2360006- Algorithmic motion planning
Project 1

Submitters:
Paulo Khayat – 212747018
Salih Hassan – 212148894

2.1. **Assume that a link is modeled as a cylinder with a radius r, and length 10r and that all spheres have equal radii. Furthermore, assume for simplicity that the center of the link is located along the x-axis with one endpoint at (0,0,0) and the other at (10r,0,0).**

2.1.1. One sphere: We will pick a sphere at the center of the link, (5r,0,0). The radius of the sphere will be such that all the link will be inside it. If we draw the triangle between the center and the edges of the cylinder, we get a radius of $\sqrt{5^2 + 0.5^2} \cdot r \cong 5.025r$

2.1.2. Two spheres: Notice there is a symmetry in this problem, and that the rings at the bases of the cylinder need to be covered, as-well as the ring along the center of the cylinder. We will place the spheres at equal distance between these rings. The sphere centers will be at (2.5r,0,0), (7.5r, 0,0). They will have radii of $\sqrt{2.5^2 + 0.5^2} \cdot r \cong 2.54r$

2.1.3. Five spheres: Let's split the cylinder into 6 rings we need to cover. These rings are located at 0r,2r,4r,6r,8r and 10r. if we choose our sphere centers at ((1r,0,0),(3r,0,0)…(9r,0,0)) and give each sphere a radius of $\sqrt{1.25}r = 1.118r$, we will cover every point on the link.

Notice, our radius is larger than $\frac{10r}{2s}$ where s is the number of spheres. This is because if we were to choose this radius. We wouldn't be able to cover some points on the link (the corners/bases/ the above defined rings)

2.1.4. Ten spheres: As always, we pick our spheres evenly along the link, displaced from the bases, so ((0.5r,0,0),(1.5r,0,0)…(9.5r,0,0)), and we'll try to cover all the r,2r…,10r rings.

So, radius will be $\sqrt{0.5}r$

2.2. **Discuss in general terms the tradeoff and effect of the number of spheres and their radius.**

A low number of spheres means less computations, we are checking far less collisions. Remember we check all collisions of links with other links, as well as links with each obstacle. However, the drawback is the increase in the possibility of false positives.

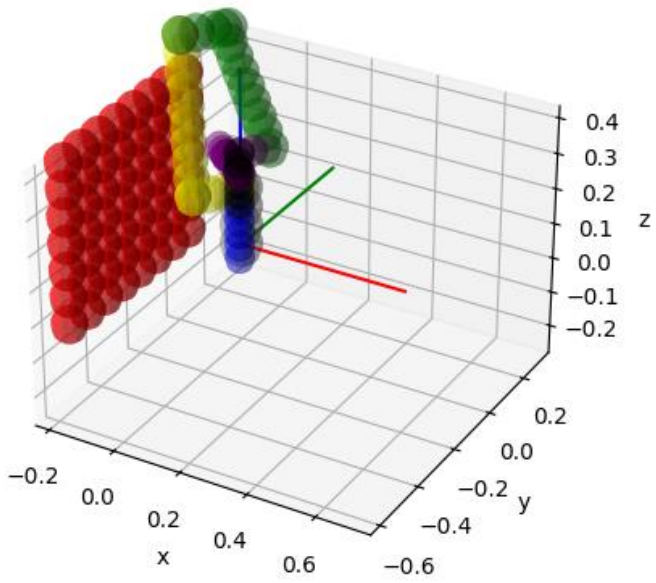
A higher number of spheres has a tighter boundary around the link.

A small number of spheres can be especially harmful in the case of checking self-collisions, where two close wrists might seem as permanently touching. (we can ignore collision between two neighboring links, but the problem is still prevalent with the existence of short links, or in configurations where the robot's wrist is folded in some way).

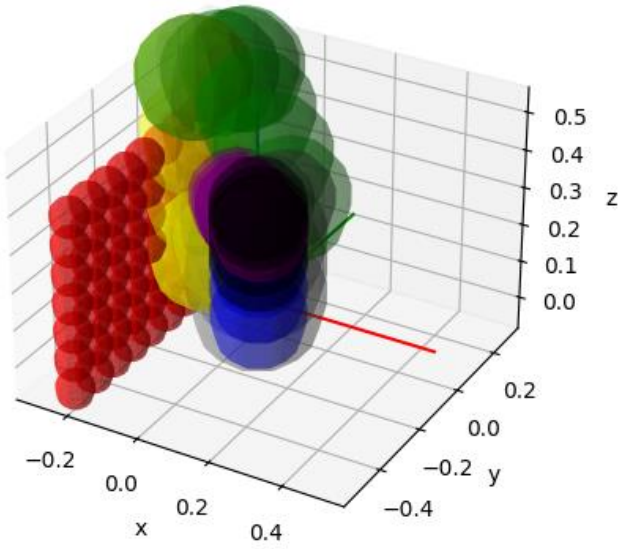
2.3. **The following snapshots depict the results from the given:**

$$\boldsymbol{configuration} = [\mathbf{-0.694}, \mathbf{-1.376}, \mathbf{-2.212}, \quad \mathbf{-1.122}, \quad \mathbf{1.570}, \mathbf{-2.26}]$$

a. $Inflation_{factor} = 1$:

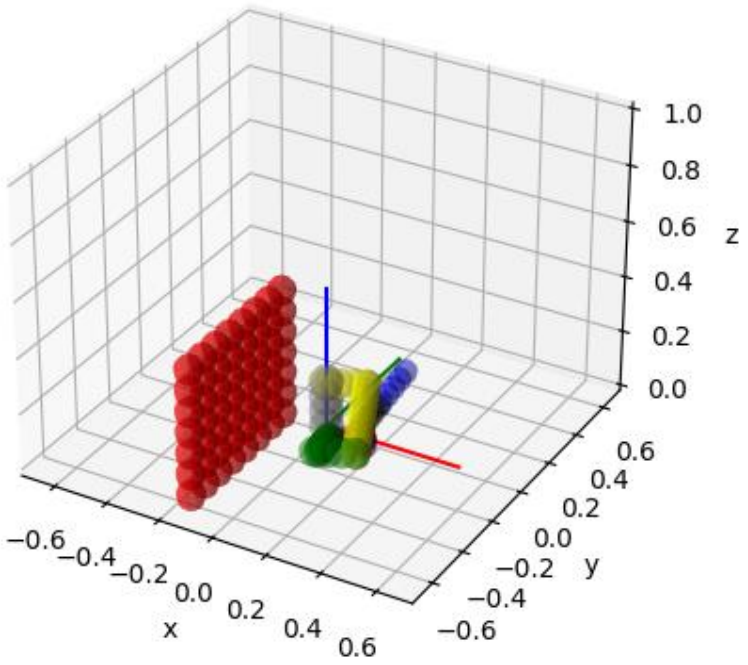


b. $Inflation_{factor} = 3$:

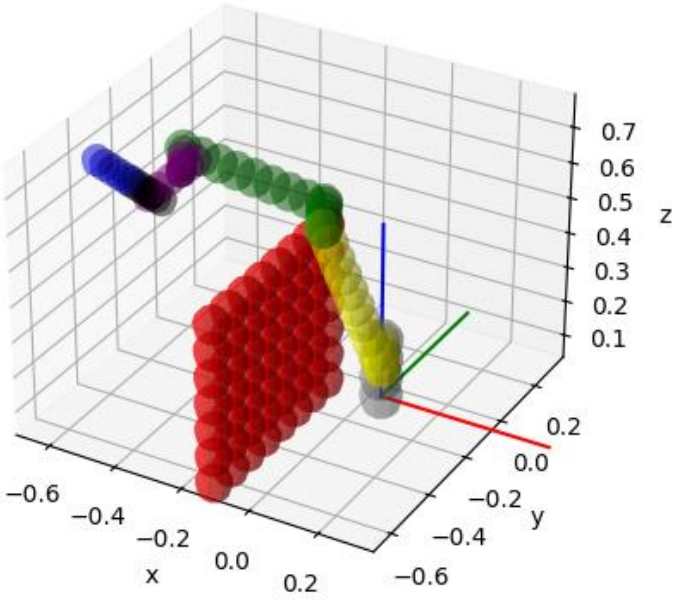


3.1 **after implementing the $is\_in\_collision()$ function, here are examples of configuration which are in collision / not in collision:**

a. an example of a configuration in collision:
the configuration is (in radians): [2.0, 0.0, 9.0, 1.0, 1.0, 3.0]

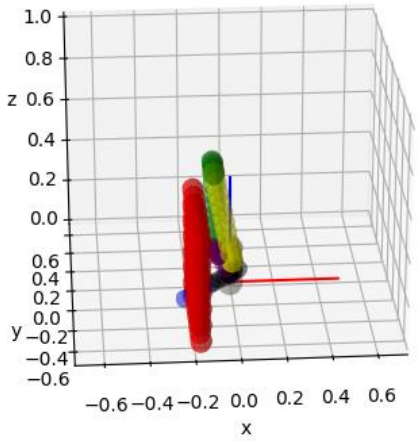


b. an example of a configuration which is not in collision:
the configuration is (in radians): [0.5, -1.2, 0.8, -0.5, 1.0, 0.7]

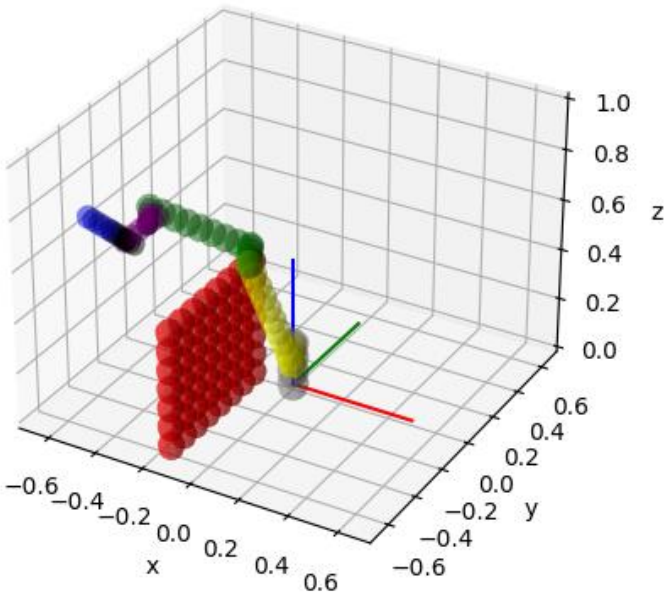


3.2 **after adding a check to the $is\_in\_collision()$ function that considers collisions with the floor, here are examples of configuration which are in collision / not in collision:**

a. an example of a configuration in collision with an obstacle:
[0, -1.376, 3, -1.122, 1.570, -2.26]



b. an example of a configuration which is not in collision:
[0.5, -1.2, 0.8, -0.5, 1.0, 0.7]



4. **Change the parameter $self.resolution$ such that the local planner returns (1) True, (2) False. Use the following configurations:**
   (1) $conf1$: $[80, -72, 101, -120, -90, -10]$ $[deg]$,
   (2) $conf2$: $[20, -90, 90, -90, -90, -10][deg]$.

   a. An example where the local planner returns True:
      When the number of configurations to check was: 2 the local planner returned: $True$.
   b. An example where the local planner returns False:
      When the number of configurations to check was: 3 and the local planner returned: $False$.

5. **For each sample in $random\_samples\_100k.npy$ compute if it's in collision using different inflation factors, and plot the overall computation time as a function of the radius used and the number of FN as a function used in range of inflation factor $[\,1.0, 1.8\,]$ in intervals of 0.1:**
   Here's the plot we received: