HW4 M0

`check_collision.m` checks the distance between each obstacle and each point in a discrete set of linearly distributed points along each robot link. If any of those distances is less than the **`link_radius`** plus the relevant **`sphere_radii[i]`**, then the robot is considered to be in collision. **`check_edge.m`** checks collisions along a linear path through configuration space by running **`check_collision.m`** on a discrete set of linearly distributed points along the path.

A possible problem with this method is that low resolutions may lead to sections of the robot or sections of the path not being sufficiently checked for collisions. If the resolution of **`check_collision.m`** is too low, two consecutive spheres along a robot link may not capture the entire cylindrical geometry of the link, and thus may not realize the robot is in collision. If the resolution of **`check_edge.m`** is too low, then two consecutive points along the path may not be in collision, but an unchecked point in between then may be in collision.

An additional issue is that a few of the joints of the robot feature perpendicular extrusions from the robot links. This may just be a quirk of the simulation, but if those are real physical features, then checking collisions with the radius of the link would fail to consider these large extrusions.

Also, high resolutions in combination with a high number of obstacles would result in a relatively costly algorithm.