SCIENTIFIC REPORT no 3

I.E.I. Research Report

AN ALGORITHMIC DESCRIPTION OF CUT-VECTOR IN COMPUTATION THE APT SYSTEM

by G.GALBIATI and S. TRUMPY

Nota interna B70-18

Pisa, July 21 1970

This research has been sponsored in part by the Air Force of Scientific Research, BRUSSELS, OAR United States Air force under contract F 61052 67 C 0097

This document is part of the research on

LANGUAGES FOR AUTOMATIC PROGRAMMING TOOLS

Alfonso CARACCIOLO DI FORINO

Contract F 61502 67 C0097

I N D E X

Introduction

This report deals with the computer-aided determination of a three-dimensional path for numerically controlled machine-tools, and describes the APT-III approach to the solution of this problems, where "APT-III" stands notoriously for the III version of "Automatically Programmed Tools System".

The report is a first step towards the formal definition of programming languages for N.C. machine tools, and it is motivated by the awareness that formalization in the field of artificial languages is needed not only for theoretical linguistic problems, but also for practical problems, both in Applied research and in Development; above all for urgently felt exigencies of standardization.

As a first step, the problem of commanding a continuous path three-dimensional motion has been circumscribed to the fundamental case of only one "check-surface" as a stopping surface for the tool, with one single intersection. After briefly discussing the interpretative problems arising from lack of determinacy, there follows a description in outline of the APT system and of its arithmetic element for computing the tool path; finally there comes a critical description of the routines in APT Section 2 that solve the problem: each is commented first and then formally defined, with added flow-charts when a visual grasp is also deemed advantageous.

The language used in routine definition follows a pattern of thought originating with ALGOL 68; therefore it is hoped it should retain the ALGOL 68 perspicuity and self-consistency in revealing the inner structure of the processes described, both within each and among themselves.

This language precisely is also presented in the beginning of the report, which - to conclude - is also motivated by the hope of presenting, as simply and as ordinately as possible, a very complex problem; a problem that an unacquainted reader might find hard from the original sources only.

Chapter A -

The ARELEM - Problem

The subject matter of this report is the complex problem known as "ARELEM Problem", which can be defined as follows: Being given two controlling surfaces, the "drive" and the "part" surface, and some tolerance limits to each; the initial tool position within tolerance of both; the tool configuration; a suitable command for continuous-path motion to a single check-surface with a single intersection point(*); to realize a three-dimensional motion of the tool along the given pair of surfaces, up to a stopping position on the given check-surface.

It is at once obvious that the problem is very general and widely indeterminate, whence more than a few interpretative problems arise.

To begin with, "to realize a motion" implies a list of successive tool positions that generate a path such that, within the tolerances allowed, it approximates the ideal path "along" the pair of the drive and the part surface - i.e. defined by their intersection. Clearly, there is no finite number of such paths, whence the problem arises of how to realize computational procedures for determining "cut-vectors"(**) not just acceptable, but of optimal length. For instance, fig.s 1 and 2 show two continuous-path motions along the D-surface, where the first one achieves optimality conditions for cut-vector lengths, while the second one is not relevant in this respect.

(*) we restrict discussion of the ARELEM Problem to this fundamental case.

(**) the vector from any tool-end location to the next one is the cut-vector, where the intersection of the tool axis with the tool bottom is the tool-end.

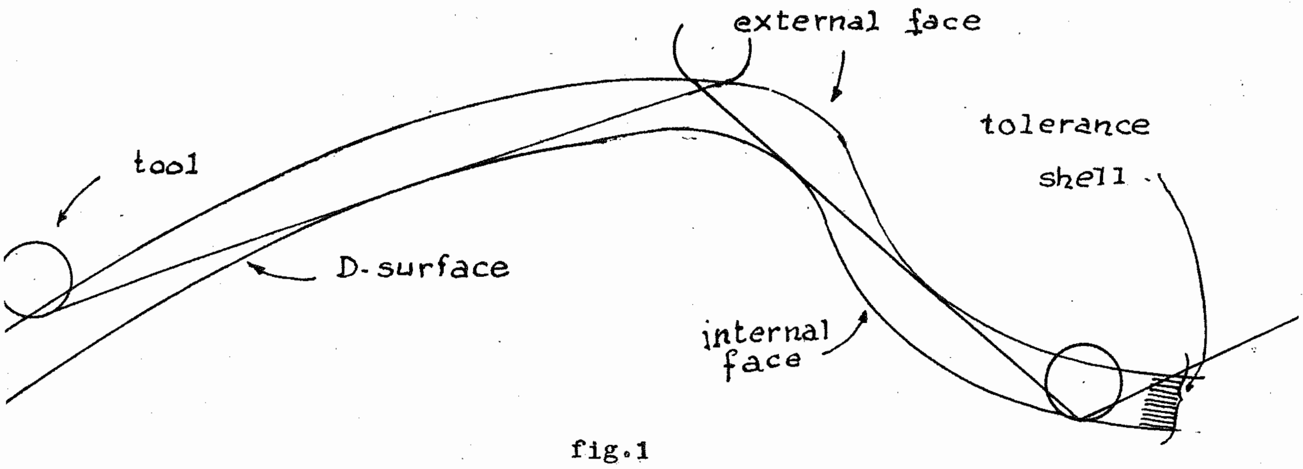


fig.1

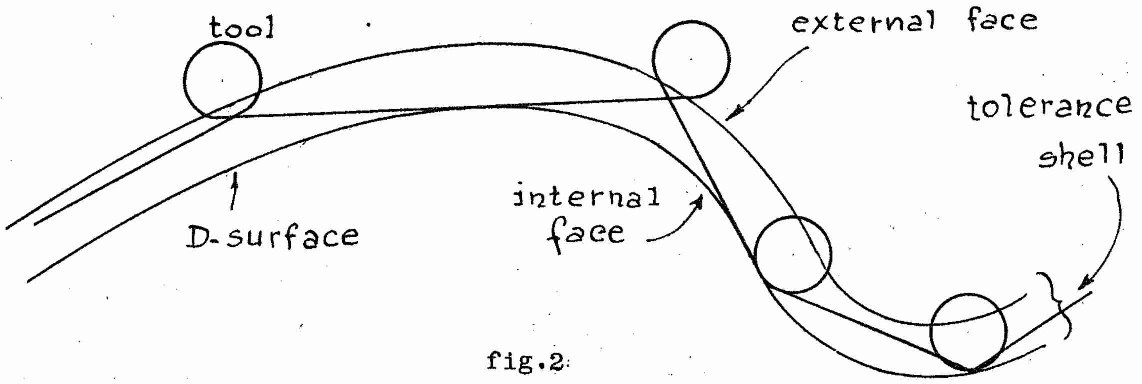
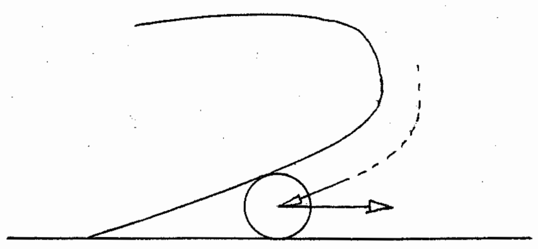
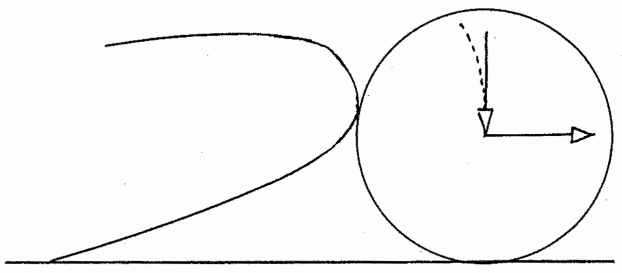


fig.2

Furthermore, there is the added, serious problem of how to control positioning of the tool and of the pair of controlling surfaces with respect to one another. To speak of motion "along" two surfaces can infact mean, either that the tool-end belongs to, or that the cutter (cutting edge or ring of the tool) is kept tangent to, their intersection. Therefore "up to a stopping position" comes to mean, respectively, a position where,either the tool-end belongs to, or the cutter is tangent to, the check-surface. Procedures for cutter positioning with respect to the drive-, the part-, and the check-surface are indeed very difficult to define, as these types of control depend heavily upon shape of the tool and tolerance limits for the surfaces. Infact the APT system defines a surface analytically, but the notion of "belonging to", and of "being tangent to", such a surface should always be understood as, respectively, belonging to its tolerance shell, or being tangent to a surface parallel to it and contained in its tolerance shell.

Finally, one added difficulty is in defining a forward direction that depends upon the tool shape. Infact, such a direction for a continuous-path motion is chosen on the basis of the previous one in the part-program and of a suitable reference point in the tool configuration, which in the simplest cases is the tool-end. Therefore, variety of tool shapes does affect relevantly this choice. The figures below, for instance, show two different tool shapes -whence two distinct motion commands - for the same type of motion alog two surfaces.


GOBACK/ . . .


GOLFT/ . . . .

This critical problem is solved in the APT system through a certain sequence of routines, at whose higher level stands the ARELEM routine described in Chapter D of this thesis at paragraph 32. It will appear to be the keystone of the whole solution, by operating as follows: first a trial cut-vector is found by mathematical computations used heuristically; then it is checked to find out whether it violates some surface tolerance or exceeds length limits for stopping on the check-surface; finally it is confirmed as a cut-vector if checks give a negative result, otherwise it is modified and tried again until accepted. The whole procedure iterates for the next cut-vector. This heuristic character in the basic stage of the ARLM3 method is thus due to the indeterminacy - whence the lack of analyticity - in the very general problem that is to be solved.

Furthermore, the ARLM3 method requires the definition of several more routines, as listed in fig.4, some mostly mathematical in character (DOST, QUAD, UNRMAL, CCURV, CENTR), others, instead, mostly heuristic (DDTABC , RADAR, TLNORM).

Besides ARLM3 itself, Chapter D presents all the computational procedures auxiliary to it, at various levels from elementary to higher ones, according to the scheme in fig.4, and in the same order as they have been realized, i.e., piecemeal, starting from the shortest and simplest, ending up with the longest and most complex ones. Therefore this order of presentation obviously differs from the sequential order one would have, if each procedure were built up as soon as called for by logical requirements in defining ARLM3.

However, it seems advantageous to examine the general organization of the APT system before describing the said routines, in order to clarify how and when each one plays its rôle.

Chapter B

The APT - III System

1. *System Sections and Section functions*

The APT system is a computer program for converting English-like statements that describe geometrically parts to be machined and corresponding cutting machine motions into input-data to the Numerically Controlled machine tool that executes the cutting operations assigned.

The APT-III version of APT is functionally organized into five major Sections numbered from 0 to 4, as follows: 0. Section 0 acts as a traffic control for the execution of the functions in Sections 1 to 4, calling them in and out, and providing them with sub-routines of its own. Thus permanently operating in the computer, it mainly controls the part-program flow, keeps track of its execution times, and modifies the standard flow in case of errors during the process. Obviously, for its very function as execution control, this Section has no output.

1. Section 1 essentially converts the part-program into numerical data for computer processing. First it resolves nested definitions such as "macros" and "looping"; then it searches for possible errors of input-format; finally it reduces to canonical form various geometrical definitions, having previously resolved - if that is the case - nested definitions therein contained, and performed correspondingly the required computations.

The output of this process of numerical conversion is PROTAP, namely the part-program in numerical form.

2. Section 2 contains the arithmetic elements for computing the tool positions associated with cutting operations on the part to be machined, with their specific tolerances.

It receives PROTAP as input, and writes its output on CLTAPE. The PROTAP data that do not require further computational processing, such as the "post-processor" commands, are simply copied on CLTAPE.

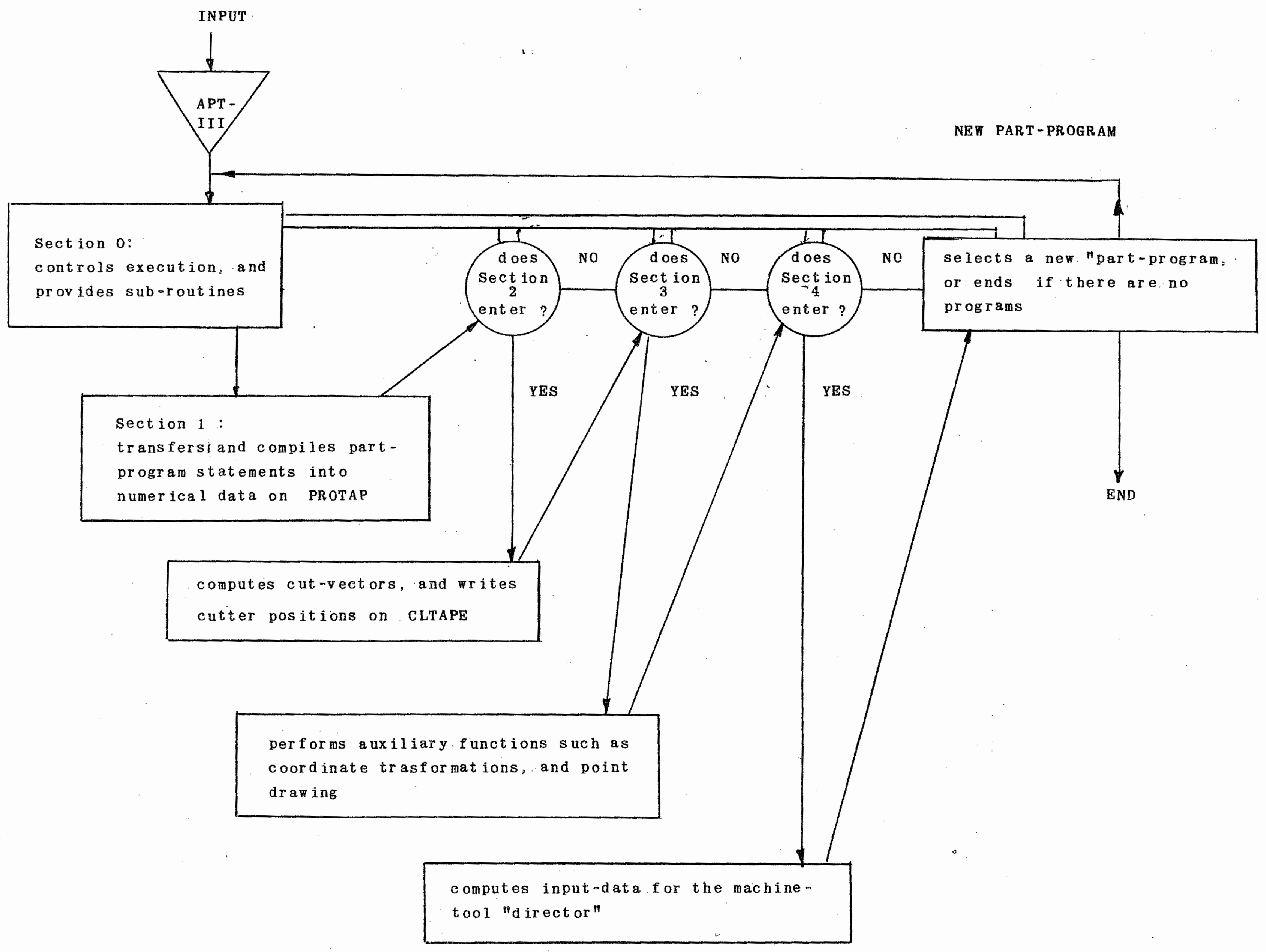
3. Section 3 executes auxiliary functions such as, for instance, coordinate transformations and point drawing.

4.Section 4 is the final stage of the APT procedure: it converts the cutter positions and the machine-tool control functions into the specific format required for each machine-tool "director".

It receives CLTAPE as input, and gives a sequence of control tapes as output.

Fig.3 shows how the five Sections are connected with the APT flow diagram.

As a last remark, it can be immediately noticed that it is precisely the Section 2 routines, listed in fig.4, that are paramount for the solution of the problem undertaken; therefore the next paragraph will deal with Section2 in some more detail.


INPUT
APT-III
NEW PART-PROGRAM
Section 0: controls execution, and provides sub-routines
does Section 2 enter ?
NO
does Section 3 enter ?
NO
does Section 4 enter ?
NO
selects a new "part-program, or ends if there are no programs
YES
YES
YES
END
Section 1 : transfers; and compiles part-program statements into numerical data on PROTAP
computes cut-vectors, and writes cutter positions on CLTAPE
performs auxiliary functions such as coordinate trasformations, and point drawing
computes input-data for the machine-tool "director"

2. *The Section 2*

As every other Section, Section 2 is endowed with a basic routine for controlling the whole Section, in our case SECTN2, which essentially performs the following functions:

1) reads record-by-record the numerical data in PROTAP resulting from conversion of the part-program statements; then, either it directs the operations flow to various computing programs of Section 2, or it forwards directly to Section 3 (CLTADE) those data that cannot be processed by Section 2;
2) searches for errors, and reports "diagnostics".

This is therefore the routine that recognizes records encoding continuous-path motion statements; in such a case it transfers command to the AREPRE routine, which at its turn performs the computations preliminary to the execution of motion statements. Next, it transfers command to the STRTUP routine if motion is of the initial type, i.e., a start-up motion; otherwise we enter the ARELEM problem proper.

This problem can be solved in two dimensions, therefore particularly simply, when the following conditions concur:

a) the part-surface is a plane;
b) one check-surface only is specified, either as a plane, or as a cylinder parallel to the tool-axis;
c) the drive-surface is either a plane, or a cylinder parallel to the tool-axis;
d) the angle for the cutter is zero (see fig.16);
e) no three-dimensional computations are specified in the part-program.

In this case, then, the ARELEM, problem is solved by the ARLM2 routine; otherwise there will take over the ARLM3 routine, which is precisely the main topic of this report, namely in Chapter D as already announced. Fig.4 below may help to follow visually the routine hierarchy in this fundamental chapter.

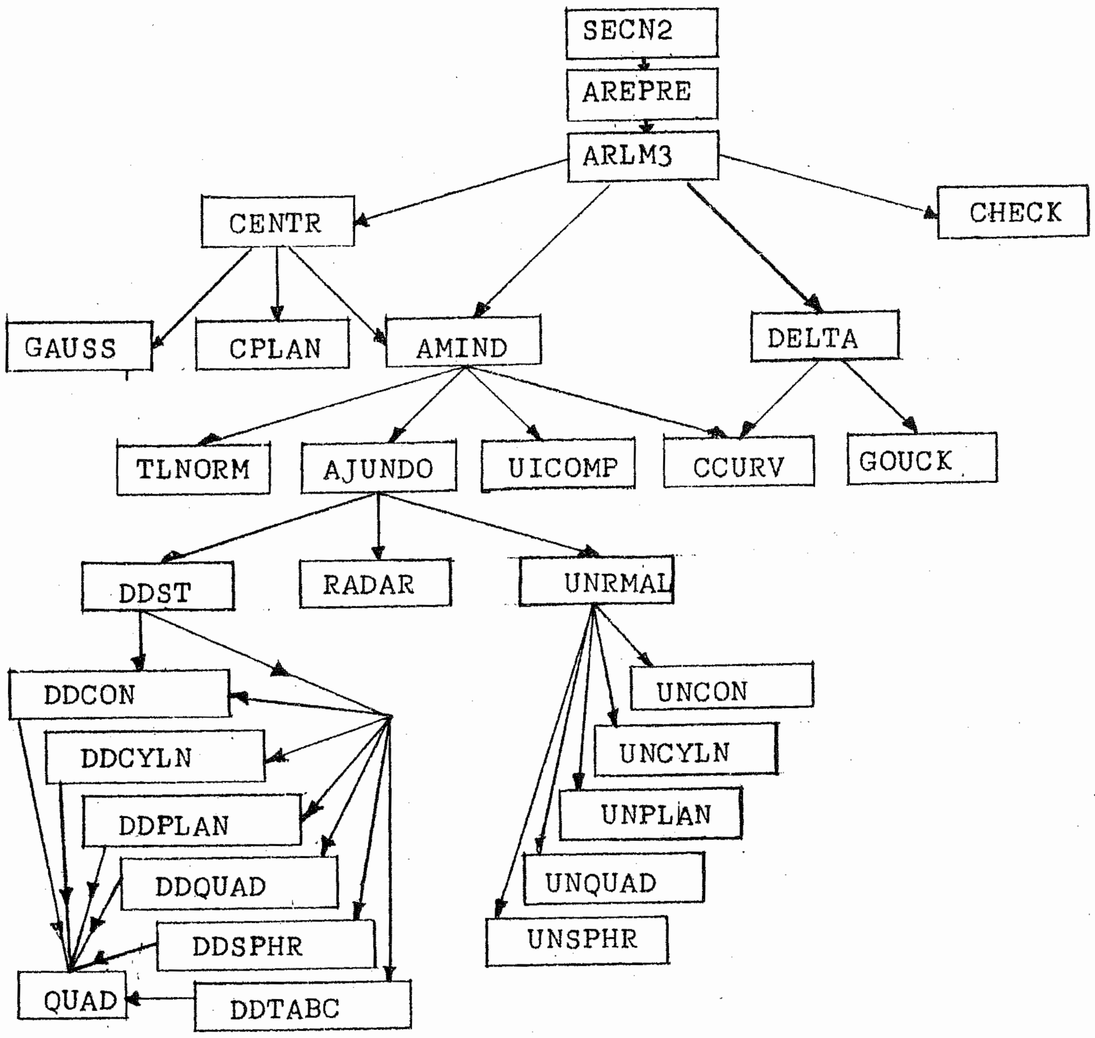


Fig.4

Finally, in order to introduce the type of language that has been used for the routine definition, one more short chapter, namely Chapter C, will precede Chapter D.

Chapter C

On the definition of Section 2 routines

1. *The Method*

Every routine in Section 2 that we consider basic, in the sense that it is somehow related to the fundamental one ARLM3, shall be described in chapter D according to the following scheme in four points:

1) the "Aim", to present what is required at the output of a routine with a given type of input;
2) preliminary "Remarks", optional, to clarify the ensuing computational process;
3) the "Process" proper, described in terms that will be discussed at paragraph 2;
4) concluding "Remarks", also optional, as critical comments about the process.

In some important cases there will also be shown a flow-chart, drawn according to the rules laid at paragraph 3.

2. *The language*

The procedure declaration statement, identified by the code word "proc", is structured as follows:

type proc NAME (parameter $\overline{V1}$, prmtr $\overline{V2}$, ..., prmtr $\overline{VN}$) where

1) "type" defines the type of the routine output parameter or parameters; the complete list of type-symbol words is to be found at page 118;
2) "NAME" stands for the routine call name, made of three or more alpha-numerical characters; the complete list for routines described or only quoted here is to be found at page 117;
3) "parameter $\overline{V1}$" defines the routine first input-element $\overline{V1}$ as belonging to a given set of elements (having some property in common, such as the

set of points, of vectors, of planes), where "parameter" is the class-name for all the sets here considered; analogously for $\overline{V2}$,.., $\overline{VN}$; the complete list of parameters is to be found at page 113

N o t e : the procedure declaration statement is often preceded by a premise that is a short reminder of the procedure-type definition.

Two examples are given below for illustrating the structure definition:

1: <u>scal</u> <u>proc</u>. DDPLAN (<u>point</u> $\overline{P}$, <u>vect</u> $\overline{AS}$, <u>plane</u> $\overline{PL}$)

2: <u>dupla</u> = (<u>point</u> $\overline{TP}$, <u>norm</u> $\overline{TN}$)

<u>dupla</u> <u>proc</u>. TLNORM (<u>vect</u> $\overline{TA}$, <u>point</u> $\overline{TE}$, <u>tool-data</u> $\overline{TD}$,
<u>surf</u> $\overline{SF}$ <u>modifier</u> $\overline{MOD}$, <u>point</u> $\overline{CC}$,
<u>vect</u> $\overline{SN}$, <u>control</u> $\overline{FCON}$)

From the logical viewpoint, and following a school of thought originating with ALGOL.68, a procedure can be looked at as a mapping P

$$P : A_1 \times A_2 \times \ldots \times A_n \longrightarrow B_1 \times B_2 \times \ldots \times B_m$$

of a set or a set-product on a set or a set-product. The mapping is "non-total" when there is some n-ple $(a_1,\ldots,a_n) \in A_1 \times \ldots A_n$ for which there is no m-ple $(b_1,\ldots,b_m) \in B_1 \times \ldots B_m$ correspondingly defined. In such cases, much more frequent than the "total" cases, the procedure output is not normal; therefore, in the following, when writing a procedure that is non-total, attention will be called on this aspect by writing two stars and a specific comment in between as follows: * .... * .

The procedure declaration statement is followed by the procedure-body, that is a sequence of APT words enclosed by the words "<u>begin</u>" and "<u>end</u>" , underscored. The body is structured as for ALGOL 60 procedures, therefore the declarative part starts soon after the first $\underline{\text{begin}}$.

There are two types of procedure body statements:

1) for those parameters that were not previously defined by the procedure declaration part (either in the premise, or in the parameter section) and are to be computed by the routine in question;
2) for those various types of routines that happen to be called in within the procedure body itself. One example will clarify this point:

```
Es. $\underline{\text{scal}}$ $\underline{\text{proc}}$  ABC  ($\underline{\text{parameter}}$ $\overline{\text{VI}}$,......, $\underline{\text{parameter}}$ $\overline{\text{VN}}$ )
    $\underline{\text{begin}}$
        $\underline{\text{dupla}}$  $\overline{\text{DUP}}$
        $\overline{\text{DUP}}$ = ABCD ($\underline{\text{parameter}}$ $\overline{\text{WI}}$,......, $\underline{\text{parameter}}$ $\overline{\text{WN}}$ )
    $\underline{\text{end}}$
```

that is, if the ABC routine calls the ABCD routine of the $\underline{\text{dupla}}$ type, then one is first to name the ABCD routine output, naming it $\overline{\text{DUP}}$, for instance, and then define $\overline{\text{DUP}}$ as "$\underline{\text{dupla}}$ $\overline{\text{DUP}}$".

Finally, the procedure-body definition makes use of the following word types:

1) words of three or more alpha-numerical characters in capital type, to define routines and names;
2) words of one or two alpha-numerical characters in capital type, to define statement addresses;
3) super-scored words to define, either input, or computation, or output parameters, plus the output of routines that happen to be called in (such as $\overline{\text{DUP}}$ in the previous example) within the procedure body;
4) under-scored words in small type, to define
   1) type of parameter (as $\underline{\text{point}}$, $\underline{\text{surf}}$),
   2) type of routine (as $\underline{\text{duple}}$),
   3) ALGOL words, or any of the new words listed and defined at page 116
5) current English words in small type, with their current meaning.

3. *The Flow-Chart*

In some cases, as a visual aid to clarify the routine process, a flow-chart will be drawn, according to the following scheme: having partitioned the routine process into parallel-serially connected coherent sub-processes after the control plan, the procedure body comes to be correspondingly partitioned into (adjoining, non-overlapping) sub-sequences of statements, whose first one in each is made to carry at its left an address word beginning with a letter (or, usually, consisting of just a letter followed by a number, optional, and a colon). To each sub-sequence thus determined there it is then made to correspond one-to-one a box in the flow-diagram, by marking it with that address-word as a symbol-word, by summarizing in it the corresponding sub-process, and by linking it to the other boxes parallel-serially after the control plan by means of arrows. Then each box thus characterized, with its symbol-word, its contents, and its arrows, is a "block" in the flow-diagram. Fig.5 below shows an example of flow-chart, for the routine outlined on the left

```
real proc NAME (......)
begin
         ------
         ------
A1:      ------
         ------
B1:      if ... then ... else...
         ------
 1:      ------
         ------
1A:      ------
         ------
 C:      ------
end
```

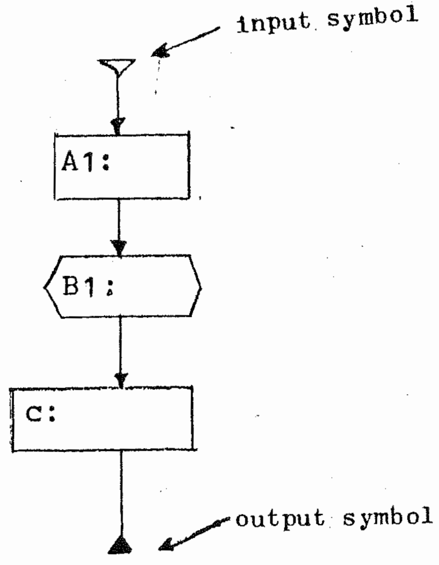


fig.5

Chapter D

Basic routines in Section 2

1. *Foreword notice*

Before endeavoring to define the Section 2 routines according to the criteria that have just been presented, it seems appropriate to announce here that there will be defined also some more routines that are not to be considered as belonging to Section 2, but should be rather grouped aside as - let us say - "service" routines, insofar as they are used in describing computational processes.

Only for routines of the Section 2 proper a distinct paragraph will be reserved, one to each, unless a different arrangement is explicitly stated.

2. *The DDST routine*

Aim: to compute the direct(*) distance (real) from a given point $(\overline{P})$ along a given direction $(\overline{\Delta S})$ to a given surface $(\overline{SF})$ subordinately to the $\overline{FIOP}$ flag stated below.

Remarks: computations depend only on the type of surface, therefore the routine main operation is that of trans-ferring control to the particular sub-routine provided for each type of surface involved. Now, the APT system also deals with 2nd order surfaces, for which the above said distance may not result uniquely determined; therefore, in such a case it is made to intervene the input flag $\overline{FIOP}$, whose value "0" or "1" or "-1", as determined in a previous compiling stage of the part-program, is used as a selection operator value respectively for

(*) i.e., the least distance in absolute value, or the least distance positively oriented

- the least distance in absolute value;
- the least distance positively oriented (previously found existing);
- the least distance positively oriented, if existing, or else the least in absolute value.

Process: The routine is defined by the following procedure

```
real proc  DDST (point $\overline{P}$, vect $\overline{\Delta S}$, surf $\overline{SF}$, flag $\overline{FIOP}$)
begin
A:    if IS-PLAN ($\overline{SF}$)(3) then
B:    begin
          DDST = DDPLAN ($\overline{P}$, $\overline{\Delta S}$, $\overline{SF}$);
          goto 1 ;
      end
A1:   if  IS-SPHR ($\overline{SF}$)(3) then
B1:   begin
          DDST = DDSPHR ($\overline{P}$, $\overline{\Delta S}$, $\overline{SF}$, $\overline{FIOP}$);
          goto 1;
      end
A2:   if IS-CON($\overline{SF}$)(3) then
B2:   begin
```

(3) Boolean procedure to decide whether the surface $\overline{SF}$ is of the type defined in the procedure NAME

```
            DDST = DDCON ($\bar{P}$, $\overline{\Delta S}$, $\overline{SF}$, $\overline{FIOP}$);
            goto 1;
       end
A3:    if  IS-CYLN ($\overline{SF}$) then
B3:    begin
            DDST = DDCYLN ($\bar{P}$, $\overline{\Delta S}$, $\overline{SF}$, $\overline{FIOP}$);
            goto 1;
       end
A4:    if  IS-QUAD ($\overline{SF}$) then
B4:    begin
            DDST = DDQUAD ($\bar{P}$, $\overline{\Delta S}$, $\overline{SF}$, $\overline{FIOP}$);
            goto 1;
       end
A5:    if  IS-TABC ($\overline{SF}$) then
B5:    begin
            DDST=DDTABC ($\bar{P}$, $\overline{\Delta S}$, $\overline{SF}$, $\overline{FIOP}$);
            goto 1;
       end
C:  *a diagnostic will be printed;*
1: end
```

Remarks: here and elsewhere no attention is paid to the case of ruled surfaces, which should be handled quite differently from that of the surfaces such as cones, cylinders, quadrics, and tabulated cylinders considered here.

The following flow-chart renders visually the routine process

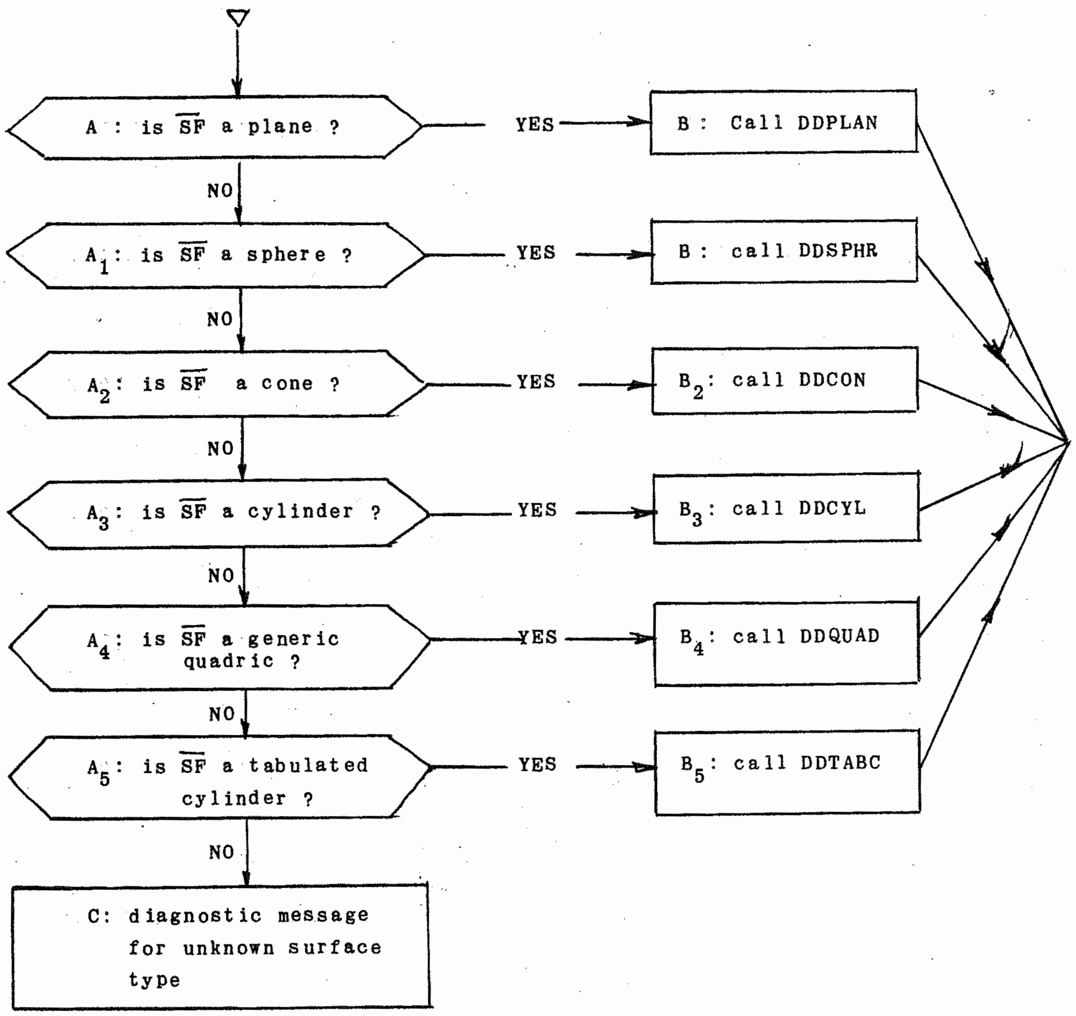


Fig. 6

3. *The DDPLAN routine*

Aim: To compute the direct distance (real) from a given point $(\overline{P})$ along a given direction $(\overline{\Delta S})$ to a given plane $(\overline{PL})$.

Process: the routine is defined by the following procedure

```
real proc  DDPLAN (point $\overline{P}$, vect $\overline{\Delta S}$ , plane $\overline{PL}$)
begin
    norm $\overline{SN}$; point $\overline{SP}$;
    $\overline{SN}$ = NORM-PLANE $(\overline{PL})$;(1)
    if $\overline{\Delta S} \times \overline{SN} = 0$(2) then *DDPLAN is not defined and a
    diagnostic will be printed* else
    $\overline{SP}$ = POINT-PLANE $(\overline{PL})$(3);
    DOPLAN = $(\overline{SP} \times \overline{SN} - \overline{P} \times \overline{SN})/(\overline{\Delta S} \times \overline{SN})$;(4)
end
```

1) This vector procedure computes the vector normal to the first plane $\overline{PL}$.

2) Where "$\times$" stands for the scalar product, and "$\wedge$", in the following, for the vector product of two vectors.

3) This procedure computes any wanted point on the plane $\overline{PL}$.

4) Infact, the direct distance λ from $\overline{P}$ to a plane is given by

$$(\overline{P} + \lambda\overline{\Delta S}) \times \overline{SN} = \overline{SP} \times \overline{SN}$$

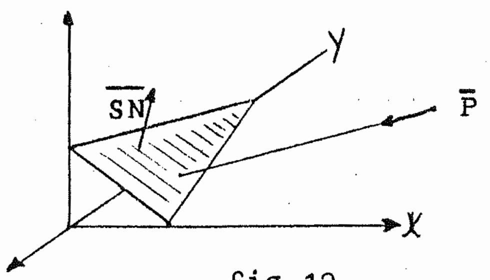


fig.12

4. The DDSPHR routine

Aim : to compute the direct distance (real) from a given point $(\overline{P})$ along a given direction $(\overline{\Delta S})$ to a given sphere $(\overline{SPHR})$ subordinately to a $\overline{FIOP}$ flag.

Process: the routine is defined by the following procedure

```
real proc  DDSPHR (point $\overline{P}$, vect $\overline{\Delta S}$, sphere $\overline{SPHR}$, flag $\overline{FIOP}$)
begin
    point $\overline{C}$; real $\overline{R}$;
    $\overline{C}$ = CENTER ($\overline{SPHR}$) (1);
    $\overline{R}$ = RADIUS ($\overline{SPHR}$) (2);
    DDSPHR = QUAD ( $\overline{\Delta S} \times \overline{\Delta S}, (\overline{P}-\overline{C}) \times \overline{\Delta S}, (\overline{P}-\overline{C}) \times (\overline{P}-\overline{C}) - \overline{R}^2, \overline{FIOP}$); (3)
end
```

1) This procedure computes the sphere center;

2) and this one the sphere radius.

3) Infact the direct distance λ from $\overline{P}$ to a sphere is given by

$|\overline{P} + \lambda \overline{\Delta S} - \overline{C}| = \overline{R}$

i.e., $(\overline{P} + \lambda \overline{\Delta S} - \overline{C}) \times (\overline{P} + \lambda \overline{\Delta S} - \overline{C}) - \overline{R}^2 = 0$

whence $\overline{A} \lambda^2 + 2 \overline{B} \lambda + \overline{C} = 0$

where
$$\overline{A} = \overline{\Delta S} \times \overline{\Delta S}$$
$$\overline{B} = (\overline{P} - \overline{C}) \times \overline{\Delta S}$$
$$\overline{C} = (\overline{P} - \overline{C}) \times (\overline{P} - \overline{C}) - \overline{R}^2$$

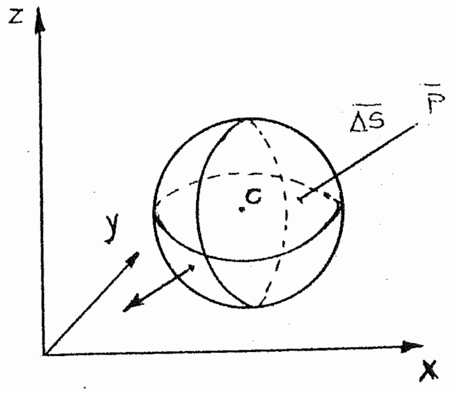


Fig. 8

5. *The QUAD routine*

Aim : to solve the equation $\bar{A}^2 + \overline{2B} + \bar{C} = 0$, and store one real root subordinately to a $\overline{FIOP}$ flag, provided roots are real, otherwise QUAD is not defined.

Remarks: when the input FIOP flag selection value is, either 0, or 1 , or -1 , then the real root stored is, respectively, either the least in absolute value, or the least positive (that has been found to exist), or the least positive if it exists otherwise the least in absolute value.

Process: the routine is defined by the following procedure

```
    real proc QUAD (real $\bar{A}$, real $\bar{\bar{B}}$, real $\bar{C}$, flag $\overline{FIOP}$)
    begin
        real $\bar{\lambda}_1, \bar{\lambda}_2$;
A:      if ($\bar{\bar{B}} \uparrow 2$ (1)$-4\ \overline{\overline{A}\overline{C}}$) < 0 then *QUAD is not defined and
```

1) the symbol $\uparrow$ means "raised to exponent".

```
    a diagnostic will be printed* else
C:  if  ($\bar{B}\uparrow 2 - 4\ \overline{AC}$) = 0 then
D:  begin
          QUAD = $-\bar{B}/\bar{A}$ ;
          goto 1;
    end
E:  $\bar{\lambda}_1$ = ($-\bar{B}$ + SQRT($\bar{B}\uparrow$ 2-4 $\overline{AC}$))/ $\bar{A}$ ;
    $\bar{\lambda}_2$ =($-\bar{B}$ - SQRT ($\bar{B}\uparrow$2 - 4 $\overline{AC}$ ))/ $\bar{A}$;
F:   if  $\overline{\text{FIOP}}$ = 0  then
G:   begin
          if $|\bar{\lambda}_1| \leq |\bar{\lambda}_2|$ then
          begin
                QUAD = $\bar{\lambda}_1$ ;
                goto 1 ;
          end
          else
          begin
                QUAD = $\bar{\lambda}_2$ ;
                goto 1;
          end
     end
     if $\overline{\text{FIOP}}$ = 1 then
H:   begin
```

```
        if $\bar{\lambda}_1 \geqslant 0$ $\wedge$(2) $\bar{\lambda}_2 \geqslant 0$ then
        begin
                QUAD = MIN ( $\bar{\lambda}_1$, $\bar{\lambda}_2$); (3)
                goto 1;
        end
        if $\bar{\lambda}_1 \geqslant 0$ then
        begin
                QUAD = $\bar{\lambda}_1$ ;
                goto 1 ;
        end
        if $\bar{\lambda}_2 \geqslant 0$ then
        begin
                QUAD = $\bar{\lambda}_2$;
                goto 1;
        end
        *a diagnostic will be printed*;
    end
I:  if  $\bar{\lambda}_1 \geqslant 0$  $\vee$(4) $\bar{\lambda}_2 \geqslant 0$ then goto H else goto G;
end
```

Remarks: the routine, after all very simple, is rendered visually by the following flow-chart

2) "$\wedge$" = "AND" = "&" = "$\cap$" in other well known notations.

3) This procedure selects the lesser value between two reals.

4) "$\vee$" = "OR" ="$\cup$" in other well known notations.

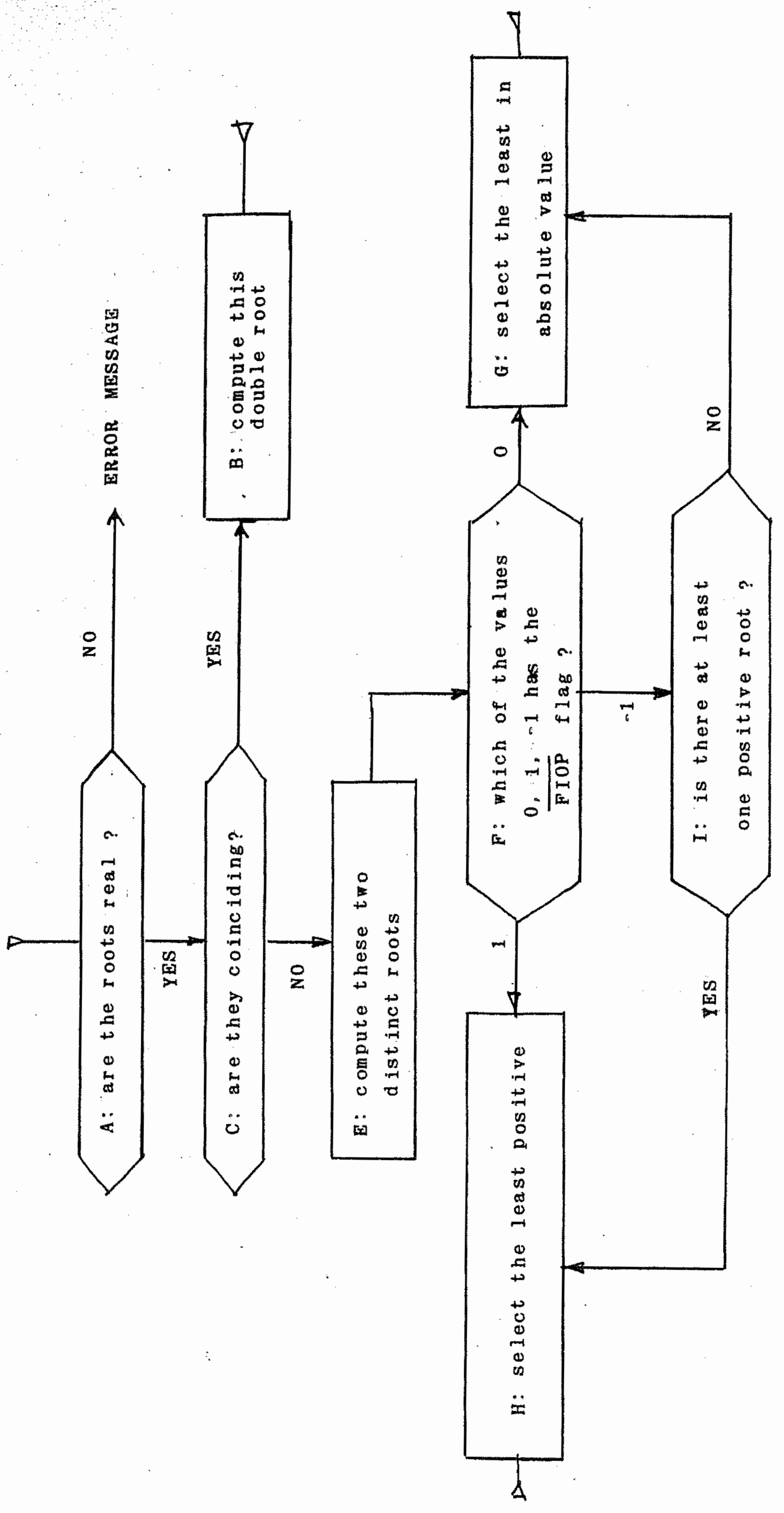

A: are the roots real ?
NO
ERROR MESSAGE
YES
C: are they coinciding?
YES
B: compute this double root
NO
E: compute these two distinct roots
F: which of the values 0, 1, -1 has the $\overline{\text{FIOP}}$ flag ?
0
G: select the least in absolute value
1
H: select the least positive
-1
I: is there at least one positive root ?
NO
YES

fig. 9

6. The DDCON routine

Aim: to compute the direct distance (real) from a given point ($\overline{P}$) along a given direction ($\overline{\Delta S}$) to a given conic surface ($\overline{CON}$) subordinately to a given ($\overline{FIOP}$) flag, as previously defined.

Process: the routine is defined by the following procedure

```
real proc DDCON (point $\overline{P}$, vect $\overline{\Delta S}$ , cone $\overline{CON}$, flag $\overline{FIOP}$
begin
        point $\overline{V}$ ;  vect $\overline{T}$ ; real $\overline{\alpha}$ , $\overline{K}$ ;
        $\overline{V}$ = VERTEX ($\overline{CON}$) (1) ;
        $\overline{T}$ = AXIS1 ($\overline{CON}$) ; (2)
        $\overline{\alpha}$ = ANGLE ($\overline{CON}$) ; (3)
        $\overline{K}$ = COS ($\overline{\alpha}$) ; (4)
```

1) This procedure computes the cone vertex $\overline{V}$. Actually here one deals with a semi-cone, i.e., generated by a half-line that projects a plane closed simple curve from a point, the vertex, external to the curve plane; therefore it is "internal" to the cone the direction from the vertex to any point on that plane internal to that curve; and, if that curve is a circle, the cone "aperture" is defined as a plane angle less than 180°.

2) This procedure computes the unit-vector $\overline{T}$ parallel to the cone axis, and internal to the cone if applied in the vertex.

3) This procedure computes the semi-aperture (acute) angle $\overline{\alpha}$ of the (circular) cone.

4) This procedure computes the cosine of $\overline{\alpha}$.

```
DDCON = QUAD {$(\overline{\Delta S} \times \overline{T}) \uparrow 2 - \overline{K} \uparrow 2\, \overline{\Delta S} \times \overline{\Delta S}$ ,
              $(\overline{\Delta S} \times \overline{T})[(\overline{P} - \overline{V}) \times \overline{T}] - \overline{K} \uparrow 2\, \overline{\Delta S} \times (\overline{P} - \overline{V})$,
              $[(\overline{P} - \overline{V}) \times \overline{T}] \uparrow 2 - \overline{K} \uparrow 2 (\overline{P} - \overline{V}) \times (\overline{P} - \overline{V})$, $\overline{FIOP}$} (5) ;

end
```

7. The DDCYLN routine

Aim: to compute the direct distance (real) from a given point $(\overline{P})$ along a given direction $(\overline{\Delta S})$ to a given cylindrical surface $(\overline{CYLN})$ subordinately to a given $\overline{FIOP}$ flag.

Process: the routine is defined by the following procedure

```
real proc DDCYLN (point $\overline{P}$, vect $\overline{\Delta S}$, cylinder $\overline{CYLN}$, flag $\overline{FIOP}$)
begin
    point $\overline{V}$; vect $\overline{T}$, real $\overline{r}_0$ ;
```

---

5) If $\overline{P} + \lambda_0 \overline{\Delta S}$ is a point on the cone, see fig. 10 a) below, then the direct distance λ_0 satisfies the equation in λ

$$|(\overline{P} + \lambda\, \overline{\Delta S} - \overline{V}) \times \overline{T}| = |\overline{P} + \lambda\, \overline{\Delta S} - \overline{V}|\, \overline{K}, \qquad 0 < K = \cos\alpha < 1$$

whence, squaring both sides,

$$[(\overline{P} + \lambda\, \overline{\Delta S} - \overline{V}) \times \overline{T}]^2 = \overline{K}^2 (\overline{P} + \lambda\, \overline{\Delta S} - \overline{V}) \times (\overline{P} + \lambda\, \overline{\Delta S} - \overline{V})$$

whence, finally, by ordering in decreasing λ powers, the coefficients thus found are the arguments, separated by comas, of the QUAD routine for DDCON.

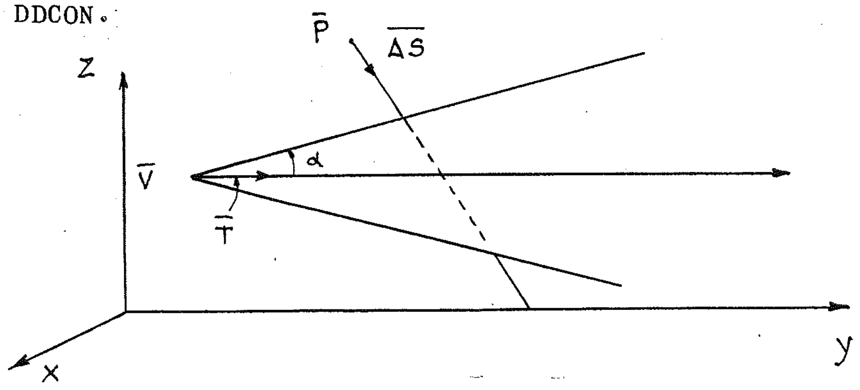


Fig.10 a)

$\overline{T}$ = AXIS2 $\overline{(CYLN)}$; (1)

$\overline{r}_o$ = RADIUS1 $\overline{(CYLN)}$; (2)

$\overline{V}$ = POINT-CYL $\overline{(CYLN)}$; (3)

DDCYLN = QUAD $(\overline{\Delta S} \times \overline{\Delta S} - (\overline{\Delta S} \times \overline{T}) \uparrow 2,$
$\overline{\Delta S} \times (\overline{P} - \overline{V}) - (\overline{\Delta S} \times \overline{T}) (\overline{P} - \overline{V}) \times \overline{T},$
$(\overline{P} - \overline{V}) \times (\overline{P} - \overline{V}) - [(\overline{P} - \overline{V}) \times \overline{T}] \uparrow 2 - \overline{r}_o \uparrow 2,$ $\overline{\text{FIOP}}$) ; (4)

<u>end</u>

---

1) this procedure computes the unit-vector $\overline{T}$ parallel to the cylinder axis;

2) this one, the cylinder radius r_o;

3) and this one, any point $\overline{V}$ on the cylinder axis.

4) if Q (see fig.10 b)) is one intersection of the cylindrical surface with the direction of the unit-vector $\overline{\Delta S}$, and λ_o the distance from $\overline{P}$ to the surface, then $Q = \overline{P} + \lambda_o \overline{\Delta S}$, and λ_o satisfies the equation in λ

$$|(\overline{P} + \lambda \overline{\Delta S} - \overline{V}) - [(\overline{P} + \lambda \overline{\Delta S} - \overline{V}) \times \overline{T}] \cdot \overline{T}| = \overline{r}_o,$$

whence, squaring both sides,

$$(\overline{P} + \lambda\overline{\Delta S} - \overline{V}) \times (\overline{P} + \lambda\overline{\Delta S} - \overline{V}) - 2[(\overline{P} + \lambda\overline{\Delta S} - \overline{V}) \times \overline{T}]^2 + [(\overline{P} + \lambda\overline{\Delta S} - \overline{V}) \times \overline{T}]^2 = r_o^2$$

where finally the coefficients of the decreasing λ powers are the arguments of the QUAD routine above.

$\lambda_o = |\overline{PQ}|$, $r_o = |\overline{OQ}|$

$\overline{OQ} = \overline{VQ} - \overline{VO}$

$\overline{VO} = \overline{VQ} \times \overline{T}\ \overline{T}$

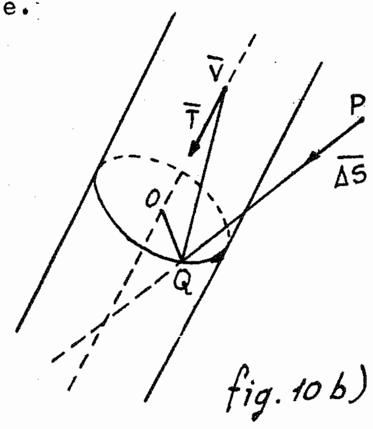


fig. 10 b)

8. The DDQUAD routine

Aim : to compute the direct distance from a point $(\overline{P})$ along the direction $(\overline{\Delta S})$ to a generic quadric $(\overline{QUAD})$ subordinately to a given $(\overline{FIOP})$ flag, as previously defined.

Process: the routine is defined by the following procedure

```
real proc DDQUAD (point $\overline{P}$, vect $\overline{\Delta S}$, quadric $\overline{QUAD}$, flag $\overline{FIOP}$)
begin
    real $\bar{x}_o,\bar{y}_o,\bar{z}_o,\bar{x}_n,\bar{y}_n,\bar{z}_n$; matrix $\overline{\Omega}$ ;
  $\overline{\Omega}$ = MATRIX $(\overline{QUAD})$;
   $\bar{x}_o$ = x - of $\overline{P}$ ;
   $\bar{y}_o$ = y - of $\overline{P}$ ;
   $\bar{z}_o$ = z - of $\overline{P}$ ;
   $\bar{x}_n$ = x - of $\overline{\Delta S}$ ;
   $\bar{y}_n$ = y - of $\overline{\Delta S}$ ;
   $\bar{z}_n$ = z - of $\overline{\Delta S}$ ;
   DDQUAD = QUAD [$(\bar{x}_n,\bar{y}_n,\bar{z}_n,0)\cdot\overline{\Omega}\cdot(\bar{x}_n,\bar{y}_n,\bar{z}_n,0)^T$,
                 $(\bar{x}_n,\bar{y}_n,\bar{z}_n,0)\cdot\overline{\Omega}\cdot(\bar{x}_o,\bar{y}_o,\bar{z}_o,1)^T$,
                 $(\bar{x}_o,\bar{y}_o,\bar{z}_o,1)\cdot\overline{\Omega}\cdot(\bar{x}_o,\bar{y}_o,\bar{z}_o,1)^T$, $\overline{FIOP}$;](2)
end
```

1) this procedure stores the square matrix $\Omega = [c_{ij}]$, $i=1,2,3,4$; $j=1,2,3,4$; $c_{ij}=c_{ji}$; to represent the quadric surface of equation

a) $\Sigma\, c_{ij}\, x_i\, x_j = 0$, which takes the well known typical form in non-homogeneous coordinates by putting $x=x_1$, $y=x_2$, $z=x_3$, $1=x_4$, and $c_{ij}=a,h,g,p,b,f,q,c,r,d$; if $(ij)=(ji)=11,12,\ldots,44$;

whence a) becomes

$(x,y,z,1)\cdot\Omega\cdot(x,y,z,1)^T = 0.$

2) Having put $\overline{P}_o$ for $\overline{P}$, and $\overline{P}_n$ for $\overline{P}_o+\overline{\Delta S}$, that value λ_o such that the parametric point $(\bar{x}_o,\bar{y}_o,\bar{z}_o) + \lambda\ (\bar{x}_n,\bar{y}_n,\bar{z}_n)$ belongs to the quadric surface and λ_o itself is the direct distance required, is one root, as selected by $\overline{FIOP}$, of the quadratic equation in λ whose coefficients in their order are the arguments of the QUAD routine above.

9. The DDTABC routine

Aim: to compute the direct distance λ_o from the point $(\overline{P})$ along the direction $(\overline{\Delta S})$ to a given tabulated cylinder $(\overline{TABC})$ subordinately to a given $(\overline{FIOP})$ flag, the point $\overline{Q} = \overline{P} + \lambda_o\overline{\Delta S}$ on such a cylinder, and the normal $(\overline{SN})$ to the cylinder in such a point Q.

Remarks: the tabulated cylinder is a special surface considered in APT (see ref [7], pag.03-04) as suitably defined by some given points (*), not analytically defined; therefore the following computational procedures are of an empirical character, often very laborious.

Process: the routine is defined by the following procedure

(*) co-planar; therefore the cylinder is defined by a plane curve ("directrix") that suitably interpolates the point cluster, and by a direction.

```
tris = (real $\bar{S}$, point $\bar{Q}$, norm $\overline{SN}$ )
tris proc. DDTABC (point $\bar{P}$, vect $\overline{\Delta S}$,tabcyl $\overline{TABC}$,flag $\overline{FIOP}$)
begin
    real $\overline{UP},\overline{VP},\overline{U\Delta S},\overline{V\Delta S},\bar{M},\bar{Q},\bar{U}_i,\bar{V}_i,\bar{d},\bar{\alpha},\bar{w}_i,\overline{RI},\overline{SI},\overline{TI},\overline{MI},\overline{QI}$;
    integer $\bar{I},\bar{n},^{(2)}$
    tricomplex $\overline{ABC}$ ; bis $\overline{ABO}$; point $\overline{AI},\overline{BI},\overline{CI},\overline{DI}$;
    $\overline{UP}$ = u-of $\bar{P}$ ; (1)
    $\overline{VP}$ = v-of $\bar{P}$;
    $\overline{U\Delta S}$ = u-of $\overline{\Delta S}$;
    $\overline{V\Delta S}$ = v-of $\overline{\Delta S}$;
A:  if ($|\overline{U\Delta S}|$ + $|\overline{V\Delta S}| < 10^{-6}$ ) then *DDTABC is not defined
    and a diagnostic will be printed* else
    $\bar{I}$ = 1
B:  if $|\overline{V\Delta S}| > |\overline{U\Delta S}| \cdot 10^5$ then goto C else goto D
    begin
        point-array $\overline{P1}$ [1:$\bar{n}$-1],$\overline{P2}$[1:$\bar{n}$-1],$\overline{P3}$[1:$\bar{n}$-1]$^{(2)}$;
```

1) the (u,v,w) coordinate system is the "reference system" of the tabulated cylinder, while the (x,y,z) system is reference system for the "part-program"; the two origins coincide, the w-axis is parallel to the cylinder axis, the u-, v-axes being suitably chosen.

2) $\bar{n}$ is the number of the given points defining the tabulated cylinder.

```
C:      $\overline{AI}$ = ARECT ($\overline{TABC}$ , $\bar{i}$) ;(3)
        $\overline{BI}$ = BRECT ($\overline{TABC}$ , $\bar{i}$) ;
        $\overline{CI}$ = CRECT ($\overline{TABC}$ , $\bar{i}$) ;
        $\overline{DI}$ = DRECT ($\overline{TABC}$ , $\bar{i}$) ;
```

3) this procedure operates as follows: given the points $\overline{P}_1, \overline{P}_2, \ldots, \overline{P}_n$ on the (u,v)-plane, it defines the segment L_o through $\overline{P}_1$ and $\overline{P}_n$, and, for $\overline{P}_{\bar{i}}, \overline{P}_{\bar{i}+1}$, see fig.11, defines the rectangle with opposite vertices in $\overline{P}_{\bar{i}}, \overline{P}_{\bar{i}+1}$ and sides respectively parallel and perpendicular to L_o.

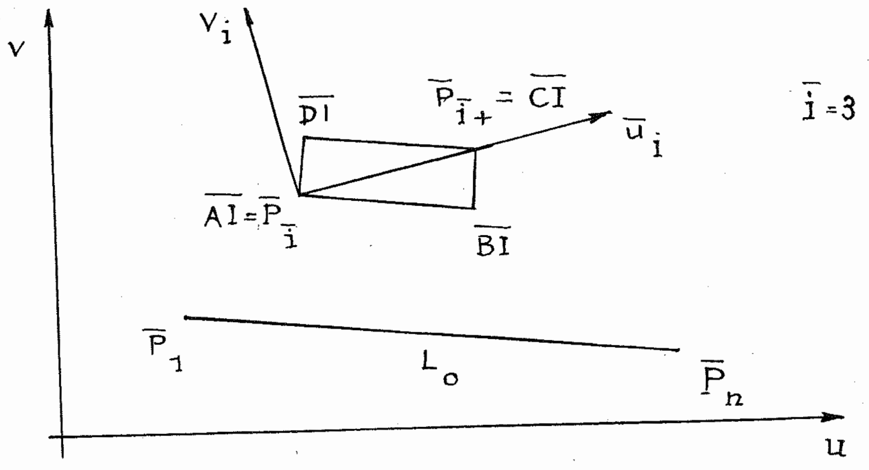


Fig. 11

The ARECT routine computes the point $\overline{AI} = \overline{P}_{\bar{i}}$, and the CRECT routine the point $\overline{CI} = \overline{P}_{\bar{i}+1}$; the other two routines, respectively, the other two vertices $\overline{BI}$ and $\overline{DI}$.

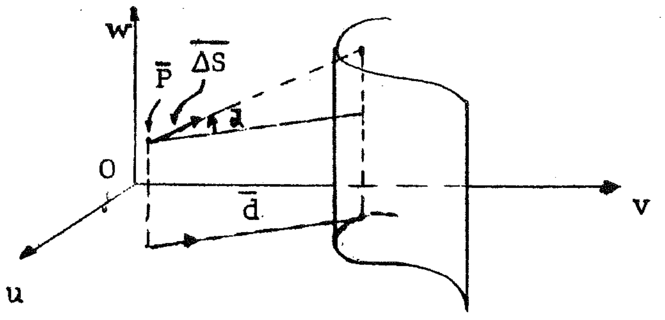
```
if $(\overline{UP} - \underline{u\text{-}of}\ \overline{AI}) = 0$ then goto E
if $(\overline{UP} - \underline{u\text{-}of}\ \overline{BI}) = 0 \ \underline{\vee}\ (\overline{UP} - \underline{u\text{-}of}\ \overline{AI})(\overline{UP} - \underline{u\text{-}of}\ \overline{BI}) < 0$
    then goto E
if $(\overline{UP} - \underline{u\text{-}of}\ \overline{CI}) = 0 \ \underline{\vee}\ (\overline{UP} - \underline{o\text{-}of}\ \overline{AI})(\overline{UP} - \underline{u\text{-}of}\ \overline{CI}) < 0$
    then goto E
if $(\overline{UP} - \underline{u\text{-}of}\ \overline{DI}) = 0 \ \underline{\vee}\ (\overline{UP} - \underline{o\text{-}of}\ \overline{AI})(\overline{UP} - \underline{u\text{-}of}\ \overline{DI}) < 0$
    then goto E
if $\bar{i} = \bar{n} - 1$ then goto Q
$\bar{i} = \bar{i} + 1;$
goto C ;
$\overline{AI} = \text{ARECT}\ (\overline{TABC},\ \bar{i})^{(3)};$
$\overline{BI} = \text{BRECT}\ (\overline{TABC},\ \bar{i});$
$\overline{CI} = \text{CRECT}\ (\overline{TABC},\ \bar{i});$
$\overline{DI} = \text{DRECT}\ (\overline{TABC},\ \bar{i});$
$\bar{M} = \overline{V\Delta S}\ /\ \overline{U\Delta S}\ ;$
$\bar{Q} = \underline{v\text{-}of}\ \bar{P} - (\underline{u\text{-}of}\ \bar{P}) \cdot \bar{M};$
if $(\underline{v\text{-}of}\ \overline{AI} - \bar{M}\ \underline{u\text{-}of}\ \overline{AI} - \bar{Q}) = 0$ then goto E
if$(\underline{v\text{-}of}\ \overline{BI} - \bar{M}\ \underline{u\text{-}of}\ \overline{BI} - \bar{Q}) = 0 \ \underline{\vee}\ (\underline{v\text{-}of}\ \overline{AI} - \bar{M}\ \underline{u\text{-}of}\ \overline{AI} - \bar{Q})(\underline{v\text{-}of}\ \overline{BI} +$
                    $- \underline{u\text{-}of}\ \overline{BI} - \bar{Q})]$ then goto E
if$(\underline{v\text{-}of}\ \overline{CI} - \bar{M}\ \underline{u\text{-}of}\ \overline{CI} - \bar{Q}) = 0 \ \underline{\vee}\ (\underline{v\text{-}of}\ \overline{AI} - \bar{M}\ \underline{u\text{-}of}\ \overline{AI} - \bar{Q})(\underline{v\text{-}of}\ \overline{BI} +$
                    $- \underline{u\text{-}of}\ \overline{CI} - \bar{Q})]$ then goto E
if$(\underline{v\text{-}of}\ \overline{DI} - \bar{M}\ \underline{u\text{-}of}\ \overline{DI} - \bar{Q}) = 0 \ \underline{\vee}\ (\underline{v\text{-}of}\ \overline{AI} - \bar{M}\ \underline{u\text{-}of}\ \overline{AI} - \bar{Q})(\underline{v\text{-}of}\ \overline{DI} +$
                    $- \underline{u\text{-}of}\ \overline{DI} - \bar{Q})]$ then goto E
```

```
D1:     if $\bar{i} = \bar{n} - 1$   then goto  Q
        $\bar{i} = \bar{i} + 1$;
        goto D ;
E:      $\overline{RI}$  = ACOEFF ($\overline{TABC}$, $\bar{i}$)(4);
        $\overline{SI}$  = BCOEFF ($\overline{TABC}$, $\bar{i}$);
        $\overline{TI}$  = CCOEFF ($\overline{TABC}$, $\bar{i}$);
F:      if $|$ $\underline{u_i\text{-of}}$ $\overline{\Delta S}| < 10^{-6}$ then goto M else goto N
M:      $\bar{U}_i$ = $\underline{u_i\text{-of}}$ $\bar{P}$ ;
        $\bar{V}_i = \overline{RI}\,(\bar{U}_i \uparrow 3) + \overline{SI}\,(\bar{U}_i \uparrow 2) + \overline{TI} \cdot \bar{U}_i$ ;
9:      $\bar{d}$ = SQRT [ ($\bar{U}_i$-$\underline{u_i\text{-of}}$  $\bar{P}$) $\uparrow$ 2 + ($\bar{V}_i$ - $\underline{v_i\text{-of}}$  $\bar{P}$) $\uparrow$ 2](5);
        $\bar{\alpha}$ = ANGLE - PL ( $\overline{\Delta S}$ , $\overline{TABC}$);(6)
```

4) The tabulated cylinder directrix is represented by a sequence of cubic parabolas of equation $v = au^3 + bu^2 + cu$, one for each $\overline{TABC}$ segment P_i, P_{i+1}, each containing the origin, which is made to coincide with the left extreme P_i of each segment , shifting the reference at each step. Therefore, three more points suffice to define a cubic parabola for each value of $\bar{i}$, and they are P_{i-1}, P_{i+1}, P_{i+2}, with suitable definitions for the end-points of the directrix. The procedure in question and the two ensuing ones compute precisely the coefficients, a,b,c, for the cubic parabola that "interpolates", as defined, the ith segment. The sequence of such interpolations from i = 1 to i = n is the so called "fairing" method.

5) This procedure computes a Pythagorean distance.

6) This procedure computes the angle α, counterclockwise, from the reference uv-plane to the versor $\overline{\Delta S}$, see fig.12 below :



```
        $\bar{W}_i$ = w-of  $\bar{P}$ + $\bar{d}$ TANG ($\bar{\alpha}$) (7)
        $\overline{P1}[\bar{i}]$ = POINT ($\bar{U}_i$, $\bar{V}_i$, $\bar{W}_i$) (8)
MI:   : if  $\bar{i}$ = $\bar{n}$ - 1  then goto Q else
        begin
            $\bar{i}$ = $\bar{i}$ + 1
            goto B
        end
N:      $\overline{MI}$ = $v_i$-of $\overline{\Delta S}$ / $u_i$-of $\overline{\Delta S}$
        $\overline{QI}$ = $v_i$-of $\bar{P}$ - ($u_i$-of $\bar{P}$)$\cdot$ $\overline{MI}$
        $\overline{ABC}$ = EQUAZ 3 ($\overline{RI}$,$\overline{SI}$,$\overline{TI}$ -$\overline{MI}$,-$\overline{QI}$) (9)
        $\overline{U1}$ = $\overline{U1}$ of $\overline{ABC}$
        $\overline{U2}$ = $\overline{U2}$ of $\overline{ABC}$
        $\overline{U3}$ = $\overline{U3}$ of $\overline{ABC}$
```

---

7) this procedure computes the $\overline{\Delta S}$ slope,i.e., real proc TANG (reala);

8) and this one stores the points where the line through $\bar{P}$ along $\overline{\Delta S}$ intersects the tabulated cylinder.

9) This procedure computes the three complex roots of

$$Au^3 + Bu^2 + Cu + D = 0$$

to satisfy the simultaneous conditions

$$\begin{cases} \overline{UI} = \overline{RI} \cdot \overline{UI}^3 + \overline{SI} \cdot \overline{UI}^2 + \overline{TI} \cdot \overline{UI} \\ \overline{VI} = \overline{MI} \cdot \overline{VI} + \overline{QI} \end{cases}$$

for $A = \overline{RI}$, $B = \overline{SI}$, $C = \overline{TI} - \overline{MI}$, $D = -\overline{QI}$

```
    if $\overline{UI}$ is a real number $\wedge$ $(0 \leq \overline{UI} \leq |\bar{P}_{\bar{i}+1} - \bar{P}_{\bar{i}}|)$ then
    begin
        $\overline{VI} = \overline{RI}\ (\overline{UI} \uparrow 3) + \overline{SI}\ (\overline{UI} \uparrow 2) + \overline{TI}\ \overline{UI}$
        $\bar{d} = SQRT\ [(\overline{UI} - u_{\bar{i}}\text{-of } \bar{P}) \uparrow 2 + (\overline{VI} - v_{i}\text{-of } \bar{P}) \uparrow 2]$
        $\bar{\alpha} = ANGLE - PL\ (\overline{\Delta S}, \overline{TABC})$
        $\overline{WI} = w\text{-of } \bar{P} + \bar{d}\ TANG\ (\bar{\alpha})$
        $\overline{PI}[\bar{i}] = POINT\ (\overline{UI}, \overline{VI}, \overline{WI})$
        goto 10
    end
10: if $\overline{U2}$ is a real number $\wedge$ $(0 \leq \overline{U2} \leq |\bar{P}_{\bar{i}+1} - \bar{P}_{\bar{i}}|)$ then
    begin
        $\overline{V2} = \overline{RI}\ (\overline{U2} \uparrow 3) + \overline{SI}\ (\overline{U2} \uparrow 2) + \overline{TI}\ \overline{U2}$ ;
        $\bar{d} = SQRT[(\overline{U2} - u_{i}\text{-of } \bar{P}) \uparrow 2 + (\overline{U2} - v_{i}\text{-of } \bar{P}) \uparrow 2]$;
        $\bar{\alpha} = ANGLE\text{-}PL\ (\overline{\Delta S}, \overline{TABC})$;
        $\overline{W2} = w\text{-of } \bar{P} + \bar{d}\ TANG\ (\bar{\alpha})$;
        $\overline{P2}[\bar{i}] = POINT\ (\overline{U2}, \overline{V2}, \overline{W2})$ ;
        goto 11;
    end
11: if $\overline{U3}$ is a real number $\wedge$ $(0 \leq \overline{U3} \leq |\bar{P}_{\bar{i}+1} - \bar{P}_{\bar{i}}|)$ then
    begin
        $\overline{V3} = \overline{RI}\ (\overline{U3} \uparrow 3) + \overline{SI}\ (\overline{U3} \uparrow 2) + \overline{TI}\ \overline{U3}$ ;
        $\bar{d} = SQRT\ [(\overline{U3} - u_{\bar{i}}\text{-of } \bar{P}) \uparrow 2 + (\overline{V3} - v_{i}\text{-of } \bar{P}) \uparrow 2]$ ;
```

```
            $\bar{\alpha}$  = ANGLE-PL ($\overline{\Delta S}$ , $\overline{TABC}$) ;
            $\overline{W3}$ = w-of  $\bar{P}$ + $\bar{d}$ TANG ($\bar{\alpha}$) ;
            $\overline{P3}[\bar{i}]$ = POINT ($\overline{U3}$, $\overline{V3}$, $\overline{W3}$);
            goto 12;
        end
P:       if $\bar{i}$ = n - 1  then goto Q else
         begin
             $\bar{i}$ = $\bar{i}$ + 1 ;
             goto B ;
         end
Q:       ABO = DIFF ($\bar{P}$, $\overline{\Delta S}$, $\overline{P1}$[1],..$\overline{P1}$[n], $\overline{P2}$[1],....,$\overline{P2}$[n],
              $\overline{P3}$[1] .... $\overline{P3}$[n], $\overline{FIOP}$ )(10);
         $\bar{S}$ of DDTABC = $\bar{S}$ of $\overline{ABO}$ ;
         $\bar{Q}$ of DDTABC = $\overline{PI}$  of $\overline{ABO}$;
         $\overline{SN}$  of DDTABC = NORM-TAB ($\bar{Q}$, $\overline{TABC}$) (11);
    end
end
```

Remark: see the flow-chart below as an illustration of this complex DDTABC routine

10) This procedure selects among the points already stored (see the previous footnote n. 8), some of which possibly not defined, that point $\overline{PI}$ whose distance $\bar{S}$ from $\bar{P}$ along $\overline{\Delta S}$ verifies the condition assigned by the $\overline{FIOP}$ flag to be the direct distance.

11) This procedure computes a normal line to the tabulated cylinder in its point $\bar{Q}$, as normal to the cubic parabola (perpendicular to its tangent) in the neighborhood of $\bar{Q}$, where in fact the interpolating curve is a tract of the cylinder directrix.

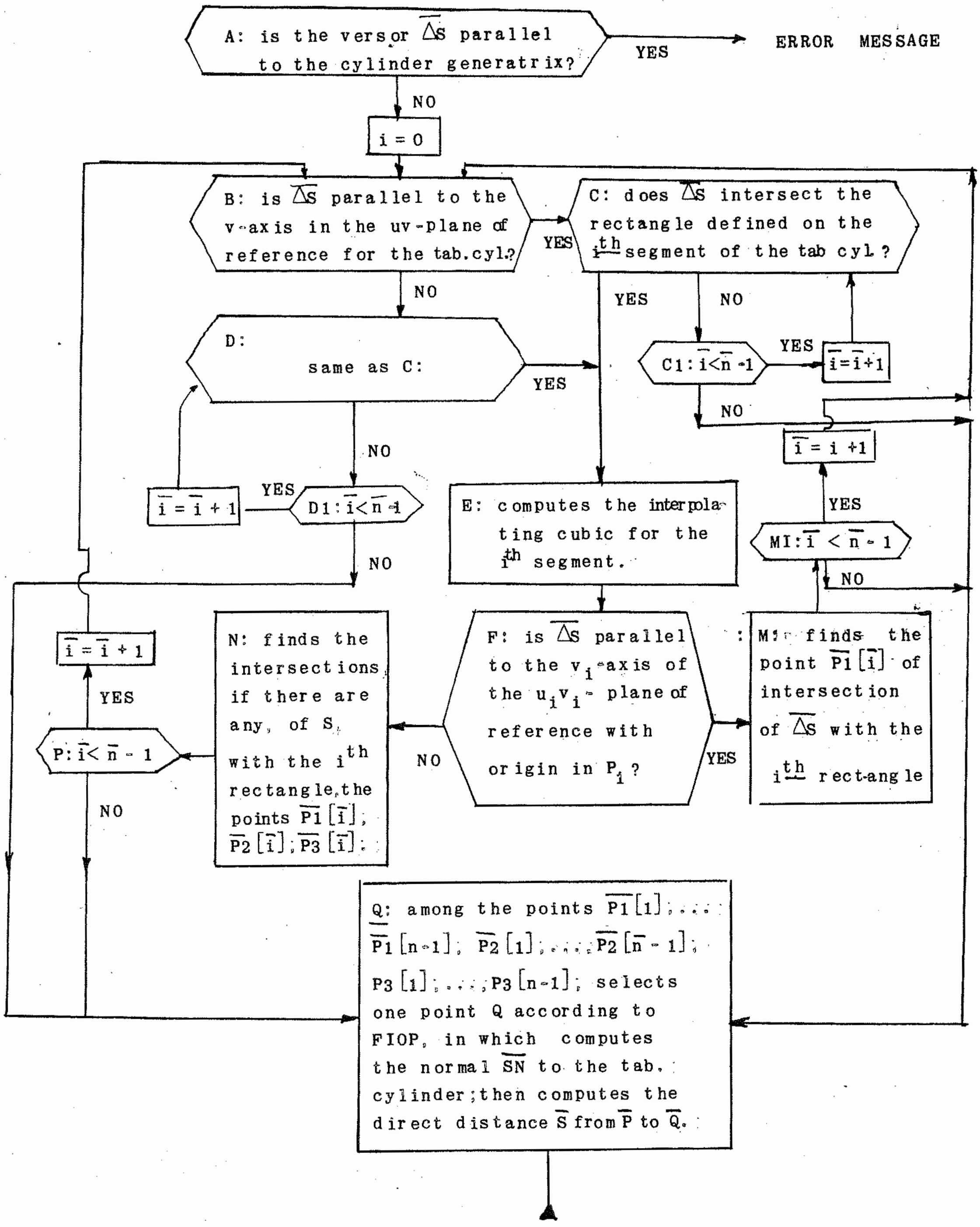

A: is the versor $\overline{\Delta s}$ parallel to the cylinder generatrix?
YES
ERROR MESSAGE
NO
i = 0
B: is $\overline{\Delta s}$ parallel to the v-axis in the uv-plane of reference for the tab.cyl.?
YES
C: does $\overline{\Delta s}$ intersect the rectangle defined on the ith segment of the tab cyl.?
NO
YES
NO
D: same as C:
YES
C1: $\bar{i}<\bar{n}-1$
YES
$\bar{i}=\bar{i}+1$
NO
$\bar{i}=i+1$
NO
YES
$\bar{i}=\bar{i}+1$
D1: $\bar{i}<\bar{n}-1$
E: computes the interpolating cubic for the ith segment.
YES
MI: $\bar{i}<\bar{n}-1$
NO
NO
$\bar{i}=\bar{i}+1$
N: finds the intersections if there are any, of S, with the ith rectangle, the points $\overline{P1}[\bar{i}]$; $\overline{P2}[\bar{i}]$; $\overline{P3}[\bar{i}]$;
F: is $\overline{\Delta s}$ parallel to the v_i-axis of the u_iv_i- plane of reference with origin in P_i?
M: finds the point $\overline{P1}[\bar{i}]$ of intersection of $\overline{\Delta s}$ with the ith rect-angle
YES
NO
YES
P: $\bar{i}<\bar{n}-1$
NO
Q: among the points $\overline{P1}[1]$;...; $\overline{P1}[n-1]$; $\overline{P2}[1]$;...;$\overline{P2}[\bar{n}-1]$; P3 [1];...;P3 [n-1]; selects one point Q according to FIOP, in which computes the normal $\overline{SN}$ to the tab. cylinder; then computes the direct distance $\overline{S}$ from $\overline{P}$ to $\overline{Q}$.

Fig. 13

10. The UNRMAL routine

Aim: to compute the unit-vector (vect) normal to a given surface (SF) in its point (SP) or in some point (SP) close to (SF).

Remarks: as previously remarked for the DDST routine, also this one operates by transfering control to some subroutine, each appropriate to the surface type in question.

Process: the following procedure defines the routine

```
vect proc  UNRMAL (point $\overline{SP}$, surf $\overline{SF}$)
begin
    if  IS-PLAN ($\overline{SF}$) then
    begin
        UNRMAL = UNPLAN ($\overline{SP}$, $\overline{SF}$),
        goto 1;
    end
    if IS-SPHR ($\overline{SF}$) then
    begin
        UNRMAL=UNSPHR ($\overline{SP}$, $\overline{SF}$),
        goto 1;
    end
    if IS-CON ($\overline{SF}$) then
    begin
```

```
        UNRMAL=UNCON ($\overline{SP}$ , $\overline{SF}$);
        goto 1;
    end
    if IS-CYLN ($\overline{SF}$) then
    begin
        UNRMAL=UNCYLN ($\overline{SP}$, $\overline{SF}$);
        goto 1;
    end
    if IS-QUAD ($\overline{SF}$) then
    begin
        UNRMAL = UNQUAD ($\overline{SP}$, $\overline{SF}$);
        goto 1;
    end
    if IS-TABC ($\overline{SF}$) then
    begin
        UNRMAL = NORM-TAB ($\overline{SP}$, $\overline{SF}$);
        goto 1;
    end
   **a diagnostic will be printed*;
1:end
```

11. The UNPLAN routine

Aim: to compute the unit vector (vect) normal to a given plane (PL) in its point (SP) or in some point (SP) close to (PL).

Process: the routine is defined by the following procedure

```
vect proc UNPLAN   (point SP, plane PL)
begin
        UNPLAN = NORM-PLANE  (PL) ;
end
```

12. The UNSPHR routine

Aim: to compute the unit- vector (vect) normal to a given sphere ($\overline{SPHR}$) in its point ($\overline{SP}$) or in some point close to the sphere.

Process: the following procedure defines the routine

```
vect proc UNSPHR (point $\overline{SP}$, sphere $\overline{SPHR}$)
begin
        point $\overline{C}$ ;
        $\overline{C}$ = CENTER  ($\overline{SPHR}$) ;
        if $\overline{SP}$ = $\overline{C}$ then* a diagnostic will be printed*  else
        UNSPHR = ($\overline{SP}$ - $\overline{C}$)/($\overline{SP}$ - $\overline{C}$) ;
```

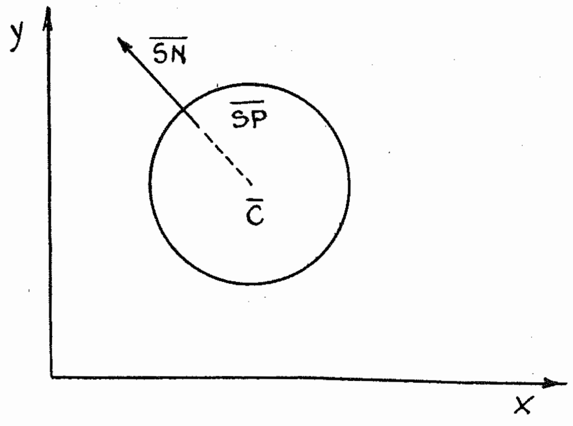


Fig.14

```
end
```

13. The UNCON routine

Aim: to compute the unit-vector (vect) normal to a given cone ($\overline{CON}$) in some point ($\overline{SP}$) on the cone or close to it.

Process: the following procedure defines the routine

```
vect proc UNCON (point $\overline{SP}$, cone $\overline{CON}$)
begin
    point $\overline{V}$; vect $\overline{T}$, $\overline{U}$; real $\overline{\alpha}$, $\overline{K}$, $\overline{\vartheta}$, $\overline{a}$ , $\overline{b}$;
    $\overline{V}$ = VERTEX ($\overline{CON}$);
    $\overline{T}$ = AXIS1 ($\overline{CON}$);
    $\overline{\alpha}$ = ANGLE ($\overline{CON}$);
    $\overline{K}$ = COS ($\overline{\alpha}$ );
    $\overline{U}$ = ($\overline{SP}$ - $\overline{V}$)/($\overline{SP}$ - $\overline{V}$ ) ;
    $\overline{\vartheta}$ = UTANGLE ($\overline{U}$, $\overline{T}$); (1)
```

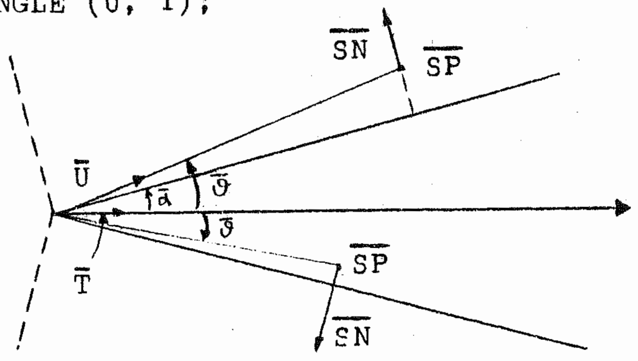


Fig.15

1) this procedure computes the acute angle of the vectors $\overline{U}$ and $\overline{T}$.

```
    if $\bar{\vartheta}=0$ $\wedge$ $|\bar{\alpha} - \bar{\vartheta}| \geqslant \frac{\pi}{2}$ then *a diagnostic will be
    printed * else
    $\bar{b}$ = COS ( $\bar{\alpha}$ ) / SEN ( $\bar{\vartheta}$ ); (2)
    $\bar{a}$ = - COS ( $\bar{\alpha} - \bar{\vartheta}$ )/SEN ( $\bar{\vartheta}$ );
    UNCON = $\bar{a}\,\bar{T} + \bar{b}\,\bar{U}$ ;
end
```

2) If the point $\overline{SP}$ is external to the cone axis and there exists the normal $\overline{SN}$ through $\overline{SP}$ to the cone, then $\overline{SN} = a\overline{T} + b\overline{U}$, where these coefficients are the roots of

$$\begin{cases} a + b \cos \theta = - \sin \alpha \\ a^2 + 2\, ab \cos \theta + b^2 = 1 \end{cases}$$

equivalent to the following conditions

$$\overline{SN} \times \overline{T} = - \sin \alpha$$

$$\overline{SN} \times \overline{SN} = 1$$

that the normal $\overline{SN}$ is to satisfy. One easily finds

$$b = \frac{\cos \alpha}{\sin \theta} \quad , \qquad a = - \frac{\cos (\alpha - \theta)}{\sin \theta}$$

where the sign for b is determined by noticing that the angle of $\overline{T}$ with the plane defined by the applied vectors $\overline{U}$ and $\overline{SN}$ is an acute angle.

14. The UNCYLN routine

Aim: to compute the unit-vector (vect) normal to a given cylinder (CYLN) in some given point (SP) of the cylinder or close to it.

Process: the following procedure defines the routine

```
vect proc UNCYLN (point $\overline{SP}$, cylinder $\overline{CYLN}$)
begin
        point $\bar{V}$; vect $\bar{T}$ ; real $\bar{r}_o$ ;
        $\bar{V}$ = POINT-CYL($\overline{CYLN}$) ;
        $\bar{T}$ = AXIS2 ($\overline{CYLN}$);
        $\bar{r}_o$ = RADIUS 1 ($\overline{CYLN}$);
        if $[(\overline{SP} - \bar{V}) \times \bar{T}]\,\bar{T} = \overline{SP} - \bar{V}$ (1) then *a diagnostic
        will be printed* else
        UNCYLN = $\{(\overline{SP} - \bar{V}) - [(\overline{SP} - \bar{V}) \times \bar{T}]\,\bar{T}\} / |(\overline{SP} - \bar{V}) +$
                                        $-[(\overline{SP} - \bar{V}) \times \bar{T}]\,\bar{T}|$;(2)
end
```

1) this case occurs if SP is on the cylinder axis.

2) where the normal (see figure below) points outward from the cylinder

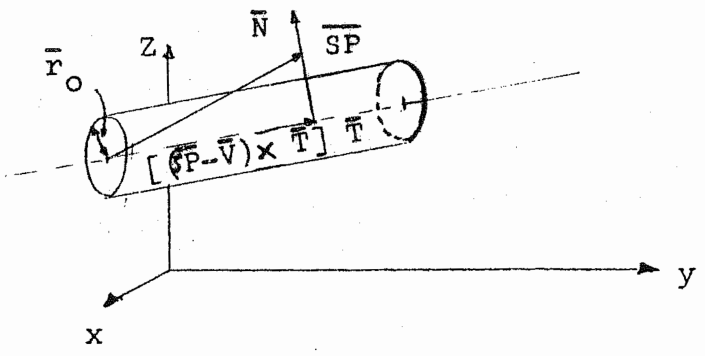


Fig.15 bis)

15. The UNQUAD routine

Aim: to compute the unit-vector (vect) normal to a given quadric surface ($\overline{QUAD}$) in a given point ($\overline{SP}$) on it or close to it.

Process: the following procedure defines the routine

```
vect proc UNQUAD (point $\overline{SP}$, quadric $\overline{QUAD}$)
begin
    real $\bar{x}_o$, $\bar{y}_o$, $\bar{z}_o$, $\bar{A}$, $\bar{B}$, $\bar{C}$, $\bar{D}$; quareal $\overline{ABC}$ ; matrix $\bar{\Omega}$
    $\bar{\Omega}$ = MATRIX ($\overline{QUAD}$) ;
    $\bar{x}_o$ = x - of $\bar{P}$;
    $\bar{y}_o$ = y - of $\bar{P}$;
    $\bar{z}_o$ = z - of $\bar{P}$;
    $\overline{ABC}$ = PL-EQUAZ[$(\bar{x}_o, \bar{y}_o, \bar{z}_o, 1)\cdot\Omega\cdot(x,y,z,1)^T = 0$]  (1) ;
    $\bar{A}$ = $\bar{A}$ of $\overline{ABC}$ ;
    $\bar{A}$ = $\bar{B}$ of $\overline{ABC}$ ;
    $\bar{C}$ = $\bar{C}$ of $\overline{ABC}$ ;

    UNQUAD = VECTOR ($\bar{A}$/SQRT($\bar{A}$↑2+ $\bar{B}$↑2 + $\bar{C}$ ↑ 2),
                     $\bar{B}$/SQRT($\bar{A}$)↑2+$\bar{B}$↑2 + $\bar{C}$ ↑ 2),
                     $\bar{C}$/SQRT($\bar{A}$)↑2+$\bar{B}$↑2 + $\bar{C}$ ↑ 2 ); (2)
end
```

1) this procedure computes the four coefficients of the equation for the plane polar to the point $\overline{SP}$ with respect to the given quadric; this equation is precisely the argument of the routine in question, as well known;

2) these coefficients, as also well known, are proportional to the director cosines of the normal to the plane, i.e., of the unit vector to be computed; which is done by this VECTOR procedure.

16. The AJUNDO routine

Aim: given a point $(\overline{P})$, a direction $(\overline{\Delta S})$, and a surface $(\overline{SF})$ to compute a) the point $(\overline{SP})$ on $(\overline{SF})$ such that, subordinately to a $(\overline{FIOP})$ flag, the distance from $(\overline{P})$ to it is the direct distance S from $(\overline{P})$ to $(\overline{SF})$; and b) the normal $(\overline{SN})$ to $(\overline{SF})$ in $(\overline{SP})$, suitably oriented with respect to $(\overline{\Delta S})$ by a certain $(\overline{FCON})$ control. A binary pointer $(\overline{Z})$ shows whether the $(\overline{SN})$ direction,as initially oriented, has been inverted.

Remarks: For the "start-up"-, or the "drive" or "part"- surfaces we want $\overline{SN} \times \overline{\Delta S} \leqslant 0$, therefore $(\overline{FCON})$ is set at the value 0 for so orienting the normal in such cases; for the "check" surfaces, instead, we want $\overline{SN} \times \overline{\Delta S} > 0$, whence $(\overline{FCON})$ is correspondingly set at the other value 1; then, according to whether $(\overline{FCON})$ either confirms or inverts the initially determined $(\overline{SN})$ orientation, $(\overline{Z})$ is made to show either the value 1 or -1.

Process: the routine is defined by the following procedure

```
        ajundo = (real S̄, point S̄P̄ , norm S̄N̄, indicator Z̄)
        ajundo proc  AJUNDO (point P̄, vect Δ̄S̄ , surf  S̄F̄,
        flag F̄ĪŌP̄, control F̄C̄ŌN̄)
  begin
        integer  ī; vect Δ̄S̄';
        ī = 0
        Δ̄S̄' = Δ̄S̄ ;
  3:    if DDST (P̄, Δ̄S̄', S̄F̄, F̄ĪŌP̄) fails then
        begin
              ī = ī + 1;
                goto 1;
        end
        goto 2
```

```
1:  if $\bar{I}$ = 15 then "a diagnostic will be printed" else
    begin
        $\overline{\Delta S}'$ = RADAR ($\bar{P}$, $\overline{\Delta S}$, $\bar{i}$);
        goto 3;
    end
2:  $\bar{S}$ of AJUNDO = $\bar{S}$ of DDST ($\bar{P}$, $\overline{\Delta S}'$, $\overline{SF}$, $\overline{FIOP}$);
    $\overline{SP}$ of AJUNDO = $\bar{P}$ + $\bar{S}$ · $\overline{\Delta S}$ ;
    $\overline{SN}$ = UNRMAL ($\overline{SP}$, $\overline{SF}$);
    $\overline{SN}$ of AJUNDO = $\overline{SN}$ of CONTROL ( $\overline{\Delta S}'$, $\overline{SN}$, $\overline{FCON}$) ; (1)
    $\bar{Z}$ of AJUNDO = $\bar{Z}$ of CONTROL ( $\overline{\Delta S}'$, $\overline{SN}$, $\overline{FCON}$);
end
```

1) This procedure operates so as to obtain $\overline{SN}$ $\overline{\Delta S}$ < 0 {$\overline{SN}$ $\overline{\Delta S}$ > 0} if ($\overline{FCON}$) = 0 {($\overline{FCON}$) = 1}, as said, whatever the initially determined ($\overline{SN}$) orientation, therefore it is thus defined:

```
begin
    $\bar{Z}$ of CONTROL = 1
    if $\overline{FCON}$ = 0 then
    begin
        if $\overline{SN}$ × $\overline{\Delta S}$ > 0 then $\overline{SN}$ of CONTROL = $-\overline{SN}$ else
        $\overline{SN}$ of CONTROL = $\overline{SN}$;
    end
    if $\overline{SN}$ × $\overline{\Delta S}$ ≤ 0 then
    begin
        $\overline{SN}$ of CONTROL = $-\overline{SN}$ ;
        $\bar{Z}$ of CONTROL = -1 ;
    end
    $\overline{SN}$ of CONTROL = $\overline{SN}$ ;
end
```

17. The RADAR routine

Aim: to compute, given a point $(\bar{P})$ and a direction $(\overline{\Delta S})$, a new direction (vect), subordinately to a (FCOUNT) flag that can take 14 distinct values, one for each of 14 different computational processes.

Remarks: this routine is called in by the AJUNDO routine when the DDST routine fails to determine the surface type as one of those accepted (see paragraph 2 of this chapter). RADAR then issues new input-data to DDST as successive, up to 14, different iterations for the $\overline{\Delta S}$ direction, from $\overline{\text{FCOUNT}} = 1$ through $\overline{\text{FCOUNT}} = 14$; after which, if DDST still fails to accept, the AJUNDO routine will send an ERROR MESSAGE. The 14 distinct directions $\overline{\Delta S}$ iteratively computed by RADAR are associated one-to-one with 14 functions $f_1, f_2, \dots, f_{14}$ of the initial values for $(\bar{P})$ and $(\overline{\Delta S})$, as follows

1) the initial $\overline{\Delta S}$ direction is dealt with as a cone axis;
2) $f_1, \dots, f_8$ define eight distinct cone generatrix lines, therefore eight distinct cones, all of semi-aperture $\theta = 30°$;
3) $f_9, \dots, f_{12}$ define four more cone generatrix lines, i.e., four more cones, all of semi-aperture $\theta = 60°$;
4) f_{13}, and f_{14} , finally, define two directions orthogonal to each other and to $\overline{\Delta S}$.

Process: the routine is defined by the following procedure

```
vect proc RADAR (point P̄, vect ΔS̄, flag FCOUNT̄)
begin
    integer j̄ ;
    j̄ = 1 ;
1:  if FCOUNT̄ = j̄ then
    begin
        RADAR = f_j (P̄, ΔS̄);
        goto 2;
    end
    j̄ = j̄ + 1;
    goto 1;
2: end
```

18. The UICOMP routine

Aim: given the vector $(\overline{SN})$ applied in some given point $(\overline{P})$, to compute a vector $(\underline{vect})$ perpendicular to the "tool-axis" $(\overline{TA})$ subordinately to a given $(\overline{FCON})$ control for a suitable orientation of $(\overline{vect})$ with respect to the assigned $(\overline{SN})$; the "tool-end" position $(\overline{TE})$ also intervenes.

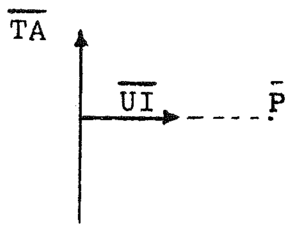
Remarks: this routine will be shown to be expedient when defining next, the AMIND routine. Now it can be said, very roughly, that the UICOMP vector takes care of that part of the tool-cutter which is involved in computing direct distances.

Process: the routine is defined by the following procedure

```
vect proc. UICOMP (vect $\overline{TA}$, point $\overline{TE}$, point $\overline{P}$, vect $\overline{SN}$
control $\overline{FCON}$)
begin
        vect $\overline{UI}$;
        if$[|(\overline{P}-\overline{TE})\times\overline{TA}|<10^{-5}SQRT(|\overline{P}-\overline{TE}|\uparrow 2-|(\overline{P}-\overline{TE})\times\overline{TA}|\uparrow 2)$ v
           $SQRT(|\overline{P}-\overline{TE}|\uparrow 2-|(\overline{P}-\overline{TE})\times\overline{TA}|\uparrow 2)<10^{-6}]$ then
           goto 1 else goto 2
1:         if $|$z-of $\overline{TA}|<10^{-6}$ then
           begin
                UICOMP = VECTOR (0,0,1);
```

```
              goto 3 ;
          end
          if (|x-of $\overline{TA}$| < $10^{-6}$ $\wedge$ |y-of $\overline{TA}$| < $10^{-6}$) then
          begin
              UICOMP = VECTOR (1,0,0);
              goto 3;
          end
2:        begin
              $\overline{UI}$ = NORM ($\overline{TA}$, $\overline{P}$) (1);
              UICOMP = CONTROL ($\overline{UI}$, $\overline{SN}$, $\overline{FCON}$);
          end
end
```

1) this procedure computes the vector perpendicular to $\overline{TA}$ and pointing towards $\overline{P}$.



19. The TLNORM routine

Aim: given the position and shape of the tool (by means of tool-axis TA and tool-end TE, and of tool-data TD); given a surface (SF) and a modifier (MOD) to monitor tool-to-surface position; given the center (CC) of a sphere approximating (SF) in the tool neighborhood and a vector (SN) normal to it - to compute a point (TP) of the tool and a unit-vector (TN) generally perpendicular to the tool in that point, subordinately to a given (FCON) control and to the (SN) orientation for defining the (TN) orientation.

Process: the routine is defined by the following procedure

```
dupla = (point TA, vect TN)

dupla proc TLNORM (vect  TA, point TE, tool-data TD,
                  surf SF,modifier MOD, point CC,
                  vect SN, control FCON ).

begin
    vect  TN, UI, HT; real WD ;
    if  MOD = ON then
    begin
        TP of TLNORM = TE ;

        TN = (TP - CC)/(TP - CC) ;
        TN of TLNORM = CONTROL (TN, SN, FCON);
        goto 1  ;
    end
    if IS-PLAN ($\overline{SF}$) $\wedge$ [($\overline{TA}$ x NORM-PLANE ($\overline{SF}$)] < $10^{-6}$ then
    goto 2 else goto 3
2:      $\overline{UI}$=UICOMP ($\overline{CC}$, $\overline{SN}$, $\overline{TA}$, $\overline{FCON}$ ) ;
        $\overline{HT}$ = HEIGHT ($\overline{TD}$);(1)
        $\overline{WD}$ = WIDTH ($\overline{TD}$); (2)
        $\overline{TP}$ of TLNORM = $\overline{TE}$ + $\overline{HT}$ + $\overline{WD}\cdot\overline{UI}$ ;
        $\overline{TN}$ of TLNORM = $\overline{UI}$ ;
        goto 1;
3:      $\overline{UI}$ = UICOMP ($\overline{CC}$, $\overline{SN}$, $\overline{TA}$, $\overline{FCON}$);
        $\overline{TP}$ of TLNORM = $\overline{TP}$ of SEVENSEQ ($\overline{TA}$,$\overline{TE}$,$\overline{TD}$,$\overline{UI}$,$\overline{CC}$,$\overline{FCON}$);
        $\overline{TN}$ of TLNORM = $\overline{TN}$ of SEVENSEQ ($\overline{TA}$,$\overline{TE}$,$\overline{TD}$,$\overline{UI}$,$\overline{CC}$,$\overline{FCON}$);
1:end
```

1) this procedure computes the tool height h (see fig.16 below), and then the vector h $\overline{TA}$

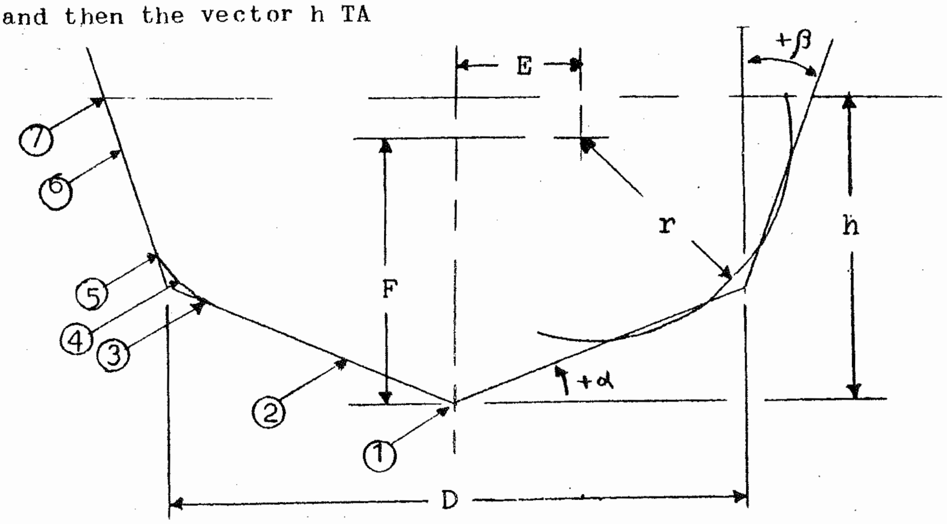
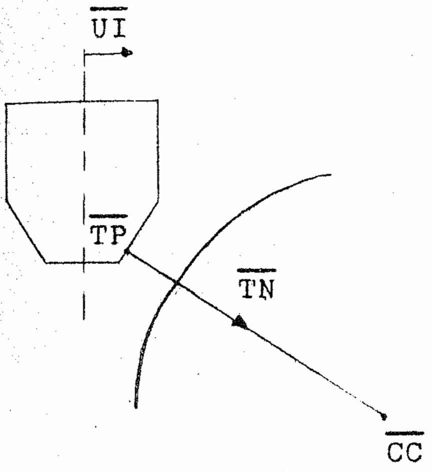


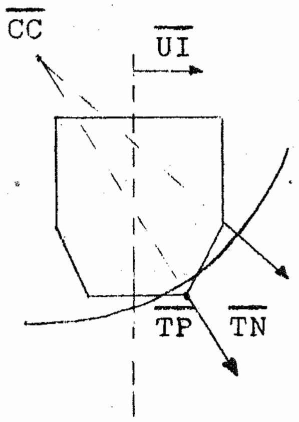
Fig.16

2) this computes the maximum tool half-width $\frac{D}{2} + \text{tg}\,\beta\,(h - \frac{D}{2}\,\text{tg}\,\alpha)$

3) this procedure examines one after another the seven

tool segments (see fig.16) from the UI vector side, in order to determine a segment the normal to which through the point CC can be defined as the normal to the tool. This procedure selects the dupla = (point TP,vect TN) for which TP is the closest to the surface, if the latter is convex or is concave with FCON set at 0, see fig.17; selects the first compatible "dupla", if the surface is concave with FCON set at 1, see fig.18.





The two figures below show two different cases of tool shape for computing the normal to the tool, in the convex (fig.19), and in the concave (fig.20) case:

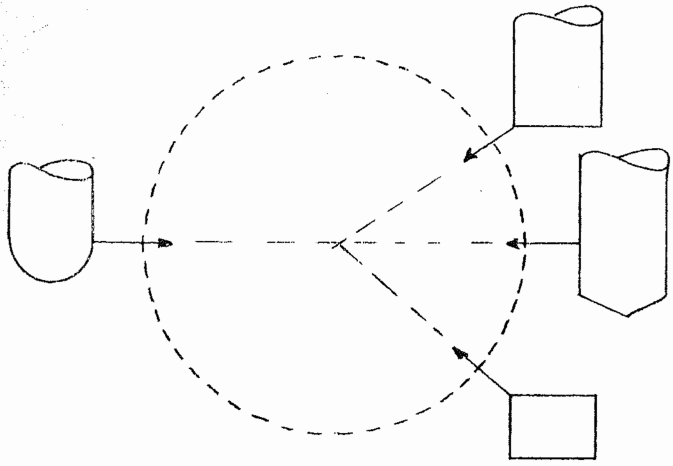


fig.19

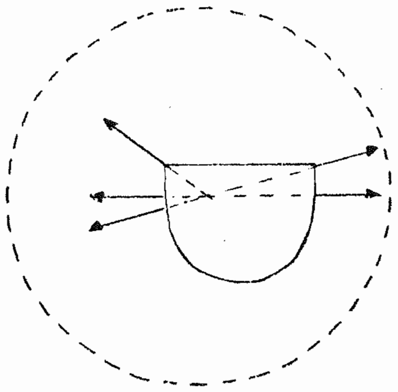


fig.20

20. The CCURV routine

Aim: given a surface $(\overline{SF})$, two points $(\overline{P}_1)$ and $(\overline{P}_2)$ of $(\overline{SF})$, and the normal $(\overline{SN})$ to $(\overline{SF})$ in $(\overline{P}_2)$, to compute the center $(\overline{PI})$ and radius $(\overline{S})$ of a sphere through $(\overline{P}_2)$ and normal to $(\overline{SN})$ in $(\overline{P}_2)$, that approximates $(\overline{SF})$ in the neighborhood of $(\overline{P}_2)$ extending to $(\overline{P}_1)$.

Process: the routine is defined by the following procedure

```
bis = (real $\overline{S}$, point $\overline{PI}$ )
bis proc  CCURV (surf $\overline{SF}$, point $\overline{P}_1$, point $\overline{P}_2$, norm $\overline{SN}$)
begin
    if  IS-PLAN ($\overline{SF}$) then
    begin
        $\overline{S}$ of CCURV = 4500;
        $\overline{PI}$ of CCURV = $\overline{P}_2 - 4500\cdot\overline{SN}$ ;
        goto  1;
    end
    if $|\overline{P}_2-\overline{P}_1| \leq 0,005$ then *as $\overline{S}$ and $\overline{PI}$ of CCURV keep
    previous value of $\overline{S}$ and of $\overline{PI}$* (1) else
    if $\overline{SN} \times (\overline{P}_2-\overline{P}_1) = 0$  then
    begin
        $\overline{S}$ of CCURV = 45;
        $\overline{PI}$  of CCURV = P2 - $45\cdot\overline{SN}$ ;
```

1) in the sense that those values for the curvature center and radius are retained for the surface that were stored in computer memory before calling CCURV.

```
        goto 1 ;
    end
    $\bar{\alpha}$ = UTANGLE [$\overline{SN}$, $(\bar{P}_2-\bar{P}_1)/|\bar{P}_2-\bar{P}_1|$];
    $\bar{S}$ of CCURV = $|\bar{P}_2-\bar{P}_1|/\ 2\cos(\bar{\alpha})$ ;
    $\overline{PI}$ of CCURV = $\bar{P}_2 - \overline{SN}$ ($\bar{S}$ of CCURV) ; (2)
1:end
```

2) The normal $(\overline{SN})$ to the surface $(\overline{SF})$ is outward oriented with respect to the solid whose boundary is $(\overline{SF})$, therefore $\bar{S}$ is negative if the solid surface is concave from $\overline{P_1}$ to $\overline{P_2}$, $(\alpha>90°)$, and is positive if such surface is convex from $\overline{P_1}$ to $\overline{P_2}$, $(\alpha<90°)$, see fig. 21 below.

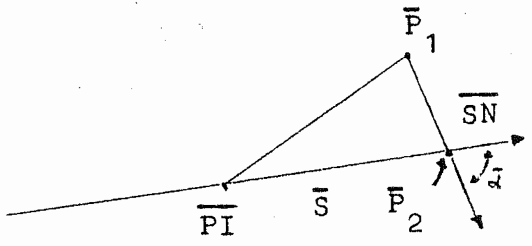


fig.21

21. The AMIND routine

Aim: to compute a normal distance[1] ($\overline{S}$), with sign, between a surface ($\overline{SF}$) and the tool-cutter (tool-axis $\overline{TA}$, tool-end $\overline{TE}$, tool-data $\overline{TD}$) subordinately to a ($\overline{FIOP}$) flag, being given the modifier ($\overline{MOD}$) for the tool-to-surface relationship, and a ($\overline{FCON}$) control for vector orientation. There are also being computed a pointer ($\overline{Z}$) and four more output parameters ($\overline{TP}$, $\overline{TN}$, $\overline{SP}$, $\overline{SN}$), that together with $\overline{S}$ completely determine the problem.

1) The ensuing computation of the "curve cutter offsets" implies the notion of "normal distance" between two surfaces; namely, the distance between a point P of the first, and a point Q of the second surface, if the line through P and Q is normal to both surfaces. Generally, a normal distance is not necessarily an absolute minimum, but a local minimum or maximum, in various concave and/or convex cases. By "cutter" it is meant the "cutter surface".

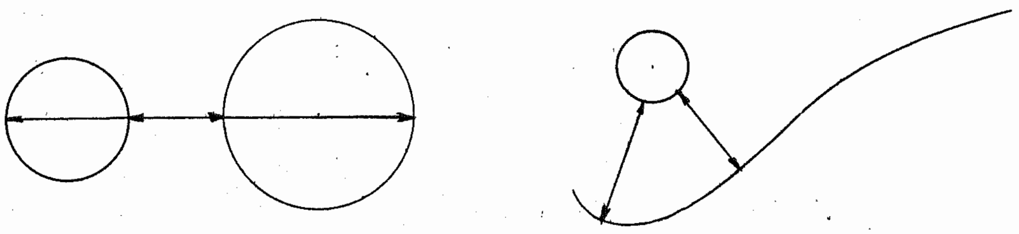


Fig. 22

Remarks: the normal distance S is computed between the point $\overline{SP}$ of $\overline{SF}$ and the point $\overline{TP}$ of the cutter, while $\overline{SN}$ and $\overline{TN}$ are unit vectors respectively normal to $\overline{SF}$ and to the cutter, and applied in $\overline{SP}$ and $\overline{TP}$; therefore they lay on the same straight line normal to both surfaces through the two points.

1) When $\overline{SF}$ is a "drive-" or a "part-surface", $\overline{S}$ is the least normal distance if the modifier is TO or TANTO, it is instead the greatest one if the modifier is PAST; in both cases $\overline{TN}$ and $\overline{SN}$ both point outward (with respect to the surface each is normal to), therefore they are opposite, see fig.s 23 and 25 below:

$\overline{MOD}$ = TO

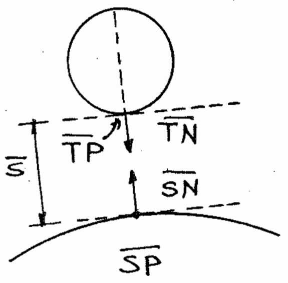


fig.23

$\overline{MOD}$ = ON

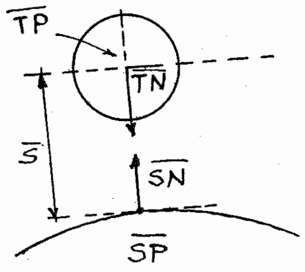


fig.24

$\overline{MOD}$ = PAST

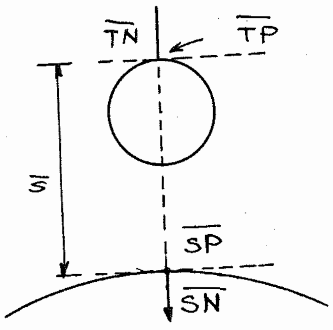


fig.25

The case of $\overline{MOD}$ = ON is similar to that of $\overline{MOD}$ = TO, only now it is $\overline{TP}$ = $\overline{TE}$, see fig.24; $\overline{S}$ is positive if the path from $\overline{TP}$ towards $\overline{SP}$ is equiverse with $\overline{TN}$, otherwise it is negative.

2) When $\overline{SF}$ is a "check-surface", both $\overline{SN}$ and $\overline{TN}$ point approximately in the direction of motion if the modifier is TO, TANTO, or ON, see fig.s 26 and 28; both point instead approximately in the opposite direction if the modifier is PAST, see fig.27; the sign for $\overline{S}$ is as previously stated.

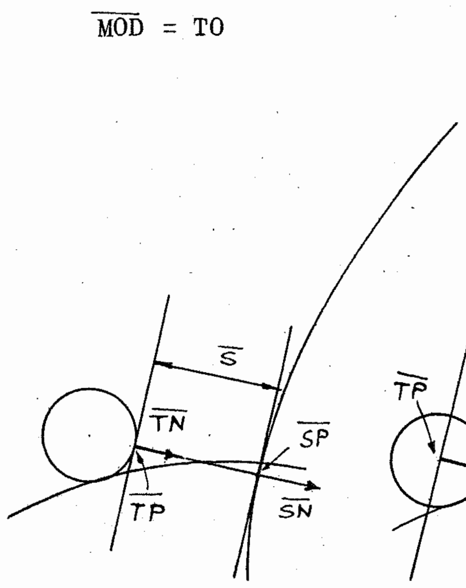


fig.26

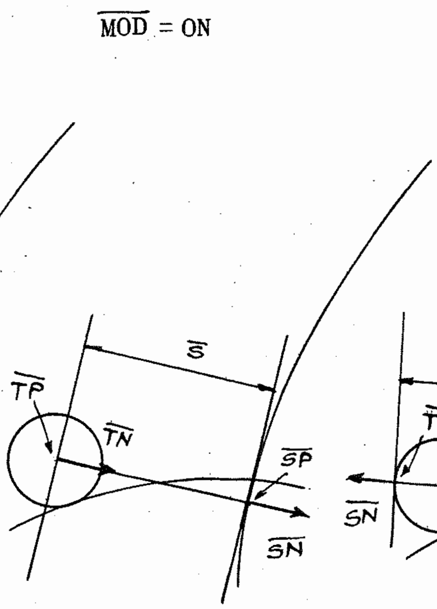


fig.27

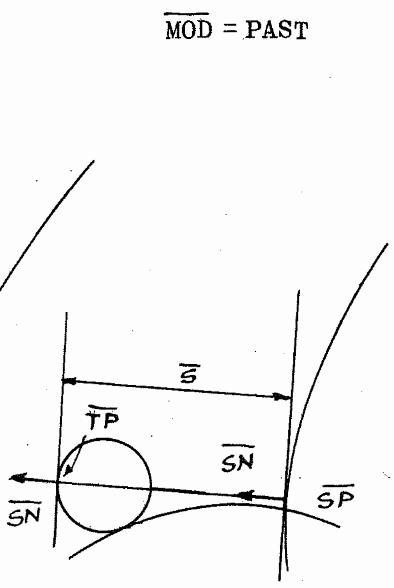


fig.28

Process: the routine is defined by the following procedure

```
sexamind = (real $\overline{S}$, point $\overline{TP}$, point $\overline{SP}$, vect $\overline{TN}$, vect $\overline{SN}$,
            indicator $\overline{Z}$)
sexamind  proc AMIND (vect $\overline{TA}$, point $\overline{TE}$, tool-data $\overline{TD}$,
                      surf $\overline{SF}$, modifier $\overline{MOD}$, control $\overline{FCON}$,
                      flag $\overline{FIOP}$)
begin
     point $\overline{TP}_o$ ; vect $\overline{UI}, \overline{SN}, \overline{TN}_o, \overline{TN}_1, \overline{SN}_o, \overline{SN}_1$; ajundo $\overline{ABCD}$,
     $\overline{EFGH}$, $\overline{LMNO}$;
     dupla $\overline{CD}$ ; bis $\overline{AB}$;

     $\overline{TP}_o$ = $\overline{TE}$ + HEIGHT ($\overline{TD}$);
     $\overline{TN}_o$ = VECTOR (1,0,0);
     $\overline{ABCD}$ = AJUNDO ($\overline{TP}_o$, $\overline{TN}_o$, $\overline{SF}$, $\overline{FIOP}$, $\overline{FCON}$);
     if $\overline{FCON}$ = 0 then
     begin
          $\overline{SN}_o$ = POINT-CUT ($\overline{TA}$,$\overline{TE}$,$\overline{TD}$,$\overline{SP}$ of $\overline{ABCD}$,$\overline{SN}$ of $\overline{ABCD}$);(1)
          $\overline{TN}_1$ = - $\overline{SN}_o$ ;
          goto 3 ;
     end
     $\overline{TN}_1$ = $\overline{SN}$ of $\overline{ABCD}$ ;
3:   $\overline{EFGH}$ = AJUNDO ($\overline{TP}_o$, $\overline{TN}_1$, $\overline{SF}$, $\overline{FIOP}$, $\overline{FCON}$);
     if $\overline{FCON}$ = 0 then
     begin
```

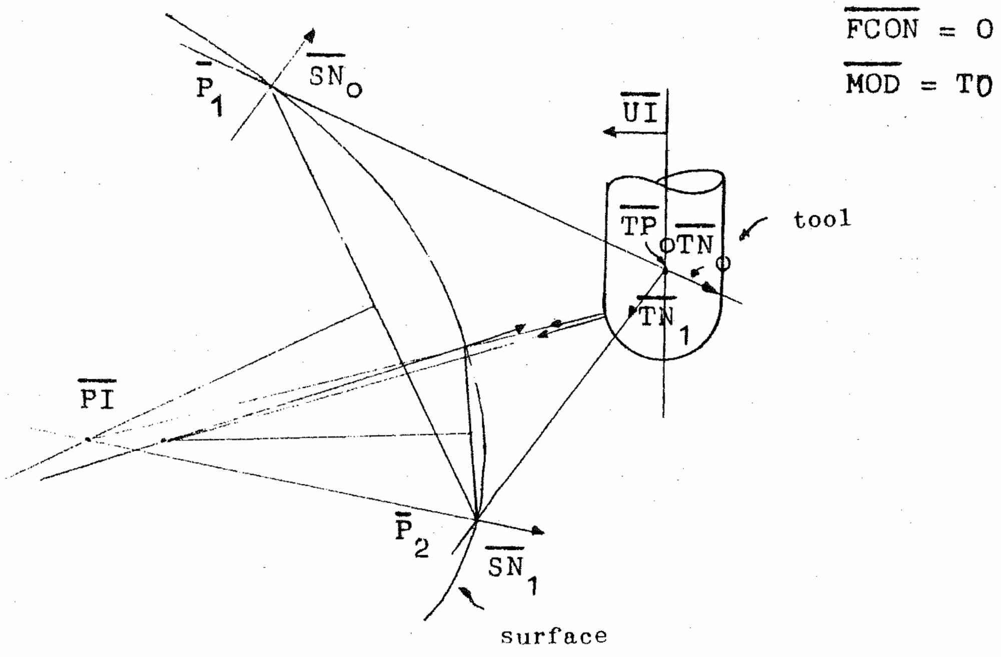
1) this procedure orients the applied vector $\overline{SN}$ in $\overline{SP}$ so that it points towards the cutter.

```
        if $\overline{MOD}$ = PAST then
        begin
               $\overline{SN}_1$= -POINT-CUT ($\overline{TA}$,$\overline{TE}$,$\overline{TD}$,$\overline{SP}$ of $\overline{EFGH}$,$\overline{SN}$ of $\overline{EFGH}$);
               goto  4;
        end
        begin
               $\overline{SN}_1$ = POINT-CUT ($\overline{TA}$,$\overline{TE}$,$\overline{TD}$,$\overline{SP}$ of $\overline{EFGH}$,$\overline{SN}$ of $\overline{EFGH}$);
               goto 4 ;
        end
end
begin
        if $\overline{MOD}$ = PAST then
        begin
               $\overline{SN}_1$ = POINT-CUT ($\overline{TA}$,$\overline{TE}$,$\overline{TD}$,$\overline{SP}$ of $\overline{EFGH}$,$\overline{SN}$ of $\overline{EFGH}$);
               goto 4;
        end
        begin
               $\overline{SN}_1$ = -POINT-CUT ($\overline{TA}$,$\overline{TE}$,$\overline{TD}$,$\overline{SP}$ of $\overline{EFGH}$,$\overline{SN}$ of $\overline{EFGH}$);
               goto  4;
        end
end
```

```
4:  $\bar{P}_1 = \overline{SP}$ of $\overline{ABCD}$;
    $\bar{P}_2 = \overline{SP}$ of $\overline{EFGH}$;
    $\overline{SN} = \overline{SN}_1$ ;
2:  $\overline{AB}$ = CCURV ($\overline{SF}$, $\bar{P}_1$, $\bar{P}_2$, $\overline{SN}$ ) ;
    $\overline{UI}$ = UICOMP ($\overline{TA}$,$\overline{TE}$,$\overline{PI}$ of $\overline{AB}$, $\overline{SN}$, $\overline{FCON}$);
    $\overline{CD}$ = SEVENSEQ ($\overline{TA}$,$\overline{TE}$,$\overline{TD}$,$\overline{UI}$,$\overline{PI}$ of $\overline{AB}$, $\overline{FCON}$);
    $\overline{LMNO}$ = AJUNDO ($\overline{TP}$ of $\overline{CD}$,$\overline{TN}$ of $\overline{CD}$,$\overline{SF}$,$\overline{FIOP}$,$\overline{FCON}$);
    if $|(\overline{SN}$ of $\overline{LMNO})\times(\overline{TN}$ of $\overline{CD})| > 0{,}9995$ then goto 1 else
    $\bar{P}_1 = \bar{P}_2$;
    $\bar{P}_2 = \overline{SP}$ of $\overline{LMNO}$ ;
    $\overline{SN} = \overline{SN}$ of $\overline{LMNO}$ ;
    goto 2 ;
1:  $\bar{Z}$ of AMIND = $\bar{Z}$ of $\overline{LMNO}$ ;
    $\bar{S}$ of AMIND = $\bar{S}$ of $\overline{LMNO}$ ;
    $\overline{SP}$ of AMIND = $\overline{SP}$ of $\overline{LMNO}$ ;
    $\overline{SN}$ of AMIND = $\overline{SN}$ of $\overline{LMNO}$ ;
    $\overline{TP}$ of AMIND = $\overline{TP}$ of $\overline{CD}$;
    $\overline{TN}$ of AMIND = $\overline{TN}$ of $\overline{CD}$;
end
```

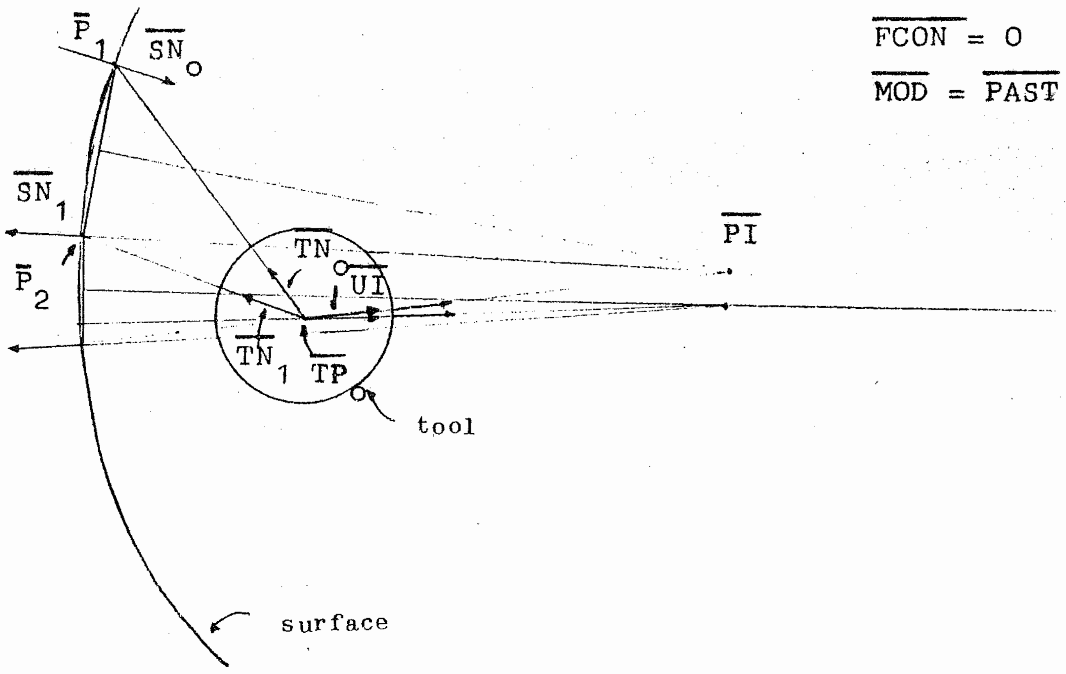
Remarks: 1) this routine solves with elegance the three-dimensional problem of the normal distance between two surfaces by means of an iterative bi-dimensional process that begins with the statement of address 4: the current planes involved are that defined by the points $\overline{P_1}$, $\overline{P_2}$ and the vector $\overline{SN}$, and that defined by the curvature center $\overline{PI}$ and the "tool-axis" $\overline{TA}$, where each of the three points in question varies in space from one step to the next in the iteration, which ends when the normal vectors, one to each surface, are almost parallel. This iteration is rapidly convergent, as practically four or five steps suffice; but there are cases, such as that of the double curvature (parabolic hyperboloid), when this algorithm fails to be convergent as it stands, and one must follow more complex routes to deal successfully with such cases.

The following four figures are an illustration of this routine.



$\overline{FCON} = 0$
$\overline{MOD} = TO$
$\overline{P}_1$
$\overline{SN}_o$
$\overline{UI}$
tool
$\overline{TP}$
$\overline{TN}$
$\overline{TN}_1$
$\overline{PI}$
$\overline{P}_2$
$\overline{SN}_1$
surface

fig.29



$\overline{P}_1$
$\overline{SN}_o$
$\overline{FCON} = 0$
$\overline{MOD} = \overline{PAST}$
$\overline{SN}_1$
$\overline{PI}$
$\overline{TN}$
$\overline{UI}$
$\overline{P}_2$
$\overline{TN}_1$
$\overline{TP}$
tool
surface

fig.30

$\overline{FCON} = 1$
$\overline{MOD} = TO$

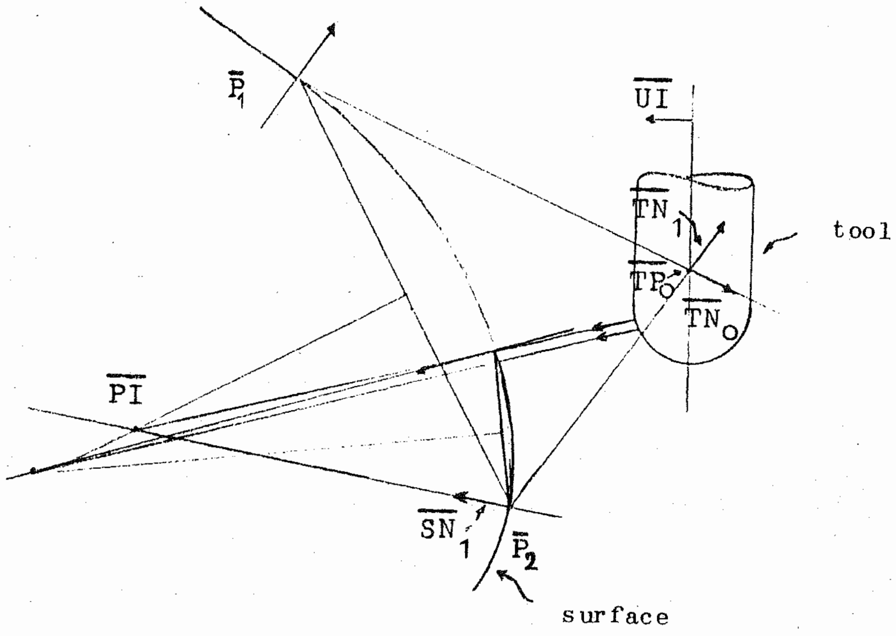


fig.31

$\overline{FCON} = 1$
$\overline{MOD} = PAST$

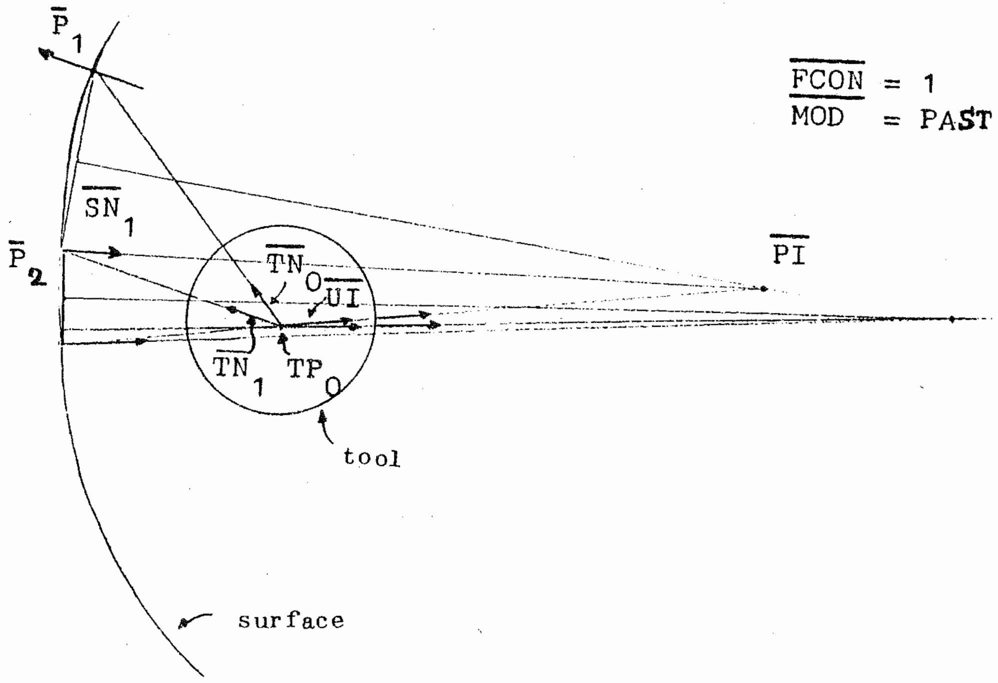


fig.32

22. The GAUSS routine

Aim: Being given the system $\{\Sigma_j \overline{V}_{ij} \overline{X}_j = \overline{D}_i\}$; $i=1,2,3$, $j=1,2,3$ to compute the vector (vect) whose components are solutions of the system.

Remarks: This routine, namely

vect proc GAUSS (real $\overline{V}_{11}, \ldots,$ real $\overline{D}_3$), takes its name from the method followed in solving the given system; therefore the Process shall be omitted. If there is no solution, or more than one solution, the routine sends an ERROR message.

23. The CENTR routine

Aim: to position the tool-cutter (tool-axis $\overline{TA}$, tool-end $\overline{TE}$, tool-data $\overline{TD}$) into a final location (tool-pos) within the tolerance limits of each of the three control surfaces ($\overline{DRIVE}$, $\overline{PART}$, $\overline{CHECK}$).

Remarks: the positioning is realized by iteration; the process begins by calling the AMIND routine three times to compute the normal distances $\overline{S}_{DS}, \overline{S}_{PS}, \overline{S}_{CS}$ and the normal vectors $\overline{TN}_{DS}, \overline{TN}_{PS}, \overline{TN}_{CS}$, with obvious meaning for the subscripts; then the GAUSS routine computes the vector $\overline{CMOVE} = \overline{X}$ by solving the system

$$\begin{cases} \overline{TN}_{DS} \times \overline{X} = \overline{S}_{DS} \\ \overline{TN}_{PS} * \overline{X} = \overline{S}_{PS} \\ \overline{TN}_{CS} \times \overline{X} = \overline{S}_{CS} \end{cases}$$

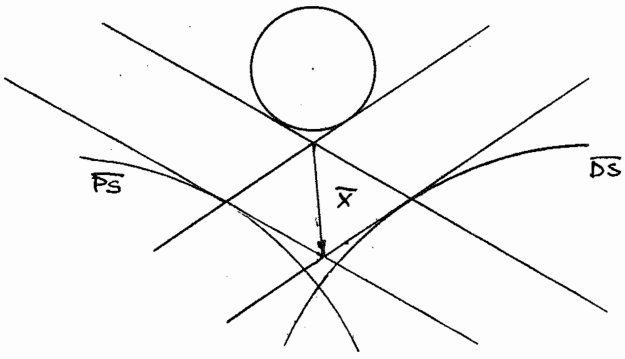


fig.35

The tool-cutter is then being displaced of the quantity $\overline{X}$ thus computed, and the process continues iteratively by calling AMIND and GAUSS again, until distances for all the surfaces concerned are within their corresponding tolerance limits.

Process: the routine is defined by the following procedure

```
tool-pos = (vect $\overline{TA}$, point $\overline{TE}$, point $\overline{TP}_{DS}$, vect $\overline{TN}_{DS}$,
            point $\overline{TP}_{PS}$, vect $\overline{TN}_{PS}$).
tool-pos proc CENTR (surface-data $\overline{DRIVE}$,
                     surface-data $\overline{PART}$,
                     surface-data $\overline{CHECK}$,
                     vect $\overline{TA}$, point $\overline{TE}$, tool-data $\overline{TD}$)
begin
    surf $\overline{DS}$,$\overline{PS}$,$\overline{CS}$ ;modifier $\overline{MODDS}$,$\overline{MODPS}$,$\overline{MODCS}$;flag
    $\overline{FIOPDS}$,$\overline{FIOPPS}$, $\overline{FIOPCS}$;, control $\overline{FCONCS}$; tolerance
    $\overline{\tau}_{DS}$, $\overline{\tau}_{PS}$, $\overline{\tau}_{CS}$; switch $\overline{FTAN}$; point $\overline{TEK}$ , $\overline{TP}_{DS}$,
    $\overline{TP}_{PS}$; sexamind $\overline{DISCS}$, $\overline{DISDS}$, $\overline{DISPS}$; real $\overline{S}_{CS}$,
    $\overline{S}_{DS}$, $\overline{S}_{PS}$; vect $\overline{TN}_{CS}$, $\overline{TN}_{DS}$, $\overline{TN}_{PS}$, $\overline{CMOVE}$; psc $\overline{ABC}$;
    $\overline{DS}$ = $\overline{SF}$ of $\overline{DRIVE}$ ;
    $\overline{PS}$ = $\overline{SF}$ of $\overline{PART}$;
    $\overline{CS}$ = $\overline{SF}$ of $\overline{CHECK}$ ;
    $\overline{MODDS}$ = $\overline{MOD}$ of $\overline{DRIVE}$;
```

```
        $\overline{MODPS} = \overline{MOD}$ of $\overline{PART}$;
        $\overline{MODCS} = \overline{MOD}$ of $\overline{CHECK}$;
        $\overline{FIOPDS} = \overline{FIOP}$ of $\overline{DRIVE}$;
        $\overline{FIOPPS} = \overline{FIOP}$ of $\overline{PART}$ ;
        $\overline{FIOPCS} = \overline{FIOP}$ of $\overline{CHECK}$;
        $\overline{FCONCS} = \overline{FCON}$ of $\overline{CHECK}$;
        $\bar{\tau}_{DS} = \bar{\tau}$ of $\overline{DRIVE}$;
        $\bar{\tau}_{PS} = \bar{\tau}$ of $\overline{PART}$ ;
        $\bar{\tau}_{CS} = \bar{\tau}$ of $\overline{CHECK}$;
A:      if a tangent or pseudo-tangent case is involved (1)
        then
        begin
                $\overline{FTAN} = 1$;
                goto S ;
        end
        begin
                $\overline{FTAN} = 0$;
                goto B ;
```

1) i.e., when compatible tool positioning is in the contact area between the "drive" and the "check" surface tolerance shells.

```
     end
B:   $\overline{TEK} = \overline{TE}$ ;
     $\overline{DISCS} = \overline{AMIND}\ (\overline{TA}, \overline{TEK}, \overline{TD}, \overline{CS}, \overline{MODCS}, \overline{FCONCS}, \overline{FIOPCS})$;
     $\bar{\bar{S}}_{CS} = \bar{\bar{S}}$ of $\overline{DISCS}$;
     $\overline{TN}_{CS} = \overline{TN}$ of $\overline{DISCS}$;
C:   $\overline{DISDS} = AMIND\ (\overline{TA}, \overline{TEK}, \overline{TD}, \overline{DS}, \overline{MODDS}, 0, \overline{FIOPDS})$;
     $\bar{\bar{S}}_{DS} = \bar{\bar{S}}$ of $\overline{DISDS}$;
     $\overline{TN}_{DS} = \overline{TN}$ of $\overline{DISDS}$;
     $\overline{TP}_{PS} = \overline{TP}$ of $\overline{DISDS}$;
D:   $\overline{DISPS} = AMIND\ (\overline{TA}, \overline{TEK}, \overline{TD}, \overline{PS}, \overline{MODPS}, 0, \overline{FIOPPS})$;
     $\bar{\bar{S}}_{PS} = \bar{\bar{S}}$ of $\overline{DISPS}$;
     $\overline{TN}_{PS} = \overline{TN}$ of $\overline{DISPS}$;
     $\overline{TP}_{PS} = \overline{TP}$ of $\overline{DISPS}$;
E:   if $|\bar{\bar{S}}_{CS}| < 1/20\ \bar{\tau}_{CS}$
     then goto F else goto L
F:   if $|\bar{\bar{S}}_{DS}| < 1/20\ \bar{\bar{\tau}}_{DS}$
     then goto G else goto L
G:   if $|\bar{\bar{S}}_{PS}| < 1/20\ \bar{\tau}_{PS}$
     then goto 6 else goto L
L:   $\overline{CMOVE}$ = GAUSS (x-of $\overline{TN}_{DS}$, y-of $\overline{TN}_{DS}$, z-of $\overline{TN}_{DS}$, $\bar{\bar{S}}_{DS}$
                        (x-of $\overline{TN}_{PS}$, y-of $\overline{TN}_{PS}$, z-of $\overline{TN}_{PS}$; $\bar{\bar{S}}_{PS}$
                        (x-of $\overline{TN}_{CS}$, y-of $\overline{TN}_{CS}$, z-of $\overline{TN}_{CS}$, $\bar{\bar{S}}_{CS}$);
     $\overline{TE} = \overline{TEK} + \overline{CMOVE}$ ;
```

```
P:    if MODCS = ON then goto 9 else
      if (TEK - TE) intersect CS then
      begin
            MODCS = REVERSE (MODCS); (2)
            goto 9;
      end
9:    if MODDS = ON then goto 10 else
      if (TEK - TE) intersect DS then
      begin
            MODDS = REVERSE (MODDS);
            goto 10;
      end
10:   if  MODPS = ON then goto Q else
      if (TEK-TE) intersect PS then
      begin
            MODPS = REVERSE (MODPS);
            goto  Q;
      end
Q:    if FTAN = 0 then goto B
S:    ABC = CPLAN (CS, DS, PS, TE);
      $TN_{CS}$ = TN of ABC ;
```

2) this procedure inverts the pair TO-PAST, i.e., replaces the modifier TO with the modifier PAST if TO is on, or PAST with TO if PAST is on, while leaving fixed the other modifiers.

```
        $\overline{S}_{CS} = \overline{S}$ of $\overline{ABC}$;
        $\overline{MODCS}$ = ON;
        $\overline{CS} = \overline{PSC}$ of $\overline{ABC}$;
        goto  B;
6:      $\overline{TE}$ of CENTR = $\overline{TEK}$ ;
        $\overline{TA}$ of CENTR = $\overline{TA}$ ;
        $\overline{TN}_{DS}$ of CENTR=$\overline{TN}_{DS}$;
        $\overline{TN}_{PS}$ of CENTR=$\overline{TN}_{PS}$;
        $\overline{TP}_{DS}$ of CENTR=$\overline{TP}_{DS}$;
        $\overline{TP}_{PS}$ of CENTR=$\overline{TP}_{PS}$;
end
```

The following flow-chart presents visually the process

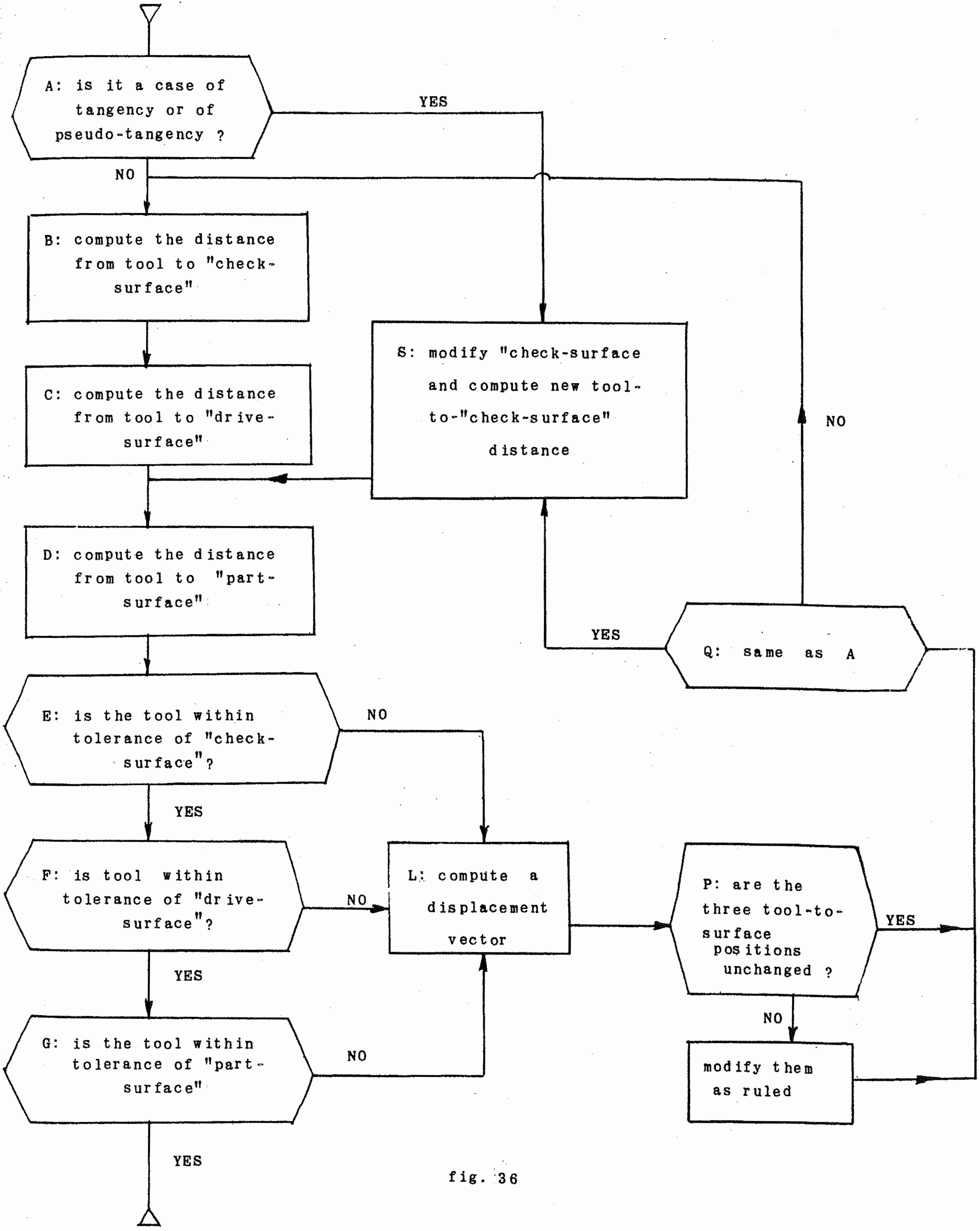

A: is it a case of tangency or of pseudo-tangency ?
YES
NO
B: compute the distance from tool to "check-surface"
C: compute the distance from tool to "drive-surface"
S: modify "check-surface and compute new tool-to-"check-surface" distance
NO
D: compute the distance from tool to "part-surface"
YES
Q: same as A
E: is the tool within tolerance of "check-surface"?
NO
YES
F: is tool within tolerance of "drive-surface"?
NO
L: compute a displacement vector
P: are the three tool-to-surface positions unchanged ?
YES
YES
G: is the tool within tolerance of "part-surface"
NO
NO
modify them as ruled
YES

fig. 36

24. CPLAN routine

Aim: in a case of tangency or of pseudo-tangency for two surfaces ($\overline{CS}$,$\overline{DS}$), to compute a plane surface ("pseudo-check" $\overline{PSC}$), the normal vector ($\overline{TN}$) to such a plane, and the non-negative distance S from the "tool-end" ($\overline{TE}$) to such a plane.

Process: the routine is defined by the following procedure

```
psc = (plane $\overline{PSC}$, norm $\overline{TN}$, real $\overline{S}$)
psc proc. CPLAN (surf $\overline{CS}$, surf $\overline{DS}$, surf $\overline{PS}$, point $\overline{TE}$)
begin
    point $\overline{P}_2$, $\overline{P}_{1CS}$, $\overline{P}_{1DS}$, $\overline{CC}_{CS}$, $\overline{CC}_{DS}$; norm $\overline{SN}_{CS}$, $\overline{SN}_{DS}$, $\overline{SN}$;
    real $\overline{S}$;
    $\overline{P}_2$ = PSEUDO-TG.($\overline{CS}$,$\overline{DS}$); (1)
    $\overline{SN}_{CS}$ = UNRMAL ($\overline{P}_2$,$\overline{CS}$);
    $\overline{SN}_{DS}$ = UNRMAL ($\overline{P}_2$, $\overline{DS}$);
    $\overline{P}_{1CS}$ = SURF-POINT($\overline{CS}$, $\overline{P}_2$); (2)
    $\overline{P}_{1DS}$ = SURF-POINT ($\overline{DS}$, $\overline{P}_2$);
```

1) This point procedure computes the contact point of the two surfaces "check" and "drive" or of their tolerance shells.

2) This point procedure computes a point of $\overline{SF}$ that belongs to a sphere with center in P and radius of one tenth of an inch.

```
    $\overline{CC}_{CS}$ = $\overline{PI}$ of CCURV ($\overline{CS}$, $\bar{P}_{1CS}$, $\bar{P}_2$, $\overline{SN}_{CS}$);
    $\overline{CC}_{DS}$ = $\overline{PI}$ of CCURV ($\overline{DS}$, $\bar{P}_{1DS}$, $\bar{P}_2$, $\overline{SN}_{DS}$);
    if a pseudo-tangent case is involved (3) then
    goto 1 else goto 2
1:  if [IS-PLAN($\overline{CS}$) v IS-PLAN($\overline{DS}$)] then goto 4
    $\overline{PSC}$ of CPLAN=BIVECT-PLAN ($\overline{CC}_{CS}$-$\overline{CC}_{DS}$, $\overline{SN}_{CS} \wedge \overline{SN}_{DS}$);(4)
```

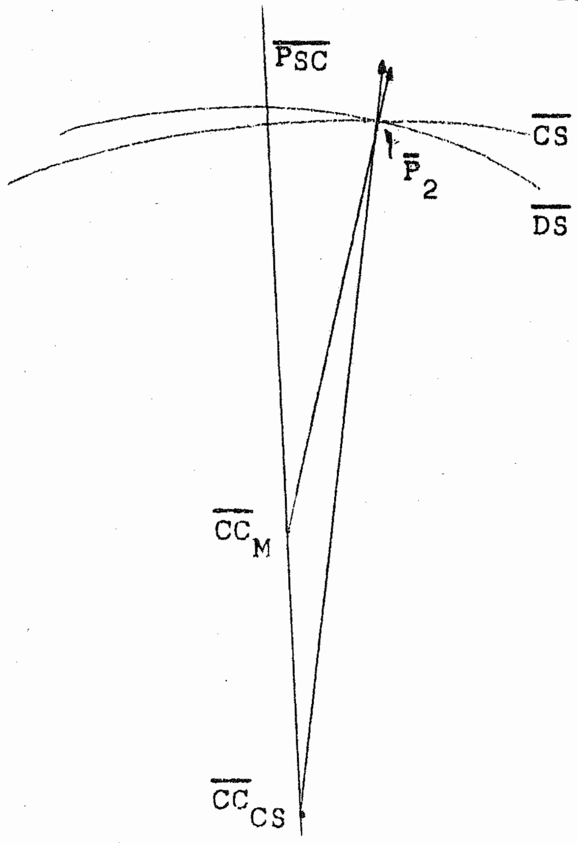


fig. 41.

3) As seen, the pseudo-tangency case is dealt with separately from that of tangency.

4) This procedure computes the plane defined by two (co-planar) vectors.

```
    goto 5;
  4: if IS-PLAN ($\overline{DS}$) then
    begin
          $\overline{PSC}$ of CPLAN = PERPLAN ($\overline{CC}_{CS}$, $\overline{SN}_{CS} \wedge \overline{SN}_{DS}$, $\overline{DS}$); (5)
          goto 5;
    end
          $\overline{PSC}$ of CPLAN = PERPLAN ($\overline{CC}_{DS}$, $\overline{SN}_{CS} \wedge \overline{SN}_{DS}$, $\overline{CS}$);
```

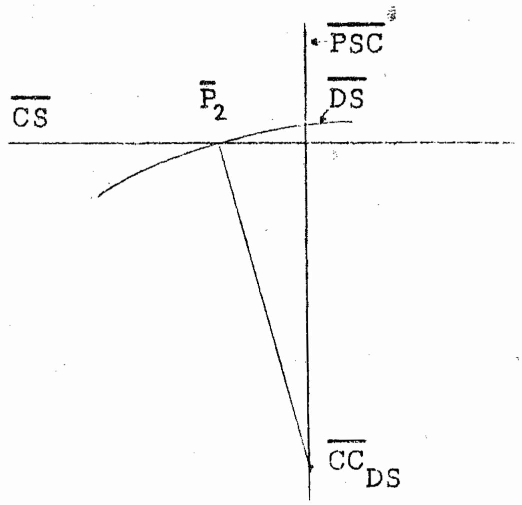


fig. 42

```
    goto 5:
 2: $\overline{PSC}$ of CPLAN = PS-PERPLAN ($\overline{CC}_{CS}$, $\overline{CC}_{DS}$, $\overline{PS}$); (6)
 5: $\overline{SN}$ = NORM-PLANE ($\overline{PSC}$);
    $\overline{S}$ = DDPLAN ($\overline{TE}$, $\overline{SN}$, $\overline{PSC}$);
    if S $\geqslant$ 0 then
```

5) This procedure computes the plane that contains a given point and a vector whose direction contains that point, and is normal to a given surface.

6) This procedure computes the plane through two points and normal to a given surface.

```
        begin
            $\bar{S}$ of CPLAN = $\bar{S}$ ;
            $\overline{TN}$ of CPLAN = $\overline{SN}$ ;
        end
        $\bar{S}$ of CPLAN = - $\bar{S}$;
        $\overline{TN}$ of CPLAN = -$\overline{SN}$;
    end
```

Remarks: 1) the two fig.s 39 and 40 illustrate the CPLAN routine computations

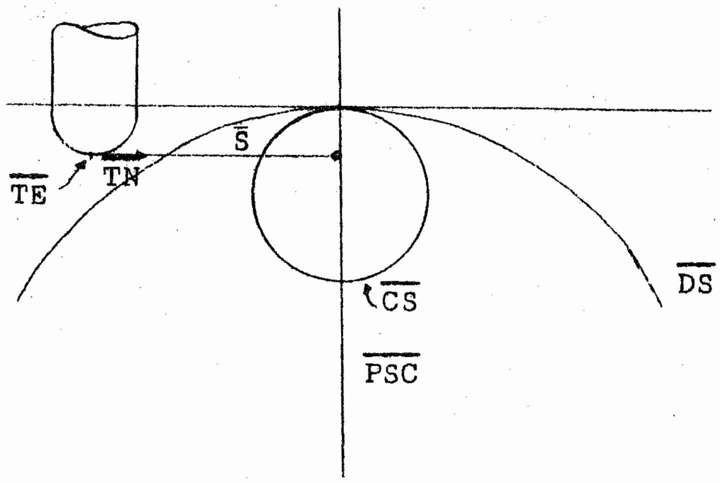


fig.39

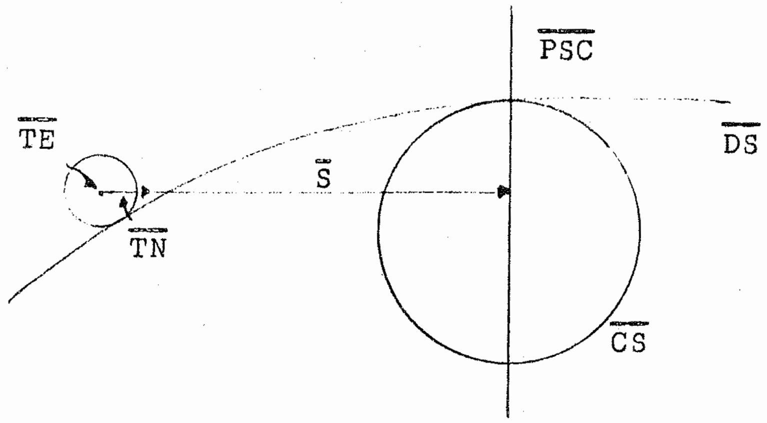


fig.40

2) The CPLAN routine has been added to prevent many possible causes of error for the CENTR routine. In cases of tangency and pseudo-tangency, infact, the planes approximating the C- and D-control surfaces tend to coincide, leading to indeterminacy conditions or non existent solutions. For instance, let us consider the case of a plane "check-surface" and a cylindrical "drive-surface", with a tangency line along which the tool is to be positioned, see fig.41 below.

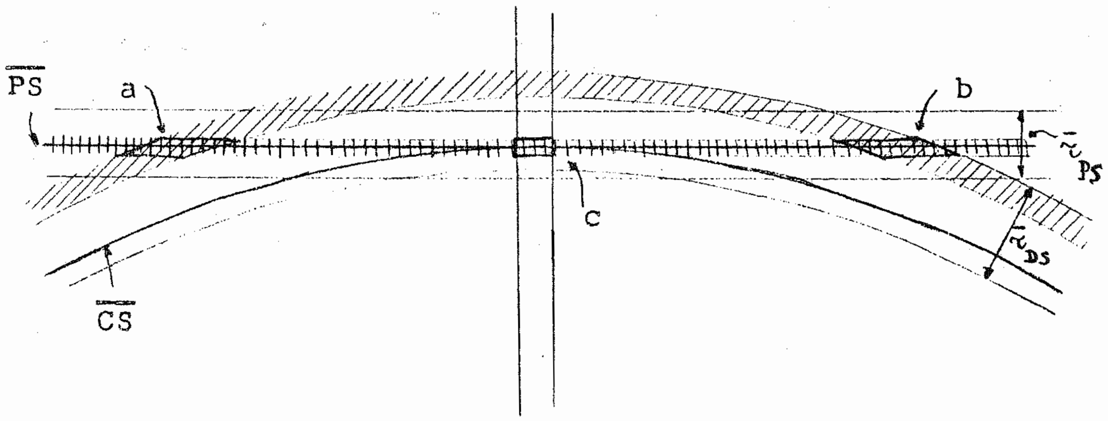


fig.41

The heavy lines are for the sections of the "check-" and the "drive-surface", the hatched stripe is for the region where the tool-cutter is to be found in order to be accepted by the CENTR routine; as previously said, the cutter is to be positioned a) with respect to a curve, within one tenth of the tolerance shell bandwidth by the least curvature side, and b) with respect to a plane within one twentieth of the tolerance shell bandwidth, by either side of the plane. Such conventions lead, in the case in question, to an indeterminacy error by the CENTR routine, for the tool may end up positioned into either of the two regions marked in fig.41 respectively "a" and "b", both far away from the tangency point.

The CPLAN routine solves the problem by introducing a "pseudo check-surface" as a plane that, generally, contains the curvature centers of the C- and D-surfaces, and is normal to both in their tangency or pseudo-tangency point. As can be seen, the solution is determinate in the region marked "c" in fig.41.

One solution of the problem, alternative to that offered by the CPLAN routine, might be that of extending the tolerance limits admissible for the CENTR routine (1/10 of total tolerance), as much as necessary for the two regions represented in fig.42 as shaded stripes to intersect in the tangency point.

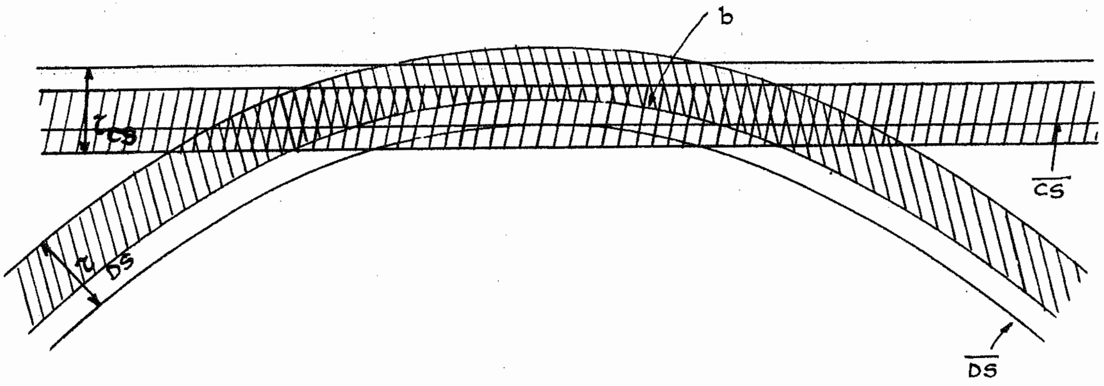


fig. 42

This solution implies shortened "cut-vectors", therefore longer computations, which inconvenience, though, concerns only the final cut-vectors. Furthermore, in order to discard acceptable, but unsuited solutions - as those possible in the periphery of the region marked "b" in fig. 42, one might add in the CENTR routine some supplementary conditions for accepting a final tool-cutter position; for instance, one might require from the cutter normal vectors (each to each of the two C- and D-surfaces) to be parallel to each other.

25. The CENTR2

Aim: to position the tool-cutter (tool-axis $\overline{\text{TA}}$, tool-end $\overline{\text{TE}}$, tool-data $\overline{\text{TD}}$) into a final location (tool-pos) within the tolerance limits of the two control surfaces ($\overline{\text{DRIVE}}$, $\overline{\text{PART}}$).

Remarks: this positioning is realized by iteration, as with the CENTR routine, to which CENTR2 is analogous, though it is not a routine of APT Section 2. Infact, it defines an imaginary "check-surface" as a plane suitably placed with respect to the tool.

Process: the routine is defined by the following procedure

```
tool-pos' = (vect $\overline{TA}$, point $\overline{TE}$, point $\overline{TP}_{DS}$, vect $\overline{TN}_{DS}$,
point $\overline{TP}_{PS}$, vect, $\overline{TN}_{PS}$).
tool-pos proc CENTR2 (surface-data $\overline{DRIVE}$,
                      surface-data $\overline{PART}$,
                      vect $\overline{TA}$, point $\overline{TE}$, tool-data $\overline{TD}$)
begin
      surf $\overline{DS}$, $\overline{PS}$; modifier $\overline{MODDS}$, $\overline{MODPS}$, flag $\overline{FIOPDS}$, $\overline{FIOPPS}$;
      tolerance $\tilde{\tau}_{DS}$, $\tilde{\tau}_{PS}$; point $\overline{TEK}$, $\overline{TP}_{DS}$, $\overline{TP}_{PS}$;
      sexamind $\overline{DISDS}$, $\overline{DISPS}$; real $\bar{s}_{DS}$, $\bar{s}_{PS}$, $\bar{s}_{CS}$; vect
      $\overline{TN}_{DS}$, $\overline{TN}_{PS}$, $\overline{CMOVE}$;
      $\overline{DS}$ = $\overline{SF}$ of $\overline{DRIVE}$;
      $\overline{PS}$ = $\overline{SF}$ of $\overline{PART}$ ;
      $\overline{MODDS}$ = $\overline{MOD}$ of $\overline{DRIVE}$ ;
      $\overline{MODPS}$ = $\overline{MOD}$ of $\overline{PART}$ ;
      $\overline{FIOPDS}$ = $\overline{FIOP}$ of $\overline{DRIVE}$ ;
      $\overline{FIOPPS}$ = $\overline{FIOP}$ of $\overline{PART}$ ;
      $\bar{\tau}_{DS}$ = $\bar{\tau}$ of $\overline{DRIVE}$;
      $\bar{\tau}_{PS}$ = $\bar{\tau}$ of $\overline{PART}$ ;
1:    $\overline{TEK}$ = $\overline{TE}$ ;
      DISDS = AMIND ($\overline{TA}$, $\overline{TEK}$, $\overline{TD}$, $\overline{DS}$, $\overline{MODDS}$, 0 , $\overline{FIOPDS}$);
      $\bar{s}_{DS}$ = $\bar{s}$ of $\overline{DISDS}$;
      $\overline{TN}_{DS}$ = $\overline{TN}$ of $\overline{DISDS}$;
      $\overline{TP}_{DS}$ = $\overline{TP}$ of $\overline{DISDS}$;
      $\overline{DISPS}$=AMIND ($\overline{TA}$,$\overline{TEK}$,$\overline{TD}$,$\overline{PS}$,$\overline{MODPS}$,0,$\overline{FIOPPS}$);
      $\bar{s}_{PS}$ = $\bar{s}$ of $\overline{DISPS}$;
      $\overline{TN}_{PS}$ = $\overline{TN}$ of $\overline{DISPS}$;
      $\overline{TP}_{PS}$ = $\overline{TP}$ of $\overline{DISPS}$;
```

```
        if $|\bar{s}_{DS}| < 1/20 \ \bar{\tau}_{DS}$ then goto 4 else goto 3
4:      if $|\bar{s}_{PS}| < 1/20 \ \bar{\tau}_{PS}$ then goto 5 else goto 3
3:      $\bar{s}_{CS} = 0$;
        $\overline{TN}_{CS} = \overline{TN}_{DS} \wedge \overline{TN}_{PS}$;
        $\overline{CMOVE}$ = GAUSS (x-of $\overline{TN}_{DS}$, y-of $\overline{TN}_{DS}$, z-of $\overline{TN}_{DS}$, $\bar{s}_{DS}$,
                               (x-of $\overline{TN}_{PS}$, y-of $\overline{TN}_{PS}$, z-of $\overline{TN}_{PS}$, $\bar{s}_{PS}$,
                               (x-of $\overline{TN}_{CS}$, y-of $\overline{TN}_{CS}$, z-of $\overline{TN}_{CS}$, $\bar{s}_{CS}$);
        $\overline{TE} = \overline{TEK} + \overline{CMOVE}$;
        if $\overline{MODDS}$ = ON then goto 10 else
        if ($\overline{TEK}$ - $\overline{TE}$ ) intersect $\overline{DS}$ then
        begin
              $\overline{MODDS}$ = REVERSE ($\overline{MODDS}$);
              goto 10;
        end
10:     if $\overline{MODPS}$ = ON then goto 1 else
        if ($\overline{TEK}$ - $\overline{TE}$) intersect $\overline{PS}$ then
        begin
              $\overline{MODPS}$ = REVERSE ($\overline{MODPS}$);
              goto 1 ;
        end
        goto 1;
5:      $\overline{TE}$ of CENTR2 = $\overline{TEK}$ ;
        $\overline{TA}$ of CENTR2 = $\overline{TA}$ ;
        $\overline{TN}_{DS}$ of CENTR2 = $\overline{TN}_{DS}$ ;
        $\overline{TN}_{PS}$ of CENTR2 = $\overline{TN}_{PS}$ ;
        $\overline{TP}_{DS}$ of CENTR2 = $\overline{TP}_{DS}$ ;
        $\overline{TP}_{PS}$ of CENTR2 = $\overline{TP}_{PS}$ ;
end
```

26. The DELTA routine

Aim: to compute two scalar quantities ($\overline{\text{ZL}}$), ($\overline{\text{DP}}$) as elements in ARLM3 for testing whether a given "cut-vector" (tool-pos $\overline{\text{INITIAL}}$, tool-pos $\overline{\text{FINAL}}$) is acceptable; and to check whether the cutter, while executing a given "cut-vector" might exceed the tolerance limits ($\overline{\tau}_{DS}$, $\overline{\tau}_{PS}$) of the two $\overline{\text{D}}$- and $\overline{\text{P}}$- control surfaces. According to whether an input-switch ($\overline{\text{IGOUG}}$) is set at 0 or at 1, the check is omitted or is performed; finally, an output switch ($\overline{\text{UGOUG}}$) is set at 1 when the ($\overline{\text{IGOUG}}$) is one and, also, the cutter is found overshooting the tolerance of at least one control surface; otherwise the output switch is set at 0.

Process: the routine is defined by the following procedure

```
cutmax = (real $\overline{ZL}$ , real $\overline{DP}$ , switch $\overline{UGOUG}$)

cutmax proc DELTA (tool-pos $\overline{INITIAL}$, tool-pos $\overline{FINAL}$,
                   tool-data $\overline{TD}$, surf $\overline{DS}$, surf $\overline{PS}$,
                   tolerance $\overline{\tau}_{DS}$, tolerance $\overline{\tau}_{PS}$ ,
                   switch $\overline{IGOUG}$).

begin
```

```
vect $\overline{TAK},\overline{TA},\overline{TNK}_{DS},\overline{TN}_{DS},\overline{TNK}_{PS},\overline{TN}_{PS},\overline{SN}_{PS},\overline{SN}_{DS}$;
point $\overline{TEK},\overline{TE},\overline{TPK}_{DS},\overline{TP}_{DS},\overline{TPK}_{PS},\overline{TP}_{PS},\overline{P1}_{PS},\overline{P2}_{PS},\overline{P1}_{DS}$,
      $\overline{P2}_{DS}$;
real $\overline{ZL}_{DS},\overline{ZL}_{PS},\overline{RC}_{DS},\overline{RC}_{PS},\overline{DP}_{PS}\overline{DP}_{DS}$; switch $\bar{C}$;
$\overline{TAK} = \overline{TA}$ of $\overline{INITIAL}$;
$\overline{TEK} = \overline{TE}$ of $\overline{INITIAL}$;
$\overline{TPK}_{DS}=\overline{TP}_{DS}$ of $\overline{INITIAL}$;
$\overline{TNK}_{DS}=\overline{TN}_{DS}$ of $\overline{INITIAL}$;
$\overline{TPM}_{PS}=\overline{TP}_{PS}$ of $\overline{INITIAL}$;
$\overline{TNK}_{PS}=\overline{TN}_{PS}$ of $\overline{INITIAL}$;
$\overline{TA} = \overline{TA}$ of $\overline{FINAL}$ ;
$\overline{TE} = \overline{TE}$ of $\overline{FINAL}$ ;
$\overline{TP}_{DS}= \overline{TP}_{DS}$ of $\overline{FINAL}$;
$\overline{TN}_{DS}=\overline{TN}_{DS}$ of $\overline{FINAL}$ ;
$\overline{TP}_{PS}=\overline{TP}_{PS}$ of $\overline{FINAL}$ ;
$\overline{TN}_{PS}=\overline{TN}_{PS}$ of $\overline{FINAL}$ ;
$\overline{P1}_{PS}$ = AXIS-POINT ($\overline{TPK}_{PS}$, $\overline{TNK}_{PS}$);(1)
$\overline{P2}_{PS}$ = AXIS-POINT ($\overline{TP}_{PS}$, $\overline{TN}_{PS}$);
$\overline{ZL}_{PS} = |\overline{P2}_{PS} - \overline{P1}_{PS}|$ ;
```

1) This procedure computes a point of the "tool-axis" as follows: given a point $\overline{TP}$ of the tool and the normal $\overline{TN}$ to the tool in $\overline{TP}$, it displaces $\overline{TP}$ back along $\overline{TN}$ as far as one cutter radius, and projects the point thus found on the "tool-axis".

```
$\overline{SN}_{PS}$ = UNRMAL ($\overline{TP}_{PS}$, $\overline{PS}$);
$\overline{RC}_{PS}$ = $\bar{S}$ of CCURV ($\overline{PS}$, $\overline{P1}_{PS}$, $\overline{P2}_{PS}$, $\overline{SN}_{PS}$);
$\overline{DP}_{PS}$ = 2 SQRT [ $\bar{\tau}_{PS}$(2 $\overline{RC}_{PS}$ - $\bar{\tau}_{PS}$)];
```

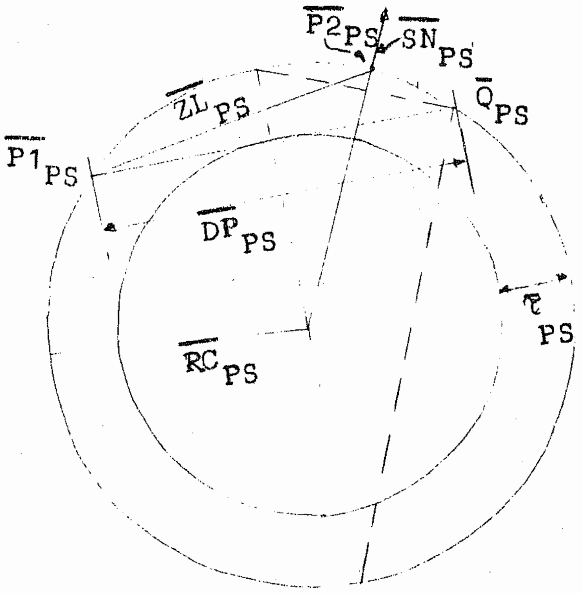


fig. 48

```
$\overline{P1}_{DS}$ = AXIS-POINT ($\overline{TPK}_{DS}$, $\overline{TNK}_{DS}$);
$\overline{P2}_{DS}$ = AXIS-POINT ($\overline{TP}_{DS}$, $\overline{TN}_{DS}$);
$\overline{ZL}_{DS}$ = $\left| \overline{P2}_{DS} - \overline{P1}_{DS} \right|$ ;
$\overline{SN}_{DS}$ = UNRMAL ($\overline{TP}_{DS}$, $\overline{DS}$);
$\overline{RC}_{DS}$ = $\bar{S}$ of CCURV ($\overline{DS}$, $\overline{P1}_{DS}$, $\overline{P2}_{DS}$, $\overline{SN}_{DS}$);
$\overline{DP}_{DS}$ = 2 SQRT [ $\bar{\tau}_{DS}$(2 $\overline{RC}_{DS}$ - $\bar{\tau}_{DS}$)];
$\overline{DP}$ of DELTA = MIN ($\overline{DP}_{PS}$, $\overline{DP}_{DS}$);
if $\overline{DP}$ of DELTA = $\overline{DP}_{PS}$ then
begin
    $\overline{ZL}$ of DELTA = $\overline{ZL}_{PS}$;
```

```
            goto 2;
        end
        $\overline{ZL}$ of DELTA = $\overline{ZL}_{DS}$;
   2:   $\overline{UGOUG}$ of DELTA = 0;
        if $\overline{IGOUG}$ = 1 then
        begin
            $\overline{C}$ = GOUGCK ($\overline{INITIAL}$, $\overline{FINAL}$,$\overline{TD}$,$\overline{DS}$,$\overline{PS}$,$\overline{\tau}_{DS}$,$\overline{\tau}_{PS}$);
            if $\overline{C}$ = 1 then $\overline{UGOUG}$ of DELTA = 1 else goto 1
        end
 1: end
```

Remarks: computations for defining on the cutter surfaces the two points $\overline{P1}_{PS}$ and $\overline{P2}_{PS}$ (required for an appropriate $\overline{RC}_{PS}$ to compute $\overline{DP}_{PS}$) are rather laborious, as the analogous computations for the $\overline{D}$-surface. Here follow some comments about. Neglecting the subscript PS for sake of simplicity, first of all we should say that to compute $\overline{DP}$ from an $\overline{RC}$ obtained by means of the points $\overline{TPK}$ and $\overline{TP}$ has been shown by experience to be little convenient; as just a look at the fig.s 44 and 45 below may help us to realize.

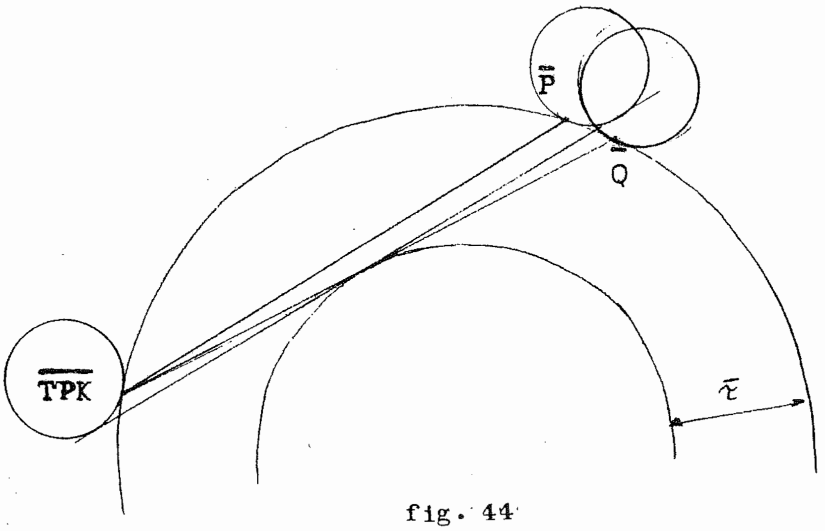


fig. 44

Infact, in a case such as this in fig.44, with a convex surface, the value $|\overline{Q} - \overline{TPK}|$ obtained for DP would exceed the optimal value $|\overline{P} - \overline{TPK}|$ that is allowed to the point $\overline{TPK}$ within the tolerance limits admitted.

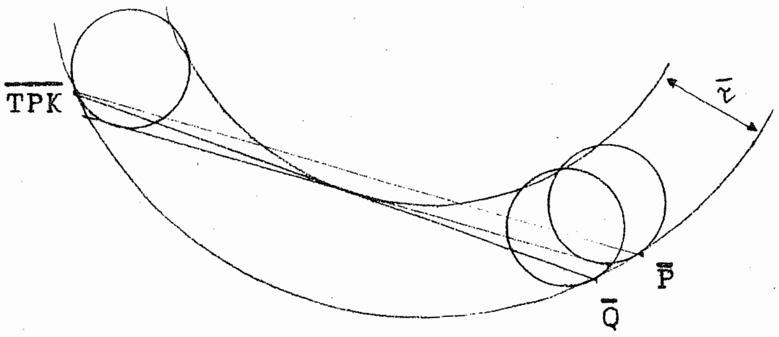


fig.45

Instead, in a case of concave surface such as in fig. 45, the value $|\overline{Q} - \overline{TPK}|$ obtained for DP would be less than the optimal value $|\overline{P} - \overline{TPK}|$ allowed to the point $\overline{TPK}$ within the tolerance limits admitted.

In general, in bi-dimensional problems the points $\overline{P1}$ and $\overline{P2}$ are obtained by projecting $\overline{TPK}$ and $\overline{TP}$ on the "tool-axis". In such cases , computations for $\overline{DP}$ are optimal, see fig.s 46 and 47. This is infact said "the method of projections".

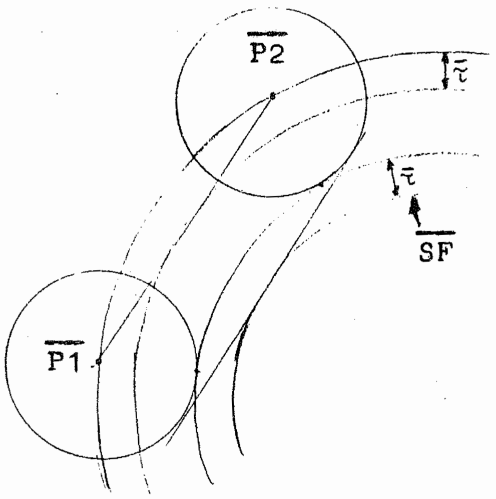


fig. 46 - convex surface

In three-dimensional problems, the projection method is replaced with the more complex process executed by the AXIS-POINT routine.

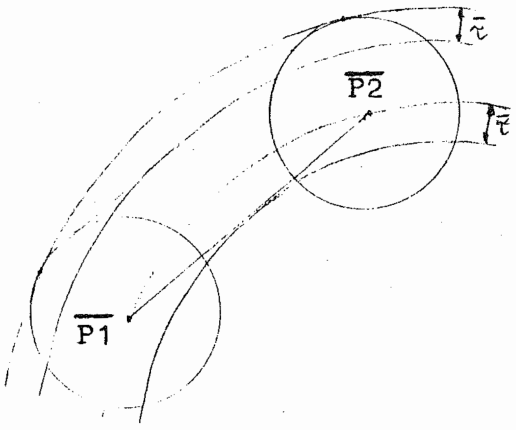


fig. 47 - concave surface

Fig.48 shows a spherical tool-cutter working on a cylindrical surface, and it can be seen that the projection method would yield progressively decreasing $\overline{DP}$.s, while the AXIS-POINT routine yields $\overline{DP}$.s of constant length, which seems intuitively clear for we are dealing with the successive positions of a sphere with respect to a cylinder.

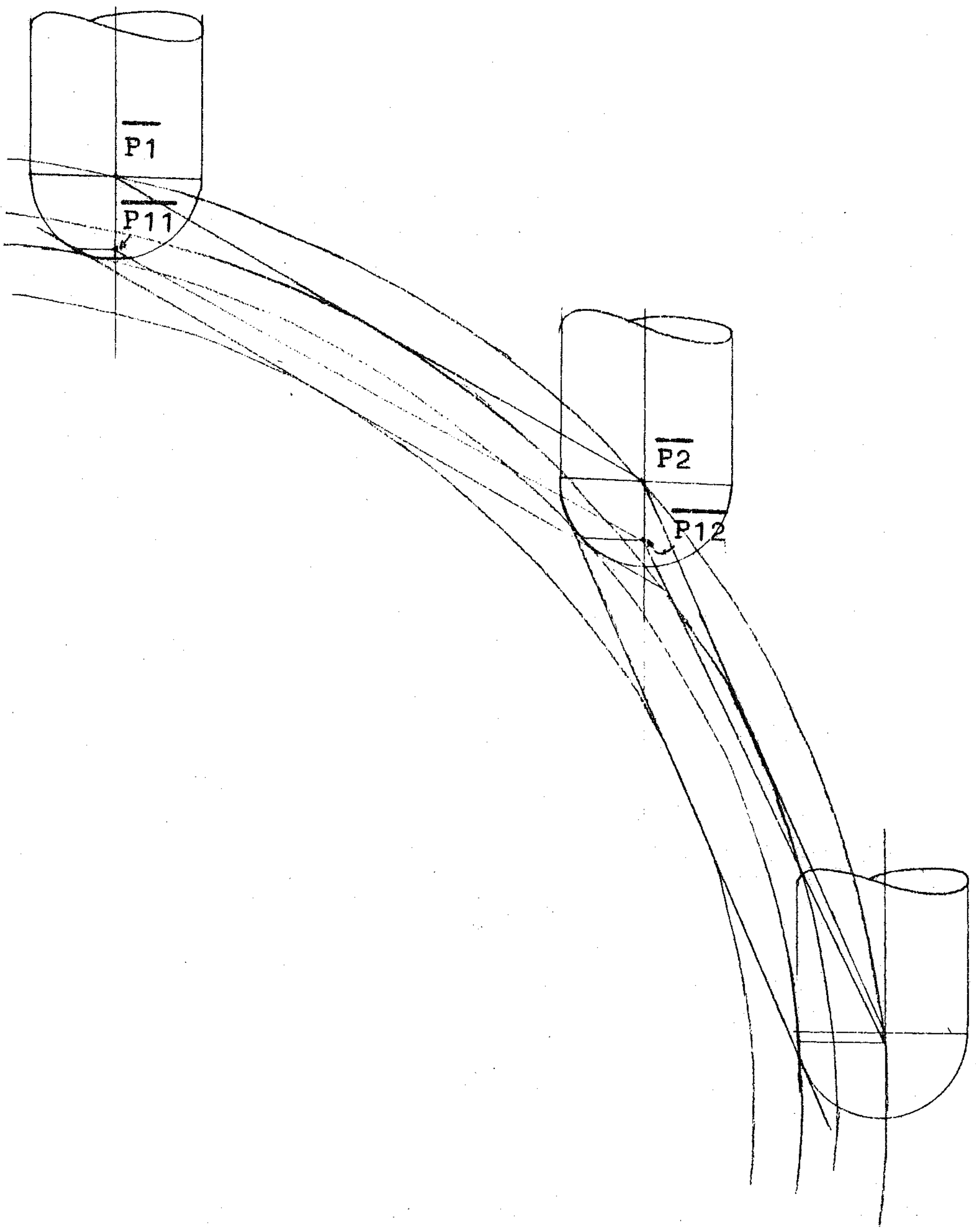


fig. 48

27. The GOUGCK routine

Aim: to check whether the cutter would violate surface tolerances ($\bar{\tau}_{DS}, \bar{\tau}_{PS}$) of at least one of the control surfaces ($\overline{PART}$, $\overline{DRIVE}$) while executing a proposed "cut-vector" (tool-pos INITIAL, tool-pos FINAL); if it does, a switch will be set at 1, otherwise, the switch is set at 0.

Remarks: this routine intervenes only if explicitly requested in the "part-program" by the "GOUGE/ON" statement, and only when the tool is in TO-position with respect to the control surfaces.

Process: the routine is defined by the following procedure

```
switch proc GOUGCK (tool-pos $\overline{INITIAL}$, tool-pos $\overline{FINAL}$,
                    tool data $\overline{TD}$, surf $\overline{DS}$, surf $\overline{PS}$,
                    tolerance $\bar{\tau}_{DS}$ , tolerance $\bar{\tau}_{PS}$)
begin
    vect $\overline{TAK}, \overline{TA}, \overline{SN}_{PS}, \overline{SN}_{DS}$; real $\bar{S}_{PS}, \bar{S}_{DS}$;
    point $\overline{TEK}, \overline{TE}, \overline{TPK}_{DS}, \overline{TP}_{DS}, \overline{TPK}_{PS}, \overline{TP}_{PS}, \overline{TE}_{O}, \overline{CC}_{PS}, \overline{CC}_{DS}$
    point-array $\overline{TETE}$ [1:2]; sexamind $\overline{ABC}$, $\overline{LMN}$;
    $\overline{TAK}$ = $\overline{TA}$ of $\overline{INITIAL}$;
    $\overline{TEK}$ = $\overline{TE}$ of $\overline{INITIAL}$;
    $\overline{TPK}_{DS}$ = $\overline{TP}_{DS}$ of $\overline{INITIAL}$;
    $\overline{TPK}_{PS}$=$\overline{TP}_{PS}$ of $\overline{INITIAL}$;
    $\overline{TA}$ = $\overline{TA}$ of $\overline{FINAL}$ ;
    $\overline{TE}$ = $\overline{TE}$ of $\overline{FINAL}$ ;
    $\overline{TP}_{DS}$=$\overline{TP}_{DS}$ of $\overline{FINAL}$ ;
    $\overline{TP}_{PS}$=$\overline{TP}_{DS}$ of $\overline{FINAL}$ ;
    $\overline{TETE}$ [1]: = $\overline{TE}$ ;
```

```
        $\overline{TETE}$ [2]: = $\overline{TE}$ ;
        $\overline{TE}_O$ = $\overline{TETE}$[1];
1:      $\overline{SN}_{PS}$ = UNRMAL ($\overline{TP}_{PS}$ , $\overline{PS}$);
        $\overline{CC}_{PS}$ = $\overline{PI}$ of CCURVE ($\overline{PS}$,$\overline{TPK}_{PS}$,$\overline{TP}_{PS}$,$\overline{SN}_{PS}$);
        $\overline{TE}_O$ = POINT-MIDWAY ($\overline{TEK}$,$\overline{TE}_O$,$\overline{CC}_{PS}$); (1)
        $\overline{ABC}$ = AMIND ($\overline{TA}$,$\overline{TE}_D$,$\overline{TD}$,$\overline{PS}$,TO,0,0);
        $\overline{TP}_{PS}$= $\overline{TP}$ of $\overline{ABC}$;
        $\overline{SN}_{PS}$ = $\overline{SN}$ of $\overline{ABC}$ ;
        $\bar{S}_{PS}$ = $\bar{S}$ of $\overline{ABC}$ ;
        if $|(\overline{TEK}-\overline{TE}_O) \times \overline{SN}_{PS}| \geqslant 0{,}0005$ then goto 1
        if $-\bar{S}_{PS} \geqslant \frac{1}{20} \quad \bar{\tau}_{PS}$  then goto 4
        $\overline{TE}_O$ = $\overline{TETE}$[2] ;
5:      $\overline{SN}_{DS}$ = UNRMAL ($\overline{TP}_{DS}$,$\overline{DS}$);
        $\overline{CC}_{DS}$ = $\overline{PI}$ of CCURVE ($\overline{DS}$, $\overline{TPK}_{DS}$, $\overline{TP}_{DS}$, $\overline{SN}_{DS}$);
        $\overline{TE}_O$ = POINT-MIDWAY ($\overline{TEK}$, $\overline{TE}_O$, $\overline{CC}_{DS}$);
        $\overline{LMN}$ = AMIND ($\overline{TA}$,$\overline{TE}$,$\overline{TD}$,$\overline{DS}$,TO,0,0);
        $\overline{TP}_{DS}$ = $\overline{TP}$ of $\overline{LMN}$ ;
        $\overline{SN}_{DS}$ = $\overline{SN}$ of $\overline{LMN}$ ;
        $\bar{S}_{DS}$ = $\bar{S}$ of $\overline{LMN}$ ;
        if $|(\overline{TEK} - \overline{TE}_O) \times \overline{SN}_{DS}| \geqslant 0{,}0005$  then goto 5
```

1) This procedure, given three points, computes the point where the plane that contains the third point and is normal to the straight line through the first two ones intersects this line.

```
        6:   if $|\bar{s}_{DS}| < \frac{1}{20}\bar{\tau}_{DS}$   then goto 4
        7:   GOUGCK= 0
             goto 9 ;
        4:   GOUGCK = 1;
   9:end
```

Remarks: the process, in short, executes a double iteration, one for the "drive-, one for the "part-surface". At each step of the iteration the tool-end is positioned closer and closer to one point of the applied vector $\overline{TE}$ - $\overline{TEK}$ where the vector $\overline{TN}$ normal to the tool is almost perpendicular to such a vector; here the check intervenes, to determine whether the cutter is within or without the tolerance in question, first for the "part-, then for the "drive-surface". The current step is shown in fig. 49.

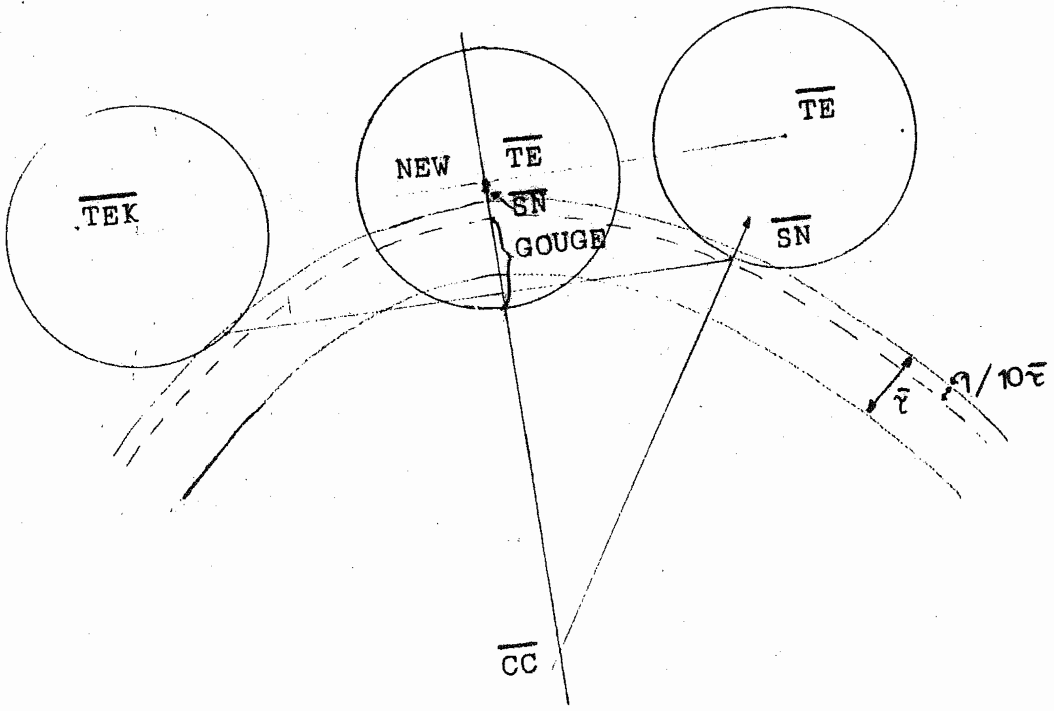


fig. 49

28. The CHECK1 routine

Aim: to compute a normal distance (S) of the tool (tool-end TEK, tool-axis TA, tool-data TD) from the "check-surface (surface-data CHECK); an approximate distance (CSD) of the tool from the said surface along a given direction (TIK); and the value of a control parameter (STCK1). Other required data: the "drive-surface" (DS), the corresponding modifier (MOD DS), the "part-surface" (PS).

Remarks: for simplicity sake, the CHECK routine of Section 2 in this report has been split into three simpler routines, CHECK1, CHECK2, and CHECK3; therefore they are not, properly speaking, "routines" of Section 2.

Process: the routine is defined by the following procedure

```
trireal = (real S, real  CSD, real STCK)
trireal proc. CHECK1 (point TEK, vect TA, tool-data TD,
                       vect TIK, surface-data  CHECK,
                       surf.DS , modifier MODDS, surf PS).
begin
     real S ; vect TN, TNI, indicator Z; surf SF ;
     modifier MODCS ; flag FIOP;

     CS = SF of CHECK ;
     MODCS = MOD of CHECK;
     FIOPCS = FIOP of CHECK ;
```

```
        if a tangent od pseudo-tangent case is involved(1)
        then goto 1
        $\overline{S} = \overline{S}$ of AMIND ($\overline{TA},\overline{TEK},\overline{TD},\overline{CS},\overline{MODCS},1,\overline{FIOPCS}$);
        $\overline{TN} = \overline{TN}$ of AMIND($\overline{TA},\overline{TEK},\overline{TD},\overline{CS},\overline{MODCS},1,\overline{FIOPCS}$);
        $\overline{Z} = \overline{Z}$ of AMIND ( $\overline{TA},\overline{TEK},\overline{TD},\overline{CS},\overline{MODCS},1,\overline{FIOPCS}$);
        $\overline{CSD}$ of CHECK1 = $\overline{S}/\overline{TN} \times \overline{TIK}$;
        $\overline{STCK}$ of CHECK1 = $\overline{Z} \times \overline{S}$;
        $\overline{S}$ of CHECK1 = $\overline{S}$ ;
        goto 2;
   1:   $\overline{S} = \overline{S}'$ of CPLAN ($\overline{CS}, \overline{DS},\overline{PS},\overline{TEK}$);
        $\overline{TN1} = \overline{TN}$  of CPLAN ($\overline{CS},\overline{DS},\overline{PS},\overline{TEK}$);
        $\overline{CSD}$  of CHECK1 = $\overline{S}/\overline{TN1} \times \overline{TIK}$;
        $\overline{TN}$=TN of AMIND ($\overline{TA},\overline{TEK},\overline{TD},\overline{CS},\overline{MODCS},1,\overline{FIOPCS}$);
        $\overline{STCK}$ of CHECK1= $\overline{TN} \times \overline{TIK}$;
        $\overline{S}$ of CHECK1 = $\overline{S}$ ;
2:end
```

Remarks: the two figures below illustrate the routine in two distinct cases.

1) the input parameters $\overline{DS}$ and $\overline{MODDS}$ are taken into account just to solve this problem.

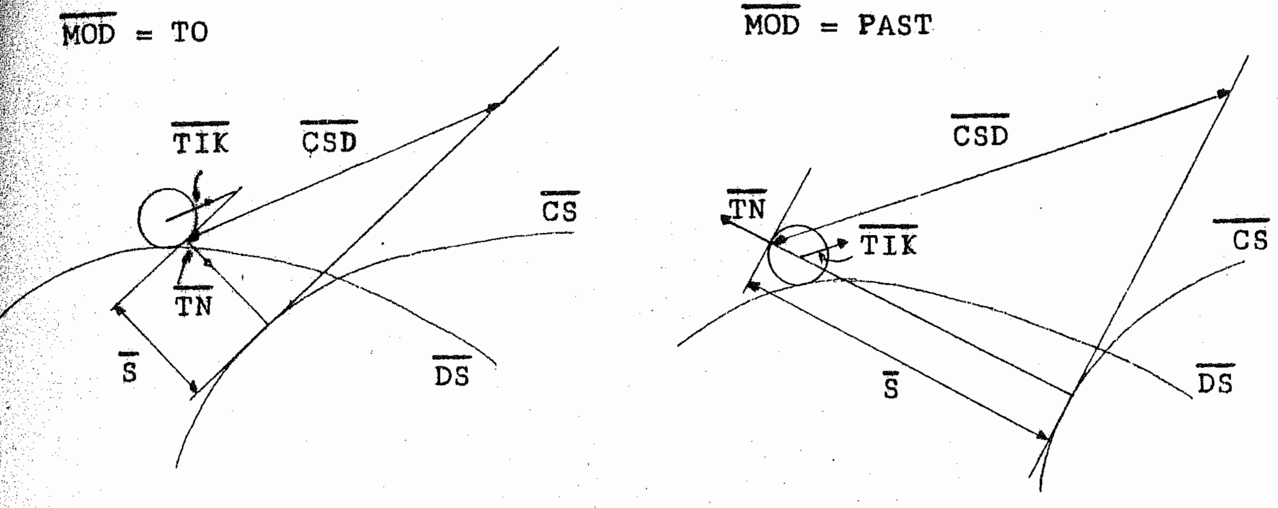


fig.50 - case of non-tangency

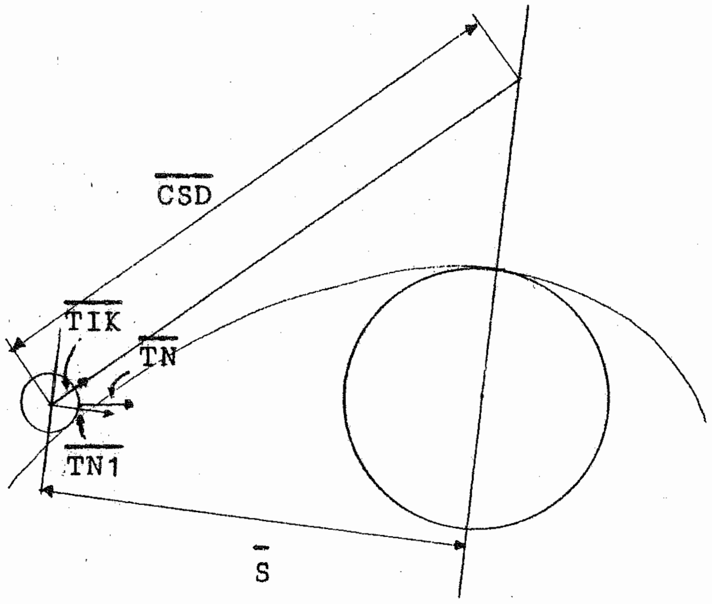


fig.51 - case of tangency

29. The CHECK2 routine

Aim: to check whether a given "cut-vector" can be held as a "final" cut-vector, and to compute the approximate distance ($\overline{CSD}$) from the tool (tool-axis $\overline{TA}$, tool-data $\overline{TD}$) to the "check-surface" (surface-data$\overline{CHECK}$) along a given direction ($\overline{TIK}$). Further data required: the forward motion direction at the extremes of the cut-vector ($\overline{TIK}$,$\overline{TI}$), two scalar quantities ($\overline{S}$,$\overline{DP}$) to characterize the origin of the cut-vector ($\overline{TEK}$) and the cut-vector itself, the drive-surface ($\overline{DS}$) with its modifier, and the part-surface ($\overline{PS}$). The output parameter $\overline{FINCUT}$ shall be set at 1 if the cut-vector is final; otherwise, at 0.

Process: the routine is defined by the following procedure

```
adupla = (switch $\overline{FINCUT}$, real $\overline{CSD}$)
adupla proc CHECK2 (point $\overline{TEK}$, vect $\overline{TIK}$, vect $\overline{TI}$,
                    surface-data $\overline{CHECK}$, surf $\overline{DS}$,
                    modifier MOODS, surf $\overline{PS}$, vect $\overline{TA}$,
                    tool-data $\overline{TD}$ , real $\overline{S}$, real $\overline{DP}$)
begin
    if $\{(|\overline{S}|-\overline{DP}) \geq 2\,\overline{DP} \vee [\overline{TI} \times \overline{TIK} \geq \cos(\pi/4)]\}$ then
        begin
            $\overline{FINCUT}$ of CHECK2 = 0;
            $\overline{CSD}$ of CHECK2 = 0 ;
        end
        $\overline{FINCUT}$ of CHECK2 = 1;
        $\overline{CSD}$ of CHECK2 = $\overline{CSD}$ of CHECK1 ($\overline{TEK}$, $\overline{TA}$, $\overline{TD}$, $\overline{TIK}$,
                                     $\overline{CHECK}$,$\overline{DS}$,$\overline{MODDS}$,$\overline{PS}$);
end
```

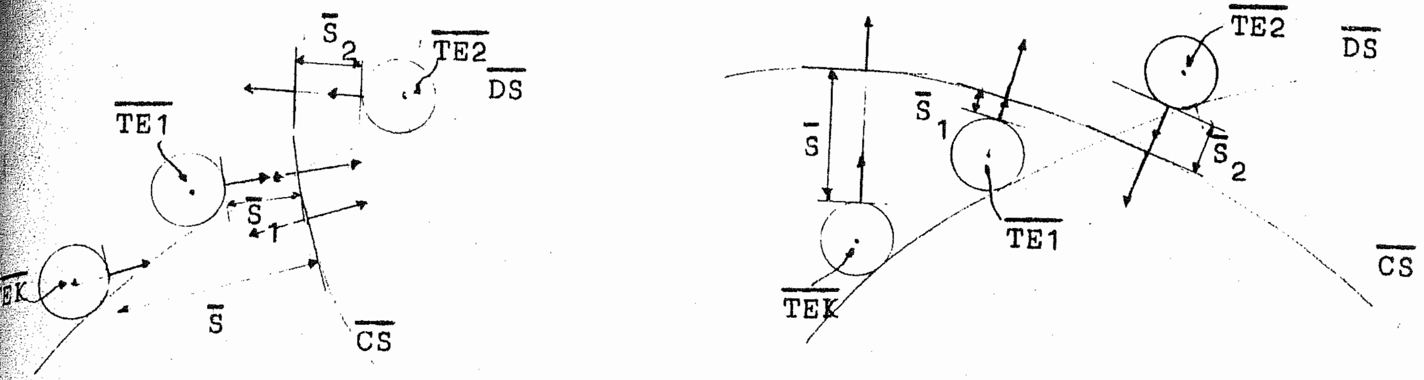
30. The CHECK3 routine

Aim: to compute a scalar quantity for checking whether the tool (tool-axis TA, tool-data TD) in a given position (TE) with a given forward motion vector (TI) is over-shooting the check-surface (surface-data CHECK) with respect to an initial position with a suitable parameter (STCK). The scalar is negative if the tool overshoots.

Process: the routine is defined by the following procedure

```
real proc CHECK3 (point TE, vect TI, real STCK,
                  surface data CHECK, vect TA,
                  tool-data TD)
begin
    real $\bar{S}$, indicator $\bar{S}$, surf CS, flag FIOPCS,
    modifier MODCS;
    CS = SF of CHECK ;
    MODCS = MOD of CHECK;
    FIOPCS = FIOP of CHECK;
   if a tangent or pseudo-tangent case is involved
   then goto 1
   $\bar{S}$ = $\bar{S}$ of AMIND (TA,TE,TD,CS,MODCS,1,FIOPCS);
   $\bar{Z}$ = $\bar{Z}$ of AMIND (TA,TE,TD,CS,MODCS,1,FIOPCS);
   CHECK3 = $\bar{Z} \cdot \bar{S} \cdot$ STCK ;
       goto 2 ;
   1:  TN = TN of AMIND (TA,TE,TD,CS,MODCS,1,FIOPCS);
       CHECK3 = STCK $\cdot$ (TN $\times$ TI);
2:end
```

Remarks: some comments and two figures to illustrate the CHECK3 computation



(a) fig.52 - a case of non-tangency (b)

in the (a) case one has:

$\overline{STCK} = \overline{S} \cdot \overline{Z} < 0$, con $\overline{S} > 0$ e $\overline{Z} = -1$

CHECK3 $(\overline{TE1}) = \overline{STCK} \cdot \overline{S}_1 \cdot \overline{Z}_1 > 0 \quad (\overline{S}_1 > 0\ ,\ \overline{Z}_1 = -1)$

CHECK3 $(\overline{TE2}) = \overline{STCK} \cdot \overline{S}_2 \cdot \overline{Z}_2 < 0 \quad (\overline{S}_2 > 0,\ \overline{Z}_2 = +1)$

in the (b) case one has:

$\overline{STCK} = \overline{S} \cdot \overline{Z} > 0$ con $\overline{S} > 0$ e $\overline{Z} = 1$

CHECK3 $(\overline{TE1}) = \overline{STCK} \cdot \overline{S}_1 \cdot \overline{Z}_1 > 0 \quad (\overline{S}_1 > 0\ ,\ \overline{Z}_1 = 1)$

CHECK3 $(\overline{TE1}) = \overline{STCK} \cdot \overline{S}_2 \cdot \overline{Z}_2 < 0 \quad (\overline{S}_2 > 0,\ \overline{Z}_2 = -1)$

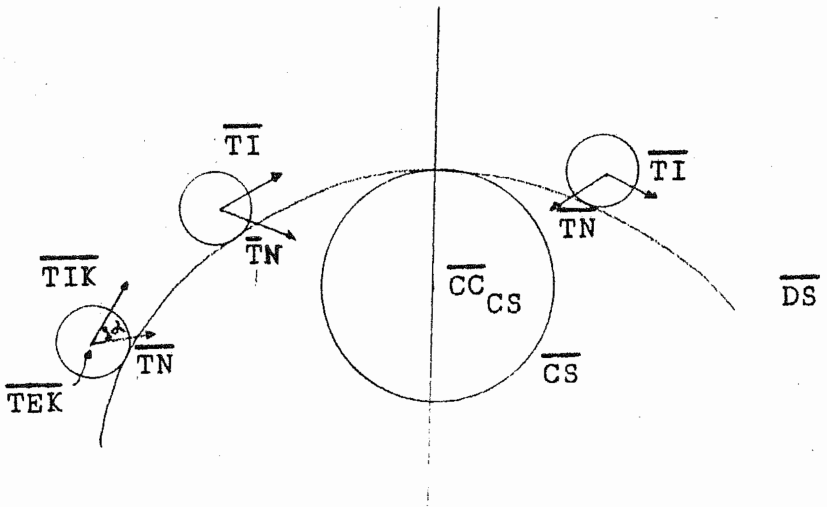


fig.53 - a case of tangency

31. The ADELTA routine

Aim: to define, from a given position ($\overline{\text{TEI}}$), a tool position (tool-axis $\overline{\text{TA}}$, tool-data$\overline{\text{TD}}$) within tolerance of the D- and P-surfaces ($\overline{\text{DRIVE}}$,$\overline{\text{PART}}$) compatibly with a reference position ($\overline{\text{INITIAL}}$) of the tool. The check switch ($\overline{\text{IGOUG}}$) ,for accepting the cut-vector (tool-pos $\overline{\text{INITIAL}}$,tool-pos $\overline{\text{FIN}}$) has been previously introduced; to compute, besides, a scalar quantity ($\overline{\text{DP}}$) to characterize the above cut-vector, and a vector ($\overline{\text{TI}}$) for defining the forward motion direction of the tool in the cut-vector final position ($\overline{\text{FIN}}$).

Remarks: the ADELTA routine does not belong to Section 2.

Process: the routine is defined by the following procedure

```
good-pos (tool-pas $\overline{\text{FIN}}$, real $\overline{\text{DP}}$, vect $\overline{\text{TI}}$)
good-pos proc ADELTA (point $\overline{\text{TEI}}$, tool-pos $\overline{\text{INITIAL}}$,
vect $\overline{\text{TA}}$, tool-data $\overline{\text{TD}}$, switch $\overline{\text{IGOUG}}$ , surface-data $\overline{\text{DRIVE}}$,
surface-data $\overline{\text{PART}}$)
begin:
surf.$\overline{\text{DS}}$,$\overline{\text{PS}}$, point $\overline{\text{TEK}}$, $\overline{\text{TEI}}$; real $\overline{\text{DP}}$,$\overline{\text{PL}}$; vect $\overline{\text{TN}}_{PS}$
$\overline{\text{TN}}_{DS}$ , tool-pos $\overline{\text{FIN}}$; cutmax $\overline{\text{TRIS}}$ ,tolerance $\overline{\tau}_{PS}$,
$\overline{\tau}_{DS}$).
$\overline{\text{TEK}}$ = $\overline{\text{TE}}$ of $\overline{\text{INITIAL}}$;
$\overline{\tau}_{PS}$ = $\overline{\tau}$ of $\overline{\text{PART}}$;
$\overline{\tau}_{DS}$ = $\overline{\tau}$ of $\overline{\text{DRIVE}}$;
$\overline{\text{PS}}$ = $\overline{\text{SF}}$ of $\overline{\text{PART}}$ ;
$\overline{\text{DS}}$ = $\overline{\text{SF}}$ of $\overline{\text{DRIVE}}$;
```

```
1:  $\overline{FIN}$ = CENTR2 ($\overline{DRIVE}$, $\overline{PART}$, $\overline{TA}$, $\overline{TEI}$, $\overline{TD}$);
    $\overline{TRIS}$ = DELTA ($\overline{INITIAL}$' $\overline{FIN}$, $\overline{TD}$, $\overline{DS}$, $\overline{PS}$, $\bar{\tau}_{DS}$, $\bar{\tau}_{PS}$,
                    $\overline{IGOUG}$);
    $\overline{ZL}$ = $\overline{ZL}$ of $\overline{TRIS}$ ;
    $\overline{DP}$ = $\overline{DP}$ of $\overline{TRIS}$ ;
    $\overline{UGOUG}$ = $\overline{UGOUG}$ of $\overline{TRIS}$ ;
    if $\overline{DP} < \overline{ZL}$ v ($\overline{ZL} \leq \overline{DP} \wedge \overline{UGOUG} = 1$) Then
    begin
        $\overline{TEI} = \overline{TEK} + 0{,}65 \left| \overline{TEI} - \overline{TEK} \right|$;
        goto 1;
    end
    $\overline{TN}_{PS}$ = $\overline{TN}_{PS}$ of $\overline{FIN}$ ;
    $\overline{TN}_{DS}$ = $\overline{TN}_{DS}$ of $\overline{FIN}$ ;
    $\overline{TI}$ of ADELTA = FOR WARD ($\overline{TEI}-\overline{TEK}$, $\overline{TN}_{PS} \wedge \overline{TN}_{DS}$);(1)
    $\overline{FIN}$ of ADELTA = $\overline{FIN}$ ;
    $\overline{DP}$ of ADELTA = $\overline{DP}$ ;
end
```

1) Given two vectors, this procedure orients the second one with respect to the first one so that their scalar product is non-negative.

32. The ARLM3 routine

Aim: to compute the sequence of successive positions ($\overline{\text{TEK}}$, $\overline{\text{TE}}[1]$, $\overline{\text{TE}}[2]$, ..., $\overline{\text{TEE}}$) of a tool (tool-data $\overline{\text{DT}}$) that is associated with an intermediate or with a terminal continuous motion command; namely, the motion data are those concerning the three control surfaces ($\overline{\text{DRIVE}}$, $\overline{\text{PART}}$, $\overline{\text{CHECK}}$), the initial tool position ($\overline{\text{INITIAL}}$) within tolerance of the D- and the P-surfaces, and the forward motion direction vector ($\overline{\text{TIK}}$) in such a position. The check-switch ($\overline{\text{IGOUG}}$) for accepting in sequence the final cut-vectors is as previously said.

Remarks: the preliminary computations required to make available some input data of the ARLM3 routine ($\overline{\text{DRIVE}}$, $\overline{\text{PART}}$, $\overline{\text{CHECK}}$, $\overline{\text{TIK}}$) are performed by the AREPRE routine(1) that starts from an intermediate or from a terminal continuous motion command, and takes into account the preceding-and also, possibly, the ensuing-motion command in the "part-program". Namely, AREPRE checks whether the initial tool position for motion is within tolerance of the two D- and P-control surfaces, computes the initial motion direction ($\overline{\text{TIK}}$), sets several flags, controls, and modifiers for computing the data concerning the control surfaces ($\overline{\text{DRIVE}}$, $\overline{\text{PART}}$, $\overline{\text{CHECK}}$).

This routine shall not be examined in detail in this report, which ends with the ARLM3 routine; the latter, infact, is comprehensive of the whole set of routines in Section 2, and is capable of obtaining a satisfactory motion trajectory by interpolating linearly the successive tool positions computed from suitable input data.

Process: the routine is defined by the following procedure

(1) see fig.4

```
cutseq = (point TEK, point-array TE, point TEE)

cutseq proc ARLM3 (tool-pos INITIAL, vect TIK,tool-data
                   TD, switch IGOUG, surface-data DRIVE,
                   surface-data PART, surface-data CHECK)

begin

     vect TA, TI, TIK; real DP, S, CSD, STCK, A ; integer i ;

     point TEK, TEI, TE, TEE ; switch UGOUG ;

     surf DS,PS; modifier MODDS; tolerance $\bar{\tau}_{DS}$, $\bar{\tau}_{PS}$;

     tool-pos FINAL,INITIAL; good-pos TRIS;cutmax ABC;

     adupla BIS;

     TA = TA of INITIAL ;

     TEK = TE of INITIAL;

     DS = SF of DRIVE;

     MODDS = MOD of DRIVE (3) ;

     $\bar{\tau}_{DS}$ = $\bar{\tau}$ of DRIVE ;

     PS = SF of PART ;

     $\bar{\tau}_{PS}$ = $\bar{\tau}$ of PART ;

A:   TEK of ARLM3 = TEK ;

     i = 0 ;

B:   TEI = TEK + 0.125.TIK ;

     FINAL = CENTR2 (DRIVE, PART, TA, TEI, TD);

     DP = DP of DELTA (INITIAL , FINAL, TD,DS,PS, $\bar{\tau}_{DS}$, $\bar{\tau}_{PS}$,

     IGOUG);
```

2) Obviously, the modifiers concerning the D- and the P-surface can be only TO or ON

```
    $\overline{S} = \overline{S}$ of CHECK1 ($\overline{TEK},\overline{TA},\overline{TD},\overline{TIK},\overline{CHECK},\overline{DS},\overline{MODDS},\overline{PS}$);
    $\overline{CSD} = \overline{CDS}$ of CHECK1 ($\overline{TEK},\overline{TA},\overline{TD},\overline{TIK},\overline{CHECK},\overline{DS},\overline{MODDS},\overline{PS}$);
    $\overline{STCK} = \overline{STCK}$ of CHECK1($\overline{TEK},\overline{TA},\overline{TD},\overline{TIK},\overline{CHECK},\overline{DS},\overline{MODDS},\overline{PS}$);
    $\overline{DP} = \overline{MIN} (\overline{DP}, \overline{CSD})$;
    $\overline{TEI} = \overline{TEK} + \overline{DP} \cdot \overline{TIK}$;
    $\overline{TRIS}$=ADELTA ($\overline{TEI},\overline{INITIAL},\overline{TA},\overline{TD},\overline{IGOUG},\overline{DRIVE},\overline{PART}$);
    $\overline{TI}$ = FORWARD ($\overline{TIK}$, $\overline{TI}$ of $\overline{TRIS}$);
    $\overline{DP} = \overline{DP}$ of $\overline{TRIS}$;
    $\overline{TE} = \overline{TE}$ of $\overline{FIN}$ of $\overline{TRIS}$;
C:  $\overline{BIS}$=CHECK2 ($\overline{TEK},\overline{TIK},\overline{TI},\overline{CHECK},\overline{DS},\overline{MODDS},\overline{PS},\overline{TA},\overline{TD},\overline{S},\overline{DP}$);
    if ($\overline{FINCUT}$ of $\overline{BIS}$= 0) $\vee$ [($\overline{FINCUT}$ of $\overline{BIS}$=1) $\wedge$ ($\overline{CSD}$ of $\overline{BIS} > \overline{DP}$)]
    then
D:  begin
          $\overline{i} = \overline{i} + 1$ ;
          $\overline{TE}[\overline{i}]$ of ARLM3 = $\overline{TE}$;
          $\overline{TEK} = \overline{TE}[\overline{i}]$ of ARLM3;
          $\overline{TIK} = \overline{TI}$ ;
          $\overline{INITIAL} = \overline{FIN}$ of $\overline{TRIS}$;
          goto B;
    end
E:  $\overline{FINAL}$ = CENTR ($\overline{DRIVE},\overline{PART},\overline{CHECK},\overline{TA},\overline{TE},\overline{TD}$);
    $\overline{TEE} = \overline{TE}$ of $\overline{FINAL}$;
```

```
F: $\overline{ABC}$ = $\overline{DELTA}$ ($\overline{INITIAL}$, $\overline{FINAL}$, $\overline{TD}$,$\overline{DS}$,$\overline{PS}$,$\tilde{\tau}_{DS}$, $\tilde{\tau}_{PS}$,$\overline{IGOUG}$);
   $\overline{ZL}$ = $\overline{ZL}$ of $\overline{ABC}$ ;
   $\overline{DP}$ = $\overline{DP}$ of $\overline{ABC}$ ;
   $\overline{UGOUG}$ = $\overline{UGOUG}$ of $\overline{ABC}$ ;
   if $\overline{ZL} \leqslant \overline{DP} \wedge \overline{UGOUG} = 0$ then
G: begin
       $\overline{TEE}$ of ARLM3 = $\overline{TEE}$ ;
       goto 1 ;
   end
H: $\overline{DP} = 0{,}65 \cdot |\overline{TEE} - \overline{TEK}|$;
   $\overline{TEI} = \overline{TEK} + \overline{DP} - \overline{TIK}$;
   $\overline{TRIS}$=ADELTA ($\overline{TEI}$,$\overline{INITIAL}$,$\overline{TA}$,$\overline{TD}$,$\overline{IGOUG}$,$\overline{DRIVE}$,$\overline{PART}$);
   $\overline{TI}$ = FORWARD ($\overline{TIK}$, $\overline{TI}$ of $\overline{TRIS}$);
   $\overline{TE}$ = $\overline{TE}$ of $\overline{FIN}$ of $\overline{TRIS}$;
L: $\bar{A}$ = CHECK3 ($\overline{TE}$,$\overline{TI}$,$\overline{STCK}$,$\overline{CHECK}$,$\overline{TA}$,$\overline{TD}$);
   if $\bar{A} < 0$ then*a diagnostic will be printed":cut-vector
   iteration to check-surface failed"* else
M: $\bar{i} = \bar{i} + 1$;
   TE[$\bar{i}$] of ARLM3 = $\overline{TE}$;
   $\overline{TIK}$ = $\overline{TI}$;
   $\overline{TEK}$ = $\overline{TE}$[$\bar{i}$] of ARLM3;
   $\overline{INITIAL}$ = $\overline{FIN}$ of $\overline{TRIS}$;
   goto E;
1:end
```

Remarks: here follows a flow-chart for ARLM3, with some comments.

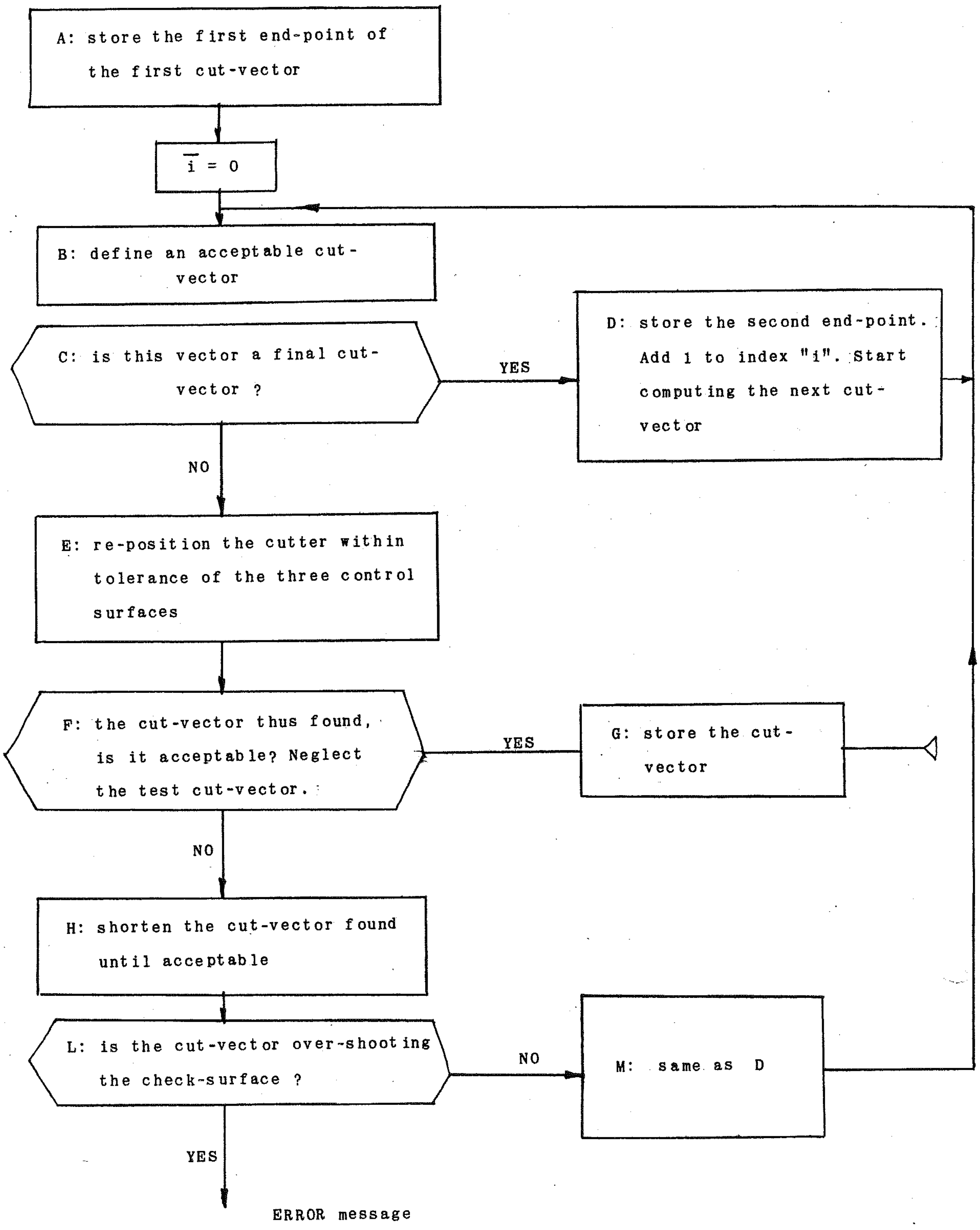

A: store the first end-point of the first cut-vector
$\bar{i} = 0$
B: define an acceptable cut-vector
C: is this vector a final cut-vector ?
YES
D: store the second end-point. Add 1 to index "i". Start computing the next cut-vector
NO
E: re-position the cutter within tolerance of the three control surfaces
F: the cut-vector thus found, is it acceptable? Neglect the test cut-vector.
YES
G: store the cut-vector
NO
H: shorten the cut-vector found until acceptable
L: is the cut-vector over-shooting the check-surface ?
NO
M: same as D
YES
ERROR message

fig. 54

The "B" block in the flow-chart is - as the whole ARLM3 routine, after all - strongly heuristic; infact, an acceptable cut-vector is being determined as follows: the cutter is displaced from the starting location $\overline{TEK}$ along the forward motion direction $\overline{TIK}$ by a fixed length, then iterated within tolerance limits for the D-and P-surfaces. From the last and the initial position the maximum acceptable cut-vector is computed. The cutter is displaced again along $\overline{TIK}$ starting from $\overline{TEK}$, by a distance that is the lesser between the quantity $\overline{DP}$ and the approximate $\overline{CSD}$ distance of the cutter, in the $\overline{TEK}$ position, from the check-surface. At this point the ADELTA routine iterates the cutter again within the tolerance limits of the two control surfaces, and, if the cut-vector thus defined is acceptable, the "C" block comes in, otherwise the $\overline{DP}$ quantity is reduced by a fixed amount, and the process of displacing along $\overline{TIK}$ and iterating within tolerances is resumed until an acceptable cut-vector is finally defined, see fig.55.

The "C" block determines whether the cut-vector thus found (TE - TEK in fig. 55) is a final cut-vector; we recall that this is indead the case when the angle of the $\overline{TI}$ and $\overline{TIK}$ vectors is greater than 25^{o}, or when the normal distance of the cutter, in $\overline{TEK}$ position, from the check-surface, is greater than three times $\overline{DP}$; in such a case, the "E" block comes in: the cutter is iterated from $\overline{TE}$ within tolerance limits of the three control surfaces in the $\overline{TEE}$ position; then, the "F" block, neglecting the $\overline{TE}$-$\overline{TEK}$ cut-vector, determines whether the $\overline{TEE}$-$\overline{TEK}$ is straightforwardly acceptable; if it is so, the computation ends, see fig.56.

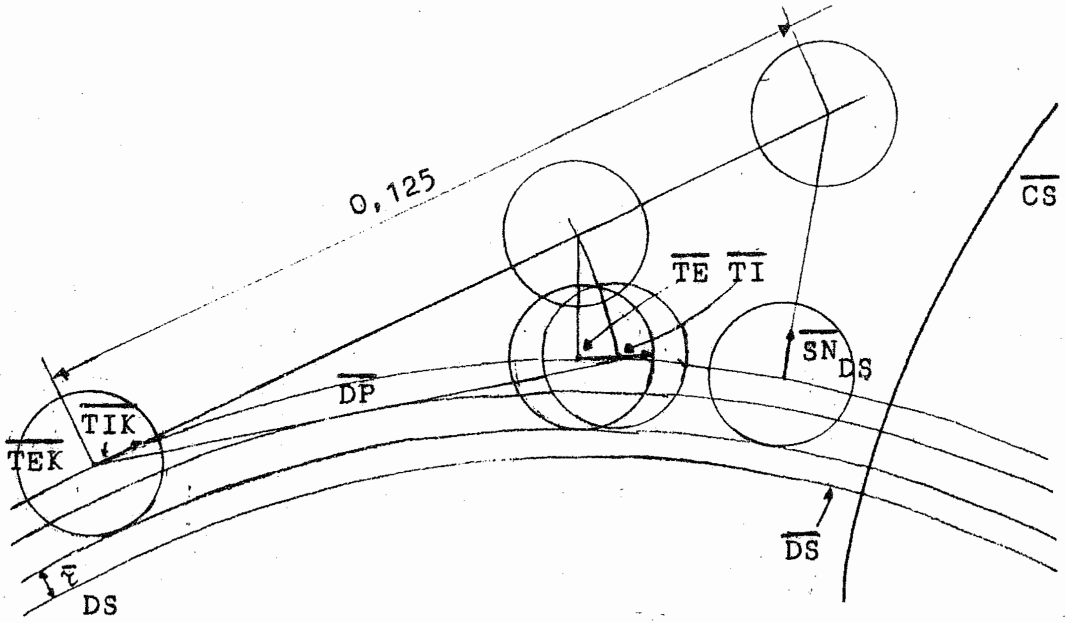

0,125
$\overline{CS}$
$\overline{TE}$ $\overline{TI}$
$\overline{SN}_{DS}$
$\overline{DP}$
$\overline{TIK}$
$\overline{TEK}$
$\overline{DS}$
$\bar{\tau}$
DS

fig. 55

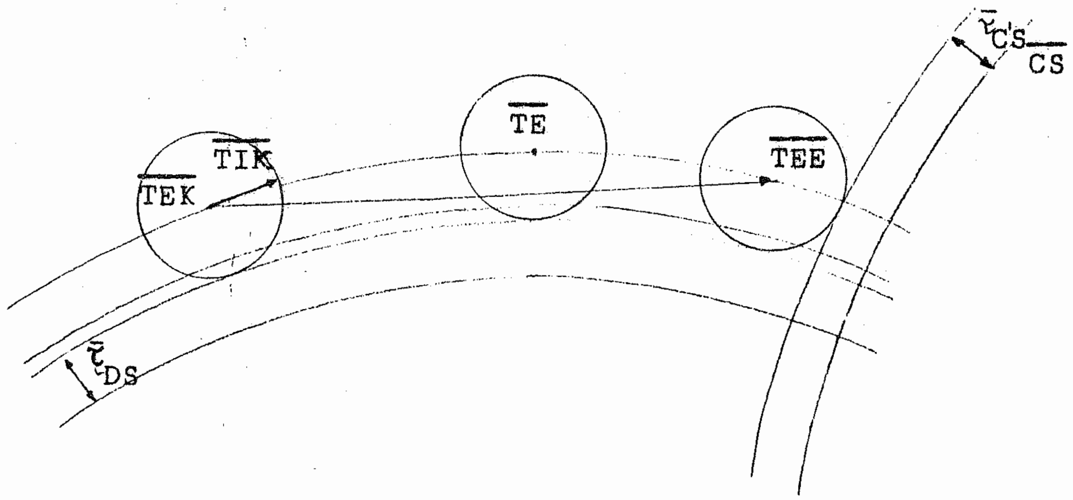

$\bar{\tau}_{CS}$
$\overline{CS}$
$\overline{TE}$
$\overline{TEE}$
$\overline{TIK}$
$\overline{TEK}$
$\bar{\tau}_{DS}$

fig. 56

Otherwise (see fig. 57), the "H" block displaces the tool along $\overline{TIK}$ by a distance $\overline{DP} = 0,65\ |\overline{TEE} - \overline{TEK}|$ and iterates it within tolerance limits of the P- and the D-surface (fig.58); if the cut-vector is not acceptable, the whole procedure is repeated with a progressively decreasing $\overline{DP}$.

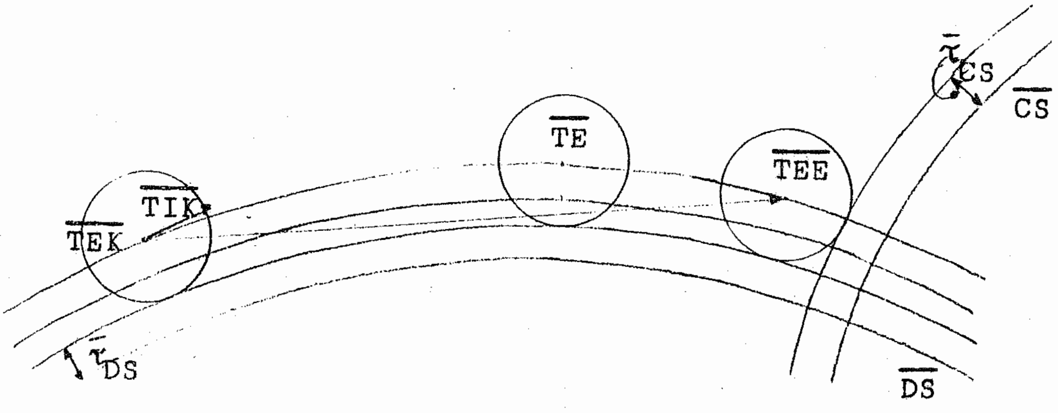


fig.57

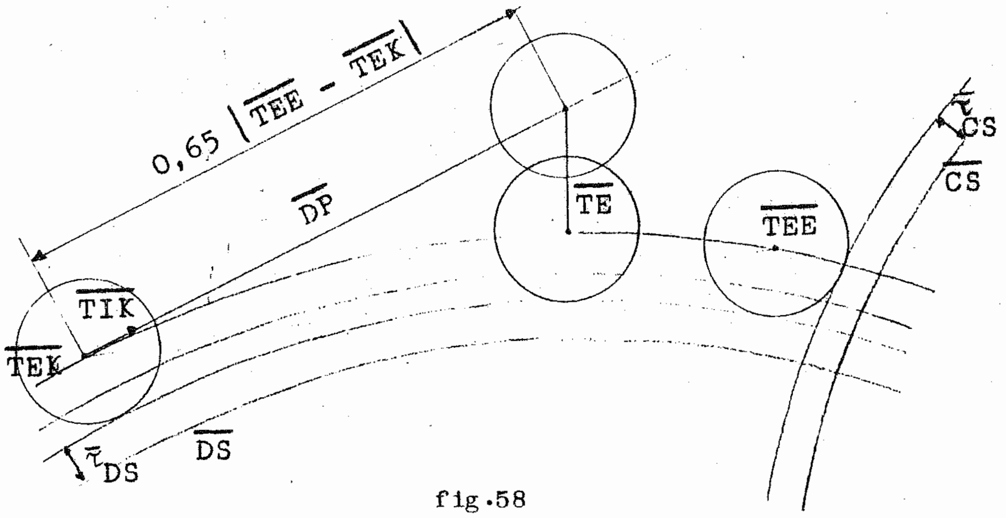


fig.58

The last check before accepting the final cut-vector is obviously that for making sure that it is not overshooting the check-surface.

Alphabetical list of Input, Output, and Computational Parameters [o]

$\underline{\text{adupla}} = (\underline{\text{switch}}\ \overline{\text{FINCUT}},\ \underline{\text{real}}\ \overline{\text{CSD}})$

$\underline{\text{ajundo}} = (\underline{\text{real}}\ \bar{\text{S}},\ \underline{\text{point}}\ \overline{\text{SP}},\ \underline{\text{norm}}\ \overline{\text{SN}},\ \underline{\text{indicator}}\ \bar{\text{Z}})$

$\underline{\text{angle}} = \underline{\text{real}}$

$\underline{\text{bis}} = (\underline{\text{real}}\ \bar{\text{S}},\ \underline{\text{point}}\ \overline{\text{PI}})$

$\underline{\text{bool}} = \{\underline{\text{true}},\ \underline{\text{false}}\}$

$\underline{\text{complex}} : (\underline{\text{real}}\ \bar{\text{A}},\ \underline{\text{real}}\ \bar{\text{B}})$

$\underline{\text{cone}} = (\underline{\text{real}}\ \bar{\text{A}},\ \underline{\text{real}}\ \bar{\text{B}},\ \underline{\text{real}}\ \bar{\text{C}},\ \underline{\text{real}}\ \bar{\text{D}},\ \underline{\text{real}}\ \bar{\text{E}},\ \underline{\text{real}}\ \bar{\text{F}},\ \underline{\text{real}}\ \bar{\text{G}})$ (1)

$\underline{\text{control}} = \underline{\text{switch}}$

$\underline{\text{cutmax}} = (\underline{\text{real}}\ \overline{\text{ZL}},\ \underline{\text{real}}\ \overline{\text{DP}},\ \underline{\text{switch}}\ \overline{\text{UGOUG}})$

$\underline{\text{cutseq}} = (\underline{\text{point}}\ \overline{\text{TEK}},\ \underline{\text{point-array}}\ \overline{\text{TE}},\ \underline{\text{point}}\ \overline{\text{TEE}})$

$\underline{\text{cylinder}} = (\underline{\text{real}}\ \bar{\text{A}},\ \underline{\text{real}}\ \bar{\text{B}},\ \underline{\text{real}}\ \bar{\text{C}},\ \underline{\text{real}}\ \bar{\text{D}},\ \underline{\text{real}}\ \bar{\text{E}},\ \underline{\text{real}}\ \bar{\text{F}},\ \underline{\text{real}}\ \bar{\text{G}})$ (1)

$\underline{\text{duo}} = (\underline{\text{vect}}\ \overline{\text{SN}},\ \underline{\text{indicator}}\ \bar{\text{Z}})$

$\underline{\text{dupla}} = (\underline{\text{point}}\ \overline{\text{TP}},\ \underline{\text{vect}}\ \overline{\text{TN}})$

$\underline{\text{flag}} = \underline{\text{integer}}$

$\underline{\text{goodpos}} = (\underline{\text{tool-pos}}\ \overline{\text{FIN}},\ \underline{\text{real}}\ \overline{\text{DP}},\ \underline{\text{vect}}\ \overline{\text{TI}})$

$\underline{\text{indicator}} = \{-1,\ 1\}$

---

[o] It is recalled that an underscored word stand for the set of elements named by that word.

(1) The arguments are the coefficients of the canonical form.

integer = $I^{(2)}$

matrix = $$\begin{pmatrix} \text{real } \bar{A} & \text{real } \bar{B} & \text{real } \bar{C} & \text{real } \bar{D} \\ \text{real } \bar{E} & \text{real } \bar{F} & \text{real } \bar{G} & \text{real } \bar{L} \\ \text{real } \bar{M} & \text{real } \bar{N} & \text{real } \bar{Q} & \text{real } \bar{P} \\ \text{real } \bar{R} & \text{real } \bar{S} & \text{real } \bar{T} & \text{real } \bar{U} \end{pmatrix}$$

modifier = {TO, ON, PAST, TANTO}

norm = vect

plane = (real $\bar{A}$, real $\bar{B}$, real $\bar{C}$, real $\bar{D}$)$^{(1)}$

point = (real x, real y, real z) $^{(4)}$

point-array = (point $\bar{P}_1$, point $\bar{P}_2$,...,point $\bar{P}_n$) $n \in I$

psc = (plane $\overline{PSC}$, norm $\overline{TN}$, real $\bar{S}$)

quadric = (real $\bar{A}$, real $\bar{B}$, real $\bar{C}$, real $\bar{D}$, real $\bar{E}$, real $\bar{F}$, real $\bar{G}$, real $\bar{H}$, real $\bar{L}$, real $\bar{M}$) $^{(1)}$

quareal = (real $\bar{A}$, real $\bar{B}$, real $\bar{C}$, real $\bar{D}$)

real = $R^{(3)}$

sexamind = (real $\bar{S}$, point $\overline{TP}$, point $\overline{SP}$, vect $\overline{TN}$, vect $\overline{SN}$, indicator $\bar{Z}$)

sphere = (real $\bar{X}$, real $\bar{Y}$, real $\bar{Z}$, real $\bar{R}$) $^{(1)}$

surf = {cylinder, cone, plane, quadric, sphere, tabcyl}

---

(1) coefficients of the canonical form

(2) for the set of integer numbers

3) for the set of real numbers

4) referring to a system of x,y,z, coordinates

surface-data = (surf. $\overline{SF}$, modifier $\overline{MOD}$, flag $\overline{FIOP}$, control $\overline{FCON}$, tolerance $\bar{\tau}$)

switch = $\{0,1\}$

tabcyl = (point $\bar{P}_1$, point $\bar{P}_2$,..point $\bar{P}_n$, vect $\bar{V}$)(5) $n \in I$

tolerance = real

tool- data = (real $\bar{A}$, real $\bar{B}$, real $\bar{C}$, real $\bar{D}$, real $\bar{E}$, real $\bar{F}$, real $\bar{G}$) (6)

tool-pos = (vect $\overline{TA}$, point $\overline{TE}$, point $\overline{TP}_{DS}$, vect $\overline{TN}_{DS}$, point $\overline{TP}_{PS}$, vect $\overline{TN}_{PS}$)

tricomplex = (complex $\overline{UI}$, complex $\overline{U2}$, complex $\overline{U3}$).

trireal = (real $\bar{S}$, real $\overline{CSD}$, real $\overline{STCK}$)

tris = (real $\bar{S}$, point $\bar{Q}$, norm $\overline{SN}$)

vect = (real x real y, real z) (4).

4) referring to a system of x,y,z, coordinates

5) These n points by a suitable interpolation determine the diretrix of the tabulated cylinder whose generatrix is V (the interpolation is realized by a sequence of cubic parabolas, one for each pair of points).

6) These seven coefficients determine the tool shape,

Some special words(o)

begin

else

end

goto

if

then

u-, v-, w- of	stands for the 1st, 2nd, 3rd coordinate of a point or of a vector in the (x,y,w) space analogously for
u_1-,v_1-,w_1-of	

(o) the words with no explanation are to be interpreted as in ALGOL 60.

Alphabetical Routine List (1)

(1) A starred routine name is one appearing in a paragraph title at the corresponding page; for an un-starred routine name the corresponding page is that where the routine is first quoted.

B I B L I O G R A P H Y

[1] S.C.Ambrosio, "NUCOL:a new language for N/C", FIAT int. pubbl.,STSA (1969).

[2] A.Caracciolo di Forino, "Sulla programmazione delle macchine utensili", IEI Research Report,Pisa (1968).

[3] A.Caracciolo di Forino, "High level problem oriented languages as advanced thinking tools",Atti del Convegno: "Il linguaggio nella società e nella tecnica",Milano,ott. 1968 (in corso di pubblicazione).

[4] A.Caracciolo di Forino, "The formal definition of machine tool languages", IEI Research Report, Pisa (1969).

[5] P.Hopewell,"Mathematical methods for cutter offset computations" IITRI, Draft paper for Prolamat (1969).

[6] "APT Dictionary" , IITRI (1965).

[7] "APT Encyclopedia",IITRI (1965).

[8] "APT Computer programs", IITRI (1965).

[9] "Proceedings of the APT technical meeting",IITRI (April 10-12, 1967).

[10] "Proceedings of the APT technical meeting",IITRI

(September 12-14, 1967).

[11] "Proceedings of the APT technical meeting", IITRI (April 9-11, 1968).

[12] "Proceedings of the APT technical meeting", IITRI (September 16-19, 1968).

[13] "First semi-annual report on ARELEM development", IITRI (January 1 - June 30, 1968).

[14] "Long range program", IITRI.

[15] "2CL part programming reference manual",N.E.L. (1967).

[16] "Il controllo numerico delle macchine utensili", IBM.

[17] "Numerical control programming for the AUTO-FROMT system", IBM-M &A - 12 (May 1961).

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Istituto di Fisica Università di Pisa (Italy)	2a. REPORT SECURITY CLASSIFICATION unclassified
	2b. GROUP

3. REPORT TITLE

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)* Scientific Interim.

5. AUTHOR(S) *(First name, middle initial, last name)*

6. REPORT DATE 21 July 1970	7a. TOTAL NO. OF PAGES 120	7b. NO. OF REFS 17
8a. CONTRACT OR GRANT NO. F61052 - 67 - C - 0097	9a. ORIGINATOR'S REPORT NUMBER(S) AER 1	
b. PROJECT NO. 9769-05		
c. 61445014	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*	
d. 681304		

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES TECH. OTHER	12. SPONSORING MILITARY ACTIVITY Air Force Office of Scientific Research - 1400 Wilson Boulevard ARLINGTON - Virginia 22209 (SRI)

13. ABSTRACT

In the computer-aided determination of a three-dimensional path for N.C. machine tools, the APT-III approach is described algorithmically, as a contribution to a much needed formalization, both for theoretical and practical purposes. The fundamental case of only one cheek-surface with one single intersection is considered. More than thirty routines are described in a language patterned after ALGOL 68.

DD FORM 1 NOV 65 1473

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Programming Language NC Machine Tool APT III System ARELEM Problem Algorithmic Description						