Motion of the robot

The study of robot motion constitutes a core component of robotics, encompassing the mathematical and computational principles that govern how a robot moves within its workspace. In the context of RoboDK, robot motion refers to the precise control and simulation of trajectories through the manipulation of joint parameters, tool orientations, and reference frames. This chapter examines the theoretical foundations and practical implementation of robotic motion within RoboDK, including joint-space and Cartesian-space movements, motion interpolation, and trajectory planning. By integrating kinematic modeling with simulation tools, RoboDK enables users to analyze, optimize, and validate robot motions, thereby enhancing the efficiency, accuracy, and safety of automated systems in industrial and research applications.

Trace the movements of the tool

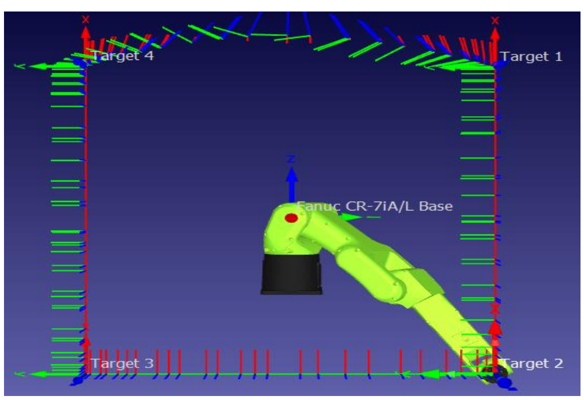
Sometimes, it is beneficial to trace the robot's movements in order to closely analyze its behavior and performance. This functionality can also be employed to generate the desired motion profile, as it provides valuable visual feedback on the robot's trajectory.

To enable the trace option: Tools → Trace → Active

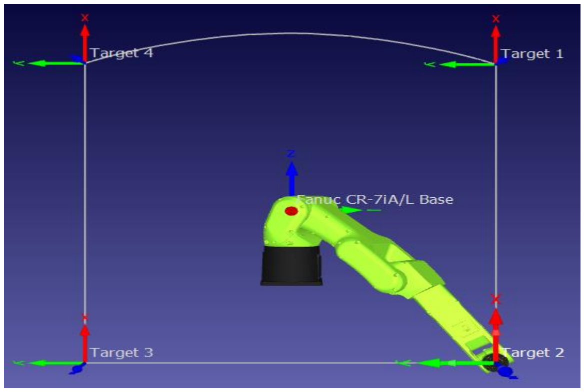
To reset or clear the tracing: Tools → Trace → Reset

Tracing can appear as references, solid lines, dots, points, and spheres.

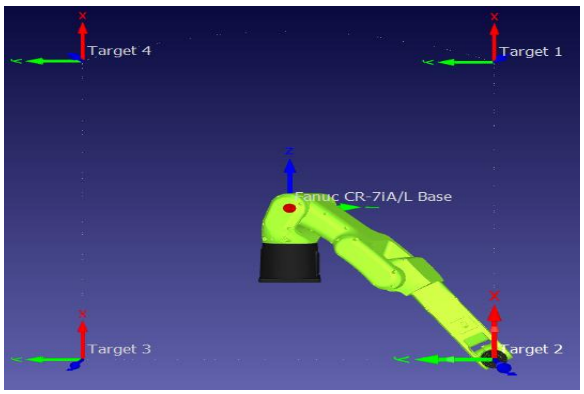
References



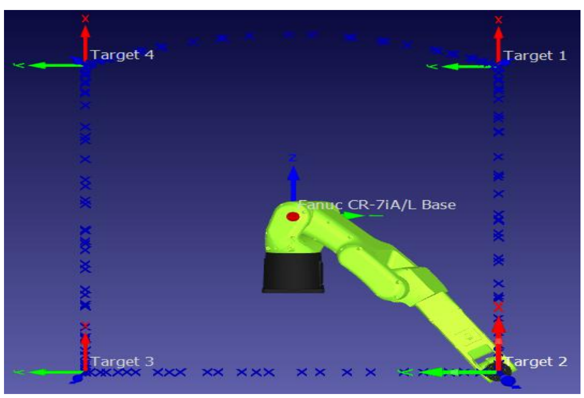
Solid lines



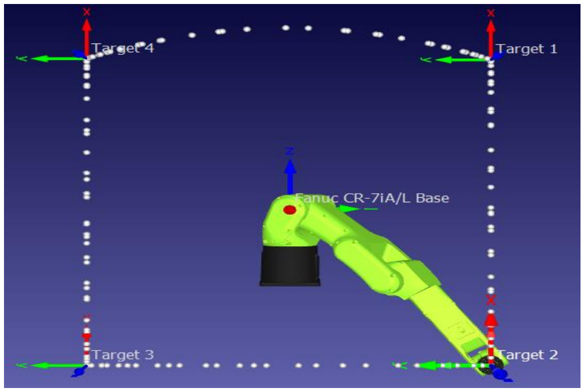
Dots



Points

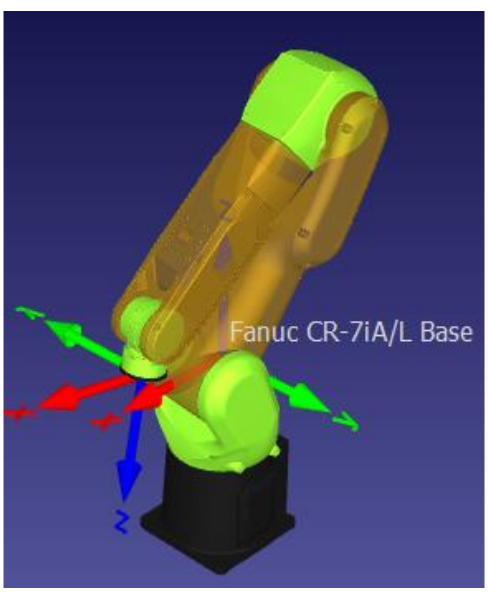


Spheres

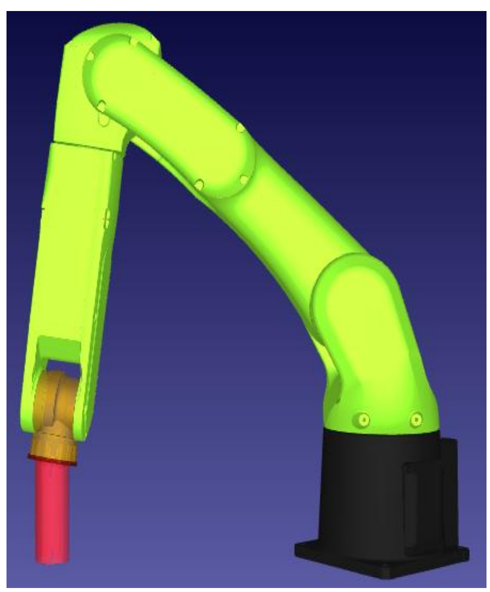


Collision Detection

Collision detection is an integral part of any robotics application, ensuring safe and reliable operation within a workspace. RoboDK provides a robust collision detection feature capable of identifying both self-collision and collisions with external objects. This feature can be toggled on or off using the toolbar icon ; when it is pressed again, it is enabled and appears as . Alternatively, collision detection can also be accessed through the Tools → Check Collisions menu option. When this feature is active, RoboDK visually indicates a collision by changing the color of the robot components or objects involved, thereby providing immediate visual feedback to the user.



Self-collision



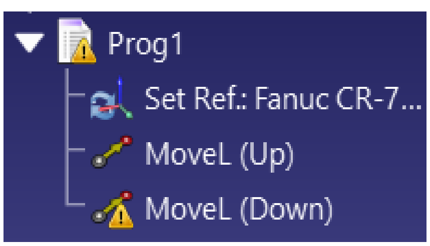
Collision with cylinder

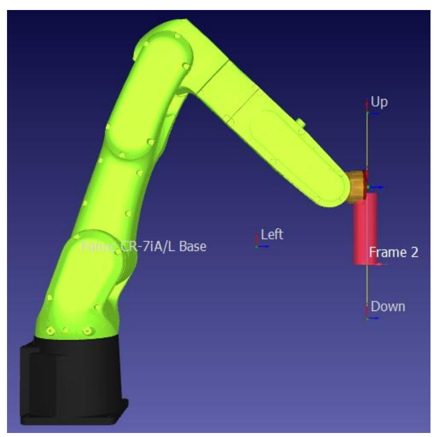
Feasible Path

In any robotic work environment, defining a feasible path is of critical importance. A feasible path must be achieved by the robot without encountering singularities, must remain within the robot's reachable workspace, and must be free from collisions. RoboDK's collision detection feature facilitates the assessment of path feasibility by automatically identifying potential interferences. When collision detection is enabled, the program halts execution upon detecting a collision and visually highlights the objects involved, allowing users to promptly identify and resolve the issue.

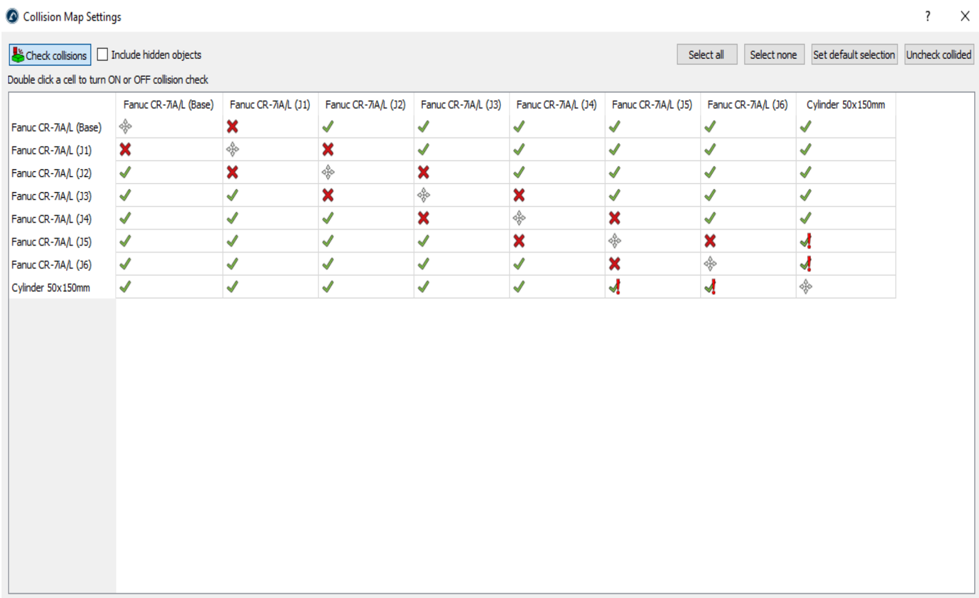
A cylinder is positioned in such a way that it obstructs the robot's path between the Up and Down targets. A simple program is then created to command the manipulator to move from the Up position to the Down position.

When collision detection is disabled, the program executes all instructions smoothly without interruption. However, when collision is detected, the program halts upon identifying a collision and highlights the specific program (in case where multiple programs are active) and the exact instruction responsible for the issue, preventing further execution until the collision is resolved.

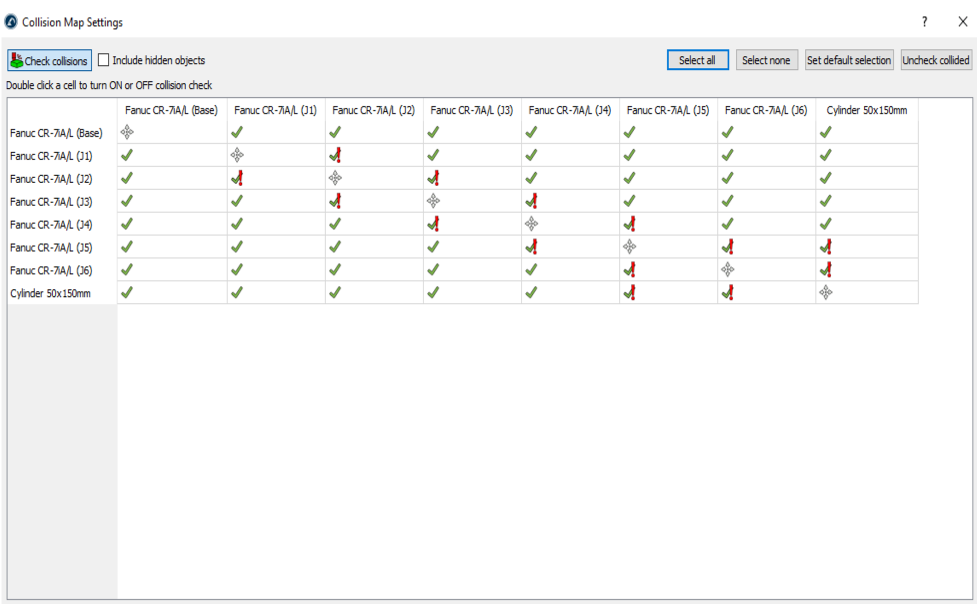




To identify which specific joints are contributing to a collision, the Collision Map can be accessed from the Tools menu. Within this map, check marks and crosses indicate whether collision detection is enabled or disabled for a particular component or interaction. Users can toggle collision checks on or off by double-clicking the corresponding cell. In this case, joints J5 and J6 are in contact with the cylindrical obstacle, which is indicated in the collision map by exclamation marks. If a specific item needs to be excluded from collision detection, double-clicking the diagonal cell with pointed arrows allows the user to enable or disable all possible collision pairs. The current configuration reflects the default settings established by the Set Default Selection option.



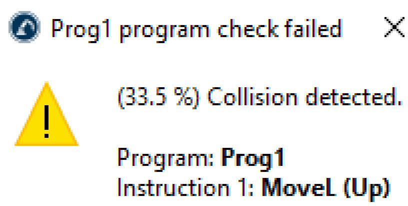
Selecting ‘Select all’ causes the meshing between adjacent joints to be recognized as collisions; for instance, J3 is reported to be in collision with J2 and J4.



The ‘Select none’ option disables all collision detection, while ‘Uncheck collided’ disables only the collisions that have already been detected.

When working with multiple programs containing numerous instructions, identifying problematic instructions one by one can be time-consuming. A more efficient approach is to right-click the main program and select ‘Check Path and Collisions’. This generates a summary highlighting where issues occur.

In this simple example, since only one program is present, the summary indicates that a contact between the robot and the object was detected in Prog1, immediately after executing Instruction 1.



Collision-Free Motion Planner

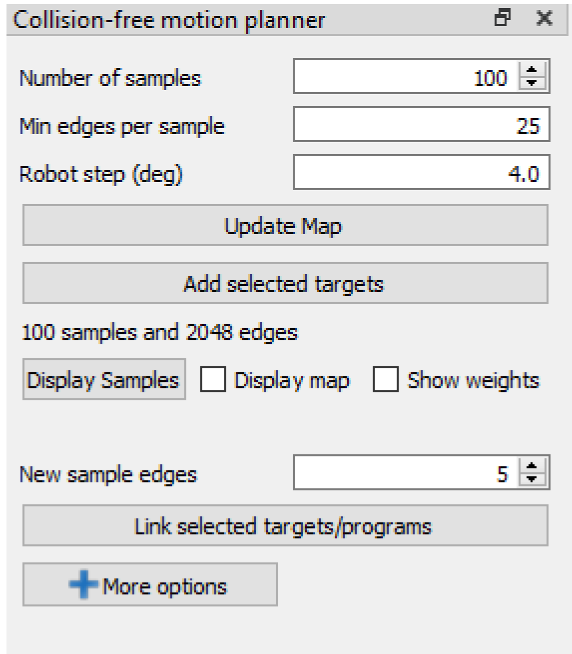
To prevent collisions, it is often effective to introduce an intermediate target point in the robot’s path. This approach is relatively straightforward when only a single robot and a few objects are involved. However, in more complex scenarios with multiple robots or potential collision points, avoiding one collision may inadvertently cause another.

RoboDK can automatically generate collision-free paths within the robot’s workspace using a motion planning algorithm known as the Probabilistic Roadmap (PRM).

To access the motion planner,

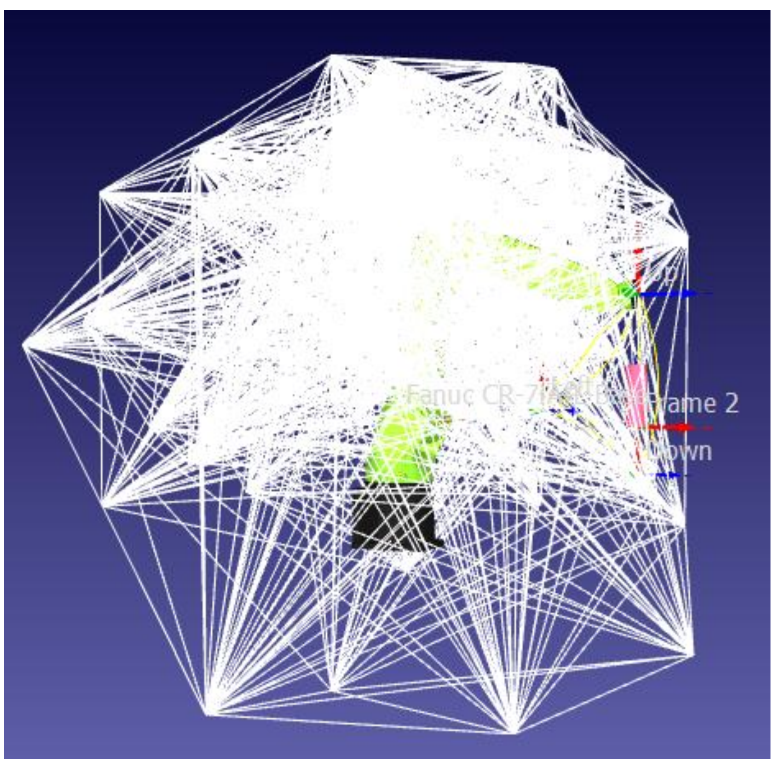
Utilities → Collision-free motion planner

A dialog box will appear, allowing you to specify the number of samples, minimum edges per sample, and robot step size (in degrees). These parameters are important because they determine the set of reachable points within the robot’s workspace.



After setting the initial parameters, you must generate a map (if this is the first time) and then update the map whenever any changes are made to the scene. Once this process is complete, all reachable nodes are calculated, and the map is defined.

Clicking ‘Display map’ will then show a network of all reachable nodes, as illustrated in the following figure.



Once the map has been generated, a path must be defined between two or more target points. This can be done in two ways:

Linking target points manually:

Select the first target point (Up), then hold the CTRL key and select the second target point (Down). Right-click on either target and choose 'Link selected targets'. This creates a program that defines a feasible path between the two points.

Linking programs:

Multiple programs can also be combined to create a single program containing feasible paths. For example, create a program UpDown (with Up as the first target and Down as the second), and another program DownUp (with Down as the first target and Up as the second).

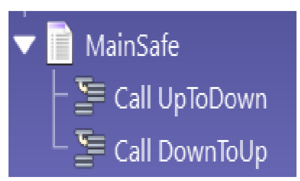
Then, select both programs and, from the Motion Planner dialog box, choose 'Link programs'.

The motion planner will automatically generate three programs as part of this process.

UpToDown

DownToUp

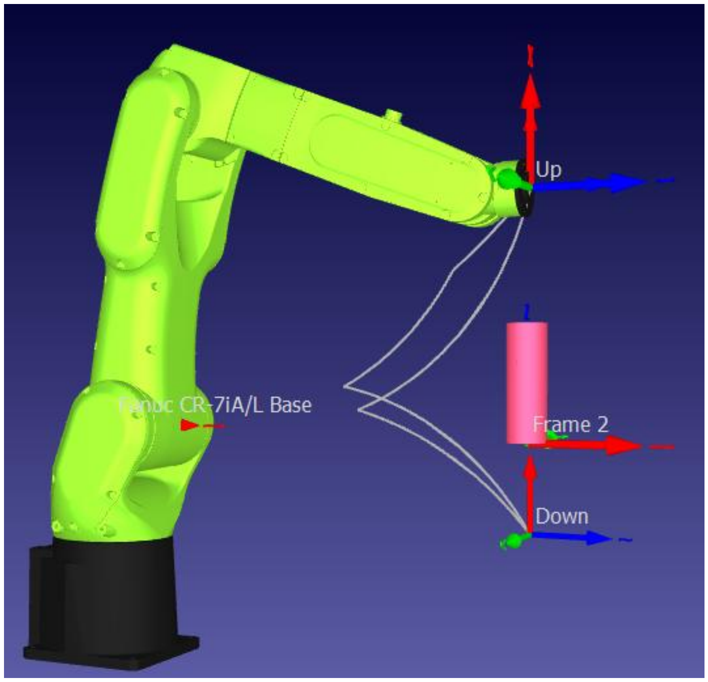
MainSafe



The UpToDown program includes four move instructions, each corresponding to one of four sample points. These points represent the most feasible positions for executing this path.

Similarly, the DownToUp program consists of three move instructions associated with three sample points.

The main program, MainSafe, completes the full motion cycle — moving from Up to Down and returning to Up. The resulting path from these motions is displayed as shown below.


Up
Fanuc CR-7iA/L Base
Frame 2
Down