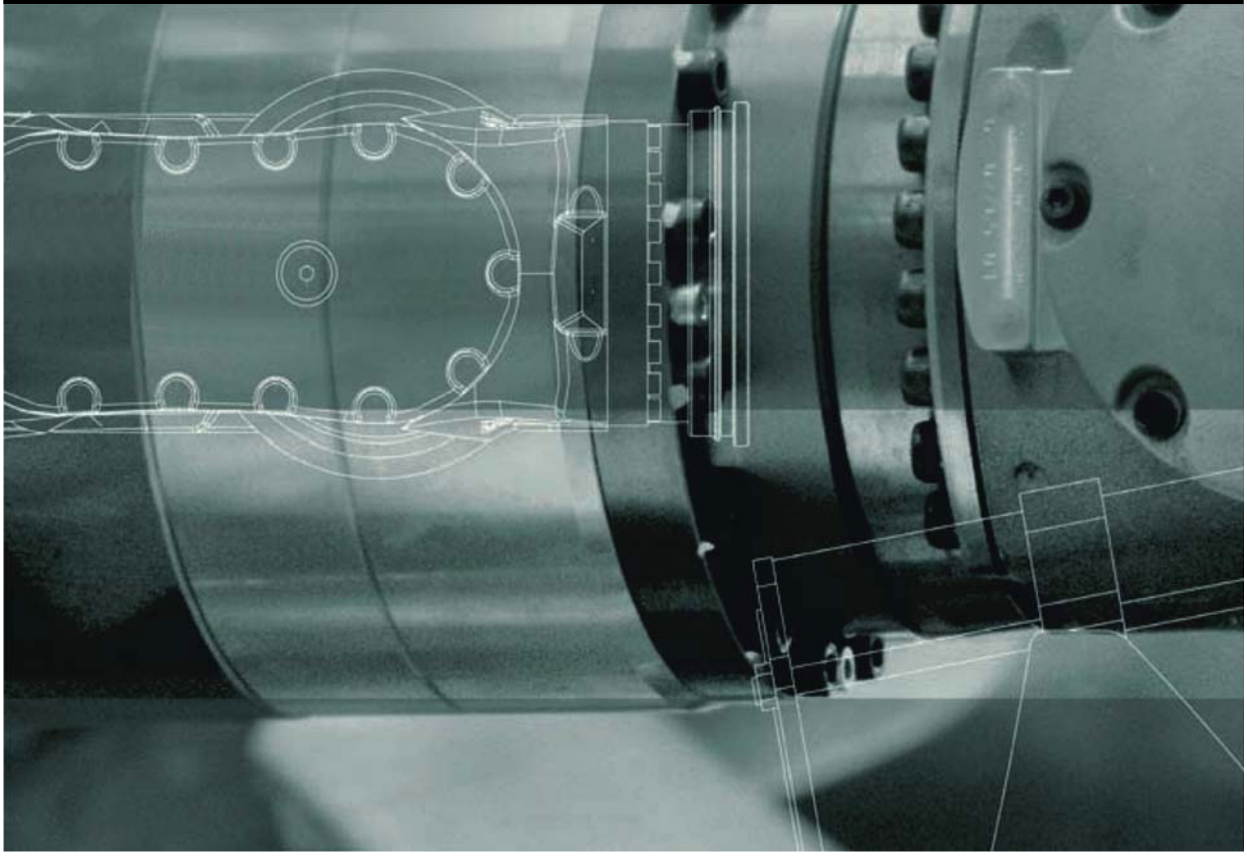
KUKA System Technology

KUKA Roboter GmbH

# KUKA.ExpertTech 3.2

## For KUKA System Software 8.2 and 8.3



Issued: 09.01.2013

Version: KST ExpertTech 3.2 V1 en (PDF)

KUKA Roboter GmbH
Zugspitzstraße 140
D-86165 Augsburg
Germany

Other functions not described in this documentation may be operable in the controller. The user has no claims to these functions, however, in the case of a replacement or service work.

We have checked the content of this documentation for conformity with the hardware and software described. Nevertheless, discrepancies cannot be precluded, for which reason we are not able to guarantee total conformity. The information in this documentation is checked on a regular basis, however, and necessary corrections will be incorporated in the subsequent edition.

Subject to technical alterations without an effect on the function.

Translation of the original documentation

KIM-PS5-DOC

Publication: Pub KST ExpertTech 3.2 (PDF) en
Bookstructure: KST ExpertTech 3.2 V1.1
Version: KST ExpertTech 3.2 V1 en (PDF)

# Contents

# 1 Introduction

## 1.1 Target group

This documentation is aimed at users with the following knowledge and skills:

- Advanced knowledge of the robot controller system
- Advanced KRL programming skills

> For optimal use of our products, we recommend that our customers take part in a course of training at KUKA College. Information about the training program can be found at www.kuka.com or can be obtained directly from our subsidiaries.

## 1.2 Industrial robot documentation

The industrial robot documentation consists of the following parts:

- Documentation for the manipulator
- Documentation for the robot controller
- Operating and programming instructions for the KUKA System Software
- Documentation relating to options and accessories
- Parts catalog on storage medium

Each of these sets of instructions is a separate document.

## 1.3 Representation of warnings and notes

**Safety**

These warnings are relevant to safety and **must** be observed.

> ⚠ DANGER These warnings mean that it is certain or highly probable that death or severe injuries **will** occur, if no precautions are taken.

> ⚠ WARNING These warnings mean that death or severe injuries **may** occur, if no precautions are taken.

> ⚠ CAUTION These warnings mean that minor injuries **may** occur, if no precautions are taken.

> NOTICE These warnings mean that damage to property **may** occur, if no precautions are taken.

> ⚠ These warnings contain references to safety-relevant information or general safety measures.
> These warnings do not refer to individual hazards or individual precautionary measures.

This warning draws attention to procedures which serve to prevent or remedy emergencies or malfunctions:

> SAFETY INSTRUCTIONS Procedures marked with this warning **must** be followed exactly.

**Notes**

These hints serve to make your work easier or contain references to further information.

ℹ Tip to make your work easier or reference to further information.

# 2 Product description

## 2.1 Overview of ExpertTech

**Functions**

ExpertTech is an add-on technology package and provides inline forms with the following functions:

- Activation and deactivation of collision detection
- Programming of PTP, LIN and CIRC motions with the approximate positioning option
- Programming of PTP_REL, LIN_REL and CIRC_REL motions with the approximate positioning option
- Programming of individual spline motions – SLIN and SCIRC – with the value assignment to system variables option
- Programming of spline segments – SPL, SLIN and SCIRC in spline blocks – with the value assignment to system variables option

**Properties**

- Motions can be inserted with a placeholder if the position of the end point is not yet known at the time of creating the program.
- Robot positions can be adopted in an inline form in one of the following ways:
  - Teaching the end point.
  - Teaching the end point as a variable.
  - Manually entering coordinates for the end point.

# 3 Safety

This documentation contains safety instructions which refer specifically to the software described here.

The fundamental safety information for the industrial robot can be found in the “Safety” chapter of the Operating and Programming Instructions for System Integrators or the Operating and Programming Instructions for End Users.

⚠ The “Safety” chapter in the operating and programming instructions must be observed. Death to persons, severe injuries or considerable damage to property may otherwise result.

# 4 Installation

## 4.1 System requirements

**Software**

- KUKA System Software 8.2 or 8.3

## 4.2 Installing or updating ExpertTech

> It is advisable to archive all relevant data before updating a software package.

**Preparation**

- Copy software from CD to KUKA USB stick.

  The software must be copied onto the stick with the file Setup.exe at the highest level (i.e. not in a folder).

> **NOTICE** Recommendation: Always use KUKA sticks. Data may be lost if sticks from other manufacturers are used.

**Precondition**

- “Expert” user group

**Procedure**

1. Connect the USB stick to the robot controller or smartPAD.
2. In the main menu, select **Start-up** > **Additional software**.
3. Press **New software**. The entry **ExpertTech** must be displayed in the **Name** column and drive **E:\** or **K:\** in the **Path** column.

   If not, press **Refresh**.
4. If the specified entries are now displayed, continue with step 5.

   If not, the drive from which the software is being installed must be configured first:

   - Press the **Configuration** button. A new window opens.
   - Select a line in the **Installation paths for options** area.

     **Note:** If the line already contains a path, this path will be overwritten.
   - Press **Path selection**. The available drives are displayed.
   - Select **E:\**. (If stick connected to the robot controller.)

     Or select **K:\**. (If stick connected to the smartPAD.)
   - Press **Save**. The window closes again.

   The drive only needs to be configured once and then remains saved for further installations.
5. Mark the entry **ExpertTech** and click on **Install**. Answer the request for confirmation with **Yes**.
6. Confirm the reboot prompt with **OK**.
7. Remove the stick.
8. Reboot the robot controller.

**LOG file**

A LOG file is created under C:\KRC\ROBOTER\LOG.

## 4.3 Uninstalling ExpertTech

> It is advisable to archive all relevant data before uninstalling a software package.

**Precondition**

- “Expert” user group

**Procedure**

1. In the main menu, select **Start-up** > **Additional software**. All additional programs installed are displayed.
2. Mark the entry **ExpertTech** and click on **Uninstall**. Reply to the request for confirmation with **Yes**. Uninstallation is prepared.
3. Reboot the robot controller. Uninstallation is resumed and completed.

**LOG file**

A LOG file is created under C:\KRC\ROBOTER\LOG.

# 5 Operation

## 5.1 Menus

The following menus and commands are specific to this technology package:

**Commands**

- **KRL Assistant**
  - CIRC_REL
  - CIRC
  - **Spline Commands**
    - SCIRC
    - SLIN
    - SPL
  - Collision Detection
  - PTP_REL
  - PTP
  - LIN_REL
  - LIN

# 6 Programming

## 6.1 Inline form for collision detection

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **Collision Detection**.

**Description**

This instruction switches collision detection on or off. The instruction refers to the next motion instruction in the program.

If collision detection is activated and the start point of the motion is approximated, the switching point for collision detection lies at the end of the approximation distance of the start point. This switching point may therefore come too late. In order to activate collision detection in time, the switching point can be delayed or brought forward in time.

- Start point is approximated: With **Delay=ON**, collision detection is activated approx. in the middle of the approximation distance of the start point.
- Start point is an exact positioning point: With **Delay=OFF**, collision detection is activated at the start point.

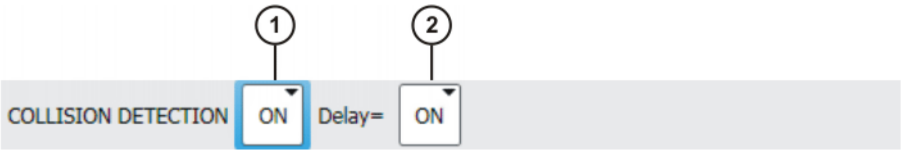


**Fig. 6-1: KRL assistant inline form “Collision detection”**

| Item | Description |
|---|---|
| 1 | Activate/deactivate collision detection.<br>■ **ON**<br>■ **OFF** |
| 2 | Shift the switching point toward the start of the motion in the case of an approximated start point.<br>■ **ON**: Use if the start point is an approximate positioning point.<br>■ **OFF**: Use if the start point is an exact positioning point. |

Further information about collision detection is contained in the “Operating and Programming Instructions for System Integrators”.

## 6.2 Inline form PTP

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **PTP**.

**Description**

Executes a point-to-point motion to the end point. The coordinates of the end point are absolute.

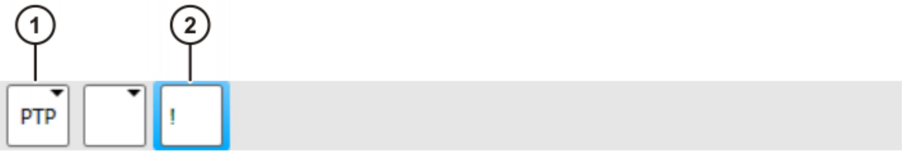


**Fig. 6-2: KRL assistant inline form “PTP”**

1 Motion type **PTP**

2 Placeholder for the end point

The following buttons are available:

| Button | Description |
|---|---|
| **{?}** | Opens a menu with the data types that are available for saving the end point.<br>Cartesian data types:<br>■ **E6POS**<br>■ **POS**<br>■ **FRAME**<br>The Cartesian coordinates refer to the BASE coordinate system.<br>Axis-specific data types:<br>■ **E6AXIS**<br>■ **AXIS**<br>Placeholder for the end point:<br>■ **!** |
| **PTP** | Opens a menu for switching the motion type.<br>■ **LIN**: LIN motion<br>■ **REL**: PTP_REL motion |
| **Cont** | Opens a menu for specifying the 1st approximate positioning parameter.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>■ **C_PTP**: The end point is approximated.<br>■ **Remove**: Deletes the approximation parameters.<br>The specification of the 1st approximate positioning parameter is sufficient for PTP-PTP approximate positioning. In the case of PTP-CP approximation, i.e. if the approximated PTP block is followed by a LIN or CIRC block, a 2nd approximate positioning parameter must also be specified. |

| Button | Description |
|---|---|
| **2nd Cont** | Opens a menu for specifying the 2nd approximate positioning parameter.<br>This button is only available if the 1st approximate positioning parameter has been defined.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>Precondition for C_DIS: 1st approximation parameter is C_PTP.<br>■ **C_VEL**: Velocity parameter: Approximation starts, at the earliest, when the velocity in the deceleration phase to the end point falls below the value of $APO.CVEL.<br>■ **C_ORI**: Orientation parameter: Approximation starts, at the earliest, when the dominant orientation angle falls below the value of $APO.CORI.<br>■ **Remove**: Deletes the approximation parameters. |
| **Touch Up** | Saves the current Cartesian position of the robot as the end point. |

## 6.3 Inline form PTP_REL

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **PTP_REL**.

**Description**

Executes a point-to-point motion to the end point. The coordinates of the end point are relative to the current position.

> A REL statement always refers to the current position of the robot. For this reason, if a REL motion is interrupted, the robot executes the entire REL motion again, starting from the position at which it was interrupted.

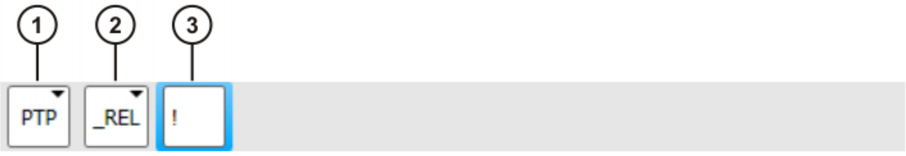


**Fig. 6-3: KRL assistant inline form “PTP_REL”**

1 Motion type **PTP**
2 Motion type **_REL**
3 Placeholder for the end point

The following buttons are available:

| Button | Description |
|---|---|
| **PTP** | Switches to PTP motion. |
| **Cont** | Opens a menu for specifying the 1st approximate positioning parameter.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>■ **C_PTP**: The end point is approximated.<br>■ **Remove**: Deletes the approximation parameters.<br>The specification of the 1st approximate positioning parameter is sufficient for PTP-PTP approximate positioning. In the case of PTP-CP approximation, i.e. if the approximated PTP block is followed by a LIN or CIRC block, a 2nd approximate positioning parameter must also be specified. |
| **2nd Cont** | Opens a menu for specifying the 2nd approximate positioning parameter.<br>This button is only available if the 1st approximate positioning parameter has been defined.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>Precondition for C_DIS: 1st approximation parameter is C_PTP.<br>■ **C_VEL**: Velocity parameter: Approximation starts, at the earliest, when the velocity in the deceleration phase to the end point falls below the value of $APO.CVEL.<br>■ **C_ORI**: Orientation parameter: Approximation starts, at the earliest, when the dominant orientation angle falls below the value of $APO.CORI.<br>■ **Remove**: Deletes the approximation parameters. |
| **Touch Up** | Do not teach a REL motion!<br>In the case of a REL motion, taught coordinates are not interpreted as the end point, but as the path to be traveled from the current robot position in a defined direction. For this reason, taught coordinates cannot generally be reached with a REL motion. |

## 6.4 Inline form LIN

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **LIN**.

**Description**

Executes a linear motion to the end point. The coordinates of the end point are absolute.

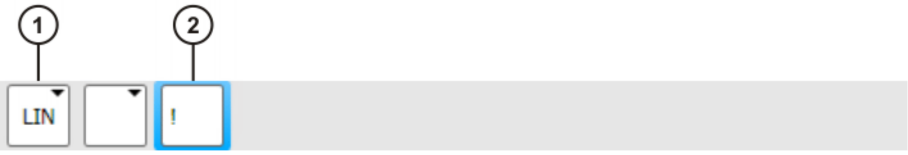


**Fig. 6-4: KRL assistant inline form “LIN”**

1 Motion type **LIN**

2 Placeholder for the end point

The following buttons are available:

| Button | Description |
|---|---|
| **{?}** | Opens a menu with the data types that are available for saving the end point.<br>Cartesian data types:<br>■ **E6POS**<br>■ **POS**<br>■ **FRAME**<br>The Cartesian coordinates refer to the BASE coordinate system.<br>Placeholder for the end point:<br>■ **!** |
| **LIN** | Opens a menu for switching the motion type.<br>■ **CIRC**: CIRC motion<br>■ **REL**: LIN_REL motion |
| **Cont** | Opens a menu for specifying the 1st approximate positioning parameter.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>■ **C_VEL**: Velocity parameter: Approximation starts, at the earliest, when the velocity in the deceleration phase to the end point falls below the value of $APO.CVEL.<br>■ **C_ORI**: Orientation parameter: Approximation starts, at the earliest, when the dominant orientation angle falls below the value of $APO.CORI.<br>■ **Remove**: Deletes the approximation parameters.<br>The specification of the 1st approximate positioning parameter is sufficient for CP-CP approximate positioning. In the case of CP-PTP approximation, i.e. if the approximated CP block is followed by a PTP block, a 2nd approximate positioning parameter must also be specified. |
| **2nd Cont** | Opens a menu for specifying the 2nd approximate positioning parameter.<br>This button is only available if the 1st approximate positioning parameter has been defined.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>Precondition for C_DIS: 1st approximation parameter is C_VEL or C_ORI.<br>■ **C_PTP**: The end point is approximated.<br>■ **Remove**: Deletes the approximation parameters. |
| **Touch Up** | Saves the current Cartesian position of the robot as the end point. |

## 6.5 Inline form LIN_REL

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **LIN_REL**.

**Description**

Executes a linear motion to the end point. The coordinates of the end point are relative to the current position.

> A REL statement always refers to the current position of the robot. For this reason, if a REL motion is interrupted, the robot executes the entire REL motion again, starting from the position at which it was interrupted.

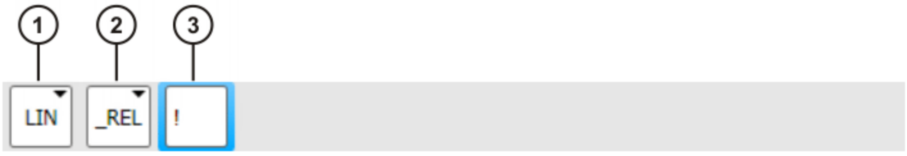


**Fig. 6-5: KRL assistant inline form “LIN_REL”**

1 Motion type **LIN**
2 Motion type **_REL**
3 Placeholder for the end point

The following buttons are available:

| Button | Description |
|---|---|
| **LIN** | Switches to LIN motion. |
| **Tool / Base** | Opens a menu for specifying the reference system for the end point coordinates.<br>■ **TOOL**: The coordinates refer to the TOOL coordinate system.<br>■ **BASE**: The coordinates refer to the BASE coordinate system. (Default) |
| **Cont** | Opens a menu for specifying the 1st approximate positioning parameter.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>■ **C_VEL**: Velocity parameter: Approximation starts, at the earliest, when the velocity in the deceleration phase to the end point falls below the value of $APO.CVEL.<br>■ **C_ORI**: Orientation parameter: Approximation starts, at the earliest, when the dominant orientation angle falls below the value of $APO.CORI.<br>■ **Remove**: Deletes the approximation parameters.<br>The specification of the 1st approximate positioning parameter is sufficient for CP-CP approximate positioning. In the case of CP-PTP approximation, i.e. if the approximated CP block is followed by a PTP block, a 2nd approximate positioning parameter must also be specified. |

| Button | Description |
|---|---|
| **2nd Cont** | Opens a menu for specifying the 2nd approximate positioning parameter.<br>This button is only available if the 1st approximate positioning parameter has been defined.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>Precondition for C_DIS: 1st approximation parameter is C_VEL or C_ORI.<br>■ **C_PTP**: The end point is approximated.<br>■ **Remove**: Deletes the approximation parameters. |
| **Touch Up** | Do not teach a REL motion!<br>In the case of a REL motion, taught coordinates are not interpreted as the end point, but as the path to be traveled from the current robot position in a defined direction. For this reason, taught coordinates cannot generally be reached with a REL motion. |

## 6.6 Inline form CIRC

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **CIRC**.

**Description**

Executes a circular motion. An auxiliary point and an end point must be specified in order for the controller to be able to calculate the circular motion.

The coordinates of the auxiliary point and end point are absolute. The auxiliary point cannot be approximated. The motion always stops exactly at this point.

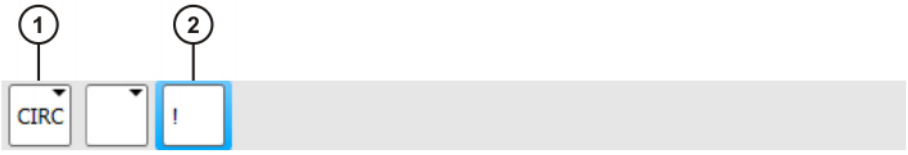


**Fig. 6-6: KRL assistant inline form “CIRC”**

1 Motion type **CIRC**

2 Placeholder for the auxiliary point

The following buttons are available:

| Button | Description |
|---|---|
| **{?}** | Opens a menu with the data types that are available for saving the end point.<br>Cartesian data types:<br>■ **E6POS**<br>■ **POS**<br>■ **FRAME**<br>The Cartesian coordinates refer to the BASE coordinate system.<br>Placeholder for the auxiliary or end point:<br>■ **!** |
| **CIRC** | Opens a menu for specifying the circular angle and switching the motion type.<br>■ **PTP**: PTP motion<br>■ **REL**: CIRC_REL motion<br>■ **CA**: The circular angle specifies the overall angle of the circular motion. This makes it possible to extend the motion beyond the programmed end point or to shorten it. The actual end point thus no longer corresponds to the programmed end point.<br>Unit: degrees. There is no limit; in particular, a circular angle greater than 360° can be programmed.<br>▪ Positive circular angle: the circular path is executed in the direction Start point › Auxiliary point › End point.<br>▪ Negative circular angle: the circular path is executed in the direction Start point › End point › Auxiliary point. |
| **Cont** | Opens a menu for specifying the 1st approximate positioning parameter.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>■ **C_VEL**: Velocity parameter: Approximation starts, at the earliest, when the velocity in the deceleration phase to the end point falls below the value of $APO.CVEL.<br>■ **C_ORI**: Orientation parameter: Approximation starts, at the earliest, when the dominant orientation angle falls below the value of $APO.CORI.<br>■ **Remove**: Deletes the approximation parameters.<br>The specification of the 1st approximate positioning parameter is sufficient for CP-CP approximate positioning. In the case of CP-PTP approximation, i.e. if the approximated CP block is followed by a PTP block, a 2nd approximate positioning parameter must also be specified. |

| Button | Description |
|---|---|
| **2nd Cont** | Opens a menu for specifying the 2nd approximate positioning parameter.<br>This button is only available if the 1st approximate positioning parameter has been defined.<br>■ **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>Precondition for C_DIS: 1st approximation parameter is C_VEL or C_ORI.<br>■ **C_PTP**: The end point is approximated.<br>■ **Remove**: Deletes the approximation parameters. |
| **Touch Up** | Saves the current Cartesian position of the robot as the auxiliary or end point. |

## 6.7 Inline form CIRC_REL

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **CIRC_REL**.

**Description**

Executes a circular motion. An auxiliary point and an end point must be specified in order for the controller to be able to calculate the circular motion.

The coordinates of the auxiliary point and end point are relative to the current position. The auxiliary point cannot be approximated. The motion always stops exactly at this point.

> A REL statement always refers to the current position of the robot. For this reason, if a REL motion is interrupted, the robot executes the entire REL motion again, starting from the position at which it was interrupted.

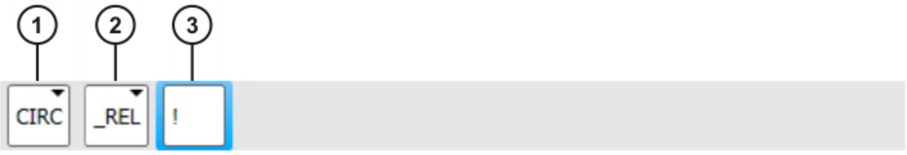


**Fig. 6-7: KRL assistant inline form “CIRC_REL”**

1 Motion type **CIRC**

2 Motion type **_REL**

3 Placeholder for the auxiliary point

The following buttons are available:

| Button | Description |
|---|---|
| **CIRC** | Opens a menu for specifying the circular angle and switching the motion type.<br>- **CIRC**: CIRC motion<br>- **CA**: The circular angle specifies the overall angle of the circular motion. This makes it possible to extend the motion beyond the programmed end point or to shorten it. The actual end point thus no longer corresponds to the programmed end point.<br>Unit: degrees. There is no limit; in particular, a circular angle greater than 360° can be programmed.<br>- Positive circular angle: the circular path is executed in the direction Start point › Auxiliary point › End point.<br>- Negative circular angle: the circular path is executed in the direction Start point › End point › Auxiliary point. |
| **Cont** | Opens a menu for specifying the 1st approximate positioning parameter.<br>- **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>- **C_VEL**: Velocity parameter: Approximation starts, at the earliest, when the velocity in the deceleration phase to the end point falls below the value of $APO.CVEL.<br>- **C_ORI**: Orientation parameter: Approximation starts, at the earliest, when the dominant orientation angle falls below the value of $APO.CORI.<br>- **Remove**: Deletes the approximation parameters.<br>The specification of the 1st approximate positioning parameter is sufficient for CP-CP approximate positioning. In the case of CP-PTP approximation, i.e. if the approximated CP block is followed by a PTP block, a 2nd approximate positioning parameter must also be specified. |
| **2nd Cont** | Opens a menu for specifying the 2nd approximate positioning parameter.<br>This button is only available if the 1st approximate positioning parameter has been defined.<br>- **C_DIS**: Distance parameter: Approximation starts, at the earliest, when the distance to the end point falls below the value of $APO.CDIS.<br>Precondition for C_DIS: 1st approximation parameter is C_VEL or C_ORI.<br>- **C_PTP**: The end point is approximated.<br>- **Remove**: Deletes the approximation parameters. |
| **Touch Up** | Do not teach a REL motion!<br>In the case of a REL motion, taught coordinates are not interpreted as the end point, but as the path to be traveled from the current robot position in a defined direction. For this reason, taught coordinates cannot generally be reached with a REL motion. |

## 6.8 Programming spline motions

Information about motion programming with spline is contained in the “Operating and Programming Instructions for System Integrators”.

### 6.8.1 Inline form SPL

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **Spline Commands** > **SPL**.

**Description**

SPL can only be programmed as a segment in a spline block. It is not possible to program an individual SPL motion.

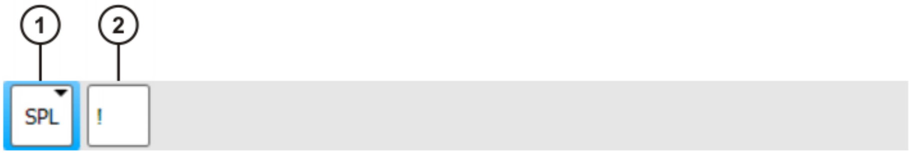


**Fig. 6-8: KRL assistant inline form “SPL”**

1 Motion type **SPL**

2 Placeholder for the end point

The following buttons are available:

| Button | Description |
|---|---|
| **{?}** | Opens a menu with the data types that are available for saving the end point.<br>Cartesian data types:<br>- **E6POS**<br>- **POS**<br>- **FRAME**<br>The Cartesian coordinates refer to the BASE coordinate system.<br>Placeholder for the end point:<br>- **!** |
| **SLIN/SCIRC** | Switches to SLIN motion. |
| **Remove** | Deletes the selected system variable. |
| **AddParam** | Opens a menu for adding a system variable. A default value is assigned to the system variable. This value can be edited.<br>- **VEL**: $VEL for the path velocity<br>- **ACC**: $ACC for the path acceleration<br>- **ORI**: $ORI_TYPE for the orientation control<br>- **JERK**: $JERK for the jerk limitation<br>- **Remove**: Deletes the selected system variable. |
| **Touch Up** | Saves the current Cartesian position of the robot as the end point. |

The value assignment to a system variable only applies for the corresponding spline segment. It has no effect on subsequent segments in a spline block.

Detailed information about the system variables can be found in the documentation **System Variables**.

### 6.8.2 Inline form SLIN

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **Spline Commands** > **SLIN**.

**Description**

SLIN can be programmed as an individual block or as a segment in a spline block.

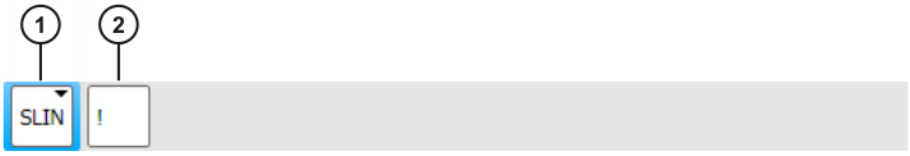


**Fig. 6-9: KRL assistant inline form “SLIN”**

1 Motion type **SLIN**

2 Placeholder for the end point

The following buttons are available:

| Button | Description |
|---|---|
| **{?}** | Opens a menu with the data types that are available for saving the end point.<br>Cartesian data types:<br>- **E6POS**<br>- **POS**<br>- **FRAME**<br>The Cartesian coordinates refer to the BASE coordinate system.<br>Placeholder for the end point:<br>- **!** |
| **SCIRC/SPL** | Switches to SCIRC motion. |
| **Remove** | Deletes the selected system variable. |
| **AddParam** | Opens a menu for adding a system variable. A default value is assigned to the system variable. This value can be edited.<br>- **VEL**: $VEL for the path velocity<br>- **ACC**: $ACC for the path acceleration<br>- **ORI**: $ORI_TYPE for the orientation control<br>- **JERK**: $JERK for the jerk limitation<br>- **Remove**: Deletes the selected system variable. |
| **Touch Up** | Saves the current Cartesian position of the robot as the end point. |

The value assignment to a system variable only applies for the corresponding spline segment. It has no effect on subsequent segments in a spline block.

Detailed information about the system variables can be found in the documentation **System Variables**.

### 6.8.3 Inline form SCIRC

**Call**

- Select the menu sequence **Commands** > **KRL assistant** > **Spline Commands** > **SCIRC**.

**Description**

SCIRC can be programmed as an individual block or as a segment in a spline block.

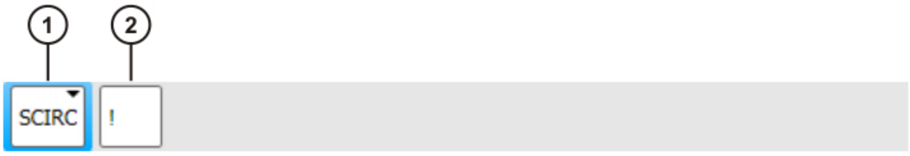


**Fig. 6-10: KRL assistant inline form “SCIRC”**

1 Motion type **SCIRC**

2 Placeholder for the auxiliary point

The following buttons are available:

| Button | Description |
|---|---|
| **{?}** | Opens a menu with the data types that are available for saving the end point.<br>Cartesian data types:<br>- **E6POS**<br>- **POS**<br>- **FRAME**<br>The Cartesian coordinates refer to the BASE coordinate system.<br>Placeholder for the auxiliary or end point:<br>- **!** |
| **SPL/SLIN** | Switches to SPL motion. |
| **Remove** | Deletes the selected system variable. |
| **AddParam** | Opens a menu for adding a system variable. A default value is assigned to the system variable. This value can be edited.<br>- **VEL**: $VEL for the path velocity<br>- **ACC**: $ACC for the path acceleration<br>- **ORI**: $ORI_TYPE for the orientation control<br>- **CIRC Type**: $CIRC_TYPE for the orientation control of the circle<br>- **JERK**: $JERK for the jerk limitation<br>- **Remove**: Deletes the selected system variable. |
| **Touch Up** | Saves the current Cartesian position of the robot as the auxiliary or end point. |

The value assignment to a system variable only applies for the corresponding spline segment. It has no effect on subsequent segments in a spline block.

> Detailed information about the system variables can be found in the documentation **System Variables**.

## 6.9 Motion with placeholders

If the position of one or more end points is not yet known when a KRL program is created, the motions can be inserted using placeholders (!).

During subsequent program execution, the program stops at the lines with placeholders (!) and the robot positions can be adopted.

## 6.10 Adopting the robot position

Robot positions can be adopted in the inline form in one of the following ways:

- Teaching the end point.
  (>>> 6.10.1 "Teaching the end point" Page 28)
- Teaching the end point as a variable.
  (>>> 6.10.2 "Teaching the end point as a variable" Page 28)
- Manually entering coordinates for the end point.
  (>>> 6.10.3 "Entering coordinates for the end point" Page 29)

### 6.10.1 Teaching the end point

**Precondition**

- A program is selected.
- Operating mode T1 or T2

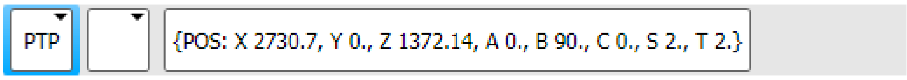
> The tool and base must be defined before robot positions can be adopted in an inline form. This is achieved, for example, by executing the INI line of the program.

> For **Touch Up**, the focus must be on the box for the end point.

**Procedure**

1. Move the TCP to the position that is to be taught as the end point.
2. Press **Touch Up** to accept the current position of the TCP as the end point and reply to the request for confirmation with **Yes**.
3. Save instruction with **Cmd Ok**.

**Example**



PTP

{POS: X 2730.7, Y 0., Z 1372.14, A 0., B 90., C 0., S 2., T 2.}

Fig. 6-11: KRL assistant inline form “PTP” (end point of type “Pos”)

### 6.10.2 Teaching the end point as a variable

**Precondition**

- A program is selected.
- Operating mode T1 or T2

> The tool and base must be defined before robot positions can be adopted in an inline form. This is achieved, for example, by executing the INI line of the program.

ℹ For **Touch Up**, the focus must be on the box for the end point.

**Procedure**

1. Move the TCP to the position that is to be adopted as the end point.
2. Enter a name for the variable.

ℹ The use of keywords reserved for KRL is not permitted. Further information about keywords is contained in the “Operating and Programming Instructions for System Integrators”.

3. Press **{?}**. A menu for selecting the data type opens. The available data types depend on the motion type.
4. Select a data type to accept the current position of the TCP as the end point and reply to the request for confirmation with **Yes**.
5. Save instruction with **Cmd Ok**.

### 6.10.3 Entering coordinates for the end point

**Precondition**

- A program is selected.
- Operating mode T1 or T2

ℹ The tool and base must be defined before robot positions can be adopted in an inline form. This is achieved, for example, by executing the INI line of the program.

ℹ For **Touch Up**, the focus must be on the box for the end point.

**Procedure**

1. Press **{?}**. A menu for selecting the data type opens. The available data types depend on the motion type.
2. Select a data type to accept the current position of the TCP as the end point.
3. Change the position data in Edit mode.
4. Save instruction with **Cmd Ok**.

## 6.11 Data types for motion programming

The following data types for motion programming are predefined:

**Structure type AXIS**

A1 to A6 are angle values (rotational axes) or translation values (translational axes) for the axis-specific movement of robot axes 1 to 6.

```
STRUC AXIS REAL A1, A2, A3, A4, A5, A6
```

**Structure type E6AXIS**

E1 to E6 are angle values or translation values of the external axes 7 to 12.

```
STRUC E6AXIS REAL A1, A2, A3, A4, A5, A6, E1, E2, E3, E4, E5, E6
```

**Structure type FRAME**

X, Y and Z are space coordinates, while A, B and C are the orientation of the coordinate system.

```
STRUC FRAME REAL X, Y, Z, A, B, C
```

**Structure types POS and E6POS**

S (Status) and T (Turn) define axis positions unambiguously.

```
STRUC POS REAL X, Y, Z, A, B, C, INT S, T
```

```
STRUC E6POS REAL X, Y, Z, A, B, C, E1, E2, E3, E4, E5, E6, INT S, T
```

# 7 Messages

Operating or programming errors may result in error messages.

| Message | Cause | Remedy |
|---|---|---|
| Invalid approximation parameter! | The selected combination of approximation parameters is invalid. | Change the approximation parameters. |
| TouchUp failed. | The point could not be taught. | --- |
| Invalid value found. | Invalid value in the inline form | Enter a valid value. |
| Value for {0} invalid! | No value assignment to system variable (box is empty) | Assign a valid value to the system variable. |
| Variable could not be created. Target type unknown. | The **Touch Up** command was executed on a non-geometric expression. | Assign the correct data type to the variable on which the **Touch Up** command is to be executed:<br>■ PTP motion: FRAME, POS or E6POS<br>■ CP motion: AXIS, E6AXIS, FRAME, POS or E6POS |
| Insert first approximation parameter! | The first approximate positioning parameter has not been entered correctly. | Enter the correct first approximate positioning parameter. |
| Value assignment of variable {0} failed. | Incorrect data type | Declare the variable and assign the correct data type. |
| Change of structure type from {0} to {1} not possible. | Data types of the displayed variables are not compatible, e.g. POS<>AXIS. | Adapt the target and source type. |
| Base coordinate system not defined. | $BASE is not defined. | Define $BASE. |
| Tool coordinate system not defined. | $TOOL is not defined. | Define $TOOL. |
| KRL-Assistant does not support this command. | The **Change** command is not supported. | --- |
| Unknown value {0} | The variable is not declared. | Declare the variable. |
| Selected parameter can't be removed! | The selected parameter cannot be removed by pressing the **Remove** button. | --- |
| Function not allowed for WITH- parameter! | The function called by the button is not supported. | --- |
| Command ambiguous, TouchUp not possible. | The focus is on the wrong box. No point could be taught. | Place the focus on the box for the auxiliary or end point and press **Touch Up**. |
| Variable could not be created. Target type unknown. | The **Touch Up** command was executed on a non-geometric expression. | Assign the correct data type to the variable on which the **Touch Up** command is to be executed:<br>■ PTP motion: FRAME, POS or E6POS<br>■ CP motion: AXIS, E6AXIS, FRAME, POS or E6POS |

# 8 KUKA Service

## 8.1 Requesting support

**Introduction**

The KUKA Roboter GmbH documentation offers information on operation and provides assistance with troubleshooting. For further assistance, please contact your local KUKA subsidiary.

**Information**

The following information is required for processing a support request:

- Model and serial number of the robot
- Model and serial number of the controller
- Model and serial number of the linear unit (if applicable)
- Model and serial number of the energy supply system (if applicable)
- Version of the KUKA System Software
- Optional software or modifications
- Archive of the software

  For KUKA System Software V8: instead of a conventional archive, generate the special data package for fault analysis (via **KrcDiag**).
- Application used
- Any external axes used
- Description of the problem, duration and frequency of the fault

## 8.2 KUKA Customer Support

**Availability**

KUKA Customer Support is available in many countries. Please do not hesitate to contact us if you have any questions.

**Argentina**

Ruben Costantini S.A. (Agency)
Luis Angel Huergo 13 20
Parque Industrial
2400 San Francisco (CBA)
Argentina
Tel. +54 3564 421033
Fax +54 3564 428877
ventas@costantini-sa.com

**Australia**

Headland Machinery Pty. Ltd.
Victoria (Head Office & Showroom)
95 Highbury Road
Burwood
Victoria 31 25
Australia
Tel. +61 3 9244-3500
Fax +61 3 9244-3501
vic@headland.com.au
www.headland.com.au

**Belgium**

KUKA Automatisering + Robots N.V.
Centrum Zuid 1031
3530 Houthalen
Belgium
Tel. +32 11 516160
Fax +32 11 526794
info@kuka.be
www.kuka.be

**Brazil**

KUKA Roboter do Brasil Ltda.
Travessa Claudio Armando, nº 171
Bloco 5 - Galpões 51/52
Bairro Assunção
CEP 09861-7630 São Bernardo do Campo - SP
Brazil
Tel. +55 11 4942-8299
Fax +55 11 2201-7883
info@kuka-roboter.com.br
www.kuka-roboter.com.br

**Chile**

Robotec S.A. (Agency)
Santiago de Chile
Chile
Tel. +56 2 331-5951
Fax +56 2 331-5952
robotec@robotec.cl
www.robotec.cl

**China**

KUKA Robotics China Co.,Ltd.
Songjiang Industrial Zone
No. 388 Minshen Road
201612 Shanghai
China
Tel. +86 21 6787-1888
Fax +86 21 6787-1803
www.kuka-robotics.cn

**Germany**

KUKA Roboter GmbH
Zugspitzstr. 140
86165 Augsburg
Germany
Tel. +49 821 797-4000
Fax +49 821 797-1616
info@kuka-roboter.de
www.kuka-roboter.de

**France**

KUKA Automatisme + Robotique SAS
Techvallée
6, Avenue du Parc
91140 Villebon S/Yvette
France
Tel. +33 1 6931660-0
Fax +33 1 6931660-1
commercial@kuka.fr
www.kuka.fr

**India**

KUKA Robotics India Pvt. Ltd.
Office Number-7, German Centre,
Level 12, Building No. - 9B
DLF Cyber City Phase III
122 002 Gurgaon
Haryana
India
Tel. +91 124 4635774
Fax +91 124 4635773
info@kuka.in
www.kuka.in

**Italy**

KUKA Roboter Italia S.p.A.
Via Pavia 9/a - int.6
10098 Rivoli (TO)
Italy
Tel. +39 011 959-5013
Fax +39 011 959-5141
kuka@kuka.it
www.kuka.it

**Japan**

KUKA Robotics Japan K.K.
YBP Technical Center
134 Godo-cho, Hodogaya-ku
Yokohama, Kanagawa
240 0005
Japan
Tel. +81 45 744 7691
Fax +81 45 744 7696
info@kuka.co.jp

**Canada**

KUKA Robotics Canada Ltd.
6710 Maritz Drive - Unit 4
Mississauga
L5W 0A1
Ontario
Canada
Tel. +1 905 670-8600
Fax +1 905 670-8604
info@kukarobotics.com
www.kuka-robotics.com/canada

**Korea**

KUKA Robotics Korea Co. Ltd.
RIT Center 306, Gyeonggi Technopark
1271-11 Sa 3-dong, Sangnok-gu
Ansan City, Gyeonggi Do
426-901
Korea
Tel. +82 31 501-1451
Fax +82 31 501-1461
info@kukakorea.com

**Malaysia**

KUKA Robot Automation Sdn Bhd
South East Asia Regional Office
No. 24, Jalan TPP 1/10
Taman Industri Puchong
47100 Puchong
Selangor
Malaysia
Tel. +60 3 8061-0613 or -0614
Fax +60 3 8061-7386
info@kuka.com.my

**Mexico**

KUKA de México S. de R.L. de C.V.
Progreso #8
Col. Centro Industrial Puente de Vigas
Tlalnepantla de Baz
54020 Estado de México
Mexico
Tel. +52 55 5203-8407
Fax +52 55 5203-8148
info@kuka.com.mx
www.kuka-robotics.com/mexico

**Norway**

KUKA Sveiseanlegg + Roboter
Sentrumsvegen 5
2867 Hov
Norway
Tel. +47 61 18 91 30
Fax +47 61 18 62 00
info@kuka.no

**Austria**

KUKA Roboter Austria GmbH
Vertriebsbüro Österreich
Regensburger Strasse 9/1
4020 Linz
Austria
Tel. +43 732 784752
Fax +43 732 793880
office@kuka-roboter.at
www.kuka-roboter.at

**Poland**

KUKA Roboter Austria GmbH
Spółka z ograniczoną odpowiedzialnością
Oddział w Polsce
Ul. Porcelanowa 10
40-246 Katowice
Poland
Tel. +48 327 30 32 13 or -14
Fax +48 327 30 32 26
ServicePL@kuka-roboter.de

**Portugal**

KUKA Sistemas de Automatización S.A.
Rua do Alto da Guerra n° 50
Armazém 04
2910 011 Setúbal
Portugal
Tel. +351 265 729780
Fax +351 265 729782
kuka@mail.telepac.pt

**Russia**

OOO KUKA Robotics Rus
Webnaja ul. 8A
107143 Moskau
Russia
Tel. +7 495 781-31-20
Fax +7 495 781-31-19
kuka-robotics.ru

**Sweden**

KUKA Svetsanläggningar + Robotar AB
A. Odhners gata 15
421 30 Västra Frölunda
Sweden
Tel. +46 31 7266-200
Fax +46 31 7266-201
info@kuka.se

**Switzerland**

KUKA Roboter Schweiz AG
Industriestr. 9
5432 Neuenhof
Switzerland
Tel. +41 44 74490-90
Fax +41 44 74490-91
info@kuka-roboter.ch
www.kuka-roboter.ch

**Spain**

KUKA Robots IBÉRICA, S.A.
Pol. Industrial
Torrent de la Pastera
Carrer del Bages s/n
08800 Vilanova i la Geltrú (Barcelona)
Spain
Tel. +34 93 8142-353
Fax +34 93 8142-950
Comercial@kuka-e.com
www.kuka-e.com

**South Africa**

Jendamark Automation LTD (Agency)
76a York Road
North End
6000 Port Elizabeth
South Africa
Tel. +27 41 391 4700
Fax +27 41 373 3869
www.jendamark.co.za

**Taiwan**

KUKA Robot Automation Taiwan Co., Ltd.
No. 249 Pujong Road
Jungli City, Taoyuan County 320
Taiwan, R. O. C.
Tel. +886 3 4331988
Fax +886 3 4331948
info@kuka.com.tw
www.kuka.com.tw

**Thailand**

KUKA Robot Automation (M)SdnBhd
Thailand Office
c/o Maccall System Co. Ltd.
49/9-10 Soi Kingkaew 30 Kingkaew Road
Tt. Rachatheva, A. Bangpli
Samutprakarn
10540 Thailand
Tel. +66 2 7502737
Fax +66 2 6612355
atika@ji-net.com
www.kuka-roboter.de

**Czech Republic**

KUKA Roboter Austria GmbH
Organisation Tschechien und Slowakei
Sezemická 2757/2
193 00 Praha
Horní Počernice
Czech Republic
Tel. +420 22 62 12 27 2
Fax +420 22 62 12 27 0
support@kuka.cz

**Hungary**

KUKA Robotics Hungaria Kft.
Fö út 140
2335 Taksony
Hungary
Tel. +36 24 501609
Fax +36 24 477031
info@kuka-robotics.hu

**USA**

KUKA Robotics Corporation
51870 Shelby Parkway
Shelby Township
48315-1787
Michigan
USA
Tel. +1 866 873-5852
Fax +1 866 329-5852
info@kukarobotics.com
www.kukarobotics.com

**UK**

KUKA Automation + Robotics
Hereward Rise
Halesowen
B62 8AN
UK
Tel. +44 121 585-0800
Fax +44 121 585-0900
sales@kuka.co.uk

# Index