

Bin Picking Studio

Introduction and Training

Version 1.6



Schedule and Agenda

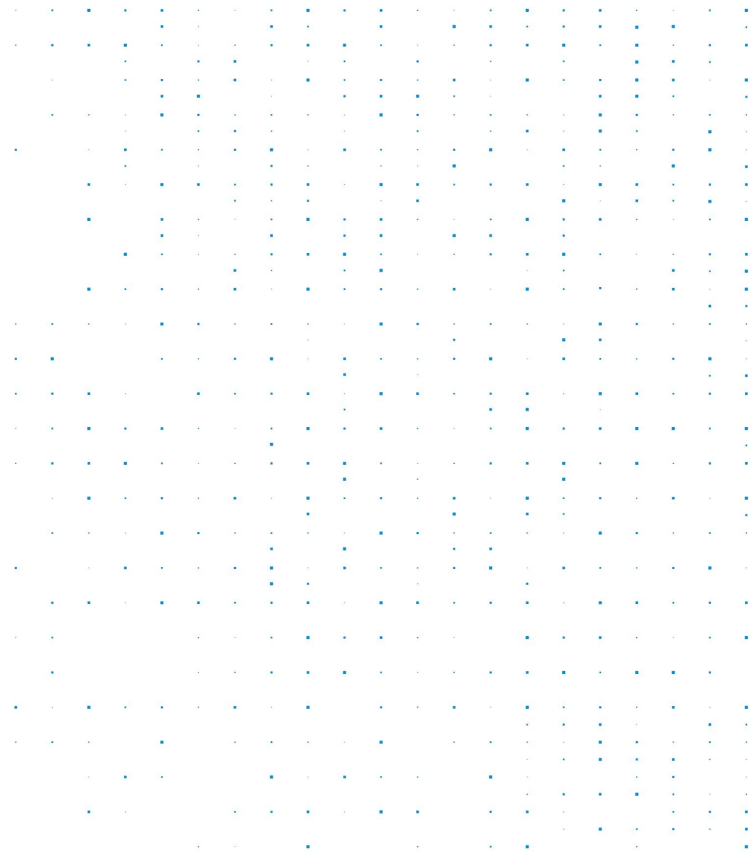
Program of standard Bin Picking Studio training:

- 09:00 - 09:15 Sales Introduction of company and products
- 09:15 - 12:00 Bin Picking Studio theory, configuration of object, grasping method, localization and calibration
- 12:00 - 13:00 Lunch break
- 13:00 - 13:30 Configuration of environment and settings
- 13:30 - 14:30 Use case and demo

Drop your objects for bin picking to presenter for practical section of the training

Overview and topology

Bin Picking Studio



Deployment

Bin Picking Client: DISCONNECTED Robot State Server: DISCONNECTED Scanner: 1

★ ID: 8 | Bin picking demo

✖ Stop bin picking ⏮ Restart

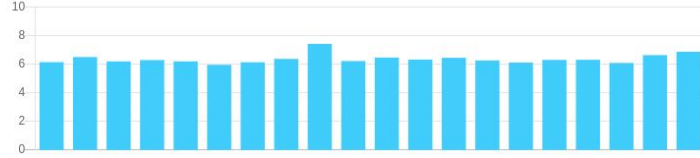
📷 Scan ▾

Cycle Time

Trajectory Requests Scan Requests

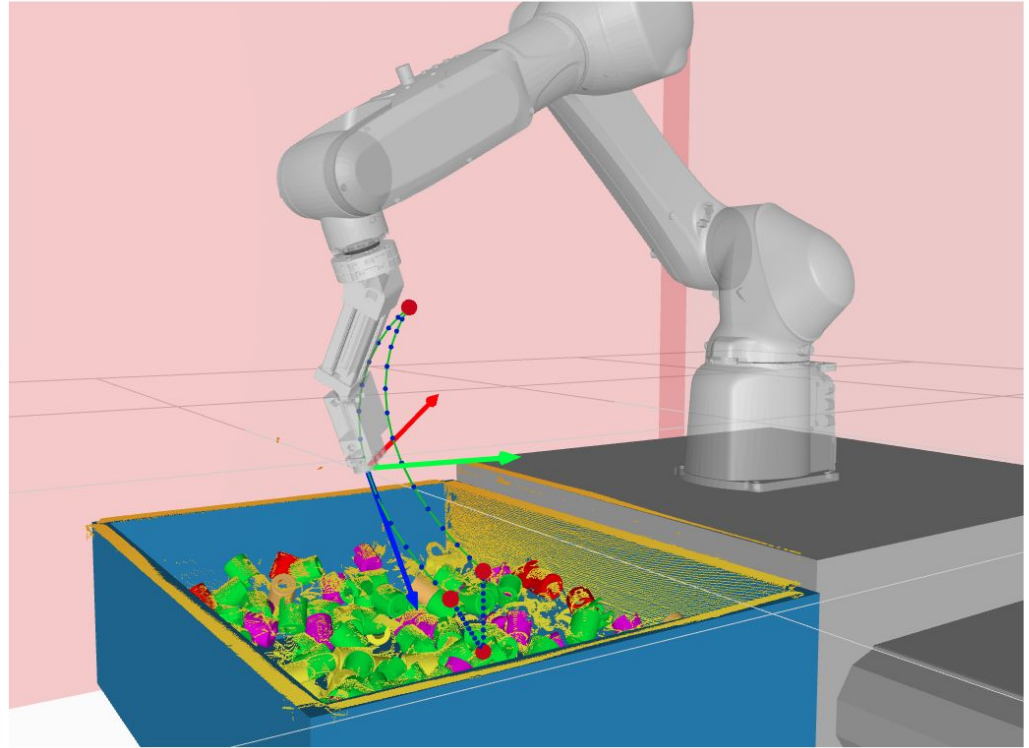
Average cycle time: 6.4 s

VS 1



19:20:21 - [INFO] Localization found the first object [Vision System 1]
 19:20:22 - [INFO] A path plan has been computed for the first picked object
 19:20:24 - [INFO] Request for trajectory received [Vision System 1]
 19:20:24 - [INFO] Picking object at location -0.376268 0.864437 -0.0721773
 19:20:26 - [INFO] Request for scan received [Vision System 1]
 19:20:26 - [INFO] Localization is stopping [Vision System 1]
 19:20:26 - [INFO] Localization stopped [Vision System 1]
 19:20:26 - [INFO] Waiting for localized objects [Vision System 1]
 19:20:28 - [INFO] Localization found the first object [Vision System 1]
 19:20:28 - [INFO] A path plan has been computed for the first picked object
 19:20:30 - [INFO] Request for trajectory received [Vision System 1]
 19:20:30 - [INFO] Picking object at location -0.376268 0.864437 -0.0721773
 19:20:32 - [INFO] Request for scan received [Vision System 1]
 19:20:32 - [INFO] Localization is stopping [Vision System 1]
 19:20:32 - [INFO] Localization stopped [Vision System 1]
 19:20:32 - [INFO] Waiting for localized objects [Vision System 1]
 19:20:34 - [INFO] Localization found the first object [Vision System 1]
 19:20:35 - [INFO] A path plan has been computed for the first picked object
 19:20:37 - [INFO] Request for trajectory received [Vision System 1]
 19:20:37 - [INFO] Picking object at location -0.376268 0.864437 -0.0721772

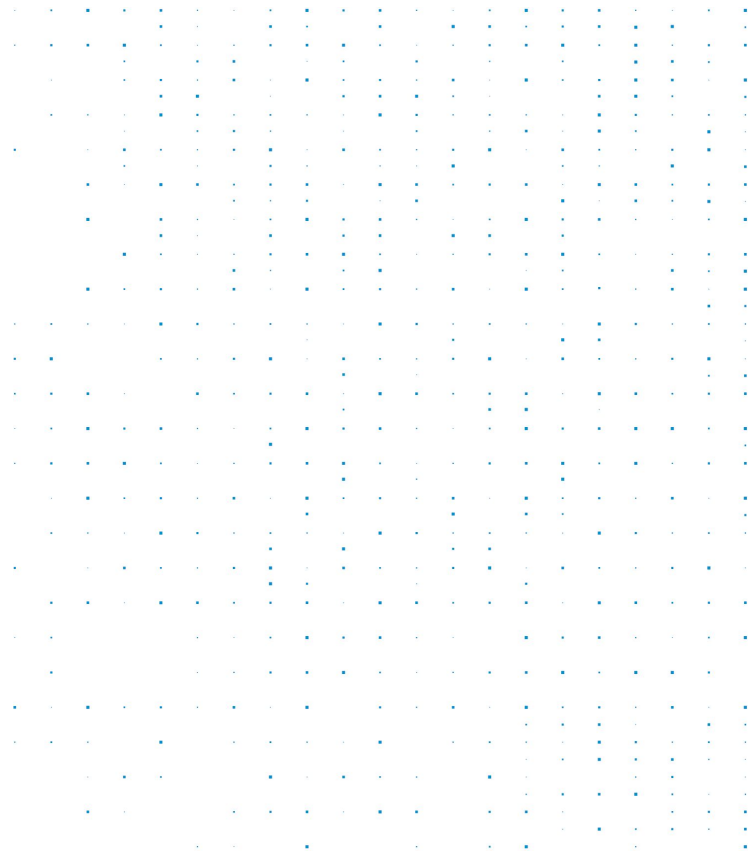
View ▾ Camera ▾ Visibility ▾ Axis ▾ Measure Point Cloud ▾ Scan ▾ Joint States ▾ Bounding Boxes ▾



Bin picking Studio is web based application

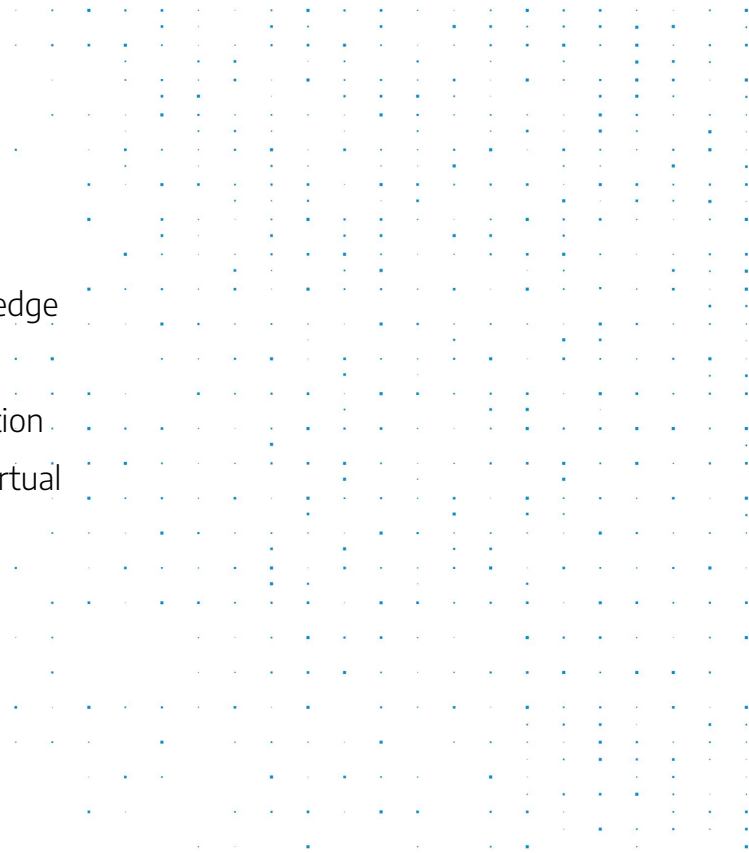
Motivation

Transforming experience and knowledge from projects delivered by Photoneo and feedback from our existing customers into a product which enables our partners to deploy robust and reliable bin picking applications



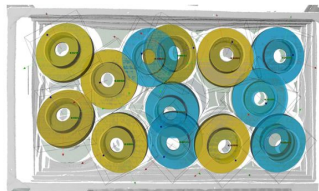
Key benefits for customers

- Web interface - accessible from remote device
- Visual representation of each step of the configuration
- Fast initial setup - Easy to use - no programming knowledge required
- Reusability of previous bin picking solutions - Quick reconfiguration
- Less factory line downtimes - configuration and testing in a virtual environment
- Tools for debugging to work around general challenges
- Wide portfolio of robotic brands
- Built-in calibration interface
- Multiple scanners in one system

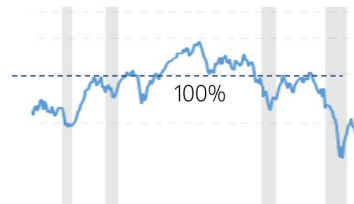


Key benefits for customers

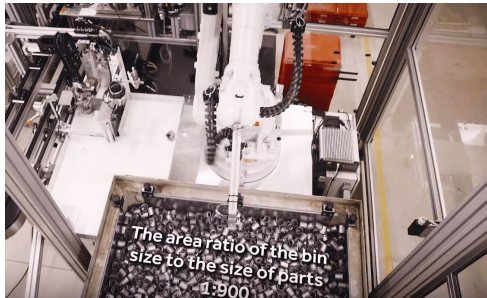
99.5% success rate



Up Time 24/7



Down to 5 seconds cycle time



Oriented placing

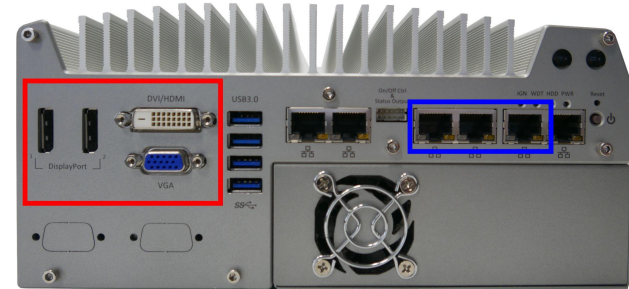


Vision controller

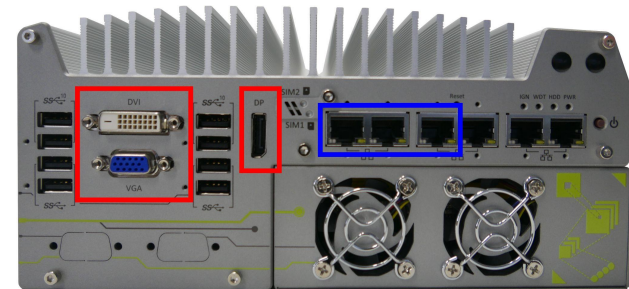
Front side of the Vision controller

- LAN ports with dedicated functions
From left to right:
 - Robot interface
 - Scanner interface
 - Network interface
- Motherboard GPU ports cannot be used with Bin Picking Studio

Type A



Type B



Vision controller

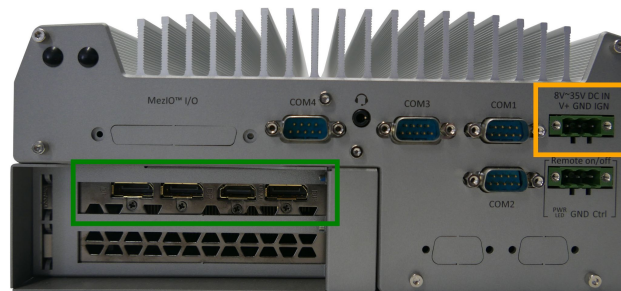
Back side of the Vision controller

- Dedicated GPU ports for connection of a monitor
 - type A: HDMI / DVI / DP
 - type B: 3xDP / HDMI or HDMI / DVI / DP
- Power socket

Type A



Type B



Hardware overview

- Vision controller is an industrial-grade device running Ubuntu operating system with pre-installed Bin Picking Studio
- PhoXi 3D scanner is the eyes of the system
- Bin Picking Studio is compatible with variety of industrial robotic brands

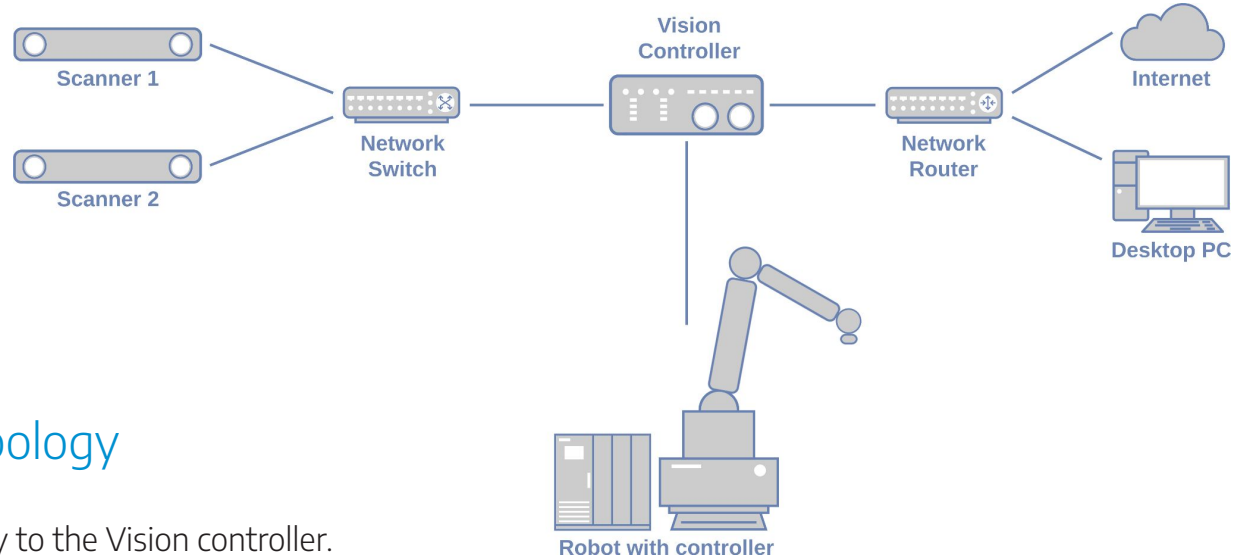


KUKA **ABB**

FANUC **Kawasaki**

YASKAWA **UNIVERSAL ROBOTS**

Topology



Exemplar hardware topology

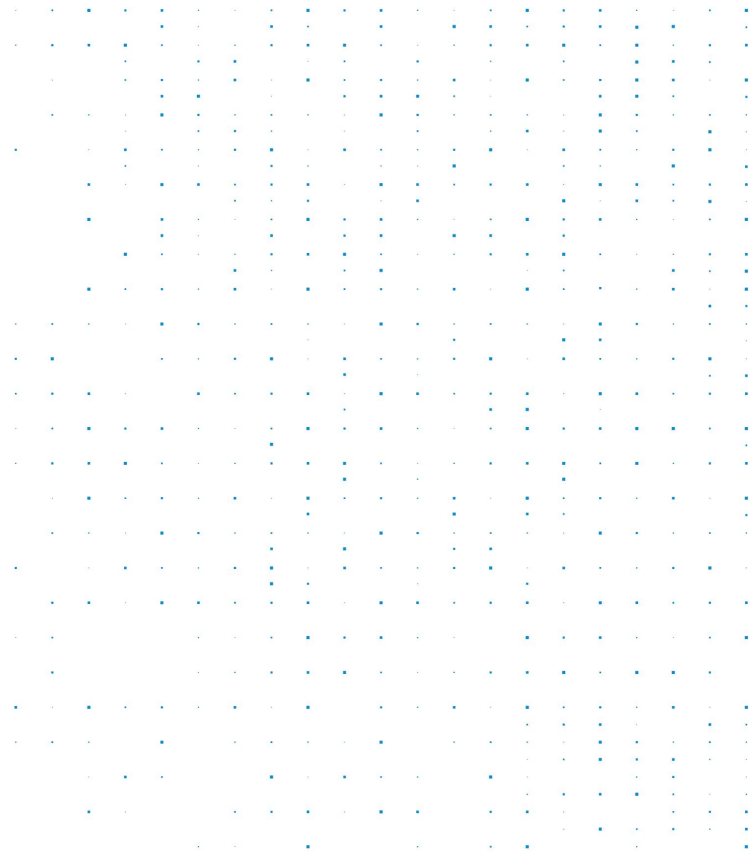
Single scanner is connected directly to the Vision controller.

In multiple-scanner applications a network switch is used.

Bin Picking Studio

configuration steps

Bin Picking Studio



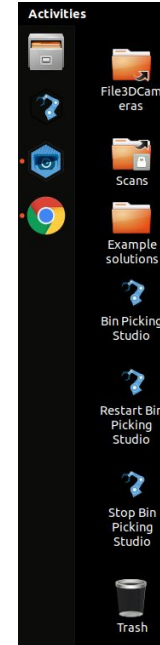
Bin Picking Studio configuration steps

When the Vision controller boots up, the Bin Picking Studio is launched automatically.

To start, stop or restart the Bin Picking Studio use the desktop/launcher shortcuts.

1. Welcome page & The first login
2. Navigation bar
3. Solution list
4. Deployment
5. Network configuration
6. PhoXi interface & Setting proper scanning parameters

The instruction on how to change the keyboard language can be found [here](#).



1. Welcome page & The first login

Initial setup of Bin Picking Studio

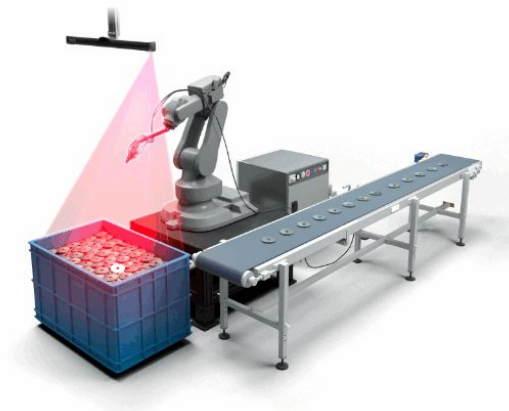
Welcome page for the initial, quick setup of:

- the installation timezone
- customer's master account
- other user accounts
 - optional, other accounts can be created at any time

Step 1/4

Welcome to Bin Picking Studio 1.6.0

In this section, we will guide you through the setup process of your Vision Controller. This process consists of setting up your timezone and creating a master account.



Continue to: Select your timezone >

1. Welcome page & The first login

Bin Picking Studio 1.6.0 by **Photoneo**

Login

Email*

Password*



☐ Stay logged in

Login

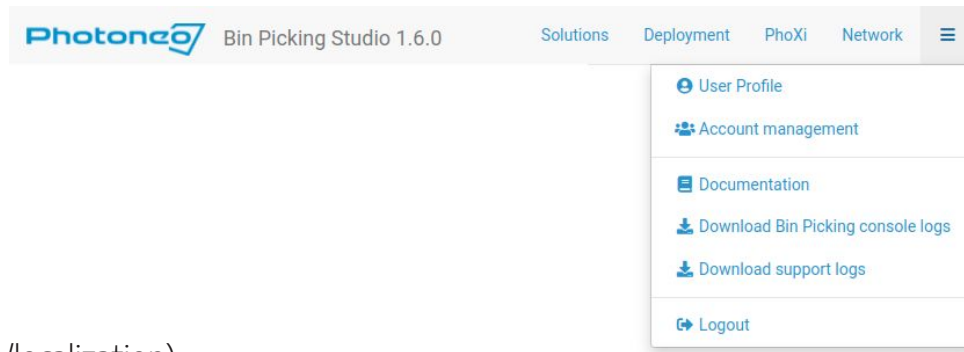
Login credentials

- Web address of Bin Picking Studio:
 - localhost
 - http://127.0.0.1
- Use one of the login credentials created during initial setup of Bin Picking Studio

2. Navigation bar

The navigation bar enables quick access to:

- Solutions list
- Deployment page
- PhoXi interface
- Network page
- currently running process (deployment/calibration/localization)



The menu button (☰) in the top right corner offers additional options:

- User profile & Account management
- Documentation
- Logs downloaders

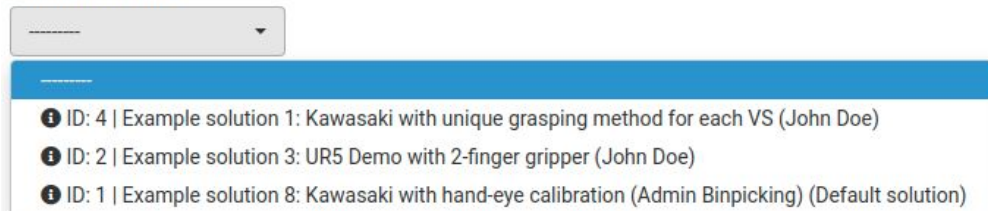
3. Solution list

☐	Name	ID	Robot	Author	Last modified	Flags	Actions
☐	Example solution 1: Kawasaki with unique grasping method for each VS	4	RS-007L	me	2022-01-31 10:20	ready	Duplicate Export Delete
☐	Example solution 3: UR5 Demo with 2-finger gripper	2	UR5	me	2022-01-31 10:20	deployed	Duplicate Export
☐	Example solution 4: KUKA Demo with 2-finger gripper	3	KR 6 R700 2	me	2022-01-31 10:20		Duplicate Export Delete
☐	Example solution 8: Kawasaki with hand-eye calibration	1	RS-007L	Admin Binpicking	2022-01-31 10:18	factory default ready	Duplicate Export

- Solution holds full configuration of your application with the exception of network parameters of your installation
- Only after completing all the steps in the solution wizard, the bin picking application can be deployed
- A completely configured solution has ready label through which it can also be directly deployed
- Currently deployed solution has the deployed label
- Solution marked as factory default is preconfigured by Photoneo and cannot be accessed directly (must be duplicated)

4. Deployment

Solution to deploy

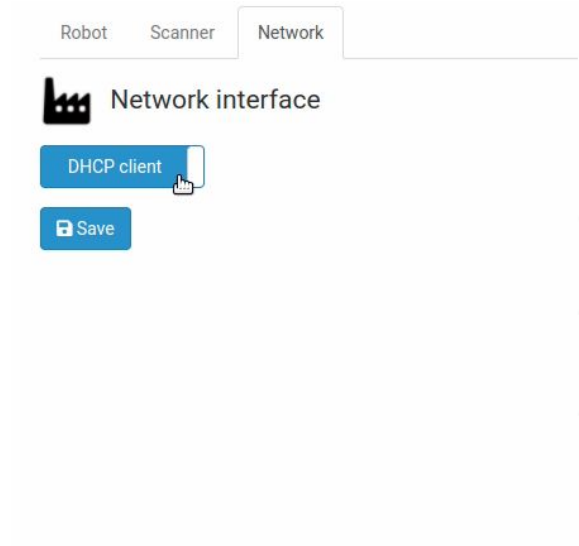


The screenshot shows a web interface for selecting a solution to deploy. At the top, there is a label "Solution to deploy" above a dropdown menu. Below the dropdown, a list of three solutions is displayed, each preceded by an information icon (i) and an ID. The solutions are:

- ID: 4 | Example solution 1: Kawasaki with unique grasping method for each VS (John Doe)
- ID: 2 | Example solution 3: UR5 Demo with 2-finger gripper (John Doe)
- ID: 1 | Example solution 8: Kawasaki with hand-eye calibration (Admin Binpicking) (Default solution)

- Select the solution to be deployed
- Only fully configured solutions are available for deployment
- A solution can be also deployed using the  label/button on the Solutions list page or directly from the Solution wizard

5. Network configuration



Network interface

- IP address of the Network interface on the Vision controller
- Physical ethernet port marked with **Network** label
- Used for connection to the Internet, the factory network, and other devices on the network
- Remote access methods:
 - VNC
 - Web server - through a web browser on another computer
 - TeamViewer

5. Network configuration

Scanner interface

- IP address of the Scanner interface on the Vision controller
- Physical ethernet port marked with **Scanner** label
- To achieve maximal robustness of the configuration, we recommend using static IPv4 address
- Multiple scanners are connected through a Gigabit network switch



5. Network configuration

Robot Scanner Network

 **Robot interface**

IPv4 address* ⓘ ↺

Subnet mask <1; 32>* ⓘ ↺

Bin Picking Server port <1024; 65535> ⓘ

 **Robot controller**

IPv4 address* ⓘ ↺

Robot State Server port <1024; 65535> ⓘ

Configure the IP addresses to be on the same subnet. Use the [Test connection](#) button to verify the connection. It is recommended not to use custom ports unless necessary.

Robot interface

- IP address of the Robot interface on the Vision controller
- Physical ethernet port marked with [Robot](#) label

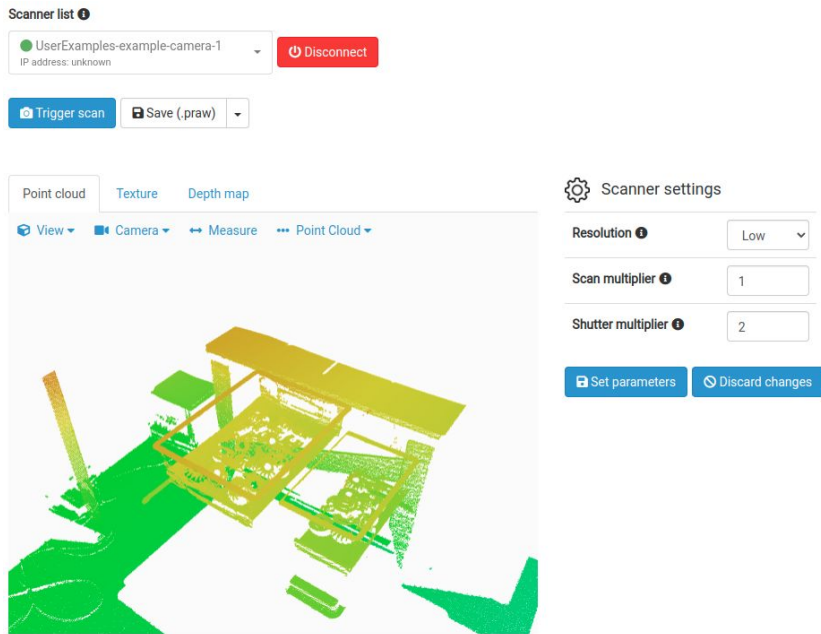
Robot controller

- IP address of the Robot controller

6. PhoXi interface & Setting proper scanning parameters

- Default scanner settings are satisfactory for most applications
- Detailed information about the scanning parameters of the PhoXi 3D Scanner is outside of the scope of this training
 - Consider attending dedicated training for PhoXi scanners
- Please refer to the materials at the end of this presentation or contact us at support@photoneo.com in case of any questions

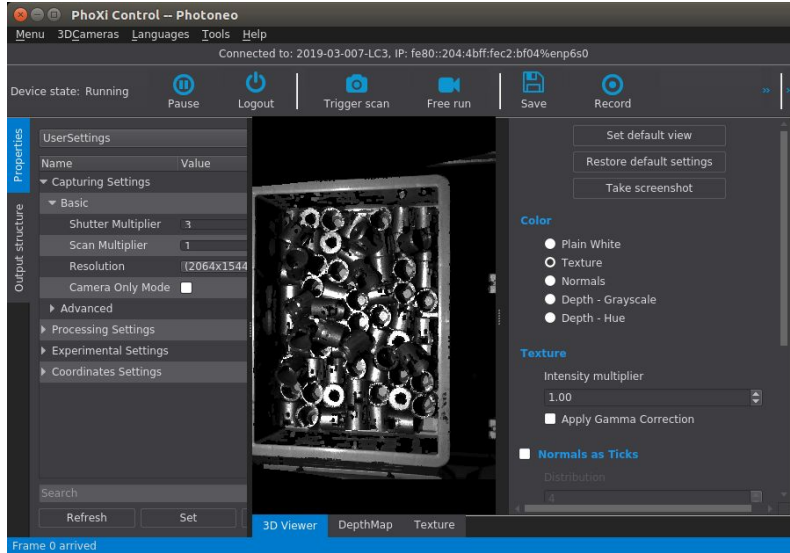
6. PhoXi interface & Setting proper scanning parameters



Integrated PhoXi Control web interface

- For testing of basic scanning parameters:
 - Resolution
 - Scan multiplier
 - Shutter multiplier
- Use [Set parameters](#) to apply your settings
- Supported formats for downloading a scan:
 - .praw
 - .ply
 - .ptx
- Use [PhoXi Control](#) for full configuration options

6. PhoXi interface & Setting proper scanning parameters



PhoXi Control

- Runs parallelly with the Bin Picking Studio
- Offers full scanner configuration options
 - configuration of all scanning parameters
 - setting the parameters into the device
 - creating custom profile with scanning parameters
 - setting the startup profile



6. PhoXi interface & Setting proper scanning parameters

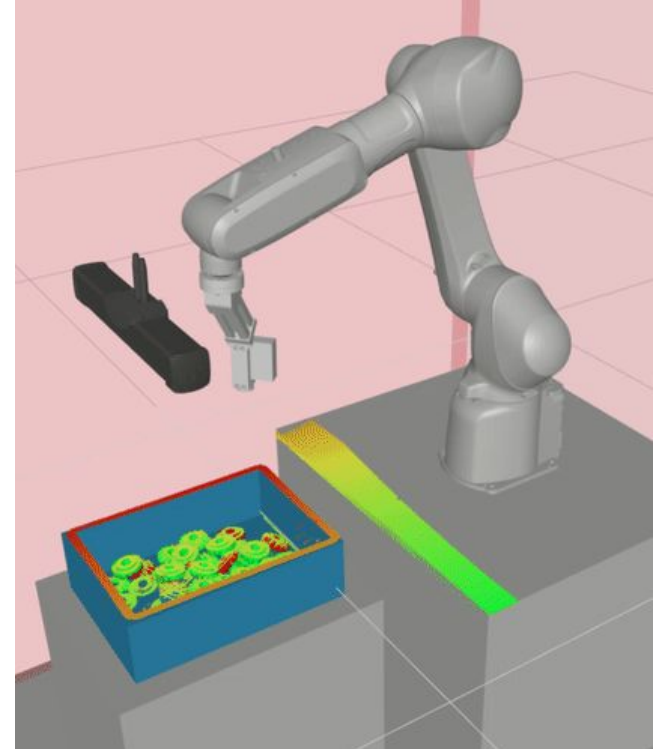
Supported scanner mount options

Above the bin in static position

- Better for cycle time optimisation
- Suitable for larger scanner models (L, XL)
- Does not limit robot maneuverability

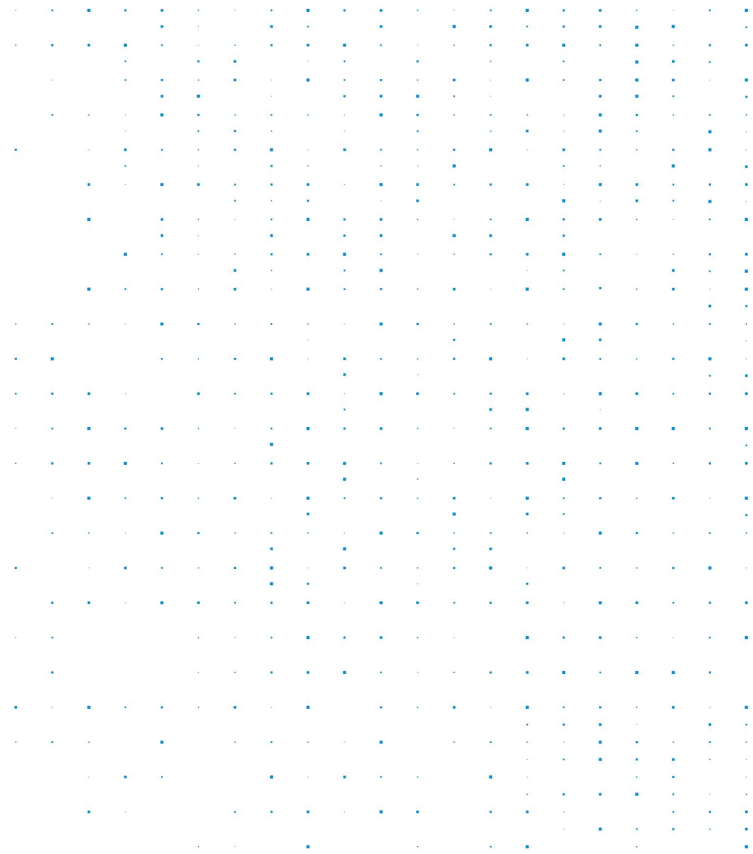
On the robotic arm

- Option to use a smaller scanner for bigger bins and thus get a higher resolution and better details
- Option of variable viewpoints, which comes very handy in case one needs to take a closer look at the corners of the bin
- Contact support@photoneo.com for an advice on L-shaped mount for the scanner



Solution configuration steps

Bin Picking Studio



Solution configuration steps

Solution	✓
Robot	✓
Gripper	✓
Picked object	✓
Gripping points	✓
Grasping methods	
Vision system	✓
Calibration	✓
Environment	
Settings	
Deployment	✓

Solution configuration wizard

- Always follow the steps from top to bottom
- If prerequisites for configuring a section are not met, notification message will inform you about what is missing
- Completed steps will have ✓ next to them

1. Solution page

Detail

AttachmentsActivity log

Name*

Example solution 3: UR5 Demo with 2-finger gripper

Solution unique ID* ⓘ

2

Description

XUndoRedoCopyPasteBoldItalicStrikethroughListBulletedListNumberedListDecrease IndentIncrease IndentLinkUnlinkImageTableHorizontal RuleSource

B I S T^x ¶ ☰ ☲ ☱ Styles Format A A ?

Main robot program: main_program_v4

Solution details and activity

- In the [Solution detail](#) tab you may name your solution, assign it a unique ID and provide a brief description
- [Attachments](#) are designed to store all your project related documents at one place
- In the solution [Activity log](#) tab you can track the changes in your project

2. Configuration of the robot

ABB IRB 1600-10/1.45

ABB



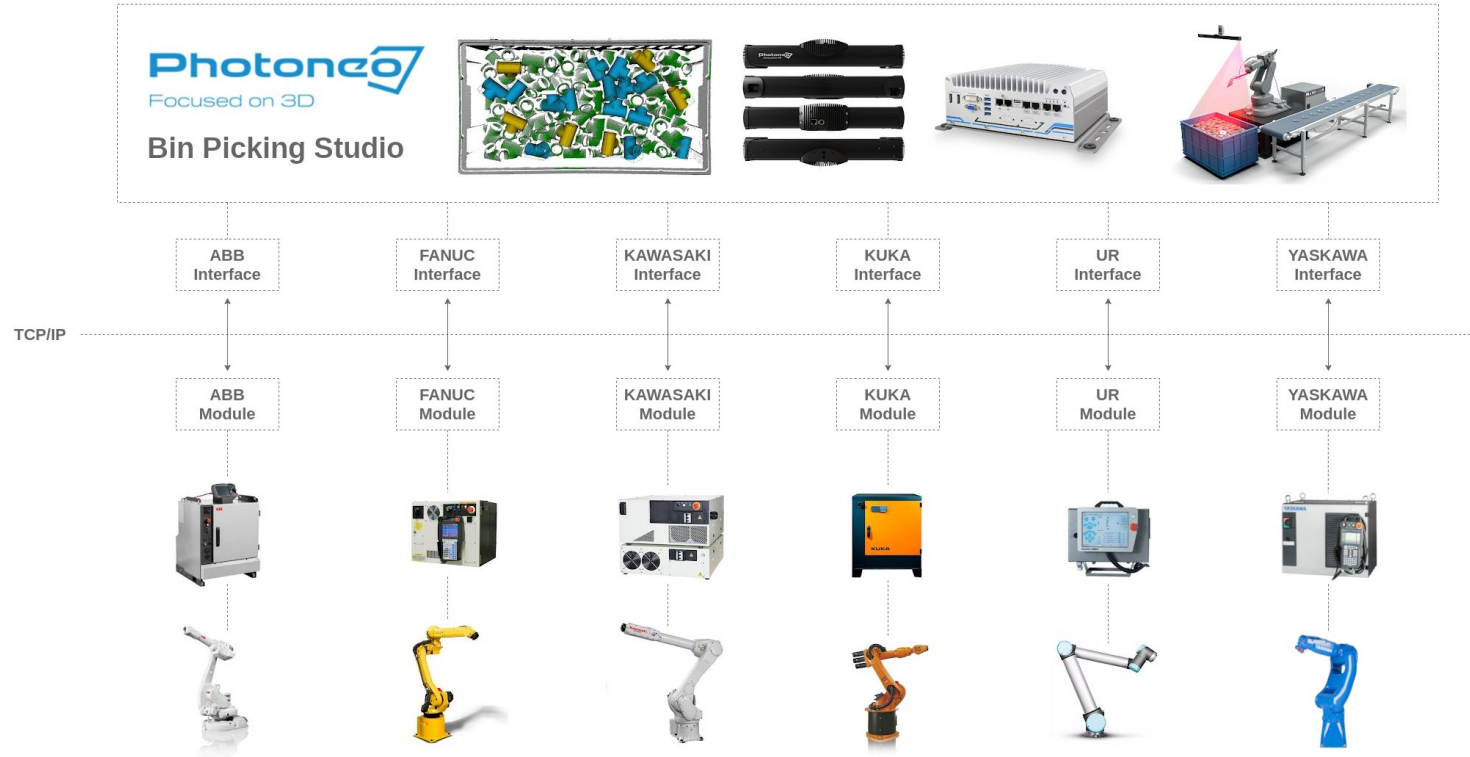
Reach: 1.45 m
Payload: 10.0 kg
Status: ●

Select

Robotic database statuses

- Validated on a real installation with a physical robot
- Model created and tested in the simulation

2. Configuration of the robot



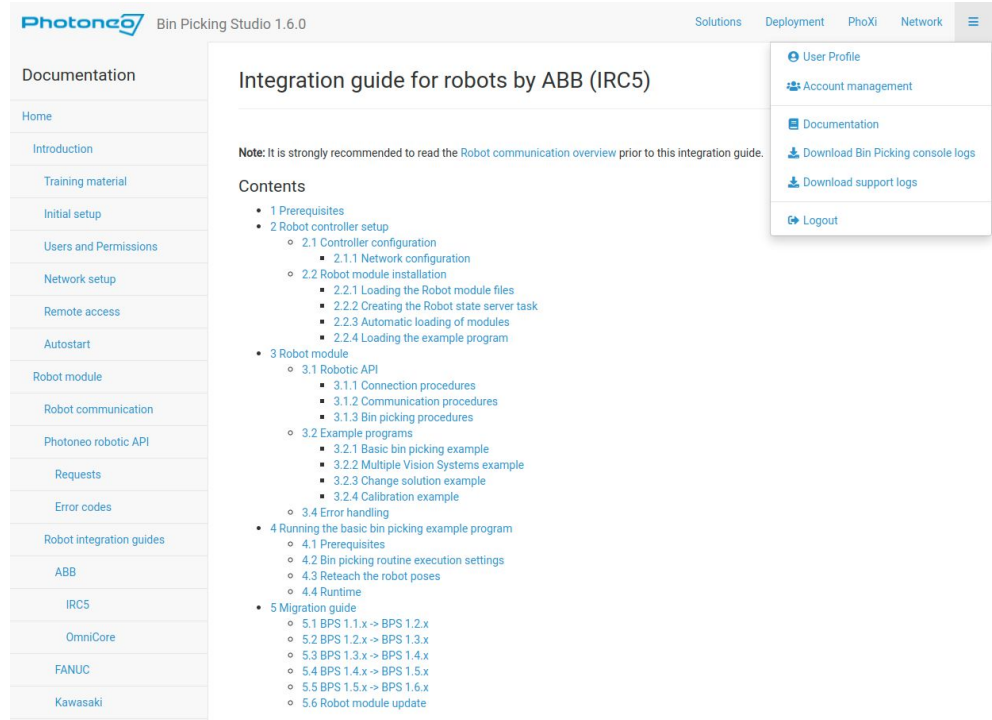
2. Configuration of the robot



2. Configuration of the robot

Robot module

- Specific for each robotic brand
- Source codes for each robotic brand are available upon request at support@photoneo.com
- BPS does not take over control of the robot, it just extends the program library with additional functions
- Integration guide for each robotic brand and robotic API description are in the Documentation section
- Check prerequisites of your robotic brand



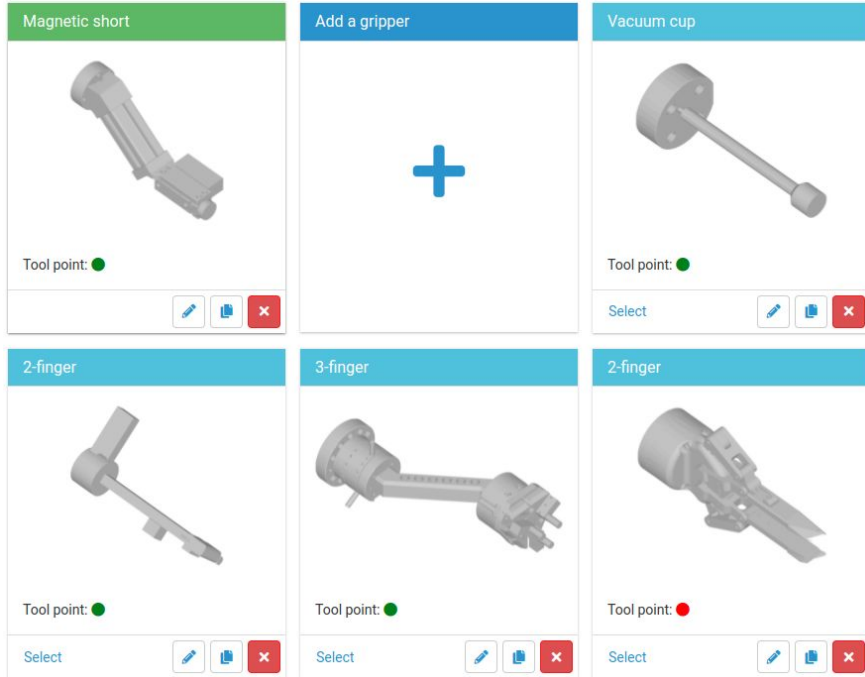
The screenshot displays the Photoneo Bin Picking Studio 1.6.0 interface. The top navigation bar includes links for Solutions, Deployment, PhoXi, Network, and a menu icon. The left sidebar contains a 'Documentation' section with a list of topics: Home, Introduction, Training material, Initial setup, Users and Permissions, Network setup, Remote access, Autostart, Robot module, Robot communication, Photoneo robotic API, Requests, Error codes, Robot integration guides, ABB, IRC5, OmniCore, FANUC, and Kawasaki. The main content area is titled 'Integration guide for robots by ABB (IRC5)'. It includes a note: 'Note: It is strongly recommended to read the Robot communication overview prior to this integration guide.' Below this is a 'Contents' section with a list of topics: 1 Prerequisites, 2 Robot controller setup (with sub-topics 2.1 Controller configuration and 2.2 Robot module installation), 3 Robot module (with sub-topics 3.1 Robotic API and 3.2 Example programs), 4 Running the basic bin picking example program (with sub-topics 4.1 Prerequisites, 4.2 Bin picking routine execution settings, 4.3 Reteach the robot poses, and 4.4 Runtime), and 5 Migration guide (with sub-topics 5.1 BPS 1.1.x -> BPS 1.2.x, 5.2 BPS 1.2.x -> BPS 1.3.x, 5.3 BPS 1.3.x -> BPS 1.4.x, 5.4 BPS 1.4.x -> BPS 1.5.x, 5.5 BPS 1.5.x -> BPS 1.6.x, and 5.6 Robot module update). On the right side of the interface, there is a user profile menu with options: User Profile, Account management, Documentation, Download Bin Picking console logs, Download support logs, and Logout.

2. Configuration of the robot

Robotic program

- Main program
 - Vision controller is the server to which the robotic controller connects to as a client
 - The main robotic program calls functions from the robotic API provided by the Robot module
- State server
 - Robotic controller is the server to which the Vision controller connects to as a client
 - Background routine broadcasting joint states and TCP pose
 - Required for
 - calibration procedure
 - path planning with hand-eye vision systems
 - real-time visualization in Bin Picking Studio

3. Configuration of the tool



Tool design & Tool database

- Defined by a CAD model
- The origin of the model (point $[0,0,0]$ in your file) must be set exactly at the mounting point of the gripper
- The system attaches the origin of the gripper model to the flange of the robot model
 - the Z-axis of the origin of the gripper model must have the same direction as the Z-axis of the flange of the robot model
- Tool point indicator (green/red) tells you whether the gripper has its tool point defined

3. Configuration of the tool

Add a new Gripper

Name*

CAD file (mesh)* ⓘ

Volumetric model (generated automatically) ⓘ

Image

Tool definition

- The tool is defined by CAD model, used:
 - for visualization
 - for definition of the tool point
 - as a collision object
- Volumetric model and image are generated automatically (or loaded manually)

3. Configuration of the tool

Position X [mm]* ⓘ

Position Y [mm]*

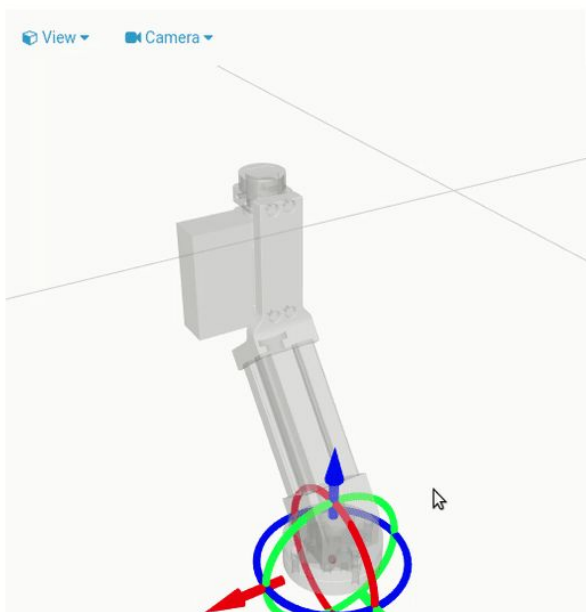
Position Z [mm]*

Rotation X [°]* ⓘ

Rotation Y [°]*

Rotation Z [°]*

☐ Enable rotation invariance ⓘ



Tool point definition

- A point on the tool where the gripper will make contact with the picked object
- Equivalent of TCP defined on robotic controller (Bin Picking Studio does not rely on the TCP definition)
- Definition of position/rotation relative to the origin (robot flange) made by:
 - moving the interactive markers in the visualizer
 - double-click on marker to relative change
 - manual input of a value in the form
- Arrows are pointing in positive direction
- Z-axis should point out of the tool
- **RGB (XYZ)** convention is used

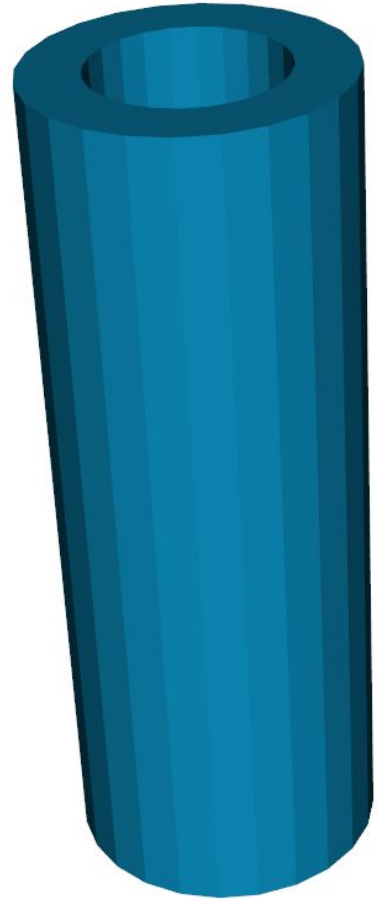
Leveraging of symmetries

Invariance/Symmetry

Consider object with shape of a cylinder and its symmetric axis to be Z-axis. It does matter how the cylinder is rotated when you want to pick from top, or from bottom but it is clearly the same when you rotate your cylinder in Z-axis, when you want to grab it from side. Thus we call cylinder an object invariant for rotation in Z-axis.

Enabling this invariance option, you can set up a series of points along any axis or combination of axes. Use slider in visualizer to preview the generated configurations.

Remember, a tool point can have its own invariance and also gripping point may have its invariance.



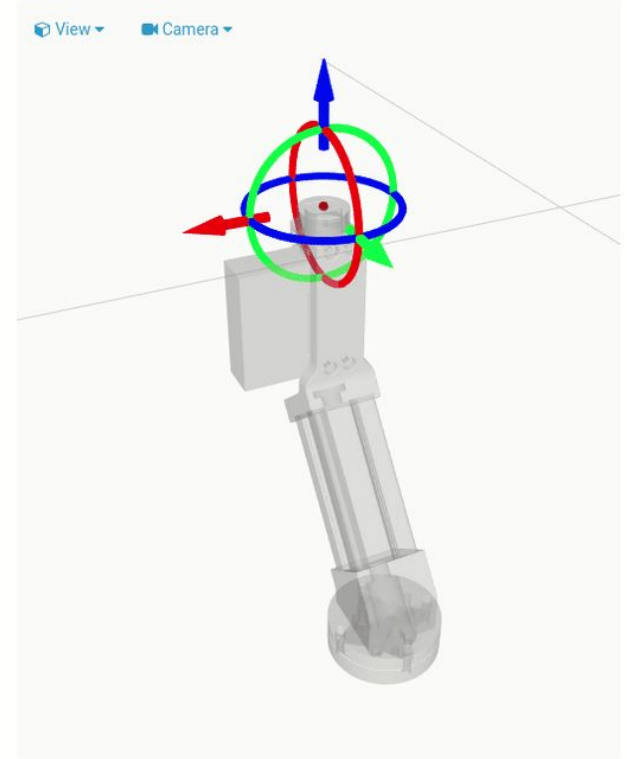
3. Configuration of the tool

Tool point invariance

- Definition of the invariance of the tool in the selected axis. Basic mode is recommended for initial testing
- Range of the invariance and number of the steps within the range
- Higher number of invariances increases the chance of successful path planning but increases the calculation time

☐ Enable rotation invariance ⓘ

Save



4. Configuration of the picked object

			☐	☐
Name	Image	Actions		
CogWheel		Edit Delete		
Cylinder		Edit Delete		
Object		Edit Delete		
TFit		Edit Delete		
Showing 1 to 4 of 4 entries			Previous	Next

Picked object definition

- The object is defined by CAD model
- Up to 4 objects can be loaded
- Later the object is linked to a Vision system
- Object is used:
 - for visualization
 - for definition of gripping points
 - to teach localization engine
 - as a collision object

4. Configuration of the picked object

Name*

Description

CAD file (mesh)* ⓘ

View ▾ Camera ▾ Thumbnail ▾



Volumetric model (generated automatically) ⓘ

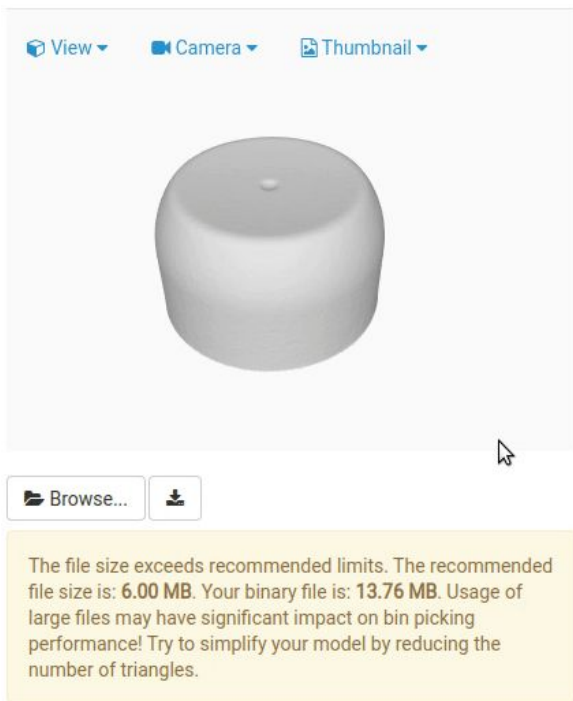
✓ Successfully generated

Image

Picked object definition

- Volumetric model and image are generated automatically (or loaded manually)
- Always provide a clear description of the model or version of the picked object, it may be useful when looking at older projects

4. Configuration of the picked object



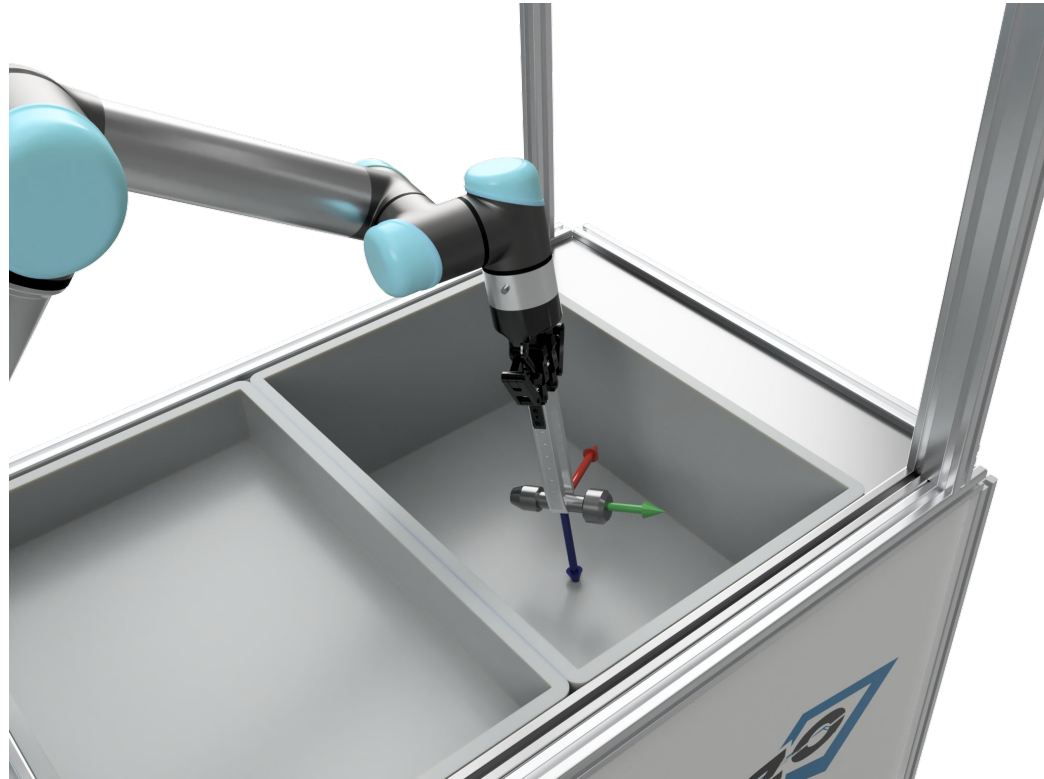
What to look out for

- Size of the CAD model - the smaller the better
 - warning is displayed when the file exceeds recommended size
 - complex models do not provide much benefit and prolong computation times
- Unclosed models - generated volumetric model might be incorrect
 - as can be seen in the visualizer, the model of the lid is invalid

5. Definition of gripping points

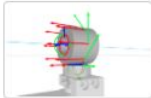
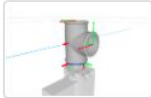
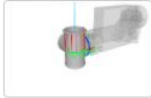
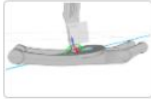
Definition of the grasp

Overlay of the coordinate system of the tool with the coordinate system of the object.



5. Definition of gripping points

Existing gripping points

ID	Thumbnail	Picked object	Priority	Active	Invariances Enabled/Total	Actions
1.		 CogWheel	✓ Normal	✓ Enabled	2 / 2	Edit Duplicate Delete
2.		 Cylinder	✓ Normal	✓ Enabled	5 / 8	Edit Duplicate Delete
3.		 TFit	↑ High	✓ Enabled	3 / 3	Edit Duplicate Delete
4.		 TFit	✓ Normal	⊘ Disabled	9 / 9	Edit Duplicate Delete
5.		 Object	✓ Normal	✓ Enabled	2 / 2	Edit Duplicate Delete

Gripping points

- Multiple objects supported
- Multiple Gripping points supported
- Option to disable gripping point for testing purposes
- Thumbnail for better orientation (large preview shown when hovered over with mouse cursor)

5. Definition of gripping points

Gripping point definition

Unique ID*

Priority*

Position X [mm]*

Position Y [mm]*

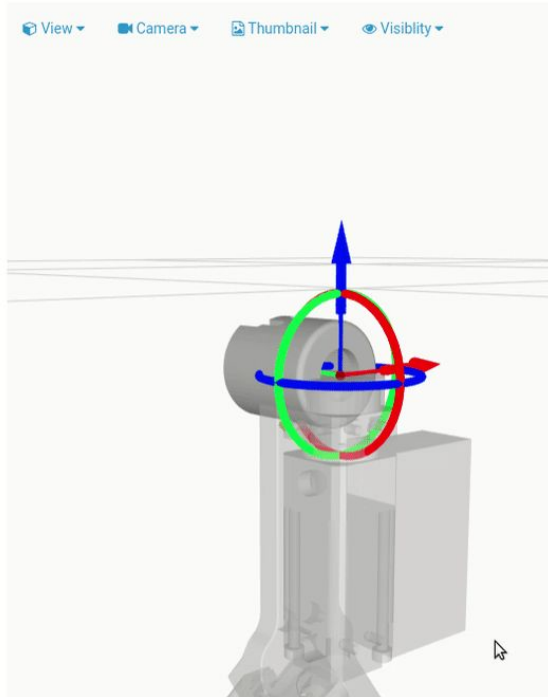
Position Z [mm]*

Rotation X [*]*

Rotation Y [*]*

Rotation Z [*]*

☐ Enable rotation invariance ⓘ

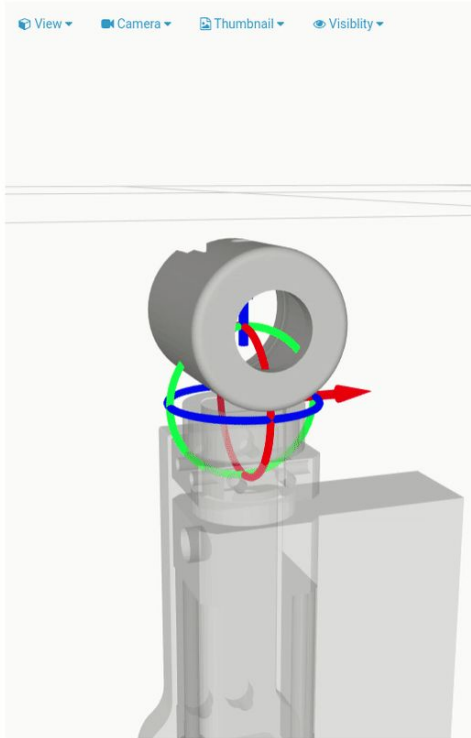


- A point on the picked object where it will be grasped
- Definition of position/rotation relative to the origin made by:
 - moving the interactive markers in the visualizer
 - double-click on marker to relative change
 - manual input of a value in the form
- Arrows are pointing in positive direction
- **RGB (XYZ)** convention is used
- Option to assign a priority (Low, Medium, High)
 - Algorithm will start to evaluate the gripping point of targeted object with High priority first
 - In case of unsuccessful calculation it will continue with Medium and Low at last

5. Definition of gripping points

☐ Enable rotation invariance ⓘ

Save



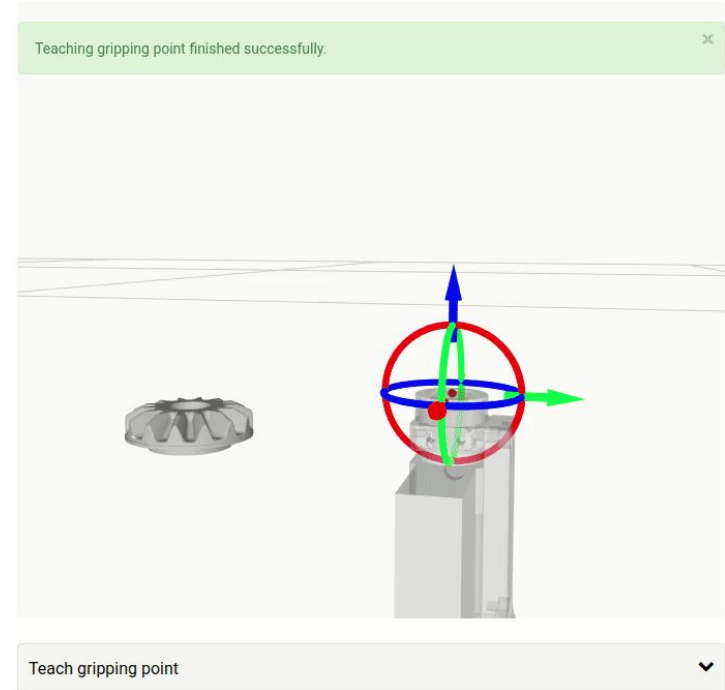
Gripping point invariance

- Definition of the invariance of the picked object in the selected axis. Basic mode is recommended for initial testing
- Range of the invariance and number of the steps within the range
- Relative position of axis of invariance to origin of picked object
- Higher number of invariances increases the chance of successful path planning, but increases the calculation time

5. Definition of gripping points

Teach gripping point

- Grip the object with the tool
- Jog the robot into the scanning area of the scanner
- Configure overlap and localization timeout
- Review found objects and select the one meeting the criteria
- Prerequisites:
 - tuned localization profile
 - successful calibration
- Unsuitable for grippers with suction cups
- **Practical use:** Suitable for bigger objects and complex tools, e.g. big metal sheets gripped by the tool consisting of several magnets



Achieving oriented placing

Teaching of placing position

- In case of oriented placing requirement, it is necessary to teach a placing position for every symmetry
- Path planning is done between the **Start** waypoint and the **End** waypoint
 - More information in the following chapter
- Placing position is handled in the robot program
- robotic API call **receive trajectory** will provide the robot with information about the tool point invariance, gripping point ID and gripping point invariance
- While teaching, disable those gripping points (or their invariances) and tool point invariances, for combination of which you have a placing position already taught, BPS will stop allowing these grasping combinations so you can teach the rest of them

Rotation lower limit [°]*

0

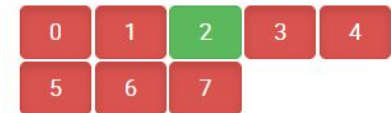
Rotation upper limit [°]*

360

Rotation number of steps*

8

Invariance list



✓ Enable all

⊗ Disable all

Save

6. Grasping method

New grasping method

Name*

+ Add new grasping method

Existing grasping methods

#	Name	Vision system			
1	Basic Grasping Method default	ID: 1	Object detector 1	 Edit	 Duplicate
		ID: 2	Object detector 2		
2	Deapproach towards Start pose	ID: 3	Object detector 3	 Edit	 Duplicate  Set default  Delete

Strategy for the object extraction

- defined as a sequence of movements and gripper operations - [bin picking routine](#)
- option to define up to 5 unique grasping methods
- a grasping method is associated with a vision system (next chapter)
- one grasping method is marked as [default](#)

6. Grasping method

Waypoint

- A point in space defined in fixed position or relative to the preceding waypoint
- Setting a reasonable waypoints can help the path planning to be more successful and time efficient
- Mandatory: Start, Approach, Grasp, Deapproach, End

Trajectory

- A sequence of points in space computed by the path planner joining two consecutive waypoints
- Joint-type or linear (trajectories between waypoints [Approach](#) and [Deapproach](#) are always linear)

Edit grasping method

Name*

Basic Grasping Method

default

Save

Used in Vision systems:

- ID: 1 Object detector 1
- ID: 2 Object detector 2

Picking Path Stages

#	Name	Actions	Trajectory Type	Position Tolerance
1	Start	Edit	N/A	N/A
2	Approach	Edit	Joint	Approx.
3	Grasp	Edit	Linear	Precise
4	Deapproach	Edit	Linear	Approx.
5	End	Edit	Joint	Precise

+ Add a new picking path stage

6. Grasping method

Start

- Starting waypoint of the bin picking routine
- Defined in the robotic controller, transferred to the Bin Picking Studio via call [Initialization request](#) from robotic API
- Usually defined above the center of the bin

Type*
Start

Procedure name ⓘ

Save



6. Grasping method

Approach

- The gripper is moved to a pose defined relatively to the located gripping point on the picked object
- In most cases, the origin of the offset is **Found gripping point** - the relative offset is set to several centimetres in the opposite direction of the gripping point's Z-axis
- Available Approach strategies:
 - Relative to Robot base
 - Relative to Found gripping point



Type*
Approach

Origin of relative offset ⓘ*
Found gripping point

Relative offset X [mm] *
0.0

Relative offset Y [mm] *
0.0

Relative offset Z [mm] ⓘ*
-110.0

☐ Plan linear trajectory ⓘ

Position tolerance ⓘ*
Approximative

Procedure name ⓘ

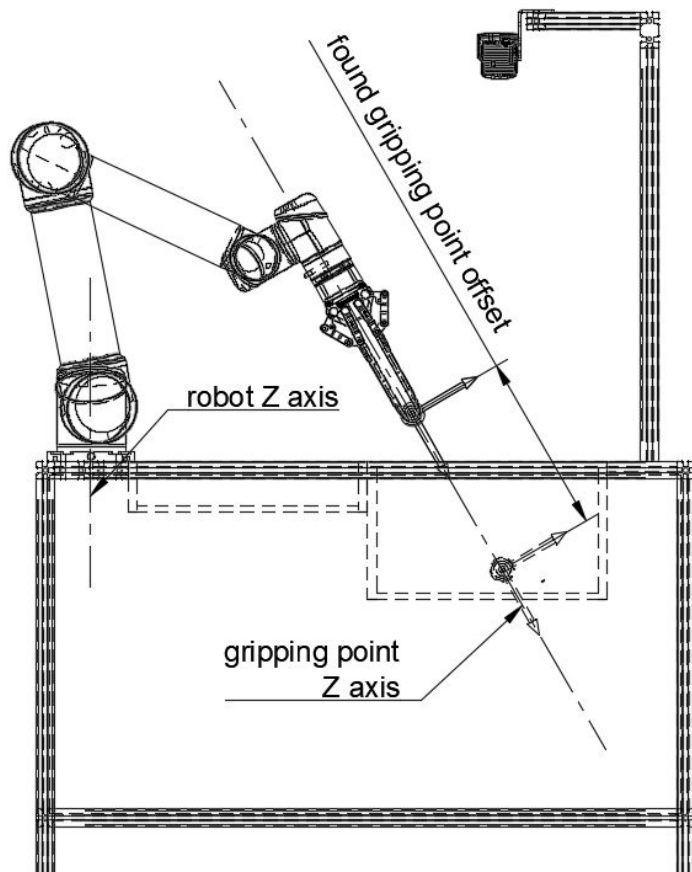
Save

6. Grasping method

Relative offset

Origin of relative offset: Found gripping point

The user defines an offset $[X, Y, Z]$ relative to a found gripping point.

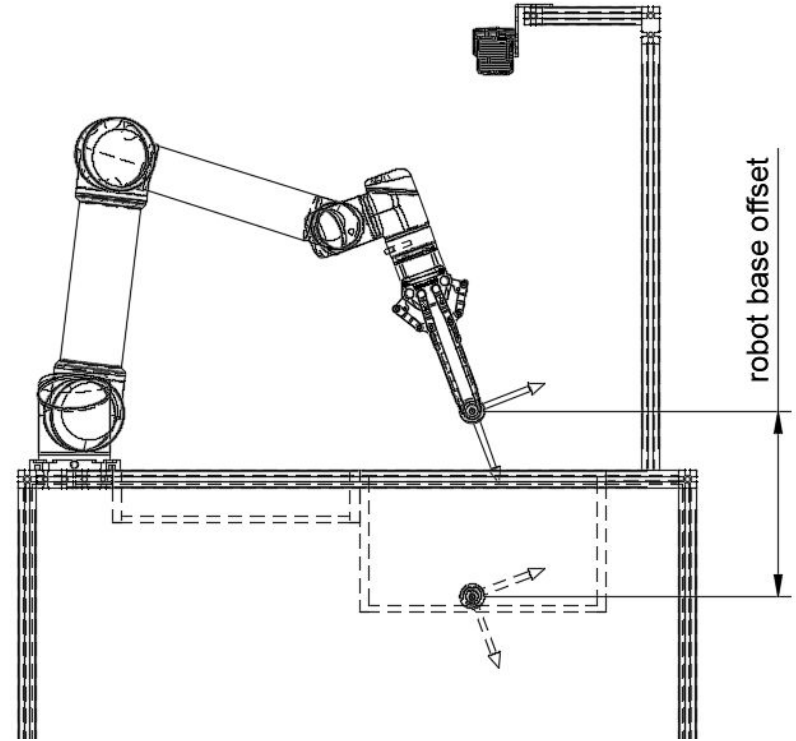


6. Grasping method

Relative offset

Origin of relative offset: [Robot](#)

The user defines an offset $[X, Y, Z]$ relative to the base of the robot.



6. Grasping method

Grasp

- Located gripping point on the picked object
- Trajectory to this waypoint is always linear
- Procedure to be executed in this waypoint is the procedure in the robotic program in which you activate the gripper - usually [Attach](#)



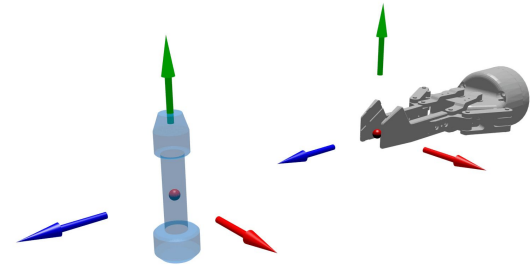
Type*
Grasp

☒ Plan linear trajectory ⓘ

Position tolerance ⓘ*
Precise

Procedure name ⓘ
Attach

Save



6. Grasping method

Deapproach

- The gripper is moved to a pose defined relatively to the located gripping point on the picked object
- In most cases, the origin of the offset is **Robot** - the relative offset is set to several centimetres in the direction of the Z-axis of robot space
- Available Deapproach origins:
 - Relative to Robot base
 - Relative to Found gripping point
 - To fixed plane
 - Relative to Start waypoint
 - Relative to End waypoint



Type*

Deapproach

Origin of relative offset ⓘ*

Robot

Relative offset X [mm] *

0.0

Relative offset Y [mm] *

0.0

Relative offset Z [mm] ⓘ*

110.0

☒ Plan linear trajectory ⓘ

Position tolerance ⓘ*

Approximative

Procedure name ⓘ

 Save

6. Grasping method

Relative to Found gripping point

Distance in negative direction of Z-axis of the found gripping point



Relative to robot

Distance in positive direction of Z-axis of Robot space



To Fixed plane

Plane defined in an absolute distance from the origin of robot's base.
Trajectory length is variable.

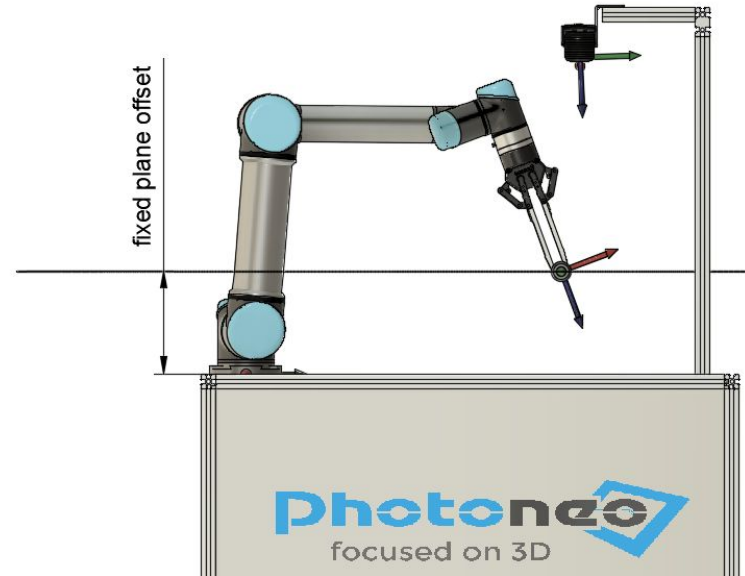


6. Grasping method

Relative offset

Origin of relative offset: **Fixed plane**

- Fixed plane is parallel with the robot base plane defined by X and Y-axis of robotic coordinate system.
- Relative offset value [mm] defines offset of the plane from the base plane in Z-axis
- The target waypoint is defined during each cycle as having the X, Y coordinates of the previous waypoint and the Z coordinate specified by the plane offset.
- **Practical use:** Solving problem with deep bins when you want to have Deapproach waypoint out of the bin for every part. With constant Deapproach defined in Robot space an object picked from the bottom of the bin will be at the edge of the bin in Deapproach, but an object picked from the upper layer will be much higher than the bin edge, possibly even out of the robot's reach.

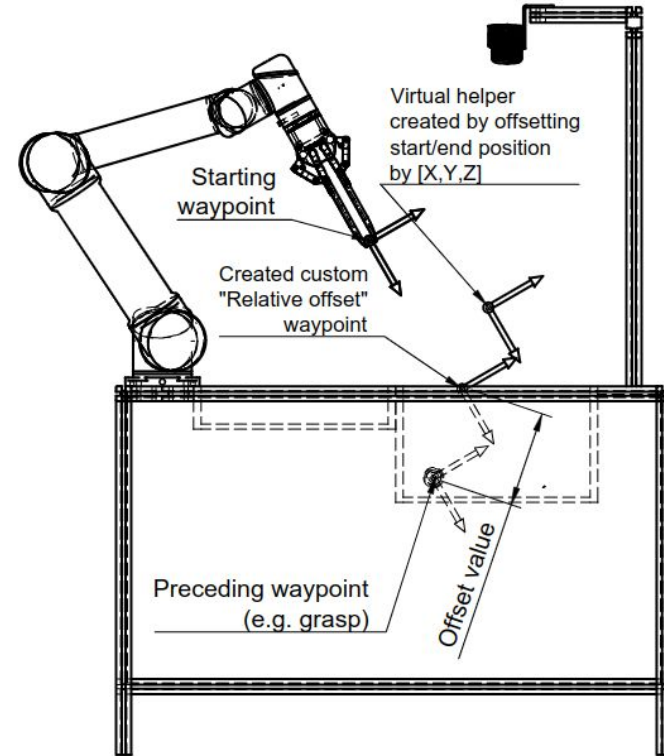


10. Grasping method

Relative offset

Origin of relative offset: **Start** or **End** waypoint

- The user may specify an offset strictly in relation to the Start or End position as specified in the robot program
- **Practical use:** Movement away from the bin walls, towards the middle of the bin (the Start waypoint)



10. Grasping method

Relative offset

Origin of relative offset: Start or End waypoint

Example:

- Start waypoint is shifted 200 mm in the direction of X-axis (Robot space)
- Imaginary line between the previous waypoint and the offsetted Start point defines the direction (previous waypoint in direction from Grasp waypoint - for a new waypoint between Start and Approach the previous one is Approach)
- Relative offset value defines the length of the trajectory (0 - new waypoint is the same as the offsetted start/end waypoint)

Type*
Relative

Name*
Direction of extraction

Origin of relative offset ⓘ*
Start

Relative offset X [mm] *
200

Relative offset Y [mm] *
0

Relative offset Z [mm] ⓘ*
0

Relative offset value [mm] ⓘ*
0

☐ Plan linear trajectory ⓘ

Position tolerance ⓘ
Approximative

Procedure name ⓘ

Save

6. Grasping method

End

- Ending waypoint of the bin picking routine
- Defined in the robotic controller, transferred to the Bin Picking Studio via call [Initialization request](#) from robotic API
- Usually defined above the center of the bin or slightly towards the placing position



Type*

End

☐ Plan linear trajectory ⓘ

Position tolerance ⓘ*

Approximative

Procedure name ⓘ

 Save

6. Grasping method

Custom fixed point

- Fixed point defined by $[X, Y, Z]$ in the robot space
- In this case, the position of the waypoint is specified relative to the robot base as a fixed point in 3D space (Absolute position X, Y, Z coordinates).
- The robot arrives to this point with the gripper oriented the same way as it was in the previous waypoint.

Type*
Fixed

Name*
Fixed point

Absolute position X [mm] ⓘ*
200

Absolute position Y [mm] ⓘ*
0

Absolute position Z [mm] ⓘ*
200

☐ Plan linear trajectory ⓘ

Position tolerance ⓘ*
Approximative

Procedure name ⓘ

Save

7. Vision system

Object detector 1	ID: 1
Scanner ID: UserExamples-example-camera-1	
Scanner model: L	
Scanner mount position: Robotic cell	
Picked object: CogWheel	
Grasping method: Basic Grasping Method	
Localization profile file: Object_detector_extrinsic_generated_sqxNp.plcf	
Gripper collision sensitivity [%]: 5.0	
Use fast scan: Off	
Calibration:	
Edit	Duplicate Delete

Object detector 2	ID: 2
Scanner ID: UserExamples-example-camera-1	
Scanner model: L	
Scanner mount position: Robotic cell	
Picked object: CogWheel	
Grasping method: Deapproach towards Start pose	
Localization profile file: Object_detector_extrinsic_generated_A0jQr5.plcf	
Gripper collision sensitivity [%]: 5.0	
Use fast scan: Off	
Calibration:	
Edit	Duplicate Delete

Add vision system

Vision system (VS)

- Single VS represents:
 - Scanner
 - Calibration
 - Picked object
 - Localization profile
 - Grasping method
 - Several settings
- Up to 4 Vision systems supported
- Deployment requests from robotic API refer to the ID of the VS

7. Vision system

Configuration - part I.

- Name the vision system
- Generated unique ID can be changed
- Associate with a scanner
 - one scanner can be associated with multiple VS
- Select scanner model
 - the model is inserted into the environment as collision object based on calibration
- Select scanner mount position
 - the mount position determines the type of calibration (extrinsic / hand-eye)

Name*

Unique ID*

Scanner ID* ⓘ

Available scanners ▼

Scanner model*

PhoXi® 3D Scanner L ▼

Scanner mount position*

Robotic cell ▼

7. Vision system

Configuration - part II.

- Link a picked object
 - one picked object can be linked with multiple vision systems
- Grasping method
 - choose specific one or the one marked default
- Localization profile - upload a PLCF (Photoneo Localization Configuration File) or create a new one
- Place where the calibration matrix is stored

Picked object*

CogWheel

Grasping method

Basic Grasping Method

Localization profile* ⓘ

[+ Create new](#) [Upload other](#) [Edit current](#) [Download](#)

Calibration matrix

0.986253	0.033168	-0.161879	260.738880
0.034027	-0.999418	0.002537	610.362953
-0.161701	-0.008011	-0.986807	1365.000692
0.000000	0.000000	0.000000	1.000000

[Edit matrix](#) [Copy to clipboard](#)

7. Vision system

Vision system settings

Some settings are unique for each VS:

- **Gripper collision sensitivity**
 - how much volume of the gripper is allowed to collide with the scan
- **Fast scan settings**
 - definition of full scan frequency and conditions for its execution, other scans are fast - only texture is captured and used for object pose verification
 - collision checking is done with old scan!

Gripper collision sensitivity [%] <0; 100>* ⓘ

5.0

On

Use fast scan ⓘ

Full scan frequency <2; 10>* ⓘ

3

Fast scan minimum objects found <1; 20>* ⓘ

3

Is-change threshold <0; 2>* ⓘ

0.5

Changed pixels fraction threshold [%] <0; 100>* ⓘ

5.0

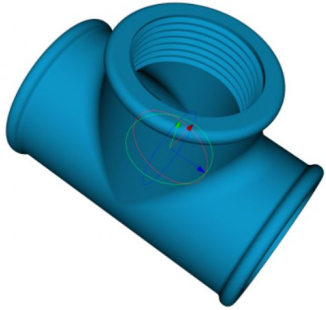
Minimum ready-to-pick objects <0; 10000>* ⓘ

1

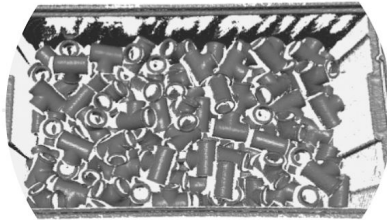
7. Vision system

CAD-based localization engine

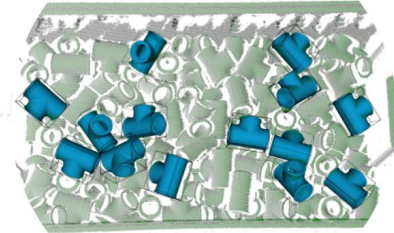
Recognition of object's pose (position and orientation) in the scene



Insert CAD model

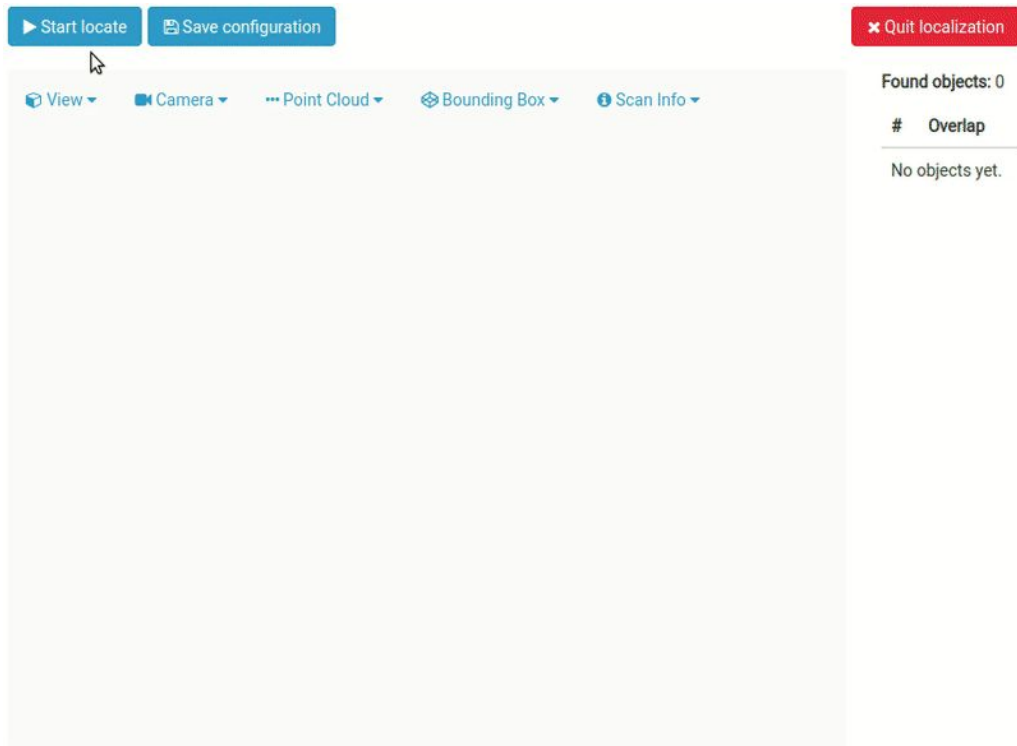


Capture 3D scene



Get localized results

7. Vision system



Photoneo Localization Configuration File (PLCF)

- Assign existing PLCF to the VS or configure and generate a new one
- Ensure a proper localization results before deployment of bin picking solution
- Segmented image (shades of green), is crucial for proper configuration of the localization settings
- For offline support please provide PLCF and several scans in PRAW format

7. Vision system

PLCF parameters

- Explanation of each parameter is inside the tooltip
- We recommend to change the Overlap to value higher than 15%
- For more precise results lowering [Global maximum feature fit overflow](#) and [Projection tolerance](#) can be of help

Timeout [ms] <1; 9999999>* ⓘ 10000	Overlap [%] <1; 100>* ⓘ 15.0	Matching algorithm* ⓘ Surfaces	Scene clustering level* ⓘ Normal
--	--	--	--

Advanced settings

Global maximum feature fit overflow <0; 100>* ⓘ

Feature fit consideration level <0; 100>* ⓘ

☐ Off Use the Empty scene feature ⓘ

Model keypoint sampling* ⓘ

Scene minimal cluster size <0; 9999999>* ⓘ

Scene maximal cluster size <0; 9999999>* ⓘ

☒ On Scene noise reduction ⓘ

Local search radius* ⓘ

Projection tolerance <0; 100>* ⓘ

Projection hidden part tolerance <0; 100>* ⓘ

Fine alignment point set* ⓘ

Fine alignment point ⓘ

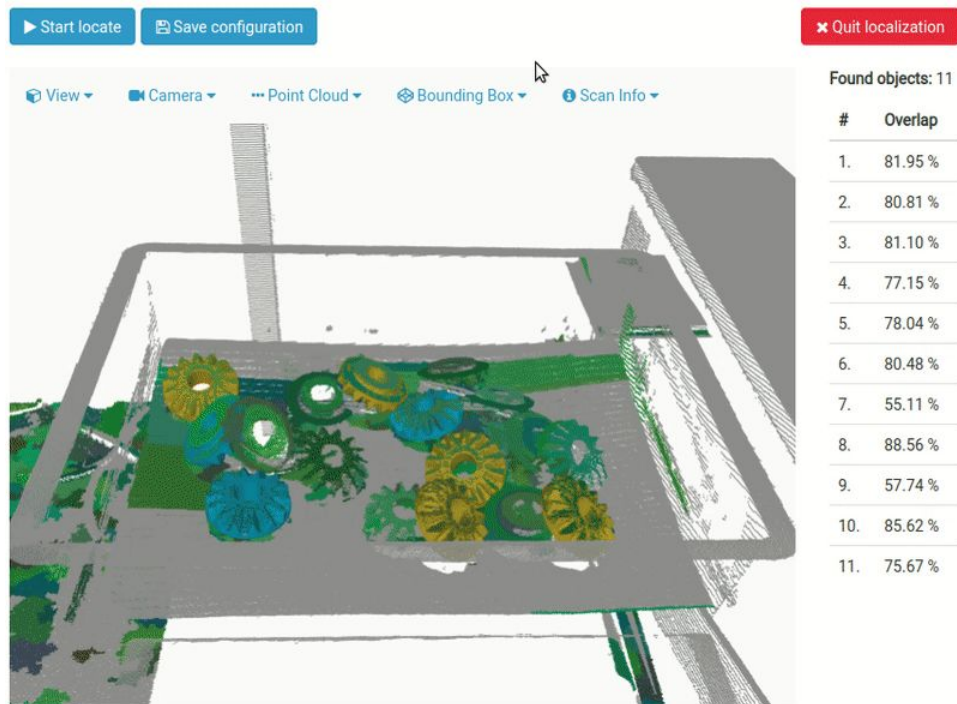
Fine alignment iterations <1; 1000>* ⓘ

The number of iterations of the fine alignment algorithm. The recommended range is <6, 30>. Higher values might slow down localization performance.

7. Vision system

Localization Bounding Box

- Bounding box defines area of interest for localization
- The Localization GUI allows definition of a box where the detection will take place, leading to higher speed and increased safety
 - Speeds up the localization by selecting candidates only from the inside of the box
 - Object has to have at least some overlay with the bounding box to be localized



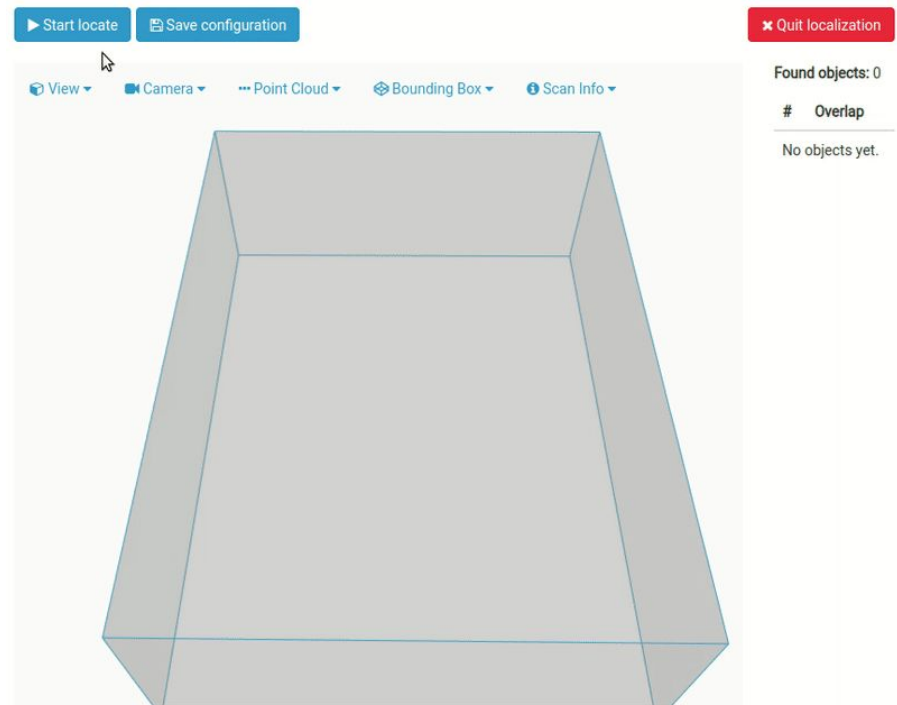
7. Vision system

Empty scene

- Report to robot that all objects have been picked
- Allow the feature and set the threshold (number of points in the whole scene or in a bounding box)



- We recommend to use a bounding box which does not include the bottom of the bin



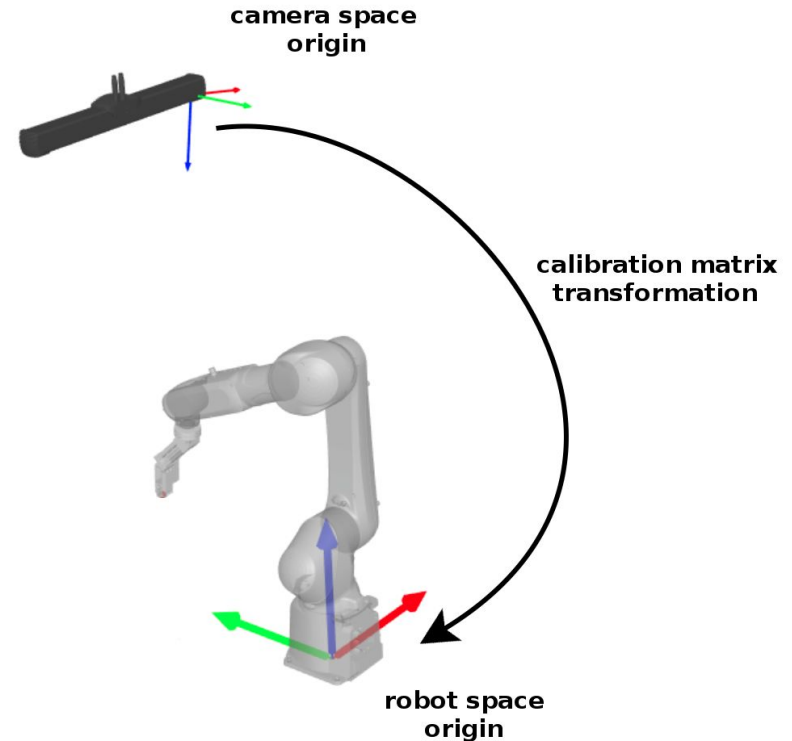
8. Calibration

Robot to Camera calibration

- Camera space
 - Origin in the centre of the sensor chip
- Robot space
 - Origin in the robot base

Transformation of the origin of the point measured in the camera space to the robot space.

After calibration the localization results are calculated directly in robot base coordinates.



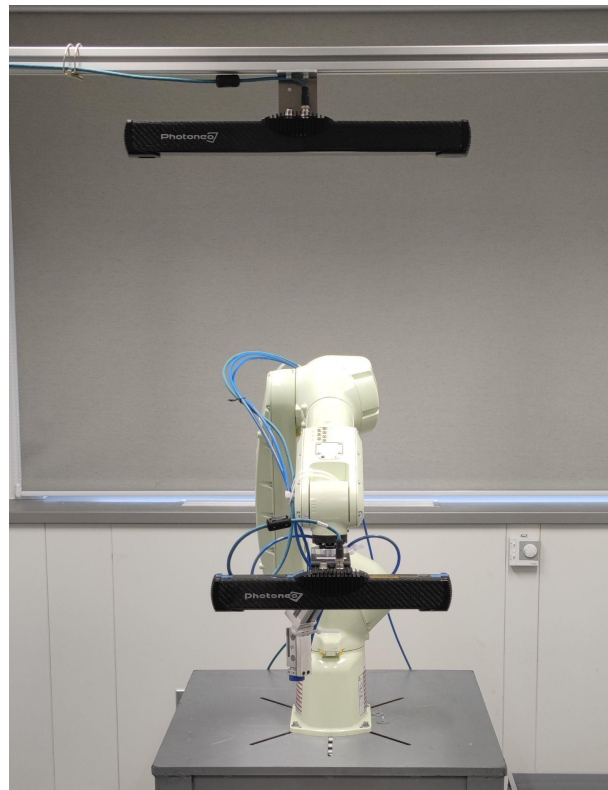
8. Calibration

Extrinsic calibration

- Scanner in static position
- Transformation from camera to the robot ball

Hand-eye calibration

- Scanner on the robotic arm - behind 6th joint
- Transformation from camera to the robot's flange
 - Transformation to the robot base depends on robot's pose during scan acquisition



8. Calibration

Requirements

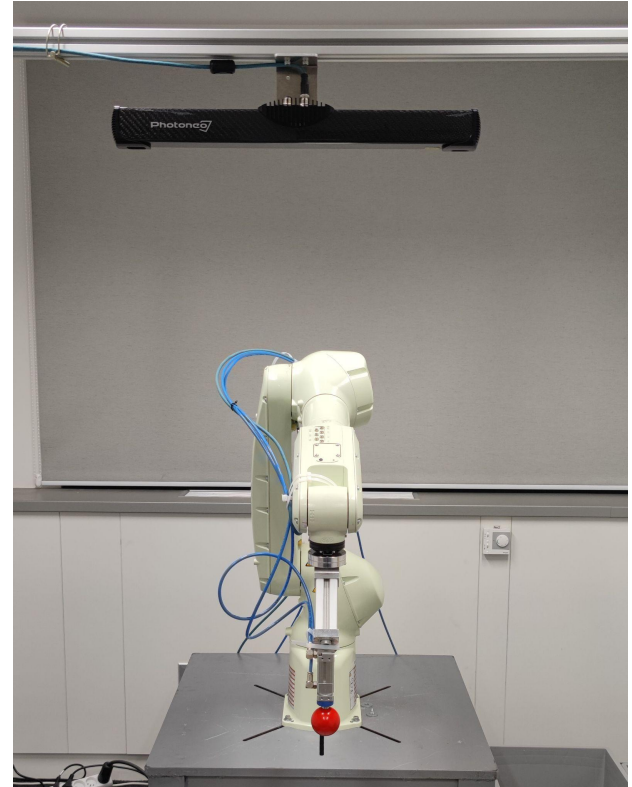
- The Vision system must be configured
- The scanner must be connected to the Vision controller
- Robot State Server task must be running - refer to the Integration guide for your robotic brand

ID	Name	Scanner ID	Scanner Status	Scanner model	Calibration type	Calibrated	Calibration accuracy [mm]	
14	Object detector	JKE-055	Available	PhoXi® 3D Scanner L	Extrinsic	✓	N/A	 Start calibration

8. Calibration

Extrinsic calibration - Preparation

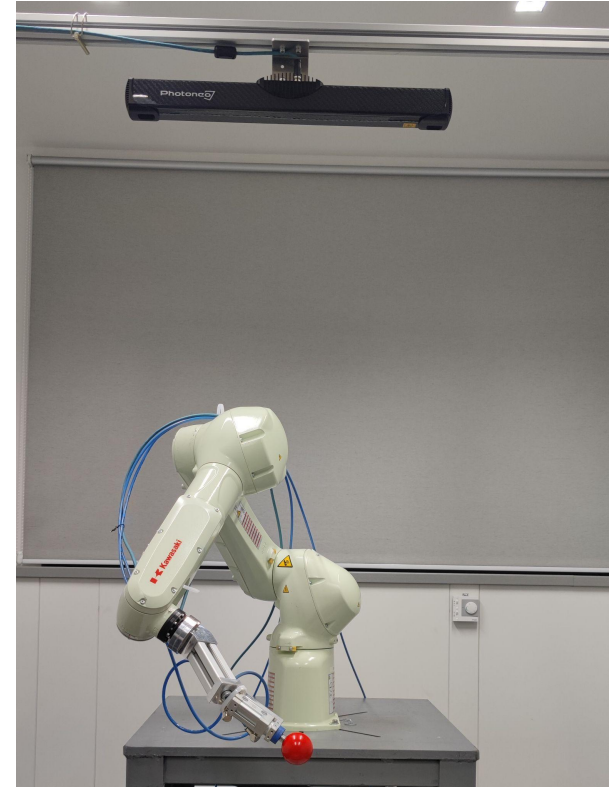
- Calibration tool - a precisely manufactured ball with smooth surface
 - we recommend to use a ping-pong ball, snooker ball
 - use bigger balls with bigger scanner models
- Measure the exact radius of the calibration ball - you will be asked for it at the beginning of the calibration process
- Mount the ball on the tool, beyond the 6th joint
- Remove the bin before calibration, it will provide more space for better overall accuracy in the volume of the bin
- TCP cannot be changed during calibration



8. Calibration

Extrinsic calibration - Poses

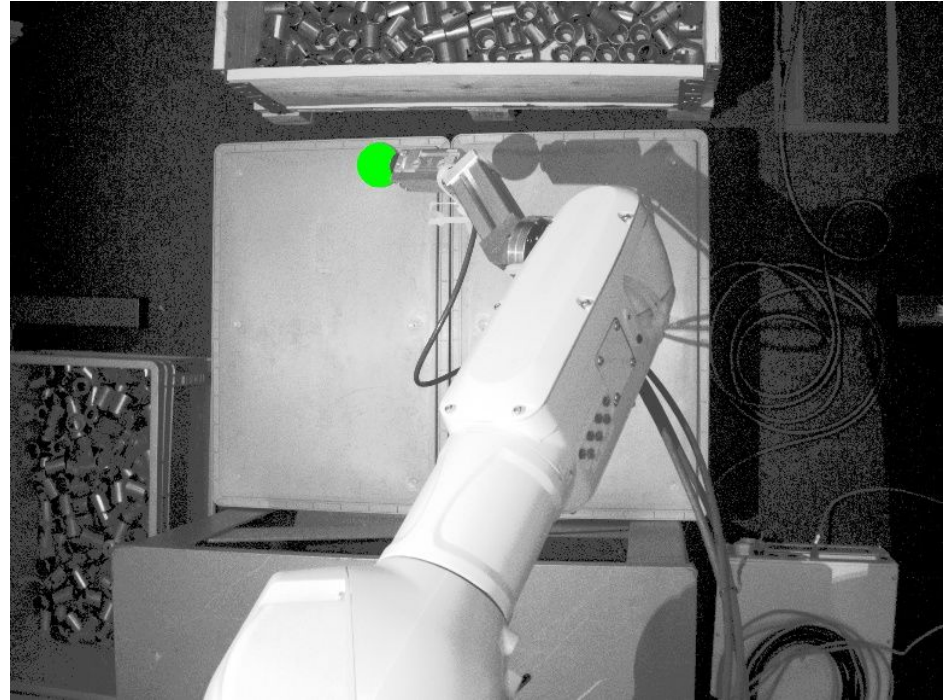
- Log the robot to capture various distinct poses
- Try to position the ball in all corners of the bin
- Rotate the joints of the robot as much as possible
- Make sure the ball is clearly visible on the scan
- Robot must be perfectly still while adding a new point
- The calibration ball must not move during the calibration process



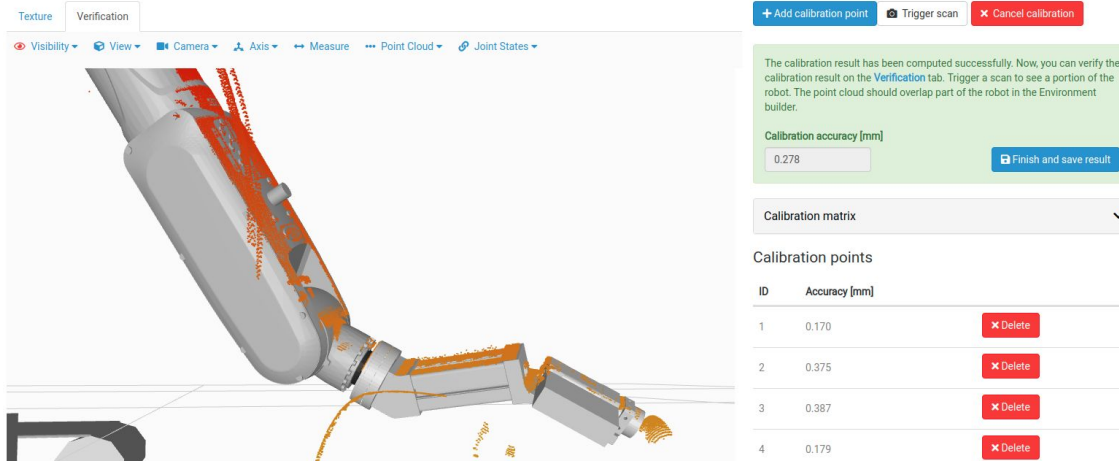
8. Calibration

Extrinsic calibration - Adding points

- Use [Add calibration point](#) button to add a new point
- Detected ball is visualized in the texture image
- Visually check if the calculated position of the calibration ball corresponds to its actual location
- 9 calibration points are mandatory
- If a point fails to be added, error message appears
- A point too similar to one of the previous ones is rejected
- Significant part of the ball's surface must be visible on the scan for successful detection



8. Calibration



Extrinsic calibration -
Verification & Finalization

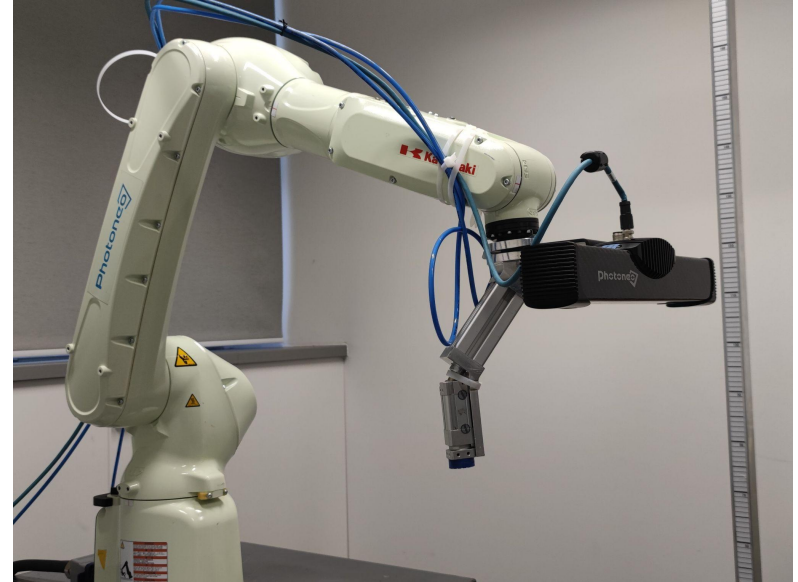
- After capturing at least 4 calibration points the point cloud is available in the 3D visualizer on the [Verification](#) tab
- Validate the overlay of the robot model with the point cloud
- Aim for the highest accuracy in the area of the bin and the tool
- Save the result by clicking the button [Finish and save result](#)

8. Calibration

Hand-eye calibration - Preparation

- Calibration tool - marker pattern provided by Photoneo, available for download [here](#)
- Position the pattern inside the bin or in its place
- The pattern must be static during the calibration procedure
- Scanner is mounted behind 6th joint
- TCP must not be set - a warning is displayed if tool0 is not used

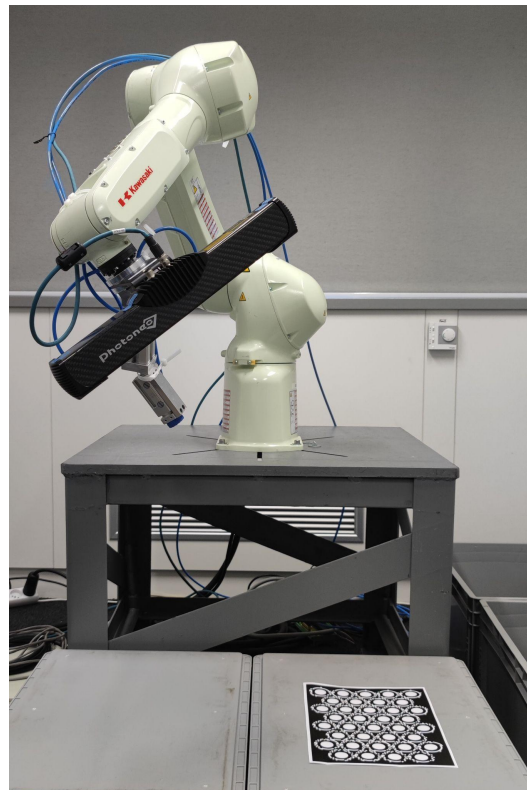
⚠ The difference between the tool center point (TCP) reported by the real robot and by the virtual robot model is too large. To prevent incorrect calibration, please make sure that the robot reports its correct flange position.
Tip: This mismatch may be caused by a **custom tool** set up in the real robot or by an incorrect **base frame offset**.
Translation error: 0.0525 m.
Rotation error: 0.0047 rad (0.27 degrees).



8. Calibration

Hand-eye calibration - Poses

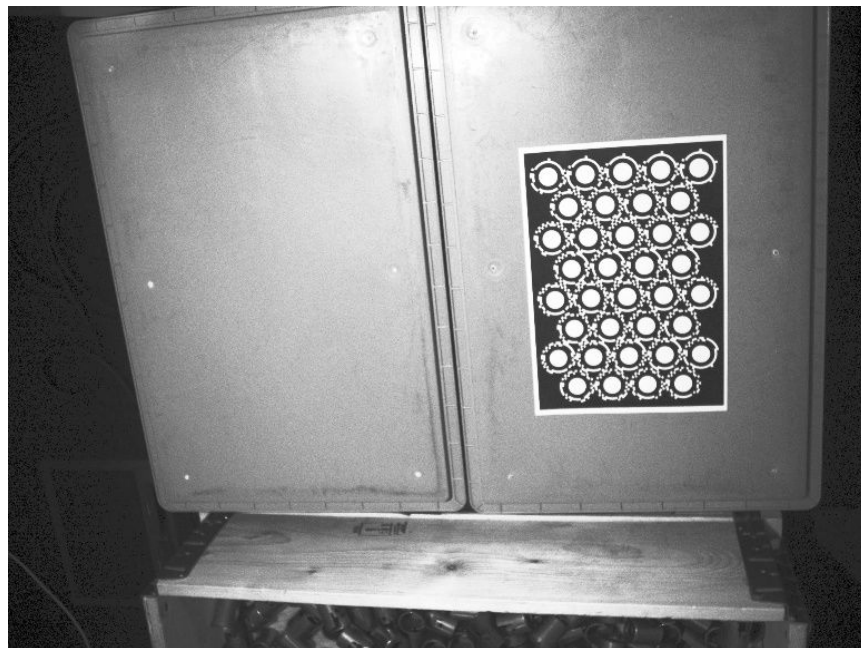
- Jog the robot to capture the marker pattern from various distinct poses
- Rotate the joints of the robot as much as possible
- Make sure the marker pattern is clearly visible on the scan
- Robot must be perfectly still while adding a new point



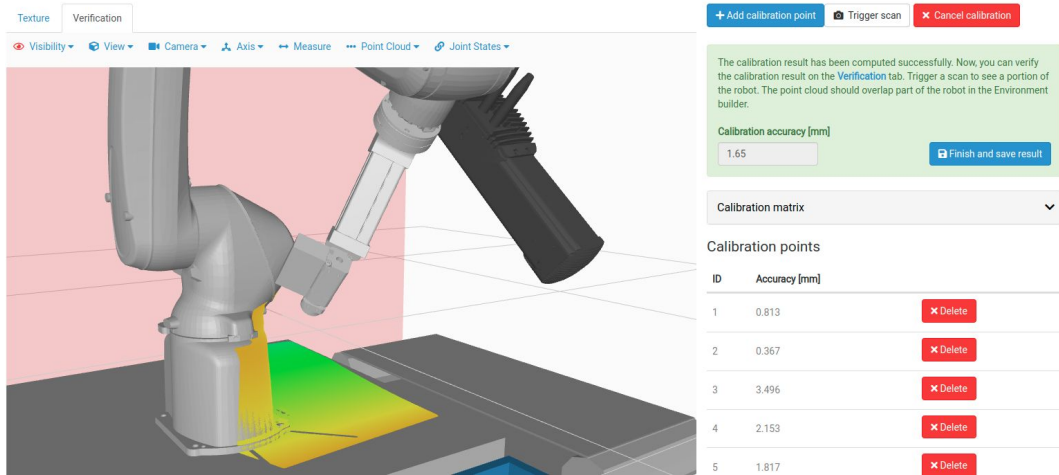
8. Calibration

Hand-eye calibration - Adding points

- Use [Add calibration point](#) button to add a new point
- 9 calibration points are mandatory
- If a point fails to be added, error message appears
- A point too similar to one of the previous ones is rejected
- Significant part of the pattern must be visible on the scan for successful detection



8. Calibration



Hand-eye calibration -
Verification & Finalization

- After capturing at least 6 calibration points the point cloud is available in the 3D visualizer on the [Verification](#) tab
- If possible, validate the overlay of the robot model with the point cloud by triggering a scan from such position that a part of the robot is visible
- Save the result by clicking the button [Finish and save result](#)

8. Calibration

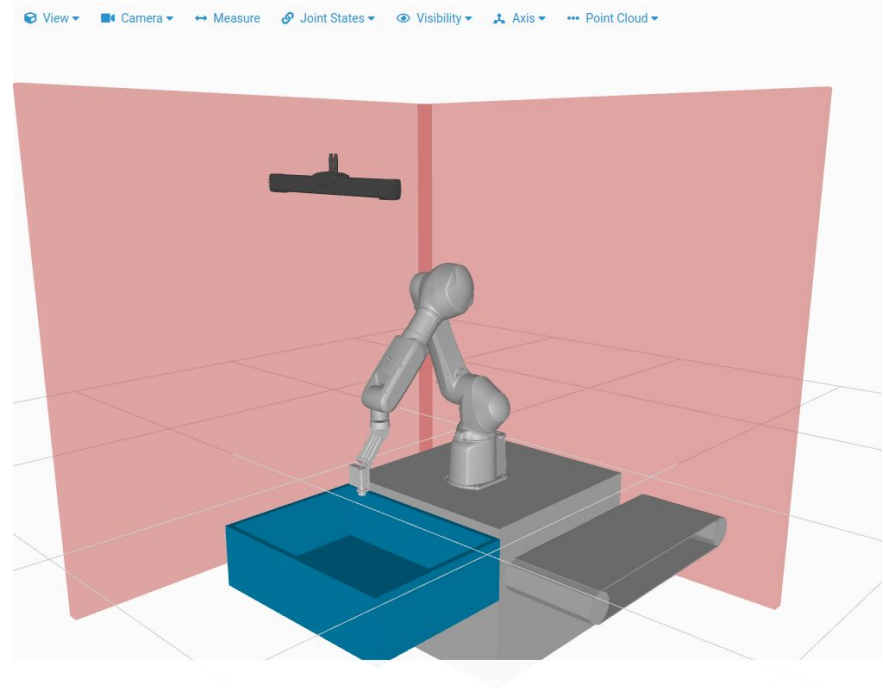
Semi-automatic calibration

- Adding calibration points directly from the robot program using the [Add calibration point](#) request
- Exemplar calibration program is part of the Robot module
 - only necessary to amend the network settings and define new calibration poses
- The user starts the calibration procedure in BPS and verifies the result once the robot program is finished

9. Environment

Scene

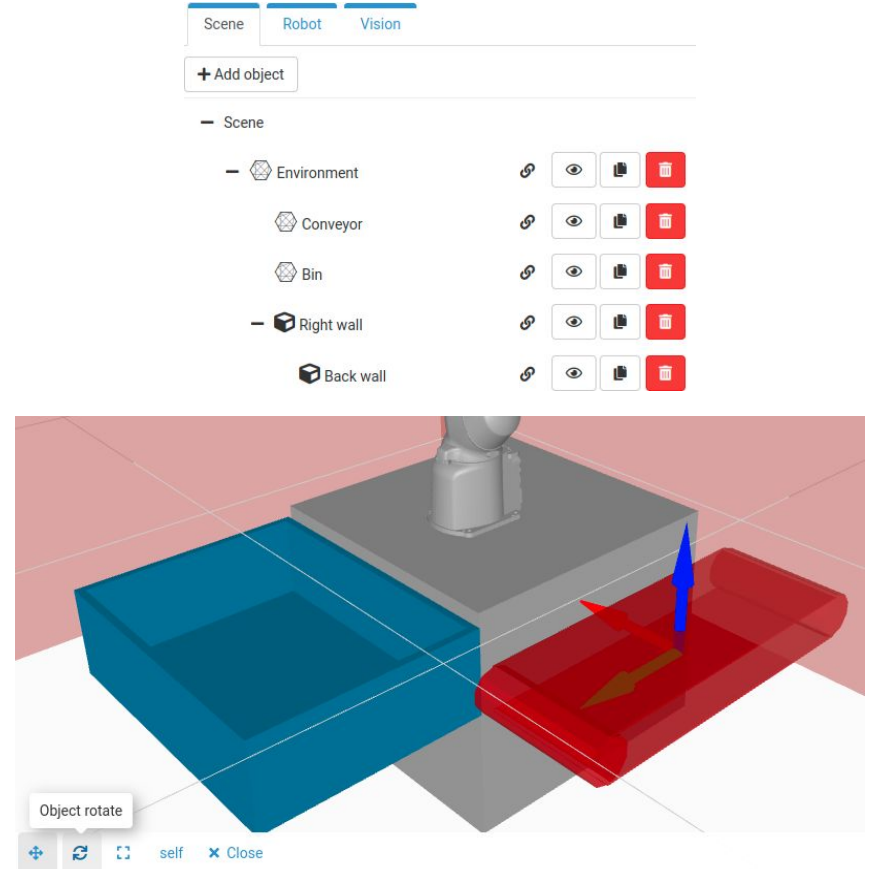
- Upload own STL or create your own collision object using the available shapes:
 - Box
 - Sphere
 - Cylinder
- Collision checking is done with every defined object
- Make sure the created model of the environment matches the reality



9. Environment

Scene

- Collision object types
 - Environment - solid grey
 - Virtual - transparent red
 - Bin - solid blue
- Option to duplicate an existing object
- Support for scaling, resizing, moving, rotating
- Hierarchical structure of objects



9. Environment

Robot

- It is highly recommended to set up joint limits in order to increase path planning performance
- Verify the joint limits by:
 - **Virtual** **jogging**
The robot must be able to reach all places in the bin with various orientations of the tool
 - **Robot** **Working** **Space**
Enables visualization of the robot's reachability and maneuverability

Joint limits ⓘ

☐ Degrees ☒ Radians

Joint 1 range [rad]



Joint 2 range [rad]



Joint 3 range [rad]



Joint 4 range [rad]



Joint 5 range [rad]



Joint 6 range [rad]

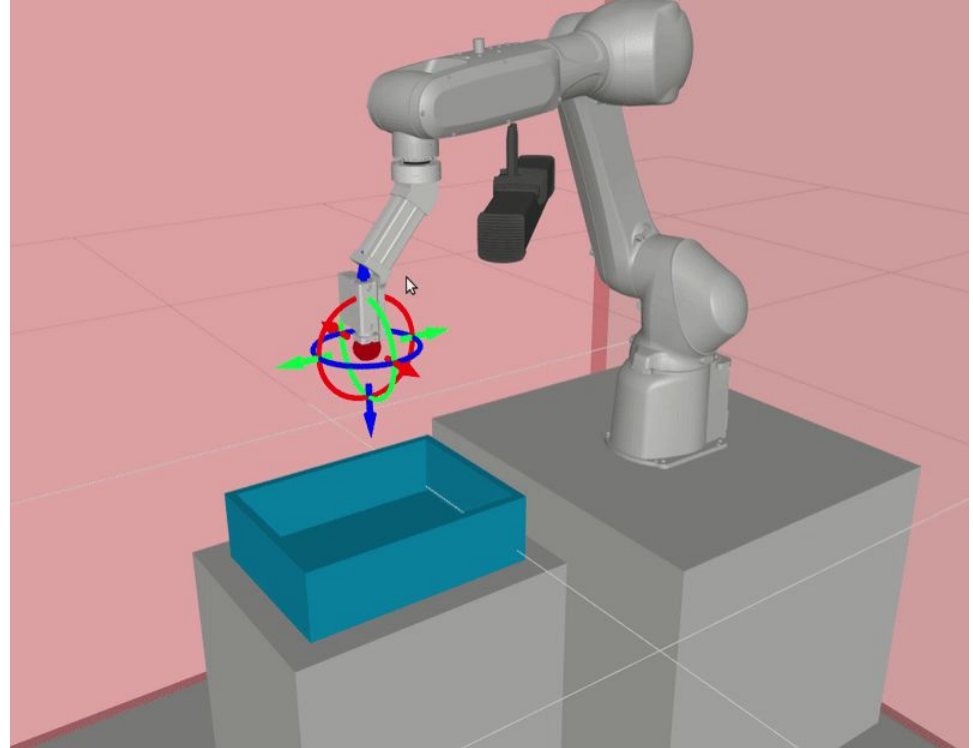


Save

9. Environment

Robot - Virtual jogging

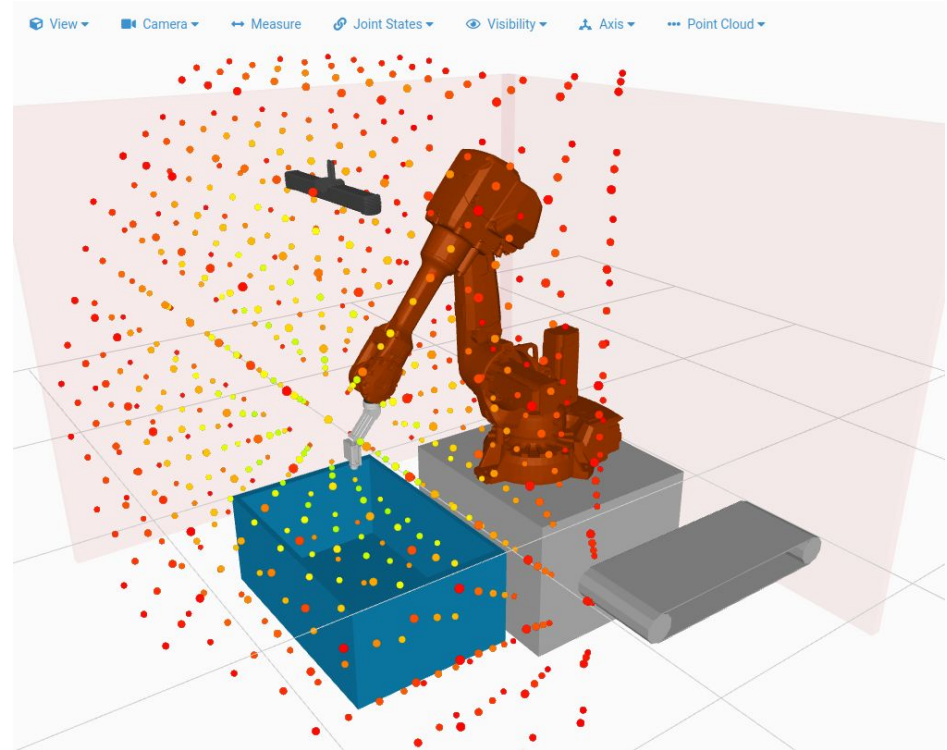
- Three options to jog the virtual robot:
 - Joint
 - Linear - Tool coordinate system
 - Linear - Robot base coordinate system
- Option to highlight collisions between the robot kinematic chain and all collision objects



9. Environment

Robot - Robot Working Space

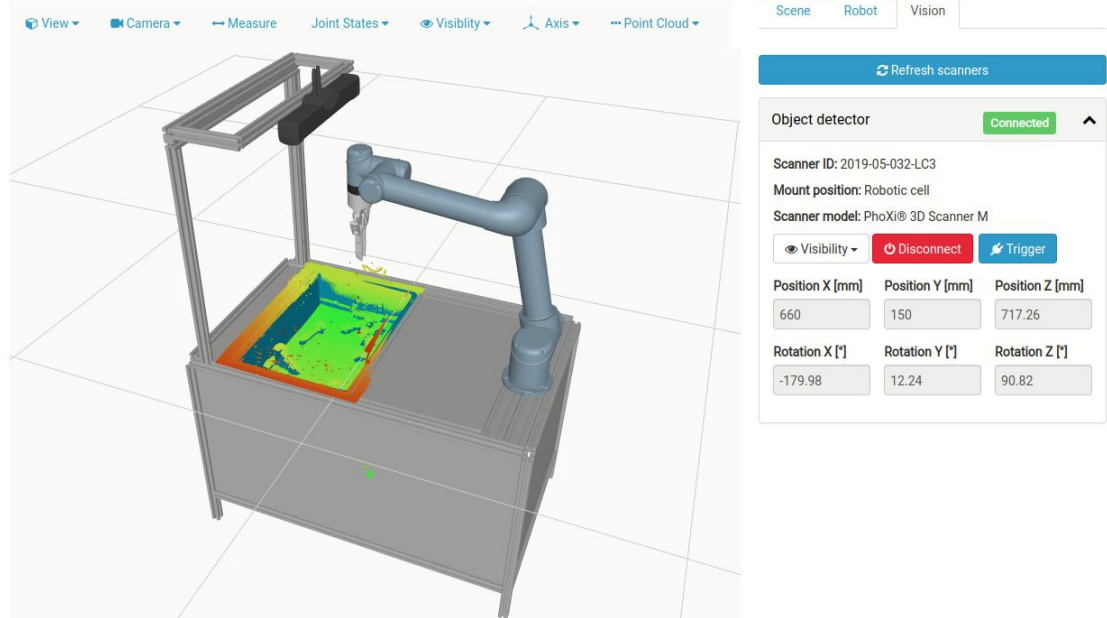
- Displays the working space in form of dots of various colors (red-green gradient) which represent the robot's ability to reach that particular point in space
- The green color means the robot is able to reach this point from (almost) all directions while the red point is reachable only from a few
- Always aim for points of colors in the upper half of the red-green color gradient in the volume of the bin and above it (in the area where the path planning is done)

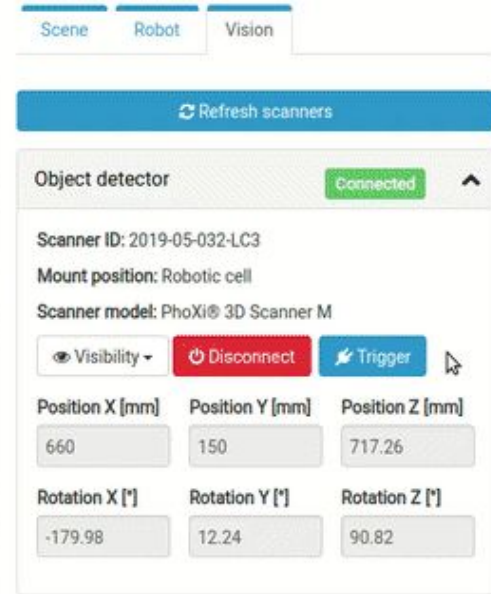
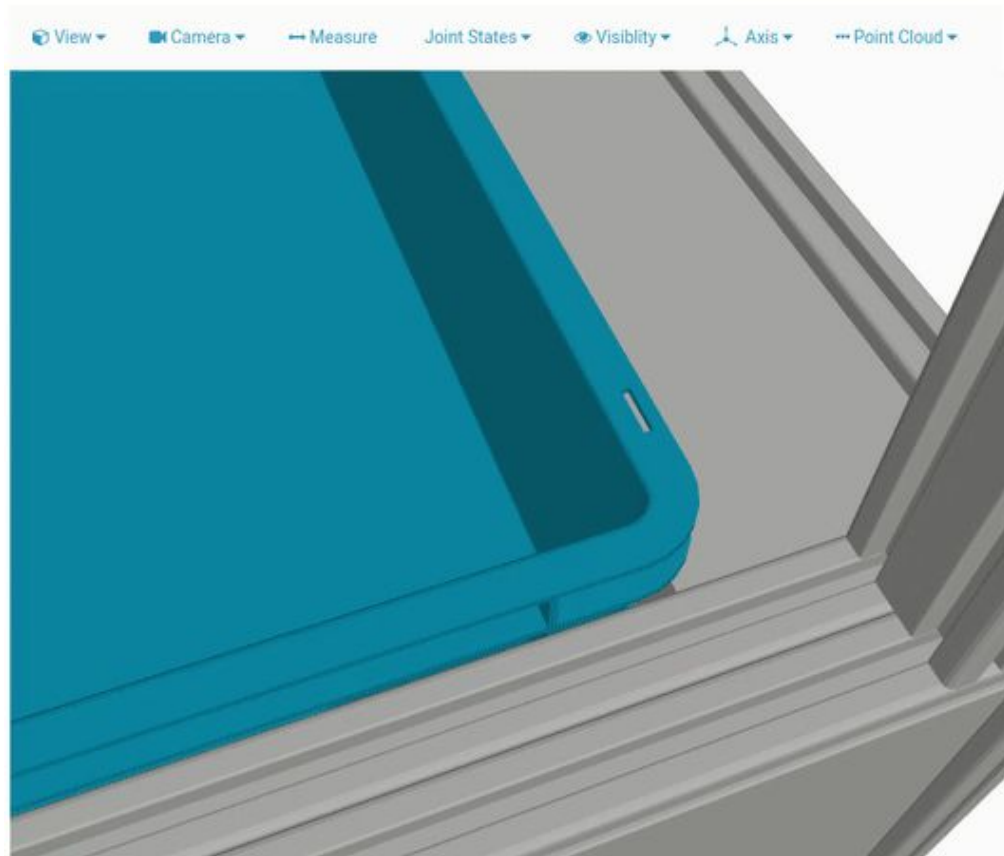


9. Environment

Vision

- Calibrated Vision systems with connectable scanners can capture scans enabling quick verification of the position of collision objects in the scanner's view
- Easier to set up the scanners' field of view thanks to the possibility to display their scanning volumes





Connected vision system after the calibration can be especially helpful when adjusting the position of the bin

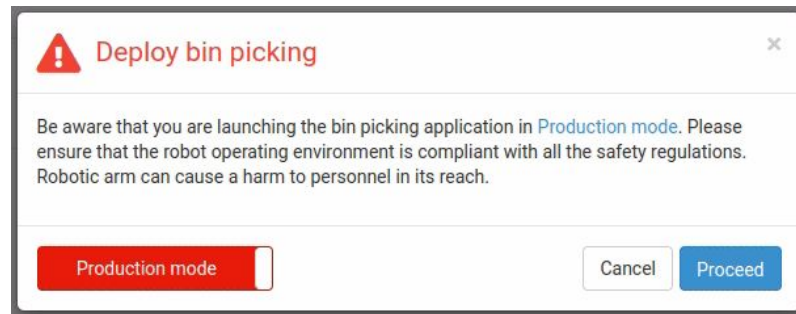
10. Deployment

Simulation mode

- Ideal for initial testing without real robot
 - real robot cannot call deployment requests
- Option to call deployment request directly from the Deployment page

Production mode

- Used during production
- Robot calls deployment requests



10. Deployment

Connection status & Console

- Connection statuses of:
 - Scanners (vision systems)
 - Robot State Server
 - Bin Picking Client (Robot main program)
- Option to stop/restart the deployed solution
- Option to call scan request in [Production](#) mode and initialization, scan and trajectory request in [Simulation](#) mode
- Console informing about past actions and events

Bin Picking Client: CONNECTED Robot State Server: CONNECTED Scanner: 1 2

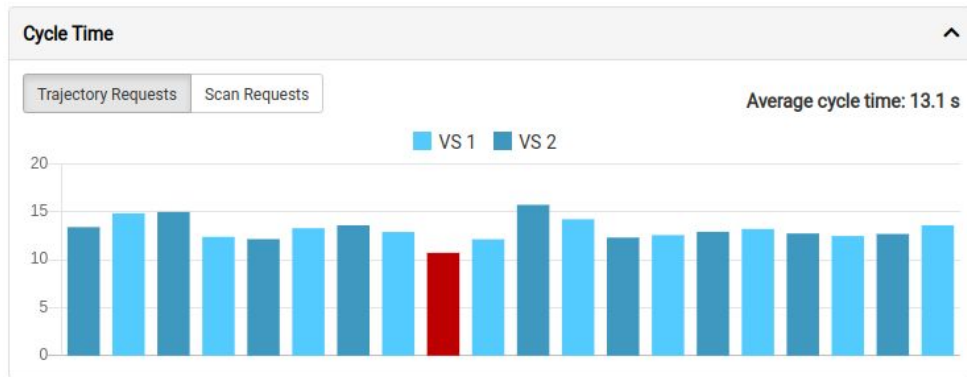
ID: 9 | Example solution 1

✖ Stop bin picking 🔄 Restart 🔍 Scan ▾

Cycle Time ▾

```
18:03:02 - [INFO] Localization finished [Vision System 1]
18:03:02 - [INFO] Localization found 20 objects [Vision System 1]
18:03:04 - [INFO] Localization found the first object [Vision System 2]
18:03:04 - [INFO] Request for trajectory received [Vision System 2]
18:03:05 - [INFO] A path plan has been computed for the first picked object
18:03:05 - [INFO] Picking object at location -0.00714177 0.679519 -0.0431379
18:03:07 - [INFO] Request for scan received [Vision System 1]
18:03:07 - [INFO] Waiting for localized objects [Vision System 2]
18:03:09 - [INFO] Localization finished [Vision System 2]
18:03:09 - [INFO] Localization found 19 objects [Vision System 2]
18:03:11 - [INFO] Localization found the first object [Vision System 1]
18:03:12 - [INFO] A path plan has been computed for the first picked object
18:03:12 - [INFO] Request for trajectory received [Vision System 1]
18:03:12 - [INFO] Picking object at location -0.0767157 0.679585 -0.0587516
18:03:15 - [INFO] Request for scan received [Vision System 2]
18:03:15 - [INFO] Waiting for localized objects [Vision System 2]
18:03:17 - [INFO] Localization finished [Vision System 1]
18:03:17 - [INFO] Localization found 19 objects [Vision System 1]
18:03:19 - [INFO] Localization found the first object [Vision System 2]
18:03:19 - [INFO] A path plan has been computed for the first picked object
18:03:20 - [INFO] Request for trajectory received [Vision System 2]
18:03:20 - [INFO] Picking object at location 0.00908747 0.563087 -0.0313297
18:03:22 - [INFO] Request for scan received [Vision System 1]
18:03:22 - [INFO] Waiting for localized objects [Vision System 1]
18:03:25 - [INFO] Localization finished [Vision System 2]
18:03:25 - [INFO] Localization found 18 objects [Vision System 2]
18:03:26 - [INFO] Localization found the first object [Vision System 1]
18:03:27 - [INFO] Request for trajectory received [Vision System 1]
18:03:27 - [INFO] A path plan has been computed for the first picked object
18:03:27 - [INFO] Picking object at location -0.0438315 0.575201 -0.0620058
18:03:29 - [INFO] Request for scan received [Vision System 2]
18:03:29 - [INFO] Waiting for localized objects [Vision System 2]
```

10. Deployment



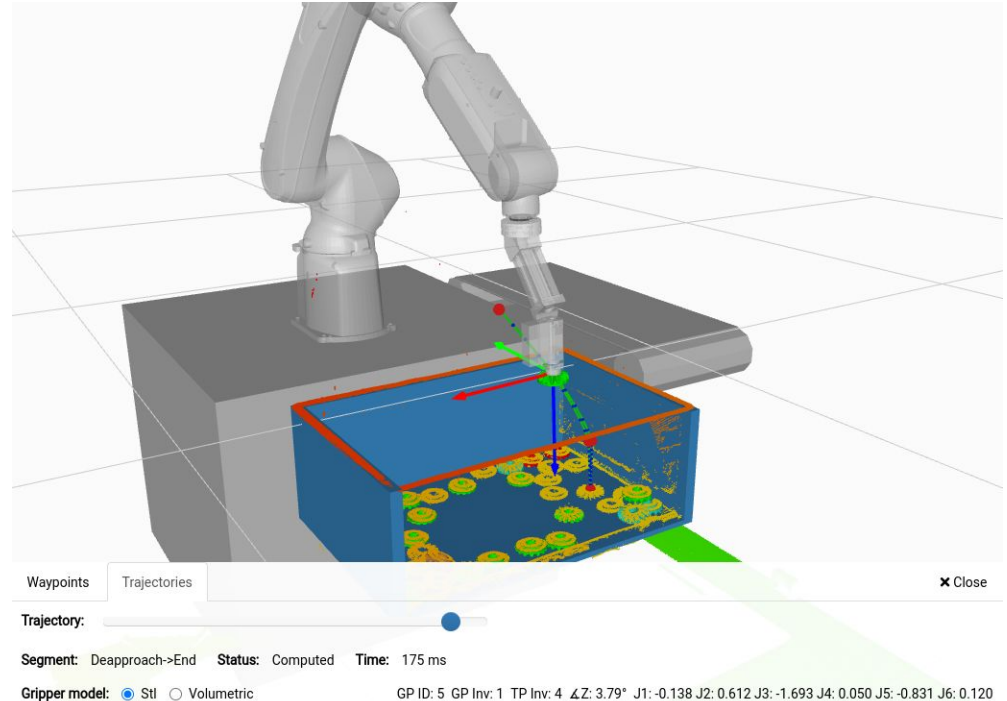
Cycle time & Object status chart

- Cycle time computed based on incoming scan / trajectory requests
- Measured *per vision system* - between requests for the same vision system
- Average cycle time calculated from visible samples
- Object status chart provides information about the percentage representation of individual *object statuses* of all localized objects

10. Deployment

Visualizer and the Inspector

- Real-time visualisation of the robot and the currently executed trajectory
- Localized objects rendered in colors corresponding to their current path planning status
- Inspector tool for complete diagnostics of every localized object



10. Deployment

Color	Status	Description
	New object	Either a newly localized object in this run or an object from the previous run which has moved and its position change has exceeded defined pose tolerance thresholds for keeping its old path plan or its part. Computation of a new path plan will start from scratch in this run. Note: Pose tolerance thresholds are adjustable in the Preserve path plan section of solution settings. For symmetrical objects also define the parameter Object symmetry axis.
	Object from the previous run	An object from the previous run whose position change is within the defined pose tolerance thresholds for keeping its old path plan (or its part). The old path plan will be evaluated in this run. Note: Pose tolerance thresholds are adjustable in the Preserve path plan section of solution settings. For symmetrical objects also define the parameter Object symmetry axis.
	Path plan from the previous run is not valid	A waypoint or a trajectory computed in the previous run is no longer valid (collision on the current scene detected) or the object failed to have a waypoint/trajectory computed in the previous run. Computation of a new path plan will start from scratch in this run.

Colour codes - BPS pipeline

These are status colors of the respective stages of the BPS pipeline in the order they are computed.

	Path plan from the previous run is valid	All waypoints and some or all trajectories are valid from the previous run. Missing trajectories will be computed in this run.
	All waypoints computed	The IK of all configured waypoints has been computed successfully.
	Trajectory planned	All trajectories have been planned successfully.
	Ready to pick	The object is ready to be picked.
	Object being picked	The trajectory for this object has been sent to the robot - it will be picked in the current cycle.

10. Deployment

Colour codes - BPS pipeline

These are status colors of failures in path computation which may arise at the various stages of the BPS pipeline. They are listed in the order of path planning stages at which they may occur.

Color	Status	Description
	No gripping points defined	The object associated with this vision system has no gripping point defined.
	Object occluded from scanner position	Other objects occlude this object from the scanner's point of view. Note: Only available when the parameter Pick only unoccluded objects is set to Enabled (non-default value).
	Grasping angle rejected	The angle between the Z-axis of the tool point and the gripper angle reference vector in the Grasp waypoint is larger than the allowed value.
	Gripper in collision in grasp pose	The gripper is in collision with the scanned scene or an already localized object in the Grasp waypoint. Note: Only available when the parameter Gripper collision checking type is set to Volumetric (default value).
	Waypoint computation failed	One of the configured waypoints has no valid IK solution.
	Trajectory planning failed	One of the trajectories failed to plan.

11. Settings

 Save

 Reset to defaults

Collision checking settings

Object

☐ Pick only unoccluded objects ⓘ

☒ Collision checking with the picked object attached to the gripper ⓘ

Collision checking type for localized objects* ⓘ

Volumetric ▾

Gripper

Gripper collision checking type* ⓘ

Volumetric ▾

When 'Volumetric' is selected, the 'Gripper collision sensitivity' setting is available in each vision system. Please, review the sensitivity value in each of your [vision systems](#).

Point cloud

Resolution of the point cloud volumetric model [m] <0.001; 0.05>* ⓘ

0.01

Minimum required size of a volumetric model segment <1; 100000>* ⓘ

20

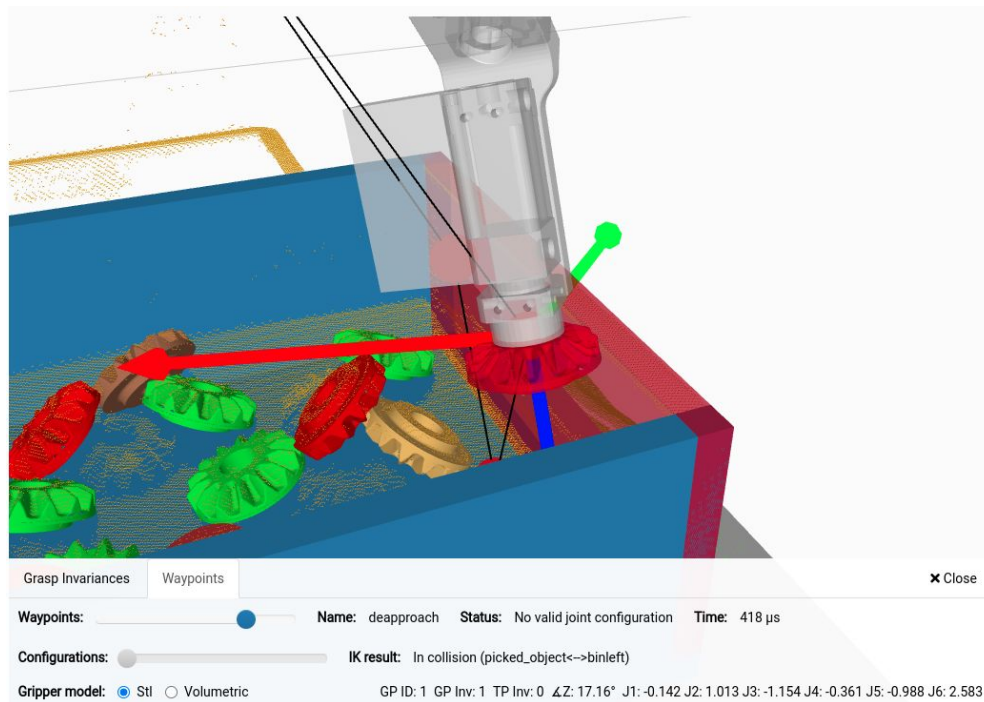
Global solution settings

- Before adjusting values in the Settings section:
 - Complete the mandatory steps of the configuration
 - Test your solution in deployment
 - Try to improve the performance by adjusting regular configuration steps
- Explanation of each parameter is inside its tooltip
- Value can be reset to default using the button 
- Related settings are grouped into categories (e.g. Collision checking settings) and sub-categories (e.g. Object, Gripper, Point cloud)
- Several settings can be adjusted for each vision system - navigate to vision system page to configure them

11. Settings

Collision checking with the picked object attached to the gripper

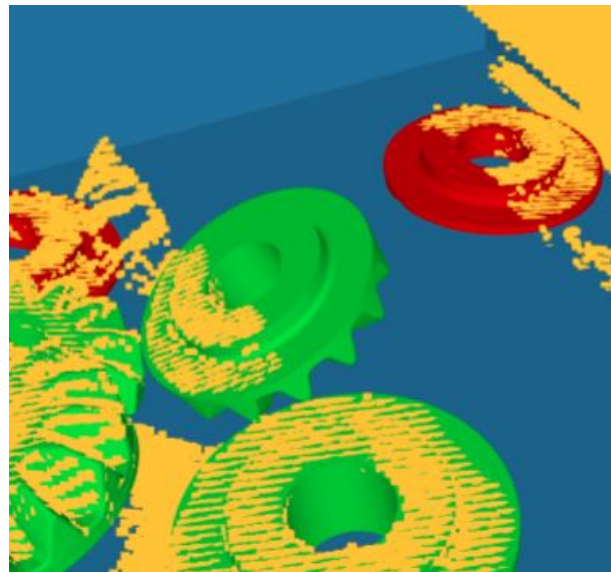
- Relevant especially when handling objects significantly larger than the gripper
- Collision checking with CAD model of the picked object
- The object is attached to the gripper based on the used gripping point - it reflects reality
- Collision checking with the grasped object is executed from Deapproach to the End waypoint including the trajectories between them
- field **IK result** informs about colliding links



11. Settings

Collision checking type for localized objects

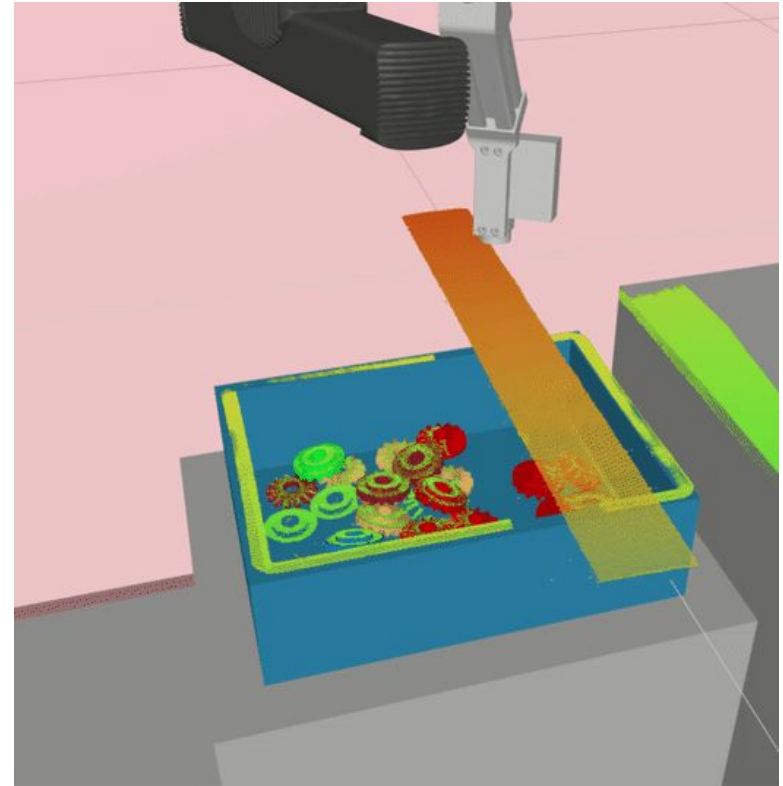
- Collision checking with other objects localized in the bin
 - The localization can detect objects even if they are not entirely visible in the scene:
 - Hidden behind another object
 - Hidden in the shadow of the bin
 - The entire CAD model of localized object is rendered into the scene and used as a collision object



11. Settings

Collision checking with point cloud

- Point cloud is considered a collision object equivalent to CAD models in the environment
- Path planning takes point cloud into the account to avoid collisions with gripper, robot body, and scanner mounted on the robot (hand-eye approach)
- The point cloud is represented via volumetric model, its properties can be customized with following settings:
 - Resolution of the point cloud volumetric model
 - Minimum required size of a volumetric model segment

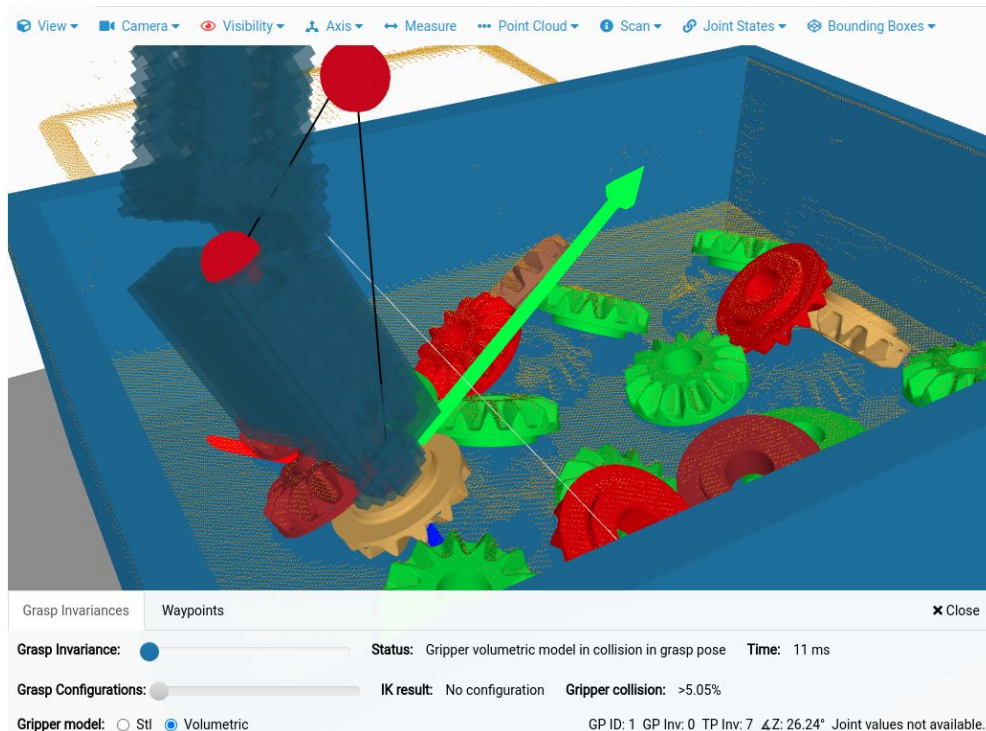


11. Settings

Gripper collision sensitivity

- Threshold percentage value of the gripper volume that may get in a collision with the scanned point cloud or with any currently localized objects (if enabled)
- Configurable for each Vision system
- Check field [Gripper collision](#) in the Inspector

Tip: May be useful especially when picking with a two-finger gripper, the robot sometimes collides with the neighboring objects.



11. Settings

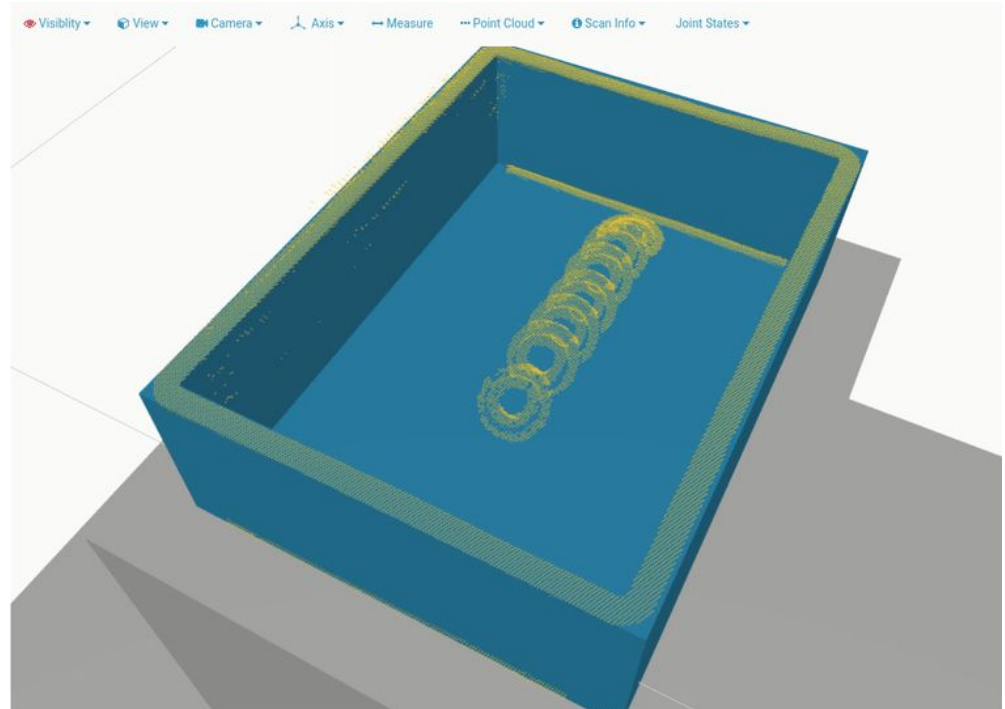
Pick only unoccluded objects

Restrict the robot from picking partially occluded objects which often means they are blocked

- the occlusion is evaluated from the scanner's camera POV

Colour codes:

- **Occluded object**
- **Pickable object**



11. Settings

Picking priority method

- Specify the direction in which the localized objects are to be picked
 - default direction is vector pointing down, so we are picking from top to bottom
- The direction vector is defined in robot space

Picking priority method* ⓘ

- Closest point of localized object in given direction (default)
- Closest point of localized object in given direction (default)**
- Centroid coordinate of localized object in given direction
- Overlap of localized object with model - reported by localization
- Number of collisions with gripper

Picking direction vector X <-1.0; 1.0>* ⓘ

0.0

Picking direction vector Y <-1.0; 1.0>* ⓘ

0.0

Picking direction vector Z <-1.0; 1.0>* ⓘ

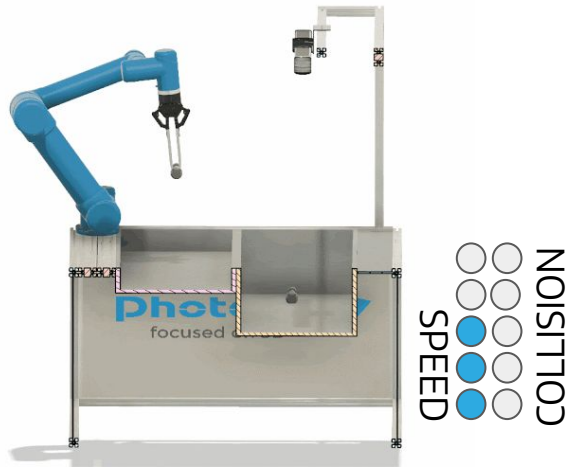
-1.0

11. Settings

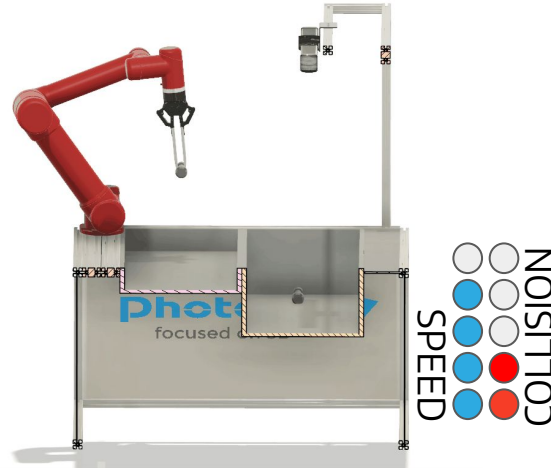
	Plan joint and linear trajectories	Plan only linear trajectories	Path planning off
Waypoints	Checked for collision	Checked for collision	Checked for collision
Linear trajectories	Checked for collision	Checked for collision	No validation
Joint trajectories	Checked for collision	No validation	No validation
Speed of calculation	Medium	Medium	High
Collision probability	No	Medium	High
Environment	Complex	No obstacles in reach	No obstacles in reach
Bin	Deep	Deep	Flat surfaces only

11. Settings

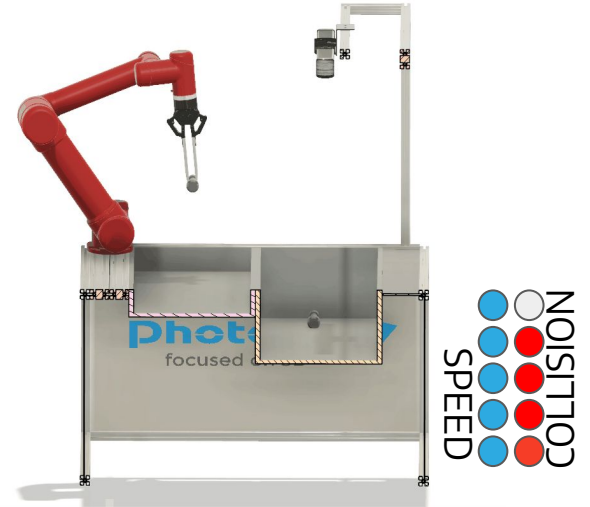
Plan joint and linear trajectories



Plan only linear trajectories



Path planning off

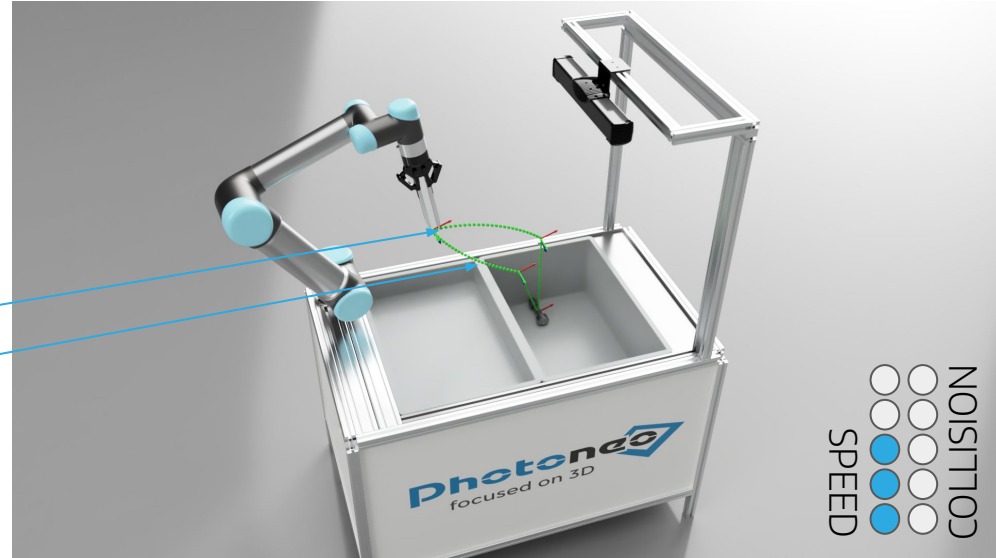


BLUE - Photoneo path planning
RED - Robot interpolation

11. Settings

Plan joint and linear trajectories

- All the trajectories are planned and validated for collision-free movement
- Waypoints and trajectories validated and collision free
- Most preferred option
- Waypoint
- Trajectory



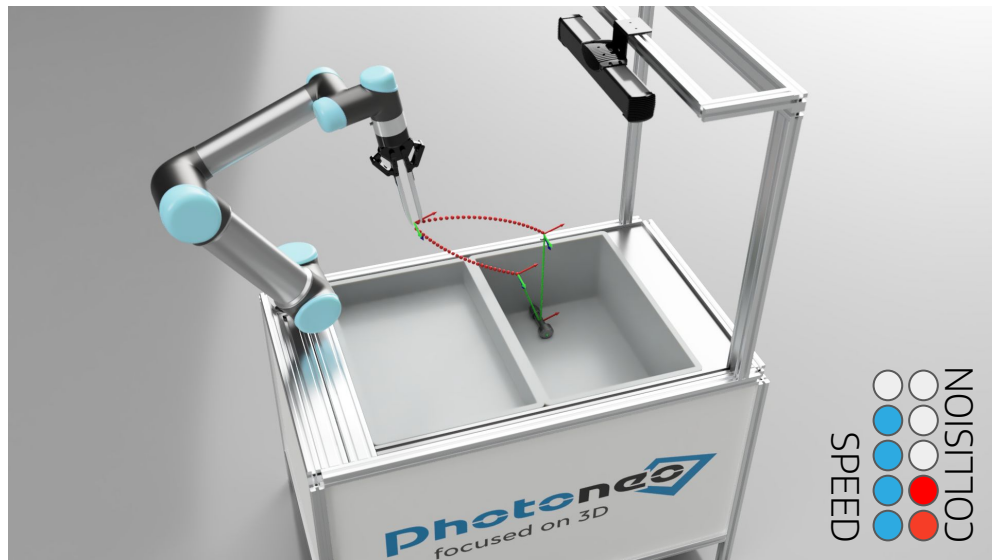
GREEN
RED

- Photoneo path planning
- Robot interpolation

11. Settings

Plan only linear trajectories

- Linear trajectories and all waypoints are validated for collision-free movement
- With default path stage settings:
 - Trajectory from Start to Approach and from Deapproach to End is interpolated by the robotic controller
 - Trajectories between Approach and Deapproach are linear (always)
- Computation time is reduced
- Higher collision probability during the trajectories interpolated by robot



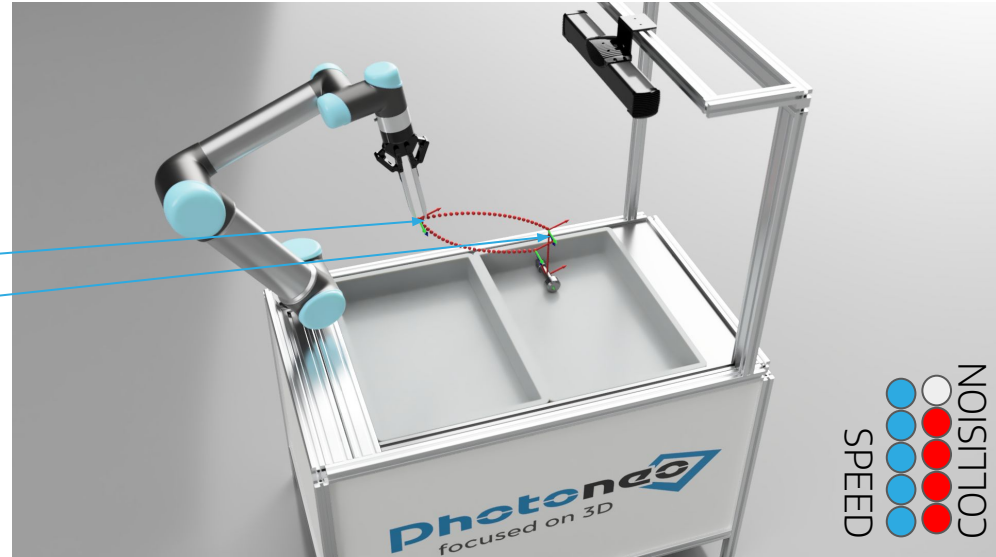
GREEN
RED

- Photoneo path planning
- Robot interpolation

11. Settings

Path planning off

- Path planning is disabled
- Only waypoints are checked to be collision-free
- Computation time significantly decreased
- Start and End waypoints
- Approach waypoint
- Avoid using deep bins



GREEN
RED

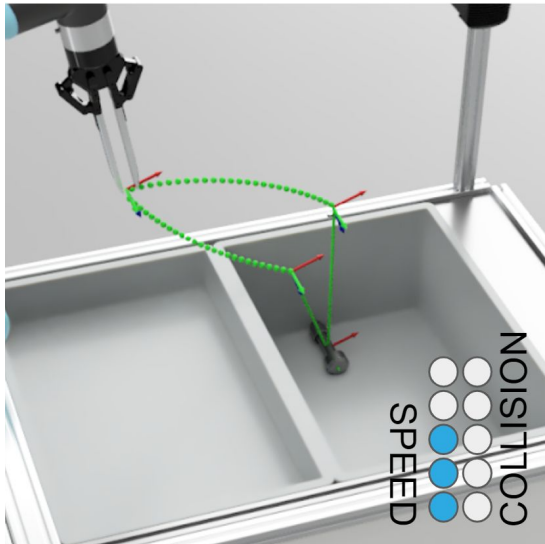
- Photoneo path planning
- Robot interpolation

11. Settings

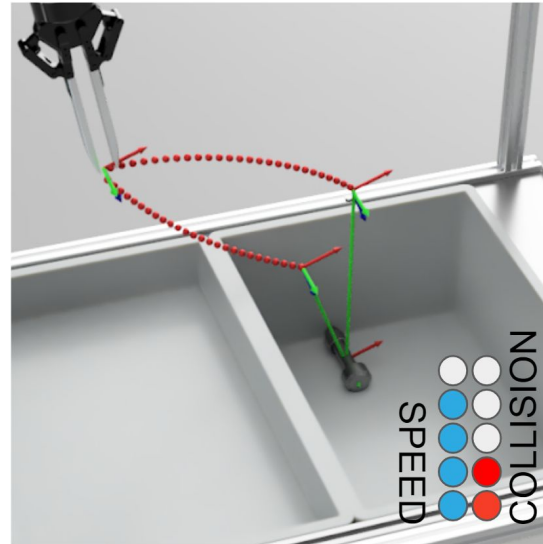
GREEN
RED

- Photoneo path planning
- Robot interpolation

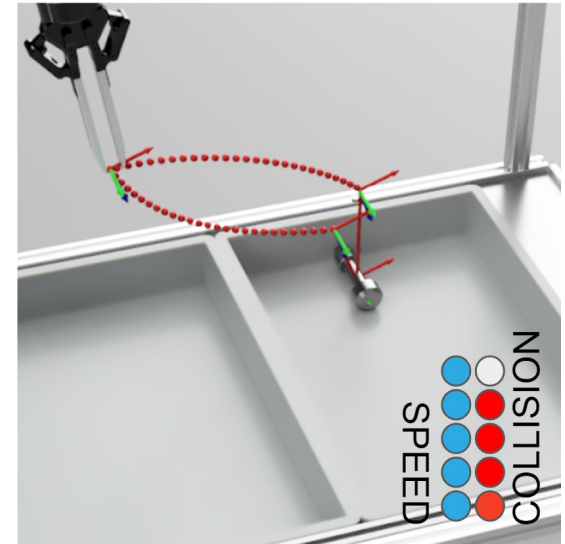
Plan joint and linear trajectories



Plan only linear trajectories



Path planning off



Accomplished with
Bin Picking Studio



Accomplished with
Bin Picking Studio



Accomplished with
Bin Picking Studio



Thank you

...and Happy picking!

support@photoneo.com

