FIELD AND SERVICE ROBOTICS (FSR) – a.y. 2024/2025

University of Naples Federico II
Department of Electrical Engineering and Information Technology

Instructor: Prof. Fabio Ruggiero, Ph.D. [fabio.ruggiero@unina.it]
Assistant: Simone D'Angelo, Ph.D. [simone.dangelo@unina.it]
Assistant: Riccardo Aliotta, M.Sc. [riccardo.aliotta@unina.it]

[Updated 07/04/2025]

HOMEWORK n. 2

1. Implement via software the path planning algorithm for a unicycle based on a cubic Cartesian polynomial. Plan a path leading the robot from the configuration $q_i = [x_i \quad y_i \quad \theta_i]^T = [0 \quad 0 \quad 0]^T$, to a random configuration $q_f = [x_f \quad y_f \quad \theta_f]^T$ generated automatically by the code such that $\|q_f - q_i\| = 1$. Then, determine a timing law over the path to satisfy the following velocity bounds $|v(t)| \leq 0.5$ m/s and $|\omega(t)| \leq 2$ rad/s. [*Hint: For the final configuration, use the command rand(1,3): save the result in a vector and divide it by its norm.*]
2. Given the trajectory in the previous point, implement via software an input/output linearization control approach to control the unicycle's position. Adjust the trajectory accordingly to fit the desired coordinates of the reference point B along the sagittal axis, whose distance to the wheel's center it is up to you. Critically comment throughout the report how the results are affected by changing the distance of B from the unicycle's reference point. [*Hint: Consider 3-4 cases for the comparison.]*
3. Consider the Bernoulli's leminscate trajectory
$$x_d(t) = \frac{r\cos((\alpha+1)t)}{1+\sin^2((\alpha+1)t)},$$
$$y_d(t) = \frac{r\cos((\alpha+1)t)\sin((\alpha+1)t)}{1+\sin^2((\alpha+1)t)},$$
where α the last digit of your matriculation number, $t \in [0, \frac{4\pi}{\alpha+1}]$, and consider two different cases with two arbitrary values for $r > 0$ that should differ of at least one order of magnitude. For both cases, design via software the linear and (almost) nonlinear controllers for a unicycle. Suppose that the unicycle at the time $t = 0$ starts from a random position generated by the code within 0.5 m from the desired initial one. Through relevant plots, critically compare the results in the report.
4. Implement via software the unicycle posture regulator based on polar coordinates, with the state feedback computed through the Runge-Kutta odometric localization method. Starting and final configurations are $q_i = [x_i \quad y_i \quad \theta_i]^T = [\alpha+1 \quad 1 \quad {}^{\pi}/_{4}]^T$, with α the last digit of your matriculation number, and $q_f = [x_f \quad y_f \quad \theta_f]^T = [0 \quad 0 \quad 0]^T$.

NOTE: It is worth recalling not to report theory in the report. **Put all the plots you think are the most important to understand the performance of the code you implemented, and critically comment on the results.** Attach the code with your submission in a ZIP file. If you overcome the submission limit on Moodle, you may link in the report a GitHub, Dropbox, or Google Drive link (make these links public, if possible, to avoid waiting for permission to download the files).
The software is left free. Matlab is anyway suggested to save time. Notice that the Simulink Robotics Toolbox already have the block with the unicycle kinematic model implemented (Robotics Toolbox -> Mobile Robot Algorithms).