Autonomous and Mobile Robotics Notes

Jose Del Valle

October 8, 2024

Contents

Chapter 1

Configuration space

A robot from a mechanical point of view can be seen as a system of N rigid bodies that move in a workspace $W = \mathbb{R}^2$ or $W = \mathbb{R}^3$. The workspace W is the physical space in which the robot operates and can move, it's essentially the geometric space in which the robot operates and can move. If the workspace is $\mathbb{R}^2$, the robot is called planar, if the workspace is $\mathbb{R}^2$ the robot is called spatial. This definition of a robot is broad enough to allow a group of robots to be defined as a robot.

Definition 1.1. *A configuration* $\boldsymbol{q}$ *is the minimal set of values that fully describe the position and orientation (pose) of the robot (of every point of the robot) at a given time in the workspace.*

$$\boldsymbol{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix}$$

The components $q_i, \ i = 1 \ldots n$ of a configuration are called generalized components. The name generalized comes from the fact that the coordinates aren't limited to express only rotations or translations, in fact they can represent any type of motion. These coordinates generalize the description of the robot's motion, meaning they adapt to any kind of mechanical system, whether it's a simple 1D slider or a complex multi-jointed arm in 3D space.

> **Warning**
>
> The special orthonormal group $SO(3)$ describes the orientation of a body using a 3×3 rotation matrix. Although straightforward, this method is not minimal, as the matrix has 9 elements, with only 3 being independent due to orthonormality constraints, leading to redundancy.
>
> Euler angles offer a minimal alternative by using 3 parameters (roll, pitch, yaw) to represent the 3 degrees of freedom. However, they suffer from gimbal lock, a singularity where two rotational axes align, causing the loss of one degree of freedom.
>
> Quaternions, a four-dimensional extension of complex numbers, avoid gimbal lock and efficiently represent rotations. However, they are not minimal either, requiring 4 parameters and a normalization constraint, which introduces redundancy.
>
> While quaternions are more robust, Euler angles may still be preferred for their minimality, despite the risk of gimbal lock, due to their simpler and more compact representation.

Definition 1.2. *A configuration space* **C** *as the set containing all valid configurations.*

$$\mathbf{C}\{\boldsymbol{q} \in \mathbb{R} | \boldsymbol{q} \text{ is a valid configuration of the robot}\}$$

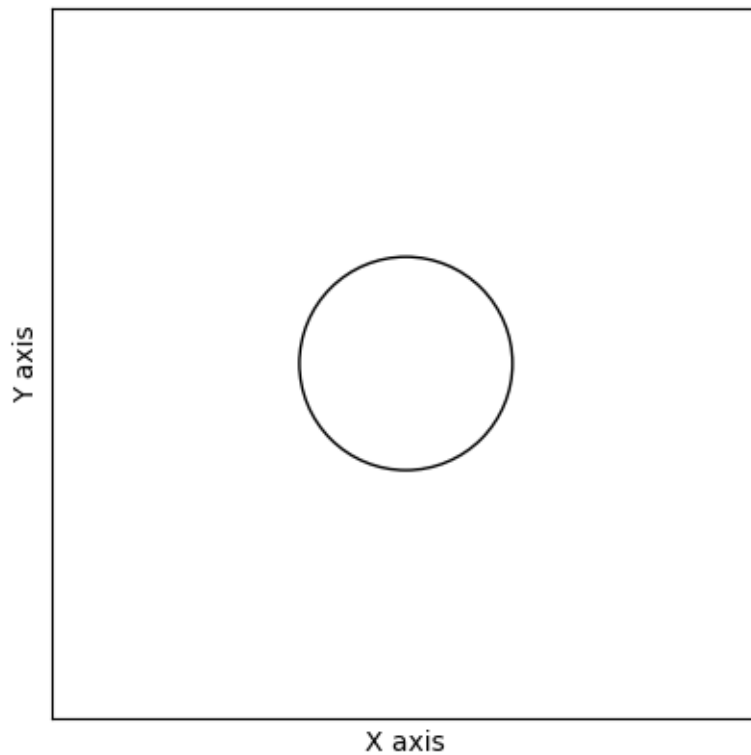**Example 1.1.** *Assume we have a robot with a negligible dimension w.r.t to to the workspace* $W = \mathbb{R}^3$

Punctiform robot
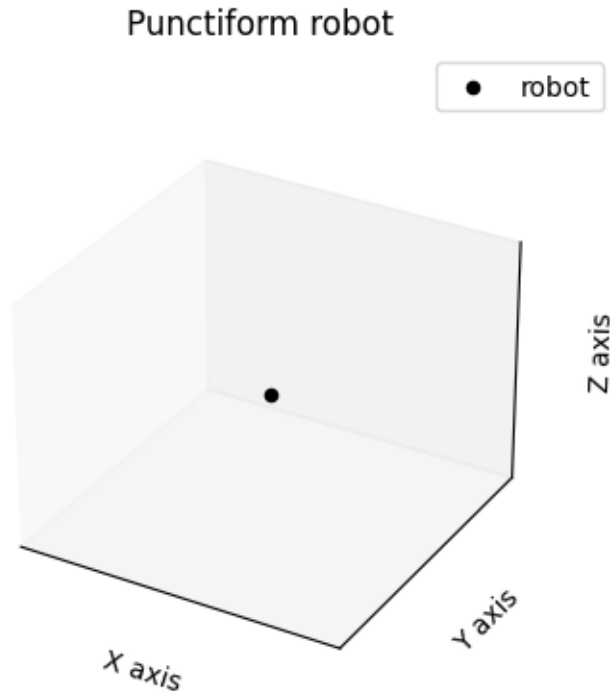
In this case the configuration vector is the following

$$\boldsymbol{q} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

> **Tip**
>
> In this case $\mathbf{C} = W = \mathbb{R}^3$, this however is not true in general.

Example 1.2. *Assume we have a circular symmetric robot in the plane, i.e. workspace $W = \mathbb{R}^2$.*
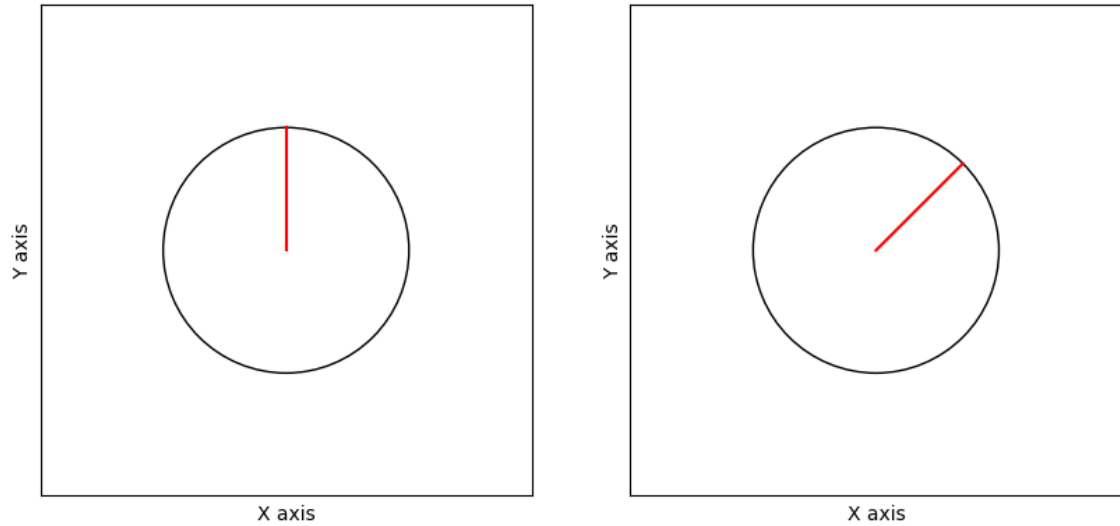

By exploiting it's symmetry, we can express the position of all of it's points w.r.t the center of mass of the robot

$$\boldsymbol{q} = \begin{bmatrix} x \\ y \end{bmatrix}$$

Example 1.3. *Assume we have a circular symmetric robot in the plane, i.e. workspace $W = \mathbb{R}^2$. Now assume that it has a tick on it.*

Now assume this two configurations:

If we where to use the previous configuration space we could not distinguish between this two configurations, because while previosly we didn't need to add rotational information, now it's required, thus the generalized vector is:

$$\boldsymbol{q} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

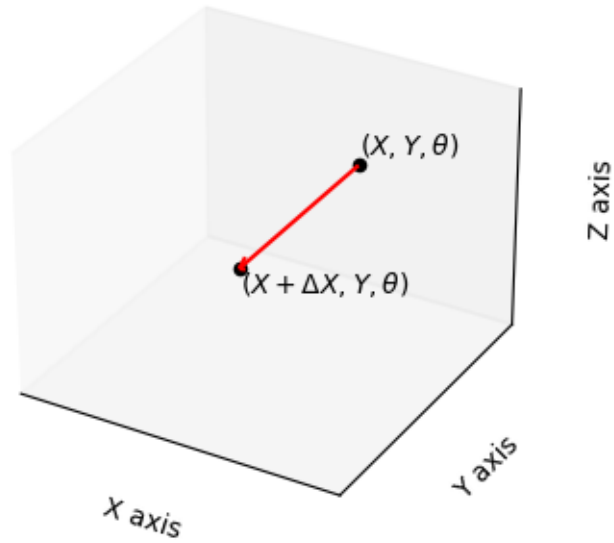In which θ is for example a rotation around the x-axis.

Tip

In this case $\mathbf{C} = \mathbb{R}^2 \times SO(2)$.

Warning

While the $SO(3)$ group is redundant, the $SO(3)$ is not, which means that $SO(2)$ is minimal and correctly used.

Now, we can represent, for example a positive movement along the x axis as

This is actually **not** a good representation of the configuration space, however it is convenient because it let's us compress our robot from a volume into a point. The need of the configuration space is infact given by the abstraction of the robots geometry.

Example 1.4. *Assume you have a planar manipulator with N joints.*
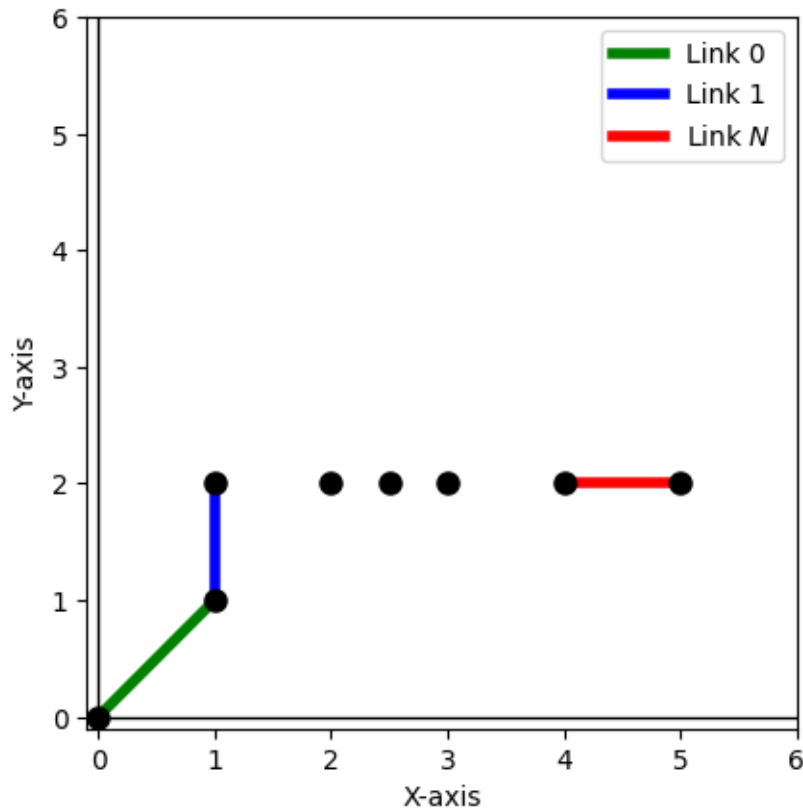
We could describe the configuration of this manupulator by taking as a position the baricenter of each link and thei inclination of each joint, this results in $3N$ parameters. This configuration would be great, however it is not minimal. A problem, is that assumes that there is no relationship between the links, however we know that the joint pose a constraint which make them dependant. Since each joint introduces 2 constraints, the number of free parameters is $3N - 2N = N$.

We can describe the N parameters as
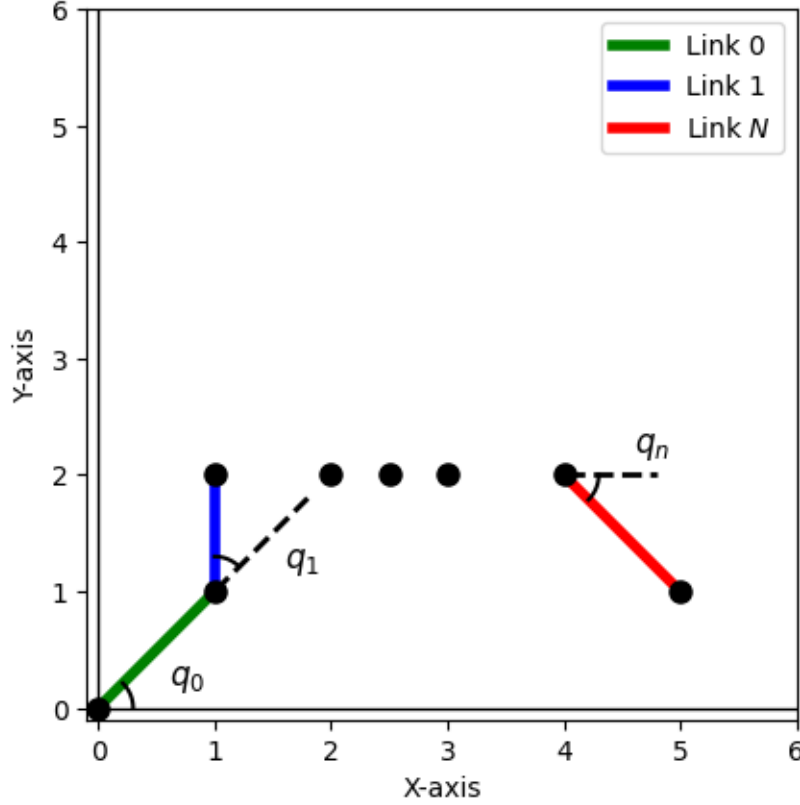
And thus our configuration $\boldsymbol{q}$, is just:

$$\boldsymbol{q} = \begin{pmatrix} q_0 \\ \vdots \\ q_n \end{pmatrix}$$

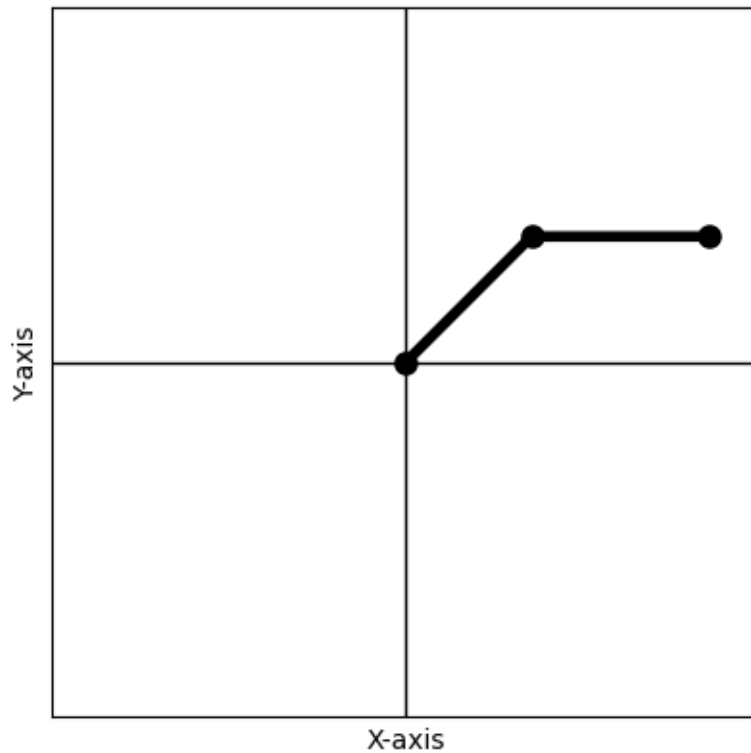*While the configuration space is $\mathbf{C} = SO(2)^N$.*

Now, if we assume we have a spatial manipulator, we have $6N$ parameters, of which $3N$ describe the position and $3N$ describe the rotation. For everything we said until now, this representation is not minimal, and due to the presence of joints there are $5N$ constraints, which means that we can treat the spatial manipulator as a planar manipulator.

Definition 1.3. *Topology is the study of the fundamental structure and properties of spaces that remain the same even when those spaces are continuously deformed. Topology ignores exact distances or angles and focuses instead on properties like connectedness or the number of holes in a space.*

In robotics, topology becomes crucial when we deal with the robot's configuration space The topology of C-space helps us understand whether the robot can smoothly move from one configuration to another without colliding with obstacles. It tells us about the overall shape and connectedness of C-space. Understanding the topology helps in tasks like: motion planning and obstacle avoidance.

Euclidean space is the regular, flat space we're used to, where distances, angles, and straight lines behave as expected. However, C-space is generally not Euclidean because it doesn't behave like a flat, regular space, in fact it can assume values in $SO(2)$, which is not closed under the sum. In non-Euclidean spaces, distances and paths behave differently.
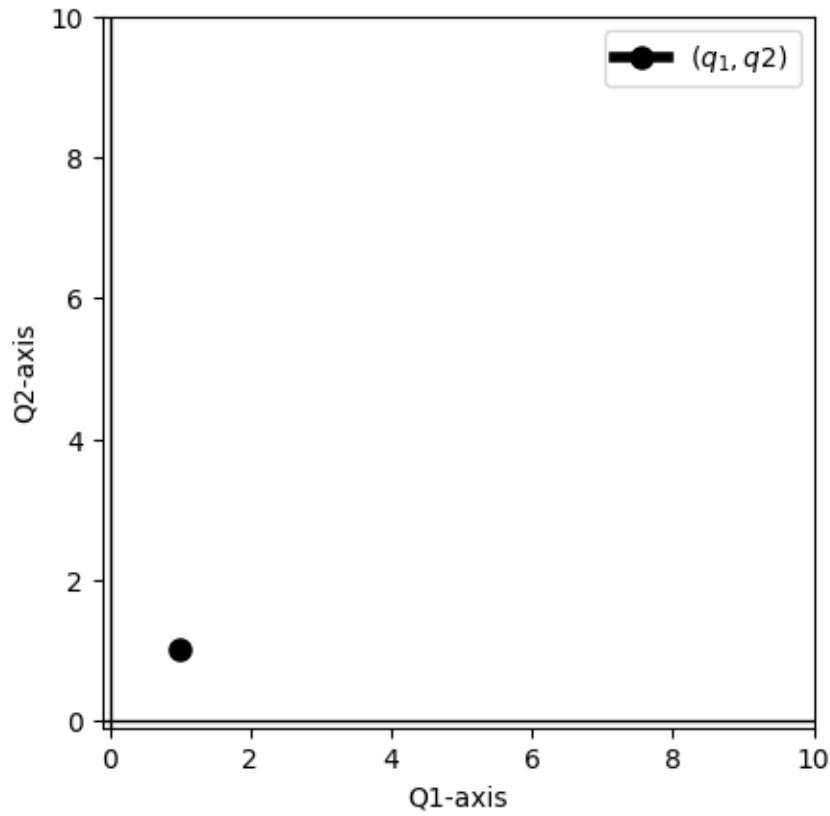
**Example 1.5.** *Assume you have a manipulator with two links*

Here we have:

$$\boldsymbol{q} = \begin{pmatrix} q_1 \\ q2 \end{pmatrix} \ , \ \mathbf{C} = SO(2) \times SO(2)$$

*Now a naive representation of the configuration space would be:*

The first thing, that we notice is that this configuration space is not injective. Assume for example that we make a full rotation, around the first axis, the robot will return to the same position, but the configuration will be different:
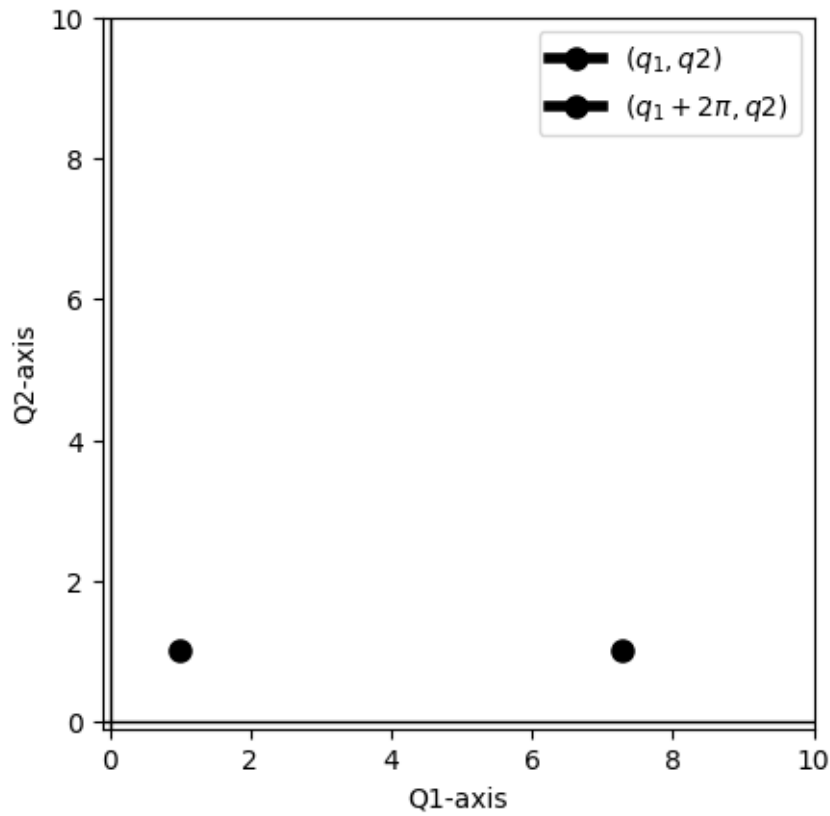
In general in fact there will be an infinite values of configurations that in the configuration space describe the same configuration in the working space.

Now, to solve this we can limit the admissibile values that both q_1 and q_2 can assume, such that $q_1, \ q_2 \ \in [0, 2\pi)$
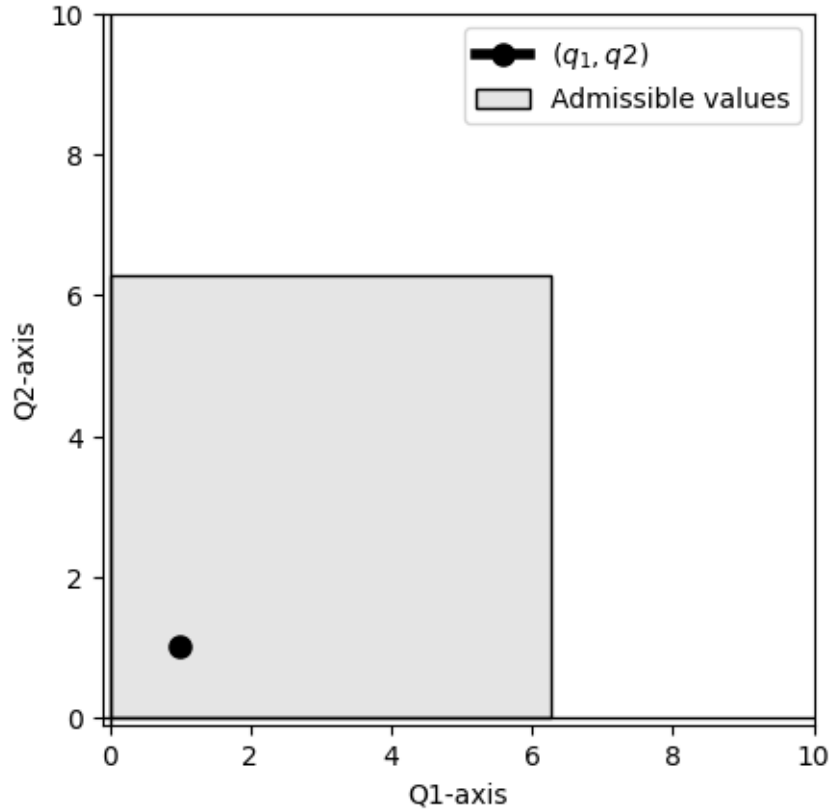
Now, the injectivity problem is solved, however a new problem appears: Assume you have two different configurations $q_A = (\epsilon, \pi)$ and $q_B = (2\pi - \epsilon, \pi)$
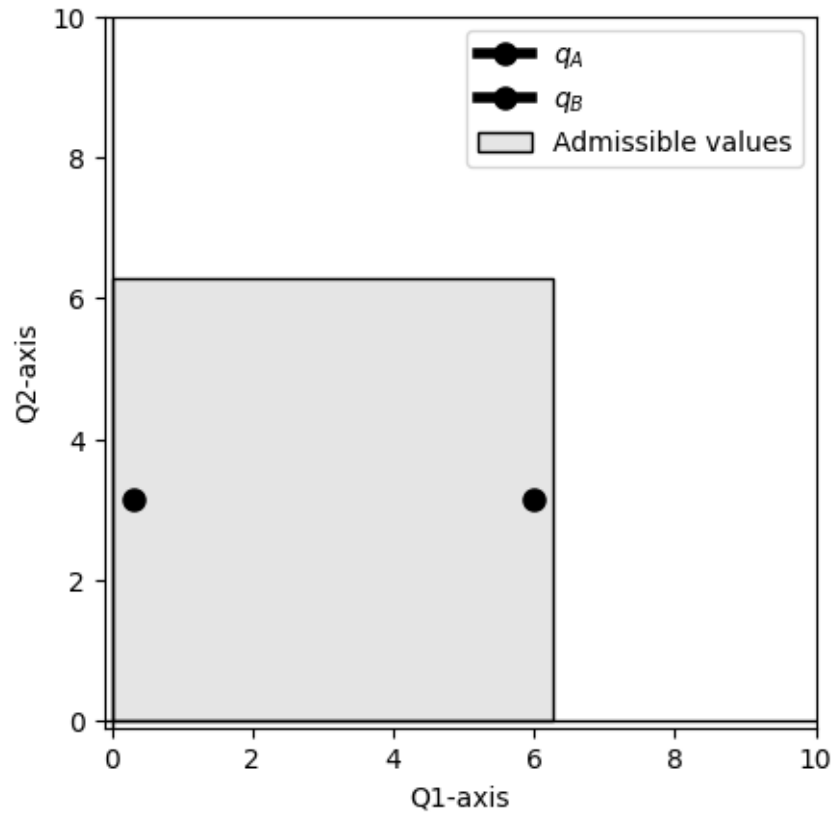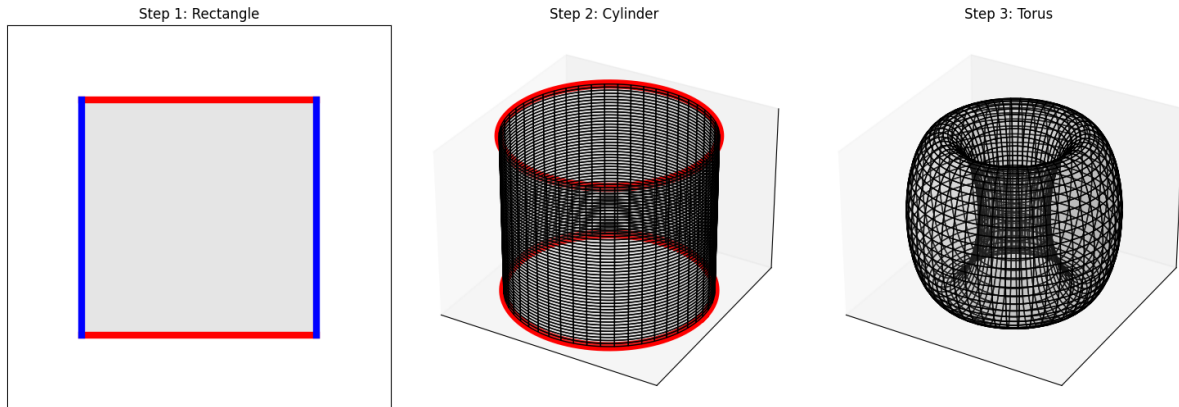
In the configuration space they seem to be distant, however in the workspace, they appear to be very similar, and thus the distance should be low:
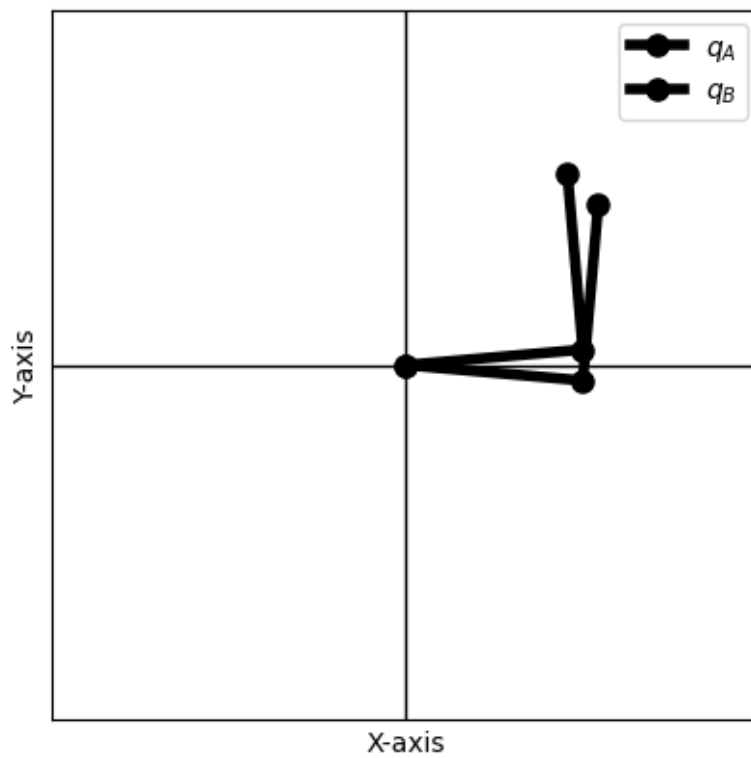
The problem is, as previously mentioned, that the configuration space is in general not Euclidean.

We can solve by transforming the admissible zone for the configuration values from a rectangle into a torus

The toroidal configuration space naturally handles the wrap-around behaviors, solving the distance problem. A torus is a manifold, which means that locally behaves like the Euclidean space. A practical example is the earth, is sperical but locally seems flat.