Optimal Kinematic Design of Robots

Lab 1: Workspace-based design of a SCARA robot

Author:
Anna Possamai

Instructor: *Philippe Wenger*
Department: Control and Robotics Engineering
Date: October 18, 2023

1 Introduction

The aim of this laboratory assignment is to illustrate the use of the concept of workspace in designing an interactive MATLAB function for a SCARA robot.

2 Work Plan

In the first exercise (Work Plan), we were tasked with developing a MATLAB function capable of plotting workspace boundaries for a planar SCARA robot. The user provides the joint limits and link lengths. The function also considers a disk-shaped obstacle that obstructs only link 1.
Direct Kinematics Model
To start, we computed the direct kinematics model to determine the end effector's position as a function of the joint angles:

$$f(q_1, q_2) = \begin{cases} x & = a_1 \cos(\alpha_1) + a_2 \cos(\alpha_1 + \alpha_2) \\ y & = a_1 \sin(\alpha_1) + a_2 \sin(\alpha_1 + \alpha_2) \end{cases} \tag{1}$$

Singularities
We then computed the Jacobian of $f(q_1, q_2)$ and set it equal to zero to identify the joint singularities. The obtained singularities occur at $q_2 = 0$ and $q_2 = \pi$.

2.1 Workspace Analysis

MATLAB Function Development
The MATLAB main function $main\_workspace$ implemented allows the user to input joint limits and link lengths.
Initially the MATLAB code for the workspace boundaries of a planar SCARA robot without the obstacle has been computed.
We created a function $workspace(l1, l2, q1_{min}, q1_{max}, q2_{min}, q2_{max})$, to plot the workspace boundaries.
Inside this function we calculated the direct kinematic model using the function $dkm(l1, l2, q1, q2)$, which takes link lengths and joint limits as inputs and outputs the corresponding (x, y) coordinates.

Workspace Visualization

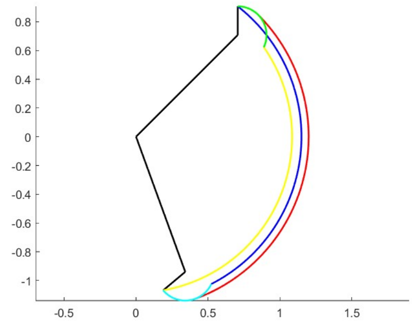

Figure 1: Workspace boundaries, without obstacle

In the figure 1, we can observe the workspace boundaries when $l_1 = 1, l_2 = 0.2, -70 \leq q_{1min} \leq 45, -70 \leq q_{2min} \leq 45$. The lines represent the end-effector paths, in particular:

- The yellow line when q_1 varies and q_2 is set to $q_{2_{min}}$.
- The blue line when q_1 varies and q_2 is set to $q_{2_{max}}$.
- The red line when q_1 varies, and q_2 is set to $q_2 = 0$. In this scenario, the robot is fully extended, and the line represents the maximum points reachable by the end effector.
- The cyan-blue when q_1 is set to $q_{1_{min}}$ and q_2 varies.
- The green line when q_1 is set to $q_{1_{max}}$ and q_2 varies.

We then developed a function called $workspace\_obst$. This function takes the link lengths l_1, l_2, the link joint limits $q_{1_{min}}, q_{1_{max}}, q_{2_{min}}, q_{2_{max}}$ and the obstacle's position and radius (x_o, y_o, r_o) as inputs.

The function first checks if the obstacle obstructs link 2. If it does, an error message is displayed, and no image is generated. Otherwise, the robot's workspace is plotted.

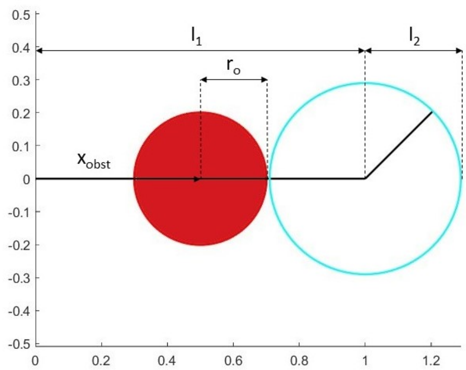

Figure 2: Condition of no obstruction of link 2

To assess whether the obstacle obstructs link 2, we analyzed the figure 2 and determined whether the sum of the obstacle's distance $\|X_{obst}\|$ and its radius r_o is greater than or equal to the difference between link 1 and link 2: $\|X_{obst}\| + r_o \geq (l_1 - l_2)$

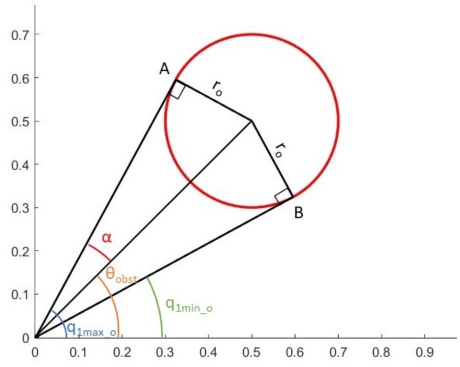

Figure 3: Angle limits condition

To plot the robot's workspace while considering the obstruction of link 1 by the obstacle, we analyzed the figure 3. Specifically, we determined that link 1 can intersect the obstacle at points A and B during the robot's movement.
To avoid passing through the obstacle during its movement, the robot halts the movement of link 1 when it intersects the obstacle.
Therefore, we designed the robot's workspace by defining the limits of joint 1 in relation to the obstacle.
The values of q_{1max_o} and q_{1min_o}, representing the maximum and minimum limits for q_1 due to the presence of the obstacle, were calculated by analyzing Figure 3. The following formula was used for the calculation:

$$q_{1max_o} = \theta_{obst} + \alpha,\ \ q_{1min_o} = \theta_{obst} - \alpha$$

Where:

$$\theta_{obst} = atan2(y_o, x_o),\ \alpha = atan2(r_o, l_t),\ l_t = \sqrt{x_o^2 + y_o^2 - r_o^2}$$

Next, we computed the direct kinematic model using the function $dkm(l1, l2, q1, q2$, as described earlier, to obtain the (x, y) coordinates corresponding to each joint value while considering the limits.
Finally, we plotted the workspace corresponding to the joint limits.

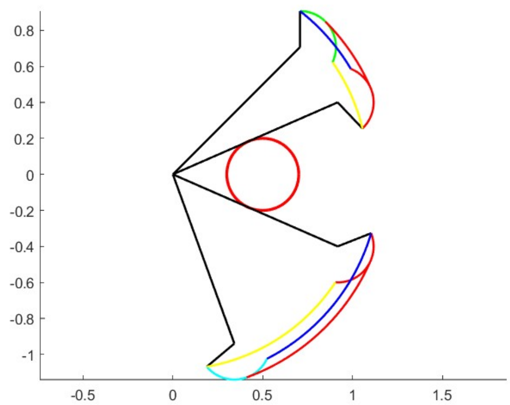

Figure 4: Workspace boundaries, links and obstacle

In this figure, you can observe the links (in black) and the obstacle obstructing link 1, represented by the red circle. As can be seen, the presence of the obstacle restricts the workspace.

It should be noted that the code does not handle the scenario where one of the limits of q_1 intersects with the obstacle. The only case currently handled is when both the minimum and maximum limits of q_1 fall within the intersection area. In this case, the trajectory is not plotted, and an error message is displayed. A possible future implementation will need to account for and manage other intersection scenarios.

3 Design for Point-to-Point Tasks

In this section, we consider a SCARA robot with joint limits of $\pm 132°$ and $\pm 141°$ for θ_1 and θ_2, respectively. A disk-shaped obstacle with a radius of 0.3 m is placed at (1.2, 0.1) m. The robot's maximum reach is set to 2 m.

It is request to find the optimal link length ratio for the robot to pick parts from an L-shaped palette with dimensions of 2 m by 1 m and a width of 0.5 m and place them into another identical palette.
In "pick and place" tasks, we typically control the joint angles directly ($\mathbf{q}$), specifying the desired positions of the actuators. This means that in the "pick and place" task, the robot may encounter singularities. However, this is not an issue since the robot is involved in a relatively straightforward operation of picking up and placing objects. Temporary singularities that may occur during the process can be managed without significant disruptions or serious consequences. However, it is important to avoid singularities in the positions where the robot needs to pick up the points.

Our goal in this case is to maximize the workspace in order to contain the L-shaped palettes. Two aspects were considered for selecting the 'best' link ratio:

1. The first is the constraint related to the end-effector maximum distance: $l_1 + l_2 \leq 2$.
2. The second pertains to obstacle avoidance concerning l_1: the difference between the distance from the obstacle and the obstacle's radius must be greater than l_1, i.e., $d_o - r_o \geq l_1$, where d_o is the distance to the center of the obstacle and r_o is the obstacle radius.

The second consideration merits further exploration. If the length of link 1 was greater than the distance $d_0 - r_0$, as illustrated in the example in the figure 5, the workspace would be divided into two parts. This is because, as depicted in the figure, link 1 cannot pass through the obstacle. This particular situation would render it impossible for the robot to transition from one side to the other, significantly reducing the workspace area. Another aspect to consider is the optimal length of the two links. In general, it can be demonstrated that the most favorable ratio between the two links is equal to 1.

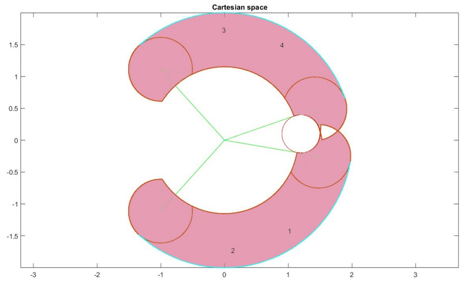

Figure 5: Workspace boundaries, links and obstacle

By adhering to these constraints, we selected the best values for the link lengths as $l_1 = 0.9$ m and $l_2 = 1.1$ m. This configuration was found to be suitable for the pick-and-place task since the robot can reach every point of the L-shaped palette and place items into another L-shaped palette. The L-shape palette was rotated by $-90°$ to fit it into the workspace, as shown in the figure 6:

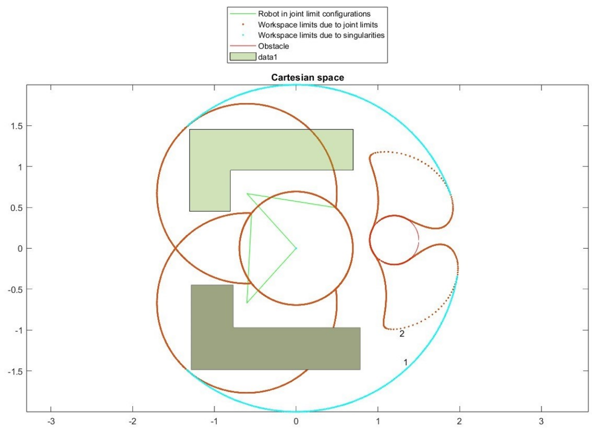

Figure 6: Workspace boundaries for pick and place task

This configuration is ideal for the pick-and-place objective since the robot can reach every point on the L-shape and transfer items to another L-shape palette.
During the movement from one palette to another, the robot may encounter a singularity. However, as said before, this is not a problem for this task because it only requires point-to-point reaching, and there is no need for a continuous path.

4 Design for Process Tasks

In this section, we address the challenges associated with process tasks.
In such scenarios, singularities can be problematic, as small variations in joint orientations can lead to significant variations in the (x, y) coordinates of the end effector.
Singularities are most likely to occur when the robot's arms are fully extended ($q_2 = 0$), especially when the robot switches from an "elbow up" to an "elbow down" configuration.

In process tasks, such as laser applications, passing through singularities can result in significant issues, including unintended movements, loss of control, and disruptions in planned trajectories. Therefore, it is essential for the robot to avoid crossing singularities in these situations.

Consider the scenario where the shape is positioned as shown in the figure 7 below. In the upper part of the figure, the robot has an ”elbow up” configuration, while in the lower part, the configuration is ”elbow down.” However, this is not an ideal configuration for the robot because it would require traversing a singularity to change its configuration,.

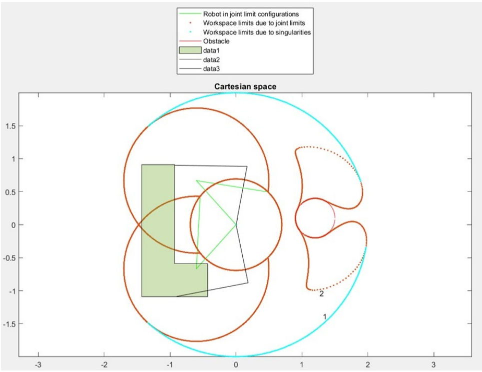

Figure 7: Configuration to avoid

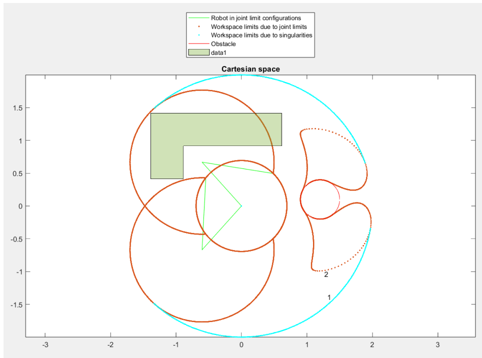

Figure 8: Workspace boundaries for process task

For link lengths, the workspace obtained in the previous exercise is suitable.
By analyzing the joint space and considering the placement of the L-shape, the previously employed configuration is suitable, as it prevents the robot from crossing singularities while following the shape’s contours, as it can be seen in the figure 8.

5 Conclusion

In conclusion, this report has presented an in-depth analysis of the workspace and singularity zones of a SCARA robot. We have demonstrated how to design a MATLAB function to plot the robot’s workspace, taking into account joint limits and obstacles. Additionally, we have explored the robot’s capabilities for point-to-point tasks, pick-and-place operations, and process tasks, addressing the challenges associated with singularities.

Understanding the workspace and singularity zones of a robot is crucial for task planning, trajectory generation, and ensuring the safe and efficient operation of robotic systems in various applications. By considering the specific requirements and constraints of each task, as well as optimizing link lengths, we can enhance the robot’s performance and reach a wide range of positions in its workspace.