QUADCOPTER SIMULATOR MANUAL
V 1.0

CARLO ALBERTO PASCUCCI

1. Introduction

In this ZIP archive all the Matlab files needed to experiment with the quadcopter simulator described in the deliverable D6-1-3 section 2 are included. Further details on the benchmark exercise, developed using this simulator, can be found on section 3 within the same deliverable.

Depending on user's needs, in using this simulator it is possible to customize any detail of the single UAV starting from physical properties, like weights and dimensions, to the inner loop stabilization MPC controller properties. Alternatively, one can use the preconfigured UAV models to perform formation flight control design and simulation.

In section 2 a list together with a short description of each file is given, while in section 3 general simulation guidelines are provided.

2. Files description and usage

The MATLAB simulation package contains the following files:

- **MPC_script.m**
  - It initializes all the model's parameters and generates the MPC controller for attitude stabilization and position control.
  - It is possible to change all quadcopter's parameters and modify the MPC controller design. A typical customization could be switching among position, angles or velocity control at the inner loop level.
- **QC_dyn.mdl**
  - It is the Simulink model of the nonlinear quadcopter's dynamics. It has four inputs (motors commands) and twelve outputs (quadcopter states).
- **QC_loop.mdl**
  - It is the Simulink model with the closed loop between the MPC controller and the UAV. User references can be set through the *User controls* block.
- **QC_MPCcl.mdl**
  - It is the Simulink library including atomic blocks for the quadrotor nonlinear dynamics *(QC_dyn)*, and four versions of the closed-loop system constituted by the MPC controller and the UAV. It is worth noting that them differ only at an input/output level. The first one *(Quadcopter MPC closed loop)* accepts references on all state variables and returns the full state measurement vector.

The second one *(Quadcopter MPC closed loop angles and pos)* takes in references on position (X, Y, Z) and gives out the attitude angles (pitch, roll and yaw) and the actual position. The third one *(Quadcopter MPC closed loop pos only)* is similar to the previous one except for the output vector, which is now only represented by the actual position in space. Finally the fourth one *(Quadcopter MPC closed loop pos energy)* returns the integral of the manipulated variables as an additional output to the actual position, while taking desired position as input reference.

- **slblocks.m**
  - If Simulink is opened with this file in the working directory, the QC_MPCcl.mdl library is automatically loaded into the Simulink library browser.
- **MPC_controller.mat**
  - This file contains a default inner loop MPC controller suitable for quadcopter's position control.
- **identifiedModel.mat**
  - This file loads in the workspace a linear third order model of the closed loop between the MPC controller and the quadrotor identified using function pem of the Identification Toolbox.
- **identifyQCmodel.m**
  - Running this script it is possible to perform the closed loop identification as described in the deliverable D6-1-3, Section 3.4.
  - Relevant parameters tunable by the user are: the quadcopter's initial state; a set of references for the trajectory; the maximum order of the identified model.
- **performanceIndex.m**
  - This is an utility file for the identified model's performance index evaluation. It is called by other scripts such as *identifyQCmodel.m*.
- **runQCsimulation.m**
  - Using this script it is possible to perform the leader-follower simulation scenario presented in the deliverable D6-1-3, section 2.2.
  - By setting the *defaults* structure it is possible to choose the outer loop controller (independent, LQR, centralized MPC, or decentralized MPC), save a simple animation of the trajectories followed by the UAVs, toggle between random or user defined initial condition, choose to perform or not a new model identification and set the followers offsets.
  - It is also possible to change leader's position references.
  - The Simulink model used within this script is *LeaderFollower2.mdl*.
  - If the user is not concerned about energy consumption the *LeaderFollower.mdl* model can be alternatively used. Note that now the last code cell must be commented out and the performance index will be the only output of this function.
- **plotTrajectories.m**
  - It is a plotting utility called by the *runQCsimulation* function to plot in the same 3D axis all references and trajectories.

- **plot_quadcopter.m**
  - It is another plotting utility called by the *plotTrajectories* function used while generating a video of the simulation.
  - It draws a couple of rectangles which represent the quadcopter.
- **LMIout_cvx.m**
  - This script performs a stabilizing output-feedback synthesis via LMI. A detailed description of this function can be found in the deliverable D6-1-3, section 3.4.2.
- **LeaderFollower.mdl**
  - This is the Simulink model for the leader-follower simulation. To model the leader and the two followers UAV, three *Quadcopter MPC closed loop angles and pos only* blocks from the *QC_MPCcl* library are used.
  - The controller subsystem allows the user to select which controller to use, but also to set leader's references and followers offsets.
- **LeaderFollower2.mdl**
  - This model is the same as the previous one, except for the substitution of the *Quadcopter MPC closed loop angles and pos only* blocks with *Quadcopter MPC closed loop angles and pos energy*, allowing the user to perform also energy consumption measurements.
- **run_all_controllers.m**
  - This script automates the benchmark exercise execution. It calls the *runQC-simulation.m* script selecting each time one of the four different controllers while returning the performance index and the energy consumption for each test.
- **run_test_all.m**
  - Major aim of this file is to test the effective performance of the toolset for simulation of swarms. In fact, each UAV model is accurate and its unstable dynamics are controlled via MPC. This might lead to considerably high simulation times, thus the interested user may test his/hers computer performance with the present script.
  - Running this file the *test_many* Simulink model will be launched performing a 20 seconds simulation with a swarm of 20 closed loop MPC stabilized quadcopters. Then the simulation time is displayed for user evaluations. Full state measurements are stored for each quadcopter.
  - *Quadcoper MPC closed loop* blocks from the *QC_MPCcl* library are used. By default initial conditions and references are the same for each quadcopter and fixed for all the duration of the simulation.
- **test_many.mdl**
  - It is the Simulink model called within the *run_test_all.m* script. UAVs can be added or deleted simply importing them from the *QC_MPCcl* library, or pasting and cutting the ones already included in the Simulink model.
- **test1.mdl**

  - Can be used as an alternative to the previous one, to test simulation's execution time involving only one quadcopter.
- **test2.mdl**
  - Another execution time test example that mixes a non linear MPC closed loop stabilized quadcopter model together with the linear identified model.
  - The identification can be performed by the *identifyQCmodel.m*. Alternatively the user can load the *identifiedModel.mat* if specifically interested to the third order model of the closed loop between the MPC controller and the quadrotor used also in the benchmark exercise.
- **test_energy.mdl**
  - Running this simulation it is possible to evaluate a single UAV energy consumption. A *Quadcoper MPC closed loop pos energy* block from the *QC_MPCcl* library is used.
  - By default the reference position is set to obtain a lower bound for the energy consumption of the UAV.
- **exp_sample.m**
  - This script is intended to help the user to generate an explicit controller for the outer loop controller (i.e. the supervisor). Results obtained with default parameters settings are shown in section 3.4.4 of the deliverable D6-1-3.

3. Simulation guidelines

In the deliverable D6-1-3 through sections 2 and 3.4 the benchmark exercise, which represents a comprehensive use of this simulator, is thoughtfully explained. However also simpler simulations can be performed by running *run_test_all.m,test_many.mdl, test1.mdl, test2.mdl, test_energy.mdl* files. All the files included in the archive are are highly customizable, hence them can be considered as templates that the user can exploit to suit his/hers needs. Just by loading the *.mat* files provided, and dragging and dropping in a Simulink blank model blocks from the *QC_MPCcl* library, many kind of simulations involving multiple agents can be built quickly, allowing the user to focus on higher level control design which is the aim of this simulator.