Chapter 4

ROBOT3D

ROBMOD

General Description

The simulation module and programming "offline", ROBMOD allows making the simulation of several cells containing a robot with a maximum of 6 axes and one or more posicioners, totaling a maximum of 6 external axes. The tool is selectable as well as the work cell.

The module ROBMOD includes several transformations of coordinates (inverse) including the more usual cinematic models. The inclusion of a robot with a specific transformation of coordinates involves the development of its inverse transformation and corresponding routine in C language and the "link" with ROBMOD. The operation of the module ROBMOD is based on menus and commands.

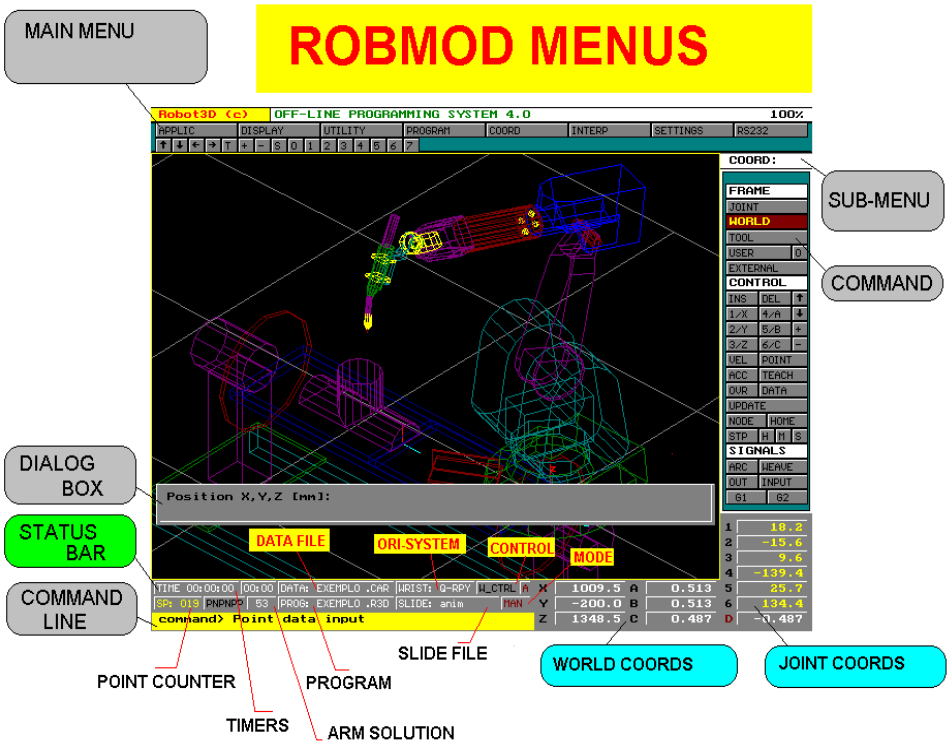


Fig. 4-1. ROBMOD screen areas.

The screen is divided in several areas:

- In the superior strip there is the main menu and the speed "override". During simulation the instructions of the program are visualized in the 3rd line

- In the right lateral strip there is the submenu if active and the corresponding commands. It is still used to show files. In the inferior area of this strip there is the display of the joint coordinates (robot or external axes), and still space for an auxiliary parameter (D).

- In the inferior strip we have an area corresponding to the "status" of the axes and command line, and an area with indication of the position and orientation of the robot relatively to the current system of coordinates, in agreement with the system of orientation of the WRIST .

- In the status bar there is information on the timers (TIME), the actual location (SP), the name of the Location file (DATA - locations), the program file (PROG - task) and of animation file (SLIDE). we can still obtain information about the system of orientation of the wrist, the type of control, the operation mode (MAN or AUT) and the type of coordinates (A - absolute; I - incremental). It is still indicated which is the configuration of the robot axes of the and respective code.

- The central area of the screen corresponds to the used Viewport

Commands and Menus

Each menu or command is activated by pressing the left button of the mouse or the initial of the command. In some cases of commands that begin for the same letter it can be necessary to carry in additional keys.

The left button of the mouse emulates the key ENTER and the right ESC. The arrows and PGUP/PGDN also have special meaning.

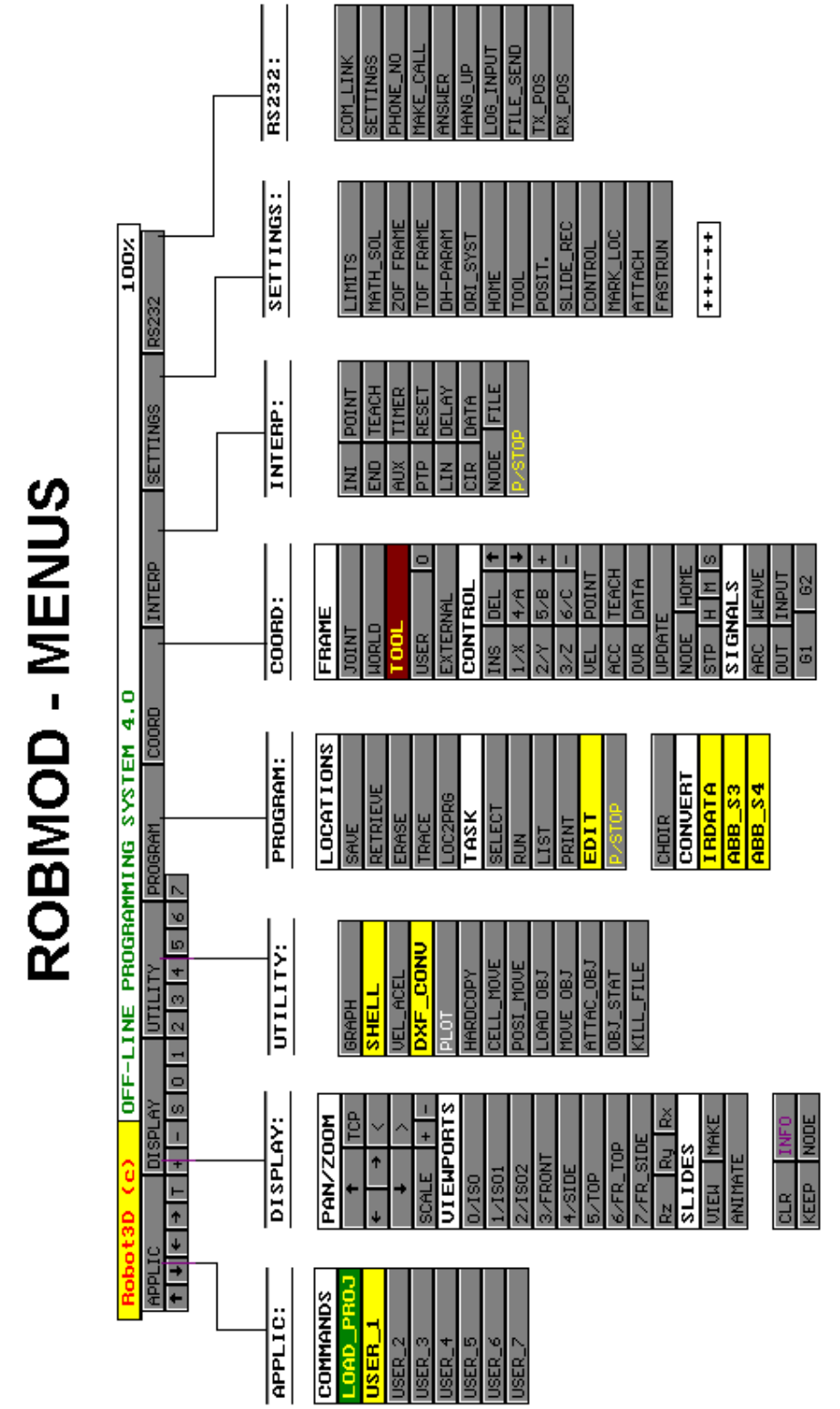


Fig. 4-2. ROBMOD menus.

The ROBMOD interface uses the following menus:

APPLIC - Menu of user Applications
Applications developed by the user using the ROBMOD library (option).
It involves the development of sources in C and compilation with the modules of the program. See the file APPLIC.DOC.

DISPLAY – Visualization Menu
Views, zoom, pan, rotation of the views according to axes Z, Y, and X. Visualização of the geometric knots. Animation, slides, etc.

UTILITY – Utility Menu
Movement graphs, DOS shell, "hardcopy", conversion of DXF, cell and posicioner adjustment, loading and displacement of objects. Coupling of objects, etc.

PROGRAM – Programming Menu
Automatic mode, execution of programs, verification of points, listing of programs and locations, edition of programs with RODIT editor, recording and conversion of locations, printing, conversion of programs (post-processing IRDATA-SIEMENS, ARLA(S3) / RAPID(S4)-ABB, etc).

COORD - Teach Pendant Menu
Manual mode, robot and posicioner movement / external axes, systems of coordinates JOINT, WORLD, TOOL, USER (9 different "frames" - ZOF) and EXTERNAL, learning of points and "playback", functions of grippers and welding, Inputs and Output signals (I/O), absolute and incremental movement, movement for a geometric knot, etc.

INTERP – Interpolation Menu
Demarcation of the trajectory, interpolation PTP, linear and circular in 3D, cycle time, recording of interpolation data (joint and linear values), distances, radius of circumferences etc.

SETTINGS – Configuration Menu
Joint Limits, ZOF (user frames), TOF (tool centers - tcp), tool change, Denavit-Hartenberg parameters, systems of orientation of the wrist, coupling of the positionar, teaching of coordinates, creation of animations, etc.

RS232 - Communications
Communications among the modules ROBMOD and an external program. In certain situations it is possible to move robots directly.

Coordinate Transformation

The file with the transformations of coordinates can be altered for creation of a new robot kinematics. The user should refer to the file TRANS.DOC (option) in the directory OBJ for information on this matter.

Obtaining Help

It is possible to obtain help carrying in F1 in any menu screen. The type of help depends on the actual menu.

APPLIC Menu

This menu is destined to allow to the advanced user the creation of an open interface. The user can build their specific routines using the ROBMOD library. The prototypes of the existent functions in the library are in the file ROB_LIB.H (option).

To use this "library" the file APPLIC.C (option) it should be edited, compiled and linked with the modules object included in the package

The user can build eight functions using his/her C code and the functions of the supplied "library." The content of those functions is only of the responsibility of the user.

The function 0 (invoqued by key 0) (LOAD_PROJ) it is an example and it allows to change the project without leaving ROBMOD. The functions 1 to 7 can also be used in to call an external program (By defect they do call the modules MOD?. EXE)

For larger explanation it should be read the file APPLIC.DOC (option) included in the OBJ directory.

DISPLAY Menu

The DISPLAY menu supplies a series of functions for visualization:

<n> VIEWPORTS
Modification of the views. Single or multiple views can be used

- 0/ISO - Isometric view
- 1/ISO1 - Isometric + projection (small windows)
- 2/ISO2 - Same. Windows of the same size
- 3/FRONT - Front view
- 4/SIDE - Side view
- 5/TOP - Top view

- 6/FR_TOP - Front/ Top view
- 7/FR_SIDE - Front/ Side view

NOTE: The disposition of the views depends on the rotations that are done according to Z,Y and X

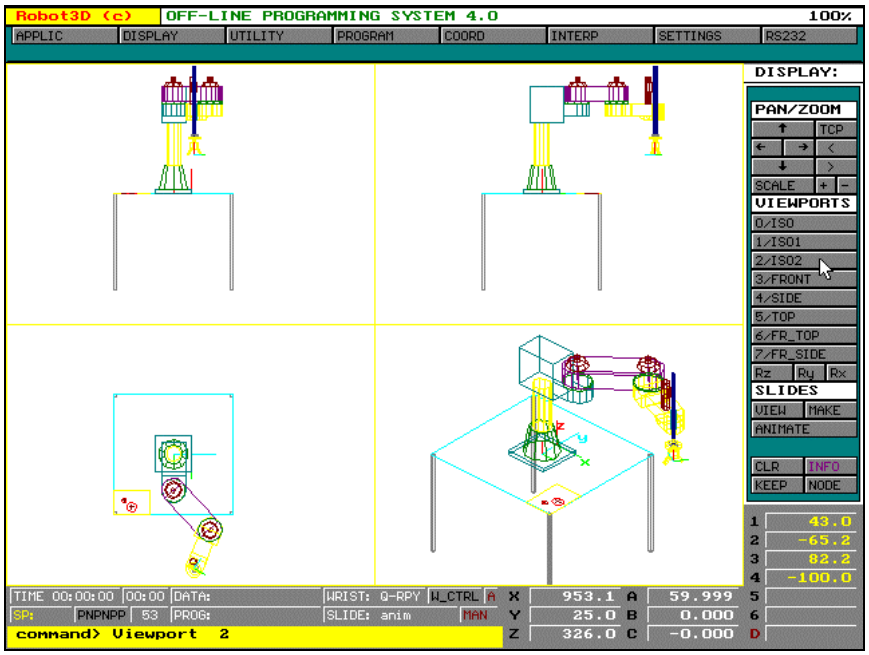


Fig. 4-3. Viewport ISO2.

SCALE
Change of the scale of the image. The scale can be modified by the keys + and - or introducing the scale factor.

PAN
The displacement (pan) of the image it is efectuated using the arrows and the factor (keys <and>). THE view can also be centered in the tool point (TCP - tool center point) of the robot

ROTATE (Rz, Ry, Rx)
Rotation of the views according to the axis Z, Y and X. If this function is activ the keys + and - they allow to do to vary the value, and SCALE to attribute a specific value. Note: function ON/OFF.

SLIDES
Slides can be visualized or recorded, or used a slides sequence to produce an animation effect.

ANIMATE
Runs a sequence of pré-recorded slides, if available. Lookat the help of the menu SETTINGS to see how a slides sequence is produced for animation.

KEEP
This function changes between the freshning of the screen or the override of images. It is useful to follow the evolution of the movements.

NODE
This ON/OFF function allows to visualize the number of the knots of the robot, positioner / external axes, and objects. The user can specify which the range to show.

CLR
Refreshes the screen.

INFO
Information of the program and copyright message.

UTILITY Menu

This menu supplies some utilities:

GRAPH
Graphic representation of the curves of movements - displacements, speeds and accelerations of the robot joints. The couple graph can also be seen if available. Please consult the help of the menu INTERP to see how to produce the data for the graphs of the movements.

VEL_ACEL
This function is used to obtain the speeds and accelerations of the robot joints by numeric differentiation starting from the file of displacements. The subsequent development of the "package" can include modules for the calculation of the couples in the committees starting from the files of the joints, transformation of coordinates and moments of masses.

SHELL
Makes a DOS call via the COMMAND. To use EXIT to return to ROBMOD. The busy memory for ROBMOD is about 490 kb.

DXF_CONV
This function, converts files DXF for format <. GEO> and the inverse
The user can also call externally utilitarian DXF2GEO.EXE and GEO2DXF.EXE to accomplish the same operation.
NOTE: The files DXF can only have the section ENTITIES. Only the entities 3DPOLY, 3DFACE and LINE are converted.

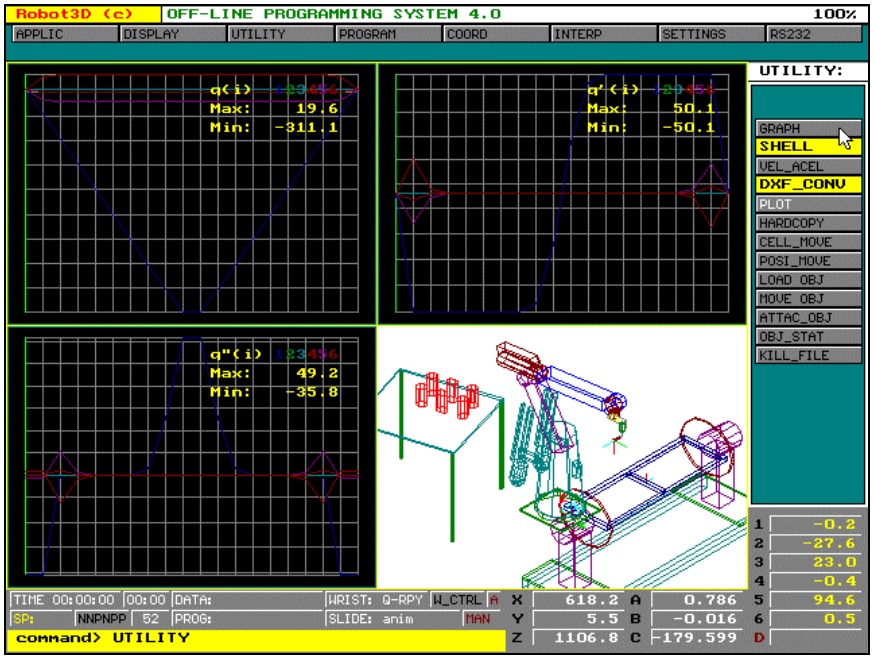


Fig. 4-4. Displacement, Velocity and Acceleration Graphfor a PTP motion

HARDCOPY
This function produces a "hardcopy" of the screen of the cell in format EPSON FX-80. Please verify the printer is connected and ON-LINE.

PLOT
This function, still no available, it will allow making the "plotagem" of the work cell.

CELL_MOVE and POSI_MOVE
These functions allow to readjust the position and orientation of the cell and of the positioner relatively to the referential WORLD of the robot. (Do not modify the position of the axes external unless necessary!)

LOAD_OBJ and MOVE_OBJ
This functions allow to manipulate objects inside of the cell. The objects can be put in any position and displacement (offset) relative the that position. The objects are geometric entities with extension <. GEO>. If an object is coupled (for instance to a posicionador - to see ATTAC_OBJ) only the "offset" can be altered, the position is calculated by the system. In automatic way it is possible to modify the position and offset of the objects with the instructions **OBJ_POS** and **OBJ_OFF**

ATTAC_OBJ
Allows to couple or to free objects. The objects can be coupled to a connection of the robot, to an external axis or TCP. In automatic way

we have the instructions **ATTACH_ROB_LINK** and **ATTACH_EXT_LINK** for coupling objects.

OBJ_STAT
This function allows to modify the presentation and color of the objects, and also to remove them.

KILL_FILE
Turns off files of the disk.

NOTES:
EPSON FX-80 is a mark registered of EPSON
IRDATA, ARLA and RAPID are designations used in the systems SIEMENS and ABB

PROGRAM Menu

The programming menu allows to implement an automatic mode of operation based on cycle logic.

The integrated interpreter of commands allows to read files of extension <. R3D> and to produce complete programs of robot. Data of points and conditions are loaded of separate files.

The interpreter of commands **R3D** especially developed for the package is an implementation of the language BASIC and it was drawn to obtain the maxim advantage of all of the simulation routines.

The files of programs can be created or edited by an external text editor. Please use the file ROBMOD.CFG to establish the directory and the edition program. The work file is passed for the editor as parameter. ROBMOD uses editor RODIT due to limitations of memory. To use other editor this it should occupy less than 100Kb.

If is not possible to run RODIT.EXE with resident ROBMOD it is necessary reconfigure the files CONFIG.SYS and way AUTOEXEC.BAT to have available about 600Kb of memory DOS.

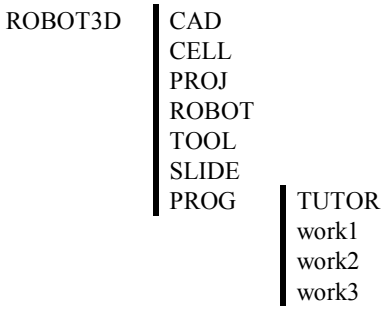
The search directory for programs and locations is defined in the file ROBMOD.CFG.

Example (file ROBMOD.CFG):

```
RODIT
1,0,1,8,1200,10,1,1
C:\ROBOT3D\PROG
```

The user shall modify the last line for the wanted directory. Alternatively the directory can be modified with the command *CHDIR*.

The basic directorys are the following ones:



The directories work1, work2 and work3 are examples directories created by the user to keep their programs and locations.

The size of a program is limited by the available memory, but given that she can chain programs through instructions CALL the only limitation is the disk memory.

The functions of this menu are divided in two different types:

- Functions related with points (locations)
- Functions related with cycle logic (task)

Functions relacted with points:

SAVE
Recording of data of points. Two different files are created, one for values of the joints and the other for Cartesian coordinates, respectively with extensions <. DAT> and <. CAR>. The conditions for processes, *ARC* and *WEAVE* are recorded in an extension file <. CND>. the configuration for the arm is included in the locations. In case the external axes have been taught about points are recorded also.
In the automatic mode RUN if it uses the teaching mode (**MARK_LOC** - menu SETTINGS) the locations are recorded by the system as the robot moves.

RETRIEVE
It recovers the file of points in coordinates of the joints <. DAT> or Cartesian <. CAR>, consonant the type of control (to see *CONTROL* in the menu *SETTINGS*). The automatic cycle will depend on this choice. If the file of coordinates Cartesian, different robots be used will execute the same task. If the file of joints is used any correction is done based in frames (ZOF).
The conditions for ARCWELD are recovered starting from the extension file <. CND>.
The data of the locations can be loaded in a program for an instruction **GET_DATA**.

Exemple of a <.CAR> file:

```
1,PTP X+2836.6,Y+2626.4,Z+810.0,A+179.999,B-0.000,C+180.000,21
2,PTP X+2836.6,Y+2626.4,Z+810.0,A+179.999,B-0.000,C+180.000,21
3,PPA R+1900.000,S+0.000,T+0.000,U+0.000,V+0.000,W+0.000
4,PTP X+2733.5,Y+1761.5,Z+810.0,A+154.999,B+0.000,C+180.000,21
5,LIN X+2733.5,Y+1461.5,Z+810.0,A+154.999,B+0.000,C+180.000,21
6,PTP X+2733.5,Y+1161.5,Z+810.0,A+129.999,B-0.000,C+180.000,21
```

(The point three corresponds to the external axis!)

NOTE: Check that a file <. CAR> it can be easly converted in a file <. R3D> if the numbers of the points are removed (of fact the command **LOC2PRG** operates according to a similar principle).

Example of a modified <. R3D> file:

```
PTP X+2836.6,Y+2626.4,Z+810.0,A+179.999,B-0.000,C+180.000,21
PTP X+2836.6,Y+2626.4,Z+810.0,A+179.999,B-0.000,C+180.000,21
PPA R+1900.000
PTP X+2733.5,Y+1761.5,Z+810.0,A+154.999,B+0.000,C+180.000,21
LIN X+2733.5,Y+1461.5,Z+810.0,A+154.999,B+0.000,C+180.000,21
PTP X+2733.5,Y+1161.5,Z+810.0,A+129.999,B-0.000,C+180.000,21
END
```

ERASE
Turns off the points of the memory. The range can be specified.

TRACE
It draws all of the points recorded in memory defining a trajectóry (path).

LOC2PRG
This function is quite useful given that can transform files of locations <. CAR> in files of programs <. R3D> using the approach of space points. The converted file, of the program is ready to be edited, simulated or post-processed.

Functions related with cycle logic:

SELECT
Select a program - file <. R3D>.

RUN
It executes the program with the interpreter R3D. The line at the beginning can be specified (see the command LIST). The user is questioned if he/she wants to remove the objects loaded, if existent.

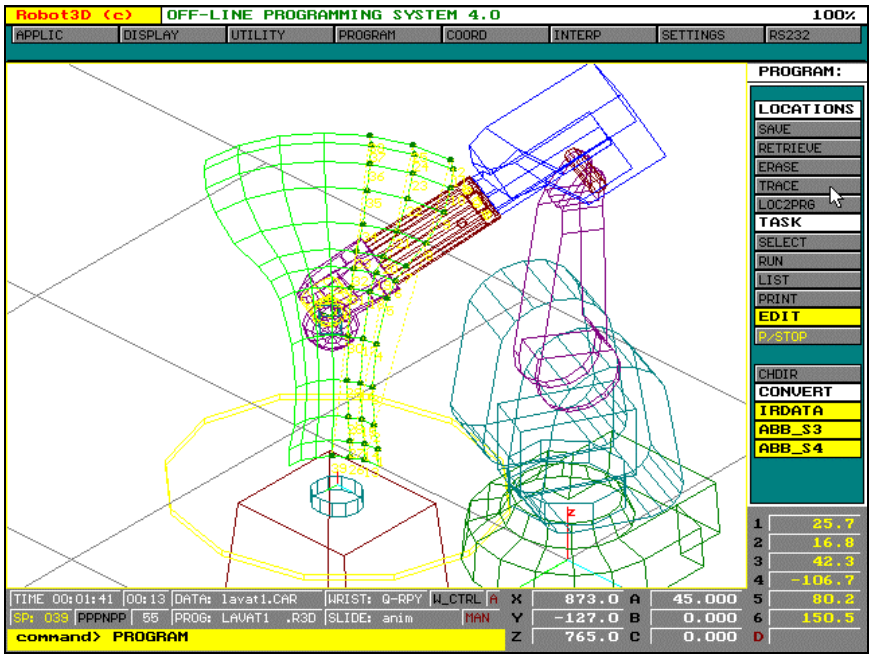


Fig. 4-5. Using the TRACE command.

NOTE: During the execution of a program (simulation) the keys of the ARROWS can be used, T, +, -, 0, 1, 2, 3, 4, 5, 6, 7, C, F, S, P for making functions of the "display":

- C - CLEAR
- F - FASTRUN (ON/OFF)
- T - PAN TCP
- S - STOP
- P - PAUSE

LIST
It shows the listing of the program selecting presenting the numbers of lines. As alternative other file type can be listed. For such press in the right key of the mouse after appearing the directóry.

EDIT
It invokes editor RODIT to edit a program file. If no file be specified creates a new file (right button of the mouse). The name of the editor to be used can be altered in the file ROBMOD.CFG. Para to obtain information on RODIT to consult the file RODIT.DOC in the directory ROBOT3D.

PRINT
Prints a listing of the program <. R3D> or of the locations <. CAR>.

CHDIR
It modifies the current directory for the search of programs and locations. The complete path should be indicated

IRDATA
It invokes external programs to convert the neutral language of and for IRDATA of SIEMENS (R3D_IRD.EXE and R3D_IRD.EXE).
Programs IRDATA should be transferred (download) for controller RCM with the software ROB.EXE (Computer Link) of SIEMENS.
For more information about characteristics and limitations of the language please read the file IRDATA.DOC (option).

ABB_S3
Converts programs of and for ARLA(s3) of ABB with utilitarian ARLA_R3D.EXE and R3D_ARLA.EXE (Option).
NOTE: Programs ARLA should be compiled for binary language and transferred with the software OLP3 of ABB ROBOTICS.
For more information about characteristics and limitations of the language please read the file ARLA.DOC (option).

ABB_S4
Converts programs of and for RAPID(s3) of ABB with utilitarian R3D_S4.EXE and S4_R3D.EXE (Option).
Programs S4 can be loaded directly in the controller of the robot.
For more information about characteristics and limitations of the language please read the file S4.DOC (option).

R3D - Robot3D Offline Programming Language

The group of commands, variables and operators are the following:

OPERATIONS
+ - / % * ^

A-W
Real variables (float). Converted to integre wever necessary.
P0-P99 parameters
NOTA: When movement instructions are executed the variables X,Y,Z,A,B,C e R,S,T,U,V,W are reserved.
DIM Varname
NOTE: A DIM Statment shall be used for each variable
Declared float variables names (max 30)

NEW
Clears variables of the memory

FOR <variable of control> = <initial value> **TO** < final value >.
...
NEXT
<initial value> and <final value> they can be expressions with parameters

```
PRINT "FOR LOOP EXAMPLE"
PRINT "(c) F.FERNANDES OUT/94"
FOR I=0 TO 5
  PRINT "I= ",I
NEXT
END
```

GOSUB <line_num>
...
<line_num>
...code
...
RETURN
NOTE: parameters are not supported!

```
PRINT "GOSUB EXAMPLE"
PRINT "(c) F.FERNANDES OUT/94"
GUSUB 50
END

50 PRINT "SUBROTINE 50"
GUSUB 60
RETURN

60 PRINT "SUBROTINE 60"
RETURN
```

JMP_SUB <line_num>,<ntimes>
...
<line_num>
  code
...
RETURN

This construction is equivalent to GOSUB but the call is repeated <ntimes>. NOTE Opposite to GOSUB this construction cannot be nested!

The following program passage is valid:

```
JMP_SUB 50,3
END

50 PRINT "SUB 50"
GOSUB 100
RETURN
```

```
100 I=I+1
RETURN
```

The following is invalid!:

```
JMP_SUB 50,3
END

50 PRINT "SUB 50"
JMP_SUB 100,2
RETURN

100 I=I+1
RETURN
```

GOTO <line_num>
NOTA: parameters are not suported!

```
PRINT "GOTO EXAMPLE"
PRINT "(c) F.FERNANDES OUT/94"
GOTO 20
PRINT "DONT CAME HERE"
20 PRINT "LABEL 20"
END
```

Attribution
<var> **=** <expression>
or <var> **=** <function>(expression)
NOTE: <var> can't be a parameter
<expression> it can be an expression with parameters

funtions:

- **SIN**(value) Value in degrees
- **COS**(value) "
- **TAN**(value) "
- **ASIN**(value) [-1,1] it returns an angle in degrees
- **ACOS**(value) [-1,1] it returns an angle in degrees
- **ATAN**(value)
- **ABS**(value)
- **EXP**(value)
- **LOG**(value)
- **SQR**(value)
- **INP_SIGNAL**(value) it returns the value of an input
- **TIMER**(value) it returns the value of a timer (timer 1 – accumulated time, timer 2 – time of the last move)

```
PRINT "EXAMPLE OF FUNCTION APLICATION "
A=500
B=10
C=A/B
E=35
R=3.1415/180
PRINT R
DELAY 3000
D=SIN(45)
PRINT "SIN 45 = ",D
END

****

PRINT "TEST OF STATUS OF INPUT"
INPUT "INPUT NUMBER> ",N
E=INP_SIGNAL(N)
IF E=0 THEN PRINT "INPUT",N,"OFF"
IF E=1 THEN PRINT "INPUT",N,"ON"
END
```

IF <expression> <operator> <expression>
THEN <instruction>
<expressão> it can be na expression with parameters

```
PRINT "IF EXAMPLE"
PRINT "(c) F.FERNANDES OUT/94"
I=5
J=10
IF I>J THEN PRINT "I>J"
IF I<J THEN PRINT "I<J"
IF I=J THEN PRINT "I=J"
END
```

OPERATORS: **>**,**<**,**=**

PRINT <argument list >

LPRINT < argument list >
Same as **PRINT** but output goes to the printer.

TX_RS232 < argument list >
Same as **PRINT** but output goes to Serial Port. It should be used utilitarian TINYCOMM or another program of communications in the remote computer. A cable type "null cable" should be used.

```
PRINT "RS232 COMUNICATIONS EXAMPLE"
```

```
PRINT "Cable should be connected and a communications program"
PRINT "must be running in remote computer..."
PRINT "USE Tinycomm.exe"
TX_RS232 "(c) F.FERNANDES 5/3/94"
ZOF 2
FOR I=0 TO 2
  PTP X+200+i,Y+300+i,Z+400,A+30,B+50,C+120
  TX_RS232 "ROBOT POSITION X=",X,"Y=",Y,"Z=",Z
  TX_RS232 "ORIENTATION A,B,C: ",A,B,C
NEXT
END
```

File manipulation

FPRINT <argument list>
Same as **PRINT** but output goes to PROGRAM.OUT. If the file is to be saved please record it with another name or it will be destroid in the next *RUN*. This shall be done in DOS (use *SHELL* - *UTILITIES* menu).

INPUT "<message>", <var>
NOTE: <var> it can be a parameter

```
PRINT "INPUT EXAMPLE"
PRINT "(c) F.FERNANDES OUT/94"
INPUT "HOW OLD ARE YOU? ",A
PRINT "YOU ANSWER ",A," YEARS"
END
```

END
End of the program.

NOTE: in the following text, expression refers to expressions that can contain parameters.

ROBOT FUNCTIONS

HOME
It puts the robot or positioner in a pre-defined position. In a simulation, this instruction, should only be used once before any movement as the cycle time is not calculated. It is useful to it after an instruction **GET_DATA** to clean any mistake of positioning of the robot.

CHG_TOOL "tool"
Changes the tool put in the wrist and it recovers the matrix of the tool from the file <ferramenta.TOF>.

```
PRINT "TOOL CHANGE EXAMPLE""
PRINT "(c) F.FERNANDES OUT/94"
CHG_TOOL "WELDING"
CHG_TOOL "PINCA"
CHG_TOOL "STANDART"
CHG_TOOL "GUN2"
END
```

LIN X<v1>,**Y**<v2>,**Z**<v3>,**A**<v4>,**B**<v5>,**C**<v6>,<code>
Lineal interpolation using coordinates. The values A, B and C depend on the system of orientation of the wrist.
<code> it is the code for the status of the axes related with the inverse kinematics.
Examples: 00111111=63 => PPPPPP; 00000001=1 => PNNNNN
If the grippers are active the two bits of the left are increased.
Example: 10-000000=128 the gripper 2 is ON.

LIN_SP <expression>
Lineal interpolation using points (locations) in memory.
NOTE: To have in attention that at present these instructions ignore the frames (the same for PTP_SP and CIR_SP). If it intends to use frames to use the format LIN X....

PTP X<v1>,**Y**<v2>,**Z**<v3>,**A**<v4>,**B**<v5>,**C**<v6>,<código>
Interpolation Point to Point sincronized using coordinates (all of the axes start and they stop at the same time - interpolation in the space of the joints). The values A, B and C depend on the system of orientation of the wrist.

PTP_SP <expression>
Interpolation sincornized Point to Point using points.

CIR X<v1>,**Y**<v2>,**Z**<v3>
X<v4>,**Y**<v5>,**Z**<v6>,**A**<v7>,**B**<v8>,**C**<v9>,<code>
Circular interpolation using coordinates. The values A, B, and C depend on the system of orientation of the wrist. The values v1 - v3 correspond to the auxiliary point and the values v4 - v9 to the final point.

CIR_SP <expression1>, <expression2>
circular interpolation: <expression1> gives the number of the auxiliary point, <expression2> gives the number of the final point.

NOTE: The circular interpolation is always ABSOLUTE. To see the instruction **PROG_REL**. .

ZOF <num>,**X**<v1>,**Y**<v2>,**Z**<v3>,**A**<v4>,**B**<v5>,**C**<v6>
User Frame. Correcting of the "zero-offset" number <num>. the function **ZOF** can be used to move and to turn points in the space 3D allowing to use the same program in different locations. The valid values are **ZOF** 0-9. The instruction **ZOF** 0 puts the robot in coordinated *WORLD*. The coordinates are optional. If no specified the values are recovered starting from the table <project.ZOF>.

```
PRINT "ZOF TEST"
PRINT "(c) F.FERNANDES OUT/94"
PRINT "ZOF 0"
ZOF 0
PRINT "THIS SHOWS ALL ZOOF's"
FOR i=0 TO 9
  ZOF i
NEXT
END
```

PPA R <v1>, **S** <v2>, **T** <v3>, **U** <v4>, **V** <v5>, **W** <v6>
Movement of the external axes or positioners. This instruction should appear in a program after the movement of the robot for the position to be define for post-processing.

NOTE: In movements using coordinates (and still in instructions **ZOF**, **OBJ_POS** and **OBJ_OFF**), the value of the corresponding variables (example: X, R) it is equaled to zero before the expression is evaluated.
For instance, an instruction of the type LIN x+10*i, Y+5,0,A+10,B+5*i,C+0 with i=5 is calculated like LIN X=50, Y=5, Z=0, A=10, B=10, C=0.
Obs. To see the instruction **PROG_REL** for exceptions

MOVE_JOINTS <v1>, <v2>, <v3>, <v4>, <v5>, <v6>
Movement of robot axis

REC_JOINTS_PAR <num>
It attributes the values of joints to parameters beginning with parameter **P**<num>

OP_GRP <expression> - valid values 1,2.
Opening of grippers.

CL_GRP < expression>.
Closing of grippers.

```
PRINT "GRIPPER EXAMPLE""
PRINT "(c) F.FERNANDES OUT/94"
CHG_TOOL "PINCA"
CL_GRP 1
OP_GRP 1
CHG_TOOL "WELDING"
CL_GRP 1
OP_GRP 1
END
```

PICK_OBJ <expression> (0..39)
Makes the connection of the defined object for <expression> to TCP of the robot (see *ATTAC_OBJ* in the menu *SETTINGS*).

FREE_OBJ <expression> (0..39)
It frees a linked object to TCP of the robot or the connection of an external axis.

LOAD_OBJ "<name_obj>"
It loads an object into the cell. The objects are numbered sequentially. The numbers of the objects can be seen in the screen of the initial menu.

OBJ_POS <num>,**X**<v1>,**Y**<v2>,**Z**<v3>,**A**<v4>,**B**<v5>,**C**<v6>
It puts an object in the work cell with origin in the location given by v1-v6.
Valid values for <num>: 0-39.
If the object is coupled its position is calculated by the system.

OBJ_OFF <num>,**X**<v1>,**Y**<v2>,**Z**<v3>,**A**<v4>,**B**<v5>,**C**<v6>
It puts an object in the cell with an offset given by v1-v6 relatively to the origin defined by the instruction **OBJ_POS**.
Valid values for <num>: 0-39.

ROBOT_LINK <num>
num< DOF(Degrees of freedom) +2
Attributes the localização of the robot link <num> to the variables X,Y,Z,A,B,C.

EXTERN_LINK <num>
num< DOF_EXT(Degrees of freedom) +2
Attributes the location of external axis link <num> to the variables X,Y,Z,A,B,C.

```
PRINT "FRAME COORDINATES OF ROBOT AND EXTERNAL LINKS"
PRINT "(c) F.FERNANDES OUT/94"
5 INPUT "ROBOT/EXTERNAL? [1/2] 3->QUIT: ",J
IF J=1 THEN GOSUB 10
IF J=2 THEN GOSUB 20
IF J=0 THEN GOTO 5
END

10 PRINT "ROBOT"
  INPUT "LINK> ",I
  ROBOT_LINK I
  PRINT "LINK ",I," X:",X," Y:",Y," Z:",Z,"A:",A," B:",B," C:",C
  DELAY 5000
RETURN


20 PRINT "EXTERNAL AXIS"
  INPUT "LINK> ",I
  EXTERN_LINK I
  PRINT "LINK ",I," X:",X," Y:",Y," Z:",Z,"A:",A," B:",B," C:",C
  DELAY 5000
RETURN
```

ATTACH_ROB_LINK <obj_num>,<link_num>
Connects the object <obj_num> to the robot link <link_num>.

ATTACH_EXT_LINK <obj_num>,<link_num>
Connects the object <obj_num> to the external link <link_num>.

ERASE_OBJS
Turn off all the objects of the memory

DELAY <n>
Pauses the execution of the program <n> miliseconds.

```
PRINT "DELAY EXAMPLE""
PRINT "(c) F.FERNANDES OUT/94"
PRINT "SUSPEND DURING 5 SECONDS"
DELAY 5000
```

```
PRINT "OK"
END
```

WAI_IL <n>
Waits that the input <n> it is LOW (0). valid Values 0-31.

WAI_IH <n>
Waits that the input <n> it is HIGH (1)

```
print "TEST PROGRAM FOR INSTRUCTION WAI_IH E WAI_IL"
print "(c) F.FERNANDES 1993"
print "WAITS UNTIL INPUT 14 GO ON"
wai_ih 14
print "WAITS UNTIL INPUT 2 GO LOW"
wai_il 2
end
```

S_O <n>
Set the output <n> in the state HIGH (1). Valid values 0-31.

RS_O <n>
Reset the output <n> to state LOW (0). Valid values 0-31.

ARC_ON <condition>
It starts the arc with parameters given by the <condition>.
Valid values 0-99. Not implemented.

ARC_OFF <condition>
Not implemented.

Attribution of speed

VEL_CPA <val> linear speed in mm/s
VEL_ALL <val> Speed for all axis in %
VEL_AXS <axis_num>,<val> Speed for axis <axis_num> in %
VEL_OV <val> Speed Factor (override) in %

Attribuition of accelerations

ACC_CPA <val> linear acceleration in mm/s2
ACC_ALL <val> Aceleration for all axis in %
ACC_AXS <eixo_num>,<val> Aceleration for axis <axis_num> in %

```
PRINT "VELOCITY AND ACCELERATION EXAMPLE"
PRINT "(c) F.FERNANDES OUT/94"
```

```
PRINT "VALUES ON THE ROBOT robot.VA FILE"
PRINT "VELOCITY FOR ALL AXIS 17%"
vel_all 17
PRINT "ACELERATION FOR ALL AXIS 24%"
acc_all 24
PRINT "LINEAR VELOCITY [mm/s]"
vel_cpa 2100
PRINT "LINEAR ACCERERATION [mm/s2]"
acc_cpa 5200
PRINT "GENERAL OVERRIDE [%]"
vel_ov 45
PRINT "VELOCITY FOR AXIS NUMBER 2 43%"
vel_axs 2,43
PRINT "ACCELERATION FOR AXIS NUMBER 3 23%"
acc_axs 3,23
END
```

WELD <condition> - valid values 0-99.
It attributes a welding condition for post-processing.
Not implemented.

$<meta_command>
Meta_command. Simulation command.

$**ZOOM_IN** <n> - Approach (n) repetitions
$ZOOM_OUT <n>
$PAN_UP <n>
$PAN_DOWN <n>
$PAN_RIGHT <n>
$PAN_LEFT <n>
$PAN_TCP <n>
$VIEWPORT <view>
$DRAW_ON - it allows to regenerate the screen
$DRAW_OFF – it inhibits the screen regeneration

```
PRINT "TESTE DE ZOOMS"
FOR I=0 TO 5
  $viewport I
NEXT
$viewport 0
$ZOOM_IN 4
$ZOOM_OUT 3
$PAN_RIGHT 4
$PAN_LEFT 3
$PAN_UP 1
$PAN_DOWN 3
$PAN_TCP
$ZOOM_IN 5
```

GET_DATA "file"

Retrieve locations from a <.DAT> or<.CAR> file. Same as *RETRIEVE* in MANUAL mode.
NOTE: to use instructions with space points (ex. **PTP_SP** 5) the locations should be in memory. If we didn't teach any point it should be used na **GET_DATA** instruction in the program.

BISIACH & CARRU users

The good way to get the correct PHI for the TAURUS SYSTEM in RCM is to use the techique of spacial points during programming.

Example:
PTP_SP... will give correct PHI value.
PTP X... will give na estimate of PHI for motions in XY plane.

CALL "filename"
Interrupts current program execution and starts a new program interpreter. Same as *SELECT* and *RUN* in MANUAL mode.

PROG_ABS
Program uses absolute coordinates (defect).

PROG_REL
Coordinates are relative to last position.
Example:

```
PROG_ABS
PTP X+500.1,Y-600,Z-100,A+0,B+30,C+180,30
PROG_REL
PTP X+0,Y+100,Z+0,A+0,B+35,C+180,30
```

The second location is absolute (500,-500,-100,0,35,180)
NOTE: Orientation is always ABSOLUTE!

Utilizadores SIEMENS RCM:

The way to write the program shall guaranty that the type of movement is correct (Incremental or Absolute)

For instance:

```
PROG_REL
50 PRINT "SP50"
PROG_REL
GOSUB 30
LIN X+30,Y+0,Z+0,A+90,B+0,C+180,21
RETURN
```

```
PROG_ABS
30 PRINT "SP30"
PROG_ABS
PRINT "WELDING"
RETURN
```

As subroutine 30 is absolute the instruction **LIN** after **GOSUB 30** doesn't have the expected result as the motion is now absolute. To correct the instruction **PROG_REL** should be placed after **GOSUB 30**:

```
GOSUB 30
-> prog_rel
```

Programming Examples in **R3D**

Example1

```
PRINT "GRIPPER & TOOL CHANGE DEMO"
PTP_SP 1
CHG_TOOL "pinca"
OP_GRP 1
FOR I=0 TO 1
  LIN_SP I+2
NEXT
CL_GRP 1
HOME
PTP_SP 2
CHG_TOOL "welding"
INPUT "Point to move> ",J
IF J<3 THEN GOSUB 100
END

100
  PRINT "Point ",J
  LIN_SP J
  LIN_SP 1
  CIR_SP 2,3
RETURN
```

Example 2:

```
PRINT "DEMO1.PRG"
DELAY 2000
PRINT "ARC WELDING EXAMPLE"
PRINT "To run with KUKA_P3 project"

TX_RS232 "COMMS TEST"
PRINT "FF, 8/9/1993"
CHG_TOOL "welding"
GET_DATA "kuka_p3"
HOME
$VIEWPORT 0
```

```
VEL_ALL 100
VEL_CPA 1000
ACC_CPA 9800
INPUT "PRESS 1 TO AVOID CILINDER WELDING > ",k
IF k=1 THEN GOTO 5
$ZOOM_OUT 1
GOSUB 10
GOSUB 30
5 INPUT "PRESS 2 TO AVOID BOX WELDING > ",k
IF k=2 THEN GOTO 8
PRINT "PERMISSION TO ROTATE TABLE (I10)"
WAI_IL 10
GOSUB 20
GOSUB 40
CALL "signals"
8 PRINT "END OF PROGRAM"
END

10 PRINT "TABLE ROTATION TO FRONT"
FOR i=0 TO 6
  PPA R+90-30*i,S+0,T+0
NEXT
RETURN

20 PRINT "TABLE ROTATION TO REAR"
FOR i=0 TO 6
  PPA R-90+30*i,S+0,T+0
NEXT
RETURN

30 PRINT "CILINDER WELDING"
PTP_SP 2
$PAN_TCP
$ZOOM_IN 2
PRINT "START WELDING"
OP_GRP 1
CIR_SP 3,4
CIR_SP 5,2
CL_GRP 1
LIN_SP 6
$ZOOM_OUT 2
RETURN

40 PRINT "FRAME TO THE BOX"
ZOF 1,X-350.000,Y-1090.000,Z+1040.000,A+0,B+0,C+0
$VIEWPORT 3
$ZOOM_IN 2
GOSUB 50
$VIEWPORT 0
$ZOOM_OUT 2
PRINT "BOX ROTATION"
PPA R+90,S+0,T+30
PRINT "FRAME RESTORE TO ROBOT BASE"
ZOF 0
PTP X-760.000,Y-840.000,Z+1400.000,A+140.000,B-10.000,C-140.000,63
PRINT "NEW FRAME BOX"
ZOF 2,X-305.000,Y-1080.000,Z+1040.000,A+30,B+0,C+0
GOSUB 50
```

```
ZOF 0
PTP X-760.000,Y-840.000,Z+1400.000,A+140.000,B-10.000,C-140.000,63
RETURN

50 PRINT "BOX WELDING SEQUENCE"
PTP X+0,Y+0,Z+0,A+30,B+0.000,C+180.000,63
PRINT "START WELDING"
S_O 10
OP_GRP 1
LIN X+200,Y+0,Z+0,A-45.000,B+0.000,C+180.000,63
GOSUB 100
LIN X+200,Y+300,Z+0,A+45.000,B+0.000,C+180.000,55
GOSUB 100
LIN X+0,Y+300,Z+0,A+135.000,B+0.000,C+180.000,23
GOSUB 100
LIN X+0,Y+0,Z+0,A-135.000,B-0.000,C+180.000,31
GOSUB 100
CL_GRP 1
RETURN

100 TX_RS232 "POSITION",X,Y,Z,A,B,C
RETURN
```

Example 3:

```
PRINT "MANIPULATION OF OBJECTS"
PRINT "FF, MARCH 94"
PRINT "To run with M6100 project"
CHG_TOOL "pinca"
GET_DATA "mansolid"
HOME
INPUT "PARTS TO MANIPULATE >",K
IF K>6 THEN K=6
PTP_SP 6
FOR I=0 TO K-1
  $ZOOM_IN 1
  IF I=0 THEN LOAD_OBJ "PECA0"
  IF I=1 THEN LOAD_OBJ "PECA1"
  IF I=2 THEN LOAD_OBJ "PECA2"
  IF I=3 THEN LOAD_OBJ "PECA3"
  IF I=4 THEN LOAD_OBJ "PECA4"
  IF I=5 THEN LOAD_OBJ "PECA5"
  OBJ_POS I,1150.001,-940.004,919.993,0.0,0.0,180.0
  PTP_SP 20
  OP_GRP 1
  LIN_SP 21
  PICK_OBJ I
  CL_GRP 1
  LIN_SP 20
  PTP_SP 22
  J=I%2
  PRINT J
  DELAY 2000
  IF J=0 THEN LIN X+1465.9-100*i,Y-239.1,Z+920.0,
A+30.0,B+0.0,C+180.0,63
  IF J=1 THEN LIN X+1465.9-100*i,Y-239.1-200,Z+920.000,
```

```
A+30.0,B+0.0,C+180.0,63
  FREE_OBJ I
  OP_GRP 1
  PTP_SP 22
NEXT
  PTP_SP 6
END
```

Example 4 (BISIACH & CARRU users):

```
CHG_TOOL "GUN8"
PRINT "Taurus Robot "
PRINT "Uses Absolute & Incremental Programming"
PTP X+2836.6,Y+2626.4,Z+810.0,A+179.999,B+0.000,C+180.000,21
PTP X+2836.6,Y+2626.4,Z+810.0,A+179.999,B+0.000,C+180.000,21
PPA R+1900.0
GOSUB 85
END


PROG_ABS
85 PRINT "SP85"
PROG_ABS
PTP X+2733.5,Y+1761.5,Z+810.0,A+154.999,B+0.000,C+180.000,21
PTP X+2733.5,Y+1461.5,Z+810.0,A+154.999,B+0.000,C+180.000,21
PTP X+2733.5,Y+1161.5,Z+810.0,A+129.999,B+0.000,C+180.000,21
PTP X+3133.5,Y+1061.5,Z+810.0,A+129.999,B+0.000,C+179.999,21
PTP X+3133.5,Y+1061.5,Z+999.0,A+129.999,B+0.000,C+179.999,21
PTP X+3013.5,Y+761.5,Z+999.0,A+129.999,B+0.000,C+179.999,21
GOSUB 44
GOSUB 30
JMP_SUB 58,5
GOSUB 43
PTP X+3013.5,Y+1081.5,Z+999.0,A+129.999,B+0.000,C+179.998,21
PTP X+3123.5,Y+1081.5,Z+999.0,A+129.998,B+0.000,C+179.998,21
PTP X+3123.5,Y+1081.5,Z+810.0,A+129.998,B+0.000,C+179.998,21
PTP X+2403.5,Y+1081.5,Z+810.0,A+104.998,B-0.001,C+179.997,21
PTP X+2403.5,Y+801.5,Z+999.0,A+104.998,B+0.000,C+179.997,21
PTP X+2483.5,Y+771.5,Z+999.0,A+104.997,B+0.000,C+179.997,21
GOSUB 44
GOSUB 30
JMP_SUB 59,5
GOSUB 43
PTP X+2333.5,Y+771.5,Z+999.0,A+89.997,B+0.000,C+179.996,21
PTP X+2333.5,Y+771.5,Z+810.0,A+89.997,B+0.000,C+179.996,21
PTP X+2333.5,Y+771.5,Z+810.0,A+89.997,B+0.000,C+179.996,21
PTP X+2333.5,Y+771.5,Z+810.0,A+89.997,B+0.000,C+179.996,21
PPA R+1200.0
PTP X+2333.5,Y+191.6,Z+999.0,A+89.997,B+0.000,C+179.995,21
PTP X+2193.5,Y+191.6,Z+999.0,A+89.997,B+0.000,C+179.995,21
GOSUB 44
GOSUB 30
PTP X+2193.5,Y+221.6,Z+999.0,A+89.997,B+0.000,C+179.995,21
GOSUB 30
PTP X+2193.5,Y+391.6,Z+999.0,A+89.997,B+0.000,C+179.995,21
GOSUB 30
```

```
JMP_SUB 54,15
GOSUB 43
PTP X+2193.5,Y+961.6,Z+999.0,A+89.997,B+0.000,C+179.995,21
PTP X+2193.5,Y+961.6,Z+810.0,A+89.997,B+0.000,C+179.994,21
PTP X+1653.5,Y+961.6,Z+810.0,A+79.996,B+0.000,C+179.994,21
PTP X+1653.5,Y+611.6,Z+890.0,A+79.996,B+0.000,C+179.993,21
PTP X+1653.5,Y+171.6,Z+999.0,A+79.995,B+0.000,C+179.993,21
PTP X+1683.5,Y+181.6,Z+999.0,A+79.995,B+0.000,C+179.993,21
GOSUB 44
GOSUB 30
PTP X+1683.5,Y+211.6,Z+999.0,A+79.995,B+0.000,C+179.993,21
GOSUB 30
PTP X+1683.5,Y+401.6,Z+999.0,A+79.995,B+0.000,C+179.992,21
GOSUB 30
JMP_SUB 70,15
PTP X+1683.5,Y+1091.6,Z+999.0,A+79.994,B+0.001,C+179.992,21
GOSUB 30
PTP X+1683.5,Y+1201.6,Z+999.0,A+79.994,B+0.001,C+179.992,21
GOSUB 30
GOSUB 43
PTP X+1653.5,Y+1001.6,Z+999.0,A+79.994,B+0.001,C+179.992,21
PTP X+1653.5,Y+1001.6,Z+810.0,A+79.994,B+0.001,C+179.992,21
PTP X+2220.7,Y+1161.2,Z+810.0,A+109.994,B-0.003,C+179.992,21
PTP X+2944.6,Y+2474.8,Z+810.0,A+179.994,B-0.009,C+179.999,21
RETURN

PROG_ABS
30 PRINT "SP30"
OP_GRP 1
CL_GRP 1
PROG_ABS
RETURN

PROG_ABS
43 PRINT "SP43"
OP_GRP 2
PROG_ABS
RETURN

PROG_ABS
44 PRINT "SP44"
CL_GRP 2
PROG_ABS
RETURN

PROG_REL
54 PRINT "SP54"
PROG_REL
LIN X+0.0,Y+30.0,Z+0.0,A+89.999,B-0.009,C+179.999,21
GOSUB 30
RETURN

PROG_REL
58 PRINT "SP58"
PROG_REL
LIN X+0.0,Y+30.0,Z+0.0,A+129.994,B-0.009,C+179.999,21
GOSUB 30
RETURN
```

```
PROG_REL
59 PRINT "SP59"
PROG_REL
LIN X+0.0,Y+30.0,Z+0.0,A+104.994,B-0.009,C+179.999,21
GOSUB 30
RETURN

PROG_REL
70 PRINT "SP70"
PROG_REL
LIN X+0.0,Y+30.0,Z+0.0,A+79.999,B-0.009,C+179.999,21
GOSUB 30
RETURN
```

Example 5 calculation of ZOF using 3 points and parameters

```
PRINT "CALCULATIN OF R MATRIX FROM 3 POINTS"
PRINT "PLESE RETRIEVE THE POINTS FROM 3ZOF.CAR PRIMEIRO"
GET_DATA "3ZOF"
PRINT "p1(x1,y1,z1) origem"
PRINT "p2(x2,y2,z2) -> p2-p1 direccao de x"
PRINT "p3(x3,y3,z3) -> ponto do plano xy entre os eixos x e y"
PRINT "A matrix e guardada no ZOF 9"
PRINT "FF 14/7/95"

P90=180/3.1415
P91=3.1415/180
PRINT "ATRIBUITION OF PARAMETERS"

PTP_SP 1
P1=X
P2=Y
P3=Z

PTP_SP 2
P4=X
P5=Y
P6=Z

PTP_SP 3
P7=X
P8=Y
P9=Z

PRINT "vector u(P10,P11,P12)"

 P10=P4-P1;
 P11=P5-P2;
 P12=P6-P3;

PRINT "vector a(P13,P14,P15)"

 P13=P7-P1;
 P14=P8-P2;
 P15=P9-P3;
```

```
PRINT "calculatin of vector w(P16,P17,P18)=uxa"

  P16=P15*P11-P14*P12;
  P17=P13*P12-P15*P10;
  P18=P14*P10-P13*P11;

PRINT "calculation of vector v(P19,P20,P21)=w*u "

  P19=P12*P17-P11*P18;
  P20=P10*P18-P12*P16;
  P21=P11*P16-P10*P17;

PRINT "normalizat of vectores u,v,w"

  P22=SQR(P10*P10+P11*P11+P12*P12);
  P23=SQR(P19*P19+P20*P20+P21*P21);
  P24=SQR(P16*P16+P17*P17+P18*P18);

PRINT "t11,t12,t13"

  P25=P10/P22;
  P26=P11/P22;
  P27=P12/P22;

PRINT "t21,t22,t23"

  P28=P19/P23;
  P29=P20/P23;
  P30=P21/P23;

PRINT "t31,t32,t33"

  P31=P16/P24;
  P32=P17/P24;
  P33=P18/P24;

  X=P1;
  Y=P2;
  Z=P3;

  P40=0.5*(P25*P25+P26*P26+P30*P30+P33*P33)
  P41=SQR(P40)
  P42=-P27/P41
  P43=ATAN(P42)
  B=P43
  IF B=0 THEN GOTO 50
  IF B=90 THEN GOTO 60
  IF B=-90 THEN GOTO 70
  GOTO 80
50
  P50=P26/P25
  P51=ATAN(P50)
  A=P51
  P53=P30/P33
  P54=ATAN(P53)
  C=P54
GOTO 100
```

```
60
  A=0
  P60=P28-P32
  P61=P29+P31
  P62=P60/P61
  P63=ATAN(P62)
  C=P63
GOTO 100

70
  A=0
  P70=-P28-P32
  P71=P29-P31
  P72=P70/P71
  P73=ATAN(P72)
  C=P73
GOTO 100

80
  P80=P30*P31-P33*P28
  P81=P33*P29-P30*P32
  P82=P80/P81
  P83=ATAN(P82)
  A=P83
  P85=P26*P31-P25*P32
  P86=P25*P29-P26*P28
  P87=P85/P86
  P88=ATAN(P87)
  C=P88


100
  P92=X
  P93=Y
  P94=Z
  P95=A
  P96=B
  P97=C
  PRINT "t31,t32,t33"
  PRINT "attribution to ZOF 9"
  ZOF 9,P92,P93,P94,P95,P96,P97
  DELAY
  2000
END
```

COORD Menu

The *COORD* menu provides an interface similar to a robot console (Teach pendant). The operation of this menu corresponds to MANUAL mode in a robot.

Motion can be done in incremental or absolute coordinate values. *JOINT, WORLD,TOOL, USER* (9 ZOF frames) and *EXTERNAL* coordinate systems are available.

The motion depend of the system choosed for orientation (RPY, ABC, etc).
When displaying the orientation in Quaternius the A,B,C,D represent quaternius Q1,Q2,Q3,Q4.

The menu provides function for changing the robot position, velocities and aceleration settings. Positions can be recorded and traced.
HOME, welding, gripper and I/O functions are available.

The TOP menu provides display functions to avoid to change to the *DISPLAY* menu. This functions can only be acessed with the mouse.

Funtions:

JOINT
Selects joint motion (robot axis). Generalized coordinates

WORLD
Selects robot base coordinate frame. Also called cartesian coordinates (X, Y, Z, A, B, C).

TOOL
Makes the actual reference frame to be located on the TCP. Actual values of coordinates are setting at 0. The key F gives acess to input the TOF number without the mouse. In version 5.0 each location has an associated tool frame.

USER
User coordinate frame (Station frame). Related with ZOF (Zero offset). The ZOF number can be selected clicking the box at the right. To input values to the ZOF see the *SETTINGS* menu. The key Z gives acess to input the ZOF number without the mouse.

NOTE: After program RUN the system returns to WORLD coordinates. To use ZOF it should be setted inside a program

IMPORTANT: **ZOF** instructions do not affect **LIN_SP**, **PTP_SP** and **CIR_SP** instructions. Only movements with coordinates are afected by ZOF instructions.
In version 5.0 the **ZOF** (user frame is recorded at each point).

EXTERNAL
Selects positioner motion. Positioner axis values ares displayed.

1/X, 2/Y, 3/Z, 4/A, 5/B, 6/C
Depending of the coordinate system selects articulated or cartesian axes. To move use + & - keys and STEP to change the angular or linear increment.

+, -
Increment position/ orientation.

STP - H,L,S
Changes the increment value for position (linear or angular).
High: 1000 mm & 30 Deg
Medium: 100 mm & 5 Deg
Small: 5 mm & 1 Deg
Default is Medium

HOME
Go to the defined home position. To change HOME position see the *SETTINGS* menu.

TEACH
Reccord point <n> (MAX 500 points).
The point reccorded has the information concerning robot and external axis. Please note that the robot motion is coupled with external axis. The motion atributes are asked from the user. In teaching the status of the grippers and the solution for the arm are reccorded.
If *ARC* or *WEAVE* is activated the condition atribute will be asked.
Use *SAVE* on *PROGRAM* menu to reccord the points in a file <name.COR>.

POINT
The robot goes to point <n> (max 500 points).

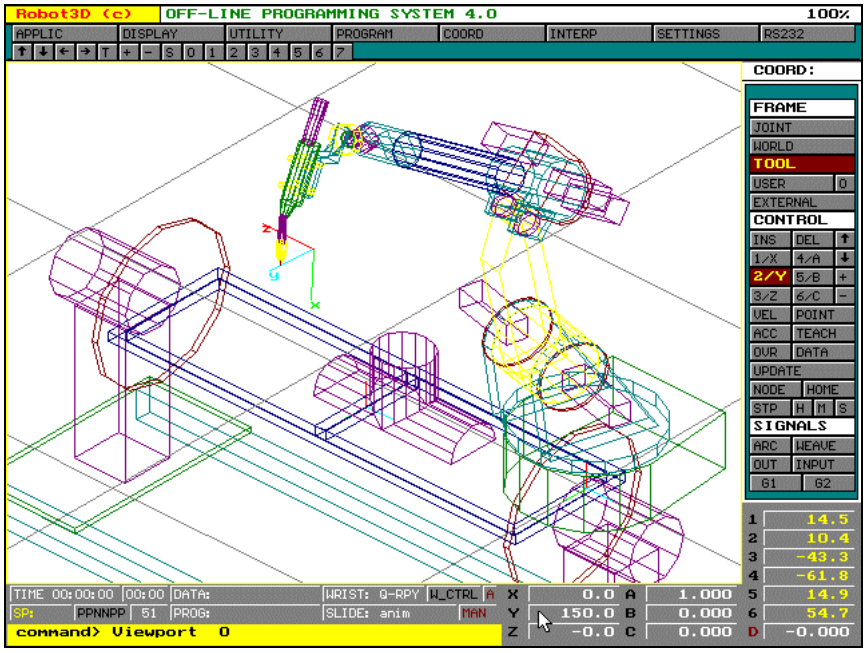


Fig. 4-6. Motion in TOOL coordinates of 150 mm em Y.

UPDATE
This function is used to simplify the changing of tool keeping the same location.
First select a point, then change tool in menu *SETTINGS* and then use *UPDATE*. The robot will be positionated in the same location. Use *TEACH* if the new position is to be reccorded.

DATA
Input of absolute coordinates to move to. Prompt values depend of the coordinate system selected.

VEL
Changes velocities.

- CPA – Linear Velocity [mm/s]
- ALL – Velocity for all axis [%]
- AXS – Velocity for one axis [%]
- AAA – Velocity for all external axis (not implem.) [%]
- AAX – Velocity for one external axis (not implem.) [%]
- OV – Velocity Override [%]

ACC
Changes accelerations.

- CPA – Linear Acceleration [mm/s2]
- ALL – Acceleration for all axis [%]

- AXS – Acceleration for one axis [%]

INPUT
Digital Input simulation

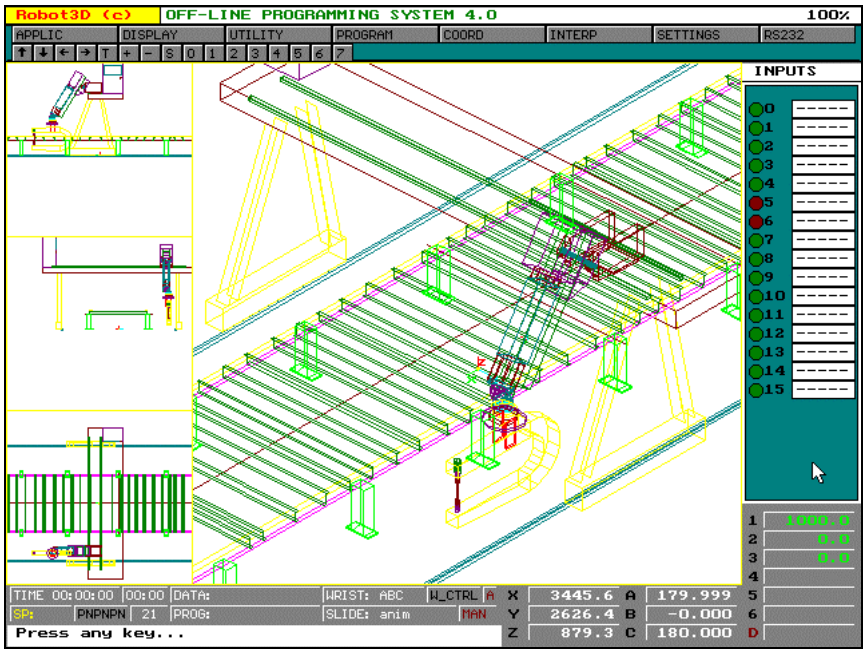


Fig. 4-7. Motion of 1000 mm in EXTERNAL Coordinates in axis 1 and simulation of INPUTS.

OUT
Simulation of Digital outputs.

NODE
Display the coordinates of a node in the cell, robot, positioner or object (*NODE* option should be active in *DISPLAY* menu). The user can decide to try to attend this position with the robot. Node shall be called from World or User 0 coordinates.

OVR
Changes general override.

G1 and G2
Dedicated Outputs- Grippers 1 and 2.

INS and DEL
Reorder points by insertion or delection (Not implemented).

UP and DOWN arrows
Point trace

ARC, WEAVE
Welding functions (Not fully implemented).

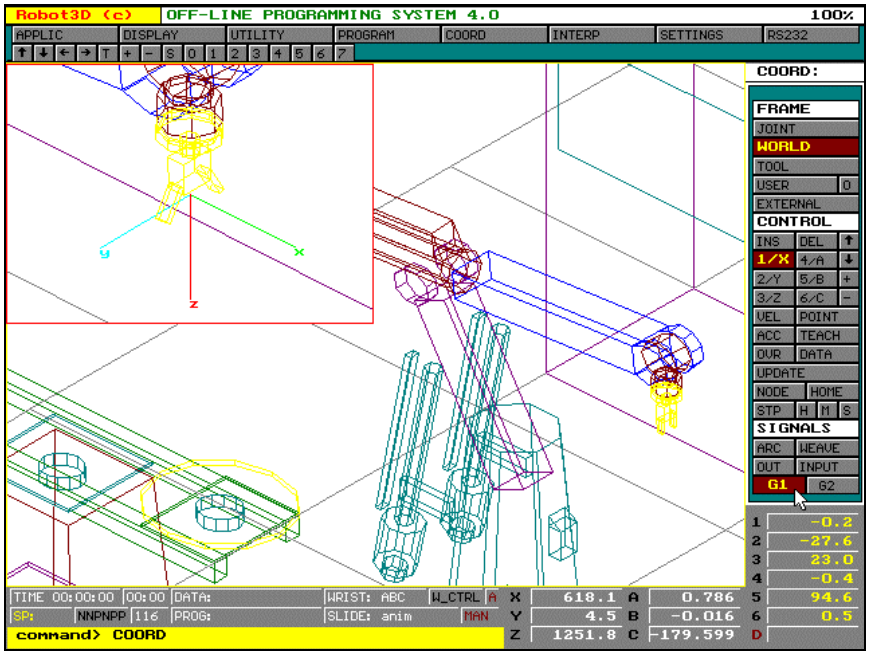


Fig. 4-8. Use of GRIPPERS.

INTERP Menu

The interpolation menu provides interpolation functions in 3D space to test movements outside a program run.

Interpolation times can be calculated with linear, circular and PTP motion. The cycle times are calculated on the basis of trapezoidal velocity profile.

The number of nodes and refreshing time can be changed. Interpolation points and slides can be reccorded.

The functions provided are:

INI, AUX, END
Reccord of points for interpolation. In automatic mode these points are recorded dynamically.

POINT, TEACH
Same as *COORD* menu.

TIMER
Freezes total timer.

RESET
Reset point data or timers.

PTP, LIN, CIR
Start motion in the interpolation mode. In LIN and CIR movements orientation interpolation occurs with the smallest angle. The auxiliary axis are sincronized with the robot axis.

PTP Point to point motion. All axis start and stop at same time. The system calculates angle diferences between initial and final points and moves the axis this diferences using a trapezoidal velocity profile relative to each axis.
LIN Linear motion. The system calculates a linear trajectory betwenn initial and end points. Along the trajectory a trapezoidal velocity profile is follow. The Orientation is interpolated between initial and end points.
CIR The system performs a 3D circular arc between initial and final points passing trough the auxiliary point. The orientation of the auxiliary point is irrelevant.

DATA
Toggle function to switch between interpolation information repport on not.

DELAY
Use this function to change the pause time between refresh of the screen in RUN mode.

FILE
Reccords interpolation data in a file. This function allows to view the motion graphs in the menu *UTILITY*.

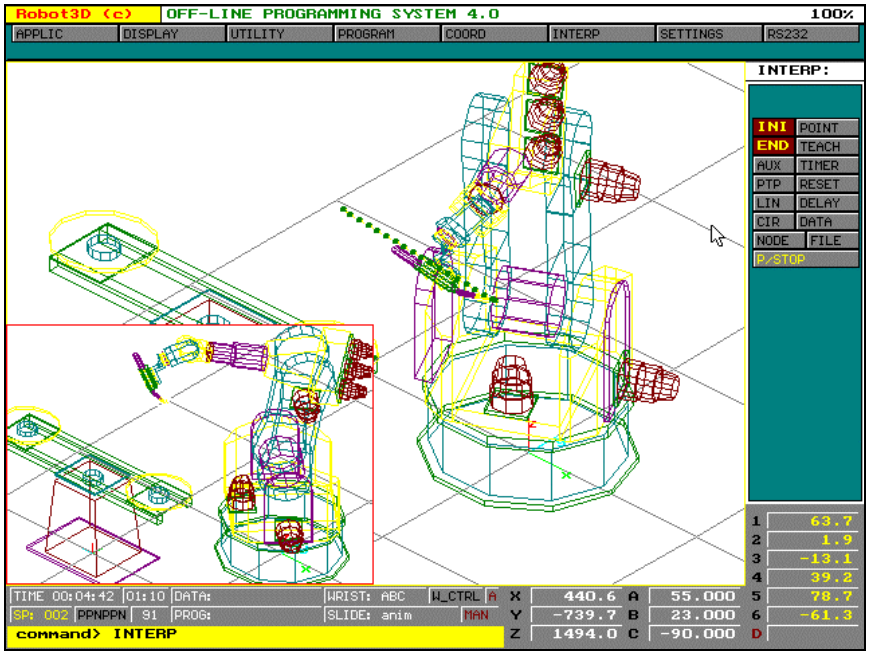


Fig. 4-9. Linear interpolatuion with 20 nodes.

FASTRUN
Changes the delay and interpolation points to increase running speed of a program. Useful in incremental programs. Can be setted during the running of a program by pressing the F key. This function is particulary useful in sopt welding aplications with bit amounts of points near each other.

NODE
Change the number of interpolation points (default 6, *FASTRUN* active 2).

EDGE
Signals the edges and vertex of a object. Change with <>. The robot can start a motion to a edge if under reach. To see how an object is loaded and positioned in the cell see the *UTILITY* Menu.

P/STOP
Is not actually a command. Used to stop or pause the interpolation.

SETTINGS Menu

The *SETTINGS* menu allows to modifi the configuration of the system.

The functions provided are:

LIMITS
Toggles mechanical limits checking On/Off. Use with precaution because is possible that user programs could not run in the real robot.

MATH_SOL
Changes the solution of inverse kinematics transformation. The user can input up to six parameters that affect the solution. See the inverse transformation file RTRANS.C (to be review).

ZOF FRAME
Changes the Zero Offset (Offset of *USER* frame relative to the *WORLD* frame). Sixteen ZOF frames can be used. ZOF 0 corresponds to the world frame or base frame. (If the robot is in *TOOL* coordinates the ZOF is teached, otherwise the user is prompted to input ZOF data).
NOTE: To use this function a ZOF must be active. See the *COORD* menu.

TOF FRAME
Changes the Tool Offset (Offset of the tool relative to the WRIST frame). The Wrist frame is connected to the last link of the robot and can be see if the <standart> tool is loaded.

Utilizadores RCM:
The TOF frame is defined by 6 parameters (X,Y,Z,Rz,Rx,Ry). Is necessary to find these parameters based on the L,H,T from RCM. (A conversion utility is supplied with the full package).

DH-PARAM
Changes Denavit-Hartenberg parameters on_line. Useful to dimensioning of robot arms. See robotics literature to see meaning of this parameters.

ORI_SYST
Changes the wrist system (e.g. RPY, EULER, RYR) on-line. Don't mix different wrist systems in the programs!.
Use ABC for RCM robots.
If the wrist_system RPY-Q (Quaternius ABB) is used the Quaternius format is used in orientation display (ABCD=Q1,Q2,Q3,Q4) and in DATA command. Robot jog is made in RPY system.
NOTE: As many matrix calculations are based in the RPY system is possible to have some misfunction in other orientation systems. (To be corrected sooner)

HOME
Changes the home position to the actual position of the robot.

TOOL
Changes the tool and corresponding TOF on-line. Mounts the tool.

POSIT
Togles between positionetr/external axis display on/off.

SLIDE_FILE
Create/validate a "slide_file" to animation

CONTROL
Type of data file retrieved (CAR or DAT). JOINT control is used with robots with unknown coordinate transformation. Don't use unless necessary!

ATTACH
Attach the robot to the positioner end frame. The robot moves as the external axis moves. The robot coordinates remain the same. Don't change unless necessary as is recorded in the project.

MARK_LOC
Marks the locations of the points during a program run in automatic mode. The point data can be later reccorded. If the teached points exceed MAXPOINTS the system prompts automaticaly to save the data points. (Max 300 points in each file)

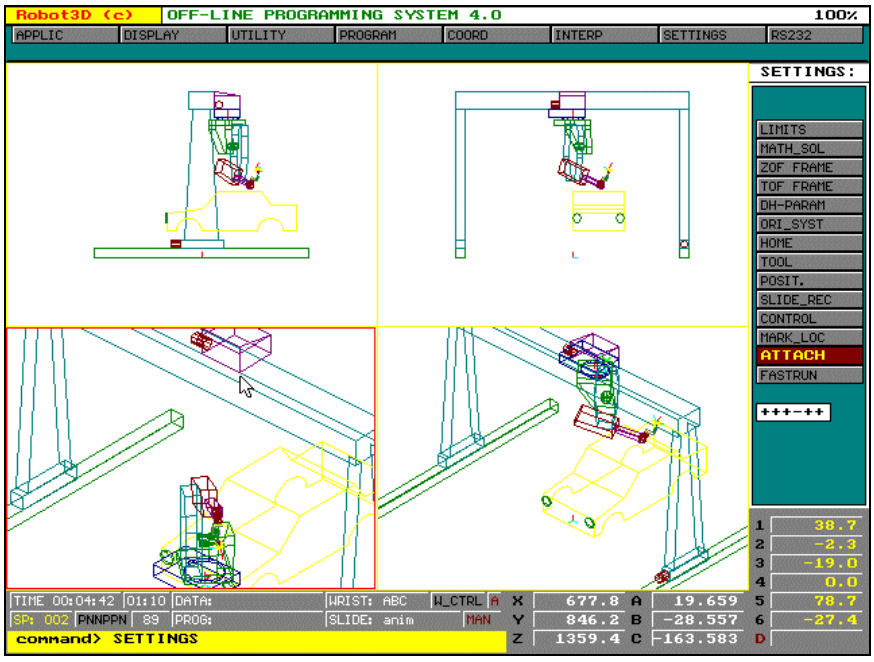


Fig. 4-10. Use of ATTACH command.

FASTRUN
Speeds up the simulation reducing the nodes in the trajectory.

RS232 menu

This menu is provided as a mean of exchanging data between Robot3D and an host computer. Use TINYCOMM on the remote computer and a NULL serial cable to test this functions.
A modem can be used to make communication at long distance:
NOTE: Don't use if a host computer is not connected!

The function are:

COM_LINK
Enable communications. Initialization.

SETTINGS
RS232 Serial settings. Is reccorded in ROBMOD.CFG

PHONE_NO
Phone number to contact trough modem.

MAKE_CALL
Place a call to host computer.

ANSWER
Answer mode

HANG_UP
Suspend communication.

LOG_INPUT
Write commands to a log file.

SEND_FILE
Send an ASCII file to remote computer.

TX_POS
Transmit position values X, Y, Z, A, B, C.

RX_POS
Receive the position with values X, Y, Z, A, B, C. The robot goes to that position.

NOTE: The instruction **TX_R232** <argument_list > is used to stablish communications in automaticmode. See *PROGRAM menu*.

Leaving the Program

To go out please press Q ou click with the mause in *QUIT* in starting screen.