HANDS-ON INTERVENTION:
Vehicle-Manipulator Systems

Lab 3: Task-Priority kinematic control (1A)

Patryk Cieślak
patryk.cieslak@udg.edu

Exercise 1 (3 points)

The goal of the exercise is to visualise the null space motions of a planar 3-link manipulator, with 3 revolute joints. Following the developments of Lab 2, the robot will be controlled using the resolved-rate motion control algorithm and the null space will be studied with the use of the null space projector.
A screen capture of an exemplary simulation output is presented in Fig. 1.

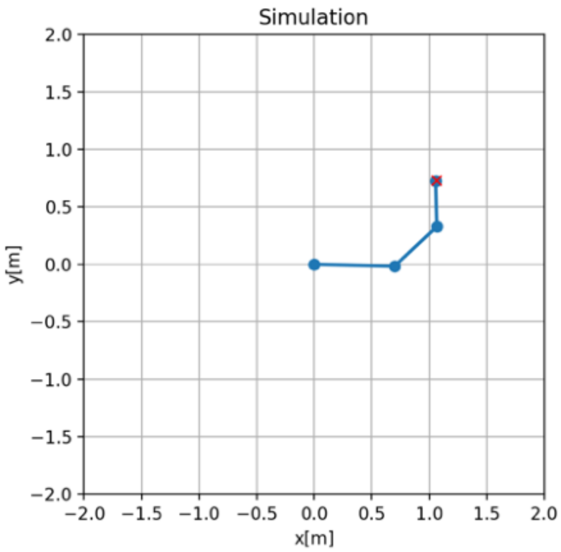

Fig 1. Simulation of the manipulator, including end-effector goal.

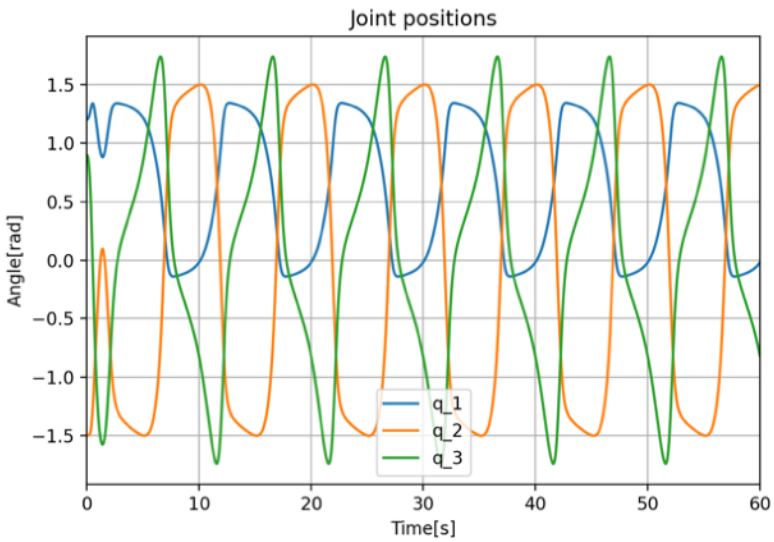

Fig 2. Positions of robot's joints during simulation.

The following elements have to be included in the simulation program:
1) Import of necessary libraries.
2) Definition of Denavit-Hartenberg parameters of a 3-link planar manipulator and its initial state.
3) Definition of parameters of the simulation.
4) Implementation of the resolved-rate motion control.
5) Implementation of the null space projector computation.
6) Implementation of an arbitrary function of time to realise null space motions (function returning the joint velocities based on simulation time).
7) Visualisation of the robot structure using the animation functionality of *Matplotlib*.
8) Visualisation of end-effector target on the plane.
9) Second, separate plot, displayed after the simulation is finished, presenting evolution of joint positions over time.

10) All plots with titles, labeled axes, proper axis limits, grid.
11) Simulation of 60s, without repeats.

The following elements have to be included in the report:
1) Drawing of the robot model, including DH parameters and coordinate systems.
2) Code of the simulation program (well formatted and commented in detail).
3) Plot presenting the visualisation of the robot structure in motion (see Fig. 1).
4) Plot presenting the evolution of robot's joints positions over time (see Fig. 2).

Exercise 2 (5 points)

The goal of this exercise is to implement the Task-Priority control algorithm for a hierarchy of two tasks (using the analytic solution), to control the manipulator simulated in Exercise 1. The main tasks of this exercise include definition of task Jacobians and errors and implementation of the control loop.

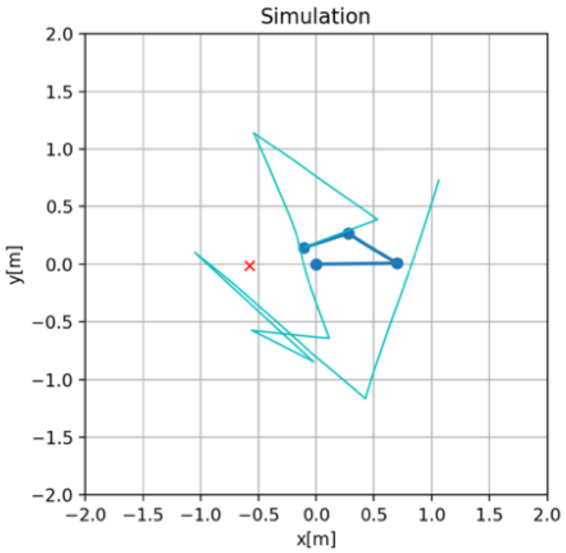

Fig 3. Simulation of the manipulator, including end-effector path.

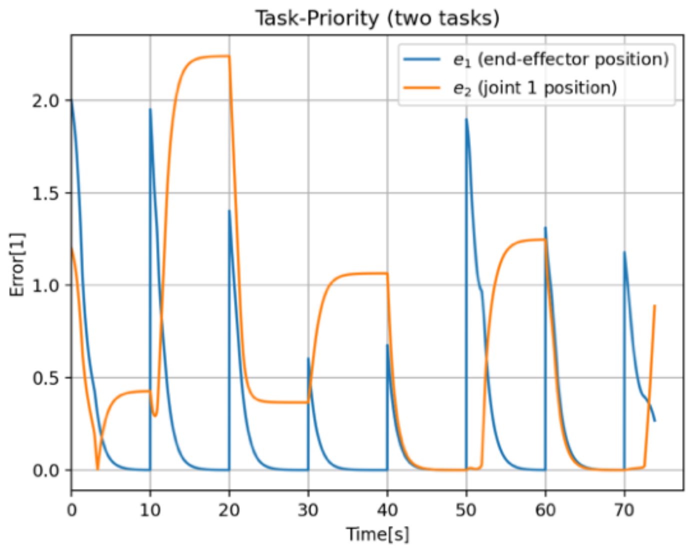

Fig 4. Evolution of the TP control errors.

The following elements have to be included in the program:
1) Import of necessary libraries.
2) Definition of Denavit-Hartenberg parameters of a planar 3-link manipulator and its initial state.
3) Definition of parameters of the simulation.
4) Definition of two tasks: position of the end-effector and position of the first joint (variables, error vectors and Jacobians).
5) Implementation of the Task-Priority algorithm based on analytical solution for two tasks.
6) Visualisation of the robot structure using the animation functionality of *Matplotlib*.
7) Visualisation of end-effector path on the plane.
8) Second, separate plot, displayed after the simulation is finished, presenting evolution of norms of tasks' error vectors over time.
9) All plots with titles, labeled axes, proper axis limits, grid.
10) Simulation of 10s, with repeats.
11) In each repeat the desired position of the end-effector set randomly.
12) The output velocity of the TP algorithm scaled to not exceed the maximum joint velocity limit.

The following elements have to be included in the report:

1) Code of the program (well formatted and commented in detail).
2) Plot presenting the visualisation of the robot structure in motion (see Fig. 3).
3) Plot presenting the evolution of the norm of control errors related to both defined tasks over time (see Fig. 4).
4) Plots from points 2) and 3) presented for two cases:
   a. End-effector position task at the top of the hierarchy.
   b. Joint position task at the top of the hierarchy.
5) Comments/observations about the functioning of the TP algorithm based on the prepared plots.

Questions (2 points)

1. What are the advantages and disadvantages of redundant robotic systems?
2. What is the meaning and practical use of a weighting matrix *W,* that can be introduced in the pseudo inverse/DLS implementation?