# APRICOT: Aerospace PRototypIng COntrol Toolbox. Dynamics, planning and control.

Andrea Ferrarelli[1], Danilo Caporale[2], Alessandro Settimi*[,2,3] and Lucia Pallottino[2]

[1]Master student, Vehicle Engineering, Università di Pisa, Largo Lucio Lazzarino 1, 56122, Pisa, Italy
[2]Centro di ricerca "E. Piaggio", Università di Pisa, Largo Lucio Lazzarino 1, 56122, Pisa, Italy
[3]Dept. of Advanced Robotics, Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova

**Abstract.** A novel MATLAB/Simulink based modeling and simulation environment for the design and rapid prototyping of state-of-the-art aircraft control systems is proposed. The toolbox, named APRICOT, is able to simulate the longitudinal and laterodirectional dynamics of an aircraft separately, as well as the complete 6 degrees of freedom dynamics. All details of the dynamics can be easily customized in the toolbox, some examples are shown in the paper. Moreover, different aircraft models can be easily integrated. The main goal of APRICOT is to provide a simulation environment to test and validate different control laws with different aircraft models. Hence, the proposed toolbox has applicability both for educational purposes and control rapid prototyping. With respect to similar software packages, APRICOT is customizable in all its aspects, and has been released as open source software. An interface with Flightgear allows for online visualization of the flight. Examples of control design with simulation experiments are reported and commented.

## 1 Introduction

Aircraft planning and control design requires a simulation environment that is highly configurable based on specific aircraft characteristics or mission objectives. Simulation environments are also necessary to design control systems and validate flight planning strategies. There are several solutions available as commercial or open source software packages. Among these, most are coded in programming languages as Java or C++ and focus on the simulation of specific vehicles as multirotor aircrafts or on generic aircrafts, see [1] and [2], while others are user interfaces for auto-pilot control systems, see [3] and [4]. In this sense, an easy to use tool for control design is missing. Other simulators are available with a commercial licence or closed source, see e.g., [5], [6], [7]. Our aim with APRICOT (Aerospace PRototypIng COntrol Toolbox) is to provide a simulator environment with a focus on control system design, which is multi-platform,

* Corresponding Author: Alessandro Settimi, alessandro.settimi@for.unipi.it.

highly customizable and open source[1]. For code, videos and details on APRICOT please refer to [8]. Moreover, being based on MATLAB/Simulink, it can be easily used for education purposes or by control system designer for rapid control prototyping in the industry.

The 6 degrees of freedom nonlinear aircraft dynamics can be simulated with a preferred degree of accuracy, meaning that aerodynamic coefficients, stability derivatives [9], actuators and sensor dynamics, disturbances in measurements, wind gusts, control limits and other characteristics can be enabled separately and with different models by simply changing related MATLAB functions or Simulink diagrams. Moreover, custom atmospheric models can be used, by default the International Standard Atmosphere model [10] is implemented. Moreover, a manual input interface has been implemented to perturb the aircraft on-line during the simulation through an external gamepad to validate the effectiveness of the implemented control laws in response to various disturbances.

The possibility to visualize the simulation in a detailed 3D environment is obtained thanks to an interface with the open-source flight simulator FlightGear [11], [12], [13], as shown in Fig. 1. Both the toolbox and the flight simulator support multiple platforms (MacOS, Linux, Windows). Also, different aircraft models can be easily loaded from various sources like XML files, DATCOM files and even custom formats can be easily supported.

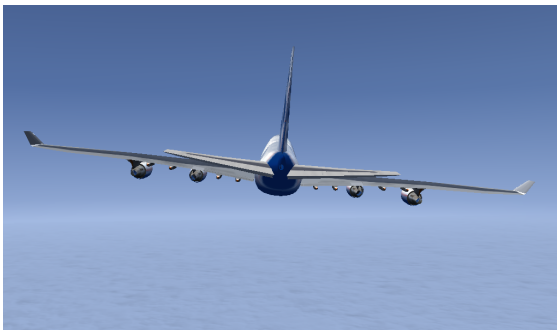


Fig. 1: APRICOT simulation animated in Flightgear.

The whole system is organized so that it is highly usable and reconfigurable, in that any detail of the simulation can be easily customized from the aircraft model to the control law used. In particular, the main features of APRICOT are the possibility to easily customize the geometric and physical characteristics of an aircraft for what concerns the model and the environmental disturbances. In particular, different unmanned aerial vehicle (UAV) models can be included and simulated in APRICOT such as quadrotors. As for the control design, control laws can be implemented with Simulink blocks and MATLAB functions; sev-

[1] APRICOT Software available at `http://aferrarelli.github.io/APRICOT/`

eral control laws are already provided in the toolbox. The APRICOT Simulink scheme is reported in Fig. 2.

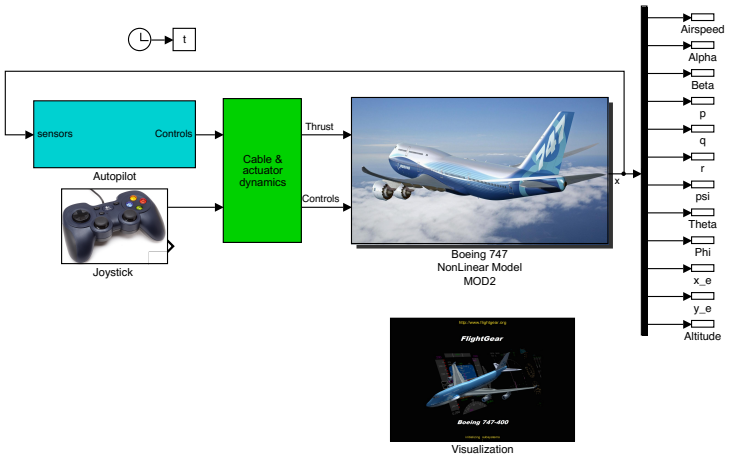



Fig. 2: APRICOT Simulink scheme

Another major difference with other simulation environments, where a control law is typically tested on the subsystem for which it has been designed (e.g. a longitudinal controller is tested on the longitudinal subsystem), APRICOT allows multiple controllers to run simultaneously on a subsystem or on the full system dynamics. In other words, with APRICOT a more realistic behavior can be reproduced and tested with separated and/or redundant controllers, providing the control designer with an easy assessment of robustness in presence of failures.

Other than robustness with respect to failure, control laws can be tested to track a given flight trajectory. An optimization based planning is used to steer the system between desired configurations while minimizing fuel consumption and actuation effort and verifying state constraints. A feed-forward control, based on the obtained trajectory, can be used jointly with a feedback controllers.

For both robustness and tracking purposed, various classical and modern control techniques, as the ones reported in [14], [15], [16], are available in APRICOT controlling both longitudinal and lateral dynamics. Linear controllers have been implemented in the toolbox to test the system even while working far from the operation point around which linear controllers have been designed. Moreover, within APRICOT the performance assessment can be conducted to analyze the whole system behaviour due to the full nonlinear dynamics of the model.

The APRICOT control laws are based on pole-placement, Linear Quadratic Regulator (LQR), Linear Quadratic Gaussian (LQG) and Linear Parameter-Varying (LPV) techniques and nonlinear Lyapunov based techniques (see, e.g.,

[17], [18] and [19] for detailed discussions on these methods and similar application examples).

To highlight the APRICOT toolbox characteristics and performance, several tests have been conducted with a Boeing 747 aircraft model and will be shown in Section 6. These experiments are available in the toolbox as demos and can be used as a guideline for control development and aircraft customization. The simulator has been tested mainly on MATLAB R2016a and FlightGear 3.4.0. Some parts of the simulator rely on the Simulink Aerospace Toolbox and the MATLAB Control System Toolbox.

## 2 Modeling and simulation environment

In this section we introduce, for convenience, the equations that describe the dynamics of an aircraft. These are classical models available in literature, see, e.g., [14], [15], and can be used as a starting point to customize any aspect of the simulation. The user interested in the planning and control design can focus on Sections 3 and following.

### 2.1 Dynamic equations

The aircraft model considered in this work is based on the full nonlinear 6 DoF dynamics.

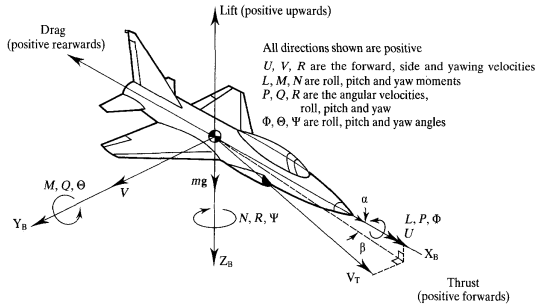



Fig. 3: Aircraft coordinated system, from [20].

The system has 6 dynamic states $x_d$ and 6 kinematic states $x_k$:

$$x_d = \begin{bmatrix} V & \alpha & \beta & p & q & r \end{bmatrix}^T \quad x_k = \begin{bmatrix} x_G & y_G & z_G & \phi & \theta & \psi \end{bmatrix}^T$$

where, referring to Fig. 3, $V = V_T$ is the airspeed, $\alpha$ is the angle of attack, $\beta$ is the sideslip angle. $p$, $q$, $r$ are roll pitch and yaw angular rates in body coordinates, respectively, $x_G$, $y_G$, $z_G$ are center of mass coordinates of the aircraft and $\phi$, $\theta$, $\psi$ are the Euler angles, in fixed frame.

The whole state vector is defined as $x^T := [x_d^T, x_k^T]$. The input vector is $u = [\delta_{th},\, \delta_e,\, \delta_a,\, \delta_r]^T$, namely the thrust, elevator, aileron and tail rudder commands. The dynamics of $x_d$ depends on the forces and moments acting on the system (*i.e.,* aerodynamic forces, engine thrust and gravitational forces) and is expressed in body coordinates:

$$\begin{cases} \dot{V} = \frac{1}{m}[-D\cos\beta + Y_A\sin\beta + X_T\cos\alpha\cos\beta - mg\,(\cos\alpha\cos\beta\sin\theta \\ \quad -\sin\beta\sin\phi\cos\theta - \sin\alpha\cos\beta\cos\phi\cos\theta)] \\ \dot{\alpha} = \frac{1}{mV\cos\beta}[-L - X_T\sin\alpha + mg\,(\cos\alpha\cos\phi\cos\theta + \sin\alpha\sin\theta)] \\ \quad + q - \tan\beta\,(p\cos\alpha + r\sin\alpha \\ \dot{\beta} = \frac{1}{mV}[\,D\sin\beta + Y_A\cos\beta - X_T\cos\alpha\sin\beta + mg\,(\cos\alpha\sin\beta\sin\theta + \\ \quad \cos\beta\sin\phi\cos\theta - \sin\alpha\sin\beta\cos\phi\cos\theta)] + p\sin\alpha - r\cos\alpha \\ \dot{p} = \frac{1}{I_{xx}}(\mathscr{L} - (I_{zz} - I_{yy})q\,r + (\dot{r} + p\,q)I_{xz}) \\ \dot{q} = \frac{1}{I_{yy}}(\mathscr{M} - (I_{xx} - I_{zz})r\,p - (p^2 - r^2)I_{zx}) \\ \dot{r} = \frac{1}{I_{zz}}(\mathscr{N} - (I_{yy} - I_{xx})p\,q + (r\,q - \dot{p})I_{zx}) \end{cases}$$

The dynamics of the kinematic states $x_k$, expressed in the body coordinates are

$$\begin{cases} \dot{x_G} = \cos\theta\cos\psi\,u + (-\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi)v \\ \quad +(\sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi)w \\ \dot{y_G} = \cos\theta\sin\psi\,u + (\cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi)v \\ \quad +(-\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi)w \\ \dot{z_G} = -\sin\theta\,u + \sin\phi\cos\theta\,v + \cos\phi\cos\theta\,w, \\ \dot{\phi} = p + (q\sin\phi + t\cos\phi)\tan\theta \\ \dot{\theta} = q\cos\phi - r\sin\phi \\ \dot{\psi} = (q\sin\phi + r\cos\phi)\sec\theta. \end{cases}$$

where $u, v, w$ are the linear velocity expressed in the body coordinates and depend on $V, \alpha, \beta$ as follows

$$\begin{cases} u = V\cos\alpha\cos\beta \\ v = V\sin\beta \\ w = V\sin\alpha\cos\beta. \end{cases}$$

All variables and coefficients in the equations above are explained in detail in the next sections.

### 2.2 Aerodynamic forces and moments

The aerodynamic drag $D$, lateral $Y_A$ and lift $L$ forces and moments (respectively $\mathscr{L}$, $\mathscr{M}$, $\mathscr{N}$) depend on the aircraft geometry, aerodynamic coefficients and dynamic pressure. They are given in wind axis coordinate frame by

$$\begin{cases} D = C_D \, q_{dyn} \, S \\ Y_A = C_Y \, q_{dyn} \, S \\ L = C_L \, q_{dyn} \, S \end{cases}$$

$$\begin{cases} \mathscr{L} = C_l \, q_{dyn} \, S \, b \\ \mathscr{M} = C_m \, q_{dyn} \, S \, \bar{c} \\ \mathscr{N} = C_n \, q_{dyn} \, S \, b \end{cases}$$

where the parameters $S$, $\bar{c}$ and $b$ are aircraft geometry characteristic, see Fig. 4, and are respectively the wing area, the mean chord of the wings and the wing length; $q_{dyn}$ is the dynamic pressure, $C_D$, $C_Y$, $C_L$ are drag, lateral and lift aerodynamic coefficients, $C_l$, $C_m$, $C_n$ are aerodynamic moment coefficients.

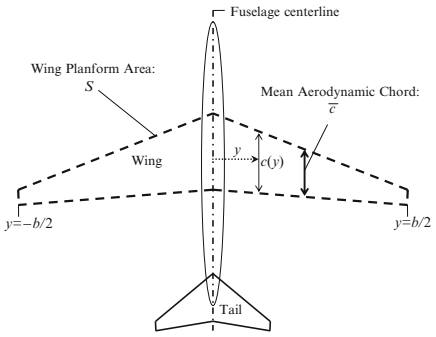



Fig. 4: Geometry of the aircraft, from [21].

The dynamic pressure is computed as:

$$q_{dyn} = \frac{1}{2} \rho V^2$$

The air density $\rho$ is altitude dependent. For this reason an atmospheric model is required, such as the ISA (International Atmospheric Model) [10] used in this work.

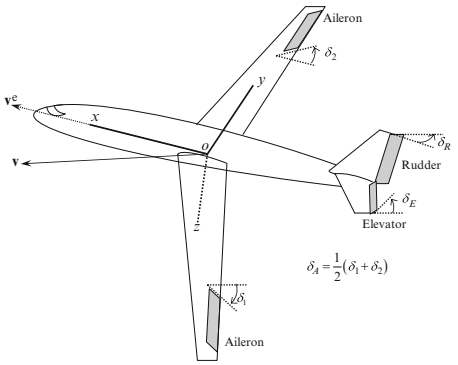



Fig. 5: Surface controls of the aircraft, from [21].

The aerodynamic coefficients depend on the system states and on the control surfaces (elevator, aileron, rudder) and can be computed around an equilibrium configuration, see Fig. 5 as described next.

*Longitudinal aerodynamic coefficients*

$$\begin{cases} C_D(\alpha, M) = C_D + C_{D_\alpha}\alpha + C_{D_M}\Delta M \\ C_L(\alpha, q, \dot{\alpha}, M, \delta_e) = C_L + C_{L_\alpha}\alpha + C_{L_q}\frac{q\bar{c}}{2V} + C_{L_{\dot{\alpha}}}\frac{\dot{\alpha}\bar{c}}{2V} + C_{L_M}\Delta M + C_{L_{\delta_e}}\delta_e \\ C_m(\alpha, q, \dot{\alpha}, M, \delta_e) = C_{m_\alpha}\alpha + C_{m_q}\frac{q\bar{c}}{2V} + C_{m_{\dot{\alpha}}}\frac{\dot{\alpha}\bar{c}}{2V} + C_{m_M}\Delta M + C_{m_{\delta_e}}\delta_e \end{cases}$$

where $\Delta M = (M - M_0)$, $M$ is the Mach number and $M_0$ is the corresponding trimmed value.

*Laterodirectional aerodynamic coefficients*

$$\begin{cases} C_Y(\beta, \delta_r) = C_{Y_\beta}\beta + C_{Y_{\delta_r}}\delta_r \\ C_l(\beta, p, r, \delta_a, \delta_r) = C_{l_\beta}\beta + C_{l_p}\frac{p\,b}{2V} + C_{l_r}\frac{r\,b}{2V} + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_r}}\delta_r \\ C_n(\beta, p, r, \delta_a, \delta_r) = C_{n_\beta}\beta + C_{n_p}\frac{p\,b}{2V} + C_{n_r}\frac{r\,b}{2V} + C_{n_{\delta_a}}\delta_a + C_{n_{\delta_r}}\delta_r \end{cases}$$

The aerodynamic coefficients derivatives can be obtained from the literature for different equilibrium configurations, as shown in Fig. 6. They can be loaded or stored in DATCOM or XML files.

### 2.3 Other forces and actuator dynamics

The engine response to the thrust lever is modeled for control purposes as a first order system with a transfer function:

$$G(s) = \frac{\tau}{s + \tau}$$

| Condition | 2 | 5 | 7 | 9 | 10 |
|---|---|---|---|---|---|
| $h$ (ft) | SL | 20,000 | 20,000 | 40,000 | 40,000 |
| $\mathbf{M}_\infty$ | 0.25 | 0.500 | 0.800 | 0.800 | 0.900 |
| $\alpha$ (degrees) | 5.70 | 6.80 | 0.0 | 4.60 | 2.40 |
| $W$ (lbf) | 564,032. | 636,636. | 636,636. | 636,636. | 636,636. |
| $I_y$ (slug-ft$^2$) | $32.3 \times 10^6$ | $33.1 \times 10^6$ | $33.1 \times 10^6$ | $33.1 \times 10^6$ | $33.1 \times 10^6$ |
| $\mathbf{C}_L$ | 1.11 | 0.680 | 0.266 | 0.660 | 0.521 |
| $\mathbf{C}_D$ | 0.102 | 0.0393 | 0.0174 | 0.0415 | 0.0415 |
| $\mathbf{C}_{L_\alpha}$ | 5.70 | 4.67 | 4.24 | 4.92 | 5.57 |
| $\mathbf{C}_{D\alpha}$ | 0.66 | 0.366 | 0.084 | 0.425 | 0.527 |
| $\mathbf{C}_{m_\alpha}$ | -1.26 | -1.146 | -.629 | -1.033 | -1.613 |
| $\mathbf{C}_{L_{\dot\alpha}}$ | 6.7 | 6.53 | 5.99 | 5.91 | 5.53 |
| $\mathbf{C}_{m_{\dot\alpha}}$ | -3.2 | -3.35 | -5.40 | -6.41 | -8.82 |
| $\mathbf{C}_{L_q}$ | 5.40 | 5.13 | 5.01 | 6.00 | 6.94 |
| $\mathbf{C}_{m_q}$ | -20.8 | -20.7 | -20.5 | -24.0 | -25.1 |
| $\mathbf{C}_{L_\mathbf{M}}$ | 0.0 | -.0875 | 0.105 | 0.205 | -.278 |
| $\mathbf{C}_{D_\mathbf{M}}$ | 0.0 | 0.0 | 0.008 | 0.0275 | 0.242 |
| $\mathbf{C}_{m_\mathbf{M}}$ | 0.0 | 0.121 | -.116 | 0.166 | -.114 |
| $\mathbf{C}_{L_{\delta_e}}$ | 0.338 | 0.356 | 0.270 | 0.367 | 0.300 |
| $\mathbf{C}_{m_{\delta_e}}$ | -1.34 | -1.43 | -1.06 | -1.45 | -1.20 |

Fig. 6: Longitudinal aerodynamic coefficients, from [22].

with $\tau > 0$. More detailed models can be easily implemented based on user needs.

The peak thrust and the Specific Fuel Consumption (SFC) of the engine are not constant but depend on altitude and Mach number as shown in Fig. 7a. Considering the engine curves and the equilibrium point, 3D engine maps are generated as in Fig. 7b and 7c via least squares polynomial fitting.

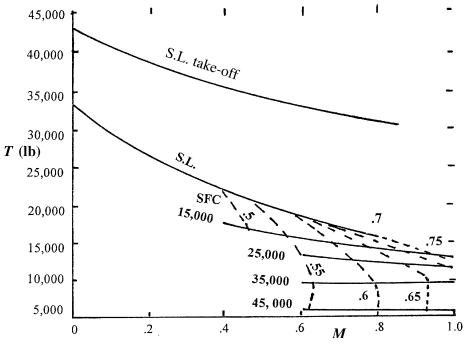
In the model are also included the actuator dynamics, rise limit and saturation of the control surfaces.

Finally the forces acting on the center of mass of the aircraft, expressed in body coordinates, are:

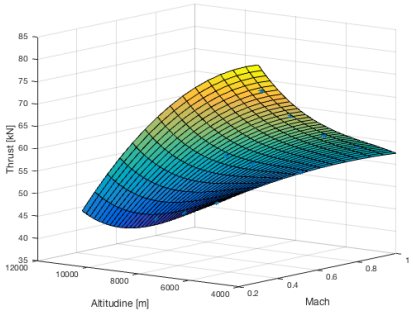
$$\begin{cases} X = -\cos\alpha\, D + \sin\alpha\, L - mg\sin\theta + X_T \\ Y = Y_A + mg\cos\theta\sin\phi \\ Z = -\sin\alpha\, D - \cos\alpha\, L + mg\cos\theta\cos\phi \end{cases}$$

where $X_T$ is the engine thrust.

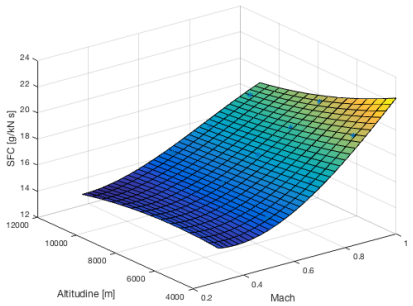
The forces and moments acting on the aircraft are computed within a Matlab Function and sent to the 6 DOF Simulink block that contains the equations of the system, see Fig. 8.




(a) Engine Thrust and SFC data.




(b) Thrust engine map.




(c) SFC engine map.

Fig. 7: Engine characteristics.

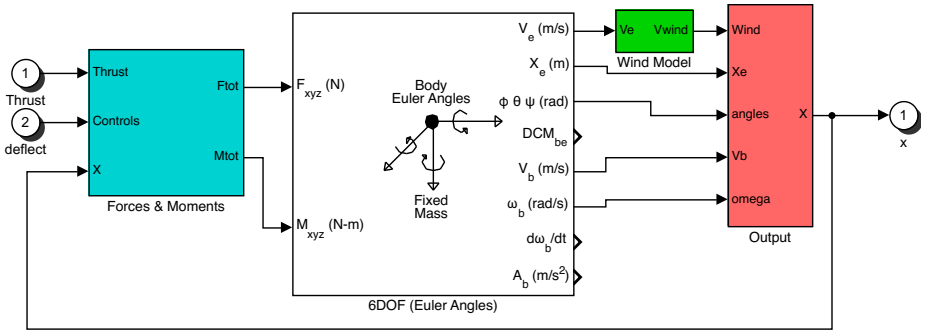



Fig. 8: Simulink model of a Boeing 747.

### 2.4 Sensor Noise and wind gusts

Noise models for sensors can be customized. By default, all sensors are affected with a zero mean gaussian noise with a variance that depends on the output type, therefore on the sensor type.

For example in Table 1 we show the default variance values considered for different outputs.

| **Output** | $\sigma^2$ |
|---|---|
| $V$ | $10^{-2}\,\mathrm{m/s}$ |
| $\theta$ | $10^{-5}\,\mathrm{rad}$ |
| $h$ | $0.1\,\mathrm{m}$ |

Table 1: Noise variance for longitudinal system.

Wind gusts are modeled via Simulink blocks. By default the model given beside Fig. 9, whose evolution is reported therein, is used.

$$V_{wind} = \begin{cases} 0 & x < 0 \\ \frac{V_m}{2}(1 - \cos(\frac{\pi x}{d_m})) & 0 \leq x \leq d_m \\ V_m & x > d_m \end{cases}$$

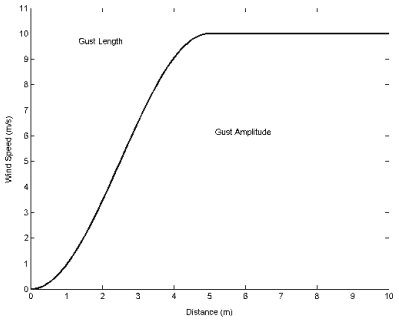



Fig. 9: Wind velocity with respect to the travelled distance.

### 2.5 Customization of the aircraft

All elements of the simulator shown in the previous sections can be customized by the user with a preferred degree of accuracy. First of all, different aircraft models can be loaded from DATCOM or XML files. MAT files can also be used to load different actuator, noise parameters and engine characteristics. Finally, wind and atmospheric models can also be loaded. Matlab functions and Simulink schemes can be implemented to replace or modify the default behavior of any part of the simulator, exception made for the basic rigid-body equations.

Some key parameters can be controlled directly with the control GUI, as shown in the next section.

## 3 Control interface

The user can choose to interact with the aircraft control system by means of a GUI (see Fig. 10) where default or user defined controllers can be loaded either on the full dynamics model, the reduced longitudinal or lateral dynamics models. User can also choose to activate different inputs and outputs for a given controller, enabling in this way fail-safe tests in different conditions of actuation or sensing. The same applies for disturbances, whose entity or models can be varied to assess the system performances in various conditions. This is particularly useful when designing a control law. By default the user can choose between the full nonlinear dynamics to be controlled, or between the longitudinal and laterodirectional subsystems. The longitudinal subsystem state vector is taken as $x_{LO} = [V, \alpha, q, \theta, h]^T$, with inputs $u_{LO} = [\delta_{th}, \delta_e]^T$. The laterodirectional subsystem state vector is taken as $x_{LD} = [r, \beta, p, \phi, \psi, y]^T$, with inputs $u_{LO} = [\delta_a, \delta_r]^T$.

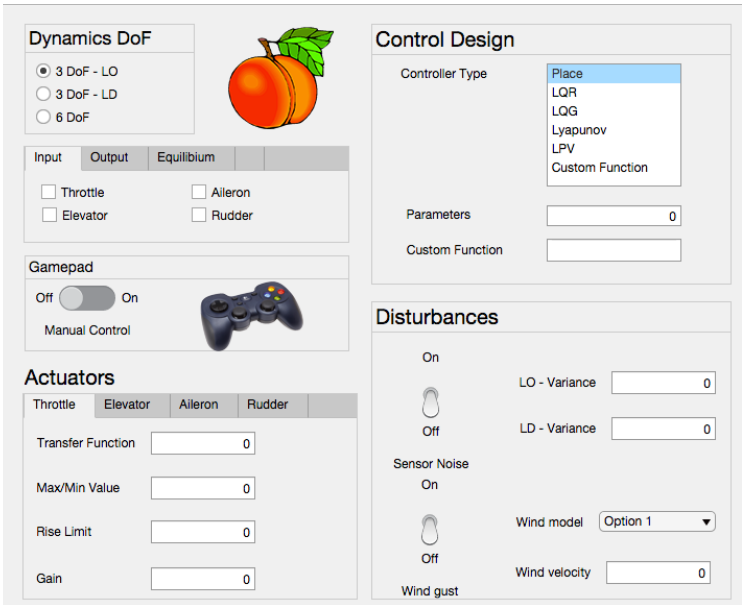



Fig. 10: The APRICOT Control GUI.

# 4 Control design

In order to show the capabilities of APRICOT, we summarize here different control laws commonly used in aerospace applications that have been implemented in the toolbox. We based our control design on the common assumption of decoupled longitudinal and lateral dynamics. In the next section we will illustrate the performances of such controllers even with the full 6 DoF dynamics.

## 4.1 Linear control design

Based on the decoupled linear systems of the flight dynamics, the design of the feedback and feed-forward controllers are done separately. In the design of the controllers the dynamics of the actuators is not considered but we take into account the saturation limits of the control surfaces and the engine thrust limit in LQR and LQG controls; the maximum thrust considered for the design of controllers is computed at the trim condition. Both the longitudinal and the laterodirectional linear system have not unstable eigenvalues but contain one or more poles at the origin, the systems are therefore not asymptotically stable. For the longitudinal system we considered as inputs the engine thrust $\delta_{th}$ and the elevator $\delta_e$, while the outputs are the wind speed $V$, the angle $\theta$ and the altitude $h$.

For the laterodirectional system the inputs considered are the aileron angles $\delta_a$ and the tail rudder angles $\delta_r$, while the outputs taken into account are angular rates $r$ and $p$, Euler angles $\phi$ and $\psi$, and the lateral position $y$.

Both the systems with this multi-input and multi-output (MIMO) configuration are fully observable, hence system observers have been implemented to estimate the system state.

Pole placement, LQR and LQG controls have been implemented in this setup and results are illustrated in Section 6. For the design of the control law, the maximum values of states and inputs considered are reported in Table 2. An

| **Variable** | **Maximum value** |
|---|---|
| $v_{max}$ | 0.5 m/s |
| $\alpha_{max}$ | 20° |
| $q_{max}$ | 20°/s |
| $\theta_{max}$ | 20° |
| $h_{max}$ | 1 m |
| $\delta_{th,max}$ | 0.1 |
| $\delta_{de,max}$ | 0.1 |

Table 2: Maximum values considered in the control design.

LQG controller has been implemented to improve the performance in presence of sensor noise and model disturbances, as outlined in section 2.4. It is easy to modify the disturbance models to simulate other conditions than those in Table 1. However, the user can readily experiment different control laws based on these methods with the available scripts.

### 4.2 Linear Parameter-Varying design

In this section we show an example of control design by means of LPV method. Tipically, LPV methods are applied in aircraft control to cope with variations of the angle of attack $\alpha$ and airspeed $V$. Here we consider a simpler setting where the longitudinal dynamics is assumed to be controlled, *i.e.,* with one of the previously illustrated designs. As a consequence, the Euler angle $\theta$ can be used as an exogenous input in the laterodirectional (LD) dynamics. Hence, a model in the following form can be obtained:

$$\dot{x}_{LD} = A(\theta)x_{LD} + B(\theta)u_{LD}$$

In this case, multiple LQR controllers can be easily designed for different values of $\theta$. Simulation results obtained with such controller are illustrated in Section 6.

### 4.3 Nonlinear control design

For nonlinear systems as the one considered in this paper nonlinear control laws are more effective when the system evolves far from the equilibrium configuration around which the linear controls are defined. A possible design of the control law is the Lyapunov based approach. Considering the longitudinal dynamics (LO) subsystem with state vector $x = [V \quad \alpha \quad q \quad \theta]^T$, its dynamics can be expressed in a control affine form as

$$\begin{cases} \dot{x} = f_{LO}(x) + g_{LO}(x) \begin{bmatrix} \delta_e \\ \delta_{th} \end{bmatrix} \\ y = x, \end{cases}$$

where $f_{LO}$ is the drift term of the dynamic function, while $g_{LO}$ is the control vector field. Based on the following Lyapunov candidate

$$V(x) = k_1 \frac{(x_1 - v_0)^2}{2} + k_2 \frac{x_2^2}{2} + k_3 \frac{x_3^2}{2} + k_4 \frac{x_4^2}{2},$$

with $k_i > 0, i = 1 \dots 4$ and where $v_0$ is the desired equilibrium velocity, we implemented different control laws, for example based on the Artstein-Sontag formula. Two of such control laws are

$$u(x) = \begin{cases} 0 & \text{if } |V(x)| < \epsilon \\ -\frac{LfV}{LgV} - \frac{\sqrt{LfV^2 + LgV^4}}{tol} & \text{if } |LgV(x)| < \text{tol} \\ -\frac{LfV}{LgV} - \frac{\sqrt{LfV^2 + LgV^4}}{LgV} & \text{otherwise} \end{cases}$$

and

$$u(x) = \begin{cases} 0 & \text{if } |V(x)| < \epsilon \\ -\frac{LfV}{LgV} - \frac{\alpha V^2}{tol} & \text{if } |LgV(x)| < \text{tol} \\ -\frac{LfV}{LgV} - \frac{\alpha V^2}{LgV} & \text{otherwise.} \end{cases}$$

Note that tolerances $\epsilon$ and $tol$ have been included to take into account practical implementation of the control algorithm. These can be easily shown to globally stabilize the system in a neighborhood of the origin.

## 5 Flight planning

Flight planning is a critical task as the aircraft is subject to constraints in actuation, energy consumption and compliance with air traffic control specifications, in order to avoid midair collisions. Besides, control authority should be minimized whenever possible to extend the life of actuation surfaces and components. It is then essential to consider these aspects in order to generate reference trajectories to be used as feed-forward control inputs, and to validate the generated plans via simulation.

In APRICOT, this is achieved through optimization-based techniques. It is possible to consider constraints on the system dynamics, states and inputs, or to obtain plans for unconstrained problems.

As discussed in Section 2.3, the fuel consumption of the aircraft is described through SFC and thrust engine maps. The fuel flow is computed as the product of SFC times the thrust $X_T$ of the engines

$$\dot{m}_{fuel} = SFC\ X_T.$$

A simplifying assumption in the planning setup can be made considering a constant SFC equal to its trimmed value, while the Thrust $X_T$ is taken as an optimization variable. It is worth noting that in APRICOT the complete computation of the fuel flow is used, see Fig. 7, and can also be customized.

The MATLAB function implementing the flight planning described above, takes as input the following parameters:

- initial conditions,
- final conditions,
- initial time,
- final time,
- discretization period.

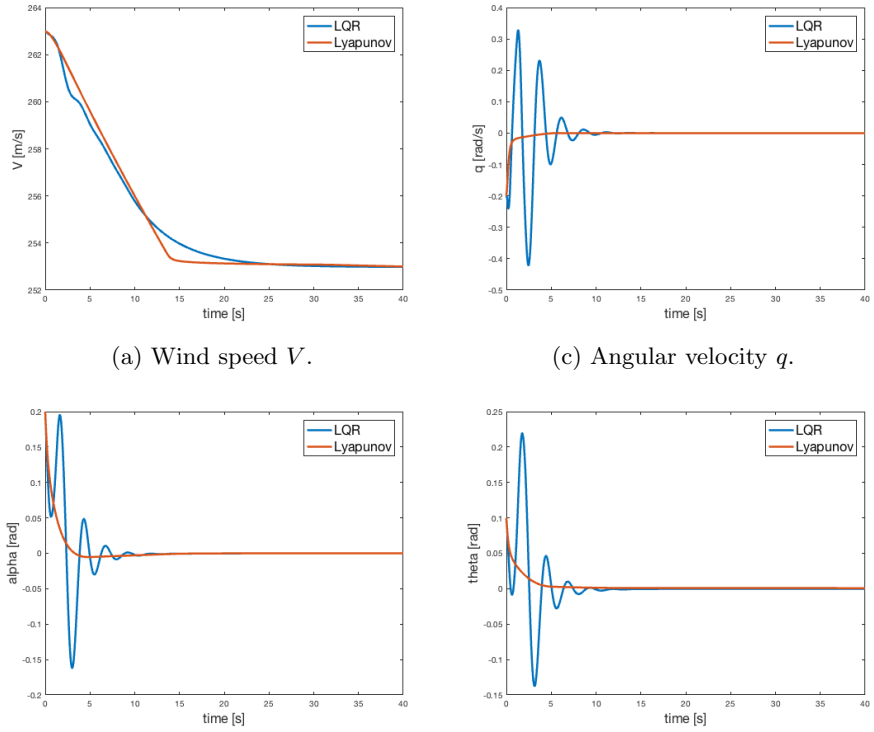
## 6 Simulations

In this Section simulation results are reported for different simulated flights with different controllers among those illustrated in the previous sections, and given as example demos in APRICOT.

**Nonlinear and LQR controls:** First, we compare the performance of the Lyapunov-based controller described in Section 4.3 with the LQR controller outlined in Section 4.1 on the nonlinear longitudinal dynamics in ideal conditions, *i.e.,* with no disturbances, but considering the actuator nonlinearities.

In Fig. 11 results are shown when initial condition is $x_0 = [V, \alpha, q, \theta]^T = [262.98, 0.2, -0.2, 0.1]^T$ around the trim condition $\bar{x} = [252.98, 0, 0, 0]^T$.

As expected the system controlled with the linear control law has higher variations of angular velocities, angular of attack and sideslip angle with respect to the behaviour obtained with the Lyapunov based control.




(a) Wind speed $V$.

(c) Angular velocity $q$.

(b) Angle of attack $\alpha$.

(d) Sideslip angle $\beta$.

Fig. 11: Comparison of Lyapunov and LQR controllers

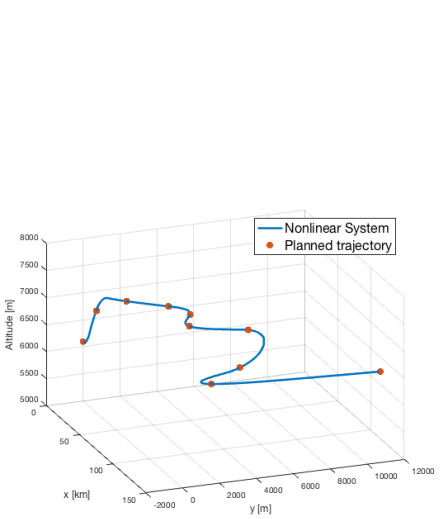
**Flight planning:** Here an example setup for longitudinal and laterodirectional planning is illustrated.

For the longitudinal dynamics, we want to compute a plan to steer the system from the initial conditions $x_0 = [V, \alpha, \theta, q, h]^T = [252.98, 0, 0, 0, 6096]^T$ to the final condition $x_f = [V, \alpha, \theta, q, h]^T = [252.98, 0, 0, 0, 7096]^T$, subject to

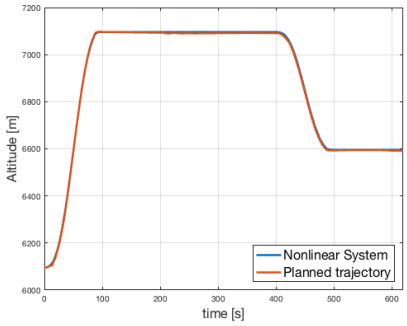
saturations of the control variable as $-0.5 \leq \delta_e \leq 0.5$ and $0 \leq \delta_{th} - \delta_{th,0} \leq 1$, with a thrust control at the equilibrim $\delta_{th,0} = 0.7972$. The altitude overshoot is limited to 10%.

In a similar way the laterodirectional planning is executed considering the following constraints on the ailerons and the tail rudder, corresponding to a maximum excursion of $\pm 20°$, and given as $-0.5 \leq \delta_a \leq 0.5$, $-0.5 \geq \delta_r \leq 0.5$.

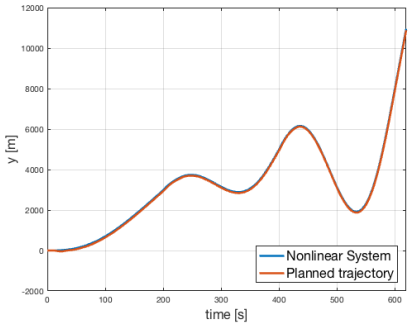
In this way it is possible to compute trajectories to steer the system between several waipoints in order to obtain a complicated path, as that in Fig. 12a whose first $100s$ are the result of the aforementioned planning problem.




(a) 3D Trajectory.




(b) Altitude.



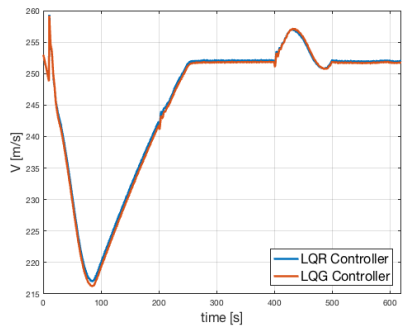

(c) $y$ position.

Fig. 12: Simulation results on the flight plan tracked with the LQG Controller.

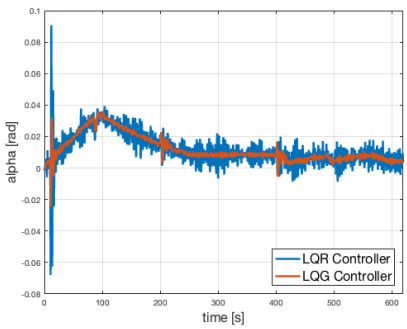
**LQR and LQG controllers:** In the case of LQR and LQG control laws described in Section 4.1, in Fig. 12 we assess the performance of the simulator and the flight planner comparing the ideal aircraft trajectory with the simulated one considering the complete nonlinear model affected by disturbances. Note that for each waypoint reported in Fig. 12a different aircraft orientations are re-

quired and correctly tracked. To better appreciate the system evolution, videos of experiments are visible on APRICOT website [8].

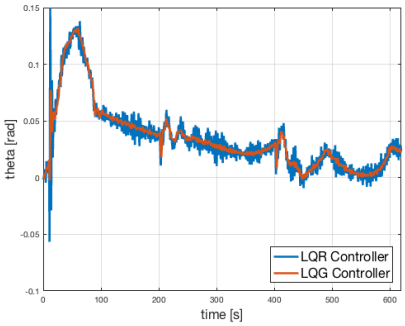
In Fig. 13 the improvement in the LQG control system performance with respect to the LQR case in presence of disturbances can be appreciated.



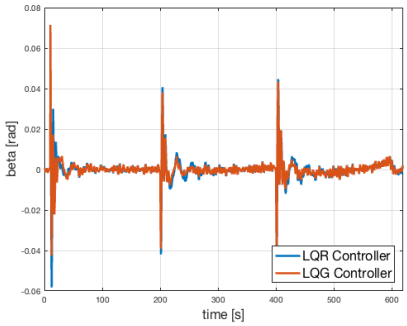

(a) Wind speed $V$.




(c) Angle of attack $\alpha$.




(b) $\theta$ Euler angle.




(d) Sideslip angle $\beta$.

Fig. 13: Comparison of LQR and LQG controllers with noise and wind gust.

**LPV controller:** Finally we consider the quadratic cost index

$$J(t) = \int_0^t x(\tau)^T Q x(\tau) + u(\tau)^T R u(\tau)\, d\tau$$

with diagonal matrices $Q$ and $R$, obtained for the laterodirectional dynamics when the aircraft recovers from a severely perturbed initial state with $\theta = 0.2$ and $\phi = 0.2$, comparing the performance of an LQR and an LPV control law, described in Section 4.2. In this example, the LPV control law has been obtained with multiple LQR controllers designed for different values of $\theta$. The longitudinal

dynamics is controlled with a single LQR controller. Simulation results are reported in Fig. 14 where the performance improvement in the cost function of the LPV controller is visible. On the other hand, the trajectory tracking performance is comparable.

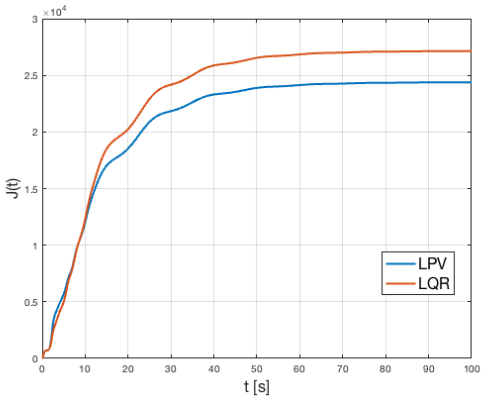



Fig. 14: Comparison between LPV and LQR performance.

## 7 Conclusions

In this work we presented a novel environment for the design and simulation of flight planning and aircraft control systems. The toolbox, named APRICOT, is highly versatile, customizable and provided with user interface such as a GUI, a gamepad interface and an external open source animation environment, namely Flightgear. The software is released as an open source MATLAB/Simulink toolbox and can be used for control design and system performance evaluations under different environmental conditions. The entire APRICOT framework is customizable for various purposes. For example, control laws can be tested for robustness with respect to external perturbations or for flight tracking. With respect to other software solutions, the main strengths of APRICOT are: the open source nature, the reconfigurable control system, the ease of use, the availability of a complete aircraft dynamics and environmental model.

Future developments are directed towards the simulation of quadrotors and other Unmanned Aerial Vehicles. With the purpose of extending APRICOT simulation to handle multi-robot systems, the simulation of both aerial and ground vehicles is under study at the present time, in order to allow the design of coordination/mission control systems.

## References


1. JSBSim. http://jsbsim.sourceforge.net/.
2. jMAVSim. https://github.com/PX4/jMAVSim.
3. Ardupilot. http://ardupilot.org/.
4. PX4. http://px4.io/.
5. Adriano Bittar, Helosman V Figuereido, Poliana Avelar Guimaraes, and Alessandro Correa Mendes. Guidance software-in-the-loop simulation using x-plane and simulink for uavs. In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, pages 993–1002. IEEE, 2014.
6. YI Jenie and T Indriyanto. X-plane-simulink simulation of a pitch-holding automatic control system for boeing 747. In *Indonesian-Taiwan Workshop, Bandung, Indonesia*, 2006.
7. Aircraft control toolbox. http://www.psatellite.com/act/index.php. Princeton Satellite Systems.
8. APRICOT. https://github.com/aferrarelli/APRICOT/.
9. Robert K Heffley and Wayne F Jewell. Aircraft handling qualities data. 1972.
10. Mustafa Cavcar. The international standard atmosphere (isa). *Anadolu University, Turkey*, 30, 2000.
11. Daniel Ondriš and Rudolf Andoga. Aircraft modeling using matlab/flight gear interface. *Acta Avionica*, 15(27), 2013.
12. Istas F Nusyirwan. Engineering flight simulator using matlab, python and flightgear. In *Simtect*. Melbourne Australia, 2011.
13. Mohammed Moness, Ahmed M Mostafa, Mohammed A Abdel-Fadeel, Ahmed I Aly, and A Al-Shamandy. Automatic control education using flightgear and matlab based virtual lab. In *8th International Conference on Electrical Engineering*, pages 1157–1160, 2012.
14. Brian L Stevens, Frank L Lewis, and Eric N Johnson. *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. John Wiley & Sons, 2015.
15. Michael V Cook. *Flight dynamics principles: a linear systems approach to aircraft stability and control*. Butterworth-Heinemann, 2012.
16. Ashish Tewari. *Advanced control of aircraft, spacecraft and rockets*, volume 37. John Wiley & Sons, 2011.
17. Labane Chrif and Zemalache Meguenni Kadda. Aircraft control system using lqg and lqr controller with optimal estimation-kalman filter design. *Procedia Engineering*, 80:245–257, 2014.
18. Andrés Marcos and Gary J Balas. Development of linear-parameter-varying models for aircraft. *Journal of Guidance, Control, and Dynamics*, 27(2):218–228, 2004.
19. Ola Härkegård and Torkel Glad. Flight control design using backstepping. *Linköping University Electronic Press*, 2000.
20. McLean Donald. *Automatic Flight Control System*. Prentice Hall, 1990.
21. Ashish Tewari. *Automatic Control of Atmospheric and Space Flight Vehicles: Design and Analysis with MATLAB® and Simulink®*. Springer Science & Business Media, 2011.
22. David A Caughey. Introduction to aircraft stability and control. In *Lecture Notes*. Cornell University, 2011.