
Mech-Viz Manual

Mech-Mind

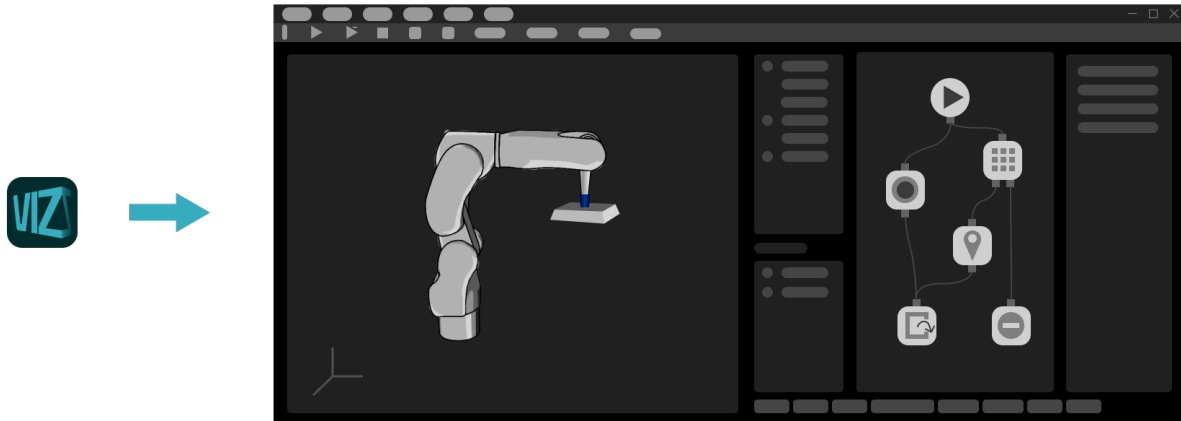
Dec 23, 2022

CONTENTS

1 Mech-Viz Quick Facts	2
2 Getting Started with Mech-Viz	8
3 Workflow	39
4 Scene	169
5 Robot	183
6 Tools and Workobjects	195
7 Collisions	226
8 Plan History	242
9 Others	252
10 Log	256
11 Debugging Tips	257
12 Supplementary Tools	277
13 Mech-Viz Appendix	312

Mech-Viz is a cutting-edge **robot intelligent programming software** that has a visualized and coding-free programming workspace and can make a simulation with one click. It has built-in intelligent algorithms such as path planning, collision detection, etc., and has an adapted robot library including models of many major brands.

As an integral part of Mech-Mind Vision System, Mech-Viz is used in conjunction with the other Mech-Mind software to realize intelligent production based on machine vision.



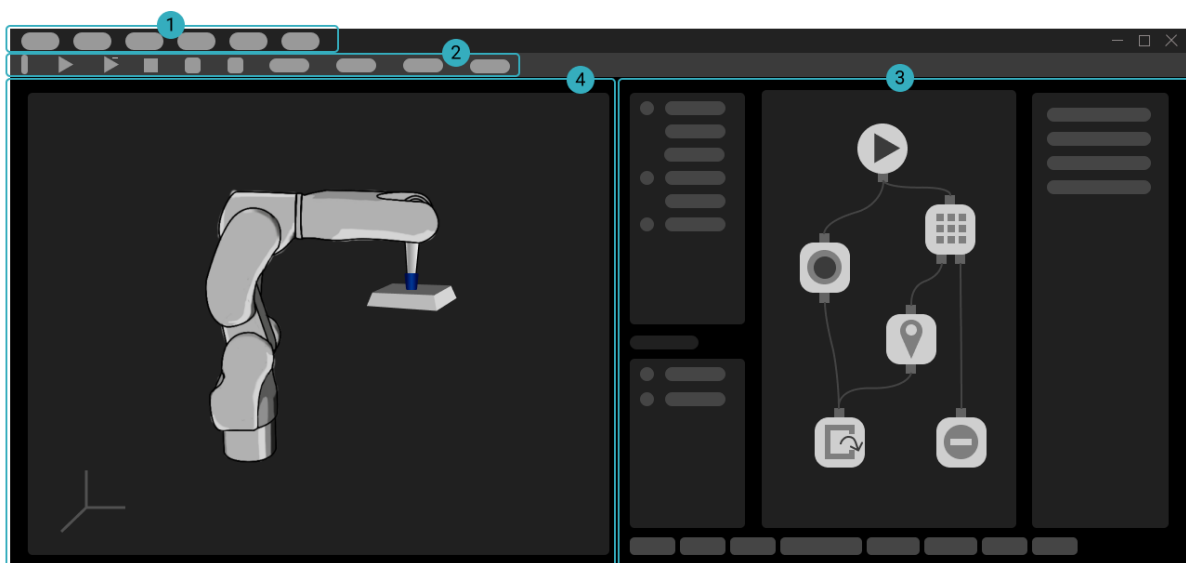
Check the chapters below to get started with Mech-Viz and create your first Mech-Viz project.

MECH-VIZ QUICK FACTS

Mech-Viz is a cutting-edge **robot intelligent programming software** that has a visualized and coding-free programming workspace and can make a simulation with one click. It has built-in intelligent algorithms such as path planning, collision detection, etc., and has an adapted robot library including models of many major brands.

As an integral part of Mech-Mind Vision System, Mech-Viz is used in conjunction with the other Mech-Mind software to realize intelligent production based on machine vision.

1.1 Overview of the Interface



The interface of Mech-Viz consists of the following parts:

- ☒ **Menu Bar**: menus for basic features such as opening and saving a project, display adjustment, tools, settings, etc.

- ☒ **Toolbar**: provides options for simulating or running the project, connecting to the robot, adjusting robot velocity, etc.

- ☒ **Functional Panels**: used to configure settings for an executable project, including the following 8 panels:

Workflow, Scene, Robot, Tools and Workobjects, Collisions, Plan History, Others, and Log





☒ *3D Simulation Area*: displays the robot path, collision detection results, workobject poses, point clouds, etc., during project simulation or execution.

1.2 3D Simulation Area

3D simulation area is a visualization window where you can edit the project. You will better understand various features in Mech-Viz if you are familiar with the the 3D simulation area.

1.2.1 Mouse Actions

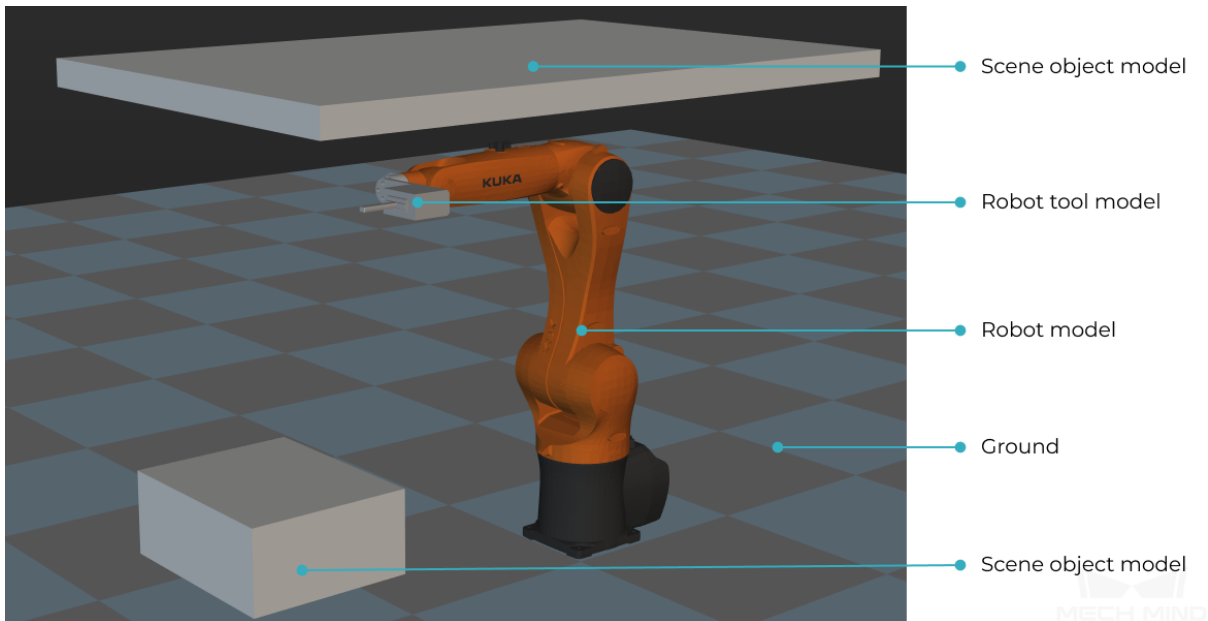
The following mouse actions can be used to adjust your view in the 3D simulation area.

<p>Rotate the view</p>	<p>Hold left button and drag</p>	
<p>Display menu for quick view adjustments</p>	<p>Right-click</p>	
<p>Pan the view</p>	<p>Hold middle button and drag</p>	
<p>Zoom in or out</p>	<p>Scroll</p>	

1.2.2 Display Scene Objects

The 3D simulation area displays the following models in a project:

- Robot model
- Model of the robot tool
- Models of scene objects
- Ground

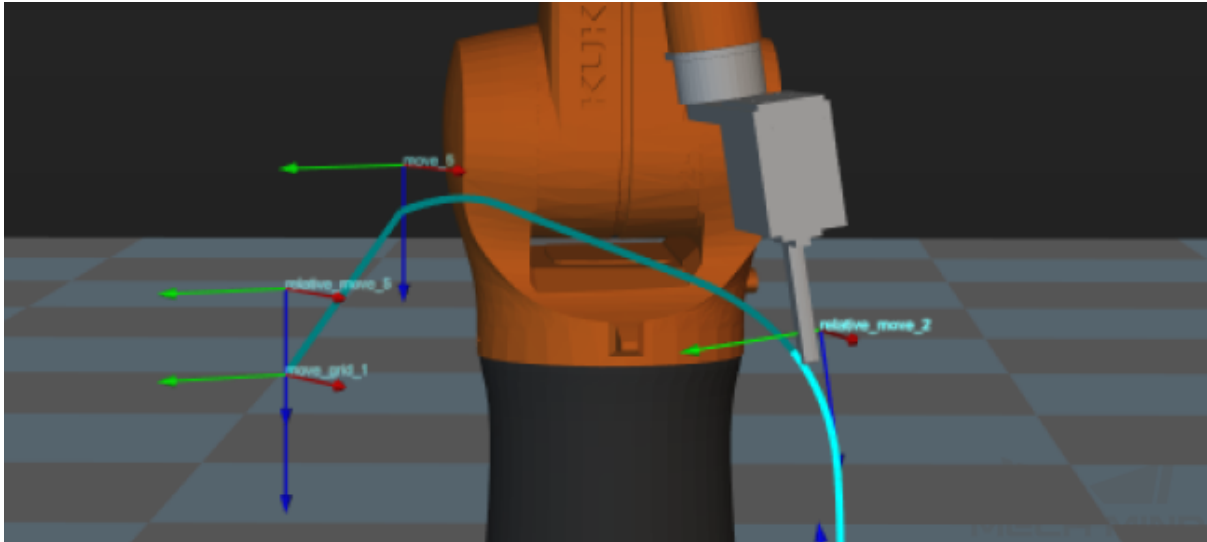


Note: The ground and the selected robot model are displayed by default for a blank project. Models of scene objects and robot tool must be added manually according to actual requirements.

1.2.3 Display Planned Path

Whether moving the real robot or just the simulated robot, the 3D simulation area displays the planned path of the robot.

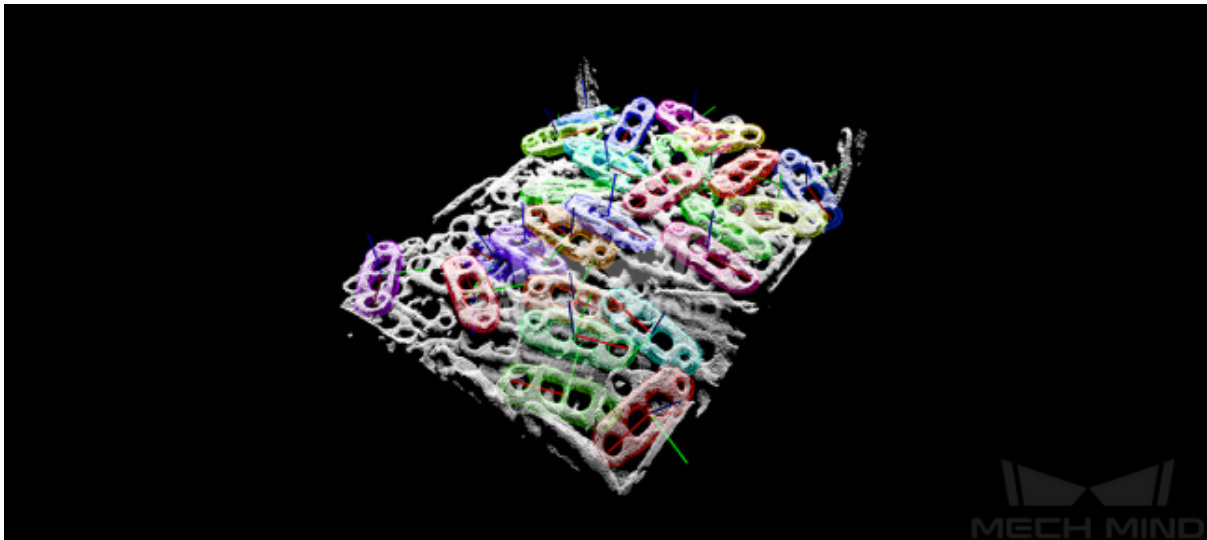
You can use the displayed path as an aid for adjusting the motion of the robot.



1.2.4 Display Received Vision Result

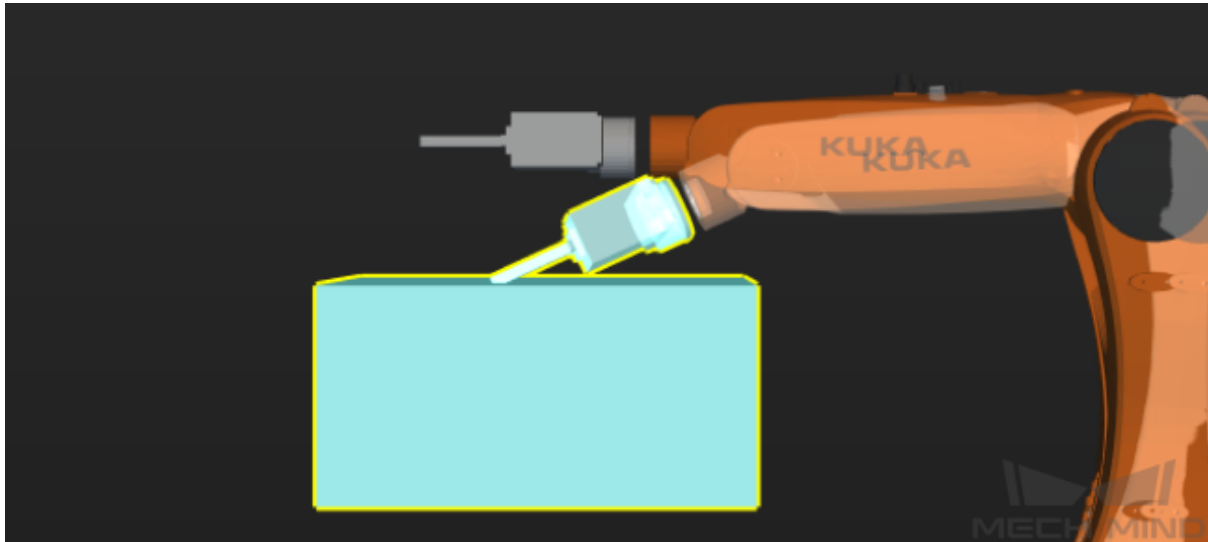
The vision result received from Mech-Vision is displayed in the 3D simulation area.

A vision result must include workobject poses and indices for the Mech-Viz project to run. A vision result may also include customized pose labels that indicate characteristics of the workobjects and point cloud of the scene and workobjects.



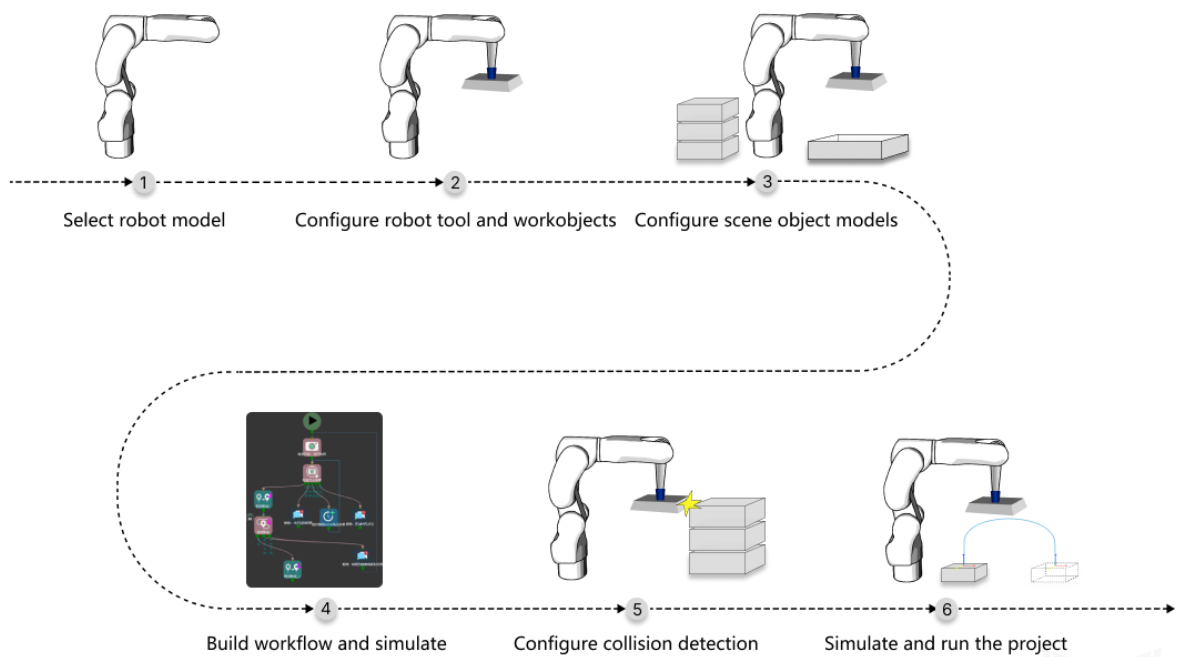
1.2.5 Display Detected Collisions

When planning a path for the robot, Mech-Viz detects whether the robot and/or robot tool collides with scene objects and workobjects. When a possible collision is detected, the two objects involved in the collision are displayed in light-blue and outlined with yellow in the 3D simulation area.



GETTING STARTED WITH MECH-VIZ

The process of building and configuring a Mech-Viz project usually consists of the following steps:



This chapter guides you through creating the [example Mech-Viz project](#) that completes a simple carton-picking task.

2.1 Preparation

Before you start building a Mech-Viz project, the following preparation must be completed:

- Connect the hardware in the system.
- Refer to `robot_integrations` and enable the communication between the robot and Mech-Center.

Note: In the example project, a UR robot is connected according to `ur_setup_instructions`.

- The pick point is provided by Mech-Vision. Please download the [example Mech-Vision project](#). Or create your own Mech-Vision project according to `boxes_single_case_no_input`.
- Prepare models of scene objects and robot tools. In real applications, these models are designed and created by the user according to actual on-site conditions.

Note: The models used in the example project are included in the project folder.

2.2 Create Your First Mech-Viz Project

To successfully create the Mech-Viz project, please read the following chapters in the listed order.

2.2.1 Select Robot Model

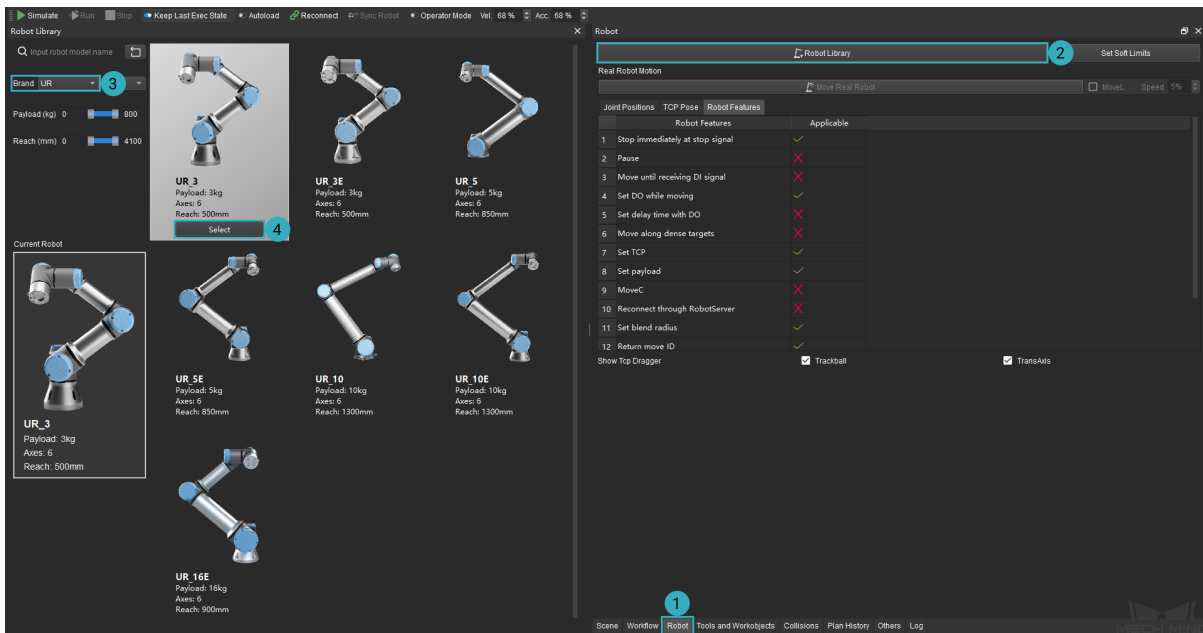
The first step of building a Mech-Viz project is import the robot model you use to Mech-Viz.

Attention: Starting from version 1.6.0, only robot models from the widely used brands are preloaded in Mech-Viz. All other robot models must be manually imported.

Please refer to [Robot Model Package](#) for instructions on downloading and importing robot model packages.

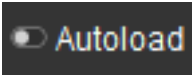
The example Mech-Viz project uses a UR_3 robot. In a real application, please select the robot that you actually use on-site.

1. After downloading and importing the robot model package, click on the **Robot** tab in the lower right.
2. Click on *Robot Library* at the top of the tab.
3. On the left side, select the robot brand from the drop-down menu of **Brand**, or use the search box above it.
4. Select the robot model.



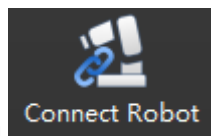
5. In the menu bar, click on *File* → *Save Project* or use the shortcut **Ctrl + S** to save the changes to the project.

Attention: While building and configuring the project, remember to save the changes promptly!

1. Click on  in the toolbar.

Note: The project must be set to autoload for it to be registered in Mech-Center.

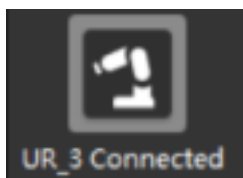
To control the real robot, complete the following in Mech-Center:



1. Open Mech-Center, and click on

2. The robot is successfully connected if:

- A message saying **Robot: server connected to the robot** shows up in the Log panel, and



- **UR_3 Connected** shows up in the Service Status panel.

2.2.2 Configure Robot Tools and Workobjects

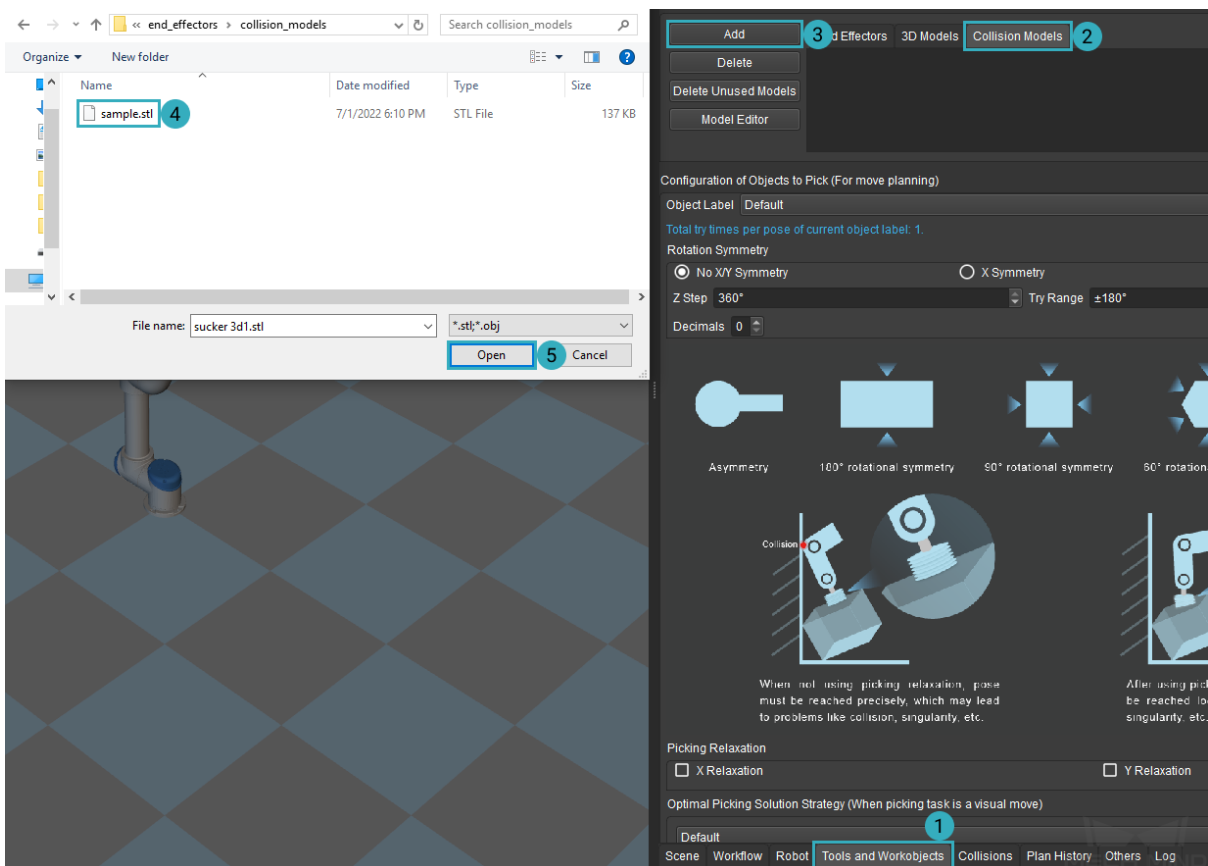
Once you finish the configurations in this chapter, you will be able to:

- Have the robot tool displayed in the 3D simulation area.
- Add robot tools to collision detection.
- Allow Mech-Viz to plan the path and picking strategy for the robot.

Add Collision Model

The robot tool model file used in the example project is located in `\sampleviz\end_effectors\collision_models`.

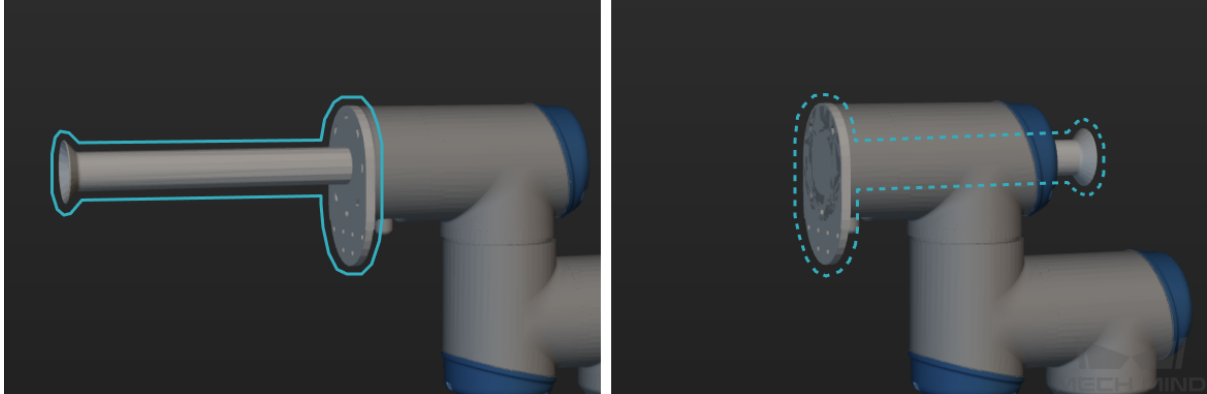
1. Click on the **Tools and Workobjects** tab in the lower right.
2. Click on the **Collision Models** tab at the top.
3. Click on **Add** on the left.
4. Select the **sample** STL file from the above location.
5. Click on **Open**.



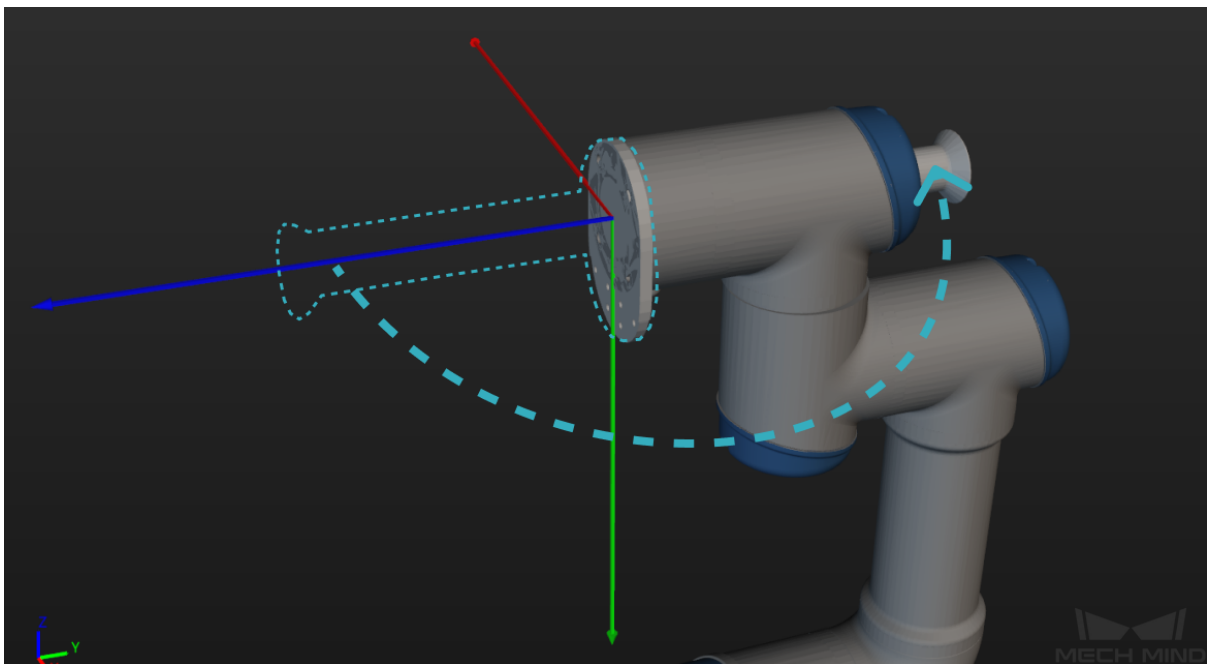
Adjust Model Position

The added tool model may not be in the correct position and thus requires adjustment.

In the figures below, the left one is a correctly positioned tool model, but the model that you just added probably looks like the one on the right.

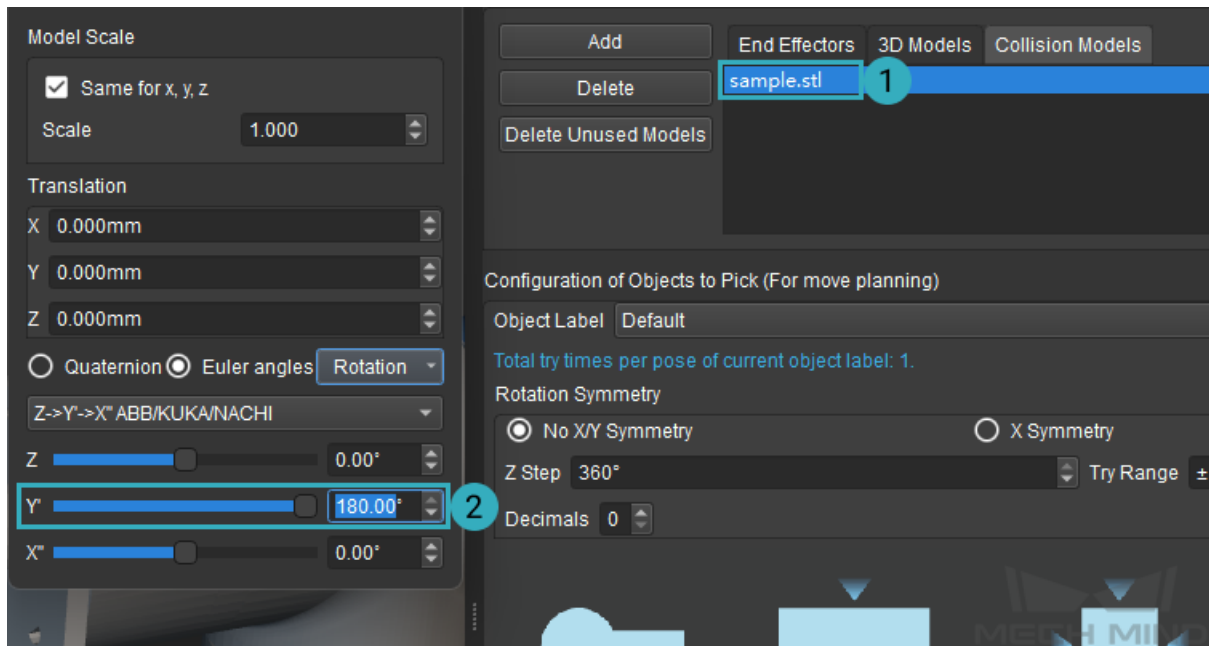


With the axes displayed, you can see that the tool model is flipped about its Y-axis (green).



So in this case, all you need to do is flip it around.

1. Double-click on the model name in the **Collision Models** tab.
2. In the pop-up window, change the value of **Y'** to 180°.

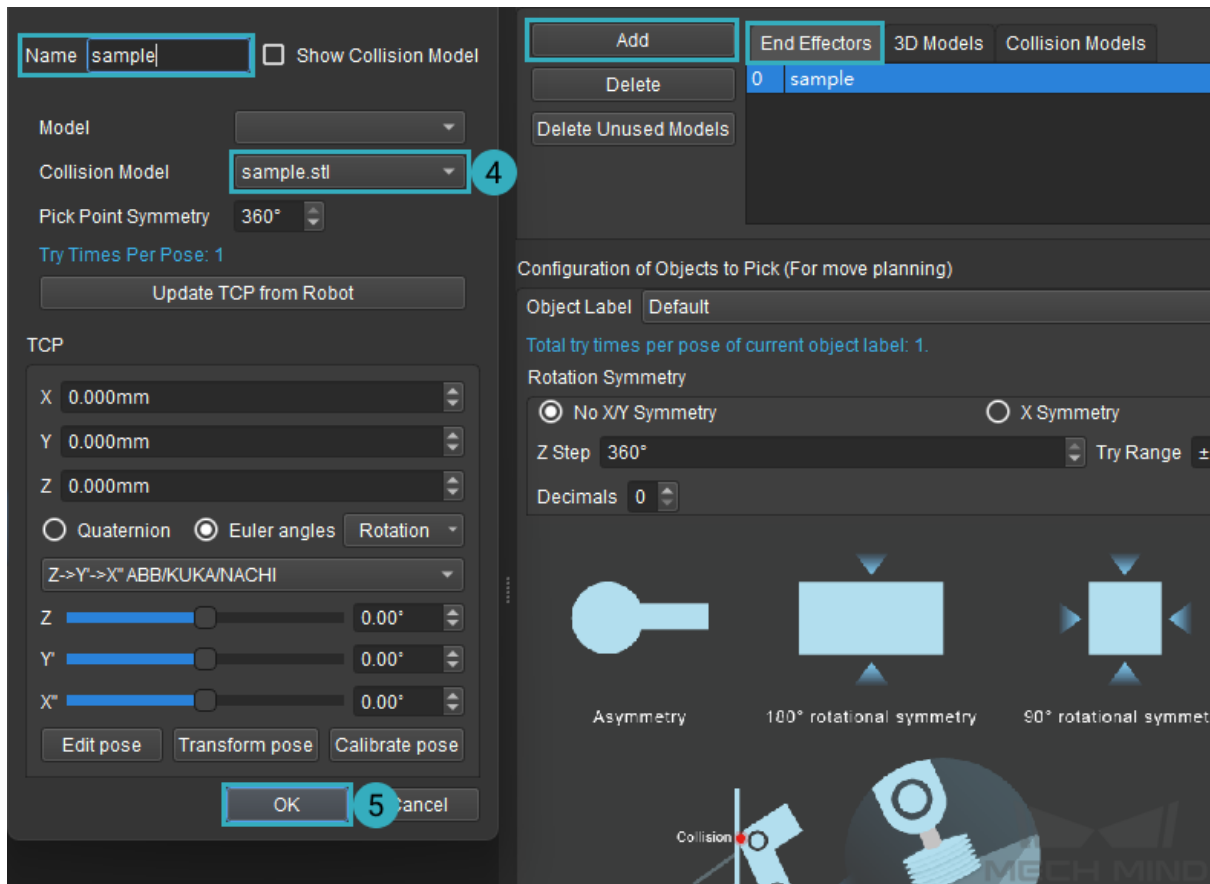


Hint: In this window, you can adjust the scale, position and rotation of the model. In real applications, please adjust these parameters according to your actual needs.

Add End Effector

After adding the tool model to **Collision Models**, you need to add it to **End Effectors** as well for adjusting the TCP and switching tools when multiple tools are used.

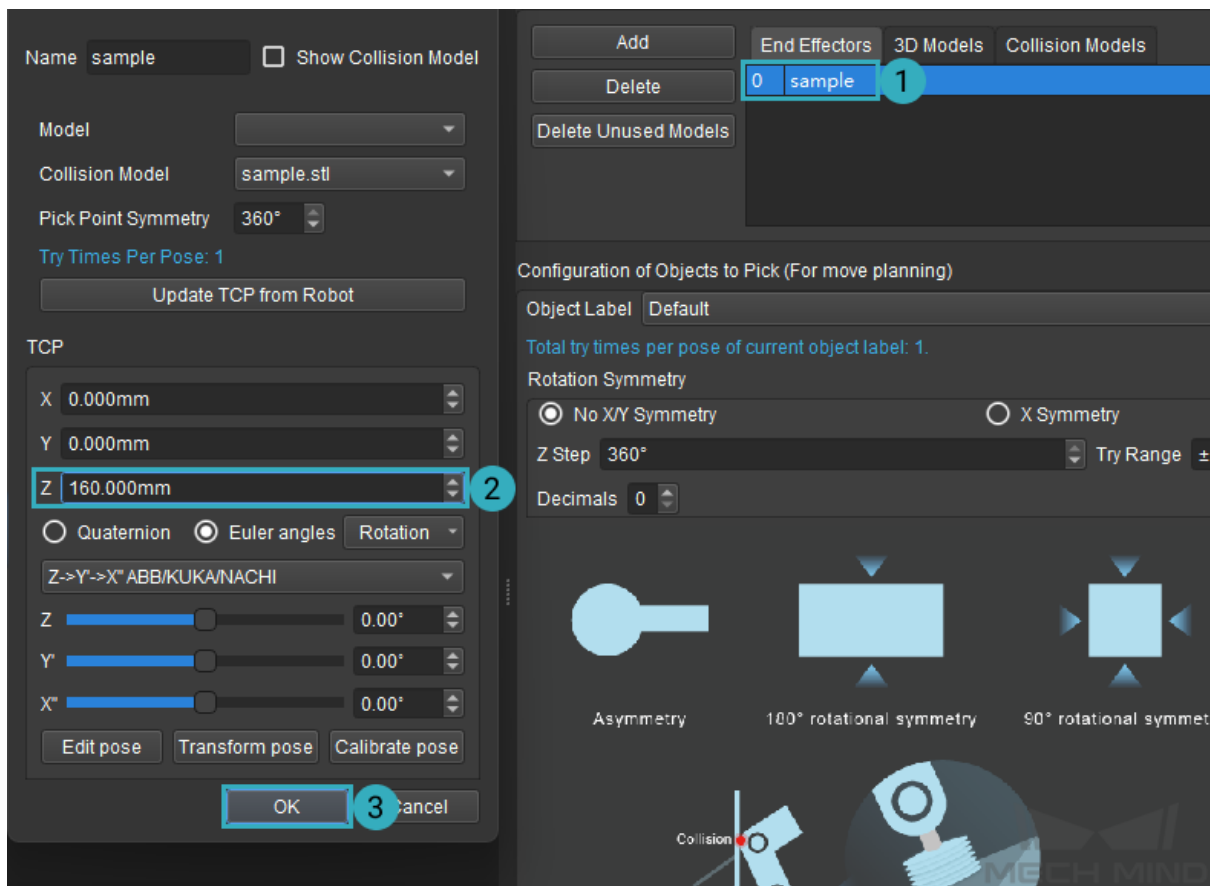
1. Click on the **End Effectors** tab to the left of **Collision Models**.
2. Click on *Add*.
3. In the pop-up window, type in a name for the robot tool.
4. Select the corresponding model from the drop-down menu of **Collision Model**.
5. Click on *OK* at the bottom to save the changes.



Adjust Tool Center Point (TCP)

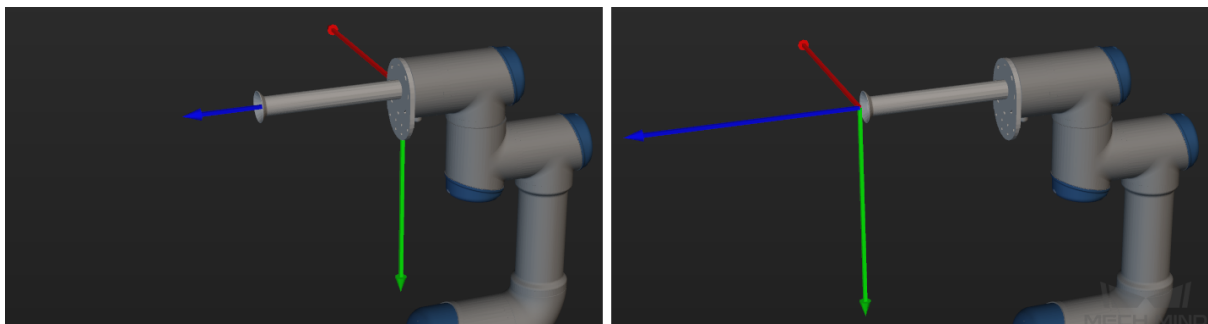
In the 3D Collision Area, the TCP is represented by the origin of the axes. Its default position is at the center of the robot flange. After adding a robot tool, you'll need to move the TCP to where the tool actually interacts with workobjects.

1. Double-click on the model name in the **End Effectors** tab.
2. In the pop-up window, increase the value of Z until the TCP is at the tip of the tool.
3. Click on *OK* to save the changes.



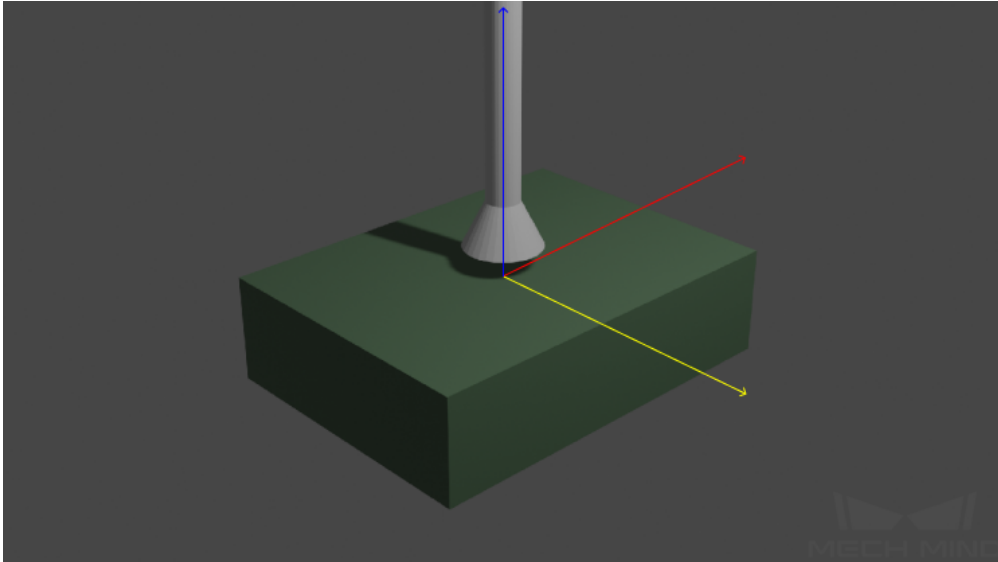
Note: In real applications, please input the calibrated TCP values. You can also use the *Update TCP from Robot* button to automatically fetch the current TCP values set on the robot.

The figure below shows the TCP before (left) and after (right) the above adjustment.



Configure the Workobject

After configuring the robot tool, you'll need to configure the workobject (in this case, the carton) so that Mech-Viz can plan the robot path and picking strategy according to the characteristics of the workobject.



The following sections in the **Tools and Workobjects** tab can be configured:

- **Rotation Symmetry:** Here you can set the rotational symmetry of the workobject. In the example project, the carton's top surface is to be picked by the robot. This surface is a rectangle, and it overlaps with itself if you rotate it 180° about the its Z-axis (blue). Since in our simple example, the carton will be placed upright and picked by its top surface, you can think of the carton as not having X- or Y-axis symmetry. Therefore, the options to choose are: **No X/Y Symmetry**, **180° for Z Step**, and **$\pm 180^\circ$ for Try Range**.
- **Picking Relaxation:** In carton picking scenarios, usually no picking relaxation is set. Because the rim of the suction cup has to be parallel to the carton surface, thus ensuring a firm grasp on the carton. Therefore, the robot must reach the picking pose precisely.
- **Optimal Picking Solution Strategy:** This parameter determines how much the robot tool is rotated during picking. In our example, set it to **Default** to avoid unnecessary rotation after the carton is picked up, which may result in dropping of the the carton.

If picking relaxation is set, the robot doesn't have to reach the picking pose precisely.

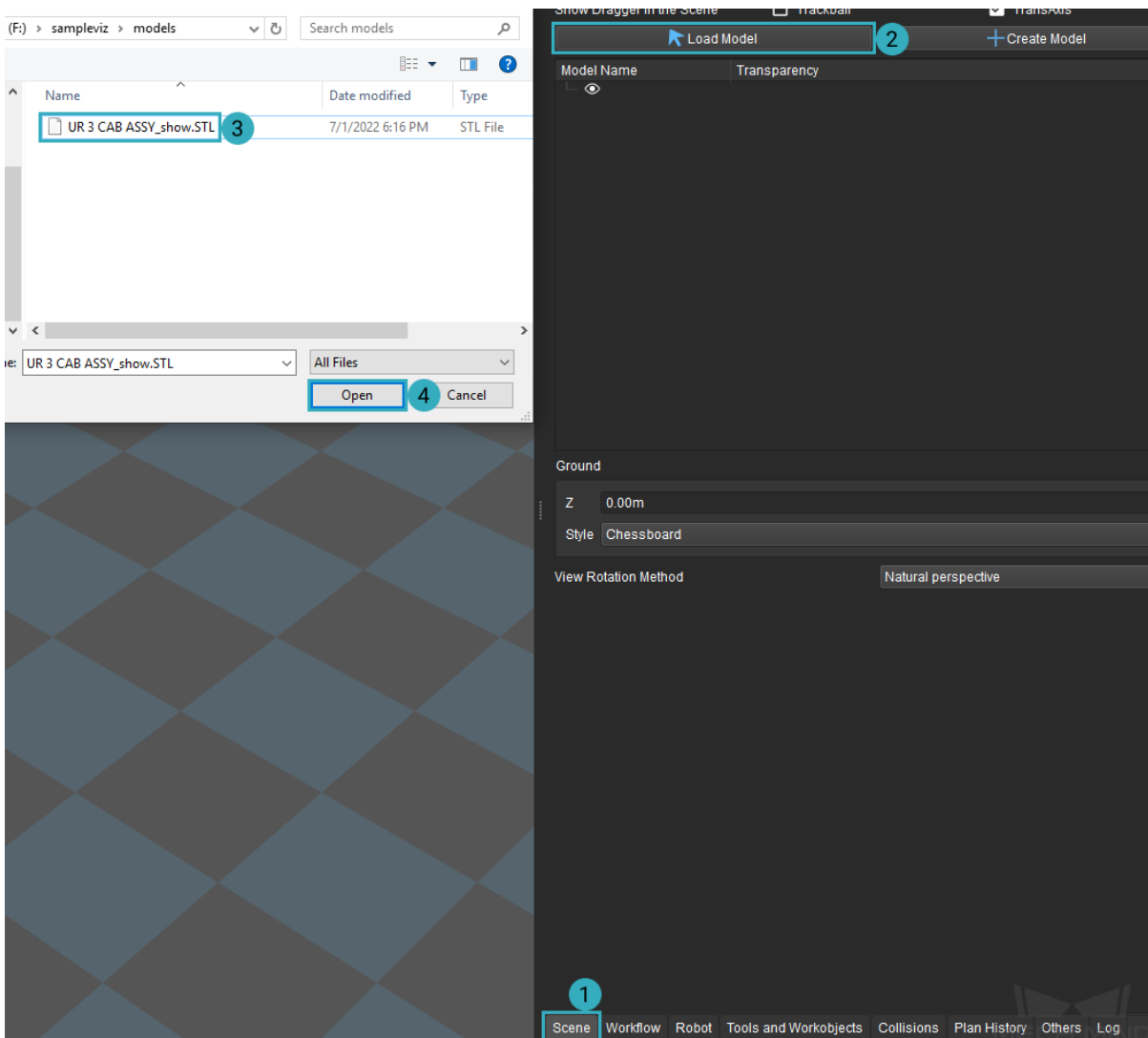
2.2.3 Configure Scene Object Models

With models of scene objects added, scene objects can be taken into consideration during collision detection, affecting the planning of path and picking strategy for the robot. These models are also displayed in the 3D simulation area, reproducing the actual scene more realistically.

In the **Scene** tab, you can create simple models directly with *Create Model*, or import already made models with *Load Model*.

In the example project, you can import a ready-made model from `\sampleviz\models`.

1. Click on the **Scene** tab in the lower right.
2. Click on *Load Model* at the top of the tab.
3. Select the **UR 3 CAB ASSY_show** STL file from the above location.
4. Click on *Open*.

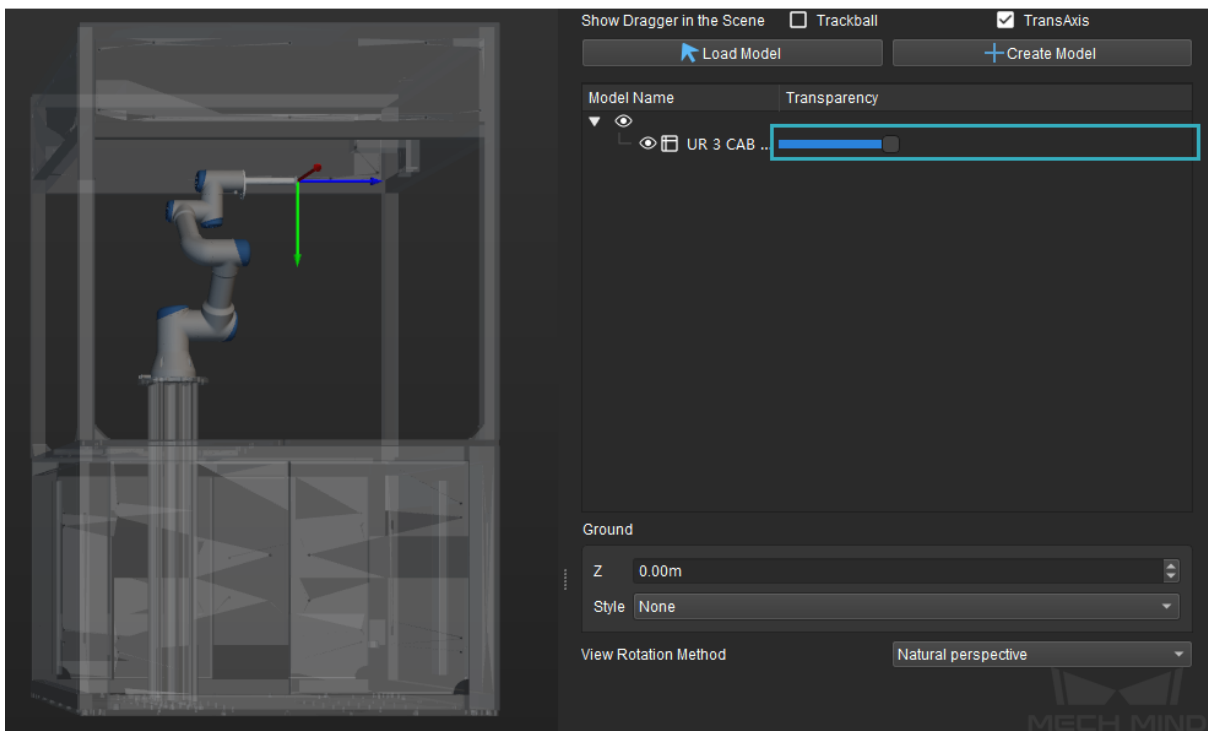
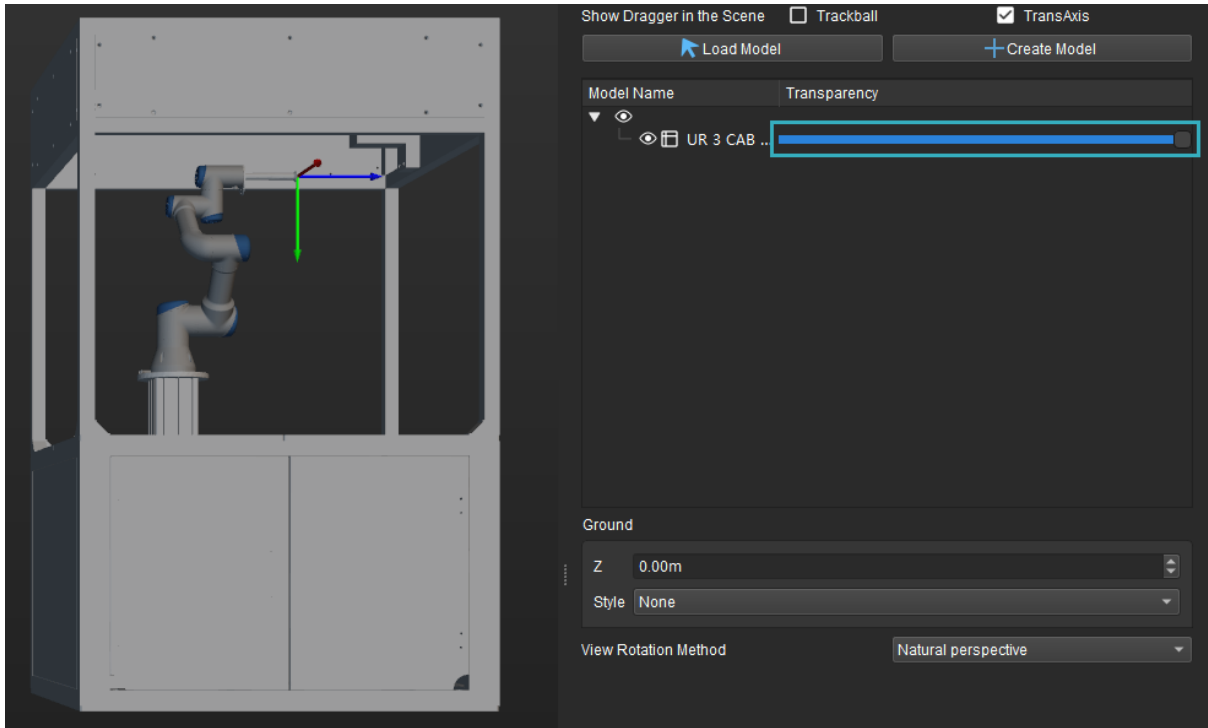


Hint: If the position of the added model is incorrect, you can adjust it by the following methods:

- Double-click on the model name and change the position and/or rotation values in the pop-up window.
 - Check the **Trackball** and/or **TransAxis** options at the top of the **Scene** tab to display the translation/rotation manipulators of the model. Hold down **Ctrl** and click on a manipulator to drag it.
-

Scene object models might block your view of the planned path, poses, other objects, etc., in the 3D simulation area.

In such cases, you can adjust the opacity of the model by the slide bar to the right of the model name.

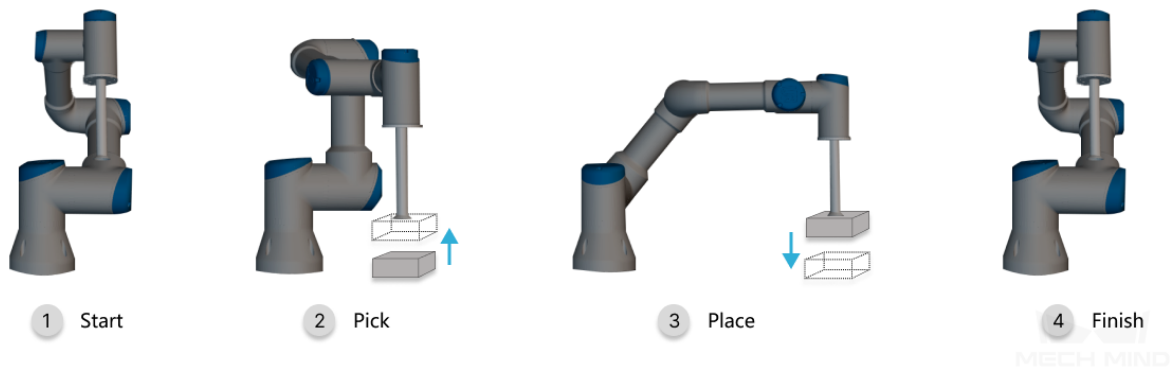


2.2.4 Configure the Workflow

Workflow is the core of a Mech-Viz project. It is where you configure what the robot does at what point.

In the **Workflow** tab, you'll add Tasks to the workflow, connect the Tasks, and adjust the parameters of Tasks to achieve the desired actions.

In the example project, you will create the following simple pick-and-place workflow.



Build the Workflow

Set Home Position

The home position is where the robot starts to move and where it will return at the end. The home position should be relatively removed from other objects, and the robot shouldn't block the camera when it's at the home position.

The *move* Task can be used to define a target pose on the robot's path and set how the robot moves to this pose.

Note: With the real robot connected, you can sync the pose of the real robot to the simulated robot in Mech-Viz.

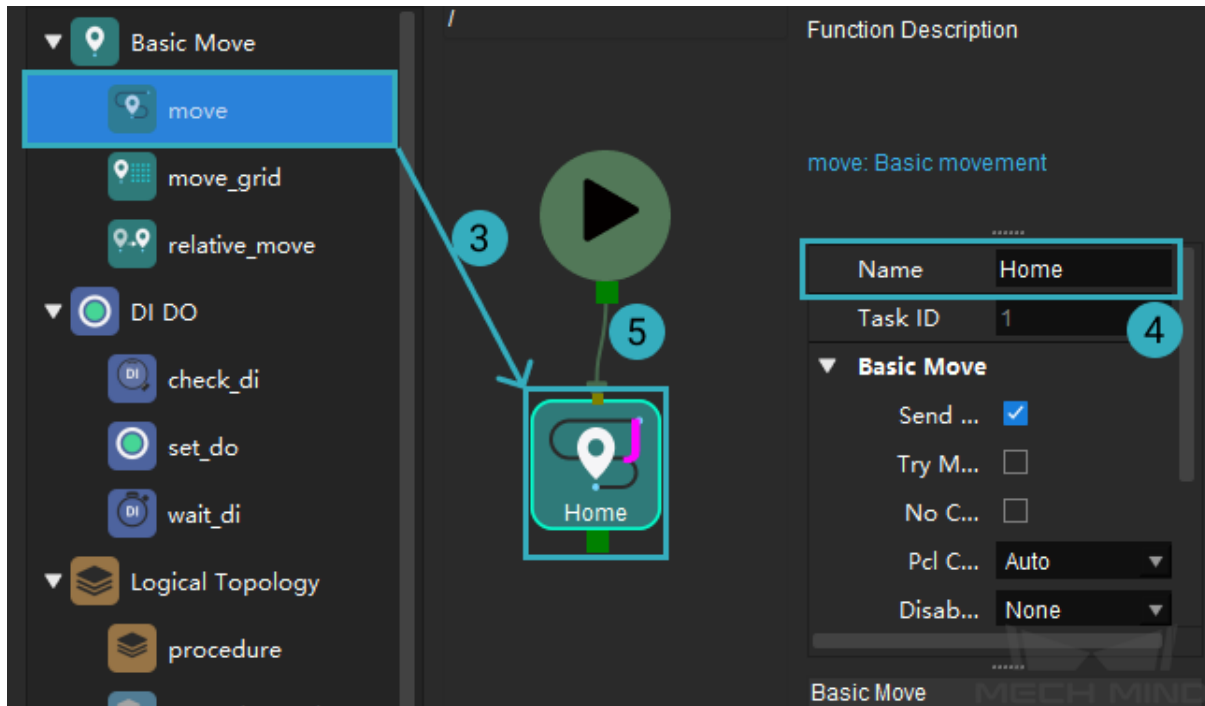
1. In Mech-Viz, click on *Sync Robot* in the toolbar. The simulated robot will move in sync with the real robot.
2. Move the real robot to the desired home position. Check the simulated robot to make sure the pose of the real robot has been synced.
3. From the **Basic Move** category in the Task library, drag the *move* Task to the programming area. The current pose of the simulated robot will be automatically set as the target.

Note: If not connected to a real robot, you can still adjust the target pose of the *move* Task in the parameter panel. The simulated robot will display the adjusted pose simultaneously.

4. In the upper part of the parameter panel, rename the Task to **Home** in the **Name** field.

5. Connect the start icon to **Home**.

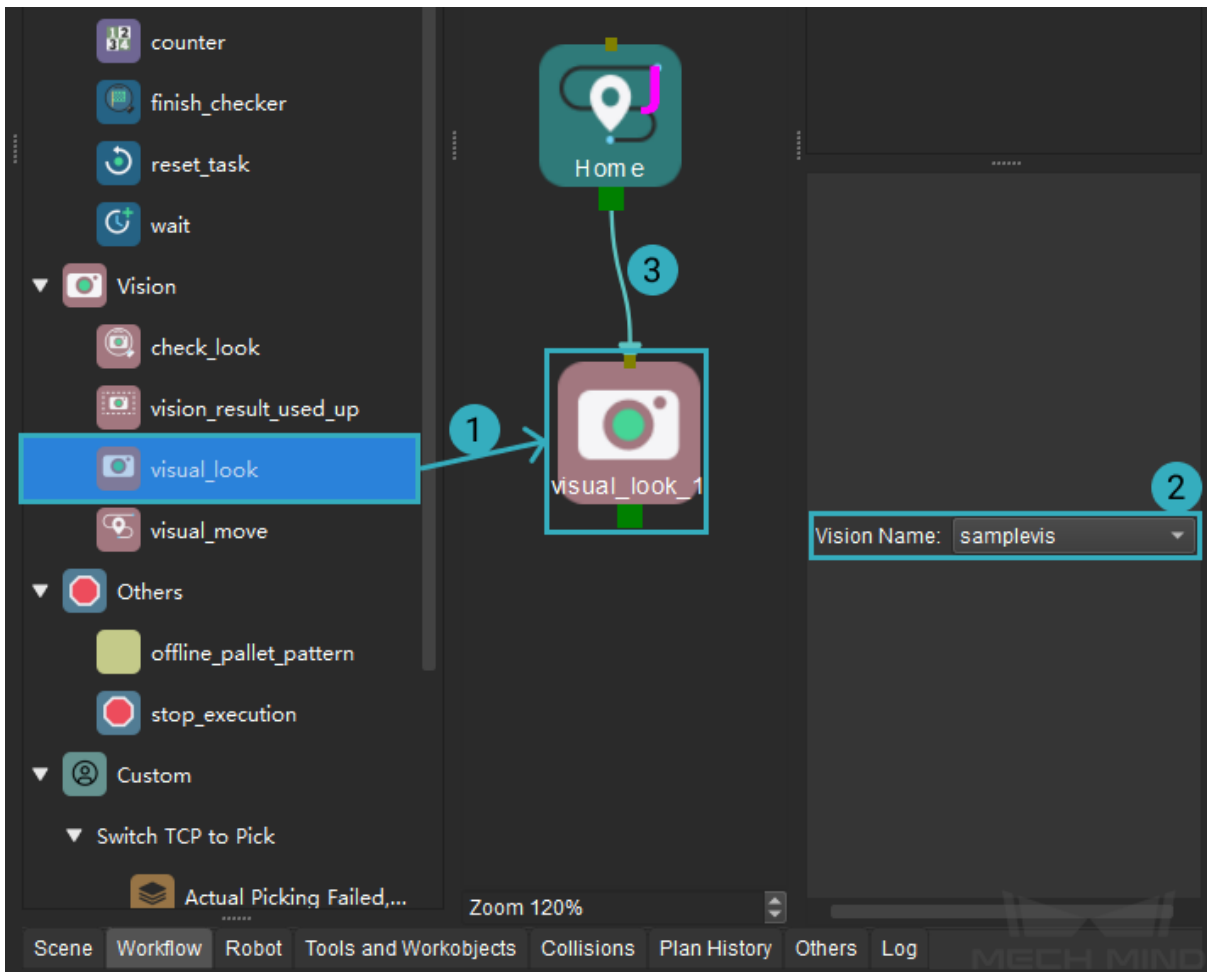
Hint: A connection is added by holding and dragging an exit port onto an entry port.



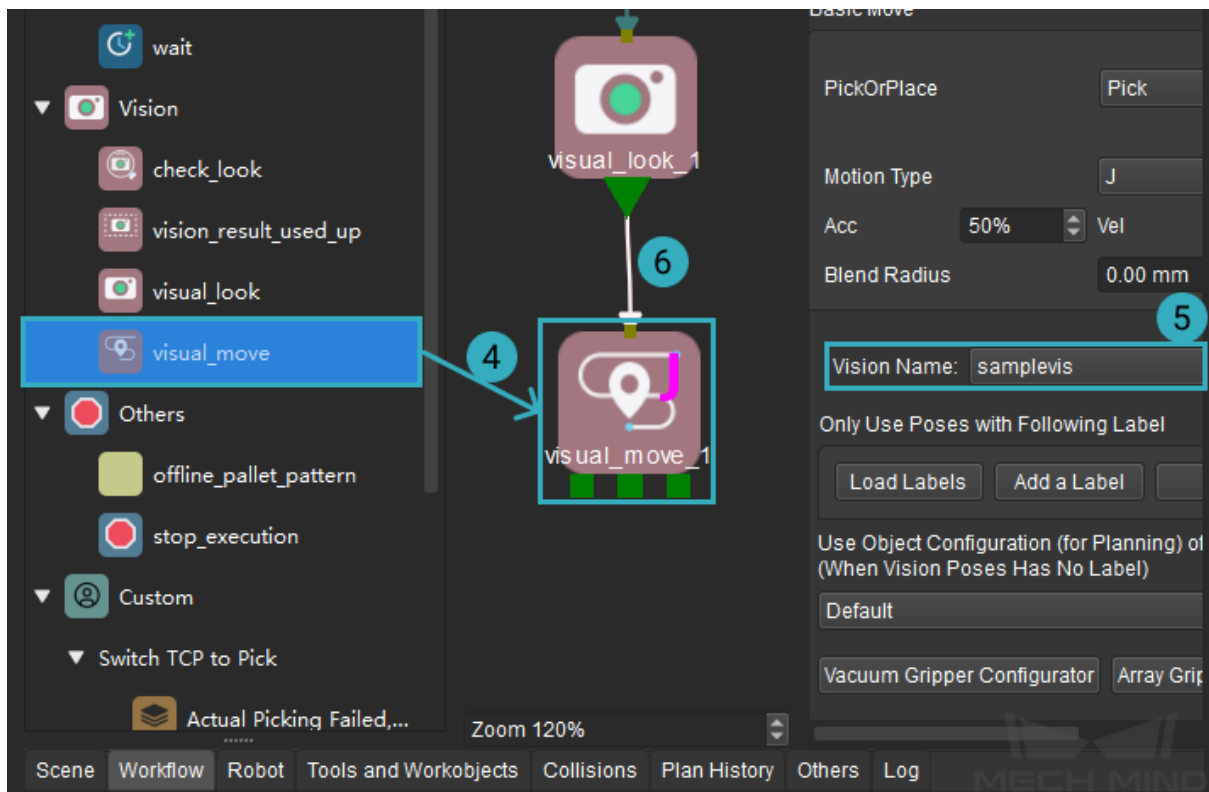
Set Picking Target

Mech-Viz receives the workobject pose from Mech-Vision, and you'd want the robot to move according to this pose. This can be accomplished with the *visual_look* and *visual_move* Tasks.

- *visual_look* Task obtains the workobject pose from the corresponding Mech-Vision project.
 - *visual_move* Task moves the robot according to the received workobject pose.
1. From the **Vision** category in the Task library, drag the *visual_look* Task to the programming area. This Task is automatically named `visual_look_1`.
 2. In the lower part of the parameter panel, select **samplevis** (the example Mech-Vision project) from the drop-down menu of **Vision Name**.
This will enable the Task to obtain workobject poses from the Mech-Vision project named **samplevis**.
 3. Connect **Home** to `visual_look_1`.



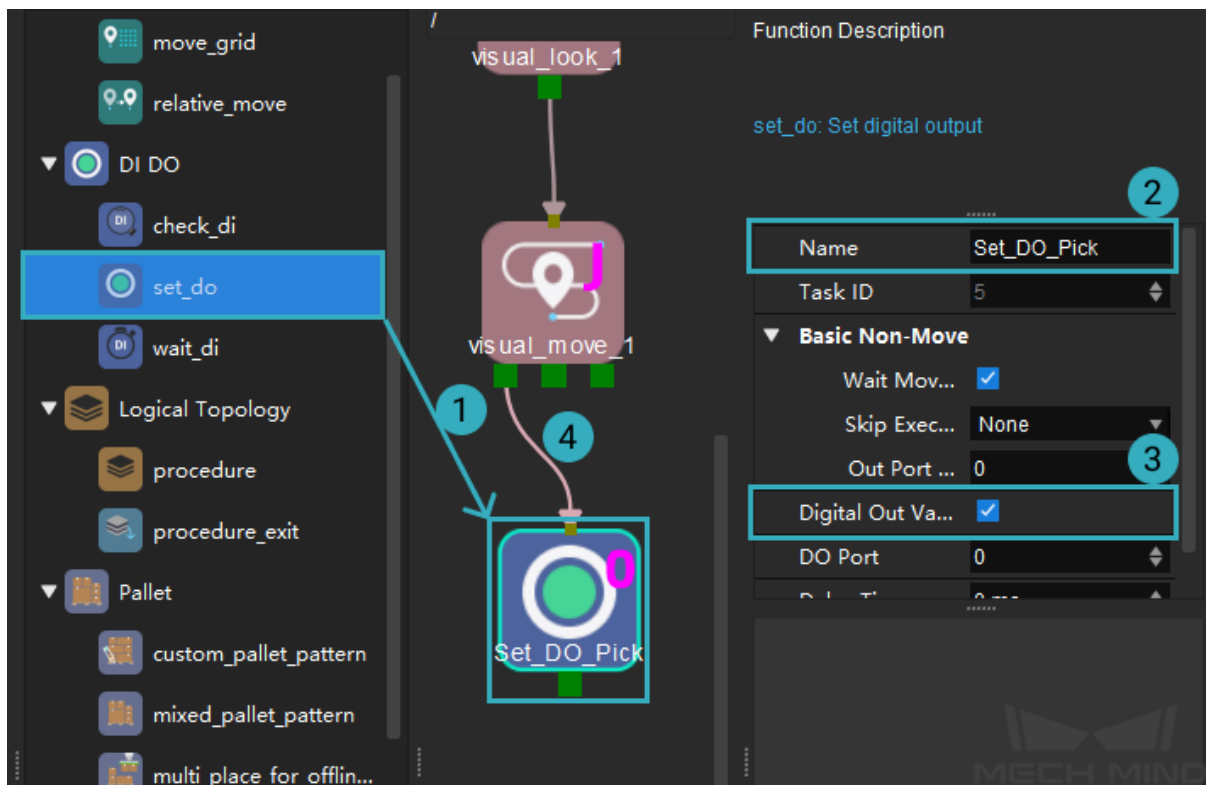
4. From the **Vision** category in the Task library, drag the *visual_move* Task to the programming area. This Task is automatically named **visual_move_1**.
5. In the lower part of the parameter panel, select **samplevis** from the drop-down menu of **Vision Name**.
6. Connect the exit port of **visual_look_1** to the entry port of **visual_move_1**.



Set DO for Picking

Once the robot reaches the picking target, the robot has to operate its tool to pick up the workobject. This is achieved with the *set_do* Task, which instructs the robot to operate its tool.

1. From the **DI DO** category in the Task library, drag the *set_do* Task to the programming area.
2. In the parameter panel, rename the Task to **Set_DO_Pick** in the **Name** field.
3. Check the checkbox of **Digital Out Value**, which tells the robot to start generating vacuum.
4. Connect the left-most exit port of **visual_move_1** to **Set_DO_Pick**.



Set Placing Target

After picking up the workobject, the robot should move to a predetermined position to drop off the workobject. Once again, the *move* Task can be used to define the placing target.

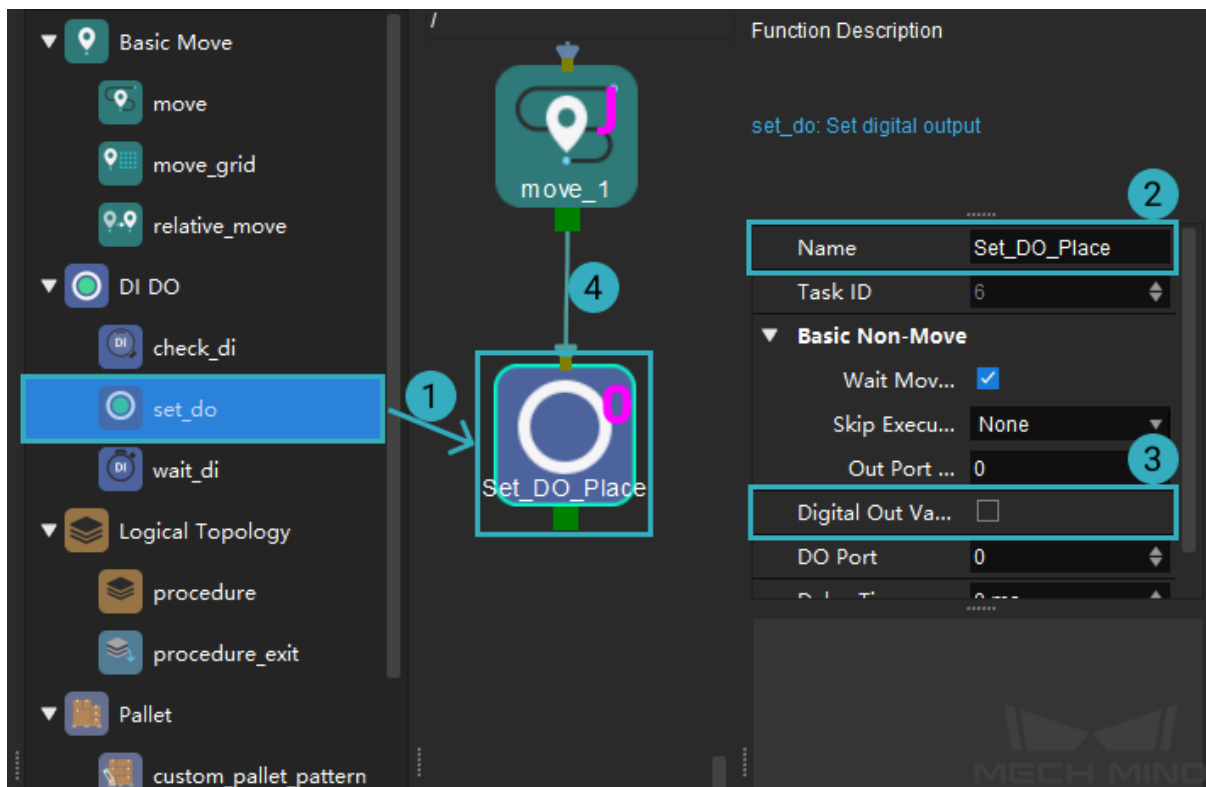
1. Move the real robot to the desired placing target. Check the simulated robot to make sure the pose of the real robot has been synced.
2. From the **Basic Move** category in the Task library, drag the *move* Task to the programming area. This Task is automatically named `move_1`.
3. Connect `Set_DO_Pick` to `move_1`.

Set DO for Placing

Once the robot reaches the placing target, the robot has to once again operate its tool to drop off the workobject, which is achieved with the *set_do* Task.

1. From the **DI DO** category in the Task library, drag the *set_do* Task to the programming area.
2. In the parameter panel, rename the Task to `Set_DO_Place` in the **Name** field.
3. Uncheck the checkbox of **Digital Out Value**, which tells the robot to release.

4. Connect `move_1` to `Set_DO_Place`.



Return to Home

After the robot drops off the workobject, the next step is to return to its home position. As you already set the home position in **Home**, now you just need to copy and paste this Task.

1. Right-click on **Home** and select **Copy**, or left-click on **Home** and then use the shortcut Ctrl + C.
2. Navigate down to the end of the workflow. Right-click on the empty area and select **Paste**, or use the shortcut Ctrl + V.
3. This pasted Task is automatically named `move_2`. Rename it to **Return_Home** in the **Name** field in the parameter panel.
4. Connect `Set_DO_Place` to `Return_Home`.

The entire workflow now looks like this:



Hint: Right-click in the programming area and select **Format** to automatically align the Tasks and adjust their spacing.

Click on *Simulate* in the toolbar, the simulated robot will move according to the workflow.

You can see that although we built the workflow according to the correct pick-and-place logic, there are several problems:

- The robot motion from the home position to the picking target is too large.
- Moving right after setting DO signals doesn't provide enough time for the robot to pick/place.
- It would be safer if the robot moves vertically down and up right before and after it performs picking and placing.

Now let's improve the workflow to resolve the problems.

Improve the Workflow

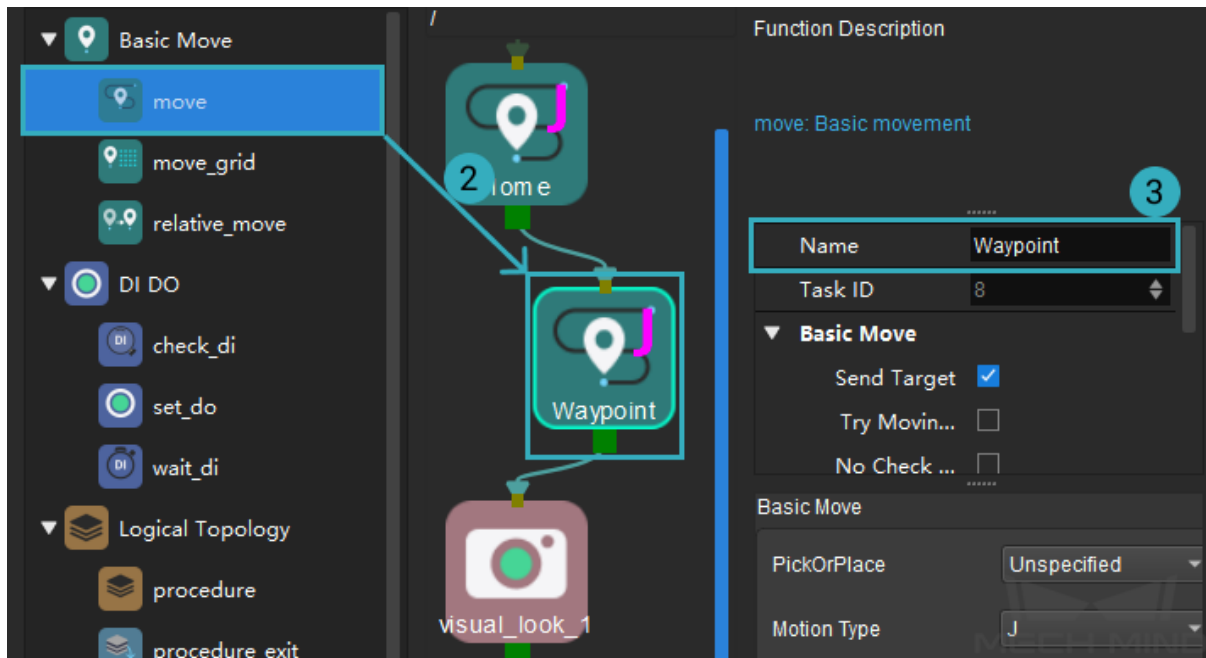
Add Waypoint

Add a waypoint between the home position and the picking target to break down the motion into two parts.

1. Move the real robot to the desired waypoint. Check the simulated robot to make sure the pose of the real robot has been synced.

Note: The robot should not block the view of the camera when its at this waypoint. You can capture an image to check if the robot is in the way.

2. From the **Basic Move** category in the Task library, drag the *move* Task to the programming area.
3. In the upper part of the parameter panel, rename the Task to **Waypoint** in the **Name** field.
4. Connect **Home** to **Waypoint** and **Waypoint** to **visual_look_1**.



Add Relative Targets for Picking

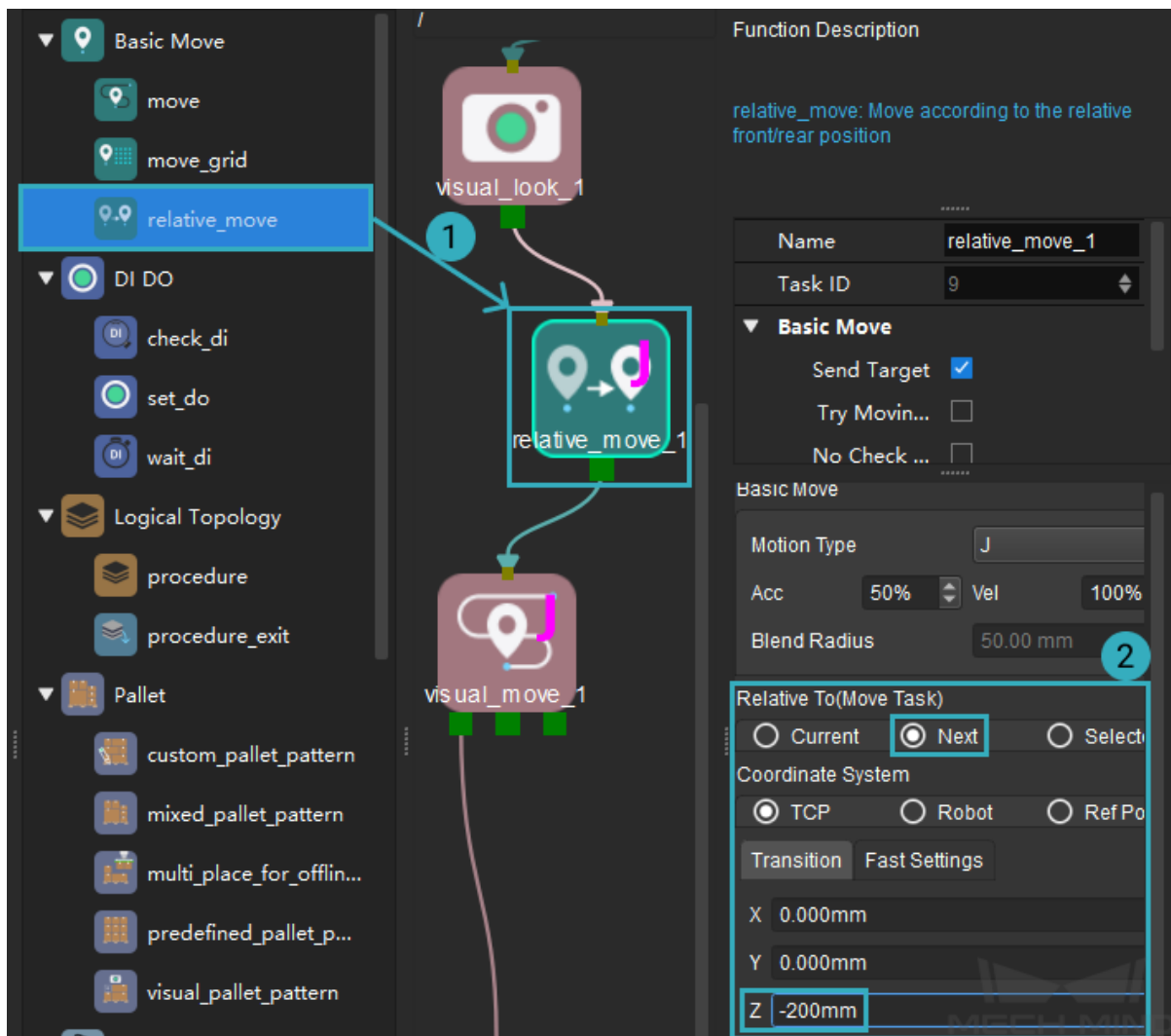
The more practical picking motion should be: the tool moves vertically down to the picking target, and then moves vertically up after it picks up the workobject.

The *relative_move* Task can be used to add targets relative to other targets.

Approaching the Picking Target

Move the robot to 200 mm above the picking target first, and then move it to the picking target from there.

1. From the **Basic Move** category in the Task library, drag the *relative_move* Task to the programming area. The Task is automatically named **relative_move_1**.
2. In the parameter panel, set the following fields:
 - **Relative To(move-type Task)**: select **Next**, which means that the target defined here is relative to the next target in the Workflow (i.e., the target of **visual_move_1**).
 - **Transition** tab: set the value of Z to -200.
3. Connect **visual_look_1** to **relative_move_1** and **relative_move_1** to **visual_look_1**.

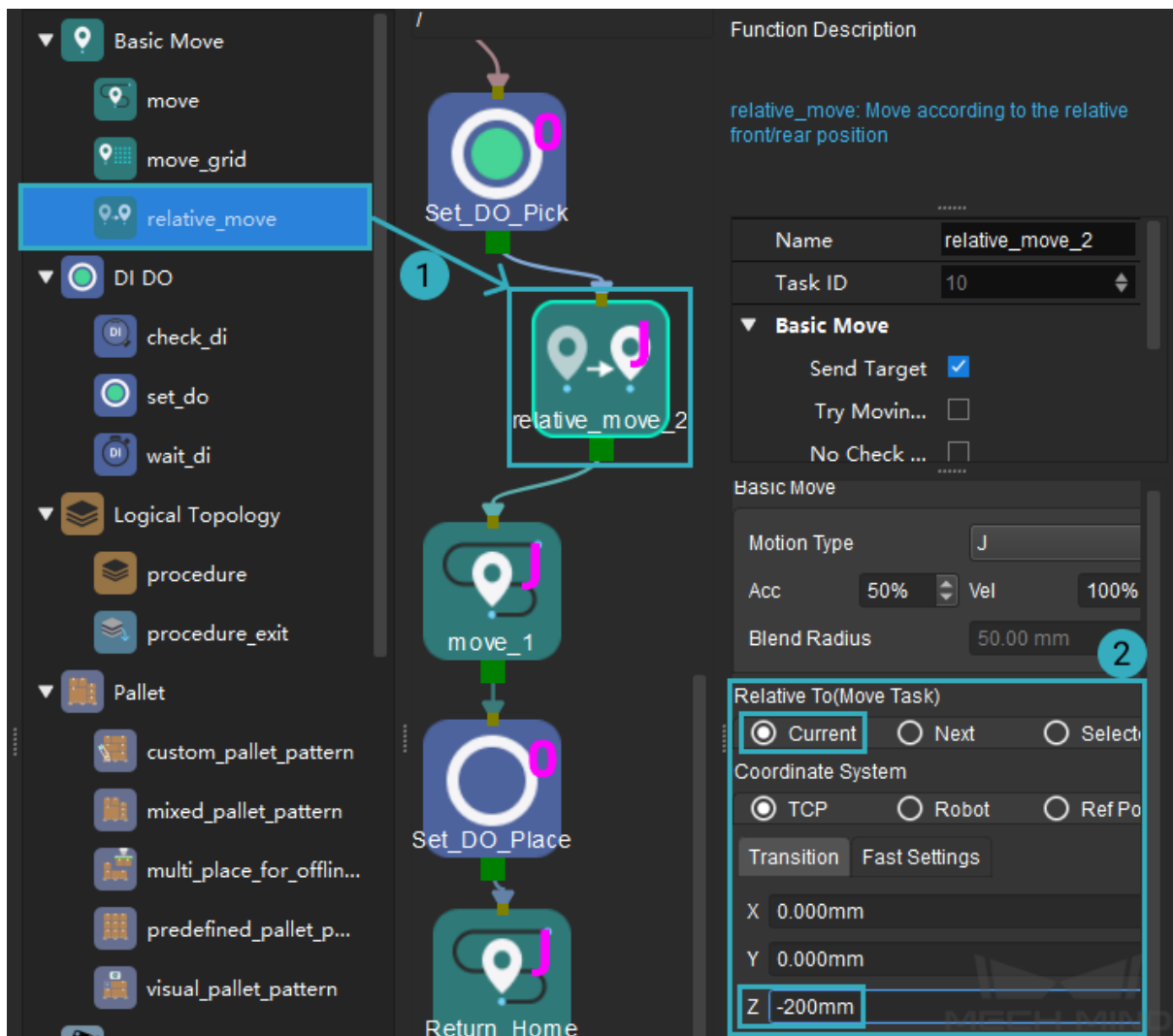


The screenshot displays the Mech-Viz software interface. On the left, a task library is visible with categories: Basic Move, DI DO, Logical Topology, and Pallet. The 'relative_move' task is highlighted in the Basic Move category. In the center, a workflow diagram shows three tasks: 'visual_look_1', 'relative_move_1', and 'visual_move_1'. A blue arrow labeled '1' points to the 'relative_move_1' task. On the right, the 'Function Description' and 'Basic Move' parameter panels are shown. The 'Basic Move' panel includes fields for Name, Task ID, Motion Type, Acc, Vel, and Blend Radius. A blue arrow labeled '2' points to the 'Blend Radius' field. Below this, the 'Relative To(Move Task)' section has radio buttons for 'Current', 'Next', and 'Select'. The 'Next' option is selected. The 'Coordinate System' section has radio buttons for 'TCP', 'Robot', and 'Ref Po'. The 'Transition' tab is active, showing 'Fast Settings' with X: 0.000mm, Y: 0.000mm, and Z: -200mm.

Departing the Picking Target

Move the robot to 200 mm above the picking target after it picks up the workobject.

1. From the **Basic Move** category in the Task library, drag the *relative_move* Task to the programming area. The Task is automatically named **relative_move_2**.
2. In the parameter panel, set the following fields:
 - **Relative To(move-type Task)**: select **Current**, which means that the target defined here is relative to the previous target in the Workflow (i.e., the target of *visual_move_1*).
 - **Transition** tab: set the value of Z to -200.
3. Connect **Set_DO_Pick** to **relative_move_2** and **relative_move_2** to **move_1**.



The screenshot displays the Mech-Viz software interface. On the left is a task library with categories: Basic Move (move, move_grid, relative_move), DI DO (check_di, set_do, wait_di), Logical Topology (procedure, procedure_exit), and Pallet (custom_pallet_pattern, mixed_pallet_pattern, multi_place_for_offlin..., predefined_pallet_p..., visual_pallet_pattern). The main workspace shows a task sequence: Set_DO_Pick, relative_move_2, move_1, Set_DO_Place, and Return Home. A red circle '1' highlights the 'relative_move' task in the library and its corresponding task in the sequence. A red circle '2' highlights the 'Basic Move' configuration panel for 'relative_move_2'. The 'Function Description' for 'relative_move' is: 'relative_move: Move according to the relative front/rear position'. The 'Basic Move' configuration panel includes: Name: relative_move_2, Task ID: 10, Send Target: checked, Try Movin...: unchecked, No Check ...: unchecked, Motion Type: J, Acc: 50%, Vel: 100%, Blend Radius: 50.00 mm. The 'Relative To(Move Task)' section has 'Current' selected. The 'Coordinate System' section has 'TCP' selected. The 'Transition' is set to 'Fast Settings'. The 'Z' coordinate is set to '-200mm'.

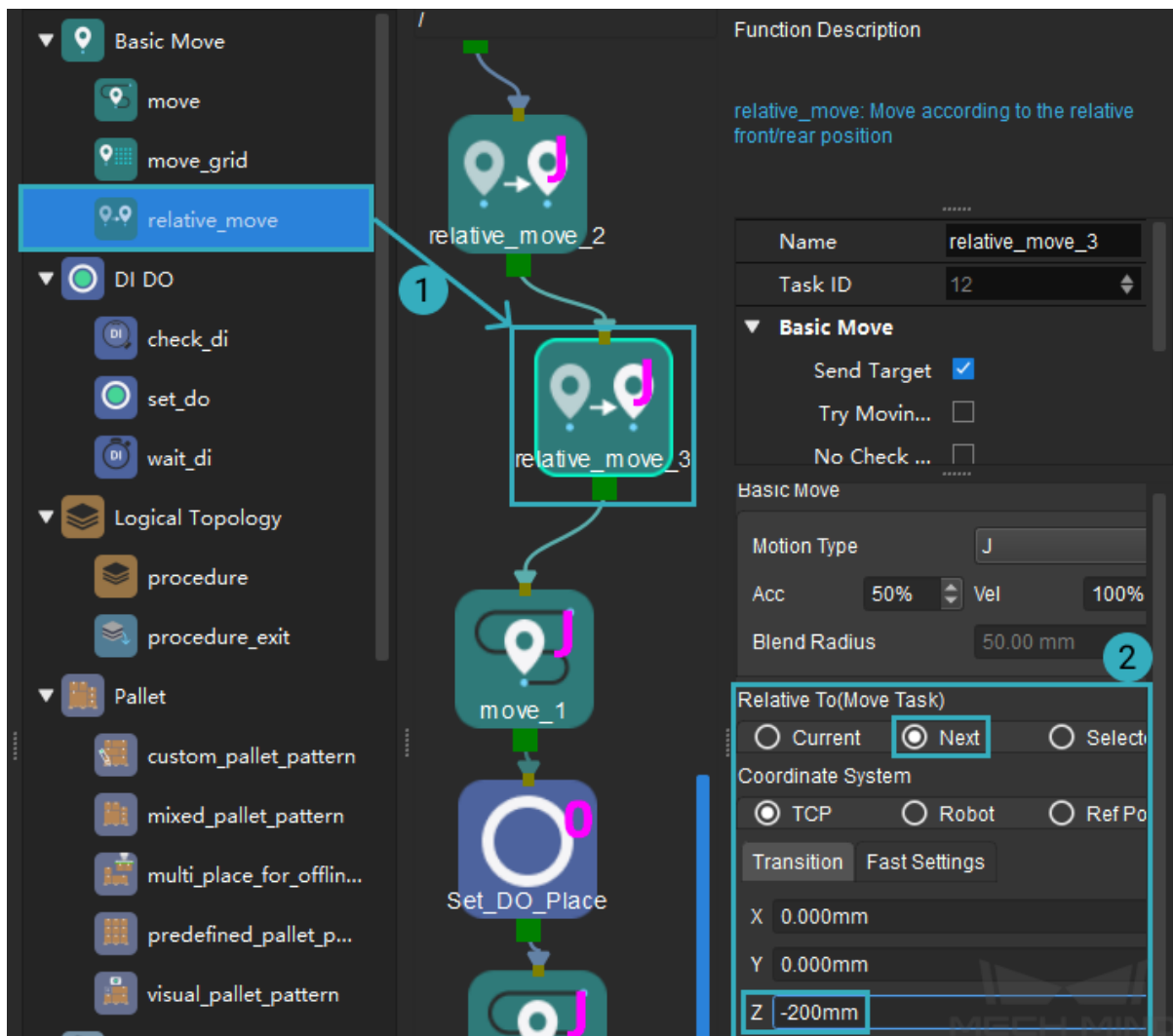
Add Relative Targets for Placing

Similarly, the more practical placing motion should be: the tool moves vertically down to the placing target, and then moves vertically up after it drops off the workobject.

Approaching the Placing Target

Move the robot to 200 mm above the placing target first, and then move it to the placing target from there.

1. From the **Basic Move** category in the Task library, drag the *relative_move* Task to the programming area. The Task is automatically named **relative_move_3**.
2. In the parameter panel, set the following fields:
 - **Relative To(move-type Task)**: select **Next**, which means that the target defined here is relative to the next target in the Workflow (i.e., the target of **move_1**).
 - **Transition** tab: set the value of **Z** to **-200**.
3. Connect **relative_move_2** to **relative_move_3** and **relative_move_3** to **move_1**.

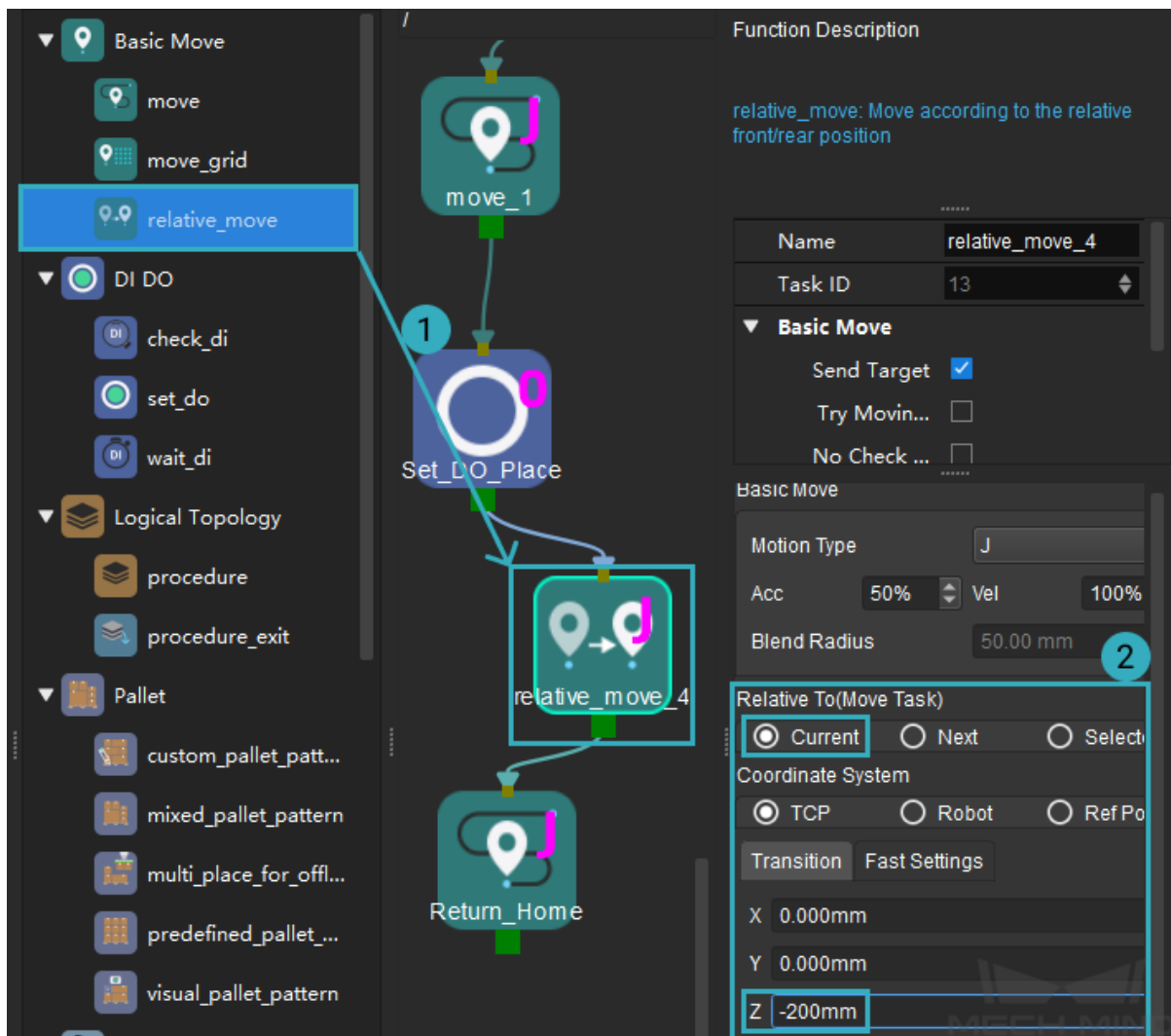


The screenshot displays the Mech-Viz software interface. On the left, a task library is visible with the 'Basic Move' category expanded, showing the 'relative_move' task. A red circle '1' highlights the 'relative_move' task being dragged into the workflow. The workflow in the center shows a sequence of tasks: 'relative_move_2', 'relative_move_3', 'move_1', and 'Set_DO_Place'. A red circle '2' highlights the 'relative_move_3' task in the workflow. On the right, the 'Function Description' and 'Basic Move' parameter panels are shown. The 'Basic Move' panel includes fields for 'Name' (relative_move_3), 'Task ID' (12), 'Motion Type' (J), 'Acc' (50%), 'Vel' (100%), and 'Blend Radius' (50.00 mm). The 'Relative To(Move Task)' section has 'Next' selected. The 'Transition' tab is active, and the 'Z' coordinate is set to '-200mm'.

Departing the Placing Target

Move the robot to 200 mm above the placing target after it drops off the workobject.

1. From the **Basic Move** category in the Task library, drag the *relative_move* Task to the programming area. The Task is automatically named **relative_move_4**.
2. In the parameter panel, set the following fields:
 - **Relative To(move-type Task)**: select **Current**, which means that the target defined here is relative to the previous target in the Workflow (i.e., the target of **move_1**).
 - **Transition** tab: set the value of Z to -200.
3. Connect **Set_DO_Place** to **relative_move_4** and **relative_move_4** to **Return_Home**.



The screenshot displays the Mech-Viz software interface. On the left, a task library is visible with categories like 'Basic Move', 'DI DO', 'Logical Topology', and 'Pallet'. The 'relative_move' task is highlighted in the 'Basic Move' category. In the center, a workflow diagram shows a sequence of tasks: 'move_1' (Basic Move), 'Set_DO_Place' (DI DO), 'relative_move_4' (Basic Move), and 'Return_Home' (Basic Move). A red circle with the number '1' is around the 'relative_move_4' task in the workflow, and a red circle with the number '2' is around its configuration panel on the right.

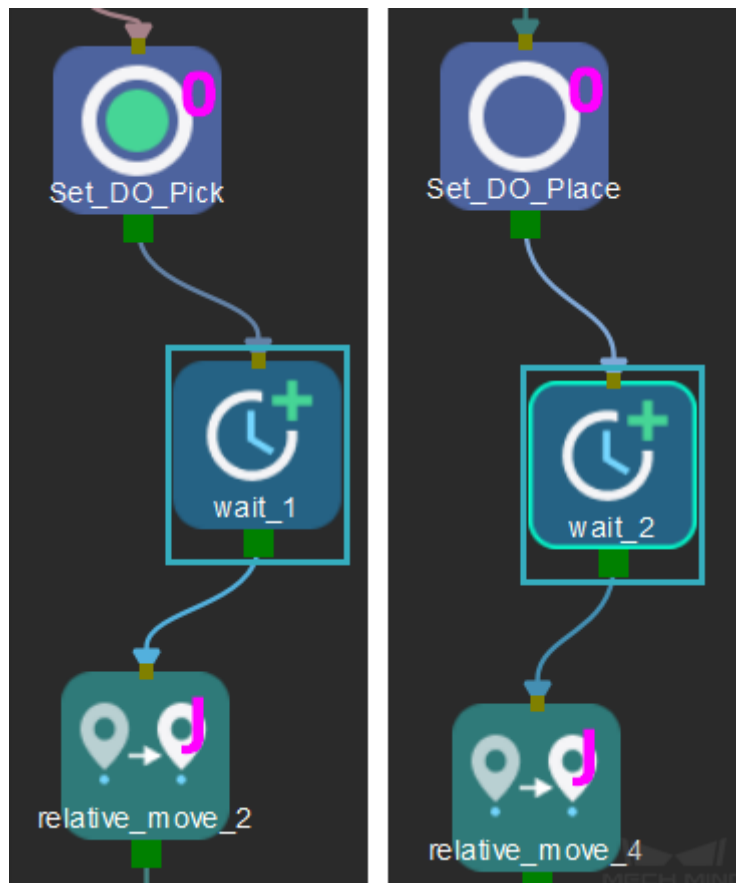
The configuration panel for 'relative_move_4' is shown on the right. It includes the following settings:

- Name:** relative_move_4
- Task ID:** 13
- Basic Move:**
 - Send Target:
 - Try Movin...:
 - No Check ...:
- Basic Move:**
 - Motion Type: J
 - Acc: 50%
 - Vel: 100%
 - Blend Radius: 50.00 mm
- Relative To(Move Task):**
 - Current
 - Next
 - Select
- Coordinate System:**
 - TCP
 - Robot
 - Ref Po
- Transition:** Fast Settings
- X:** 0.000mm
- Y:** 0.000mm
- Z:** -200mm

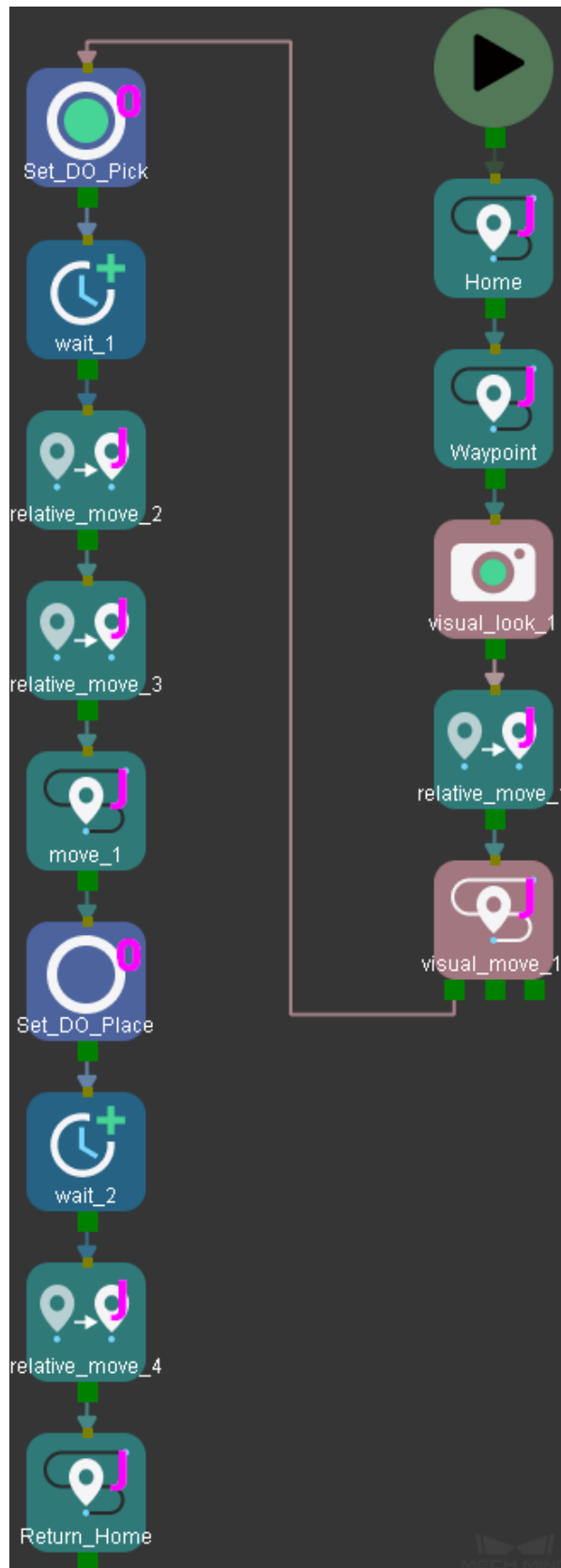
Wait after Setting DO Signals

Make the robot wait a little while after the DO signal is set to ensure that the picking and releasing actions are fully completed. This can be accomplished with the *wait* Task.

1. From the **Tools** category in the Task library, drag the *wait* Task to the programming area. The Task is automatically named *wait_1*.
2. Connect *Set_DO_Pick* to *wait_1* and *wait_1* to *relative_move_2*.
3. From the **Tools** category in the Task library, drag the *wait* Task to the programming area. The Task is automatically named *wait_2*.
4. Connect *Set_DO_Place* to *wait_2* and *wait_2* to *relative_move_4*.



The complete workflow should look like this:

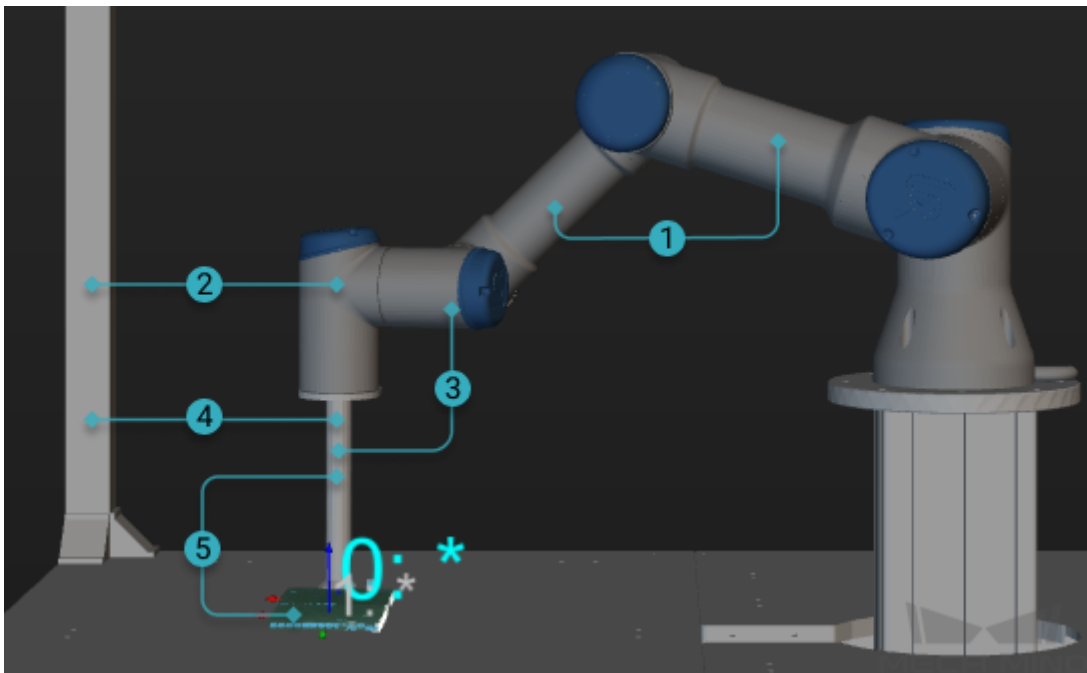


Click on *Simulate* in the toolbar, the simulated robot will move according to the workflow. You can see that the robot path is more smooth and safe now.

2.2.5 Configure Collision Detection

To prevent the robot from colliding with other objects while it performs its tasks, collision detection must be configured.

In the **Collisions** tab, click on *Collision Detection Configuration* to set between which two objects you wish to check for collision.



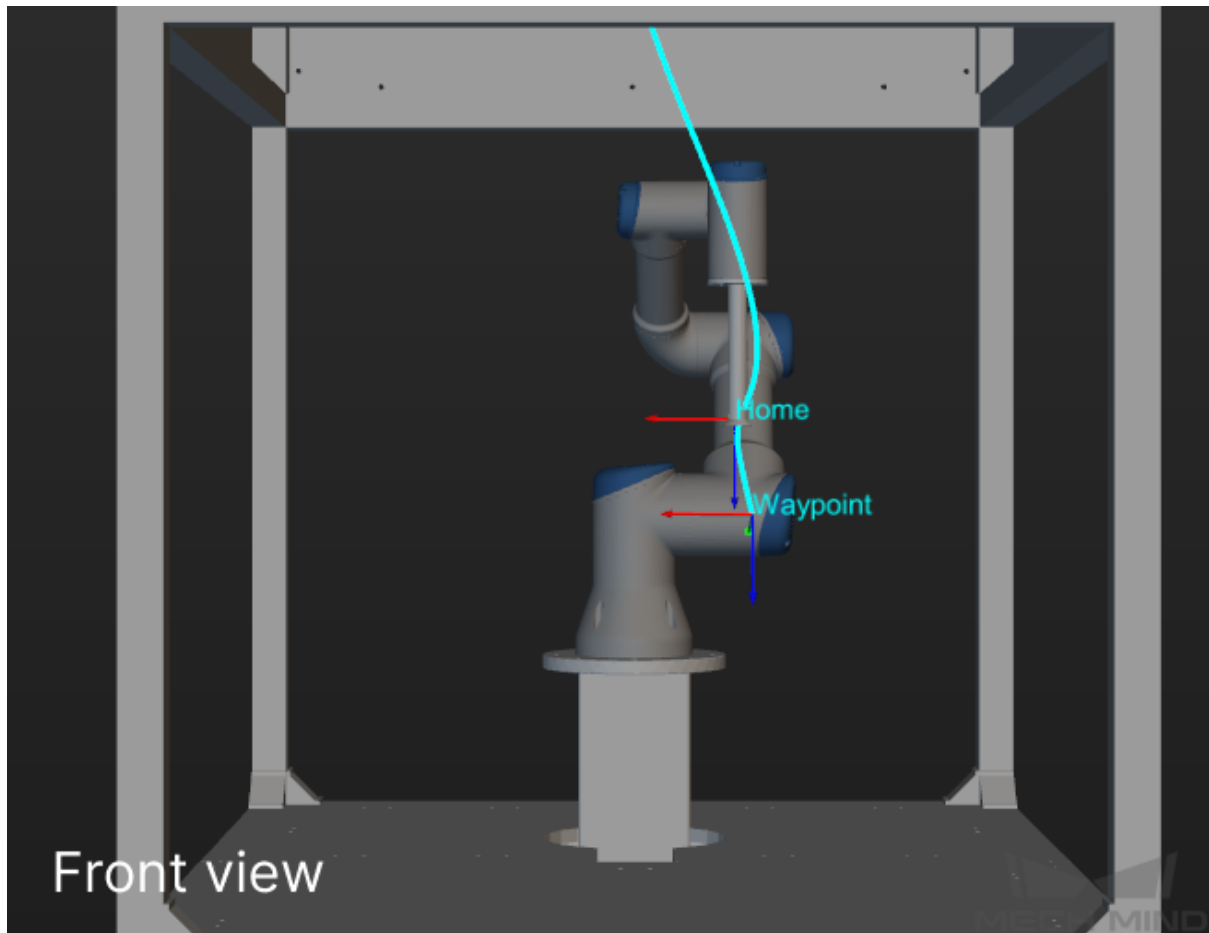
The following object pairs are checked for collision by default:

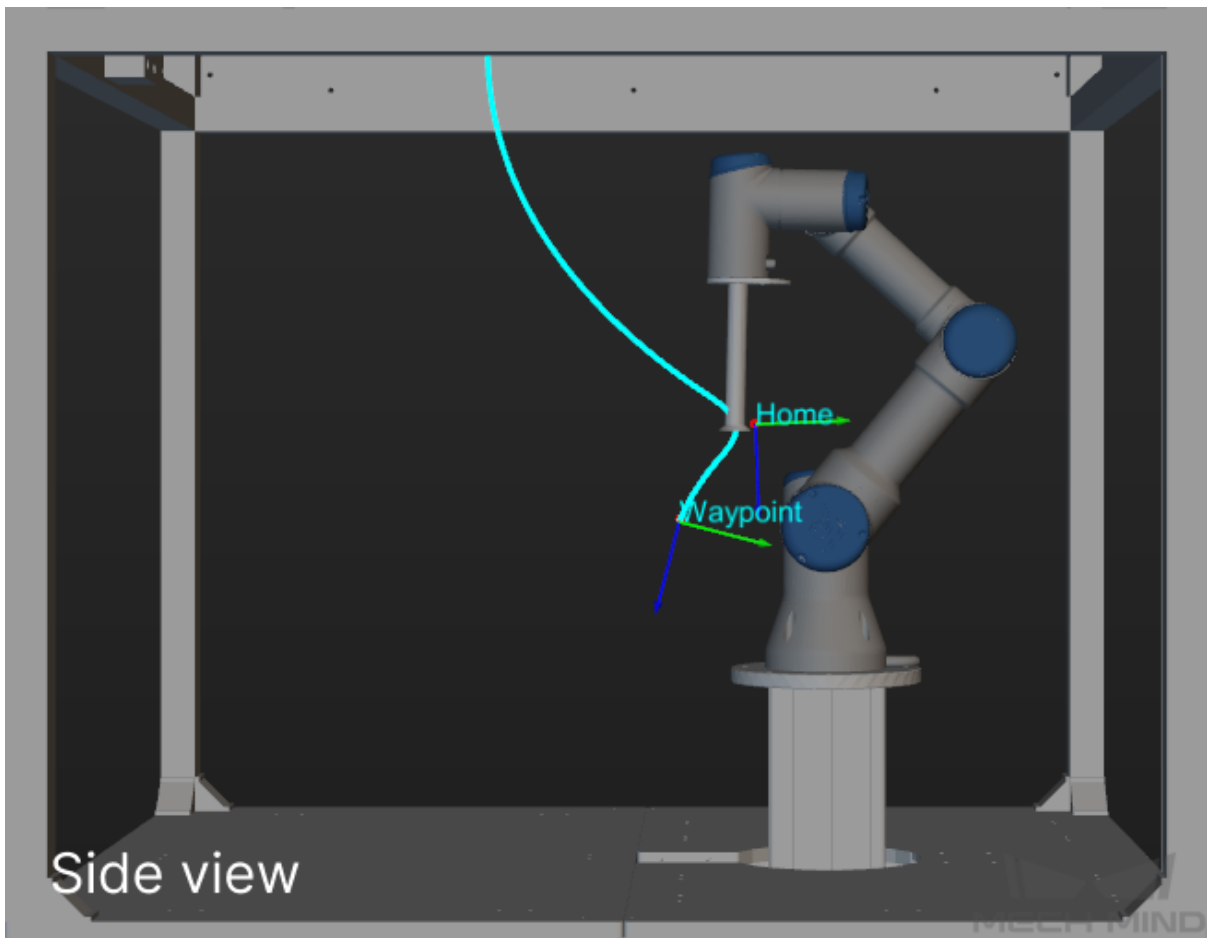
1. Robot links and robot links
2. Robot links and scene objects
3. Robot links and robot tool (end effector)
4. Scene objects and robot tool
5. If you turn on **Detect collision between point cloud and others**, the collision between point cloud and robot tool is also detected.

2.2.6 Simulate the Project

After finishing all the configurations in the previous chapters, click on *Simulate* to simulate the project.

The following are different views of the simulated robot's motion.





Hint: If errors are reported during simulation, you can check the *Plan History* and *Log* tabs to locate the errors.

Congratulations on creating your first Mech-Viz project!

You can check out the following chapters to learn more about each tab in the project workspace.

- *Workflow*
- *Scene*
- *Robot*
- *Tools and Workobjects*
- *Collisions*
- *Plan History*
- *Others*
- *Log*

Check the chapter below to learn about how to use Tasks, the building blocks of projects.

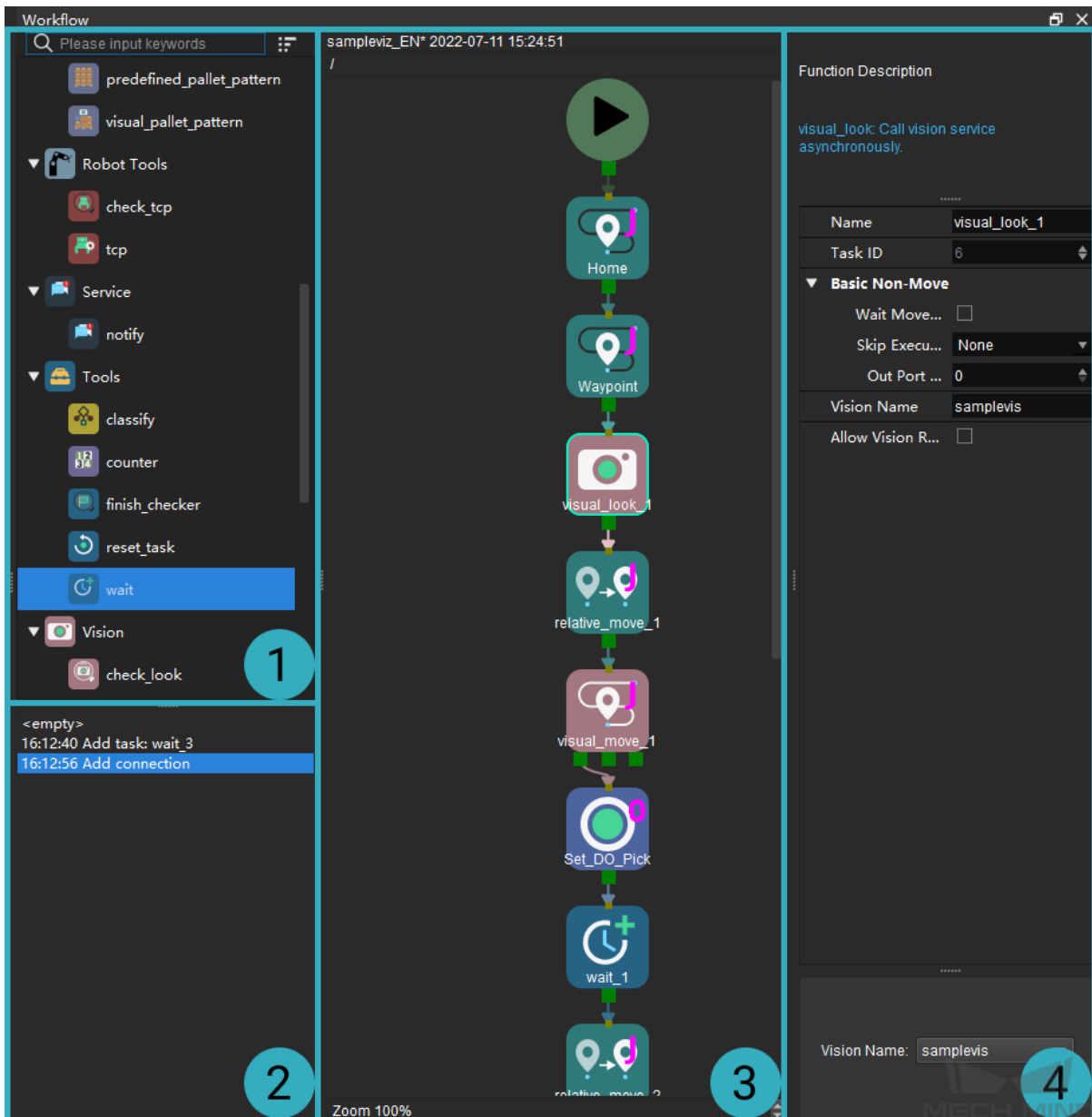
WORKFLOW

Tasks are fundamental units of a Mech-Viz project. In the workflow panel, you can connect different Tasks based on the logic of the programs to construct Mech-Viz projects that can meet requirements of various demanding applications, and you may also adjust the parameters of the Tasks for better performance.

Check the following section below for an **introduction of the Workflow interface**

3.1 Interface of Workflow

The Workflow panel is the main interface for robot programming, and it mainly consists of 4 areas.



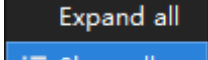
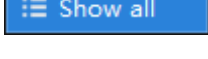

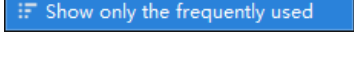
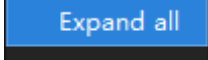
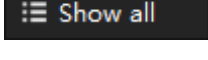
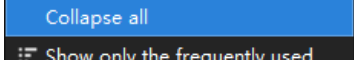
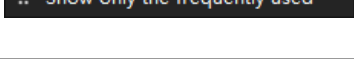


The Workflow tab consists of four areas:

1. Task library
2. Programming record
3. Graphical programming workspace
4. Parameter panel

1. Task Library

A Task is a graphical programming module used to enable a specific function. Tasks are the basis of robot programming in Mech-Viz. By default, only the frequently used Tasks are displayed in the Task Library.

	Click to display all Tasks
	Click to display frequently used Tasks only
 	Right click and select <i>Show all</i> in the context menu to display all Tasks
 	Right click and select <i>Show only the frequently used</i> in the context menu to display frequently used Tasks only
 	Right click and select <i>Expand all</i> in the context menu to expand all Task categories
 	Right click and select <i>Collapse all</i> in the context menu to collapse all Task categories

All Tasks are divided into various Task categories according to their functions. For example, “Basic Move” includes Tasks related to basic motions of the robot, such as *move*, *relative_move*, and *move_grid*.

In addition, the Tasks can be categorized as “move-type Tasks” and “non-move Tasks”, as shown in the table below.

Move-type Tasks	All Tasks that belong to “Basic Move” and “Pallet” categories and the Task <i>visual_move</i>
Non-move Tasks	All Tasks that are not categorized as “move-type Tasks”

2. Programming Record Area

All your actions in the graphical programming interface and parameter panel are recorded here.

Click on a record to return the workflow to that point in time.

Note: If you clicked on an older record and then performed some new action, the records newer than the one you clicked on are removed, and the new action is recorded right after the clicked record.

3. Graphical Programming Workspace

Here you can connect Tasks to program the robot. Selecting and connecting the Tasks properly is the core of constructing a Mech-Viz project. For detailed instructions, please refer to *Basic Features of Skills*.

4. Parameter Panel

Select a Task in the workflow to display all its parameters here.

If you check **Function Description** in the **View** menu, a brief description of the selected Task is also displayed at the top of the parameter panel.

Check the section below for an introduction to the basic features of Tasks

3.2 Basic Features of Skills

Tasks are divided into the following categories:

3.2.1 Entry Port and Exit Port

Each Task has an entry port and an exit port. Tasks are connected through these ports.



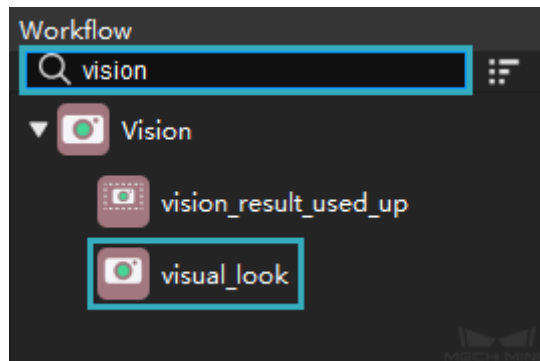
1. Entry port
2. Exit port

3.2.2 Search for, Add, and Delete a Task

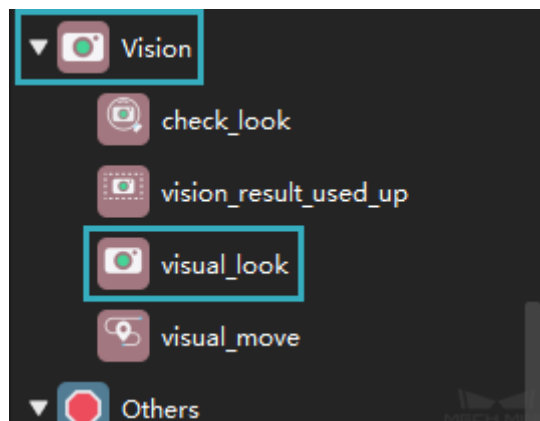
Using *visual_look* as an example:

Search for a Task

- In the search box at the top of the Task library, enter the name of a Task or a keyword.

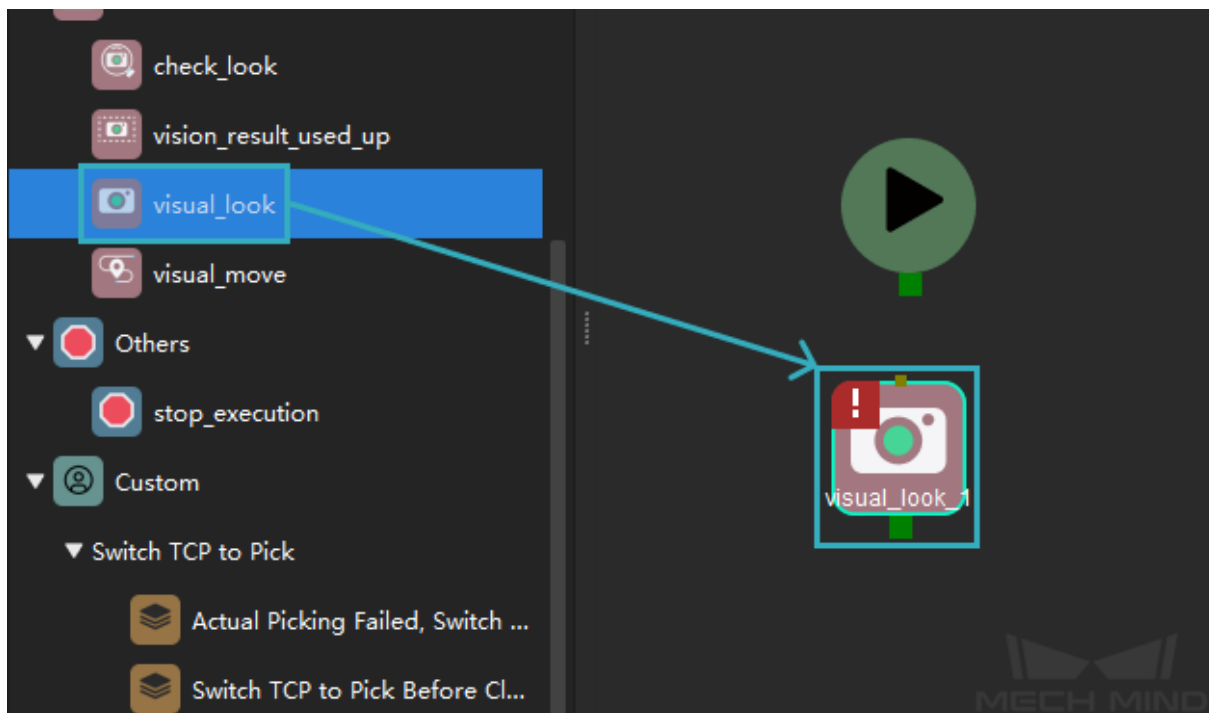


- Look in the relevant category for the Task you want.



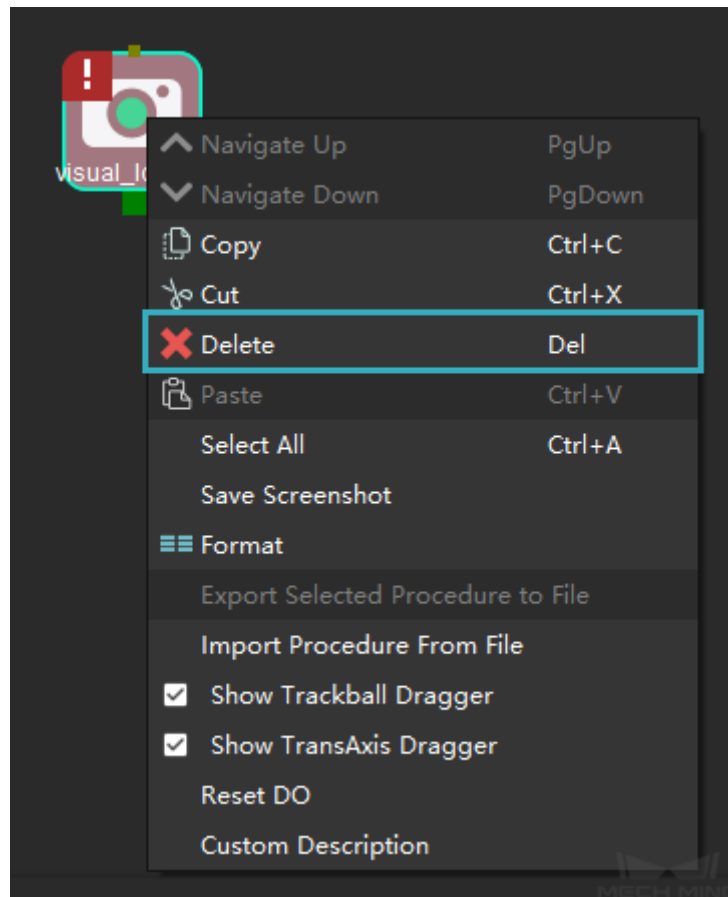
Add a Task

To add a Task to the workflow, drag it from the Task library to the graphical programming workspace.



Delete a Task

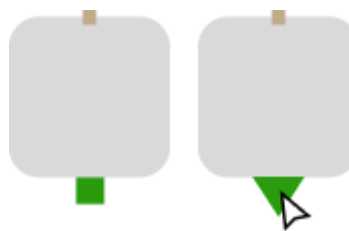
- In the graphical programming workspace, select the Task you want to delete, and then press the Delete key.
- Right-click on the Task you want to delete and select **Delete**.



3.2.3 Create and Delete Connections of Tasks

Create a Connection

1. Move the cursor onto the exit port of a Task, and the exit port icon changes from a square to a triangle.



2. Drag the triangle to the entry port of another Task to create a connection.



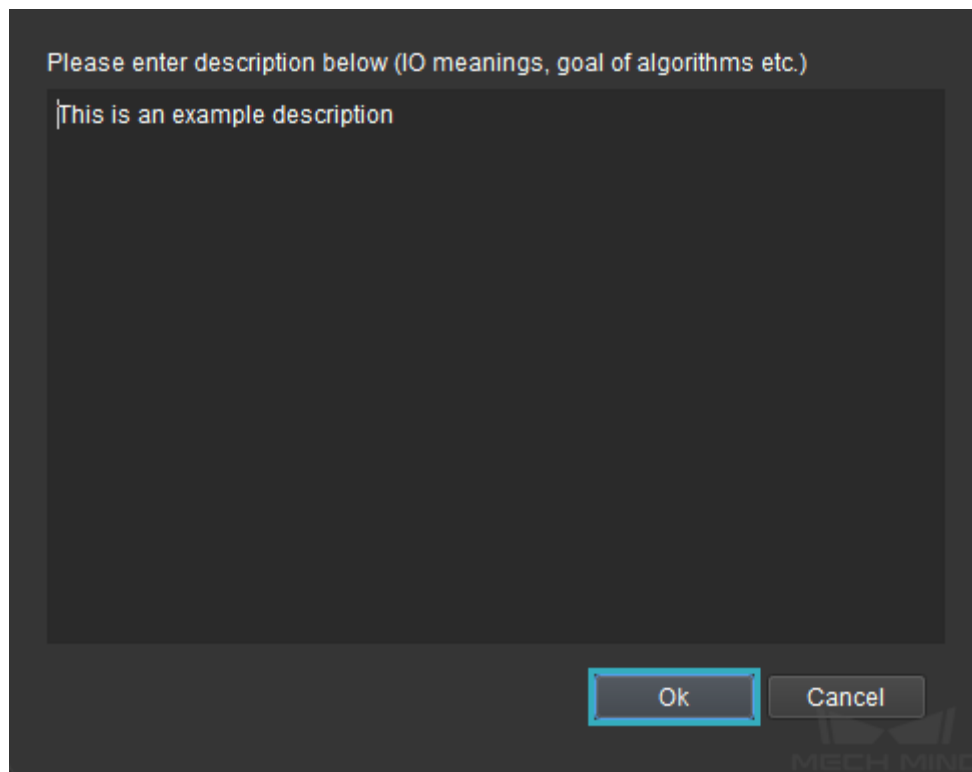
Delete a Connection

- Click on the connection you want to delete to select it, and then press the **Delete** key.
- Click on the connection you want to delete to select it, and then right-click and select **Delete**.

Hint: The color of a selected connection becomes brighter.

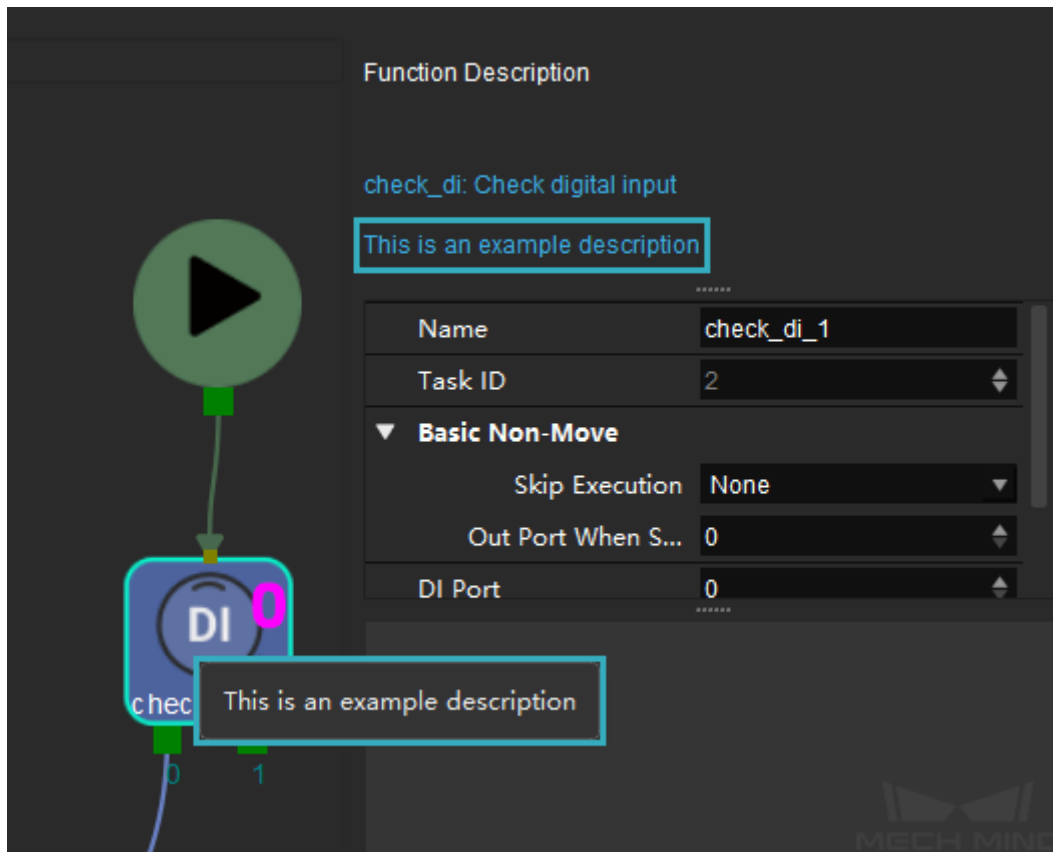
3.2.4 Customize Description

1. Right-click on a Task in the graphical programming workspace, and select **Custom Description**.
2. In the pop-up window, enter your description for the Task (such as things to note for other people) and click on **OK**.



To view the custom description:

- Hover the cursor on the Task, the description will show up.
- In the menu bar, enable **Function Description** in the **View** menu. Now when you select the Task, you can see the custom description at the top of the parameter panel.



Check the sections below for functions, applications and parameters of Tasks

3.3 General Parameters of Tasks

This chapter introduces you to the general parameters of Tasks.

3.3.1 General Parameters of Move-Type Tasks

Send Target Selected by default to send target poses to the receiver, such as the robot. When this option is unselected, the target pose will not be sent. However, the target will remain in the planned path.

Try Moving Smoothly through Following Non-Moves

Unselected by default. When non-move Tasks, such as *visual_look*, *set_do*, *check_di*, etc., are connected between move-type Tasks, the robot's path planning will be interrupted, and the actual robot will take a short pause, reducing the smoothness of running. When this option is selected, the project will continue to run without waiting for the current move-type Task to complete execution, and therefore the robot can move in a smooth way without pauses. However, enabling this option may cause the execution of the Task to end prematurely.

Note:

Why will this option cause the execution of the Task to end prematurely?

Mech-Viz will send multiple poses simultaneously to the robot when the project is running. When the currently returned JPs of the robot correspond to the last pose sent by Mech-Viz, Mech-Viz will assume that the robot has moved to the last position. For example, there are 10 move-type Tasks in a path, and the pose of move_5 is the same as that of the last move Task. When the robot moves at low speed, it sends JPs to Mech-Viz when it moves to move_5 Task, Mech-Viz may mistakenly determine that the robot has finished the move-type Tasks and prematurely ends the Tasks since the poses of the move_5 and the last Task are the same in the path.

No Check Collision with Placed Obj

Unselected by default, i.e., the collision with the already placed objects will not be detected. When this option is selected, the collisions between the robot, end effector, and placed objects will be detected.

In palletizing scenarios, the two possible cases of error are as follow:

1. In palletizing scenarios, when the robot is placing a carton, the carton to place may come into light contact with the placed cartons while no deformation will be caused. After Mech-Viz detects this collision in simulation, it will plan other positions for placing the carton, and therefore a full stack cannot be formed.
2. Usually, the TCP of a suction cup is inside the suction cup model instead of on the surface of it. Under this circumstance, the suction cup may be embedded in the model of the carton to be picked in the simulation of picking, while Mech-Viz does not detect the collision between the end effector and the object to be picked. After the robot places the object and the carton model turns into an object model in the scene, a collision between the suction cup and the carton will be detected and the palletizing cannot be completed.

When this option is selected, no collision between the robot, end effector, and the placed object will be detected, and the above two cases of errors can be avoided.

Note: Move-type Tasks contain the **Pick or Place** setting. Usually, move-type Tasks implemented before *visual_move* should be set to **Pick**, and those after *visual_move* should be set to **Place**.

Pcl Collision Check Mode Select the proper mode according to the requirement of the on-site situation. Usually, the default setting **Auto** can be used. **NotCheck** mode can be used in move-type Tasks before the robot picks the object, and **Check** mode can be used after the robot picks the object.

Auto: Default setting. Only the point cloud collisions for Tasks before and after *visual_move* and *visual_move* will be detected.

NotCheck: Point cloud collisions for all move-type Tasks will not be detected.

Check: Point cloud collisions for all move-type Tasks will be detected.

Attention: When *Collision Detection Configuration* → *Detect collision between point cloud and others* is enabled, Mech-Viz will detect collisions between the robot, end effector, and point cloud when planning the path. Normally, Mech-Viz detects whether the robot collides with other objects during picking and placing. When there are point cloud outliers, non-existing collisions will be detected, which leads to errors in path planning.

Disable Object Symmetry This parameter will only take effect when **Move Target Type** of the Task is set to **Obj Pose**.

None: Default setting, i.e., do not disable symmetry on any axis.

AxisZ: Only disable Z-axis symmetry.

AxisXy: Only disable X-axis and Y-axis symmetry.

All: Disable symmetry on all axes.

Once the object symmetry is disabled, the robot will place the objects strictly according to the target poses.

Note: In some special cases, objects are not pickable due to their peculiar poses. Setting **Rotation Symmetry** under *Tools and Workobjects* → *Configuration of Objects to Pick (for move planning)* may solve this problem. Based on the settings of Rotation Symmetry, when the default object pose is not pickable, Mech-Viz will calculate candidate poses and determine whether the candidate poses are pickable. As the candidate poses calculated based on the settings of Rotation Symmetry are different from the default one, the consistency of the objects' place poses cannot be guaranteed.

Picked Object Collision Check Mode

Not Check Collision with Scene Object/Robot

Unselected by default. Once this option is selected, the collisions between the picked objects with the **scene objects/robot** will not be detected, and therefore the calculation workload of collision detection will be reduced, the planning speed can be increased, and the cycle time can be shortened. It is usually enabled in the first one or two move-type Tasks after the robot picks the object.

Please enable this option cautiously as there may be collision risks.

Note:

- When **Detect collision between picked object and others** under *Collision Detection Configuration* → *Picked Object Configuration* is enabled, Mech-Viz will detect whether the model of the picked object collides with the models of the scene objects and the robot.
- In palletizing projects, the calculated carton dimensions have millimeter-level errors with the actual dimensions, and frictions between cartons may occur during picking but no collisions will occur. For some move-type Tasks that will obviously not cause collisions, detecting such collisions only adds to the calculation workload and planning time, and consequently extending the cycle time.
- In palletizing projects, enabling **Not Check Collision with Scene Object** does not affect the collision detection between the picked carton and the placed cartons. This option can be enabled when there are scene objects under the stack to avoid failure of finding the palletizing solution.

Not Check Collision with Cloud

Unselected by default. Once this option is selected, the collisions between the **picked object models** with the **point clouds in the scene** will not be detected, and therefore the calculation workload of collision detection will be reduced, the planning speed can be increased, and the cycle time can be shortened.

Note:

- When both **Detect collision between picked object and others** under *Collision Detection Configuration* → *Picked Object Configuration* and **Detect collision between point cloud and others** under *Collision Detection Configuration* → *Point Cloud Configuration* are enabled, Mech-Viz will detect whether the model of the picked object will collide with the point cloud of the scene.
 - When Mech-Vision sends the point cloud and object model to Mech-Viz, the point cloud and the object model are fitted together. After the robot picks the object, the model moves along the planned path, and the collision between the model of the picked object and the point cloud will occur.
 - It is known that the model of the picked object will have false collisions with the point cloud. Detecting such collisions unnecessarily adds to the calculation workload and extends the planning time.
-

3.3.2 General Parameters of Non-Move Tasks

Skip Execution

None: Default setting. Do not skip current Task.

WhenSimu: Skip current Task during simulation. The exit port is specified by **Out Port When Skip**.

Always: Skip current Task during simulation and running the actual project. The exit port is specified by **Out Port When Skip**.

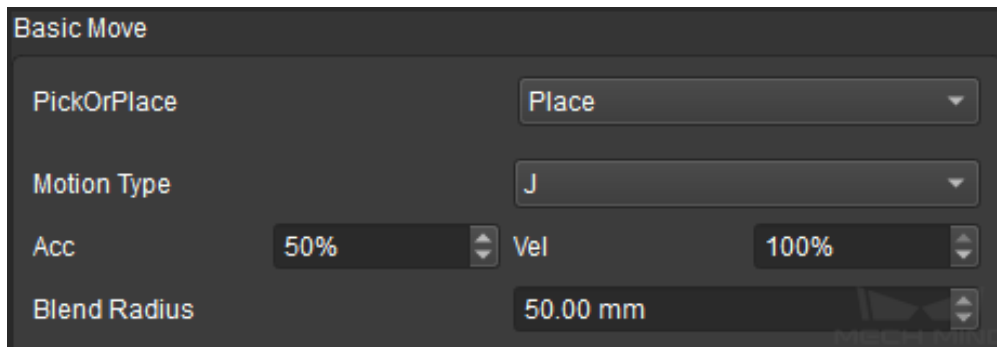
Instruction: When **WhenSimu** or **Always** is selected, the current Task will be skipped and the subsequent Task will be executed when running the project. If this parameter is set to **None** in *check_di* Task in the project, and there is no external input signals during simulation, the project will be stopped when executing to this Task. Set the parameter to **WhenSimu** or **Always** in the *check_di* Task and then the project will continue to simulate.

Out Port When Skip

This parameter will take effect when **Skip Execution** is set to **WhenSimu** or **Always**. It specifies the exit port when skipping a Task.

3.3.3 Basic Move Parameters

Basic Move parameters of move-type Tasks are used to set the **velocity** and **move type** for the robot when it moves to the target.



Move Target Type

J: Joint motion, which guides the robot to move in a curved path. It is less likely to reach singularities in the path for joint motion. This motion type is applicable to scenarios where the requirement of path accuracy is not strict and the robot moves in a large space.

L: Linear motion, which guides the robot to move linearly. This motion type is applicable to scenarios where there is a strict requirement of the path accuracy, such as welding, gluing, and certain types of picking.

Acc (Acceleration) & Vel (Velocity)

Default setting: The default acceleration is set to 50%, and the default velocity is set to 100%.

Instruction: The acceleration and velocity specify the cycle time of the robot. The acceleration value is usually set below the velocity, or else the robot may not move steadily.

Attention: The velocities of *visual_move* and its prior and subsequent Steps should be relatively low to make sure that the objects can be picked steadily.

Blend Radius

Default setting: 50.00mm

Definition: The blend radius refer to the distance between the target and the point where the robot starts to turn. The larger the blend radius, the more smoother the robot motion transitions are.

Instruction: Usually, the default setting can be used. If the robot moves in a relatively small space, please set the blend radius to a smaller value. If the robot moves in a relatively large space without obstacles and the distance between two consecutive path segments are long, please set the blend radius to a larger value.

PickOrPlace

Default setting: Unspecified

Pick: Please select this option when the move-type Task is before *visual_move*.

Place: Please select this option when the move-type Task is after *visual_move*.

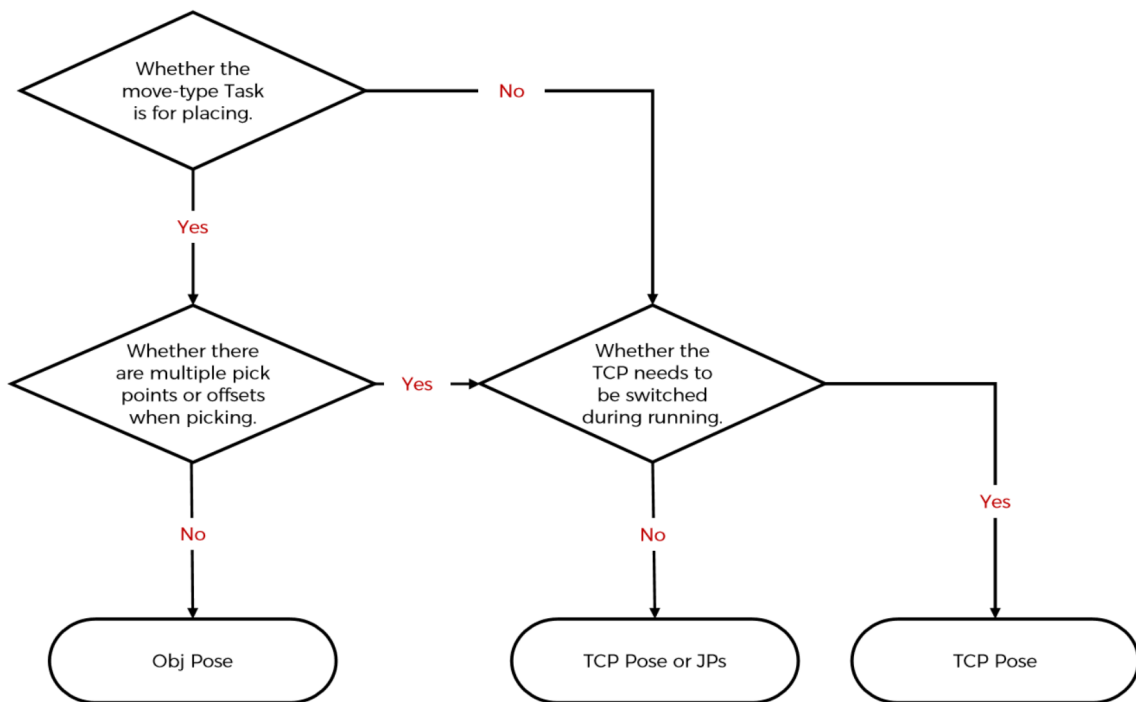
Instruction: The pick-or-place setting can be used to check the logic of the Mech-Viz project.

Please set the corresponding move-type Task to Pick or Place based on the execution logic and make sure that the Place Task follows after the Pick Task.

3.3.4 Move Target Type

- TCP: The move target will be represented by the X, Y, Z values and Euler angles or quaternions of the tool coordinate system.
- JPs: The move target will be represented by the joint positions of the robot.
- Obj Pose: The move target will be represented by the X, Y, Z values and Euler angles or quaternions of the object coordinate system.

The method to determine the move target type is shown in the figure below.



Hint: Click the corresponding button to enter the pop-up window for related settings.

- TCP & Obj Pose

Move Target Type

TCP Pose
 Jps
 Obj Pose

Jps Constraint

Shoulder:
 Elbow:
 Wrist:

X:

Y:

Z:

Quaternion
 Euler angles

Z: -60.00°

Y': 0.00°

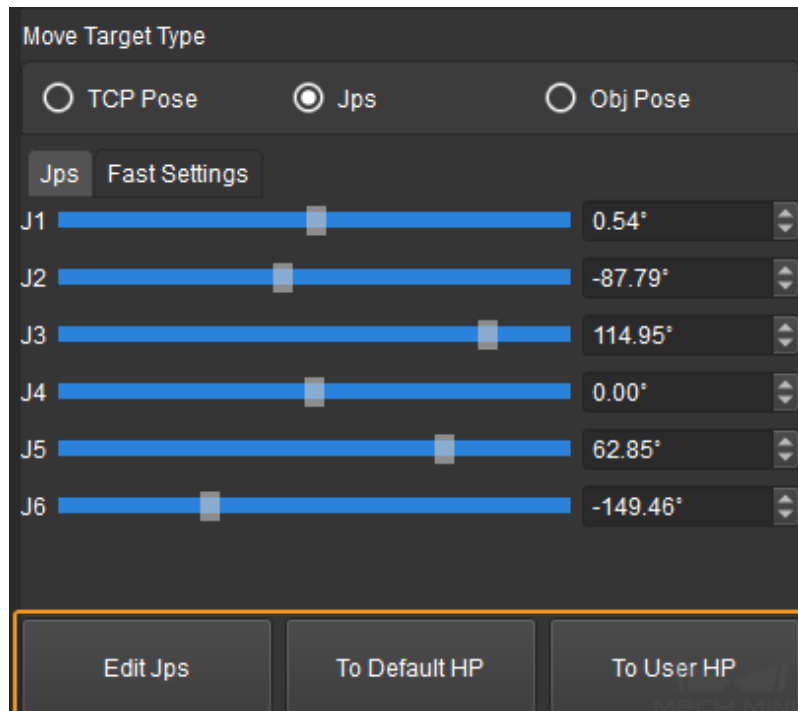
X": 180.00°

Edit Pose: Enter quaternion or Euler angle directly to adjust the pose. You can also copy and paste the pose.

Transform Pose: Transform the current pose to a new one. It is suitable for fine-tuning.

Calibrate Pose: Set the coordinate of P1, P2, and P3 according to the instruction, and then calibrate the object pose with three-point method. It is suitable for scenarios where the objects are tilted and their poses cannot be easily determined. For instance, when a cuboid tilts, its pose is hard to determine. Calibrate pose can be used here to calculate the cuboid's pose and therefore the robot can run based on the calibrated pose.

- JPs

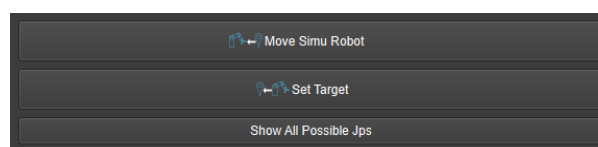


Edit JPS: Similar to **Edit Pose** above; edit the JPs by copying and pasting or editing the JPs in either radians or degrees, which can be switched based on the actual requirement.

To Default HP: Click to bring the robot back to the home position specified in the robot configuration file (i.e. the default home position in Mech-Viz).

To User HP: Click to bring the robot back to the home position defined by the user. The user home position can be set in *Robot* → *Joint Positions*. If the user home position is not set, the default home position in Mech-Viz will be used.

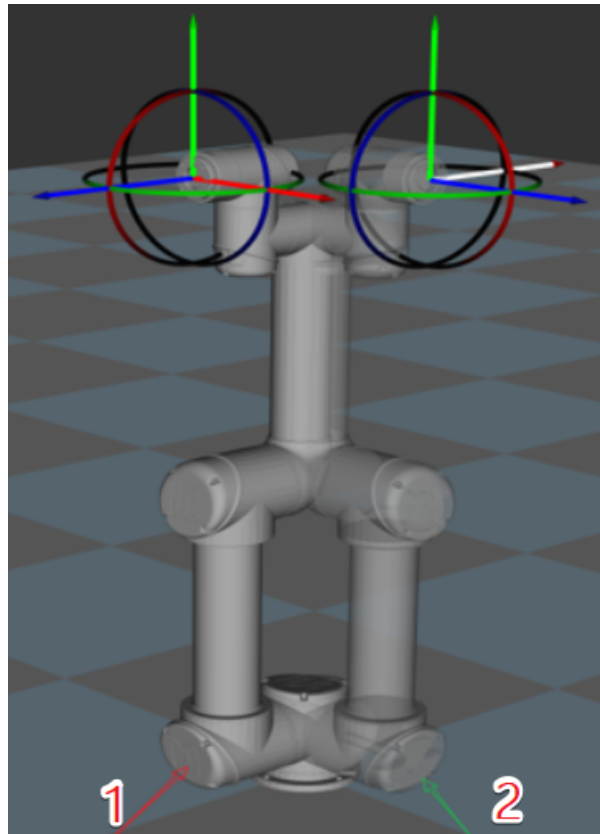
- Fast Settings



Move Simu Robot: Move the simulated robot model to the current target position of the real robot, that is, move the simulated robot model from position 1 to position 2. Only the position of the simulated robot will be changed and the position of the real robot will not.

Set Target: Set the move target to the position of the simulated robot model, that is, set the move target from position 1 to position 2. The position of the simulated robot will not be changed, while the move target will be changed.

Show all possible JPs: Show all the JPs solutions to the current move target. The maximum number of solutions is eight. Click *Move Simu Robot* to move the robot to the pose of the optimal solution.



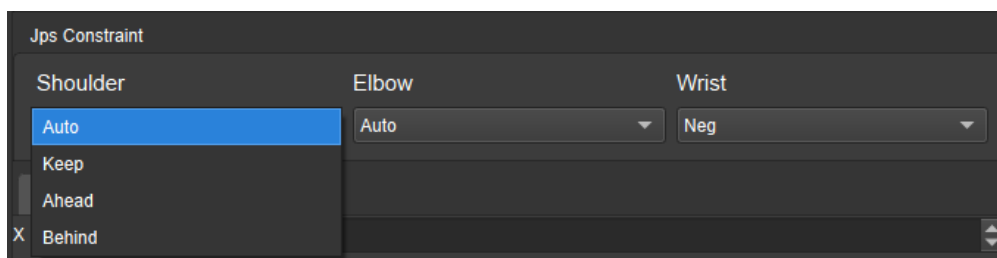
3.3.5 JPs Constraint

Definition of Terms

Shoulder: The relative positional relationship between the center of the wrist and Axis1. Axis1 refers to the rotation center axis of the robot axis 1.

Elbow: Relative positional relationship between wrist and LowerArm. LowerArm refers to the line connecting the rotation centers of the robot axis 2 and axis 3.

Wrist: Wrist refers to axis 5 of the robot. The negative wrist JPs suggests that the robot's wrist can rotate in the negative direction.



Options

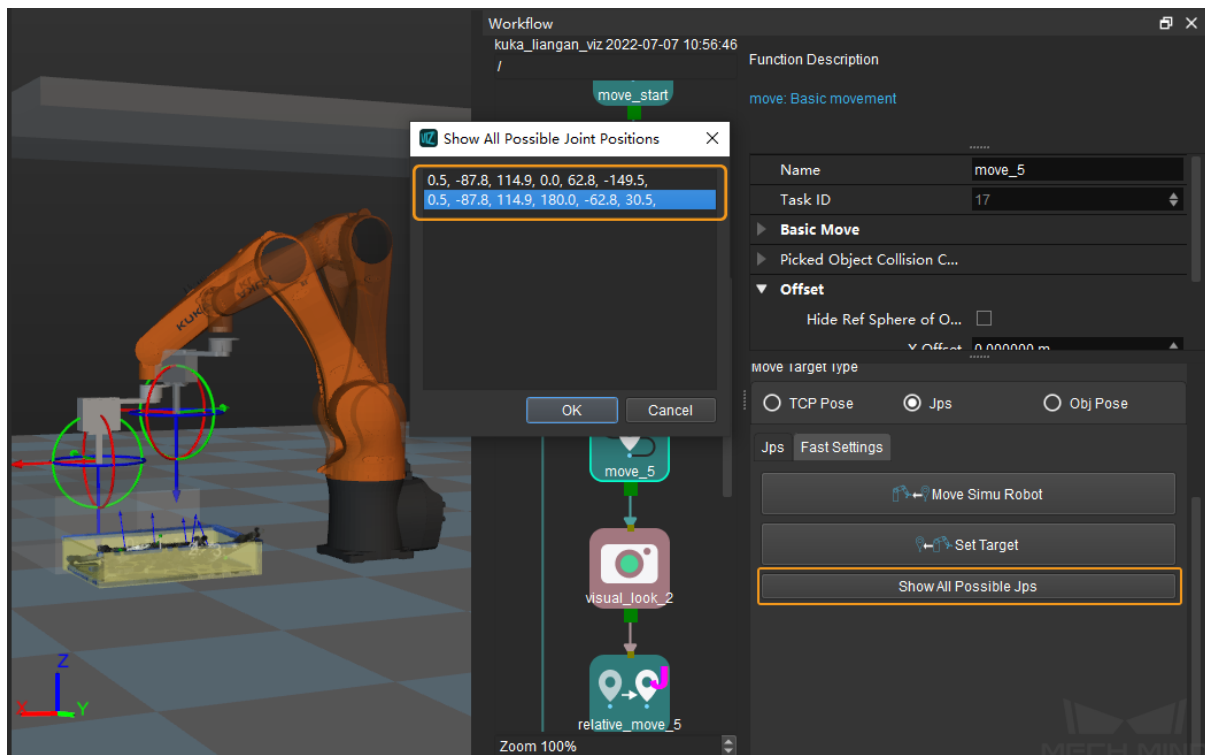
Auto: There is no constraints on the motion of this joint. The optimal solution is the one in which each axis rotates the least.

Keep: Constrain the next JPs solution according to the current JPs status. For example, if the current JP of the axis 3 is positive, only solutions in which the JP of axis 3 is positive will be considered for the next target.

Ahead: The wrist center is ahead of Axis1.

Behind: The wrist center is behind Axis1.

After clicking *Fast Settings* → *Show All Possible JPs*, a window showing all the JPs solutions to the current target will pop up. After clicking one of the solutions, the corresponding pose of the solution will be displayed in the 3D simulation area. Therefore, you will know the possible solutions under different constraints, as shown below.



Attention:

1. JPs constraint settings only apply to 6-axis robots. It is considered that 4-axis robots do not have abnormal shoulder, elbow, or wrist positions.
2. *relative_move*, *smart_pallet_pattern*, and *mixed_pallet* do not support JPs constraint settings. There are default JPs constraint settings for these Tasks, that is, the statuses of the shoulder, elbow, and wrist will not change, and the robot's motion does not cross solution systems (does not go through singularities).

3.4 Basic Move

Mech-Viz motion Task is a basic Task that controls the robot to complete various actions. By setting the parameters, different brands of robots can achieve the desired motion effect.

3.4.1 dynamic_move

Description

Used in eye in hand mode, the initial pose is the highest pose, and the subsequent motion changes dynamically with the height of the object in the field of view.

Parameters

BasicMove See *General Parameters of Move-Type Tasks*

relativeZ Based on the height of the highest object acquired by vision, it will calculate the height required for this movement according to this parameter. This height cannot be greater than the height of the initial pose, and the minimum cannot be less than *minZ*.

minZ The minimum height in Z direction that the robot is actually allowed to reach.

Basic Move See *Basic Move Parameters*

3.4.2 move

Description

Set a target pose in the robot's motion path and the way to move to that pose

Parameters

BasicMove See *General Parameters of Move-Type Tasks*

hideRefSphere Hide reference ball model.

Offset X/Y/Z offset: if the movement type is TCP Pose or Obj Pose, it is the X/Y/Z offset relative to the specified point in the corresponding reference system.

Example: if the Z-direction offset is 0.2 m, when it is TCP, the actual position of the robot's movement is the position where the specified point extends 0.2 m in the positive direction along the z-axis of the TCP coordinate system.

Basic Move See *Basic Move Parameters*

3.4.3 move_grid

Description

Move the points in a specified array

Parameters

Basic parameters of move-type Tasks See *General Parameters of Move-Type Tasks*

Index

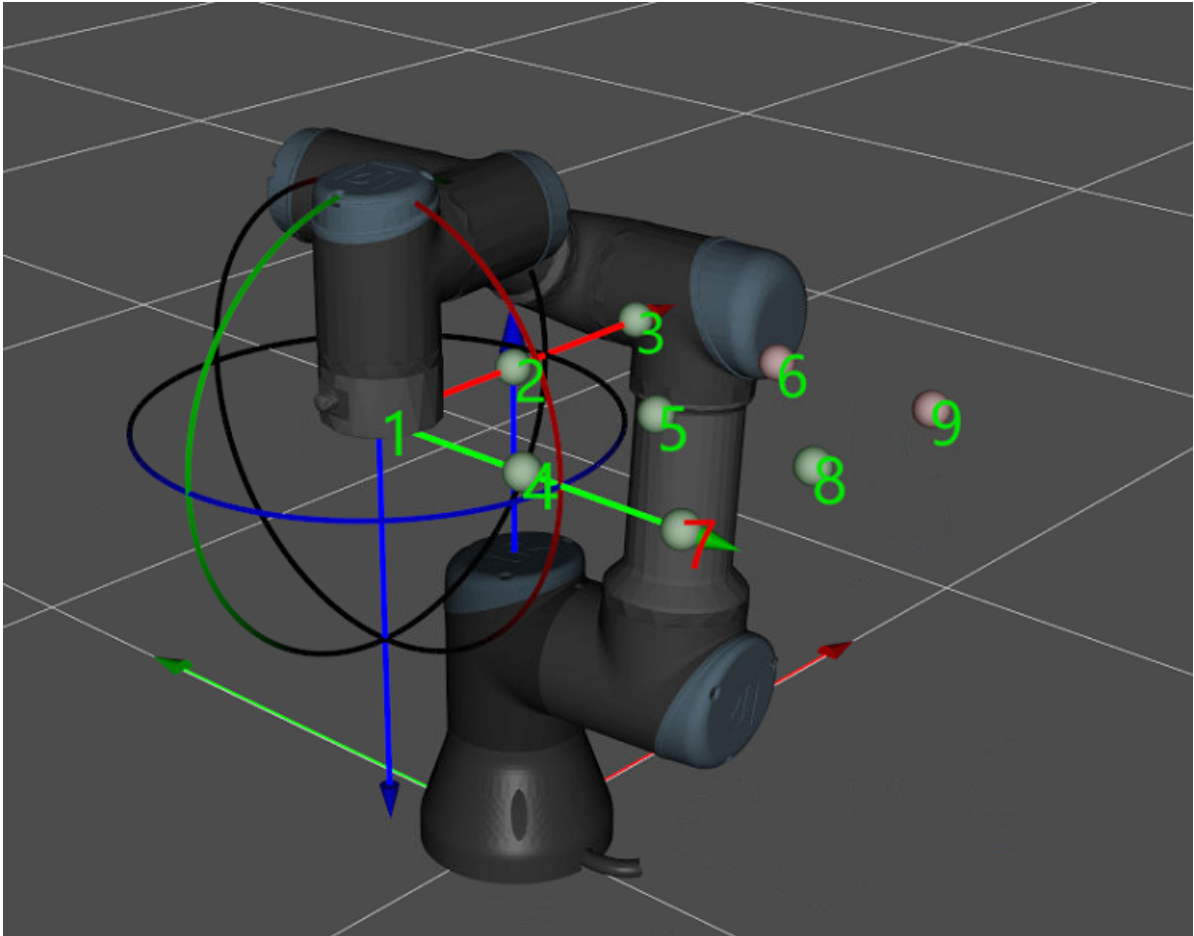
- **startIndex**: array from a specified starting point.
- **curlIndex**: the number of points in the current array.

Grid

- **xCount**: the number of points in the X direction of the array.
- **xSpace**: the interval between every two points in the X direction of the array.
- **yCount**: the number of points in the Y direction of the array.
- **ySpace**: the interval between every two points in the Y direction of the array.
- **zCount**: the number of points in the Z direction of the array.
- **zSpace**: the interval between every two points in the Z direction of the array.

Basic Move See *Basic Move Parameters*

Note: Select Show Base. Array points and the number of each point are shown in Mech-Viz, as shown in the figure:



Show target grid

Base Pose Determine the pose of the array starting point by X, Y, Z coordinates and quaternions or Euler angles (the overall array will rotate with it).

3.4.4 move_list

Description

Move by the points in a specified sequence

Parameters

Basic & BasicMove See *General Parameters of Move-Type Tasks*

Index

- **startIndex**: array from a specified starting point.
- **curlIndex**: the number of points in the current array.

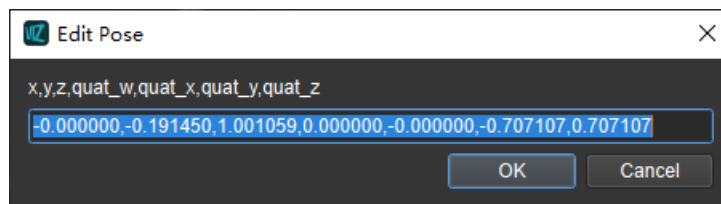
move_list When it is checked as True, all motion points in the sequence will be gone through at one time.

visionTrajPose When it is checked as True, it is considered as a visually planned path sequence. This function is often used for path scenarios such as glue coating.

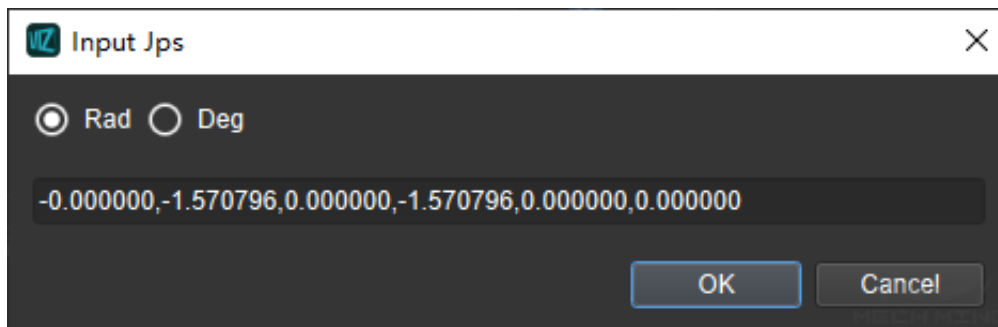
useGlobalMotionParams Whether all motion points in the path use global motion parameters.

Basic Move See *Basic Move Parameters*

Add TCP As shown in the figure below, inputting the tool pose by rows can add a new movement point.



Add Jps As shown in the figure below, inputting the joint angle by rows can add a new movement point.



3.4.5 outer_move

Description

Perform target pose to be moved obtained from external services

This Task is used by Mech-Viz to obtain the target pose moved from an external service and run the robot to that pose. It must be used in conjunction with the adapter

Parameters

Basic parameters of move-type Tasks See *General Parameters of Move-Type Tasks*

Service Name The external server name registered by the adapter on the Mech-Center. This parameter must be unified with the adapter to obtain the Task interface via the server name and send to the destination.

get J By default it is False, the position where the last planning ended will be used for the initial position of the software planning trajectory; if it is checked as True, the initial position of the software planning trajectory will be updated to the robot joint position obtained from the external. It is usually used when Mech-Viz does not fully control the robot motion

3.4.6 relative_move

Description

Relative movement based on the known motion points

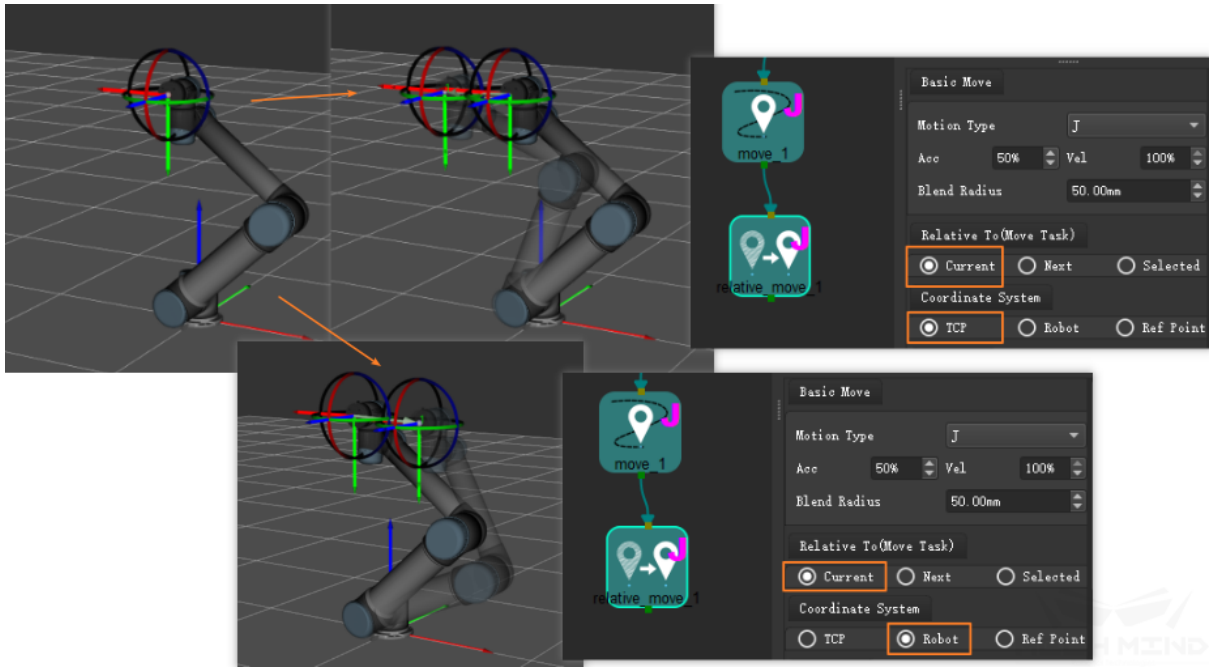
Parameters

Basic & Basic Move See *General Parameters of Move-Type Tasks*

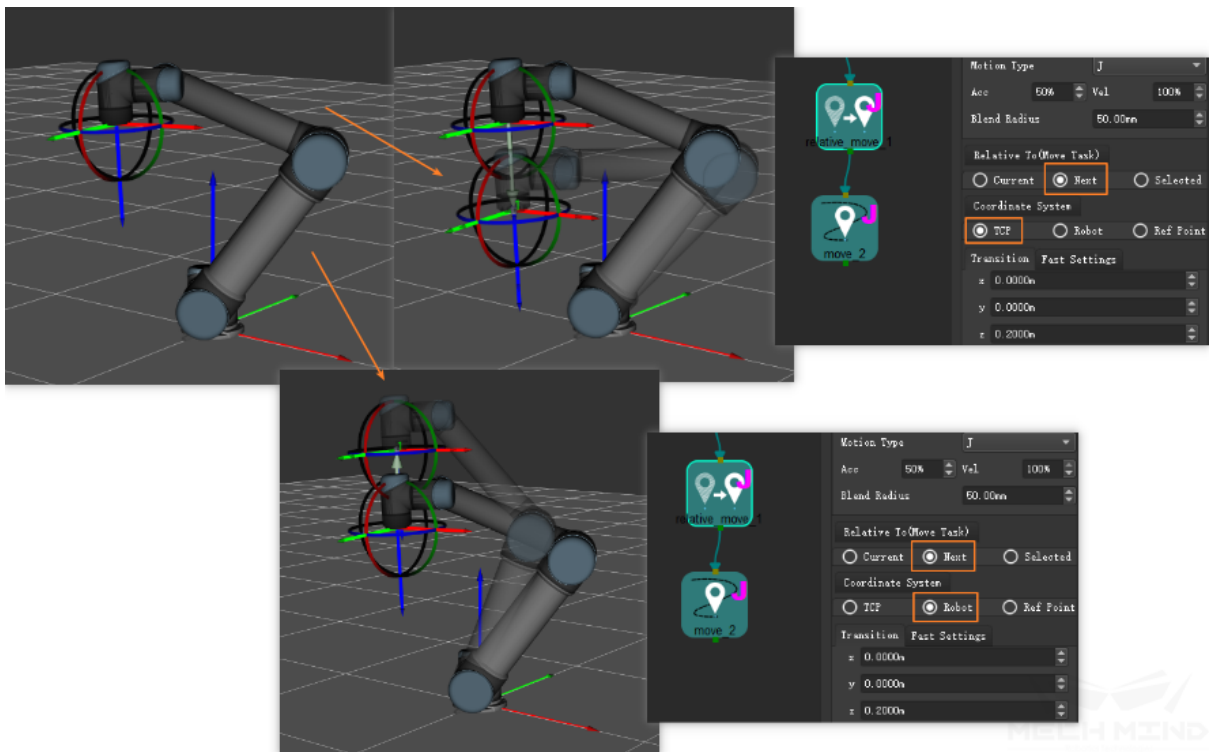
Basic Move See *Basic Move Parameters*

Relative to (move-type Tasks) The relative position setting includes the setting of the reference point, the relative coordinate system, and the offset vector. Wherein the reference point can be *Current*, or you can choose the *Next point* in the track. The coordinate system where the offset vector is located can be *TCP*, or *Robot*, and *Ref Point*. The offset vector is expressed by x, y, and z.

If you choose *TCP*, robot will relative move based on the TCP coordinate system; and if you choose *Robot*, robot will relative move based on the Base coordinate system. The comparison between moving based on two coordinate systems shown in the figure below:



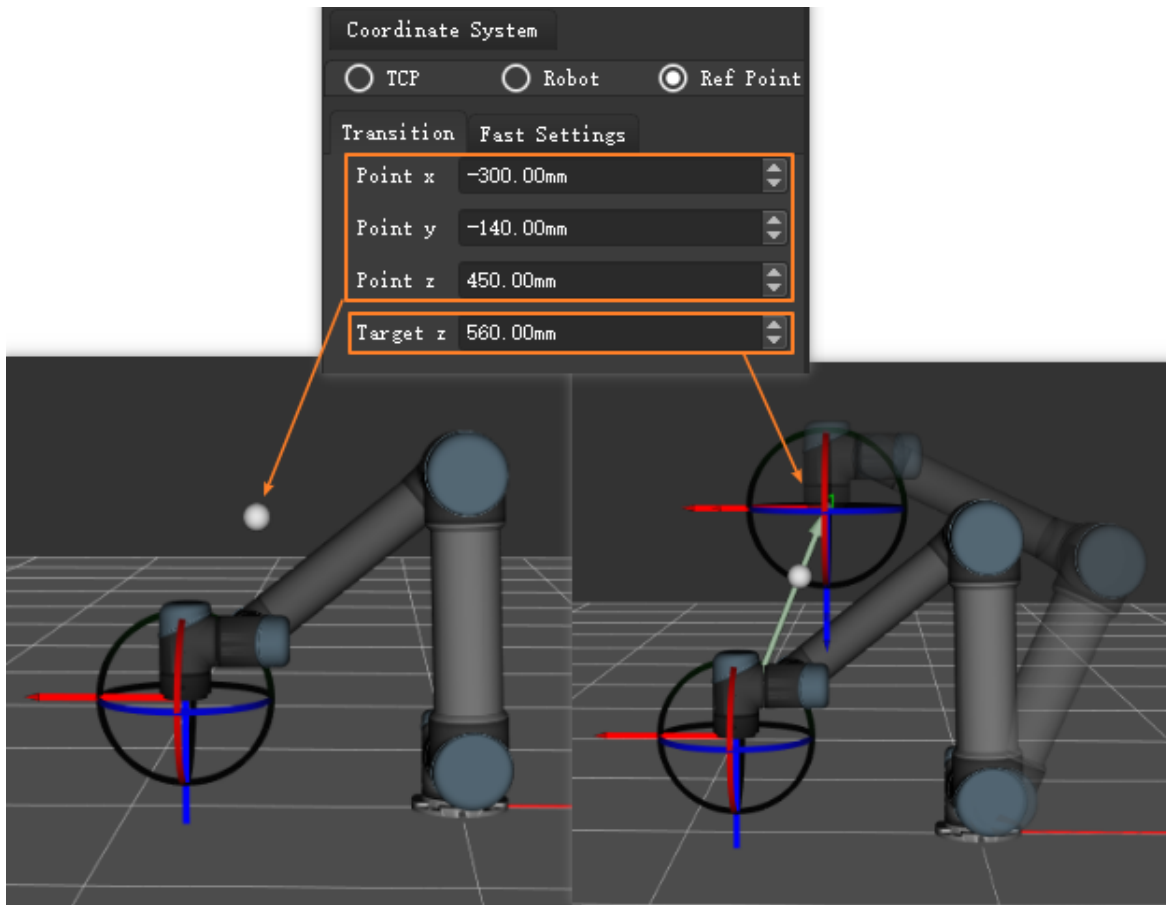
relative to current



relative to next (point)

Reference point description When the detected objects are in deep box, you can choose *Ref Point* to

avoid collision of end effector and box during vertical picking(default Z direction of relative_move), relative_move will use the *Ref Point* as the target direction, and the movement distance will be the preset value.



relative_move when Coordinate System is Ref Point

3.5 DI/DO

3.5.1 check_di

Description

Detect the signal value of the specified DI port of the robot. There are two output ports, if the value is 0, this task will output from left port, otherwise from right port.

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

port DI port of the robot to be checked

prePlanOutPort Pre-planned output port

Setting basis: the default value is -1. If the value is -1, the plan will be interrupted, that is, two relative tasks before and after this task cannot be planned together

If the plan shall not be interrupted, write 0 when the task output from left port in most cases, otherwise write 1.

3.5.2 check_di_list

Description

Check if the signal values of multiple DI ports of the robot are same as the setting values. If all values are as expected, this task will output from left port, otherwise from right port.

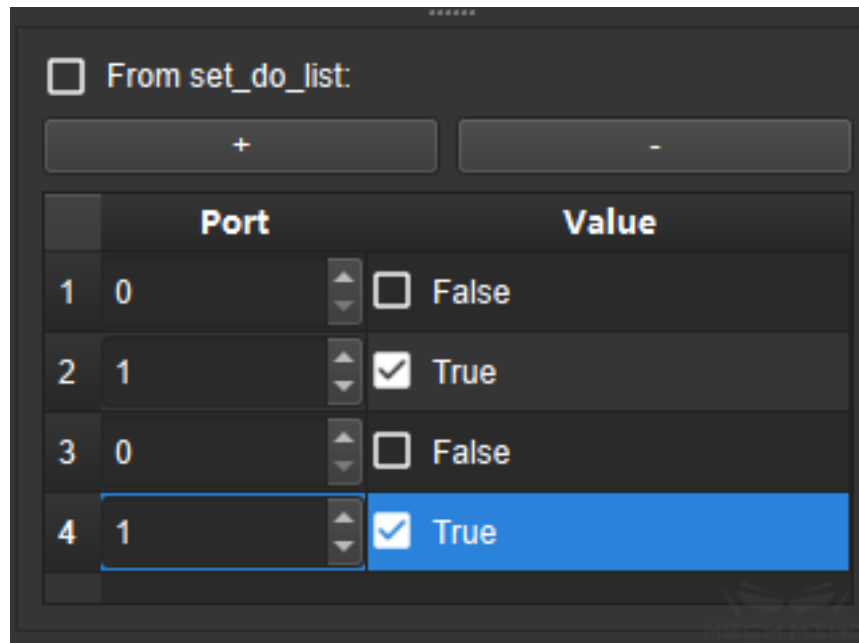
Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

ioMappingConfigPath Using with the *From Task* is checked

From Task

- If it is unchecked, add the port and corresponding value to be checked



- If it is checked, select *set_do_list* which need to be checked and the json file corresponding to *ioMappingConfigPath*

Application: Check if all the DO port are enabled as expected when using *set_do_list* control multiple suction cups.

json file format:

```
{
  "4": {"port":0,"value":true},
  "5": {"port":1,"value":false}
}
```

Note:

By checking the port value of the selected *set_do_list*, find the real corresponding port to be checked in the json file and check whether the value is correct.

As shown in the picture, 4, 5 are the ports set for the selected *set_do_list*, Port 4 is mapped to DI port 0, and port 5 is mapped to DI port 1. Then check if the value of port 0 is true and the value of port 1 is false.

If a port in the selected *set_do_list* is not correspondingly found in the json file, an exception occurs.

3.5.3 set_do

Description

Set the specified DO port signal of the robot

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

digital_out_value Set value of the robot digital output port, if it is checked as True, the value is 1, otherwise the value is 0.

port The digital output port number of Robot

delayTime Change the DO value after the setting time

3.5.4 set_do_list

Description

Set the multiple specified DO ports signal of the robot

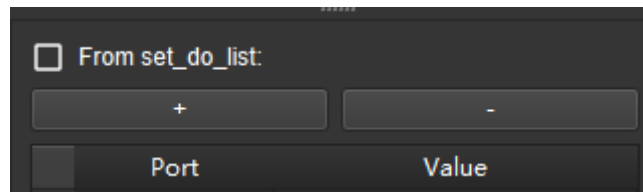
This Task is used to control multiple digital outputs at the same time. There are two application scenes:

- Constantly enable control several suckers at the same time, the port number can be defined by the user directly on the interface
- Use individually controllable combined suckers, which require a sucker offset when grasping, or when grasping multiple boxes at one time, the *visual_move* provides the port number that needs to be controlled for opening

Parameters

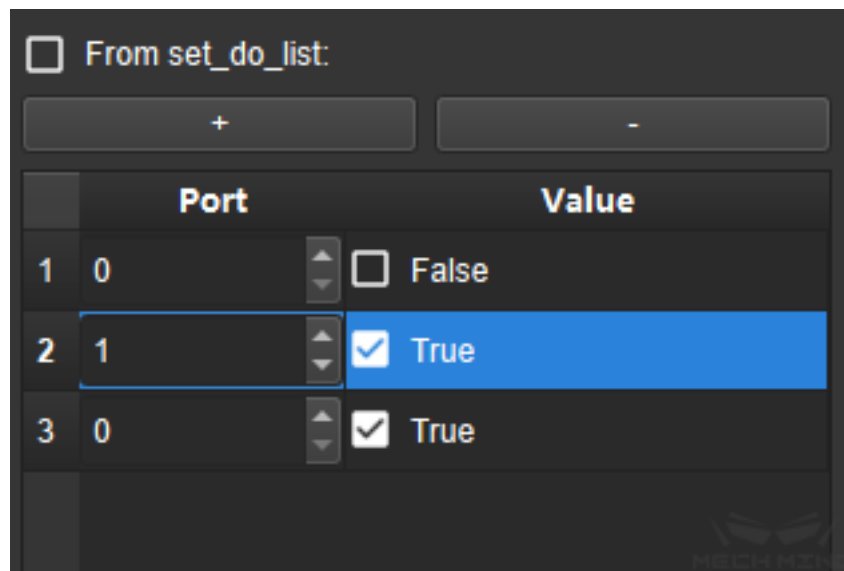
BasicNonMove See *General Parameters of Non-Move Tasks*

receiverName if the adapter is required to send the set DO status to an external device, enter the service name registered by the adapter on the Mech-Center to receive the IO status

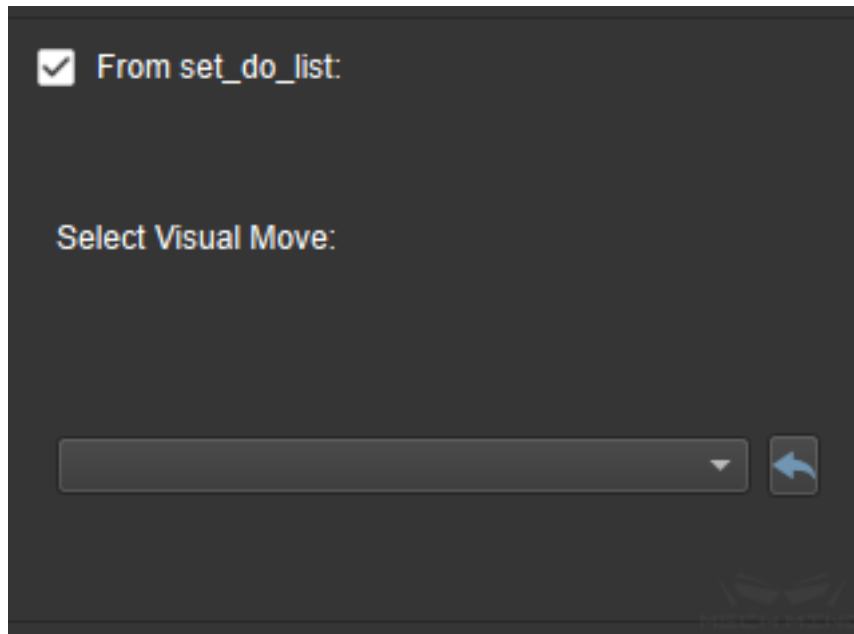


From Task:

- When the DO port to be opened at the same time is fixed, it is unnecessary to check From Task. Add or delete the number of DO by + or - , and then change the corresponding port number and value, as shown in the figure below



- When using *visual_move* and using the offset strategy or multipick strategy, the DO port to be opened each time is not fixed, which is provided by *visual_move*. At this time, check *From Task*, as shown in the figure below, and elect the corresponding *visual_move* name below.



3.5.5 wait_di

Description

Wait for the signal from the specified DI port of robot to reach the preset value

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

waitdi_timeout Setting of pending for DI timeout. The default value is -1. When it is -1, it will keep pending.

prePlanOutPort Pre-planned output port.

Setting basis: the default value is -1. If the value is -1, the plan will be interrupted, that is, two relative tasks before and after this task cannot be planed together

If the plan shall not be interrupted, write 0 when the task output from left port in most cases, otherwise write 1.

3.6 Logical Topology

3.6.1 branch_by_guidepost

3.6.2 branch_by_msg

Description

When an external device sends a branch command to the adapter, the adapter calls this Task after processing the command, and runs the corresponding branch

This Task performs two functions in the program:

- Run the corresponding branch process based on the data of the external device
- Interrupt the running program, and continue to run when the external signal is received

Parameters

objectName Since the adapter calls this Task by names, it is required to change the object name when using this Task

- When running different branches, the names can be written as: branch_1, branch_2
- If it is used for interruption, the names can be written as: interrupt_1, interrupt_2

BasicNonMove See *General Parameters of Non-Move Tasks*

size The number of output ports for this Task, for example, the size is 3, this Task has 3 output ports, which can be connected to three processes respectively; if this Task is used for interruption, set size to 1.

affectFutureMovement True by default, interrupting the planning of subsequent logic in the software (when this Task is used to execute different branches, since it does not know which branch will be run in the future, it will definitely affect program planning);

If it is only used to interrupt and wait for external signals, the subsequent program logic will not change. This parameter can be set to False.

runInAdvanceOnPlanned The default is False, the Task will not receive commands from external devices in advance; if True is checked, the commands sent by external devices can be temporarily stored. When the program is proceeded to this Task again, the temporary stored branch number will run directly to optimize the process.s

3.6.3 procedure & procedure_exit

procedure is used to process multi-layer tasks.

procedure_exit is applied in the situation that there are multiple possible results by the program, select an exit port to exit current procedure; if the multiple outputs are not involved in procedure, use procedure alone.

Example of procedure_exit is used

The screenshot displays a mission editor for 'root/mission_1'. The mission flow is as follows:

- Starts with a play button icon.
- Task: **visual_look_2** (eye icon in a square).
- Task: **move_3** (location pin icon in a square).
- Task: **relative_move_3** (two location pins with an arrow icon in a square).
- Task: **visual_move_1** (eye icon and location pin with a double-headed arrow icon in a square).
- Task: **relative_move_4** (two location pins with an arrow icon in a square).
- Task: **mission_exit_1** (square with a blue arrow pointing out of a square).

The **mission_exit_1** task is highlighted with a red box, and a red arrow points from it to the properties panel on the right.

Property	Value
Name	mission_exit_1
Basic Non-Move	
Skip E...	None ▼
Out P...	0
Exit Port	1

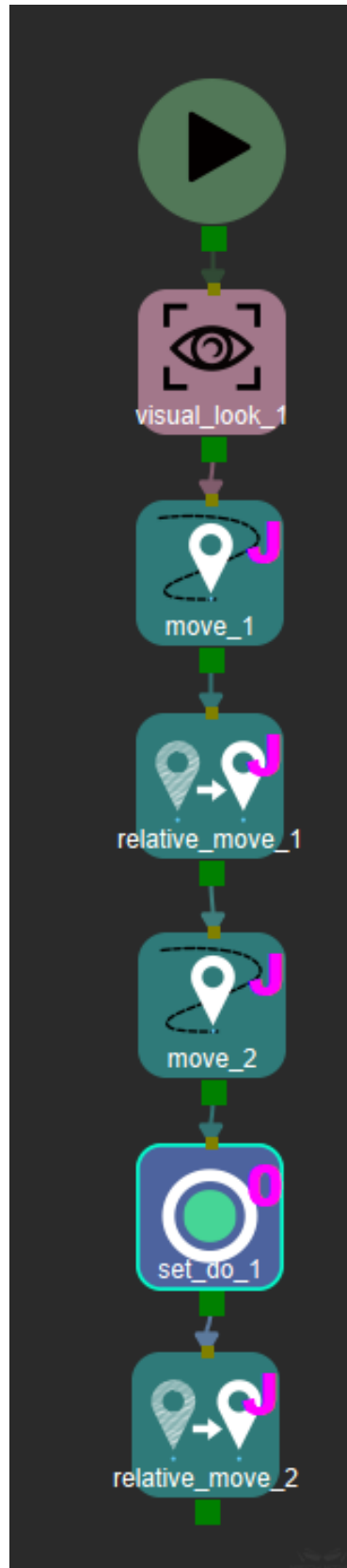
As shown in the figure, there is a visual_move in this procedure. Due to the possibilities of visual recognition: success and no pose or failed to pick, the two exits are required to connect different processing logics respectively. Perform the following operation:

- Connect procedure_exit Task at the port2 of visual_move;
- Set the exit_port in the property of procedure_exit to 1. (exit_port defaults to 0)

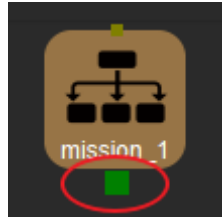
Back to the previous layer, you may see that the procedure becomes two output ports, and users can connect them with different processing logic as required.



Example for using procedure alone



In the figure above, the process flow in procedure does not have multiple possible results. This procedure has only one output port. As shown in the figure below, when the program finishes running the last Task, it will jump out of procedure directly.

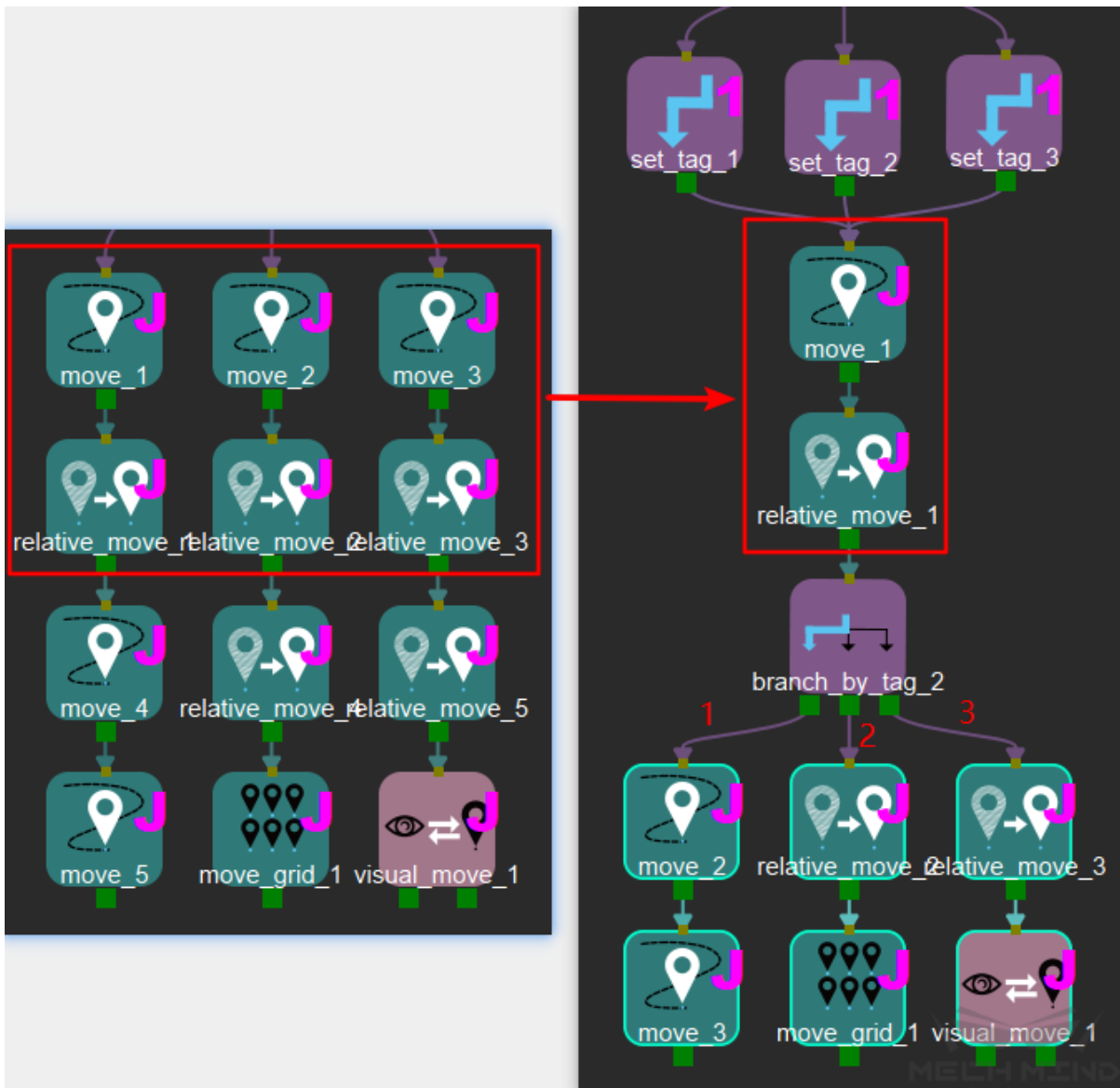


3.6.4 procedure_exit

3.6.5 set_guidepost & branch_by_guidepost

The scene for this combination is: when there is a common part in multiple logical branches, the common part is not required to be copied by multiple times, use set_guidepost to mark the current branch, merge it into the common part, and then use branch_by_guidepost Task to restore the program running logic to the original branch.

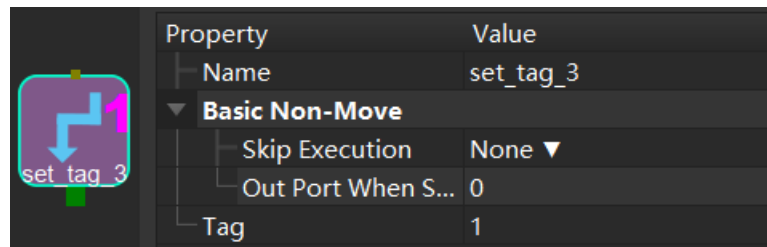
The following figure is an example:



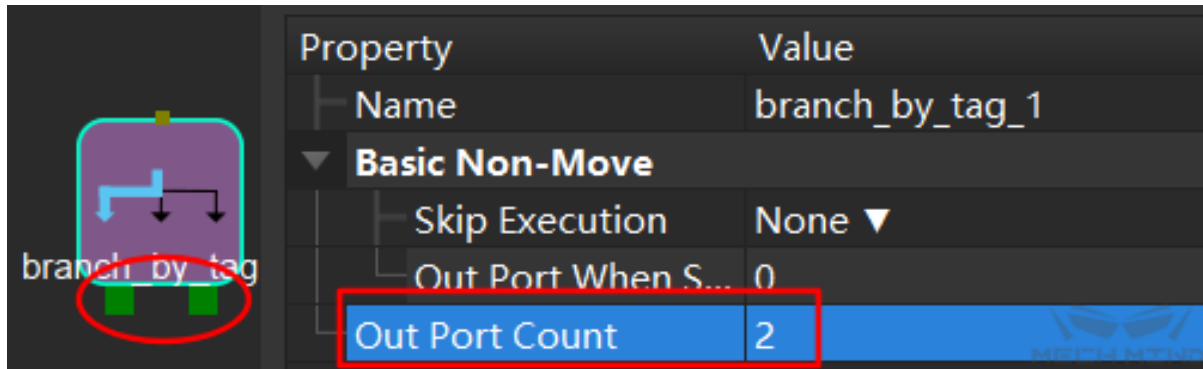
The three branches have common Tasks: move and relative_move. No matter which branch, they will proceed to the same position when completing the previous process. At this time, the same Task can be selected, and set_guidepost Tasks are added to the branches to mark the current branch number. When the program finishes move and relative_move Tasks, branch_by_guidepost will restore the program running logic to the previously running branch according to the current identification.

Parameter settings

1. set_guidepost



- Basic parameters for non-movement, see *General Parameters of Non-Move Tasks*
 - routePort: the ID of the current branch, used to restore the branch of the branch_by_guidepost Task
2. branch_by_guidepost



- Basic parameters for non-movement, see *General Parameters of Non-Move Tasks*
- size: output ports of the branch_by_guidepost Task, based on the total number of routePort

3.7 Pallet

Palletizing related Tasks are divided into five according to different application requirements. Users can choose one of them to connect according to the requirements.

3.7.1 custom_pallet_pattern

Description

Using custom_pallet_pattern Task, you can define any pallet pattern and is available at any time, but correspondingly, the use of it is relatively complicated.

Parameters

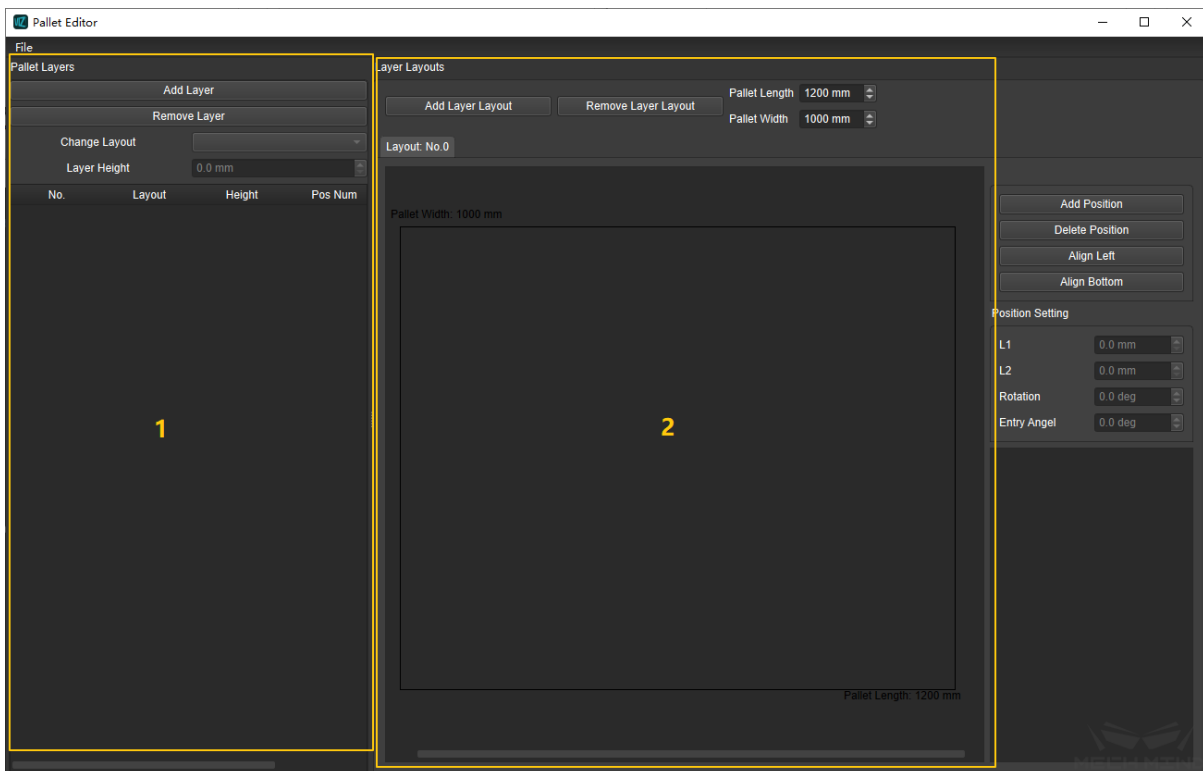
palletEdit

In addition to *Parameters*, *custom_pallet_pattern* has the functions of pallet edit and pattern loading



Click *Pallet Editor* to pop up the interface of the pallet pattern editor. When the pallet pattern editing is finished, the pallet pattern will be saved in the project folder as a json file. Click *Load Pattern* to select the completed json file to load the edited pallet pattern.

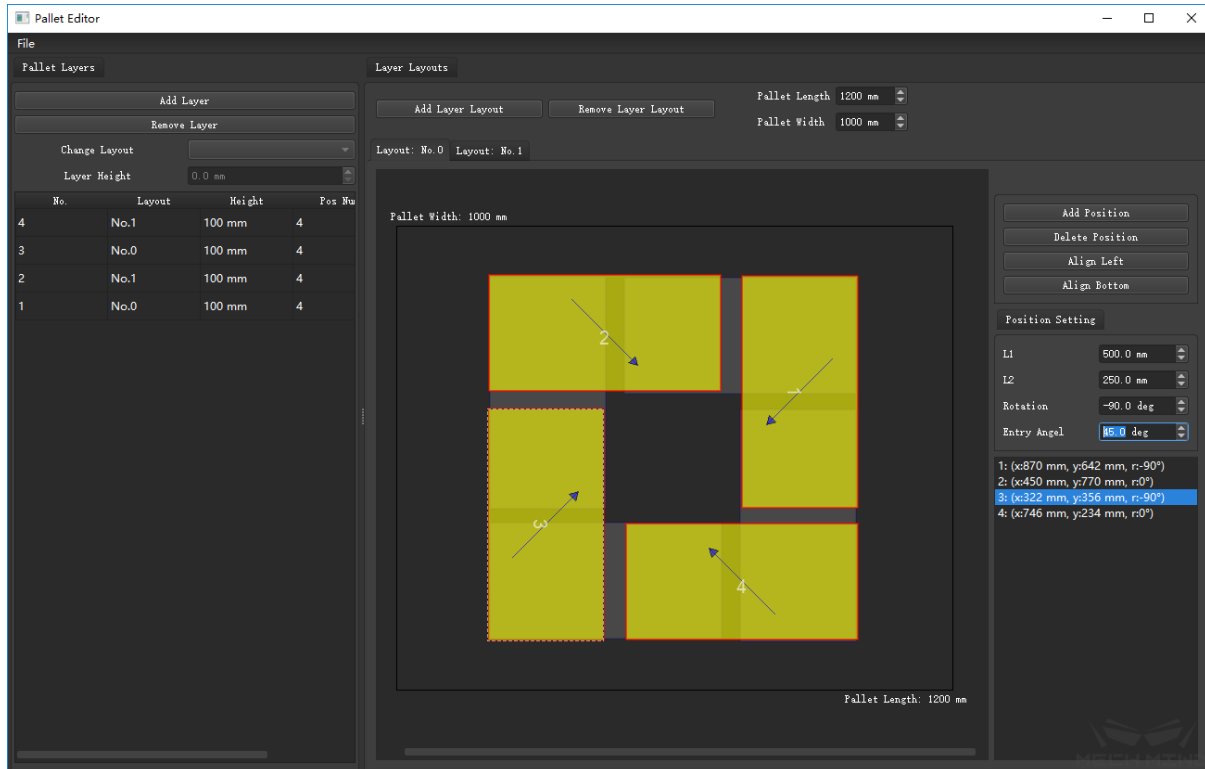
The interface of Pallet Editor is shown below. The left and right sides are the overall editing area and layout editing area respectively.



The number of stacking layers, the layout and height of each layer can be set in the overall editing area;

The number of boxes, the position adjustment of boxes, the specific layout editing, and the parameters such as box length and width, rotation, and plane cut-in angle editing can be set in the layout editing area.

An example of an edited pallet pattern:



As shown on the left side of the figure above, the whole stack has four layers, but there are only two layouts, wherein the layout layer No.0 is used for the odd-numbered layers and the layout No.1 is used for the even-numbered layers.

The right side of the figure is a top view of the pallet pattern, wherein each yellow/gray block represents a box, a yellow box represents a box in the current layout, and a gray box represents a box in another layout.

The operation is as follows:

- Pallet Layers Tab:
 - Click *Add Layer* to increase the number of layers to 4, click each layer one by one, and select a layout for each layer in the **Change Layout** menu.
- Layer Layouts Tab:
 - Click *Add Layer Layout* to add different layout modes.
 - In a certain layout mode, click *Add Position* to add a box. The box layout supports some common operations such as *Align Left* and *Align Bottom*.
 - When a single box is selected, you may change the length (L1), width (L2), rotation angle(Rotation), and cut-in angle(Entry Angle) of the box (that is, the box's entry angle on the plane) in Position Setting Tab.
 - After editing, you may click *File* → *Save to File* in the upper left corner and save it to an external json file for further use.

3.7.2 mixed_pallet_pattern

Description

mixed_pallet_pattern is used to stack objects with different sizes. The pallet pattern can be automatically generated according to the parameters set by user and the size of the target box.

Parameters

In addition to [Parameters](#), the following parameters can be set for mixed_pallet_pattern Task:

workmode

- Online: Plan the stacking for each new box without knowing the size.
- Offline: Plan all the boxes when the size for all boxes to be stacked are known. This function is used to adjust the stacking parameters during the debugging phase. You may read the json file to view the planned stacking. It does not support connecting and running physical robots.
- fileName: Read box json file name in Offline mode.

captureTwiceToUpdate Default False, it can be choose when you can't get the full size at the first photo, and you need to take a second photo because of the lack of the height information of the box.

When [visual_move](#) has grasped the box but has no box height information, mixed_pallet_pattern will estimate the box height and plan according to the estimated height when calculating the placement position and weight;

When running this Task to take a second photo to get the height of the box, the software makes a second planning. At this time, mixed_pallet_pattern will uses the box size given by two visual recognition to calculate.

Pallet Type Pallet common parameters:

PalletDimension

- palletSizeX: Set pallet length
- palletSizeY: Set pallet weight
- palletMaxZ: Set max pallet height

FixedSequence

- isFixedSequence: Default False, it can be choose when the order of boxes is fixed.
- futureGainFactor: This parameter takes effect when :guiabel:` isFixedSequence` is checked. If this parameter is increased, more consideration will be given to the subsequent box stackability when palletizing, and more space will be reserved.

gapWidth The width of the gap between boxes, the unit is meter (m). To ensure that the given box size is smaller than the actual size to prevent collisions. Recommended setting: 0.01 – 0.02m

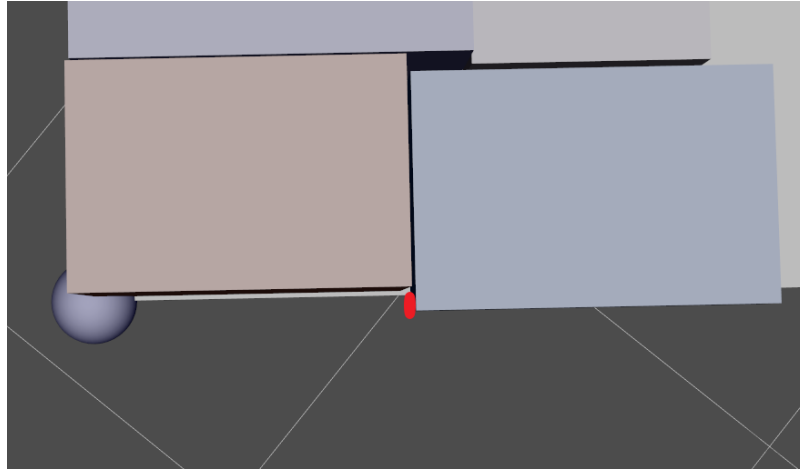
TaggedBox

- Tag on Boundary: Default False.it can be choose when the boxes have the tag and the tag is on boundary.

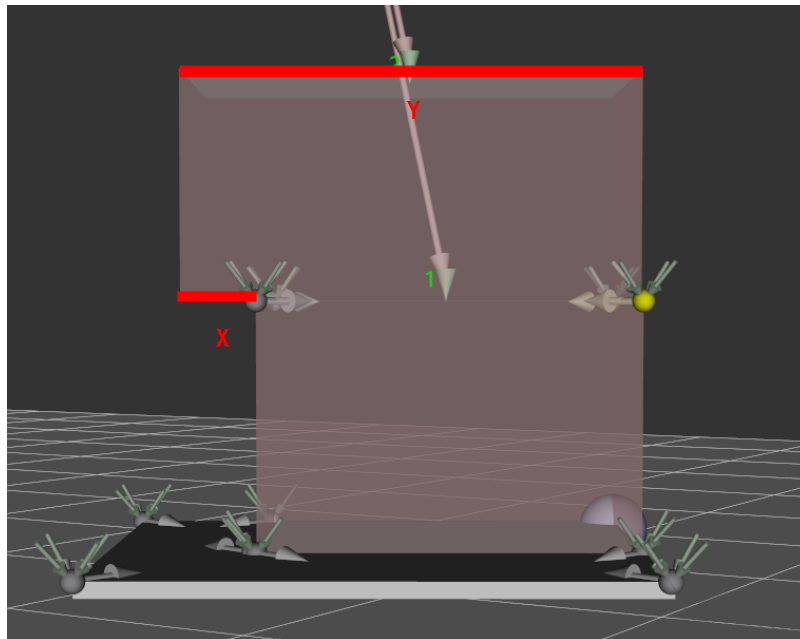
- tagToBoundaryDist: The farthest distance from the box tag face to the edge of the pallet when stacking.

Structured

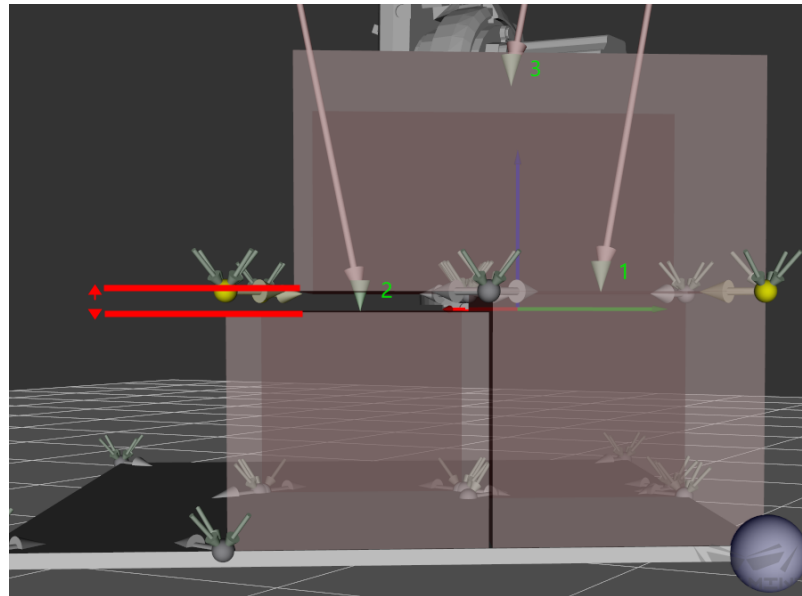
- Setting
 - allowedExcess: Allow the box to exceed the width of the edges. The recommended setting is 0.02 – 0.05m, as shown below:



- boxExceedRatio: The upper box is allowed to exceed the maximum proportion of the plane on which it is pressed. As shown in the figure, the maximum Y/X does not exceed this parameter.

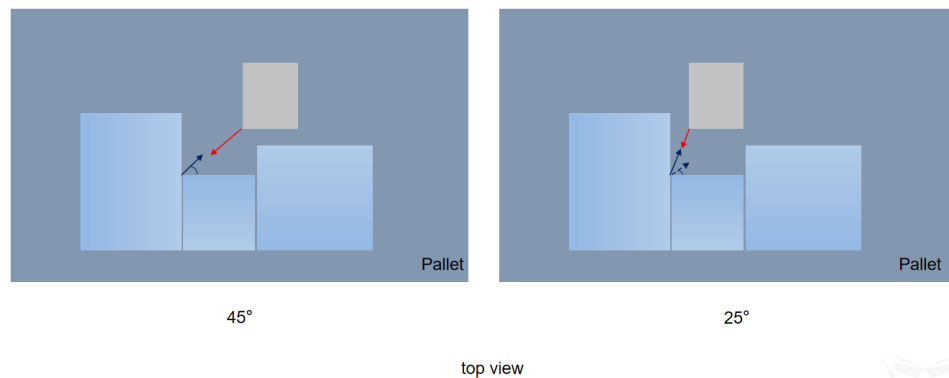


- layerHeightDiff: When the size of the box to be stacked on the upper layer is larger than the box on the lower layer, the large box on the upper layer is allowed to be stacked on a plane whose height difference does not exceed this parameter.

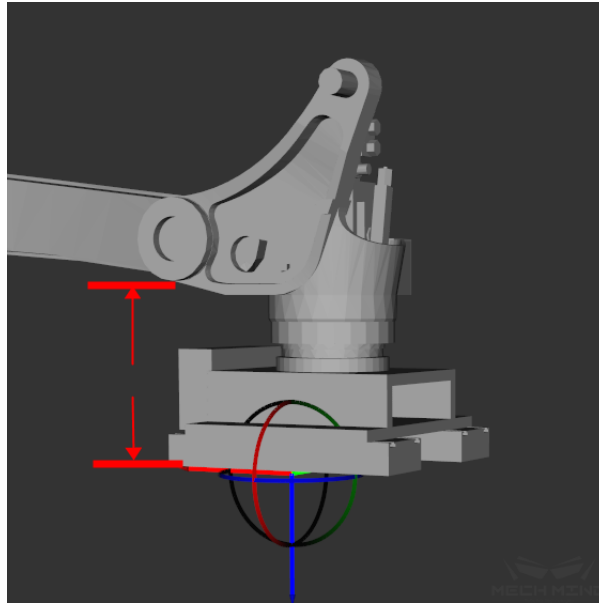


- `samplingRate`: Plan the sampling rate (sample/meter) for the box position. The higher the sampling rate, the more accurate the result, but the slower the speed. Recommended values are 200, 500, and 1000 sample/m.
- `cornerFreeAngle`: In the projection direction of the pallet, the angle between the entry path of the box and the side of the adjacent box. As shown in the figure, if this parameter is set too large, a U-shaped empty area may be left during stacking; if it is set too small, the actual stacking boxes may collide with adjacent boxes. It is recommended to set 15 – 30 degrees.

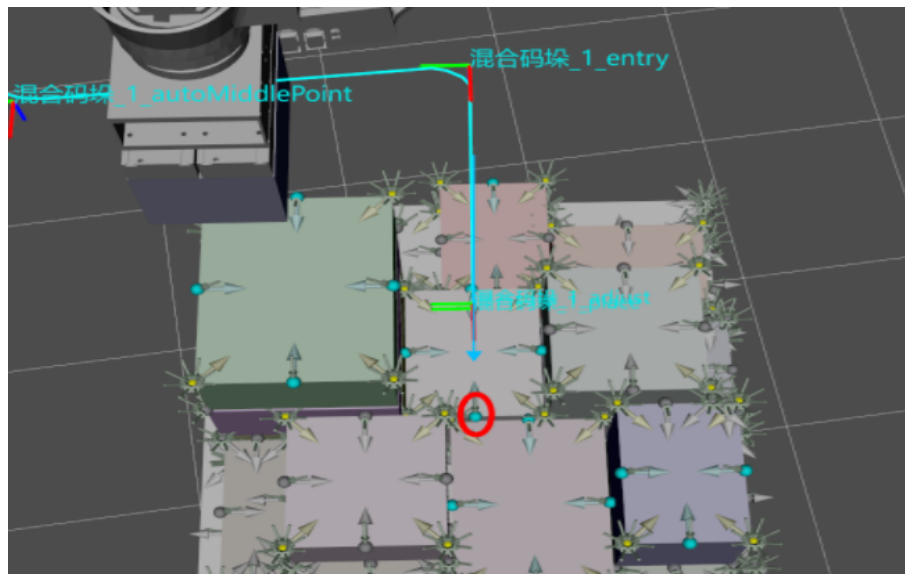
- objects which are placed
- object need to be placed



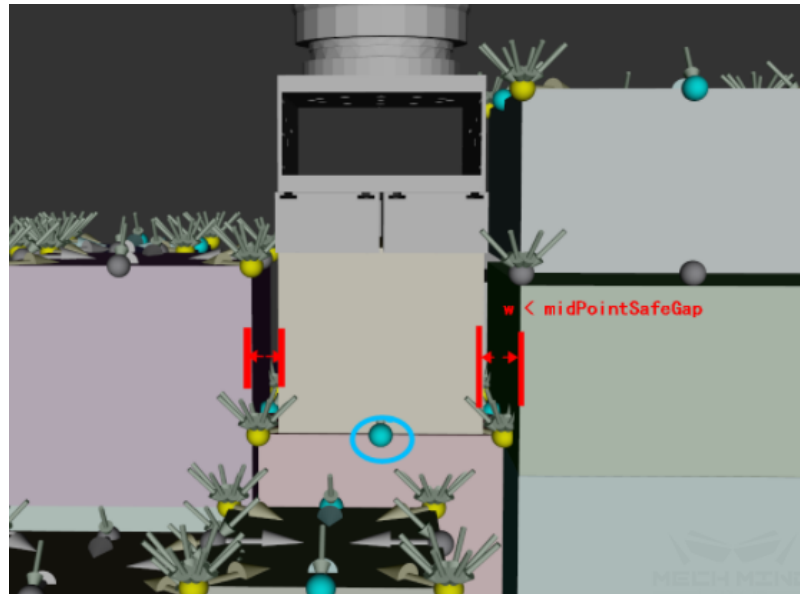
- `endEffectorDimZ`: The vertical distance from the bottom surface of the robot's end to the bottom surface of the robot's penultimate joint, as shown in the figure below, to prevent some placement poses from causing the end to collide with the robot body.



- enableMidPoint: The default is True. In addition to the corner points will be as candidate positions, the midpoint between the corner points will also be used as candidate positions. When there are U-shaped grooves in the stack, the boxes may be inserted into the grooves in a trajectory parallel to the adjacent boxes. The middle point is displayed in cyan. As shown in the figure



- midPointSafeGap: This parameter will be effective when setting Use Midpoint to True. When the box is inserted at the midpoint candidate position, the gap left on both sides will be larger than this value. As shown in the figure below.



If the robot motion error and the box size error allow, this parameter can be set to a value smaller than *gapWidth*, which will significantly improve the pallet pattern when workMode is online.

- Weight: The parameters in this group are weights, which decides the box's place position in the pallet.
 - adjacentAreaWeight: The higher the value of this parameter, the more likely that the candidate position where the side touches the box in the stack will be executed, and vice versa.
 - supportAreaWeight: The higher the value of this parameter, the larger the supported area (area which is less exceed the lower surface area) is more likely to be executed, and vice versa.
 - baseHeightWeight: The higher the value of this parameter, the lower the target position is more likely to be executed, and vice versa.
 - projectedDistToCornerWeight: The higher the value of this parameter, the shorter the distance from the position to the priority corner is projected on the diagonal of the pallet, which is easier to be performed.
 - supportBoxNumWeight: The higher the value of this parameter, the position above where more boxes have been stacked is more likely to be executed, and vice versa. The higher the value makes it easier to plan a relatively more stable stack with reducing the compactness of the stack.

It is recommended to use multiples as the adjustment interval during initial adjustment.

Unstructured

- Weight: The parameters in this group are weights, which decides the box's place position in the pallet.
 - heightWeight: The higher this parameter, the more consideration should be given to the height of the stack type. The height of the stack type should not be too high.
 - capacityWeight: The lower this parameter, the more compact the stack.

- roughnessWeight: The higher this parameter, the smoother the top layer of the stack.
- areaWeight: When stacking from boxes, the lower this parameter, the boxes with larger projected area will be selected first.
- distanceToBoundaryWeight: The higher this parameter, the closer the box to the pallet boundary.
- baseHeightWeight: In multi-layer stacking, the lower this parameter, the easier it is to stack the boxes on the lower plane

3.7.3 multi_pick_palletizing

Contents will be added soon!

3.7.4 predefined_pallet_pattern

Contents will be added soon!

3.7.5 smart_pallet_pattern

Description

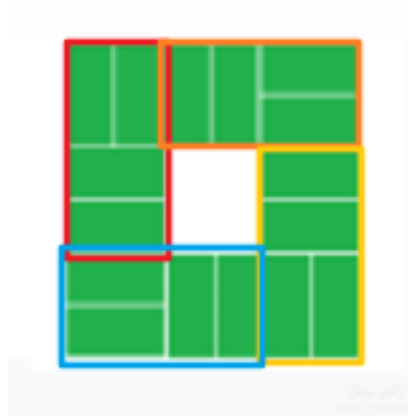
In addition to *Parameters* , smart_pallet_pattern Task has settings for palletizing type and object size

Parameters

Common Settings

- reverseOddEven: The default is False. When True is selected, the swap the parity layer
- palletPattern:
 - Windmill: as shown in the figure





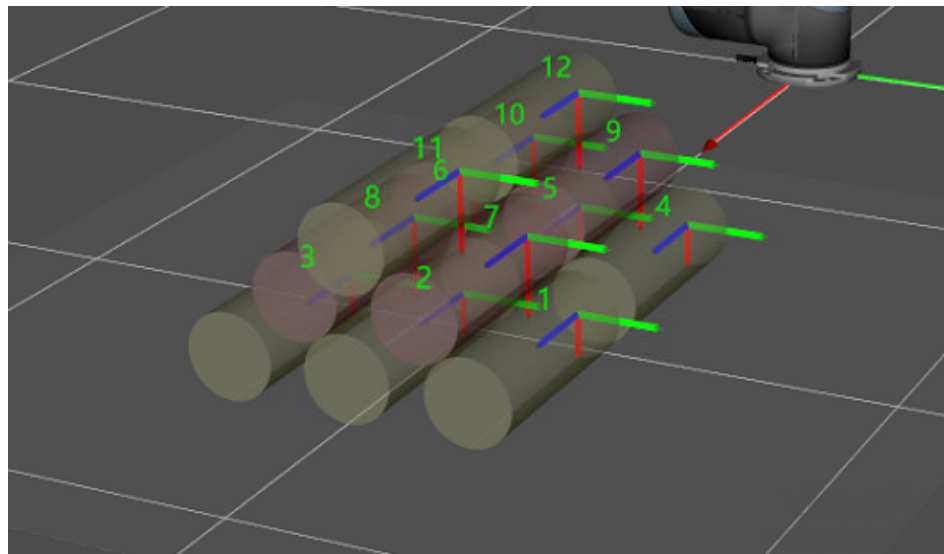
Different color boxes in the figure represent a unit. The user can configure total rows and columns in each unit

- Grid: as shown in the figure



The user can configure total rows and columns in each unit

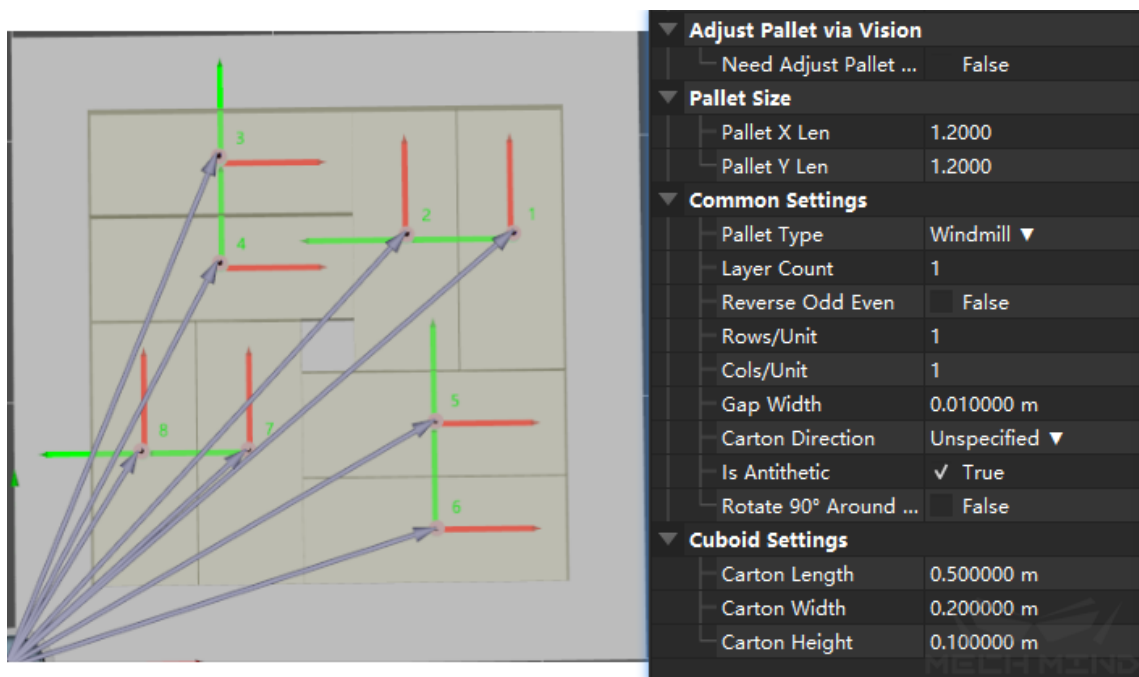
- Cylinder: it is used for cylindrical palletizing and can be configured in *CylinderInfos*



- rowsPerUnit: Set the row number of the box in a unit
- colsPerUnit: Set the column number of the box in a unit
- layerNum: Set the number of layer for palletizing
- palletLength: Set the length of the pallet, the unit is m
- palletWidth: Set the width of the pallet, the unit is m

- gapWidth: Set the spacing between each box, the unit is m
- cartonDirection: It is only applicable to windmill pallet pattern. If the object is a sack with an opening, it may specify the opening orientation; you may choose Unspecified, Inward or Outward.
- isAntithetic: It only takes effect when the pallet pattern is square; if True is selected, the parity layers use different placement layouts to enhance the stability of the stack; if False is selected, parity layers are placed in the same layout. If the pallet pattern is not square, even if it is hooked, it is impossible to use different placement of the parity layer.
- adjustToSquare: The default is False; if True is selected, even if the pallet pattern is not square, you may adjust the pallet pattern to square by adjusting the gap of the boxes. It is usually used in conjunction with *isAntithetic*.

CartonInfos This set of parameters is only used for visual display when setting the stacking parameters, as shown in the figure.



When running the program, the relevant information of the box is inputted by external service, and the manual setting is invalid at this time.

CylinderInfos This set of parameters is used to set the shape parameters of the cylindrical object, the number of stacking rows and columns, and the spacing between the rows/columns, etc.

3.7.6 visual_pallet_pattern

Description

Based on the function of *smart_pallet_pattern* , visual_pallet_pattern recognizes pose of the uppermost object based on visual recognition, matches it with the known pallet pattern, locates the position of the stack, and obtains the position and entry path to be placed this time.

Parameters

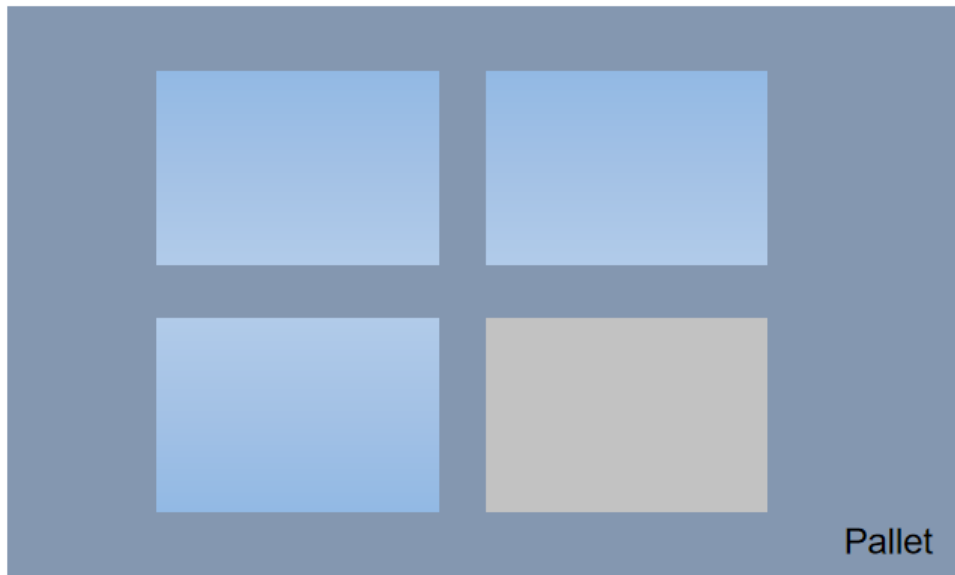
Since the function of visual_pallet_pattern is based on smart_pallet_pattern, the setting for pallet pattern parameters see *smart_pallet_pattern*

1. **xyTolerance** Tolerance of x and y directions for visual position and stacked position, unit is m.
2. **zTolerance** Tolerance of z directions for visual position and stacked position, unit is m.
3. **rotZTolerance** Rotation tolerance value of visual position and stacked position around Z axis, unit is °;

Note: If the above tolerance values are all within the preset range, the pose given by the vision can be seemed same as the stacked pose

4. **visionName** It is used to identify the vision project name for the stack. The default is vision3d, which can be manually modified.
5. **needCapture** True is selected by default. When the program proceed to this Task, it will take a picture to identify the pallet pattern, and it will be automatically set to False after recognition; the principle is as follows:

- objects which are placed
- the point can place next time



top view



When the camera takes a picture of the stacked object (half stack), set this item to False, at this time, identify the pose of uppermost and match the pallet pattern, calculate the pose to be stacked based on the known pose, and set this item back to True.

Therefore, if a new half stack is changed during the process, it is required to be manually set to True for the picture recognition.

3.7.7 Parameters

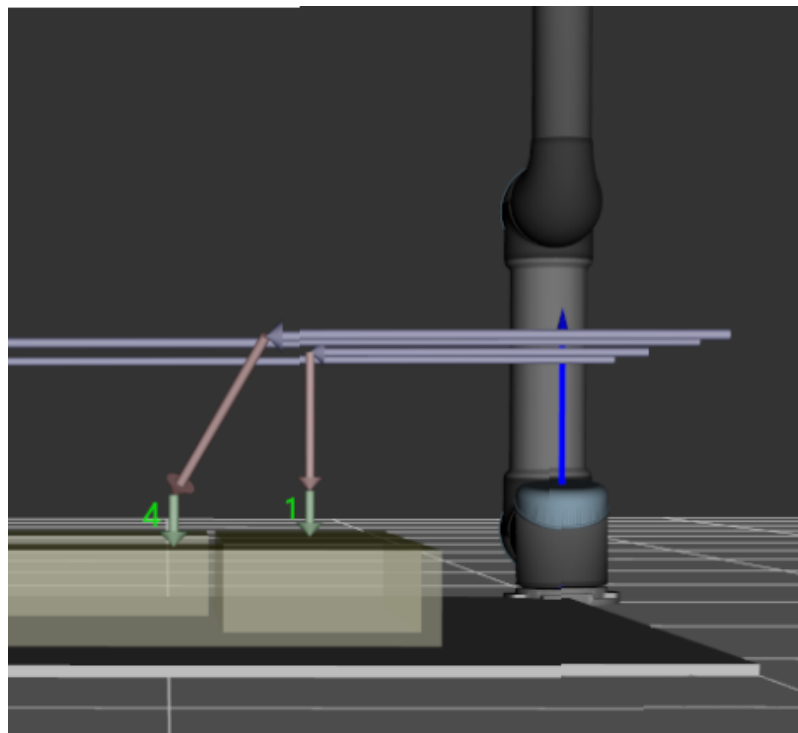
Index The meaning of this parameter is to index the box to be placed, counting starts from 0. The application method is: when starting the code from an empty stack, this value starts from 1. If four boxes have already been stacked, the stacking process is suspended for some reasons at this time. When the stacking is started again, it is not necessary to manually remove the boxes that have been stacked on the pallet and start again from the beginning. Instead, you only need to set the start index to 4. The program can automatically continue to execute the palletizing process from the fifth box.

BasicPalletSetting

- hideTrajectory
- movesCount

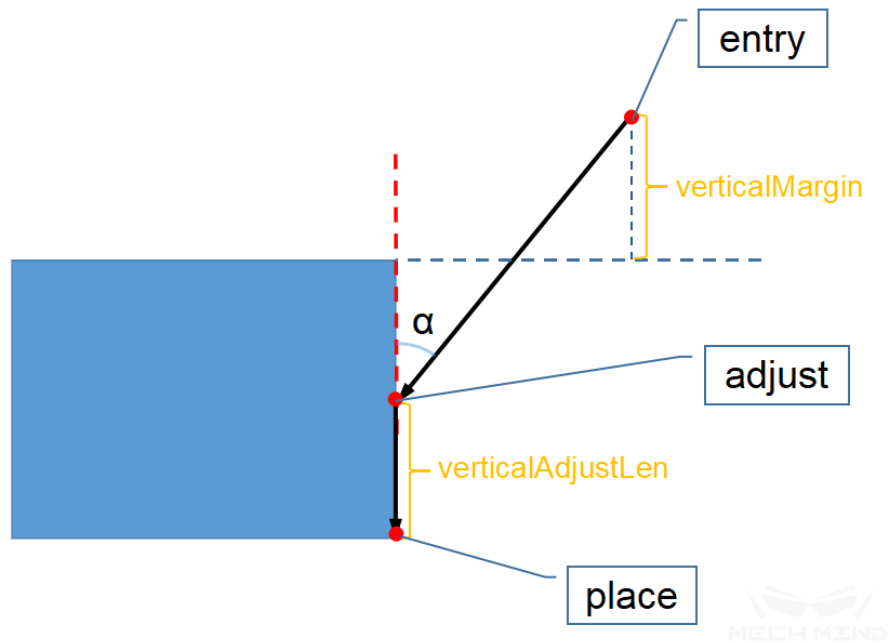
MotionControl

- autoMiddlePointForceMoveJ
- entryForceMoveJ
- adjustForceMoveJ
- placeForceMoveJ
- accVelScaleRatio: As shown in the figure below, when approaching the stack, it is divided into three stage. The first stage approaching the stack is in purple, and the other two stage are the paths for actually placing the boxes. According to the field application, sometimes the robot is required to move faster when approaching the stack, and slower when actually placing the boxes, in such a case, the speed/acceleration of the two stages of approaching the stack (purple) and actually placing the boxes can be restricted by this parameter. The speed and acceleration approaching the stack (purple) are the values specified in *Basic Move Parameters*, referred to as v1, and the speed/acceleration of the next two stages of robot motion are $v1 \times$ this parameter.



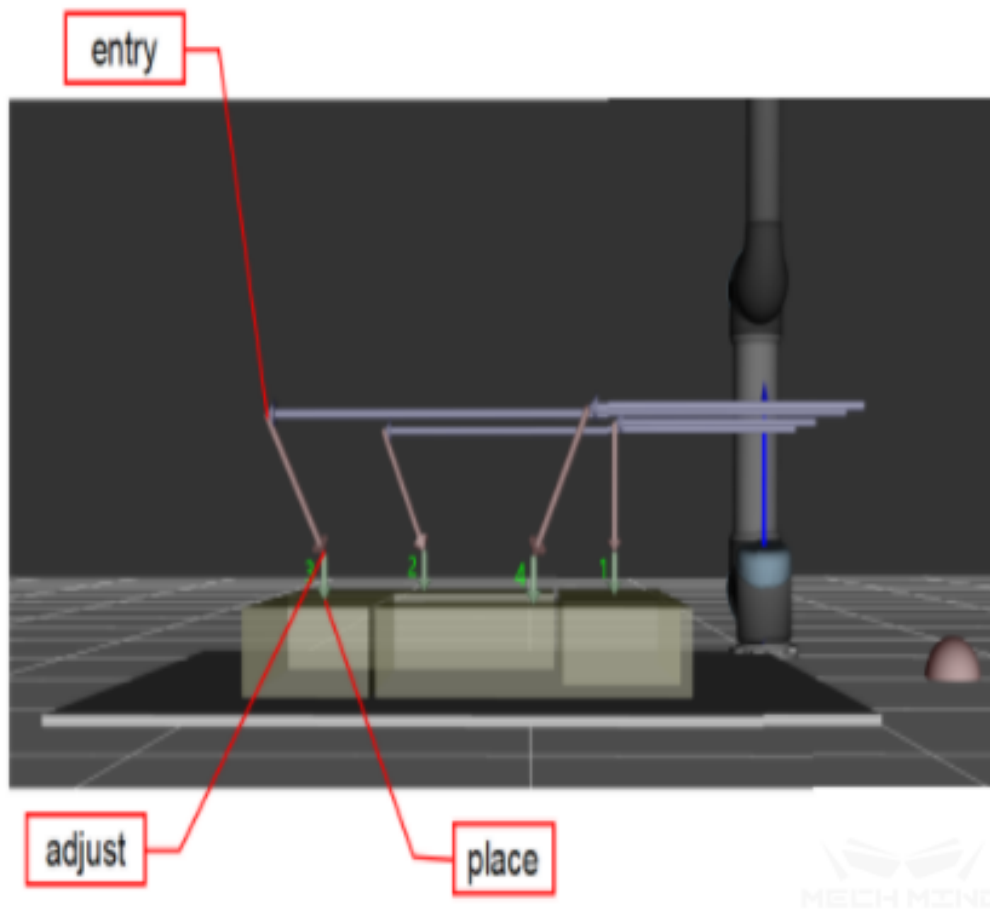
EntryAdjust There are three parameters in this set, which control the entry path of boxes together when they are stacked. The purpose is to avoid the collision of the stacked boxes due to accuracy or other reasons when the robot directly lowers the boxes vertically. Adjust the entry path so that the box approaches the stacked box at a certain angle and then lowers it vertically;

For each box, there are 4 positions in the stack. The three parameters in this set control the three parameters, as shown by the red dots in the figure below, which are entry, adjust and place. The perspective below is the front view of the box.

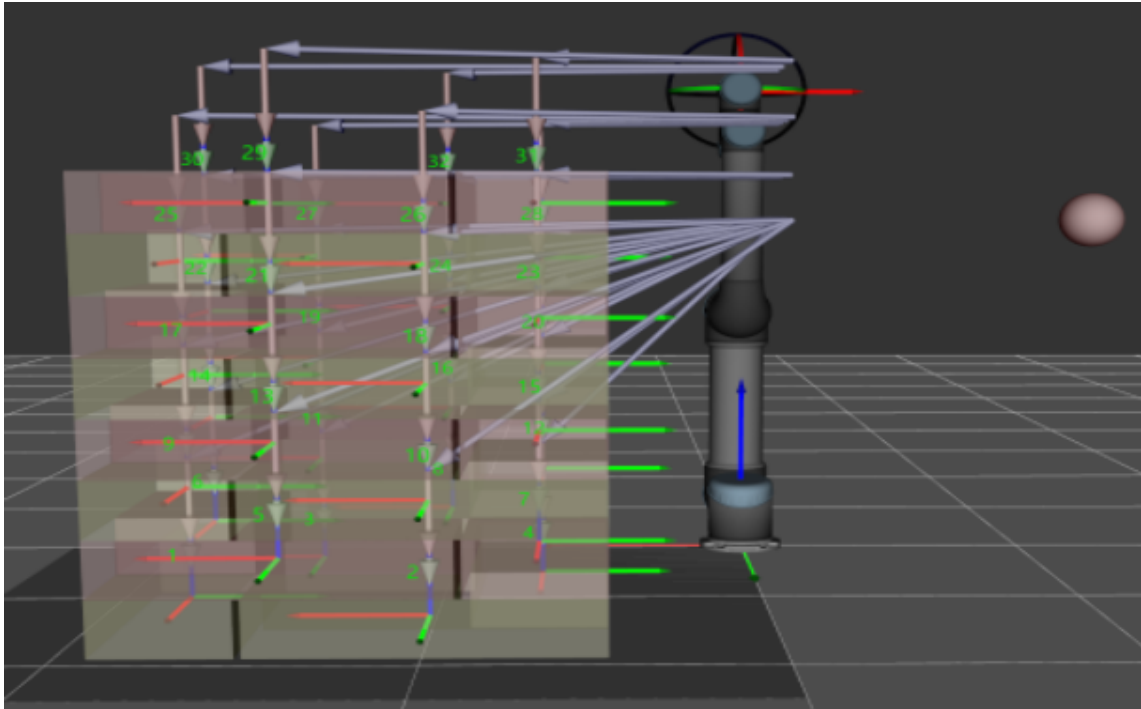


- **verticalAdjustLenRatio**: This parameter is the height ratio when approaching the stacked box, which is equal to $(\text{verticalAdjustLen} / \text{box height})$. The value range is 0 – 1, with no unit. The default value is 0.5.
- **verticalMargin**: This parameter is the height margin of the cut-in point. The value ranges from 0 to infinity, and the unit is millimeter. Generally, it is set to a number greater than 0 to leave a margin. The specific value depends on the application scene.
- **entryAngleZ**: This parameter is α in the above figure. It is the angle between the path entry-adjust and the vertical direction. The unit is $^{\circ}$, and the value range is $-80 - 80^{\circ}$. The recommended default value is $30^{\circ} - 45^{\circ}$.

autoMiddlePoint: The pink ball as shown in the figure below, this set of parameters controls the fourth position in the stack is mentioned above-the position of the middle point.



- X/Y: Set the position x, y of the pink ball in the base coordinate system of the robot. The software will automatically calculate reasonable midpoint coordinates for stacks of different heights based on this position.
- minZ: The minimum absolute z-height when approaching (purple path). (And the maximum height of this floor). As shown below



- isMiddlePointTrajVertical
- extendedDist

Attention: The autoMiddlePoint is only an indication of the direction when approaching the stack, not the point that the robot actually arrives. Therefore, the pink ball shall be as far away from the pallet as possible. If the ball is too close to the stack, during the process of placing the box, it may cause crushing collision.

AdjustPalletViaVision This parameter can dynamically adjust the position of the stack by the vision service

- needAdjustPalletPose: The default is False. When the position of the stack is required to be adjusted dynamically, select it as true. When the program proceeds to this task, it will call the vision service to identify the location of the stack.
- visionToAdjustPallet: Fill in the name of the vision project to identify the location of the stack. When the program proceeds to this task, it will call the vision service according to the vision project name.

BasicMove

Here you can set the position of pallet and move parameters, etc. The function is the same as other *Basic Move Parameters* , so it won't be described here.

3.8 Robot Tools

3.8.1 call_robot_function

3.8.2 check_tcp

Description

This Task is applied to the scenario that robot switches end effectors to picking. It can check if the current end effector is the selected end effector, if same, it will output from left port, else it will output from right port.

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks* .

Set EndEffector Select the expected end effector to check if the current end effector is right.

3.8.3 get_jps

Description

This Task is used to obtain the joint angle information of the robot currently registered in Mech-Center and synchronize the joint angle information with Mech-Viz.

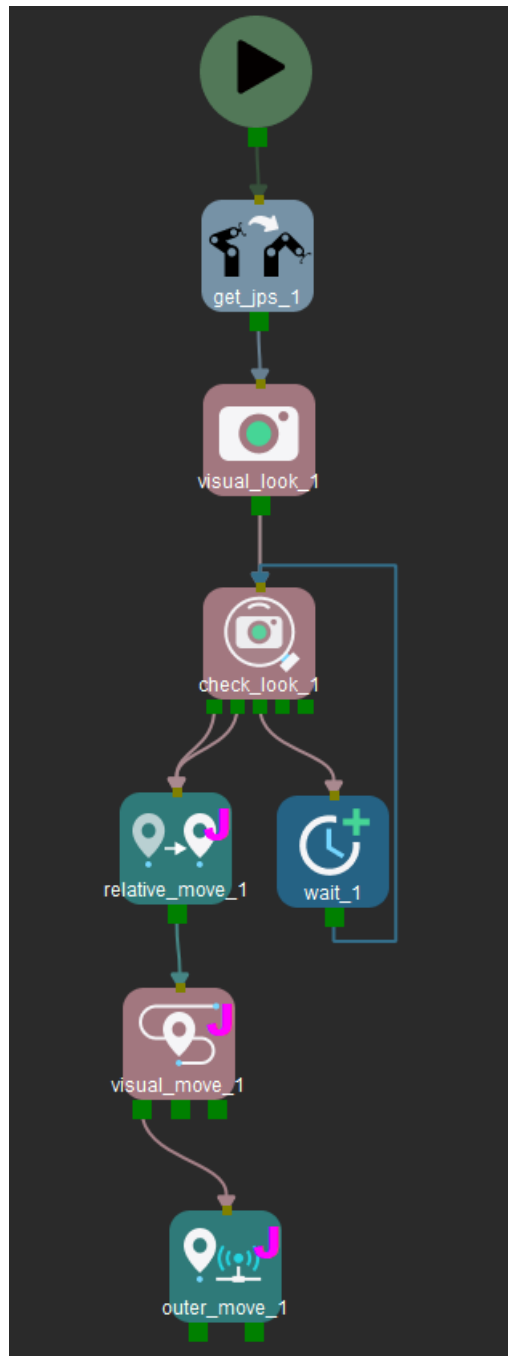
Usage Scenario

When Mech-Viz does not fully control all movements of the robot, you can use this Task to obtain the joint angle information when the robot is not guided by Mech-Viz to maintain the control.

Parameters

Basic Non-Move Please refer to *General Parameters of Non-Move Tasks* for detailed information.

Application Example



Before **Visual_move** triggers the corresponding Mech-Vision project, Mech-Viz can obtain the joint angle information of the real robot through **get_jps**, and use it as the starting position in the planned path.

3.8.4 payload

Description

This Task is used for setting the payload of robot

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

payloadId If there exists multiple payloads, the index of payload in use need to be specified.

payload Fill the maximum payload including end effector and objects(kg)

cog to flange x/y/z: Set the center of gravity of load in the flange coordinate system.(According to the brand of robot, not required)

3.8.5 set_pick_state

Description

This Task generally is used for changing the pick state if picking failed

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

Pick State Select the pick state to `Hold` or `None` according to the previous logic

3.8.6 set_robotiq

Description

This Task enables the RobotIQ gripper to be controlled directly through Mech-Viz. RobotIQ gripper is controlled by serial port based on modbus protocol, PC or IPC can control its position/speed/force through 485 serial port (or USB to 485)

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

pos Range: 0~255; 0 is the state where the jaws are opened the most

speed Action speed, range: 0~255.

force Torque, range: 0~255.

comPort The cluster communication port number connected to RobotIQ, it can be viewed in the device manager

Init Robotiq Initialize the COM connection, it will be initialized automatically during running and simulation

Run Robotiq After connecting with Robotiq, you can click this button, it will move the gripper according to the set parameters to preview

3.8.7 tcp

Description

Used in scenarios where different TCPs need to be switched; connect this Task before grabbing, select the required end effector, and switch between different TCPs for grabbing.

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

Set EndEffector In the drop-down menu is the end effector that has been added in *Robot*, select the end effector to be used currently.

3.8.8 transfer_control

Description

This Task is used to temporarily transfer control to the robot and wait to continue to control the robot.

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks* for details.

Size The number of output ports.

Check Jps Whether to check the joint angle of the robot before transferring control.

Jps Before Transfer This option will only appear when the *Check Jps* option is checked. You need to enter the Jps value required to check the Jps.

3.9 Service

Service Task is mainly used for data interaction between Mech-Viz and external devices, and is generally used with adapters.

Note: It is required for such Tasks to be named in the properties, and notify the adapter developer.

3.9.1 notify

Description

Use this Task when the program proceeds to a node and the external device is required for the process

For example, when a certain branch is proceeded by the program and an external device is required to send a branch command again, a notify Task can be connected at the end of this branch, and a message is sent to the adapter to obtain the command of the external device

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

serviceName The adapter obtains the message of this Task by service name. If there are multiple notify Tasks in the project, all notify Tasks should use the same service name

message The adapter executes different logic according to the content of the message, such as if the branch operation ends, the message can be set to: finish

actionWhenFailed when the notify message is not sent successfully, the actions done by the software

needRobotStop the default is True, the robot will pause when proceeding to this Task; if set to False, the robot can send messages while running

timeout If no message is sent exceeding this time, perform the actions setted at actionWhenFailed, the unit is ms

..vision_proxy/vision_proxy

3.10 Tools

3.10.1 classify

Description

Run to different branches according to the object label corresponding to the actual grasp pose given by *visual_move*

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

getLabelFromUpdatePickedObj Usage scenario: The label of grasp object cannot be acquired from *visual_move*, it is acquired from second-time detection by *update_picked_obj* Once it is checked as True, the drop-down list *Select a update_picked_obj* will appear in the bottom, as shown in the picture, select the Task name which need to be classified.

Property	Value
Name	classify_1
Basic Non-Move	
Skip Execution	None ▼
Out Port Whe...	0
Get Label from U...	<input checked="" type="checkbox"/> True

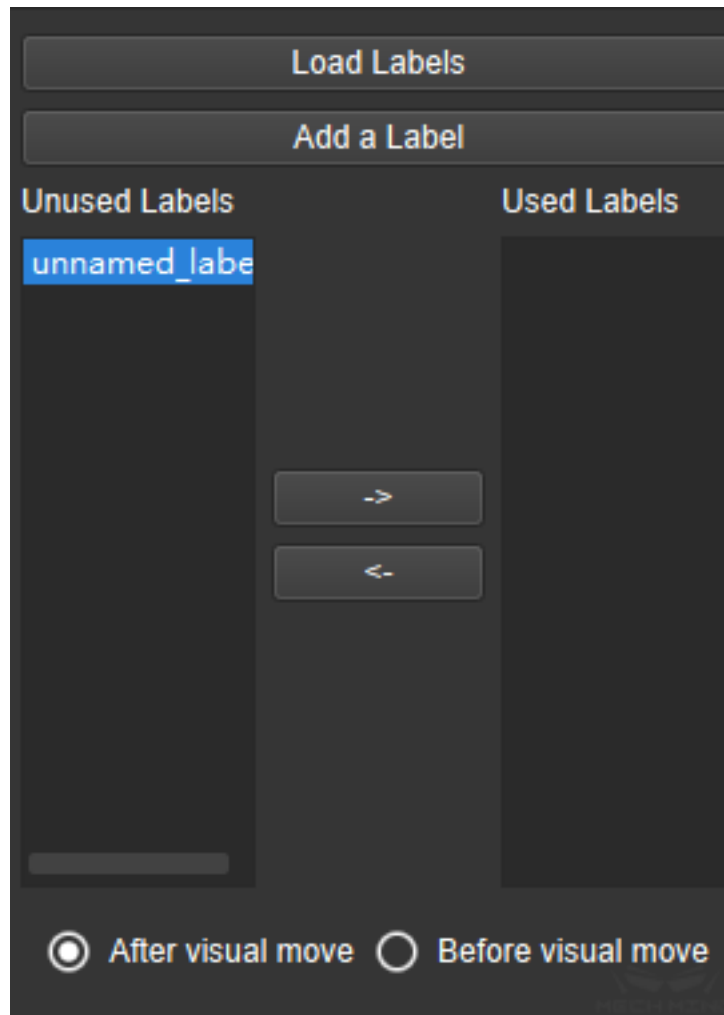
Unused Labels

Used Labels

Select a update_picked_obj:

▼

Load Labels Choose the labels.json file to load the predefined labels. Click or can add or delete corresponding port of labels



After visual move Default unselected, it can be choose when the objects need to be placed in different position according to labels after picking. According to the object label corresponding to the actual grasp pose given by *visual_move* , the program will run into different branch

Before visual move If the grasping depends on the labels given by current detection so that using different end effectors, this function can be choose. Due to this function is relate to the actual picked object's label given by subsequent *visual_move* , the dependent Task need to be selected in the drop-down list *Select a visual_move*

3.10.2 counter

Description

Count the execution times of mission or *branch_by_msg* , or count the picking times. Generally it is used with *reset_task* .

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

countertype Types of counts: execution and pickedcount

3.10.3 finish_checker

Description

It is used to ensure *Basic Move* with index such as *move_list* , *move_grid* ,etc. finish execution in a single cycle to avoid loops.

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

Select indexed move Select the move-type Task which need to be checked. Exit from port 1 if it is completed, exit from port 0 if it is not completed.

3.10.4 index_change

Description

This Task can be used with all the *Basic Move* which has the property of *curlIndex* , or *counter* . It will change the *curlIndex* of specified move-type Task or the *count* of counter according to *step* every execution.

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

step The number that increase/decrease the current index or count.

For example: If the step is 0, it will not change the index/count of specified Task, and the Task will accumulate the execution times; While the step is -1, the index/count of specified Task will minus 1 every execution; so on and so forth.

Select counter or indexed move Select the *counter* whose count need to be changed, or *Basic Move* whose index need to be changed.

3.10.5 reset_task

Description

Reset the selected Task, it can be used:

- To reset the indexed move or counter, such as *move_grid* , *counter*
- To reset the move-type Task which move target is "Place", it will clear the placed object in scene
- To reset the last *visual_move* when multiple *visual_move* share vision results, it will clear the vision result of this detection
- To trigger some special function. Such as Pallet task which *needAdjustPalletPose* is checked, except the first time running, it will also update the pose of pallet after being reset

Parameters

Select indexed move or counter Select the Task name which need to be reset in the drop-down list

3.10.6 wait

Description

When execute this Task, wait for the specified time, unit: ms

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

wait_time The duration to wait, the unit is ms

3.11 Trajectory

The Task in this group is applied to gluing.

3.11.1 trajectory_procedure

Description

Perform multiple continuous dense point movements according to the template file of trajectory point. It need to be used with *visual_move* .

Parameters

trajPointJsonPath Pre-edited trajectory point json file. When the program is running, the number of points given by the vision service must be consistent with the number of points in the template.

Recommended solution of moving continuous trajectory

If all the gluing trajectory need to be moved at once without interrupt, it is recommended to check the *useAll* in *visual_move* .

Note: The number of trajectory points provided by the vision service in this way does not need to be fixed, and it is relatively simple to use.

3.12 Vision

3.12.1 check_look

Description

Connect with *visual_look* to make project do different processes based on the vision result.

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

Get Vision Services Select *visual_look* which need to be checked

Branch explanation

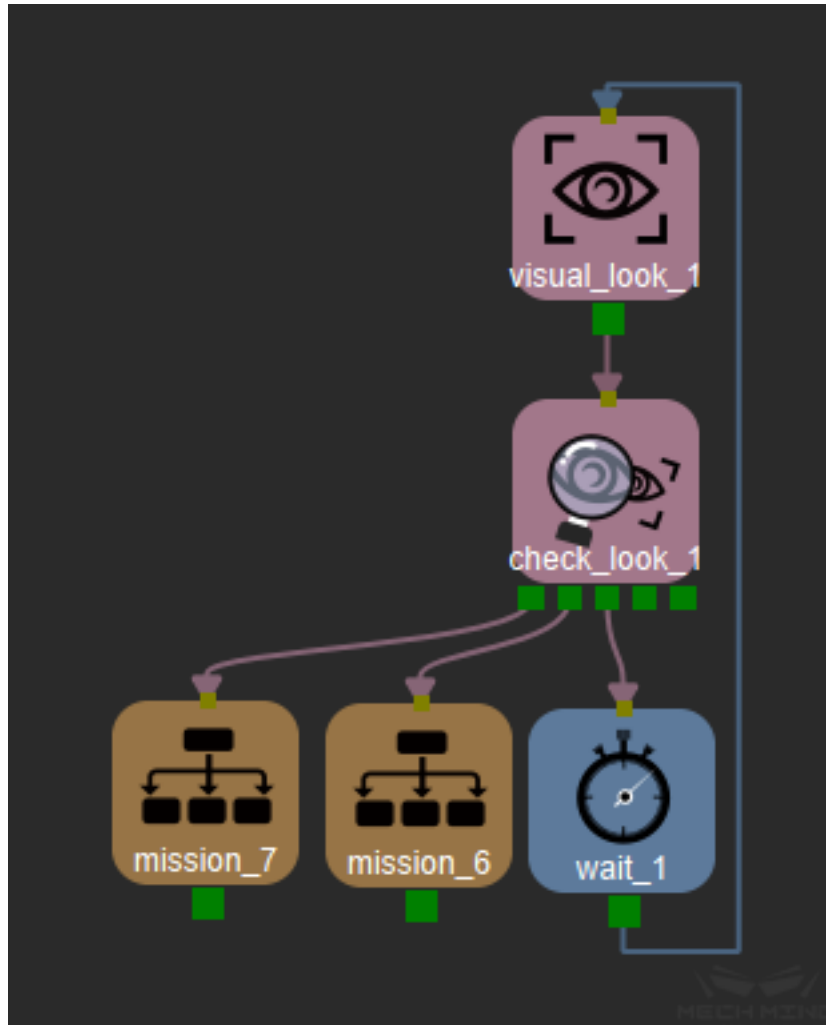
hasResult: If the visual recognition provides the result, then take this exit.

noResult: If the visual recognition does not provide the result, then take this exit.

notCalled: Applicable to multiple *visual_look*, used in loop nesting. This exit can be ignored if it follows *visual_look*.

unfinished: The visual processing is not finished. This exit usually follows waiting and returns to the entrance.

noCloudInRoi: No point cloud is in the Region of Interest (ROI) of vision (It is often used to determine whether there are any object and distinguish unidentified objects).



3.12.2 update_pallet_pose

Description

Update the pallet pose according to the vision result. The camera can be triggered at any time/robot pose to update the pallet pose.

If the pallet pose need to be updated before each palletizing, check *AdjustPalletViaVision* in the task of *Pallet*.

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

visionToUpdatePallet The name of the vision project which will update the pallet pose

Select palletizing task Select the name of the palletizing Task that needs to update the pallet pose in the project

3.12.3 update_picked_obj

Description

Update the size and pose of the grasped object by second-detection

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

visionName The vision project name to second detect

prePlanOutPort Default is 0, module will output from the left port "NoNeedUpdate" to speed up the project. It can be set to -1 if it is not necessary to preplan.

Accuracy threshold sizeThre/transThre/rotThre: if the size/translations/rotation of box are all smaller than the specified threshold, it will be considered as the same, this Task will output from the left port "NoNeedUpdate"

3.12.4 update_scene_object

Contents will be added soon!

3.12.5 vision_result_used_up

Description

This Task is used only if the *reuseVisionResult* of *visual_move* is checked. Check if all visual results in the current detection have been used up. This task is only for checking.

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

Select visual move: Select *visual_move* which need to be checked

3.12.6 visual_look

Description

Call the vision service (run vision project) and transmit the result to Mech-Viz

ROI: Region of Interest

Parameters

BasicNonMove See *General Parameters of Non-Move Tasks*

Vision name Select the vision project name registered on Mech-Center; click Get Vision Services to refresh the registered services.

3.12.7 visual_move

Function

This Task guides the robot to move according to the poses received from vision services.

Exit Port	Description
Success	The path was planned successfully.
Plan failure	The path planning failed.
Other failures	There are no available poses for planning. Possible reasons: Mech-Vision did not output poses. The poses output by Mech-Vision cannot meet the requirement.

Sample Scenario

This Task is usually used when the robot picks objects.

Parameters

General Parameters

Please refer to *General Parameters of Move-Type Tasks* for detailed descriptions.

Critical

All Targets in One Move

Once this option is selected, the robot will move through all poses received from the vision service in sequence at one time.

This option is usually used when the robot moves in a fixed path where no DO signals will be sent, such as the gluing application.

This parameter cannot be adjusted with other parameters at the same time. Once this option is selected, other parameters will be hidden.

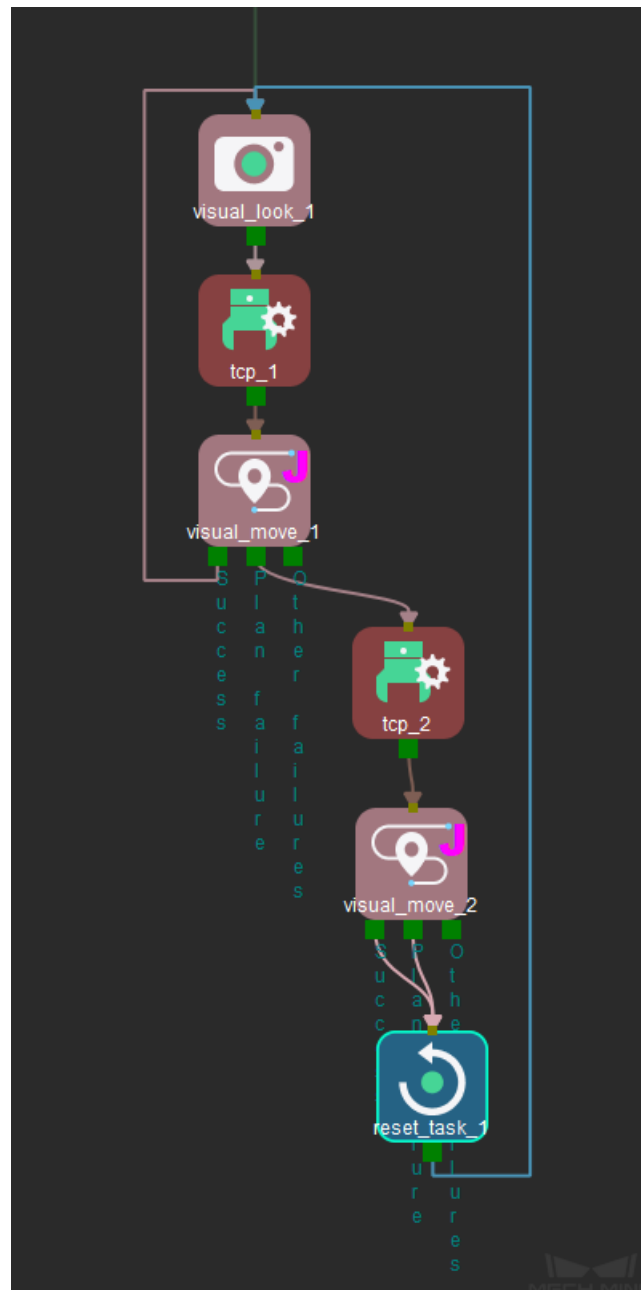
Reuse Vision Result

When there are multiple pickable objects in one vision result and it can be guaranteed that **picking one object will not affect the poses of other objects**, this option can be selected. The vision result will be reused and no new image will be captured until all pickable objects have been picked.

Share Vision Result

This option can be used to share vision results from the same *visual_look* Task among multiple *visual_move* Tasks. You should specify a **Vision Name** once this option is selected.

Application example: When the objects cannot be picked easily, various end effectors may be used to pick the object for several times. If the first attempt to pick fails, the current vision result will be discarded by default. If **Share Vision Result** is selected, the vision result will be used again when another end effector is used and you do not need to capture an image again. Please connect a *reset_task* Task after the last *visual_move* Task to clear the shared vision result, or else the old vision result will be used repeatedly until all pickable objects have been picked even if the poses of other objects have been changed and a new vision result is available, which may cause an error in the project.

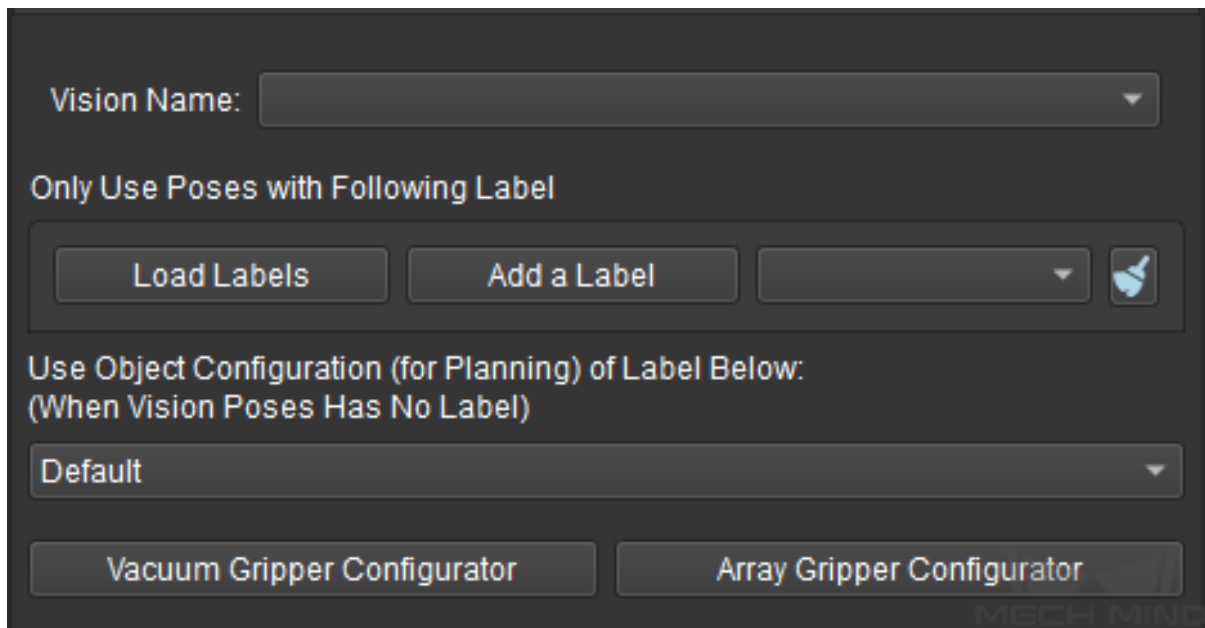



Operational Mode

There are three types of operational modes as shown below, and each of them has different parameters.

1. *Regular Mode*
2. *Array Gripper*
3. *Depallet Vacuum Gripper*

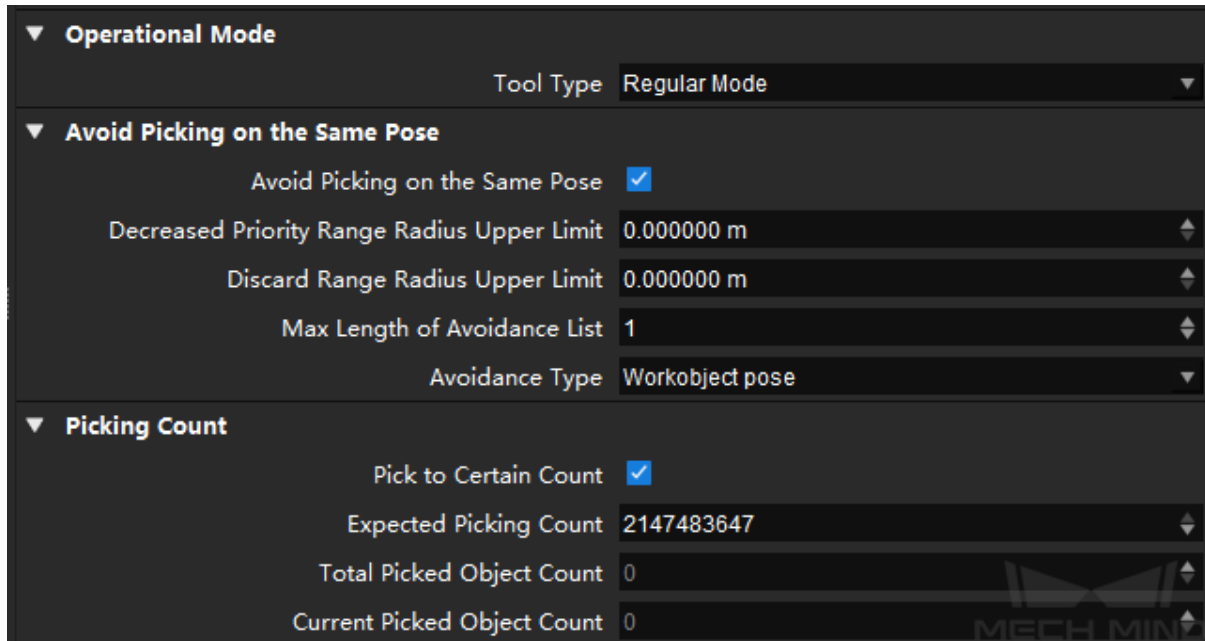
Other Settings



Parameter	Description
Vision Name	The project selected here is usually the same as the one in <i>visual_look</i> .
Only Use Poses with Following Label	The vision result may contain different labels. Once a specific label is selected, only the pickable objects with the selected label will be picked.
	Click <i>Load Labels</i> or <i>Add a Label</i> to select the label. Once the label is selected successfully, the label name will be displayed on the button on the right.
	Click  to delete the selected label.
Use Object Configuration (for Planning) of Label Below:	Select the object label set in <i>Tools and Workobjects</i> → <i>Configuration of Objects to Pick</i> , and the corresponding object configuration will be applied in <i>visual_move</i> .
	Select Default when there is no label in the vision result.
	Select the corresponding label in the vision result when there is an available one.
Vacuum Gripper Configurator	Please refer to <i>Vacuum Gripper Configurator</i> for detailed information.
Array Gripper Configurator	Please refer to <i>Array Gripper Configurator</i> for detailed information.

Regular Mode

The parameters in the regular mode are shown in the figure below.



Avoid Picking on the Same Pose

This parameter group is mainly used to avoid picking on the same pick point in scenarios where the object cannot be picked successfully.

Decreased Priority Range Radius Upper Limit

Default setting: 0

Description: If the distance between the current pick point and the most recently tried pick point is less than this value, the two pick points will be considered the same one, and the priority for picking will be downgraded.

Discard Range Radius Upper Limit

Default setting: 0

Description: If the distance between the current pick point and the lastly tried pick point is less than this value, the pick point will be discarded for picking.

Example: For example, if the robot only moves the workpiece but fails to pick it on the first attempt, there is a possibility of successful picking on the next attempt. In this case, **Decreased Priority Range Radius Upper Limit** can be set to downgrade the priority for picking the workpiece, while the object pose will be kept. When the robot fails to move the workpiece at all in the first attempt, it is highly unlikely that the workpiece can be picked successfully in the next attempt, and therefore **Discard Range Radius Upper Limit** can be set to discard the pose directly.

Max Length of Avoidance List

Default setting: 1

Description: The maximum length of the avoidance list.

Example: Assuming that the value is set to 2 and Mech-Vision outputs 3 poses when pose 1 is used and the picking fails, pose 1 will be recorded. Pose 2 will also be recorded if the picking fails. However, when pose 3 is recorded, pose 1 will be discarded, and only pose 2 and pose 3 will be kept in the avoidance list.

Avoidance Type

Workobject pose: Record the pose of the workobject. If the object has 3 pick points, and one of them was recorded as used during a failed picking, the rest two will not be included in the avoidance list.

Workobject: Record the workobject. If the object has 3 pick points, and one of them was recorded as used during a failed picking, the rest two will be included in the avoidance list as well and the object will not be prioritized for picking next time.

Picking Count

This parameter group is used to count the picked objects and calculate the rest objects to be picked. Once you enter an **Expected Picking Count**, the **Total Picked Object Count** and **Current Picked Object Count** will be calculated automatically.

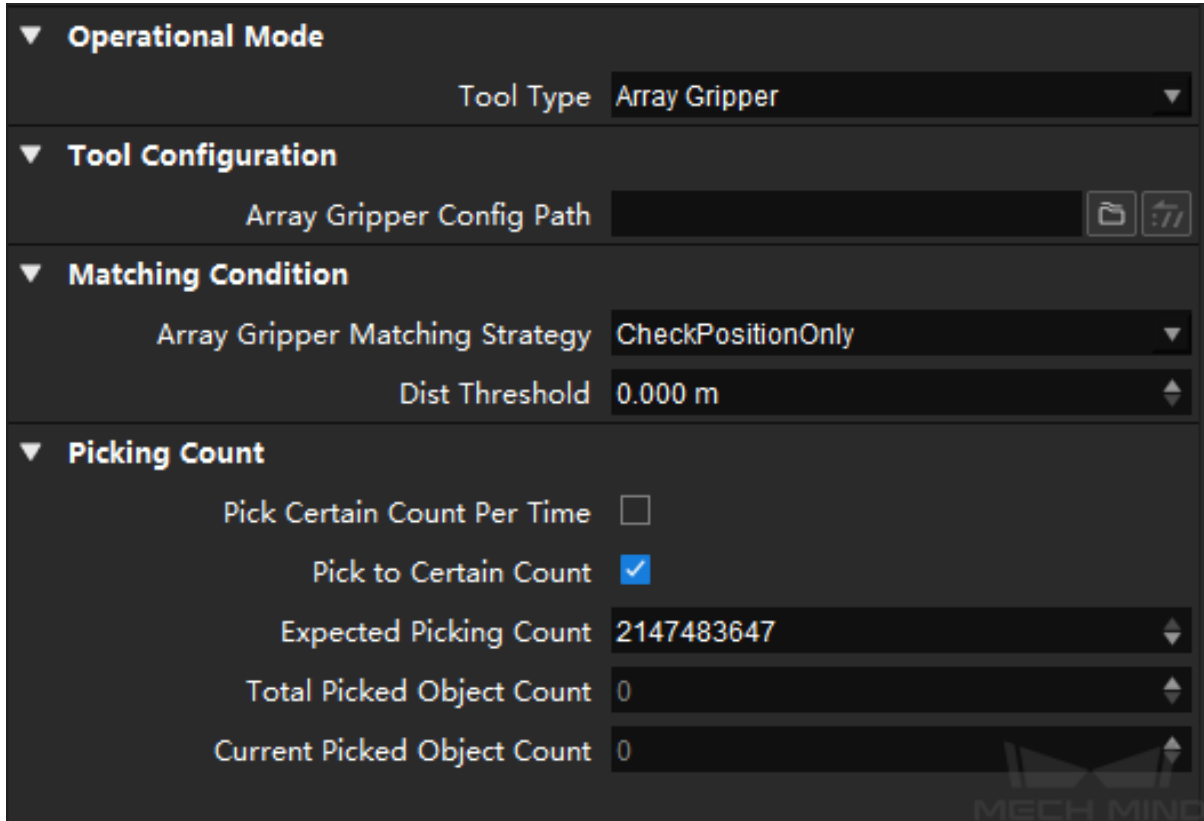
Expected Picking Count The maximum number of picked objects.

Total Picked Object Count The number of picked objects, which is counted automatically and cannot be modified.

Current Picked Object Count The number of the currently picked object, which is counted automatically and cannot be modified.

Array Gripper

At present, the array gripper mode only supports multiple end tools that are arranged in a single row. The parameters in the array gripper mode are shown in the figure below.



Tool Configuration

Array Gripper Config Path Click  to select the array gripper configuration file. You can use the [Array Gripper Configurator](#) to configure the array gripper and export the configuration file in JSON format.

Matching Condition

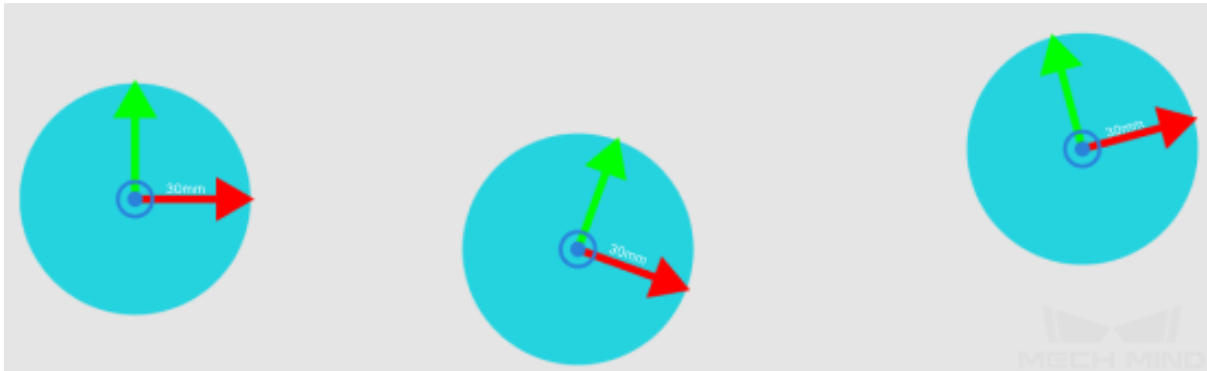
Array Gripper Matching Strategy When **CheckPositionOnly** is selected, you can only adjust **Dist Threshold**. When **CheckPositionAndOrientation** is selected, you can adjust both **Dist Threshold** and **Angle Threshold**.

Dist Threshold

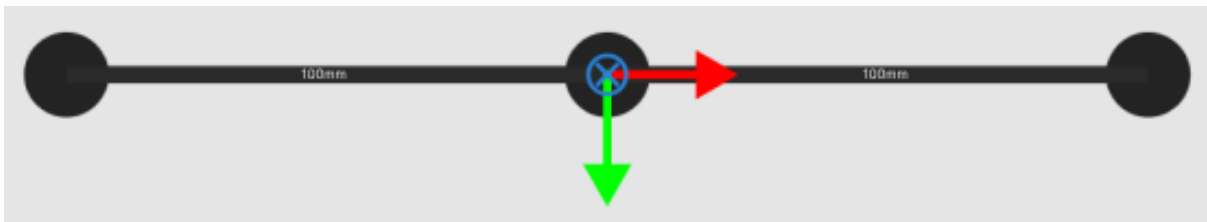
Description: The picking pose will be within a circle with the object pose as the center and the set **Dist Threshold** as the radius.

Examples:

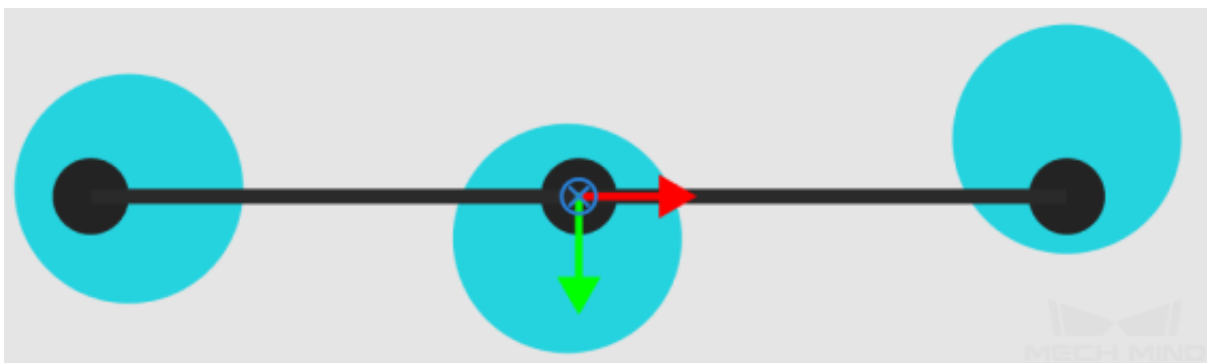
1. When the **Dist Threshold** is set to 30 mm, the possible areas for picking are shown in the figure below.



2. An array gripper with 3 end tools that are 100 mm away from each other is shown in the figure below.



3. In the path planning, the software will find a position where all three TCPs fit in the possible areas for picking (the blue circles as shown below).



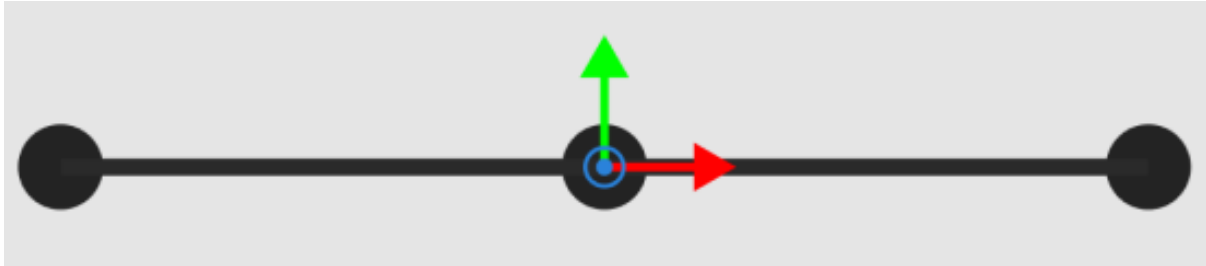
Hint: If three TCPs cannot be matched with the possible areas for picking at the same time, the software will try to match two TCPs instead.

Angle Threshold

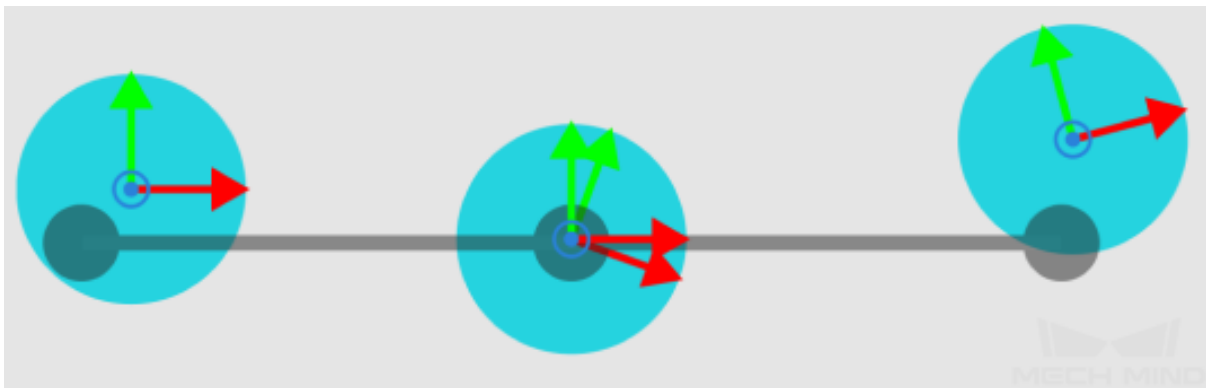
Description: The angle between the picking pose and the TCP.

Examples:

1. The TCP will be rotated 180° around its own X-axis and its Z-axis will be upward and parallel with that of the picking pose.



2. If the angle between the picking pose and the TCP is within the **Angle Threshold**, the picking pose will be used for picking, or else it will be discarded.



Hint: In the **Array Gripper** mode, it is acceptable that the objects in the middle of a combination is missed as long as the combination is not rotationally symmetric. For example, if an array gripper has 4 end tools which are numbered as 0, 1, 2, and 3, and there are 3 objects which are arranged as OOXO (O represents the object and X represents the empty place), tool 0, 1, and 3 should be started. After being rotated 180°, the combination changes from OOXO to OXOO, which is not symmetric.

Picking Count

This parameter group is used to count the picked objects and calculate the rest of the objects to be picked. Once you enter an **Expected Picking Count**, the **Total Picked Object Count** and **Current Picked Object Count** will be calculated automatically.

Expected Picking Count The maximum number of picked objects.

Total Picked Object Count The number of picked objects, which is counted automatically and cannot be modified.

Current Picked Object Count The number of the picked object this time, which is counted automatically and cannot be modified.

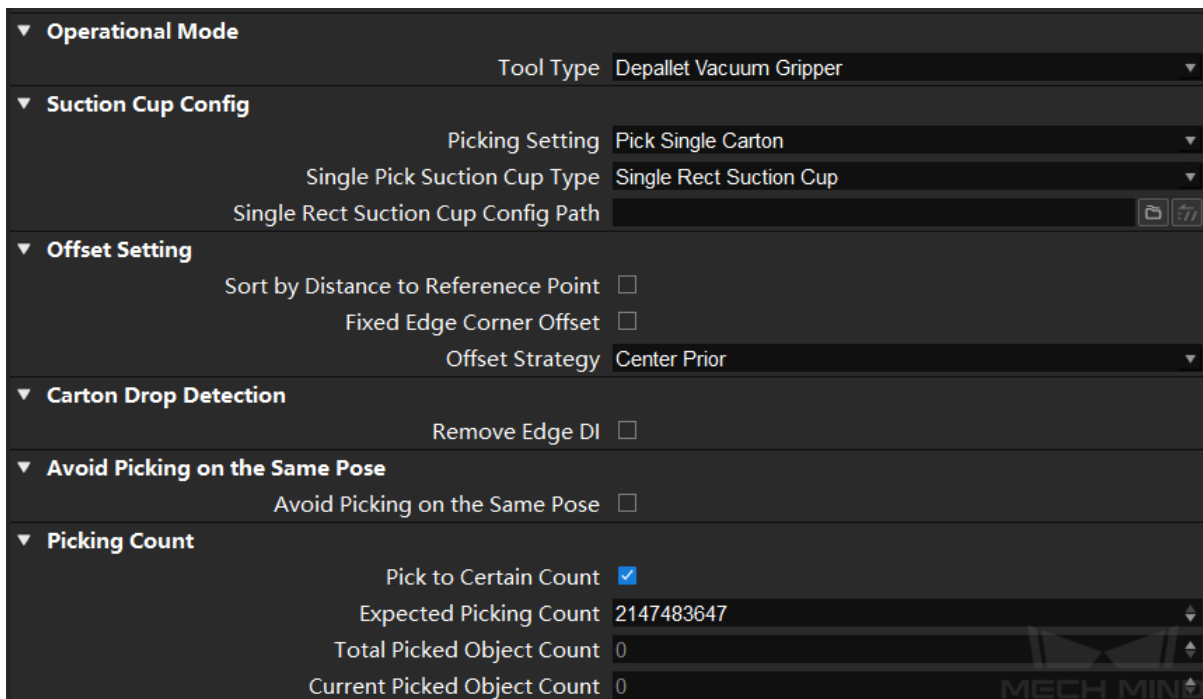
Depallet Vacuum Gripper

According to different parameters set in **Picking Setting** and **Single Pick Suction Cup Type**, there are four parameter combinations in the Depallet Vacuum Gripper mode as shown below.


1. *Pick Single Carton + Single Rect Suction Cup*
2. *Pick Single Carton + Side Suction Cup*
3. *Pick Multi Carton + Single Rect Suction Cup*
4. *Pick Multi Carton + Parallel Suction Cups*

Pick Single Carton + Single Rect Suction Cup

The parameters are shown below.



Suction Cup Config

Single Rect Suction Cup Config Path Click  to select the suction cup configuration file. You can use the *Vacuum Gripper Configurator* to configure the suction cups and output a configuration file in JSON format.

Offset Setting

Offset Strategy

Center Prior: A picking pose without offset (the centers of the suction cup and the carton coincide) will be prioritized in path planning. If the software fails to plan a path, other picking poses with offsets will be tried.

Corner Prior: A picking pose with offset will be prioritized in path planning, and poses without offset will be tried last.

Corner Alignment Only: Picking poses without offsets will not be considered in path planning.

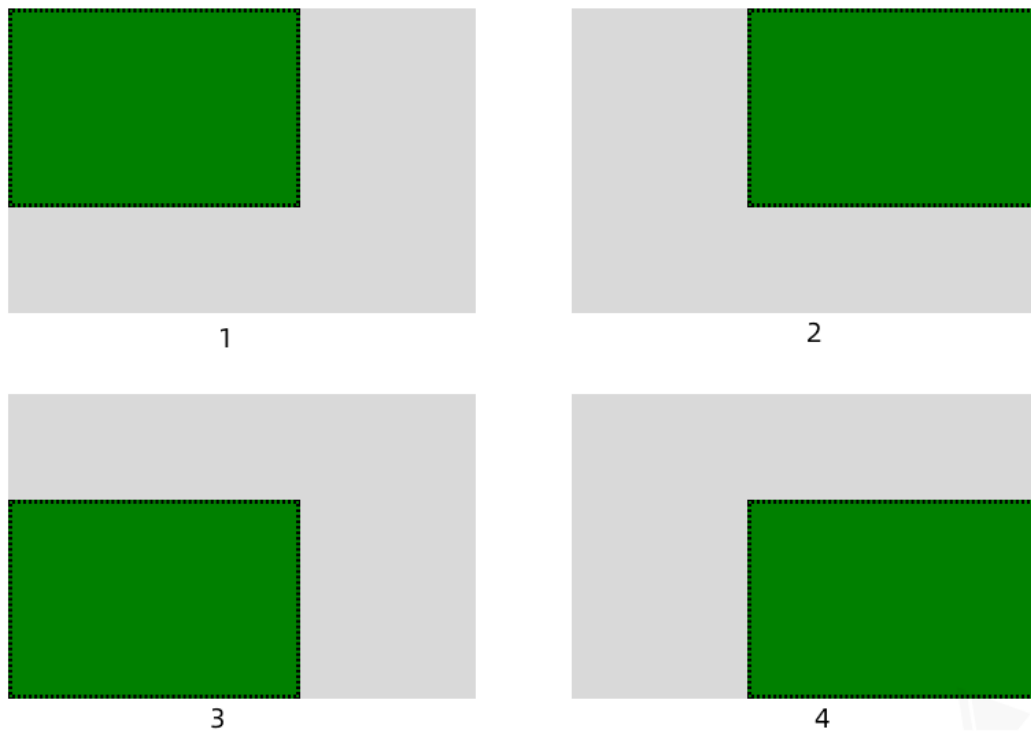
Center Prior is applicable to mixed case depalletizing to avoid collisions. If the cartons do not need to be stacked on a pallet after being picked (e.g., just need to be placed on a conveyor belt), this offset strategy is not necessary.

Sort by Distance to Reference Point - Reference Point X/Y

Instruction: You need to set a reference point if you want to enable this parameter, and the robot will prioritize picking poses of which the suction cup is close to the reference point. This setting and the object symmetry settings cannot be enabled at the same time.

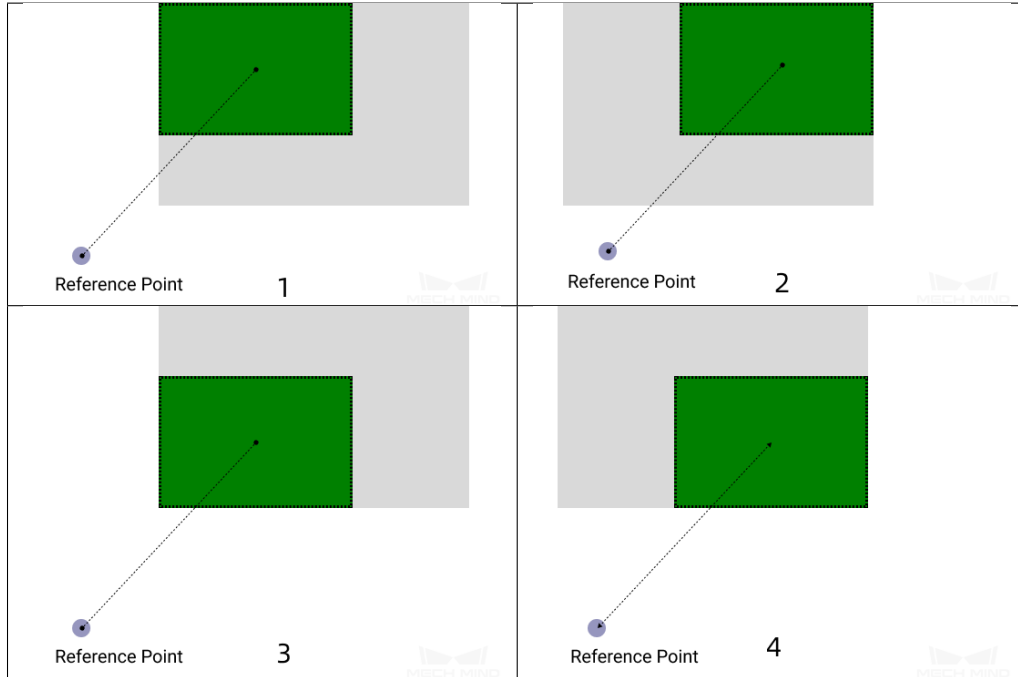
Example:

As shown in the figure below (top view perspective), the green rectangles represent cartons, and the gray rectangles represent suction cups. There are four picking strategies in total.



The reference point is in the lower left corner to the carton, and the relative position of the reference point and the carton is fixed. Picking poses of which the suction cup is nearest to the reference point will be tried first.

Therefore, the picking poses in the figure below will be tried in the order of $2 \rightarrow 1 \rightarrow 4 \rightarrow 3$.



Keep Only High Priority Offset—Max Plan Results Count

Instruction: The default setting is 2. Please set the count according to the total planning results when a reference point is introduced.

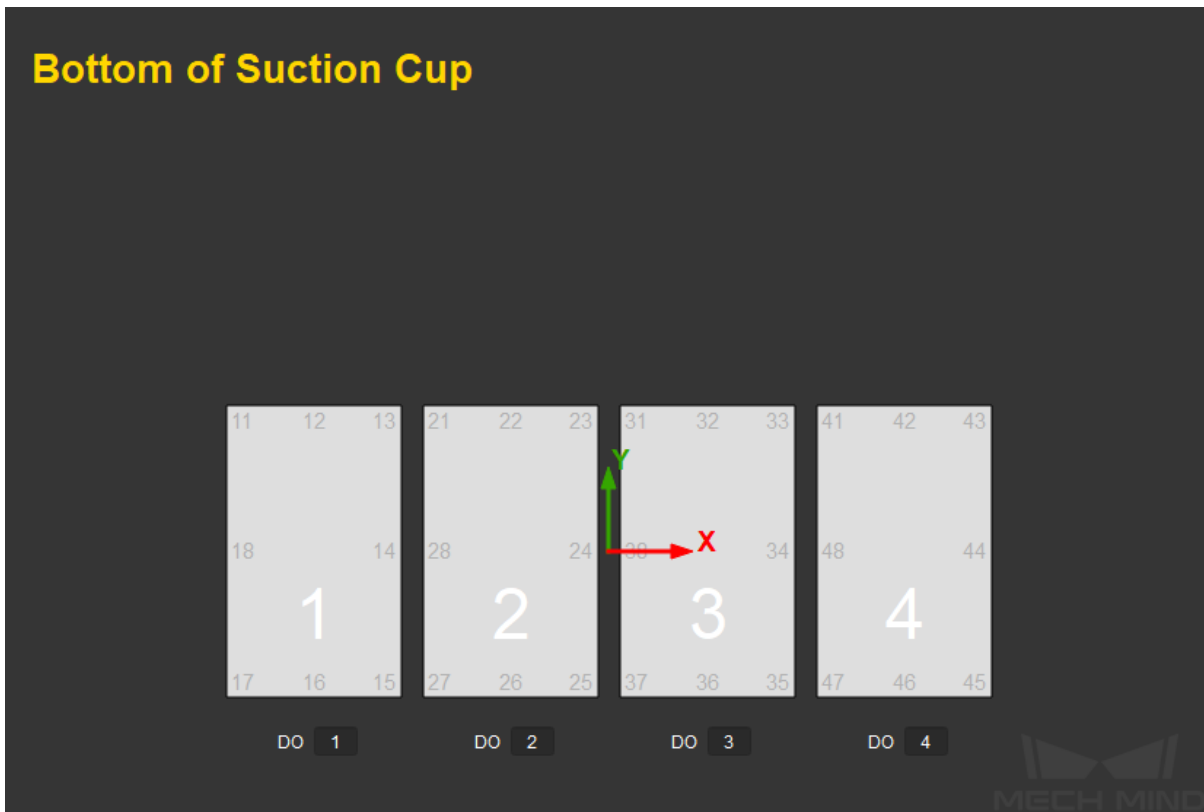
Example: If this option is enabled, only planning results No. 2 and No. 1 in the above example will be kept.

Fixed Edge Corner Offset—Fixed Edge Corner Offset

Enter the edge corner label numbers.

The numbering rules of the edge/corner: The first digit of the label number indicates the number of the suction cup, and the second digit of the label number indicates whether it is an edge or corner, where 1, 3, 5, and 7 represent the corner and 2, 4, 6, and 8 represent the edge. 0 represents the center of the suction cup.

Bottom of Suction Cup



Please refer to *Offset Strategy* for detailed information.

Avoid Picking on the Same Pose

This parameter group is mainly used to avoid picking with the same picking pose in scenarios where the object cannot be picked successfully.

Decreased Priority Range Radius Upper Limit

Default setting: 0

Description: If the distance between the current pick point and the most recently tried pick point is less than this value, the two pick points will be considered the same one and the priority for picking will be downgraded.

Discard Range Radius Upper Limit

Default setting: 0

Description: If the distance between the current pick point and the most recently tried pick point is less than this value, the pick point will be discarded for picking.

Example: For example, if the robot only moves the workpiece but fails to pick it in the first attempt, there is a possibility of successful picking in the next attempt. In this case, **Decreased Priority Range Radius Upper Limit** can be set to decrease the priority for picking the workpiece, while the object pose will be kept. When the robot fails to move the workpiece at all in the first attempt, it is highly unlikely that the workpiece can be picked successfully in the next attempt, and therefore **Discard Range Radius Upper Limit** can be set to discard the pose directly.

Max Length of Avoidance List

Default setting: 1

Description: The maximum length of the avoidance list.

Example: Assuming that the value is set to 2 and Mech-Vision outputs 3 poses, when pose 1 is used and the picking fails, pose 1 will be recorded. Pose 2 will also be recorded if the picking fails. However, when pose 3 is recorded, pose 1 will be discarded and only pose 2 and pose 3 will be kept in the avoidance list.

Avoidance Type

Workobject pose: Record the pose of the workobject. If the object has 3 pick points, and one of them was recorded as used during a failed picking, the rest two will not be included in the avoidance list.

Workobject: Record the workobject. If the object has 3 pick points, and one of them was recorded as used during a failed picking, the rest two will be included in the avoidance list as well and the object will not be prioritized for picking next time.

Picking Count

This parameter group is used to count the picked objects and calculate the rest objects to be picked. Once you enter an **Expected Picking Count**, the **Total Picked Object Count** and **Current Picked Object Count** will be calculated automatically.



Expected Picking Count The maximum number of picked objects.

Total Picked Object Count The number of picked objects, which is counted automatically and cannot be modified.

Current Picked Object Count The number of the currently picked object, which is counted automatically and cannot be modified.


Pick Single Carton + Side Suction Cup

The parameters are shown below.

▼ Operational Mode		Tool Type	Depallet Vacuum Gripper
▼ Suction Cup Config		Picking Setting	Pick Single Carton
		Single Pick Suction Cup Type	Side Suction Cup
		Carton Border to Suck	Unspecified
		Single Rect Suction Cup Config Path	 
▼ Offset Setting		Sort by Distance to Reference Point	<input type="checkbox"/>
		Fixed Edge Corner Offset	<input type="checkbox"/>
▼ Avoid Picking on the Same Pose		Avoid Picking on the Same Pose	<input type="checkbox"/>
▼ Picking Count		Pick to Certain Count	<input checked="" type="checkbox"/>
		Expected Picking Count	2147483647
		Total Picked Object Count	0
		Current Picked Object Count	0

Suction Cup Config

Carton Border to Suck You can select from **Unspecified**, **Long Side**, and **Short Side** according to the actual situation.

Single Rect Suction Cup Config Path Click  to select the suction cup configuration file. You can use the [Vacuum Gripper Configurator](#) to configure the suction cups and output a configuration file in JSON format.

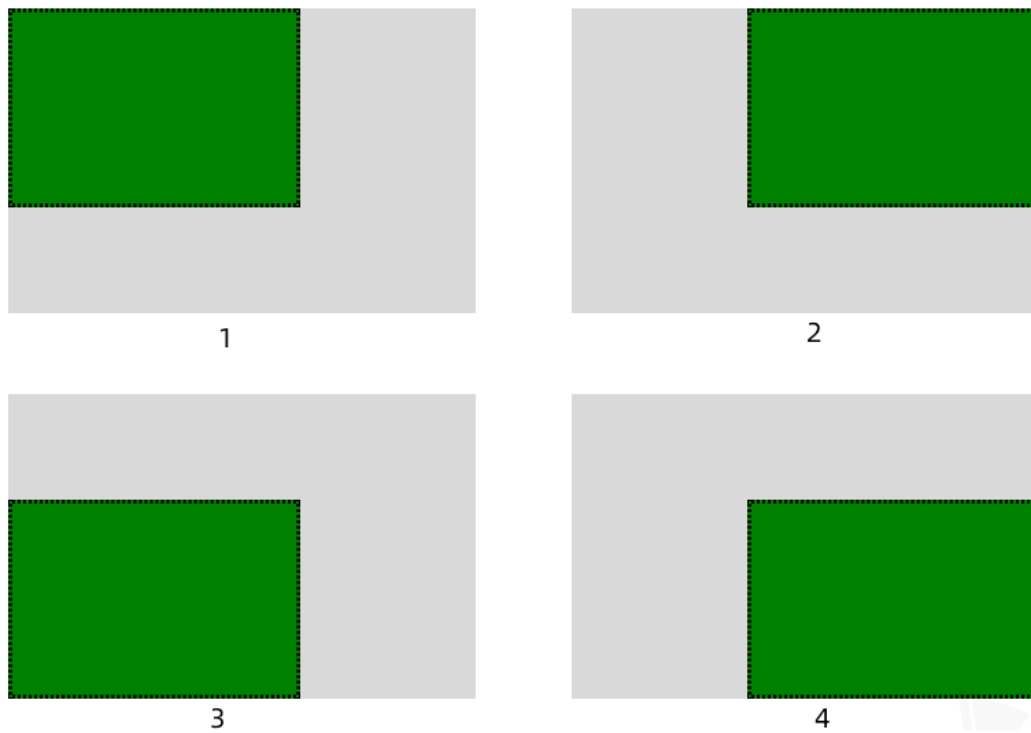
Offset Setting

Sort by Distance to Reference Point - Reference Point X/Y

Instruction: You need to set a reference point if you want to enable this parameter, and the robot will prioritize picking poses of which the suction cup is close to the reference point. This setting and the object symmetry settings cannot be enabled at the same time.

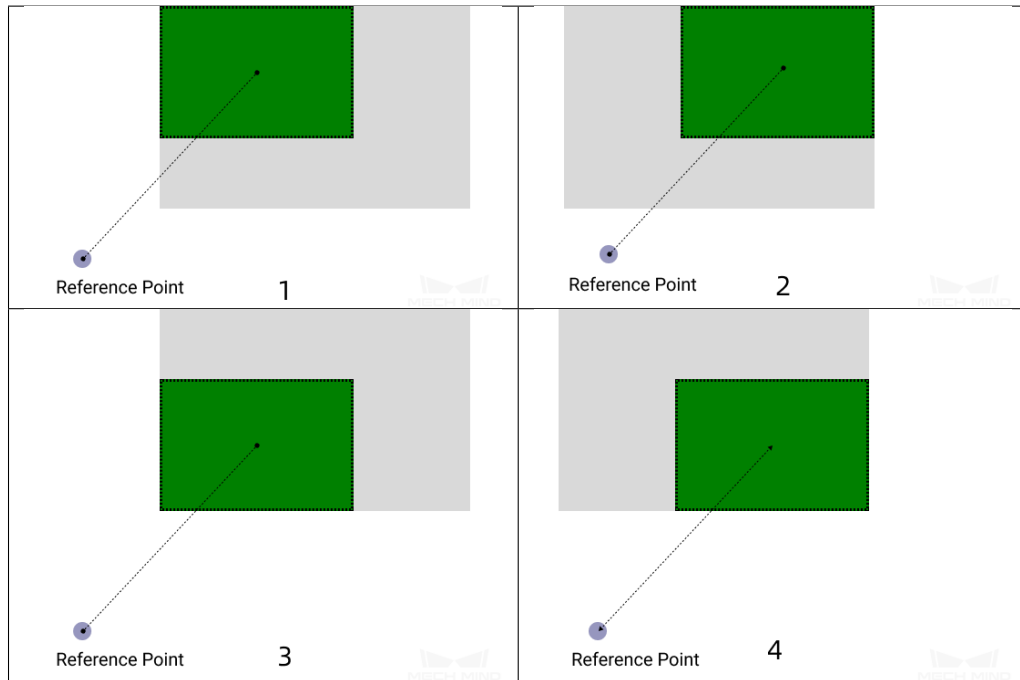
Example:

As shown in the figure below (top view perspective), the green rectangles represent cartons, and the gray rectangles represent suction cups. There are four picking strategies in total.



The reference point is in the lower left corner to the carton, and the relative position of the reference point and the carton is fixed. Picking poses of which the suction cup is nearest to the reference point will be tried first.

Therefore, the picking poses in the figure below will be tried in the order of $2 \rightarrow 1 \rightarrow 4 \rightarrow 3$.



Keep Only High Priority Offset—Max Plan Results Count

Instruction: The default setting is 2. Please set the count according to the total planning results when a reference point is introduced.

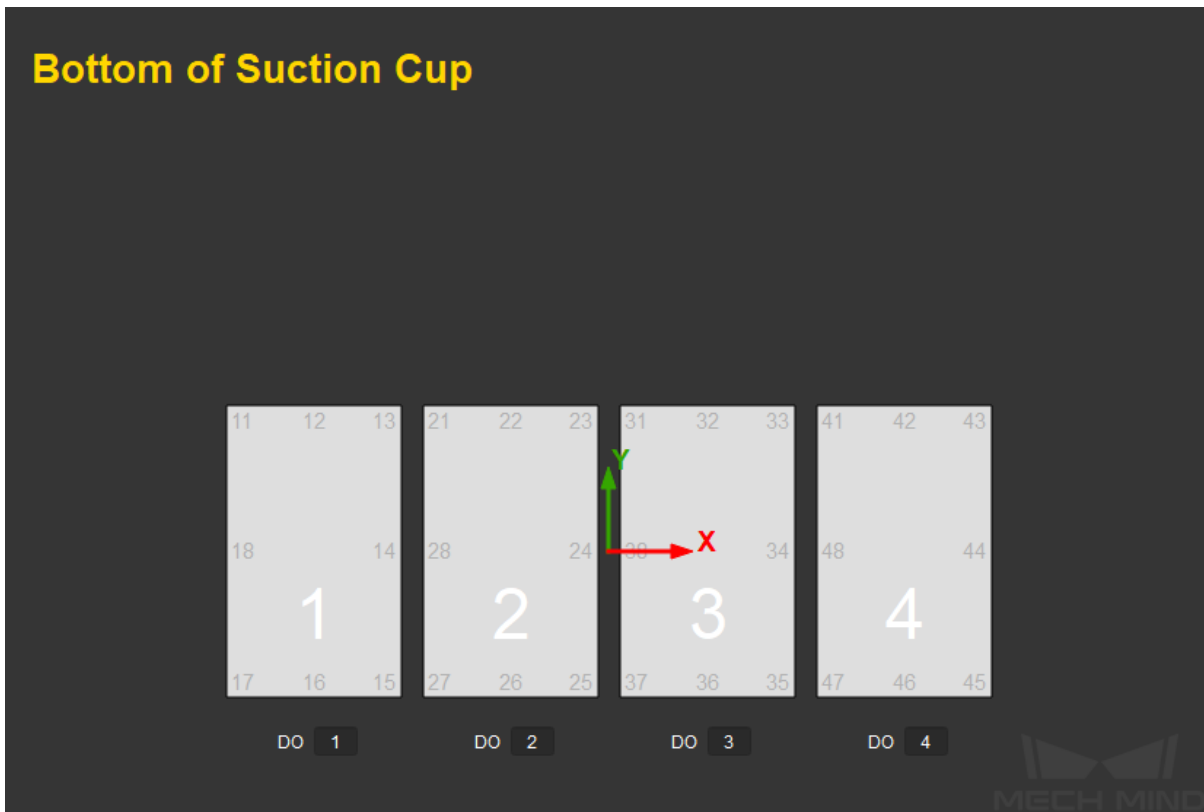
Example: If this option is enabled, only planning results No. 2 and No. 1 in the above example will be kept.

Fixed Edge Corner Offset—Fixed Edge Corner Offset

Enter the edge corner label numbers.

The numbering rules of the edge/corner: The first digit of the label number indicates the number of the suction cup, and the second digit of the label number indicates whether it is an edge or corner, where 1, 3, 5, and 7 represent the corner and 2, 4, 6, and 8 represent the edge. 0 represents the center of the suction cup.

Bottom of Suction Cup



Please refer to [Offset Strategy](#) for detailed information.

Avoid Picking on the Same Pose

This parameter group is mainly used to avoid picking with the same picking pose in scenarios where the object cannot be picked successfully.

Decreased Priority Range Radius Upper Limit

Default setting: 0

Description: If the distance between the current pick point and the most recently tried pick point is less than this value, the two pick points will be considered the same one and the priority for picking will be downgraded.

Discard Range Radius Upper Limit

Default setting: 0

Description: If the distance between the current pick point and the most recently tried pick point is less than this value, the pick point will be discarded for picking.

Example: For example, if the robot only moves the workpiece but fails to pick it in the first attempt, there is a possibility of successful picking in the next attempt. In this case, **Decreased Priority Range Radius Upper Limit** can be set to decrease the priority for picking the workpiece, while the object pose will be kept. When the robot fails to move the workpiece at all in the first attempt, it is highly unlikely that the workpiece can be picked successfully in the next attempt, and therefore **Discard Range Radius Upper Limit** can be set to discard the pose directly.

Max Length of Avoidance List

Default setting: 1

Description: The maximum length of the avoidance list.

Example: Assuming that the value is set to 2 and Mech-Vision outputs 3 poses, when pose 1 is used and the picking fails, pose 1 will be recorded. Pose 2 will also be recorded if the picking fails. However, when pose 3 is recorded, pose 1 will be discarded and only pose 2 and pose 3 will be kept in the avoidance list.

Avoidance Type

Workobject pose: Record the pose of the workobject. If the object has 3 pick points, and one of them was recorded as used during a failed picking, the rest two will not be included in the avoidance list.

Workobject: Record the workobject. If the object has 3 pick points, and one of them was recorded as used during a failed picking, the rest two will be included in the avoidance list as well and the object will not be prioritized for picking next time.

Picking Count

This parameter group is used to count the picked objects and calculate the rest objects to be picked. Once you enter an **Expected Picking Count**, the **Total Picked Object Count** and **Current Picked Object Count** will be calculated automatically.



Expected Picking Count The maximum number of picked objects.

Total Picked Object Count The number of picked objects, which is counted automatically and cannot be modified.


Current Picked Object Count The number of the currently picked object, which is counted automatically and cannot be modified.

Pick Multi Carton + Single Rect Suction Cup

The parameters are shown below.

▼ Operational Mode	
Tool Type	Depallet Vacuum Gripper
▼ Suction Cup Config	
Picking Setting	Pick Multi Carton
Multi Pick Suction Cup Type	Single Rect Suction Cup
Single Rect Suction Cup Config Path	 
▼ Carton Combination	
Carton Combination Strategy	Free
Dist Thre for Carton Combination	0.000000 m
Angle Thre for Carton Combination	0.00000 rad
Calc All Combination Plan	<input type="checkbox"/>
▼ Offset Setting	
Sort by Distance to Referenece Point	<input type="checkbox"/>
Fixed Edge Corner Offset	<input type="checkbox"/>
Offset Strategy	Center Prior
Suction Cup X Direction	Parallel With Carton Group Long Side
▼ Carton Drop Detection	
Remove Edge DI	<input type="checkbox"/>
▼ Picking Count	
Pick to Certain Count	<input checked="" type="checkbox"/>
Expected Picking Count	2147483647
Total Picked Object Count	0
Current Picked Object Count	0
Picking Entire Row	<input type="checkbox"/>
Pick Certain Count Per Time	<input type="checkbox"/>

Suction Cup Config

Single Rect Suction Cup Config Path Click  to select the suction cup configuration file. You can use the [Vacuum Gripper Configurator](#) to configure the suction cups and output a configuration file in JSON format.

Carton Combination

Calc All Combination Plan

Select to calculate all possible carton combinations when different **Carton Combination Strategies** are used.

Please refer to [Carton Combination](#) for detailed information.

Offset Setting

Offset Strategy

Center Prior: A picking pose without offset (the centers of the suction cup and the carton coincide) will be prioritized in path planning. If the software fails to plan a path, other picking poses with offsets will be tried.

Corner Prior: A picking pose with offset will be prioritized in path planning, and poses without offset will be tried last.

Corner Alignment Only: Picking poses without offsets will not be considered in path planning.

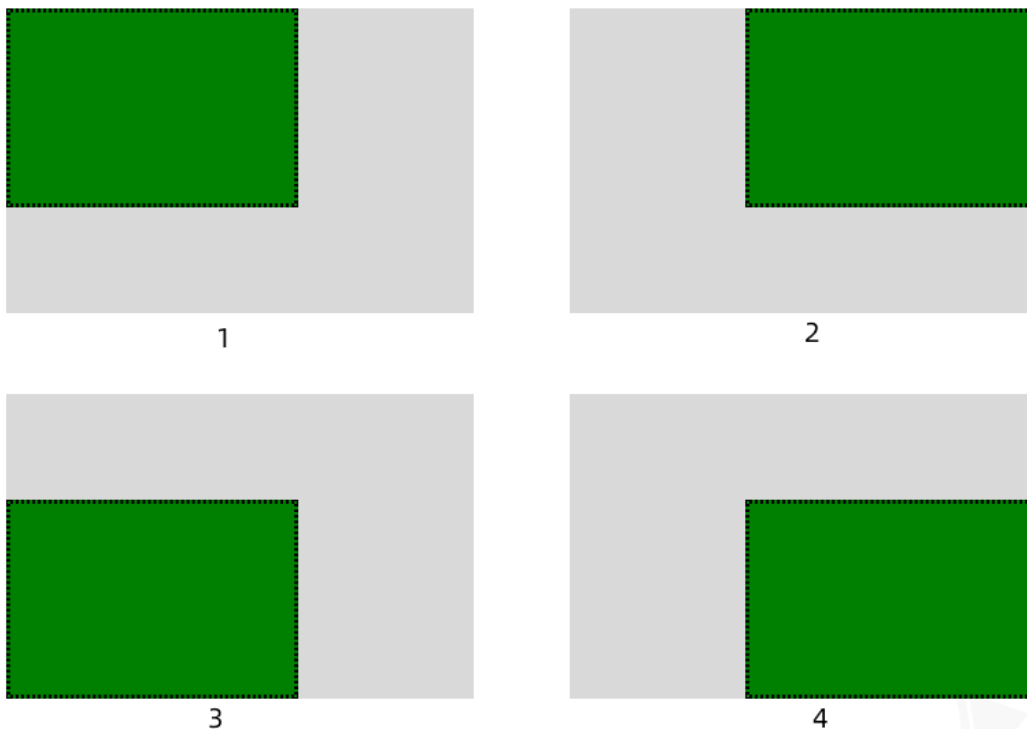
Center Prior is applicable to mixed case depalletizing to avoid collisions. If the cartons do not need to be stacked on a pallet after being picked (e.g., just need to be placed on a conveyor belt), this offset strategy is not necessary.

Sort by Distance to Reference Point - Reference Point X/Y

Instruction: You need to set a reference point if you want to enable this parameter, and the robot will prioritize picking poses of which the suction cup is close to the reference point. This setting and the object symmetry settings cannot be enabled at the same time.

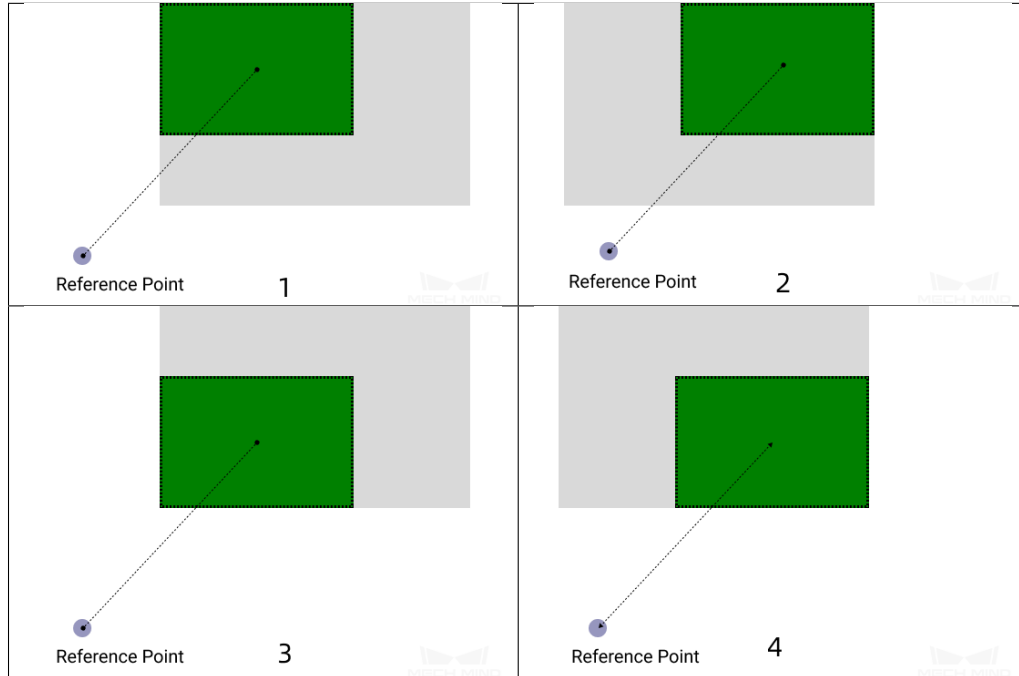
Example:

As shown in the figure below (top view perspective), the green rectangles represent cartons, and the gray rectangles represent suction cups. There are four picking strategies in total.



The reference point is in the lower left corner to the carton, and the relative position of the reference point and the carton is fixed. Picking poses of which the suction cup is nearest to the reference point will be tried first.

Therefore, the picking poses in the figure below will be tried in the order of 2 → 1 → 4 → 3.



Keep Only High Priority Offset—Max Plan Results Count

Instruction: The default setting is 2. Please set the count according to the total planning results when a reference point is introduced.

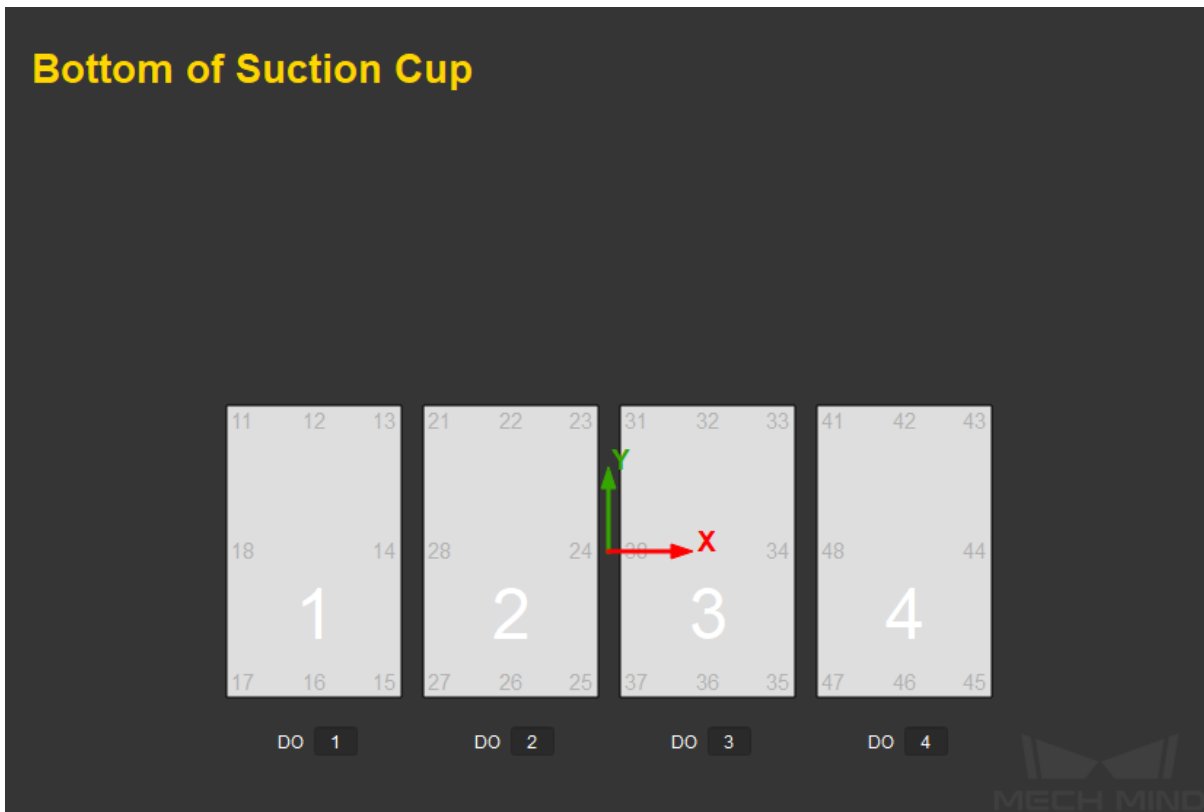
Example: If this option is enabled, only planning results No. 2 and No. 1 in the above example will be kept.

Fixed Edge Corner Offset—Fixed Edge Corner Offset

Enter the edge corner label numbers.

The numbering rules of the edge/corner: The first digit of the label number indicates the number of the suction cup, and the second digit of the label number indicates whether it is an edge or corner, where 1, 3, 5, and 7 represent the corner and 2, 4, 6, and 8 represent the edge. 0 represents the center of the suction cup.

Bottom of Suction Cup



Please refer to [Offset Strategy](#) for detailed information.

Carton Drop Detection

Application scenario:

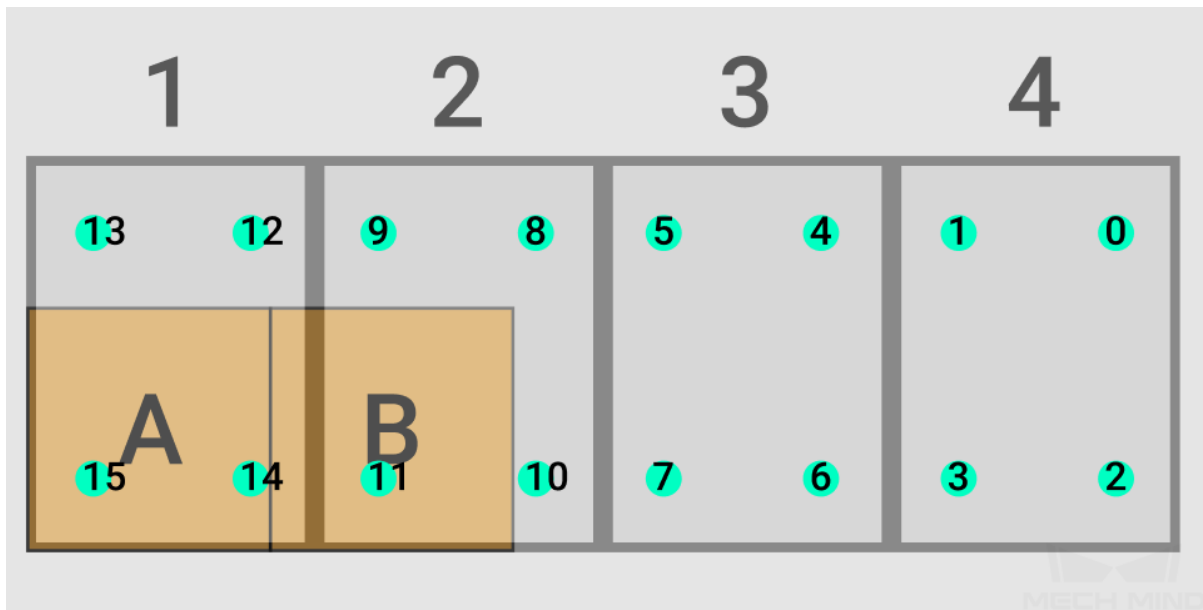
In actual carton depalletizing projects, DI check points will be added to the working surface of the suction cups to monitor whether the cartons are picked successfully and whether the cartons are dropped from the robot.

Prerequisites:

1. Select the suction cup configuration file in *Suction Cup Config* → *Single Rect Suction Cup Config Path*. Please refer to [Vacuum Gripper Configurator](#) for more information about obtaining a suction cup configuration file.
2. If *set_do_list* is used in the project, select the **Get Do List From VisualMove** parameter, and select the corresponding **visual_move**.
3. If *check_di_list* is used in the project, select the **Get DI List From VisualMove** parameter, and select the corresponding **visual_move**.

Example:

As shown in the figure below, the suction cup has 4 sections, and each section contains 4 DI check points. When the robot picks the combination of cartons A and B, the software will automatically determine which sections of the suction cup, that is, sections 1 and 2 in this example, should be started. Also, check points 8, 9, 10, 11, 12, 13, 14, and 15 will be used in the carton drop detection.



Remove Edge DI When DI check points are used in the carton drop detection and DI check points are at the edge of the carton, the software may mistakenly detect that the carton is dropped due to picking deviation, loose suction on the edge, deformation of the carton, or other reasons. Under such circumstances, you can set a **Dist from Carton Edge To Remove DI** to remove DI check points on the edge of the carton to avoid detection errors.

Dist from Carton Edge To Remove DI The distance inward from the carton edge, within which the DI check points will not trigger the carton drop detection. The part in which the DI will be removed is displayed in red in the 3D simulation area.

Picking Count

This parameter group is used to count the picked objects and calculate the rest objects to be picked. Once you enter an **Expected Picking Count**, the **Total Picked Object Count** and **Current Picked Object Count** will be calculated automatically.

Expected Picking Count The maximum number of picked objects.

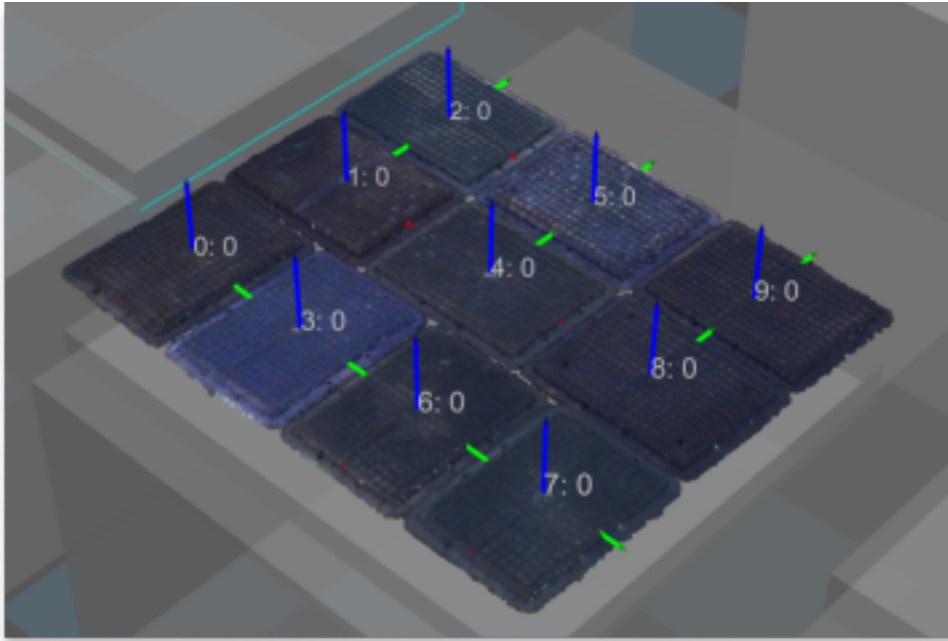
Total Picked Object Count The number of picked objects, which is counted automatically and cannot be modified.

Current Picked Object Count The number of the currently picked object, which is counted automatically and cannot be modified.

Picking Entire Row

Description: Once this parameter is enabled, the software will check whether there is another carton at the end of the row along the carton combination direction. If there is no other carton, the current carton combination will be kept, or else the carton combination will be discarded. For better performance, this parameter can be used with **Carton Size Scale Ratio for Combination** in **Carton Combination**.

Example: The crates are combined along the Y-axis (in green) as shown below. The possible combinations include [0,3,6,7], [1,2], [4,5], and [8,9]. Since only [0,3,6,7] includes the entire row and there are no other cartons in the Y-axis direction, the carton combination [0,3,6,7] will be picked first.



Pick Certain Count Per Time This parameter specifies the number of objects picked at one time. Once this parameter is set, only combinations with the specified number of objects will be picked. If no combination meets the requirement, no object will be picked and the Task will take the **Other failures** exit port.

Picking Count Per Time The number of objects to be picked at one time. This value must be an integer starting from 1.

Pick Multi Carton + Parallel Suction Cups

The parameters are shown below.

▼ Operational Mode

Tool Type Depallet Vacuum Gripper

▼ Suction Cup Config

Picking Setting Pick Multi Carton

Multi Pick Suction Cup Type Parallel Suction Cups

Parallel Suction Cup Config Path [Folder Icon] [Refresh Icon]

▼ Carton Combination

Carton Combination Strategy Free

Dist Thre for Carton Combination 0.000000 m

Angle Thre for Carton Combination 0.00000 rad

Carton Size Scale Ratio for Combination 90.00 %

Calc All Combination Plan

▼ Offset Setting

Sort by Distance to Reference Point

Fixed Edge Corner Offset

▼ Carton Drop Detection

Remove Edge DI

▼ Picking Count

Pick to Certain Count

Expected Picking Count 2147483647


Total Picked Object Count 0

Current Picked Object Count 0

Picking Entire Row

Pick Certain Count Per Time

Suction Cup Config

Parallel Suction Cup Config Path Click  to select the suction cup configuration file. You can use the [Vacuum Gripper Configurator](#) to configure the suction cups and output a configuration file in JSON format.

Carton Combination

Calc All Combination Plan

Select to calculate all possible carton combinations when different **Carton Combination Strategies** are used.

Please refer to [Carton Combination](#) for detailed information.

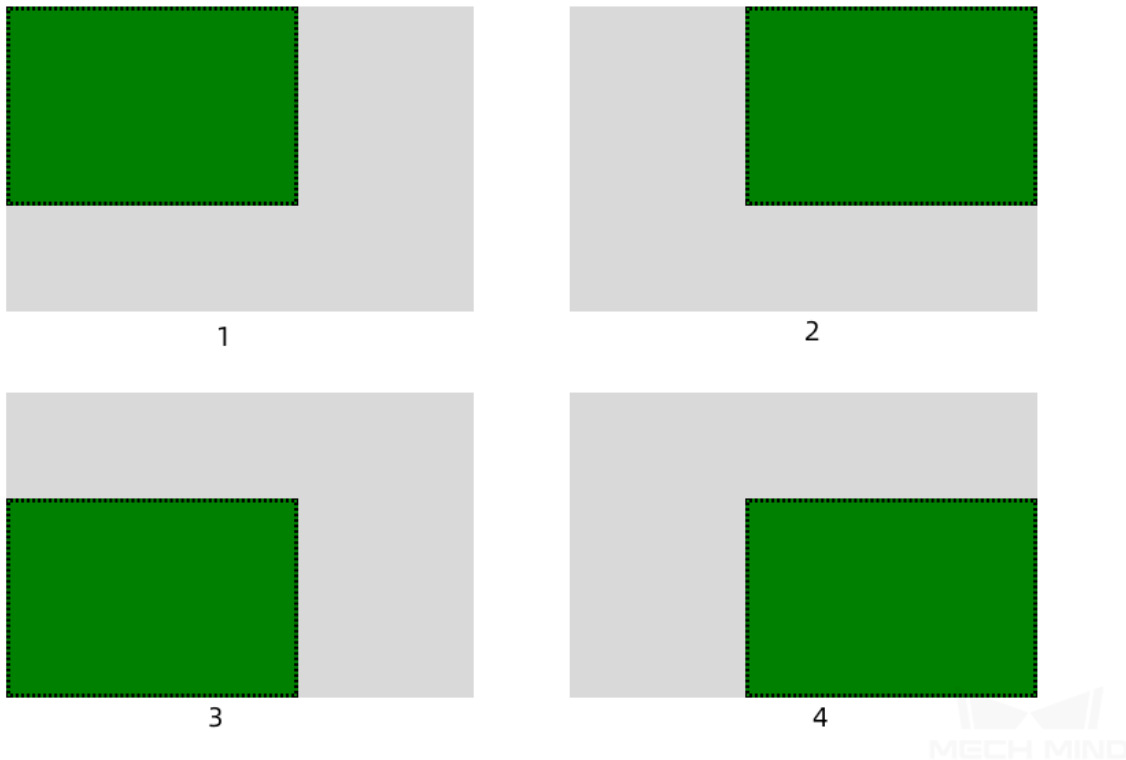
Offset Setting

Sort by Distance to Reference Point - Reference Point X/Y

Instruction: You need to set a reference point if you want to enable this parameter, and the robot will prioritize picking poses of which the suction cup is close to the reference point. This setting and the object symmetry settings cannot be enabled at the same time.

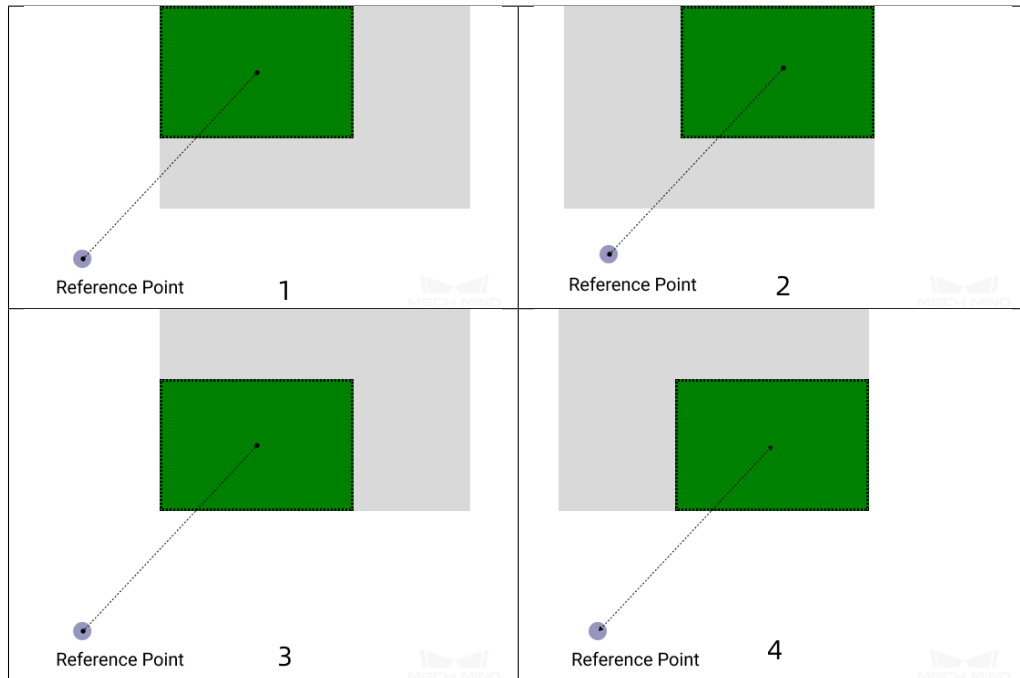
Example:

As shown in the figure (top view perspective) below, the green rectangles represent cartons, and the gray rectangles represent suction cups. There are four picking strategies in total.



The reference point is in the lower left corner to the carton, and the relative position of the reference point and the carton is fixed. Picking poses of which the suction cup is nearest to the reference point will be tried first.

Therefore, the picking poses in the figure below will be tried in the order of $2 \rightarrow 1 \rightarrow 4 \rightarrow 3$.



Keep Only High Priority Offset - Max Plan Results Count

Instruction: The default setting is 2. Please set the count according to the total planning results when a reference point is introduced.

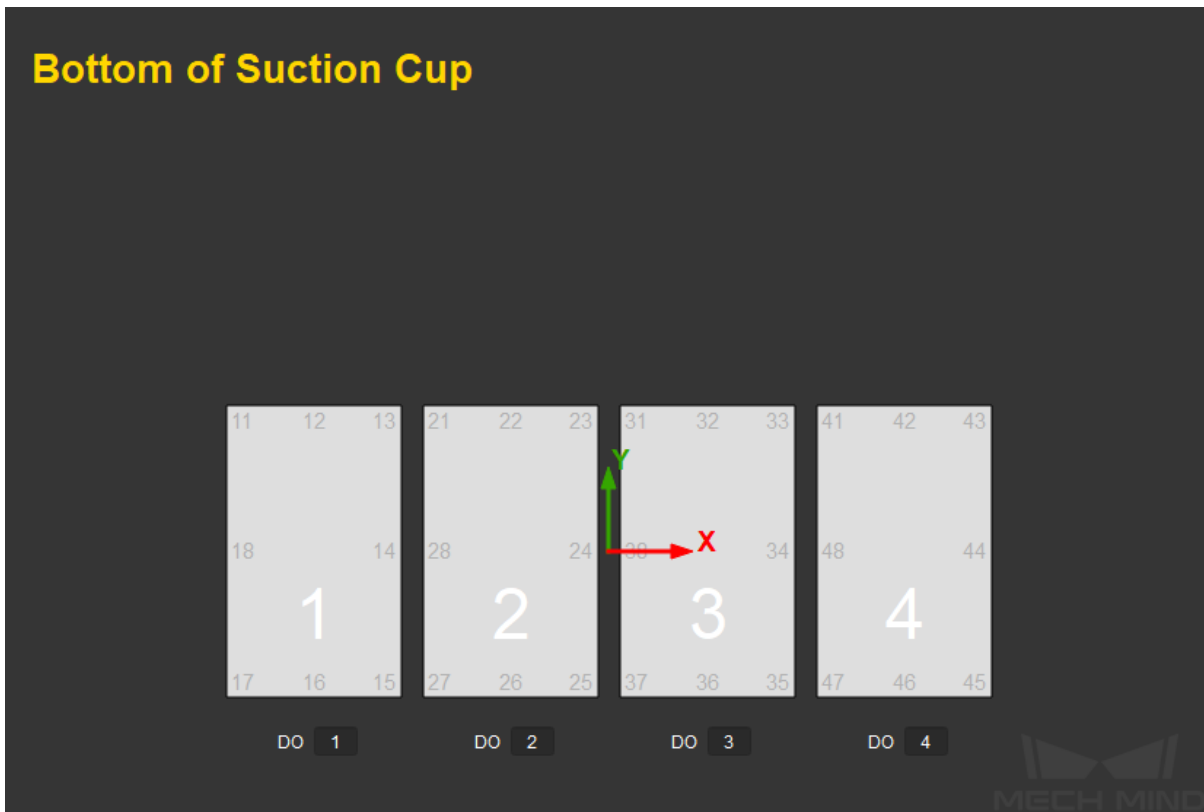
Example: If this option is enabled, only planning results No. 2 and No. 1 in the above example will be kept.

Fixed Edge Corner Offset - Fixed Edge Corner Offset

Enter the edge corner label numbers.

The numbering rules of the edge/corner: The first digit of the label number indicates the number of the suction cup, and the second digit of the label number indicates whether it is an edge or corner, where 1, 3, 5, and 7 represent the corner and 2, 4, 6, and 8 represent the edge. 0 represents the center of the suction cup.

Bottom of Suction Cup



Please refer to [Offset Strategy](#) for detailed information.

Carton Drop Detection

Application scenario:

In actual carton depalletizing projects, DI check points will be added on the working surface of the suction cups to monitor whether the cartons are picked successfully and whether the cartons are dropped by the robot.

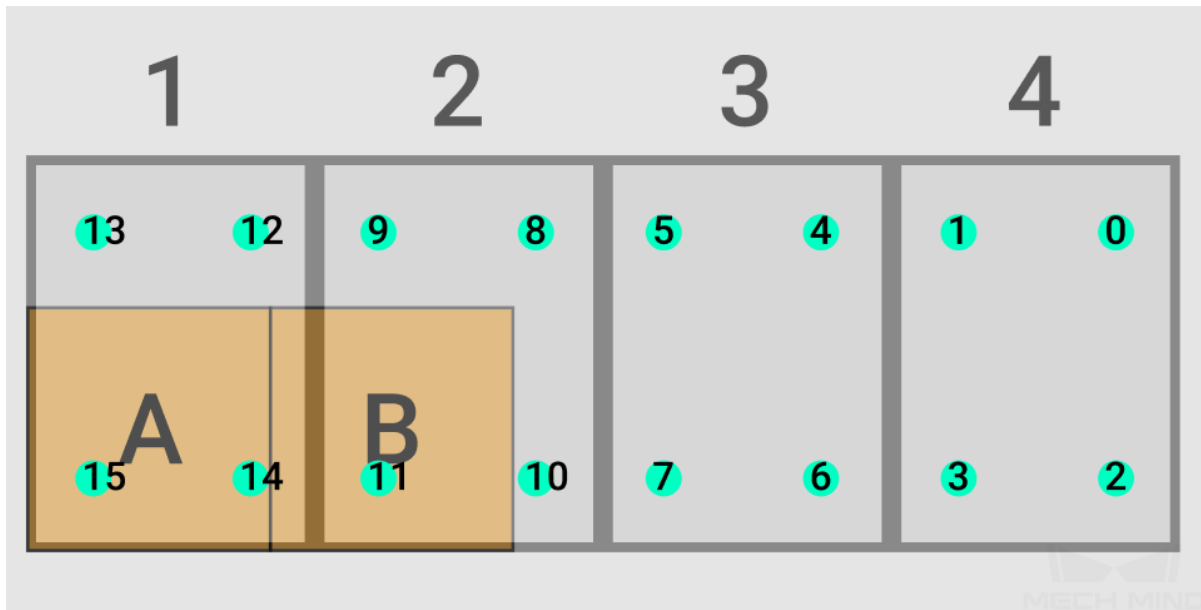
Prerequisites:

1. Select the suction cup configuration file in *Suction Cup Config* → *Parallel Suction Cup Config Path*. Please refer to [Vacuum Gripper Configurator](#) for more information about obtaining a suction cup configuration file.
2. If *set_do_list* is used in the project, select the **Get Do List From VisualMove** parameter, and select the corresponding **visual_move**.
3. If *check_di_list* is used in the project, select the **Get DI List From VisualMove** parameter, and select the corresponding **visual_move**.

Example:

As shown in the figure below, the suction cup has 4 sections, and each section contains 4 DI check points. When the robot picks the combination of cartons A and B, the software will automatically determine which sections of the suction cup, that is, sections 1 and 2 in this

example, should be started. Also, check points 8, 9, 10, 11, 12, 13, 14, and 15 will be used in the carton drop detection.



Remove Edge DI When DI check points are used in the carton drop detection and DI check points are at the edge of the carton, the software may mistakenly detect that the carton is dropped due to picking deviation, loose suction on the edge, deformation of the carton, or other reasons. Under such circumstances, you can set a **Dist from Carton Edge To Remove DI** to remove DI check points on the edge of the carton to avoid detection errors.

Dist from Carton Edge To Remove DI The distance inward from the carton edge, within which the DI check points will not trigger the carton drop detection. The part in which the DI will be removed is displayed in red in the 3D simulation area.

Picking Count

This parameter group is used to count the picked objects and calculate the rest objects to be picked. Once you enter an **Expected Picking Count**, the **Total Picked Object Count** and **Current Picked Object Count** will be calculated automatically.

Expected Picking Count The maximum number of picked objects.

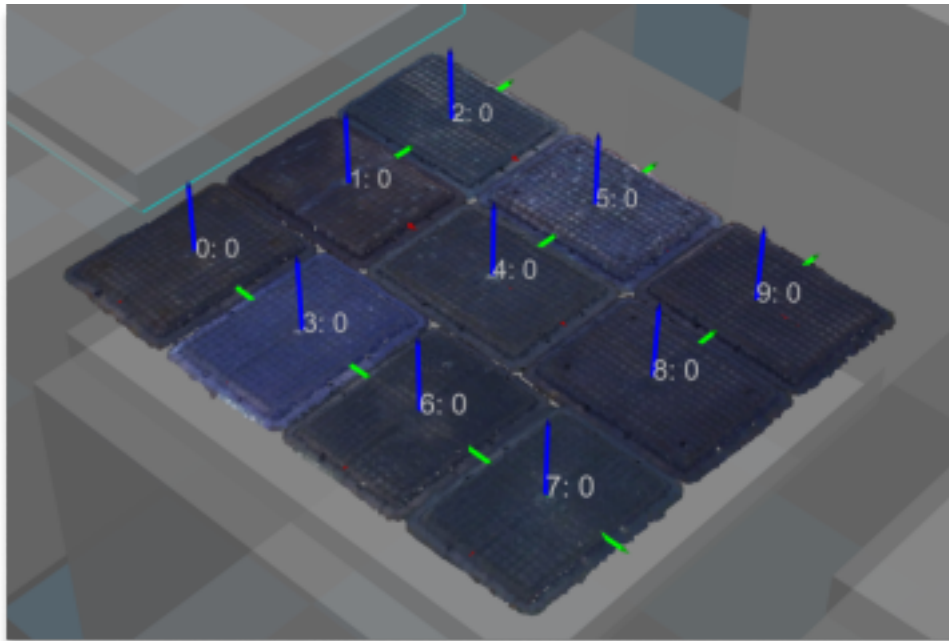
Total Picked Object Count The number of picked objects, which is counted automatically and cannot be modified.

Current Picked Object Count The number of the currently picked object, which is counted automatically and cannot be modified.

Picking Entire Row

Description: Once this parameter is enabled, the software will check whether there is another carton at the end of the row along the carton combination direction. If there is no other carton, the current carton combination will be kept, or else the carton combination will be discarded. For better performance, this parameter can be used with **Carton Size Scale Ratio for Combination** in **Carton Combination**.

Example: The crates are combined along the Y-axis (in green) as shown below. The possible combinations include [0,3,6,7], [1,2], [4,5], and [8,9]. Since only [0,3,6,7] includes the entire row and there are no other cartons in the Y-axis direction, the carton combination [0,3,6,7] will be picked first.



Pick Certain Count Per Time This parameter specifies the number of objects picked at one time. Once this parameter is set, only combinations with the specified number of objects will be picked. If no combination meets the requirement, no object will be picked and the Task will take the **Other failures** exit port.

Picking Count Per Time The number of objects to be picked at one time. This value must be an integer starting from 1.

Carton Combination

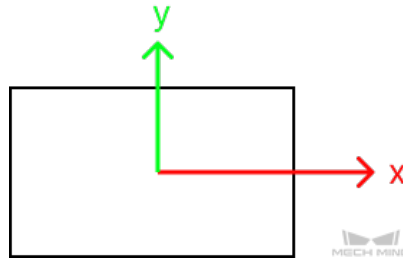
Carton combinations can be divided into two types, which are **combined according to the carton pose** and **combined according to the customized reference frame**.

Type	Carton Combination Strategy
Combined according to the carton pose	Free
	According to Obj Coord Sys
Combined according to the customized reference frame	According to Customized Coord Sys

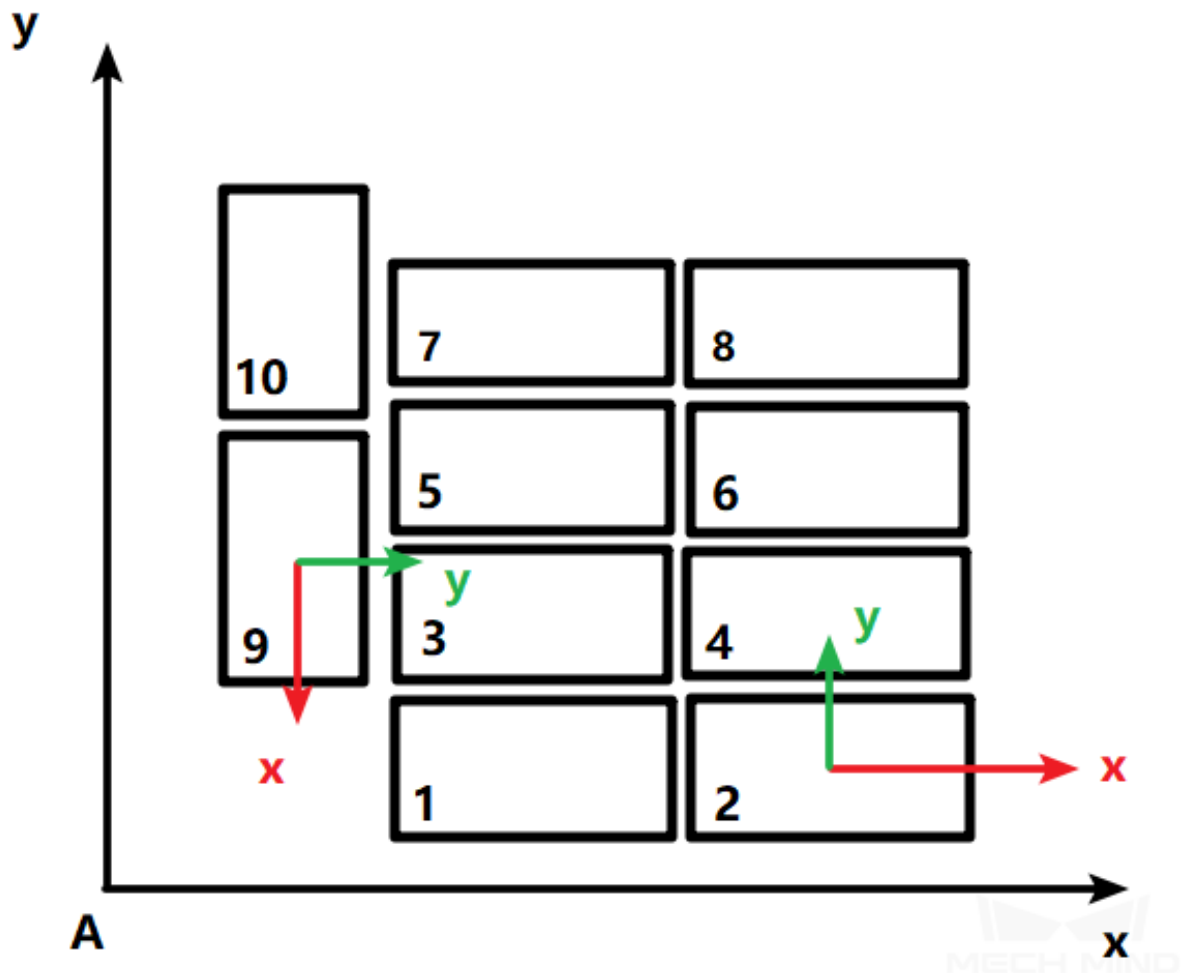
Combined according to the Carton Pose

Description of the Combination

By default, the X-axis of the carton pose is parallel with the long side of the carton, and the Y-axis is parallel with the short side, as shown below.



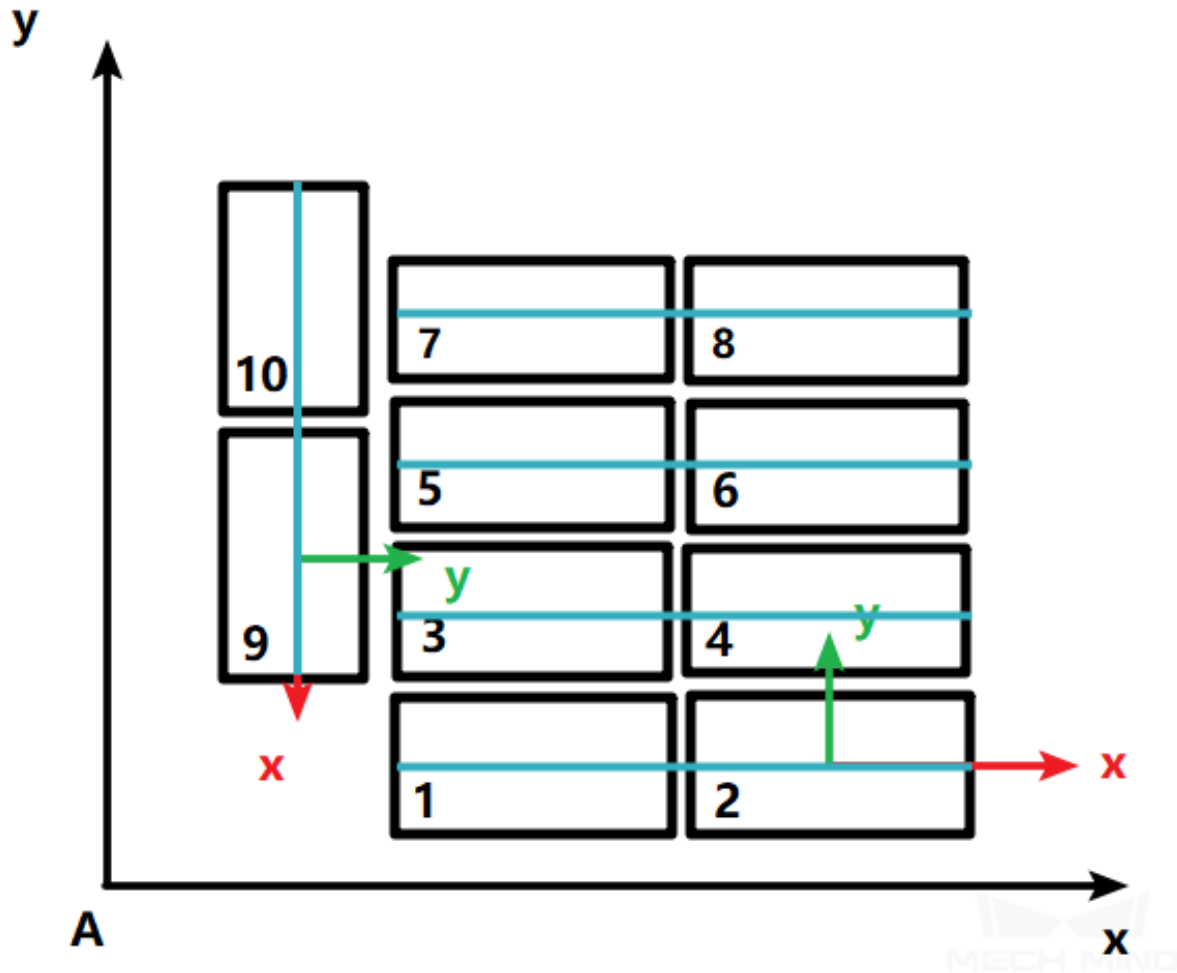
The pose combination can be divided into “free combination”, “combined along the X-axis of the carton”, and “combined along the Y-axis of the carton”. The figure below shows the top view of the cartons.



- Combined along the X-Axis of the Carton

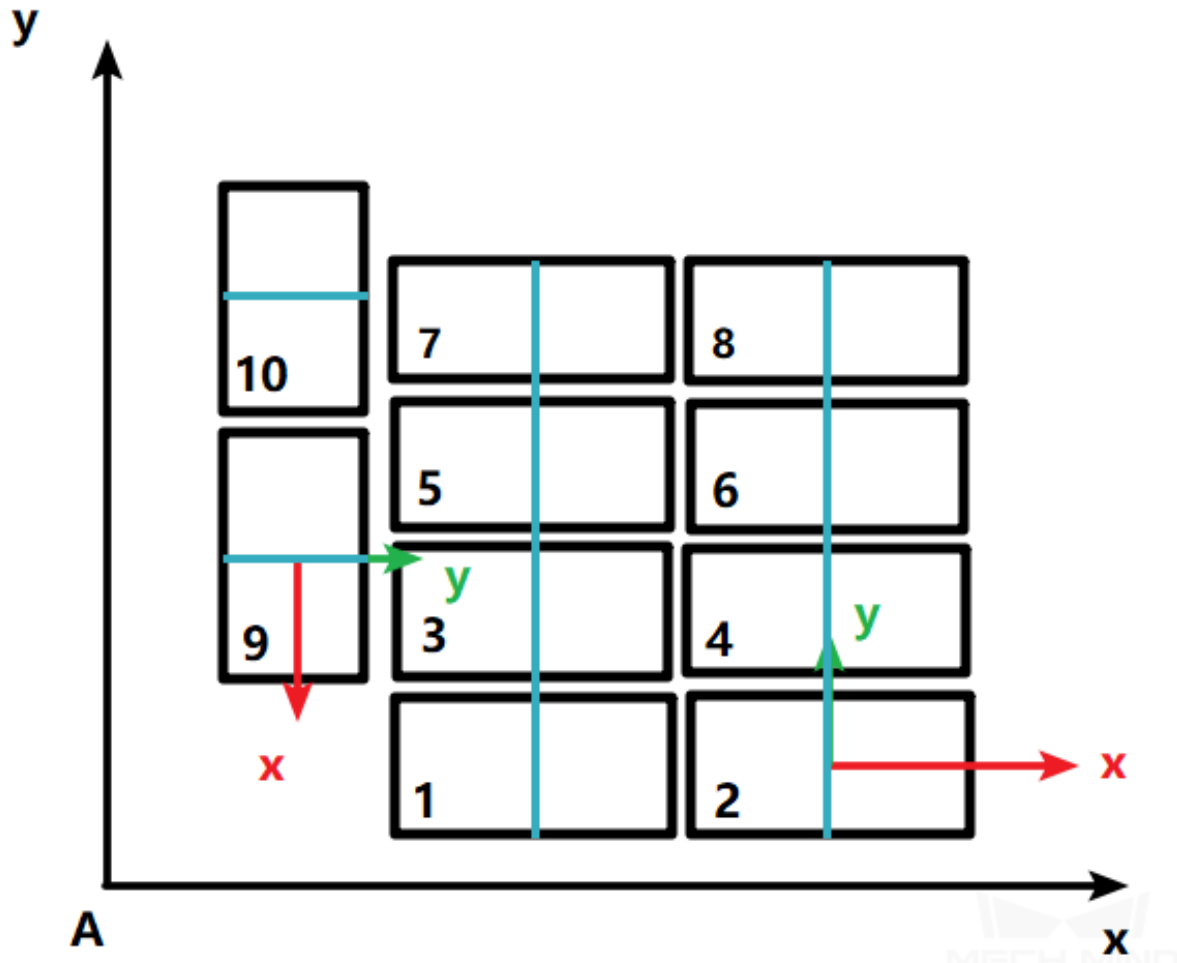
Carton Combination Strategy: According to Obj Coord Sys Obj Coord Sys Restriction: Along Axis X

There are 5 carton combinations as shown below, which are [1,2], [3,4], [5,6], [7,8], and [9,10].



- Combined along the Y-Axis of the Carton Carton Combination Strategy: According to Obj Coord Sys Obj Coord Sys Restriction: Along Axis Y

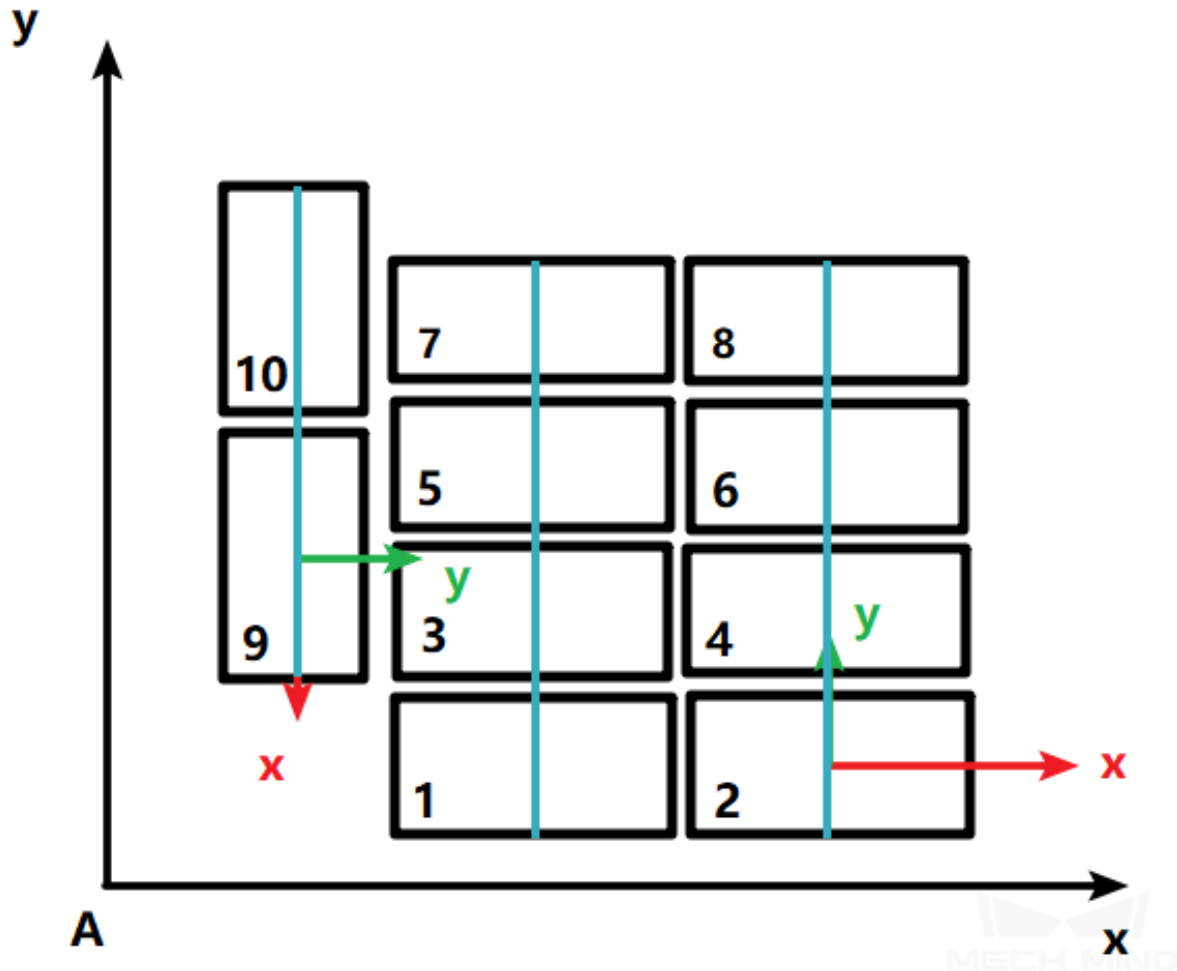
There are 4 carton combinations as shown below, which are [1,3,5,7], [2,4,6,8], [9], and [10].



• Free Combination Carton Combination Strategy: Free

The free carton combination strategy will combine as many cartons as possible in a combination automatically, and the combination may be made along the X-axis or Y-axis of the cartons.

There are 3 carton combinations as shown below, which are [1,3,5,7], [2,4,6,8], and [9,10].



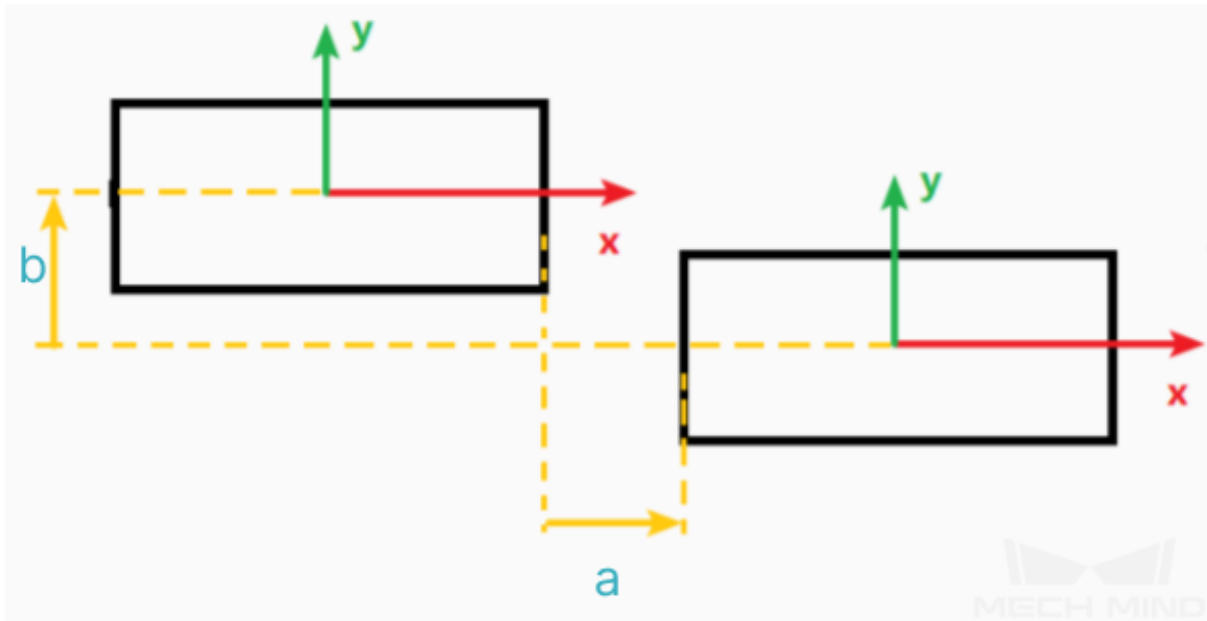
Parameter Descriptions

Dist Thre for Carton Combination: Select a carton, and use the direction of the X-axis or Y-axis in the carton reference frame as the combination direction to find other cartons that can be combined.

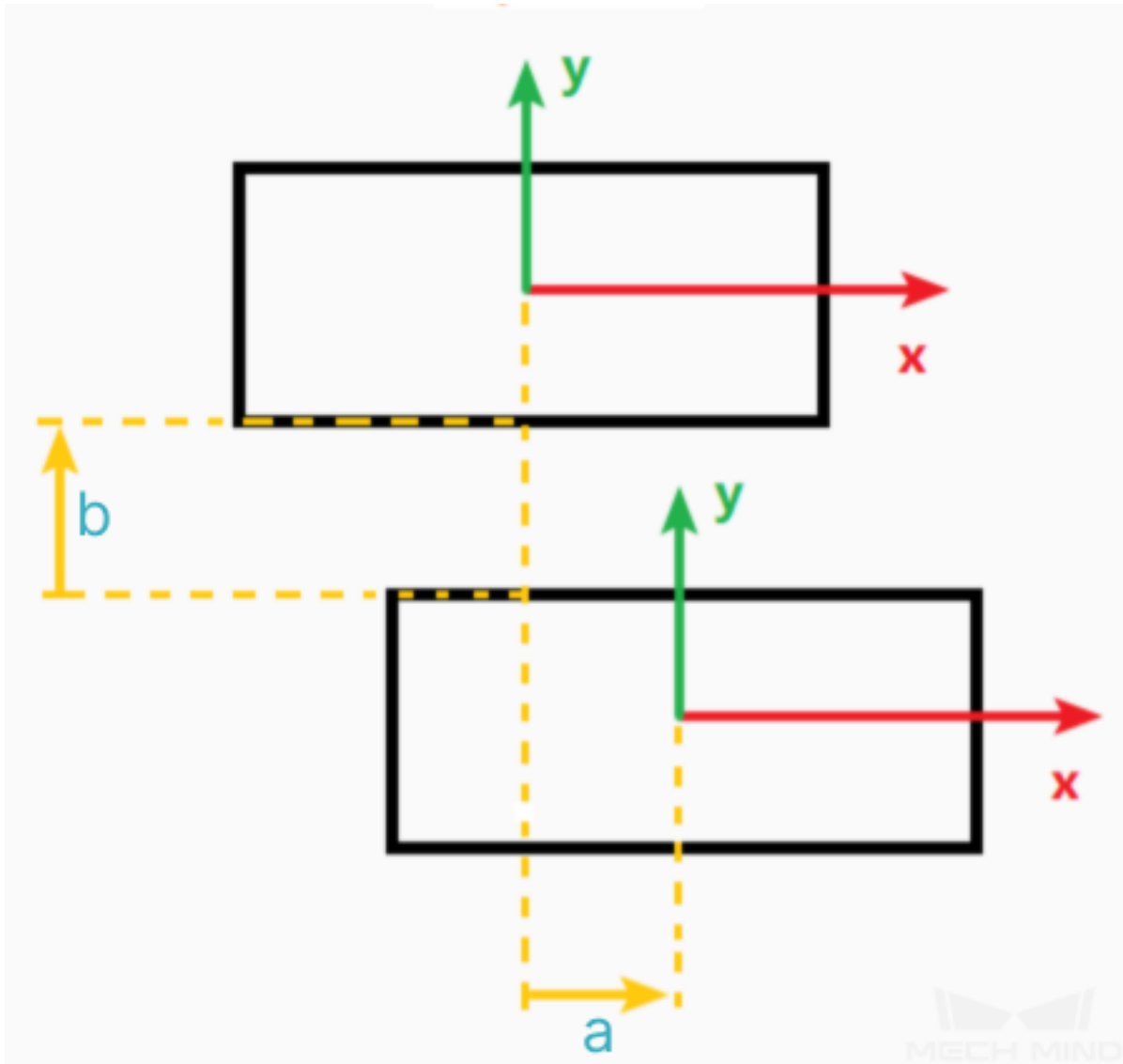
Assuming that the distance between two cartons along the X-axis is a, and the distance along the Y-axis is b, the combination distance is $\sqrt{a^2 + b^2}$.

Cartons with the combination distance that is within this threshold can be combined, or cannot be combined otherwise.

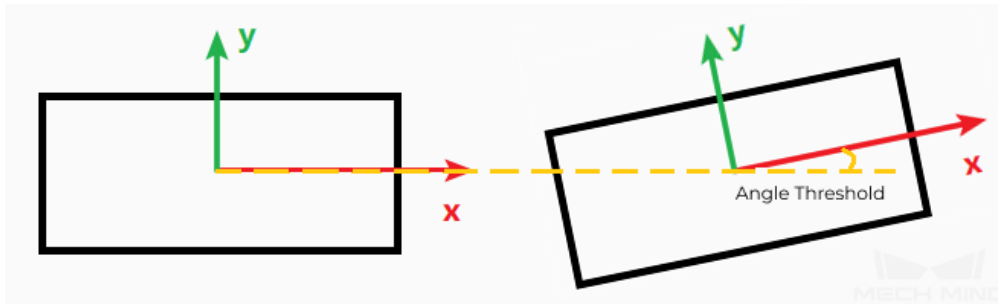
- Combine along the X-axis of the cartons:



- Combine along the Y-axis of the cartons:



Angle Thre for Carton Combination: The acceptable difference in the rotation angle of the combined cartons around the Z-axis. Cartons with angle differences smaller than this threshold can be combined, or cannot be combined otherwise. The figure below shows the top view of the cartons, and the angle threshold for carton combination is shown in the figure below.



Carton Size Scale Ratio for Combination: Usage scenario: When parallel suction cups are used to pick multiple cartons, the detected carton dimensions may not be accurate. If the detected carton is large or the cartons are overlapping, it will be hard to determine whether the suction cup covers the non-target cartons in offset picking. Under this circumstance, reducing the size of the detected boxes can avoid the impact of dimension errors.

Value range: 0–100%. The default value is 90%. It is not recommended to use a value less than 80%.

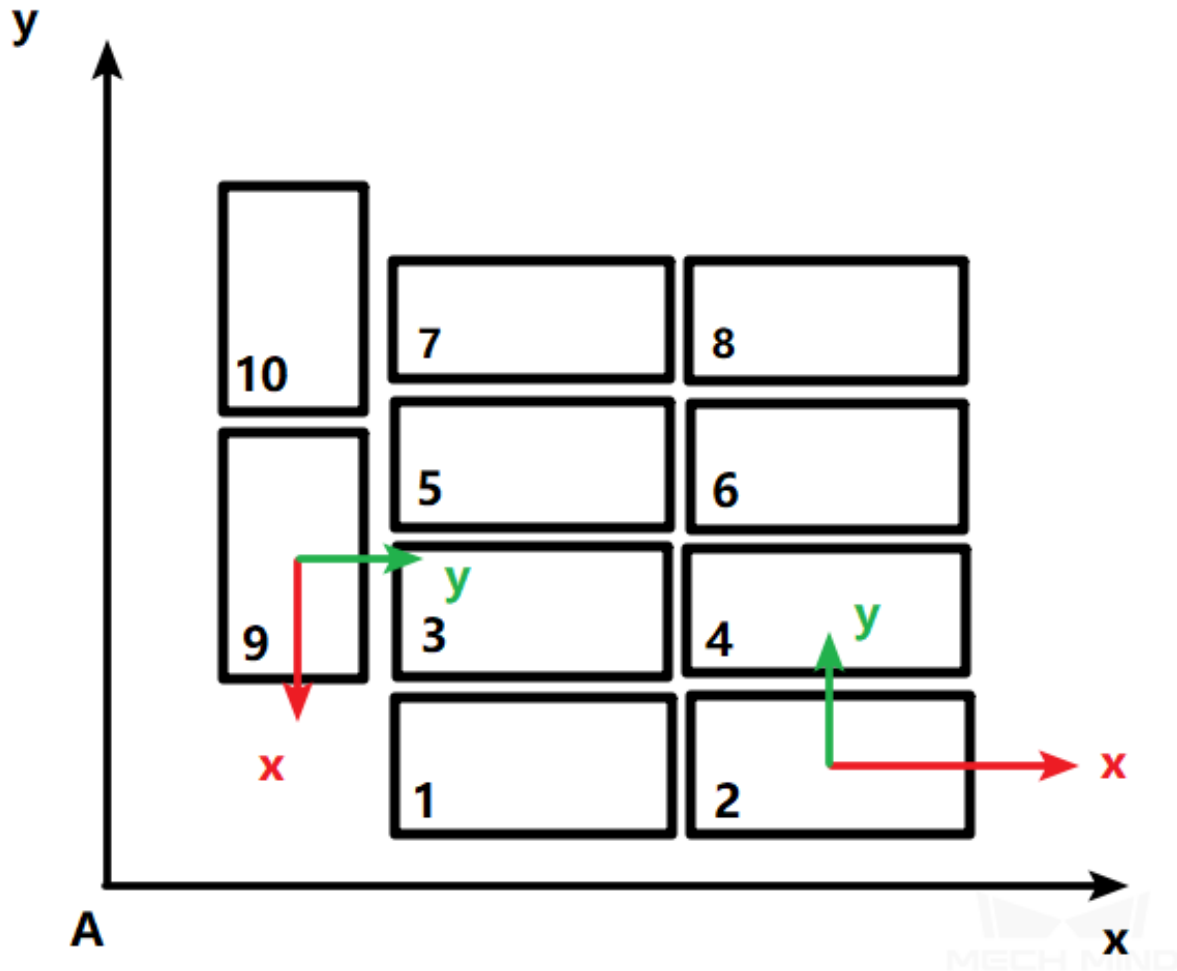
Hint: This parameter can be used with the **Picking Entire Row** in **Picking Count** to improve performance.

Combined according to the Customized Reference Frame

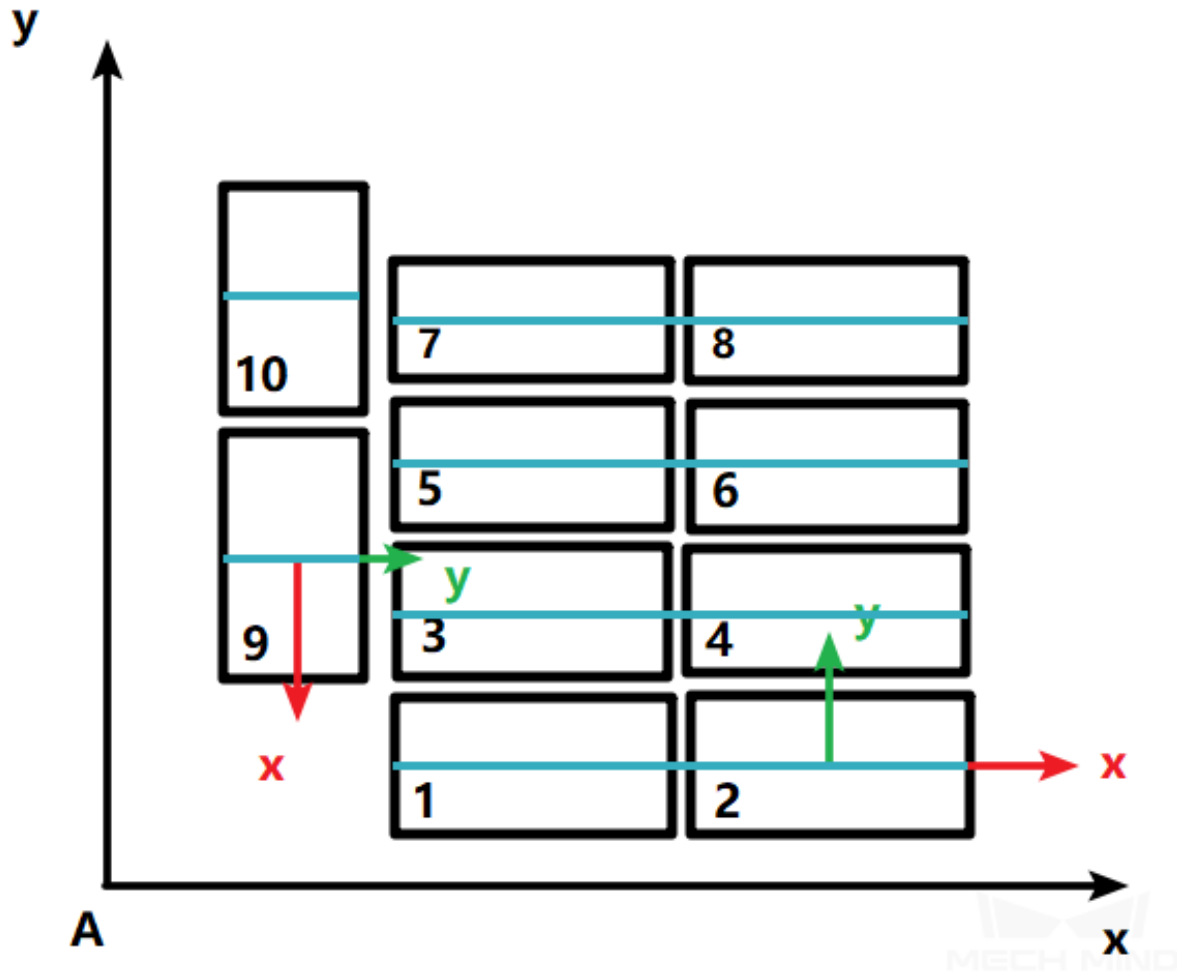
Description of the Combination

You can customize a reference frame in the 3D simulation area. The following examples are based on the reference frame A.

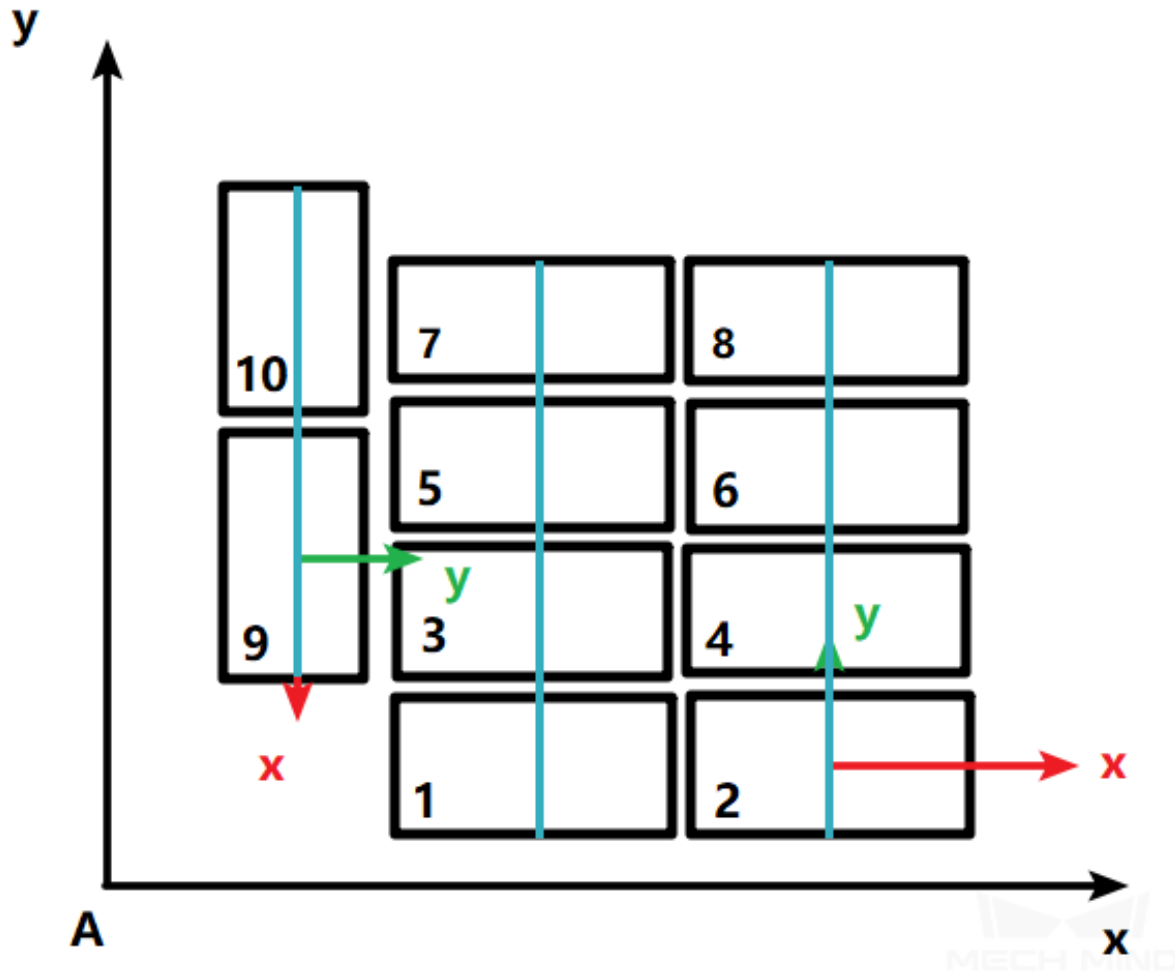
According To Customized Coord Sys can be divided into **Along Axis X** and **Along Axis Y**.



- Along Axis X: There are 6 carton combinations as shown below, which are [1,2], [3,4], [5,6], [7,8], [9], and [10].



- Along Axis Y: There are 3 carton combinations as shown below, which are [1,3,5,7], [2,4,6,8], and [9,10].



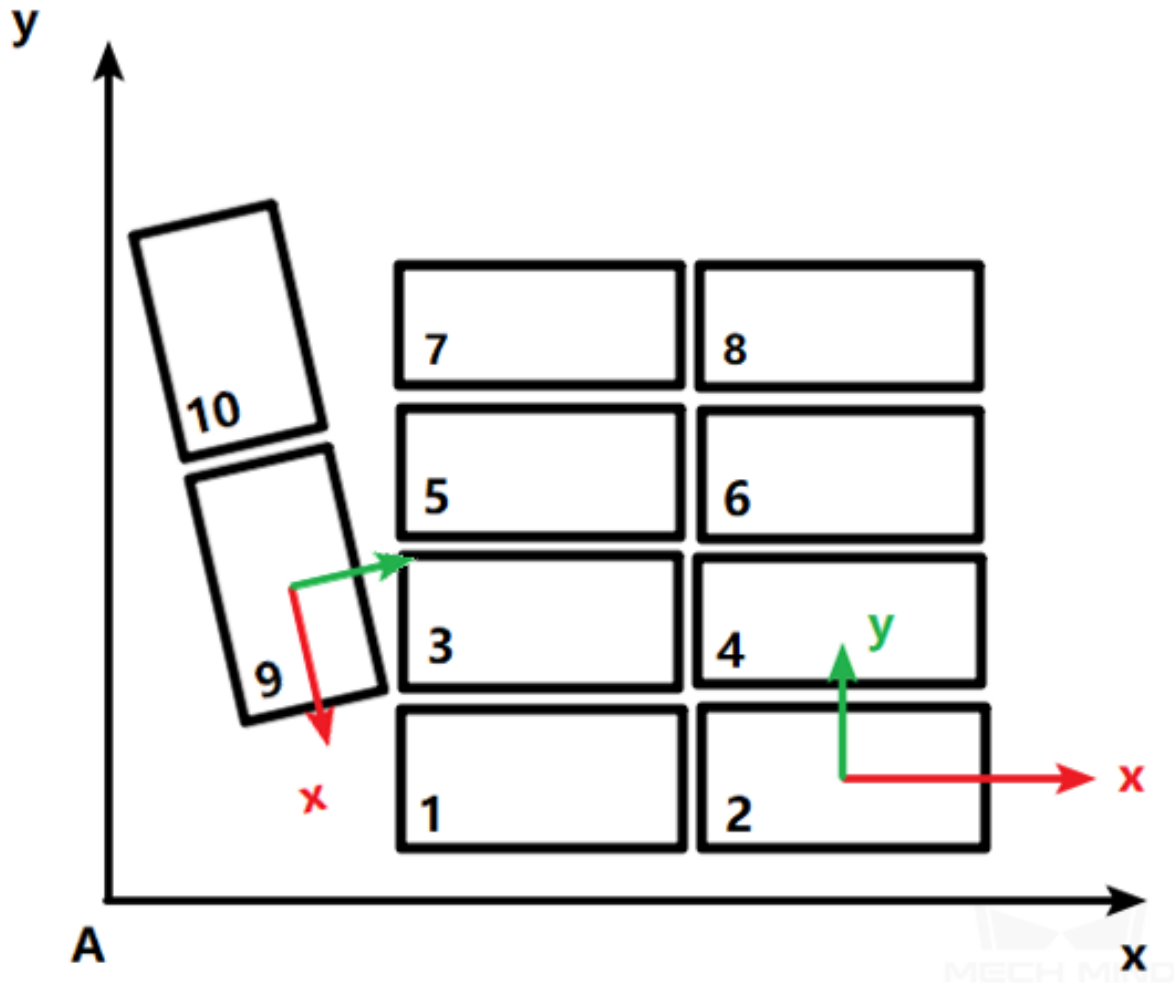
Carton Combination Logic

1. Determine whether the angle between the carton and the target axis of reference frame A is smaller than the **Angle Thre of Customized Coord Sys**, i.e., whether the carton is parallel or perpendicular to the combination direction.
2. Combine the selected cartons along the carton axis.

The following exmple shows how to combine the cartons **along axis X in the customized reference frame A**.

1. The software will calculate the angles between the X-axes of each carton and the reference frame, and then calculate the angles between the Y-axes of each carton and the reference frame. The relatively small angles will be compared with the **Angle Thre of Customized Coord Sys**, if the angle is below the threshold, the corresponding carton can be combined, or else it cannot be combined.

As shown in the figure below, the angles of carton 9 and 10 are beyond the threshold, and the angles between the X-axes of other cartons and the reference frame A are within the threshold.



2. Combine the cartons which are filtered by the angle threshold along the axis of its own reference frame. The combination logic is restricted by Dist Thre for Carton Combination and Angle Thre for Carton Combination.

Parameter Descriptions

▼ Carton Combination	
Carton Combination Strategy	According To Customized Coord Sys ▼
Customized Coord Sys Restriction	Along Axis Y ▼
Customized Coord Sys Pos X	0.000000 m ▲▼
Customized Coord Sys Pos Y	0.000000 m ▲▼
Customized Coord Sys Pos Z	0.000000 m ▲▼
Customized Coord Sys Rot Around Z	0.000000 rad ▲▼
Angle Thre of Customized Coord Sys	0.000000 rad ▲▼
Dist Thre for Carton Combination	0.000000 m ▲▼
Angle Thre for Carton Combination	0.000000 rad ▲▼
Carton Size Scale Ratio for Combination	90.00 % ▲▼

Customized Coord Sys Pos X/Y/Z The customized X/Y/Z coordinates (relative to the world frame) of the origin in the customized reference frame

Customized Coord Sys Rot Around Z The rotation angle of the customized reference frame around Z-axis (relative to the world frame)

Angle Thre of Customized Coord Sys The maximum angle between the X/Y-axis of the carton and the target axis of the customized reference frame. If the angle is less than the threshold, the carton can be combined. Otherwise, the cartons cannot be combined.

Dist Thre for Carton Combination See above.

Angle Thre for Carton Combination See above.

Carton Size Scale Ratio for Combination See above.

Offset Strategy

This section covers the following topics:

- *Strategies for Single Rectangle Suction Cup*
 - *Parallel With Carton Group Long Side*
 - *Parallel With Combination Direction*
- *Strategies for Parallel Suction Cups*

Strategies for Single Rectangle Suction Cup

The calculation of offset setting for single rectangle suction cups is based on the combination of point cloud collision detection and the removal of the picked object's point cloud, and it requires the point cloud data and carton dimensions in the vision result.

You can select from various offset strategies according to the actual requirement of picking.

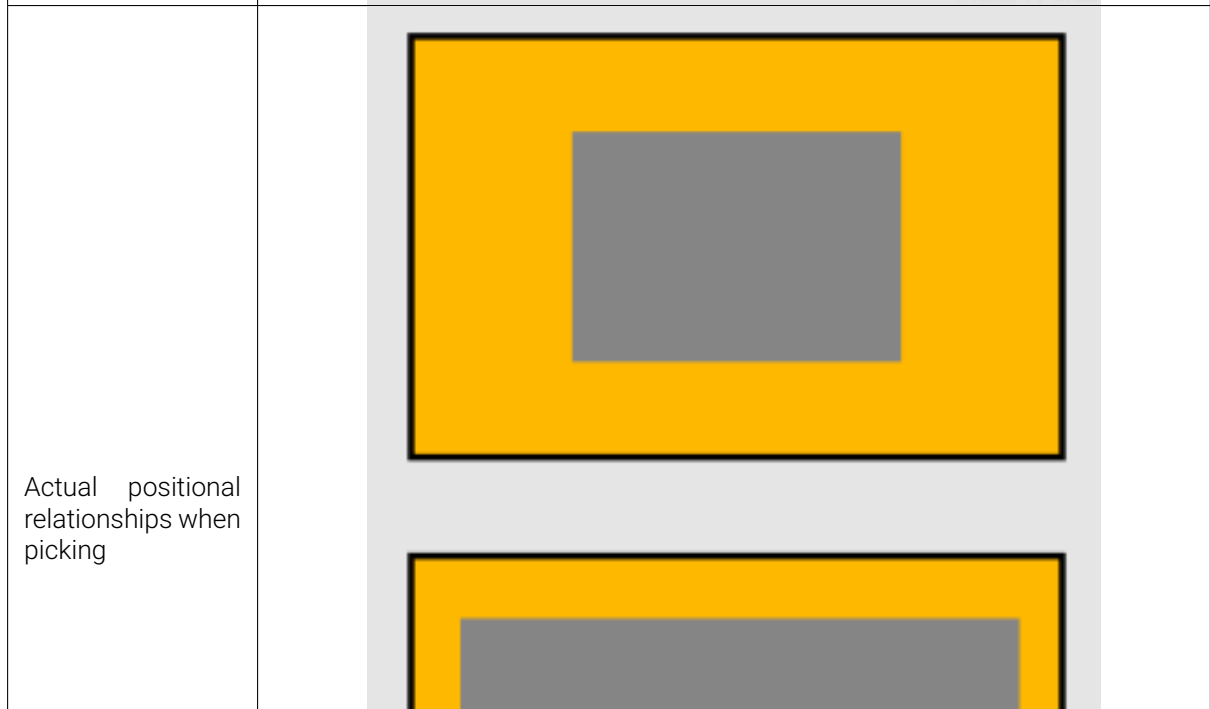
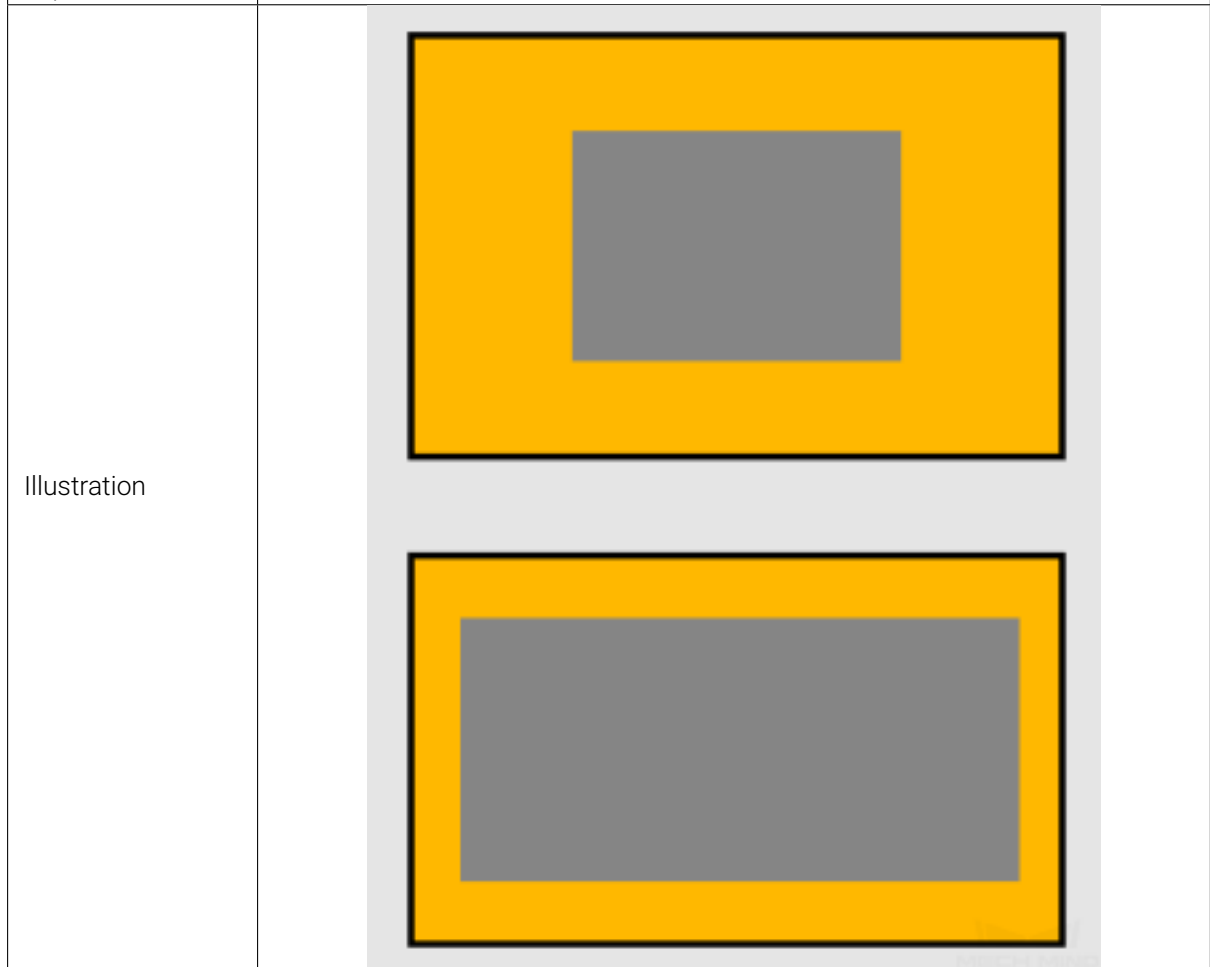
According to the positional relationship between the suction cup and the carton group	Center Prior
	Corner Prior
	Corner Alignment Only
According to the directional relationship between the suction cup and the carton group	Parallel With Carton Group Long Side
	Parallel With Combination Direction

Parallel With Carton Group Long Side

Note: In the illustrations below, the yellow rectangle represents the carton, and the gray rectangle represents the suction cup.

Scenario 1:

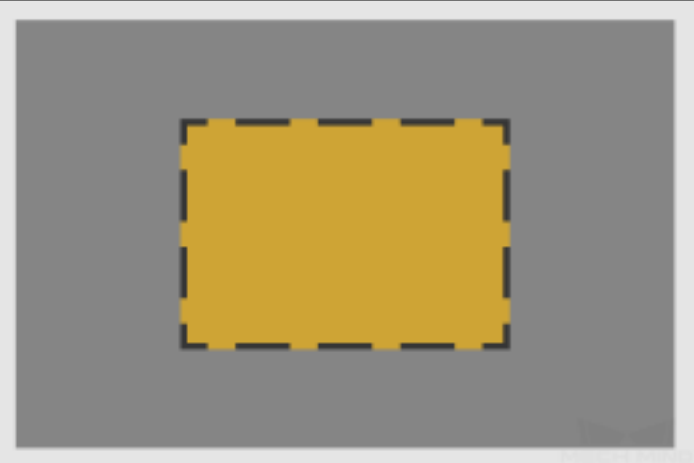
Size relationship between the carton and suction cup
 Long side of the carton > Short side of the carton > Long side of the suction cup > Short side of the suction cup or Long side of the carton > Long side of the suction cup > Short side of the carton > Short side of the suction cup



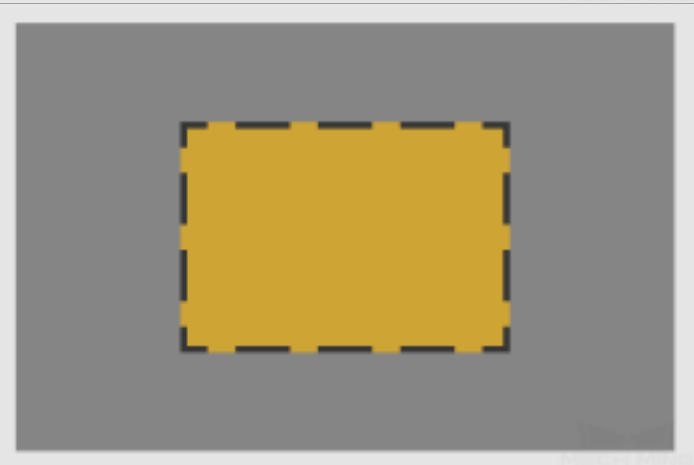
Scenario 2:

Size relationship between the carton and suction cup Long side of the suction cup > Short side of the suction cup > Long side of the carton > Short side of the carton

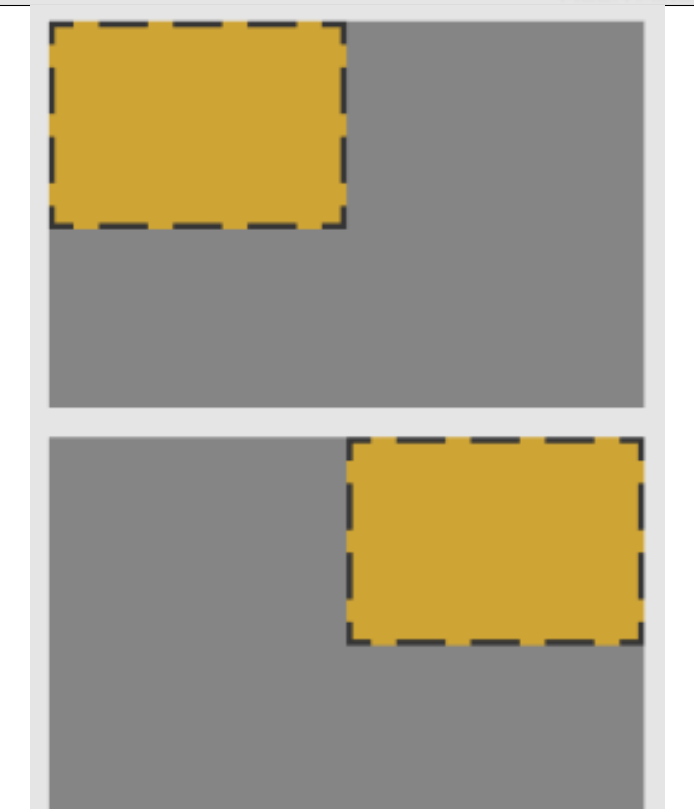
Illustration



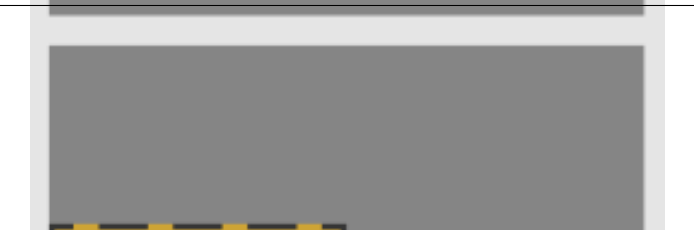
Suction cup position 1



Suction cup position 2



312 Vision



Scenario 3:

Size relationship between the carton and suction cup
 Long side of the suction cup > Long side of the carton > Short side of the suction cup > Short side of the carton

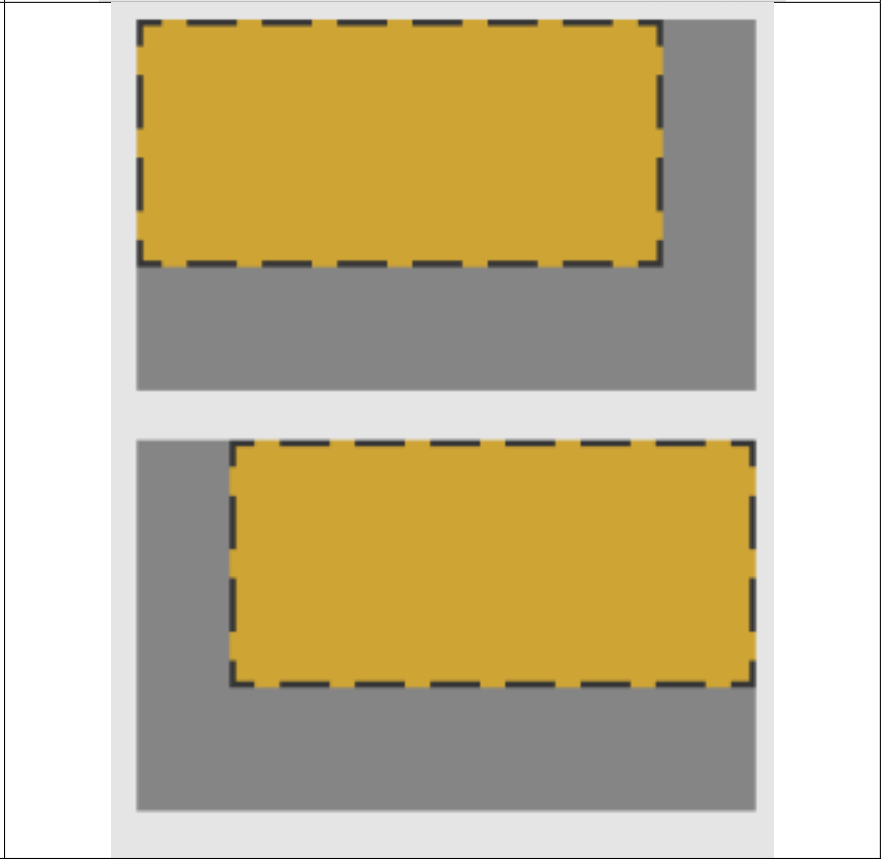
Illustration



Suction cup position 1





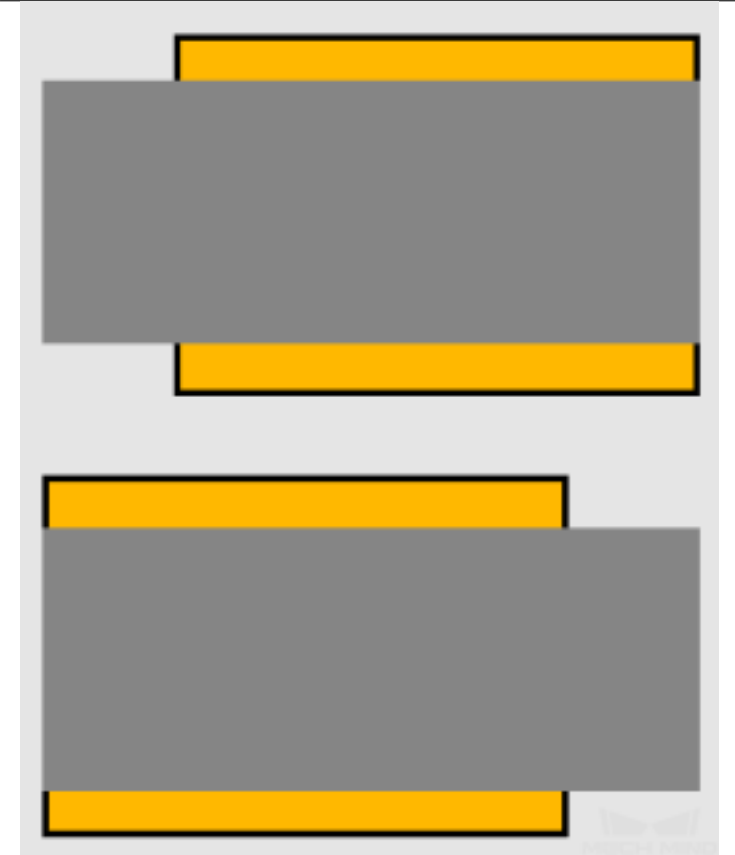
Suction cup position 2





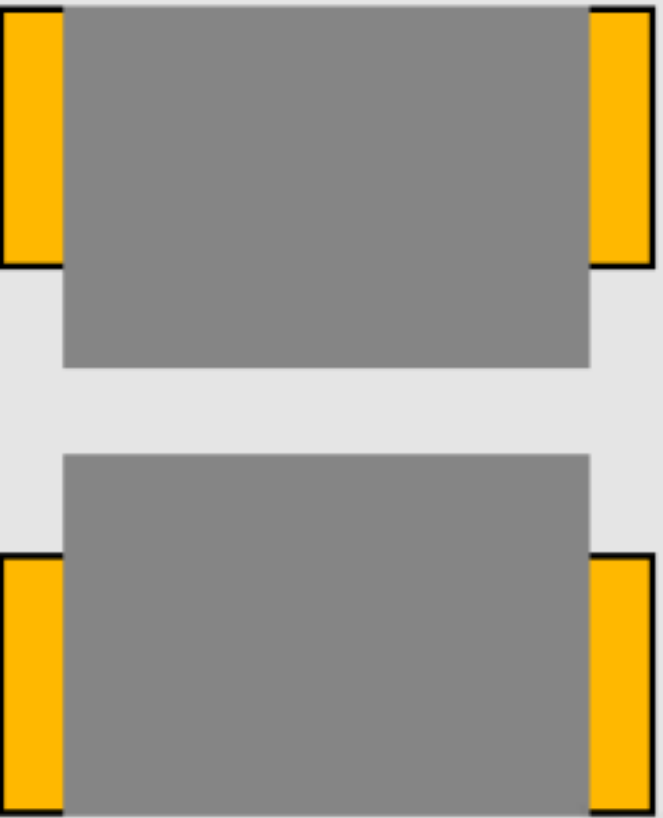
3.12. Vision



Scenario 4:

Size relationship between the carton and suction cup	Long side of the suction cup > Long side of the carton > Short side of the carton > Short side of the suction cup
Illustration	
Suction cup position 1	
Suction cup position 2	
Offset Strategy	<p>If Center Prior is selected, position 1 instead of position 2 will be used for picking first.</p>
3.12. Vision	<p>If Corner Prior is selected, position 2 instead of position 1 will be used for picking first.</p>



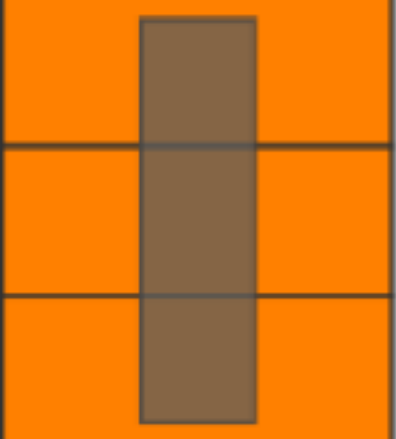
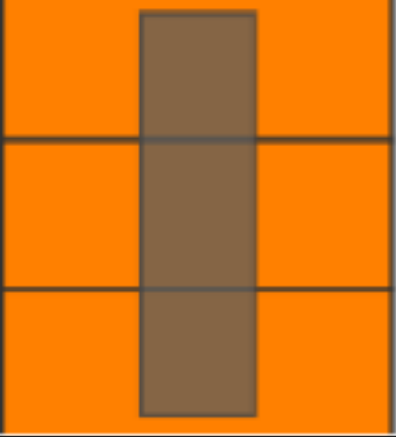
Scenario 5:

<p>Size relationship between the carton and suction cup</p>	<p>Long side of the carton > Long side of the suction cup > Short side of the suction cup > Short side of the carton</p>
<p>Illustration</p>	
<p>Suction cup position 1</p>	
<p>Suction cup position 2</p>	
<p>Offset Strategy</p>	<p>If Center Prior is selected, position 1 instead of position 2 will be used for picking first.</p>
<p>3.12. Vision</p>	<p>If Corner Prior is selected, position 2 instead of position 1 will be used for picking first.</p>



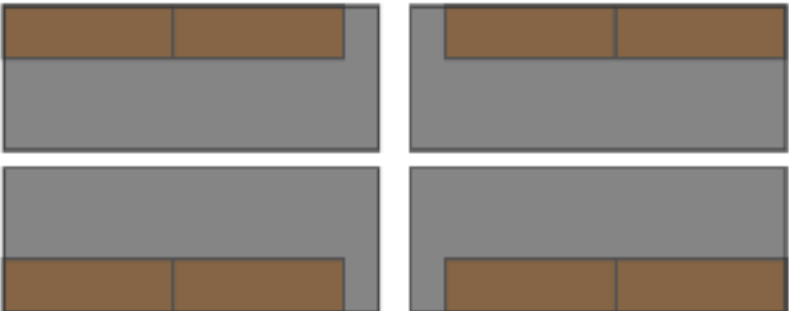


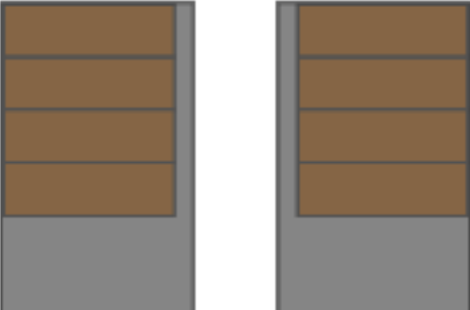

Parallel With Combination Direction

Note: In the illustrations below, the orange rectangle represents the carton, and the gray rectangle represents the suction cup.

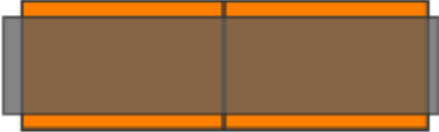

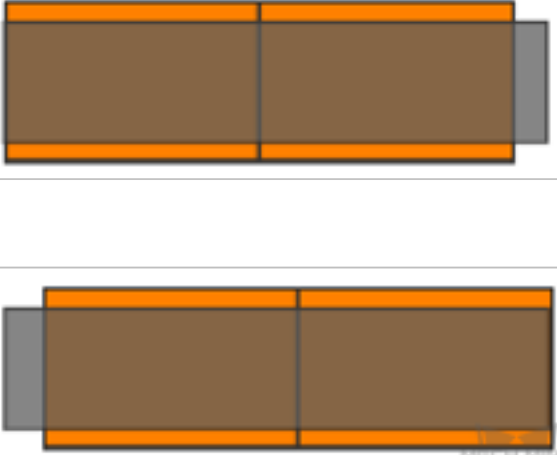
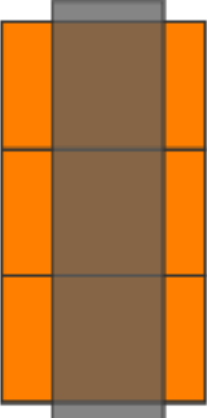
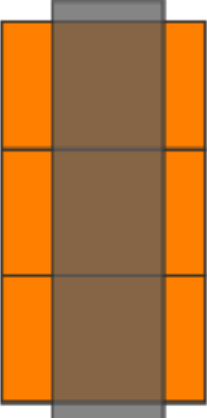
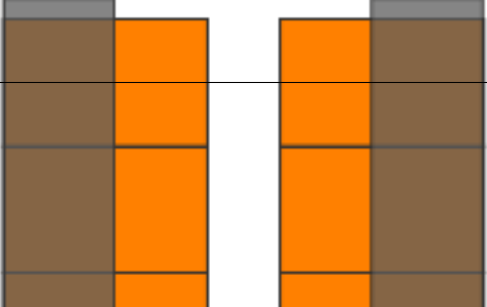
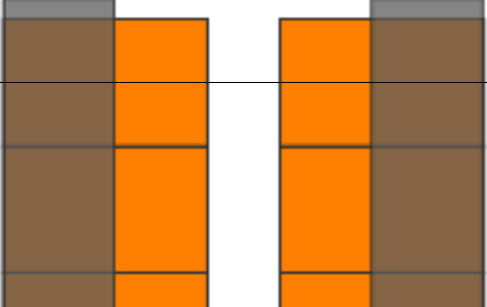
Scenario 1:

Size relationship between the carton and suction cup		Long side of the carton > Short side of the carton > Long side of the suction cup > Short side of the suction cup or Long side of the carton > Long side of the suction cup > Short side of the carton > Short side of the suction cup	
Combine suction cups along the X direction of the carton	Illustration		
	Suction cup position		
Combine suction cups along the Y direction of the carton	Illustration		
	Suction cup position		
Offset Strategy		Any strategy is applicable.	



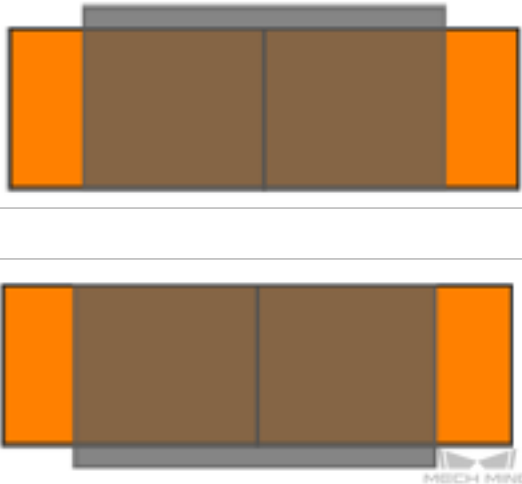

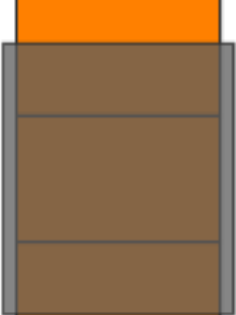


Scenario 2:

Size relationship between the carton and suction cup		Long side of the suction cup > Short side of the suction cup > Long side of the carton > Short side of the carton or Long side of the suction cup > Long side of the carton > Short side of the suction cup > Short side of the carton	
Combine suction cups along the X direction of the carton	Illustration		
	Suction cup position 1		
	Suction cup position 2		
Combine suction cups along the Y direction of the carton	Illustration		
	Suction cup position 1		
	Suction cup position 2		
3.12. Vision	Suction cup position 2		

Scenario 3:

Size relationship between the carton and suction cup	Long side of the suction cup > Long side of the carton > Short side of the carton > Short side of the suction cup	
Combine suction cups along the X direction of the carton	Illustration	
	Suction cup position 1	
	Suction cup position 2	
Combine suction cups along the Y direction of the carton	Illustration	
	Suction cup position 1	
		
3.12. Vision	Suction cup position 2	

Scenario 4:

Size relationship between the carton and suction cup	Long side of the carton > Long side of the suction cup > Short side of the suction cup > Short side of the carton	
Combine suction cups along the X direction of the carton	Illustration	
	Suction cup position 1	
	Suction cup position 2	
Combine suction cups along the Y direction of the carton	Illustration	
	Suction cup position 1	
		
3.12. Vision	Suction cup position 2	

Strategies for Parallel Suction Cups

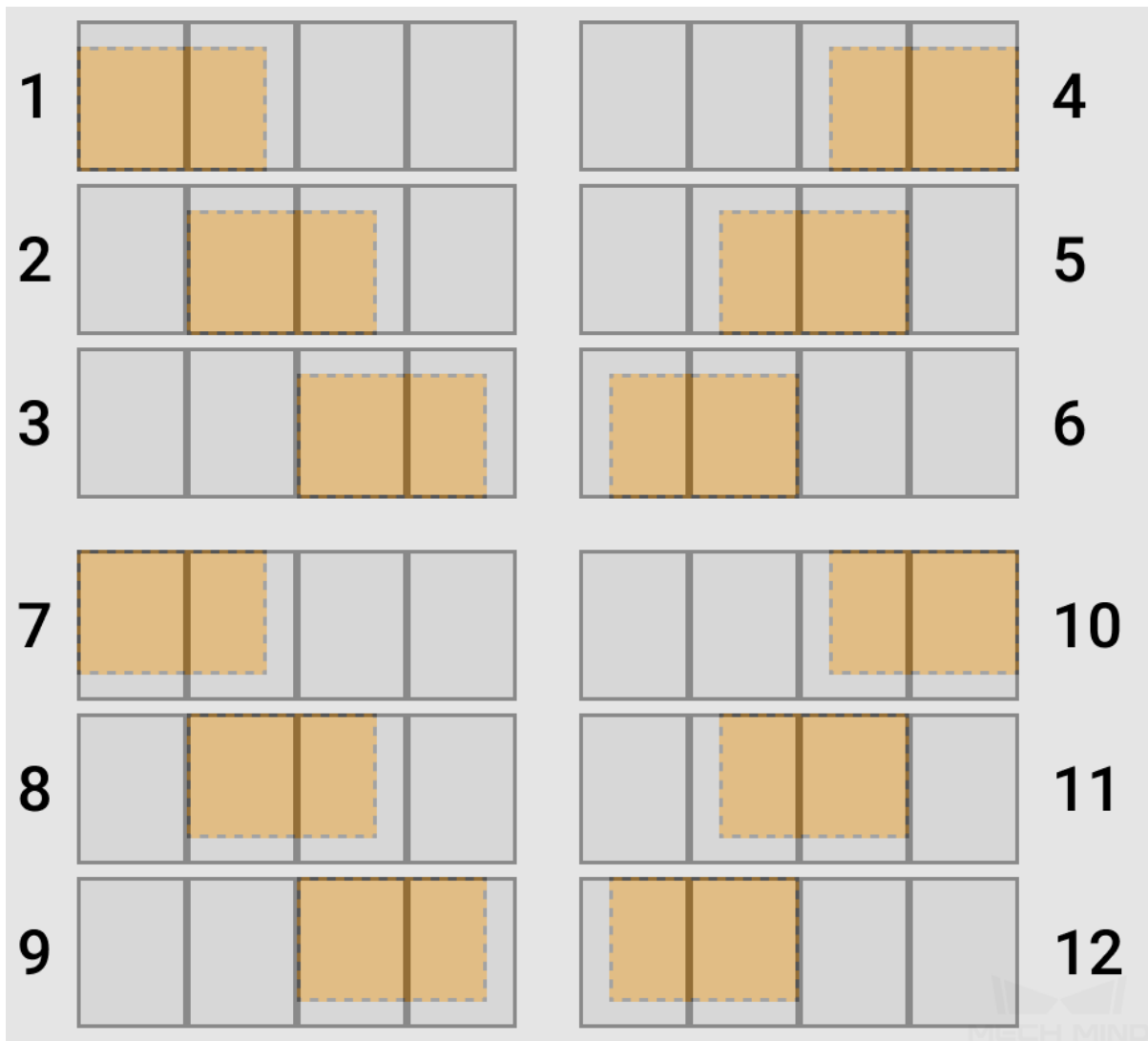
The offset strategies for parallel suction cups are different from that of single rectangle suction cups. Only one offset setting is available when **Parallel Suction Cups** is set as the **Multi Pick Suction Cup Type**, and it entirely depends on the vision result (except point clouds) and carton dimensions provided by the vision service. If the carton is not detected or the dimensions provided by the vision service are wrong, the calculation of the offset setting will be incorrect.

Compared with the offset strategies for a single rectangle suction cup, the offset strategy for parallel suction cups can be considered as a **Corner Prior** offset strategy when the long side of the suction cup is parallel with the direction of the carton group. However, unlike the single rectangle suction cup, each corner of the suction cup sections will be involved in the calculation of the offset setting.

The following example is based on parallel suction cups that contain four sections. The yellow rectangle represents the carton, and the gray rectangle represents the suction cup.

Offset Strategy: Corner Prior + Long side of the suction cup is parallel with the direction of the carton group

Possible suction cup positions: 12 in total, as shown below:



Each section of the parallel suction cups can be started independently. In actual application, even part of the suction cups covers objects that are not supposed to be picked, as long as the suction cup section is not started, those objects will not be picked.

3.13 Others

3.13.1 stop_execution

Description

For the complex usage scenerio, like multi-layer-nested structure: a *procedure & procedure_exit* nests a *procedure & procedure_exit*, etc. If there has no task to be executed in a inner layer, and the whole project

need to be stopped, you can connect this Task

Parameters

exitReason You can describe the reason of stop briefly

Check the chapters below to learn about the functional panels of Mech-Viz and how to configure a project.

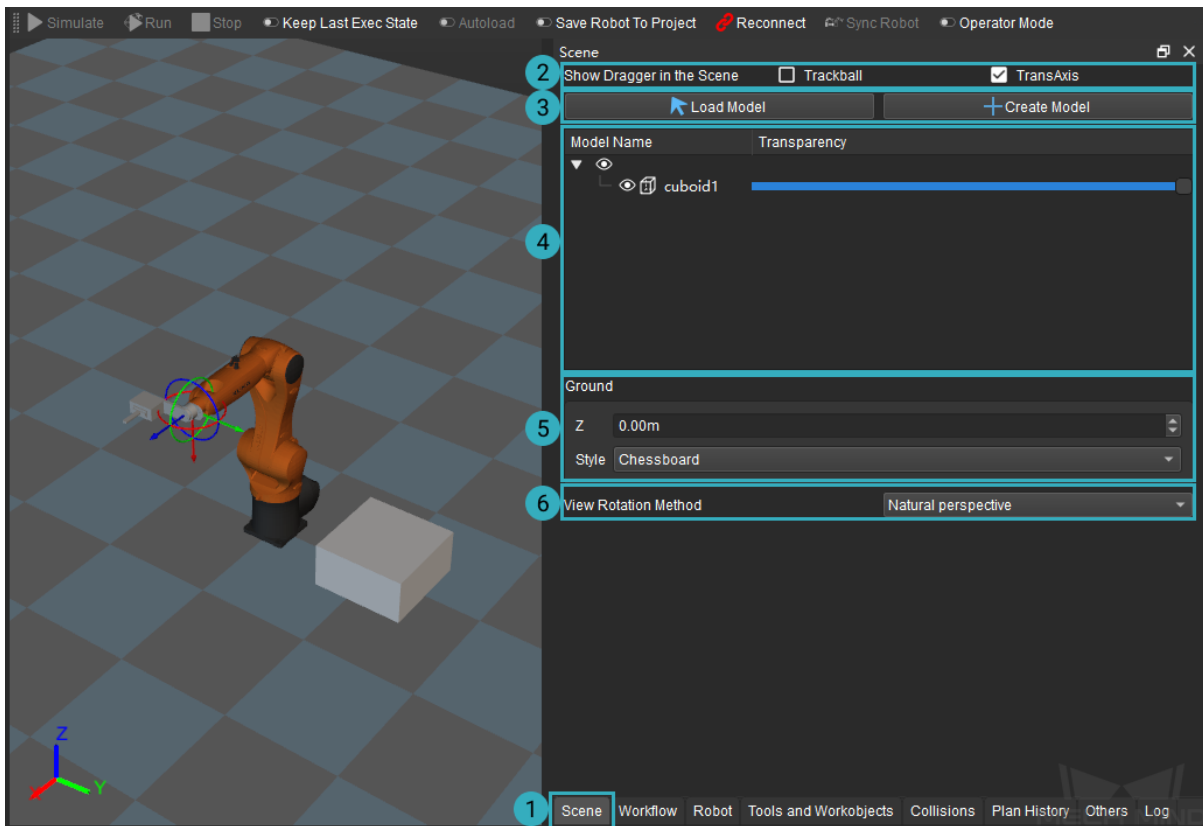
SCENE

When planning the path with Mech-Viz, it is not sufficient to use only one robot model in the 3D simulation area. In order to make the scene closer to the real scenarios and therefore facilitate path planning, collision detection, etc, the models of relevant devices (e.g. camera, base of the robot, etc) and objects (e.g. table, bin, etc) should be added to the scene.

This section covers the following topics:

- *Show Dragger in the Scene*
- *Load and Create Model*
- *Copy, Paste, Delete, and Rename the Model*
- *Edit Model*
- *Adjust Transparency of the Model*
- *Configure the Height and Style of the Ground*
- *View Rotation Method*

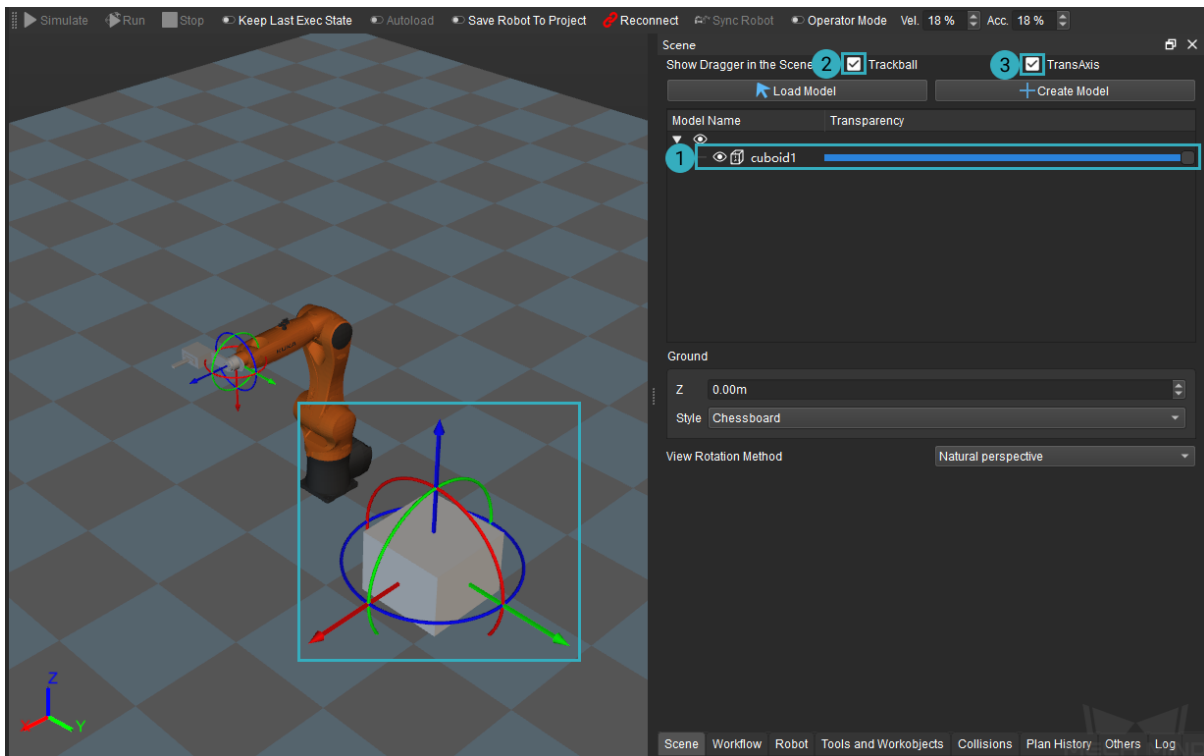
Click on the **Scene** tab in the lower right corner to open the scene configuration panel.



You can configure whether to show the dragger in the scene in , load or create a model in , configure the model in , configure the height and style of the ground in and , and select a view rotation method in .

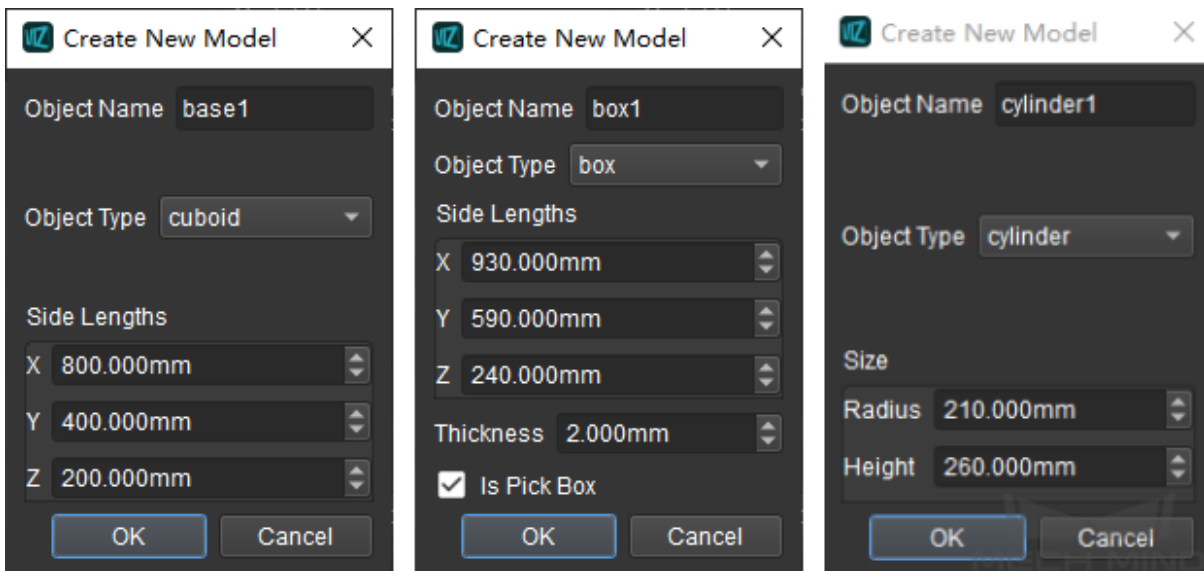
4.1 Show Dragger in the Scene

Select the object model in the scene, click **Trackball** or **TransAxis** will display the dragger of the object. Press and hold the Ctrl key, and then click and drag one of the axes to adjust the object pose, as shown below.



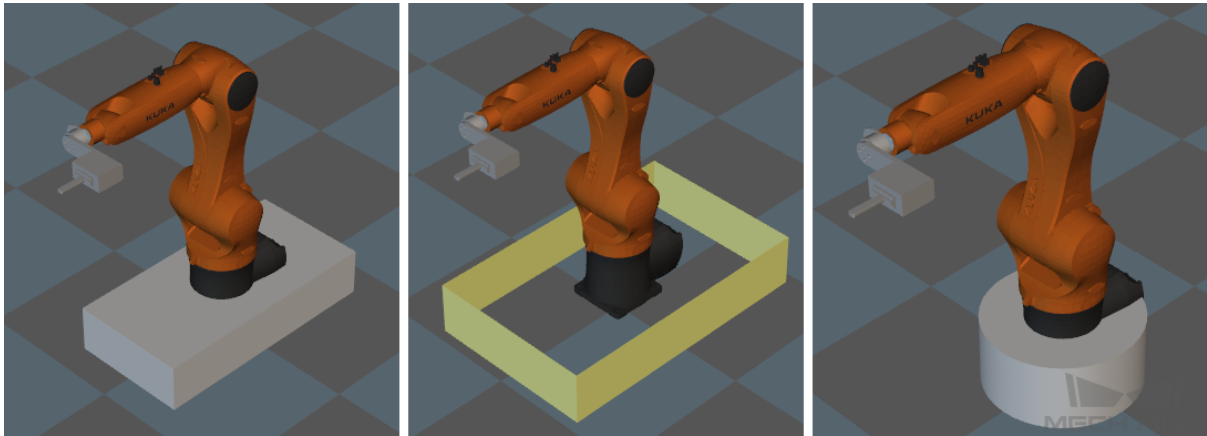
4.2 Load and Create Model

- Click on *Create Model* to add a new object model in the scene. Set the object name, type, and side lengths in the *Create New Model* window.



The created models are shown below: cuboid1 on the left, box1 in the middle and cylinder1 on the

right.



Attention:

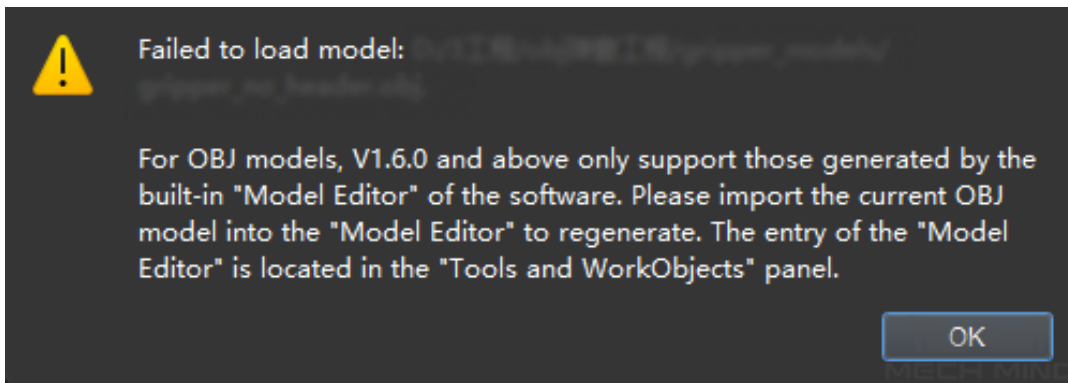
- As for the **object type**, a cuboid is solid and a box is hollow.
- If **Is Pick Box** is checked when the object type is selected as “box”, the color of the box will be different from other objects.
- The pose of the object model cannot be specified when creating the model. Please go to *Edit 3D Object*→ *Object Pose* to adjust the object pose later.

- If you need to use a model from other existing projects, click on *Load Model* to load an existing model (support STL, OBJ, DAE and other formats). Select the model you need in the pop-up window and click on *Open*.

Attention: After loading the model, sometimes the model may be too large for the scene. It is because of the transformation between meter and millimeter. Please double click on the model, select **Model Affine** in the **Edit 3D Object** window and change the **scale** into **0.001**.

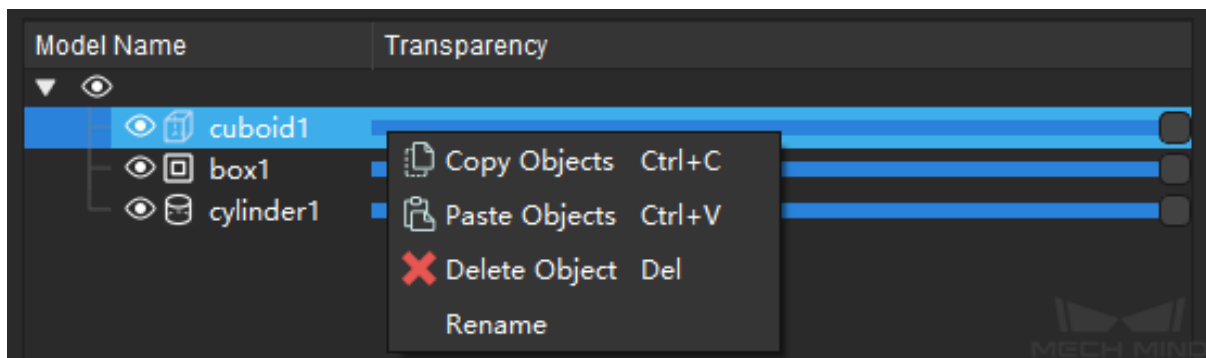
If the collision model is in OBJ format, please read *Notes for OBJ Collision Models* first to avoid collisions.

If the import OBJ scene model is illegal, an alert window as shown below will pop up. Please follow the instructions in the pop-up window and re-export the model from the model editor.



4.3 Copy, Paste, Delete, and Rename the Model

Right click on the added model and you can choose to copy, paste, delete, and rename the model in the context menu.



The shortcuts are shown below.

Options	Shortcuts
Copy	Ctrl+C
Paste	Ctrl+V
Delete	Delete
Multi-select	Ctrl
Undo	Ctrl+Z
Restore	Ctrl+Y

Tip:

- When you copy and paste an object, the sub-objects will be copied and pasted as well.
- After copying an object, if you would like to make it appear in the same class as the parent object, please click at the blank space of the model configuration area to unselect any object first, and then press Ctrl+V to paste the object. If you would like to make the copied object as a sub-object of a certain parent object, please right click on the parent object and select *Paste Objects*.

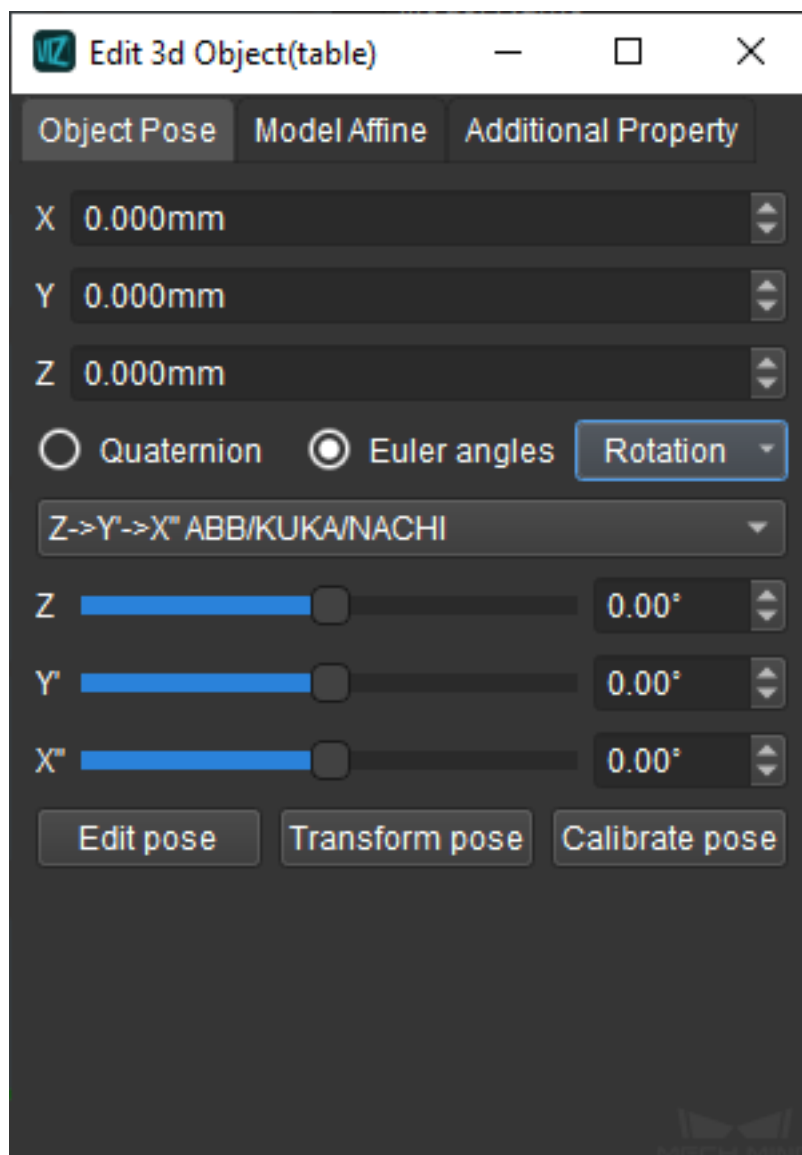
- The copies of the object will be numbered in sequence.

4.4 Edit Model

After loading or creating a model in the scene, you may need to adjust the pose of the object. Please double click on the model to open the **Edit 3D Model** window.

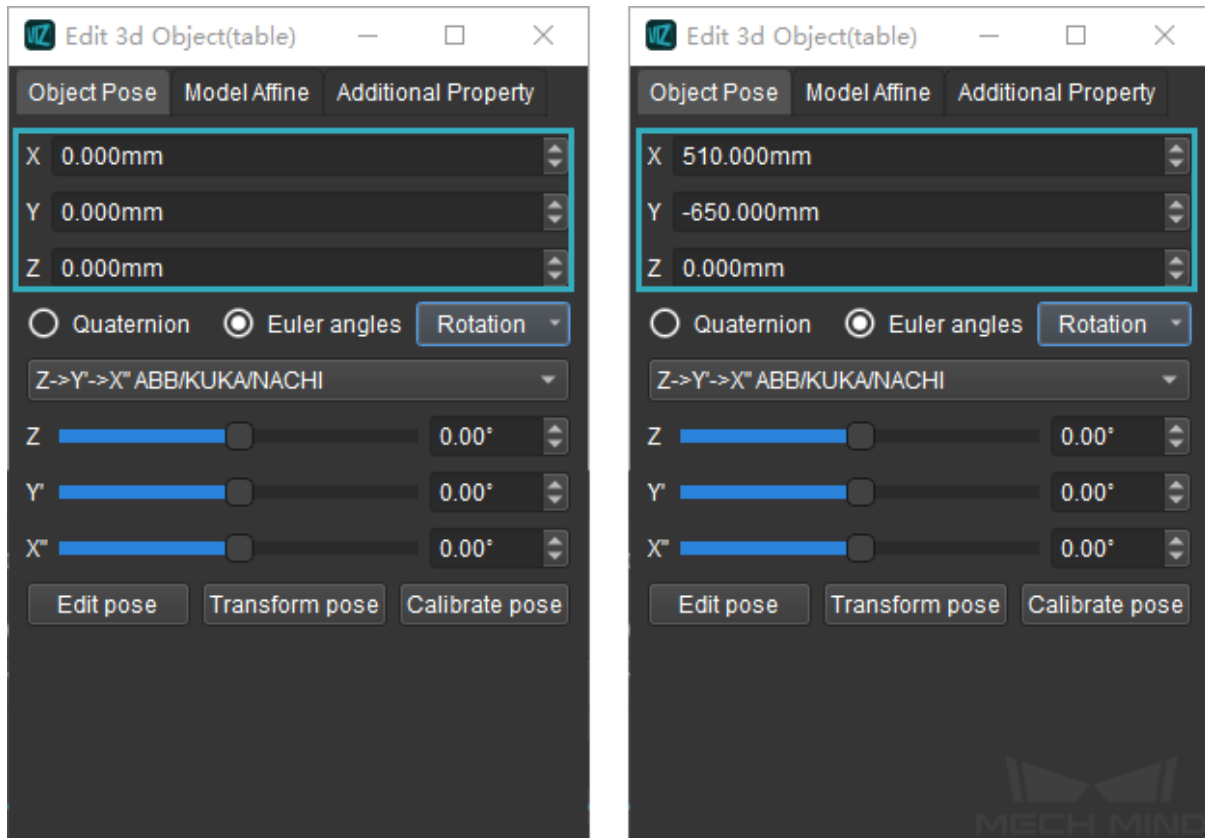
4.4.1 Object Pose

Adjust the object pose in the **object pose** tab.

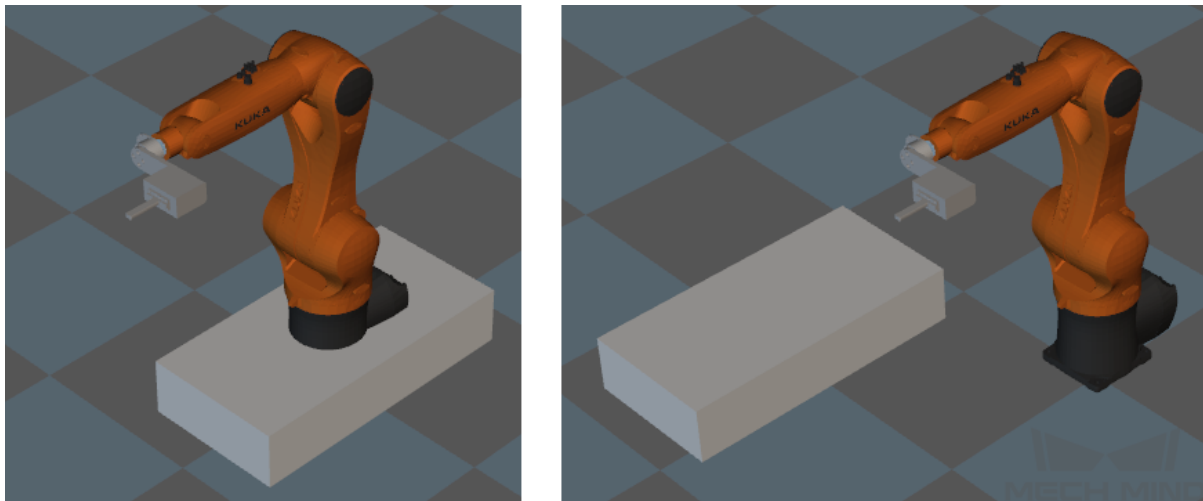


- X, Y, Z: adjust the position of the object to the robot.

The figure below demonstrates the parameters before and after adjusting.

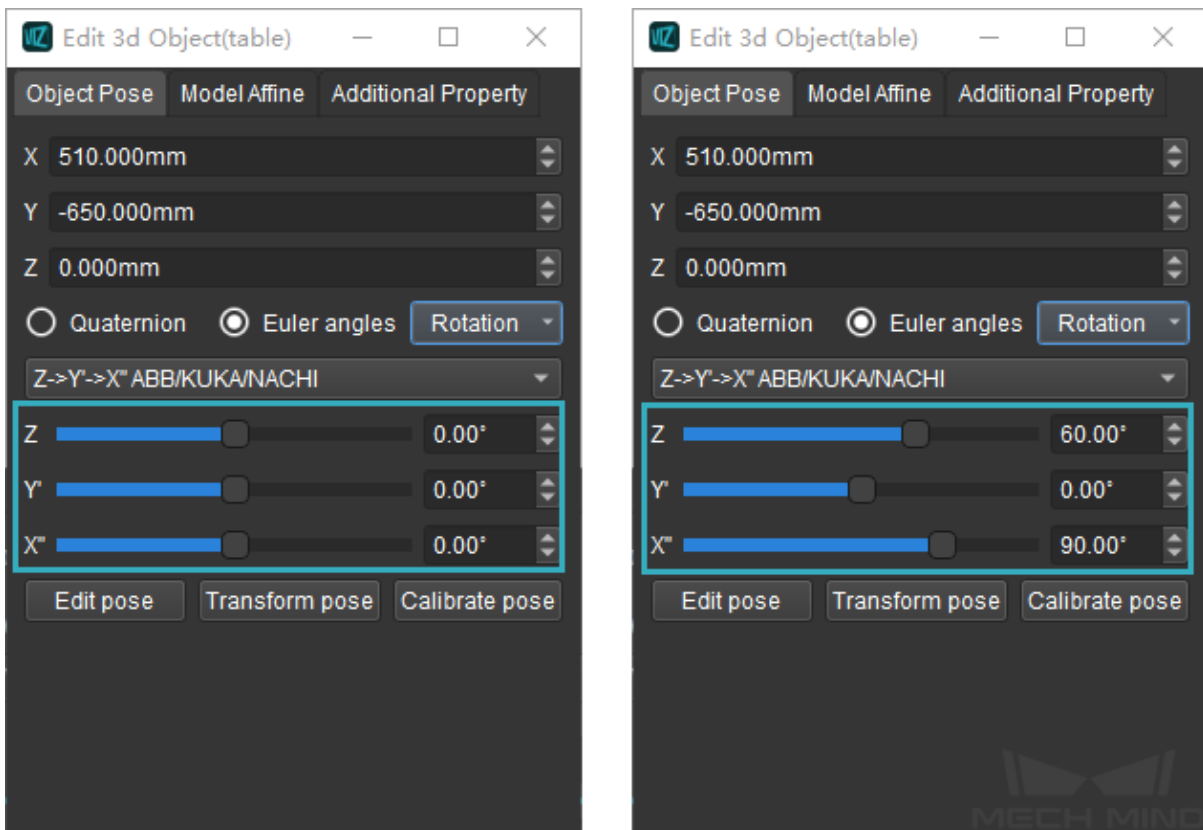


The figure below demonstrates the scene before and after adjusting.

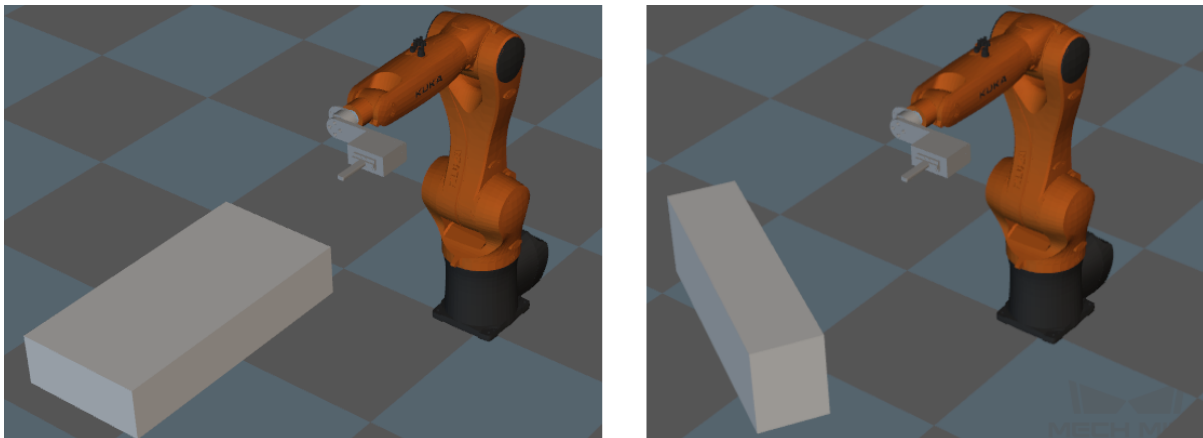


- **Quaternion** and **Euler angles**: adjust the rotation of the object to the robot. When **Euler angles** is checked, please select the sequence of the rotational axis.

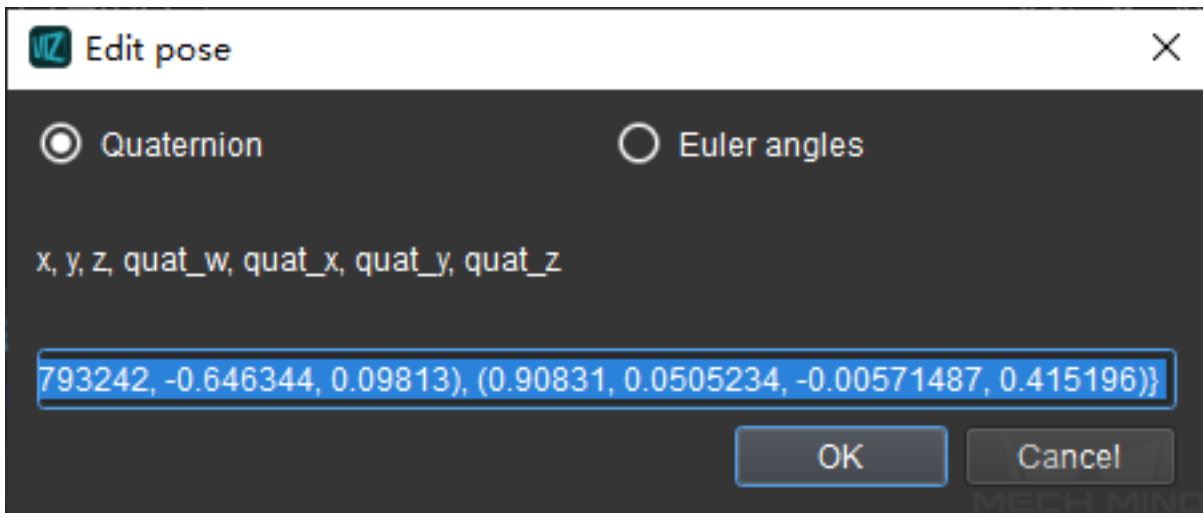
The figure below demonstrates the parameters before and after adjusting.



The figure below demonstrates the scene before and after adjusting.

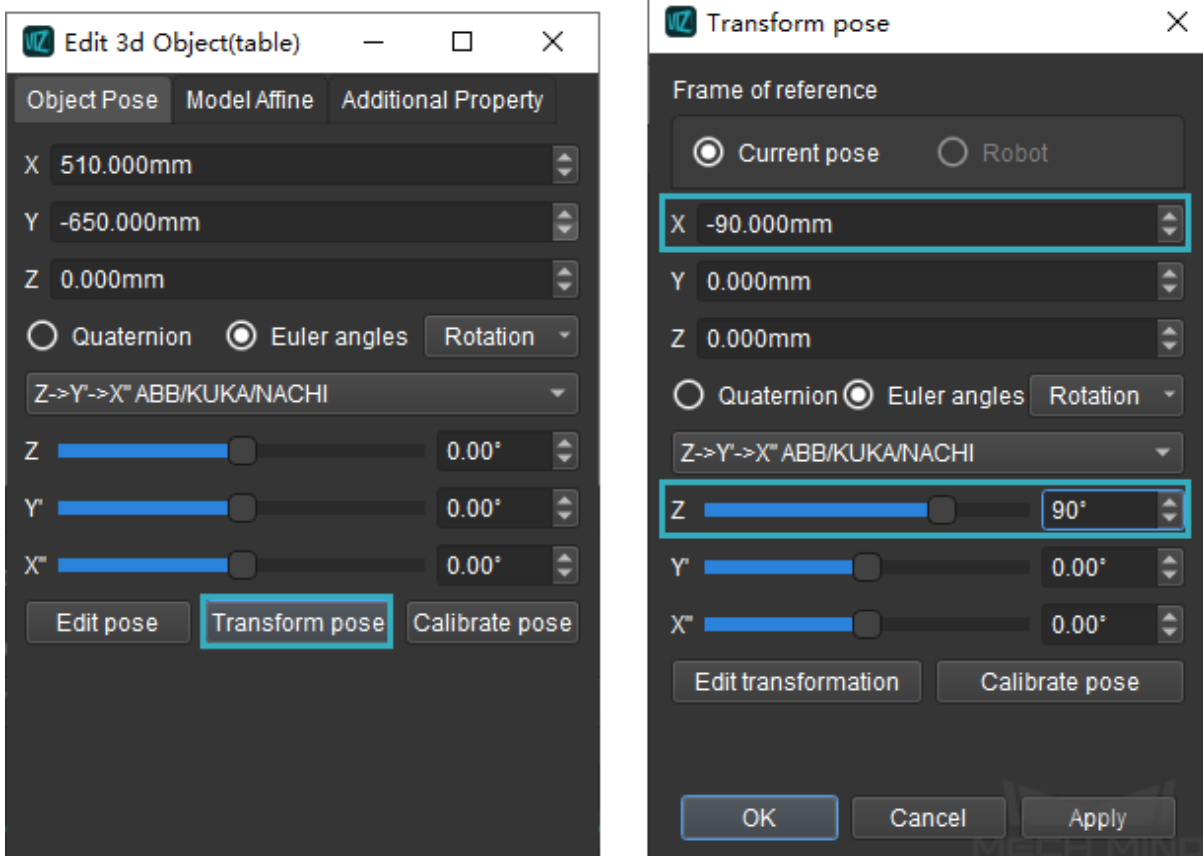


- **Rotation:** includes options as auto align, edit rotation (enter a quaternion), 90° Clockwise Around X, 90° Clockwise Around Y, etc.
- **Edit Pose:** enter a value directly to edit the pose. Please note that the unit used here is meter.

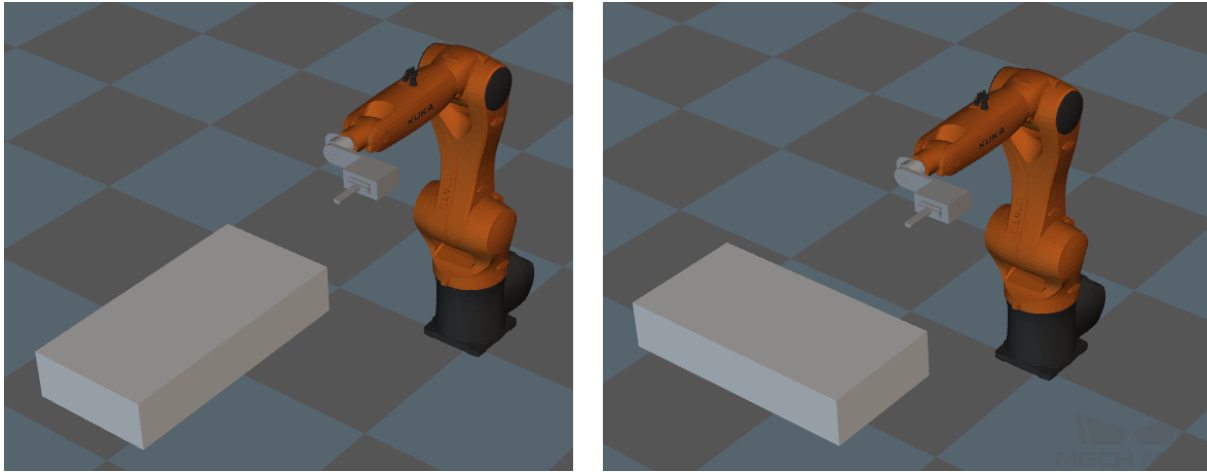


- **Transform Pose:** transform the current pose to a new one. A new coordinate system will be established based on the current pose. Therefore, this option is suitable for adjusting complex poses.

The figure below demonstrates the parameters before and after adjusting.



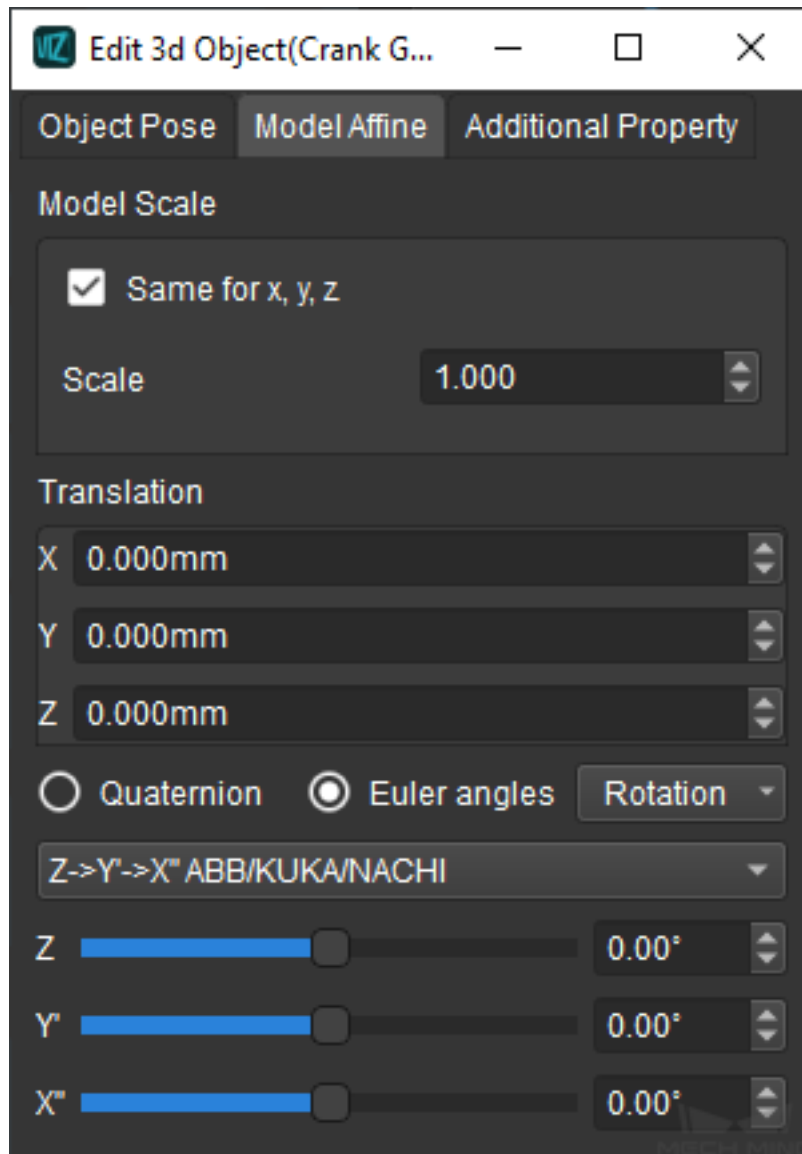
The figure below demonstrates the scene before and after adjusting.



- **Calibrate Pose:** set the coordinate of P1, P2, and P3 according to the instruction, and then calibrate the end effector pose with three-point method.

4.4.2 Model Affine

You can adjust the object pose and apply an affine transformation in the **Model Affine** tab.

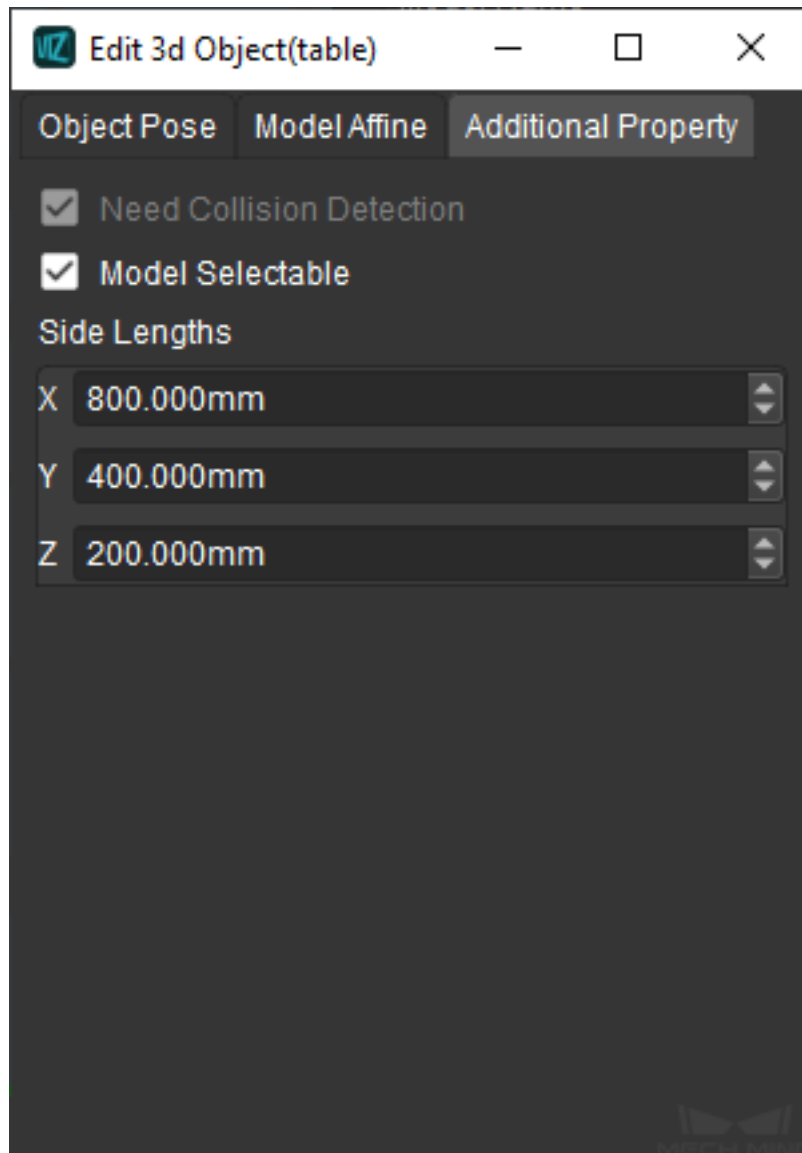


Model Scale: scale the model by setting the model scale. If you need to scale the model in the X, Y, or Z-direction independently, please uncheck **Same for x, y, z** and enter the scale ratio in the X, Y, or Z-direction respectively.

Attention: The Model Affine tab is only available for loaded models.

4.4.3 Additional Property



Set the side lengths and other parameters in the Additional Property tab.

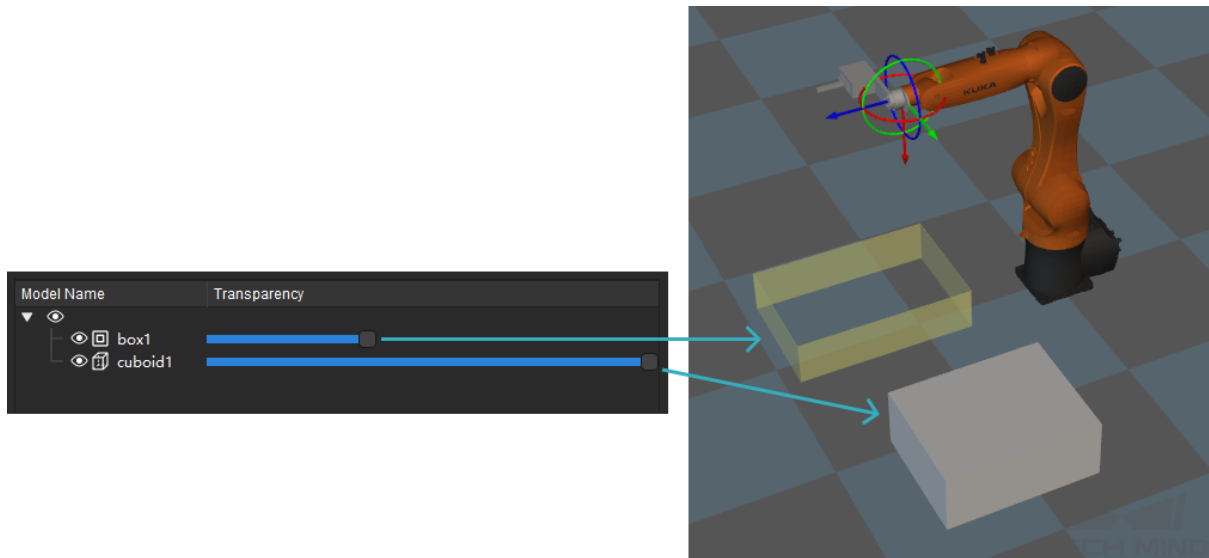


- **Need Collision Detection:** checked by default.
- **Model Selectable:** check to make the model selectable.
- **Side Lengths:** enter the side lengths of the object.

4.5 Adjust Transparency of the Model

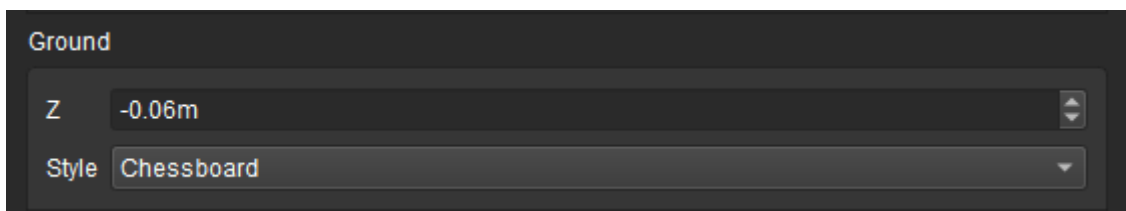
It may be distracting when there is too much objects in the scene or the objects are overlapping. In such case, you can configure to whether display the model or not and adjust the transparency of the model.

Click on the icon on the left to display or hide the model. The icon  indicates that the model is displayed, and  indicates that the model is hid.



4.6 Configure the Height and Style of the Ground

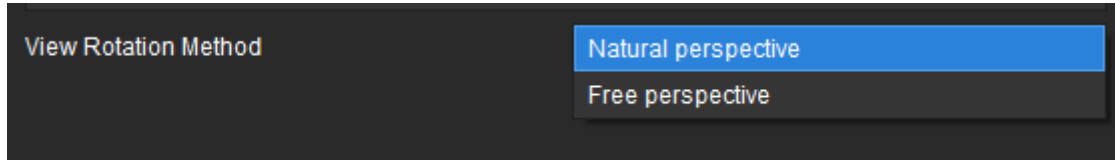
You can set the height and style of the ground in the **Ground** tab.



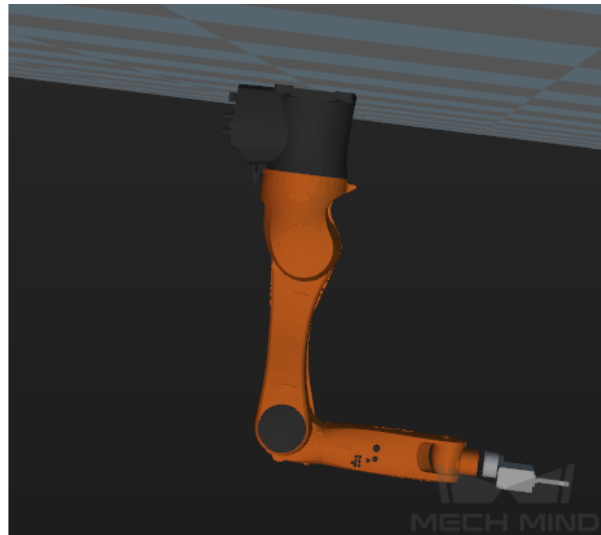
- **Z**: set to adjust the height of the ground, either a positive or negative number is feasible.
- **Style**: available options include **Chessboard**, **Grid**, **Gridlines**, and **None**. It is recommended to select the first two styles.

4.7 View Rotation Method

View Rotation Method includes **Natural perspective** and **Free perspective**. You can click and drag to view the scene. For both perspectives, the center of the view is the center of the coordinate system. Under natural perspective, the maximum degree of rotation about the X-axis or Y-axis is 180° , and the maximum degree of rotation about the Z-axis is 360° . Under free perspective, the the maximum degree of rotation about either X-axis, Y-axis, or Z-axis is 360° .



The figure on the left gives an example of scene under natural perspective, and the figure on the right gives an example of the scene under free perspective. The scene on the right cannot be viewed under natural perspective.



ROBOT

The first thing you might do when creating a project in Mech-Viz is importing a robot model. The **Robot** panel is mainly used for robot configuration and simulation.

This section covers the following topics:

- *Configure the Robot*
- *Real Robot Motion*
- *Joint Positions*
- *TCP*
- *Robot Features*
- *Show TCP Dragger*

Click on the **Robot** tab in the lower right corner to open the robot panel.


Robot Library

Brand Axes


Payload (kg) 0 800

Reach (mm) 0 4100


Current Robot




KUKA_KR6_R900_SIXX
 Payload: 6 kg
 Axes: 6
 Reach: 901.5 mm




KUKA_KR6_R700_SIXX
 Payload: 6 kg
 Axes: 6
 Reach: 706.7 mm



KUKA_KR6_R900_2
 Payload: 6.5 kg
 Axes: 6
 Reach: 901 mm



KUKA_KR6_R900_SIXX
 Payload: 6 kg
 Axes: 6
 Reach: 901.5 mm



KUKA_KR10_R1100_2
 Payload: 11.1 kg
 Axes: 6
 Reach: 1101 mm

If you cannot find the robot you need in the Robot Library, please refer to [Robot Model Package](#) or click on [Online robot gallery](#) in the lower left corner to check whether the robot model is available.


- If there is an available robot model package of the type you need, you can download it by clicking on the type of the robot you need. Then you can import the robot model package to Mech-Viz by either of the following methods.
 - Select the MROB file and drag it to the interface of Mech-Viz.
 - Go to *Tools* → *Robot Library Tools* → *Import Robot*, select the downloaded MROB file and click on *Open*.

A message saying **Robot import finished** indicates that the robot model package has been imported successfully.

If there is not an available robot model, you will need to [create a robot model](#) yourself and then import it to

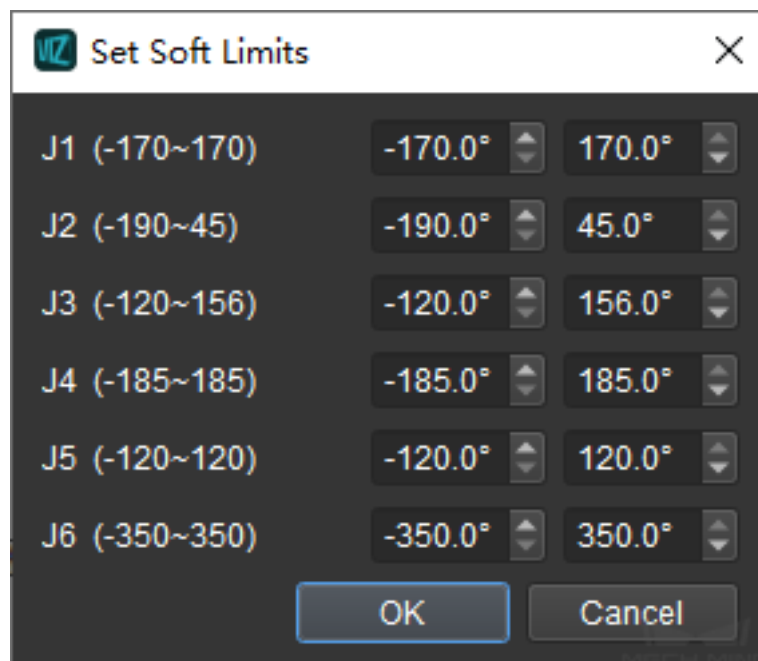
Mech-Viz.

The descriptions of the options in the Robot Library are as follows.

- **Search Box:** enter the brand and model of the robot to search the robot model you need. Click on  to clear the content in the search box. You can also set the desired **Brand**, **Axes**, **Payload**, and **Reach** to filter the robot.
- **Search Result:** the robot model will appear on the right if there is any matching result. Hover the cursor over the robot and a *Select* button will appear, click on the button and then the selected robot model will appear in the 3D simulation area.
- **Current Robot:** displays the robot model which is used in the 3D simulation area at present.

Hint: Please note that the keywords in the search box are not case sensitive. It is recommended to enter the brand name or model name independently to search the robot. Please use a underscore to connect the words when entering the brand name and model name together (e.g. abb_crb15000), for a space may lead to no result.

Soft limits are the movement limits of each axis set in the software to avoid the risks of collision. Click on *Set Soft Limits* and a window to configure the limits will appear, as shown below.

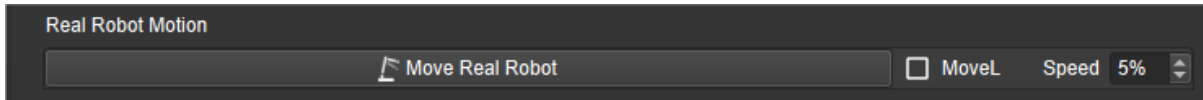


After the soft limits are set, the joint positions in relevant Tasks (e.g. move) should be selected from the range set here.

Attention: The soft limits set here will only take effect when Mech-Viz is used, and will not affect the soft limits set on the teach pendant.

5.2 Real Robot Motion

If a real robot has been connected, click on *Move Real Robot* will move the real robot to make its JPs the same as that of the simulated robot. You can select **MoveL** to make the robot move linearly and set the speed for the move.



Attention: Please click on the *Sync Robot* on the toolbar to synchronize the connected real robot.

You can configure the **Joint Positions** and **TCP** to adjust the pose of any robot.

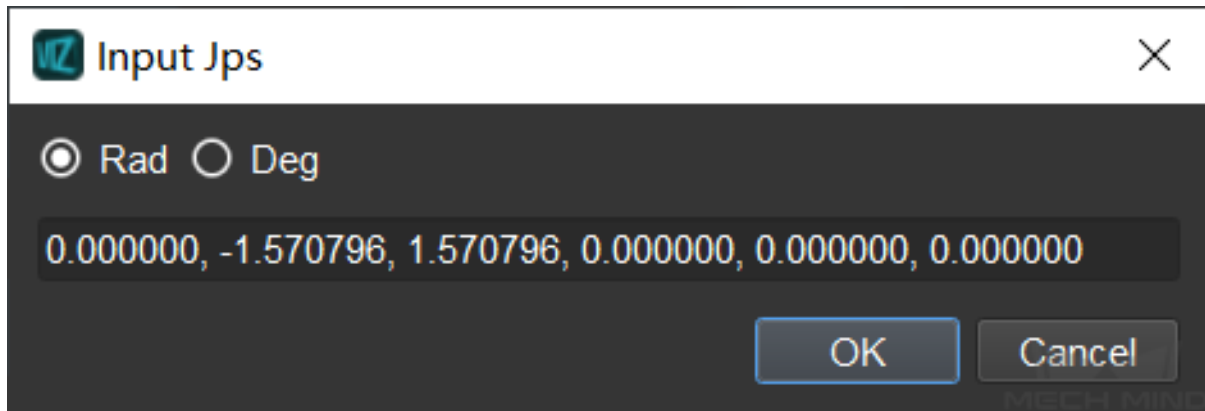
5.3 Joint Positions

Click on the **Joint Positions** tab to view the JPs of each axis of the simulated robot. You can drag the slider or enter a value in the box on the right to adjust the JPs, and the robot in the 3D simulation area will move according to the adjustment in real time.



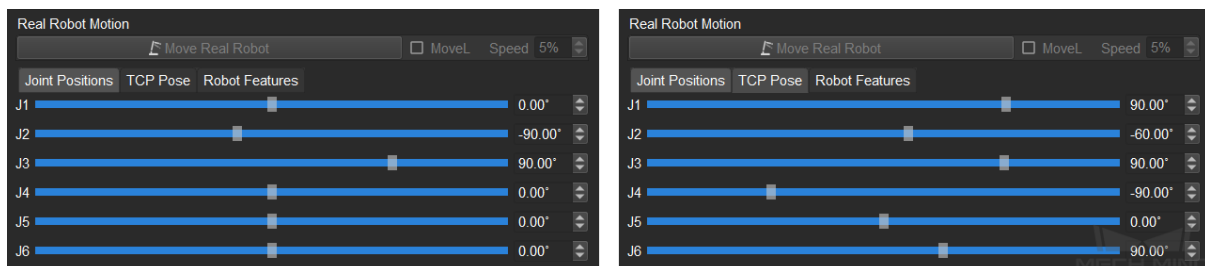
- Edit JPs: enter the value of JPs in radians or degrees in the pop up window.

Hint: $1^\circ = \pi/180$, $1 \text{ rad} = (180/\pi)^\circ$

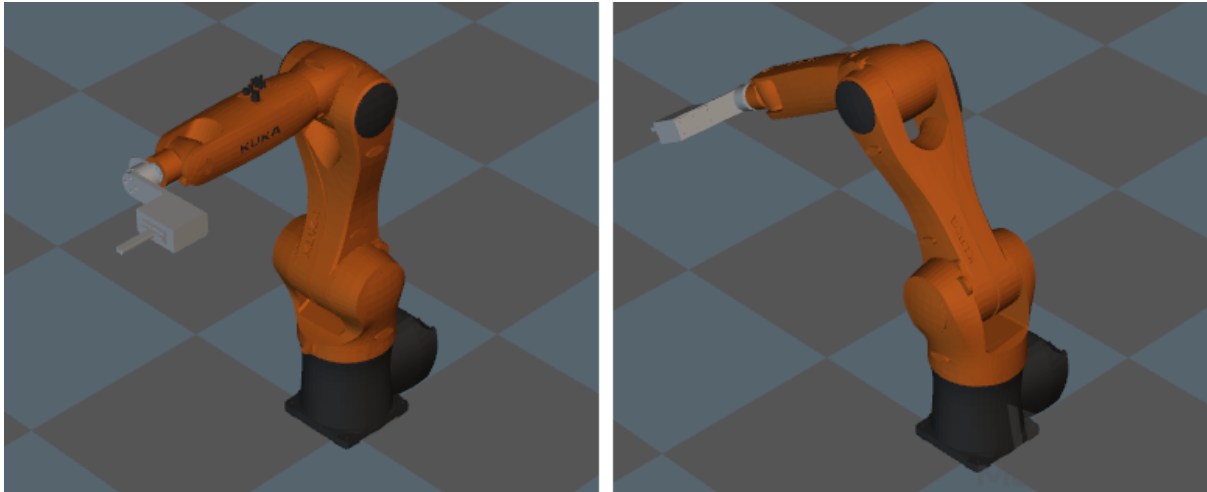


- **To Default HP:** click to bring the robot back to the home position specified in the robot configuration file (i.e. the default home position in Mech-Viz).
- **To User HP:** click to bring the robot back to the home position defined by the user.
- **Set User HP:** drag the slider or enter a value of JPs in the **Real Robot Motion** panel to adjust the robot to a pose that you would like to set as the home position, and then click on *Set User HP* to set it as the home position.

The figure below shows the JPs before and after being adjusted.



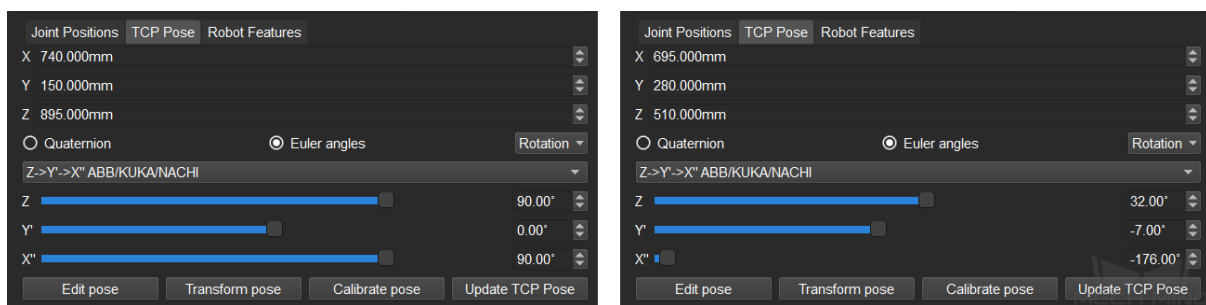
The figure below shows the corresponding robot pose before and after the adjustment.



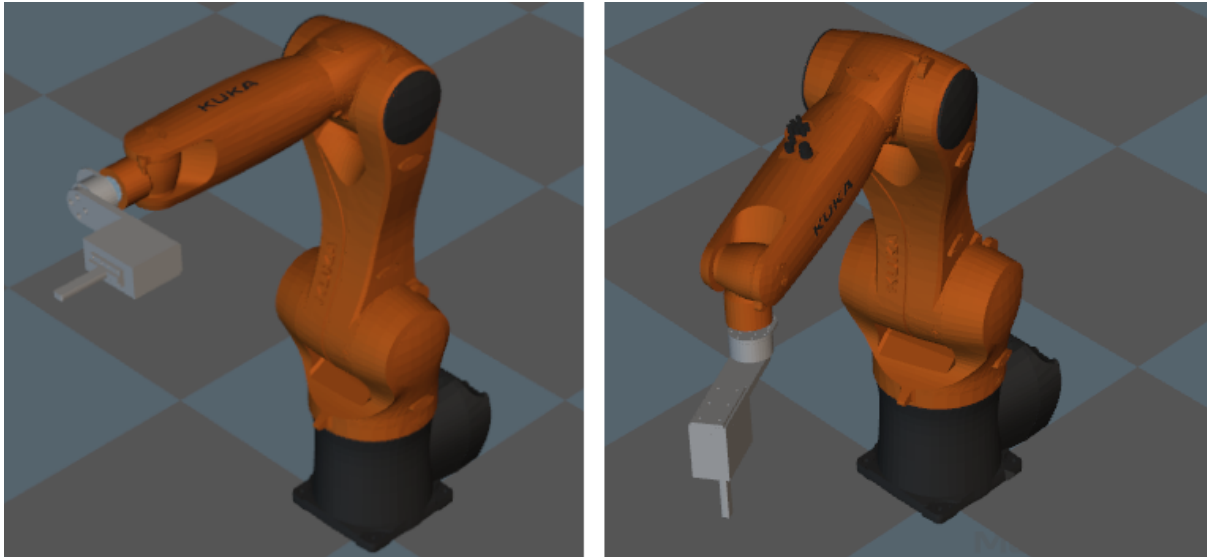
5.4 TCP

Click on the **TCP** tab to view the current TCP of the simulated robot. You can adjust the TCP by changing values in quaternions or Euler angles in the panel, or click on *Edit pose* and enter the pose directly.

The figure below shows the TCP before and after being adjusted.

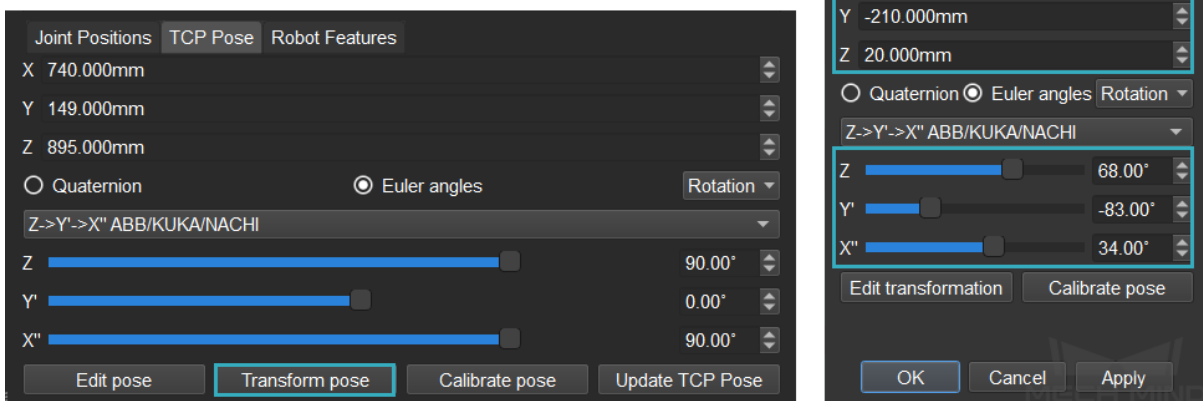


The figure below shows the corresponding TCP before and after the adjustment.

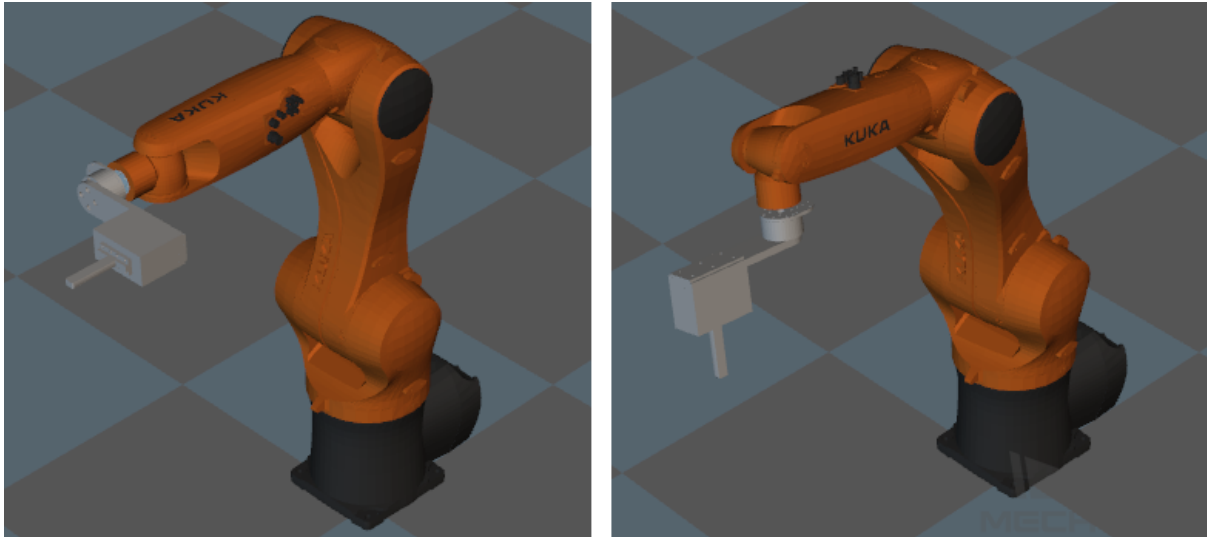


- **Transform pose:** transform current pose to a new pose. Select a frame of reference first, and then edit the transformation with the options in the window. The robot will move according to the settings in real time. Click on *Apply* and then *OK* to save the settings. If you do not want to apply the transformation, click on *Cancel* to exit, and the robot will move back to the original pose.

The figure below shows the TCP before and after being adjusted.



The figure below shows the corresponding TCP before and after the adjustment.



- **Calibrate pose:** set the coordinate of P1, P2, and P3 according to the instruction, and then calibrate the TCP with three-point method.

Mech-Viz
✕

Three-point method 1

P1 is the origin.
 P2 is a point on the X-axis.
 P3 is a point on the X-Y plane.
 Please make sure the X-coordinate of P2 and the Y-coordinate of P3 are above zero.

P1

x y z Get position

P2

x y z Get position

P3

x y z Get position

OK Cancel Apply

- **Update TCP:** click to obtain the TCP from a real robot when the robot is connected.

5.5 Robot Features

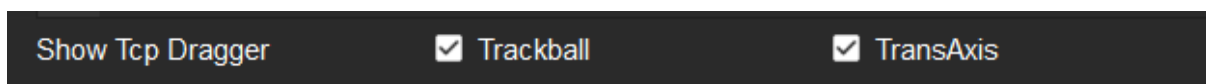
Click on **Robot Features** tab to view whether the robot features are applicable in Mech-Viz.

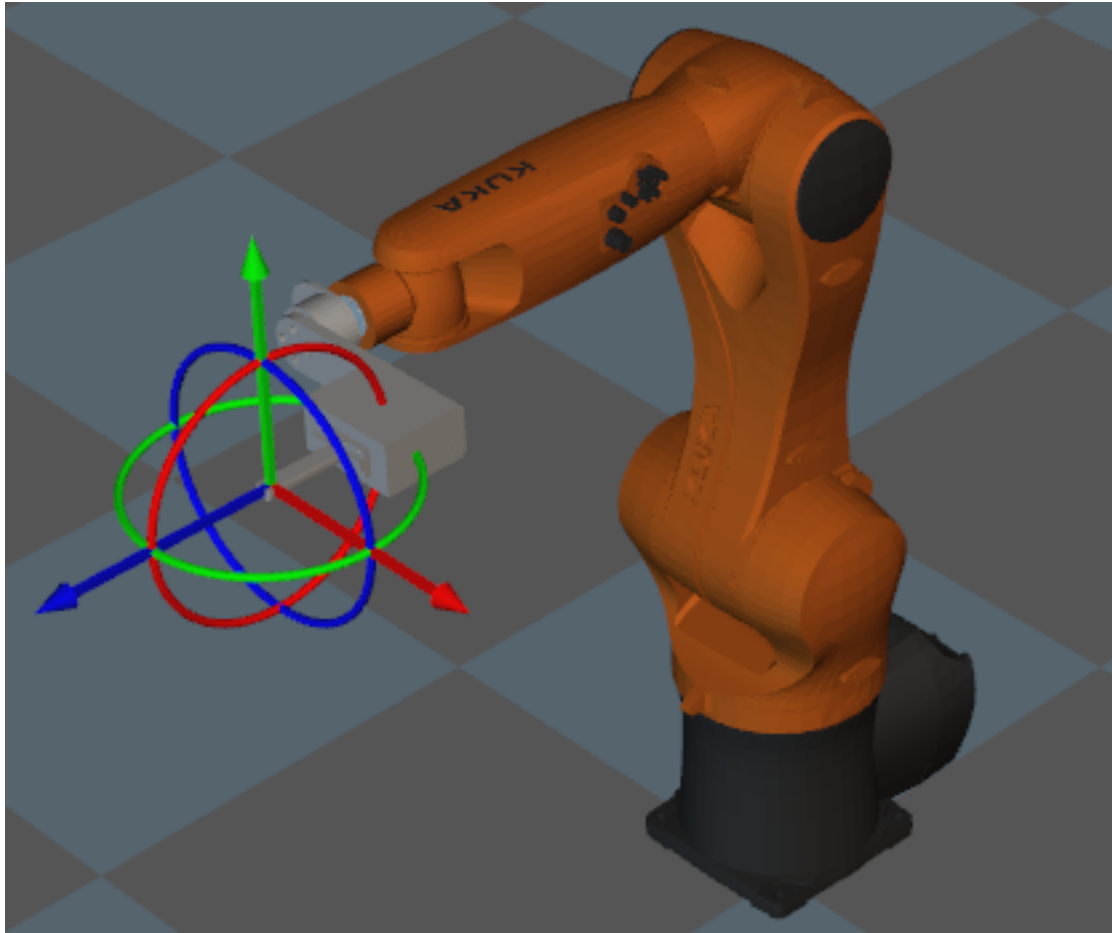
Joint Positions			TCP Pose			Robot Features		
Robot Features						Applicable		
1	Stop immediately at stop signal					✓		
2	Pause					✗		
3	Move until receiving DI signal					✗		
4	Set DO while moving					✗		
5	Set delay time with DO					✓		
6	Move along dense targets					✓		
7	Set TCP					✓		
8	Set payload					✓		
9	MoveC					✗		
10	Reconnect through RobotServer					✗		

5.6 Show TCP Dragger

This option is used to hide or display the TCP dragger at the end of the robot. Uncheck the box before **Trackball** and **TransAxis** to hide the dragger.

Hint: Press and hold the **Ctrl** key and you can drag the dragger to adjust the TCP more intuitively.





TOOLS AND WORKOBJECTS

You can configure end effectors and work objects in the **Tools and Workobjects** panel.

End effectors are usually connected to the end of robot arms. They act as hands of the robot to interact with the surroundings.



End Effector Configuration

Workobjects refer to objects to be picked. Each workobject has a corresponding pose. Mech-Viz computes the pose of end effector according to workobject poses, and therefore guides the robot to pick.

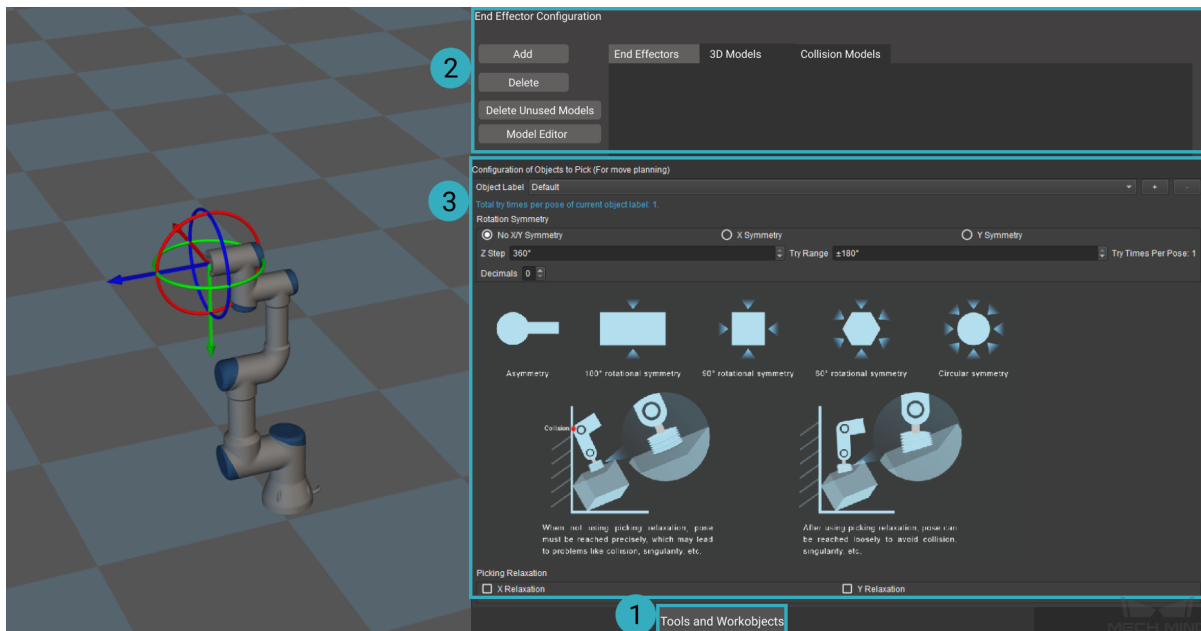
Workobjects Configuration

Check the section below to learn about rotational symmetry and picking relaxation.

Rotational Symmetry and Picking Relaxation

Click the **Tools and Workobjects** tab in the lower right corner of the interface to open the panel.  is where you can configure the end effector and  is where you can configure the workobjects to be picked.

Hint: If you cannot see the **Tools and Workobjects** tab in the interface, please go to *View* and check the box before **Tools and Workobjects**.



6.1 End Effector Configuration

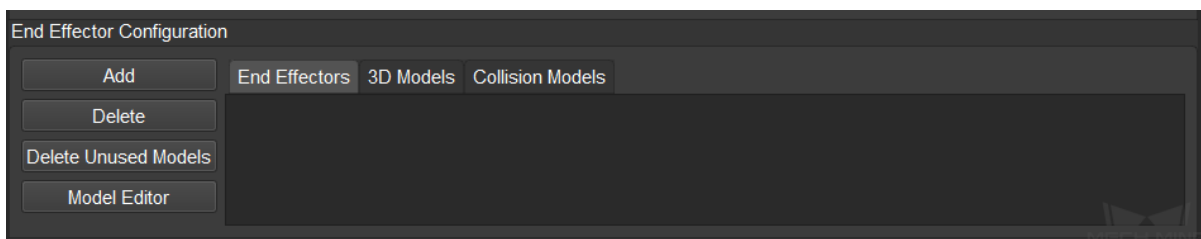
In order to facilitate picking, you can configure the end effector in the **Tools and Workobjects** panel.

This section covers the following topics:

- *End Effector Configuration Panel*
- *Add an Model*
- *Recommended Method for Using Models*
- *Configure the End Effector Model*
- *Define the TCP*

6.1.1 End Effector Configuration Panel

The end effector configuration panel is shown below.



The functions of the tabs and buttons are described as follows.

- **End Effectors:** displays the added end effector models.
- **3D Models:** displays the added 3D models.
- **Collision Models:** displays the added collision models.
- **Add:** adds an end effector model, 3D model, or collision model.
- **Delete:** deletes a selected end effector model, 3D model, or collision model.
- **Delete Unused Models:** deletes unused 3D models or collision models.
- **Model Editor** used to simplify models. Please refer to *Model Editor* for detailed information.

Hint:

- When multiple end effector models are added, the first one in the list will be used by default.
 - You can also right click on the added model and select **Delete** to delete the model.
-

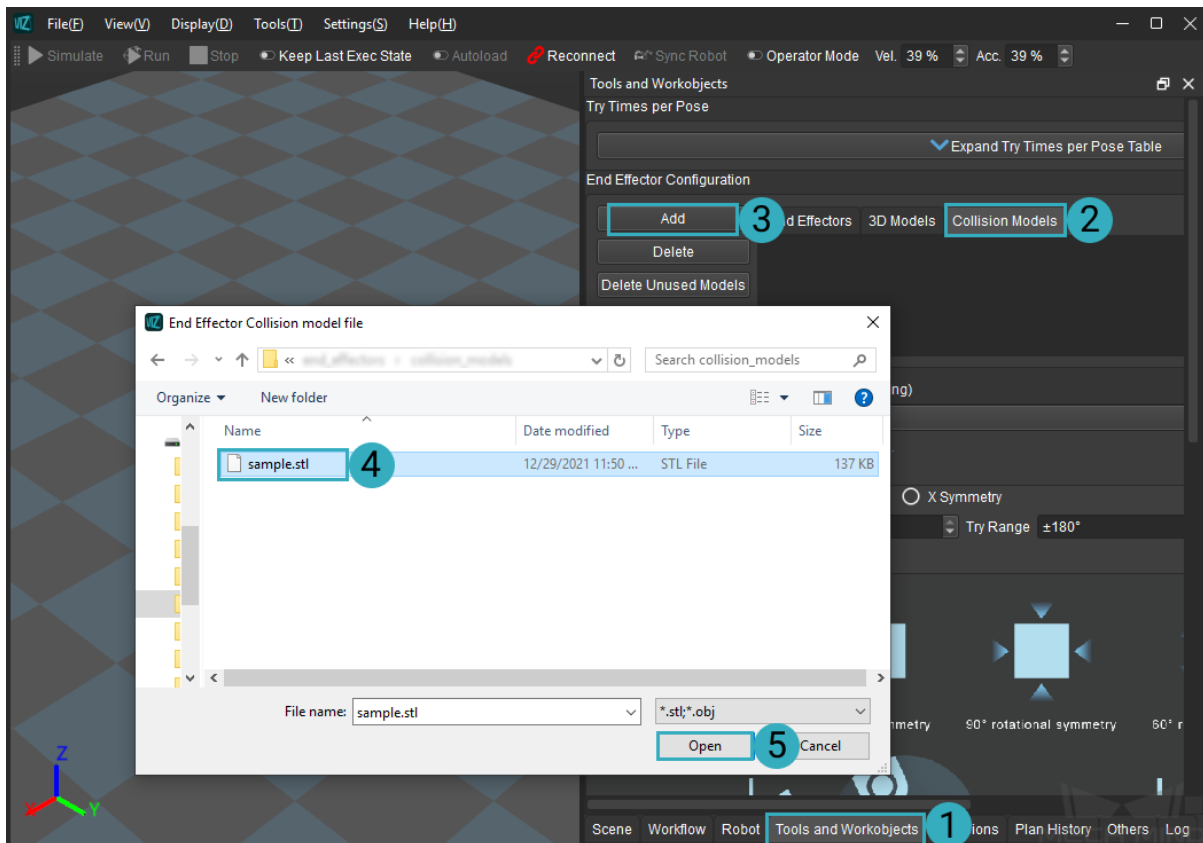
6.1.2 Add an Model

Adding a 3D model and collision model may facilitate collision detection and path planning in Mech-Viz.

3D models can only be used for visualization, while collision models can be used for collision detection as well. Therefore, adding a collision model is more recommended.

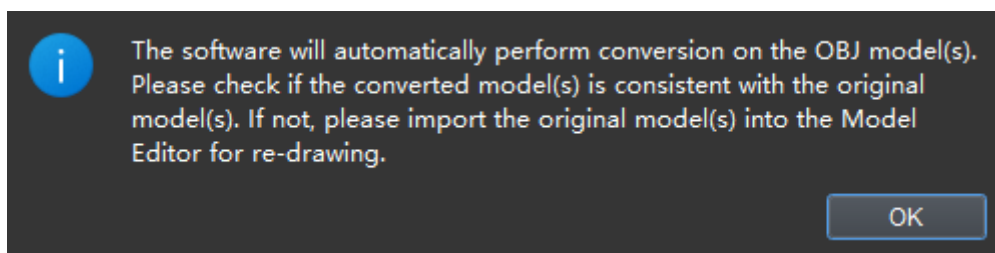
If the collision model is in OBJ format, please read *Notes for OBJ Collision Models* first to avoid collisions.

1. Click on the *Tools and Workobjects* tab, and then go to the **End Effector Configuration** panel. Select *Collision Models*→ *Add*, select the model file you need in the file selection window, and then click on *Open*.



Model Type	Instruction
OBJ models that are entirely composed of convex polyhedra	Load normally
OBJ models not entirely composed of convex polyhedra	Follow the instruction in the pop-up window to convert the model

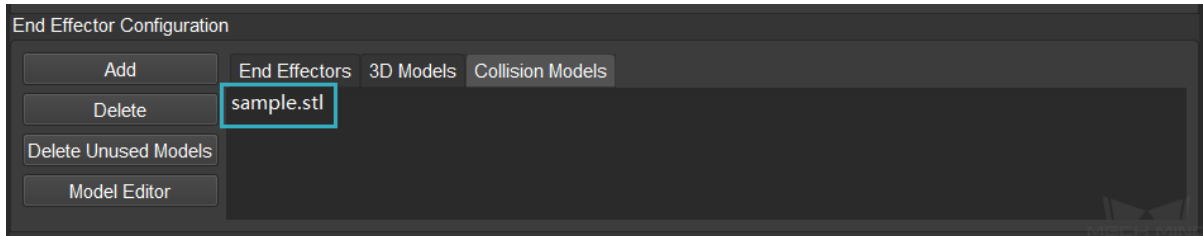
The pop-up window that suggests to convert the model is shown below.



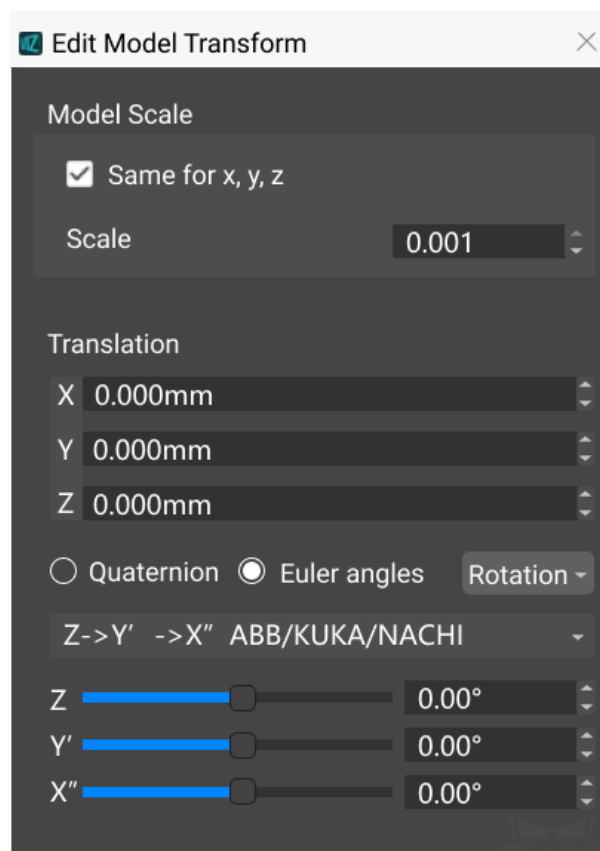
Click *OK* to convert and load the model. The shape of the converted model may change. In this

case, please use Blender or *Model Editor* in Mech-Viz to edit the models before importing to make sure that the model can meet the requirement.

2. Double click on the added model in the **Collision Models** list.

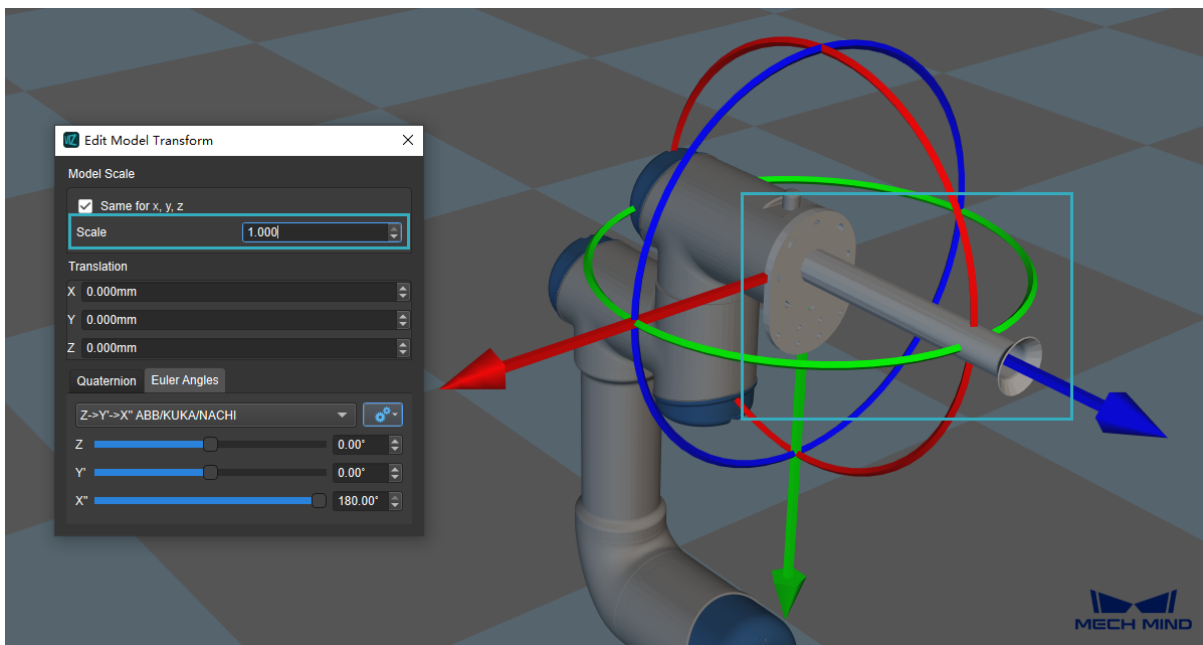
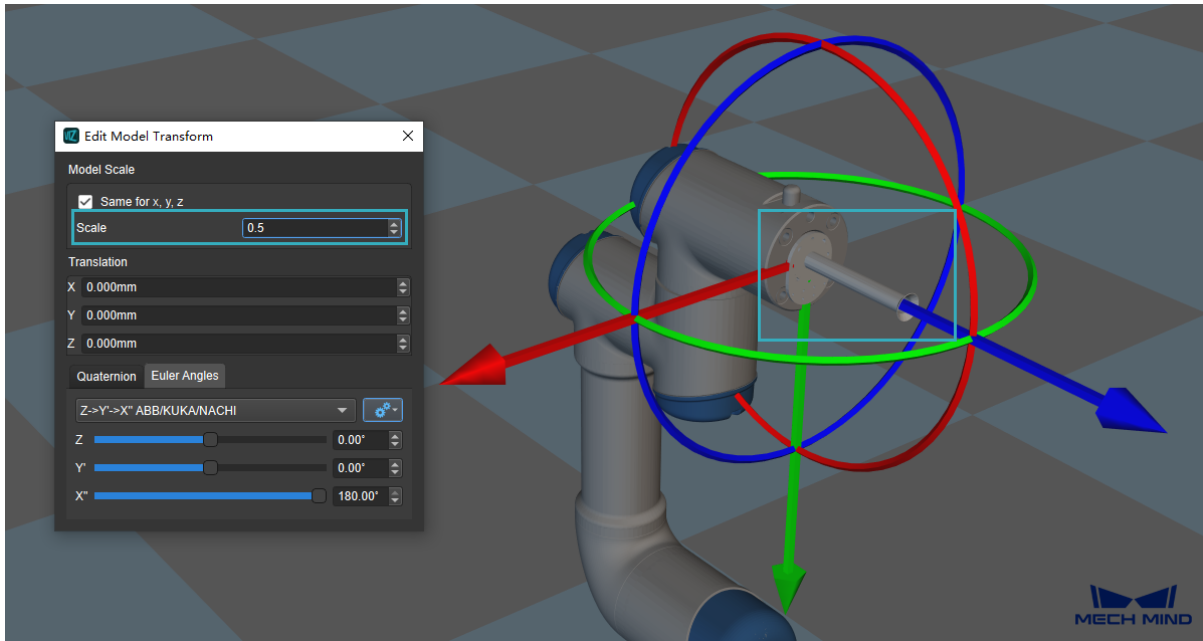


3. Configure the parameters in the pop-up **Edit Model Transform** window.

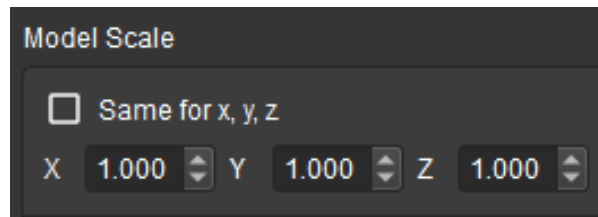


The functions of the buttons and options in the window are described as follows.

- **Model Scale:** scale the model by setting the model scale. The figures below show an example of setting different model scales.

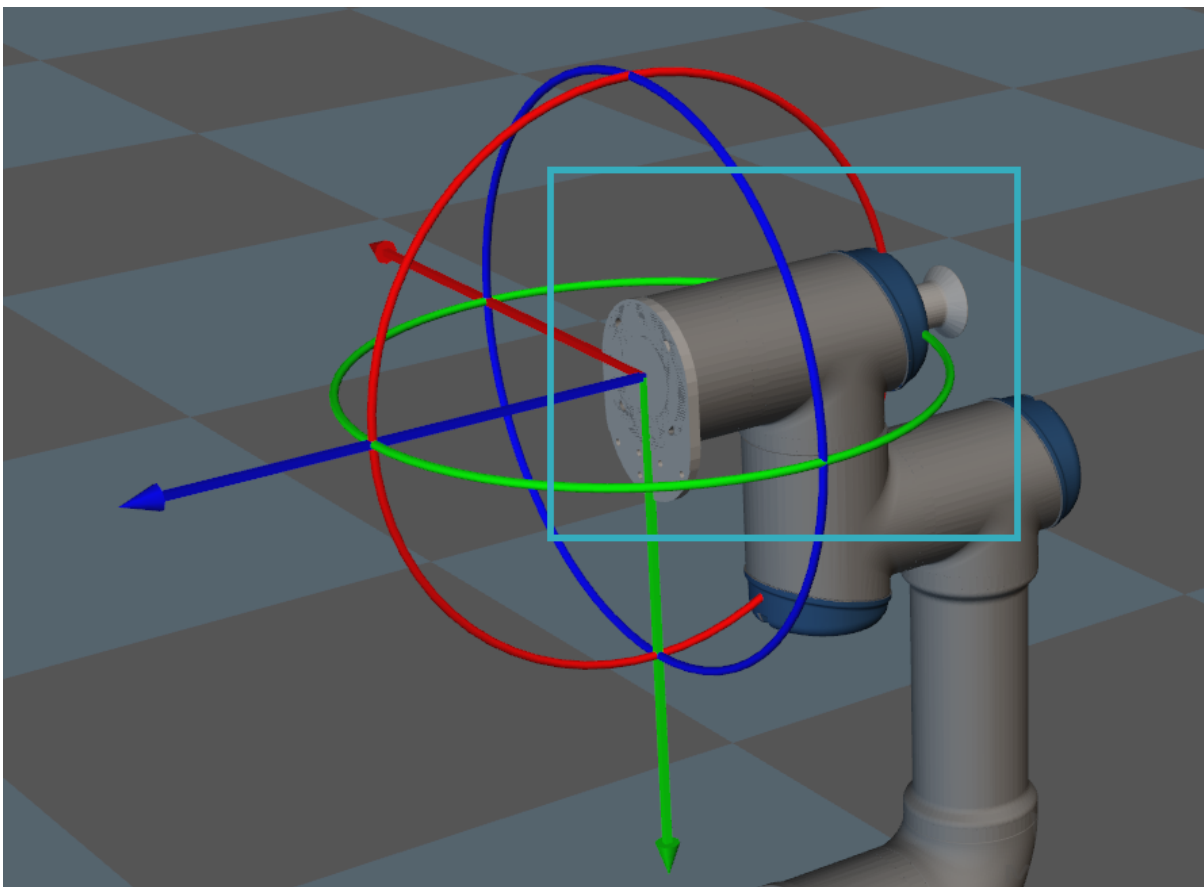


If you need to scale the model in the X, Y, or Z-direction independently, please uncheck **Same for x, y, z** and enter the scale ratio in the X, Y, or Z-direction respectively.

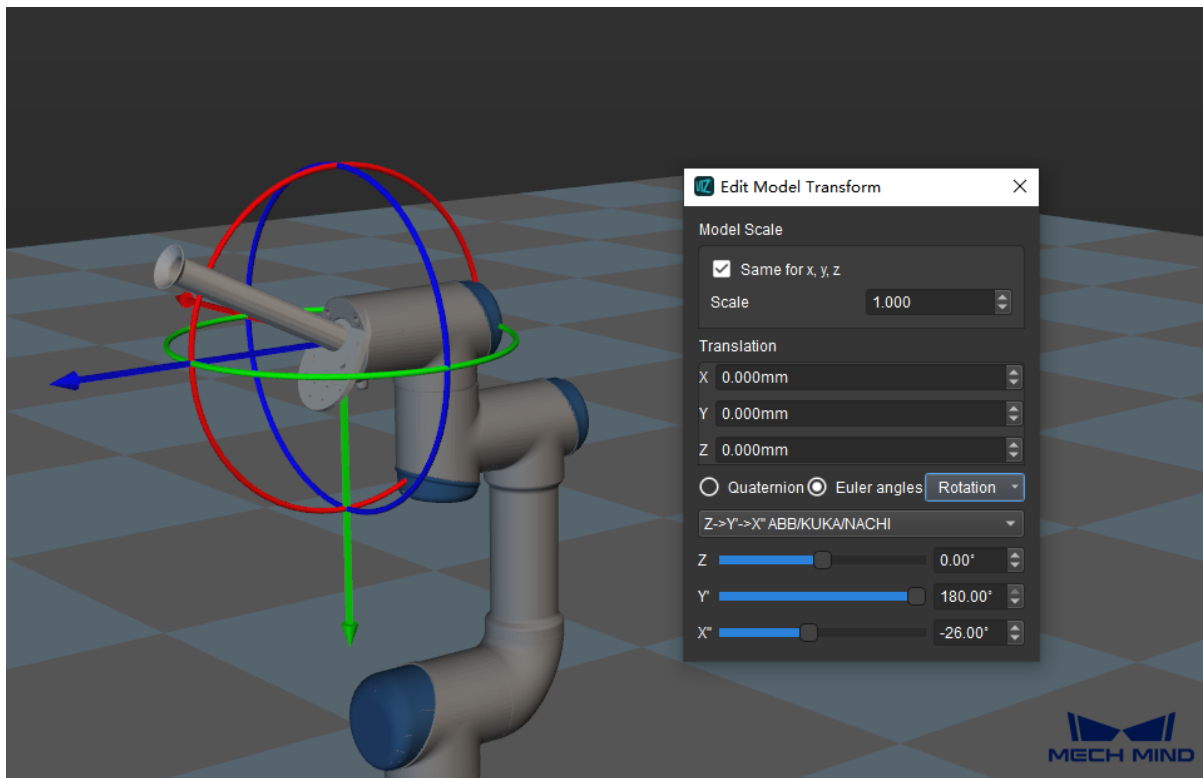


- **Translation:** adjust the position of the collision model to the robot flange by adjusting the value of X, Y, and Z.

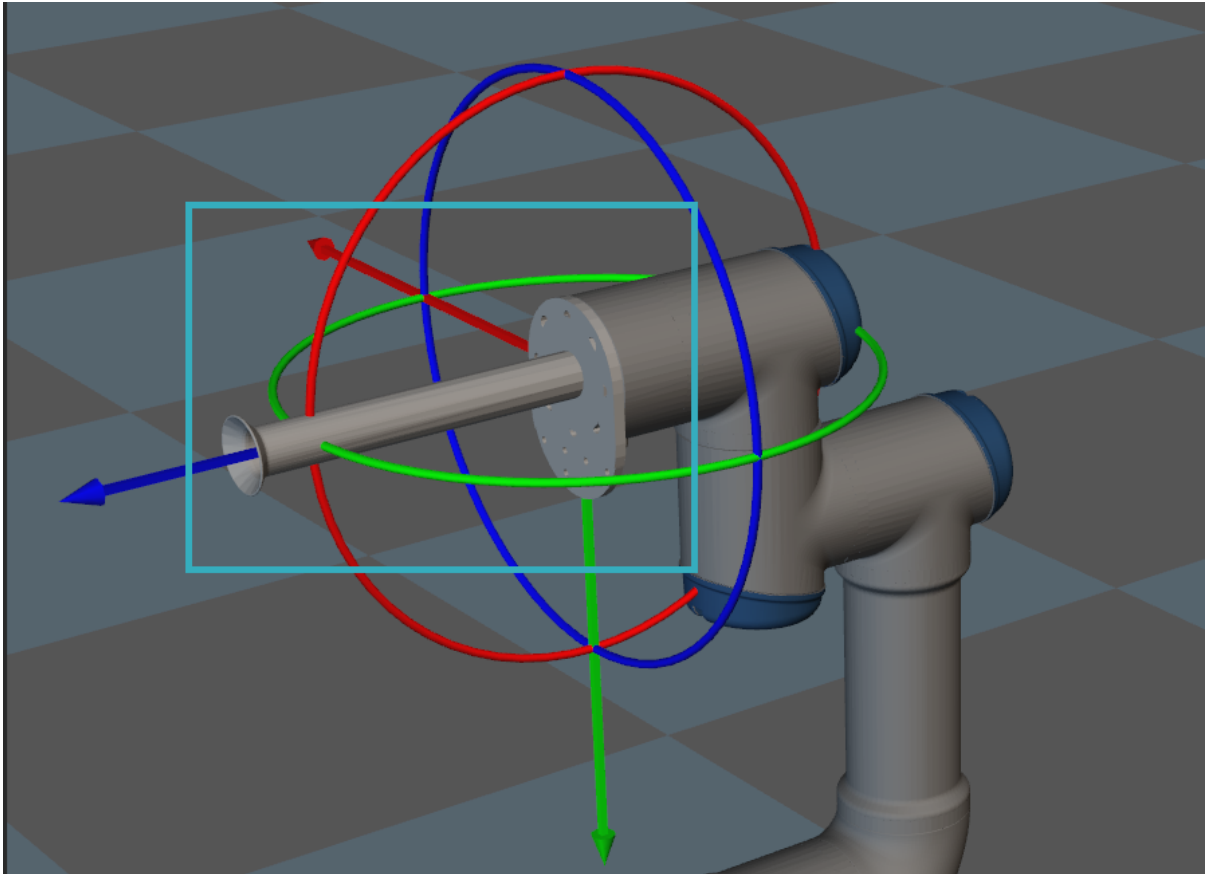
Click on the added collision model in the **Collision Models** list and check whether the position of the model is correct. The figure below shows an example of incorrect position of the collision model. The vacuum gripper is embedded into the robot model.



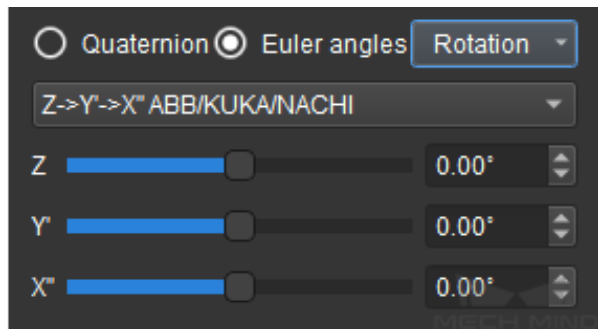
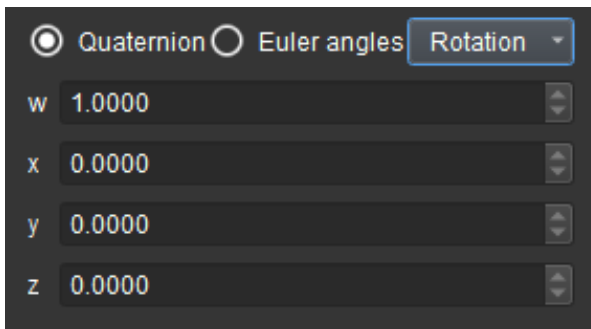
You can adjust the position of the collision model by entering a value or scrolling the bar to adjust the X, Y, and Z coordinate.



The correct position of the collision model is shown below.



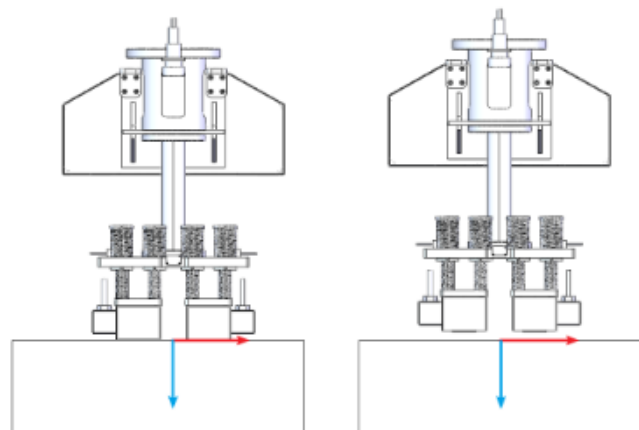
- **Quaternion and Euler Angles:** quaternion and Euler angles are introduced to describe the rotation of the model. You can enter a value or scroll the bar to configure the rotation.



6.1.3 Recommended Method for Using Models

- For projects without specific requirements for displaying, it is recommended to add a collision model only instead of adding a 3D model. The reasons for doing so are as follows.
 - The file size of the project can be reduced. A 3D model for visualization usually contains more details and therefore the file size is larger.
 - The setting process can be simplified. If both the 3D model and collision model are added, you will need to adjust the position of two types of models separately when adjusting the position of the end effector. If only the collision model is added, you can just adjust the position of the collision model.
- The dimensions of the model should be slightly larger than the actual end effector. The reasons are as follows.
 - There is a deviation between the planned path and the actual path that the robot takes.
 - Due to the deviation of the end effector itself and different installation methods, there may be a difference between the model and the actual end effector.
- The length of the model in the Z-direction should be slightly shorter than the actual length.

When a vacuum gripper or cushion spring is used, a certain degree of compression should be allowed in the Z-direction. In order to ensure the accuracy of the collision detection, it is recommended that the length of the model in the Z-direction should be slightly shorter than the actual length.



Note: In the tool reference frame, the Z-direction is defined as the path that the end effector takes to approach the work object. The length of the Z-direction refers to the distance between the TCP (tool center point) and the surface of the robot flange.

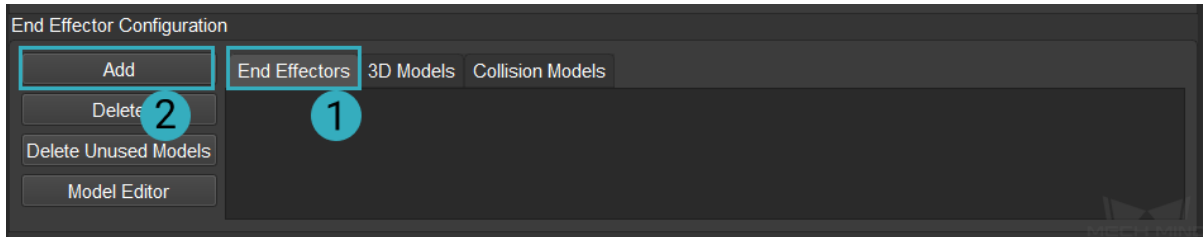
- It is recommended to simplify the model.

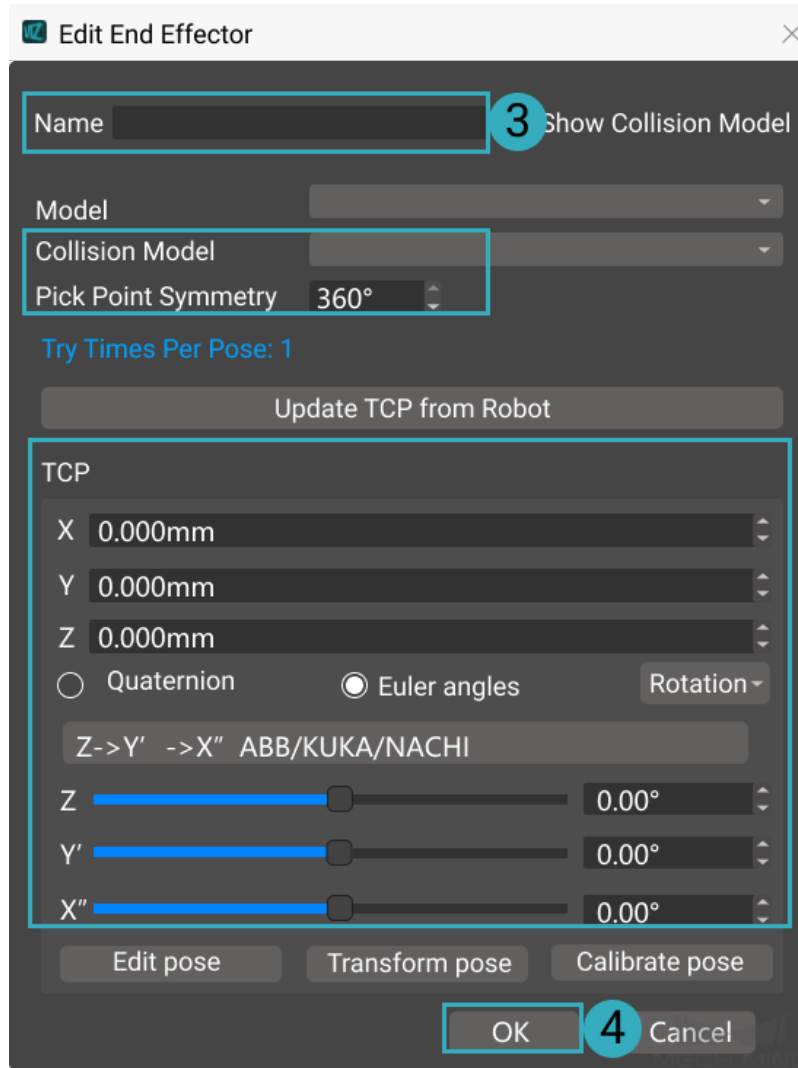
A model with more detailed surface features will increase the computational effort in collision detection. In order to reduce the computational effort, it is recommended to simplify the model. Common practices to simplify models include removing faces inside the model, replacing curved surfaces with flat surfaces, etc.

Please refer to *STL Models Simplification* and *OBJ Models Simplification* for more detailed instructions.

6.1.4 Configure the End Effector Model

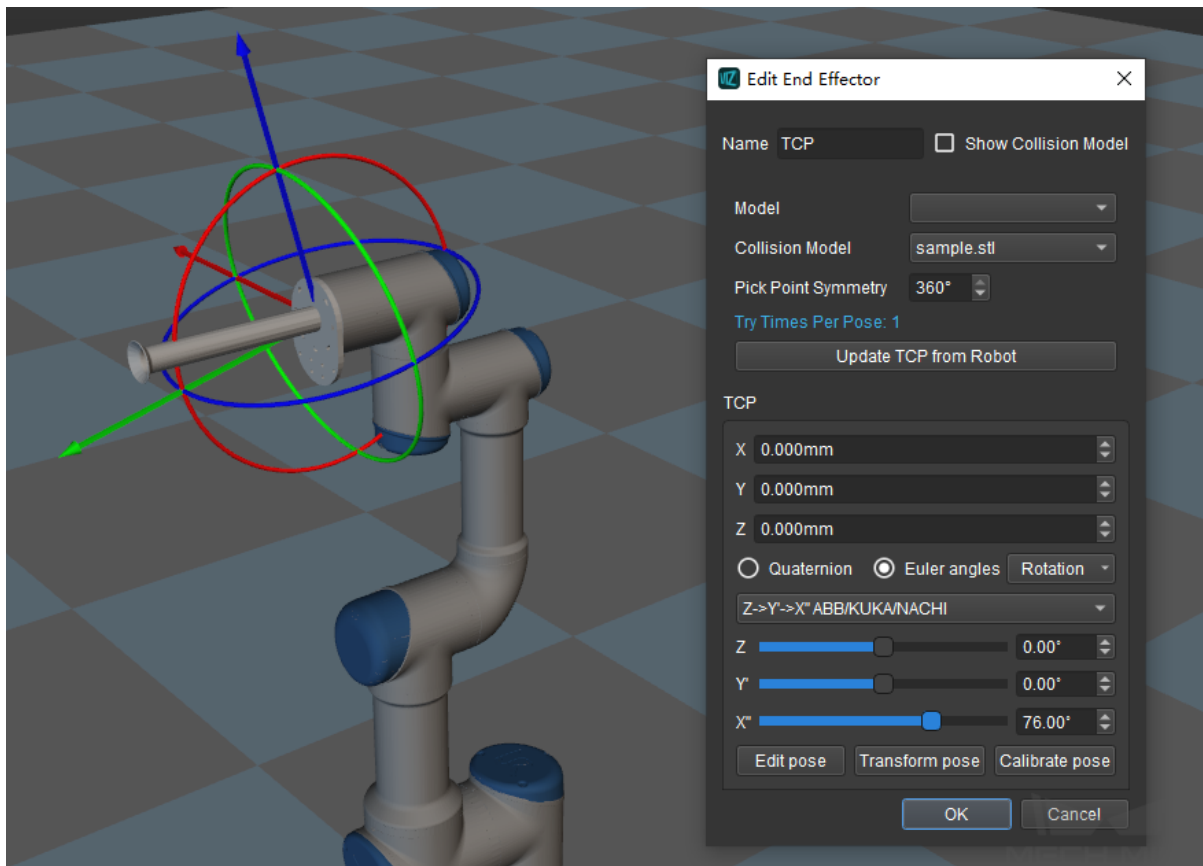
After adding a collision model, you will need to add and configure the end effector model and define a TCP. Click on *End Effectors* in the **End Effector Configuration** panel, then click on *Add*. Configure the parameters in the pop-up **Edit End Effector** window, and click on *OK* to save settings.



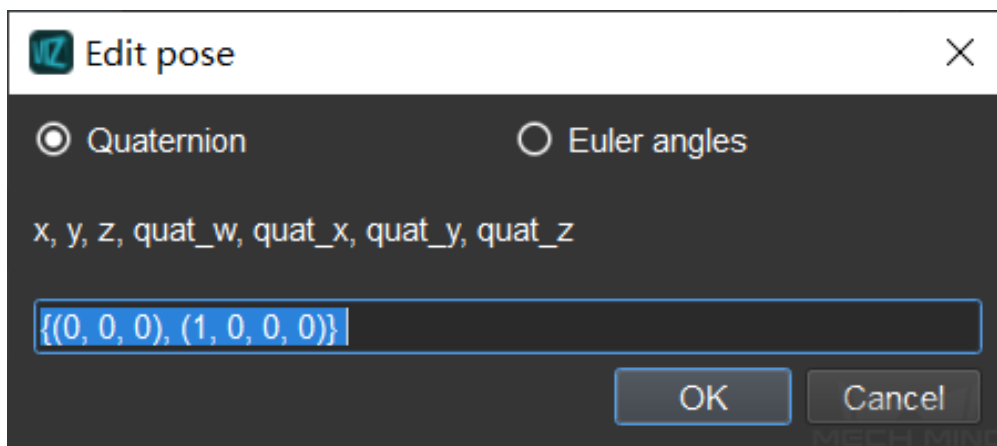


The functions of the buttons and options in the window are described as follows.

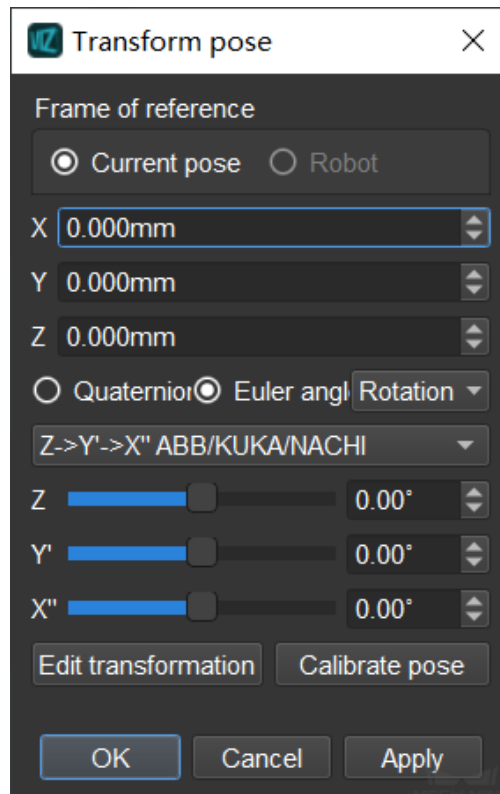
- **Name:** name the end effector.
- **Model:** select the added 3D model in the drop down menu.
- **Collision Model:** select the added collision model in the drop down menu.
- **Pick Point Symmetry:** used for selecting an optimal picking pose. This parameter is related to the way that the end effector picks the object, please set a proper value according to actual situation.
- **Update TCP from Robot:** sends the TCP status of the real robot to Mech-Viz.
- **TCP:** adjust the position of the end effector to the robot flange by adjusting the value of X, Y, and Z.



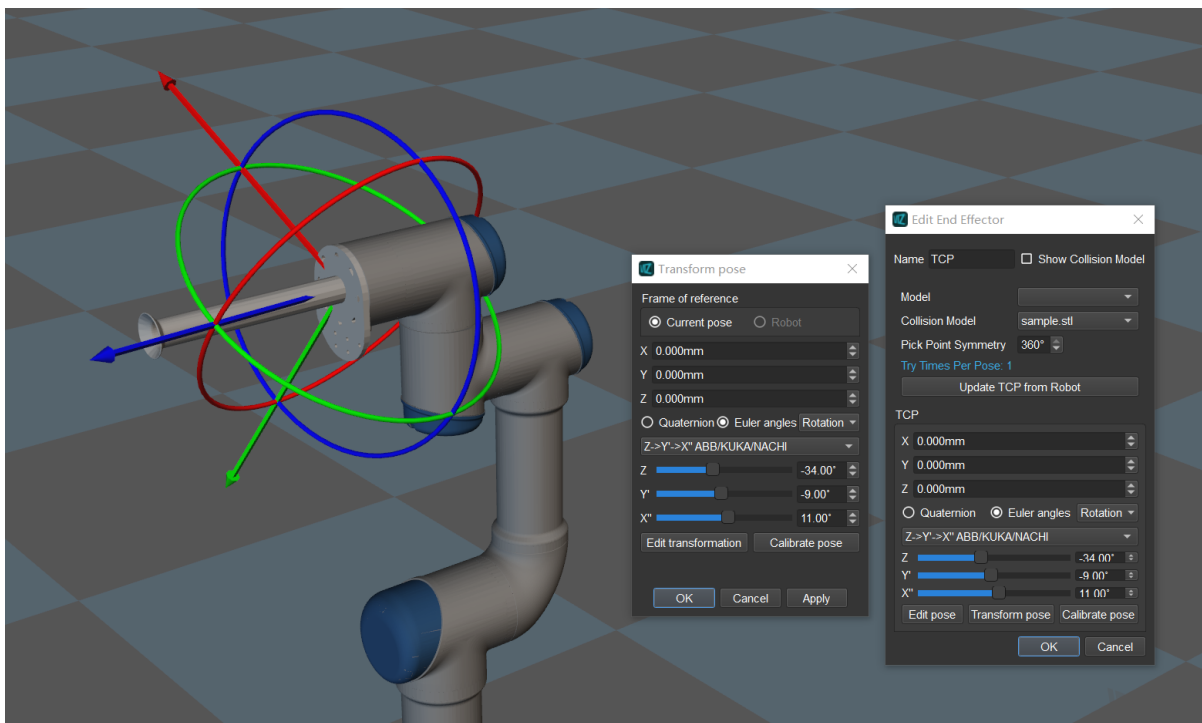
- **Quaternion and Euler Angles:** quaternion and Euler angles are introduced to describe the rotation of the model.
- **Edit Pose:** enter quaternion or Euler angle directly to adjust the position of the end effector.



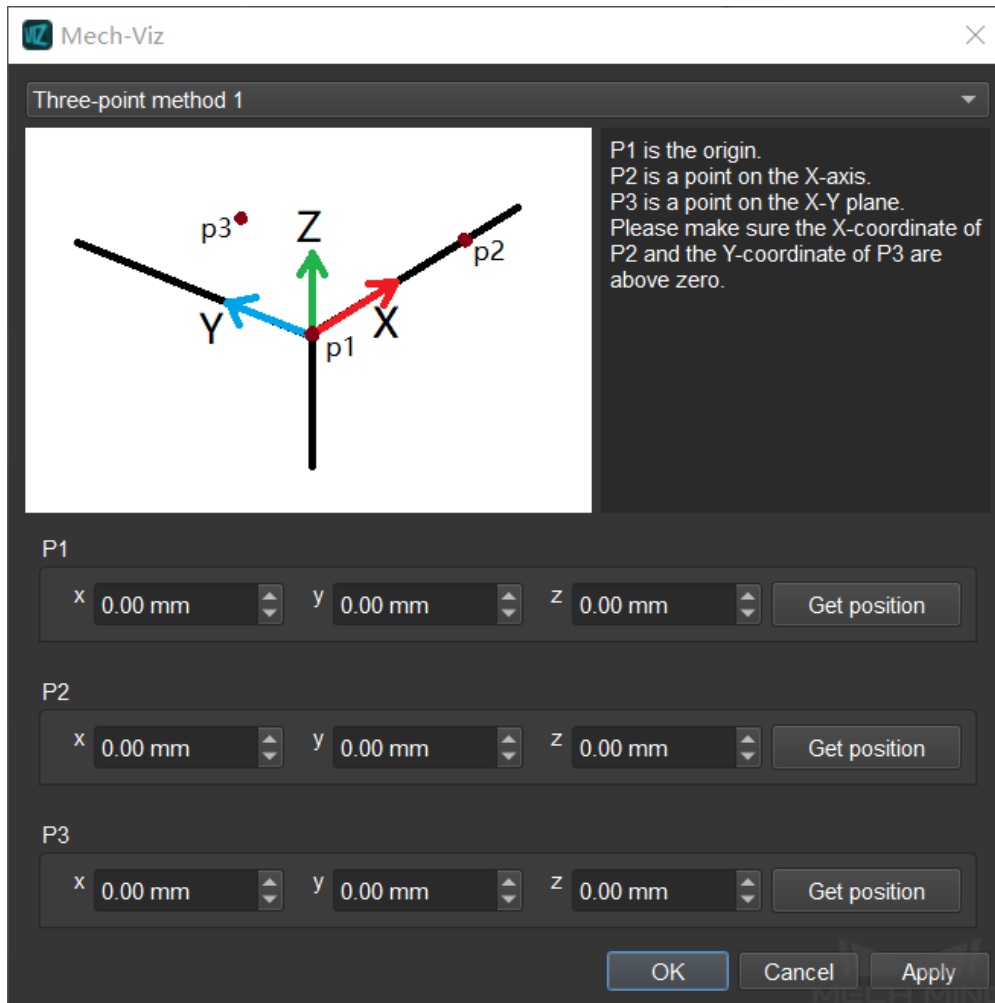
- **Transform Pose:** transform current pose of the TCP to a new one.



You can adjust the position of the TCP by entering a value or scrolling the bar to adjust the X, Y, and Z coordinate.



- **Calibrate Pose:** set the coordinate of P1, P2, and P3 according to the instruction, and then calibrate the end effector pose with three-point method.

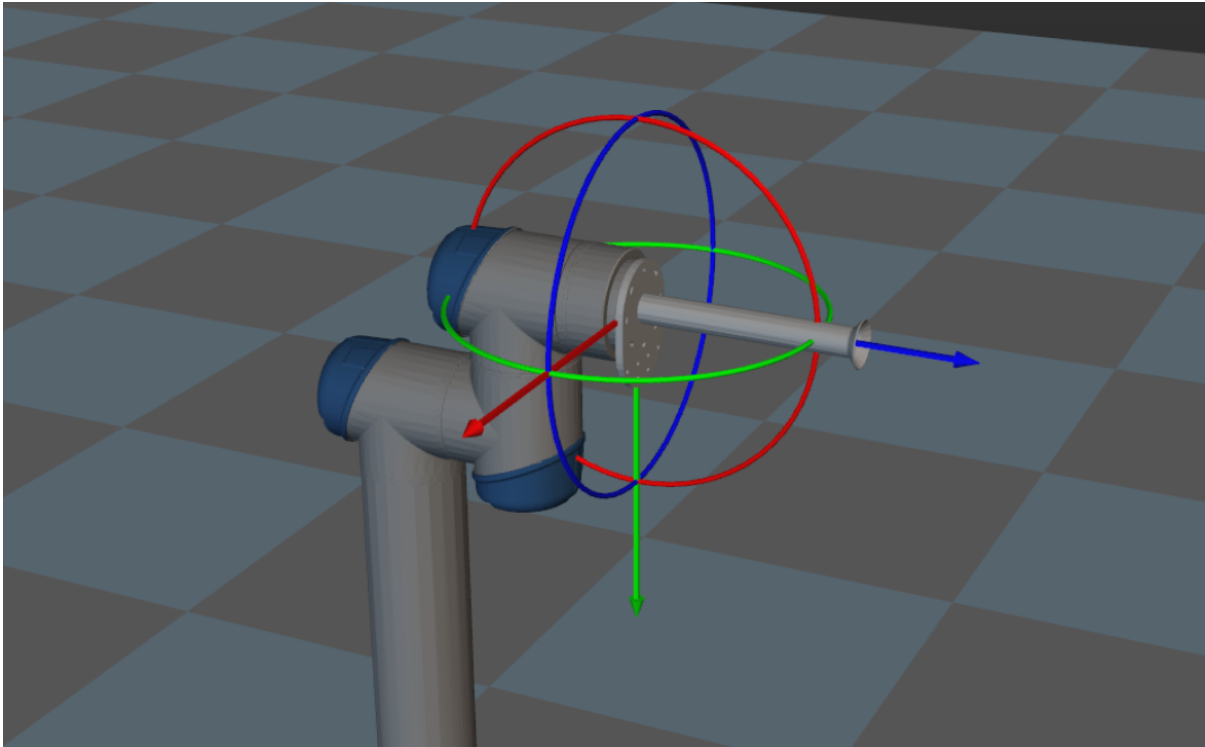


6.1.5 Define the TCP

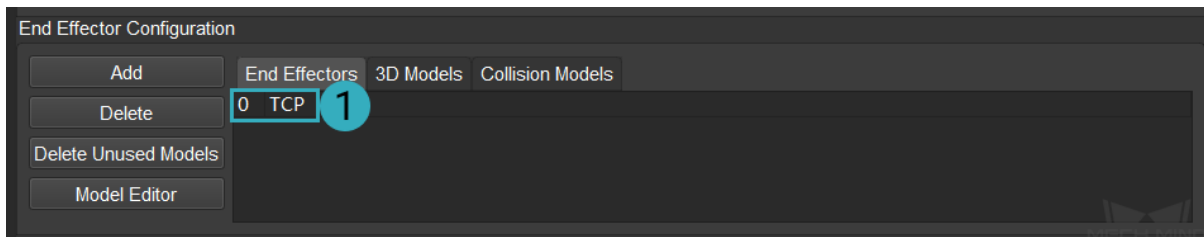
In order to complete tasks such as picking, we usually say that the robot should move to a specific point in space, which actually means its TCP should move to that point.

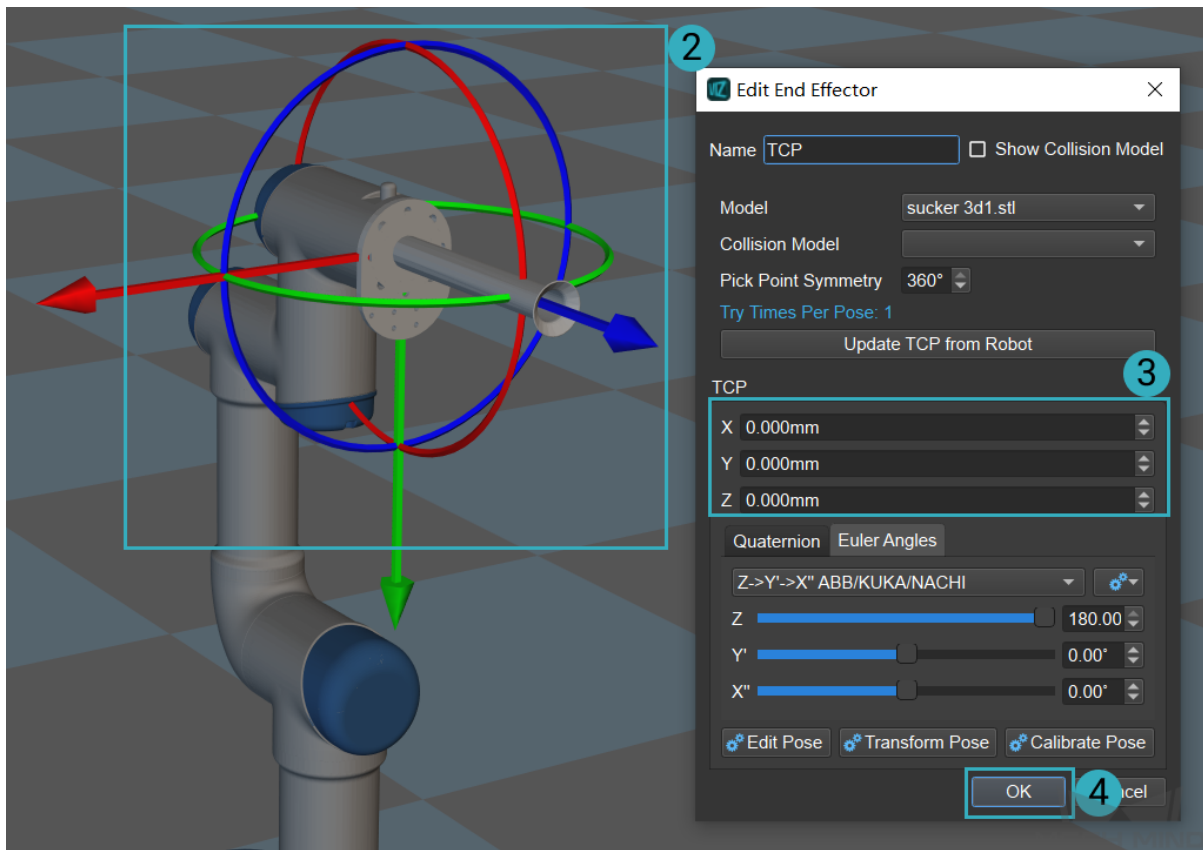
After configuring the end effector, you will need to define the TCP.

The TCP is represented by the center of the rotation manipulator at the end of the robot model. The position of the TCP by default is shown below.

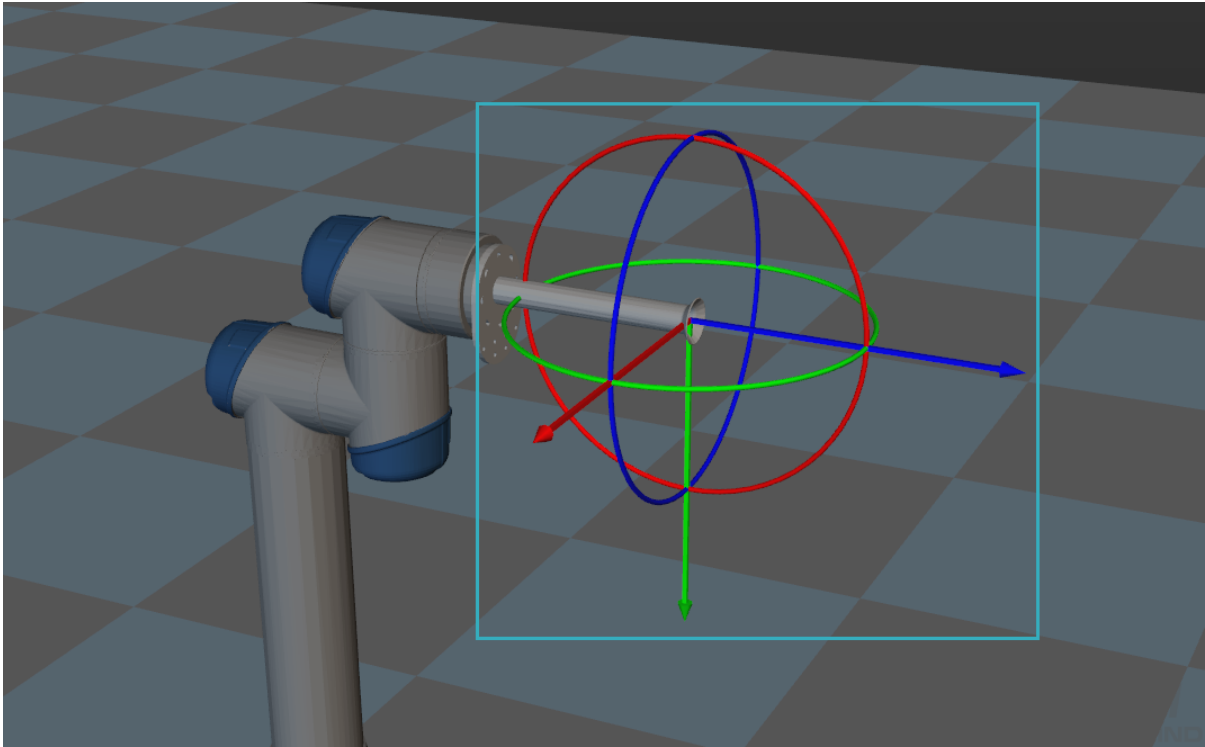


Double click the TCP name in the **End Effectors** list to open the **Edit End Effector** window. Configure the parameters in the **TCP** panel to adjust the position of the TCP to the end of the end effector. After configuration, click on *OK* to save settings.





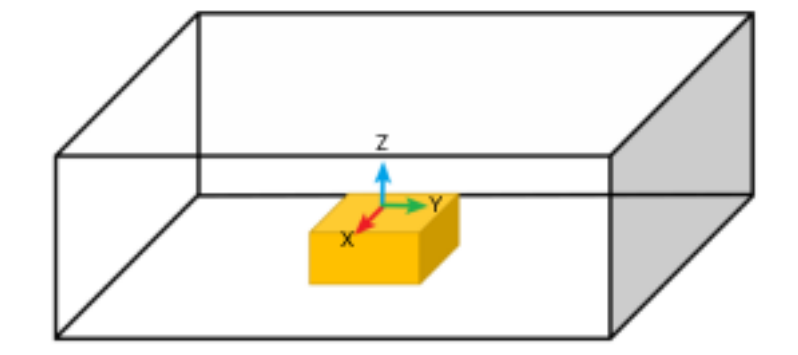
The correct position of the TCP after configuration is shown below.



6.2 Workobjects Configuration

In actual situations, there can be multiple solutions of the possible pick points when picking the same object. In order to increase the number of solutions and the success rate of picking, you can configure the **Rotation Symmetry**, **Picking Relaxation**, and other parameters of the objects in the **Configuration of Objects to Pick** panel.

A **Pick point** represents the position on the object where the robot can pick. The point is determined in the object reference frame, and therefore its position and direction are relative to the object. Usually, the Z-axis in the pick point is pointing upwards, and the blue vector is the Z-axis as shown in the figure below.



This section covers the following topics:

- *Pick Point Selection Strategy*
- *Symmetry of the Pick Point*
- *Picking Relaxation*
- *Optimal Pick Point Selection Strategy*

6.2.1 Pick Point Selection Strategy

Both Mech-Viz and Mech-Vision contributes to the selection of pick point. The main process of selecting a pick point are as follows.

1. Mech-Vision filters and sorts the workobject poses according to the height of the plane in which the object is located, inclination of the object, etc.
 - Sort by either the X, Y, or Z value of the workobject pose
 - Sort by the distance between the candidate pose and reference pose
 - Filter the invalid poses according to the angle between the candidate pose and reference pose
2. In order to avoid collision when picking the object, based on the workobject pose list sent by Mech-Vision, Mech-Viz will take the object rotational symmetry and picking relaxation into consideration when selecting the pick point.
 - Select the pick point according to the order of the workobject pose list sent by Mech-Vision
 - If there is a risk of collision when picking the object whose pose is listed first, Mech-Viz will try another pick point on the same object considering the rotation symmetry, picking relaxation and the optimal picking solution strategy set by the user.
 - If neither pick point is feasible on the object whose pose is listed first, Mech-Viz will select the pick point based on the next workobject pose in the list.

6.2.2 Symmetry of the Pick Point


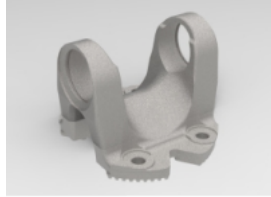
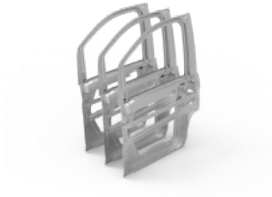
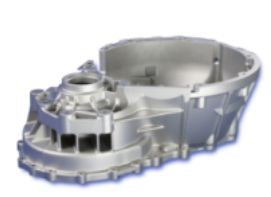
Note:

- The “Symmetry” in this chapter refers to “rotational symmetry”.
 - A pattern with rotational symmetry will coincide with itself after rotating around a certain point by a certain angle. The point is the center of rotation, and the angle it rotates is the angle of rotational symmetry.
-

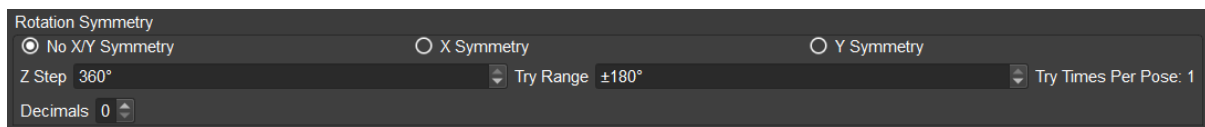
In actual projects, the objects to be picked often have rotational symmetry. The robot can pick or place the same object in different ways, and there is no impact on the result of picking or placing. This section will describe the following 3 situations.

- *Objects Without Rotational Symmetry*
- *Objects Symmetric about the X/Y-Axis*
- *Objects Symmetric about the Z-axis*

Objects Without Rotational Symmetry

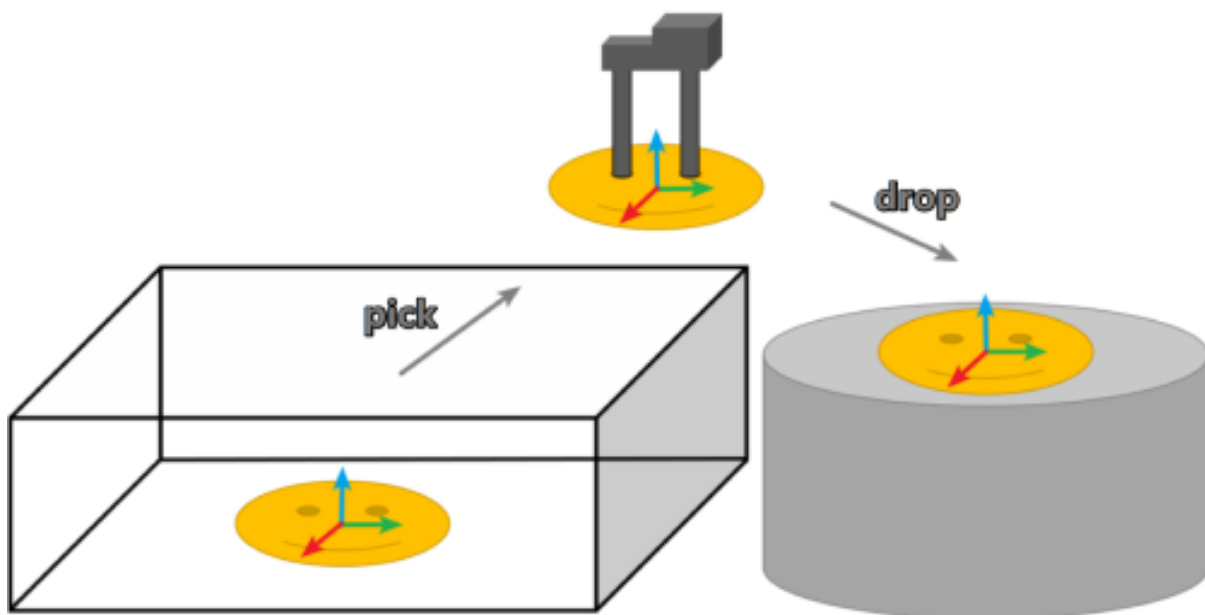
			
Track pad of the bulldozer	Drive shaft flange yoke	Sheet metal of the car door	Transmission bell housing

When the object does not have rotational symmetry, and the end effector is not made of soft material and cannot move in a relatively flexible way, and the objects must be placed accurately, please keep the default settings (No X/Y Symmetry, Z Step 360°, Try Range ±180°).




Application Example:

Both the end effector and workobject in the figure below do not have rotational symmetry, and the object must be placed on the platform in a specified pose.

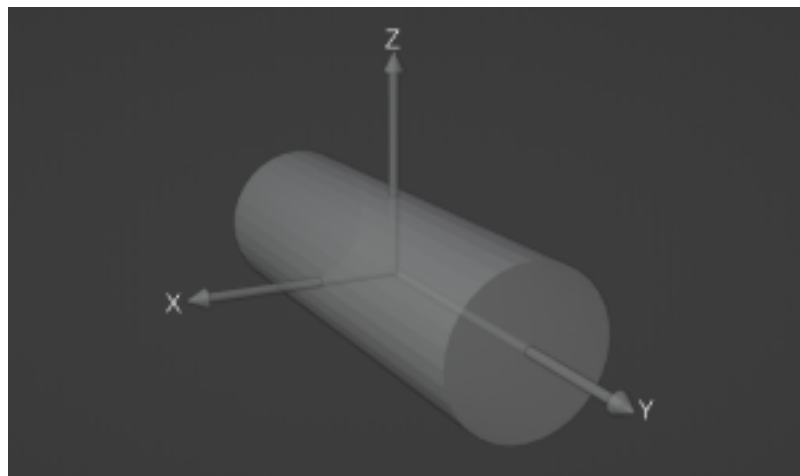


Objects Symmetric about the X/Y-Axis

			
<p>Highly reflective steel rods that are arranged neatly</p>	<p>Steel rods that are stacked randomly</p>	<p>Oil pipe inserts</p>	<p>Bolts</p>

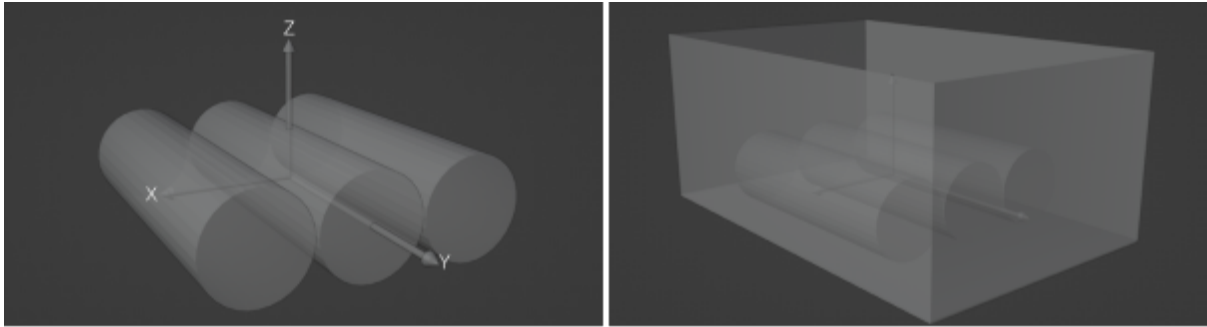
If the end effector is not made of soft material and cannot move in a relatively flexible way, and the object is in the shape of rods or bars with curved surface, the objects are usually symmetric about the X-axis or Y-axis. However, it is not symmetric about the Z-axis.

The workpiece in the figure below is symmetric about the Y-axis by any angle of rotational symmetry (360°).



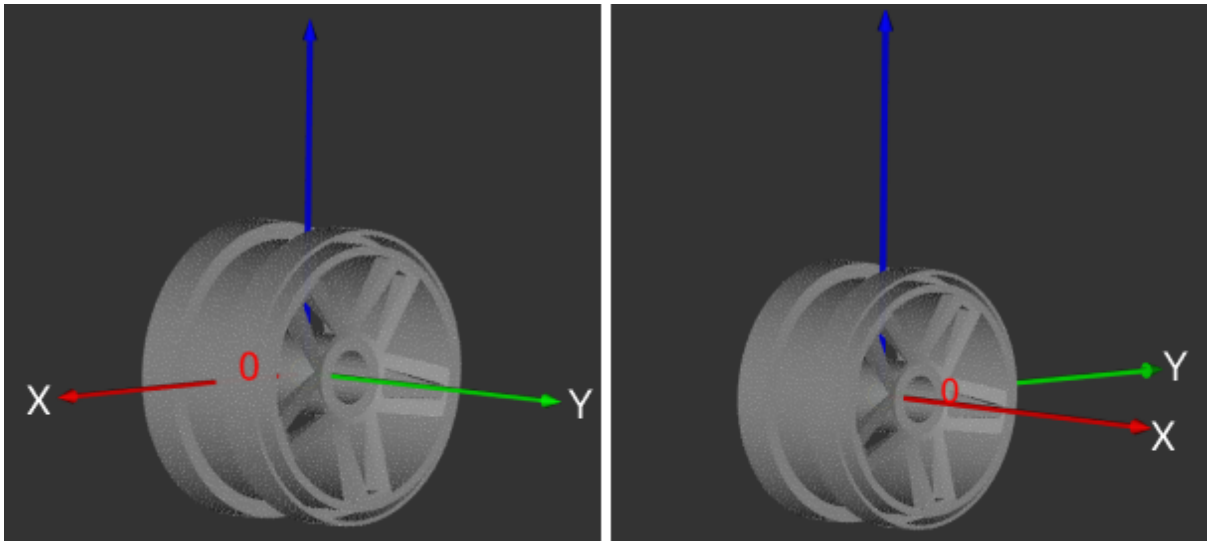
Step: Please set to a proper value. A value that is too small may increase the planning time; while a value that is too large may result in missing the pick point.

Try Range: The value of **Try Range** is usually determined by the exposed area of the side of the workpiece. As shown below, the angle of rotational symmetry of the workpiece is 360°. However, since the workpiece is placed in a bin and surrounded by other workpieces, the pickable range is much less than 360°.



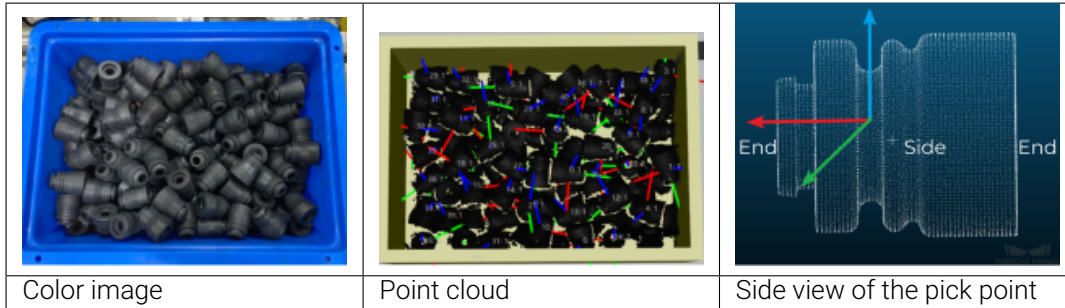
Note: The selection of the axis about which the object rotates depends on the settings of pick point in Mech-Vision.

In the left figure, the Y-axis of the pick point is the axis of rotation. While in the right figure, the X-axis of the pick point is the axis of rotation.

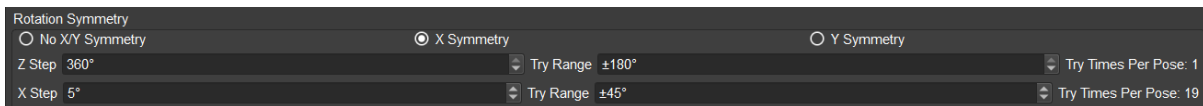


Application Example:

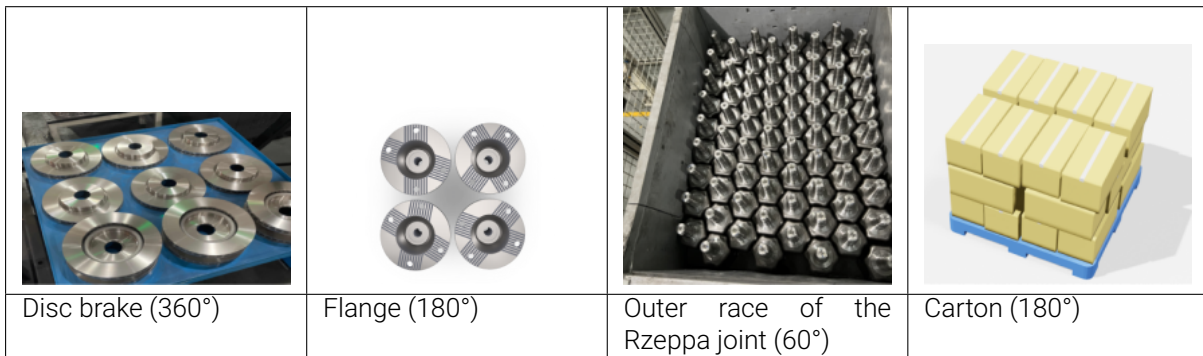
The small workpieces are stacked randomly, and they have an angle of rotational symmetry of 360° about the X-axis of the pick point.



The settings of rotational symmetry are shown below.

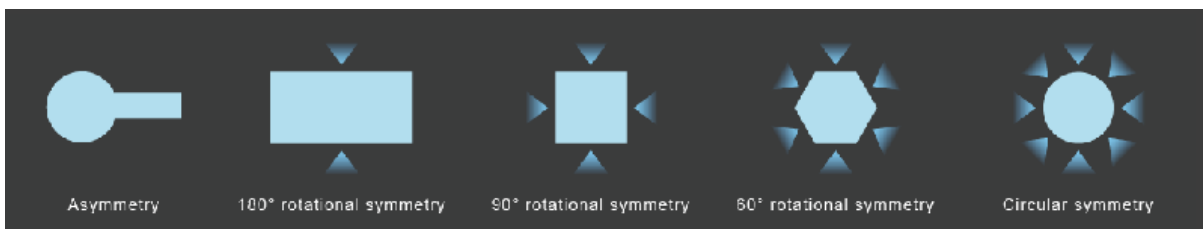


Objects Symmetric about the Z-axis



For objects that are symmetric about the Z-axis, such as cartons, discs, and columns, the end effector can rotate around the Z-axis of the pick point to pick the object, even if the end effector does not have rotational symmetry itself.

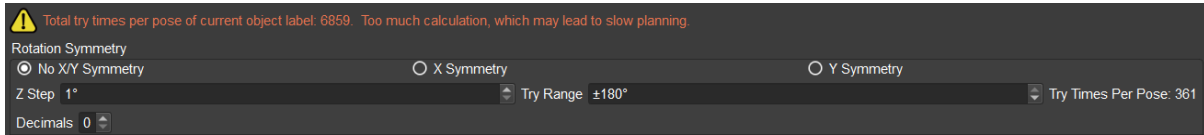
If you consider the patterns in the figure below as top views of the workpieces on a table, all objects have rotational symmetry about the Z-axis.



- Plan both picking and placing (The object should be placed accurately)

Please set the **Z Step** to the angle of rotational symmetry value of the object.

Note: If the object is rotational symmetric by any angle, it is recommended to set the **Z Step** to no less than 10° to avoid a slow planning.



- Plan the picking but not placing (The object does not have to be placed accurately)

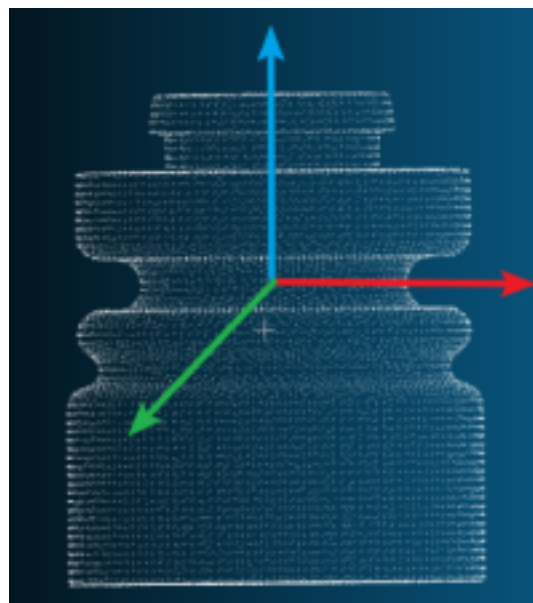
Under this circumstance, the end effector is only required to pick the object accurately, and the rotational symmetry of the object need not to be considered. You can set the **Z Step** according to the scenario where the object is rotational symmetric by any angle.

The example scenarios are as follows:

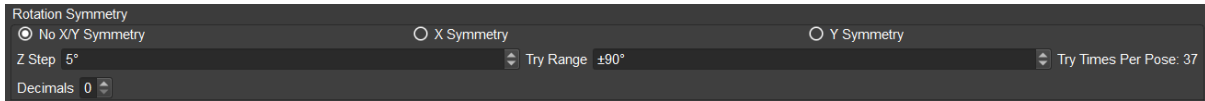
- In novel picking scenario, a suction cup is used to pick the objects and drop them to a specified position or into a bin.
- When an electromagnet is used to pick metal parts, it will pick the metal parts first and then drop them to a specified position or into a bin.

Application Example:

For workpieces that are symmetric about the X-axis or Y-axis, they can be considered as being symmetric about the Z-axis if they will be picked from the top end which is facing upwards.



The settings of rotational symmetry are shown below.



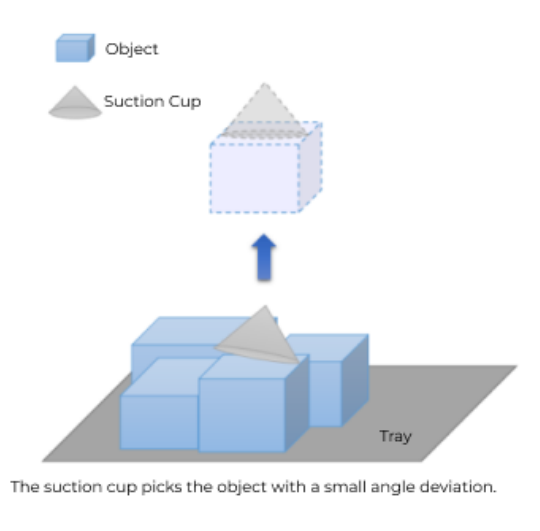
6.2.3 Picking Relaxation

When the end effector is made of soft material or the object is adaptable to various picking methods, a certain angle deviation between the end effector and the object can be allowed when picking. Principally, picking relaxation is a tolerance based on the pick point. By setting the picking relaxation, the robot can utilize such tolerance flexibly to avoid problems such as collisions and singularities.

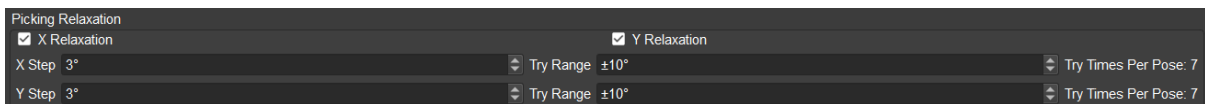
The two common usage scenarios are as follows:

End Effector Made of Soft Material

The figure below shows the scenario where a suction cup is used to pick parcels.

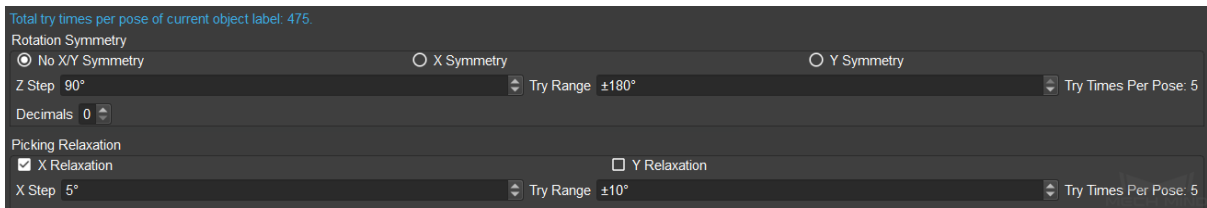


Since the sponge attached to the end of the suction cup is soft, a certain angle deviation on the X-axis direction and Y-axis direction is allowed during picking. Therefore, the **picking relaxation** can be set. According to the allowable deformation of suction cup, it is recommended to set the **Try Range** to $\pm 5^\circ \sim \pm 10^\circ$, and the **Step** to $2^\circ \sim 5^\circ$.



Use both Rotational Symmetry and Picking Relaxation

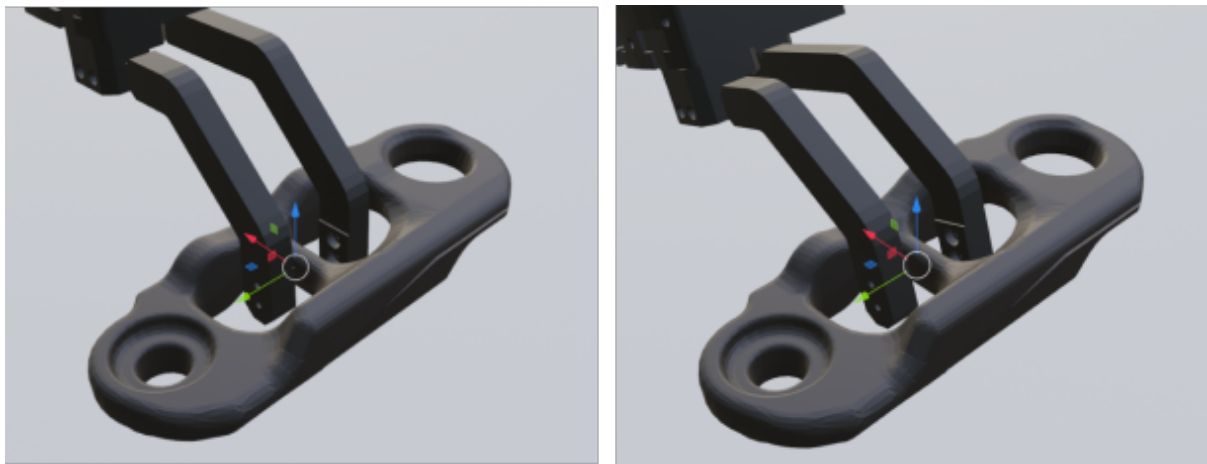
The parcels in the above figure have 90° rotational symmetry about the Z-axis, and therefore you can set the **No X/Y Symmetry** and the corresponding **Step** as shown below.



Attention: During the path planning, the **Try Times Per Pose of Rotational Symmetry** will be multiplied by that of **Picking Relaxation**, and the takt time will be increased if there are too many try times.

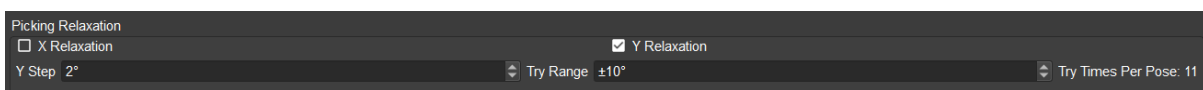
Accurate Placing Not Required

The figure below shows the scenario where an end effector picks an track link. The track link does not have rotational symmetry. When the end effector picks the track link, the end effector can rotate around the Y-axis direction by a certain angle. The Z-axis of the tool pose before picking is vertical down, and the tool pose will change after rotating. The end effector should offset the angle it rotated when placing the object. Since the object does not have to be placed accurately, the change of the tool pose will not affect the final placing result.



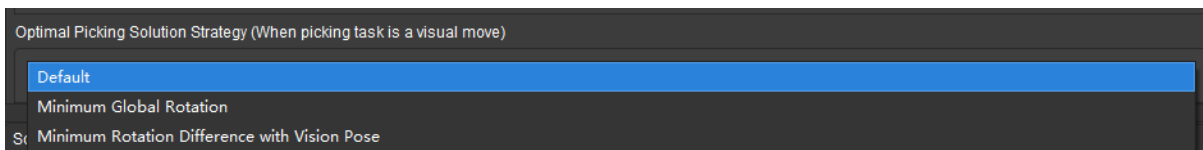


The settings of picking relaxation are shown below.



6.2.4 Optimal Pick Point Selection Strategy

The Optimal Pick Point Selection Strategy panel is shown below.



The descriptions of the modes are as follows.

- **Default:** When the **Default** mode is selected, the **Minimum Global Rotation** strategy will be utilized.
- **Minimum Global Rotation:** When this mode is selected, during the whole pick-and-place process, pick point that causes the minimum rotation degree of the end effector's Z-axis will be defined as the optimal pick point. The advantage of this mode is that it can prevent the end effector from rotating in vain after picking the object, and avoid falling the picked object.
- **Minimum Rotation Difference with Vision Pose:** When this mode is selected, the pick point that has the minimum angle of deviation with the workobject pose sent by Mech-Vision will be defined as the optimal pick point. Assuming that the workobject pose is $[0,0,0,15,0,0]$, and the Z step of the object is set to 60° , the robot will prioritize to pick the workobject with pose $[0,0,0,75,0,0]$ and $[0,0,0,-45,0,0]$.

6.3 Rotational Symmetry and Picking Relaxation

This section introduces basics about rotational symmetry and picking relaxation.

- *The Rotational Symmetry of Tools*
- *The Rotational Symmetry of Workobjects*
- *Picking Relaxation*

6.3.1 The Rotational Symmetry of Tools

The tools specified here are also known as end effectors or grippers. The rotational symmetry is the property a tool has when it looks the same after some rotation by a partial turn. For example:

- A rectangular vacuum gripper's degree of rotational symmetry is 180° .



- A 3-jaw lathe chuck's degree of rotational symmetry is 120° .



The manifestation of rotational symmetry in respect of "parameters"	The tool is symmetrical along the Z axis in the reference frame of the tool center point.
The manifestation of rotational symmetry in respect of "picking"	For an object in a fixed location, multiple picking poses that take the degree of rotational symmetry as a stepsize will lead to the same effect.
The manifestation of rotational symmetry in respect of "placing"	If the tool rotates a certain degrees for picking due to the application of tool's rotational symmetry, the tool should rotate back to the original position when placing the workobject.

6.3.2 The Rotational Symmetry of Workobjects

Workobjects refer to objects to be picked. A workobject's degree of rotational symmetry is the number of distinct orientations in which it looks exactly the same for each rotation. For example:

- A cylinder is rotationally symmetric in any directions.



- A carton's degree of rotational symmetry is 180°.



The manifestation of rotational symmetry in respect of "parameters"	The workobject is symmetric about the X axis	The degree of rotational symmetry
	The workobject is symmetric about the Y axis	
	The workobject is symmetric about the Z axis	
The manifestation of rotational symmetry in respect of "picking"	When the tool cannot rotate	If the workobject rotates a certain angle whose degree is a multiple of the workobject's degree of rotational symmetry, all the workobject poses after rotation can be regarded as the same.
	When the workpiece is in fixed location	If the tool rotates a certain angle whose degree is a multiple of the workobject's degree of rotational symmetry, the results of all the pickings can be regarded as the same.
The manifestation of rotational symmetry in respect of "placing"	If the tool rotates a certain degrees due to the application of workobject's rotational symmetry, the tool does not need to rotate to the original position when placing the workobject. Workobjects that are expected to be placed in a specific pose do not have the rotational symmetry in the strictest sense, and picking relaxation should be applied in this case.	

6.3.3 Picking Relaxation

Picking relaxation describes the degree of flexibility when picking the object. It consists of angle relaxation and distance relaxation.

- Angle Relaxation



- Distance Relaxation

The TCP is regarded as coinciding with the object pose as long as the TCP stays within a certain distance on a specific plane related to the object.

The manifestation of picking relaxation in respect of "parameters"	X Relaxation	The threshold and stepsize of the relaxation
	Y Relaxation	
	Z Relaxation	
The manifestation of picking relaxation in respect of "picking"	If the workobject is in fixed location, the robot will try picking poses that will not lead to collisions within the relaxation threshold.	
The manifestation of picking relaxation in respect of "placing"	Placing according to the workobject pose	The deviation in the workobject's pose due to the application of picking relaxation should be compensated at workobject placing.
	Placing according to the TCP	The tool is moved to a specified position, while the workobject's pose does not need to be compensated.

COLLISIONS

In applications such as machine tending and (de)palletizing, avoiding collisions between the robot and other objects is crucial to keeping the project going non-stop.

Mech-Viz can detect possible collisions and plan the robot path accordingly, so that the resulting path is collision-free.

If collision cannot be avoided, Mech-Viz will stop the project and display the detected collision in the 3D simulation area, so that you can adjust the project accordingly.

	Robot Links	Scene Objects	End Effector (Surface)	Point Cloud	Detected Cuboid	Picked Cuboid	Placed Cuboid
Robot Links	Collision is detected when there is any contact.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.
Scene Objects	Collision is detected when there is any contact.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.
End Effector (Surface)	Collision is detected when there is any contact.	Collision is detected when there is any contact.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.
Point Cloud	Only allow collision contacts smaller than configured threshold. 3 mm ³	Do not check collision or collision doesn't exist.	Only allow collision contacts smaller than configured threshold. 0, 0 cm ²	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.
Detected Cuboid	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.
Picked Cuboid	Collision is detected when there is any contact.	Collision is detected when there is any contact.	Do not check collision or collision doesn't exist.	Allow collision contacts larger than configured threshold meanwhile in decreasing trend in a trajectory, and allow collision contacts smaller than configured threshold as well. 200 mm ³	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.
Placed Cuboid	Collision is detected when there is any contact.	Do not check collision or collision doesn't exist.	Collision is detected when there is any contact.	Do not check collision or collision doesn't exist.	Do not check collision or collision doesn't exist.	Collision is detected when there is any contact.	Do not check collision or collision doesn't exist.

Collision occurs between two objects. In the **Collisions** tab, you can set between which two types of objects you'd like to check for collisions, such as point cloud and other objects. The above chart displays

the collisions checked in the current project, and the color of each cell corresponds to the specific collision detection settings for the object pair.

Check the following chapter to learn about **Computation Settings** which affects speed of collision detection.

7.1 Computation Settings

In **Computation Settings**, you can set the extent of collision computation to perform and plan history to save. The options affect the speed of project execution.

Different combinations of options have the following effects:

<div style="background-color: #333; color: #ccc; padding: 2px;">Stop computing the current solution when the collision contact exceeds the threshold (for continuous operations) ▾</div> <div style="background-color: #333; color: #ccc; padding: 2px;">Save in plan history (slow, for parameter adjustment and collision visualization) ▾</div>	Full calculation performed and full record retained, slow, used for debugging the project.
<div style="background-color: #333; color: #ccc; padding: 2px;">Compute complete collision contacts of each candidate solution (for parameter adjustment and collision visualization) ▾</div> <div style="background-color: #333; color: #ccc; padding: 2px;">Save in plan history (slow, for parameter adjustment and collision visualization) ▾</div>	Partial calculation performed and full record retained; clicking on collision-related records in Plan History will visualize the calculated part of collision.
<div style="background-color: #333; color: #ccc; padding: 2px;">Compute complete collision contacts of each candidate solution (for parameter adjustment and collision visualization) ▾</div> <div style="background-color: #333; color: #ccc; padding: 2px;">Do not save in plan history (for continuous operations) ▾</div>	Full calculation performed and partial record retained; clicking on collision-related records in Plan History does not visualize the collision.
<div style="background-color: #333; color: #ccc; padding: 2px;">Stop computing the current solution when the collision contact exceeds the threshold (for continuous operations) ▾</div> <div style="background-color: #333; color: #ccc; padding: 2px;">Do not save in plan history (for continuous operations) ▾</div>	Patial calculation performed and partial record retained; fast, used during actual production.

Check the following chapters to learn about the configurations of collision detection.

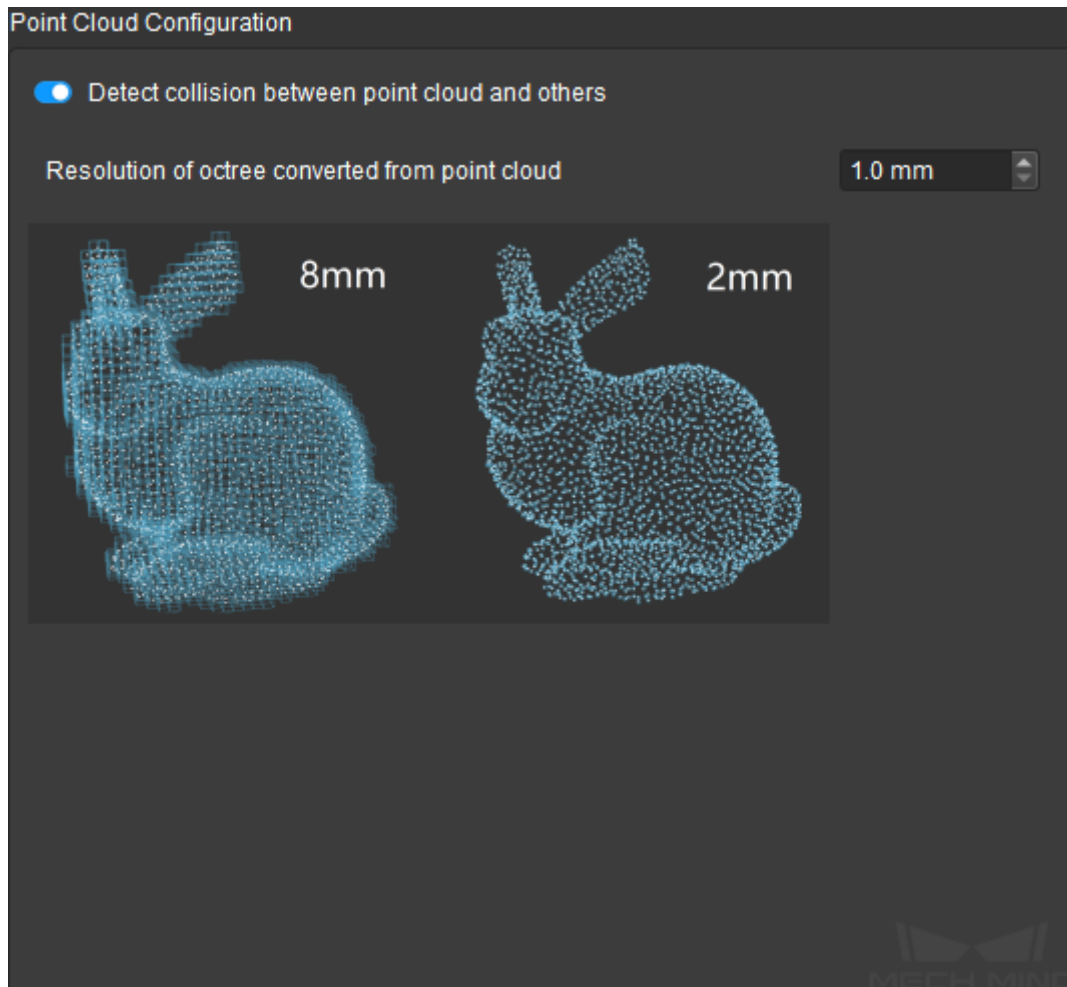
7.2 Collision Detection Configuration

In this window, you can configure the following objects involved in collision detection:

7.2.1 Point Cloud Configuration

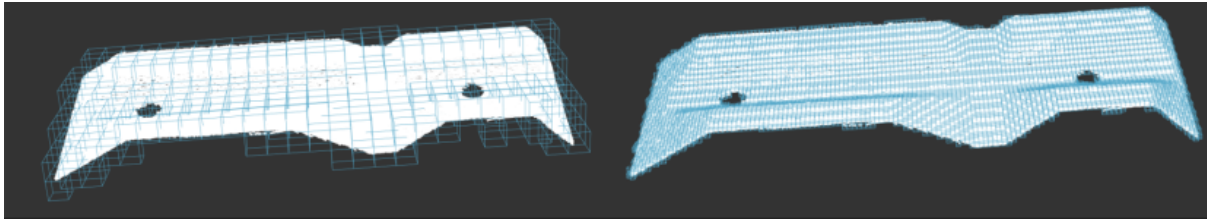
The point cloud used in collision detection usually includes the workobjects and some surrounding objects (such as the bin).

This point cloud is sent from Mech-Vision using the `send_point_cloud_to_external_service` Step. Before you enable **Detect collision between point cloud and others**, please make sure that this Step is used in the corresponding Mech-Vision project.



A point cloud is made of points, which are zero-dimensional and thus cannot be used for collision detection. Therefore, Mech-Viz converts the point cloud to an octree structure first. In the octree structure, the three-dimensional space is divided into little cubes, each centered around a point in the point cloud. Collisions with the point cloud are recognized when other objects come into contact with these cubes.

By adjusting **Resolution of octree converted from point cloud**, you can change the edge length of each little cube. In the figure below, the edge lengths are 10 mm (left) and 2 mm (right).



For the same point cloud:

- The shorter the edge length of each cube, the more cubes, the more accurate the collision calculation, and the longer it takes to calculate collisions.
- The longer the edge length of each cube, the fewer cubes, the less accurate the collision calculation, but the shorter it takes to calculate collisions.

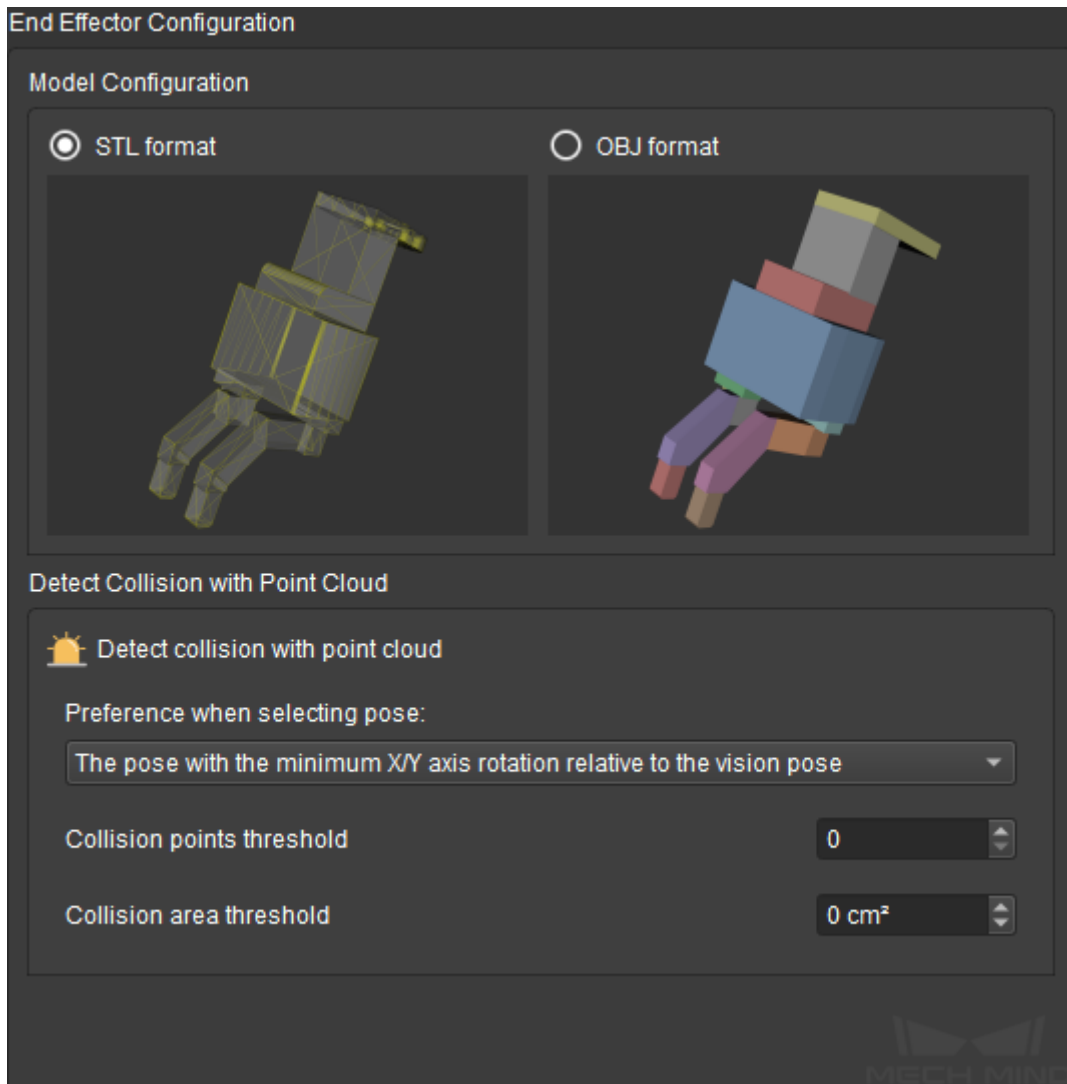
Therefore, the edge length should be set based on your requirements for collision calculation accuracy and project execution speed.

After you enable **Detect collision between point cloud and others**, Mech-Viz will check the collisions between the point cloud and robot tool (end effector), in addition to the default object pairs.

	Robot Links	Scene Objects	End Effector (Solid)	Point Cloud	Detected Non-Cuboid	Picked Non-Cuboid	Placed Non-Cuboid
Robot Links							
Scene Objects							
End Effector (Solid)							
Point Cloud			5 mm ³				
Detected Non-Cuboid							
Picked Non-Cuboid							
Placed Non-Cuboid							

7.2.2 End Effector Configuration

In this pane, you can select the format of robot tool model used and configure the detection of collisions between the robot tool and point cloud.

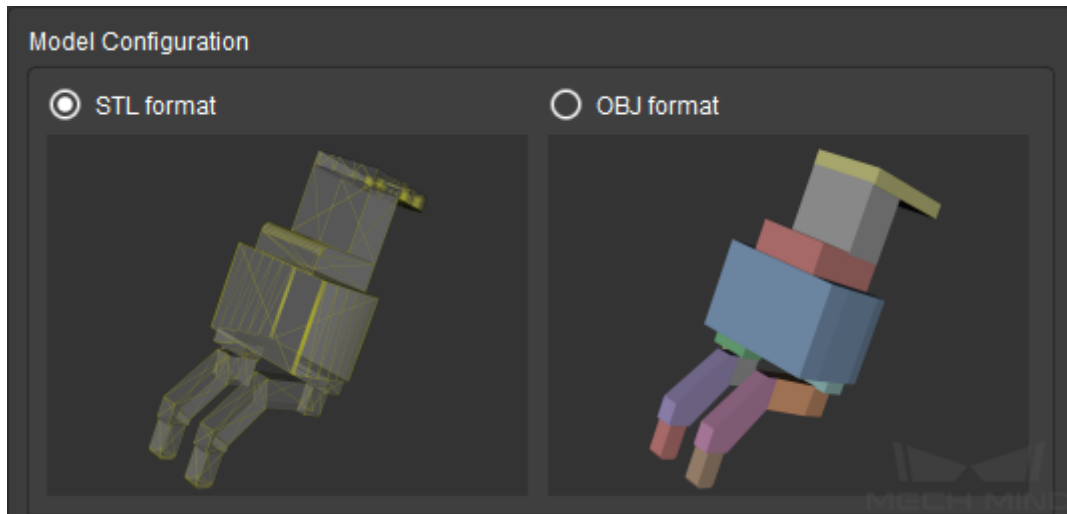


Note: Collisions between the following object pairs involving the robot tool are checked by default and cannot be configured:

- Robot links and robot tool
- Scene objects and robot tool

Configurations made in this pane only takes effect if you have added a robot tool model. For instructions, please refer to [Add an Model](#).

Model Configuration



Select the format of robot tool model you added, STL or OBJ.

The different formats affect the detection of collisions between the tool and point cloud. With OBJ format, collisions with the interior of the robot tool can be detected.

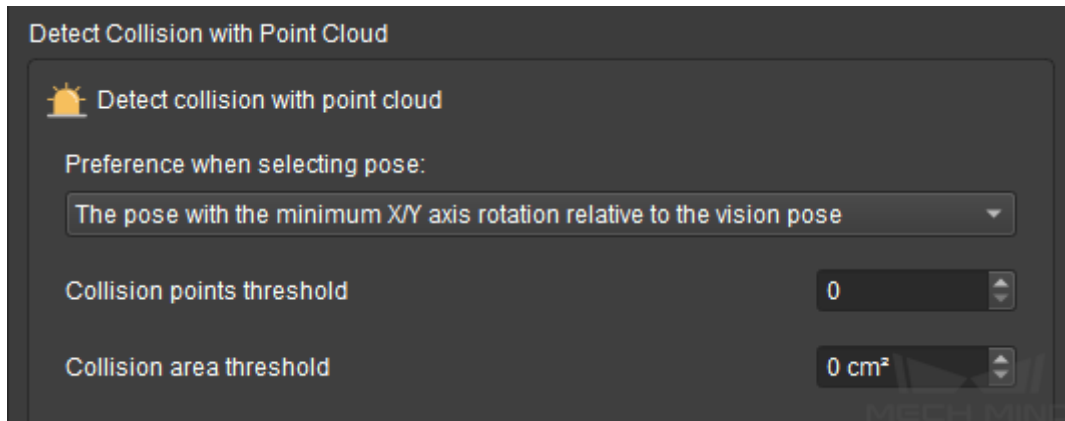
	STL	OBJ
Model type	Mesh (linked triangles)	Assembly of convex polyhedra (solid)
Industrial 3D modeling	Widely used	Rarely used
Collision detection	Only collisions with the surface can be detected	Precise and efficient
Parameter adjustment	difficult for complicated applications	Relatively easy and highly adaptable

For more detailed information on the two file formats, please refer to [Basic Principle of Collision Detection](#).

Attention: Starting from version 1.6.0, OBJ models that contain concave polyhedra cannot be used for collision detection anymore. OBJ models used in collision detection must be fully made of convex polyhedra.

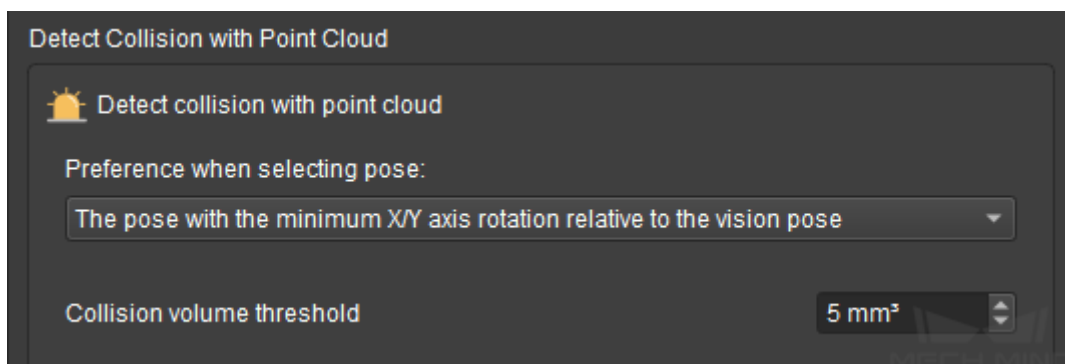
Parameters in the **Detect Collision with Point Cloud** section is automatically adapted to your selection in the **Model Configuration** section. The following sections introduces the parameters by model format.

Detect Collision with Point Cloud - STL



- **Preference when selecting pose:** candidate workobject target poses are generated by rotating from the received vision pose according to *Configuration of Objects to Pick* in the **Tools and Workobjects** tab. Your selection here affects how the target pose is determined.
 - **The pose with the minimum X/Y axis rotation relative to the vision pose:** select the first candidate pose (i.e., the least rotated about the X- or Y-axis) whose collision is below the set threshold as the target pose. Once selected, computation of candidate poses is stopped.
 - **The pose with the minimum collision within the X/Y axis rotation range:** compute all candidate poses and select the pose with the least collision as the target pose.
- **Collision points threshold:** set how many points in the point cloud are allowed to collide with the robot tool at a target pose.
The default value is 0 (no contact between the robot tool and point cloud is allowed).
- **Collision area threshold:** set how large an area of the tool model is allowed to collide with the point cloud at a target pose.
The default value is 0 (no contact between the robot tool and point cloud is allowed).

Detect Collision with Point Cloud - OBJ



- **Preference when selecting pose:** candidate workobject target poses are generated by rotating from the received vision pose according to *Configuration of Objects to Pick* in the **Tools and Workobjects** tab. Your selection here affects how the target pose is determined.
 - **The pose with the minimum X/Y axis rotation relative to the vision pose:** select the first candidate pose (i.e., the least rotated about the X- or Y-axis) whose collision is below the set threshold as the target pose. Once selected, computation of candidate poses is stopped.
 - **The pose with the minimum collision within the X/Y axis rotation range:** compute all candidate poses and select the pose with the least collision as the target pose.
- **Collision volume threshold:** set how large a volume of the tool model is allowed to collide with the point cloud at a target pose.

The default value is 5 (contact of 5 mm³ or less between the robot tool and point cloud is allowed).

Tips for Parameter Adjustment

The key factor to consider is: whether it is okay for the robot tool to touch or even move other workobjects during picking.

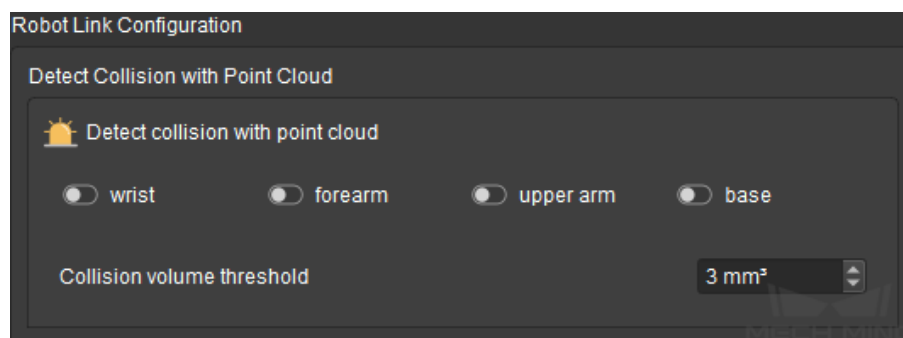
Consider picking up jelly beans with chopsticks. Even if the chopsticks touch or move other jelly beans when picking up a jelly bean, these other jelly beans will not be adversely affected (e.g., broken).

In cases like this, such as picking up randomly piled small metal workpieces from a bin, you can set higher collision thresholds and allow some collision to occur, so that a higher rate of successful picking can be guaranteed. However, you should note that the picked workobject might also move when the tool touches/moves other workobjects.

In cases where collisions might break the workobjects or have other adverse effects on the application, you should decrease the collision thresholds to ensure the planned path is collision-free.

7.2.3 Robot Link Configuration

In this pane, you can select the part(s) of the robot to be taken into consideration during the collision detection.

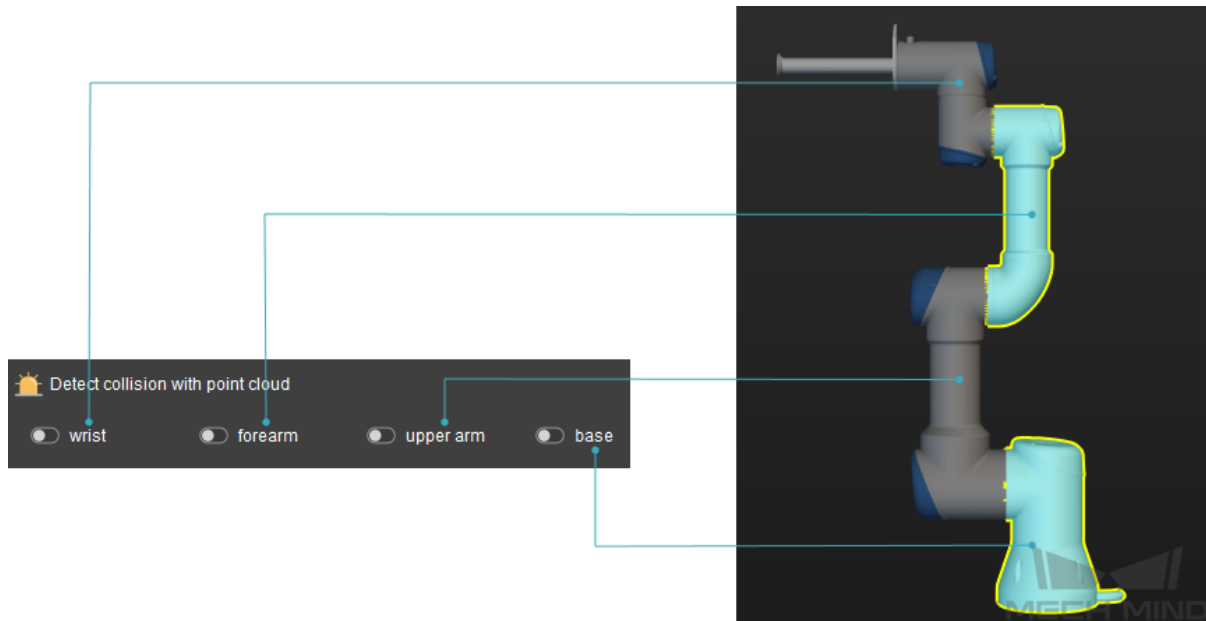


Note: This pane is configurable only after you enable **Detect collision between point cloud and others**.

Note: Collisions between the following object pairs involving the robot links are checked by default and cannot be configured:

- Robot links and robot links
- Robot links and scene objects
- Robot links and robot tool

The robot is divided into four parts: wrist, forearm, upper arm and base. You can enable the part(s) that you'd like to check for collisions. Enabled parts are highlighted in the 3D simulation area.



After you enable any of the robot parts, the collisions between robot links and point cloud will be checked.

	Robot Links	Scene Objects	End Effector (Surface)	Point Cloud	Detected Non-Cuboid	Picked Non-Cuboid	Placed Non-Cuboid
Robot Links	Enabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
Scene Objects	Enabled	Enabled	Disabled	Disabled	Disabled	Disabled	Disabled
End Effector (Surface)	Enabled	Enabled	Enabled	Disabled	Disabled	Disabled	Disabled
Point Cloud	0 mm ³	Disabled	0, 0 cm ²	Enabled	Enabled	Disabled	Disabled
Detected Non-Cuboid	Enabled	Enabled	Enabled	Enabled	Enabled	Disabled	Disabled
Picked Non-Cuboid	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Disabled
Placed Non-Cuboid	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled

- **Collision volume threshold:** set how large a volume of each robot link is allowed to collide with the point cloud at a target pose.

The default value is 0 (no contact between robot links and point cloud is allowed).

Note: If you have enabled **Detect collision between picked object and others**, the collisions between robot links and picked workobjects are also checked. However, **Collision volume threshold** only

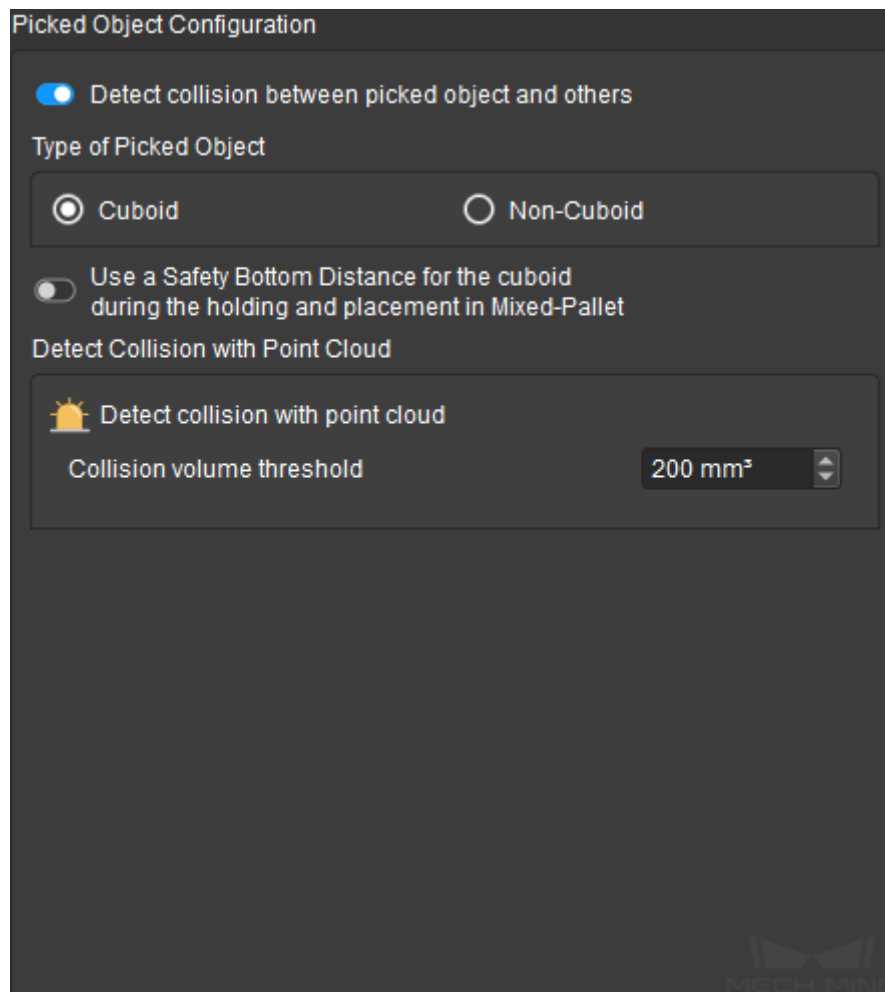
applies to the collisions between robot links and point cloud. That is, any contact between robot links and picked workobjects is regarded as collision occurred.

	Robot Links	Scene Objects	End Effector (Surface)	Point Cloud	Detected Cuboid	Picked Cuboid	Placed Cuboid
Robot Links	Light Blue	Grey	Grey	Grey	Grey	Grey	Grey
Scene Objects	Light Blue	Light Grey	Grey	Grey	Grey	Grey	Grey
End Effector (Surface)	Light Blue	Light Grey	Light Grey	Grey	Grey	Grey	Grey
Point Cloud	Orange (0 mm ³)	Light Grey	Orange (0, 0 cm ²)	Light Grey	Grey	Grey	Grey
Detected Cuboid	Light Blue	Light Grey	Light Grey	Light Grey	Light Grey	Light Grey	Light Grey
Picked Cuboid	Light Blue	Light Grey	Light Grey	Yellow (0 mm ³)	Light Grey	Light Grey	Light Grey
Placed Cuboid	Light Blue	Light Grey	Light Grey	Light Grey	Light Grey	Light Blue	Light Grey

7.2.4 Picked Object Configuration

In this pane, you can enable and configure the detection of collisions between picked workobjects and other objects.

After enabling **Detect collision between picked object and others**, you can select the type of picked workobjects, and the parameters are automatically adapted to your selection.



Picked workobjects are divided into two types, cuboid and non-cuboid.

The following sections introduce each type and the correlated parameters separately.

Cuboid Picked Workobjects

Picked Object Configuration

Detect collision between picked object and others

Type of Picked Object

Cuboid
 Non-Cuboid

Use a Safety Bottom Distance for the cuboid during the holding and placement in Mixed-Pallet

Safety Bottom Distance of cuboid 5 mm

Detect Collision with Point Cloud

Detect collision with point cloud

Collision volume threshold 200 mm³

Collision detection requires object models. Cuboid workobject models can be automatically generated by Mech-Viz if object dimensions in the format of **Size3DList** are received from Mech-Vision.

Mech-Vision Steps that can output object dimensions include `calc_poses_and_dimensions_of_rectangles`, `calc_poses_and_dimensions_from_planar_point_clouds`, `read_object_dimensions`, etc. The generated object dimensions are sent to Mech-Viz through the `procedure_out` Step.

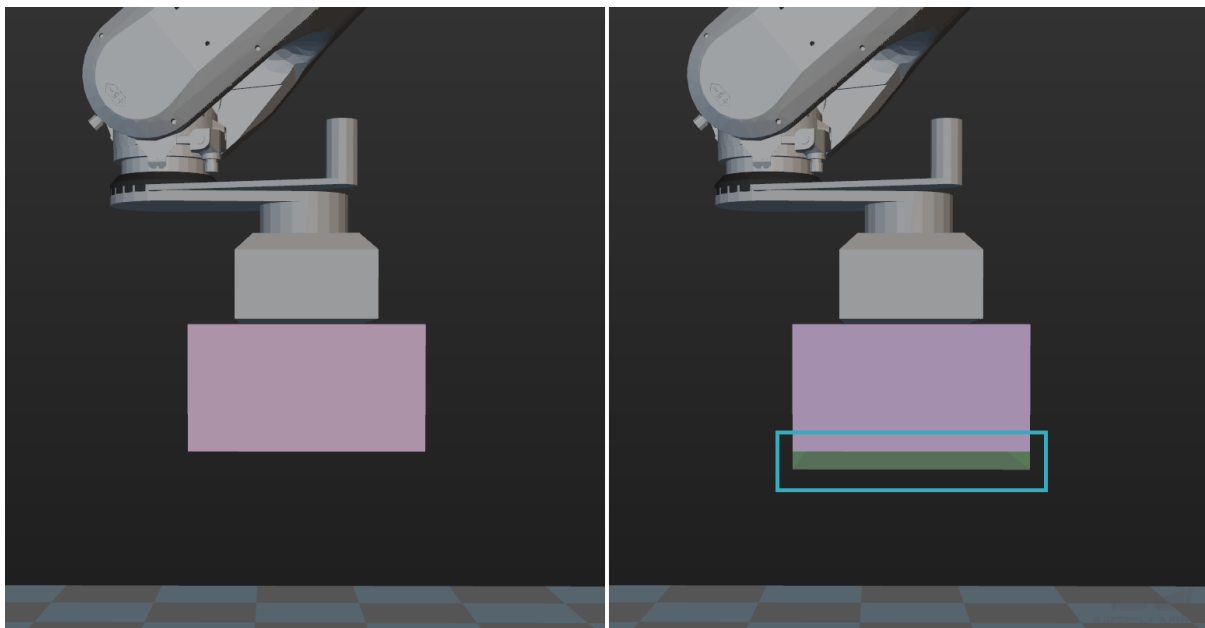
Safety Bottom Distance

In palletization applications, the carton dimensions input to Mech-Viz may not be accurate enough, and the carton itself might deform when it's pick up. These may result in inaccuracies in collision detection.

The safety bottom distance setting adds the set value to the obtained height dimension of the cuboid model during collision detection. This way, on the planned path, the bottom surface of the carton is in fact farther away from the whatever is below it, avoiding collision with palletized cartons.

To set a safety bottom distance, please enable **Use a Safety Bottom Distance for the cuboid during the holding and placement in Mixed-Pallet**, and then adjust the value for **Safety Bottom Distance of cuboid**. The default value is 5 mm.

The setting is visualized in the 3D simulation area when you simulate or run the project. The following figure shows the picked cuboid model without a safety bottom distance (left) and with a safety bottom distance of 30 mm (right).



Detect Collision with Point Cloud (Cuboid)

Here you can set the collision volume threshold with point cloud.

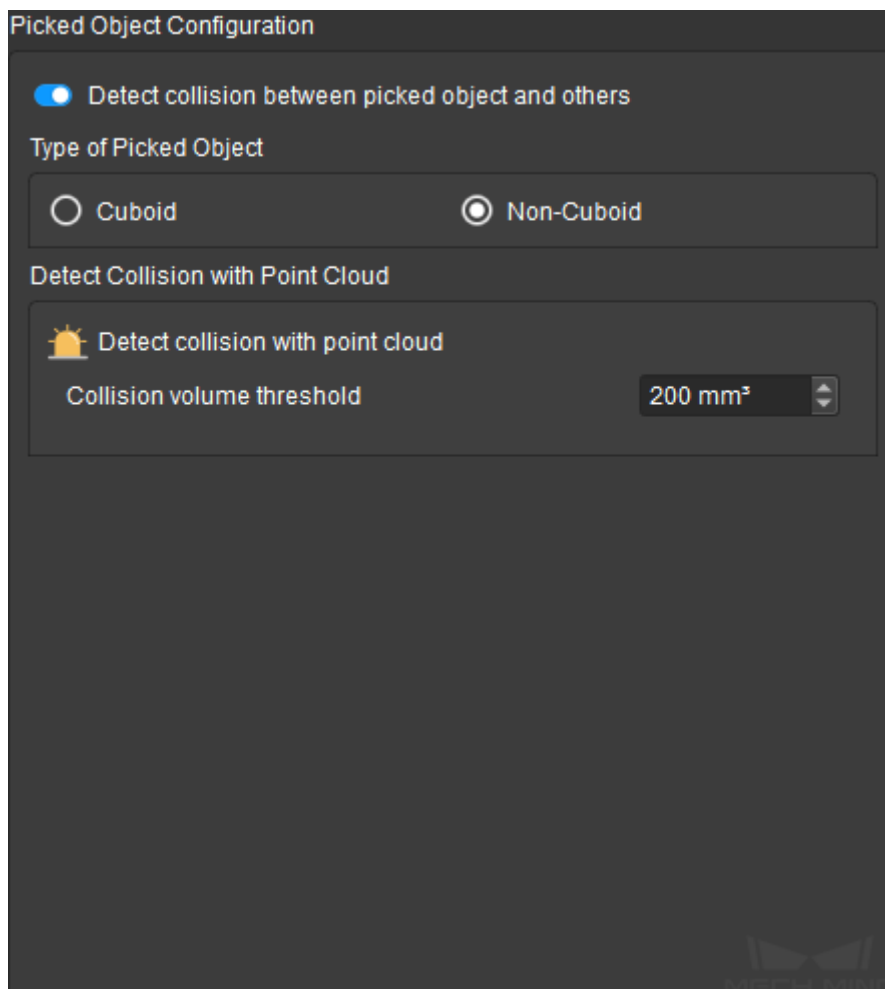
- **Collision volume threshold:** set how large a volume of the picked workobject is allowed to collide with the point cloud at a target pose.

The default value is 0 (no contact between the picked workobject and point cloud is allowed).

With **Detect collision between point cloud and others (Cuboid)** and **Detect collision between picked object and others**, the following object pairs are checked for collisions.

	Robot Links	Scene Objects	End Effector (Surface)	Point Cloud	Detected Cuboid	Picked Cuboid	Placed Cuboid
Robot Links							
Scene Objects							
End Effector (Surface)							
Point Cloud			0, 0 cm ²				
Detected Cuboid							
Picked Cuboid	5 mm	5 mm		0 mm ³			
Placed Cuboid						5 mm	

Non-Cuboid Picked Workobjects



Collision detection requires object models. Non-cuboid workobject models must be prepared beforehand and added to the Mech-Viz project folder.

- Move the STL and BINVOX files of the workobject to the **collision_models** folder under the project folder.

Note: The file names must be identical to the pose label of the same object set in Mech-Vision.

- In Mech-Vision send_point_cloud_to_external_service Step, check **Send Object Information**.

Detect Collision with Point Cloud (Non-Cuboid)

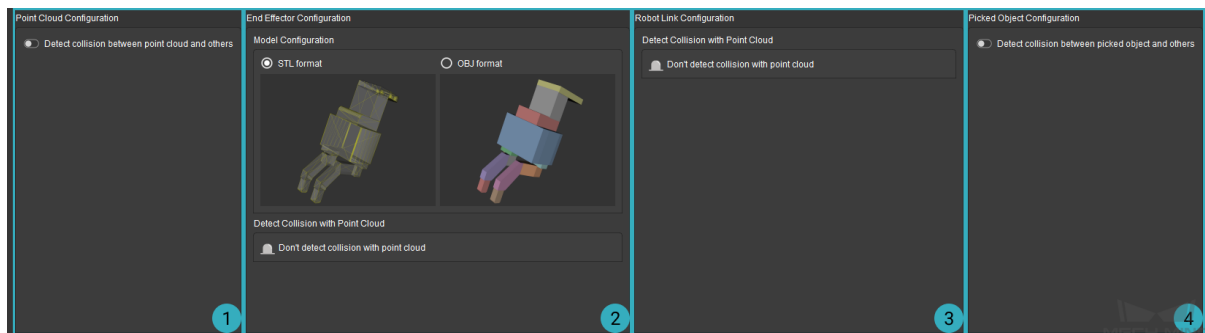
Here you can set the collision volume threshold with point cloud.

- **Collision volume threshold:** set how large a volume of the picked workobject is allowed to collide with the point cloud at a target pose.

The default value is 0 (no contact between the picked workobject and point cloud is allowed).

With **Detect collision between point cloud and others (Non-Cuboid)** and **Detect collision between picked object and others**, the following object pairs are checked for collisions.

	Robot Links	Scene Objects	End Effector (Surface)	Point Cloud	Detected Non-Cuboid	Picked Non-Cuboid	Placed Non-Cuboid
Robot Links							
Scene Objects							
End Effector (Surface)							
Point Cloud			0, 0 cm ²				
Detected Non-Cuboid			0 mm ³				
Picked Non-Cuboid		0		0 mm ³	0 mm ³		
Placed Non-Cuboid							



Please note that the following object pairs are checked for collision by default and cannot be configured:

- Robot links and robot links
- Robot links and scene objects
- Robot links and robot tool (end effector)
- Scene objects and robot tool

	Robot Links	Scene Objects	End Effector (Surface)	Point Cloud
Robot Links				
Scene Objects				
End Effector (Surface)				
Point Cloud				

PLAN HISTORY

You can check the detailed planning information and error description in the **Plan History** panel, which facilitates to debug the project.

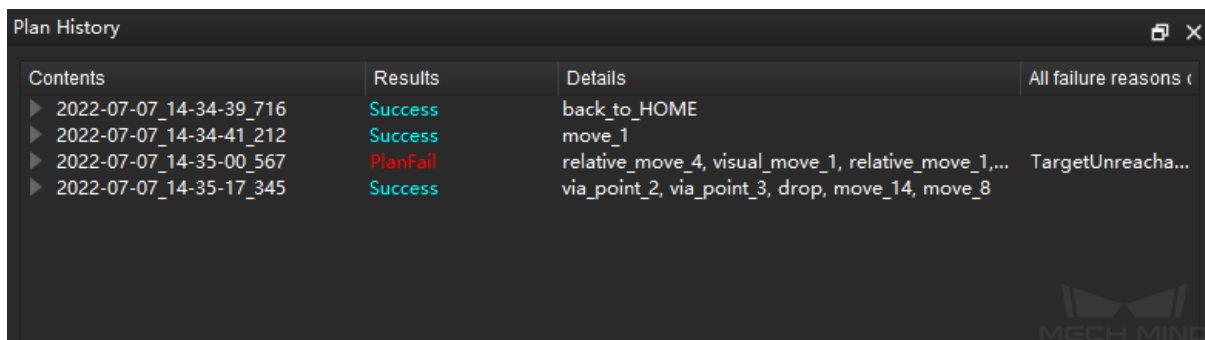
This section covers the following topics:

- *Overview*
- *Visualize Collision*
- *Details*

8.1 Overview

8.1.1 Functions

When running a project in Mech-Viz, if a collision occurs, the point in the path is not achievable, or other error occurs, you can check the detailed cause in **Plan History**.



The overall plan history is laid out in a tree structure, which contains every node in the whole process of path planning. Comparing to the log, plan history contains more details of every node (especially when there is an error) in the process.

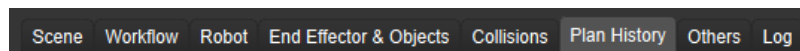
- Some errors are serial errors, which means that if the error occurs at one of the nodes, the final plan result will fail. For example, a certain joint position is set wrong, and the robot cannot move to the planned point.

- Some errors are parallel errors, which means that the final plan result will still succeed if the planning at one of the parallel nodes is successful. For example, when Mech-Viz attempts to pick with the pick points sent by Mech-Vision, if either one of the pick points works, the workobject can be picked successfully.




Hint: You can focus on the nodes where errors occur when running the project.

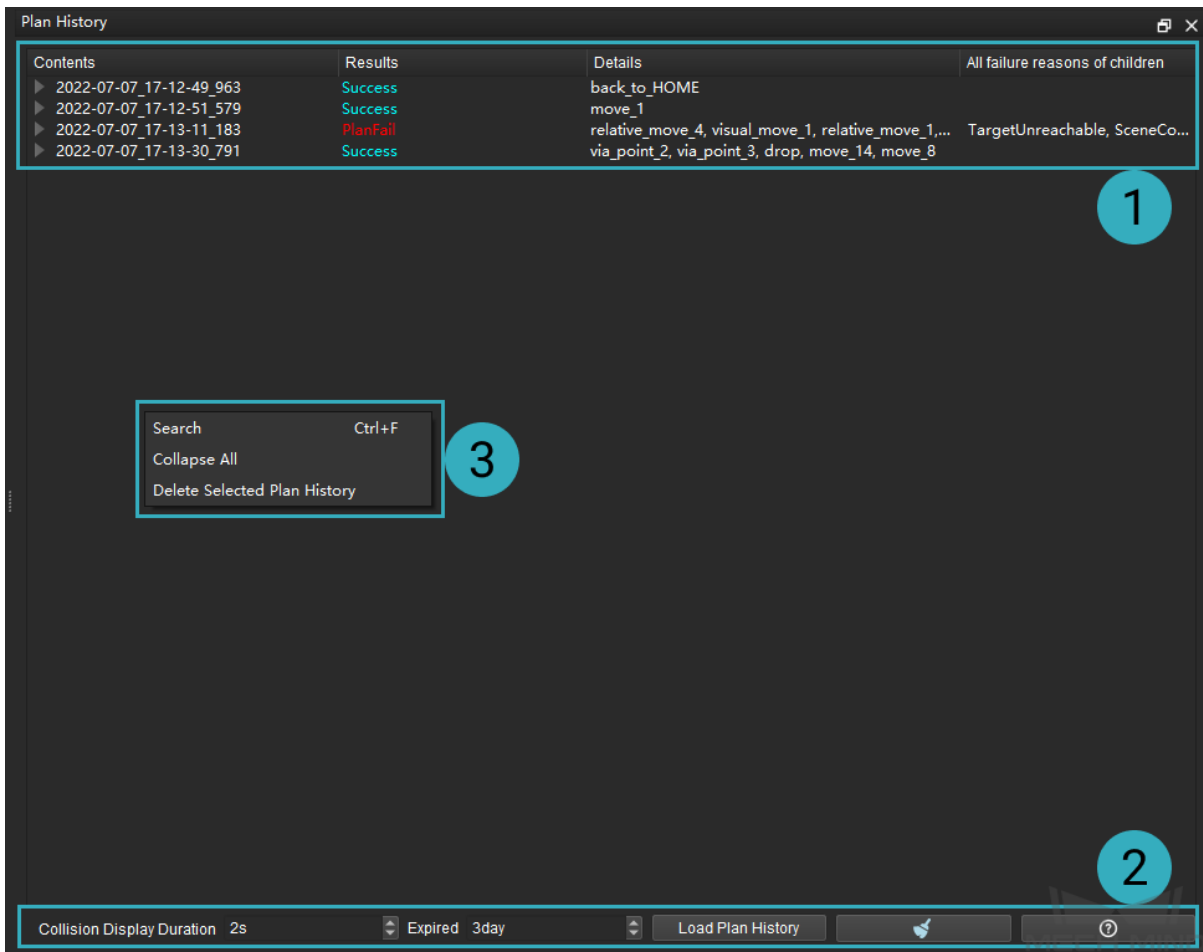
8.1.2 Interface Introduction

Click on the **Plan History** tab in the lower right corner of the interface to open the panel.



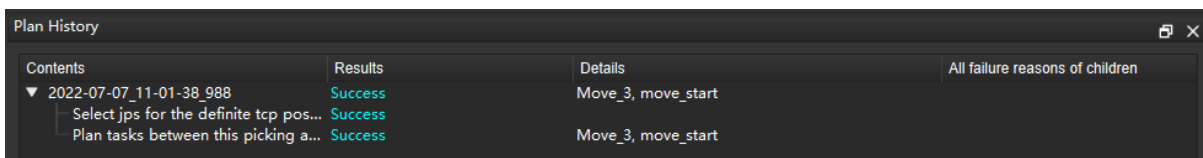
Hint: If you cannot see the tab, please go to *View* and check the box before **Plan History**.

The interface mainly includes 3 area:  for displaying planning content, result (Success or PlanFail), and all failure reasons;  includes options as display duration time of the collision, expired time, load plan history, clear, and help; right click in the plan history panel and the context menu  provides options as **Search**, **Collapse All**, and **Delete Selected Plan History**.

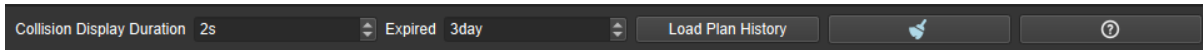


The functions of the options and buttons in the panel are as follows:

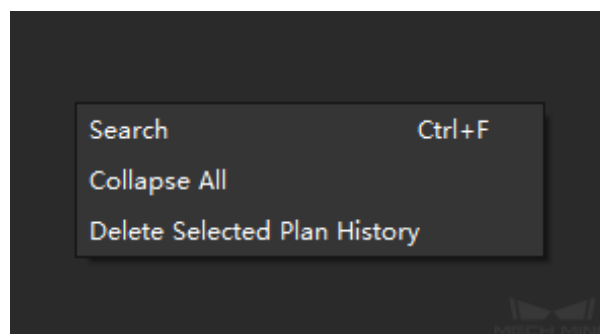
- The options on the top:



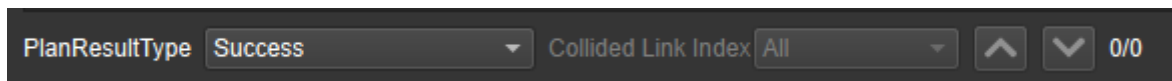
- **Contents:** displays the detailed time and description of the planning process; expand to show the planning contents at every level.
- **Results:** shows whether the planning is successful or not. You may expand the failed planning content for detailed information.
- **Details:** shows the Task used in the path and other details.
- **All failure reasons of children:** shows the cause of failure when the planning fails.
- The options and buttons at the bottom:



- **Collision Display Duration:** the collision will be displayed in the simulation area for the time duration you set.
 - **Expired:** after the expired time, the previous plan history will be overwritten by the current plan history.
 - **Load Plan History:** loads the saved plan history.
 - **Clear:** clears the current plan history.
 - **Help:** click to read the introduction of the plan history.
- The options in the context menu after right clicking:



- **Search:** after selecting *Search*, two filter buttons will appear at the bottom of the panel. You can search for the planning item you need according to the plan result type and the collided link index.

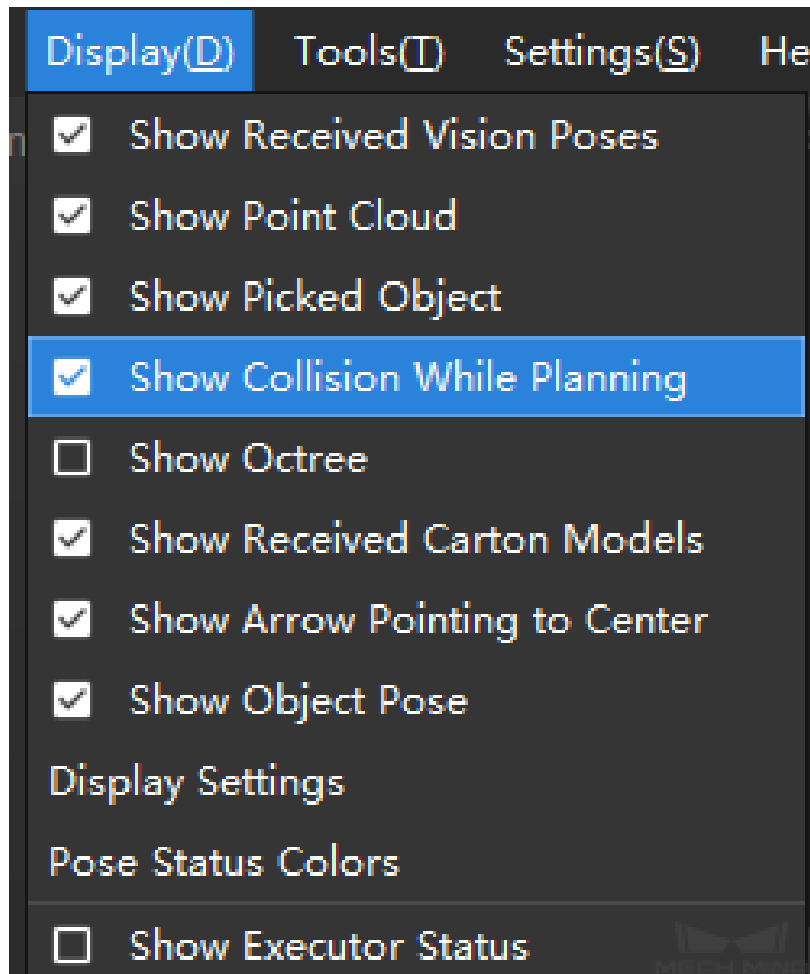


- **Collapse All:** collapses all sub-nodes in the panel and only displays the parent nodes.
- **Delete Selected Plan History:** deletes the selected plan history item.

8.2 Visualize Collision

8.2.1 Display Collision While Planning

If you would like to visualize the collision, go to *View* and check the box before **Display Collision While Planning**.



- Collision with the scene object
 - Click on the task in which the collision occurs, and the object in the scene will be highlighted.
 - Click on the object that collides with the robot, and the object will be highlighted.
- Collision with the point cloud
 - Click on the task in which the collision occurs, and the object in the scene will be highlighted.
 - Click on the collision volume, and the point cloud which collides with the robot will be highlighted.
 - When the point cloud collides with the robot, if the collision is not recorded, only the other collided object will be highlighted.
- Collision with the workobject
 - When the workobject collides with the robot, it will be highlighted.

8.2.2 Compute collision contacts and record collision contacts

Compute collision contacts and Record collision contacts are used for visualization for collision. Please refer to *Computation Settings* for detailed information.

8.3 Details

8.3.1 Success

As shown in the figure below, there are failed sub-nodes (❌) under the parent node (✅). However, the planning at the last sub-node succeeded, and therefore the planning at its parent node succeeded as well.

Contents	Results	Details	All failure reasons of children
▶ 2022-07-07_17-15-35_578	Success	back_to_HOME	
▶ 2022-07-07_17-15-37_316	Success	move_1	
▼ 2022-07-07_17-15-38_066	Success(has failed child nodes)	relative_move_4, visual_move_1, relative_move_1,...	SceneCollision.
Select jps for the definite tcp pos...	Success		
Plan tasks between this picking a...	Success	relative_move_4	
▶ Plan from pick until before next ...	Success(has failed child nodes)	visual move 1, relative move 1, via point, via po...	SceneCollision.
▼ Try 1/22 pick candidates	Success(has failed child nodes)		SceneCollision. 1
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 0°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 0°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 5°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 5°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 5°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 5°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 5°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 5°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision. 2
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 10°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	SymmetryTryFail	0°, 15°(obj-Z, X-axis rotation increment)	SceneCollision.
▶ Apply symmetry/relaxatio...	Success	0°, 0°(obj-Z, X-axis rotation increment)	SceneCollision. 3
▶ 2022-07-07_17-15-42_945	Success	move_3	

8.3.2 Self Collision

Contents	Results	Details	All failure reasons of children
▼ 2022-07-07_09-56-57_713 Current Task Link 1 Link 2	SelfCollision Move_2 Wrist Forearm	Wrist and Forearm collided Jps: (3.40339, -1.5708, 3.21141, -1.5708, 0, 0)	SelfCollision.

Cause: As shown in the details, the wrist and forearm of the robot collided with each other. You can expand the parent node information and find that the collision occurred in the task **Move_2**.

Fix: Adjust the joint positions and pose of the tool to avoid self collision.

8.3.3 Collide with Scene Object

Contents	Results	Details	All failure reasons of children
▼ 2022-07-08_09-56-49_688 Select jps for poses after ▼ Check trajectory collision...	SceneCollision Success SceneCollision	Home EndEffector and scene object [boxwho...	SceneCollision. SceneCollision.
Current Task Trajectory Start Trajectory End Link 1 Scene Object Name Scene Object Pose Scene Object Size	Home, end effector nam... initial jps Home, end effector nam... EndEffector boxwholebox[-y] {(-0.320502, -0.543936, 0... (0.57, 0.03, 0.31)	Jps: {-1.81392, -0.372419, -0.614204, -... TcpPose: {-0.160434, -0.583779, 0.307... Jps: {-1.36033, -0.345304, -0.005763, -...	

Cause: As shown in the details, the end effector collided with the scene objects. You can expand the parent node information to check the pose of the scene object.

Fix: Adjust the pose of the robot or the tool to avoid colliding with the scene object; or you can adjust the pose of the collision model.

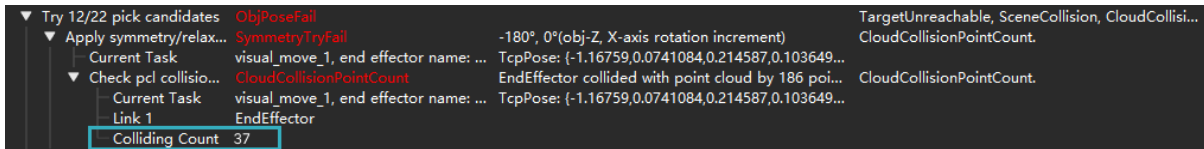
8.3.4 Target Unreachable

Contents	Results	Details	All failure reasons of children
▼ 2022-07-07_10-00-26_286 Current Task Reference for Jps Selection	TargetUnreachable Move_1, end effector name: 0 ...	0/0/0-origin/revision/final solution count, {should... TcpPose: (1.7,2.54569,2.14608,0.5,-0.5,0.5,0.5) Jps: {}	TargetUnreachable.

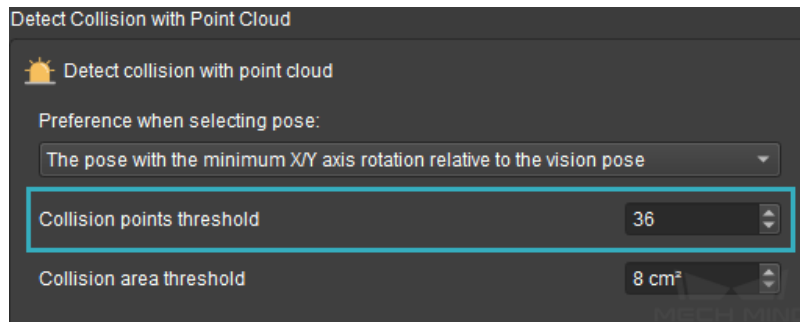
Cause: The target in **Move_1** is out of the reachable range of the robot.

Fix: Adjust the pose.

8.3.5 Cross the Collision Points Threshold



Cause: The number of the collision points are 37 when the picking pose as shown in the figure above is attempted. As it outnumbered the collision points threshold which is set to 37, a collision was detected.



Fix: Check whether the real robot do collide with other objects first. Then go to the **Collision** panel and select **Compute complete collision contacts of each candidate solution** and **Save in plan history**. You can also adjust the collision points threshold, pick point, the pose of TCP and end effector model, etc. to avoid the collision.

Collisions

Computation Settings

Compute collision contacts: Compute complete collision contacts of each candidate solution (for parameter adjustment and collision visualization) **2**

Record collision contacts: Save in plan history (slow, for parameter adjustment and collision visualization)

Collision Detection Configuration

Point cloud is converted to octree with resolution: 1 mm.
Prefer the pose with the minimum XY axis rotation relative to the vision pose.

	Robot Links	Scene Objects	End Effector (Surface)	Point Cloud	Detected Non-Cuboid	Picked Non-Cuboid	Placed Non-Cuboid
Robot Links							
Scene Objects							
End Effector (Surface)							
Point Cloud	5 mm ³		36, 8 cm ²				
Detected Non-Cuboid							
Picked Non-Cuboid							
Placed Non-Cuboid							

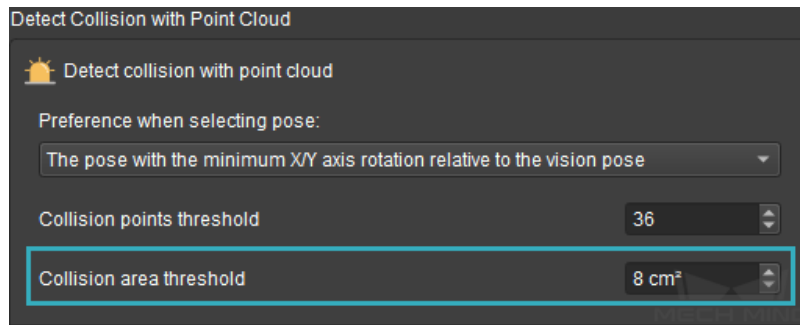
Scene Workflow Robot End Effector & Objects Collisions **1** history Others Log

8.3.6 Cross the Collision Area Threshold

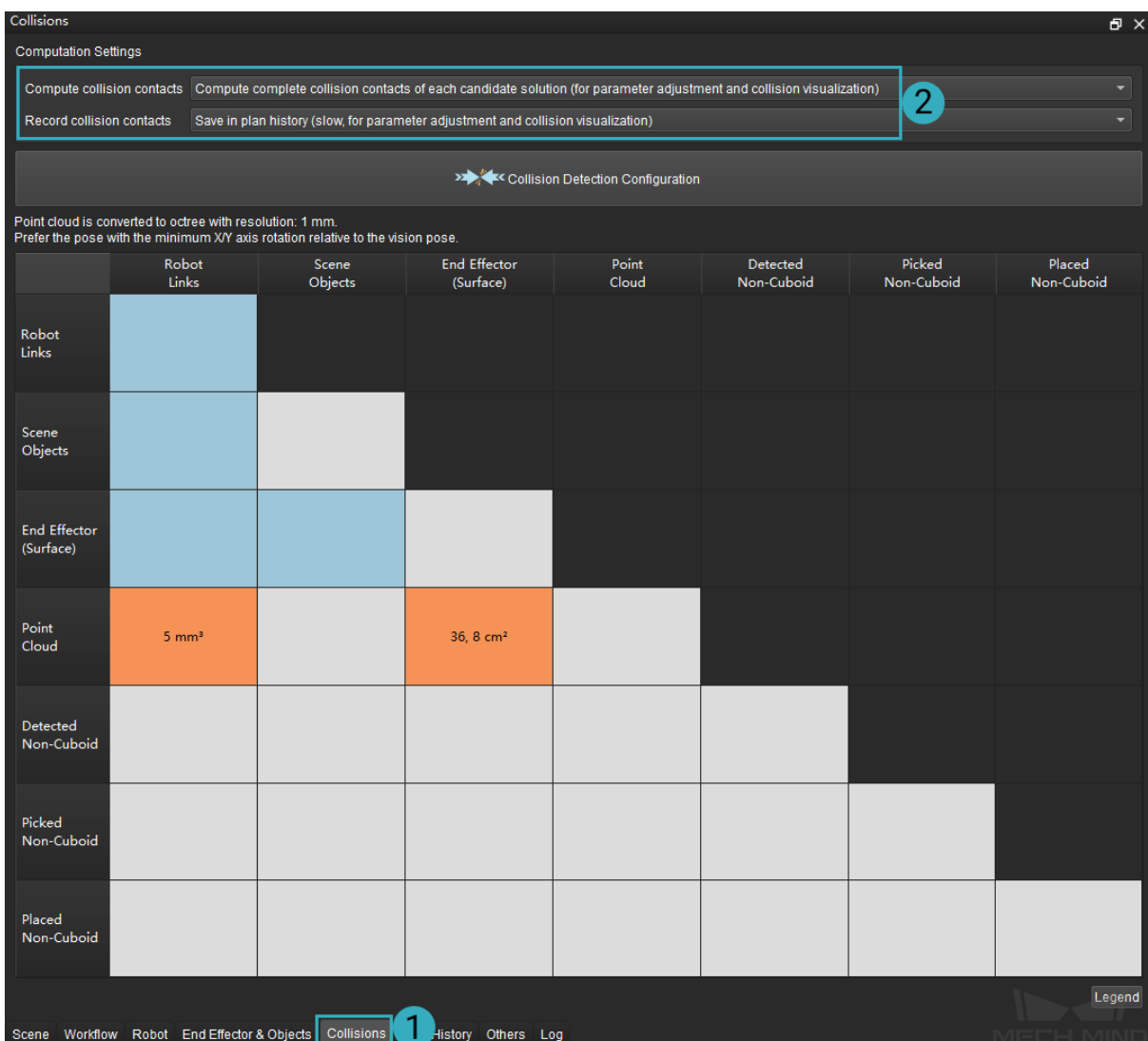
```

    Apply symmetry/relax... SymmetryTryFail 0°, -5°(obj-Z, X-axis rotation increment) CloudCollisionArea.
    - Current Task visual_move_1, end effector name: ... TcpPose: {-1.76639,0.202913,0.247263,-0.197874,...
    - Solve pick with pe... Success
    - Select jps for pick... Success
    Check trajectory c... CloudCollisionArea EndEffector slid through pcl by 65.3 cm^2 CloudCollisionArea.
    - Current Task visual_move_4, end effector name: ... Jps: {-1.51876,0.213305,-0.368872,-0.138563,-0.9...
    - Trajectory Start relative_move_4, end effector name... TcpPose: {-1.77414,0.241009,0.339397,-0.197874,...
    - Trajectory End visual_move_1, end effector name: ... Jps: {-1.52317,0.217014,-0.38029,-0.140454,-0.91...
    Link 1 EndEffector
    Colliding Area 8.00000006
    
```

Cause: The collision area is 8.00000006 cm² when the picking pose as shown in the figure above is attempted. As it outnumbered the collision area threshold which is set to 8 cm², a collision was detected.



Fix: Check whether the real robot do collide with other objects first. Then go to the **Collision** panel and select **Compute complete collision contacts of each candidate solution** and **Save in plan history**. You can also adjust the collision points threshold, pick point, the pose of TCP and end effector model, etc. to avoid the collision.



Collisions

Computation Settings

Compute collision contacts: Compute complete collision contacts of each candidate solution (for parameter adjustment and collision visualization) **2**

Record collision contacts: Save in plan history (slow, for parameter adjustment and collision visualization)

Collision Detection Configuration

Point cloud is converted to octree with resolution: 1 mm.
 Prefer the pose with the minimum XY axis rotation relative to the vision pose.

	Robot Links	Scene Objects	End Effector (Surface)	Point Cloud	Detected Non-Cuboid	Picked Non-Cuboid	Placed Non-Cuboid
Robot Links							
Scene Objects							
End Effector (Surface)							
Point Cloud	5 mm ³		36, 8 cm ²				
Detected Non-Cuboid							
Picked Non-Cuboid							
Placed Non-Cuboid							

Scene Workflow Robot End Effector & Objects **Collisions** **1** History Others Log

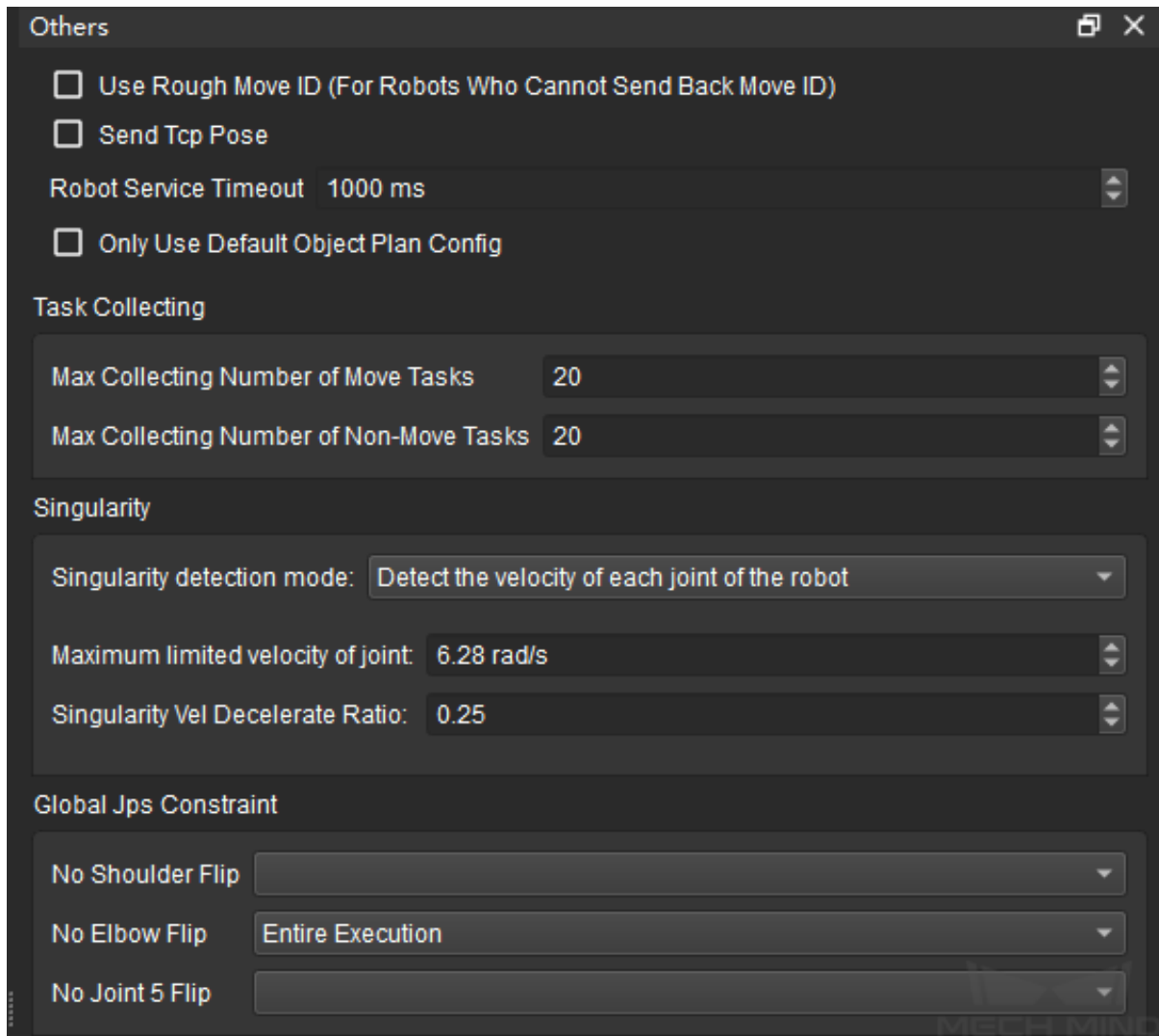
Legend

OTHERS

When Mech-Viz is used to plan paths, you will need to configure operation settings and Task collecting settings in this panel. This section mainly covers the following topics:

- *Operation Settings*
- *Task Collecting*
- *Singularity*
- *Global JPs Constraint*

Click on the **Others** tab in the lower right corner of the interface to open the panel, as shown below.



9.1 Operation Settings

Use Rough Move ID (For Robots Who Cannot Send Back Move ID): Mech-Viz will compute a rough Move ID. When the rough Move ID is used, some non-move Tasks (e.g., *visual_look*, *set_do*) will not interrupt the sending of moves, so that the move transition of the robot can be smoother.

Send TCP: By default, Mech-Viz sends poses in JPs. After this option is checked, Mech-Viz will send poses in TCP.

Robot Service Timeout : The service timeout period of all kinds of communication with the robot (Tasks as *set_do*, *check_di*, etc.) except for move command. The default timeout period is 1000ms.

Only Use Default Object Plan Config: After this option is enabled, even if there are multiple object labels, the symmetry of the default object will be still used.

9.2 Task Collecting

Max Collecting Number of Move Tasks: The maximum number of the move-type Tasks in a project. If the number of move-type Tasks exceeds this threshold, Mech-Viz will fail to plan the path.

Max Collecting Number of Non-Move Tasks The maximum number of the non-move Tasks in a project. If the number of non-move Tasks exceeds this threshold, Mech-Viz will fail to plan the path.

9.3 Singularity

If the robot reaches the point of a singularity, it may not be able to execute certain moveL commands. In order to avoid singularities in the planned path, singularity detection should be enabled.

Singularity detection mode:

- **Detect the velocity of each joint of the robot**

In this mode, you will need to set the maximum limited velocity of joint and velocity decelerate ratio to filter plan results that may lead to singularities.

Maximum limited velocity of joint: When the velocity of joint exceeds this value, Mech-Viz will assume that a singularity occurs.

Singularity Vel Decelerate Ratio: When the velocity decelerate ratio exceeds this value, Mech-Viz will assume that a singularity occurs.

Note: Velocity decelerate ratio = reduced speed (or acceleration) / set original speed (or acceleration)

- **Detect joint angle of the robot**

In this mode, you will need to set the joint and its bound of angle to filter plan results that may lead to singularities. **Only linear motion will be detected.**

Select joint: Select a joint of the robot.

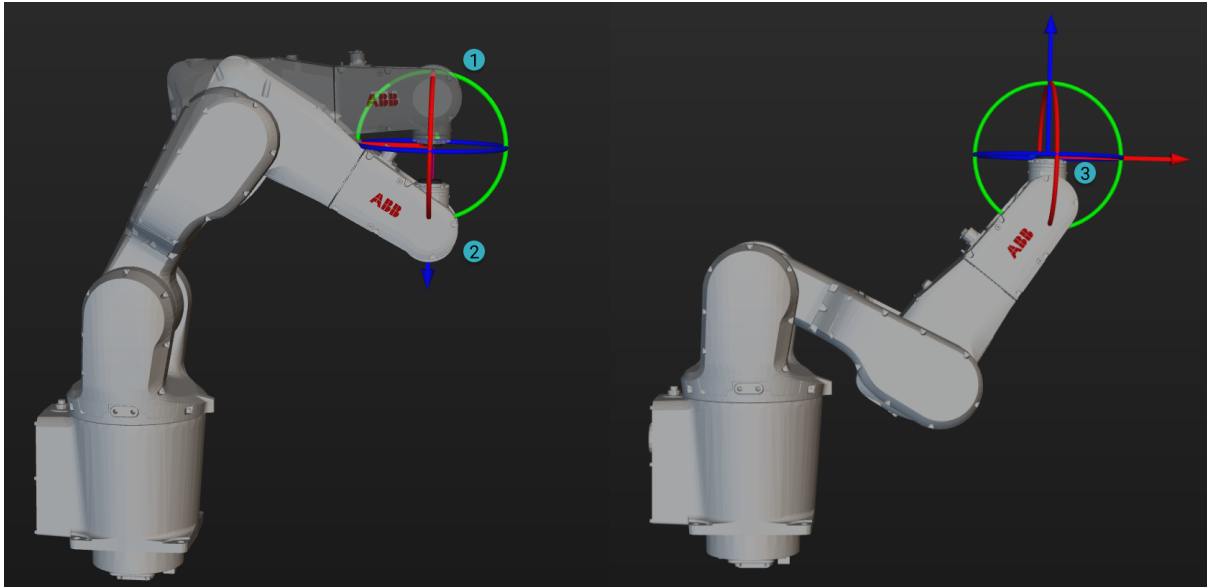
Lower bound of angle: When the bound of angle is lower than this value, Mech-Viz will assume that a singularity occurs.

Upper bound of angle: When the bound of angle is higher than this value, Mech-Viz will assume that a singularity occurs.

9.4 Global JPs Constraint

No Shoulder Flip/No Elbow Flip/No Joint 5 Flip: You can choose to reduce unnecessary rotation of the robot **While Holding** objects or during the **Entire Execution**.

However, setting the shoulder/elbow/Joint 5 all to “no rotation” is not necessarily the best choice. An example is shown in the figure below.



When the robot needs to move from Ⓧ to Ⓧ and “No Joint Flip” is set, unnecessary motion is required to reach that point, and the robot may end in position Ⓧ . Therefore, in this case, only **No Shoulder Flip** and **No Elbow Flip** need to be set.

Note: The priority of *JPs Constraint* for move-type Tasks is higher than the global settings here. If there is no setting in move-type Tasks, the global settings will take effect.

Check the chapter below to learn about debugging tips about Mech-Viz projects.

DEBUGGING TIPS

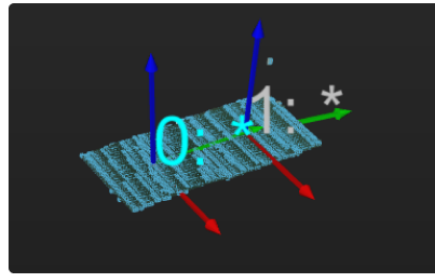
In this chapter, you will learn about debugging tips, which can improve the efficiency of using Mech-Viz and increase the computation speed in the project.

11.1 Modify Display Settings

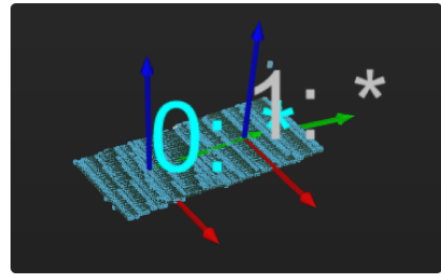
Go to *Display*→*Display Settings* to modify the display effect of the simulation.

Display Effect when Different Parameters are Set

Vision Pose Tag Size

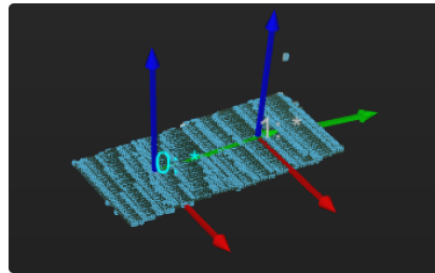


0.05

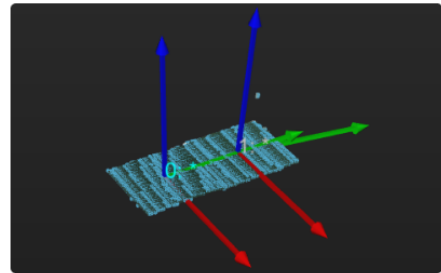


0.07

Vision Pose Dragger Size

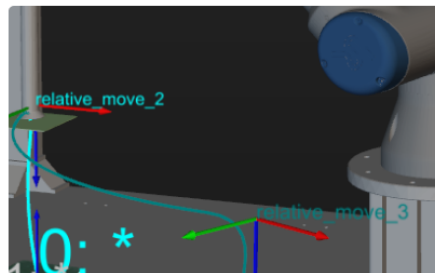


0.05



0.07

Trajectory Text Size

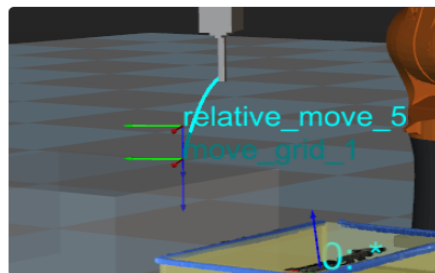


0.02



0.04

Trajectory Dragger Size



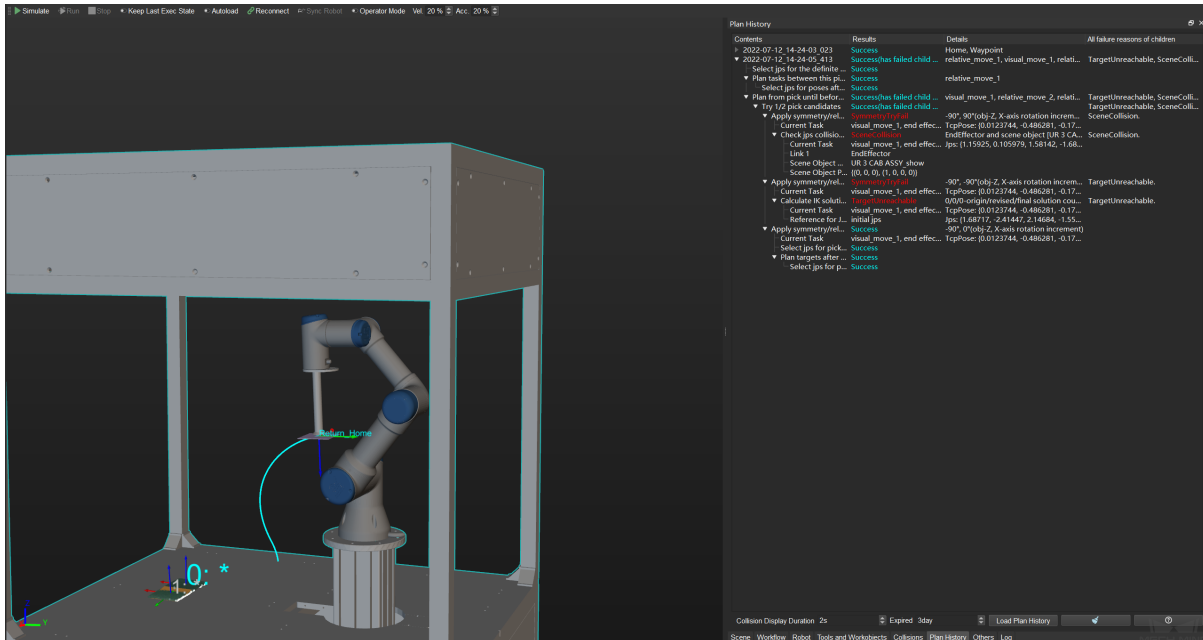
0.05



0.10

11.2 Analyze Problems according to the Plan History

You can check the plan history and detailed descriptions of the errors in the **Plan History** panel to debug the project.

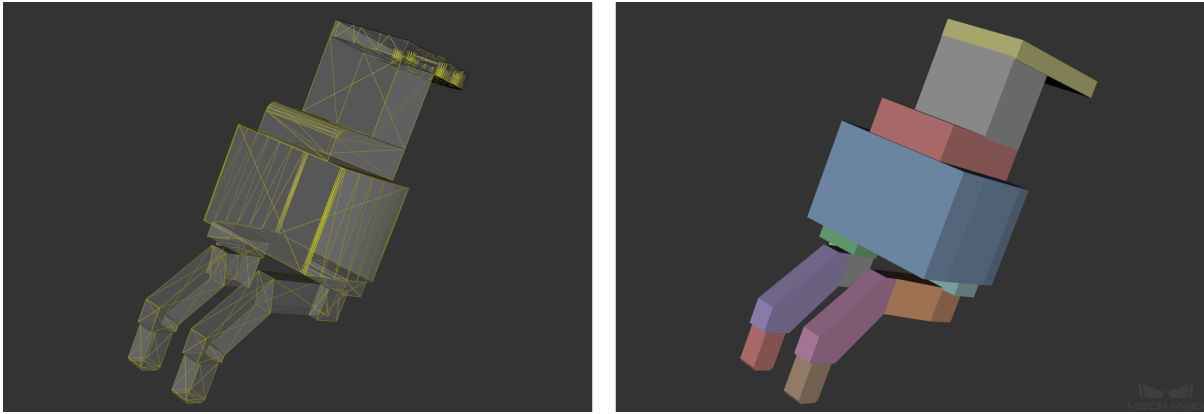


Please refer to [Plan History](#) for detailed information.

11.3 Model Simplification

STL and OBJ are two of the most common file formats used for tool and scene object 3D models. *STL* files describe only the surface geometry of a three-dimensional object, while *OBJ* files contain a three-dimensional object, including 3D coordinates, texture maps, polygonal faces, and other object information. Simply put, models in STL format can be seen as void, while models in OBJ formats can be seen as solid.

The model on the left is an STL file, and the one on the right is an OBJ file.



In collision detection, only the intersection on the surface of the object can be detected when a model in STL format is used; while both intersections on the surface and inside the object can be detected when a model in OBJ format is used.

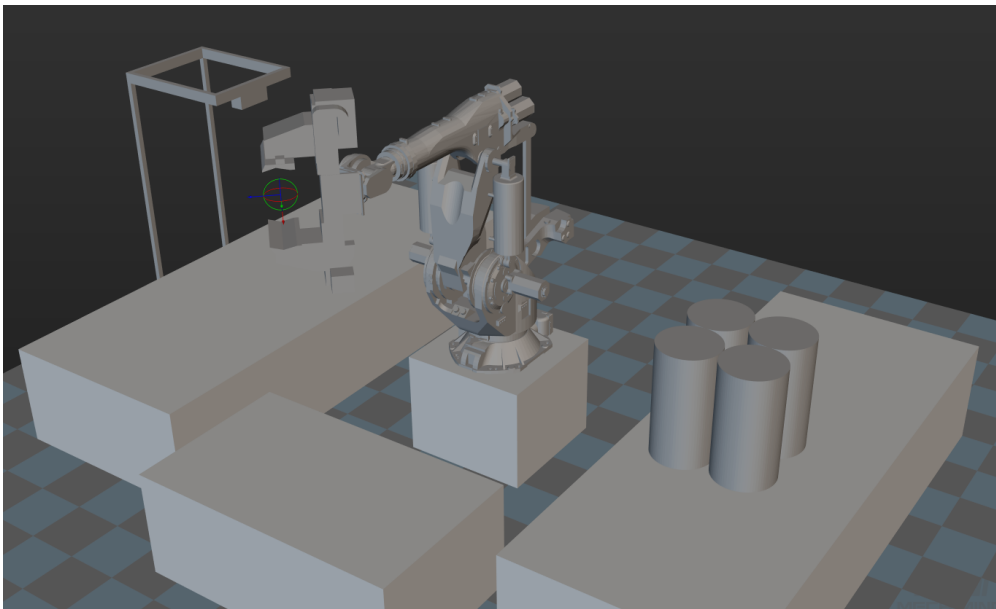
This chapter covers the basic principle of collision detection when using 3D models in STL and OBJ formats and the method for simplifying the models.

11.3.1 Basic Principle of Collision Detection

STL File Format

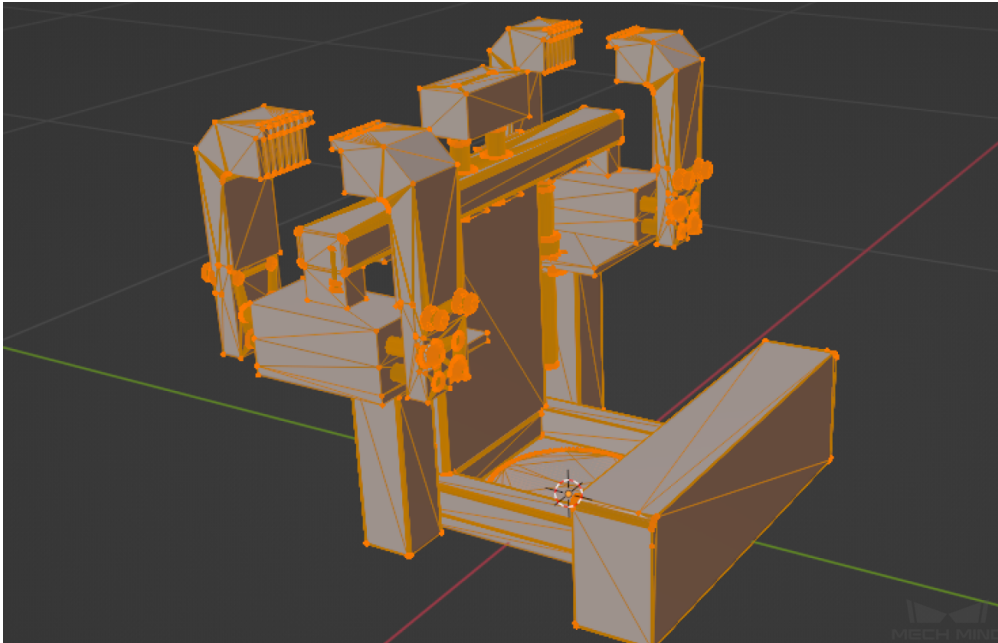
STL is a file format native to the stereolithography CAD software created by 3D Systems. An STL file describes a raw, unstructured triangulated surface by the unit normal and vertices of the triangles using a three-dimensional Cartesian coordinate system.

In Mech-Viz projects, the end effector model and the scene object model are usually in STL format, as shown below.

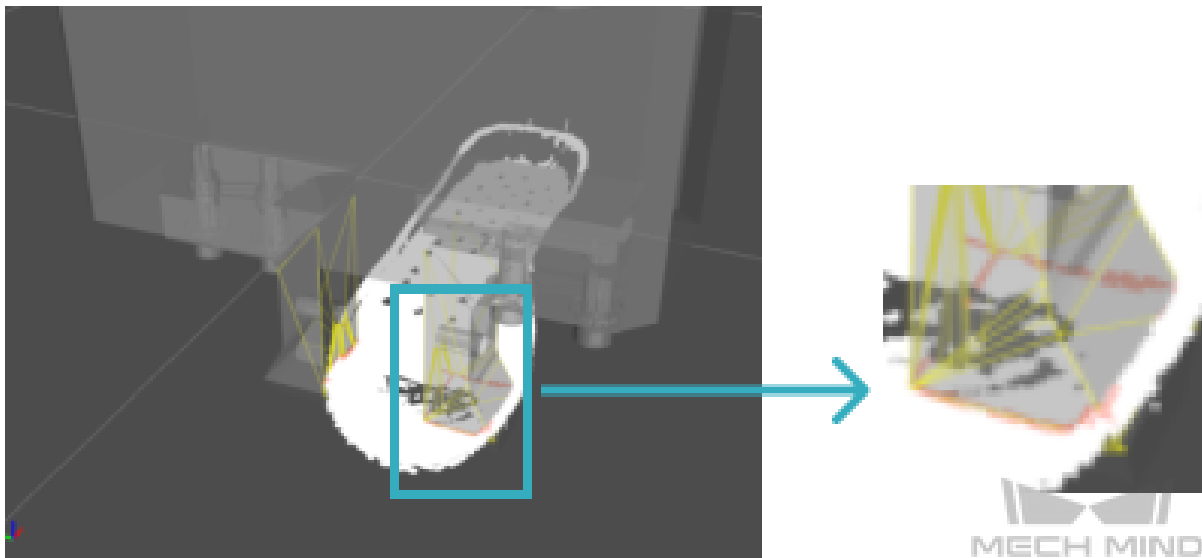


Principle of Collision Detection When Using Models in STL Format

A STL file describes 3D objects based on triangle meshes and only contains surface information. In this case, when using a model in STL format in collision detection, only the collision/intersection between the points and the triangle meshes on the surface of the model will be detected. The points inside the model will not be included in the computation of collision detection. The figure below shows an model in STL format.

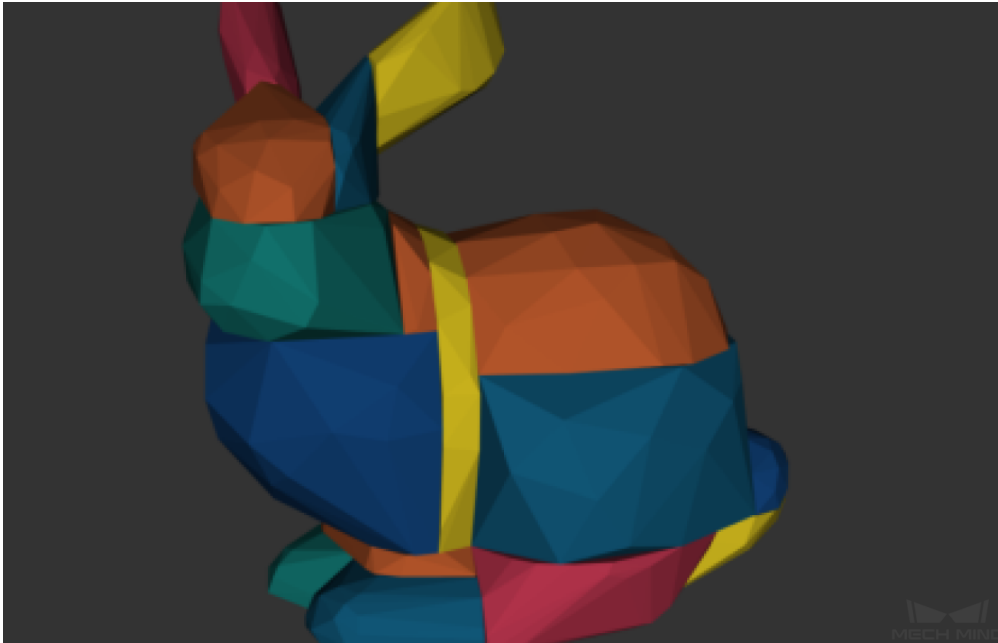


As shown in the figure below, the yellow part indicates the triangle mesh where a collision occur, and the orange part indicates the point cloud that collides with the collision model.



OBJ File Format

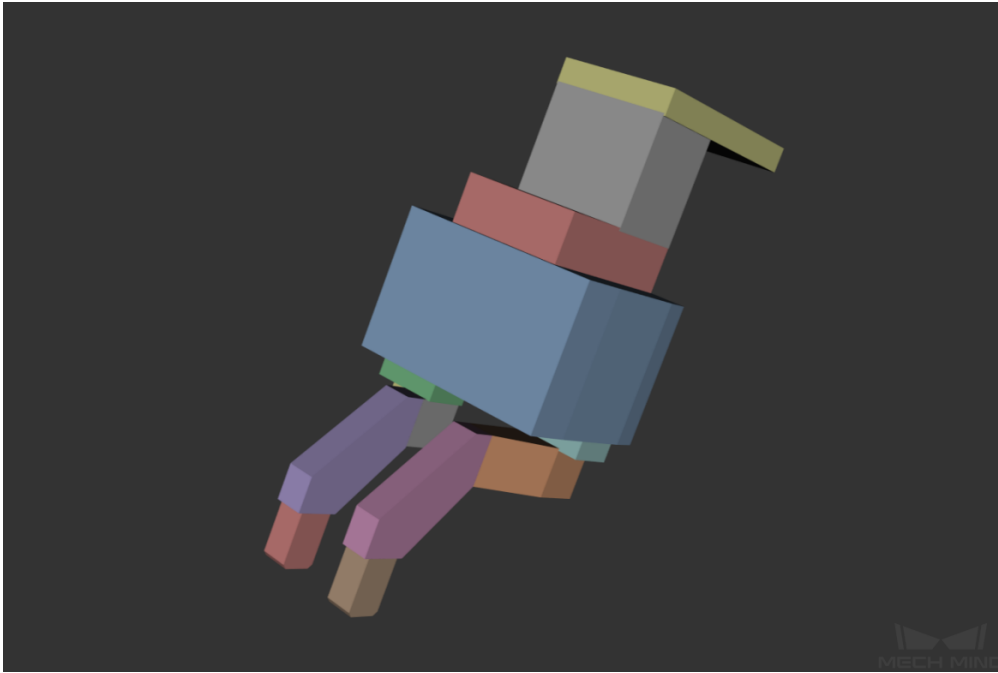
An OBJ file may contain vertex data, free-form curve/surface attributes, elements, free-form curve/surface body statements, connectivity between free-form surfaces, grouping and display/render attribute information. The most common elements are geometric vertices, texture coordinates, vertex normals and polygonal faces. Simply put, models in STL format can be seen as void, while models in OBJ formats can be seen as solid. The figure below shows a model in OBJ format.



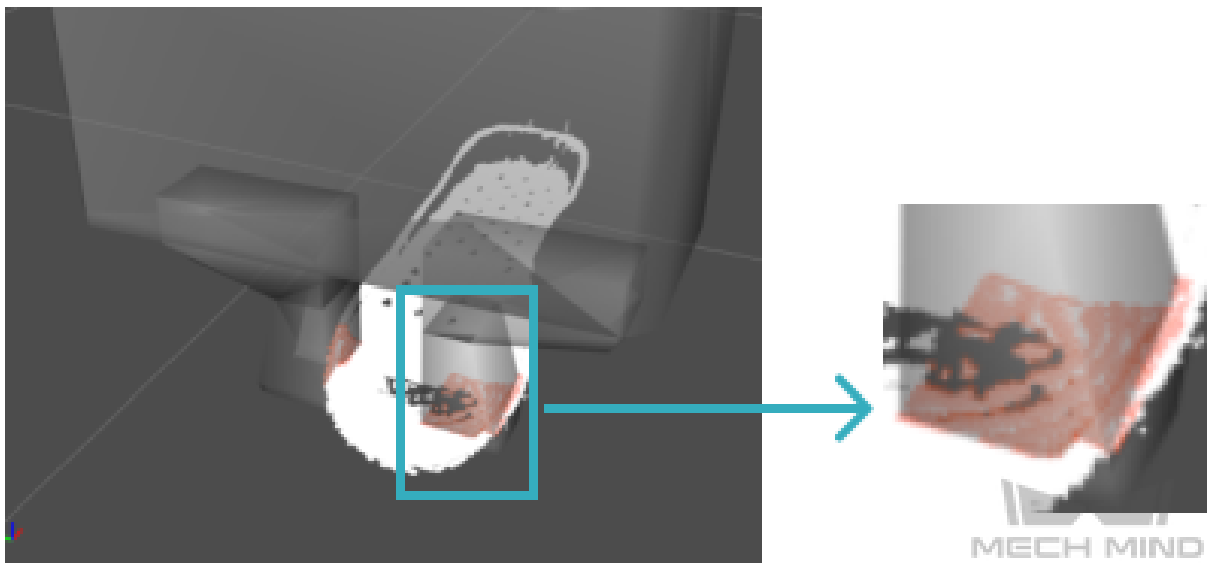
Note: Since OBJ files are saved in plain text, you can open them with a text editor, such as Microsoft Notepad (Windows) or Apple TextEdit (Mac), or a source code editor. You may need to rename the .obj file extension to .txt for the text editor to recognize it.

Principle of Collision Detection When Using Models in OBJ Format

When using a model file in OBJ format in collision detection, the end effector model will be divided into multiple “solid” convex polyhedra first, and then Mech-Viz will compute the intersection between the points and convex polyhedra. The points inside the end effector model will be included in the computation as well, and the detection result will be more accurate.



As shown in the figure below, the orange part indicates where the collision occurs.



11.3.2 STL Models Simplification

Basic Methods

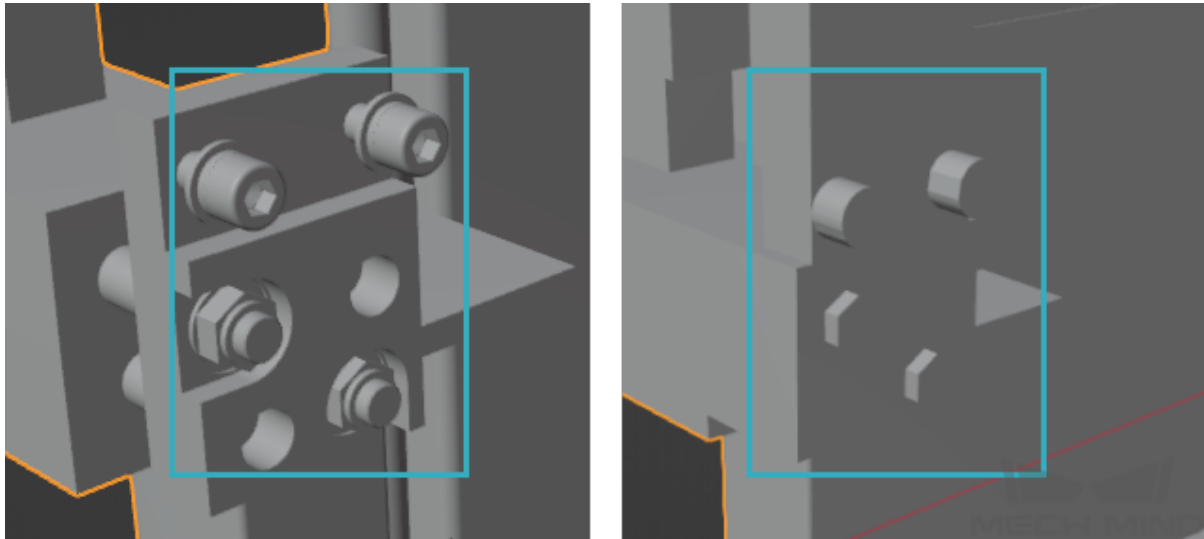
The size of the end effector model file should be no more than 1MB.

Reduce the number of the triangle meshes

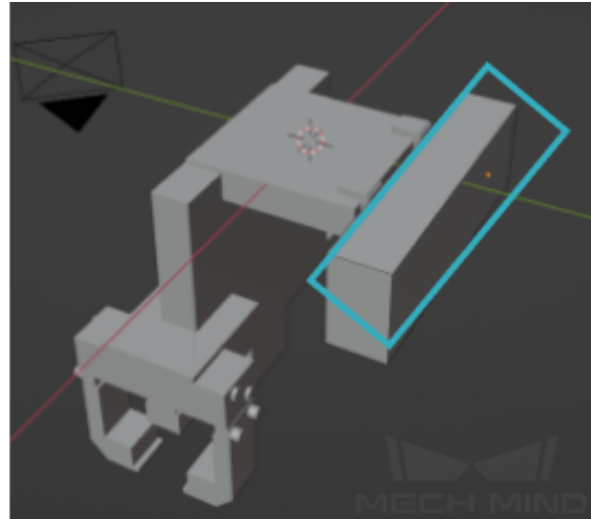
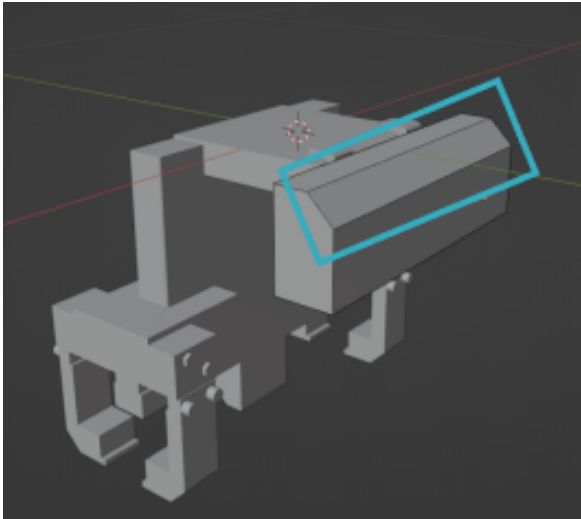
Too many triangle meshes of the model may slow down the computation speed in the project. In order to increase the processing efficiency, the unnecessary surface features of models in STL format should be removed.

The unnecessary surface features mainly includes the following:

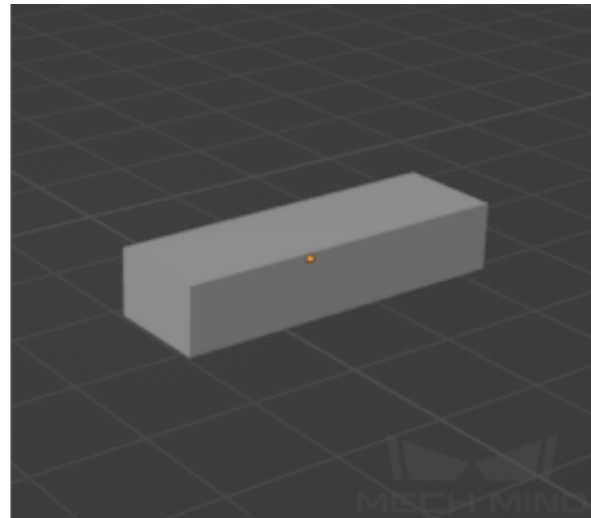
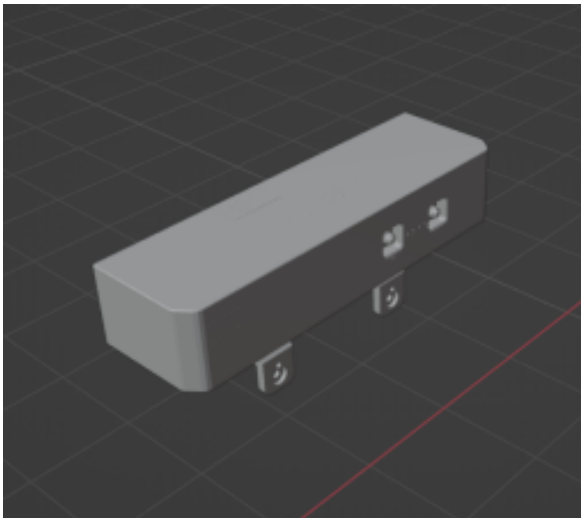
- Round parts, such as screws, screw threads, through-holes, countersunk holes, gaskets, suction cups, etc. The threaded holes, screws and other structures on the end effector should be kept and simplified as well, as shown in the following figure.



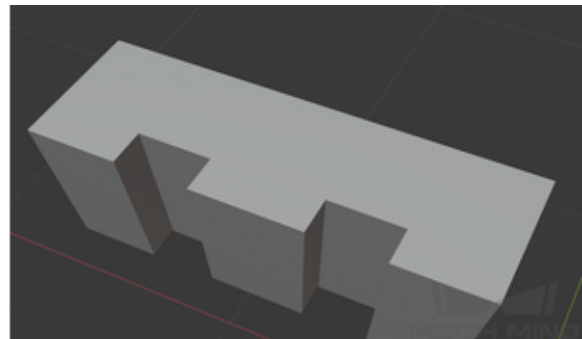
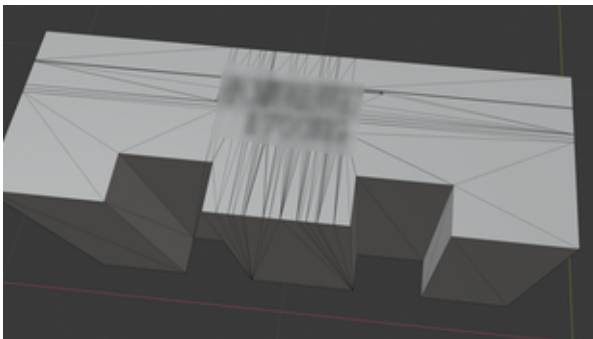
The chamfers or fillets should be removed as well, as shown below.



- Interior structures, such as that of an end effector or camera. For example, the camera can be simplified as a cuboid, and the unnecessary interior structure can be wrapped inside the polyhedra directly, as shown below.

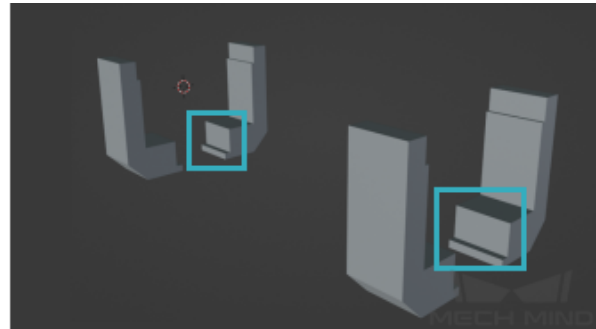
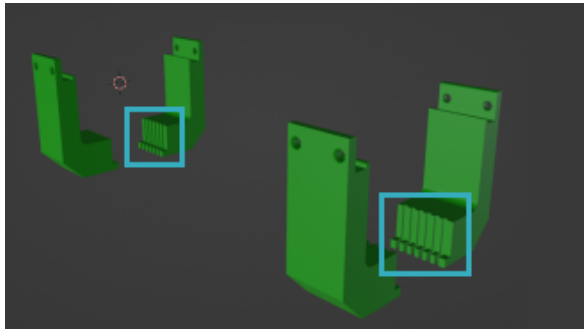


- Others, such as texts, grids, external devices, cables, etc.

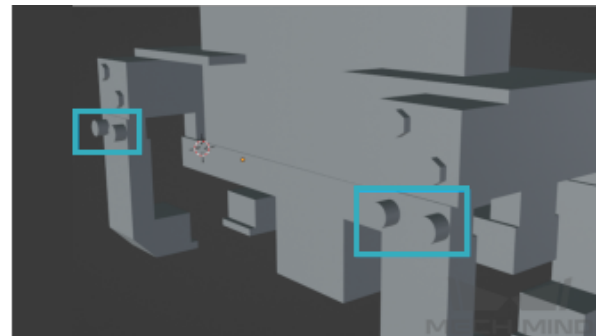


Keep the original shape of the end effector

- Keep the basic shape of the end effector, such as curved parts, the shape of the gripper, etc., as shown below.

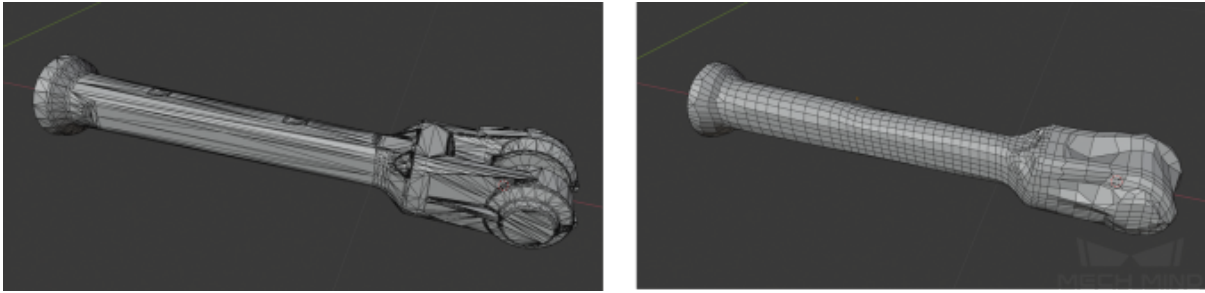


Keep the protrusions of the end effector, such as sensors, cylinders, protruding screws, etc., as shown below.


Adjust the shape of the end effector according to actual needs

- For projects with high requirements on collision detection, even a slight deviation will lead to a collision or damage to the workobjects, and therefore the details of the end effector should be emphasized.
- Parts of the end effector that will not collide with the point clouds of workobjects and scene objects during picking can be removed.
- Adjust the lines of the end effector model according to the actual situation of the project to make it more reasonable.

The figure below shows a model before and after being simplified.

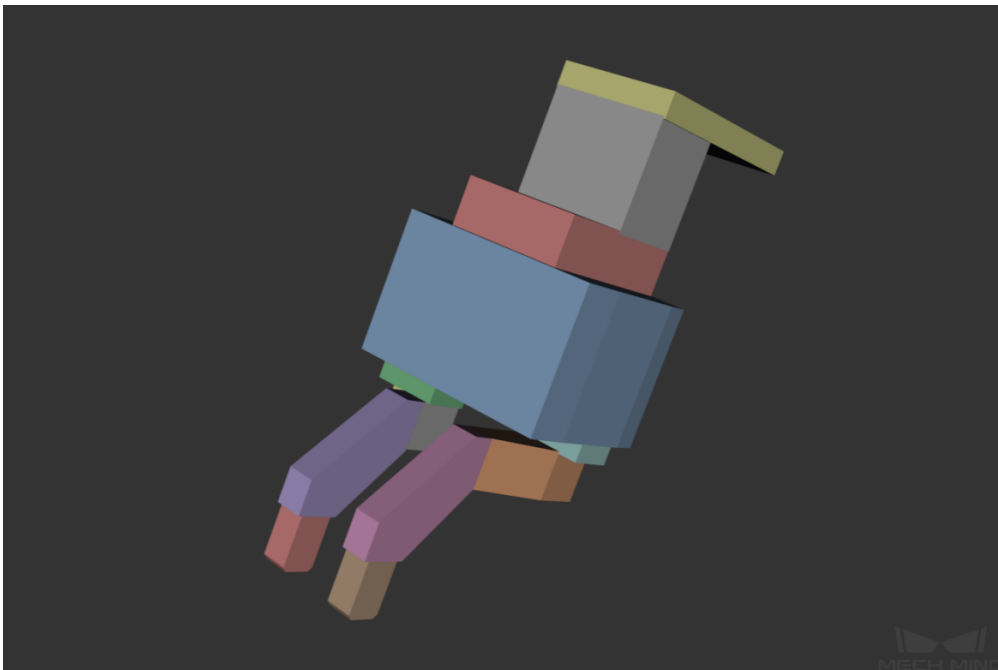


Hint: Mech-Viz provides a built-in model editor to simplify models. Please refer to [Model Editor](#) for detailed instructions.

11.3.3 OBJ Models Simplification

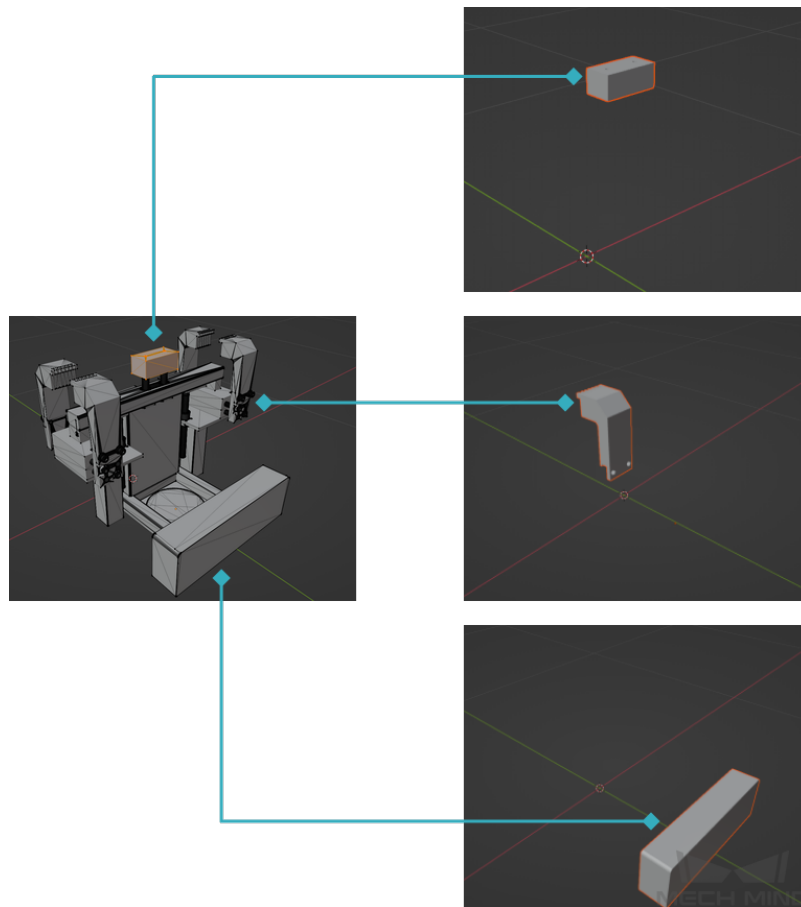
Basic Methods

- The size of the end effector model file should be as small as possible and less than 1MB.
- The end effector model should be processed into a combination of multiple convex polyhedra, as shown below.
- The unnecessary details on the surface of the model should be removed.



Make the model into a combination of multiple convex polyhedra

Split the object into multiple parts, as shown below. Process the parts into convex polyhedra, and then assembly the processed parts into one piece and export in OBJ format.

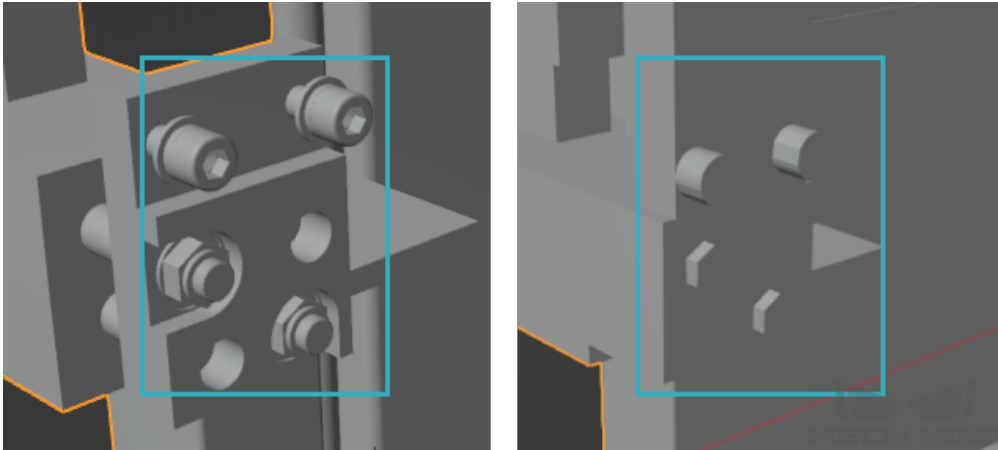


Remove unnecessary surface features

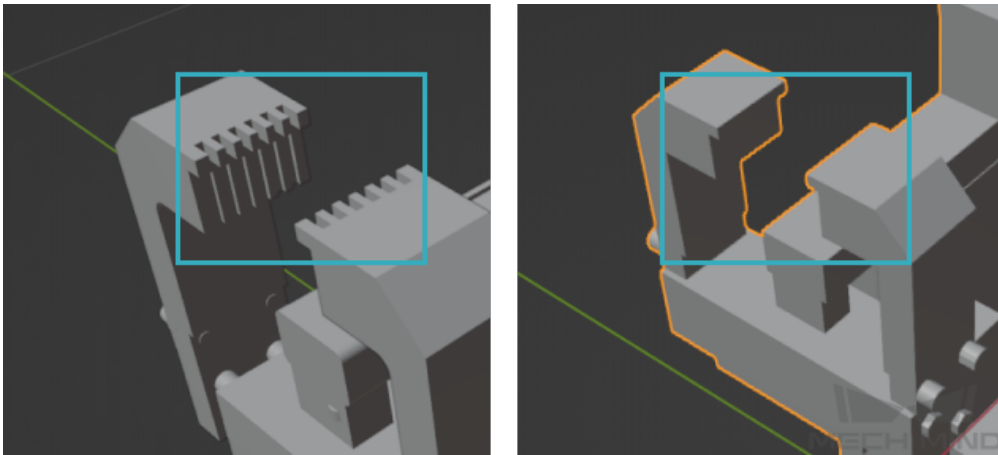
Similar as STL files, too many surface features of the model may slow down the computation speed in the project. In order to increase the processing efficiency, the unnecessary surface features of models in OBJ format should be removed as well.

The unnecessary surface features mainly includes the following:

- Round parts, such as screws, screw threads, through-holes, countersunk holes, gaskets, chamfers, fillets, suction cups, etc. The threaded holes, screws and other structures on the end effector should be kept and simplified as well, as shown in the following figure.



- Interior structures, such as that of an end effector or camera. For example, the camera can be simplified as a cuboid, and the unnecessary interior structure can be wrapped inside the polyhedra directly, as shown below.

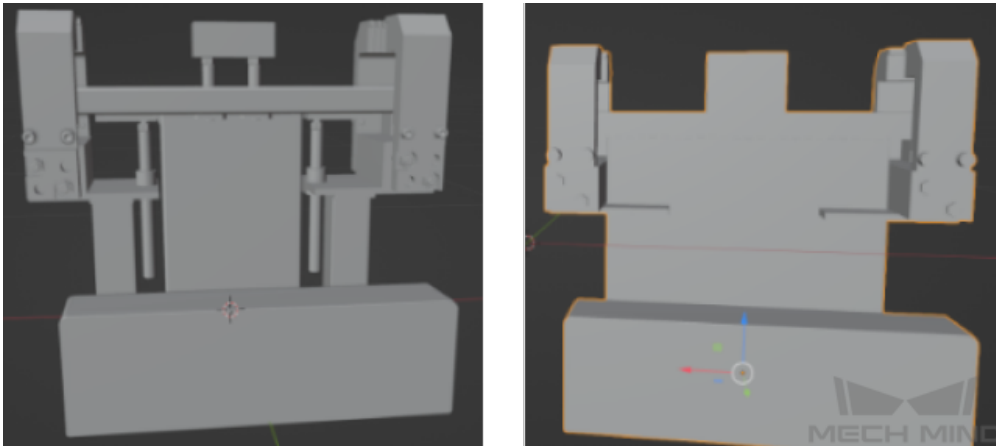


- Surface features, such as serrations, screw threads, etc., as shown below.



Keep the original shape of the end effector

Keep the basic shape of the end effector, such as protrusions, curved parts, and the shape of the gripper, as shown below.



Adjust the shape of the end effector according to actual needs

- For projects with high requirements on collision detection, even a slight deviation will lead to a collision or damage to the workobjects, and therefore the details of the end effector should be emphasized.
- Parts of the end effector that will not collide with the point clouds of workobjects and scene objects during picking can be removed.
- Adjust the lines of the end effector model according to the actual situation of the project to make it more reasonable.

Hint: Mech-Viz provides a built-in model editor to simplify models. Please refer to [Model Editor](#) for detailed instructions.

11.4 Save and Load Vision Records

The saved vision records can be used to reproduce the bugs in the simulated project or in real project on site.

Vision records enables to run a Mech-Viz project without running the corresponding Mech-Vision project. When an error occurs, the Vision records, Mech-Viz project file, plan history, log, and other test data can be sent to the developers who are not on site, and then they can track and fix the bugs based on the data received.

This section covers the following instructions.

- *Save Vision Records*
 - *Save poses only*
 - *Save poses and point clouds*
- *Load Vision Records*
 - *Use the saved vision records*
 - *Tips*

11.4.1 Save Vision Records

Go to the menu bar and then select *Tools* → *Set Vision Records* to open the **Set Vision Records** window.

Save poses only



After running the project, the poses will be stored in **vision_records\Mech-Vision project name\running date of the project** in the installation directory of Mech-Viz, as shown below.

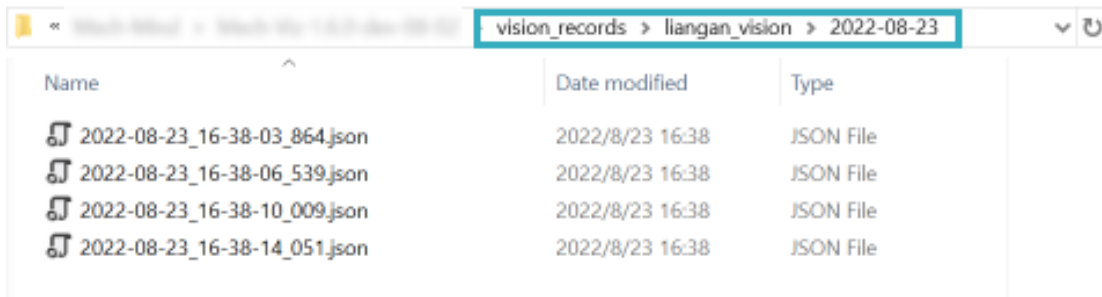


Figure 1 poses

Save poses and point clouds



After running the project, the poses and point clouds will be stored in **vision_records\Mech-Vision project name\running date of the project** in the installation directory of Mech-Viz, as shown below.

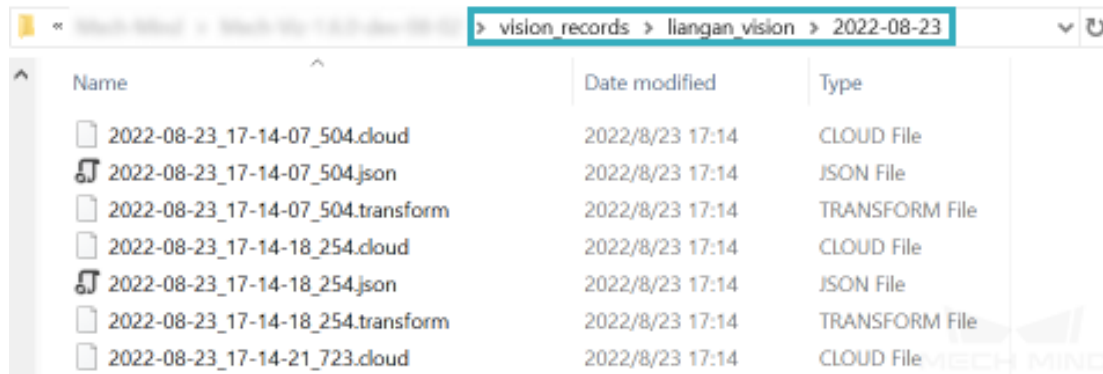


Figure 2 poses and point clouds

In a scenario where point clouds are required, the point cloud file will be saved every time an image was captured. The size of the point cloud files are large and they will take up memory. Therefore, it is recommended to check **Pcl** during testing, and uncheck it after the project starts to run stably.

Attention: Besides the vision_records folder, please save the corresponding Mech-Viz project as well to utilize the vision records and reproduce the bug.

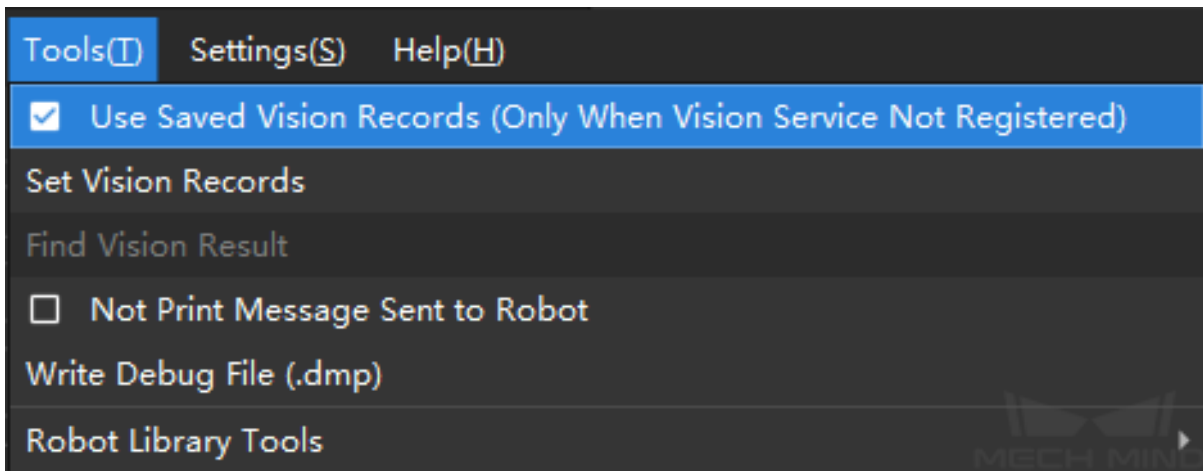
11.4.2 Load Vision Records

Use the saved vision records

1. Open the Mech-Viz project which encountered an error and the vision records have been saved.
2. Make sure that the Mech-Vision project is not registered in Mech-Center, or else the Mech-Vision project will be utilized instead of the vision records.
3. Save the vision records in the vision_records folder in the installation directory of Mech-Viz.

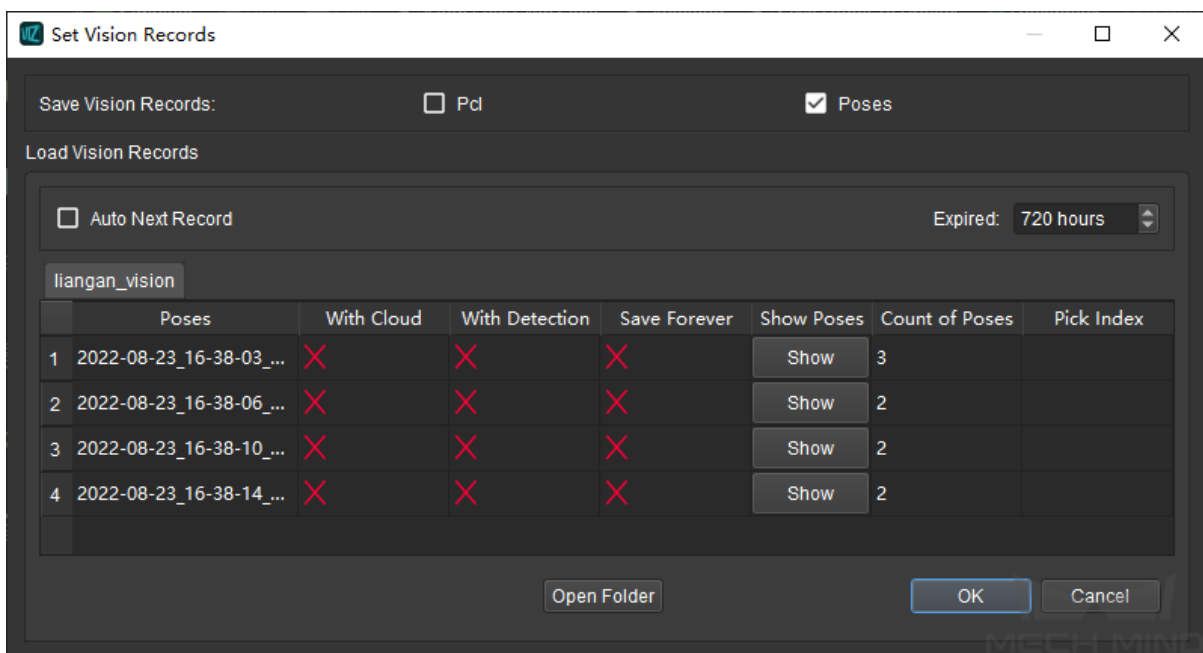
Hint: You can go to *File* → *Open Executable File in Explorer* to locate the installation directory quickly.

4. Click on **Tools** and check **Use Saved Vision Records (Only When Vision Service Not Registered)**.



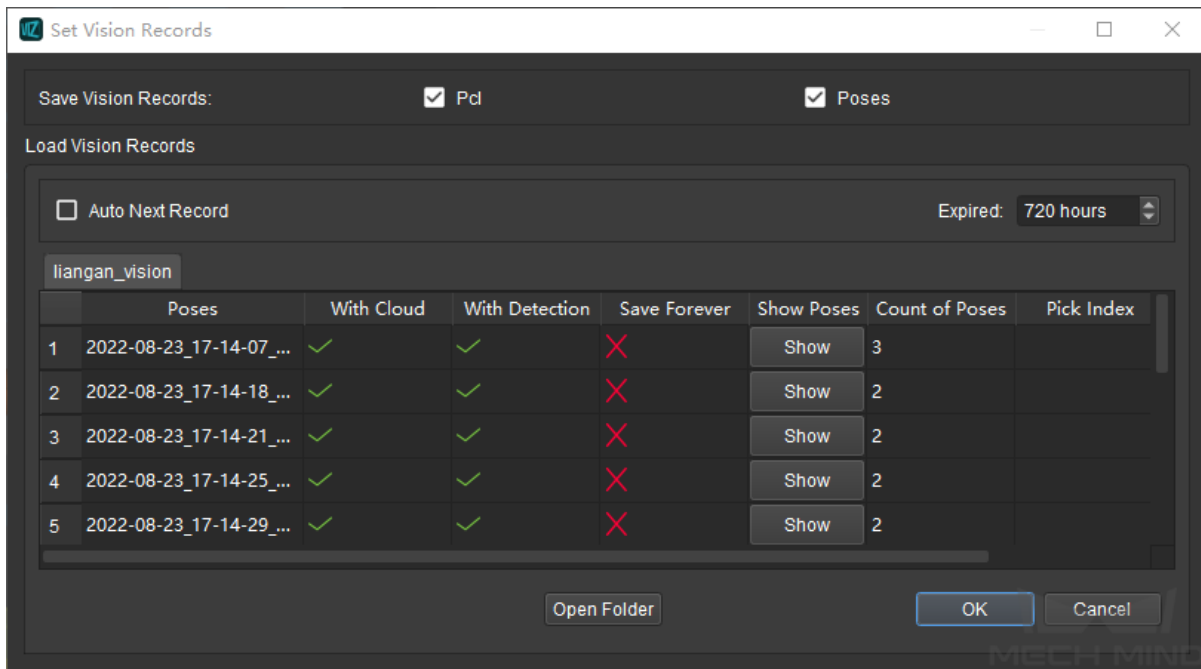
5. Click on *Simulate* or *Run* in the toolbar. Mech-Viz will check whether there is any saved vision records in the `vision_records` folder, and a **Set Vision Records** window will pop up.

- The figure below shows the window when the vision records contain poses only.



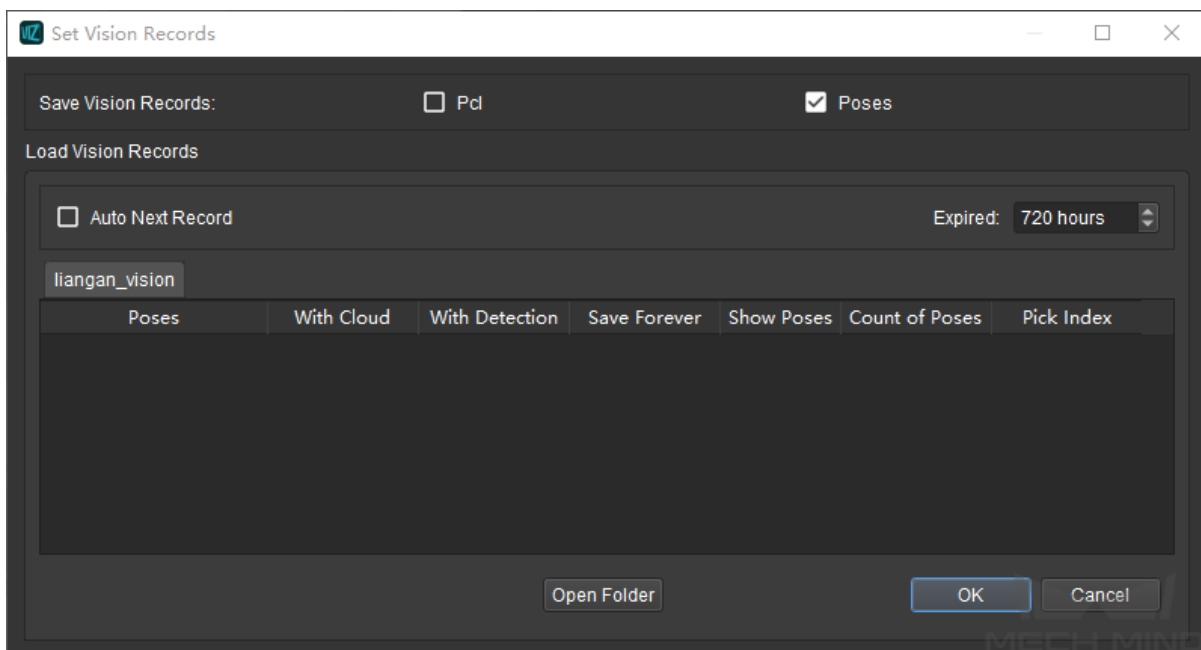
The vision records correspond to the data in *Figure 1*.

- The figure below shows the window when the vision records contain poses and point clouds.

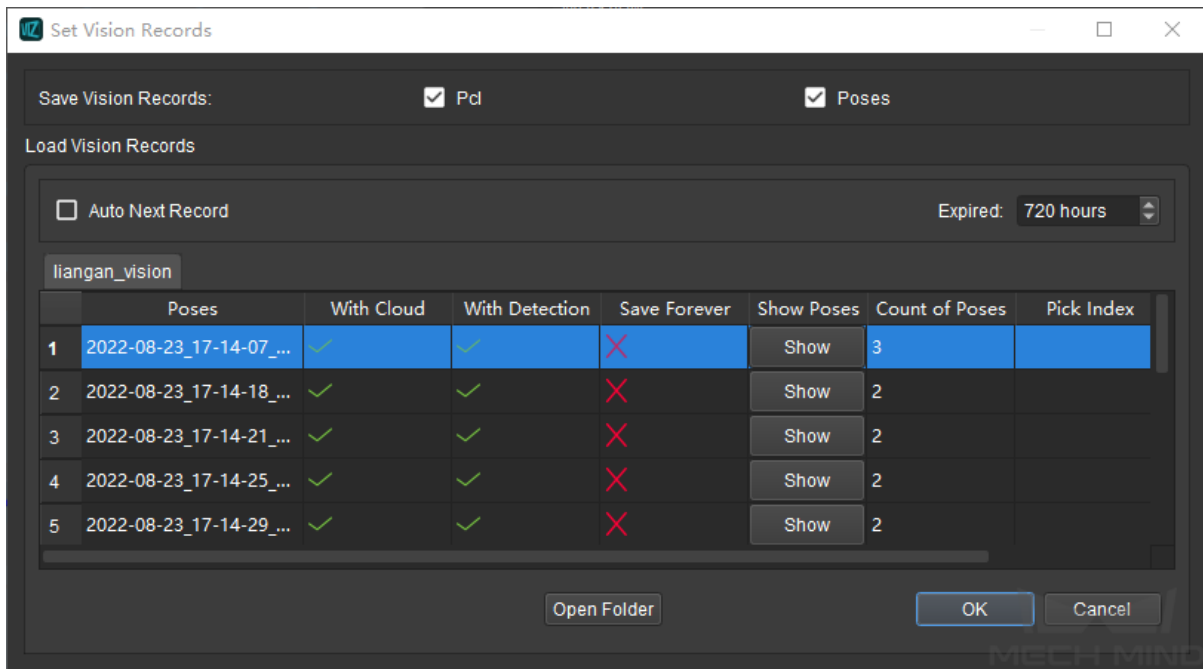


The vision records correspond to the data in *Figure 2*.

- The figure below shows the window when there is no available vision records.

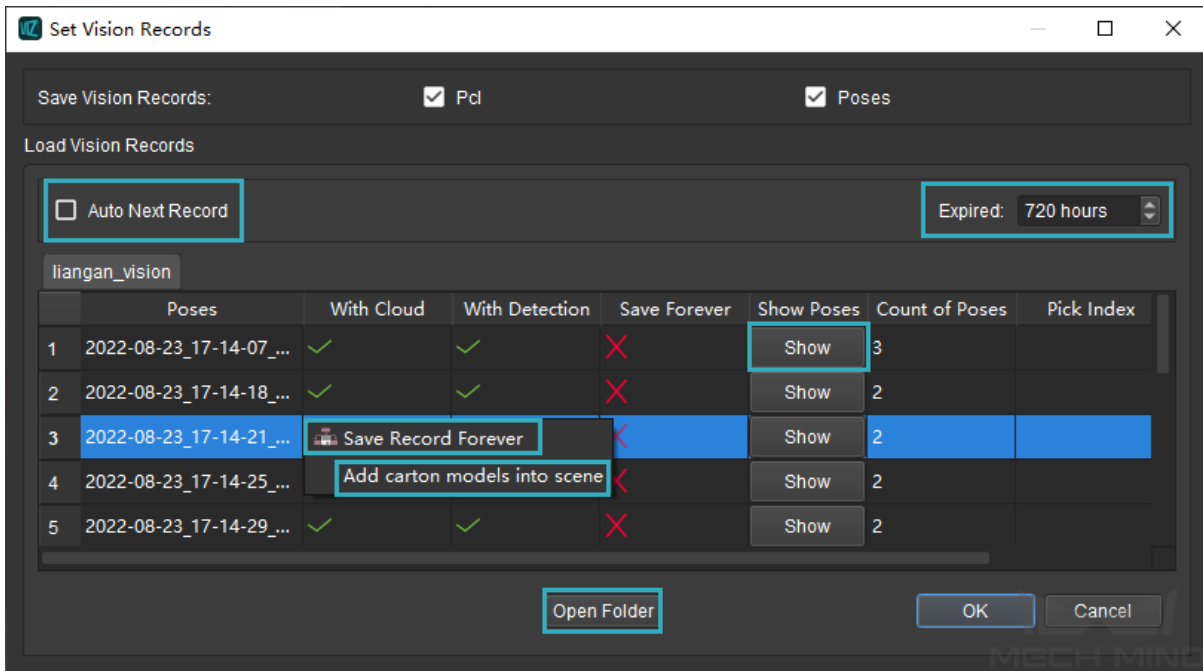


6. Select the vision record you would like to use and click on *OK*, Mech-Viz will reproduce the condition based on the current record.



Attention: After using vision records to reproduce the condition, please make sure to UNCHECK Use Saved Vision Records (Only When Vision Service Not Registered) to avoid errors.

Tips



Auto Next Record	If this option is not checked, the selected vision record will be used repeatedly. If this option is checked, the vision records will be automatically used one after another from the selected vision record, and the project will be stopped when the last vision record is used.
Expired	The saved vision records will be deleted after the expiration time.
Open Folder	Click to open the vision_records folder in the installation directory.
Display	Click on <i>Show</i> to display the pose of this record in the 3D simulation area.
Right click on any	one of the vision records, two options will be displayed in the context menu.
Save Record Forever	After selecting this option, the selected vision record will be saved forever and will not be affected by the expiration time.
Add carton models into scene	Click to load and display the carton models in the scene.

Check the chapter below to learn about how to use the supplementary tools in Mech-Viz to improve efficiency.

SUPPLEMENTARY TOOLS

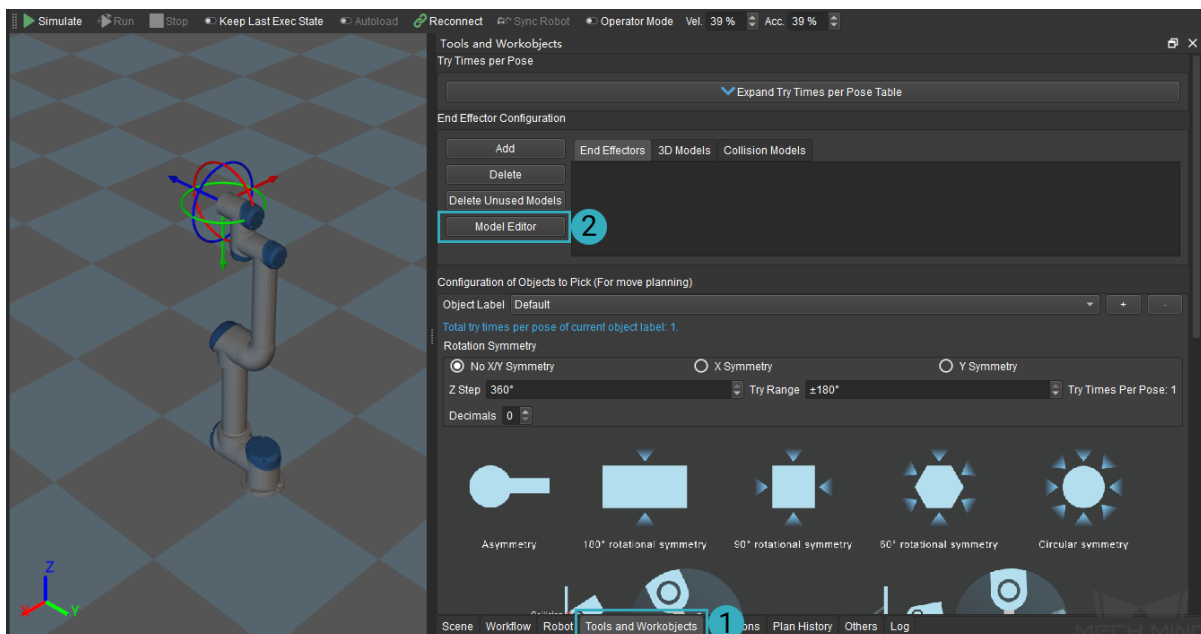
This chapter introduces built-in supplementary tools in Mech-Viz, which can be used to improve the computation speed of the project and accuracy of vision results.

The *Vacuum Gripper Configurator* and *Array Gripper Configurator* are integrated in the Task *visual_move*. The *Pallet Editor* is integrated in the Task *custom_pallet_pattern*.

12.1 Model Editor

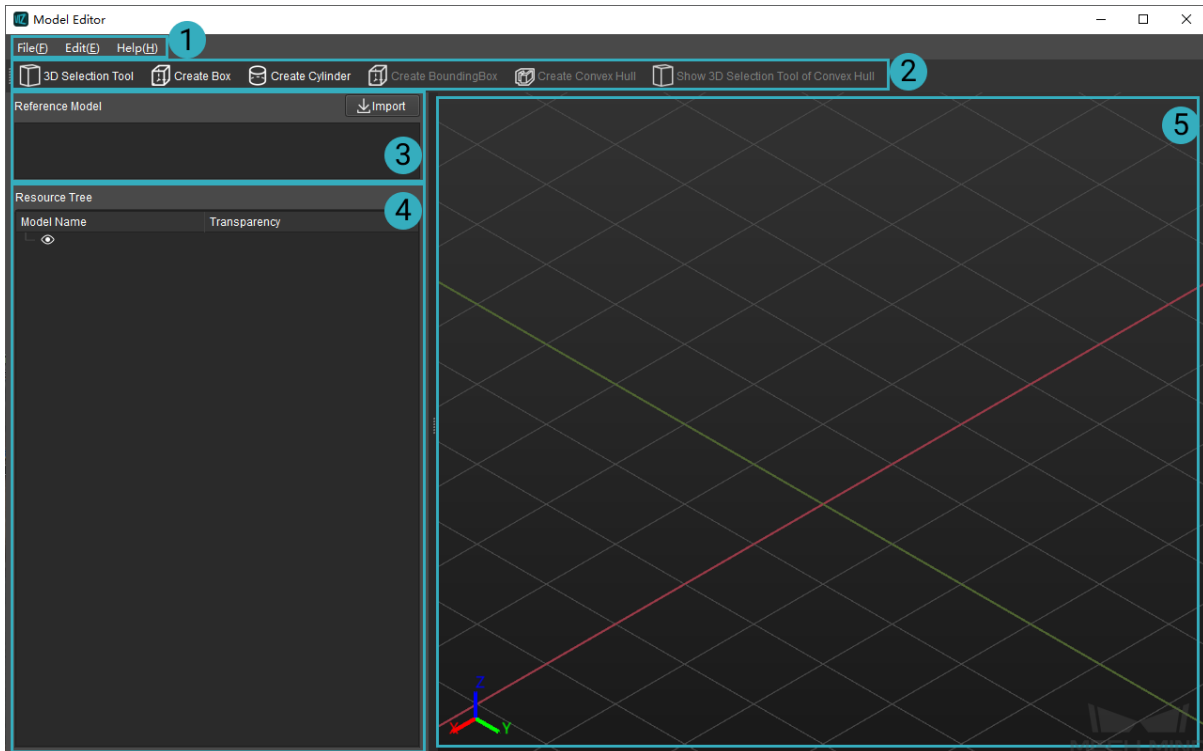
Mech-Viz provides a **Model Editor** to facilitate the simplification of end effector models and scene models, and export collision models in STL and OBJ formats. This section will show you how to use the model editor.

Click on the **Tools and Workobjects** tab in the lower right corner of the interface to open the panel, and then click on *Model Editor* in the **End Effector Configuration** panel to open the **Model Editor**.



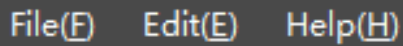
12.1.1 Interface Introduction

The main interface of model editor consists of the following 5 parts.



1. **Menu Bar**: provides options including **File**, **Edit**, and **Help**.
2. **Toolbar**: provides features including **3D Selection Tool**, **Create Box**, **Create Cylinder**, **Create BoundingBox**, **Create Convex Hull**, and **Show 3D Selection Tool of Convex Hull**.
3. **Reference Model**: displays the imported original model which is used as a reference.
4. **Resource Tree**: displays all the created geometric solids and convex hulls.
5. **Editing Workspace**: where you can edit the model.

Menu Bar



File

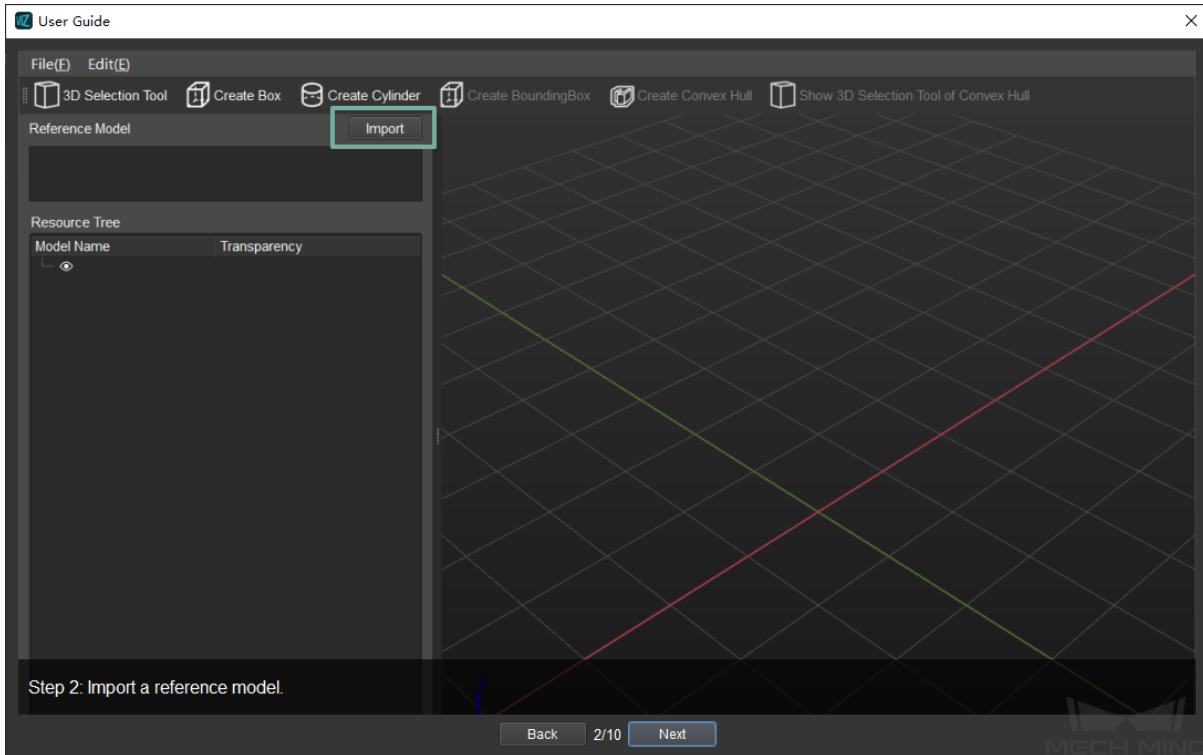
Options	Descriptions	Shortcut
New	Create a new editing project	Ctrl+N
Open	Open an existing M3D file to edit	Ctrl+O
Save	Save a newly created editing project in M3D format or save changes on an existing editing project	Ctrl+S
Save As	Save the editing project in M3D format to a specified directory	Ctrl+Shift+S
Import	Import an original model	N/A
Export	Export the simplified model	N/A
Exit	Exit the model editor window	N/A

Edit

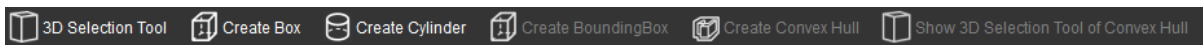
Options	Descriptions	Shortcut
Undo	Undo the change	Ctrl+Z
Redo	Redo the change	Ctrl+Y

Help

Click to view a short demo of how to use the model editor.



Toolbar

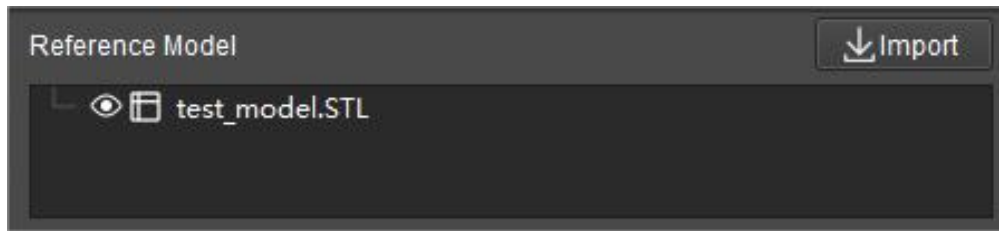


Options	Descriptions	Short-cut
3D Selection Tool	Create a cuboid-shaped selection frame	N/A
Create Box	Create a cuboid model	N/A
Create Cylinder	Create a cylinder model	N/A
Create BoundingBox	Create a bounding box within the space selected by the 3D selection tool	Shift + B
Create Convex Hull	Create a convex hull within the space selected by the 3D selection tool	Shift + C
Show 3D Selection Tool of Convex Hull	Display the selected 3D space when a convex hull is created	N/A

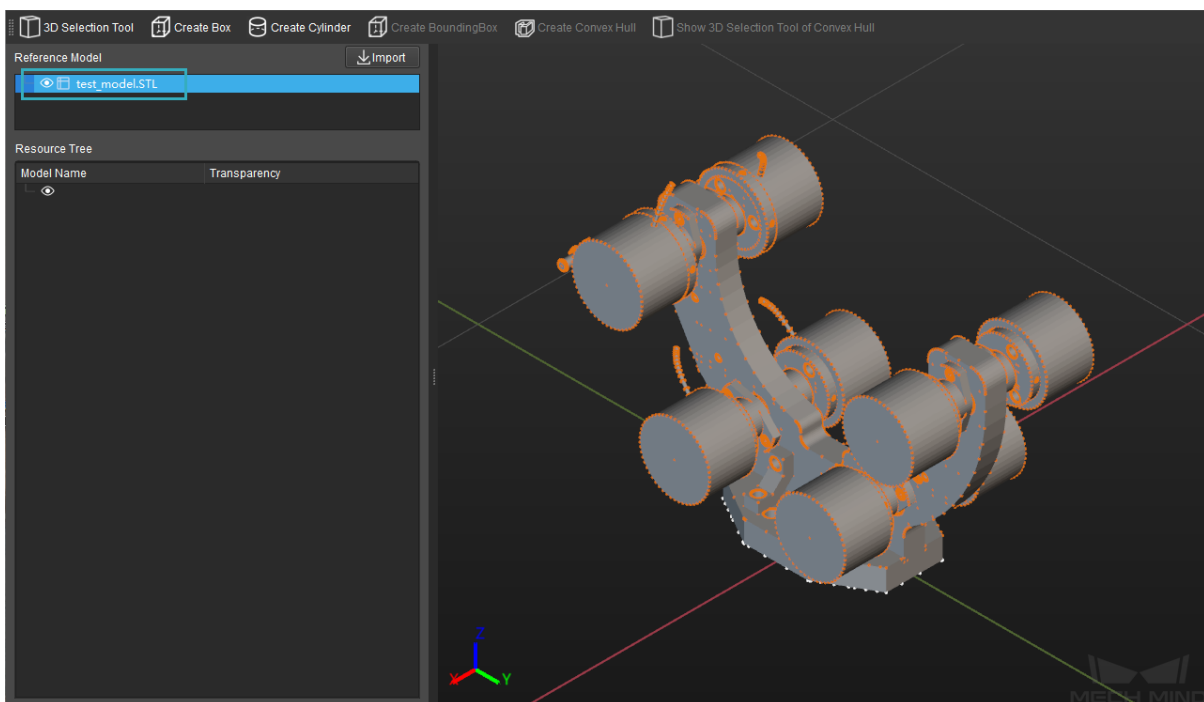
Reference Model


The name of the imported original model will be displayed in the panel as shown below.

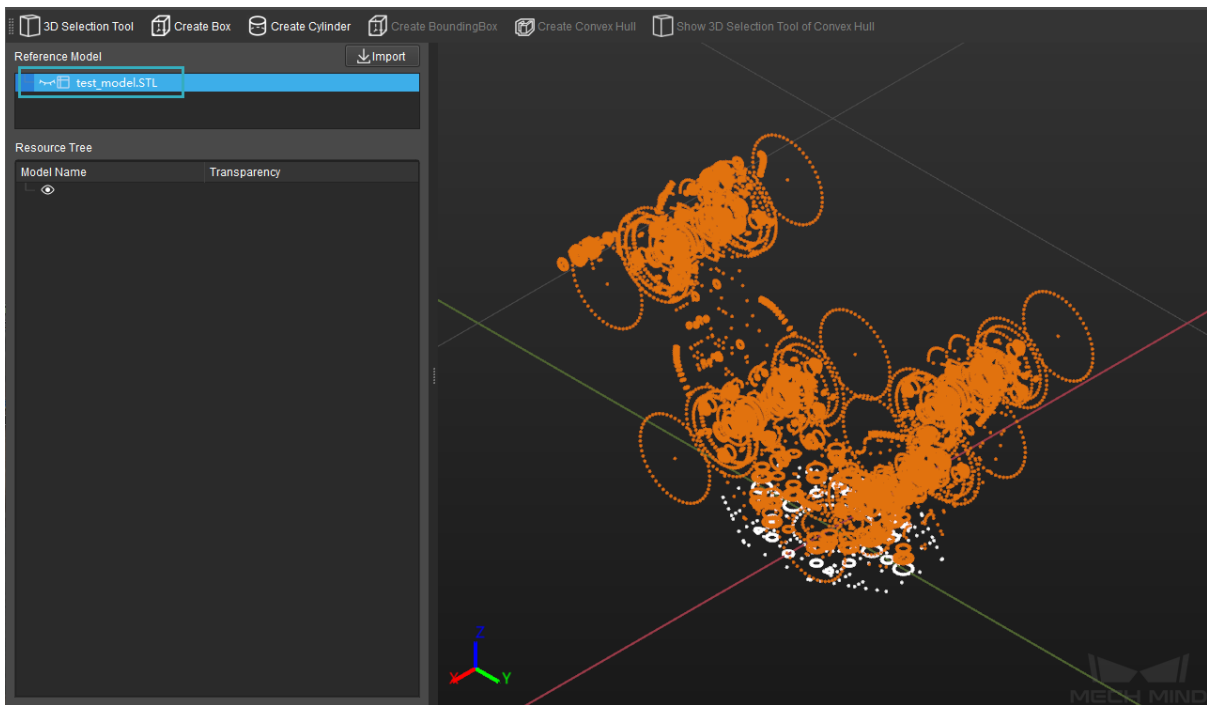
You can import models in STL, STP, STEP, and OBJ formats.



Click on the name of the reference model in the panel, and the vertices of the model will be displayed in the editing workspace, as shown below.

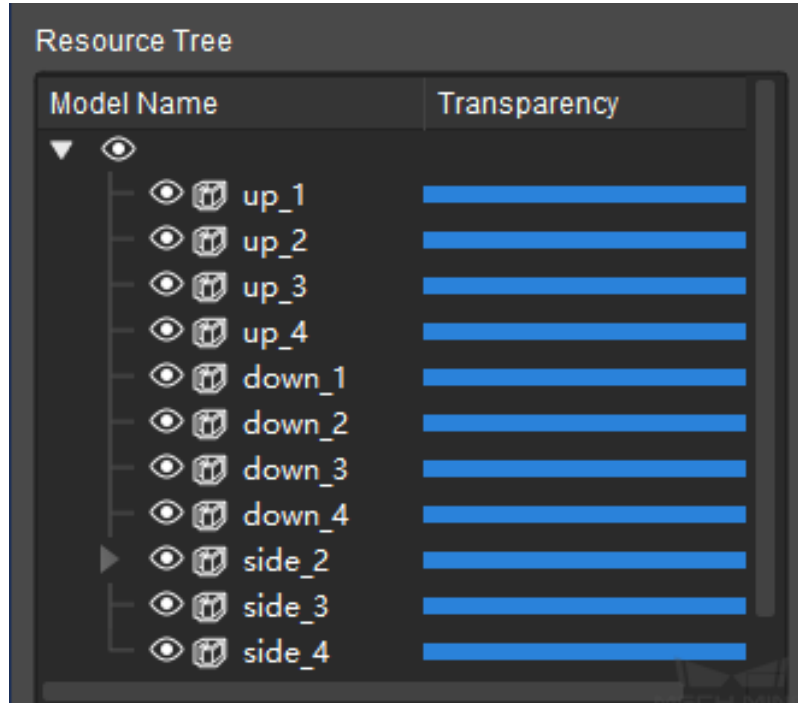


Click on the  icon next to the model name to display or hide the body of the model. The figure below is an example of displaying all the vertices and no facets of the model.



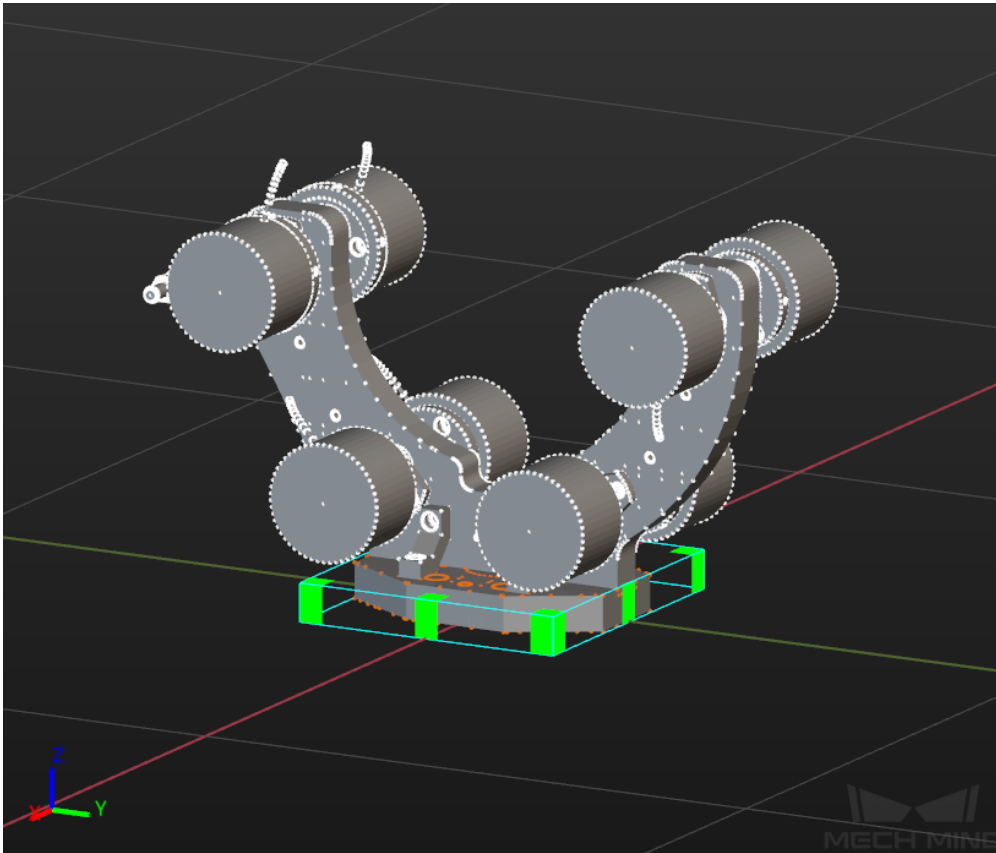
Resource Tree

As shown below, all the created geometric solids and convex hulls will be displayed in this panel, and you can adjust the transparency of them.











Editing Workspace

The editing workspace is shown below. You can edit and view the edited model here.



- Scroll up and down with the mouse to zoom in and out in the editing workspace.
- Click and hold the left button on the mouse and drag in any direction to rotate the view.
- Click and hold the scroll wheel and drag in any direction to pan the view.
- Right click the mouse and you can switch the view by selecting a view in the context menu as shown below.

	Delete Object	Del
	Front View	2
	Back View	8
	Top View	5
	Left View	4
	Right View	6
	Isometric View	7
	Switch Persp/Ortho View	9

Hope this section has provided you an overview of model editor. Please read on to find out *how to use model editor* in a general way.

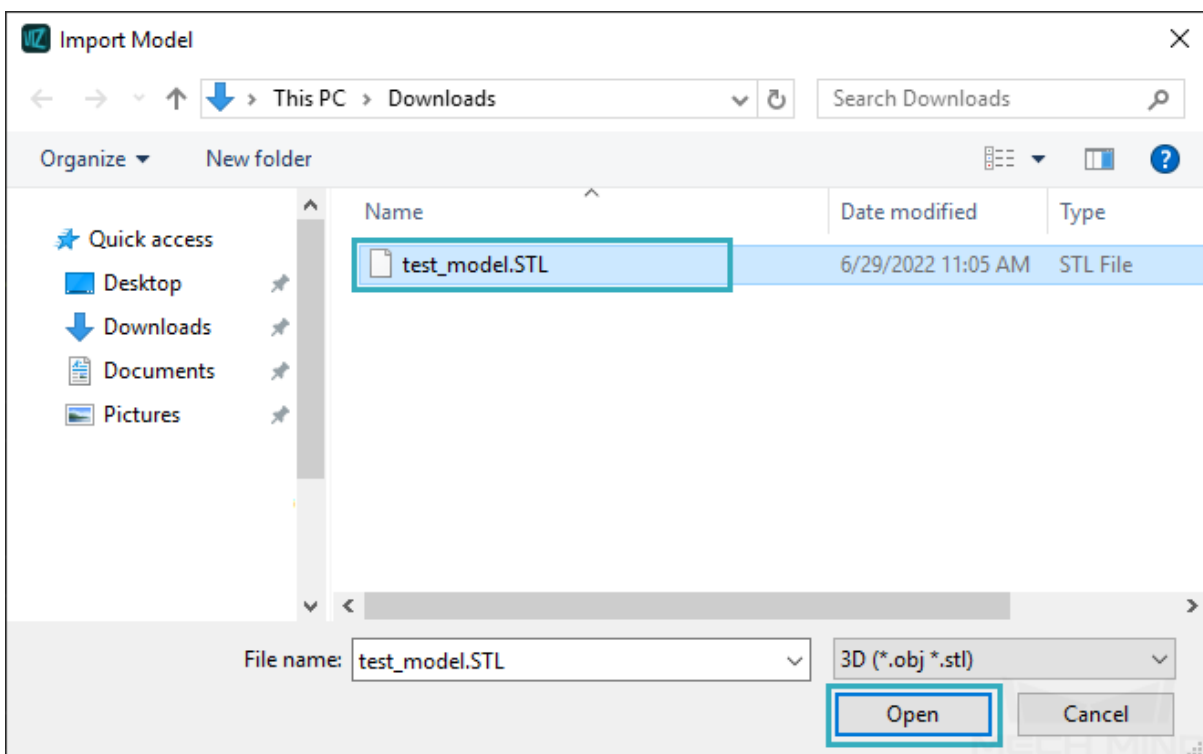
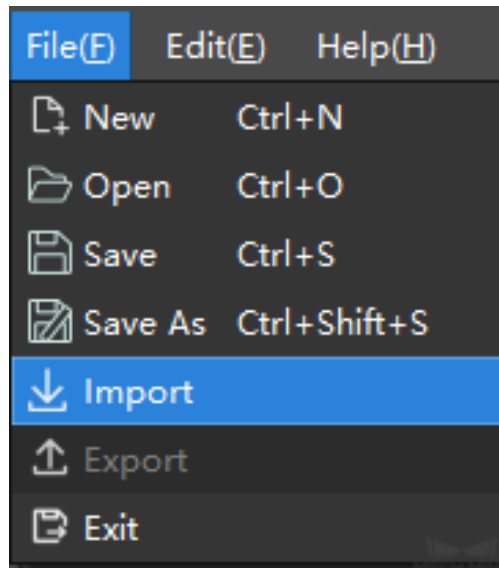
12.1.2 General Process for Using Model Editor

The general process for using the model editor mainly includes the following steps:

- *Import the Reference Model*
- *Select Vertices and Create Convex Hull*
- *Create Geometric Solids*
- *Export the Model and Save the Project*

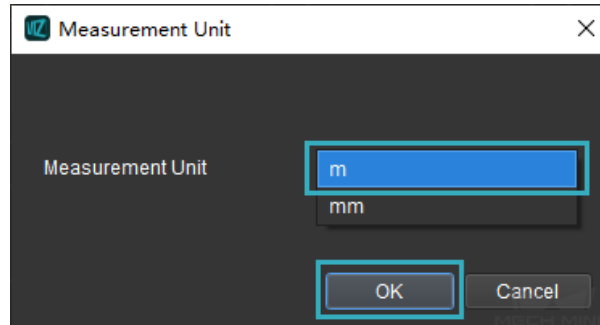
Import the Reference Model

1. Go to *File* → *Import* in the menu bar, or click on the *Import* button in the **Reference Model** panel, and then select the needed model file in the pop-up window, and click on *Open*.

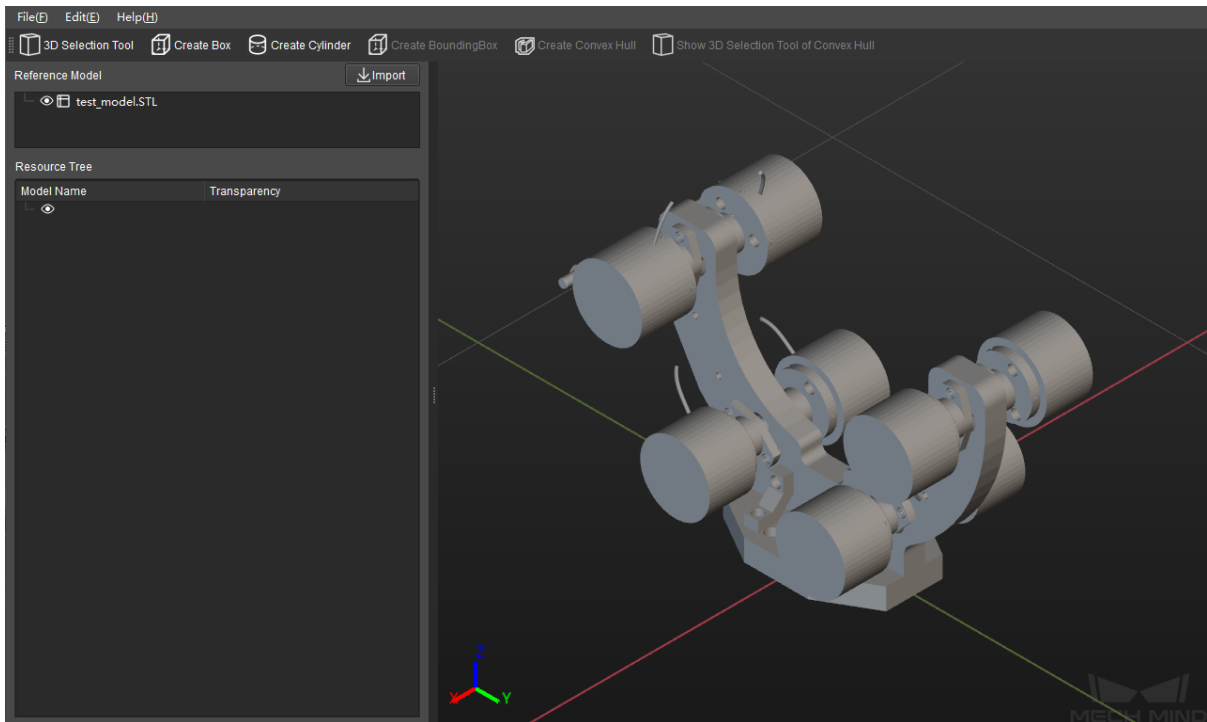


Hint: You can import models in STL, STP, STEP, and OBJ formats.

2. Select the proper unit (m or mm) of the model in the pop-up **Measurement Unit** window, and then click on *OK*.

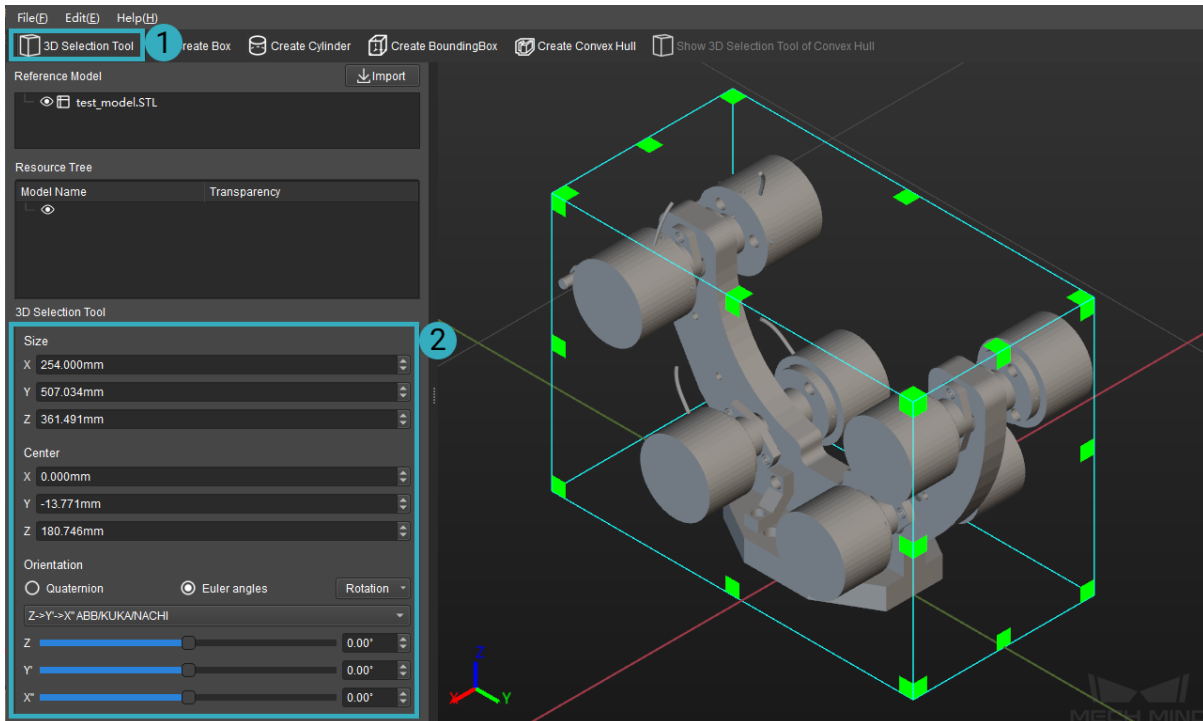


3. The model will be displayed in the editing workspace if it is imported successfully, as shown below.



Select Vertices and Create Convex Hull

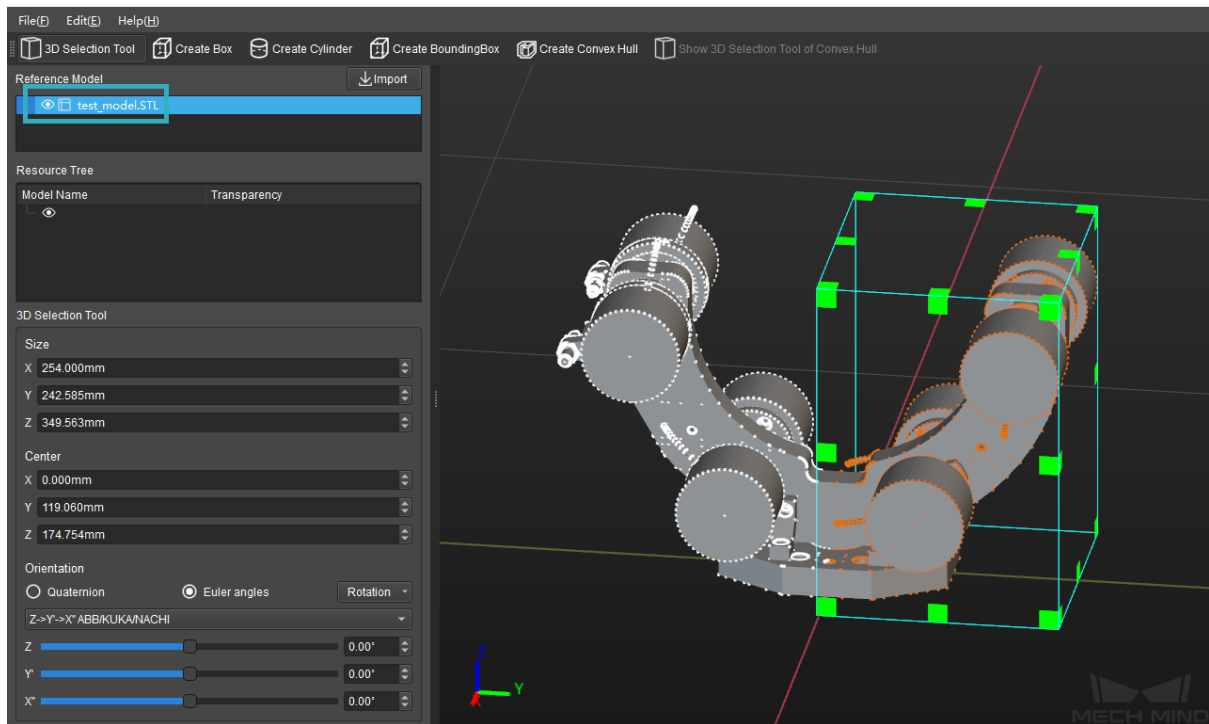
1. Click on *3D Selection Tool* to create a cuboid-shaped selection frame. You can adjust the parameters in the panel on the left to adjust the dimensions and position of the 3D frame, and you can also use the mouse and keyboard to adjust its dimensions and position more intuitively.



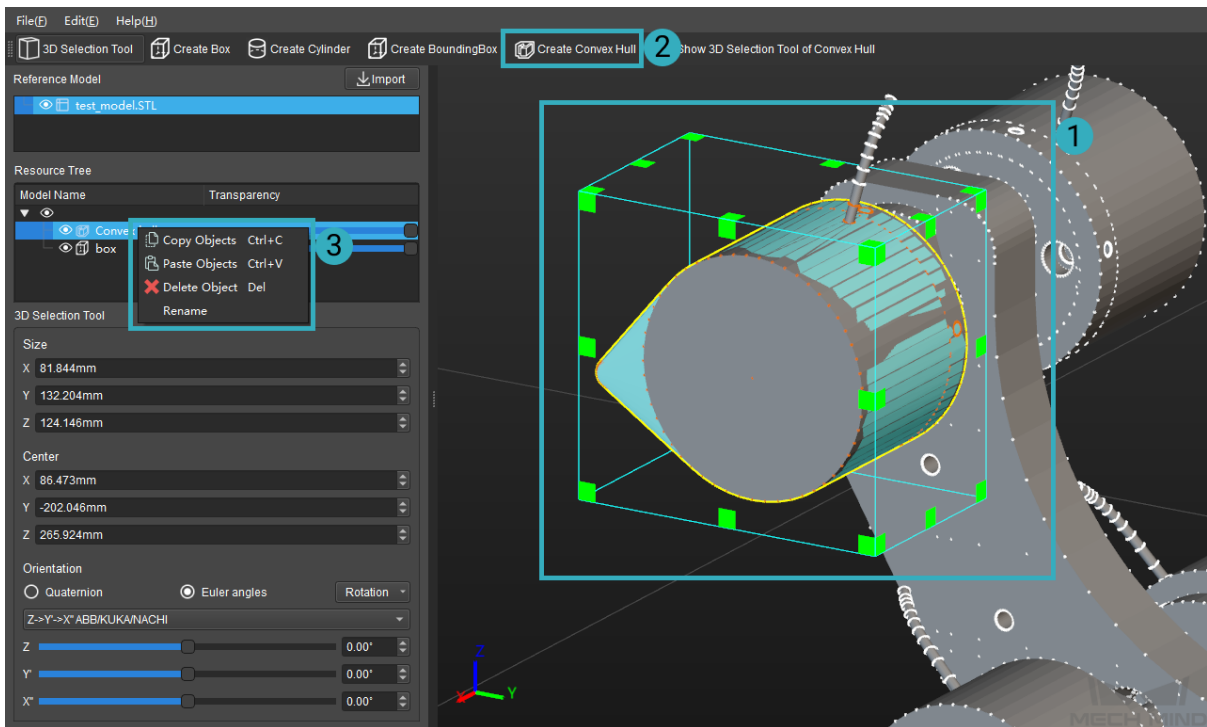
Hint:

- Press and hold the **Ctrl** key, hold down the left mouse button and drag the vertices of the 3D frame to adjust the dimensions.
- Press and hold the **Ctrl** key, hold down the left mouse button and drag the surfaces of the 3D frame to adjust the position.

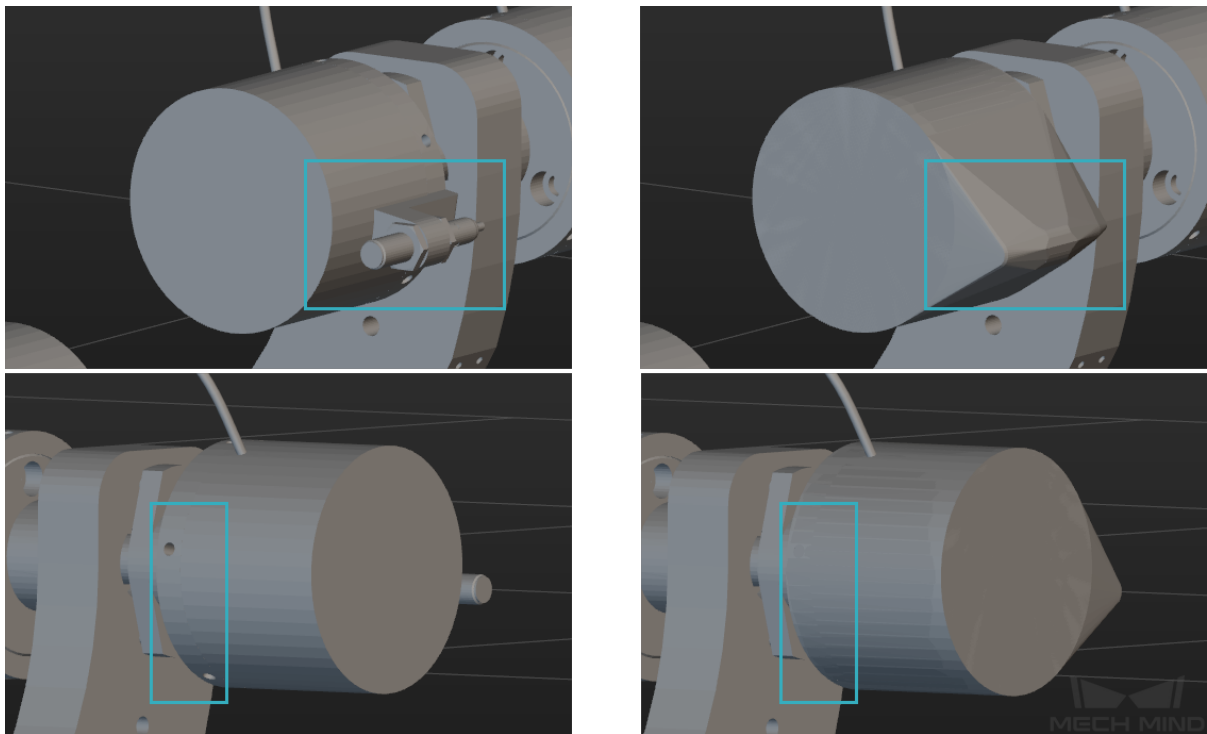
2. Click on the name of the model in the **Reference Model** panel, the vertices of the model will be displayed in the editing workspace. The vertices inside the 3D frame will be displayed in orange, and the vertices outside the 3D frame will be displayed in white, as shown below.



- Adjust the position and dimensions of the 3D frame to select certain part of the model, and then click on *Create Convex Hull* to simplify the part. The created convex hull will be displayed in the **Resource Tree** panel. Right click on the name of the convex hull, and you can copy, paste, delete, and rename it by selecting the option you need in the context menu.

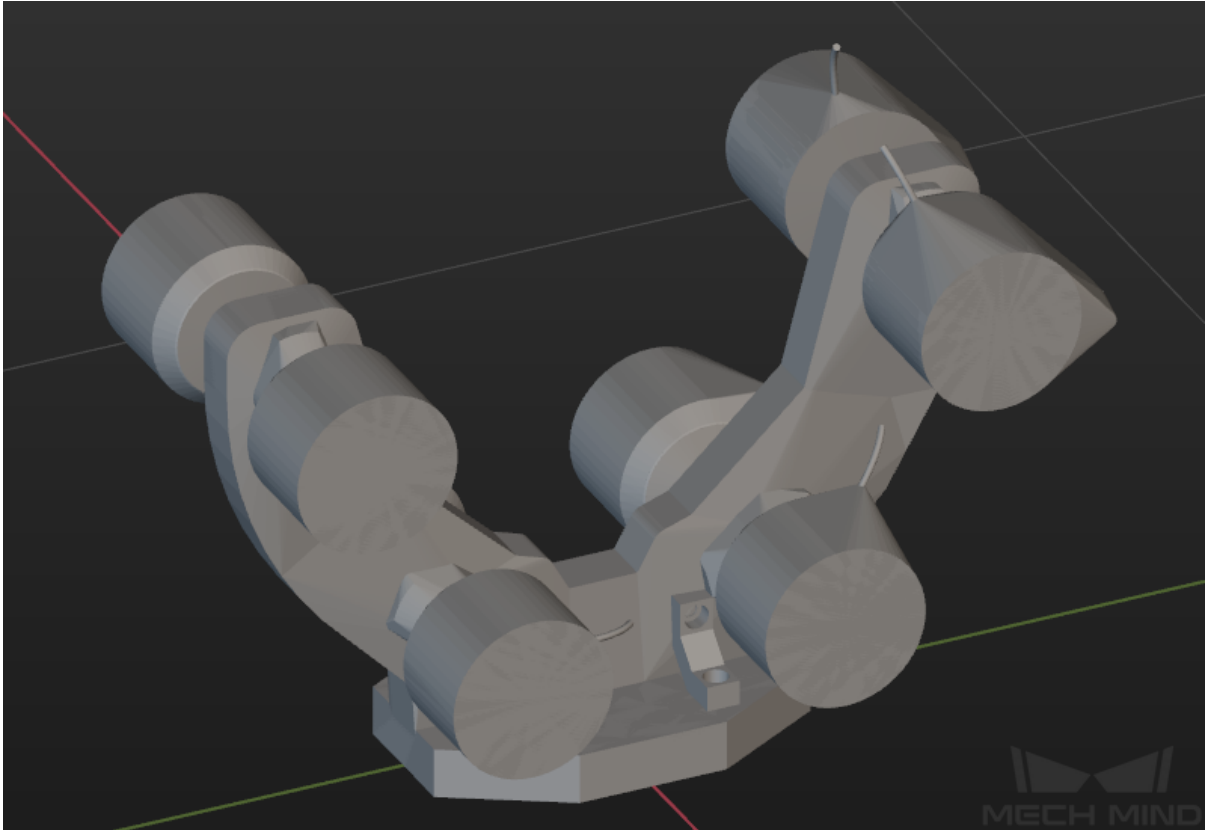


The figure below shows an example of a part of the model before and after being simplified.



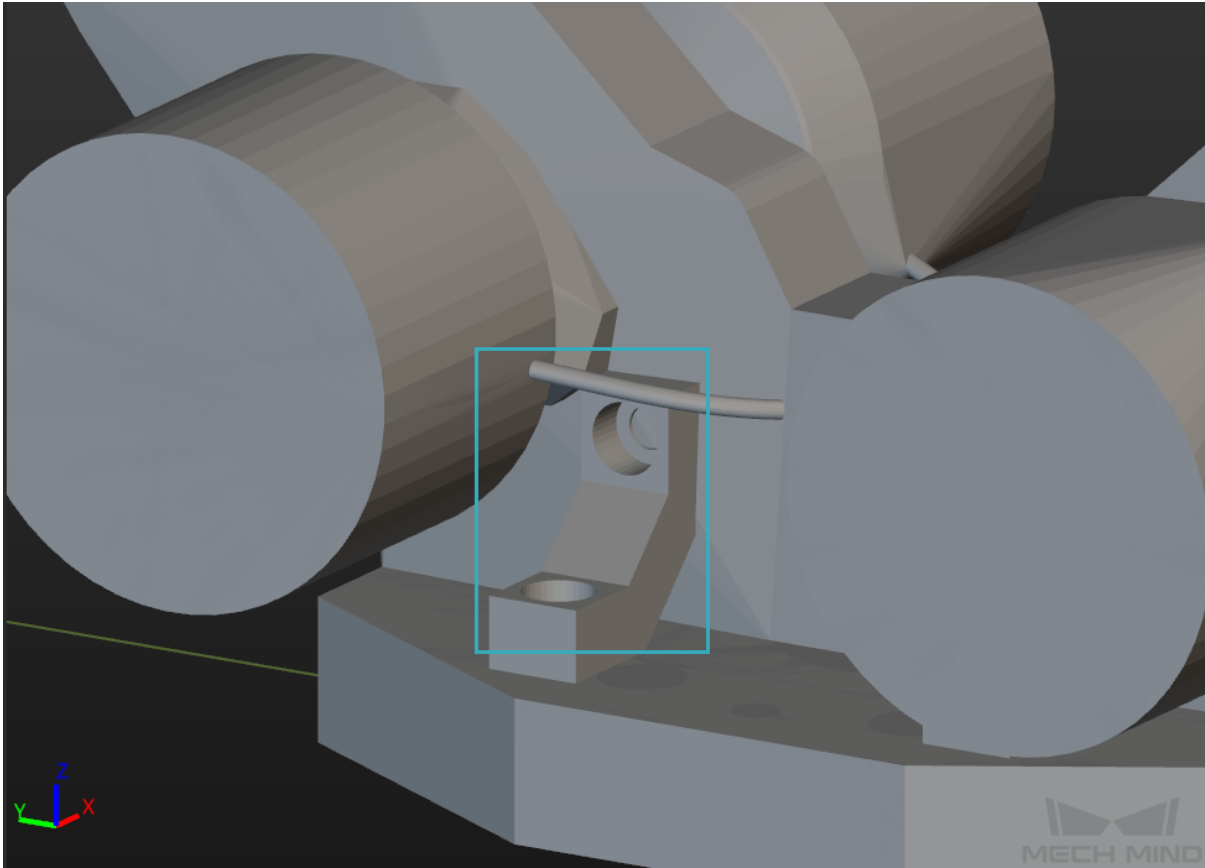
Hint: Please refer to *STL Models Simplification* and *OBJ Models Simplification* for basic methods of simplifying models.

4. Create more convex hulls to simplify the model, as shown below.



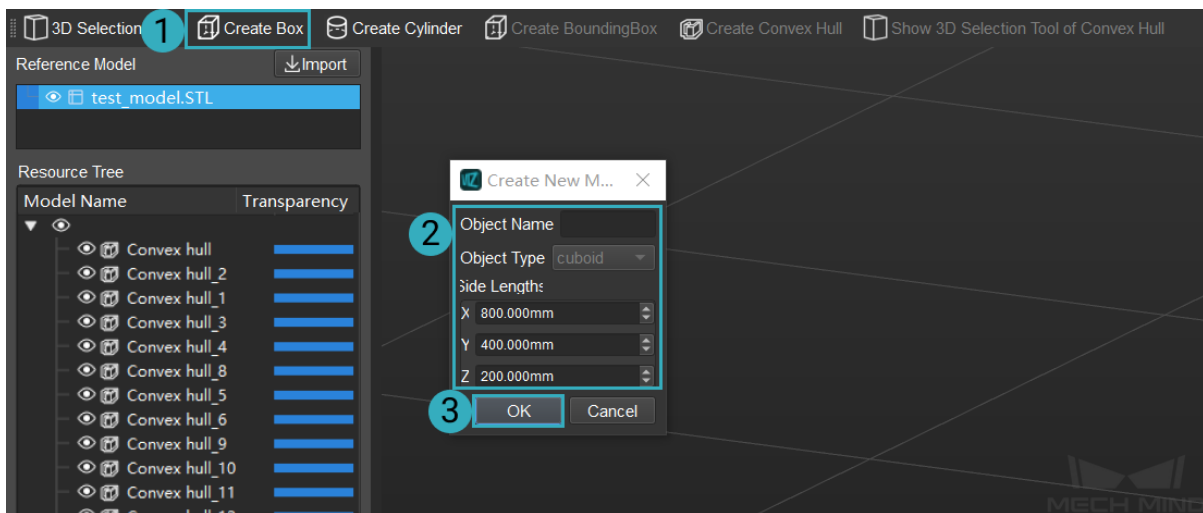
Create Geometric Solids

Although it is convenient to create convex hulls with 3D selection tool to simplify the model, some parts with complex structure of the model cannot be selected with a 3D cuboid-shaped frame in a simple way, as shown below.

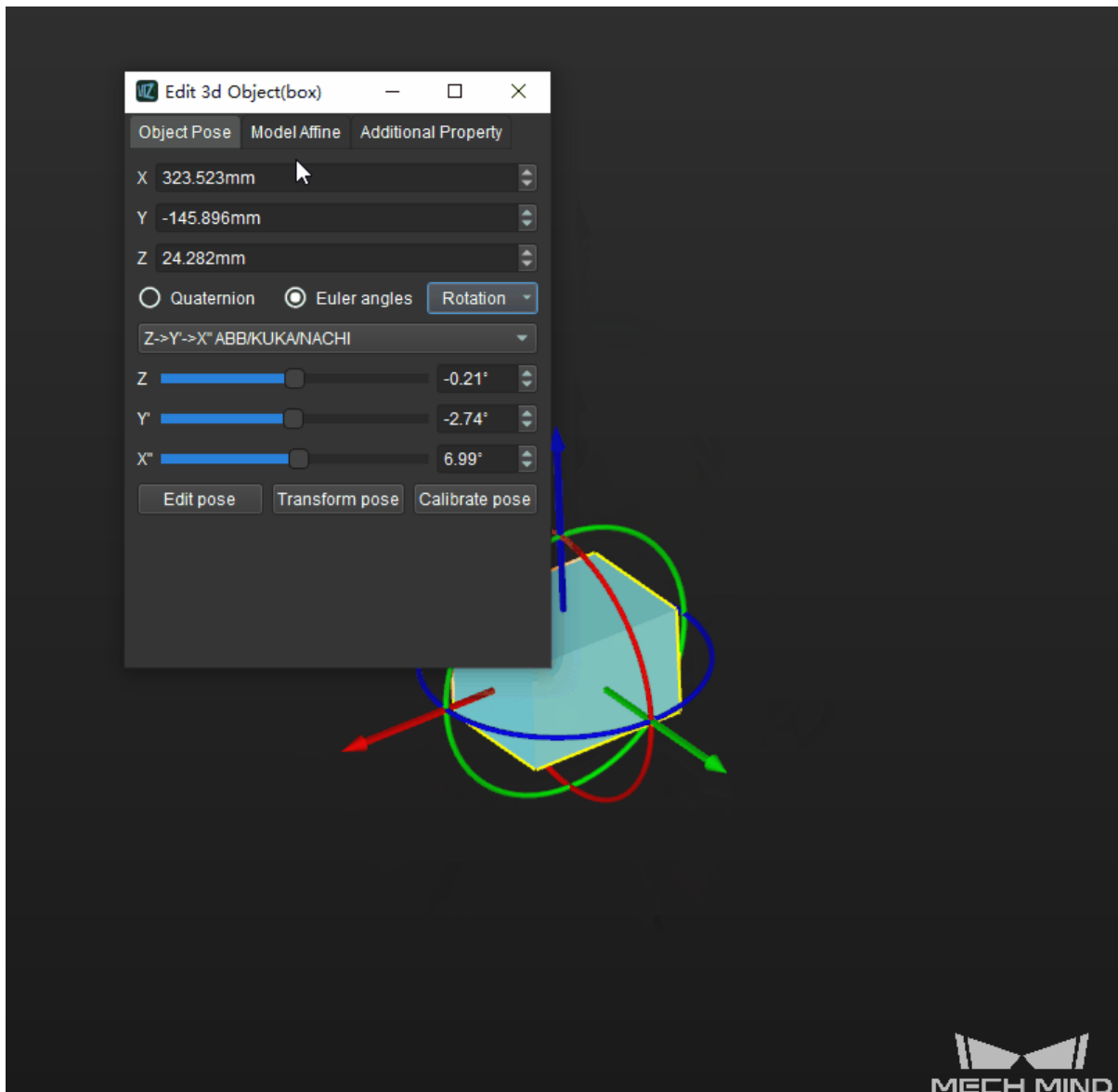


For such parts with complex structures, you can create some geometric solids with similar shape to replace them. The detailed instructions are as follows.

1. Click on *Create Box*, name the object and set the side lengths in the pop-up **Create New Model** window. The side lengths can be an approximate one, for you can fine-tune them in the following steps. Click on *OK* to save the settings.

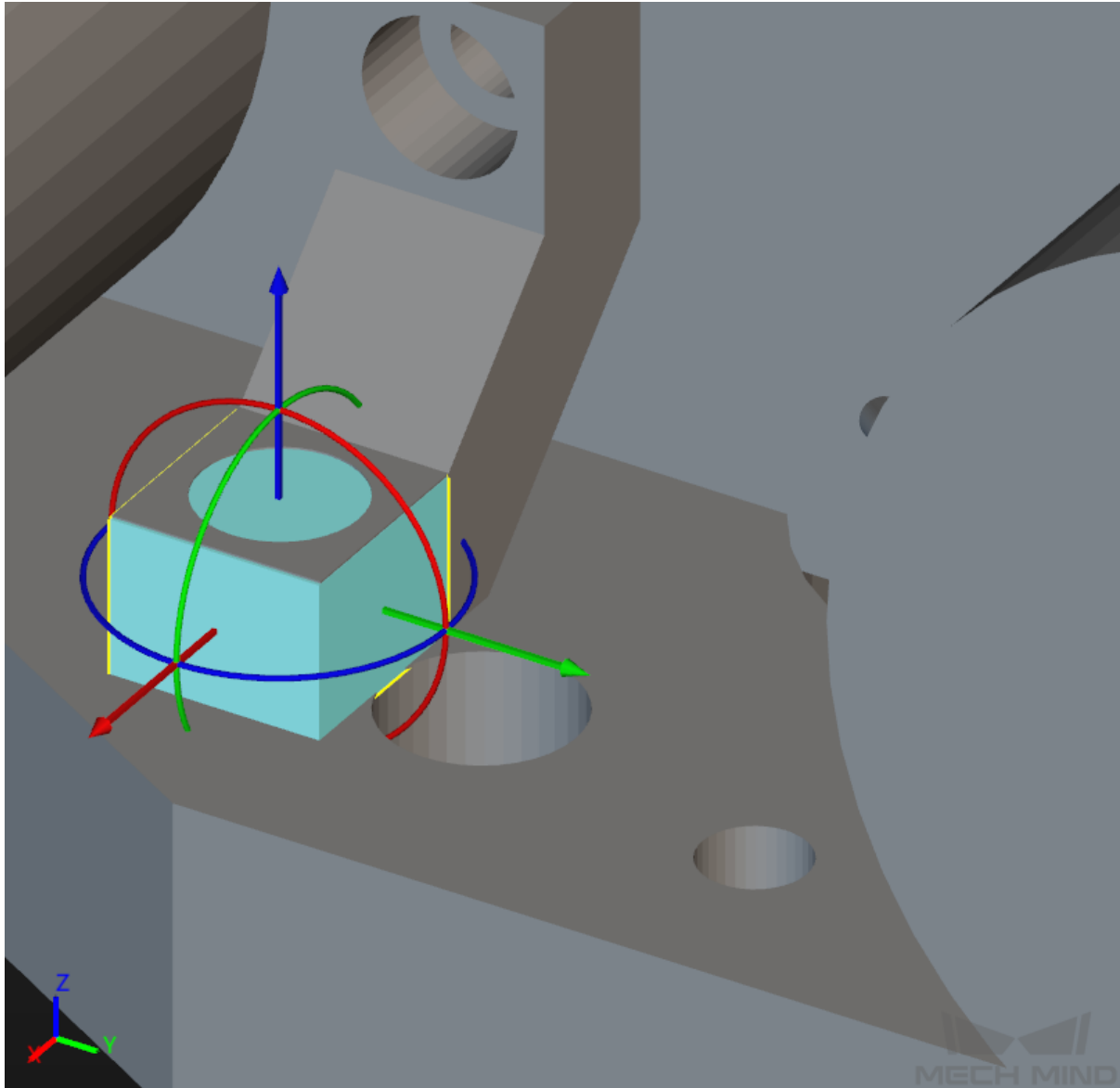


2. As shown below, click on the cuboid to display the dragger. Press and hold the **Ctrl** key, and drag the dragger with left mouse button to translate and rotate the cuboid. Double click on the cuboid, and you can configure relevant parameters of the cuboid in the **Edit 3D Object** window.

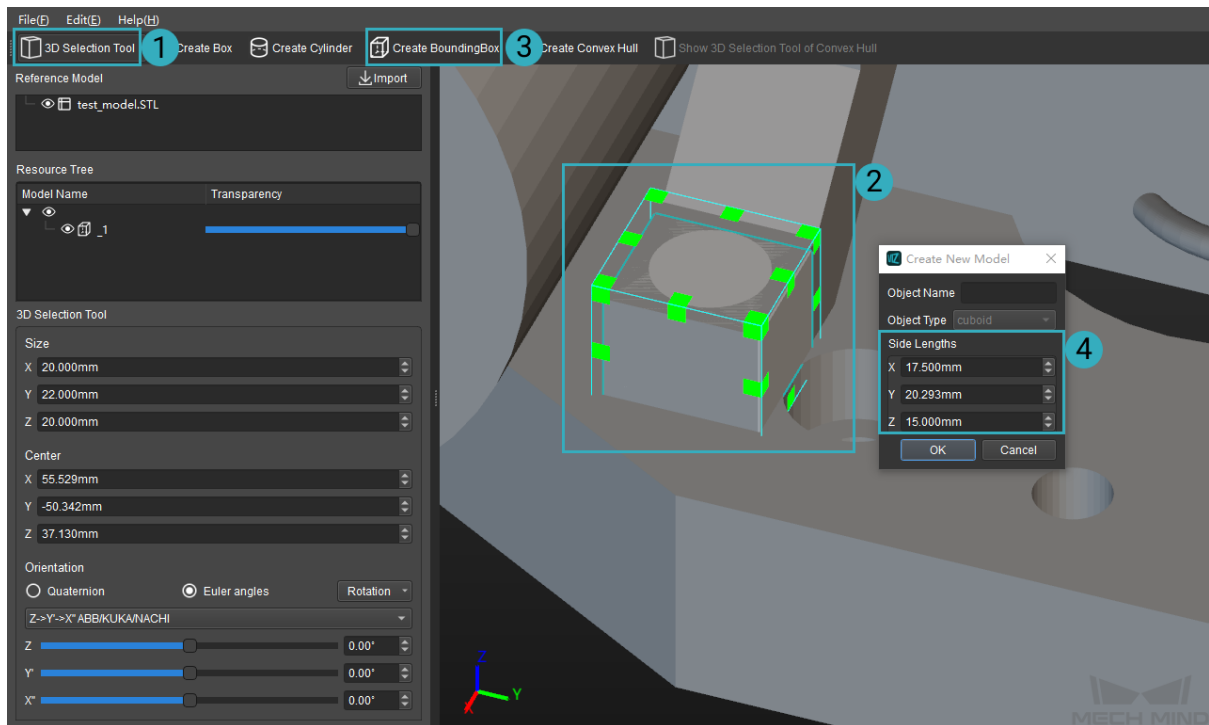


Hint: The way to configure the parameters of the created geometric solids in the **Edit 3D Object** window is the same as that of the scene objects. Please refer to [Scene](#) for detailed information.

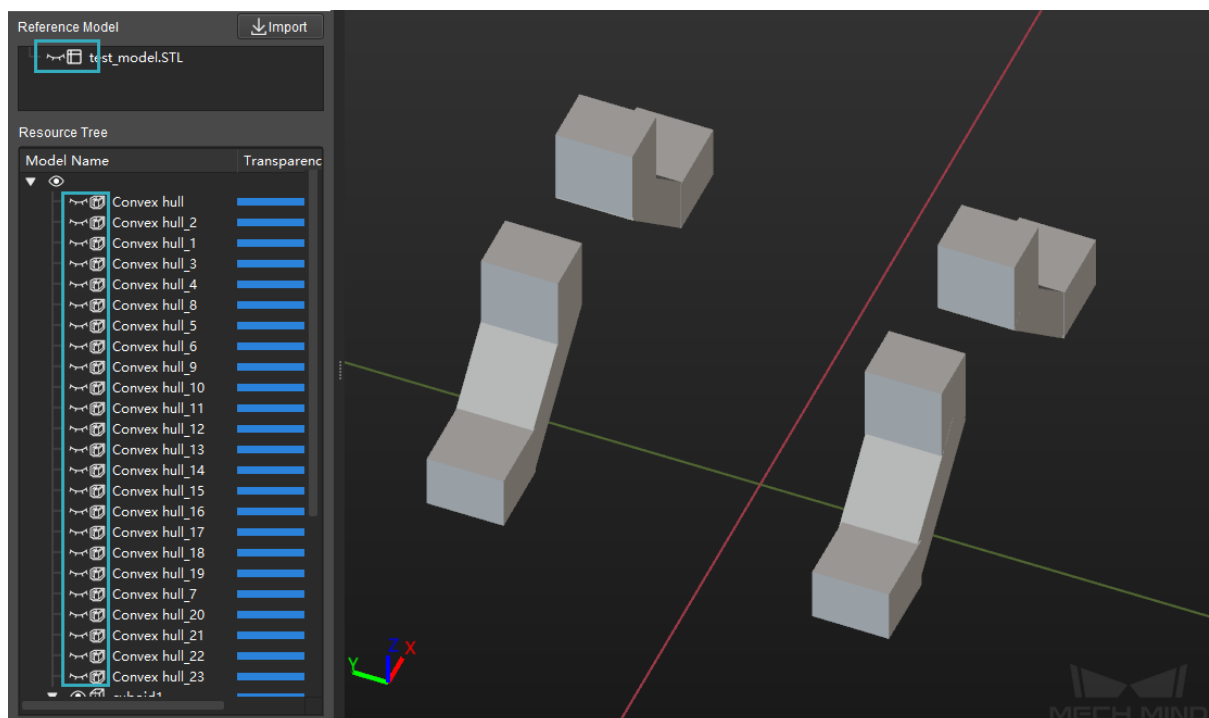
3. Drag the cuboid to the part with a hole on it. Adjust the side lengths and position of the cuboid to make it almost the same size as the part to be filled and therefore fill the hole.




You can also create a bounding box to estimate the dimensions of the cuboid. Click on *3D Selection Tool*, select the part to be estimated, and then click on *Create BoundingBox*. Then you can view the approximate dimensions of the part.



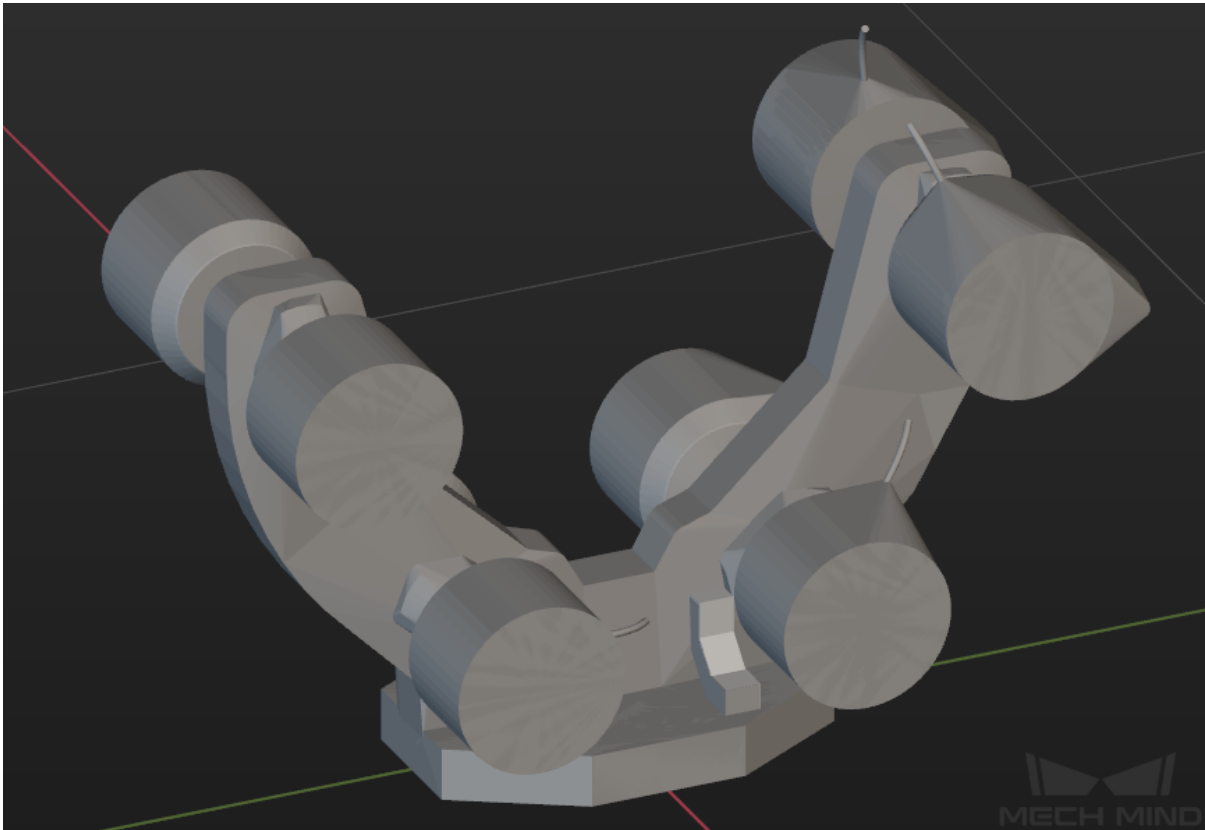
4. Fill other small holes on the model in the same way. The final result is shown below.



Tip: In order to view the edited part more clearly, you can click on the  icon next to

other convex hulls or the model to hide the irrelevant parts.

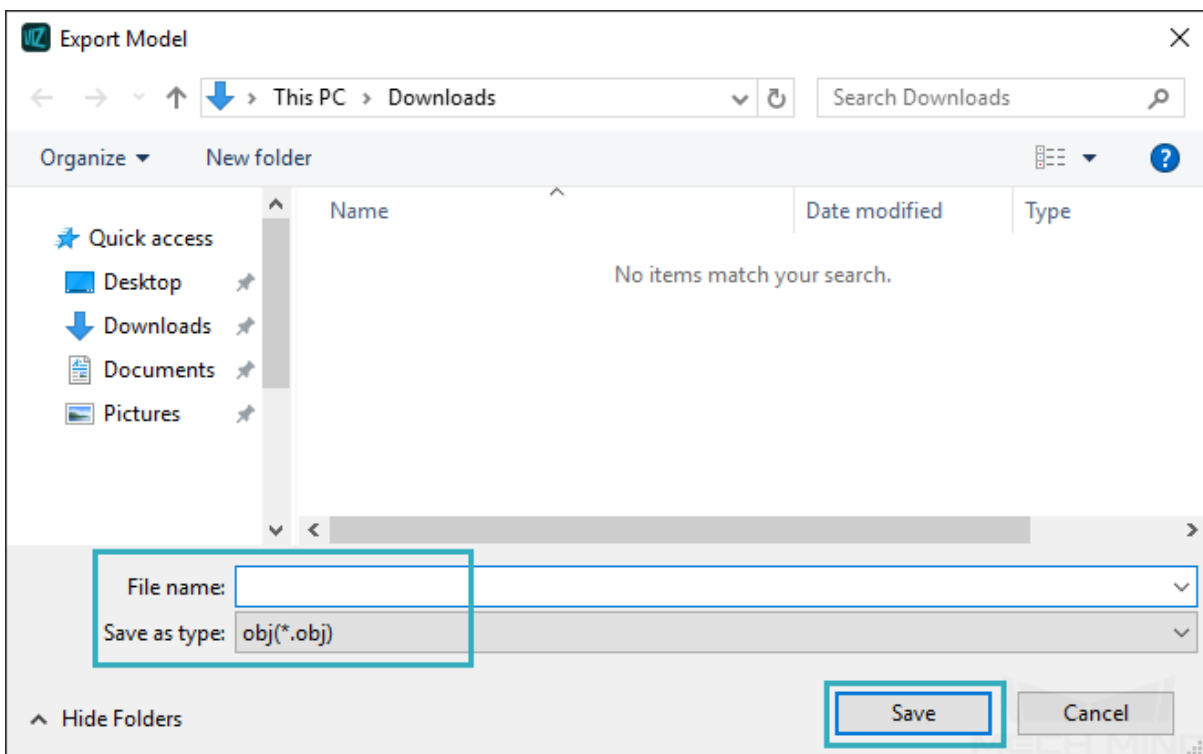
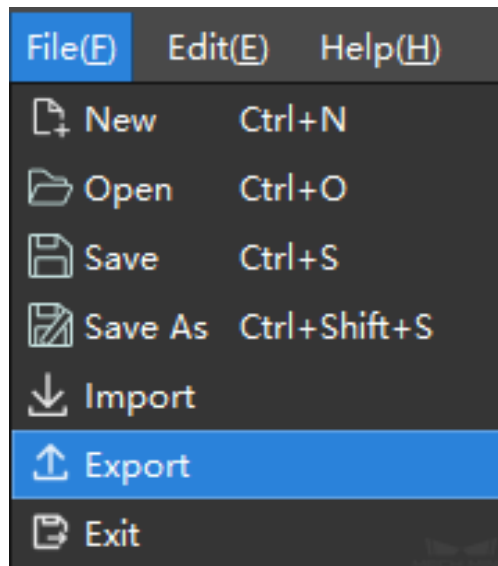
5. The figure below is an example of the simplified model after the complex parts have been filled with geometric solids.



Export the Model and Save the Project

Export the Simplified Model

Select *File* → *Export* in the menu bar, select a directory to save the model, name the model and select the file type (STL or OBJ), and then click on *Save* to export the simplified model.

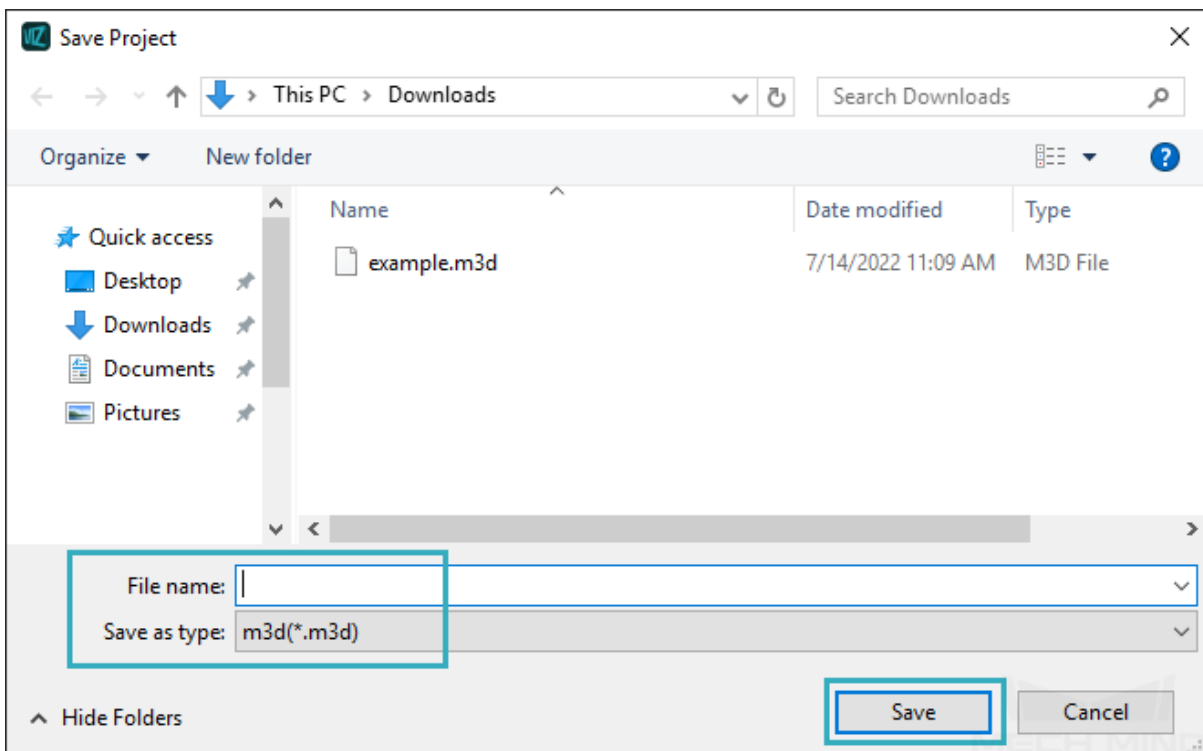
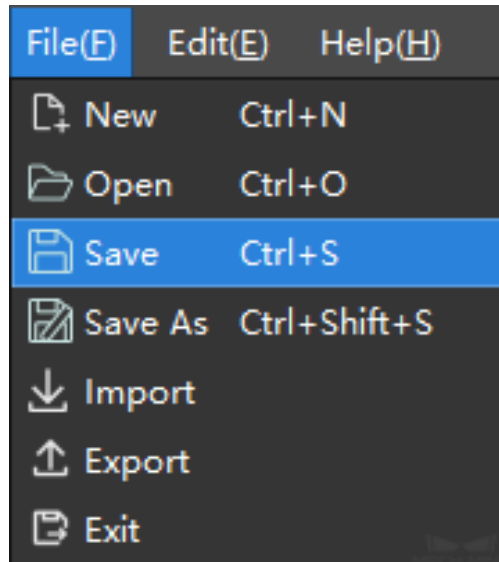


Hint: The exported model can be added as an end effector model or collision model. Please refer to [End Effector Configuration](#) for detailed instructions.

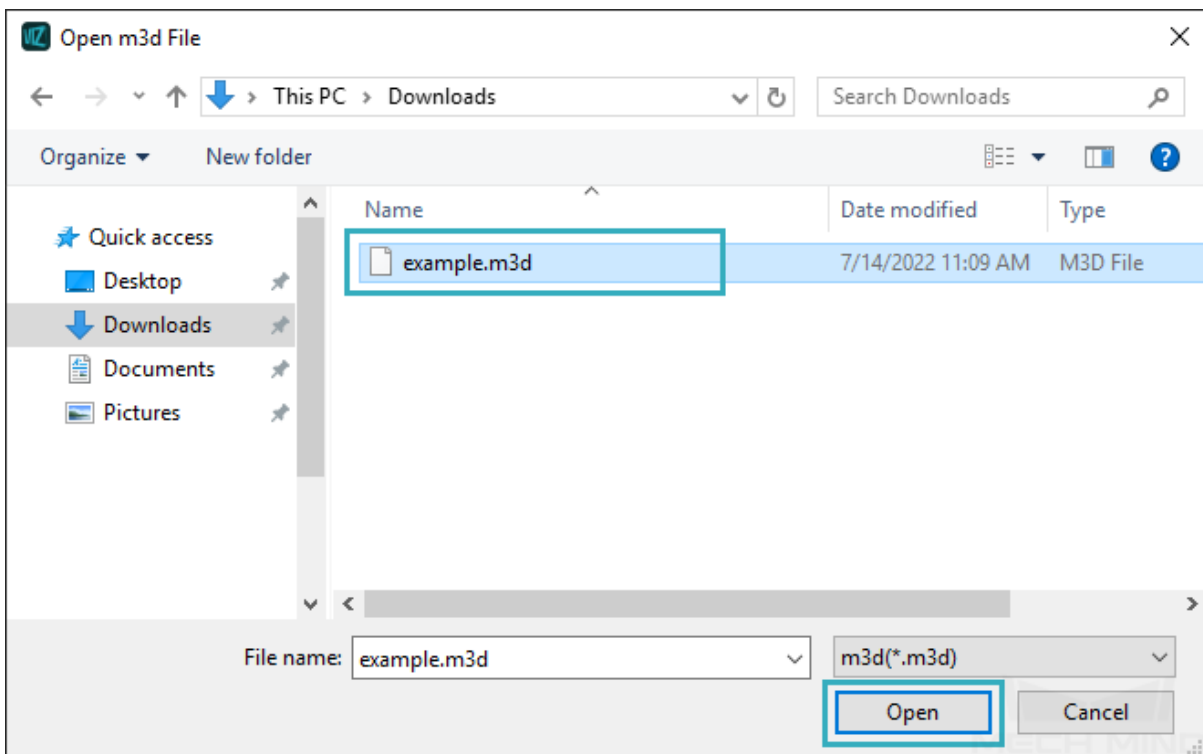
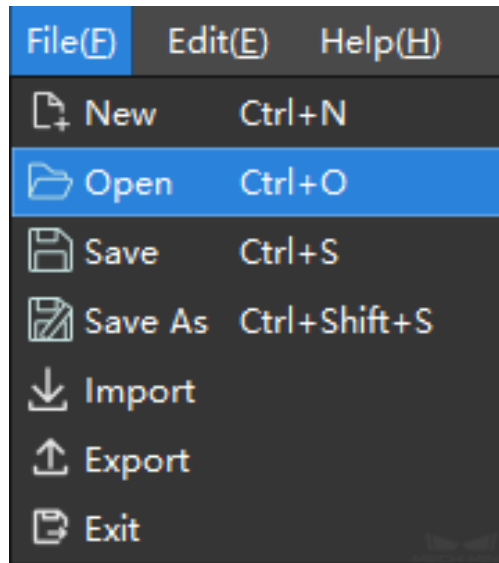
Save the Project

In order to facilitate further editing on the simplified model, you can save the editing project as well. The saved project is in M3D format, which contains all editing history, and you can open it in the model editor and re-edit the model.

Select *File* → *Save*, select a directory to save the model, name the model and then click on *Save* to save the project as a M3D file.



If you need to re-edit the model, go to *File* → *Open*, select the M3D file in the pop-up window and click on *Open* to open the project.



Hint: After re-editing the model, clicking on *Save* will save all changes to the existing M3D file, while clicking on *Save As* will save the file as a new M3D file, and you can specify a new directory to save it.

12.2 Vacuum Gripper Configurator

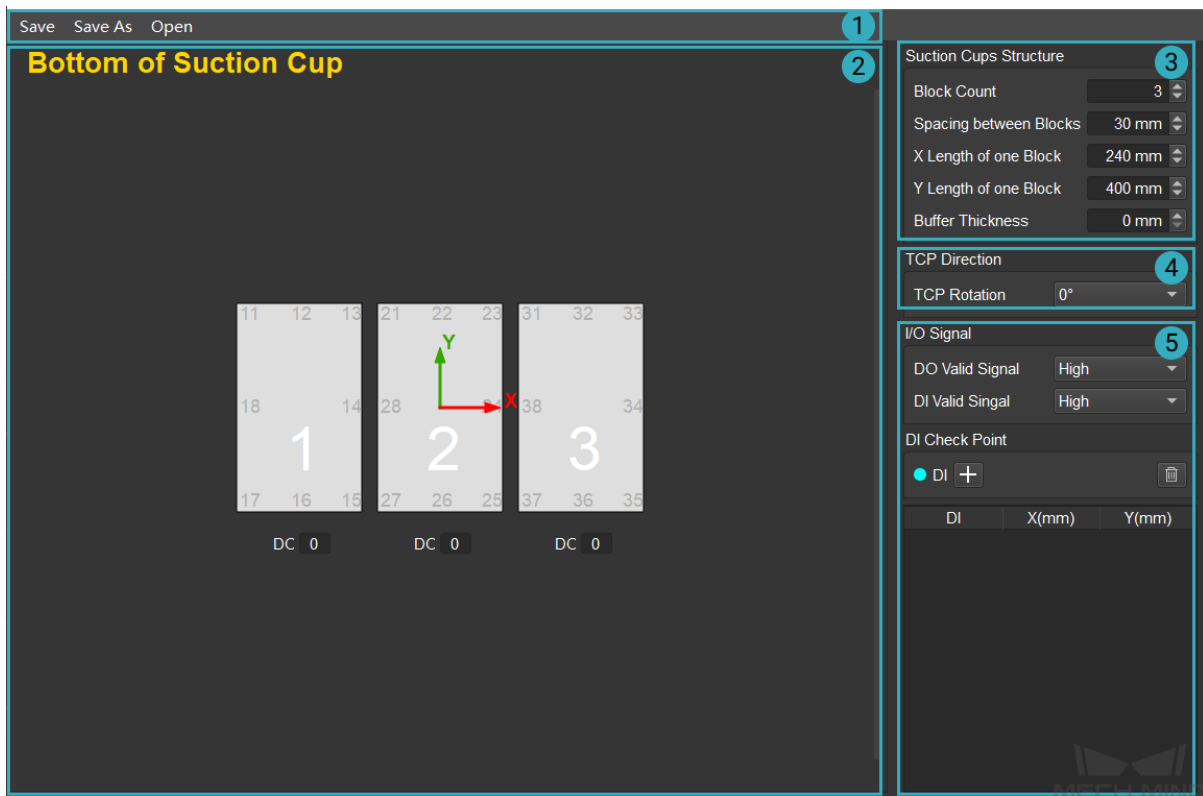
Vacuum gripper configurator supports to configure vacuum grippers of rectangular shape, one block, multi-blocks, and in a single row. Vacuum grippers of round shape, multi-blocks, or in multiple rows are currently not supported.

After configuration, you can use the configuration file in corresponding Tasks and open the configuration file in the vacuum gripper configurator as well.

Hint:

- Please configure according to the actual vacuum gripper of the robot.
- For vacuum grippers that are horizontally installed, multiple blocks structure and DI check are not supported.

12.2.1 Introduction of the Interface



The main interface of vacuum gripper configurator consists of 5 parts:

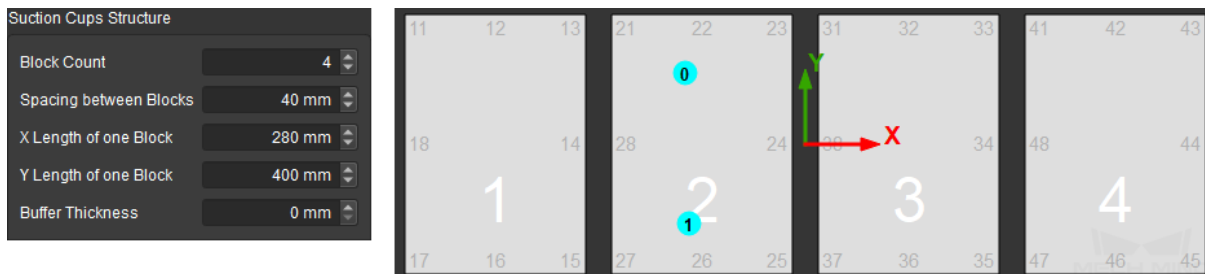
1. Menu Bar
2. Visualization Area

3. Vacuum Gripper Configuration Panel
4. I/O Signal Configuration Panel
5. TCP Direction Configuration Panel

12.2.2 Vacuum Gripper Configuration

1. Configure the structure of suction cups

Set the **Block Count**, **Spacing between Blocks**, **X Length of one Block**, and **Y Length of one Block** in the **Suction Cups Structure**, and the structure of suction cups on the bottom side will be displayed in the **Visualization Area** in real time, as shown below.

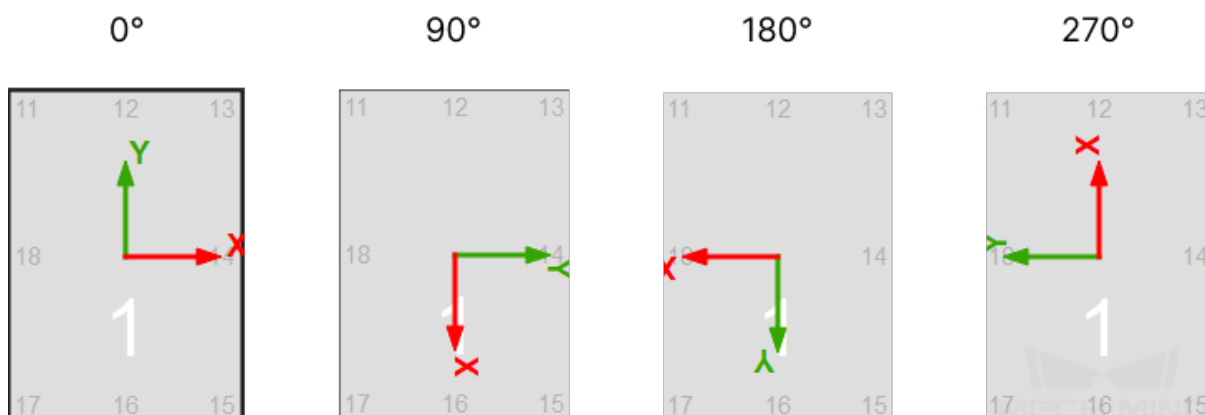


Buffer Thickness is used to set the thickness of the buffer used for collision detection. One side of the buffer is pressed against the plane where the TCP is. The buffer thickness of the operative vacuum gripper block will be the same as the set thickness, while the buffer thickness of inoperative vacuum gripper block will be 0 and it will not be included in the collision detection.

Hint: For multiple blocks that share one DO, they can be viewed as a whole block. You can create one large block to represent them in the vacuum gripper configurator.

2. Configure TCP direction

Different TCP directions correspond to different reference frames, as shown below. Please select according to actual needs.



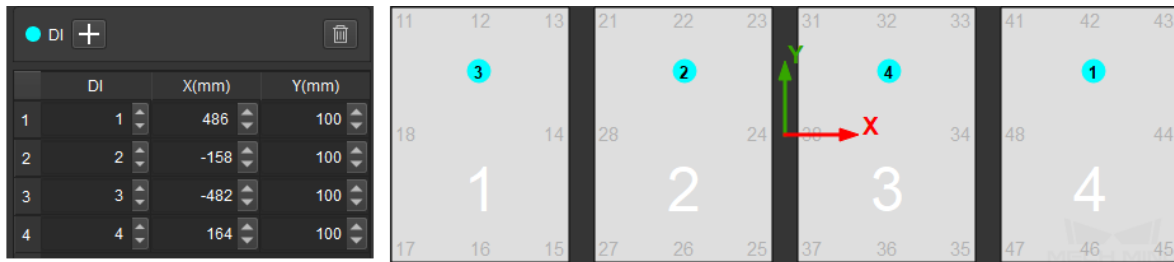
3. Configure I/O signal

Please choose between **High** and **Low** according to the actual voltage level of the DI and DO.

4. Configure DI check point

Click on + in the **DI Check Point Panel** to add DI check points. The default value of the DI is 0 and the check point is located in the center of the vacuum gripper.

Modify the name of the check point in the **DI** column. Drag the check point to a proper position, and the coordinates of the check point will be displayed in the **DI Check Point Panel** in real time. Also, you can modify the coordinates of the check point to move it as well, as shown below.



If you need to delete the check point, select it and click on .

5. Set the number of DO

Enter the number in the box under the corresponding vacuum gripper in the visualization area.

12.2.3 Save or Utilize the Configuration File

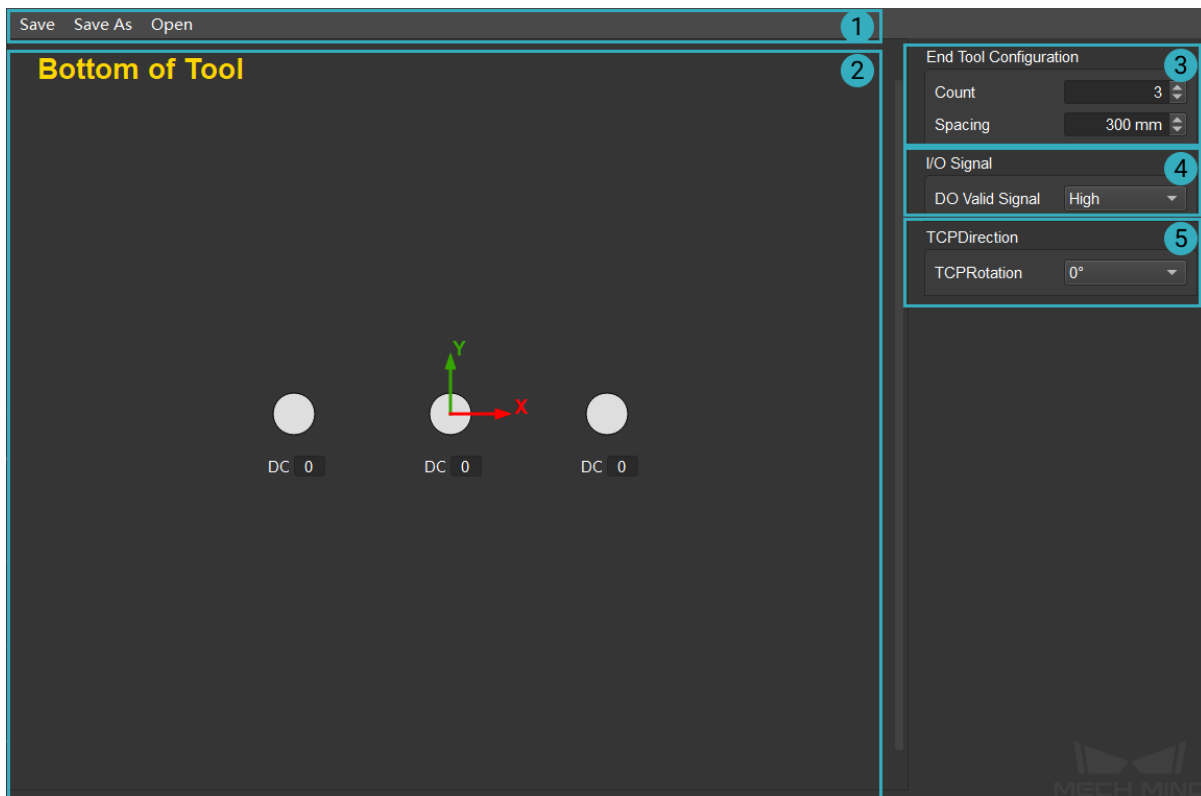
After configuration, click on *Save* or *Save As* to save the JSON file to a specified directory.

Click on *Open* to open an existing vacuum gripper configuration file in JSON format.

12.3 Array Gripper Configurator

Array gripper configurator is used to visualize and configure the array of the tools at the end of the gripper.

12.3.1 Introduction of the interface

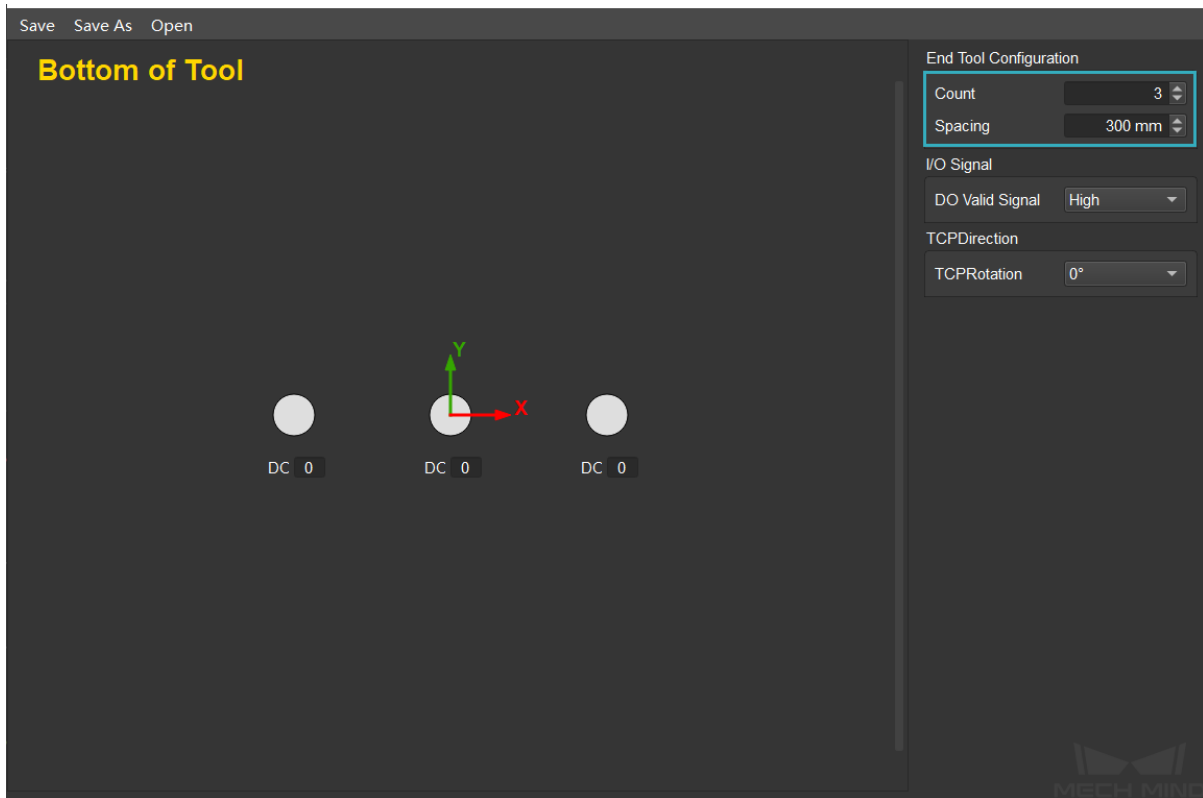


The main interface of array gripper configurator consists of 5 parts:

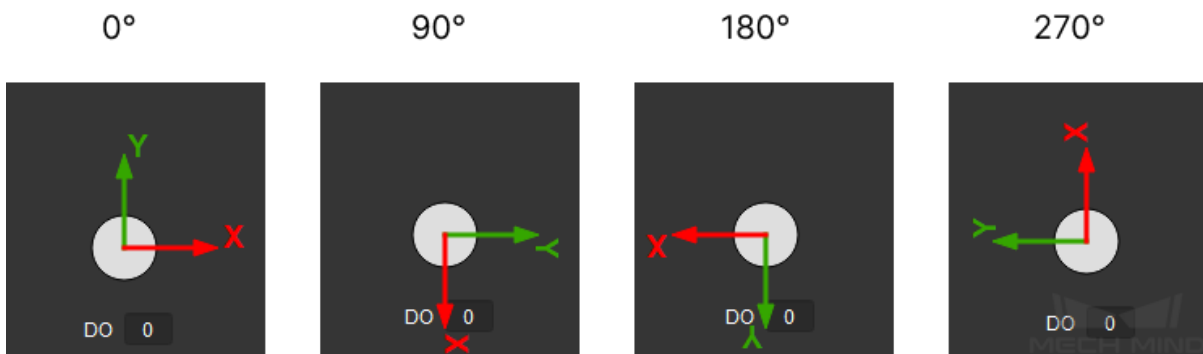
1. Menu Bar
2. Visualization Area
3. End Tool Configuration Panel
4. I/O Signal Configuration Panel
5. TCP Direction Configuration Panel

12.3.2 Tool Array Configuration

1. Set the **Count** and **Spacing** in the **End Tool Configuration Panel**, and the tool array at the bottom will be displayed in the **Visualization Area** in real time, as shown below.



2. Please choose between **High** and **Low** according to the actual voltage level of the DO.
3. Please set the TCP direction according to actual situation. The relationship between the TCP direction and reference frame is shown below.



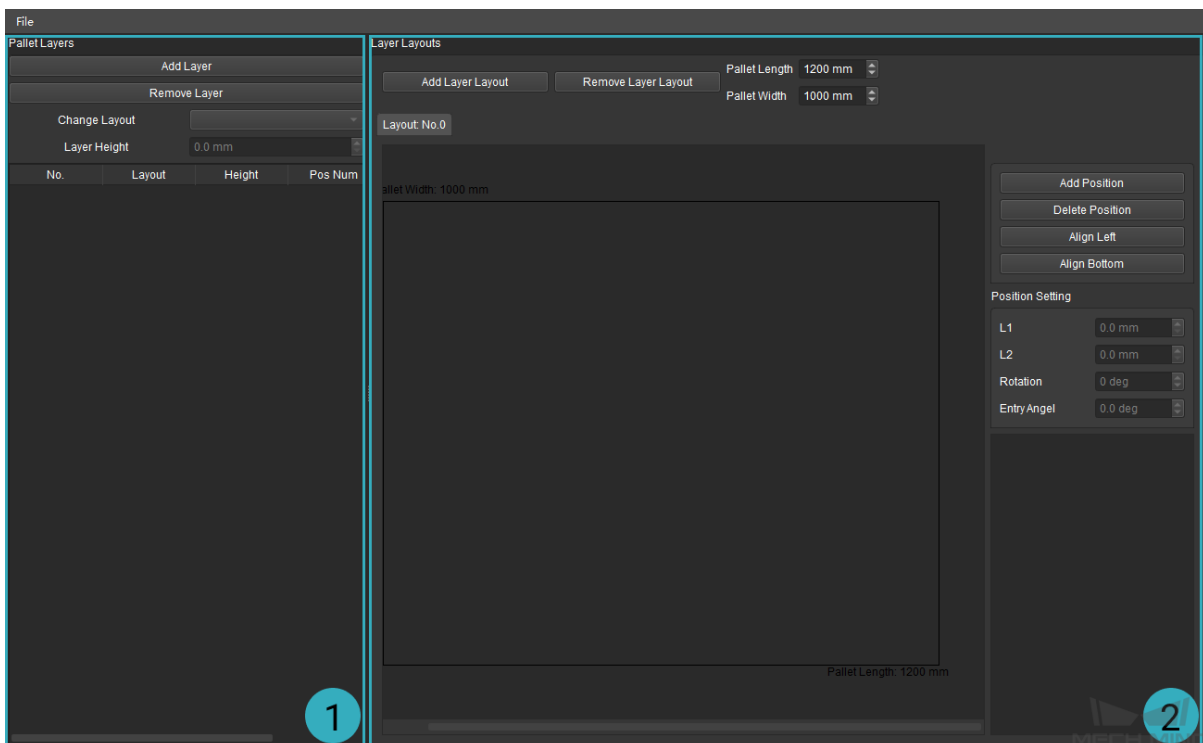
12.3.3 Save or Utilize the Configuration File

After configuration, click on *Save* or *Save As* to save the JSON file to a specified directory. Click on *Open* to open an existing tool array configuration file in JSON format.

12.4 Pallet Editor

12.4.1 Introduction of the Interface

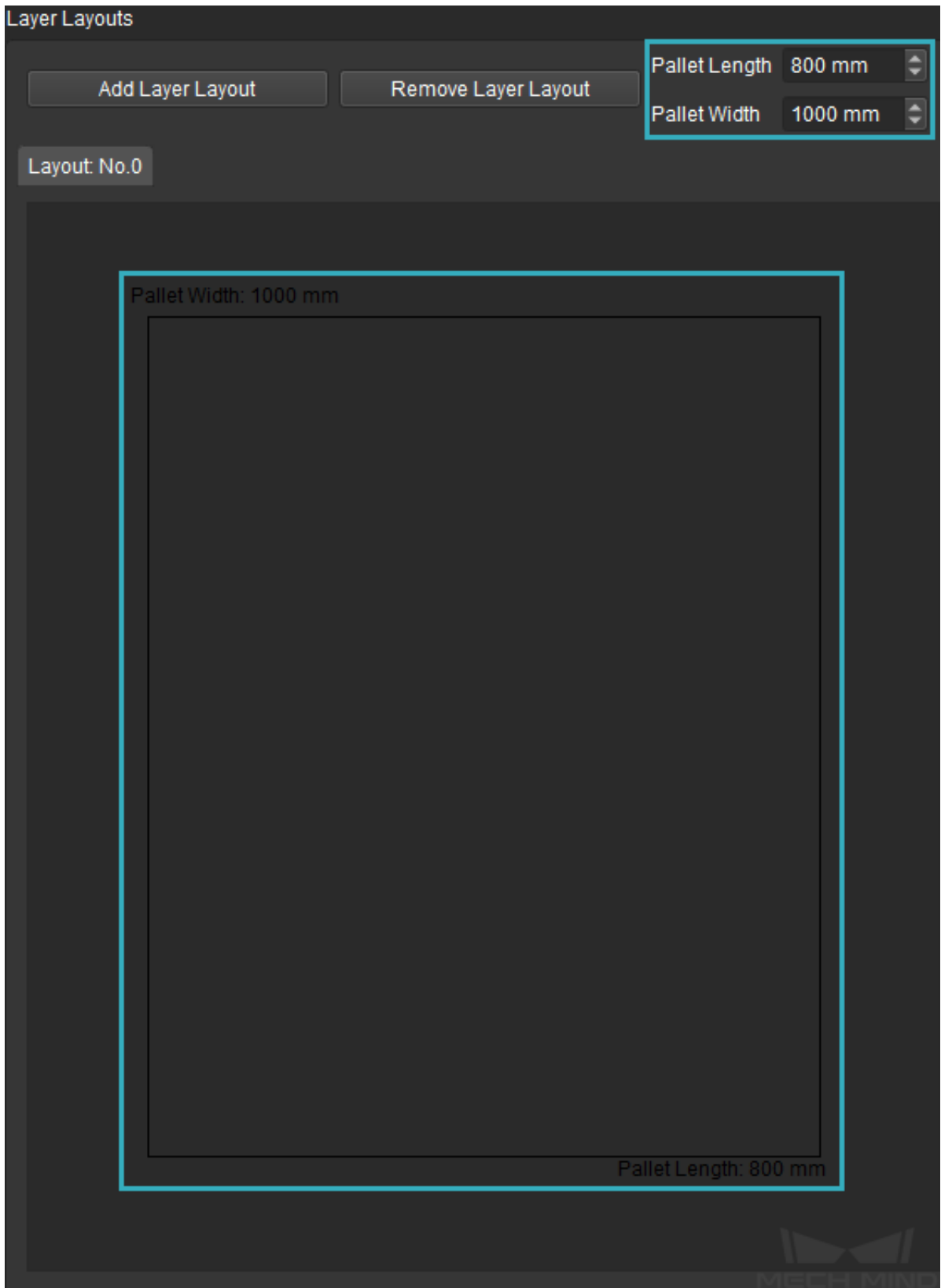
The main interface of pallet editor can be divided into Overall Editing Area (1) and Layout Editing Area (2).



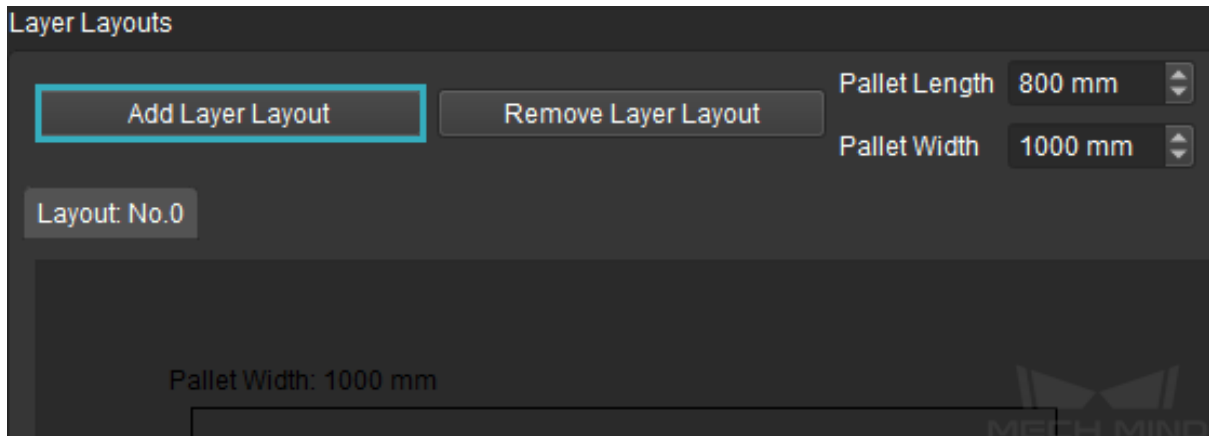
1. Overall Editing Area: Set the number of layer, layout of each layer, and the height of layer.
2. Layout Editing Area: Set the number, position, and size of the cartons to edit the layout.

12.4.2 Edit Pallet

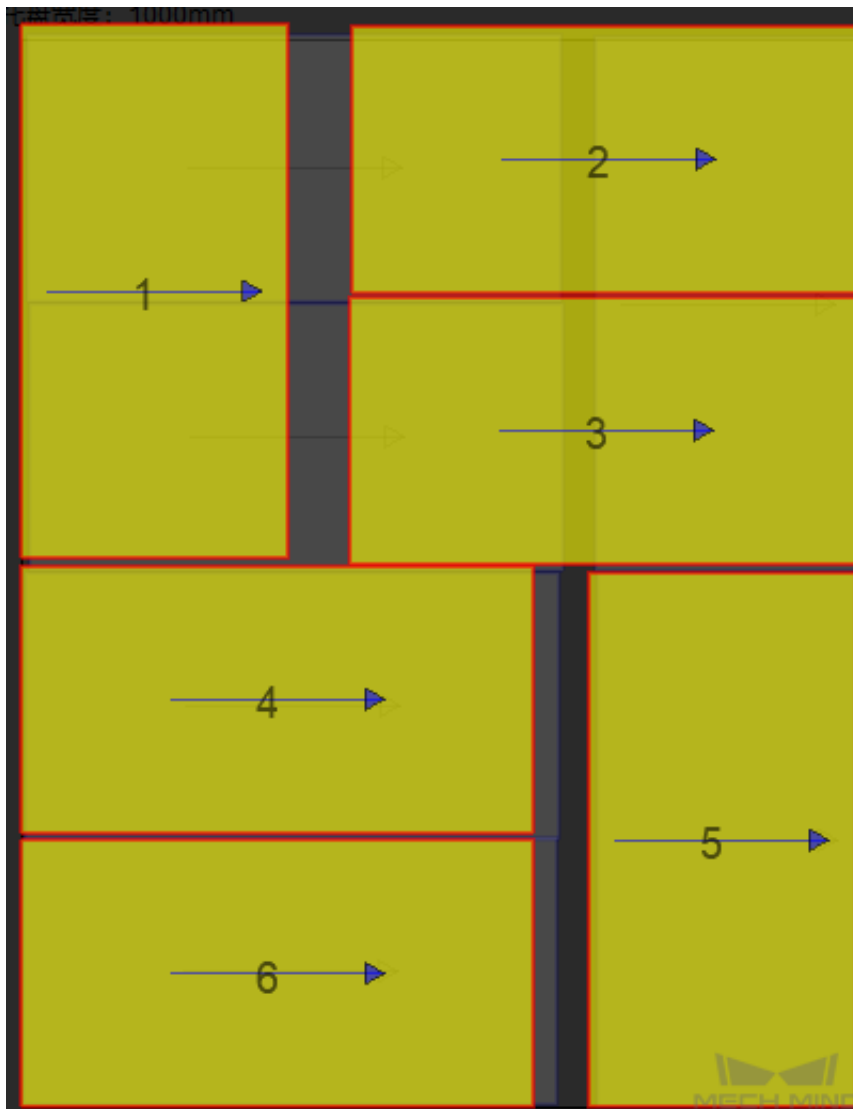
1. Set the pallet length and pallet width, which are the length and width of the pallet tray.



2. Click on *Add Layer Layout* in the **Layout Editing Area** to add a new layer layout.



3. Select a layout and click on *Add Position* to add cartons. You can drag the carton to adjust the positions. Select the individual carton and set the length, width, rotation, and entry angle in the **Position Setting** panel. If you want to delete a carton, select it and click on *Delete Position*.



4. Click on *Add Layer* in the Overall Editing Area to add new layers.

Pallet Layers

Add Layer

Remove Layer

Change Layout

Layer Height

No.	Layout	Height	Pos
4	UNDEFINED	100 mm	
3	UNDEFINED	100 mm	
2	UNDEFINED	100 mm	
1	UNDEFINED	100 mm	

Select the layer, select a customized layer layout in the drop-down list of **Change Layout**, and enter the **Layer Height**. The method to configure each layer is the same.

Change Layout 2

Layer Height 3

No.	Layout	Height	Pos Num
4	No.3	100 mm	6
3	No.2	100 mm	6
2	No.3	100 mm	6
1	No.2	100 mm	6

1

Hint: Pos Num is equal to the number of cartons in each layer and will be updated automatically.

5. After configuration, go to *File* → *Save to File* and specify a directory to save the configuration file in JSON format. Go to *File* → *Load From File*, and you can open the existing pallet configuration file in the pallet editor.
-

Appendix

MECH-VIZ APPENDIX

Here you can find detailed introduction of the Mech-Viz's interface and additional information for advanced features.

13.1 Fundamentals

13.1.1 Robot

Unless otherwise specified, robot in this article refers to a system of rigid bodies connected by joints, as shown in *Figure 1*.



Figure 1. Example of robot

13.1.2 TCP (Tool Center Point)

Robots are usually equipped with end effectors, which are used to interact with objects in the surrounding world. The tool center point (TCP) is the tip point of the end effector. In order to complete tasks such as picking, we usually say that the robot should move to a specific point in space, which actually means its TCP should move to that point.

13.1.3 Joint Position (Jps)

Joint position, commonly known as joint angle, is the position of each joint of the robot. Joint positions are a set of parameters, and the number of sets of parameters equals the number of joint axes. For example, the set of joint positions of a six-axis robot is 6.

13.1.4 Pose

The position and orientation of objects in space are collectively called poses. The pose is expressed in translation and rotation (quaternion or Euler angle). In Mech-Viz, poses are divided into tool pose and object pose, as shown in *Figure 2*.

TCP

The position of the tool center point relative to the robot base.

Object Pose (Obj Pose)

The object pose is the pose of the object's center relative to the base of the robot. When an object is attached to the end of the robot, the default initial position of the object is the same as the tool pose, but the orientation is opposite. If there is *Symmetry* or *Pick Tool Offset*, the relative transformation of the object pose and tool pose may change.

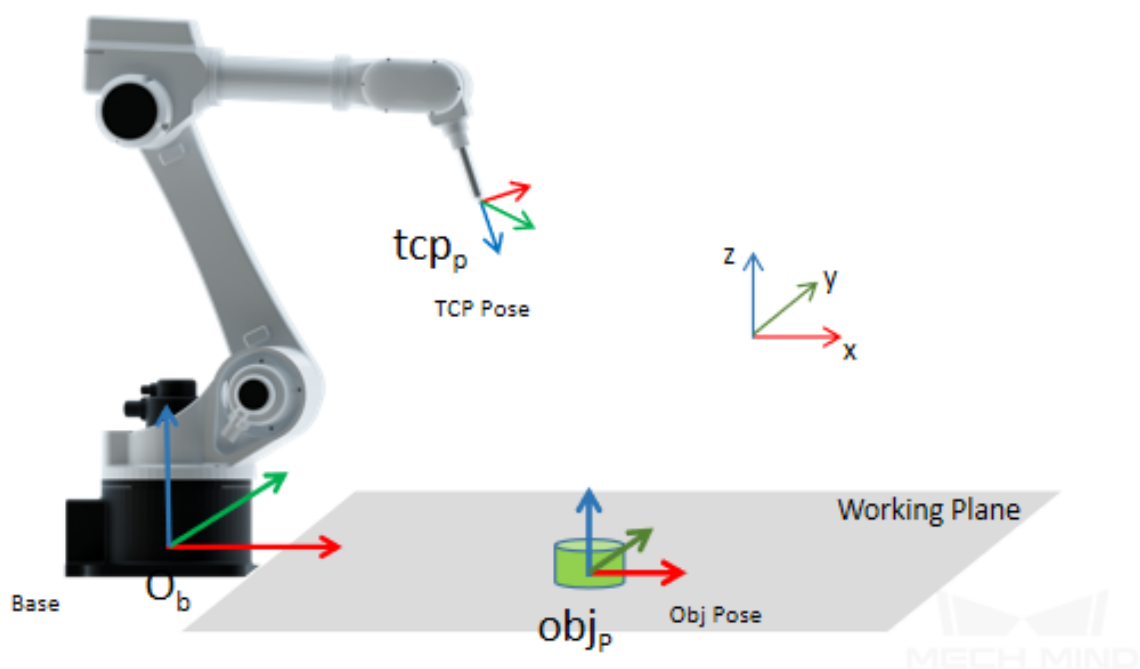


Figure 2. Poses

Attention: The direction of the Z-axis of the object pose must be the opposite of the direction of the Z-axis of the TCP during the grasping process.

Pick Point

Pick point is the position on the object that can be grasped by the robot.

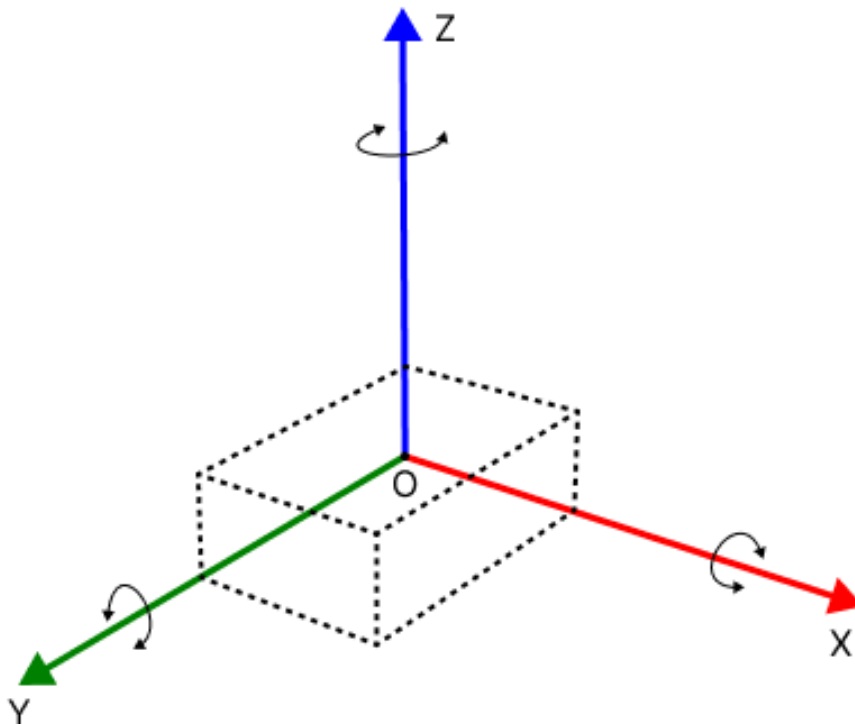
Picking Pose

Picking Pose is the pose of the robot when it picks the object.

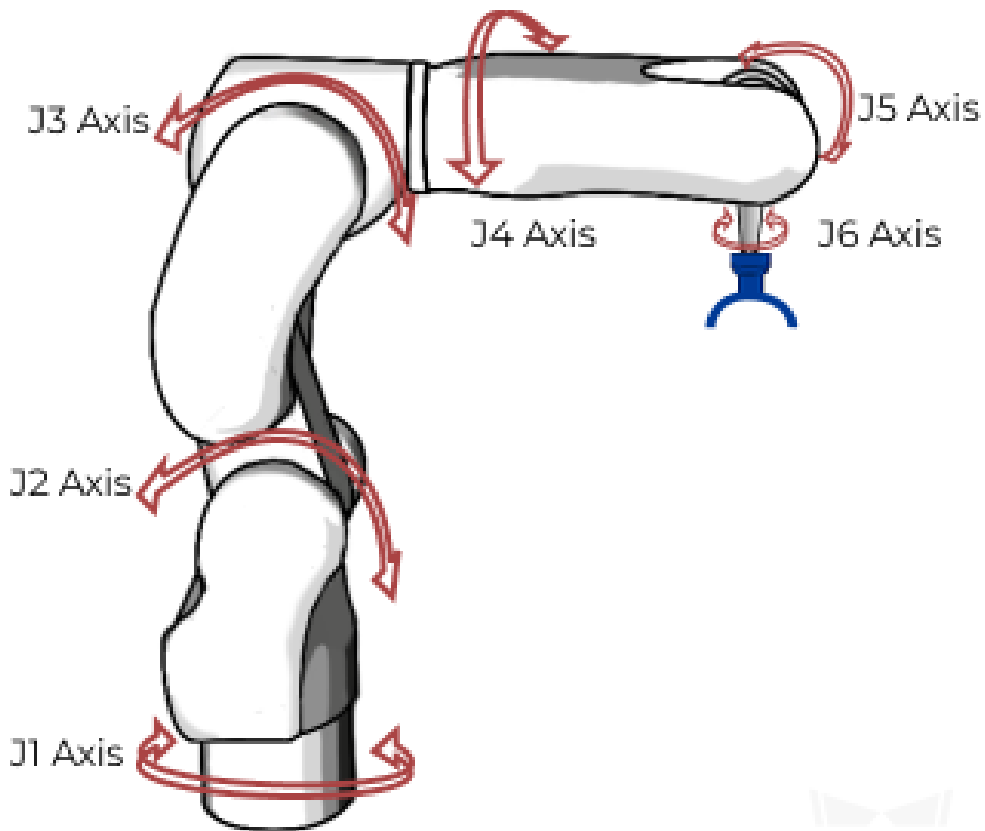
13.1.5 Degrees of Freedom

Degrees of freedom describe the level of freedom of movement. A Degree of Freedom (DoF) is an independent joint that can provide freedom of movement for the robot, either in a rotational or translational (linear) sense.

An object that is not constrained has 6 degrees of freedom in 3D space. As shown in the figure below, the cuboid in the 3D coordinate system has 3 degrees of freedom (translated along the X, Y, and Z axes) as to translation and 3 degrees of freedom (rotations about the X, Y, and Z axes) as to rotation.



The degrees of freedom of a common 6-axis robot is 6, as shown below.



13.1.6 Forward Kinematics

Forward Kinematics computes the pose of the end effector from given joint angles. When a set of robot joint angles is given, there is only one solution of the tool pose.

13.1.7 Inverse Kinematics

Inverse Kinematics computes the joint angles from a given end effector pose. For a specific tool pose, the corresponding joint angles may have multiple sets of solutions or no solutions.

13.1.8 Pick-Hold-Place

Picking, holding and placing describe the way that robot interacts with objects. These actions are an integral part of applications of symmetry in *Basic Move* and transformation changes/resets between object poses and tool poses.

Picking

The state that robot moves to the pick point pose and controls the end effector to pick the object by changing the output signal is called “picking”. Once the “picking” state becomes effective, the relative position of the object and the robot’s TCP is bonded, and Mech-Viz will plan the robot’s trajectory according to the object’s *Symmetry*.

Holding

The state after the robot picking up the object and before the object being placed is called “holding”. In the “holding” state, the bound relationship between the object and the robot always exists. This state will take effect automatically after the object is picked, and cannot be configured in Mech-Viz.

Placing

The state that robot reaches the pose of placing and controls its end effector to release object by changing the output signal is called “placing”. When the “placing” state becomes effective, the relative position of the object and the robot’s TCP is no longer bonded.

13.1.9 Symmetry

Each object to be picked has a corresponding object pose. According to the object pose, Mech-Viz will change the tool pose to control the robot to pick, as described in *Pose*. In practice, objects often have symmetry. For these objects, the robot can pick or place in various ways according to its **symmetry angle**, and the results are the same.

Symmetry angle is the smallest angle for which the object can be rotated around the X/Y/Z axis to coincide with itself. For example: the symmetry angle of a square is 90°; of a rectangle is 180°; of a regular hexagon is 60°; of a circle or a ring is 0°; no rotation symmetry is 360°, as shown in *Figure 3*.

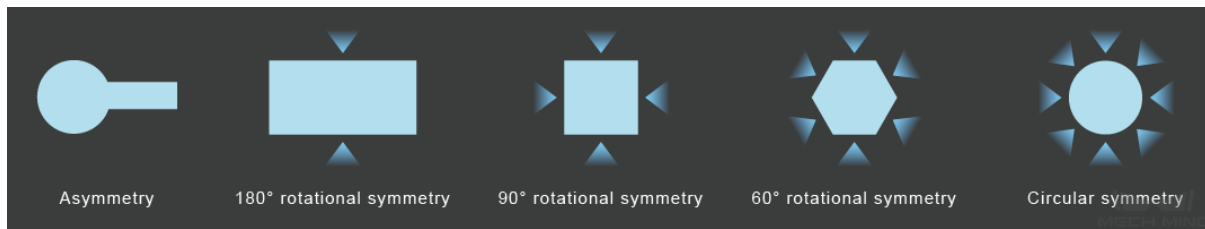


Figure 3. Symmetry angle of objects

Among them, the symmetry around the Z axis is called the strong axis symmetry, and the symmetry around the X or Y axis is called the **weak axis symmetry**.

weak axis symmetry: Usually, between X and Y axis, the axis with a relative stronger symmetry will be regarded as the weak axis and its symmetry will be used in the actual calculation. For example, if an object has 90° symmetry angle around X axis and 0° symmetry angle around Y axis, the Y axis will be regarded as the weak axis, and therefore the symmetry angle of weak axis is 0° .

Users can set the symmetry of the object and use Mech-Viz to select an optimal trajectory of picking and placing to improve accessibility and reduce the rotation of the end effector.

Symmetry of Object

Object's symmetry is a necessity for the adjustment of object's pose. For example, when the symmetry of the object is 180° , the robot can freely pick or place in either "forward" or "backward" direction to reduce the rotation of the end effector, as shown in *Figure 4*.

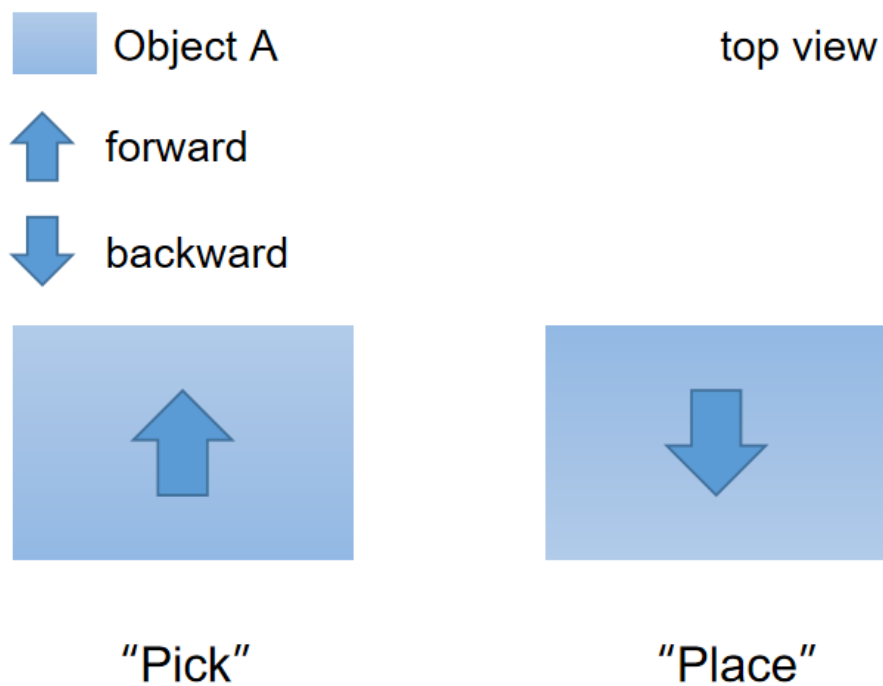


Figure 4. Object Symmetry

The two states of the object can be regarded as the same, and robot can pick in the forward direction and place in the backward direction.

Symmetry of Pick Point

The symmetry of the pick point is related to the way that the end effector picks the object and is used to select an optimal pose of picking. For example, when the object is smaller than the size of a suction cup, the object is picked by the vacuum gripper with offset in order to prevent collision with objects nearby during the picking process. In this case, the object symmetry does not take effect, and the pick point has a symmetrical angle of 180°. You can rotate the pick tool at the initial pick point to "forward pick" or "backward pick", as shown in [Figure 5](#) below:

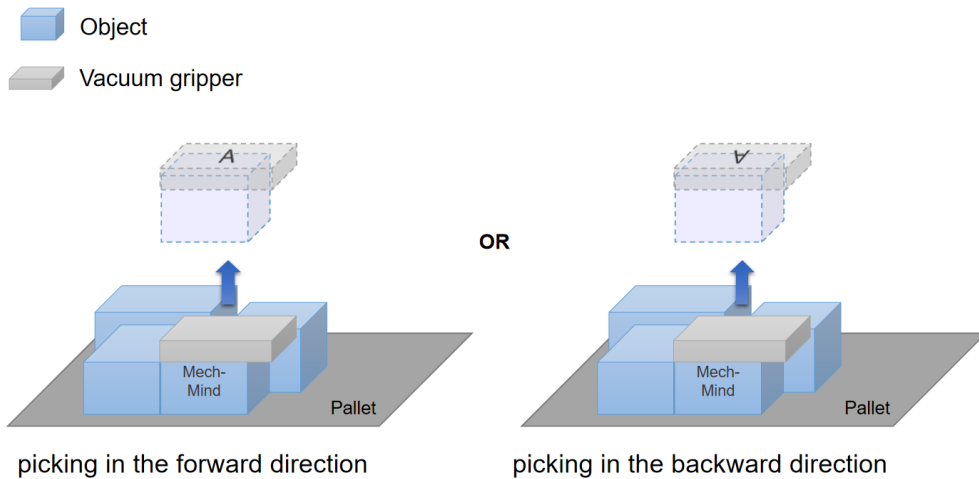
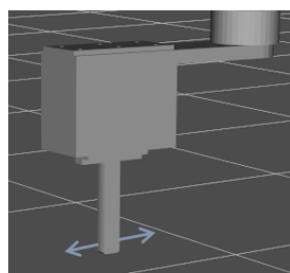


Figure 5. Forward and backward pick

In order to plan the optimal trajectory of movement, Mech-Viz will consider both the symmetry of the pick point and the object during the picking process; and the object pose will be selected according to the symmetry of the object during "holding" and "placing". For example: when the object is a connecting rod, you can use an adaptive gripper to pick the ring from inside out, as shown in *Figure 6* below:



Adaptive gripper

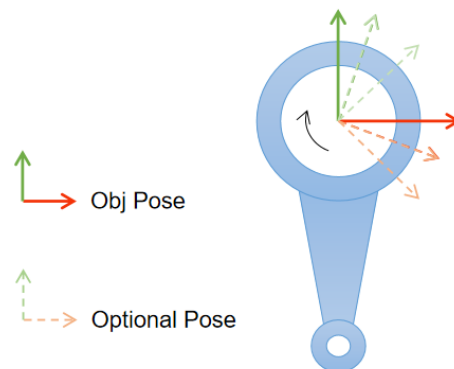


Figure 6. Pick a ring from inside out

The symmetry of the pick point here is 0° , and any angle is optional during picking. Therefore, the symme-

try of the pick point should be set with a reasonable trial step size (such as 1°, 10°, etc.). However, since there is no object's symmetry, if you want to place the object in a set position, Mech-Viz will restore the relative transformation between the end effector and the object during picking.

13.1.10 Pick Tool Offset

When the visual result guides the robot to pick objects of small size or more complex shapes, in order to avoid collision with other objects, the TCP can be offset to a certain point of the object to pick. After assigning the type of tool and strategy of picking, Mech-Viz calculates the value of bias and determines with the point cloud collision to check whether it can be picked or not. As shown in *Figure 7*.

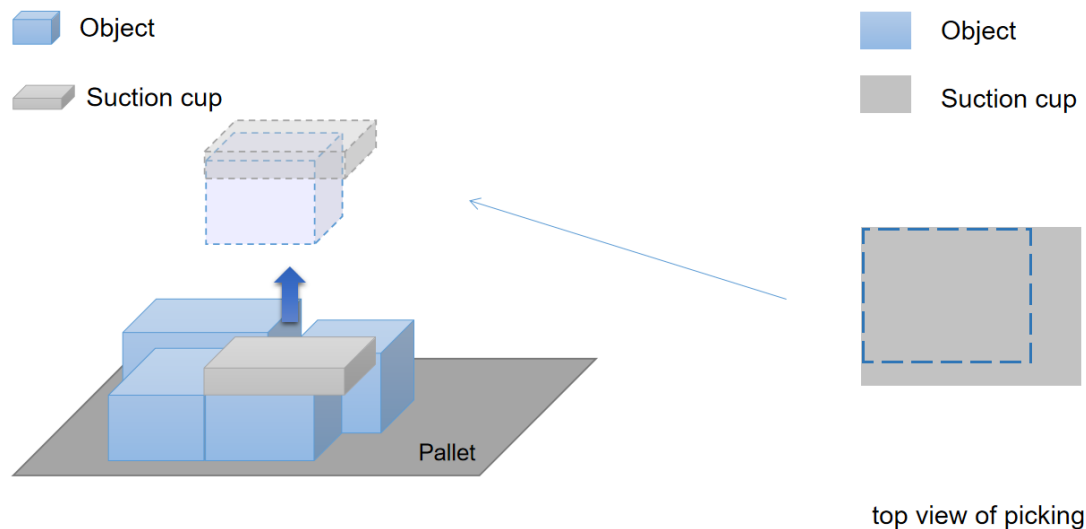


Figure 7. Pick tool offset

Similarly, in mix palletizing tasks, robot can avoid collision with objects nearby with the usage of pick tool offset .

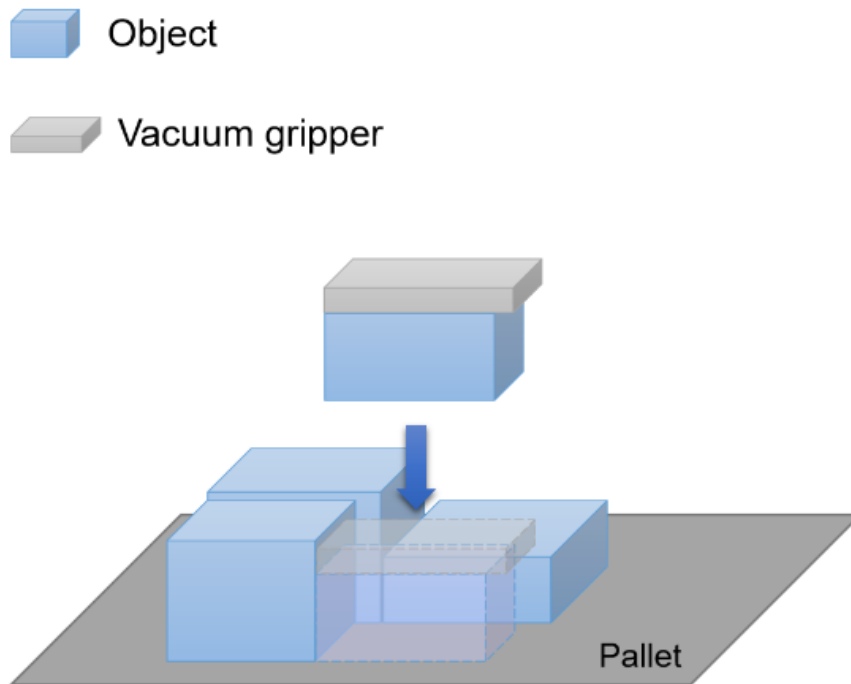


Figure 8. Pick tool offset to avoid collision

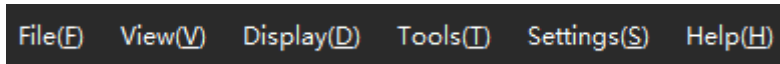
Note: For more detail about the using of bias, please see [visual_move](#).

13.1.11 Collision Model Type

There are 4 main types of collision models in Mech-Viz:

1. Basic geometry (only include cuboid at present): This type of model mainly includes cuboid and end effector which is only loaded with a 3D model but without a collision model in the scene.
2. Mesh (only include the surface information): This type of model only enables collision detection of the surface. The main models are robot joints and end effector loaded with STL collision model.
3. Convex polyhedron assembly: This type of model is used as a solid for collision detection, and the main model is an end effector loaded with OBJ collision model.
4. Octree: The main models are point cloud and the end effector loaded with binvox collision model.

13.2 Mech-Viz Menu Bar



Below you can find the detailed description of every option in Mech-Viz’s menu bar.

13.2.1 File

Option	Description	Shortcut
Open Project	Select a project folder to open in Mech-Viz	Ctrl + O
Save Project	Save the changes to the current project	Ctrl + S
Save Project to JSON	Save the project, and save the VIZ file in the project folder as JSON	Ctrl + Shift + S
Save Project as	Save the project to the designated location	None
	Only the VIZ file and collision models are saved	None
Recent Projects	Display the recently opened projects, click on a project path to open the project	None
Open Project in Explorer	Open the current project’s folder in File Explorer	None
Open Executable File in Explorer	Open the folder where the Mech-Viz software is located	None
Backup Project	Create a backup of the current project	None
	All contents in the current project folder are copied and pasted	None
Close Project	Close the current project	None
Exit	Exit Mech-Viz	Ctrl + Q

Note:

- Mech-Viz 1.4.0 and above saves the project file in VIZ format. Previous versions save the project file in JSON format.
- If you want to use Mech-Viz 1.6.1 or above versions to open a previous version’s project, please read [Open Projects of Mech-Viz 1.6.0 or Lower](#) first.
- Please wait for the project to complete loading before running the project.

If you do not select any options in the **Version compatibility** pop-up window when you open Mech-Viz, and click *Run* in Meh-Center directly, a message saying that **“Project loading by Mech-Viz in progress. Failed to start the Mech-Viz project: XXX”** will appear in the log panel in Mech-Center. In this case, please select Yes in the **Version compatibility** window and then click *Run* in Mech-Center to run the project again.

13.2.2 View

Used to configure the display of Mech-Viz.

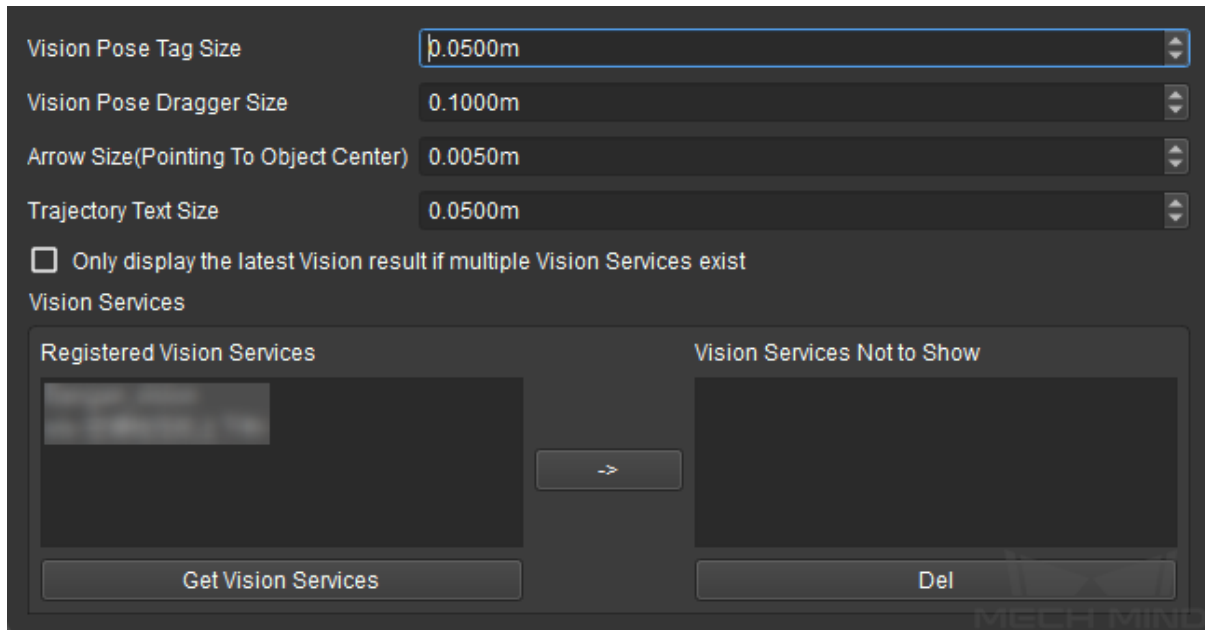
Option	Description
Scene	Display the Scene tab, checked by default
Workflow	Display the Workflow tab, checked by default
Robot	Display the Robot tab, checked by default
Tools and Workobjects	Display the Tools and Workobjects tab, checked by default
Collisions	Display the Collisions tab, checked by default
Plan History	Display the Plan History tab, checked by default
Others	Display the Others tab, checked by default
Log	Display the Log tab, checked by default
Function Description	Display function description of selected Task in the Workflow tab
Default Layout	Reset the layout of Mech-Viz to default

13.2.3 Display

Used to configure the display of the 3D simulation area.

Option	Description
Show Received Vision Poses	Display the vision poses, indices, and labels received from Mech-Vision, checked by default
Show Point Cloud	Display the point cloud received from Mech-Vision, checked by default
Show Picked Object	Display the model of the picked workobject, checked by default Model is displayed if either of the following is satisfied: 1. Vision poses and object dimensions are received from Mech-Vision 2. Non-cuboid object model is present in the project folder
Show Collision While Planning	Display collisions detected during path planning, checked by default
Show Octree	Display the octree structure of the object point cloud Detect collision between point cloud and others in <i>Collisions</i> → <i>Collision Detection Configuration</i> must be enabled
Show Received Carton Models	Display models of cartons The project must receive vision poses and object dimensions from Mech-Vision to generate carton models
Show Arrow Pointing to Center	Display the arrows pointing from pick points to the geocenter Multiple pick points must be set on the point cloud model in Mech-Vision
Show Object Pose	Display the pick points on workobjects
Display Settings	Provide more display settings, please see below for detailed information
Pose Status Colors	Color code for vision pose indices and labels

Display Settings



Option	Description
Vision Pose Tag Size	Set the size of the vision pose indices and labels
Vision Pose Dragger Size	Set the size of the vision pose arrows
Arrow Size(Point To Object Center)	Set the size of the arrow that points to geocenter
Trajectory Text Size	Set the text size of task names on the planned path
Only Display the latest Vision result if multiple Vision Services exist	When checked, only the last received vision result is displayed in the 3D simulation area
Registered Vision Services	List of Mech-Vision projects registered in Mech-Center
->	Add the selected Mech-Vision project to Vision Services Not to Show
Vision Services Not to Show	List of Mech-Vision projects whose vision results are not displayed in the 3D simulation area
<i>Get Vision Services</i>	Refresh the list of Registered Vision Services
<i>Del</i>	Delete the selected Mech-Vision project from Vision Services Not to Show

13.2.4 Tools

Option	Description
Use Saved Vision Records (Only When Vision Service Not Registered)	When checked, Mech-Viz project will use saved vision records to run
Set Vision Records	Configure the vision records to be saved and/or loaded For detailed instructions on using vision records, please see Save and Load Vision Records
Find Vision Result	Highlight the vision point with the searched index in the 3D simulation area
Not Print Message Sent to Robot	When checked, messages sent to the robot by Mech-Viz are not displayed in the log
Write Debug File (.dmp)	Generate DMP file to be sent to Mech-Mind Technical Support for debugging

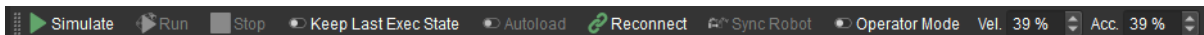
13.2.5 Settings

Option	Description
Set Mech-Center Address	Set the IP address of Mech-Center Usually, the IP address is automatically detected and set, and you don't need to do anything
Lock/Unlock Project	When checked, all modification to the project are prevented (you can still simulate and run the project) The password must be 6 digits or more and made up of numbers only
Options	Set display language, units of length and angle, etc.
Logging Level	Set the level of log messages to be displayed in the Log panel All messages of the selected level and above are displayed

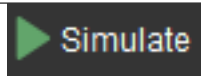
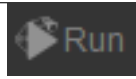
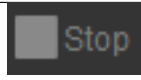
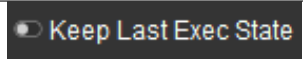
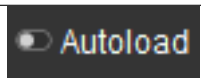
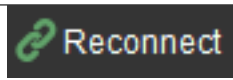
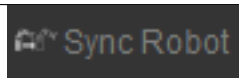
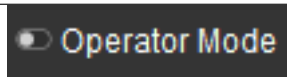
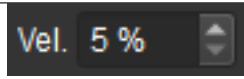
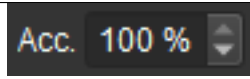
13.2.6 Help

Option	Description
About	Display software version and copyright information
Change Log	Open the software changelog in your browser

13.3 Mech-Viz Toolbar



Below you can find the detailed description of every option in Mech-Viz's toolbar.

Option	Description
	Simulate the project (only move the simulated robot)
	Run the project (move both the simulated and real robot)
	Stop the execution of the project
	Keep the last execution state Mainly used in palletization applications, allow resumption of palletization from where left off
	When enabled, the current project is autoloaded when Mech-Viz is opened
	Reconnect to the real robot
	Sync the pose of the real robot to the simulated robot
	When enabled, the processing speed of Mech-Viz is improved
	Set robot velocity
	Set robot acceleration

13.4 Create and Import the Robot Model

The basic procedures for creating and importing a robot model are shown below:

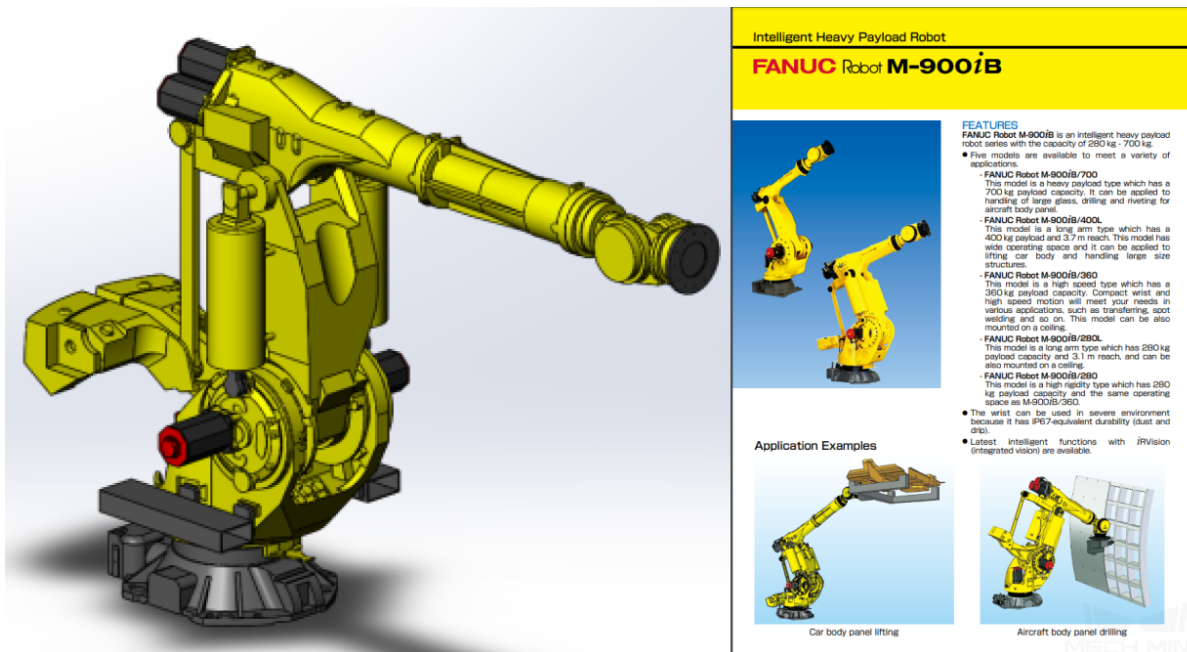
- *Obtain the CAD File and Specifications of the Robot*
- *Write the [robot]_algo.json File*
 - *[robot]_algo.json Template File*
 - *Obtain the Configuration of the Robot (algo_type)*
 - *Obtain the DH Parameters (dh, dhPassive)*
 - *Obtain the Motion Ranges of Each Joint (minlimits, maxlimits)*
 - *Obtain Other Parameters*
- *Write the [robot]_profile.json Parameter File*
- *Use Solidworks to Construct a STL Model of the Entire Robot*

- Import the CAD Model in SolidWorks
- Create Reference Frames
- Export STL Models of Robot Joints
- Import the Model File in Mech-Viz

This section describes how to create a robot model and import it in Mech-Viz V1.6.0. A FANUC M-900iB/400L robot will be used as an example.

13.4.1 Obtain the CAD File and Specifications of the Robot

Before creating the robot model, please obtain the CAD file and the datasheet which contains DH parameters and motion ranges of each joint of the robot. You can download the relevant documentation on the official website of the robot. The figure below shows the CAD file and descriptions of FANUC M-900iB/400L.



Hint: Some robot websites will provide models in X_T format. Comparing to STEP files, X_T files perform better and takes less time in reconstruction. Therefore, it is recommended to use models in X_T format.

13.4.2 Write the [robot]_algo.json File

[robot]_algo.json Template File

The template file of [robot]_algo.json is stored in `resource/robot/algo_example.json` in the installation directory of Mech-Viz. You can create a new [robot]_algo.json file based on the template.

Hint: Please refer to [Description of \[robot\]_algo.json file](#) for descriptions of parameters in [robot]_algo.json.

Obtain the Configuration of the Robot (algo_type)

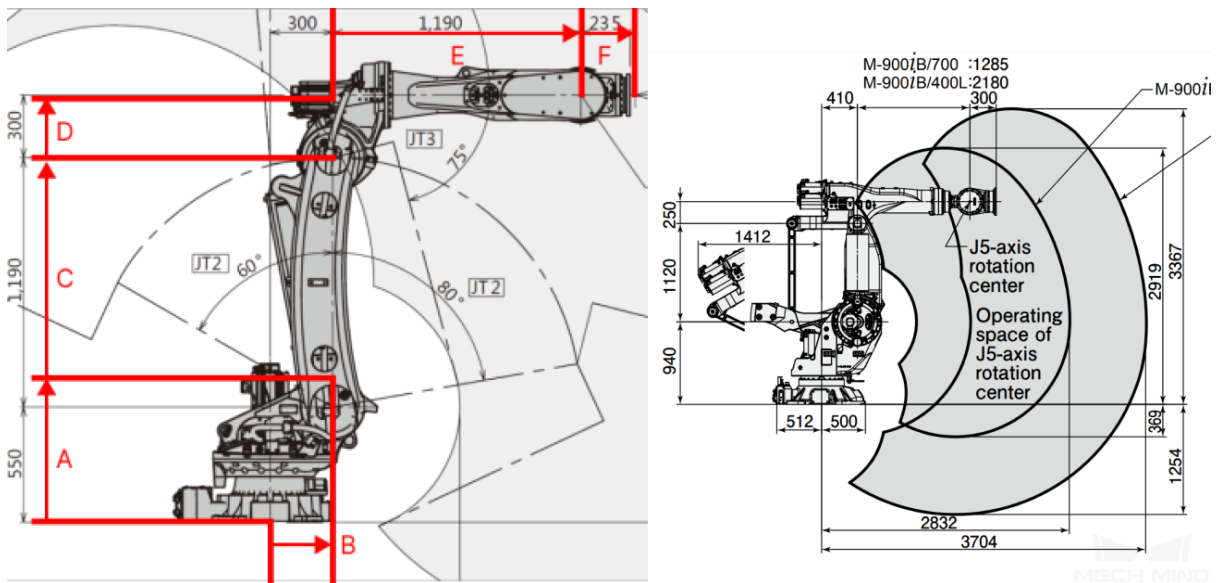
Different robots have different configurations. In Mech-Viz, different robot configurations are described by different frames of reference and DH parameter definitions. Please refer to [Robot Configurations](#) for examples of different robot configurations.

Unlike UR collaborative robot, FANUC M-900iB/400L is a common 6-axis split-ball type wrist industrial robot, which belongs to the SphericalWrist_SixAxis configuration.

Obtain the DH Parameters (dh, dhPassive)

Please refer to [Robot Configurations](#) and find the corresponding configuration file of the robot. Then you can confirm the dh value in [robot]_algo.json according to the DH parameters in the specifications of the robot.

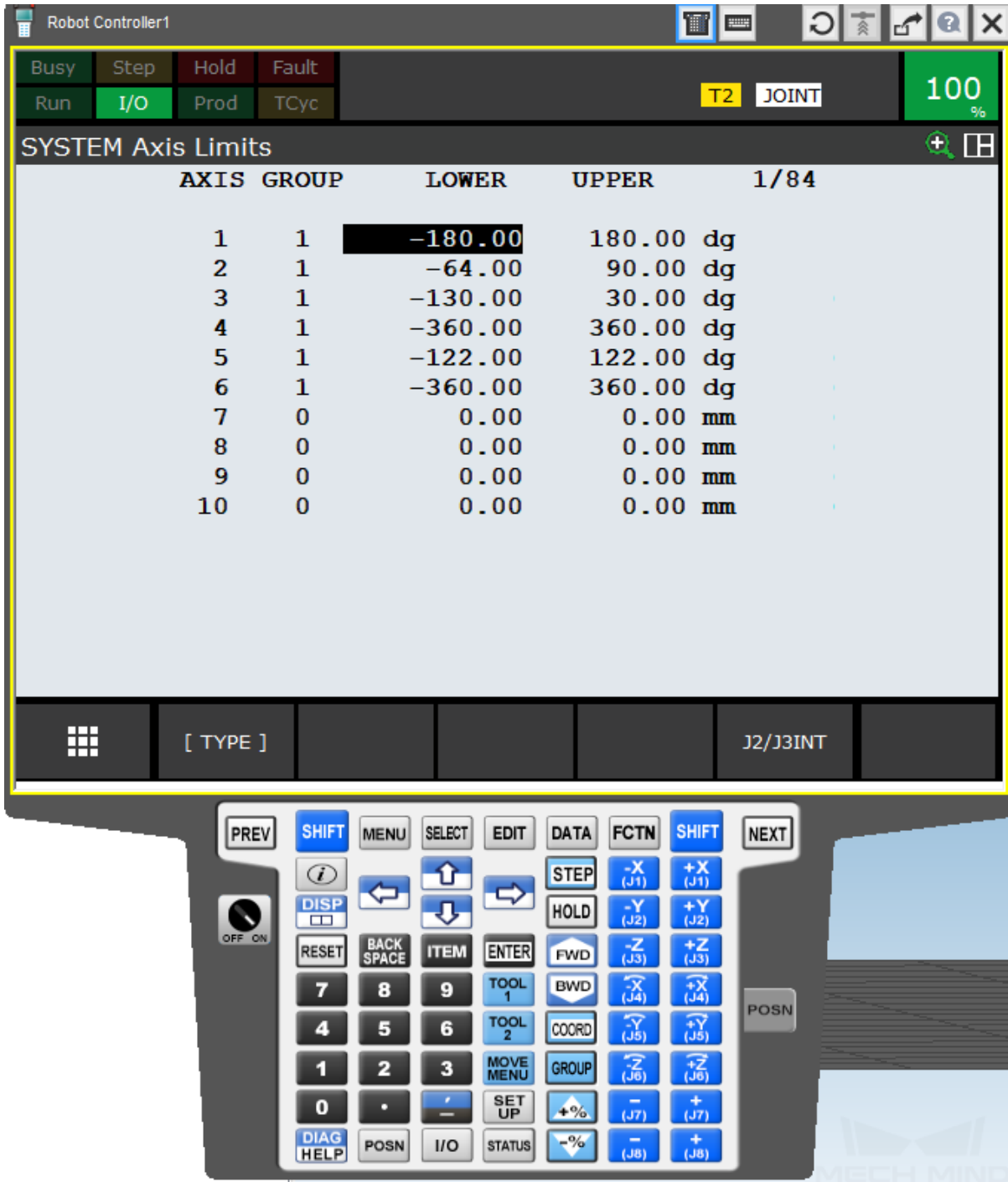
The dh parameters of FANUC M-900iB/400L are $a=0.940$, $b=0.410$, $c=1.120$, $d=0.250$, $e=2.180$, $f=0.300$.



In the above figure, the configuration diagram of SphericalWrist_SixAxis robot is on the left, and the specifications of FANUC M-900iB/400L is on the right.

Obtain the Motion Ranges of Each Joint (minlimits, maxlimits)

Usually, the motion ranges of joints are specified in the documentation of the robot. However, you need to open the robot simulation software RoboGuide of FANUC robots and then search for the motion ranges in this case.



Robot Controller1

Busy Step Hold Fault
Run I/O Prod TCyc

T2 JOINT 100%

SYSTEM Axis Limits

AXIS	GROUP	LOWER	UPPER	1/84
1	1	-180.00	180.00	dg
2	1	-64.00	90.00	dg
3	1	-130.00	30.00	dg
4	1	-360.00	360.00	dg
5	1	-122.00	122.00	dg
6	1	-360.00	360.00	dg
7	0	0.00	0.00	mm
8	0	0.00	0.00	mm
9	0	0.00	0.00	mm
10	0	0.00	0.00	mm

[TYPE] J2/J3INT

PREV SHIFT MENU SELECT EDIT DATA FCTN SHIFT NEXT

DISP ← ↑ → STEP -X (J1) +X (J1)

RESET BACK SPACE ITEM ENTER FWD -Y (J2) +Y (J2)

7 8 9 TOOL 1 BWD -Z (J3) +Z (J3)

4 5 6 TOOL 2 COORD -X (J4) +X (J4)

1 2 3 MOVE MENU GROUP -Y (J5) +Y (J5)

0 . - SET UP -Z (J6) +Z (J6)

DIAG HELP POSN I/O STATUS +% - (J7) + (J7)

-% - (J8) + (J8)

Obtain Other Parameters

For common robot brands, you can refer to the parameters of other created robot models. If there is not any available parameters to reference, please contact the site engineers for other parameters.

Attention:

- Please use English punctuation marks in the JSON parameter file.
- The `mastering_joints`, `axis_flip`, `base_z_offset` attributes should correspond to those specified in the **robot simulation software**.
- The `axis_flip` attribute is coupled with `minlimits` and `maxlimits`. Sometimes it is necessary to switch the value of the upper and lower limits and reverse the positive and negative signs of the values at the same time.
- `J6` in the `mastering_joints` attribute can be neglected easily. This attribute affects **Euler angles** and need to be checked and saved carefully.

13.4.3 Write the `[robot]_profile.json` Parameter File

`[robot]_profile.json` template file is stored in `resource/robot/profile_example.json` in the installation directory of Mech-Viz. You can create a new `[robot]_profile.json` file based on the template.

Hint:

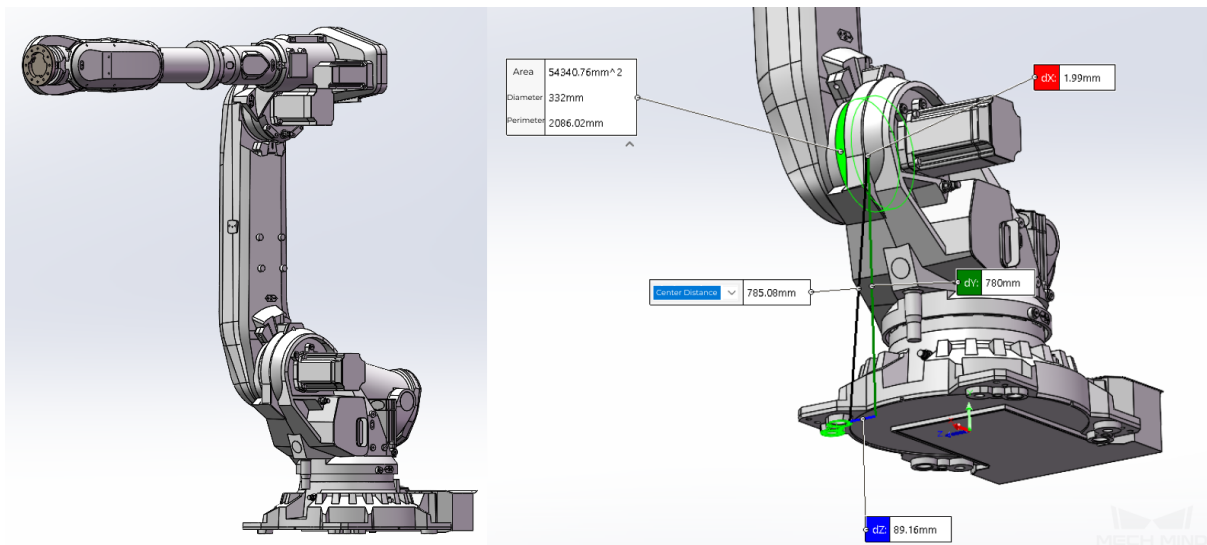
- Please refer to [Description of `\[robot\]_profile.json` file](#) for descriptions of parameters in `[robot]_profile.json`.
- This file is not mandatory.

13.4.4 Use Solidworks to Construct a STL Model of the Entire Robot

Once all the DH parameters are obtained, you can start to create reference frames for each axis on the robot model.

Import the CAD Model in SolidWorks

Open the CAD model of the robot in SolidWorks, as shown below.



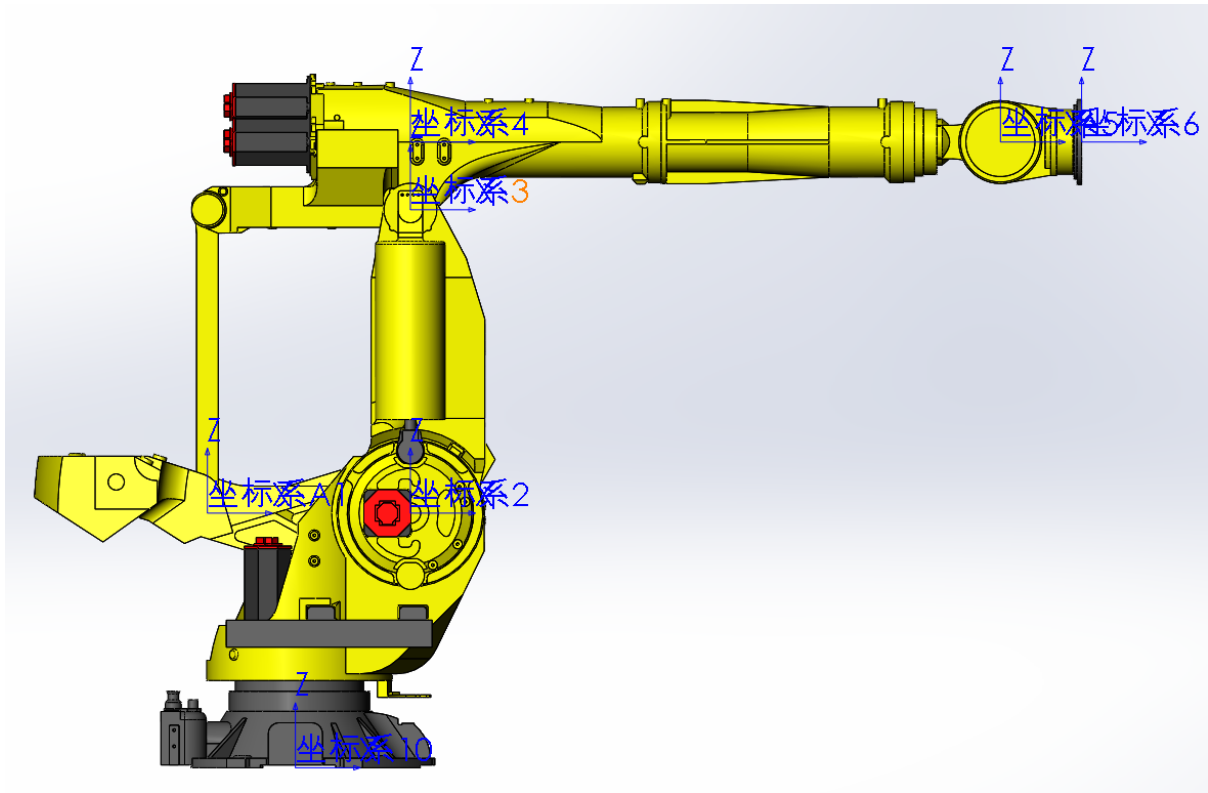
In the above figure, the CAD robot model is on the left and an assembly reference diagram is on the right.

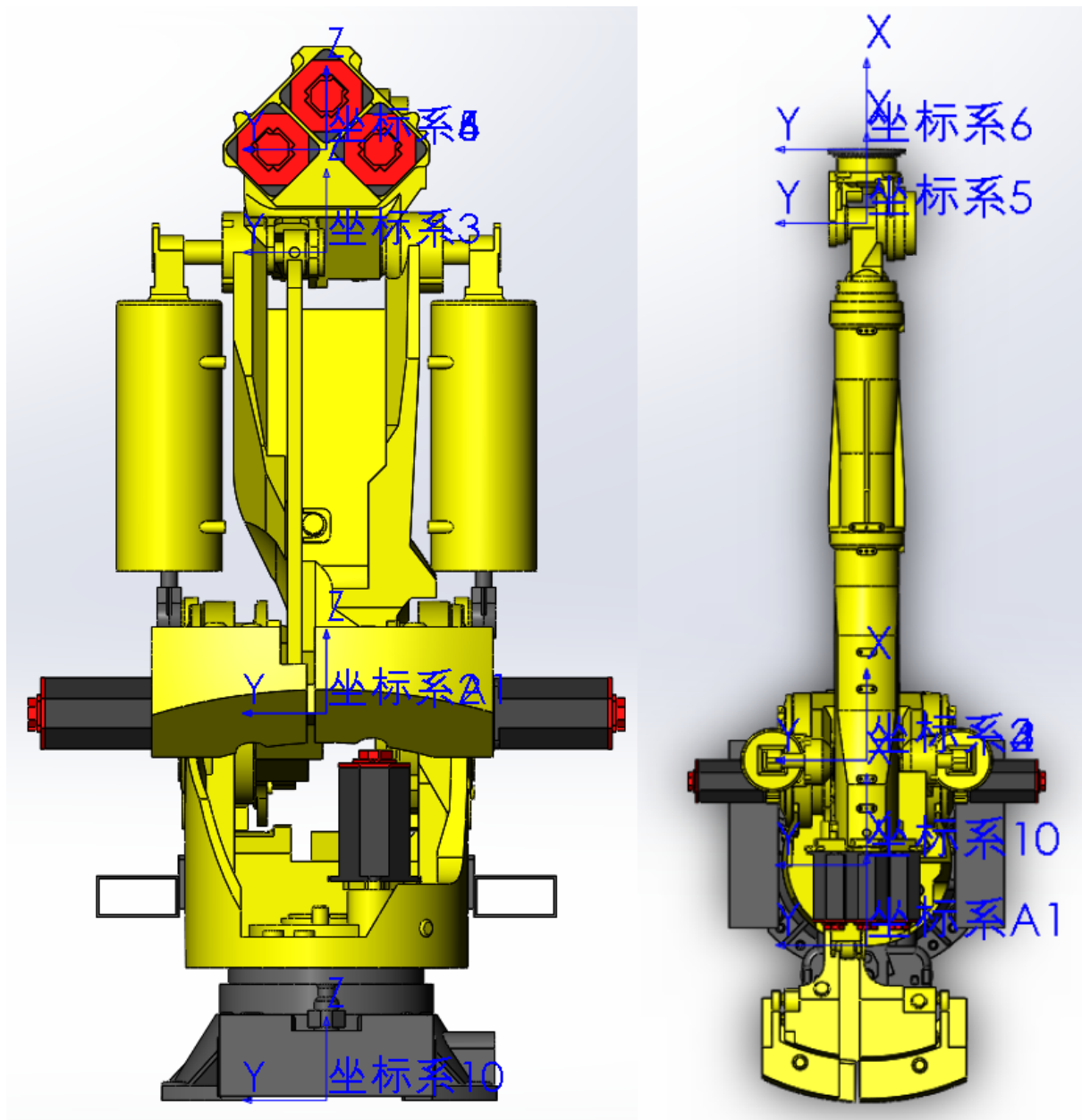
Hint:

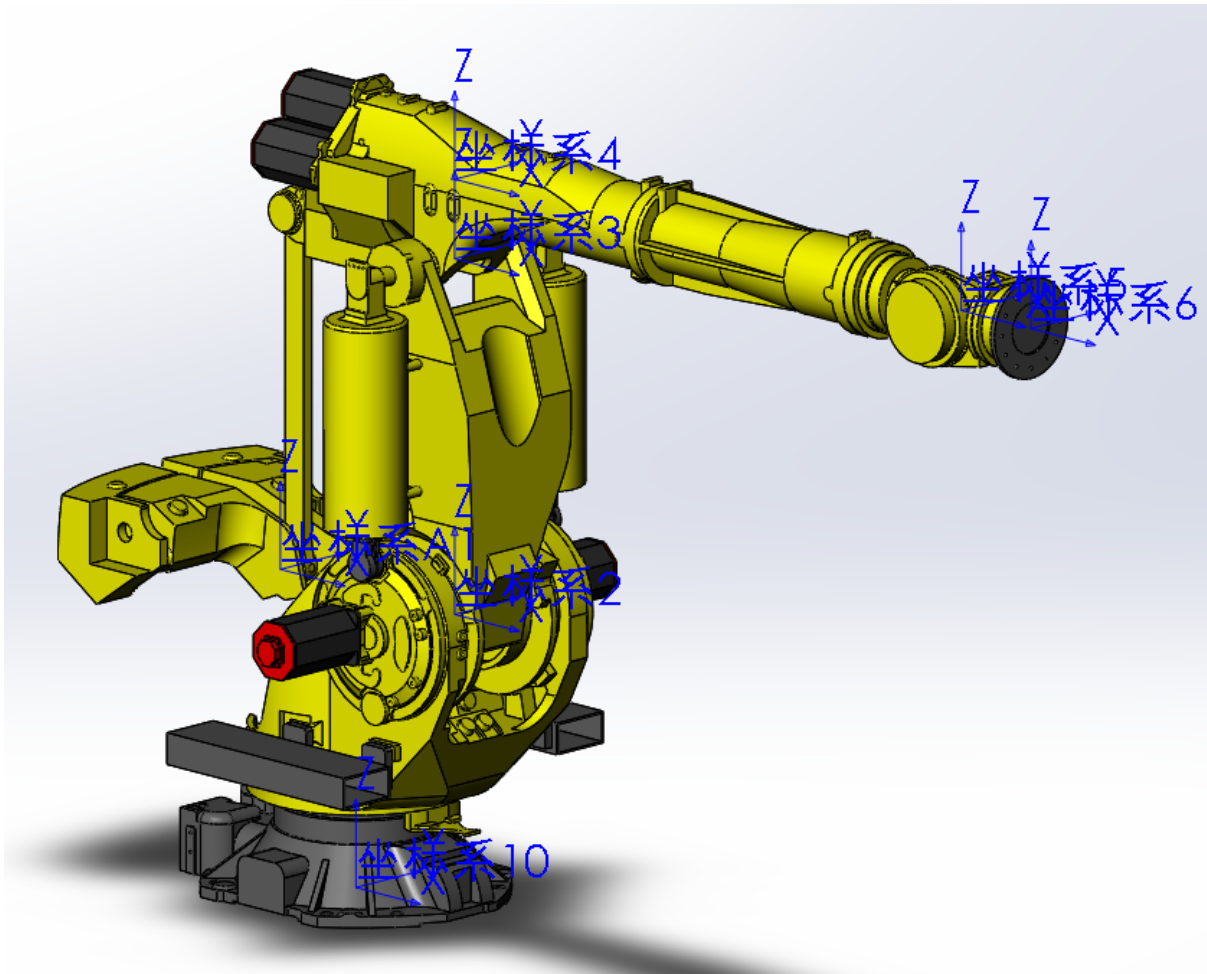
- It is recommended to download an entire robot model that has been assembled completely.
 - If you can download only the parts, you will need to assemble them yourself and check each joint according to the DH parameters.
 - You may remove the details of the model and keep only the structure that may affect collision detection to reduce the time to import the model.
-

Create Reference Frames

Please refer to [Robot Configurations](#) and create reference frames of each axis. As for FANUC M-900iB/400L, the configuration is SphericalWrist_SixAxis, and the robot pose should be as shown in the figure below: J1 axis is aligned with the base, J2 axis stays vertically upward, J3 axis and J4 axis stays horizontally forward, and J5 axis stays forward rather than downward. Once the pose is determined, all parts should be fixed.





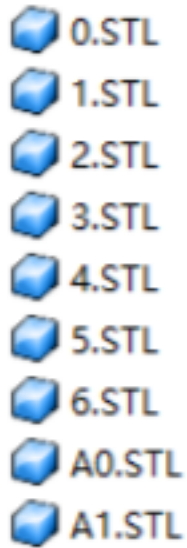


Export STL Models of Robot Joints

It is recommended to hide other parts on the assembly when exporting the joint model in turn.

Save the exported file in STL format, set the output to binary and the unit to meter.

The entire robot model of FANUC M-900iB/400L in STL format is shown below.

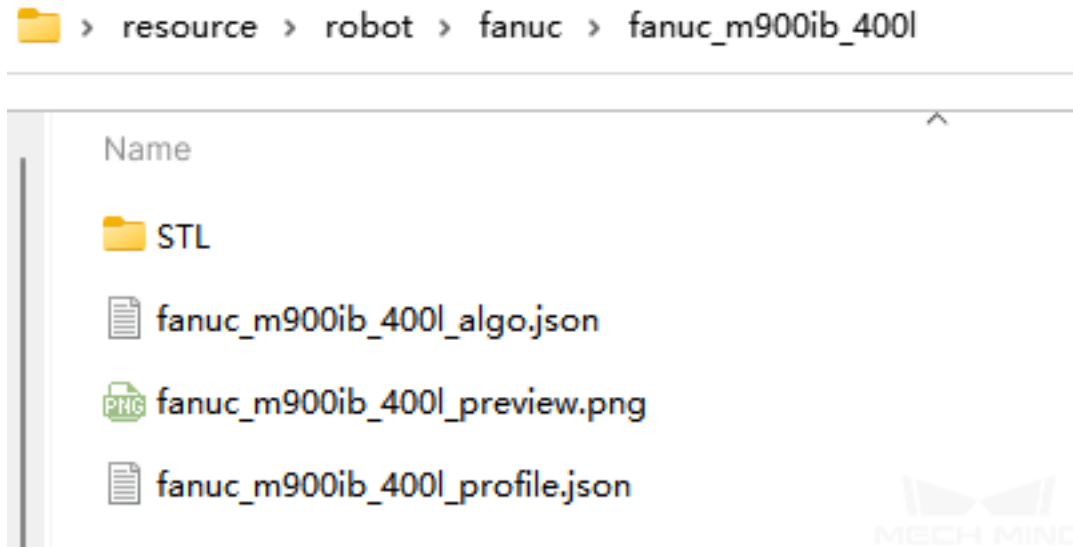


Now you have completed creating a robot model.

13.4.5 Import the Model File in Mech-Viz

The robot library of Mech-Viz is stored in `resource/robot` in the installation directory. Robot models of the same brand are stored in the same folder which is named after the brand name, and all the files of the robot model are stored in the folder which is named after the model name.

As for FANUC M900iB/400L, the model files are stored in `resource/robot/fanuc_m900ib_400l`. The descriptions of each file are as follows.



- STL: the robot model file
- fanuc_m900ib_400l_algo.json: robot kinematic parameter file
- fanuc_m900ib_400l_preview.png: image of the robot
- fanuc_m900ib_400l_profile.json: parameter description file of the robot

Attention:

- Use lower case letters to name the folders.
- The robot model file supports STL, DAE, and OBJ format. Different formats of model files are stored in a folder which is named after the format name respectively.
- STL model can be used for both display and collision detection.
- DAE model can only be used for display.
- OBJ model can only be used for collision detection.
- A complete robot model set should include at least one model used for display and one for collision detection.

Copy and paste all files into the corresponding directory and the robot model will be imported successfully. Then you can select the newly imported robot from the robot library in Mech-Viz.

机器人库

输入机器人型号

机器人品牌 可动轴数

机器人负载(kg) 0 800

工作范围(mm) 0 4100

当前机器人



FANUC_M900IB_400L
 额定负载: 400kg
 可动轴数: 6
 可达半径: 3704mm



UR_5
 额定负载: 5kg
 可动轴数: 6
 可达半径: 850mm

机器人

机器人库 设置软限位

真实机器人运动 移动真实机器人 直线运动 速度: 5%

关节角 工具位姿 机器人特性

	机器人特性	适用
1	收到停止信号后立即停止	✗
2	暂停	✗
3	收到DI信号前继续移动	✗
4	设置OO时机器人可以不停	✗
5	设置OO中可设置延时	✗
6	沿密集目标点移动	✗
7	设置TCP	✗
8	设置负载	✗
9	弧线运动	✗

显示Top拖拽器 旋转 平移

场景 工作流程 机器人 工具和工件 碰撞检测 规划历史 其他 日志

13.5 Robot Configurations

Axes		Six	Six
Types of Configuration	Name	UR_UR5_Like	SphericalWrist_SixAxis
	Description	6 axis collaborative robot	Common split-ball type wrist industrial robot
Illustrations	DH Specification		
	Reference frame and axis No.		
Interpretation		0: Reference frame 10; 1: Reference frame 10; 2: Reference frame 2; 3: Reference frame 3;	4: Reference frame 4; 5: Reference frame 5; 6: Reference frame 6;
[robot]_algo example		<pre>{ "algo_type": "UR_UR5_Like", "robot_type": "UR_16E", "dh": [A, B, D, G, F, H], "shoulder_offset": C, "elbow_offset": E, "min_limits": [J1_min, J2_min, J3_min, J4_min, J5_min, J6_min], "max_limits": [J1_max, J2_max, J3_max, J4_max, J5_max, J6_max], "mastering_joints": [J1, J2, J3, J4, J5, J6], "axis_flip": "J1J2J3J4J5J6" }</pre>	<pre>{ "algo_type": "SphericalWrist_SixAxis", "robot_type": "KAWASAKI_CX110L", "dh": [A, B, C, D, E, F], "min_limits": [J1_min, J2_min, J3_min, J4_min, J5_min, J6_min], "max_limits": [J1_max, J2_max, J3_max, J4_max, J5_max, J6_max], "mastering_joints": [J1, J2, J3, J4, J5, J6], "axis_flip": "J1J2J3J4J5J6" }</pre>
13.5. Robot Configurations		↪ Unnecessary }	↪ Unnecessary 340 }

13.6 Description of [robot]_algo.json file

[robot]_algo.json is a robot configuration file which records the property information of the robot. It defines the robot type, DH parameters, range of motion, home position of joints, rotation of joints, etc. The code fragment below is extracted from [robot]_algo.json file as an example.

```
{
  "algo_type": "UR_UR5_Like",
  "robot_type": "UR_16E",
  "dh": [ A, B, D, G, F, H ],
  "shoulder_offset": C,
  "elbow_offset": E,
  "min_limits": [ J1 min, J2 min, J3 min, J4 min, J5 min, J6 min ],
  "max_limits": [ J1 max, J2 max, J3 max, J4 max, J5 max, J6 max ],
  "mastering_joints": [ J1, J2, J3, J4, J5, J6 ],      # Unnecessary
  "axis_flip": "J1J2J3J4J5J6"                       # Unnecessary
}
```

The descriptions of attributes are as follows.

- **algo_type**

The type of configuration of the robot. The robot models are divided into 13 categories in Mech-Viz V1.6.0. For more information about the robot configuration, please refer to [Robot Configurations](#).

- **robot_type**

The type name of the robot, which should be the same as the folder name of the robot model file, or else the robot model cannot be located.

- **dh**

The DH parameters of the joints of the robot.

- **minlimits, maxlimits**

Defines the minimum and maximum range of motion for each axis. Except for FANUC and NACHI, the accurate range of motion for each axis can be found in the product specification for robots of most brands.

Hint: With this attribute, each joint will be added with two parameters, i.e., the minimum range of motion and the maximum range of motion. This attribute relates to **Axis_flip**, and the two parameters and their negative/positive values should be reversed under certain circumstances.

- **mastering_joints**

Defines the positions of joints when their rotation angles are 0 degree. When the value of parameters are all set to 0, the robot pose is the same as the one during modeling.

Hint: For some robots such as KUKA robots of all series, when the value of J2 and J3 are set to 0, its pose is not same as the default home position in Mech-Viz, and therefore the value of J2 and J3 in mastering_joints should be adjusted.

- **axis_flip**

Defines the direction of rotation of the axes. After modeling the robot, please modify this attribute if the direction of rotation does not match with that of the real robot.

Hint: This attribute will affect other attributes such as `minlimits`, `maxlimits`, `link3_dynamic_limits`, `link4_dynamic_limits`, etc.

When using a newly created robot model, please make sure that the settings of the following attributes correspond with the real robot.

- **axis_flip**

Check if the rotation of the axes of the real robot corresponds to the settings of this attribute.

- **dh (Important)**

You can check if the pose of the simulated robot in Mech-Viz corresponds to that of the real robot by using the following two methods.

- Synchronize the JPs and compare the TCP
- Synchronize the TCP and compare JPs

Hint: The closer the DH parameters of the real robot are to the theoretical values in the parameter file, the more precise the robot is. An error of less than 1mm is generally considered acceptable.

- **mastering_joints**

Check if the pose of the simulated robot in Mech-Viz corresponds to that of the real robot.

Hint: Please pay attention to axis-1, axis-4, and axis-6 while checking.

- **min_limits/max_limit**

Check if the pose of the simulated robot in Mech-Viz corresponds to that of the real robot.

Hint: The soft limits of the robot must not exceed its hard limits.

13.7 Description of [robot]_profile.json file

[robot]_profile.json records some basic information of the robot, including home position, payload, reach, and number of axes. The code fragment below is extracted from [robot]_profile.json file as an example.

```
{
  "home_jps": [0,90,0,0,90,0],
  "max_tcp_vel": 7,
  "max_joint_vel": 500,
  "reach": 2.55,
  "payload": 40,
```

(continues on next page)

(continued from previous page)

```

    "axes": 6
  }

```

The descriptions of attributes are as follows.

- **home_jps**
Define the default home position in Mech-Viz; the units used here are degrees.
- **max_tcp_vel**
Define the maximum TCP velocity (unit: m/s) for filtering the singularities.
- **max_joint_vel**
Define the maximum joint velocity (unit: m/s) for filtering the singularities.
- **reach**
Robot reach (unit: m) that will be displayed in the robot library.
- **payload**
Payload of the robot (unit: kg) that will be displayed in the robot library.
- **axes**
The number of the axes that will be displayed in the robot library.

13.8 Notes for OBJ Collision Models

Table 1

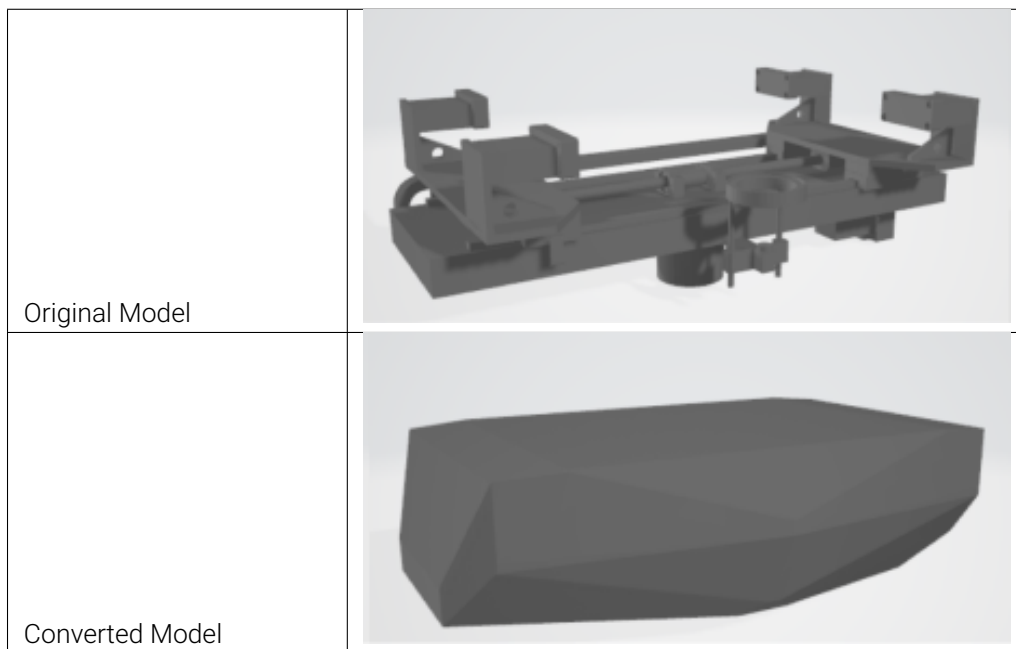
Source of the OBJ Model	Output using the STL to OBJ script	Export by the Model Editor
Legality (Whether it is an assembly of convex polyhedra)	Legal	Legal
Identifier in the file	None	mmind_convex 1.6.1
How to use models from Mech-Viz project 1.6.0 in 1.6.1	Re-import	Run the project directly
How to use models from Mech-Viz of versions before 1.6.0 in 1.6.1	Re-import	NA (There is no model editor in Mech-Viz of versions before 1.6.0)

Table 2

Source of the OBJ Model	OBJ file converting tool	Export by other software
Legality (Whether it is an assembly of convex polyhedra)	May not be legal	May not be legal
Identifier in the file	mmind_convex script	None
How to use models from Mech-Viz project 1.6.0 in 1.6.1	Re-import	Re-import
How to use models from Mech-Viz of versions before 1.6.0 in 1.6.1	NA (Models of this source were not used in Mech-Viz of versions before 1.6.0)	Re-import

Note:

- **“Re-import”** refers to adding the collision detection model again in *Tool and Workobjects* → *Collision Models* and convert the OBJ model following the instructions in the pop-up window.
- The shape of the model may change after re-importing. In this case, please use *Blender* or *Model Editor* in Mech-Viz to edit the models that are not entirely composed of convex polyhedra before importing to make sure that they can meet the requirement.



13.9 Open Projects of Mech-Viz 1.6.0 or Lower

Since the source of the OBJ models are different, their legalities are different as well. Please refer to [Notes for OBJ Collision Models](#) for detailed instructions.

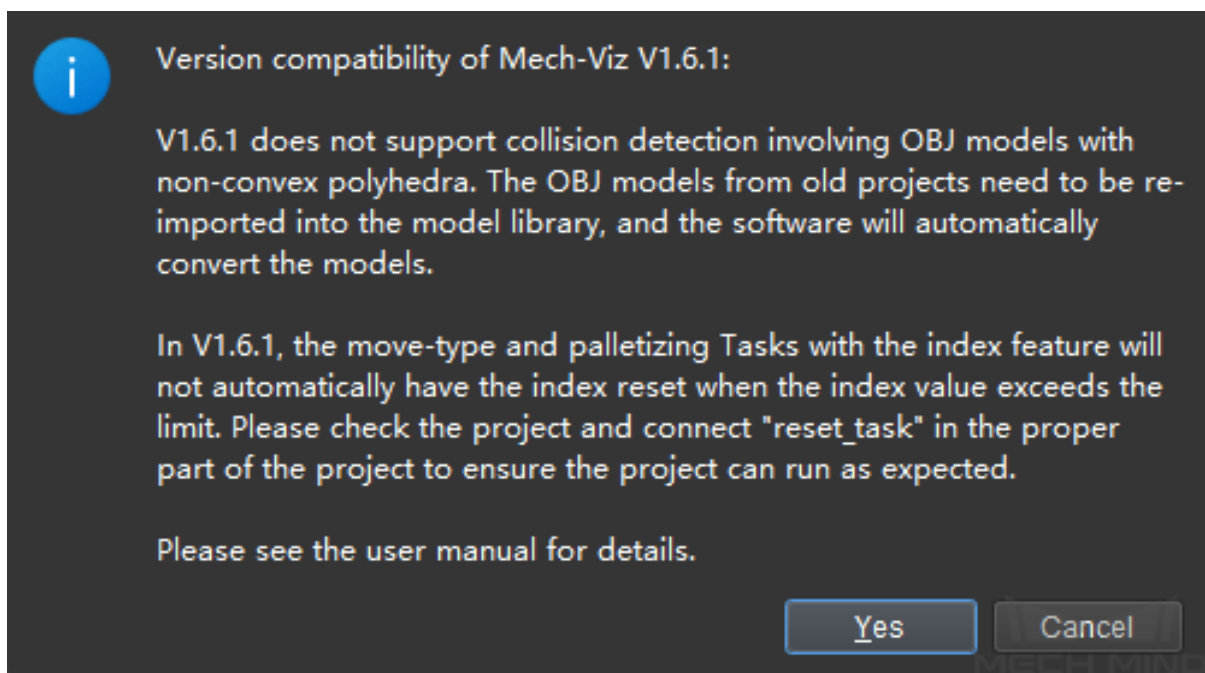
When you open a project created with Mech-Viz 1.6.0 or lower in Mech-Viz 1.6.1, there will be pop-up windows with instructions for OBJ models of different types. Please proceed according to the instructions in the pop-up windows.

13.9.1 Models in the Collision Detection Model Library

Note: Collision detection model library refers to the `collision_models` folder in the project folder.

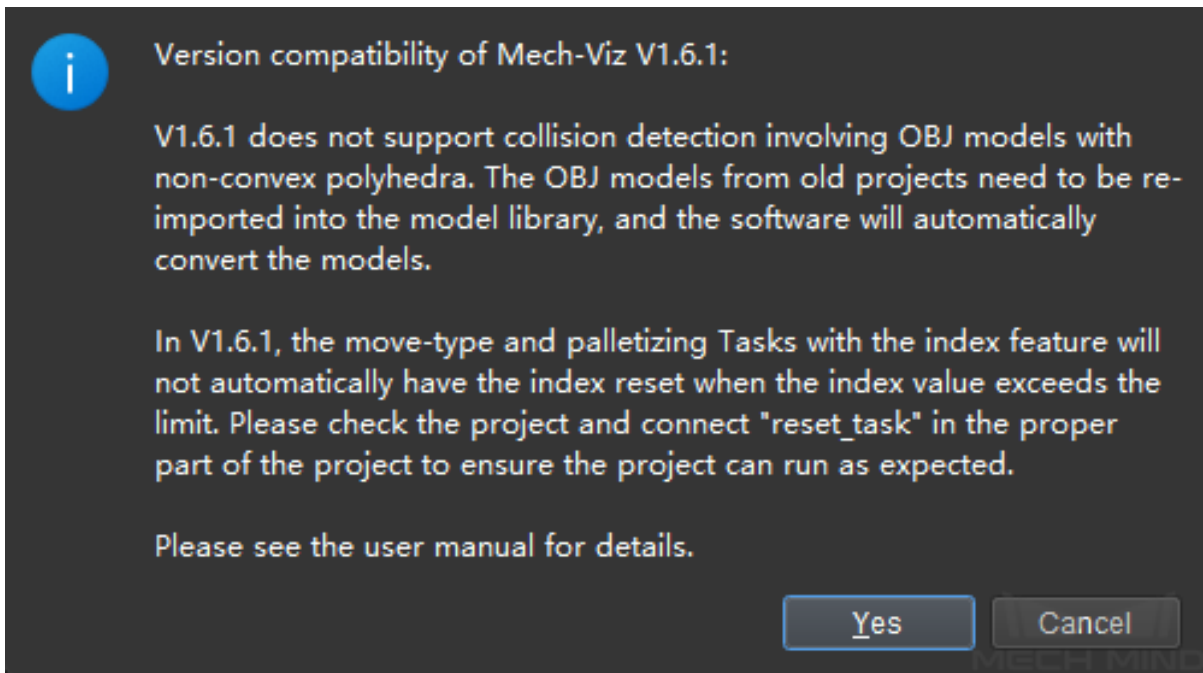
1. When the collision detection model library **only contains models exported by the Model Editor**:

If models exported by the Model Editor are saved in the collision detection model library and set as the end effector models, a **Version compatibility of Mech-Viz V1.6.1** window will pop up when opening the project. Click Yes to close the pop-up window, and the project can be opened, run, or simulated normally.

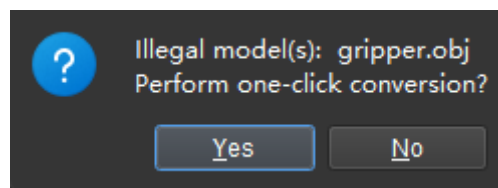


2. When the collision detection model library only contains **illegal models without identifier** or **illegal models with the `mmind_convex` script identifier**:

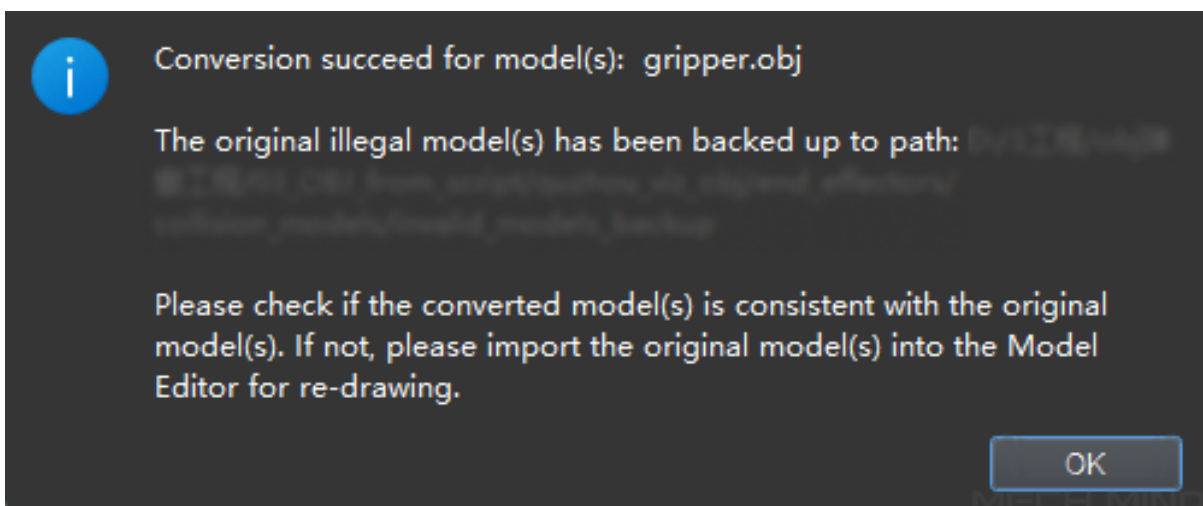
If **illegal models without identifier** or **illegal models with the `mmind_convex` script identifier** are saved in the collision detection model library, a **Version compatibility of Mech-Viz V1.6.1** window will pop up when opening the project.



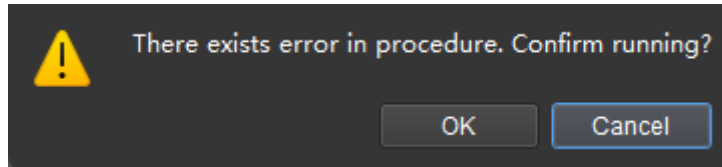
Click Yes and an **Illegal model(s)** window will pop up.



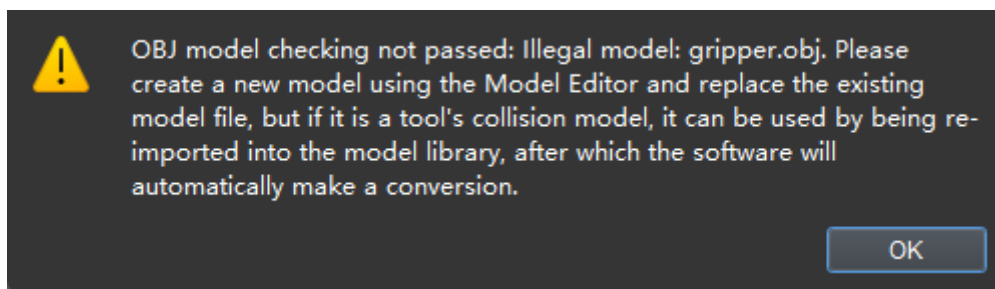
- Click Yes to open the project and convert the models. After the conversion is complete, a **Conversion succeed for model(s)** window will pop up. Click OK to close the pop-up window, and the project can be run or simulated normally.



- If the project is not saved, models in `end_effectors\collision_models` remain illegal without identifiers.
- If the project is saved, the models are converted to assemblies of convex polyhedra with identifiers.
- Click *NO* to open the project. However, an alert saying “**There exists error in procedure. Confirm running?**” will pop up if you want to simulate or run the project.



If you click *OK* to continue running, an **OBJ model checking not passed** window will pop up.

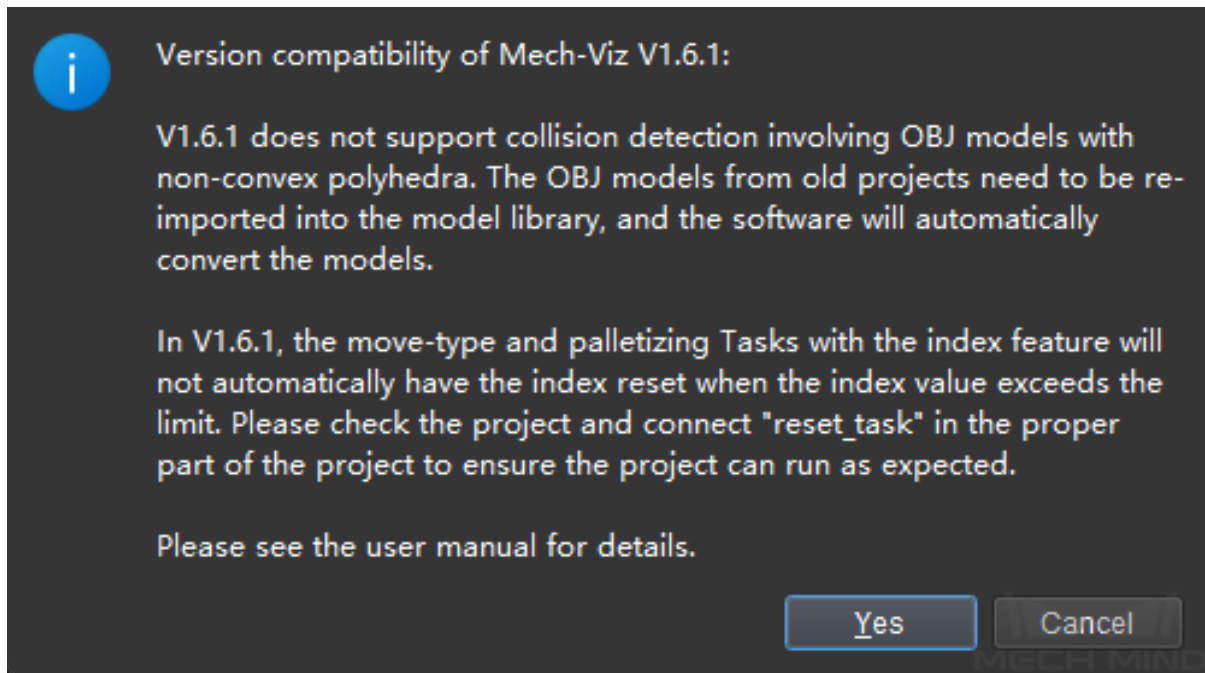


3. When the collision detection model library contains models of various types:

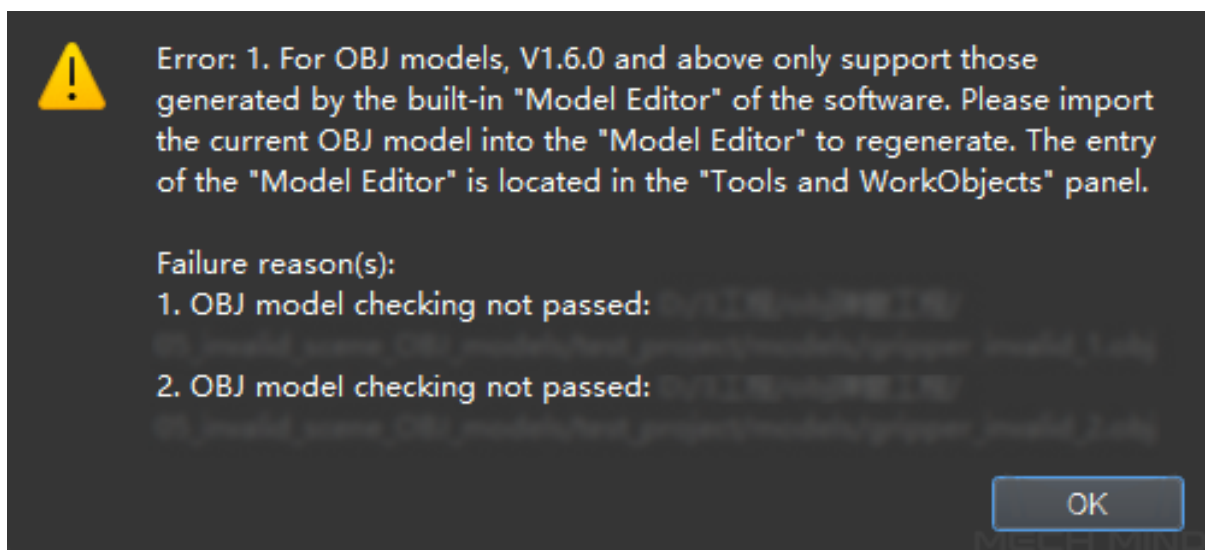
If the collision detection model library contains both legal models and illegal models, windows asking whether to convert illegal models will pop up, as described above.

13.9.2 Models in the Scene Model Library

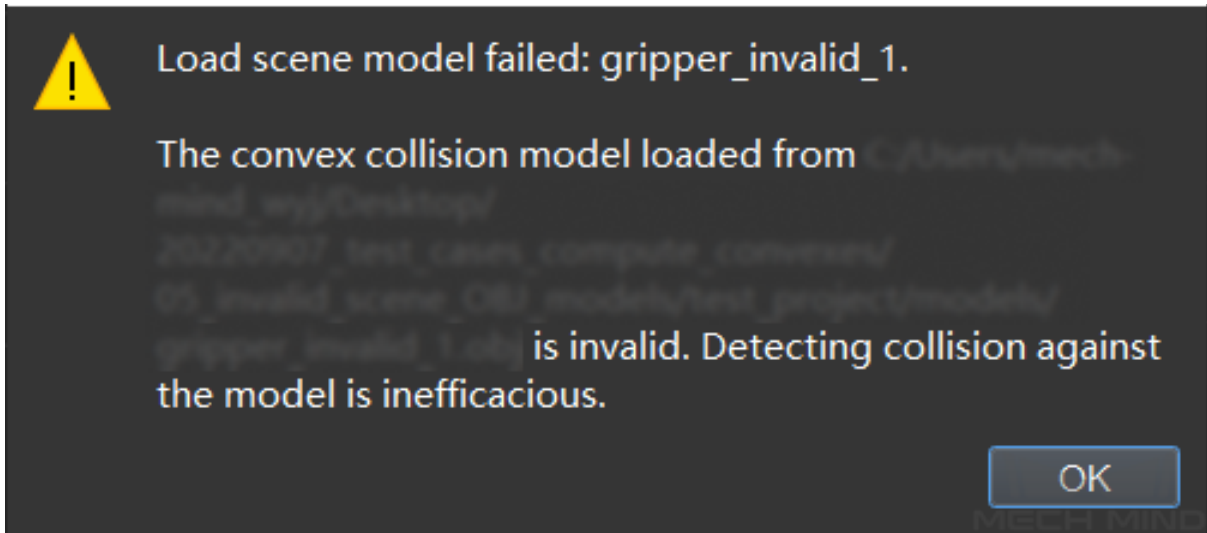
If the scene model library contains illegal models with identifiers, a **Version compatibility of Mech-Viz V1.6.1** window will pop up when opening the project.



Click Yes, and an error message as shown below will pop up.



Click *OK* to open the project. However, the illegal scene models will not be displayed in the 3D simulation area. If you want to simulate the project, a **Load scene model failed** will pop up.



Except for the above two scenarios, please follow the instructions in the pop-up windows to continue.