IRBCAM®

Users Manual

http://www.irbcam.com

December 7, 2020

Feature List

IRBCAM offers the following features.

- CAD/CAM files to industrial robot code for most common brands
- Compatible with CAM software which can generate APT-CL or G-Codes
- More than 10 million coordinate points possible in one setup
- Coordinated motion of robot, linear and rotary external axes
- Reachability, singularity and collision checking including external axes
- Fully configured wrist/elbow/base and safe robot paths
- 2D and 3D plotting with animated robot and path verification
- Toolpath optimizer which aids the user in finding good settings
- Command-line interface with possibility to use as post-processor

Contents

Quick Steps to IRBCAM

This section summarises the steps required to get started with IRBCAM. The steps required are the following:

- Generate your 2D or 3D drawing in a CAD program, such as AutoCad or SolidWorks.
- Load CAD file into a CAM package, such as SurfCam or VisualMill and generate tool cutting paths. Save as APT-CL file (recommended) or ISO G-code.
- Load APT-CL file or G-codes into IRBCAM and check that the programmed robot coordinates have no collisions, out of reach or singularity problems.
- When the path has been verified with IRBCAM, generate robot code, load into robot controller and start machining.

Nomenclature

This section explains some of the terms and terminology which are used throughout the manual.

- Cubes: Cubes are general elements used to represent tables, fences, walls, obstacles, etc. in your station. The user can specify each cube using width, length and height and a colour. There can be many cubes in a station.
- Geometry file: There can only be one geometry file in a station and this object represents the part to be machined. The dimensions of the geometry file should match the generated toolpath. Geometry files must be converted to the IRBLIB format using the CADConverter.exe and placed in the installation directory ..\IRBCAM\geometry
- Linear table coordinate system (or linear table frame): The linear table frame specifies the position and orientation of a linear, actuated table relative to the world coordinate system. If a geometry file is included in the station, it will be specified relative to the linear table frame. When the actuated table moves, the geometry file will move with it.
- Object coordinate system (or object frame): An object frame is specified relative to the user coordinate system. This feature is available on for example ABB and ADEPT robots, but not all robot brands. An example situation where the object frame is useful, is as follows: The user frame is calibrated to a corner of a working table bolted to the floor, while the object frame specifies

translation and rotation relative to that corner. The user frame will normally never change and needs to be calibrated only once. It is easier and faster to calibrate an object frame relative to the corner of a table (user frame), than to the base frame of the robot.

- Robot coordinate system (or robot base frame): The robot base frame specifies the location and rotation of the robot relative to the world frame. Normally, the robot base frame and the world frame are identical. In some situations it can be useful to make the robot frame different from the world frame, for example when the robot is mounted on a linear track or when the robot is ceiling mounted (upside-down).

- Roll, Pitch, Yaw angles: These three angles specify the orientation of coordinate systems. The Roll angle (RZ) is about the world Z-axis, the Pitch angle (RY) is about the current Y-axis, while the Yaw angle (RX) is about the current X-axis. Not all robot manufacturers use Roll, Pitch and Yaw angles, hence IRBCAM includes an angle converter from manufacturer specific angles to Roll, Pitch and Yaw.

- Rotary axis coordinate system (or rotary axis frame): The rotary axis frame specifies the position and orientation of an actuated turntable relative to the world coordinate system. If a geometry file is included in the station, it will be specified relative to the rotary axis frame. When the turntable rotates, the geometry file will rotate with it.

- Station: A system setup in IRBCAM is saved in a Station file with file extension IRB. The station file describes location and orientation of objects such as linear track, robot, turntable, linear actuated table, geometry file, stationary CAD, tool and cubes. Station files can be sent to and opened on another computer as long as custom-made objects (geometry, stationary CAD and tool) are placed in the installation directories, normally C:\Program Files\IRBCAM\tool (or \geometry).

- Stationary CAD: These objects are used when more detail is required compared to cubes. There can be many stationary CAD files in a station, and as opposed to geometry files these will not move when a turntable or linear actuated table moves.

- Toolpath: A toolpath contains either 3- or 5-axis information about tool tip location and orientation. IRBCAM supports toolpaths on APT-CL or G-Code formats. These files must be generated by separate CAM software, such as SurfCam, VisualMill or others.

- Tool coordinate system (or tool frame): The tool frame is located at the mounting flange of the robot and specifies the distance and orientation of

the tool tip relative to the mounting flange. Normally, the Z-axis of the tool frame points straight out from the mounting flange, but this is not standardised and definitions vary between the robot manufacturers. The advantage of using a tool frame definition becomes evident when the user wants to change the tool. Instead of having to modify the entire robot program, only the tool frame has to be updated.

- Track coordinate system: This coordinate system specifies the position and orientation of a linear track relative to the world coordinate system. A robot will be mounted on top of the track, and is specified relative to the track coordinate system. When the linear track moves, the robot base frame will move with it.

- User coordinate system (or user frame): This frame is relative to the robot base frame and is used to define a local coordinate system, typically on a working table. The advantage of having a user frame definition becomes evident when the user wants to move the part on the working table. Instead of having to modify the entire robot program, only the user frame has to be updated. Another advantage of defining a user frame is the fact that the CAM software which generates the toolpath does not need to know the location and orientation of the robot. The origin of the user frame defined on the robot and the placement of the physical part to be machined relative to that user frame should match the origin defined in the CAM software.

- World coordinate system (or world frame): The world frame is always located at the position X=Y=Z=0 with angles Roll=Pitch=Yaw=0 degrees. Normally, when the robot is in the home position, the upper arm points along the positive X-axis of the world frame, but this is not standardised and definitions vary between the robot manufacturers.

1 Safety Precautions

Industrial robots can cause significant damage to personnel, infrastructure and surrounding equipment if not operated properly. When using robot code generated by IRBCAM, the following safety precautions must be made.

- The robot must only be operated by personnel who have received proper training. In particular, the operator must know how to calibrate user frames and tool frames and to make sure that these frames are identical with the ones used by IRBCAM.
- The operator must not modify the generated robot code, for example by removing safety information embedded in the code such as arm configuration data.
- No personnel must enter inside the workspace of the robot in the following situations: A) The controller is in automatic mode, B) the controller is in manual mode when the operator is using the teach pendant, C) the machining spindle is running.
- Safety walls and/or doors must be used to protect the operator from potential loose parts such as the milling tool or from material being removed.
- IRBCAM is intended for robotic machining of soft materials. Machining of hard materials (such as steel) with a robot can cause serious damage to tooling and/or the robot itself.

2 Introduction

This document describes the IRBCAM software, an interface between G-Codes and the APT-CL format and the most common industrial robot languages. The APT-CL format is one of the most common CAM formats, and most CAM software can export to this generic machine-independent format. IRBCAM can also work with G-codes, see section 5 for more information. IRBCAM has been tested extensively on SurfCam, AlphaCam and CATIA v5, ProEngineer and VisualMill, which are some of the common CAM platforms.

IRBCAM connects the powerful capabilities of modern CAD/CAM platforms with the flexibility, accuracy and speed of industrial type robots. While such robots are normally not rigid enough to be used for hard metals, they are well suited for rapid prototyping and mould production in light metals, wood and foam materials. An industrial robot offers a cost-efficient alternative for 5-axis machining in lightweight materials. The repeatability of a robot is typically 0.1-0.2mm, while a typical absolute accuracy is 0.3-0.5mm, depending on the size of the workspace. However, this type of accuracy requires that the operator carefully calibrates the tool and the work object coordinate frames. When these frames are not calibrated carefully, position errors of several millimeters are not unusual.

The total workspace volumes that can be achieved, depends on the robot model type and varies from 0.5x0.5x0.5 cubic metres to more than 2x2x2 cubic metres. By placing the robot on a linear external axis, the workspace can be extended to very large distances (systems up to 100 metres have been built) in one of the workspace directions. By including a rotary external axis, the robot can machine large objects from all sides.

3 System Requirements

IRBCAM runs under the Windows (NT, 2000, XP, Vista, Win7, Win8) operating system and the software is available in both 32-bit and 64-bit versions. On Mac and Linux operating systems a 64-bit version is available. The PC running IRBCAM should have at least 1GB of RAM. If large APT files are used, then more working memory is recommended. The 32-bit version can utilize RAM up to 4GB, while the 64-bit version does not have this limitation. IRBCAM has been successfully tested on computers with 16GB and more RAM and toolpaths with more than 10 million coordinates. A relatively fast CPU is recommended for the collision detection functions. The Intel Core-i7 4770S 3.1GHz is an example of a CPU which runs the collision detection relatively fast. A good low-cost CPU is the Intel Pentium G860 3.0GHz (collision detection about 25-30% slower than with the i7-4770S).

4 Coordinate Systems

This section contains useful background information about common coordinate systems (user and tool frames) for industrial robots, as well as a short description of a common definition of axis rotations (roll, pitch, yaw).

4.1 User and Tool Frames

To program an industrial robot efficiently, it is strongly recommended to use two types of coordinate systems: 1) The User Frame and 2) the Tool Frame. The two main advantages of using these two coordinate frames are: 1) it becomes easy to change tool (only the tool coordinates need to be modified, and not the entire program) and 2) it becomes easy to move the entire toolpath inside the workspace of the robot.

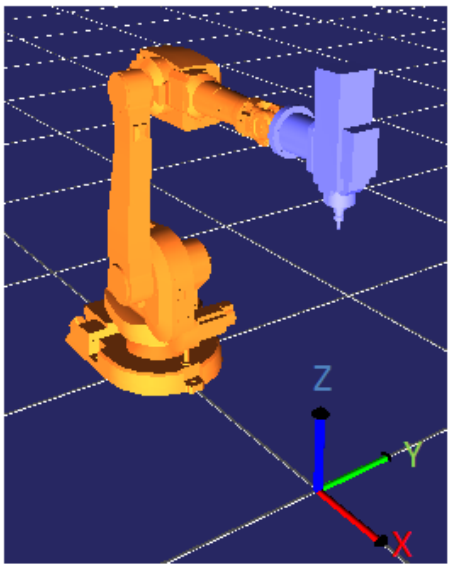


Figure 1: Definition of the user frame: X=1000mm, Y=0, Z=0 and the axes in the default directions.

All the major robot brands use the same default definition of the user frame, see Figure 1. The X-axis (red colour) points in the direction straight out from the robot base. The Y-axis (green colour) points to the left as seen from the robot base, while the Z-axis (blue colour) points from the floor towards the ceiling. The 0,0,0 position of the user frame is located at the base of the robot. The user can freely move the user frame in all three directions (X,Y,Z), as well as rotate the axes in any directions (roll, pitch, yaw). The user frame normally consists of 6 numbers (X,Y,Z plus roll, pitch, yaw). Note that ABB uses quaternions (4 numbers) instead of the

3 rotations. IRBCAM contains a converter between quaternions and the three (roll, pitch, yaw) angles.

The definition of the tool frame is slightly different between the various robot brands. The European brands ABB, Kuka and Comau use the definition shown in Figure 2 (left) while the Japanese brands Motoman and Fanuc use the definition shown in Figure 2 (right). The difference between the two definitions, is a 180° rotation about the tool Z-axis. IRBCAM is based on the European definition. Hence, when using IRBCAM with the Japanese brands, care must be taken when defining the tool frame coordinates, see for example section 7.13. For ABB, Kuka and Comau

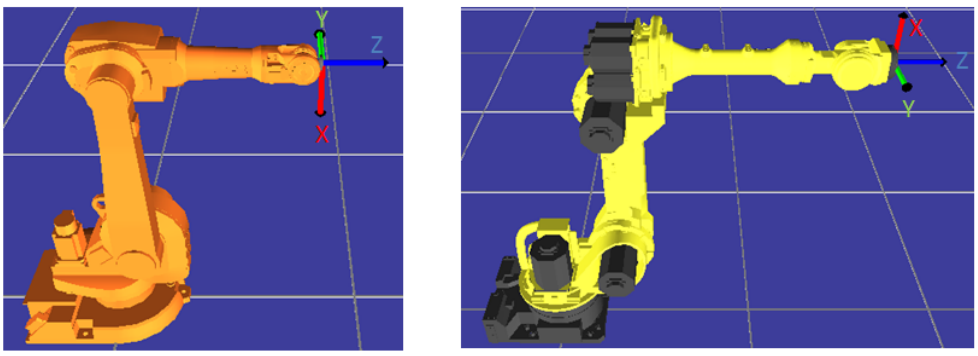


Figure 2: Definition of the tool frame. Left: ABB, Kuka and Comau definition. Right: Motoman and Fanuc definition.

robots, the default tool frame has the X-axis (red colour) pointing towards the floor (see Figure 2, left) when the robot is in the home position, while for Fanuc and Motoman robots the default tool X-axis points towards the ceiling (see Figure 2, right). Both definitions use the Z-axis (blue colour) as the main tool direction (for example the direction of the milling tool).

4.2 Roll, Pitch, Yaw Angles

IRBCAM uses the roll (RZ), pitch (RY) and yaw (RX) angles for describing rotations. Figure 3 illustrates three examples of single roll, pitch, yaw rotations, each 90°. The positive direction of rotation is given by the right-hand rule. Place the thumb in the axis direction - the positive rotational direction is then given by the remaining fingers.

Figure 4 illustrates a combined roll, pitch, yaw rotation. Roll is always performed first, followed by Pitch about the *new* Y-axis, followed by Yaw about the *new* X-axis.

Note that the majority of robot brands use roll, pitch, yaw angles, while some brands use quaternions and Z-Y-Z rotations. IRBCAM consistently uses roll, pitch, yaw for user- and tool frames and converts these angles to quaternions and Z-Y-Z angles when necessary.

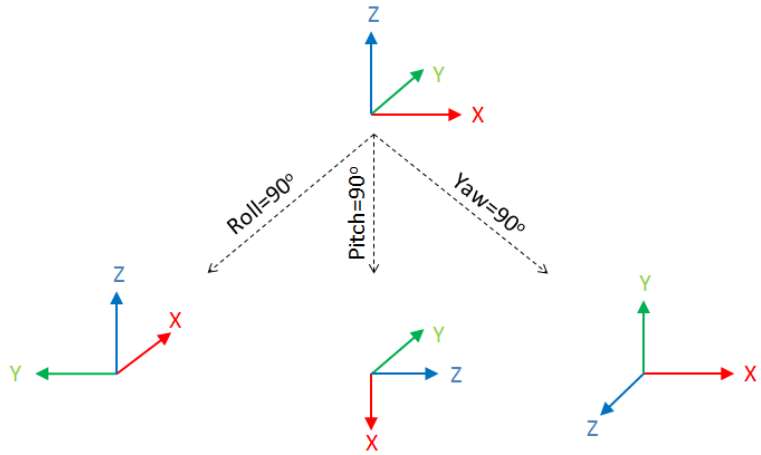


Figure 3: Three examples of single roll, pitch, yaw rotations.

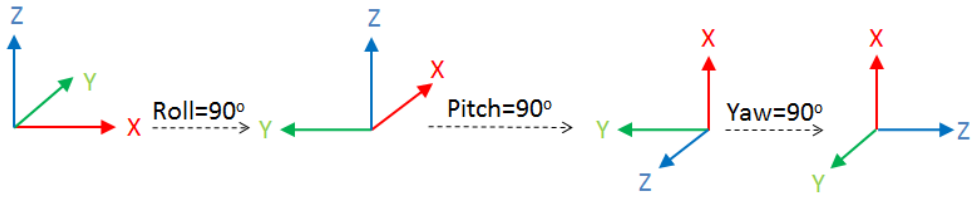


Figure 4: An example of a combined single roll, pitch, yaw rotation.

5 Input Formats

This section briefly describes the IRBCAM input formats: APT-CL and G-Codes.

5.1 APT-CL

Table 1 shows the APT-CL commands which are supported by IRBCAM.

UNITS/MM	Coordinate units. MM or INCH.
GOTO/X,Y,Z	Linear 3-axis move
GOTO/X,Y,Z,I,J,K	Linear 5-axis move, (I,J,K) is the tool vector. Default tool vector is (0,0,1).
CIRCLE/XC,YC,ZC,I,J,K,R	Arc/circular move where (XC,YC,ZC) is the circle centre, (I,J,K) is a vector normal to the arc/circle plane and R is the radius.
INDIRV/VX,VY,VZ TLON,GOFWD/ (CIRCLE/XC,YC,ZC,R),ON, (LINE/XC,YC,ZC,X,Y,Z)	Catia 3-axis circle format. VX,VY,VZ is the motion direction vector, XC,YC,ZC is the circle centre, X,Y,Z is the arc end-point.
HELICAL/CENTER,XC,YC,ZC, INDIRV,VX,VY,VZ, AXIS,AX,AY,AZ, PITCH,P,RADIUS,R,ANGLE,A, HEIGHT,H,ROUND,D,END,X,Y,Z	Catia helical format. XC,YC,ZC, VX,VY,VZ same as circle, AX,AY,AZ helical direction vector, p-pitch distance, R-radius, A-total helical angle,H-total height, D-number of revs,X,Y,Z-end point.
RAPID	Rapid (no contact) move.
FEDRAT/MMPM,V	Feed rate V. Units can be MMPM (mm per minute) or IPM (inches per minute).
LOADTL/x	Tool number x.
SPINDL/RPM,x	Spindle speed x.
MSYS/(12 numbers)	Coordinate transformation. MCS, $$*CATIA0, $$->CSYS also supported.
CYCLE/FEDTO,F,MMPM,V, RAPTO,P1,RTRCTO,P2, PULBAC,S	F is depth of the drill cycle, V is the speed in units MMPM or IPM, P1 is the rapid-to distance, P2 is the retract-to distance, S=98 means pull-back to P2, S=99 means pullback to P1. The parameters DWELL and INCR are not used by IRBCAM.
CYCLE/OFF	End of drilling cycle

Table 1: List of APT-CL Commands Supported by IRBCAM.

Notes:
A GOTO/ command is required both immediately before and after a CIRCLE/ command to define the start and end-points of the arc. If the start and end-points

are equal, then a full-circle is defined. There are two normal vectors (I,J,K) for a circle plane. These two different vectors are used to define the rotational direction. In the XY-plane, (I,J,K)=(0,0,-1) specifies a clockwise rotational direction, while (I,J,K)=(0,0,1) specifies a counter-clockwise direction. The tool vector during the circle/arc motion, is specified in the GOTO/ command immediately before the CIRCLE/ command.

Helical motion can be specified in two different ways: 1) Use the Catia HELICAL/ command or 2) if the Z-values in a full circle and the GOTO/ commands immediately before and after are different, then a helix motion is defined, see the generated APT file in the example in section 7.2. Below is an example using the Catia syntax for a lead-in circle and helical:

```
GOTO/-161.11973,-463.8388,253,0,0,1

INDIRV/-0.31849,-0.94793,0,TLON,GOFWD/,$
(CIRCLE/-156.38010,-465.43127,253,5),$
ON,(LINE/-156.3801,-465.43127,253,-157.97256,-470.17089,253)

HELICAL/CENTER,0,0,253,INDIRV,0.94793,-0.31849,0,AXIS,0,0,1,PITCH,10,$
RADIUS,496,ANGLE,9108,HEIGHT,253,ROUND,25.3,END,495.9753,-4.95004,0
```

The tooltrace of this example in IRBCAM is shown in Figure 5.

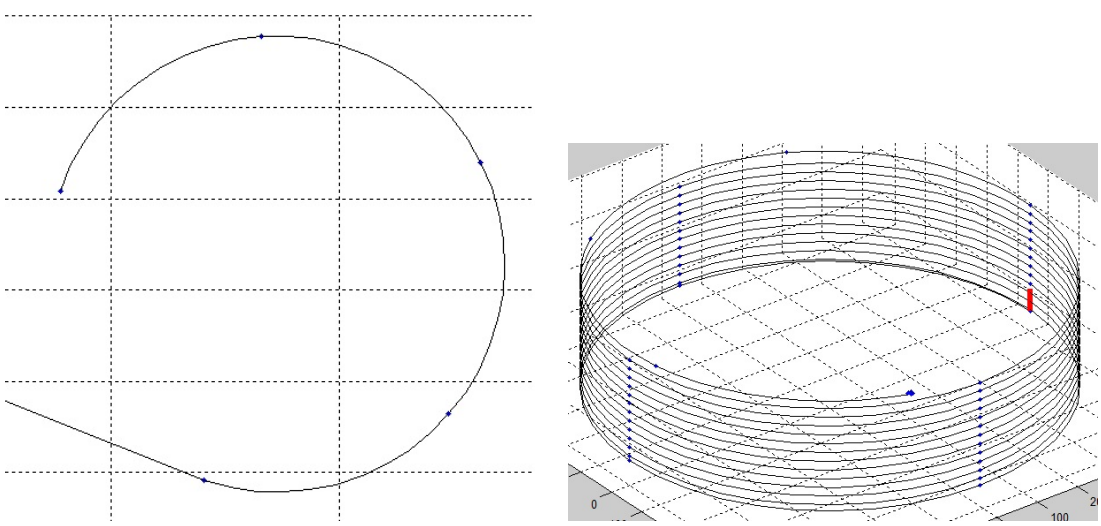


Figure 5: Tooltrace showing Catia syntax for circular lead-in (left) and helical (right).

The 12 numbers after MSYS/ are comma-separated. The first 3 numbers specify axes translations, while the remaining 9 numbers specify the axis rotations.

5.2 G-Codes

Table 2 shows the G-codes currently supported by IRBCAM.

G00	RAPID move
G01	Linear move
G02	Clockwise (CW) arc
G03	Counter-clockwise (CCW) arc
G17	XY-Plane
G18	ZX-Plane
G19	YZ-Plane
G20	Mirror X-axis
G21	Mirror Y-axis
G22	Mirror X&Y-axes
G23	No mirroring
G43	Tool Z-axis compensation
G70	Imperial units
G71	Metric units
G80	Start drill cycle
G81	End drill cycle
G90	Absolute programming
G91	Incremental programming
G92	Origin shift
G98	Drill cycle: Retract old z
G161	Relative arc centres
G162	Absolute arc centres

F x	Feedrate x MMPM
%L NAME	Procedure NAME
%	End of procedure
LL NAME	Procedure call
M3	Spindle CW
M3 Sx	Spindle RPM = x
M4	Spindle CCW
M4 Sx	Spindle RPM = x
M5	Spindle stop
M6 Tx	Tool number x
M17	End of procedure
M29	End of procedure

Table 2: List of G-Codes Supported by IRBCAM.

For arcs/circles, the additional parameters I,J,K or R are used. For example:

```
G1 X100          ; arc start point
G2 X90Y10I-5J5   ; arc end point and relative offsets (I,J) to centre point
```

In absolute circle centre mode (G161), the same example will be:

```
G161 G1 X100     ; arc start point
G2 X90Y10I95J5   ; arc end point (X,Y) and centre point (I,J)
```

The same example with radius (R) would be:

```
G1 X100          ; arc start point
G2 X90Y10R7.07   ; arc end point and radius
```

In 5-axis mode, IRBCAM currently supports the following notation:

```
G1 X0Y0Z0 I0J0K1   ; where I,J,K define the tool orientation
G1 X0Y0Z0 A0B0C0   ; where C,B,A are axis rotations (transformations)
```

The following example shows two different approaches to 5-axis G-Code program-

ming. The first approach is to change the tool orientation using the (I,J,K) parameters. These three numbers define the normal vector of the tool.

Example 1: Using tool vector

```
G1 X0
G3 I50
G1 X100
G1 X100I1J0K0
; Tool in X-direction
G2 K-50
```

Example 2: Using axis rotation

```
G1 X200I0J0K1
; Tool in Z-direction
G3 I50
G1 X300
G1 X0Z300B90
; Rotate 90° about Y
G2 I50
```

The combined program (example 1 and 2) is illustrated in Figure 6.

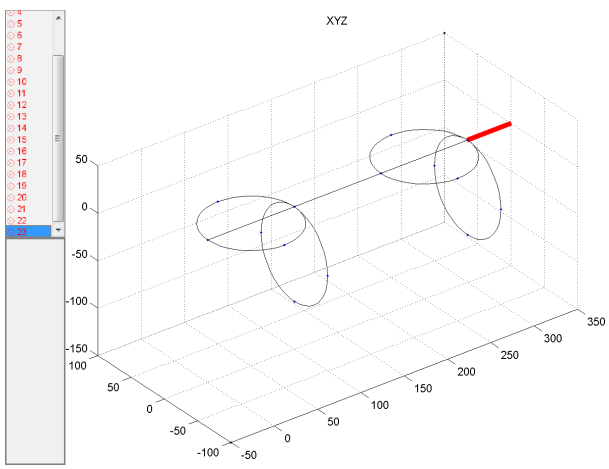


Figure 6: Illustration of two approaches for 5-axis G-Code programming.

Notes:

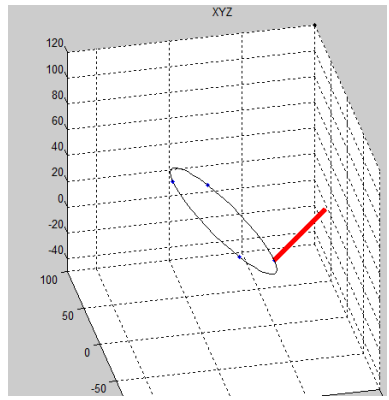
The tool vector is usually normalised to have a length equal to one. If the user specified a tool vector (I,J,K) with a length different to one, IRBCAM will automatically re-scale (normalise) the vector.

The circle plane is always normal to the (I,J,K) vector. When using axis rotations (A,B,C), the circle plan is also always normal to the (I,J,K) vector, but note that the tool vector is rotated by (A,B,C).

The G-code G18 is the same as A90C90, while G19 is the same as C-90.

Axis rotations (A,B,C) can be combined with tool vector orientation (I,J,K). The following G-code and figure is an example:

```
G1 X0Z300B90 ; Rotate 90° about Y
G1 I-1K1 ; Re-orient the tool
G2 I-35K-35 ; Full-circle
```



Example drill cycle:

```
G1 Z3
G81 G98 X4 Y5 Z1.5 R2.8 F33
G80
```

The following moves will take place.

1. a traverse parallel to the XY plane to (4,5,3)
2. a traverse parallel to the Z-axis to (4,5,2.8)
3. a feed parallel to the Z-axis to (4,5,1.5), feedrate F33
4. a traverse parallel to the Z-axis to (4,5,3)

Without the parameter G98, the 4th move will retract to (4,5,2.8).

5.3 Text Files

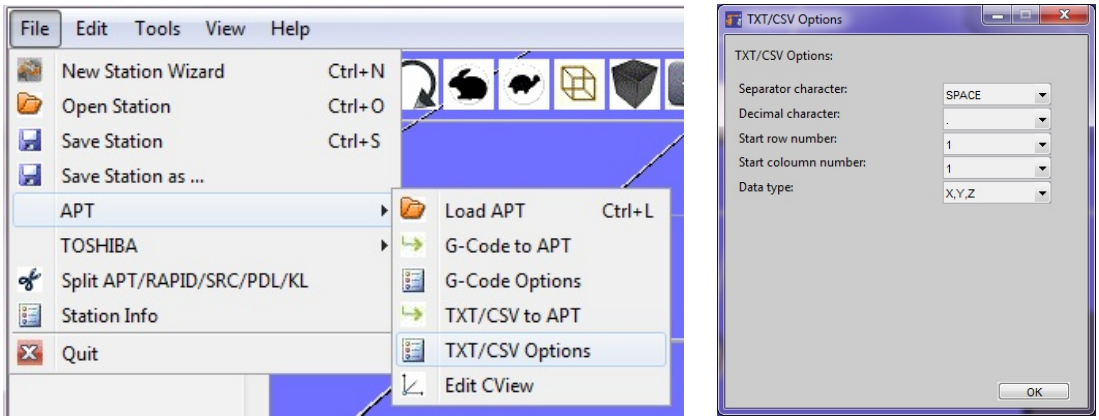


Figure 7: Text file import and options.

IRBCAM can also import pure ASCII text files or CSV files from Excel. Fig. 7 shows the menu (left) and the TXT/CSV Options dialog (right). Parameters such as the separator character, decimal character, row and column start numbers can be configured by the user. The data type can either be 3-axis (X,Y,Z) or 5-axis (X,Y,Z,I,J,K), where I,J,K is the tool vector. ASCII file import is also supported by a command-line option, see Table 7.

6 3D Navigation and Options

Figure 8 shows the key combinations for the 3D navigation in IRBCAM. The arrow keys are used, in combination with CTRL, ALT and SHIFT.

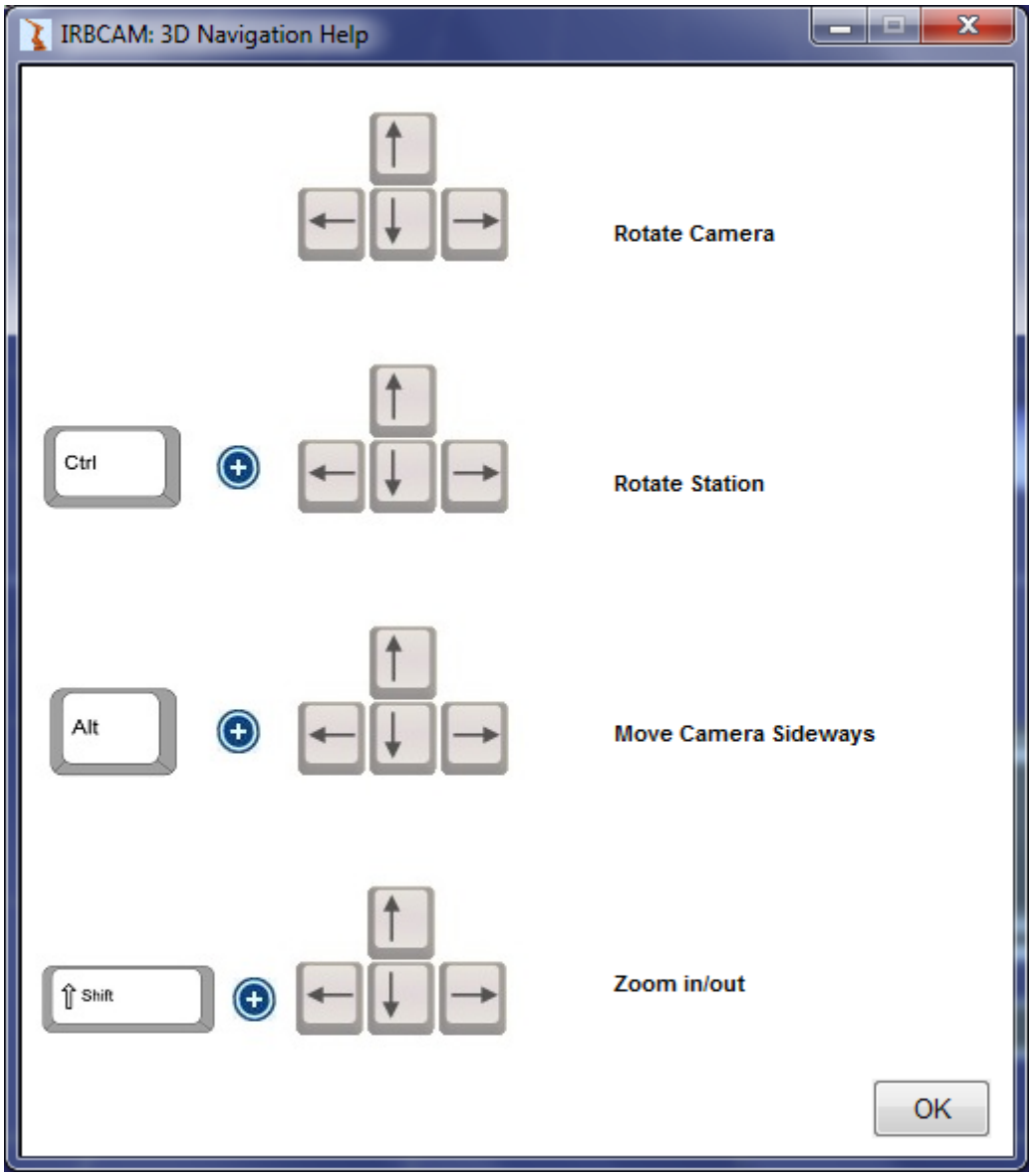


Figure 8: Key combinations for 3D navigation.

The IRBCAM options screen can be accessed from the menu 'Tools - Options' or by pressing CTRL+J and is shown in Fig. 9. First, the user can define a text editor (in this example Notepad++) to view the generated robot code. At the top left there are options related to the display mode of IRBCAM. The first option defines the

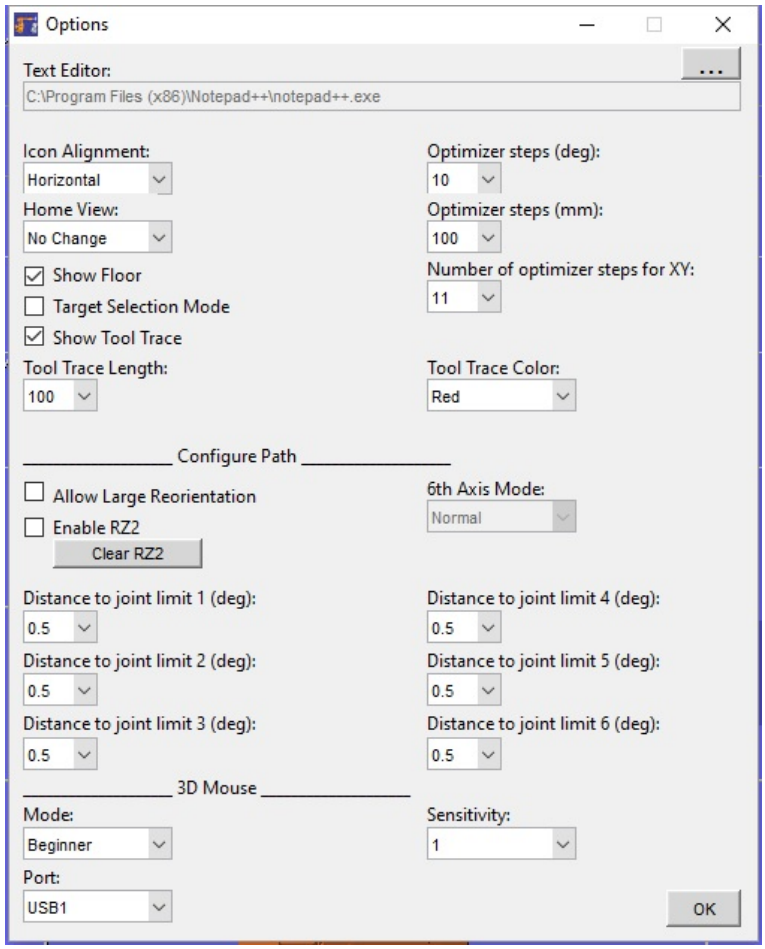


Figure 9: IRBCAM Options.

placement of the navigation icons (horizontal or vertical). The next option defines the home 3D view which can be defined as either No Change, Default or Current. The display of the floor can be turned off. This may be useful when some objects are located below $Z = 0$ or the user would like to view the station from below through the floor. When 'Target Selection Mode' is disabled, the XY, XZ, YZ and XYZ plots can be zoomed. When this option is enabled, the user can click on a single target in the same plots to edit values. IRBCAM will go directly to the robot target closest to the clicked point. The last three display mode options define the tool trace. At the top right there are three options (optimizer steps) which are used to define the behaviour of the Optimizer (accessed through CTRL+I).

The options at the bottom are all related to path configuration. When the option 'Allow Large Reorientation' is enabled tool roll angle is allowed to change by 40 degrees between any two robot targets. The default is only 3 degrees. The next option defines the robot's 6th axis mode. An example using this option is given in section 7.9. The options related to the parameter RZ2 can be used to configure

complex 5-axis paths, see section 7.15 for an example. Finally, there are 6 options (one for each joint) which determine the safety distance to the robot's joint limits. The default value is 0.5 degrees, but this number can be increased to 30 degrees for each individual axis. These could be very useful options to influence how a target is configured. For example, if a station is defined with a linear track, one might want to put a 30 degree additional joint limit constraint on axis 2 to avoid large bending torques on the track. Another useful case is when very long linear moves are programmed. IRBCAM does not interpolate between the points and it could happen that the robot controller reaches a joint limit half-way between the points. In such cases, increasing the safety distance to the joint limits could solve the problem.

Starting from Build 325 (2016) IRBCAM supports a 3D space mouse from 3DConnexion. Three new options in Fig. 9 are related to this input device. Mode can be either Beginner or Expert. In Beginner mode the space mouse only responds to the dominant degree of freedom given by the user. In Expert mode multiple degrees of freedom can be manipulated simultaneously. Sensitivity (1,2 or 3) indicates how fast the 3D display responds to the user's input. Sensitivity equal to 1 gives the slowest response. The third parameter specifies which port the space mouse is connected to (USB1 $\cdots$ USB4 or COM1 $\cdots$ COM4). Fig. 10 shows the three different status icons for the space mouse. These icons appear together with the other navigation icons at the top left of the IRBCAM 3D display window. To enter the space mouse mode, the user has to click the icon with the white background (left hand symbol in Fig. 10). If a space mouse is connected to the computer and the settings in IRBCAM Options are correct, then the space mouse icon will change to a green background, indicating translation mode. The 3D display can then be translated in the X,Y and Z directions by using the space mouse. See for example https://www.youtube.com/watch?v=866O6_ItJPs. To enter rotation mode, the user has to click button number one on the space mode. In rotation mode the icon changes to a yellow background (right hand symbol in Fig. 10). Even if a user could potentially manipulate both translation and rotation at the same time using a space mouse, the two modes have been separated in IRBCAM to make navigation easier. By clicking button number one a secod time, the space mouse returns to translation mode. To exit the space mouse mode, the user must press space mouse button number two. The icon will then return to a white background.

Figure 10: 3D Space Mouse: Status Icons. Left: Space mouse not activated. Middle: Space mouse activated in translation mode. Right: Space mouse activated in rotation mode.

7 Examples

This section contains several examples of how to define a station, use IRBCAM as an offline programming tool or how to load an APT file or G-code, configure the path and generate robot code. The examples include a single robot, a robot with a rotary table, a robot mounted on a linear track and the combined setup including both a linear track and a rotary axis. Unless explicitly stated, the units used in the examples are all metric (mm).

7.1 Offline Programming with IRBCAM

This section demonstrates how IRBCAM can be used as an offline programming tool without the need for an additional CAM software package. The robot used in this example is an ABB IRB6400-2.4-M97.

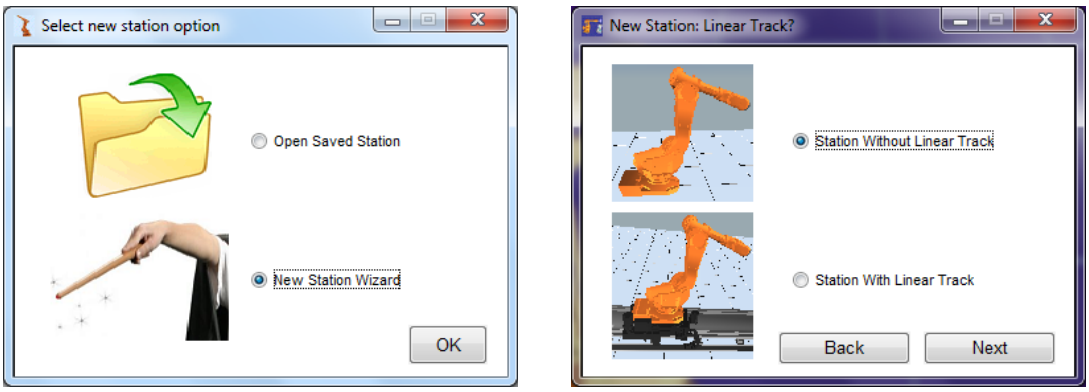


Figure 11: Initial options when creating new station.

Figure 11 shows the initial options. First, choose 'New Station Wizard' then 'Station without Linear Track'. In Figure 11 click 'Next' to proceed to the next stage. At any stage, a selection can be reversed by clicking on 'Back'. Figure 12 shows the robot and tool selection. Select the ABB IRB6400-2.4-M97 robot then 'Robot Holds Tool'.

Figure 13 shows the tool selection window and the options for external axes. Choose 'Spindle-UBR' followed by 'Station Without External Axis'. Figure 14 shows the definition screen for the user frame. Define the user frame at position X=2000 and Z=1000. Note that the Red, Green and Blue coordinate axes for the user frame will move when the user enters new coordinates. Figure 15 shows the next screens in the station setup process. Select 'Station Without User Object' and 'Add Cube'. Figure 16 shows the cube definition screen. Just leave the cube at the default position with default size in this example. This cube represents for example a machining table. Finally, select 'No More Cubes' as the last step in the station definition process.

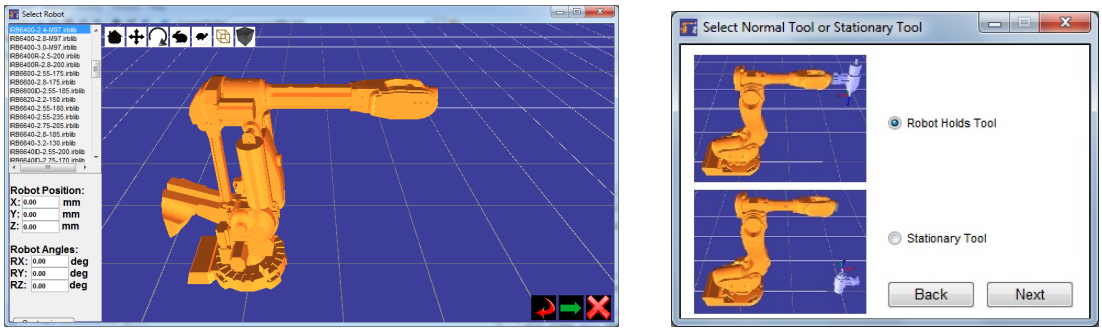


Figure 12: Selection of IRB6400-2.4-M97 robot and type of tool.

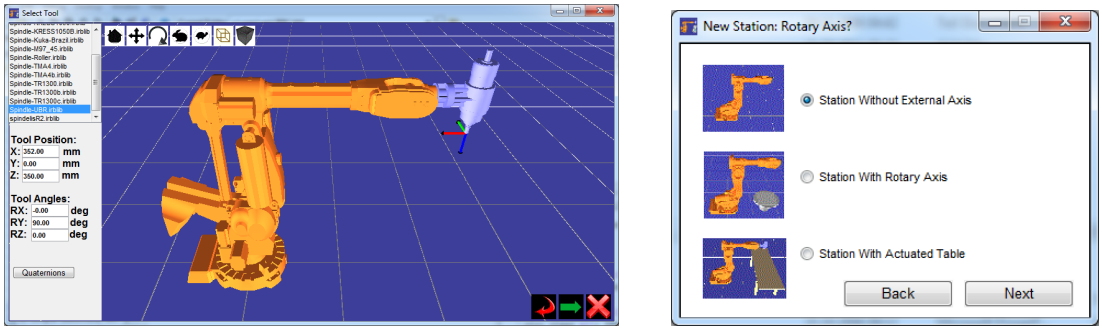


Figure 13: Spindle UBR and no external axis.

Figure 17 shows the main window in IRBCAM after the new station has been saved. This station currently has no target (robot coordinates) defined. The user now has the option to load robot coordinates from an APT file or from G-codes, or to manually enter new coordinates using IRBCAM's offline programming features. In this example, we will demonstrate the offline programming features. First, add a new target as shown in Figure 17 (left). This can be done using the menus (Edit - Target - Add Target) or by pressing CTRL+E. In Figure 17 (right), the new target has been added by default to the position (0,0,0) in the current user frame. To enter the Combination View shown in Figure 17 (right) use the menus (View - Combination View) or press CTRL+2.

When defining the second robot target as shown in Figure 18 (left) the user has to decide if the new point should appear before or after the first target. In this example, add the second target after the first and edit the X-value to be 475, Figure 18 (right).

Next, define the remaining robot targets as illustrated in Figure 19 and in Table 3. In total, there should be 19 targets. Targets 3,6,9,15 and 17 are mid-points in circular motions. To define this attribute, right-click on a target in the target list (upper left-hand side of the main window) and click on the button '»'. The list of attributes will appear for the given target, as illustrated in Figure 20. The attributes are: 1) linear or arc motion, 2) speed, 3) tool number and 4) spindle RPM. Note that only the

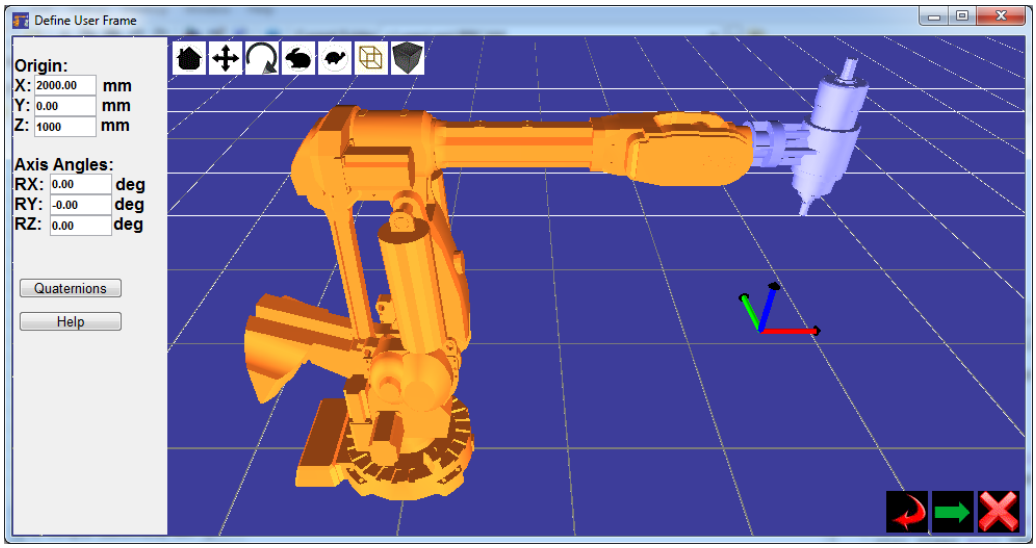


Figure 14: Definition of user frame.

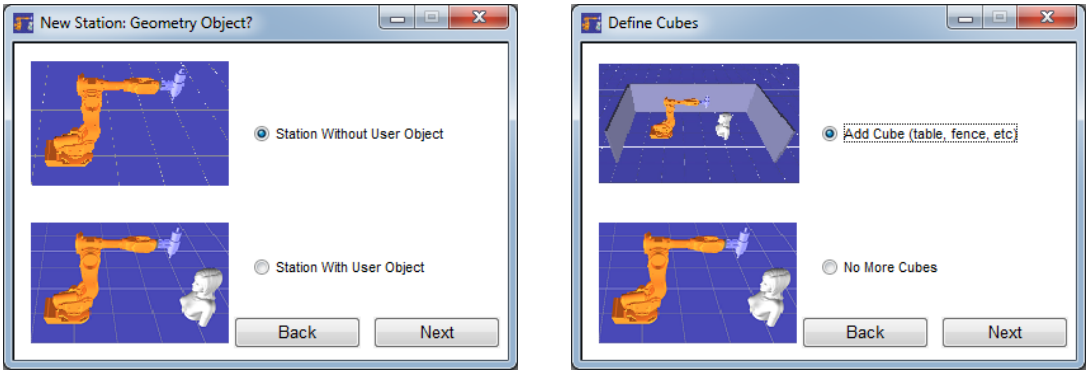


Figure 15: Selection of user object and cubes.

mid-points in the arc are specified, not the start and end-points.

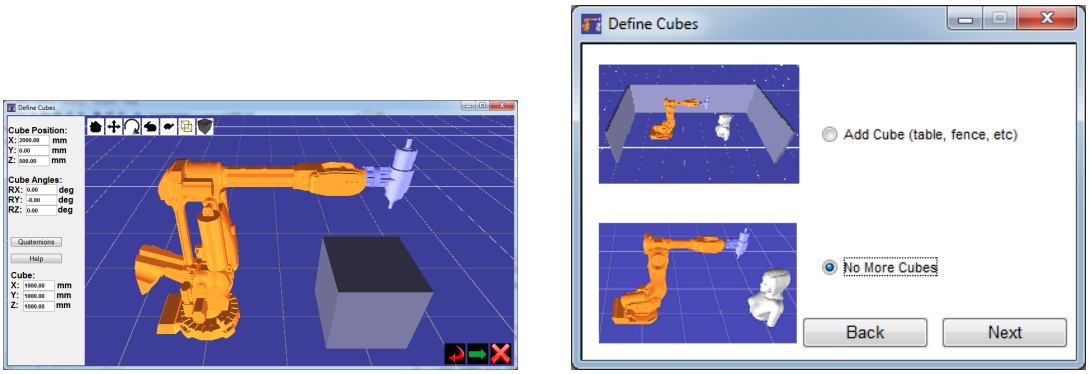


Figure 16: Definition of cube.

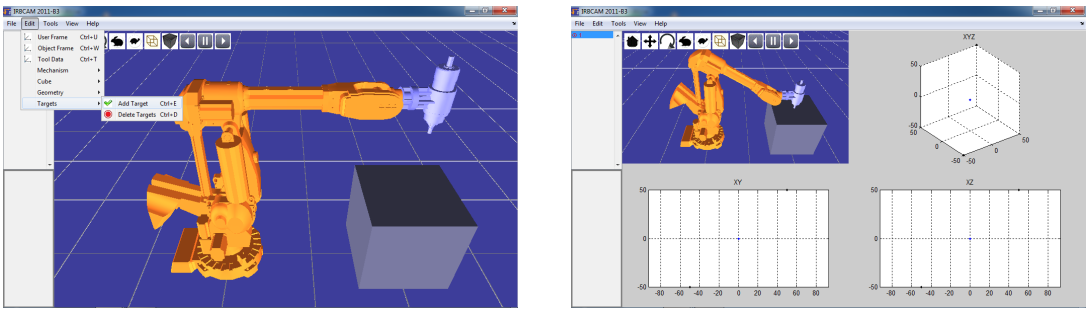


Figure 17: Addition of first target (point).

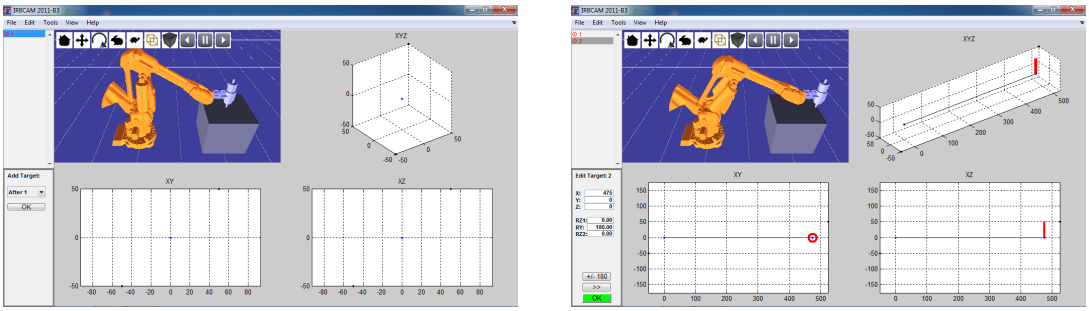


Figure 18: Definition of second robot target.

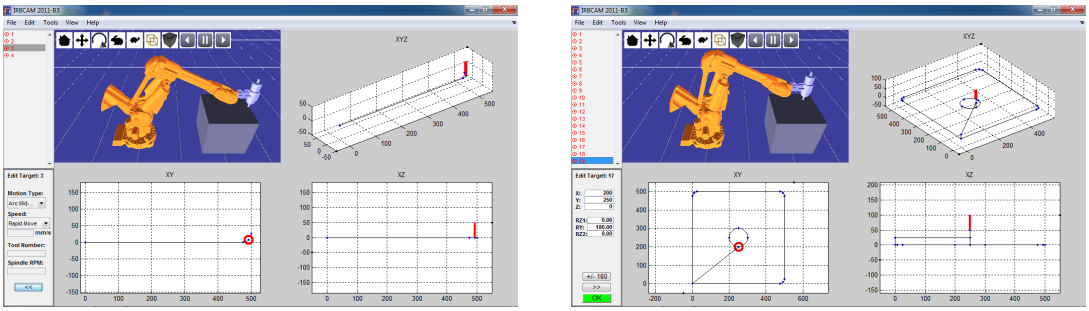


Figure 19: Definition of remaining points.

Point	X	Y	Z	Speed	Arc	Tool	RPM
1	0	0	0	150		1	1000
2	475	0	0				
3	492.68	7.32	0		✔		
4	500	25	0				
5	500	475	0				
6	492.68	492.68	0		✔		
7	475	500	0				
8	25	500	0				
9	7.32	492.68	0		✔		
10	0	475	0				
11	0	0	0				
12	0	0	25	300			
13	250	200	25				0
14	250	200	0	150			1000
15	300	250	0		✔		
16	250	300	0				
17	200	250	0		✔		
18	250	200	0				
19	250	200	50				

Table 3: Coordinate points and attributes for offline programming example.

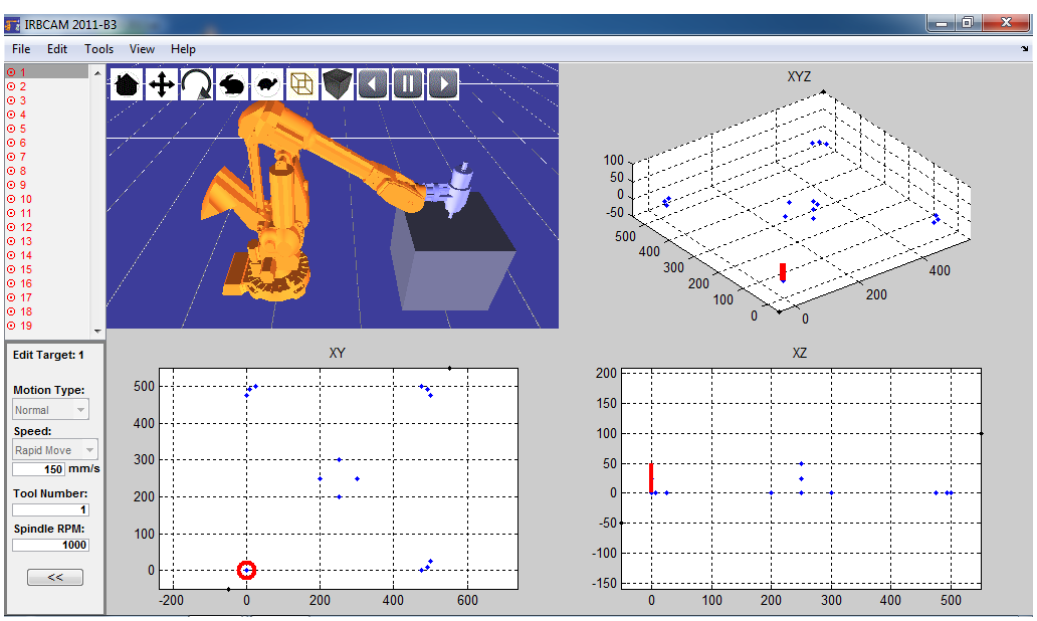


Figure 20: Addition of attributes such as speed, tool number and spindle RPM.

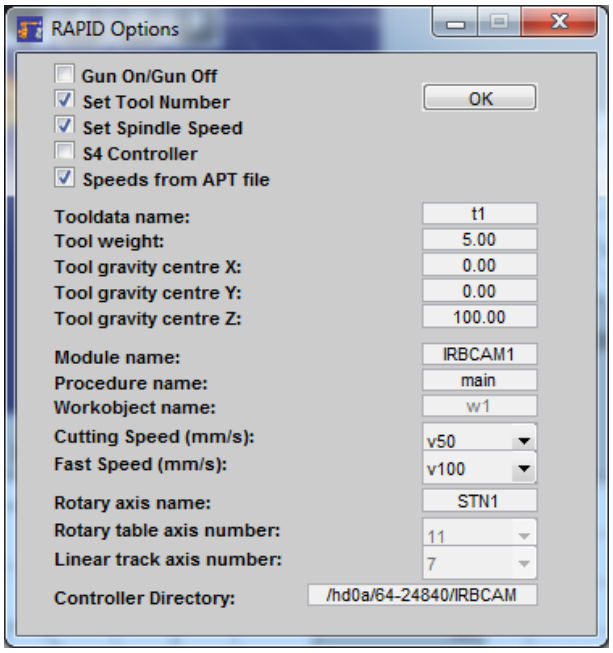


Figure 21: RAPID export options.

Figure 21 shows the export option for the final robot code, for ABB robots the export code is in the RAPID language. In order to enable editing of tool number, spindle RPM and speed for individual robot points (as illustrated in Figure 20), the export options 'Set Tool Number', 'Set Spindle Speed' and 'Speeds from APT file' must be selected, as shown in Figure 21.

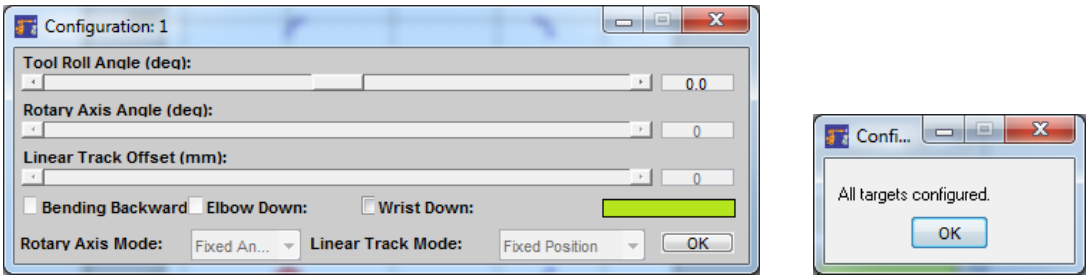


Figure 22: Path configuration.

Before the robot code can be exported, the toolpath must be configured. From the menus, select 'Tools - Configure Path' or press CTRL+K and the configuration window, Figure 22 (left), appears. In this example, just accept the default settings and press OK. If everything goes well, all the robot targets will be configured successfully, as shown in Figure 22 (right). After configuration, the RAPID code can be exported. The final output should look like the code in Figure 23. Note the function calls for spindle speed and tool number. These functions are hardware

specific, and must be written by the user to interface with the correct digital and analog input/output signals. Note also that both linear (MoveL) and circular moves (MoveC) appear in the RAPID code, and that the speed varies between 150mm/s and 300mm/s.

```
PROC main()
  MoveAbsJ [[-0.000,2.409,31.957,-0.000,-31.957,0.000],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v100,fine,t1;
  SetSpindle 1000;
  LoadTool 1;
  MoveL [[0,0,0],[0,0,1,0],[-1,-1,0,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=150,z0,t1\WObj:=w1;
  MoveL [[475,0,0],[0,0,1,0],[-1,-1,0,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=150,z0,t1\WObj:=w1;
  MoveC [[492.68,7.32,0],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],[[500,25,0],[0,0,1,0],[0,0,-1,0],[9E+
  MoveL [[500,475,0],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=150,z0,t1\WObj:=w1;
  MoveC [[492.68,492.68,0],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],[[475,500,0],[0,0,1,0],[0,0,-1,0],[
  MoveL [[25,500,0],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=150,z0,t1\WObj:=w1;
  MoveL [[7.32,492.68,0],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=150,z0,t1\WObj:=w1;
  MoveL [[0,475,0],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=150,z0,t1\WObj:=w1;
  MoveL [[0,0,0],[0,0,1,0],[-1,-1,0,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=150,z0,t1\WObj:=w1;
  MoveL [[0,0,25],[0,0,1,0],[-1,-1,0,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=300,z0,t1\WObj:=w1;
  SetSpindle 0;
  MoveL [[250,200,25],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=300,z0,t1\WObj:=w1;
  SetSpindle 1000;
  MoveL [[250,200,0],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=150,z0,t1\WObj:=w1;
  MoveC [[300,250,0],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],[[250,300,0],[0,0,1,0],[0,0,-1,0],[9E+9,9
  MoveC [[200,250,0],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],[[250,200,0],[0,0,1,0],[0,0,-1,0],[9E+9,9
  MoveL [[250,200,50],[0,0,1,0],[0,0,-1,0],[9E+9,9E+9,9E+9,9E+9,9E+9,9E+9]],v50\V:=150,z0,t1\WObj:=w1;
ENDPROC
```

Figure 23: Final robot code (RAPID).

Note that while user and tool frames in IRBCAM use Roll-Pitch-Yaw angles (Z-Y-X), the coordinate points use Z1-Y-Z2 angles. The APT file or the user defines the Z1 and Y angles, while the Z2 angle is defined by IRBCAM when configuring the path. The angle Z2 equals the Tool Roll Angle which is specified in the path configuration, Figure 22 (left). The user decides a value for Z2 only for the first target. IRBCAM will then decide the Z2 angles for all the remaining targets based on the user's initial choice for the Z2 angle. In the example in this section, only 3-axis coordinates were defined. In such cases, the angles are Z1=0 and Y=180 degrees.

Fig. 24 (left) shows the IRBCAM statistics screen (Tools - Statistics or press CTRL+Z). The figure shows that there are 14 linear moves and 5 arc moves. The shortest linear move is 25mm and the longest linear move is 475mm. The total length of the toolpath is 2.70m and the estimated machining time about 17 seconds. The maximum dimensions are 500mm in both X and Y directions and 25mm in the Z direction. At this stage, you may want to save your station with 'File - Save Station' (or press CTRL+S).

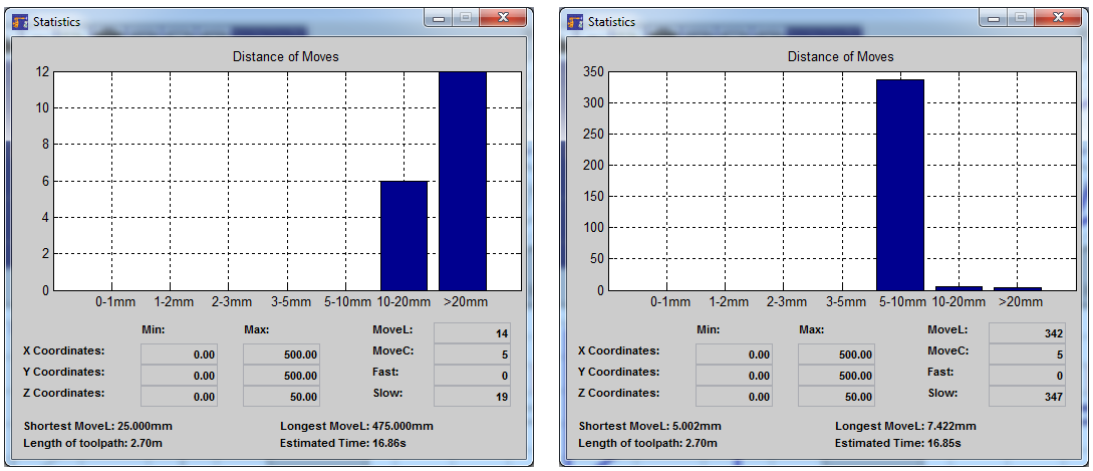


Figure 24: IRBCAM statistics (left: current, right: after maximum distance operation).

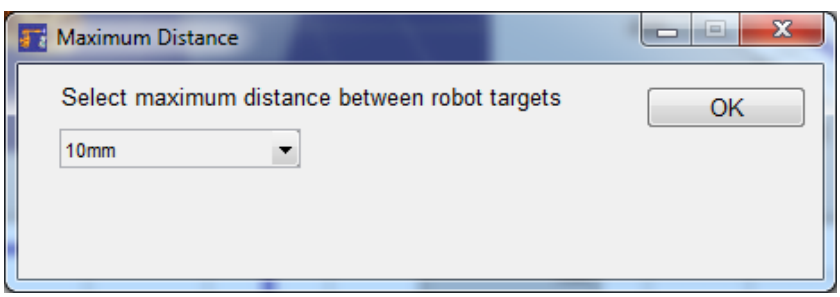


Figure 25: Maximum distance parameter.

Although the RAPID code exported in Fig. 23 will run without any problems on an IRB6400-2.4-M97 robot, in the remainder of this example the maximum distance function in IRBCAM will be demonstrated. This function can reduce the longest linear move which the robot has to make. Select 'Tools - Maximum Distance' (or press CTRL+M) and the window in Fig. 25 will appear. Select maximum distance equal to 10mm and press OK. After the operation is complete, select the statistics window again (CTRL+Z) and the window in Fig. 24 (right) should appear. Now the toolpath contains 342 linear moves and 5 circular moves (note: the circular moves

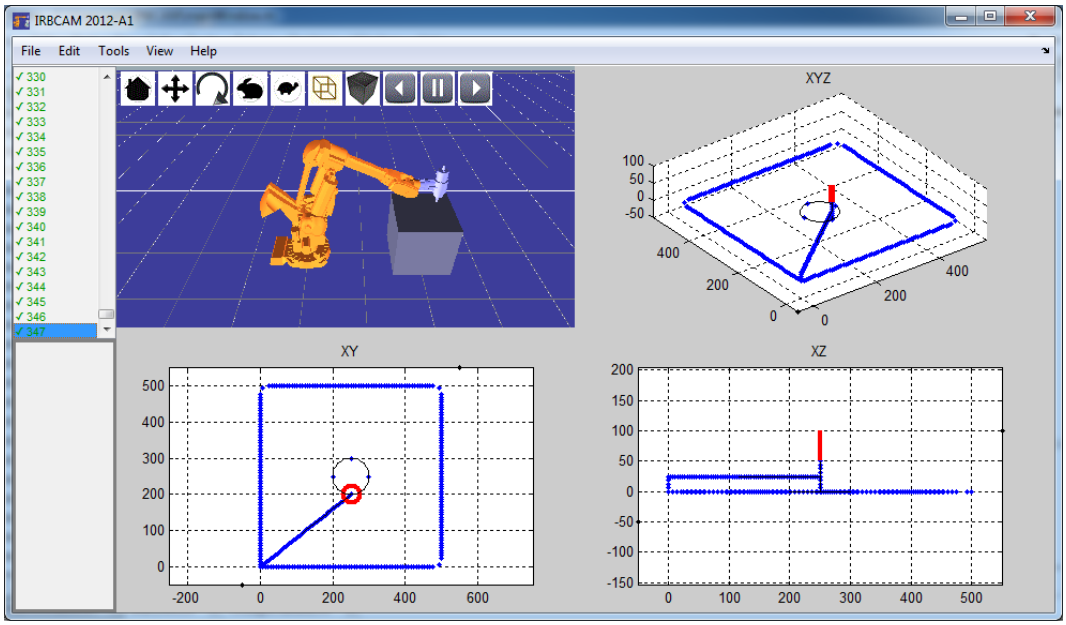


Figure 26: IRBCAM station after maximum distance operation.

are not affected by the maximum distance operation. The bars in Fig. 24 (right) which show moves longer than 10mm represent the circular moves). The shortest linear move has now become 5mm while the longest move 7.422mm. The other parameters in the statistics window are as before.

Fig. 26 shows the new IRBCAM main window after the new toolpath has been configured. As seen in this example, the maximum distance operation can significantly increase the number of robot targets. In this example, it is recommended to only export the 19 robot targets to the controller (as in Fig. 23). The maximum distance function can be used when the controller reports problems with long linear moves. Such problems can occur if the robot has to change configuration during the linear move, or if the path goes near a singularity of the robot. With shorter maximum moves, IRBCAM has more options available to avoid such problems.

In this example the maximum distance function has been demonstrated for a 3-axis toolpath. For 5-axis toolpaths, in addition to the X,Y,Z values, the two rotational angles will also be interpolated. Hence, for a 5-axis toolpath with very large rotational changes (90° or higher), the maximum distance function can be used to make the rotational changes in the toolpath smaller and hence easier to configure.

7.2 FANUC R1000iA-80F Robot and G-Code Programming

IRBCAM can make robot programs from G-codes. In this section the G-code in Table 4 will be used as an example. First, create a new station using the 'New Station Wizard'. Select a FANUC-R1000iA-80F robot and the Spindle-TMA4 and accept the default settings. Place the user frame at X=2000, Y=0, Z=1000.

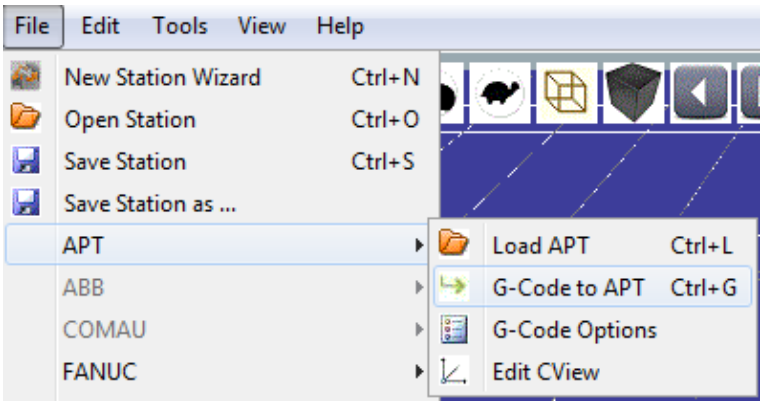


Figure 27: G-Code to APT Conversion.

The G-code in Table 4 is available in the IRBCAM distribution files. It is saved in the 'C:\Program Files\IRBCAM\apt' directory with the filename 'ABBA.nc'. When loading G-codes into IRBCAM, the first step is to convert the G-codes to the APT-CL format. In the menus, select 'File - APT - G-code to APT', or press CTRL-G, see Figure 27. Select 'ABBA.nc' and save the converted APT file to for example 'ABBA.apt'. Then, IRBCAM will load the converted APT file (accept the default settings when loading the APT). Next, go to 'View - XYZ' or press CTRL+6 and the screen shown in Figure 28 should appear.

Note that because of the code G19, the toolpath is placed in the YZ-plane. As can be seen in Figure 28, the red tool vector points in the negative X-direction. This direction will be very difficult for the robot to reach, hence let us turn the user frame by 180° about the Z-axis. Select 'Edit - User Frame' or press CTRL+U and set RZ=180. Click on 'OK'. Next to the spindle in Figure 29, you can now see that the X-axis (red colour) of the user frame points towards the robot, whereas the default direction is away from the robot. After this operation, it is no problem for the robot to orient the tool in the negative X-direction.

Finally, select 'Tools - Configure Path' or press CTRL+K. Set the Tool Roll Angle to 90° and click 'OK'. The entire toolpath should then be configured and the list of coordinates on the left-hand side of the screen should change from red to green colour. Select 'View - Combination View' or press CTRL+2. The screen should then look like Figure 29. After this, the robot code can be generated by selecting 'FANUC - Save Karel' in the menus.

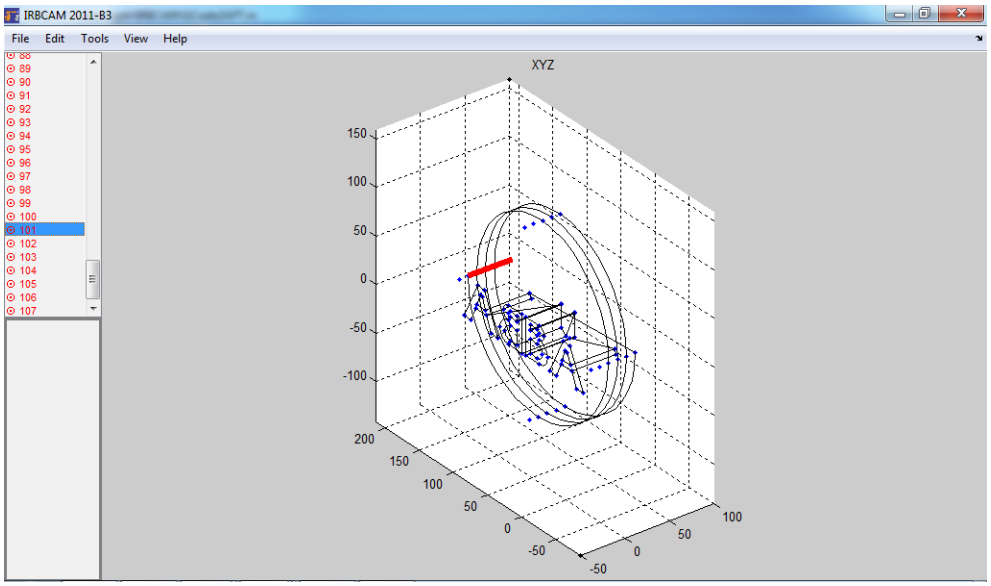


Figure 28: 3D View of Imported G-Codes.

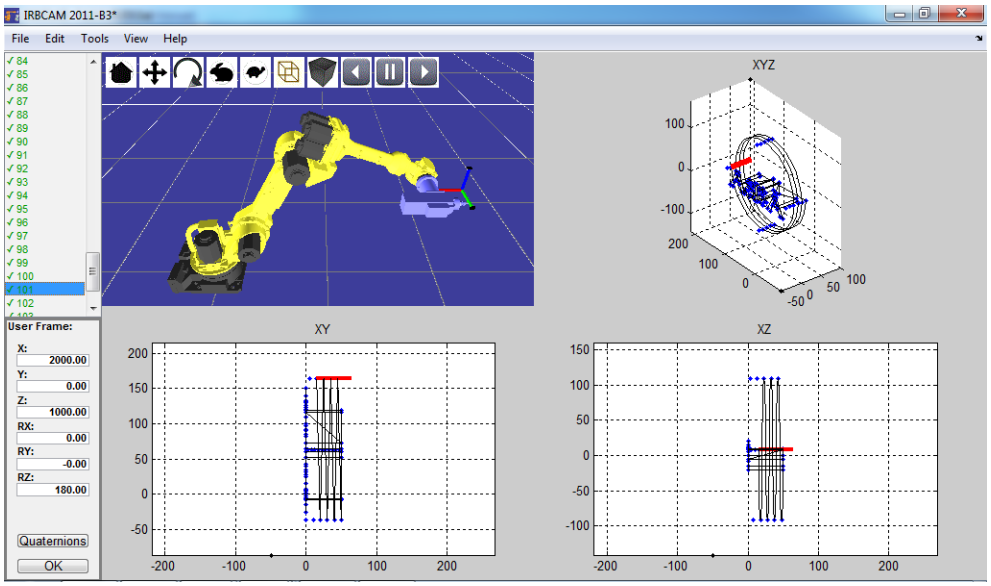


Figure 29: Configured Toolpath Based on G-Code.

```
; This program uses:
; G20 = mirror about X
; G23 = No mirroring
; G92 = Translate origin
; Others (G21 Mirror Y, G22=X&Y)

G71 ; Metric units
G90 ; Absolute Programming
F3000 ; Feedrate 3000 mm/min
G19 ; Transformation XY to YZ
M4 S1000 ; Spindle on, 1000 RPM
M6 T1 ; Tool number 1

%L A ; Sub-routine for letter A
G0X-8Y-5 ; RAPID move
G1Z0 ; Slow (contact) move
G1X8
G1X15Y-20
G1X25
G1X5Y20
G1X-5
G1X-25Y-20
G1X-15
G1X-8Y-5
G0Z50
G0X-6Y0
G1Z0
G1X6
G1X1.5Y11.5
G1X-1.5
G1X-6Y0
G1Z50
% ; End sub-routine
```

```
%L B ; Sub-routine for letter B
G0X-25Y-20Z50
G1Z0
G1Y20
G1X-5
G2Y5J-7.5 ; Clockwise arc
G2Y-20J-12.5
G1X-25
G0Z50
G0X-17.5Y-15
G0Z0
G1Y0
G1X-7.5
G2Y-15J-7.5
G1X-17.5
G0Z50
G0X-17.5Y10
G0Z0
G1Y15
G1X-7.5
G2Y9J-3
G1X-17.5
G0Z50
% ; End sub-routine

LL A ; Call A
G20 G92X35Y0 LL B ; Call B
G23 G92X90Y0 LL B ; Call B
G92X125Y0 LL A ; Call A

G92X0Y0
G0X-36
G3Z0I100K10 ; Helix, pitch 10mm
G0Z50
M5 ; Spindle off
```

Table 4: Example G-code program with function calls, translation and mirroring.

7.3 ABB IRB2400 Robot with no External Axes

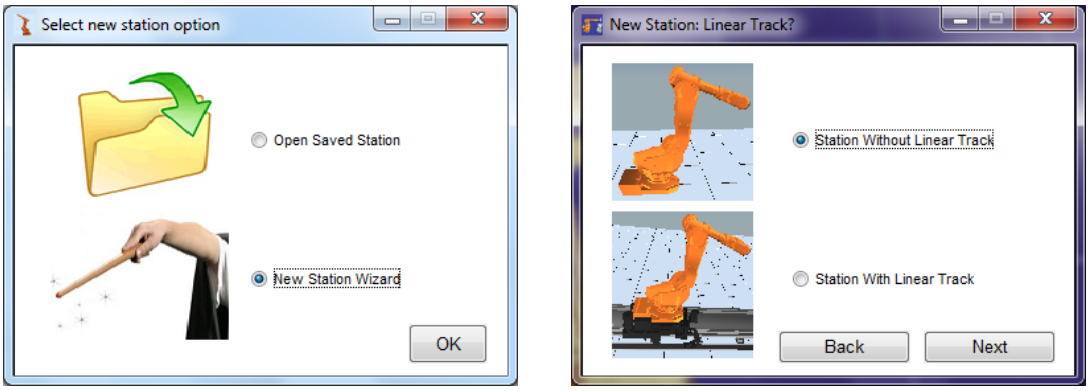


Figure 30: Initial options when creating new station.

Figure 30 shows the initial options. First, choose the 'New Station Wizard' followed by 'Station without Linear Track'. In Figure 30 (right) click on 'Next'. At any stage, a selection can be reversed by clicking on 'Back'.

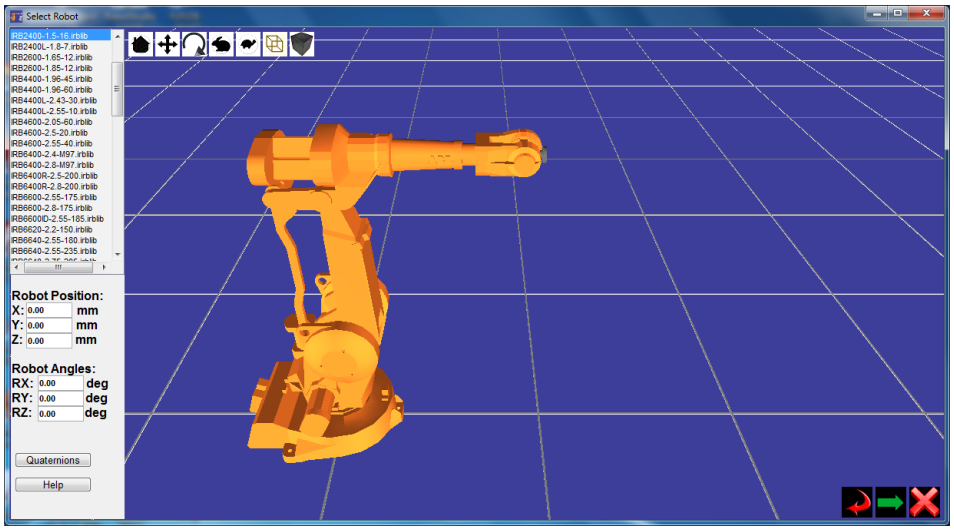


Figure 31: Selection of IRB2400-1.5-16 robot.

Figure 31 shows the library of ABB robots. Select the IRB2400-1.5m-16kg robot. Leave the robot in the default position (0,0,0) with no rotations. Click on → (the green arrow icon in the bottom right-hand corner). Note that the position of the icons (horizontal or vertical) can be defined in the options screen (CTRL+J) in the main IRBCAM window which appears after the station has been defined. Fig-

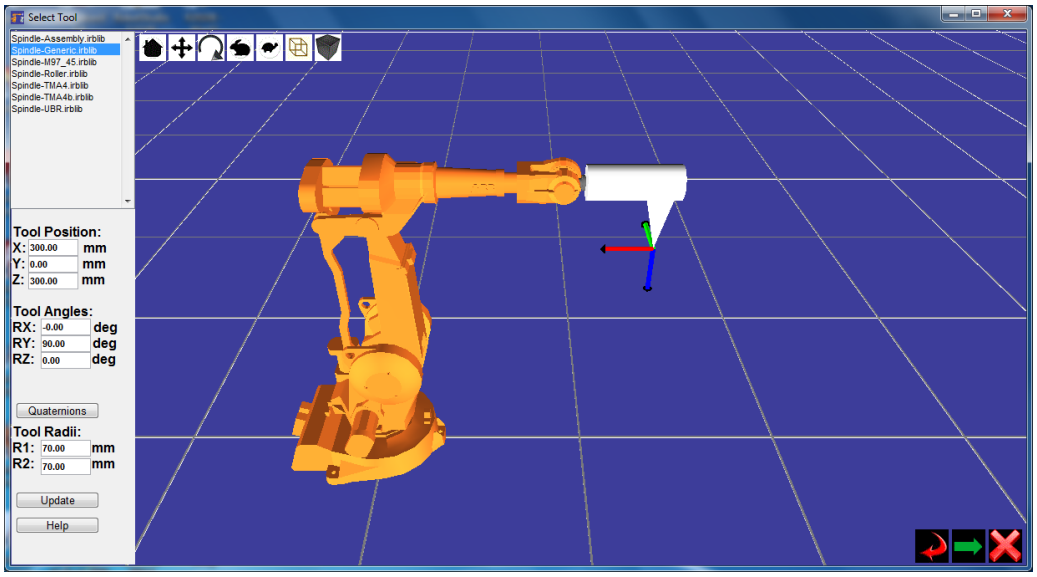


Figure 32: Selection of generic tool.

ure 32 shows the library of robot tools. In this example, the generic tool will be used. Accept the default settings for the tool which are ($X = 300, Y = 0, Z = 300$) and rotation $RY = 90^o$. The Tool Radii can be modified to change the graphical appearance of the generic tool. These values are just for the 3D graphical visualisation and they have no impact on the generated toolpath and RAPID code. In the

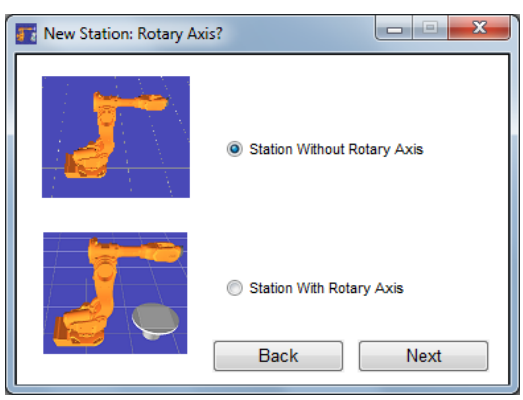


Figure 33: Selection of rotary axis.

dialog box shown in Figure 33 select no rotary table for this example. In Figure 34

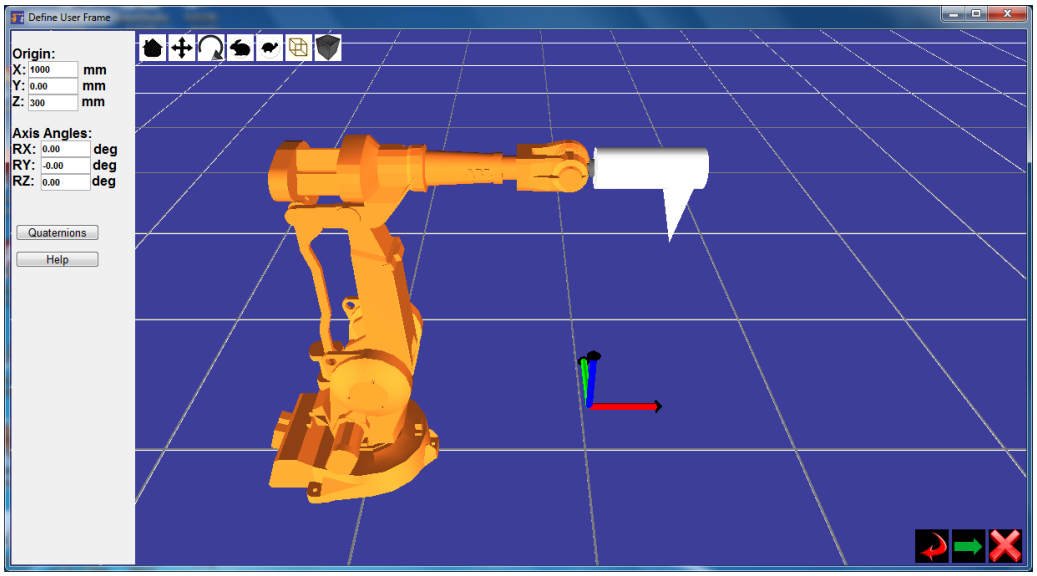


Figure 34: Definition of user frame.

the user frame (UFRAME) is defined. In this example the following values are used: $X = 1000mm, Y = 0, Z = 300mm$ and no rotations. In Figure 34 the X-axis of the coordinate frame is shown with Red colour, the Y-axis with Green colour and the Z-axis with Blue color (one way to remember the colour scheme is XYZ=RGB). Note that only the user frame (UFRAME) is defined in this step, not the object frame (OFRAME). Figure 35 shows the next two options. In this example, no user geom-

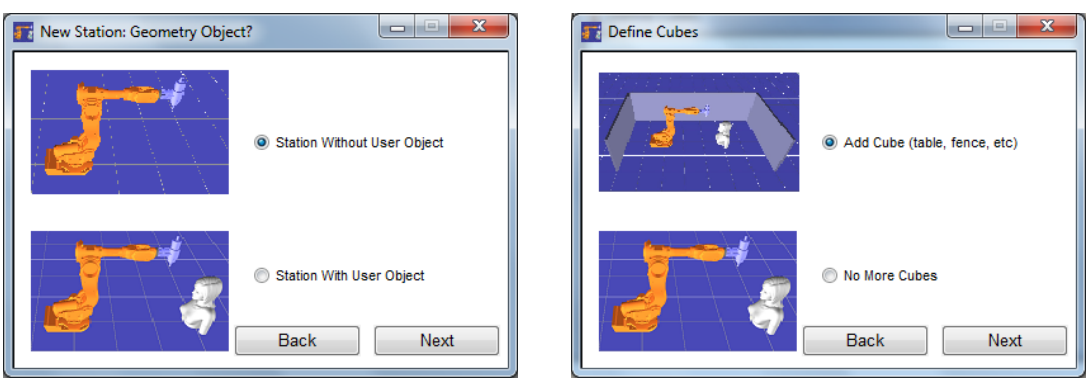


Figure 35: Selection of user object and cubes.

etry (CAD file) is loaded into the station. The next option (Cube definition) allows simple cubical objects to be put into the station. These objects could represent machining tables, fences, control cabinet or other obstacles. Figure 36 shows the definition of the machining table by using a cubical object. The centre position of the cube equals $X = 1000mm, Y = 0, Z = 150mm$ with no rotations. The length, width and height of the cube are defined as $1000mm, 1000mm, 300mm$, respec-

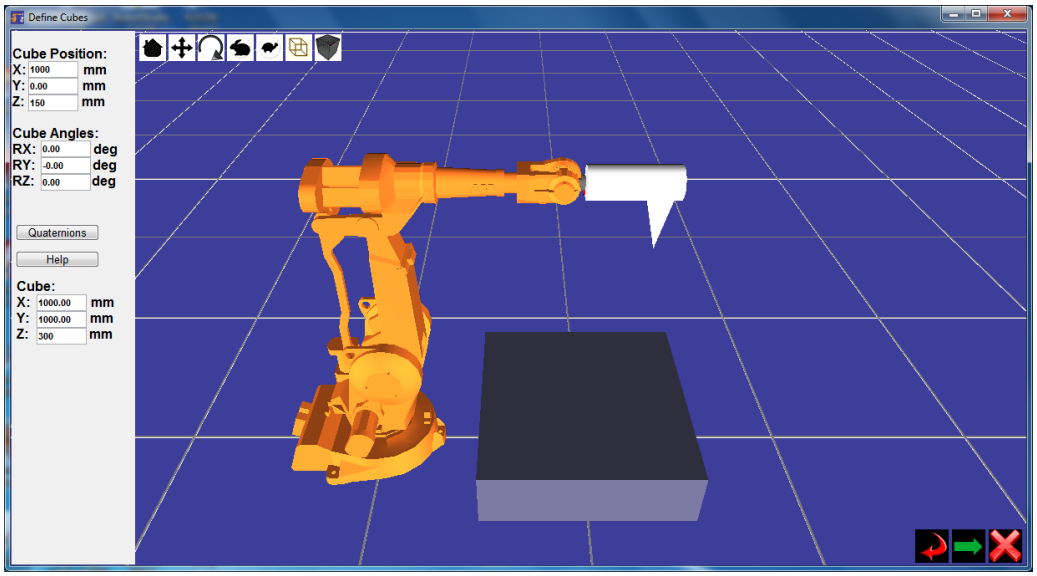


Figure 36: Definition of cube.

tively. Note that the position of the cube equals the mid-point of the cube. Hence, when the position is set to $150mm$ and the height is set to $300mm$, then the cube strethces from the floor at $0mm$ to a height of $300mm$, which in this case equals the Z-position of the user frame defined in the previous step. It is possible to have

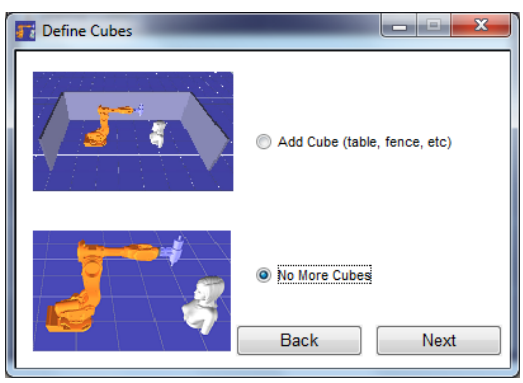


Figure 37: Ending the selection of cubes.

as many cubical objects as you like. When all the cubes are defined, Figure 37 shows how to proceed. After this step, the station can be saved as an '*.irb' file, in this case for example 'irb2400.irb'.

After the station has been defined and saved, an APT file can be loaded. The file 'example.apt' is included in the installation directory 'C:\Program Files\IRBCAM\apt\Example.apt'. Choose 'File - APT - Load APT' from the menu (or press CTRL+L). When loading an APT file, the minimum distance between the robot coordinates must be defined,

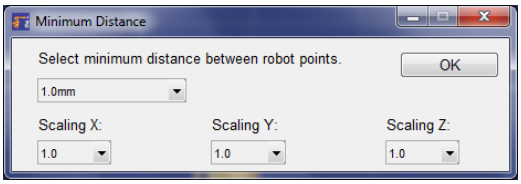


Figure 38: Minimum distance between points.

see Figure 38. Accept the default setting of $1.0mm$. Note that this function only removes a coordinate if there is no re-orientation of the tool from the previous coordinate. Also leave the scaling factors for X,Y and Z equal to the default settings of $1.0mm$.

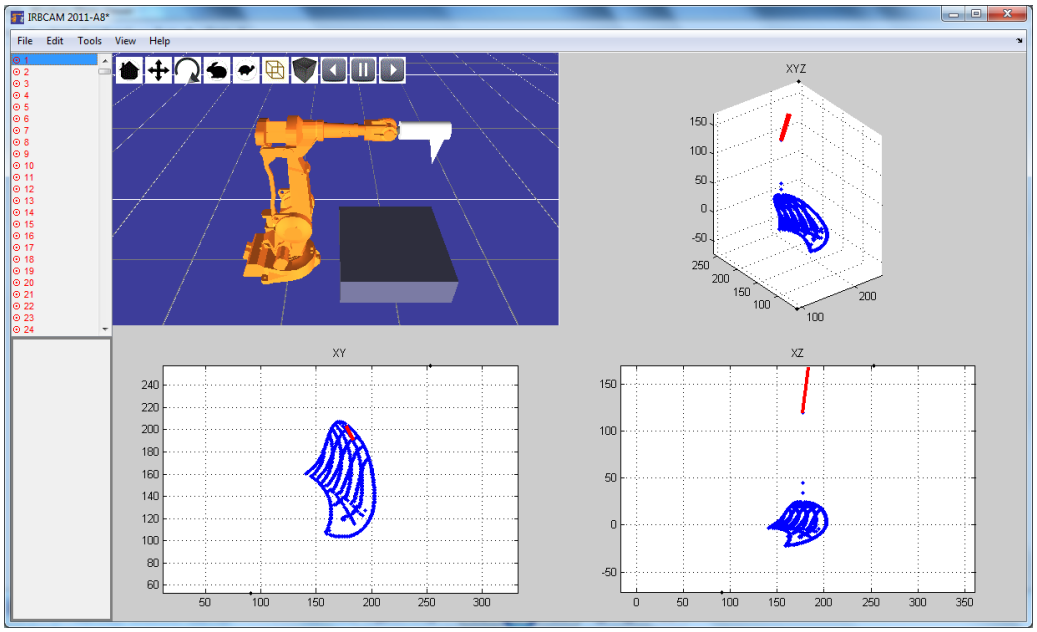


Figure 39: Load APT file into station.

After the APT file has been loaded into the station, choose 'View - Combination View' (or press CTRL+2) and the screen should look like Figure 39. The top left window shows the robot, the tool and the machining table. The other three windows show the toolpath in XYZ, XY and XZ views. To the left, the list of robot targets (positions) is displayed. In this example, there should be a total of 966 targets. Note that these targets are displayed with a red colour, which means that they are not yet configured.

In the menus, select 'Tools - Configure Path' (or press CTRL+K). The screen should then look like Figure 40. At the bottom of the screen, the configuration window appears. In this example, since there is no rotary or linear axis, the tool roll angle is the only parameter which can be adjusted. Try moving the slider for the tool roll

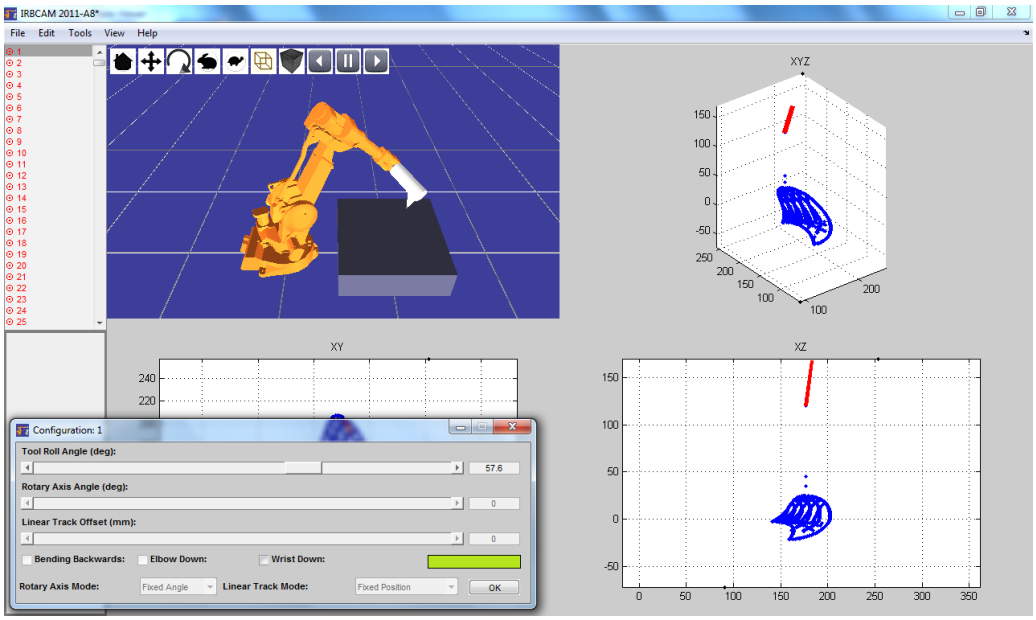


Figure 40: Path Configuration.

angle, and you will see how the robot adjusts its configuration accordingly. You can also click the check-box for 'Wrist down' and note how the robot's wrist changes accordingly. Finally, choose a tool roll angle of 57.6 degrees and press the OK button.

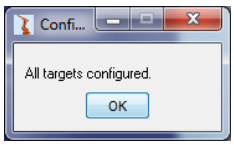


Figure 41: Targets configured.

All targets should in this example be configured, and the message shown in Figure 41 should appear. At the same time, the colour in the robot target list at the left of the screen should change from red to green. You can now use 'View - Animate Forwards' (or CTRL+F) or 'View - Animate Backwards' (or CTRL+B) to verify the toolpath. Select any function from the menus to abort the animation.

Next, select 'File - RAPID - RAPID Options' and the window in Figure 42 will appear. In this case, just accept the default options. These options will be explained in more detail in section 10.

Finally, to generate the RAPID code, select 'File - RAPID - Save RAPID' (or press CTRL+R). Select a filename (you can choose '*.prg' or '*.mod' extension), and the example is completed. This RAPID code can now be transfered to the robot controller and executed.

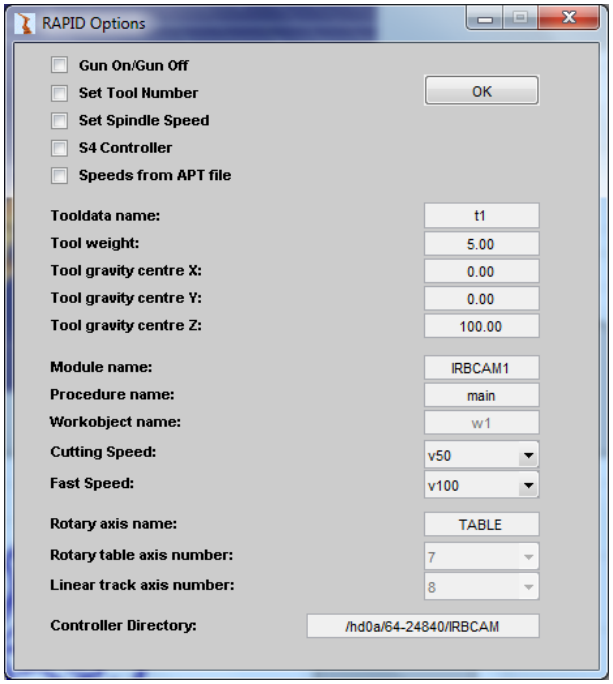


Figure 42: RAPID export options.

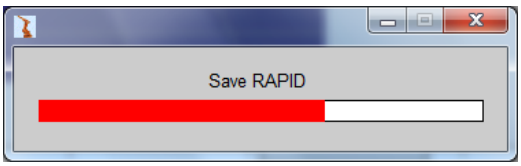


Figure 43: Save RAPID code.

7.4 ABB IRB6600 Robot Mounted on a Linear Track

This example will demonstrate robotic milling of a larger object by using an IRB6600-2.8m-175kg robot mounted on a linear track.

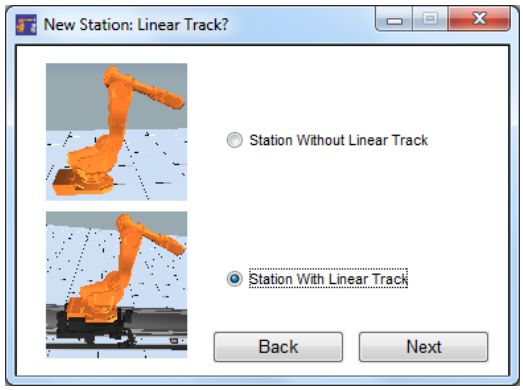


Figure 44: Option for selecting linear track.

Figure 44 shows the initial option in the 'New Station Wizard'. Select 'Station With Linear Track' and click 'Next'.

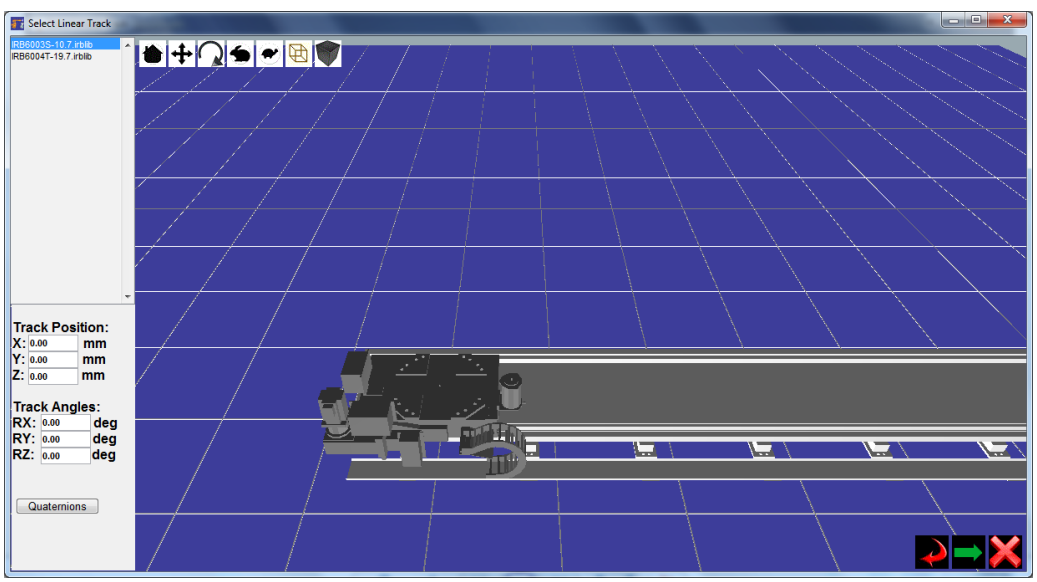


Figure 45: Linear track library.

Figure 45 shows the library of linear tracks. Choose the track IRB6003S-10.7 with default settings. This track has a length of $10.7m$. Click on $\rightarrow$ to continue to the robot selection.

Figure 46 shows the library of robots. Choose the IRB6600-2.8-175 with default settings and click $\rightarrow$. Note that the Z-position is by default set to $442mm$ which is

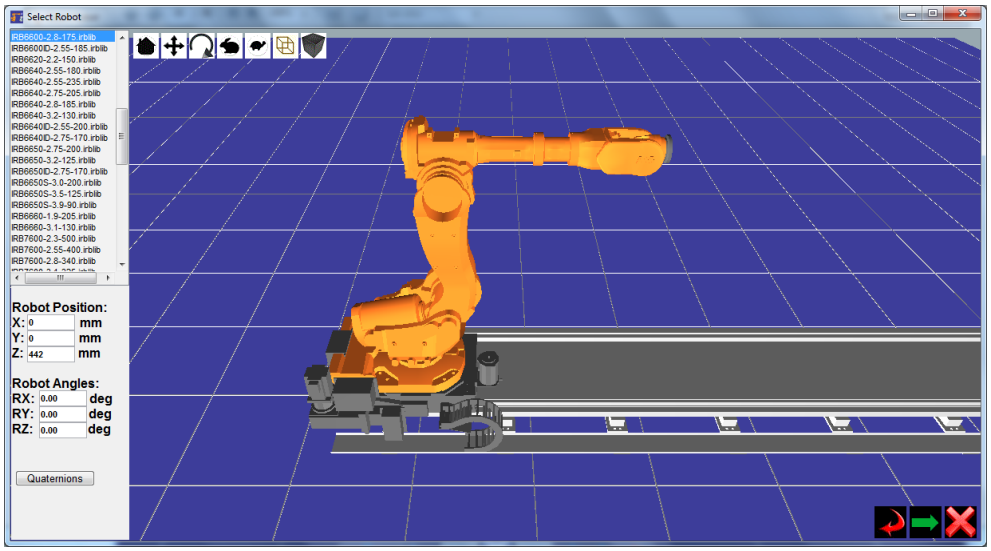


Figure 46: Robot library.

the robot mounting height of the track IRB6003S-10.7. Figure 47 shows the tool library. Choose the Spindle-UBR with default settings and click →.

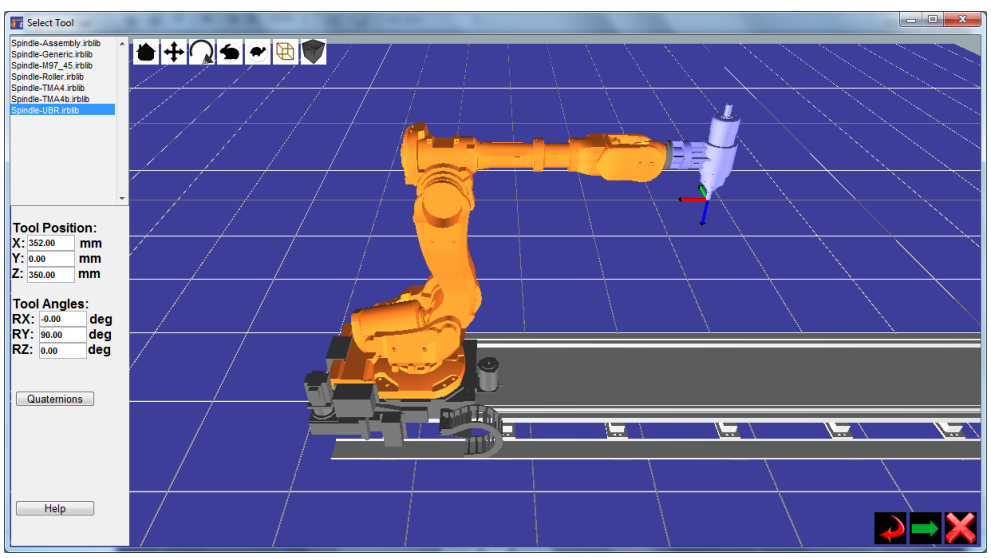


Figure 47: Tool library.

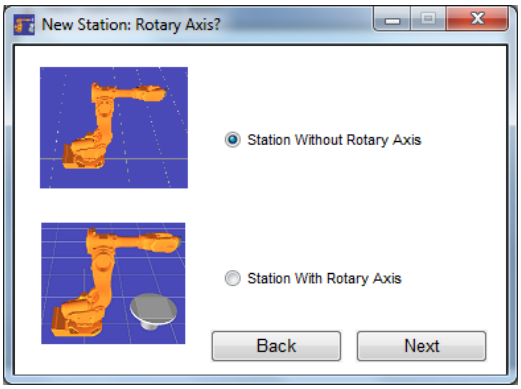


Figure 48: Option for rotary table.

In Figure 48, choose 'Station Without Rotary Axis' and click 'Next'.

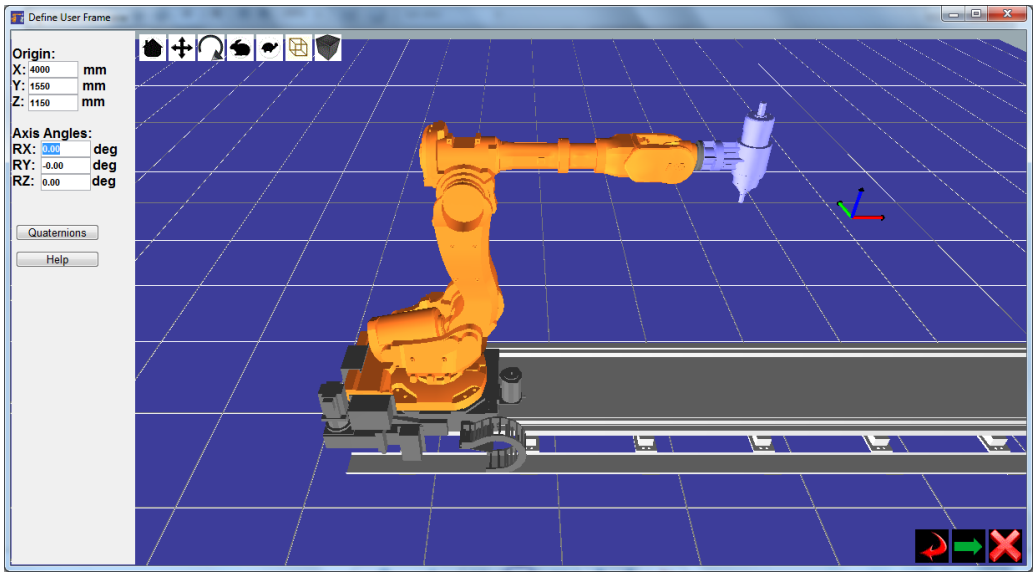


Figure 49: Definition of user frame.

In Figure 49 define the user frame equal to $X = 4000mm$, $Y = 1550mm$ and $Z = 1150mm$ and no rotations. Click on $\rightarrow$.

In Figure 50 choose 'Station With User Object' and click on 'Next'.

In Figure 51 choose the user object named 'BoatHull'. The object is automatically placed at the origin of the user frame defined in the previous step. The position and orientation in this window correspond to the object frame (OFRAME) on the robot. Keep the default settings and click on $\rightarrow$.

Next, we want to define the machining table. In Figure 52 select 'Add Cube' and click 'Next'.

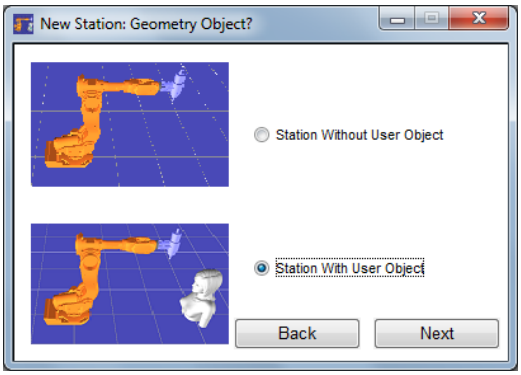


Figure 50: Option for user objects.

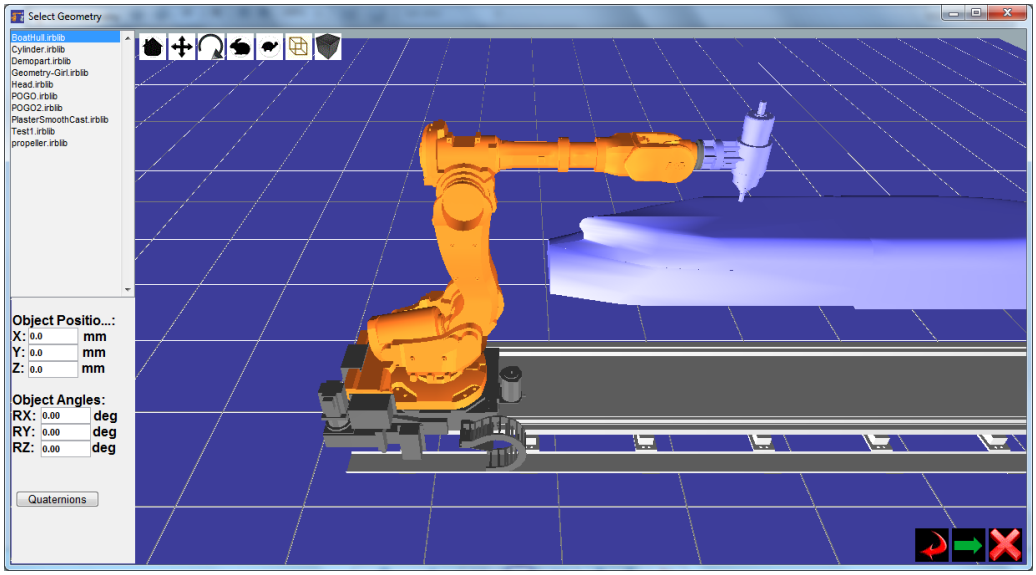


Figure 51: User object library.

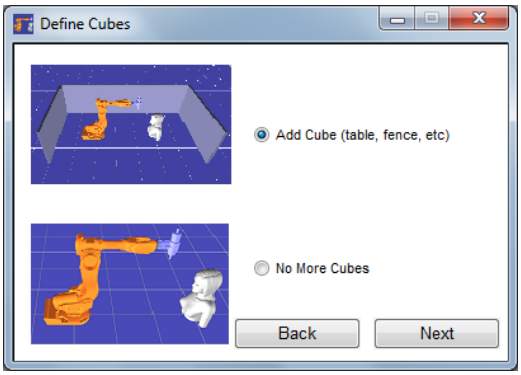


Figure 52: Option for defining cubical objects.

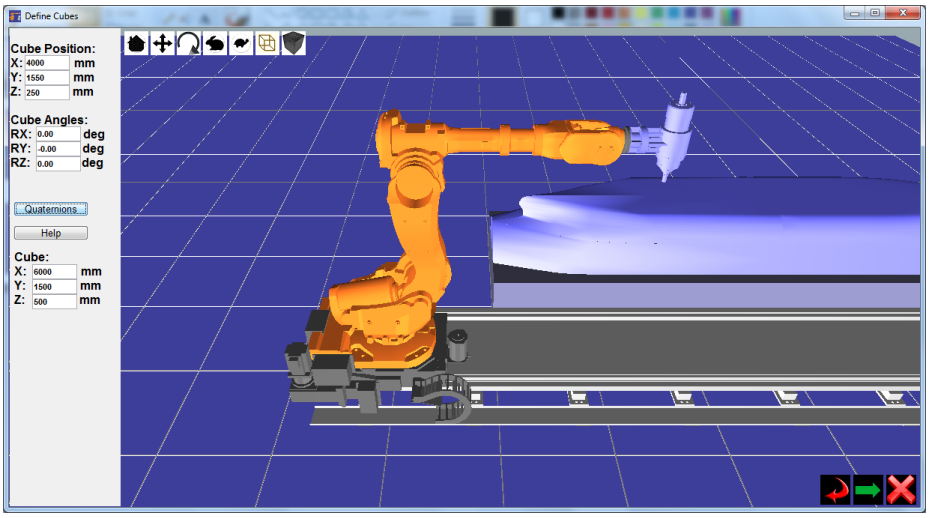


Figure 53: Definition of machining table.

In Figure 53 we define the position and size of the cube. Set the position to $X = 4000$, $Y = 1550$ and $Z = 250$ and the width, length and height to $6000, 1500, 500$, respectively. Click $\rightarrow$, no more cubes and save the station.

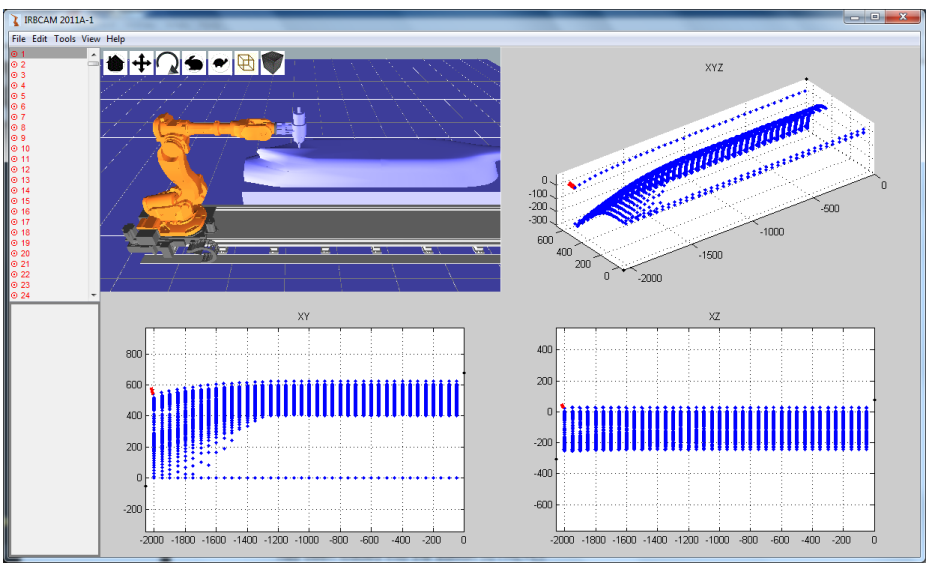


Figure 54: APT file loaded in Combination View.

Figure 54 shows the combination view (CTRL+2) after the APT file 'BoatHull.apt' has been loaded into the station (CTRL+L).

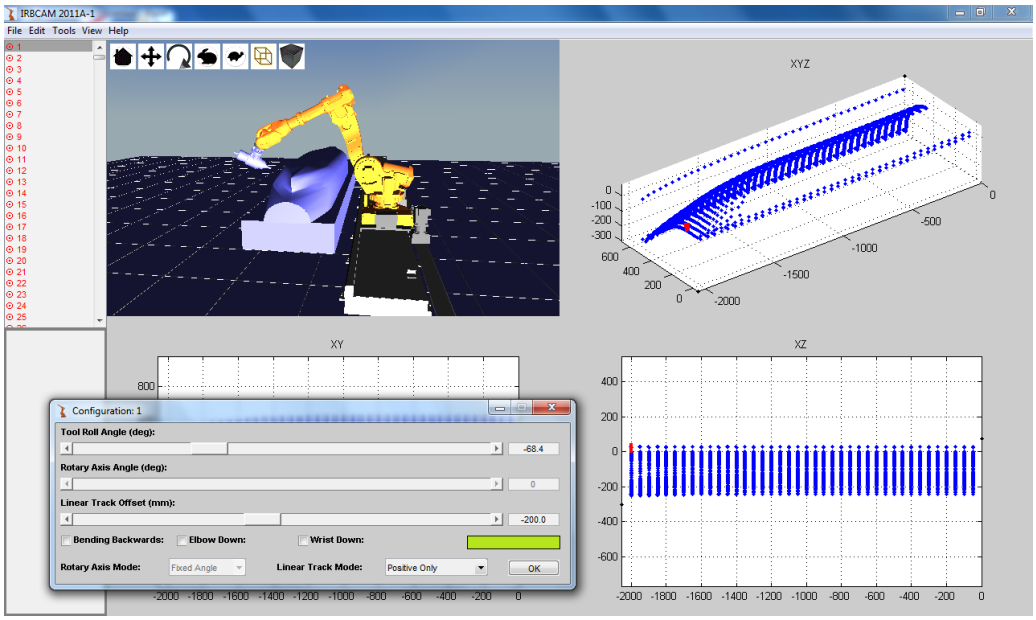


Figure 55: Path Configuration.

Figure 55 shows the toolpath configuration (Tool - Configure Path or CTRL+K). The 3D graphics window has been rotated by holding the CTRL key and using the arrow keys on the keyboard. In the previous example in section 7.3 there was only one parameter which had to be defined (the Tool Roll Angle). When there is a linear track in the station, there is one more parameter: The Linear Track Offset. Move the slider for the track offset and you will see which effect this parameter has. Try with the following settings: Tool Roll Angle = -68.4 degrees and Linear Track Offset equal to $-200mm$. Set the Linear Track Mode to 'Dynamic Position' and click on OK.

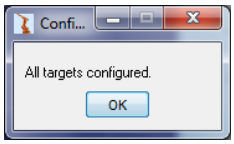


Figure 56: Configuration Succeeded.

Figure 56 shows that all the targets were configured. In cases when we can not find parameter settings which work, we can use the 'Optimizer'.

Figure 57 shows the Optimizer (Tools - Optimizer or CTRL+I). In this window choose Linear Track Mode = Dynamic Position. The default parameter to optimize is the Tool Roll Angle. Click on 'Optimize' and wait for the results which are shown in Figure 57. A bar with value equal to 1 in this figure means that the corresponding Tool Roll Angle will configure the entire path. In this example, all roll angles will work, except for angles between -90° to -120°. Note that the value for

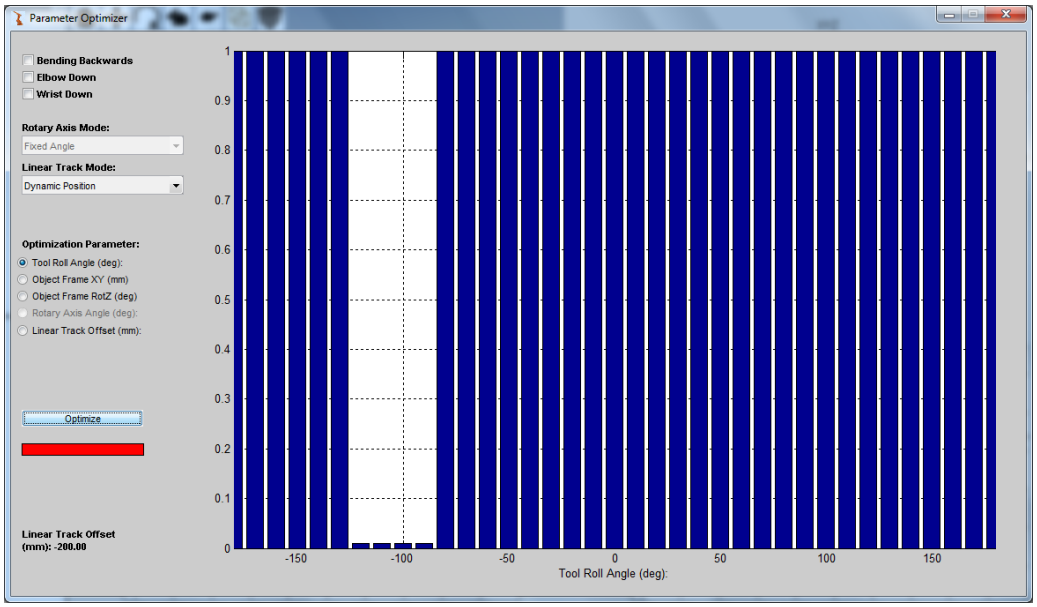


Figure 57: Path Optimizer.

the linear track offset of $-200mm$ displayed in the bottom left corner of Figure 57 in the optimizer is taken from the configurator screen in Figure 55 when optimizing the Tool Roll Angle.

By selecting 'View - Linear Track' (or CTRL+8) the linear track motion can be inspected. We see from Figure 58 that the track moves from $1793.6mm$ to $3749.97mm$ with a total travelled distance of $1956.34mm$. This completes the example with the linear track motion.

In just a few minutes, we have managed to configure a robot station with a linear track motion and a toolpath of several thousand coordinates. The configured path can now be exported as a RAPID file (CTRL+R) or as a MOD+ROB combination (CTRL+M). The MOD+ROB option allows for uninterrupted milling of very large files (only limited by the size of the flash-disk), while RAPID files need to be split if there are more than about 24,000 targets.

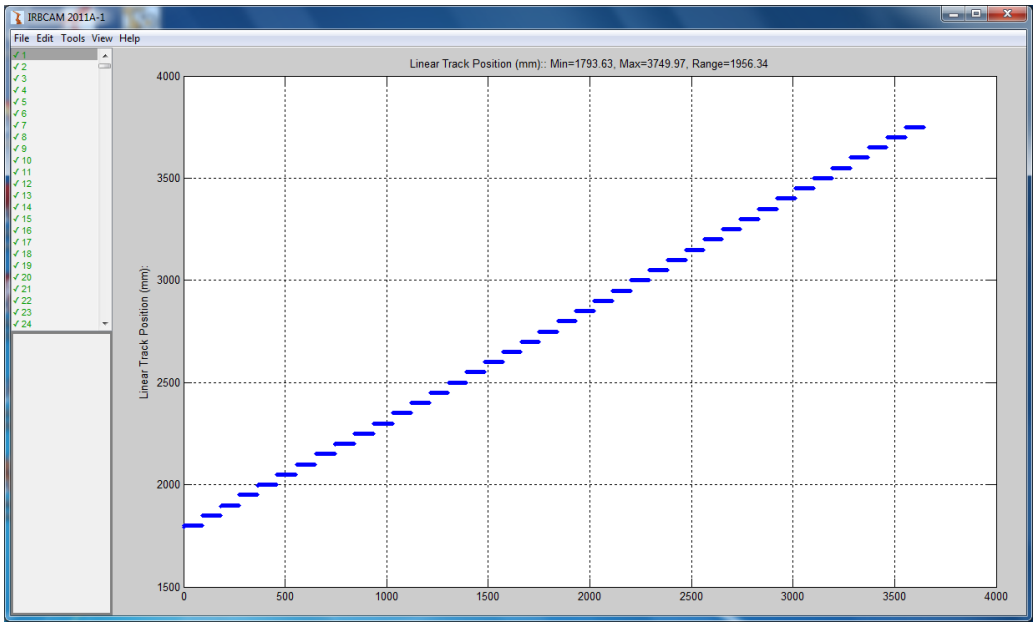


Figure 58: Linear Track Motion.

7.5 ABB IRB6400 Robot with Rotary Table - Dynamic Angle Mode 1

In this section an example with an IRB6400-2.4-M97 robot and a vertical axis rotary table is presented. Figure 59 shows the tool library when an IRB6400-2.4-M97

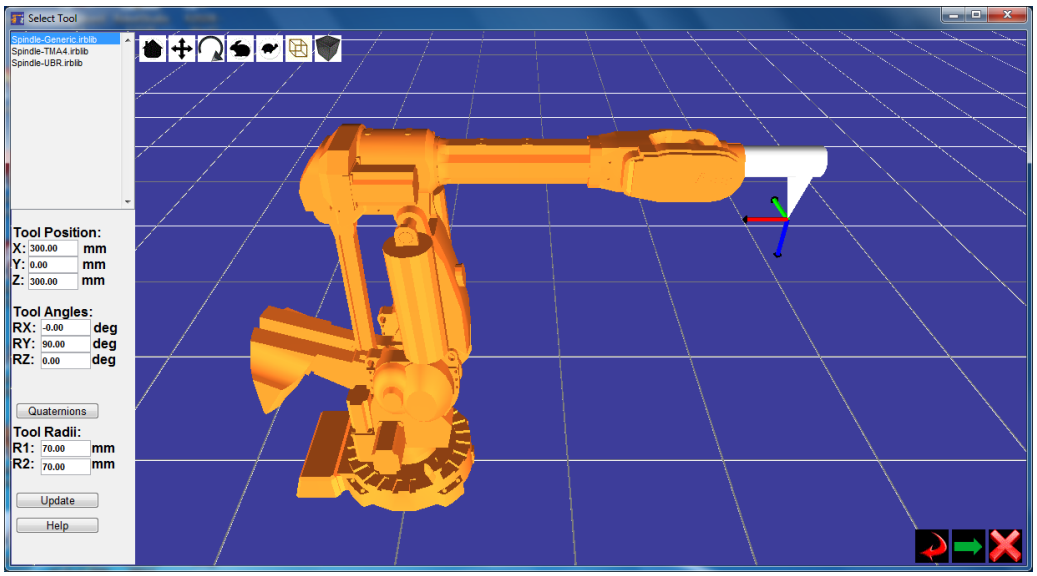


Figure 59: IRB6400-2.4-M97 robot with generic tool.

robot has been selected. Select the generic tool and set the X-distance equal to $500mm$. Click 'Update' to update the graphics of the generic tool, then ✔.

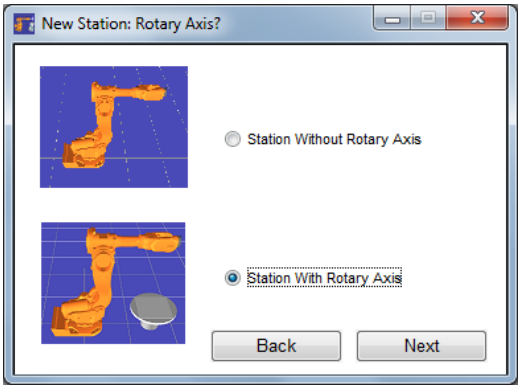


Figure 60: Choice for rotary axis.

In Figure 60 choose 'Station With Rotary Axis' and click OK.

Figure 61 shows the library for the rotary axes. The library contains a limited number of turntables. Howeverm the provided library should be sufficient for most

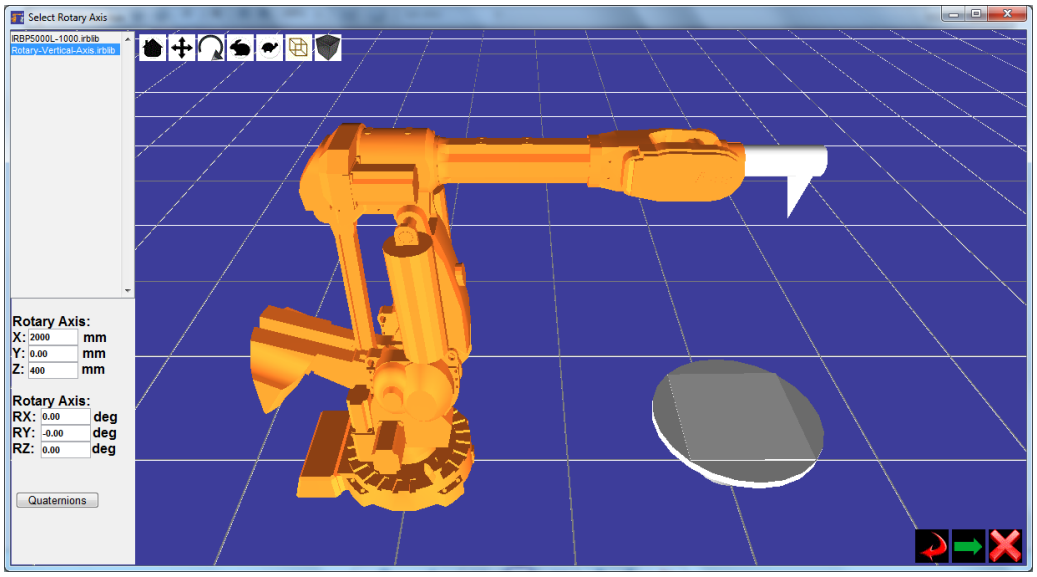


Figure 61: Library for rotary axis.

applications, since the models can be positioned freely and the rotational axis oriented in any direction. In this example, choose the model 'Rotary-Vertical-Axis' and position it at $X = 2000mm$, $Y = 0$ and $Z = 400mm$. Click ✔.

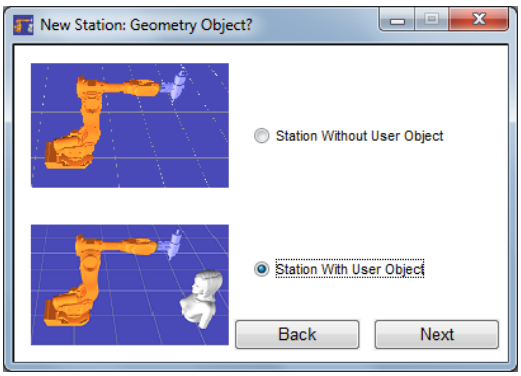


Figure 62: Choice for geometry object.

In Figure 62 choose 'Station With User Object' and click OK.

Figure 63 shows the library for user objects. Select the model 'Geometry-Girl'. The model is automatically positioned at the origin of the user frame, which is located on the rotary axis. Accept the default settings and click ✔.

In this example we do not need any cubical objects. In Figure 64 select 'No More Cubes' and click OK.

Figure 65 appears when the APT file 'girl.apt' has been loaded (CTRL+L), the Combination View (CTRL+2) has been selected and the configurator has been

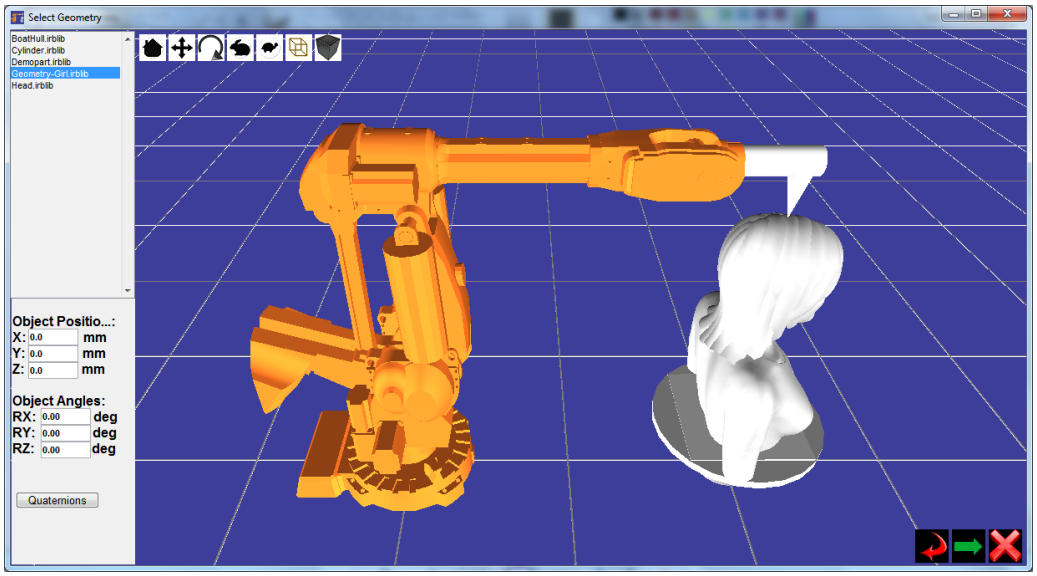


Figure 63: Library for geometry objects.

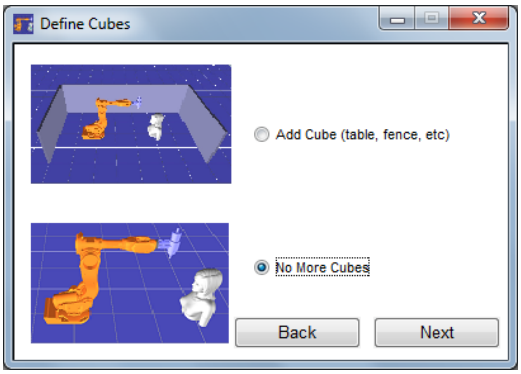


Figure 64: Choice for cubical objects.

opened (CTRL+K). Set the Tool Roll Angle equal to 90° and the Rotary Axis Angle also to 90°. Change the Rotary Axis Mode from 'Fixed Angle' to 'Dynamic Angle 1' and click OK. Dynamic Angle Mode 1 is used for sculpturing, when tbe milling tool is mainly oriented parallel to the table plane.

With these settings the entire path of more than 7000 coordinates is configured successfully. By selecting 'View - Rotary Axis' (or pressing CTRL+7), the rotary axis angular movement is displayed as in Figure 66. It can be seen from this figure that the initial rotary angle is 90° as selected in the configurator and that the total movement of the rotary axis in this example is 360^0, from 90° to -270°. The configured toolpath can now be saved as either a single RAPID file or a MOD+ROB combination. This concludes the example of a robot and a rotary axis.

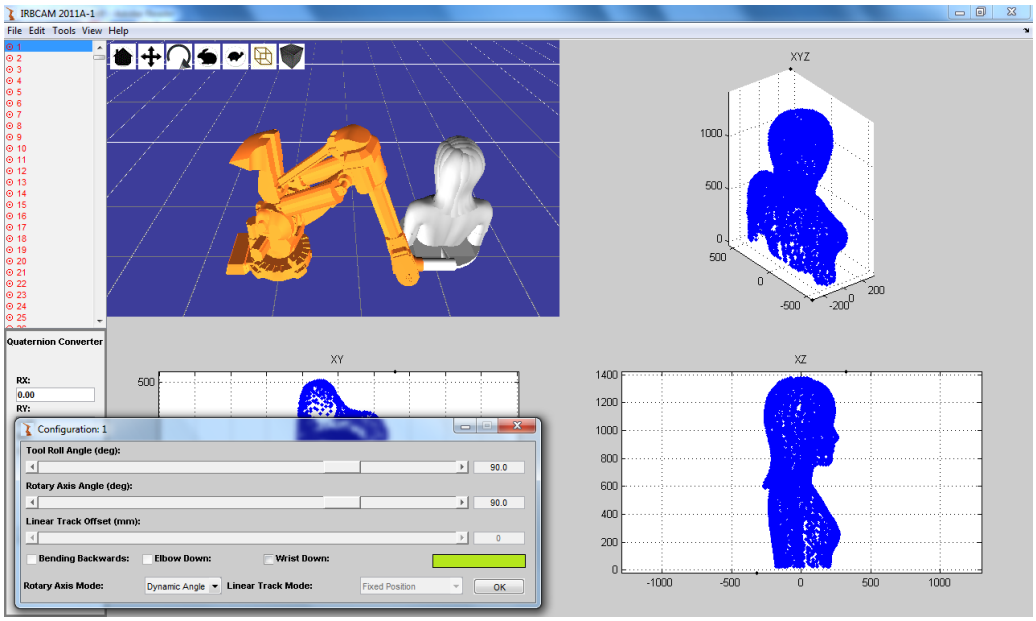


Figure 65: Path configuration including rotary axis.

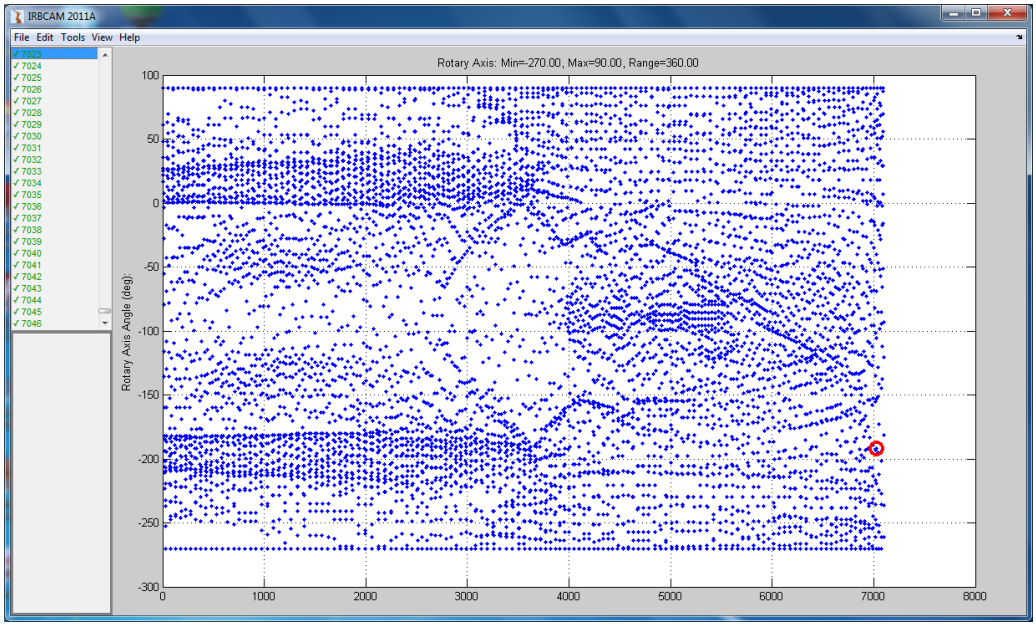


Figure 66: Rotary axis movements.

7.6 ABB IRB6400 Robot with Rotary Table - Dynamic Angle Mode 2

In this section the example is similar to the previous section, except that Dynamic Angle Mode 2 will be demonstrated. As in the previous example, an IRB6400-2.4-M97 robot with a vertical axis rotary table are used. Use the New Station Wizard and follow the steps in the previous example as illustrated in Figs. 59-61. In Figure 67 (left) choose 'Station Without User Object' and click OK. In Figure 67

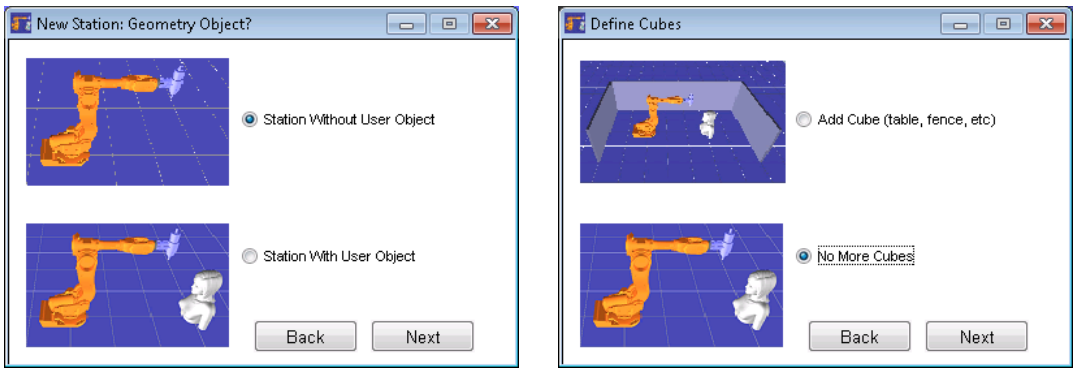


Figure 67: Choice for geometry object (left) and cubes (right).

(right) choose 'No More Cubes', click 'Next' and save the station.

Point	X	Y	Z	Arc
1	200	0	50	
2	200	0	0	
3	0	200	0	✔
4	-200	0	0	
5	0	-200	0	✔
6	200	0	0	
7	200	50	0	

Table 5: Coordinate points and attributes for offline programming example.

Similar to the example in section 7.1, enter the coordinates in Table 5. Use the menu 'Edit - Targets - Add Target' or press CTRL+E to add a new target. Target number 3 and 5 in Table 5 should be defined as 'Arc Mid-Point'. To define an arc mid-point, right-click on target 3 or 5 in the target list (top left), click on '»' and then select 'Motion Type: Arc Mid-Point' and click on '«' to save. Note that a target can not be defined as an arc-mid point if it is the last point in the list.

Figure 68 (left) shows the combination view (CTRL+2) after all the targets in Table 5 have been defined. A full circle with radius 200 is defined with the initial and final

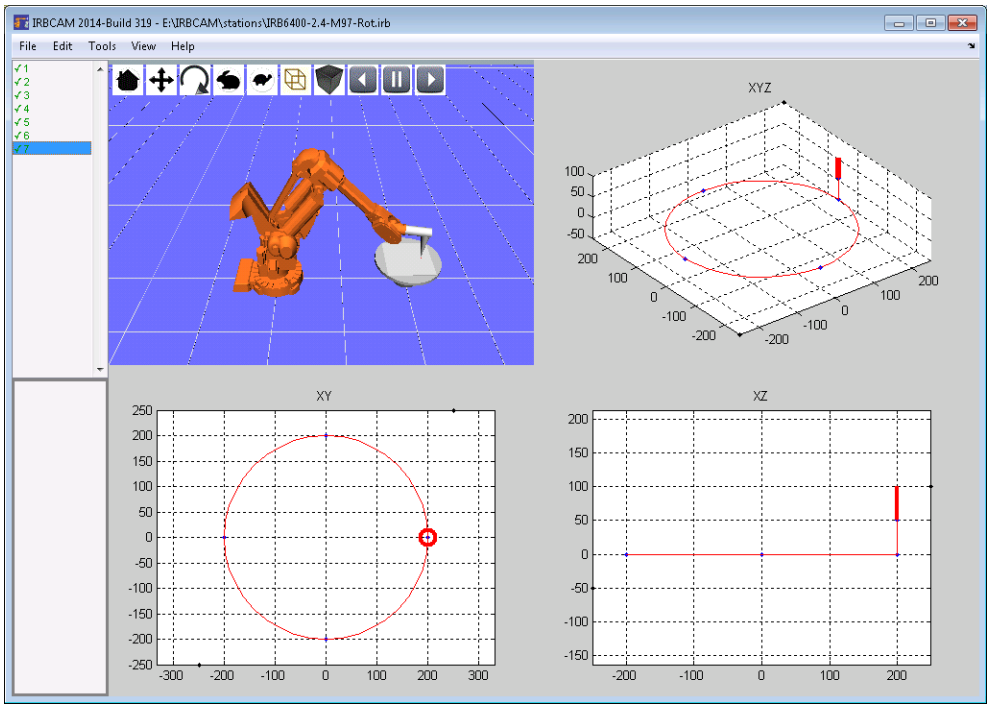


Figure 68: Combination view (CTRL+2) after the targets in Table 5 have been defined.

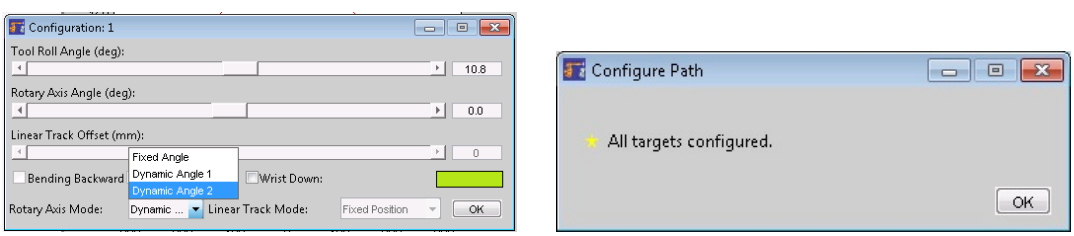


Figure 69: Configuration options (left) and status window after configuration (right).

lift point at $X = 50$ and $Z = 50$. Figure 69 shows the configuration options which appear after selecting 'Tools - Configure Path' or pressing CTRL+K. Define 'Rotary Axis Mode' equal to 'Dynamic Angle 2' and 'Tool Roll Angle' equal to 10.8 degrees and press OK. All targets should be defined successfully as shown in Figure 69 (right). In this example the rotary axis performs the entire motion of the circle, while the robot and the tool are stationary. The robot only moves at the initial and final lift points.

As illustrated by the examples in sections 7.5 and 7.6 'Dynamic Angle 1' is used primarily for sculpturing-like toolpaths where the tool is oriented in the plane defined by the turntable. 'Dynamic Angle 2' is used when the tool is oriented normal to (or close to normal to) the plane defined by the turntable.

7.7 ABB IRB4400 Robot with Linear Track and Rotary Axis

In this final example a setup including an IRB4400 robot with both a linear track and a rotary axis will be demonstrated. Select the linear track 'IRB6003S-10.7', the robot 'IRB4400-1.96-45' and the tool 'Spindle-TMA4', all with the default settings. Select 'Station With Rotary Axis'.

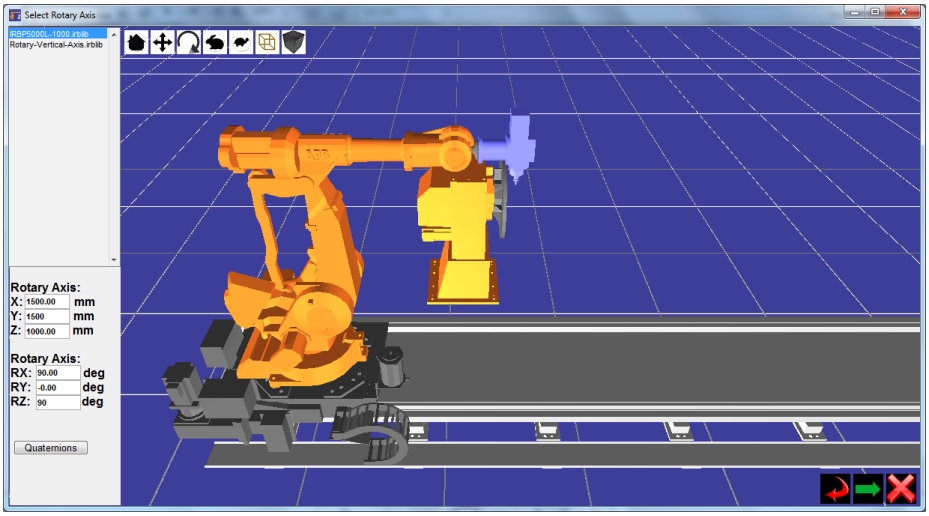


Figure 70: Library of rotary axes.

In Figure 70 the rotary axis library is shown. Select the rotary axis 'IRBP5000L-1000' and position this axis at $X = 1500mm$, $Y = 1500mm$ and $Z = 1000mm$. Rotate the rotary axis 90° about Z, such that the rotary axis is aligned in the same direction as the linear track.

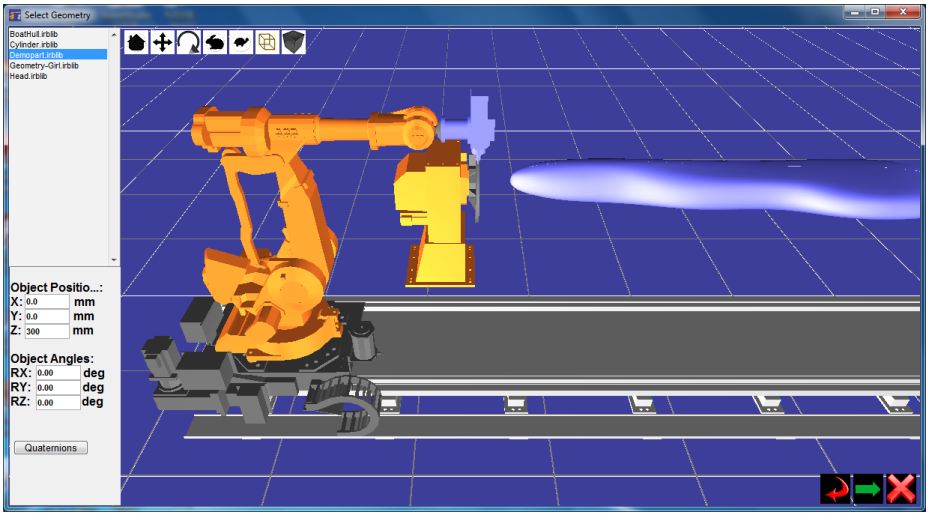


Figure 71: Selection of user geometry.

Figure 71 shows the library of user geometries. Select the object 'Demopart' and position this object $Z = 300mm$ out from the origin of the rotary axis. This distance $Z = 300mm$ corresponds to the robot's object frame (OFRAME).

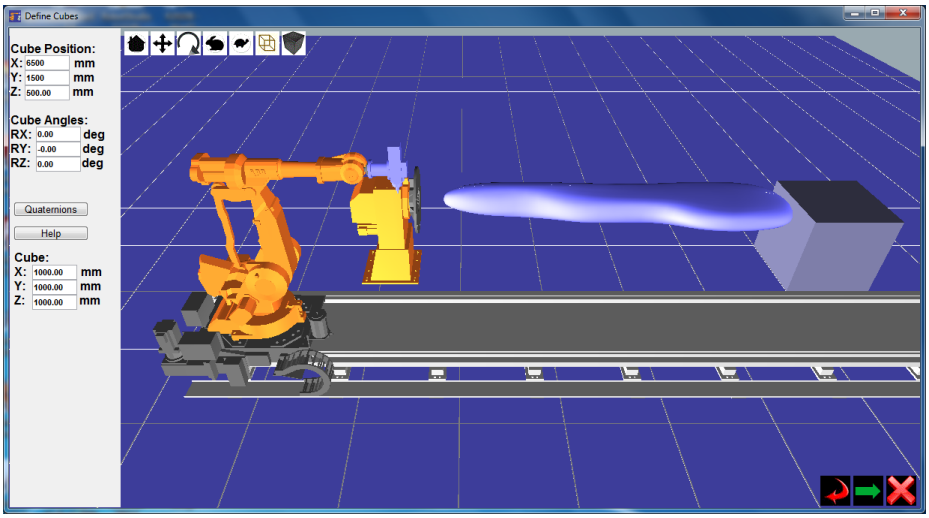


Figure 72: Definition of cubical object.

Define a cubical object at the other end of the Demopart object, as shown in Figure 72. Position the cube at $X = 6500mm$, $Y = 1500mm$ and $Z = 500mm$. This cube represents a simplified support mechanism for the Demopart.

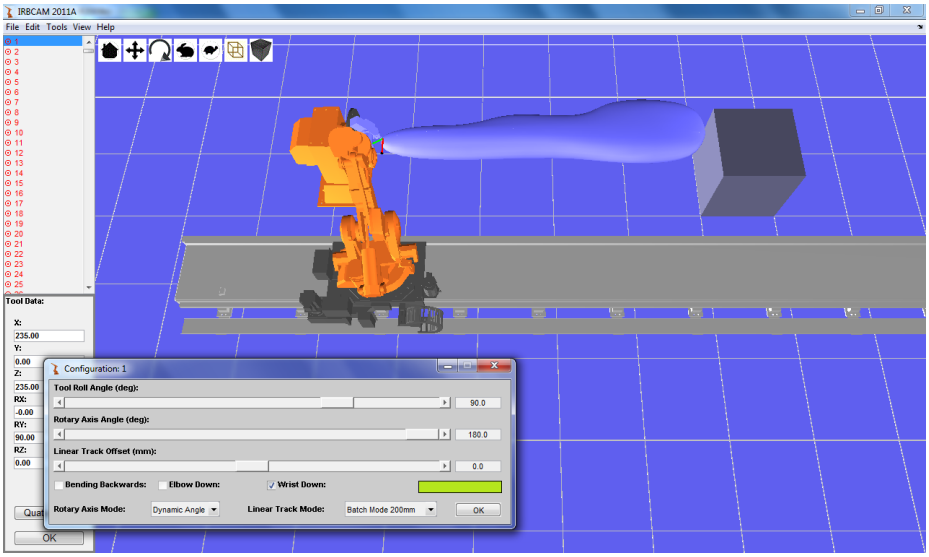


Figure 73: Path configuration.

Figure 73 shows the configuration screen after the APT file 'Demopart.apt' has been loaded (CTRL+L), the Combination View has been selected (CTRL+2) and the configurator has been opened (CTRL+K). In this example, all three sliders are activated for the Tool Roll Angle, the Rotary Axis Angle and the Linear Track Offset. Set the Tool Roll Angle to 90°, the Rotary Axis Angle to 180° and the Linear Track Offset to zero. Set Rotary Axis Mode to 'Dynamic Angle' and the Linear Track Mode to 'Batch Mode 200mm'. This last option means that the linear track will try to stay constant, but when it has to move it moves by at least $200mm$. Mark the check-box 'Wrist Down', which changes the configuration of the wrist. This option can be useful to avoid twisting of cables mounted to the robot's upper arm. Finally, click on OK.

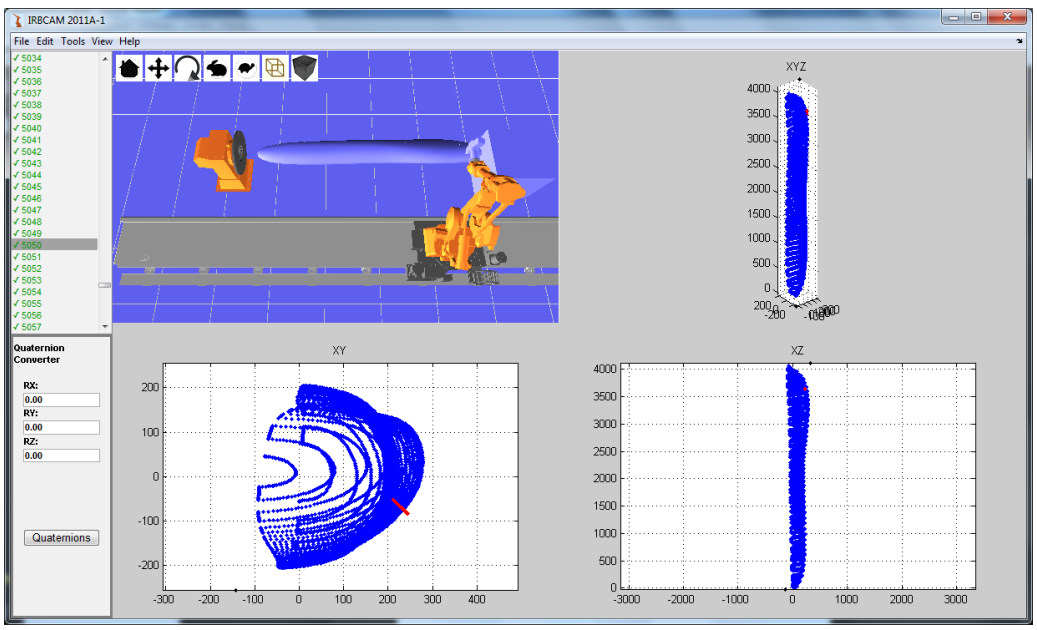


Figure 74: Configured toolpath.

Figure 74 shows the fully configured path of more than 5000 robot targets. When selecting the 'Dynamic Angle' mode for the rotary axis, IRBCAM tries to keep the orientation of the spindle equal to the first target which was configured manually with the configurator window in Figure 73.

Figure 75 shows the angular movements of the rotary axis (CTRL+7). In this case, the total movement of the rotary axis is about 180°.

Figure 76 shows the linear track movements. Note that the track tries to stay constant, but when it moves it moves in steps of $200mm$ corresponding to the 'Batch Mode 200mm' as selected in Figure 73. This concludes the example of using a robot mounted on a linear track in combination with a rotary axis. The configured toolpath can be exported as a single RAPID file or as a MOD+ROB combination.

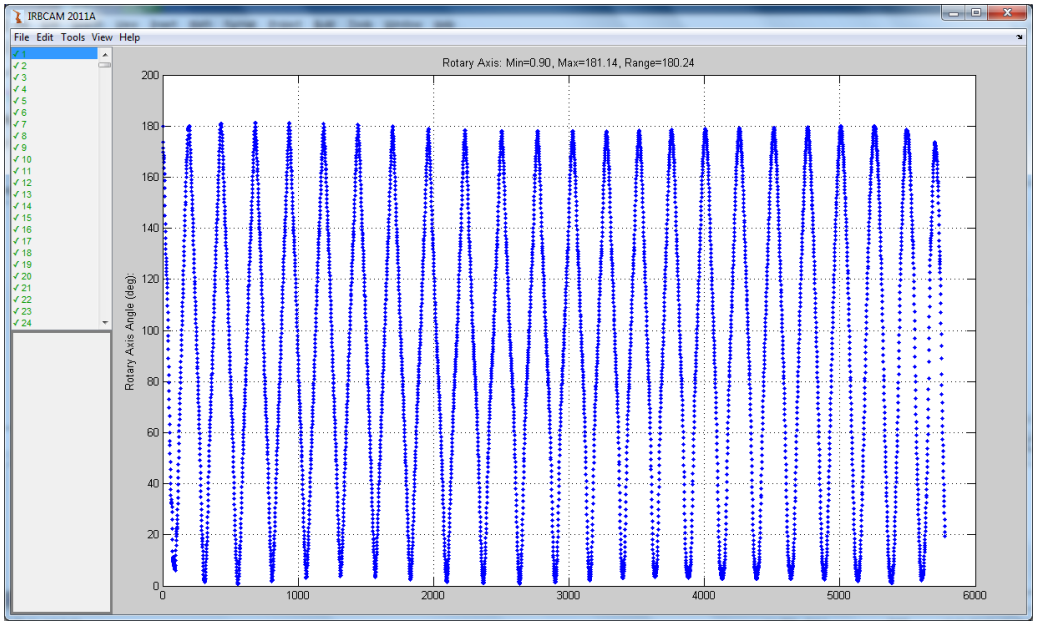


Figure 75: Rotary axis angular movements.

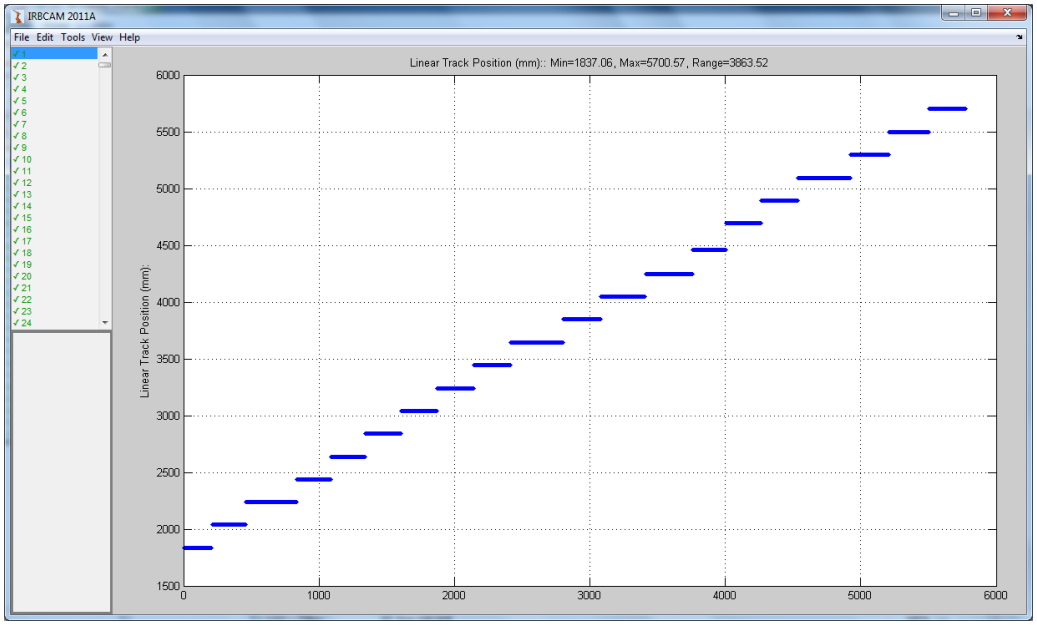


Figure 76: Linear track movements.

7.8 ABB IRB6600 Robot with 3-Axis Construction Views

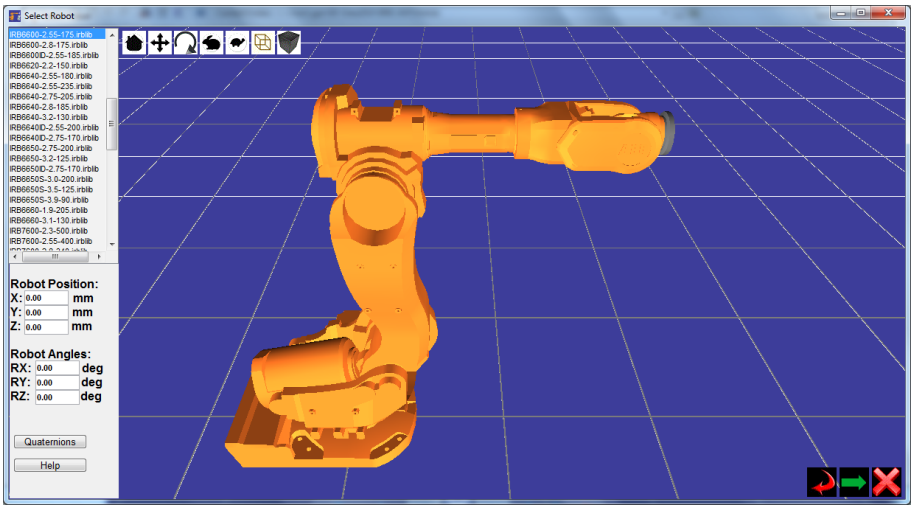


Figure 77: New Station: IRB6600 Robot.

In this example, it will be demonstrated how to configure a 3-axis toolpath with several construction views. First, define a new station without a linear track and select the the IRB6600 robot as shown in Figure 77. Next, select the TMA4 spindle as shown in Figure 78. Accept the default settings of the spindle.

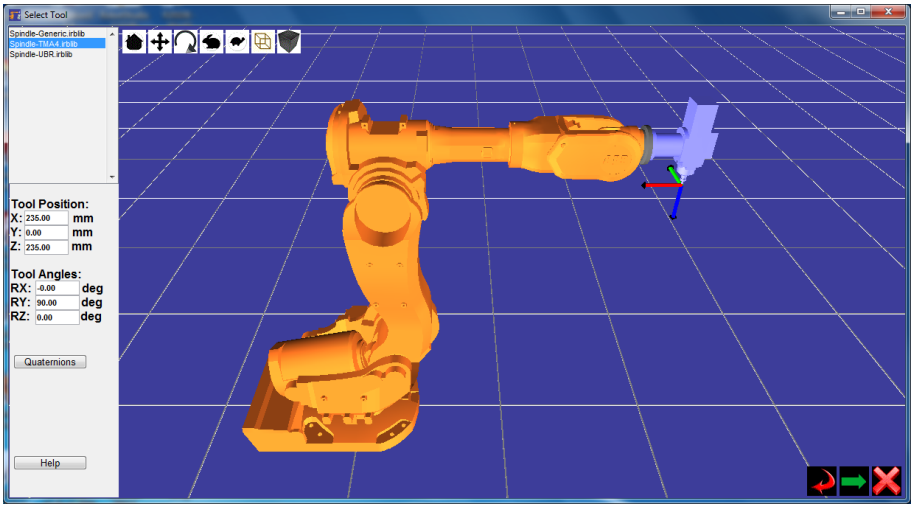


Figure 78: New Station: TMA4 Spindle.

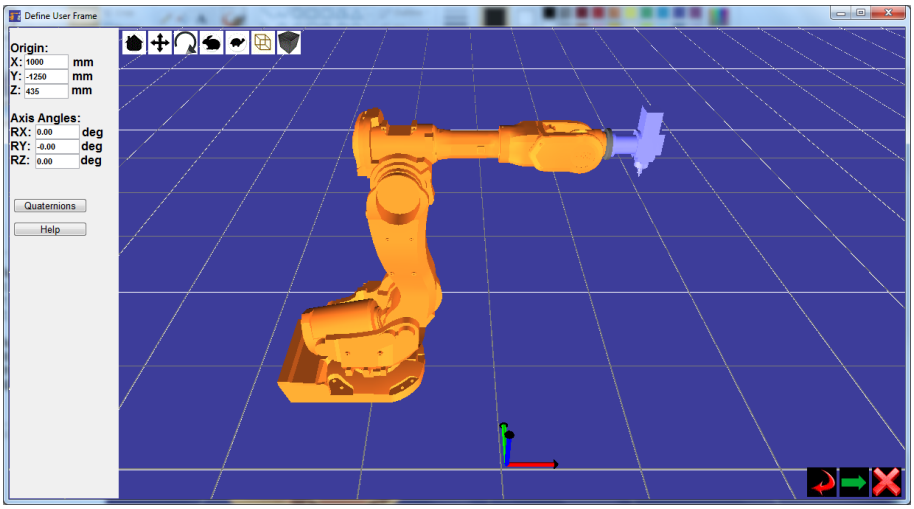


Figure 79: New Station: Definition of user frame.

Define the user object origin at X=1000mm, Y=-1200mm and Z=435mm as shown in Figure 79. Select a station with user object and place the user object 'Head' at X=500mm, Y=1200mm as shown in Figure 80. The 'Head' position corresponds to the object frame (OFRAME) of the robot, and the position is relative to the user frame (UFRAME).

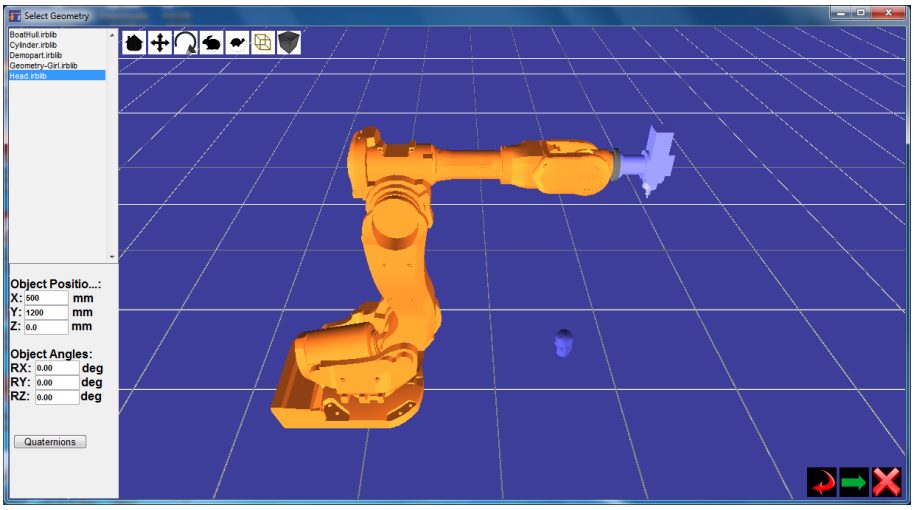


Figure 80: Linear track movements.

In this example, we will demonstrate how to add a cube after the station has been defined. In the new station wizard, do not add any cubes. After the wizard is completed and the station has been saved, select 'Edit - Cube - Add' in the menus as illustrated in Figure 81.

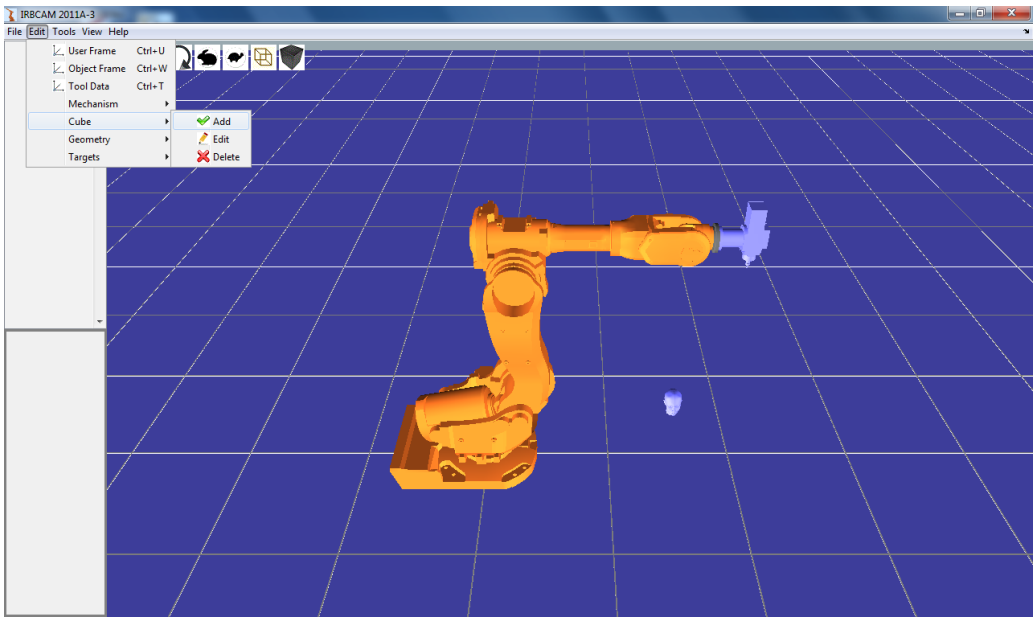


Figure 81: Addition of Cube.

Figure 82 shows the next screen which appears. Here you select the color of the new cube. The three sliders correspond to the components of Red, Green, Blue (RGB) colors.

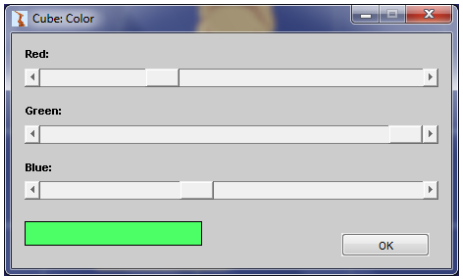


Figure 82: Color selection for new cube.

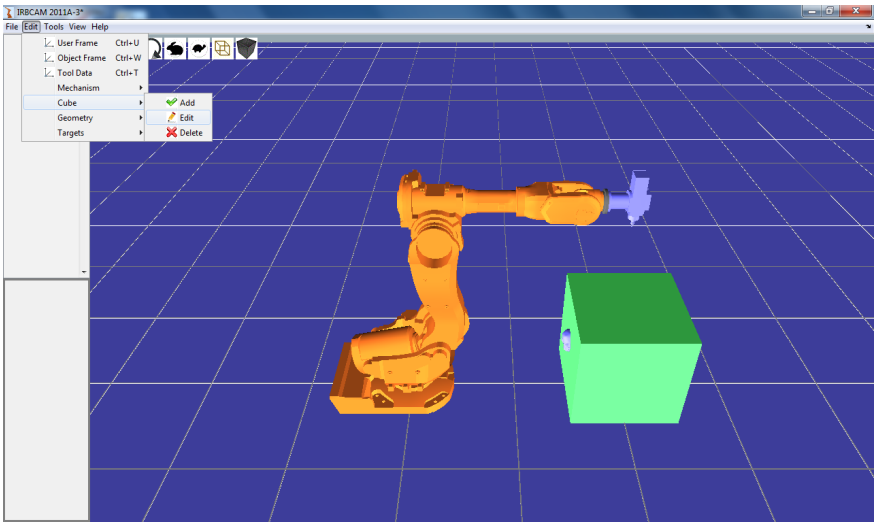


Figure 83: Edit Cube.

Next, select 'Edit - Cube - Edit' in the menus, as shown in Figure 83. Set the cube position to X=1500mm and Z=217.5mm. Set the cube dimensions to X=1000mm, Y=2400mm and Z=450mm. This cube represents a machining table.

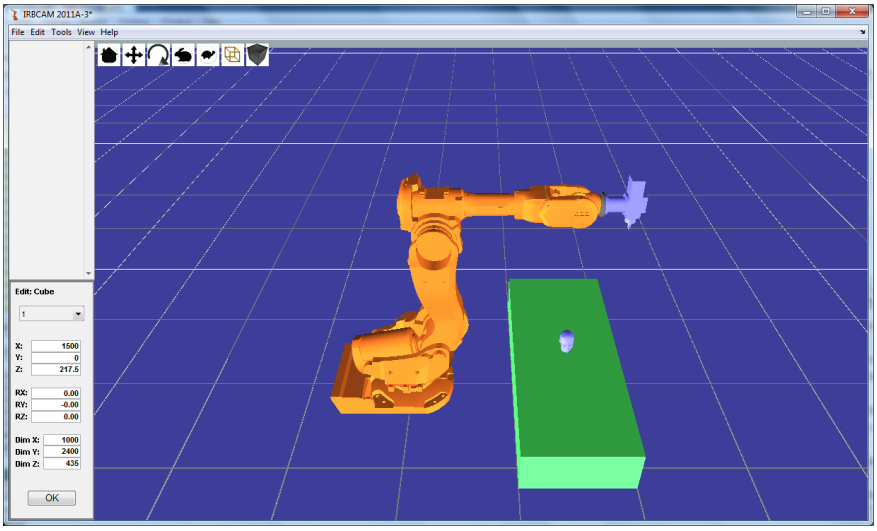


Figure 84: Cube Parameters.

Load the APT file 'Head.apt' into the station. This file was generated by SurfCam and contains about 8100 coordinates and 5 different construction views (top of the head and four sides). In order to machine this toolpath with the robot, we need to add transition points (4 in total) between the construction views. The first transition occurs after coordinate 2292. Select 'View - Combination View' in the menus (or CTRL+2), click on coordinate 2292 in the list of coordinates, followed by 'Edit - Targets - Add Target' as shown in Figure 85. In the dropdown box which appears,

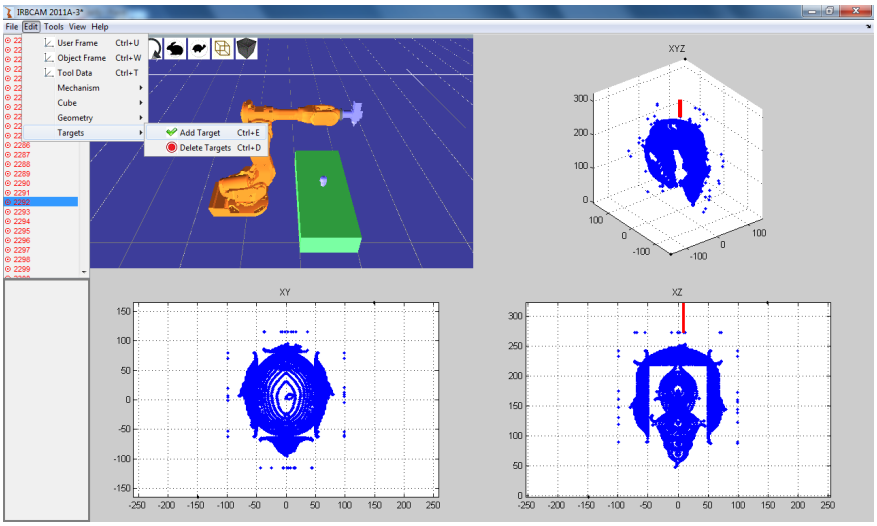


Figure 85: Transition 1 after coordinate 2292.

select 'After 2292', see Figure 86.

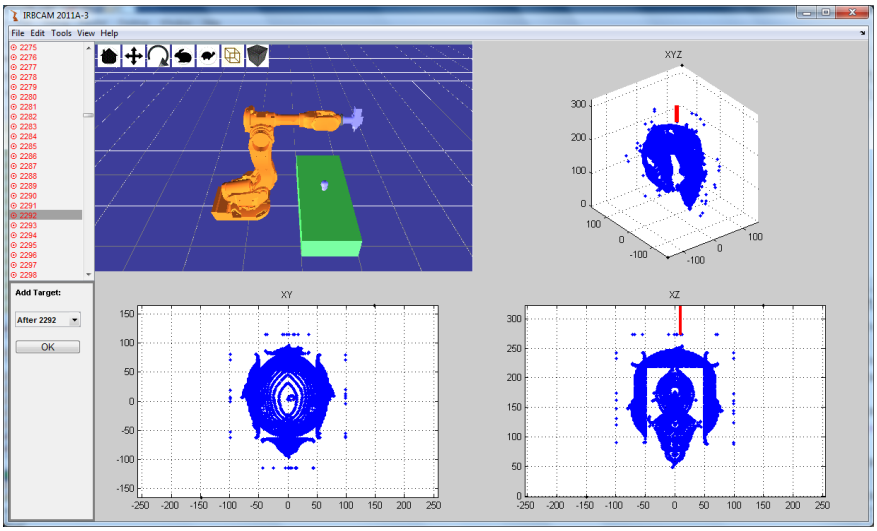


Figure 86: Addition of new coordinate after 2292.

IRBCAM automatically suggests a new position and tool orientation for the new coordinate as shown in Figure 87. The new position and tool vector are placed in the middle between the neighbouring coordinates. For this first transition, we are satisfied with the tool vector, but we edit the position to X=0mm, Y=-150mm and Z=270mm to avoid any collisions between the spindle and the work object, see Figure 87.

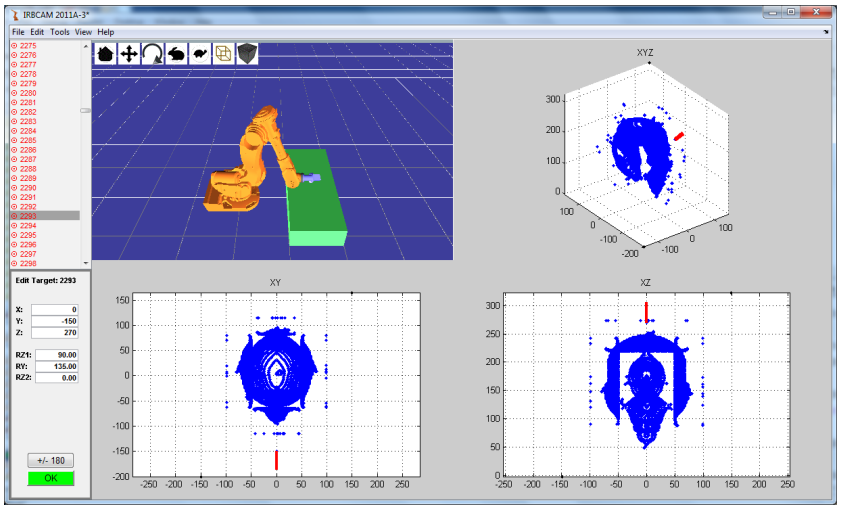


Figure 87: Edit new coordinate 2293.

Figure 87 shows the second transition point which is added after coordinate 4032. Again, we are satisifed with the suggested tool vector, but change the position to X=100mm and Y=-100mm.

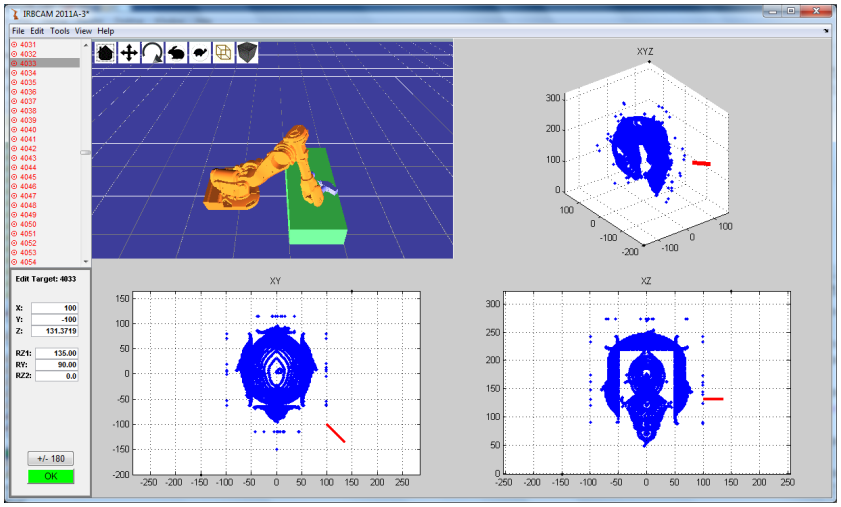


Figure 88: Edit new coordinate 4033.

Figure 89 shows the third transition point added after coordinate 5855. Again, we are satisifed with the suggested tool vector and change the position vector to X=100mm and Y=100mm. There is no need to edit the tool roll angle (RZ2), since this angle will be automatically adjusted by IRBCAM when the toolpath is configured. Finally, the fourth transition point is added after coordinate 6349. The

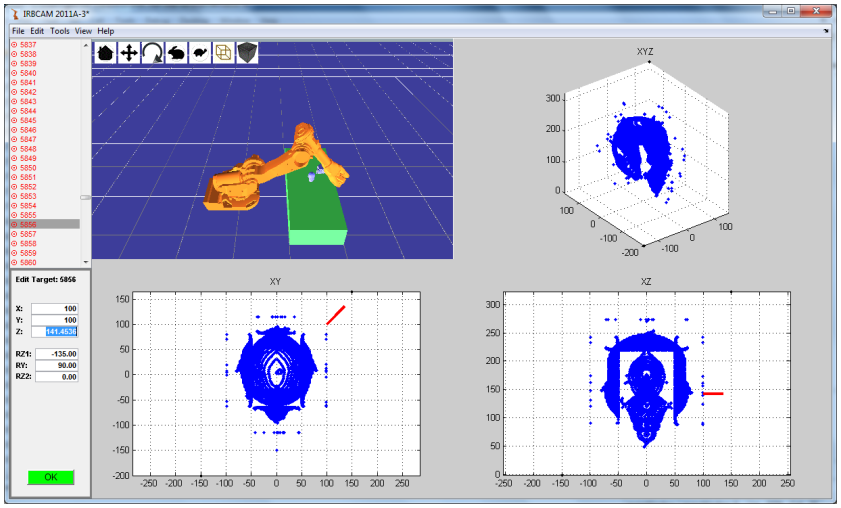


Figure 89: Edit new coordinate 5856.

suggested toolvector is OK, but we edit the position to X=-100mm and Y=100mm.

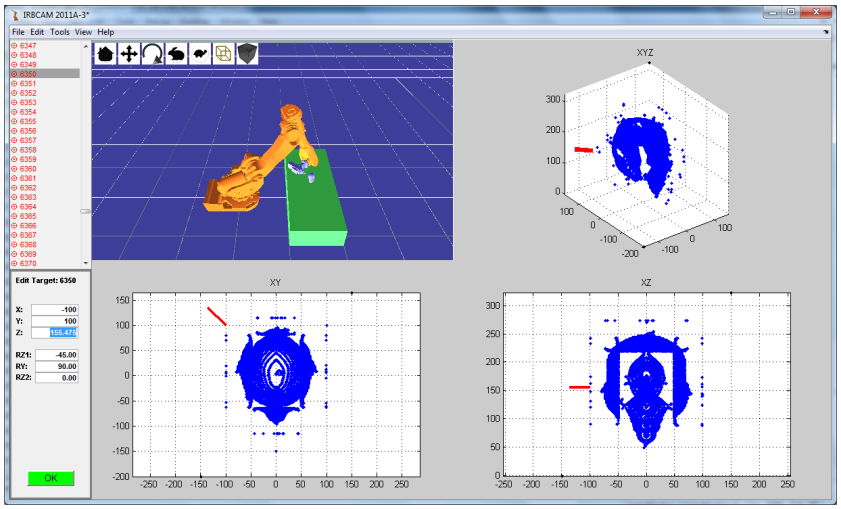


Figure 90: Edit new coordinate 6350.

Finally, the entire toolpath with 5 different construction views can be configured (CTRL+K) with the settings shown in Figure 91. After the toolpath has been configured, have a look at the four transition points and notice that IRBCAM has adjusted the tool roll angles such that all the four transitions are smooth with as little tool re-orientation as possible.

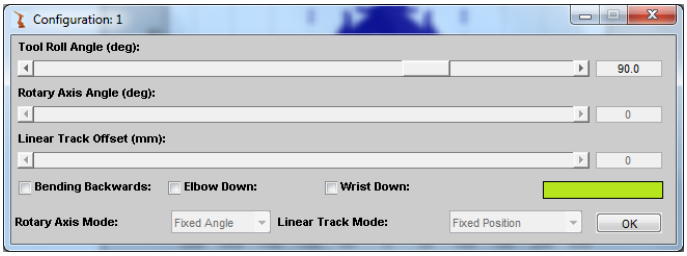


Figure 91: Toolpath configuration.

7.9 KUKA KR60-3 Robot and Roller Mode

Define a new station with a KUKA KR60-3 robot, the tool 'Spindle-Roller' and the object 'Cylinder', as shown in Fig. 92. Define the user frame (CTRL U) at the position (1500,0,300) and rotation RZ=90°.

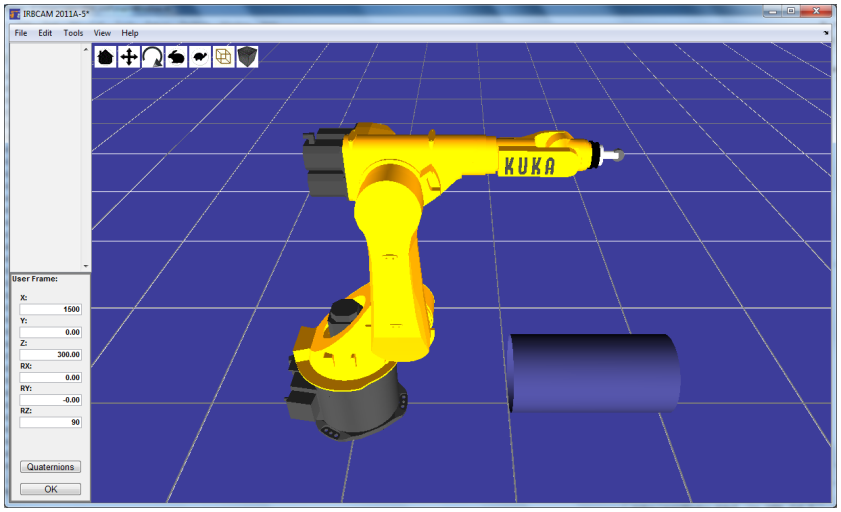


Figure 92: Definition of station with KUKA KR60-3 robot, the tool Spindle-Roller and Cylinder object.

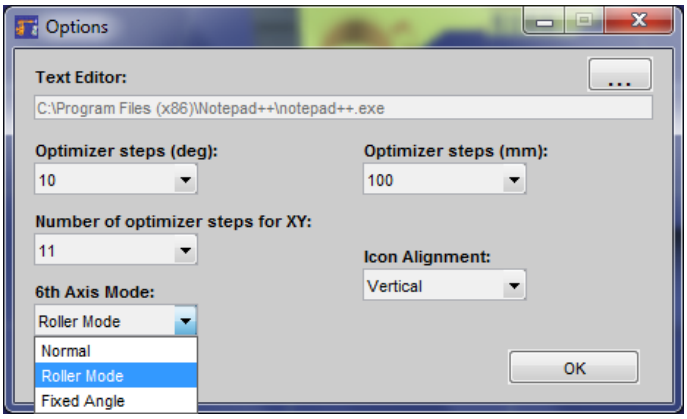


Figure 93: IRBCAM Options

In 'Tools - Options' choose 'Roller Mode'. Note that this mode is only possible for straight tools, ie. tools which have zero lengths in the X and Y directions.

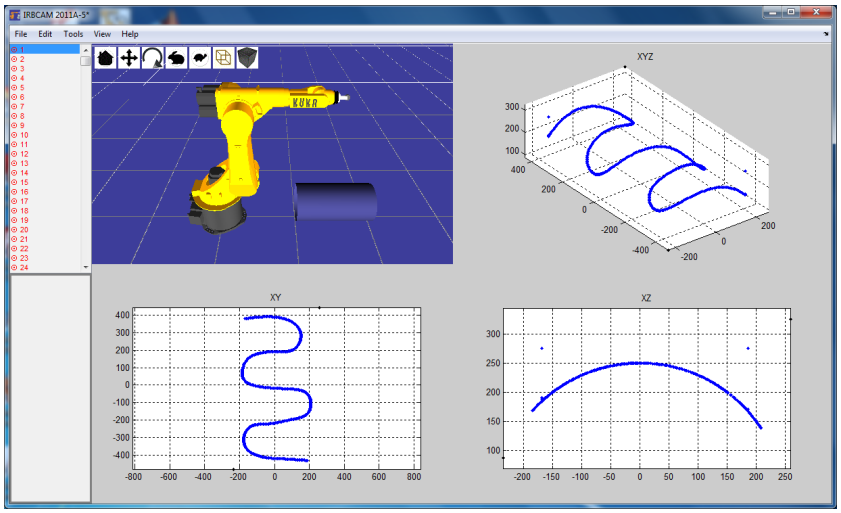


Figure 94: Combination View and Toolpath roller.apt.

Load the toolpath 'Cylinder.apt'. The Combination View (CTRL 2) should look like Fig. 94. Next, choose Tools - Configure Path (CTRL K) as shown in Fig. 95. In this example the roller tool is already aligned with the toolpath, hence the tool roll angle can be kept at 0 degrees. With the 'Cylinder.apt' toolpath, the roller will rotate 180 degrees. Often with the Roller Mode, the 6th axis will reach it's joint limits. In such cases, the 'Wrist Down' option may be selected in Fig. 95, or the tool roll angle could be defined at $\pm$ 180 degrees to avoid that the 6th axis will reach it's joint limit. In this example, the toolpath can be configured either with 'Wrist Down' and the roll angle starting at 0 degrees, or no 'Wrist Down' but tool roll angle starting at -180 degrees.

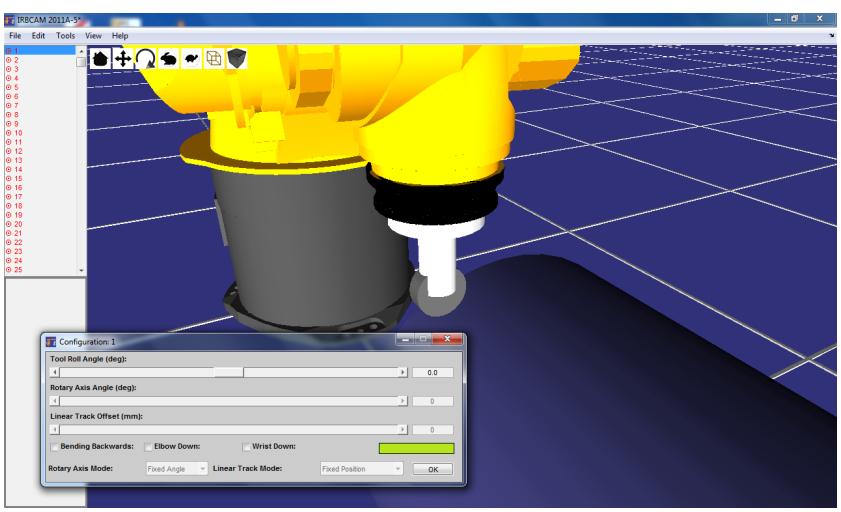


Figure 95: Configure Toolpath.

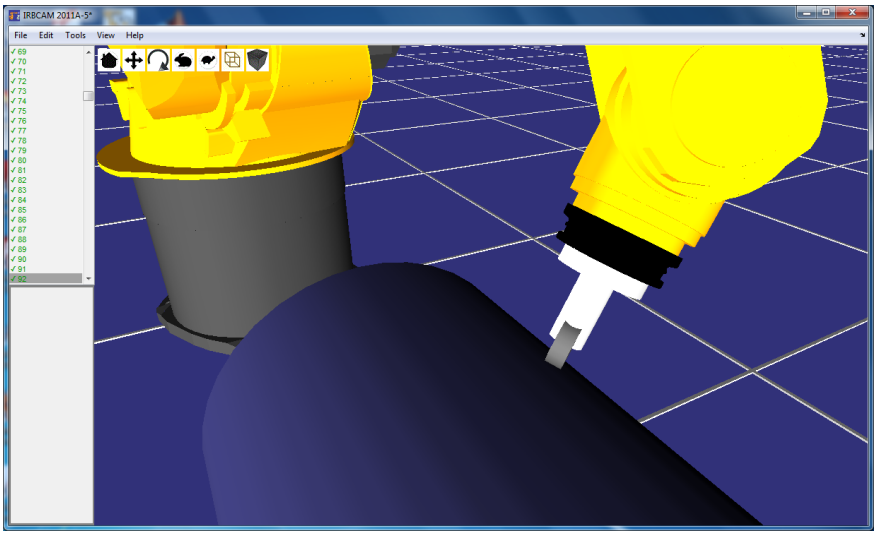


Figure 96: Roller orientation at the first corner.

Fig. 96 shows the roller tool at the first turning curve. In Roller Mode, the tool will rotate such that the 6th axis of the robot is always aligned in the direction of the toolpath.

7.10 ABB IRB140 Robot and Fixed Axis 6

In certain applications, when a straight tool is used, it may be useful to lock axis 6 to a fixed value. A straight tool means that the X and Y values in the tool data are both equal to zero. When axis 6 is locked, the robot effectively becomes a 5-axis machine.

In this example, select an IRB140 robot with the user frame located at (500,0,0), select the tool Spindle-Roller and no external axes or user objects. Fig. 97 shows the options screen (CTRL+J). When there is a straight tool defined in the station (like Spindle-Roller), it is possible to select a fixed 6th axis as shown. Load the

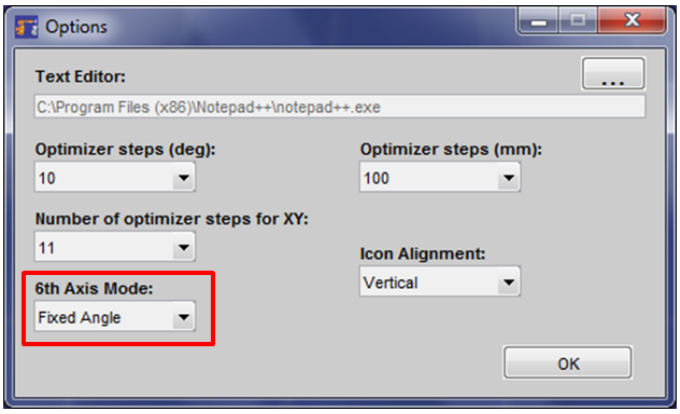


Figure 97: Options screen.

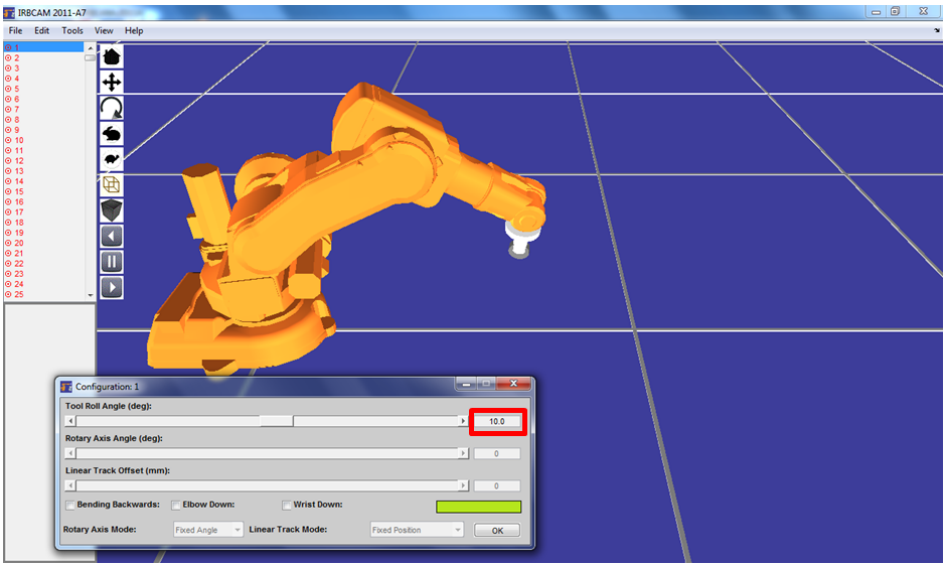


Figure 98: Configure path. The tool roll (axis 6 value) is set to 10°.

APT file 'apt\Example.apt' from IRBCAM's installation directory. Select the Path Configurator (CTRL+K), as illustrated in Fig. 98. When axis 6 is fixed as in this example, the 'Tool Roll Angle' in the configurator becomes the fixed angle for axis 6. In this example, set this angle to 10°, as marked with a red rectangle in Fig. 98.

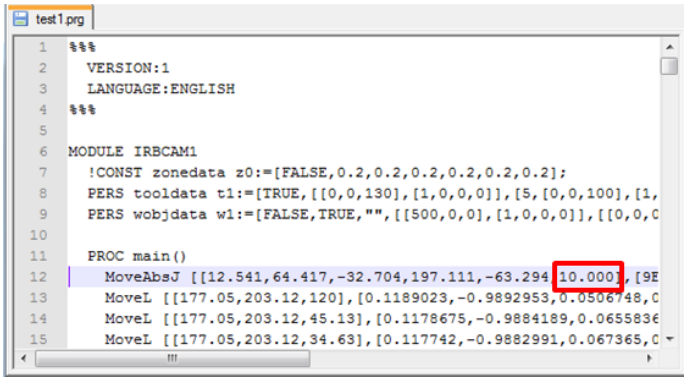


Figure 99: RAPID code. Note the axis 6 value in the MoveAbsJ command.

Fig. 99 shows the generated RAPID code. Note the axis 6 value in the first MoveAbsJ command (marked with a red rectangle). The axis 6 angle is fixed to 10°.

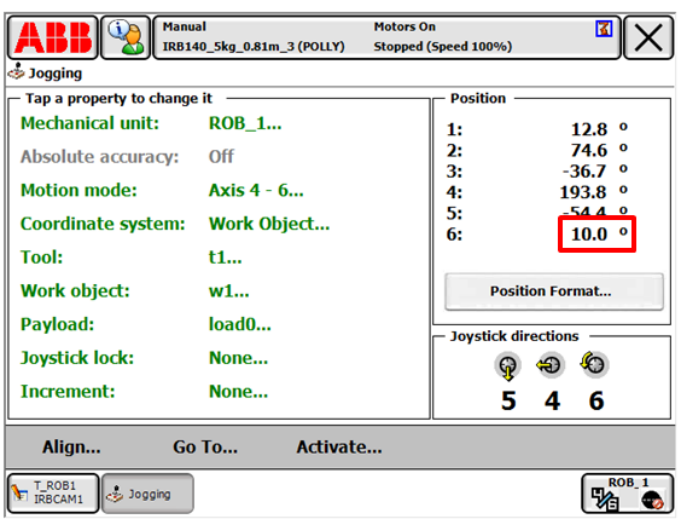


Figure 100: FlexPendant showing the angle of axis 6.

Fig. 100 shows the FlexPendant in jogging mode. Start the RAPID program and execute one of the MoveL commands after the initial MoveAbsJ. Stop the program and go to the jogging screen. As seen from Fig. 100 the angle of axis 6 will remain at 10° during the entire robot program.

7.11 ABB IRB2400 Hanging Robot

In this example the use of IRBCAM with a hanging robot will be demonstrated. Figure 101 shows an example 3D Contour toolpath in SurfCam. The toolpath is

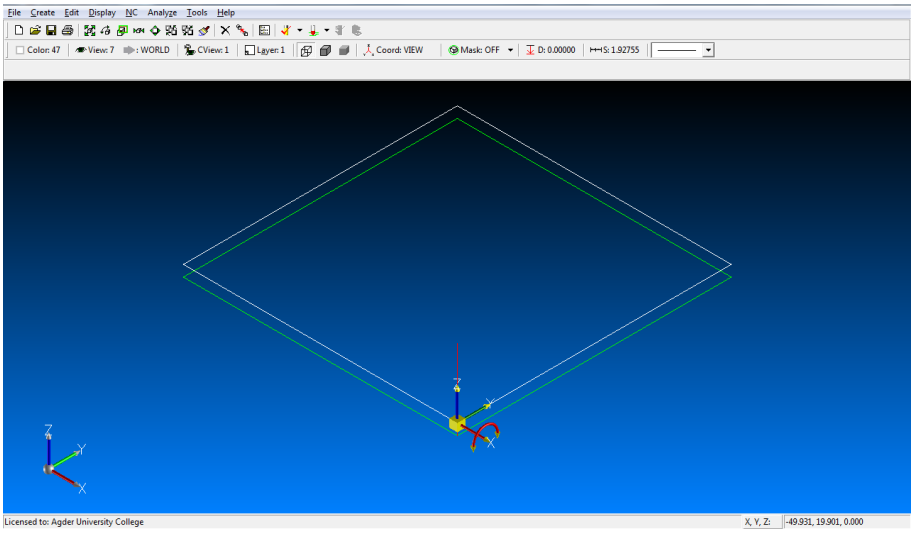


Figure 101: Example toolpath in SurfCam - 3D Contour.

aligned with the XY plane and the APT file is exported with Construction View (CVIEW) 1. The five points in the square can also easily be created inside IRBCAM. The points are: (0,0,0), (100,0,0), (100,100,0), (0,100,0) and (0,0,0). Fig-

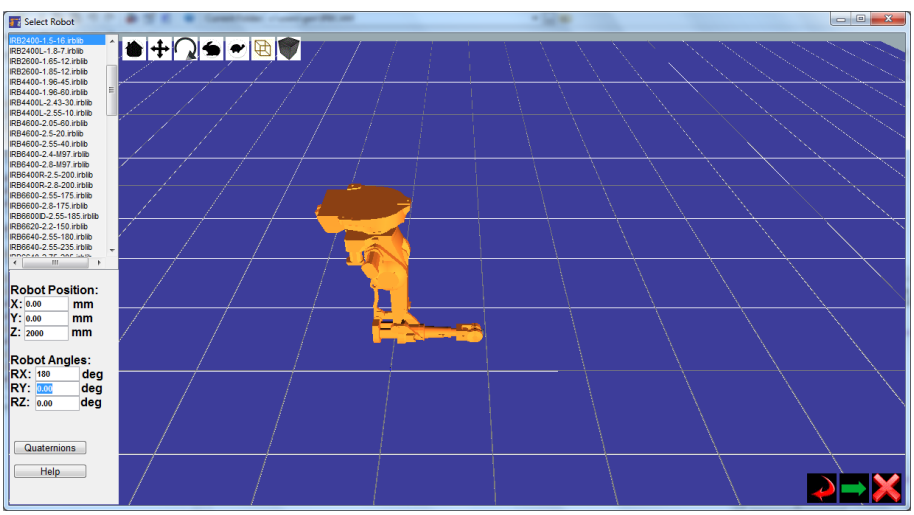


Figure 102: Robot Configuration - Hanging IRB2400, $Z = 2000$mm and $R_X = 180^o$.

ure 102 shows the configuration screen in IRBCAM. Select the IRB2400-1.5-16 robot and change the Z-position to 2000mm and the X-rotation to $R_X = 180^o$.

Select the Generic Tool with parameters $X = 300$mm, $Z = 300$mm and $R_Y = 90^o$ and define the user frame at position $X = 1000$mm and $Z = 500$mm, as illustrated in Figures 103 and 104.

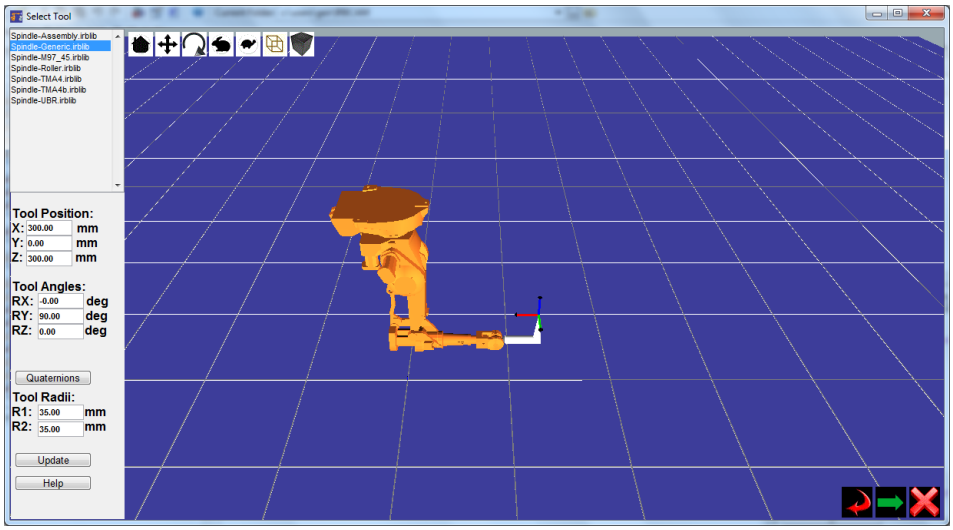


Figure 103: Tool Definition - Generic Tool $X = Z = 300$mm and $R_Y = 90^o$.

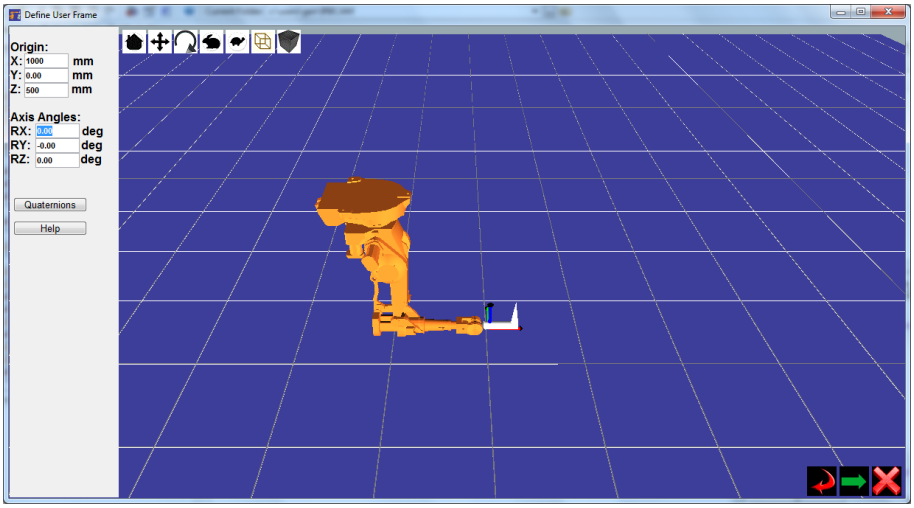


Figure 104: User Frame Definition - $X = 1000$mm and $Z = 500$mm.

Load the APT file into IRBCAM and configure the path with a tool roll angle of 0°, as illustrated in Figure 105. Export the toolpath to a rapid program, for example in a file called ex1.prg.

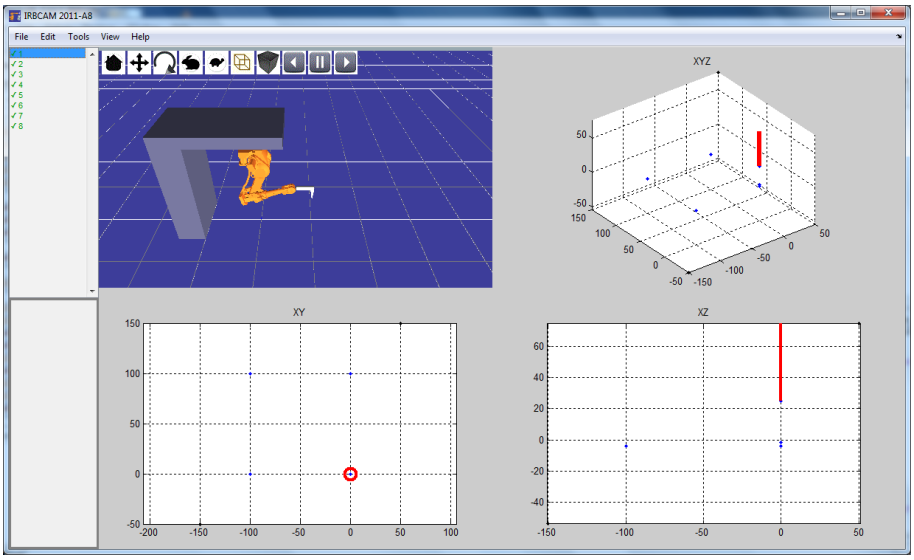


Figure 105: APT Path Loaded into IRBCAM. Configured with Tool Roll Angle of 0°.

In the ABB FlexPendant, change the robot base definition to $Z = 2.0$m and the quaternion (0,1,0,0), which is the same as the rotation $R_X = 180^o$, as shown in Figure 106. Finally, to verify that the RAPID program generated by IRBCAM is correct, the program ex1.prg can be loaded into the FlexPendant and verified with RobotStudio, as shown in Figure 107.

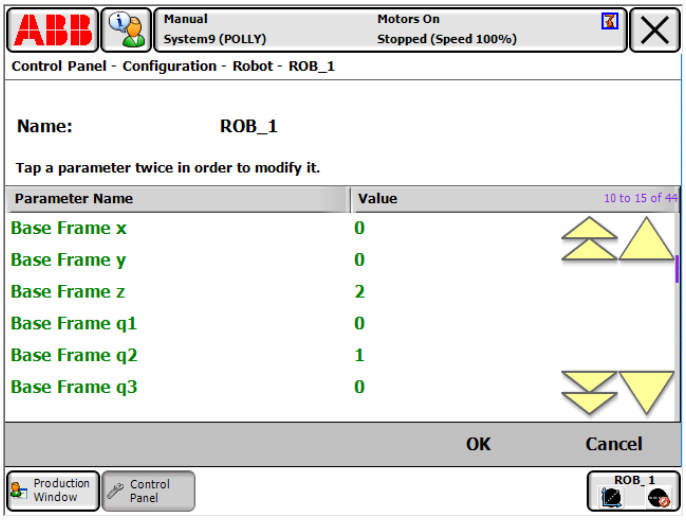


Figure 106: Robot Base Definition in the FlexPendant. $Z = 2.0$m and Quaternion = [0,1,0,0].

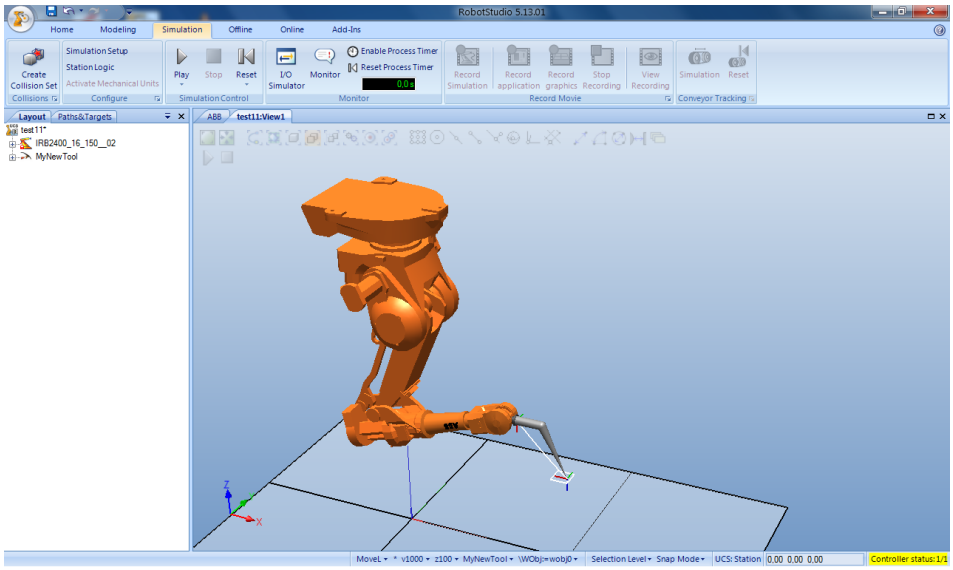


Figure 107: Path Verification and Tool Trace in RobotStudio.

7.12 ABB IRB6400 Robot with Actuated Linear Table

This section demonstrates the use of a linear actuated table. Placing the user object on an actuated linear table can sometimes be a cost-efficient alternative to placing the robot on a linear track. Figure 108 shows the option screen in the 'New Station Wizard'. Select the last option (Station with Actuated Table), then 'Next'. Figure 109 shows the definition screen for the actuated table. In this example, set

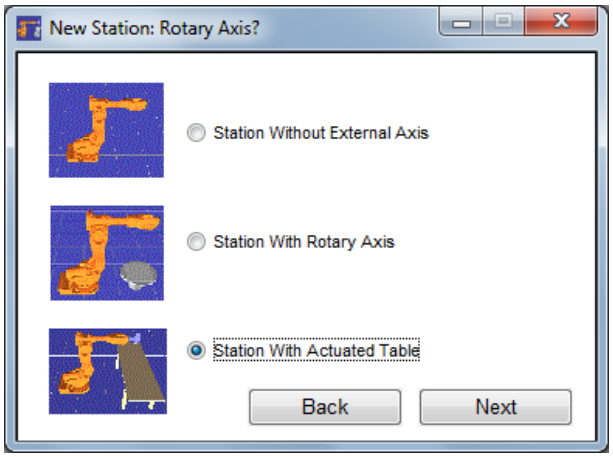


Figure 108: Selection of Actuated Table.

the Width and Length of the table to 1500mm and 6000mm, respectively. Leave the table in the default position (X=2000,Y=0,Z=500).

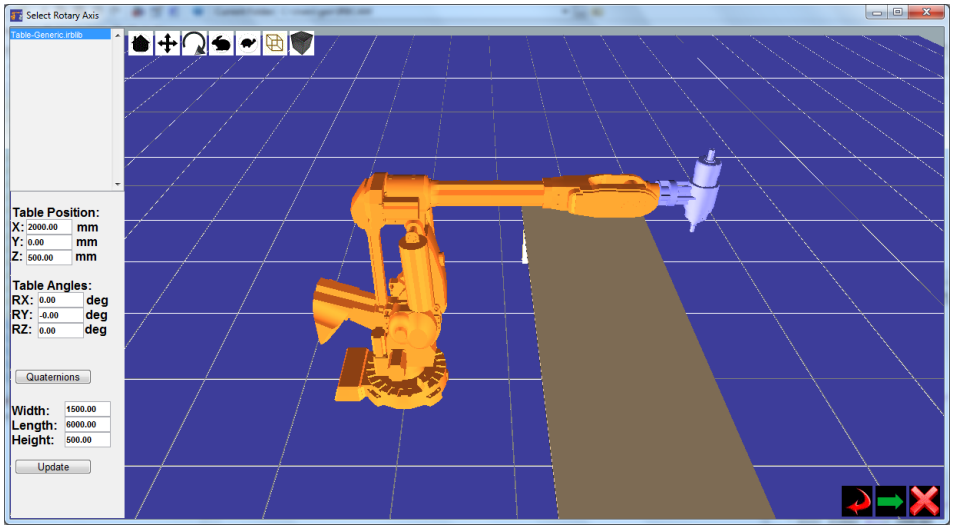


Figure 109: Definition of Linear Actuated Table.

Next, select 'Station with User Object' as shown in Figure 110.

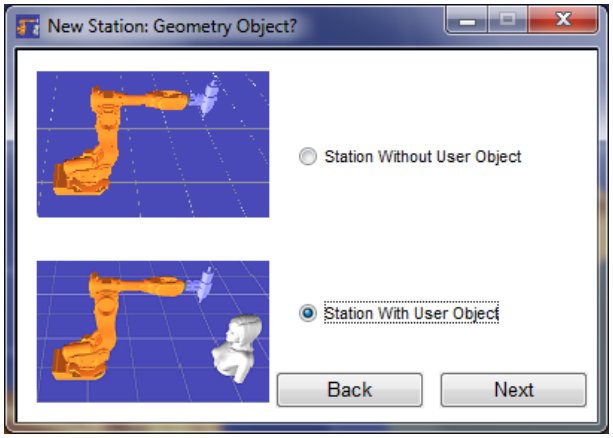


Figure 110: Addition of User Geometry.

Select the BoatHull CAD file as illustrated in Figure 111. Move the object to Z=710mm and rotate RZ by 90° to place it correctly on the machining table.

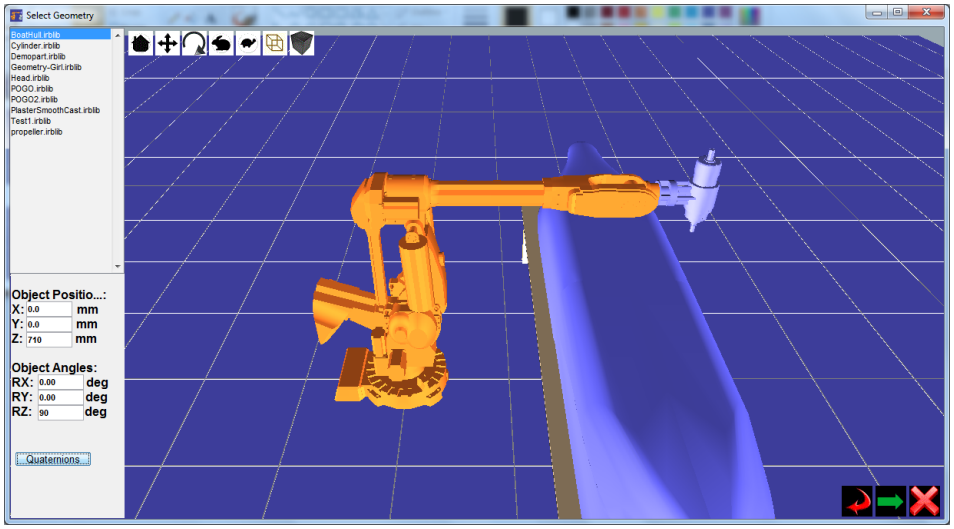


Figure 111: User Geometry: Definition of Object Frame.

Figure 112 shows the Path Configuration window (Menu item 'Tool - Configure Path' or press CTRL+K). Define the following parameters: Tool Roll Angle = 145°, Actuated Table Offset = -2000mm, Actuated Table Mode = Dynamic. Press 'OK'.

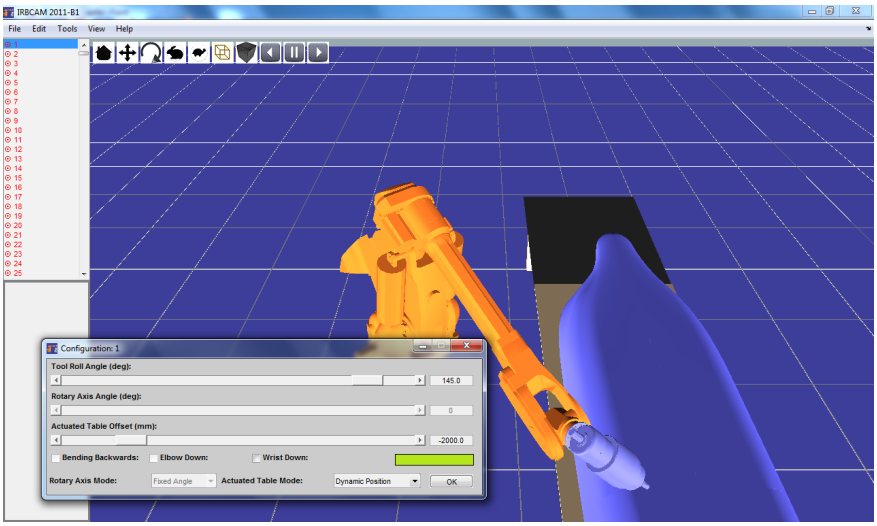


Figure 112: Robot with Actuated Table: Path Configuration.

After the path has been configured, select Menu item 'View - Actuated Table' (or press CTRL+9). The plot shown in Figure 113 will appear. This plot shows the dynamic table motion, from -2000mm to +837.07mm, a total travel range of 2837.07mm.

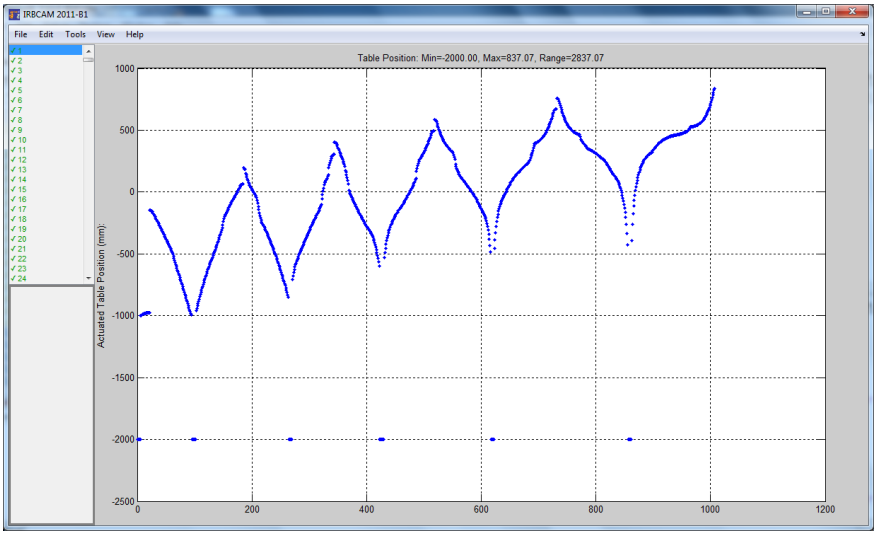


Figure 113: Generated Linear Table Motion.

7.13 Motoman UP50N Robot and 5-Axis Toolpath

The setup for Motoman robots is slightly different compared to other robot types, as will be illustrated in this example. First, select the 'New Station Wizard' and 'Station Without Linear Track' as shown in Figures 114 and 115.

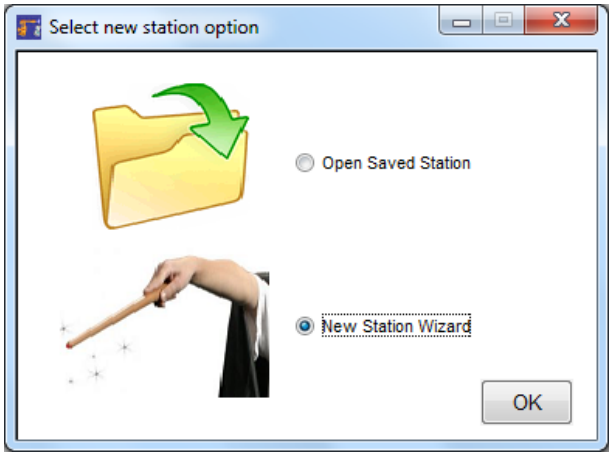


Figure 114: New Station Wizard.

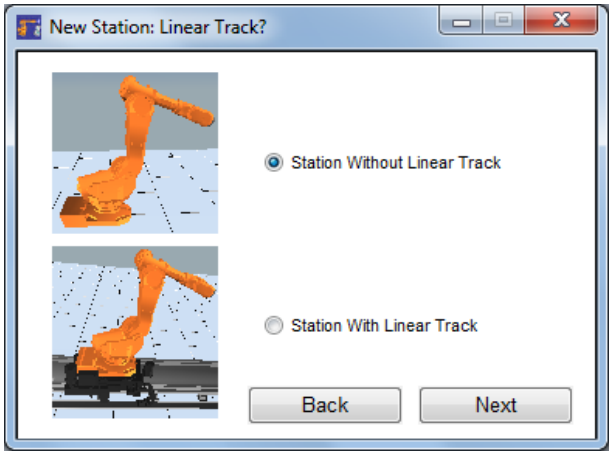


Figure 115: Station Without Linear Track.

Next, select the Motoman UP50N robot, as shown in Figure 116.

One difference between Motoman robots and for example ABB and Kuka robots, is the tool axis definition. The Motoman tool coordinate system is rotated 180° about the Z-axis compared to ABB and Kuka. For the Motoman controller NX100, the tool definition is stored in a file named TOOL.CND. An example tool definition from the TOOL.CND file could look like this:

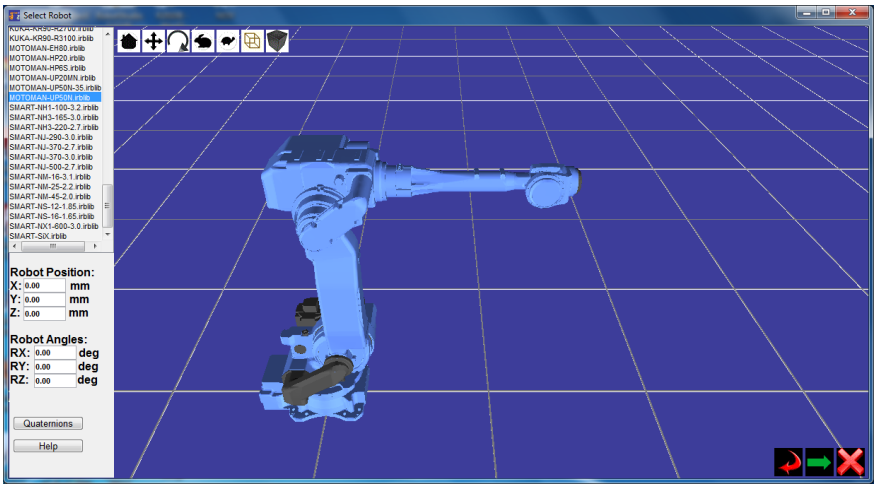


Figure 116: Selection of Motoman-UP50N Robot.

```
///NAME SPINDLE1
-65.955,40.805,184.031,0.00,-60.00,-31.25
```

In IRBCAM, modified values of the Motoman tool values are entered, as shown in Figure 117. The modifications are summarised as follows:

IRBCAM Tool X	=	**-** Motoman Tool X	=	65.955
IRBCAM Tool Y	=	**-** Motoman Tool Y	=	-40.805
IRBCAM Tool Z	=	Motoman Tool Z	=	184.031
IRBCAM Tool RX	=	Motoman Tool RX	=	0.00^o
IRBCAM Tool RY	=	Motoman Tool RY	=	-60.00^o
IRBCAM Tool RZ	=	**180°+** Motoman Tool RZ	=	148.75^o

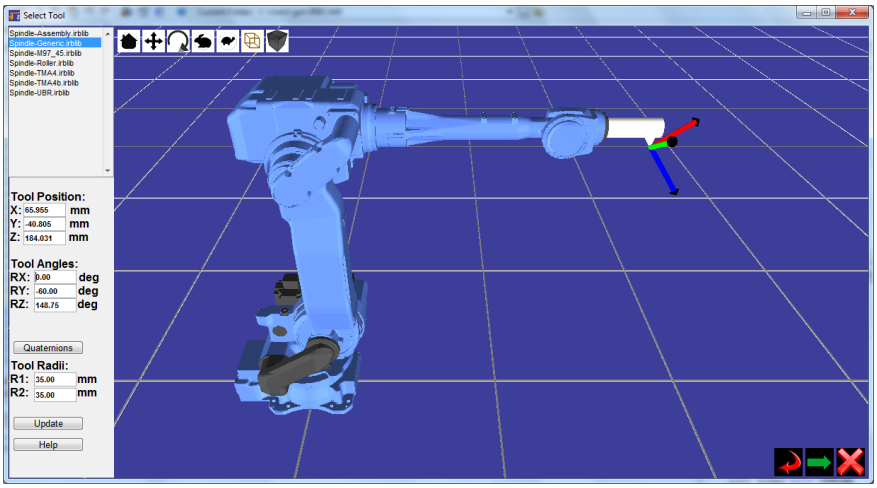


Figure 117: Definition of Motoman Tool Coordinates.

Another common tool angle definition is the 90° pitch tool, shown for example in Fig. 47. With this tool, the Motoman TOOL.CND file would look like this:

```
///NAME SPINDLE1
-352.00,0.00,350.00,0.00,-90.00,0.00
```

The IRBCAM angles for this tool would be: RX=0, RY=-90, RZ=180 (which have the same effect as RX=180, RY=-90, RZ=0).

The coloured lines (vectors) at the tip of the tool in Figure 117 indicate the different axes. The red vector equals the IRBCAM Tool X vector, the green vector equals the IRBCAM Tool Y vector while the blue vector indicates the IRBCAM Tool Z vector.

Next, select 'Station Without External Axis' as shown in Figure 118.

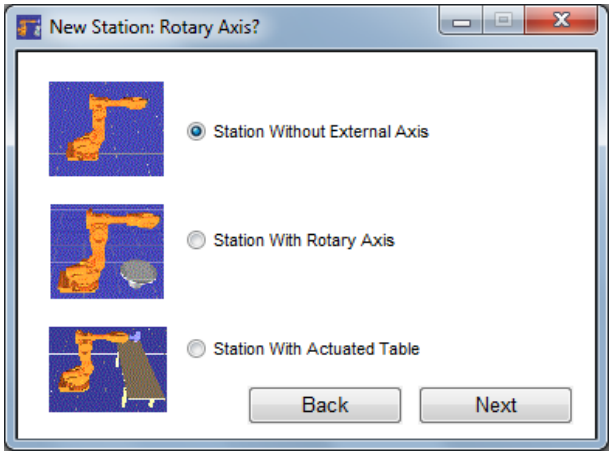


Figure 118: Station Without External Axis.

On the Motoman NX100 controller, the user frame definition is located in a file called UFRAME.CND. A typical UFRAME.CND file looks like the following:

```
//UFRAME 1
///NAME UFR1
///TOOL 1
///GROUP 1,0,0,0,0,0,0,0
///PULSE
////RORG C000=-1,-30974,-82179,0,0,-9636
////RXX C001=-1,14179,-30518,-3,1826,-9634
////RXY C002=38618,-14980,-66577,-55267,3682,10533
////BUSER 1153.952,-11.781,335.815,0.00,0.00,0.00
```

The coordinates C000, C001, C002 are here listed as PULSE, and can not be used in IRBCAM. However, the last line, the ////BUSER, shows the corresponding

X,Y,Z,RX,RY,RZ values for the user frame. These values should be entered into the IRBCAM definition screen, as shown in Figure 119. The user frame definition for Motoman robots is the same as for ABB and Kuka robots, hence no modifications of the values are necessary for the Motoman UFRAME.

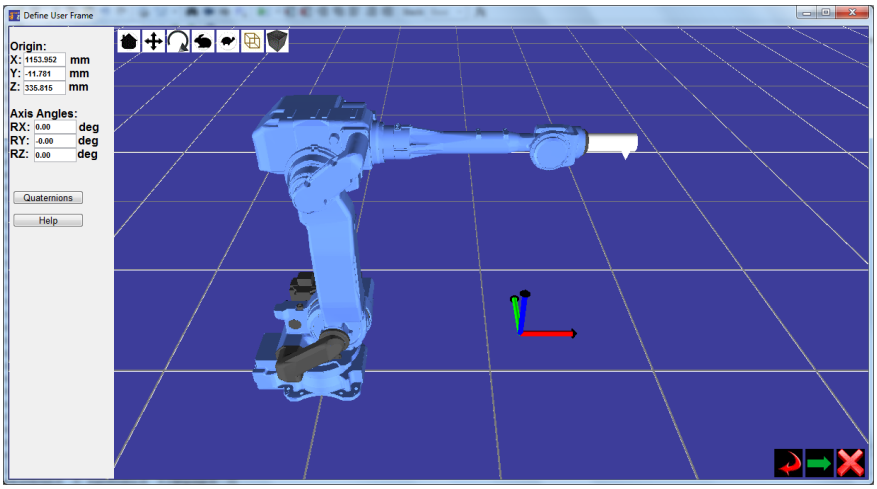


Figure 119: User Frame Location.

Next, select a 'Station Without User Object' as shown in Figure 120 (left).

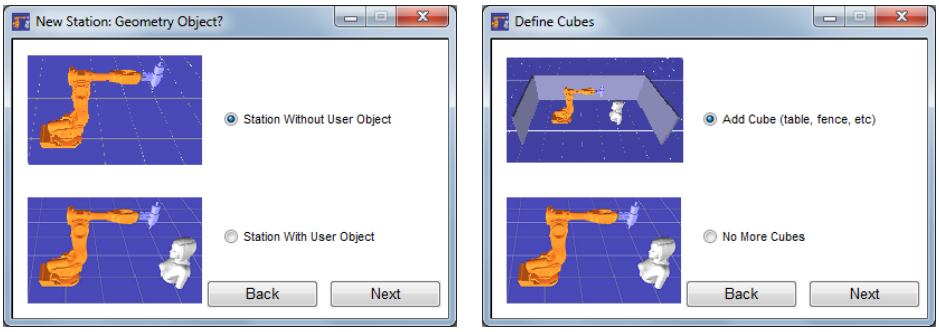


Figure 120: Station Without User Object (left). Cube Definition (right).

Next, we would like to add a fixed machining table to the station. In Figure 120 (right) select 'Add Cube'. In Figure 121 define the position of the cube (the machining table) to X=1154, Y=0 and Z=117. Define the size of the cube to X=1000, Y=1000 and Z=335 (all values in millimeters). After this cube has been defined, select 'No More Cubes' and save the station as an '.IRB' file.

Figure 122 shows the main IRBCAM window after the APT file 'apt Example.apt' has been loaded. The red numbers in the menu at the top left shows the imported robot positions, in total there should be 766 positions when a minimum distance of 1.0mm was selected when importing the APT file.

In the menus, select 'Tools - Configure Path' (or press CTRL+K) and the configurator screen should appear as seen at the bottom left in Figure 122. With a Tool Roll Angle of -118.8 degrees, the entire path should configure without any problems.

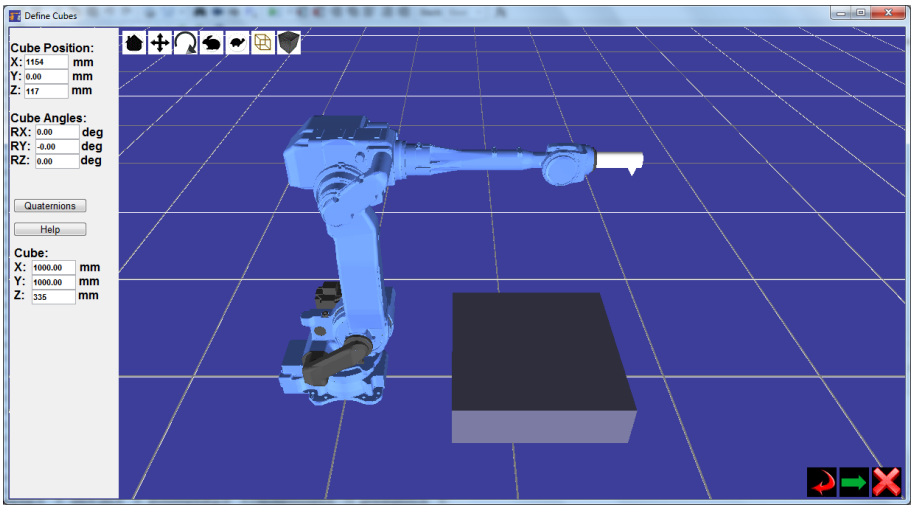


Figure 121: Size and Position of Fixed Machining Table.

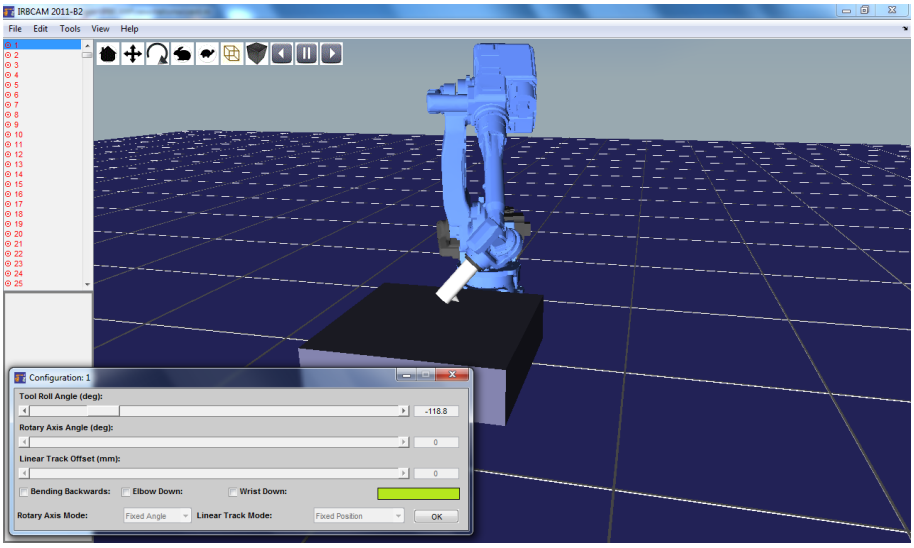


Figure 122: Path Configuration.

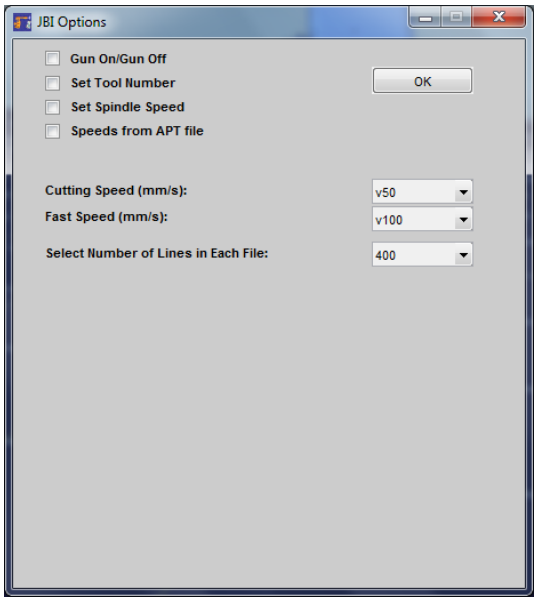


Figure 123: Motoman Inform III (JBI) Export Options.

Finally, the Inform III robot program for the NX100 controller can be exported. Figure 123 shows the JBI export options available from the menu 'File - Motoman'. The NX100 controller has an upper limit on how many points can be stored in each file. Normally, this limit is 1000 coordinates. IRBCAM will split long programs into several JBI files. In the IRBCAM export options you can define the maximum number of lines stored in each file. In this example with the 'Example.apt' file, there are 766 coordinates. Hence, if you select a maximum of 400 coordinates in each file, IRBCAM will export two JBI files. If you name the export file 'TEST', IRBCAM will generate the two files 'TEST0001.JBI' and 'TEST0002.JBI'. The first file will contain 400 coordinates, while the second file will contain the remaining 366 coordinates. The last line in 'TEST0001.JBI' will call the job 'TEST0002' automatically, so the entire toolpath can be executed without user intervention. Other options are the robot speeds, as well as options for enabling tool change and gun on/off, see section 10 for further details about these options.

7.14 ABB IRB6650S and Stationary Tool

In this example the use of a stationary tool will be demonstrated. First, select

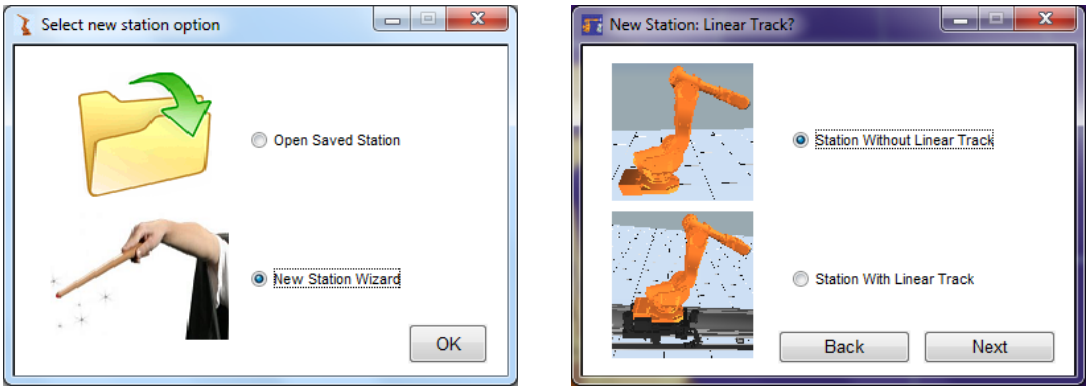


Figure 124: Initial options when creating new station.

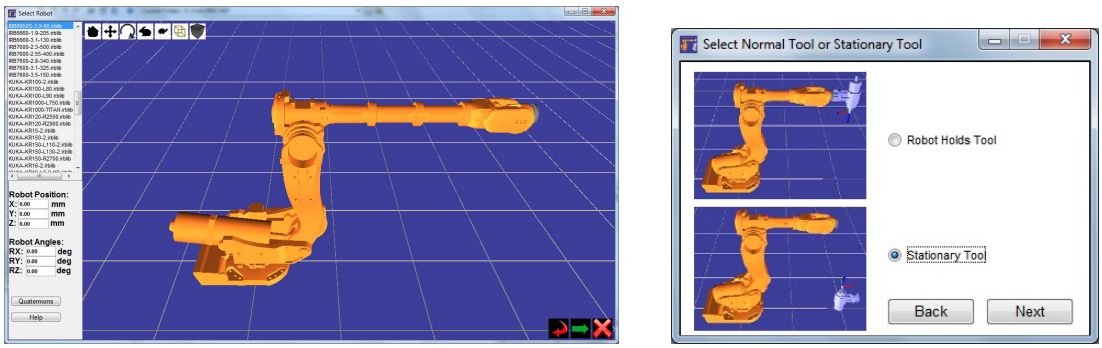


Figure 125: Selection of IRB6650S-3.9-90 robot and type of tool.

'New Station Wizard' and 'Station Without Linear Track' as shown in Figure 124. Next, select the IRB6650S-3.9m-90kg robot as shown in Figure 125. This robot is selected because of the long arm which will be needed in this example. Next, select 'Stationary Tool' as shown in Figure 125. Next, select 'Spindle-UBR' and place the base position of this spindle at X=3200, Y=0, Z=2400 and rotation RZ=180 degrees as shown in Figure 126. Note that the base position is different from the tool centre point (TCP). The TCP can modified by clicking on 'Tool Tip' at the bottom left of the screen. In this example, the default tool tip position will be used. Next, select the default user frame location at X=Y=Z=0 and rotations RX=RY=RZ=0. When a stationary tool is used, the user frame is attached to axis 6 of the robot. Next, select 'Station With User Object' and the object model 'Geometry-Girl' as shown in Figure 127.

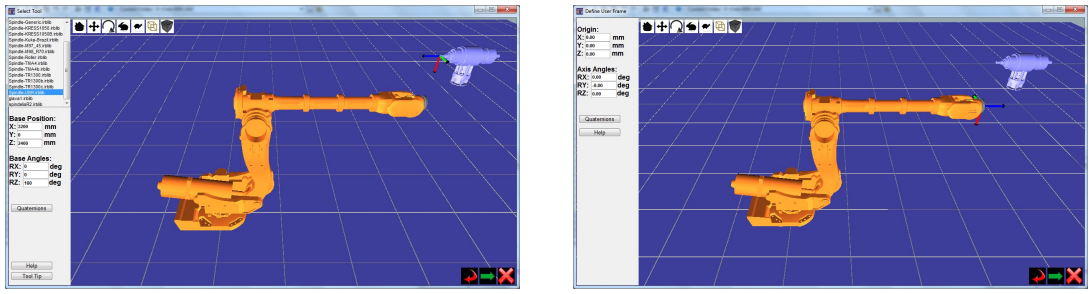


Figure 126: Selection of Spindle-UBR and User Frame.

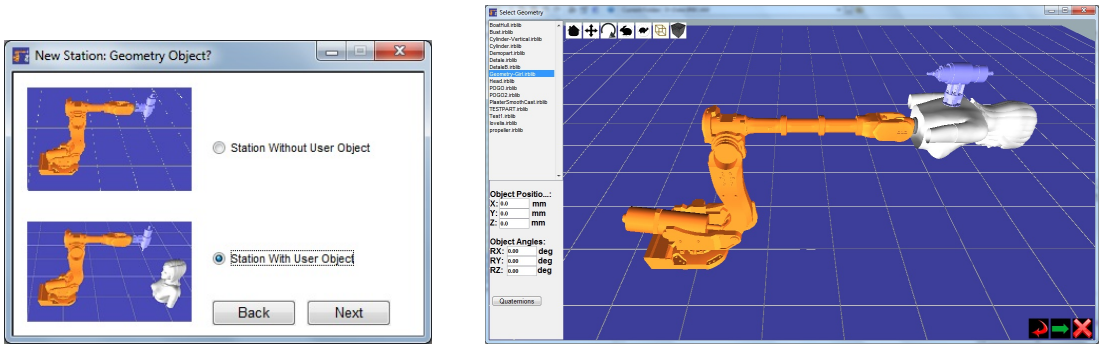


Figure 127: Selection of User Object.

Next, select 'Add Cubes' and define the first cube location at X=3600, Y=0, Z=2700 with dimensions X=1000, Y=1000, Z=600 as shown in Figure 128. Define the second cube at X=4200, Y=0, Z=1500 with dimensions X=400, Y=1000, Z=3000. These two cubes represent the support structure for the stationary spindle.

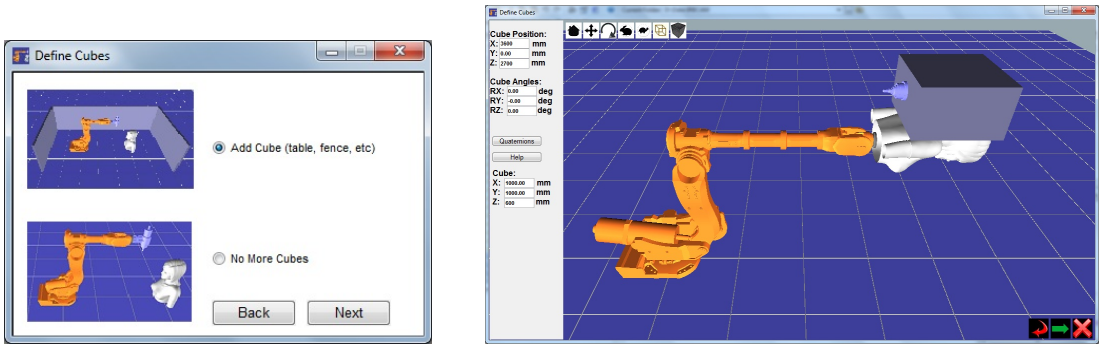


Figure 128: Definition of Cubes.

Afer the station has been saved, load (CTRL+L) the APT file girl.apt. Then, try to configure the path by selecting 'Tools - Configure Path' or press CTRL+K and the configuration window shown in Figure 129 will appear. In this window, select 'Wrist Down' and 'Tool Roll Angle - Dynamic Angle'. After clicking on 'OK' IRBCAM will

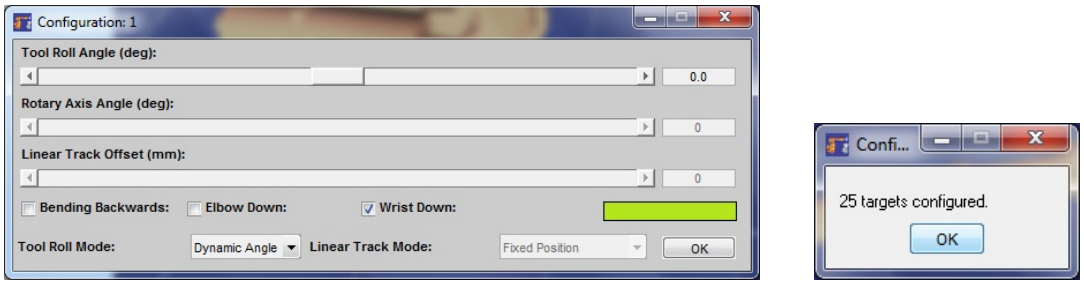


Figure 129: Path Configuration: First attempt.

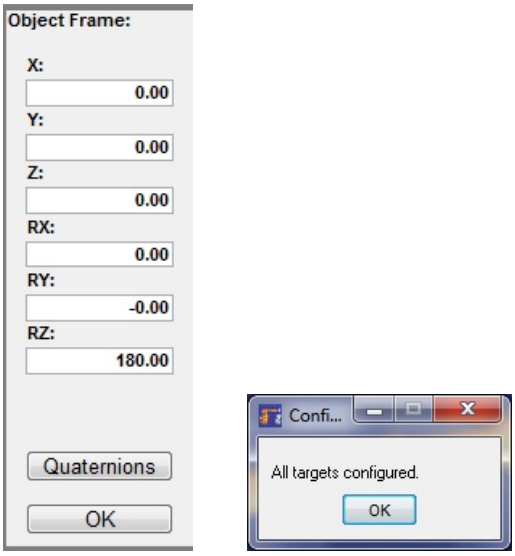


Figure 130: Path Configuration: Modification of Object Frame.

only manage to configure the first 25 targets, as shown in Figure 129 (right). The reason for this limitation, is the angle restriction of ± 360 degrees for the 6th axis of the IRB6650S. To solve this problem, the starting angle of axis 6 is important. For example, it is not possible to turn 360 degrees in the positive direction if axis 6 starts target 1 with a positive angle. If, however, target 1 starts with a negative angle for axis 6, the full 360 degree turn in the positive direction will be possible. To overcome this problem, the user or object frame of the robot can be used. When a stationary tool is used, the user and object frames are attached to the robot. By rotating one of these two frames by 180 degrees, the 3D model of the girl will also rotate by 180 degrees. Select 'Edit - Object Frame' and set the rotation RZ=180 as shown in Figure 130. This operation will also change the initial angle of axis 6. Finally, select 'Tools - Configure Path' (or CTRL+K) and configure the path with the same parameters as in the first attempt. This time all the targets will be configured successfully.

Figure 131 shows the IRBCAM station after the successful configuration with a

stationary tool and full 360 degree rotations using axis 6 of the robot. The reason why the long arm IRB6650S robot is used in this example, is the relatively large object distance in the Z direction. If a robot with shorter arm had been used, axis 5 of the robot would have reached its angular limits at the bottom or the top of the user object.

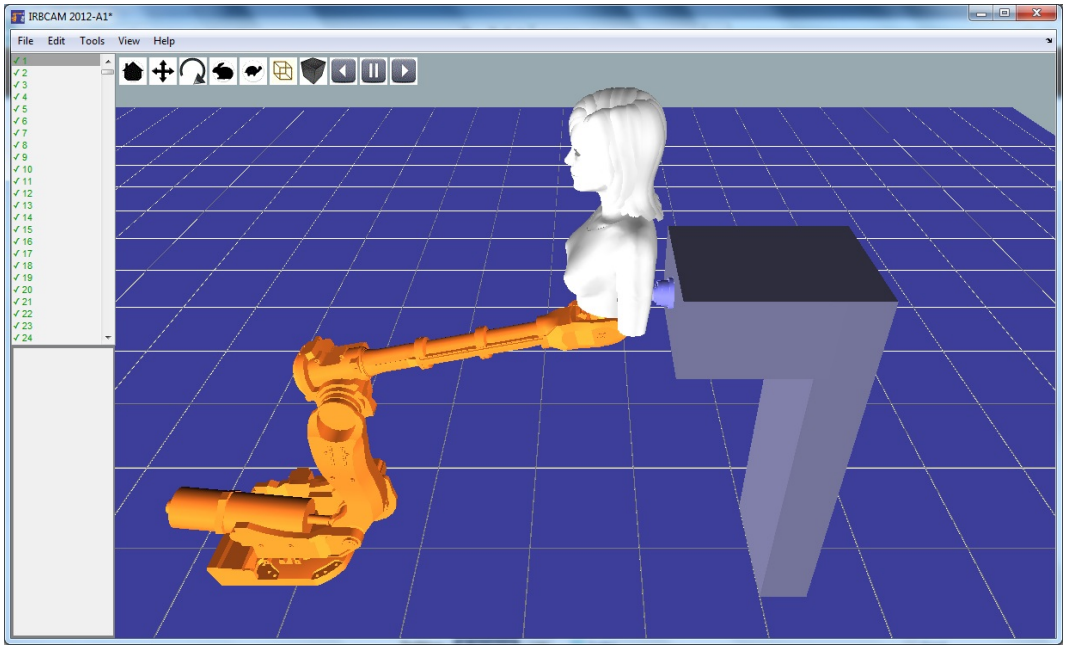


Figure 131: Stationary Tool: Station After Successful Configuration.

7.15 ABB IRB6400 Configuring Complex 5-Axis Paths

In this example the configuration of difficult 5-axis toolpaths will be demonstrated. Often, it is the case that one single parameter setting will not be able to configure a complex toolpath. In such cases, it will be beneficial to split the toolpath into smaller sections, configure each section separately and finally merge the configured sections into the final toolpath.

In this example the configuration and merging of several APT files will be demonstrated. First, define a new station consisting of an IRB6400-2.4-M97 robot, a generic tool with data equal to (X=350, Y=0, Z=80, RX=0, RY=90, RZ=0) and user frame equal to (X=1700, Y=130, Z=530, RX=RY=RZ=0). Next, load the APT file 'Merge-1.apt' (File - APT - Load APT or use CTRL+L). The window in

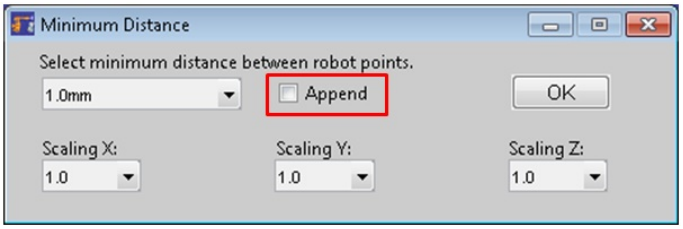


Figure 132: Load APT with Append option introduced in version 2012-B1.

Fig. 132 will appear and notice the 'Append' checkbox. To start with, leave this option unchecked and press OK. Select 'View - Combination View or use CTRL+2). The IRBCAM window should look like in Fig. 133. Next, open Tools - Optimizer

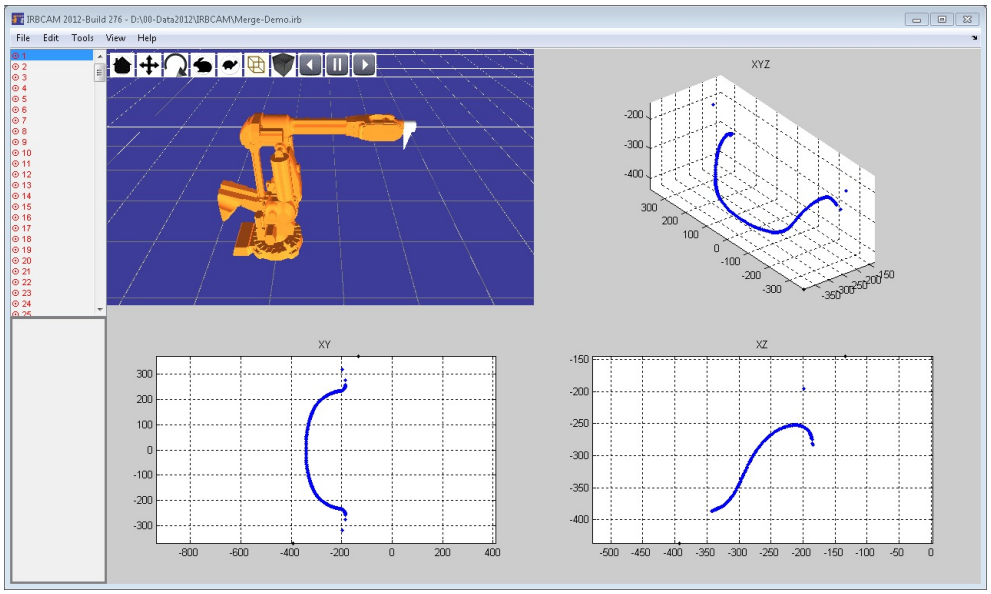


Figure 133: Combination View after loading Merge-1.apt.

(CTRL+I). With 'Wrist Down' and 'Dynamic Angle 1', tool roll angles in the range -180 to -120 and +120 to +180 degrees are OK as seen in Fig. 134. In this exam-

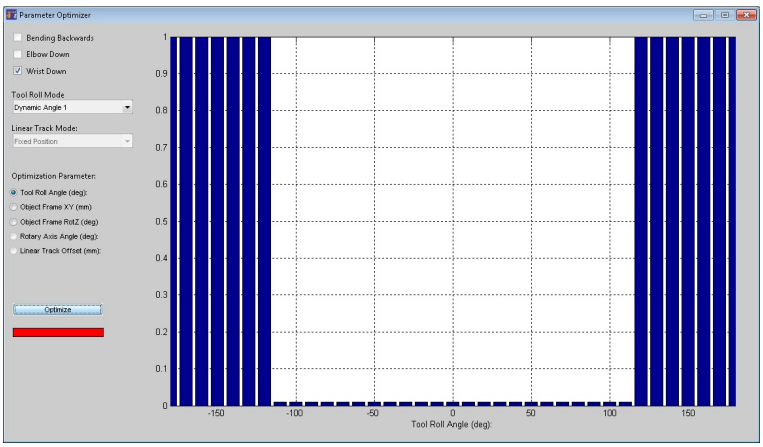


Figure 134: Optimizer for Merge-1.apt.

ple, let us choose the tool roll angle equal to +130 degrees and configure the path (CTRL+K) with the settings shown in Fig. 135 (left). After configuration, right-click

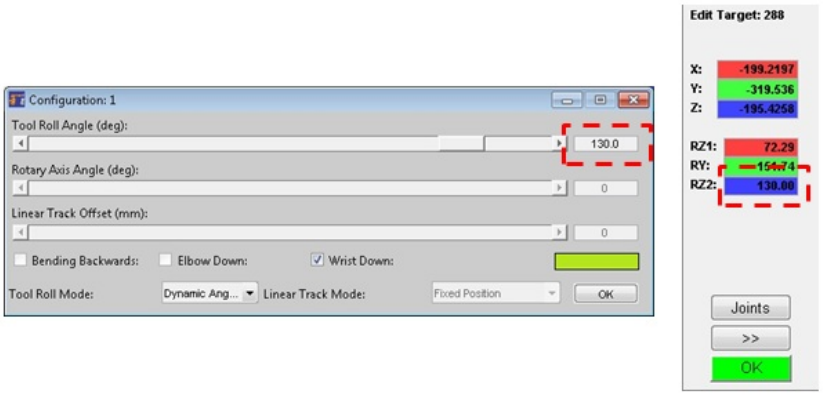


Figure 135: Configuration of Merge-1.apt.

on the last target in the list, number 288 in this case. The information in Fig. 135 (right) should appear. Notice that the angle RZ2 still equals +130 at the end of the path (equal to the initial choice of +130 degrees).

Next, load the APT file 'Merge-2.apt' (CTRL+L) without the Append option. Select the Optimizer (CTRL+I) and the tool roll angles from +50 to +90 degrees should be OK, as seen in Fig. 136. Next, Next, let us try to combine the two paths: 'Merge-1.apt' and 'Merge-2.apt'. When loading 'Merge-2.apt', select the 'Append' option. Select 'View - Combination View' (CTRL+2) and the IRBCAM window should look like in Fig. 137.

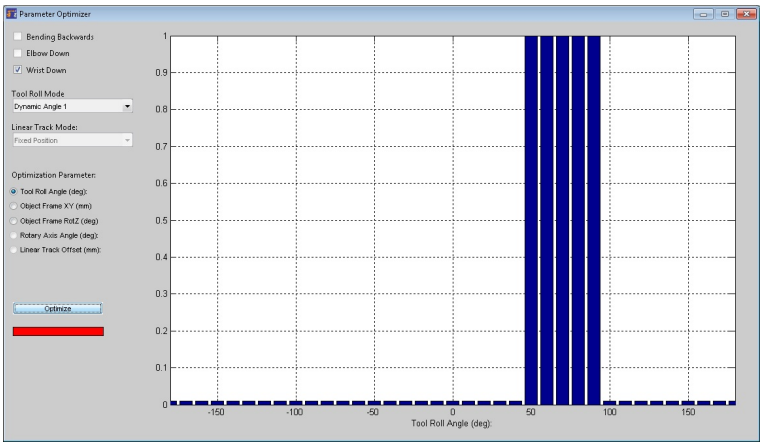


Figure 136: Optimizer for Merge-2.apt.

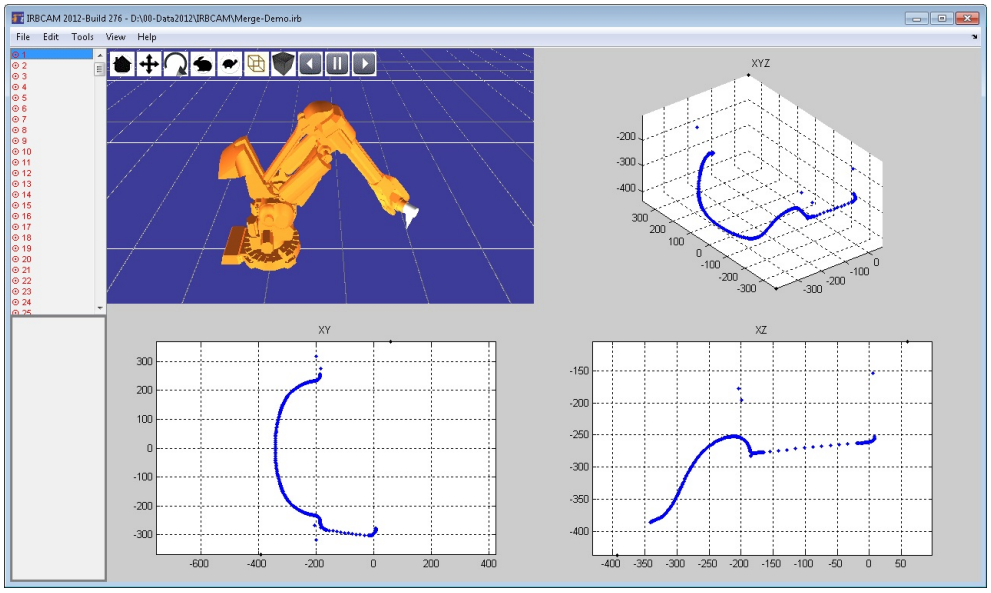


Figure 137: Combination View after loading both Merge-1 and Merge-2 with Append option.

The configuration of this combined toolpath with an initial tool roll angle of +130 degrees will fail. In fact, the Optimizer will not find any initial tool roll angle which will configure the combined path of 'Merge-1' and 'Merge-2', even if the two paths configured individually. The problem is the transition between the two paths. 'Merge-1' ended with an RZ2 value equal to +130 degrees, while 'Merge-2' was configured with a tool roll angle of +50 degrees. This difference of +80 degrees is too abrupt and IRBCAM will not allow it. To make a smooth transition between 'Merge-1' and 'Merge-2' with additional targets added, first select the function 'Tools - Maximim Distance' (CTRL+M). The transition between the two paths occur between the targets 287-290. Set the start to target 287 and the end to target 290, as seen in Fig. 138, and leave the distance to the default value of 10mm. After the Maxi-

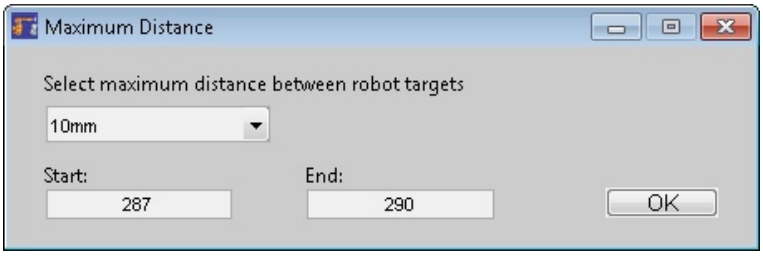


Figure 138: Maximum distance for transition between Merge-1 and Merge-2.

mum Distance operation, additional targets have been added to the transition area, see Fig. 139 (dotted red rectangle in XYZ plot). Next, open the 'Tools - Options'

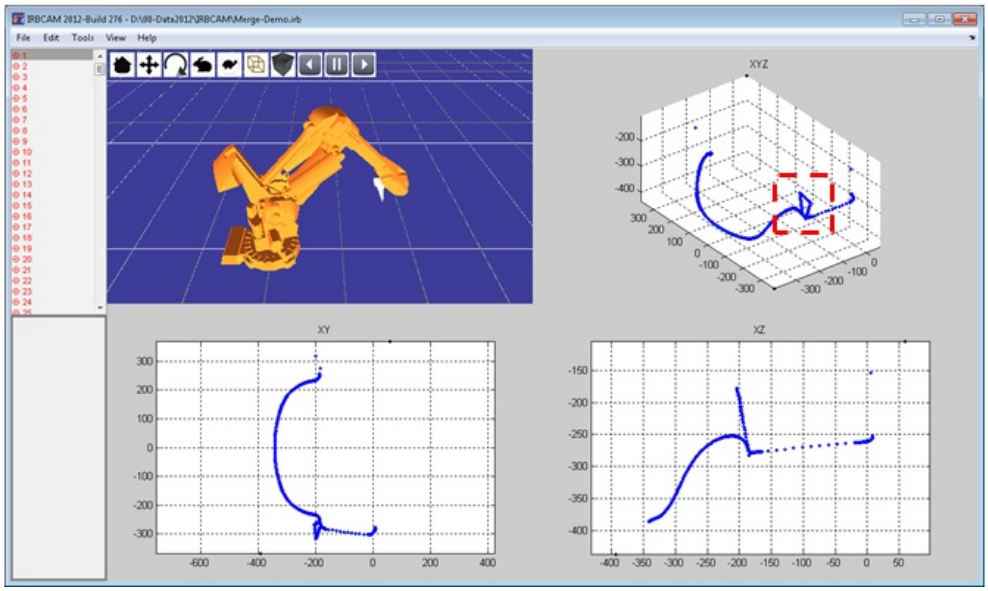


Figure 139: Combination View after Maximum Distance operation in transition area.

(CTRL+J) and select 'Enable RZ2', see Fig. 140 (left). This option allows IRBCAM

to force a lock on the RZ2 values during configuration. Right-click on target number 287 and lock RZ2 to +130 degrees (start point of transition). Then click on OK. Right-click on target 335 and lock RZ2 to +50 degrees (end point of transition), then OK. See Fig. 140 (right) for the correct settings. Now, the combined toolpath consisting of 'Merge-1' and 'Merge-2' can be configured with the same settings as in Fig. 135 (left). At this stage, save the station file (CTRL+S) before continuing the example.

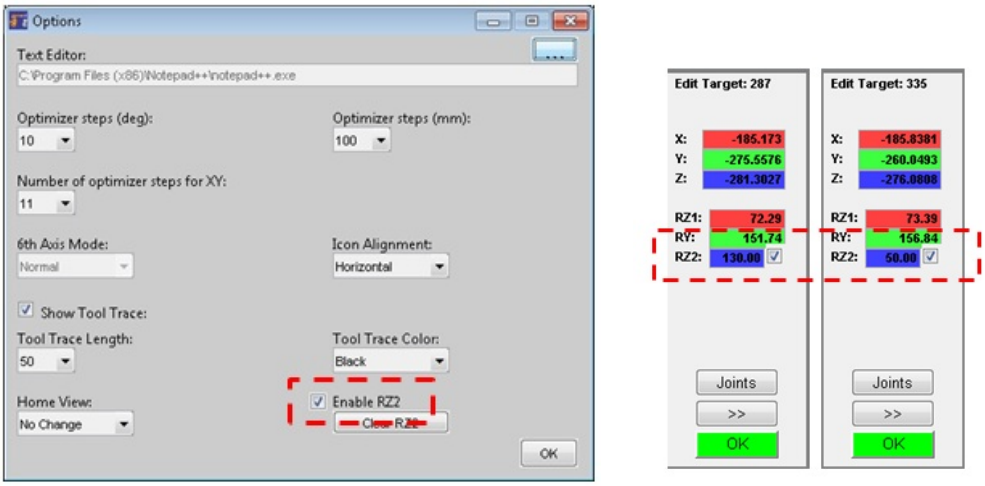


Figure 140: Left: Enable RZ2 option. Right: RZ2 locked values for targets 287 and 335.

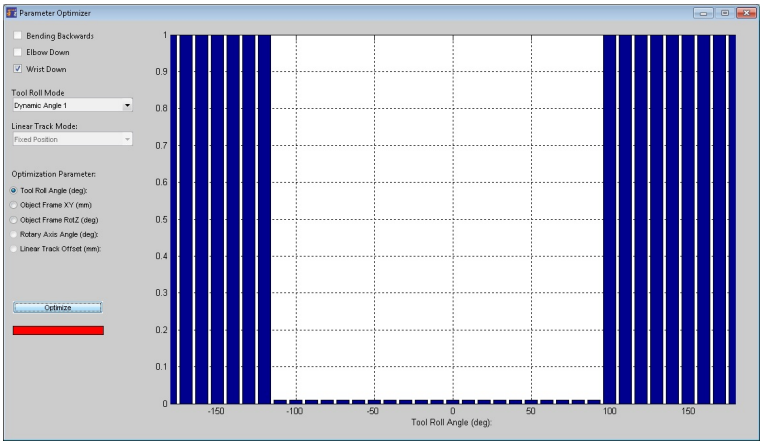


Figure 141: Optimizer for Merge-3.apt.

Next, load 'Merge-3.apt' without the Append option and run the Optimizer. As seen from Fig. 141, angles -180 t0 -120 degrees and +100 to +180 degrees are possible. The angle of +100 will be chosen, since it is closest to the +50 degrees selected for 'Merge-2'. Open the saved station (CTRL+O) containing the paths 'Merge-1' and

'Merge-2'. Then load 'Merge-3' with the Append option. Target 412 is the last one in 'Merge-2', while target 413 is the fist one in 'Merge-3'. For this transition, IRBCAM is able to configure it without the need for inserting additional transition targets with the Maximum Distance function. Hence, set the RZ2 value equal to +50 degrees for target 412 and +100 degrees for target 413, see Fig. 142. It should now be possible to configure paths 'Merge-1' to 'Merge-3' with the configuration settings in Fig. 135 (left). Save the station (CTRL+S) before continuing.

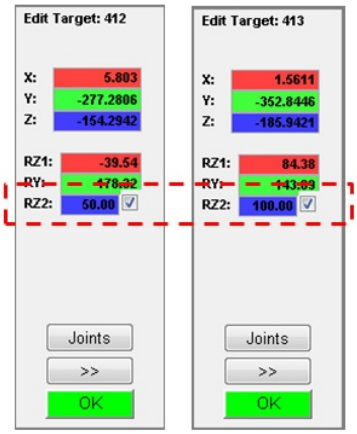


Figure 142: RZ2 locked values for transition targets 412 and 413.

Finally, load the path 'Merge-4.apt' without the Append option and run the Optimizer. As seen in Fig. 143 the tool roll angles -100 to -80 degrees are possible. Load back the previously saved station (CTRL+O) and add 'Merge-4' with the Append option.

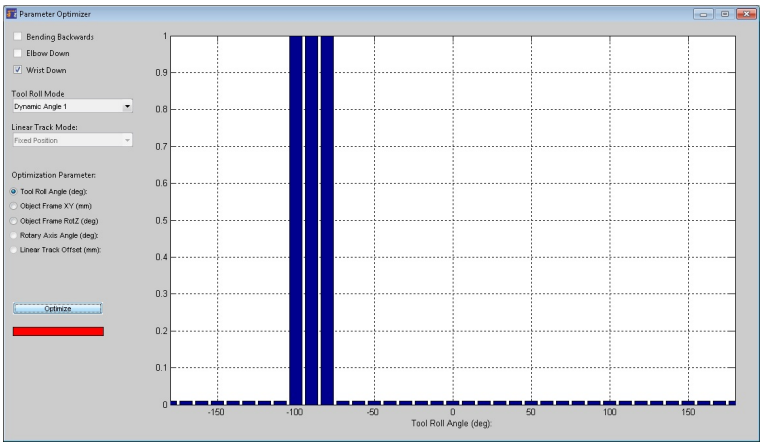


Figure 143: Optimizer for Merge-4.apt.

As for the previous transition, IRBCAM is able to handle the transition between 'Merge-3' and 'Merge-4' without the need for additional targets. Set RZ2 equal to

+146 degrees for target 940 (the last one in 'Merge-3') and -90 for target 941 (the first one in 'Merge-4'), see Fig. 144.

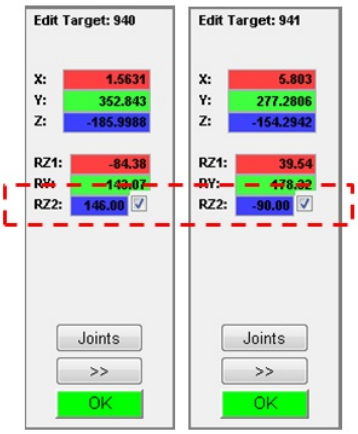


Figure 144: RZ2 locked values for transition targets 940 and 941.

The reason why it is +146 degrees for 'Merge-3' and not +100 which was defined for the start point of 'Merge-3' (target 413), is that the mode 'Dynamic Angle 1' is selected which allows IRBCAM to gradually modify the RZ2 angle from +100 to +146 degrees for the 'Merge-3' path. With the lock values in Fig. 142 IRBCAM is now finally able to configure the entire toolpath. The final IRBCAM window with the entire toolpath configured is shown in Fig. 145.

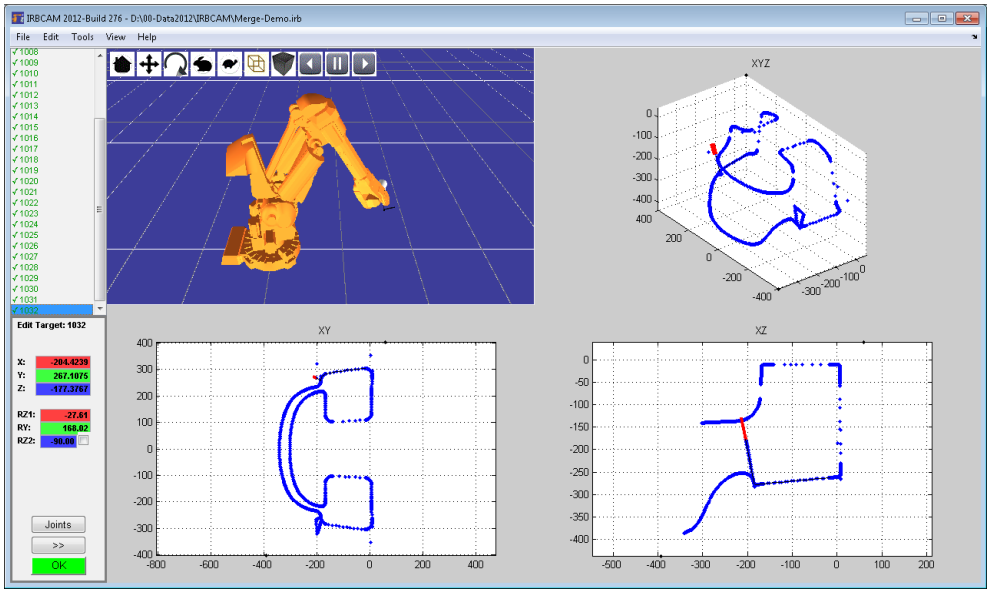


Figure 145: Combination View after final configured toolpath consisting of four files: 'Merge-1.apt' to 'Merge-4.apt'.

7.16 Gantry-Tau Milling of 8m Wind Turbine Blade

In this example a variant of a 5-degree-of-freedom (DOF) Gantry-Tau parallel kinematic machine (PKM) will be used to follow the surface of an 8m long wind turbine blade mounted on a rotary axis. This PKM is characterised by three arm clusters having 3-2-1 links, respectively. The PKM is designed to have a large workspace in one direction, while the link lengths are kept short which is an advantage for stiffness and accuracy. The PKM consists of three linear base actuators and three telescopic link actuators. The machine can manipulate the tool with 5 DOF, since the arm with a single telescopic link does not provide an additonal DOF. This telescopic link is used to increase the workspace and stiffness in the Y-direction. PKMs usually have limited maximum tool orientation angles compared to serial robots. The 5-DOF Gantry-Tau is able to achieve approximately ± 45 degree orientations in the global Z and X-directions. However, by using an additional rotary axis in the station, large and complex toolpaths requiring 360 degree tool re-orientations can be followed.

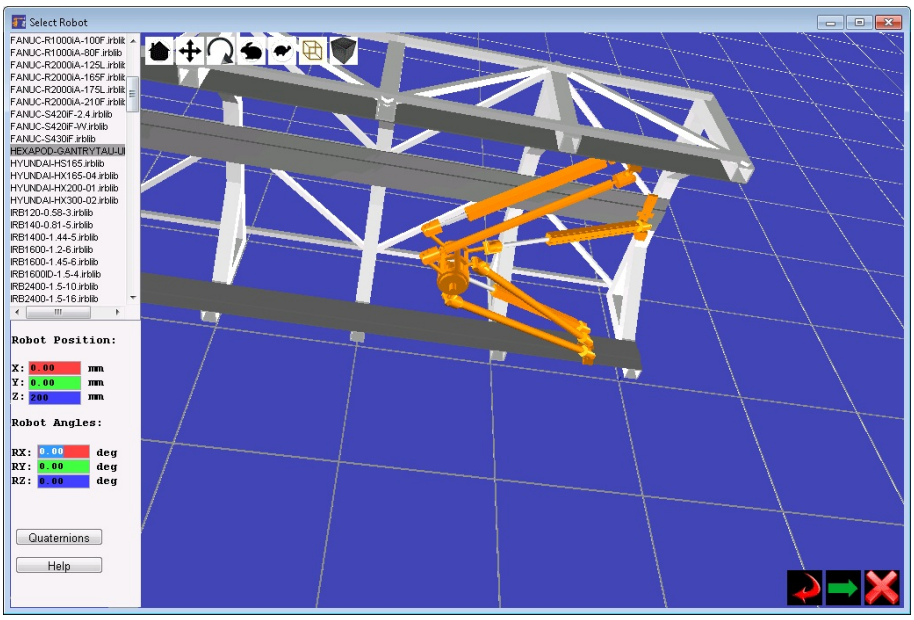


Figure 146: Parallel Kinematic Machine: HEXAPOD-GANTRYTAU-UIA.

Fig. 146 shows the "New Station Wizard" when selecting the Hexapod-GantryTau-UiA PKM. As shown in the figure, the base frame of the robot is lifted 200mm in the world Z-direction. Fig. 147 shows the tool selection wizard. Select "Spindle-Generic" with tool data X=Y=0, Z=150, RX=RY=RZ=0. Click on "Update" to update the tool graphics according to Fig. 147. Fig. 148 shows the selection of the rotary axis IRBP5000L-1000. Position the rotary axis at X=500, Y=Z=1000 and RX=90, RY=0, RZ=90. Fig. 149 shows the addition of the geometry user object Blade.irblib. Leave the geometry file in the default position X=Y=Z=RX=RY=RZ=0.

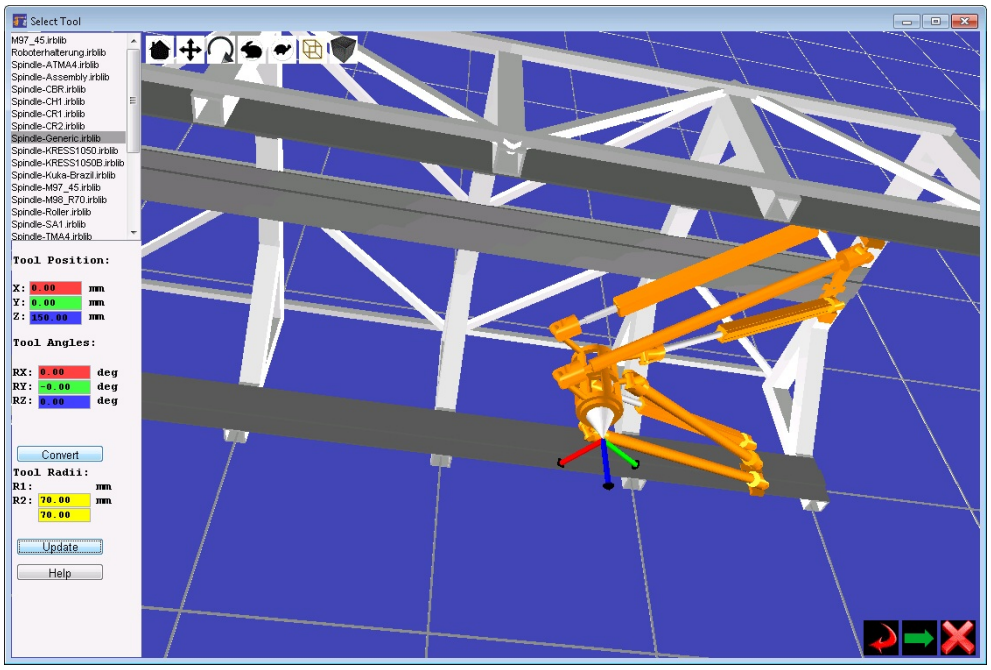


Figure 147: HEXAPOD-GANTRYTAU-UIA: Tool selection.

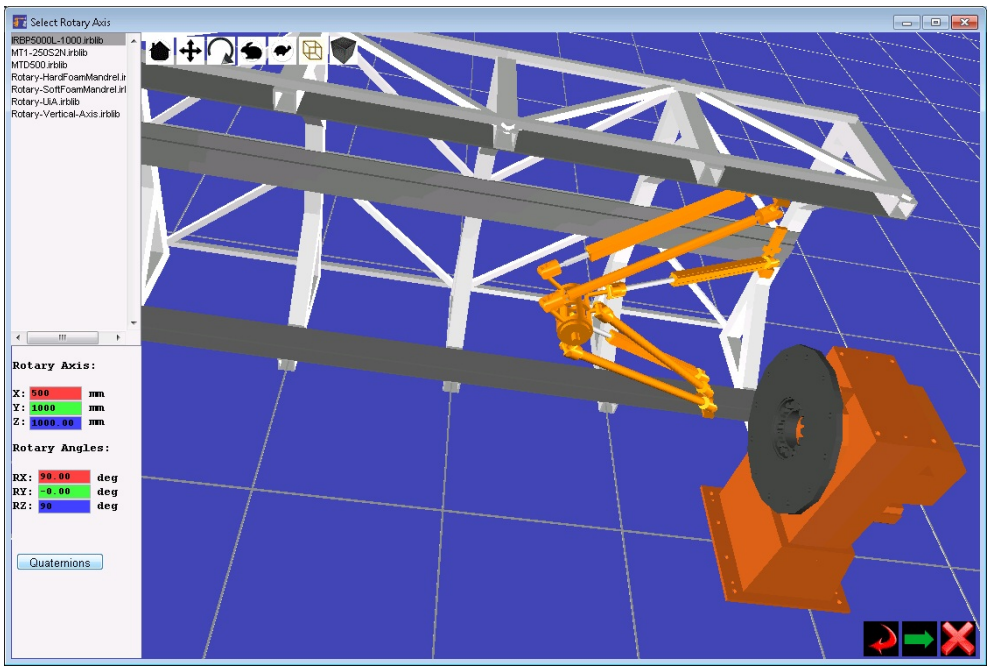


Figure 148: HEXAPOD-GANTRYTAU-UIA: Selection of Rotary Axis.

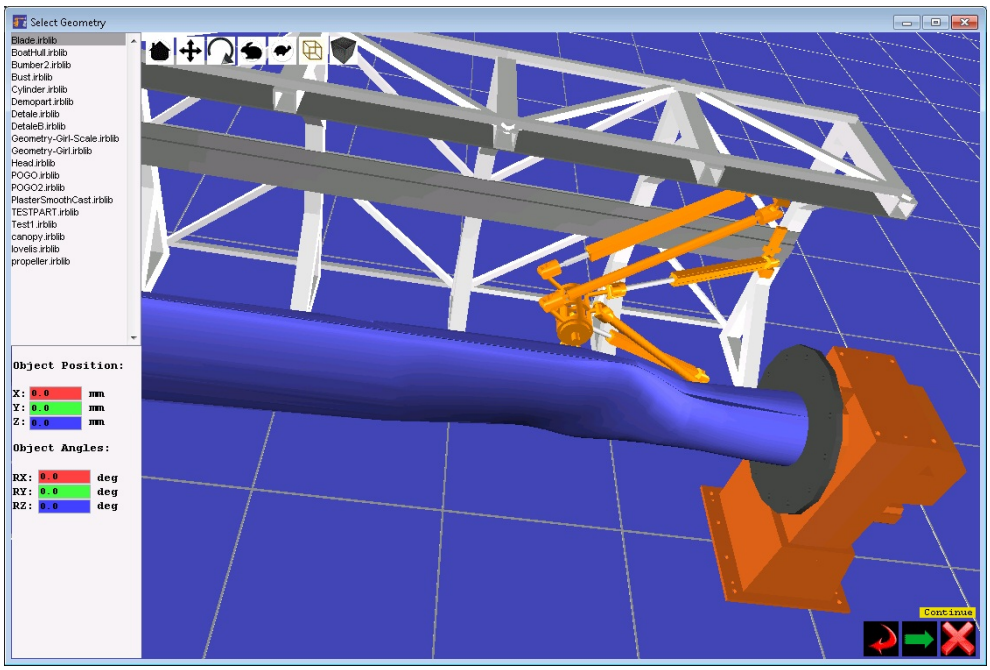


Figure 149: HEXAPOD-GANTRYTAU-UIA: Addition of Wind Turbine Blade.

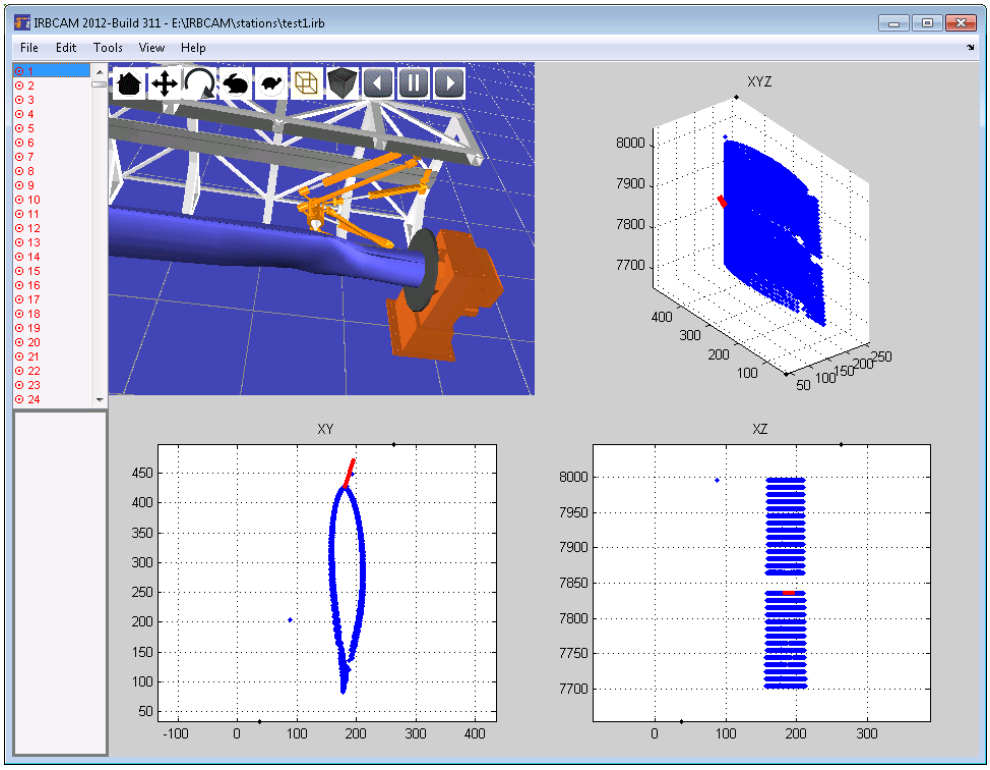


Figure 150: HEXAPOD-GANTRYTAU-UIA: Load APT file Blade.apt.

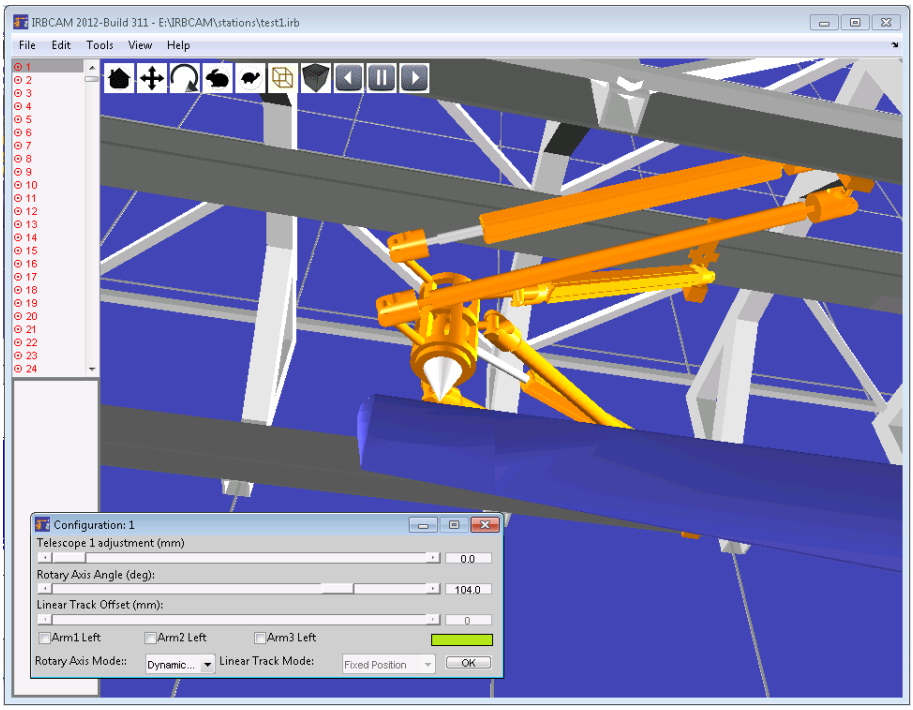


Figure 151: HEXAPOD-GANTRYTAU-UIA: Path Configuration Settings.

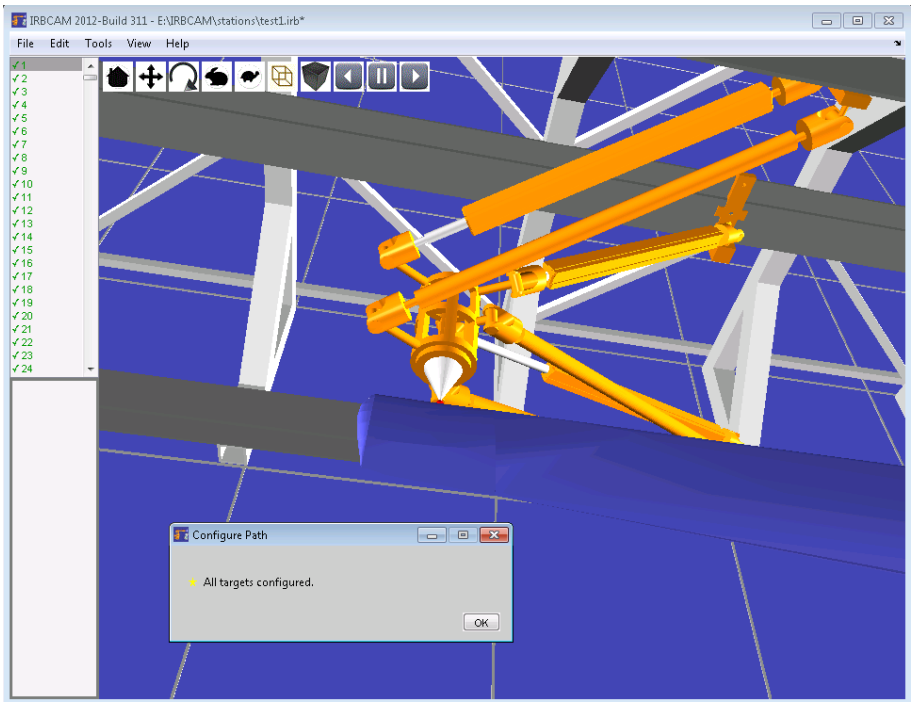


Figure 152: HEXAPOD-GANTRYTAU-UIA: Path Configured.

After the station file has been saved, Fig. 150 shows the main window in Combination View (CTRL+2) after the APT file Blade.apt has been loaded. Select "Tools - Configure Path" (or press CTRL+K) and the configuration window in Fig. 151 will appear. Select "Rotary Axis Angle" equal to 104 degrees and "Rotary Axis Mode" equal to "Dynamic Angle 1". The "Telescope 1 Adjustment" is a parameter which can be used to define the length of the single telescopic link. Normally, IRBCAM tries to keep this link length as short as possible for maximum stiffness, plus an additional length specified by "Telescope 1 Adjustment". In this example, the tool is relatively close to the base actuator for the single link. Hence, it is not possible to make the telescopic link very short and the parameter "Telescope 1 Adjustment" has no effect in this example. Fig. 152 shows the result after the path configuration. For each of the three arm clusters, the user can configure the arms to be either to the left or to the right of the tool. In this example, all three arms are configured as "Left". With these settings the approximately 4000 robot targets are successfully configured.

7.17 HYUNDAI HS165 and Drilling Cycles

In this example a Hyundai HS165 robot is used and an APT file containing drilling cycles. Fig. 153 shows the station after the "New Station Wizard" has completed. The user frame is defined at X=1500, Y=0, Z=750, RX=RY=RZ=0. Notice that Hyundai robots have the base frame rotated 90 degrees about the Z-axis compared to most other robot brands. In Fig. 153 Combination View (CTRL+2) was selected after the APT file Cycles.apt was loaded into the station. The file Cycles.apt contains the following drilling cycles:

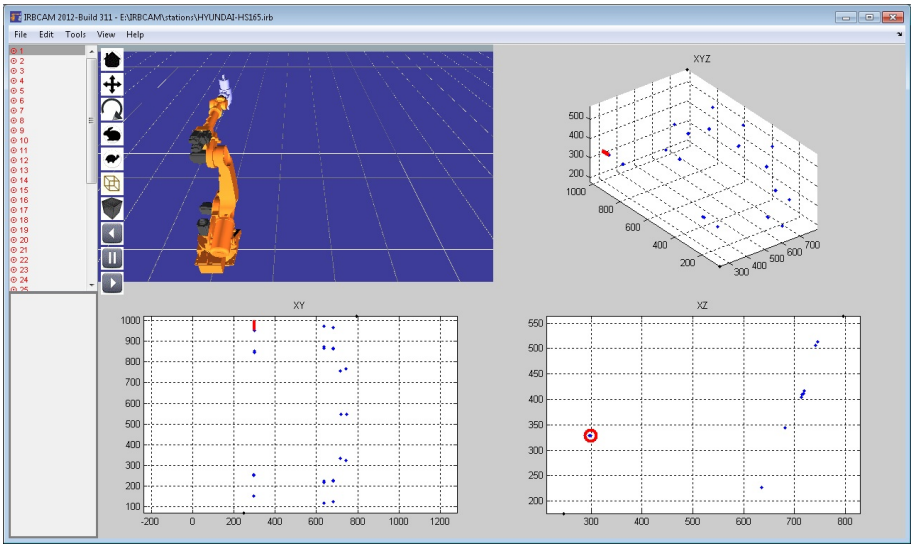


Figure 153: Hyunday HS165: Station definition.

```
CYCLE/CUSTM1,FEDTO,6,MMPM,1500,RAPTO,50,RTRCTO,100,PULBAC,98
GOTO/299.52625,851.85272,328.88016,0,1,0
GOTO/635.33087,872.65063,226.64124,0,1,0
GOTO/681.97394,868.01251,344.75,0,1,0
GOTO/715.53589,757.49652,409.42993,0.25881904,0.08418598,0.96225017
GOTO/719.79822,545.00757,417.15109,0.25881904,0,0.96592581
GOTO/715.53546,332.49484,409.42923,0.25881904,-0.08418598,0.96225017
GOTO/681.97833,222.22798,344.73831,0,-1,0
GOTO/635.33344,217.48853,226.63429,0,-1,0
GOTO/296.40384,249.4655,328.79007,0,-1,0
CYCLE/OFF
```

Between `CYCLE/` and `CYCLE/OFF` there are 9 GOTO instructions, which each defines a drilling cycle. The parameters after `CYCLE/` are described in more detail in Table 1 and apply to all the 9 GOTO instructions. This toolpath contains quite large

re-orientations of the tool and it can be difficult to find settings which will configure the path. The Optimizer (CTRL+I) can be used to find parameters which work. Fig. 154 shows the Optimizer window after a search for acceptable user frame

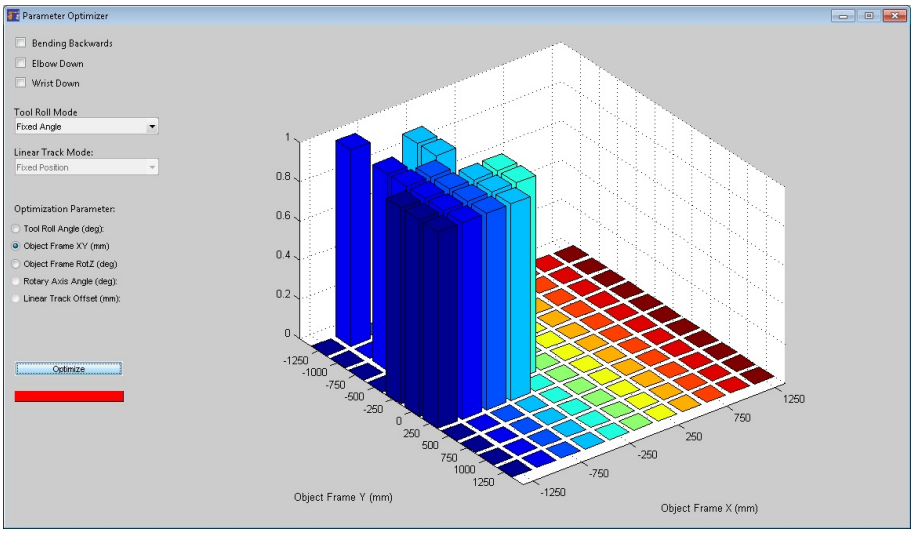


Figure 154: Hyunday HS165: Optimizer.

location. A value of 0 zero means not configurable, while a value of 1 means configurable. It can be seen from Fig. 154 that by shifting the user frame by -500mm in the X-direction and 0mm in the Y-direction, the toolpath configures. Hence, the original userframe of X=1500, Y=0, Z=750 is modified to X=1000, Y=0, Z=750. It

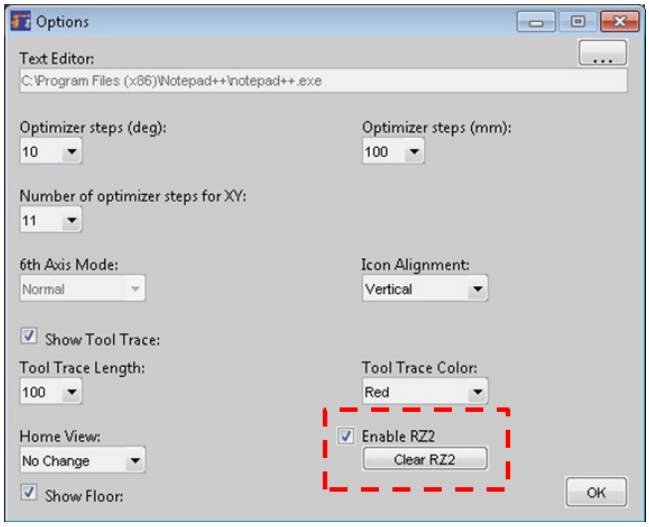


Figure 155: Hyunday HS165: General Options.

is now possible to configure the toolpath the new userframe and a tool roll angle

of for example -45 degrees. However, because of the large re-orientations in the toolpath, the tool roll angle makes large jumps and towards the end of the path the tool collides with the robot's upper arm. To avoid this behaviour the tool roll angle for some of the targets can be specified and constrained. This feature is activated using "Tools - Options" (or press CTRL+J) and selecting "Enable RZ2", see Fig. 155. Next, right-click targets 15, 30 and 31 and constrain RZ2 (the tool roll angle) to 0,90 and 90 degrees as shown in Fig. 156. After that, configure the toolpath using "Tools - Configure Path" (or press CTRL+K) using a starting tool roll angle of -45 degrees as shown in Fig. 157.

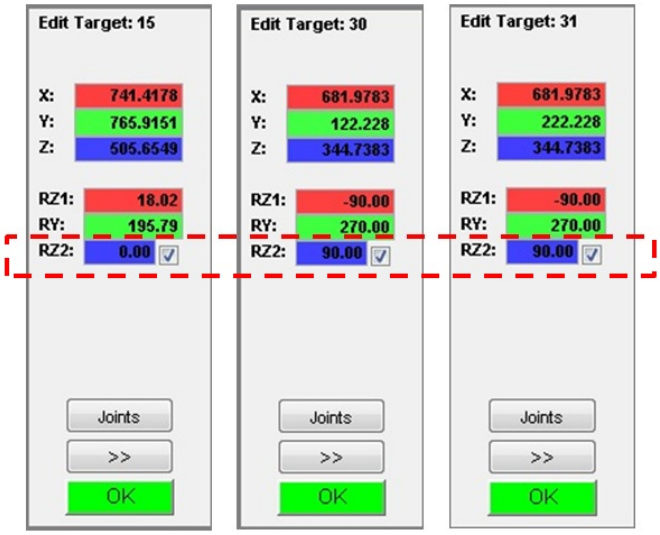


Figure 156: Hyunday HS165: Targets 15, 30 and 31.

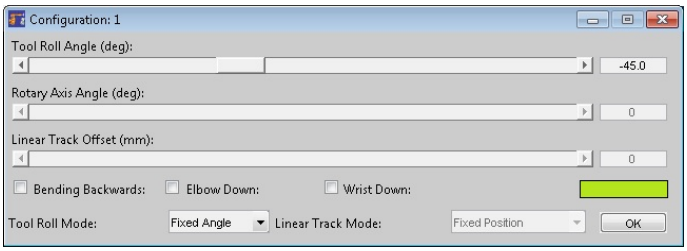


Figure 157: Hyunday HS165: Path configuration settings.

Finally, verify that all 43 targets behave as expected by scrolling through the target list in the upper-left part of the main window. Examples of robot configurations are shown in Figs. 158-160 for targets 1, 29 and 43.

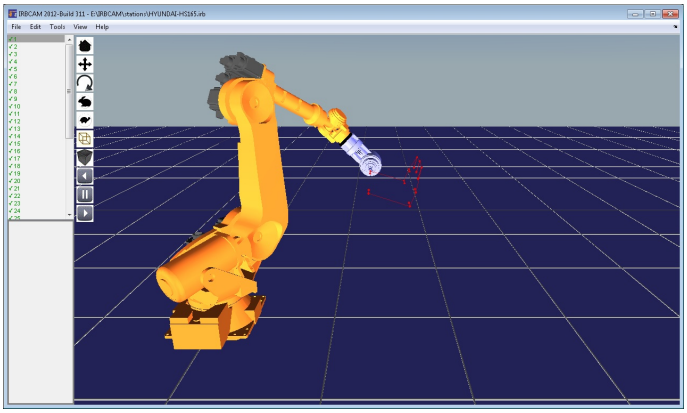


Figure 158: Hyunday HS165: Configured Target 1.

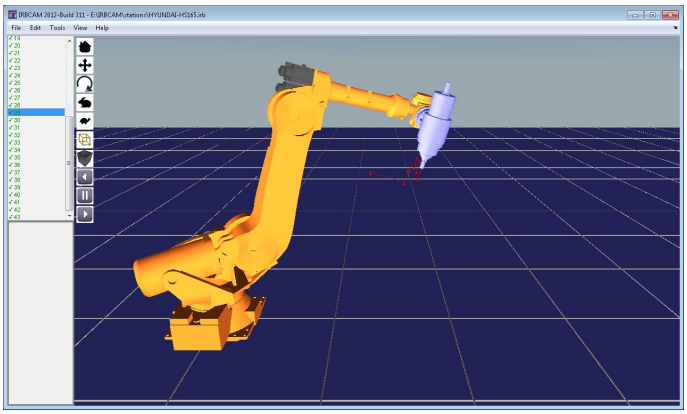


Figure 159: Hyunday HS165: Configured Target 29.

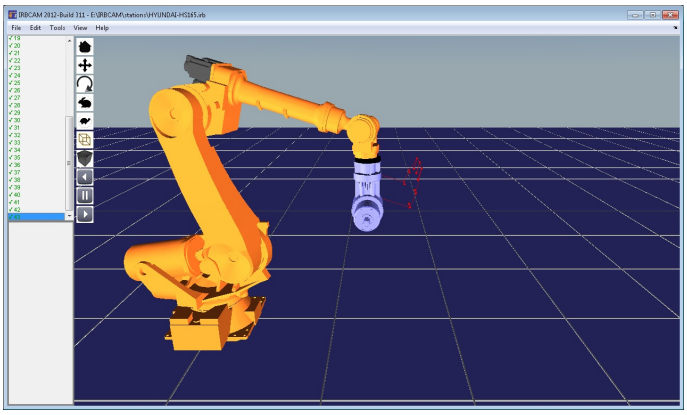


Figure 160: Hyunday HS165: Configured Target 43.

8 Collision Detection

The collision detection function in IRBCAM (CTRL+X or Tools - Collision Detection) allows the user to check for contact between the objects inside a robot station. Figure 161 shows the options available to the user. Detection for 6 tool contact sce-

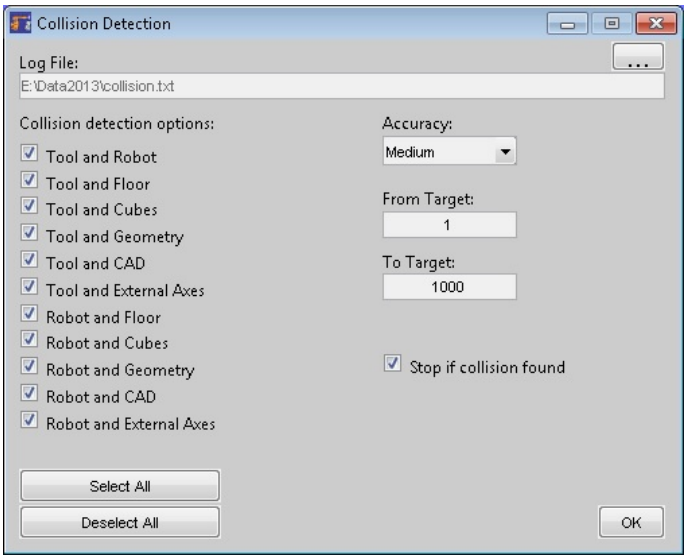


Figure 161: Collision Detection Options.

narios is available: Tool against Robot, Floor, Cubes, Geometry, CAD and External Axes. In addition, there is detection for 5 robot contact scenarios: Robot against Floor, Cubes, Geometry, CAD and External Axes. Since the collision detection can take a long time for very large toolpaths, a Log File is specified where all information about potential collisions is recorded for later inspection. In this way, the collision detection can, for example, be run overnight and inspected the following morning. For small toolpaths the collision detection takes only a few seconds or minutes and can be inspected immediately.

Three accuracy levels are available to the user: High (accuracy level 2mm / 79 thou), Medium (accuracy level 5mm / 197 thou) and Low (accuracy level 10mm / 394 thou). The Low accuracy level is significantly faster to compute than the High level. Hence, it is recommended to use the Low accuracy level for initial collision detection testing. If close tolerances are used in a station and the robot or tool is supposed to move past objects with very small clearances, then the Medium or High accuracy levels shouls be used.

The default setting is to check the entire toolpath for collisions, but the user can decide to check only selected critical portions of the toolpath. In the options screen in Figure 161 the start and end targets for the collision detection can be specified ('From Target' and 'To Target'). If the last option, 'Stop if collision found', is disabled

IRBCAM checks the entire toolpath between 'From Target' to 'To Target' for collisions. If the last option is enabled, then the collision detection will stop and the Log File will be generated when the first collision in the toolpath is detected.

8.1 Collision Detection: IRB6400 with Rotary Axis

Define a robot station as shown in Figure 162. Select the robot IRB6400-2.4-

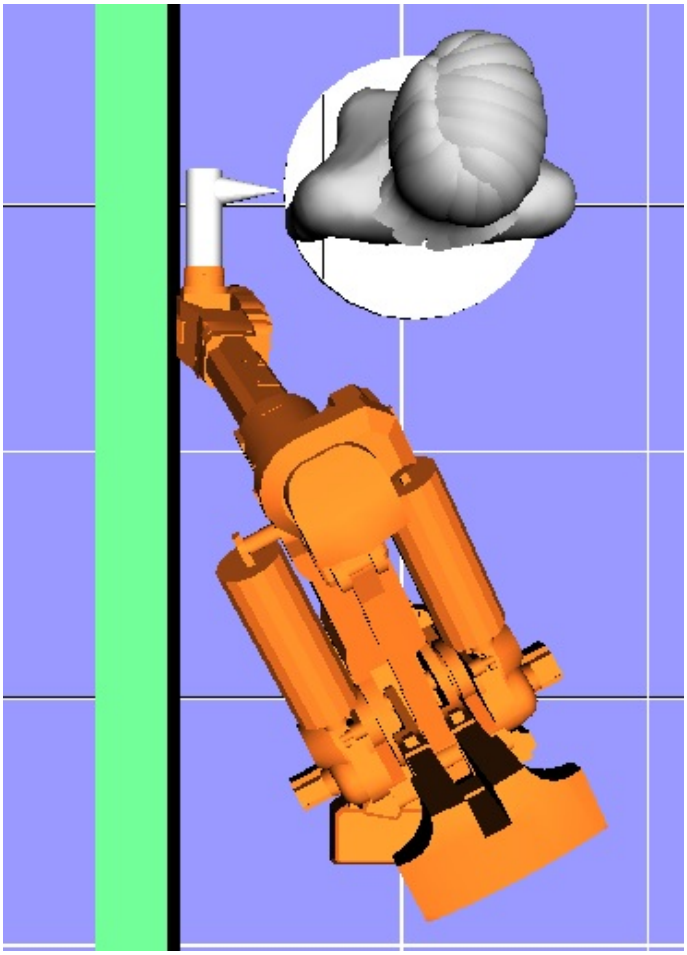


Figure 162: Robot Station: IRB6400 with Rotary Axis (View - Orthogonal Views - 3D Graphics Z+).

M97 and use the Spindle-Generic with tool data X=300mm, Y=0, Z=300mm and RY=90. Use the rotary axis Rotary-Vertical-Axis and position it at X=2000mm, Y=0 and Z=400mm. Include the user geometry Girl.irblib and define both userframe and objectframe equal to zero. Define a cube with position X=Y=Z=1000mm and dimensions X=6000mm, Y=200mm, Z=2000mm (the wall in Fig. 162).

After saving the defined station, create a single coordinate target (CTRL+E) with X=0, Y=500mm, Z=0, RZ1=90 and RY=-90. Configure the target (CTRL+K) with the settings shown in Figure 163.

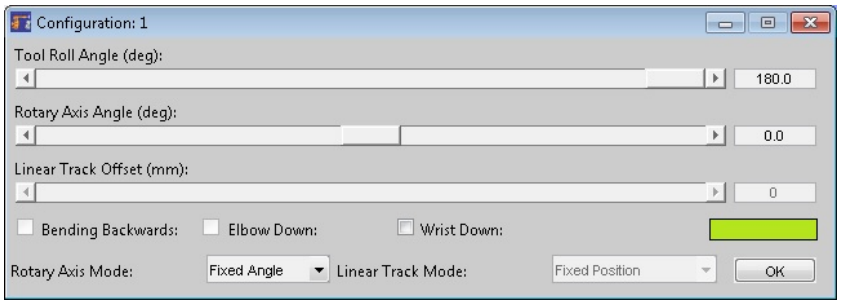


Figure 163: Robot Station: Configuration Parameters.

Run the collision detection function (Tools - Collision Detection or press CTRL+X) with the parameters shown in Figure 161 and accuracy level low. The generated text file will then contain the following:

```
Date: YYYY-MM-DD HH:MM:SS

Station Info:
-------------
Robot: IRB6400-2.4-M97C.irblib
Tool: Spindle-Generic.irblib
Geometry: Geometry-Girl.irblib
Linear Track: No
Rotary Axis: Rotary-Vertical-Axis.irblib
Actuated Table: No

Configuration:
--------------
Tool Roll Angle (deg): 180.00
Tool Roll Mode: Fixed Angle
Bending Backwards: No
Elbow Down: No
Wrist Down: No

Collision detection options:
----------------------------
Accuracy: Low
From Target: 1
To Target: 1
```

```
Tool and Robot: Yes
Tool and Floor: Yes
Tool and Cubes: Yes
Tool and Geometry: Yes
Tool and CAD: Yes
Tool and External Axes: Yes
Robot and Floor: Yes
Robot and Cubes: Yes
Robot and Geometry: Yes
Robot and CAD: Yes
Robot and External Axes: Yes

Analysis:
---------
Target 1: Arm #4 - Cube #1
```

The Log file contains the station information as well as all the collision detection options. In this particular example a collision between arm 4 of the robot and the cube is detected. By shifting the cube (Edit - Cube - Edit) from Y=1000mm to Y=1100mm, this collision can be removed. Another possibility to avoid the collision is to reconfigure the path by changing the Rotary Axis Angle in Figure 163 from 0 to 180 degrees and the Tool Roll Angle from 180 to 0 degrees. Collisions between the tool and the rotary axis can be avoided by right-clicking on Target 1 and increasing the Z-position of the target, for example to Z=50mm.

9 Tool Orientation Methods

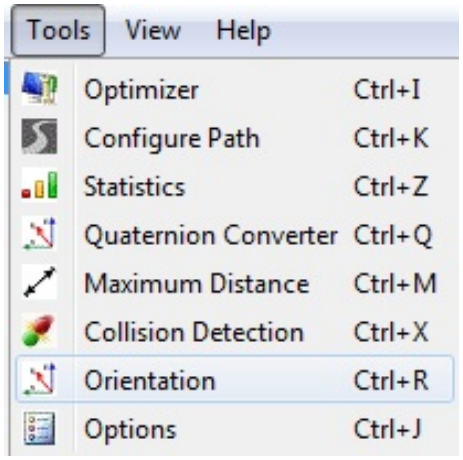


Figure 164: Tool orientation menu.

The tool orientation methods are found in the menu 'Tools - Orientation', see Fig. 164 or by pressing CTRL+R. Currently, four methods are supported as il-

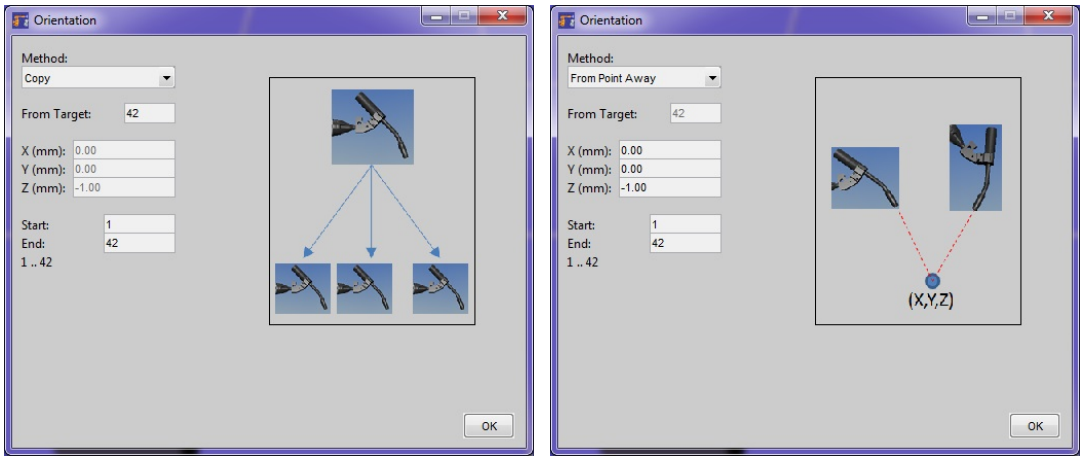


Figure 165: Tool orientation: Left: Copy. Right: From Point Away.

lustrated in Figs. 165-166. The first method, Copy, is useful when the tool orientation is defined correctly for one target and the user wants to apply this orientation to several other targets. The orientation in 'From Target' is copied to the targets 'Start' to 'End'. The next method is 'From Point Away'. The tool is oriented away from the point specified by the numeric fields for X,Y,Z. The next method, 'Through Point' works in a similar way. The tool is oriented through the point specified by the numeric fields for X,Y,Z.

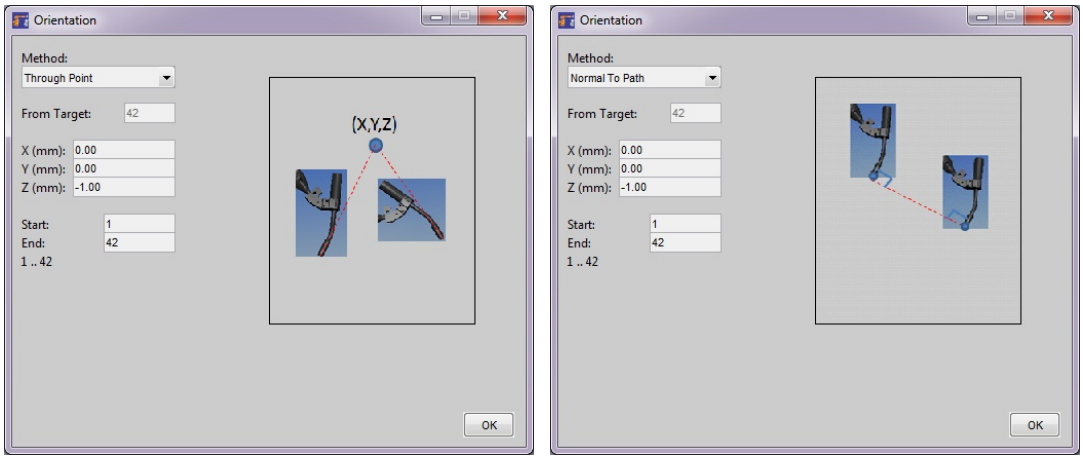


Figure 166: Tool orientation: Left: Through Point. Right: Normal to Path.

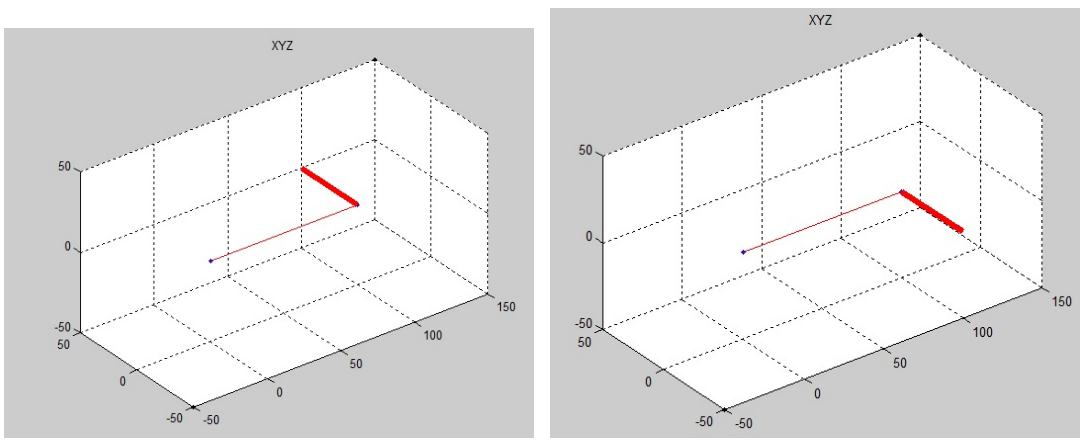


Figure 167: Tool orientation: Left: Through Point. Right: Normal to Path.

The next method, 'Normal to Path', is useful to align the tool perpendicular to the tool path. Since there are infinitely many vectors which are normal to the path, the user must tell IRBCAM which one is wanted by specifying a vector in the fields X,Y,Z. Fig. 167 shows the tool vector (red color) with two different values of X,Y,Z (left picture X=0, Y=0, Z=-1, right picture X=0, Y=0, Z=1).

10 Robot Code Output Options

This section describes the output options for ABB RAPID Code. The options are similar for Kuka, Comau, Motoman and Fanuc robots and the template files for these robot brands can be found in C:\Program Files\IRBCAM\krl, C:\Program Files\IRBCAM\pdl2, C:\Program Files\IRBCAM\jbi and C:\Program Files\IRBCAM\karel.

Figure 42 shows the RAPID export options which can be accessed from the menu 'File - ABB - RAPID Options'. In the following subsections, these options will be desribed. Note, however, that further configuration of the RAPID code output is possible by editing the template files located at: C:\Program Files\IRBCAM\rapid. In these files, any text starting with the % symbol will be replaced by IRBCAM. In addition, the user can add his own commands to these template files. For example, in 'Footer.txt' any commands can be entered which are to be executed when the RAPID code is finished, for example moving the robot back to a home position. The file 'template.txt' is used for the MOD+ROB export, while all the other template files are used for the RAPID file export.

10.1 Gun On/Gun Off

At the top of Figure 42 there are five check-boxes. The first check-box is named 'Gun On/Gun Off'. By selecting this option, the RAPID code will contain function calls for turning the tool on and off as the robot moves along the toolpath. This option is useful for waterjet, laser, plasma cutting, etc. Before a fast move defined in the APT file, the GunOff function will be called. Before a normal cutting speed, the GunOn function will be called.

The actual function calls can be edited by the user. The template files which can be edited are located at:
C:\Program Files\IRBCAM\rapid\BeforeCut.txt
C:\Program Files\IRBCAM\rapid\AfterCut.txt

For example, the user may want to add a delay of 2.0 seconds (WaitTime 2.0) after 'GunOn;' in the text-file 'BeforeCut.txt'.

10.2 Tool Number

The next check-box enables automatic tool change. If this option is selected, the RAPID code will contain a function call if there is a tool change in the APT-file.

The actual function call for the tool change can be edited by the user. The template file is located at:
C:\Program Files\IRBCAM\rapid\LoadTool.txt

The default contents of this template file is the single line '`LoadTool %ToolNumber;`'. IRBCAM will automatically replace the symbol `%ToolNumber;` with the actual tool number defined in the APT file. The user must write his own '`LoadTool`' function which moves the robot to the correct position for the tool change, opens and closes the tool cabinet, drops the current tool, picks up the new tool and performs the tool calibration (if relevant). The '`LoadTool`' function will typically interface to digital inputs and outputs of the toolchange equipment.

10.3 Spindle Speed

The next check-box enables automatic control of the spindle RPM. If this option is selected, the RAPID code will contain a function call whenever there is a change of spindle speed in the APT file.

The actual function call for the tool change can be edited by the user. The template file is located at:
C:\Program Files\IRBCAM\rapid\SpindleSpeed.txt

The default contents of this template file is the single line '`SetSpindle %SpindleSpeed;`'. IRBCAM will automatically replace the symbol `%SpindleSpeed;` with the actual speed defined in the APT file. The user must write his own '`SetSpindle`' function which interfaces with the frequency converter of the spindle. The '`SetSpindle`' function will typically interface to digital and/or analog inputs and outputs of the frequency converter.

10.4 S4 Controller

By checking this box, IRBCAM will output RAPID code for the older S4 controller. By unchecking this box, the RAPID code will be exported for the S4C+ and IRC5 controllers.

10.5 Speeds from APT File

By checking this box, IRBCAM will output robot speeds as specified in the APT file. By unchecking this box, the fast (no-contact) speed and the cutting speed will be defined by the two drop-down items further down in Figure 42.

10.6 Tool Data

The next five items are all related to the tool definition. First, the tool data name is defined. It can be an advantage to choose a short name for the tool, to make the final RAPID file as small as possible. The tool data definition is required in every single MoveL/MoveC instruction, hence this name will be repeated many times in the RAPID code. The next four items are tool weight and centre of gravity. It is important to define these parameters accurately to make sure that the robot moves along the path as accurately as possible. These parameters are particularly important if the toolpath contains high accelerations.

10.7 Module and Procedure Names

The next three items define the module, procedure and work object name. In the robot controller, the module and procedure names must be unique and there can only be one main procedure. Hence, it is useful to be able to change the names. If the main procedure in the robot controller is already used for other tasks, then the RAPID code generated by IRBCAM cannot use this name. As for the name of the tool, it can be an advantage to choose a short name for the work object, to make the final RAPID file as small as possible. The work object definition is required in every single MoveL/MoveC instruction, hence this name will be repeated many times in the RAPID code.

10.8 External Axes

After the speed definitions, the next three items are related to external axis equipment. First, the name of the rotary unit must be specified. This name should match the system definition for the external axis (the name can be found in the backup file MOC.cfg under 'MECHANICAL_UNIT'). The next two items define the logical axis numbers for any rotary or linear track. The logical axis numbers for external equipment range from 7 to 12 (the first 6 axes are used for the robot).

10.9 Controller Directory

Finally, the controller directory can be specified for S4C+ and IRC5 controllers. This directory is only required if the toolpath is exported as a MOD+ROB combination (CTRL+M). In this case, the MOD file needs to know the path to where the ROB file (containing the coordinates) is located on the robot controller. When the toolpath is exported as MOD+ROB, then the system module CAMSYS.SYS must be pre-loaded into the controller. The CAMSYS.SYS file is located at: C:\Program

Files\IRBCAM\rapid\CAMSYS.SYS. If the toolpath is exported as a single RAPID file, then the controller directory name is not used. The robot controller can typically only hold about 24,000 targets in memory. Hence, if the toolpath contains more than about 24,000 targets, then the MOD+ROB export option should be used. Alternatively, a single robot code output file can be split into smaller parts using 'File - Split APT/RAPID/KRL/JBI'.

11 Stepper Mode

IRBCAM has a stepper mode functionality which can be used to control a turntable which is not fully integrated with the robot controller. This allows for adding a low-cost solution for the turntable, if an indexed table is sufficient as opposed to continuous rotation angles. Fig. 168 shows the menu item where the stepper mode

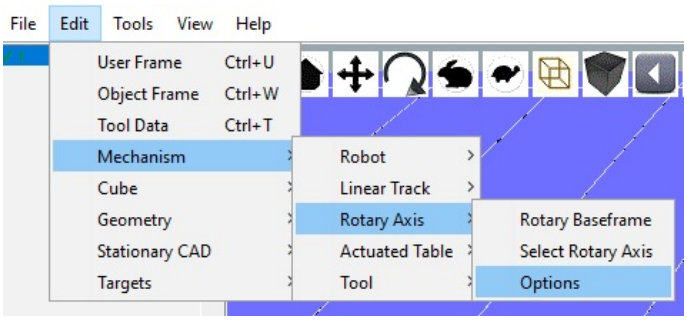


Figure 168: Stepper Mode Menu Item.

parameters can be defined for turntables (also called rotary axes in this manual). Note that this mode will only appear in the menu if a stepper mode license of IRBCAM has been installed. After the Options have been selected in Fig. 168,

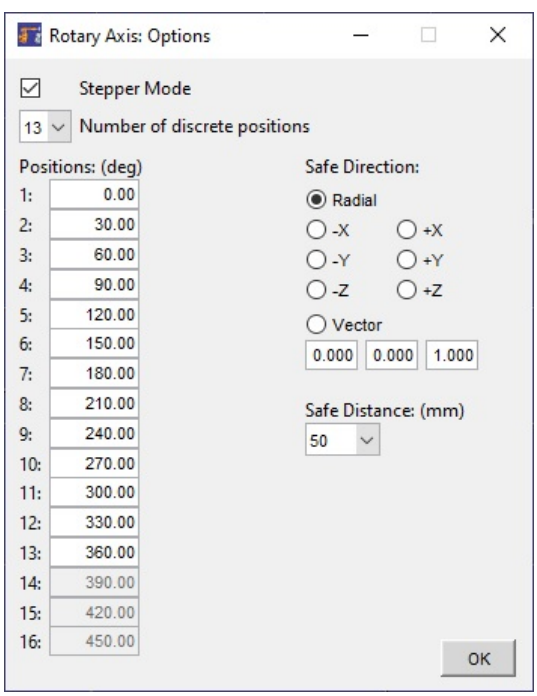


Figure 169: Stepper Mode Options.

the dialog window shown Fig. 169 will appear. This window shows that IRBCAM

accepts up to 16 discrete angles for the turntable. In this example, only 13 angles will be used, from 0° to 360° in steps of 30°. The safe direction is selected as 'Radial' and the safe distance is defined to be $50mm$. The safety parameters mean that when the turntable needs to move, the tool is retracted from the part in the radial direction out from the centre of the turntable with a distance of $50mm$ from the current toolpath. If the user wishes to do so, the safety direction can alternatively be chosen in the X,Y or Z directions, or in any user-defined direction. For example, the end-user may wish to move the tool upwards in the +Z direction when the turntable moves. +Z could be a better choice for parts which are not very high (low Z-values) but need a turntable with a large radius. If the turntable's radius is very large, a radial safety move could then cause the safety points to be out of reach for the robot.

When configuring the toolpath, IRBCAM will try to configure with the initial angle of the rotary axis. If a target point is not reachable for the robot, the turntable will be moved to another discrete angle in the user-defined list and IRBCAM will try again. This process is done automatically by IRBCAM, and the end-user does not have to manually define the discrete angles of the turntable in the toolpath, except for the initial starting angle.

```
1   %%%
2     VERSION:1
3     LANGUAGE:ENGLISH
4   %%%
5
6   MODULE IRBCAM1
7     !CONST zonedata z0:=[FALSE,0.2,0.2,0.2,0.2,0.2,0.2];
8     PERS tooldata t1:=[TRUE,[[235,0,235],[0.7071068,0,0.7071068,0]],[5,[0,0,100],[1,0,0,0],0.01,0.01,0.01]];
9     PERS wobjdata w1:=[FALSE,TRUE,"STN1",[[1500,500,1000],[0.7071068,0.7071068,0,0]],[[0.0,0.0,0.0],[1.0,0.0,0.0,0.0]]];
10
11    PROC F_()
12      MoveAbsJ [[4.593,2.656,45.705,-2.872,-50.102,105.927],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,fine,t1;
13      ROT_0;
14      MoveL [[177.05,203.115,120],[0.0606439,0.0075104,-0.9905638,-0.1226754],[0,-1,1,1],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,z0,t1\WObj:=w1;
15      MoveL [[177.05,203.115,45.128],[0.0606439,0.0075104,-0.9905638,-0.1226754],[0,0,1,1],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,z0,t1\WObj:=w1;
16      ROT_1;
17      MoveL [[177.05,203.115,34.628],[0.0606439,0.0075104,-0.9905638,-0.1226754],[0,0,1,1],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,z0,t1\WObj:=w1;
18      MoveL [[175.769,205.658,24.522],[0.0606439,0.0075104,-0.9905638,-0.1226754],[0,0,1,1],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,z0,t1\WObj:=w1;
19      MoveL [[174.771,206.186,24.5],[0.0632715,0.0079258,-0.9902259,-0.1240425],[0,0,1,1],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,z0,t1\WObj:=w1;
20      MoveL [[173.71,206.643,24.437],[0.0660479,0.0083527,-0.9898971,-0.1251863],[0,0,1,1],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,z0,t1\WObj:=w1;
21      ROT_2;
22      MoveL [[172.441,207.018,24.293],[0.0693541,0.0088343,-0.9895572,-0.12605],[0,0,1,1],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,z0,t1\WObj:=w1;
23      MoveL [[179.396,203.045,24.336],[0.0510064,0.0059471,-0.9919607,-0.1156585],[0,0,1,1],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,z0,t1\WObj:=w1;
24      MoveL [[181.04,201.564,24.14],[0.0465907,0.0052379,-0.992647,-0.1115971],[0,0,1,1],[9E+9,9E+9,9E+9,9E+9,0.00,9E+9]],v100,z0,t1\WObj:=w1;
25    ENDPROC
26  ENDMODULE
```

Figure 170: Example of Exported Code.

Fig. 170 shows an example of generated robot code for an ABB controller with a stepper mode turntable. As seen in the generated code (lines 13, 16 and 21), three discrete angles are used by IRBCAM. These are called by the functions Rot_0, Rot_1 and Rot_2 representing the angles 0°, 30° and 60° as defined in Fig. 169. The functions Rot_0, Rot_1, $\cdots$, Rot_N must be defined by the end-user, since these are normally very user-specific. For example, digital IO can be used to change the stepper angles, or a bus like CANbus.

12 IRBCAM as a Post-Processor

For most users the robot is fixed, any external axis is fixed, the machining table is fixed and user/object frames are in fixed positions. Hence, it is usually the first path configuration in IRBCAM which may be difficult to get working. Subsequent translations and path configurations will probably be very similar to the first.

When this is the case, the graphical interface does not have to be opened up every time to generate robot code. IRBCAM can be run both as a Windows GUI application or as a command-line application.

The different command-line parameters are as follows:

- `irbcam.exe -h | more` - Displays the following command-line options

```
Usage: IRBCAM <mode> <mindist> <irb-file> <apt-file> <out-file> <optargs>
     : mode=0 to open the APT file in the GUI (no <out-file>)
     : mode=1 to generate RAPID (ABB)
     : mode=2 to generate MOD+ROB (ABB)
     : mode=3 to generate KRL (KUKA)
     : mode=4 to generate KRL+ROB (KUKA)
     : mode=5 to generate PDL2 (COMAU)
     : mode=6 to generate JBI (MOTOMAN)
     : mode=7 to generate KAREL (FANUC)
     : mode=8 to generate LS (FANUC)
     : mode=9 to generate JOB (HYUNDAI)
     : mode=10 to generate V+ (ADEPT)
     : mode=11 to generate NACHI code
     : mode=12 to generate RAPL-2 (CRS)
     : mode=13 to generate VAL3 (STAUBLI)
     : mode=14 to generate ASC (OTC-DAIHEN)
     : mode=15 to generate PAC (DENSO)
     : mode=16 to generate AS (KAWASAKI)
     : mode=17 to generate PRG (MITSUBISHI)
     : mode=18 to generate SCRIPT (UNIVERSAL)
     : mode=19 to generate SCOL (TOSHIBA)
     : mindist is the minimum distance between robot coordinates
     : irb-file is the station file from the GUI
     : apt-file is the input APT file from the CAM software
     : out-file contains the output with configured RAPID code
     : <optargs> see manual chapter 10 for optional arguments
```

- `irbcam.exe -h > file.txt` - Saves the command-line options to a file

- `irbcam.exe` - No parameters, IRBCAM starts as a Windows GUI application

The following shows an exampe command-line call for IRBCAM:

```
cd "c:\Program Files\IRBCAM"
IRBCAM 1 1.0 c:\irb4400.irb c:\Demopart.apt c:\test1.prg
```

This example will generate ABB RAPID code with a minimum distance between the coordinates of 1.0mm. The configuration file irb4400.irb and the APT file Demopart.apt are used to output the RAPID code in the file test1.prg.

If the configuration is successful, IRBCAM will output the following message:

```
1
IRBCAM path configuration succeeded.
```

The first '1' means successful and can be used by other software to check the status of the IRBCAM output. Alternatively, if something goes wrong during the conversion, the output could for example look like:

```
0
IRBCAM Command-line operation failed.
Check if the file names are correct
and that the IRB file is set up correctly.
Error message: Configuration data missing in IRB file.
```

If you get this error message, it means that the IRB file does not contain the configuration data for the Tool Roll Angle, the Rotary Axis Angle and the Linear Track Offset. These parameters must be defined in the path configuration window in the GUI (CTRL+K), see Figure 171. In this case, the Tool Roll Angle has been set

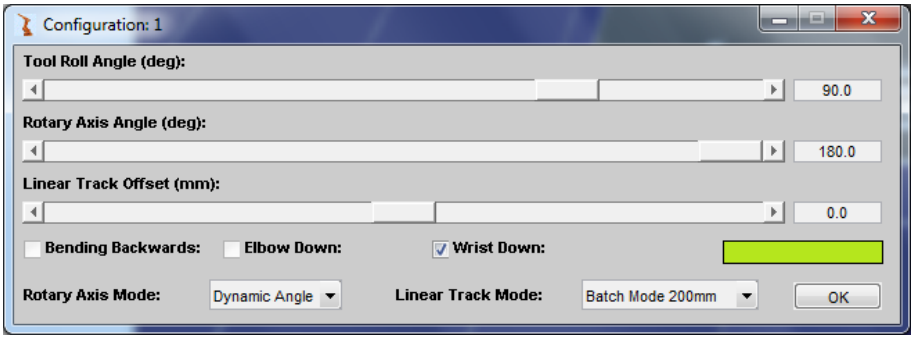


Figure 171: Path configuration window.

to 90°, the Rotary Axis Angle to 180° and the Linear Track Offset to $0mm$. Additional parameters which are defined are: 'Rotary Axis Dynamic Angle', 'Linear

Track Batch Mode 200mm' and 'Wrist Down'. When these parameters have been defined and the path has been configured correctly in the GUI, save the IRB file. This IRB file can now be used with the post-processor.

With the command-line mode IRBCAM can be integrated into CAM software as a post-processor. For example, in SurfCam, the following commands can be added to the 'PostMenu.cfg' file to add IRBCAM as a post-processor. In SurfCam 2014 the file 'PostMenu.cfg' is normally located in the directory:
C:\Users\Public\SURFCAM\SURFCAM2014\Config.

```
PostItem ABB RAPID CODE
 Status ABB RAPID CODE
 Command "C:\Program Files\SURFCAM\SURFCAM2014\INC2APT" -I -FAR "%p%n" -O "%p%N.apt"
 Task D:\IRBCAM.bat "%p%N.apt" "%N"
 Task    "C:\Program Files\SURFCAM\SURFCAM2014\Apps\editNC\editNC" "D:\%N.prg"
```

The file 'IRBCAM.bat' in this example is shown below:

```
@echo off
cd "C:\Program Files\IRBCAM"
C:
start /wait IRBCAM.exe 1 1.0 D:\cfg.irb %1 "D:\%2.prg"
```

12.1 Optional Parameters

IRBCAM supports several optional command-line parameters. Below is an example using two of these:
`IRBCAM 1 1.0 station.irb aptfile.apt robfile.prg -uframex 1200 -tooly 250`

In this example the User Frame X-position is changed to 1200mm and the Tool Data Y-value is changed to 250mm. The optional parameters can only be used at the end of the command-line, and override the values stored in 'station.irb'. Table 6 shows the complete list of optional parameters. The values n in the table must be replaced by the actual values.

Note that if the optional parameters for rotation (-uframeRx, -uframeRy or -uframeRz) are used, then the optional parameters for the quaternions (-uframeq1 $\cdots$ -uframeq4) are ignored. The same applies with the rotation parameters for the object and the tool frames.

For various reasons, a user may want to turn off IRBCAM's reachability and singularity checking. This can be done with the optional command-line parameter '-configure 0'. However, it is recommend to keep configuration checking on, and this parameter is on by default. Robot programs generated with configuration off,

are not likely to run on the robot controller without turning configuration checking off. Some programs will not run at all, even with configuration checking off on the controller. For example, on ABB controllers the RAPID command for turning configuration checking off is 'ConfL\Off'. One situation where turning configuration checking off may be useful, is when a user wants to run the same toolpath on several different types of robot models, and wants to save time by defining only one common IRBCAM station for all the robots. This option is only meant for expert users, who are confident that unconfigured toolpaths will run without problems on the robot controller. When configuration checking is turned off, the initial tool roll angle defined either in the station file or by the optional parameter '-toolroll n' will be used for the entire toolpath. IRBCAM will not attempt to modify the tool roll angle when configuration checking is turned off.

An example command-line call when including collision detection is as follows:

```
IRBCAM 1 1.0 s1.irb apt1.apt out.prg -collision_file collision.txt
-collision_accuracy 2
```

The options here mean the following:

1. 1 - Means that the generated robot code is for ABB (RAPID)
2. 1.0 - Minimum distance between targets (in mm)
3. s1.irb - Name of the input station file
4. apt1.apt - Name of the input toolpath file (APT-CL format)
5. out.prg - Name of the output robot code file
6. -collision_file collision.txt - Enables collision detection and specifies the output filename
7. -collision_accuracy 2 - Specifies the collision detection accuracy level (2= Medium= 5mm / 197 thou)

If the toolpath is successfully configured and the collision detection functions can be evaluated on the entire toolpath, then the output from IRBCAM will be as follows:

```
1
IRBCAM path configuration succeeded.
1
IRBCAM collision checking succeeded.
```

In this case, two files will be generated: 1) out.prg (the ABB RAPID code) and 2) collision.txt (the summary of the collision detection analysis).

Optional Parameter	**Explanation**
-toolroll n	Tool Roll Angle (n in degrees)
-uframex n	User Frame X Position (n in mm)
-uframey n	User Frame Y Position (n in mm)
-uframez n	User Frame Z Position (n in mm)
-uframeq1 n	User Frame Quaternion Component q1 (n from -1 to +1)
-uframeq2 n	User Frame Quaternion Component q2 (n from -1 to +1)
-uframeq3 n	User Frame Quaternion Component q3 (n from -1 to +1)
-uframeq4 n	User Frame Quaternion Component q4 (n from -1 to +1)
-uframeRx n	User Frame X-Axis Rotation (n in deg)
-uframeRy n	User Frame Y-Axis Rotation (n in deg)
-uframeRz n	User Frame Z-Axis Rotation (n in deg)
-oframex n	Object Frame X Position (n in mm)
-oframey n	Object Frame Y Position (n in mm)
-oframez n	Object Frame Z Position (n in mm)
-oframeq1 n	Object Frame Quaternion Component q1 (n from -1 to +1)
-oframeq2 n	Object Frame Quaternion Component q2 (n from -1 to +1)
-oframeq3 n	Object Frame Quaternion Component q3 (n from -1 to +1)
-oframeq4 n	Object Frame Quaternion Component q4 (n from -1 to +1)
-oframeRx n	Object Frame X-Axis Rotation (n in deg)
-oframeRy n	Object Frame Y-Axis Rotation (n in deg)
-oframeRz n	Object Frame Z-Axis Rotation (n in deg)
-toolx n	Tool Data X Value (n in mm)
-tooly n	Tool Data Y Value (n in mm)
-toolz n	Tool Data Z Value (n in mm)
-toolq1 n	Tool Data Quaternion Component q1 (n from -1 to +1)
-toolq2 n	Tool Data Quaternion Component q2 (n from -1 to +1)
-toolq3 n	Tool Data Quaternion Component q3 (n from -1 to +1)
-toolq4 n	Tool Data Quaternion Component q4 (n from -1 to +1)
-toolRx n	Tool Frame X-Axis Rotation (n in deg)
-toolRy n	Tool Frame Y-Axis Rotation (n in deg)
-toolRz n	Tool Frame Z-Axis Rotation (n in deg)

Table 6: *Optional command-line parameters. The variable n must be replaced by a number. Note that the quaternions must be properly scaled. The requirement is that $q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1$.*

Optional Parameter	**Explanation**
-configure n	Configuration off/on (n: 0 or 1)
-cviewx n	Coordinate X Translation (n in mm)
-cviewy n	Coordinate Y Translation (n in mm)
-cviewz n	Coordinate Z Translation (n in mm)
-cviewrx n	Coordinate X Rotation (n in degrees)
-cviewry n	Coordinate Y Rotation (n in degrees)
-cviewrz n	Coordinate Z Rotation (n in degrees)
-maxdistance n	Max allowed distance for linear moves ($n \geq 1.0$ in mm)
-collision_file name.txt	Output file name for collision detection
-collision_accuracy n	Accuracy level (n=1=High, n=2=Medium, n=3=Low)
-startpercent n	Multi-thread parameter (n between 0 and 100)
-endpercent n	Multi-thread parameter (n between 0 and 100)
-textfile 1	Imports an ASCII text file instead of APT, see Sec. 5.3
-reorient n	Re-orients the tool (n=1,2,3,4). See Sec. 9
-statusWindow 1	Opens a graphical status window
-opengl software	Forces OpenGL software emulation
-opengl hardware	Forces OpenGL hardware
-opengl hardwarebasic	Forces OpenGL hardware basic

Table 7: *Optional command-line parameters (continued).*

12.2 Example of IRBCAM integration with SurfCam

In this section an integrated example with IRBCAM used as a post-processor in SurfCam will be demonstrated. First, the robot station needs to be defined. In

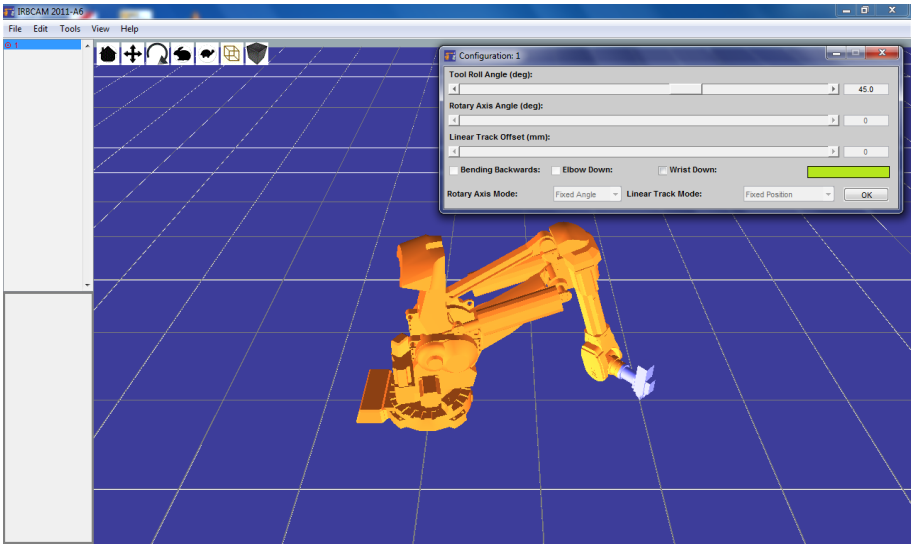


Figure 172: IRBCAM configuration screen.

this example, we will use an IRB6400R-2.5-200kg robot with the spindle TMA4 from the standard library. Select the user frame at (2000,0,0) (the default location) and no user object or cubes. Save the station configuration in the file 'irb6400.irb'. This station file will later be used in post-processor mode. In order for IRBCAM to work as a post-processor, we need to define parameters such as the initial tool-roll angle, wrist up/down, etc. To be able to define these parameters, we need at least one coordinate in the station. Hence, select 'Edit - Targets - Add Target' and accept the default values (0,0,0,0^o,180°,0°) and click 'OK'. The only reason why we need this single target, is to enable the path configurator. Select the path configurator (CTRL+K) as illustrated in Figure 172 and define the initial Tool Roll Angle equal to 45° and finally save the station data to 'irb6400.irb'.

In SurfCam create a polyline by selecting 'Create - Line - String - Polyline'. Define the polyline by the following five coordinates: (0,0,0) - (100,0,0) - (100,100,0) - (0,100,0) and (0,0,0). Click 'Done' to complete the polyline. Next, select 'NC - 3-Axis - Contour 3D' to generate a 3-axis toolpath along the polyline.

If the SURFCAM.PST file has been updated according to section 12, then the ABB RAPID Code option should appear in the NC Operations Manager, as illustrated in Figure 173. Click on 'Post' to generate the ABB Rapid Code. If the post-processor is successful, the generated code will be opened by editNC, as illustrated in Figure 174.

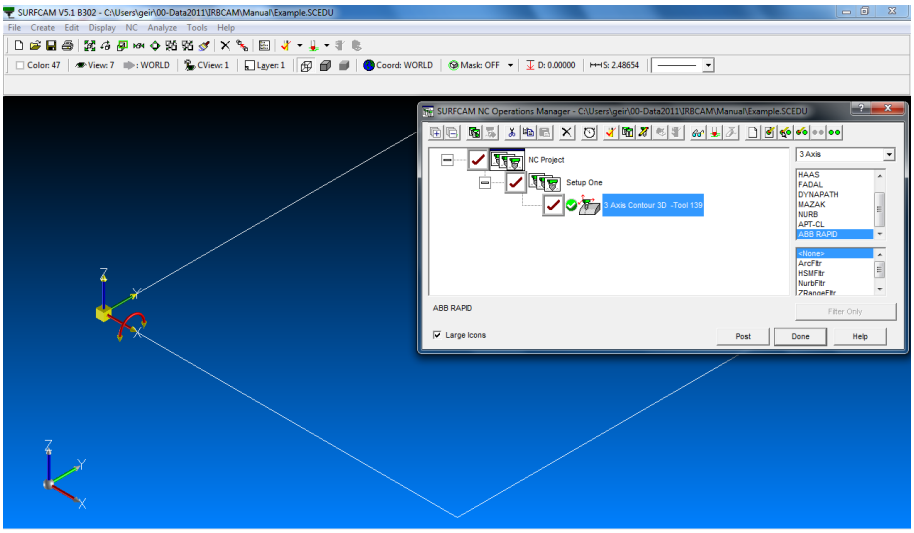


Figure 173: SurfCam NC Operations Manager.

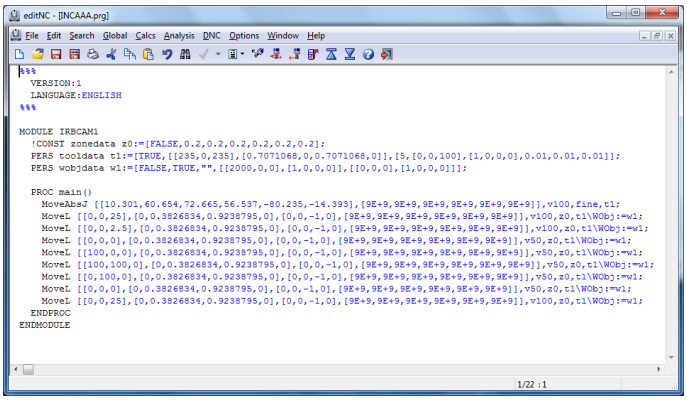


Figure 174: editNC showing generated RAPID code.

It should be noted that the post-processor will only be able to generate code if the initial settings in Figure 172 are matching (relatively closely) the current toolpath in SurfCam. In the example in this section, the position (0,0,0) of the single target which was used by the configurator screen in IRBCAM, coincides with the first coordinate in the polyline in SurfCam. In this way, we can be quite certain that the IRBCAM configuration will be OK. When generating new toolpaths with very different starting positions and/or orientations (in 5-axis mode), the initial configuration from IRBCAM may not be very good and the post-processor in SurfCam may have problems generating code. In such cases, it is recommended to open the graphical user interface of IRBCAM and configure the path there. The updated configuration settings can then be saved as an IRB file and used with the post-processor in SurfCam.

As an alternative, it may be useful to have several IRB configuration files with differ-

ent settings and to make one entry in PostMenu.cfg for each of these configuration files.

As a third alternative, the generated APT file in SurfCam can automatically be opened in the IRBCAM GUI for further editing and path configuration. In this case, the following post-processor code can be used.

```
PostItem IRBCAM GUI
  Status  IRBCAM GUI
  Command "C:\Program Files\SURFCAM\SURFCAM2014\INC2APT" -I "%p%n" -O "%p%N.apt"
  Chdir "C:\Program Files\IRBCAM"
  Task "C:\Program Files\IRBCAM\IRBCAM" 0 1.0 c:\cfg.irb %p%N.apt
```

12.3 Example of IRBCAM integration with VisualMill

In this section an integrated example with IRBCAM used as a post-processor in VisualMill 2012 will be demonstrated. It is assumed that a robot station is already defined in IRBCAM and saved with the filename S1.irb. Two Windows Batch files are needed. Save the following text in a file named: IRBCAM1.bat

```
@echo off
cd "C:\Program Files\IRBCAM"
IRBCAM.EXE 0 1.0 "C:\Users\Public\S1.irb" %1
```

Save the following text in a file named: IRBCAM2.bat

```
@echo off
cd "C:\Program Files\IRBCAM"
IRBCAM.EXE 1 1.0 "C:\Users\Public\S1.irb" %1 "C:\Users\Public\abb1.prg"
"C:\Program Files (x86)\Notepad++\notepad++" "C:\Users\Public\abb1.prg"
```

IRBCAM1.bat starts up IRBCAM with the pre-defined station S1.irb. The parameter %1 will be replaced by the output file (on APT CLS format) from VisualMill. The first parameter after IRBCAM.EXE equals 0, which means that IRBCAM will open up in GUI-mode. The second parameters equals 1.0 (unit mm) which defines the minimum allowed distance between the points in the APT CLS file.

IRBCAM2.bat uses IRBCAM as a traditional post-processor without the graphical interface. The first parameter after IRBCAM.EXE equals 1, which means that robot code on the ABB RAPID language syntax will be generated, see the command-line options in the beginning of this chapter. The second parameter (1.0) specifies the minimum distance between the points as in IRBCAM1.bat. The robot code is in this example saved in the file "C:\Users\Public\abb1.prg". The last line in IRBCAM2.bat causes Notepad++ (if installed) to open up the generated ABB RAPID code.

Figure 175 shows the required post-processor settings in VisualMill 2012. Set the "Current Post-Processor" to APT CLS IJK and "Program to send the posted file to:" to IRBCAM1.bat (including the entire file path, in this case C:\Users\Public \IRBCAM1.bat.

Next, we will create a small example in VisualMill and post the result to IRBCAM. In Figure 176 a Box with dimensions 300mm x 300mm x 4mm is created and then a toolpath is created using 3-Axis Spiral Machining. Select the flat area to machine and use a ballnose tool with diameter 30mm as shown in Figures 177 and 178. Finally, right-click on "Machining Job - Post APT CLS IJK" and select "Post" as shown in Figure 179. Select the filename of the generated APT file from VisualMill, then wait until the IRBCAM GUI opens as shown in Figure 180. The toolpath shown

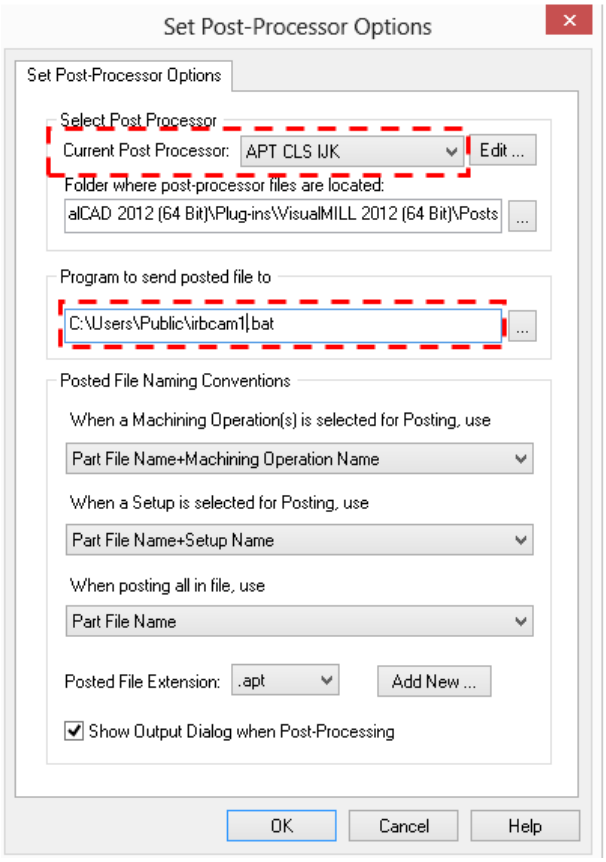


Figure 175: VisualMill 2012: Post-Processor Options and IRBCAM1.bat.

in IRBCAM (Figure 180) should match the toolpath in VisualMill (Figure 179). At this stage, the toolpath is not configured to run on the robot. The user must now continue with configuring the toolpath (CTRL+K) in a similar way as in the examples in Section 7.

Figure 181 shows the generated robot code opened in Notepad++ when IRBCAM2.bat is used instead of IRBCAM1.bat in Figure 175. In order to be able to generate robot code directly, it is important that the parameter "Tool Roll Angle" is defined in the station file. Before saving the station file S1.irb in IRBCAM, add a new robot target (CTRL+E) and configure (CTRL+K) this single target with the desired tool roll angle. Then save S1.irb (CTRL+S) and it is ready to be used together with the batch file IRBCAM2.bat.

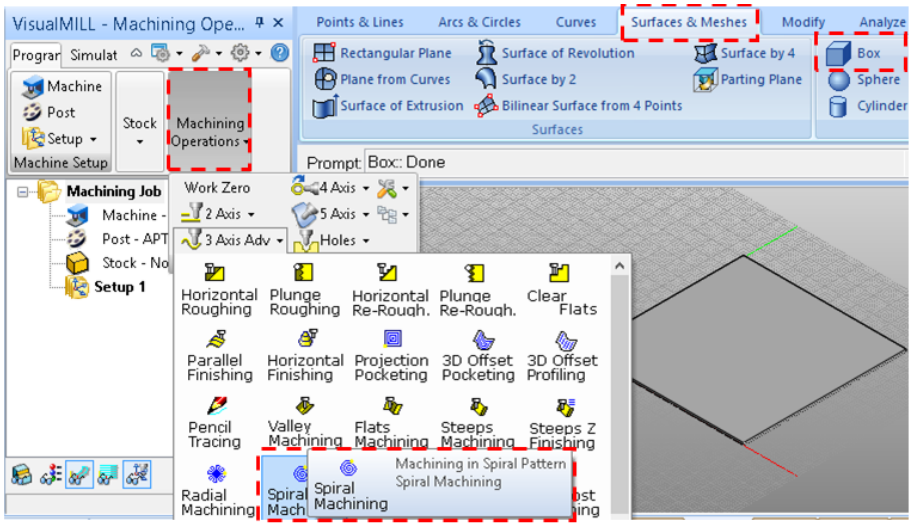


Figure 176: VisualMill 2012: Create Box and 3-Axis Spiral Machining.

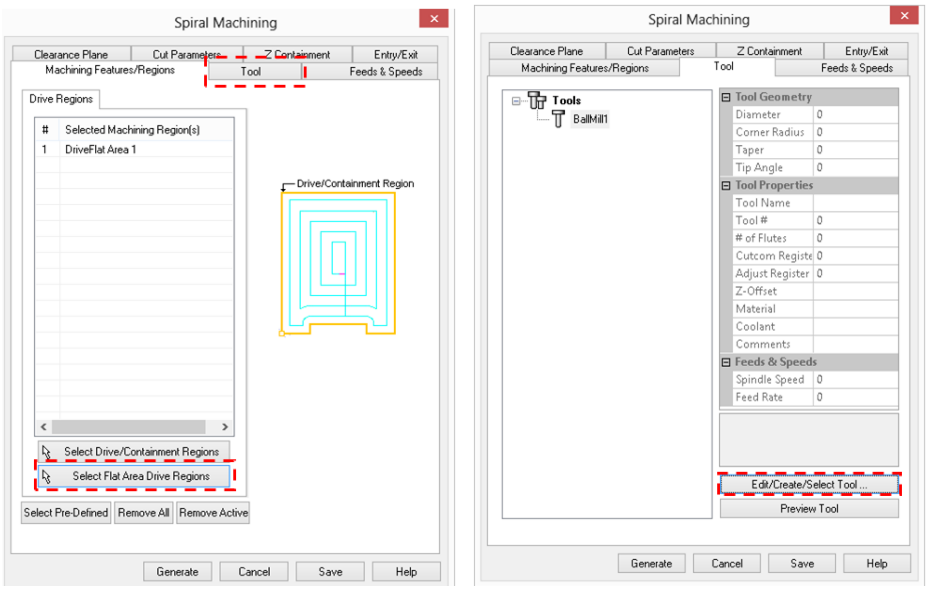


Figure 177: VisualMill 2012: Spiral Machining Options.

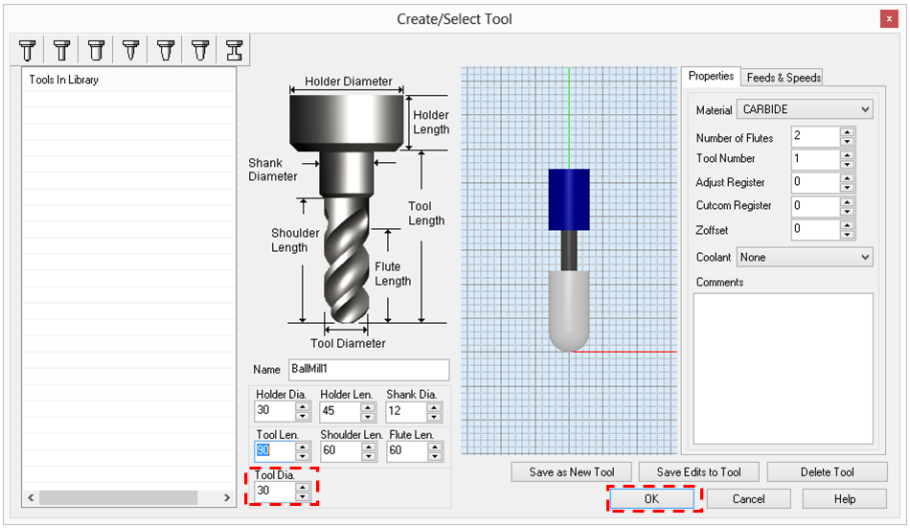


Figure 178: VisualMill 2012: Ballnose Tool Definition.

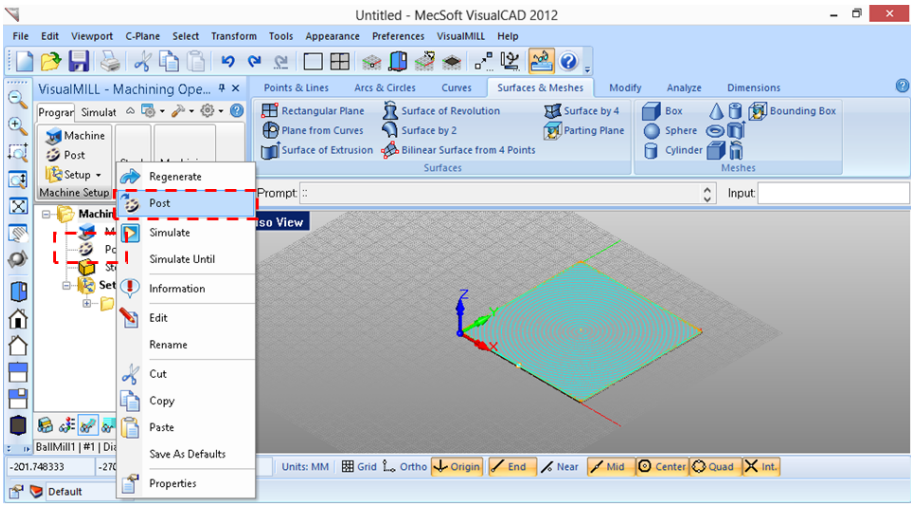


Figure 179: VisualMill 2012: Generate Post.

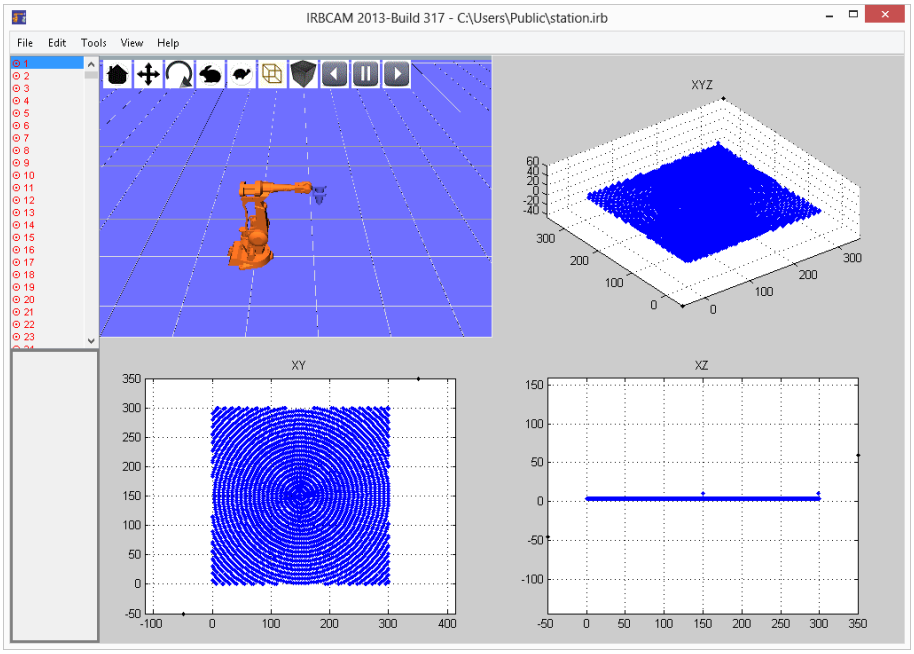


Figure 180: IRBCAM: GUI opened directly from VisualMill 2012.

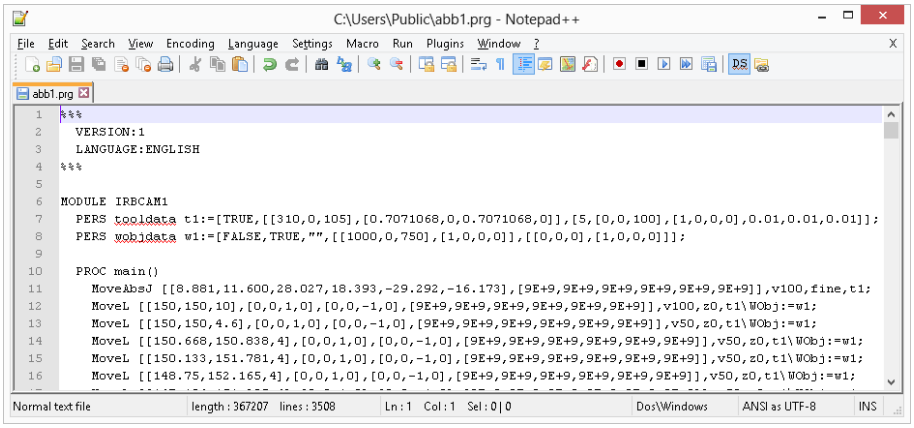


Figure 181: Generated robot code (ABB RAPID) using IRBCAM2.bat and opened in Notepad++.

12.4 Autodesk Fusion 360 and IRBCAM

Autodesk Fusion 360 has become a popular cloud-based CAD/CAM software package. Fusion 360 can generate toolpath files on APT-CLS format which are compatible with IRBCAM. This section contains an example where a 2D DXF file is uploaded to Fusion 360, a 2D contour toolpath is generated and an APT-CLS file is exported. This APT-CLS file is finally imported into IRBCAM, configured and a robot program is generated.

Fig. 182 shows a screenshot of the Upload screen in Fusion 360. After the file has been uploaded and loaded into Fusion 360, select "Design - Manufacture" as shown in Fig. 183. As shown in Fig. 184 select "Design - Manufacture - 2D Contour". The next step is to define the milling tool as shown in Fig. 185. After the milling tool has been defined, selected the desired contour as shown in Fig. 186. Finally, export the 2D contour toolpath to the ISO APT-CLS format as shown in Figs. 187 and 188.

To import the APT-CLS file into IRBCAM, see Fig. 189. In this example the toolpath is scaled down to 10% in the X,Y and Z directions, while the minimum distance between the points is set to $5mm$. The minimum distance filter will reduce the accuracy of the toolpath slightly, but it will make sure that the robot does not get too many points close together to execute. Alternatively, the "Smoothing Parameter" in Fusion 360 can be used instead to filter the points. Fig. 190 shows the toolpath configured to run on an ABB IRB6400-2.4m robot. At this stage the robot program can be exported using the menu item "File - ABB - Save RAPID".

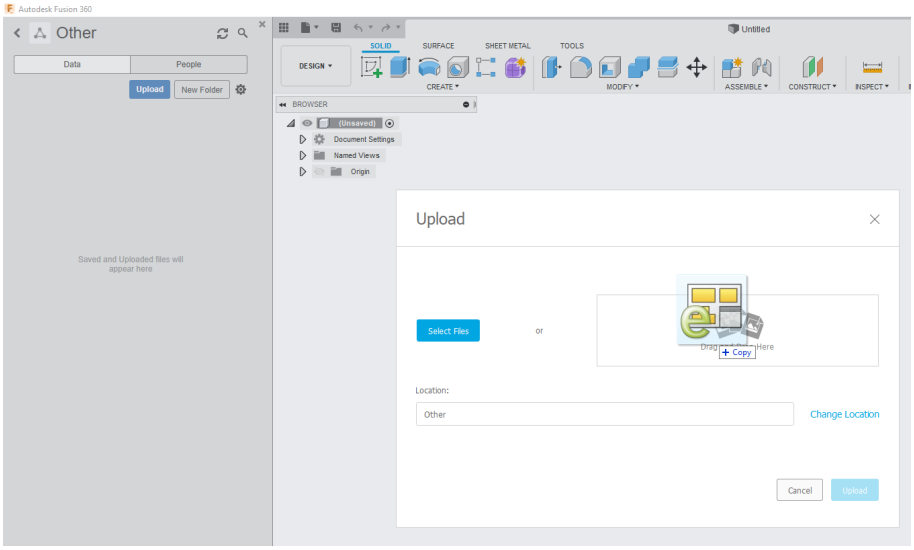


Figure 182: Upload CAD file into Fusion 360.

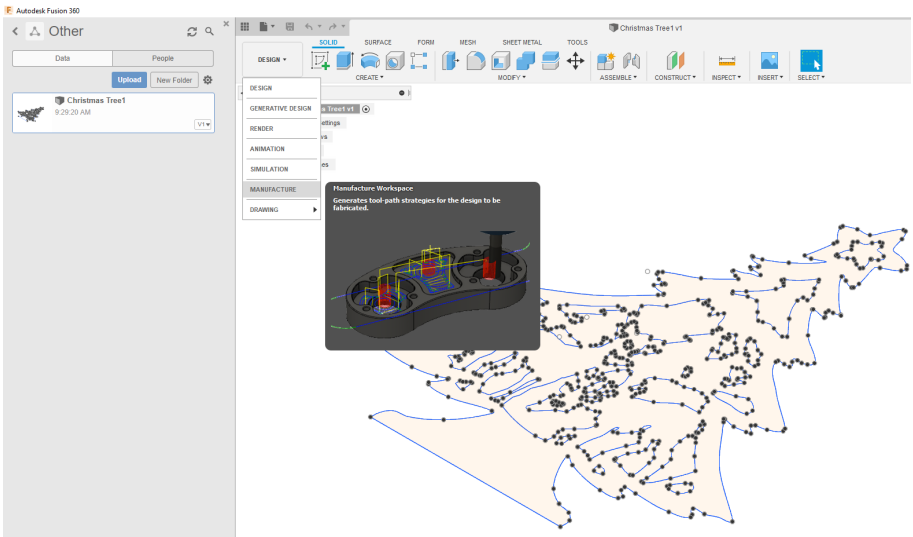


Figure 183: Fusion 360: Design - Manufacture.

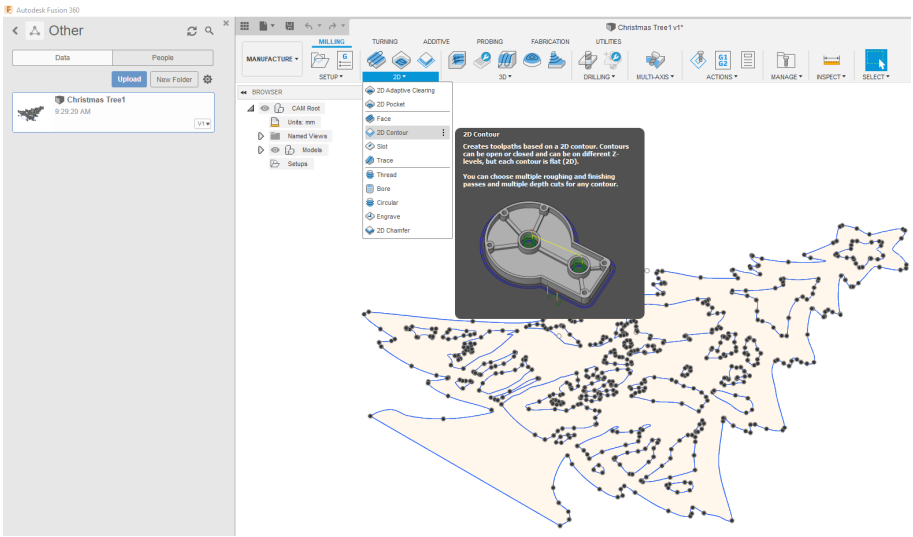


Figure 184: Fusion 360: Design - Manufacture - 2D Contour.

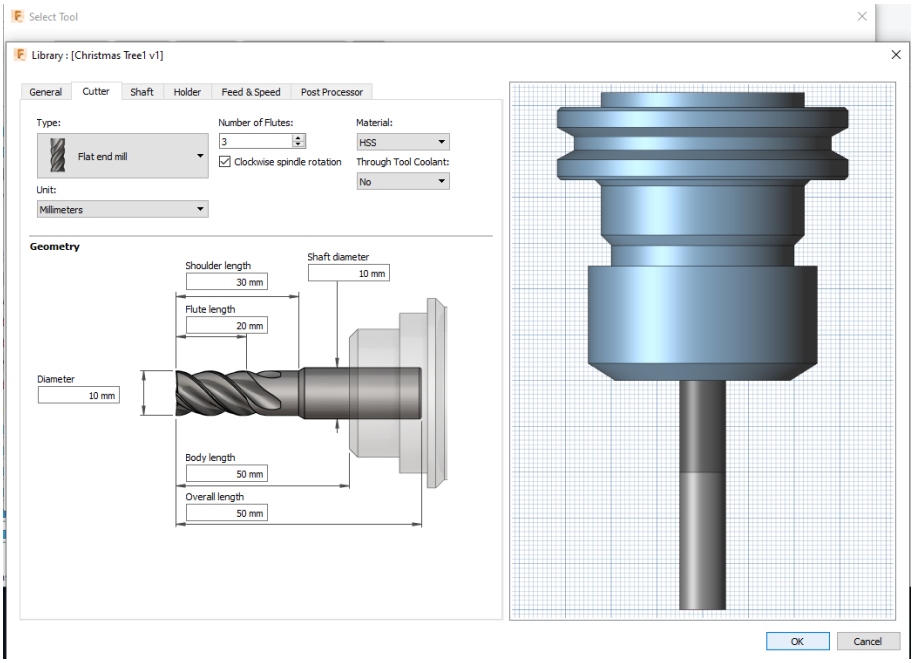


Figure 185: Fusion 360: Define milling tool.

Figure 186: Fusion 360: Contour Selection.

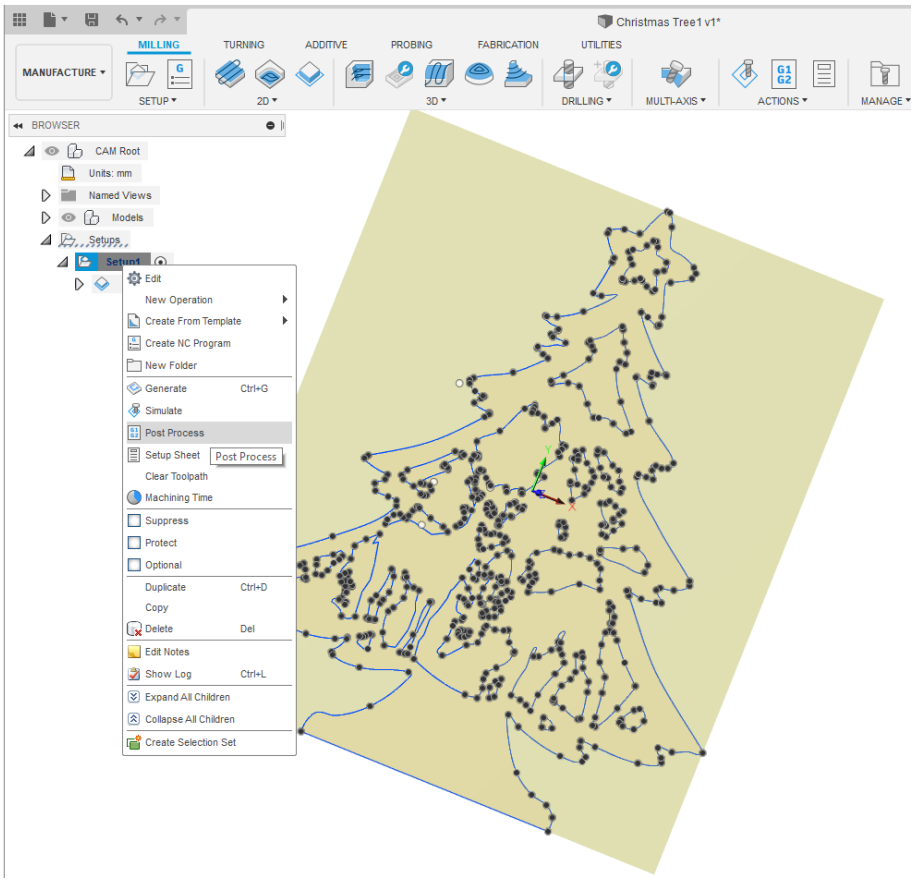


Figure 187: Fusion 360: Post Process to APT-CLS format.

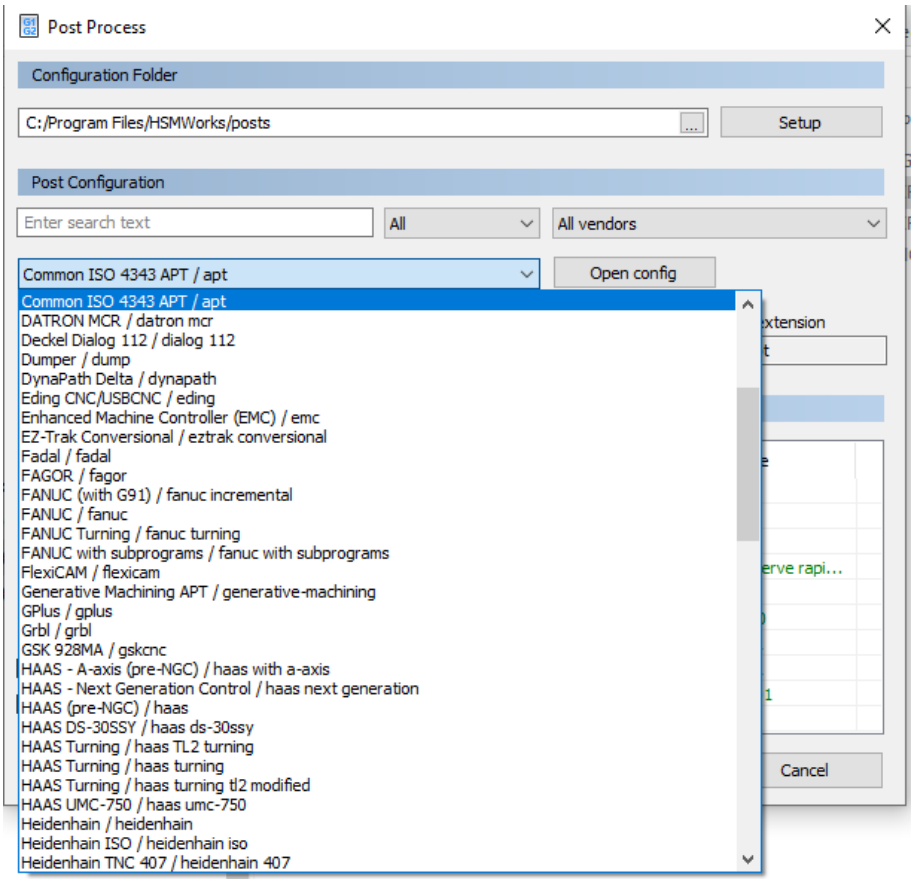


Figure 188: Fusion 360: Common ISO 4343 APT / apt.

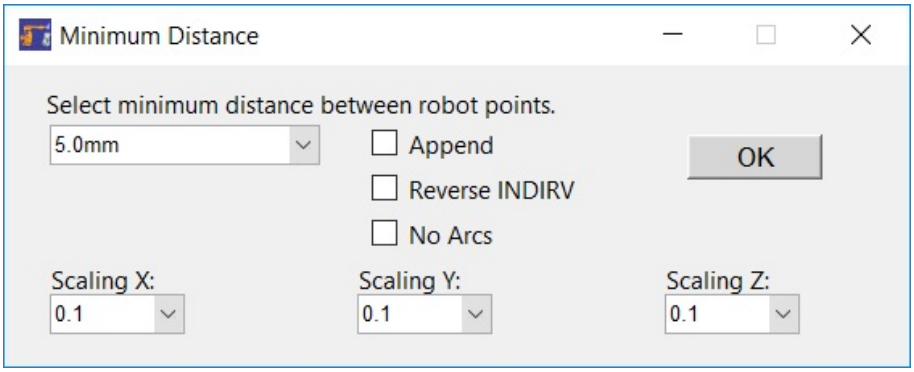


Figure 189: Scaling and Filtering APT File when Importing to IRBCAM.

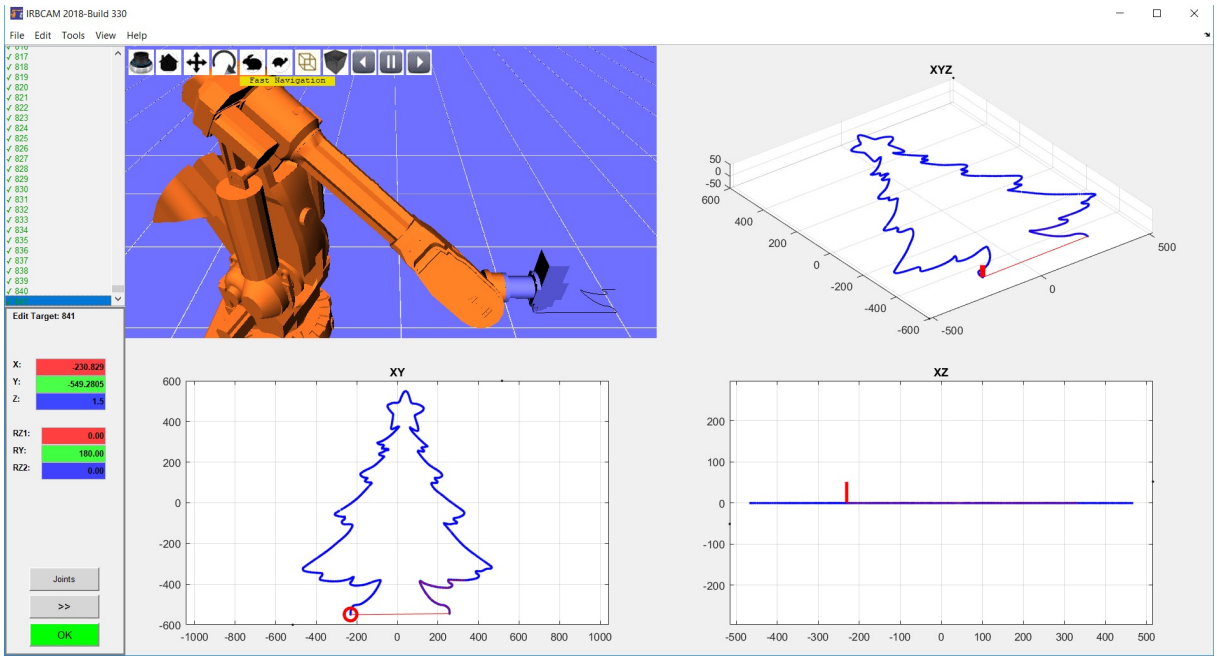


Figure 190: Toolpath After Configuration to Run on an ABB IRB6400-2.4m Robot.

12.5 Running IRBCAM on Multiple Cores

Modern CPU's have multiple cores, but the standard GUI version of IRBCAM is not able to utilise all this available computing power. Most of IRBCAM's algorithms are sequential and hence run on a single CPU core only. One way to overcome this limitation is to use IRBCAM's command-line interface and run several instances of IRBCAM in parallel. An example .bat file script to achieve this on Windows is shown below.

```
C:
cd "C:\Program Files\IRBCAM
start IRBCAM.exe 1 1.0 c:\test1.irb c:\test1.apt c:\test1.prg -startpercent 0 -endpercent 25
start IRBCAM.exe 1 1.0 c:\test1.irb c:\test1.apt c:\test2.prg -startpercent 25 -endpercent 50
start IRBCAM.exe 1 1.0 c:\test1.irb c:\test1.apt c:\test3.prg -startpercent 50 -endpercent 75
start IRBCAM.exe 1 1.0 c:\test1.irb c:\test1.apt c:\test4.prg -startpercent 75 -endpercent 100
```

In this script, four independent processes of IRBCAM are started up in parallel. The optional parameters '-startpercent' and '-endpercent' are used. The first process converts the first 25% of the input APT file, the second process the next 25% of the APT file, and so on. The configured toolpaths are stored in four separate PRG files, test1.prg to test4.prg (mode=1 converts to the ABB RAPID language). The station file test1.irb must contain all the required parameters needed to configure the toolpath, for example tooldata and userframe definitions, as well as the initial tool-roll angle. test1.irb normally would contain a single robot target only, to keep the file small and to allow definition of the initial tool roll angle.

If all the four processes above are able to configure the toolpath successfully, the final output will be as follows (two lines of text for each process):

```
1
IRBCAM path configuration succeeded.
1
IRBCAM path configuration succeeded.
1
IRBCAM path configuration succeeded.
1
IRBCAM path configuration succeeded.
```

One potential drawback of such parallel processing of the APT file, is the fact that each process starts out with the same tool roll angle as defined in the test1.irb file. When the entire APT file is configured sequentially in the GUI version of IRBCAM, the tool roll angle changes dynamically depending on for example the distance to the joint limits and singularities. Hence, there is no guarantee that an APT file which is possible to configure sequentially in the GUI version of IRBCAM will also configure successfully when split into parallel processes as in the example above. The potential benefit, however, is large. By parallelising the processing of the APT file, the overall computation time can be reduced by a factor approximately equal to the number of CPU cores. This is particularly of benefit for large toolpaths and for time-consuming calculations such as collision detection.

13 Statistics

IRBCAM contains a menu item called "Tools - Statistics". By selecting this item (or pressing CTRL+A) after an APT file has been loaded, a window as shown in Figure 191 will appear. This window shows the minimum and maximum values in the

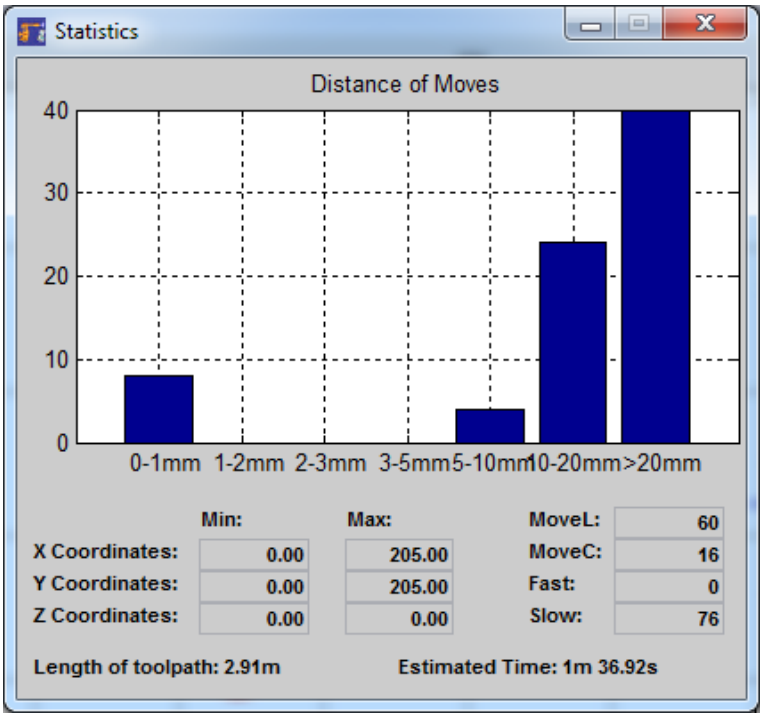


Figure 191: Robot coordinate statistics.

X,Y,Z directions, the number of MoveL vs MoveC instructions, the number of fast vs slow moves and the distribution of the coordinate distances. This information (especially the min/max values) can often be useful when configuring a toolpath. For example, if the maximum X position is very large and this position is not reachable by the robot, the user or object frames can be compensated in the negative X direction to shift the toolpath into the reach of the robot.

The second last row of the statistics window shows the shortest and the longest linear move. On some robot controllers, very long linear moves can be problematic, especially if the robot has to change configuration during the move or approach a singularity. To reduce the longest move and insert intermediate points, the menu 'Tools - Maximum Distance' can be used.

At the bottom the total length of the toolpath and the estimated time are shown. These numbers take both linear and circular moves into account. In practice, the time will be slightly longer than the estimate given by IRBCAM, since the robot normally has to reduce the speed at corners.

14 CAD Converter

The CAD converter is a separate program (CADConverter.exe) which can be used to convert your own CAD files to User Geometries or Tools to the IRBLIB format which can be used in IRBCAM. The CAD converter can be used either as a command-line tool or with a graphical user interface (GUI). The command-line options are specified below:

```
Usage: CADConverter <outfile> <infile1> <infile2> ... <infileN> <optargs>
     : With no command-line arguments, the GUI opens
     : The input files VRML (*.wrl) or STL (*.stl) are converted to the
     : IRBLIB format

Optional Arguments:
-tool num: If num=1 create tool, otherwise geometry
-X num   : X translation (mm or inch) of input geometries
-Y num   : Y translation (mm or inch) of input geometries
-Z num   : Z translation (mm or inch) of input geometries
-RX num  : RX rotation (deg) of input geometries
-RY num  : RY rotation (deg) of input geometries
-RZ num  : RZ rotation (deg) of input geometries
-sX num  : X scaling of input geometries (default 1)
-sY num  : Y scaling of input geometries (default 1)
-sZ num  : Z scaling of input geometries (default 1)
-tX num  : Tooldata X (mm or inch)
-tY num  : Tooldata Y (mm or inch)
-tZ num  : Tooldata Z (mm or inch)
-tRX num : Tooldata RX (deg)
-tRY num : Tooldata RY (deg)
-tRZ num : Tooldata RZ (deg)
-colR num: Color R (Red, between 0 and 1), STL only
-colG num: Color G (Green, between 0 and 1), STL only
-colB num: Color B (Blue, between 0 and 1), STL only
```

When calling the CAD converter from the command-line, it will run as a background process. In a batch file, if it is important to wait until the CAD converter is finished with it's task, call it with for example: `START /wait CADConverter -h`. The units (mm or inch) are determined by the user's settings defined using the GUI in IRBCAM.EXE or CADConverter.exe. The default unit is mm.

Figure 192 shows the opening screen of the CAD Converter when the GUI is opened (no command-line parameters). The Converter opens up in Wireframe mode, which makes imported CAD files located below the floor visible. CAD files

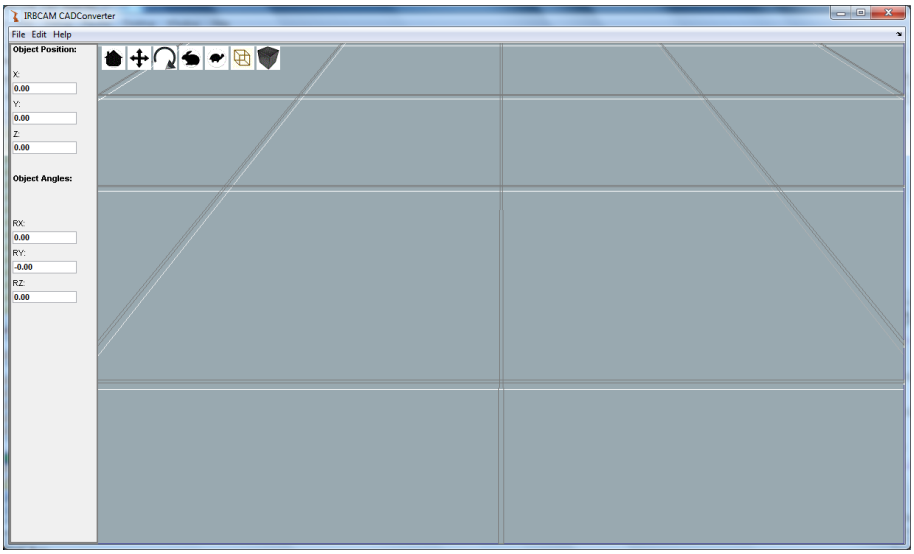


Figure 192: CAD Converter: Opening Screen.

saved as VRML (Virtual Reality Meta Language) or STL (Stereolithography) files can be converted to the IRBLIB format. Make sure that you draw your own CAD files in a CAD program which can export to at least one of these two formats. The first step is to load a VRML or STL file into the Converter.

Figure 193 shows the screen after a CAD file has been loaded. This CAD data is already located in the correct position. Otherwise, the translations X,Y,Z and the angles RX,RY,RZ in Figure 193 can be used to translate and rotate the imported CAD data.

In this example, a new tool will be generated. For new tools, it is important that the Z-axis of the CAD file corresponds to the direction straight out of the mounting flange of the robot. The X-axis of the CAD file will be pointing towards the floor when the tool is mounted on the robot and when the robot is in the zero position.

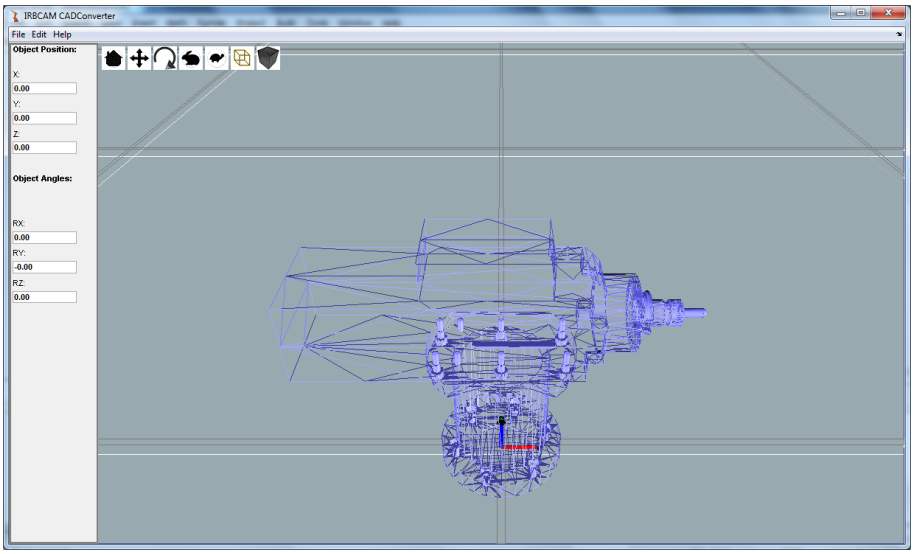


Figure 193: CAD Converter: Screen after a CAD file has been loaded.

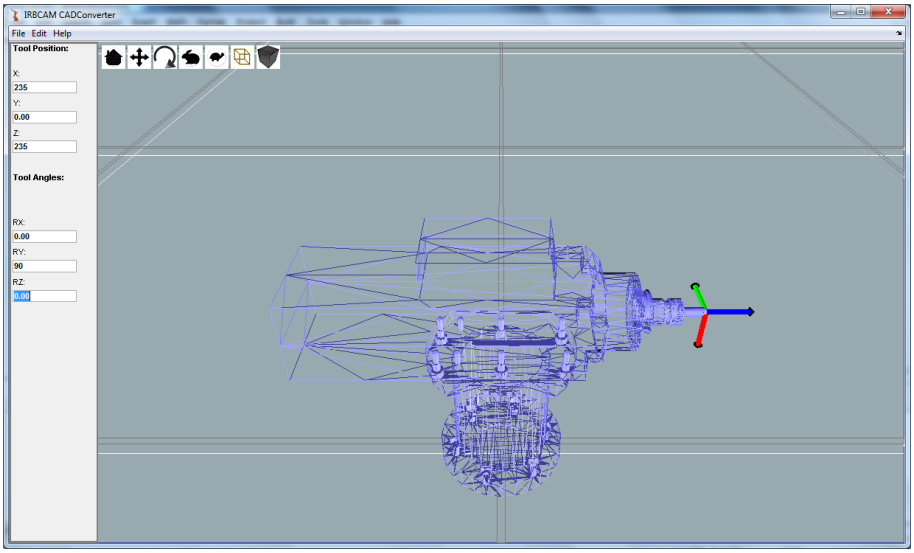


Figure 194: CAD Converter: Screen after a tool coordinate frame has been shifted to the correct location.

Note that the default location for the coordinate frame of the tool is located at the origin (0,0,0). The first step is to shift this coordinate frame to the correct location. Go to the menu 'Edit - Tool Data' (or CTRL+T) and enter the following values: X=235, Y=0, Z=235, RY=90°. You should then see the screen as shown in Figure 194. Note that the blue vector in Figure 194 is the Z-axis of the tool, and this vector must point in the direction of the cutting tool. That is the reason why the rotation RY=90° is required in this example. Figure 195 shows the screen when 'Edit

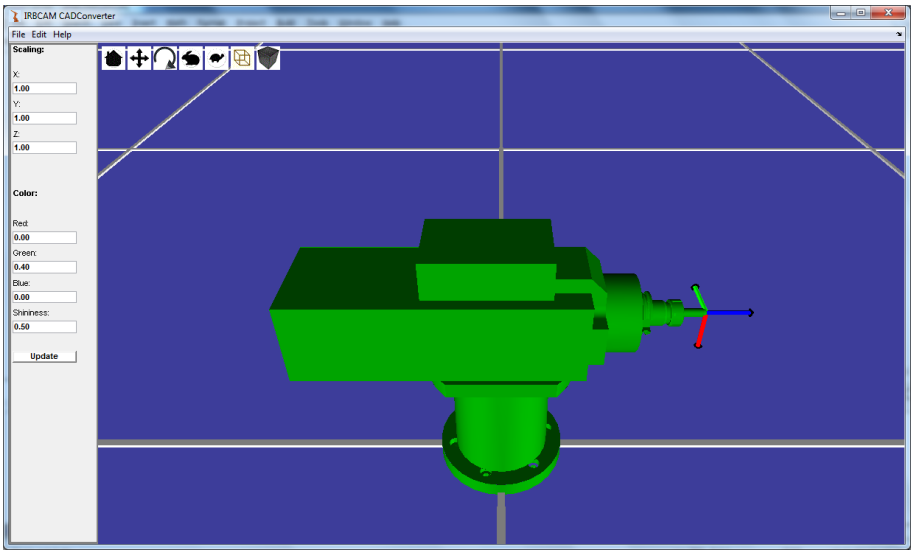


Figure 195: CAD Converter: Geometry properties.

- Geometry Properties' (or CTRL+P) has been selected. In this mode, the CAD file can be scaled in the X,Y and Z directions and it is possible to change the appearance of the object. In Figure 195 the 'Solid Mode' has been selected (as opposed to Wireframe) by using the Icons located at the top of the 3D graphics window. The RGB colours have been changes to 0,0.4,0. Click on 'Update' to change the colours in the graphics window. (All other parameters update automatically).

At this stage, the CAD file can be saved as an IRBLIB file. Choose 'File - Save Tool' (or CTRL+L) to convert and save the IRBLIB file. To make the new IRBLIB file available in the tool selection in the 'New Station Wizard' in IRBCAM, the file must be placed in the following directory:

`..\Program Files\IRBCAM\tool`

Note that on some operating systems, this directory will be Read-Only. In this case, save the IRBLIB file somewhere else first and then manually copy the IRBLIB file to the location above using Windows Explorer.

It is also possible to save the file as 'File - Save Geometry' (or CTRL+S). In this case, the tool data will be ignored and the generated file should be placed in the installation directory:

`..\Program Files\IRBCAM\geometry`

Files placed in the 'geometry' folder will become available in the 'New Station Wizard' in IRBCAM when selecting 'Station With User Object'. The reason why these directories have to be used, is to make it possible to transfer the Station files (*.irb) between two computers with different directory structures. Even if IRBCAM is installed somewhere else than the default `C:\Program Files` directory (for example `D:\Programs`, the transfer of station files between different computers will work as long as the new Tool and Geometry files are placed in the installation directories `..:\IRBCAM\tool` and `..:\IRBCAM\geometry`.

Note that for Motoman robots, the definition of the tool coordinate axes is different compared to ABB and Kuka robots. For Motoman robots, the X (red vector) and Y (green vector) axes are rotated 180 degrees about the Z-axis (blue vector). Figure 196 shows the same tool with the ABB/Kuka definition (Left, RY=90°) and the Motoman definition (Right, RY=-90°, RZ=180°).

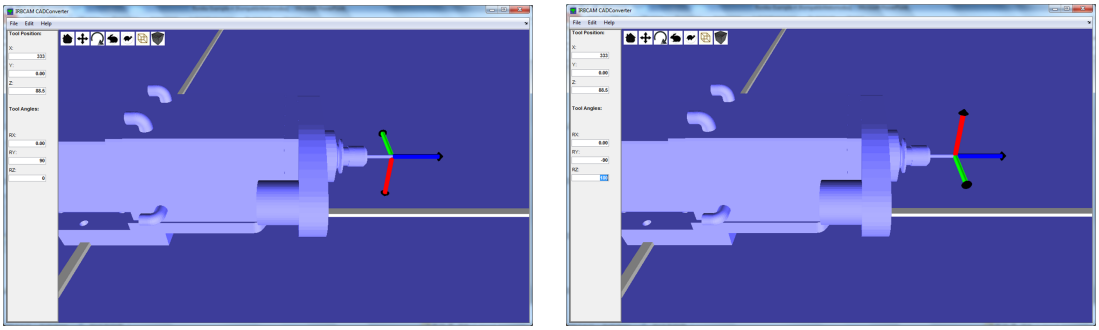


Figure 196: CAD Converter: Tool Definition on ABB/Kuka (Left) and Motoman (Right).

15 ABB Robot Controllers

The quick steps to uploading files (on ABB controllers) by using IRBCAM are the following:

- Option 1) Configure the Ethernet connection to the robot controller, see Figure 197. Connect to the robot with an FTP client (for example FileZilla)
- Option 2) On older controllers (S4 and S4C) put the RAPID code on a floppy disk.
- Option A) Upload the RAPID program. With this option, the number of robot coordinates is limited by the free memory size of the controller, typically 24,000 robot targets.
- Option B) Upload the system module "CAMSYS.SYS". Load this system module into the controller from the robot's teach pendant. This needs to be done only once. Upload a ROB coordinate file to the controller. Finally, upload the main module which reads and executes the ROB file to the robot controller. With this option, the number of robot coordinates is limited by the size of the controller's flash disk, typically several million robot targets.
- With both options A) and B), the programs can be run both in manual and automatic mode.

For ABB robots, the generated RAPID program and data points can be run on the S4, S4C, S4C+ and IRC5 robot controllers. While IRBCAM works on controllers that only have floppy disks, S4C+ and IRC5 type controllers have the added advantage of a large flash-disk and an Ethernet connection. For CAM files with several hundred thousand data points, IRBCAM can split the points across several floppy disks that can be inserted sequentially. Hence, the older S4C robot controller can execute as many CAM points as the newer S4C+ and IRC5 controllers. The advantage of S4C+ and IRC5 is the possibility of FTP file transfers and an uninterrupted execution of large CAM programs (with the MOD+ROB option). On a typical S4C+ controller, more than 3 million CAM points can be executed without interruption or the need for user interference.

When IRBCAM runs together with a stand-alone S4C+ or IRC5 system, an X-Ethernet cable and the service channel is one convenient way of transferring the CAM files. Figure 197 shows the required IP configuration on the PC running IRBCAM when connected to the service channel on an S4C+ system. The settings are the same for IRC5, except that the subnet mask should be set to $255.255.255.0$.

In order to use the MOD+ROB option, the "Advanced Functions" software module from ABB is required to access the CAM data points. The figures 199 to 201

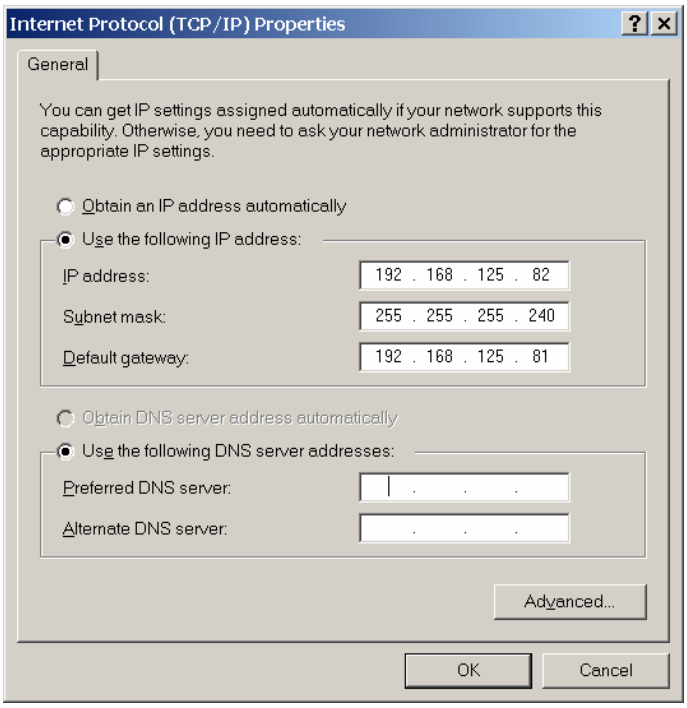


Figure 197: Configuration of PC IP-address when using the robot service channel.

show how to check which software options are installed on the robot controller. In Figure 201 "Advanced Functions" must be present. On IRC5 the option Advanced Functions is split into several separate options. The option required on IRC5 is called "File and Serial Channel Handling"..

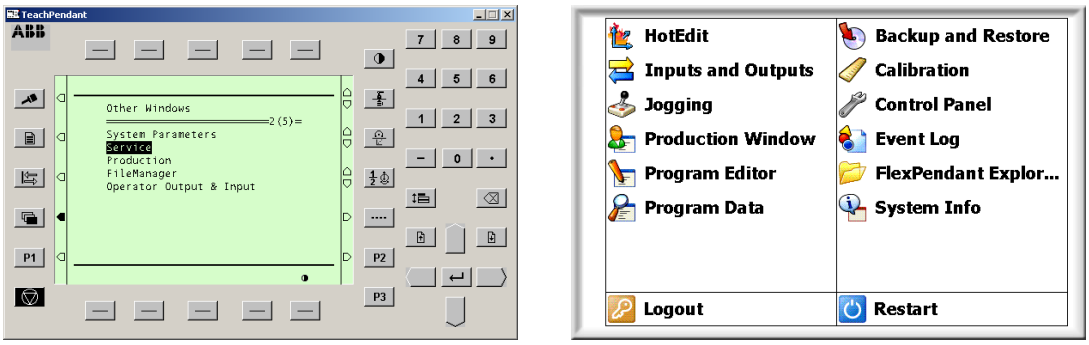


Figure 198: Service Screen on ABB (Left: S4C+, Right: IRC5).

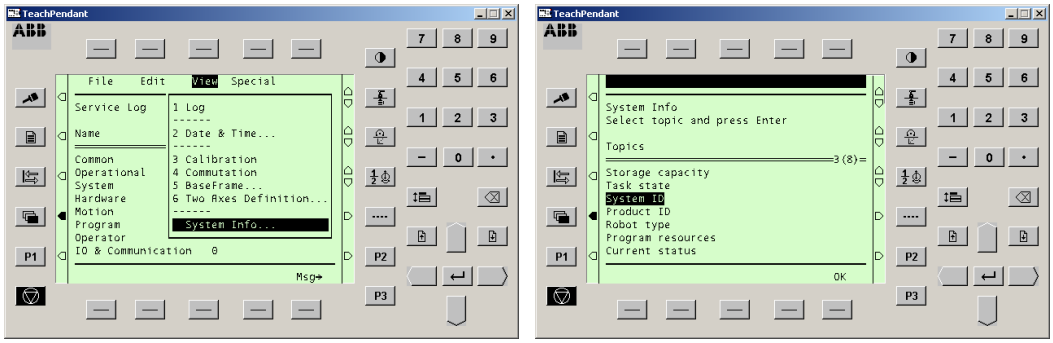


Figure 199: System Info and ID (ABB robots).

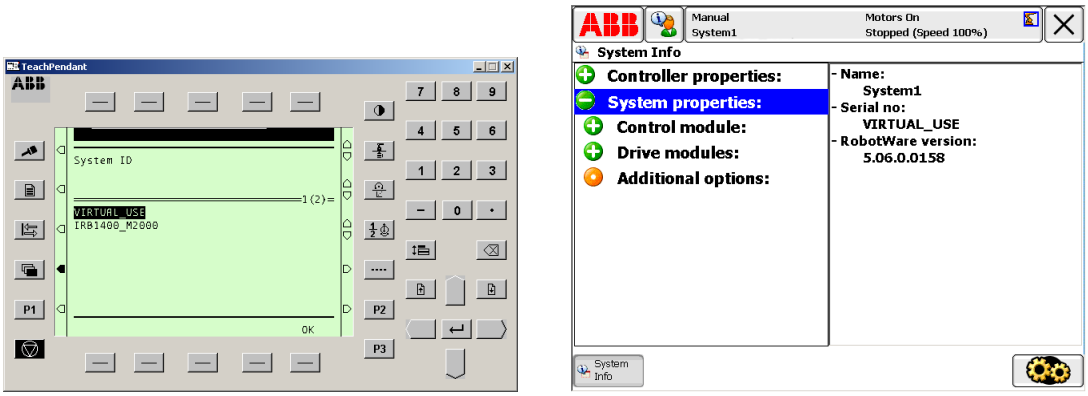


Figure 200: System Id Details (Left: ABB S4C+, Right: ABB IRC5)

16 Calibration

This chapter applies mainly to ABB robot, but the calibration approach is similar for other robot brands. The easiest way to calibrate the robot's tool and work object coordinate frames, is to use the built-in functionality in the robot controller. The tool will normally be calibrated first and the work object second. When calibrating the tool manually, a sharp point should be available in the work station. The operator then moves the tip of the tool to the sharp point with four different orientations of the tool. It is important to do this manual operation as accurately as possible. Any errors from this step will be directly visible in the final machined object, especially in 5-axis mode. A built-in function in the controller can then calculate the tooldata from the four positions recorded.

In this section the steps required to define a work object is presented in more detail. Note that (on ABB robots) a work object actually consists of two coordinate systems: the user frame (UFRAME) and the object frame (OFRAME), see Fig. 202. The operator can decide to calibrate only one of these frames or both.

From the jogging window on the teach pendant, the current work object can be

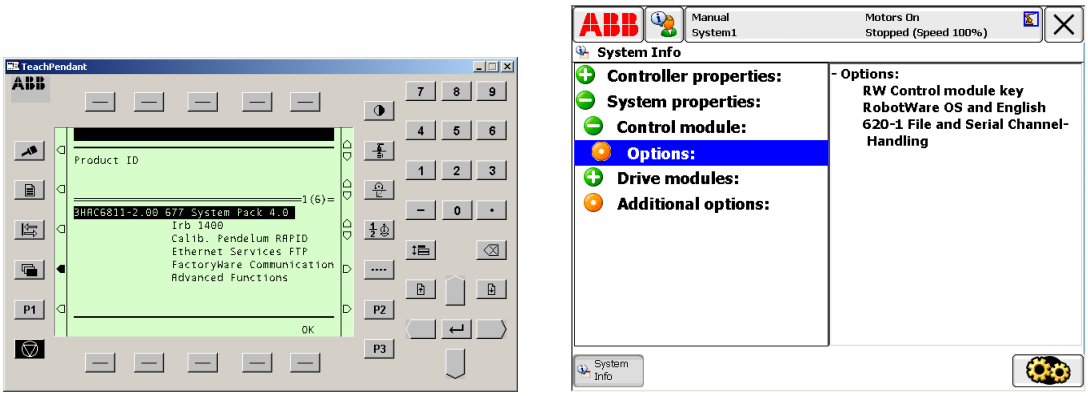


Figure 201: Product Id showing installed software options (Left: ABB S4C+, Right: ABB IRC5)

selected, see Figure 203. When the screen in Figure 204 appears, press the "Define..." button on the teach pendant and Figure 205 will appear. In Figure 205 the user coordinate frame can be defined by three points: the origin (X1), one point on the X-axis (X2) and one point on the Y-axis (Y1). The operator will have to manually jog the robot to the three points and press "ModPos" on the teach pendant. The object frame can often easily be calibrated manually once the user frame is defined, especially when the user and object frames have the same orientation. In that case the oframe quaternions will be $1, 0, 0, 0$ and the offsets X, Y, Z can be measured manually.

When the definition of the work object frame is completed, the operator can return to the Instruction view on the teach pendant screen as shown in Figure 206. By selecting "Edit - Value" from the menu on the teach pendant, the screen in Figure 207 will appear showing the calibrated coordinates and quaternions.

Figures 208 and 209 show an example of how the user frame is defined. The three coordinate points X1, X2 and Y1 are shown in Figure 208 and the final result is shown in Figure 209. The UFRAME coordinates are 987.766, 320.628, 186.489 while the quaternions are 0.712462, 0.002028, 0.001521, -0.701707. These values are recorded and entered when setting up a new station, see section 7.

The user frame part is in the previous example defined to the corner of the machining table. However, the object to be machined is usually located in a different position than the origin of the machining table. The object frame (OFRAME) can be used to define the translation and rotation of the object from the corner of the machining table to the actual position. In this case the OFRAME is a simple translation from the UFRAME (no rotations) and this translation can be measured by a ruler. In this example the OFRAME is measured simply as $X = 107, Y = 0, Z = 24$.

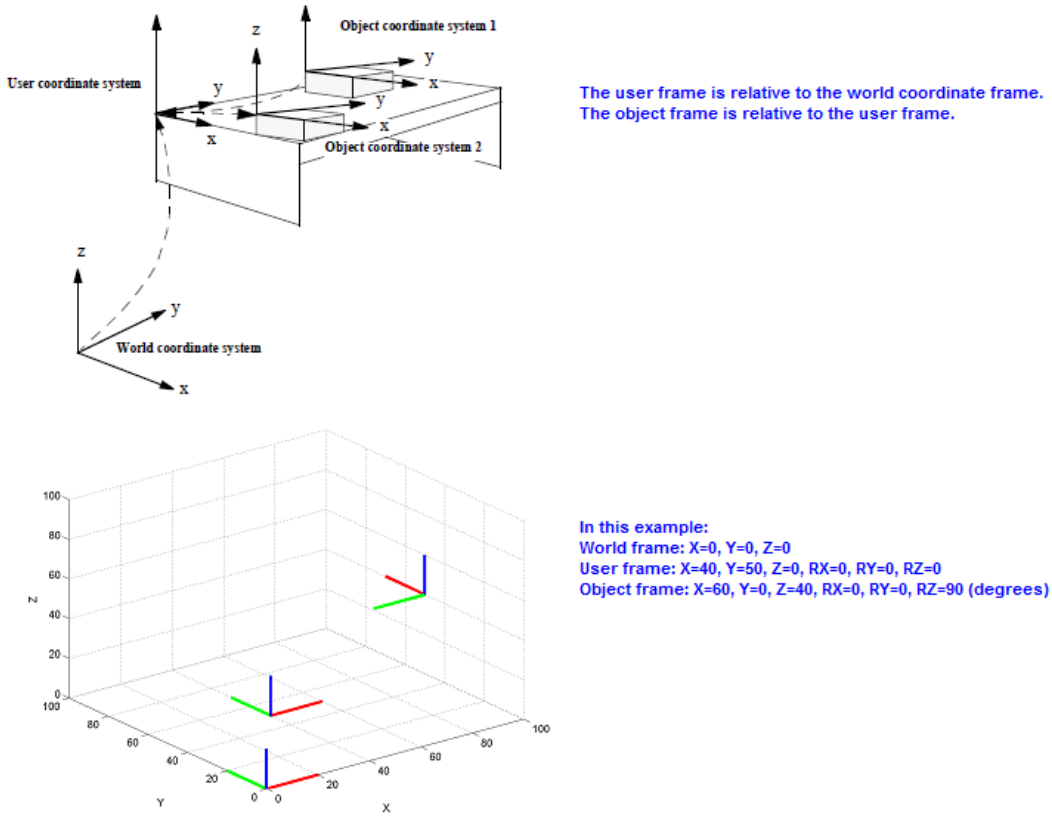


Figure 202: Definition of user (UFRAME) and object frames (OFRAME) on ABB robots.

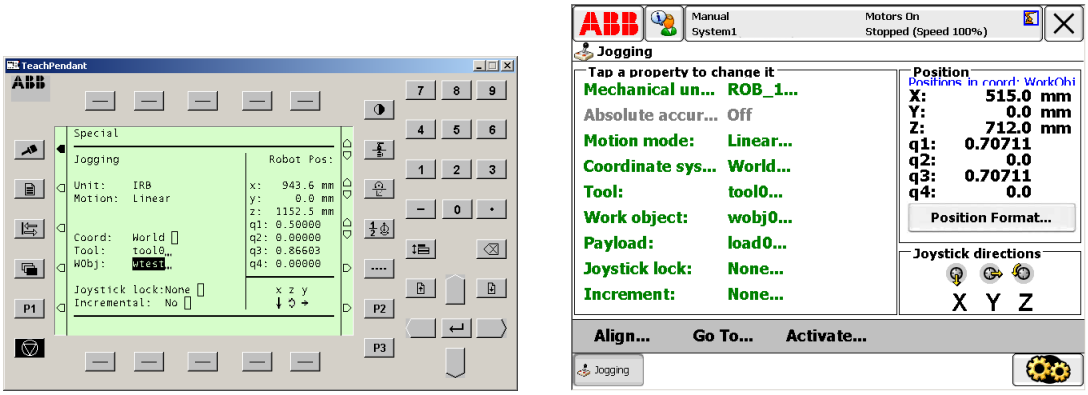


Figure 203: Work object definition step 1 (Left: S4C+, Right: IRC5).

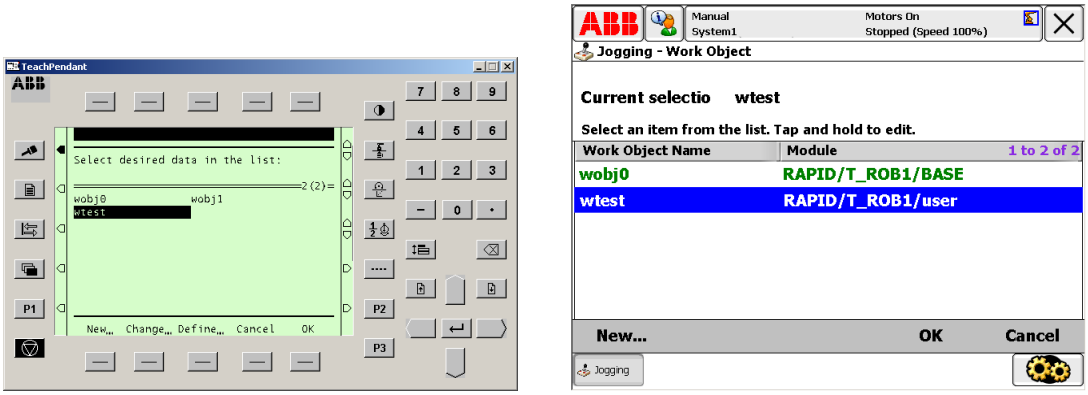


Figure 204: Work object definition step 2 (Left: S4C+, Right: IRC5).

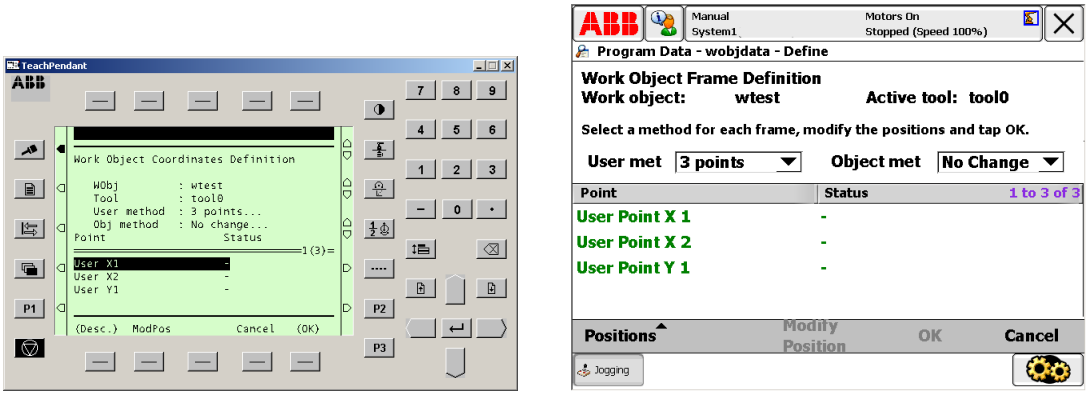


Figure 205: Work object definition step 3 (Left: S4C+, Right: IRC5).

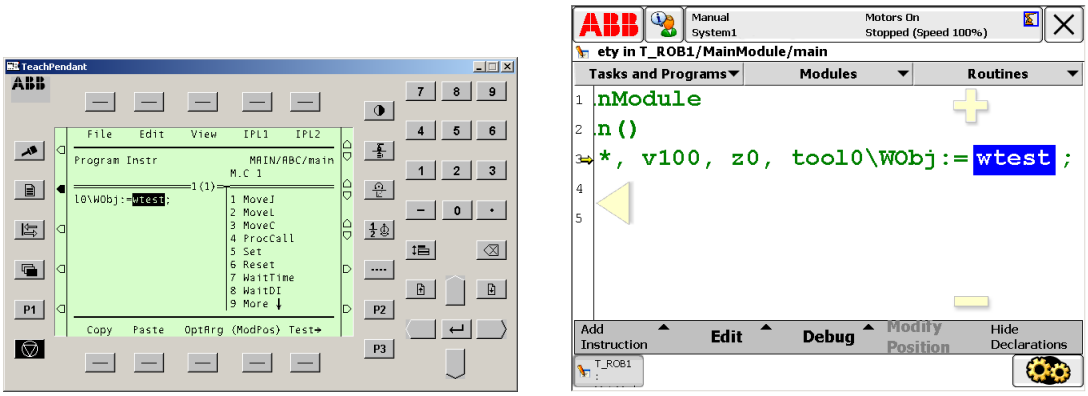


Figure 206: Work object definition step 4 (Left: S4C+, Right: IRC5).

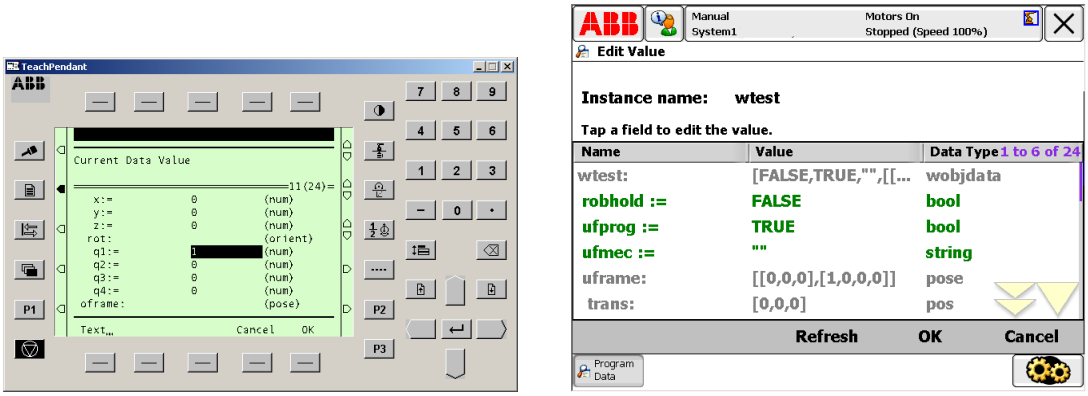


Figure 207: Work object definition step 5 (Left: S4C+, Right: IRC5).

Figure 208: Work object definition. Top: Origin, Middle: X-Axis, Bottom: Y-axis

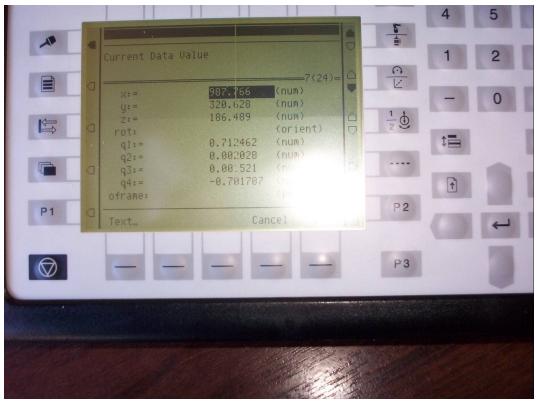


Figure 209: Work object result.

17 Load and Unload of RAPID Split Files

As an alternative to executing toolpaths with the MOD+ROB option, ABB RAPID programs can also be executed using a combination of the functions calls Load, CallByVar and UnLoad. The ABB controllers, at least the older ones, can only hold one program in memory at a time. The maximum number of toolpath lines is typically 24,000. In order to execute toolpaths with a much larger number of points than this, the individual files can be loaded, executed and unloaded one by one.

IRBCAM can split the toolpath into many smaller RAPID programs, by using the options shown in Fig. 210. In this example "Add Lift Points" is selected. This option ensures that the tool lifts off the part when the toolpath is split. This is to avoid potential damage to the part due to small vibrations of the robot and spindle when the next split file is loaded by the controller. The procedure name in this example is selected as F_ which will be explained later. The number of lines in each split file is selected as 20,000. When saving the RAPID code in this example, the chosen filename should be F.mod. The individual split files will then be named F_1.mod, F_2.mod, F_3.mod, etc.

The following RAPID code can be used as the main program to load and execute the individual split files of the toolpath.

```
%%%
  VERSION:1
  LANGUAGE:ENGLISH
%%%

MODULE MILL
  PROC main()
    var num choice;       !1 for Roughing, 4 for Finishing
    var num N;            !Number of split files
    var num strt;         !Where to start
    var num i;            !Loop counter
    var string piece;     !Counter on string format
    var string prog_name; !R=Roughing, F=Finishing

    AccSet 80,25;
    TPErase;
    TPReadFK choice,"Name of program?","R","","","F","";
    IF choice=1 THEN
      prog_name:="R";
    ELSEIF choice=4 THEN
      prog_name:="F";
    ENDIF
```

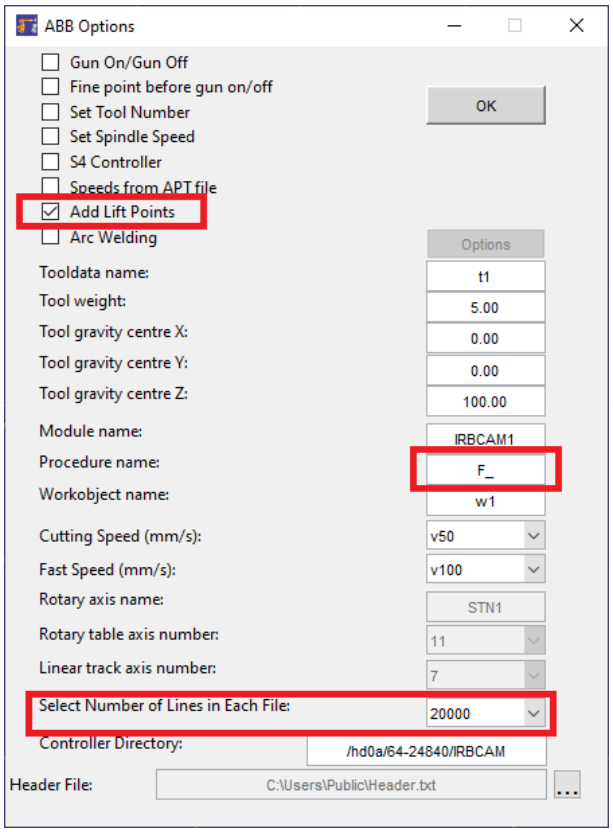


Figure 210: ABB RAPID options.

```
    TPErase;
    TPReadNum N,"How many pieces does the program contain?";
    TPReadNum strt,"Where to start it?";
    FOR i FROM 1 TO N DO
      piece:=NumToStr(strt,0);
      Load "HOME:" \File:=prog_name+"_"+piece+".mod";
      CallByVar prog_name+"_",strt;
      UnLoad "HOME:" \File:=prog_name+"_"+piece+".mod";
      Incr strt;
    ENDFOR
  ENDPROC
ENDMODULE
```

In this example two types of filenames can be selected. R stands for roughing and the split files would then be named R_1.mod, R_2.mod, R_3.mod, etc. If roughing files are generated, then R_ should be defined in Fig. 210 instead of F_. F stands for finishing and the split files would be named F_1.mod, F_2.mod, F_3.mod, etc. After the type of file is chosen (R or F), the operator has to enter the number of

split files (N) followed by the starting number. A starting number different from 1 can be useful if a previous toolpath execution was not finished completely, and the operator wants to resume the milling operation where it was previously stopped.

The directory "HOME:" in the program above can be changed to something else, for example "HOME:/IRBCAM" if the operator prefers to use a separate folder. Alternatively, it can be "pc:" if a PC disk has been mounted to the controller, or even "flp1:" if a floppy-to-USB interface is used. The use of standard floppy disks is not recommended, since these are both slow and can not store large toolpath files.

18 KUKA KRC1/C2 Controller

Chapter 18.1 contains information about how to set up networking for the KRC1/C2 controller and how to copy a SRC program generated by IRBCAM from an external PC onto the controller. Chapter 18.2 contains information about how dynamic loading / drip-feeding can be set up on the KRC1/C2 controller.

18.1 KRC1/C2 Networking

Fig. 211 shows how to connect the network cable to the KRC1/C2 controller. Note that the connector used is normally connected to the teach-pendant. Disconnect the existing cable in the RJ45 connector and connect the network cable for your own local network. To setup the IP-address on the robot controller, you first have

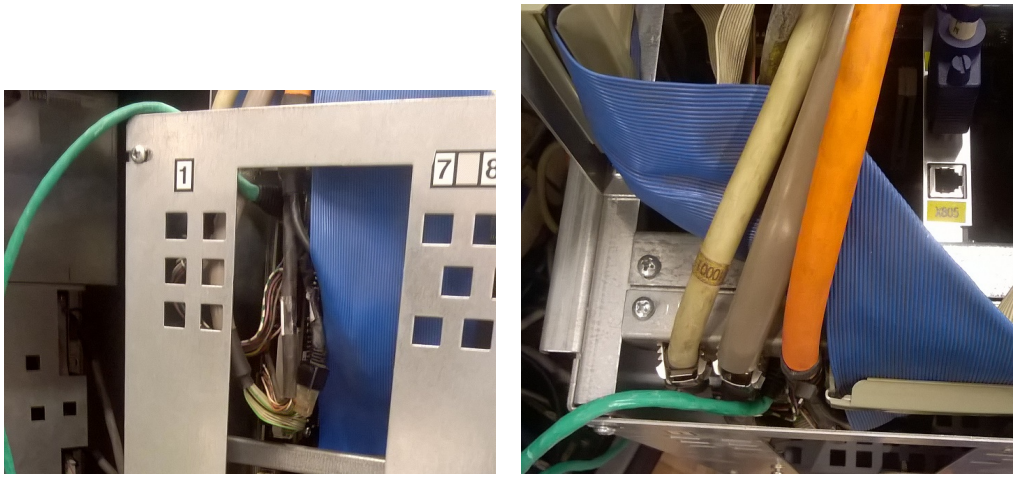


Figure 211: KRC1/C2: Connection of network cable.

to enter Expert Mode on the teach pendant. Fig. 212 shows how to do this (in this example the teach pendant is setup with Swedish language). Select 'Configure - User Group', then select 'Expert'. Type the password kuka to enter 'Expert Mode'. Next, simultaneously press CTRL and ESC on the teach pendant to enter the embedded Windows operating system as shown in Fig. 213. Select 'Control Panel - Networking'. If you have no mouse connected to the controller, use the TAB and ENTER keys to navigate through the dialog windows. ALT and ESC are also useful to enter and leave the menus. Fig. 214 shows how to define the example IP-address 192.168.1.101 for the controller. After this has been done, the controller must be rebooted. When the controller has been rebooted, the controller's network disk can be accessed from an external PC running Windows. Fig. 215 shows the File Explorer when the robot's network disk has been mapped to the PC's Z: drive. By using the File Explorer copy the generated file from IRBCAM

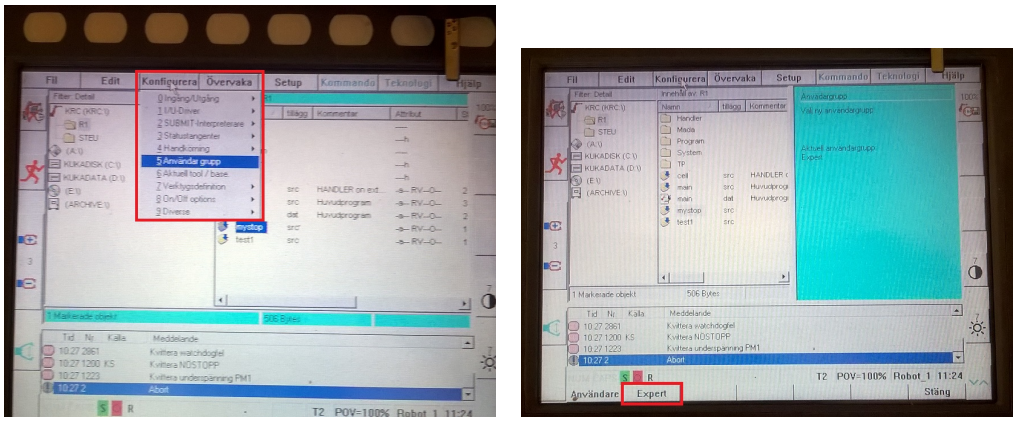


Figure 212: KRC1/C2: Expert User.

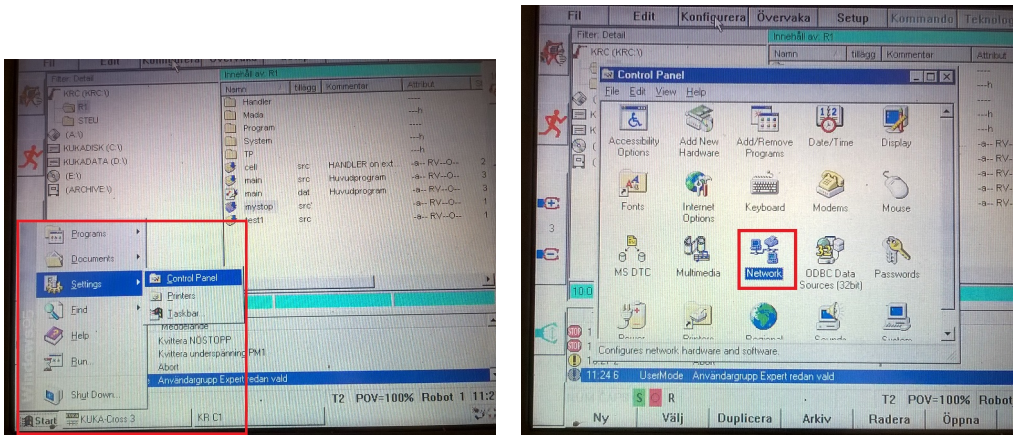


Figure 213: KRC1/C2: Control Panel and Network Settings.

onto the controller's disk, in this example the filename is test1.src. The SRC file must be copied from the controller's network disk to the internal memory of the controller. Fig. 216 shows how to do this. First, select 'File - Copy' on the network disk, then navigate to the controller memory at KRC:\R1 and finally select 'File - Paste'. When the SRC file has been copied to the internal memory of the controller, activate test1.src on the teach-pendant as shown in Fig. 217. After the SRC file has been chosen/activated, the program can be run from the teach-pendant and the robot will finally start to move.

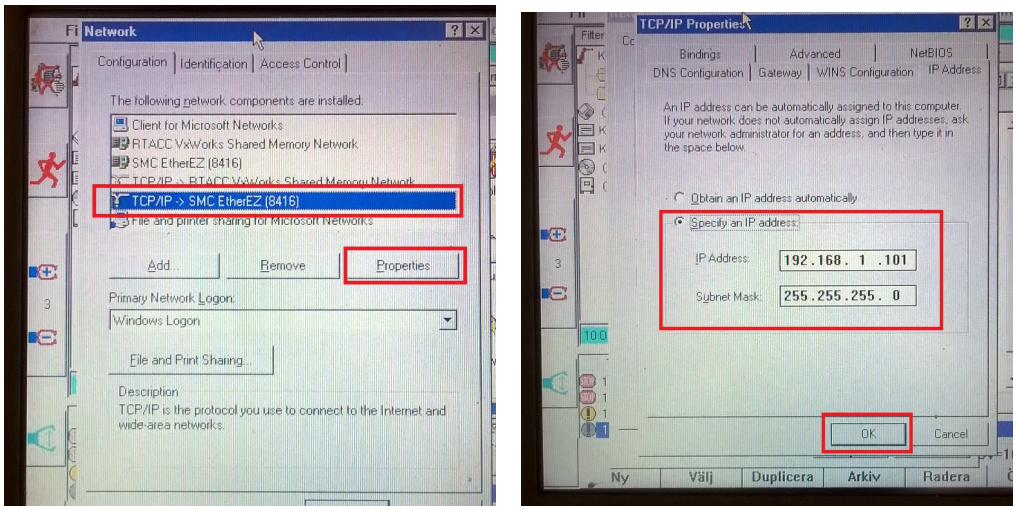


Figure 214: KRC1/C2: IP-address.

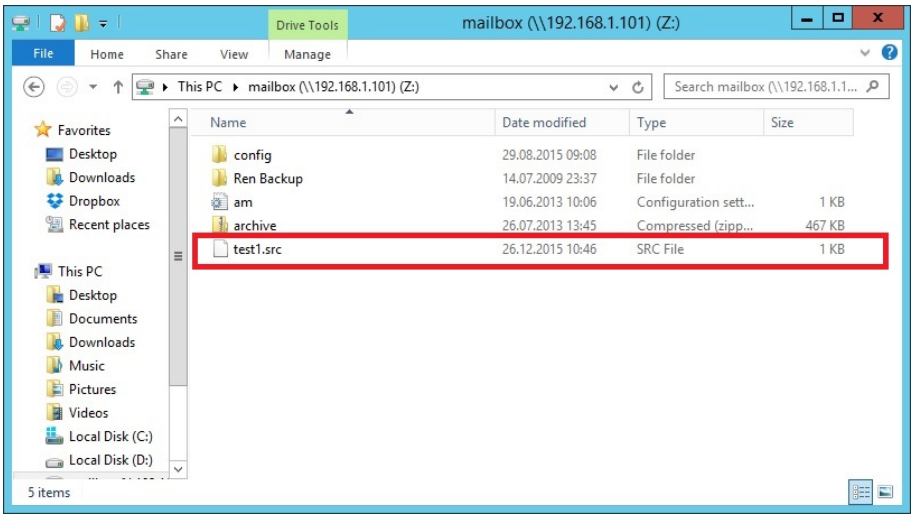


Figure 215: KRC1/C2: File Explorer in Windows on External PC.

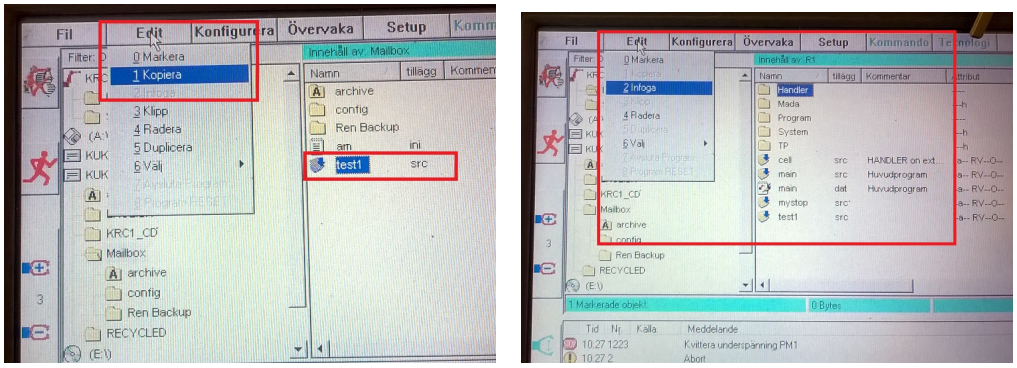


Figure 216: KRC1/C2: Copy+Paste From Network Disk to the Controller Internal Memory.

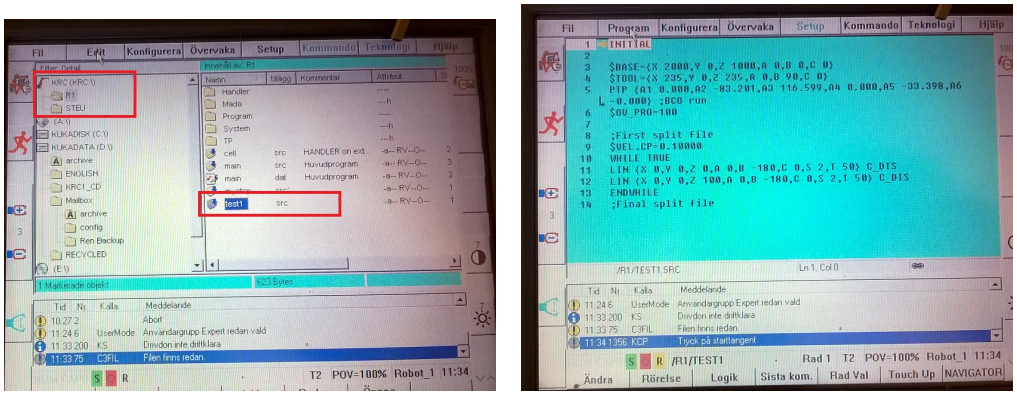


Figure 217: KRC1/C2: Select and Run Program Generated by IRBCAM.

18.2 KRC1/C2 Dynamic Loading

It is possible to save robot code for KUKA robots as two files (SRC+ROB). The SRC file contains code to load robot targets into memory using the RS-232 connection between the controller (vxWorks) and the Win95 operating system. The ROB file contains the actual robot targets, external axes, spindle and tool information. By exporting to the SRC+ROB combination, very large toolpaths can be run on a KR C1/C2 controller without the operator having to continuously load small files into memory.

To set up the controller to use the SRC+ROB format, the communication settings for both COM1 and COM2 must be defined as follows:

```
BAUD=9600
CHAR_LEN=8    ; 7,8
STOP_BIT=1    ; 1,2  at time not changeable
PARITY=0      ; EVEN=2, ODD=1, NONE=0

PROC=4        ; 3964R=1, SRVT=2, WTC=3, XONXOFF=4
```

These settings can be found in the file C:\KRC\ROBOTER\INIT\serial.ini

In addition, the settings in the same file for the XONXOFF protocol should be:

```
[XONXOFF]
CHAR_TIMEOUT=20       ; msec   Timeout after last received character
                      ; to recognize the end of telegram
MAX_TX_BUFFER=2       ; 1..5
MAX_RX_BUFFER=2       ; 1..20
SIZE_RX_BUFFER=100    ; 1..2048 longest expected telegram length + 15 characters
```

In the file C:\KRC\ROBOTER\INIT\hw_inf.ini, the following modifications have to be made:

```
[SERIAL]
COM1=DISABLE
COM2=ENABLE
```

Note for KRC1, COM2 can only be enabled for controllers of type KRC1 (KRC2 spec). After making these modifications to serial.ini and hw_inf.ini, the robot controller must be rebooted. A null-modem cable connected between COM1 and COM2 is also needed, see Fig. 218.

When the SRC+ROB files are generated by IRBCAM, copy the files to the KUKA controller. Normally, the files can be copied to the controller's Windows disk, typically located at D:\Mailbox. You also need to copy the Windows 95 program called

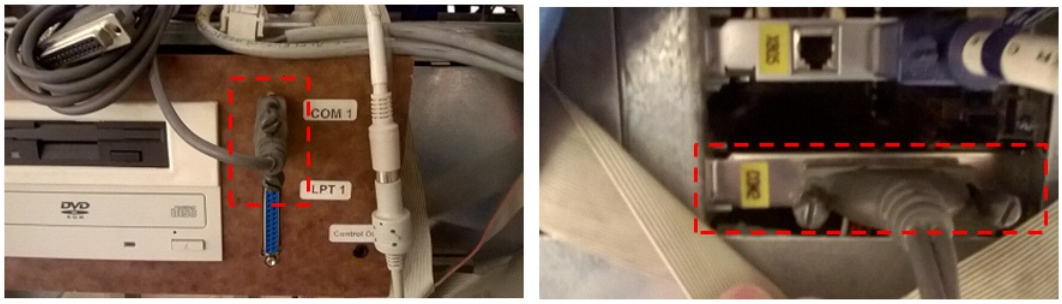


Figure 218: Null-modem cable between COM1 (Win95) and COM2 (vxWorks) in KRC1 controller.

KRC1.EXE from C:\Program Files\IRBCAM\krl to the KUKA controller's disk. In this manual, it is assumed that KRC1.EXE is copied to D:\Mailbox\config.

After copying these 3 files from the PC to the controller, copy the SRC file to the controller's internal memory, typically located at KRC:\R1. After that, select and open the SRC file. Switch the controller key to Automatic Mode and press ①, then run the SRC program. The program will then stop and wait for data on the RS-232 channel, see Fig. 219. The program will wait on the program line: `WAIT FOR ($DATA_SER2 <> 0)`.

Next, enter Windows 95 by pressing the Windows Key (CTRL ESC on the teach pendant). Then select 'Run' as shown in Fig. 220 and then type D:\Mailbox\config\KRC1.EXE as shown in Fig. 221. Use the TAB key to select 'OK', then press ENTER. The IRBCAM-KRC1 interface program will then open. Press ALT-F on the teach-pendant, then the File menu will open as shown in Fig. 222. Use arrow down to select 'File Open', press ENTER. Next, select the ROB file generated by IRBCAM. In this example this file is called test1.rob, as shown in Fig. 223. If File Open succeeds, the total number of robot targets and a status message 'File OK' will appear, as shown in Fig. 224. Finally, press ALT-C on the teach-pendant and select 'Send Data' as shown in Fig. 225. KRC1.EXE will now send 100 robot targets at a time to the vxWorks controller using the COM-ports. When 100 targets have been successfully sent, the robot will start to move and execute these 100 targets. Then, the robot will stop and the next 100 targets will be transferred. This process will continue until all the robot targets have been transferred and executed. The interface also supports external axes, spindle speed and tool change parameters.

The KRC1 interface includes a checksum and error checking. If an error occurs during the COM-port communication, then KRC1 may freeze and not respond to keyboard inputs. To force KRC1.EXE to stop communicating on the COM port, the SRC program 'mystop.src' can be executed. In particular, the command `CWRITE (SPD,SW_T,MW_T,''0'')`, see Fig. 226. The PTP command afterwards does not have to be executed. The SRC program 'mystop.src' can be found in the directory

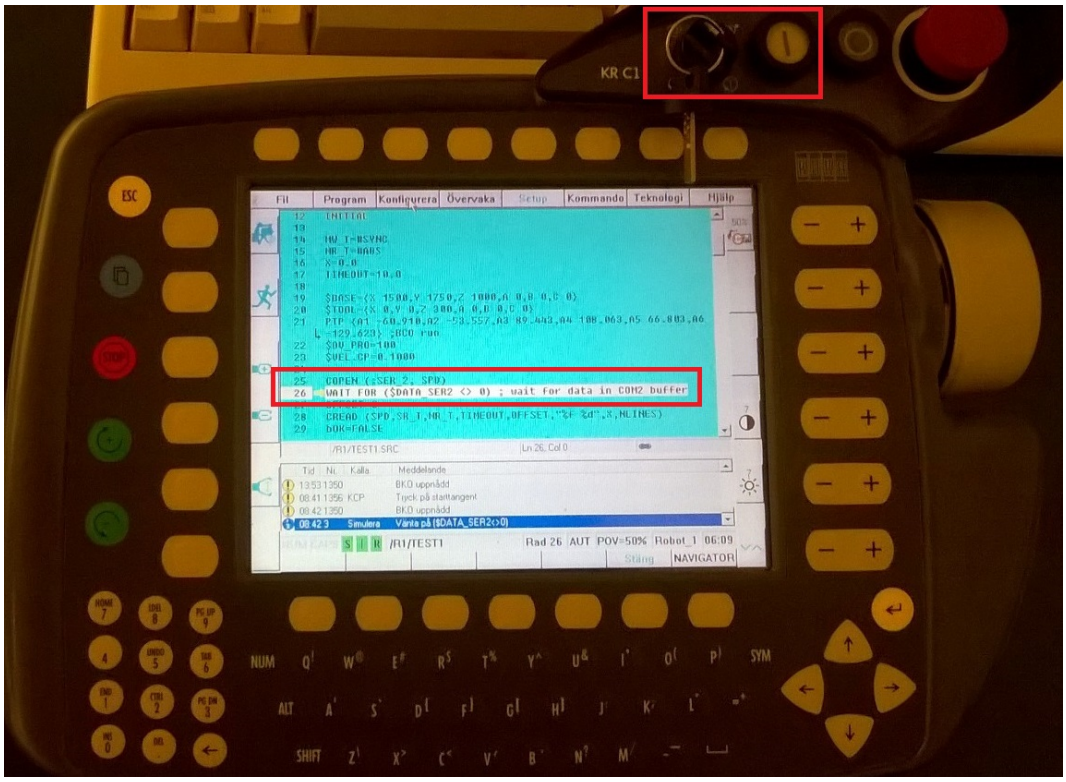


Figure 219: Starting SRC program in Automatic Mode. Waiting for data from Win95 application.

C:\Program Files\IRBCAM\krl. After all the robot targets have been transferred, or the communication has been aborted using 'mystop.src', then a small dialog window will appear, see Fig. 227.

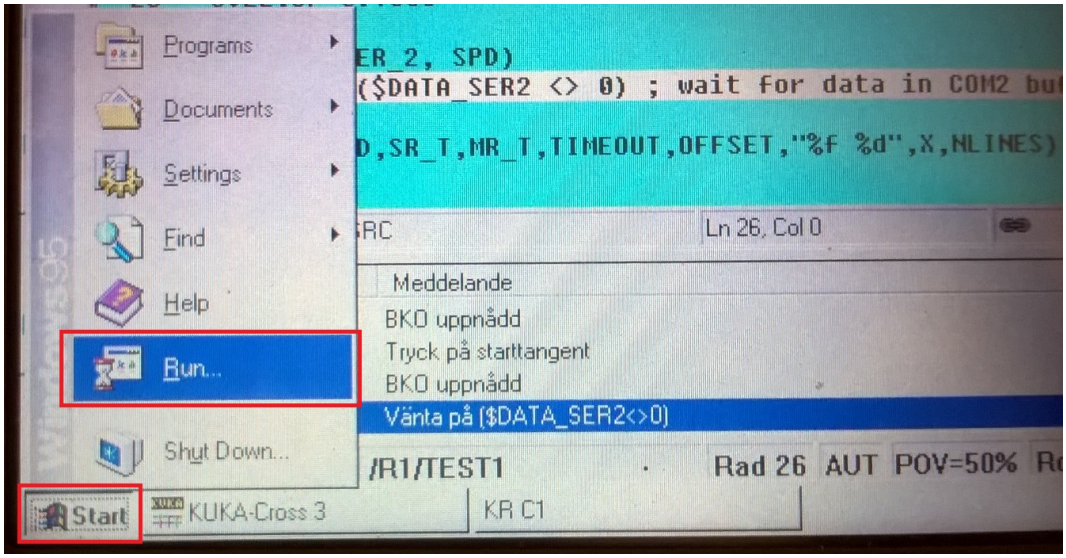


Figure 220: Windows Key - Start Menu - Run.

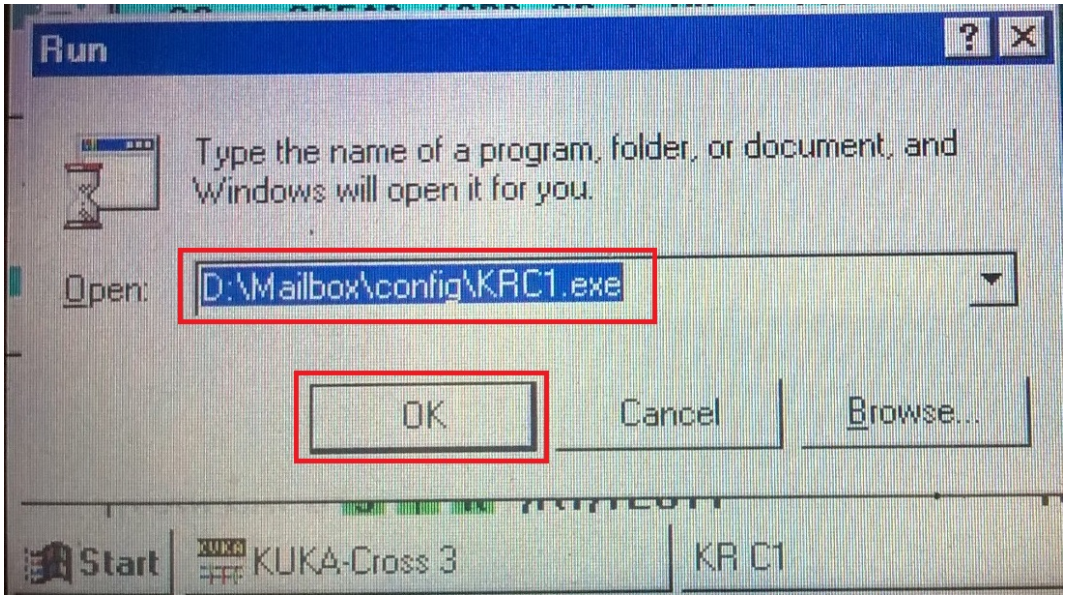


Figure 221: Run IRBCAM - Windows 95 application, KRC1.exe.

Figure 222: File Open - To open *.rob file generated by IRBCAM.

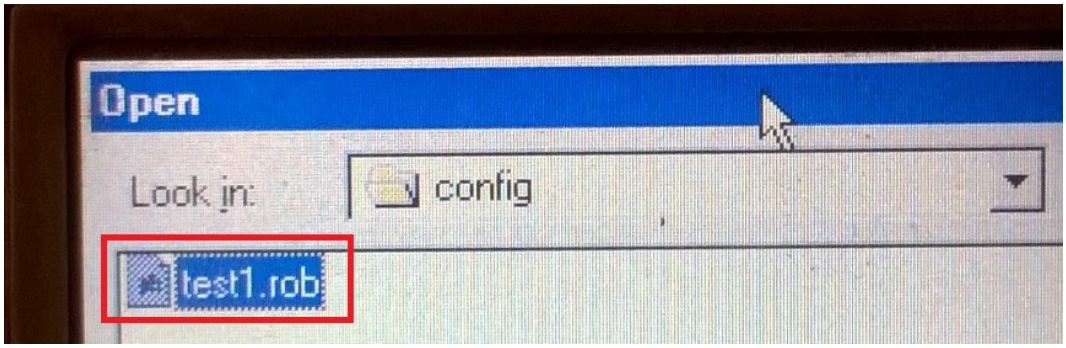


Figure 223: In this example test1.rob is loaded on the KRC1 controller (in Windows 95).

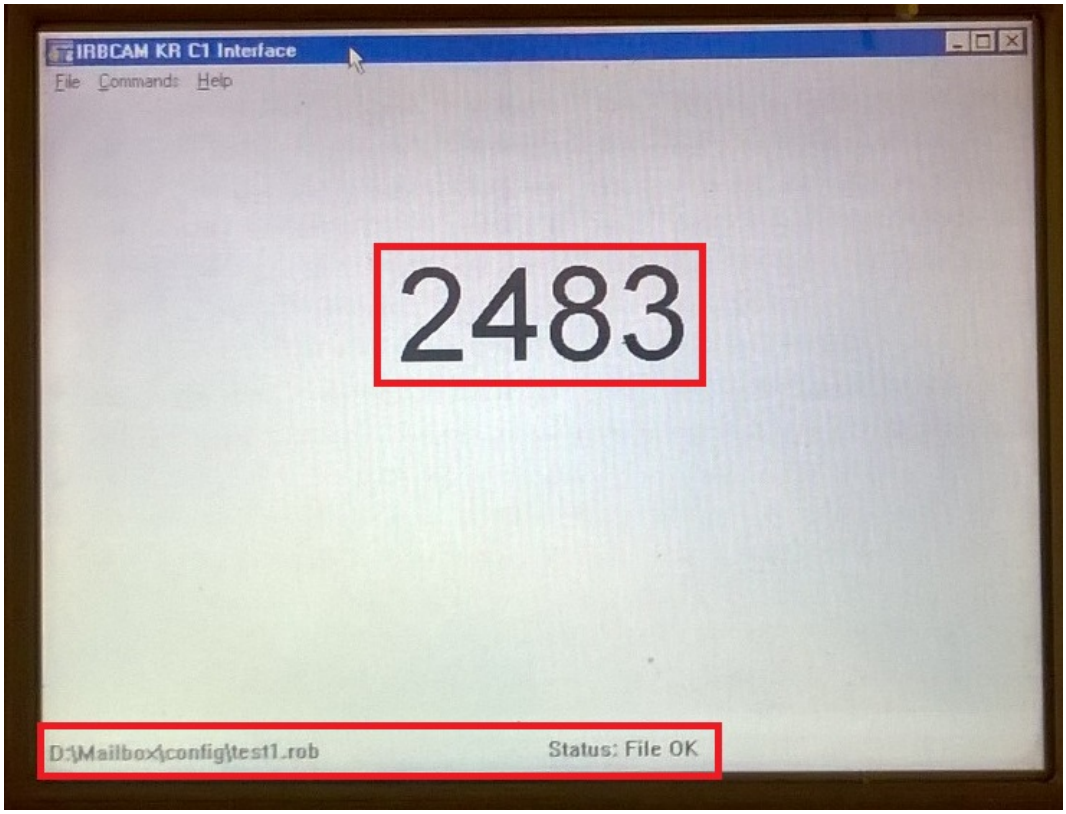


Figure 224: Status messages after test1.rob has been loaded.

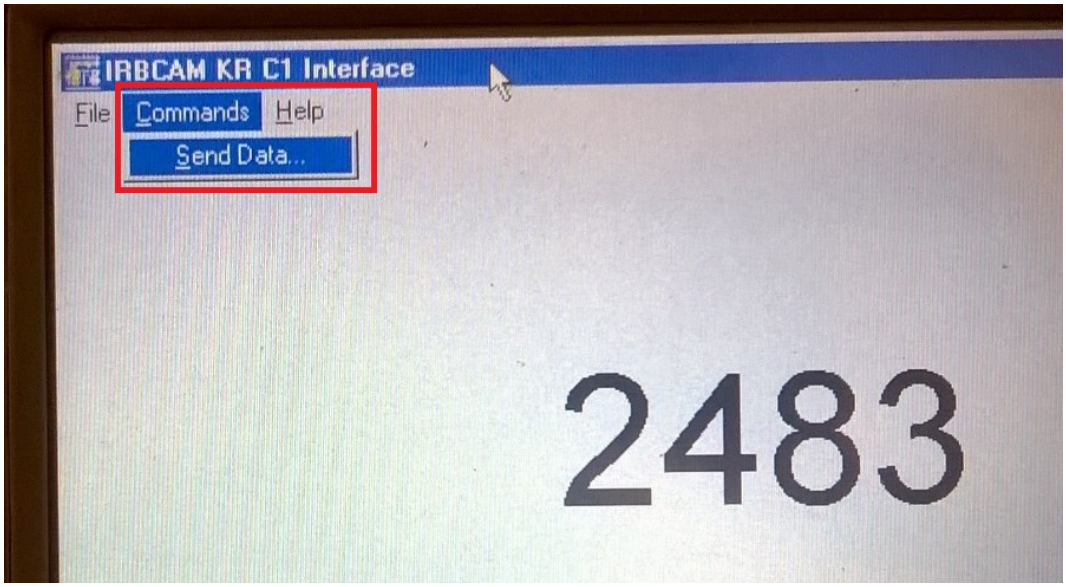


Figure 225: Send data to SRC program.

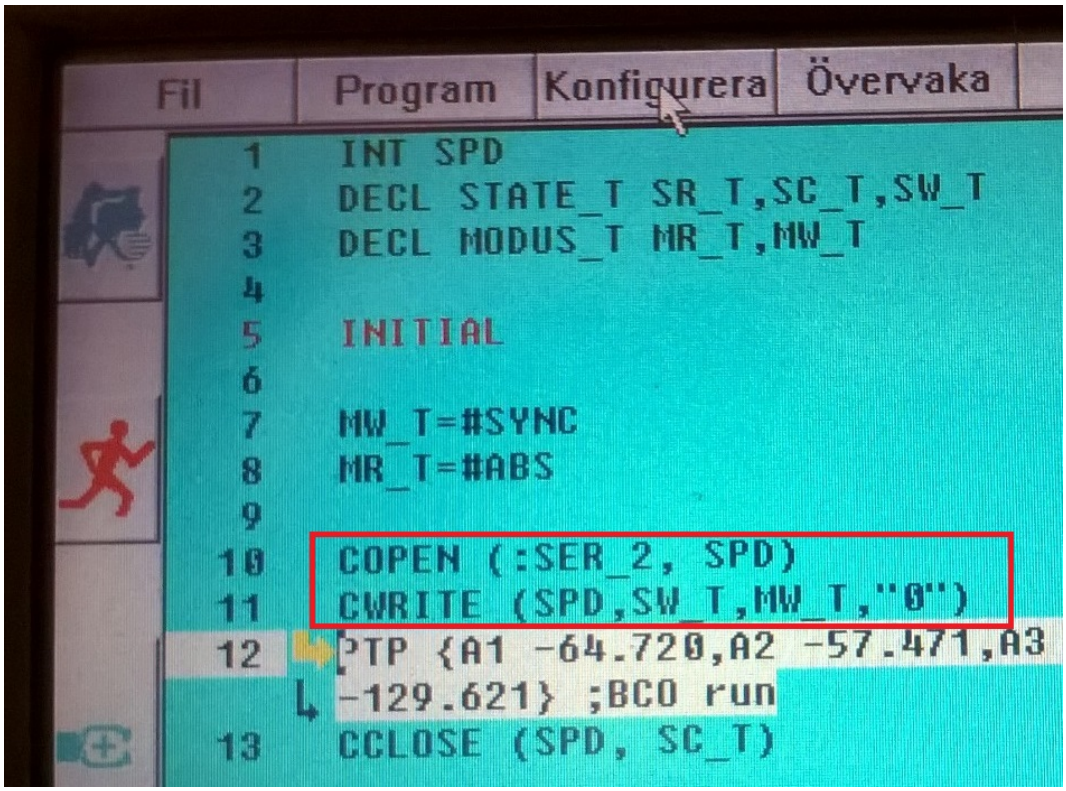


Figure 226: To stop KRC1.exe from sending data, run mystop.src.

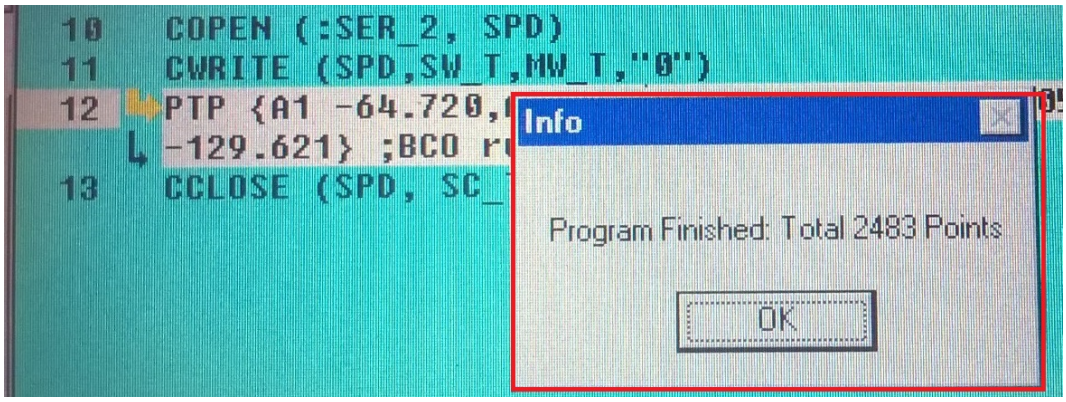


Figure 227: Info window after KRC1.exe has successfully sent all the data or is aborted by mstop.src.

Index