Use of a Task-Pilot-Vehicle (TPV) Model as a Tool for Flight Simulator Math Model Development

Robert K. Heffley*
Robert Heffley Engineering, Cupertino, CA, 95014

The Task-Pilot-Vehicle (TPV) model structure is able to represent the combined pilot-vehicle system in order to produce realistic performance of many real-world flight tasks and maneuvers. The scheme employs a set of graphical user interfaces for setting up flight task scenarios, setting pilot decision and control parameters, running simulations, and analyzing results. It includes a concise method for defining flight tasks and maneuvers, and pilot control strategy and technique. The TPV model has run with several vehicle math models, including CASTLE, *FlightLab*©, *RotorGen2*©, as well as linear state-space models. The current TPV model is implemented in Simulink® and uses *FlightGear* open-source software to provide a visual 3D display of the simulation. The TPV model scheme is useful for rapid prototyping system design and as a simulation or flight planning tool. This paper describes use of the TPV modeling scheme in the context of its application to vehicle math model development.

I. Introduction

VEHICLE math model development typically is a long, labor-intensive process that results in a product that sometimes is not tested or examined effectively until a human pilot examines it in a manned-simulator environment. At that stage math model adjustment or troubleshooting can be expensive and involve recurring manned-simulator work.

This paper describes a math model development process that aids and can accelerate model development using a simulation scheme that emulates flying the vehicle math model in a realistic context at a very early stage and without manned simulation.

The Task-Pilot-Vehicle (TPV) math model architecture shown in Fig. 1 has been developed and applied in several projects spanning the past ten years.[1-5] The TPV model is able to represent the combined pilot-vehicle system in a context that represents realistic performance of real-world flight tasks and maneuvers. It is the basis of a rapid prototyping tool that has been applied in several ways.

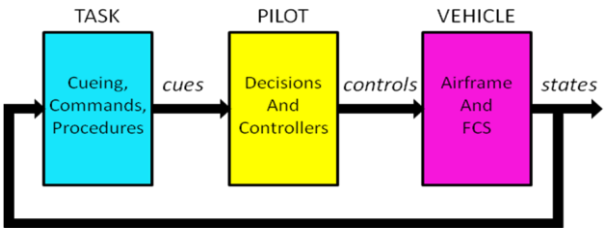


Figure 1. Basic TPV Model Structure.

This paper describes recent experience using the TPV model scheme as an aid to vehicle math model development. The TPV model described is the most recent version in its continually evolving form. Combined with several other conventional system analysis tools, a vehicle math model can be rapidly configured and tested prior to actual manned simulator use.

The following is a description of the TPV modeling scheme along with a general description of the overall vehicle modeling process. While one specific vehicle math model form is used as an example, the same process, including the TPV model itself, can be applied to any vehicle model form and vehicle type.

* Owner, Robert Heffley Engineering, 21070 Homestead Road, Cupertino, CA 95014, AIAA Senior Member.

II. Overview of the Task-Pilot-Vehicle (TPV) System Model

Fig. 2 shows the above TPV model structure as it is implemented in a *Simulink*® block diagram that represents the overall task-pilot-vehicle system. Each of the main blocks in the feedback loop, from left to right, are represented pictorially and include the task, the pilot, and the vehicle, respectively. Various output forms can include an array of time-history plots and a *FlightGear* 3-D visualization of the aircraft from a choice of viewpoints.

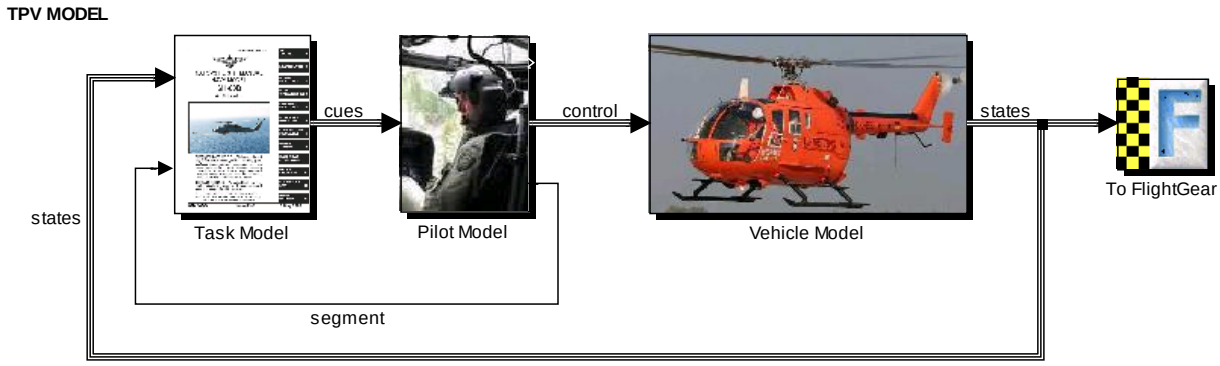


Figure 2. Simulink® TPV Modeling Environment Expressed as a Simulink Block Diagram.

The purpose of this math model form is to provide a facsimile of the human pilot performing a realistic flight task or maneuver using a given vehicle math model. The vehicle can be linear, non-linear, or even a vehicle math model under development. The task model can consist of a simple continuous tracking task or it can be a complex series of segments that mimics a realistic sequence of events or segments. The pilot is represented as having two main functions, decision-making and controlling.

The modularity of the model allows for substitution of alternative model forms so long as the basic module interfaces are consistent with the input/output relationships listed in Table 1. Note that the term "cues" includes commands per the task definition.

In general, the task model can be defined independent of the pilot and vehicle models. The pilot model is dependent on the specific vehicle but may not vary much with flight condition. The arrays of state, cue, and control variables depend upon the specific aircraft type and flight task being performed.

Notwithstanding the variety of flight tasks and vehicles that may be of interest, the TPV model form described here has been capable of simulating a wide range of task and vehicle cases. These range from helicopter maneuvers and shipboard terminal operations, to STOVL and tilt-rotor takeoff and landing, to the complex fixed-wing carrier-landing task. These are all able to be defined using the concise task and pilot model functions that are part of the current TPV model scheme.

III. History

The TPV model concept is an extension of several "discrete-maneuver" models developed and applied to analysis of several airplane and helicopter flight tasks such as landing, deceleration, quick-stop, etc.[6-12] This class of pilot models differs from "tracking tasks" in not being continuous. There is a definable start and end.

One simple example of an early discrete maneuver model is taken from Ref. 6 and illustrated in Fig. 3. (In this case we would now interpret the visual-perspective block on the left as a "task" model and, on the left, the combined "pilot" and "vehicle" models.) The value of this particular model was to obtain a closed-form solution of a decelerating approach to hover that closely represented actual piloted-approaches for a UH-1 helicopter.[13] The visual cueing model was based on the "perceived range" function presented in Ref. 14. An example of the data fitted to data from one such approach is shown in Fig. 3. Note that the plotted deleration profile would be approximately proportional to the pich attitude during the decelaration.

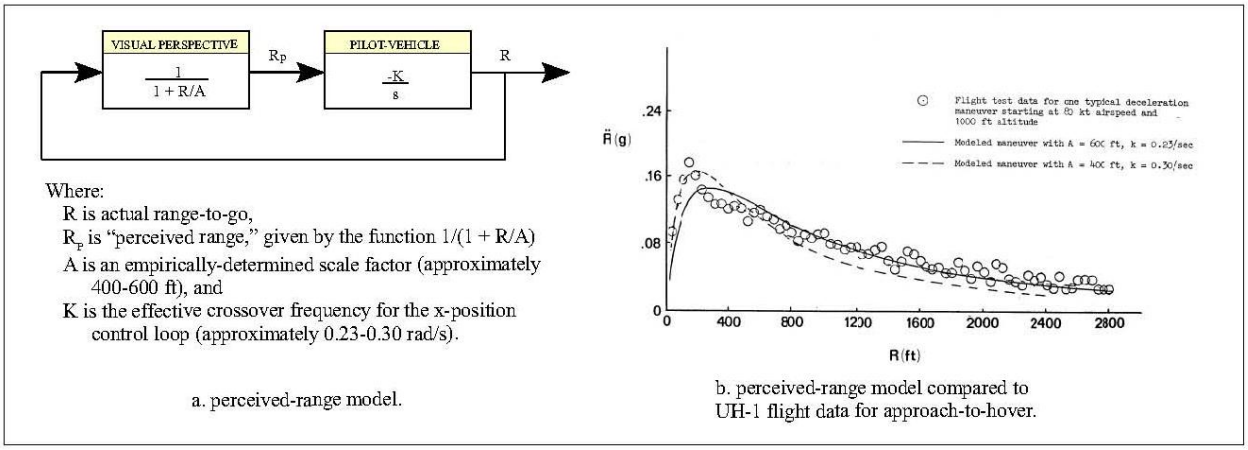


Figure 3. Perceived-Range Deceleration Model—An Early TPV Model Form.

The TPV model consists of a series of discrete maneuvers or task elements that are connected by pilot decisions for when to transition from one segment to the next and by a shift in pilot control strategy or technique appropriate to a given segment.

The original TPV model was devised to represent a helicopter pilot performing some of the ADS-33E[15] demonstration maneuvers in a SBIR sponsored by the AAFD[†] at Ames Research Center[1-3]). This entailed the characterization of specific tasks (e.g., ADS-33E precision hover, depart/abort, and pirouette maneuvers) using the ADS-33E task descriptions, consideration of pilot commentary, and analysis of actual manned-simulation data for validation.

The TPV scheme was further developed under a NAVAIR SBIR[4-5] in order to examine the ship-aircraft interface for a series of fixed- and rotary-wing aircraft and ship types, particularly with respect to the effects of ship-generated air wakes. This work included applying the TPV model to the carrier landing task, helicopter approach to, landing on, and departure from a guided-missile destroyer (DDG) deck, and VSTOL and tilt-rotor operations from an amphibious assault ship (LHA) flattop deck.

Fig. 4. Shows 3-D visualizetions of various cases modeled for near-ship takeoff and landing flight tasks. Of these, the F/A-18 carrier landing task was the most complex task modeled.

The present TPV model status is the result of an in-house development effort to create a refined architecture based in a Matlab and Simulink environment with an open-source 3-D visualization component using *FlightGear*[16]. This version of the TPV model uses a generalized task model scheme that accommodates a wide range of complex task scenarios and the ability to be easily modified if necessary. The vehicle

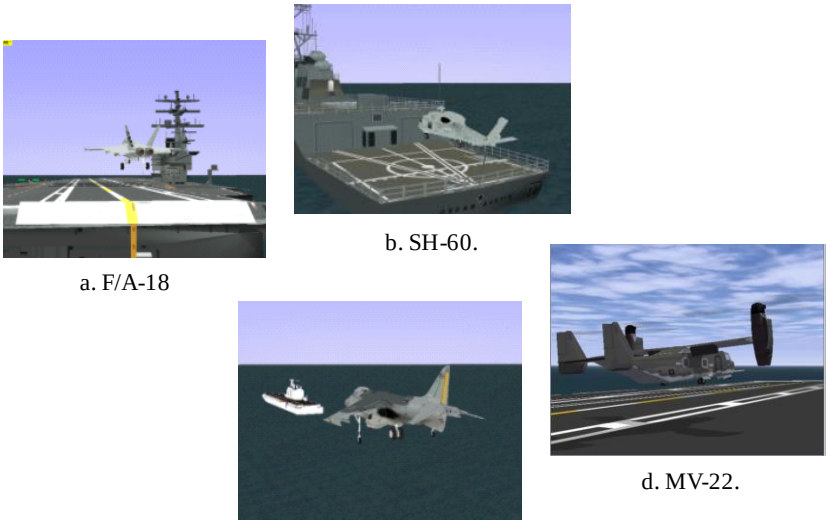


Figure 4. Cases Modeled by TPV Simulation in NAVAIR SBIR Project.

[†] U.S. Army Aeroflightdynamics Directorate.

module can be adapted to various math model forms, but is currently configured either as a simple linear force-and-moment model or a *RotorGen2* nonlinear model. Both run with general nonlinear equations-of-motion and a simple ground contact model. Many of the components in the current Simulink TPV model are represented as Simulink S-functions that could be translated to various programming languages. Finally, several supporting utilities have been implemented to enable quick set up of task, pilot, and vehicle characteristics and manipulation of simulation output for a variety of uses. This paper will focus on the present TPV model scheme.

IV. Task Model Scheme

The task model receives state variable inputs from the vehicle model and outputs cues to the pilot model. The two subsystems comprising the task model in Fig. 5 are the task-segment-dependent *command generator* in series with the pilot *cue generator*.

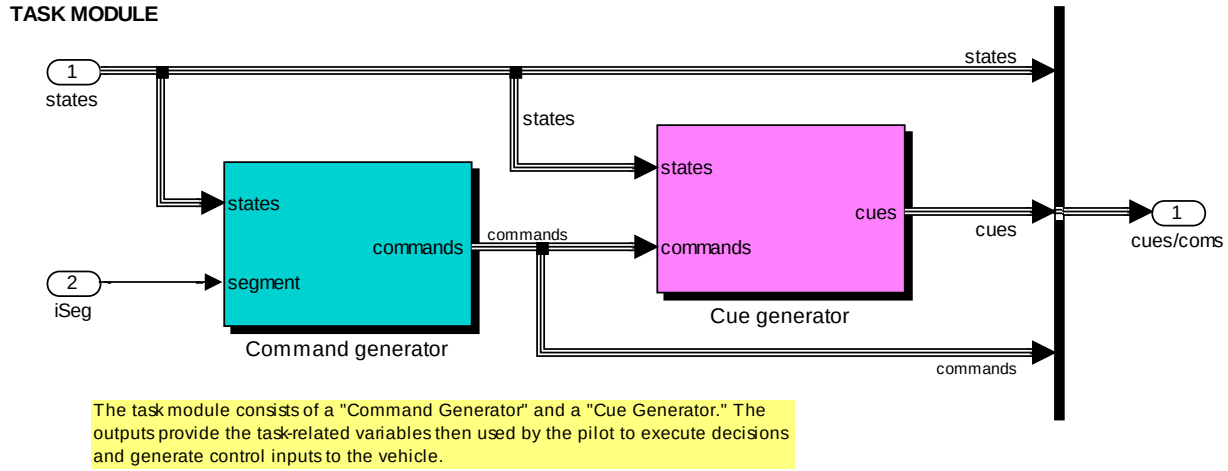


Figure 5. Task Model Components

A. Command Generator

The command generator supplies the pilot model with appropriate pilot controller commands for each of the four controller axes during each task segment. This block contains a Matlab S-function consisting of a table lookup process that sets up commands for each axis depending upon the task segment. The commands are obtained from the task setup file containing task descriptions for all flight tasks defined by the user. The task definition scheme is discussed shortly.

B. Cue Generator

The role of the cue generator is to transform vehicle state variables into cues that the pilot observes from cockpit instruments or senses visually, proprioceptively, or through motion. In the current model random noise can be added to provide a level of uncertainty or error in sensed information. This is one means of modeling a degraded visual environment.

The cue generator is presently composed of an array of several ad hoc functions that provide many kinds of cueing variables that span the range of currently defined flight tasks. In the future these will be expressed more concisely in an S-function.

The current cueing functions include:

- Aircraft-relative-to-ship states,
- Position and velocity states transformed according to convenient task axes,
- Flight path, glideslope, lineup angles,
- Control activity (an empirical function related to how much the pilot is moving controls),
- Ground-contact states (weight-on-wheels, at rest, transition from ground to air, etc.,
- Perceived range cue, and
- Actual states (attitude, heading, velocities, position, etc., and
- Any of the above with random noise added to represent "uncertainty" in cue information.

In most cases it is convenient to begin the process of designing a task model by using the actual state variables as "implicit" cues for the pilot (e.g., the actual pitch Euler angle used as an implicit pitch attitude pilot cue).

Cues can be easily expanded to include any desired functions such as the "tau" parameter developed by Padfield[17-18] or the array of motion cues used by Hess in his structural-model representation of the pilot-vehicle system.[19-20]

C. Task Definition Scheme

The task model can be configured with varying levels of complexity depending upon the application. In general, the most direct formulation is to define a set of serial segments based on a standard task description such as might be found in aircrew training manuals or operations manuals. For example, the US Navy NATOPS manuals for each carrier aircraft contain detailed descriptions of the sequence of events for fixed-wing recovery (e.g., Ref. 21) or for helicopter recovery and launch from the decks of air-capable ships.[22, 23] In the case of an F/A-18 carrier approach starting from a racetrack pattern, about twelve segments are required to perform the approach starting at a downwind position and ending with arrestment on the deck.[4, 5]

1. General Procedure

The general procedure for setting up a given flight task consists of:

1. Assemble descriptive information, including documentation (e.g., aircraft flight manual, NATOPS manuals, aircrew training manuals, and direct pilot commentary that describes how a task is performed, specific standards and criteria for performance, and quantitative measurements and data when available,
2. Identify task segments
3. Define criteria for transitioning from one task to the next,
4. For each segment, define likely control strategy for managing aircraft states,
5. Define the outer-most-loop commands in each control axis,
6. Define and practical limits on inner control loops such as maximum pitch or roll attitude,
7. Integrate the above information into the generalized task-definition scheme.

The following examples illustrate this procedure.

2. *Depart/Abort Maneuver Example*

The ADS-33E "depart/abort maneuver" is a standard demonstration for ensuring handling qualities adequate for an aborted helicopter takeoff that must end in a limited distance. The task description from ADS-33E[15] (item 1 of the above general procedure) is:

> From a stabilized hover at 35 ft wheel height (or no greater than 35 ft external load height) and 800 ft from the intended endpoint, initiate a longitudinal acceleration to perform a normal departure. At 40 to 50 knots groundspeed, abort the departure and decelerate to a hover such that at the termination of the maneuver, the cockpit shall be within 20 ft of the intended endpoint. It is not permissible to overshoot the intended endpoint and move back. If the rotorcraft stopped short, the maneuver is not complete until it is within 20 ft of the intended endpoint. The acceleration and deceleration phases shall be accomplished in a single smooth maneuver. For rotorcraft that use changes in pitch attitude for airspeed control, a target of approximately 20 degrees of pitch attitude should be used for the acceleration and deceleration. The maneuver is complete when control motions have subsided to those necessary to maintain a stable hover.

This description is accompanied by the diagram shown in Fig. 6.

Based on this, we may prescribe four basic segments for the TPV task model, consisting of:

1. Perform a steady hover at the start (hold initial x-, y-, z-position, and heading),
2. Accelerate along the runway (command nose-down pitch while observing limit, maintain y- and h-position, and heading),
3. At 40 kt decelerate to the course endpoint (command x = 790ft, maintain y-, h-position, and heading, observe pitch-up limit),
4. Upon control motions subsiding mark finish, pause then end the task.

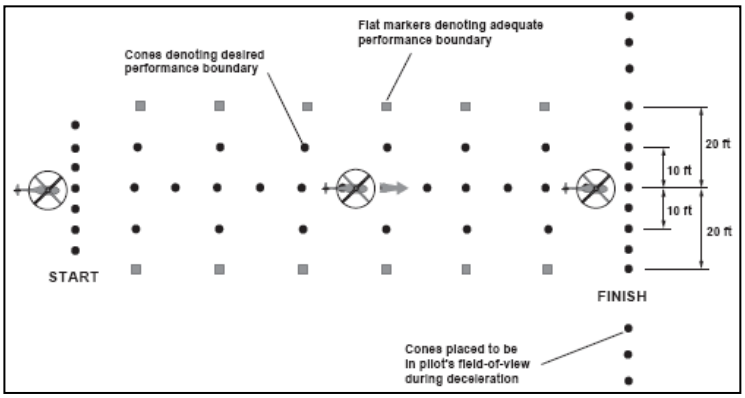


Figure 6. ADS-33E Depart/Abort Demonstration Maneuver.[15]

The TPV model implemented for this flight task results in the summary from the "Task Model Parameters" GUI illustrated in Fig. 7. (Performance of this maneuver will be illustrated in Section IX.C.)

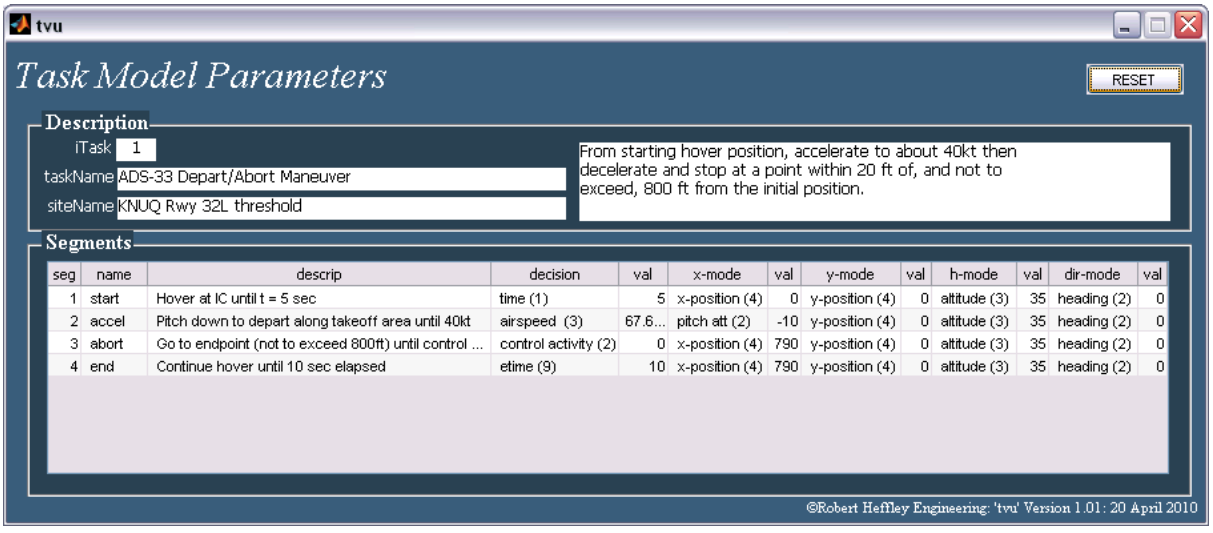


Figure 7. Task Model GUI Example for ADS-33E Depart/Abort Maneuver.

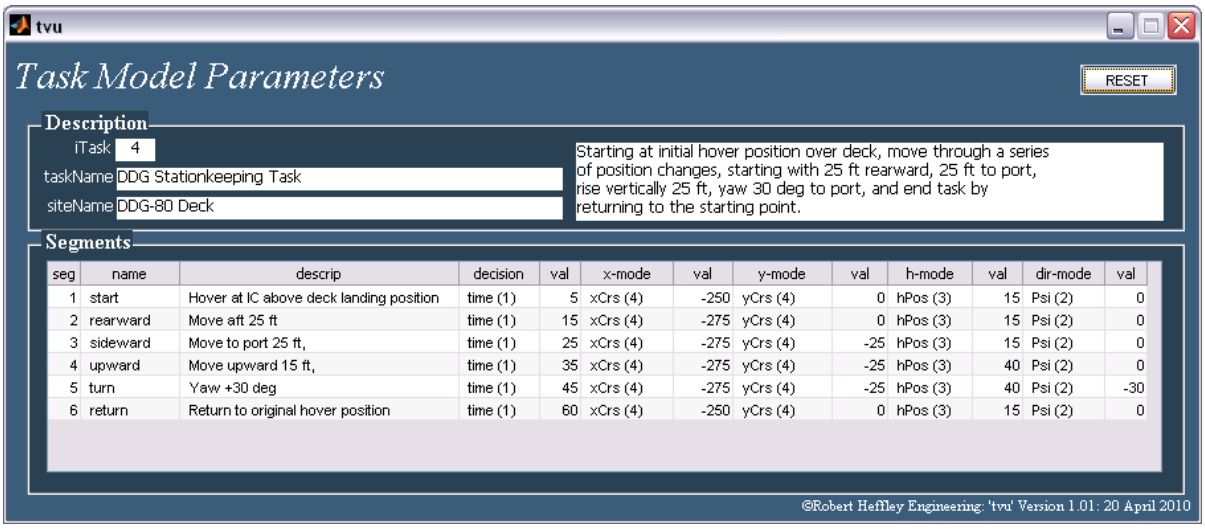
Consider next a second task description example.

3. DDG Deck Stationkeeping Maneuver Example

The "DDG deck stationkeeping maneuver" is a flight task developed to gain human-pilot task performance data from a manned-simulation at the NAVAIR Manned Flight Simulator (MFS) facility.[5] The maneuver consisted of a series of position and heading changes in four dimensions, namely:

1. Perform a steady hover over the deck at the start (hold initial x-, y-, z-position, and heading),
2. After 5 sec move aft 25 ft (command a rearward position change, maintain y- and h-position, and maintain heading),
3. After 15 sec move to port 25 ft (command a sideward position change, maintain x- and h-position, and maintain heading),
4. After 15 sec increase height 15 ft (command an h-position change, maintain x- and y-position, and maintain heading),
5. After 15 sec yaw counterclockwise 30 deg (command a heading change, maintain x- , y- and h-position),
6. After 15 sec return to the original position and heading then end the task.

This is a particularly simple task to model and simulate because it consists of segment transitions that are only a function of time. This results in the task summary illustrated in Fig. 8.



seg	name	descrip	decision	val	x-mode	val	y-mode	val	h-mode	val	dir-mode	val
1	start	Hover at IC above deck landing position	time (1)	5	xCrs (4)	-250	yCrs (4)	0	hPos (3)	15	Psi (2)	0
2	rearward	Move aft 25 ft	time (1)	15	xCrs (4)	-275	yCrs (4)	0	hPos (3)	15	Psi (2)	0
3	sideward	Move to port 25 ft,	time (1)	25	xCrs (4)	-275	yCrs (4)	-25	hPos (3)	15	Psi (2)	0
4	upward	Move upward 15 ft,	time (1)	35	xCrs (4)	-275	yCrs (4)	-25	hPos (3)	40	Psi (2)	0
5	turn	Yaw +30 deg	time (1)	45	xCrs (4)	-275	yCrs (4)	-25	hPos (3)	40	Psi (2)	-30
6	return	Return to original hover position	time (1)	60	xCrs (4)	-250	yCrs (4)	0	hPos (3)	15	Psi (2)	0

Figure 8. Task Example for DDG Deck Stationkeeping Maneuver.

This flight task is useful for examing the closed-loop response of the pilot-vehicle model in all axes of control. It also offers the basis for a standard maneuver that could be used for performance by a human pilot in order to obtain pilot model parameters. (Section V.C. will examine extraction of pilot model parameters for performance of this task from manned-simulator data.)

V. Pilot Model Scheme

The pilot model consists of two subsystems, the decision-making function and the controller function as shown in Fig. 9. In addition, a manual control input is available from a joystick.

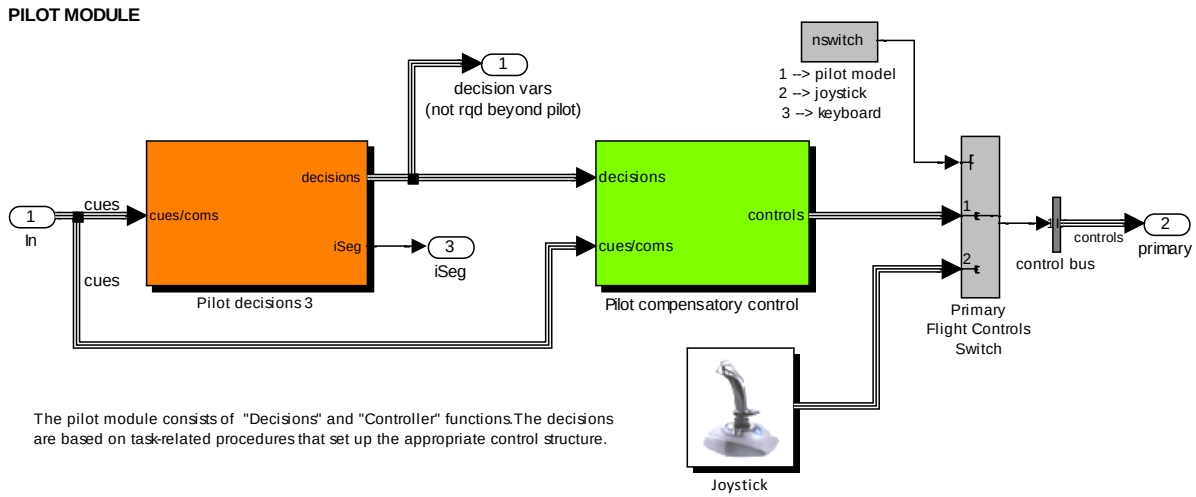


Figure 9. Pilot Model Components

A. Pilot Decision-Making Function

Pilot decisions are based on the task-defined tests of the specified transition variable against the respective pilot cue. Upon segment transition, the task-defined control mode is set to the appropriate mode and the pilot controller provides closed-loop management of the subsequent segment.

The pilot decision function is implemented in Simulink as an S-function. Its job predominantly is to monitor cues for the purpose of determining when to make a segment change and a possible change in pilot control strategy (just as a human pilot would do). Upon the segment change, the pilot decision function chooses the pilot controller mode and applies the command via the task command generator.

B. Pilot Controller Function

The pilot controller function is loosely based on the Hess structural pilot model form (Ref. 19) shown here in Fig. 10.

The actual TPV model implementation consists of a nested series of loops beginning with inner rate and attitude loops and extending to outer velocity and position loops. Fig. 11 shows a typical inner loop controller (pitch attitude).

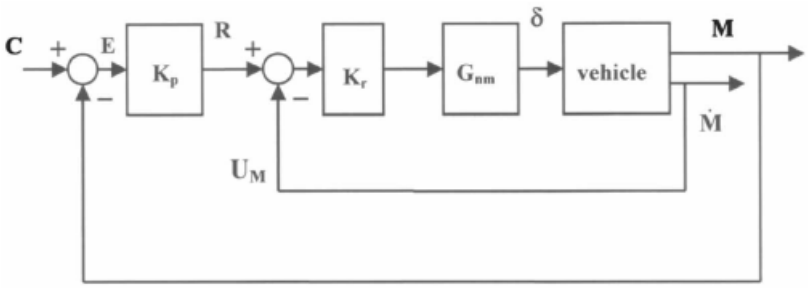


Figure 10. Hess Structural Pilot Model Form.

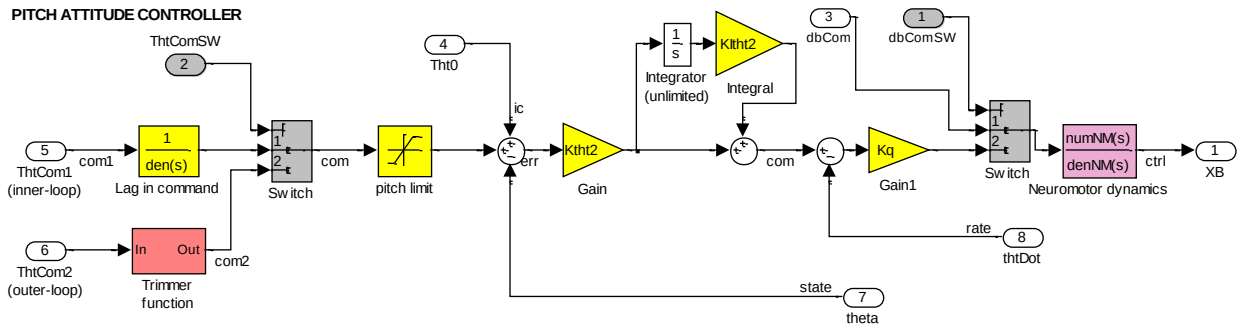


Figure 11. Typical TPV Inner-Loop Pilot Model Structure (pitch axis).

This controller structure contains a neuromuscular lag (or delay) function, an inner-loop on pitch rate, an integrator in the pitch command loop, a pitch loop gain (equal to the desired crossover frequency), a pitch command limiter, and a switch between either a pitch-attitude command or an outer-loop command of x-velocity or x-position. While this scheme provides broad flexibility in approximating human pilot behavior, it can normally be configured using only the pitch-rate and pitch-attitude gains and a first- or second-order neuromuscular lag.

An example of the corresponding outer-loop controller supported by the pitch attitude controller is shown in Fig. 12. It includes two gain elements, x-velocity and x-position.

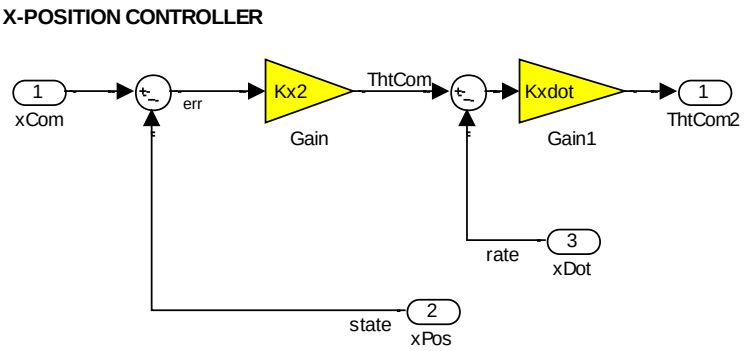


Figure 12. Typical TPV Outer-Loop Pilot Model Structure (x-axis).

Fig. 13 shows the "Pilot Model Parameters" GUI with a typical set of pilot model parameters for all axes of control. This set would normally be configured for a specific vehicle and, if desired, for a specific pilot or pilot skill set. Of course, the TPV model permits pilot parameters to be adjusted to reflect variables such as visual environment, "high-gain" vs "low-gain," skill level, or control technique.

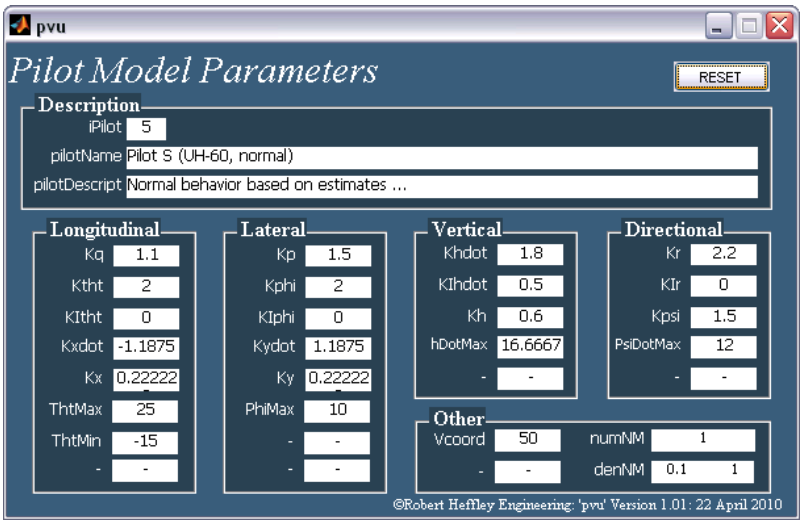


Figure 13. Example of Pilot Parameter Set as Shown by "Pilot Model Parameters" GUI.

C. Determination of Pilot Model Parameters

The pilot model is affected by noise in the cueing in order to represent a degraded visual environment (DVE). The effects of pilot skill or pilot background can be adjusted in the controller structure (e.g., use of control crossfeeds, compensation, and delay). Normally the crossover frequencies of each successive outer loop are separated by a factor of 2.5 or 3.

A collection of pilot models is assembled in a single Matlab file accessed by the TPV model, 'pilotSetup.m'. Individual pilot models are created for specific aircraft and possibly a variety of skill levels, flight tasks, visual conditions, cue availability, etc. As with the task model, pilot model parameters are defined in a Matlab cell array containing both numerical values and text descriptions.

Quantification of pilot control parameters (gains, compensation, delay, etc.) can be based on measurements of human pilot behavior or on estimates. One technique for deriving a human pilot's crossover frequency is shown in Fig. 14. This shows an intentional longitudinal position change for a skilled Navy pilot hovering above DDG deck in a manned-simulation of an SH-60 helicopter (i.e., the Stationkeeping maneuver described earlier in Fig. 7). The upper two plots show x-velocity and x-position over several seconds. The lowest plot shows a phase-plane trajectory for the rearward position change occurring at about 0 sec. Using the ratio of peak velocity to magnitude of position change (about 7ft/s/68ft = 0.10) and applying the factor of 2.4[‡] to estimate the crossover frequency yields a value of about 0.25rad/s. This value can be used directly to set the pilot's position gain Kx (also the value of crossover frequency for regulation of x-position). Note that this value also agrees well with the perceived-range model analysis presented earlier.

Using a variety of techniques such as shown above, we can develop reasonable values for pilot model gains, compensation, and command limits for each of the primary control axes. Further, based on pilot commentary, it is possible to infer changes in control strategy or technique that can be implemented directly in the TPV controller model.

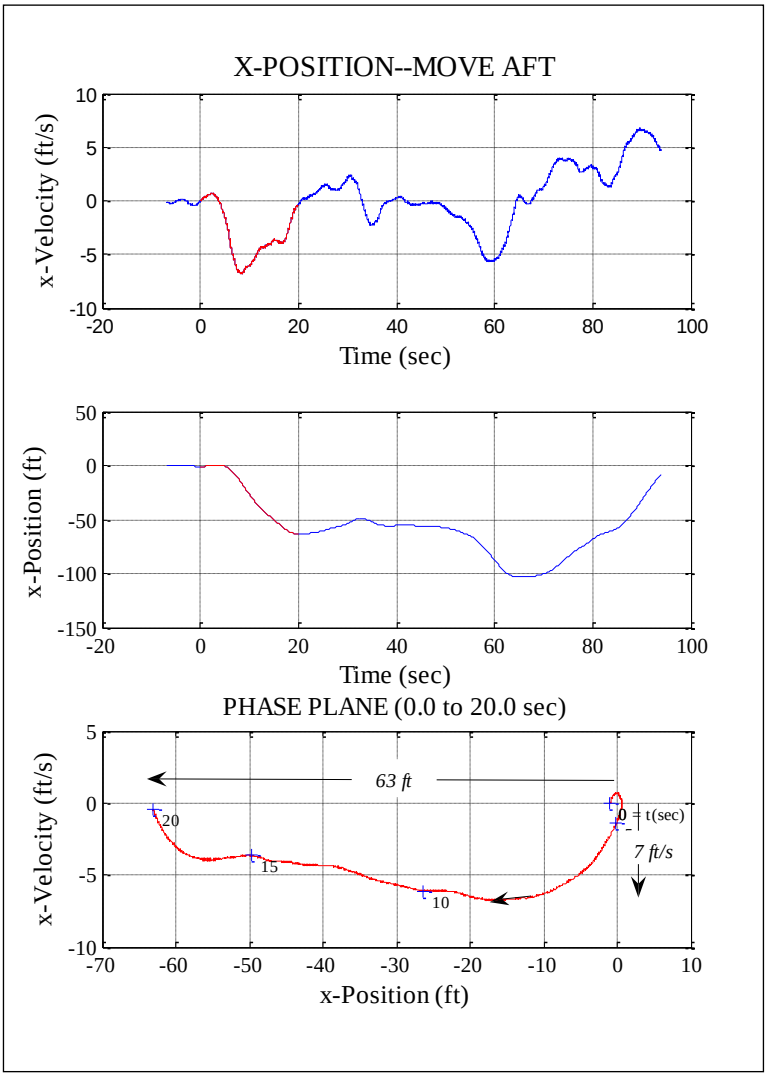


Figure 14. Summary of x-Position Response as a Human Pilot Moves Leftward in the StationKeeping Task (Reference 5).

[‡] The "2.4" factor yields an estimate of crossover frequency based on peak rate for a unit position change in an equivalent second-order system with damping ratio 0.8 (Ref. 25).

VI. Vehicle Model Accomodation

The vehicle model may be represented in any form that uses the primary control inputs from the pilot model and produces output states sufficient to generate the necessary cues for the task model. It is convenient to configure the vehicle model as shown in Fig. 15. This arrangement consists of subsystem blocks containing the flight control system, force and moment calculations, and equations of motion. However, as we shall describe, various vehicle models have been implemented using their own particular forms.

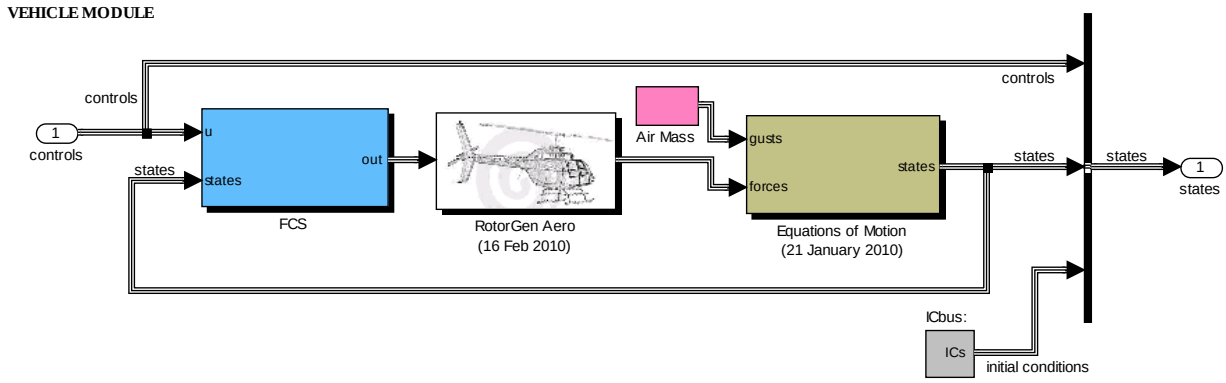


Figure 15. Vehicle Model Components

Several vehicle models have been implemented into the TPV model, including both linear and nonlinear forms. The nonlinear models have included several CASTLE aircraft models,[26] the ART *FlightLab*© model,[27] and *RotorGen2*©.[28]

CASTLE models were run by a Simulink TPV model using a Simulink S-function, "simcas," described in Ref.29. This requires UNIX and Windows connectivity supplied by Hummingbird software. In the case of *FlightLab*©, Advanced Rotorcraft Technology, Inc. created a similar S-function to link *FlightLab*© software with Simulink while both ran on a Linux platform. *RotorGen2*© runs directly on Microsoft Windows and is configured as a set of S-functions that provide either a single-rotor or tandem-rotor helicopter configuration or a fixed-wing aircraft configuration.

As a result of using the TPV model for several vehicle model forms, we have found that it is particularly useful to provide an easy substitution of models, say between a complete nonlinear aerodynamic model and a simplified linear version of the same vehicle. This allows the user to examine the performance of the more complex nonlinear model form against the simpler form in a realistic task context, thus assessing the operational simulation tradeoffs. More complex models may require substantially more computational time or CPU power while the linear form may be able to run several times faster than real time.

Note that one important advantage of the TPV simulation is that, even if the model must run slower than real time, it still allows a facsimile pilot-in-the-loop solution. A manned-simulation must always require a computer capable of a solution in real time and without excessive computational delay. Conversely, if the TPV simulation can run faster than real time, it realizes that time advantage. A beneficial consequence could be use of a Monte Carlo type of analysis performed much faster than real time.

VII. 3-D Visualization

The capability to observe directly flight task performance either from the cockpit or from the point of view of an outside observer is particularly useful. This capability is provided in the TPV model by *FlightGear*, an open-source software package.[16]

The TPV model employs *FlightGear* using the Matlab/Simulink Aerospace Blockset.[30] *FlightGear* portrays TPV model performance using the position and orientation variables output from the vehicle module. Many realistic air-vehicle models are available as downloads from the *FlightGear* website.[16] *FlightGear* also contains an array of terrain and seascape models. These models can be augmented by many other 3-D models available on the internet at low or no cost. Some models can be used without modification. In other cases, the user may wish to enhance the

models in various ways, including cosmetic appearance, alteration of cockpit instruments, or details in control surface or landing gear articulation. One example, creation of a ship environment, is shown below.

The array of ships and aircraft illustrated in Fig. 16 was created using a combination of *FlightGear* models (H-60 and CVN) and a DDG model downloaded from a no-cost source.[31] Each of these models was modified for use with the SimulinkTPV model. The *FlightGear* H-60 was repainted gray and given transparent main and tail rotor disks. The DDG was color detailed and a ship wake added. A second DDG was placed adjacent to the *FlightGear* CVN model as shown.

Figure 16. Navy Near-Ship Environment Using *FlightGear* Software—Chase View.

The runway environment used for several ADS-33E demonstration maneuvers is shown in Fig. 17. Here the pilot's view from a CH-53E cockpit is shown with Moffett Field's Hangar One on the left side.

Figure 17. Moffett Field Runway 32L Environment Using *FlightGear* Software—Cockpit View.

3-D visualization of TPV model performance provides at least two important benefits. First, it provides an immediate troubleshooting clue that may help to identify a problem in development of the TPV model or adjustment of model parameters. Thus it can be a surrogate to use of a human pilot to check out a simulation. (We expand on this in Section IX.)

Similar views are obtainable using the NAVAIR CasView software available for use with CASTLE math models. Figure 18 shows an F/A-18 on final approach to a CVN.

Figure 18. CASTLE F/A-18 At the Ramp Using CasView 3-D Software.

The second benefit is to provide an added medium of documentation, either to describe the TPV simulation or to complement the analysis that might be applied to simulation results. For example, *FlightGear* can be run with simultaneous multiple windows on a PC along with the Simulink TPV model. These windows may include forward and side views from the cockpit along with views from an external observer. When juxtaposed with time-domain and phase-plane plots of task performance and system states, a complete view of the overall system is possible.

VIII. Graphical User Interfaces

Several graphical user interfaces (GUIs) support use of the TPV simulation math model. Fig. 19 shows the GUI that enables selection of the task, pilot, and vehicle from a pre-determined collection of functions. Following selection of conditions, the user can run a task execution. In addition, the user can open the Simulink model for inspection or modification, open task, pilot, or vehicle setup files, plot results, publish a run summary, and set the Simulink model pace.

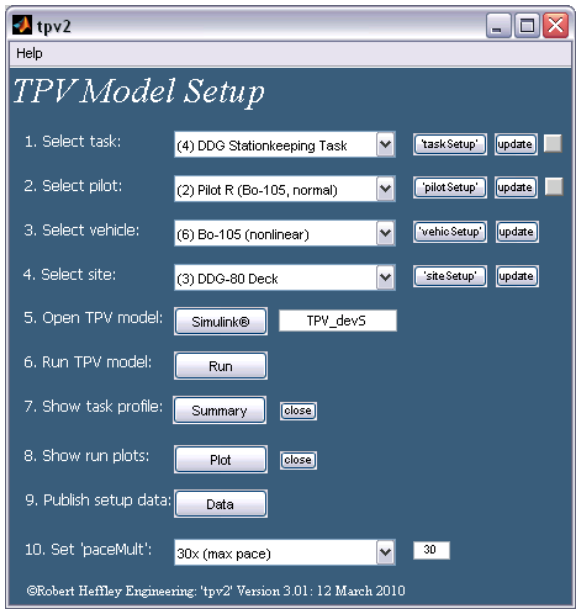


Figure 19. GUI for Selection of TPV Model Conditions.

Fig. 20 loads a *FlightGear* model to display the actions of the TPV model. The user has a wide selection of aircraft, including R44, MD500, Bo105, AH-1, UH-1, MH-60, CH-47, CH-53, MV-22, and others. Also, up to three windows can be opened to permit views from the cockpit (e.g., forward and side views) and a view from any point of an outside observer.

Figure 20. GUI for Selection of *FlightGear* Model.

The use of *FlightGear* in conjunction with the TPV model often requires the ability to manipulate position and orientation in order to rescale models or evaluate positions on the terrain model. Fig. 21 shows the utility GUI that enables direct control of position and orientation relative to any terrain benchmark (latitude, longitude, and altitude). In addition the vehicle center of rotation can be set relative to its reference datum point.

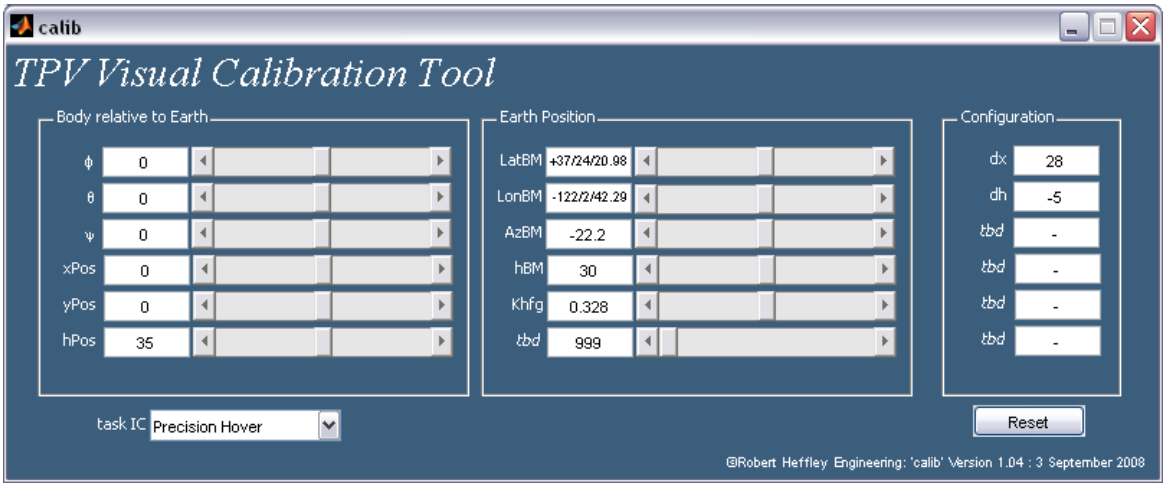


Figure 21. GUI for Calibration of *FlightGear* Model Size and Position.

IX. Vehicle Math Model Development Process Using TPV Simulaton

The Math model development process can be performed interactively by a combination of generating a candidate model, then immediately running a TPV simulation to explore its characteristics while a pilot model executes specified flight tasks or maneuvers. This process is illustrated here using the *RotorGen2* math model and its associated design software. The same general process would be applicable to other vehicle math model forms along with their own design procedures and tools, so long as there were a direct connection to the TPV model environment.

A. *RotorGen2* Model Building Procedure

The *RotorGen2* math model consists of a generic model form mainly used for rotorcraft but also adaptable to fixed-wing or other air-vehicle types. *RotorGen2* is defined using analytic functions rather than sets of lookup tables. Thus the model can be expressed as a set of parameters. Most of the parameters are associated directly with rotor-type, geometric, and mass features. Some parameters are empirical factors that aid in matching validation data. For a given vehicle all parameters are defined in a single data file.

After assembling an intial setup data file, various trim and flight-dynamics features can be rapidly generated using the GUI shown in Fig. 22. This is used to obtain trim conditions at any combination of x-, y-, and h-velocities, stability derivatives, time response, frequency response, and comparisons with available validation data. The GUI gets these results using the actual Simulink vehicle model that runs in the TPV simulation.

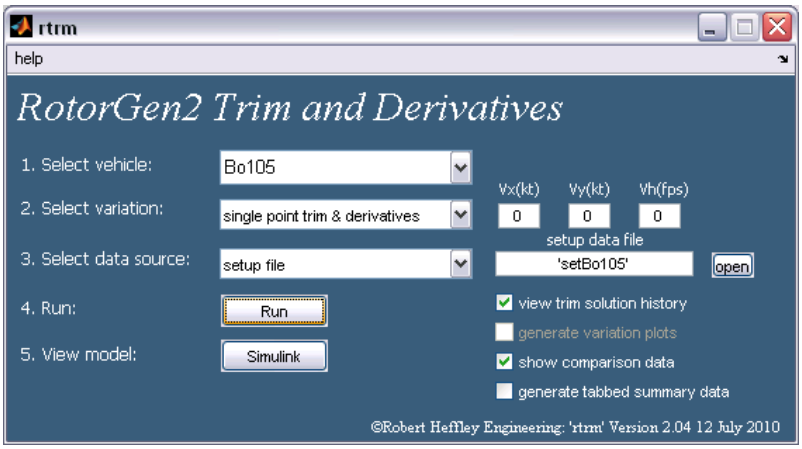


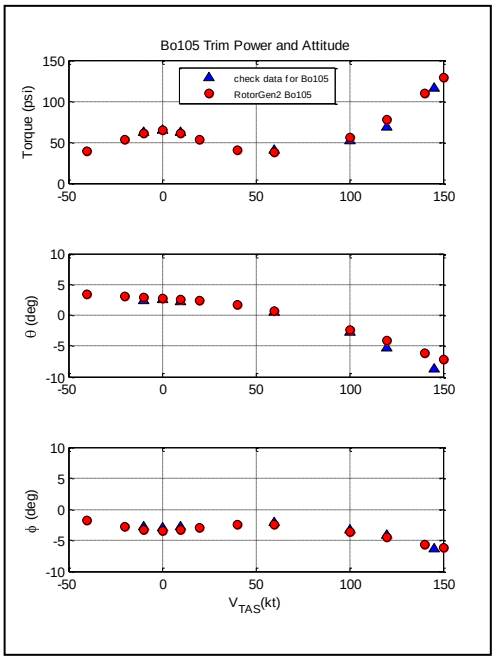
Figure 22. GUI for Generating Trim and Derivative Solutions.

B. Model Analysis Procedure

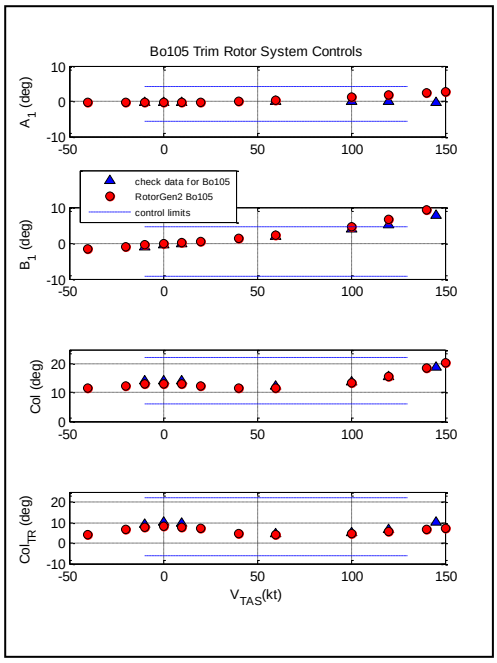
Following vehicle model parameter definition, the next step in the development process is to analyze the resulting flight dynamics in terms of trim conditions and dynamic response. This is most effective when direct comparison data are available. The GUI shown above generates quickly the following plots that summarize these characteristics and permit assesement of the quality of the math model. Immediate adjustment of model parameters is can be made, followed by reassessment of results.

Fig. 23 shows some examples of the trim summary for a range of forward speeds in level flight. The complete summary generated includes:

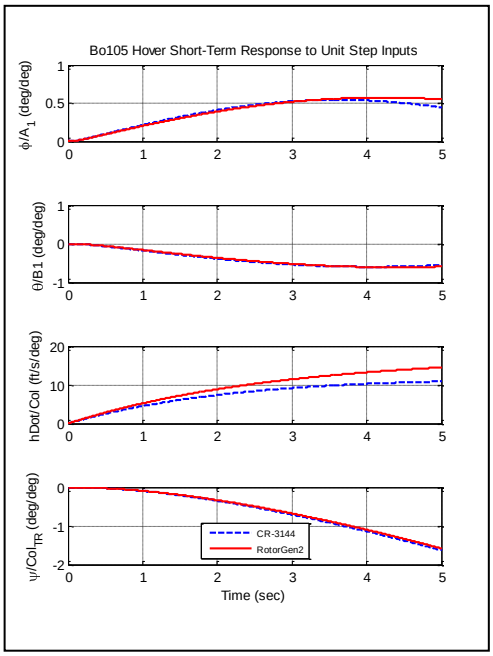
- Trim variations over x-, y-, and h-velocities,
- Time responses to control steps for all primary axes (pitch/longitudinal cyclic, heave/collective, etc.), and
- Frequency response for all primary axes (Bode plots and factored transfer functions).



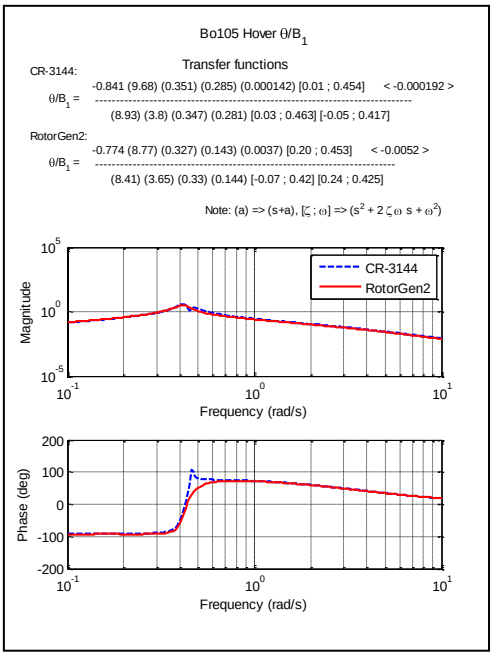
a.Trim torque and attitude.



b.Trim rotor controls.



c. Time response—primary axes.



d. Frequency response (pitch axis).

Figure 23. Partial Output from *RotorGen2* Trim and Derivatives GUI.

Upon reasonable adjustment of these characteristics, the vehicle model can be directly evaluated in performance of any desired set of flight tasks and maneuvers using the TPV simulation model as described below.

C. Flight Task Evaluation Procedure

1. Examination of Model Performing Basic Flight Maneuvers

The first TPV evaluation may be simply a basic flight maneuver that checks for reasonable adjustment of a pilot model followed by nominal testing of maneuver performance. The flight task found most useful for this is the hover stationkeeping maneuver described earlier. This involves flying a stable hover maneuver along with step changes in all four axes of control. This maneuver can be quickly run and evaluated both from the step responses easily observed in time history plots and from an external observer view point of a 3-D visualization. Fig. 24 shows a montage of the information that can be easily viewed in order to make a quick assessment of the vehicle math model behavior in a realistic flight task context. Note that for the conditions set by the TPV Model Setup GUI, three windows are presented along with Simulink time-history scopes of cockpit controls and primary state variables.

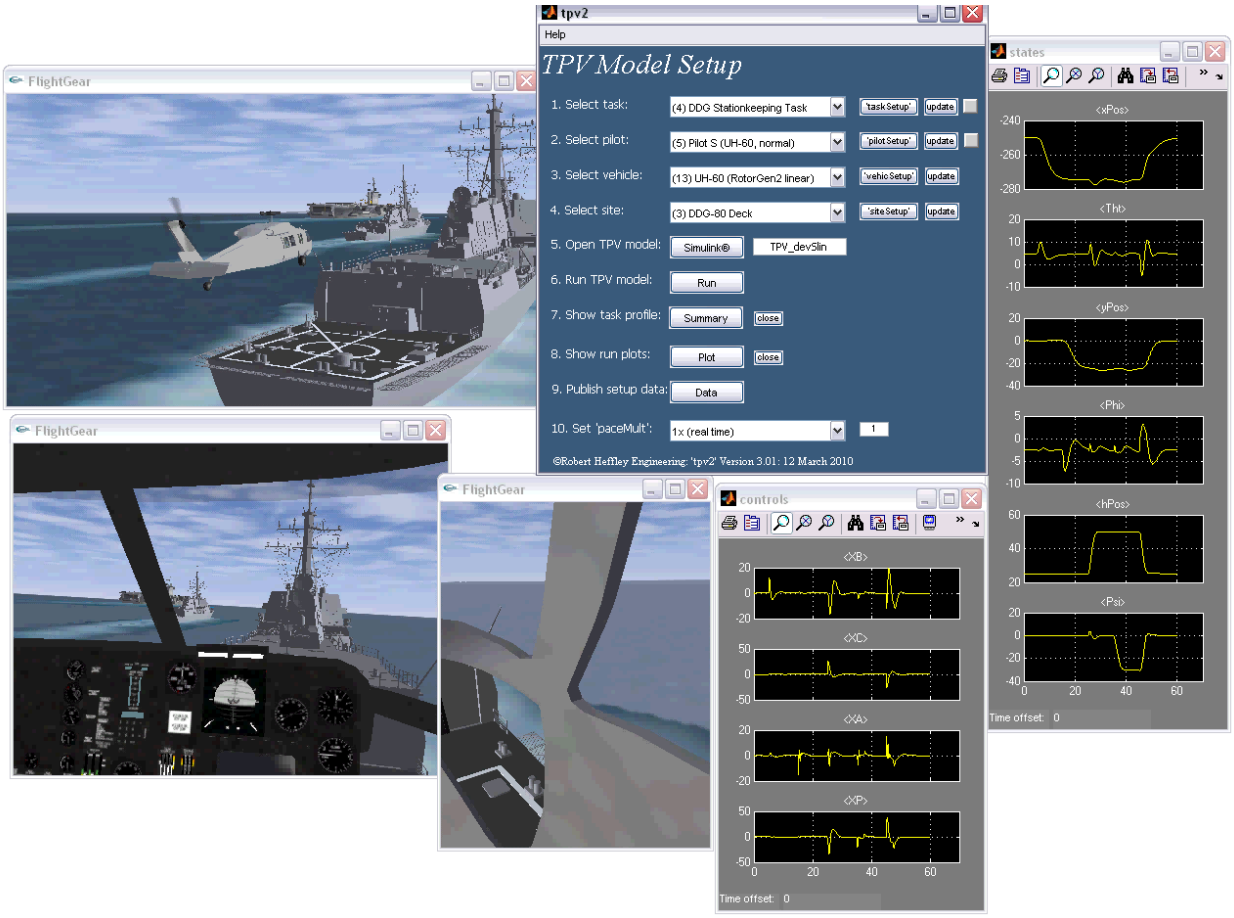


Figure 24. Quick Summary of Math Model Performance Using the TPV Simulation of a Multi-Axis Basic Flight Maneuver.

Once the vehicle model development is considered satisfactory, then other flight task assessments can be made such as we demonstrate next.

2. Comparison with Manned-Simulator or Flight Data

The TPV model software is also designed to enable direct comparison with piloted flight or simulator data. This is useful either to aid in setting task or pilot model parameters, or to permit additional analysis of flight task performance using the TPV model as an extension to manned simulation.

Fig. 25 shows a TPV model superimposed on actual piloted-simulator data for the ADS-33E precision-hover maneuver run on the NASA Ames Vertical Motion Simulator (VMS). This case involves runs made by three evaluation pilots and indicates the amount of variation that may be found. It can also show where the TPV pilot or task model may need adjustment. (For example, it appears that the TPV pilot model is managing height control too aggressively compared to the human pilots.)

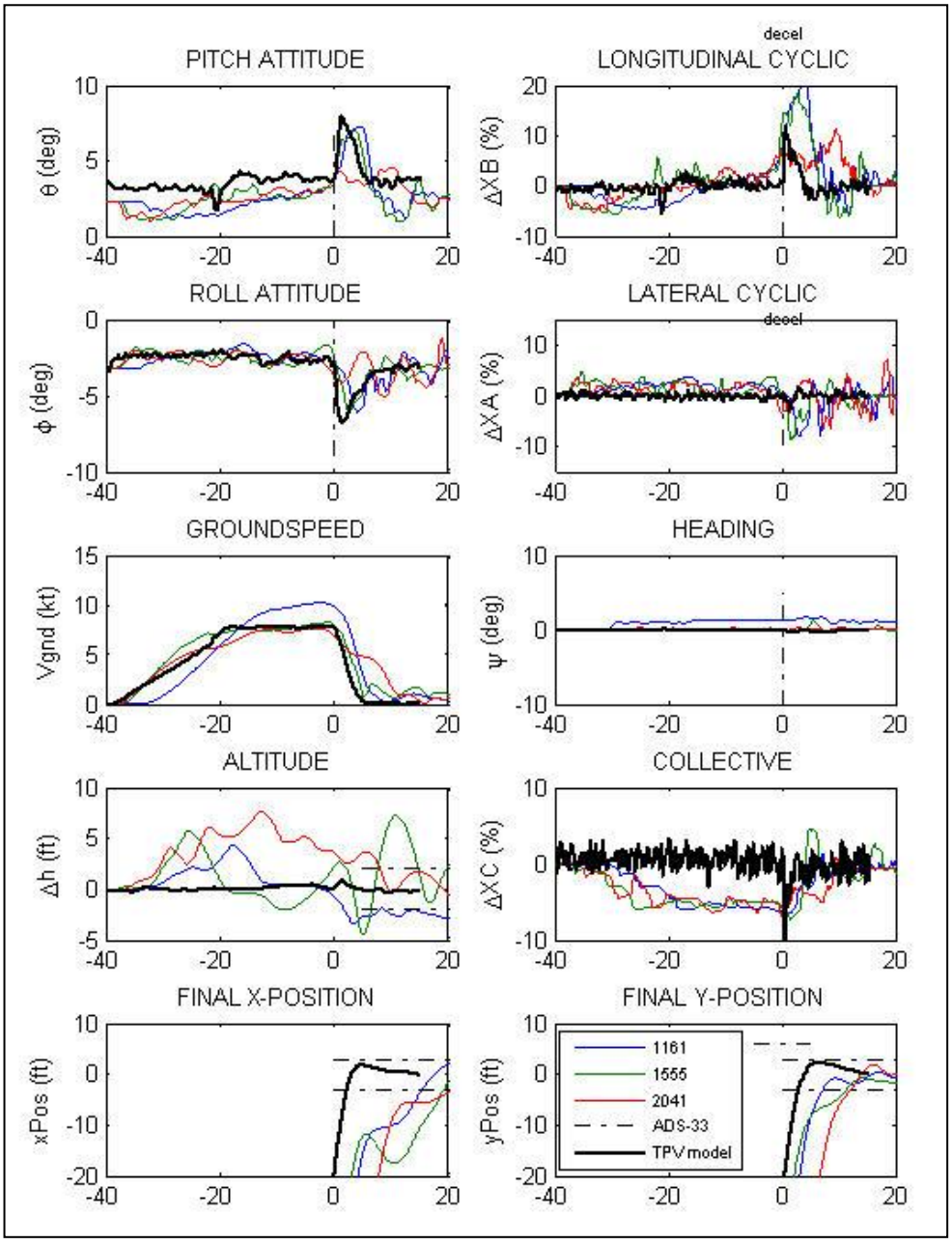


Figure 25. TPV Model Overplotted on Several Ames VMS Precision-Hover Piloted-Simulator Runs.

Another example is given in Fig. 26 with an example of the ADS-33E depart/abort maneuver performed by the TPV simulation and plotted with manned simulator data from the VMS.

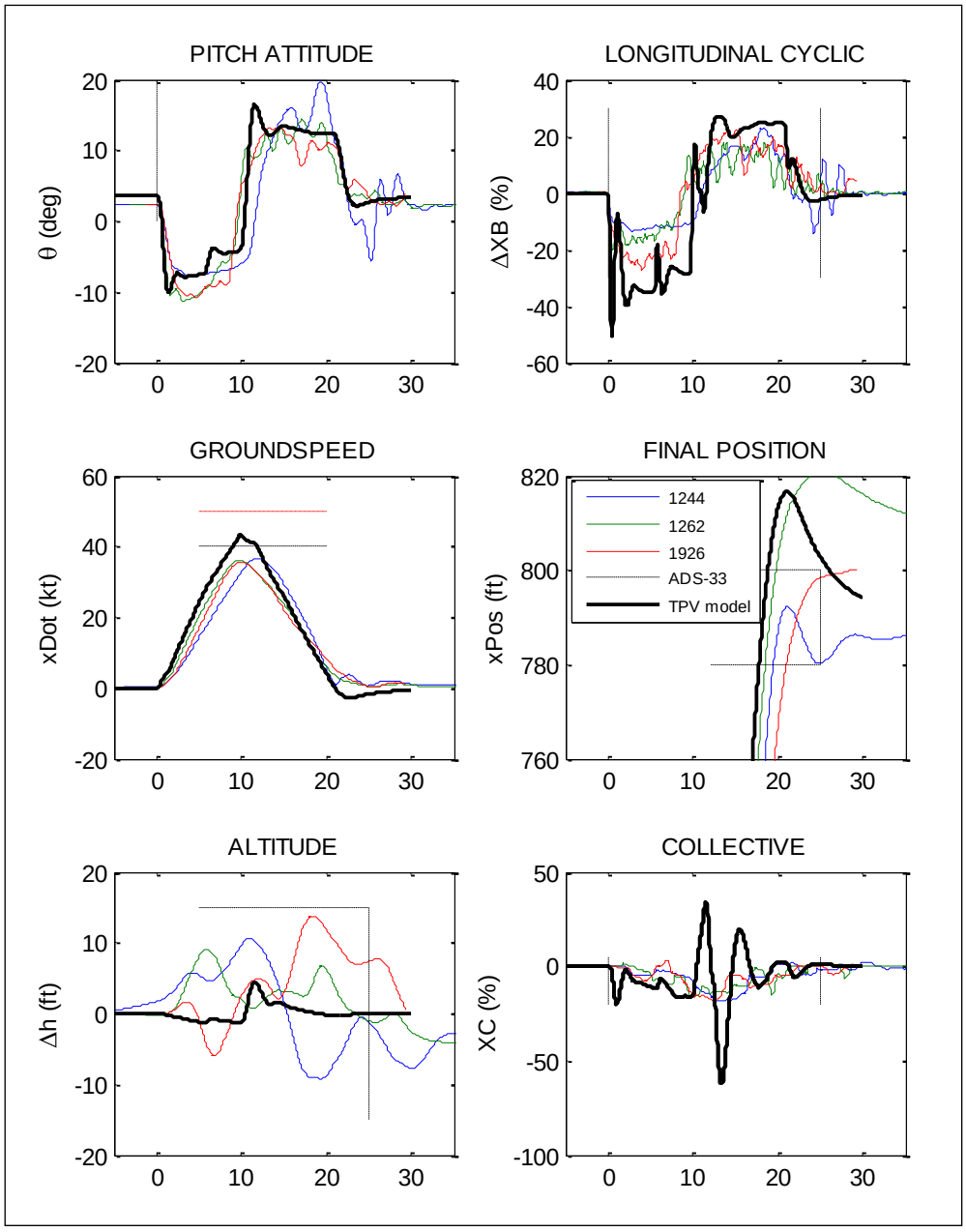


Figure 26. TPV Model Overplotted on Several Ames VMS Depart/Abort Piloted-Simulator Runs.

As a result of the wide range of vehicle model adjustment, analysis, and evaluation tools such as illustrated here, it is possible to quickly construct a math model to serve its desired function with likely probability of success. Further, this process can be done using a desktop computer workstation. The use of manned-simulator facilities can be more effectively devoted to their primary research function rather than math model development and evaluation.

X. Conclusions

The TPV simulation provides a full-context environment for exploring manned (or unmanned) flight tasks and maneuvers within the confines of a desktop computer. It transcends the limits of an open-loop vehicle simulation math model or of a pilot-in-the-loop simulation of a simple tracking task. The TPV model permits realistic simulation of complex multi-segment tasks.

The TPV model Simulink environment described in this paper includes 3D visualization as well as conventional time-history information normally available with Matlab and Simulink. The visual modality is a powerful asset not only for viewing simulation solutions but also for troubleshooting of system modification during development.

Advantages of the current TPV model include concise task and pilot definition, acceptance of a range of vehicle model forms, and an open-source image generation application, i.e., *FlightGear*. Task and pilot functions are defined by general structural forms that are set up using concise arrays. Vehicle models used to date include CASTLE, *FlightLab*©, and *RotorGen2*© plus low-order linear perturbation models. TPV models of several air-vehicle types have been demonstrated, including fixed-wing, helicopter, STOVL, and tilt-rotor

The TPV math model software is presently being used as a tool for developing and rapid prototyping of helicopter simulator math models now under development. It enables a pilot model to begin flying the helicopter math model as it undergoes refinement of aerodynamic and flight control system models. The result of such testing by a pilot model can provide an immediate indication of handling qualities, flight control deficiencies, and likely observations by a real pilot when finally run in a manned simulator environment.

XI. References

[1]Heffley, Robert K. and Ronald A. Hess, "Computer Modeling and Simulation for Helicopter Task Analysis," U.S. Army Aviation and Missile Command, USAAMCOM TR 99-D-04 (limited distribution), October 1999.

[2]Heffley, Robert K., Ronald A. Hess, and Yasser Zeyada; "Computer Modeling and Simulation for Helicopter Task Analysis," USAAMCOM TR 02-D-16 (limited distribution), June 2002.

[3]Hess, R. A., Zeyada, Y., and Heffley, R. K., "Modeling and Simulation for Helicopter Task Analysis," *Journal of the American Helicopter Society*, Vol. 47, No. 4, 2002, pp. 243-252.

[4]Heffley, Robert K., Simon M. Bourne, David Mitchell, and Ronald A. Hess, "Pilot Behavioral Modeling for Flight Operations Near Ships," RHE-NAV-2004-01 (limited distribution), March 2004.

[5]Heffley, Robert K., Simon M. Bourne, David Mitchell, and Ronald A. Hess, "Pilot Behavioral Modeling for Flight Operations Near Ships," RHE-NAV-TR 2007-1 (limited distribution), 1 May 2007.

[6]Heffley, R. K., "A Model for Manual Decelerating Approaches to Hover," *Proceedings of the Fifteenth Annual Conference on Manual Control*, AFFDL-TR-79-3134, November 1979, pp. 545-554.

[7]Heffley, R. K., Pilot Models for Discrete Maneuvers, AIAA Paper 82-1519CP, August 1982.

[8]Heffley, R. K., "A Pilot-in-the-Loop Analysis of Several Kinds of Helicopter Acceleration/Deceleration Maneuvers," *Helicopter Handling Qualities*, NASA CP 2219, April 1982.

[9]Heffley, R. K., T. M. Schulman, R. J. Randle, Jr., and W. F. Clement, "An Analysis of Airline Landing Data Based on Flight and Training Simulator Measurements," NASA CR 166404 (STI TR 1172-1R), August 1982.

[10]Heffley, R. K., "Pilot Workload Factors in the Total Pilot-Vehicle-Task System," *Proceedings of the Human Factors Society 27th Annual Meeting*, October 1983.

[11]Heffley, R. K., Pilot Workload Modeling for Aircraft Flying Qualities Analysis, NADC-82094-60, May 1984.

[12]Heffley, R. K., S. M. Bourne, and W. S. Hindson, "Helicopter Pilot Performance for Discrete-Maneuver Flight Tasks," *Proceedings of the Twentieth Annual Conference on Manual Control*, NASA Conference Publication 2341, June 1984, pp. 223-231.

[13]Moen, Gene C., Daniel J. CiCarlo, and Kenneth R. Yenni, "A Parametric Analysis of Visual Approaches to Helicopters," NASA TN D-8275, December 1976.

[14]Gilinsky, Alberta, "Perceived Size and Distance in Visual Space," *Psychological Review*, Vol. 58, 1951, pp. 460-482.

[15]Anon.; "Aeronautical Design Standard: Handling Qualities Requirements for Military Rotorcraft," ADS-33E-PRF, U. S. Army Aviation and Missile Command, 21 March 2000.

[16]Olson, Curtis L.; "Introduction to FlightGear," http://flightgear.org/introduction.html, [cited March 2010].

[17]Padfield, G. D., *Flight Dynamics*, Blackwell Publishing, Oxford, 1983, pp. 541-559.

[18]Jump, M. and G. Padfield, "Tau Flare or not Tau Flare: that is the question: Developing Guidelines for an Approach and Landing Sky Guide," AIAA -2005-6404.

[19]Hess, R. A., "Obtaining Multi-Loop Pursuit-Control Pilot Models from Computer Simulation," AIAA 2007-247, January 2007.

[20]Hess, Ronald A. and Federico Marchesi, "Analytical Assessment of Flight Simulator Fidelity Using Pilot Models," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 3, May-June 2009.
[21]Anon., *NATOPS Flight Manual Navy Model F/A-18A/B/C/D*, A1-F18AC-NFM-000, 1 Dec 1985, Change 3 – 15 June 1987.
[22]Anon., NATOPS Flight Manual Navy Model SH-60B Aircraft, A1-H60BB-NFM-000, 1 May 2000.
[23]Anon., "Helicopter Operating Procedures for Air-Capable Ships NATOPS Manual," NAVAIR 00-80T-122, 1 November 2003.
[24]Hess, R. A., and Marchesi, F., "Modeling the Human Pilot Controlling a Rotorcraft with Time-Varying Dynamics," *Proceedings of the American Helicopter Society 65th Annual Forum*, Gaylord Texan Resort, Grapevine, Texas, May 27-29, 2009.
[25]Heffley, Robert K., Simon M. Bourne, Howard C. Curtiss, Jr., William S. Hindson, and Ronald A Hess; "Study of Helicopter Roll Control Effectiveness Criteria," NASA CR 177404 (USAAVSCOM TR 85-A-5), April 1986.
[26]Anon., CASTLE Controls Analysis and Simulation Test Loop Environment Version 5.5, User's Manual," SAIC Report No. 01-1393-1548-B003, August 2000.
[27]Anon.; "FLIGHTLAB Development Software," URL: http://www.flightlab.com/flightlab.html [cited 5 July 2010] .
[28]Heffley, R. K., Synopsis of RotorGen2©, http://rhef.net/docs/Sim_modeling/Synopsis%20of%20RotorGenR3.pdf, [cited 12 March 2010].
[29]Magyar, Thomas J. and Anthony B. Page, "Integration of the CASTLE Simulation Executive with Simulink," AIAA-2001-4121, August 2001.
[30]Anon., "Aerospace Blockset 3 Data Sheet," URL: http://www.mathworks.com/products/aeroblks/, The MathWorks, September 2007.
[31]3D ModelWorks, http://www.3dmodelworks.com/ .