Experiments Guide

Experiment 01

Motion Programming Inline Form

# Aim of the experiment

* Test Program execution
* Understand PTP motion
* Get Familiar with program creation and editing in "User" interface

# Preconditions

* Knowledge of how to use Navigator.
* Knowledge of operating modes.
* Theoretical knowledge in motion programming (PTP type)
* Tool and base coordinate system calibration.

# Introduction

## User Programming

Inline forms are available in the KSS for frequently used instruction. They simplify programming and facilitates user interface with controller without the need of knowing detail information about KUKA programming Language

# Explaining Program structure

Program structure previews a simple KRL syntax. The DEF line indicates the name of the program; this can be hidden or displayed. Declaration section after DEF-line, where variables and their data types declared.

INI - line contains the internal variables and parameters.

Mind that PTP command motion is the first command in any KRL program to be fully defined. The "HOME" position is not a program specific. It is used as the first and last position. The HOME position is stored with following values in the robot controller:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A6 | A5 | A4 | A3 | A2 | A1 | Axis |
| 0 | **0** | **0** | **90** | **-90** | **0** | **Value** |

# Procedure

While in the user interface, make sure to choose the right directory KRC:\R1\.click "New." to create a new module.

Press "Select." to execute the program automatically.

# Setting the program Override

Program override is the velocity of the robot during Program execution. This value is specified as a percentage of the programmed velocity. Realize that in T1 mode, the maximum velocity is 250 mm/s. To modify the program override, touch the POV/HOV status indicator and slide the bar to the required value.

# Starting Program forward (manual)

After selecting the program, and the operating mode. Hold the enabling switch down and wait until the status bar indicates "Drives ready".

Experiment 02

Motion programming inline form

# Aim of the experiment

1. Understand PTP motion
2. Get Familiar with program creation and editing in "User" interface.

# Preconditions

1. Familiar with Experiment 01
2. Theoretical knowledge in motion programming (PTP type)

# Introduction

## Point to point motion type (PTP)

Robot guide TCP from the current position along the fastest path to the end point specified.

The first motion in the program must be PTP; Status and turns only defined in PTP command and ignored in CP motion.

# "Teaching" program description

"teaching" program is an application of online robot programming using teach pendant. The robot can move in PTP motion type to the point specified by the user. The end point assigning can be applied in the different operating modes, either axis-specific or Cartesian.

# Procedure

While in the user interface, make sure to choose the right directory KRC:\R1\.Select "New." to create a new module.

Press "Select." to execute the program automatically.

Add new motion command by pressing "Motion.". The inline form provides several settings. First, choose which kind of movement required; In this case, "PTP." is chosen.

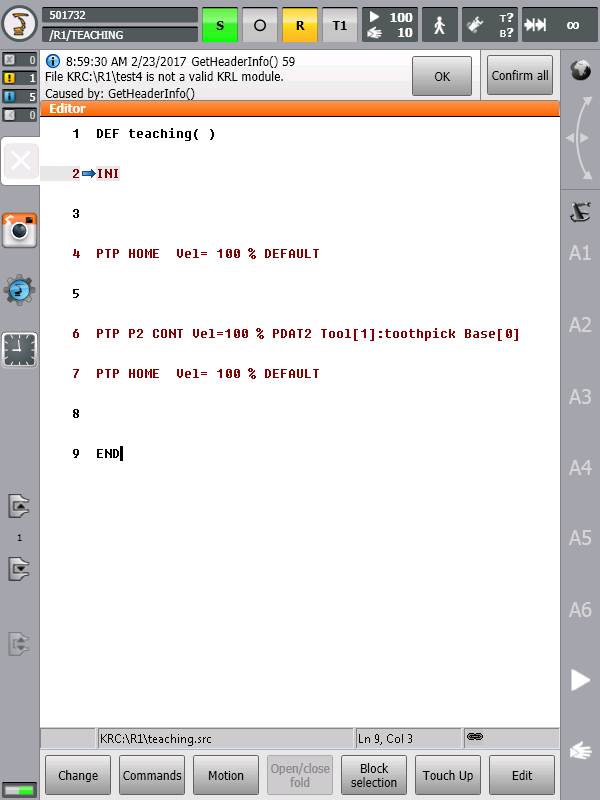
Next field specifies the position you want the robot to move. As a default, the first position is called "P1" and the index increments every time creating a new one. Names can be overwritten; in the program, the position called "P2."

# Explaining Program structure

As stated in "Experiment 01", the program should start by "PTP HOME ... " and ends with the same command to ensure that all the information needed for the robot to move is fully defined.

The next motion command is specified as PTP. Selecting "CONT." means that the end point is approximated this helps in executing next command early. In the last field, there is the name of motion data settings. It is created automatically so no need to change anything.

After setting all parameters, Robot moved to the desired position. By pressing "touch up" button position is saved.



Experiment 03

Motion programming (Expert level)

# Aim of the experiment

1. Understand and perform basic programs using KRL
2. Get Familiar with program creation and editing in "Expert" interface

# Preconditions

1. Passing Experiment 02.
2. KRL basic knowledge.
3. Motion programming theoretical knowledge.
4. Tool and base coordinate system calibration.

# Introduction

## Expert group

In the Expert interface, can achieve advanced programming using the KRL programming language and perform complex application programs including subprograms, interrupt programming, loops, and program branches.

# "test 3" Program Description

This program allows moving the robot to the HOME position using KRL commands. The program is similar to the one applied in "Experiment 01".

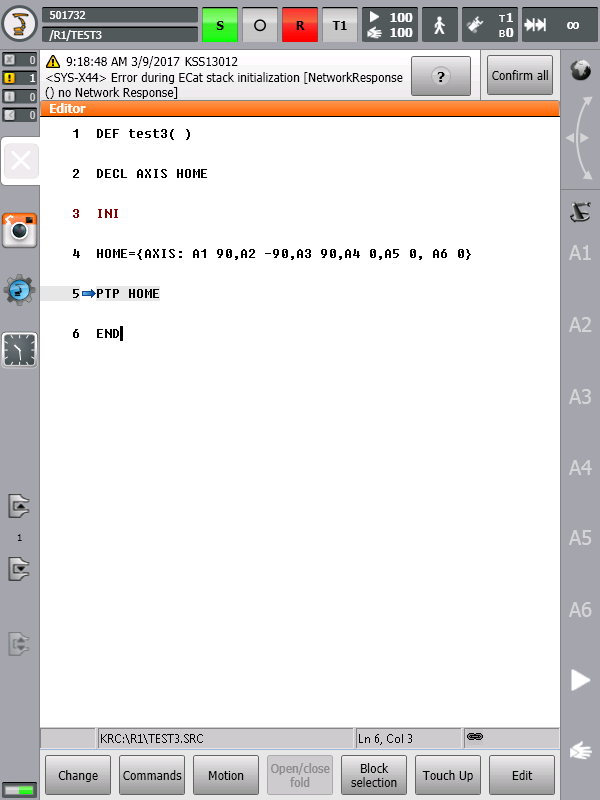
# Procedure

While in the Expert interface, make sure to choose the right directory KRC:\R1\.Click "New." to create a new module.

* 1. In the Declaration section, HOME declared as variable, Axis-specific data type > DECL AXIS HOME
  2. In the initialization section, HOME assigned be the required values > HOME={AXIS: A1 90, A2 -90, A3 90, A4 0, A5 0, A6 0}
  3. The motion type selected is PTP > PTP HOME

User Manual provides the procedure for changing user group to Expert, in operation section > change user group.

 User Manual provides the procedure for changing user group to Expert, in operation section > change user group.



Experiment 04

Motion programming (Expert level)

LIN motion

# Aim of the experiment

# Understand and perform basic programs using KRL

# Get Familiar with program creation and editing in "Expert" interface

# Understand and implement LIN motion command

# Preconditions

# Familiar with Experiment 03

# KRL basic knowledge.

# Motion programming theoretical knowledge.

# Tool and base coordinate system calibration.

# Introduction

## LIN motion

In the case of linear motion, the KRC "KUKA Robot Controller" calculates a straight line from the current position (the last point programmed in the program) to the point specified by the motion command. Linear motion is programmed using the keywords LIN or LIN\_REL in connection with the specification of the endpoint.

LIN motion categorized as a continuous path.

# "test 10" Program Description

The program moves the robot from current position to a point specified in the Cartesian way. The path generated is linear using the LIN motion type. The program is written in KRL using the Expert user.

**BAS (#tool 01)**

**BAS (#base 01)**

**; Initialization of TOOL and BASE coordinate system**

**PTP HOME Vel=100 % PDAT1**

**; PTP motion (in line command) to the system variable "HOME"**

**$APO.CDIS = 20**

**$APO.CORI = 20**

**$APO.CVEL = 0.2**

**; System variables of end point approximations**

**PTP HOME**

**LIN{X 20, Y 0, Z 0, A 180, B 0, C 0} C\_VEL**

**; LIN motion instruction with activation of velocity approximation**

# Procedure

1. While in the Expert interface, make sure to choose the right directory KRC:\R1\.Click "New." to create a new module.
2. In the initialization section, define the base and tool coordinate system for the robot.

* BAS( #tool 01)
* BAS( #base 01)

Another way for initialization is to use system variables

* $TOOL = TOOL\_DATA[1]
* $BASE = BASE\_DATA[1]

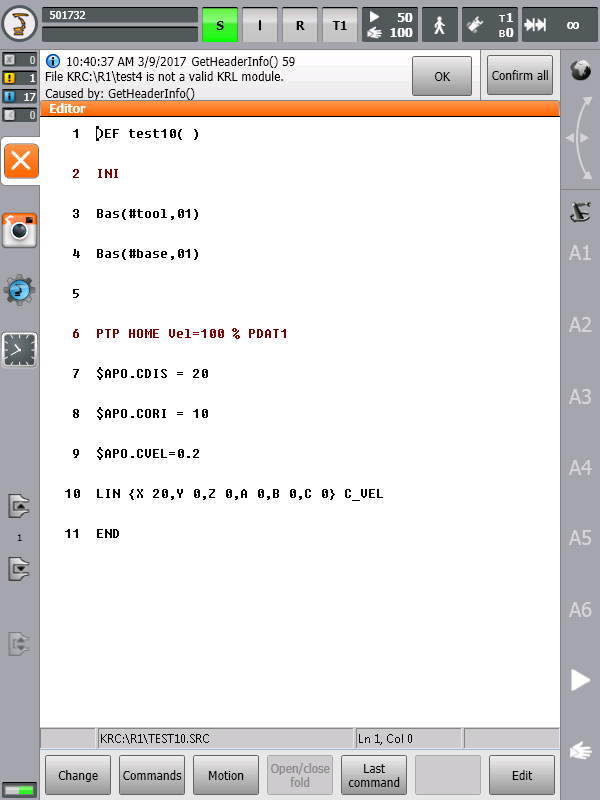
1. Start to assign values to end point specifications: approximation distance, approximation orientation, and approximation velocity. These specifications can be adjusted using system variables: $APO.CDIS, $APO.CORI, and $APO.CVEL.
2. Add PTP command to **HOME** position. This is necessary in every program.
3. To generate the linear path, type LIN instruction and activate approximate positioning by inserting one of the keywords: C\_DIS, C\_ORI, and C\_VEL. at the end of the command line

# Empirical Information

In KRL program, the first motion instruction must declare a firm initial situation. Bear in mind to start any motion program with a PTP motion. PTP motion can be defined in Cartesian and axis-specific coordinates unlike CP motions (LIN and CIRC).

HOME position is specified in axis-specific coordinate. In this program, the PTP motion written as inline form.

 For further information about the system variables used in Experiment 04, check User Manual in KRL-guide section > System Variables .



Experiment 05

Motion programming (Expert level)

PTP-POSE motion

# Aim of the experiment

* Understand and perform basic programs using KRL
* Get Familiar with program creation and editing in "Expert" interface
* Understand and implement PTP motion command in Cartesian coordinate
* Understand "Status and turns" concept

# Preconditions

* Passing Experiment 04
* KRL basic knowledge.
* Motion programming theoretical knowledge.
* Tool and base coordinate system calibration.

# Introduction

## Status and Turns

Moving the robot into a point can produce different axis positions for the same TCP (tool center point).

Referring to KRL -guide > structure type in the user manual, The entries “S” and “T”-status and turns- in a POS structure type are used to define an unambiguous position; For this reason, it is important to start any program instruction by defining the status and turn. Since "S" and "T" not taken into consideration in the CP- motion, the first line must be complete PTP structure.

**Status and Turn both require integer entries, which should be made in binary form.**

# "PTP" Program Description

In this program, the first motion instruction is not the default HOME position- AXIS datatype variable – but instead the statement is using complete PTP instruction of type POS.

The program moves the robot to the point assigned to the HOME variable. The first motion is to the origin of base and tool system specified. Then the robot moves in linear and absolute shifts in X-axis and y –axis.

**DECL POS HOME**

**; Declaration of HOME variable, POS structure data type.**

**BAS (#tool 01)**

**BAS (#base 01)**

**; Initialization of TOOL and BASE coordinate system**

**HOME = {POS: X 0, Y 0, Z 0, A 180, B 0, C 0, S 'B010', T 'B0011'}**

**PTP HOME**

**; PTP motion to the system variable "HOME"**

**HOME = {POS: X 0, Y 200, Z 0, A 180, B 0, C 0}**

**PTP HOME**

**HOME = {POS: X 200}**

**PTP HOME**

**HOME = {POS: X 20}**

**PTP HOME**

**END**

# Procedure

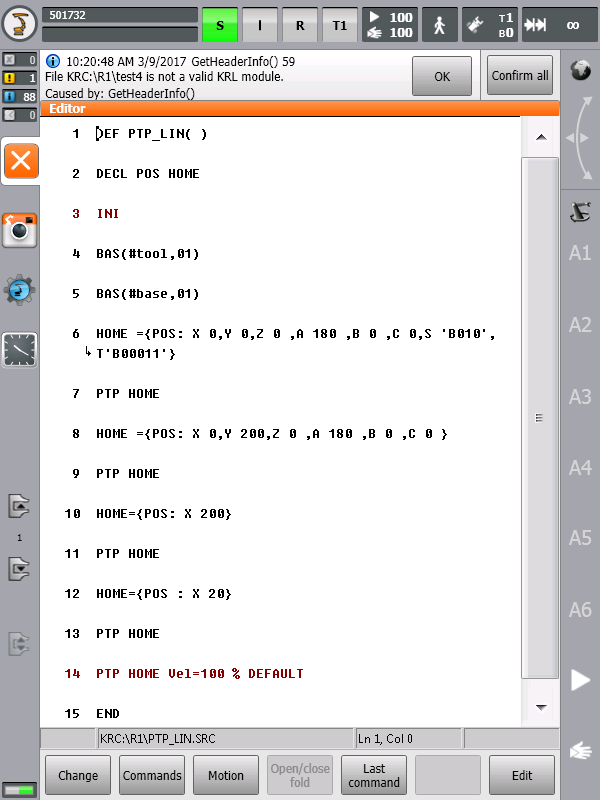
1. While in the Expert interface, make sure to choose the right directory KRC:\R1\.Click "New." to create a new module.
2. In the initialization section, define the base and tool coordinate system for the robot.

* BAS( #tool 01)
* BAS( #base 01)

Another way for initialization is to use system variables

* $TOOL = TOOL\_DATA[1]
* $BASE = BASE\_DATA[1]

1. Write PTP motion of type POS.
2. Add PTP command to **HOME** position. This is necessary in every program.



Experiment 06

Motion programming (Expert level)

Orientation Control

# Aim of the experiment

* Understand and perform basic programs using KRL
* Get Familiar with program creation and editing in "Expert" interface
* Understand and implement motion programming command in Cartesian coordinate with orientation control

# Preconditions

* Familiar with Experiment 05
* KRL basic knowledge.
* Motion programming theoretical knowledge.
* Tool and base coordinate system calibration.

# Introduction

## Orientation Control

Defining a point in space requires three translational values beside three rotational Specifications.

A, B, C are the essential elements of KUKA KRL language. These values describe the orientations of tool in the base frame. KRL uses Euler angles (Z-Y-X) instead of fixed angles( X-Y-Z)

# "CONE" Program Description

In this program, the TCP changes orientation using for loop.

**DEF cone( )**

**DECL INT i**

**; Declaration of counter 'i' variable.**

**DECL FRAME pose**

**; Declaration of 'pose' variable, FRAME structure data type.**

**$VEL.CP=0.7**

**$APO.CDIS= 50**

**$TOOL=TOOL\_DATA[2]**

**; Initialization of TOOL**

**PTP {A1 0,A2 -90,A3 90,A4 0,A5 90,A6 100}**

**$BASE=$POS\_ACT**

**LIN\_REL{B -10}**

**pose = {X 0,Y 0,Z 0,A 0,B -10,C 0}**

**FOR i= 1 TO 360**

**$BASE = $BASE : {X 0,Y 0,Z 0,A 1,B 0,C 0}**

**pose = pose : {X 0,Y 0,Z 0,A -1,B 0,C 0}**

**LIN pose C\_DIS**

**ENDFOR**

**LIN\_REL{B 10}**

**END BAS (#base 01)**

 For further information about the system variables used in Experiment 04, check User Manual in KRL-guide section > System Variables .

Experiment 07

Gripper (Expert User)

# Aim of the experiment

* Understand and perform basic programs using KRL
* Get Familiar with program creation and editing in "Expert" interface
* Program gripper to toggle between open and close states in KRL

# Preconditions

* Install gripper to the robot
* KRL basic knowledge.
* Air line connection AIR1.
* Tool and base coordinate system calibration.

Experiment 08

Gripper 2 (Expert User)

# Aim of the experiment

* Understand and perform programs using both Expert and Inline mode
* Make Modifications to inline form instructions
* Use Geometric Operator
* Program gripper to toggle between open and close states in KRL

# Preconditions

* Install gripper to the robot
* KRL basic knowledge.
* Air line connection AIR1.
* Tool and base coordinate system calibration.

This experiment is a palletizing demo, where cubes are aligned with equal spaces. Only one cube position will be taught, and the robot will automatically calculate position offsets. This is done by the geometric operator ( : ), that adsds the offset variable to the first position.

First, we need to define two variables:

* + POS RR [offset postion]
  + INT I [for loop counter]

The X axis of RR position will be set to I in the for loop, then the robot will move to anew point consisting of the original one and the offset.

**DEF grip2( )**

**INT i**

**DECL POS RR**

**BAS (#INITMOV,0 )**

**RR= {X 0,Y 0,Z 0,A 0,B 0, C 0}**

**BAS(#VEL\_CP,100)**

**FOR i =0 TO 200 STEP 100**

**FOLDs:**

**RR.X = i**

**PTP XP1:RR**

**ENDFOR**

**END**