Bin Picking Studio 1.12

Introduction and Training

Covers Bin Picking Studio 1.12 and Locator Studio 1.5. Based on the VGR documentation.

Training agenda

01 Getting started

BPS and LS, solution types, vision controller, users, network

02 Studio interface

Solutions, PhoXi page, software modules

03 Robot communication

Robot module, channels, the picking cycle

04 BPS solution setup

Robot, gripper, picked object, gripping points, grasping methods

05 Vision system and localization

Sensors, scanning profiles, CAD / AI / Layer engines

06 Calibration

Types, procedure, verification

07 Environment

Scene, virtual robot, joint limits, scene states

08 Settings

How settings work and the most important ones

09 Deployment and troubleshooting

Deployment page, object status, Inspector, support

10 Locator Studio specifics

LS setup and the reported pose

01

Getting started

IN THIS CHAPTER

Bin Picking Studio and Locator Studio

The five solution types

The vision controller

First start-up and user roles

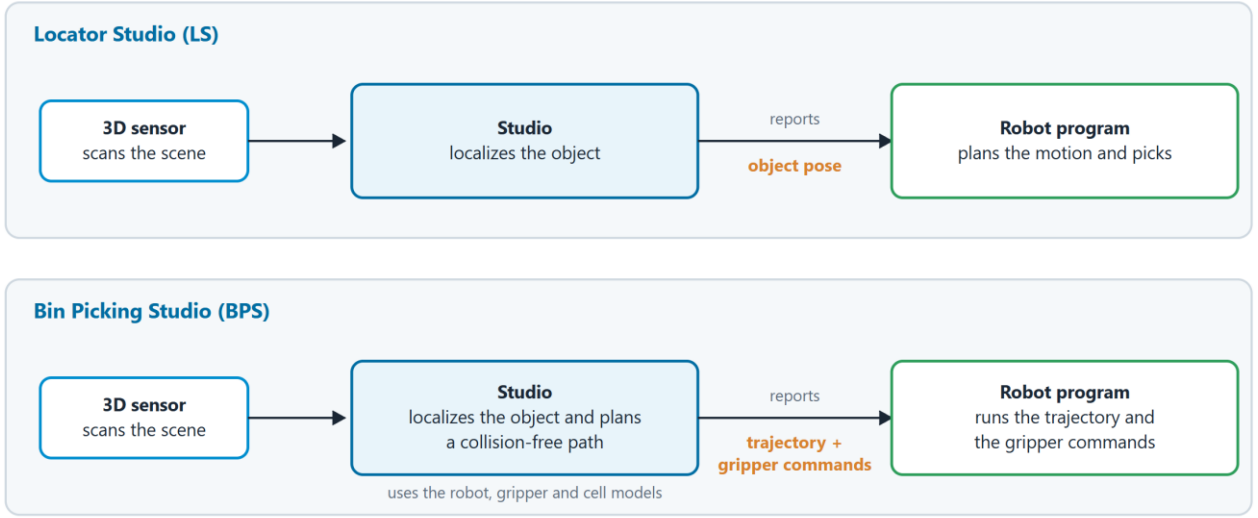
Network setup

Remote access

Autostart

One Studio, two products

- **Locator Studio (LS)** reports the **pose of the localized object**. The robot program plans the motion itself.
- **Bin Picking Studio (BPS)** reports a **collision-free trajectory** and the gripper commands. To plan the path it needs models of the robot, the gripper and the cell.
- In both cases the Studio **never takes control** of the robot. It only answers the action requests that the robot program sends.
- The solution type you choose decides whether a solution is BPS or LS.



In both cases the robot program sends the requests and decides what to do with the result.

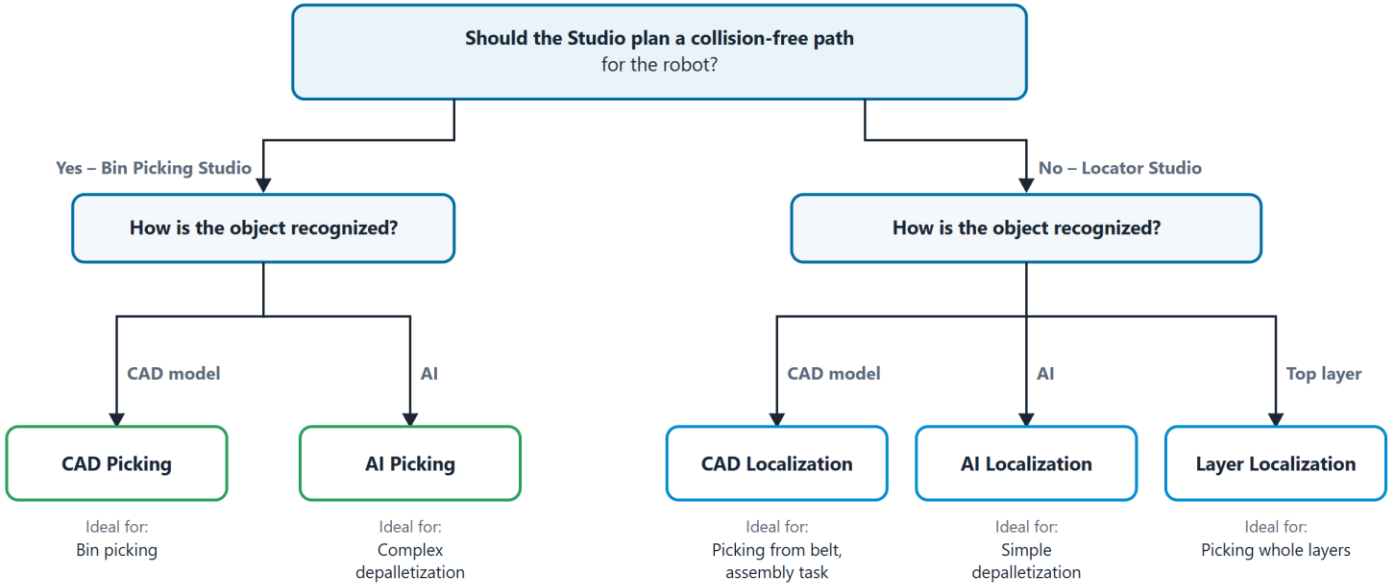
The five solution types

You choose the type in the Create a new solution dialog.

Solution type	Product	Recognition by	Reports	Ideal for
CAD Localization	LS	CAD model	Object pose	Picking from belt, assembly task
AI Localization	LS	AI	Object pose	Simple depalletization
Layer Localization	LS	Top layer	Top-layer pose	Picking whole layers
CAD Picking	BPS	CAD model	Collision-free trajectory	Bin picking
AI Picking	BPS	AI	Collision-free trajectory	Complex depalletization

Tip: Which types you can create depends on the **software modules** (licenses) of the installation. Without a license the **Create** button is disabled.

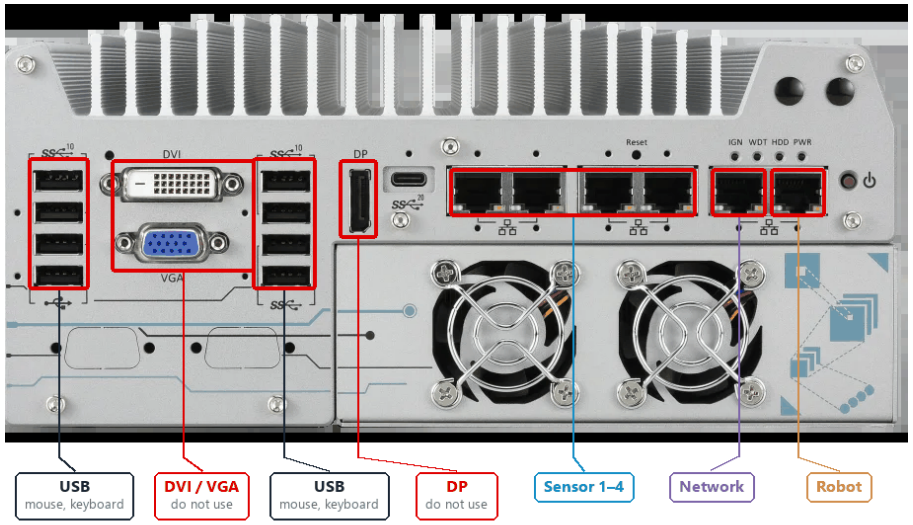
Which type to choose



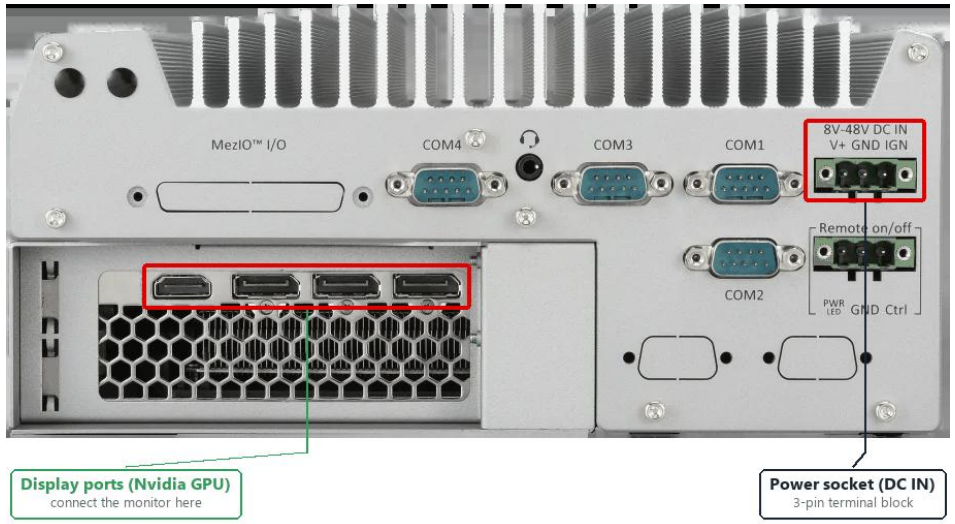
1. **Should the Studio plan a collision-free path for the robot?** Yes: a BPS type (CAD Picking, AI Picking). No: an LS type.
2. **How should the object be recognized?** By its CAD model, by AI, or as a whole top layer (Layer Localization, LS only).

The vision controller

An industrial computer with Ubuntu 22.04 and the Studio pre-installed.



Front: Ethernet and USB ports

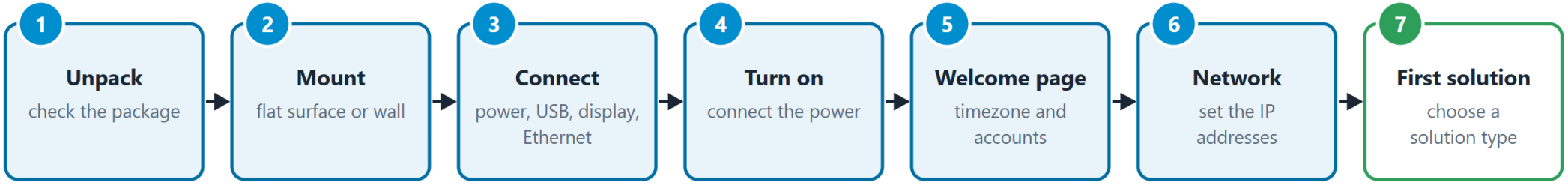


Back: Nvidia GPU display ports and DC IN

- Six Ethernet ports, left to right: **Sensor 1–4**, **Network**, **Robot**.
- The DVI, VGA and DP ports of the integrated graphics are **not supported**.

- Connect the monitor to the **Nvidia GPU** ports on the back. No monitor: use the HDMI dummy plug.
- Ambient temperature at most **50 °C**. On a wall, keep the heatsink fins vertical.

First start-up



The Welcome page guides you through the first setup

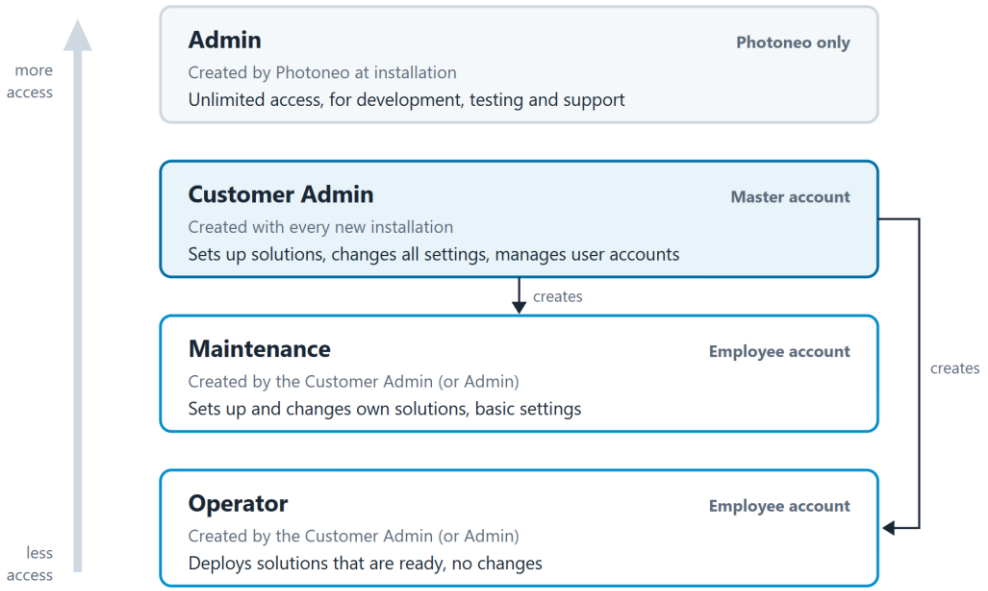
1. **Timezone**: click your location on the map. You can change it later on the Customization page.
2. **Master account**: the **Customer Admin** account that manages the Studio and the employee accounts.
3. **Employee accounts** (optional): accounts with the **Maintenance** or **Operator** role. You can add them later.

Then continue with the network setup and your first solution.

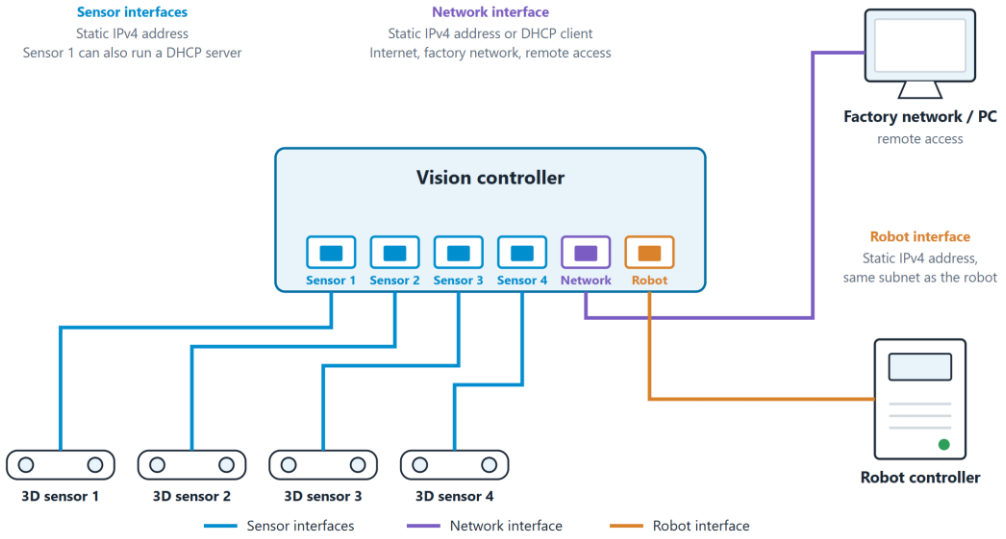
User roles

- **Admin**: Photoneo employees only, for development and support.
- **Customer Admin**: the Master account. Sets up solutions, changes the network and manages the user accounts.
- **Maintenance**: creates and changes its own solutions and uses the PhoXi page. No user management, no network settings.
- **Operator**: deploys solutions marked **ready**. Cannot change solutions, grippers or settings.

Accounts are created in the menu > **Account management** > **Create account**.



Network: three kinds of interfaces

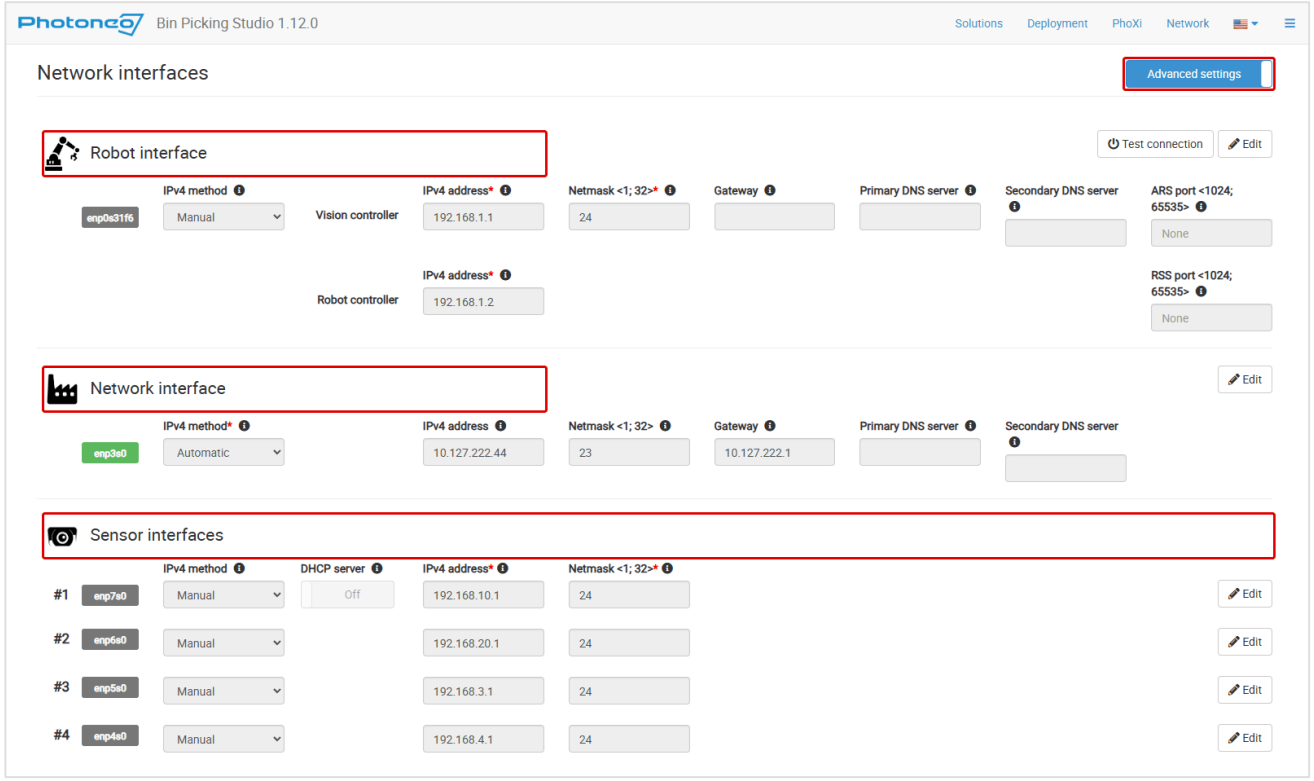


Interface	Ports	Connect	IP address
Sensor interfaces	Sensor 1–4	Photoneo 3D sensors, directly or through a Gigabit switch	Static. Interface 1 can run a DHCP server
Network interface	Network	Factory network, Internet, a computer for remote access	Static or DHCP
Robot interface	Robot	The robot controller, directly	Static, same subnet as the robot

- Set all interfaces on the **Network** page of the Studio, never in the Ubuntu network panel.
- Robot interface defaults: vision controller **192.168.1.1**, robot controller **192.168.1.2**. Use **Test connection** after saving.
- Subnets must not overlap each other or the reserved ranges **169.254.0.0/16**, **172.18.0.0/28** and **192.168.101.0/24**.

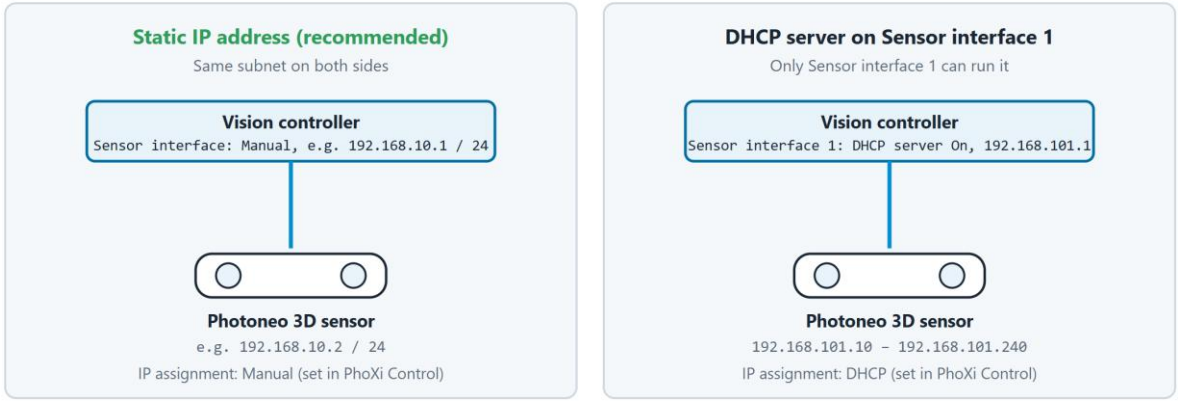
The Network page

- One block per interface: **Robot interface**, **Network interface**, **Sensor interfaces**.
- Click **Edit**, change the values, then **Save** or **Discard**. Other blocks are locked meanwhile.
- **Advanced settings** shows the gateway, the DNS servers and the **ARS / RSS** ports.
- Leave the ARS and RSS ports empty unless a Photoneo technician advises otherwise. A custom port must be set in the Robot module too.



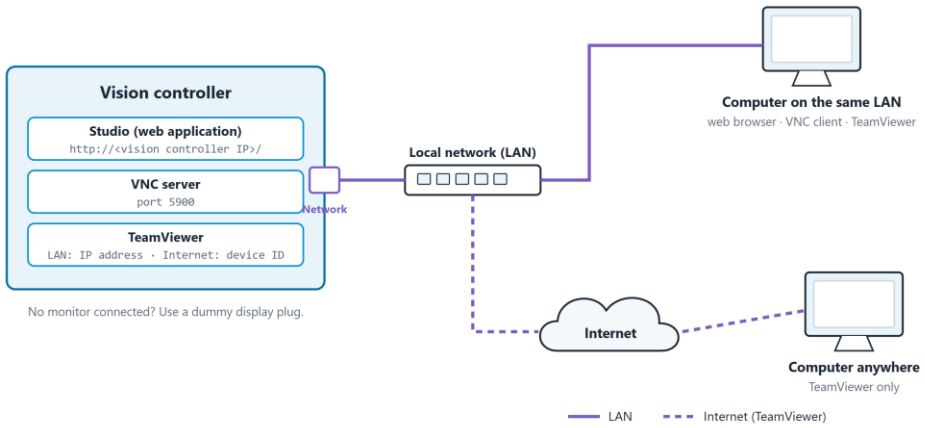
Sensor IP addresses

- A sensor prefers its fixed IPv6 link-local address and falls back to IPv4, so **always give it a valid IPv4 address**.
- **Recommended**: a static address on the sensor interface and on the sensor (PhoXi Control), on the same subnet.
- **Or**: turn on the DHCP server of **Sensor interface 1**. It then uses 192.168.101.1 and gives sensors .10–.240.
- In PhoXi Control, **N/A** in the IPv4 field means the configuration is invalid.



Tip: More than four sensors? Connect a Gigabit switch to any sensor port.

Remote access

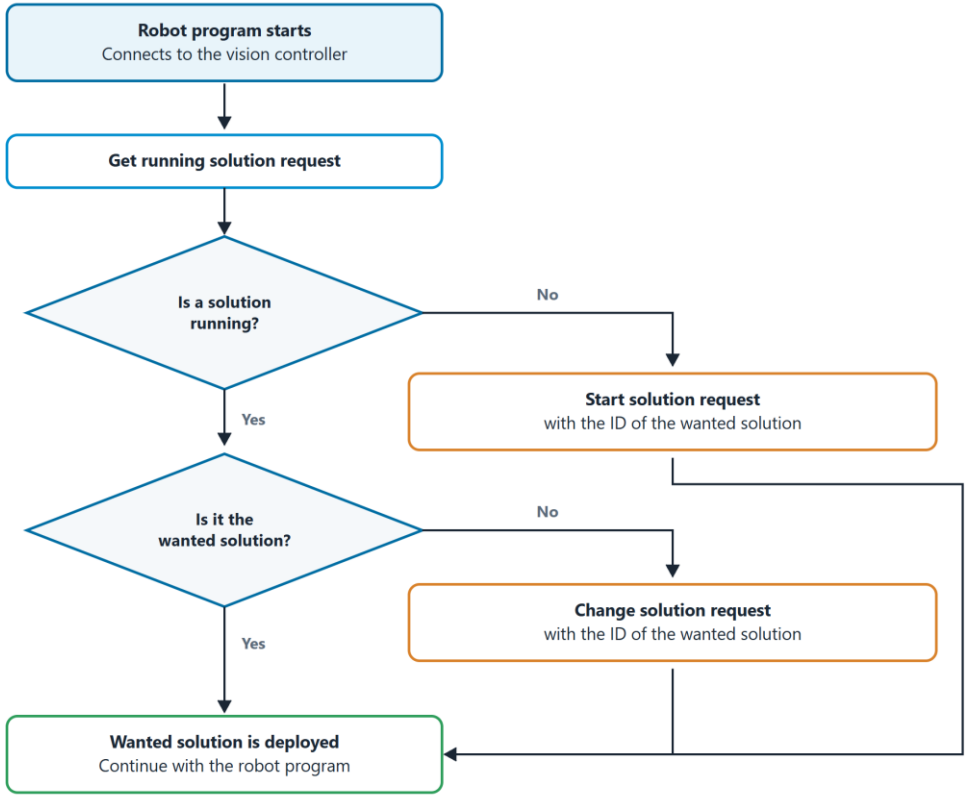


	Web browser	VNC	TeamViewer
You see	The Studio only	The whole desktop	The whole desktop
PhoXi Control	No	Yes	Yes
Network	Local network	Local network	Local network or Internet
Connect with	IP address	IP address, port 5900	IP address or device ID
Password	Studio account	From Photoneo support	Shown in TeamViewer

- Connect through the port labelled **Network**. A **static IP address** is recommended for VNC and TeamViewer.
- No monitor on the vision controller? Plug in the **dummy display plug**.
- Browser: **Google Chrome** is recommended.

Autostart

- There is **no autostart setting**. After a restart the Studio restores its last state: a deployed solution is deployed again.
- The robot program decides which solution runs:
  - **Get running solution**: which solution is deployed?
  - **Start solution**: deploy one when none is running.
  - **Change solution**: replace the running one.
- The solution must be marked **ready** and its robot brand must match the robot that sends the request.



02

Studio interface

IN THIS CHAPTER

Navigation bar and menu

Solutions page

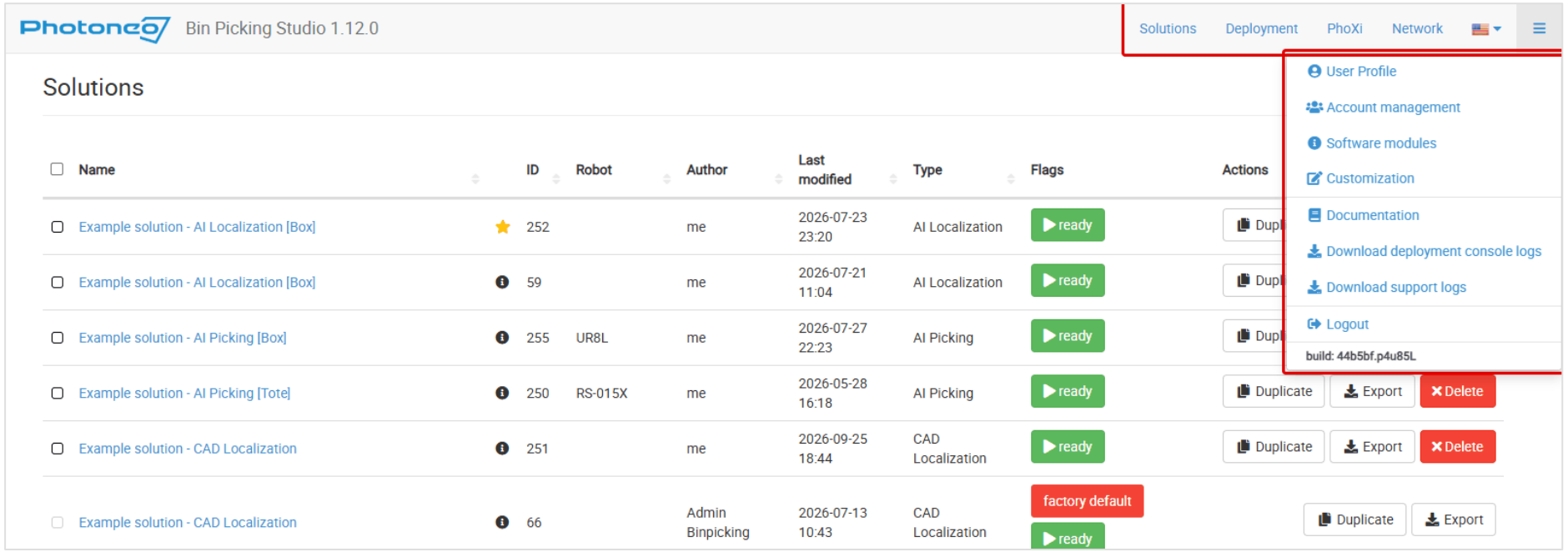
Creating a solution

The solution page and setup steps

PhoXi page

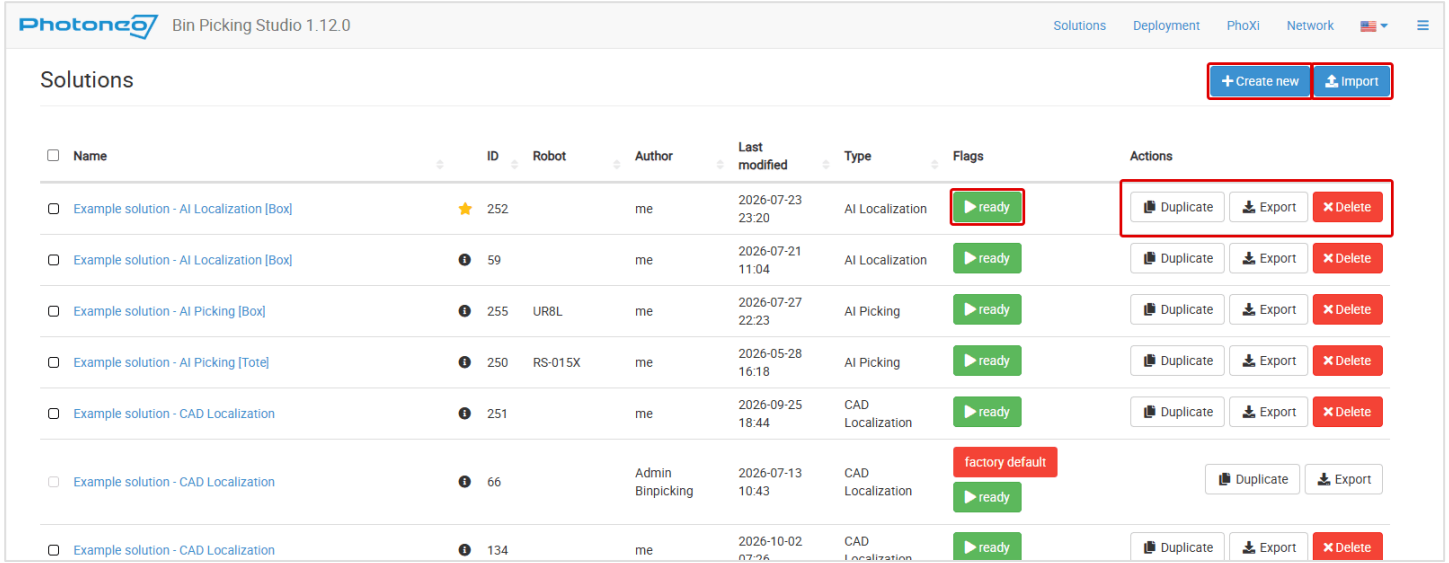
Software modules and Customization

Navigation bar and menu



- **Top bar:** Solutions, Deployment, PhoXi, Network and a language selector.
- **Menu:** User Profile, Account management, Software modules, Customization, Documentation, Download deployment console logs, Download support logs, Logout.
- The menu also shows the **build number** of the Studio.

Solutions page



- **ready**: can be deployed. Click the flag to deploy.
- **last run**: deployed last.
- **factory default**: preloaded, cannot be edited or deleted. Duplicate it first.

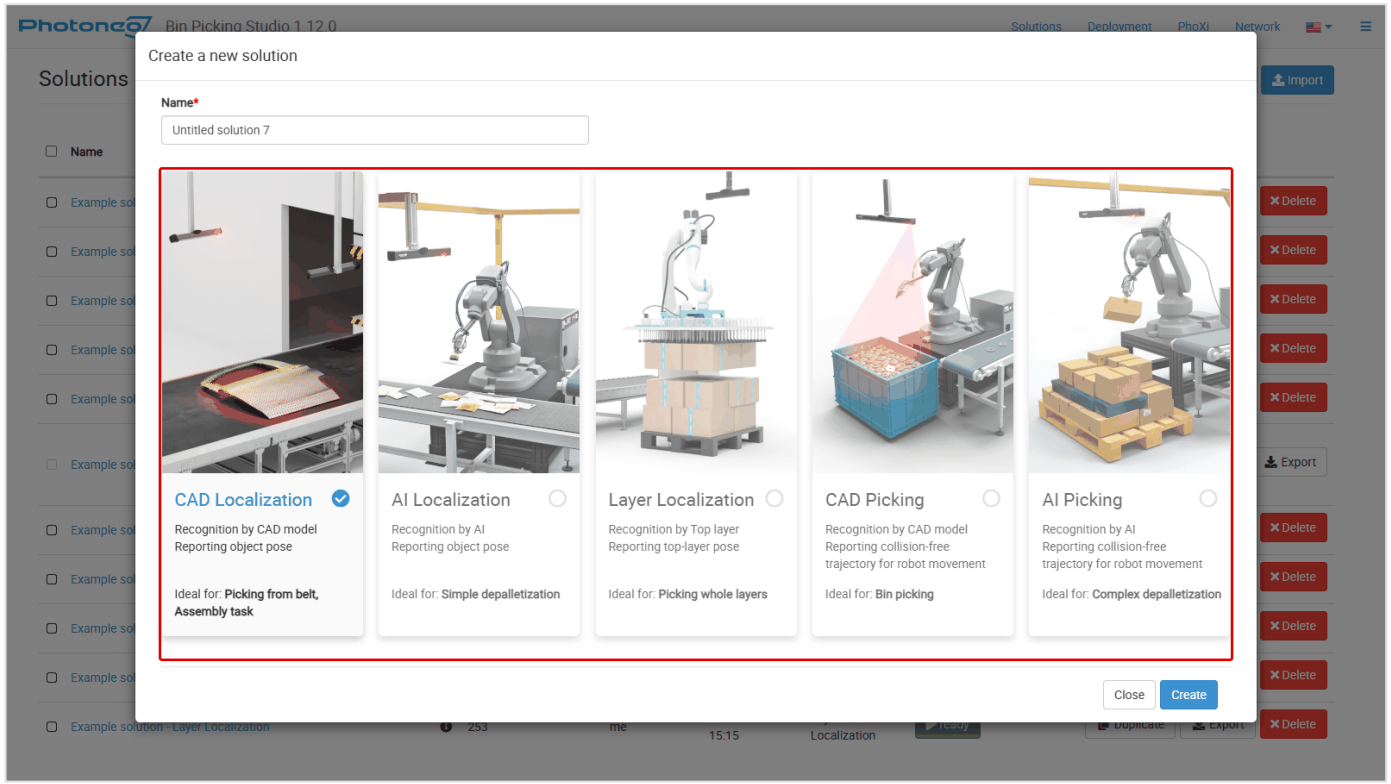
- **Duplicate**, **Export** and **Delete** in each row.
- Solutions are exported and imported as .pbcf files, with their attachments.
- Example solutions: CAD Picking, AI Localization [Box], Layer Localization and more.

Creating a solution

1. Click **Create new** on the Solutions page.
2. Enter the **Name**.
3. Choose the solution type.
4. Click **Create**.

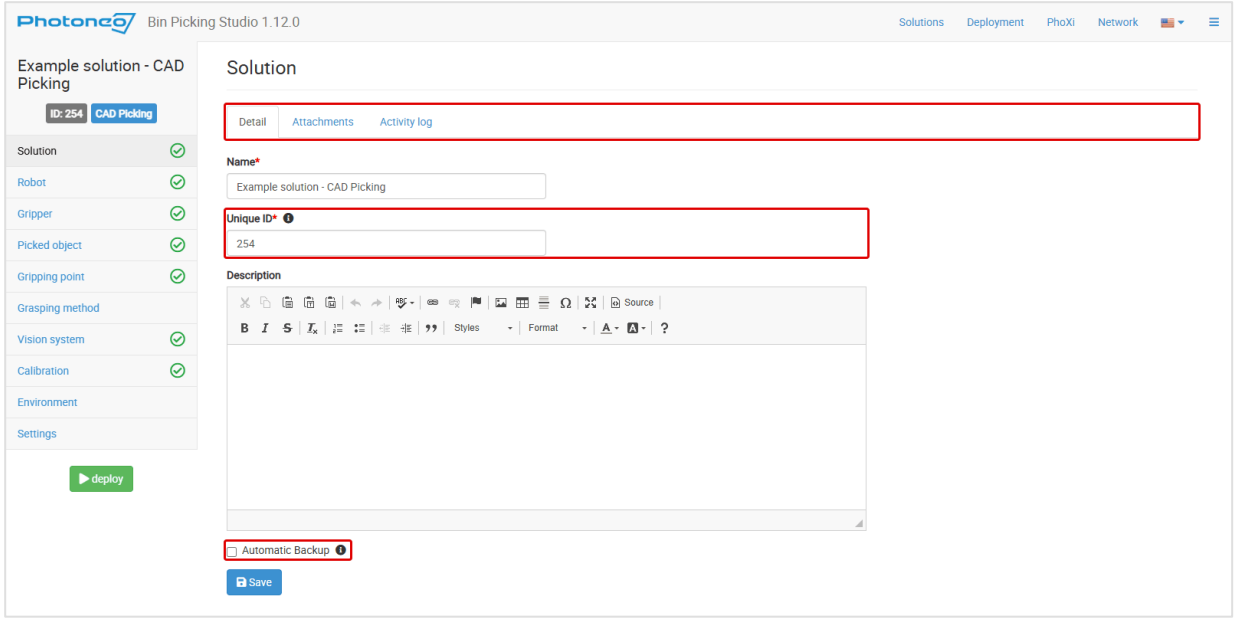
No license for the type? The dialog says "No valid license found for this solution type". Check the **Software modules** page.

To start faster, **Duplicate** an example solution, for example "Example solution - CAD Picking".

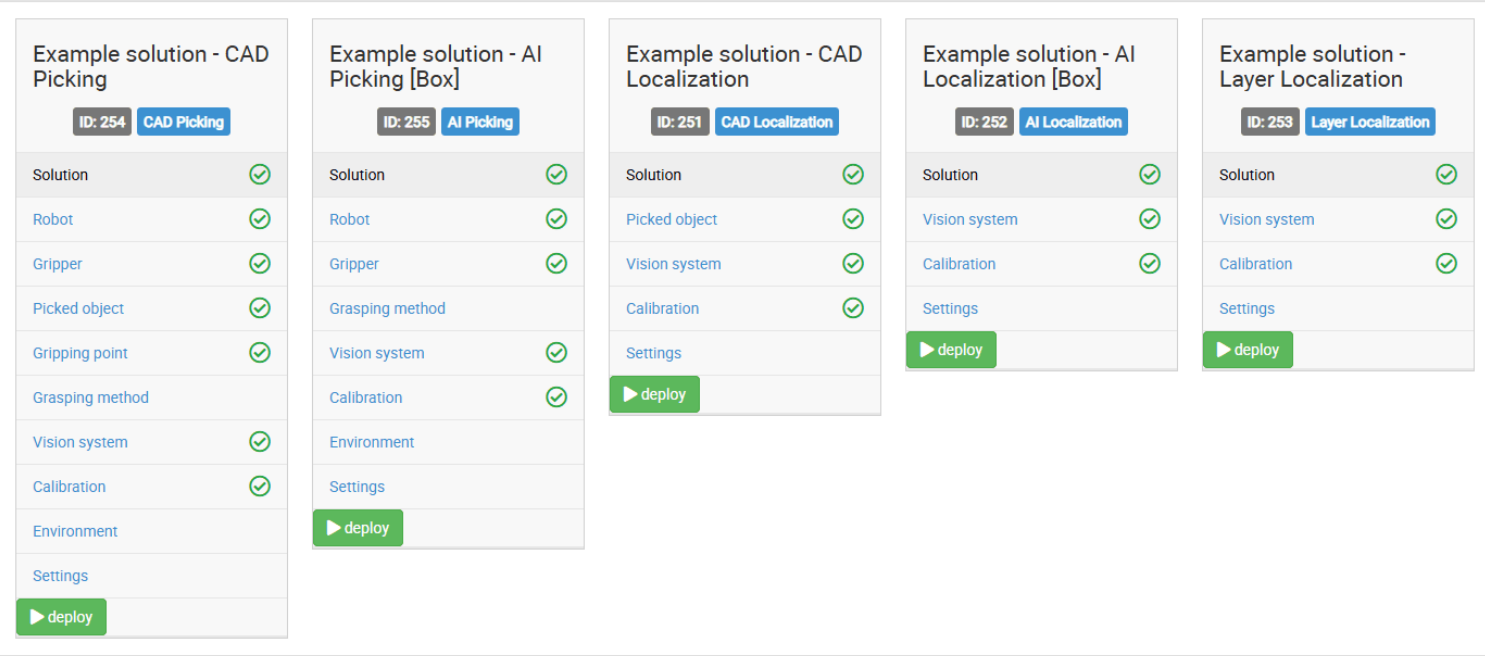


The solution page

- **Detail:** name, **Unique ID** (used in the robot's requests), description and **Automatic Backup**.
  - Automatic Backup exports the solution on deployment after a change, to ~/Desktop/Backup. It keeps 3 exports per solution and the 3 last deployed solutions.
- **Attachments:** robot program backups and other files. Up to 20 files, 15 MB each, 100 MB in total.
- **Activity log:** who changed what. Filter it and export it to CSV.



Setup steps depend on the solution type

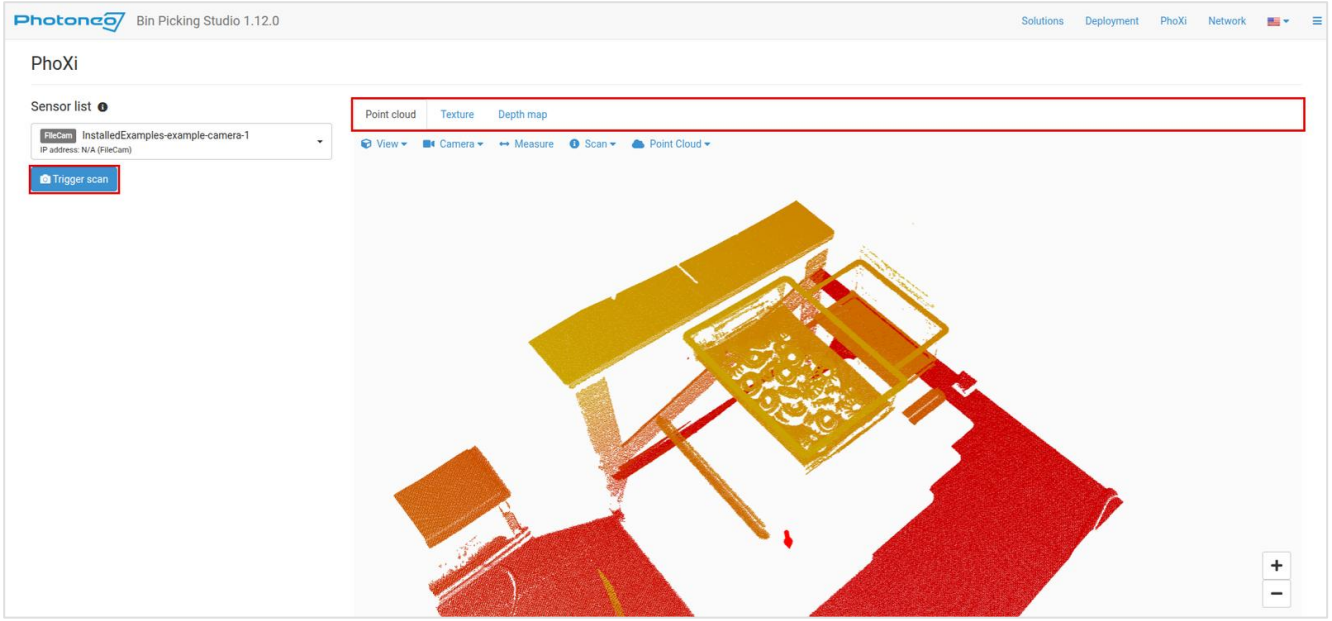


- The solution menu shows only the steps the type needs, in this order. Completed steps get a green check.
- BPS types add **Robot**, **Gripper**, **Grasping method** and **Environment**. CAD types add **Picked object**; CAD Picking also **Gripping point**.
- When all required steps are done, the **deploy** button appears.

PhoXi page

1. Select a sensor in the **Sensor list** (PhoXi, MCC, size S/M/L/XL, status).
2. Click **Connect**.
3. Click **Trigger scan**.
4. Check the **Point cloud**, **Texture** and **Depth map** tabs.

- Use it to check what the sensor sees. Full sensor configuration is done in **PhoXi Control** (over VNC or TeamViewer).
- A sensor connected here is also connected on the Environment page.



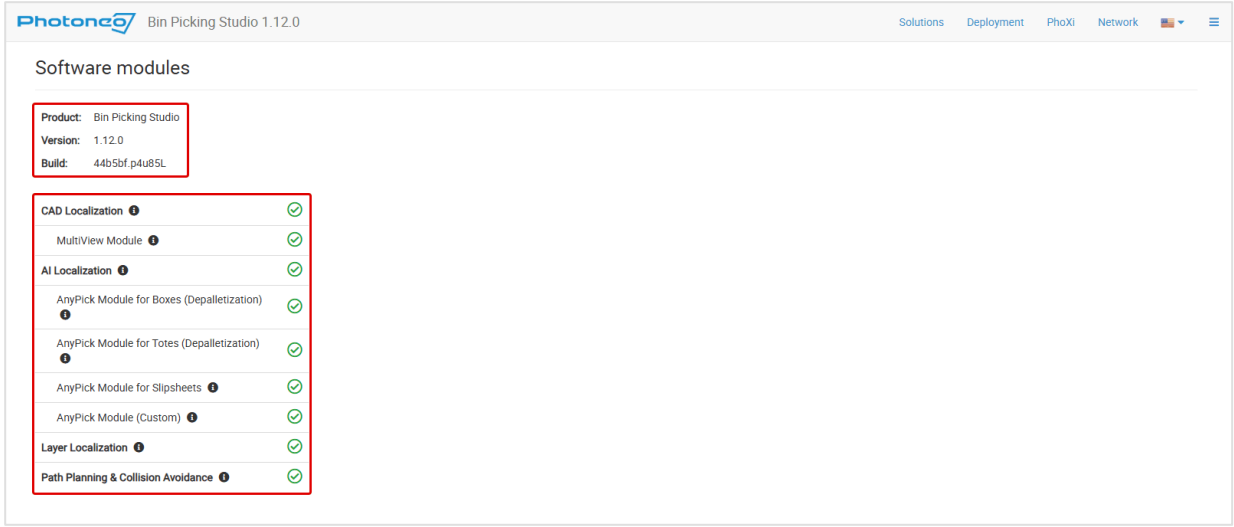
Software modules and Customization

Software modules

- Product, version and build of the Studio.
- Engines: **CAD Localization** (+ MultiView), **AI Localization** (+ AnyPick modules for boxes, totes, slipsheets, custom), **Layer Localization**, **Path Planning & Collision Avoidance**.
- To unlock a module, contact your sales representative.

Customization

- Primary color, logo, login page text, error page text and timezone.



03

Robot communication

IN THIS CHAPTER

Robot module

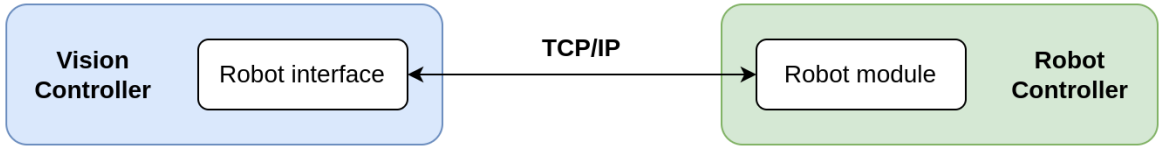
The two communication channels

A bin picking cycle

Other action requests

Robot module

- The **Robot module** is installed on the robot controller. There is one for each robot brand.
- It extends the robot's library with the **Photoneo robotic API**: a set of jobs or routines the robot programmer calls.
- Communication is **TCP/IP** only. Check the **Prerequisites** in the integration guide of your brand.
- The Studio never moves the robot. The robot programmer implements the gripper commands and sets speed, acceleration and accuracy.
- **Extreme caution** during commissioning: set safe speeds and be ready to stop the robot.



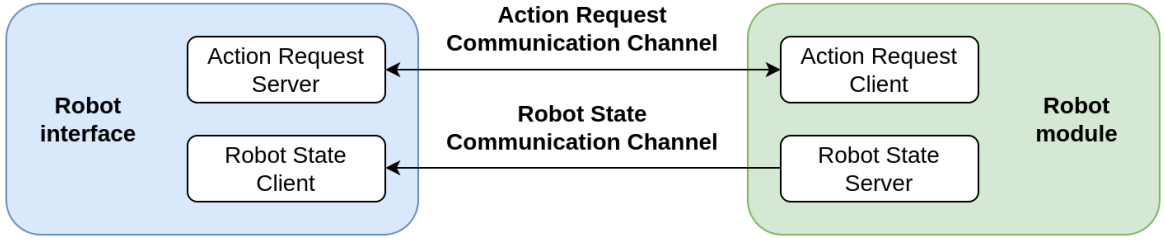
Two communication channels

Action Request channel

- Server on the vision controller (**ARS**), client in the robot program.
- The robot sends action requests; the Studio answers.

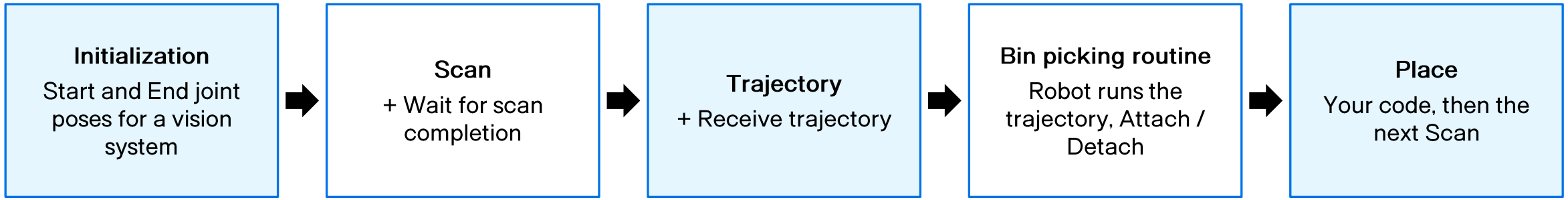
Robot State channel (BPS)

- Server on the robot (**Robot State Server**), client on the vision controller.
- Sends joint positions and the TCP pose. Used for teaching gripping points, calibration, hand-eye vision systems and the live robot view.
- Needed by BPS, optional for LS.



A bin picking cycle

The basic action requests of a BPS robot program.



- Send **Scan** as soon as the robot has cleared the scanning area. The Studio localizes and plans while the robot places the part.
- Send **Trajectory** just before the pick, so the Studio has the most candidates to choose from.
- **Always scan again** before the next Trajectory request. Objects may have moved during the last pick.
- If the cell can detect a failed pick, send **Pick-failed**. The object is then tried later.
- Most requests need the **vision system ID**.

Other action requests

Request	What it does
Change scene state	Switches the active state of the cell model (see Environment)
Change localization bounding box	Activates another bounding box before the next Scan
Get vision system status	Reports the state of a vision system
Trigger scan / Localize on the last scan	MultiView scans; localize again without a new scan
Add calibration point, Start / Save / Stop automatic calibration	Calibration from the robot program
Get running / Start / Change / Stop solution	Choose and start the solution from the robot program
LS: Scan, Get objects	Locator Studio: scan, then get the object poses

04

BPS solution setup

IN THIS CHAPTER

Robot: model, mount orientation, base frame offset

Gripper and tool point

Picked object

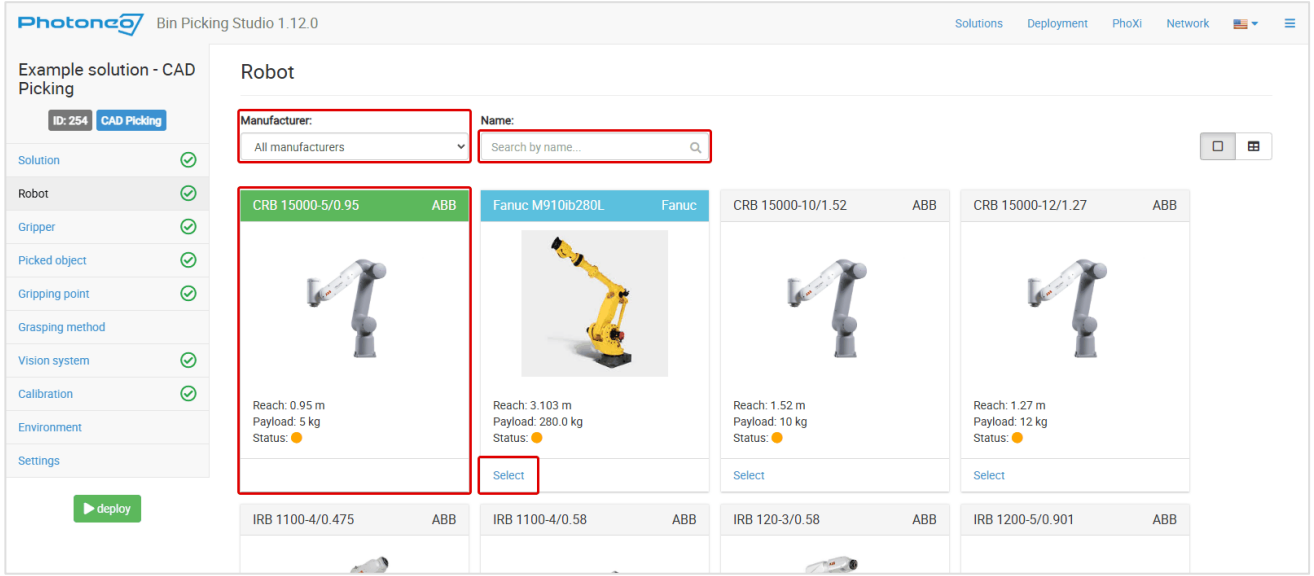
Gripping points and invariances

Teaching a gripping point, oriented placing

Grasping methods

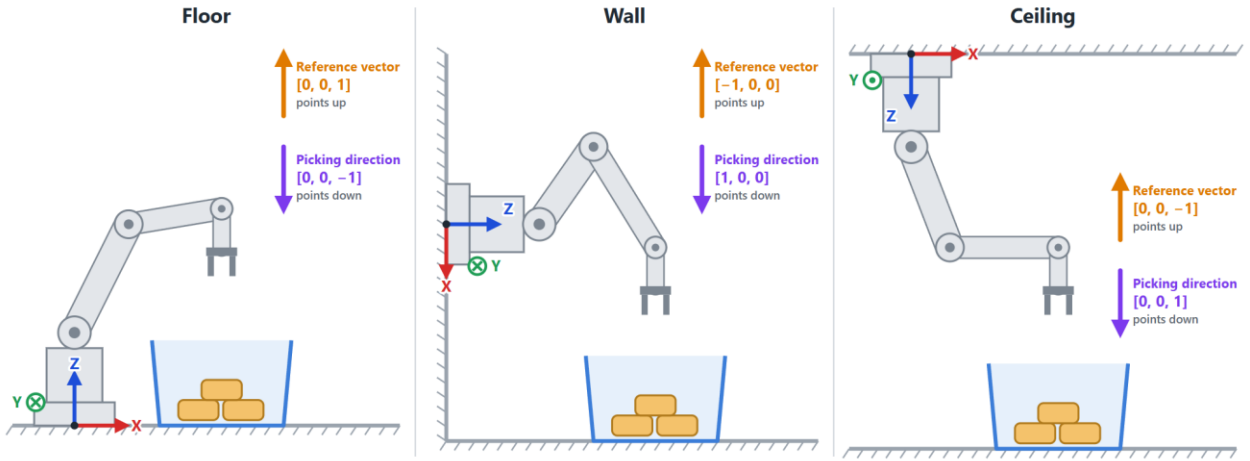
Robot step

- Find the model with the **Manufacturer** and **Name** filters and click **Select**. Custom robots have a blue frame.
- Changing the robot **resets the joint limits**, **removes the calibration point datasets** and **turns off automatic calibration**. Check the joint limits and calibrate again.
- The Studio checks only the **brand** of the connected robot, not the model. Select the correct model.



Robot mount orientation

Some settings are vectors in robot space. Change them for a wall or ceiling robot.

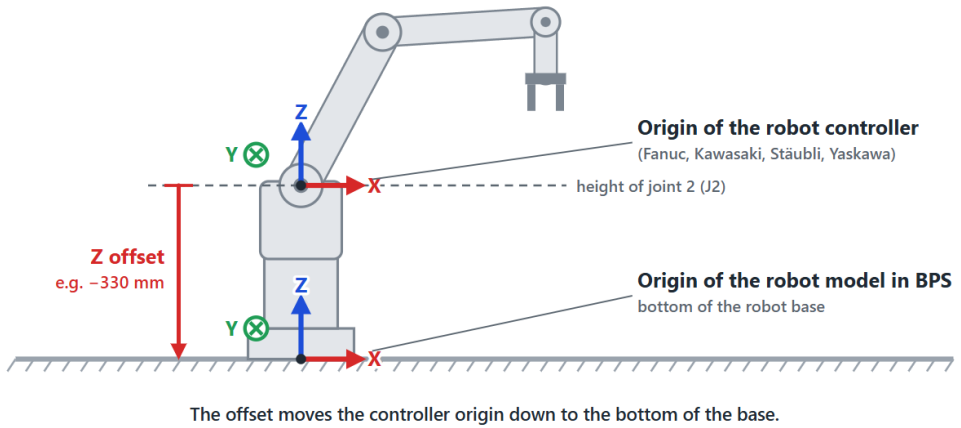


Setting	Floor	Wall	Ceiling
Gripper angle > Reference vector (points up)	0, 0, 1 (default)	−1, 0, 0	0, 0, −1
Picking priority > Picking direction vector	0, 0, −1 (default)	1, 0, 0	0, 0, 1
Path stage 100 mm up, origin Robot	0, 0, 100	−100, 0, 0	0, 0, −100

Wall values depend on how the robot is turned. Kawasaki uses the Y-axis as the forward direction. Rotate the Environment scene to match the real cell.

Base frame offsets

Fanuc, Kawasaki, Stäubli and Yaskawa



The offset moves the controller origin down to the bottom of the base.

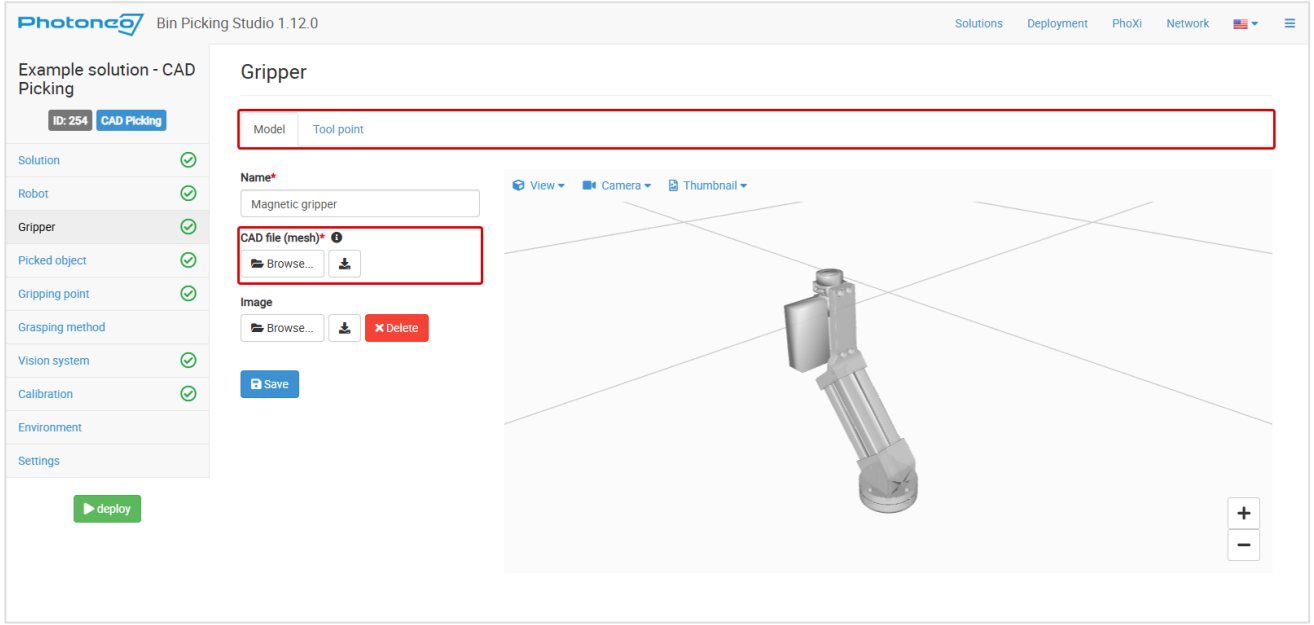
- The controller's origin is at the height of **J2**; the BPS robot model has it at the **bottom of the base**.
- Without the Z offset, calibration fails. Set it once, before calibration.

Brand	Where to set the Z offset
Fanuc	Menu > Setup > Frames > User Frame 1
Kawasaki	Aux Functions > Advanced Settings > Base Coordinates
Stäubli	Data > photoneo_common > frame > fBaseLink
Yaskawa	Robot > User Coordinate > USER COORDINATE 1

Note: The docs list the Z offset of every robot of these brands. Changing a base or user frame can move positions already taught in it.

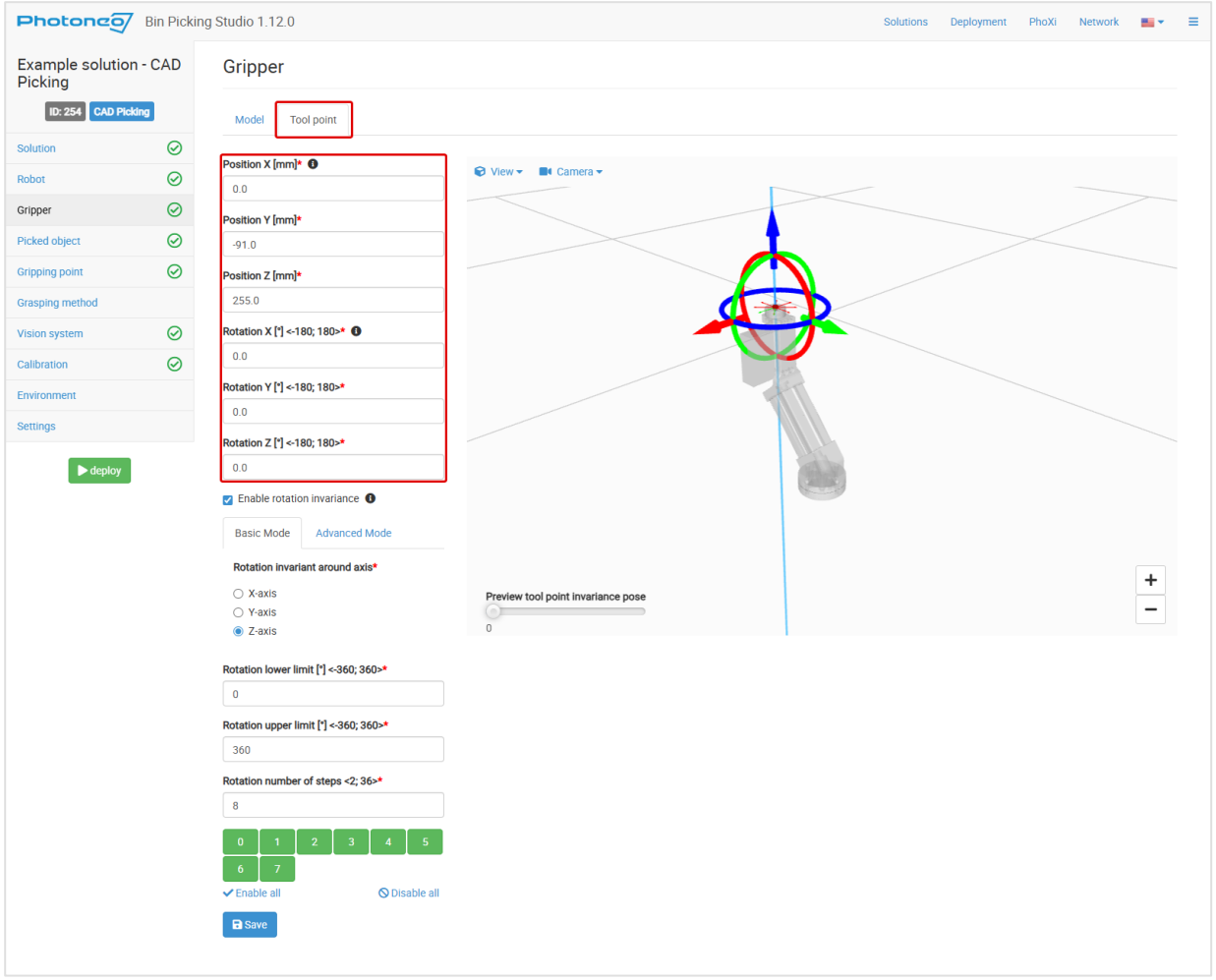
Gripper

- Grippers are **shared between solutions**. Default grippers cannot be edited: duplicate them first.
- **CAD file (mesh)**: STL in **millimeters**. Include extra gear such as valves and split boxes, so path planning avoids them.
- The origin of the model must be the **mounting point** on the robot flange.
- Selecting another gripper **removes the calibration point datasets**. Calibrate again.



Tool point

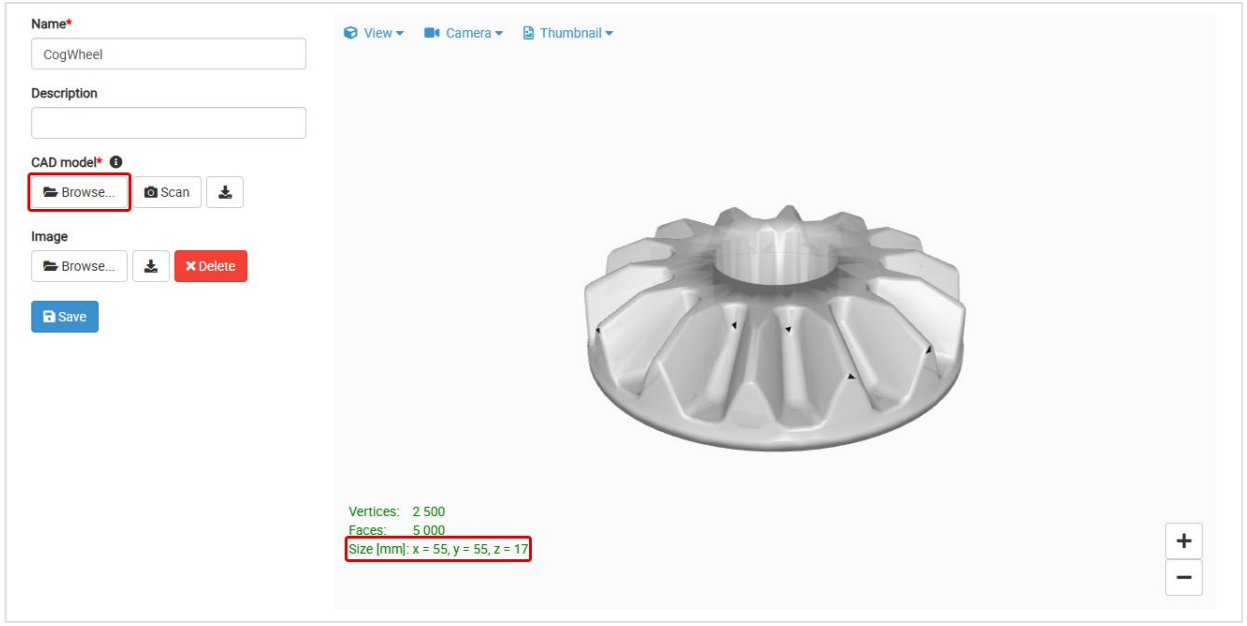
- The point on the gripper that meets a gripping point of the object.
- BPS uses **its own tool point**, not the TCP set on the robot controller.
- Defined relative to the origin of the gripper model. The **Z-axis points out of the gripper**.
- **Enable rotation invariance**: axis, lower and upper limit, and 2–36 steps. Preview the poses with the slider.



Picked object

CAD Picking (and CAD Localization). AI types have no Picked object step.

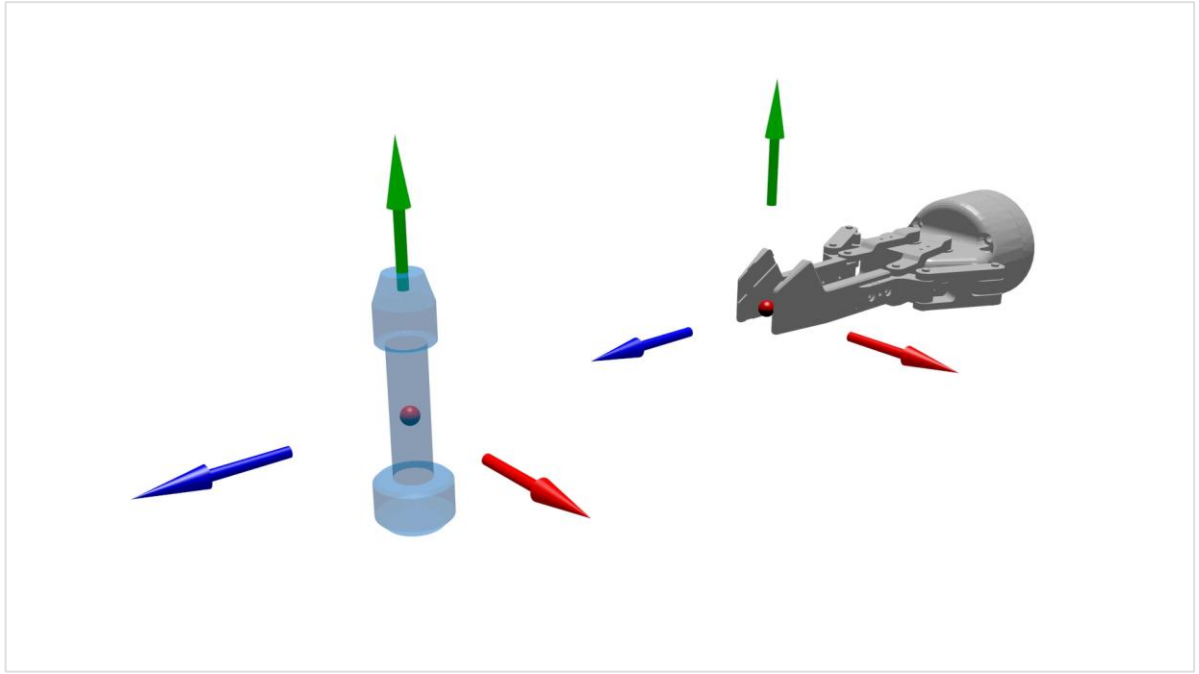
- **CAD model**: STL or PLY in mm. **Browse...** uploads a file; **Scan** takes the model from a connected sensor.
- Check before you upload:
  - units in **millimeters** (check **Size [mm]**)
  - no holes, broken faces or extra parts
  - **normals point outward**
  - **origin** inside or on the object
- The **Mesh tools** tab: cropping, simplification, filtering, normals and origin.
- Give the object a clear description of its model or version.



Tip: Use a **CAD file** for small rigid parts in random orientation. Use a **scan + mesh tools** for big, semi-oriented parts always seen from one side.

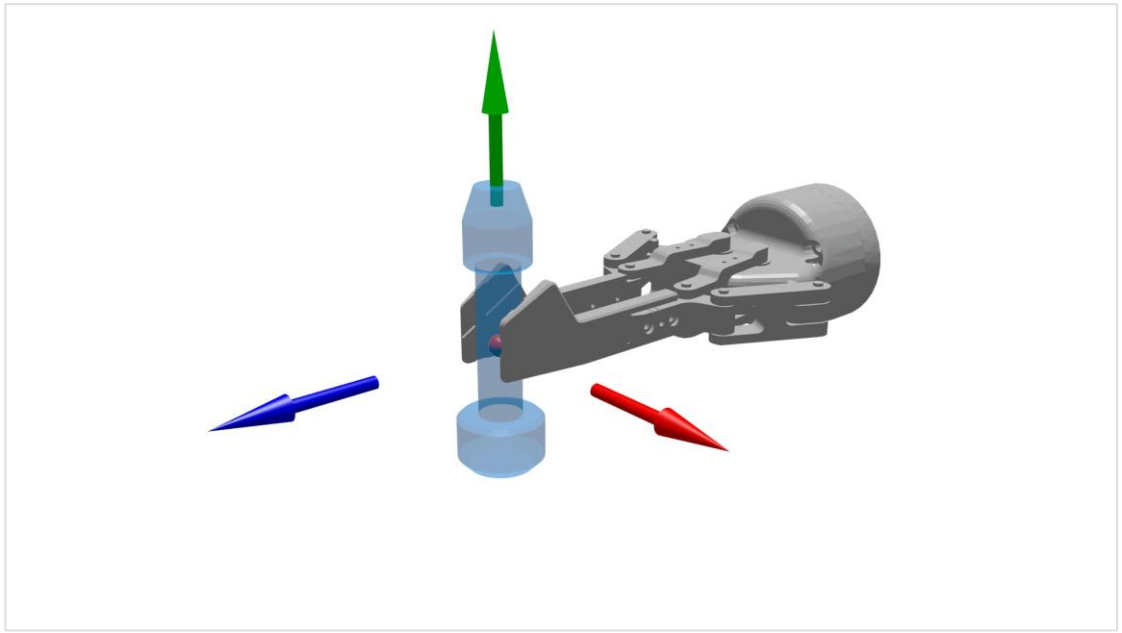
Tool point meets gripping point

- **Tool point**: a point on the gripper (Gripper step). One per gripper.
- **Gripping point**: a point on the object where it may be grasped, relative to the origin of the object model. Any number per object.
- During the grasp, the two coordinate systems are **laid on top of each other**.
- The gripping point's Z-axis points into the object.
- In AI Picking, the Studio places the gripping points itself.



Invariances

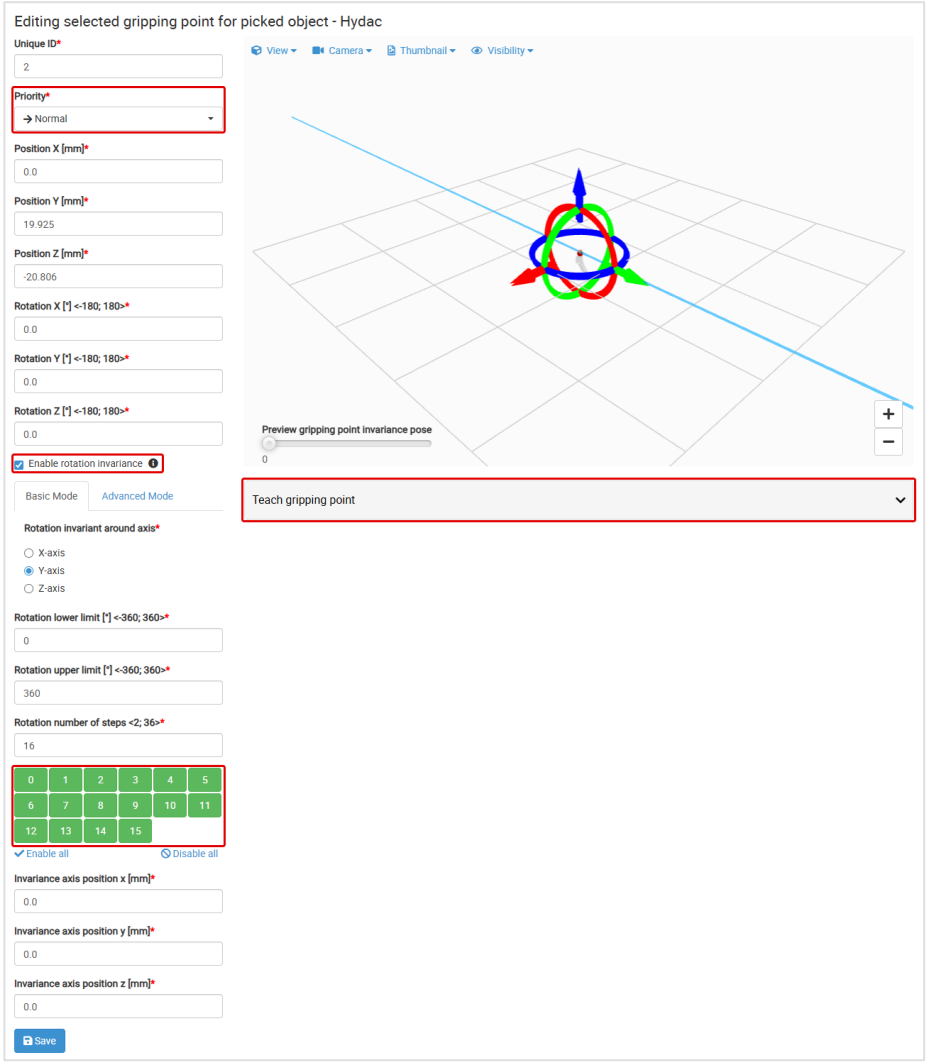
- An **invariance** is a turn around an axis after which the grasp is the same. It gives the robot more ways to grasp, without more gripping points.
- Object (green): a cylinder grasped from the side is the same at any angle.
- Two-finger gripper (blue): the same at 0° and 180°.
- Tool point and gripping point invariances **multiply**: 8 × 2 = 16 grasps.
- **Avoid the same axis on both**: a suction cup over a gear gives 8 × 8 = 64 combinations, mostly the same grasp. It slows down picking.



Gripping point step

1. Select the **Picked object** and click **Add new gripping point**.
2. Set the position and rotation, or drag the marker in the 3D viewer.
3. Optional: **Enable rotation invariance** and disable steps the gripper cannot use.
4. Click **Save**.

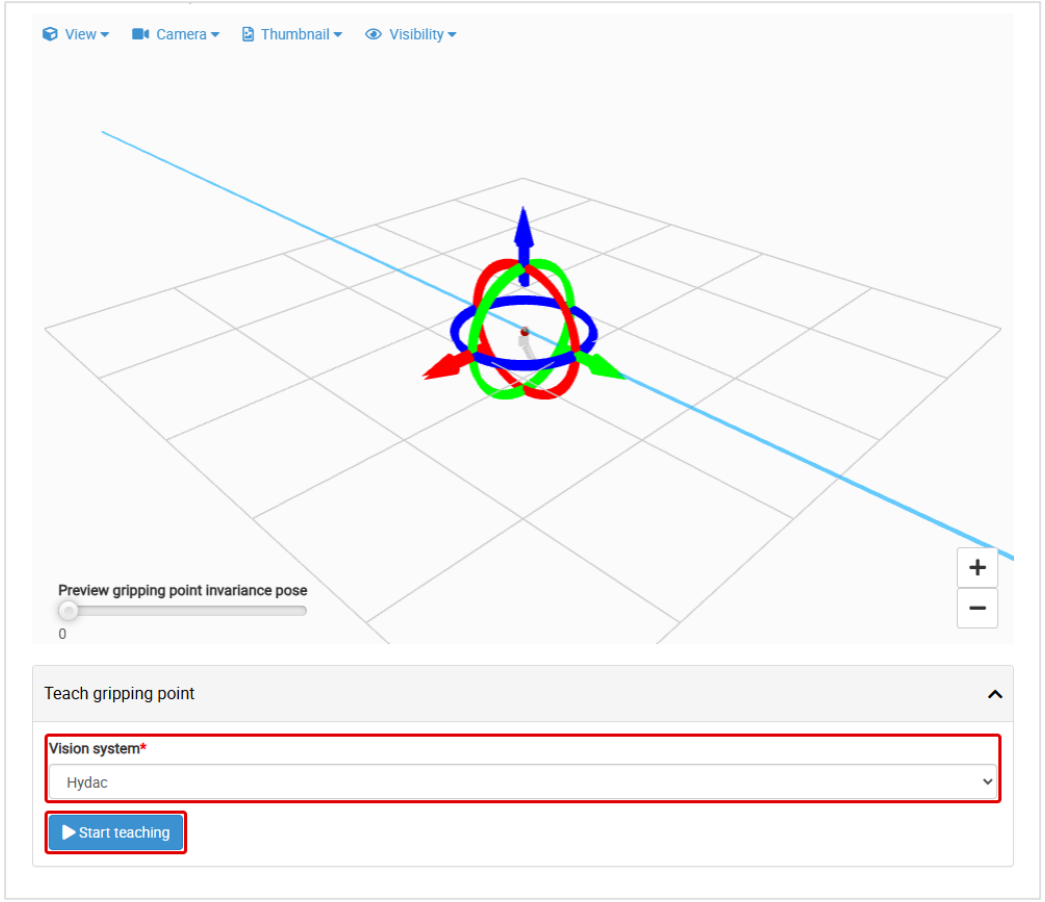
- **Priority** (Low / Normal / High) is used when the picking priority method is **Gripping Point Priority**.
- Disable a gripping point to test without it.



Teaching a gripping point

1. Open the gripping point and the **Teach gripping point** panel. Select the vision system and click **Start teaching**.
2. Place one object in view and scan.
3. Move the robot with the teach pendant until the gripper grasps the object the way you want.
4. Reload the robot pose and click the first row of the table (the closest object).
5. Finish teaching, check the result and **Save**.

- Needs a localization profile, a calibration, a sensor **not** on the robot arm, a running **Robot State Server** and a correct tool point.



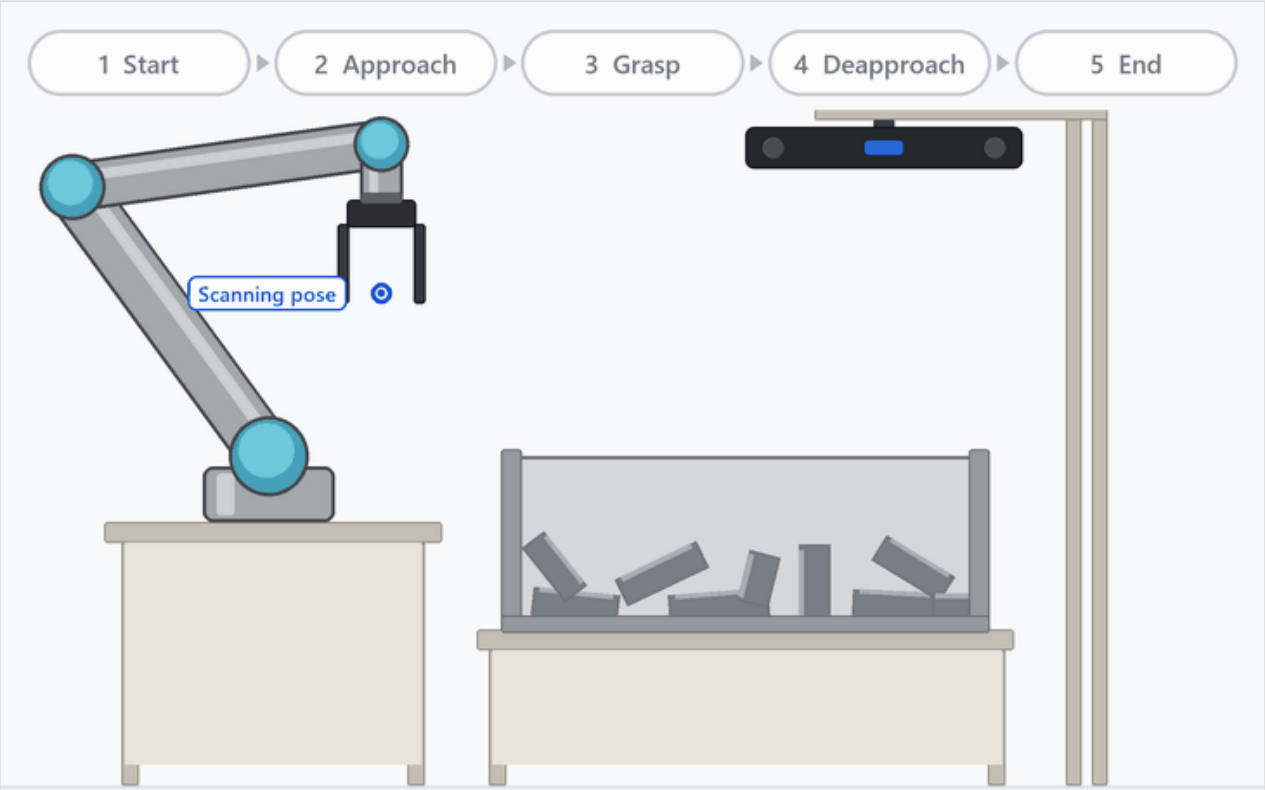
Oriented placing

- With each trajectory the robot receives the **placing data**:
  - tool point invariance
  - gripping point ID
  - gripping point invariance
- Example: 2 tool point invariances × 3 gripping points × 4 invariances = **24** grasping configurations.
- Teach only a few base placing positions and compute the rest from the placing data.
- To teach one configuration, temporarily disable the other gripping points and invariances.



Grasping methods

- A grasping method is **how the robot picks one object**: the waypoints, the moves between them and the robot procedures.
- **Path stage**: a waypoint, the move to it and an optional procedure.
- **Bin picking routine**: all path stages, from **Start** to **End**.
- A new solution has the **Basic Grasping Method**. Add or duplicate methods when vision systems need different paths.
- A vision system uses the **current default** or one named method.



Scanning pose: the robot is out of the scanner's view, the bin is scanned

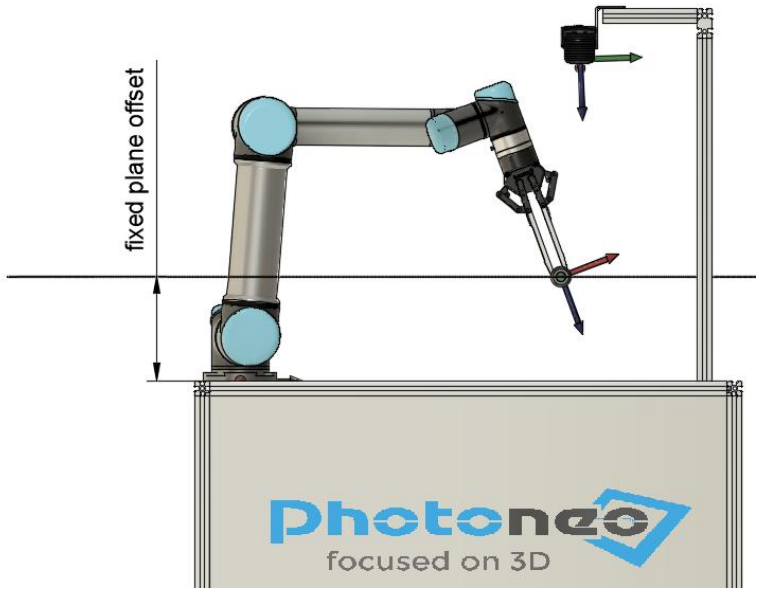
The five mandatory path stages

Stage	What happens	Pose comes from
1 **Start**	Prepares the robot and gripper for a new pick	The robot program (Initialization request)
2 **Approach**	Gripper moves to an offset from the found gripping point, usually a few cm back along its Z-axis	Offset from the found gripping point or the robot base
3 **Grasp**	Gripper moves to the found gripping point; the procedure closes the gripper (e.g. Attach)	Found gripping point and the tool point
4 **Deapproach**	Gripper moves away, usually tens of cm up along the robot's Z-axis	An offset (see next slide)
5 **End**	Gripper moves to the end pose; post-picking operations	The robot program (Initialization request)

- You can add **custom path stages** between them. Drag a custom row to move it.
- Moves to Grasp, to Deapproach and to every stage between Approach and Deapproach are **always linear**.

Origin of the relative offset

Origin	The waypoint is	For
Found gripping point	Offset in the gripping point's system. Its Z points into the object, so use a negative Z	Approach, Deapproach, custom
Robot	Offset from the previous waypoint in robot base coordinates. +Z is up (floor robot)	Approach, Deapproach, custom
Fixed plane	On a plane parallel to XY, XZ or YZ of the robot base	Deapproach, custom
Start / End	On the way towards the Start or End pose (or a helper point)	Deapproach, custom

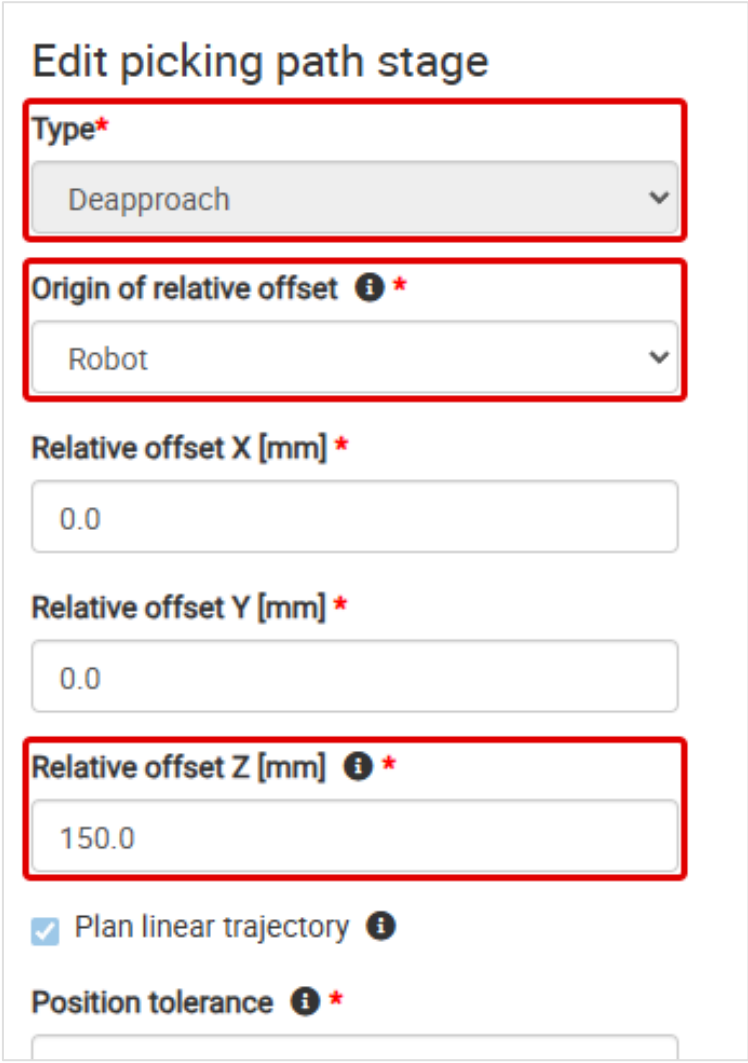


Fixed plane: the same height for every part

Tip: Deep bin? Use a **Fixed plane** Deapproach so every part leaves the bin at the same height. Near the walls? Use **Start / End** as the origin so the part moves towards the middle of the bin.

Path stage settings

- **Type**: **Relative** to an origin, or **Fixed** position in robot space (keeps the previous orientation).
- **Plan linear trajectory**: a straight move. Otherwise the planner may plan a joint move.
- **Position tolerance**:
  - **Precise**: the robot reaches the waypoint exactly.
  - **Approximative**: the robot blends into the next move. The procedure may run while moving.
- **Procedure name**: **Attach**, **Detach**, **User 1–3**, defined in the robot program.



05

Vision system and localization

IN THIS CHAPTER

Vision system configuration

Scanning profiles

Localization engines

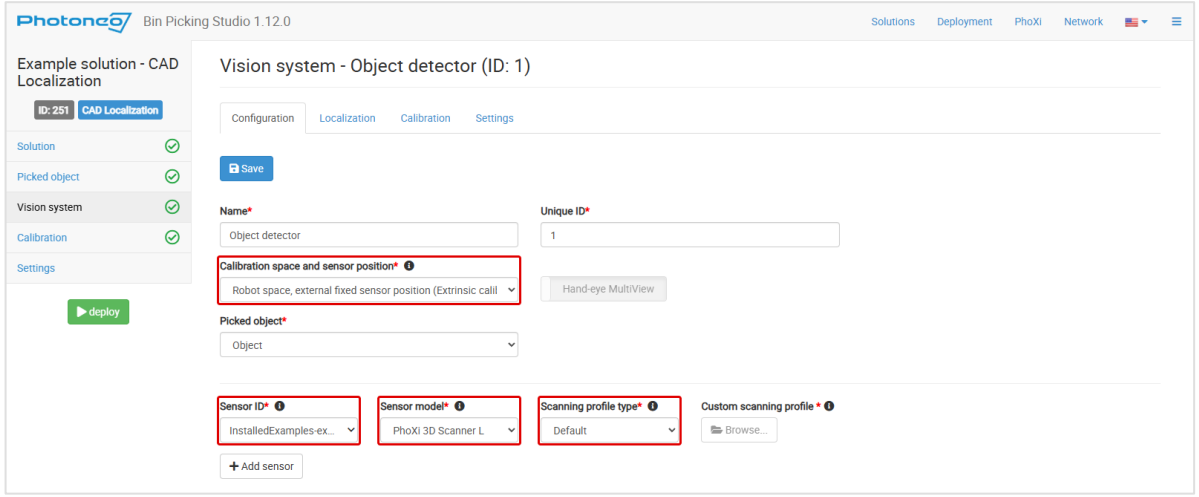
CAD localization and CAD Model Detail

AI and Layer localization

MultiView and bounding boxes

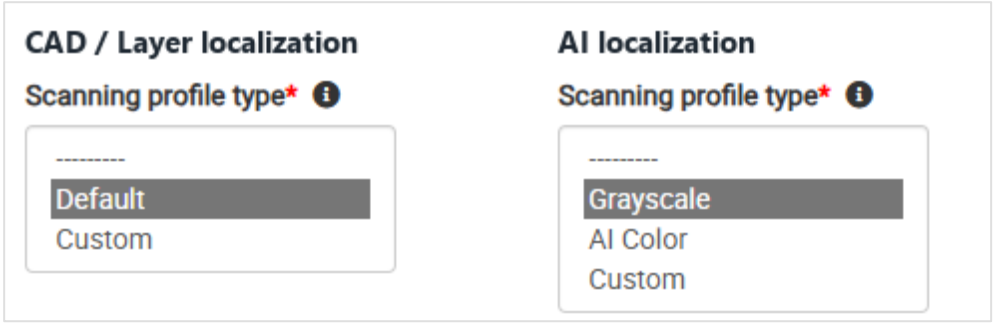
Vision system

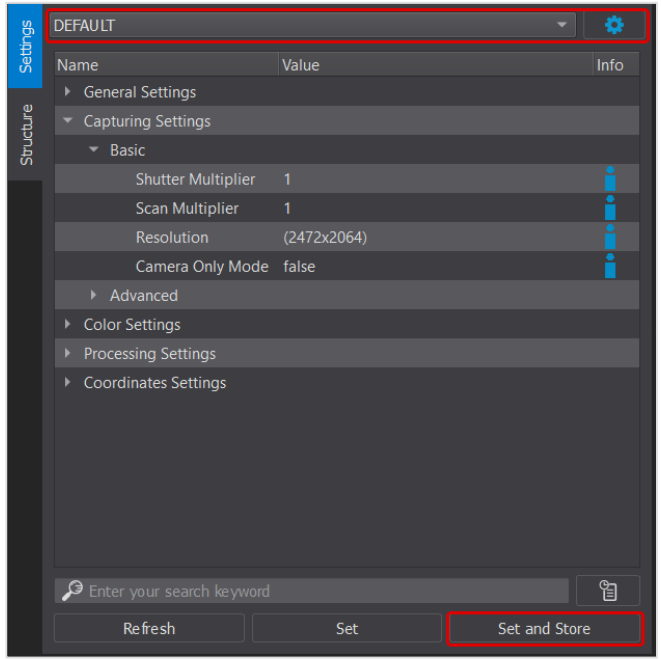
- A vision system ties together the **3D sensor**, its **calibration**, the **picked object**, the **grasping method**, the **localization profile** and its own settings.
- **Calibration space and sensor position**: extrinsic (fixed sensor), hand-eye (sensor on the flange) or marker space (LS only).
- Sensors: **PhoXi 3D Scanner**, **MotionCam-3D (Color)**, **Alpha 3D Scanner**. Check the **Sensor model**: it is a collision object.
- One sensor can serve several vision systems. MultiView uses several sensors in one.
- The robot's requests refer to the **vision system ID**.



Scanning profiles

- **Default** (CAD / Layer) and **Grayscale** (AI) profiles suit a wide range of objects. **AI Color** is for sensors with a color camera.
- Best results: a **Custom** profile. Clone a factory profile in PhoXi Control, tune it, **Set and Store**, export the .phop file and upload it to the vision system.
- Calibration uses a built-in profile. In hard conditions, turn on **Use custom scanning profile for calibration**.
- The profile is uploaded to the sensor when a process starts, under its **file name**:
  - different profiles on one sensor: use **unique names** (add the vision system ID)
  - the same profile: use the **same file name** to skip switching
- The profile "Default" in PhoXi Control is **not** the Studio's Default.





Localization engines

An engine finds each object and its pose in the point cloud.

Engine	Finds objects by	Solution types	Ideal for
CAD Localization	The 3D (CAD) model	CAD Localization, CAD Picking	Belt picking, assembly, bin picking
AI Localization	Neural networks	AI Localization, AI Picking	Simple and complex depalletization
Layer Localization	The top layer of a stack	Layer Localization (LS only)	Picking whole layers
Hybrid CAD + AI	CAD model and neural networks	On demand only	The most complicated cases

Auxiliary features

- **CAD Model Detail**: model features and feature alignment for nearly symmetrical objects.
- **MultiView**: merges scans from several viewpoints before CAD localization.
- **Bounding boxes**: limit the localization to regions of interest.

CAD localization

1. **Segmentation**: the scene is split into segments; too small and too large ones are left out.
2. **Searching**: model points are matched to scene points; each match is a candidate.
3. **Fine alignment**: candidates are refined with ICP.
4. **Checks**: enough **Overlap**, occlusion and model feature checks.

- Stops on a **stop criterion**: time limit, number of objects, or time without a new object.
- The profile (.plcf) holds the model and all settings. Nothing is saved until **Save configuration**.
- **Scan & Locate** takes a new scan; **Locate** tests changed settings on the same scan.

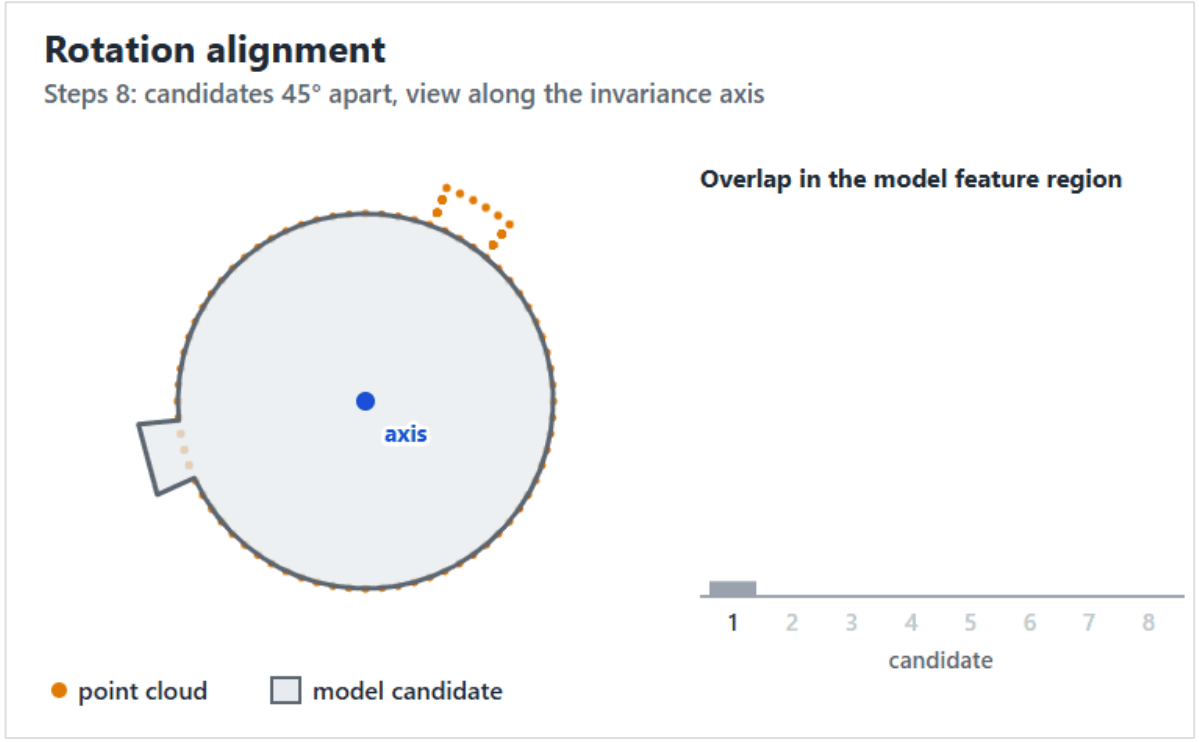


CAD Model Detail

- For objects that are **almost symmetrical**, where a small detail decides the pose.

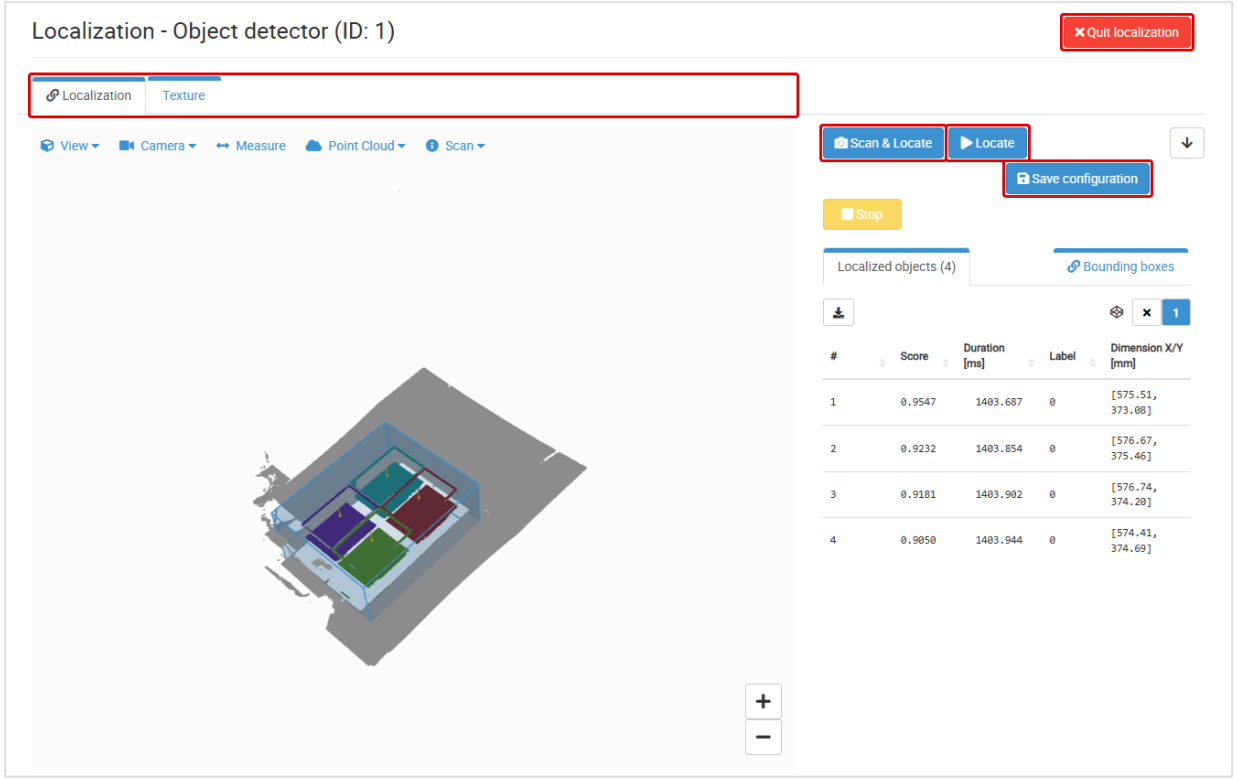
1. **Model feature**: mark the detail on the model with a region of interest.
2. **Feature alignment**: tell the engine how the object can be wrongly turned (**Rotation**, **Flip and Rotation**) or shifted (**Translation**). It tries those poses and keeps the best match.

- The feature must be **clearly visible in the scan**.
- Optional: try tuning the localization first.



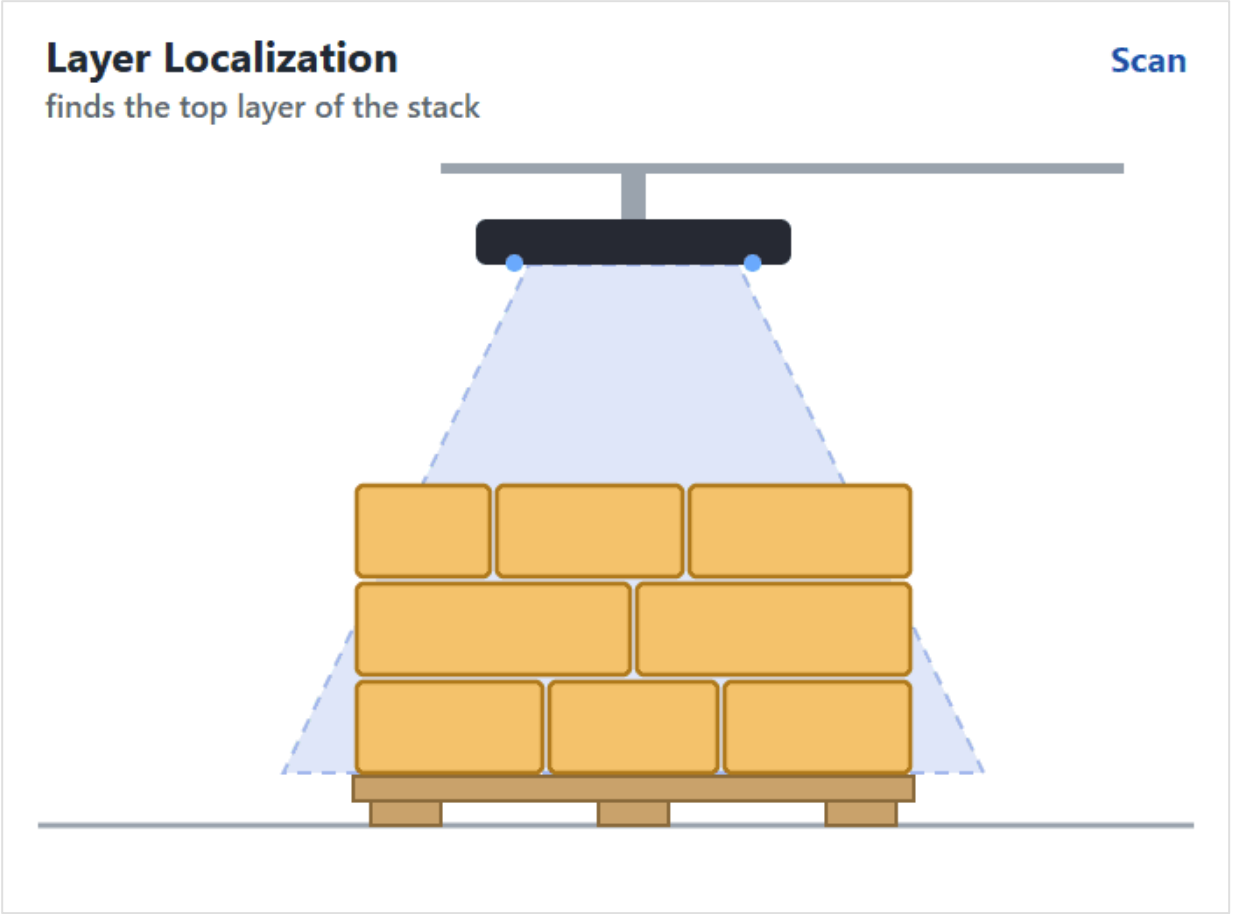
AI localization

- A **neural network** finds the objects. No 3D model, so no Picked object step.
- Networks: **AnyPick** modules for boxes, totes, slipsheets, or custom (retrained for your objects).
- The profile is a .pacf file, edited like the CAD profile.
- **Initial gripping point placement method**:
  - dedicated: *Center of Box*, *Center of Tote*
  - general: *Centroid*, *Largest Inscribed Circle*, *Keypoint from NN*
- Reports the X/Y dimensions and the angle of each object.



Layer localization (LS)

- Finds the **top layer** of a stack and reports its pose.
- Ideal for picking whole layers.
- Steps: **Vision system**, **Calibration**, **Settings**. No Picked object.
- Profile: a .pdcf file on the Localization tab.
- Layer settings: **Constraint – Object dimensions** and **Constraint – Points above localized layer**.
- The robot gets the layer with the **Get objects** request.



MultiView

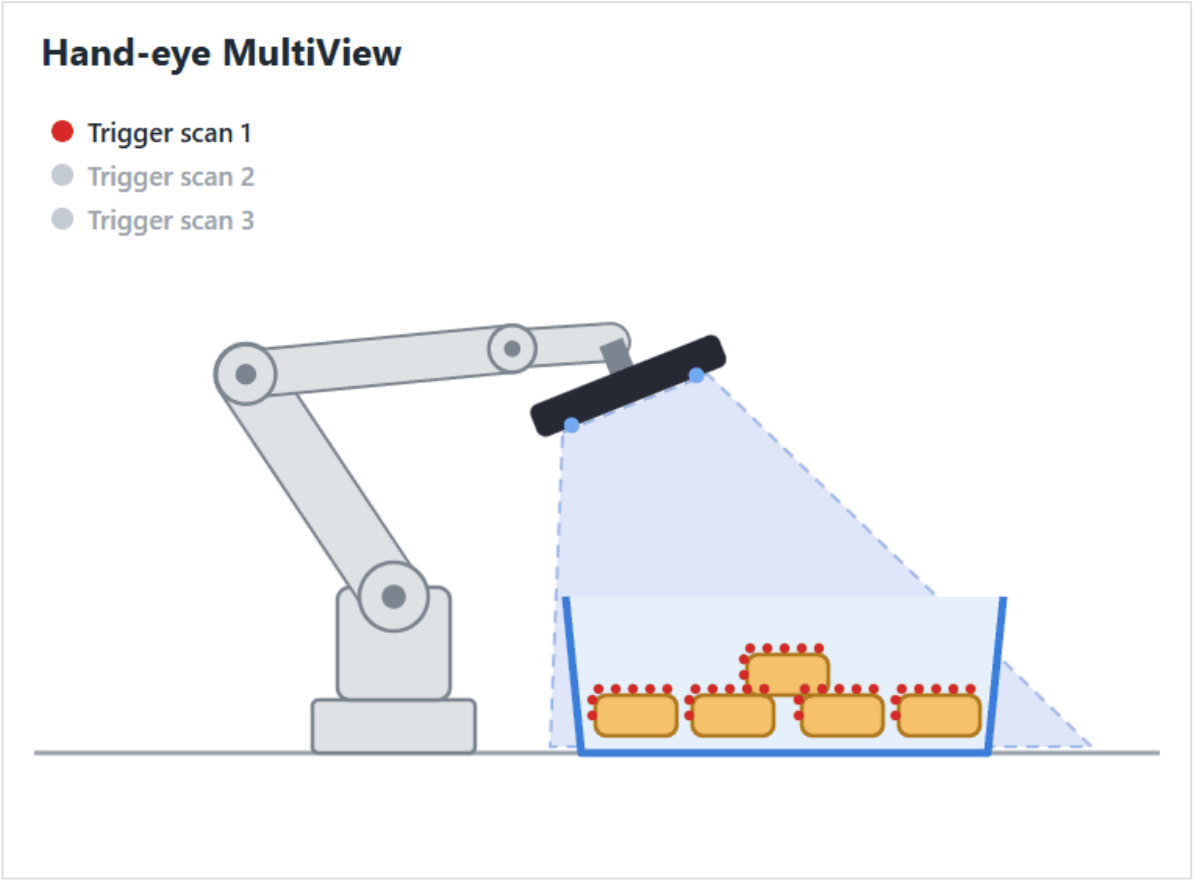
- Merges scans from several viewpoints into one point cloud, so difficult scenes are more complete.

Static MultiView

- Several fixed sensors. Calibrate the primary sensor to the robot, then **Calibrate together** for the others.

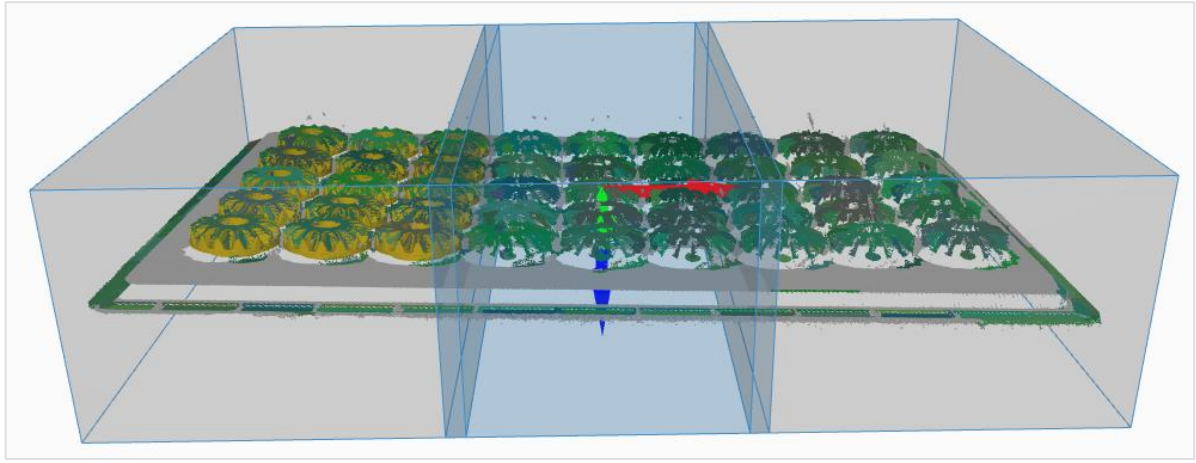
Hand-eye MultiView

- One sensor on the robot. **Static** mode: the robot stops for every scan. **Dynamic**: only for the first (MotionCam-3D in Camera mode).
- Robot program: **Trigger scan** for each view, then **Scan** to merge and localize.



Bounding boxes

- A region of interest: the engine looks for objects **only inside the active box**.
- Use it for several bins in one scene, to remove false positives, and for speed.
- Only **one box is active** at a time (blue in the viewer). **X** turns boxes off.
- The box with the **Initial** flag is active after deployment.
- The robot switches boxes with **Change localization bounding box**, sent **before** the Scan request.
- Export and import as .pbbcf files.



06

Calibration

IN THIS CHAPTER

What calibration does

Calibration types

Before you start

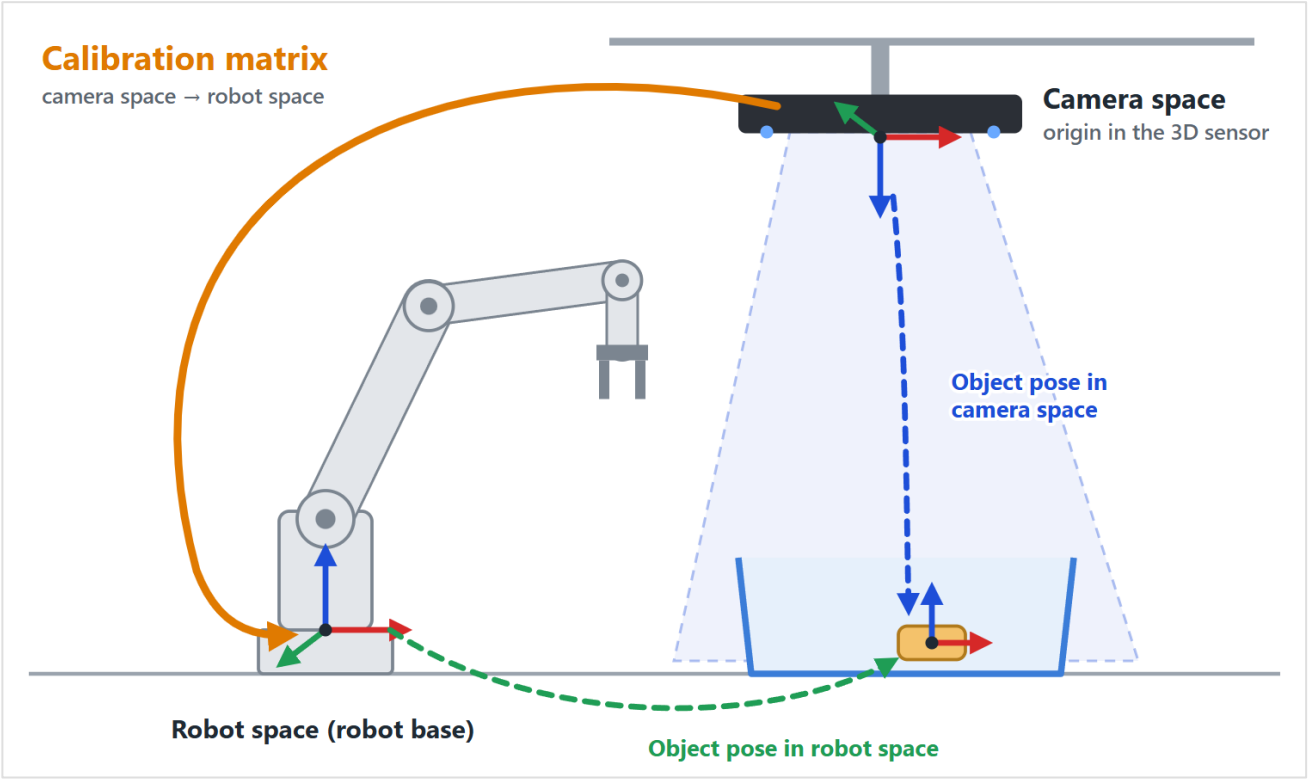
Calibration modes

Calibrating step by step

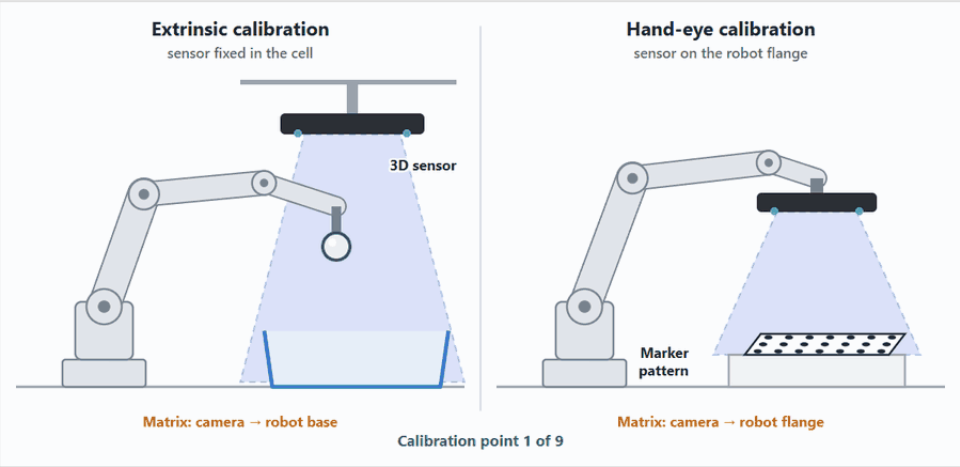
Verification and tips

What calibration does

- The sensor reports poses in **camera space**; the robot works in **robot space**.
- Calibration finds the **calibration matrix** between them. The Studio applies it to every localized pose.
- Poor calibration is usually the **largest source of picking errors**.
- If the sensor, its mount or the robot moves, **calibrate again**.



Calibration types



Type	Sensor	Tool	Good to know
Extrinsic	Fixed, usually above the bin	Ball on the flange or gripper	Robot may move while the sensor scans
Hand-eye	On the robot, behind the last joint	Marker pattern	Needs the TCP for every scan; robot stands still
Marker space (LS)	Usually fixed	Marker pattern	One scan; then a user frame at the marker origin
Multi-sensor	Several fixed sensors	Marker pattern	Secondary sensors to the primary (Static MultiView)

- The type follows from **Calibration space and sensor position** in the vision system.
- Both tools, the calibration ball and the calibration board, come with the package.

Before you start

- **Network**: the vision controller reaches the sensor and the robot.
- **Vision system**: Sensor ID and calibration space set.
- **Ball**: firmly mounted, mostly visible, round, smooth, not shiny. Larger is better.
- **Marker pattern**: in reach and in view, perfectly flat, must not move.
- **Base frame offset** set (Fanuc, Kawasaki, Stäubli, Yaskawa).
- **Do not change the tool** during calibration.
- **Robot State Server** running (BPS). Without it you can still calibrate manually and type in the TCP.

Tool to select	Extrinsic	Hand-eye
BPS calibration	Any tool	tool0
BPS localization	Does not matter	Does not matter; RSS must run
LS calibration	Any tool	Any tool
LS localization	Does not matter	Same tool as in calibration; send its TCP in Scan

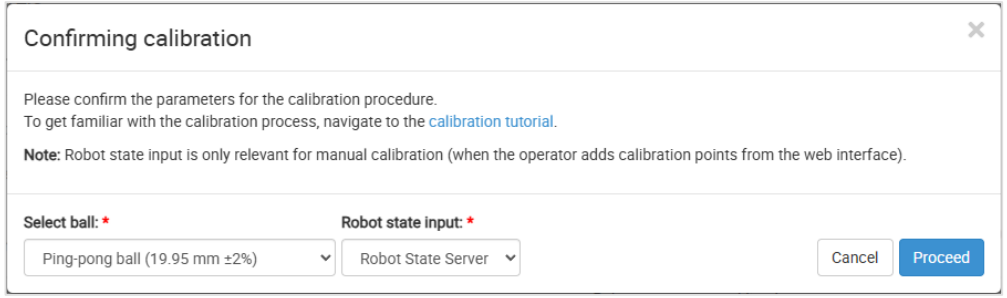
Calibration modes

Mode	Start and save	Add calibration points	Robot pose comes from
Manual	Operator, in the Studio	Operator: Add calibration point	BPS: Robot State Server or typed in. LS: typed in
Semi-automatic	Operator, in the Studio	Robot program: Add calibration point request	BPS: Robot State Server. LS: the request
Automatic	Robot program: Start / Save / Stop requests	Robot program	BPS: Robot State Server. LS: the request

- The Robot module has a **template program** for semi-automatic calibration. Teach your own poses in it. The automatic requests are already there, commented out.
- **Automatic calibration** repeats an operator calibration with the same poses. It needs at least 9 points and the **Minimal allowed accuracy**.
- **Simulated calibration** (Environment > Vision > Edit) is for testing without a robot. Such a solution **cannot** run in Production mode.

Calibrating step by step

1. Open the **Calibration** step, check the sensor status and click **Start calibration**.
2. Confirm the ball (extrinsic) and the **Robot state input** (Robot State Server or Operator).
3. Move the robot to a pose and **Add calibration point**. A found ball is green on the Texture tab.
4. Repeat until **9 points**, with poses as different as possible.
5. Check the **Calibration accuracy**. Replace bad points and points shown as *N/A*.
6. Verify, then **Finish and save result**.

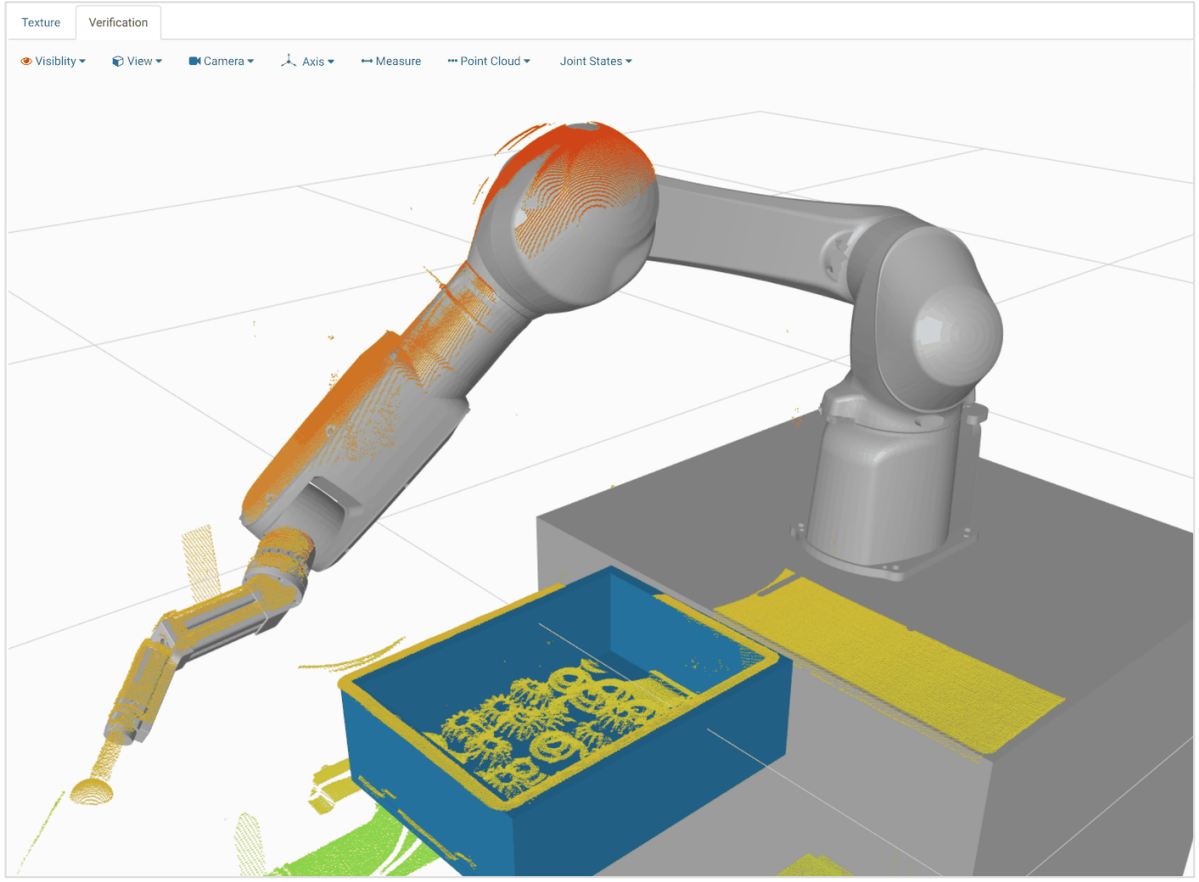


Verify the result

- From the 4th point (extrinsic) or the 6th (hand-eye), the **Verification** tab shows the point cloud with the robot model.
- Move the robot into view and **Trigger scan**. A good calibration: the cloud **lies on the robot model**.
- Accuracy is in mm, lower is better. A few mm can still be fine.

Tips

- Large joint differences between poses; cover the whole bin.
- Cloud shifted along Z? Check the **base frame offset**.
- Ball visible but not found? It may be outside the scanning volume.



07

Environment

IN THIS CHAPTER

Scene: the collision model of the cell

Robot tab: virtual robot and joint limits

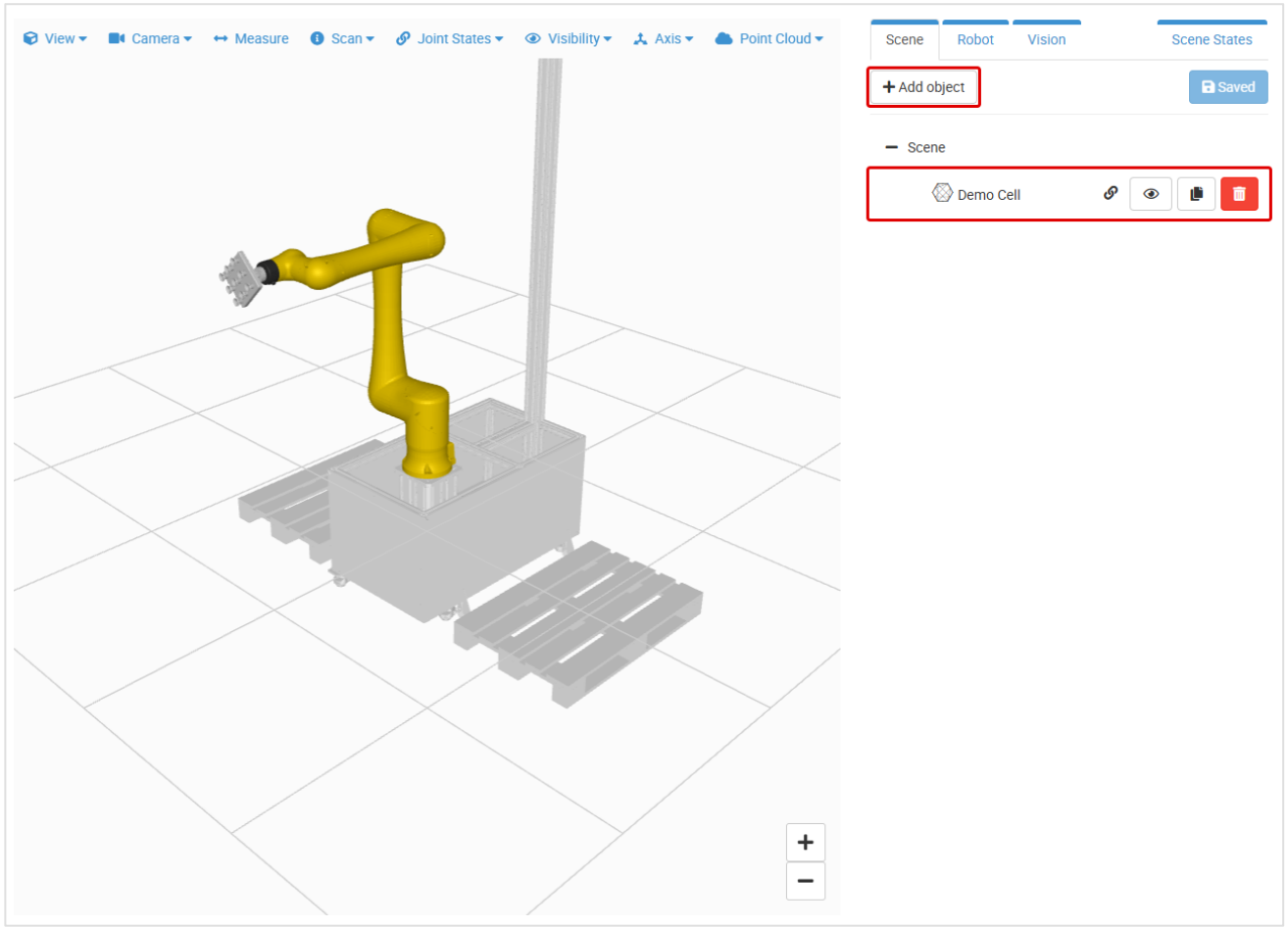
Robot working space

Vision tab: check the scene with a scan

Scene states

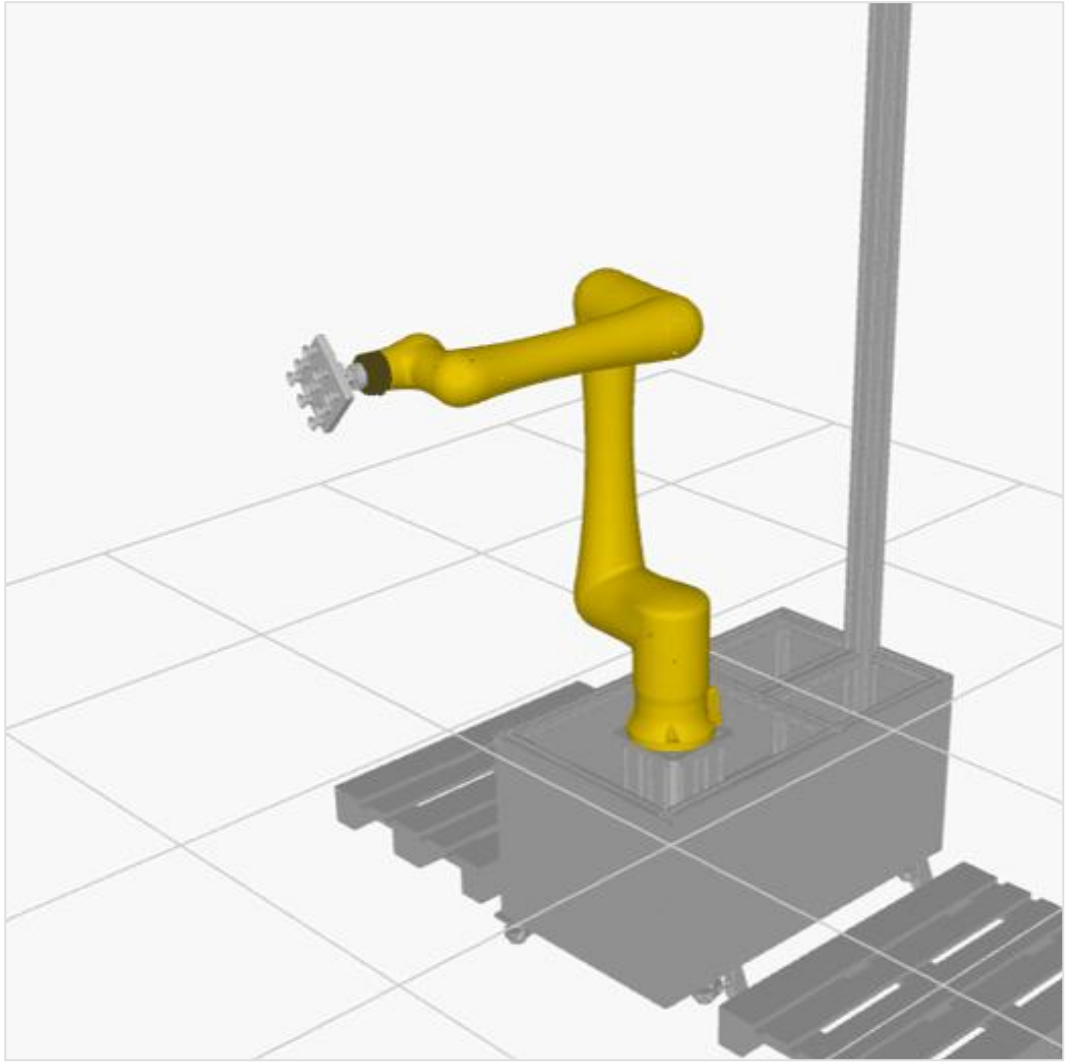
Scene

- The sensor never sees the whole cell. Everything in the robot's reach must be in the **scene**.
- Add objects: **Upload STL**, **Box**, **Sphere**, **Cylinder** or **Group**. Objects form a tree.
- Types: **Bin** (blue), **Environment** (grey), **Virtual collision object** (red, see-through).
- Use simple, slightly **inflated** bodies. Keep CAD models small.
- Align the scene origin with the robot base. Click **Save scene**.



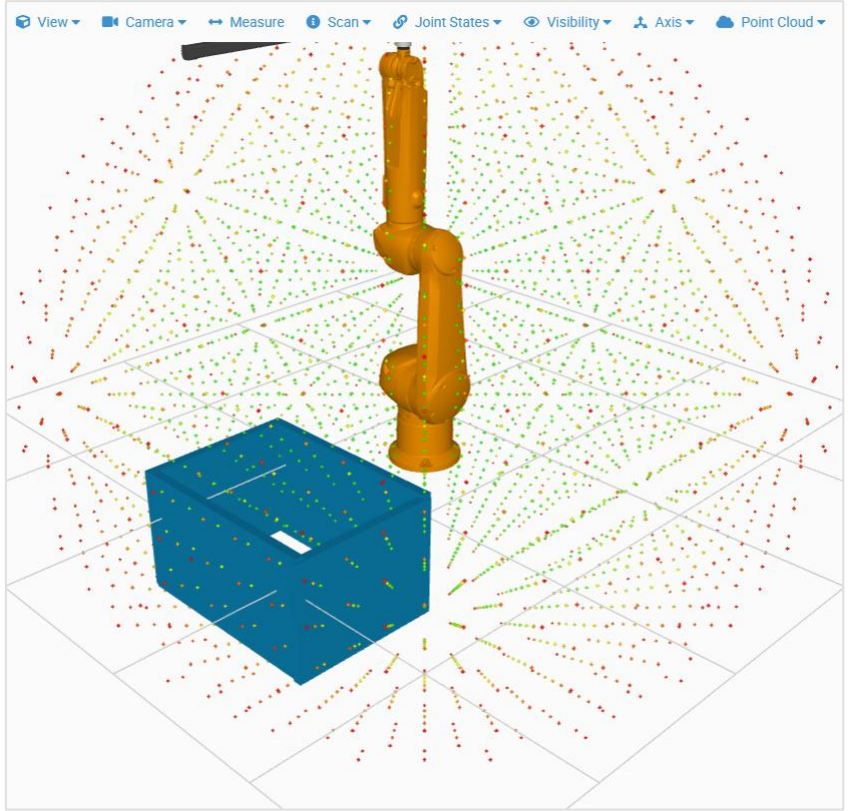
Robot tab

- Move the **virtual** robot; the real robot does not move.
- **Motion mode**: Joint, Linear (tool), Linear (robot base), or **Robot module** to show the real robot.
- **Highlight collisions** marks colliding parts.
- Messages: **Unable to compute inverse kinematics**, **Robot is in collision**.
- Works only with a saved scene, a selected robot and a gripper with a tool point.



Joint limits and working space

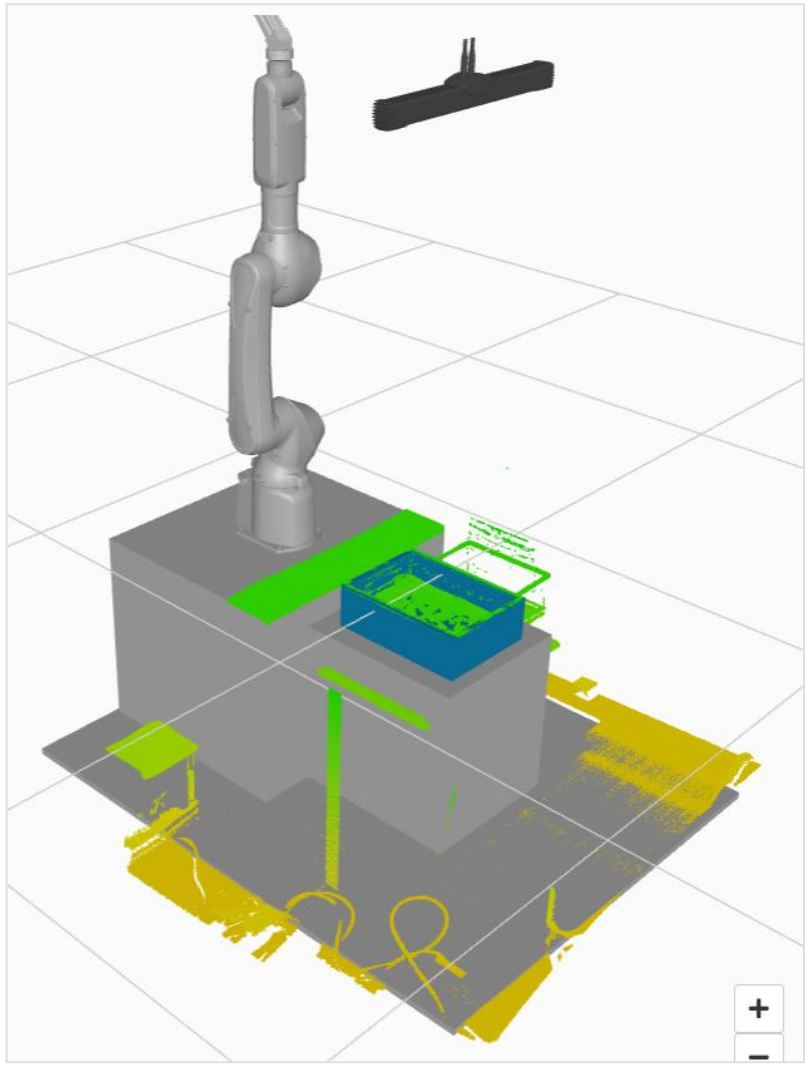




- Joint limits make path planning **faster and more successful**. Set them.
- A limit can only narrow the manufacturer's range. Empty = manufacturer's limit.
- Too tight limits make planning fail. Check that the robot reaches every part of the bin.
- **Generate** shows the working space: green = reachable from almost all directions, red = only a few.
- Aim for the **greener half** in and above the bin. (Robots with Rapid IK only.)

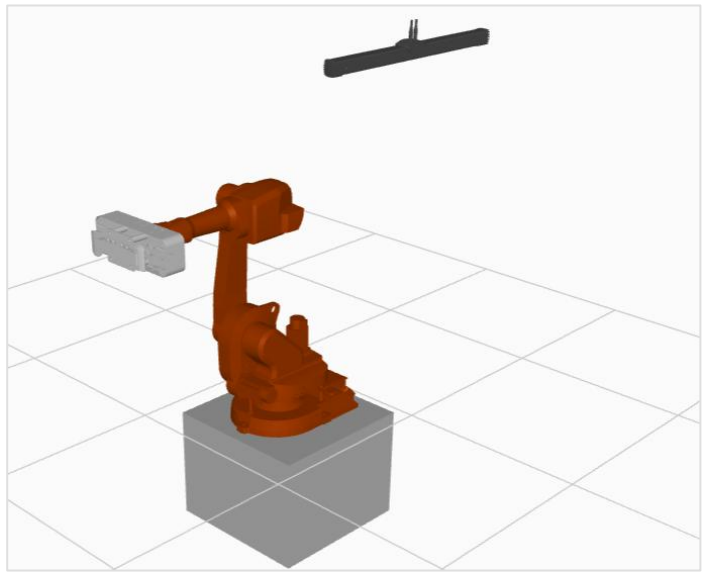
Vision tab

- Lists the vision systems: **Calibrated**, **Uncalibrated** or **Simulated calibration**.
- **Trigger** a scan and move the collision objects until they match the point cloud.
- **Edit** places the sensor by hand (simulated calibration): good for tuning, not for production.
- Only one sensor can be connected at a time.

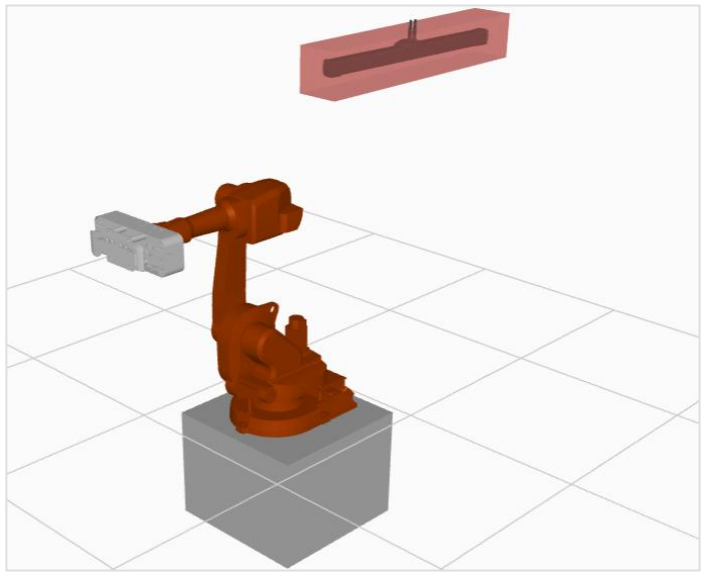


Scene states

When the cell changes during picking, for example a cover opens or a second bin moves in.



Scene state 1



Scene state 2

- An object assigned to a state is in the collision model only while that state is active.
- Up to **four** states. One is **Initial**: active after deployment.
- The robot switches states with **Change scene state**.
- The **Active Scene State** selector on the Robot tab tests a state.

08

Settings

IN THIS CHAPTER

How settings are organized

Collision checking

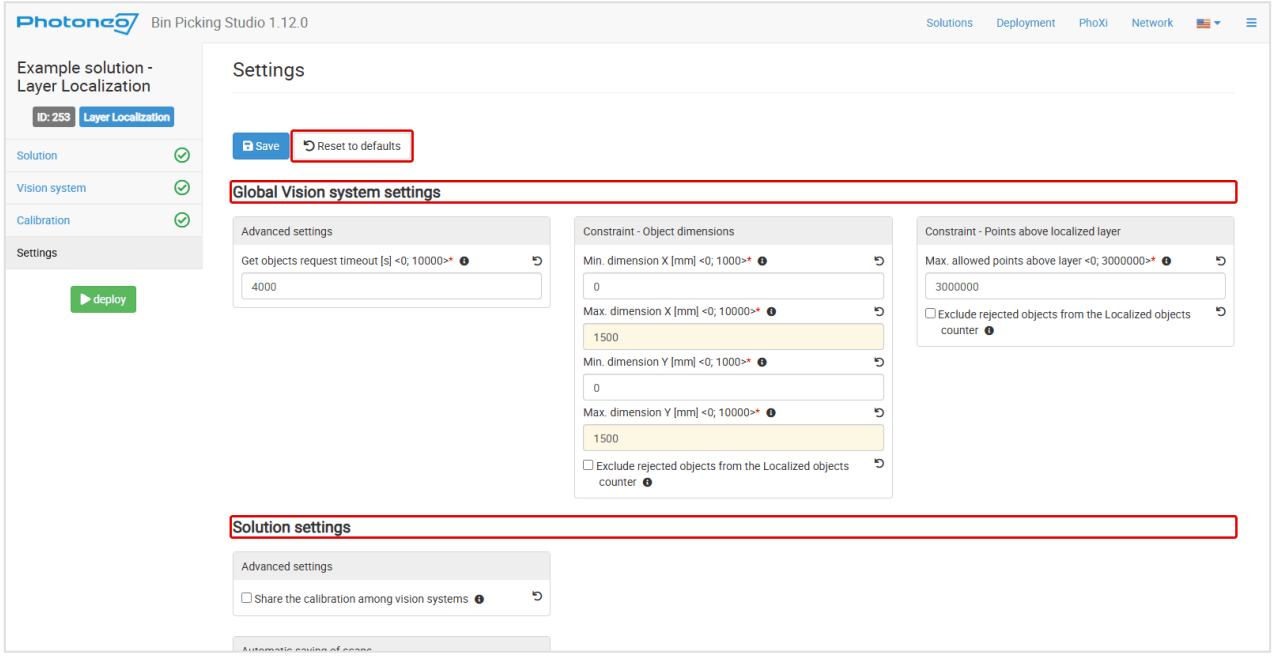
Path planning

Picking priority and gripper angle

Fast scan and saving scans

How settings are organized

- **Global Vision system settings** apply to all vision systems unless overridden.
- **Solution settings** apply to the whole solution.
- A vision system can **override a whole section** on its Settings tab. The section on the Settings page then shows the vision system ID.
- Changed values are **yellow**; the reset button restores the default. The info icon shows the help text.
- Which sections you see depends on the solution type.



Collision checking (BPS)

Setting	What it does	Default
Gripper collision checking type	Volumetric: allows a share of the gripper volume in collision with the scan / objects. Model: no contact at all	Volumetric
Gripper collision sensitivity [%]	Share of the gripper volume allowed in collision. Too low: many rejections. Too high: collisions	5.0
Collision checking with the picked object attached	The picked object is attached to the gripper from the Deapproach stage on	On
Pick only unoccluded objects	Pick only objects the camera sees fully	Off
Resolution of the point cloud volumetric model [m]	Voxel size of the scene model	0.01
Minimum required size of a volumetric model segment	Filters small segments, for example reflections	20

Tip: With the picked object attached, the distance from Grasp to Deapproach must be **larger** than the biggest dimension of the object.

Path planning type

	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	Not planned
Joint trajectories	Checked for collision	Not checked	Not planned
Computation time	Longest	Shorter	Shortest
Use	Default, most cases	Carefully	Carefully

- **Path planning off**: waypoints are sent without trajectories and **Plan linear trajectory** does not work; the robot moves in joint space.
- Also in this section: **Path planning timeout [s]** (3.0), **Path planning attempts** (3), **Linear trajectory sampling step [m]** (0.01) and **Prefer linear trajectory** (Off).
- All can be overridden per vision system.

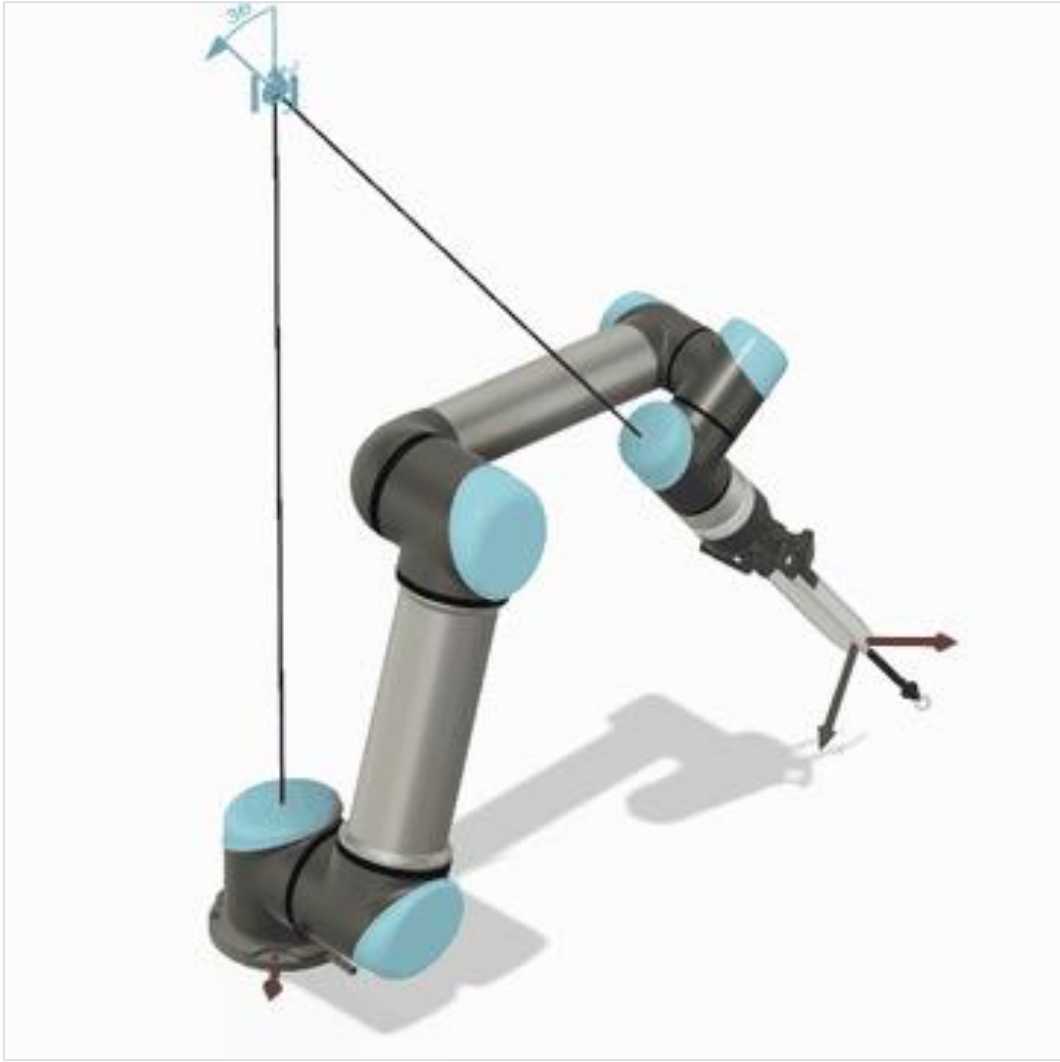
Picking priority and gripper angle

Picking priority

- Decides the order in which objects go through path planning.
- Methods: closest point or centroid in a direction (default **0, 0, −1**: top first), localization score, number of collisions with the gripper, **Gripping Point Priority**.

Gripper angle

- **Angular threshold** (default **50°**) between the tool's Z-axis and the **reference vector** (default **0, 0, 1**).
- Grasps above the threshold are rejected: status *Grasping angle rejected*.



Fast scan and saving scans

Fast scan (CAD Picking, CAD Localization)

- Captures only the texture. Object poses come from memory; only their presence is checked.
- Full scans are still needed regularly: **Full scan frequency** (default 3).
- Start with one fast scan per full scan, then try up to five.

Automatic saving of scans

- Saves the last set of scans when something goes wrong, for support and analysis.
- On internal issues: **On** by default.
- On path planning failed, no object found, pick failed: **Off** by default. Turn them on **before** the problem happens.
- Folder size limit: 3 GB; older scans are deleted.

09

Deployment and troubleshooting

IN THIS CHAPTER

Deploying a solution

Diagnostics, toolbar, console and cycle timer

Object status

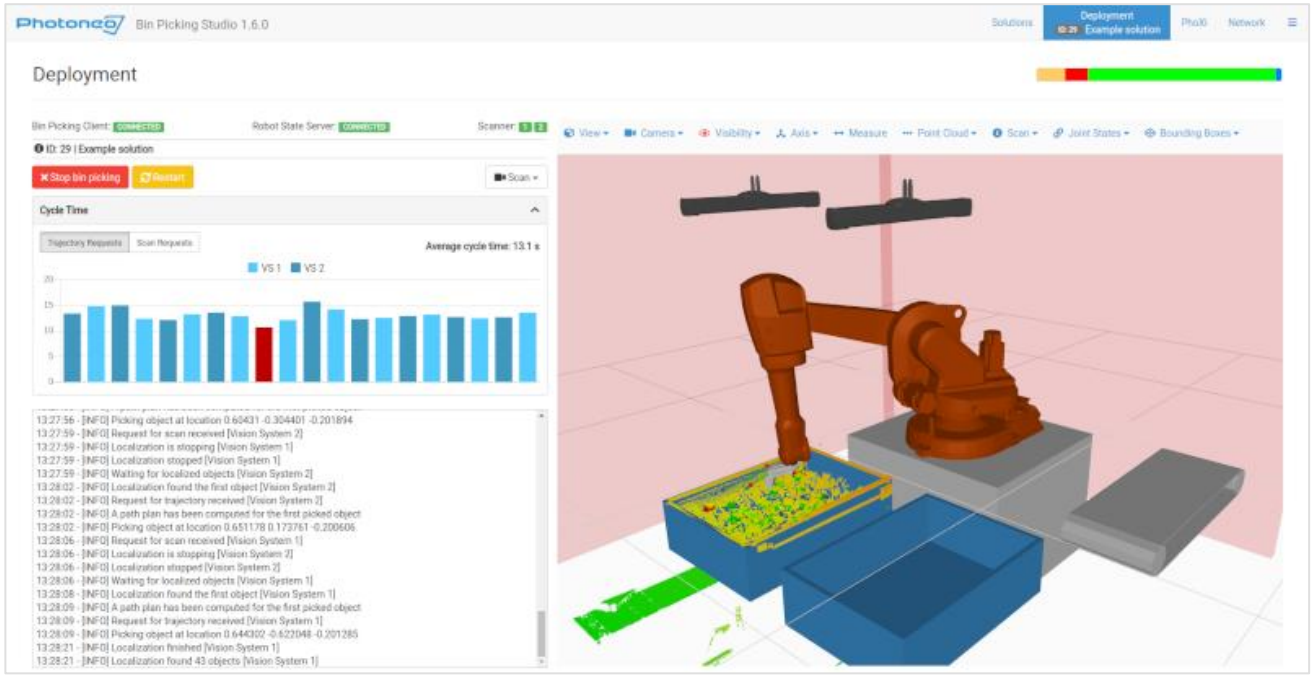
The Inspector

Fixing path planning failures

Asking Photoneo support

Deploying a solution

- Deploy from the **ready** flag, from the **Deployment shortcut** in the solution, or on the **Deployment** page.
- Choose the mode:
  - **Production**: the robot program sends the requests.
  - **Simulation**: you send the requests from the toolbar; the real robot is not needed. Hand-eye vision systems still need the real robot.
- Buttons to **stop** and **restart** the deployment.



Diagnostics, toolbar and console

Hardware diagnostics

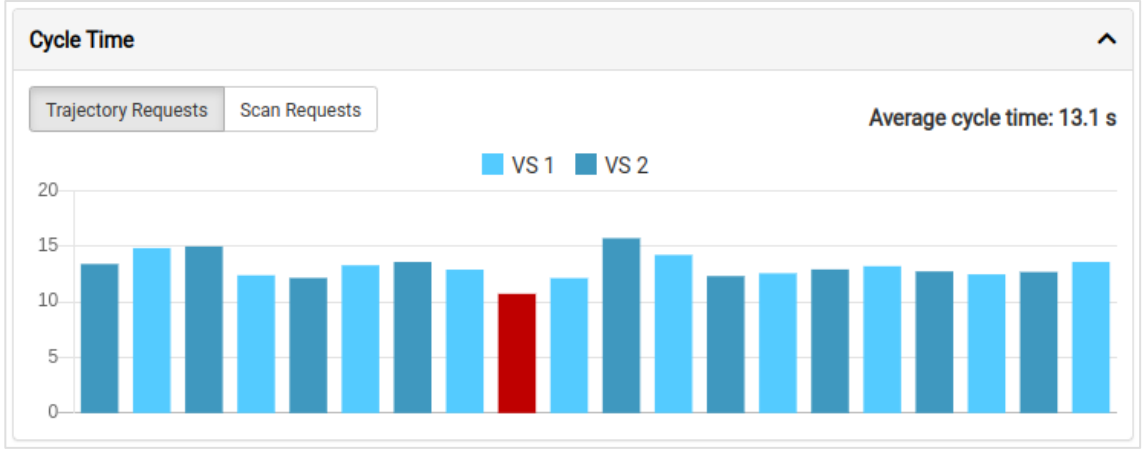
- **Action Request Client**, **Robot State Server** and one **sensor indicator** per vision system.
- Green **CONNECTED**, orange **WARNING**, red **DISCONNECTED**.

Action request toolbar

- Production: **Scan** only.
- Simulation: **Initialization**, **Change scene state**, **Change localization bounding box**, **Scan**, **Trajectory**.

Console and cycle timer

- Log levels **INFO**, **WARN**, **ERROR**, **FATAL**.
- Cycle time = time between two Scan or Trajectory requests of one vision system. Last 20 cycles; failed requests in red.



Object status (BPS)

The color of each localized object shows how far it got in path planning.

Pipeline stages, in order


New object


Object from the previous run


Path plan from the previous run is not valid


Path plan from the previous run is valid


All waypoints computed


Trajectory planned


Ready to pick


Object being picked

Failure statuses, in order


No gripping points defined


Object occluded from scanner position


Grasping angle rejected

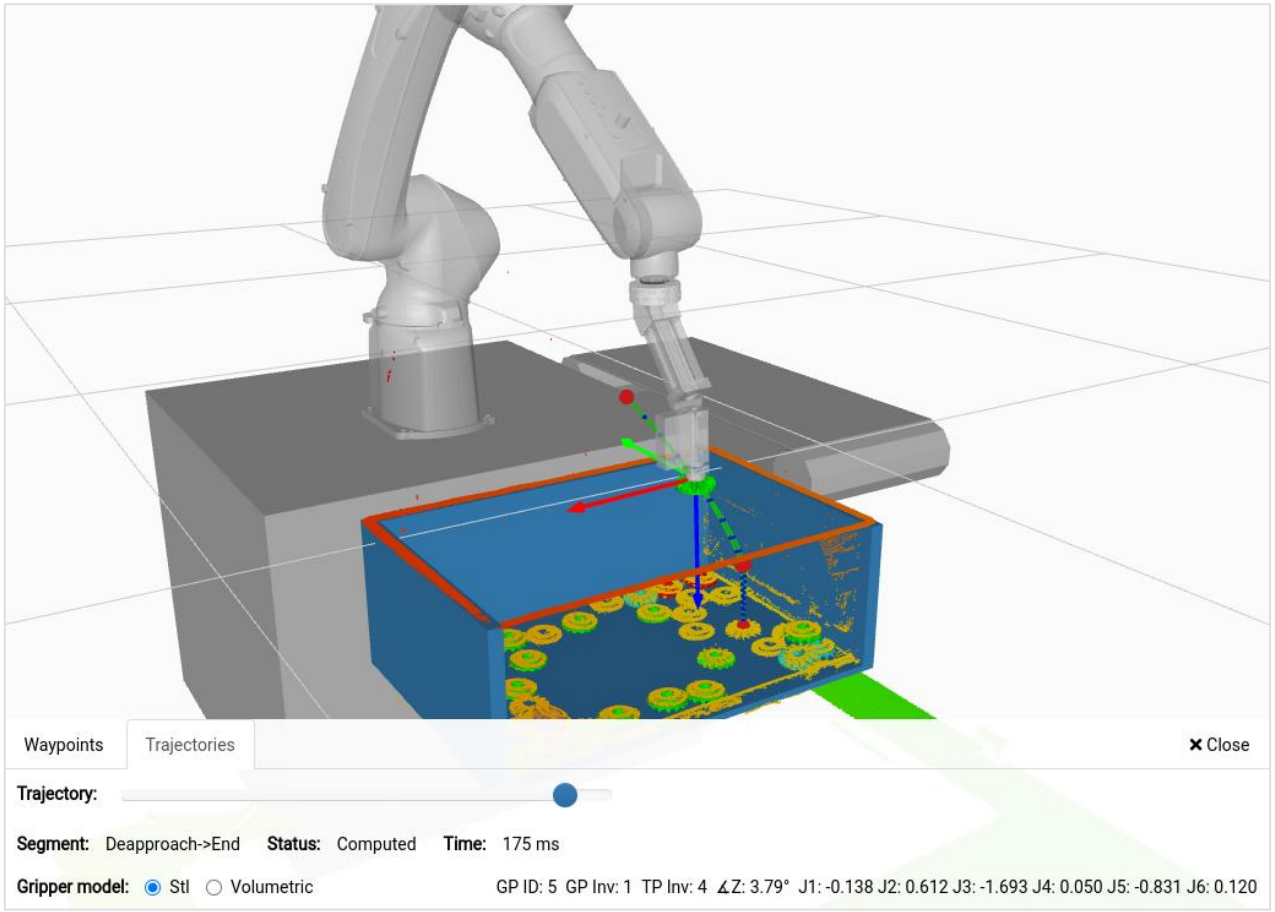

Gripper in collision in grasp pose


Waypoint computation failed


Trajectory planning failed

The Inspector

- Click a localized object, then **Show Inspector**.
- Always shown: **GP ID**, **GP Inv**, **TP Inv** and **∡Z** (angle to the gripper reference vector).
- Tabs, depending on the status:
  - **Grasp invariances**: the grasp pose, IK result, gripper collision
  - **Waypoints**: each waypoint, its status and IK result; the colliding pair is shown in red
  - **Trajectories**: plays the planned motion; a failed segment is a black line

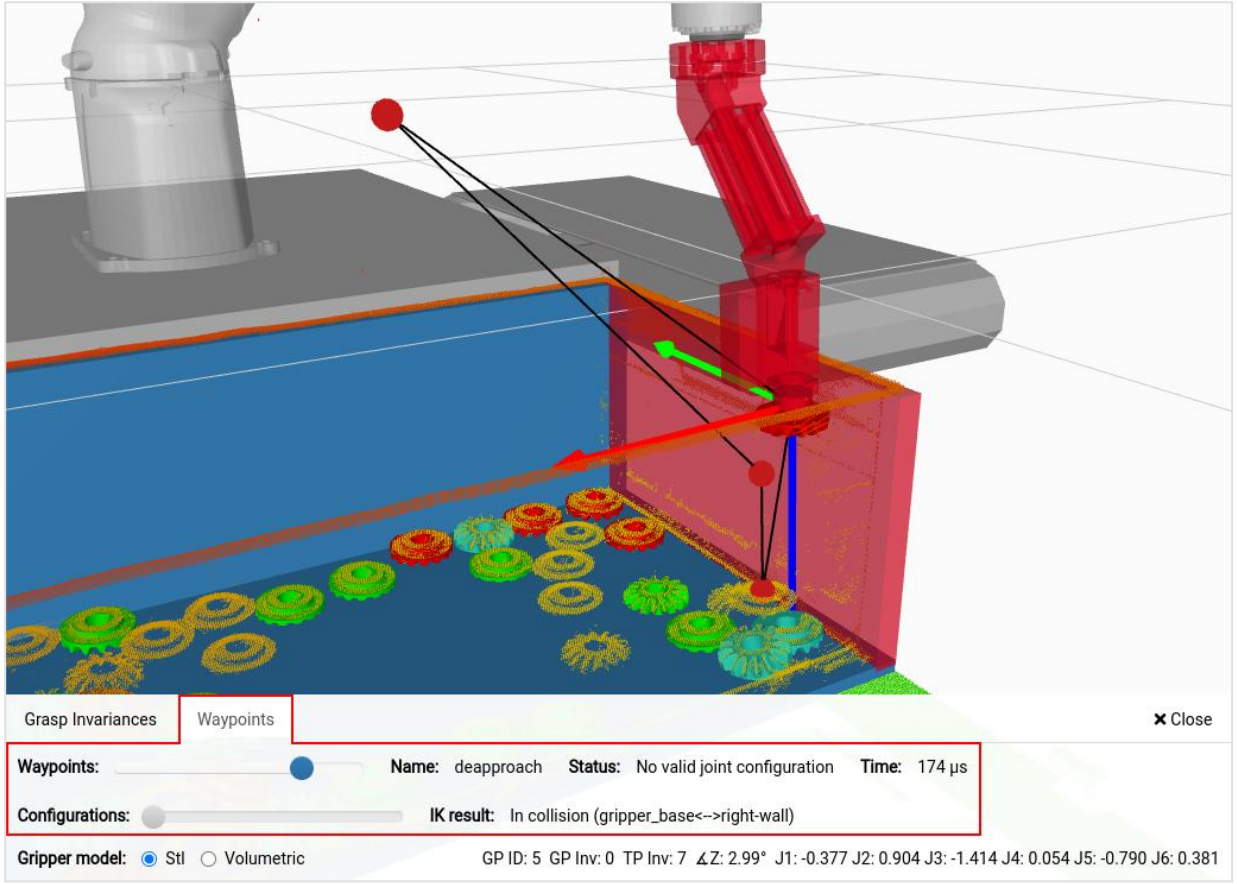


Fixing path planning failures

Status	First things to check
Grasping angle rejected	Tool point and gripping points; Angular threshold; reference vector for wall / ceiling robots
Gripper in collision in grasp pose	Often a real collision: picked later. Otherwise Gripper collision sensitivity, gripping points, scan quality
Waypoint: Unreachable	Joint limits, a reachable gripping point, bin position, IK solver timeout
Waypoint: In collision	The colliding pair in the Inspector; gripping points and grasping method; scan quality
Cartesian threshold failed	A joint turns more than 0.3 rad between points: set joint limits, lower the sampling step
Linear / joint trajectory is too long	More than 100 points: bring the two waypoints closer, raise the sampling step
End position mismatch	Adjust joint limits so the End waypoint has only the End pose configuration

Common collision situations

- **Scan ↔ gripper in Approach**: Approach is too close to the object; reflections above it collide. Move Approach further away.
- **Picked object ↔ scan in Deapproach**: the Grasp-to-Deapproach distance is smaller than the object. Make it larger.
- **Picked object ↔ bin wall in Deapproach**: lean away from the walls. Use Start / End as the origin, with Start / End above the middle of the bin.
- **Floating points** from reflections: raise **Minimum required size of a volumetric model segment**.



Troubleshooting overview

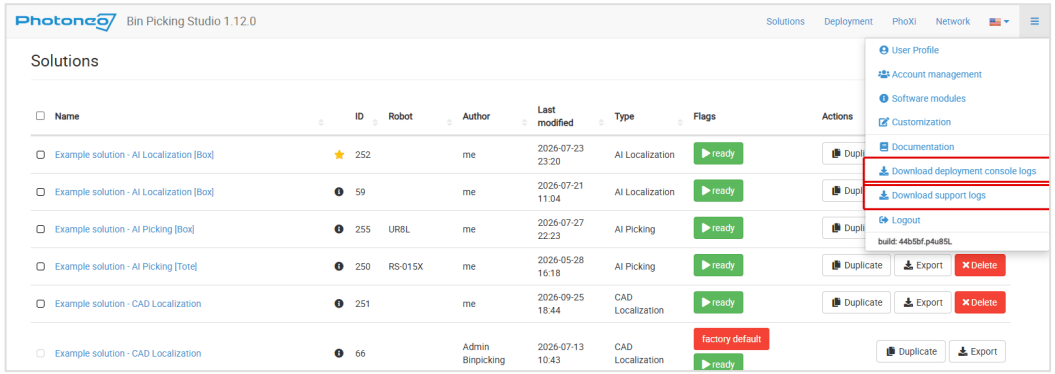
Problem	Where to look
The robot can't connect	Hardware diagnostics; Test connection on the Network page; error codes 3 and 5
The sensor isn't found	Sensor indicator; IP addresses of interface and sensor; PhoXi page
Calibration is inaccurate or fails	Diverse poses, base frame offset, tool, Robot State Server, custom calibration profile
No object found or objects rejected	Error codes 202 / 502–505; bounding box; constraints; scanning profile
Path planning fails or the gripper collides	Object status and Inspector; collision situations; save the scans

Asking Photoneo support

Send your request to **support@photoneo.com**. Describe the problem and attach:

1. **Support logs**: menu > **Download support logs**.
2. **Deployment console logs**: menu > **Download deployment console logs**.
3. **Saved scans** of the situation. Turn on **Automatic saving of scans** before the problem happens.
4. The **solution export** (.pbcf).

After a fatal error, the **An error occurred** dialog offers the support logs and a link to the Photoneo Help Center.



10

Locator Studio specifics

IN THIS CHAPTER

LS solutions

Reported pose and origin offset

Pose constraints

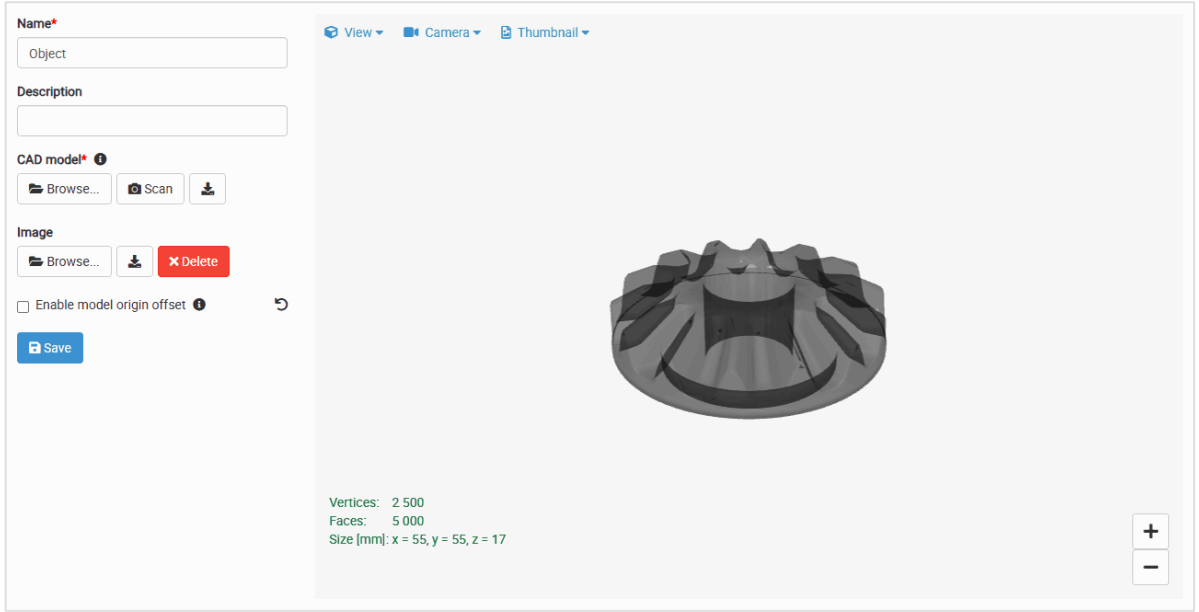
Locator Studio solutions

Solution type	Setup steps
CAD Localization	Picked object, Vision system, Calibration, Settings
AI Localization	Vision system, Calibration, Settings
Layer Localization	Vision system, Calibration, Settings

- No Robot, Gripper, Grasping method or Environment: the **robot program plans the motion**.
- The robot sends **Scan**, then gets the poses with **Get objects**. A timeout applies (**Get objects request timeout [s]**).
- Robot side: the Robot module, or your own Action Request Client on any compatible controller or PLC. The Robot State channel is optional.
- Calibration: extrinsic, hand-eye, or **marker space** (one scan; then define a user frame at the marker origin).
- Hand-eye LS: use the same tool as in calibration and send its TCP in the Scan request.

Reported pose

- By default LS reports the **origin of the object model**.
- **Enable model origin offset** to report another point: type translation and rotation, drag the marker, or click **Automatic** (center of the geometry).
- Example: origin in the middle of the top face with Z pointing down. The tool point can then move straight to the reported pose.
- **Constraint – Angle** rejects objects whose Z-axis is more than **25°** (default) from the vector **0, 0, −1**.



Where to find more

Documentation (VGR 1.12)

- Getting Started
- Studio Interface
- Common Setup (BPS & LS)
- BPS Solution Setup
- LS Solution Setup
- Deployment
- Robot Module: integration guides, action requests, error codes

Support

- **support@photoneo.com**
- Photoneo Help Center: www.photoneo.com/Help-Center

In the Studio

- The info icon next to each field shows its help text.
- Menu > **Documentation**

Thank you

...and happy picking!

support@photoneo.com