Activity Tracking for Pilot Error Detection from Flight Data

Todd J. Callantine,
San Jose State University/NASA Ames Research Center, MS 262-4, Moffett Field, CA 94035, USA.
tcallantine@mail.arc.nasa.gov

Abstract: This paper presents an application of activity tracking for pilot error detection from flight data. It describes the Crew Activity Tracking System (CATS), in-flight data collected from the NASA Langley Boeing 757 Airborne Research Integrated Experiment System aircraft, and a model of B757 flight crew activities. It then presents an example of CATS detecting actual in-flight crew errors.

Keywords: human error detection, activity tracking, glass cockpit aircraft

Introduction

This paper describes an application of the Crew Activity Tracking System (CATS) that could contribute to future efforts to reduce flight crew errors. It demonstrates how CATS tracks crew activities to detect errors, given flight data and air traffic control (ATC) clearances received via datalink. CATS implements a so-called 'intent inference' technology, called activity tracking, in which it uses a computational 'engineering' model of the operator's task, together with a representation of the current operational context, to predict nominally preferred operator activities and interpret actual operator actions.

CATS was originally implemented to track the activities of Boeing 757 (B757) 'glass cockpit' pilots, with a focus on automation mode errors (Callantine and Mitchell, 1994). The CATS activity tracking methodology was validated as a source of real-time knowledge about B757 automation usage to support a pilot training/aiding system (Callantine, Mitchell, and Palmer, 1999). CATS has since proven useful as an analysis tool for assessing how operators use procedures developed to support new operational concepts (Callantine, 2000). It also serves as a framework for developing agents to represent human operators in incident analyses and distributed simulations of new operational concepts (Callantine, 2001a).

The research described here draws in large part from these earlier efforts. In particular, the CATS model of B757 flight crew activities has been expanded and refined. The representation of operational context used to reference the model to predict nominally preferred activities has similarly undergone progressive refinement. And, while the idea of using CATS to detect flight crew errors from flight data is not new, this paper presents an example of CATS detecting a genuine, in-flight crew error from actual aircraft flight data.

Using CATS to detect errors from flight data has several potential benefits (Callantine, 2001b). First, CATS provides information about procedural errors that do not necessarily result in deviations, and therefore would not otherwise be reported (cf. Johnson, 2000). Second, CATS enables airline safety managers to 'automatically' incorporate information about a detected error into a CATS-based training curriculum. Other pilots could 'relive' a high-fidelity version of the context in which another crew erred. Increasing the efficiency and fidelity of information transfer about errors to the pilot workforce in this way would likely yield safety benefits. A safety-enhancement program that uses CATS to detect errors would improve training by requiring safety and training managers to explicate policies about how an aircraft should preferably be flown.

The paper is organized as follows. It first describes the CATS activity tracking methodology, and information flow in CATS. The paper then describes a CATS implementation for detecting pilot errors. It first describes flight data obtained for this demonstration from the NASA Langley B757 Airborne Research Integrated Experiment System (ARIES) aircraft. It next describes two key representations. The first is a portion of a CATS model of B757 flight operations. The second is a representation of the constraints conveyed by ATC clearances that plays a key role in representing the current operational context (Callantine, 2002b). An example from the available flight data then illustrates CATS detecting pilot errors. The paper concludes with a discussion of future research challenges. A lengthier report on this research appears in Callantine (2002a).

Activity Tracking

Activity tracking is not merely the detection of operational 'deviations' (e.g., 'altitude below glidepath'). The activity tracking methodology involves first predicting the set of expected nominal operator activities for the current operational context, then comparing actual operator actions to these

predictions to ensure operators performed correct activities. In some situations, various methods or techniques may be acceptable; therefore the methodology also includes a mechanism for determining that, although operator actions do not match predictions exactly, the actions are nonetheless correct. In this sense, CATS is designed to 'track' flight crew activities in real time and 'understand' that they are error-free. As the example below illustrates, 'errors' CATS detects include those that operators themselves detect and rapidly correct; such errors may nonetheless be useful to examine.

CATS identifies two types of errors: errors of omission, and errors of commission. It further identifies errors of commission that result when the 'right action' is performed with the 'wrong value.' CATS does not base these determinations on a 'formulaic' representation of how such errors would appear in a trace of operator activities, nor attempt to further classify errors (e.g., 'reversals'). This is because the CATS model does not represent the 'steps' of procedures explicitly as 'step A follows step B;' instead it represents procedures implicitly by explicitly specifying the conditions under which operators should preferably perform each action. CATS predicts concurrent actions whenever the current context satisfies conditions for performing two or more activities. CATS interprets concurrent actions whenever the granularity of action data identifies them as such.

Like analysis techniques that rely on a 'reflection' of the task specification in a formal model of a system (e.g., Degani and Heymann, 2000), CATS relies on a correctly functioning system to reflect the results of actions (or inaction) in its state. CATS identifies errors by using information in the CATS model that enables it to assess actions (or the lack thereof, in the case of omissions) in light of the current operational context and the future context formed as a result of operator action (or inaction). Thus, one might view the CATS error detection scheme as 'closing the loop' between a representation of correct task performance and the controlled system, and evaluating feedback from the controlled system to ensure it 'jibes' with correct operator activities. Given that the system is operating normally and providing 'good data,' this is a powerful concept.

Crew Activity Tracking System (CATS): Figure 1 generically depicts information flow in CATS, between a controlled system and CATS, and between CATS and applications based on it. CATS uses representations of the current state of the controlled system and constraints imposed by the environment (including performance limits on the controlled system) to derive the current operational context. CATS then uses this representation to generate predictions from its model of operator activities. CATS compares detected operator actions to its predicted activities, and it assesses actions that it cannot immediately interpret as matching a prediction by periodically referencing the activity model until it receives enough new context information to disambiguate possible interpretations.

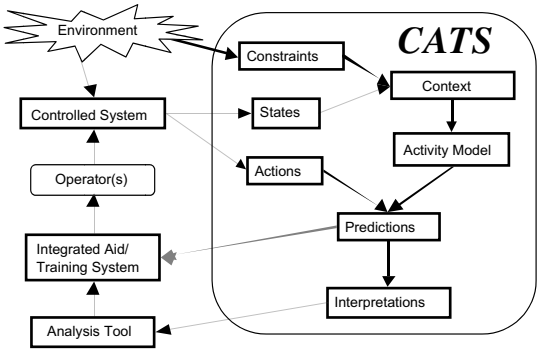

Figure 1 – Information flow within and between CATS and a generic human-machine system, with applications to error analysis, aiding, and training.

CATS Implementation for Flight Data Error Detection

The following subsections specifically describe the implementation of CATS for detecting pilot errors from flight data. The first is devoted to the flight data itself. The second illustrates a portion of the CATS model, and the third describes how CATS generates the current operational context using a representation of ATC clearance constraints. The CATS model fragment includes portions relevant to an example of CATS detecting pilot errors presented in the fourth subsection. The following subsections all assume some knowledge of commercial aviation and a B757-style autoflight system. A detailed description of the Boeing 757 autoflight system mode usage is provided in Callantine, Mitchell, and Palmer (1999); see Sarter and Woods (1995), and Wiener (1989) for discussions of mode errors and automation issues.

B757 ARIES Flight Data: The NASA Langley B757 ARIES aircraft, with its onboard Data Acquisition System (DAS), provided the flight data for this research (Figure 2). The DAS collects data at rates in excess of 5 Hz, using onboard computers that perform sensor data fusion and integrity checking. In future applications such functionality may be required within CATS, so that data can be acquired directly from aircraft data busses. Table 1 shows the collection of values that comprise the data set. The data include information from important cockpit systems. The rightmost column of Table 1 shows data CATS derives from the sampled values using filters. Included are crew action events CATS derives from the values of control states. Target value settings on the MCP are derived with 'begin' and 'end' values, as in formal action specification schemes (cf. Fields, Harrison, and Wright, 1996). The present error-detection application focuses on interactions with the autoflight system MCP,

Figure 2 – Data Acquisition System (DAS) onboard the NASA B757 ARIES aircraft (inset).

Table 1 – Available B757 ARIES data, including derived states and action events (rightmost column). The B757 ARIES DAS collects some variables from multiple sources.

Time variables	**NAV/COMM data**	**AFDS modes**	**FMC-A/T internal data**	***Derived states***
time	dme_range	fl_ch_engd	fmc_at_mach_mode_reqd	vert_speed
time1	left_dme_freq	hdg_hold_engd	fmc_at_airspeed_mode_reqd	alt_cap_engaged
time2	right_dme_freq	hdg_sel_engd	fmc_active_climb	spd_win_auto_chng
time3	left_dme_dist	land_2_green	fmc_climb_mode_reqd	ap_cmd_engd
Environmental information	right_dme_dist	land_3_green	fmc_active_cruise	***Derived MCP actions***
total_air_temp	left_vhf_freq	alt_hold_engd	fmc_con_mode_reqd	set MCP hdg
true_wind_dir	right_vhf_freq	vnav_armed_engd	fmc_crz_mode_reqd	set MCP alt
wind_speed	**FMC data**	lnav_armed_engd	fmc_active_descent	set MCP spd
AC position/attitude	fmc_target_airspeed	speed_mode_engd	fmc_display_annunc_on	set MCP mach
baro_alt	fmc_selected_altitude	thrust_mode_engd	fmc_eng_ident_1	set MCP vs
baro_corr	fmc_selected_airspeed	loc_engd	fmc_eng_ident_2	hdg sel press
flight_path_angle	fmc_selected_mach	vert_spd_engd	fmc_eng_ident_3	hdg hold press
ground_speed	fmc_crz_altitude	apprch_armed_engd	fmc_eng_ident_4	lnav press
computed_airspeed	fmc_eta	loc_armed_engd	fmc_eng_ident_5	vnav press
calibrated_airspeed	fmc_desired_track	back_course_armed_engd	fmc_eng_ident_6	spd press
mach	fmc_wpt_bearing	glideslope_engd	fmc_eng_ident_7	apprch press
magnetic_heading	fmc_cross_track_dist	**MCP Speed display status**	fmc_eng_ident_8	loc press
magnetic_track_angle	fmc_vert_dev	mcp_speed_display_blank	fmc_eng_ident_9	alt hold press
pitch_angle	fmc_range_to_alt	**Autothrottle**	fmc_eng_ident_10	vs mode press
radio_altitude	fmc_wide_vert_dev	at_armed	fmc_ga_mode_reqd	fl ch press
roll_angle	**AFDS states**	**MCP switches**	fmc_idle_thr_reqd	thrust mode press
true_track_angle	ap_cmd_ctr_engd	hdg_sel_reqd	fmc_msg_annunciated	mach toggled
iru_potential_vert_speed	ap_cmd_cen_gc_huh	hdg_hold_reqd	throttle_retard_reqd	c ap cmd switch press
hybrid_lat	ap_cmd_cen_gr_huh	lnav_reqd	pitch_speed_control_engd	l ap cmd switch press
hybrid_lon	left_ap_cmd_engd	vnav_reqd	vnav_operational	r ap cmd switch press
AC configuration/controls	ap_cmd_left_engd	spd_reqd	lnav_operational	arm autothrottles
left_engine_epr	right_ap_cmd_engd	apprch_reqd	tmc_valid	***Other derived actions***
right_engine_epr	ap_cmd_right_engd	loc_reqd	**VNAV submodes**	tune left VHF
flap_pos	ap_cmd_center_engd	alt_hold_reqd	fmc_vnav_speed_operational	tune right VHF
speed_brake_handle	ap_cws_center_engd	vs_mode_reqd	fmc_vnav_path_operational	set flaps
left_throttle_pos	ap_cws_left_engd	fl_ch_reqd	fmc_vnav_alt_operational	set spoilers
right_throttle_pos	ap_cws_right_engd	thrust_mod_reqd	**Thrust ratings**	
gross_weight	ap_in_control	**IAS/Mach toggle**	fmc_rating_1_reqd	
MCP target values	fd_c_on	mach_toggled	fmc_rating_2_reqd	
sel_mcp_altitude	fd_fo_on	**Crew Alert levels**	fmc_offset_annunciated	
sel_mcp_heading	fd_on_c	crew_alert_level_a	fmc_throttle_dormant_reqd	
sel_mcp_speed	fd_on_fo	crew_alert_level_b	fmc_thr_mode_reqd	
sel_mcp_vert_speed	**AFDS switches**	crew_alert_level_c	fmc_to_mode_reqd	
mcp_flare_retard_rate	ap_cmd_center_reqd	**Status data**	req_1_valid_resv	
sel_mcp_mach	ap_cmd_right_reqd	eec_valid	req_2_valid_resv	
MCP bank angle settings	ap_cws_center_reqd	engine_not_out		
bank_angle_lim_flaps_25	ap_cws_left_reqd			
bank_angle_lim_flaps_15	ap_cws_right_reqd			
bank_angle_lim_auto	ap_cmd_left_reqd			

so it only uses some of the available data. Also, for the present application, cockpit observations provide required clearance information.

CATS Model of B757 Navigation Activities: Figure 3 depicts a fragment of the CATS model used to detect errors from B757 ARIES data. The model decomposes the highest level activity, 'fly glass cockpit aircraft,' into sub-activities as necessary down to the level of pilot actions. Figure 3 illustrates eight actions. All actions derivable from the data are included in the full model. Each activity in the model is represented with conditions that express the context under which the activity is nominally preferred, given policies and procedures governing operation of the controlled system. The parenthesized numbers in Figure 3 refer to Table 2, which lists the 'and-or trees' that comprise these rules. For comparison to other work that considers human errors involved with CDU manipulations (e.g., Fields, Harrison, and Wright, 1997), the model fragment in Figure 3 shows just one of numerous FMS configuration tasks. However, because the B757 ARIES flight data do not include CDU data, modeling these tasks is not relevant to the present application.

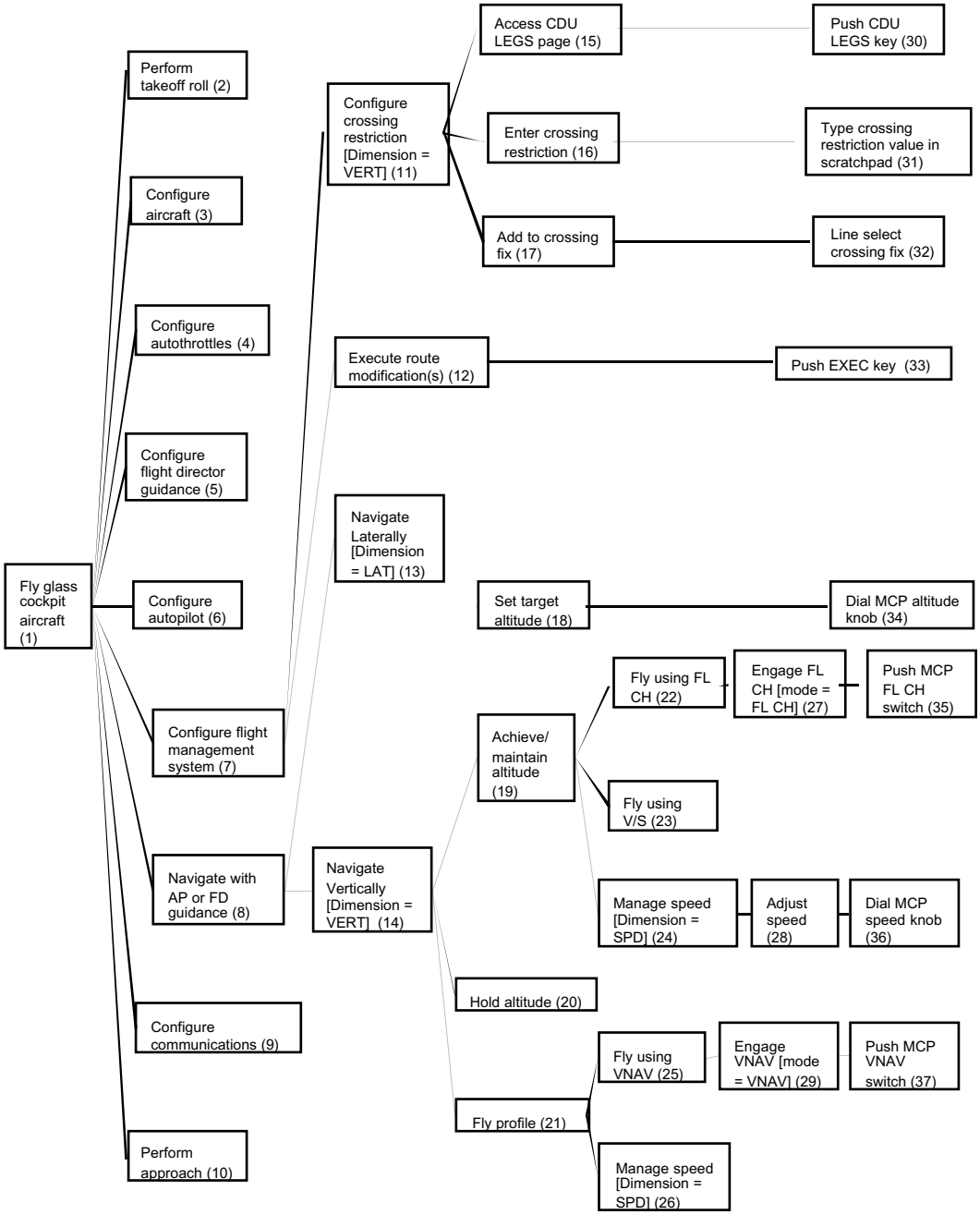

Figure 3 – Fragment of CATS model for B757 operations.

Table 2 – AND-OR trees of conditions under which the CATS model in Figure 3 represents activities as 'nominally preferred.' CATS predicts an activity when its conditions, plus all the conditions of its parent activities are satisfied by the current operational context.

(1) start-of-run
(2) (not above-runway-elevation)
(3) (and (not above-clean-speed) (not flight-surfaces-within-limits) (not gear-within-limits))
(4) (not autothrottle-armed)
(5) (not flight-director-on)
(6) [(and (not autopilot-cmd-mode-engaged) above-1000-feet-AGL)]
(7) (or (not programmed-route-within-limits) route-uplink-received)
(8) (and above-1000-feet-AGL (or autopilot-cmd-mode-engaged flight-director-on))
(9) (not comm-frequency-within-limits)
(10) (or approaching-glideslope-intercept-point approach-localizer-intercept-point)
(11) (not crossing-restriction-within-limits)
(12) route-modifications-within-limits
(13) (or autopilot-cmd-mode-engaged flight-director-on)
(14) (or autopilot-cmd-mode-engaged flight-director-on)
(15) (not cdu-page-LEGS)
(16) (and cdu-page-LEGS (not crossing-restriction-built))
(17) (and cdu-page-LEGS crossing-restriction-built)
(18) (not mcp-altitude-within-limits)
(19) (or (and (not current-altitude-within-limits) (not profile-within-limits-for-now)) expedite-needed)
(20) (and current-altitude-within-limits (not profile-within-limits-for-now))
(21) profile-within-limits-for-now
(22) (or (not altitude-close-to-target) expedite-needed)
(23) altitude-close-to-target
(24) (or fl-ch-engaged vs-engaged)
(25) profile-within-limits-for-now
(26) vnav-engaged
(27) (not fl-ch-engaged)
(28) (not target-speed-within-limits)
(29) (and (not vnav-engaged) (not capturing-required-altitude))
(30) (not cdu-page-LEGS)
(31) (not crossing-restriction-built)
(32) crossing-restriction-built
(33) route-modifications-within-limits
(34) (not mcp-altitude-within-limits)
(35) mcp-altitude-within-limits
(36) (not target-speed-within-limits)
(37) mcp-altitude-within-limits

Representation of ATC Clearance Constraints for Context Generation: Environmental constraints play a key role in defining the goals that shape worker behavior in complex sociotechnical systems (Vicente, 1999). CATS also relies on a representation of environmental constraints to construct a representation of the current operational context (see Figure 1). These factors motivated recent research on an object-oriented representation of the constraints ATC clearances impose on flight operations (Callantine, 2002b). Figure 4 shows the representation, which represents three key dimensions of constraints: vertical, lateral, and speed. CATS employs a rule base that enables it modify this constraint representation to reflect the constraints imposed (or removed) by each new ATC clearance.

CATS generates a summary of the current operational context suitable for evaluating the conditions under which activities are preferred, in order to predict activities, and for determining whether an operator action it did not expect is in error. Whenever the state or constraints change, CATS examines salient relationships to generate a set of 'context specifiers' that summarizes the current operational context; these are the descriptive clauses that appear in the conditions listed in Table 2. CATS also uses the constraint representation to maintain a record of compliance with constraints. This is important not only for context generation, but also for logging flight path deviations.

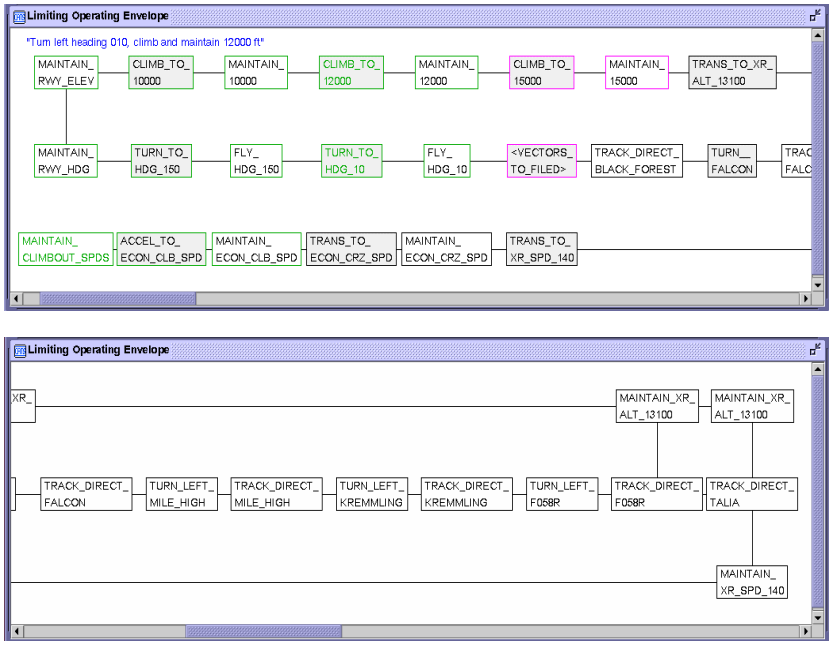

Figure 4 – Snapshot of a CATS representation of environmental constraints constructed from the filed flight plan and modified by ATC clearances.

Error Detection Example: The paper now presents an example of CATS detecting errors from B757 ARIES flight data collected during actual flight test activities. (A series of snapshots, including some of the entire CATS interface, illustrate the example.) Although the data are real, in the flight test environment, strict procedures about how the pilots should preferably fly the airplane are unreasonable. Nonetheless, by imposing the model depicted in part in Figure 3, CATS was able to detect errors, and the errors were not contrived. While the errors CATS detects are insignificant, because they in no way compromised safety, the exercise nonetheless demonstrates the viability of CATS for error detection. On the SUN Blade1000™ test platform, the CATS Java™ code processes the flight data at approximately between twelve and twenty-two times real time.

Figure 5 shows the CATS interface at the start of the scenario (Scenario Frame 1). The crew has just received a clearance to "climb and maintain 16,000 feet." CATS modifies its representation of ATC clearance constraints accordingly, and using the updated context, predicts that the crew should set the new target altitude on the MCP by dialing the MCP altitude knob.

In Scenario Frame 2 (Figure 6), a pilot instead pushes the VNAV switch. Because CATS has not predicted this action, it cannot interpret the action initially. CATS instead continues processing data. In Scenario Frame 3 (Figure 7), CATS has received enough new data to interpret the VNAV switch press action. Had the action been correct, the autoflight system would have reflected this by engaging the VNAV mode and commencing the climb. However, VNAV will not engage until a new target altitude is set. To assess the VNAV switch press with regard to the current context, in which airplane is still in ALT HOLD mode at 12,000 feet, CATS searches its model to determine if any parent activities of the VNAV switch press contain information linking the action to a specific context. CATS finds that the 'engage VNAV' activity should reflect VNAV mode engagement in the current context (see Figure 3). Because this is not the case, CATS flags the VNAV switch press as an error. Meanwhile, CATS still expects the crew to dial the MCP altitude knob.

In Scenario Frame 4 (Figure 8), a pilot does begin setting the MCP altitude. CATS interprets this action as matching a current prediction, but with an incorrect value, as the altitude setting has not yet reached 16,000. CATS does not flag this action as a 'wrong value' error, however, because it is only the start of the altitude setting. CATS continues to predict 'dial MCP altitude knob' because the context specifier 'mcp-altitude-within-limits' is not generated when the current MCP target altitude is compared to the value specified by the representation of ATC constraints (see Figure 3 and Table 2).

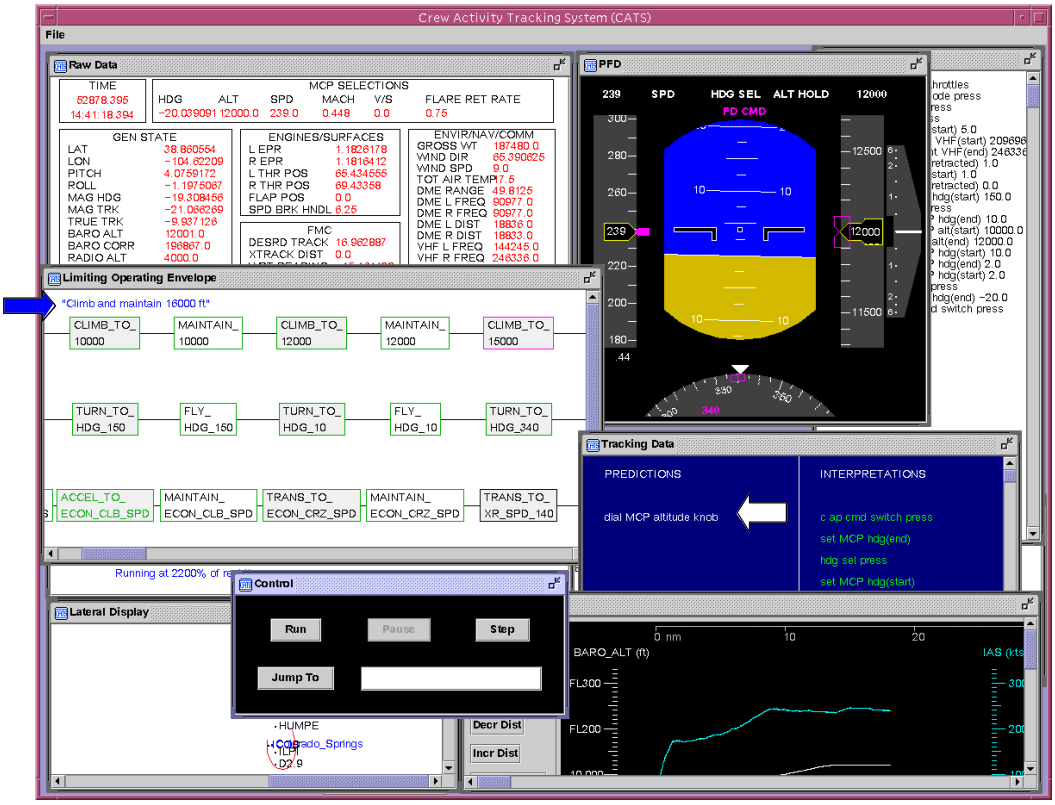

Figure 5 – Scenario Frame 1: In response to a clearance to climb, CATS predicts the crew should set the new target altitude on the MCP by dialing the MCP altitude knob.

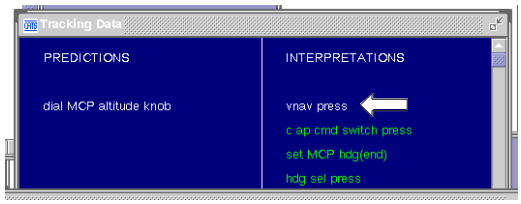

Figure 6 – Scenario Frame 2: CATS detects that a crew member pressed the VNAV switch instead of setting the MCP altitude.

In Scenario Frame 5 (Figure 9), one pilot pushes the VNAV switch a second time before the altitude setting is complete. As the other pilot completes the altitude setting, CATS interprets the end of the altitude setting action as matching its prediction. In Scenario Frame 6 (Figure 10), CATS detects that a pilot has pressed the FL CH switch (perhaps to begin the climb in FL CH mode, since VNAV did not engage). Because the MCP target altitude is now properly set, CATS predicts the crew should engage VNAV, which is preferred according to the CATS model.

CATS detects a second FL CH switch press in Scenario Frame 7 (Figure 11). Perhaps a pilot performed this action as 'insurance' to engage a mode to begin the climb. Because FL CH mode engages, and this is reflected in CATS' representation of the current context, CATS interprets both FL CH switch presses as correct acceptable alternative actions. By this time, CATS has also flagged the second VNAV switch press as an error. In the final frame of the scenario (Scenario Frame 8, Figure 12), the aircraft has begun climbing in FL CH mode. At this point the crew opts to engage VNAV mode. At last, CATS detects the predicted VNAV switch press and interprets it as correct.

Conclusions and Future Research

The above example demonstrates that CATS can detect errors from flight data. Although the errors CATS detects are inconsequential, this research indicates CATS can provide contextual information useful for disambiguating the causes of deviations or unusual control actions that arise in incident or

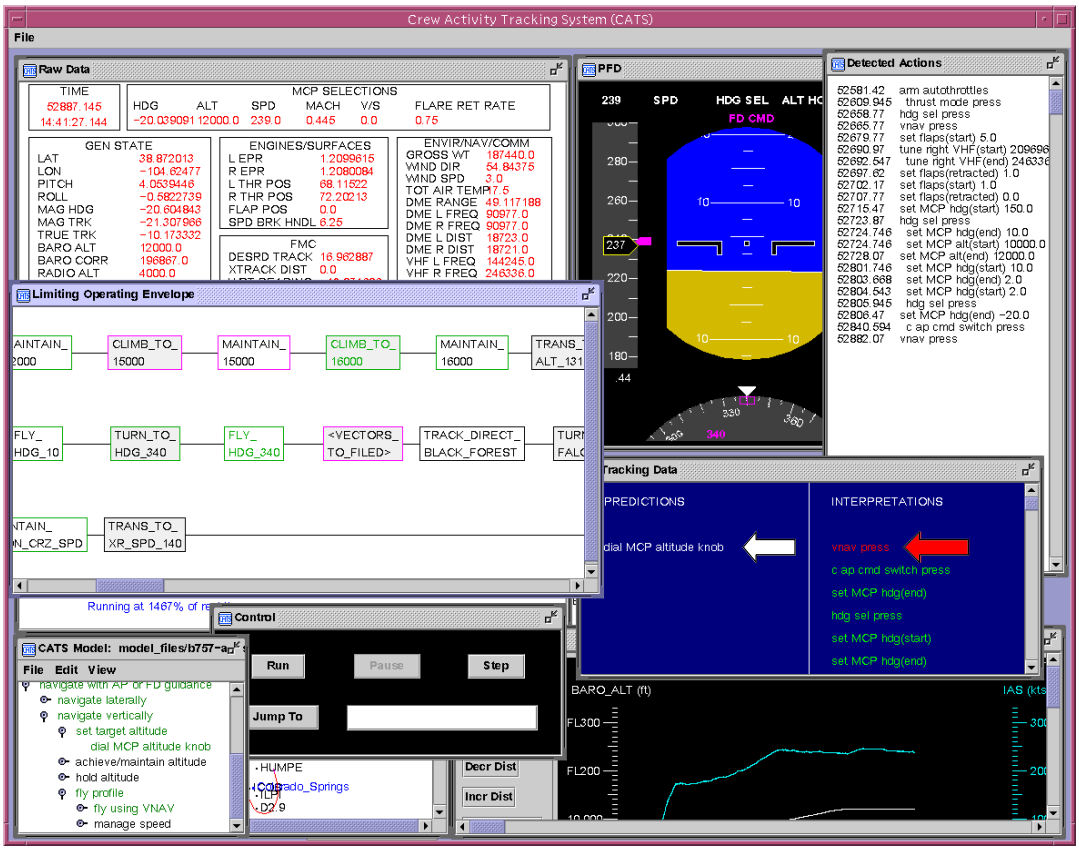

Figure 7 – Scenario Frame 3: CATS cannot reconcile the VNAV switch press with the current context, and therefore flags it as an error; CATS is still expecting the crew to dial the MCP altitude knob.

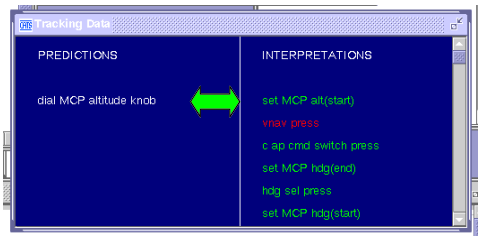

Figure 8 – Scenario Frame 4: CATS detects a pilot starting to dial the MCP altitude, and interprets it as matching its prediction, but with the wrong value (not an error, because the action is only the start of the altitude setting).

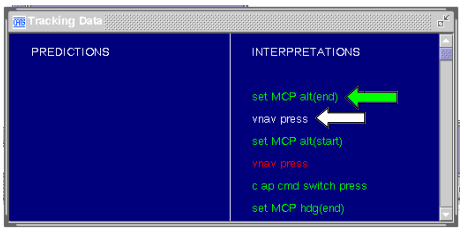

Figure 9 – Scenario Frame 5: A second VNAV switch press, before the altitude setting is finished.

accidents. Discoveries made using CATS can be incorporated into training curricula by connecting a CATS-based training system to a simulator and allowing pilots to 'fly' under conditions that correspond the actual context of an error-related event. Such capabilities are also useful outside the airline arena as they support both fine-grained cognitive engineering analyses and human performance modeling research.

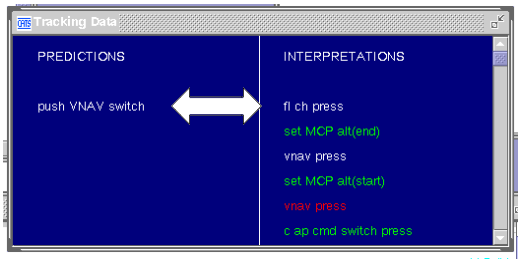

Figure 10 - Scenario Frame 6: CATS detects that the crew has now opted to engage FL CH mode by pressing the FL CH switch. But because the altitude is now properly set, CATS now predicts the crew should push the VNAV switch to engage VNAV (the preferred mode according to the CATS model).

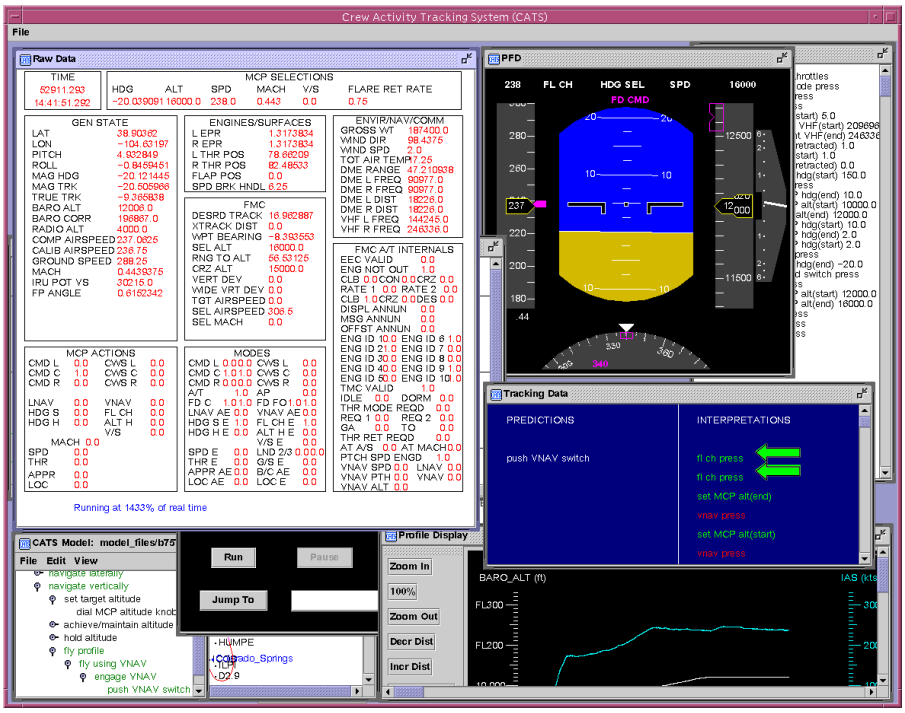

Figure 11 - Scenario Frame 7: CATS detects a second 'insurance' FL CH switch press, and interprets it as acceptable as it did the first FL CH switch press.

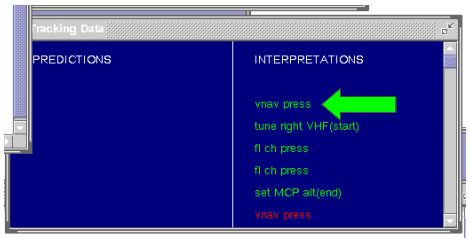

Figure 12 - Scenario Frame 8: The crew opts to engage VNAV; CATS detects the predicted VNAV switch press and interprets it as correct (elapsed time from Scenario Frame 1 is ~42 secs).

Using CATS with flight data collected at 'continuous' rates results in better performance. Event-based data, such as those available from the NASA ACFS, require more complicated interpolation methods to avoid temporal 'gaps' in the CATS representation of context that can adversely affect CATS performance. Important directions for further research involve improving the coverage of flight data to include the FMS and CDUs, as well as work on methods to automatically acquire ATC clearance information. This research indicates that, if CATS has access to data with full, high-fidelity coverage

of the controlled system displays and controls, it can expose the contextual nuances that surround errors in considerable detail.

Acknowledgements
This work was funded under the System Wide Accident Prevention element of the FAA/NASA Aviation Safety Program. Thanks to the NASA Langley B757 Flight Test team for their assistance with data collection.

References
Callantine, T. (2000). A glass cockpit crew activity analysis tool. SAE Technical Paper 200-01-5522. Warrendale, PA: SAE International.

Callantine, T. (2001a). Agents for analysis and design of complex systems. *Proceedings of the 2001 International Conference on Systems, Man, and Cybernetics,* October, 567-573.

Callantine, T. (2001b). The crew activity tracking system: Leveraging flight data for aiding, training, and analysis. *Proceedings of the 20th Digital Avionics Systems Conference*, 5.C.3-1—5.C.3-12 (CD-ROM).

Callantine, T. (2002a). Activity tracking for pilot error detection from flight data. NASA Contractor Report 2002-211406, Moffett Field, CA: NASA Ames Research Center.

Callantine, T. (2002b). A representation of air traffic control clearance constraints for intelligent agents. *Proceedings of the 2002 IEEE International Conference on Systems, Man, and Cybernetics*, Hammamet, Tunisia, October.

Callantine, T., and Mitchell, C. (1994). A methodology and architecture for understanding how operators select and use modes of automation in complex systems. *Proceedings of the 1994 IEEE Conference on Systems, Man, and Cybernetics*, 1751-1756.

Callantine, T., Mitchell, C., and Palmer, E. (1999). GT-CATS: Tracking operator activities in complex systems. NASA Technical Memorandum 208788, Moffett Field, CA: NASA Ames Research Center.

Degani, A. and Heymann, M. (2000). Some formal aspects of human automation interaction. NASA Technical Memorandum 209600, Moffett Field, CA: NASA Ames Research Center.

Fields, R., Harrison, M., and Wright, P. (1997). THEA: Human error analysis for requirements definition. Technical Report 2941997, York, UK: University of York Computer Science Department.

Fields, R., Wright, P., and Harrison, M. (1996). Temporal aspects of usability: Time, tasks and errors. SIGCHI Bulletin 28(2).

Johnson, C. (2000). Novel computational techniques for incident reporting. In D. Aha & R. Weber (Eds.), *Intelligent Lessons Learned Systems: Papers from the 2000 Workshop* (Technical Report WS-00-03), Menlo Park, CA: AAAI Press, 20-24.

Sarter, N., and Woods, D. (1995). How in the world did we ever get into that mode? Mode error and awareness in supervisory control. *Human Factors*, 31(1), 5-19.

Vicente, K. (1999). *Cognitive work analysis: Toward safe, productive, and healthy computer-based work.* Mahwah, NJ: Lawrence Erlbaum Associates.

Wiener, E. (1989). The human factors of advanced technology ("glass cockpit") transport aircraft. NASA Contractor Report 177528, Moffett Field, CA: NASA Ames Research Center.