Using ROBWIN v7.1.14 with ROBOFORTH II v13x up

RobWin © Sands Technology International; This is a Windows based application which runs in the computer. It communicates with the robot controller and also saves robot software on disk. It keeps an identical copy of the controller data in memory while you use it.
Roboforth II © D. N. Sands. This is a complete system which resides in the robot controller. It is based on the FORTH control language.
This manual concerns the project manager, RobWin. For in depth explanations of RoboForth see other manuals.

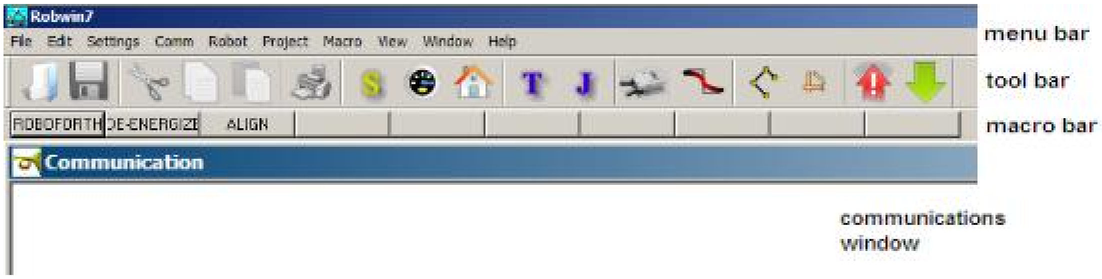
Note: Only type or click what is underlined. When you've typed it press the enter key.
Programming the robot requires communication with the robot controller. When ROBWIN is running there is a communication window. You can type things in this window and they are sent directly to the controller i.e. this window is like a terminal. ROBWIN also sends commands to the controller, most of which appear in the communication window. ROBWIN also has the ability to download data to the controller and upload data from it, and can save or reload data from disk.

Install

Run robwin7.1.14.msi (or higher) to install. We suggest you change the destination folder to your robot folder. Then copy the cfg and mac files into the robot directory. Create an icon on desktop if you wish.

Launch

Double-click on the ROBWIN icon.
Along the top are the menu buttons, below that is the robot tool bar and below that is the macro bar (see later) and below that appears the communication window. All commands typed in the communications window are sent to the controller and the controller's response appears there. Clicking a button and a lot of other functions generally just insert a command into the communications window to save you from typing it.



Load/check settings: As soon as Robwin 7 loads you should then load the settings file. On the menu bar click settings, open file. If you have an R12 or R17 click on R12R17.CFG; if an R19 click on R19.CFG. For a 6-axis R12 click on r12-6.cfg. Axis names may be manually set using settings.
Note that if you have been using RobWin2 v6.x the cfg files are not compatible. You will need to save your old ones and create new ones.
caps lock – All ROBOFORTH commands are in upper case, ensure CAPS LOCK is on.
Switch the key switch to cold start and switch on the robot controller.
In the communications window (the console) you should see a herald which includes the words 'cold start' and a chevron prompt **>**.
Type ROBOFORTH into the console. The response should be OK and a new line with a chevron prompt**>**
Note: When you type anything in the console all buttons are grayed out. If you back off what you typed they will still be grayed. Press **esc** if in doubt.

RobWin 7 functions

Tool bar:

The start button S is the same as typing START into the communications window. This energizes motors, zeros all position counts, sets default values for SPEED etc.

The Calibrate button types CALIBRATE Robot moves to the calibrate position and calibrates all axes.

The Home button is the same as typing HOME Robot moves to the HOME position where all motor counts are zero.

T invokes the teach pad in joint mode.

J invokes the teach pad in Cartesian mode.

operates the gripper. Click to close the gripper, click again to open. (“toggles” the gripper)

Sends the command SMOOTH to the controller. This sets CONTINUOUS mode and also ADJUST to reduce speed so the route can be RUN . (see RoboForth manual, section 7.2.6)

Sets JOINT mode

Sets CARTESIAN mode

Uploads data from controller memory to Robwin project

Downloads or reloads the text from the ED2 edit window. Actually starts by reloading the ED1 file which is a list of place and route names so they get loaded before any text that uses them.

Drop down menus:

File

Do not use file when you want to write a complete robot program. For that use project.
Open – opens a text file for editing. You can subsequently download that file into the controller with
Download current – downloads the text in the current window to the controller.
Open project.ed2 – if you have a project open and have closed the ed2 window this will re-open it.
Close, save, save-as – normal Windows functions
Download file – downloads a text file directly to the controller without viewing or editing
Load binary – loads a binary image from disk to controller memory. You need to specify
Bank (0 or 1). Basically bank 0 is where RoboForth, signature files etc. are kept and bank 1 is just for data such as routes or places.
Start address in hex
Length of file in hex
Name of file (usually with .RAM extension)
Save binary – copies a memory image from controller to
Same information as Load binary.

Edit

These functions are the same as any text editor, along with the legacy ctrl-C ctrl-V ctrl-X ctrl-Z ctrl-Y etc.
Note: you can not use editing keys in the communications window.

Settings

The settings files end in .CFG. Settings files from earlier versions of RobWin are not compatible.
When you launch RobWin for the first time immediately load a settings file.
Thereafter when you launch RobWin the last settings file used will be automatically loaded.
These files have been created by ST using the configuration functions.
New – create new settings file.
Open file – open an existing settings file as above.
Close, Save, Save-as – save settings file as per Windows convention.
Configuration -

Simulate – if you have no controller connected you can check this box for simulate mode.

Hide mode – when a text file is downloaded the text is sent to the RoboForth command line just as if you have typed it all. If this box is not checked you will see each line typed into RoboForth line by line. This does have some uses in debugging. If this box is checked each line is invisible and replaced with a chevron (greater-than) > character. So you will simply see rows of >>>>>>

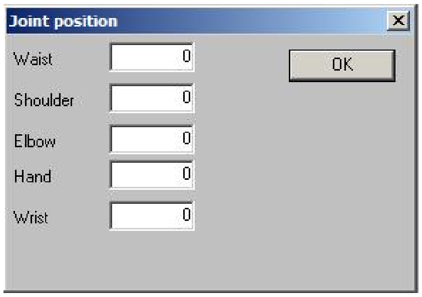
Joint names – the critical item is number of axes. The system will lock up if you put in 6 for a 5-axis system or vice-versa. You can then rename each axis, for example an R12 will normally be Waist, Shoulder, Elbow, Hand, Wrist. The 6th axis if there is one (axis 5 on the list counting from 0) will be track or yaw. You can choose different names if you prefer a different convention.
The number of axes should also match the value of #AXES in RoboForth (#AXES ?).
Both the above values would normally be 5 for R12 and R17, 4 for R19, 6 for an R12-6.
Cartesian names – for a jointed arm (R12, R17) these will normally be X Y Z PITCH W (W is roll). If there are 6 axes the last two will be ROLL and YAW. If there is a track fitted then there would normally only be 5 Cartesian axes. The track position is not added in to the X dimension either in RobWin or in RoboForth.
Again the number of Cartesian axes should match RoboForth. In RoboForth version 13.6 up there is an additional variable #CARTS (#CARTS ?). This would normally be the same as #AXES i.e. 5 for R12 and R17, 6 for R12-6 and 4 for R19.
Font – set your preferred font for each window.

Comm

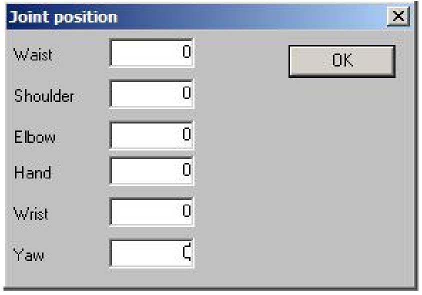
Re-open – useful if you closed the communications window to test a supervisor. Only one program at a time can use a single port.
Configure – choose the COM port you are using, normally 1 for a serial port, or between 5 and 9 for a USB converter. Baud rate should be 19200 unless you have decided to increase the baud rate in the controller – instructions for that are in the system manual.
Save Communication – copies the contents of the communications window to a file. Enter the file name; saved as a text file .txt.

Robot

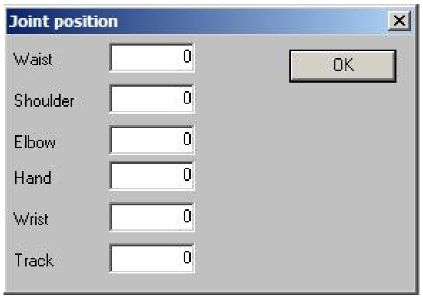
Joint position – shows current position of the robot in joint motor counts (same as JOINTWHERE). If there are 6 axes then the additional axis is shown as yaw or track:



Waist 0
Shoulder 0
Elbow 0
Hand 0
Wrist 0

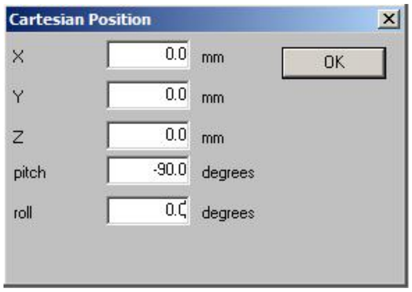


Waist 0
Shoulder 0
Elbow 0
Hand 0
Wrist 0
Yaw 0

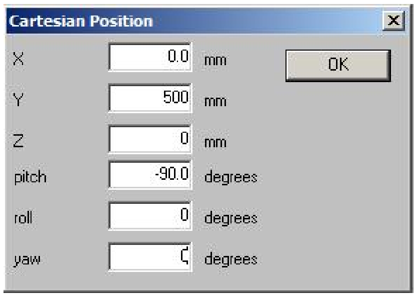


Waist 0
Shoulder 0
Elbow 0
Hand 0
Wrist 0
Track 0

Cartesian position – shows current position in Cartesian coordinates, 5 or 6 axes as selected in settings, configuration:



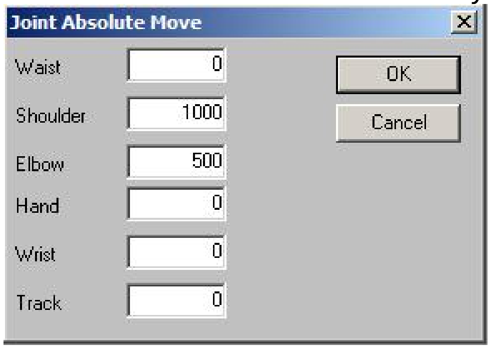
X 0.0 mm
Y 0.0 mm
Z 0.0 mm
pitch -90.0 degrees
roll 0.0 degrees



X 0.0 mm
Y 500 mm
Z 0 mm
pitch -90.0 degrees
roll 0 degrees
yaw 0 degrees

Joint absolute move – looks the same as above but you can enter your own values into the boxes. When you click OK the robot joints go to the positions entered. This is confirmed with Joint position or WHERE. You can use negative numbers.

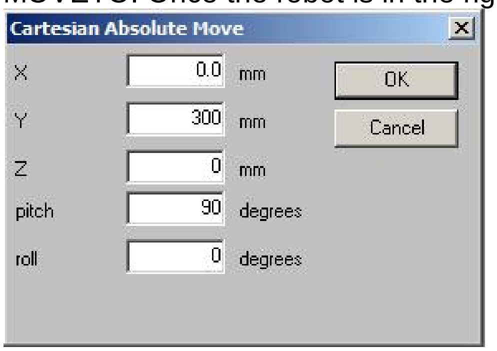
This time there is a cancel button if you wish to abort the motion.



Waist 0
Shoulder 1000
Elbow 500
Hand 0
Wrist 0
Track 0

Joint relative move – same as above but the robot moves **by** the amounts specified e.g. if shoulder position is 1000 and you enter 500 then the shoulder will move to 1500.

Cartesian absolute move – sends the robot to the absolute Cartesian position that you specify. This can be more convenient than the MOVETO commands in RoboForth because you can set up the hand and wrist (and yaw) parameters at the same time whereas in RoboForth you must use TOOL to set them up before you use MOVETO. Once the robot is in the right workspace you can then use MOVETO commands or TOOL


Cartesian Absolute Move

X 0.0 mm

Y 300 mm

Z 0 mm

pitch 90 degrees

roll 0 degrees

This will move the robot to zero on the X axis, 300mm from center on the Y axis and zero on the Z axis (which for R12/R17 means level with the shoulder pivot). Wrist rotates to 90 degrees i.e. points downwards.

Cartesian relative move – same as above but moves the robot **by** the amount specified.
Start, Calibrate, Home – sends the commands START CALIBRATE HOME
Teach – same as TEACH but sets the teach speed with a dialog box.
Jog – same as JOG but sets the jog increment with a dialog box.
Grip toggle – same as the grip icon in the tool bar.
Joint mode – sends JOINT and sets joint mode in the controller.
Cartesian mode – sends CARTESIAN and sets Cartesian mode in the controller.
Segmented – sends SEGMENTED which cancels CONTINUOUS path mode.
Continuous – sends CONTINUOUS which sets CONTINUOUS path mode.

Project

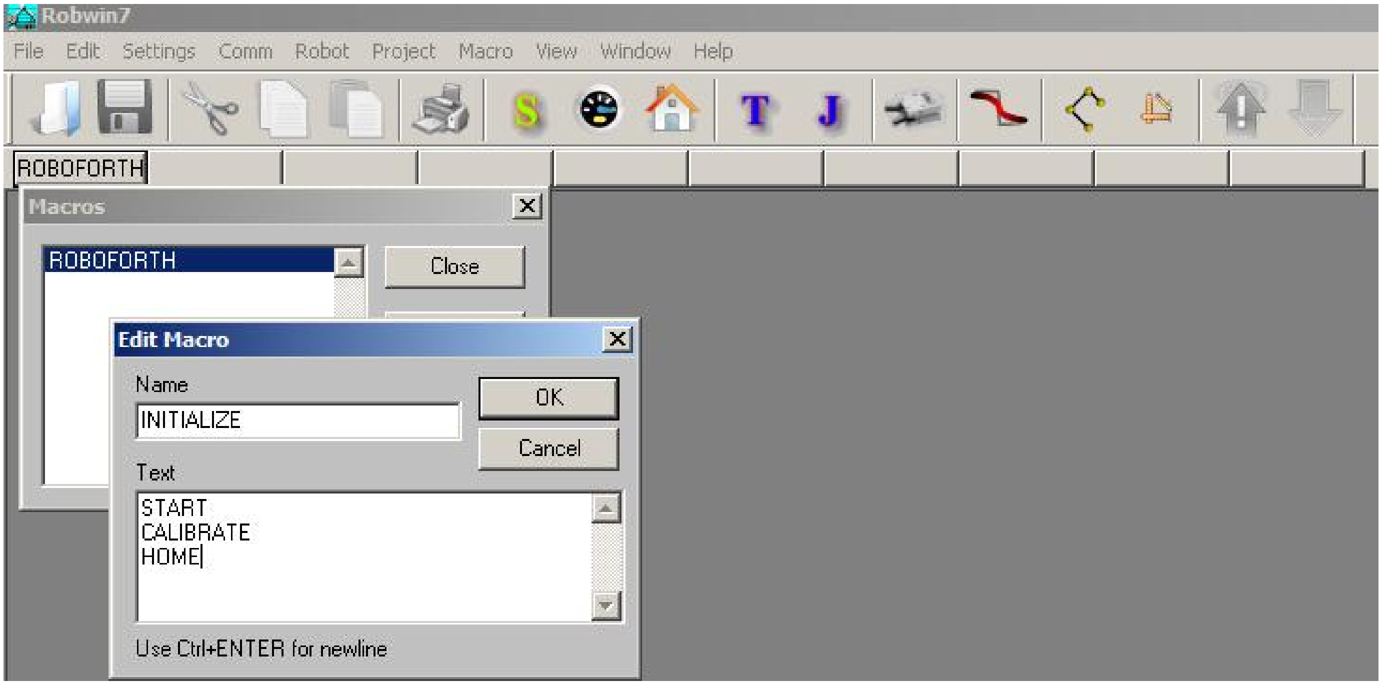
New – start a new project.
This creates 3 new windows:
Routes – lists and controls all the routes you create (see RoboForth manual).
Places – lists and controls all the places you create (see RoboForth manual).
ED2 window – this is where you write your RoboForth program that determines what the robot does and when, utilizing the data (routes and places) that you have created.
When the project is saves it creates 3 new files:
name.RUN – this is a binary image of the routes and places you created (in the upper bank of memory)
name.ED1 – this is a list of the names of the routes and places that are compiled into the dictionary.
name.ED2 – this is the text you create in the ED2 window.
Open – open an existing project
Close, Save, Save-as – usual Windows conventions.

To start a new project click **project**, **new** and choose a name for your project for example **myprog**.
When the project is open you will see three more windows: routes, places and a window labeled myprog.ED2
We advise (from experience) to move the communications window to the bottom and to re-arrange the other three so all are visible.
The system sends commands ROBOFORTH and STARTOVER to the controller which you can see in the communications window. This clears out the controller ready for the new session.

Macro

The macro bar is a line of 10 buttons that you can program yourself. You give a button your own name and also specify the text that is sent to the controller when the button is pressed.

1. click macro
2. open an existing file or create a new one
3. click macro again
4. click edit
5. you can see a list of existing buttons if any.
6. click new to add a new button
7. enter the name of the button that will appear on the actual button
8. in the lower box enter the text that will be sent to the controller. This can be more than one word. If so you can put the words on one line, separated by spaces or on new lines – type control-enter for a new line as in the example below.
9. click ok
10. For another button repeat from 6.
11. to edit an existing button highlight the existing button name and click edit
12. to use a button simply click it and the associated text is sent to the controller.


Robwin7
File Edit Settings Comm Robot Project Macro View Window Help
S
T
J
ROBOFORTH
Macros
Close
Edit Macro
OK
Cancel
Use Ctrl+ENTER for newline

ROBOFORTH

Name: INITIALIZE

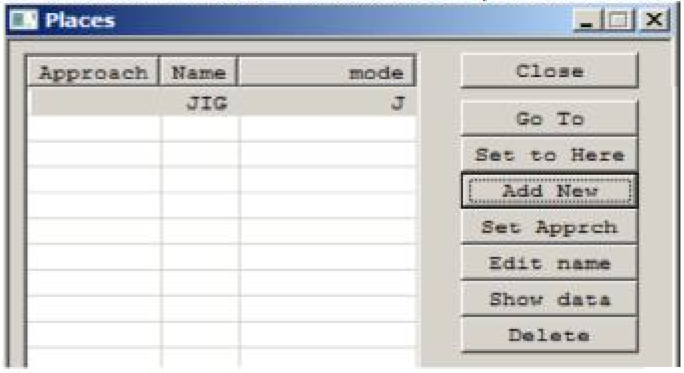
Text:

```
START
CALIBRATE
HOME
```

Teaching the robot is a process of moving to a position and learning that position. There are two devices for learning robot positions:
a ROUTE - this is a named list of positions which the robot follows in sequence.
a PLACE - this is a named single position to which the robot will go when required.
The user must choose which is most suitable for the task and must also choose a name which describes what the route or place means in the context of the application.
A task can involve more than one route or place or a mixture. All the routes and places will be saved together by ROBWIN in 2 files regardless of how many routes or places you have.

PLACES

Move the robot to any position you like using any of the methods described above.
Click the Places dialog box. If you can't see it click **Project > Places** This brings up list of existing places (empty).
Click **Add New**, enter a name for the place, for example JIG and click OK.



Approach	Name	mode
	JIG	J

To make the robot move to this position click the communications window and enter the name of the place, for example suppose the place name were JIG. Enter:

HOME	robot goes to home position
JIG	robot moves back to the place position.

If you regard this position as a target position you can now move the robot back to a position which is safe for the robot to move to before it goes into the target position. You then click on**Set Apprch** in the Places window.
The PLACE now has two positions: the approach position and the target position. Example of using this feature

HOME	Moves all the joints to zero count positions.
JIG	Moves robot to JIG *via* the approach position.
WITHDRAW	Moves robot back to the approach position.

Only ever use the withdraw command after the place name has been executed.
Note: just creating the place was enough to record it's position. Just entering the place name is enough to move the robot there. A Place is self learning and self executing.
If you want to change the coordinates of a place move the robot to the new position and click on the**Set to here** button.

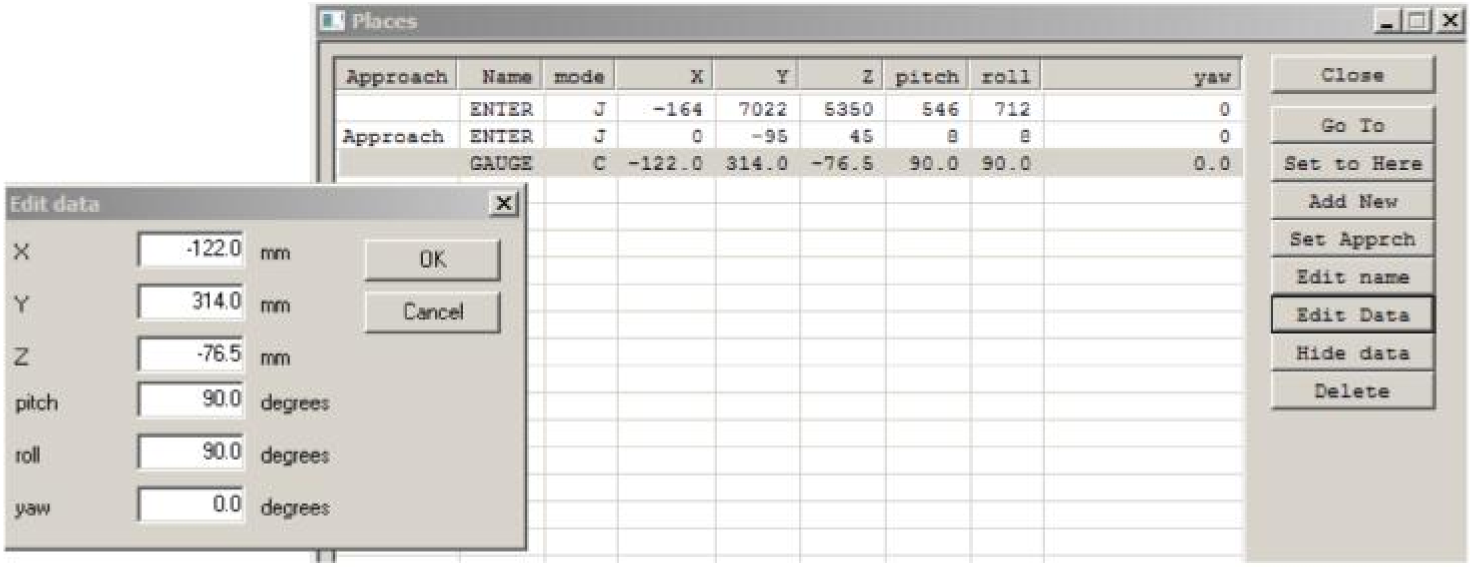
When you create text in the ED2 window, eg

```
: TASK
HOME
JIG
WITHDRAW
;
```

click the down arrow ⬇ to download to the controller.
Each time you edit the text click ⬇ again to download to the controller.

In the Places window you will also see a mode J or C. If you create a place while you are in JOINT mode the coordinates learned will be joint coordinates and the mode will be J. If you create the place while you are in CARTESIAN mode then the coordinates learned will be Cartesian and the mode will be C. If you execute a place-name by typing its command, if the place is a J type the robot will go to the joint coordinates within that place and the system will change to JOINT mode. If the place is a C type the robot will go to the Cartesian coordinates within that place and the system will change to CARTESIAN mode.

If you click **Show data** the places window opens up to show the actual coordinates learned for each PLACE.



Approach	Name	mode	X	Y	Z	pitch	roll	yaw
	ENTER	J	-164	7022	5350	546	712	0
Approach	ENTER	J	0	-95	45	8	8	0
	GAUGE	C	-122.0	314.0	-76.5	90.0	90.0	0.0

X	-122.0	mm
Y	314.0	mm
Z	-76.5	mm
pitch	90.0	degrees
roll	90.0	degrees
yaw	0.0	degrees

You can now edit the data manually if you wish. Click the name of the PLACE. That line is highlighted and the header line at the top changes to show the coordinates names (Joint or Cartesian). Click edit data to edit a line. A new box appears and you can change the value in any box. Click OK to save the new value(s).

You can also delete or rename a PLACE. However this may cause the project to be reloaded to get the words in the right order.

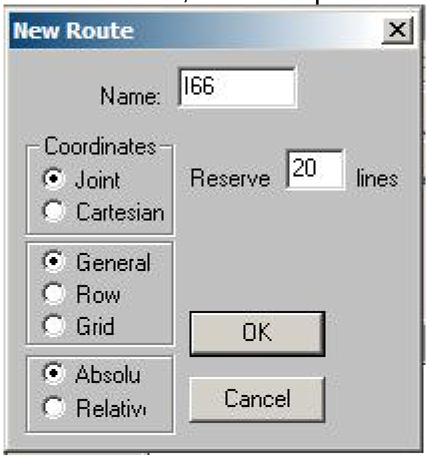
ROUTES

Find the **routes** window or click **Project Routes**.

Click the **New** button

SEQUENTIAL ROUTE IN JOINT MODE

Enter a name, for example I66



Name: I66

Coordinates: ● Joint ○ Cartesian

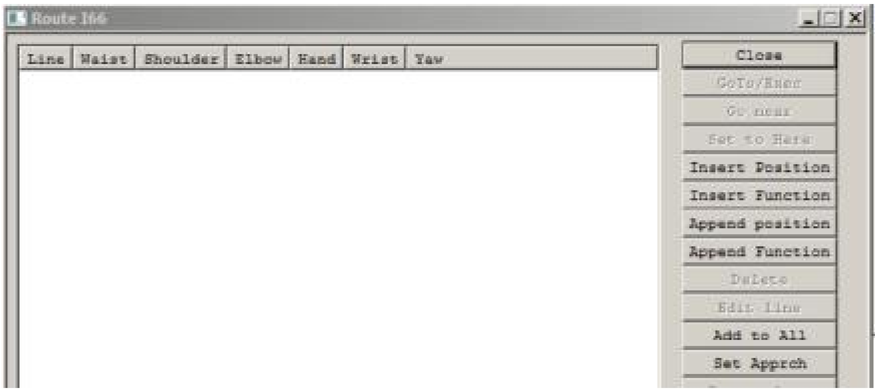
Reserve 20 lines

● General ○ Row ○ Grid

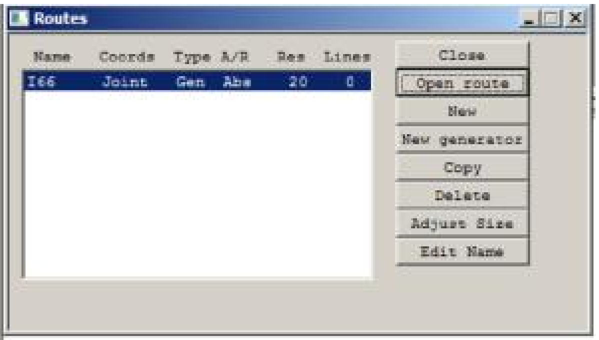
● Absolu ○ Relativ

You must then reserve some space for the route. The default is 20 but you can reserve hundreds if you think you'll need them.

Click OK and this creates a window for the new route:

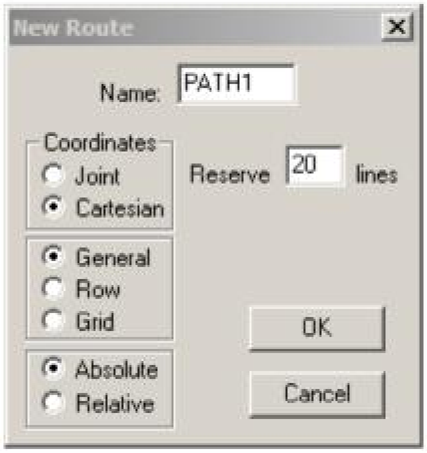


Line	Waist	Shoulder	Elbow	Hand	Wrist	Yaw



Name	Coords	Type	A/R	Res	Lines
I66	Joint	Gen	Abs	20	0

However Cartesian coordinates are easier to use. When you create a new route tick the Cartesian box.



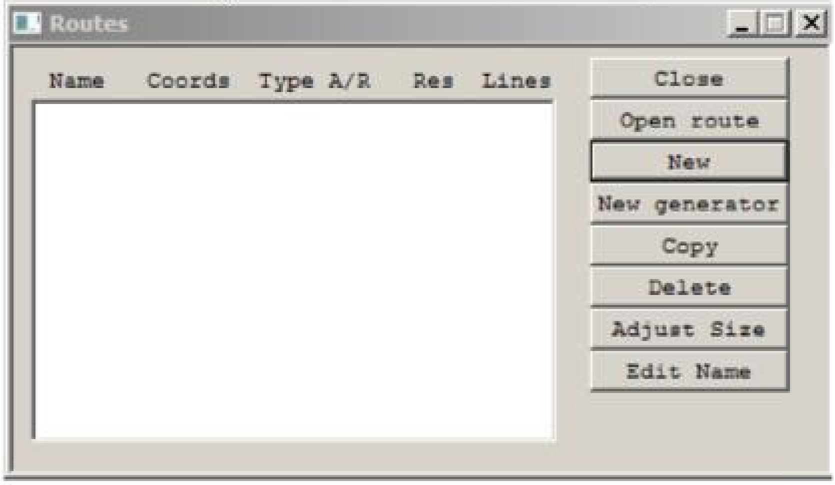
Name: PATH1

Coordinates: ○ Joint ● Cartesian

Reserve 20 lines

● General ○ Row ○ Grid

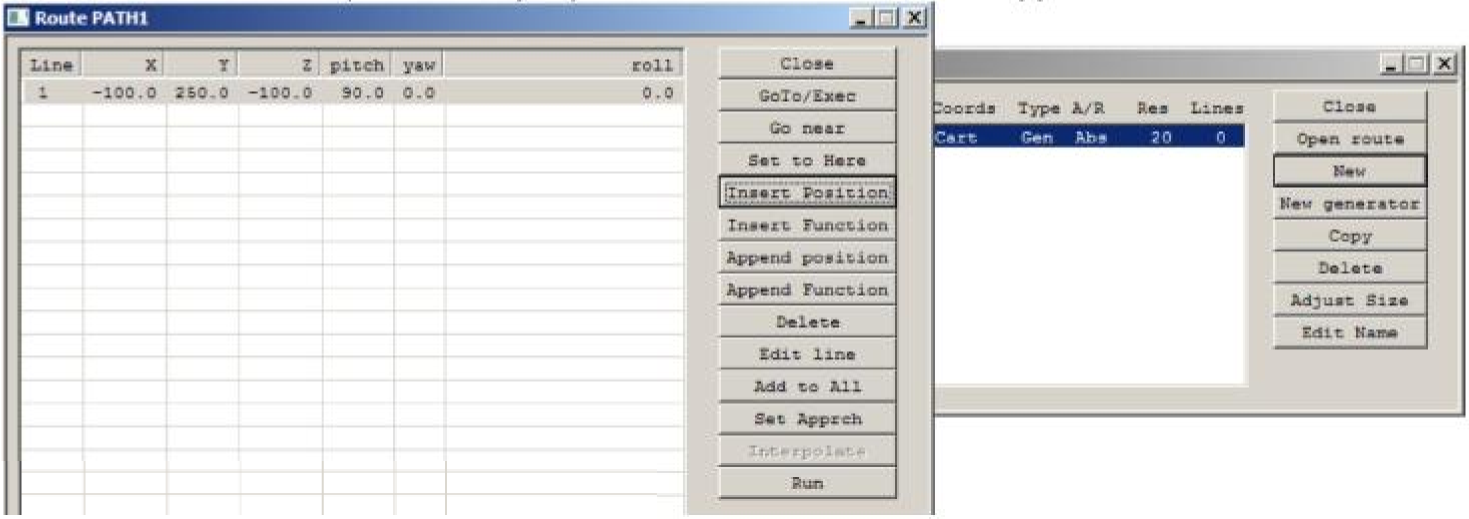
● Absolute ○ Relative



Name	Coords	Type	A/R	Res	Lines

Learning

Move the robot to desired position using any method described above. Click**Append Position**



Line	X	Y	Z	pitch	yaw	roll
1	-100.0	250.0	-100.0	90.0	0.0	0.0

Coords	Type	A/R	Res	Lines
Cart	Gen	Abs	20	0

(note the columns are adjustable including the border to the right of roll)
For the next position highlight the next blank line and click**Append Position** again and so on, building a list of positions. If you want to insert a position in front of an existing line then click the line to highlight it and click **Insert Position** Don't forget to click back inside the communications window before you type a command.
To RUN a route click Run or type RUN in the communication window. The robot will run from point to point stopping at each learned point.

To keep the robot running through all the points without stopping click the SMOOTH button
The definition of SMOOTH is CONTINUOUS NORMAL ADJUST so SMOOTH sets CONTINUOUS mode and sets speed to the default speed (as set with START) reduced if necessary for the currently selected route.
(see the RoboForth manual, continuous path)

Editing a route

To change the data in a line move robot to new position, highlight the line to change and click
Set to Here
To insert a line move the robot to desired position, highlight the line and click**Insert Position**. The new line will be inserted in front of the highlighted line.
You can also edit the actual data in a line with **Edit Line**
You can also insert or learn a function with **Append Function or Insert Function**. The function could be, for example GRIP or UNGRIP. These commands will then be carried out as the robot runs through the route.
The Insert Function also asks for a value. GRIP and UNGRIP require no value but you could, for example, change speed by inserting function SPEED then a value. The DSP will then change speed when it gets to that line. It will stay changed until set to another value. Another possible function is MSECS with a value, for example MSECS then 1200 inserts a 1.2 second delay.
You can only run a route with functions in segmented mode apart from speed changes. For I/O actions permitted in continuous mode see the RoboForth manual section 7.2.4

Using the teach pad to learn a route

Click on T button to activate the teach pad, as described above. Do not type TEACH – it is not the same. For Cartesian mode click the J button and jog the robot to the desired position.

When satisfied with the position press the red Ö key (or green on Android pad). This learns the position into the currently selected route. The red **X** deletes the last line learned.

Creating a task with both routes and places

Click on the ED2 window. You can now enter definitions in this window, for example

```
: TASK
JIG
I66 RUN
;
```

To download this file to the controller click on the ⬇ button. This also automatically saves the file to disk.
To test this function simply type

```
TASK
```

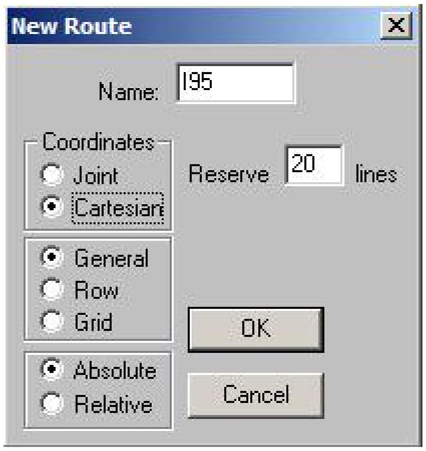
A variation might be:

```
: TASK
10000 SPEED !
JIG
20000 SPEED !
I66 CONTINUOUS ADJUST RUN
10000 SPEED !
;
```

Each time you click the download button the new text is automatically saved in the .ED2 file however you need to click **project, save** to save the data on disk (saves to the .ED1 and .RUN files).

SEQUENTIAL ROUTE IN CARTESIAN MODE

In the routes window click New. In the route box enter the name of the route then click the Cartesian radar button and enter the number of lines to be reserved.



New Route

Name: I95

Coordinates
○ Joint
● Cartesian

Reserve 20 lines

● General
○ Row
○ Grid

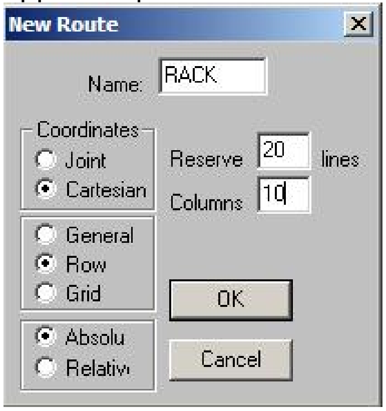
● Absolute
○ Relative

Reference routes in Cartesian mode

Row

A Row is a list of positions that would not be RUN but are just used as references, for example positions in a rack of test tubes.

In the routes window click New, enter the name of the route then click the Cartesian and Row radar buttons and enter the number of lines to reserve and the number of positions in the route. Naturally the number of lines reserved should be more than the number of positions or columns in the row. Allow at least one more line for an approach position. Click OK.


New Route

Name: RACK

Coordinates: ○ Joint ● Cartesian

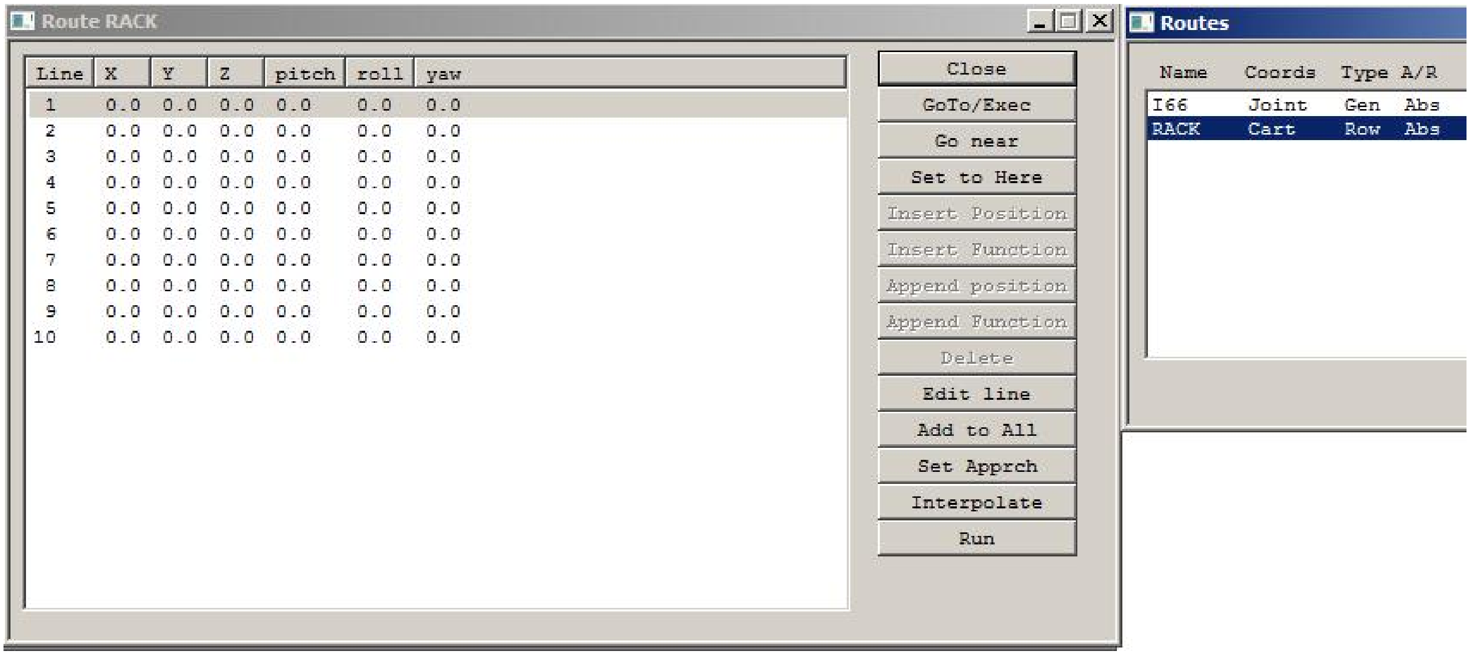
Reserve 20 lines

Columns 10

○ General ● Row ○ Grid

● Absolu ○ Relativ

.


Route RACK

Line	X	Y	Z	pitch	roll	yaw
1	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0
8	0.0	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0	0.0	0.0

Routes

Name	Coords	Type	A/R
I66	Joint	Gen	Abs
RACK	Cart	Row	Abs

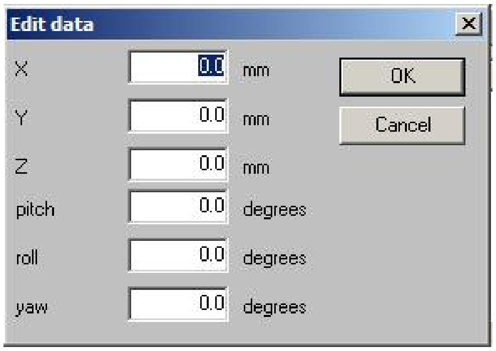
Move the robot to the first position and highlight line 1

Then move to the last position and highlight the last line (10 in this case)

Click **Interpolate**

To change an end point (line 1 or 10) move the robot there and click**Set to Here.** Then click **Interpolate** again

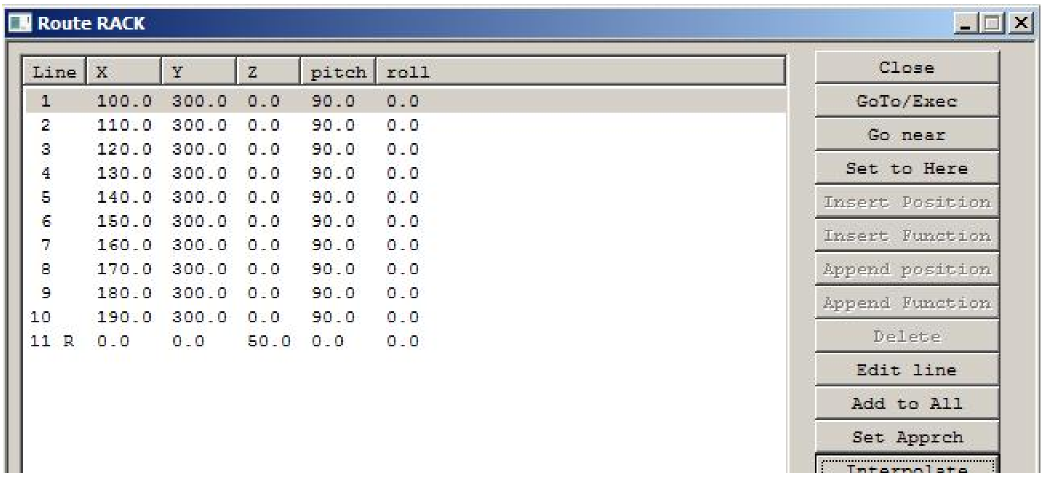
You can also edit a line manually with **Edit Line** This brings up a dialog box called “Edit data” which asks for the amounts to move in each axis.



X	0.0	mm
Y	0.0	mm
Z	0.0	mm
pitch	0.0	degrees
roll	0.0	degrees
yaw	0.0	degrees

You can shift all the coordinates of all the lines in any direction with **Add to All**. The same Edit data box appears and values entered there are added to each line in the route.

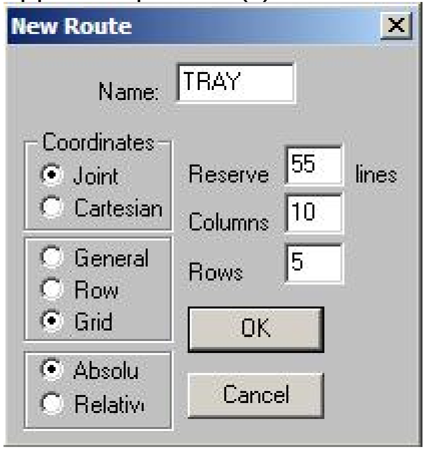
Similar to the approach position of a PLACE name you can also have an approach position for a ROW. It is simply an extra line (11 in this case) with relative coordinates in it. These coordinate values are added to any of the lines by the controller to give an approach position for all the lines.
To create the line go to any position (usually the last line because that is the last one you have learned) and do a relative move, for example 0 0 50.0 MOVE to move up 50mm in Z. Or use the teach pad in Jog mode. Then click **Set Approach**. The relative move will appear as the extra line with an R by it. This means the data in this line is relative. Whenever you GoTo this line the robot will simply move by the relative amounts (50mm in Z in this example). Highlight any line and click **Go near** and the robot will go to the approach position for that line which is the sum of that line and the R line. In RoboForth the commands are n NEAR and n INTO.



Line	X	Y	Z	pitch	roll
1	100.0	300.0	0.0	90.0	0.0
2	110.0	300.0	0.0	90.0	0.0
3	120.0	300.0	0.0	90.0	0.0
4	130.0	300.0	0.0	90.0	0.0
5	140.0	300.0	0.0	90.0	0.0
6	150.0	300.0	0.0	90.0	0.0
7	160.0	300.0	0.0	90.0	0.0
8	170.0	300.0	0.0	90.0	0.0
9	180.0	300.0	0.0	90.0	0.0
10	190.0	300.0	0.0	90.0	0.0
11 R	0.0	0.0	50.0	0.0	0.0

Grid

To create a matrix choose new route. In the dialog box enter the name of the route, click grid the number of rows and collums. Make sure you reserve enough space for the number of lines plus an extra one or two for approach position(s)


New Route

Name: TRAY

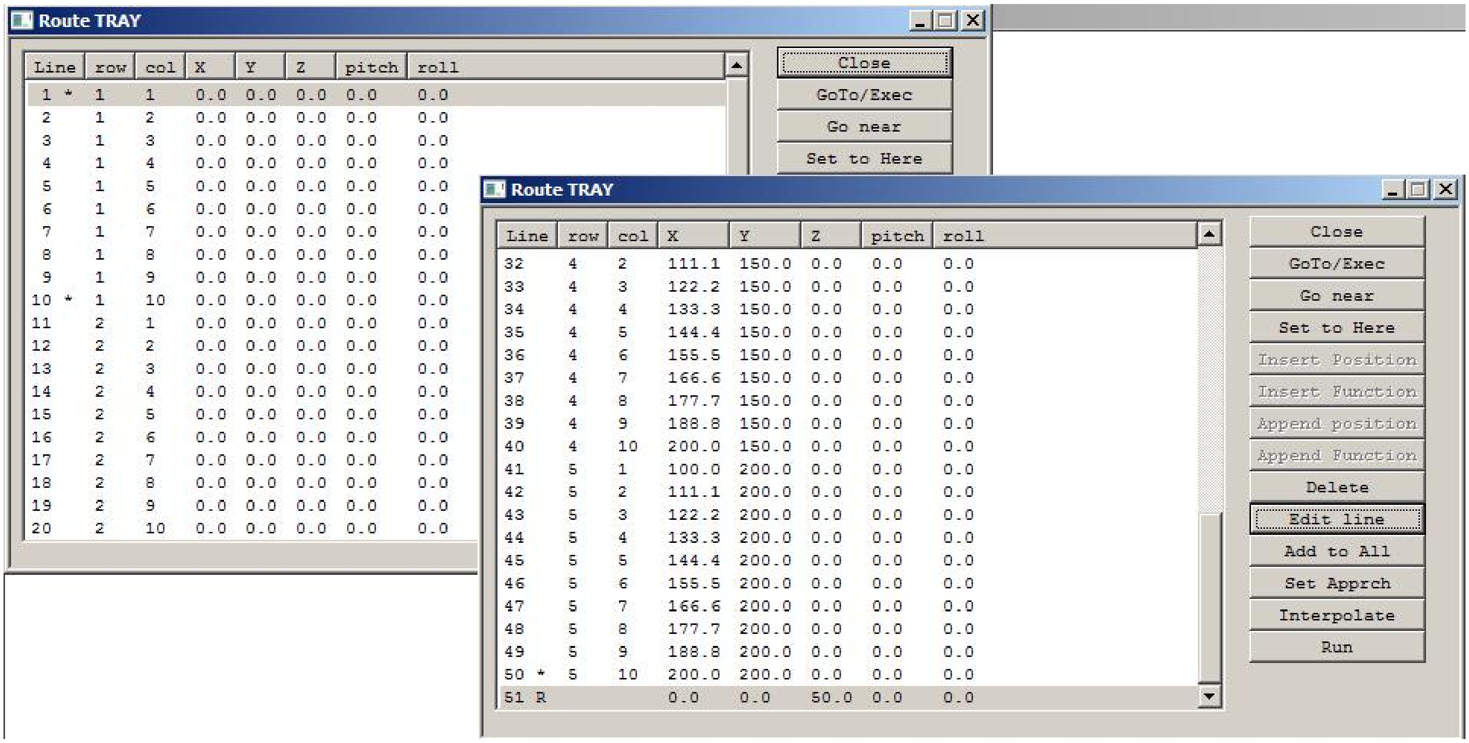
Coordinates
● Joint
○ Cartesian

○ General
○ Row
● Grid

● Absolu
○ Relativ

Reserve 55 lines
Columns 10
Rows 5

With a grid you need to learn 3 corners of the matrix. These are indicated by an asterisk by the relevant lines. Once you have learned each of the 3 lines click**interpolate**. Next you can learn an approach position in the same way as for a row. See example below with 10 columns by 5 rows plus one approach position.


Route TRAY

Line	row	col	X	Y	Z	pitch	roll
1 *	1	1	0.0	0.0	0.0	0.0	0.0
2	1	2	0.0	0.0	0.0	0.0	0.0
3	1	3	0.0	0.0	0.0	0.0	0.0
4	1	4	0.0	0.0	0.0	0.0	0.0
5	1	5	0.0	0.0	0.0	0.0	0.0
6	1	6	0.0	0.0	0.0	0.0	0.0
7	1	7	0.0	0.0	0.0	0.0	0.0
8	1	8	0.0	0.0	0.0	0.0	0.0
9	1	9	0.0	0.0	0.0	0.0	0.0
10 *	1	10	0.0	0.0	0.0	0.0	0.0
11	2	1	0.0	0.0	0.0	0.0	0.0
12	2	2	0.0	0.0	0.0	0.0	0.0
13	2	3	0.0	0.0	0.0	0.0	0.0
14	2	4	0.0	0.0	0.0	0.0	0.0
15	2	5	0.0	0.0	0.0	0.0	0.0
16	2	6	0.0	0.0	0.0	0.0	0.0
17	2	7	0.0	0.0	0.0	0.0	0.0
18	2	8	0.0	0.0	0.0	0.0	0.0
19	2	9	0.0	0.0	0.0	0.0	0.0
20	2	10	0.0	0.0	0.0	0.0	0.0

Route TRAY

Line	row	col	X	Y	Z	pitch	roll
32	4	2	111.1	150.0	0.0	0.0	0.0
33	4	3	122.2	150.0	0.0	0.0	0.0
34	4	4	133.3	150.0	0.0	0.0	0.0
35	4	5	144.4	150.0	0.0	0.0	0.0
36	4	6	155.5	150.0	0.0	0.0	0.0
37	4	7	166.6	150.0	0.0	0.0	0.0
38	4	8	177.7	150.0	0.0	0.0	0.0
39	4	9	188.8	150.0	0.0	0.0	0.0
40	4	10	200.0	150.0	0.0	0.0	0.0
41	5	1	100.0	200.0	0.0	0.0	0.0
42	5	2	111.1	200.0	0.0	0.0	0.0
43	5	3	122.2	200.0	0.0	0.0	0.0
44	5	4	133.3	200.0	0.0	0.0	0.0
45	5	5	144.4	200.0	0.0	0.0	0.0
46	5	6	155.5	200.0	0.0	0.0	0.0
47	5	7	166.6	200.0	0.0	0.0	0.0
48	5	8	177.7	200.0	0.0	0.0	0.0
49	5	9	188.8	200.0	0.0	0.0	0.0
50 *	5	10	200.0	200.0	0.0	0.0	0.0
51 R			0.0	0.0	50.0	0.0	0.0

Curve Generator

The curve generator allows you to construct a path mathematically that comprises straight lines and curves by specifying the length and direction of lines, radius, angle and orientation of curves.

To do this you start with a reference route that lists the specifications of the curve. You then click the generate button and an associated route is filled with the calculated coordinates the robot is to follow. Don't forget that when you run this route you will most likely need SMOOTH mode i.e. SMOOTH RUN If this results in too low speed use SETTINGS to increase both SPEED and ACCEL then use SMOOTH again.

Click **New generator** and provide a name. This will create two routes, the target route using your provided name and the generator route with the same name preceded by underscore, e.g. R1 and _R1. It is the target route R1 that you will eventually RUN. It’s useful to open this route so you can see the calculated coordinates appear.

Reserve enough lines for each. The default is 20 lines for the generator and 200 for the ultimate coordinates however you typically need more than 20 for complex shapes. Change to 100.
Next click **Append Function.** Choose SDP (Start,Direction,Point). Leave Param A and B at zero.
Once you have done this you need to enter 3 lines of coordinates:
The first line following is the start point. Its hand angle values are also copied to all lines in the generated route.
The second line is any point on the line of the starting direction. Its angle values are ignored. The actual coordinates are not important, it just tells the system what direction we are going in.
The third line is any point on the plane we wish to define to the left of the starting direction. Its angle values are also ignored. Being to the left defines which side of the plane we are considering as upwards and how we measure angles as positive anticlockwise.

Sometimes it is easier just to click **Append position** 3 times and then edit them.

Make a straight line
1 Click **Append** Function and choose **SDP** - leave both params zero.
2 Then enter the 3 coordinates: first line is the starting position of the line. You would typically use the current position of the robot so just click **Append position**. You can edit this later.
3 Pick some position on your line. This is simply the direction the line is heading. It can have X Y and Z values for a line at an angle in space. Probably the easiest is to move the robot to a position on your propose line and click **Append position**.
4 Click Append position again. The data in this third line is not used. Any values are ok and ignored.
5 Click **Append Function** and choose **INC**. Param A is the distance between points. This determines the number of points. If the increment does not fit in the number of points then the number of points is rounded up. This may leave the actual increment less than you specified.
6 Click **Append Function** and choose **LIN**. Param A is the length of the line in mm. Leave Param B zero.
7 Click **Generate.** The target route will fill with the calculated coordinates. The total length of the line will be the length you specified.
A line length of 100mm and an increment of 10mm will produce 11 lines.

Make a curve

1 Click **Append** Function and choose **SDP** - leave both params zero.
2 Then enter the 3 coordinates: first line is the starting position of the curve. You would typically use the current position of the robot so just click **Append position**. You can edit this later.
3 Pick some position on a line in the direction that the robot would follow if it were a straight line. This line is a tangent to the curve being generated. Probably the easiest is to move the robot to a position on this hypothetical line and click **Append position**.
4 Click Append position again. This needs to be *any* point *to the left* of the curve, i.e. somewhere inside the curve. This tells the system the plane of the curve. A good place to choose would be the center of the curve but that is not critical – anywhere on the plane will do.
If the X value of this point is positive of the starting point then the curve will arc towards the positive X value. If the X value is negative then the curve will arc to negative X. A similar rule for the other axes.
An easy way of doing the above 3 lines (steps 2,3,4) is to get the robot to the starting position using the Jog function. Press the tick key. Move sideways one increment only and press tick for the second line, then move at right angles to this (in the direction of the curve) one increment and press tick a third time. Then escape.
5 Click **Append Function** and choose **INC**. Param A is the distance between points. This determines the number of points. If the increment does not fit in the number of points then the number of points is rounded up. This may leave the actual increment less than you specified. Leave Param B at 0.
6 Click **Append Function** and choose **ARA**. Param A is the radius of the curve in mm. Param B is the angle the curve goes in degrees.
7 Click **Generate.** The target route will fill with the calculated coordinates. The total length of the line (computed with trigonometry) will be the length you specified.

Add a curve to a straight line

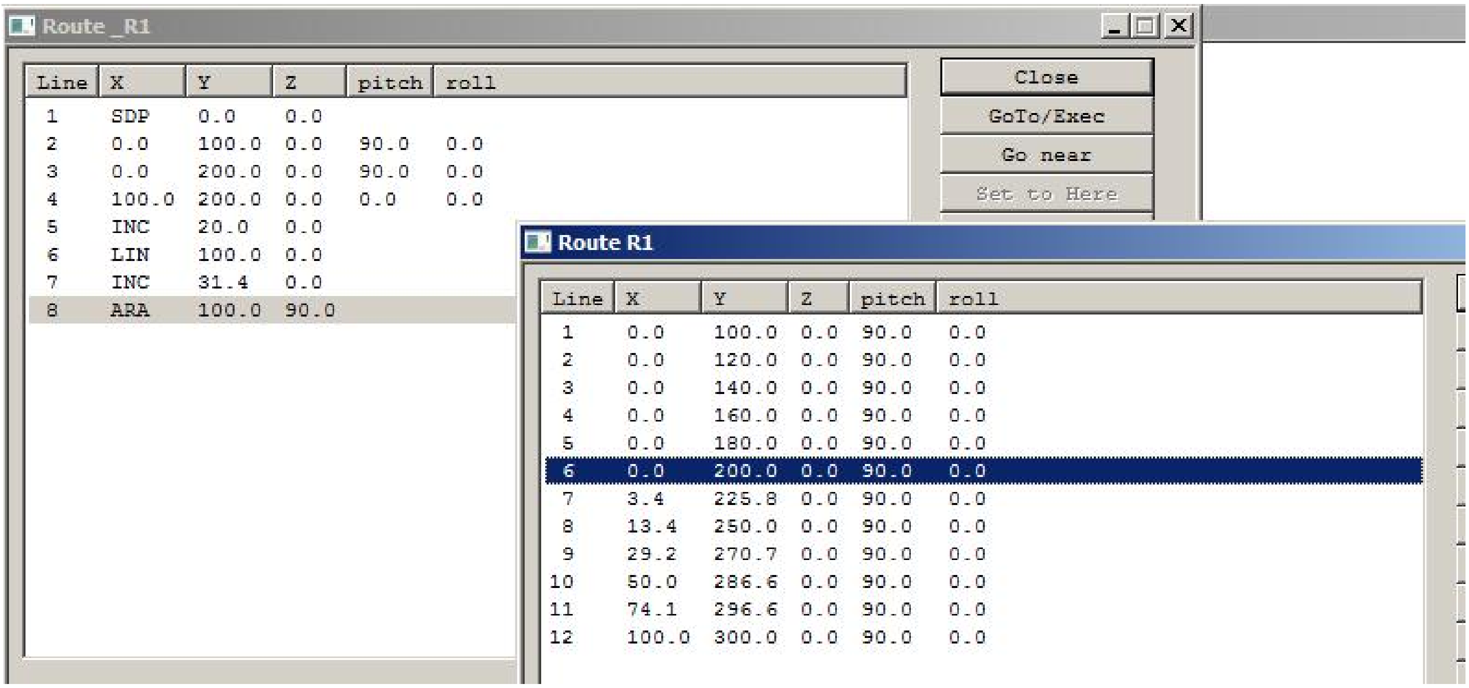
1 Having created the line as in “make a straight line” above, go back and edit line 4 so that coordinate lies on the plane on which the curve will be made.
2 Click **Append Function** and choose **ARA**. Param A is the radius of the curve in mm. Param B is the angle the curve goes in degrees.
3 Click **Generate.** The target route will fill with the calculated coordinates. The total length of the line (computed with trigonometry) will be the length you specified.

When you have a shape made from more than one segment (e.g. straight line plus a curve or two curves) you will see in the target route pairs of lines with the same coordinates. You need to delete one of them.

Such a route may be run in continuous mode – CONTINUOUS RUN. If you have two lines the same you will get the “too tight” error message. If you do not have 2 lines the same but get this message then you need to reduce SPEED or increase ACCEL. A word ADJUST reduces speed to a valid value. The second and more logical mode is CONTINUOUS TIMED. Set a value for SEGTIME using (val) SEGTIME ! Or using SETTINGS. Now when you RUN the route the time for each segment will be the same. Since you made all the segments the same length with INC the result is more or less constant speed. If you get “too tight” error it's because the SEGTIME is too short for one of the segments.

One thing you can do is add a very short segment to the beginning (with insert) and to the end (with append) which allows an acceleration and deceleration space. Then you can reduce the value of SEGTIME and increase the real speed.

Example and explanation.



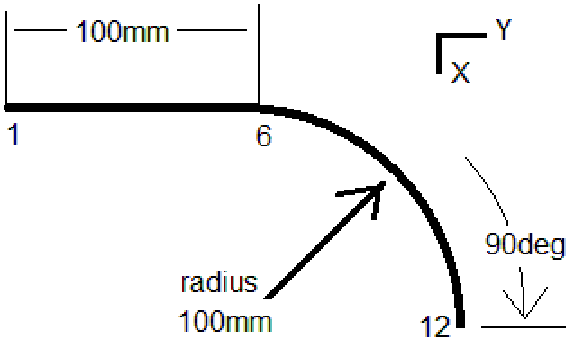
Route _R1

Line	X	Y	Z	pitch	roll
1	SDP	0.0	0.0		
2	0.0	100.0	0.0	90.0	0.0
3	0.0	200.0	0.0	90.0	0.0
4	100.0	200.0	0.0	0.0	0.0
5	INC	20.0	0.0		
6	LIN	100.0	0.0		
7	INC	31.4	0.0		
8	ARA	100.0	90.0		

Route R1

Line	X	Y	Z	pitch	roll
1	0.0	100.0	0.0	90.0	0.0
2	0.0	120.0	0.0	90.0	0.0
3	0.0	140.0	0.0	90.0	0.0
4	0.0	160.0	0.0	90.0	0.0
5	0.0	180.0	0.0	90.0	0.0
6	0.0	200.0	0.0	90.0	0.0
7	3.4	225.8	0.0	90.0	0.0
8	13.4	250.0	0.0	90.0	0.0
9	29.2	270.7	0.0	90.0	0.0
10	50.0	286.6	0.0	90.0	0.0
11	74.1	296.6	0.0	90.0	0.0
12	100.0	300.0	0.0	90.0	0.0

In _R1
Line 1 the starting command SDP
Line 2 the starting coordinates of the line, also specifying the hand pitch and roll.
Line 3 the heading of the line – i.e. heading off in the Y direction in this case.
Line 4 a point on the plane of the curve and inside it. So this curve will arc round in the X direction.
Line 5 set increment to 20mm
Line 6 make a line 100mm long.
Line 7 for the curve, change the increment to 31.4mm
Line 8 specify the curve as having a radius of 100mm and an arc of 90 degrees.
Then click generate.
In R1 you can see the straight line from line 1 to line 6, 100mm long comprising 6 lines and 5 segments. Then the curve starts. At a radius of 100mm and an increment of 31.4 this works out to 5 segments, i.e. from lines 6 to 12. It finishes up at X=100 and Y=300 and heading in the X direction.



Make a circle

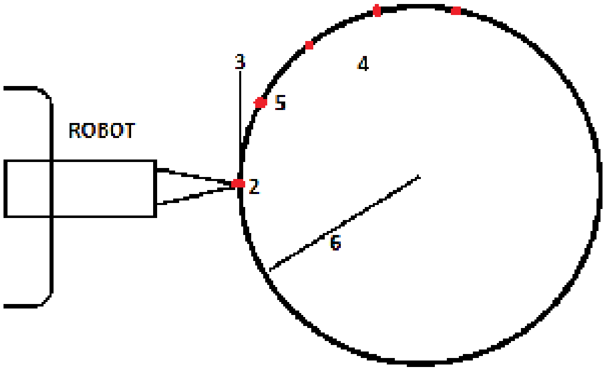
A circle is just a curve 360 degrees.

Drive the robot to the nearest point on the circle to the robot. If you want the hand to be horizontal you will need to enter the length of the hand.

When you send the robot to a particular Cartesian position it is the center of the wrist that occupies that position, the intersection of pitch and roll axes. You will want the center of the object to be there so you need to measure that distance.

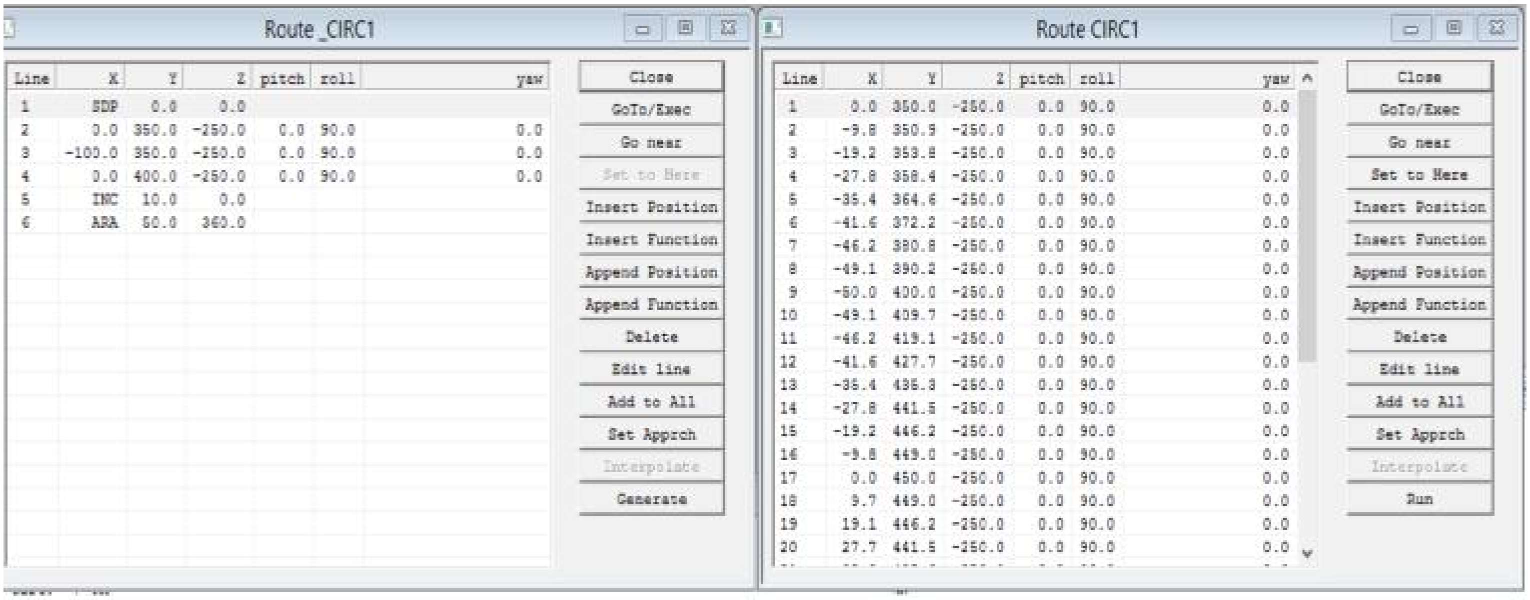
Enter
TOOL
WRIST PITCH 0.0
WRIST ROLL 90.0 or 0.0 as appropriate
TOOL LENGTH 130.0 (or whatever distance it is)
ALIGN N
EXECUTE? Y
OK

In the routes window click new generator and give it a name. Let's choose CIRC for the name.
You will get 2 new routes, _CIRC and CIRC



1 Click **Append** Function and choose **SDP** - leave both params zero.
2 Then enter the 3 coordinates: first line is the starting position of the curve. You would typically use the current position of the robot so just click **Append position**. You can edit this later.
3 Pick some position on a line in the direction that the robot would follow if it were a straight line. This line is a tangent to the curve being generated. Probably the easiest is to move the robot to a position on this hypothetical line and click **Append position**.
4 Click Append position again. This needs to be *any* point *to the left* of the curve, i.e. somewhere inside the curve. This tells the system the plane of the curve. A good place to choose would be the center of the curve but that is not critical – anywhere on the plane will do.
If the X value of this point is positive of the starting point then the curve will arc towards the positive X value. If the X value is negative then the curve will arc to negative X. A similar rule for the other axes.
An easy way of doing the above 3 lines (steps 2,3,4) is to get the robot to the starting position using the Jog function. Press the tick key. Move sideways one increment only and press tick for the second line, then move at right angles to this (in the direction of the curve) one increment and press tick a third time. Then press esc to get back to RobWin. For the Nexus press esc then its stop button.

5 Click **Append Function** and choose **INC**. Param A is the distance between points. This determines the number of points. If the increment does not fit in the number of points then the number of points is rounded up. This may leave the actual increment less than you specified. Leave Param B at 0.
6 Click **Append Function** and choose **ARA**. Param A is the radius of the curve in mm. Param B is the angle the curve goes in degrees.
7 Click **Generate.** The target route will fill with the calculated coordinates. The total length of the line (computed with trigonometry) will be the length you specified.



Line	X	Y	Z	pitch	roll	yaw
1	SDP	0.0	0.0			
2	0.0	350.0	-250.0	0.0	90.0	0.0
3	-100.0	350.0	-250.0	0.0	90.0	0.0
4	0.0	400.0	-250.0	0.0	90.0	0.0
5	INC	10.0	0.0			
6	ARA	50.0	360.0			

Line	X	Y	Z	pitch	roll	yaw
1	0.0	350.0	-250.0	0.0	90.0	0.0
2	-9.8	350.9	-250.0	0.0	90.0	0.0
3	-19.2	353.8	-250.0	0.0	90.0	0.0
4	-27.8	358.4	-250.0	0.0	90.0	0.0
5	-35.4	364.6	-250.0	0.0	90.0	0.0
6	-41.6	372.2	-250.0	0.0	90.0	0.0
7	-46.2	380.8	-250.0	0.0	90.0	0.0
8	-49.1	390.2	-250.0	0.0	90.0	0.0
9	-50.0	400.0	-250.0	0.0	90.0	0.0
10	-49.1	409.7	-250.0	0.0	90.0	0.0
11	-46.2	419.1	-250.0	0.0	90.0	0.0
12	-41.6	427.7	-250.0	0.0	90.0	0.0
13	-35.4	435.3	-250.0	0.0	90.0	0.0
14	-27.8	441.5	-250.0	0.0	90.0	0.0
15	-19.2	446.2	-250.0	0.0	90.0	0.0
16	-9.8	449.0	-250.0	0.0	90.0	0.0
17	0.0	450.0	-250.0	0.0	90.0	0.0
18	9.7	449.0	-250.0	0.0	90.0	0.0
19	19.1	446.2	-250.0	0.0	90.0	0.0
20	27.7	441.5	-250.0	0.0	90.0	0.0

At any time you can change the parameters, for example to change the radius edit line 6 of _CIRC1 and click Generate again. Or change the starting position in line 2. The center of the circle is of course line 2 plus the radius.

Don't forget that when you run this route you will most likely need SMOOTH mode.
SMOOTH RUN
If this results in too low speed use SETTINGS to increase both SPEED and ACCEL then use SMOOTH again.
Or enter high values directly into SPEED and ACCEL e,g,
20000 SPEED ! 2000 ACCEL ! SMOOTH RUN
once SMOOTH has calculated the best speed it is not necessary to use it again unless you change to another route.

Alternatively you may wish to go round the circle at constant speed. This is done by setting a fixed time for each increment of the circle. Suppose you have increments of 2mm. Enter
100 SEGTIME !
TIMED RUN
Then the robot will proceed at 2mm per 100 mS or 20mm per second.

IMPORTANT STEP WHEN CREATING A ROUTE WITH MORE THAN ONE SECTION

If you add a section with the same starting coordinates as the last line of the previous section you will see 2 consecutive lines with the same coordinates. These will not run in SMOOTH mode and in TIMED mode the robot will stop and pause on that coordinate for SEGTIME. Therefore at the end of each section make a note of the last line. When the target route is complete go back to those lines and delete the duplicates. If you click the **generate** button you will have to delete those lines again.

Using RobWin with Bluetooth.

When you switch on the controller the red LED on the adapter will come on and the blue LED will start flashing.
On your PC go to bluetooth and search for devices. You will see serial adapter. Pair with that using the PIN 1234.
Next go to control panel, device manager and expand COM ports. You will see 2 new COM ports. Note the lower number.
Go to comms, configure and choose the number from above. RobWin does not accept COM numbers greater than 9.
There may be a few seconds delay then you will see a message. Ignore any error message, press enter and you should see OK.

Using RobWin to communicate with a remote robot over the Internet.

You will need to install Virtual Serial Port from the folder Ethernet > HW VSP3s.
Run HW VSP3s.exe
Check Standalone Application
click next, next, install
Confirm firewall exceptions, answer yes, finish.
Virtual Serial Port is now installed.
On desktop double-click the icon HW Virtual Serial Port
Click the virtual serial port tab.
Select a com, e.g. com3
Click create COM
As soon as the red cross Delete COM appears communication is ready
Run RobWin7
If you are not already in communication click Comm, configure. choose COM3 (or whatever port you created with the VSP).
Note: communication over the Internet is very slow.

Validate

It is a bad idea to define words that are already defined in RoboForth. For example
TELL WAIST 1000 MOVE works as you expect.
But if you define
USER MOVE
now MOVE no longer has its original meaning. TELL …. MOVE will never work again.
Click Project Validate and this will list all words you have created (including Places and Routes) that were already in the RoboForth dictionary.