PART 6: Simulation

This section is based on a MATLAB/Simulink environment available on GitHub:

`https://github.com/DavidWuthier/uas-31390`

Follow the installation instructions to get started. Note that the main model file is `uas_main.slx` and that it relies on variables in the base workspace that can be loaded by running the script `uas_parameters.m` first.

Exercise 6.1: Navigating a 2D maze

When the simulation environment is up-and-running, you should be able to see a drone flying through a maze from one end to the other. The waypoints are containted in the variable `route`, and the corresponding scaling and offset are 1 m and 0 m (respectively).

Task: Recompute `route` using the code from exercise 4.1 to navigate from points (0, 0, 1) to (3, 5, 1).
Report: Provide a X-Y plot of the achieved trajectory.

Exercise 6.2: Re-implementing the position controller

In the `quadcopter` block, there is a position controller cascaded on an attitude controller. These two controllers are based on `papers/lee2011control.pdf`. The inputs of the position controller are setpoint positions, velocities and accelerations, and the outputs are roll, pitch, yaw and thrust commands.

Task: Replace the position controller with your own implementation. You can leave out the setpoint velocities and accelerations.
Report: Explain your approach and show the responses corresponding to step inputs of magnitude 1 m, 3 m, and 9 m in the x direction.

Exercise 6.3: Re-implementing the attitude controller

The inputs of the attitude controller are setpoint roll, pitch, yaw and thrust, and the outputs are force and torques commands that are translated into motor velocities by the mixer.

Task: Replace the attitude controller with your own implementation.
Report: Explain your approach and show the responses corresponding to step inputs of magnitude 5°, 15°, and 30° m in the forward pitch direction.

Exercise 6.4: Aggressively navigating a 2D maze

In the purple area, there are some blocks that output positions, velocities and accelerations using the variables generated by the script `uas_trajectory.m`. This script is based on

`papers/mellinger2011minimum.pdf` and the API's documentation can be found here:

`https://se.mathworks.com/matlabcentral/fileexchange/74573-traj_gen-matlab`

Task: Connect the blocks from the purple area to the quadcopter, comment the unused blocks, replace the position controller with the default one in case you didn't implement velocity and acceleration setpoints, and fix `uas_trajectory.m` to generate a trajectory that allows the drone to fly from (0,0,1) to (9,9,1) in less than 5 s without touching any of the walls.
Report: describe your solution and show the plots of the time profile as well as the x-y plot of the trajectory.