tanulmányok

52/1976

MTA Számítástechnikai és Automatizálási Kutató Intézet Budapest

COMPUTER AND AUTOMATION INSTITUTE,
HUNGARIAN ACADEMY OF SCIENCES

COMPUTER AIDED DESIGN

(A Proposal for a Synergistic Graphic Drafting System
Based on an Evaluation of the Hungarian DIALOGUS System)

Warren A. POTAS
Visiting Research Associate
Brown University,
Providence, R.I., USA

Reports 52/1976

Responsible Publisher

T.Vámos

ISBN 963 311 026 2

Készült az Országos Müszaki Könyvtár és Dokumentációs
Központ házi sokszorositójában
F.v.: Janoch Gyula

C O N T E N T S

ABSTRACT

A concise framework for evaluating the user interface aspects of sophisticated interactive graphics systems is offered, furnishing the basis for the evaluation undertaken of the DIALOGUS system for programming numerically controlled machines. Proposals are presented for modifying DIALOGUS so as to create a system expected to be more appealing and efficient for the user.

INTRODUCTION

The task of programming numerically controlled machines has recently been accomplished by specifying all machining actions in a specialized language and then compiling these programs to produce control tapes (typically in the actual form of paper tape) for the machines. Such tapes contain in an international standard format all information needed to govern the operation of a numerically controlled machine and, sometimes, information for workshop level editing of the program. Most of the languages used for machining descriptions belong to a class of similar languages evolved from the APT language developed about a decade ago.

The programming process described above is essentially a means of encoding geometrical and spatial concepts in terms of a written language. It would appear to be more natural to deal with such concepts in a direct graphical manner, and thus the employment of computer graphics terminals as input/output media for specification of machine tool programs seems very appropriate and is beginning.

DIALOGUS SYSTEM

One of the efforts so far is the Hungarian DIALOGUS system, created by Gyula Pikler, et al., and still under development. This system is designed to create a graphically based interactive environment which resembles APT in orientation. The drawing component of the system (presently the only completed porton) has recently been used and examined. It was concluded that this system is not fully taking advantage of the power afforded by its sophisticated graphics terminal and computer system hardware to provide a productive and congenial environment for the user. Therefore, this paper, which reviews the DIALOGUS system, also proposes a system considered to be significantly superior for the same purposes.

The DIALOGUS drawing subsystem (DIALOGUS/Drawing) is driven by a tree-following mechanism in which the entire system monologue is pre-defined in branching node-choice form. Drawing is accomplished through choice among the options presented at each node of this tree and response to information requests. While such a mechanism for interaction is not inherently bad, in DIALOGUS/Drawing it is found that the variety of interaction possibilities among geometrical elements has been unnecessarily restricted, and questions are required to be answered regardless of whether the information is really needed. Furthermore, due to the manner in which permissable user actions are stringently governed by the monologue, genuine real-time interaction is denied while the system appears overly pedantic. At all times the user is confronted by a system interrogation or imperative. Thus the system more resembles a serial batch processor, polling the user for all possibly relevant information, digesting it, and making any allowed modifications to the data structure or display, than a sophisticated design tool under the user's control.

Figures 1, 2 and 3 respectively, depict the monologue trees for constructions of points, (infinite length) lines and circles in DIALOGUS/Drawing. The branches represent choices to

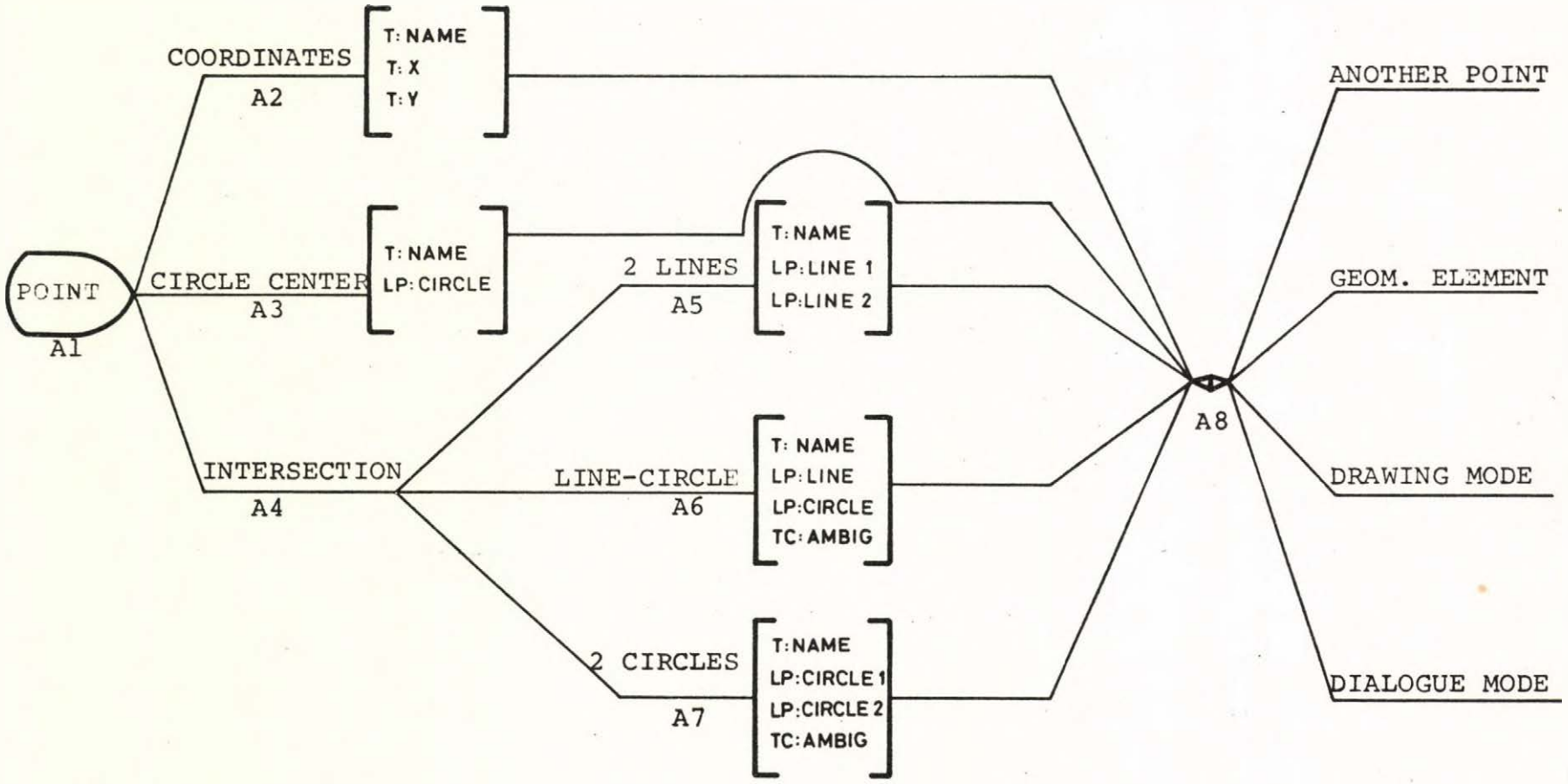

FIGURE 1

DIALOGUS/Drawing: POINT Definition

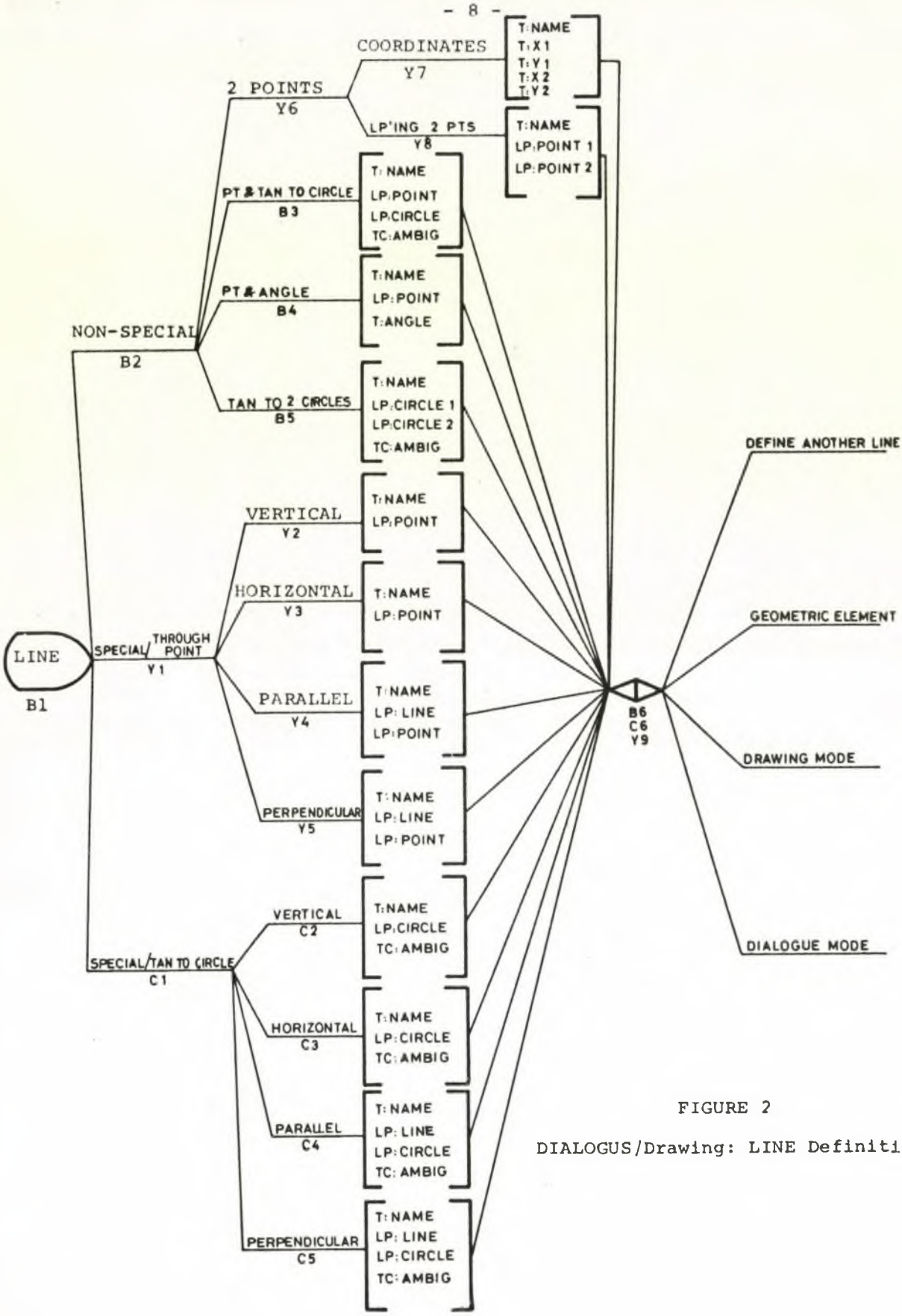

FIGURE 2

DIALOGUS/Drawing: LINE Definition

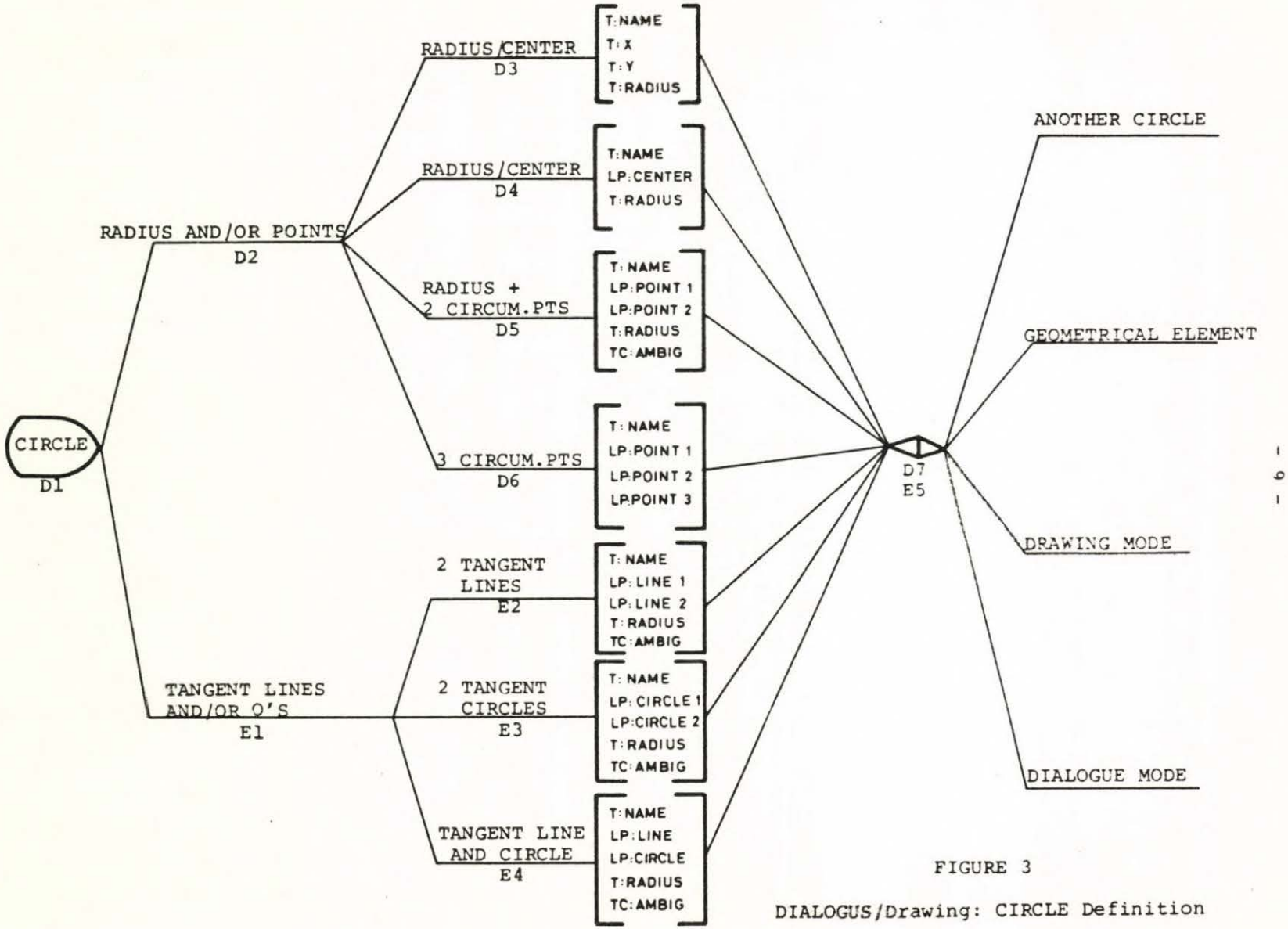

FIGURE 3

DIALOGUS/Drawing: CIRCLE Definition

be indicated by cursor; "T:" signifies information which is requested to be typed in; "LP:" indicates use of the lightpen to select already in place geometrical elements; and "TC:" represents the requirement that ambiguities as to placement of a new element be resolved by use of the tracking cross symbol manipulated by tracking ball.

BASIS OF EVALUATION

Some basic principles governing the design of the proposal submitted herein have been as follows:

i/ Presentation of choices and options to the user within the same logic framework with which he is prepared to deal with and think about them;

ii/ Maximal use of the computer to process such choices and options into forms consistent with the principle above;

iii/ Offering, for each instance information is requested, use of the most expressive and appropriately manipulated tool (e.g. lightpen, cursor, function keys, tracking ball, keyboard) for conveying the user's intent;

iv/ Minimization of the frequency with which the user must switch among manipulative tools.

The first principle really has two components - the order in which the interaction unfolds and the number of disparate concepts which the user is called on to deal with at any time. The order in which information is solicited from the user should correspond to the manner in which the user would himself like to think of his work as progressing. Furthermore, a user should be called on to deal at any moment only with

those concepts which are immediately relevant to him. As has been demonstrated by psychological research, there is a limited number of concepts which a person can deal with at any one time using his short-term memory (7±2). Furthermore, the human mind appears to operate on the basis of "closure", the continual attempt to deal with the environment of mental processes so as to be able to purge or resolve the concepts fecund in short-term memory. It is significant to note, however, that there seems to be no limit to the complexity of a concept which can be regarded, given an appropriate succinctness of expression. However, a period of learning may be required before a person becomes familiar enough with a new notation, terminology or other means of expression, to be able to conceive of complex concepts as single entities.

Principles three and four, concerned with the flow of information into the computer system, require a balance to be struck between optimality of the tool for its purpose and the manipulative problem of frequently changing tools. Another factor which enters is the possible boredom of a graphics terminal user if, for example, only a single tool is made available and, especially, if all communication from the computer is identical in format. One possible solution is to allow the user to make his own choice among several different tools which are egually applicable in a situation, such as the lightpen and the cursor for making menu selections.

EVALUATION AND PROPOSED ALTERNATIVES*

Defining Geometrical Elements

DIALOGUS/Drawing, as well as this proposal, calls for placement of the basic geometrical elements of point, line and circle exactly, by utilizing geometrical relationships, on an

*The Appendix present supplementary criticisms of the DIALOGUS/Drawing system.

implicit coordinate grid. An important point of understanding is that the supporting graphic system cannot be a system for doing "free-hand" creative drawing. Rather, the system must be such wherein the display provides only a visual approximation of objects whose location and relationships have been expressed exactly and so stored in the underlying data structure.

Whereas in DIALOGUS/Drawing any positioning of a new geometrical element may be accomplished only in terms of entities which have already been explicitly defined (or coordinates), this proposal allows the more natural course of defining elements "on the fly". For example: in DIALOGUS/Drawing if the user wishes to define a circle whose center is the intersection of two lines, he must, before he thinks of placing the circle, first explicitly define a point whose location is that intersection. Within this proposal, however, is the capability for the user, during definition of the circle, to identify the intersection of the two lines as that point which is to be the center. In other words, points can always be defined when the need arises through geometrical subroutining. There appears to be no need for lines or circles to be invoked as subroutines. This proposal should thus dramatically reduce the total number of elements which define a drawing by eliminating most of the explicitly defined points. The facility for placing elements is thus more direct and in keeping with the way the user regards the problem.

Figure 4 presents the complete proposal. It may first be noted that the entire drawing system can be presented easily on one page whereas DIALOGUS/Drawing requires three. In this figure, "LP:" and "T:" have the same lightpen and type-in meaning as for Figures 1., 2. and 3. However, a new symbol appears, to indicate point-subroutining: "( )". Thus whenever (POINT) appears, the choice of an explicit point or all possible definitions of a point are possible. (*POINT) also includes the tangency definition of a point. It can be seen that the scheme presented in Figure 4 corresponds more intuitively to customary concepts of geometrical construction

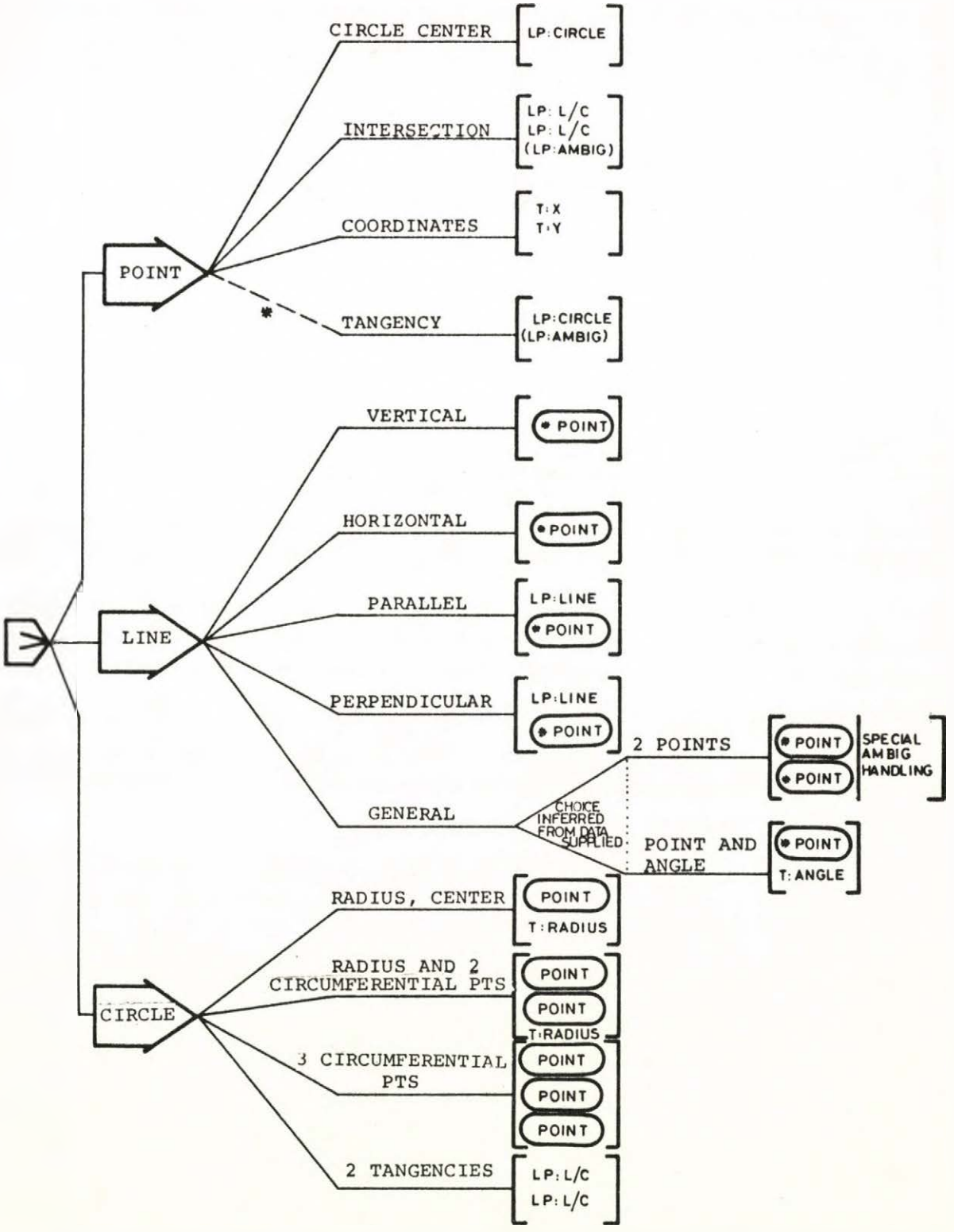

FIGURE 4

NEW PROPOSAL: POINT, LINE and CIRCLE Definitions

than that of Figure 1 , 2 and 3 together (DIALOGUS/Drawing).

In this proposal the means of selecting among decision branches would be changed. In DIALOGUS/Drawing the cursor controls are solely reserved for, and are the sole means of making, such selections. Because the lightpen is frequently used, as are the other manipulative tools of the terminal to a lesser extent, the resulting constant switching among tools can be disruptive of the design process. Thus in this proposal the lightpen would be considered the principle means of making branch decisions. As the cursor controls have no other use in the system, it would be possible to enable them for making branch decisions, thereby allowing the user to employ whichever tool was more convenient at any moment for this major component of activity.

Use of the tracking ball - tracking cross would be eliminated by this proposal. DIALOGUS/Drawing always requires use of the tracking cross for resolution of possible positional ambiguities, as for example identification of which of the two points defined by the intersection of two circles is desired. However, the requirement that the tracking cross be used is true even if the two circles are tangent or even non-intersecting! Even when a distinction is required in a particular situation, use of the tracking cross is both cumbersome and imprecise since its positioning sometimes requires the imaginary construction of lines or regions to be sure of the correct placement of the symbol. Whenever a case of ambiguity actually arises, this proposal calls for all possible ambiguous elements to be displayed, and then for the intended element to be lightpenned by the user.

Geometrical Element Names

Another departure of this proposal from that of DIALOGUS/Drawing is the complete elimination of the requirement that every single geometrical element be assigned a unique name by the user before it is created. As it stands, this requirement is extraordinarily disruptive of the design

process. Names for elements are relevant only in the context of workshop level editing of a machining program. To the extent that such editing is contemplated, the drafting system could automatically generate names for all elements. It could also be possible for the user to assign a personally chosen name to any particular element be desired.

Defining Contours

The most radical departure of this proposal from DIALOGUS/Drawing is the manner in which contours are specified. Contours represent paths which machine tools will follow, areas to be stamped, etc., and are defined in terms of segments of the geometrical elements which have been placed on the implicit grid. Figure 5 depicts the monologue tree available in DIALOGUS/Drawing for defining contours. This approach results in an often excrutiatingly tedious, cumbersome effort involving multitudinous actions employing the disparate tools of cursor, lightpen and tracking cross in the effort to define a contour.

The contour definition method offered by this proposal, on the other hand, allows the user to explicitly and directly "point out" the contour, just as he visualizes it, and with minimal activity on his part. Due to its non-verbal nature, the interaction proposed for contour definition is best shown graphically, by example, as in Figure 6. The user first lightpens the geometrical element with which he wishes to start the contour. If the element embodies more than one segment (a segment is a portion of a line or circle lying between two intersections), all segments are marked as choices, and the user selects the desired one. All segments radiating from the intersections at either end of the first segment are then marked and the user lightpens his next choice. Now the direction of the contour becomes established, and segments are marked which emanate from the new free intersection. This process continues until the contour closes

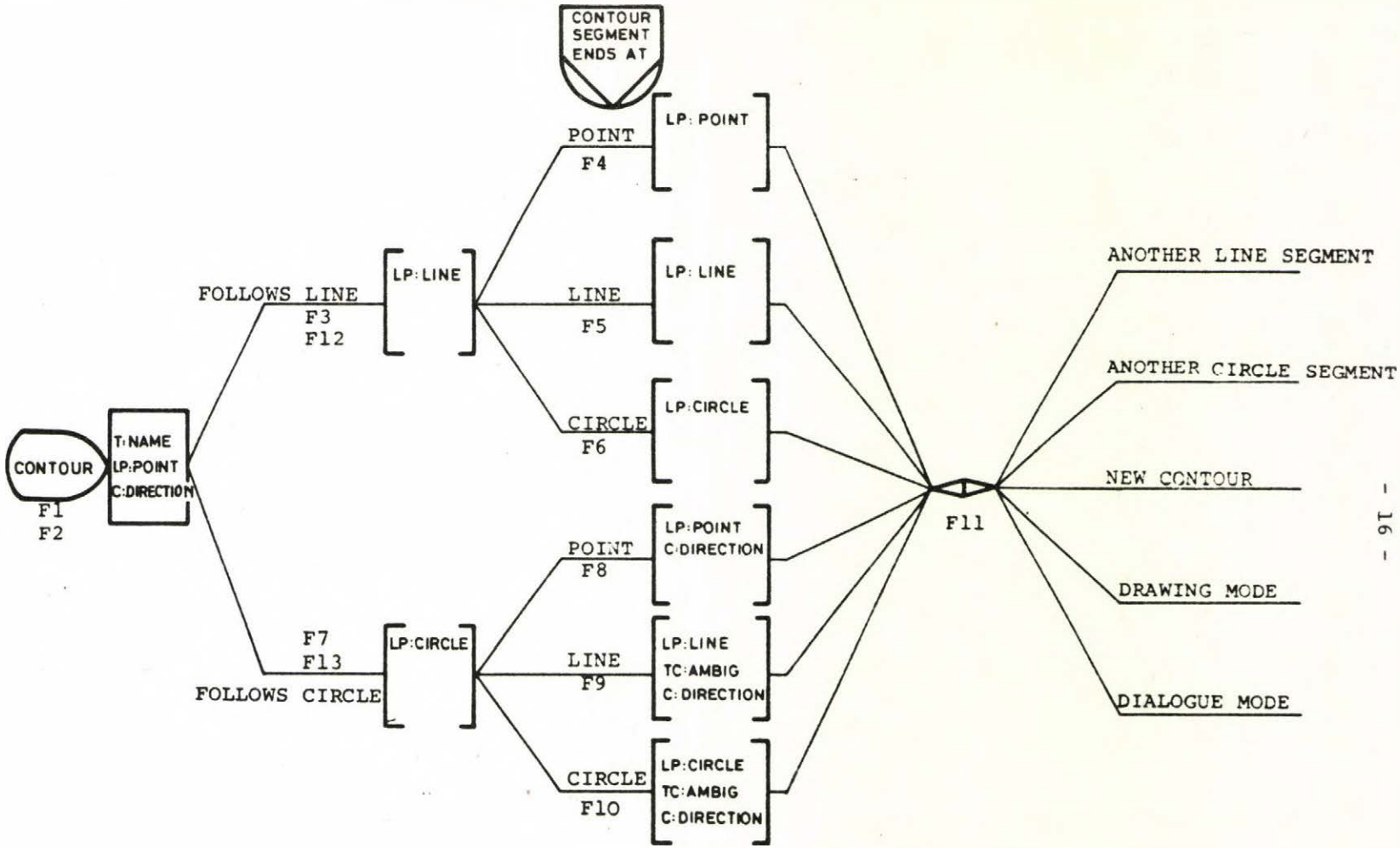

FIGURE 5

DIALOGUS/Drawing: CONTOUR Definition

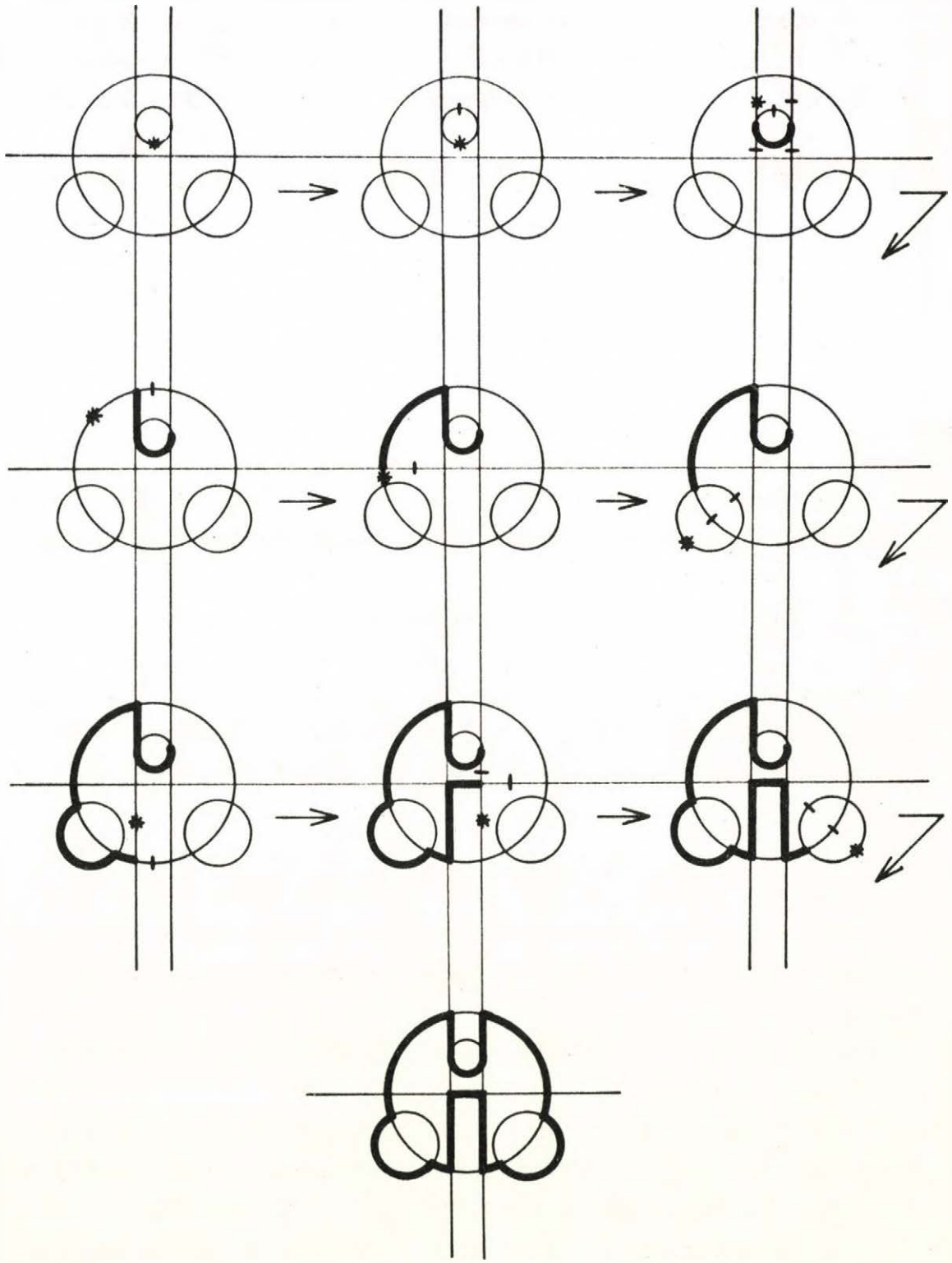

FIGURE 6

NEW PROPOSAL: Example of CONTOUR Definition

on itself or the user signals its completion.

In most cases, it is possible to simplify the contour drawing process even further by observing all or some of the following conventions (which were employed in Figure 6):

i/ No segments of infinite length are ever offered as choices (this would actually be a contradiction of the definition of segment);

ii/ No segments are offered as choices if they intersect the contour so far identified at other than its free end;

iii/ Segments are automatically included in the contour by the system, without user intervention, if, after application of the rules above, there is only the choice of a single segment.

Thus in the example offered in Figure 6, the contour was defined with merely 9 lightpen actions. By contrast, DIALOGUS/Drawing would require 65 actions (33 cursor selections, 25 lightpen actions and 7 resolutions of "ambiguities" by tracking cross) to define this same contour.

Figure 7 presents an example which appears ideally suited to take advantage of the fact that in the DIALOGUS/Drawing system contours following a line or curve are regarded as single entities even if they actually consist of multiple segments. Thus it might be expected that DIALOGUS/Drawing would allow the indicated contour of Figure 7 to be drawn with far fewer actions than under this proposal. This is not the case, however. This proposal requires significantly fewer actions than DIALOGUS/Drawing: 56 lightpen actions versus 63 actions (consisting of 32 cursor choices, 25 lightpen actions, and 6 resolutions of "ambiguities" by tracking cross). The contour consists of 64 segments.

The amount of computation required by the proposed contour definition approach would increase dramatically over that of

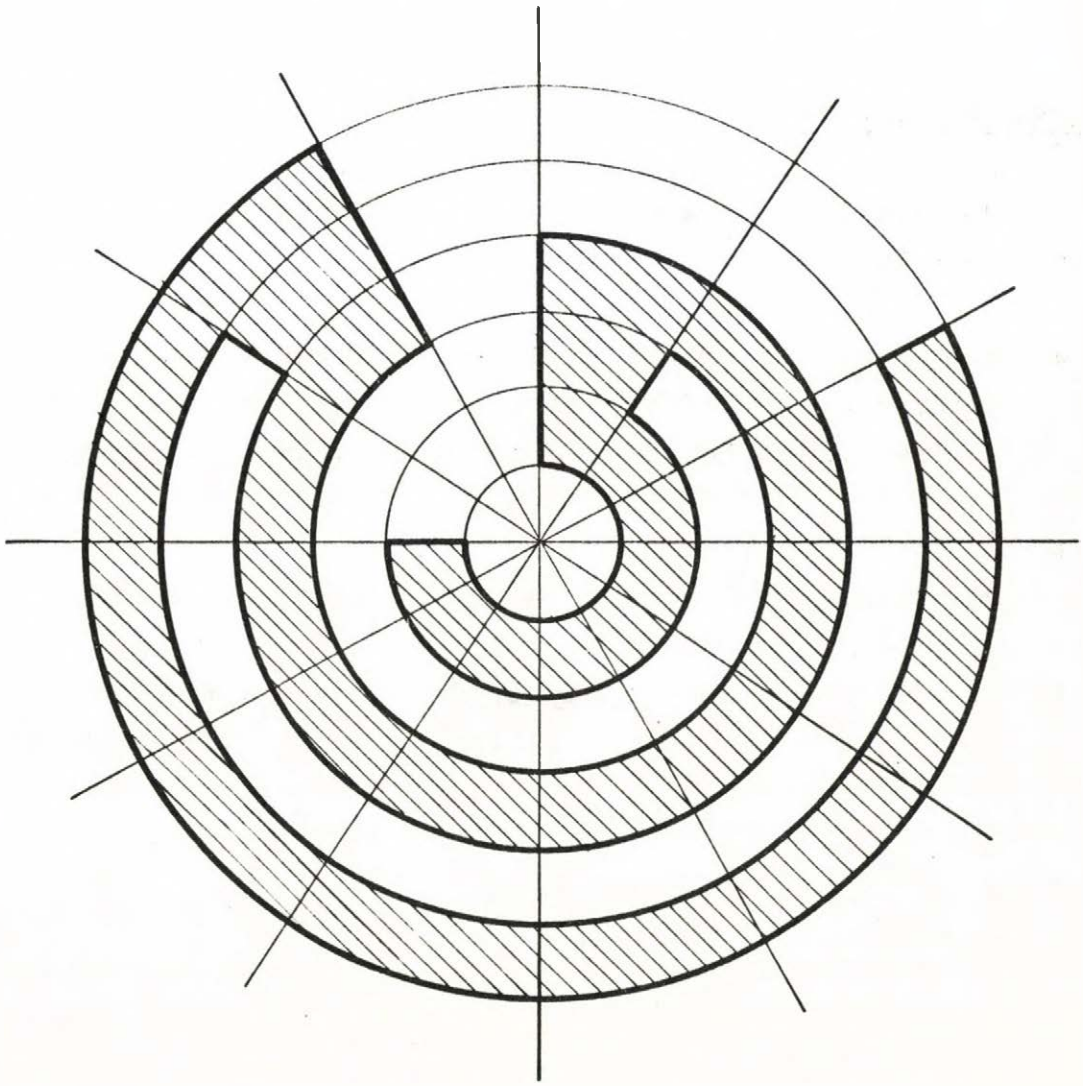

FIGURE 7

Contour Definition Example

DIALOGUS/Drawing: 63 actions (32 cursor, 25 lightpen and 6 tracking cross)
New Proposal: 56 lightpen actions

contour comprised of 64 segments

simultaneously being more succinct, less pedantic, and more in line with his accustomed means of thinking about design. It is felt that the proposed system is a truly synergistic tool for use in the mechanical design process.

ACKNOWLEDGEMENTS

I would like to express my appreciation to the following individuals and organizations:

- To all my colleagues at SZTAKI for their generous assistance and advice of both a technical and personal nature during the course of my program in Budapest;

- To József Hatvany and Andries van Dam for their efforts in making this research exchange program possible;

- To the Hungarian Academy of Sciences and the U.S. National Science Foundation for their funding of the program.

APPENDIX

Listed below are those features of DIALOGUS/Drawing which the author found to detract from the system's desirability. Most of these items are very narrow in scope; the more global aspects of the system have been treated in the body of this paper. It should be noted that these comments were based on examination of a system still under development and that many of the cited "problems" could be altered quite easily. References of the form "$XX" are to specific branch points in the monologue trees of the system, and most may be located on the diagrams of the DIALOGUS system presented here Figures 1, 2 , 3 and 5.

DISPLAY SCREEN FORMAT

1/ The presence of the stationary cursor symbol in windows which have no need of it or are vacant (the error and sub-system descriptor windows) is distracting.

2/ The large physical separation of the error messages at the top of the screen from the control information at the bottom of the screen is not desirable, as the appearance of an error message may not at first be noticed due to the user's concentration on the lower portion.

3/ The lack of scale information on the drawing window poses a severe difficulty when defining points in terms of coordinates.

SYSTEM MONOLOGUE TREE ORGANIZATION AND CONTENTS

1/ Since the designer of the system has stated that $Z2 (grid dimensioning) is intended to be used only once for a given drawing at the very beginning, it is annoying to have to encounter this panel every time drawing mode is re-entered for a given picture.

2/ Message texts and choices are often unclear or ambiguous, especially to the novice user, and can be annoying to the familiar user. Words and phrases are often neither precisely nor consistently used. An example: phrases such as "I give the two points by pointing at (a két pontot megadom rámutatással)" or "pointing at was wrong (a rámutatás hibás volt)" should be changed to refer to the instrument to be used for 'pointing', the lightpen. Another case: unless the user has learned what really happens if any of the following three choices are made ($B6), they are virtually inscrutable at face value:

- "new line, please (új egyenest kérek)
- geometrical element, please (geometriai elemet kérek)
- else geometry, please (egyéb geometriát kérek)
- I stop geometry (geometriát abbahagyom)"

A system should be designed so as to be clear for both new and accustomed users.

3/ The ordering of choices in cursor menus in sometimes inconsistent. For example: in $B1 a reference to a choice involving a circle is made before the choice involving a point, whereas in most other instances the ordering is always point, then line, and then circle. Another example: In $U4 the non-local choice, "re-specify dialogue mode", precedes the local choice, "re-specify the viewing window".

4/ Decisions are sometimes required to be made before the implications of such decisions are known. For example: in $B1 the choice must be made between "special" and "non-special" lines even though at that point the user does not know what the difference is. Moreover, the tangent special case could more simply be regarded as a novel way of defining a point.

5/ Choice menus are sometimes inconsistent and insufficiently exhaustive of reasonable possibilities for defining elements. For instance in defining a circle based on three circumferential points have been previously defined as such, and no other option, such as specification of a point in terms of coordinates, is offered. However in $D3/$D4 such choice is provided. Similarly in $Y6, a line may be defined in terms of two points which have both either been predefined or which will both be specified as coordinates, but no provision is made for each point being defined in a separate manner.

6/ In different choice menus, the identical choice may be referred to differently, leading to confusion as to whether the choices are really the same. An example:"Geometriából kilépek" in $F11 and "Geometriát abbahagyom" in $T19 are identical in result.

7/ At that point at which $U3 is asked (during window translation and zoom), the picture should already have been translated to reflect the newly specified window center. Not doing so only serves to make the job of the user harder in specifying the required zoom factor, whereas the system already has the information neccessary to perform the translation. A system should always make maximal use of data as soon as it becomes available, in order to make things easier for the user.

8/ During alteration of the window, $U2/K1 and $U3 are redundant and should be combined into a single question such as "by what factor should the scale change?".

9/ Some choice menus are completely unneccessary. $U4 is an example. After newly specifying the window, it should be unlikely that the user would want to immediately re-specify the window.

CURSOR MANIPULATION

1/ All menu choices should be immediately adjacent. A given choice should not take more than one line; otherwise the relationship between menu choice and the number of times the move-cursor button need be pushed becomes unpredictable, and every menu becomes a special case.

2/ To the user, the position of the cursor on the choice menu when it appears comes to be regarded as a default choice. As presently structured, the cursor location on the bottommost choice is not related to any rational selection of defaults among the choices in a given panel.

3/ A very curious means of "beating the system" was discovered. In every case in which a choice menu is presented, if the user keeps pressing the move-cursor-down key after the cursor is at the bottom of the panel, the text in that block will begin to move upward with concommitant loss of the topmost line each time. Subsequent use of the move-cursor-up key does not bring back the text, but choice may be made as if the text was present.

4/ In order to help minimize cumbersome switching among terminal tools, menu choices should be selected through use of the lightpen, which is already an essential tool and as well or better suited to the purpose, rather than the cursor which has no other purpose.

NAMING OF ELEMENTS/CONTOURS

1/ It is perceived that the system allows a contour to be given the same name as a geometrical element. This may or may not be desirable but represents a significant policy decision if names really matter.

DATA ENTRY VIA TYPE-IN

1/ Whenever the system rejects typed in data, the data entry field is not cleared prior to the user re-entering data, although the cursor is re-positioned. Moreover, any leftover characters to the right of the cursor which are not specifically blanked out become part of the new data

read by the system. This violates a principle often adhered to in interactive systems that when reading from the "screen" everything to the right of the cursor is ignored.

LIGHTPEN ACTIVITY

1/ In $A6/K1 a lightpen hit could not be effected on the line within one half inch of the intersection being defined as a point. In $A6/K2, however, the actual point of intersection could be lightpenned in designating the circle.

AMBIGUITY RESOLUTION

1/ The user is always asked to resolve a case in which ambiguity may under certain conditions be present, even though in a particular case there may be no ambiguity. An example is $A7 in which a point is defined as the intersection of two circles which were constructed to be tangent.

2/ It is not terribly fast or convenient to move the tracking symbol large distances on the screen with the tracking ball.

CONTOUR DRAWING

1/ The necessity of defining the direction of the entirety of a contour (before it has been drawn) as clockwise or counterclockwise is completely absurd and is of absolutely no interest or concern to the user. The fact that the particular algorithm employed to compute areas requires this information simply means that either the system must be able to impute this information itself, or else the algorithm should be changed. It is fundamentally unsound for any system to request information from the user which the user himself has no direct need of and no concern for.

2/ Use of the symbols "+" and "-" to indicate clockwise and counterclockwise(or is it vice versa?) is pure folly. Even the system designers are never immediately sure which is which.

3/ The requirement that the user tell the system what kind of element a contour will next follow is a major annoyance. The system could easily determine this from the user's lightpenning of the desired element. The same is true for the kind of element at which a contour segment will terminate.

DIALOGUS/Drawing, but the computational problem would not be prohibitively severe. This increase in intensity of computer utilization results on the other hand in a 660% reduction, from 65 to 9, in the number of user actions required to define the first example contour, while achieving a 12% improvement in an example tailored for DIALOGUS/Drawing. The tradeoff chosen in this proposal would surely be appreciated by users, even if response time lenghtened slightly. However, the response time problem is far more amenable to solution, via customized subroutining or special firmware for example.

Window Manipulation

Yet another difference between the two systems lies in their approach to window manipulation (translation and zoom). DIALOGUS/Drawing requires that the user complete whatever drawing actions he is persuing and enter a different mode ("Design Modification") before he can alter the window.

This proposal calls for the window translate and zoom controls to be available asynchronously with all other operations, so that at any point in the drawing process the scale or viewing center can be altered.

CONCLUSIONS

It is felt that the proposals set forth herein will result in a dynamic, truly interactive mechanical design drawing system directly responsive to the user's needs and at the same time structured in such a way as to maximize productivity of and enjoyment by the user. This is achieved by allowing the user to deal with concepts in the same parsimonious fashion with which he has dealt with them in his previous experience with the task of constructing such designs. Compared to DIALOGUS/Drawing, this proposal offers a far simpler overall environment for the user to have to learn to deal with, while

ABSENT FEATURES

1/ The ability to write and recall designs to/from disk or some other kind of offline storage is lacking.

GENERAL CONSIDERATIONS

1/ Errors should be detected and reported as soon as feasible. For example, if a non-intersecting line and circle are specified in $A6, then $A6/K3 should not be reached wherein the user is requested to use the tracking cross to resolve the ambiguity. There is clearly no ambiguity; indeed, an error condition exists.

2/ The user is required to switch among display station tools with burdensome regularity (especially among cursor controls, lightpen, tracking cross, keyboard and "END" key). In particular, the switch to the tracking cross to resolve ambiguities, as in $A6, is not really needed were the system to resolve the problem differently. A user should not unnecessarily have to change the tool he is using.

HARDWARE CONSIDERATIONS

1/ Even the smaller character size on the GD-71 display terminal is too large to allow many characters to be placed on a line, especially near the extremities of the round screen, where DIALOGUS has its text areas. The 30 characters per line offered by these text areas is vastly insufficient for presentation of meaningful messages and choices to the user.

2/ As is often true for applications making use of display terminals, the "SOM" ("END") key is marvelously labeled with a meaningless name. A naive user would have no idea of any possible relevance for this key and would be likely to forget it once told. Furthermore, the prominence of its position on the keyboard does not correspond to its significance.

DIALOGUS/Drawing, but the computational problem would not be prohibitively severe. This increase in intensity of computer utilization results on the other hand in a 660% reduction, from 65 to 9, in the number of user actions required to define the first example contour, while achieving a 12% improvement in an example tailored for DIALOGUS/Drawing. The tradeoff chosen in this proposal would surely be appreciated by users, even if response time lenghtened slightly. However, the response time problem is far more amenable to solution, via customized subroutining or special firmware for example.

Window Manipulation

Yet another difference between the two systems lies in their approach to window manipulation (translation and zoom). DIALOGUS/Drawing requires that the user complete whatever drawing actions he is persuing and enter a different mode ("Design Modification") before he can alter the window.

This proposal calls for the window translate and zoom controls to be available asynchronously with all other operations, so that at any point in the drawing process the scale or viewing center can be altered.

CONCLUSIONS

It is felt that the proposals set forth herein will result in a dynamic, truly interactive mechanical design drawing system directly responsive to the user's needs and at the same time structured in such a way as to maximize productivity of and enjoyment by the user. This is achieved by allowing the user to deal with concepts in the same parsimonious fashion with which he has dealt with them in his previous experience with the task of constructing such designs. Compared to DIALOGUS/Drawing, this proposal offers a far simpler overall environment for the user to have to learn to deal with, while

ABSENT FEATURES

1/ The ability to write and recall designs to/from disk or some other kind of offline storage is lacking.

GENERAL CONSIDERATIONS

1/ Errors should be detected and reported as soon as feasible. For example, if a non-intersecting line and circle are specified in $A6, then $A6/K3 should not be reached wherein the user is requested to use the tracking cross to resolve the ambiguity. There is clearly no ambiguity; indeed, an error condition exists.

2/ The user is required to switch among display station tools with burdensome regularity (especially among cursor controls, lightpen, tracking cross, keyboard and "END" key). In particular, the switch to the tracking cross to resolve ambiguities, as in $A6, is not really needed were the system to resolve the problem differently. A user should not unnecessarily have to change the tool he is using.

HARDWARE CONSIDERATIONS

1/ Even the smaller character size on the GD-71 display terminal is too large to allow many characters to be placed on a line, especially near the extremities of the round screen, where DIALOGUS has its text areas. The 30 characters per line offered by these text areas is vastly insufficient for presentation of meaningful messages and choices to the user.

2/ As is often true for applications making use of display terminals, the "SOM" ("END") key is marvelously labeled with a meaningless name. A naive user would have no idea of any possible relevance for this key and would be likely to forget it once told. Furthermore, the prominence of its position on the keyboard does not correspond to its significance.