Flight Simulation and Control using the Julia Language

Conference Paper · January 2022

DOI: 10.2514/6.2022-2354

2 authors:

Umberto Saetti
University of Maryland
49 PUBLICATIONS **245** CITATIONS
SEE PROFILE

Joseph F. Horn
Pennsylvania State University
128 PUBLICATIONS **1,577** CITATIONS
SEE PROFILE

Flight Simulation and Control using the Julia Language

Umberto Saetti *
Auburn University, Auburn, AL 36849, USA

Joseph F. Horn †
Pennsylvania State University, University Park, PA, 16802, USA

The objective of this paper is to demonstrate the use of the Julia language for developing complex flight simulation models and performing flight control design. Three simulation models are developed: a simple helicopter model (J-SimpleHel), a higher-fidelity helicopter model (J-GenHel), and a fighter jet model (J-F16). These models are validated against flight test data, when available, and are used to predict the dynamic characteristics of the three aircraft in consideration. Model-following control laws are implemented for each aircraft to enhance the response characteristics.

I. Introduction

The Julia language (Ref. 1) is a recently-developed, open-source, and compiled programming language specifically intended for scientific computing. Julia takes advantage of modern techniques for executing dynamic languages efficiently, which is fundamental for scientific computing. Dynamic languages are considered highly productive but generally lacking in performance compared to compiled languages. Interpreted, two-tiered, high-level environments such as MATLAB® or Octave (Ref. 2) provide greatly increased convenience and productivity. However, compiled languages such as C and Fortran remain standard languages for computationally intensive problems. Julia, on the other hand, provides the performance of a statically compiled language while providing interactive dynamic behavior and productivity like Python, MATLAB® or Octave. Julia is based on a one-tiered approach such that it possible for much of Julia's library to be written in Julia itself, while also incorporating best-of-breed C and Fortran libraries. As such, Julia is particularly indicated for scientific computing, and could provide a convenient framework for complex flight dynamics simulations, where these simulations may include heavy aerodynamic or aeroelastic computations. One convenient feature of Julia is the support of direct calling of C and Fortran libraries without glue code. This is a particularly interesting feature in the aerospace engineering community as large amounts of diverse simulation models were developed in these two languages in the past, with examples ranging from the General Helicopter (GenHel) flight dynamics simulation model (Ref. 3) to the aeroacoustic solver PSU-WOPWOP (Ref. 4). As such, legacy simulation models can be readily integrated within novel simulations developed in Julia.

As such, the objective of this paper is to demonstrate the use of the Julia language for fixed-wing aircraft and rotorcraft flight simulation through the development of three simulation models: a simple helicopter model (J-SimpleHel), a higher-fidelity helicopter model (J-GenHel), and a fighter jet model (J-F16). Additionally, flight control laws will be implemented and tested in simulations for each of these models. As part of this scope, a Julia software package is developed that includes these three simulation models, and made freely available.

The paper begins with an overview of the simulation architecture, which includes mathematical and implementation aspects of the dynamic models, and trim, linearization, and model order reduction routines. This is followed by a summary of the three 6-degree-of-freedom simulation models and by a detailed explanation of a nonlinear dynamic inversion (NDI) flight control law that is applied to each of these models. The simulation models are validated against flight test data and other simulation models in the literature. Simulation results demonstrate that the NDI controller is able to provide desired stability and response characteristics.

*Assistant Professor, Department of Aerospace Engineering, Member AIAA.
†Professor, Department of Aerospace Engineering, Associate Fellow AIAA.

II. Simulation Architecture

A. Dynamic Models

The flight simulation models are developed as nonlinear time-invariant dynamical systems in first-order form:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right) \tag{1a}$$

$$\boldsymbol{y} = \boldsymbol{g}\left(\boldsymbol{x}, \boldsymbol{u}\right) \tag{1b}$$

where $\boldsymbol{x} \in \mathcal{R}^n$ is the state vector, $\boldsymbol{u} \in \mathcal{R}^m$ is the control input vector, and $\boldsymbol{y} \in \mathcal{R}^p$ is the output vector. In the software package, the functions containing the dynamics of each aircraft are named after the name of the model (*i.e.*, `SimpleHel.jl`, `GenHel.jl`, and `F16.jl`).

B. Linearization

This section describes the theoretical and implementation aspects of the linearization routine, `linearize.jl`. Consider the nonlinear system of Eq. (1) at equilibrium:

$$\dot{\boldsymbol{x}}_e = \boldsymbol{f}\left(\boldsymbol{x}_e, \boldsymbol{u}_e\right) \tag{2a}$$

$$\boldsymbol{y}_e = \boldsymbol{g}\left(\boldsymbol{x}_e, \boldsymbol{u}_e\right) \tag{2b}$$

where $\boldsymbol{x}_e$ and $\boldsymbol{u}_e$ are the trim state and control input vectors such that the state dynamics $\dot{\boldsymbol{x}}_e$ is constant. Consider the case where small disturbances are applied to the state, control input, and output vectors:

$$\boldsymbol{x} = \boldsymbol{x}_e + \Delta\boldsymbol{x} \tag{3a}$$

$$\boldsymbol{u} = \boldsymbol{u}_e + \Delta\boldsymbol{u} \tag{3b}$$

$$\boldsymbol{y} = \boldsymbol{y}_e + \Delta\boldsymbol{y} \tag{3c}$$

Then, a Taylor series expansion is performed on the state dynamics:

$$\boldsymbol{f}\left(\boldsymbol{x}_e + \Delta\boldsymbol{x}, \boldsymbol{u}_e + \Delta\boldsymbol{u}\right) = \boldsymbol{f}\left(\boldsymbol{x}_e, \boldsymbol{u}_e\right) + \frac{\partial \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} \Delta\boldsymbol{x} + \frac{\partial \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{u}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} \Delta\boldsymbol{u} + O\left(\Delta\boldsymbol{x}^2, \Delta\boldsymbol{u}^2\right) \tag{4}$$

By neglecting those terms with order higher than first, and by re-organizing one obtains:

$$\Delta\dot{\boldsymbol{x}} = \frac{\partial \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} \Delta\boldsymbol{x} + \frac{\partial \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{u}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} \Delta\boldsymbol{u} \tag{5}$$

where:

$$\frac{\partial \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{x}, \boldsymbol{u})}{\partial x_1} & \cdots & \frac{\partial f_1(\boldsymbol{x}, \boldsymbol{u})}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n(\boldsymbol{x}, \boldsymbol{u})}{\partial x_1} & \cdots & \frac{\partial f_n(\boldsymbol{x}, \boldsymbol{u})}{\partial x_n} \end{bmatrix}_{\boldsymbol{x}_e, \boldsymbol{u}_e} = \boldsymbol{A} \tag{6a}$$

$$\frac{\partial \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{u}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{x}, \boldsymbol{u})}{\partial u_1} & \cdots & \frac{\partial f_1(\boldsymbol{x}, \boldsymbol{u})}{\partial u_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n(\boldsymbol{x}, \boldsymbol{u})}{\partial u_1} & \cdots & \frac{\partial f_n(\boldsymbol{x}, \boldsymbol{u})}{\partial u_m} \end{bmatrix}_{\boldsymbol{x}_e, \boldsymbol{u}_e} = \boldsymbol{B} \tag{6b}$$

Similarly, a Taylor series expansion is performed on the output equation:

$$\boldsymbol{g}\left(\boldsymbol{x}_e + \Delta\boldsymbol{x}, \boldsymbol{u}_e + \Delta\boldsymbol{u}\right) = \boldsymbol{g}\left(\boldsymbol{x}_e, \boldsymbol{u}_e\right) + \frac{\partial \boldsymbol{g}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} \Delta\boldsymbol{x} + \frac{\partial \boldsymbol{g}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{u}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} \Delta\boldsymbol{u} + O\left(\Delta\boldsymbol{x}^2, \Delta\boldsymbol{u}^2\right) \tag{7}$$

By neglecting those terms with order higher than first, and by re-organizing one obtains:

$$\Delta\boldsymbol{y} = \frac{\partial \boldsymbol{g}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} \Delta\boldsymbol{x} + \frac{\partial \boldsymbol{g}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{u}}\bigg|_{\boldsymbol{x}_e, \boldsymbol{u}_e} \Delta\boldsymbol{u} \tag{8}$$

where:

$$\left.\frac{\partial \boldsymbol{g}\left(\boldsymbol{x},\boldsymbol{u}\right)}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_e,\boldsymbol{u}_e} = \left[\begin{array}{ccc} \frac{\partial g_1(\boldsymbol{x}_e,\boldsymbol{u}_e)}{\partial x_1} & \cdots & \frac{\partial g_1(\boldsymbol{x}_e,\boldsymbol{u}_e)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_p(\boldsymbol{x}_e,\boldsymbol{u}_e)}{\partial x_1} & \cdots & \frac{\partial g_p(\boldsymbol{x}_e,\boldsymbol{u}_e)}{\partial x_n} \end{array}\right]_{\boldsymbol{x}_e,\boldsymbol{u}_e} = \boldsymbol{C} \tag{9a}$$

$$\left.\frac{\partial \boldsymbol{g}\left(\boldsymbol{x},\boldsymbol{u}\right)}{\partial \boldsymbol{u}}\right|_{\boldsymbol{x}_e,\boldsymbol{u}_e} = \left[\begin{array}{ccc} \frac{\partial g_1(\boldsymbol{x}_e,\boldsymbol{u}_e)}{\partial u_1} & \cdots & \frac{\partial g_1(\boldsymbol{x}_e,\boldsymbol{u}_e)}{\partial u_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_p(\boldsymbol{x}_e,\boldsymbol{u}_e)}{\partial u_1} & \cdots & \frac{\partial g_p(\boldsymbol{x}_e,\boldsymbol{u}_e)}{\partial u_m} \end{array}\right]_{\boldsymbol{x}_e,\boldsymbol{u}_e} = \boldsymbol{D} \tag{9b}$$

Then, the linearized system can be written as:

$$\Delta\dot{\boldsymbol{x}} = \boldsymbol{A}\Delta\boldsymbol{x} + \boldsymbol{B}\Delta\boldsymbol{u} \tag{10a}$$

$$\Delta\boldsymbol{y} = \boldsymbol{C}\Delta\boldsymbol{x} + \boldsymbol{D}\Delta\boldsymbol{u} \tag{10b}$$

In practice, the partial derivatives are computed numerically using a central difference scheme. For instance, the elements of the $\boldsymbol{A}$ matrix are found as follows:

$$A_{ij} = \frac{\partial f_i\left(\boldsymbol{x},\boldsymbol{u}\right)}{\partial x_j} \approx \frac{f_i\left(\boldsymbol{x}_e + \Delta\boldsymbol{x}_j,\boldsymbol{u}\right) - f_i\left(\boldsymbol{x}_e - \Delta\boldsymbol{x}_j,\boldsymbol{u}\right)}{2\Delta x_j} \tag{11}$$

where the perturbations are performed one by one for each state, such that:

$$\Delta\boldsymbol{x}_j^T = \left[x_{e_1} \ \cdots \ x_{e_j} + \Delta x_j \ \cdots \ x_{e_n}\right] \tag{12}$$

A similar procedure is adopted for finding the elements of the $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{D}$ coefficient matrices.

C. Trimming Algorithm

This section describes the trimming routine, `trimmer.jl`. In fixed-wing aircraft flight dynamics and control, it is desirable to be able to obtain the trim condition of a simulated aircraft so that linearized models of the dynamics can be derived at that condition, and used for stability analysis and flight control design.

The trim problem can be stated as follows. Given a prescribed function in time $\boldsymbol{x}_e$, the goal is to solve for a subset of the state vector $\boldsymbol{x}_s \in \mathcal{R}^l$, with $l \leq n$, and for a subset of the control vector $\boldsymbol{u}_s \in \mathcal{R}^o$, with $o \leq m$, subject to Eq. (2a). The trim variables for this problem are thus given by augmenting the subset of the state vector with the control vector:

$$\boldsymbol{\Theta}^T = \left[\boldsymbol{x}_s^T \ \boldsymbol{u}_s^T\right] \tag{13}$$

which leads to $l + o$ trim variables. The constraints are given by the n state derivatives, which can be set to arbitrary quantities known as trim targets:

$$\dot{\boldsymbol{x}}_e^T = [\dot{x}_1 \ \dot{x}_2 \ \cdots \ \dot{x}_n] \tag{14}$$

Then, the trim problem to be solved is given by:

$$\boldsymbol{e}\left(\Theta\right) = \dot{\boldsymbol{x}}_e - \boldsymbol{f}\left(\boldsymbol{x}_s,\boldsymbol{u}_s\right) = \boldsymbol{0} \tag{15}$$

where $\boldsymbol{e}\left(\Theta\right)$ is the error vector. It is clear that, to make the problem square such that a unique solution exists, the number of trim variables must be equal to that of the constraints (*i.e.*, $l + o = n$). It follows that $n - l - o$ conditions still need to be specified. Note that if the m inputs are given and the corresponding equilibrium solution is required, then the problem in consideration becomes a closed system as $o = 0$ and $l = n$. On the other hand, in the case where one or more (possibly all) of the m control inputs are unknown, then each input is used to ensure some desired condition. For aerospace vehicles such as conventional fixed-wing aircraft and helicopters, for which the vehicle dynamics are invariant with respect to position and heading (see, *e.g.*, Ref. 5), the position and heading can be arbitrarily assigned and removed from the vector of unknowns. Since these vehicles typically employ control about four axes (*i.e.*, roll, pitch, yaw, and heave) leading to four control inputs, fixing the three components of the zeroth harmonic of the position (x, y, z) and heading (ψ) at equilibrium leads to a square problem.

In practice this problem is solved iteratively, in that a candidate solution is refined over a series of computational steps until a convergence criteria is reached. Consider the candidate solution at iteration k of the algorithm, $\boldsymbol{\Theta}_k$. One iteration of the algorithm begins with evaluating the cost function in Eq. (15). If the infinity norm of the cost function is less than an arbitrary tolerance (*i.e.*, $\|\boldsymbol{e}_k\|_\infty$), then a solution is found. If not, the algorithm proceeds with linearizing the system about the candidate solution. More specifically, the function `linearize.jl` is used to compute the system and control matrices at iteration k (*i.e.*, $\boldsymbol{A}_k$ and $\boldsymbol{B}_k$). These matrices are used to define the Jacobian matrix of the trimming algorithm:

$$\boldsymbol{J}_k = [\boldsymbol{A}_k \; \boldsymbol{B}_k] \tag{16}$$

where $\boldsymbol{J}_k \in \mathbb{R}^{n \times (n+m)}$. The m columns corresponding to those states and/or controls that are specified are truncated from the Jacobian matrix, leading to a modified Jacobian matrix $\hat{\boldsymbol{J}}_k \in \mathbb{R}^{n \times n}$. Next, the solution is updated using the Newton-Raphson method (Ref. 6):

$$\hat{\boldsymbol{\Theta}}_{k+1} = \boldsymbol{\Theta}_k - \hat{\boldsymbol{J}}_k^{-1} \boldsymbol{e}_k \tag{17}$$

The next iteration of the algorithm then proceeds with this new candidate solution until the stopping criteria is met.

D. Model Order Reduction

This section explains the model order reduction routine, `modred.jl`. Because the measurement or estimation of those states with dynamics faster than the rigid-body dynamics (*e.g.*, flapping or inflow states) is usually challenging or impractical, reduced-order models are be obtained to ease control system design. Ideally, these reduced-order models do not include the parasitic dynamics but still retain part of the high-order response characteristics. This can be achieved through residualization, a portion of singular perturbation theory that pertains to linear time-invariant systems (Ref. 7). Assuming that one or more states of the system have stable dynamics which are faster than those of the remaining states, the state vector is partitioned into fast and slow components:

$$\Delta \boldsymbol{x}^T = \left[\Delta \boldsymbol{x}_\mathrm{s}^T \; \Delta \boldsymbol{x}_\mathrm{f}^T\right] \tag{18}$$

Hereafter, the notation is simplified by dropping the Δ in front of the linearized perturbation state and control vectors while keeping in mind that these vectors represent perturbations from equilibrium. Then, the system in Eq. (10a) can be re-written as:

$$\begin{bmatrix} \dot{\boldsymbol{x}}_\mathrm{s} \\ \dot{\boldsymbol{x}}_\mathrm{f} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_\mathrm{s} & \boldsymbol{A}_\mathrm{sf} \\ \boldsymbol{A}_\mathrm{fs} & \boldsymbol{A}_\mathrm{f} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_\mathrm{s} \\ \boldsymbol{x}_\mathrm{f} \end{bmatrix} + \begin{bmatrix} \boldsymbol{B}_\mathrm{s} \\ \boldsymbol{B}_\mathrm{f} \end{bmatrix} \boldsymbol{u} \tag{19}$$

By neglecting the dynamics of the fast states (*i.e.*, $\dot{\boldsymbol{x}}_\mathrm{f} = 0$) and performing a few algebraic manipulations, it can be shown (Ref. 8) that the equations for a reduced-order system with the state vector comprised of the slow states is :

$$\dot{\boldsymbol{X}}_\mathrm{s} = \hat{\boldsymbol{A}} \boldsymbol{x}_\mathrm{s} + \hat{\boldsymbol{B}} \boldsymbol{u} \tag{20}$$

where:

$$\hat{\boldsymbol{A}} = \boldsymbol{A}_\mathrm{s} - \boldsymbol{A}_\mathrm{sf} \boldsymbol{A}_\mathrm{f}^{-1} \boldsymbol{A}_\mathrm{fs} \tag{21a}$$

$$\hat{\boldsymbol{B}} = \boldsymbol{B}_\mathrm{s} - \boldsymbol{A}_\mathrm{sf} \boldsymbol{A}_\mathrm{f}^{-1} \boldsymbol{B}_\mathrm{f} \tag{21b}$$

III. Simulation Models

A. Simple Helicopter Model

The first flight dynamic model included in the software package is J-SimpleHel, a simple helicopter model based on Ref. 9 representative of a utility helicopter similar to a UH-60. The model contains a 6-degree-of-freedom nonlinear rigid-body dynamic model of the fuselage, a quasi-static model of the blade flapping, and uses static aerodynamic models for fuselage, main and tail rotor inflows, and empennage. The state vector is:

$$\boldsymbol{x}^T = [u \; v \; w \; p \; q \; r \; \phi \; \theta \; \psi \; x \; y \; z] \tag{22}$$

where:

$u,\ v,\ w$ are the longitudinal, lateral, and vertical velocities in the body-fixed frame,
$p,\ q,\ r$ are the roll, pitch, and yaw rates,
$\phi,\ \theta,\ \psi$ are the Euler angles, and
$x,\ y,\ z$ are the positions in the North-East-Down (NED) frame.
The control vector is:

$$\boldsymbol{u}^T = \begin{bmatrix} \delta_{\text{lat}} & \delta_{\text{lon}} & \delta_{\text{col}} & \delta_{\text{ped}} \end{bmatrix} \tag{23}$$

where δ_{lat} and δ_{lon} are the lateral and longitudinal cyclic inputs, δ_{col} is the collective input, and δ_{ped} is the pedal input. Because this helicopter model is relatively simple, a relatively small amount of parameters is required to fully describe the helicopter dynamics. As such, this model can easily adapted to any conventional helicopter platform (*i.e.*, conventional main and tail rotor configurations) and is intended for preliminary stability and performance analyses.

B. High-Fidelity Helicopter Model

The second flight dynamic model included in the software package is J-GenHel, a more complex helicopter model based on the General Helicopter (GenHel) flight dynamics simulation model (Ref. 3). This model is representative of a utility helicopter similar to a UH-60. The model contains a 6-DoF rigid-body dynamic model of the fuselage, nonlinear aerodynamic lookup tables for the fuselage, rotor blades, and empennage, rigid flap and lead-lag rotor blade dynamics, a three-state Pitt-Peters inflow model (Ref. 10), and a Bailey tail rotor model (Ref. 11). The state vector is:

$$\boldsymbol{x}^T = \begin{bmatrix} u\ v\ w\ p\ q\ r\ \phi\ \theta\ \psi\ x\ y\ z\ \beta_0\ \beta_{1c}\ \beta_{1s}\ \beta_{0D}\ \dot{\beta}_0\ \dot{\beta}_{1c}\ \dot{\beta}_{1s}\ \dot{\beta}_{0D}\ \zeta_0\ \zeta_{1c}\ \zeta_{1s}\ \zeta_{0D}\ \dot{\zeta}_0\ \dot{\zeta}_{1c}\ \dot{\zeta}_{1s}\ \dot{\zeta}_{0D}\ \lambda_0\ \lambda_{1c}\ \lambda_{1s}\ \lambda_{0T}\ \psi_{\text{MR}} \end{bmatrix} \tag{24}$$

where:
$u,\ v,\ w$ are the longitudinal, lateral, and vertical velocities in the body-fixed frame,
$p,\ q,\ r$ are the roll, pitch, and yaw rates,
$\phi,\ \theta,\ \psi$ are the Euler angles,
$x,\ y,\ z$ are the positions in the North-East-Down (NED) frame,
$\beta_0,\ \beta_{1c},\ \beta_{1s},\ \beta_{0D}$ are the flapping angles in multi-blade coordinates,
$\zeta_0,\ \zeta_{1c},\ \zeta_{1s},\ \zeta_{0D}$ are the lead-lag angles in multi-blade coordinates,
$\lambda_0,\ \lambda_{1c},\ \lambda_{1s}$, are the main rotor induced inflow ratio harmonics,
λ_{0T} is the tail rotor induced inflow ratio, and
ψ_{MR} is the azimuth angle of a reference main rotor blade.

In the simulation model, these states are partitioned into fuselage, rotor, and tail rotor states as follows:

$$\boldsymbol{x}_{\text{F}}^T = \begin{bmatrix} u\ v\ w\ p\ q\ r\ \phi\ \theta\ \psi\ x\ y\ z \end{bmatrix} \tag{25a}$$

$$\boldsymbol{x}_{\text{R}}^T = \begin{bmatrix} \beta_0\ \beta_{1c}\ \beta_{1s}\ \beta_{0D}\ \dot{\beta}_0\ \dot{\beta}_{1c}\ \dot{\beta}_{1s}\ \dot{\beta}_{0D}\ \zeta_0\ \zeta_{1c}\ \zeta_{1s}\ \zeta_{0D}\ \dot{\zeta}_0\ \dot{\zeta}_{1c}\ \dot{\zeta}_{1s}\ \dot{\zeta}_{0D}\ \lambda_0\ \lambda_{1c}\ \lambda_{1s}\ \psi_{\text{MR}} \end{bmatrix} \tag{25b}$$

$$\boldsymbol{x}_{\text{TR}} = \begin{bmatrix} \lambda_{0T} \end{bmatrix} \tag{25c}$$

$$\tag{25d}$$

The control vector is:

$$\boldsymbol{u}^T = \begin{bmatrix} \delta_{\text{lat}} & \delta_{\text{lon}} & \delta_{\text{col}} & \delta_{\text{ped}} \end{bmatrix} \tag{26}$$

where δ_{lat} and δ_{lon} are the lateral and longitudinal cyclic inputs, δ_{col} is the collective input, and δ_{ped} is the pedal input.

C. Fighter Jet Model

The third model included in the software package is J-F16, a nonlinear simulation model of the rigid-body dynamics of an F-16. The model is taken from Ref. 12, 13. The state vector is:

$$\boldsymbol{x}^T = \begin{bmatrix} V_T\ \alpha\ \beta\ \phi\ \theta\ \psi\ p\ q\ r\ x\ y\ z\ P_a \end{bmatrix} \tag{27}$$

where:
V_T is the total speed,
α is the angle of attack,
α is the sideslip angle,

$\phi,\ \theta,\ \psi$ are the Euler angles,
$p,\ q,\ r$ are the roll, pitch, and yaw rates,
$x,\ y,\ z$ are the positions in the North-East-Down (NED) frame, and
P_a is the engine power.

The control vector is:

$$\boldsymbol{u}^T = [\delta_t\ \delta_a\ \delta_e\ \delta_r] \tag{28}$$

where δ_t is the throttle, δ_a is the aileron deflection, δ_e is the elevator deflection, and δ_r is the rudder deflection.

IV. Flight Control Design

A. Control Architecture

The architecture chosen for the autonomous flight control law is Dynamic Inversion (DI), a popular model-following scheme among aircraft and rotorcraft manufacturers, and within the aerospace flight controls community in general. Application of DI control laws to rotorcraft can be found in, *e.g.*, Refs. 8, 14–19. A key aspect of DI is the reliance on model inversion to cancel the plant dynamics and track a desired reference model. One convenient feature of DI is that it inverts the plant model in its feedback linearization loop, which, compared to other more conventional model-following control strategies such as Explicit Model Following (EMF), eliminates the need for gain scheduling. However, the plant model used for feedback linearization still needs to be scheduled with the flight condition. A generic DI controller as applied to a linear system is shown in Fig. 1. The key components are a command model (also known as commend model or reference model) that specifies desired response to pilot commands, a feedback compensation on the tracking error, and an inner feedback loop that achieves model inversion (also known as feedback linearization loop).

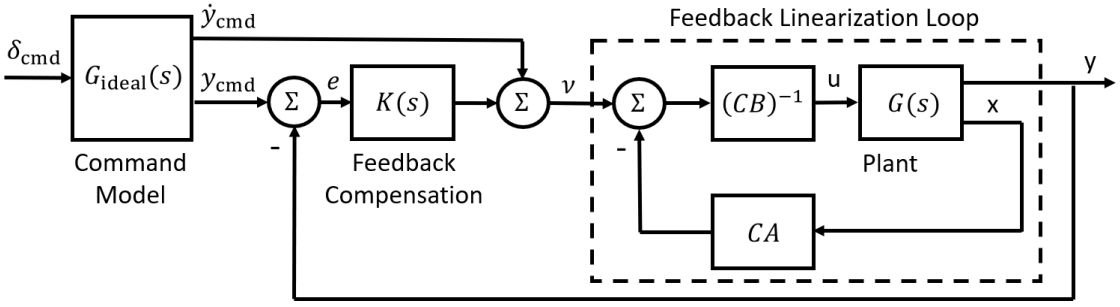


Fig. 1 DI controller as applied to a linear system.

A single-loop Dynamic Inversion (DI) control law largely based on Refs. 8, 14, 15 is designed to enhance the stability and response characteristics of the aircraft. The control law achieves stability, disturbance rejection, and desired response characteristics about the roll, pitch, and yaw. The desired response type is Rate Command / Attitude Hold (RCAH).

B. Linear Models

The very first step toward the development of a DI flight control law is to obtain linear models representative of the aircraft dynamics across the flight conditions of interest. For this reason, linear models are derived by trimming the simulation models at incremental speeds V in forward level flight and subsequently linearizing about each trim condition:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}(V)\boldsymbol{x} + \boldsymbol{B}(V)\boldsymbol{u} \tag{29}$$

where the coefficient matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are functions of the total speed of the aircraft, V.

The rotorcraft models are trimmed with zero sideslip (*i.e.*, $\psi = \arctan(V_y/V_x)$, where V_x and V_y are the desired longitudinal and lateral speeds in heading frame) for speeds below 60 kts, and with zero bank angle (*i.e.*, $\phi = 0$) for speeds above 60 kts. As such, for the simple helicopter model, the trim variables are chosen as the following set of

states and control inputs:

$$\boldsymbol{\Theta}^T = \begin{cases} \begin{bmatrix} u\ v\ w\ p\ q\ r\ \phi\ \theta\ \delta_{\text{lat}}\ \delta_{\text{lon}}\ \delta_{\text{col}}\ \delta_{\text{ped}} \end{bmatrix} & V < 60 \text{ kts} \\ \begin{bmatrix} u\ v\ w\ p\ q\ r\ \theta\ \psi\ \delta_{\text{lat}}\ \delta_{\text{lon}}\ \delta_{\text{col}}\ \delta_{\text{ped}} \end{bmatrix} & V \geq 60 \text{ kts} \end{cases} \tag{30}$$

whereas the trim targets are:

$$\dot{\boldsymbol{x}}_e^T = \begin{bmatrix} \dot{u}\ \dot{v}\ \dot{w}\ \dot{p}\ \dot{q}\ \dot{r}\ \dot{\phi}\ \dot{\theta}\ \dot{\psi}\ \dot{x}\ \dot{y}\ \dot{z} \end{bmatrix} \tag{31}$$

For the more complex helicopter model, the trim variables are chosen as:

$$\boldsymbol{\Theta}^T = \begin{cases} \begin{bmatrix} u\ v\ w\ p\ q\ r\ \phi\ \theta\ \boldsymbol{x}_{\text{R}}^T\ \boldsymbol{x}_{\text{TR}}^T\ \delta_{\text{lat}}\ \delta_{\text{lon}}\ \delta_{\text{col}}\ \delta_{\text{ped}} \end{bmatrix} & V < 60 \text{ kts} \\ \begin{bmatrix} u\ v\ w\ p\ q\ r\ \theta\ \psi\ \boldsymbol{x}_{\text{R}}^T\ \boldsymbol{x}_{\text{TR}}^T\ \delta_{\text{lat}}\ \delta_{\text{lon}}\ \delta_{\text{col}}\ \delta_{\text{ped}} \end{bmatrix} & V \geq 60 \text{ kts} \end{cases} \tag{32}$$

whereas the trim targets are:

$$\dot{\boldsymbol{x}}_e^T = \begin{bmatrix} \dot{\boldsymbol{x}}_{\text{F}}^T\ \dot{\boldsymbol{x}}_{\text{R}}^T\ \dot{\boldsymbol{x}}_{\text{TR}}^T \end{bmatrix} \tag{33}$$

All trim targets are set to zero except the derivative of the longitudinal position in the heading frame, which is set equal to the desired longitudinal speed (*i.e.*, $\dot{x} = V$). Additionally, in J-GenHel, the derivative of the main rotor azimuth angle of a reference blade is set equal to the main rotor angular speed (*i.e.*, $\dot{\psi}_{\text{MR}} = \Omega$). Then, for the more complex helicopter model, the order of the linearized models is reduced by means of residualization. This is achieved by choosing the slow states as the rigid-body states, and the fast states as the remaining states:

$$\boldsymbol{x}_{\text{s}} = \boldsymbol{x}_{\text{F}} \tag{34a}$$

$$\boldsymbol{x}_{\text{f}}^T = \begin{bmatrix} \boldsymbol{x}_{\text{R}}^T\ \boldsymbol{x}_{\text{TR}}^T \end{bmatrix} \tag{34b}$$

$$\tag{34c}$$

The trim variables chosen for the fighter jet model are:

$$\boldsymbol{\Theta}^T = [V_T\ \alpha\ \beta\ \phi\ \theta\ \psi\ p\ q\ r\ P_a\ \delta_t\ \delta_a\ \delta_e\ \delta_r] \tag{35}$$

whereas the trim targets are:

$$\dot{\boldsymbol{x}}_e^T = \begin{bmatrix} \dot{V}_T\ \dot{\alpha}\ \dot{\beta}\ \dot{\phi}\ \dot{\theta}\ \dot{\psi}\ \dot{p}\ \dot{q}\ \dot{r}\ \dot{P}_a\ \dot{x}\ \dot{y}\ \dot{z}\ a_y \end{bmatrix} \tag{36}$$

where a_y is the lateral acceleration in body axes.

C. Flight Control Law

To model the single-loop DI controller, a modified state vector is defined such that the states that are fed back to the controller are the angular rates:

$$\hat{\boldsymbol{x}}^T = [p\ q\ r] \tag{37}$$

along with modified system and control matrices $\hat{\boldsymbol{A}}(V)$ and $\hat{\boldsymbol{B}}(V)$. These modified matrices are found by truncating those rows and columns of matrices $\boldsymbol{A}(V)$ and $\boldsymbol{B}(V)$ corresponding to the states omitted in $\hat{\boldsymbol{x}}$. Because only three control axes are controlled (*i.e.*, roll, pitch, and yaw), then a modified control input vector is defined as well. The modified control input vector for the rotorcraft models is:

$$\hat{\boldsymbol{u}}^T = \begin{bmatrix} \delta_{\text{lat}}\ \delta_{\text{lon}}\ \delta_{\text{ped}} \end{bmatrix} \tag{38}$$

such that the heave axis, and thus the collective stick, is left open loop. The modified control input vector for the fighter jet model is:

$$\hat{\boldsymbol{u}}^T = [\delta_a\ \delta_e\ \delta_r] \tag{39}$$

Additionally, the following output vector is defined, corresponding to the controlled variables of the nonlinear system (*i.e.*, the aircraft dynamics):

$$\boldsymbol{y}^T = [p\ q\ r] \tag{40}$$

The output matrix that relates the state vector to the output vector is:

$$\boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{41}$$

Table 1 Command model parameters.

	Rotorcraft		Fighter Jet	
Axis	k [rad/(s-%)]	ω_n [rad/s]	k [rad/(s-deg)]	ω_n [rad/s]
Roll	$\pi/100$	3.5	$290(\pi/180)/21.5$	10.0
Pitch	$\pi/100$	4.5	$(8g/834)/25$	10.0
Yaw	0.644	2.0	$1/30$	4.0

To have the control inputs appear in explicitly in the output equations, the angular rates equations must be differentiated once (Ref. 8, 12):

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \boldsymbol{C}\hat{\boldsymbol{A}}\hat{\boldsymbol{x}} + \boldsymbol{C}\hat{\boldsymbol{B}}\hat{\boldsymbol{u}} \tag{42}$$

The objective of the DI control law is that the output $\boldsymbol{y}$ tracks a reference trajectory $\boldsymbol{y}_{\text{cmd}}(t)$ given by:

$$\boldsymbol{y}_{\text{cmd}}^T = [p_{\text{cmd}}\ q_{\text{cmd}}\ r_{\text{cmd}}] \tag{43}$$

with desired response characteristics. For this reason, the reference trajectory is fed through first-order command models (also known as command filters or ideal responses) which dictate the desired response of the system. The command models are also used to extract the first derivative of the filtered reference trajectory for use in the proportional-integral (PI) compensator described below. The command models are of the following form:

$$G_{\text{ideal}}(s) = \frac{k}{\tau s + 1} \tag{44}$$

where τ is the first-order command model time constant, which is the inverse of the command model break frequency (*i.e.*, $\tau = 1/\omega_n$). Table 1 shows the values used for the parameters of the command models of the inner loop.
PI compensation is used to reject external disturbances and to compensate for discrepancies between the approximate model used in this derivation and the actual bare-airframe dynamics of the aircraft. The resulting DI control law is found by solving for the control vector in Eq. (42), leading to:

$$\hat{\boldsymbol{u}} = \left(\boldsymbol{C}\hat{\boldsymbol{B}}\right)^{-1}\left(\boldsymbol{\nu} - \boldsymbol{C}\hat{\boldsymbol{A}}\hat{\boldsymbol{x}}\right) \tag{45}$$

where $\boldsymbol{\nu}$ is the pseudo-command vector and $\boldsymbol{e}$ is the error as defined respectively in Eqs. (46) and (47).

$$\begin{bmatrix} \nu_p \\ \nu_q \\ \nu_r \end{bmatrix} = \begin{bmatrix} \dot{p}_{\text{cmd}} \\ \dot{q}_{\text{cmd}} \\ \dot{r}_{\text{cmd}} \end{bmatrix} + \boldsymbol{K}_P \begin{bmatrix} e_p \\ e_q \\ e_r \end{bmatrix} + \boldsymbol{K}_I \begin{bmatrix} \int e_p dt \\ \int e_q dt \\ \int e_r dt \end{bmatrix} \tag{46}$$

$$\boldsymbol{e} = \boldsymbol{y}_{\text{cmd}} - \boldsymbol{y}; \tag{47}$$

The 3-by-3 diagonal matrices $\boldsymbol{K}_P, \boldsymbol{K}_I$ identify the proportional and integral gain matrices, respectively. Note that the coefficient matrices $(\boldsymbol{C}\hat{\boldsymbol{B}})^{-1}$ and $\boldsymbol{C}\hat{\boldsymbol{A}}$ are functions of the total velocity of the aircraft V. For this reason, from a practical standpoint, these matrices are computed offline at incremental longitudinal speeds and stored. When the linearized DI controller is implemented on the nonlinear aircraft dynamics, the coefficient matrices $(\boldsymbol{C}\hat{\boldsymbol{B}})^{-1}$ and and $\boldsymbol{C}\hat{\boldsymbol{A}}$ are computed at each time step via interpolation based on the current total velocity $V(t)$ and on the tabled stored offline. It is important to note that what is implemented on the nonlinear aircraft dynamics is linearized DI. However, because the coefficient matrices are scheduled with the longitudinal speed, where scheduling effectively introduces a nonlinear relation between the aircraft states and the feedback control input, the controller implemented is effectively nonlinear DI (NDI) (Ref. 8). A block diagram of the linearized DI flight control law is shown in Fig. 2.

Because helicopters are flown differently at high-speed flight (i.e., greater than 60 kts) and low-speed flight (*i.e.*, lower than 40 kts), different control strategies are needed to control the yaw rate. Above 60 kts, turn coordination is

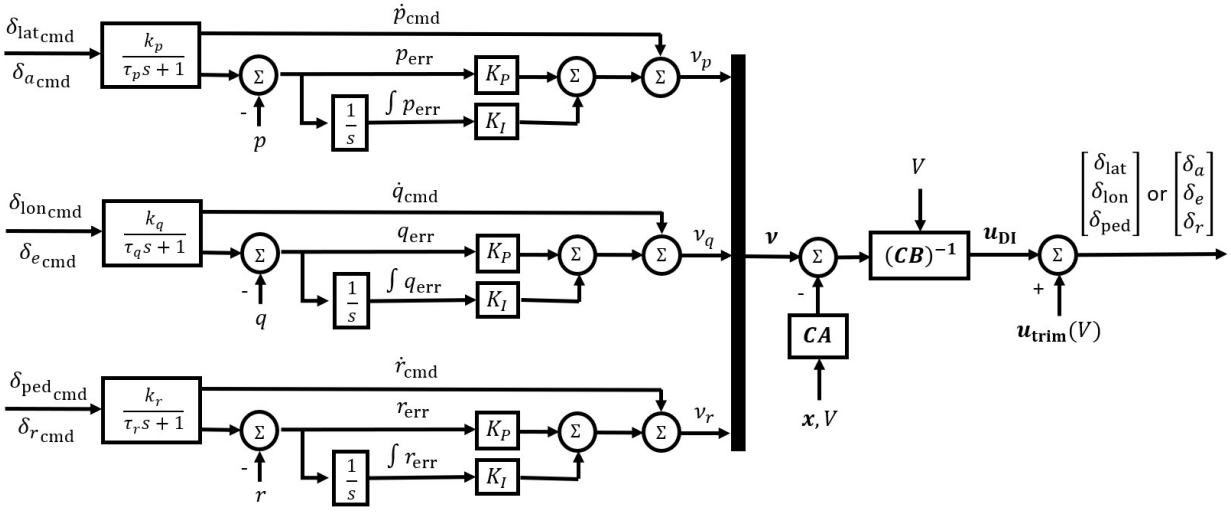


Fig. 2 Dynamic inversion flight control law.

used; below 40 kts no turn coordination (Ref. 20) is used; between 40 and 60 kts a blend between the two is used. These three control strategies are summarized as follows:

$$r'_{\text{cmd}} = \begin{cases} r_{\text{cmd}} & V < V_{\text{LS}} \\ r_{\text{cmd}} + \dfrac{g}{V}\sin\phi\left(\dfrac{V - V_{\text{LS}}}{V_{\text{HS}} - V_{\text{LS}}}\right) & V_{\text{LS}} \le V < V_{\text{HS}} \\ r_{\text{cmd}} + \dfrac{g}{V}\sin\phi & V \ge V_{\text{HS}} \end{cases} \tag{48}$$

where $V = \sqrt{V_x^2 + V_y^2 + V_z^2}$ is the total speed of the aircraft, $V_{\text{LS}} = 40$ kts, and $V_{\text{HS}} = 60$ kts. On the other hand, fixed-wing aircraft are typically flown using turn coordination across their flight envelope. As such, turn coordination is always enforced on the fighter jet model.

Since the heave axis is left open loop, feedback compensation does not provide adjustments to potential losses in altitude during turns causes by rotating the thrust vector around the roll axis. This is why turn compensation is added as a feed-forward signal to the rotorcraft controller. The turn compensation law relates the pitch rate with yaw rate and bank angle of the rotorcraf, and is given by:

$$q'_{\text{cmd}} = \begin{cases} q_{\text{cmd}} & V < V_{\text{LS}} \\ q_{\text{cmd}} + r\sin\phi\left(\dfrac{V - V_{\text{LS}}}{V_{\text{HS}} - V_{\text{LS}}}\right) & V_{\text{LS}} \le V < V_{\text{HS}} \\ q_{\text{cmd}} + r\sin\phi & V \ge V_{\text{HS}} \end{cases} \tag{49}$$

The flight control law is implemented in the scripts `SimpleHel_DI.jl`, `GenHel_DI.jl`, and `F16_DI.jl`. The coefficient matrices tables used in the DI control law are obtained with he script `DI_design.jl`, which must always be run after any changes are made to the dynamic model.

1. Error Dynamics

Feedback compensation is needed to ensure the system tracks the command models. It can be demonstrated (Ref. 12) that for a DI control law the output equation must be differentiated n times for the controls to appear explicitly in the output equation:

$$e^{(n)} = y_{\text{cmd}}^{(n)} - \nu \tag{50}$$

For those outputs equations that require being differentiated only once, a PI control strategy is applied to the pseudo-command vector:

$$\nu = \dot{y}_{\text{cmd}}(t) + K_P e(t) + K_I \int_0^t e(\tau)d\tau \tag{51}$$

Table 2 Disturbance rejection break frequencies and damping ratios.

	Rotorcraft		Fighter Jet	
Axis	ω_n [rad/s]	ζ	ω_n [rad/s]	ζ
Roll	3.5	0.7	10.0	0.7
Pitch q	4.5	0.7	10.0	0.7
Yaw r	2.0	0.7	4.0	0.7

Substituting Eq. (51) into Eq. (50) leads to the closed-loop error dynamics:

$$\dot{e}(t) + K_P e(t) + K_I \int_0^t e(\tau) d\tau = 0 \tag{52}$$

The gains are chosen such that the frequencies of the error dynamics are of the same order as the command filters (*i.e.*, first order), ensuring that the bandwidth of the response to disturbances is comparable to that of an input given by a pilot or outer loop. By taking the Laplace transform, and therefore switching to the frequency domain, the error dynamics becomes

$$e(s)\left(s^2 + sK_P + K_I\right) = 0 \tag{53}$$

To obtain the gains that guarantee the desired response, the error dynamics of Eq. (53) is set equal to the following second-order system:

$$s^2 + 2\zeta\omega_n s + {\omega_n}^2 = 0 \tag{54}$$

yielding the following proportional and integral gains:

$$K_P = 2\zeta\omega_n \tag{55a}$$

$$K_I = {\omega_n}^2 \tag{55b}$$

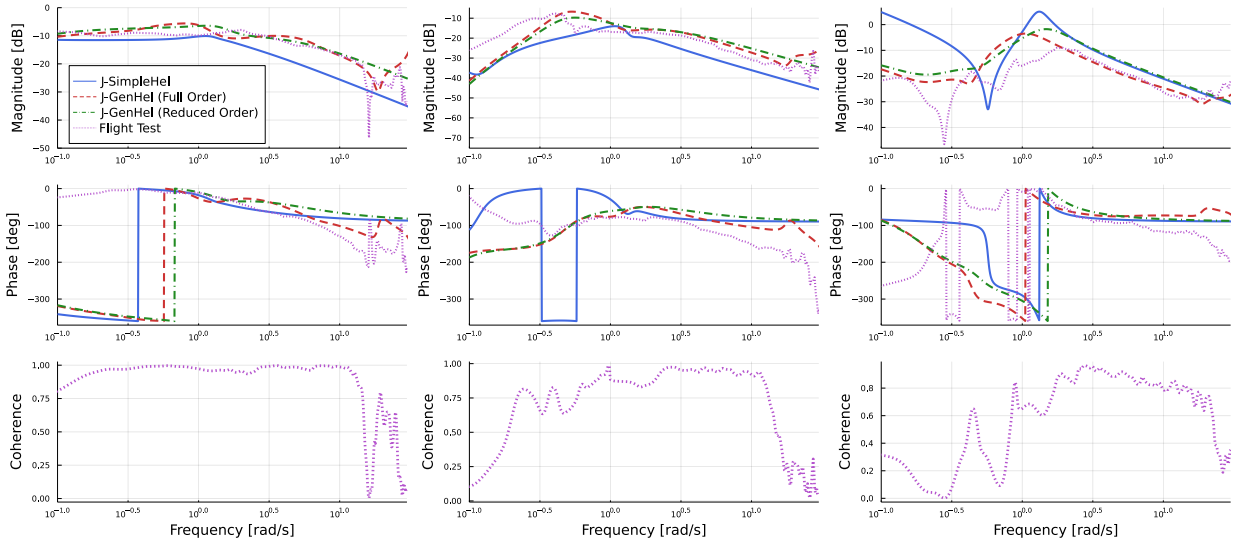
Table 2 show the natural frequencies, damping ratios, and time constants, respectively, for the single-loop strategy. Because the plant is inverted in the feedback linearization loop such that the system being controlled is effectively a set of integrators, there is no need for gain scheduling. However, the plant model used for feedback linearization still requires to be scheduled with the flight condition. The command filter properties and error dynamics gains are found in the script `DI_design.jl`,

V. Results

A. Rotorcraft Models

To validate the open-loop dynamics of both the simple and higher-fidelity rotorcraft models, the on-axis frequency responses are compared with those obtained from flight test data of the U.S. Army Rotorcraft Aircrew System Concepts Airborne Laboratory (RASCAL) JUH-60A helicopter. Figure 3 shows the on-axis frequency responses at 80 kts forward flight. This figure indicate a good match between the frequency responses of the full-order linearized J-GenHel dynamics and that of the flight test data. The reduced-order linearized J-GenHel dynamics provides a good approximation of the UH-60 dynamics up to about 4 rad/s, after which it shown reduced phase roll off in the roll and pitch axes (Figs. 3a and 3b), but not in the yaw axis (Fig. 3c). This is due to the absence of the states associated with the parasitic dynamics (*i.e.*, flap, lead-lag, inflow, *etc.*) in the reduced-order model. Frequency responses from J-SimpleHel show reduced accuracy compared to J-GenHel given the simpler nature of the model.

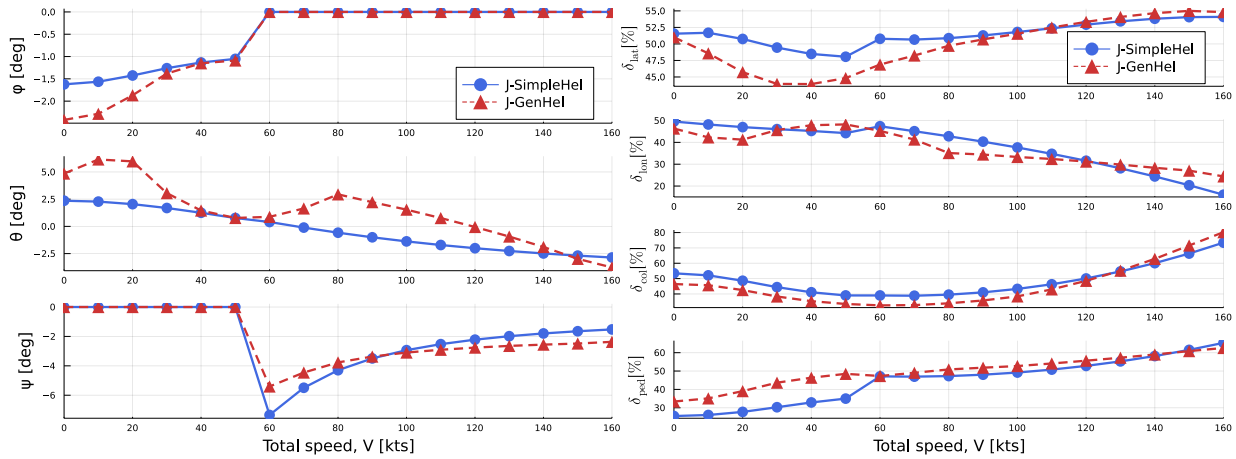
Next, key trim state and control variables are compared for level flight at varying flight speeds, as shown in Fig. 4. The J-SimpleHel and J-GenHel helicopter models are trimmed at speeds ranging from hover to 160 kts at increments of 5 kts at a weight of 16000 lb. Figure 4a shows the trim attitude for varying speeds. In this figure, it is shown that the trim pitch attitude from J-SimpleHel is generally lower than that from J-GenHel. This may be explained by the fact that J-SimpleHel does not incorporate an interference model of the main rotor wake on the horizontal stabilizer, as opposed to J-GenHel. As such, the missing download on the horizontal stabilizer indices lower pitch attitudes, especially at low



(a) $\frac{p}{\delta_{\text{lat}}}$ (b) $\frac{q}{\delta_{\text{lon}}}$ (c) $\frac{r}{\delta_{\text{ped}}}$

Fig. 3 On-axis frequency responses comparison with JUH-60A RASCAL flight test data at 80 kts level flight.

speed, where the horizontal stabilizer fully operates in the wake of the main rotor. The mismatch in trim pitch attitudes is smaller at high speed, where the interference between main rotor wake and horizontal stabilizer is less significant. Figure 4a shows the trim stick inputs for varying speeds. Notably, the collective stick follows the traditional helicopter power curve (Ref. 21) and the trim longitudinal cyclic stick droops with forward speed, which suggest that the models were implemented correctly. Generally, the trim stick inputs with speed of the two models are similar.



(a) Euler angles. (b) Stick inputs.

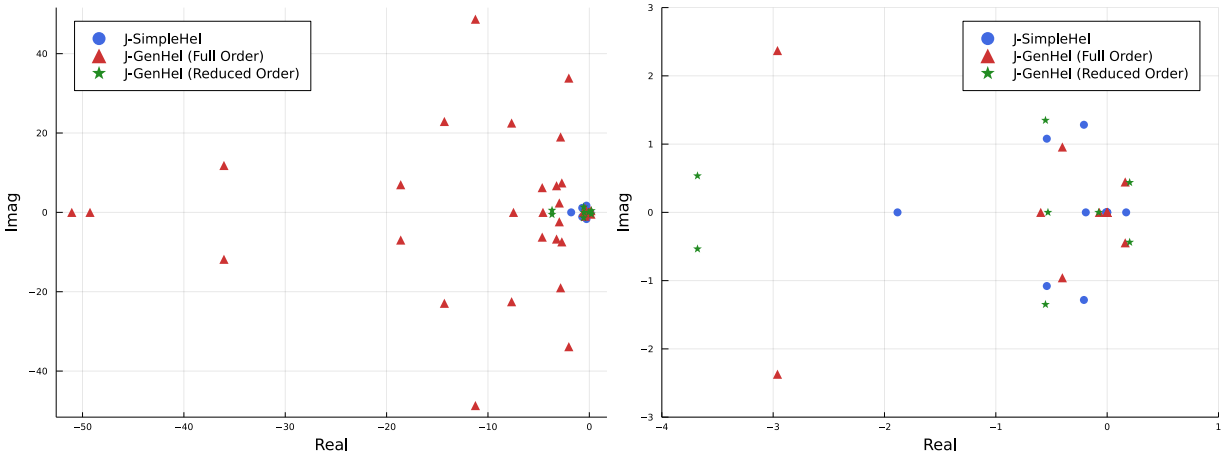
Fig. 4 Rotorcraft models trim state and control variables for varying speed.

The eigenvalues of the two models are compared, along with those of the reduced-order model, as shown in Fig. 5. The analysis focuses on the the 80 kts flight condition. Figure 5a shows the all the eigenvalues of the J-GenHel model, including those of the rigid-body, main rotor, and tail rotor dynamics. The rigid-body poles are all clustered at low frequency, as indicated by the reduced-order model and J-SimpleHel eigenvalues. A close up of the low-frequency poles associated with the rigid-body dynamics is shown in Fig. 5b. These eigenvalues are also summarized in Table 3, which identifies the modes based on analysis of the eigenvectors. Notably, the reduced-order model eigenvalues

Table 3 Rigid-body eigenvalues of the rotorcraft models at 80 kts level flight.

Mode	J-SimpleHel Eigenvalues	J-GenHel Eigenvalues (Full Order)	J-GenHel Eigenvalues (Reduced Order)
Spiral	-0.0109	-0.0701	-0.0767
Phugoid	0.1706	0.1623 ± 0.4456	0.2020 ± 0.4377
Dutch Roll	-0.5420 ± 1.0792	-0.4022 ± 0.9577	-0.5318 ± 1.3487
Coupled Roll/Pitch Short Period	-0.2086 ± 1.2833	-2.9611 ± 2.3710	$-3.6797 \pm 0.5351i$
Coupled Roll/Pitch Short Period	-0.1918	-0.5965	-0.5317
Roll Subsidence	-1.8830	-	-

nearly overlay corresponding full-order model poles, with the exception of the poles with real parts around -2, where there is some discrepancy in damping. This mismatch was previously observed in Ref. 8. In any case, this analysis shows that the reduced-order model is indeed a good approximation of the low frequency dynamics of the aircraft. It is also worth noting that while J-SimpleHel eigenvalues are somewhat similar to those of J-GenHel, those poles normally corresponding to the phugoid mode (and thus complex conjugates) become two real poles, one slightly positive and one negative.



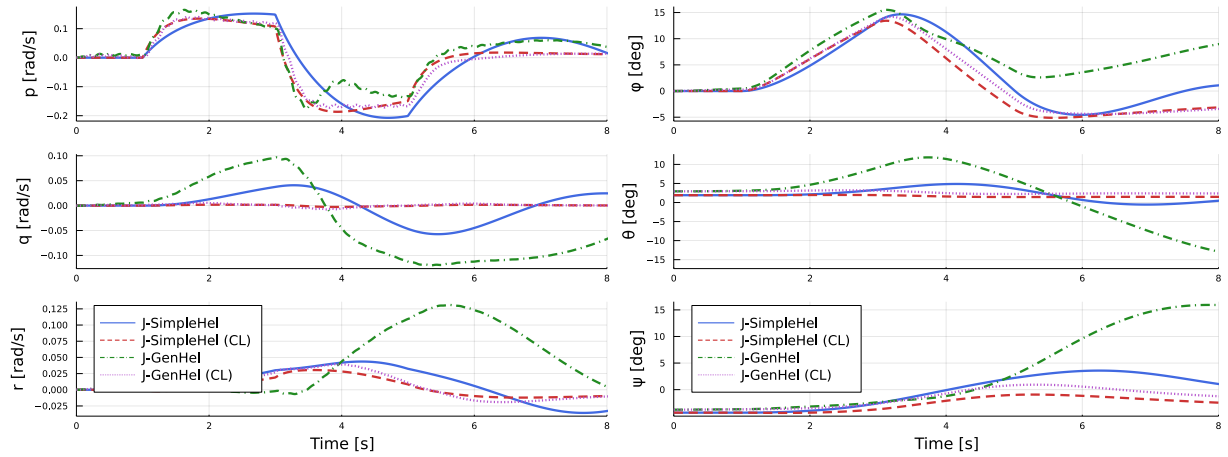
(a) Full order. (b) Rigid-body.

Fig. 5 Rotorcraft models eigenvalues at 80 kts level flight.

The controller performance is demonstrated through flight simulations. Figure 6 shows the open- and closed-loop responses to a 5% lateral stick doublet for the J-SimpleHel and J-GenHel models at 80 kts. As shown in Fig. 6a, the controller tracks the commanded roll rate doublet while enhancing the off-axis response, as is clearly seen in the pitch rate plot. The mitigation of the off-axis response is also seen in the pitch attitude plot of Fig. 6b. These results suggests that the flight control law implemented successfully stabilizes the rotorcraft dynamics while providing desired response characteristics. It is worth noting, however, that the closed-loop dynamics may diverge if the simulation is run for long enough, depending on the nature of the transmission zeros of the closed-loop dynamics of each model. The interested reader is invited to consult Ref. 8 for more details on the topic and possible solutions.

B. Fighter Jet Model

Like for the rotorcraft models, to help validate the model, the F-16 dynamics is trimmed in level flight at incremental speeds, as shown in Fig. 7. This figure shows decreasing pitch attitude and elevator angle for increasing speeds, in



(a) Angular rates.

(b) Attitude.

Fig. 6 Open- and closed-loop responses to a 5% lateral stick doublet for the J-SimpleHel and J-GenHel models at 80 kts.

addition to monotonically increasing power setting for increasing speeds. These are common behaviors of fixed-wing aircraft, which suggest the correct implementation of the J-F16 model. Next, the eigenvalues are computed at 600 ft/s level flight and shown in Fig. 8. These eigenvalues are also reported in Table 4, which identifies the modes based on analysis of the eigenvectors. Notably, the aircraft exhibits a non-traditional phugoid mode with two real poles, one positive at very low frequencies, and one negative at higher frequencies.

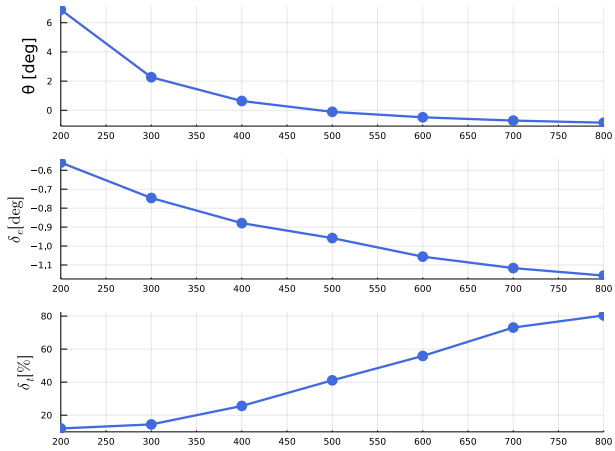


Fig. 7 F-16 model trim state and control variables for varying speed.

Like for the rotorcraft models, the controller performance is demonstrated via flight simulations. These flight simulations are shown in Fig. 9 and correspond to the open- and closed-loop responses to a 5 deg lateral stick doublet at 600 kts. Figure 9a shows how the controller tracks the commanded roll rate while providing reduced off-axis response when compared to the open-loop response. Similar observations can be made for the Euler angles responses shown in Fig. 9a. These results indicate that the flight control law implemented provides desired response characteristics while improving the off-axis response.

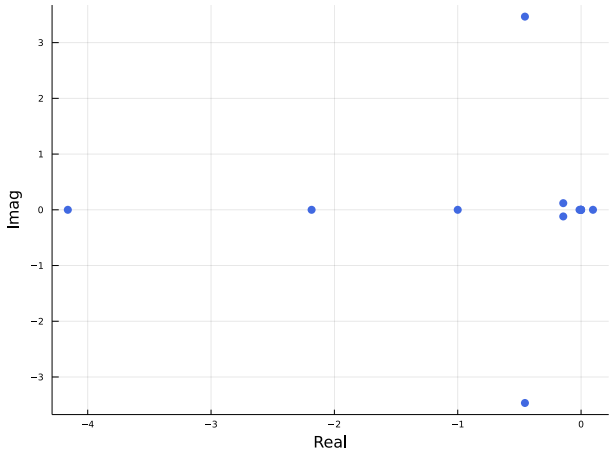
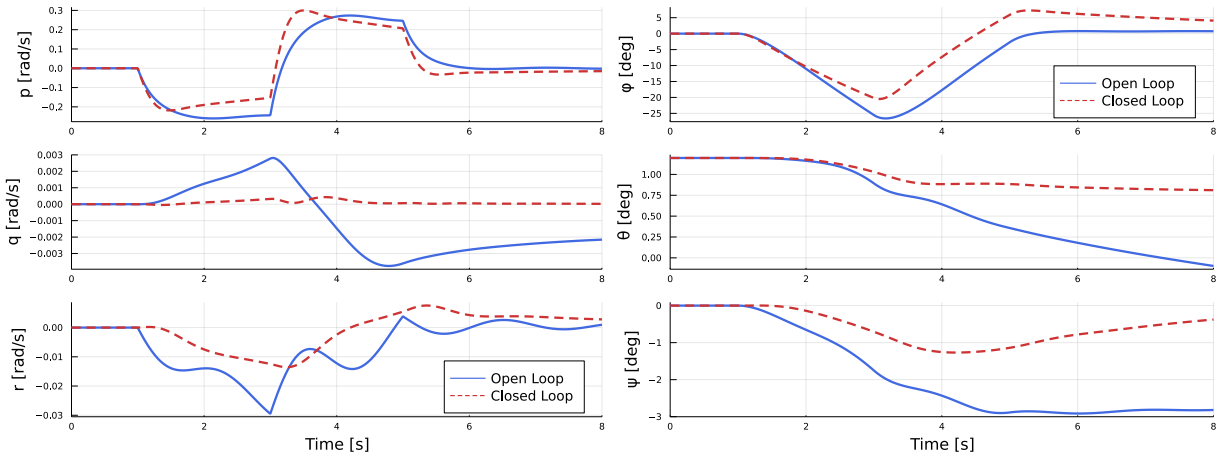


Fig. 8 F-16 model eigenvalues at 600 ft/s level flight.

Table 4 F-16 eigenvalues at 600 ft/s level flight.

Mode	J-F16 Eigenvalues
Spiral	-0.0124
Phugoid	0.0896
Short Period	$-0.1421 \pm 0.1103i$
Dutch Roll	$-0.4550 \pm 0.1103i$
Pitch Subsidence	-2.1849
Roll Subsidence	-4.1612



(a) Angular rates.

(b) Attitude.

Fig. 9 Open- and closed-loop responses to a 5 deg lateral stick doublet for the J-F16 model at 600 ft/s.

VI. Conclusion

The paper presented the development of three flight simulation models using the Julia language: a simple helicopter model (J-SimpleHel), a higher-fidelity helicopter model (J-GenHel), and a fighter jet model (J-F16). These models were validated against flight test data, where available. The two rotorcraft models were compared to assess similarities and

discrepancies in relation to the complexity of the models using both time and frequency domain analyses. In addition, flight control laws based on a nonlinear dynamic inversion (NDI) architecture were developed and implemented in Julia. Simulation results demonstrate that the NDI flight control law are able to provide desired stability and response characteristics. Based on this work, it is concluded that the Julia language is indeed a suitable tool for flight simulation and control analyses. Future work may focus on computing performance comparisons with equivalent flight simulation models developed in MATLAB®.

Acknowledgments

The Rotorcraft Aircrew System Concepts Airborne Laboratory (RASCAL) flight data used in this research/investigation was provided to the Pennsylvania State University under the U.S. Army/Navy/NASA Vertical Lift Research Center of Excellence, Agreement No. W911W6-17-2-0003.

References

[1] Bezanson, J., Karpinski, S., Shah, V. B., and Edelman, A., "Julia: A Fast Dynamic Language for Technical Computing," *arXiv: 1209.5145*, 2012, pp. 1–27.

[2] Murphy, M., "Octave: A Free, High-Level Language for Mathematics," *Linux Journal*, 1997, pp. 1–27.

[3] Howlett, J. J., "UH-60A Black Hawk Engineering Simulation Program. Volume 1: Mathematical Model," Tech. rep., NASA-CR-166309, 1980.

[4] Brentner, K. S., Bres, G. A., Perez, G., and Jones, H. E., "Maneuvering Rotorcraft Noise prediction: A New Code for a New Problem," *Proceedings of the AHS Aerodynamics, Acoustics, and test Evaluation Specialist Meeting*, San Francisco, CA, Jan 23–25, 2002.

[5] Frazzoli, E., Dahlel, M. A., and Feron, E., "Maneuver-Based Motion Planning for Nonlinear Systems With Symmetries," *IEEE Transactions on Robotics*, Vol. 21, No. 6, 2005, pp. 1077–1091.

[6] Ben-Israel, A., "A Newton-Raphson Method for the Solution of Systems of Equations," *Journal of Mathematical Analysis and Applications*, Vol. 15, No. 2, 1966, pp. 243–252.

[7] Kokotovic, P. V., O'Malley, R. E., and Sannuti, P., "Singular Perturbations and Order Reduction in Control Theory, an Overview," *Automatica*, Vol. 12, No. 2, 1976, pp. 123–132.

[8] Horn, J. F., "Non-Linear Dynamic Inversion Control Design for Rotorcraft," *Aerospace*, Vol. 6, No. 38, 2019.

[9] Padfield, G. D., *Helicopter Flight Dynamics: Including a Treatment of Tiltrotor Aircraft*, John Wiley & Sons, 2018.

[10] Pitt, D. M., and Peters, D. A., "Theoretical Prediction of Dynamic-Inflow Derivatives," *Proceedings of the* 6^{th} *European Rotorcraft and Powered Lift Aircraft Forum*, Sept 16-19, 1980.

[11] Bailey, F. J., "A Simplified Theoretical Model of Determining the Characteristics of a Lifting Rotor in Forward Flight," Tech. rep., NACA Report No. 716, 1941.

[12] Stevens, B. L., Lewis, F. L., and N., J. E., *Aircraft Control and Simulation: Dynamics, Controls, Design, and Autonomous Systems*, John Wiley & Sons, 2015.

[13] Garza, F. R., and Morelli, E. A., "A Collection of Nonlinear Aircraft Simulations in MATLAB," Tech. rep., NASA TM-2003-212145, 2003.

[14] Saetti, U., Horn, J. F., Villafana, W., Brentner, K. S., and Wachspress, D., "Rotorcraft Simulations with Coupled Flight Dynamics, Free Wake, and Acoustics," *Proceedings of the* 72^{th} *Annual Forum of the American Helicopter Society*, 2016.

[15] Saetti, J., U., Horn, J. F., Lakhmani, S., Lagoa, C., and Berger, T., "Design of Dynamic Inversion and Explicit Model Following Control Laws for Quadrotor UAS," *Journal of the American Helicopter Society*, Vol. 65, No. 3, 2020, pp. 1–16(16). https://doi.org/https://doi.org/10.4050/JAHS.65.032006.

[16] Scaramal, M., Horn, J. F., and Saetti, U., "Load Alleviation Control using Dynamic inversion with Direct Load Feedback," *Proceedings of the* 77^{th} *Annual Forum of the Vertical Flight Society*, 2021.

[17] Caudle, D. B., "DAMAGE MITIGATION FOR ROTORCRAFT THROUGH LOAD ALLEVIATING CONTROL," Master's thesis, The Pennsylvania State University, University Park, PA, 2014.

[18] Spires, J. M., and Horn, J. F., "Multi-Input Multi-Output Model-Following Control Design Methods for Rotorcraft," *AHS 71st Annual Forum*, Virginia Beach, VA, May 5-7, 2015.

[19] Yomchinda, T., "REAL-TIME PATH PLANNING AND AUTONOMOUS CONTROL FOR HELICOPTER AUTOROTATION," Ph.D. thesis, Pennsylvania State University, May, 2013.

[20] Blakelock, J. H., *Automatic Control of Aircraft and Missiles*, John Wiley & Sons, 1965. https://doi.org/https://doi.org/10.1017/S0001924000057912.

[21] Leishman, J. G., *Principles of Helicopter Aerodynamics*, Cambridge University Press, 2006.