GB

Programming manual
IFM_O2D5xx_UR_Pick_Place_Program

Version 1.0

O2D5xx

11491951 / 00 11 / 2022

Contents

1 Preliminary note

You will find instructions, technical data, approvals and further information using the QR code on the unit / packaging or at www.ifm.com.

1.1 Symbols used

✓	Requirement
▶	Instructions
▷	Reaction, result
[...]	Designation of keys, buttons or indications
→	Cross-reference
!	Important note Non-compliance may result in malfunction or interference.
i	Information Supplementary note

1.2 Legal and copyright information

1.3 Disclaimer of warranties

ifm electronic gmbh disclaims to the fullest extent authorized by law any and all warranties, whether express or implied, including, without limitation, any implied warranties of title, non-infringement, quiet enjoyment, integration, merchantability or fitness for a particular purpose.

Without limitation of the foregoing, ifm expressly does not warrant that:

- the software will meet your requirements or expectations,
- the software or the software content will be free of bugs, errors, viruses or other defects,
- any results, output, or data provided through or generated by the software will be accurate, up-to-date, complete or reliable,
- the software will be compatible with third party software,
- any errors in the software will be corrected.

Demo software and templates

Demo software and templates are provided “as is” (that is: excluding warranty) and “as available”, without any warranty of any kind, either express or implied. The user acknowledges and agrees to use the software at user’s own risk. In no event shall ifm be held liable for any direct, indirect, incidental or consequential damages arising out of the use of or incorrect use the software. The user may use the software solely for demonstration purposes and to assess the software functionalities and capabilities.

Customer-specific software

1. The software created and used has been put together by ifm especially for the customer using modular software components made by ifm for numerous applications (standard software modules) and adapted to the contractual service required (customer-specific application program).

2. Upon complete payment of the purchase price for the customer-specific application program, ifm transfers the non-exclusive, locally and temporarily unrestricted usage right thereof to the customer, without the customer acquiring any rights of any kind to the standard software module on which the individual or customer-specific adaptation is based. Notwithstanding these provisions, ifm reserves the right to produce and offer customer-specific software solutions of the same kind for other customers based on other terms of reference. In any case ifm retains a simple right of usage of the customer–specific solution for internal purposes.
3. By accepting the program, the user acknowledges and agrees to use the software at user's own risk. By accepting the program, the user also acknowledges that the software meets the requirements of the specifications agreed upon. ifm disclaims any and all warranties, in particular regarding fitness of the software for a particular purpose.

1.4 Open source information

This product can contain Free Software or Open Source Software from various software developers which is subject to the following licenses: General Public License version 1, version 2 and version 3 (General Public License version 3 in conjunction with the GNU Compiler Collection Runtime Library Exception version 3.1), Lesser General Public License version 2.1, Lesser General Public License version 3, Berkeley Software Distribution (BSD-2-Clause, BSD-3-Clause, BSD-4-Clause), MIT-License (MIT), Python Software Foundation License 2.0, Pearl Artistic License and Artistic License 2.0, Microsoft Public License, Apache Software License Version 1.0, 1.1 und 2.0, ISC License, libpng License, zlib Licence, the Academic Free License version 2.1. For the components subject to the General Public License in their respective versions the following applies:

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation. If version 1 applies to the software: either version 1 of the License or (at your option) any later version; if version 2 (or 2.1) applies to the software: either version 2 (or 2.1) of the License or (at your option) any later version; if version 3 applies to the software: either version 3 of the License or (at your option) any later version. The following disclaimer of the software developers applies to the software components that are subject to the General Public License or the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License and the GNU Lesser General Public License for more details.

The responsibility of ifm electronic gmbh for ifm products, in the case of product-specific software, remains unaffected by the above disclaimer. Please note that the firmware for the ifm products is in some cases provided free of charge.

The price of the ifm products has then to be paid for the respective device itself (hardware) and not for the firmware. For the latest information on the license agreement for your product please visit www.ifm.com

For binaries that are licensed under any version of the GNU General Public License (GPL) or the GNU LGPL you may obtain the complete corresponding source code of the GPL software from us by sending a written request to: opensource@ifm.com or to ifm electronic gmbh, Friedrichstraße 1, 45128 Essen, Germany.

We charge €30 for each request. Please write "source for product Y" in the memo line of your payment. Your request should include (i) the name of the covered binary, (ii) the name and the version number of the ifm product, (iii) your name and (iv) your return address.

This offer is valid to anyone in receipt of this information. This offer is valid for at least three years (from the date you received the GPL/LGPL covered code).

2 Safety instructions

Please read the operating instructions prior to set-up of the device. The device must be suitable for the application without any restrictions.

If the operating instructions or the technical data are not adhered to, personal injury and damage to property can occur.

3 Intended use

The programming manual describes

- the robot-sensor calibration via the ifm Vision Assistant software,
- the program flow of the example program on the Polyscope of the Universal Robot,
- the functions and parameters of the UR script file.

The example program "IFM_O2D5xx_UR_Pick_Place_Program" contains a sample application and UR script file. The UR script file is loaded onto the controller of the Universal Robot. This simplifies communication to the O2D5xx sensor via TCP/IP.

4 System requirements

The following software and hardware is required to calibrate the O2D5xx sensor and transfer the sample application to the controller.

Software

The following software is needed:

- ifmVisionAssistant software: 2.5.18 or higher
- Firmware O2D5xx sensor: 1.27.9941 or higher

Hardware

The following hardware is needed:

- a Universal Robot UR3, UR5, UR10, UR16, UR3e, UR5e, UR10e, UR16e or UR20.
- USB stick for transferring the example program to the controller. The USB stick must be formatted with FAT32.
- Tool tip for approaching the marker points.

To print a tool tip yourself, an STL file is available in the project folder "3D_prints_calibration_tools".

5 Setup

The setup connects the O2D5xx sensor, calibrates it and configures the process interface.

5.1 Connecting the O2D5xx sensor to Universal Robot

The physical connection of the O2D5xx sensor to the Universal Robot is ideally made via a network switch. By connecting the notebook with the ifm Vision Assistant software to the network switch, the O2D5xx sensor can be easily configured.

Ethernet cables with M12 connectors are recommended for data transmission, for example EVC894 (5 m length).

Connecting the O2D5xx sensor to the network

After the O2D5xx sensor and the Universal Robot are physically connected, the devices are connected to the network.

▶ Connect the ifm Vision Assistant software to the O2D5xx sensor.

▶ Select the [Device setup] in the ifm Vision Assistant.

▶ Select the [Network] section.

▶ Set the [IP address] and the [Subnet mask].

  ▷ Example for IP address: 192.168.0.69

  ▷ Example for subnet mask: 255.255.255.0

▶ Click on the [Save] button: .

Connecting Universal Robot to the network

After the O2D5xx sensor and the Universal Robot are physically connected, the devices are connected to the network.

▶ Start the Polyscope software of the Universal Robot control unit.

▶ Select the [Settings] in Polyscope.

▶ Select the [System] area.

▶ Select the [Network] submenu.

▶ Set the [IP address] and the [Subnet mask].

  ▷ Example for IP address: 192.168.0.100

  ▷ Example for subnet mask: 255.255.255.0

▶ Click on the [Apply] button:

The O2D5xx sensor and the Universal Robot must be on the same network.

The set IP addresses may only be used for the corresponding devices.

5.2 Calibrating the O2D5xx sensor with Universal Robot

A wizard is provided for calibrating the O2D5xx sensor with the coordinate system of the Universal Robot. The "Robot Sensor Calibration" wizard is part of the ifm Vision Assistant software. The wizard guides you step by step through the calibration.

When calibrating the Universal Robot, the coordinates between the Universal Robot and the O2D5xx sensor are transformed by moving the tool tip to positions on a marker sheet. The coordinates of the Universal Robot are transmitted manually to the O2D5xx sensor. The O2D5xx sensor is additionally calibrated with a marker sheet.

Start the "Robot Sensor Calibration" wizard in the ifm Vision Assistant:

▶ Select the [Application] area.

▶ Select an application and select the [Edit application] button.

▶ Select the [Images & trigger] function.

▶ Select the [Trigger & general] section.

▶ At the end of the area, select the [Calibrate] button.

▶ Select the [Robot sensor calibration] button.

  ▷ The "Robot sensor calibration" wizard starts.

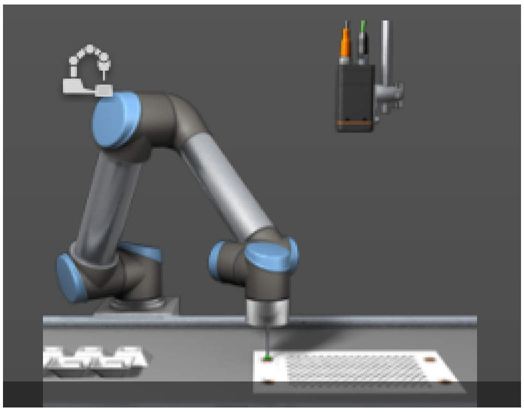

Fig. 1: “Robot sensor calibration” wizard

Preparing the calibration

The working distance between the O2D5xx sensor and the work level is decisive for the selection of the marker sheet. The working distance is measured with the assistant or manually with a measuring tape. When using a measuring tape, the working distance is measured from the orange seal of the O2D5xx sensor.

With the working distance measured, the appropriate marker sheet is selected and printed.

If the measurement is between the values of 2 marker sheets, select the larger marker sheet.

Preparing the calibration:

▶ Select the appropriate marker sheet from the list.

  ▷ The appropriate marker sheet must be selected in the list for the calibration check to succeed in the last step of the wizard.

▶ Print out the matching marker sheet.

If the O2D5xx sensor is mounted tilted or inclined, the calibration will be inaccurate.

▶ Mount the O2D5xx sensor vertically to the work level.

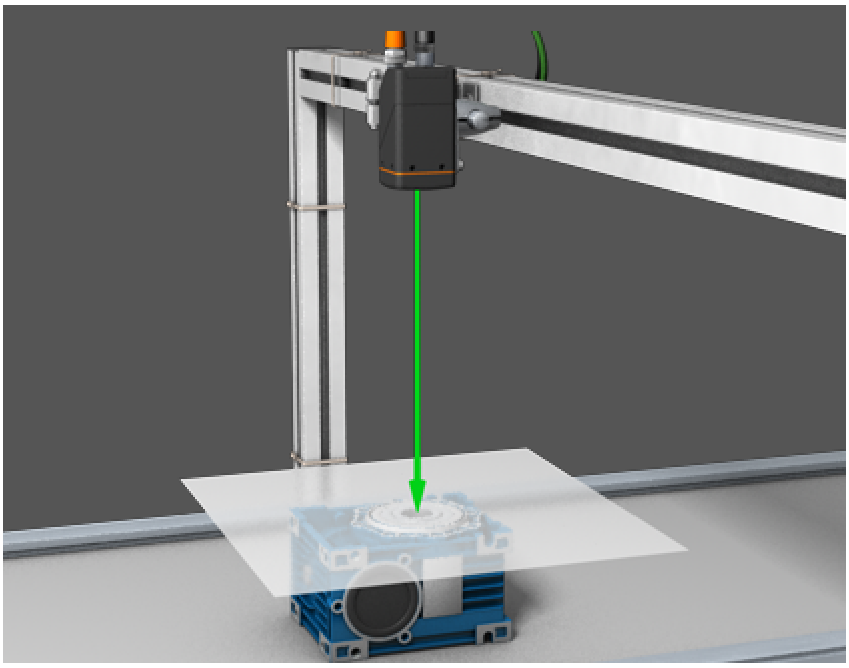

Fig. 2: Vertically mounted O2D5xx sensor with working distance highlighted in green

Image settings, focus and exposure

To adjust the focus, the marker blade is placed under the O2D5xx sensor at the level of the work level. The [Autofocus] function is used to adjust the focus. The dots on the marker sheet are then displayed in focus.

The image can additionally be rotated by 180 degrees, which can make it easier to learn the dot pattern.

To set the exposure, use the [Auto exposure] function in the next step of the wizard. Then the dots on the marker sheet are displayed with the best possible contrast.

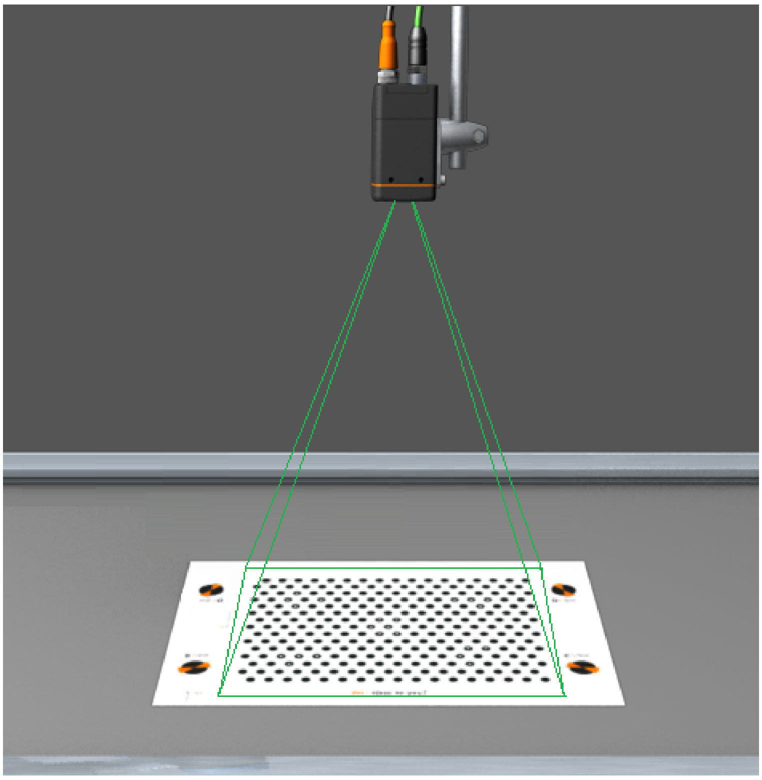

Fig. 3: Marker sheet placed below the O2D5xx sensor

Robot coordinates

To set the robot coordinates, the tool tip of the Universal Robot is placed one after the other over the 4 marking points of the marker sheet.

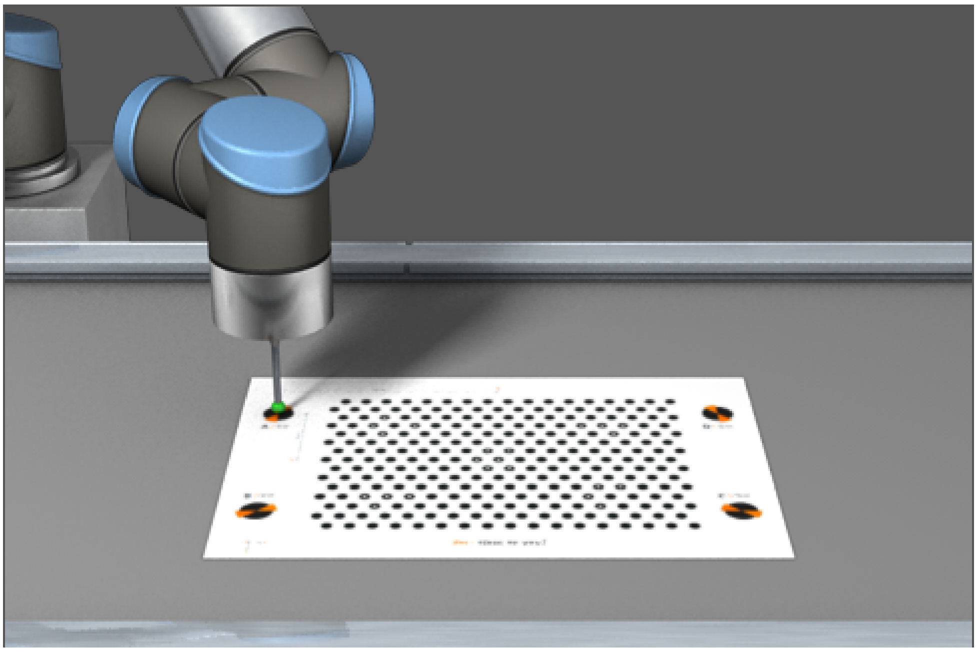

Fig. 4: Tool tip placed centrally over marking point.

The tool tip is moved with Polyscope to the 4 marking points A to D one after the other with the

- Freedrive mode or
- [Move] menu.

For accurate calibration, place the tool tip as centrally as possible over the marking points.

Transferring the measured values of the marking points:

▶ Transfer the measured values of the 4 marking points from Polyscope to the ifm Vision Assistant.

  ▷ The measured values are in Polyscope in the [Move] menu. In Polyscope, the measured values are given in [mm] and in the ifm Vision Assistant in [m].

▶ Select the [Teach] button:

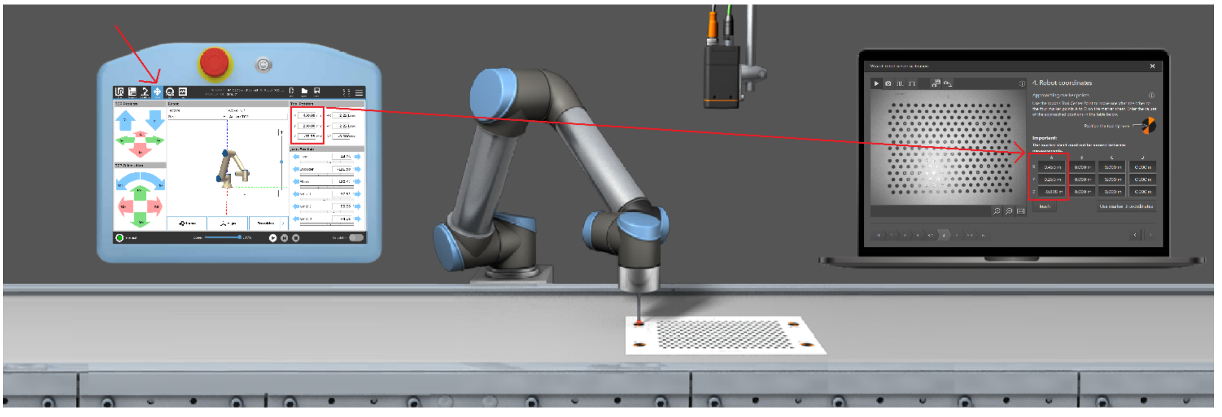

Fig. 5: Transferring the measured values from Polyscope to the ifm Vision Assistant

Sensor calibration

To establish the sensor coordinates, up to 16 images of the marker sheet are taken at different positions in the field of view of the O2D5xx sensor. For each image, the marker sheet is moved and rotated slightly.

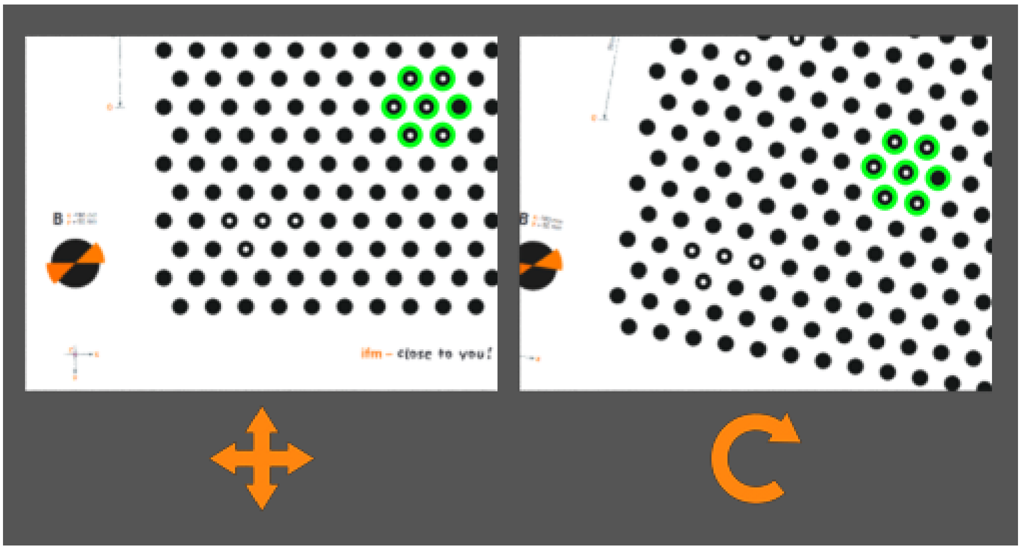

Fig. 6: Rotated marker sheet

When shifting and rotating, the dot pattern must be visible in the image field of the O2D5xx sensor.

The quality of the sensor calibration is indicated in percent. A value of > 85 % is necessary for good sensor calibration.

Calibration of the Z offset

If the marker sheet was not level with the work level during calibration, this can be compensated for with the Z-offset.

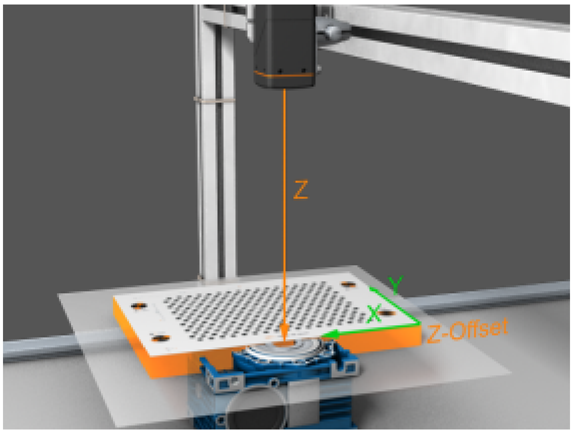

Fig. 7: Calibrating the Z offset

The marker sheet was above the work level:

▶ Enter a positive value in the [Z offset] input field.

The marker sheet was below the work level:

▶ Enter a negative value in the [Z offset] input field.

Large values for Z offset degrade the accuracy of the calibration.

Test

The last step of the wizard is to check the accuracy of the calibration.

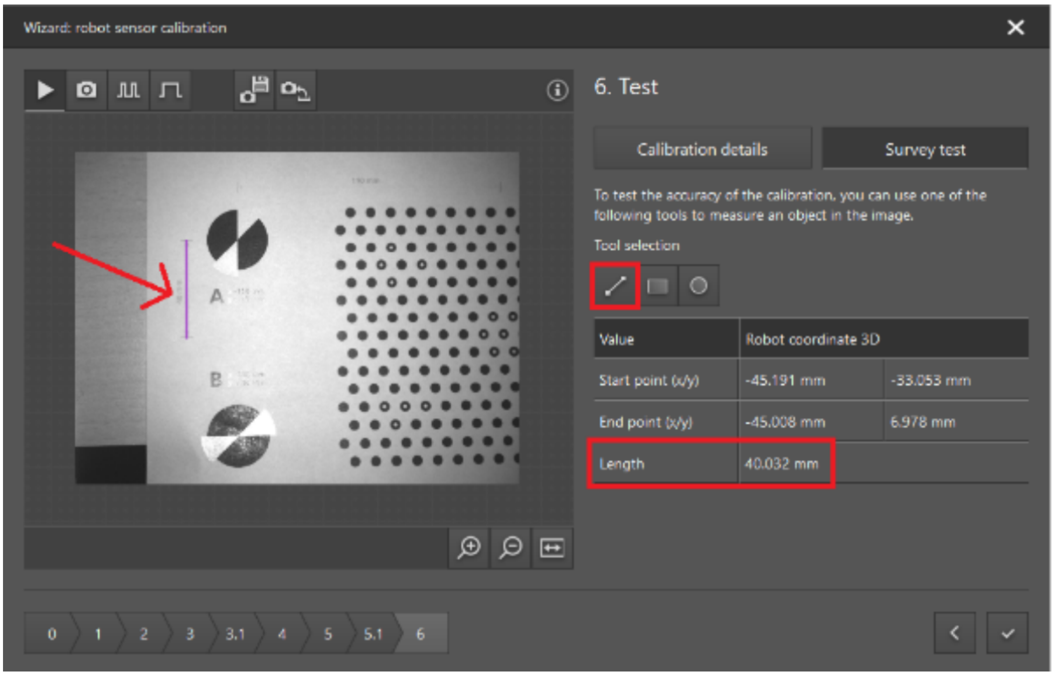

Fig. 8: Checking the accuracy

Length information is printed on the marker sheet (see arrow in screenshot). The buttons [Line], [Rectangle] and [Circle] are used to check the lengths of the marker sheet in the live image.

5.3 Configuring the process interface

The configuration of the process interface sets the structure of the output via TCP/IP. The following values are output:

- number of contour objects found in a ROI (Region of Interest),
- translational values X, Y and Z of each object in the coordinate system of the Universal Robot (scaled by 1000 for output of [mm]),
- rotatory value Rx of each object (constant value "-180 degrees"),
- rotatory value Ry of each object (constant value "0 degrees"),
- rotatory value Rz of each object (rotational position of the object),

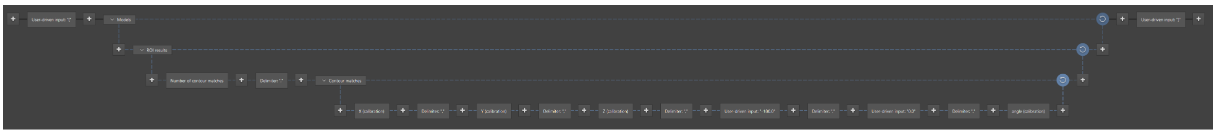

Fig. 9: Provided values

Configuring the process interface in the ifm Vision Assistant:

▶ Select the [Application] area.

▶ Select an application and select the [Edit application] button.

▶ Select the [Interfaces] function.

▶ In the [Presets] list, select the configuration for the Universal Robot.

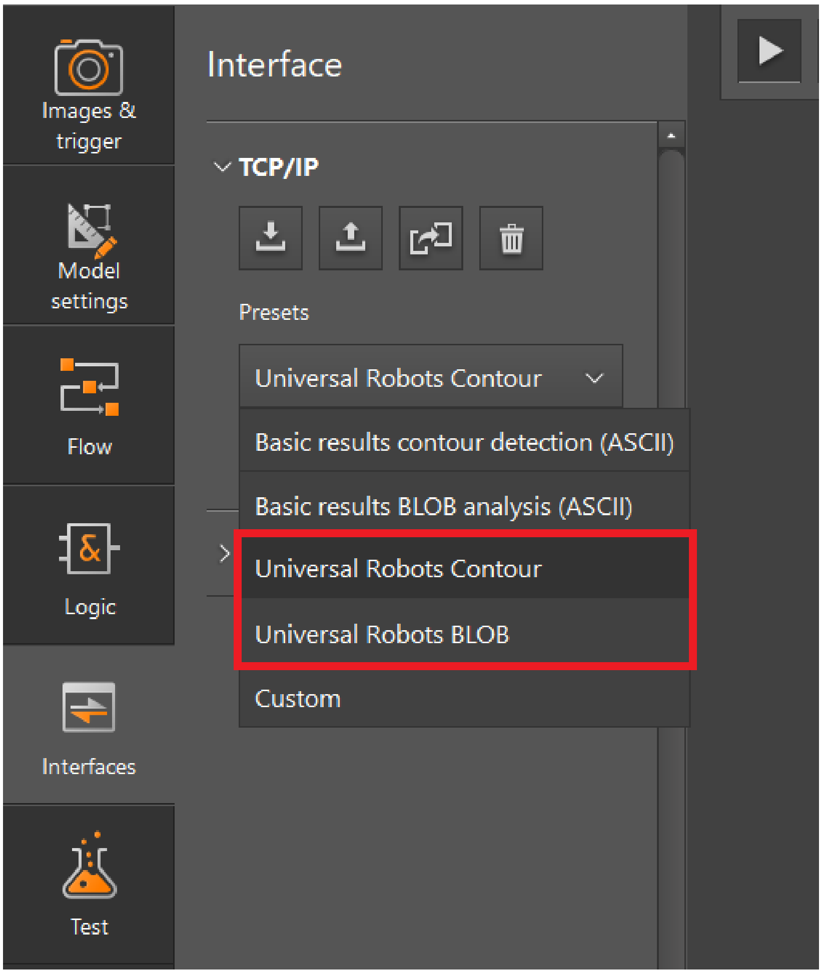

Fig. 10: Configuration [Object Results Universal Robots (ASCII)]

6 Example program

Flow

The Pick & Place example program runs cyclically in 8 steps.

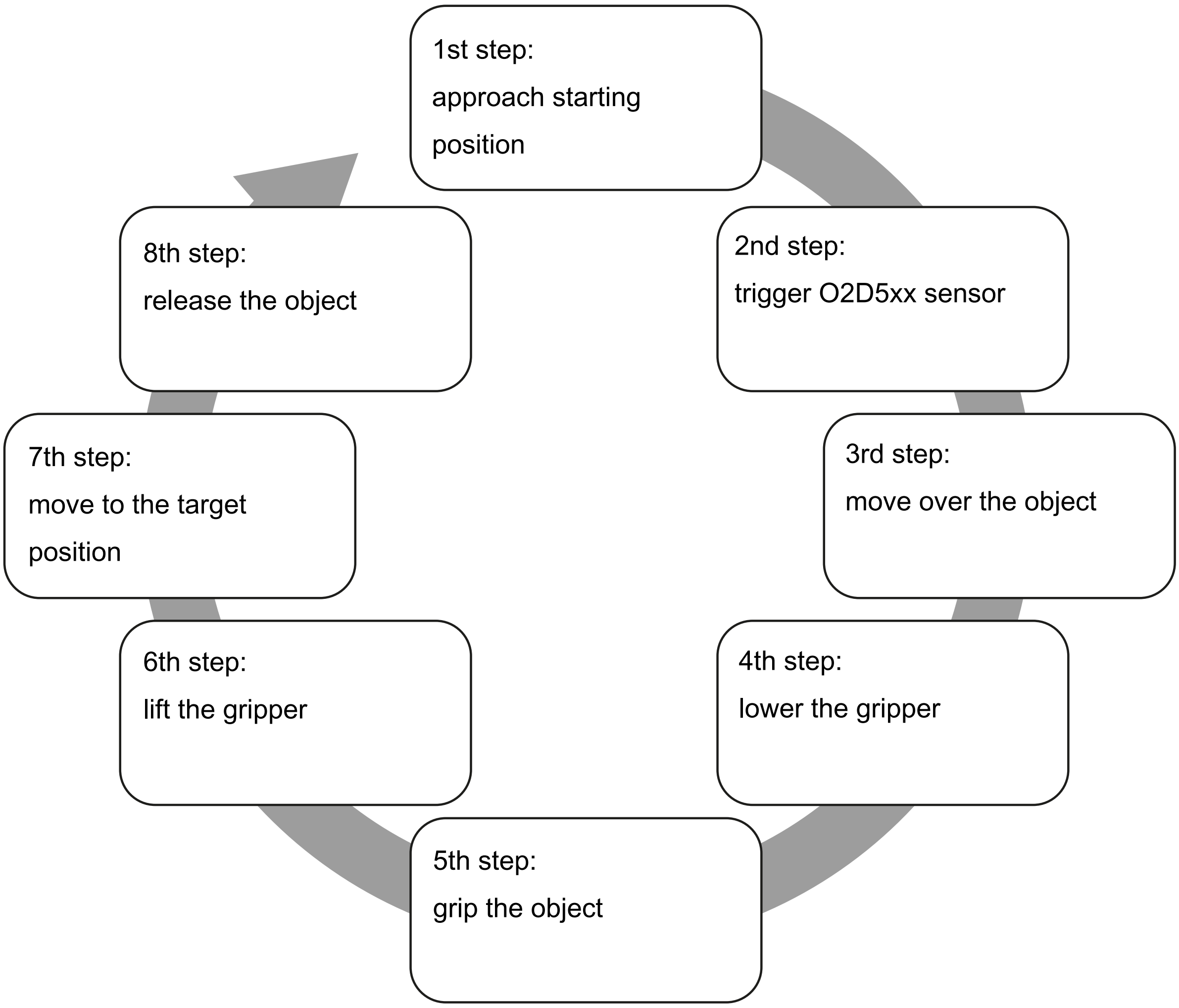

Fig. 11: Program sequence

In the 1st step, the starting position is approached. In the 2nd step, the O2D5xx sensor is triggered by the Universal Robot controller. If a result is found, the 3rd to 8th place is used. Continue with the step for moving the object.

Structure

The code of the example program is commented and describes the cyclic programme flow of a pick & place application with the Universal Robot.

```
     Init Variables
(1)  ▼ BeforeStart
(2)    Application Setup
         'Load IFM script file.'
         Script: IFM_O2D5xx_UR_Connection_Lib.script
         'Setup start position and end position.'
         MoveJ
(3)        Pos_Start
(4)        Pos_End
         CON_initialize_start_pose(Pos_Start)
         CON_initialize_end_pose(Pos_End)
         'Setup pick and elevate offset values.'
(5)      pickOffsetZ:=...
(6)      elevateOffsetZ:=...
(7)    Tool Setup
         'Activate the tool.'
(8)      Es fehlt: Greifer aktivieren
         'Open gripper.'
(9)      Es fehlt: Greifer
(10)   Sensor Setup
         'Connect to IFM sensor.'
(11)     CON_initialize_sensor_variables("O2D5xx", "192.168.0.69", 50010, 3)
         If CON_initialize_sensor_connection()==False
           Popup: Connection to IFM sensor failed!
(12) ▼ Robot Program
       Pick&Place Program
         Steps 1-8
           'Step 1: Move to start position.'
           CON_pick_place_move_step(1)
           'Step 2: Send trigger to sensor and check result.'
           triggerResult:=CON_read_sensor_data()
           Wait: 0.5
           'Step 2: Check if an object was found, else wait 2 seconds for next trigger.'
           If triggerResult≟1
             Wait: 2.0
           'Step 2: Check length of input array. If missmatching close socket and popup error.'
           If triggerResult≟2
             CON_close_connection("O2D5xx")
             Popup: Data missing in input array! Check your PCIC configuration on the sensor!
           'Step 3-7: Pick & Place sequence on found object.'
           If triggerResult≟0
             Steps 3-8
               'Step 3: Move to position above the object.'
               CON_pick_place_move_step(3)
               'Step 4: Lower gripper.'
               CON_pick_place_move_step(4)
               'Step 5: Close gripper and pick object.'
(13)           Es fehlt: Greifer
               'Step 6: Elevate gripper.'
               CON_pick_place_move_step(6)
               'Step 7: Move object to desired position.'
               CON_pick_place_move_step(7)
               'Step 8: Open gripper and place object.'
               Es fehlt: Greifer
```

Fig. 12: Example program

1 `BeforeStart` : Pre-sequence, is executed once at the start of the program.

2 `Application Setup` : Sets the application.

3 `Pos_Start` : Sets a start position for the gripper in standby. As long as no object has been found, the gripper remains in the start position. The start position must be outside the field of view of the O2D5xx sensor.

4 `Pos_End` : Sets an end position for the gripper. The gripper moves to the end position to put an object down.

5 `pickOffsetZ` : Sets the distance in positive Z direction when the gripper is positioned over an object. Unit: Metre [m], recommended value:" `0.02` "

6 `elevateOffsetZ` : Sets the distance in positive Z direction when the gripper lifts an object. Unit: Metre [m]

7 `Tool Setup` : Sets the gripping or suction tool.

8 Activates the gripping or suction tool. If the gripping or suction tool is already activated before the program starts, this line can be deleted.

9 Opens the gripping tool.

10 `Sensor Setup` : Sets the O2D5xx sensor.

11 `CON_initialize_sensor_variables` : Sets the IP address, port number and protocol version of the TCP/IP from the O2D5xx sensor.

12 `Robot Program` : Main sequence, is executed cyclically.

13 Closes the gripping tool.

7 UR script file

The UR script file "IFM_O2D5xx_UR_Connection_Lib.script" contains the following functions.

Function	Description
CON_initialize_start_pose (→ CON_initialize_start_pose 🗋 18)	Initialising the global variable STARTPOS (Pose of the start position).
CON_initialize_end_pose (→ CON_initialize_end_pose 🗋 18)	Initialising the global variable ENDPOS (Pose of the end position).
CON_initialize_sensor_variables (→ CON_initialize_sensor_variables 🗋 19)	Initialising the global variables SENSOR (name of the O2D5xx sensor), IP (IP address of the O2D5xx sensor, PORT (Telnet port) and TCPIPVERSION (TCP/IP protocol version).
CON_initialize_sensor_connection (→ CON_initialize_sensor_connection 🗋 19)	Establishing the socket connection with the O2D5xx sensor.
CON_read_sensor_data (→ CON_read_sensor_data 🗋 19)	Triggering the O2D5xx sensor and reading input arrays, configured via the process interface (number of objects, X, Y, Z, Rx, Ry, Rz). Check the number of objects and the size of the input array.
CON_pick_place_move_step (→ CON_pick_place_move_step 🗋 19)	Moving the robot arm according to the pick & place step pattern.
CON_convert_XYZRPY_to_axis_angle (→ CON_convert_XYZRPY_to_axis_angle 🗋 20)	Scaling the translational values X, Y and Z and converting the Euler angles into the rotation vector. The final pose to be approached is output.
CON_close_connection (→ CON_close_connection 🗋 20)	Closing the socket connection with the O2D5xx sensor.

Tab. 1: Functions of the script file

7.1 CON_initialize_start_pose

Version

Version V1.0

Description

Initialising the global variable STARTPOS (Pose of the start position).

Input parameter

Name	Data type	Description
Pos_Start	pose	Waypoint in space.

Tab. 2: Function CON_initialize_start_pose

7.2 CON_initialize_end_pose

Version

Version V1.0

Description

Initialising the global variable ENDPOS (Pose of the end position).

Input parameter

Name	Data type	Description
Pos_End	pose	Waypoint in space.

Tab. 3: Function CON_initialize_end_pose

7.3 CON_initialize_sensor_variables

Version

Version V1.0

Description

Initialising the global variable SENSOR (name of the O2D5xx sensor), IP (IP address of the O2D5xx sensor, PORT (Telnet port) and TCPIPVERSION (TCP/IP protocol version).

Input parameter

Name	Data type	Description
sensor	string	Name of the O2D5xx sensor (e.g.: “O2D520“).
sensor_ip	string	IP address of the O2D5xx sensor (default: "192.168.0.69").
port	numeric	Port used (default: “0010”).
tcp_ip_version	numeric	TCP/IP protocol version used (default: “3”).

Tab. 4: Function CON_initialize_sensor_variables

7.4 CON_initialize_sensor_connection

Version

Version V1.0

Description

Setting up the socket connection with the O2D5xx sensor.

Input parameter

No input parameters.

7.5 CON_read_sensor_data

Version

Version V1.0

Description

- Triggering the O2D5xx sensor and reading input arrays, configured via the process interface (number of objects, X, Y, Z, Rx, Ry, Rz).
- Check the number of objects and the size of the input array.

Input parameter

No input parameters.

7.6 CON_pick_place_move_step

Version

Version V1.0

Description

Moving the robot arm according to the pick & place step pattern.

Input parameter

Name	Data type	Description
step	numeric	Move the robot arm to a step number. (→ Example program 🗋 15) For example, step number "1" moves the robot arm to the start position.

Tab. 5: Function CON_pick_place_move_step

7.7 CON_convert_XYZRPY_to_axis_angle

Version

Version V1.0

Description

Scaling the translational values X, Y and Z and converting the Euler angles into the rotation vector. The final pose to be approached is output.

Input parameter

Name	Data type	Description
array	pose	The pose of the object with the translational values XYZ and rotational values RPY.

Tab. 6: Function CON_convert_XYZRPY_to_axis_angle

7.8 CON_close_connection

Version

Version V1.0

Description

Closing the socket connection with the O2D5xx sensor.

Input parameter

No input parameters.