Dobot VX500 Smart Camera User Guide

Issue: V1.3
Date: 2024-05-16
Shenzhen Yuejiang Technology Co., Ltd. | China

Disclaimer

To the maximum extent permitted by applicable law, the products described (including its hardware, software and firmware, etc.) in this document are provided **AS IS**, which may have flaws, errors or faults. Dobot makes no warranties of any kind, express or implied, including but not limited to, merchantability, satisfaction of quality, fitness for a particular purpose and non-infringement of third party rights. In no event will Dobot be liable for any special, incidental, consequential or indirect damages resulting from the use of our products and documents.

Before using our product, please thoroughly read and understand the contents of this document and related technical documents that are published online, to ensure that the robot arm is used on the premise of fully understanding the robot arm and related knowledge. Please use this document with technical guidance from professionals. Even if follow this document or any other related instructions, damages or losses may happen in the using process. Dobot shall not be considered as a guarantee regarding all security information contained in this document.

The user has the responsibility to make sure following the relevant practical laws and regulations of the country, in order that there is no significant danger in the use of the robot arm.

Shenzhen Yuejiang Technology Co., Ltd.

Address: Room 1003, Building 2, Chongwen Garden, Nanshan iPark, Liuxian Blvd, Nanshan District, Shenzhen, Guangdong Province, China

Website: www.dobot-robots.com

Preface

Purpose

This document introduces the installation and usage of Dobot VX500 smart camera, which is convenient for users to understand and use the smart camera.

Intended audience

This document is intended for:

- Customer
- Sales Engineer
- Installation and Commissioning Engineer
- Technical Support Engineer

Supporting version

Product	Version
VX500 plugin	V1.1.x
CRA controller firmware	V4.5.0.4 and above
CRA end firmware	V6.3.1.1 and above
DobotStudio Pro	V4.5.0.0 and above
VX500 SmartCamera firmware	V3.1.0 240221 and above
DobotCalibrate program	V1.0.6 and above

NOTICE

The schemes created by the older version of VX500 plugin are not available in the current version. You need to create new schemes.

Revision history

Date	Version	Revised content
2024/05/16	V1.3	• Added a note to prohibit hot plugging and unplugging of the camera • Added instructions for binding 2.5D positioning code sizes to schemes
2024/03/29	V1.2	Based on the plugin V1.1.1
2023/10/18	V1.1	Details optimization, based on the plugin V1.01
2023/07/03	V1.0	The first release

Symbol conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
⚠ DANGER	Indicates a hazard with a high level of risk which, if not avoided, could result in death or serious injury.
⚠ WARNING	Indicates a hazard with a medium level or low level of risk which, if not avoided, could result in minor or moderate injury, robot arm damage.

⚠ NOTICE	Indicates a potentially hazardous situation which, if not avoided, could result in robot arm damage, data loss, or unanticipated result.
ℹ NOTE	Provides additional information to emphasize or supplement important points in the main text.

Contents

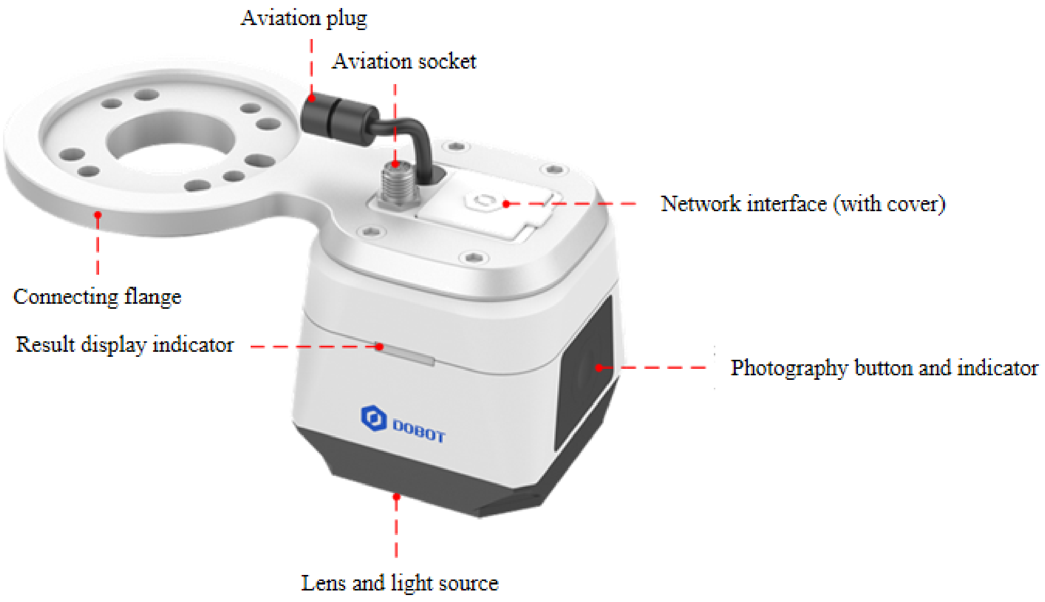
1. Product Introduction

Dobot VX500 smart camera adapts to Dobot CRA series robots (except CR20A), which is plug-and-play. With an embedded high-performance vision system, the smart camera helps the robot identify directions, position accurately and execute vision schemes to realize Dobot universal vision solutions.

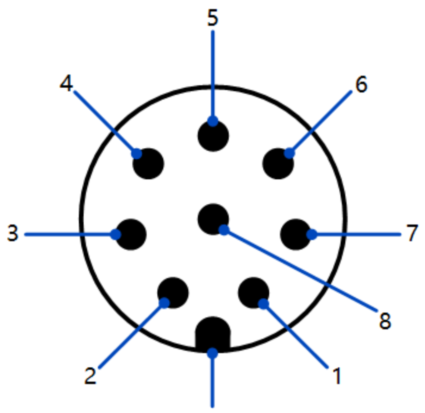
It is equipped with DobotStudio Pro for field deployment, which makes debug requirements even lower and brings a more comprehensive and cost-effective option for vision detection. Dobot VX500 allows to:

1) Use the DobotStudioPro to simultaneously define vision solutions and accomplish vision tasks with the robot;
2) Realize 2.5D positioning, 2D positioning, measurement and recognition;
3) Customize calibration files for different visual scenes;
4) Customize camera parameters.

The VX500 smart camera is shown below.



- **Aviation plug**: Connect to the aviation socket of the robot arm for RS485 communication between the camera and the robot. The pins are defined as follows.



Pin	Definition	Pin	Definition
1	485A	5	24V output

Pin	Definition	Pin	Definition
2	485B	6	Digital output 2
3	Digital input 2	7	Digital output 1
4	Digital input 1	8	GND

- **Aviation socket**: When the end tool needs to communicate with the robot arm via an aviation plug, you can connect the aviation plug to this socket for communicating with the robot arm. The socket type is the same as the aviation socket on the end of CRA series robots.

  NOTE

  The camera communicates with the robot arm via RS485, so tools connected to this socket are recommended to communicate with the robot arm via digital input/output. If the tool connected to this socket also uses RS485 for communication, data conflicts may occur.

- **Network interface**: Ethernet interface with protective silicone cover. The camera needs to be connected to the computer via a network cable during the camera configuration, and can be used offline without a network connection when running a project.

  NOTE

  For the connection between the robot arm and the computer, you need to use another network cable or WiFi, independent of this network interface.

- **Photography button and indicator**: When setting **Photography trigger mode** to **Camera button**, you can press the circular button to trigger photography. There are three indicators around the button, namely power indicator (PWR), network indicator (LNK) and status indicator (STS). See the corresponding silkscreen on the camera for details.
- **Lens and light source**: Camera lens and 14 LED light sources.
- **Result display indicator**: Display the running result (OK/NG).
- **Connecting flange**: For installing the camera to the end of the robot.

2. Software/Hardware Installation and Connection

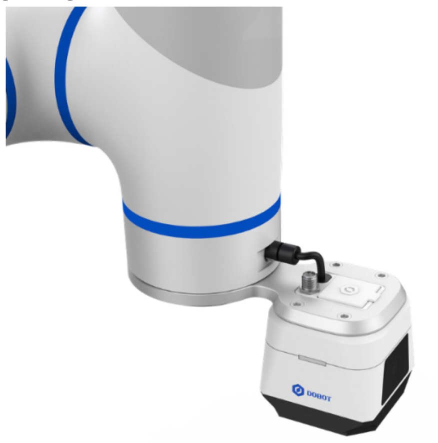
The VX500 smart camera is installed in the eye-in-hand mode, that is, the camera is installed at the end of the robot. The smart camera is assembled with the connecting flange before delivery, so you just need to install the camera on the end of the robot. The design of the connecting flange to the side of the robot arm conforms to ISO 9409-1, which is suitable for CRA series (except CR20A).

2.1 Hardware installation and connection

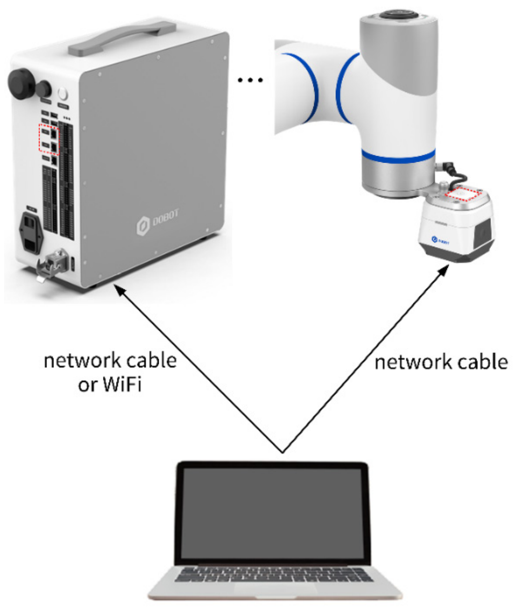
1. Connect the aviation plug of the VX500 smart camera to the aviation socket on the end of the robot arm.

> **NOTICE**
>
> It is prohibited to plug and unplug the camera with electricity, before plugging and unplugging the aviation plug, please turn off the power of the controller and make sure that the indicator lights of the robot and the camera are all off.

2. Install the connecting flange to the end of the robot arm using four M6 screws.

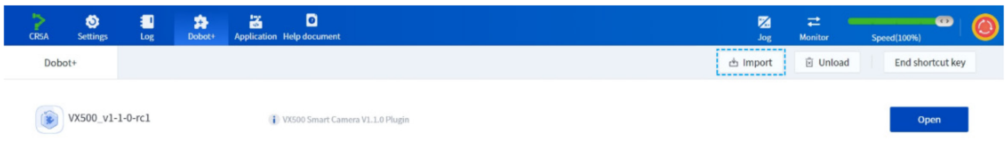


3. Connect the VX500 smart camera to the computer using a network cable. The network cable is used to debug the camera and can be unplugged when you run the project.
4. Connect the robot controller to the computer using network cable or WiFi. The network cable or WiFI is used for the computer to control the robot arm through DobotStudio Pro.



2.2 Software installation and connection

1. Download the SmartCamera plugin package from Dobot website or contact Dobot technical support engineer to obtain it, including the SmartCamera plugin (VX500_v1-1-1), calibration program installation package (DobotCalibrateSetup.exe), and the upgrade firmware package.
2. Double-click **DobotCalibrateSetup.exe** to install the calibration program. The installation requires administrator authority and the installation path can be self-defined. Before using the plugin, you need to open the calibration program and let it run in the background.
3. Open DobotStudio Pro and connect to the robot. For specific procedure, see DobotStudio Pro user guide.
4. Open the Dobot+ page, **import** and **install** the VX500_v1-1-1 plugin.



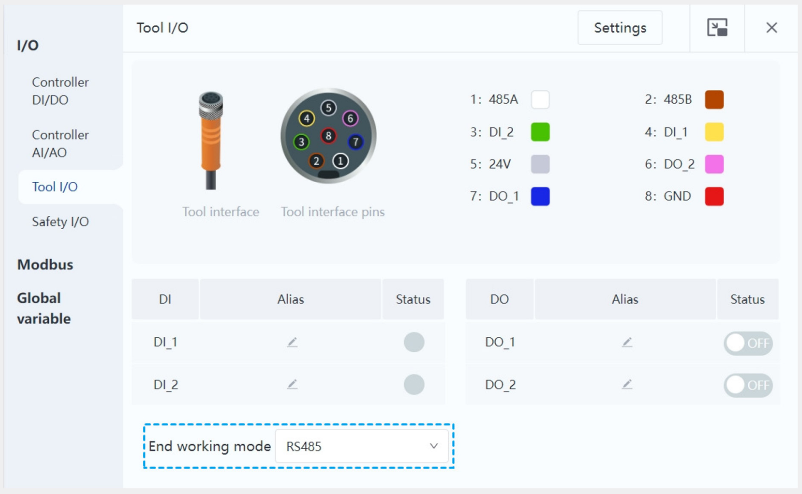
NOTE

Only Chinese and English are available for the current version of the VX500 plugin.

5. Open the VX500 plugin, and wait for the plugin to search for the smart camera connected to the computer via the network cable.

NOTE

- **Please turn off the firewall on your computer before connecting to the camera, otherwise it may cause the communication stuck between the plugin and the camera.**
- Make sure that the working mode of the end of the robot arm is RS485 before connecting to the camera, which can be checked and modified in the **Monitor > Tool I/O** page.



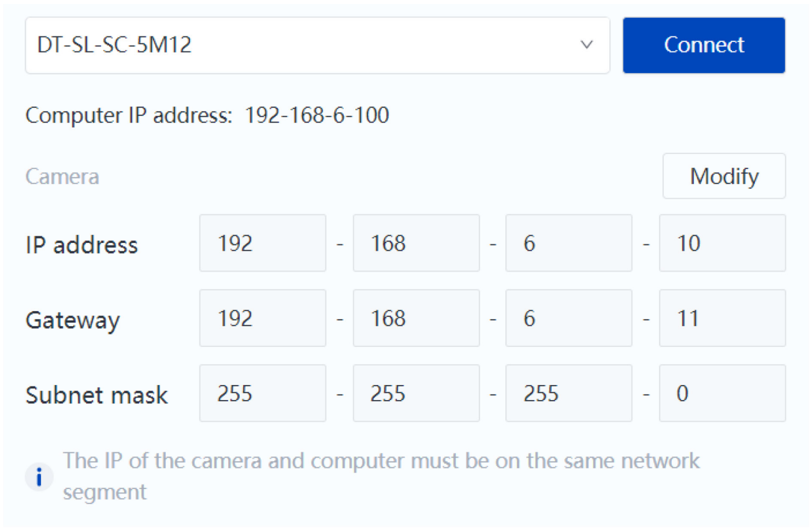
DI	Alias	Status
DI_1		
DI_2		

DO	Alias	Status
DO_1		OFF
DO_2		OFF

End working mode: RS485

6. After the plugin searches for the camera, you will see the IP address of the computer and camera on the page. Modify the IP addresses of the camera and computer, keeping them in

the same network segment.


DT-SL-SC-5M12 | Connect

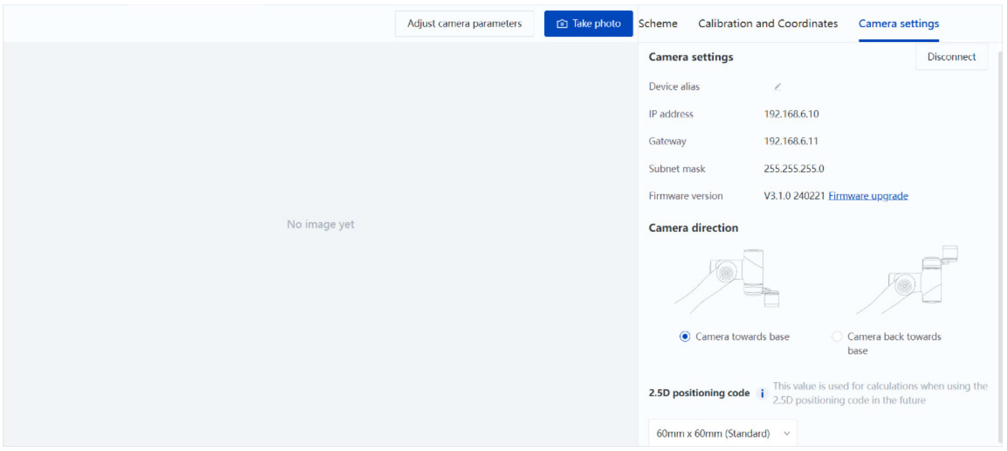
Computer IP address: 192-168-6-100

Camera | Modify

IP address	192	168	6	10
Gateway	192	168	6	11
Subnet mask	255	255	255	0

The IP of the camera and computer must be on the same network segment

7. Click **Connect**, and the plugin starts to connect to the smart camera. After connection, you will enter the Camera settings page.



Camera settings

Device alias	
IP address	192.168.6.10
Gateway	192.168.6.11
Subnet mask	255.255.255.0
Firmware version	V3.1.0 240221 Firmware upgrade

Camera direction

[✓ Camera towards base] Camera back towards base

2.5D positioning code: This value is used for calculations when using the 2.5D positioning code in the future

60mm x 60mm (Standard)

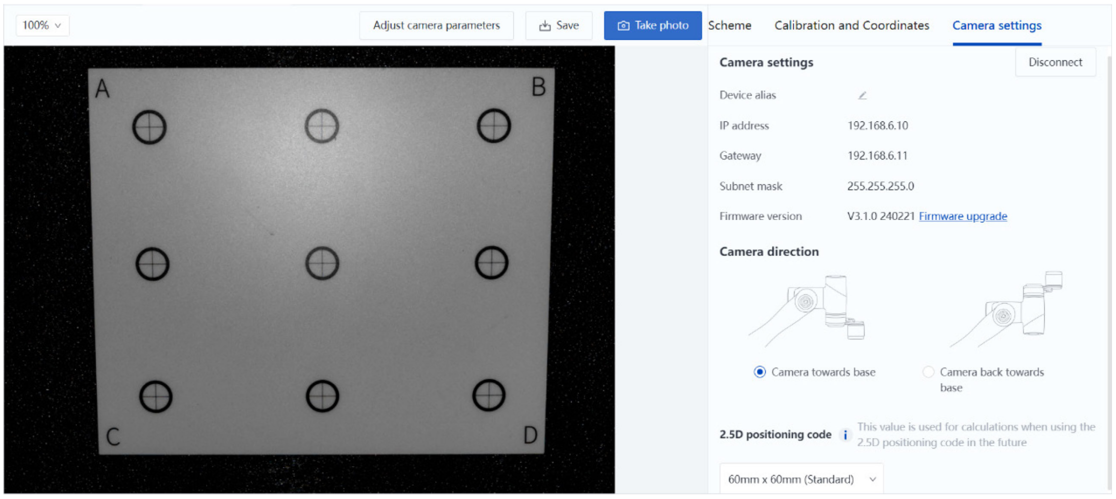
You can click **Disconnect** to disconnect from the current camera and search for the camera again.

8. Check the **Firmware version**. If the firmware version is lower than **V3.1.0 240221**, click **Firmware upgrade**, and select the upgrade file (with a file suffix of dav or bin) in the pop-up window.

NOTICE

- The path to the upgrade file only supports English and the current language of the operating system.
- During the upgrade, do not disconnect the camera or cut off the power.
- If the version of the upgrade target is inconsistent with the first two digits of the current version, the old version of the scheme cannot be used after the upgrade. Please create a new scheme after the upgrade.
- If you want to upgrade from V3.0.1 230625 (the initial version of the first batch of shipped cameras) to V3.1.0 240221, you need to upgrade to V3.0.6 24018 first, and then upgrade from V3.0.6 240118 to V3.1.0 240221.
- If the firmware version is an earlier dated version of V3.1.0 (e.g. V3.1.0 240116), you can upgrade directly to V3.1.0 240221.

3. Software Main Interface



Camera settings

Device alias	
IP address	192.168.6.10
Gateway	192.168.6.11
Subnet mask	255.255.255.0
Firmware version	V3.1.0 240221 Firmware upgrade

2.5D positioning code: This value is used for calculations when using the 2.5D positioning code in the future

60mm x 60mm (Standard)

The left side of the page displays the camera image.

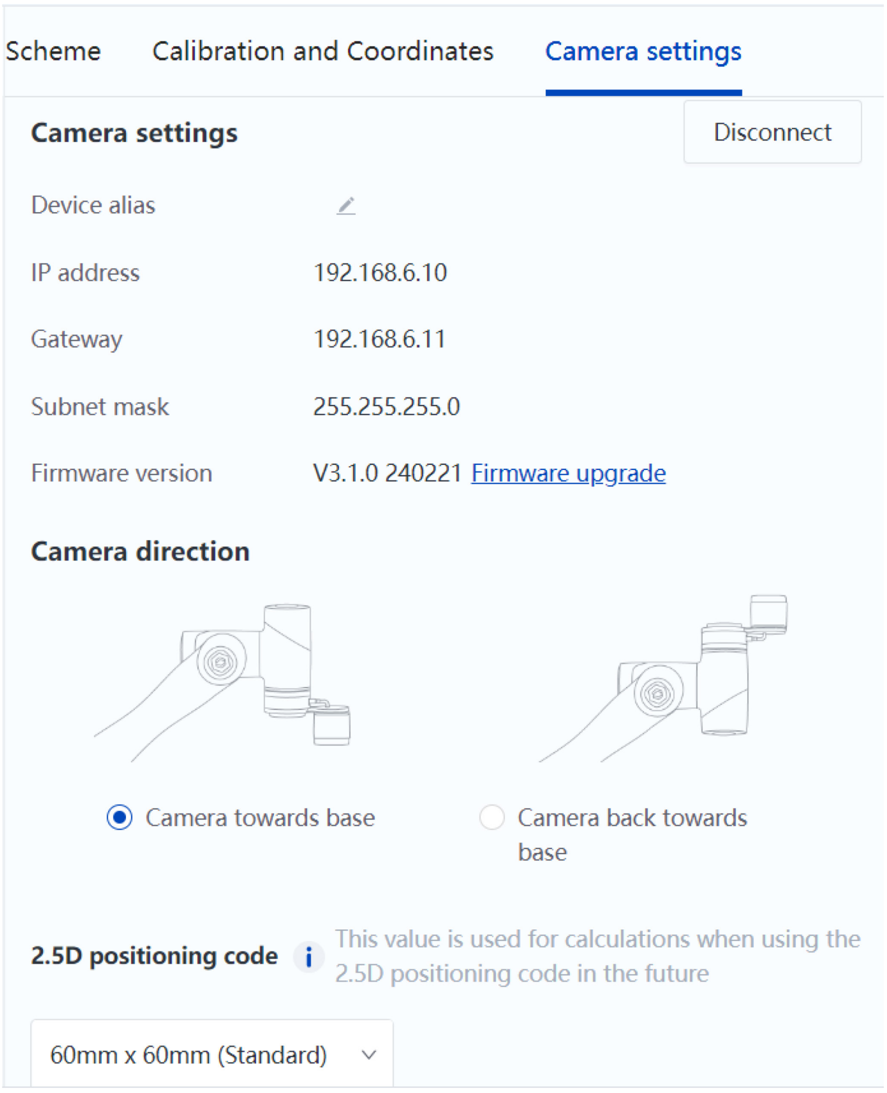
- After connection, the image area displays the last image taken by the camera.
- You can click **Take photo** on the top right corner of the image to control the camera to take photos (you need to set the **Photography trigger mode** to **Manual photography**, see Camera parameters for details).
- When there is an image in the image area, the **Save** button is displayed, you can click it to save the current image to the local.
- You can click **Adjust camera parameters** to open the **Camera parameters** page, see Camera parameters for details.
- You can adjust the image size by selecting the display scale from the drop-down list on the top left corner of the page.

The right side of the page displays the camera settings and related functions.

- You can modify the basic settings of the camera on the Camera settings page, see Camera Settings and Parameters for details.
- You can manage the calibration files and coordinate systems on the **Calibration and Coordinates** page, see Calibration and Coordinates for details.
- You can manage the running schemes of the camera on the **Scheme** page, see Scheme for details.

4. Camera Settings and Parameters

4.1 Camera settings



Camera settings

Device alias

IP address 192.168.6.10

Gateway 192.168.6.11

Subnet mask 255.255.255.0

Firmware version V3.1.0 240221 Firmware upgrade

Camera direction

[✓ Camera towards base] Camera back towards base

2.5D positioning code This value is used for calculations when using the 2.5D positioning code in the future

60mm x 60mm (Standard)

4.1.1 Camera settings

- **Device alias** is editable. You can set aliases to distinguish different camera devices.
- **IP/Gateway/Subnet mask** shows the current network configuration of the camera.
- **Firmware version** shows the current version of the camera (can be upgraded), see the last step in Software Installation and Connection for details.

4.1.2 Camera direction

Select according to the actual direction of the camera.

- Camera lens towards base
- Camera lens back towards base

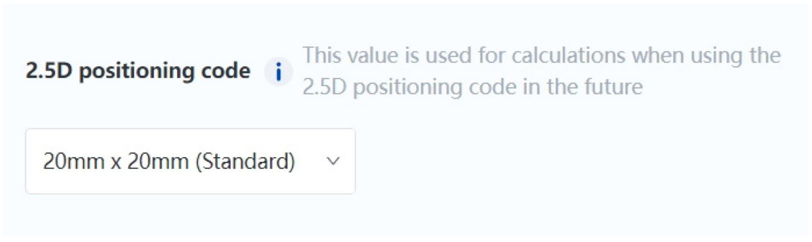
NOTE

- To use the 2.5D positioning function, you need to keep the camera in the selected direction during calibration, so that there are no restrictions when using the positioning function.
- To use the 2D positioning function, you need to keep the camera in the selected direction during calibration and positioning.

4.1.3 2.5D positioning code

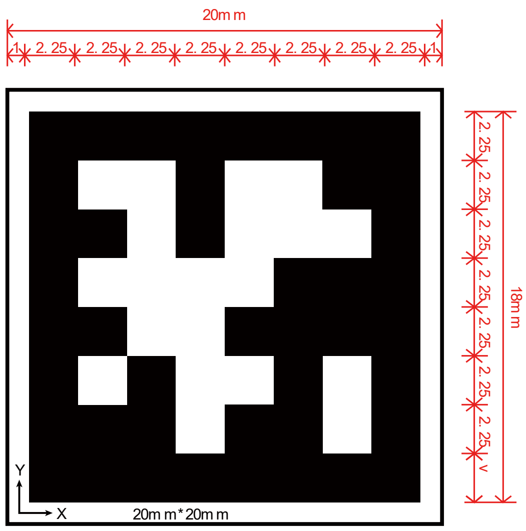
Select the actual size of the 2.5D positioning code (select based on what you actually use).

- The standard product is a VX500 shipping accessory, and the size of the standard product is the overall size including the outer frame of the positioning code.

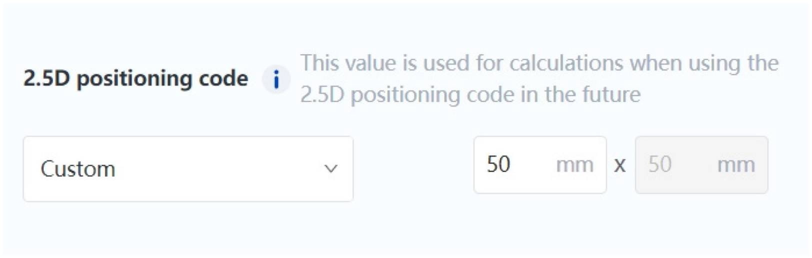


2.5D positioning code — This value is used for calculations when using the 2.5D positioning code in the future

20mm x 20mm (Standard)



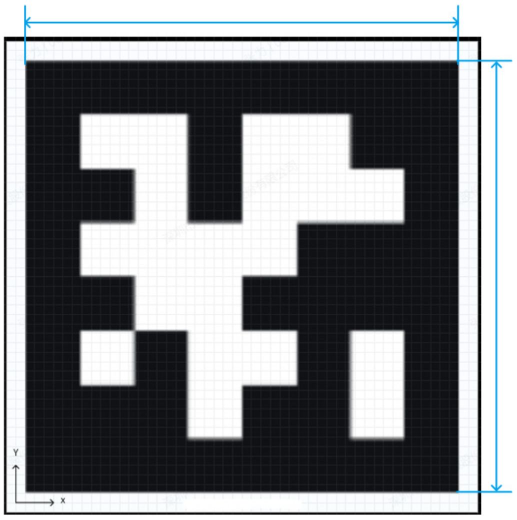
- If you need to use a customized 2.5D positioning code, select **Custom** and enter the size (range: 10mm – 300mm).



2.5D positioning code — This value is used for calculations when using the 2.5D positioning code in the future

Custom — 50 mm x 50 mm

The size of the customized positioning code is the size of the black area inside the positioning code, as shown in the figure below.



NOTE

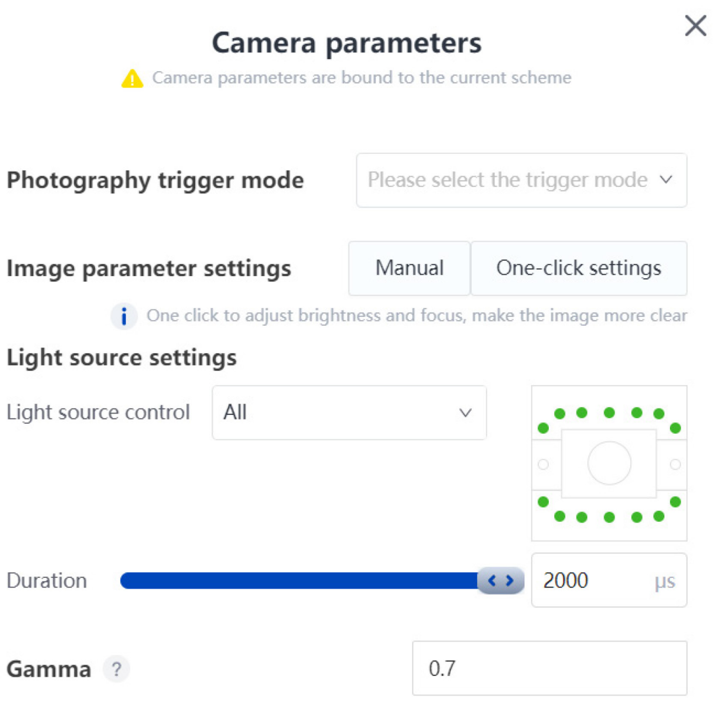
- If you want to print the 2.5D positioning code by yourself, you can contact Dobot technical support engineer to obtain the drawing.
- The size of the positioning code is set globally, and different sizes of positioning codes cannot be mixed.
- There are multiple styles of positioning codes of the same size, corresponding to different serial numbers (see 2.5D coordinate system for details). Select it according to your actual needs.
- 2.5D positioning code size setting and scheme binding:
    - Binds the currently set positioning code size when a new scheme is created.
    - When switching schemes, the positioning code size is also automatically switched to the scheme-bound size.
    - After modifying the positioning code size, edit and save the current scheme to make the new size effective.

4.2 Camera parameters

You can click **Adjust camera parameters** above the image to open the **Camera parameters** page.

NOTE

Camera parameters are bound to the scheme. Any modifications to camera parameters will only take effect for the current scheme. (New cameras come with an empty scheme by default).


Camera parameters

Camera parameters are bound to the current scheme

Photography trigger mode: Please select the trigger mode

Image parameter settings: Manual | One-click settings

One click to adjust brightness and focus, make the image more clear

Light source settings

Light source control: All

Duration: 2000 μs

Gamma: 0.7

4.2.1 Photography trigger mode

The smart camera supports the following photography triggering modes. Regardless of the trigger mode, the photography that triggers automatically when you set the camera parameters is always effective.

- **Auto photography**: The camera automatically takes photos at regular intervals (non-real-time imaging).
- **Manual photography**: The camera takes photos when you click **Take photo** of the plugin.
- **Camera button**: Trigger photography through the button on the side of the camera.

The trigger mode used when the smart camera communicates with the robot not belongs to the above modes, so this option will be left blank after automatic calibration or starting running a project. You can reset if required.

4.2.2 Image parameter settings

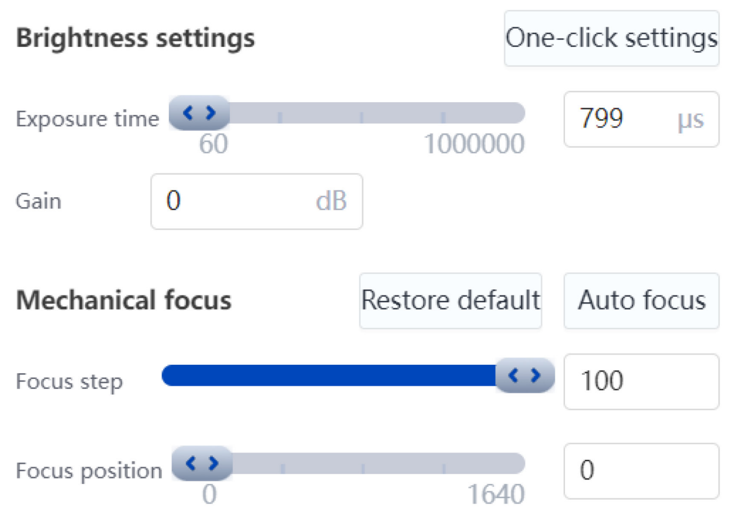
One-click settings

Click **One-click settings**, and the camera will automatically adjust the camera parameters (brightness and focus) based on the site environment.

NOTE: The automatic adjustment will take some time, during which the smart camera cannot be operated.

If the image after the one-click settings does not meet your needs, you can set the detailed parameters manually.

Manual settings



Brightness settings

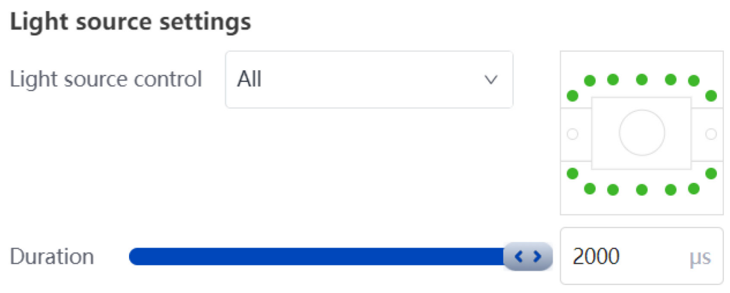
- Click **One-click settings** to automatically adjust the brightness of the camera.
- If **One-click settings** does not meet your requirements, you can manually adjust the brightness of the camera image by modifying the exposure time.

Mechanical focus

- Click **Auto focus**, the camera will automatically adjust the focus position in fixed focus steps and ultimately select the focus position with the sharpest overall image. Click **Restore default** to restore the focus position to the default value.

  If the contrast of the outlines in the image is not clear, the auto-focus may fail. In this case, please focus manually.
- **Focus step**: Set the focus step when adjusting focus.
- **Focus position**: If **Auto focus** does not meet your requirements, you can modify the focus position manually.

4.2.3 Setting light source



- **Light source control**: Select the mode of light source control, including **All**, **Custom** and **None**.
  - **All**: All light beads are on, and the duration setting is effective for all light beads.
  - **Custom**: You can set the on/off status of light beads for each area (click the corresponding area in the diagram on the right, green: on, white: off) and the duration separately. When switched to this option, all light beads are off by default.

    NOTE: The light beads on the left and right sides of the lens are device sights, which indicate the position of the vision field. Click the corresponding light bead to turn on/off the device sight (red: on, white: off).
  - **None**: All light beads are off, and the following settings are ineffective.

- **Duration**: Set the light-up duration of each light bead. It can be set only when the **Light source control** is **All**, the set parameters are also valid for the light beads turned on in the **Custom** mode. After turning off all the light beads and saving, the duration is 100 (minimum) when you open the camera settings again.

4.2.4 Setting gamma

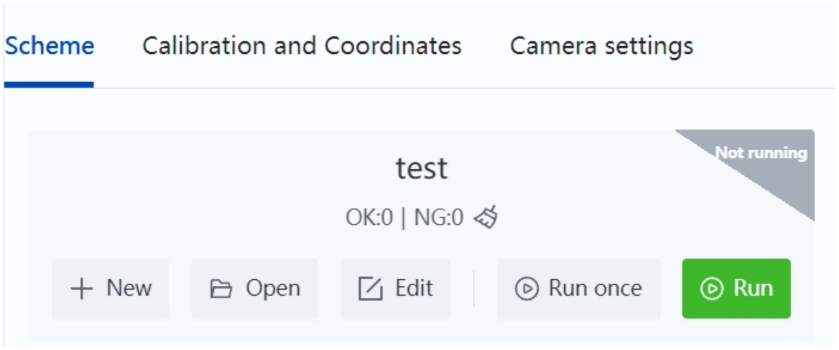
Gamma ?

0.7

You can manually set the **Gamma** to adjust the image contrast. If the Gamma value is between 0.5 and 1, the dark area of the image will become more bright. If the Gamma value is between 1 and 4, the dark area of the image will become less bright. The default value is 0.7.

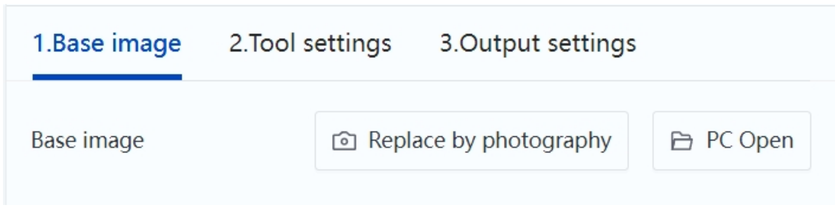
5. Scheme

A scheme is a collection of tools, and multiple tools can be set in a scheme. There is an empty scheme without any tools and configurations that comes with the VX500 smart camera, you need to create your own scheme according your needs.



5.1 Creating scheme

1. Click **New** to create a new scheme.
2. Set the reference image. All tools other than the positioning tool require the ROI (Region of Interest) to be drawn on the reference image.



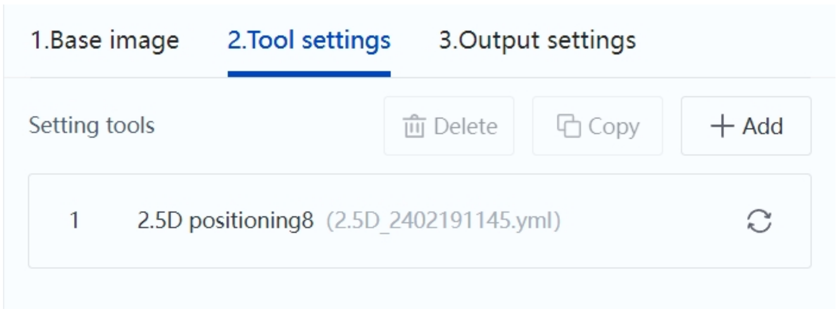
- If the base image has not been set, click **Obtain by photography** and use the photographed image as the base image.
- When there is a base image, the **Obtain by photography** button turns to **Replace by photography**. You can click it to update the base image.
- Click **PC Open**, you can import an image from the PC as a base image.

NOTE

For the base image imported from the PC, you need to make sure that the image is in jpg, bmp or png format, and that the image resolution is the same as the current resolution of the camera.

It is recommended to take photos and save them to the local via smart camera before importing.

3. Go to the **Tool settings** tab, you can click **Add** to add a new tool. See Tool description for details. Before using the positioning tool, complete hand-eye calibration and create coordinate systems first by referring to Calibration and Coordinates.



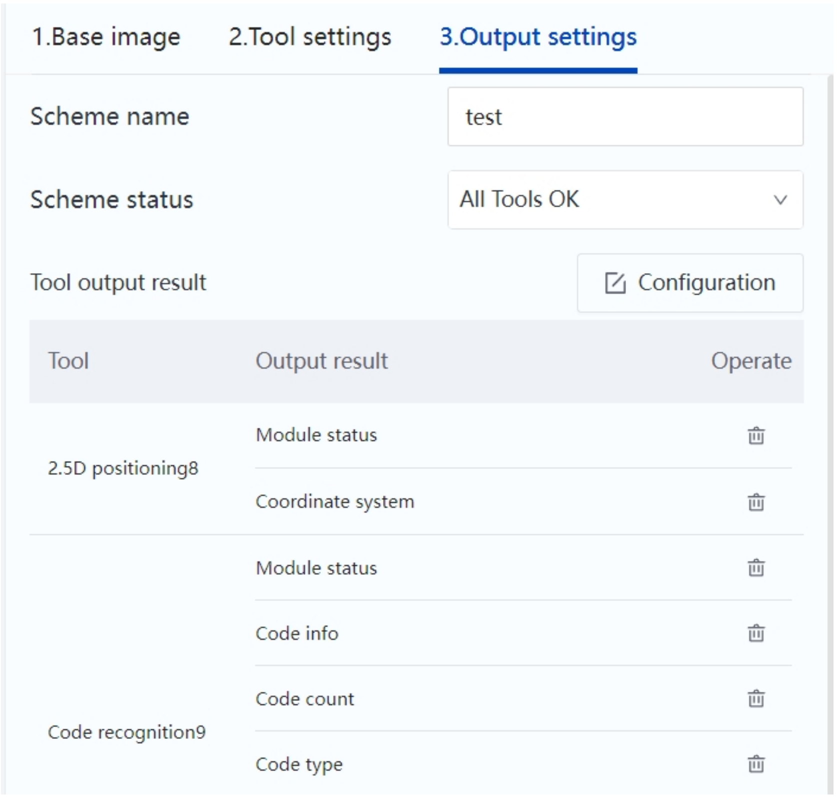
After selecting the added tool:

- You can click **Delete** to delete the tool.
- You can click **Copy** to make a copy of the same tool.

Up to 40 tools can be included in a scheme, and each tool has its own limit. For example, the 2.5D positioning and code recognition module only allow one respectively. You also need to consider the memory usage of the camera. As some modules occupy much memory, it may cause the memory limit to be reached even there are less than 40 tools. In this case, you cannot add more tools.

- You can click ↻ on the right of the 2.5D positioning tool to modify the calibration file to which the tool is bound.
- 2D positioning tool is not editable.
- You can click ☑ on the right of the other tools to modify the tool settings.

4. Go to the **Output settings** tab, enter the scheme name, and select the scheme status. Scheme status:
   - **All tools OK**: When the module status of all tools in the scheme is OK, the scheme running result is OK.
   - **Any tool OK**: When the module status of any tool in the scheme is OK, the scheme running result is OK.



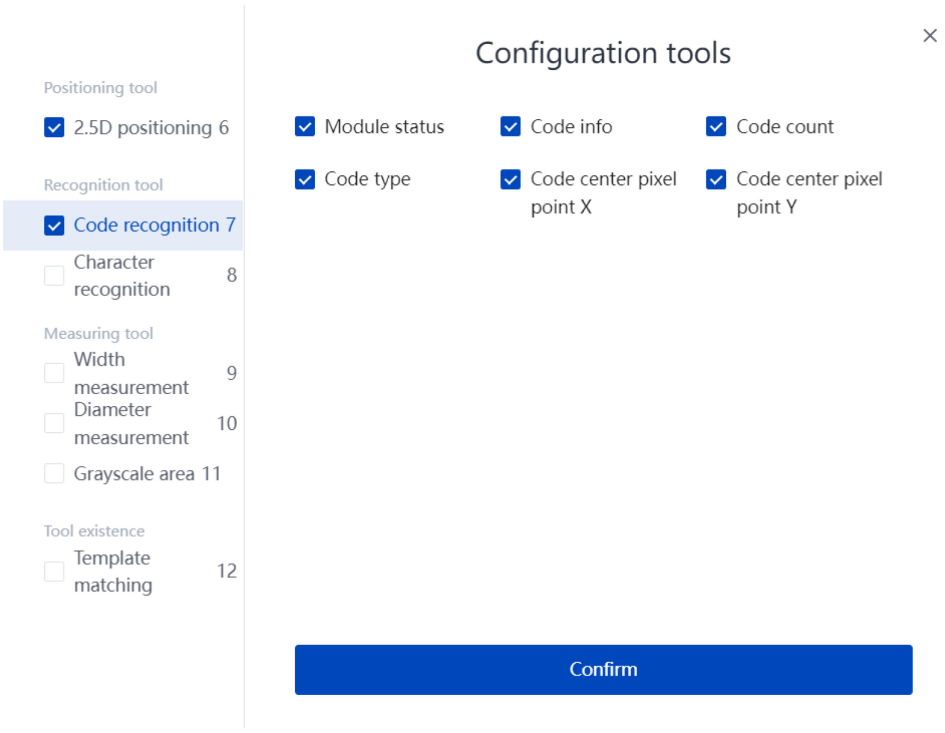
Scheme name: test

Scheme status: All Tools OK

Tool output result

Tool	Output result	Operate
2.5D positioning8	Module status	🗑
	Coordinate system	🗑
Code recognition9	Module status	🗑
	Code info	🗑
	Code count	🗑
	Code type	🗑

5. Click **Configuration** to open the tool output result page. You can select the output results of the added tools (can select multiple). Only the selected output results will be output and can be viewed through the interface or called through programming.



Positioning tool

- [✓] 2.5D positioning 6

Recognition tool

- [✓] Code recognition 7
- [ ] Character recognition 8

Measuring tool

- [ ] Width measurement 9
- [ ] Diameter measurement 10
- [ ] Grayscale area 11

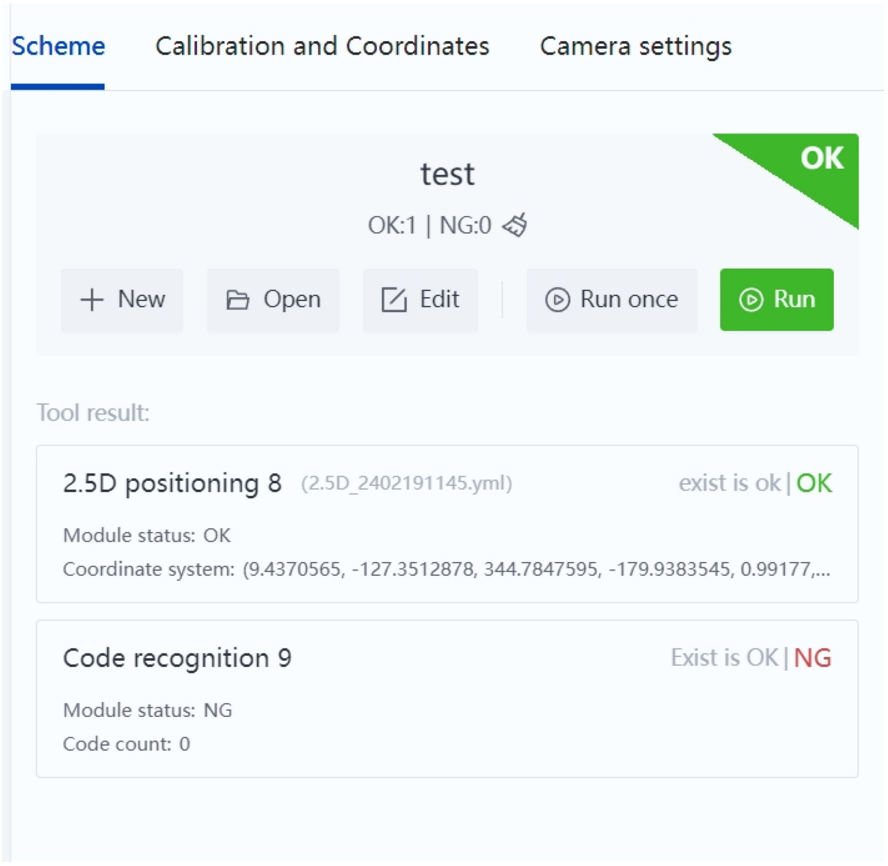
Tool existence

- [ ] Template matching 12

Configuration tools

✓ Module status	✓ Code info	✓ Code count
✓ Code type	✓ Code center pixel point X	✓ Code center pixel point Y

5.2 Running scheme



test

OK:1 | NG:0

Tool result:

2.5D positioning 8 (2.5D_2402191145.yml) exist is ok | OK

Module status: OK

Coordinate system: (9.4370565, -127.3512878, 344.7847595, -179.9383545, 0.99177,...

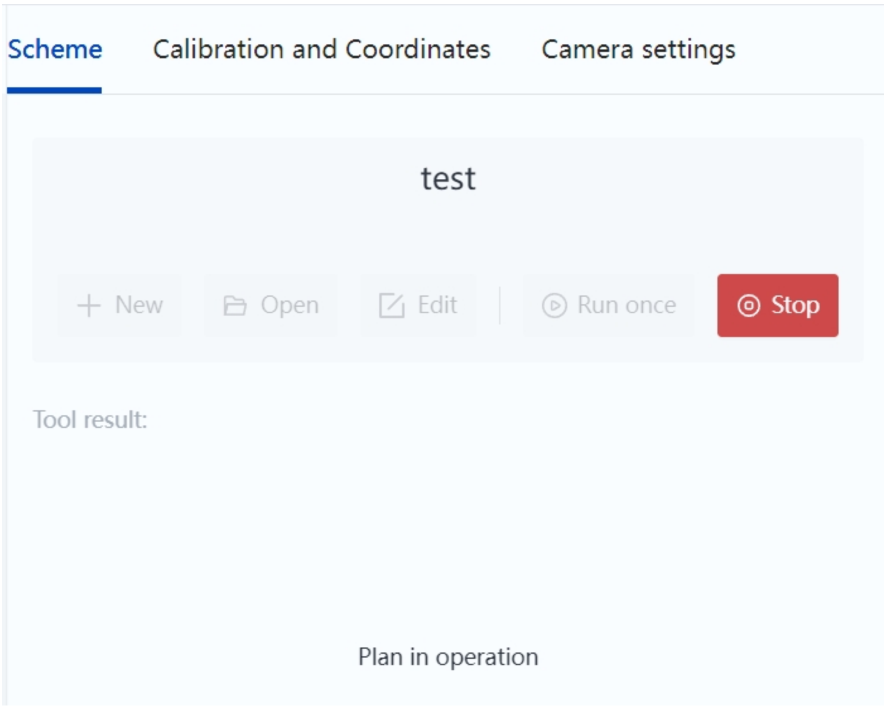
Code recognition 9 Exist is OK | NG

Module status: NG

Code count: 0

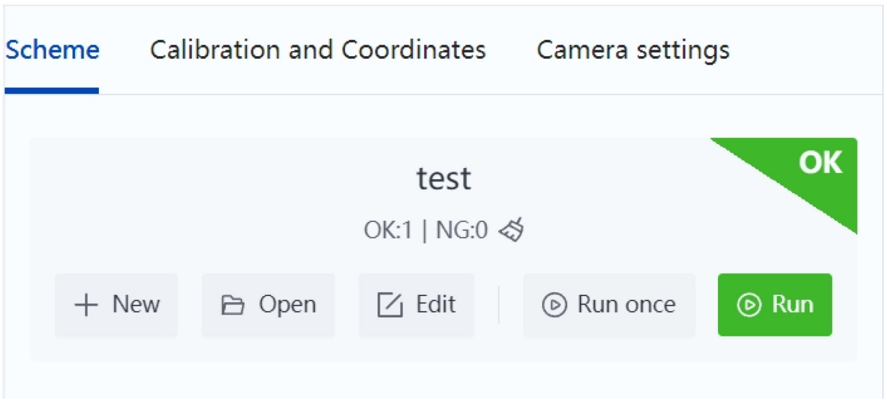
- Click **Run once** to run the current scheme one time (take a photo and recognize it one time), and display the running results.
  - The corner mark on the right of the scheme name represents the real-time running result of the current scheme.

  - The count of the running results of the current scheme is displayed under the scheme name. You can click ⟲ to reset the count to zero.
  - The tool result area displays the running results of each tool in the current scheme.
- Click **Run** to run the current scheme continuously. The running results will not be displayed while running continuously.



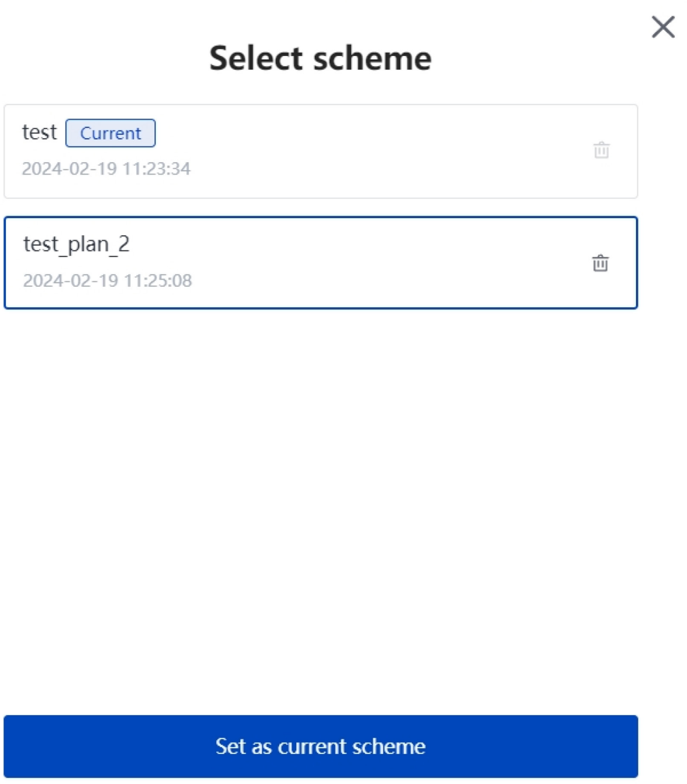
After running the scheme, you can write program by referring to Blockly/Script programming and Demos.

5.3 Managing scheme



Click **Edit** to modify the current scheme.

Click **Open** to open the **Select scheme** page.



- Select a scheme and click **Set as current scheme**, you can set the selected scheme as the current scheme.
- Click 🗑 to delete the scheme. The current scheme cannot be deleted.

6. Calibration and Coordinates

6.1 Vision scheme overview

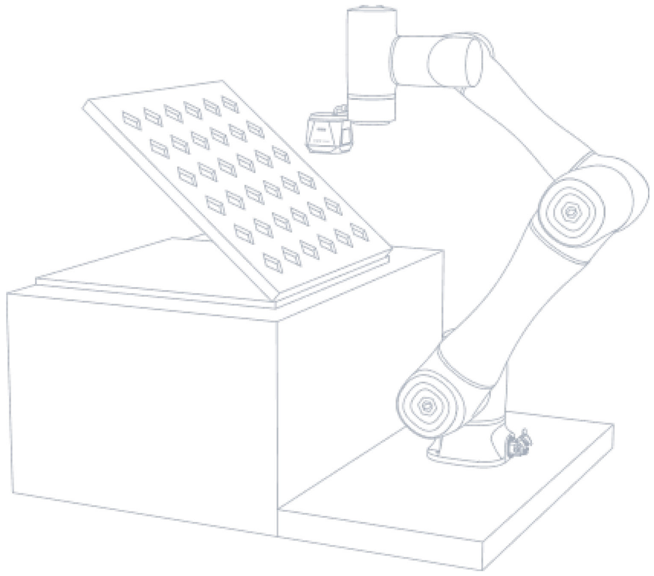
6.1.1 2.5D vision scheme

2.5D positioning is a self-developed algorithm by Dobot, which can solve the problem of inaccurate positioning caused by spatial height conversion (such as uneven ground, tilting, etc.), while conventional 2D positioning cannot solve such problem. For example, if the robot arm is installed on an AGV and moves with it to the workstation, the robot arm can accurately position the target point without the need for precise positioning by the AGV or the need for even ground, as long as the 2.5D positioning code is within the vision field of the camera and can be correctly recognized.

The 2.5D scheme recognizes the 2.5D positioning code rather than directly recognizing the target. It is mainly realized by the following steps:

1) 2.5D hand-eye calibration: Realize the conversion from image coordinates to spatial physical coordinates. Recalibration is required only when the positional relationship between the camera and the robot arm (hand-eye position) changes.
2) Create 2.5D coordinate system: Create a 2.5D coordinate system by recognizing the 2.5D positioning code.
3) Add tool: Add 2.5D positioning tool and output settings to the scheme.
4) Save target point: Teach the target point in the project and save it under the created 2.5D coordinate system.

After the target point is saved, as long as the relative position relationship between the target point and the 2.5D positioning code remains unchanged, the robot arm can accurately locate the target point by recognizing the 2.5D positioning code.



6.1.2 2D vision scheme

2D positioning realizes the conversion from image coordinates to plane physical coordinates through 2D calibration (the distance between the camera and the plane cannot be changed, and the camera lens must be parallel to the target plane). Based on template matching, the coordinates of the object in a specific shape within the camera vision field can be obtained. You can pick up the object or create a coordinate system based on the object to locate the object within the coordinate system.

The 2D scheme positions by recognizing a specified image in the work area. It is mainly realized by the following steps:

1) 2D hand-eye calibration: Realize the conversion from image coordinates to plane physical coordinates. In addition to the change in hand-eye position, recalibration is also required when the

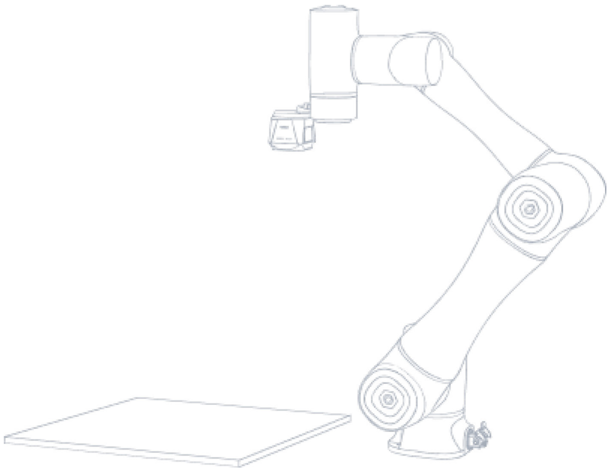
photo point in the actual application scene is different from the calibrated photo point.

2) Create 2D coordinate system: Create a 2D coordinate system by recognizing the image specified by the user.

3) Add tool: Add 2D positioning tool and output settings to the scheme.

4) Save target point: Teach the target point in the project and save it under the created 2D coordinate system.

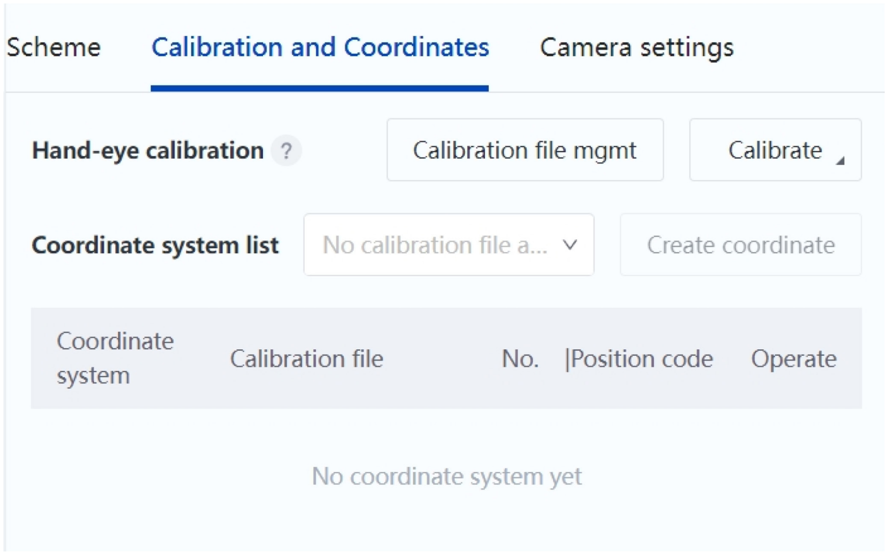
After the target point is saved, as long as the relative position relationship between the target point and the specified image remains unchanged (e.g., the specified image is the target object), the robot arm can accurately locate the target point by recognizing the image.



6.2 2.5D calibration and coordinate system

6.2.1 2.5D calibration

You can perform hand-eye calibration on **Calibration and Coordinates** page.



Coordinate system	Calibration file	No.	Position code	Operate

No coordinate system yet

The hand-eye calibration is used to confirm the conversion relationship between the physical coordinate system at the end of the robot arm (hand) and the image coordinate system of the camera (eye), so as to convert from the image coordinates of visual recognition to the physical coordinates of the robot arm motion.

NOTE

To ensure accuracy, when performing 2.5D calibration, you need to:

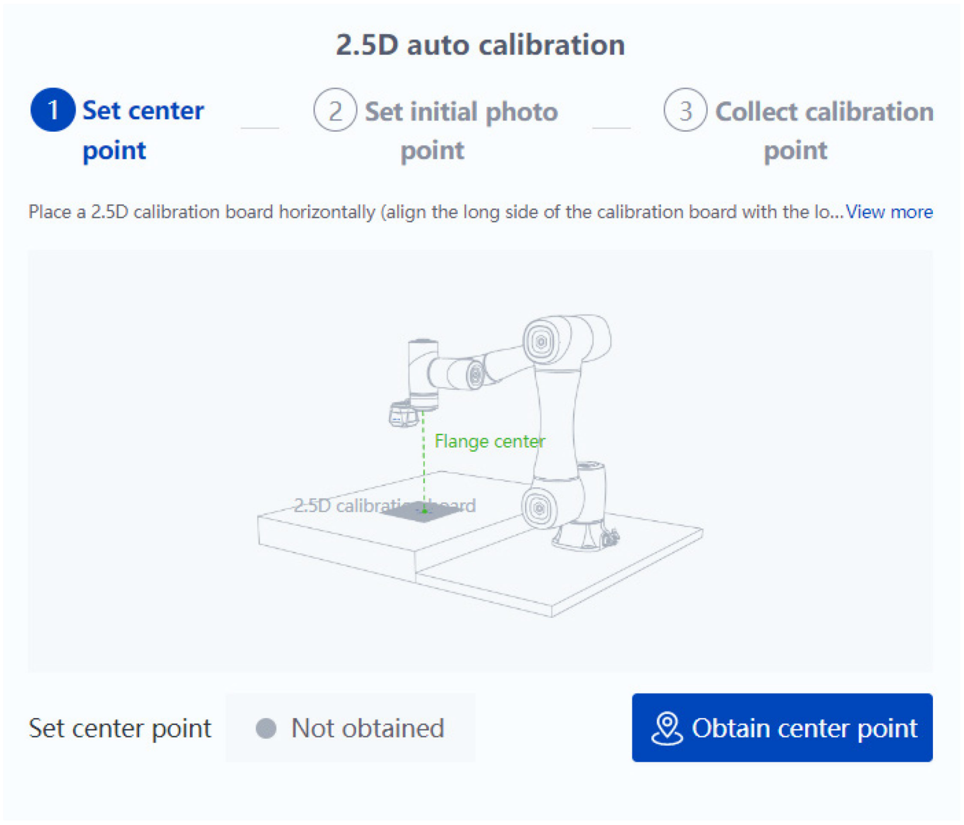
1. Set the User coordinate system and Tool coordinate system to 0 in Jog panel.
2. Keep the joint angle of J3 positive when setting the calibration points.

2.5D automatic calibration

After you teach the center point of the calibration board and the initial photo point, the robot arm automatically generates 25 calibration points, and moves to the corresponding points one by one to take a photo. Then a calibration file will be generated based on the photography result. To use this calibration method, you need to ensure that there are no obstacles within the workspace of the robot arm.

Click **Calibrate > 2.5D auto calibration** to enter the “2.5D auto calibration” page.

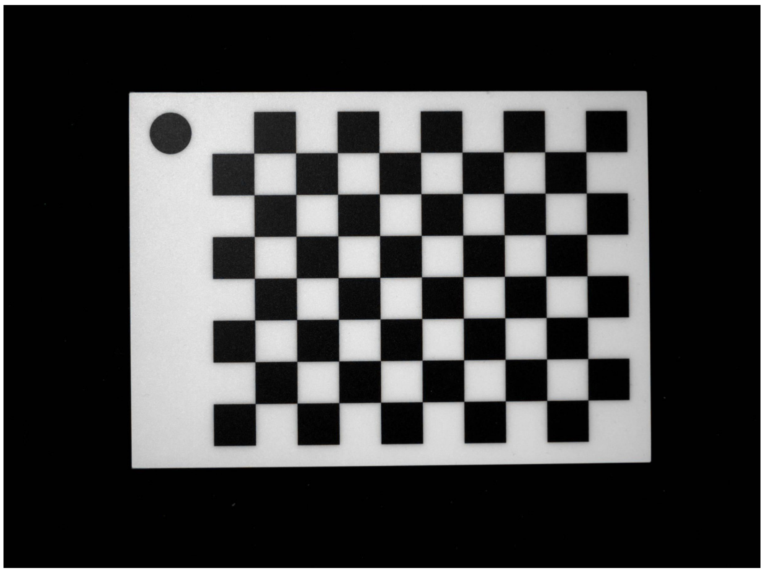
① **Set center point**



Place a 2.5D calibration board horizontally (align the long side of the calibration board with the lo...View more

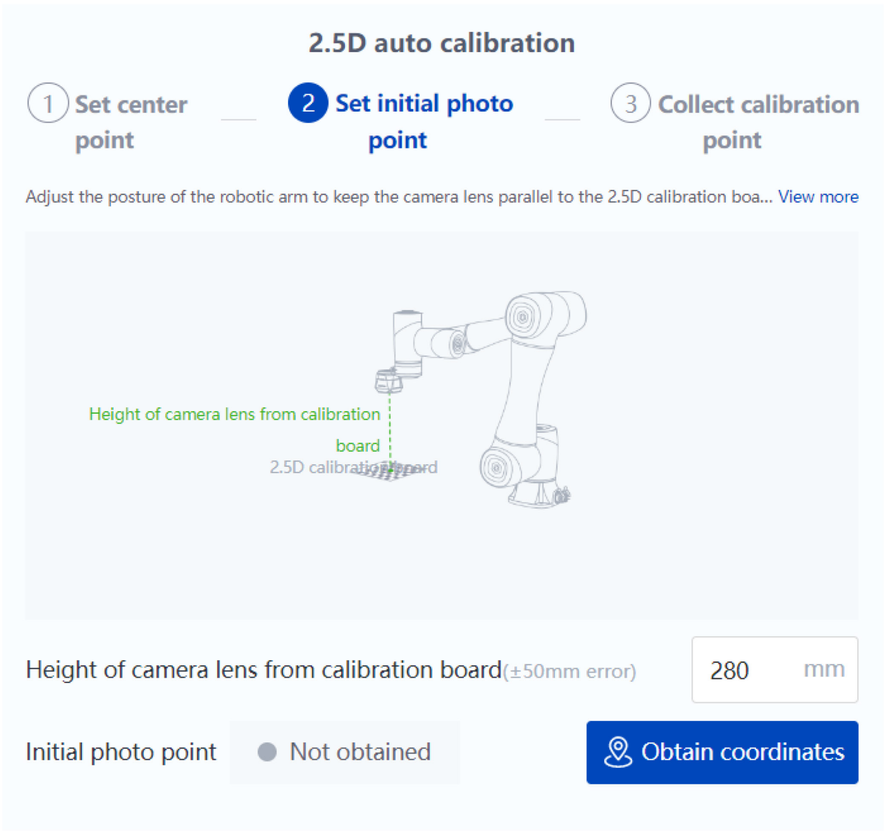
Set center point ● Not obtained

1) Place a 2.5D calibration board horizontally (align the long side of the calibration board with the long side of the vision field). It is recommended to set the calibration background to white or black, as shown in the figure below.



2) Move the robot arm to align the center of the end flange with the center of the calibration board (You can install a calibration needle at the end of the robot arm for alignment, with no height requirement, and try to keep the error less than 10mm). After alignment, **Obtain center point**.

② **Set initial photo point**



2.5D auto calibration

1 Set center point — 2 Set initial photo point — 3 Collect calibration point

Adjust the posture of the robotic arm to keep the camera lens parallel to the 2.5D calibration boa... View more

Height of camera lens from calibration board(±50mm error) 280 mm

Initial photo point ● Not obtained

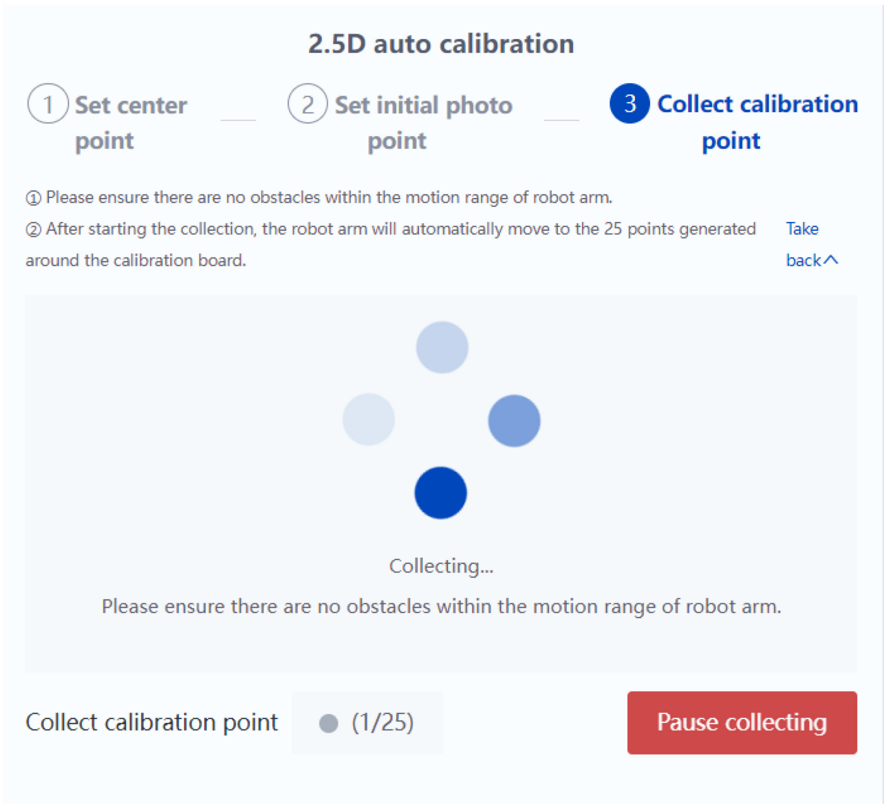
Obtain coordinates

1) Adjust the posture of the robot arm to keep the camera lens parallel to the 2.5D calibration board, and ensure that the calibration board is located in the center of the image vision field and **accounts for 70% of the vision field**. It is recommended that the camera lens be at a height of 250mm – 350mm from the calibration board, RY is 0°, RX is ±180° (camera towards base) or 0° (camera back towards base).

2) Set the **Height of camera lens from calibration board** according to the actual height, and adjust the camera parameters to make the image clear, then **Obtain coordinates**.

③ Collect calibration point

The robot has automatically generated 25 points. Please ensure there are no obstacles within the motion range of the robot arm. Click **Start collecting**, and the robot will automatically move to the points and collect the images.



① Please ensure there are no obstacles within the motion range of robot arm.

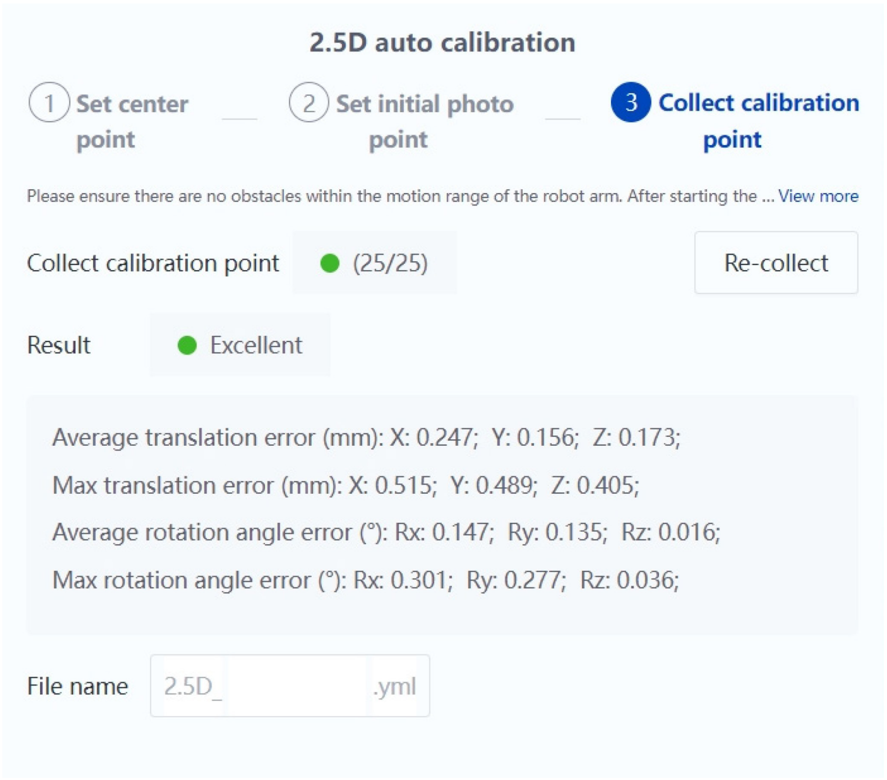
② After starting the collection, the robot arm will automatically move to the 25 points generated around the calibration board.

Collecting...

Please ensure there are no obstacles within the motion range of robot arm.

Collect calibration point ● (1/25)

Decide whether to recalibrate according to the calibration result and error data.



Please ensure there are no obstacles within the motion range of the robot arm. After starting the ...

Collect calibration point ● (25/25)

Result ● Excellent

Average translation error (mm): X: 0.247; Y: 0.156; Z: 0.173;

Max translation error (mm): X: 0.515; Y: 0.489; Z: 0.405;

Average rotation angle error (°): Rx: 0.147; Ry: 0.135; Rz: 0.016;

Max rotation angle error (°): Rx: 0.301; Ry: 0.277; Rz: 0.036;

File name 2.5D_ .yml

- When the calibration result is **Excellent**, enter the file name (can only contain English letters, numbers and underscores) and save.

- When the calibration result is **Good**, you can view the error data and decide whether to recalibrate or save the calibration file directly according to the site requirement for calibration accuracy.
- When the calibration result is **Failed**, you need to set and calibrate again.

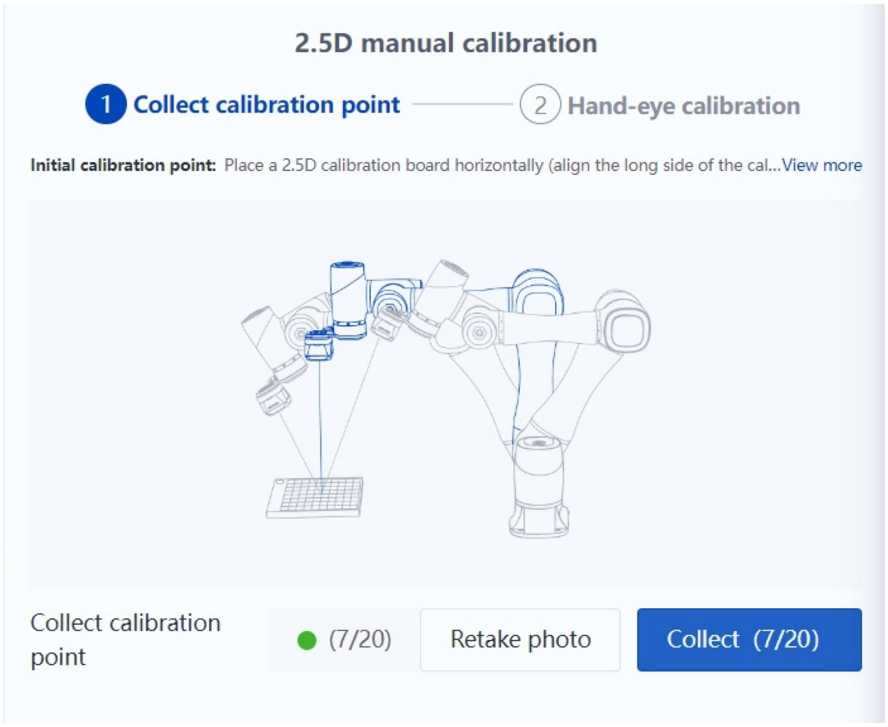
After saving the file, the interface returns to the **Calibration and Coordinates** page. As the calibration file is bound to the robot, you need to recalibrate when the robot is changed.

2.5D manual calibration

If automatic calibration is impossible due to scene limitations (e.g. there are obstacles within the motion range of robot arm), you can use 2.5D manual calibration. 2.5D manual calibration includes 20 self-planned calibration points at which the camera can capture a clear and complete 2.5D calibration board from different distances and angles. You need to manually control the robot arm to move to the 20 calibration points and take photos, and the system will generate a calibration file based on the photography result.

Click **Calibrate > 2.5D manual calibration** to enter the "2.5D manual calibration" page.

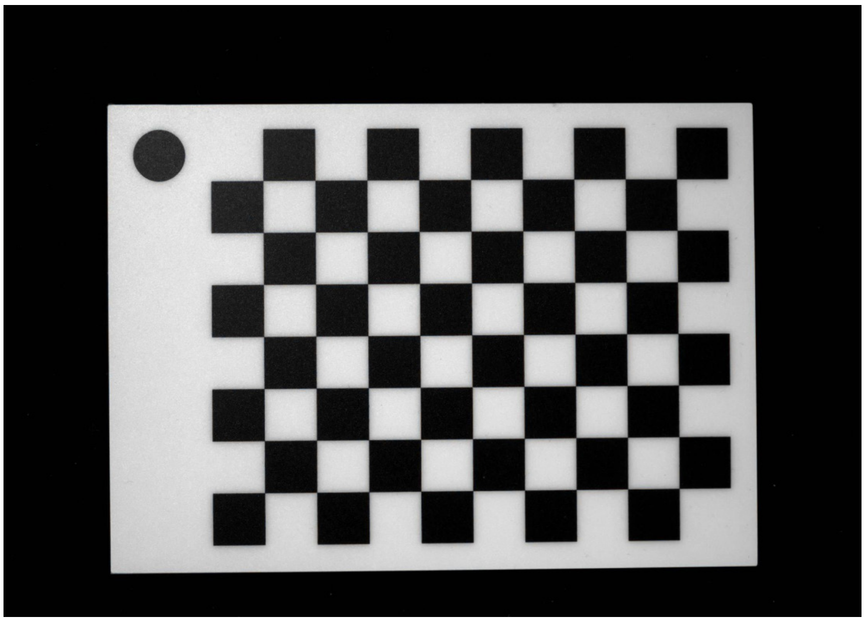
① Collect calibration point



Initial calibration point: Place a 2.5D calibration board horizontally (align the long side of the cal...View more

Initial calibration point

1) Place a 2.5D calibration board horizontally (align the long side of the calibration board with the long side of the vision field). It is recommended to set the calibration background to white or black, as shown in the figure below.



2) Adjust the posture of the robot arm to keep the camera lens parallel to the 2.5D calibration board, and ensure that the calibration board is located in the center of the image vision field and **accounts for 70% of the vision field**. It is recommended that the camera lens be at a height of 250mm – 350mm from the calibration board, RY is 0°, RX is ±180° (camera towards base) or 0° (camera back towards base).
3) Adjust the camera parameters to make the image clear, and **collect** calibration points.

Other calibration points

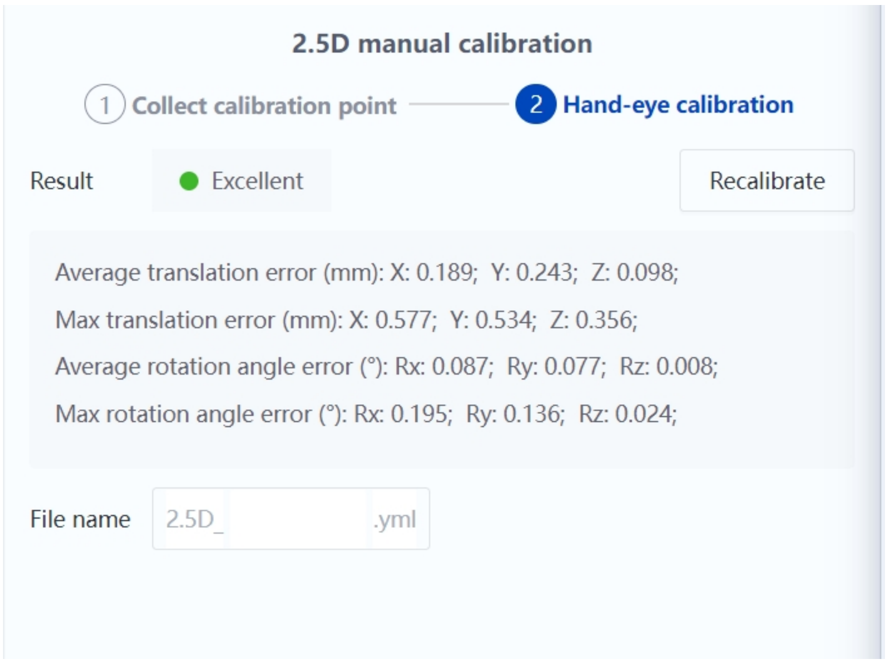
1) Adjust the posture of the robot arm to make a large change in its posture (any of the X/Y/Z values is greater than 20mm, any of the RX/RY/RZ values is greater than 3°).
2) **Collect** calibration points. Repeat the steps above until 20 calibration points are collected successfully.

NOTE

It is not allowed to adjust the camera parameters after the initial calibration points are collected. Please adjust the posture of the robot arm to make the calibration board clear and within the vision field. When taking photos (collecting), the robot arm remains stationary.

② Hand-eye calibration

Decide whether to recalibrate according to the calibration result and error data.


2.5D manual calibration

Result ● Excellent

Average translation error (mm): X: 0.189; Y: 0.243; Z: 0.098;

Max translation error (mm): X: 0.577; Y: 0.534; Z: 0.356;

Average rotation angle error (°): Rx: 0.087; Ry: 0.077; Rz: 0.008;

Max rotation angle error (°): Rx: 0.195; Ry: 0.136; Rz: 0.024;

File name 2.5D_ .yml

- When the calibration result is **Excellent**, enter the file name (can only contain English letters, numbers and underscores) and save.
- When the calibration result is **Good**, you can view the error data and decide whether to recalibrate or save the calibration file directly according to the site requirement for calibration accuracy.
- When the calibration result is **Failed**, you need to set and calibrate again.

After saving the file, the interface returns to the **Calibration and Coordinates** page. As the calibration file is bound to the robot, you need to recalibrate when the robot is changed.

6.2.2 Common causes of calibration failure

If the calibration error is large (greater than 2 mm) and causes failure, follow the steps below to troubleshoot.

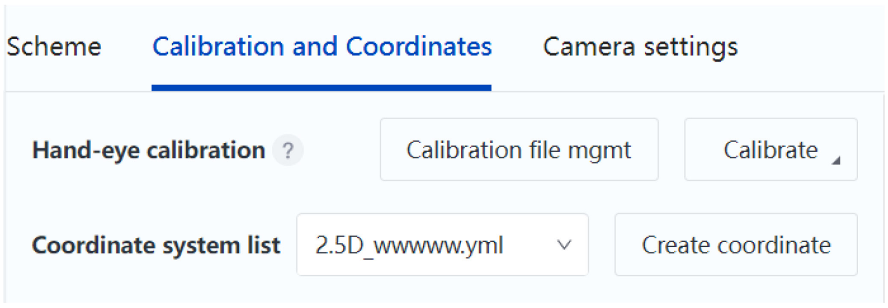
1. Check if the camera and robot arm are firmly installed.
2. Check if the calibration board is fixed relative to the position of the robot arm base.

Example: Using an AGV equipped with a robotic arm for calibration to ensure that the AGV car cannot shake/move.

3. Check if the robot arm is calibrated using a positive (J3>0) posture, otherwise there may be large errors.
4. If the calibration still fails, it may be due to poor accuracy of the robot arm body, and the robot arm needs to be replaced or recalibrated.

If the collection of calibration points fails, check whether the calibration board is out of view or whether the camera parameters are set appropriately. It is recommended to try turning off the light source and adjusting the exposure time to make the image clear.

6.2.3 Creating coordinate system based on 2.5D positioning code

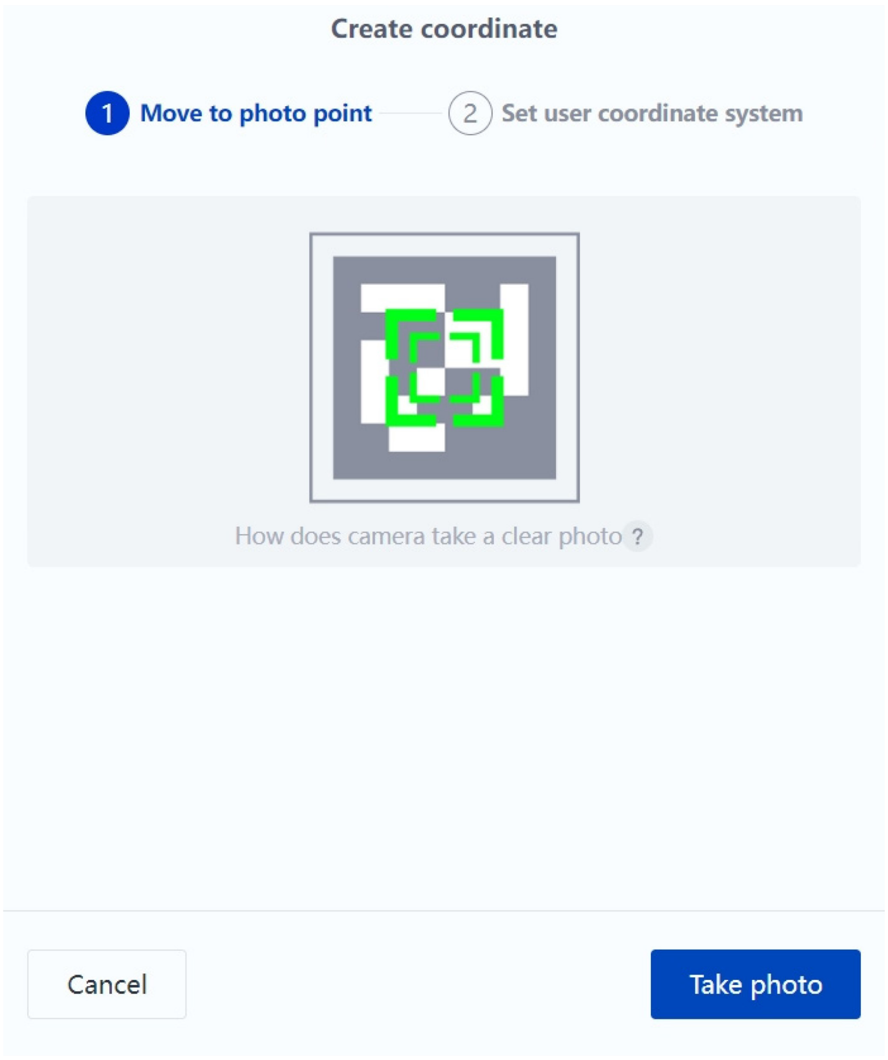


Hand-eye calibration ?

Coordinate system list 2.5D_wwwww.yml

Before creating a coordinate system, you need to select a saved 2.5 D calibration file. Then click **Create coordinate system**.

① **Move to photo point**


Create coordinate
1 Move to photo point
2 Set user coordinate system
How does camera take a clear photo ?
Cancel
Take photo

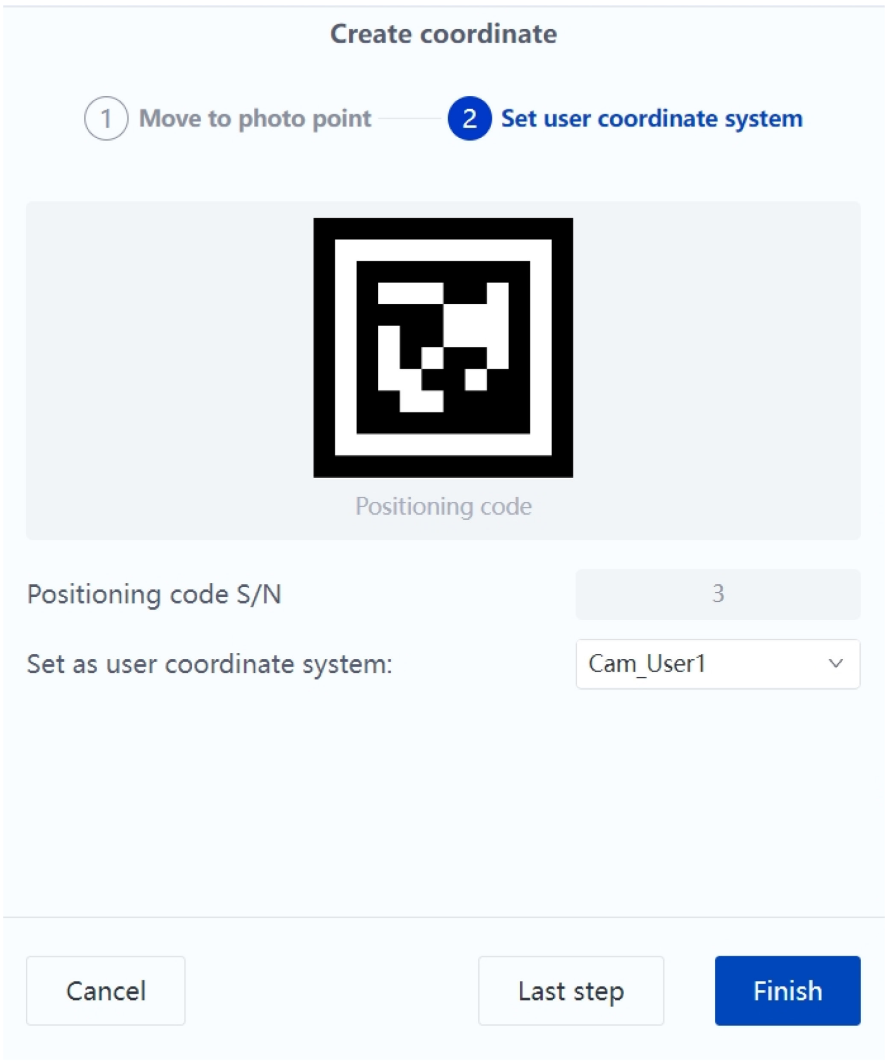
Fix the 2.5D positioning code on a flat surface. Move the robot arm to make the camera lens parallel to the 2.5D positioning code, and make the 2.5D positioning code in the center of the image (the recommended height is 190mm from the front of the camera lens to the calibration board). Then refocus the camera and click **Take photo** when the image is clear. Click **Next** after the 2.5D positioning code is recognized successfully.

NOTE

1. Before taking a photo, you need to set the user and tool coordinate systems to 0 in Jog panel.
2. It is recommended to save the point in the project to serve as the photo point next time when you modify the coordinate system.

② **Set user coordinate system**

The system will recognize the serial No. of the calibration board based on the photography result. Select the name of the user coordinate system that you want the calibration board to correspond to, and click **Finish** to save it. If the coordinate system fails to be recognized, adjust the brightness of the image by adjusting the exposure time manually.


Create coordinate

1 Move to photo point — 2 Set user coordinate system

Positioning code S/N: 3

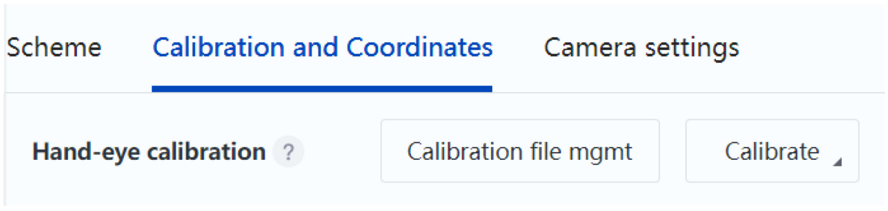
Set as user coordinate system: Cam_User1

After you save the coordinate system, the interface returns to the **Calibration and Coordinates** page and displays the created coordinate system and corresponding calibration board. Clicking the calibration board diagram can view a larger image.

6.3 2D calibration and coordinate system

6.3.1 2D calibration

You can perform hand-eye calibration on **Calibration and Coordinates** page.



The hand-eye calibration is used to confirm the conversion relationship between the physical coordinate system at the end of the robot arm (hand) and the image coordinate system of the camera (eye), so as to convert from the image coordinates of visual recognition to the physical coordinates of the robot arm motion.

NOTE

To ensure accuracy, when performing 2D calibration, you need to:

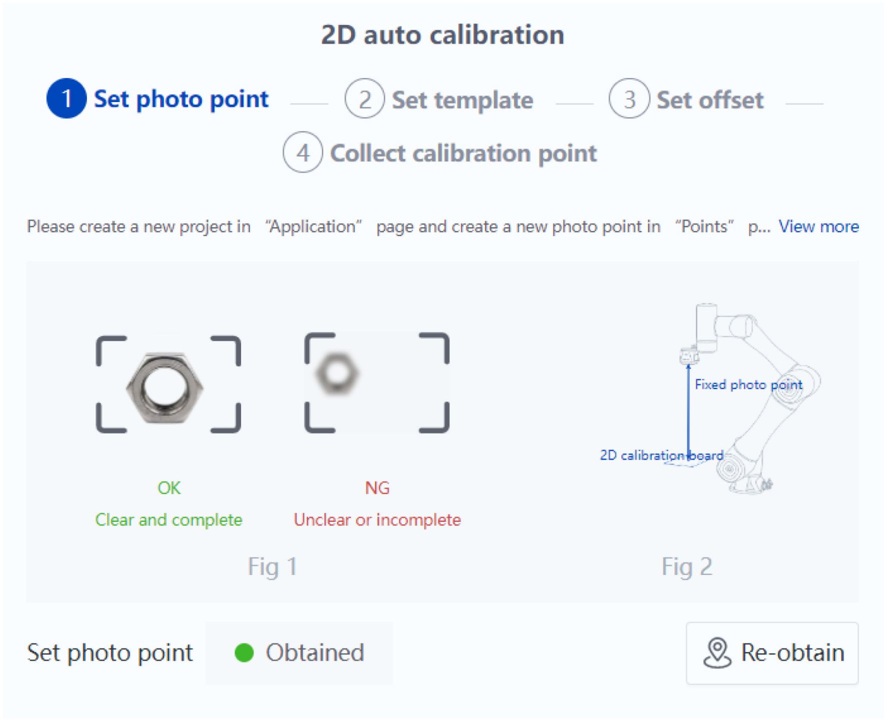
1. Set the user coordinate system to 0 in Jog panel.
2. Keep the joint angle of J3 positive when setting the calibration points.

2D automatic calibration

For 2D automatic calibration, the robot arm automatically generates 14 calibration points after you set the photo point, template and offset. Then the robot arm moves to the corresponding points one by one to obtain the image coordinates and physical coordinates, and finally the system will generate a calibration file. To use this calibration method, you need to ensure that there are no obstacles within the workspace of the robot arm.

Click **Calibrate > 2D auto calibration** to enter the "2D auto calibration" page.

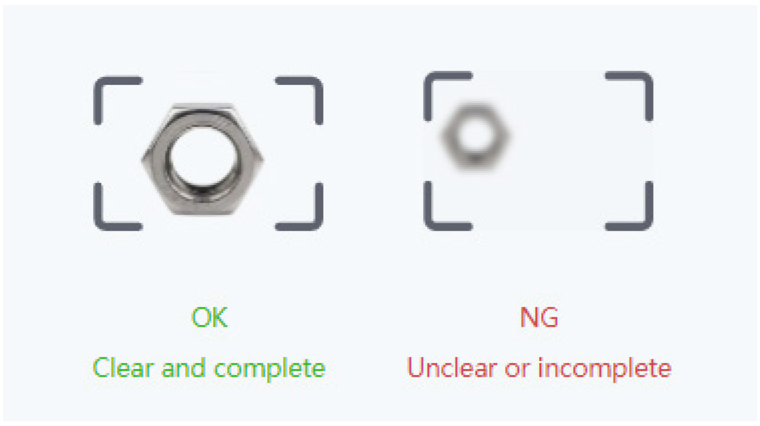
① **Set photo point**


2D auto calibration

Please create a new project in "Application" page and create a new photo point in "Points" p... View more

Adjust photo point

1) Please create a new project in **Application** (Blockly/Script programming) and create a new point (recommended to set the alias as "2D photo point") in **Points** page. Manually modify RX to 180°/-180° and RY to 0°, and move to that point to keep the camera lens parallel to the work plane (e.g., the plane where the detected object is located in the actual application).
2) Adjust the X, Y, Z, and RZ of the robot arm so that the working detection area accounts for 90% of the image vision field (refer to the figure below).



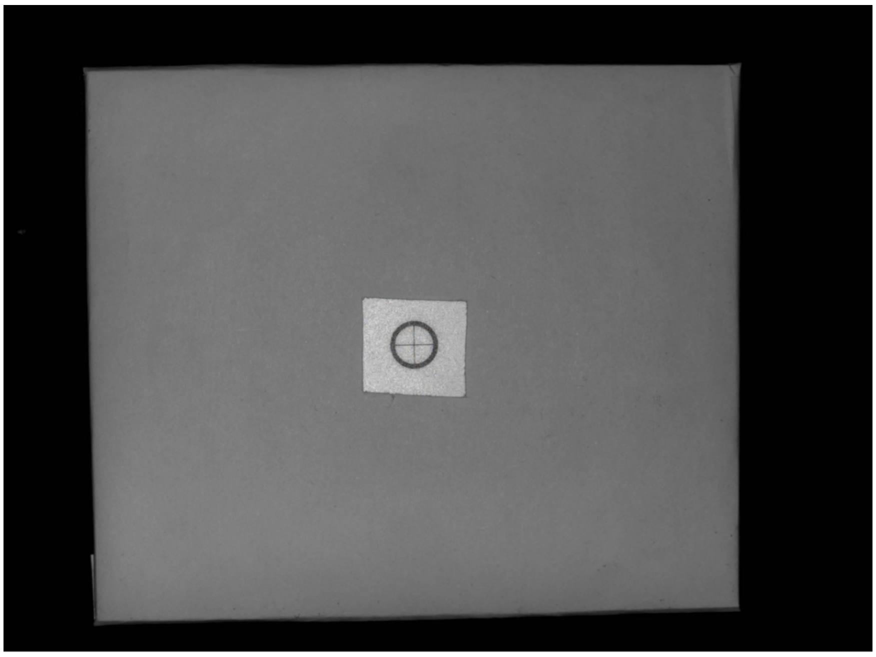
3) Overwrite the current point to "2D photo point".

NOTE

As the 2D calibration only supports taking photos at fixed point, when using the calibration file, make sure that the robot arm is at the same point every time you take a photo.

Place 2D calibration board

1) Place the 2D calibration board on the work plane and cover it with white paper, leaving only one circle and making sure that the circle is in the center of the vision field.

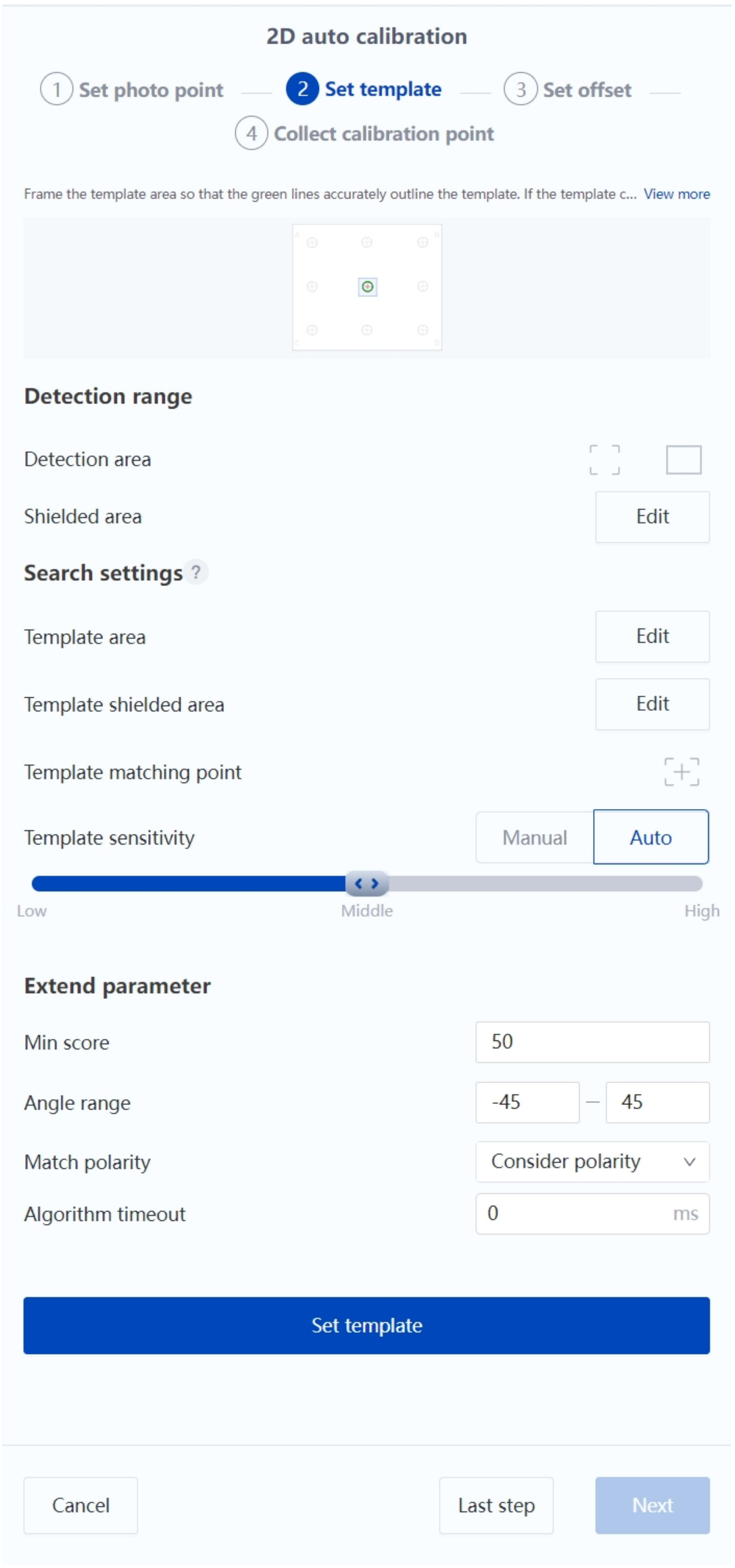


2) Adjust the camera parameters to make the image clear, and click **Obtain photo point**.

NOTE

During 2D calibration, if you find that the image taken by the camera is not clear, you should stop the calibration and adjust the camera parameters to make the image clear, and then recalibrate. After calibration is completed, the focus parameters can no longer be adjusted.

② **Set template**



2D auto calibration

1 Set photo point — 2 Set template — 3 Set offset — 4 Collect calibration point

Frame the template area so that the green lines accurately outline the template. If the template c... View more

Detection range

Detection area

Shielded area: Edit

Search settings ?

Template area: Edit

Template shielded area: Edit

Template matching point

Template sensitivity: Manual [✓ Auto]

Low — Middle — High

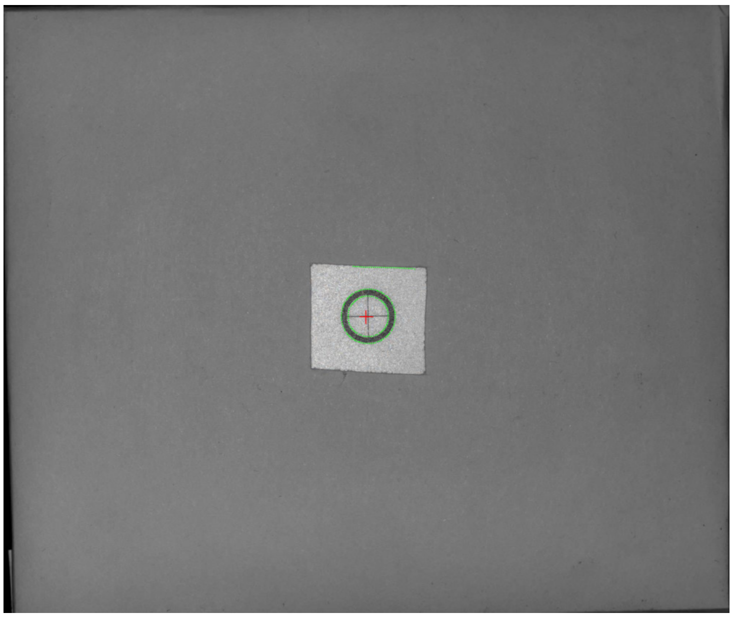
Extend parameter

Min score	50
Angle range	-45 — 45
Match polarity	Consider polarity
Algorithm timeout	0 ms

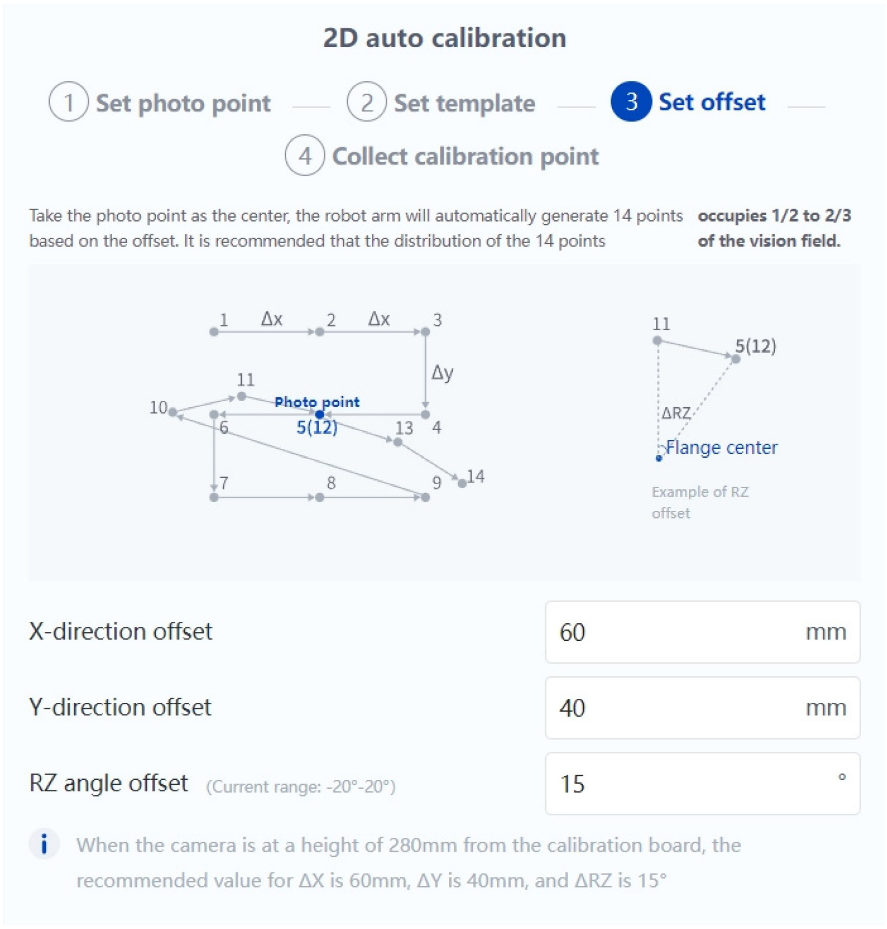
Make sure the calibration board is in the center of the screen. Click **Edit** on the right of

Template area, frame the circle in the center of the calibration board using the rectangle tool. Then click on the right of **Template matching point**, and click the intersection of the cross in the center of the circle. For more details on advanced settings of templates, refer to the instructions for Template matching.

If the green line in the image accurately outlines the template, click **Set template**. If not, adjust the template parameters and re-recognize it.



③ Set offset


2D auto calibration

Take the photo point as the center, the robot arm will automatically generate 14 points based on the offset. It is recommended that the distribution of the 14 points **occupies 1/2 to 2/3 of the vision field.**

X-direction offset	60 mm
Y-direction offset	40 mm
RZ angle offset (Current range: -20°-20°)	15 °

When the camera is at a height of 280mm from the calibration board, the recommended value for ΔX is 60mm, ΔY is 40mm, and ΔRZ is 15°

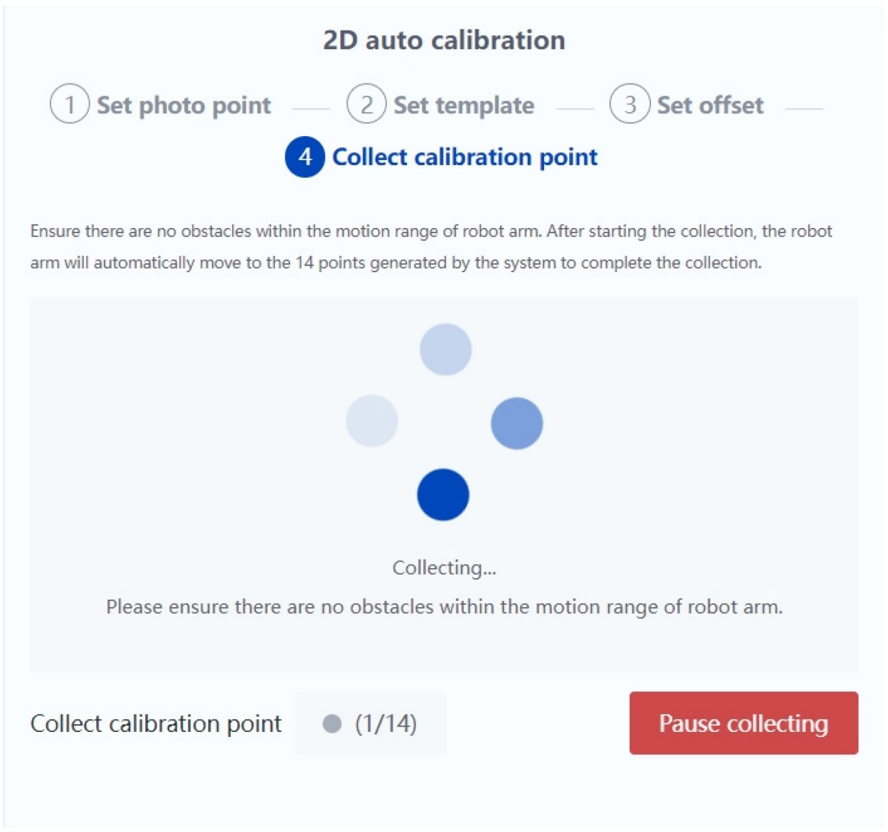
Set the offset of the automatically generated points. The plugin will automatically generate 14 points centered on the photo point based on the offset you set. The first nine points are translation points (translation along the X and Y axes), the last five points are rotation points (rotation along the Rz axis only, the distribution will be different depending on the angles). Point 5 and point 12 have the same coordinates.

Please make sure that all 14 point templates are within the vision field of the camera and that they do not collide with obstacles or the robot arm when the robot is moving. When the camera is at a height of 280mm from the calibration board, the recommended offset is 60mm in X direction, 40mm in Y direction, and 15° in RZ angle.

Click **Next** when the settings are completed.

④ Collect calibration point

Make sure there are no obstacles within the motion range of robot arm, click **Start collecting** and the robot arm will start moving automatically and take photos to recognize the calibration points.

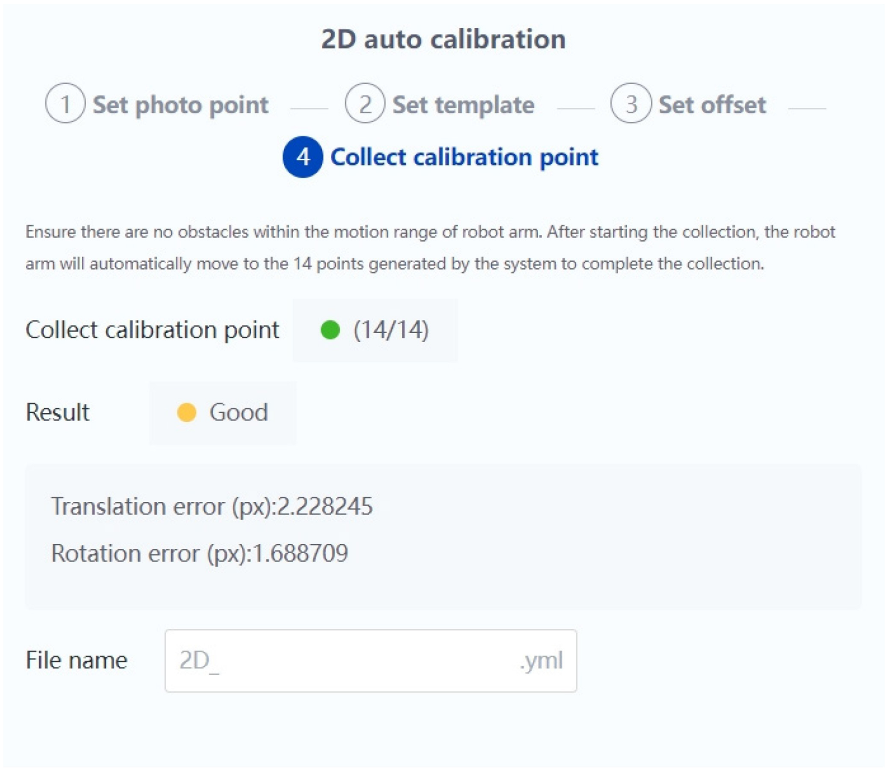


Ensure there are no obstacles within the motion range of robot arm. After starting the collection, the robot arm will automatically move to the 14 points generated by the system to complete the collection.

Collecting...

Please ensure there are no obstacles within the motion range of robot arm.

After collecting the points, decide whether to complete the calibration according to the calibration result and site requirement for accuracy (file name can only contain English letters, numbers and underscores).


2D auto calibration

Ensure there are no obstacles within the motion range of robot arm. After starting the collection, the robot arm will automatically move to the 14 points generated by the system to complete the collection.

Collect calibration point ● (14/14)

Result ● Good

Translation error (px):2.228245

Rotation error (px):1.688709

File name 2D_ .yml

- When the calibration result is **Excellent**, enter the file name (can only contain English letters, numbers and underscores) and save.
- When the calibration result is **Good**, you can view the error data and decide whether to recalibrate or save the calibration file directly according to the site requirement for calibration accuracy.
- When the calibration result is **Failed**, you need to set and calibrate again.

After saving the file, the interface returns to the **Calibration and Coordinates** page. As the calibration file is bound to the robot, you need to recalibrate when the robot is changed.

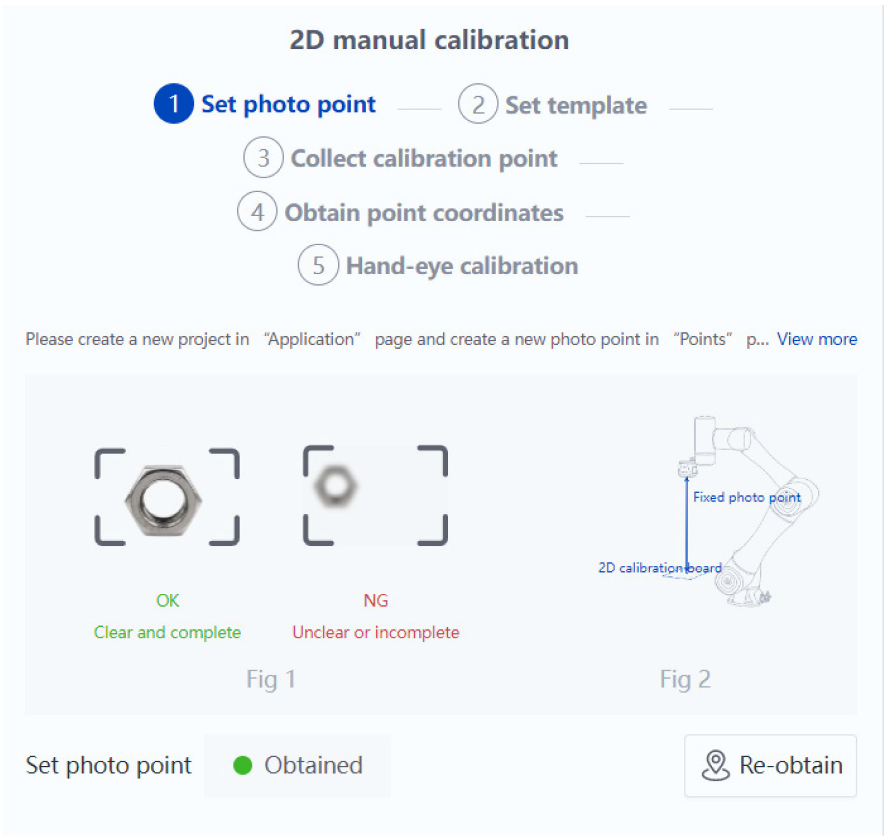
2D manual calibration

If automatic calibration is impossible due to scene limitations (e.g. there are obstacles within the motion range of robot arm), you can use 2D manual calibration. In addition, if the angle of the recognized object changes greatly when using the 2D positioning function, the positioning accuracy of 2D manual calibration may be higher (provided that the needle tip is precisely aligned with the template matching point during manual calibration).

For 2D manual calibration, you need to collect nine calibration points based on the template match in the photographed image. Then control the robot arm to move to the corresponding points in sequence and record the point information to generate a calibration file.

Click **Calibrate > 2D manual calibration** to enter the “2D manual calibration” page.

① Set photo point


2D manual calibration

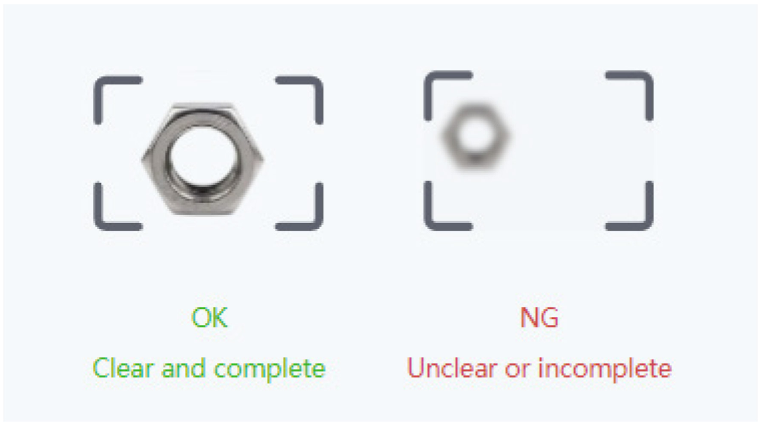
1 Set photo point — 2 Set template — 3 Collect calibration point — 4 Obtain point coordinates — 5 Hand-eye calibration

Please create a new project in "Application" page and create a new photo point in "Points" p... View more

Set photo point ● Obtained Re-obtain

Adjust photo point

1) Please create a new project in **Application** (Blockly/Script programming) and create a new point (recommended to set the alias as "2D photo point") in **Points** page. Manually modify RX to 180°/-180° and RY to 0°, and move to that point to keep the camera lens parallel to the work plane (e.g., the plane where the detected object is located in the actual application).
2) Adjust the X, Y, Z, and RZ of the robot arm so that the working detection area accounts for 90% of the image vision field (refer to the figure below).



3) Overwrite the current point to "2D photo point".

NOTE

As the 2D calibration only supports taking photos at fixed point, when using the calibration file, make sure that the robot arm is at the same point every time you take a photo.

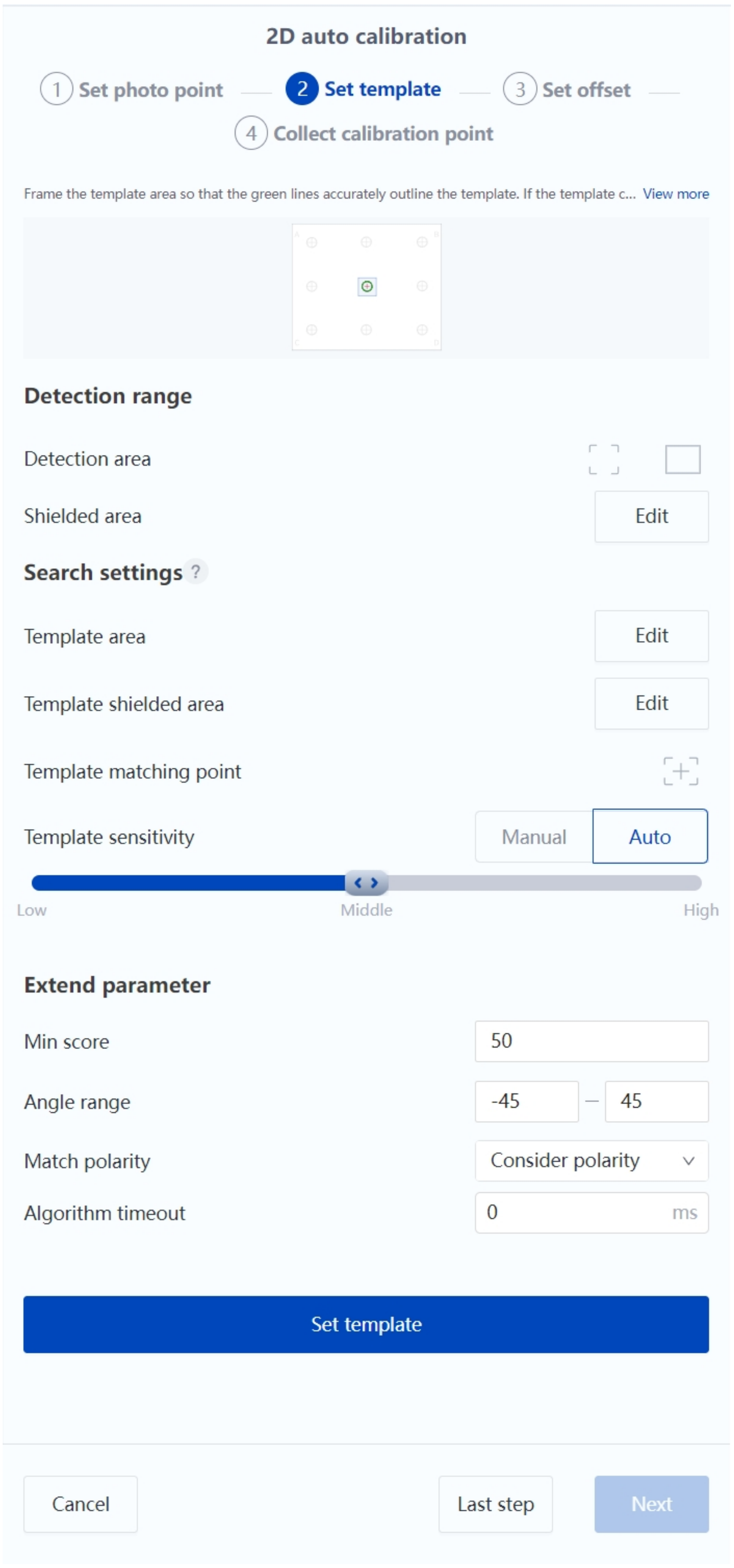
Place 2D calibration board

1) Place the 2D calibration board on the work plane, making sure that the calibration board is in the center of the image vision field (remove the coverings if the eight circles were covered in previous 2D automatic calibration).
2) Adjust the camera parameters to make the image clear, and click **Obtain photo point**.

NOTE

During 2D calibration, if you find that the image taken by the camera is not clear, you should stop the calibration and adjust the camera parameters to make the image clear, and then recalibrate. After calibration is completed, the focus parameters can no longer be adjusted.

② Set template


2D auto calibration

1 Set photo point — 2 Set template — 3 Set offset — 4 Collect calibration point

Frame the template area so that the green lines accurately outline the template. If the template c... View more

Detection range

Detection area

Shielded area — Edit

Search settings ?

Template area — Edit

Template shielded area — Edit

Template matching point

Template sensitivity: Manual [✓ Auto]

Low — Middle — High

Extend parameter

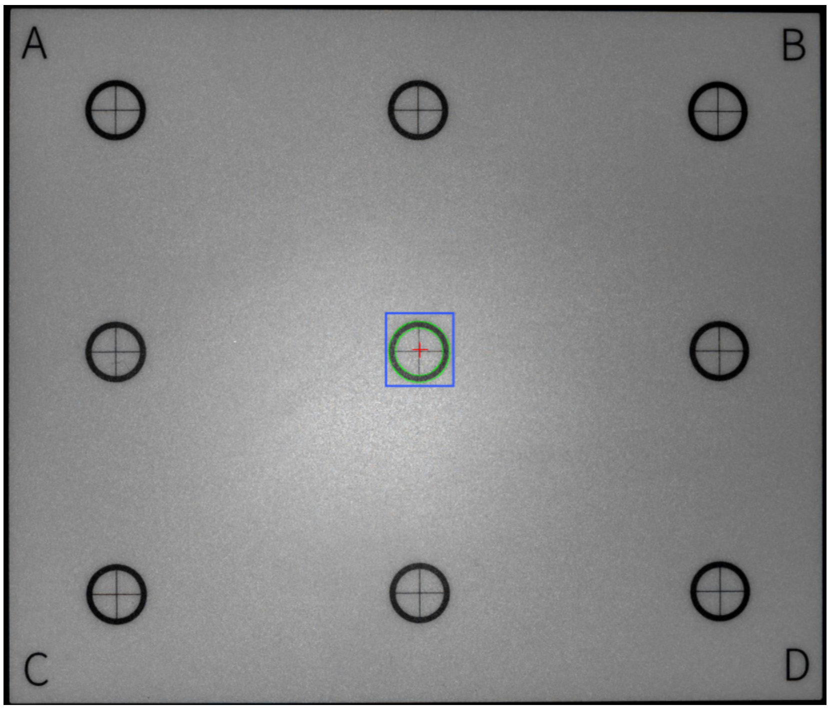
Min score	50
Angle range	-45 — 45
Match polarity	Consider polarity
Algorithm timeout	0 ms

Set template

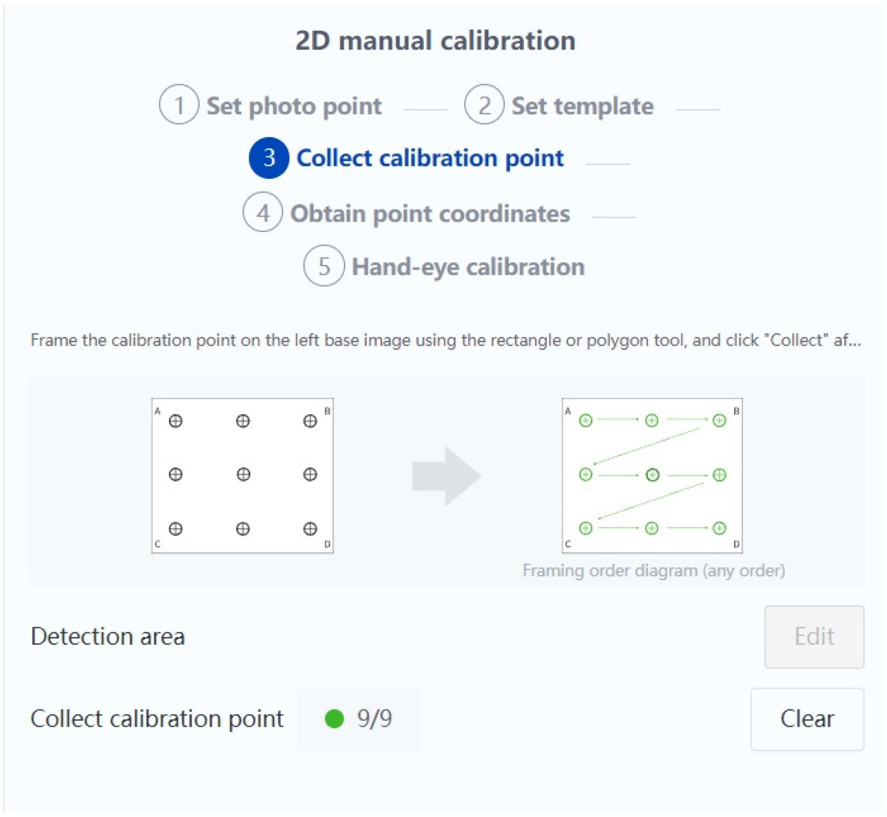
Cancel Last step Next

Make sure the calibration board is in the center of the screen. Click on the right of **Template area**, frame any circle on the calibration board using the rectangle tool. Then click on the right of **Template matching point**, and click the intersection of the cross in the center of the circle. For more details on advanced settings of templates, refer to the instructions for Template matching.

If the green line in the image accurately outlines the template, click **Set template**. If not, adjust the template parameters and re-recognize it.



③ Collect calibration point


2D manual calibration
1 Set photo point
2 Set template
3 Collect calibration point
4 Obtain point coordinates
5 Hand-eye calibration

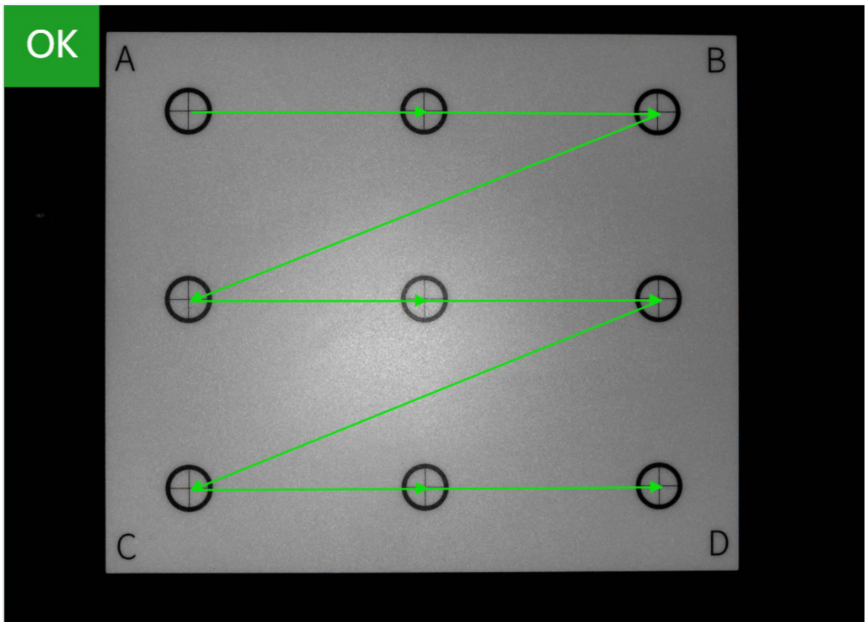
Frame the calibration point on the left base image using the rectangle or polygon tool, and click "Collect" af...

Framing order diagram (any order)

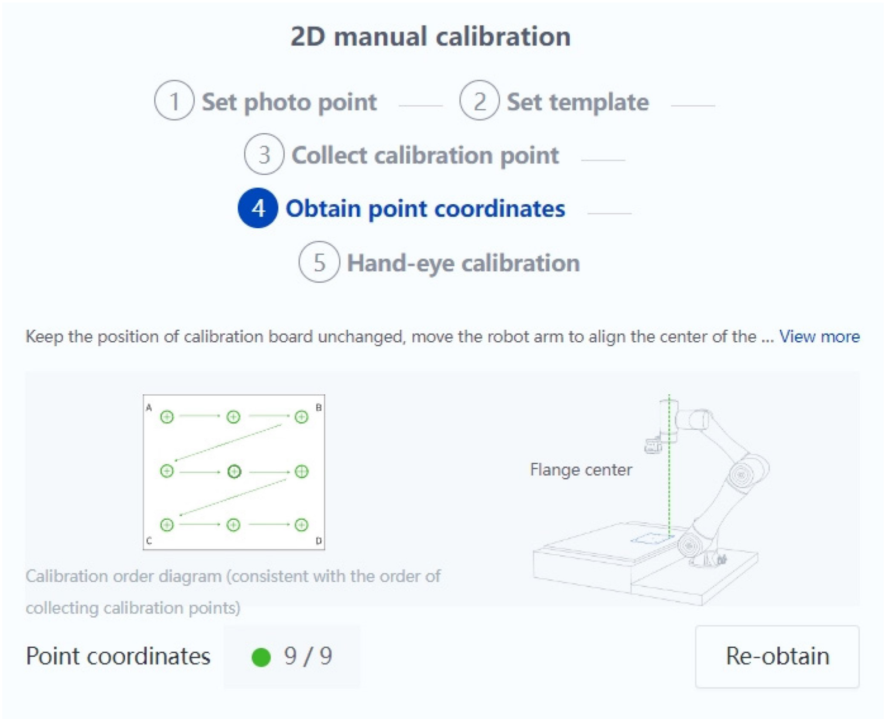
Detection area Edit

Collect calibration point ● 9/9 Clear

1) Draw a rectangle in the base image on the left, frame a single circle, and click **Collect**. The system will recognize the calibration points based on the template.
2) Repeat the steps above to complete the collection of 9 calibration points.



④ Obtain point coordinates



2D manual calibration

Keep the position of calibration board unchanged, move the robot arm to align the center of the ... View more

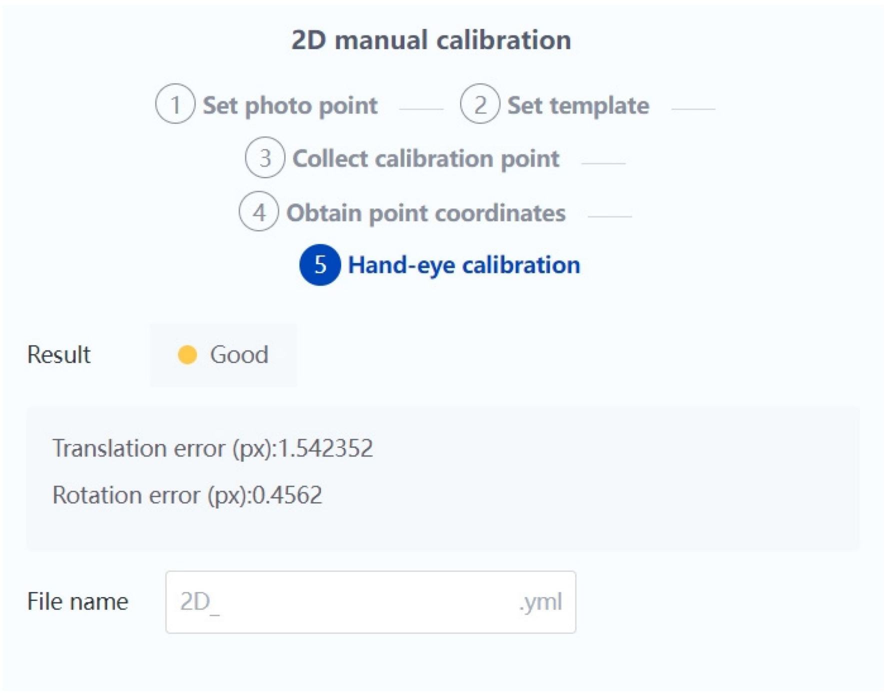
Calibration order diagram (consistent with the order of collecting calibration points)

Point coordinates ● 9 / 9

1) Keep the position of calibration board the same as the base image, move the robot arm to align the center of the robot flange with the first calibration point (which can be aligned using calibration needles or other tools), and click **Obtain** to obtain the point.
2) Obtain the coordinates of nine points one by one in the order in which the calibration points were collected.

⑤ Hand-eye calibration

Decide whether to complete the calibration according to the calibration result and site requirement for accuracy (file name can only contain English letters, numbers and underscores).



Result ● Good

Translation error (px):1.542352

Rotation error (px):0.4562

File name 2D_ .yml

- When the calibration result is **Excellent**, enter the file name (can only contain English letters, numbers and underscores) and save.
- When the calibration result is **Good**, you can view the error data and decide whether to recalibrate or save the calibration file directly according to the site requirement for calibration accuracy.
- When the calibration result is **Failed**, you need to set and calibrate again.

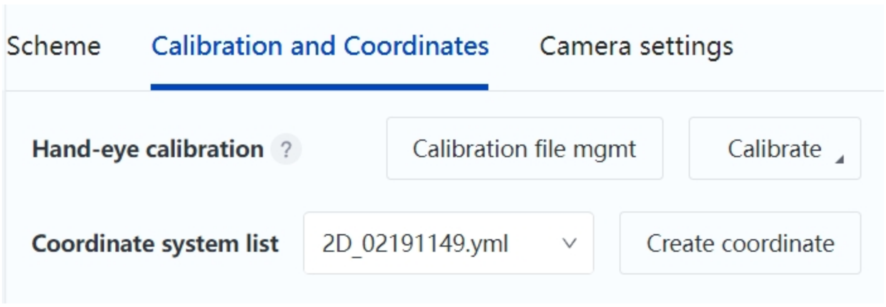
After saving the file, the interface returns to the **Calibration and Coordinates** page. As the calibration file is bound to the robot, you need to recalibrate when the robot is changed.

6.3.2 Common causes of calibration failure

If the calibration error is large (greater than 3 pixels) and causes failure, follow the steps below to troubleshoot.

1. Check if the camera and robot arm are firmly installed.
2. Check if the calibration board is fixed relative to the position of the robot arm base.
3. Check if the robot arm is calibrated using a positive (J3>0) posture, otherwise there may be large errors.
4. Check if the calibration is not completed according to the requirements of the operation guide (e.g. the calibration board needs to be placed on a horizontal plane, the camera lens needs to be kept parallel to the work plane, etc.).
5. If the calibration still fails, it may be due to poor accuracy of the robot arm body, and the robot arm needs to be replaced or recalibrated.

6.3.3 Creating coordinate system based on 2D template

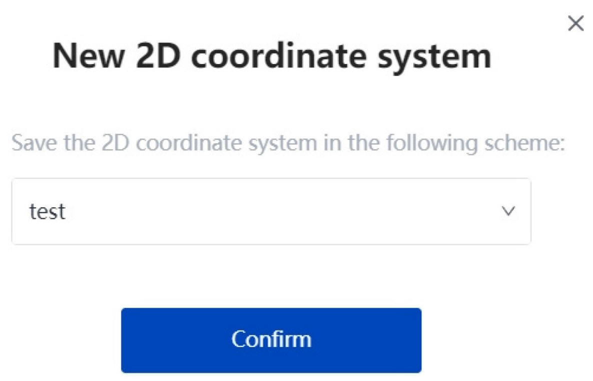


Coordinate system list 2D_02191149.yml

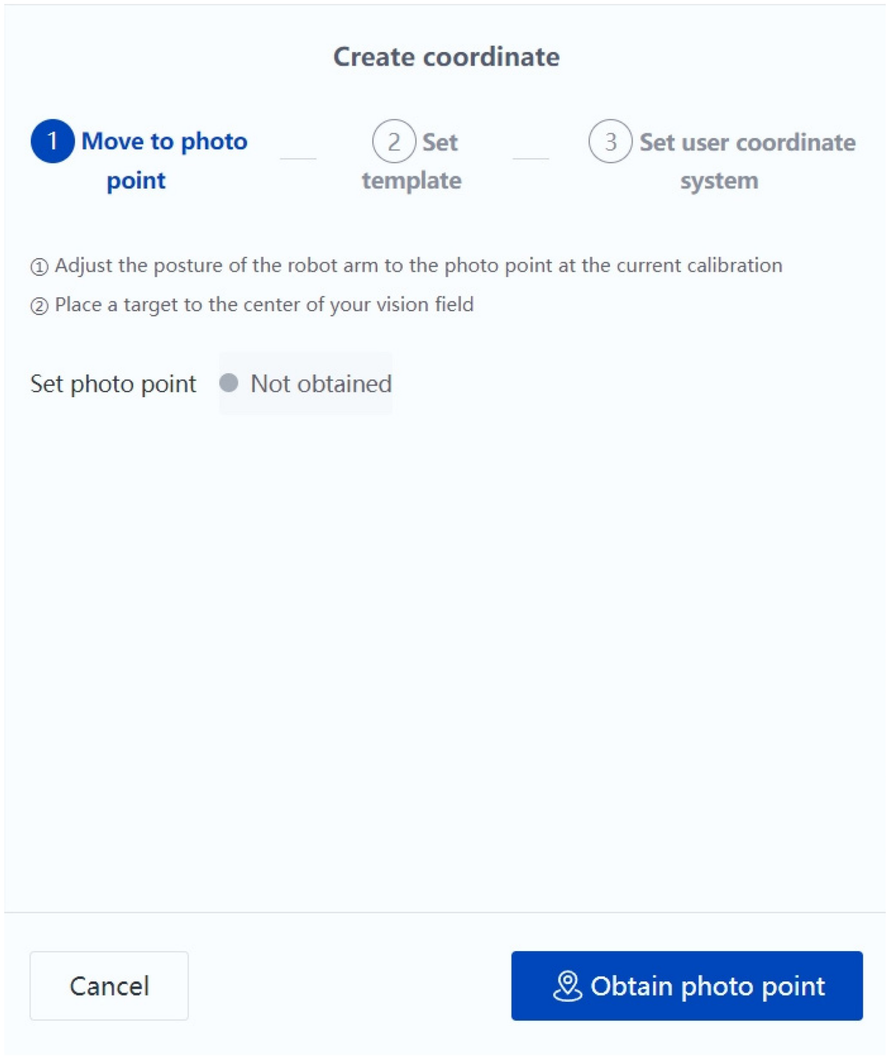
Before creating a coordinate system, you need to select a saved 2 D calibration file. Then click

Create coordinate system.

To establish a 2D coordinate system, you need to select the scheme to which the coordinate system belongs. You can use the 2D positioning tool only if the scheme is associated with a 2D coordinate system.



① **Move to photo point**



① Adjust the posture of the robot arm to the photo point at the current calibration

② Place a target to the center of your vision field

Set photo point ● Not obtained

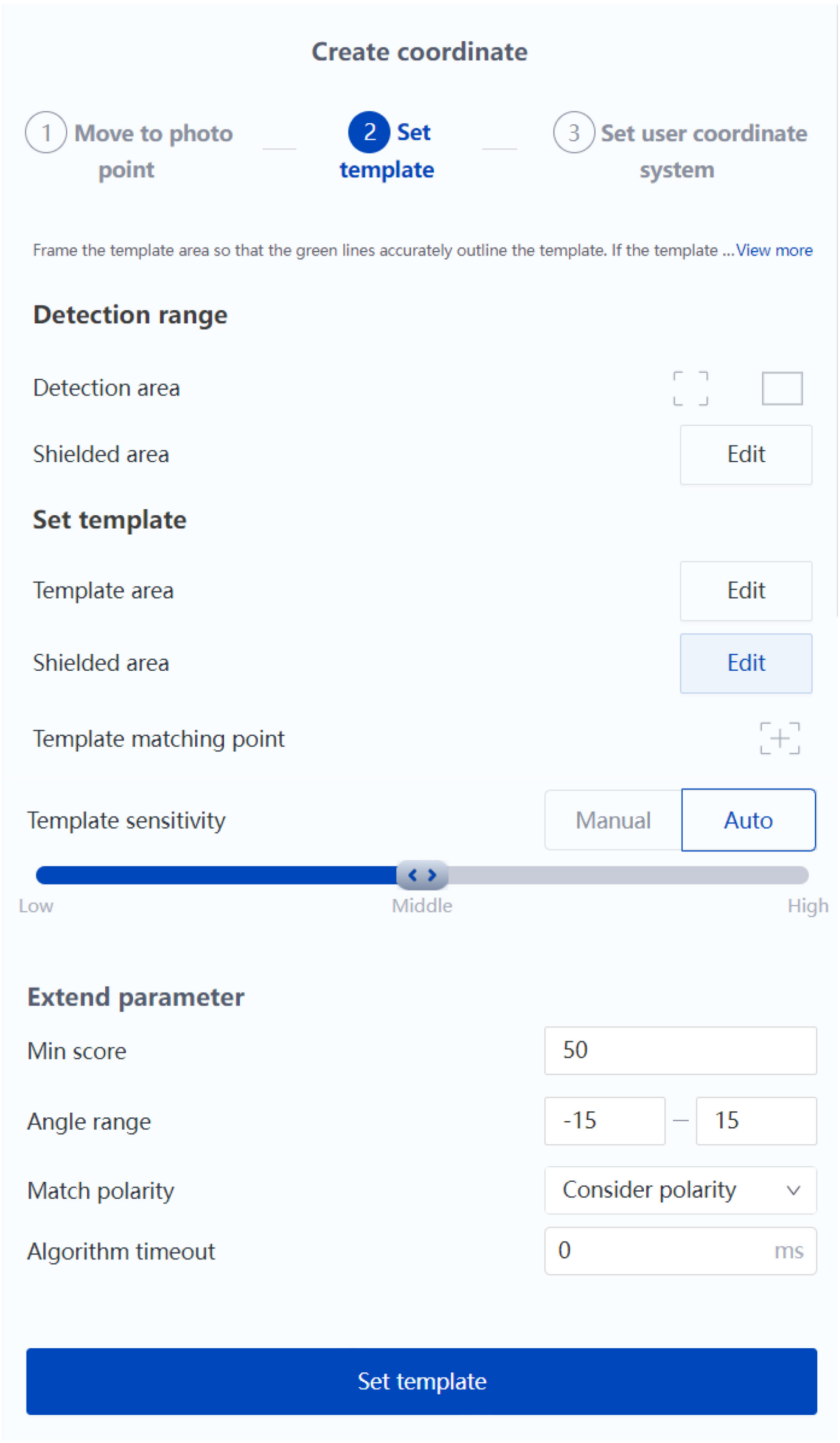
1) Move the robot arm to the photo point during 2D calibration.
2) Place the marker to be used as the origin of the coordinate system (can be the object to be positioned) to the center of the vision field, and it is clear and complete. Then click **Obtain**

photo point.

NOTE

Set the user coordinate system to 0 in Jog panel before taking a photo.

② **Set template**



Create coordinate

1 Move to photo point — 2 Set template — 3 Set user coordinate system

Frame the template area so that the green lines accurately outline the template. If the template ...View more

Detection range

Detection area	
Shielded area	Edit

Set template

Template area	Edit
Shielded area	Edit
Template matching point	
Template sensitivity	Manual [✓ Auto]

Low — Middle — High

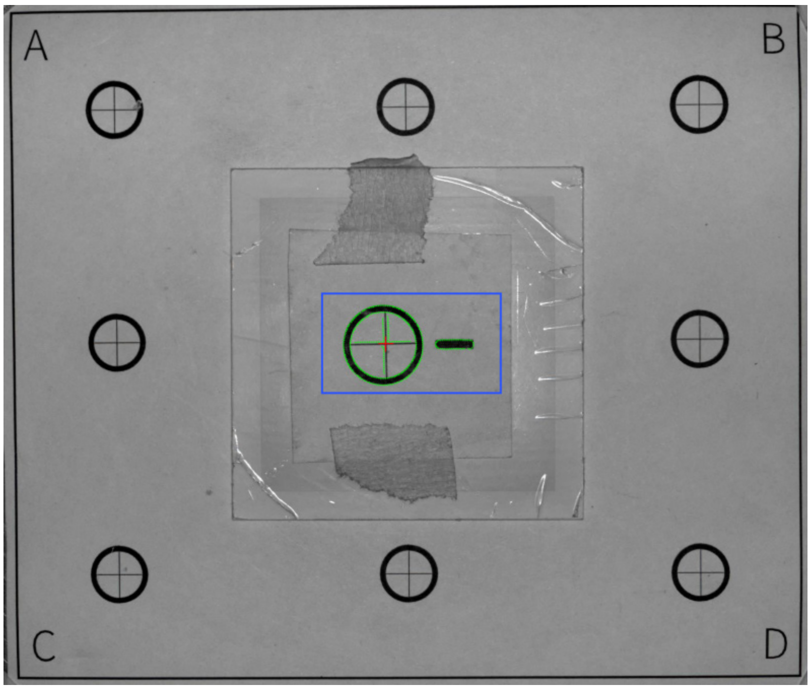
Extend parameter

Min score	50
Angle range	-15 — 15
Match polarity	Consider polarity
Algorithm timeout	0 ms

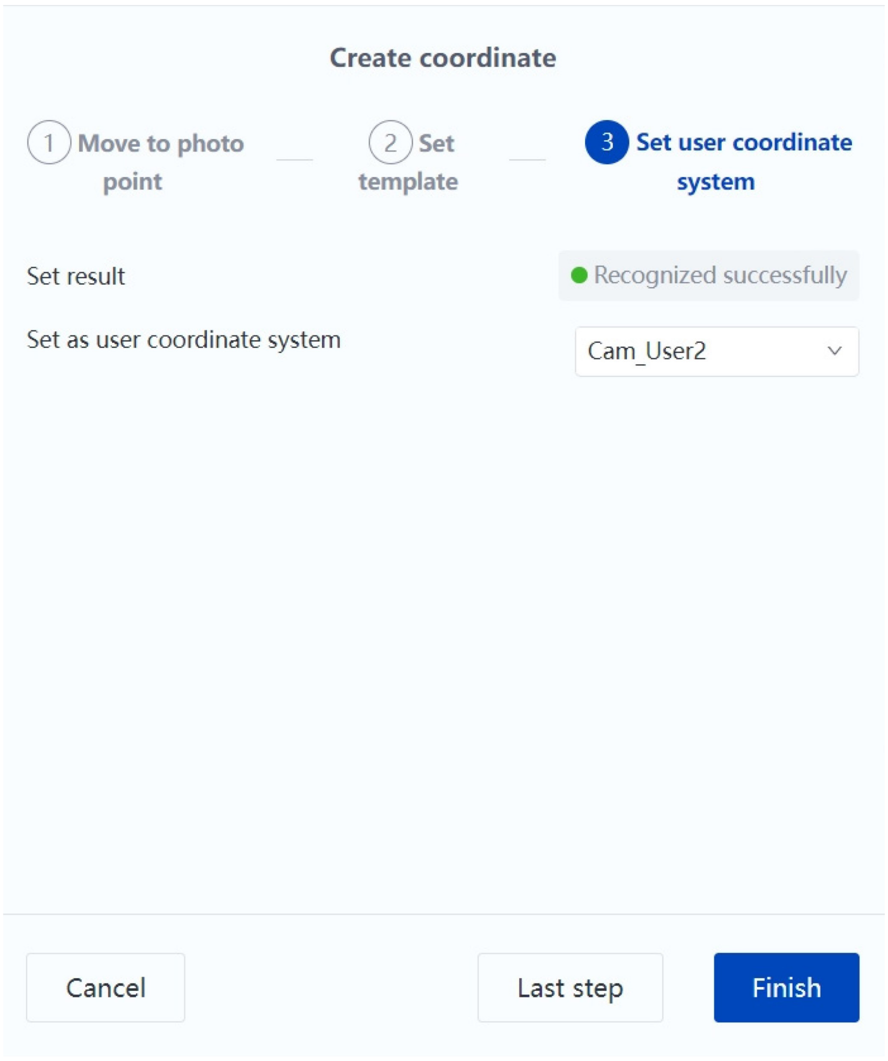
Set template

Make sure the calibration board is in the center of the screen. Click **Edit** on the right of **Template area**, frame the circle in the center of the calibration board using the rectangle tool. Then click on the right of **Template matching point**, and click the intersection of the cross in the center of the circle. For more details on advanced settings of templates, refer to the instructions for Template matching.

If the green line in the image accurately outlines the template, click **Set template**. If not, adjust the template parameters and re-recognize it.



③ Set user coordinate system



Set result: Recognized successfully

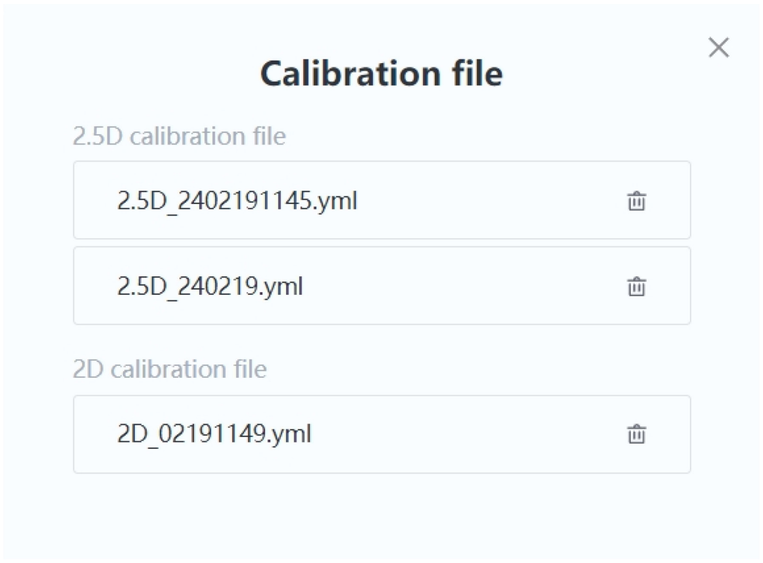
Set as user coordinate system: Cam_User2

After successful recognition, select the name of the user coordinate system that you want the template to correspond to, and click **Finish** to save it. If the coordinate system fails to be recognized, check whether the image is clear and whether the template has been set.

After you save the coordinate system, the interface returns to the **Calibration and Coordinates** page and displays the created coordinate system.

6.4 Calibration file management

You can click **Calibration file mgmt** to view all saved calibration files, and click 🗑 to delete the specified calibration file. The name of the calibration file cannot be modified. If you modify the name of the calibration file by directly modifying the controller file, it may cause an exception in SmartCamera plugin.


Calibration file

2.5D calibration file

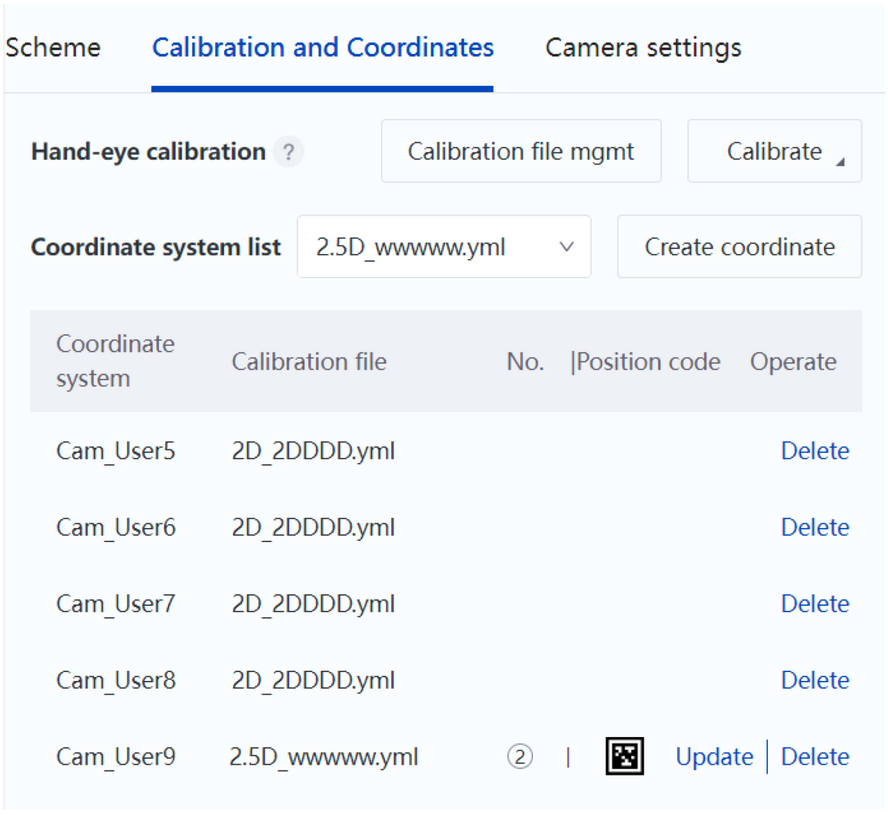
2.5D_2402191145.yml

2.5D_240219.yml

2D calibration file

2D_02191149.yml

6.5 Coordinate system management



Coordinate system	Calibration file	No.	Position code	Operate
Cam_User5	2D_2DDDD.yml			Delete
Cam_User6	2D_2DDDD.yml			Delete
Cam_User7	2D_2DDDD.yml			Delete
Cam_User8	2D_2DDDD.yml			Delete
Cam_User9	2.5D_wwwww.yml	②		Update | Delete

- When the calibration file is a 2.5D calibration file, you will see **Update** in the **Operate** column of the created coordinate system, which is used to update the 2.5D coordinate system directly from this page. Adjust the robot arm position and camera parameters referring to the operations of creating the coordinate system to make the 2.5D positioning code clearly visible. Then click **Update** to directly update the 2.5D coordinate system.
- When the calibration file is a 2D calibration file, updating the coordinate system is not supported. To update the 2D coordinate system, you need to create a new coordinate system and overwrite the coordinate system you want to update in the last step.

Click **Delete** in the **Operate** column of the created coordinate system to delete the corresponding coordinate system.

NOTICE

Do not modify or clear the camera coordinate system (Cam_User) in the **User coordinate system** page of DobotStudio Pro, otherwise it may cause the camera project to run abnormally.

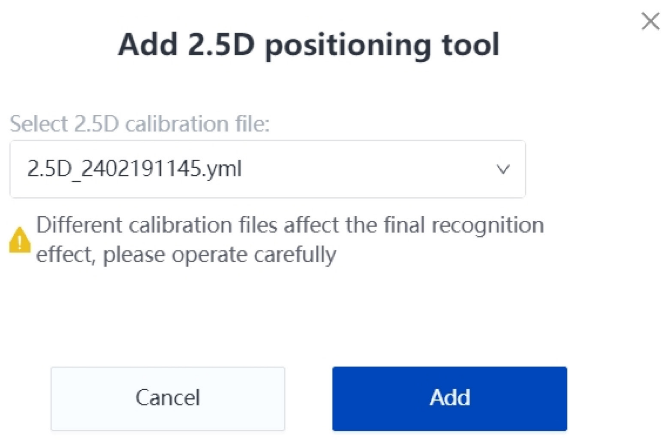
7. Vision tool

7.1 Positioning tool

7.1.1 2.5D positioning

When you select a 2.5D calibration file and create a 2.5D coordinate system, you can update the 2.5D user coordinate system using this tool together with programming.

When adding a 2.5D positioning tool, you need to select the 2.5D calibration file used for positioning.


Add 2.5D positioning tool

Select 2.5D calibration file: 2.5D_2402191145.yml

Different calibration files affect the final recognition effect, please operate carefully

Output result: module status (OK/NG), coordinate system.

7.1.2 2D positioning

When you select a 2D calibration file and create a 2D coordinate system, you can update the 2D user coordinate system using this tool together with programming.

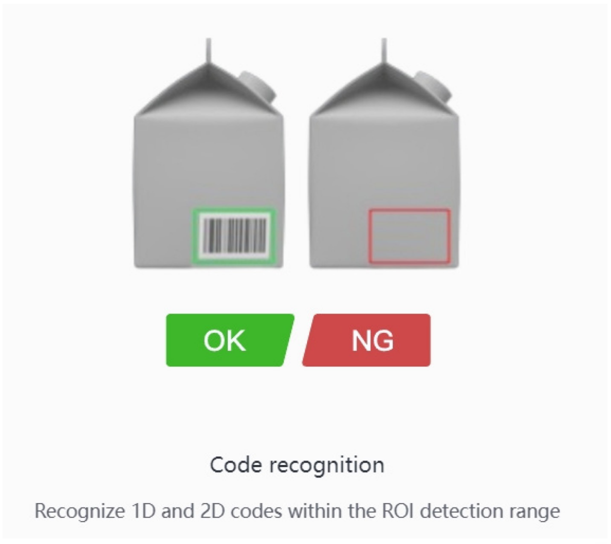
The tool is bound to the 2D coordinate system one to one. You can directly use the tool after add it, with no need to set specific parameters.

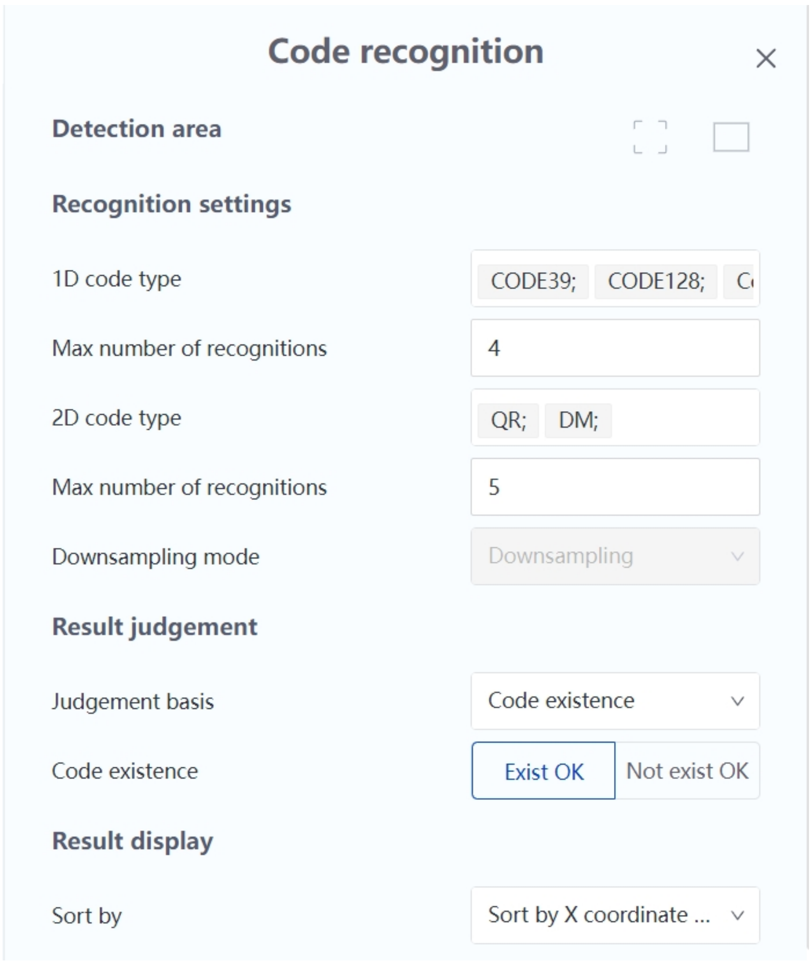
Output result: module status (OK/NG), coordinate system.

7.2 Recognition tool

7.2.1 Code recognition

The code recognition can recognize 1D and 2D codes in the detection area.





Code recognition

Detection area

Recognition settings

1D code type	CODE39; CODE128; C
Max number of recognitions	4
2D code type	QR; DM;
Max number of recognitions	5
Downsampling mode	Downsampling

Result judgement

Judgement basis	Code existence
Code existence	[✓ Exist OK] Not exist OK

Result display

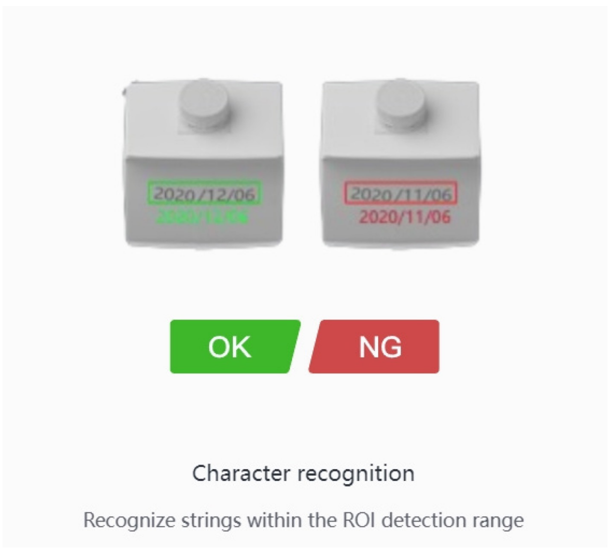
Sort by	Sort by X coordinate ...

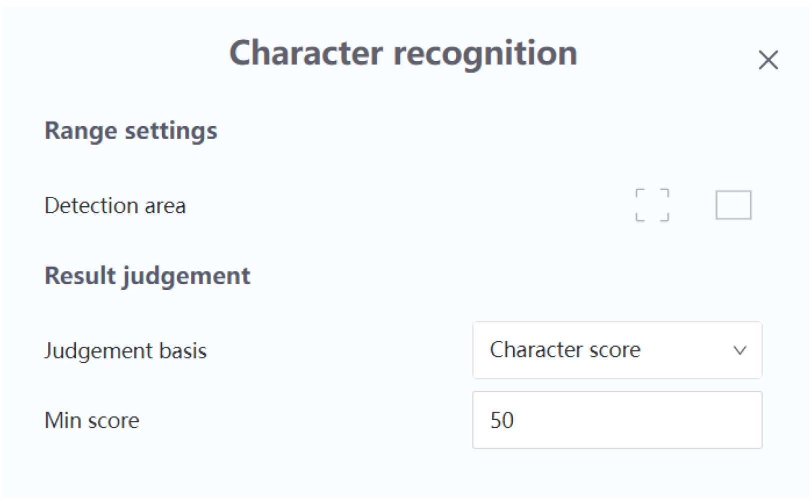
1. Set the detection area according to your actual need, with full screen by default. You can also select the detection area with the rectangle tool.
2. Set the code type to be recognized (can select multiple) and the maximum number of codes to be recognized each time (range: 1 – 200).
   - 1D code types: CODE39, CODE128, Codabar, EAN8, EAN13, UPCA, UPCE, ITF25 and CODE93 (all selected by default).
   - 2D code types: QR and DM (all selected by default).
3. Set the basis for judging the tool result.
   - When the basis is set to **Code existence**, **Exist OK** means that if at least one code is recognized, it is OK. **Not exist OK** means that if no code is recognized, it is OK.
   - When the basis is set to **Min code score**, you need to set the minimum score. When the actual score is higher than the minimum score, it is OK, otherwise it is NG.
   - When the basis is set to **Code count**, you need to set the count range. If the count of the recognized codes is within the range, it is OK, otherwise it is NG.
4. Set the sorting method for the output.

Output result: Module status (OK/NG), code information, code count, code type, code center pixel point X, code center pixel point Y.

7.2.2 Character recognition

The character recognition can recognize 1D and 2D codes in the detection area.





Character recognition

Range settings

Detection area

Result judgement

Judgement basis: Character score

Min score: 50

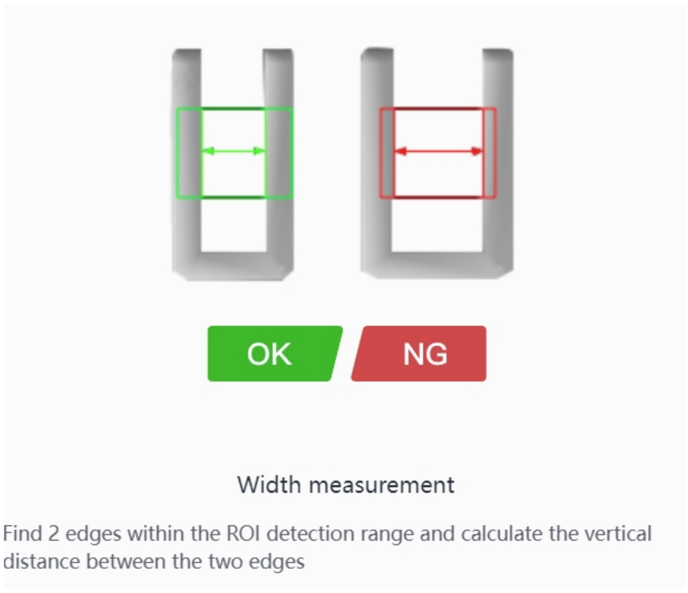
1. Set the detection area according to your actual need, with full screen by default. You can also select the detection area with the rectangle tool.
2. Set the basis for judging the tool result.
   - When the basis is set to **Character score**, you need to set the minimum score. If the actual score is higher than the minimum score, it is OK, otherwise it is NG.
   - When the basis is set to **Character count**, you need to set the count range of characters. If the recognized character is within the range, it is OK, otherwise it is NG.
   - When the basis is set to **Base character**, you need to set the standard character content. If the standard character content is recognized, it is OK, otherwise it is NG.

Output result: Module status (OK/NG), character information.

7.3 Measuring tool

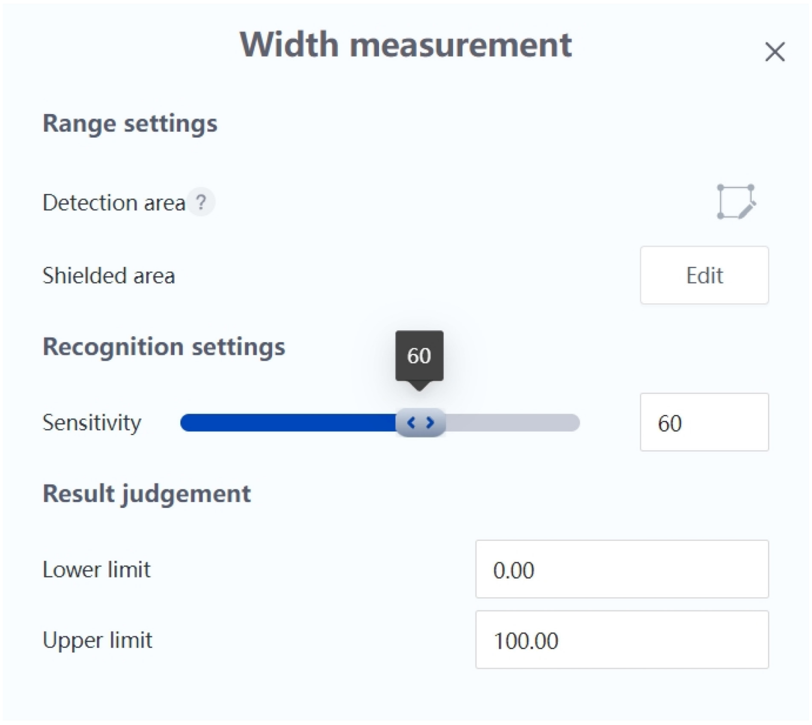
7.3.1 Width measurement

Width measure can find two edges in the detection area and calculate the vertical pixel distance of the two edges.



Width measurement

Find 2 edges within the ROI detection range and calculate the vertical distance between the two edges



Width measurement

Range settings

Detection area ?

Shielded area — Edit

Recognition settings

Sensitivity: 60

Result judgement

Lower limit: 0.00

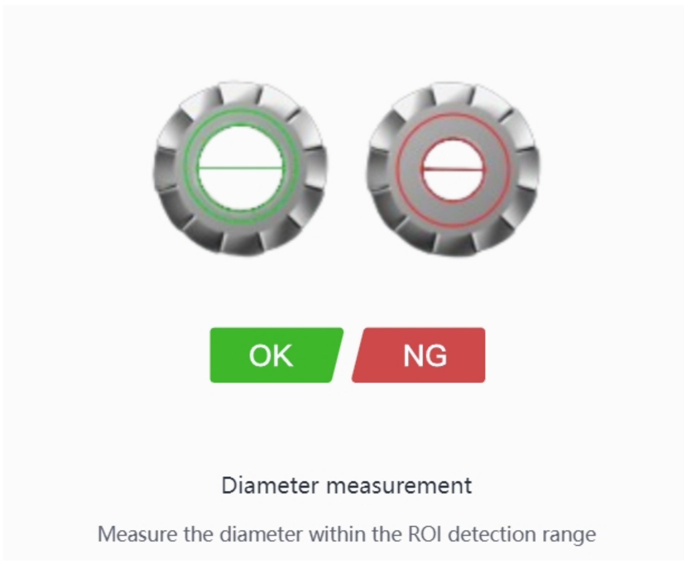
Upper limit: 100.00

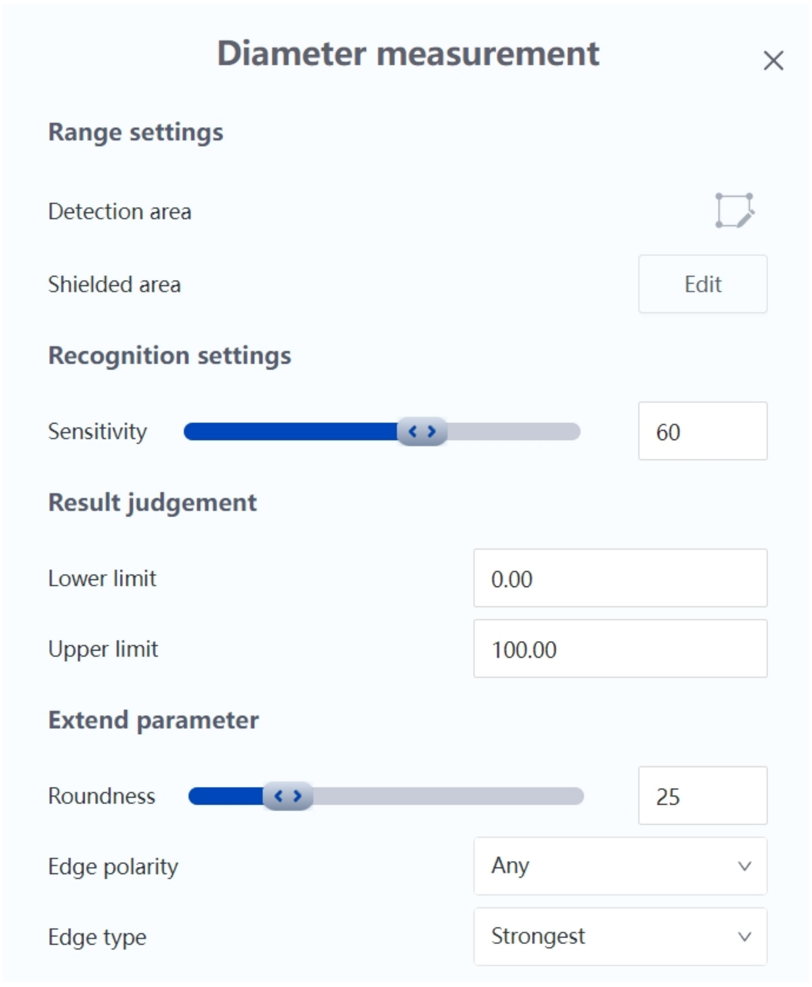
1. Draw two edges of which the width is to be measured within the detection area through .
2. If there is a part of the detection area that needs to be shielded, you can click **Edit** and draw the shielded area using the polygon tool.
3. Set the sensitivity of the width detection according to your need.
4. Set the OK range of the tool result. If the measured pixel width is within the range, it is OK, otherwise it is NG.

Output result: Module status (OK/NG), measurement result (pixel value).

7.3.2 Diameter measurement

Diameter measure can find circles in the detection area and calculate the pixel diameter of the circle.




Diameter measurement

Range settings

Detection area	
Shielded area	Edit

Recognition settings

Sensitivity	60

Result judgement

Lower limit	0.00
Upper limit	100.00

Extend parameter

Roundness	25
Edge polarity	Any
Edge type	Strongest

1. Use the tool , click the center of the circle to be measured to generate a fixed-size circular detection area. You can adjust the size by yourself. If the clicked area is too close to the edge of the image, the circular detection area will exceed the image area and the drawing will fail.
2. If there is a part of the detection area that needs to be shielded, you can click **Edit** and draw the shielded area using the polygon tool.
3. Set the sensitivity of circle detection according to your need.

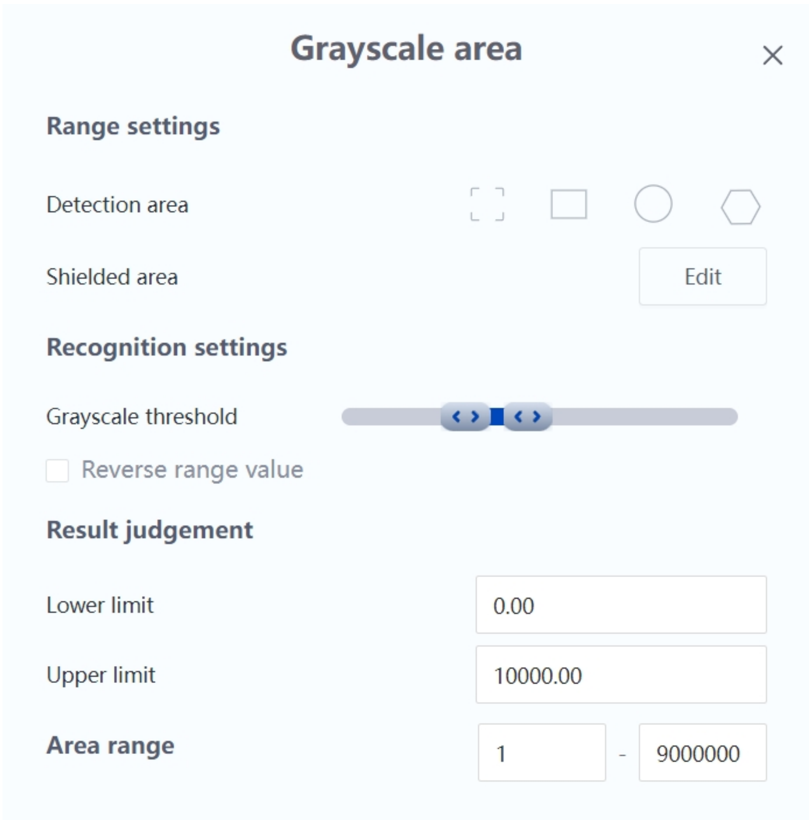
4. Set the OK range of the tool result. If the measured pixel diameter is within the range, it is OK, otherwise it is NG.
5. If you find that the results of circle detection are not satisfactory after running, you can set the extended parameters.
   - Roundness: Minimum ratio of the points involved in the circle fitting to the total number of points. The higher the value, the closer the found circle is to a regular circle. If the roundness of the detected circle exceeds this value, it is judged as a circle, otherwise it is not considered to be a circle.
   - Edge polarity: Specify the colour transition of the detected edge. Edge polarity is related to the direction, with the direction determined by the arrow direction of the ROI area.
     - Black to White: Transition from an area with lower grayscale to the edge of an area with higher grayscale.
     - White to Black: Transition from an area with higher grayscale to the edge of an area with lower grayscale.
     - Any: Both of the edges above are detected.
   - Edge type: Specify the detection type of edge.
     - Strongest: only the set of edge points with the largest gradient threshold within the range is detected and fitted to the circle.
     - Maximum: only the set of edge points with the greatest distance away from the circle center within the range are detected and fitted to the circle.
     - Minimum: only the set of edge points with the smallest distance away from the circle center within the range are detected and fitted to the circle.

Output result: Module status (OK/NG), measurement result (pixel value).

7.3.3 Grayscale area

Grayscale area can calculate the pixel area that meets the grayscale range within the detection area.

OK
NG
Grayscale area
Calculate the pixel value that meets the grayscale requirement within the ROI detection range


Grayscale area

Range settings

Detection area

Shielded area Edit

Recognition settings

Grayscale threshold

Reverse range value

Result judgement

Lower limit	0.00
Upper limit	10000.00
Area range	1 - 9000000

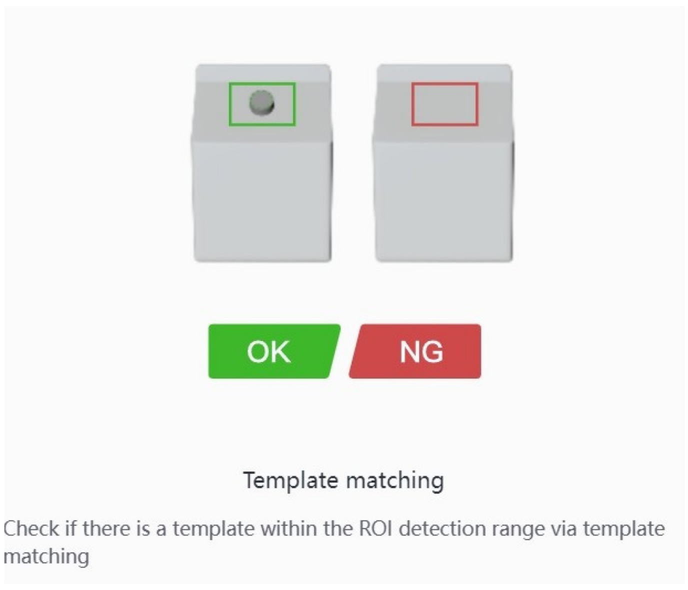
1. Set the detection area according to your actual need, with full screen by default. You can also select the detection area through the rectangle/circle/polygon tool.
2. If there is a part of the detection area that needs to be shielded, you can click **Edit** and draw the shielded area using the polygon tool.
3. Adjust the grayscale threshold to be recognized according to your actual detection need. The system defaults to take the range between the two values. If you want to adopt the range at both ends, you can switch on **Reverse range value** function.
4. Set the OK range of the tool result. If the measured average pixel area is within the range, it is OK, otherwise it is NG.
5. Set the grayscale area range, pixel blocks that are not within the range will be discarded.

Output result: Module status (OK/NG), measurement result (pixel value).

7.4 Tool existence

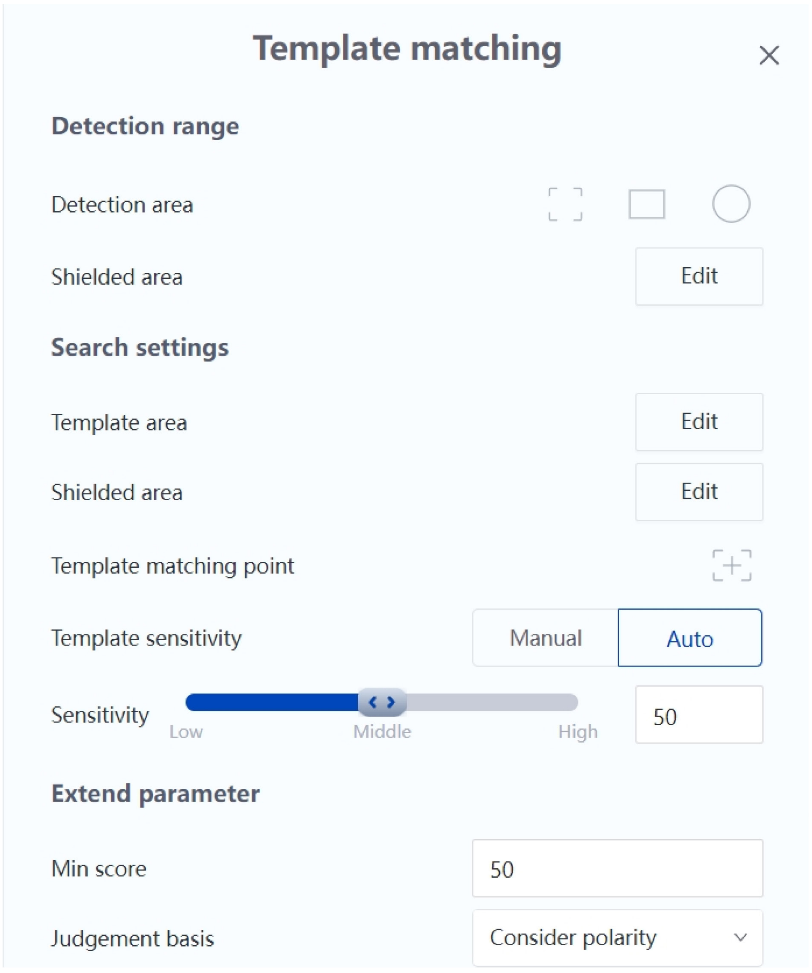
7.4.1 Template matching

The template match is used to detect the specific figure in an image and obtain the image coordinates of its template match point.



Template matching

Check if there is a template within the ROI detection range via template matching


Template matching

Detection range

Detection area

Shielded area: Edit

Search settings

Template area: Edit

Shielded area: Edit

Template matching point

Template sensitivity: Manual [✓ Auto]

Sensitivity: Low / Middle / High — 50

Extend parameter

Min score: 50

Judgement basis: Consider polarity

1. Set the detection area according to your actual need, with full screen by default. You can also select the detection area with the rectangle or polygon tool.
2. If there is a part of the detection area that needs to be shielded, you can select the tool and draw the shielded area within the detection area.
3. Set the template area according to your actual need. You can set the match point manually after selecting the area using the rectangle/polygon tool.
4. If there is a part of the template area that needs to be shielded, you can select the tool

and draw the shielded area within the template area.

5. Adjust the template sensitivity according to your actual need, which supports automatic and manual adjustment.
   - For automatic mode, you need to set the sensitivity level.

     The higher the sensitivity, the more features will be identified in the template area and the more accurate the match position will be. However, high sensitivity may cause noise to be misidentified as features, so it is recommended to set it according to your actual need.
   - For manual mode, you need to set the scale and grayscale threshold.

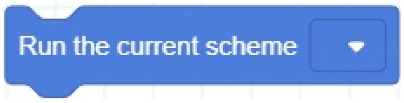
     The scale value divided by 10 is the downsampling coefficient. Assuming the parameter is set to 50, the downsampling coefficient corresponds to 5, i.e. take 1 pixel every 5 pixels in each row and column of the original image to make up an image. The smaller the value, the longer the detection time; the larger the value, the less detailed the image contour, which may affect the detection results. As a result, you need to balance the settings according to your actual need.

     The grayscale threshold can set the range of grayscale value for the detected area. The area that meets the requirement will be recognized.
6. Set the minimum score. The minimum score indicates the degree of similarity between the feature template and the target in the image, i.e. the similarity threshold (range: 0 - 1, with 1 indicating a perfect match). Only when the similarity of the target to be searched for reaches this threshold can it be searched.
7. Set whether to match polarity. Polarity indicates the colour transition from the feature pattern to the background. When the edge polarity of the target does not match the polarity of the feature template, you need to set it to **Not consider polarity** if it is necessary to ensure that the target is found. If it is not necessary to ensure that the target is found, you can set it to **Consider polarity** to reduce the searching time.
8. Set the angle range. If the target of the set template changes angle which is within the angle range, it can still be recognized. If it is not within the angle range, it cannot be recognized.
9. Set the algorithm timeout. If the actual algorithm timeout exceeds this value, the algorithm will stop detecting and output NG when it reaches the set timeout. If this parameter is set to 0, the algorithm timeout function is not enabled and the algorithm detection time is not limited.

Output result: Matching status (OK/NG), matching pixel point X, matching pixel point Y, and angle R.

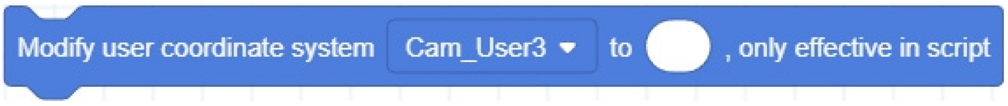
8. Blockly/Script programming

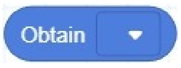
8.1 Block description

1. 

Description: Take photos and run the tool, and save the output.

Parameter: If the scheme is running at the time of programming, the drop-down box will show the name of the current scheme, otherwise the drop-down box will be blank. It is for viewing only, cannot be modified.

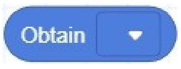
2. 

Description: It is used with the 

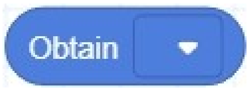
block. Modify the specified coordinate system to the coordinate system output by the 2.5D positioning/2D positioning tool. The modification is valid only when the project is running, and the modified coordinate system changes back to the original one after the project stops.

Parameter:

1) The user coordinate system to be modified, only supports the user coordinate system created by the SmartCamera plugin.

2) The modified coordinate system value, obtained by 

block.

3. 

Description: Obtain the output of the current running scheme.

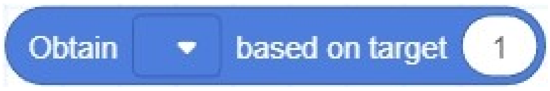
Parameter: Select the configured output result of the current running scheme from the drop-down list.

Return: Specified output result.

- The positioning tool will report an error and stop the project if it cannot find the coordinates, and other tools will return "nil" if they cannot obtain the specified results.
- The module status is "OK" or "NG", the coordinate format is {X, Y, Z, RX, RY, RZ}, and the other outputs return the corresponding values or strings.

NOTE

If the recognized code information/character information contains line breaks (/r or /n) or semicolons (;), the returned result will fail to be parsed.

4. 

Description: Obtain the output of the current running scheme. For recognition tools that may recognize more than one target, you can specify to obtain the output result of target X through this block.

Parameter:

1) Select the configured output result of the current running scheme from the drop-down list.

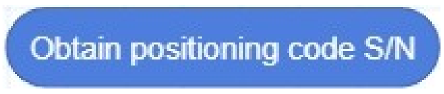
2) Enter the number of the target. For tools that can only output one result, this parameter is invalid and the first result is always output.

Return: Specified output result based on the specified target.

- The positioning tool will report an error and stop the project if it cannot find the coordinates, and other tools will return “nil” if they cannot obtain the specified results.
- The module status is “OK” or “NG”, the coordinate format is {X, Y, Z, RX, RY, RZ}, and the other outputs return the corresponding values or strings.

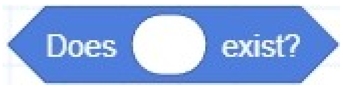
NOTE

If the recognized code information/character information contains line breaks (/r or /n) or semicolons (;), the returned result will fail to be parsed.

5. 
Obtain positioning code S/N

Description: Obtain the 2.5D positioning code serial No. recognized by the 2.5D positioning tool in the current running scheme.

Return: The recognized 2.5D positioning code serial No. It returns void if 2.5D positioning code is not recognized or no 2.5D positioning tool is in the scheme.

6. 
Does exist?

Description: Used with other oval blocks, it can be used to determine whether the result to be obtained exists or not.

Return: It returns **true** if the value of the oval block is not null, otherwise it returns **false**.

8.2 Script commands

1. **Command**: RunVX500Project(SolutionName)

   Description: Run the current scheme.

   Parameter:

   SolutionName: The name of the currently running scheme in the plugin. Entering another name is invalid.

   Example:

   RunVX500Project("2.5D_test")

2. **Command**: GetVX500ModelRes(moduleName, moduleId, type, index)

   Description: Obtain the output of the current running scheme.

   Parameter:

   - Model: Tool name. Range:
     - "macapriltag": 2.5D positioning
     - "matchlocate": 2D positioning
     - "idemodule": Code recognition
     - "dlocrdetect": Character recognition
     - "widthmeasure": Width measurement
     - "diametermeasure": Diameter measurement
     - "greyarea": Grayscale area

  - "imageexist": Template matching
- moduleId: Tool ID, i.e. the number after the tool name that has been added in the scheme.
- type: Type of the output result. Range:
  - "state": Module status, “OK” or “NG”
  - "diameter": Diameter
  - "width": Width
  - "matchX": Matching point X
  - "matchY": Matching point Y
  - "matchAng": Angle R
  - "centerX": Code center pixel point X
  - "centerY": Code center pixel point Y
  - "count": Code count
  - "content": Code information / Character information
  - "type": Code type
  - "area": Measurement result of grayscale area
  - "coord": Coordinate, in the format {X,Y,Z,RX,RY,RZ}
- index: Optional parameter. For recognition tools that may recognize more than one target, you can specify to obtain the output result of target X through this parameter. 1 by default.

Return: Specified output result. The positioning tool will report an error and stop the project if it cannot find the coordinates, and other tools will return “nil” (null value in Lua) if they cannot obtain the specified results.

Example:

```
local coord1 = GetVX500ModelRes("macapriltag", 1, "coord")
```

NOTE

If the recognized code information/character information contains line breaks (/r or /n) or semicolons (;), the returned result will fail to be parsed.

3. **Command**: SetVX500CamUser(Cam_User, User_table)

Description: Modify the specified coordinate system. The modification is valid only when the project is running, and the modified coordinate system changes back to the original one after the project stops.

Parameter:

- Cam_User: The name of the coordinate system to be modified.
- User_table: The modified coordinate system, can be obtained through the GetVX500ModelRes command.

Example:

```
local coord1 = GetVX500ModelRes("macapriltag", 1, "coord")
SetVX500CamUser("Cam_User1", coord1)
```

4. **Command**: GetVX500CodeID()

Description: Obtain the 2.5D positioning code serial No. recognized by the 2.5D positioning tool in the current running scheme.

Return: The recognized 2.5D positioning code serial No.

9. Demos

9.1 2.5D positioning and picking

The program below shows an example of 2.5D positioning and picking of a workpiece.

- P6 is the first photo point of the 2.5D positioning code. Ensure that the 2.5D positioning code is clearly visible when photographed, and select User coordinate system 0 when teaching the point.
- Cam_User2 is the 2.5D coordinate system based on the 2.5D positioning code. When the program is running, the robot will update the coordinate system based on the recognized 2.5D positioning code.
- P7 is the second photo point of 2.5D positioning code. It is recommended to use the photo point when establishing the coordinate system and modify the user coordinate system of the point to Cam_User2. If you do not have high requirement for positioning accuracy, you can skip the secondary photography and directly pick up the workpiece after updating the user coordinate system for the first time.

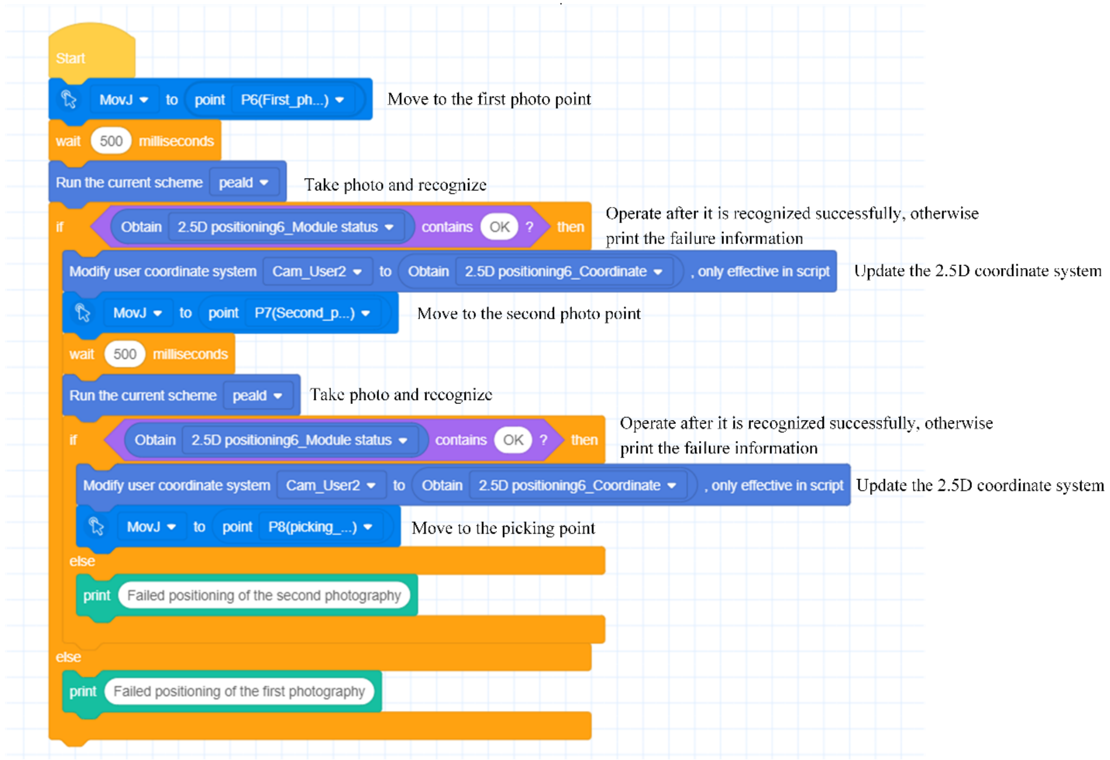
NOTE

The positioning principle of secondary photography: When the camera is far away from the positioning code, the positioning accuracy is poor, but the camera has a large vision field, so it can find the positioning code in a wider range. When the camera is close to the positioning code, the positioning accuracy is higher although the positioning code is easy to go out of the vision field. Therefore, you can set the first photo point to a farther point, update the coordinate system once, and approach the positioning code based on this coordinate system, then take a second photo to update the coordinate system to the one with higher accuracy.

In application, it is recommended that the second photo point be the same as the photo point when creating the 2.5D coordinate system, and the height of the first photo point can be set to be the same as that of the second photo point, so as to ensure that the positioning code is within the camera's vision field.

- P8 is the workpiece picking point. When teaching the point, you need to select Cam_User2 coordinate system and the position of the 2.5D positioning code relative to the workpiece must remain unchanged.

Blockly DEMO:



```
Start
MovJ to point P6(First_ph...)                                      Move to the first photo point
wait 500 milliseconds
Run the current scheme peald                                        Take photo and recognize
if Obtain 2.5D positioning6_Module status contains OK ? then        Operate after it is recognized successfully, otherwise
                                                                    print the failure information
    Modify user coordinate system Cam_User2 to Obtain 2.5D positioning6_Coordinate , only effective in script    Update the 2.5D coordinate system
    MovJ to point P7(Second_p...)                                   Move to the second photo point
    wait 500 milliseconds
    Run the current scheme peald                                    Take photo and recognize
    if Obtain 2.5D positioning6_Module status contains OK ? then    Operate after it is recognized successfully, otherwise
                                                                    print the failure information
        Modify user coordinate system Cam_User2 to Obtain 2.5D positioning6_Coordinate , only effective in script    Update the 2.5D coordinate system
        MovJ to point P8(picking_...)                               Move to the picking point
    else
        print Failed positioning of the second photography
else
    print Failed positioning of the first photography
```

Script DEMO:

```
MovJ(P6) -- Move to the first photo point
Wait(500)
RunVX500Project("peald") -- Take photo and recognize
if (GetVX500ModelRes("macapriltag", 6, "state") == "OK") -- Operate after it is recognized successfully,
otherwise print the failure information
then
    local coord1 = GetVX500ModelRes("macapriltag", 6, "coord")
    SetVX500CamUser("Cam_User2", coord1) -- Update the 2.5D coordinate system
    MovJ(P7) -- Move to the second photo point
    Wait(500)
    RunVX500Project("peald") -- Take photo and recognize
    if (GetVX500ModelRes("macapriltag", 6, "state") == "OK") -- Operate after it is recognized successfully,
otherwise print the failure information
    then
        local coord2 = GetVX500ModelRes("macapriltag", 6, "coord")
        SetVX500CamUser("Cam_User2", coord2) -- Update the 2.5D coordinate system
        MovJ(P8) -- Move to the picking point
    else
        print("Failed positioning of the second photography")
    end
else
    print("Failed positioning of the first photography ")
end
```

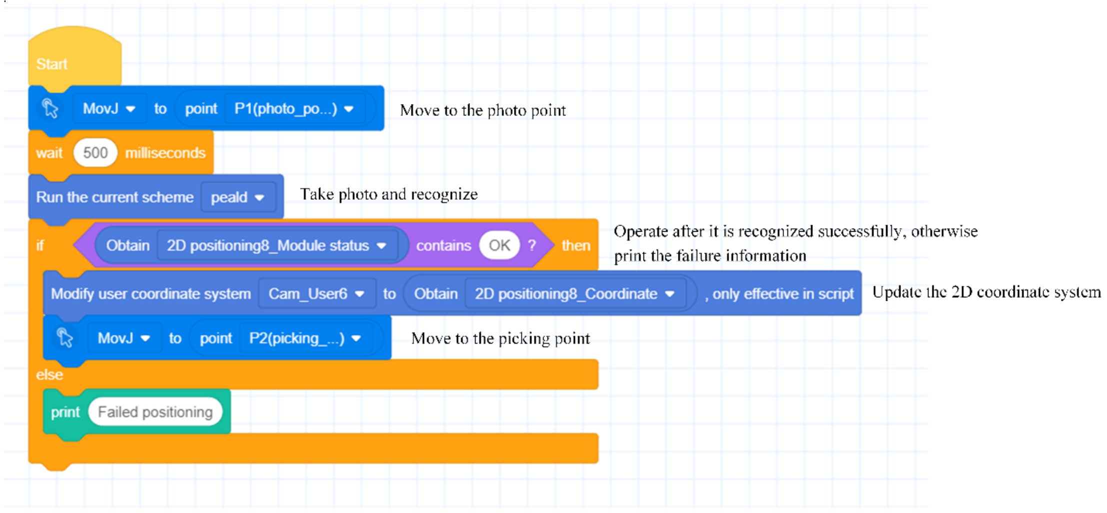
9.2 2D positioning and picking

The program below shows an example of 2D positioning and capture of a workpiece.

- P1 is the 2D photo point, which must be the same as the photo point used in 2D calibration and establishing the 2D coordinate system.

- Cam_User6 is the 2D coordinate system based on the template match. When the program is running, the robot will update the coordinate system based on the matched template (template used when establishing 2D coordinate system).
- P2 is the workpiece picking point. When teaching the point, you need to select Cam_User6 coordinate system. In actual applications you can use the workpiece or other objects with obvious features as the template, as long as the position of the template relative to the workpiece remains unchanged.

Blockly DEMO:


Start
MovJ to point P1(photo_po...) Move to the photo point
wait 500 milliseconds
Run the current scheme peald Take photo and recognize
if Obtain 2D positioning8_Module status contains OK ? then Operate after it is recognized successfully, otherwise print the failure information
Modify user coordinate system Cam_User6 to Obtain 2D positioning8_Coordinate , only effective in script Update the 2D coordinate system
MovJ to point P2(picking_...) Move to the picking point
else
print Failed positioning

Script DEMO:

```
MovJ(P1) -- Move to the photo point
Wait(500)
RunVX500Project("peald") -- Take photo and recognize
if (GetVX500ModelRes("matchlocate", 14, "state") == "OK") -- Operate after it is recognized successfully,
otherwise print the failure information
then
   local coord1 = GetVX500ModelRes("matchlocate", 14, "coord")
   SetVX500CamUser("Cam_User6", coord1) -- Update the 2D coordinate system
   MovJ(P2) -- Move to the picking point
else
   print("Failed positioning")
end
```

Appendix A Technical specifications

Positioning accuracy		
2.5D	Working height: 180 mm – 250 mm (Distance between working position and code: 100 mm / 300 mm): ±0.26 mm/±1.1 mm	
2D	Working height: 220 mm – 400 mm ±0.5mm	
Smart camera		
Sensor type	CMOS, global shutter	
Pixel size	3.2 µm * 3.2 µm	
Sensor size	1/1.7”	
Resolution	2368 * 1760	
Maximum acquisition rate	30 fps	
Dynamic range	71.4 dB	
Signal-to-noise ratio	41 dB	
Gain	0 dB – 18 dB	
Exposure time	16 µs – 1 sec	
Pixel format	Mono 8	
Colour	Mono	
Electrical characteristics		
Data interface	Ethernet, 1 external button input, aviation connector	
Power supply	24 VDC	
Maximum power consumption	48 W@24 VDC	
Structure		
Lens interface	M12-mount, manual focusing	
Focal length	12.4 mm	
Light source	14 white LED	
Indicator	• PWR: power indicator • LNK: network indicator • STS: status indicator • OK/NG: result display indicator	
Weight	500 g	
Detection range		
Object distance	60 mm	3000 mm
Vision field	37.89 mm * 28.16 mm	1894.4 mm * 1408 mm
Single-pixel accuracy	0.016 mm	0.8 mm