# Importing 3D objects

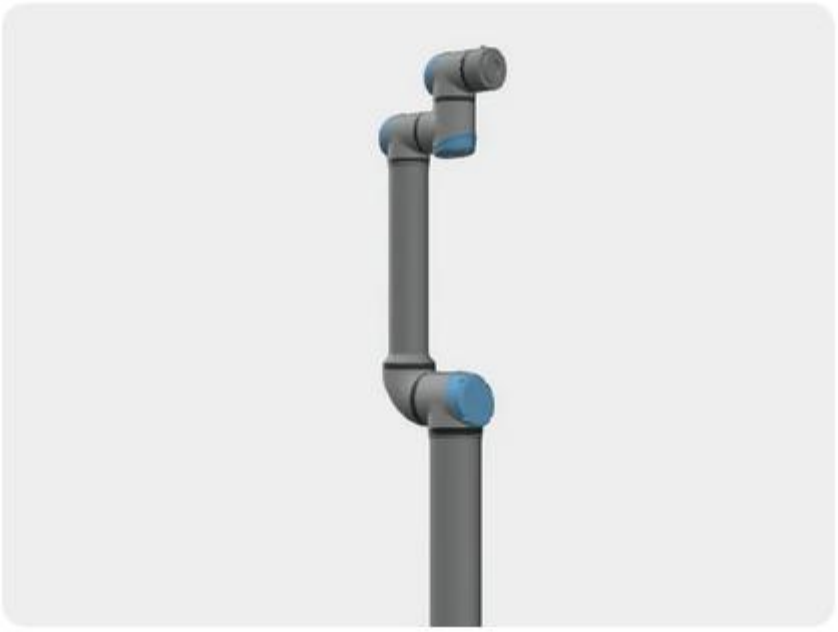
Modern robotic simulation environments rely heavily on accurate 3D models to represent robots, tools, and workcells with high fidelity. Although RoboDK offers an extensive built-in library containing over 600 industrial robots from major manufacturers, real-world applications often require importing custom components, end-effectors, or even entirely new robot designs. This capability allows engineers and researchers to simulate unique setups, verify kinematic behavior, and optimize processes before physical implementation.

In this chapter, we explore the process of importing 3D objects—such as robot parts, workpieces, and fixtures—into RoboDK. We will particularly focus on how commonly used CAD formats like STL and STEP (STP) can be integrated seamlessly into RoboDK stations. To illustrate this workflow, a Universal Robot UR10e model will be used as a practical example, demonstrating how CAD data obtained from the manufacturer's website can be prepared, imported, and aligned within the RoboDK environment for simulation purposes.

## Importing 3D models into RoboDK

While RoboDK provides a comprehensive library with over 600 industrial robot models, users may sometimes need to work with a robot or component that isn't included. In such cases, RoboDK allows you to import custom 3D models directly into your station. This feature supports common CAD file formats such as STL and STEP (STP), enabling seamless integration of external models for simulation or customization purposes.

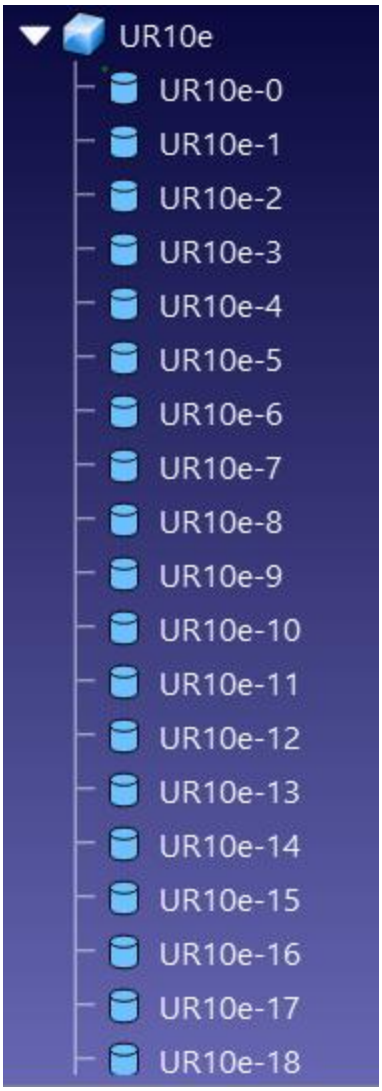
In this section, we will demonstrate the process of importing a 3D model using the Universal Robot UR10e as an example. Although this robot already exists in the RoboDK library, we will use it as a proof of concept to show how CAD data—available from the Universal Robots website—can be brought into RoboDK for visualization, configuration, and further simulation tasks.



UR10e

UR10e STEP

The STEP file can be directly imported into RoboDK's station; however, it initially appears as a single rigid object, meaning that no links or joints are yet defined. To separate the model into its individual components, right-click on the robot's name in the station tree to open the context menu, and then select Split Object. RoboDK will automatically create a hierarchical tree structure containing all the child elements of the imported model. In this case, the splitting process generates approximately 6,350 individual objects, representing the detailed parts of the robot assembly.



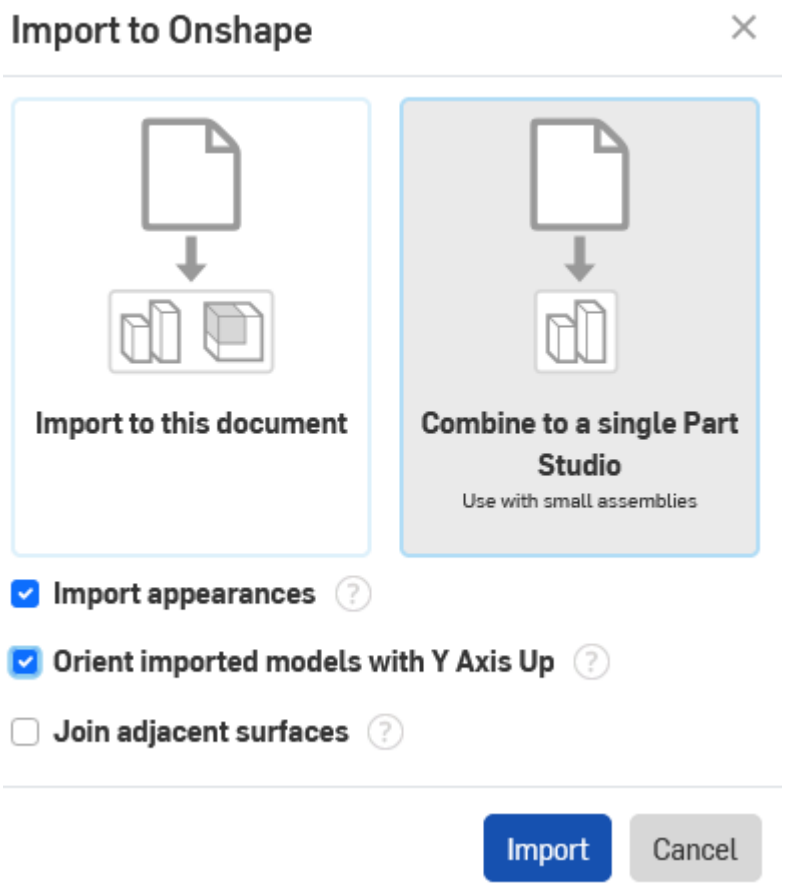

The UR10e is a six-degree-of-freedom (6-DoF) robot that must be properly identified and defined within RoboDK. At this stage, each link of the robot consists of multiple smaller objects, meaning that the links do not yet appear as single, unified bodies. Manually identifying and defining the base and individual links from these numerous objects would be extremely difficult—if not practically impossible. Fortunately, there is a more straightforward and time-efficient approach to achieve this setup.

## Import 3D models not directly to RoboDK

It is recommended to import the CAD file into a CAD software of your choice, such as AutoCAD, SolidWorks, CATIA, ANSYS, or Onshape. In this guide, we use Onshape as the test environment because it is freely accessible and available to all users.

In Onshape, start by creating a new document. Then, using the "+" button in the bottom toolbar, you can import the CAD model into your workspace. The upload process may take some time as the model is transferred to the Onshape server and processed. Once the import is complete, you will receive a notification confirming that the model is ready for use in your workspace. You can use the following settings while importing the CAD model.

⚠ Alert: Don't forget to enable 'Combine to a single Part Studio', 'Import Appearances' and 'Orient imported models with Y Axis Up', before pressing the Import button.




Once the model is uploaded, you can view the list of available parts. In the case of the UR10e, there are 23 parts in total. It is recommended to remove any components that are not essential for defining the manipulator's links, such as cables or connectors. After cleaning up the model, you should be left with the base link and links 1–6. Rename these parts appropriately to ensure they can be easily identified later.

Next, right-click on the Part Studio containing the imported CAD model and select the Export option. In the Export dialog, make sure to adjust the following settings:

- Format should be STEP
- Preprocessing should be automatic.
- Export unique parts as individual files.

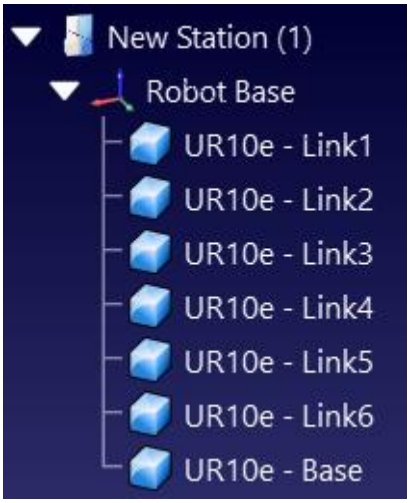
By clicking the Export button, Onshape generates a ZIP file containing seven individual STEP files. These files represent the robot's parts, which are now ready to be imported into RoboDK.

## Import 3D model from Onshape to RoboDK

First, define a robot base reference frame and then drag and drop all the part files into the station. Ensure that all parts are properly nested under the defined reference frame. Importing all the robot components may take some time.

Once the import is complete, you may notice that the model's textures appear different from the original CAD files. To correct this, go to Tools → Change Colors, and select the objects one by one to remove any unwanted colors that do not belong to the model. Typically, the last color in the stack is the incorrect one. Right-click on that color and select Delete to restore the original textures of the robot.

Finally, verify that the hierarchy is correctly structured — only the base and links should appear in the list, and none of the robot's parts should be misaligned or displaced.



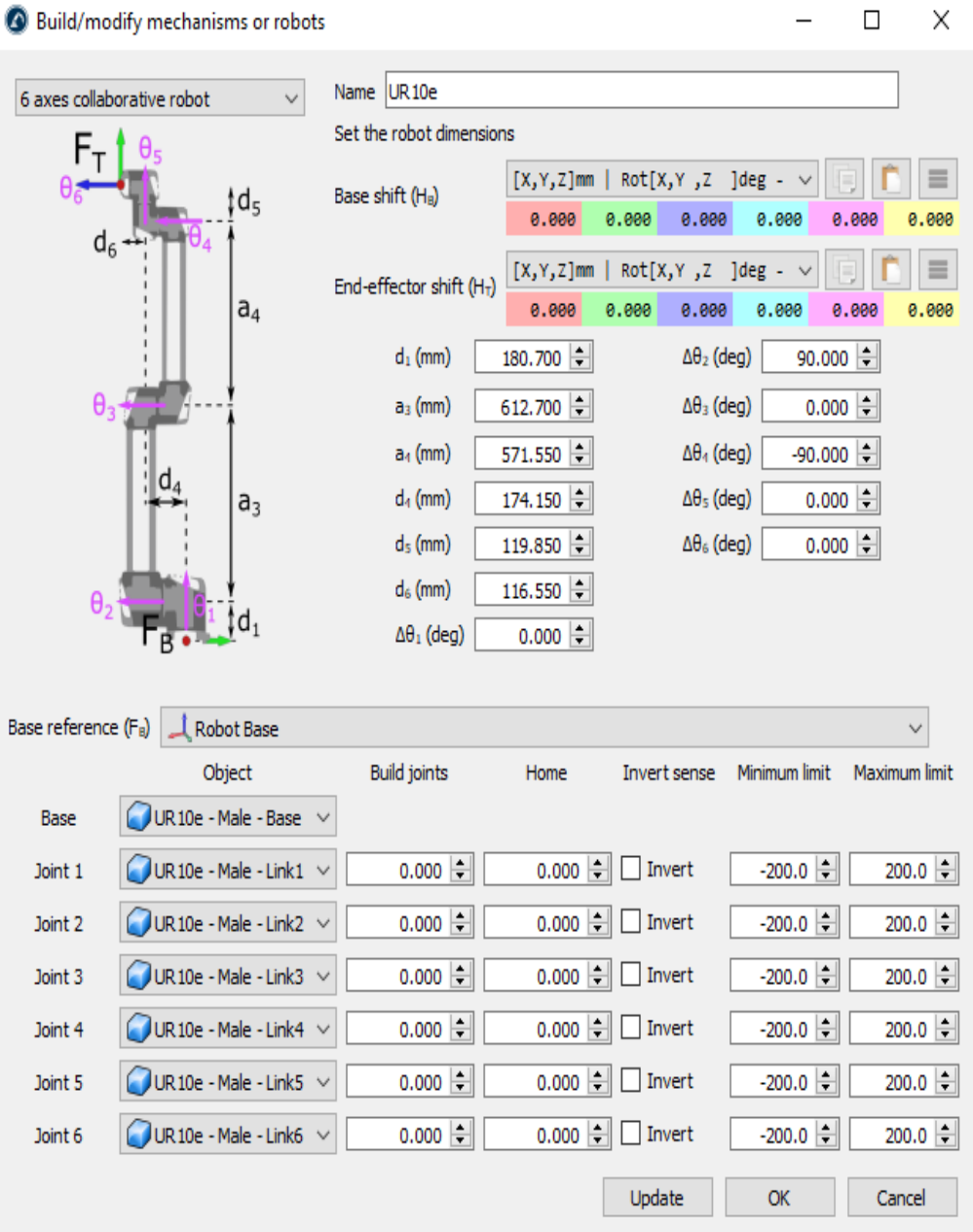


## Model Mechanism or Robot

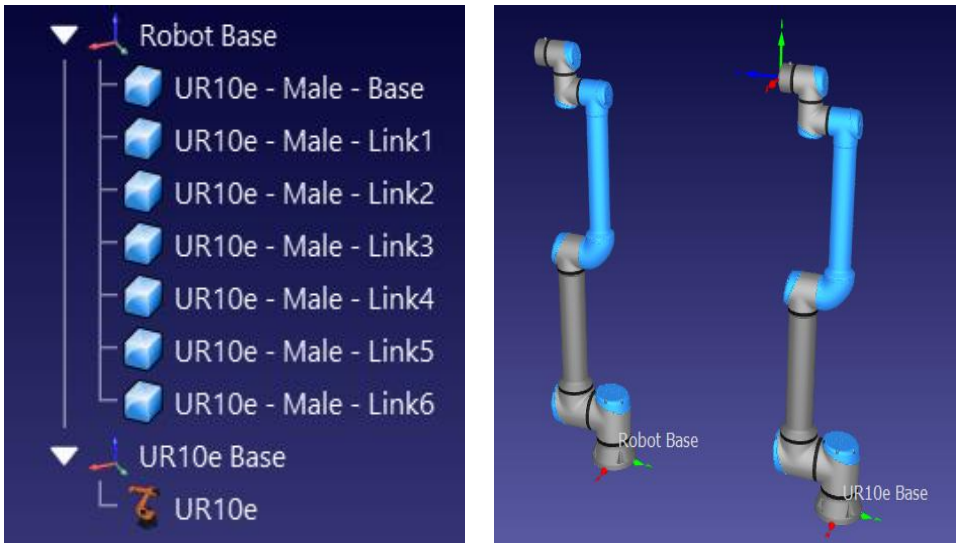
Now it is time to define the Denavit–Hartenberg (DH) parameters of the robot to accurately establish the joint axes. The DH parameters can be obtained directly from the robot's datasheet or, in most cases, provided by the manufacturer as a DH table.

From the Utilities menu, select Model Mechanism or Robot. Choose the appropriate robot or mechanism type and assign it a suitable name. If the base and links have been properly named, they will automatically appear in the corresponding Base and Joint Information fields.

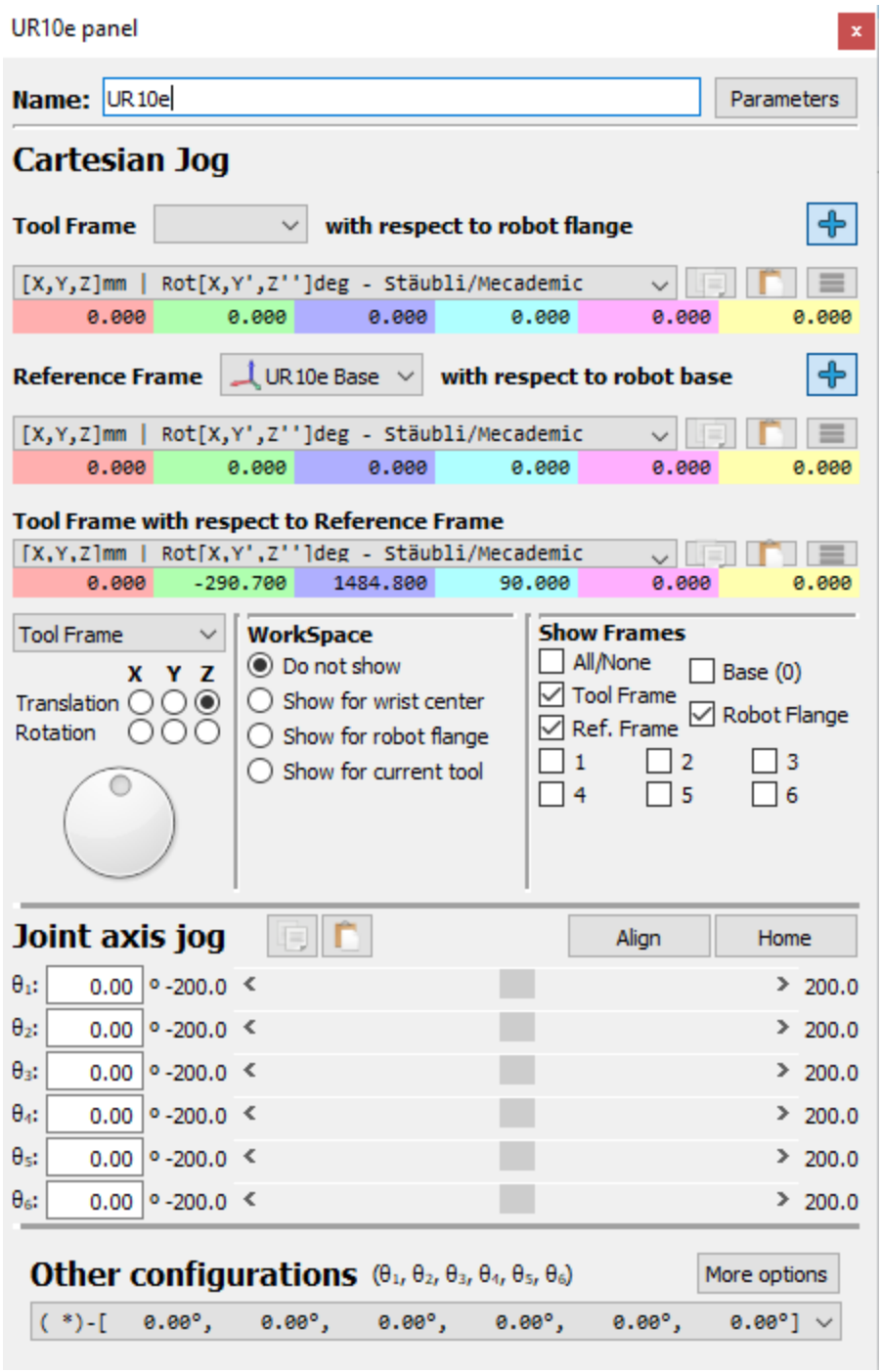
The DH parameters can then be defined as illustrated in the following image.




Once you are satisfied with the defined DH parameters, you can further refine the model by setting additional options, such as the home position and joint constraints (maximum and minimum limits for each joint). After completing this process, a new instance of the robot model will be created and displayed in the workstation, using the same name specified in the Mechanism dialog box.




Double-click on the newly defined robot in the tree to open a panel where you can inspect all joint axis movements. If any joint does not behave as expected, this indicates that the DH parameters may not have been defined correctly, or there is an issue with the reference frames. In such cases, revisit the previous steps to ensure that all definitions and references are accurate.




## Some Important Considerations

When comparing the robot generated from the DH parameters with the UR10e available in RoboDK's library, a few minor differences become apparent. In the imported model, each part comes with its own reference axis, and variations in the definition of these reference frames can lead to differences in the link twists. The effect of this can also be observed in the home position (vertical upright) specified in the following table, which reflects the same variations.

| Home Position | RoboDK library | Onshape |
|---|---|---|
| $\theta_1$ | 0 | 0 |
| $\theta_2$ | 270 | 0 |
| $\theta_3$ | 0 | 0 |
| $\theta_4$ | 270 | 0 |
| $\theta_5$ | 0 | 0 |
| $\theta_6$ | 0 | 0 |

At the home position, when tool frame axis is recorded with respect to the base reference frame, the pose information is recorded as

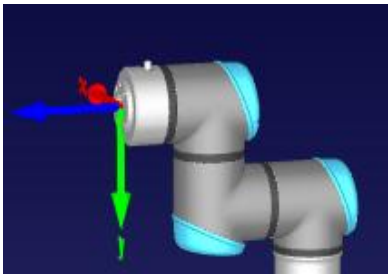
Onshape

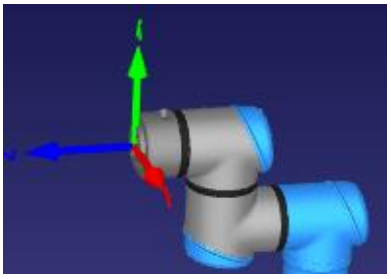
$$(X, Y, Z, Roll, Pitch, Yaw) = (0, -290.7, 1484.8, 90, 0, 0)$$

RoboDK

$$(X, Y, Z, Roll, Pitch, Yaw) = (0, -290.7, 1484.48, 0, -127.27, -127.27)$$

This behavior is also evident when examining the reference frame of the tool. While both frames share the same origin, their x- and y-axes are oriented in opposite directions. Nonetheless, the key consideration is the z-axis, which determines the orientation axis and remains aligned in both cases.


RoboDK


Onshape