1 First

So once we derived the main tool to find the prioritized solution we still need to find out why priority is such an important element in developing high level controllers.

- Priority allows us to decompose complex problems.
- We can implement easier controller for each task.
- Collision avoidance can be implemented separately.

2 Second

To perform collision avoidance, we used few control points which are nothing but points on the structure of the manipulator that are used to keep track of its distance from the obstacles. Whenever this distance goes down below a certain threshold we will push the point away until we reach a safe zone.

3 Third

In this simulation we wanted the manipulator to execute 4 different task which are:

- A cartesian positioning task, so we want the end effector to take place in the 3D space. And it will saturate 3 degrees of freedom.
- the number of degrees of freedom is the number of variables that can varies independently from each other, here it is the number of joints
- The second task is an orientation task, we want the third axis to keep being vertical. It is manly used to simulate any kind of auxiliary task occupying 1 DOF.
- Third and fourth tasks are associated with collision avoidance and saturates 1 degree of freedom each

So we need 6 degrees of freedom when the manipulator has 7 joints which means 7 degrees of freedom

4 Fourth

As already said the first task is a Cartesian positioning one. In this particular application we wanted it to follow a linear path and later on, a point to point path.

The Jacobian for this task is the analytical Jacobian of the direct kinematics.

To perform collision avoidance for this task we should execute a repulsive velocity at the end effector level but that would mean use 3 more DOF when we only have 1 left. So we used the said repulsive velocity as a feed-forward term to add to the commanded one.

How to compute this repulsive velocity?

We described a certain amount of constraints defining the profile of its magnitude, while its direction will be along the distance from the obstacle.

Those constraints include:

- Maximum speed at 0 distance from an obstacle.
- Speed tending to zero at the threshold distance from the obstacle.
- Smooth descending curve

All of this generates the given formula.

5 Fifth

For the link orientation task we needed first to define the task. Thinking the third link axis as a vector, we transformed it into spherical coordinates and took its elevation. Our control objective is to keep it to zero.

The jacobian of this task is easily obtained from the shown formulas.

6 Sixth

Third and fourth task are associated with the movement of the control points. Here the control points has been positioned respectively on the end-effector, on the origin of the DH frame associated with the 4^{th} joint and finally the third is on the mid point of the second link axis.

Collision avoidance for the first task has already been implemented, and for these two task we would like to do the same thing we did on the first task but we will be facing the same problem ass before. So to adapt the solution to the dimension of these tasks, we projected the repulsive velocity on the distance of the control point from the obstacle. So the jacobian associated with these tasks is the analytical one projected on the same vector.

7 Seventh

Prioritized solution requires an implementation of the ordered tasks. Here we stacked all the tasks in a vector where the first one will have the lowest priority, which means it is the most important.

This vector is divided into two part: the first contains the *critical* tasks, the ones which are in a dangerous region and the ones whose execution is mandatory. The second part contains all the tasks in a safe region and the ones whose execution is auxiliary.

In order to evaluate which are the critical tasks, we need a cost function. Since almost every task is associated with the movement of a control point, we used the minimum distance from the obstacles and we accommodated it for the second task, assigning to it a constant cost greater than the threshold.

To perform the reordering we will do it first for the safe part of the stack, so as to compute also which tasks has to move to the critical part. The we can move to reordering also the critical sub vector.