Task priority assignment with collision avoidance.

Stefano De Filippis & Marco Menchetti

October 2019

Abstract

In this paper we will face the problem of task priority resolution using a fast computation of the priority matrix (here *Flacco Matrix*) and the resulting joint velocities.

Collision avoidance for several control points has been taken as a high priority task in this case as well as trajectory tracking.

Introduction

Due to their high dexterity and the absence of non-holonomic constraints, manipulators has been used to perform a wide range of operation, and sometimes even more of them at the same time.

A handy yet practical example is the one considered below: a manipulator moving in a cluttered environment, trying to complete a trajectory tracking task and, at the same time, avoid collision with obstacles nearby.

Approach

Due to the high complexity of the task we divided our control scheme into 3 main blocks:

1. **Task priority matrix:** to compute in a fast way the resulting joint velocity, executing the task velocities coming from the prioritized stack of tasks.

2. **Priority resolution algorithm:** to organize the stack of tasks depending on the each ones' *generalized cost*. This concept will be further explained above.

3. **Control algorithm:** to generate the reference joint velocity that the manipulator should execute.

1 Task definition and associated cost

TODO: brief description

1.1 Assigned tasks

1. Trajectory tracking + Collision avoidance 0.

2. Elbow up (i.e. horizontal).

3. collision avoidance 1.

4. collision avoidance 2.

1.1.1 Collision avoidance

1.2 Cost definition

TTTTHE DISTANCE: TODO.

2 Task priority matrix

3 Priority resolution algorithm

We defined the stack of tasks as: TODO

4 Control algorithm

We switch every "n" so as: TODO

5 Code

6 Results

Contents