AIAA SciTech Forum
8–12 January 2018, Kissimmee, Florida
2018 AIAA Guidance, Navigation, and Control Conference

10.2514/6.2018-1838

Retrospective Cost Adaptive PID Control of Quadcopter/Fixed-Wing Mode Transition in a VTOL Aircraft

Ahmad Ansari[†], Ningyuan Zhang[††] and Dennis S. Bernstein[‡]

Aerospace Engineering Department, University of Michigan, 1320 Beal Ave., Ann Arbor, MI 48109

Although novel aircraft designs can be quickly conceived and built, the ability to develop and test these concepts is impeded by the inability to rapidly prototype autopilots that can accommodate unconventional airframes with nonstandard sensor/actuator configurations. For vehicles that are complex or expensive and that are intended for military or civil certification and production, autopilots are traditionally designed based on a combination of multi-body modeling, computational fluid dynamics, and wind tunnel testing. However, detailed models are not available for experimental flight vehicles, and thus manual autopilot tuning is needed without guarantees of success. To overcome this problem, the present paper focuses on an adaptive control technique that requires minimal modeling and limited tuning. In particular, we apply retrospective cost adaptive (RCAC) control to three aircraft simulation models. The first aircraft is a quadcopter, and the second is a fixed-wing vehicle. Next, we consider a VTOL vehicle that can fly in both quadcopter and fixed-wing modes. The RCAC tunings for the VTOL aircraft in quadcopter and fixed-wing mode are chosen to be identical to the tunings of the quadcopter and fixed-wing aircraft despite the differences in vehicle geometry. The goal is to assess the ability of RCAC to control different airframes with the same tunings, while also assessing the ability to control the VTOL aircraft during mode transition.

Nomenclature

u	=	input
z	=	performance variable
$K_{\rm P}, K_{\rm I}, K_{\rm D}$	=	proportional, integrator, derivative gains
θ	=	controller coefficients
ϕ	=	feedback vector
$G_{\rm f}$	=	filter transfer function applied to ϕ and u
$\mathbf{z}$	=	Z-transform variable
R_θ	=	controller-coefficient penalty
R_u	=	control penalty
Θ, Φ, Ψ	=	pitch, roll, yaw angles
${\rm F_E}$	=	Earth frame defined with the axes $\hat\imath_{\rm E}$ and $\hat\jmath_{\rm E}$ horizontal, while the axis $\hat k_{\rm E}$ points downward
${\rm F_{AC}}$	=	aircraft body-fixed frame defined with $\hat\imath_{\rm AC}$ pointing out the nose of the aircraft,

[†]Ph.D. Candidate, Aerospace Engineering Department, University of Michigan, Ann Arbor. ansahmad@umich.edu
[††]Graduate Student, Aerospace Engineering Department, University of Michigan, Ann Arbor. ningyuan@umich.edu
[‡]Professor, Aerospace Engineering Department, University of Michigan, Ann Arbor. dsbaero@umich.edu

		$\hat{\jmath}_{\rm AC}$ pointing out the right wing, and $\hat{k}_{\rm AC}$ pointing downward
P, Q, R	=	angular velocity of $\rm F_{AC}$ relative to $\rm F_E$ in direction $\hat{\imath}_{\rm AC}$, $\hat{\jmath}_{\rm AC}$, $\hat{k}_{\rm AC}$
$V_{\rm AC}$	=	airspeed
h	=	altitude
X	=	East coordinate
Y	=	North coordinate
R_i	=	throttle of an $i^{\rm th}$ rotor of a quadcopter
$\boldsymbol{T}$	=	throttle of an engine of a fixed-wing aircraft
$\boldsymbol{e}$	=	elevator
$\boldsymbol{a}$	=	ailerons
$\boldsymbol{ev}$	=	elevons

I. Introduction

UAV technology has ushered in a new age of flight innovation, where novel vehicle designs can be quickly conceived, built, and tested. To meet mission objectives, these vehicles may employ fixed or variable geometry as well as nonstandard sensor/actuator configurations. However, the ability to develop and test new vehicle concepts is impeded by the lack of one key technological ingredient, namely, the ability to rapidly prototype autopilots that can accommodate unconventional vehicles.

Traditionally, there are two ways to prototype an autopilot. The first way is to develop a detailed structural and aerodynamic model of the vehicle, through a combination of multi-body modeling, computational fluid dynamics, and wind tunnel testing. This approach is standard practice for vehicles that are complex or expensive and that are intended for military or civil certification and production. However, for many experimental flight vehicles, such as low-cost designs, rapidly changing configurations, or highly complex hypersonic vehicles, detailed modeling is either incomplete or nonexistent. The second way is to program an off-the-shelf autopilot and manually tweak the control gains until the performance is acceptable. This approach may fail if the proposed vehicle concept is far from existing vehicles. This means that, when an innovative aircraft design is proposed, the developers must face either expensive and time-consuming modeling or excessively prolonged testing with no guarantee of success.

To overcome this problem, the present paper focuses on adaptive control, which requires a minimal amount of modeling and a minimal amount of manual tuning. This approach could be directly applicable to an experimental flight program in which structural modeling, computational fluid dynamics, and wind tunnel testing are either incomplete or nonexistent.

Adaptive control has been viewed as an enabling technology for autopilots as early as the X-15 program[1,2] and as recently as the NASA Airborne Subscale Transport Aircraft Research (AirSTAR) testbed.[3] These and other techniques do not fulfill the vision of truly rapid prototyping since they depend on an autopilot designed for a nominal model and they have limited ability to accommodate high levels of uncertainty.

Using the NASA GTM model,[4–7] retrospective cost adaptive control (RCAC) was applied to the nominal aircraft as well as unknown off-nominal conditions.[8] RCAC is a direct, discrete-time adaptive control technique for stabilization, command following, and disturbance rejection.[9] As a discrete-time approach, RCAC is motivated by the desire to implement control algorithms that operate at the sensor sample rate without the need for controller discretization. This also means that the required modeling information can be estimated based on data sampled at the same rate as the control update.

RCAC was originally motivated by the notion of retrospectively optimized control, where past controller coefficients used to generate past control inputs are re-optimized in the sense that if the re-optimized coefficients had been used over a previous window of operation, then the performance would have been better. Unlike signal processing applications such as estimation and identification, however, it is impossible to change past control inputs, and thus the re-optimized controller coefficients are used only to generate the next control input.

RCAC was originally developed within the context of active noise control experiments.[10] The algorithm used in[10] is gradient-based, where the gradient direction and step size are based on different cost functions. In subsequent work,[11] the gradient algorithm was replaced by batch least-squares optimization. In both[10] and,[11] the modeling information is given by Markov parameters (impulse response components) of the open-loop transfer function G_{zu} from the control input u to the performance variable z. More recently, in,[12] a recursive least squares (RLS) algorithm was used, along with knowledge of the NMP zeros of G_{zu}. The approaches in[10–12] are closely related in the sense that all of the NMP zeros outside of the spectral radius of G_{zu} are approximate zeros of a polynomial whose coefficients are Markov parameters of G_{zu}; this polynomial is a truncated Laurent expansion of G_{zu} about infinity. RCAC uses a filter $G_{\rm f}$ to define the retrospective cost by filtering the difference between the actual past control inputs and the re-optimized control inputs. To construct $G_{\rm f}$, Markov parameters are used in,[10, 11, 13] and NMP zeros are used in.[12, 14]

More recently,[9] RCAC was reinterpreted as a residual-minimization technique involving the transfer function between a virtual control input and the performance variable. In particular, it was shown in[9] that the filter $G_{\rm f}$ used to define the retrospective performance variable provides a target model for an intercalated transfer function. In effect, RCAC updates the controller coefficients so as to fit the intercalated transfer function to the target model. This mechanism supports the principle underlying RCAC. The fact that the target model $G_{\rm f}$ requires limited modeling information (primarily sign and NMP zeros) makes it possible to apply RCAC to highly uncertain plants.

In the present paper, we apply RCAC to three aircraft. The first aircraft is a quadcopter, and the second is a fixed-wing vehicle. These vehicles provide an initial test for obtaining the filter $G_{\rm f}$ and the weightings used by RCAC to construct the retrospective cost function. Next, we consider a third vehicle, which is capable of VTOL flight; this vehicle has the ability to fly in both quadcopter and fixed-wing modes. We then apply the same filters and weightings chosen for the quadcopter to the VTOL vehicle operating in the quadcopter mode. Similarly, we apply the same filters and weightings chosen for the fixed-wing aircraft to the VTOL vehicle operating in the fixed-wing mode. This approach allows us to assess the ability of RCAC to control different airframes with the same tunings, while also assessing the ability to control the VTOL aircraft during mode transition. To do this, we interface RCAC with the Pixhawk firmware (PX4 flight stack) to simulate adaptive control of a quadcopter, fixed-wing, and VTOL aircraft models. The goal is to adaptively tune the PID controllers using minimal modelling information by employing the retrospective cost algorithm given in Section II.

All simulations are performed within the context of Gazebo.[15] Gazebo is an open-source high-fidelity 3D simulation environment for autonomous robots, which includes a quadcopter, fixed-wing, and VTOL aircraft models. The PX4 flight stack is a collection of guidance, navigation, and control algorithms for autonomous drones, including PID controllers for quadcoptor, fixed-wing, and VTOL aircraft. To reflect the absence of additional modeling information, in all the simulations considered in this paper, we choose the initial controller coefficients $\theta(0) = 0$. This means that the gains $K_{\rm P}$, $K_{\rm I}$, and $K_{\rm D}$ are initialized at zero in the RC-P, RC-PI, and RC-PID control algorithms.

The contents of the paper are as follows. Section II presents the algorithm for retrospective cost adaptive PID control. Section III explains the quadcopter control architecture, and presents the simulation results for a quadcopter flight. Section IV explains the fixed-wing control architecture, and presents the simulation results for a fixed-wing flight. Using the quadcopter and fixed-wing control architectures presented in Sections III and IV, Section V presents the VTOL control architecture along with the simulation results.

II. Retrospective Cost Adaptive PID Control

A. Plant Model

Consider the discrete-time system

$$x(k+1) = Ax(k) + Bu(k) + D_1 w(k), \tag{1}$$

$$z(k) = E_1 x(k) + E_0 w(k), \tag{2}$$

where $x(k) \in \mathbb{R}^{l_x}$ is the state, $u(k) \in \mathbb{R}^{l_u}$ is the input, $w(k) \in \mathbb{R}^{l_w}$ is the exogenous signal, and $z(k) \in \mathbb{R}^{l_z}$ is the performance variable. This model may represent a sampled-data version of a continuous-time plant with sample time T_s, in which case $x(k)$ denotes the state at time $t = kT_\mathrm{s}$. The goal is to develop an adaptive output feedback controller that minimizes z in the presence of the exogenous signal w with limited modeling information about (1), (2). The components of w can represent either command signals to be followed, external disturbances to be rejected, or both, depending on the choice of D_1 and E_0. No controllability and observability assumptions are made concerning the state space realization since the adaptive control algorithm requires input-output model information rather than details of the state space realization.

B. The Controller

Define

$$z_\mathrm{int}(k) \overset{\triangle}{=} z_\mathrm{int}(k-1) + T_\mathrm{s} z(k), \tag{3}$$

$$z_\mathrm{diff}(k) \overset{\triangle}{=} \frac{1}{T_\mathrm{s}}[z(k) - z(k-1)], \tag{4}$$

where $z_\mathrm{int}(k)$ is the integrator state and $z_\mathrm{diff}(k)$ is the differentiator state. Define the discrete-time PID controller

$$u(k) = K_\mathrm{P}(k)z(k) + K_\mathrm{I}(k)z_\mathrm{int}(k) + K_\mathrm{D}(k)z_\mathrm{diff}(k), \tag{5}$$

where $K_\mathrm{P} \in \mathbb{R}^{l_u \times l_z}$, $K_\mathrm{I} \in \mathbb{R}^{l_u \times l_z}$, and $K_\mathrm{D} \in \mathbb{R}^{l_u \times l_z}$ are the proportional, integral, and derivative gain matrices, respectively. We rewrite (5) as

$$u(k) = \phi(k)\theta(k), \tag{6}$$

where the regressor matrix $\phi(k)$ is defined by

$$\phi(k) \overset{\triangle}{=} \begin{bmatrix} z(k) \\ z_\mathrm{int}(k) \\ z_\mathrm{diff}(k) \end{bmatrix}^\mathrm{T} \otimes I_{l_u} \in \mathbb{R}^{l_u \times l_\theta} \tag{7}$$

and

$$\theta(k) \overset{\triangle}{=} \mathrm{vec}\begin{bmatrix} K_\mathrm{P}(k) & K_\mathrm{I}(k) & K_\mathrm{D}(k) \end{bmatrix} \in \mathbb{R}^{l_\theta}, \tag{8}$$

where $l_\theta \overset{\triangle}{=} 3l_u l_z$, "$\otimes$" is the Kronecker product, and "vec" is the column-stacking operator. By modifying ϕ and θ, we can obtain special cases of the PID controller, for example, P, PI, and PD control laws.

C. Retrospective Performance Variable

We define the retrospective control as

$$\hat{u}(k) = \phi(k)\hat{\theta} \tag{9}$$

and the corresponding retrospective performance variable as

$$\hat{z}(k) \overset{\triangle}{=} z(k) + \phi_{\mathrm{f}}(k)\hat{\theta} - u_{\mathrm{f}}(k), \tag{10}$$

where $\hat{\theta} \in \mathbb{R}^{l_\theta}$ is determined by optimization below, and $\phi_{\mathrm{f}}(k) \in \mathbb{R}^{l_z \times l_\theta}$ and $u_{\mathrm{f}}(k) \in \mathbb{R}$ are filtered versions of $\phi(k)$, and $u(k)$, respectively, defined by

$$\phi_{\mathrm{f}}(k) \overset{\triangle}{=} G_{\mathrm{f}}(\mathbf{q})\phi(k), \tag{11}$$

$$u_{\mathrm{f}}(k) \overset{\triangle}{=} G_{\mathrm{f}}(\mathbf{q})u(k). \tag{12}$$

The filter G_{f} has the form

$$G_{\mathrm{f}}(\mathbf{q}) \overset{\triangle}{=} D_{\mathrm{f}}^{-1}(\mathbf{q})N_{\mathrm{f}}(\mathbf{q}), \tag{13}$$

where D_{f} and N_{f} are polynomial matrices and D_{f} is asymptotically stable and monic. The choice of G_{f} is discussed in the next section.

D. Construction of G_{f}

In,[10, 11, 16–18] G_{f} is based on the Markov parameters of the control-to-performance transfer matrix G_{zu}. In particular, for all complex numbers z whose absolute value is greater than the spectral radius of A, it follows that

$$G_{zu}(\mathbf{q}) = E_1(\mathbf{q}I - A)^{-1}B = \sum_{i=0}^{\infty} \frac{H_i}{\mathbf{q}^i}, \tag{14}$$

where, for all, $i \geq 1$, the i^{th} Markov parameter of G_{zu} is defined by

$$H_i \overset{\triangle}{=} E_1 A^{i-1} B. \tag{15}$$

Truncating (14) yields G_{f} given by the $n_{\mathrm{f}}^{\mathrm{th}}$-order Markov parameter-based FIR filter

$$G_{\mathrm{f}}(\mathbf{q}) = \sum_{i=1}^{n_{\mathrm{f}}} \frac{H_i}{\mathbf{q}^i}. \tag{16}$$

In,[9] G_{f} is shown to provide a target model for the intercalated transfer function between the virtual control perturbation and z. This interpretation leads to guidelines for constructing G_{f} in terms of limited modeling data that consists of the leading signs, relative degree, and NMP zeros of G_{zu}.

E. Retrospective Cost Function

Using the retrospective performance variable $\hat{z}(k)$, we define the retrospective cost function

$$J(k,\hat{\theta}) \overset{\triangle}{=} \sum_{i=1}^{k} [\hat{z}^{\mathrm{T}}(i)\hat{z}(i) + (\phi(i)\hat{\theta})^{\mathrm{T}} R_u \phi(i)\hat{\theta}] + (\hat{\theta} - \theta(0))^{\mathrm{T}} R_\theta (\hat{\theta} - \theta(0)), \tag{17}$$

where R_u and R_θ are positive definite.

Proposition 1: Let $P(0) = R_\theta^{-1}$. Then, for all $k \geq 1$, the retrospective cost function (17) has a unique global minimizer $\theta(k)$, which is given by

$$\theta(k) = \theta(k-1) - P(k-1)\tilde{\phi}(k)^{\mathrm{T}}\Gamma(k)^{-1}[\tilde{\phi}(k)\theta(k-1) + \tilde{z}(k)], \tag{18}$$

$$P(k) = P(k-1) - P(k-1)\tilde{\phi}(k)^{\mathrm{T}}\Gamma(k)^{-1}\tilde{\phi}(k)P(k-1), \tag{19}$$

where

$$\tilde{\phi}(k) \triangleq \begin{bmatrix} \phi_{\rm f}(k) \\ \phi(k) \end{bmatrix} \in \mathbb{R}^{(l_z+l_u)\times l_\theta}, \tag{20}$$

$$\tilde{R}(k) \triangleq \begin{bmatrix} 1 & 0 \\ 0 & R_u(k) \end{bmatrix} \in \mathbb{R}^{(l_z+l_u)\times(l_z+l_u)}, \tag{21}$$

$$\tilde{z}(k) \triangleq \begin{bmatrix} z(k) - u_{\rm f}(k) \\ 0 \end{bmatrix} \in \mathbb{R}^{(l_z+l_u)}, \tag{22}$$

$$\Gamma(k) \triangleq \tilde{R}(k)^{-1} + \tilde{\phi}(k)P(k-1)\tilde{\phi}(k)^{\rm T}. \tag{23}$$

III. Adaptive Control of Quadcopter Flight

The interface of PX4 with Gazebo for a quadcopter (QC) is shown in Figure 1. The control architecture consists of three loops. The outer loop employs the mission planner[19] to follow position commands $(X_{\rm ref}, Y_{\rm ref}, Z_{\rm ref})$ by generating Euler-angle reference commands $(\Phi_{\rm ref}, \Theta_{\rm ref}, \Psi_{\rm ref})$ for the middle loop. The middle loop uses three P controllers and generates Euler-angle-rate commands $(\dot{\Phi}_{\rm ref}, \dot{\Theta}_{\rm ref}, \dot{\Psi}_{\rm ref})$, which is then transformed to angular-velocity commands $(P_{\rm ref}, Q_{\rm ref}, R_{\rm ref})$ for the inner loop. The inner loop uses a combination of three static-feedfoward (F) and three PID controllers to generate angular-acceleration commands $(\dot{P}_{\rm ref}, \dot{Q}_{\rm ref}, \dot{R}_{\rm ref})$ for the mixer, which generates the quadcopter control inputs. The control inputs R_1, R_2, R_3, and R_4 are the rotor throttle settings.

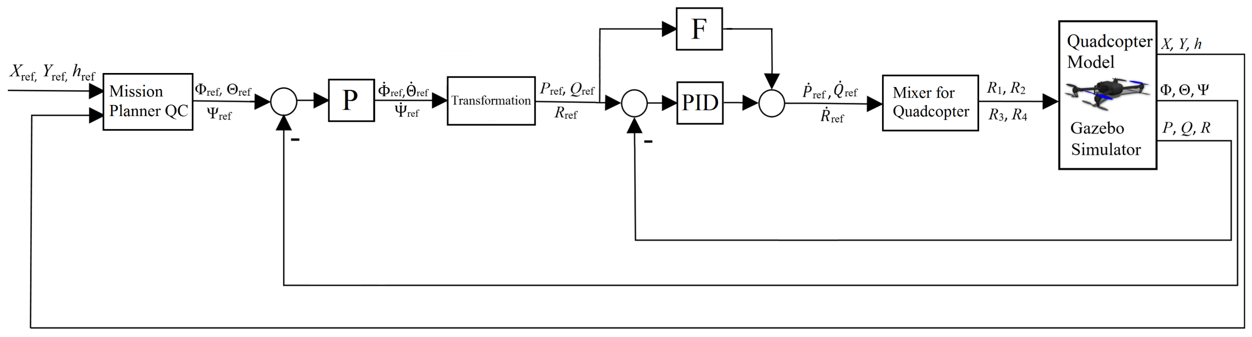

Figure 1. Interface of the PX4 flight stack with Gazebo for a quadcopter. The middle loop consists of three P controllers to follow Euler-angle commands $(\Phi_{\rm ref}, \Theta_{\rm ref}, \Psi_{\rm ref})$, whereas the inner loop consists of three feedforward-augmented-PID controllers to follow angular-velocity commands $(P_{\rm ref}, Q_{\rm ref}, R_{\rm ref})$.

To adaptively fly a quadcopter using the PX4 control architecture (as shown in Figure 1), we remove the feedforward controllers, and replace the fixed-gain PX4 P and PID controllers with RC-PI and RC-PID controllers, respectively. The performance variable z, the control input u, the filter $G_{\rm f}$, and the weightings for the middle-loop RC-PI and the inner-loop RC-PID controllers are given in the following subsection. Since Gazebo uses a nonlinear quadcopter model, the guidelines for choosing $G_{\rm f}$ given in[9] cannot be used directly. The entries of $G_{\rm f}$ are thus chosen based on the numerical response of a quadcopter model under nominal conditions. In addition, for all simulations associated with each architecture, we use identical tuning parameters for QC controller; consequently, no special tuning for QC controller is used for particular simulations in this paper.

A. Quadcopter Adaptive Control Architectures

1. RC-PI Controller for Roll-Angle Commands

The performance variable z and the control input u for RC-PI control of roll-angle commands in the middle loop are given by

$$z(k) \triangleq \Phi_{\rm ref}(k) - \Phi(k), \tag{24}$$

$$u(k) \triangleq \dot{\Phi}_{\rm ref}(k). \tag{25}$$

The filter and weightings are given by $G_{\rm f}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 0.02$, and $R_\theta = 0.1 I_2$.

2. RC-PI Controller for Pitch-Angle Commands

The performance variable z and the control input u for RC-PI control of pitch-angle commands in the middle loop are given by

$$z(k) \triangleq \Theta_{\rm ref}(k) - \Theta(k), \tag{26}$$

$$u(k) \triangleq \dot{\Theta}_{\rm ref}(k). \tag{27}$$

The filter and weightings are given by $G_{\rm f}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 0.04$, and $R_\theta = 0.1 I_2$.

3. RC-PI Controller for Yaw-Angle Commands

The performance variable z and the control input u for RC-PI control of yaw-angle commands in the middle loop are given by

$$z(k) \triangleq \Psi_{\rm ref}(k) - \Psi(k), \tag{28}$$

$$u(k) \triangleq \dot{\Psi}_{\rm ref}(k). \tag{29}$$

The filter and weightings are given by $G_{\rm f}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 0.05$, and $R_\theta = 0.3 I_2$.

4. RC-PID Controller for Roll-Axis Angular-Velocity Commands

The performance variable z and the control input u for RC-PID control of angular-velocity commands about the roll-axis in the inner loop are given by

$$z(k) \triangleq P_{\rm ref}(k) - P(k), \tag{30}$$

$$u(k) \triangleq \dot{P}_{\rm ref}(k). \tag{31}$$

The filter and weightings are given by $G_{\rm f}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 1$, and $R_\theta = 5 I_3$.

5. *RC-PID Controller for Pitch-Axis Angular-Velocity Commands*

The performance variable z and the control input u for RC-PID control of angular-velocity commands about the pitch-axis in the inner loop are given by

$$z(k) \triangleq Q_{\rm ref}(k) - Q(k), \tag{32}$$

$$u(k) \triangleq \dot{Q}_{\rm ref}(k). \tag{33}$$

The filter and weightings are given by $G_{\rm f}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 0.4$, and $R_\theta = 4I_3$.

6. *RC-PID Controller for Yaw-Axis Angular-Velocity Commands*

The performance variable z and the control input u for RC-PID control of angular-velocity commands about the yaw-axis in the inner loop are given by

$$z(k) \triangleq R_{\rm ref}(k) - R(k), \tag{34}$$

$$u(k) \triangleq \dot{R}_{\rm ref}(k). \tag{35}$$

The filter and weightings are given by $G_{\rm f}(\mathbf{z}) = 1$, $R_u = 5$, and $R_\theta = I_3$.

B. Numerical Simulation

The Gazebo quadcopter model that we use in the simulation is shown in Figure 2. The radius of all the four rotor blades is 0.13 m.

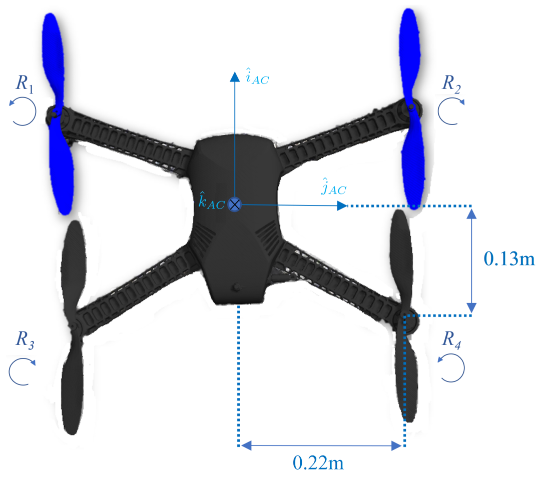

Figure 2. Gazebo quadcopter model.

Figure 3 shows the reference waypoints $(X_{\rm ref}, Y_{\rm ref})$ and altitude $h_{\rm ref}$ for the mission planner. Note that the command is to take off and then fly East, then North, then East, and then hover over the last waypoint at an altitude of 20 m. Note that the quadcopter follows the commanded altitude, while flying between and over the commanded waypoints.

Figure 4 shows the performance of middle-loop RC-PI controllers, as well as, the gains $(K_{\rm P}, K_{\rm I})$, and the controllers output $(\dot{\Phi}_{\rm ref}, \dot{\Theta}_{\rm ref}, \dot{\Psi}_{\rm ref})$. Note that the quadcopter follows Euler angle commands

$(\Phi_{\rm ref}, \Theta_{\rm ref}, \Psi_{\rm ref})$. At $t = 50.8$ sec, the command-following errors in the Euler angles Φ, Θ, and Ψ are -0.01 deg, 0.1 deg, and 0.5 deg, respectively. Figure 5 shows the performance of inner-loop RC-PID controllers, as well as, the gains $(K_{\rm P}, K_{\rm I}, K_{\rm D})$, and the controllers output $(\dot{P}_{\rm ref}, \dot{Q}_{\rm ref}, \dot{R}_{\rm ref})$. At $t = 50.8$ sec, the command-following errors in the angular-velocity components P, Q, and R are 0.3 deg/sec, 1 deg/sec, and -0.2 deg/sec, respectively. Figure 6 shows the quadcopter control inputs generated by the mixer.

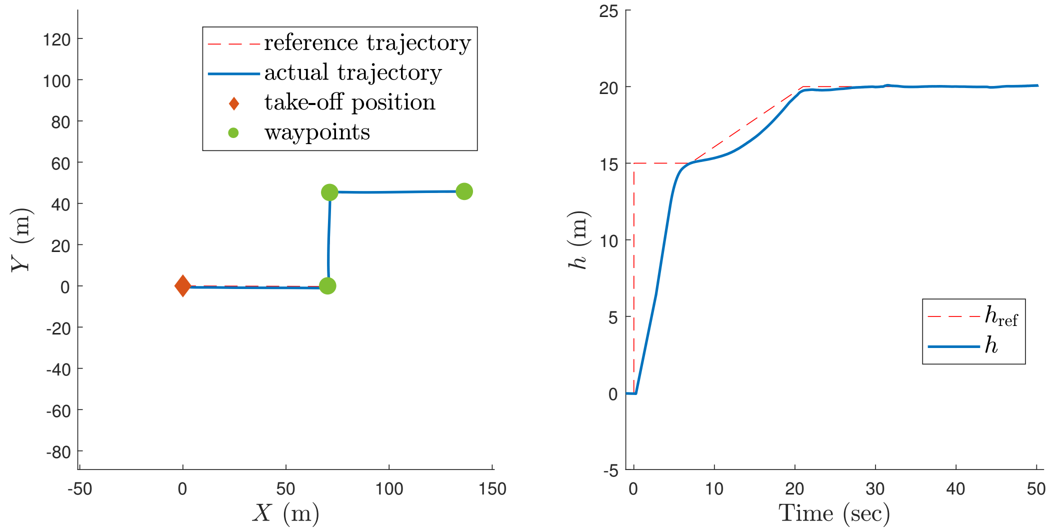

Figure 3. Trajectory of the quadcopter. The command is to take off, and then fly to East, then North, then East, and then hover over the last waypoint at an altitude of 20 m. The quadcopter follows the commanded altitude by flying between and over the commanded waypoints.

IV. Adaptive Control of Fixed-Wing Flight

The interface of PX4 with Gazebo for a fixed-wing (FW) aircraft is shown in Figure 7. The control architecture consists of three loops. The outer loop employs the mission planner[20] to follow position commands $(X_{\rm ref}, Y_{\rm ref}, Z_{\rm ref})$ by generating Euler-angle reference commands $(\Phi_{\rm ref}, \Theta_{\rm ref})$ for the middle loop. The middle loop uses two P controllers and generates Euler-angle-rate commands $(\dot{\Phi}_{\rm ref}, \dot{\Theta}_{\rm ref})$. Note that the yaw-angle-rate command $\dot{\Psi}_{\rm ref} = 0$. The Euler-angle-rate commands are then transformed to angular-velocity commands $(P_{\rm ref}, Q_{\rm ref}, R_{\rm ref})$ for the inner loop. The inner loop uses a combination of three static-feedfoward (F) and three PID controllers to generate angular-acceleration commands $(\dot{P}_{\rm ref}, \dot{Q}_{\rm ref} \dot{R}_{\rm ref})$ for the mixer, which generates the aircraft control inputs. The control inputs $\boldsymbol{T}$, $\boldsymbol{e}$, and $\boldsymbol{a}$, are the throttle setting, elevator deflection, and ailerons deflection, respectively. Comparing Figures 1 and 7, note that the fixed-wing interface is similar to the quadcopter interface, except that, for the fixed-wing interface, there is no yaw-angle controller in the middle loop and no angular-velocity D controller in the inner loop.

To adaptively fly a fixed-wing aircraft using the PX4 control architecture (as shown in Figure 7), we remove the feedforward controllers, and replace the fixed-gain PX4 P and PI controllers with RC-PI and RC-PI controllers, respectively. The performance variable z, the control input u, the filter $G_{\rm f}$, and the weightings for the middle-loop RC-PI and the inner-loop RC-PI controllers are given in the following subsection. Since Gazebo uses a nonlinear fixed-wing aircraft model, the guidelines for choosing $G_{\rm f}$ given in[9] cannot be used directly. The entries of $G_{\rm f}$ are thus chosen based on the numerical response of a fixed-wing aircraft model under nominal conditions. In addition, for all simulations associated with each architecture, we use identical tuning parameters for FW controller; consequently, no special tuning for FW controller is used for particular simulations in this paper.

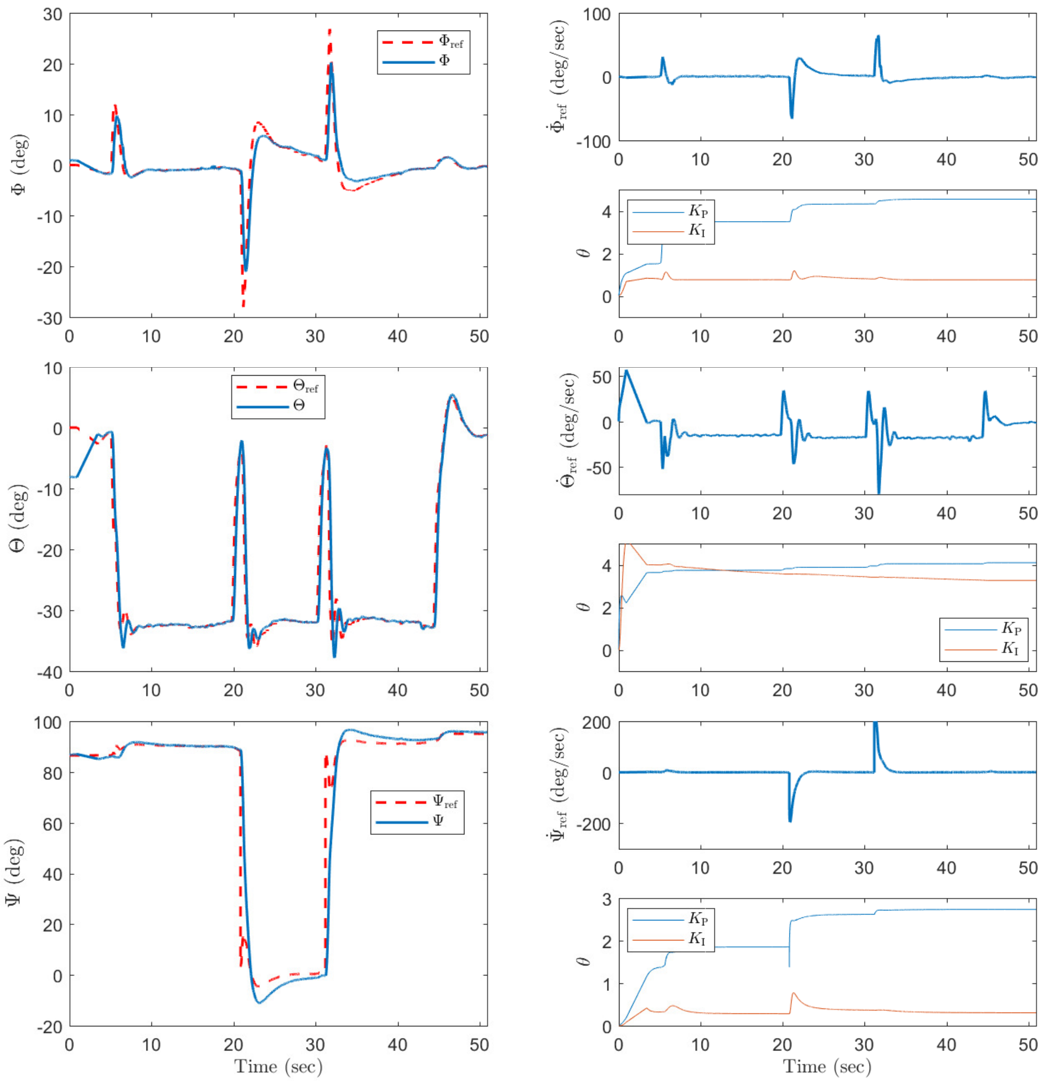

Figure 4. Performance of the quadcopter middle-loop RC-PI controllers for Euler-angle commands. The controller coefficients θ, which are initialized at zero, converge for each RC-PI controller. At $t = 50.8$ sec, the command-following errors in the Euler angles Φ, Θ, and Ψ are -0.01 deg, 0.1 deg, and 0.5 deg, respectively.

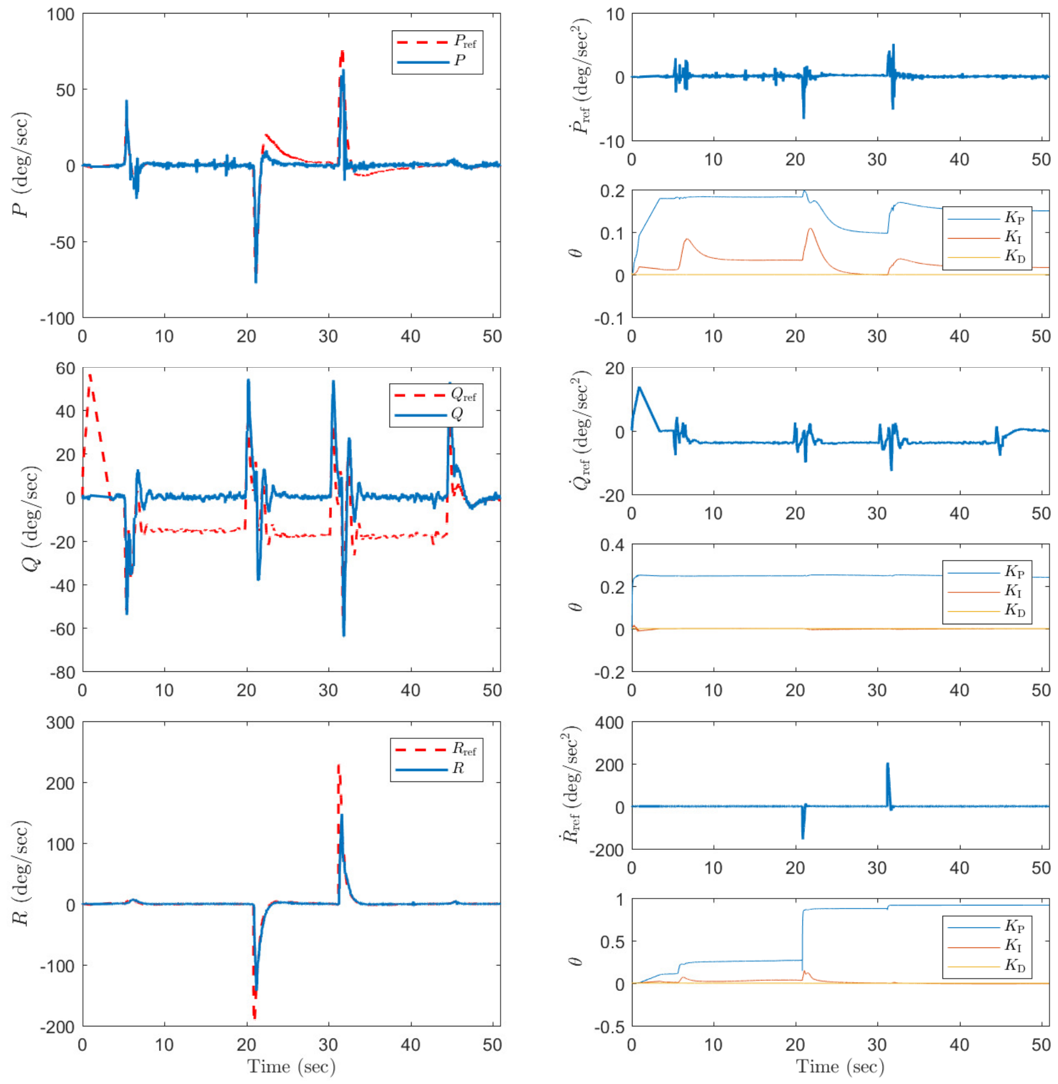

Figure 5. Performance of the quadcopter inner-loop RC-PID for angular-velocity commands. The controller coefficients θ converge for each RC-PID controller. At $t = 50.8$ sec, the command-following errors in the angular-velocity components P, Q, and R are 0.3 deg/sec, 1 deg/sec, and -0.2 deg/sec, respectively.

A. Fixed-wing Adaptive Control Architectures

1. *RC-PI Controller for Roll-Angle Commands*

The performance variable z and the control input u for RC-PI control of roll-angle commands in the middle loop are given by

$$z(k) \triangleq \Phi_{\rm ref}(k) - \Phi(k), \tag{36}$$

$$u(k) \triangleq \dot{\Phi}_{\rm ref}(k). \tag{37}$$

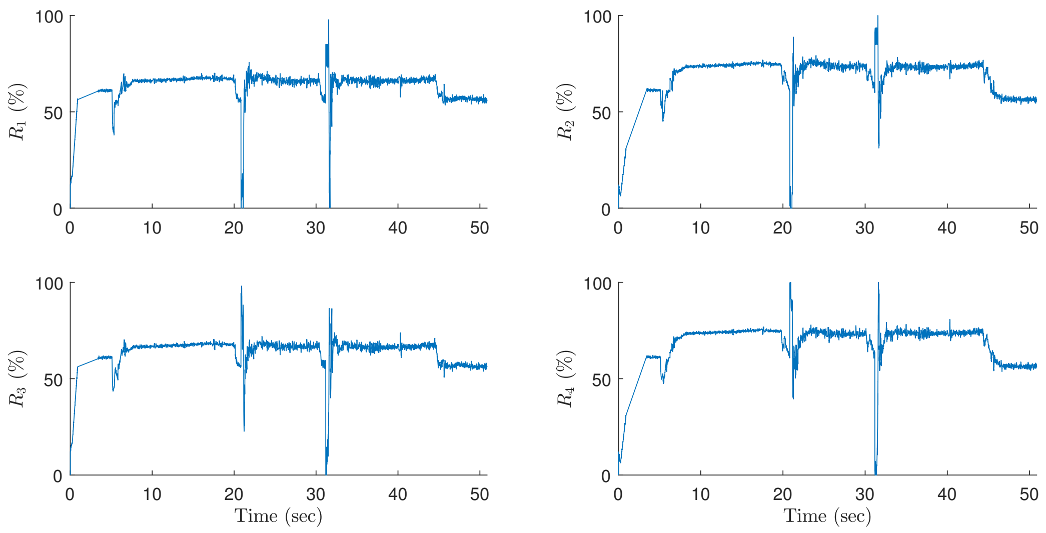

Figure 6. Quadcopter control inputs corresponding to Figure 5.

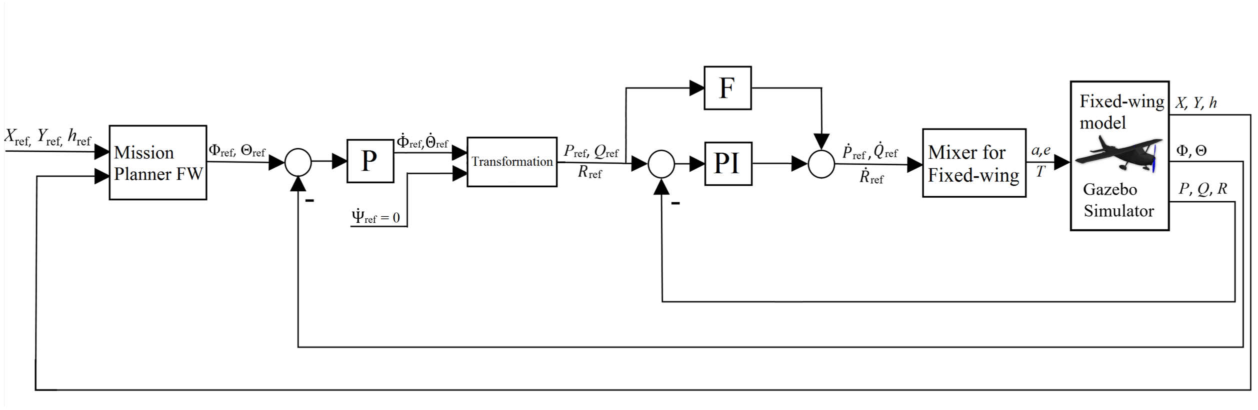
$X_{\mathrm{ref}}, Y_{\mathrm{ref}}, h_{\mathrm{ref}}$
Mission Planner FW
$\Phi_{\mathrm{ref}}, \Theta_{\mathrm{ref}}$
P
$\dot\Phi_{\mathrm{ref}}, \dot\Theta_{\mathrm{ref}}$
$\dot\Psi_{\mathrm{ref}} = 0$
Transformation
$P_{\mathrm{ref}}, Q_{\mathrm{ref}}$
R_{ref}
F
PI
$\dot P_{\mathrm{ref}}, \dot Q_{\mathrm{ref}}$
$\dot R_{\mathrm{ref}}$
Mixer for Fixed-wing
a,e
T
Fixed-wing model
Gazebo Simulator
X, Y, h
Φ, Θ
P, Q, R
-
-

Figure 7. Interface of the PX4 flight stack with Gazebo for a fixed-wing aircraft. The middle loop consists of two P controllers to follow the Euler-angle commands $(\Phi_{\mathrm{ref}}, \Theta_{\mathrm{ref}})$, whereas the inner loop consists of three feedforward-augmented-PI controllers to follow the angular-velocity commands $(P_{\mathrm{ref}}, Q_{\mathrm{ref}}, R_{\mathrm{ref}})$.

The filter and weightings are given by $G_{\mathrm{f}}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 0.03$, and $R_\theta = 0.1 I_2$.

2. RC-PI Controller for Pitch-Angle Commands

The performance variable z and the control input u for RC-P control of pitch-angle commands in the middle loop are given by

$$z(k) \triangleq \Theta_{\mathrm{ref}}(k) - \Theta(k), \tag{38}$$

$$u(k) \triangleq \dot\Theta_{\mathrm{ref}}(k). \tag{39}$$

The filter and weightings are given by $G_{\mathrm{f}}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 0.01$, and $R_\theta = 0.1 I_2$.

3. *RC-PI Controller for Roll-Axis Angular-Velocity Commands*

The performance variable z and the control input u for RC-PID control of angular-velocity commands about the roll-axis in the inner loop are given by

$$z(k) \triangleq P_{\mathrm{ref}}(k) - P(k), \tag{40}$$

$$u(k) \triangleq \dot{P}_{\mathrm{ref}}(k). \tag{41}$$

The filter and weightings are given by $G_{\mathrm{f}}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 20$, and $R_\theta = 2I_2$.

4. *RC-PI Controller for Pitch-Axis Angular-Velocity Commands*

The performance variable z and the control input u for RC-PID control of angular-velocity commands about the pitch-axis in the inner loop are given by

$$z(k) \triangleq Q_{\mathrm{ref}}(k) - Q(k), \tag{42}$$

$$u(k) \triangleq \dot{Q}_{\mathrm{ref}}(k). \tag{43}$$

The filter and weightings are given by $G_{\mathrm{f}}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 1$, and $R_\theta = I_2$.

5. *RC-PI Controller for Yaw-Axis Angular-Velocity Commands*

The performance variable z and the control input u for RC-PID control of angular-velocity commands about the yaw-axis in the inner loop are given by

$$z(k) \triangleq R_{\mathrm{ref}}(k) - R(k), \tag{44}$$

$$u(k) \triangleq \dot{R}_{\mathrm{ref}}(k). \tag{45}$$

The filter and weightings are given by $G_{\mathrm{f}}(\mathbf{z}) = 1/\mathbf{z}$, $R_u = 0.01$, and $R_\theta = I_2$.

B. Numerical Simulation

The Gazebo fixed-wing aircraft model that we use in the simulation is shown in Figure 8. The model has a wing span of 0.94 m and tip-chords of length 0.12 m. It has one propeller engine T, one set of differential ailerons a, one rudder r, and one elevator e.

Figure 9 shows the reference waypoints $(X_{\mathrm{ref}}, Y_{\mathrm{ref}})$ and altitude h_{ref} for the mission planner. Note that the command is to take off and then loiter around the commanded waypoint while maintaining the altitude at 40 m. Note that the fixed-wing aircraft follows the commanded altitude, while loitering around the commanded waypoint.

Figure 10 shows the performance of middle-loop RC-PI controllers, as well as, the gains $(K_{\mathrm{P}}, K_{\mathrm{I}})$, and the controllers output $(\dot{\Phi}_{\mathrm{ref}}, \dot{\Theta}_{\mathrm{ref}})$. Note that the aircraft follows Euler angle commands $(\Phi_{\mathrm{ref}}, \Theta_{\mathrm{ref}})$. At $t = 76.7$ sec, the command-following errors in the Euler angles Φ, and Θ are 0.4 deg, and 0.2 deg, respectively. Figure 11 shows the performance of inner-loop RC-PI controllers, as well as, the gains $(K_{\mathrm{P}}, K_{\mathrm{I}})$, and the controllers output $(\dot{P}_{\mathrm{ref}}, \dot{Q}_{\mathrm{ref}}, \dot{R}_{\mathrm{ref}})$. At $t = 76.7$ sec, the command-following errors in the angular-velocity components P, Q, and R are 2.4 deg/sec, -12 deg/sec, and 16 deg/sec, respectively. Figure 12 shows the fixed-wing control inputs generated by the mixer.

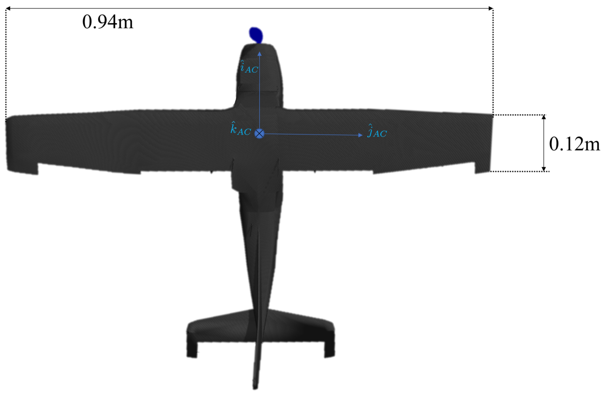

Figure 8. Gazebo fixed-wing aircraft model.

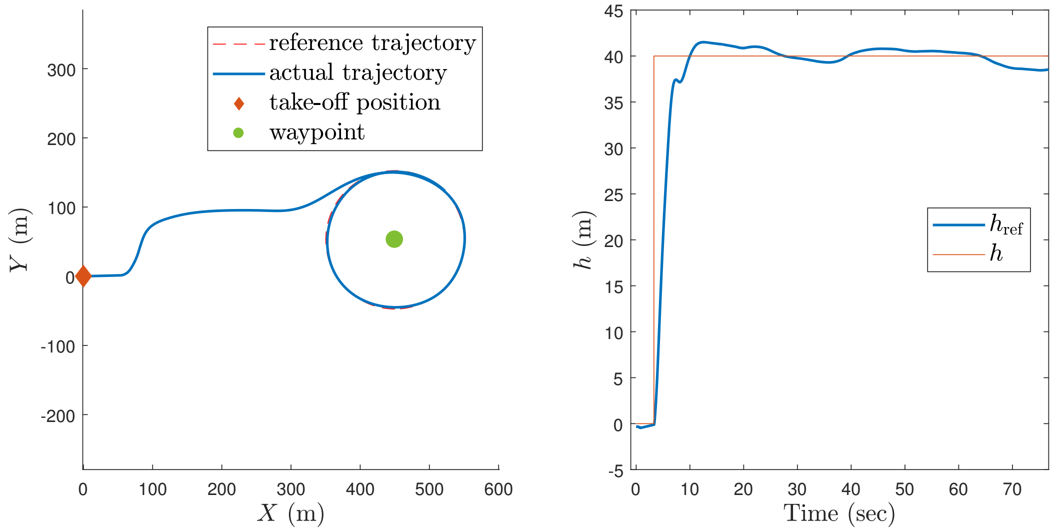

Figure 9. Trajectory of the fixed-wing aircraft. The command is to take off and then loiter around the commanded waypoint while maintaining the altitude at 40 m. The aircraft follows the commanded altitude, while loitering around the commanded waypoint.

V. Adaptive Control of VTOL Flight

The interface of PX4 with Gazebo for a VTOL aircraft is shown in Figure 13. The PX4 controller flies the aircraft in two different modes, namely, the quadcopter mode and the fixed-wing mode. For vertical takeoff and landing, the PX4 controller employs the quadcopter-mode controller and uses all four rotors; for cruise flight, it employs the fixed-wing-mode controller and uses the pusher, and elevons. During the transition from quadcopter mode to fixed-wing mode and vice versa, the PX4 controller switches between the quadcopter-mode controller and the fixed-wing-mode controller based on the airspeed V_{AC}. Comparing Figures 1, 7 and 13, note that the QC-mode and FW-mode controllers are the same as the QC and FW controllers discussed in Section III and Section IV, respectively.

Let T_{onset}, T_{end}, $V_{\mathrm{AC,TO}}$, and $V_{\mathrm{AC,TE}}$ denote the transition onset time, transition end time, airspeed at the beginning of transition, and airspeed at the end of transition, respectively. At the beginning of transition from QC to FW mode, the pusher starts, and then the pusher throttle-setting increases linearly until the VTOL aircraft reaches a pre-defined $V_{\mathrm{AC,TE}}$, after which the transition ends. This transition is modeled by

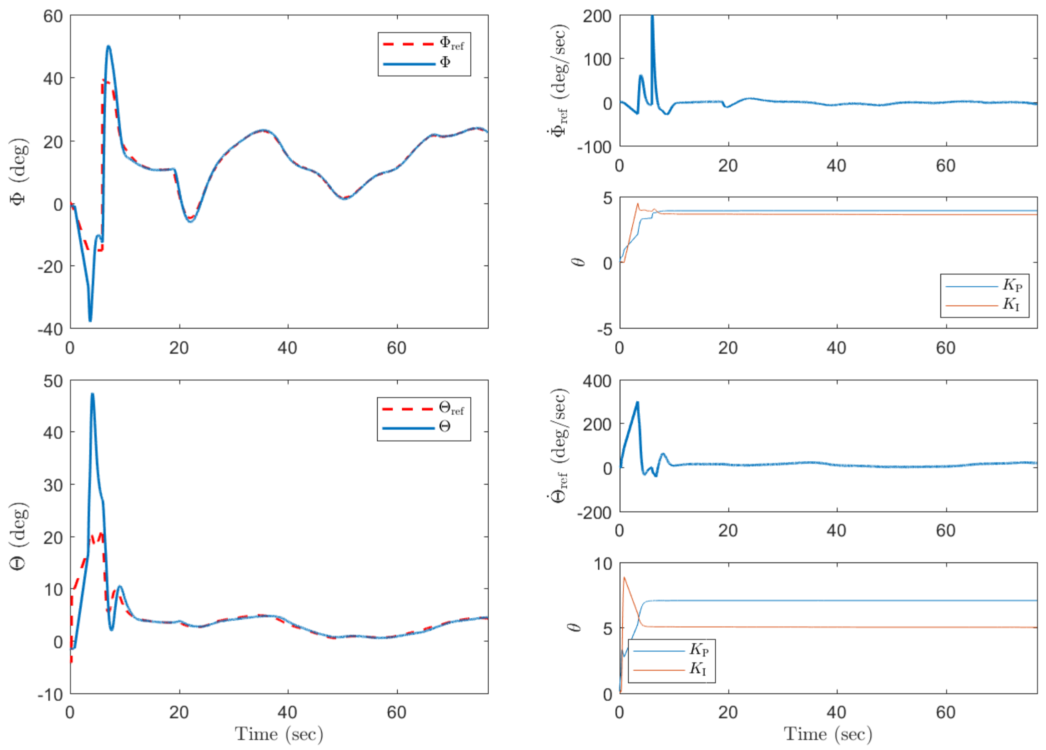

Figure 10. Performance of the fixed-wing middle-loop RC-PI controllers for Euler-angle commands. The controller coefficients θ, which are initialized at zero, converge for each RC-PI controller. At $t = 76.7$ sec, the command-following errors in the Euler angles Φ and Θ are 0.4 deg and 0.2 deg, respectively.

the switching gain $K_{\rm switch}$ (shown in Figure 13), which is given by

$$K_{\rm switch}(t) = \begin{cases} 1, & t < T_{\rm onset}, \\ 1 - \frac{V_{\rm AC} - V_{\rm AC,TO}}{V_{\rm AC,TE} - V_{\rm AC,TO}}, & T_{\rm onset} \le t \le T_{\rm end}, \\ 0, & t > T_{\rm end}. \end{cases} \tag{46}$$

Note that, if $K_{\rm switch} = 1$, then the VTOL aircraft flies using only the QC controller, whereas, if $K_{\rm switch} = 0$, then the VTOL aircraft flies using only the FW controller. Furthermore, note that, during the transition, $0 \le K_{\rm switch} \le 1$, and the VTOL aircraft flies using both the QC and FW controllers. To adaptively fly a VTOL aircraft using the PX4 control architecture (as shown in Figure 13), we replace the QC and FW mode controllers with retrospective cost adaptive QC and FW controllers as discussed in Section III and Section IV, respectively. Note that the RCAC tunings for the VTOL aircraft in quadcopter and fixed-wing mode are chosen to be identical to the tunings of the quadcopter and fixed-wing aircraft despite the differences in vehicle geometry.

A. Numerical Simulation

The Gazebo VTOL aircraft model that we use in the simulation is shown in Figure 14. The model has a wing span of 2.14 m and a wing area of 0.85 m^2. It has four rotors R_1, R_2, R_3, R_4, one pusher propeller T, and one set of differential elevons ev.

Figure 15 shows the reference waypoints $(X_{\rm ref}, Y_{\rm ref})$ and altitude $h_{\rm ref}$ for the mission planners. Note

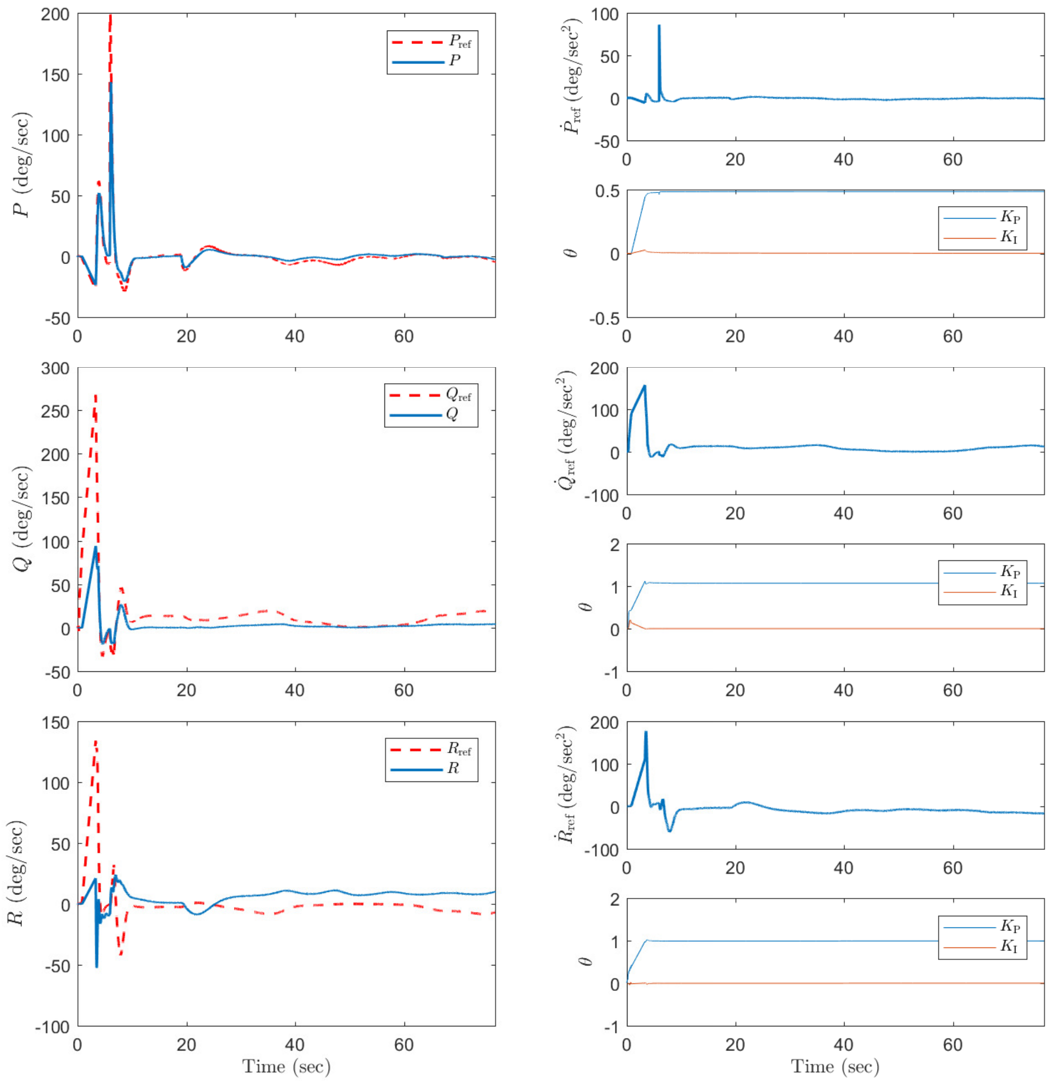

Figure 11. Performance of the fixed-wing inner-loop RC-PI for angular-velocity commands. The controller coefficients θ converge for each RC-PI controller. At $t = 76.7$ sec, the command-following errors in the angular-velocity components P, Q, and R are 2.4 deg/sec, -12 deg/sec, and 16 deg/sec, respectively.

that the command is to take off vertically in the quadcopter mode, then transition to the fixed-wing mode, and then loiter around the commanded waypoint while maintaining the altitude at 20 m. Note that the VTOL aircraft follows the commanded altitude with some transients at the transition, while loitering around the commanded waypoint.

Figure 16 shows the performance of middle-loop QC-mode and FW-mode controllers, as well as, the gains $(K_{\rm P}, K_{\rm I})$, and the controllers output $(\dot{\Phi}_{\rm ref}, \dot{\Theta}_{\rm ref}, \dot{\Psi}_{\rm ref})$. Note that the controller coefficients θ, which are initialized at zero, converge for for both the QC and FW mode controllers, and the VTOL aircraft follows Euler angle commands $(\Phi_{\rm ref}, \Theta_{\rm ref}, \Psi_{\rm ref})$ in both QC and FW modes. Since there is no yaw-angle control in the FW-mode controller, therefore, we do not plot Ψ and $\Psi_{\rm ref}$ after the transition. Before the transition, at $t = 51$ sec, the command-following errors in the Euler angles Φ, Θ, and Ψ are -0.52 deg, -0.93 deg, and

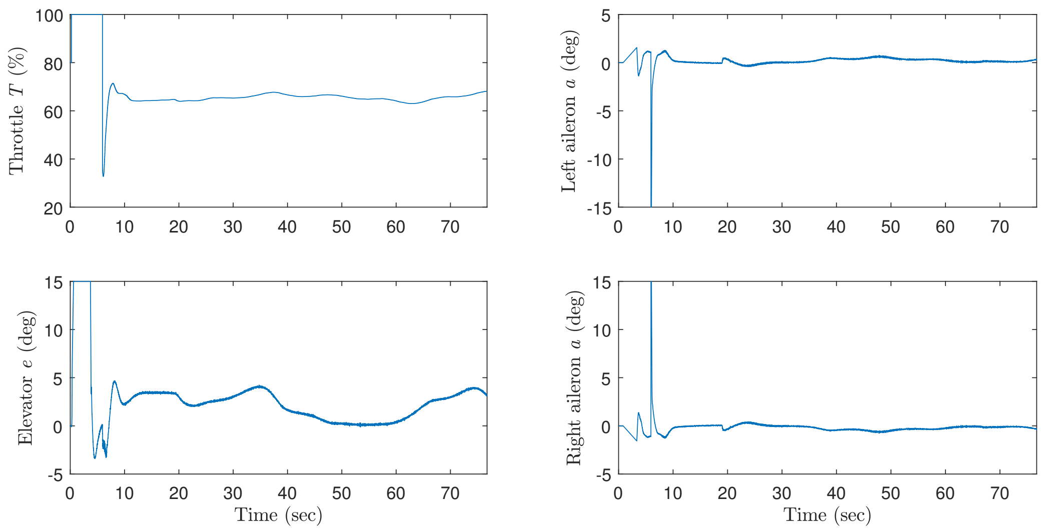

Figure 12. Fixed-wing aircraft control inputs corresponding to Figure 11.

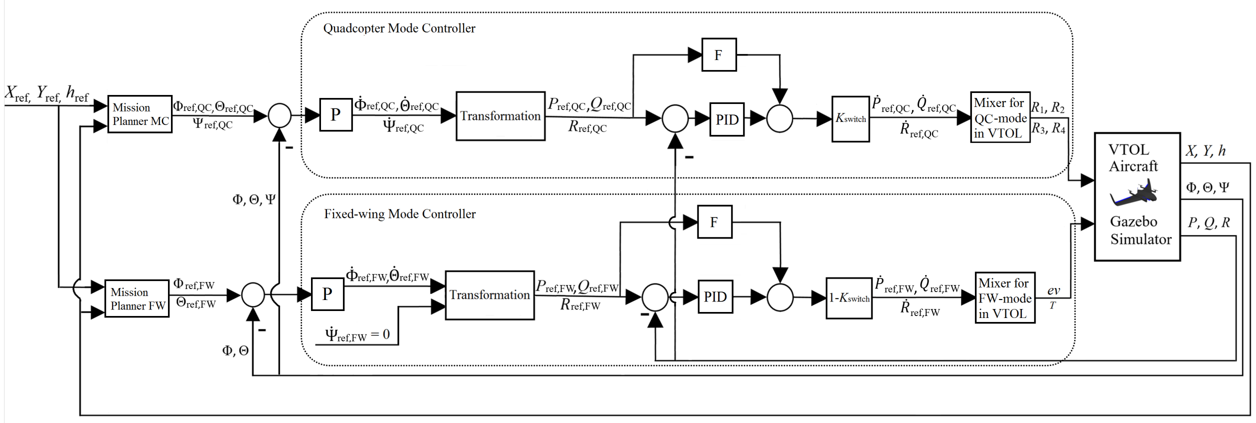

Figure 13. Pixhawk control architecture for mode transition of a VTOL aircraft in the Gazebo simulator.

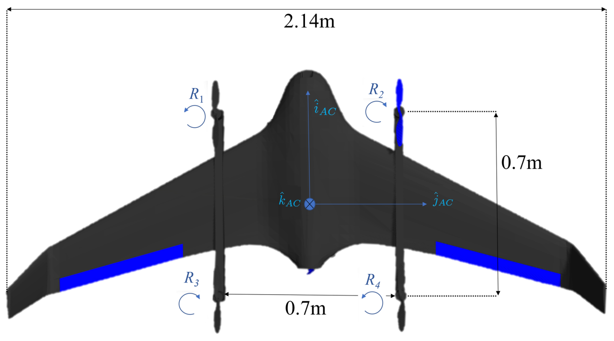

Figure 14. Gazebo VTOL aircraft model.

−1.3 deg, respectively. After the transition, at $t = 100$ sec, the command-following errors in the Euler angles Φ, and Θ are 0.2 deg, and −0.1 deg, respectively.

Figure 17 shows the performance of inner-loop QC-mode and FW-mode controllers, as well as, the gains $(K_{\rm P}, K_{\rm I},, K_{\rm D})$, and the controllers output $(\dot{P}_{\rm ref}, \dot{Q}_{\rm ref}, \dot{R}_{\rm ref})$. Before the transition, at $t = 51$ sec, the command-following errors in the angular-velocity components P, Q, and R are 2.8 deg/sec, 0.6 deg/sec, and -10 deg/sec, respectively. After the transition, at $t = 100$ sec, the command-following errors in the angular-velocity components P, Q, and R are 1.9 deg/sec, -10 deg/sec, and 14 deg/sec, respectively. Figures 18 and 19 show the VTOL aircraft control inputs generated by the QC-mixer and FW-mixer, respectively. Note that, during the transition, the magnitude of FW control inputs increase from their initial zero value, whereas, QC control inputs decrease to zero.

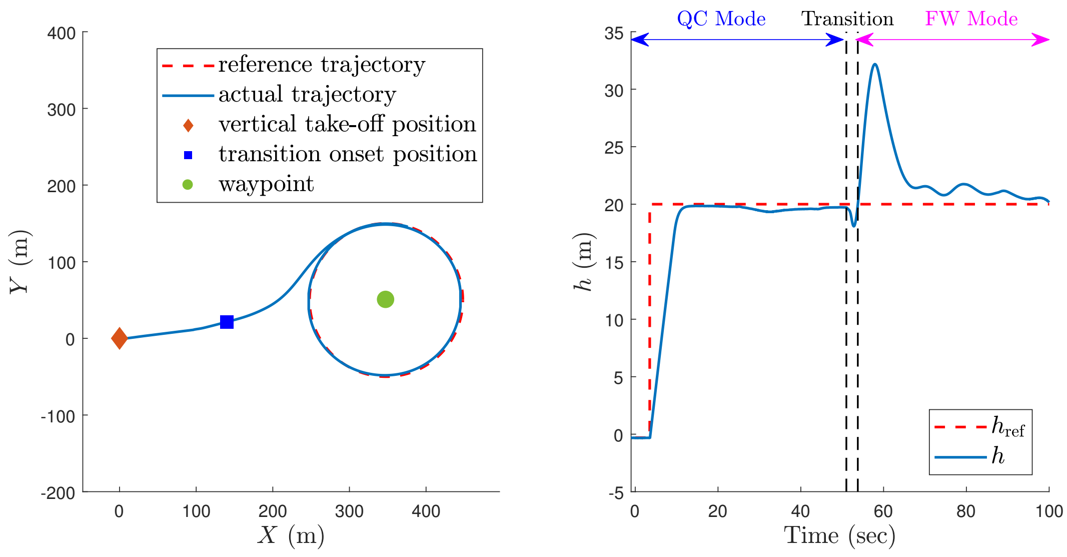

Figure 15. Trajectory of the VTOL aircraft. The command is to take off vertically in the quadcopter mode, then transition to the fixed-wing mode, and then loiter around the commanded waypoint while maintaining the altitude at 20 m. The VTOL aircraft follows the commanded altitude with some transients at the transition, while loitering around the commanded waypoint.

VI. Conclusions

This paper explored the feasibility of using retrospective cost adaptive control (RCAC) to adaptively tune the autopilot for an aircraft with minimal modeling information. For this purpose, RCAC was specialized to the case of PID control laws. To demonstrate the ability of RCAC to operate with limited modeling information, RCAC was initially applied to quadcopter and fixed-wing aircraft. Next, RCAC was applied to a third vehicle, namely, a VTOL aircraft, which can be operated in either quadcopter mode or fixed-wing mode. The geometry of the VTOL aircraft is significantly different from the quadcopter and fixed-wing aircraft. Despite these differences, RCAC was applied to the VTOL aircraft by using the filter and weightings used previously for the quadcopter and fixed-wing vehicles. By simulating various flight scenarios, it was shown that RCAC was able to control the VTOL aircraft in both the quadcopter and fixed-wing modes as well as during transition from quadcopter to fixed-wing flight. It should be stressed that, for all of these cases, RCAC was the only operational controller; no inner loop controller based on a nominal model was used. This ability shows that RCAC can control an aircraft with extremely limited modeling information.

Combined with the results of,[8] these results show that RCAC can provide a flexible and robust autopilot for aircraft that are either poorly modeled or subject to emergency conditions. The next phase of this work is to demonstrate the potential usefulness of RCAC for rapidly prototyping autopilots for aircraft that are not well modeled, either due to complex physics, emergency conditions, or experimental reconfiguration.

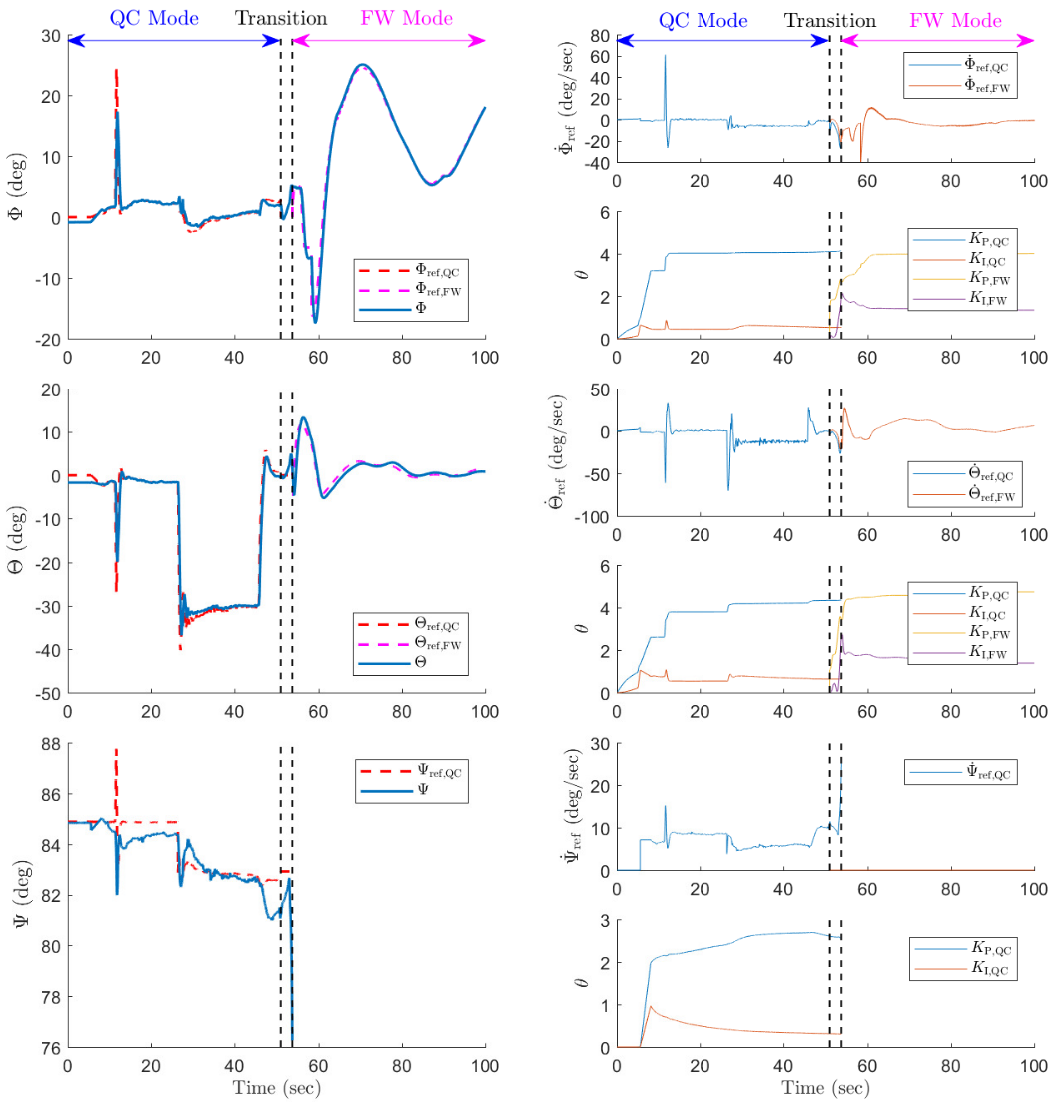

Figure 16. Performance of the VTOL middle-loop QC-mode and FW-mode controllers for Euler-angle commands. The controller coefficients θ, which are initialized at zero, converge for both the QC and FW mode controllers. Before the transition, at $t = 51$ sec, the command-following errors in the Euler angles Φ, Θ, and Ψ are -0.52 deg, -0.93 deg, and -1.3 deg, respectively. After the transition, at $t = 100$ sec, the command-following errors in the Euler angles Φ and Θ are 0.2 deg and -0.1 deg, respectively.

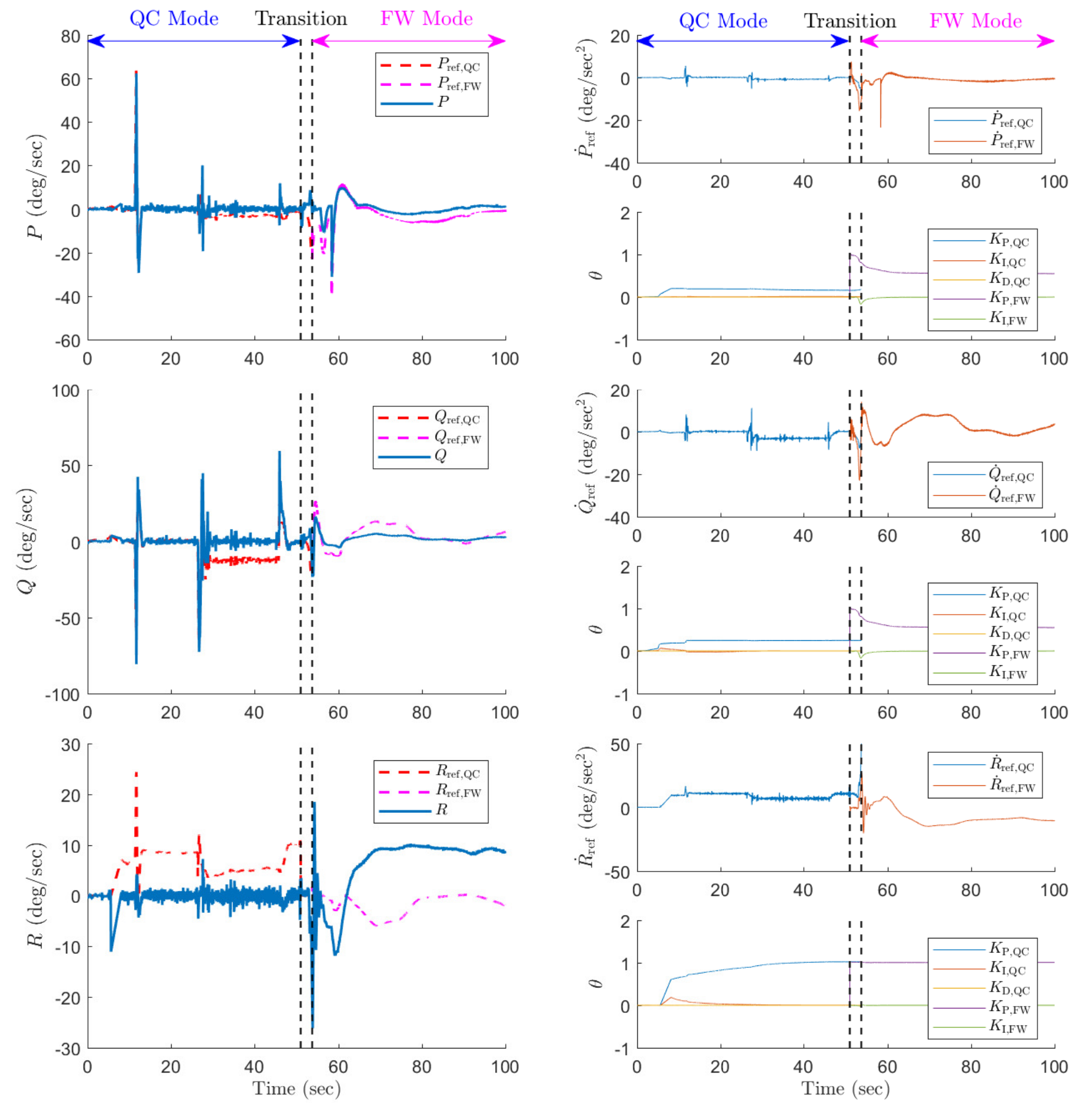

Figure 17. Performance of the fixed-wing inner-loop QC-mode and FW-mode controllers for angular-velocity commands. The controller coefficients θ, which are initialized at zero, converge for both the QC and FW mode controllers. Before the transition, at $t = 51$ sec, the command-following errors in the angular-velocity components P, Q, and R are 2.8 deg/sec, 0.6 deg/sec, and -10 deg/sec, respectively. After the transition, at $t = 100$ sec, the command-following errors in the angular-velocity components P, Q, and R are 1.9 deg/sec, -10 deg/sec, and 14 deg/sec, respectively.

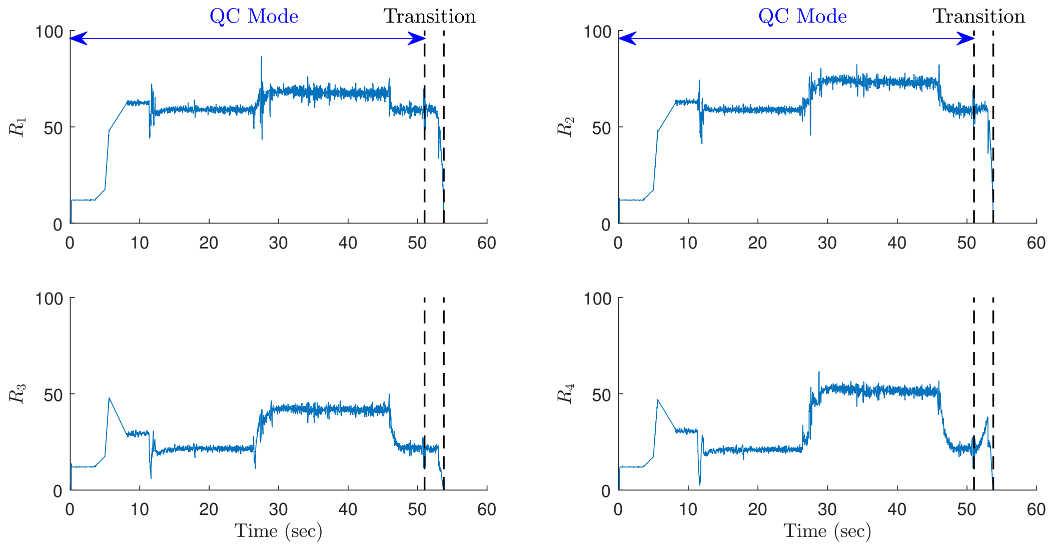

Figure 18. VTOL aircraft control inputs for the quadcopter mode. Note that, during the transition, the control inputs decrease to zero.

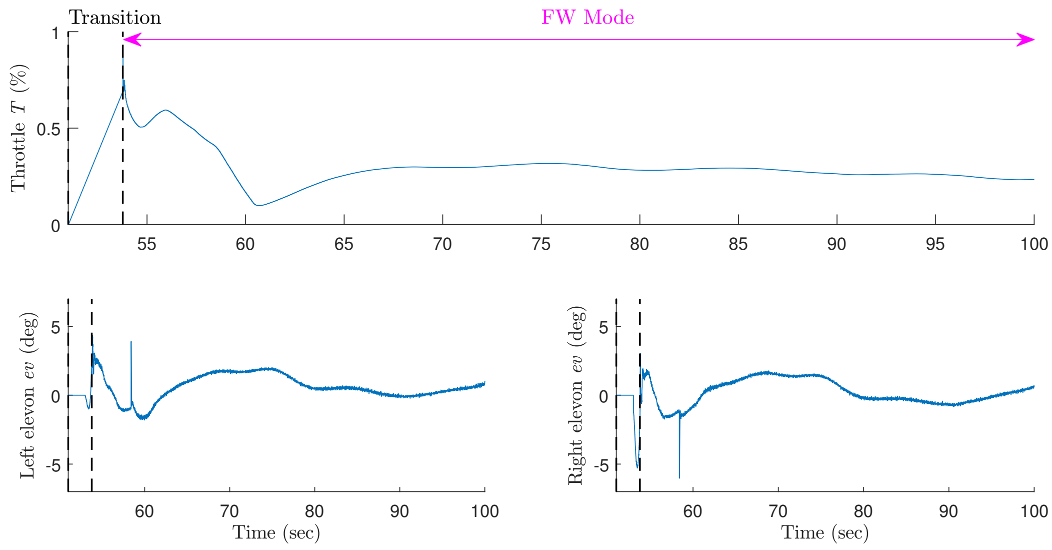

Figure 19. VTOL aircraft control inputs for the fixed-wing mode. Note that, during the transition, the magnitudes of the control inputs increase from their initial zero value.

References

[1]Z. T. Dydek, A. M. Annaswamy, and E. Lavretsky. Adaptive Control and the NASA X-15-3 Flight Revisited: Lessons Learned and Lyapunov-stability-based Design. *IEEE Contr. Sys. Mag.*, 30:32–48, 2010.

[2]J. S. Orr and C. J. Dennehy. Analysis of the X-15 Flight 3-65-97 Divergent Limit-Cycle Oscillation Lyapunov-stability-based Design. *AIAA J. Aircraft*, 54:135–148, 2017.

[3]N. Hovakimyan, C. Cao, E. Kharisov, E. Xargay, and I. M. Gregory. L_1 Adaptive Control for Safety-Critical Systems. *IEEE Contr. Sys. Mag.*, 31:54–104, 2011.

[4]T. Jordan, W. Langford, C. Belcastro, J. Foster, G. Shah, G. Howland, and R. Kidd. Development of a Dynamically Scaled Generic Transport Model Testbed for Flight Research Experiments. *AUVSI Unmanned Unlimited, Arlington, VA*, 2004.

[5]R. M. Bailey, R. W. Hostetler, K. N. Barnes, C. Belcastro, and C. Belcastro. Experimental Validation: Subscale Aircraft Ground Facilities and Integrated Test Capability. In *Proc. AIAA Guid. Nav. Cont. Conf.*, page 110, San Francisco, CA, 2005. AIAA-2005-6433.

[6]T. L. Jordan and R. M. Bailey. NASA Langleys AirSTAR Testbed: A Subscale Flight Test Capability for Flight Dynamics and Control System Experiments. In *Proc. AIAA Guid. Nav. Cont. Conf.*, pages 18–21, Honolulu, HI, 2008. AIAA-2008-6660.

[7]I. Cunningham, D. E. Cox, D. G. Murri, and S. E. Riddick. A Piloted Evaluation of Damage Accommodating Flight Control Using a Remotely Piloted Vehicle. In *Proc. AIAA Guid. Nav. Cont. Conf.*, Portland, OR, 2011. AIAA-2011-6451.

[8]A. Ansari and D. S. Bernstein. Retrospective Cost Adaptive Control of the Generic Transport Model Under Uncertainty and Failure. *Journal of Aerospace Information Systems*, 14(3):123–174, 2017.

[9]Y. Rahman, A. Xie, and D. S. Bernstein. Retrospective Cost Adaptive Control: Pole Placement, Frequency Response, and Connections with LQG Control. *IEEE Contr. Sys. Mag.*, 37:28–69, October 2017.

[10]R. Venugopal and D. S. Bernstein. Adaptive Disturbance Rejection Using ARMARKOV System Representations. *IEEE Trans. Contr. Sys. Tech.*, 8:257–269, 2000.

[11]M. A. Santillo and D. S. Bernstein. Adaptive Control Based on Retrospective Cost Optimization. *AIAA J. Guid. Contr. Dyn.*, 33:289–304, 2010.

[12]J. B. Hoagg and D. S. Bernstein. Retrospective Cost Model Reference Adaptive Control for Nonminimum-Phase Systems. *J. Guid. Contr. Dyn.*, 35:1767–1786, 2012.

[13]S. Dai, Z. Ren, and D. S. Bernstein. Adaptive Control of Nonminimum-Phase Systems Using Shifted Laurent Series. *Int. J. Contr.*, 90:409–422, 2017.

[14]A. Ansari and Dennis S Bernstein. Adaptive Control of an Aircraft with Uncertain Nonminimum-Phase Dynamics. In *Proc. Amer. Contr. Conf.*, pages 844–849, Chicago, IL, 2015.

[15]N. Koenig and A. Howard. Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator. In *Intelligent Robots and Systems Conference*, pages 2149–2154. IEEE, 2004.

[16]E. D. Sumer, J. B. Hoagg, and D. S. Bernstein. Broadband Disturbance Rejection Using Retrospective Cost Adaptive Control. In *Proc. IEEE Conf. Dec. Contr.*, pages 1–10, Fort Lauderdale, FL, October 2012.

[17]A. Ansari, A. Prach, and D. S. Bernstein. Adaptive Trim and Trajectory Following for a Tilt-Rotor Tricopter. In *American Control Conference (ACC), 2017*, pages 1109–1114. IEEE, 2017.

[18]A. Ansari, M. J. Yu, and D. S. Bernstein. Exploration and Mapping of an Unknown Flight Envelope. In *Proc. Dyn. Sys. Contr. Conf.*, pages 523–528, Los Angeles, CA, 2014.

[19]D. Mellinger and V. Kumar. Minimum Snap Trajectory Generation and Control for Quadrotors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2520–2525. IEEE, 2011.

[20]S. Park, J. Deyst, and J. P. How. A New Nonlinear Guidance Logic for Trajectory Tracking. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 16–19, 2004.