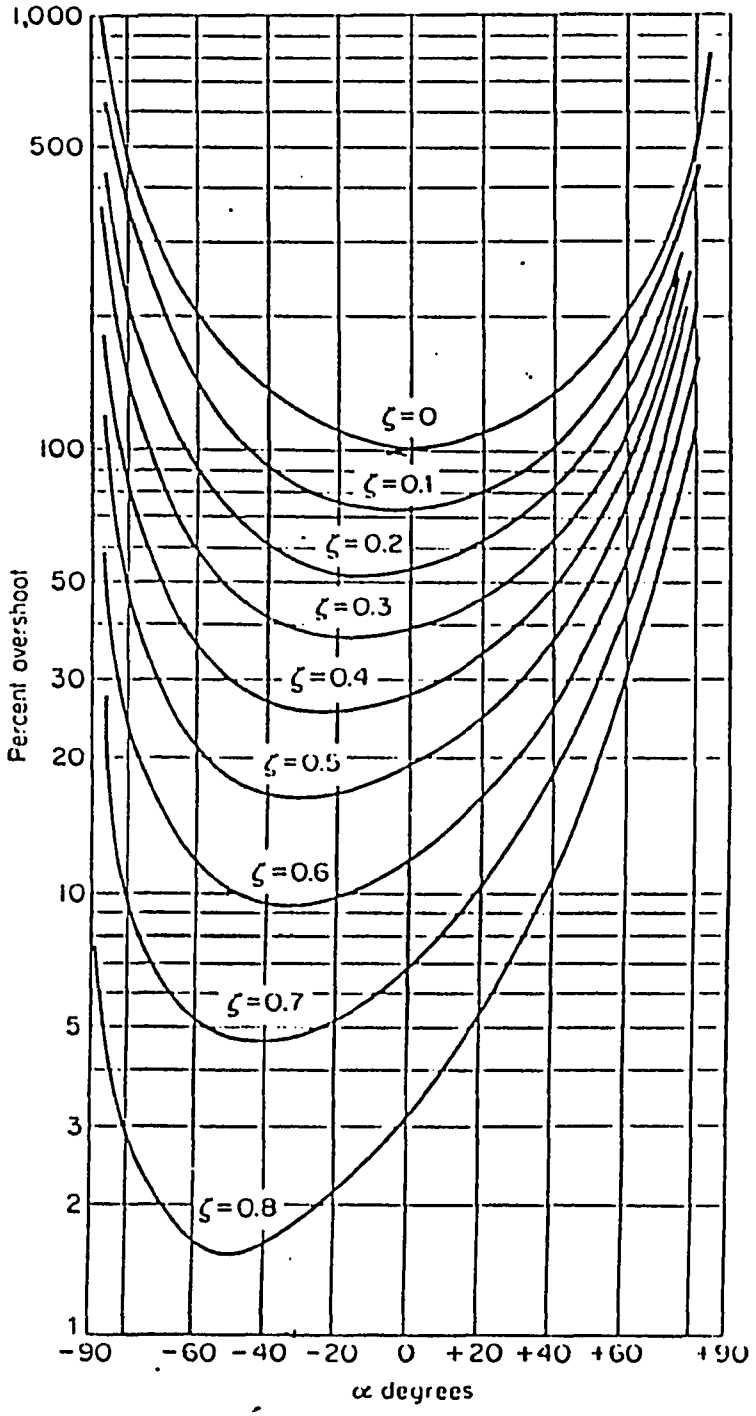


Fig 3.2.a Percent overshoot of second-order sampled-data control systems

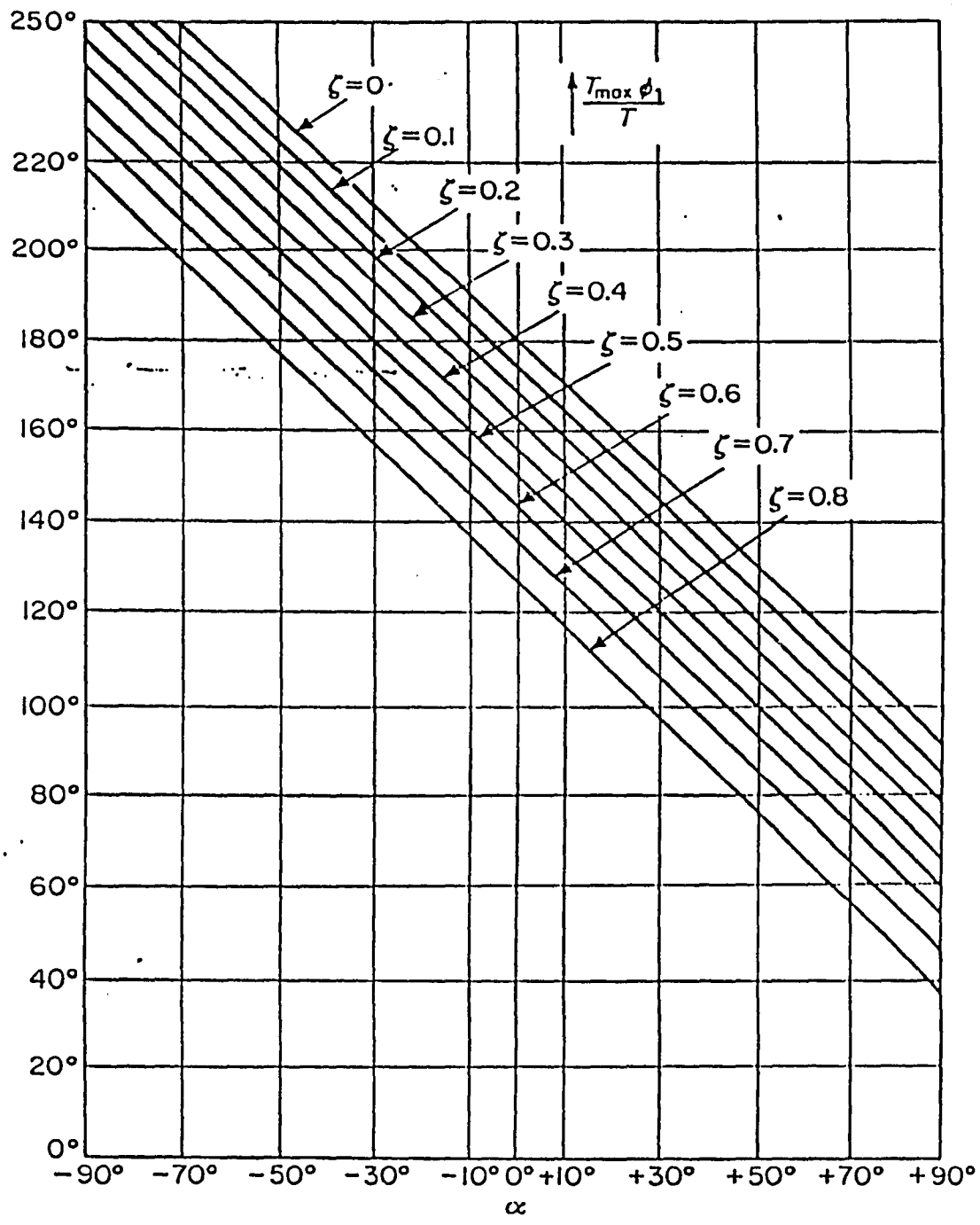


Fig 3.2.b Peak time of step response of second-order sampled-data systems versus α.

If the location of the zero is already determined, using the previous information, dominant poles can also be located It should not be forgotten that such approach assumes that **M(z)** has only one zero and two poles or that the second order approximation of **M(z)** is valid. On the other hand, if **M(z)** does not have only one zero, but also has all the RHP zeros of **G(z)** and some or all of the LHP zeros then the above determination of dominant poles should be viewed as the first iteration towards completing the design.

For realizability of **C(z)** , **M(z)** and **G(z)** have to have same relative degree . Therefore , in some cases sufficiently many negligible poles (i.e.; poles at or near **z=0**) need to be included in **M(z)**.

Once one gets the poles and zeros configuration of **M(z)**, by using the zero steady-state error condition, the gain of **M(z)** can be calculated. This implies that for zero steady state error the following condition should hold;

$$M(z)\Big|_{z=1} = 1 \tag{3.6}$$

Ramp error constant K_v ,as expressed through **M(z)**, is

$$\frac{1}{TK_v} = -\frac{d(M(z))}{dz}\Big|_{z=1} \tag{3.7}$$

Since **M(1)=1**, then

$$\frac{1}{TK_v} = -\frac{d(M(z))}{dz}\Big|_{z=1} = -\left\{\frac{d}{dz}(\ln(M(z))\right\}\Big|_{z=1} \tag{3.8}$$

$$= -\left\{\frac{d}{dz}[\ln(K) + \ln(MN(z)) - \ln(MD(z))]\right\}\Big|_{z=1} \tag{3.9}$$

$$= -\left\{\frac{MN'(z)}{MN(z)} - \frac{MD'(z)}{MD(z)}\right\}\Big|_{z=1} \tag{3.10}$$

$$= -\left\{\sum_{i=1}^{m}\frac{1}{1-z_i} - \sum_{i=1}^{n}\frac{1}{1-p_i}\right\} \tag{3.11}$$

Since all the closed-loop poles and zeros are known, one can calculate K_v, simply by using one of the above equations.

If the desired K_v value is not achieved then by adding a dipole near $z=1$ new K_v value can be calculated by using the following formula:

$$\frac{1}{TK_{v,NEW}} = \frac{1}{TK_{v,OLD}} + \frac{1}{1-p_i} - \frac{1}{1-z_i} \qquad (3.12)$$

where $K_{v,OLD}$ is defined as K_v in Eq.(3.7). For a given pole location, p_i, the zero location can be calculated. One should remember that dipole gives high overshoot as it is moved away from $z=1$ [2], and as it is moved towards $z=1$ it creates implementation problems. One has to decide about the trade off. By proper dipole, K_v can be improved without affecting the step response at all. But an important issue is that for the type 2 systems, K_v becomes a condition to be satisfied. For type 2 systems K_v is infinity. Using dipole enables the program to handle this type of systems. 3 (or higher) type of systems are not handled in this version of the program.

Example :

$$G(z) = H(z) = (.5) \cdot \frac{z+1}{(z-1)^2}$$

Design a controller which meets the following design specifications:

- no more than 10% overshoot,
- no more than $3T$ peak time,
- damping coefficient, $\zeta \approx .6$
- $K_v = \infty$ (zero steady state error to ramp input).

From Fig.3.2.a and overshoot requirement $\Rightarrow \alpha = -30° = -\pi/6$ rad.

From Fig.3.2.b and peak time requirement $\Rightarrow \Phi_1 = 70° = 7\pi/18$ rad.

So ,

$$|p_1| = \exp\left\{-\Phi_1 \tan(\sin^{-1}(\zeta))\right\} = .4$$

Location of dominant poles and zero are ;

$$p_{1,2} = |p_1| \cdot e^{-j\phi_1} = (.137) \pm j(.376)$$

$$z_z = .371$$

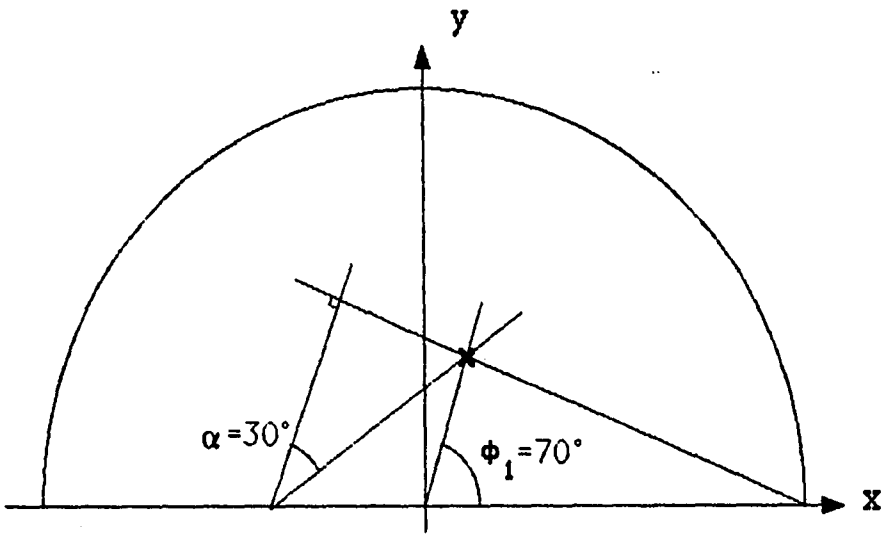


Fig. 3.3 Location of dominant poles and zero.

Therefore the optimum solution is

$$M'(z) = (.646)\ \frac{z + (.371)}{z^2 - (.274)\,z + (.16)}$$

One can see that the above closed-loop transfer function does not satisfy K_v requirement. As explained before, after fixing the closed-loop transfer function, K_v can be improved without affecting the overall step response. Now one has to analyze the various options, since $M'(z)$ and $G(z)$ are known.

- Cancel the system zero and add a new one (at z_z=-.371),
- Using the system zero to find the location of poles (z_z=-1),
- Use both zeros but calculate the location of poles with respect to added zero.

Solving this problem by hand for three different options will take quite a while, unless using a package like the one developed in this thesis. Let us solve this problem by using the first option. Solutions along the other options are equally easy.

First one has to satisfy the ramp constant requirement, $K_v=\infty$. For $M'(z)$:

$$\frac{1}{TK_v} = \frac{2-(.274)}{1-(.274)+(.16)} - \frac{1}{1+(.371)} = 1.22 \text{ sec}^{-1}$$

Since $K_v \to \infty$, one has to add a dipole without affecting the step respose at all. So using the Eq.(3.12),

$$\frac{1}{K_v} = 0 = 1.22 + \frac{1}{1-p_i} + \frac{1}{1-z_i}$$

$$\text{for } p_i = .97 \Rightarrow z_i \cong .971$$

Thus the closed-loop transfer function becomes,

$$M(z) = M'(z) \cdot \frac{1-p_i}{1-z_i} \cdot \frac{z-p_i}{z-z_i}$$

$$M(z) = (.67)\frac{(z+.371)(z-.971)}{[z^2-(.274)z+(.16)](z-.97)}$$

Next one has to check the step response, to see whether the design specifications are met or not. If they do then solve for the controller. Step response is as follows:

$$\begin{aligned} Y(z) &= M(z)\,R(z) \\ &= (.67)\frac{(z+.371)(z-.971)}{[z^2-(.274)z+(.16)](z-.97)}\,\frac{z}{z-1} \\ &= 0+(.67)z^{-1}+(1.1)z^{-2}+(1.11)z^{-3}+(1.04)z^{-4}+\ldots \end{aligned}$$

The results are:

- 11% over shoot,
- 3 T peak time,
- $\zeta \approx .6$
- $K_v=\infty$

Finally solving for the controller, C(z) :

$$C(z) = \frac{1}{(.05)} \frac{(z-1)^2}{z+1} (.67) \frac{[z+(.371)][z-(.971)]}{(z-1)^2 (z+.086)}$$

$$C(z) = 13.34 \frac{z-(.371)}{z+1} \frac{z-(.971)}{z+(.086)}$$

Design is done. It is suggested to solve the same example using the program after finishing reading the user manual.

3.2.. Computer Implementation of the Synthesis Design Method

A step by step implementation of the synthesis design procedure is given next.

First, one has to compute the **z**-transforms of the system blocks (as discussed in Ch.2).

$G(z) = \mathcal{Z} \{ G_H(s) G(s) \}$, sampling period at **T**

$H(z) = \mathcal{Z} \{ G_H(s) G(s) H(s) \}$, sampling period at **T**

$G'(z) = \mathcal{Z} \{ G_H(s) G(s) \}$, sampling period at **T/N**

As it has been discussed in Ch.2, **G(z)** zeros and **z**-transform of **H(s)** poles give us the closed-loop system zeros. Next, one needs to extract the location of the system's zeros and rename them in two group as LHP and RHP zeros. Since the RHP zeros can not be cancelled, these are included in **M(z)** automatically. Besides, one should be able to add any of the LHP zeros in **M(z)**, as well (if necessary).

One has to choose one of the following design options :

- In order to use this option, system has to have at least one stable zero. If there are more than one; then, one of these has to be specified as the dominant zero. Then the program calculates the optimum (2) poles and zero location by using the damping coefficient and peak time specifications. For the on line calculation of control measures (i.e. overshoot, damping coefficient ,etc.), the specified zero is used.

- Second option is to be used if there are no stable zeros. Therefore the control measures can not be calculated.This option is same as the first one, but since no stable zero exists it can not locate the poles and zero.

- Last option allows one to add a zero in **M(z)**. Location of the poles and zero are calculated in the same way as is done in the first option. The only difference is that, this option uses the added zero for the calculation of the control measures.

Next, the calculation of control measures, geometric quantities, and the location of the closed-loop poles emphasizing the interactive nature of the program is highlighted.

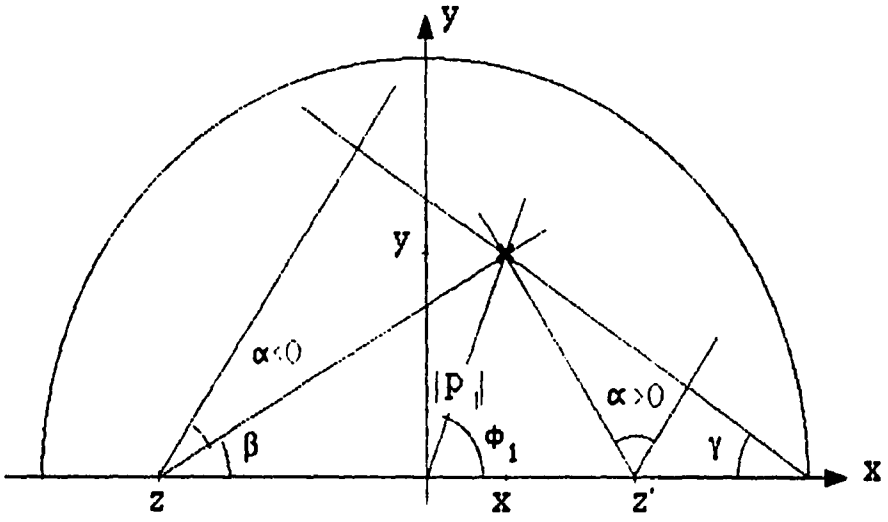


Fig. 3.4 Unity circle with all the quantities and locations on it.

In the figure above ,

$$\Phi_1 = \tan^{-1}\left(\frac{y}{x}\right) \quad , \quad \gamma = \tan^{-1}\left(\frac{y}{1-x}\right) \quad , \quad \beta = \tan^{-1}\left(\frac{y}{x-z}\right) \tag{3.13}$$

$$\alpha + \beta + \gamma = \frac{\pi}{2} \quad \Rightarrow \quad \alpha = \frac{\pi}{2} - \beta - \gamma \tag{3.14}$$

$$|p_1| = \exp\left\{-\Phi_1 \tan\left(\sin^{-1}(\zeta)\right)\right\} \tag{3.15}$$

$$\text{overshoot} = \frac{\sqrt{1-\zeta^2}}{\cos(\alpha)} \exp\left\{-\tan\left(\sin^{-1}(\zeta)\right)\left[\sin^{-1}(\zeta) - \alpha + \pi\right]\right\} \tag{3.16}$$

$$\frac{T_{max}}{T} = \frac{1}{\Phi_1} \cdot \left(\sin^{-1}(\zeta) - \alpha + \pi\right) \tag{3.17}$$

Knowing the location of the two dominant poles and one dominant zero, it is straightforward to calculate the above variables. For the user guidance, the same equations are used.

If damping coefficient, ζ, is given, by trying different values of α in Eq.(3.16), one can find the α value that minimizes the overshoot. Since the peak time, T_{max}/T, is given, Φ_1 can be calculated from Eq.(3.17). One can find the location of the dominant poles or zero by using Eq.(3.15) and Eq.(3.14), respectively.

At this stage, all the unstable closed-loop zeros are included in **M(z)** together with some of the stable ones, and the second order approximation of the closed-loop system has already been done One can relocate the two dominant poles and added zero almost everywhere in the unit circle. The two dominant poles have to be complex and the zero has to be on the real axis. When one is satisfied with the pole/zero picture, one can proceed towards viewing the closed-loop step response. First order system design will be included in the future versions.

Next one has to make sure that **C(z)** is realizable. As it is mentioned before, this requires **M(z)** and **G(z)** to have the same relative degree. That means that one needs to add sufficiently many negligible poles in **M(z)**.

Then by using Eq.(3.6) and Eq.(3.11), gain of the **M(z)** and K_v are calculated respectively. At this stage one has to add a dipole due to K_v improvment or requirement, as discussed earlier in this chapter.

Since **M(z)** is known, one can solve for **P(z)** and finally for **C(z)**. From the definition of **P(z)** and **C(z)**;

$$P(z) \triangleq \frac{M(z)}{G(z)} \tag{3.18}$$

$$C(z) = \frac{P(z)}{1 - P(z)\,H(z)} \tag{3.19}$$

Since **M(z)**, **P(z)**, **G(z)**, and **H(z)** are all rational functions, calculation of **C(z)** requires three rational function multiplications :

- $M(z)\,[G(z)]^{-1}$
- $P(z)\,H(z)$
- $P(z)\,[1 - P(z)\,H(z)]^{-1}$

One needs a subroutine for rational functions multiplication.

Let **A** and **B** be two rational functions. **AN**(.) and **BN**(.) denote the zeros of **A** and **B**, respectively. **AD**(.) and **BD**(.) denote the poles of **A** and **B**, respectively. Therefore ,

$$A(.)\,B(.) = AG\,\frac{AN(.)}{AD(.)}\,BG\,\frac{BN(.)}{BD(.)} = AG\,BG\,\frac{AN(.)}{AD(.)}\,\frac{BN(.)}{BD(.)}$$

where **AG** and **BG** denote the coefficients of the highest numerator degree of **A** and **B**, respectively (note that they are 1, now). Comparing the roots of those four polynomials pole zero cancellation is done easily. Suppose that rational function **D(z)** is the common factor of **A** and **B** Hence ,

$$\begin{aligned} A(.)\,B(.) &= A'(.)\,\frac{D(.)}{D(.)}\,B'(z) = A'(z)\,B'(z) \\ &= AG\,BG\,\frac{A'N(.)}{A'D(.)}\,\frac{B'N(.)}{B'D(.)} \\ &= G\,\frac{N(.)}{D(.)} \end{aligned}$$

where :

- **G** is **GA.GB**
- **N(.)** is product of **A'N(.)** and **B'N(.)**
- **D(.)** is product of **A'D(.)** and **B'D(.)**

In this subroutine two rational functions are compared and after that the necessary pole/zero cancellations are done. The most important problem here is how close two points should be, in order to be considered to be in the same exact location.

Last thing before viewing the step response is the calculation of fast sampling transfer function. One can find the step response with in between samples , if one has the fast sampled transfer function. This is also shown in Ch.2.3 .

Checking the closed-loop step response is important, because the calculation of the response parameters (from nomograms) was based on the single dominant zero assumption. If more than one dominant zero exists, one should not rely on these calculations, and should look at the actual step response, instead.

A design procedure is based on trying different options or configurations for getting the best possible result. One needs to compare different solutions. The best way of doing it is to look at the different solutions simultaneously. Meanwhile one needs to be able to access and manipulate the new solutions. When all these options are considered, one would be led to the kind of screen setting that is developed in this software.

3.3.. Pseudocode for The Synthesis Design Method

Step 1 Read **G(z)**, **H(z)** and **G'(z)** from the disk.

Step 2 Find the system zeros and classify into stable and unstable zeros.

Step 3 Include all of the unstable zeros in **M(z)**.

Step 4 Ask user to include stable system zeros.

Step 5 Ask user to choose one of the following design methods:

- Using damping coefficient and peak time specifications find the optimum dominant pole and zero locations and replace the zero location with one of the stable zeros.
- Let user initialize the design.
- Same as the first one but instead of replacing the optimum zero, include it in **M(z)**.

Step 6 If second option is chosen go to the next step. Let user specify damping coefficient and relative peak time. Also ask user if $\mathbf{K_v}$ improvment is desired. If the answer is yes then let user enter desired $\mathbf{K_v}$ value

Step 7 Draw unit circle and step responses.

Step 8 If the second option is chosen then locate cursor to (0,0.5 j). If not, then locate cursor to the dominant pole location and show the stable system zeros with the added zero if any.

Step 9 Let user change dominant pole and zero location. Zero location can not be changed unless the zero is an added one. If the second option is chosen then go to the Step 11.

Step 10 Calculate α angle, Φ angle, damping coefficient ζ, relative peak time, overshoot values as explained previously and display these values.

Step 11 Display the cursor location and the dominant zero location.

Step 12 Move cursor or the added zero

Step 13 If the step response is desired then go to next step. If not then go back to the step 9.

Step 14 For the realizability of $\mathbf{C(z)}$, let user specify the location of the negligible poles.At this step $\mathbf{M(z)}$ is completely specified.

Step 15 If $\mathbf{K_v}$ improvment is desired and the calculated $\mathbf{K_v}$ value for the system is smaller than the desired value, then go to the next step.If not, call the dipole adding subroutine. This subroutine is called automatically if the system is type two. By using desired and actual $\mathbf{K_v}$ values and pole location, calculate the zero location of the dipole. User can change the zero location by changing the pole location of the dipole.

Step 16 Calculate $\mathbf{P(z)}$.

Step 17 Calculate $\mathbf{1-P(z).H(z)}$.

Step 18 Calculate $\mathbf{C(z)}$.

Step 19 Calculate $\mathbf{V(z)}$.

Step 20 Using $\mathbf{G'(z)}$ calculate $\mathbf{Y'(z)}$ and display it.

Step 21 If desired save this step response and go to Step 9.

4. ROOT LOCUS DESIGN METHOD

4.1.. Theoretical Aspects of the Root Locus Design Method

Referring to the discretized block diagram (Fig. 3.1), the closed-loop transfer function is;

$$M(z) = \frac{Y(z)}{R(z)} = \frac{C(z)\,G(z)}{1 + C(z)\,H(z)} \tag{4.1}$$

Thus the closed-loop characteristic equation for this system is ,

$$1 + C(z)\,H(z) = 0 \tag{4.2}$$

The Root Locus method provides for the geometric location of the closed-loop poles depending on a single design parameter.

The purpose is to find a $C(z)$, such that roots of the characteristic equation will be stable and closed-loop step response will meet the design specifications.

A step by step root locus design method is as follows.

Step1: Find the roots of the characteristic equation ($1 + C(z)\,H(z)$) for different values of **K**. This first try attempts to use a proportional control strategy, $C(z) = K$.

Step2: If the closed-loop system is unstable for all values of **K**, or the root locus plot is such that the control design requirements are not going to be met, then try a higher order controller. In other words, the open loop transfer function is to be modified, and a root locus plot is redrawn.

Step3: For a "satisfactory looking" root locus plot, one needs to move along the root locus branch (as explained in step 4).

Step4: One needs to pick up a convenient location for the closed-loop dominant poles and to see the closed-loop step response at the selected root locus points to test if the transient behaviour meets the requirements. If it does not then Step 2 is revisited.

As was discussed in the previous chapter unstable poles and zeros pose constraints on the synthesis of **M(z)**. Root locus design method does not have any limitations other than realizability of **C(z)** (i.e. the number of **C(z)** zeros should not exceed the number of **C(z)** poles). Root locus does not have the advantage that the synthesis design method have in imposing a second-order closed-loop behaviour, but on the other hand Root Locus design generally generates controllers of lower order than the ones generated by the synthesis method.

4.2.. Software Development and Implementation

The following example illustrates the Root Locus design steps.

Example :

Consider a system which has the following transfer functions.

$$G(z) = H(z) = (.6387)\,\frac{z + 1.2589}{(z-1)(z-2)}$$

Characteristic equation for this system is,

$$(z-1)(z-2) + (.6387)[z + 1.2589] = 0$$

By trying a proportional control the characteristic equation is modified as follows,

$$(z-1)(z-2) + K(.6389)[z + 1.2589] = 0$$

Since the Root Locus branches (Fig. 4.1) are outside the unity circle, one needs to try higher order controllers. As a first trial add one pole and zero at **z**=-.8 and **z**=.9, respectively. So, the characteristic equation becomes;

$$1 + KC(z)H(z) = 0$$

where,

$$C(z) = \frac{z - (.9)}{z + (.8)}$$

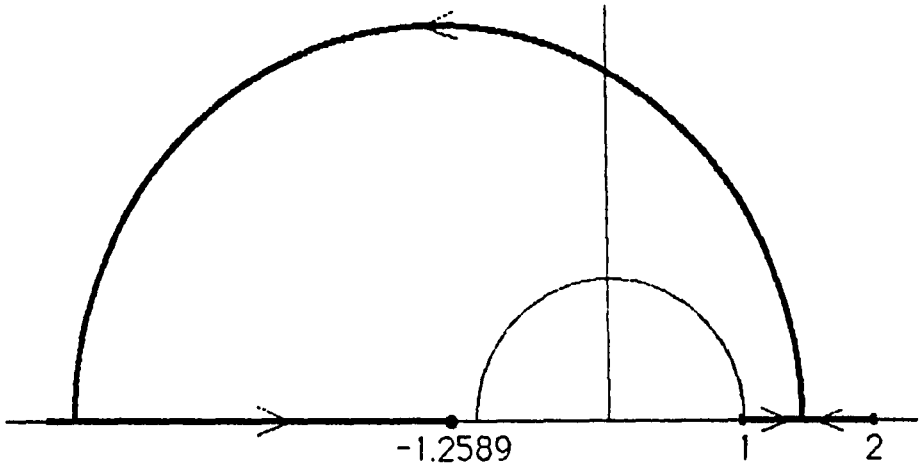


Fig. 4.1 Root Locus of the system in the example.

rewriting characteristic equation after substituting the **C(z)**;

$$(z-1)(z-2)[z+(.8)]+K(z+1.2589)[z-(.9)]=0$$

and the resulting Root Locus plot is :

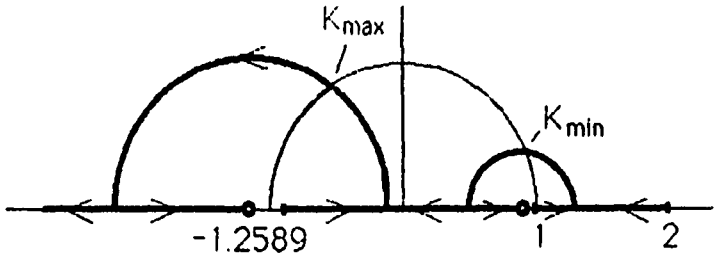


Fig. 4.2 Root Locus of the system in the example with the **C(z)**.

As it can be seen by Fig.4.2, if **K** is in between K_{min} and K_{max}, all root locus branches will be inside the unit circle. So one needs to look at the step responses for different points of that part of the root locus. If the requirements are met then the design is complete. If not then different **C(z)** needs to be tried. More generally :

First **G(z)** and **H(z)** need to be calculated The closed-loop

characteristic equation is formed. To draw the root locus one solves for the polynomial roots for many given values of **K**. The user decides on the desired range of values for **K** and it is done interactively

As it is explained in Ch.2, roots of a 4th or higher order polynomials are calculated by Baristow Method. This method is based on minimizing a certain error. In some cases this error can not be reached and the program never truncates. To avoid this situation, the number of iterations is limited. If number of iterations passes this limit then the approximation is accepted and the program goes to the next step. One can interrupt the program to specify a new value, or stop the calculation if a satisfactory looking root locus is drawn.

At this stage the root locus is drawn and shown together with the unity circle and a user specified closed-loop damping coefficient curve. One occasionally may need to find the location of break away points, or intersection points of root locus with the unit circle. This may necessitate rescaling of **x** and **y** axes. Such rescaling of the **x** and **y** may cause the unity circle to lose its circular shape.

Root Locus Design involves manipulations of open-loop poles and zeros (i.e. shifting, adding, deleting) One needs to be able to add a pole or a zero of the open-loop anywhere inside or outside the unity circle, in order to pull root locus branches into the unity circle. If pole/zero cancellations are desired (inside the unity circle , of course) a cursor should be moved to such locations. To modify the controller during the design, it is also necessary to move the cursor to any pole and zero location of **C(z)**. In other words, what is needed is an easy and accurate access to the poles and zeros of both the system and the controller.

C(z) has to be proper or strictly proper to be realizable. The program was designed to monitor that **C(z)** will never become improper. After finding a controller, **C(z)**, that gives a "nice-looking" root locus, one needs to move along the root locus branches to finalize the selection of the controller gain When selecting a point on the root locus to be a closed-loop dominant pole, it is desired to see its coordinates, and the coordinates of the points on the other branches (for the same gain value). This will be assisted by viewing

the closed-loop step response.

The calculation of the closed-loop step response involves three rational function multiplications, which are

- $C(z)\,H(z)$
- $(1 + C(z)\,H(z))^{-1}\,C(z)$
- $(1 + C(z)\,H(z))^{-1}\,C(z)\,G(z)$

Such calculations were the topic of Ch. 3.

By calculating the closed-loop transfer function $M(z)$, the closed-loop unity step response $Y(z)$ is found through long division;

$$Y(z) = \frac{z}{z-1}\,M(z)$$

where;

$$M(z) = \frac{C(z)\,G(z)}{1 + C(z)\,H(z)}$$

4.3.. Pseudocode of Root Locus Method Program

Step 1 Read $G(z)$, $H(z)$ and $G'(z)$ from the disk. Read also lower and upper bounds of K, Φ, x, y together with increments (Δ_i; i=K, Φ, x, y).

Step 2 Specify damping coefficient and the minimum error for the calculation of polynomial roots.

Step 3 Calculate $V(z)$ (defined in Ch. 2.3).

Step 4 Specify the damping coefficient and the minimum error (used for the calculation of the roots).

Step 5 Draw frame of the z-plane.

Step 6 Calculate $C(z)$. $H(z)$ and cancel the common factors.

Step 7 Calculate denominator of $(C(z)\,H(z))$, $D(z)$,and the numerator of $(C(z)H(z))$, $N(z)$.

Step 8 Calculate roots of the characteristic equation $D(z)+K\,N(z)$ starting from K_{min} to K_{max} and put them in array ROOT.

Step 9 Locate the cursor at (x_{min},0). Draw the cursor pointers.

Step 10 Draw the root locus using the array ROOT.

Step 11 Show the scaling of one of the followings: **K** , **Φ** , **x** , **y**.

Step 12 Show the cursor location.

Step 13 Stand by for one of the following user commands and come back to this step again, unless otherwise specified.

- Step 13.1 Move the cursor up, down, to the left and right together with the pointers.
- Step 13.2 Change the minimum cursor movement.
- Step 13.3 Scaling of **K** , **Φ** , **x** , **y** : rescale and display desired one or the one rescaled most recently.
- Step 13.4 Display the pole location of **C(z)** that is closest to the cursor location. If desired move cursor to that location.
- Step 13.5 Display the zero location of **C(z)** that is closest to the cursor location. If desired move cursor to that location.
- Step 13.6 Add a pole in **C(z)** and display the zeros and the poles of **C(z)**.
- Step 13.7 Add a zero in **C(z)**, if it has relative degree larger than 1 and display the zeros and the poles of **C(z)**.
- Step 13.8 Draw the root locus with respect to new **C(z)**.
- Step 13.9 Clean up the screen and draw the root locus with the most recently calculated poles.
- Step 13.10 Start all over again (go to Step 4).
- Step 13.11 Hide or display the zeros and the poles of **C(z)**.
- Step 13.12 Move the cursor to the nearest decimal unit (i.e. -0.1, 0.8, 2.1, etc.).
- Step 13.13 Go to the next screen mode to move along the root locus.
  - Step 13.13.1 Display one of the followings **K** , **Φ** , **x** , **y**.
  - Step 13.13.2 Display **K** , closed-loop poles and highlight them on the root locus.
  - Step 13.13.3 Go back to the previous screen mode (Step 10).
  - Step 13.13.4 Start calculating the step response for the desired closed-loop pattern.

Step 13.13.4.1 To open space in the memory save array ROOT on disk and erase that array open new arrays for the calculation of the **C(z)**.

Step 13.13.4.2 Calculate **M(z)** and the step response.

Step 13.13.4.3 Add a dipole in **C(z)** if the $\mathbf{K_v}$ improvement is desired.

Step 13.13.4.3 Delete a dipole if it is added.

Step 13.13.4.4 Display **C(z)** and **M(z)**.

Step 13.13.4.5 Get a print out of **C(z)** and **M(z)**.

Step 13.13.4.6 Go back to the previous screen after deleting the arrays opened for the step response calculation, loading back the array ROOT and drawing the **z**-plane continue from Step 13.13.1 .

5. USER MANUAL

5.1.. How to Start-up The Program

This program is written for the IBM PC/XT/AT and their compatibles. During the development of the program, DOS 2.1 and higher versions of DOS were used.

First one needs to boot the system up with DOS. Then load GRAPHICS, GRAFTABL and finally BASICA. Then one can employ BASIC programs. In order to load the MENU, type RUN"MENU↵, where ↵ denotes ENTER or RETURN. Then the program menu will be displayed on the screen, as follows.

```
..                    PROGRAM MENU                          ..

..      E     TO ENTER A NEW PROBLEM          <S>           ..
..      A     TO ENTER A NEW PROBLEM          <Z>           ..
..      D     TO DISPLAY THE SYSTEM                         ..
..      Z     TO DISCRETIZE THE SYSTEM                      ..
..      F     TO DISPLAY THE DISCRETIZED SYSTEM             ..
..      S     FOR THE SYNTHESIS DESIGN METHOD               ..
..      R     FOR THE ROOT LOCUS DESIGN METHOD              ..

..      Q           QUIT AND EXIT                           ..

..                PRESS CAPS LOCK PLEASE                    ..
```

Press **E** to enter a new problem in s domain, **A** to enter a new problem in **z** domain; **D** to display the system (s domain); **Z** to dicretize the system; **F** to display the dicretized model; **S** to employ synthesis design method; **R** to employ root locus design method and **Q** to quit and exit from the program

5.2.. How to Enter a New Problem Model

Assuming an example let **G(s)** and **H(s)** be

$$G(s) = \frac{10}{s\,(s+5)\,(s+10)} \quad , \quad H(s) = 1$$

Entering a new problem means, that one needs to type in **G(s)** and **H(s)** or in other words to type four polynomials. Only one of the polynomials,

the denominator of G(s) will be shown here to demonstrate how data is to be entered.

First one needs to specify the degree of the polynomial. Then one needs to enter the polynomial coefficients or roots. It should be displayed in both forms. One also needs to access any old data for the corrections. After entering all four polynomials, one can get the print out of each. The following description shows how to enter the denominator of G(s) with every screen settings.

```
ABOUT  H(s) and G(s)

G(s) DENOMINATOR DEGREE  :?
G(s) NUMERATOR DEGREE    :
H(s) DENOMINATOR DEGREE  :
H(s) NUMERATOR DEGREE    :
```

Enter the degrees of the polynomials, as it is shown below.

```
ABOUT  H(s) and G(s)

G(s) DENOMINATOR DEGREE  :? 3 ←
G(s) NUMERATOR DEGREE    :? ←
H(s) DENOMINATOR DEGREE  :? ←
H(s) NUMERATOR DEGREE    :? ←
 ARE THESE CORRECT (N) ? ? ←
```

Now one can start entering the polynomials, in the given order.

```
G(s) DENOMINATOR

CHARACTERISTIC POLYNOMIAL (1) OR ROOTS (2) ?
```

Type **1** and press ← to enter the polynomial coefficients as follows:

```
G(s) DENOMINATOR

INPUT 3   TH DEGREE COEFFICIENT
      ? 1 ←
```

```
G(s) DENOMINATOR

INPUT 3    TH DEGREE COEFFICIENT
      ? 1 ←
      IS IT CORRECT (N) ? ←
```

Press **N** and ← if it is wrong and reenter the data. Highest degree coefficient can not be reentered after this stage is passed.

```
                G(s) DENOMINATOR

          INPUT  2     TH DEGREE COEFFICIENT
                 ? 15 ←
```

```
                G(s) DENOMINATOR

          INPUT  1     TH DEGREE COEFFICIENT
                 ? 50 ←
```

```
                G(s) DENOMINATOR

          INPUT  0     TH DEGREE COEFFICIENT
                 ? 0 ←
```

Then **G(s)** is displayed

```
                G(s) DENOMINATOR

                0     .s^0  ( 0 )
                50    .s^1  ( 50 )
                15    .s^2  ( 15 )
                1     .s^3  ( 1 )
                 ARE THESE CORRECT (N) ?
```

If any of the coefficients is wrong, except for the highest degree, it can be reentered. That is followed by displaying the roots of it,

```
                ROOTS OF G(s) DENOMINATOR

                s( 1 )= -5          0      .j
                s( 2 )= -10         0      .j
                s( 3 )=  0          0      .j
                .. PRESS ANY KEY TO CONTINUE ..
```

This follows by giving the other polynomials. Next it is shown how to enter the same polinomial by using its roots.

```
                G(s) DENOMINATOR

          CHARACTERISTIC POLYNOMIAL (1) OR ROOTS (2) ? 2 ←
```

Then enter the roots in any order. It is enough to enter only one of the

complex roots. Because its conjugate is added automatically by the program.

```
                G(s) DENOMINATOR

   CHARACTERISTIC POLYNOMIAL (1) OR ROOTS (2) ? 2 ←
              1     TH ROOT :
         REAL PART = ? ←
      IMAGINARY PART = ? ←
      IS IT CORRECT (N) ? ←
```

```
                G(s) DENOMINATOR

                1 TH ROOT =  0        0 J

                2     TH ROOT :
         REAL PART = -5 ←
      IMAGINARY PART = ? ←
      IS IT CORRECT (N) ? ←
```

```
                G(s) DENOMINATOR

                1 TH ROOT =  0        0 J
                2 TH ROOT =  -5       0 J

                3     TH ROOT :
         REAL PART = -10 ←
      IS IT CORRECT (N) ? ←
```

```
                G(s) DENOMINATOR

                1 TH ROOT =  0        0 J
                2 TH ROOT =  -5       0 J
                3 TH ROOT =  -10      0 J
   G(s) DENOMINATOR POLYNOMIAL
   0              s^ 0
   50             s^ 1
   15             s^ 2
                  s^ 3
              .. PRESS ANY KEY TO CONTINUE ..
```

This is the end of this phase of entering G(s) denominator.

Entering the other polynomials is done in a similar way. After entering the four polynomials, G(s) and H(s) are diplayed in the following form:

```
                    G(s)
     NUM.          DEN.          NUM. ROOT(S)          DEN. ROOT(S)
3:  1.0000E+00                *                    * -1.0000E+10  0.0000E+00
2:  1.5000E+01                *                    * -5.0000E+00  0.0000E+00
1:  5.0000E+01                *                    *  0.0000E+00  0.0000E+00
0:  1.0000E+00  0.0000E+00    *   GAIN = 1.0000E+00
                    H(s)
     NUM.          DEN.          NUM. ROOT(S)          DEN. ROOT(S)
0:  1.0000E+00  1.0000E+00    *   GAIN = 1.0000E+00
```

One can get the same picture as a printout. Pressing space bar loads the z-transform program

5.3.. Discretization of the System

```
                SAMPLING PERIOD T = ? .1 ←

          # OF POINTS IN BETWEEN EACH SAMPLE ? 1 ←

                WHAT IS THE HOLD DEVICE
                   (0:ZOH/1:FOH/2:EH)

                MODIFICATION (Y) ? ←

          T = .1      m=1          ZOH DEVICE
                    G(z)
     NUM.          DEN.          NUM. ROOT(S)          DEN. ROOT(S)
3:  1.0000E+00                *                    * 1.0000E+10  0.0000E+00
2:  1.0000E+00 -1.9744E+00    * -2.6170E+00  0.0000E+00  * 3.6788E-01  0.0000E+00
1:  2.7976E+00  1.1975E+00    * -1.8064E-01  0.0000E+00  * 6.0653E-01  0.0000E+00
0:  4.7274E-01 -2.2313E-01    *   GAIN = 1.1649E-04
                    H(z)
     NUM.          DEN.          NUM. ROOT(S)          DEN. ROOT(S)
3:  1.0000E+00                *                    * 1.0000E+10  0.0000E+00
2:  1.0000E+00 -1.9744E+00    * -2.6170E+00  0.0000E+00  * 3.6788E-01  0.0000E+00
1:  2.7976E+00  1.1975E+00    * -1.8064E-01  0.0000E+00  * 6.0653E-01  0.0000E+00
0:  4.7274E-01 -2.2313E-01    *   GAIN = 1.1649E-04
                    G'(z)
     NUM.          DEN.          NUM. ROOT(S)          DEN. ROOT(S)
3:  1.0000E+00                *                    * 1.0000E+10  0.0000E+00
2:  1.0000E+00 -2.3853E+00    * -3.1091E+00  0.0000E+00  * 7.7879E-01  0.0000E+00
1:  3.3302E+00  1.8577E+00    * -2.2108E-01  0.0000E+00  * 6.0653E-01  0.0000E+00
0:  6.8737E-01 -4.7237E-01    *   GAIN = 1.7346E-05
```

In discretization procedure, one needs to enter the sampling period **T**, number of samples between every **T** samples, type of hold device, and modification constant (default is 1). After the calculation of **G(z)**, **H(z)** and **G'(z)**, they are displayed. This is done as follows.As it is seen above the sampling period is .1 second, but step response is displayed every .05 second. In this example a ZOH device is used. It is assumed that there is no computation delay (i.e. m=1). One can also get the printout of above picture

One can also feed a system with its discretized model. In this case sampling period is assumed to be 1 second. This does not effect the design method except for the K_v value.

5.4.. Synthesis Design Method

Before starting Synthesis Design Method, let us review the design constraints.

- Unstable zeros of **G(z)** must be zeros of **M(z)**
- Unstable poles of **G(z)** must be zeros of 1-**M(z)**.
- Unstable mapped poles of **H(s)** must be zeros of **M(z)**.

Recall that the current version of the program handles the first and the last requirements only. Thus **G(z)** should be stable.

The designer has no control over the unstable zeros of **G(z)** or mapped poles of **H(s)**. The user may or may not though add stable zeros of **G(z)** or mapped poles of **H(s)** to the synthesized closed-loop transfer function **M(z)**.

Then one needs to decide whether or not to add new zeros to **M(z)** There are two different screen modes. One shows all the data regarding dominant poles and zero (i.e. location of the pole and zero, damping coefficient, etc. as explained in Ch. 2) and the other shows only the location of dominant poles and zero and damping coefficient. One can use the first screen mode if the system has at least one zero inside the unit circle, which is the first design option. The second option can be used if the system has no zeros inside the unit circle. If one tries to use first screen mode even though the system has no zeros in the unit circle, the program does not accept it and consider it as a second option. If one wants to use first screen mode, the one that shows all the data, without having a zero inside the unit

circle; then one has to choose the third design option. The last option adds a zero inside the unit circle and depending on the dominant poles and the zeros, all the data is displayed. Last option can be chosen in any condition whether the system has any zeros inside the unit circle or not at all.

There may be design specifications to satisfy, such as damping coefficient, peak time, overshoot, and velocity error coefficient. Overshoot requirements in the case of two dominant poles and one dominant zero can be expressed by a damping coefficient (recall the nomograms shown in Ch. 2). Using the damping coefficient and peak time requirement, optimum dominant poles and dominant zero locations are calculated. If the last design option is chosen then the poles are located and are calculated from the poles and the zero. If it is the first design option then one of the zeros inside the unit circle is used instead of the optimum one. Second option does not have that feature. To satisfy the velocity error requirement program assigns a dipole near $z=1$ ("**lag compansation**"). Under the upper half of the unit circle three frames are used to display three different step responses. If the first and the third design options are chosen, the calculated data will be displayed as well. This is demonstrated in the coming example.

One may relocate the dominant poles within the unit circle. One should remember though that these must remain complex. If the third design option is chosen one can also relocate the added zeros in the unity circle along the real axis.

Next one needs to enter the location of the negligible poles. The number of the necessarry negligible closed-loop poles to assure realizability of the controller is calculated by the program automatically.

If a $\mathbf{K_v}$ improvement is desired, the location of the lag compansation dipole is displayed. This subroutine is activated whenever $\mathbf{K_v}$ improvement is requested by the designer and the system has two open-loop poles at $z=1$. Note that if the actual $\mathbf{K_v}$ value is better then the desired one then the subroutine is not called even though the $\mathbf{K_v}$ improvement is requested. The closer the dipole is to $z=1$, the larger is the closed-loop overshoot.

Then one invokes the closed-loop step response for the designed $\mathbf{M(z)}$.

M(z), **C(z)** and step response may be saved. At any stage one can start a new design without losing the saved responses. Ofcourse printouts of **M(z)**, **C(z)** and step response are available upon user request

Example :

Let us try to find the controller for the previous example (of Section 5.2).

Assume the following design specifications :

damping coefficient	ζ	≈ 6
peak overshoot		$\leq 10\%$
peak time	T_p	$\approx 3.T$
velocity error coefficient	K_v	$\geq 100\ sec^{-1}$

To use Synthesis Method press **S** in MENU Then the zeros inside the unit circle, will be listed Enter the number of the zeros to be included in **M(z)**.

```
-----------------------------------------------------------------
      # of THE ZERO , YOU WANT TO INCLUDE

      ZERO( 1) = -1.8064E -01      j+0.0000E+00

          NUMBER OF ZERO #:? 1 -
-----------------------------------------------------------------
```

In the above example the system has one zero inside the unit circle and it is included in **M(z)**. A list of the zeros and the design options is then displayed.

```
-----------------------------------------------------------------
   ZERO( 1) = -2.6170E +00   j+0.0000E+00
   ZERO( 2) = -1.8064E -01   j+0.0000E+00

   IF THE SYSTEM HAS A ONE REAL LHP ZERO          :1/2/3
   IF NOT                                         :2/3
1 - ALL DATA REGARDING DOMINANT POLES AND ZERO.
2 - ONLY THE COORDINATES.
3 - ADD AN EXTRA ZERO.
-----------------------------------------------------------------
```

Since the system has one zero inside the unit circle then one can choose any of the three previously mentioned design options. For choosing the first one press **1** . If the system has more then one zero inside the unit circle, one needs to choose one of them for the calculation of the dominant poles and approximated overshoot and peak time. If the system has only one zero and the first design option is chosen the zero is considered to be dominant. Then one needs to enter the desired damping coefficient, peak time and K_v value

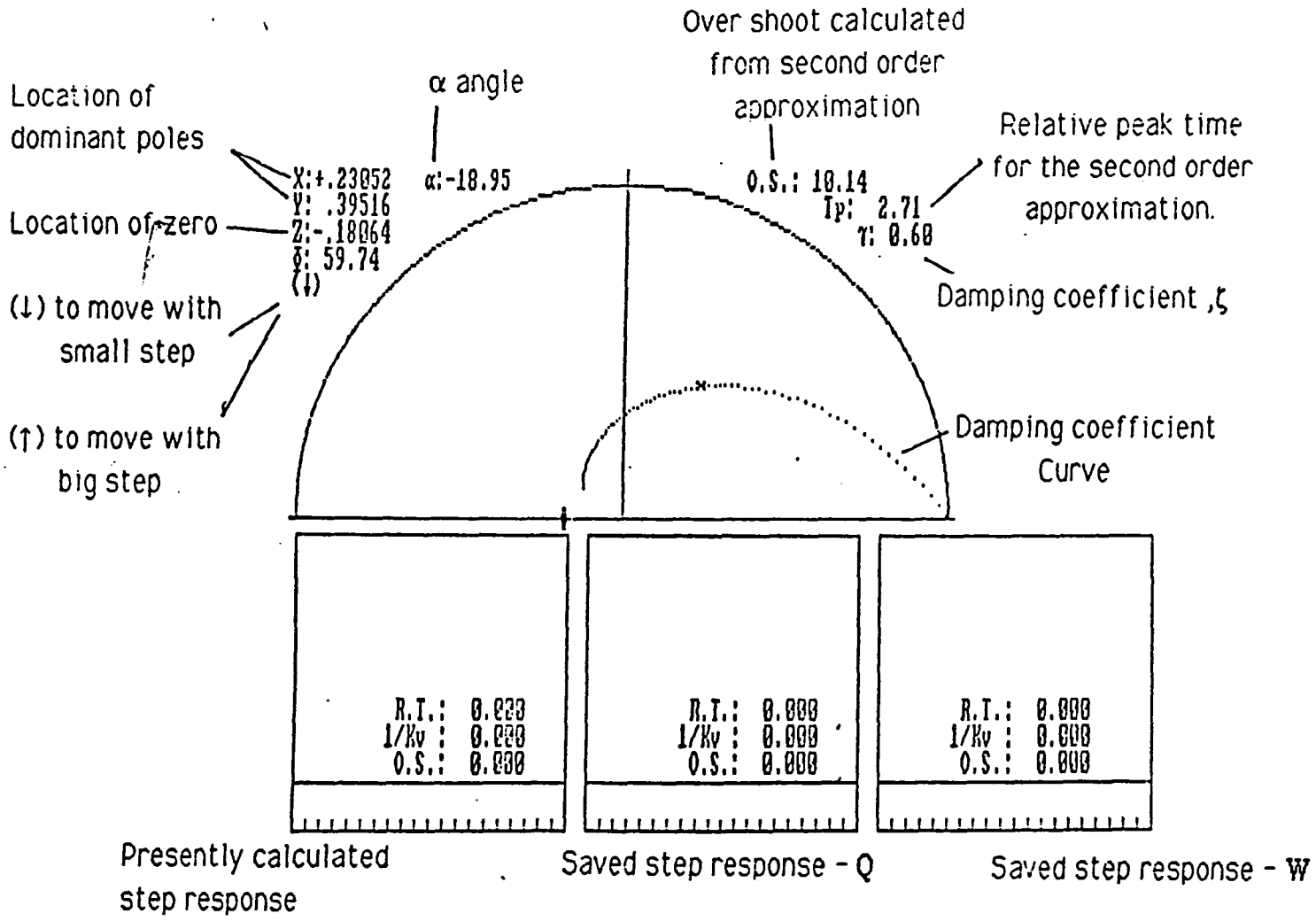


Fig. 5.1 Screen mode in the Synthesis Design method.

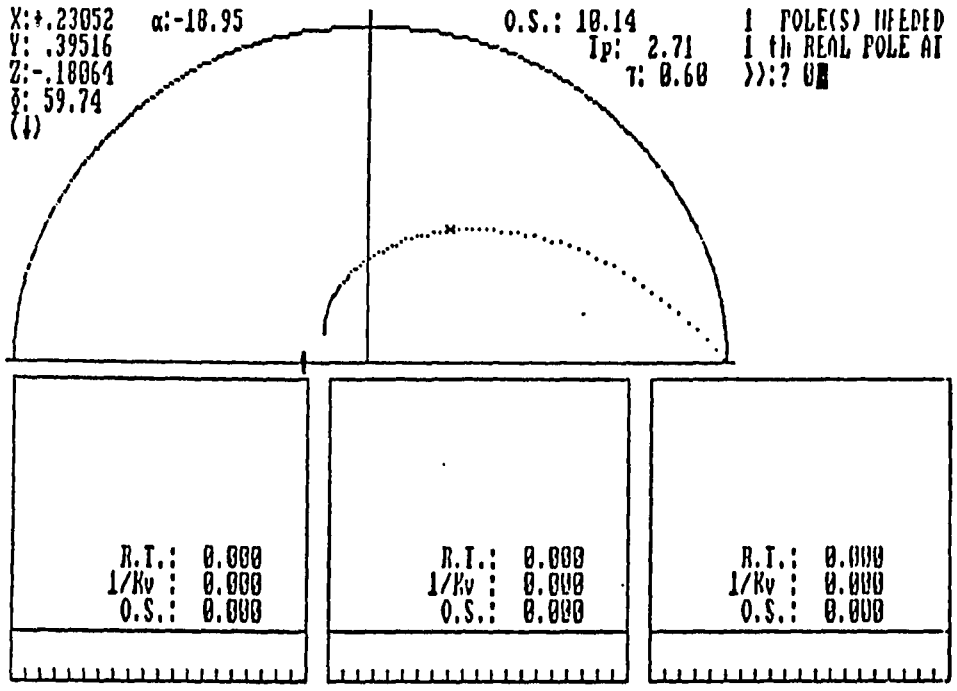


Fig 5.2 Entering the negligible poles.

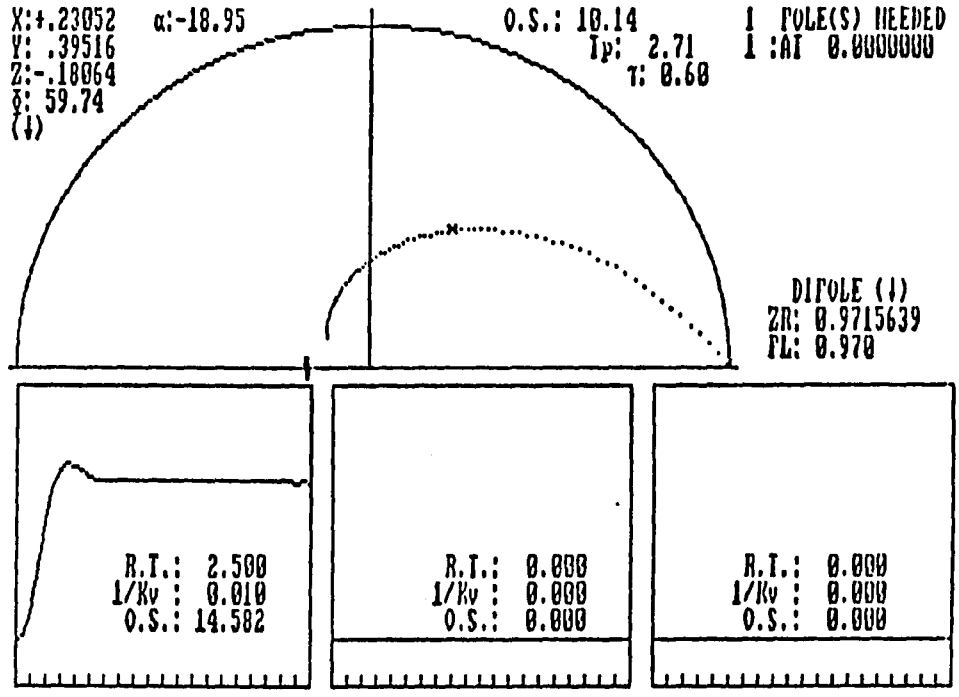


Fig. 5.3 Adding the dipole.

```
ZERO( 1) = -2.6170E +00      j+0.0000E+00
ZERO( 2) = -1.8064E -01      j+0.0000E+00
        INPUT ZETA :? .6 ←
        INPUT Tp/T :? 3 ←
1/Kv IMPROVEMENT (Y) :? Y
   DESIRED 1/Kv      :? .1 ←
```

As one can see above, zero at -.18064 is the dominant zero, damping coefficient is .6 , peak time is 3**T** and $1/K_v$ is .01 . All the design requirements have already been entered except for the peak overshoot. At this point one will see the Fig. 5.1 on the screen.

Second order approximation of the system predicts a 10% overshoot, a .6 damping coefficient, and (2.71)**T** peak time. Let us view the step response. Press **S** to get the step response. Then one will see Fig. 5.2 on screen, meaning that the **S** command was premature.

As one can see, one need to locate a negligible pole to assure realizability of the controller. Let the negligible pole location be at **z**=0. By pressing ← or 0 and ← that can be done. Next is the dipole adding option (for improving K_v, Fig. 5.3).

If the arrow points down (↓) then it gives increaments as much as (.001). If it is up (↑) then the increament is as much as (.01). Let us see what kind of step response one would get after choosing the dipole. Press **S** to continue and observe the actual step response

As can be seen the rise time and K_v requirements are satisfied but the overshoot is too high. Press **Q** and save this step response in the second frame. Then the screen looks like Fig. 5.4 .

One can try to relocate dominant poles, negligible poles or the lag compensation dipole. To illustrate how it works we try the latter. Pressing **S** again, and locate the negligible pole at zero. One can now relocate the dipole. This time the pole is placed at **z**=.98 . To do that press **J** and the arrow will point upwards. Then by pressing **L** the pole is set to .98 and the zero of the dipole will be automatically calculated based on the desired value of K_v. To get the new step response, press **S** again. As can be seen overshoot is still too large. Press **W** to save this step response in the third frame. Then the screen looks like Fig 5.5 .

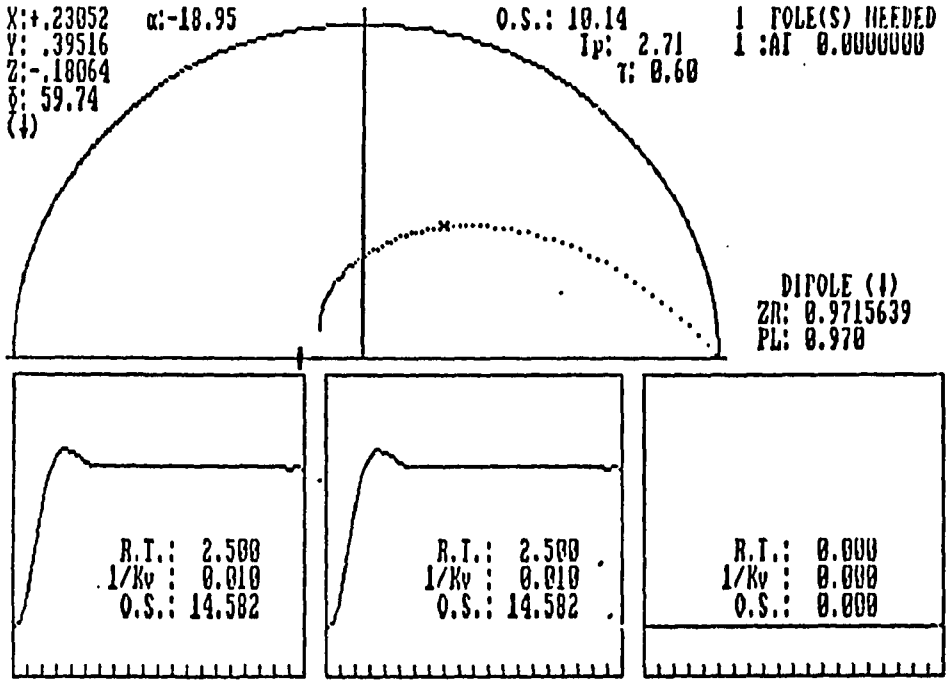


Fig. 5.4 After the step response is saved in the 2nd frame.

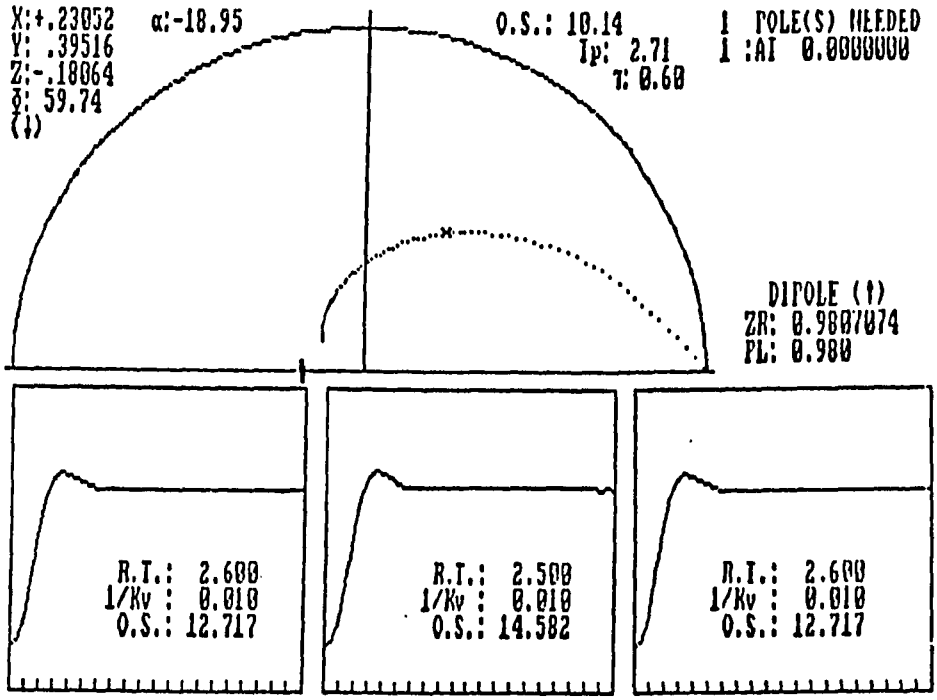


Fig. 5.5 Step response after the dipole is relocated.

It is obvious that the dominant poles need to be relocated to get smaller overshoot. One needs to try another dominant pole location. Let this location be at (x=.12917, y=.3375). This trial and its result shown in the Fig. 5.6 . As one can see the design specifications are met. Press **O** to read the controller and closed-loop transfer function (Fig. 5.7).

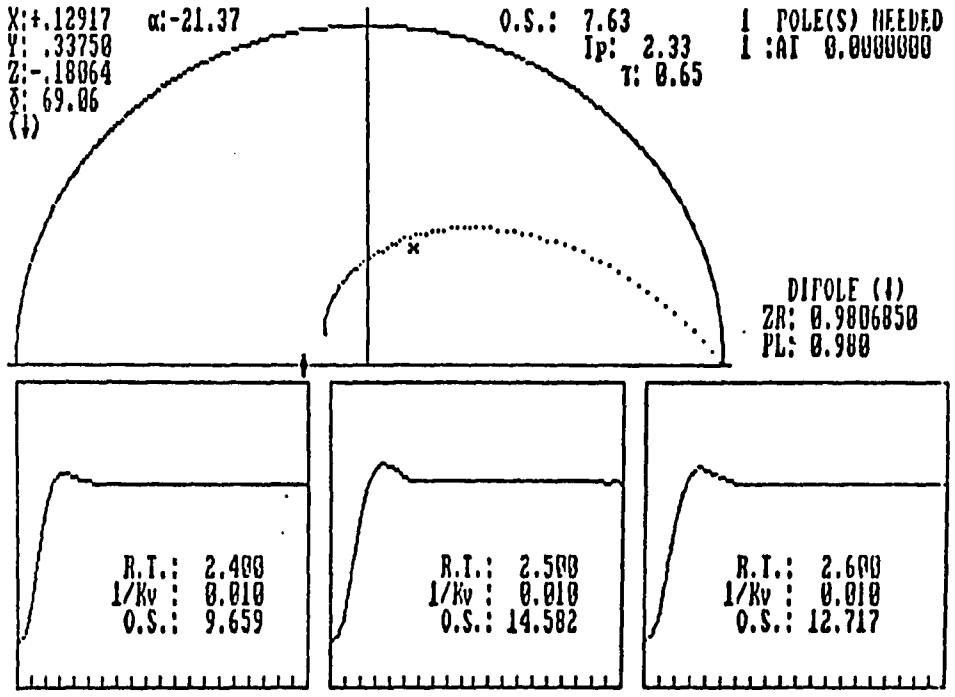


Fig. 5.6 After the dominant poles relocated.

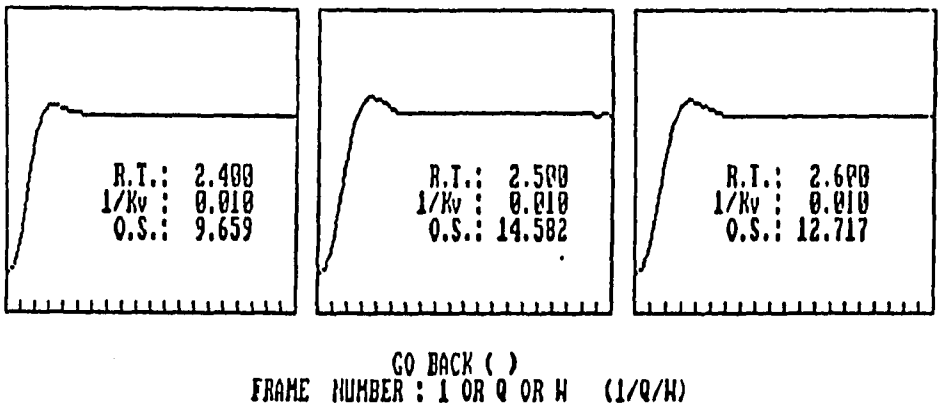
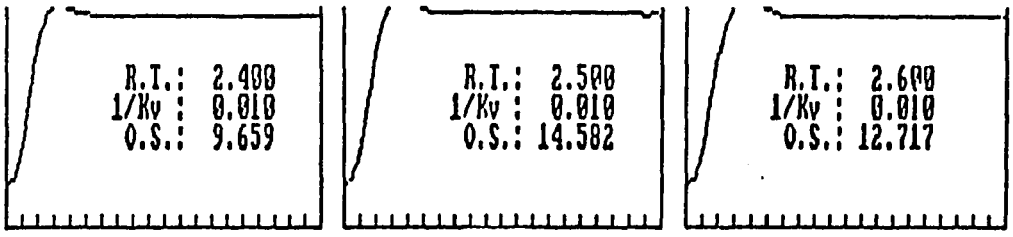


Fig. 5.7 After **O** is pressed.



M(z)

	NUM.	DEN.		NUMERATOR ROOT(S)			DENOMINATOR ROOT(S)	
4:		1.0000E+00	*			*	9.8000E-01	0.0000E+00
3:	1.0000E+00	-1.2383E+00	*	9.8069E-01	0.0000E+00	*	1.2917E-01	3.3750E-01
2:	1.8170E+00	3.8376E-01	*	-1.8064E-01	0.0000E+00	*	1.2917E-01	-3.3750E-01
1:	-2.2709E+00	-1.2798E-01	*	-2.6170E+00	0.0000E+00	*	0.0000E+00	0.0000E+00
0:	-4.6360E-01	0.0000E+00	*	GAIN :	2.11500E-01			

C(z)

	NUM.	DEN.		NUMERATOR ROOT(S)			DENOMINATOR ROOT(S)	
3:	1.0000E+00	1.0000E+00	*	3.6788E-01	0.0000E+00	*	9.9894E-01	0.0000E+00
2:	-1.9551E+00	-4.4983E-01	*	6.0653E-01	0.0000E+00	*	-2.7455E-01	1.5092E-01
1:	1.1787E+00	-4.5037E-01	*	9.8069E-01	0.0000E+00	*	-2.7455E-01	-1.5092E-01
0:	-2.1882E-01	-9.8051E-02	*	GAIN :	1.81567E+03			

PRINT OUT (Y)

Fig. 5.8 After **1** is pressed.

By pressing **1** one gets **M(z)** and **C(z)** for the first shown response as seen in the Fig.5.8 .One can see that the design specifications are achived:

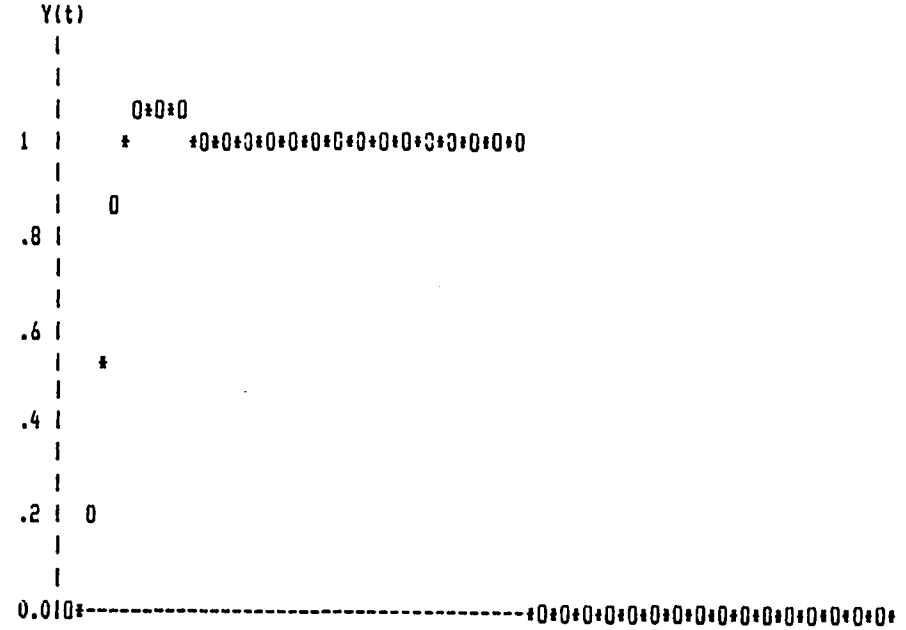
damping coefficient	ζ = .65	≈ .6
peak overshoot	= 9.6%	≤ 10%
peak time	T_p = 3.T	= 3.T
velocity error coefficient	K_v = 100 sec^{-1}	≥ 100 sec^{-1}

. The following printout is also available upon request. In the printout, in addition to controller and the closed-loop transfer functions (in the same form as in Fig. 5.8) the step response is also printed as seen in the Fig. 5.9.

```
----------------------------      N(z)      ----------------------------
     NUM.         DEN.          NUMERATOR ROOT(S)         DENOMINATOR ROOT(S)
4:              1.0000E+00 *                           *  9.8000E-01  0.0000E+00
3: 1.0000E+00 -1.2383E+00 *  9.8069E-01  0.0000E+00  *  1.2917E-01  3.3750E-01
2: 1.8170E+00  3.8376E-01 * -1.8064E-01  0.0000E+00  *  1.2917E-01 -3.3750E-01
1:-2.2709E+00 -1.2798E-01 * -2.6170E+00  0.0000E+00  *  0.0000E+00  0.0000E+00
0:-4.6360E-01  0.0000E+00 *        GAIN :

----------------------------      C(z)      ----------------------------
     NUM.         DEN.          NUMERATOR ROOT(S)         DENOMINATOR ROOT(S)
3: 1.0000E+00  1.0000E+00 *  3.6788E-01  0.0000E+00  *  9.9894E-01  0.0000E+00
2:-1.9551E+00 -4.4983E-01 *  6.0653E-01  0.0000E+00  * -2.7455E-01  1.5092E-01
1: 1.1787E+00 -4.5037E-01 *  9.8069E-01  0.0000E+00  * -2.7455E-01 -1.5092E-01
0:-2.1882E-01 -9.8051E-02 *        GAIN :   1.81567E+03
```

STEP RESPONSE




```
OVERSHOOT            = 10
RAMP-ERROR CONS. (Kv) = 1.003418E-02
RISE TIME            = 2.4
```

O shows the step response at iT, $i=0,1,2,\ldots$

* shows the step response between samples.

Fig. 5.9 Program printout of the step response.

A summary of commands :

B to start over with new options.
O to see the **M(z)** and **C(z)**
J to change cursor movement increaments.
Q to save the **M(z)** and **C(z)** in **Q**.
W to save the **M(z)** and **C(z)** in **W**.
S to start calculating the step response.
R to move cursor up.
C to move cursor down.
F to move cursor left.
D to move cursor right.
K only in 3rd option, to move the zero to the left.
L only in 3rd option, to move the zero to the right.

After **S** is pressed one needs to enter negligible pole location. When it is done, one can do the followings if the K_v improvement is desired or the system is type two.

J to change increaments for pole relocation.
(↓) : .001
(↑) : .01
K to move the pole to the left.
L to move the pole to the right.
S to get out of the subroutine and continue calculation of the step response.

5.5.. Root Locus Design Method

Let **G(z)** be a discretized system transfer function as follows:

$$G(z) = \frac{z + .5}{(z - 1)(z - .5)}$$

First, one needs to enter this system in the program. Since the discretized model of the system is given, press **A** in the MENU program. One then can enter **G(z)**. This procedure is same as the entering the **G(s)**, and will not be shown. When it is done MENU is loaded back. Then press **R** to use Root Locus Program.

As soon as the program is loaded, one is asked to enter damping coefficient and EPS value. Damping coefficient is used to draw a refrence damping coefficient curve for the user convenience. EPS is a parameter used by the root solving subroutine to terminate the iterations [10].

One then sees the following picture on the screen, Fig. 5.10. When the program runs, cursor is located at ($\mathbf{x}_{min}$,0), Δ is zero and the gain **K** is displayed on the right upper corner. No less then 18 function keys (on the keyboard) are featured. These will be reviewed next. The reader is also referred to a summary list at the end of this chapter.

Starting with the movement of the cursor. Suppose, one needs to move the cursor first to the right, then upwards, to the left or downwards all by one unit. Then first press **J** and observe that Δ=1, now. Then press **F** , **D** , **R** , **C** in that given order. One can do the same thing for Δ=.1 . Observe that when Δ=.00001 , cursor movement may not be seen. Refer to the cursor positions to observe that indeed the cursor is moving.

Next is the scaling of **x**, **y**, **K** and Φ. Initially **x** and **y** axes have no scaling information. For that press **N** and see " **X/Y/K/Φ** ^ " at the location where **K** has been displayed. If one needs to view the scaling of **x** axes press (together with SHIFT) **X** then the **x** is displayed at the same place. Do the same thing for **y** axes. Now suppose one needs more root locus points between the same gain values. Press **N** and after " **X/Y/K/Φ** ^ " is

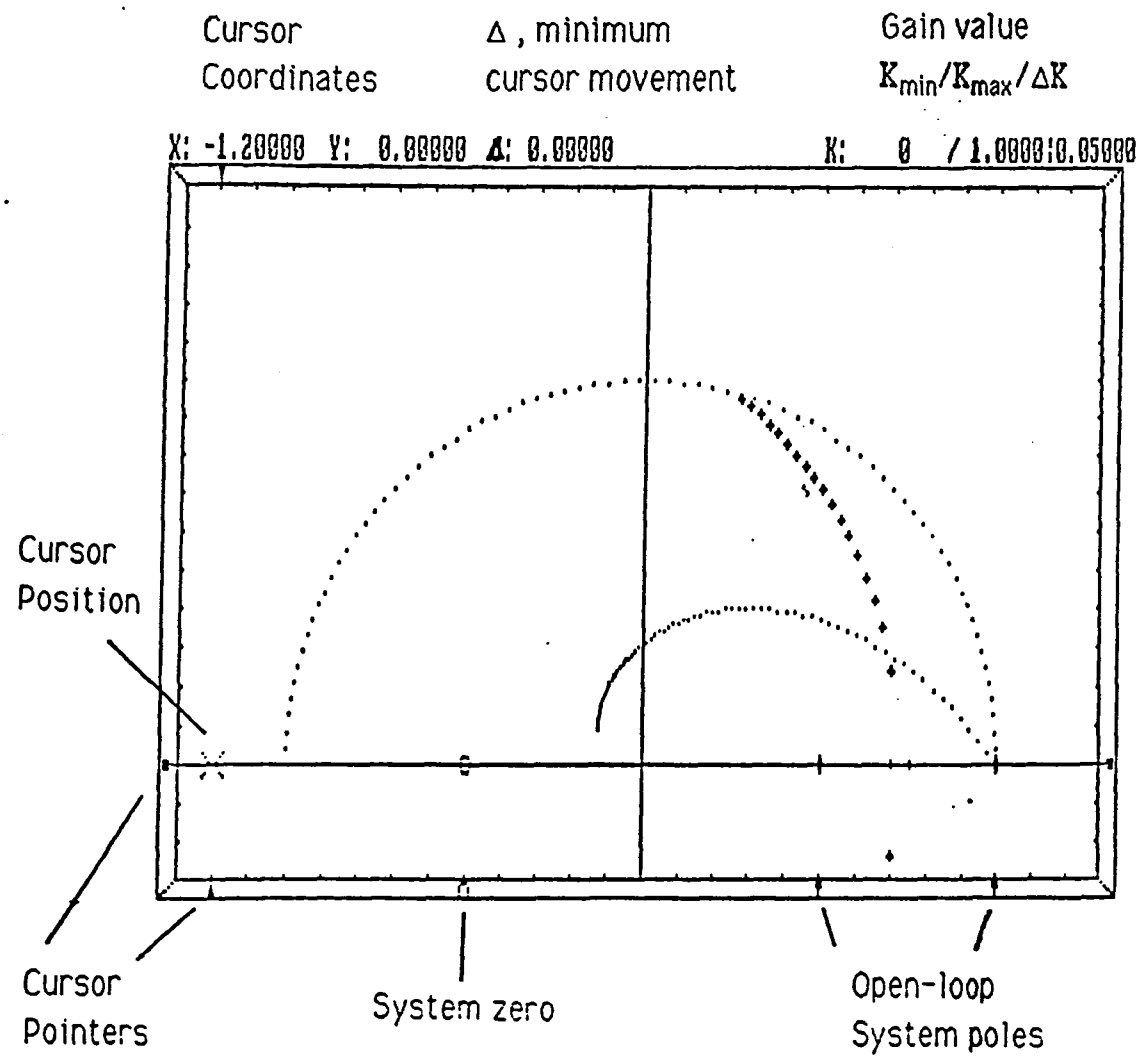


Fig. 5.10 Root locus of $(z-1)(z-.5)+K(z+.5)=0$

displayed press **K** . Note that the letter is capital this time. Then one is asked to enter K_{min} , K_{max} and ΔK , one by one. Note that minimum ΔK value is $(K_{max}-K_{min})/50$. Suppose that part of the root locus goes outside the current frame and one needs to see that part. That requires **x**-axes scaling and/or **y**-axes scaling. **x** and **y** rescaling is done in the same way as that of **K**. However this time Δx can not be smaller than $(x_{max}-x_{min})/30$. Fig. 5.11 shows what happens after the scaling is done along the **x**-axes. Try to choose Δx as smallest desired value. If the program rejects it, then increase that value. Second $x_{max}-x_{min}$ can not be smaller than .0001 and this time Δx is set to .00001 . Similarly for **y**. Φ was defined in Ch. 3, Fig. 3.2 . Corresponds to the starting and ending angles of the unit circle and activated by **F** . Φ_{max} can not be larger than 180° and $\Delta\Phi$ need not be specified.

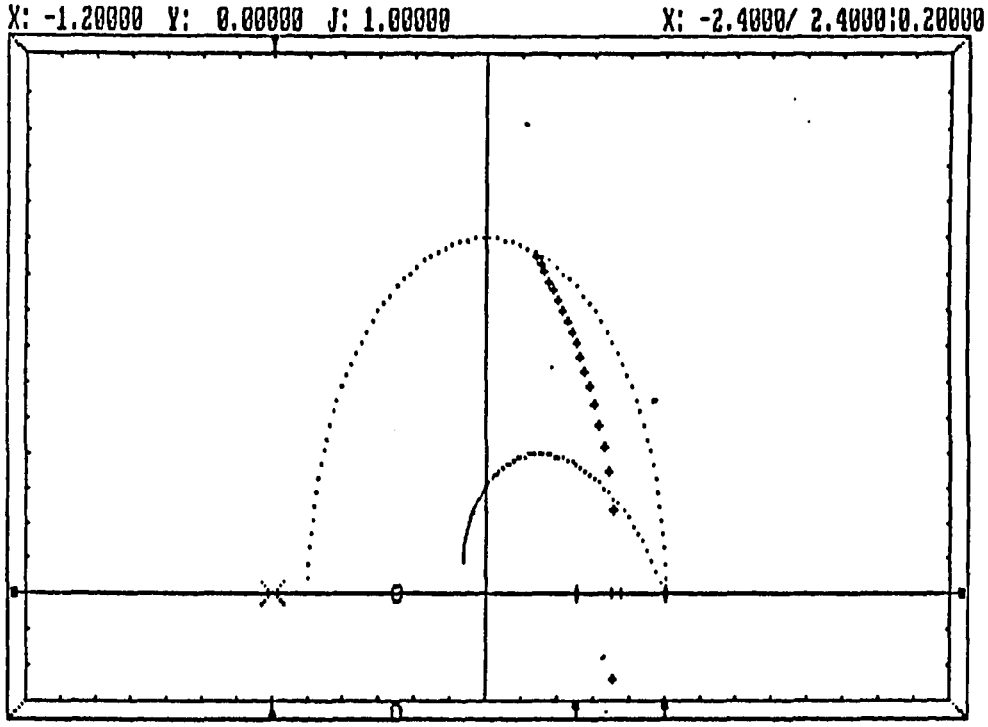


Fig. 5.11 After **x** is rescaled.

One can draw different damping coefficient curves by pressing **Q** and entering the new value. When the new curve is drown, old ones are not erased. If one wants to see only the currently specified curve then press **S** after entering the new value. Pressing **S** cleans the screen and root locus is drown back on the clean frame.

Now, suppose that the root locus (on the screen) looks promising in terms of satisfying the design specifications. One, needs now to move along

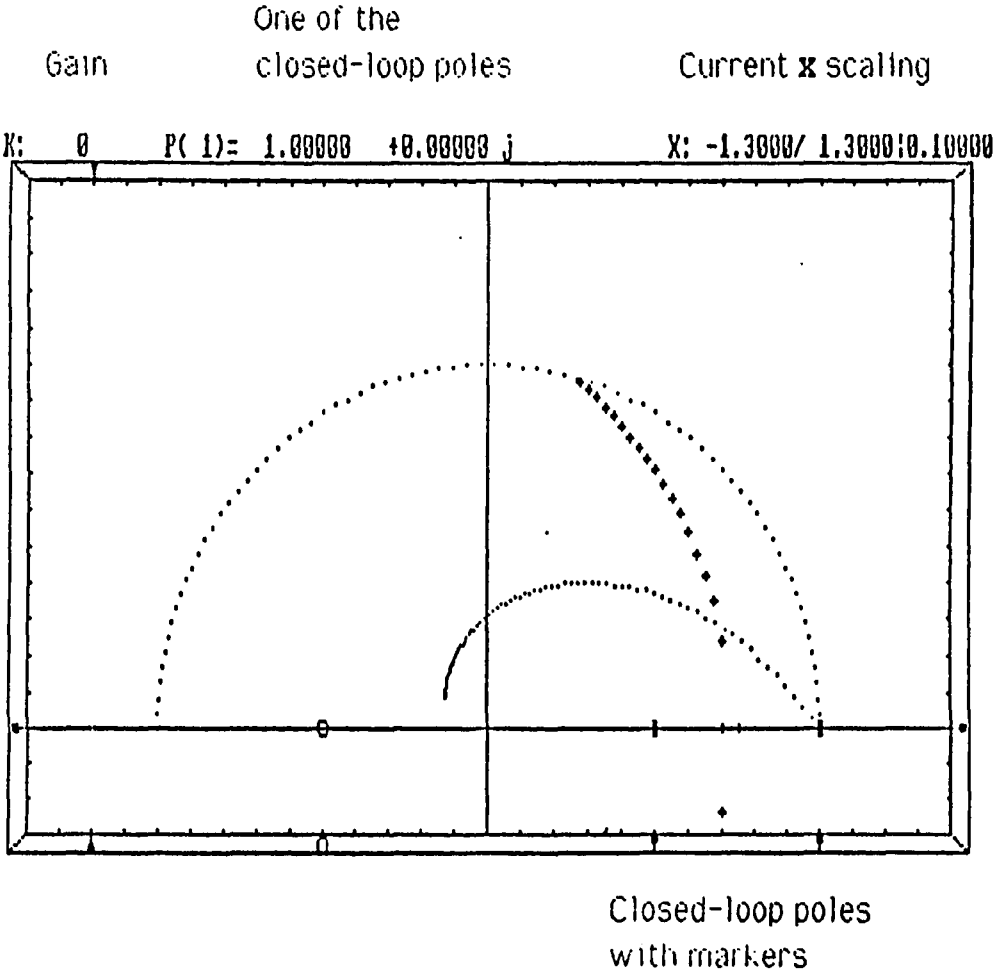


Fig. 5.12 Second screen mode, to move along the root locus

the root locus to the location of the desired closed-loop dominant pole. To do that press **O** , Fig. 5.12 is what the screen looks like after that. Then the cursor, x , disappears and and little markers are seen around the the roots that are closed loop poles for the present gain value. **K** is zero at the beginning and is displayed at the place where **x, y** and Δ used to be. To the right of **K**, one of the closed loop poles is displayed. To see the other closed-loop poles that correspond to the same **K** value simply press the **space bar** . Moving along the root locus means increasing or decreasing the gain. Pressing **R** or **C** increases or decreases, respectively, the gain. After selecting the gain and observing the closed-loop poles there are only two options: back to the "design board" or forward towards completing the design. One may now calculate the closed-loop step response (that corresponds to the selected gain).

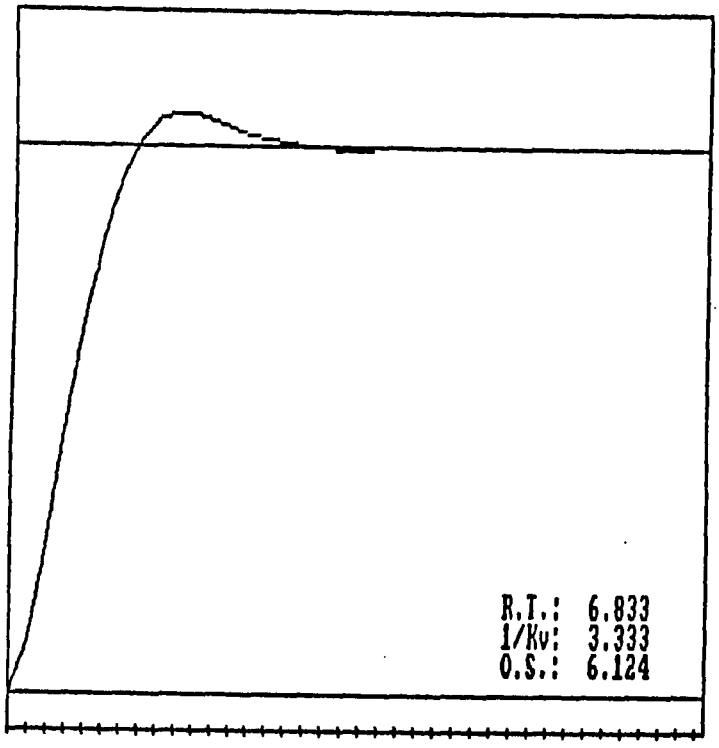


Fig. 5.13 Step response for **K**=.3 in the previous root locus.

To get the step response for these pole/zero configuration press **O** . The " **IN PROCESS** " sign is displayed. Next the step response is displayed on the screen together with the measured system rise time, velocity error coefficient and the peak overshoot, Fig. 5.13. To demonstrate the itarative nature of the design procedure, suppose that the result is unsatisfactory and that the controller for this system is as follows:

$$C'(z) = \frac{z - .4}{z + .2}$$

where

$$C'(z) = \frac{C(z)}{K}$$

That means that one needs to add (or delete) zeros and poles. This option is provided in the first screen mode (the one with the cursor (×) option). So press **B** twice and observe the result. One now can add poles and zeros. Do not forget that improper controllers are not allowed. Therefore poles are added first. To add a pole at **z**=-.2 move cursor to **z**=-.2 and press **P** . Then move cursor to **z**=.4 and press **Z** . Fig 5.14 shows the screen after pole/zero addition is done. Press **S** to see new root locus,Fig. 5.15 . Pressing **H** shows the controller. Pressing **H** second time hides the controller but this takes some time. Press **O** now to move along the root locus. For **K**=.7 root locus and the damping coefficient curve intersects. So increase the gain such that **K**=.7, Fig. 5.16 . Then get the step response for that value by pressing **O**, Fig. 5.17 . At this stage one can see the **C(z)** and **M(z)** by pressing **O** again or get the print out of **C(z)**, **M(z)** and step response by pressing **P** . When the characteristic equation has a pole and zero that are close to each other, that can cause significant numerical errors in the long division solution.

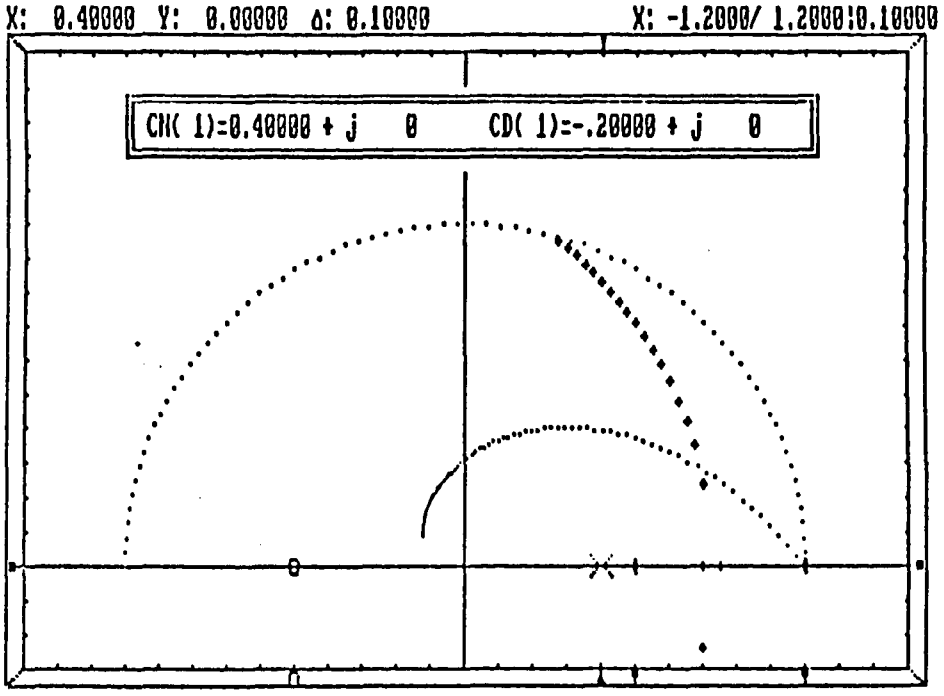


Fig. 5.14 After the controller is added.

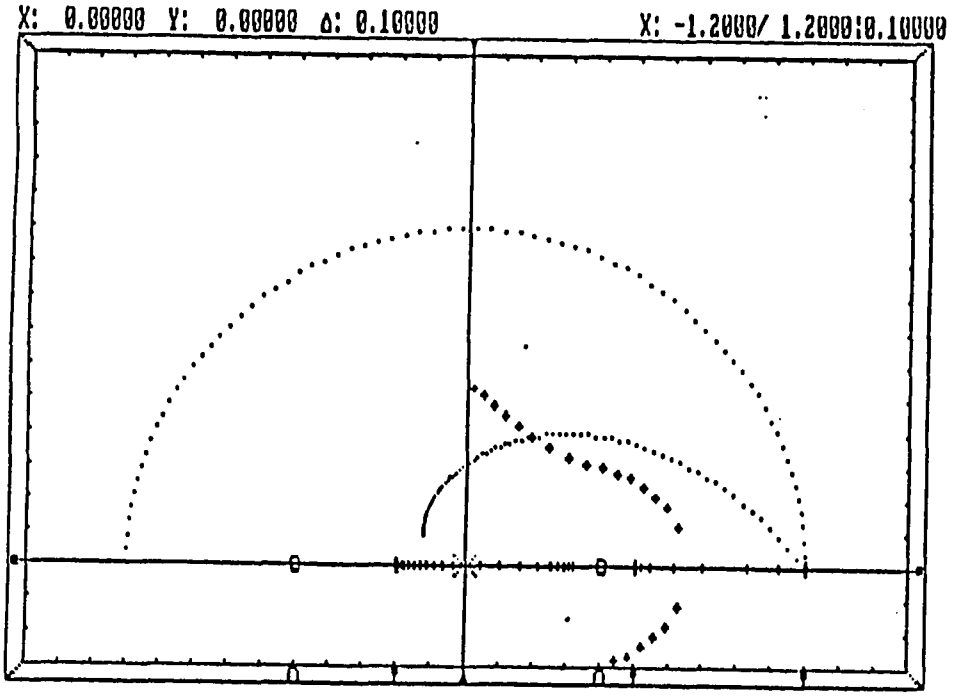


Fig. 5.15 New root locus.

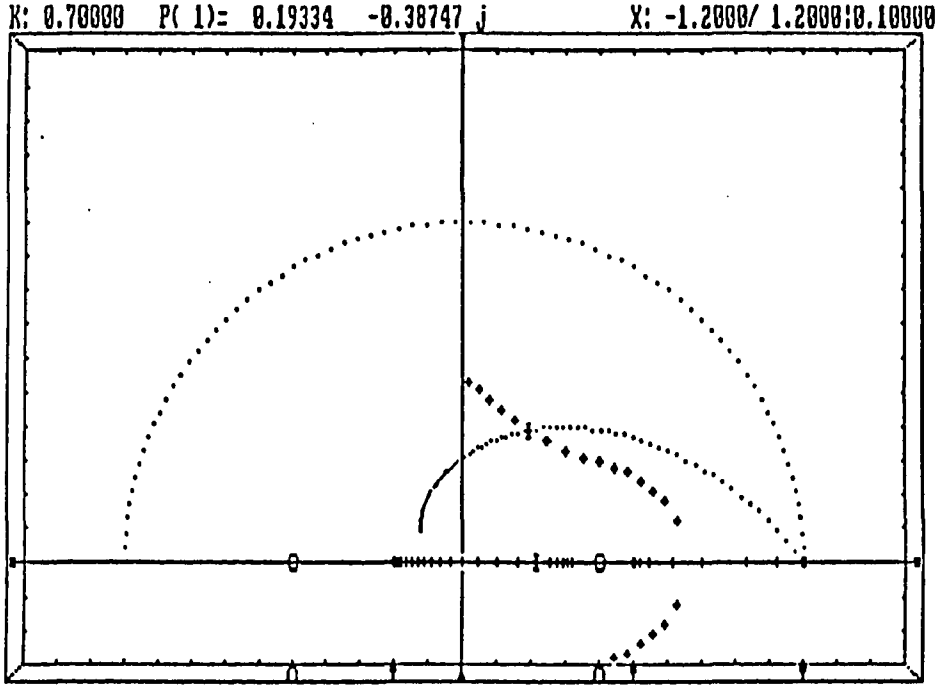


Fig. 5.16 Cursor at K=.7 .

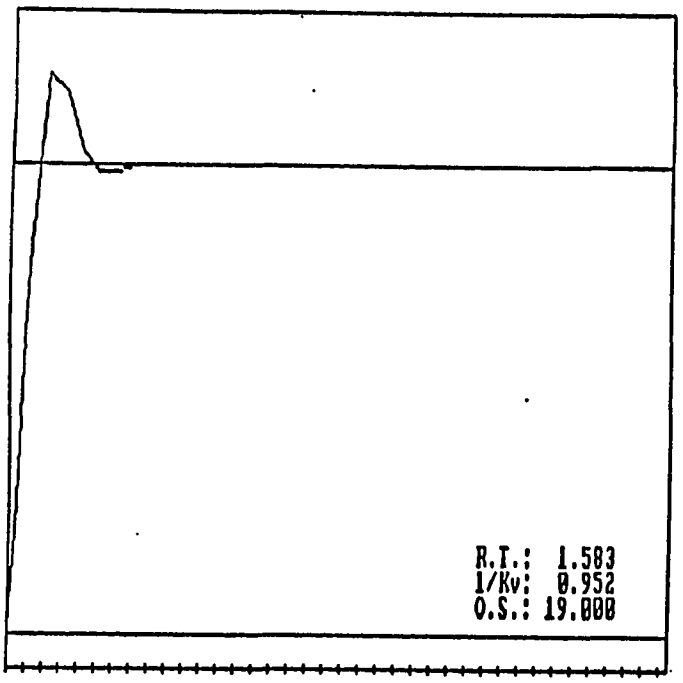


Fig. 5.17 Step response for K=.7 .

These roundoff errors usually become significant at times much greater than the peak time. To suppress that part of the step response, one may press **C** and enter the new number of desired response points. One can improve K_v by pressing **I**. This adding automatically provisions for a dipole to the characteristic equation. The zero value is the design parameter of the dipole option and the pole value is calculated depending on the old and desired K_v values as was previously discussed in Ch. 3. This subrotine is identical to the one in synthesis design method. **K** and **L** keys are used to move zero either to the left or to the right, respectively. **J** changes the increaments and **S** add that dipole to the system and returns the new step response. One may cancel this dipole by pressing **N**. One can try another gain values by going back to the previous screen, by pressing **B**.

Additional key functions that are used in the first screen mode are as follows. One may want to move the cursor to the nearest open-loop poles and zeros for the purpose of exact cancellation or to get a reading of the exact location. This is done by pressing **L**. Then one sees the following on the screen: " **Location of P/Z (P/Z/)** ". Pressing **P** or **Z** displayes the nearest pole or zero location, respectively. Press **P**. Then " **x**=.5 **y**=0 (**Y**/) " is displayed on the screen. If this is the desired pole or zero location, pressing **Y** moves cursor to that location. As a result the cursor is placed exactly at **z**=.5. Similarly for controller poles and zeros by pressing **G**. By that one can move to the nearest controller pole or zero. These options save time and makes pole/zero cancellation easier and more robust numerically. Pressing **M** moves cursor to the nearest tenth decimal point number (i.e. -.3, .7, 1.4).

A summary of commands

E to exit from the root locus program.
M to move cursor nearest tenth decimal point number.
J to change cursor movement increaments.
B to start over the design.
Q to draw new damping caefficient curve.
N to rescale or to get the scaling of **x**, **y**, **K** and **Φ**.
R to move cursor up.
C to move cursor down.
F to move cursor left.
D to move cursor right.
G to move cursor to the nearest **C'(z)** pole/zero location.
L to move cursor to the nearest system pole/zero location.
S to draw the root locus for the current **C'(z)**.
Z to add a zero in **C(z)**.
P to add a pole in **C(z)**.
I to draw the root locus on clean screen.
H to show or hide **C(z)**.
O to move along the root locus.
  B to go back to the design board.
  R to increase the gain.
  C to decrease the gain.
  N to get the scaling of **x**, **y**, **K** and **Φ**.
  O to get the step response.
    B to go back to the root locus board.
    E to exit from the program.
    O to get the **C(z)** and **M(z)**.
    P to get the printout of **C(z)** and **M(z)**.
    C to limit upper boundary of the step response.

I to improve K_v.

- **K** to move zero location to the left.
- **L** to move zero location to the right.
- **J** to change increaments
- **S** to add the dipole and start calculation of the step response

6. CONCLUSION

6.1.. Program Limitations

Current version of the program has the following limitations.

1.. **G(s)** and **H(s)** have to be strictly proper transfer functions. This is very easy to overcome and this limitation will be excluded in the next version.

2.. Root solving program does not give good results for 13^{th} and higher order systems. Besides that for higher orders inputting program also gives errors due to insufficient screen usage. In the next version different root solving algorithm will be used, most likely Lagguerre's Method. Precision is the biggest problem that also causes other problems as will be mentioned in 3. Inputting program will completly be changed to a more friendly and more efficient one.

3.. Again due to the limited accuracy of evaluating exponential functions in BASIC the present version of the program is limited to sampling periods **T** that are no smaller than 1 msec. This limitation can be waived in later versions of the program upon adoption of more accurate methods or accessing math-coprocessor library with BASIC.

4.. Average available RAM after deducting the memory space occupied by the CAD programs is about 30K out of the total of 60K ram used by BASIC. This limits the maximum system order to about 30.

5.. The use of points in between samples for systems of order higher than 10 is not recommended due to excessive computation time. For an example a step response of a 10^{th} order system with one point in between samples and a total of 200 points per screen requires 200 divisions of 20^{th} order polynomials one by another which takes about 45 seconds on an IBM PC/AT. The same task for a 6^{th} order system takes about 25 seconds.

6.. The more points in between samples that are taken the larger cumulative effect of roundoff errors. This is another presicion problem.

7.. In the Synthesis design method no more than one zero can be added in **M(z)** freely. Most importantly the stabilization of the system can not handled in this design method. Systematic screening of candidates C(z) may be done using cascade control pole-placement strategy [12] or the parametrization of all stabilizing controllers [13].

8.. In root locus plots only the most current 50 points are saved in the memory. Since the "old" root locus points are not erased (unless scaling of **x** or **y** is asked) from the screen, the number of root locus points that show up on the screen is practically unlimited.

9.. One of the biggest problems is that for the beginners the program is quite confusing, because there are about 10 keys in the Synthesis and 20 in the Root Locus programs that have functions and one has to refer to the manual each time. Therefore programs will be menu driven in the next versions.

6.2.. Needed Extensions of The Program

While the present version of the CAD program focuses on direct digital control design, it is recognized that many other systematic design strategies are widely used. For instance, the CAD program [14] offers very conveniently designs based on Pole Placement and LQG methods.

There exist many CAD programs for continuous-time SISO controller design and indeed the design of many industrial digital controllers is done in the analog domain either through transformation to the **W**-plane or through analog design and controller discretization [1-5].

An obvious extension of the program described in this paper is to create an analog design version of the program using many of the existing subroutines. In addition to Root Locus and Synthesis Design methods, a Bode-diagram design method should be added featuring loop frequency

response shaping combined with closed-loop step response on a split screen to verify that accomplished phase and gain margines indeed lead to acceptable transient response parameters.

The integration of the analog and digital design parts of the program combined with a large menu of analog controller discretizing options will allow the user the choice of the most suitable digital controller based on a verified closed-loop control performance.

Later versions of the programs of the program should address accuracy problems in implementing control coefficients on a microprocessor using fixed-point arithmetics and quantization noise in the A/D, D/A and the control algorithm,with the objective of optimizing the controller structure.

REFERENCES

[1] Kuo, B. C., *Digital Control Systems*, Holt, Rinehart and Winston, 1980.

[2] Kuo, B. C., *Analysis and Synthesis of Sampled-Data Control Systems*, Prentice Hall, 1963.

[3] Ragazzini, J. R. and G. F. Franklin, *Sample-Data Control Systems*, McGraw-Hill, 1958.

[4] Phillips, C. L. and H. T. Nagle, *Digital Control Systems Analysis and Design*, Prentice Hall, 1984.

[5] Katz, P. , *Digital Control Using Microprocessor*, Prentice Hall, 1981.

[6] Åstrom, K. J., P. Hagander and J. Sternby, "Zeros of Sampled Systems," Automatica, June 1985, 31-38.

[7] Jury, E. I.,"Hidden Oscillations in Sampled-Data Control Systems," Trans. AIEE, 75, 391, 1956.

[8] Truxal, J. G., *Automatic Feedback Control System Synthesis*, McGraw-Hill, 1955.

[9] Chen, C. T., *Linear System Theory and Design*, Holt, Rinehart and Winston, 1984.

[10] Beckett, R. and J. Hurt, *Numerical Calculations and Algorithms*, McGraw-Hill, New York, 1967.

[11] Dahlquist, G. and Å. Bjorck, *Numerical Methods*, Prentice-Hall, 1974.

[12] Wolovich, W. A., *Linear Multivariable Systems*, Springer Verlag, New York, 1979.

[13] Vidyasagar, M., *Control System Synthesis--A Factorization Approach*, The MIT Press, 1985.

[14] "PC-REG--Controller Design and Simulation," Adaptech--Control Systems and Signal Processing Software, France, 1986.

APPENDIXES

A .. MENU Program

This program diplayes the design stages and loads the chosen program which could be inputting a new problem, sysnthesis design method, etc.

```
10 REM  ****************************************
11 REM  *                                        *
12 REM  *    M     M EEEEEE MM    M U     U     *
13 REM  *    MM   MM E      M M   M U     U     *
14 REM  *    M M M M E      M  M  M U     U     *
15 REM  *    M  M  M EEEE   M   M M U     U     *
16 REM  *    M     M E      M    MM U     U     *
17 REM  *    M     M EEEEEE M     M  UUUUU      *
18 REM  *                                        *
19 REM  ****************************************
100 CLS:SCREEN 0,1,0:KEY OFF:LOCATE 5,1
110 PRINT ,"..              PROGRAM MENU                ..":PRINT
120 PRINT ,"..    E  TO ENTER A NEW PROBLEM        <s>  .."
130 PRINT ,"..    A  TO ENTER A NEW PROBLEM        <z>  .."
140 PRINT ,"..    D  TO DISPLAY THE NEW PROBLEM          .."
150 PRINT ,"..    Z  TO DISCRETIZE THE SYSTEM            .."
160 PRINT ,"..    F  TO DISPLAY THE DISCRETIZED SYSTEM   .."
170 PRINT ,"..    S  FOR THE SYNTHESIS DESIGN METHOD     .."
180 PRINT ,"..    R  FOR THE ROOT LOCUS DESIGN METHOD    .."
190 PRINT
200 PRINT ,"..    Q      QUIT                           .."
210 PRINT:PRINT
220 PRINT ,"..          PRESS CAPS LOCK PLEASE           .."
230 C$=INKEY$:IF C$="" THEN GOTO 230
240 IF C$="E" THEN RUN"H1.BAS
250 IF C$="A" THEN RUN"H2.BAS
260 IF C$="D" THEN RUN"H3.BAS
270 IF C$="Z" THEN RUN"H4.BAS
280 IF C$="F" THEN RUN"H4.BAS
290 IF C$="S" THEN RUN"H5S.BAS
300 IF C$="R" THEN RUN"H5R.BAS
310 IF C$="Q" THEN SYSTEM
320 GOTO 230
```

B .. Input Program

100-260 To input the polynomial degrees.
1000-1830 To input **G(s)** denominator.
- 1000-1060 To choose how to input the polynomial.
- 1070-1410 To input the polynomial by its coefficients.
- 1410-1780 To input the polynomial by the roots.
- 1790-1810 To save the polynomial on the disk by its coefficients.

2000-2790 To input **G(s)** numerator.
- 2000-2060 To choose how to input the polynomial.
- 2070-2350 To input the polynomial by its coefficients.
- 2360-2740 To input the polynomial by the roots.
- 2750-2770 To save the polynomial on the disk by the roots.

3000-3830 Same as the 1000-1830.
4000-4790 Same as the 2000-2790.
4800 To close the file that the data saved in.
5000-6040 Root solving subroutine.
7000-7490 To open the file and reads back the data to display and printout.

```
100 REM DEGREE INPUT            '**************************************
110 KEY OFF:CLS:SCREEN 0,0:LOCATE 5,3,1
120 PRINT "   ABOUT   H(s)  and   G(s)   ":LOCATE 8,1
130 PRINT "   G(S)  DENOMINATOR DEGREE     : "
140 PRINT "   G(S)  NUMERATOR   DEGREE     : "
150 PRINT "   H(S)  DENOMINATOR DEGREE     : "
160 PRINT "   H(S)  NUMERATOR   DEGREE     : "
170 LOCATE  8,35: INPUT GDD
180 LOCATE  9,35: INPUT GND
190 LOCATE 10,35: INPUT HDD
200 LOCATE 11,35: INPUT HND
210 IF GND>GDD OR HND>HDD THEN PRINT " .. ONLY PROPER CASES ..":GOTO 170
220 INPUT "          ARE THESE CORRECT (N) ? ";IN$
230 IF IN$="N" THEN GOTO 170
240 OPEN "O",#1,"DATA":WRITE #1,GDD,GND,HDD,HND
250 DEGR=GDD+HDD+2
260 DIM POLY(DEGR),ROOTS(DEGR,1),A(DEGR),B(DEGR),C(DEGR)
1000 REM  G(S) DENOMINATOR          '**************************************
1010 CLS :LOCATE 3,10: PRINT "     G(S)  DENOMINATOR     "
1020 PRINT:IF GDD=0 THEN GOTO 1070
1030 LOCATE 5,6:INPUT "CHARACTERISTIC POLYNOMIAL (1) OR ROOTS (2) ";IN
1040 IF IN<>1 AND IN<>2 THEN GOTO 1030
1050 DIM GDEN(GDD),GZERO(GDD,1)
1060 IF IN=2 THEN GOTO 1410
1070 LOCATE 5,5:PRINT " INPUT   ";GDD;"   TH DEGREE COEFFICIANT          "
1080 INPUT GDEN(GDD)
1090 INPUT "IS IT CORRECT (Y) ";IN$:IF IN$="N" OR GDEN(GDD)=0 THEN GOTO 1070
1100 GNORM=GDEN(GDD)
1110 GDEN(GDD)=1
1120 IF GDD = 0 THEN GOTO 1190
1130 FOR L1=1 TO GDD
1140 LOCATE 5,5:PRINT " INPUT   ";GDD-L1;"   TH DEGREE COEFFICIANT          "
1150 INPUT GDEN(GDD-L1)
1160 GDEN(GDD-L1)=GDEN(GDD-L1)/GNORM
1170 NEXT L1
1180 IF GDD = 0 THEN GOTO 1790
1190 LOCATE 5,5:PRINT"                                          "
1200 FOR L1=0 TO GDD
1210 PRINT "   ";GNORM*GDEN(L1);TAB(20);" . s^ ";L1;"   ( ";GDEN(L1);" )   "
1220 POLY(L1)=GDEN(L1)
1230 NEXT L1:LOCATE 7+GDD,1
1240 IF GDD=0 THEN GOTO 1790
1250 INPUT "  ARE THESE CORRECT (N) ";IN$
1260 IF IN$<>"N" THEN GOTO 1320
1270 INPUT "  WHICH DEGREE COEF. ";IN
1280 IF IN => GDD THEN GOTO 1270
1290 INPUT "       COEFFICIENT : ";GDEN(IN)
1300 GDEN(IN)=GDEN(IN)/GNORM
```

```
1310 GOTO 1190
1320 DEGR=GDD
1330 GOSUB 5000
1340 CLS:LOCATE 5,5:PRINT "  ROOTS OF G(S) DENOMINATOR  "
1350 FOR L1=1 TO GDD
1360 GDRO(L1,0)=ROOTS(L1,0)
1370 GDRO(L1,1)=ROOTS(L1,1)
1380 PRINT"       S( ";L1;" )= ";GDRO(L1,0);TAB(35);GDRO(L1,1);TAB(50);".j"
1390 NEXT L1
1400 GOTO 1790
1410 GDEN(0)=1
1420 L1=0
1430 L1=L1+1
1440 PRINT "            ";L1;" TH ROOT : "
1450 INPUT"     REAL PART = ";GDRO(L1,0)
1460 IF GDD=L1 THEN GOTO 1480
1470 INPUT " IMAGINARY PART = ";GDRO(L1,1)
1480 INPUT "IS IT CORRECT (N) ";IN$
1490 IF IN$="N" THEN GOTO 1440
1500 LOCATE 4+L1,1:PRINT "   ";L1;" TH ROOT = ";GDRO(L1,0);"  ";GDRO(L1,1);" j
                        "
1510 IF GDRO(L1,1)<>0 THEN GOTO 1590
1520 SPARE2=0
1530 FOR L2=0 TO L1
1540 SPARE1=GDEN(L2)
1550 GDEN(L2)=-GDRO(L1,0)*SPARE1+SPARE2
1560 SPARE2=SPARE1
1570 NEXT L2
1590 GOTO 1730
1590 B=-2*GDRO(L1,0)
1600 C=GDRO(L1,0)^2+GDRO(L1,1)^2
1610 SPARE3=0
1620 SPARE2=0
1630 FOR L2=0 TO L1+1
1640 SPARE1=GDEN(L2)
1650 GDEN(L2)=C*SPARE1+B*SPARE2+SPARE3
1660 SPARE3=SPARE2
1670 SPARE2=SPARE1
1680 NEXT L2
1690 GDRO(L1+1,0)=GDRO(L1,0)
1700 GDRO(L1+1,1)=-GDRO(L1,1)
1710 L1=L1+1
1720 LOCATE 4+L1,1:PRINT "   ";L1;" TH ROOT = ";GDRO(L1,0);"  ";GDRO(L1,1);" j
                        "
1730 IF L1<GDD THEN GOTO 1430
1740 PRINT "     G(s)  DENOMINATOR POLYNOMIAL   "
1750 FOR L1=0 TO GDD
1760 PRINT "         ";GDEN(L1);TAB(23);"  s^ ";L1;"       "
1770 NEXT L1
```

```
1780 GNORM=GDEN(GDD)
1790 FOR L1=0 TO GDD:WRITE #1,GDEN(L1): NEXT L1
1800 IF GDD=0 THEN GOTO 1820
1810 FOR L1=1 TO GDD:WRITE #1,GDRO(L1,0),GDRO(L1,1): NEXT L1
1820 PRINT ,,".. PRESS ANY KEY TO CONTINUE .."
1830 C$=INKEY$:IF C$="" THEN GOTO 1830
2000 REM  G(S)   NUMERATOR          '****************************************
2010 CLS :LOCATE 3,10: PRINT "     G(S)    NUMERATOR     "
2020 PRINT:IF GND=0 THEN GOTO 2070
2030 INPUT "     CHARACTERISTIC POLYNOMIAL (1) OR ROOTS (2) ";IN
2040 IF IN<>1 AND IN<>2 THEN GOTO 2030
2050 DIM GNUM(GND),GNRO(GND,1)
2060 IF IN=2 THEN GOTO 2360
2070 FOR L1=0 TO GND
2080 LOCATE 5,5:PRINT " INPUT   ";GND-L1;"   TH DEGREE COEFFICIANT          "
2090 INPUT GNUM(GND-L1)
2100 GNUM(GND-L1)=GNUM(GND-L1)/GNORM
2110 IF L1 = 0 AND GNUM(GND) = 0 THEN GOTO 2080
2120 NEXT L1
2130 LOCATE 5,5:PRINT"                                                  "
2140 FOR L1=0 TO GND
2150 PRINT "   ";GNORM*GNUM(L1);" . s^ ";L1;"   ( ";GNUM(L1);" )   "
2160 POLY(L1)=GNUM(L1)
2170 NEXT L1:LOCATE 7+GND,1
2180 INPUT "  ARE THESE CORRECT (N) ";IN$
2190 IF IN$<>"N" THEN GOTO 2250
2200 INPUT "  WHICH DEGREE COEF. ";IN
2210 IF IN > GND  THEN GOTO 2200
2220 INPUT "       COEFFICIENT : ";GNUM(IN)
2230 IF GNUM(GND)=0 THEN GOTO 2220
2240 GNUM(IN)=GNUM(IN)/GNORM
2250 GOTO 2130
2260 IF GND=0 THEN GOTO 2750
2270 DEGR=GND
2280 GOSUB 5000
2290 CLS:LOCATE 5,5:PRINT "  ROOTS OF G(S) NUMERATOR  "
2300 FOR L1=1 TO GND
2310 GNRO(L1,0)=ROOTS(L1,0)
2320 GNRO(L1,1)=ROOTS(L1,1)
2330 PRINT"       S( ";L1;" )= ";GNRO(L1,0);TAB(35);GNRO(L1,1);TAB(50);".j"
2340 NEXT L1
2350 GOTO 2750
2360 GNUM(0)=1:L1=0
2370 INPUT "  INPUT   GAIN  K = ";GK:INPUT "  IS  IT  CORRECT (N) ";IN$
2380 IF GK = 0 OR IN$ = "N" THEN GOTO 2370
2390 L1=L1+1
2400 PRINT "            ";L1;" TH ROOT : "
2410 INPUT"     REAL PART = ";GNRO(L1,0)
```

```
2420 IF GND=L1 THEN GOTO 2440
2430 INPUT " IMAGINARY PART = ";GNRO(L1,1)
2440 INPUT "IS IT CORRECT (N) ";IN$
2450 IF IN$="N" THEN GOTO 2400
2460 LOCATE 4+L1,1:PRINT "   ";L1;" TH ROOT = ";GNRO(L1,0);"  ";GNRO(L1,1);" j
                          "
2470 IF GNRO(L1,1)<>0 THEN GOTO 2550
2480 SPARE2=0
2490 FOR L2=0 TO L1
2500 SPARE1=GNUM(L2)
2510 GNUM(L2)=-GNRO(L1,0)*SPARE1+SPARE2
2520 SPARE2=SPARE1
2530 NEXT L2
2540 GOTO 2690
2550 B=-2*GNRO(L1,0)
2560 C=GNRO(L1,0)^2+GNRO(L1,1)^2
2570 SPARE3=0
2580 SPARE2=0
2590 FOR L2=0 TO L1+1
2600 SPARE1=GNUM(L2)
2610 GNUM(L2)=C*SPARE1+B*SPARE2+SPARE3
2620 SPARE3=SPARE2
2630 SPARE2=SPARE1
2640 NEXT L2
2650 GNRO(L1+1,0)=GNRO(L1,0)
2660 GNRO(L1+1,1)=-GNRO(L1,1)
2670 L1=L1+1
2680 LOCATE 4+L1,1:PRINT "   ";L1;" TH ROOT = ";GNRO(L1,0);"  ";GNRO(L1,1);" j
                          "
2690 IF L1<GND THEN GOTO 2390
2700 PRINT "     G(s)    NUMERATOR POLYNOMIAL   "
2710 FOR L1=0 TO GND
2720 GNUM(L1)=GNUM(L1)*GK
2730 PRINT "         ";GNUM(L1);TAB(23);"  s^ ";L1;"       "
2740 NEXT L1
2750 FOR L1=0 TO GND:WRITE #1,GNUM(L1): NEXT L1
2760 IF GND=0 THEN GOTO 2780
2770 FOR L1=1 TO GND:WRITE #1,GNRO(L1,0),GNRO(L1,1): NEXT L1
2780 PRINT ,,".. PRESS ANY KEY TO CONTINUE  .."
2790 C$=INKEY$:IF C$="" THEN GOTO 2790
3000 REM  H(S) DENOMINATOR          '**************************************
3010 CLS :LOCATE 3,10: PRINT "     H(S)  DENOMINATOR     "
3020 PRINT:IF HDD=0 THEN GOTO 3070
3030 INPUT "     CHARACTERISTIC POLYNOMIAL (1) OR ROOTS (2) ";IN
3040 IF IN<>1 AND IN<>2 THEN GOTO 3030
3050 DIM HDEN(HDD),HORO(HDD,1)
3060 IF IN=2 THEN GOTO 3410
3070 LOCATE 5,5:PRINT " INPUT   ";HDD;"   TH DEGREE COEFFICIANT          "
```

```
3080 INPUT HDEN(HDD)
3090 INPUT "IS IT CORRECT (N) ";IN$:IF IN$="N" OR HDEN(HDD)=0 THEN GOTO 3070
3100 HNORM=HDEN(HDD)
3110 HDEN(HDD)=1
3120 IF HDD = 0 THEN GOTO 3190
3130 FOR L1=1 TO HDD
3140 LOCATE 5,5:PRINT " INPUT   ";HDD-L1;"   TH DEGREE COEFFICIANT          "
3150 INPUT HDEN(HDD-L1)
3160 HDEN(HDD-L1)=HDEN(HDD-L1)/HNORM
3170 NEXT L1
3180 IF HDD = 0 THEN GOTO 3790
3190 LOCATE 5,5:PRINT"                                                    "
3200 FOR L1=0 TO HDD
3210 PRINT "   ";HNORM*HDEN(L1);TAB(20);" . s^ ";L1;"   ( ";HDEN(L1);" )   "
3220 POLY(L1)=HDEN(L1)
3230 NEXT L1:LOCATE 7+HDD,1
3240 IF HDD=0 THEN GOTO 3790
3250 INPUT "  ARE THESE CORRECT (N) ";IN$
3260 IF IN$<>"N" THEN GOTO 3320
3270 INPUT "  WHICH DEGREE COEF. ";IN
3280 IF IN => HDD THEN GOTO 3270
3290 INPUT "       COEFFICIENT : ";HDEN(IN)
3300 HDEN(IN)=HDEN(IN)/HNORM
3310 GOTO 3190
3320 DEGR=HDD
3330 GOSUB 5000
3340 CLS:LOCATE 5,5:PRINT "  ROOTS OF H(S) DENOMINATOR  "
3350 FOR L1=1 TO HDD
3360 HDRO(L1,0)=ROOTS(L1,0)
3370 HDRO(L1,1)=ROOTS(L1,1)
3380 PRINT"       S( ";L1;" )= ";HDRO(L1,0);TAB(35);HDRO(L1,1);TAB(50);".j"
3390 NEXT L1
3400 GOTO 3790
3410 HDEN(0)=1
3420 L1=0
3430 L1=L1+1
3440 PRINT "            ";L1;" TH ROOT : "
3450 INPUT"     REAL PART = ";HDRO(L1,0)
3460 IF HDD=L1 THEN GOTO 3480
3470 INPUT " IMAGINARY PART = ";HDRO(L1,1)
3480 INPUT "IS IT CORRECT (N) ";IN$
3490 IF IN$="N" THEN GOTO 3440
3500 LOCATE 4+L1,1:PRINT "   ";L1;" TH ROOT = ";HDRO(L1,0);"  ";HDRO(L1,1);" j
                          "
3510 IF HDRO(L1,1)<>0 THEN GOTO 3590
3520 SPARE2=0
3530 FOR L2=0 TO L1
3540 SPARE1=HDEN(L2)
```

```
3550 HDEN(L2)=-HDRO(L1,0)*SPARE1+SPARE2
3560 SPARE2=SPARE1
3570 NEXT L2
3580 GOTO 3730
3590 B=-2*HDRO(L1,0)
3600 C=HDRO(L1,0)^2+HDRO(L1,1)^2
3610 SPARE3=0
3620 SPARE2=0
3630 FOR L2=0 TO L1+1
3640 SPARE1=HDEN(L2)
3650 HDEN(L2)=C*SPARE1+B*SPARE2+SPARE3
3660 SPARE3=SPARE2
3670 SPARE2=SPARE1
3680 NEXT L2
3690 HDRO(L1+1,0)=HDRO(L1,0)
3700 HDRO(L1+1,1)=-HDRO(L1,1)
3710 L1=L1+1
3720 LOCATE 4+L1,1:PRINT "   ";L1;" TH ROOT = ";HDRO(L1,0);"  ";HDRO(L1,1);" j
                    "
3730 IF L1<HDD THEN GOTO 3420
3740 PRINT "     H(s)  DENOMINATOR POLYNOMIAL   "
3750 FOR L1=0 TO HDD
3760 PRINT "         ";HDEN(L1);TAB(23);"  s^ ";L1;"       "
3770 NEXT L1
3780 HNORM=HDEN(HDD)
3790 FOR L1=0 TO HDD:WRITE #1,HDEN(L1): NEXT L1
3800 FOR L1=0 TO HDD:WRITE #1,HDEN(L1): NEXT L1
3810 IF HDD=0 THEN GOTO 3930
3820 FOR L1=1 TO HDD:WRITE #1,HDRO(L1,0),HDRO(L1,1): NEXT L1
3830 PRINT ,,".. PRESS ANY KEY TO CONTINUE  .."
3840 C$=INKEY$:IF C$="" THEN GOTO 3840
4000 REM  H(S)   NUMERATOR          '*************************************
4010 CLS :LOCATE 3,10: PRINT "     H(S)    NUMERATOR     "
4020 PRINT:IF HND=0 THEN GOTO 4070
4030 INPUT "     CHARACTERISTIC POLYNOMIAL (1) OR ROOTS (2) ";IN
4040 IF IN<>1 AND IN<>2 THEN GOTO 4030
4050 DIM HNUM(HND),HNRO(HND,1)
4060 IF IN=2 THEN GOTO 4360
4070 FOR L1=0 TO HND
4080 LOCATE 5,5:PRINT " INPUT   ";HND-L1;"   TH DEGREE COEFFICIANT          "
4090 INPUT HNUM(HND-L1)
4100 HNUM(HND-L1)=HNUM(HND-L1)/HNORM
4110 IF L1 = 0 AND HNUM(HND) = 0 THEN GOTO 4080
4120 NEXT L1
4130 LOCATE 5,5:PRINT"                                          "
4140 FOR L1=0 TO HND
4150 PRINT "   ";HNORM*HNUM(L1);" . s^ ";L1;"   ( ";HNUM(L1);" )   "
4160 POLY(L1)=HNUM(L1)
```

```
4170 NEXT L1:LOCATE 7+HND,1
4180 INPUT "  ARE THESE CORRECT (N) ";IN$
4190 IF IN$<>"N" THEN GOTO 4260
4200 INPUT "  WHICH DEGREE COEF. ";IN
4210 IF IN > HND  THEN GOTO 4200
4220 INPUT "       COEFFICIENT : ";HNUM(IN)
4230 IF HNUM(HND)=0 THEN GOTO 4220
4240 HNUM(IN)=HNUM(IN)/HNORM
4250 GOTO 4130
4260 IF HND=0 THEN GOTO 4750
4270 DEGR=HND
4280 GOSUB 5000
4290 CLS:LOCATE 5,5:PRINT "  ROOTS OF H(S) NUMERATOR  "
4300 FOR L1=1 TO HND
4310 HNRO(L1,0)=ROOTS(L1,0)
4320 HNRO(L1,1)=ROOTS(L1,1)
4330 PRINT"       S( ";L1;" )= ";HNRO(L1,0);TAB(35);HNRO(L1,1);TAB(50);".j"
4340 NEXT L1
4350 GOTO 4750
4360 HNUM(0)=1:L1=0
4370 INPUT "  INPUT   GAIN  K = ";HK:INPUT "  IS  IT  CORRECT (N) ";IN$
4380 IF HK = 0 OR IN$ = "N" THEN GOTO 4370
4390 L1=L1+1
4400 PRINT "            ";L1;" TH ROOT : "
4410 INPUT"     REAL PART = ";HNRO(L1,0)
4420 IF HND=L1 THEN GOTO 4440
4430 INPUT " IMAGINARY PART = ";HNRO(L1,1)
4440 INPUT "IS IT CORRECT (N) ";IN$
4450 IF IN$="N" THEN GOTO 4400
4460 LOCATE 4+L1,1:PRINT "   ";L1;" TH ROOT = ";HNRO(L1,0);"  ";HNRO(L1,1);" j
                          "
4470 IF HNRO(L1,1)<>0 THEN GOTO 4550
4480 SPARE2=0
4490 FOR L2=0 TO L1
4500 SPARE1=HNUM(L2)
4510 HNUM(L2)=-HNRO(L1,0)*SPARE1+SPARE2
4520 SPARE2=SPARE1
4530 NEXT L2
4540 GOTO 4590
4550 B=-2*HNRO(L1,0)
4560 C=HNRO(L1,0)^2+HNRO(L1,1)^2
4570 SPARE3=0
4580 SPARE2=0
4590 FOR L2=0 TO L1+1
4600 SPARE1=HNUM(L2)
4610 HNUM(L2)=C*SPARE1+B*SPARE2+SPARE3
4620 SPARE3=SPARE2
4630 SPARE2=SPARE1
```

```
4640 NEXT L2
4650 HNRO(L1+1,0)=HNRO(L1,0)
4660 HNRO(L1+1,1)=-HNRO(L1,1)
4670 L1=L1+1
4680 LOCATE 4+L1,1:PRINT "   ";L1;" TH ROOT = ";HNRO(L1,0);"  ";HNRO(L1,1);" j
                          "
4690 IF L1<HND THEN GOTO 4390
4700 PRINT "     H(s)    NUMERATOR POLYNOMIAL   "
4710 FOR L1=0 TO HND
4720 HNUM(L1)=HNUM(L1)*HK
4730 PRINT "          ";HNUM(L1);TAB(23);"  s^ ";L1;"       "
4740 NEXT L1
4750 FOR L1=0 TO HND:WRITE #1,HNUM(L1): NEXT L1
4760 IF HND=0 THEN GOTO 4780
4770 FOR L1=1 TO HND:WRITE #1,HNRO(L1,0),HNRO(L1,1): NEXT L1
4780 PRINT ,,".. PRESS ANY KEY TO CONTINUE  .."
4790 C$=INKEY$:IF C$="" THEN GOTO 4790
4800 WRITE #1,HOLD,TC
4810 CLOSE #1
4820 GOTO 7000
5000 REM *-*-*-*-  ROOTS
5010 N=DEGR
5020 K=0
5030 IF POLY(K)<>0 THEN GOTO 5060
5040 K=K+1
5050 GOTO 5030
5060 N=N-K
5070 FOR L1=0 TO DEGR
5080 ROOTS(L1,0)=0
5090 ROOTS(L1,1)=0
5100 NEXT L1
5110 FOR L1=0 TO N
5120 A(L1)=POLY(L1+K)
5130 NEXT L1
5140 IF N<4 THEN GOTO  5520
5150 IT=0:H0=((N-1)*A(N-1))^2-2*N*(N-1)*A(N)*A(N-2)
5160 IF H0<0 THEN GOTO 5260
5170 DET=A(N-1)+SQR(H0)
5180 IF DET=0 THEN DET=.00001
5190 P01=N*A(N)/DET
5200 DET=A(N-1)-SQR(H0)
5210 IF DET=0 THEN R=1:S=-1:GOTO 5310
5220 P02=N*A(N)/DET
5230 R=-(P01+P02)
5240 S=-P01*P02
5250 GOTO 5310
5260 DET=(A(N-1))^2+ABS(H0)
5270 IF DET=0 THEN R=1:S=-1:GOTO 5310
```

```
5280 MG=N*A(N)/DET
5290 R=-2*MG*A(N-1)
5300 S=-N*A(N)*MG
5310 IT=IT+1:SPARE1=0:SPARE2=0:SPARE3=0:SPARE4=0
5320 FOR L1=0 TO N
5330 B(N-L1)=CDBL(A(N-L1))+CDBL(R)*CDBL(SPARE1)+CDBL(S)*CDBL(SPARE2)
5340 SPARE2=CDBL(SPARE1)
5350 SPARE1=CDBL(B(N-L1))
5360 C(N-L1)=CDBL(B(N-L1))+CDBL(R)*CDBL(SPARE3)+CDBL(S)*CDBL(SPARE4)
5370 SPARE4=CDBL(SPARE3)
5380 SPARE3=CDBL(C(N-L1))
5390 NEXT L1
5400 DET=CDBL(C(1))*CDBL(C(3))-CDBL(C(2))*CDBL(C(2))
5410 IF DET=0 THEN DET=.000001
5420 DR=(CDBL(B(1))*CDBL(C(2))-CDBL(B(0))*CDBL(C(3)))/CDBL(DET)
5430 DS=(CDBL(B(0))*CDBL(C(2))-CDBL(B(1))*CDBL(C(1)))/CDBL(DET)
5440 R=CDBL(R)+CDBL(DR)
5450 S=CDBL(S)+CDBL(DS):IF IT>70 THEN GOTO 5470
5460 IF ABS(CDBL(DR))>1E-08 OR ABS(CDBL(DS))>1E-08 THEN GOTO 5310
5470 AA=1
5480 FOR L1=0 TO N-2
5490 A(L1)=CDBL(B(L1+2))
5500 NEXT L1
5510 GOTO 5870
5520 REM   3'rd   DEGREE  ROOTS
5530 IF N<3 THEN GOTO 5920
5540 AA=CDBL(A(2))/CDBL(A(3)):B=CDBL(A(1))/CDBL(A(3)):C=CDBL(A(0))/CDBL(A(3))
5550 P=CDBL(B)-CDBL((CDBL(AA))^2#)/3#
5560 Q=CDBL(CDBL(C)-CDBL(AA)*CDBL(B)/3#+(2#*CDBL((CDBL(AA))^3))/CDBL(27))
5570 US=CDBL((CDBL(P)/3#)^3#+CDBL((CDBL(Q)/2#)^2))
5580 IF US<0 THEN GOTO 5720
5590 U=CDBL(SQR(CDBL(US)))
5600 VV=CDBL(U)-CDBL(Q)/2#
5610 V=CDBL(SGN(VV)*CDBL((ABS(CDBL(VV)))^(1#/3)))
5620 WW=CDBL(-CDBL(U)-CDBL(Q)/2#)
5630 W=CDBL(SGN(WW)*(ABS(CDBL(WW)))^(1#/3))
5640 ROOTS(N,0)=CDBL(CDBL(V)+CDBL(W)-CDBL(AA)/3#)
5650 REAL=CDBL(-(CDBL(V)+CDBL(W))/2#-CDBL(AA)/3#)
5660 IMAG=CDBL(CDBL(SQR(3#)/2#)*(CDBL(V)-CDBL(W)))
5670 ROOTS(N-1,0)=CDBL(REAL)
5680 ROOTS(N-2,0)=CDBL(REAL)
5690 ROOTS(N-1,1)=CDBL(IMAG)
5700 ROOTS(N-2,1)=-CDBL(IMAG)
5710 GOTO 5800
5720 U=-CDBL((CDBL(CDBL(CDBL(P)/3#)^3#))^(1#/2#))
5730 TE=-CDBL(CDBL(Q)/2#/CDBL(U))
5740 PI=CDBL(4*ATN(1))
5750 TE=CDBL(PI)/2-CDBL(ATN(CDBL(TE)/(CDBL(SQR(1#-CDBL(TE)*CDBL(TE)+1E-10#)))))
```

```
5760 KC=CDBL(2#*(CDBL(U))^(1#/3))
5770 ROOTS(N,0)=CDBL(CDBL(KC)*CDBL(COS(CDBL(TE)/3#))-CDBL(AA)/3#)
5780 ROOTS(N-1,0)=CDBL(KC)*CDBL(COS(CDBL(TE)/3#-2#*CDBL(PI)/3#))-CDBL(AA)/3#
5790 ROOTS(N-2,0)=CDBL(KC)*CDBL(COS(CDBL(TE)/3#+2#*CDBL(PI)/3#))-CDBL(AA)/3#
5800 N=N-3
5810 GOTO 6040
5820 REM   2'nd  DEGREE   ROOTS
5830 IF N<2 THEN GOTO 6020
5840 AA=CDBL(A(2))
5850 R=-CDBL(A(1))
5860 S=-CDBL(A(0))
5870 B=-CDBL(R)
5880 C=-CDBL(S)
5890 DELTA=CDBL((CDBL(B))^2#-4#*CDBL(AA)*CDBL(C))
5900 IF DELTA<0 THEN GOTO 5940
5910 ROOTS(N,0)=-CDBL((CDBL(B)+CDBL(SQR(CDBL(DELTA+1E-38))))/2#/CDBL(AA))
5920 ROOTS(N-1,0)=-CDBL((CDBL(B)-CDBL(SQR(CDBL(DELTA+1E-38))))/2#/CDBL(AA))
5930 GOTO 6000
5940 REAL=-CDBL(CDBL(B)/2#/CDBL(AA))
5950 IMAG=CDBL(CDBL(SQR(ABS(CDBL(DELTA))))/2#/CDBL(AA))
5960 ROOTS(N,0)=CDBL(REAL)
5970 ROOTS(N-1,0)=CDBL(REAL)
5980 ROOTS(N,1)=CDBL(IMAG)
5990 ROOTS(N-1,1)=-CDBL(IMAG)
6000 N=N-2
6010 IF N>1 THEN GOTO 5140
6020 IF N<1 THEN GOTO 6040
6030 ROOTS(N,0)=-A(0)/A(1)
6040 RETURN
7000 REM ********************************
7010 PO=0:OPEN "I",#1,"DATA"
7020 INPUT #1,GDD,GND,HDD,HND
7030 FOR L=0 TO GDD      :INPUT #1,GDE(L)              :NEXT L
7040 FOR L=1 TO GDD      :INPUT #1,GDR(L,0),GDR(L,1)   :NEXT L
7050 FOR L=0 TO GND      :INPUT #1,GNU(L)              :NEXT L
7060 FOR L=1 TO GND      :INPUT #1,GNR(L,0),GNR(L,1)   :NEXT L
7070 FOR L=0 TO HDD      :INPUT #1,HDE(L)              :NEXT L
7080 FOR L=1 TO HDD      :INPUT #1,HDR(L,0),HDR(L,1)   :NEXT L
7090 FOR L=0 TO HND      :INPUT #1,HNU(L)              :NEXT L
7100 FOR L=1 TO HND      :INPUT #1,HNR(L,0),HNR(L,1)   :NEXT L
7110 CLOSE #2:D=GDD+HDD
7120 REM ------------------------------------------------------------------
7130 DIM SNPL(D+2),      SDPL(D+2),      SNRT(D+2,1),    SDRT(D+2,1)
7140 CLS:REM ***********
7150 GGN=GNU(GND):SND=GND:FOR L=0 TO SND
7160 SNPL(L)=GNU(L)/GGN:SNRT(L,0)=GNR(L,0):SNRT(L,0)=GNR(L,0):NEXT L
7170 SGA=GGN:SDD=GDD:FOR L=0 TO SDD
7180 SDPL(L)=GDE(L):SDRT(L,0)=GDR(L,0):SDRT(L,1)=GDR(L,1):NEXT L
```

```
7190 S$="G(s)":SGA=GGN:XX=1
7200 IF PO<>1 THEN GOSUB 7320 ELSE GOSUB 7440
7210 REM **********
7220 HGN=HNU(HND):SND=HND:FOR L=0 TO SND
7230 SNPL(L)=HNU(L)/HGN:SNRT(L,0)=HNR(L,0):SNRT(L,0)=HNR(L,0):NEXT L
7240 SGA=HGN:SDD=HDD:FOR L=0 TO SDD
7250 SDPL(L)=HDE(L):SDRT(L,0)=HDR(L,0):SDRT(L,1)=HDR(L,1):NEXT L
7260 S$="H(s)":SGA=HGN:XX=GDD+5
7270 IF PO<>1 THEN GOSUB 7320 ELSE GOSUB 7440
7280 PRINT ,,".. PRINT OUT (Y) .."
7290 C$=INKEY$:IF C$="" THEN GOTO 7290
7300 IF C$<>"Y" THEN RUN"H3
7310 PO=1:GOTO 7150
7320 REM ------------------------------------------------------------
7330 Q$=                                        "   ----------------------------
    &     ---------------------------  "
7340 W$="NUM.        DEN.         NUMERATOR ROOT(S)       DENOMINATOR ROOT(S) "
7350 R$=                    "##:            ##.####^^^^ +
   + ##.####^^^^ ##.####^^^^"
7360 T$=                    "##:##.####^^^^ ##.####^^^^ + ##.####^^^^ ##.####^^^
^  + ##.####^^^^ ##.####^^^^"
7370 Y$="##:##.####^^^^ ##.####^^^^ +       GAIN=  ##.####^^^^     "
7380 U$="##:##.####^^^^ ##.####^^^^ +                                 "
7390 LOCATE XX,1:PRINT USING Q$;S$:PRINT TAB(8);W$:FOR L=SDD TO SND+1 STEP -1
7400 PRINT USING R$;L;SDPL(L);SDRT(L,0);SDRT(L,1):NEXT L
7410 FOR L=SND TO 1 STEP -1
7420 PRINT USING T$;L;SNPL(L);SDPL(L);SNRT(L,0);SNRT(L,1);SDRT(L,0);SDRT(L,1)
7430 NEXT L:PRINT USING Y$;0;SNPL(0);SDPL(0);SGA:PRINT:RETURN
7440 REM ------------------------------------------------
7450 LPRINT USING Q$;S$:PRINT TAB(8);W$:FOR L=SDD TO SND+1 STEP -1
7460 LPRINT USING R$;L;SDPL(L);SDRT(L,0);SDRT(L,1):NEXT L
7470 FOR L=SND TO 1 STEP -1
7480 LPRINT USING T$;L;SNPL(L);SDPL(L);SNRT(L,0);SNRT(L,1);SDRT(L,0);SDRT(L,1)
7490 NEXT L:LPRINT USING Y$;0;SNPL(0);SDPL(0);SGA:PRINT:RETURN
```

C .. Discretization Program

20-41	To get **G(s)** and **H(s)**.
50-65	To input from the user (**T**, **m** , etc.).
100-160	Define arrays.
1000-1250	Calculation of $G(s).G_H(s)$.
1260	Calls **z**-transformation subroutine.
1290-1340	To save **G(z)**.
2000-2340	Same calculations for **H(s)** $G(s).G_H(s)$.
3000-3340	Same calculations for $G(s).G_H(s)$, but this time for **T'** and **m'**.
4000-4160	To find **P** (explained in Ch. 2).
4170-4280	To find **QA**.
4290-4530	To find **A** and **b**.
4540-4720	To solve for **x=(1/A).b** .
4740-4930	To find the samples **g(i.T)**.
4940-5080	Calculation of **N(z)**.
5500-	To display and geta printout of **G(z)**, **H(z)** and **G'(z)**.

```
20 REM__________GET  THE  DATA : G(s) and H(s)_____________________________
21 SCREEN 0,0:KEY OFF:OPEN "I",#1,"DATA"
22 INPUT #1,GDD,GND,HDD,HND:DIM GDEN(GDD),GNUM(GND),HDEN(HDD),HNUM(HND)
23 GHADD=HDD+GDD:DIM GDRO(GDD,1),GNRO(GDD,1),HDRO(GHADD,1),HNRO(GHADD,1)
24 FOR L1=0 TO GDD:INPUT #1,GDEN(L1):NEXT L1:IF GDD = 0 THEN GOTO 26
25 FOR L1=1 TO GDD:INPUT #1,GDRO(L1,0),GDRO(L1,1):NEXT L1
26 FOR L1=0 TO GND:INPUT #1,GNUM(L1):NEXT L1:IF GND = 0 THEN GOTO 28
27 FOR L1=1 TO GND:INPUT #1,GNRO(L1,0),GNRO(L1,1):NEXT L1
28 FOR L1=0 TO HDD:INPUT #1,HDEN(L1):NEXT L1:IF HDD = 0 THEN GOTO 30
29 FOR L1=1 TO HDD:INPUT #1,HDRO(L1,0),HDRO(L1,1):NEXT L1
30 FOR L1=0 TO HND:INPUT #1,HNUM(L1):NEXT L1:IF HND = 0 THEN GOTO 32
31 FOR L1=1 TO HND:INPUT #1,HNRO(L1,0),HNRO(L1,1):NEXT L1
32 CLOSE #1:OPEN "O",#1,"DATA"
33 WRITE #1,GDD,GND,HDD,HND
34 FOR L1=0 TO GDD:WRITE #1,GDEN(L1):NEXT L1:IF GDD = 0 THEN GOTO 36
35 FOR L1=1 TO GDD:WRITE #1,GDRO(L1,0),GDRO(L1,1):NEXT L1
36 FOR L1=0 TO GND:WRITE #1,GNUM(L1):NEXT L1:IF GND = 0 THEN GOTO 38
37 FOR L1=1 TO GND:WRITE #1,GNRO(L1,0),GNRO(L1,1):NEXT L1
38 FOR L1=0 TO HDD:WRITE #1,HDEN(L1):NEXT L1:IF HDD = 0 THEN GOTO 40
39 FOR L1=1 TO HDD:WRITE #1,HDRO(L1,0),HDRO(L1,1):NEXT L1
40 FOR L1=0 TO HND:WRITE #1,HNUM(L1):NEXT L1:IF HND = 0 THEN GOTO 50
41 FOR L1=1 TO HND:WRITE #1,HNRO(L1,0),HNRO(L1,1):NEXT L1
50 REM ------------------------------------------------------------------
51 CLS
52 LOCATE 1,20:INPUT "SAMPLING  PERIOD  T = ";T:IF T=0 THEN GOTO 52
53 LOCATE 3,11:INPUT " # OF POINTS IN BETWEEN EACH SAMPLE  ";BS
54 LOCATE 5,19:PRINT "WHAT IS THE HOLD DEVICE  "
55 LOCATE 6,22:PRINT "(0:ZOH/1:FOH/2:EH)"
56 C$=INKEY$:IF C$<>"0" AND C$<>"1" AND C$<>"2" THEN GOTO 56
57 HOLD=VAL(C$):TC=0:ADD=1
58 LOCATE  7,15:IF HOLD=2 THEN INPUT "  TIME CONSTANT FOR EH  A = ";TC
59 IF HOLD=1 THEN ADD=2
60 M=1:LOCATE 9,23:PRINT "MODIFICATION   (Y) ?    "
61 C$=INKEY$:IF C$="" THEN GOTO 61
62 IF C$<>"Y"  THEN GOTO 65
63 LOCATE 10,24:INPUT "MODY. RATE  (0-1) ";M
64 IF M<0 OR M>1 THEN GOTO 63
65 WRITE #1,HOLD,TC:CLOSE #1
100 REM ----------------------------------------------------------------
110 GHADD=GDD+HDD+ADD
120 DIM SFNRO(GHADD,2),CC(GHADD,GHADD+2),PFFN(GHADD,5+GHADD),NUMRO(GHADD,3)
130 DIM POLY(9*GDD+GHADD),F(10),G(GHADD),FSNRO(GHADD,3),DEN(GHADD),NUM(GHADD)
140 DIM ZDEN(GHADD),ZN(GHADD),ZNUM(GHADD),ROOTS(9*GDD+GHADD,1)
145 DIM SNPL(GHADD+9),SDPL(GHADD+9),SNRT(GHADD+9,1),SDRT(GHADD+9,1)
150 F(0)=1:FOR L1=1 TO 10:F(L1)=L1*F(L1-1):NEXT L1
160 OPEN "O",#2,"ZTR1"
1000 REM ---------------------------------------------------------------
1005 LST=0:PRINT :PRINT :PRINT ,"       ..  PLEASE  WAIT  .."
1010 FOR L1=0 TO GDD:DEN(L1)=GDEN(L1):NEXT L1
```

```
1020 FOR L1=0 TO GND:NUM(L1)=GNUM(L1):NEXT L1
1030 IF GDD>0 THEN GOTO 1050
1040 WRITE #2,0,GNUM(0),0:GOTO 2000
1050 DD=GDD:ND=DD-1:FOR L1=1 TO GDD
1060 NUMRO(L1,0)=1:NUMRO(L1,1)=GDRO(L1,0):NUMRO(L1,2)=GDRO(L1,1):NEXT L1
1080 IF HOLD<>1 THEN GOTO 1160
1090 FOR L1=1 TO GDD
1100 IF NUMRO(L1,2)<>0 OR NUMRO(L1,1)<>-1/T THEN NEXT L1:GOTO 1160
1110 DD=DD-1:FOR L2=L1 TO ZDD
1120 SWAP NUMRO(L2,1),NUMRO(L2+1,1):SWAP NUMRO(L2,2),NUMRO(L2+1,2):NEXT L2
1130 SPARE1=0:FOR L2=0 TO DD
1140 DEN(DD-L2)=DEN(DD-L2)-SPARE1/T:SPARE1=DEN(DD-L2):NEXT L2
1150 FOR L1=0 TO DD-1:SWAP DEN(L1+1),DEN(L1):NEXT L1
1160 FOR L1=0 TO GDD+ADD:ZDEN(L1)=0:NEXT L1:ZDEN(0)=1:L1=0
1170 L1=L1+1:IF L1>DD THEN GOTO 1245
1180 IF NUMRO(L1,2)<>0 THEN GOTO 1210
1190 B0=EXP(T*NUMRO(L1,1)):SPARE1=0:FOR L2=0 TO L1
1200 SPARE2=ZDEN(L2):ZDEN(L2)=SPARE1-B0*SPARE2:SPARE1=SPARE2:NEXT L2:GOTO 1170
1210 B00=EXP(NUMRO(L1,1)*T):B0=B00^2:B1=-2*B00*COS(NUMRO(L1,2)*T)
1220 L1=L1+1:SPARE1=0:SPARE2=0:FOR L2=0 TO L1
1230 SPARE3=ZDEN(L2):ZDEN(L2)=SPARE1+SPARE2*B1+SPARE3*B0
1240 SPARE1=SPARE2:SPARE2=SPARE3:NEXT L2:GOTO 1170
1245 ZDD=DD+HOLD-2*INT(HOLD/2)+1-INT(M)
1247 ZND=ZDD-1-INT(1-M)
1250 FOR L1=0 TO DD:SWAP ZDEN(ZDD-L1),ZDEN(DD-L1):NEXT L1
1260 GOSUB 4000
1290 WRITE #2,ZND:FOR L1=0 TO ZND:WRITE #2,ZN(L1):NEXT L1
1300 FOR L1=0 TO ZDD/2:SWAP ZDEN(L1),ZDEN(ZDD-L1):NEXT L1
1320 WRITE #2,ZDD:FOR L1=1 TO DD:Q=EXP(T*NUMRO(L1,1)):W=T*NUMRO(L1,2)
1330 WRITE #2,Q*COS(W),Q*SIN(W):NEXT L1
1340 FOR L1=DD+1 TO ZDD:WRITE #2,0,0:NEXT L1
2000 REM ------------------------------------------------------------------------
2005 PRINT :PRINT ,"       .. PLEASE WAIT .."
2010 FOR L1=0 TO GDD+ADD:DEN(L1)=0:NUM(L1)=0:NEXT L1
2020 FOR L1=0 TO GDD:FOR L2=0 TO HDD
2030 DEN(L1+L2)=HDEN(L2)*GDEN(L1)+DEN(L1+L2):NEXT L2:NEXT L1
2040 FOR L1=0 TO GND:FOR L2=0 TO HND
2050 NUM(L1+L2)=HNUM(L2)*GNUM(L1)+NUM(L1+L2):NEXT L2:NEXT L1
2060 IF HDD+GDD>0 THEN GOTO 2080
2070 WRITE #2,0,GNUM(0)*HNUM(0),0:GOTO 3000
2080 DD=GDD+HDD:ND=DD-1:FOR L1=1 TO GDD
2090 NUMRO(L1,0)=1:NUMRO(L1,1)=GDRO(L1,0):NUMRO(L1,2)=GDRO(L1,1):NEXT L1
2100 FOR L1=1 TO HDD:L2=L1+GDD
2110 NUMRO(L2,0)=1:NUMRO(L2,1)=HDRO(L1,0):NUMRO(L2,2)=HDRO(L1,1):NEXT L1
2130 IF HOLD<>1 THEN GOTO 2210
2140 FOR L1=1 TO DD
2150 IF NUMRO(L1,2)<>0 OR NUMRO(L1,1)<>-1/T THEN NEXT L1:GOTO 2210
2160 DD=DD-1:FOR L2=L1 TO DD
```

```
2170 SWAP NUMRO(L2,1),NUMRO(L2+1,1):SWAP NUMRO(L2,2),NUMRO(L2+1,2):NEXT L2
2180 SPARE1=0:FOR L2=0 TO DD
2190 DEN(DD-L2)=DEN(DD-L2)-SPARE1/T:SPARE1=DEN(DD-L2):NEXT L2
2200 FOR L1=0 TO DD-1:SWAP DEN(L1+1),DEN(L1):NEXT L1
2210 FOR L1=0 TO GDD+HDD+ADD:ZDEN(L1)=0:NEXT L1:ZDEN(0)=1:L1=0
2220 L1=L1+1:IF L1>DD THEN GOTO 2295
2230 IF NUMRO(L1,2)<>0 THEN GOTO 2260
2240 BO=EXP(T*NUMRO(L1,1)):SPARE1=0:FOR L2=0 TO L1
2250 SPARE2=ZDEN(L2):ZDEN(L2)=SPARE1-BO*SPARE2:SPARE1=SPARE2:NEXT L2:GOTO 2220
2260 BOO=EXP(NUMRO(L1,1)*T):BO=BOO^2:BI=-2*BOO*COS(NUMRO(L1,2)*T)
2270 L1=L1+1:SPARE1=0:SPARE2=0:FOR L2=0 TO L1
2280 SPARE3=ZDEN(L2):ZDEN(L2)=SPARE1+SPARE2*BI+SPARE3*BO
2290 SPARE1=SPARE2:SPARE2=SPARE3:NEXT L2:GOTO 2220
2295 ZDD=DD+HOLD-2*INT(HOLD/2)+1-INT(M)
2297 ZND=ZDD-1-INT(1-M)
2300 FOR L1=0 TO DD:SWAP ZDEN(ZDD-L1),ZDEN(DD-L1):NEXT L1
2310 GOSUB 4000
2340 WRITE #2,ZND:FOR L1=0 TO ZND:WRITE #2,ZN(L1):NEXT L1
2350 FOR L1=0 TO ZDD/2:SWAP ZDEN(L1),ZDEN(ZDD-L1):NEXT L1
2361 IF HOLD<>1 OR HDD=0 THEN GOTO 2440
2363 FOR L1=1 TO HDD
2365 NUMRO(L1,0)=1:NUMRO(L1,1)=HDRO(L1,0):NUMRO(L1,2)=HDRO(L1,1):NEXT L1
2367 FOR L1=1 TO DD
2369 IF NUMRO(L1,2)<>0 OR NUMRO(L1,1)<>-1/T THEN NEXT L1:GOTO 2440
2370 HDD=HDD-1:FOR L2=L1 TO HDD
2373 SWAP NUMRO(L2,1),NUMRO(L2+1,1):SWAP NUMRO(L2,2),NUMRO(L2+1,2):NEXT L2
2440 WRITE #2,HDD:FOR L1=1 TO HDD:O=EXP(T*NUMRO(L1,1)):W=T*NUMRO(L1,2)
2450 WRITE #2,O*COS(W),O*SIN(W):NEXT L1
3000 REM ------------------------------------------------------------------
3005 LST=1:PRINT :PRINT ,"       .. PLEASE  WAIT  .."
3010 T1=T:T=T1/(BS+1):MJ=M:DL=T1*(1-MJ):SH=INT(DL/T):M=1+SH-DL/T
3020 IF ABS(1-M)<.00001 THEN M=1
3030 FOR L1=0 TO GDD+ADD:DEN(L1)=0:NUM(L1)=0:NEXT L1
3040 FOR L1=0 TO GDD:DEN(L1)=GDEN(L1):NEXT L1
3050 FOR L1=0 TO GND:NUM(L1)=GNUM(L1):NEXT L1
3060 IF GDD>0 THEN GOTO 3080
3070 WRITE #2,0,GNUM(0),0,1:GOTO 3380
3080 DD=GDD:ND=DD-1:FOR L1=1 TO GDD
3090 NUMRO(L1,0)=1:NUMRO(L1,1)=GDRO(L1,0):NUMRO(L1,2)=GDRO(L1,1):NEXT L1
3110 IF HOLD<>1 THEN GOTO 3190
3120 FOR L1=1 TO GDD
3130 IF NUMRO(L1,2)<>0 OR NUMRO(L1,1)<>-1/T THEN NEXT L1:GOTO 3190
3140 DD=DD-1:FOR L2=L1 TO ZDD
3150 SWAP NUMRO(L2,1),NUMRO(L2+1,1):SWAP NUMRO(L2,2),NUMRO(L2+1,2):NEXT L2
3160 SPARE1=0:FOR L2=0 TO DD
3170 DEN(DD-L2)=DEN(DD-L2)-SPARE1/T:SPARE1=DEN(DD-L2):NEXT L2
3180 FOR L1=0 TO DD-1:SWAP DEN(L1+1),DEN(L1):NEXT L1
3190 FOR L1=0 TO GDD+ADD:ZDEN(L1)=0:NEXT L1:ZDEN(0)=1:L1=0
```

```
3200 L1=L1+1:IF L1>DD THEN GOTO 3275
3210 IF NUMRO(L1,2)<>0 THEN GOTO 3240
3220 BO=EXP(T*NUMRO(L1,1)):SP1=0:FOR L2=0 TO L1
3230 SP2=ZDEN(L2):ZDEN(L2)=SP1-BO*SP2:SP1=SP2:NEXT L2:GOTO 3200
3240 BOO=EXP(NUMRO(L1,1)*T):BO=BOO^2:B1=-2*BOO*COS(NUMRO(L1,2)*T)
3250 L1=L1+1:SPARE1=0:SPARE2=0:FOR L2=0 TO L1
3260 SPARE3=ZDEN(L2):ZDEN(L2)=SPARE1+SPARE2*B1+SPARE3*BO
3270 SPARE1=SPARE2:SPARE2=SPARE3:NEXT L2:GOTO 3200
3275 ZDD=DD+HOLD-2*INT(HOLD/2)+1-INT(M)
3277 ZND=ZDD-1-INT(1-M)
3280 FOR L1=0 TO GDD-WAR:SWAP ZDEN(ZDD-L1),ZDEN(GDD-WAR-L1):NEXT L1
3290 GOSUB 4000
3320 WRITE #2,ZND:FOR L1=0 TO ZND:WRITE #2,ZN(L1):NEXT L1
3330 FOR L1=0 TO ZDD/2:SWAP ZDEN(L1),ZDEN(ZDD-L1):NEXT L1
3360 WRITE #2,ZDD+SH:FOR L1=0 TO SH-1:WRITE #2,0:NEXT L1
3370 FOR L1=0 TO ZDD:WRITE #2,ZDEN(L1):NEXT L1
3380 WRITE #2,BS+1,HOLD,TI,MI,EXP(TC*TI):CLOSE #2
3390 GOTO 5500
4000 REM ------------------------------------------------------------------------
4010 DDD=DD+ADD:FOR L1=1 TO DD
4020 SPNRO(L1,0)=1:SPNRO(L1,1)=NUMRO(L1,1):SPNRO(L1,2)=NUMRO(L1,2):NEXT L1
4030 FOR L1=DD+1 TO DD+ADD:SPNRO(L1,0)=1:SPNRO(L1,1)=0:SPNRO(L1,2)=0:NEXT L1
4040 SPNRO(DDD,1)=TC:SPNRO(DDD,2)=0
4050 REM ------------------------------------------------------------------------
4060 IF DDD<>1 THEN GOTO 4070
4070 FDD=1:FSNRO(1,0)=1:FSNRO(1,3)=1:FSNRO(1,1)=SPNRO(1,0)
4080 GOTO 4260
4090 FOR L1=1 TO DDD-1
4100 FOR L2=L1+1 TO DDD
4110 D20=SPNRO(L2,0):D11=SPNRO(L1,1):D21=SPNRO(L2,1)
4120 D12=ABS(SPNRO(L1,2)):D22=ABS(SPNRO(L2,2)):D10=SPNRO(L1,0)
4130 IF D20<>0 AND D11=D21 AND D12=D22 THEN SPNRO(L1,0)=D10+1:SPNRO(L2,0)=0
4140 ELSE
4150 NEXT L2
4160 NEXT L1
4170 REM ------------------------------------------------------------------------
4180 FOR L=1 TO DDD:IF SPNRO(L,2)<>0 THEN SPNRO(L,0)=SPNRO(L,0)/2:NEXT L
4190 SUM=0:FOR L1=1 TO DDD:SUM=SUM+SGN(SPNRO(L1,0)):NEXT L1
4200 FDD=SUM:I=0:FOR L1=1 TO FDD
4210 I=I+1:IF SPNRO(I,0)=0 THEN GOTO 4210
4220 FSNRO(L1,0)=SPNRO(I,0):FSNRO(L1,1)=SPNRO(I,1)
4230 FSNRO(L1,2)=SPNRO(I,2):FSNRO(L1,3)=SPNRO(I,0)
4240 NEXT L1
4250 REM ------------------------------------------------------------------------
4260 FOR L1=0 TO DD:SWAP DEN(DDD-L1),DEN(DD-L1):NEXT L1
4270 IF HOLD<>2 THEN GOTO 4290
4280 FOR L1=0 TO DD:DEN(L1)=DEN(L1)-TC*DEN(L1+1):NEXT L1
4290 REM ------------------------------------------------------------------------
```

```
4300 ROW=0:FOR L1=0 TO DDD:FOR L2=0 TO DDD+2:CC(L1,L2)=0:NEXT L2:NEXT L1
4310 FOR L1=1 TO FDD
4320 FOR L2=0 TO DDD:POLY(L2)=DEN(L2):NEXT L2
4330 IF FSNRO(L1,2)<>0 THEN GOTO 4390
4340 BO=FSNRO(L1,1):FOR L2=1 TO FSNRO(L1,0)
4350 ROW=ROW+1:SPARE1=0:FOR L3=0 TO DDD-L2
4360 POLY(DDD-L3)=POLY(DDD-L3)+SPARE1*BO
4370 SPARE1=POLY(DDD-L3):CC(L3+L2,ROW)=POLY(DDD-L3):NEXT L3
4380 PFFN(ROW,0)=1:PFFN(ROW,1)=L2:PFFN(ROW,2)=BO:NEXT L2:GOTO 4470
4390 BI=-2*FSNRO(L1,1):BO=(FSNRO(L1,1))^2+(FSNRO(L1,2))^2
4400 FOR L2=1 TO FSNRO(L1,0)
4410 ROW=ROW+1:SPARE2=0:SPARE1=0:FOR L3=0 TO DDD-2*L2
4420 POLY(DDD-L3)=POLY(DDD-L3)-SPARE1*BI-SPARE2*BO:SPARE2=SPARE1
4430 SPARE1=POLY(DDD-L3):CC(L3+2*L2-1,ROW)=POLY(DDD-L3)
4440 CC(L3+2*L2,ROW+1)=POLY(DDD-L3):NEXT L3
4450 PFFN(ROW,0)=2:PFFN(ROW,1)=L2:PFFN(ROW,2)=FSNRO(L1,1)
4460 PFFN(ROW,3)=FSNRO(L1,2):NEXT L2:ROW=ROW+1
4470 NEXT L1
4480 REM ------------------------------------------------------------------------
4490 FOR L1=0 TO ND:CC(DDD-L1,DDD+1)=NUM(L1):NEXT L1
4500 IF HOLD<>1 OR WAR=1 THEN GOTO 4550
4510 BO=1/T:IF LST=1 THEN BO=BO/(BS+1)
4515 SPARE1=0:FOR L2=0 TO ND+1
4520 SPARE2=CC(DDD-L2,DDD+1):CC(DDD-L2,DDD+1)=SPARE1+SPARE2*BO
4530 SPARE1=SPARE2:NEXT L2
4540 REM ------------------------------------------------------------------------
4550 FOR L2=1 TO DDD:CC(L2,DDD+2)=L2:NEXT L2
4560 FOR L2=1 TO DDD
4570 RWO=L2:LIEN=L2:FOR L3=L2 TO DDD
4580 IF ABS(CC(L3,L2))>ABS(CC(RWO,L2)) THEN RWO=L3
4590 IF ABS(CC(L2,L3))>ABS(CC(L2,LIEN)) THEN LIEN=L3:NEXT L3
4600 IF RWO=L2 AND LIEN=L2 THEN GOTO 4650
4610 IF ABS(CC(L2,LIEN))>ABS(CC(RWO,L2)) THEN GOTO 4630
4620 FOR L3=1 TO DDD+1:SWAP CC(L2,L3),CC(RWO,L3):NEXT L3:GOTO 4650
4630 FOR L3=1 TO DDD:SWAP CC(L3,L2),CC(L3,LIEN):NEXT L3
4640 SWAP CC(L2,DDD+2),CC(LIEN,DDD+2)
4650 FOR L3=1 TO DDD
4660 IF L3=L2 THEN GOTO 4690
4670 CONST=CC(L3,L2)/CC(L2,L2):FOR L4=1 TO DDD+1
4680 CC(L3,L4)=CC(L3,L4)-CONST*CC(L2,L4):NEXT L4
4690 NEXT L3:NEXT L2
4700 REM ------------------------------------------------------------------------
4710 FOR L2=1 TO DDD:PFFN(CC(L2,DDD+2),4)=CC(L2,DDD+1)/CC(L2,L2):NEXT L2
4720 FOR L1=1 TO FDD:FSNRO(L1,3)=FSNRO(L1,0):NEXT L1
4730 REM ------------------------------------------------------------------------
4740 ROW=0
4750 ROW=ROW+1:IF ROW>DDD THEN GOTO 4950
4760 IF PFFN(ROW,0)=2 THEN GOTO 4830
```

```
4770 PFFN(ROW,5)=0:C1=PFFN(ROW,1)-1:C2=PFFN(ROW,4)
4780 IF M<>1 THEN GOTO 4800
4790 PFFN(ROW,5)=0^C1/F(C1)*C2
4800 FOR L2=1 TO DDD:KT=(L2-1+M)*T
4810 PFFN(ROW,5+L2)=KT^C1/F(C1)*C2*EXP(PFFN(ROW,2)*KT):NEXT L2
4820 GOTO 4750
4830 ROW=ROW+1:FOR L1=1 TO DDD:PFFN(ROW,5+L1)=0:PFFN(ROW-1,5+L1)=0:NEXT L1
4840 C1=PFFN(ROW-1,1):C2=PFFN(ROW-1,2):C3=PFFN(ROW-1,3)
4850 C5=PFFN(ROW-1,4):C6=PFFN(ROW,4):CC=(C6+C2*C5)/C3
4860 IF M<>1 THEN GOTO 4880
4870 PFFN(ROW,5)=C5
4880 FOR L2=1 TO DDD:KT=(L2-1+M)*T
4890 PFFN(ROW,5+L2)=EXP(C2*KT)*(C5*COS(C3*KT)+CC*SIN(C3*KT))
4900 IF C1<>2 THEN GOTO 4920
4910 PFFN(ROW,5+L2)=EXP(C2*KT)*
       ((C5*KT/2/C3)*SIN(C3*KT)-CC*(SIN(C3*KT)-C3*KT*COS(C3*KT))/C3^2/2)
4920 NEXT L2
4930 GOTO 4750
4940 REM ------------------------------------------------------------------------
4950 FOR L1=0 TO DDD:SUM=0:FOR L2=1 TO DDD
4960 SUM=SUM+PFFN(L2,5+L1):NEXT L2
4970 G(L1)=SUM:NEXT L1
4980 REM FOR L1=0 TO DDD:PRINT " G(";L1;")=";G(L1):NEXT L1
4990 SPARE2=0:FOR L2=0 TO DDD:SPARE1=G(L2)
5000 G(L2)=SPARE1-SPARE2*(EXP(TC*T)):SPARE2=SPARE1:NEXT L2
5010 IF HOLD<>1 THEN GOTO 5040
5020 SPARE2=0:FOR L2=0 TO DDD:SPARE1=G(L2)
5030 G(L2)=SPARE1-SPARE2:SPARE2=SPARE1:NEXT L2
5040 FOR L1=0 TO ZDD/2:SWAP ZDEN(L1),ZDEN(ZDD-L1):NEXT L1
5050 FOR L1=0 TO DDD:ZN(L1)=0:FOR L2=0 TO L1
5060 ZN(L1)=ZN(L1)+ZDEN(L2)*G(L1-L2):NEXT L2:NEXT L1
5070 FOR L1=0 TO ZDD/2:SWAP ZN(L1),ZN(ZDD-L1):NEXT L1
5080 RETURN
5500 REM ------------------
5505 OPEN "I",#2,"ZIR1"
5510 INPUT #2,GND:DIM GNU(GND):FOR L1=0 TO GND:INPUT #2,GNU(L1):NEXT L1
5520 DIM GNR(GND,2):INPUT #2,GDD:DIM GDR(GDD,2):DIM GDE(GDD)
5530 FOR L1=1 TO GDD:INPUT #2,GDR(L1,0),GDR(L1,1):NEXT L1
5540 INPUT #2,HND:DIM HNU(HND):FOR L1=0 TO HND:INPUT #2,HNU(L1):NEXT L1
5550 DIM HNR(HND,2):INPUT #2,HDD:DIM HDR(HDD,2):DIM HDE(HDD)
5560 FOR L1=1 TO HDD:INPUT #2,HDR(L1,0),HDR(L1,1):NEXT L1
5570 INPUT #2,VND:DIM VNU(VND):FOR L1=0 TO VND:INPUT #2,VNU(L1):NEXT L1
5580 INPUT #2,VDD:DIM VDE(VDD):FOR L1=0 TO VDD:INPUT #2,VDE(L1):NEXT L1
5590 INPUT #2,N,M,HOLD,T,N,AA:CLOSE #2:D=GDD+HDD
5592 IF HOLD=0 THEN DD$="ZOH"
5594 IF HOLD=1 THEN DD$="FOH"
5596 IF HOLD=2 THEN DD$=" EH"
5598 CLS:PRINT ,"T =";T,"m =";M,DD$;"  DEVICE"
```

```
6000 REM ------------------------------------------------------------------
6030 IF PO=1 THEN LPRINT ,"T =";T,"m =";M,DD$;"  DEVICE"
6040 GGN=GNU(GND):DEGR=GND:FOR L=0 TO GND:POLY(L)=GNU(L)/GGN:NEXT L:GOSUB 8000
6050 FOR L=1 TO GND:GNR(L,0)=ROOTS(L,0):GNR(L,1)=ROOTS(L,1):NEXT L:SND=GND
6060 FOR L=0 TO SND:SNPL(L)=GNU(L)/GGN:SNRT(L,0)=GNR(L,0):SNRT(L,1)=GNR(L,1)
6070 NEXT L:SGA=GGN:DEGR=GDD:FOR L=1 TO GDD:ROOTS(L,0)=GDR(L,0)
6080 ROOTS(L,1)=GDR(L,1):NEXT L:GOSUB 9000:SDD=GDD:FOR L=0 TO SDD
6090 GDE(L)=POLY(L):NEXT L:FOR L=0 TO SDD:SDPL(L)=GDE(L):SDRT(L,0)=GDR(L,0)
6100 SDRT(L,1)=GDR(L,1):NEXT L:S$="G(z)":XX=2
6103 IF PO<>1 THEN GOSUB 6500 ELSE GOSUB 7000
6105 REM ---------------
6110 HGN=HNU(HND):DEGR=HND:FOR L=0 TO HND:POLY(L)=HNU(L)/HGN:NEXT L:GOSUB 8000
6120 FOR L=1 TO HND:HNR(L,0)=ROOTS(L,0):HNR(L,1)=ROOTS(L,1):NEXT L:SND=HND
6130 FOR L=0 TO SND:SNPL(L)=HNU(L)/HGN:SNRT(L,0)=HNR(L,0):SNRT(L,0)=HNR(L,0)
6140 NEXT L:SGA=HGN:DEGR=HDD+GDD:FOR L=1 TO HDD:ROOTS(L,0)=HDR(L,0)
6150 ROOTS(L,1)=HDR(L,1):NEXT L:FOR L=1 TO GDD:ROOTS(HDD+L,0)=GDR(L,0)
6160 ROOTS(HDD+L,1)=GDR(L,1):NEXT L:GOSUB 9000:SDD=HDD+GDD:FOR L=0 TO SDD
6170 SDPL(L)=POLY(L):NEXT L:FOR L=1 TO SDD:SDRT(L,0)=ROOTS(L,0)
6180 SDRT(L,1)=ROOTS(L,1):NEXT L:S$="H(z)":XX=GDD+5
6190 IF PO<>1 THEN GOSUB 6500 ELSE GOSUB 7000
6200 REM ---------------
6210 VGN=VNU(VND):DEGR=VND:FOR L=0 TO VND:POLY(L)=VNU(L)/VGN:NEXT L:GOSUB 8000
6220 FOR L=1 TO VND:VNR(L,0)=ROOTS(L,0):VNR(L,1)=ROOTS(L,1):NEXT L:SND=VND
6230 FOR L=0 TO SND:SNPL(L)=VNU(L)/VGN:SNRT(L,0)=VNR(L,0):SNRT(L,0)=VNR(L,0)
6250 NEXT L:SGA=VGN:DEGR=VDD:FOR L=0 TO VDD:POLY(L)=VDE(L):NEXT L:GOSUB 8000
6260 FOR L=1 TO VDD:VDR(L,0)=ROOTS(L,0):VDR(L,1)=ROOTS(L,1):NEXT L:SDD=VDD
6270 FOR L=0 TO SDD:SDPL(L)=VDE(L):SDRT(L,0)=VDR(L,0):SDRT(L,1)=VDR(L,1):NEXT L
6280 S$="G'(z)":XX=2*GDD+HDD+8
6281 IF PO<>1 THEN GOSUB 6500 ELSE GOSUB 7000
6285 PRINT ,,".. PRINT OUT (Y) .."
6286 C$=INKEY$:IF C$="" THEN GOTO 6286
6287 IF C$="Y" THEN PO=1:GOTO 6000
6290 RUN"MENU.BAS
6500 REM ------------------------------------------------------------
6510 0$=                                  *  ----------------------------
     &     ----------------------------  *
6520 W$="NUM.        DEN.         NUMERATOR ROOT(S)       DENOMINATOR ROOT(S) *
6530 E$=                    "##:           ##.####^^^^ *
   * ##.####^^^^ ##.####^^^^"
6540 R$=                    "##:           ##.####^^^^ *
   * ##.####^^^^ ##.####^^^^"
6550 T$=                    "##:##.####^^^^ ##.####^^^^ * ##.####^^^^ ##.####^^^
^  * ##.####^^^^ ##.####^^^^"
6560 J$=                    "##:##.####^^^^ ##.####^^^^ *
   *                        *
6570 Y$="##:##.####^^^^ ##.####^^^^ *       GAIN= ##.####^^^^   *
6580 U$="##:##.####^^^^ ##.####^^^^ *                                  *
6590 LOCATE XX,1:PRINT USING 0$;S$:PRINT TAB(6);W$
```

```
6600 IF SND<>SDD OR SND<>0 THEN GOTO 6620
6610 PRINT USING J$;0;SGA;1:RETURN
6620 FOR L=SND+1 TO SDD:PRINT USING E$;L;SDPL(L);SDRT(L,0);SDRT(L,1):NEXT L
6630 IF SDD-SND<>2 THEN GOTO 6650
6640 PRINT USING R$;SDD-1;SDPL(SDD-1);SDRT(SDD-1,0);SDRT(SDD-1,1)
6650 FOR L=SND TO 1 STEP -1
6660 PRINT USING T$;L;SNPL(L);SDPL(L);SNRT(L,0);SNRT(L,1);SDRT(L,0);SDRT(L,1)
6670 NEXT L:PRINT USING Y$;0;SNPL(0);SDPL(0);SGA:PRINT:RETURN
7000 REM ------------------------------------------------
7010 LPRINT USING Q$;S$:LPRINT TAB(8);K$
7020 IF SND<>SDD OR SND<>0 THEN GOTO 7040
7030 LPRINT USING J$;0;SGA;1:RETURN
7040 FOR L=SND+1 TO SDD:LPRINT USING E$;L;SDPL(L);SDRT(L,0);SDRT(L,1):NEXT L
7050 IF SDD-SND<>2 THEN GOTO 7070
7060 LPRINT USING R$;SDD-1;SDPL(SDD-1);SDRT(SDD-1,0);SDRT(SDD-1,1)
7070 FOR L=SND TO 1 STEP -1
7080 LPRINT USING T$;L;SNPL(L);SDPL(L);SNRT(L,0);SNRT(L,1);SDRT(L,0);SDRT(L,1)
7090 NEXT L:LPRINT USING Y$;0;SNPL(0);SDPL(0);SGA:LPRINT:RETURN
8000 REM   ROOTS  OF  THE  POLY    '*****************************************
8010 REM DIM ROOTS(N),QV(N),QA(N),QB(N),QC(N)
8020 N=DEGR:K=0
8030  IF POLY(K)<>0 THEN GOTO 8050
8040 K=K+1:GOTO 8030
8050  N=N-K
8060 FOR L1=0 TO DEGR:ROOTS(L1,0)=0:ROOTS(L1,1)=0:NEXT L1
8070 FOR L1=0 TO N:QV(L1)=POLY(L1+K):NEXT L1
8080  IF N<4 THEN GOTO  8230
8090 IT=0:HO=((N-1)*QV(N-1))^2-2*N*(N-1)*QV(N)*QV(N-2):IF HO<0 THEN GOTO 8130
8100 DET=QV(N-1)+SQR(HO):IF DET=0 THEN DET=.00001
8110 PO1=N*QV(N)/DET:DET=QV(N-1)-SQR(HO):IF DET=0 THEN DET=.00001
8120 PO2=N*QV(N)/DET:R=-(PO1+PO2):S=-PO1*PO2:GOTO 8160
8130  DET=(QV(N-1))^2+ABS(HO)
8140 IF DET=0 THEN DET=.00001
8150 MG=N*QV(N)/DET:R=-2*MG*QV(N-1):S=-N*QV(N)*MG
8160  SPARE1=0:SPARE2=0:SPARE3=0:SPARE4=0:FOR L1=0 TO N
8170 QB(N-L1)=QV(N-L1)+R*SPARE1+S*SPARE2:SPARE2=SPARE1:SPARE1=QB(N-L1)
8180 QC(N-L1)=QB(N-L1)+R*SPARE3+S*SPARE4:SPARE4=SPARE3:SPARE3=QC(N-L1):NEXT L1
8190 DET=QC(1)*QC(3)-QC(2)^2:IF DET=0 THEN DET=.000001
8200 DR=(QB(1)*QC(2)-QB(0)*QC(3))/DET:DS=(QB(0)*QC(2)-QB(1)*QC(1))/DET
8205 IT=IT+1:IF IT>60 THEN GOTO 9220
8210 R=R+DR:S=S+DS:IF ABS(DR)>7E-08 OR ABS(DS)>7E-08 THEN GOTO 8160
8220 AA=1:FOR L1=0 TO N-2:QV(L1)=QB(L1+2):NEXT L1:GOTO 8400
8230  REM   3'rd   DEGREE  ROOTS
8240 IF N<3 THEN GOTO 8370
8250 AA=QV(2)/QV(3):B=QV(1)/QV(3):CC=QV(0)/QV(3):P=B-(AA^2)/3
8260 Q=CC-AA*B/3+(2*(AA^3))/27:US=(P/3)^3+(Q/2)^2:IF US<0 THEN GOTO 8310
8270 U=SQR(US):VV=U-Q/2:V=SGN(VV)*(ABS(VV))^(1#/3):WWW=-U-Q/2
8280 W=SGN(WWW)*(ABS(WWW))^(1#/3):ROOTS(N,0)=V+W-AA/3:REAL=-(V+W)/2-AA/3
```

```
8290 IMAG=SQR(3)/2*(V-W):ROOTS(N-1,0)=REAL:ROOTS(N-2,0)=REAL
8300 ROOTS(N-1,1)=IMAG:ROOTS(N-2,1)=-IMAG:GOTO 8350
8310  U=(-((P/3)^3))^(1/2):TE=-Q/2/U:PI=4*ATN(1)
8320 TE=PI/2-ATN(TE/SQR(1-TE*TE+1E-10)):KC=2*U^(1#/3)
8330 ROOTS(N,0)=KC*COS(TE/3)-AA/3:ROOTS(N-1,0)=KC*COS(TE/3-2*PI/3)-AA/3
8340 ROOTS(N-2,0)=KC*COS(TE/3+2*PI/3)-AA/3
8350  N=N-3
8360 GOTO 8500
8370  REM   2'nd  DEGREE   ROOTS
8380 IF N<2 THEN GOTO 8480
8390 AA=QV(2):R=-QV(1):S=-QV(0)
8400 B=-R:C=-S:DELTA=R^2-4*AA*C
8410 IF DELTA<0 THEN GOTO 8440
8420 ROOTS(N,0)=-(B+SQR(DELTA))/2/AA:ROOTS(N-1,0)=-(B-SQR(DELTA))/2/AA
8430 GOTO 8460
8440  REAL=-B/2/AA:IMAG=SQR(ABS(DELTA))/2/AA:ROOTS(N,0)=REAL
8450 ROOTS(N-1,0)=REAL:ROOTS(N,1)=IMAG:ROOTS(N-1,1)=-IMAG
8460  N=N-2
8470 IF N>1 THEN GOTO 8080
8480  IF N<1 THEN GOTO 8500
8490 ROOTS(N,0)=-QV(0)/QV(1)
8500  RETURN
9000 REM-------------------------------------------------------------------
9010 FOR L=0 TO DEGR:POLY(L)=0:NEXT L:POLY(0)=1:L=0
9020 L=L+1
9030 IF L>DEGR THEN RETURN
9040 IF ROOTS(L,1)<>0 THEN GOTO 9120
9050 SPARE2=0:FOR E=0 TO L
9060 SPARE1=FOLY(E)
9070 POLY(E)=SPARE2-ROOTS(L,0)*SPARE1
9080 SPARE2=SPARE1:NEXT E
9090 GOTO 9020
9100 B=-2*ROOTS(L,0)
9110 C=ROOTS(L,0)^2+ROOTS(L,1)^2
9120 SPARE2=0:SPARE3=0
9130 L=L+1
9140 FOR E=0 TO L
9150 SPARE1=POLY(E)
9160 POLY(E)=C*SPARE1+B*SPARE2+SPARE3
9170 SPARE3=SPARE2:SPARE2=SPARE1:NEXT E
9180 GOTO 9020
```

D - Synthesis Method

20-40	Definitions and value settings.
40-95	To get the **G(z), H(z)** ang **G'(z).**
100-210	Warning for type-2 systems.
300-440	Array Definition.
450-470	Format Definitions.
500-560	Calculation of **V(z).**
600-640	Since the both numerators saved in polynomial form, calculate their roots
650-760	To include unstable system zeros to **M(z)**
800-920	To add stable system zeros.
950-970	To set the cursor location.
1000-1270	To design options and evaluation of user choice
1300-1470	To redraw the split screen step responses.
1500-1810	To input the command.
1850-1980	Calculation of control variables.
2000-2080	"First guess" of **M(z)**
2100-2210	To input the negligable poles.
2400-2790	To add a dipole (lag compansation subroutine).
2800-3180	Calculation of **C(z)** .
3200-3630	Calculation of step response and other control variables
3800-3920	To calculate K_v .
4000-4400	To draw the frame (**z**-plane).
4450-4540	To input ζ, T_p/T and K_v .
4550-4730	To save the data in the array **Q** .
4750-4930	To save the data in the array **W** .
5000-5070	Formats.
5100-5750	To display **C(z)** and **M(z)**
5800-6440	To get the printout.
7000-7370	Multiplication of two rational function
8000-8520	Root solving subroutine.

```
20 REM *-*-*-*-*-*-*-*-*-
25 CLS:PI2=2*ATN(1):SAM=5:J=1:OS=1:AD$="("+CHR$(25)+")":ADD$="("+CHR$(25)+")"
30 JJ=.001:PL=.97:SCREEN 2,0,0:LINE (0,190)-(639,200),0,BF
35 DEF FNATAN(X,Y)=(1-SGN(ATN(Y/(X+1E-08))))*PI2+ATN(Y/(X+1E-08))
40 REM *-*-*-*-*-*-*-*-*-
45 OPEN "I",#2,"ZTR1"
50 INPUT #2,GND:DIM GNU(GND):FOR L1=0 TO GND:INPUT #2,GNU(L1):NEXT L1
55 DIM GNR(GND,2):INPUT #2,GDD:DIM GDR(GDD,2):DIM GDE(GDD)
60 FOR L1=1 TO GDD:INPUT #2,GDR(L1,0),GDR(L1,1):NEXT L1
65 INPUT #2,HND:DIM HNU(HND):FOR L1=0 TO HND:INPUT #2,HNU(L1):NEXT L1
70 DIM HNR(HND,2):INPUT #2,HDD:DIM HDR(HDD,2):DIM HDE(HDD)
75 FOR L1=1 TO HDD:INPUT #2,HDR(L1,0),HDR(L1,1):NEXT L1
80 INPUT #2,VNG:DIM VNU(VNG):FOR L1=0 TO VNG:INPUT #2,VNU(L1):NEXT L1
85 INPUT #2,VDG:DIM VDE(VDG):FOR L1=0 TO VDG:INPUT #2,VDE(L1):NEXT L1
90 INPUT #2,NNN,HOLD,T,M,TC
95 CLOSE #2
100 REM *-*-*-*-*-*-*-*-*-
110 D=HDD+GDD+3
120 REM *-*-*-*-*-*-*-*-*-
130 FOR L=1 TO GDD:IF GDR(L,0)=1 AND GDR(L,1)=0 THEN ACCT=ACCT+1
140 NEXT L:FOR L=1 TO HDD:IF HDR(L,0)=1 AND HDR(L,1)=0 THEN ACCT=ACCT+1
150 NEXT L:ACT=SGN(INT(ACCT/2)):DKV=0:IF ACCT<2 THEN GOTO 210
160 LOCATE  5,5:PRINT ,"!!       YOU HAVE N TYPE SYSTEM (N>1)      !!"
170 LOCATE  7,5:PRINT ,"!!      PROGRAM ADDS A DIPOLE TO  M(z)     !!"
180 LOCATE  9,5:PRINT ,"!!           TO SATISFY   1/Kv =0          !!"
190 LOCATE 13,5:PRINT ,"..              PRESS ANY KEY              .."
200 C$=INKEY$:IF C$="" THEN GOTO 200
210 LOCATE 15,5:PRINT ,"..              PLEASE WAIT                .."
300 REM *-*-*-*-*-*-*-*-*-
310 DIM POLY(D),        UNR(D,2),       STR(D,2),       ROOTS(D,1)
320 DIM MNU(D),         MDE(D),         MNR(D,1),       MDR(D,1)
330 DIM PNU(D),         PDE(D),         PNR(D,1),       PDR(D,1)
340 DIM CUN(D),         CED(D),         CRN(D,1),       CRD(D,1)
350 DIM TTP(2*D),       YYP(2*D),       TTR(2*D,2),     YYR(2*D,2)
360 DIM UUP(2*D),       IIP(2*D),       UUR(2*D,2),     IIR(2*D,2)
370 DIM QNU(D),         QDE(D),         QNR(D,1),       QDR(D,1)
380 DIM QCNU(D),        QCDE(D),        QCNR(D,1),      QCDR(D,1)
390 DIM WNU(D),         WDE(D),         WNR(D,1),       WDP(D,1)
400 DIM WCNU(D),        WCDE(D),        WCNR(D,1),      WCDR(D,1)
410 DIM A(200),         B(200),         C(200),         D(200)
420 DIM AA(NNN+2),      BB(NNN+2),      S(D),           E(200)
430 DIM Q(200),         W(200)
440 DIM QV(D),          QA(D),          QB(D),          QC(D+2)
450 REM *-*-*-*-*-*-*-*-*-
460 Q1$="###.## ":Q2$="###.### ":MMM=CINT(10/NNN)
470 Q3$="####^^^^ ":SSS=NNN*MMM  :BBB=200/MMM
500 REM *-*-*-*-*-*-*-*-*-
510 AA(1)=-TC:AA(NNN+1)=1:NNN=NNN+1:IF HOLD <>1 THEN GOTO 530
```

```
520 FOR L=0 TO NHN:AA(NHN+1-L)=AA(NHN-L):NEXT L:NHN=NHN+1
530 DHD=NHN:BB(0)=-1:BB(NHN)=1:IF HOLD=1 THEN BB(0)=0
540 SP2=0:FOR L=0 TO NHN+1:SP1=BB(L):BB(L)=SP2-SP1*TC^(1/NHN):SP2=SP1:NEXT L
550 IF HOLD<>1 THEN GOTO 610
560 SP2=0:FOR L=0 TO DHD:SP1=BB(L):BB(L)=SP2-SP1:SP2=SP1:NEXT L
600 REM *-*-*-*-*-*-*-*-
610 GGN=GNU(GND):DEGR=GND:FOR L1=0 TO GND:POLY(L1)=GNU(L1)/GGN:NEXT L1
620 GOSUB 8000:FOR L=1 TO GND:GNR(L,0)=ROOTS(L,0):GNR(L,1)=ROOTS(L,1):NEXT L
630 HGN=HNU(HND):DEGR=HND:FOR L=0 TO HND:POLY(L)=HNU(L)/HGN:NEXT L:GOSUB 8000
640 FOR L1=1 TO HND:HNR(L1,0)=ROOTS(L1,0):HNR(L1,1)=ROOTS(L1,1):NEXT L1
650 REM *-*-*-*-*-*-*-*-
660 CLS:LINE(0,190)-(639,200),0,BF
670 C$="          ZERO(##)=+#.####^^^^    j+#.####^^^^    "
680 L=0:G=0:FOR L1=1 TO GND
690 IF ((GNR(L1,0))^2+(GNR(L1,1))^2)^(.5)<1 THEN GOTO 710
700 G=G+1:MNR(G,0)=GNR(L1,0):MNR(G,1)=GNR(L1,1):GOTO 720
710 L=L+1:STR(L,0)=GNR(L1,0):STR(L,1)=GNR(L1,1)
720 NEXT L1:FOR L1=1 TO HDD
730 IF ((HDR(L1,0))^2+(HDR(L1,1))^2)^(.5)<1 THEN GOTO 750
740 G=G+1:MNR(G,0)=HDR(L1,0):MNR(G,1)=HDR(L1,1):GOTO 760
750 L=L+1:STR(L,0)=HDR(L1,0):STR(L,1)=HDR(L1,1)
760 NEXT L1:ST=L:US=G:IF ST=0 THEN GOTO 960
800 REM *-*-*-*-*-*-*-*-
810 CLS:LINE (0,190)-(639,200),0,BF
820 PRINT "          # of THE ZERO , YOU WANT TO INCLUDE ":PRINT
830 FOR L1=1 TO ST:PRINT USING C$;L1;STR(L1,0);STR(L1,1):NEXT L1
840 GG=0:FOR L1=1 TO ST:S(L1)=0:NEXT L1
850 IF GG+1>ST THEN GOTO 960
860 LOCATE ST+4,12:INPUT "NUMBER OF ZERO #:";I
870 IF 0>I OR I>ST OR S(I)=1 THEN GOTO 860
880 IF I=0 THEN GOTO 960
890 S(I)=1:GG=GG+1:MNR(G+GG,0)=STR(I,0):MNR(G+GG,1)=STR(I,1)
900 IF STR(I,1)=0 THEN GOTO 850
910 IF STR(I,1)=-STR(I-1,1) THEN LL=I-1 ELSE LL=I+1
920 S(LL)=1:GG=GG+1:MNR(G+GG,0)=STR(LL,0):MNR(G+GG,1)=STR(LL,1):GOTO 850
950 REM *-*-*-*-*-*-*-*-
960 LOCATE 1,1:MNDD=G+GG:FOR L1=1 TO MNDD
970 PRINT USING C$;L1;MNR(L1,0);MNR(L1,1):NEXT L1:XX=245
980 X1=245:Y1=55
1000 REM *-*-*-*-*-*-*-*-
1010 PRINT "                                                    "
1020 PRINT "       IF THE SYSTEM HAS A ONE REAL LHP ZERO : 1/2/3  "
1030 PRINT "       IF  NOT                              :   2/3  "
1040 PRINT "    1 - ALL DATA REGARDING DOMINANT POLES AND ZERO.    "
1050 PRINT "    2 - ONLY THE COORDINATES .                        "
1060 PRINT "    3 - ADD AN EXTRA ZERO .                           "
1070 D$=INKEY$:IF D$="" THEN GOTO 1070
```

```
1100 REM *-*-*-*-*-*-*-*-*-
1110 CT=VAL(D$):IF CT<1 OR CT>3 THEN GOTO 1070
1120 IF CT=1 AND ST<1 AND US>0 OR CT=2 THEN CT=2:GOTO 1310
1130 CLS:LINE (0,190)-(639,200),0,BF
1140 FOR L=1 TO MNDD:LOCATE L+1,1:PRINT USING C$;L;MNR(L,0);MNR(L,1):NEXT L
1150 IF CT=3 THEN GOTO 1270
1160 IF CT=1 AND ST=1 THEN I=US+1:GOTO 1260
1165 IF CT=1 AND ST<1 THEN CT=2:GOTO 1310
1170 FOR L=1 TO ST:IF STR(L,1)=0 THEN GOTO 1230
1180 NEXT L:BEEP:PRINT ,".. COMPLEX ZERO(S) .."
1190 BEEP:PRINT ,".. ONLY 2 OR 3 , WAIT  .."
1200 BEEP:PRINT ,".. PRESS ANY KEY TO CONTINUE .."
1210 C$=INKEY$:IF C$="" THEN 1210
1220 GOTO 650
1230 LOCATE 5+MNDD,10:INPUT "  NUMBER OF DOMINANT ZERO #:";I
1240 IF I<1 OR I>MNDD THEN GOTO 1230
1250 IF MNDD=1 AND (MNR(I,1)<>0 OR ABS(MNR(I,0))>1) THEN CT=2:GOTO 810
1255 IF MNDD>1 AND (MNR(I,1)<>0 OR ABS(MNR(I,0)))>1) THEN GOTO 1230
1260 GOSUB 4450:XX=MNR(I,0)*240+245:GOTO 1310
1270 GOSUB 4450:MNR(MNDD+1,0)=ZZZ:ST=ST+1
1300 REM *-*-*-*-*-*-*-*-*-
1310 CLS:FOR L=190 TO 200:LINE(0,L)-(640,L),0:NEXT L:GOSUB 4000
1320 FOR L=1 TO BBB:LINE (5+MMM*(L-1),185-45*C(L-1))-(5+MMM*L,185-45*C(L))
1330 NEXT L:LOCATE 21,10:PRINT USING" R.T."+Q2$;RT
1340 LOCATE 22,10:IF ABS(KV)>99 THEN PRINT USING"1/Kv "+Q3$;KV:GOTO 1360
1350 PRINT USING"1/Kv "+Q2$;KV
1360 LOCATE 23,10:PRINT USING" O.S."+Q2$;100*(OS-1):FOR L=1 TO BBB
1370 LINE (220+MMM*(L-1),185-45*Q(L-1))-(220+MMM*L,185-45*Q(L)):NEXT L
1380 LOCATE 21,37:PRINT USING" R.T."+Q2$;ORT
1390 LOCATE 22,37:IF ABS(OKV)>99 THEN PRINT USING"1/Kv "+Q3$;OKV:GOTO 1410
1400 PRINT USING"1/Kv "+Q2$;OKV
1410 LOCATE 23,37:PRINT USING" O.S."+Q2$;OOS:FOR L=1 TO BBB
1420 LINE (435+MMM*(L-1),185-45*W(L-1))-(435+MMM*L,185-45*W(L)):NEXT L
1430 LOCATE 21,62:PRINT USING" R.T."+Q2$;WRT
1440 LOCATE 22,62:IF ABS(WKV)>99 THEN PRINT USING"1/Kv "+Q3$;WKV:GOTO 1460
1450 PRINT USING"1/Kv "+Q2$;WKV
1460 LOCATE 23,62:PRINT USING" O.S."+Q2$;WOS
1470 GOSUB 4000:GOSUB 4100:GOSUB 4200:GOTO 1810
1500 REM *-*-*-*-*-*-*-*-*-
1510 LINE (ZZZ,103)-(ZZZ,107):FOR R=1 TO MNDD-INT(R/3)
1520 CIRCLE (MNR(R,0)*240+245,105-ABS(MNR(R,1))*100),3,,,,1.5:NEXT R
1530 C$=INKEY$:IF C$="" THEN GOTO 1530
1540 IF C$<>"R" AND C$<>"D" AND C$<>"F" AND C$<>"C" AND C$<>"S" AND C$<>"K" AND
       C$<>"L" AND C$<>"B" AND C$<>"O" AND C$<>"J" AND C$<>"Q" AND C$<>"W" AND
       C$<>"E"                          THEN GOTO 1530
1545 IF C$="E" THEN RUN"MENU.BAS"
1550 IF C$="B" THEN GOTO 650
1560 IF C$="O" THEN GOTO 5570
1570 IF C$<>"J" THEN GOTO 1600
```

```
1580 IF J= 1 THEN J=24:ADD$="("+CHR$(24)+")":LOCATE 5,1:PRINT ADD$;GOTO 1530
1590 IF J=24 THEN J=1 :ADD$="("+CHR$(25)+")":LOCATE 5,1:PRINT ADD$:GOTO 1530
1600 IF C$="Q" THEN GOSUB 4550:GOTO 1530
1610 IF C$="W" THEN GOSUB 4750:GOTO 1530
1620 IF C$="S" THEN GOTO 2010
1630 IF C$<>"F" THEN GOTO 1660
1640 IF CINT(((105-Y1)^2*144/25+(X1-245+J)^2)^(.5))>240 THEN GOTO 1530
1650 GOSUB 4150:X1=CINT(X1+J):GOSUB 4100:GOTO 1810
1660 IF C$<>"D" THEN GOTO 1690
1670 IF CINT(((105-Y1)^2*144/25+(X1-245-J)^2)^(.5))>240 THEN GOTO 1530
1680 GOSUB 4150:X1=CINT(X1-J):GOSUB 4100:GOTO 1810
1690 IF C$<>"R" THEN GOTO 1720
1700 IF CINT(((105-Y1+J*5/12)^2*144/25+(X1-245)^2)^(.5))>240 THEN GOTO 1530
1710 GOSUB 4150:Y1=CINT(Y1*12/5-J)/12*5:GOSUB 4100:GOTO 1810
1720 IF C$<>"C" THEN GOTO 1750
1730 IF CINT(((105-Y1-J*5/12)^2*144/25+(X1-245)^2)^(.5))>240 OR Y1+J*5/12)104
     THEN GOTO 1530
1740 GOSUB 4150:Y1=CINT(Y1*12/5+J)/12*5:GOSUB 4100:GOTO 1810
1750 IF CT<>3 THEN GOTO 1810
1760 IF C$<>"Y" THEN GOTO 1790
1770 IF CINT(ABS(XX-245-J))>240 THEN GOTO 1530
1780 GOSUB 4250:XX=CINT(XX-J):GOSUB 4200:GOTO 1810
1790 IF CINT(ABS(XX-245+J))>240 THEN GOTO 1530          'C$<>"L"
1800 GOSUB 4250:XX=CINT(XX+J):GOSUB 4200
1810 X=(X1-245)/240:Y=((105-Y1)*12/5)/240:Z=(XX-245)/240:R=(X*X+Y*Y)^(.5)
1850 REM *-*-*-*-*-*-*-*-
1860 PHI1=FNATAN(X,Y):ZETA=-LOG(R)/SQR((LOG(R))^2+PHI1^2):GAMA=FNATAN(1-X,Y)
1870 IF CT=2 THEN GOTO 1900
1880 BETA=FNATAN(X-Z,Y):ALFA=(BETA+GAMA-PI2):ZZET=-ZETA/SQR(1-ZETA^2)
1890 OMEG=ATN(ZZET)-ALFA+2*PI2:OSHO=SQR(1-ZETA^2)/COS(ALFA)*EXP(ZZET*OMEG)
1900 TPOT=OMEG/PHI1:LOCATE 1,1:PRINT USING "X:+.#####";X
1910 PRINT USING "Y: .#####";Y:PRINT USING "Z:+.#####";Z
1920 PRINT USING "Φ:###.##";PHI1/PI2*90:IF CT=2 THEN GOTO 1970
1930 IF CT=2 THEN GOTO 1970
1940 LOCATE 1,11:PRINT USING "  α:+##.##";ALFA/PI2*90
1950 LOCATE 1,43:PRINT USING "O.S.:###.##  ";OSHO*100
1960 LOCATE 2,48:PRINT USING "  Tp:###.##  ";TPOT
1970 LOCATE 3,52:PRINT USING " τ: #.##";ZETA
1975 LOCATE 5,1:PRINT ADD$
1980 GOTO 1510
2000 REM *-*-*-*-*-*-*-*-
2010 LINE (488,0)-(639,103),0,BF
2020 FOR L=0 TO MNDD+2:MNU(L)=0:MDE(L)=0:NEXT L:L=0
2030 MND=MNDD:MNU(0)=1:IF CT<>3 THEN GOTO 2070
2040 MND=MND+1:MNR(MND,0)=Z:MNR(MND,1)=0
2070 MDD=MND+GDD-GND:IF MDD<2 THEN MDD=2
2080 MDR(MDD,0)=X:MDR(MDD,1)=Y:MDR(MDD-1,0)=X:MDR(MDD-1,1)=-Y
```

```
2100 REM *-*-*-*-*-*-*-*-*-
2110 IF MDD=2 THEN GOTO 2210
2120 LOCATE 1,62:PRINT MDD-2;" POLE(S) NEEDED":FOR L1=1 TO MDD-2
2130 LOCATE L1+1,62:PRINT L1;"th REAL POLE AT"
2140 LOCATE L1+2,62:INPUT " ))):";NPR:IF NPR(=-1 AND NPR=)1 THEN GOTO 2140
2150 LOCATE L1+1,62:PRINT USING "## :AT ##.####### ";L1;NPR
2160 LOCATE L1+2,62:PRINT "                ":MDR(L1,0)=NPR:MDR(L1,1)=0:NEXT L1
2210 GOSUB 3800
2400 REM *-*-*-*-*-*-*-*-*-
2410 IF KV(ABS(DKV) OR (ACT=0 AND (ACT=1 OR KVINF=0)) THEN 2610
2420 AD$="("+CHR$(25)+")":KVC=KV-DKV
2425 ZR=(KVC*(1-PL)+PL)/(KVC*(1-PL)+1):GOTO 2510
2430 C$=INKEY$:IF C$="" THEN 2430
2440 IF C$()"K" AND C$()"L" AND C$()"J" AND C$()"S" THEN 2430
2450 IF C$ ="S" THEN GOTO 2560
2460 IF C$()"J" THEN GOTO 2490
2470 IF JJ=.01  THEN JJ=.001:AD$="("+CHR$(25)+")":GOTO 2510
2480 IF JJ=.001 THEN JJ=.01 :AD$="("+CHR$(24)+")":GOTO 2510
2490 IF C$="K" AND ABS(PL-JJ)(.999 THEN PL=PL-JJ
2500 IF C$="L" AND ABS(PL+JJ)(.999 THEN PL=PL+JJ
2510 LOCATE 11,66:PRINT " DIPOLE "+AD$:KVC=KV-DKV
2520 ZR=(KVC*(1-PL)+PL)/(KVC*(1-PL)+1)
2530 LOCATE 12,65:PRINT USING "ZR: #.####### ";ZR
2540 LOCATE 13,65:PRINT USING "PL: #.###   ";PL
2550 GOTO 2430
2560 MND=MND+1:MNR(MND,0)=ZR:MNR(MND,1)=0
2570 MDD=MDD+1:MDR(MDD,0)=PL:MDR(MDD,1)=0
2610 GOSUB 3800:LOCATE 15,3:PRINT "*"
2760 SNUM=0:FOR L1=0 TO MND :SNUM=SNUM+MNU(L1):NEXT L1
2770 SDEN=0:FOR L1=0 TO MDD :SDEN=SDEN+MDE(L1):NEXT L1:NU=SDEN/SNUM
2800 REM *-*-*-*-*-*-*-*-*-
2810 IF MND>0 THEN FOR L=1 TO MND:YYR(L,0)=MNR(L,0):YYR(L,1)=MNR(L,1):NEXT L
2815 IF GDD(1 THEN 3OTO 2830
2820 FOR L=1 TO GDD:LL=L+MND:YYR(LL,0)=GDR(L,0):YYR(LL,1)=GDR(L,1):NEXT L
2830 FOR L=1 TO MDD:UUR(L,0)=MDR(L,0):UUR(L,1)=MDR(L,1):NEXT L
2835 IF GND(1 THEN GOTO 2850
2840 FOR L=1 TO GND:LL=L+MDD:UUR(LL,0)=GNR(L,0):UUR(LL,1)=GNR(L,1):NEXT L
2850 YYY=MND+GDD:UUU=MDD+GND:EPS=.00001:GOSUB 7000:GOSUB 7200:PGN=NU/GGN
2860 PND=TTT:PDD=III:PNU(0)=TTP(0):PDE(0)=IIP(0)
2870 FOR L=1 TO PND:PNR(L,0)=TTR(L,0):PNR(L,1)=TTR(L,1):PNU(L)=TTP(L):NEXT L
2880 FOR L=1 TO PDD:PDR(L,0)=IIR(L,0):PDR(L,1)=IIR(L,1):PDE(L)=IIP(L):NEXT L
2890 LOCATE 15,4:PRINT "*"
2900 REM *-*-*-*-*-*-*-*-*-
2910 IF PND>0 THEN FOR L=1 TO PND:YYR(L,0)=PNR(L,0):YYR(L,1)=PNR(L,1):NEXT L
2915 IF HND(1 THEN GOTO 2930
2920 FOR L=1 TO HND:LL=L+PND:YYR(LL,0)=HNR(L,0):YYR(LL,1)=HNR(L,1):NEXT L
2930 FOR L=1 TO PDD:UUR(L,0)=PDR(L,0):UUR(L,1)=PDR(L,1):NEXT L
2940 FOR L=1 TO HDD:LL=L+PDD:UUR(LL,0)=HDR(L,0):UUR(LL,1)=HDR(L,1):NEXT L
```

```
2950 FOR L=1 TO GDD:LL=L+FDD+HDD:UUR(LL,0)=GDR(L,0):UUR(LL,1)=GDR(L,1):NEXT L
2960 YYY=PND+HND:UUU=PDD+HDD+GDD:EPS=.00001:GOSUB 7000:GOSUB 7200
2965 CNG=TTT:CDG=III:CED(0)=IIP(0)
2980 FOR L=1 TO CDG:CRD(L,0)=IIR(L,0):CRD(L,1)=IIR(L,1):CED(L)=IIP(L)
2990 CUN(L)=0:NEXT L:FOR L=0 TO CNG:CUN(L)=TTP(L):NEXT L
3000 FOR L=0 TO CDG:CUN(L)=CED(L)-PGN*HGN*CUN(L):POLY(L)=CUN(L):NEXT L
3005 DEGR=CDG:IF ACCT=0 OR ACCT>2 THEN GOTO 3025
3010 FOR L=1 TO ACCT:S1=0:FOR L1=0 TO CDG-L:POLY(CDG-L1)=POLY(CDG-L1)+S1
3015 S1=POLY(CDG-L1):NEXT L1:CRN(CDG-L+1,0)=1:CRN(CDG-L+1,1)=0:NEXT L
3020 DEGR=CDG-ACCT:IF ACCT>2 THEN DEGR=CDG-2
3023 FOR L=ACCT TO CDG:SWAP POLY(L-ACCT),POLY(L):NEXT L
3025 GOSUB 8000:LOCATE 15,5:PRINT "*"
3030 FOR L=1 TO DEGR:CRN(L,0)=ROOTS(L,0):CRN(L,1)=ROOTS(L,1):NEXT L
3040 AA=0:FOR L=1 TO DEGR:IF CRN(L,1)<>0 THEN GOTO 3085
3042 AA=AA+1:LL=0 :LOCATE 16,L+2:PRINT "*"
3045 DS=0:DS1=0:DS2=0:FOR L1=0 TO CDG:DS =DS -CUN(L1)*(CRN(L,0))^L1:NEXT L1
3050 RRT=CRN(L,0)+.0000001:FOR L1=0 TO CDG:DS1=DS1-CUN(L1)*(RRT)^L1:NEXT L1
3055 RRT=CRN(L,0)-.0000001:FOR L1=0 TO CDG:DS2=DS2-CUN(L1)*(RRT)^L1:NEXT L1
3060 IF ABS(DS1)>ABS(DS2) THEN SS=-1 ELSE SS=1
3070 TT=0:LL=LL+1:DSA=0:RRT=CRN(L,0)+SS*LL*(.0000001)
3073 FOR L1=0 TO CDG:DSA=DSA-CUN(L1)*(RRT)^L1:NEXT L1
3075 IF ABS(DSA)<ABS(DS) THEN DS=DSA:GOTO 3070
3076 CRN(L,0)=RRT-SS*LL*(1E-08)
3077 LOCATE 16,L+2:PRINT " "
3085 NEXT L
3100 REM *-*-*-*-*-*-*-*-*-
3110 FOR L=1 TO CDG:YYR(L,0)=CRD(L,0):YYR(L,1)=CRD(L,1):NEXT L
3120 FOR L=1 TO PND:LL=L+CDG:YYR(LL,0)=PNR(L,0):YYR(LL,1)=PNR(L,1):NEXT L
3130 FOR L=1 TO CDG:UUR(L,0)=CRN(L,0):UUR(L,1)=CRN(L,1):NEXT L
3140 FOR L=1 TO PDD:LL=L+CDG:UUR(LL,0)=PDR(L,0):UUR(LL,1)=PDR(L,1):NEXT L
3150 YYY=PND+CDG:UUU=PDD+CDG:EPS=.00001:GOSUB 7000:GOSUB 7200:CGN=NU/GGN
3160 CNG=TTT:CDG=III:CUN(0)=TTP(0):CED(0)=IIP(0)
3165 FOR L=1 TO CNG:CRN(L,0)=TTR(L,0):CRN(L,1)=TTR(L,1):CUN(L)=TTP(L):NEXT L
3170 FOR L=1 TO CDG:CRD(L,0)=IIR(L,0):CRD(L,1)=IIR(L,1):CED(L)=IIP(L):NEXT L
3180 LOCATE 15,6:PRINT "*"
3200 REM *-*-*-*-*-*-*-*-*-
3210 FOR L=1 TO BBB:LINE (5+MMM*(L-1),185-45*C(L-1))-(5+MMM*L,185-45*C(L)),0
3220 NEXT L:FOR L=1 TO BBB:A(L)=0:B(L)=0:C(L)=0:D(L)=0:E(L)=0:NEXT L
3230 FOR L=0 TO PND:A(L*NNN)=PNU(L):NEXT:FOR L=0 TO PDD:B(L*NNN)=PDE(L):NEXT
3240 FOR L=0 TO MND:  C(L+1)=NU*MNU(L)                :NEXT L   :C(0)=0
3250 FOR L=0 TO MDD:  D(L)  =   MDE(L)                :NEXT L   :SP2=0
3260 FOR L=0 TO MDD+1:SP1=D(L):D(L)=SP2-SP1:SP2=SP1   :NEXT L
3270 FOR L=0 TO INT(200/NNN):         E(L)=C(MDD+1)
3280 FOR L2=1 TO MDD+1:C(MDD+2-L2)=C(MDD+1-L2)-E(L)*D(MDD+1-L2):NEXT L2:NEXT L
```

```
3300 REM *-*-*-*-*-*-*-*-*-
3305 OS=-1:IF NNN=1 THEN FOR L=0 TO BBB:C(L)=E(L):NEXT L:GOTO 3560
3310 PPD=NNN*PDD:PPN=NNN*PND:LOCATE 15,7:PRINT "*"
3320 FOR L=0 TO PPN+VNG:C(L)=0:NEXT L
3330 FOR L1=0 TO PPN:FOR L2=0 TO VNG
3340 C(L1+L2)=A(L1)*PGN*VNU(L2)+C(L1+L2):NEXT L2:NEXT L1:PPN=PPN+VNG
3350 FOR L=0 TO PPD+VDG:A(L)=C(L):C(L)=0:NEXT L
3360 FOR L1=0 TO PPD:FOR L2=0 TO VDG
3370 C(L1+L2)=B(L1)*VDE(L2)+C(L1+L2):NEXT L2:NEXT L1:PPD=PPD+VDG
3380 FOR L=0 TO PPN+NHN:B(L)=C(L):C(L)=0:NEXT L
3390 FOR L1=0 TO PPN:FOR L2=0 TO NHN
3400 C(L1+L2)=A(L1)*AA(L2)+C(L1+L2):NEXT L2:NEXT L1:PNN=PPN+NHN
3410 FOR L=0 TO PPD+DHD:A(L)=C(L):C(L)=0:NEXT L
3420 FOR L1=0 TO PPD:FOR L2=0 TO DHD
3430 C(L1+L2)=B(L1)*BB(L2)+C(L1+L2):NEXT L2:NEXT L1:DDD=PPD+DHD
3440 FOR L=0 TO DDD:B(L)=C(L):C(L)=0:NEXT L
3500 REM *-*-*-*-*-*-*-*-*-
3510 LOCATE 15,2:PRINT "      ":FOR L1=0 TO BBB
3520 IF NNN*INT(L1/NNN)<>L1 OR A(DDD)=0 OR A(DDD)=E(DDD) THEN GOTO 3540
3530 FOR L=0 TO DDD:A(L)=E(INT(L1/NNN))/A(DDD)*A(L):NEXT L
3540 C(L1)=A(DDD)
3550 FOR  L2=1 TO DDD:A(DDD+1-L2)=A(DDD-L2)-C(L1)*B(DDD-L2):NEXT L2:NEXT L1
3560 FOR L1=1 TO BBB:IF C(L1-1)<=C(L1) THEN NEXT L1
3565 OS=C(L1-1):FOR L1=1 TO BBB
3567 LINE (5+MMM*(L1-1),185-45*C(L1-1))-(5+MMM*L1,185-45*C(L1)):NEXT L1
3570 FOR L1=6 TO 180:IF POINT(L1,140)<>1 THEN NEXT L1
3580 RT=(L1-5)/NNN/MMN:LOCATE 21,10:PRINT USING" R.T."+@2$;RT:LOCATE 22,10
3590 IF ABS(KV)>99 THEN PRINT USING"1/kv "+@3$;KV:GOTO 3610
3600 PRINT USING"1/Kv "+@2$;KV
3610 LOCATE 23,10:PRINT USING" O.S."+@2$;100*(OS-1)
3620 GOSUB 4000
3630 GOTO 1530
3800 REM *-*-*-*-*-*-*-*-*-
3810 FOR L=1 TO MND:YYR(L,0)=MNR(L,0):YYR(L,1)=MNR(L,1):NEXT L
3820 FOR L=1 TO MDD:UUR(L,0)=MDR(L,0):UUR(L,1)=MDR(L,1):NEXT L
3830 YYY=MND:UUU=MDD:EPS=.00001:GOSUB 7000:GOSUB 7200
3840 MND=TTT:FOR L=1 TO MND:MNR(L,0)=TTR(L,0):MNR(L,1)=TTR(L,1):NEXT L
3850 MDD=III:FOR L=1 TO MDD:MDR(L,0)=IIR(L,0):MDR(L,1)=IIR(L,1):NEXT L
3860 FOR L=0 TO MND:MNU(L)=TTP(L):NEXT L:FOR L=0 TO MDD:MDE(L)=IIP(L):NEXT L
3870 SU1=0:FOR L=0 TO MND:SU1=SU1+L*MNU(L):NEXT L
3880 SU2=0:FOR L=0 TO MND:SU2=SU2+  MNU(L):NEXT L:IF SU2=0 THEN SU2=1E-10
3890 SU3=0:FOR L=0 TO MDD:SU3=SU3+L*MDE(L):NEXT L
3900 SU4=0:FOR L=0 TO MDD:SU4=SU4+  MDE(L):NEXT L:IF SU4=0 THEN SU4=1E-10
3910 KV=SU3/SU4-SU1/SU2
3920 RETURN
```

```
4000 REM *-*-*-*-*-*-*-*- CIRCLE,LINE,LINE
4010 CIRCLE (245,105),240,,,0.3.141
4020 LINE (0,105)-(490,105):LINE (245,0)-(245,105)
4030 XXL=110:XLL=199:GOSUB 4300
4040 IF CT=2 OR ZTA=0 THEN GOTO 4090
4050 FOR TWN=.010797 TO 3.141593 STEP .04
4060 R=EXP(-ZTA*TWN):PH=(1-ZTA^2)^(.5)*TWN
4070 PSET (240*R*COS(PH)+245,105-R*SIN(PH)*100)
4080 NEXT TWN
4090 RETURN
4100 REM *-*-*-*-*-*-*-*- POLE SET FOR X1,Y1
4110 LINE (X1-3,Y1-1)-(X1+3,Y1+1)
4120 LINE (X1-3,Y1+1)-(X1+3,Y1-1)
4130 RETURN
4150 REM *-*-*-*-*-*-*-*- POLE RESET FOR X1,Y1
4160 LINE (X1-3,Y1-1)-(X1+3,Y1+1),0
4170 LINE (X1-3,Y1+1)-(X1+3,Y1-1),0
4180 RETURN
4200 REM *-*-*-*-*-*-*-*- ZERO SET FOR XX
4210 LINE (XX-1,104)-(XX+1,104)
4220 LINE (XX-1,106)-(XX+1,106)
4230 RETURN
4250 REM *-*-*-*-*-*-*-*- ZERO RESET FOR XX
4260 LINE (XX-1,104)-(XX+1,104),0
4270 LINE (XX-1,106)-(XX+1,106),0
4280 RETURN
4300 REM *-*-*-*-*-*-*-*- LINE,LINE
4310 FOR L=SSS TO 200 STEP SSS
4320 LINE (  5+L,XLL-3)-(  5+L,XLL-1)
4330 LINE (220+L,XLL-3)-(220+L,XLL-1)
4340 LINE (435+L,XLL-3)-(435+L,XLL-1):NEXT L
4350 LINE (  5,XXL)-(  5,XLL):LINE (205,XXL)-(205,XLL)
4360 LINE (220,XXL)-(220,XLL):LINE (420,XXL)-(420,XLL)
4370 LINE (435,XXL)-(435,XLL):LINE (635,XXL)-(635,XLL)
4380 LINE (  5,XXL)-(205,XXL):LINE (  5,XLL)-(205,XLL)
4390 LINE (220,XXL)-(420,XXL):LINE (220,XLL)-(420,XLL)
4400 LINE (435,XXL)-(635,XXL):LINE (435,XLL)-(635,XLL)
4410 RETURN
4450 REM *-*-*-*-*-*-*-*-
4460 PRINT:INPUT "          INPUT  ZETA:";ZTA:INPUT "          INPUT  Tp/T:";KK
4470 ZZT=ZTA/(1-ZTA^2)^(.5):SS=1000:FOR LF=-1.05 TO .091 STEP .01
4480 OSI=SS:SS=100*((1-ZTA^2)^(.5))/COS(LF)*EXP(-ZZT*(ATN(-ZZT)-LF+3.141593))
4490 IF OSI-SS>0 THEN NEXT LF
4500 LFO=LF:KVIMP=0:IF ACT=0 THEN INPUT " 1/Kv IMPROVEMENT (Y):";C$
4510 IF ACT=0 AND C$="Y" THEN INPUT "         DESIRED 1/Kv:";DKV:KVIMP=1
4520 PHI=1/KK*(ATN(-ZZT)-LFO+3.141593):RP=EXP(-ZZT*PHI):X=RP*COS(PHI):SS=1
4530 Y=RP*ABS(SIN(PHI)):GAMA=FNATAN(1-X,Y):L=Y^2+(1-X)^2:L=SQR(L):X1=245+240*X
4540 Y1=105-100*Y:ZZZ=X-L*SIN(GAMA)/TAN(PI2-GAMA+LFO):XX=245+240*ZZZ:RETURN
```

```
4550 REM *-*-*-*-*-*-*-*-
4560 GO=1:QKV=KV:QNU=NU:QOS=100*(OS-1):QND=MND:QDD=MDD:QRT=RT:QCGN=CGN
4570 FOR L1=0 TO QND :   QNU(L1)=MNU(L1)                          :NEXT L1
4580 FOR L1=1 TO QND : QNR(L1,0)=MNR(L1,0):QNR(L1,1)=MNR(L1,1) :NEXT L1
4590 FOR L1=0 TO QDD :   QDE(L1)=MDE(L1)                          :NEXT L1
4600 FOR L1=1 TO QDD : QDR(L1,0)=MDR(L1,0):QDR(L1,1)=MDR(L1,1) :NEXT L1
4610 FOR L1=0 TO CNG :  QCNU(L1)=CUN(L1)  :NEXT L1:QCND=CNG
4620 FOR L1=1 TO CNG :QCNR(L1,0)=CRN(L1,0):QCNR(L1,1)=CRN(L1,1):NEXT L1
4630 FOR L1=0 TO CDG :  QCDE(L1)=CED(L1)  :NEXT L1:QCDD=CDG
4640 FOR L1=1 TO CDG :QCDR(L1,0)=CRD(L1,0):QCDR(L1,1)=CRD(L1,1):NEXT L1
4650 FOR L=1 TO BBB
4660 LINE (220+MMM*(L-1),185-45*Q(L-1))-(220+MMM*L,185-45*Q(L)),0:NEXT L
4670 Q(0)=C(0):FOR L=1 TO BBB:Q(L)=C(L)
4680 LINE (220+MMM*(L-1),185-45*Q(L-1))-(220+MMM*L,185-45*Q(L)):NEXT L
4690 LOCATE 21,37:PRINT USING" R.T."+Q2$;QRT
4700 LOCATE 22,37:IF ABS(QKV)>99 THEN PRINT USING"1/Kv "+Q3$;QKV:GOTO 4720
4710 PRINT USING"1/Kv "+Q2$;QKV
4720 LOCATE 23,37:PRINT USING" O.S."+Q2$;QOS
4730 RETURN
4750 REM *-*-*-*-*-*-*-*-
4760 WW=1:WKV=KV:WNU=NU:WOS=100*(OS-1):WND=MND:WDD=MDD:WRT=RT:WCGN=CGN
4770 FOR L1=0 TO WND :   WNU(L1)=MNU(L1)                          :NEXT L1
4780 FOR L1=1 TO WND : WNR(L1,0)=MNR(L1,0):WNR(L1,1)=MNR(L1,1) :NEXT L1
4790 FOR L1=0 TO WDD :   WDE(L1)=MDE(L1)                          :NEXT L1
4800 FOR L1=1 TO WDD : WDR(L1,0)=MDR(L1,0):WDR(L1,1)=MDR(L1,1) :NEXT L1
4810 FOR L1=0 TO CNG :  WCNU(L1)=CUN(L1)  :NEXT L1:WCND=CNG
4820 FOR L1=1 TO CNG :WCNR(L1,0)=CRN(L1,0):WCNR(L1,1)=CRN(L1,1):NEXT L1
4830 FOR L1=0 TO CDG :  WCDE(L1)=CED(L1)  :NEXT L1:WCDD=CDG
4840 FOR L1=1 TO CDG :WCDR(L1,0)=CRD(L1,0):WCDR(L1,1)=CRD(L1,1):NEXT L1
4850 FOR L=1 TO BBB
4860 LINE (435+MMM*(L-1),185-45*W(L-1))-(435+MMM*L,185-45*W(L)),0:NEXT L
4870 W(0)=C(0):FOR L1=1 TO BBB:W(L1)=C(L1)
4880 LINE (435+MMM*(L1-1),185-45*W(L1-1))-(435+MMM*L1,185-45*W(L1)):NEXT L1
4890 LOCATE 21,62:PRINT USING" R.T."+Q2$;WRT
4900 LOCATE 22,62:IF ABS(WKV)>99 THEN PRINT USING"1/Kv "+Q3$;WKV:GOTO 4920
4910 PRINT USING"1/Kv "+Q2$;WKV
4920 LOCATE 23,62:PRINT USING" O.S."+Q2$;WOS
4930 RETURN
5000 REM *-*-*-*-*-*-*-*-
5010 Q$=                                  "   ----------------------------
     &     ----------------------------  "
5020 W$="NUM.        DEN.         NUMERATOR ROOT(S)       DENOMINATOR ROOT(S) "
5030 E$=                  "##:##.###^^^^ ##.####^^^^ +
   + ##.####^^^^ ##.####^^^^"
5040 R$=                  "##:            ##.####^^^^ +
   + ##.####^^^^ ##.####^^^^"
5050 T$=                  "##:##.####^^^^ ##.####^^^^ + ##.####^^^^ ##.#### ^^
^  + ##.####^^^^ ##.####^^^^"
5060 Y$="##:##.####^^^^ ##.####^^^^ +        GAIN :  ##.###^^^^        "
5070 U$="##:##.####^^^^ ##.####^^^^ +                                  "
```

```
5100 REM *-*-*-*-*-*-*-*-*-
5110 LOCATE 15,23:PRINT "FRAME  NUMBER : 1 OR Q OR W   (1/Q/W) "
5120 LOCATE 14,34:PRINT " GO BACK ( ) "
5130 C$=INKEY$:IF C$="" THEN GOTO 5130
5140 IF C$=" " THEN GOTO 1300
5150 IF C$<>"1" AND C$<>"Q" AND C$<>"W" THEN GOTO 5130
5160 IF C$<>"1" AND (C$<>"Q" OR QQ<>1) AND (C$<>"W" OR WW<>1) THEN GOTO 5130
5200 REM *-*-*-*-*-*-*-*-*-
5210 LOCATE 13,1:PRINT USING Q$;"M(z)"
5220 PRINT "       ";W$:IF C$="Q" OR C$="W" THEN GOTO 5330
5240 FOR L=MDD TO MND+1 STEP -1
5250 PRINT USING R$;L;MDE(L);MDR(L,0);MDR(L,1):NEXT L
5260 FOR L=MND TO 1 STEP -1
5270 PRINT USING T$;L;MNU(L);MDE(L);MNR(L,0);MNR(L,1);MDR(L,0);MDR(L,1)
5280 NEXT L:PRINT USING Y$;0;MNU(0);MDE(0);MU:PRINT:PRINT USING Q$;"C(z)"
5290 PRINT "       ";W$:FOR L=CDG TO 1 STEP -1
5300 PRINT USING T$;L;CUN(L);CED(L);CRN(L,0);CRN(L,1);CRD(L,0);CRD(L,1)
5310 NEXT L:PRINT USING Y$;0;CUN(0);CED(0);CGN:GOTO 5540
5320 REM *-*-*-*-*-*-*-*-*-
5330 LOCATE 15,1:IF C$="W" THEN GOTO 5440
5350 FOR L=QDD TO QND+1 STEP -1
5360 PRINT USING R$;L;QDE(L);QDR(L,0);QDR(L,1):NEXT L
5370 FOR L=QND TO 1 STEP -1
5380 PRINT USING T$;L;QNU(L);QDE(L);QNR(L,0);QNR(L,1);QDR(L,0);QDR(L,1)
5390 NEXT L:PRINT USING Y$;0;QNU(0);QDE(0);QNU:PRINT:PRINT USING Q$;"C(z)"
5400 PRINT "       ";W$:FOR L=QCDD TO 1 STEP -1
5410 PRINT USING T$;L;QCNU(L);QCDE(L);QCNR(L,0);QCNR(L,1);QCDR(L,0);QCDR(L,1)
5420 NEXT L:PRINT USING Y$;0;QCNU(0);QCDE(0):QCGN:GOTO 5540
5430 REM *-*-*-*-*-*-*-*-*-
5440 LOCATE 14,1:PRINT USING E$;WDD;WNU:WDE(WDD);WDR(WDD,0);WDR(WDD,1)
5450 FOR L=WDD TO WND+1 STEP -1
5460 PRINT USING R$;L;WDE(L);WDR(L,0);WDR(L,1):NEXT L
5470 FOR L=WND TO 1 STEP -1
5480 PRINT USING T$;L;WNU(L);WDE(L);WNR(L,0);WNR(L,1);WDR(L,0);WDR(L,1)
5490 NEXT L:PRINT USING Y$;0;WNU(0);WDE(0);WNU:PRINT:PRINT USING Q$;"C(z)"
5500 PRINT "       ";W$:FOR L=WCDD TO 1 STEP -1
5510 PRINT USING T$;L;WCNU(L);WCDE(L);WCNR(L,0);WCNR(L,1);WCDR(L,0);WCDR(L,1)
5520 NEXT L:PRINT USING Y$;0;WCNU(0);WCDE(0);WCGN
5530 REM *-*-*-*-*-*-*-*-*-
5540 PRINT "         PRINT OUT (Y)"
5550 C$=INKEY$:IF C$="" THEN GOTO 5550
5560 IF C$="Y" THEN GOTO 5800
5570 CLS:LINE(0,190)-(639,200),0,BF:XXL=3:XLL=92:GOSUB 4300
5580 REM *-*-*-*-*-*-*-*-*-
5590 FOR L=1 TO BBB:LINE(5+MMM*(L-1),80-45*C(L-1))-(5+MMM*L,80-45*C(L))
5600 NEXT L:LOCATE 7,10:PRINT USING" R.T."+02$;RT
5610 LOCATE 8,10:IF ABS(KV)>99 THEN PRINT USING"1/Kv "+03$;KV:GOTO 5630
5620 PRINT USING"1/Kv "+02$;KV
5630 LOCATE 9,10:PRINT USING" O.S."+02$;100*(OS-1)
```

```
5640 REM *-*-*-*-*-*-*-*-*-
5650 FOR L=1 TO BBB:LINE (220+MMM*(L-1),80-45*Q(L-1))-(220+MMM*L,80-45*Q(L))
5660 NEXT L:LOCATE 7,37:PRINT USING" R.T."+Q2$;QRT
5670 LOCATE 8,37:IF ABS(QKV)>99 THEN PRINT USING"1/kv "+Q3$;QKV:GOTO 5690
5680 PRINT USING"1/Kv "+Q2$;QKV
5690 LOCATE 9,37:PRINT USING" O.S."+Q2$;QOS
5700 REM *-*-*-*-*-*-*-*-*-
5710 FOR L=1 TO BBB:LINE (435+MMM*(L-1),80-45*W(L-1))-(435+MMM*L,80-45*W(L))
5720 NEXT L:LOCATE 7,62:PRINT USING" R.T."+Q2$:WRT
5730 LOCATE 8,62:IF ABS(WKV)>99 THEN PRINT USING"1/kv "+Q3$;WKV:GOTO 5750
5740 PRINT USING"1/Kv "+Q2$;WKV
5750 LOCATE 9,62:PRINT USING" O.S."+Q2$;WOS:GOTO 5000
5800 REM *-*-*-*-*-*-*-*-*-
5810 LPRINT CHR$(15)
5820 LPRINT USING Q$;"M(z)":LPRINT "       ";W$:IF IN=1 OR IN=2 THEN GOTO 5940
5830 LPRINT USING E$;MDD;MU;MDE(MDD);MDR(MDD,0);MDR(MDD,1)
5840 IF MDD-MND<>2 THEN GOTO 5860
5850 LPRINT USING R$;MDD-1;MDE(MDD-1);MDR(MDD-1,0);MDR(MDD-1,1)
5860 FOR L=MND TO 1 STEP -1
5870 LPRINT USING T$;L;MNU(L);MDE(L);MNR(L,0);MNR(L,1);MDR(L,0);MDR(L,1)
5880 NEXT L:LPRINT USING Y$;0;MNU(0);MDE(0):LPRINT:LPRINT USING Q$;"C(z)"
5890 LPRINT "       ";W$:FOR L=CDG TO 1 STEP -1
5900 LPRINT USING T$;L;CUN(L);CED(L);CRN(L,0);CRN(L,1);CRD(L,0);CRD(L,1)
5910 NEXT L:LPRINT USING Y$;0;CUN(0);CED(0);CGN
5920 FOR L=0 TO 200:E(L)=C(L):NEXT L:EOS=OS:EKV=KV:ERT=RT:GOTO 6170
5930 REM *-*-*-*-*-*-*-*-*-
5940 IF IN=2 THEN GOTO 6060
5950 LPRINT USING E$;QDD;QNU;QDE(QDD);QDR(QDD,0);QDR(QDD,1)
5960 IF QDD-QND<>2 THEN GOTO 5980
5970 LPRINT USING R$;QDD-1;QDE(QDD-1);QDR(QDD-1,0);QDR(QDD-1,1)
5980 FOR L=QND TO 1 STEP -1
5990 LPRINT USING T$;L;QNU(L);QDE(L);QNR(L,0);QNR(L,1);QDR(L,0);QDR(L,1)
6000 NEXT L:LPRINT USING Y$;0;QNU(0);QDE(0):LPRINT:LPRINT USING Q$;"C(z)"
6010 LPRINT "       ";W$:FOR L=QCDD TO 1 STEP -1
6020 LPRINT USING T$;L;QCNU(L);QCDE(L);QCNR(L,0);QCNR(L,1);QCDR(L,0);QCDR(L,1)
6030 NEXT L:LPRINT USING Y$;0;QCNU(0);QCDE(0);QCGN
6040 FOR L=0 TO 200:E(L)=Q(L):NEXT L:EOS=1+QOS/100:EKV=QKV:ERT=QRT:GOTO 6170
6050 REM *-*-*-*-*-*-*-*-*-
6060 LPRINT USING E$;WDD;WNU;WDE(WDD);WDR(WDD,0);WDR(WDD,1)
6070 IF WDD-WND<>2 THEN GOTO 6090
6080 LPRINT USING R$;WDD-1;WDE(WDD-1);WDR(WDD-1,0);WDR(WDD-1,1)
6090 FOR L=WND TO 1 STEP -1
6100 LPRINT USING T$;L;WNU(L);WDE(L);WNR(L,0);WNR(L,1);WDR(L,0);WDR(L,1)
6110 NEXT L:LPRINT USING Y$;0;WNU(0);WDE(0):LPRINT:LPRINT USING Q$;"C(z)"
6120 LPRINT "       ";W$:FOR L=WCDD TO 1 STEP -1
6130 LPRINT USING T$;L;WCNU(L);WCDE(L);WCNR(L,0);WCNR(L,1);WCDR(L,0);WCDR(L,1)
6140 NEXT L:LPRINT USING Y$;0;WCNU(0);WCDE(0);WCGN
6150 FOR L=0 TO 200:E(L)=W(L):NEXT L:EOS=1+WOS/100:EKV=WKV:ERT=WRT
```

```
6160 REM *-*-*-*-*-*-*-*-*-
6170 LPRINT CHR$(14);TAB(12);"STEP RESPONSE";CHR$(15)
6180 LPRINT "   Y(t)":LPRINT "    !":MX=CINT(EOS*15)+1:DF=0
6190 C$="":FOR L1=0 TO 73
6200 IF CINT(15*E(L1))=MX-DF AND L1 MOD NUM=0 THEN C$=C$+"0":GOTO 6230
6210 IF CINT(15*E(L1))=MX-DF THEN C$=C$+"*":GOTO 6230
6220 C$=C$+" "
6230 NEXT L1:DF=DF+1
6240 IF DF=MX+1 THEN GOTO 6330
6250 FOR L1=0 TO 10:RED=.2*(MX-DF+1)/3
6260 IF MX-DF+1=3*L1 THEN GOTO 6280
6270 NEXT L1:GOTO 6320
6280 IF RED>1 THEN LPRINT STR$(RED)+"!"+C$
6290 IF RED=1 THEN LPRINT STR$(RED)+"  !"+C$
6300 IF RED<1 THEN LPRINT STR$(RED)+" !"+C$
6310 GOTO 6190
6320 LPRINT "    !"+C$:GOTO 6190
6330 C$="!":IF E(0)=0 THEN C$=C$+"0"
6340 FOR L1=1 TO 73
6350 IF CINT(15*E(L1))=0 AND L1 MOD NUM=0 THEN C$=C$+"0":GOTO 6380
6360 IF CINT(15*E(L1))=0 THEN C$=C$+"*":GOTO 6380
6370 C$=C$+"-"
6380 NEXT L1
6390 LPRINT " 0.0"+C$
6400 LPRINT "     0    5    1    5    2    5    3    5    4    5    5    5    6
    5    7  3"
6410 LPRINT "          OVERSHOOT              = ";CINT((EOS-1)*100)
6420 LPRINT "          RAMP-ERROR CONS. (Kv) = ";EKV
6430 LPRINT "          RISE TIME              = ";ERT
6440 GOTO 5570
7000 REM *-*-*-*-*-*-*-*-*-
7010 FOR L=1 TO YYY:YYR(L,2)=0:NEXT L:FOR L=1 TO UUU:UUR(L,2)=0:NEXT L
7020 IF YYY=0 OR UUU=0 THEN 7080
7030 FOR L1=1 TO YYY:FOR L2=1 TO UUU
7040 IF ABS(YYR(L1,0)-UUR(L2,0))>EPS OR ABS(YYR(L1,1)-UUR(L2,1))>EPS OR
        YYR(L1,2)=1 OR UUR(L2,2)=1 THEN 7060
7050 YYR(L1,2)=1:UUR(L2,2)=1:GOTO 7070
7060 NEXT L2
7070 NEXT L1
7080 TTT=0:IF YYY=0 THEN 7120
7090 FOR L=1 TO YYY:IF YYR(L,2)=1 THEN GOTO 7110
7100 TTT=TTT+1:TTR(TTT,0)=YYR(L,0):TTR(TTT,1)=YYR(L,1)
7110 NEXT L
7120 III=0:IF UUU=0 THEN 7160
7130 FOR L=1 TO UUU:IF UUR(L,2)=1 THEN GOTO 7150
7140 III=III+1:IIR(III,0)=UUR(L,0):IIR(III,1)=UUR(L,1)
7150 NEXT L
7160 RETURN
```

```
7200 REM *-*-*-*-*-*-*-*-*-
7210 FOR L=0 TO TTT:TTP(L)=0:NEXT L:FOR L=0 TO III:IIP(L)=0:NEXT L
7220 L=0:E=0:TTP(0)=1:IIP(0)=1
7230 L=L+1:IF L>TTT THEN GOTO 7300
7240 IF TTR(L,1)<>0 THEN GOTO 7270
7250 SPARE2=0:FOR L2=0 TO L:SPARE1=TTP(L2):TTP(L2)=SPARE2-TTR(L,0)*SPARE1
7260 SPARE2=SPARE1:NEXT L2:GOTO 7230
7270 B=-2*TTR(L,0):C=TTR(L,0)^2+TTR(L,1)^2:SPARE2=0:SPARE3=0:L=L+1
7280 FOR L2=0 TO L:SPARE1=TTP(L2):TTP(L2)=C*SPARE1+B*SPARE2+SPARE3
7290 SPARE3=SPARE2:SPARE2=SPARE1:NEXT L2:GOTO 7230
7300 E=E+1:IF E>III THEN GOTO 7370
7310 IF IIR(E,1)<>0 THEN GOTO 7340
7320 SPARE2=0:FOR L2=0 TO E:SPARE1=IIP(L2):IIP(L2)=SPARE2-IIR(E,0)*SPARE1
7330 SPARE2=SPARE1:NEXT L2:GOTO 7300
7340 B=-2*IIR(E,0):C=IIR(E,0)^2+IIR(E,1)^2:SPARE2=0:SPARE3=0:E=E+1
7350 FOR L2=0 TO E:SPARE1=IIP(L2):IIP(L2)=C*SPARE1+B*SPARE2+SPARE3
7360 SPARE3=SPARE2:SPARE2=SPARE1:NEXT L2:GOTO 7300
7370 RETURN
8000 REM   ROOTS  OF  THE  POLY    '*-*-*-*-*-*-*-*-
8010 REM DIM ROOTS(N),QV(N),QA(N),QB(N),QC(N)
8020 N=DEGR:K=0
8030  IF POLY(K)<>0 THEN GOTO 8050
8040 K=K+1:GOTO 8030
8050  N=N-K
8060 FOR L1=0 TO DEGR:ROOTS(L1,0)=0:ROOTS(L1,1)=0:NEXT L1
8070 FOR L1=0 TO N:QV(L1)=POLY(L1+K):NEXT L1
8080  IF N<4 THEN GOTO  8240
8090 IT=0:HQ=((N-1)*QV(N-1))^2-2*N*(N-1)*QV(N)*QV(N-2):IF HQ<0 THEN GOTO 8130
8100 DET=QV(N-1)+SQR(HQ):IF DET=0 THEN DET=.00001
8110 PQ1=N*QV(N)/DET:DET=QV(N-1)-SQR(HQ):IF DET=0 THEN DET=.00001
8120 PQ2=N*QV(N)/DET:R=-(PQ1+PQ2):S=-PQ1*PQ2:GOTO 8160
8130  DET=(QV(N-1))^2+ABS(HQ)
8140 IF DET=0 THEN DET=.00001
8150 HG=N*QV(N)/DET:R=-2*HG*QV(N-1):S=-N*QV(N)*HG
8160  SPARE1=0:SPARE2=0:SPARE3=0:SPARE4=0:FOR L1=0 TO N
8170 QB(N-L1)=CDBL(QV(N-L1))+CDBL(R)*CDBL(SPARE1)+CDBL(S)*CDBL(SPARE2)
8172 SPARE2=CDBL(SPARE1):SPARE1=CDBL(QB(N-L1))
8180 QC(N-L1)=CDBL(QB(N-L1))+CDBL(R)*CDBL(SPARE3)+CDBL(S)*CDBL(SPARE4)
8182 SPARE4=CDBL(SPARE3):SPARE3=CDBL(QC(N-L1)):NEXT L1
8190 DET=CDBL(CDBL(QC(1))*CDBL(QC(3))-CDBL(QC(2))*CDBL(QC(2)))
8195 IF DET=0 THEN DET=1E-08
8200 DR=CDBL((CDBL(QB(1))*CDBL(QC(2))-CDBL(QB(0))*CDBL(QC(3)))/CDBL(DET))
8203 DS=CDBL((CDBL(QB(0))*CDBL(QC(2))-CDBL(QB(1))*CDBL(QC(1)))/CDBL(DET))
8210   IT=IT+1:IF IT>60 THEN GOTO 8230
8220 R=CDBL(R)+CDBL(DR):S=CDBL(S)+CDBL(DS)
8222 IF ABS(CDBL(DR))>1E-08 OR ABS(CDBL(DS))>1E-08 THEN GOTO 8160
8230 AA=1:FOR L1=0 TO N-2:QV(L1)=CDBL(QB(L1+2)):NEXT L1:GOTO 8410
8240  REM   3'rd   DEGREE  ROOTS
```

```
8250 IF N<3 THEN GOTO 8380
8260 AA=CDBL(QV(2))/CDBL(QV(3)):B=CDBL(QV(1))/CDBL(QV(3))
8265 CC=CDBL(QV(0))/CDBL(QV(3)):P=CDBL(B)-CDBL(CDBL(((CDBL(AA))^2#))/3#)
8270 Q=CDBL(CDBL(CC)-CDBL(AA)*CDBL(B)/3#+(2#*CDBL((CDBL(AA))^3#))/27#)
8275 US=CDBL(CDBL((CDBL(P)/3#))^3#)+CDBL((CDBL(Q)/2#)^2#)
8276 IF US<0 THEN GOTO 8320
8277 U=CDBL(SQR(CDBL(US))):VV=CDBL(CDBL(U)-CDBL(Q)/2#)
8278 V=SGN(VV)*CDBL((CDBL(ABS(VV)))^(1/3#)):WWW=-CDBL(U)-CDBL(Q)/2#
8290 W=SGN(WWW)*CDBL((CDBL(ABS(WWW)))^(1#/3))
8293 ROOTS(N,0)=CDBL(V)+CDBL(W)-CDBL(AA)/3#
8295 REAL=CDBL(-(CDBL(V)+CDBL(W))/2#-CDBL(AA)/3#)
8300 IMAG=CDBL(SQR(3)/2#)*CDBL(CDBL(V)-CDBL(W)):ROOTS(N-1,0)=CDBL(REAL)
8310 ROOTS(N-2,0)=CDBL(REAL):ROOTS(N-1,1)=CDBL(IMAG):ROOTS(N-2,1)=-CDBL(IMAG)
8315 GOTO 8360
8320  U=CDBL((CDBL((-CDBL(P)/3#)^3#))^(1#/2)):TE=-CDBL(Q)/2#/CDBL(U):PI=4*ATN(1)

8330 TE=CDBL(PI/2#-ATN(CDBL(TE)/CDBL(SQR(1-CDBL(TE)*CDBL(TE)+1E-10))))
8333 KC=CDBL(2#*CDBL(U^(1#/3)))
8340 ROOTS(N,0)=CDBL(CDBL(KC)*CDBL(COS(CDBL(TE/3#)))-CDBL(AA/3#))
8345 ROOTS(N-1,0)=CDBL(CDBL(KC)*CDBL(COS(CDBL(TE/3#-2#*PI/3#)))-CDBL(AA/3#))
8350 ROOTS(N-2,0)=CDBL(CDBL(KC)*CDBL(COS(CDBL(TE/3#+2#*PI/3#)))-CDBL(AA/3#))
8360  N=N-3
8370 GOTO 8510
8380  REM  2'nd  DEGREE  ROOTS
8390 IF N<2 THEN GOTO 8490
8400 AA=CDBL(QV(2)):R=-CDBL(QV(1)):S=-CDBL(QV(0))
8410 B=-CDBL(R):C=-CDBL(S):DELTA=CDBL(CDBL(B)^2#)-4#*CDBL(AA)*CDBL(C)
8420 IF DELTA<0 THEN GOTO 8450
8430 ROOTS(N,0)=-(CDBL(B)+CDBL(SQR(CDBL(DELTA))))/2#/CDBL(AA)
8435 ROOTS(N-1,0)=-(CDBL(B)-CDBL(SQR(CDBL(DELTA))))/2#/CDBL(AA)
8440 GOTO 8470
8450  REAL=-CDBL(B)/2#/CDBL(AA):IMAG=CDBL(SQR(ABS(CDBL(DELTA))))/2#/CDBL(AA)
8455 ROOTS(N  ,0)=CDBL(REAL):ROOTS(N  ,1)= IMAG
8460 ROOTS(N-1,0)=CDBL(REAL):ROOTS(N-1,1)=-IMAG
8470  N=N-2
8480 IF N>1 THEN GOTO 8380
8490  IF N<1 THEN GOTO 8510
8500 ROOTS(N,0)=-CDBL(QV(0))/CDBL(QV(1))
8510  RETURN
8520 FOR L=1 TO CDD:PRINT CDR(L,0);CDR(L,1);CDR(L,2):NEXT L
```

E - Root Locus Method

20-45	To get **G(z)**, **H(z)** and **G'(z)**.
50-70	To get **K** , **x** , **y** , **Φ** from the disk.
100-190	Definitions of arrays.
200-260	Calculation of **V(z)**
300-340	To find the roots of both numerators.
350-410	Draw the **z**-plane.
450-670	Input command.
1000-2115	Calculation of step response and over all closed-loop transfer function.
2200-2500	To display **C(z)** , **M(z)** and other command list to modify step response or improve K_v
3000-3400	Multiplication of two rational function
3410-3550	To display **C(z)** .
3600-3670	To input ζ and EPS.
4000-4420	To draw the frame.
4500-5060	To scaling.
5100-5120	To draw the frame.
5150-5170	To save the scales.
5200-5320	To move cursor.
6000-6050	About plotting the roots
6100-6250	To find the proper format.
6300-6320	About plotting of the roots.
6500-6620	To calculate closed-loop poles
6700-6890	To get pole/zero of **C(z)**
6900-7080	To get pole/zero of **H(z)**.
7100-7270	To calculate polynomial form of **G(z)** and **H(z)** denominators.
7500-7750	To Add pole/zero.
7800-7870	To draw new root locus
7900-7920	To cancel common factors of **C(z)**
8000-8350	To move along the root locus.
10000-10500	To display **C(z)** , **M(z)** and get the printout of them.
11000-11710	Root solving Subroutine

```
21 CLS:PI2=2*ATN(1):SAM=5:OS=1:JJ=5:KEY OFF:CX$="K"
23 DEF FNATAN(X,Y)=(1-SGN(ATN(Y/(X+1E-08))))*PI2+ATN(Y/(X+1E-08))
25 LOCATE 10,30:PRINT "..   PLEASE WAIT   ..":OPEN "I",#2,"ZTR1"
27 INPUT #2,GND:DIM GNU(GND):FOR L1=0 TO GND:INPUT #2,GNU(L1):NEXT L1
29 DIM GNR(GND,1):INPUT #2,GDD:DIM GDR(GDD,1):DIM GDE(GDD)
31 FOR L1=1 TO GDD:INPUT #2,GDR(L1,0),GDR(L1,1):NEXT L1
33 INPUT #2,HND:DIM HNU(HND):FOR L1=0 TO HND:INPUT #2,HNU(L1):NEXT L1
35 DIM HNR(HND,1):INPUT #2,HDD:DIM HDR(HDD,1):DIM HDE(HDD)
37 FOR L1=1 TO HDD:INPUT #2,HDR(L1,0),HDR(L1,1):NEXT L1
39 INPUT #2,VNG:DIM VNU(VNG):FOR L1=0 TO VNG:INPUT #2,VNU(L1):NEXT L1
41 INPUT #2,VDG:DIM VDE(VDG):FOR L1=0 TO VDG:INPUT #2,VDE(L1):NEXT L1
43 INPUT #2,NNN,HOLD,T,M,TC:CLOSE #2:MMM=INT(12/NNN)+INT(NNN/9)
45 BBB=1.5*CINT(300/MMM):D=HDD+GDD:GOSUB 7100:GOSUB 7200:SCREEN 2,0,0
50 REM >>>>>-----<<<<<
60 OPEN "I",#3,"WNDW":INPUT #3,RT,RD,EP,TRT,TND,TEP
70 INPUT #3,LEF,RIG,HS,LOW,UPP,VS:CLOSE #3:EPS=.0000001
100 REM >>>>>-----<<<<<
110 D=D+10
120 DIM POLY(D),        RR(D,1),        ROOTS( 52,2*D)
130 'DIM HNU(D),         HDE(D),         HNR(D,1),       HDR(D,1)
140 DIM YYP(D),         UUP(D),         YYR(D,2),       UUR(D,2)
150 DIM TTP(D),         IIP(D),         TTR(D,2),       IIR(D,2)
160 DIM CNU(10),        CDE(10),        CRN(10,1),      CRD(10,1)
170 'DIM A(200),         B(200),         C(200),         D(200)
180 DIM AA(NNN+2),      BB(NNN+2),      S(D),           E(200)
190 'DIM QV(D),          QA(D),          QB(D),          QC(D)
200 REM >>>>>-----<<<<<
210 AA(1)=-TC:AA(NNN+1)=1:NHN=NNN+1:IF HOLD <>1 THEN GOTO 230
220 FOR L=0 TO NHN:AA(NNN+1-L)=AA(NHN-L):NEXT L:NHN=NHN+1
230 DHD=NHN:BB(0)=-1:BB(NHN)=1:IF HOLD=1 THEN BB(0)=0
240 SP2=0:FOR L=0 TO NHN:SP1=BB(L):EB(L)=SP2-SP1*TC^(1/NNN):SP2=SP1:NEXT L
250 IF HOLD<>1 THEN GOTO 310
260 SP2=0:FOR L=0 TO DHD:SP1=BB(L):BB(L)=SP2-SP1:SP2=SP1:NEXT L
300 REM >>>>>-----<<<<<
310 GGN=GNU(GND):DEGR=GND:FOR L1=0 TO GND:POLY(L1)=GNU(L1)/GGN:NEXT L1
320 GOSUB 11000:FOR L=1 TO GND:GNR(L,0)=RR(L,0):GNR(L,1)=RR(L,1):NEXT L
330 HGN=HNU(HND):DEGR=HND:FOR L=0 TO HND:POLY(L)=HNU(L)/HGN:NEXT L:GOSUB 11000
340 FOR L1=1 TO HND:HNR(L1,0)=RR(L1,0):HNR(L1,1)=RR(L1,1):NEXT L1
350 REM >>>>>-----<<<<<
360 CLS:HDE(0)=1:CDG=0:CNG=0:CUN(0)=1:CED(0)=1:GOSUB 3600:GOSUB 4000
370 X=LEF*10^5:REM IF X<LEF OR X>RIG THEN X=INT(100000!*(LEF+(RIG-LEF)/2))
380 Y=LOW*10^5:REM IF Y<LOW OR Y>UPP THEN Y=INT(100000!*(LOW+(UPP-LOW)/2))
390 GOSUB 6500:IF Y<0 THEN Y=0
400 GOSUB 5400:GOSUB 5640:GOSUB 5770:CX$="K"
410 GOSUB 5400:LLL=RT:RRR=RD:SSS=EP:GOSUB 6100
```

```
450 REM >>>>>-----<<<<<
460 FXY$="X:###.#####  Y: ##.#####  "
470 LOCATE 1,2:PRINT USING FXY$+": #.#####  ";X*10^-5;Y*10^-5;J*10^-5
480 C$=INKEY$:IF C$="" THEN GOTO 480
490 IF C$<>"R" AND C$<>"D" AND C$<>"F" AND C$<>"C" AND C$<>"S" AND C$<>"R" AND
      C$<>"G" AND C$<>"O" AND C$<>"J" AND C$<>"P" AND C$<>"Z" AND C$<>"N" AND
      C$<>"L" AND C$<>"H" AND C$<>"I" AND C$<>"E" AND C$<>"Q" AND C$<>"M"
      THEN 480
495 '  Q* W  E* R* T  Y  U  I* O* P*
496 '   A  S* D* F* G* H* J* K  L*
497 '    Z* X  C* V  B* N* M*
500 IF C$="E" THEN RUN"MENU
505 IF C$="M" THEN GOSUB 5350:GOTO 470          'NEAREST DECIMAL POINT
510 IF C$="J" THEN JJ=(JJ+1) MOD 6:J=10^(5-JJ):GOTO 470
520 IF C$="B" THEN GOTO 360
530 IF C$="O" THEN GOSUB 4370:GOTO 470          'NEW ZETA CURVE
540 IF C$="N" THEN GOSUB 4500:GOTO 470          'RESCALE
550 IF C$="F" THEN GOSUB 5210:GOTO 470          'RIGHT +
560 IF C$="D" THEN GOSUB 5240:GOTO 470          'LEFT  -
570 IF C$="R" THEN GOSUB 5270:GOTO 470          'UP    !
580 IF C$="C" THEN GOSUB 5300:GOTO 470          'DOWN  +
590 IF C$="G" THEN GOSUB 6700:GOTO 470          'GET P/Z FROM C(z)
600 IF C$="L" THEN GOSUB 6900:GOTO 470          'LOCATION OF P/Z
610 IF C$="S" THEN GOSUB 7300:GOTO 470          'NEW R.L.
620 IF C$="Z" THEN GOSUB 7500:GOTO 470          'ADD ZERO
630 IF C$="P" THEN GOSUB 7700:GOTO 470          'ADD POLE
640 IF C$="I" THEN GOSUB 6300:GOTO 470          'REDRAW
650 IF C$="H" THEN GOSUB 3410                   'C(Z)
660 IF C$="Q" THEN GOTO 8000                    'ALONG R.L.
670 GOTO 480
1000 REM >>>>>-----<<<<<
1001 OPEN "O",#2,"ROOTS"
1002 KKK=0:FOR K=RT TO ND STEP EP:KKK=KKK+1:FOR L=1 TO CGHD
1003 WRITE #2,ROOTS(KKK,2*L-1),ROOTS(KKK,2*L):NEXT L:NEXT K:CLOSE #2
1010 ERASE ROOTS:DIM MNU(D),MDE(D),MNR(D,1),MDR(D,1),D(NNN*D+200)
1011             DIM A(NNN*D+200),B(NNN*D+200),C(NNN*D+200)
1015 FOR L=0 TO MDD:MNR(L,0)=0:MNR(L,1)=0:MNU(L)=0:NEXT L
1020 FOR L=0 TO MDD:MDR(L,0)=0:MDR(L,1)=0:MDE(L)=0:NEXT L
1030 FOR L=0 TO 200:A(L)=0:B(L)=0:C(L)=0:D(L)=0:E(L)=0:NEXT L
1100 REM >>>>>-----<<<<<
1110 LOCATE 1,54:PRINT "        IN  PROCESS        "
1120 FOR L=1 TO CNG:YYR(L,0)=CRN(L,0):YYR(L,1)=CRN(L,1):NEXT L
1130 FOR L=1 TO MND:LL=L+CNG:YYR(LL,0)=MNR(L,0):YYR(LL,1)=MNR(L,1):NEXT L
1140 FOR L=1 TO CDG:UUR(L,0)=CRD(L,0):UUR(L,1)=CRD(L,1):NEXT L
1150 FOR L=1 TO MDD:LL=L+CDG:UUR(LL,0)=MDR(L,0):UUR(LL,1)=MDR(L,1):NEXT L
1160 FOR L=1 TO GDD:LL=L+CDG+MDD:UUR(LL,0)=GDR(L,0):UUR(LL,1)=GDR(L,1):NEXT L
1170 YYY=CNG+MND:UUU=CDG+MDD+GDD:EPFS=.00001:GOSUB 3000:GOSUB 3200
1180 MND=TTT:FOR L=0 TO MND:MNU(L)=TTP(L):NEXT L
1190 MDD=III:FOR L=0 TO MDD:MDE(L)=IIP(L):NEXT L:DEGR=III
```

```
1200 FOR L=1 TO MDD:MDR(L,0)=IIR(L,0):MDR(L,1)=IIR(L,1):NEXT L:PMD=MDD
1210 FOR L=0 TO MDD:MNU(L)=MDE(L)+CGN*MGN*MNU(L):POLY(L)=MNU(L):NEXT L
1220 GOSUB 11000:FOR L=1 TO MDD:MNR(L,0)=RR(L,0):MNR(L,1)=RR(L,1):NEXT L
1250 REM >>>>>-----<<<<<
1260 FOR L=0 TO MND:MNU(L)=0:NEXT L:FOR L=0 TO MDD:MDE(L)=0:NEXT L
1270 FOR L=1 TO CNG:YYR(L,0)=CRN(L,0):YYR(L,1)=CRN(L,1):NEXT L
1280 FOR L=1 TO MDD:LL=L+CNG:YYR(LL,0)=MDR(L,0):YYR(LL,1)=MDR(L,1):NEXT L
1290 FOR L=1 TO CDG:UUR(L,0)=CRD(L,0):UUR(L,1)=CRD(L,1):NEXT L
1300 FOR L=1 TO MND:LL=L+CDG:UUR(LL,0)=MNR(L,0):UUR(LL,1)=MNR(L,1):NEXT L
1310 YYY=CNG+MDD:UUU=CDG+MND:EPPS=.0005:GOSUB 3000:GOSUB 3200
1320 MND=TTT:FOR L=1 TO MND:MNR(L,0)=TTR(L,0):MNR(L,1)=TTR(L,1):NEXT L
1330 MDD=III:FOR L=1 TO MDD:MDR(L,0)=IIR(L,0):MDR(L,1)=IIR(L,1):NEXT L
1340 FOR L=0 TO MND:MNU(L)=TTP(L):NEXT L:FOR L=0 TO MDD:MDE(L)=IIP(L):NEXT L
1350 PND=MND:PDD=MDD:FOR L=0 TO MDD:A(L*NNN)=CGN*MNU(L):B(L*NNN)=MDE(L):NEXT L
1400 REM >>>>>-----<<<<<
1410 FOR L=0 TO MND:MNU(L)=0:NEXT L:FOR L=0 TO MDD:MDE(L)=0:NEXT L
1420 FOR L=1 TO GND:YYR(L,0)=GNR(L,0):YYR(L,1)=GNR(L,1):NEXT L
1430 FOR L=1 TO MND:LL=L+GND:YYR(LL,0)=MNR(L,0):YYR(LL,1)=MNR(L,1):NEXT L
1440 FOR L=1 TO GDD:UUR(L,0)=GDR(L,0):UUR(L,1)=GDR(L,1):NEXT L
1450 FOR L=1 TO MDD:LL=L+GDD:UUR(LL,0)=MDR(L,0):UUR(LL,1)=MDR(L,1):NEXT L
1460 YYY=GND+MND:UUU=GDD+MDD:EPPS=.0005:GOSUB 3000:GOSUB 3200
1470 MND=TTT:FOR L=0 TO MND:MNU(L)=TTP(L):NEXT L
1480 MDD=III:FOR L=0 TO MDD:MDE(L)=IIP(L):NEXT L:MGN=CGN*GGN
1490 FOR L=1 TO MND:MNR(L,0)=TTR(L,0):MNR(L,1)=TTR(L,1):NEXT L
1500 FOR L=1 TO MDD:MDR(L,0)=IIR(L,0):MDR(L,1)=IIR(L,1):NEXT L
1550 REM >>>>>-----<<<<<
1560 SU1=0:FOR L=0 TO MND:SU1=SU1+L*MNU(L):NEXT L
1570 SU2=0:FOR L=0 TO MND:SU2=SU2+  MNU(L):NEXT L
1580 SU3=0:FOR L=0 TO MDD:SU3=SU3+L*MDE(L):NEXT L
1590 SU4=0:FOR L=0 TO MDD:SU4=SU4+  MDE(L):NEXT L
1600 IF MND=0 THEN SUM1=0 ELSE SUM1=SU1/SU2
1610 IF MDD=0 THEN SUM2=0 ELSE SUM2=SU3/SU4
1620 KV=SUM2-SUM1
1700 REM >>>>>-----<<<<<
1710 CCC=BBB:FOR L=1 TO BBB:C(L)=0:D(L)=0:E(L)=0:NEXT L:FOR L=0 TO MND
1720 C(L+1)=MGN*MNU(L):NEXT L:C(0)=0:FOR L=0 TO MDD:D(L)=MDE(L):NEXT L:S2=0
1730 FOR L=0 TO MDD+1:S1=D(L):D(L)=S2-S1:S2=S1:NEXT L:FOR L=0 TO BBB/NNN
1740 E(L)=C(MDD+1):FOR L2=1 TO MDD+1:C(MDD+2-L2)=C(MDD+1-L2)-E(L)*D(MDD+1-L2)
1750 NEXT L2:NEXT L:IF NNN=1 THEN FOR L=0 TO BBB:C(L)=E(L):NEXT L:GOTO 1970
1800 REM >>>>>-----<<<<<
1810 PPD=NNN*PDD:PPN=NNN*PND:FOR L=0 TO PPN+2*VDG:C(L)=0:NEXT L
1820 FOR L1=0 TO PPN:FOR L2=0 TO VNG:C(L1+L2)=A(L1)*VNU(L2)+C(L1+L2)
1830 NEXT L2:NEXT L1:PPN=PPN+VNG:FOR L=0 TO PPD+VNG:A(L)=C(L):C(L)=0:NEXT L
1840 FOR L1=0 TO PPD:FOR L2=0 TO VDG:C(L1+L2)=B(L1)*VDE(L2)+C(L1+L2)
1850 NEXT L2:NEXT L1:PPD=PPD+VDG:FOR L=0 TO PPN+VDG:B(L)=C(L):C(L)=0:NEXT L
1860 FOR L1=0 TO PPN:FOR L2=0 TO NHN:C(L1+L2)=A(L1)*AA(L2)+C(L1+L2)
1870 NEXT L2:NEXT L1:PHN=PPN+NHN:FOR L=0 TO PPD+NHN:A(L)=C(L):C(L)=0:NEXT L
1880 FOR L1=0 TO PPD:FOR L2=0 TO DHD:C(L1+L2)=B(L1)*BP(L2)+C(L1+L2)
1890 NEXT L2:NEXT L1:DDD=PPD+DHD:FOR L=0 TO DDD:B(L)=C(L):C(L)=0:NEXT L
```

```
1900 REM >>>>>-----<<<<<
1910 OS=1:CCC=BBB:FOR L1=0 TO BBB
1920 IF NNN*INT(L1/NNN)<>L1 OR A(DDD)=0 OR A(DDD)=E(DDD) THEN GOTO 1940
1930 FOR L=0 TO DDD:A(L)=E(INT(L1/NNN))/A(DDD)*A(L):NEXT L
1940 C(L1)=A(DDD):IF C(L1)>OS THEN OS=C(L1):IF OS>9 THEN 2010
1950 FOR  L2=1 TO DDD:A(DDD+1-L2)=A(DDD-L2)-C(L1)*B(DDD-L2):NEXT L2:NEXT L1
1960 IF CCC=BBB THEN GOTO 2010
1970 OS=0:FOR L=0 TO CCC:IF C(L)=>OS THEN OS=C(L):NEXT L
1980 IF OS<1 THEN OS=1
2000 REM >>>>>-----<<<<<
2010 CLS:O=CINT(10*OS)+1:H=187:HH=1:HHH=198:N=0:NN=296
2020 M=10*INT(H/O):FOR L=1 TO CCC:XTRM=N+MMM*L
2030 LINE (XTRM-MMM,H+HH-M*C(L-1))-(XTRM,H+HH-M*C(L)):NEXT L:XTRM=N+MMM*BBB
2040 FOR L=0 TO BBB STEP NNN:P=N+MMM*L:LINE (P,HHH-1)-(P,HHH+1):NEXT L
2050 FOR L=N TO NN:IF POINT(L,HHH-10-M)<>1 THEN NEXT L
2060 RIT=(L-N)/NNN/MMM:LINE(N,H+HH)-(XTRM,H+HH)
2070 LINE(N,HHH-10-M)-(N+MMM*BBB,HHH-10-M):LINE(N,HH)-(XTRM,HHH),,B
2080 LOCATE 21,40:PRINT USING" R.T.:###.###";RIT:LOCATE 22,40
2090 IF ABS(KV)>99 THEN PRINT USING" 1/Kv:###.^^^^";KV :GOTO 2110
2100 PRINT USING" 1/Kv:###.###";KV
2110 LOCATE 23,40:PRINT USING" O.S.:##%.###";100*(OS-1)
2115 IF KVORG<>1 THEN KVRGN=KV:KVORG=1
2200 REM >>>>>-----<<<<<
2210 C$=INKEY$:IF C$="" THEN GOTO 2210
2220 IF C$<>"B" AND C$<>"P" AND C$<>"E" AND C$<>"I" AND
       C$<>"N" AND C$<>"O" AND C$<>"C"              THEN GOTO 2210
2230 IF C$="E" THEN RUN"MENU.BAS
2240 IF KKVV=1 AND C$="N" THEN CNG=CNG-1:CDG=CDG-1:KKVV=0:GOTO 1000
2250 IF C$<>"B" THEN GOTO 2260
2253 CLS:CNG=CNG-1:CDG=CDG-1:ERASE MNU,MDE,MNR,MDR,A,B,C,D
2254 DIM ROOTS(52,2*D):QW=0:FOR K=RT TO ND STEP EP:QW=QW+1:FOR L=1 TO CGHD
2255 WRITE #2,ROOTS(QW,2*L-1),ROOTS(QW,2*L):NEXT L:NEXT K:GOSUB 6300:GOTO 8020
2260 IF C$="O" THEN JJ=0:GOTO 10000
2265 IF C$="P" THEN JJ=1:GOTO 10000
2270 IF C$<>"C" THEN GOTO 2300
2280 LOCATE 3,60:INPUT " CUT FROM  =":CCC
2290 IF CCC>BBB THEN GOTO 2210 ELSE GOTO 1970
2300 IF C$<>"I" THEN GOTO 2220
2350 REM >>>>>-----<<<<<
2360 LOCATE 3,60:PRINT " 1/Kv  DESIRED =":LOCATE 4,61:INPUT KVD
2370 AD$="("+CHR$(25)+")":JJ=.001:ZR=.97:GOTO 2460
2380 C$=INKEY$:IF C$="" THEN 2380
2390 IF C$<>"K" AND C$<>"L" AND C$<>"J" AND C$<>"S" THEN 2380
2400 IF C$ ="S" THEN GOTO 2500
2410 IF C$<>"J" THEN GOTO 2440
2420 IF JJ=.01  THEN JJ=.001:AD$="("+CHR$(25)+"1":GOTO 2460
2430 IF JJ=.001 THEN JJ=.01 :AD$="("+CHR$(24)+")":GOTO 2460
```

```
2440 IF C$="K" AND ABS(ZR-JJ)<.999 THEN ZR=ZR-JJ
2450 IF C$="L" AND ABS(ZR+JJ)<.999 THEN ZR=ZR+JJ
2460 LOCATE 8,60:PRINT " DIPOLE "+AD$:PL=1-KVD/KV*(1-ZR)
2470 LOCATE 9,60:PRINT USING "ZR: #.###      ";ZR
2480 LOCATE 10,60:PRINT USING "PL: #.########";PL
2490 GOTO 2390
2500 IF KKVV=0 THEN CNG=CNG+1:CDG=CDG+1:KKVV=1
2510 CRN(CNG,0)=ZR:CRN(CNG,1)=0:CRD(CDG,0)=PL:CRD(CDG,1)=0:GOTO  1000
3000 REM >>>>>-----<<<<<
3010 FOR L=1 TO YYY:YYR(L,2)=0:NEXT L:FOR L=1 TO UUU:UUR(L,2)=0:NEXT L
3020 IF YYY=0 OR UUU=0 THEN 3080
3030 FOR L1=1 TO YYY:FOR L2=1 TO UUU
3040 IF ABS(YYR(L1,0)-UUR(L2,0))>EPPS OR ABS(YYR(L1,1)-UUR(L2,1))>EPPS OR
          YYR(L1,2)=1 OR UUR(L2,2)=1 THEN 3060
3050 YYR(L1,2)=1:UUR(L2,2)=1:GOTO 3070
3060 NEXT L2
3070 NEXT L1
3080 TTT=0:IF YYY=0 THEN 3120
3090 FOR L=1 TO YYY:IF YYR(L,2)=1 THEN GOTO 3110
3100 TTT=TTT+1:TTR(TTT,0)=YYR(L,0):TTR(TTT,1)=YYR(L,1)
3110 NEXT L
3120 III=0:IF UUU=0 THEN 3160
3130 FOR L=1 TO UUU:IF UUR(L,2)=1 THEN GOTO 3150
3140 III=III+1:IIR(III,0)=UUR(L,0):IIR(III,1)=UUR(L,1)
3150 NEXT L
3160 RETURN
3200 REM >>>>>-----<<<<<
3210 FOR L=0 TO TTT:TTP(L)=0:NEXT L:FOR L=0 TO III:IIP(L)=0:NEXT L
3220 L=0:E=0:TTP(0)=1:IIP(0)=1
3230 L=L+1:IF L>TTT THEN GOTO 3300
3240 IF TTR(L,1)<>0 THEN GOTO 3270
3250 SPARE2=0:FOR L2=0 TO L:SPARE1=TTP(L2):TTP(L2)=SPARE2-TTR(L,0)*SPARE1
3260 SPARE2=SPARE1:NEXT L2:GOTO 3230
3270 B=-2*TTR(L,0):C=(TTR(L,0))^2+(TTR(L,1))^2:SPARE2=0:SPARE3=0:L=L+1
3280 FOR L2=0 TO L:SPARE1=TTP(L2):TTP(L2)=C*SPARE1+B*SPARE2+SPARE3
3290 SPARE3=SPARE2:SPARE2=SPARE1:NEXT L2:GOTO 3230
3300 E=E+1:IF E>III THEN GOTO 3400
3310 IF IIR(E,1)<>0 THEN GOTO 3340
3320 SPARE2=0:FOR L2=0 TO E:SPARE1=IIP(L2):IIP(L2)=SPARE2-IIR(E,0)*SPARE1
3330 SPARE2=SPARE1:NEXT L2:GOTO 3300
3340 B=-2*IIR(E,0):C=(IIR(E,0))^2+(IIR(E,1))^2:SPARE2=0:SPARE3=0:E=E+1
3350 FOR L2=0 TO E:SPARE1=IIP(L2):IIP(L2)=C*SPARE1+B*SPARE2+SPARE3
3360 SPARE3=SPARE2:SPARE2=SPARE1:NEXT L2:GOTO 3300
3400 RETURN
```

```
3410 REM >>>>>-----<<<<<
3420 IF H=1 THEN H=0:LINE (85,24)-(560,24+8*(CDG+2)),0,BF:GOSUB 6300:RETURN
3430 IF CDG=0 THEN GOSUB 6300:RETURN
3440 LINE (85,24)-(560,24+8*(CDG+2)),0,BF:LOCATE 4,12
3450 PRINT "╔═══════════════════════════════════════════════════════╗"
3460 LOCATE 5+CDG,12
3470 PRINT "╚═══════════════════════════════════════════════════════╝"
3480 FOR L=1 TO CDG:LOCATE 4+L,12:PRINT "║":LOCATE 4+L,70:PRINT "║":NEXT L
3490 FOR L=1 TO CNG:F$="":LRS=CRN(L,0):GOSUB 6160:F$=F2$:LRS=CRN(L,1)
3500 GOSUB 6160:F$=F$+" + j "+F2$:LOCATE 4+L,13
3510 PRINT USING " CN(##)="+F$;L;CRN(L,0);CRN(L,1):NEXT L
3520 FOR L=1 TO CDG:F$="":LRS=CRD(L,0):GOSUB 6160:F$=F2$:LRS=CRD(L,1)
3530 GOSUB 6160:F$=F$+" + j "+F2$:LOCATE 4+L,42
3540 PRINT USING " CD(##)="+F$;L;CRD(L,0);CRD(L,1):NEXT L
3550 H=1:RETURN
3600 REM >>>>>-----<<<<<
3610 LOCATE 1,5:INPUT " ZETA:";ZT
3620 IF ZT<0 OR ZT>1 THEN GOTO 3610
3630 LOCATE 3,5:PRINT "  EPS:1E-8":LOCATE 3,20:PRINT " ( ,N) "
3640 C$=INKEY$:IF C$="" THEN GOTO 3640
3650 IF C$<>"N" THEN LOCATE 3,21:PRINT "O.K.  ":GOTO 3670
3660 LOCATE 3,20:INPUT "  EPS:";EPS
3670 RETURN
4000 REM >>>>>--- FRAME & ZETA CURVE & CIRCLE ---<<<<<
4010 CLS:LINE (0,190)-(639,200),0,BF
4020 XN=CINT((RIG-LEF)/HS)+2:YN=CINT((UPP-LOW)/VS)+2
4030 XNN=INT(615/XN)  :YNN=INT(180/YN)  :XM=(185-YNN*YN)/2:HM=(639-XN*XNN)/2
4040 F2X1=INT(HM):F2X2=F2X1+XN*XNN:F2Y1=13        :F2Y2=14+YN*YNN
4050 F1X1=F2X1-12:F1X2=F2X2+12    :F1Y1=F2Y1-5    :F1Y2=F2Y2+5
4060 LINE (F1X1,F1Y1)-(F1X2,F1Y2),,B
4070 LINE (F2X1,F2Y1)-(F2X2,F2Y2),,B
4080 LINE (F1X1,F1Y1)-(F2X1,F2Y1):LINE (F1X1,F1Y2)-(F2Y1,F2Y2)
4090 LINE (F1X2,F1Y2)-(F2X2,F2Y2):LINE (F1X2,F1Y1)-(F2Y2,F2Y1)
4100 XUN=XNN/HS:YUN=YNN/VS:XRF=F2X1-(LEF-HS)*XUN:YRF=F2Y2+(LOW-VS)*YUN
4110 IF XRF>F2X1 AND XRF<F2X2 THEN LINE (XRF,F2Y1)-(XRF,F2Y2)
4120 IF YRF>F2Y1 AND YRF<F2Y2 THEN LINE (F2X1,YRF)-(F2X2,YRF)
4130 FOR L=LEF TO RIG+HS STEP HS:PSET(L*XUN+XRF,F2Y1+1)
4140 PSET(L*YUN+XRF,F2Y2-1):NEXT L:FOR L=LOW TO UPP+VS STEP VS
4150 LINE (F2X1,YRF-L*YUN)-(F2X1+2,YRF-L*YUN)
4160 LINE (F2X2,YRF-L*YUN)-(F2X2-2,YRF-L*YUN):NEXT L
4170 GOSUB 4390:FOR ANG=TRT TO TND+TEP STEP TEP
4180 XX=XRF+XUN*COS(ANG):YY=YRF-YUN*SIN(ANG)
4190 IF F2X1<=XX AND XX<=F2X2 AND F2Y2=>YY AND YY=>F2Y1 THEN PSET (XX,YY)
4200 NEXT ANG
4210 FOR L=1 TO CNG:XCR=CRN(L,0):YCR=CRN(L,1):GOSUB 4270:GOSUB 4320:NEXT L
4220 FOR L=1 TO HND:XCR=HNR(L,0):YCR=HNR(L,1):GOSUB 4270:GOSUB 4220:NEXT L
4230 FOR L=1 TO CDG:XCR=CRD(L,0):YCR=CRD(L,1):GOSUB 4270:GOSUB 4330:NEXT L
4240 FOR L=1 TO HDD:XCR=HDR(L,0):YCR=HDR(L,1):GOSUB 4270:GOSUB 4330:NEXT L
4250 FOR L=1 TO GDD:XCR=GDR(L,0):YCR=GDR(L,1):GOSUB 4270:GOSUB 4330:NEXT L
4260 RETURN
```

```
4270 XCR=XCR*XUN+XRF:IF XCR>F2X2 THEN XCR=F2X2+12
4280 IF XCR<F2X1 THEN XCR=F2X1-12
4290 YCR=YRF-YCR*YUN:IF YCR>F2Y2 THEN YCR=F2Y2+6
4300 IF YCR<F2Y1 THEN YCR=F2Y1-3
4310 RETURN
4315 REM >>>>>--- SUPPORTING SUBROUTINES FOR 4000 ---<<<<<
4320 CIRCLE (XCR,YCR),3,,,,.8:CIRCLE (XCR,F2Y2+3),3,,,,.8:RETURN
4330 LINE (XCR-3,YCR  )-(XCR+3,YCR  )  :LINE (XCR-1,YCR-1)-(XCR-1,YCR+1)
4340 LINE (XCR  ,YCR-2)-(XCR  ,YCR+2)  :LINE (XCR+1,YCR-1)-(XCR+1,YCR+1)
4350 LINE (XCR-2,F2Y2+1)-(XCR+2,F2Y2+1):LINE (XCR,F2Y2)-(XCR,F2Y2+4)
4360 LINE (XCR-2,F2Y2+2)-(XCR+2,F2Y2+2):RETURN
4370 LINE (0,0)-(320,7),0,BF:LOCATE 1,5:INPUT "ZETA=";ZT
4380 IF ZT>1 OR ZT<0 THEN GOTO 4370
4390 FOR ANG=TRT TO TND+TEP STEP TEP
4400 R=EXP(-ZT*ANG):F=SQR(1-ZT^2)*ANG:XX=XRF+XUN*R*COS(F):YY=YRF-YUN*R*SIN(F)
4410 IF F2X1<=XX AND XX<=F2X2 AND F2Y2=>YY AND YY=>F2Y1 THEN PSET (XX,YY)
4420 NEXT ANG:RETURN
4500 REM >>>>>--- X/Y/K/F ---<<<<<
4510 LOCATE 1,54:PRINT "   RESCALE  X/Y/K/F ( ^ )  "
4520 CX$=INKEY$:IF C$="" THEN GOTO 4520
4530 IF CX$<>"X" AND CX$<>"Y" AND CX$<>"K" AND CX$<>"F" AND
       CX$<>"x" AND CX$<>"y" AND CX$<>"k" AND CX$<>"f" THEN 4520
4540 IF CX$="x" THEN CX$="X":LLL=LEF:RRR=RIG:SSS=HS:GOSUB 6100:RETURN
4550 IF CX$="y" THEN CX$="Y":LLL=LOW:RRR=UPP:SSS=VS:GOSUB 6100:RETURN
4560 IF CX$="k" THEN CX$="K":LLL= RT:RRR= ND:SSS=EP:GOSUB 6100:RETURN
4580 IF CX$="f" THEN CX$="F":W=90/PI2:LLL=TRT*W:RRR=TND*W:SSS=TEP*90*PI2:CX$="▒"
                :GOSUB 6100:RETURN
4590 IF CX$="X" THEN GOSUB 4650:RETURN
4600 IF CX$="Y" THEN GOSUB 4750:RETURN
4610 IF CX$="K" THEN GOSUB 4700:RETURN
4620 IF CX$="F" THEN CX$="▒":GOSUB 5000:RETURN
4650 REM >>>>>--- X ---<<<<<
4660 LINE (432,0)-(639,7),0,BF
4670 LOCATE 1,54:INPUT "  X - SMALL :";LEF:LINE (540,0)-(639,7),0,BF
4680 LOCATE 1,54:INPUT "  X - BIG   :";RIG:LINE (540,0)-(639,7),0,BF
4690 LEF=10^-4*INT(10^4*LEF):RIG=10^-4*INT(10^4*RIG)
4700 IF ABS(LEF-RIG)<=.0001 THEN HS=10^-5:GOTO 4740
4710 IF LEF=>RIG THEN GOTO 4670
4720 LOCATE 1,54:INPUT "  X - SCALE :";HS:HS=ABS(HS)
4730 IF HS>RIG-LEF OR INT((RIG-LEF)/HS)>30 OR HS=0 THEN GOTO 4720
4740 LLL=LEF:RRR=RIG:SSS= HS:GOSUB 5100:RETURN
4750 REM >>>>>--- Y ---<<<<<
4760 LINE (432,0)-(639,7),0,BF
4770 LOCATE 1,54:INPUT "  Y - SMALL :";LOW:LINE (540,0)-(639,7),0,BF
4780 LOCATE 1,54:INPUT "  Y - BIG   :";UPP:LINE (540,0)-(639,7),0,BF
4790 LOW=10^-4*INT(10^4*LOW):UPP=10^-4*INT(10^4*UPP)
4800 IF ABS(LOW)>ABS(UPP) THEN SWAP LOW,UPP
4810 IF ABS(LOW-UPP)=.0001 THEN VS=10^-5:GOTO 4850
4820 IF LOW=>UPP THEN GOTO 4770
4830 LOCATE 1,54:INPUT "  Y - SCALE :";VS:VS=ABS(VS)
4840 IF VS>UPP-LOW OR INT((UPP-LOW)/VS)>30 OR VS=0 THEN GOTO 4830
4850 LLL=LOW:RRR=UPP:SSS= VS:GOSUB 5100:RETURN
```

```
4900 REM >>>>>--- K ---<<<<<
4910 LINE (432,0)-(639,7),0,BF
4920 LOCATE 1,54:INPUT "  K - START :";RT :LINE (540,0)-(639,7),0,BF
4930 LOCATE 1,54:INPUT "  K - END   :";ND :LINE (540,0)-(639,7),0,BF
4940 IF RT=>ND THEN GOTO 4910
4950 LOCATE 1,54:INPUT "  K - STEP  :";EP
4960 IF CINT((ND-RT)/EP)+1> 51 OR EP=0 THEN GOTO 4950
4970 LLL= RT:RRR= ND:SSS= EP:GOSUB 6500:GOSUB 6100:GOSUB 5150:RETURN
5000 REM >>>>>--- ∑ ---<<<<<
5010 LINE (432,0)-(639,7),0,BF
5020 LOCATE 1,54:INPUT "  ∑ - START :";TRT:LINE (540,0)-(639,7),0,BF
5030 LOCATE 1,54:INPUT "  ∑ - END   :";TND
5040 IF TRT=>TND OR TND>180 OR TRT<0 THEN GOTO 5010
5050 LLL=TRT:RRR=TND:SSS=(TND-TRT)/75:TRT=TRT/45*ATN(1)
5060 TND=TND/45*ATN(1):TEP=(TND-TRT)/75:GOSUB 5150:GOSUB 4170:RETURN
5100 REM >>>>>--- DRAW ---<<<<<
5110 CLS:GOSUB 6300:GOSUB 5400:GOSUB 5640:GOSUB 5770
5120 GOSUB 5150:RETURN
5150 REM >>>>>--- SAVE ---<<<<<
5160 OPEN "O",#3,"WNDW":WRITE #3,RT,ND,EP,TRT,TND,TEP
5170 WRITE #3,LEF,RIG,HS,LOW,UPP,VS:CLOSE #3:GOSUB 6100:RETURN
5200 REM >>>>>--- LEFT-RIGHT-UP-DOWN ---<<<<<
5210 IF X+J=>100000!*RIG+1 THEN GOTO 5230
5220 GOSUB 5570:X=INT(X+J):GOSUB 5400:GOSUB 5640
5230 RETURN
5240 IF X-J<=100000!*LEF-1 THEN GOTO 5260
5250 GOSUB 5570:X=INT(X-J):GOSUB 5400:GOSUB 5640
5260 RETURN
5270 IF Y+J=>100000!*UPP+1 THEN GOTO 5290
5280 GOSUB 5710:Y=INT(Y+J):GOSUB 5400:GOSUB 5770
5290 RETURN
5300 IF Y-J<=100000!*LOW-1 OR 10^-5*(Y-J)<LOW OR Y-J<0 THEN GOTO 5320
5310 GOSUB 5710:Y=INT(Y-J):GOSUB 5400:GOSUB 5770
5320 RETURN
5350 REM >>>>>--- X,Y-POINT ---<<<<<
5360 GOSUB 5570:GOSUB 5710:X=CINT(X/100001*10000:Y=CINT(Y/10000)*10000
5370 GOSUB 5400:GOSUB 5640:GOSUB 5770:RETURN
5400 REM >>>>>--- X,Y-POINT ---<<<<<
5410 GOSUB 5420:GOSUB 5500:RETURN
5420 REM >>>>>--- RESET ---<<<<<
5430 IF ABS(XF)>10000 OR ABS(YF)>10000 THEN RETURN
5440 IF F2X1=>XF AND XF=>F2X2 AND F2Y1=>YF AND YF=>F2Y2 THEN RETURN
5450 PRESET (XF-7,YF-3):PRESET(XF-5,YF-2):PRESET(XF-3,YF-1)
5460 PRESET (XF+7,YF-3):PRESET(XF+5,YF-2):PRESET(XF+3,YF-1)
5470 PRESET (XF-7,YF+3):PRESET(XF-5,YF+2):PRESET(XF-3,YF+1)
5480 PRESET (XF+7,YF+3):PRESET(XF+5,YF+2):PRESET(XF+3,YF+1):PRESET(XF,YF)
5490 XF=X*10^-5*XUN+XRF:YF=YRF-Y*10^-5*YUN:RETURN
```

```
5500 REM >>>>>--- SET ---<<<<<
5510 IF ABS(XF)>10000 OR ABS(YF)>10000  THEN RETURN
5520 IF F2X1=>XF AND XF=>F2X2 AND F2Y1=>YF AND YF=>F2X2 THEN RETURN
5530   PSET (XF-7,YF-3):  PSET(XF-5,YF-2):  PSET(XF-3,YF-1)
5540   PSET (XF+7,YF-3):  PSET(XF+5,YF-2):  PSET(XF+3,YF-1)
5550   PSET (XF-7,YF+3):  PSET(XF-5,YF+2):  PSET(XF-3,YF+1):  PSET(XF,YF)
5560   PSET (XF+7,YF+3):  PSET(XF+5,YF+2):  PSET(XF+3,YF+1):  RETURN
5570 REM >>>>>--- RESET X-POINT ---<<<<<
5580 IF ABS(XF)>10000 OR  F2X1=>XF AND XF=>F2X2 THEN RETURN
5590 LINE (XF  ,F2Y1-1)-(XF  ,F2Y1-4),0: LINE (XF  ,F2Y2+1)-(XF  ,F2Y2+4),0
5600 LINE (XF-1,F2Y1-4)-(XF-1,F2Y1-3),0: LINE (XF+1,F2Y1-4)-(XF+1,F2Y1-3),0
5610 LINE (XF-1,F2Y2+4)-(XF-1,F2Y2+3),0: LINE (XF+1,F2Y2+4)-(XF+1,F2Y2+3),0
5620 PRESET (XF-2,F2Y1-4)              : PRESET (XF+2,F2Y1-4)
5630 PRESET (XF-2,F2Y2+4)              : PRESET (XF+2,F2Y2+4)       :RETURN
5640 REM >>>>>--- SET X-POINT ---<<<<<
5650 IF ABS(XF)>10000 OR F2X1=>XF AND XF=>F2X2 THEN RETURN
5660 LINE (XF  ,F2Y1-1)-(XF  ,F2Y1-4)  : LINE (XF  ,F2Y2+1)-(XF  ,F2Y2+4)
5670 LINE (XF-1,F2Y1-4)-(XF-1,F2Y1-3)  : LINE (XF+1,F2Y1-4)-(XF+1,F2Y1-3)
5680 LINE (XF-1,F2Y2+4)-(XF-1,F2Y2+3)  : LINE (XF+1,F2Y2+4)-(XF+1,F2Y2+3)
5690 PSET   (XF-2,F2Y1-4)              : PSET   (XF+2,F2Y1-4)
5700 PSET   (XF-2,F2Y2+4)              : PSET   (XF+2,F2Y2+4)       :RETURN
5710 REM >>>>>--- RESET Y-POINT ---<<<<<
5720 IF ABS(YF)>10000 OR F2Y1=>YF AND YF=>F2Y2 THEN RETURN
5730 LINE (F2X1-1,YF  )-(F2X1-10,YF  ),0:LINE (F2X2+1,YF  )-(F2X2+10,YF  ),0
5740 LINE (F2X1-6,YF-1)-(F2X1-10,YF-1),0:LINE (F2X2+6,YF-1)-(F2X2+10,YF-1),0
5750 LINE (F2X1-6,YF+1)-(F2X1-10,YF+1),0:LINE (F2X2+6,YF+1)-(F2X2+10,YF+1),0
5760 RETURN
5770 REM >>>>>--- SET Y-POINT ---<<<<<
5780 IF ABS(YF)>10000 OR F2Y1=>YF AND YF=>F2Y2 THEN RETURN
5790 LINE (F2X1-1,YF  )-(F2X1-10,YF  )  :LINE (F2X2+1,YF  )-(F2X2+10,YF  )
5800 LINE (F2X1-6,YF-1)-(F2X1-10,YF-1)  :LINE (F2X2+6,YF-1)-(F2X2+10,YF-1)
5810 LINE (F2X1-6,YF+1)-(F2X1-10,YF+1)  :LINE (F2X2+6,YF+1)-(F2X2+10,YF+1)
5820 RETURN
6000 REM >>>>>--- PLOT OF ROOTS ---<<<<<
6010 IF ABS(XCR*XUN+XRF)>10000 OR ABS(YRF-YCR*YUN)>10000 THEN RETURN
6020 IF F2X1>XCR*XUN+XRF OR XCR*XUN+XRF>F2X2 THEN RETURN
6030 IF F2Y2<YRF-YUN*YCR OR YRF-YUN*YCR<F2Y1 THEN RETURN
6040 LINE (XCR*XUN+XRF,YRF-YCR*YUN-1)-(XCR*XUN+XRF,YRF-YCR*YUN+1)
6050 LINE (XCR*XUN+XRF-2,YRF-YCR*YUN)-(XCR*XUN+XRF+2,YRF-YCR*YUN):RETURN
6100 REM >>>>>------<<<<<
6110 LRS=LLL :GOSUB 6160:F1$=F2$+"/"
6120 LRS=RRR :GOSUB 6160:F1$=F1$+F2$+"|"
6130 LRS=SSS :GOSUB 6160:F1$=F1$+F2$
6140 LOCATE 1,45:PRINT "                                "
6150 LOCATE 1,54:PRINT USING " &: "+F1$;CX$;LLL;RRR;SSS:RETURN
```

```
6160 REM >>>>>------<<<<<
6170 IF ABS(LRS)=0                                  THEN F2$="   #   ":RETURN
6180 IF ABS(LRS)=>0       AND ABS(LRS)<1            THEN F2$="#.#####":RETURN
6190 IF ABS(LRS)=>1       AND ABS(LRS)<10           THEN F2$="##.####":RETURN
6200 IF ABS(LRS)=>10      AND ABS(LRS)<100          THEN F2$="###.###":RETURN
6210 IF ABS(LRS)=>100     AND ABS(LRS)<1000         THEN F2$="####.##":RETURN
6220 IF ABS(LRS)=>1000    AND ABS(LRS)<10000        THEN F2$="#####.#":RETURN
6230 IF ABS(LRS)=>10000   AND ABS(LRS)<100000!      THEN F2$=" ######":RETURN
6240 IF ABS(LRS)=>100000! AND ABS(LRS)<1000000!     THEN F2$="#######":RETURN
6250 IF ABS(LRS)=>1000000!                          THEN F2$="###^^^^":RETURN
6300 REM >>>>>--- PLOT ROOTS ---<<<<<
6305 KVORG=0:IF KKVV=1 THEN CNG=CNG-1:CDG=CDG-1:FIVV=0
6310 GOSUB 4020:KKK=0:FOR K=RT TO ND STEP EP:KKK=KKK+1:FOR L=1 TO CGHD
6320 XCR=ROOTS(KKK,2*L-1):YCR=ROOTS(KKK,2*L):GOSUB 6000:NEXT L:NEXT K:RETURN
6500 REM >>>>>--- ROOTS ---<<<<<
6510 FOR L=1 TO CNG:YYR(L,0)=CRN(L,0):YYR(L,1)=CRN(L,1):NEXT L
6520 FOR L=1 TO HND:LL=L+CNG:YYR(LL,0)=HNR(L,0):YYR(LL,1)=HNR(L,1):NEXT L
6530 FOR L=1 TO CDG:UUR(L,0)=CRD(L,0):UUR(L,1)=CRD(L,1):NEXT L
6540 FOR L=1 TO HDD:LL=L+CDG:UUR(LL,0)=HDR(L,0):UUR(LL,1)=HDR(L,1):NEXT L
6550 FOR L=1 TO GDD:LL=L+CDG+HDD:UUR(LL,0)=GDR(L,0):UUR(LL,1)=GDR(L,1):NEXT L
6560 YYY=CNG+HND:UUU=CDG+HDD+GDD:EPPS=.00001:GOSUB 3000:GOSUB 3200
6570 PDD=III:FOR L=0 TO PDD:PDE(L)=IIP(L):PNU(L)=0:NEXT L
6580 PND=TTT:FOR L=0 TO PND:PNU(L)=TTP(L):NEXT L
6590 KK=0:CGHD=PDD:DEGR=PDD:FOR KG=RT TO ND STEP EP:KK=KK+1:FOR L1=0 TO PDD
6600 POLY(L1)=PDE*L1)*KS*HSN*PNU(L1):NEXT L1:GOSUB 11000:FOR L2=1 TO PDD
6610 ROOTS(KK,2*L2-1)=RR(L2,0):ROOTS(KK,2*L2)=RR(L2,1)
6620 XCR=RR(L2,0):YCR=RR(L2,1):GOSUB 6000:NEXT L2:NEXT KG:RETURN
6700 REM >>>>>--- P/Z OF C(z) ---<<<<<
6710 LINE (0,0)-(320,7),0,BF
6720 LOCATE 1,3:PRINT "   C :POLE / ZERO (P/Z/ )   "
6730 C$=INKEY$:IF C$="" OR C$<>"Z" AND C$<>"P" AND C$<>" " THEN GOTO 6730
6740 IF C$=" " THEN RETURN
6750 DIS=10000:IF C$="P" THEN 6830
6760 IF CNG=0 THEN RETURN
6770 FOR L=1 TO CNG:DSS=((X*10^-5-CRN(L,0))^2+(Y*10^-5-CRN(L,1))^2)^(.5)
6780 IF DIS=>DSS THEN DIS=DSS:XCL=CRN(L,0):YCL=CRN(L,1):CLL=L
6790 NEXT L:LOCATE 1,3:PRINT USING FXY$+" (Y/ ) ";XCL;YCL
6800 C$=INKEY$:IF C$="" THEN GOTO 6800
6810 IF C$<>"Y" THEN RETURN
6820 X=INT(10^5*XCL):Y=INT(10^5*YCL):GOSUB 5400:RETURN
6830 IF CDG=0 THEN RETURN
6840 FOR L=1 TO CDG:DSS=((X*10^-5-CRD(L,0))^2+(Y*10^-5-CRD(L,1))^2)^(.5)
6850 IF DIS=>DSS THEN DIS=DSS:XCL=CRD(L,0):YCL=CRD(L,1):CLL=L
6860 NEXT L:LOCATE 1,3:PRINT USING FXY$+" (Y/ ) ";XCL;YCL
6870 C$=INKEY$:IF C$="" THEN GOTO 6870
6880 IF C$<>"Y" THEN RETURN
6890 X=INT(10^5*XCL):Y=INT(10^5*YCL):GOSUB 5400:RETURN
```

```
6900 REM >>>>>--- P/Z OF H(z) ---<<<<<
6910 LINE (0,0)-(320,7),0,BF
6920 LOCATE 1,3:PRINT "   LOCATION  OF  P / Z  (P/Z/ )    "
6930 C$=INKEY$:IF C$="" OR C$<>"P" AND C$<>"Z" AND C$<>" " THEN GOTO 6930
6940 IF C$=" " THEN RETURN
6950 DIS=10000:IF C$="Z" THEN 7010
6960 FOR L=1 TO GDD:DSS=((X*10^-5-GDR(L,0))^2+(Y*10^-5-GDR(L,1))^2)^(.5)
6970 IF DIS=>DSS THEN DIS=DSS:XCL=GDR(L,0):YCL=GDR(L,1)
6980 NEXT L:FOR L=1 TO HDD:DSS=((X*10^-5-HDR(L,0))^2+(Y*10^-5-HDR(L,1))^2)^(.5)
6990 IF DIS=>DSS THEN DIS=DSS:XCL=HDR(L,0):YCL=HDR(L,1)
7000 NEXT L:GOTO 7040
7010 FOR L=1 TO HND:DSS=((X*10^-5-HNR(L,0))^2+(Y*10^-5-HNR(L,1))^2)^(.5)
7020 IF DIS=>DSS THEN DIS=DSS:XCL=HNR(L,0):YCL=HNR(L,1)
7030 NEXT L
7040 LOCATE 1,3:PRINT USING FXY$+" (Y/ ) ";XCL;YCL
7050 C$=INKEY$:IF C$="" THEN GOTO 7050
7060 IF C$<>"Y" THEN RETURN
7070 X=INT(100000!*XCL):Y=INT(100000!*YCL):GOSUB 5570:GOSUB 5710
7080 GOSUB 5400:GOSUB 5640:GOSUB 5770:RETURN
7100 REM >>>>>--- POLY FORM OF G'DEN ---<<<<<
7110 FOR E=0 TO GDD:GDE(E)=0:NEXT E:E=0:GDE(0)=1
7120 E=E+1:IF E>GDD THEN GOTO 7170
7130 IF GDR(E,1)<>0 THEN GOTO 7150
7140 S2=0:FOR U=0 TO E:S1=GDE(U):GDE(U)=S2-GDR(E,0)*S1:S2=S1:NEXT U:GOTO 7120
7150 B=-2*GDR(E,0):C=GDR(E,0)^2+GDR(E,1)^2:S2=0:S3=0:E=E+1
7160 FOR U=0 TO E:S1=GDE(U):GDE(U)=C*S1+B*S2+S3:S3=S2:S2=S1:NEXT U:GOTO 7120
7170 RETURN
7200 REM >>>>>--- POLY FORM OF H'DEN ---<<<<<
7210 FOR E=0 TO HDD:HDE(E)=0:NEXT E:E=0:HDE(0)=1
7220 E=E+1:IF E>HDD THEN GOTO 7270
7230 IF HDR(E,1)<>0 THEN GOTO 7250
7240 S2=0:FOR U=0 TO E:S1=HDE(U):HDE(U)=S2-HDR(E,0)*S1:S2=S1:NEXT U:GOTO 7220
7250 B=-2*HDR(E,0):C=HDR(E,0)^2+HDR(E,1)^2:S2=0:S3=0:E=E+1
7260 FOR U=0 TO E:S1=HDE(U):HDE(U)=C*S1+B*S2+S3:S3=S2:S2=S1:NEXT U:GOTO 7220
7270 RETURN
7500 REM >>>>>--- ADD A ZERO ---<<<<<
7510 IF SQR((X*10^-5)^2+(Y*10^-5)^2)=>1 THEN RETURN
7520 IF Y =0 AND CNG+1>CDG THEN RETURN
7530 IF Y<>0 AND CNG+2>CDG THEN RETURN
7540 LINE (85,24)-(560,24+8*(CDG+2)),0,BF
7550 CNG=CNG+1:CRN(CNG,0)=X*10^-5:CRN(CNG,1)= Y*10^-5:IF Y=0 THEN GOTO 7570
7560 CNG=CNG+1:CRN(CNG,0)=X*10^-5:CRN(CNG,1)=-Y*10^-5
7570 GOSUB 7800:H=0:GOSUB 3410:RETURN
7700 REM >>>>>--- ADD A POLE ---<<<<<
7710 IF SQR((X*10^-5)^2+(Y*10^-5)^2)=>1 THEN RETURN
7720 LINE (85,24)-(560,24+8*(CDG+2)),0,BF
7730 CDG=CDG+1:CRD(CDG,0)=X*10^-5:CRD(CDG,1)=Y*10^-5:IF Y=0 THEN GOTO 7750
7740 CDG=CDG+1:CRD(CDG,0)=X*10^-5:CRD(CDG,1)=-Y*10^-5
7750 GOSUB 7800:H=0:GOSUB 3410:RETURN
```

```
7800 REM >>>>>--- DRAW NEW ROOT LOCUS ---<<<<<
7810 YYY=CNG:FOR L=1 TO CNG:YYR(L,0)=CRN(L,0):YYR(L,1)=CRN(L,1):NEXT L
7820 UUU=CDG:FOR L=1 TO CDG:UUR(L,0)=CRD(L,0):UUR(L,1)=CRD(L,1):NEXT L
7830 EPS=.00001:GOSUB 3000:GOSUB 3200
7840 CNG=TTT:FOR L=1 TO TTT:CRN(L,0)=TTR(L,0):CRN(L,1)=TTR(L,1):NEXT L
7850 CDG=III:FOR L=1 TO III:CRD(L,0)=IIR(L,0):CRD(L,1)=IIR(L,1):NEXT L
7860 FOR L=0 TO TTT:CUN(L)=TTP(L):NEXT L:FOR L=0 TO III:CED(L)=IIP(L):NEXT L
7870 RETURN
7900 REM >>>>>--- CHECK C(z) ---<<<<<
7910 H=0:GOSUB 7800:GOSUB 4000:GOSUB 6500:GOSUB 5570:GOSUB 5710:GOSUB 5400
7920 GOSUB 5640:GOSUB 5770:C$="K":GOSUB 6100:RETURN
8000 REM >>>>>--- MOVE ALONG THE ROOT LOCUS ---<<<<<
8010 I=0:KK=1:H=0:Q=1:GOSUB 5420
8020 LINE (0,0)-(304,7),0,BF
8030 F$="(##)=###.#####  +##.##### j      ":GOTO 8210
8040 C$=INKEY$:IF C$="" THEN GOTO 8040
8050 IF C$<>" " AND C$<>"R" AND C$<>"C" AND C$<>"B" AND C$<>"O" AND C$<>"N"
     THEN GOTO 8040
8060 IF C$<>"N" THEN GOTO 8110
8070 LOCATE 1,54:PRINT "       x / y / k / f       "
8080 C$=INKEY$:IF C$="" THEN GOTO 8080
8090 IF C$<>"x" AND C$<>"y" AND C$<>"f" AND C$<>"k" THEN GOTO 8080
8100 CX$=C$:GOSUB 4540:GOTO 8040
8110 IF C$<>" " THEN GOTO 8140
8120 I=I+1:I=I MOD CGHD:Q=I+1
8130 LOCATE 1,15:PRINT USING "F"+F$;Q;ROOTS(KK,2*Q-1);ROOTS(KK,2*Q):GOTO 8040
8140 IF C$="O" THEN CGH=((KK-1)*EP+RT):ZR=.97:GOTO 1000
8150 IF C$<>"B" THEN GOTO 8180 ELSE GOSUB 5500
8160 LINE (0,0)-(354,7),0,BF:FOR L=1 TO CGHD
8170 XXCR=ROOTS(KK,2*L-1):YYCR=ROOTS(KK,2*L):GOSUB 8250:NEXT L:GOTO 450
8180 FOR L=1 TO CGHD:XXCR=ROOTS(KK,2*L-1):YYCR=ROOTS(KK,2*L):GOSUB 8240:NEXT L
8190 IF C$="C" AND KK>1 AND (KK-1)*EP+RT=>RT THEN KK=KK-1:GOTO 8210
8200 IF C$="R" AND (KK+1)*EP+RT<=ND THEN KK=KK+1
8210 FOR L=1 TO CGHD:XXCR=ROOTS(KK,2*L-1):YYCR=ROOTS(KK,2*L):GOSUB 8290:NEXT L
8220 LRS=(KK-1)*EP+RT:GOSUB 6160:LOCATE 1,2
8230 PRINT USING "K: "+F2$;(KK-1)*EP+RT      :C$=" ":Q=1:GOTO 8130
8240 REM >>>>>--- SUPPORTING SUBROUTINES FOR 8000 ---<<<<<
8250 XXCR=XXCR*XUN+XRF:YYCR=YYCR*YUN+YRF
8260 IF XXCR<F2X1 OR XXCR>F2X2 OR YYCR>F2Y2 OR YYCR<F2Y1 THEN RETURN
8270 LINE (XXCR-1,YYCR-2)-(XXCR+1,YYCR-2),0
8280 LINE (XXCR-1,YYCR+2)-(XXCR+1,YYCR+2),0:RETURN
8290 XXCR=XXCR*XUN+XRF:YYCR=YRF-YYCR*YUN
8300 IF XXCR<F2X1 OR XXCR>F2X2 THEN RETURN
8310 IF YYCR<F2Y1 OR YYCR>F2Y2 THEN RETURN
8320 LINE (XXCR-1,YYCR-2)-(XXCR+1,YYCR-2)
8330 LINE (XXCR-1,YYCR+2)-(XXCR+1,YYCR+2)
8340 LINE (XXCR  ,YYCR-1)-(XXCR  ,YYCR+1):RETURN
8350 STOP
```

```
10000 REM >>>>>--- PRINT & PRINT OUT ---<<<<<
10010 O$=                                   "   ----------------------------
     &     ---------------------------- "
10020 W$="NUM.        DEN.         NUMERATOR ROOT(S)       DENOMINATOR ROOT(S) "

10030 E$=                  "##:            ##.####^^^^ +
    + ##.####^^^^ ##.####^^^^"
10040 T$=                  "##:##.####^^^^ ##.####^^^^ + ##.####^^^^ ##.####^^
^^  + ##.####^^^^ ##.####^^^^"
10050 Y$="##:##.####^^^^ ##.####^^^^ +        GAIN :  ##.###^^^^         "
10060 IF JJ=1 THEN GOTO 10200
10065 PRINT USING O$;"M(z)"
10070 PRINT "       ";W$:FOR L=MDD TO MND+1 STEP -1
10080 PRINT USING E$;L;MDE(L);MDR(L,0);MDR(L,1):NEXT L
10090 FOR L=MND TO 1 STEP -1
10100 PRINT USING T$;L;MNU(L);MDE(L);MNR(L,0);MNR(L,1);MDR(L,0);MDR(L,1):NEXT L
10110 PRINT USING Y$;0;MNU(0);MDE(0);MGN:PRINT:PRINT USING O$;"C(z)"
10120 PRINT "       ";W$:FOR L=CDG TO CNG+1 STEP -1
10130 PRINT USING E$;L;CED(L);CRD(L,0);CRD(L,1):NEXT L
10140 FOR L=CNG TO 1 STEP -1
10150 PRINT USING T$;L;CUN(L);CED(L);CRN(L,0);CRN(L,1);CRD(L,0);CRD(L,1):NEXT L
10160 PRINT USING Y$;0;CUN(0);CED(0);CGN
10190 GOTO 2010
10200 LPRINT CHR$(15)
10210 LPRINT USING O$;"M(z)"
10220 LPRINT "       ";W$:FOR L=MDD TO MND+1 STEP -1
10230 LPRINT USING E$;L;MDE(L);MDR(L,0);MDR(L,1):NEXT L
10240 FOR L=MND TO 1 STEP -1
10250 LPRINT USING T$;L;MNU(L);MDE(L);MNR(L,0);MNR(L,1);MDR(L,0);MDR(L,1):NEXT
10260 LPRINT USING Y$;0;MNU(0);MDE(0);MGN:LPRINT:LPRINT USING O$;"C(z)"
10270 LPRINT "       ";W$:FOR L=CDG TO CNG+1 STEP -1
10280 LPRINT USING E$;L;CED(L);CRD(L,0);CRD(L,1):NEXT L
10290 FOR L=CNG TO 1 STEP -1
10300 LPRINT USING T$;L;CUN(L);CED(L);CRN(L,0);CRN(L,1);CRD(L,0);CRD(L,1):NEXT
10310 LPRINT USING Y$;0;CUN(0);CED(0);CGN
10320 LPRINT "   Y(t)":LPRINT "    !":MX=CINT(OS*15)+1:DF=0
10330 C$="":FOR L1=0 TO 73
10340 IF CINT(15*C(L1))=MX-DF AND L1 MOD NUN=0 THEN C$=C$+"O":GOTO 10370
10350 IF CINT(15*C(L1))=MX-DF THEN C$=C$+"*":GOTO 10370
10360 C$=C$+" "
10370 NEXT L1:DF=DF+1
10380 IF DF=MX+1 THEN GOTO 10470
10390 FOR L1=0 TO 10:RED=.2*(MX-DF+1)/3
10400 IF MX-DF+1=3*L1 THEN GOTO 10420
10410 NEXT L1:GOTO 10460
10420 IF RED>1 THEN LPRINT STR$(RED)+"!"+C$
10430 IF RED=1 THEN LPRINT STR$(RED)+"  !"+C$
10440 IF RED<1 THEN LPRINT STR$(RED)+" !"+C$
10450 GOTO 10330
```

```
10460 LPRINT "    !"+C$:GOTO 10330
10470 C$="!":IF C(0)=0 THEN C$=C$+"0"
10480 FOR L1=1 TO 73
10490 IF CINT(15*C(L1))=0 AND L1 MOD NNN=0 THEN C$=C$+"0":GOTO 10520
10500 IF CINT(15*C(L1))=0 THEN C$=C$+"*":GOTO 10520
10510 C$=C$+"-"
10520 NEXT L1
10530 LPRINT " 0.0"+C$
10540 LPRINT "    0   5   1   5   2   5   3   5   4   5   5   5
6   5   7  3"
10550 LPRINT "         OVERSHOOT            = ";CINT((OS-1)*100)
10560 LPRINT "         RAMP-ERROR CONS. (Kv) = ";KV
10570 LPRINT "         RISE TIME            = ";RIT
10580 GOTO 2220
11000 REM >>>>>--- ROOT SOLVING PROGRAM ---<<<<<
11010 N=DEGR:K=0
11020 DIM QV(N),QA(N),QB(N),QC(N)
11030  IF POLY(K)<>0 THEN GOTO 11050
11040 K=K+1:GOTO 11030
11050  N=N-K
11060 FOR L=0 TO DEGR:RR(L,0)=0:RR(L,1)=0:NEXT L
11070 FOR L=0 TO N:QV(L)=POLY(L+K):NEXT L
11080 NN=0: IF N<4 THEN GOTO  11350
11090 HQ=((N-1)*QV(N-1))^2-2*N*(N-1)*QV(N)*QV(N-2):IF HQ<0 THEN GOTO 11130
11100 DET=QV(N-1)+SQR(HQ):IF DET=0 THEN DET=.00001
11110 PQ1=N*QV(N)/DET:DET=QV(N-1)-SQR(HQ):IF DET=0 THEN DET=.00001
11120 PQ2=N*QV(N)/DET:R=-(PQ1+PQ2):S=-PQ1*PQ2:GOTO 11160
11130  DET=(QV(N-1))^2+ABS(HQ)
11140 IF DET=0 THEN DET=.00001
11150 MG=N*QV(N)/DET:R=-2*MG*QV(N-1):S=-N*QV(N)*MG
11160 NN=NN+1: SPARE1=0:SPARE2=0:SPARE3=0:SPARE4=0:FOR L1=0 TO N
11170 QB(N-L1)=CDBL(QV(N-L1))+CDBL(R)*CDBL(SPARE1)+CDBL(S)*CDBL(SPARE2)
11180 SPARE2=CDBL(SPARE1):SPARE1=CDBL(QB(N-L1))
11190 QC(N-L1)=CDBL(QB(N-L1))+CDBL(R)*CDBL(SPARE3)+CDBL(S)*CDBL(SPARE4)
11200 SPARE4=CDBL(SPARE3):SPARE3=CDBL(QC(N-L1)):NEXT L1
11210 DET=CDBL(QC(1))*CDBL(QC(3))-CDBL(QC(2))*CDBL(QC(2))
11220 IF DET=0 THEN DET=.000001
11230 DR=(CDBL(QB(1))*CDBL(QC(2))-CDBL(QB(0))*CDBL(QC(3)))/CDBL(DET)
11240 DS=(CDBL(QB(0))*CDBL(QC(2))-CDBL(QB(1))*CDBL(QC(1)))/CDBL(DET)
11250 IF NN>70 THEN GOTO 11340
11260 IF INKEY$<>" " THEN GOTO 11320
11270 LOCATE 1,54:PRINT "   EPS  OR  QUIT  (E/Q)   "
11280 C$=INKEY$:IF C$="" OR C$<>"E" AND C$<>"Q" THEN GOTO 11280
11290 IF C$<>"E" THEN GOTO 11710
11300 LOCATE 1,54:PRINT "                          "
11310 LOCATE 1,60:INPUT "  EPS :";EPS:GOTO 11080
11320 R=CDBL(R)+CDBL(DR):S=CDBL(S)+CDBL(DS)
11330 IF ABS(DR)>EPS OR ABS(DS)>EPS THEN GOTO 11160
```

```
11340 AA=1:FOR L1=0 TO N-2:QV(L1)=QB(L1+2):NEXT L1:GOTO 11600
11350  REM   3'rd   DEGREE  ROOTS
11360 IF N<3 THEN GOTO 11570
11370 AA=CDBL(QV(2))/CDBL(QV(3)):B=CDBL(QV(1))/CDBL(QV(3))
11380 CC=CDBL(QV(0))/CDBL(QV(3)):P=CDBL(B)-CDBL((CDBL(AA))^2)/CDBL(3)
11390 Q=CDBL(CC)-CDBL(AA)*CDBL(B)/3+(2*CDBL((CDBL(AA))^3))/27
11400 US=CDBL((CDBL(P)/3)^3)+CDBL((CDBL(Q)/2)^2):IF US<0 THEN GOTO 11480
11410 U=CDBL(SQR(CDBL(US))):VV=CDBL(U)-CDBL(Q)/2
11420 V=SGN(VV)*CDBL(CDBL((ABS(VV)))^(1#/3)):WWW=-CDBL(U)-CDBL(Q)/2
11430 W=SGN(WWW)*CDBL((CDBL(ABS(WWW)))^(1#/3))
11440 RR(N,0)=CDBL(V)+CDBL(W)-CDBL(AA)/3:REAL=-(CDBL(V)+CDBL(W))/2-CDBL(AA)/3
11450 IMAG=CDBL(SQR(3)/2#)*(CDBL(V)-CDBL(W)):RR(N-1,0)=REAL:RR(N-2,0)=REAL
11460 RR(N-1,0)=CDBL(REAL):RR(N-2,0)=CDBL(REAL)
11470 RR(N-1,1)=CDBL(IMAG):RR(N-2,1)=-CDBL(IMAG):GOTO 11550
11480  U=CDBL((-(CDBL(P/3#)))^3)^(1/2))
11490 TE=-CDBL(Q/2#/U):PI=4#*ATN(1)
11500 TE=PI/2#-CDBL(ATN(CDBL(TE)/SQR(1#-CDBL(TE)*CDBL(TE)+1E-10)))
11510 KC=2#*CDBL((CDBL(U))^(1#/3))
11520 RR(N,0)=CDBL(KC)*CDBL(COS(TE/3#))-AA/3#
11530 RR(N-1,0)=CDBL(KC)*CDBL(COS(TE/3#-2*PI/3#))-AA/3#
11540 RR(N-2,0)=CDBL(KC)*CDBL(COS(TE/3#+2*PI/3#))-AA/3#
11550  N=N-3
11560 GOTO 11710
11570  REM   2'nd  DEGREE   ROOTS
11580 IF N<2 THEN GOTO 11690
11590 AA=QV(2):R=-QV(1):S=-QV(0)
11600 B=-R:C=-S:DELTA=CDBL(B^2)-4#*CDBL(AA)*CDBL(C)
11610 IF DELTA<0 THEN GOTO 11650
11620 RR(N,0)=-(CDBL(B)+CDBL(SQR(DELTA)))/2/CDBL(AA)
11630 RR(N-1,0)=-(CDBL(B)-CDBL(SQR(DELTA)))/2/CDBL(AA)
11640 GOTO 11670
11650  REAL=-CDBL(B)/2#/CDBL(AA):IMAG=CDBL(SQR(ABS(DELTA)))/2#/CDBL(AA)
11660 RR(N,0)=REAL:RR(N-1,0)=REAL:RR(N,1)=IMAG:RR(N-1,1)=-IMAG
11670  N=N-2
11680 IF N>1 THEN GOTO 11080
11690  IF N<1 THEN GOTO 11710
11700 RR(N,0)=-QV(0)/QV(1)
11710 ERASE QV,QA,QB,QC:RETURN
```