Delta robot application manual
CVT conveyor tracking

www.deltaww.com

V1.00

DELTA
Smarter. Greener. Together.

Conveyor Tracking (CVT)

This section introduces the operation of Delta robots along with conveyor belts.

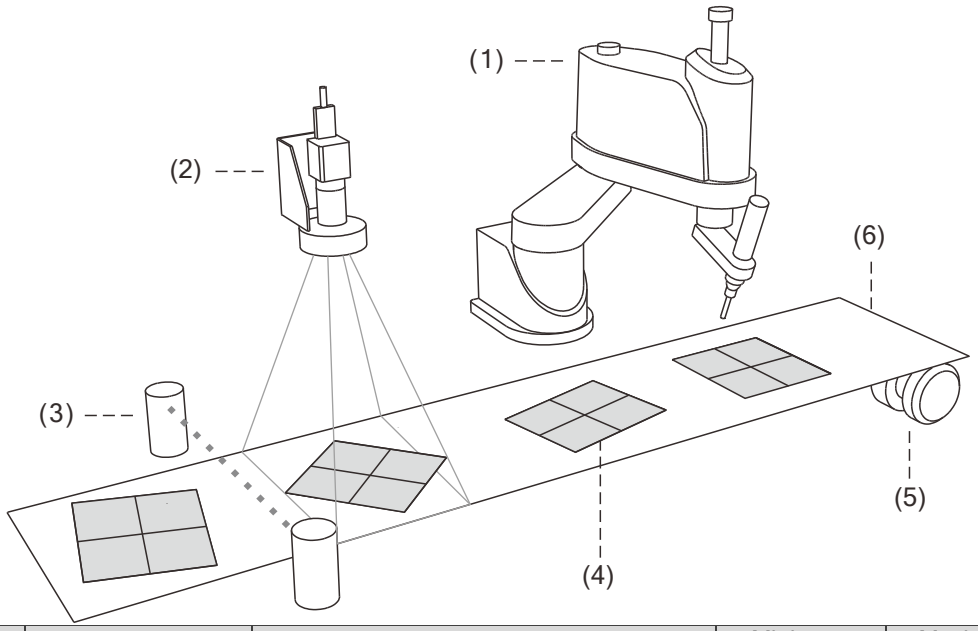
Delta's robot Conveyor Tracking application (CVT) allows robots to follow the movement of the conveyor belt to perform the operations. This section describes the system architecture, system configuration, general parameters, command descriptions, and application examples, respectively.

Important: the CVT function is a paid robot function, i.e., you must apply for this function and input the serial number to activate this function.

1. System architecture

This system is suitable for applications that require processing workpieces on moving conveyor belts, such as gluing, pick-and-place, and alignment. The overall layout of the system is shown in the following figure. You can increase quantity of the item according to requirements:

The system must contain:



Item	Name	Specifications	Minimum quantity	Maximum quantity
1	Robot	Delta robot	1 unit	4 units
2	Vision system	Vision system that supports physical DI camera*1	-	4 units
3	Sensor	24V output	-	3 pieces*3
4	Object	-	-	-
5	Encoder	A/B/Z 5V differential signal output*2	1 piece	3 pieces*4
6	Conveyor belt	-	1 set	3 sets*4

Important: this function of CVT must use the DO of the robot to trigger the camera or the DI to receive the signal. Do not connect a relay, because doing so could result in inconsistent triggering time.

Note:

*1. The quantity of vision systems and sensors can be selected and installed according to the system used.

*2. The encoder type can be either incremental or absolute.

*3. One set of conveyor belt works with one piece of sensor.

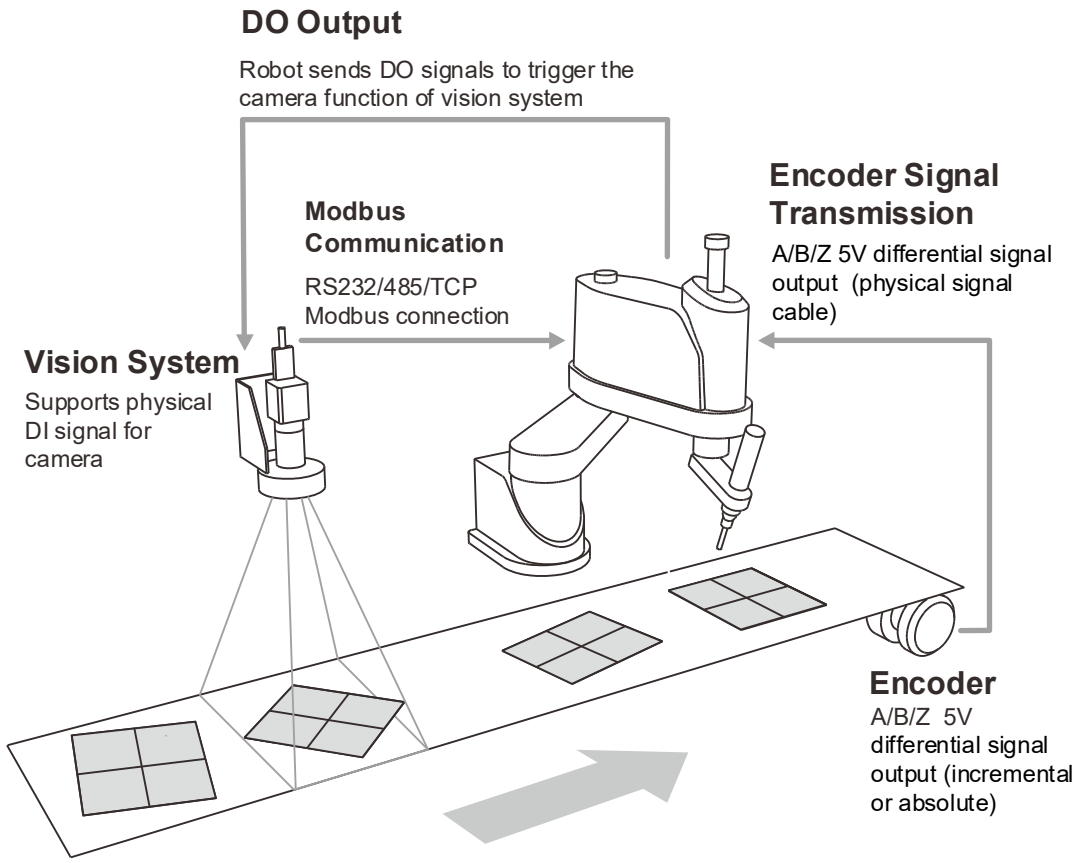
*4. One set of conveyor belt works with one piece of encoder.

2. System configuration

2.1 One robot works with one conveyor belt

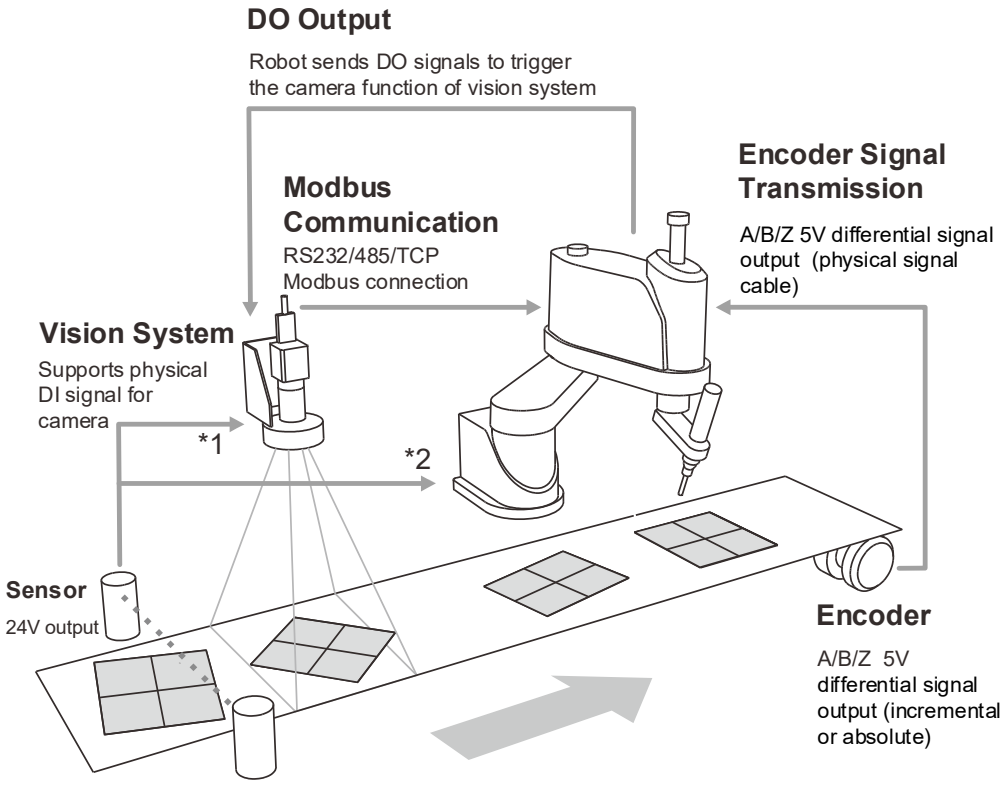
Trigger the camera with internal DO singal (with vision system but no sensor installed)	
System installation	a. RS232/485/TCP Modbus is required for connection between the vision system and robot communications. b. Connect the robot's DO1 to the vision system. c. Connect the encoder to the EXT.ENC Channel 1 of the robot.

System configuration



Trigger the camera with external sensor (with vision system and sensor installed)	
System installation	a. Connect the vision system and robot using R232. b. Connect the sensors in parallel and to DI1 of the robot and vision system. c. The encoder connects to EXT.ENC Channel 1 of the robot.

System configuration



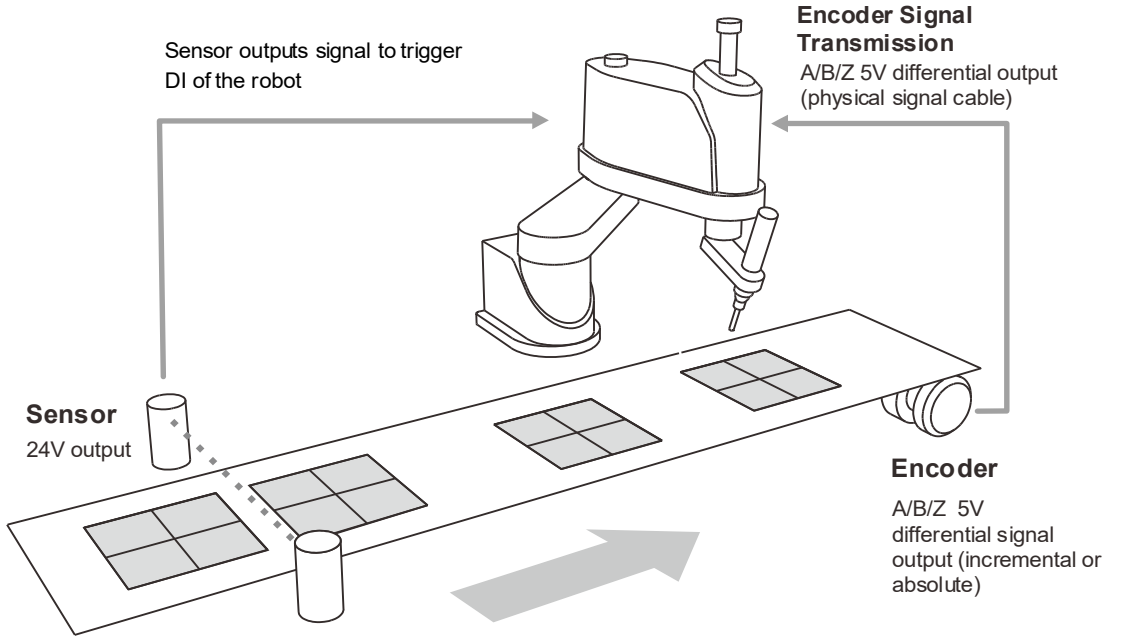
Note:

*1. The sensor outputs the signal to the vision system and triggers the camera to take pictures.

*2. The sensor outputs the signal to the robot and triggers the DI of the robot.

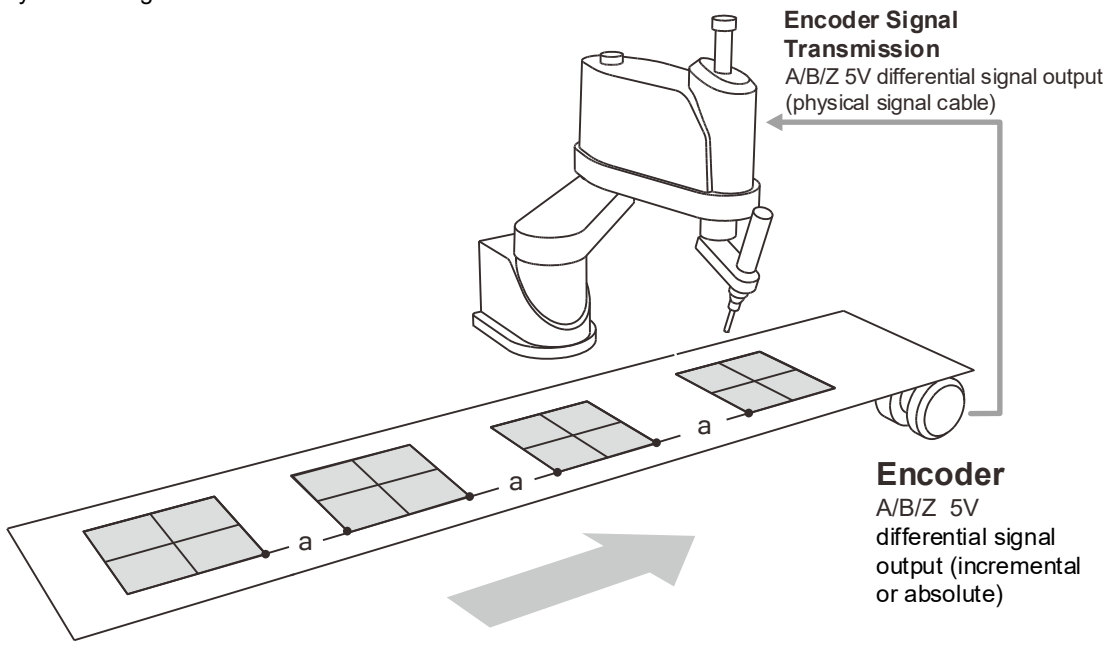
External sensor trigger (with sensor installed)	
System installation	a. Connect the sensor to DI1 of the robot b. Connect the encoder to the EXT.ENC Channel 1 of the robot

System configuration



Pitch trigger (no vision system and no sensor installed)	
System installation	Connect the encoder to the EXT.ENC Channel 1 of the robot.

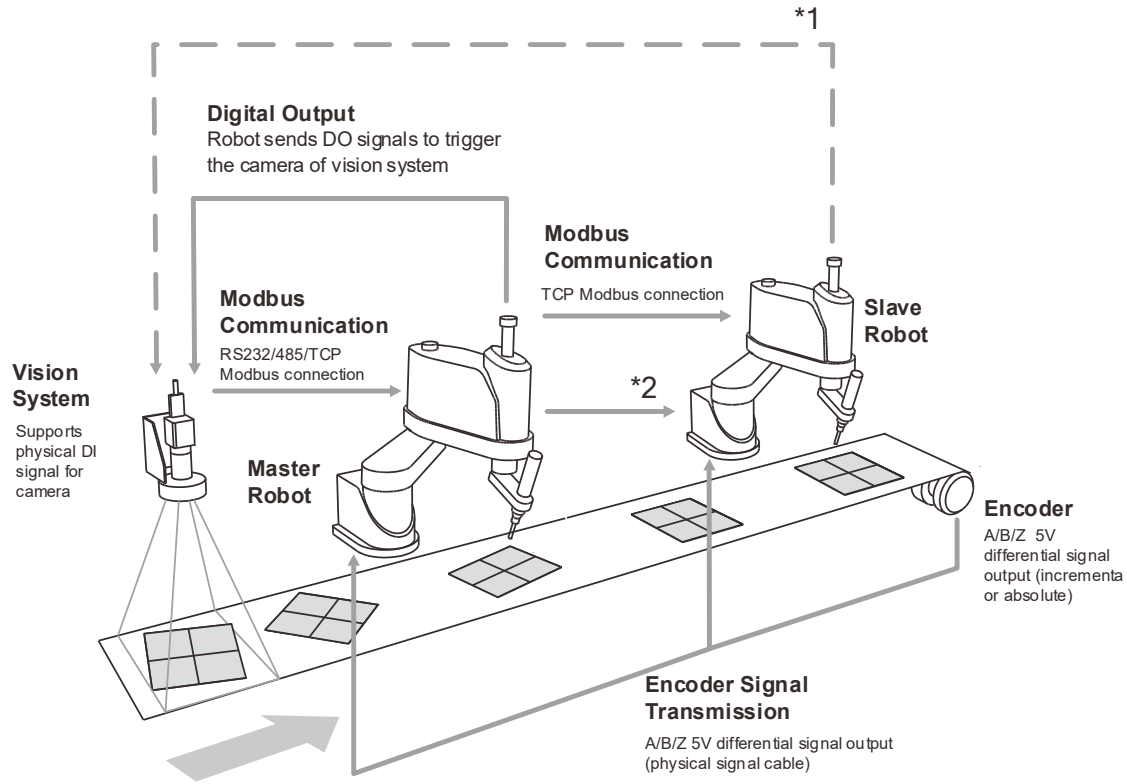
System configuration



2.2 Two robots work with one conveyor belt

Trigger the camera with external sensor (with vision system installed but no sensor)	
System installation	a. Connect the vision system and first robot using R232. b. Connect the first (master) robot's DO1 to the vision system. c. Connect the first (master) robot's DO1 to the second (slave) robot's DI1. d. Connect the encoder to the EXT.ENC Channel 1 of the two robots.

System configuration



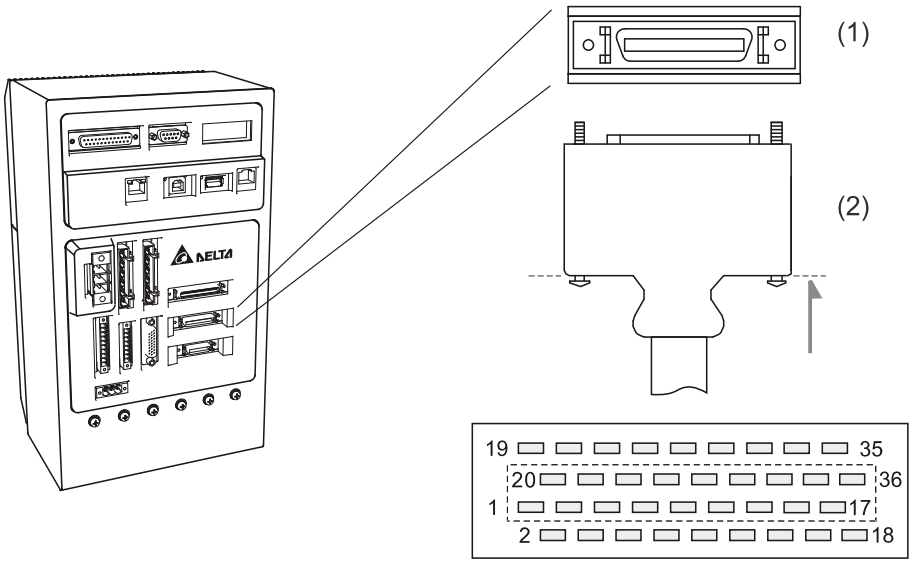
Note:

*1. Physical signal cable: the slave robot sends the DO signals to trigger the vision system to take pictures. This signal cable is for individual robot adjustment; can be removed when two robots and the system are integrated.

*2. Physical signal cable: the master robot sends the DO signal to the salve robot.

2.3 Encoder

The output format of the encoder must conform to the external encoder input format supported by the robot (A/B/Z 5V differential signal output); connect the encoder cable to the first channel of the robot's external encoder input interface.



PIN	NAME	PIN	NAME	PIN	NAME	PIN	NAME
1	Z+_1	10	-	19	Z+_3	28	-
2	Z-_1	11	GND	20	Z-_3	29	GND
3	B+_1	12	5V	21	B+_3	30	5V
4	B-_1	13	Z+_2	22	B-_3	31	Z+_4
5	A+_1	14	Z-_2	23	A+_3	32	Z-_4
6	A-_1	15	B+_2	24	A-_3	33	B+_4
7	5V	16	B-_2	25	5V	34	B-_4
8	GND	17	A+_2	26	GND	35	A+_4
9	-	18	A-_2	27	-	36	A-_4

3. Description of CVT general parameters

The CVT general parameters described in this section are automatically generated whenever a project is created. This section describes each parameter.

cvtUFIdx CVT User frame number	
Explanation of parameter	Among CVT functions, a User frame is used to represent the coordinate system of the object. Therefore, you must select a set of User frame, and this User frame number must be consistent with the User frame number used for the processing position. The default is to use the first set of the User frame; the value range: 1 - 9.

cvtFactor_den CV Speed unit conversion coefficient (denominator)	
Explanation of parameter	The source of the conveyor belt speed is the encoder; the transmitted position signal is the pulse signal; the calculation unit of the robot controller is micrometer (um), therefore, the two require a conversion coefficient. Among CVT functions, this coefficient is a fractional type.
Example	If the conveyor belt travels 12345 um each run, the encoder outputs 1000 pulses (or PUU). Fill 12345 for the **conveyor belt speed conversion coefficient (numerator)**, and the **conveyor belt speed conversion coefficient (denominator)** is 1000.

cvtFactor_num CV Speed unit conversion coefficient (numerator)	
Explanation of parameter	The source of the conveyor belt speed is the encoder; the transmitted position signal is the pulse signal; the calculation unit of the robot controller is micrometer (um), therefore, the two require a conversion coefficient. Among CVT functions, this coefficient is a fractional type.
Example	If the conveyor belt travels 12345 um each run, the encoder outputs 1000 pulses (or PUU). Fill 12345 for the **conveyor belt speed conversion coefficient (numerator)**, and the **conveyor belt speed conversion coefficient (denominator)** is 1000.

interval Conveyor belt speed average value filter time	
Explanation of parameter	Among CVT functions, this provides the function of average value filter, which can filter the possible noise from the encoder signal. The unit of this parameter is ms.
Example	If this parameter is set to 10, it means that the robot controller will use the average speed of the conveyor belt at 10 ms as the conveyor belt speed per 1 ms.

trans_ccd_x Vision coordinate system X direction offset value	
Explanation of parameter	The vision coordinate system is located at the X direction offset value of the World frame.

trans_ccd_y Vision coordinate system Y direction offset value	
Explanation of parameter	The vision coordinate system is located at the Y direction offset value of the World frame.

vuPix2UmNum	Vision distance unit conversion coefficient (numerator)
Explanation of parameter	The position unit output by the vision system; the floating-point number may vary according to different vision systems, therefore, the coefficient conversion unit is required. Among CVT functions, this coefficient is designed as a fractional type. Example description --CCD Unit Ratio vuPix2UmNum = 10 vuPix2UmDen = 1 --When the unit of the position value sent by the vision system is 0.01, the object offset distance conversion coefficient vuPix2UmNum should be set to 10 at this time to conform to the system unit 0.001 of the robot controller. --CCD Unit Ratio vuPix2UmNum = 1 vuPix2UmDen = 1 --When the unit of the position value sent by the vision system is 0.001, the object offset distance conversion coefficient vuPix2UmNum should be set to 1 at this time to conform to the system unit 0.001 of the robot controller.

vuPix2UmDen	Object offset distance conversion unit coefficient (denominator)
Explanation of parameter	The position unit output by the vision system; the floating-point number may vary according to different vision systems, therefore, the coefficient conversion unit is required. Among CVT functions, this coefficient is designed as a fractional type. The default value is 1; see vuPix2UmNum for an example.

vuAgRatioNum	Object final angle conversion coefficient (numerator)
Explanation of parameter	The angle unit output by the vision system; the floating-point number may vary according to different vision systems, therefore, the coefficient conversion unit is required. Among CVT functions, this coefficient is designed as a fractional type. Example description --CCD Unit Ratio vuAgRatioNum = 10 vuAgRatioDen = 1 --When the unit of the angle value sent by the vision system is 0.01, the object offset distance conversion coefficient vuAgRatioNum should be set to 10 to conform to the system unit of the robot controller 0.001. --CCD Unit Ratio vuAgRatioNum = 1 vuAgRatioDen = 1 --When the unit of the angle value sent by the vision system is 0.001, the object offset distance conversion coefficient vuAgRatioNum should be set to 1 to conform to the system unit 0.001 of the robot controller.

vuAgRatioDen	Object final angle conversion coefficient (denominator)
Explanation of parameter	The angle unit output by the vision system; the floating-point number may vary according to different vision systems, therefore, the coefficient conversion unit is required. Among CVT functions, this coefficient is designed as a fractional type. The default value is 1. For an example, see vuAgRatioNum.

vuXYExchgFlag	Vision data XY value exchange flag (the default is 0)
Explanation of parameter	Set to 0 if the XY axes of the vision coordinate system are in the same direction as the XY axes of the World coordinate system. If they are opposite, set the vision XY value exchange flag to 1.

4. Description of CVT commands

The purpose of the Conveyor Tracking (CVT) application is so that the robot can work synchronously with the conveyor belt as it moves to complete the dynamic tracking.

- CVT_ChangeMotion (Change to CVT motion mode)
- CVT_SelectMode (Select CVT tracking mode)
- CVT_SetTriggerMode (Select CVT trigger mode)
- CVT_SetCamTrigTime (Set DO timing trigger interval)
- CVT_SetCamTrigDist (Set DO pitch trigger interval)
- CVT_CamTrigDistStart (Enable DO pitch trigger encoder reset)
- CVT_CamTrigDistStop (Disable DO pitch trigger encoder reset)
- CVT_CalRobotTrigLine (Configure the CVT start line)
- CVT_CalZoneEndLine (Configure the CVT end line)
- CVT_SetObjType (Set the object type title)
- CVT_GetObjType (Read the current object type title)
- CVT_GetCVSpeed (Read the conveyor belt speed)
- CVT_GetEncoderPulseCount (Read the value of the encoder)
- CoordTransform_2D_Use (Transform pixel values)
- CoordTransform_2D_Reset (Reset transformed pixel value)
- CVT_SetNGZoneExistArea (Set the effective area of the forbidden zone)
- CVT_VelIn (Enable conveyor belt tracking mode)
- CVT_VelOut (Disable conveyor belt tracking mode)
- CVT_ObjDoneFlag (Set the done flag of the object)
- CVT_ChkBotTrigCond (Feedback signal of the tracking condition)
- CVT_TrigSwitchON (Turn on the receive flag)
- CVT_TrigSwitchOFF (Turn off the receive flag)
- CVT_GetQueueWorkCnt (Quantity of objects to track)
- CVT_GetQueueIdx (Number of object in queue for tracking)
- CVT_SafetyCtrlFlag (Configure the light curtain mode of the robot)
- CVT_SetUserDefineDI (Set the DI number of external trigger mode)
- CVT_SetUserDefineDO (Set the DO number of internal trigger mode)
- CVT_SetVisionDirection (Set the opposite direction of vision angle)
- CVT_OverEndLineCnt (Quantity of unprocessed objects which are over the line)

- CVT_GetWorkInfo (Outputs currently tracked object coordinates)
- CVT_Master (Set robot as master)
- CVT_Robot2 (Set second slave robot)
- CVT_Robot3 (Set third slave robot)
- CVT_Robot4 (Set fourth slave robot)
- CVT_Slave (Set robot as slave)
- CVT_SetOverEndLineDO (Set over-line DO output)

4.1 CVT_ChangeMotion (Change to CVT motion mode)

Have the robot changed to CVT motion mode.

■ Syntax

CVT_ChangeMotion( )

Example description

```
CVT_ChangeMotion( )  -- Change to CVT motion mode
```

4.2 CVT_SelectMode (Select CVT tracking mode)

Set whether the CVT mode works with the vision system.

■ Syntax

CVT_SelectMode(CV_Index, Mode)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Mode[number]

Select the tracking mode.

1: vision tracking;

2: no vision tracking.

Example description

```
CVT_SelectMode(1,1)  --First group of CVT is set as vision tracking
CVT_SelectMode(2,2)  -- Second group of CVT is set as no vision tracking
```

4.3 CVT_SetTriggerMode (Select CVT trigger mode)

Select whether or not the CVT trigger mode has a matching sensor.

■ **Syntax**

CVT_SelectMode(CV_Index, Mode)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Mode[number]

Trigger mode.

1: internal trigger (DO trigger, no sensor);

2: external trigger (DI trigger, with sensor).

Example description

```
CVT_SetTriggerMode (1,1)    -- First group of CVT is set as DO trigger
CVT_SetTriggerMode (2,2)    -- Second group of CVT is set as DI trigger
```

4.4 CVT_SetCamTrigTime (DO timing trigger interval)

If CVT_SetTriggerMode is set as internal trigger, you must set the DO trigger interval.

Important: do not use this command with CVT_SetCamTrigDist.

■ **Syntax**

CVT_SelectMode(CV_Index, Time)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Time[number]

Unit is ms; the range is an integer from 1 to 65535.

Example description

```
CVT_SetCamTrigTime (1,500)    -- First group of CVT is set to 500 ms for each DO trigger interval time
```

4.5 CVT_SetCamTrigDist (DO pitch trigger interval)

If CVT_SetTriggerMode is set as DO trigger, you must set the trigger interval.

Important: do not use this command with CVT_SetCamTrigTime.

■ Syntax

CVT_SelectMode(CV_Index, Time)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Time[number]

Unit is mm; the range is an integer from 1 to 65535.

Example description

```
CVT_SetCamTrigDist (1,100)  -- First group of CVT is set to 100 mm for each DO trigger interval distance
```

4.6 CVT_CamTrigDistStart (Enable DO pitch trigger encoder reset)

- When you use a pitch trigger command, enable the encoder reset function.
- It is necessary to execute CVT_SetCamTrigDist before executing this command.
- Make sure the robot has executed CVT_SetCamTrigDist or CVT_CamTrigDistStop before executing this command.

■ Syntax

CVT_CamTrigDistStart(CV_Index)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Example description 1

```
--When the program has just started running, but CVT_CamTrigDistStart has not been executed yet:
CVT_SetCamTrigDist (100)        -- CVT is set to 100 mm for each DO trigger interval distance
CVT_CamTrigDistStart(1)         -- Enable encoder reset of first group of CVT
```

Example description 2

```
-- When the program has started running, and CVT_CamTrigDistStart has been executed:
CVT_CamTrigDistStop(1)          -- Disable encoder reset of first group of CVT
CVT_SetCamTrigDist (100)        -- CVT is set to 100 mm for each DO trigger interval distance
CVT_CamTrigDistStart(1)         -- Enable encoder reset of first group of CVT
```

4.7 CVT_CamTrigDistStop (Disable DO pitch trigger encoder reset)

This command disables the encoder reset function.
While the program is running, if you want to reset the pitch of the pitch trigger, you need to execute this command first and then execute CVT_SetCamTrigDist.

■ Syntax

CVT_ CamTrigDistStop (CV_Index)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Example description

```
--When the program has started executing, and CVT_CamTrigDistStart has been executed:
CVT_CamTrigDistStop(1)          -- Disable encoder reset of first group of CVT
CVT_SetCamTrigDist (100)        -- CVT is set to 100 mm for each DO trigger interval distance
CVT_CamTrigDistStart(1)         -- Enable encoder reset of first group of CVT
```

4.8 CVT_CalRobotTrigLine (Configure the CVT start line)

Configure the CVT start line (perpendicular to the direction of conveyor belt movement).

■ Syntax

robotTrigLine = CVT_CalRobotTrigLine(robotTrigLinePX, robotTrigLinePY, cmpstVectorX, cmpstVectorY)

robotTrigLinePX[number]

The X coordinate of the start line positioning point; unit is um, and the range is an integer from -999999 to +999999.

robotTrigLinePY[number]

The Y coordinate of the start line positioning point; unit is um, and the range is an integer from -999999 to +999999.

cmpstVectorX[number]

The X component of the conveyor belt vector; the range is an integer from -32767 to +32767.

cmpstVectorY[number]

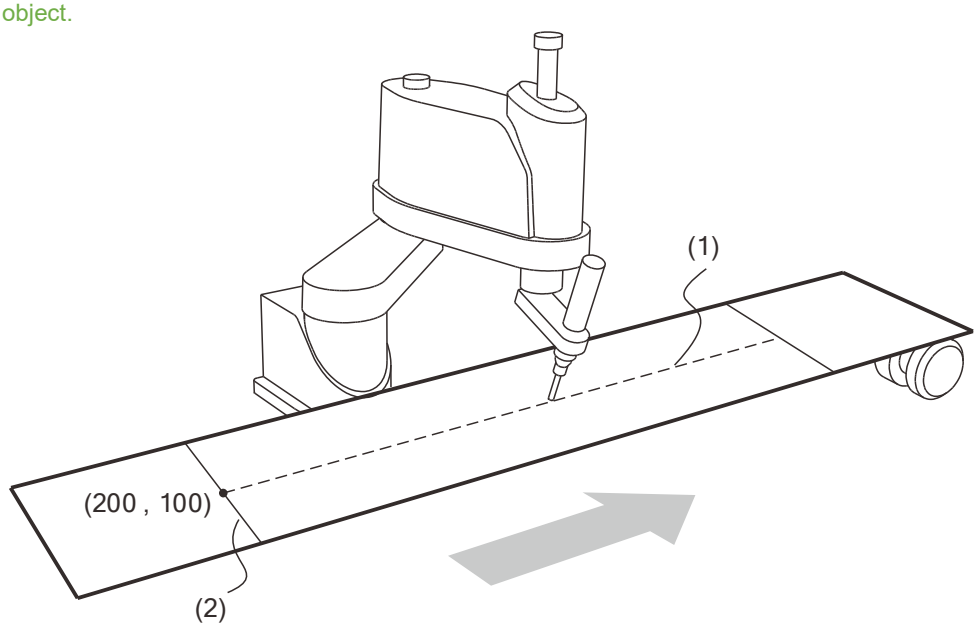
The Y component of the conveyor belt vector; the range is an integer from -32767 to +32767.

robotTrigLine

Set the start line. This variable is input to the CVT initialization command. Do not modify the title of this variable.

Example description

```
robotTrigLine = CVT_CalRobotTrigLine (200000, -100000, 0, 1000)
--When the object passes through the robot coordinates (200,-100) on the virtual start line perpendicular
to the conveyor belt, and if the robot is in standby at this time, the robot will track and process the
object.
```



(1) CV Vector(0.1000) (2) Start line

4.9 CVT_CalZoneEndLine (Configure the CVT end line)

Configure the CVT end line (perpendicular to the direction of conveyor belt movement).

■ Syntax

zoneEndLine = CVT_CalZoneEndLine(zoneEndPX, zoneEndPY, cmpstVectorX, cmpstVectorY)

zoneEndPX[number]

The X coordinate of the end line positioning point; unit is um, and the range is an integer from -999999 to +999999.

zoneEndPY[number]

The Y coordinate of the end line positioning point; unit is um, and the range is an integer from -999999 to +999999.

cmpstVectorX[number]

The X component of the conveyor belt vector; the range is an integer from -32767 to +32767.

cmpstVectorY[number]

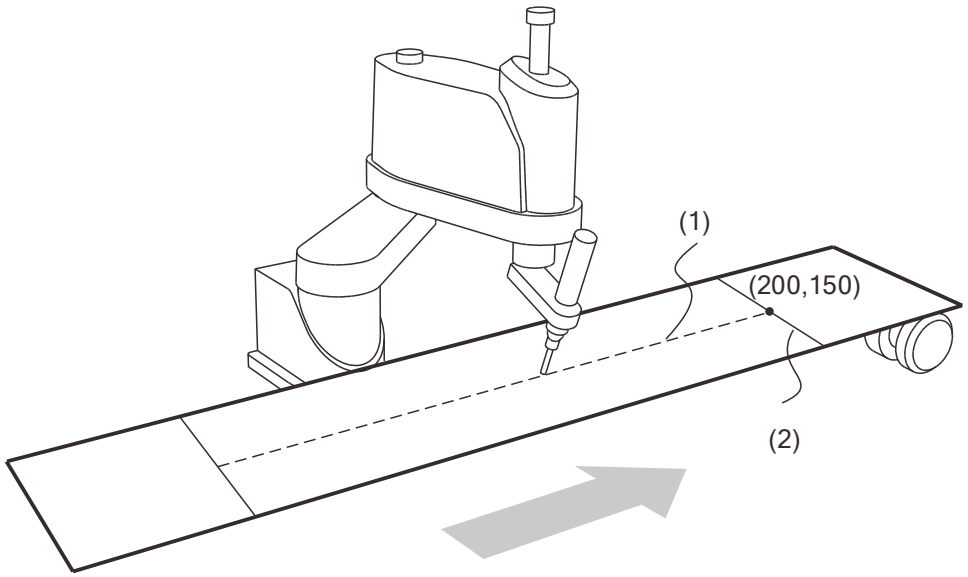
The Y component of the conveyor belt vector; the range is an integer from -32767 to +32767.

zoneEndLine

Set the end line. This variable is inputto the CVT initialization command. Do not modify the title of this variable.

Example description

```
zoneEndLine = CVT_CalZoneEndLine(200000, 150000, 0, 1000)    -- When the target object passes through the robot coordinates (200, 150) on the virtual end line perpendicular to the conveyor belt, and if the object has not been tracked at this time, the object for processing will be excluded; if it has already been tracked, the robot will ignore the end line and continue to track.
```



(1) CV Vector(0.1000) (2) End line

4.10 CVT_SetObjType (Set the object type title)

When there are multiple objects in a project, you can set the type titles of the objects with up to 10 obejct types This command can be used with CVT_GetObjType. ObjType can go up to ObjType10. If it is not set with a number, all type titles are 0.

■ Syntax

CVT_SetObjectType (CV_Index, ObjType1, ObjType2, …)

CV_Index[number]

CVT group, the input range is an integer from 1 to 3.

ObjType1[number]

Set the title for the first set of object type. The input range is an integer from 1 to 255. If it is not filled in, the type is set to 0.

ObjType2[number]

Set the title for the second set of object type. The input range is an integer from 1 to 255. If it is not filled in, the type is set to 0.

Example description

```
CVT_SetObjectType (1, 2, 5)   -- Set the first conveyor belt; the object type title of the first group of storage spaces is 2; the object type title of the second group of storage spaces is 5; and for the 3rd to 10th groups of storage spaces, the object type titles are 0 by default.
```

4.11 CVT_GetObjType (Read the current object type title)

Read the current object type title.

■ Syntax

ObjType=CVT_GetObjType(CV_Index)

CV_Index[number]

CVT group, the input range is an integer from 1 - 3.

ObjType[number]

Read the current object type title; the value range is an integer from 0 to 255.

Example description

```
CVT_SetObjectType (1, 2, 5)   -- Set the first conveyor belt; the object type title of the first group of storage spaces is 2; the object type title of the second group of storage spaces is 5; and for the 3rd to 10th groups of storage spaces, the object type title default value is 0
ObjType=CVT_GetObjType (1)        -- Read the type of the object currently being tracked by the first conveyor belt. If the current object is the second type, the value of ObjType is 5.
```

4.12 CVT_GetCVSpeed (Read the conveyor belt speed)

Obtain current speed of the conveyor.

■ **Syntax**

CVSpeed=CVT_GetCVSpeed(CV_Index)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

CVSpeed[number]

The current conveyor speed in units of mm/sec; the range is an integer from 1 to 65535.

Example description

```
CVSpeed=CVT_GetCVSpeed(1)          -- Read the conveyor speed of first group of CVT
```

4.13 CVT_GetEncoderPulseCount (Read the value of the encoder)

Obtain the current value of the conveyor belt encoder.

■ **Syntax**

EncPos=CVT_GetEncoderPulseCount (CV_Index)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

EncPos[number]

The current value of the conveyor belt encoder; the range is an integer from -2147483648 to +2147483647.

Example description

```
EncPos=CVT_GetEncoderPulseCount (1)  -- Read the value of encoder of first group of CVT
```

4.14 CoordTransform_2D_Use (Transform pixel values)

Perform four-point calibration on the robot and vision system; convert the pixel value of the camera pixel into the value in units of mm for the robot.

■ **Syntax**

CoordTransform_2D_Use(CV_Index, C1, C2, C3, C4, P1, P2, P3, P4)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

C1[number]

The value of the first pixel of vision. Input values such as X and Y in units of pixels.

C2[number]

The value of the second pixel of vision. Input values such as X and Y in units of pixels.

C3[number]

The value of the third pixel of vision. Input values such as X and Y in units of pixels.

C4[number]

The value of the fourth pixel of vision. Input values such as X and Y in units of pixels.

P1[number]

The value of the first point of the robot. Input values such as X and Y in units of mm.

P2[number]

The value of the second point of the robot. Input values such as X and Y in units of mm.

P3[number]

The value of the third point of the robot. Input values such as X and Y in units of mm.

P4[number]

The value of the fourth point of the robot. Input values such as X and Y in units of mm.

Example description

```
C1=Point(279.113,667.338)      -- The value of the first pixel of vision; in units of pixels
C2=Point(2069.116,672.439)     -- The value of the second pixel of vision; in units of pixels
C3=Point(2068.123,1447.155     -- The value of the third pixel of vision; in units of pixels
C4=Point(274.305,1451.973)     -- The value of the fourth pixel of vision; in units of pixels

P1 = Point(285.741,-194.639)   -- The value of the first point of the robot; in units of mm
P2 = Point(205.741,-194.639)   -- The value of the second point of the robot; in units of mm
P3 = Point(205.741,-94.639)    -- The value of the third point of the robot; in units of mm
P4 = Point(285.741,-94.639)    -- The value of the fourth point of the robot; in units of mm

CoordTransform_2D_Use(1,C1, C2, C3, C4, P1, P2, P3, P4)        -- Transform the first group of CVT
                                                                  pixel values to robot values
```

4.15 CoordTransform_2D_Reset (Reset pixel transform)

Reset the transformed pixel values for the vision system and robot.

■ **Syntax**

CoordTransform_2D_Reset (CV_Index)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Example description

```
CoordTransform_2D_Reset(1)        -- Reset vision pixel transformation of first group of CVT
```

4.16 CVT_SetNGZoneExistArea (Set the effective area of the forbidden zone)

Set the effective area of the forbidden zone. The system calculates the center point based on P1 - P4 for robot vision calibration. When the workpiece leaves the existing forbidden zone on the conveyor belt and the workpiece's distance from the center point is greater than the radious set for the forbidden zone, the workpiece is regarded as "out of the existing forbidden zone".

■ **Syntax**

CVT_SetNGZoneExistArea (CV_Index, P1, P2, P3, P4, radius)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

P1[number]

The value of the first point of the robot during calibration. Input values such as X and Y in units of mm.

P2[number]

The value of the second point of the robot during calibration. Input values such as X and Y in units of mm.

P3[number]

The value of the third point of the robot during calibration. Input values such as X and Y in units of mm.

P4[number]

The value of the fourth point of the robot during calibration. Input values such as X and Y in units of mm.

radius[number]

The radius of the forbidden zone in units of mm.

Example description

```
P1 = Point(285.741,-194.639)      -- The value of the first point of the robot during calibration in units
of mm
P2 = Point(205.741,-194.639)      -- The value of the second point of the robot during calibration; in
units of mm
P3 = Point(205.741,-94.639)       -- The value of the third point of the robot during calibration; in
units of mm
```

```
P4 = Point(285.741,-94.639)          -- The value of the fourth point of the robot during calibration in units of mm
radius =66                           --Radius of the forbidden zone in units of mm
CVT_SetNGZoneExistArea(1, P1, P2, P3, P4, radius)      -- Set the effective area of the forbidden zone of the first group of CVT
```

4.17 CVT_VelIn (Enable conveyor belt tracking mode)

Enable the conveyor belt tracking mode of the robot.

■ **Syntax**

CVT_VelIn (CV_Index)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Example description

```
CVT_VelIn (1)      --The robot starts to move in accordance with the speed of the first conveyor belt
```

4.18 CVT_VelOut (Disable conveyor belt tracking mode)

Disable the conveyor belt tracking mode of the robot.

■ **Syntax**

CVT_VelOut (CV_Index)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Example description

```
CVT_VelOut (1)     --The robot stops following the movement of the conveyor belt
```

4.19 CVT_ObjDoneFlag (Set the done flag of the object)

When the robot completes tracking, the done flag of the set object is marked as completed; otherwise, when the tracking is not complete, it is set as incomplete.

■ **Syntax**

CVT_ObjDoneFlag (CV_Index, DoneFlag)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

DoneFlag[number]

Done flag for the CVT operation.

0: tracking work is not complete;

1: the tracking work is complete.

Example description

```
CVT_ObjDoneFlag (1,0)                --Inform the system that the robot has not completed the workpiece processing currently being tracked on the first conveyor belt.
-------------------------------------------------
Robot operation procedure
-------------------------------------------------
CVT_ObjDoneFlag (1,1)                --Inform the system that the robot has completed the worpiece porcessing currently being tracked on the first conveyor belt.
```

4.20 CVT_ChkBotTrigCond (Feedback signal of the tracking condition)

Determine whether the object to be tracked meets the tracking conditions.

■ **Syntax**

TrackFlag = CVT_ChkBotTrigCond (CV_Index)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

TrackFlag[number]

Feedback signal of the tracking condition.

0: tracking conditions have not been established;

1: tracking conditions have been established;

2: the target has been locked and tracked.

Example description

```
while CVT_ChkBotTrigCond (1) == 0 do end        --Wait for the object to be processed to meet the tracking conditions
```

4.21 CVT_TrigSwitchON (Turn on the receive flag)

Enable the system to receive the trigger signal. This command is used with CVT_TrigSwitchOFF. CVT_TrigSwitchON is on by default.

■ Syntax

CVT_TrigSwitchON (CV_Index)

CV_Index[number]

CVT group; the input range is an integer 1 - 3.

Example description

```
CVT_TrigSwitchON (1)   -- Turn on receive flag for first group of CVT
```

4.22 CVT_TrigSwitchOFF (Turn off the receive flag)

This command tuns off the signal receiving of the system, so any DIO triggering or received vision data will be ignored, and the system does not perform conveyor tracking. This is used along with CVT_TrigSwitchON.

■ Syntax

CVT_TrigSwitchOFF (CV_Index)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

Example description

```
CVT_TrigSwitchOFF (1)        -- Turn off receive flag for first group of CVT
```

4.23 CVT_GetQueueWorkCnt (Quantity of objects to track)

Get the total quantity of current workpieces to be tracked.

■ **Syntax**

WorkCnt = CVT_GetQueueWorkCnt (CV_Index)

CV_Index[number]

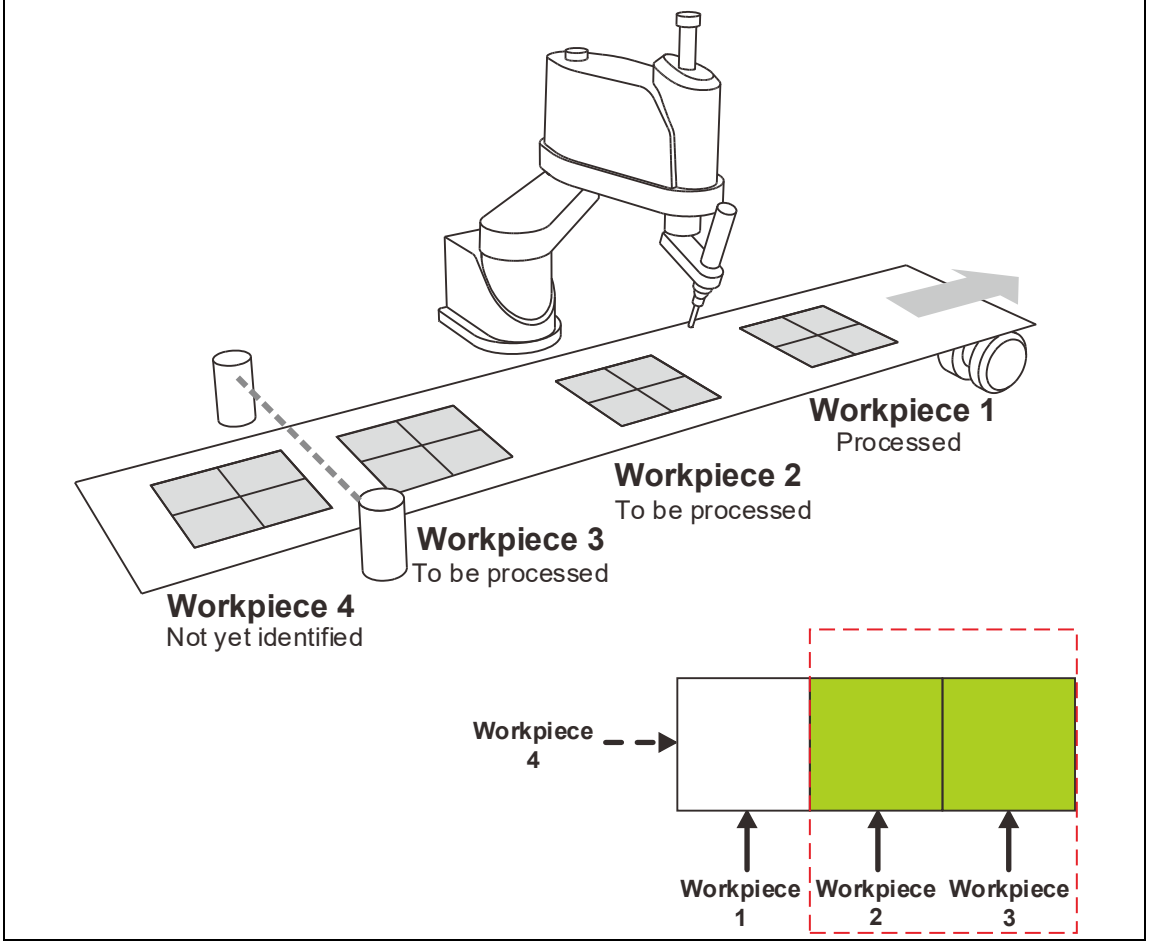
CVT group; the input range is an integer from 1 to 3.

WorkCnt[number]

The quantity of workpieces to be processed; the range of four and five-axis robots is 1 - 100, and the range of six-axis robots is 1 - 10.

Example description

```
WorkCnt = CVT_GetQueueWorkCnt (1) -- The value of WorkCnt is 2 at this time
```



4.24 CVT_GetQueueIdx (Number of object in queue for tracking)

Get the number of current workpieces in queue for tracking.

■ Syntax

HeadIndex = CVT_GetQueueIdx (CV_Index)

CV_Index[number]

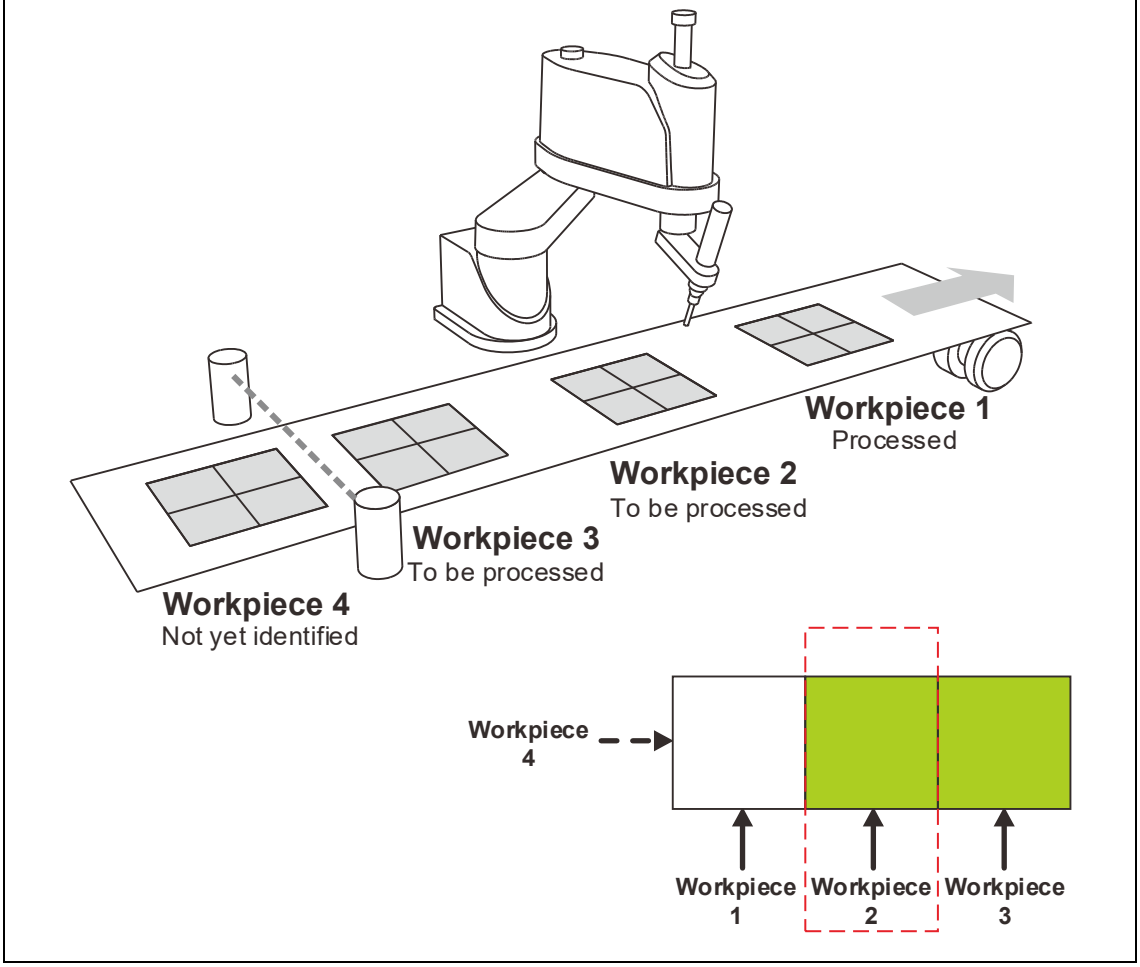
CVT group, the input range is 1 - 3.

HeadIndex[number]

The number of workpieces in queue for processing; the range of four and five-axis robots is 1 - 100, and the range of six-axis robots is 1 - 10.

Example description

```
HeadIndex = CVT_GetQueueIdx (1) -- The HeadIndex value is 2 at this time
```



4.25 CVT_SafetyCtrlFlag (Configure the light curtain mode of the robot)

Set the robot action after the light curtain is triggered during conveyor tracking operation.

■ **Syntax**

CVT_SafetyCtrlFlag (SafetyFlag)

SafetyFlag[number]

Light curtain mode. The default value is 0.

0: after the light curtain is triggered, the robot stops operating;

1: after the light curtain is triggered, the robot continues to operate.

Example description

```
CVT_SafetyCtrlFlag (1)  -- Trigger the safety light curtain during tracking; the robot continues the tracking, and does not stop unless it no longer meets tracking condition
```

4.26 CVT_SetUserDefineDI (DI number of external trigger mode)

When CVT_SetTriggerMode is set to external trigger, you can specify the DI number used for the trigger condition of the conveyor belt. If it is not set to external trigger, the first group of conveyor belts is 1, the second group of conveyor belts is 2, and the third group of conveyor belts is 3 by default.

■ **Syntax**

CVT_SetUserDefineDI (CV_Index, DIIndex)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

DIIndex[number]

Set the DI number for external trigger; the range is 1 - 24.

Example description

```
CVT_SetUserDefineDI (1 ,10)  -- Set the DI number of the external trigger of the first group of conveyor belts to DI 10
```

4.27 CVT_SetUserDefineDO (Set the DO number of internal trigger mode)

When CVT_SetTriggerMode is set to internal trigger, you can specify the DO number used for the trigger condition of the conveyor belt. If it is not set to internal trigger, the first group of conveyor belts is 1, the second group of conveyor belts is 2, and the third group of conveyor belts is 3 by default.

■ **Syntax**

CVT_SetUserDefineDO (CV_Index, DOIndex)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

DOIndex[number]

Set the DO number for internal trigger; the range is 1 - 12.

Example description

```
CVT_SetUserDefineDO (1 ,9)  -- Set the DO number of the internal trigger of the first group of conveyor belts to DO 9
```

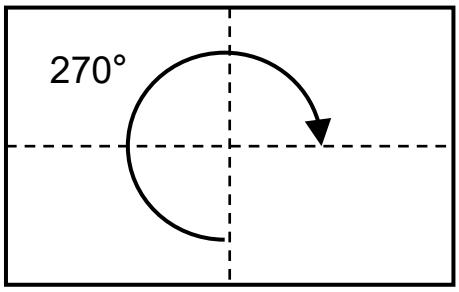
4.28 CVT_SetVisionDirection (Set the opposite direction of vision angle)

Reverse the output direction of the vision angle.

■ **Syntax**

CVT_SetVisionDirection (1)

Example description




```
--When no command is entered, the angle received by the robot is 270 degrees
CVT_SetVisionDirection (1)    -- After the vision angle is set to the opposite direction, the angle received by the robot is -270 degrees
```

4.29 CVT_OverEndLineCnt (Quantity of unprocessed objects which are over the line)

Count the quantity of the unprocessed workpieces which are over the end line.

■ **Syntax**

OverIndex = CVT_OverEndLineCnt (CV_Index)

CV_Index[number]

CVT group, the input range is 1 - 3.

OverIndex[number]

The quantity of the unprocessed workpieces that exceed the end line. The range is an integer from 0 to 60,000. When the number is greater than 60,000, it automatically returns to 0.

Example description

```
OverIndex = CVT_OverEndLineCnt (1)        -- Read the quantity of workpieces exceeding the end line in the first group of CVT
```

4.30 CVT_GetWorkInfo (Outputs currently tracked workpiece coordinates)

Output the coordinates of the currently tracked workpiece at the time the picture is taken by the vision system.

■ **Syntax**

PosX,PosY,PosC= CVT_GetWorkInfo (CV_Index)

CV_Index[number]

CVT group, the input range is 1 - 3.

PosX[number]

The vision X direction value of the currently tracked workpiece; in units of mm.

PosY[number]

The vision Y direction value of the currently tracked workpiece; in units of mm.

PosC[number]

The vision angle of the workpiece currently being tracked, in degrees.

Example description

```
PosX,PosY,PosC= CVT_GetWorkInfo(1)        -- Read the coordinates of the currently tracked workpiece in first group of CVT during vision system photographing.
```

4.31 CVT_Master (Set robot as master)

Set the robot as the master of the master-slave architecture.

■ **Syntax**

CVT_Master (RobotNumber, CV_Index, ObjectNumber)

RobotNumber[number]

Set the number of robots for the work. The input range is an integer from 2 to 4. Use along with the command CVT_Robot2,3,4.

CV_Index[number]

The specified conveyor belt group; the input range is an integer from 1 to 3.

ObjectNumber[table]

The quantity of workpieces to be continuously processed by each robot.

Example description

```
--When the architecture is 2 robots and 2 CVT groups
CVT_Master(2,1,{2,2})          -- Set this robot as the master.Set two robots as a group. For the actions of the first group of CVT, the first robot processes two objects in a row, then the second robot processes 2 objects in a row, then the first robot processes 2 objects in a row, and so on.
CVT_Master(2,2,{1,2})          -- Set this robot as the master. Set two robots as a group. For the actions of the second group of CVT, the first robot processes one object, then the second robot processes 2 objects in a row, then the first robot processes 1 object, and so on.
```

4.32 CVT_Robot2 (Set second slave robot)

Set the IP and status of the second slave robot.

■ **Syntax**

CVT_Robot2 (IP, status)

IP[string]

Set the IP address of the slave robot to be connected.

status[word]

Set the running status of the slave robot.

true: start running, and the data is transmitted normally;

false: stop running, and data transmission is stopped.

Example description

```
CVT_Robot2("192.168.1.2",true) --Configure the IP of the second slave robot to 192.168.1.2 and the status to on
```

4.33 CVT_Robot3 (Set third slave robot)

Set the IP and status of the third slave robot.

■ **Syntax**

CVT_Robot3 (IP, status)

IP[string]

Set the IP address of the slave robot to be connected.

status[word]

Set the running status of the slave robot.

true: start running, and the data is transmitted normally;

false: stop running, and data transmission is stopped.

Example description

```
CVT_Robot3("192.168.1.3",true) --Configure the IP of the third slave robot to 192.168.1.3 and the status to on
```

4.34 CVT_Robot4 (Set fourth slave robot)

Set the IP and status of the fourth slave robot.

■ **Syntax**

CVT_Robot4 (IP, status)

IP[string]

Set the IP address of the slave robot to be connected.

status[word]

Set the running status of the slave robot.

true: start running, and the data is transmitted normally;

false: stop running, and data transmission is stopped.

Example description

```
CVT_Robot4("192.168.1.4",true) --Configure the IP of the fourth slave robot to 192.168.1.4 and the status to on
```

4.35 CVT_Slave (Set robot as slave)

Set the robot as the slave of the master-slave architecture. All robots are slaves by default. So, this function is usually not required.

■ Syntax

CVT_Slave( )

Example description

```
CVT_Slave()        --Set this robot as slave
```

4.36 CVT_SetOverEndLineDO (Set over-line DO output)

Set to send the DO signal as a reminder when the object goes over the end line. The DO signal On duration is about 0.04 seconds.

■ Syntax

CVT_SetOverEndLineDO (CV_Index, DOIndex)

CV_Index[number]

CVT group; the input range is an integer from 1 to 3.

DOIndex[number]

Set the DO number for internal trigger; the range is 1 - 12.

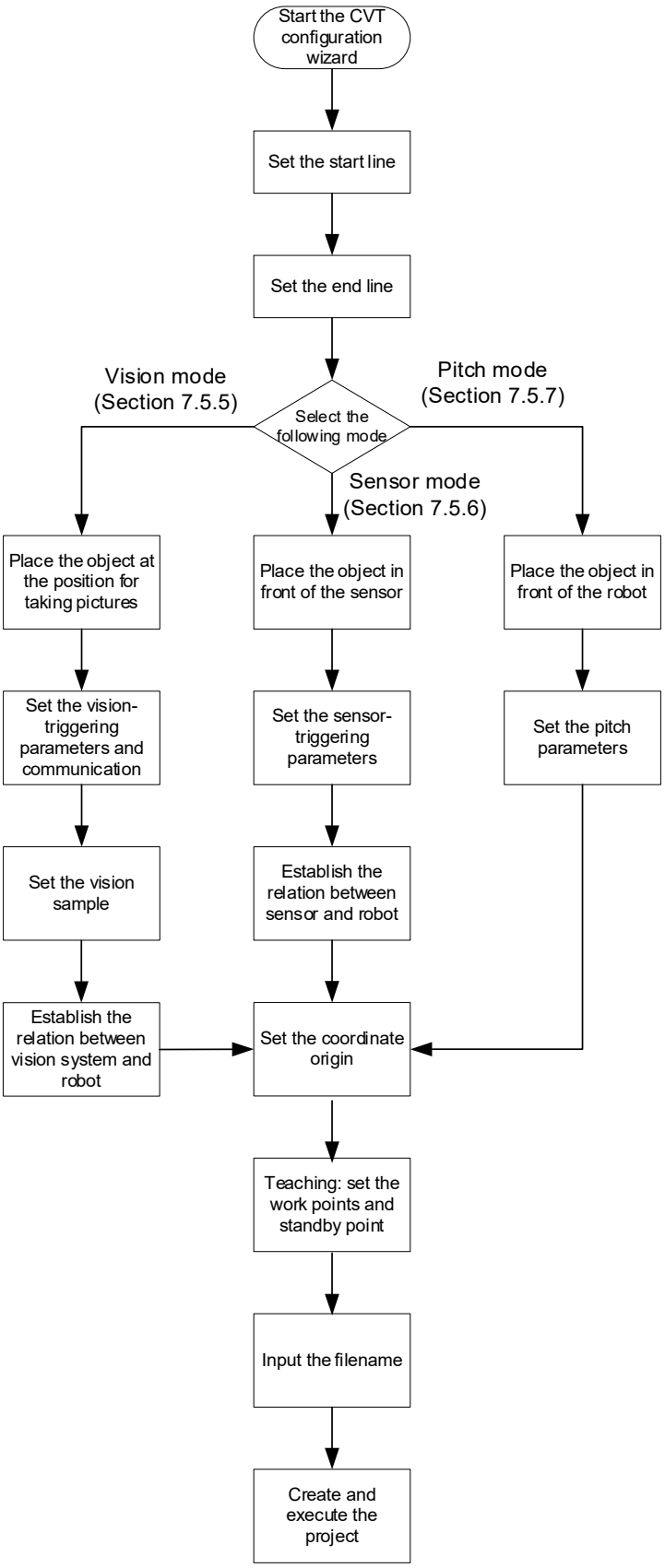
This DO number cannot be the same as the DO number of the internal trigger mode set by CVT_SetUserDefineDO.

Example description

```
CVT_SetOverEndLineDO (1, 10)        --Set the over-line DO output of this robot to DO10
```

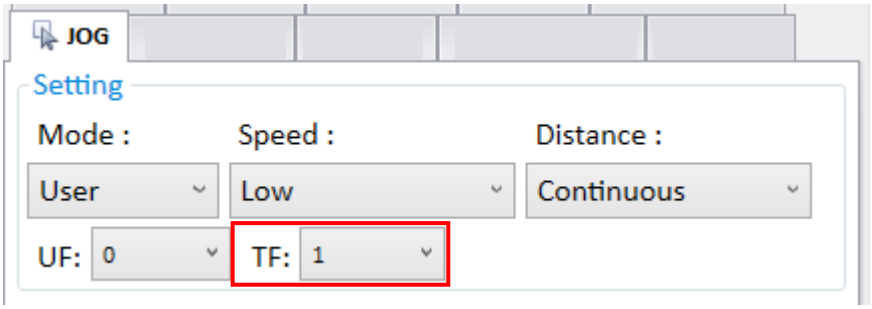
5. CVT configuration wizard

The following is the setup process of the CVT configuration wizard. For detailed setup steps, see sections 7.5.1 - 7.5.7.

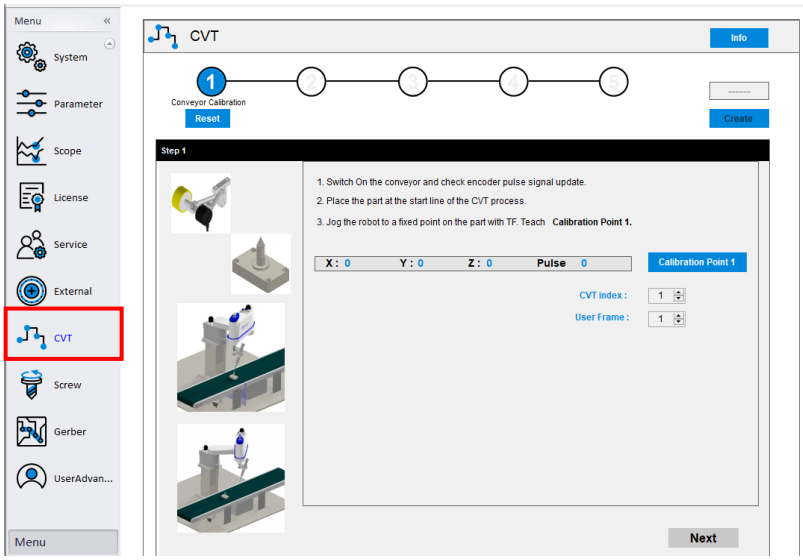


5.1 Open the configuration wizard

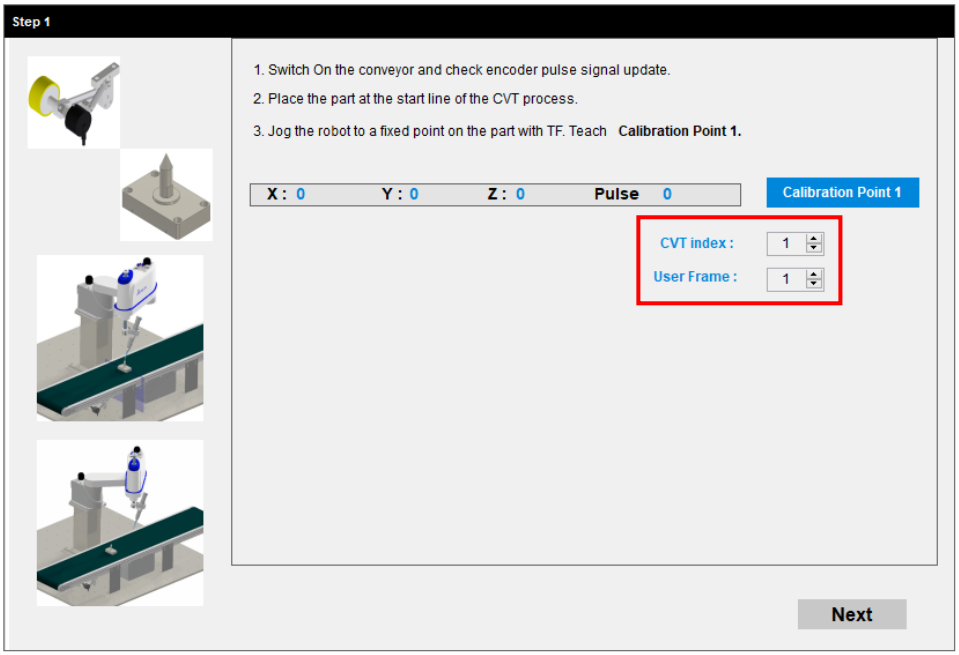
1. First open the DRAStudio software, calibrate the robot tool coordinates TF, and select to use this group of TF in the field to complete all the following actions; this example uses TF1.



2. Go to [Menu], open the [CVT] page, and run the configuration wizard.



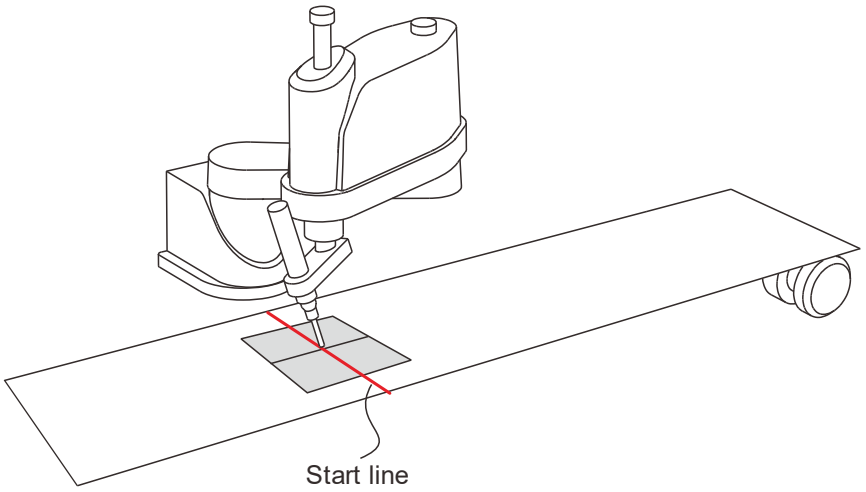
3. Set [CVT index] and [User Frame].

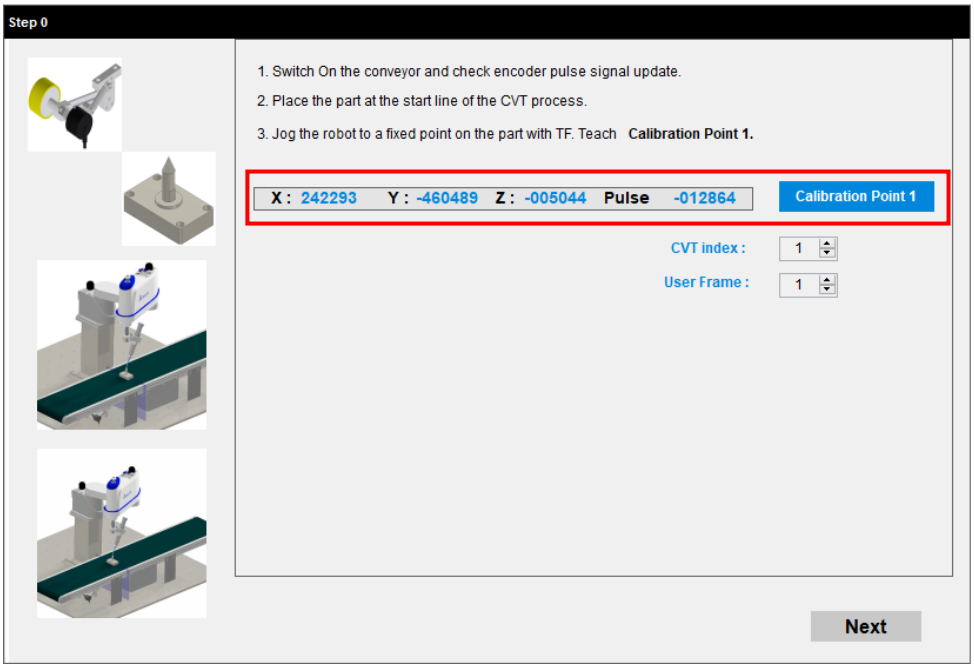


5.2 Configure tracking start line

The tracking start line defines the position from which the robot starts tracking the object. Once the object has gone over this line, the robot starts tracking.

1. Place the object on the conveyor belt and move the robot to the feature point of the object.
2. Press [Calibration Point 1]; this position is the default tracking start line.
3. Press [Next Page].

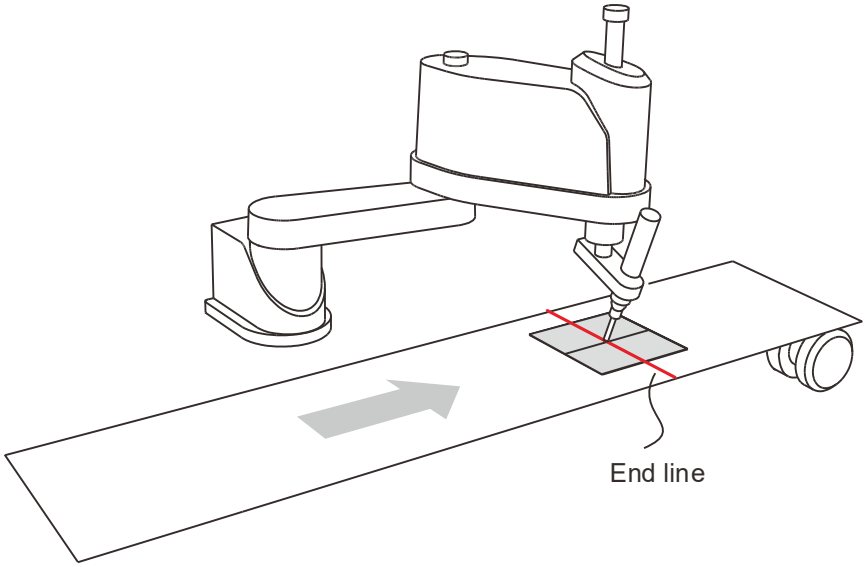


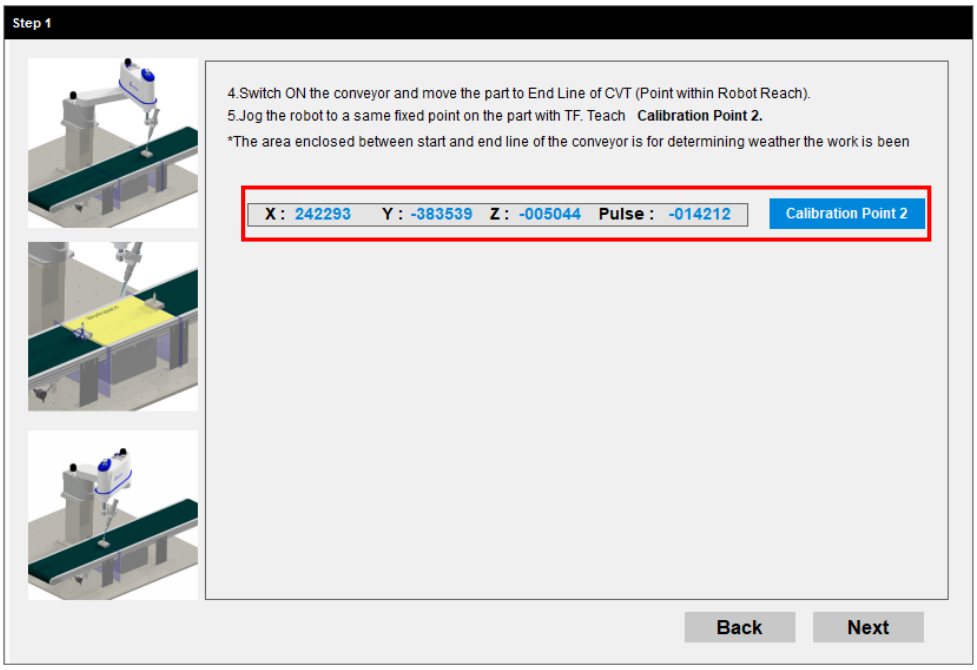


5.3 Configure the tracking end line

If the object has not started to be tracked and goes over this end line, the robot will not track. If the object starts to be tracked before the end line is gone over, the end line is ignored.

1. Activate the conveyor belt to bring the object to the preset processing end position.
2. Move the robot to the feature point of the object (same feature position as the previous step), and then press [Calibration Point 2], this position is the preset end line to track.
3. Press [Next Page].

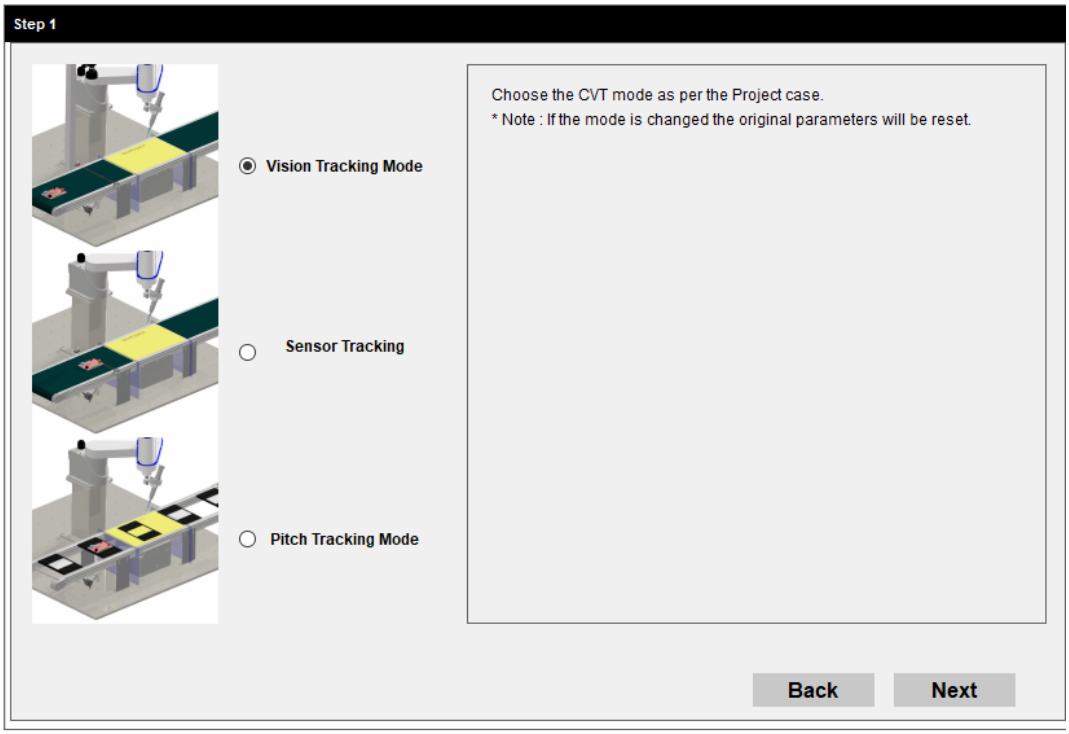




5.4 Select tracking mode

Select the tracking modes according to the system requirements:

- Vision Tracking Mode
- Sensor Tracking Mode
- Pitch Tracking Mode

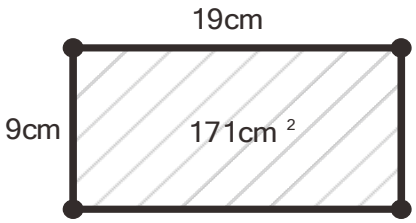


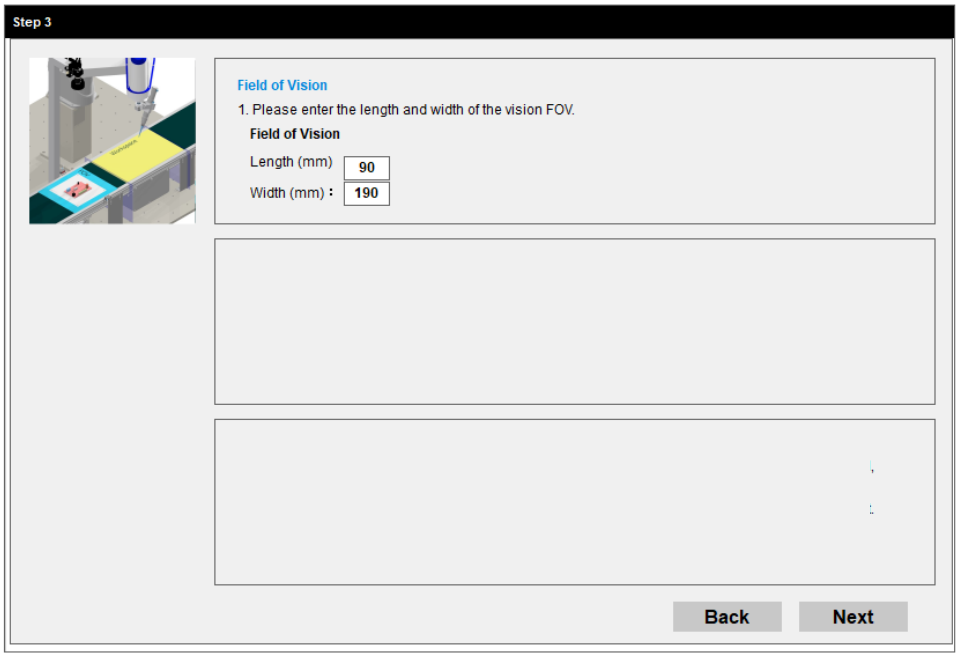
5.5 Vision Tracking Mode

The system architecture of this mode includes a vision system. The descriptions in this section apply whether or not sensors are added.

1. Vision Environment Settings

   1.1 First, measure the field of vision (FOV) and input the dimensions.

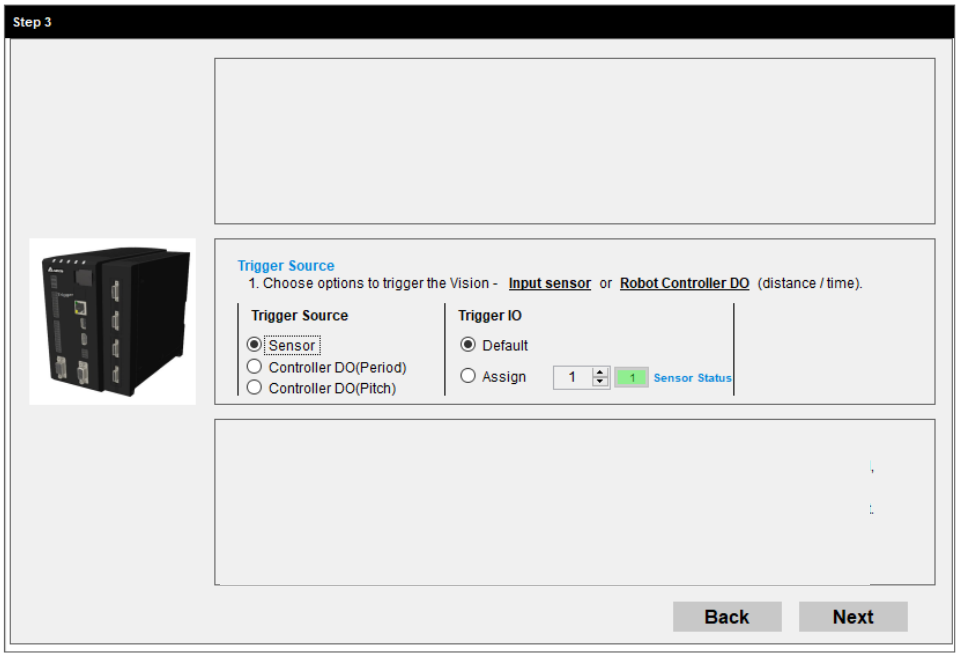




1.2 Select Trigger Source

■ **Sensor**

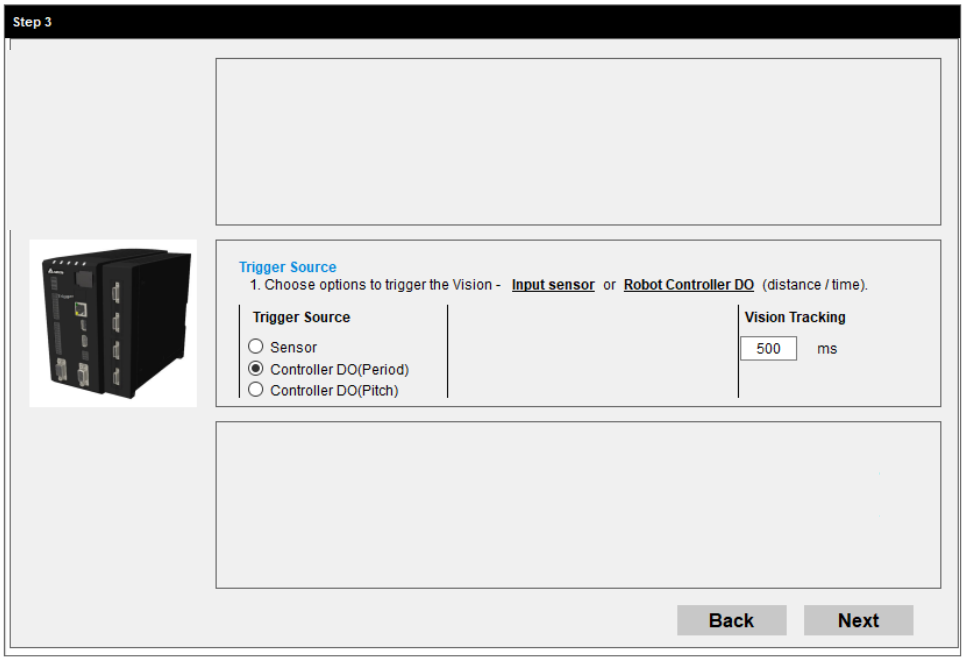
If the sensor is selected as the trigger source, after the sensor receives the signal, it triggers the vision camera and the robot system at the same time. For system configuration, see section 7.2.1 One robot works with one conveyor belt. The trigger IO is the DI pin corresponding to the robot controller. The DI pin can be specified in the range of 1 - 24, and the default value is 1.



■ Controller DO (Period)

If Controller DO (Period) is selected, the robot controller will trigger the vision system to take pictures at a fixed time interval.

The trigger IO is the DO pin corresponding to the robot controller. The DO pin can be specified in the range of 1 - 12, and the default value is 1. The user can set the interval trigger time in [Vision Tracking], which must be greater than the vision processing time plus the communication timeout time in units of ms.

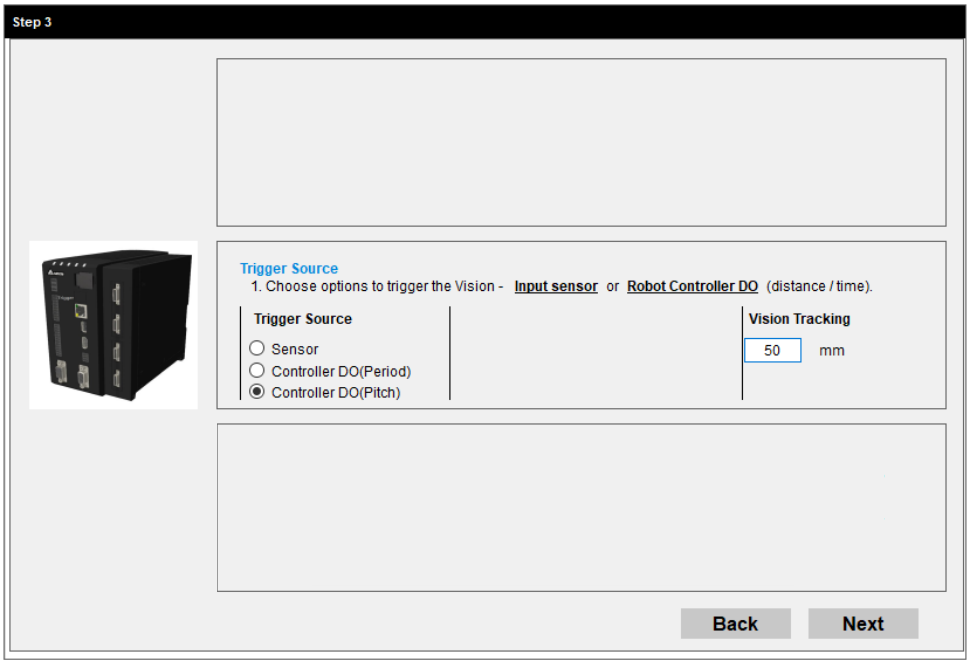


■ Controller DO (Pitch)

If Controller DO (Pitch) is selected, the robot controller triggers the vision system to take pictures at a pitch.

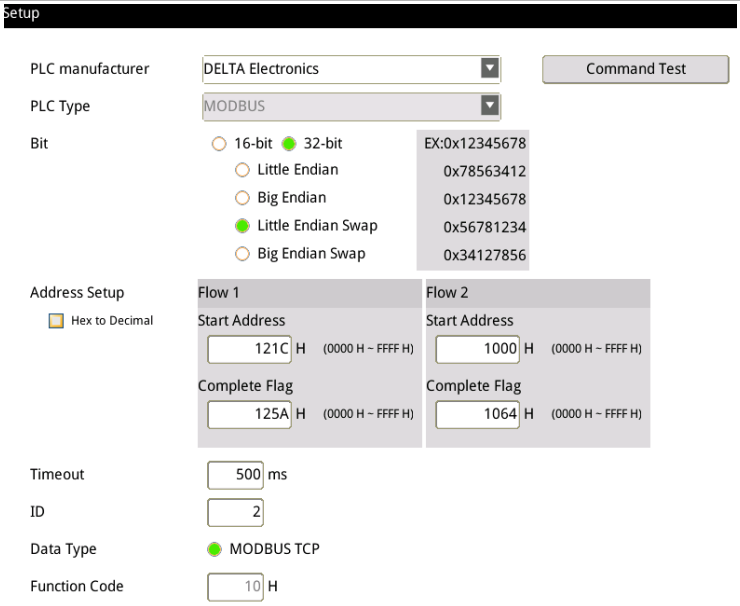
The trigger IO is the DO pin corresponding to the robot controller; can be specified in the range of 1 - 12, and the default value is 1.

[Vision tracking] is for setting the interval triggering distance. The interval distance should not be set too short, since the vision system needs to be able to process recognition in units of mm.



1.3 Set the communication output format (this example uses DMV3000G TCP Modbus)

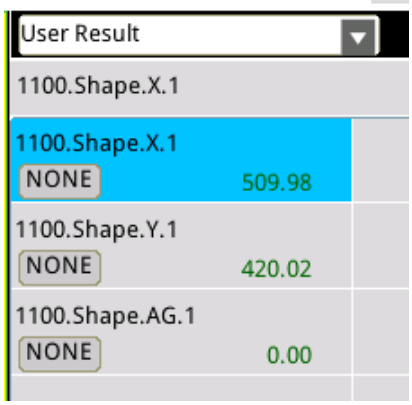
- In Bit option, select 32-bit.
- Set the timeout time to 500 ms (setting the initial value to be 500 ms is recommended; it may be adjusted depending on the project).
- Set the Modbus communication address; the starting address is 121C, and the done flag storage address is 125A.
- Set the station ID to 2.

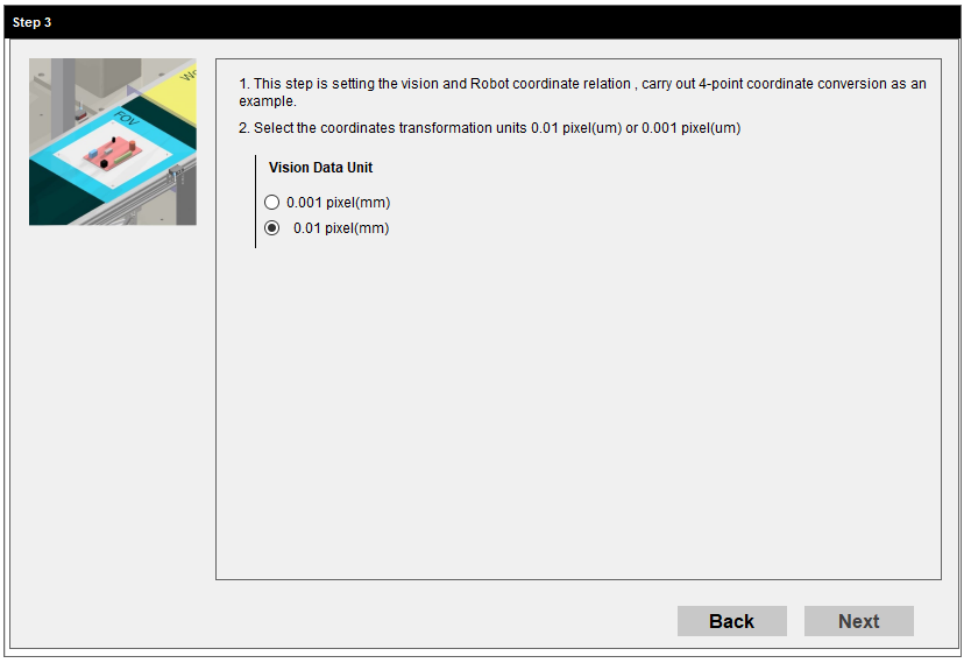


1.4 Press [Next Page].

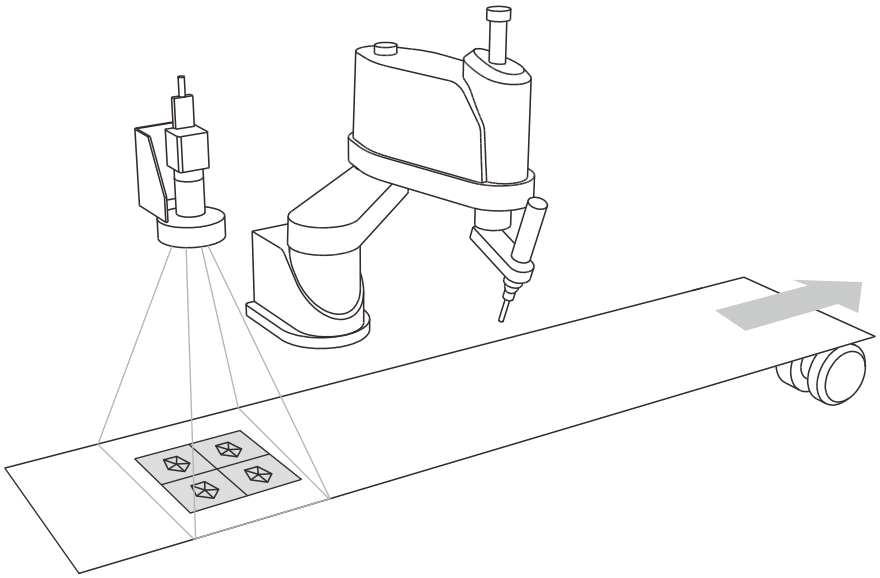
2. Configure the vision data

2.1 In [Vision Data Unit], select the corresponding units. If the output of the example DMV3000G is in two decimal places, select [0.01 pixel].



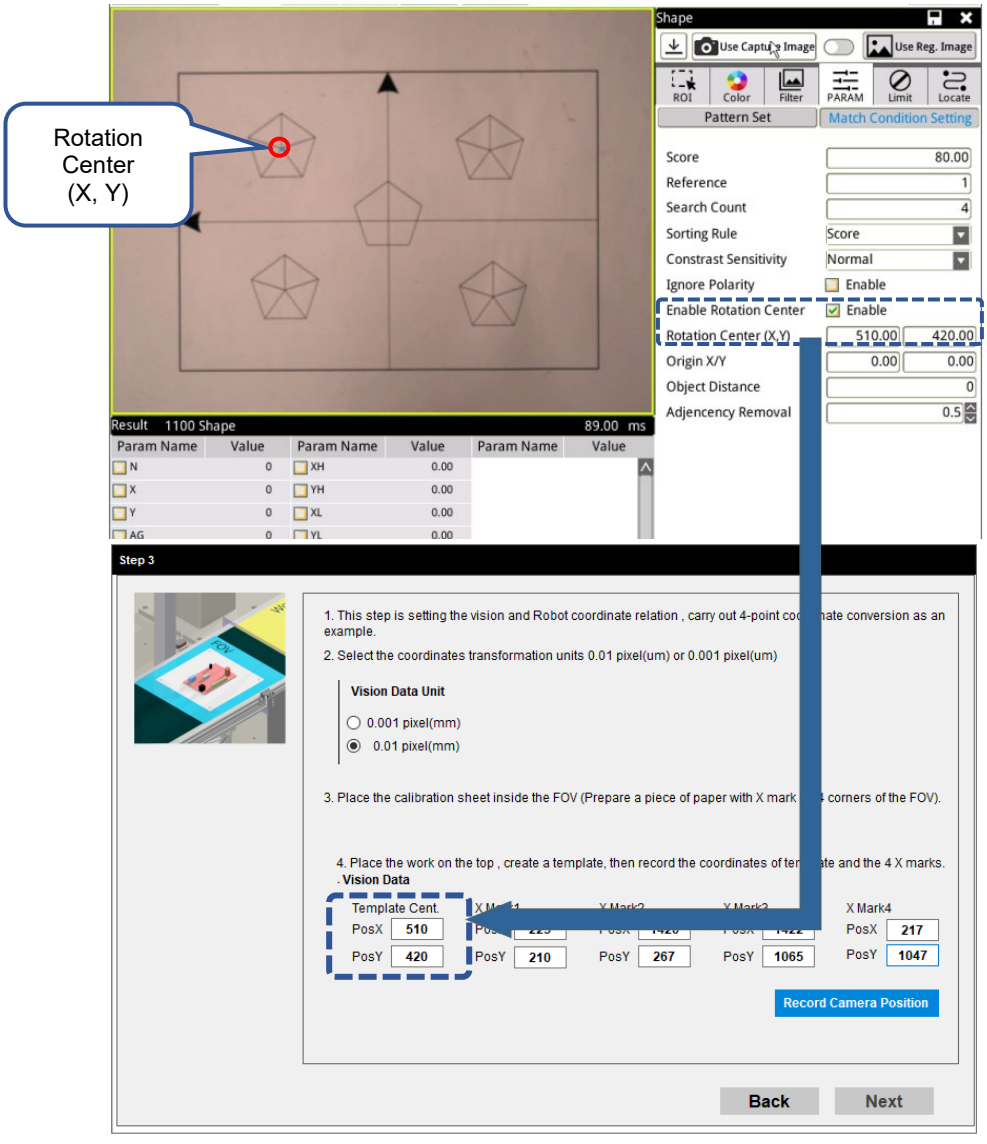


2.2 The vision sample is placed on the conveyor belt and moved into the field of vision of the vision system.

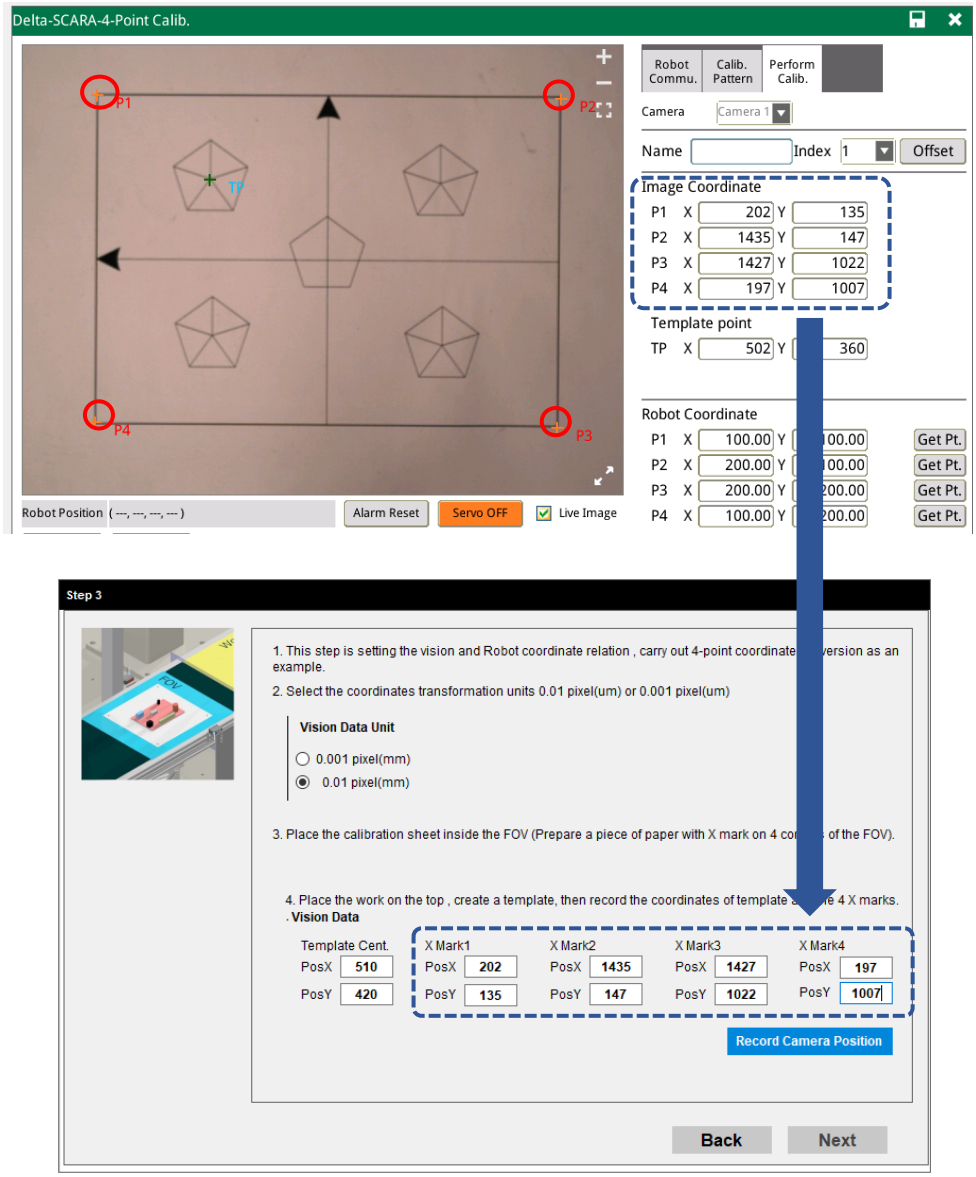


2.3 Build the vision samples.

2.4 Set a custom rotation center (X, Y) for the sample and enter pixel values in the UI.



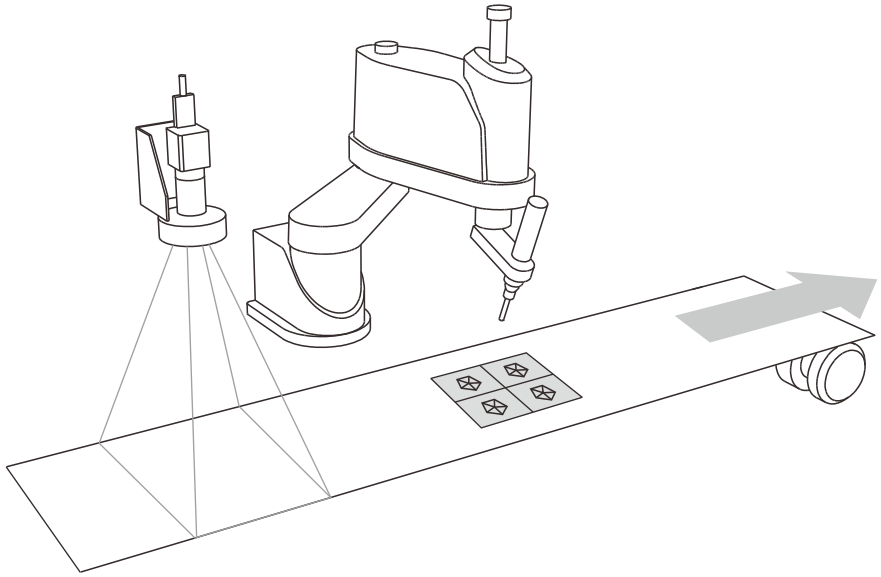
2.5 Get the positions of 4 points on the paper and input their coordinates to the UI.



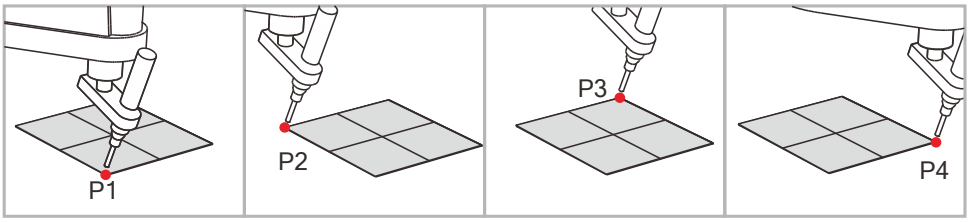
2.6 Click [Record Camera Position].

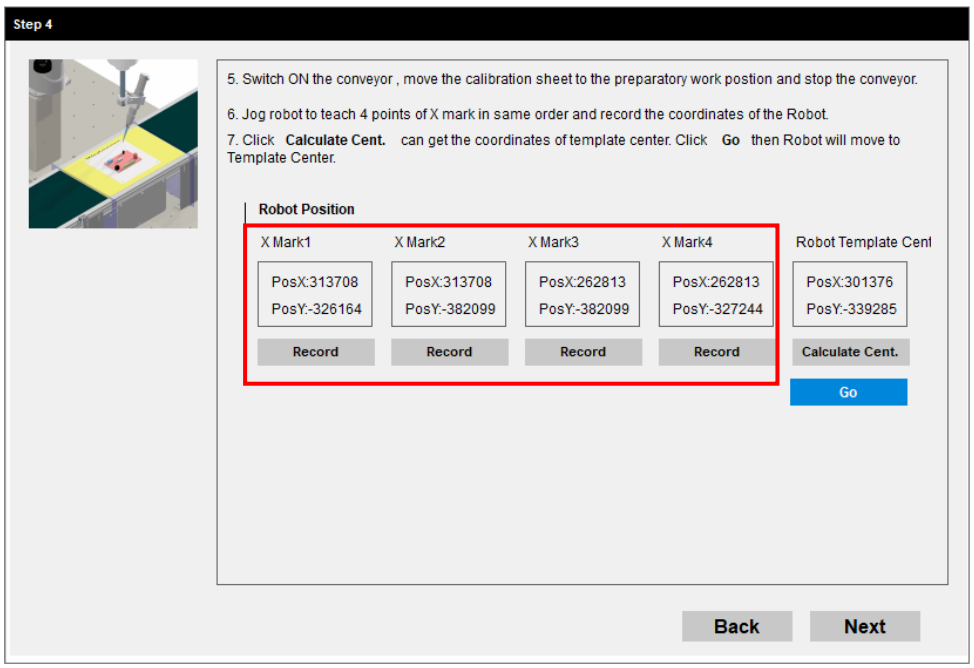
3. Set the conversion relationship between vision and robot.

3.1 Start the conveyor belt and move the four-point calibrated paper and sample into the working area.

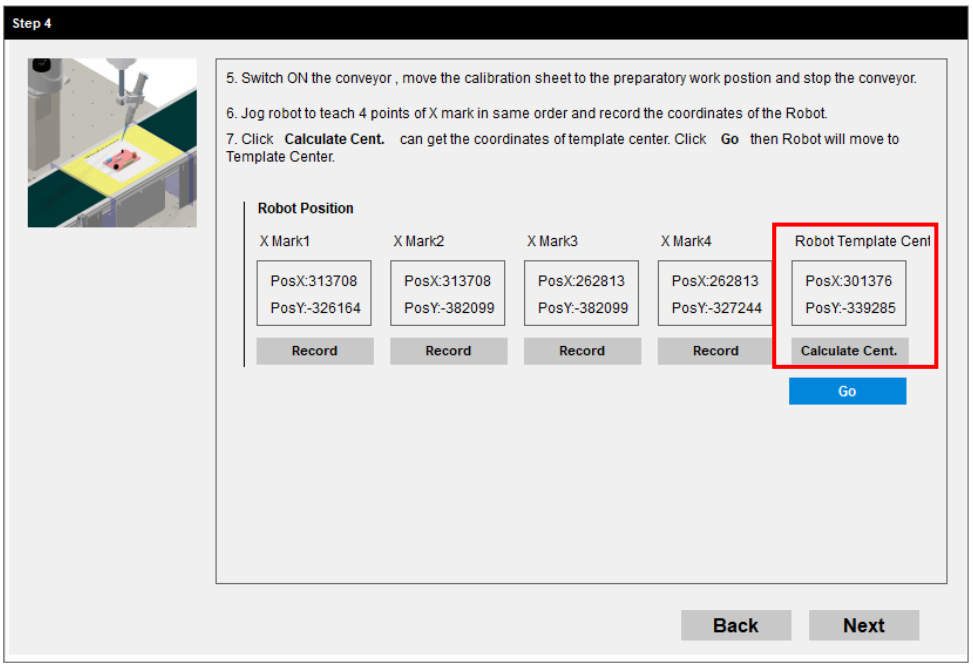


3.2 Refer to Step 2.5, move the robot to the four positions in sequence and click [Record].



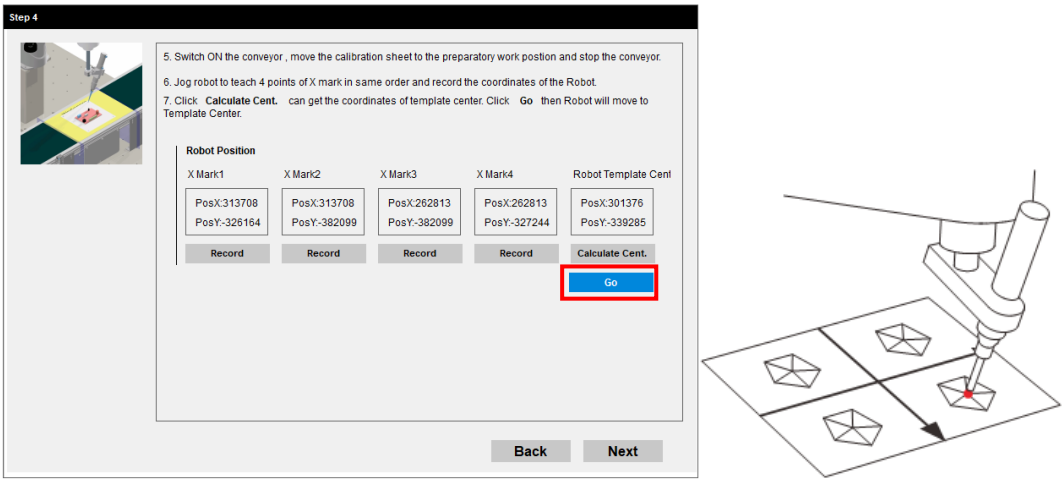


3.3 Click [Calculate Cent.].



3.4 Click [Move], and the robot will move to the center point of the object; the position is the same as the custom Rotation Center (X, Y).

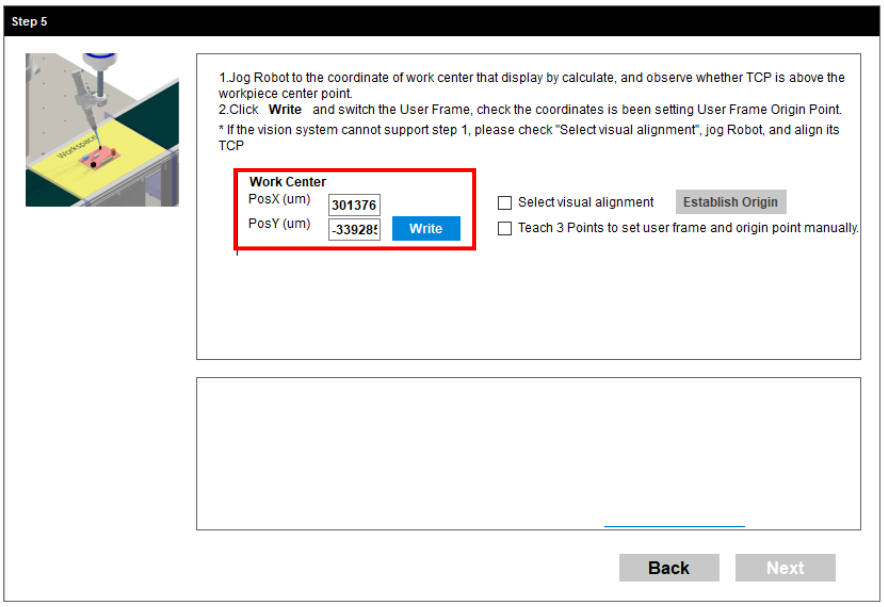
Important: pay attention to the height of the end of the robot when moving, and confirm whether it contacts the object.



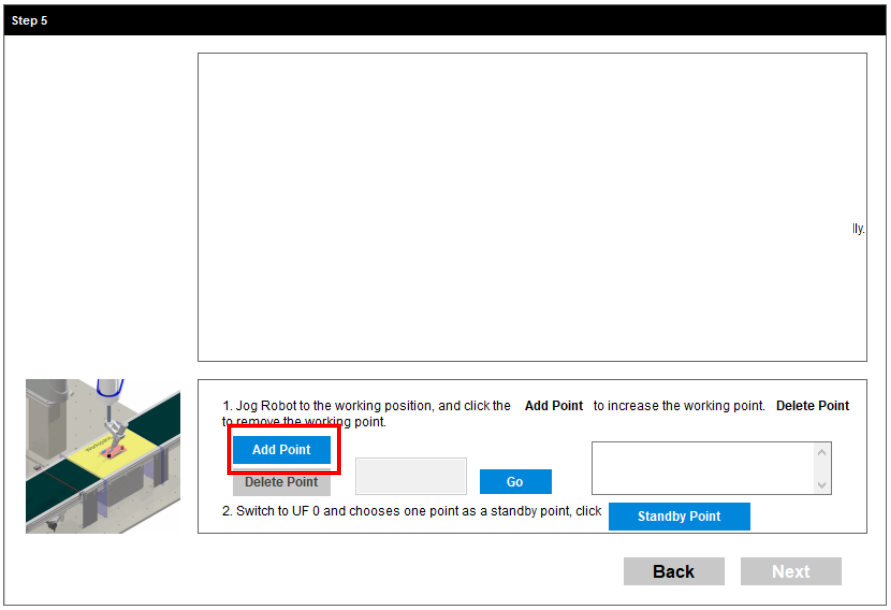
3.5 Click to go to the next page.

4. After the robot successfully moves to the custom Rotation Center (X, Y) of the object, the value is automatically input to the field in the UI.

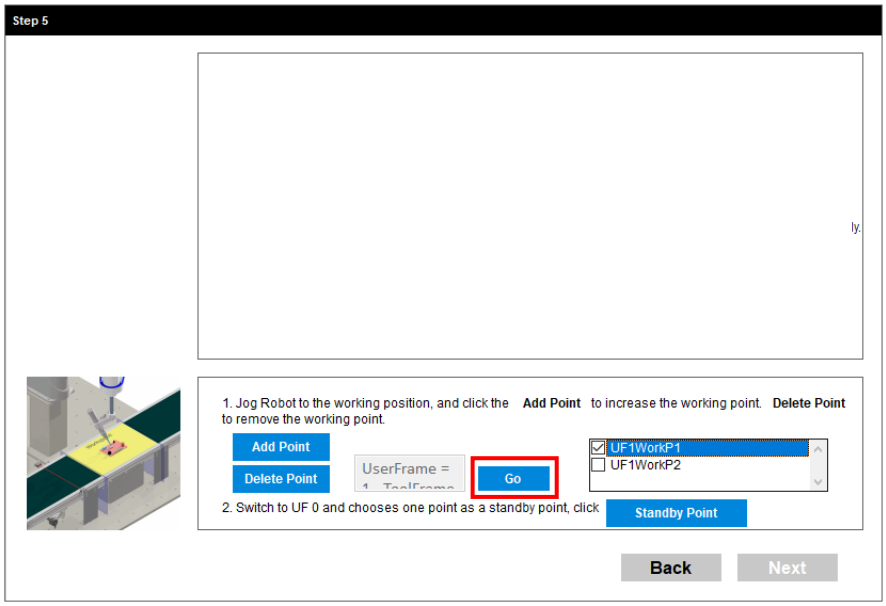
5. After confirming the value, click [Write].



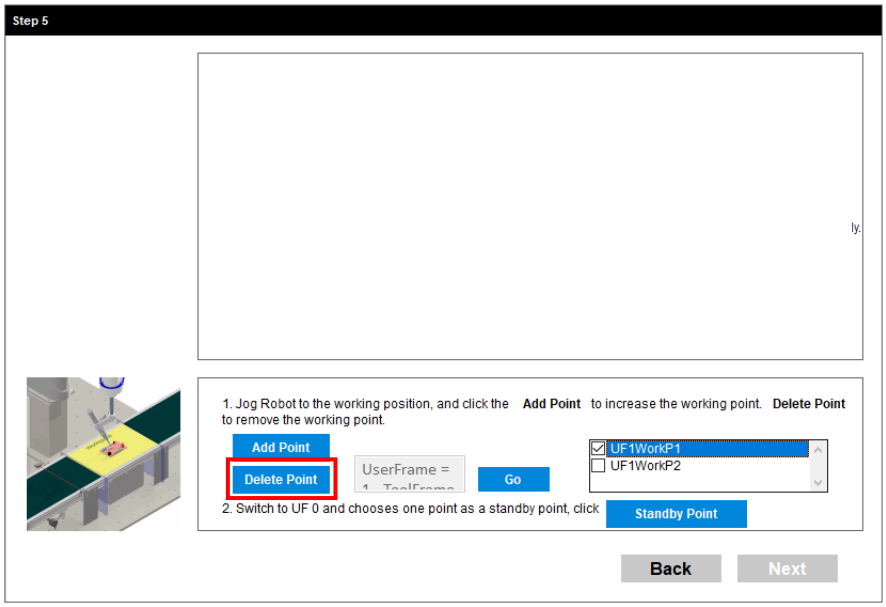
6. Move the robot to the work point, and click [Add Point] to teach the work points in sequence.



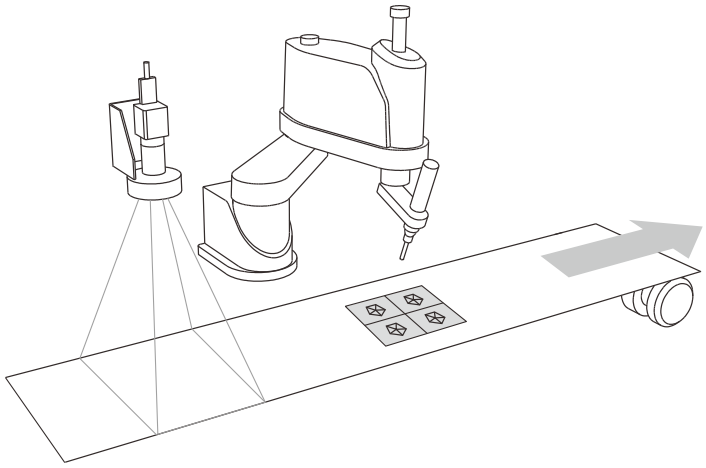
7. To confirm a point, click on the point, and then click [GO] to move to the corresponding point.



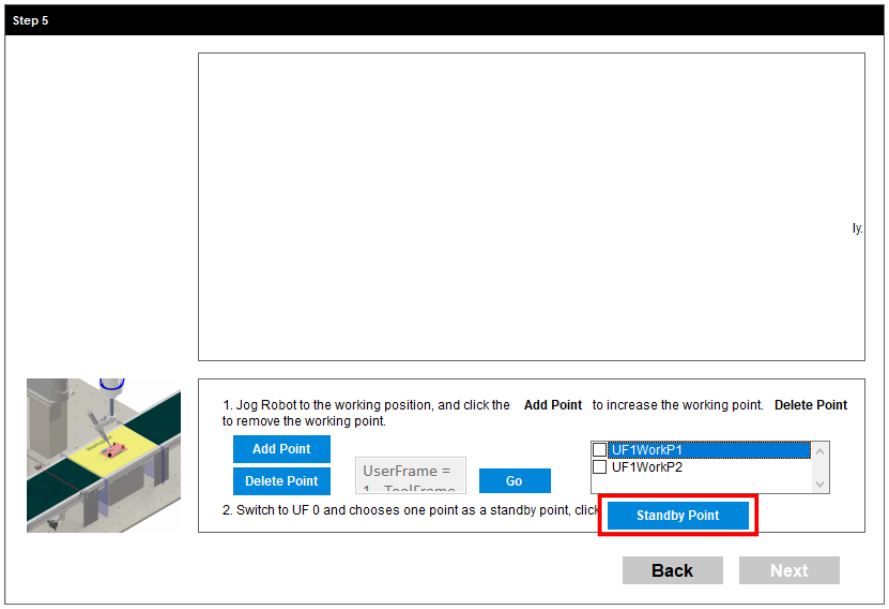
8. If the teaching is wrong and the point needs to be deleted, you can select the point and click [Delete Point].



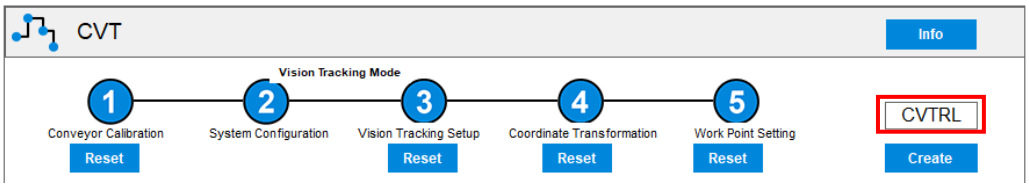
9. Move the robot to the standby position and change the User Frame to UF 0.



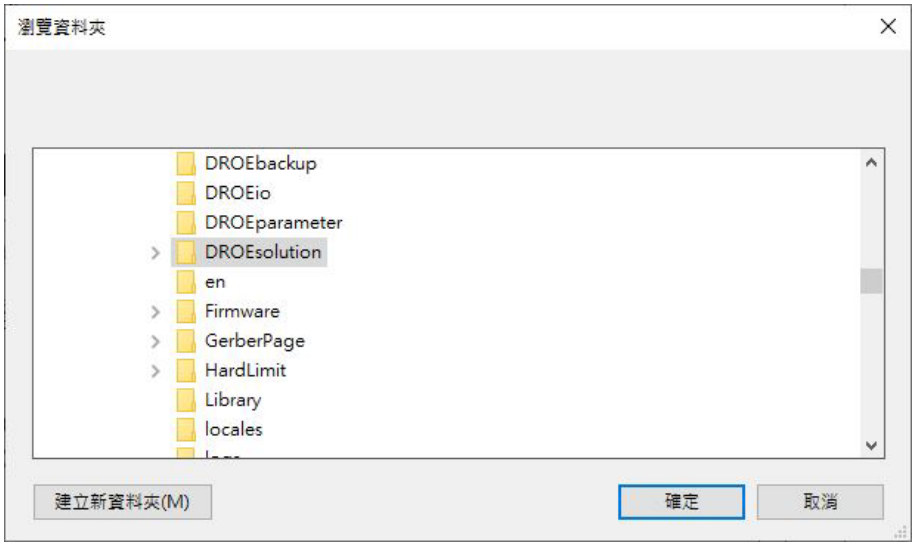
10. Click [Standby Point].



11. In the upper right [Project Title] field, fill in the title of this project.



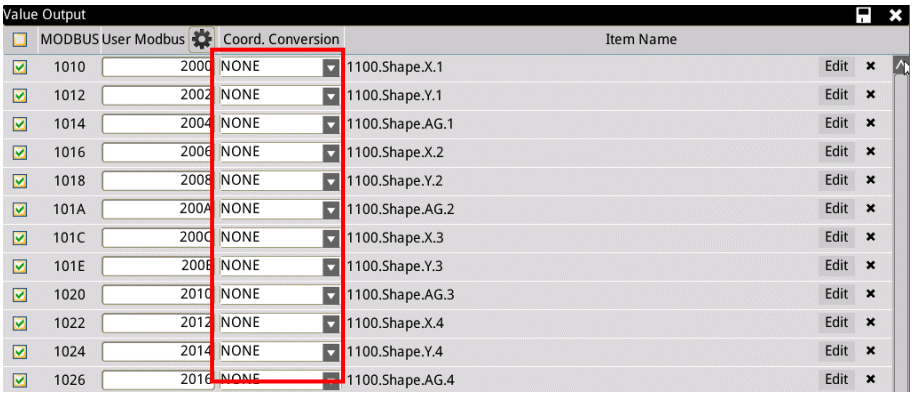
12. Click [Create] to create a project and save it to the [DROEsolution] folder.



13. Configure vision output data.

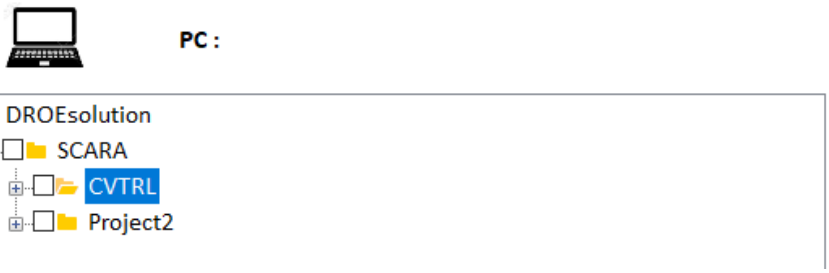
13.1 Add the object information in sequence. You can enter 10 sets of object information, the order is as follows:

- The coordinate value of the object in the X direction (Double Word length)
- The coordinate value of the object in the Y direction (Double Word length)
- AG angle rotation value of the object (Double Word length)



Important: you must select [NONE] for [Coord. Conversion].

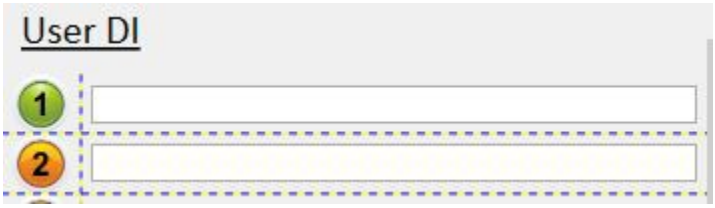
14. Open the project and execute the program.



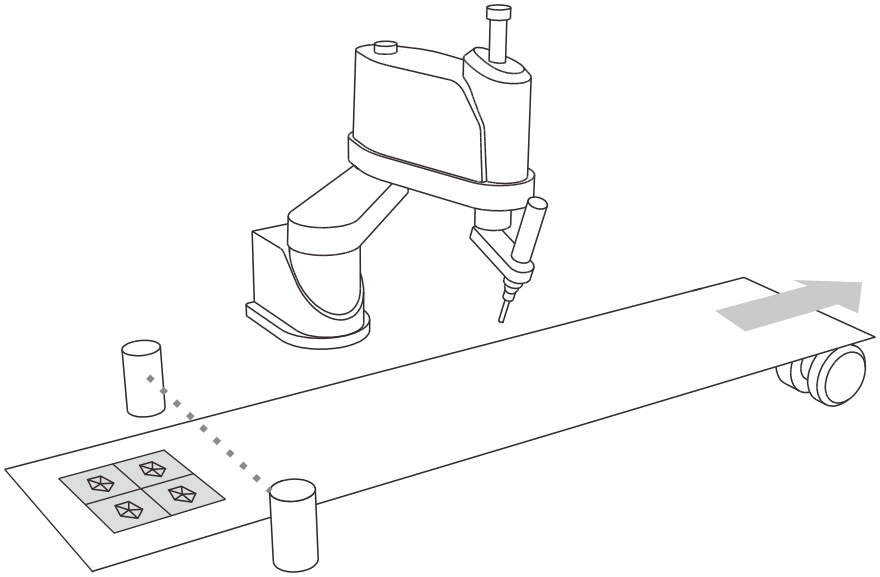
5.6 Sensor Tracking Mode

The system architecture of this mode includes only sensors and no vision system.

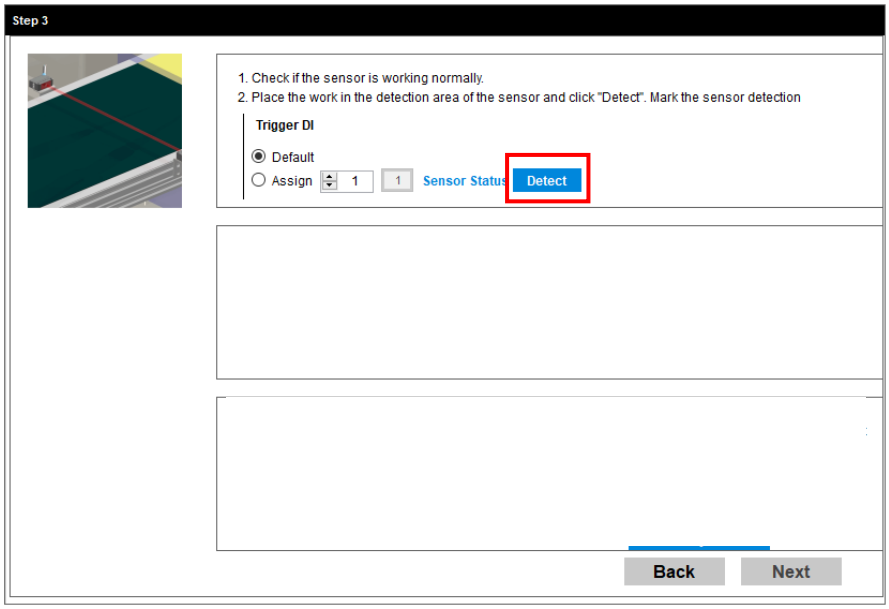
1. Check the sensor. Check whether the robot controller has received the signal correctly when the sensor detects the signal.



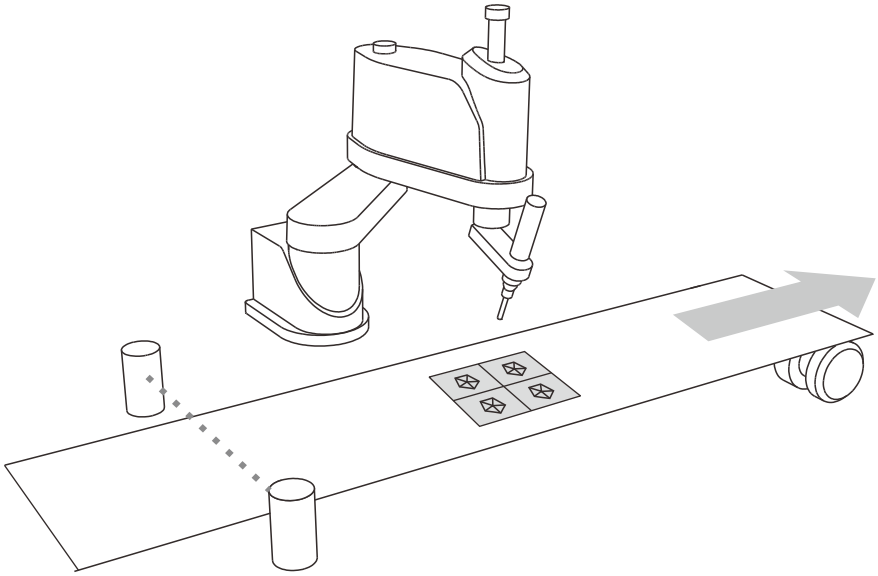
2. Place the object on the conveyor belt before the sensor.



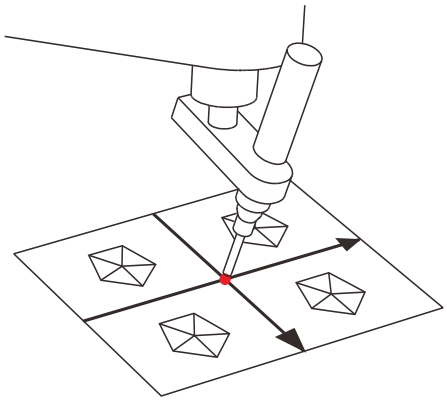
3. Set the DI point for the sensor to receive the signal; the default is 1.
4. Click [Detect].



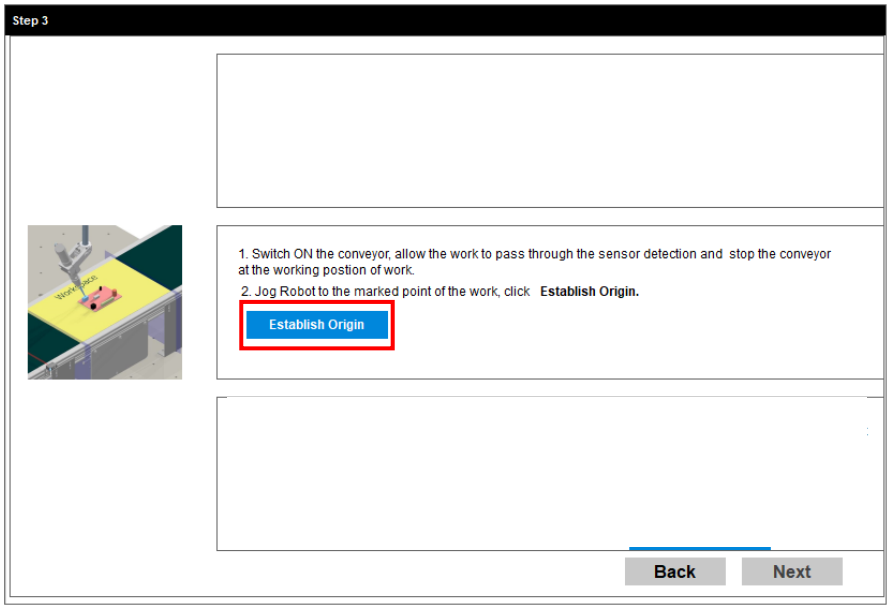
5. Start the conveyor belt so the object passes the sensor and stops in front of the robot.



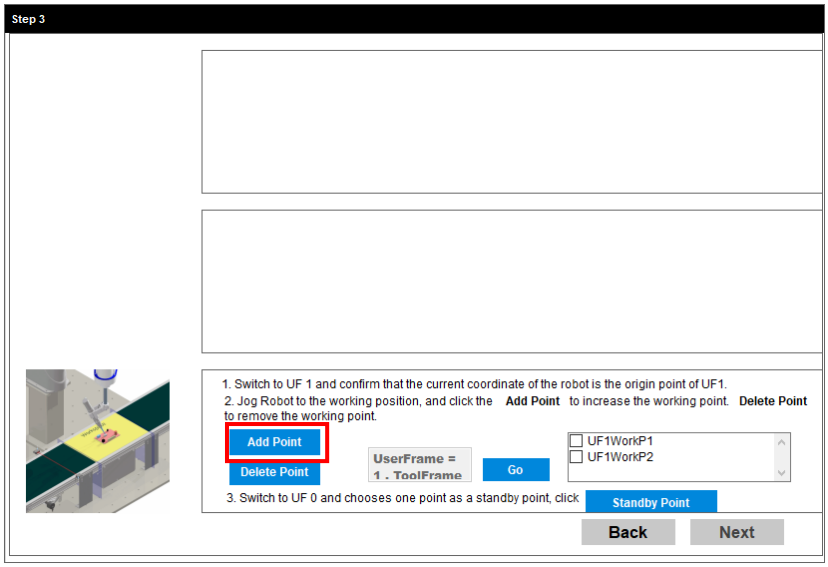
6. Using the corresponding tool frame, move the robot to the center point of the object.



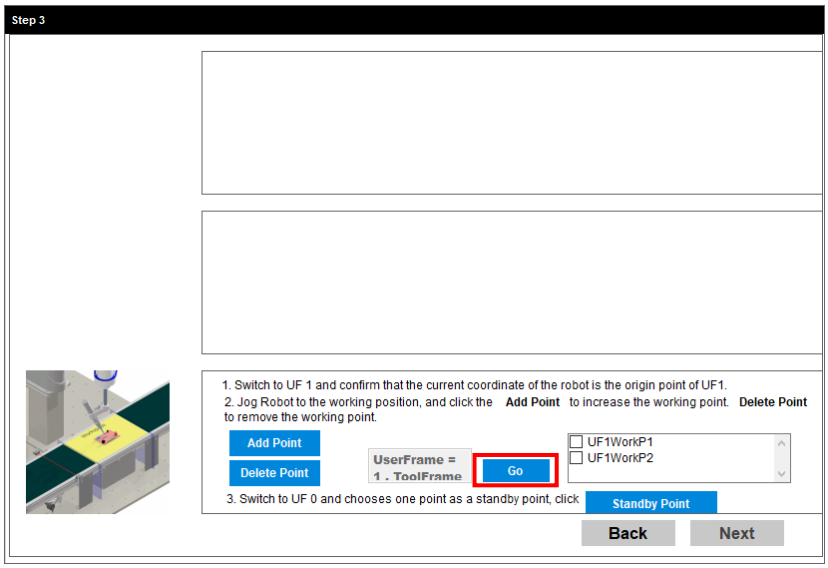
7. Click [Establish Origin].



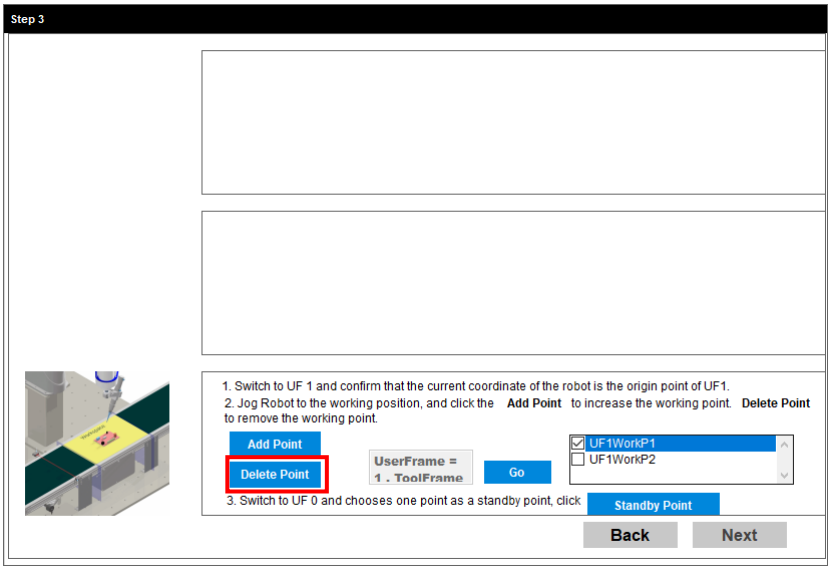
8. Move the robot to the work point, and click [Add Point] to teach the points in sequence.



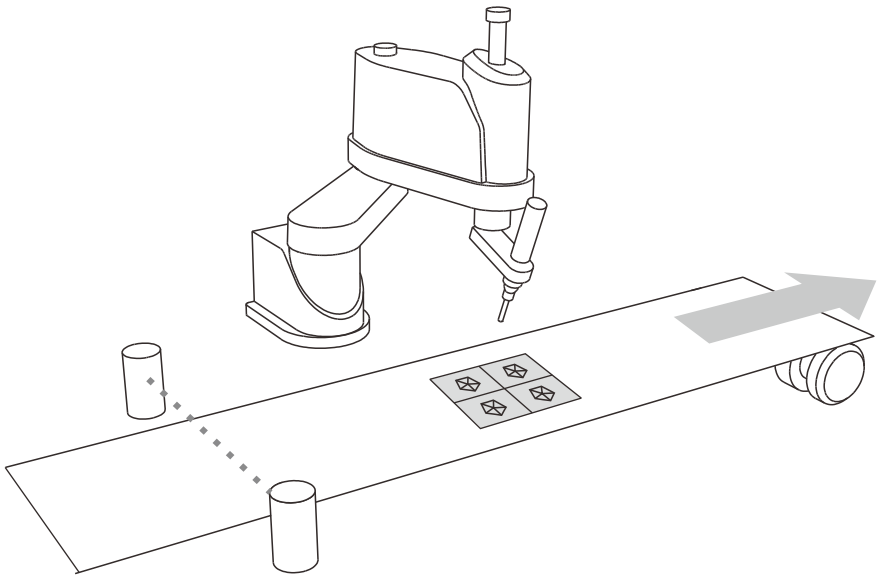
9. To confirm a point, click on the point, and then click [GO] to move to the corresponding point.



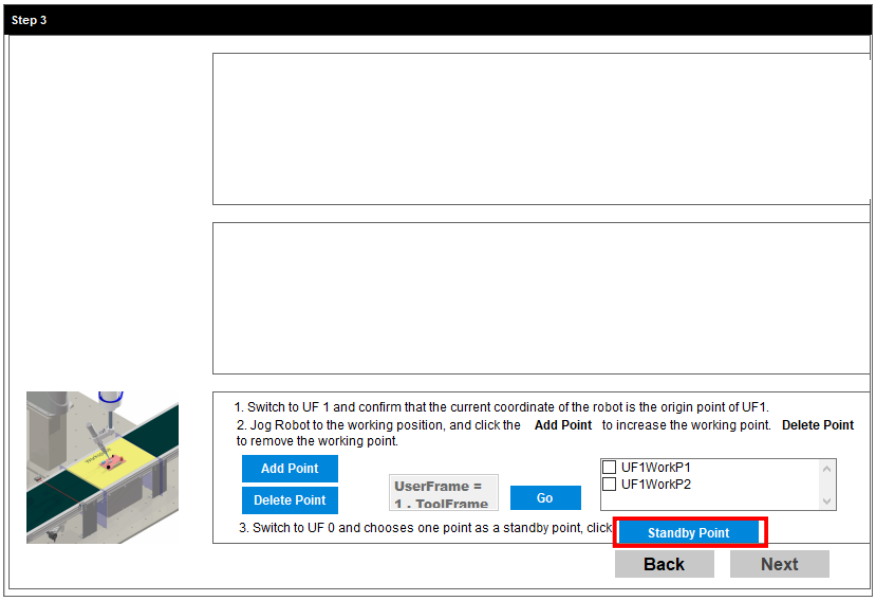
10. If the teaching is wrong and the point needs to be deleted, you can select the point and click [Delete Point].



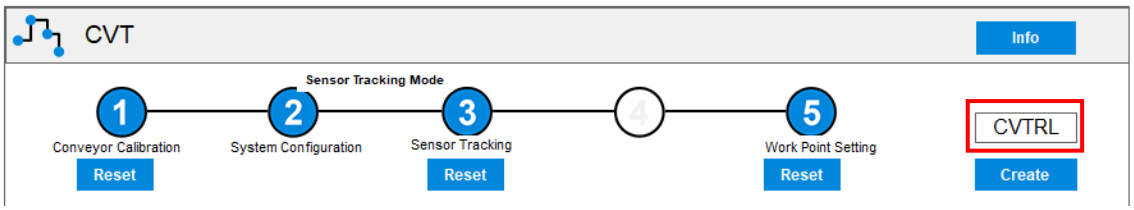
11. Move the robot to the standby position and change the User frame to UF 0.



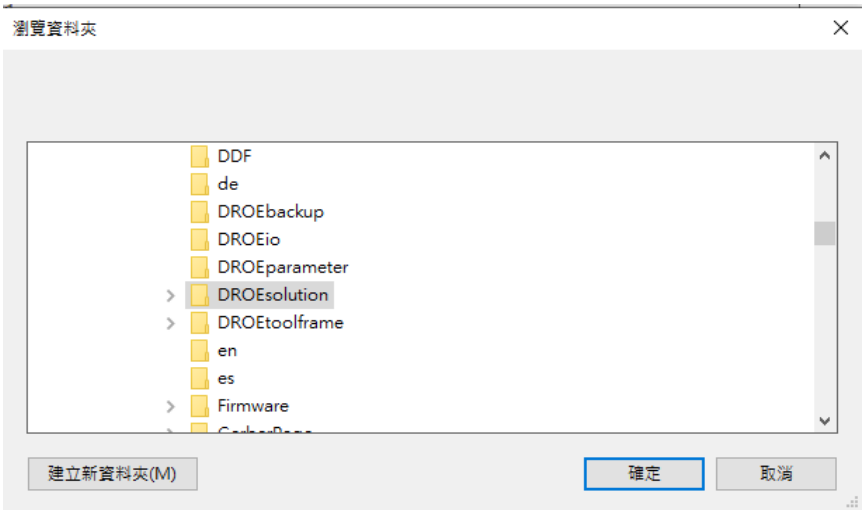
12. Click [Standby Point].



13. In the upper right [Project Title] field, fill in the title of this project.

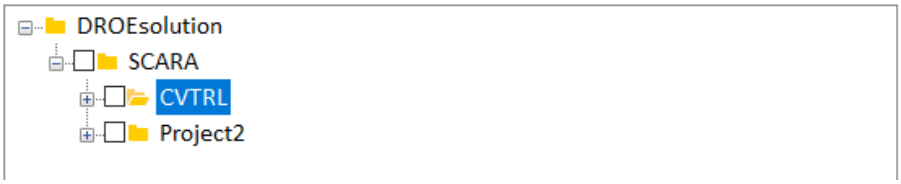


14. Click [Create] to create a project and save it to the [DROEsolution] folder.



15. Open the project and execute the program.

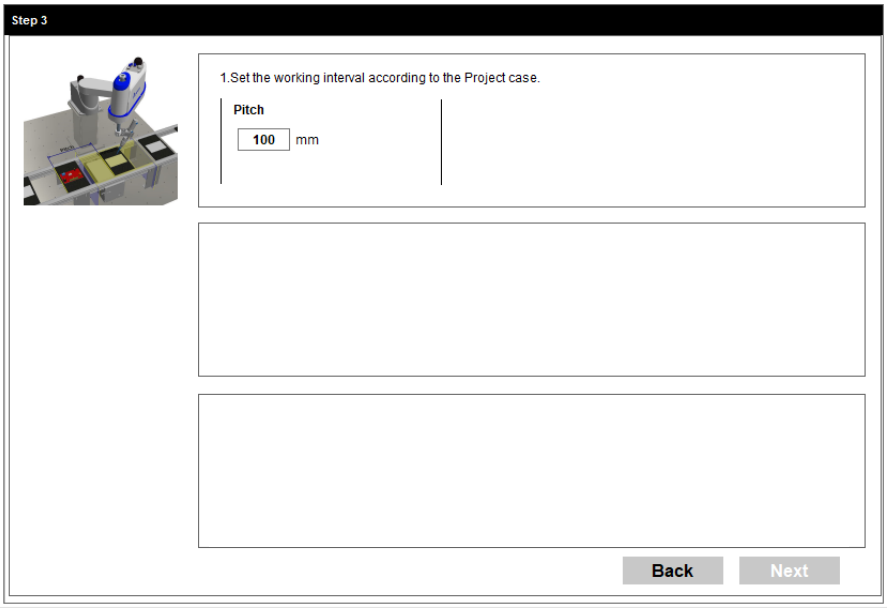
PC :



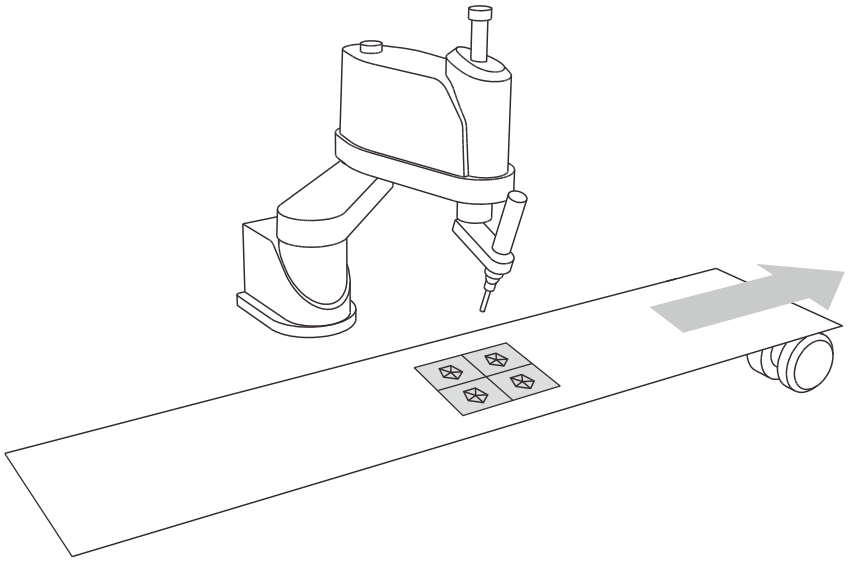
5.7 Pitch Tracking Mode

The system architecture of this mode is that no vision system and sensors are added, only a fixed pitch is tracked.

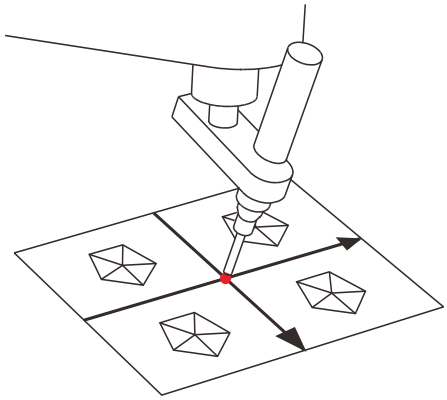
1. In [Pitch], input the distance between the operation objects.



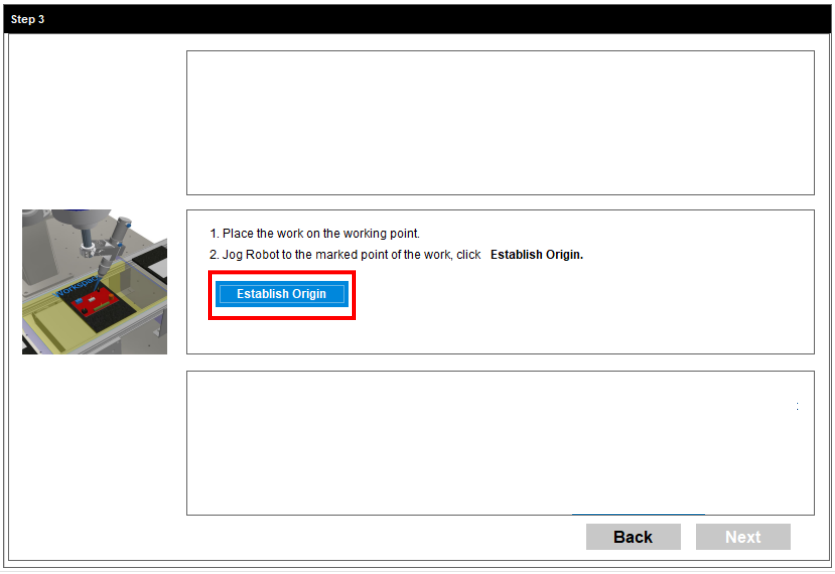
2. Place the object in front of the robot.



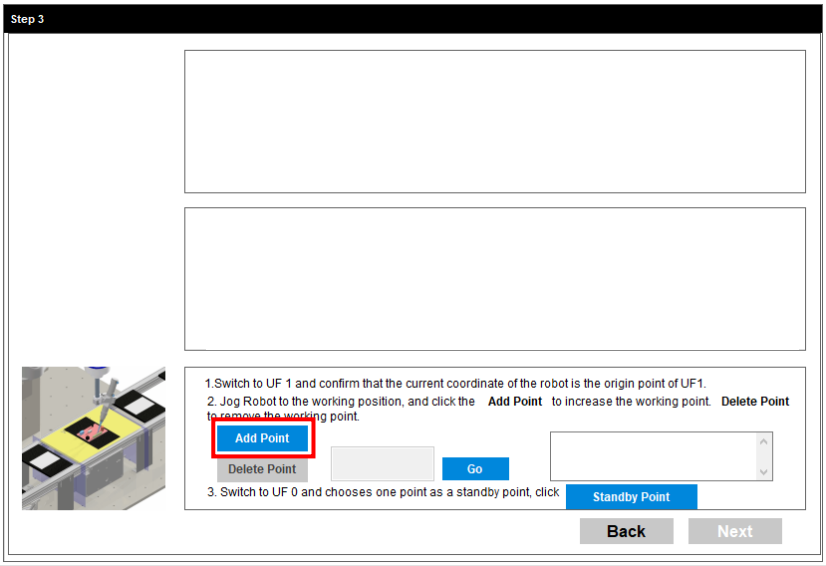
3. Using the corresponding tool coordinates, move the robot to the center point of the object.



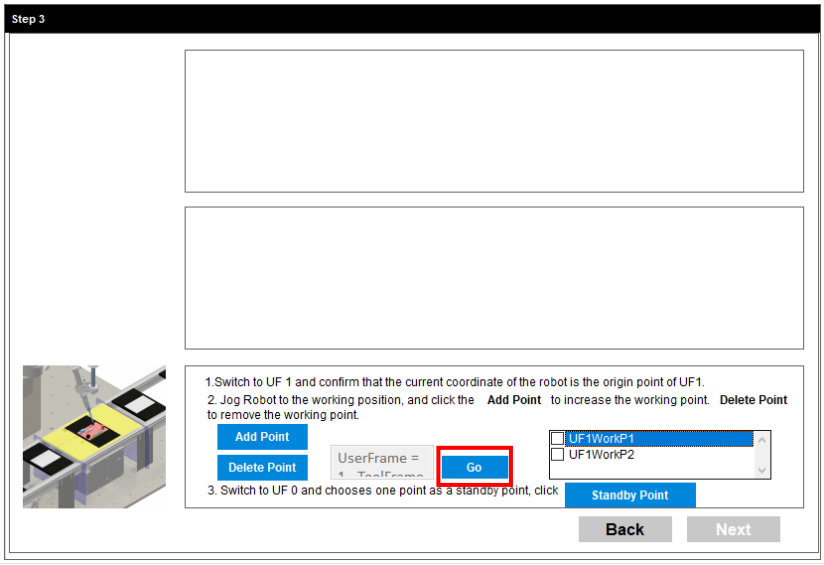
4. Click [Establish Origin].



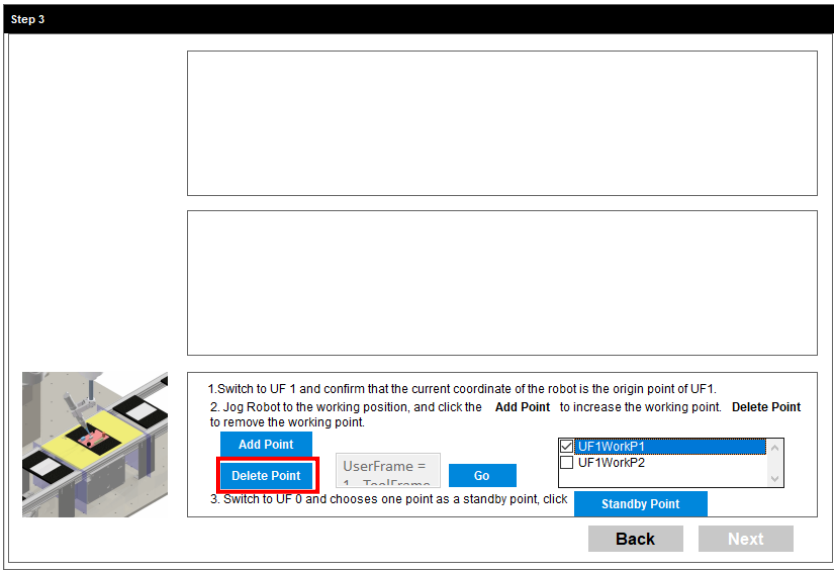
5. Move the robot to the work point, and click [Add Point] to teach the points in sequence.



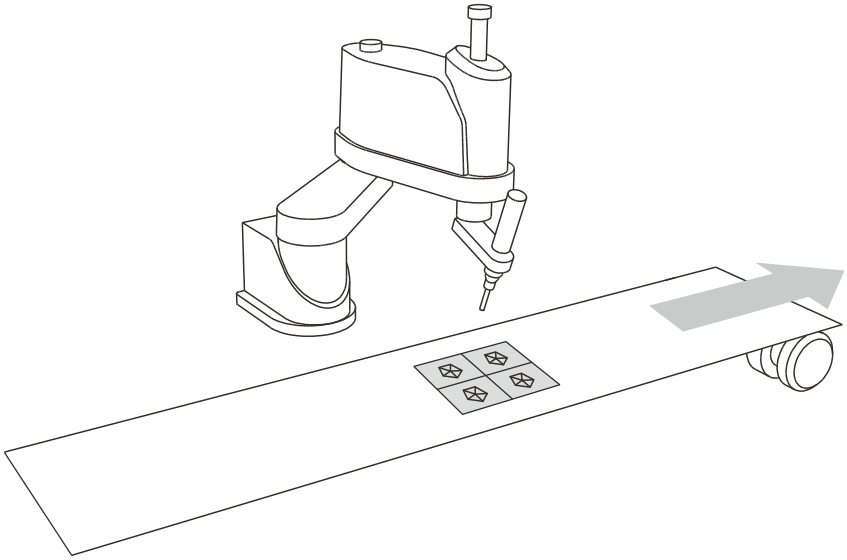
6. To confirm a point, click on the point, and then click [GO] to move to the corresponding point.



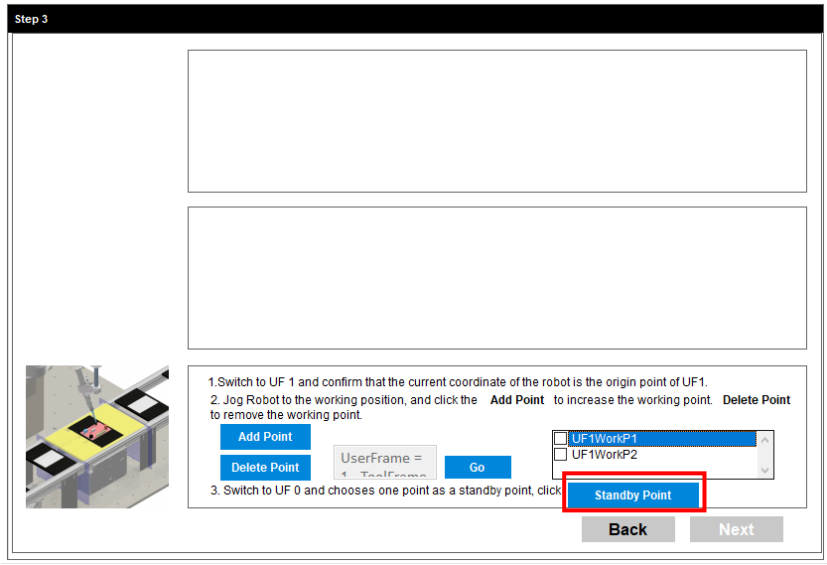
7. If the teaching is wrong and the point needs to be deleted, you can select the point and click [Delete Point].



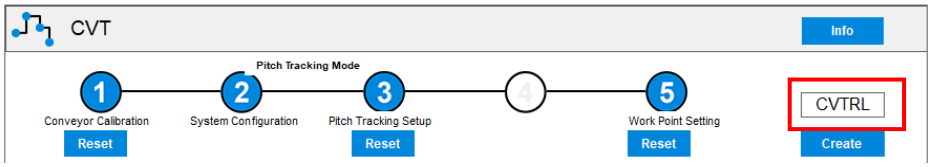
8. Move robot to the standby position and change the User frame to UF0.



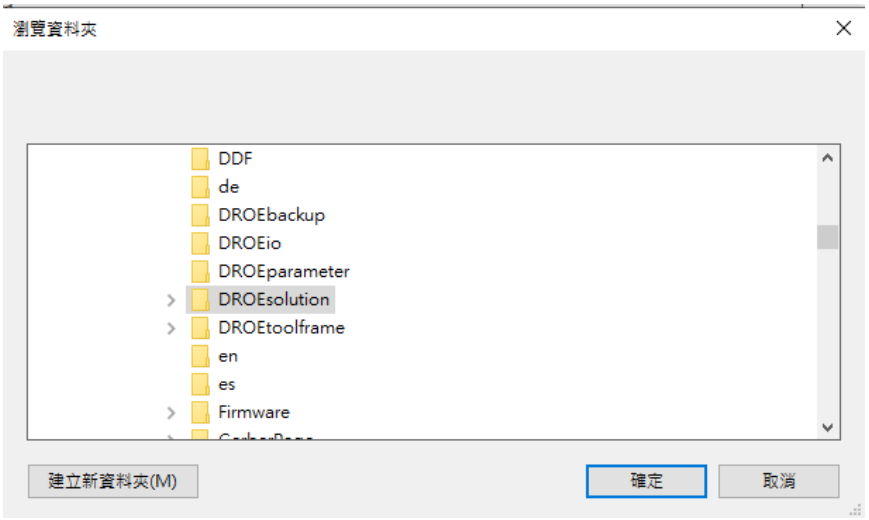
9. Click [Standby Point].



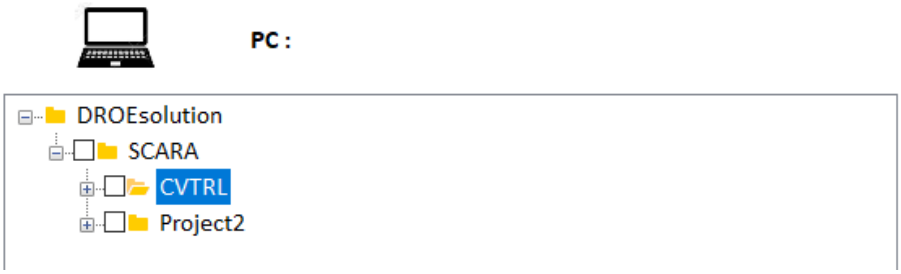
10. In the upper right [Project Title] field, fill in the title of this project.



11. Click [Create] to create a project and save it to the [DROEsolution] folder.



12. Open the project and execute the program.



6. Troubleshooting steps

If the equipment does not operate normally after you configure the devices with the setting procedure, follow the instructions to troubleshoot.

Controller error (program cannot run)	
Cause of problem	1. Firmware not updated to the correct version. 2. The CVT serial number is not input to the controller.
Check and correction steps	1. Check if the firmware version is correct. 2. Open the Delta robot operating software DRAStudio and go to the [Certificate] page, then go to the [DCS/DCV] tab and check whether the CVT serial number has expired.

Conveyor belt direction is wrong	
Cause of problem	An error occurred in the CVT configuration wizard when calibrating the start line and end line (described in Sections 7.5.2 and 7.5.3), resulting in the CVT program being unable to calculate the conveyor belt direction vector correctly.
Check and correction steps	1. Check if the directional vector of the conveyor belt movement is correct. If the conveyor belt and robot are positioned horizontal, the value of the parameter cmpstVectorZ should be set to 0. If the moving direction of the conveyor belt is in the X direction of the robot, the value of the parameter cmpstVectorY should be close to 0. If the moving direction of the conveyor belt is in the Y direction of the robot, the value of the parameter cmpstVectorX should be close to 0.

```
---CVT Template---
CVT_ChangeMotion()
RobotServoOn()
---->CV1Setting<----
CVT_SelectMode(1, 1)--1:With Vision, 2:Without Vision
CVT_SetTriggerMode(1, 2)--1:DO, 2:DI
--Encoder Unit Ratio
cvtFactor_num = 614
cvtFactor_den = 1000
--Encoder AMF window(ms)
interval = 10
--CCD & Robot Transform Offset(um)
trans_ccd_x = 0
trans_ccd_y = -1000000
rotat_ccd_c = 0
--CCD Unit Ratio
vuPix2UmNum = 10
vuPix2UmDen = 1
vuAgRatioNum = 10
vuAgRatioDen = 1
vuXYExchgFlag = 0 -- X,Y Data Exchange Flag 0:No,1:Exchange
--CV relative Robot's Directional Vector
cmpstVectorX = 0
cmpstVectorY = 1000
cmpstVectorZ = 0--Normally Default Zero
```

Vision system error	
Cause of problem	1. Vision system not taking pictures. 2. Modbus position setting error. 3. Communications and connections settings error. 4. Communication timeout time is greater than the time interval for taking pictures (7.5.5 Vision Tracking Mode, step 1.2). 5. Communication timeout is set too small (e.g., 10 ms). 6. Communication line is subject to interference. 7. Serial port is wired incorrectly. 8. Serial port connector is not fastened.
Check and correction steps	1. Check if the firmware version includes the CVT function. 2. Check if the robot controller DO1 is connected to the camera signal output point of the vision system. 3. Check if the X, Y, and AG values are accurately transmitted to the corresponding Modbus position when the camera captures the object. 4. Stop the robot program, trigger the vision system to take pictures, and check whether the Modbus address of the done flag of the robot controller is 1. 5. Check if the communication settings of the robot controller/vision system are compatible (7.5.5 Vision Tracking Mode, step 1.3). 6. Check if the communication timeout time is smaller than the time between the pictures (7.5.5 Vision Tracking Mode, step 1.2). 7. Check if there are devices around that cause interference. 8. Check if the serial port cable has a jumper (needs to be the version without a jumper). 9. Check if the connector of the serial port is secured.

Encoder error	
Cause of problem	1. Encoder connection error. 2. Encoder coupling is not secured. 3. Slippage between encoder and conveyor belt.
Check and correction steps	1. Check if the encoder is connected to the first group of external encoder. 2. Check if the encoder is wired correctly. 3. Check if the encoder connector is secured and the wiring is correct 4. Start Delta's robot operating software DRAStudio, go to the [CVT] page, open [Info], read the encoder position, start the conveyor belt and rotate the encoder, and read whether the encoder value is correct. If the value is wrong, it may be a wiring error or the coupling is not secured 5. For example: if a 1000P encoder is used, the value should increase by 4000 pulses after the encoder rotates once. 6. Check the encoder for interference signals; use isolated wires and connect the housing.

The robot tracks the wrong position	
Cause of problem	1. Tool coordinate calibration error. 2. Four-point calibration error. 3. The offset parameters between the camera and the robot are set incorrectly. 4. The origin of User frame is set incorrectly. 5. Applied the wrong User frame coordinates for the points.
Check and correction steps	1. Rotate the tool coordinate group in use to confirm whether the tool coordinate is correct; if it is not correct, reset. 2. Check if the tool coordinates are correctly selected and calibrated when conducting four-point calibration. 3. Check if the four-point calibration value is correct. 4. Adjust the offset parameters between the camera and the robot to make the robot track the correct position. 5. If the tracking position still deviates after rotating the object, move the object under the camera to take a picture, rebuild the sample, and repeat the steps from section 7.5.5. 6. Confirm whether the point applies the matching User frame (conveyor belt 1: UF1; conveyor belt 2: UF2).

```
---CVT Template---
CVT_ChangeMotion()
RobotServoOn()
---->CV1Setting<----
CVT_SelectMode(1, 1)--1:With Vision, 2:Without Vision
CVT_SetTriggerMode(1, 2)--1:DO, 2:DI
--Encoder Unit Ratio
cvtFactor_num = 614
cvtFactor_den = 1000
--Encoder AMF window(ms)
interval = 10
--CCD & Robot Transform Offset(um)
trans_ccd_x = 0
trans_ccd_y = -1000000
rotat_ccd_c = 0
--CCD Unit Ratio
vuPix2UmNum = 10
```

Position exceeds software limit (robot iscompletely straightened)	
Cause of problem	1. The end line setting position is too close to the robot's limit position for movement. 2. The direction vector of the conveyor belt is set incorrectly.
Check and correction steps	1. Set the position of the end line along the direction of movement of the conveyor belt and modify it to be towards the origin of the robot. 2. Check if the direction vector of the conveyor belt is correct.

Re-process	
Cause of problem	1. Configured value of parameter NGZoneRadius is set too small. 2. The direction vector of the conveyor belt is set incorrectly.
Check and correction steps	1. Increase the value of the parameter NGZoneRadius (adjust according to the case). 2. Check if the directional vector of the conveyor belt is correct.