CS545 Robotics HW3: **Inverse Kinematics**

Stephen Iota
iota@usc.edu
SID: 6862013543
November 5, 2021

Problem 1.

(a) $\mathbf{q} = [0., 0., 0.],\ \mathbf{L} = [1., 1., 1.] \ \rightarrow\ fk(\mathbf{q}, \mathbf{L}) = [3., 0., 0.]$

(b) $\mathbf{q} = [0.3, 0.4, 0.8],\ \mathbf{L} = [0.8, 0.5, 1.] \ \rightarrow\ fk(\mathbf{q}, \mathbf{L}) = [1.9215064, 1.14848143, 0.]$

(c) $\mathbf{q} = [1., 0., 0.],\ \mathbf{L} = [3., 1., 1.] \ \rightarrow\ fk(\mathbf{q}, \mathbf{L}) = [3.62090692, 2.52441295, 0.]$

Problem 2. See figure 1.

Problem 3. Implemented `line_sphere_intersection` in **collision.py**.

Problem 4. See figure 2.

Problem 5. For larger values of r, solutions may not exist. If the obstacle is too big to reach around, the robot is unable to reach its end effector to the goal state. However, this can be mitigated by choosing smarter values of initial posiiton. Keeping in mind the robot joint bounds, for some starting states the robot does not need to reach around and can take a shorted path. For more discussion, see figure 3.

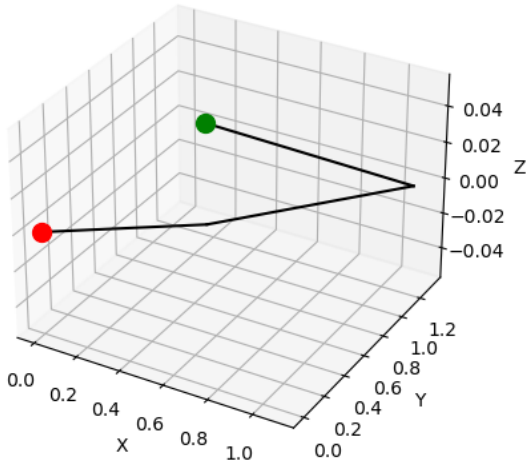

Figure 1: Solution state computed through inverse kinematics. The base of the robot is denoted by red, and the goal end-effector position in green. The solution state results in the robot arm touching the green goal state.

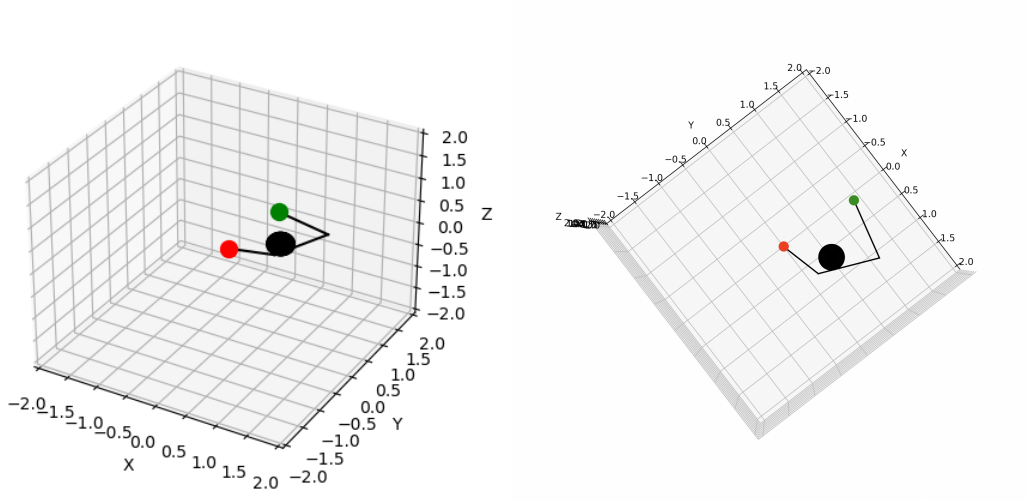

Figure 2: Solution state with an obstacle (black blob), using the collision detection algorithm. In the solution state, the robot's arms reach around the obstacle to reach the goal position. The robot's second arm rests along the outside of the obstacle, as seen in the view from the top.

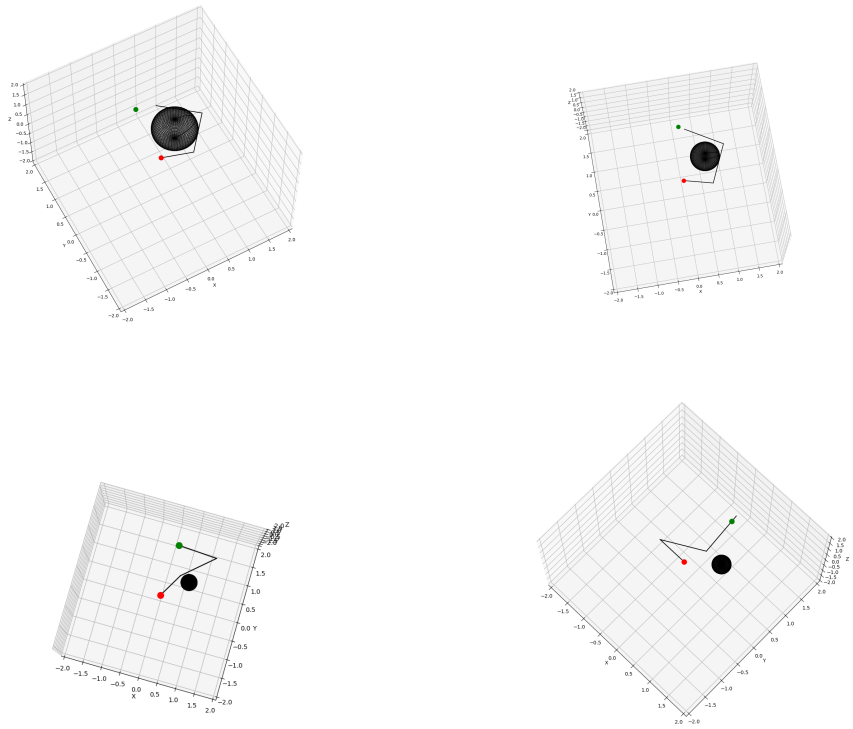

Figure 3: Top two images show different (larger) r values. In these cases, there is no solution as the robot's arms are not long enough to reach around the obstacle. Bottom two figures correspond to different starting states. Noteably in the bottomleft image, due to the initial state, the robot's arms are limited by their bounds and can only overshoot the goal position.