SAE TECHNICAL PAPER SERIES

1999-01-1434

A Method for Accident Reconstruction and Neighborhood Analysis Using an Autonomous Situational Model of Flight and Flight Recorder Data

Ivan Y. Burdun

[obsolete affiliation information omitted]

Reprinted From: Proceedings of the 1999 Advances in Aviation Safety Conference (P-343)

The Engineering Society For Advancing Mobility Land Sea Air and Space®
INTERNATIONAL

Advances in Aviation Safety Conference and Exposition
Daytona Beach, Florida
April 13-15, 1999

ISSN 0148-7191

Printed in USA

1999-01-1434

A Method for Accident Reconstruction and Neighborhood Analysis Using an Autonomous Situational Model of Flight and Flight Recorder Data

Ivan Y. Burdun

[obsolete affiliation information omitted]

ABSTRACT

Flight accidents with modern aircraft are often a result of complex dynamics of the "pilot (automaton[1]) - vehicle - operational environment" system. When a "critical mass" of the system's complexity exceeds a certain level, a "chain reaction" of irreversible cause-and-effect links can be spontaneously triggered in the system behavior leading to a catastrophe. An affordable, practically tested technique is proposed to complement current methods of flight accident analysis. A generic situational model of the system behavior and a computer a[illegible] virtual test article. This model include[illegible] freedom non-linear flight dynamics mod[illegible] ational pilot model ("silicon pilot"), models of anticipated operational factors (conditions), and a tool for flight scenario planning. Available flight recorder data are used to tune the model and reconstruct the accident. Then the model is used for in-depth examination of the accident's "neighborhood" in autonomous "what-if" simulation experiments under actual and hypothetical conditions. The latter may include pilot errors, piloting tactics variations, onboard system's failures and errors, and weather conditions, as well as combinations of these factors. Programming and piloting skills are not mandatory for the user. Potential applications include: flight accident investigation under uncertainty, advanced pilot training, research into aircraft practical aerodynamics, design of automatic flight control systems, and onboard AI technologies for flight safety.

RESEARCH TASK

PROBLEM – Flight accidents are often a result of complex dynamics and negative interactions in the "pilot (automaton) - vehicle - operational environment" system. Blaming a single operational factor for an accident, such as "human error" or "mechanical failure", would mean to underestimate the problem. A side effect of such reactive, "mono-factor" approaches to flight safety is a danger of reoccurrence of past accident patterns in the future. Flight accidents with modern aircraft can be explained using the notion of "chain reaction"[2]. When a "critical mass" of the system's complexity exceeds a certain level, "chain reaction" a of several interrelated events and processes[3] can be spontaneously triggered in flight. Flight automation based on human- or computer-centered principles can make potentially catastrophic links in the system dynamics even less predictable and manage-

[obsolete affiliation information omitted]

[illegible]e two goals of flight accident analysis. [illegible] to identify a cause-and-effect chain responsible for a catastrophe. The second one is to develop a proactive, physics-based remedial strategy for prevention or resolution of the given *and* similar accident patterns in the future. To achieve these goals, adequate analytical methods are required. Manned simulations, flight testing, flight dynamics analysis and formal logic methods used for accident examination have inherent limitations. This explains in part the fact that the causes of several aviation catastrophes remain unknown [2, 3].

SOLUTION APPROACH – An affordable, practically tested technique is proposed for flight accident analysis under uncertainty. A generic model of the "pilot (automaton) - vehicle - operational environment" system behavior and a computer (PC) are employed as a virtual test article. This model includes a situational pilot model ("silicon pilot"), models of key (anticipated) operational factors of flight, a tool for flight scenario planning, and a non-linear six-degree-of-freedom model of the vehicle motion. Flight records and other source data are used to tune the model, identify unknown weather conditions and reconstruct the accident's profile with the assistance of the "silicon pilot" and scenario planning tool. Then, this model can be used for detailed examination of a complex situa-

1. control mechanism designed to follow automatically a predetermined sequence of operations or respond to encoded instructions
2. irreversible propagation of strong cause-and-effect links in the system behavior [4, 1, 6]
3. which are not critically dangerous alone

tional sub-domain around the accident (accident's "neighborhood") in autonomous[4] simulation experiments. The objective is to assess the sensitivity of the accident scenario to key operational conditions in a systematic way. These factors may include pilot errors, piloting tactics variations, onboard system's failures and errors, weather effects, as well as combinations of these conditions.

BENEFITS – As the result, various operational hypotheses concerning a complex flight accident can be checked in detail. A distinguishing feature is that this task can be accomplished by a non-pilot and without a flight simulator. Programming skills are not mandatory for the user. However, a pre-requisite for the use of this method is the availability of a comprehensive database of the vehicle's input characteristics. In simulation experiments with the model, real, hypothetical and mixed scenarios can be modeled in detail, quantitatively evaluated and stored in compact formats on a computer for future reuse. Thus, possible alternative developments of the accident and its "neighborhood" can be quantified and compared. Potential application domains include flight accident investigation under uncertainty conditions; advanced pilot training; research into aircraft applied aerodynamics, automatic flight control systems, and onboard AI technologies for flight safety [4, 6].

PAPER CONTENT – In this paper, a generic method is proposed for quantitative and qualitative analysis of a flight accident under uncertainty based on an autonomous flight situation model. A system of input data structures of the model is presented in detail. Main research steps of the analysis method are briefly described. A case study of a severe flight accident, which has been examined by means of this method, can be found in [4]. This paper is addressed to researchers, managers and students working in the sector of flight safety enhancement.

INTRODUCTION TO METHOD

In this section, general concepts of the method will be introduced.

DEFINITIONS – The *system* under examination is a "pilot (automaton) - vehicle - operational environment" system. The main *operational factors* (or conditions) of flight, which may contribute to flight accidents, are associated[5] with the three major constituents of the system, namely: human pilot (or automaton), vehicle and its subsystems, and external operational environment. Therefore, flight accident analysis should be performed on the system level. The *subject* of analysis are the system dynamics under complex (multi-factor) flight situations.

The *flight situation* can be defined as a recognizable fragment of flight, which lasts from several seconds to several minutes, depending on the scope of analysis and vehicle mission. Each flight situation has specific objectives, logic and operational content, which require coherent pilot control tactics. Normally, a flight situation is associated with (and thus can be named after) a distinctive phase, stage or mode of flight. It also includes key demanding operational conditions (factors) affecting the situation. The *complex (multi-factor) flight situation* is a flight situation, which incorporates several interacting demanding conditions (factors).

The *"chain reaction" situation* is a complex flight situation with quick and irreversible propagation, towards a catastrophe, of strong cause-and-effect links between several component factors [4, 1]. Under such chaining conditions any subsequent control input may become inadequate or inefficient. Therefore, the *flight accident* can be considered as a complex or "chain reaction" flight situation. One of the requirements to the flight accident analysis process is to have a capability of generating comprehensive knowledge of complex system dynamics in an accident in the presence of several operational factors. As a "knowledge generator", an autonomous flight situation model is proposed.

AUTONOMOUS FLIGHT SITUATION MODEL – The *autonomous flight situation model* is a system of generic algorithms and data structures, which are designed for modeling and simulation of the behavior of the "pilot (automaton) - vehicle - operational environment" system under standard and complex flight situations. In this process, a human pilot and a flight simulator are not required. The model consists of the following main components:

- a situational pilot model ("silicon pilot")
- a tool for automated planning and execution of flight scenarios in simulation experiments
- mathematical models of key operational factors of flight (pilot errors, rain and wind conditions, mechanical failures, runway surface condition, atmospheric condition, and some other), and
- a six-degree-of-freedom non-linear mathematical model of the vehicle motion.

As the result of such synthesis, the model is capable of adequate description and flexible computer simulations of complex system dynamics under various demanding operational conditions. Again, a pilot and a flight simulator are not required.

INPUT REQUIREMENTS – A pre-requisite for successful application of this method is the availability of a six-degree-of-freedom non-linear mathematical model of the vehicle motion under anticipated conditions. In particular, the following groups of input characteristics are required: aerodynamic characteristics, stability and control derivatives; moments and products of inertia; engine data

4. the autonomous flight simulation experiment means that neither a pilot nor a flight simulator are required [1]
5. except for the acts of terrorism and other *force majeure* situations

(thrust characteristics, including reversed thrust, and fuel consumption, etc.); specifications of the automatic flight control system (sensors, logic, actuators, effectors, etc.); landing gear characteristics (shocks, wheels, brakes, control, etc.).

PURPOSE AND TASKS – The *purpose* of the proposed method is to help identify and mitigate “chain reaction” situations in the behavior of the “pilot - vehicle - operational environment” system based on autonomous computer simulation experiments and past flight accident patterns.

There are two tasks in this analysis process. The *first task* is to reconstruct the accident and calculate its characteristics, using the autonomous flight situation model and flight recorder data. The objective is to reveal an invariant, physics-based causal pattern (scenario) of the accident. The *second task* is to examine the system dynamics under hypothetical “neighboring” flight situations. The objective here is to assess the sensitivity of the accident scenario(s) to various demanding operational conditions and thus reveal a "what-if" (branching) structure of accident's "neighborhood". The reconstructed flight accident and the cause-and-effect structure of its “neighborhood” constitute the output of the flight accident analysis process. Finally, a subset of safe situations, which are sufficiently robust to anticipated fluctuations in key operational conditions, can be proposed as a basis for recovery tactics.

PREVIOUS APPLICATIONS – In 1985-98 more of this method had been applied to study about 35 problems in flight safety related fields for 17 aircraft types and three design projects, including airplanes, helicopters, a tilt-rotor aircraft and an aerospace vehicle [1, 5].

TECHNICAL CAPABILITIES – A list of capabilities of the autonomous flight situation model includes:

- simulation of various actual, hypothetical, and mixed flight cases with a required degree of accuracy and detail[6]
- flexible planning of various complex flight scenarios
- "what … if ... ?" flight experimentation capability
- fast tuning on to new, "neighboring" or derivative situations without the necessity to recompile software
- simulation of a given flight scenario in exact detail or with modifications at any time in the future
- identification of key operational conditions of flight (e.g.: wind shear, heavy shower)
- virtual 'freezing' of selected system state and control variables to check work hypotheses
- autonomy and independence (flight simulator hardware, programming and piloting skills are not required).

6. provided that a comprehensive aerodynamic and other input data base of the vehicle is available

FLIGHT SCENARIO DATA SYSTEM

In this section, a system of definitions and unified data structures, which constitute the input of the autonomous flight situation model (the flight situation scenario), are described. The level of description is sufficient for professional planning and execution of simulation experiments with the model on a computer.

MAIN OBJECTS – A list of main data objects of the autonomous flight situation model includes the following[7]:

- flight variable (*v*)
- flight event (**E**)
- flight process (Π)
- elementary flight situation (**s**`)
- piloting task (**T**)
- system state observer (**O**)
- control procedure (**P**)
- onboard system’s failure (**F**)
- rain type process (**R**)
- wind type process (**W**)
- time-history type process (**H**)
- runway surface condition (**Y**)
- flight situation scenario (**S**).

These objects are sufficient for comprehensive modeling and simulation of complex flight situations, including accidents. The rest of this section contains a detailed description of these objects with examples, as well as the relationships between them.

FLIGHT VARIBLE – The *flight model variable* (flight variable, model variable, or system variable), *v*, is a time-dependent parameter, which describes a certain aspect of the system behavior. Flight variables can be grouped as follows:

- numeric, symbolic, fuzzy, linguistic, etc. (mathematical classification)
- discrete and continuous (classification by time occurrence)
- vehicle dynamics, flight control, airborne system functions and failures, external conditions (classification by system components).

Examples are as follows: altitude, airspeed, TAS, IAS, wing AoA, Euler parameter e_0, roll acceleration (in stability axes), g-factor, wind gradient, horizontal distance, flap position, heading angle, last flight event, wheels position flag, yawing moment coefficient due to thrust asymmetry (body axes), left-hand gear shock absorber displacement, elevator deflection due to autopilot, total lift coefficient (body axes), rain intensity, horizontal wind component (earth axes), left engine thrust (body axes), roll rate (stability axes), roll acceleration (earth axes),

7. for other objects ref. [4]

atmospheric pressure, etc. Note that the nomenclature of variables produced in the autonomous model is much broader than a set of variables recorded in a test flight.

In the system component classification, main vectors of flight variables are: $\boldsymbol{x} = (x^1, ..., x^p)$ - vehicle dynamics, $\boldsymbol{u} = (u^1, ..., u^q)$ - flight control, and $\boldsymbol{w} = (w^1, ..., w^r)$ - external (weather) conditions. The *vocabulary of flight variables* is represented as an ordered set, $V = \{ v^1, ..., v^k, ..., v^{N(V)} \}$. Note that in practical applications $N(V) \in \{200, ... , 1000\}$.

The frame-specification of a flight variable from V includes its code, minimum and maximum values (for displaying and checking purposes), name, measurement unit, coordinate system (if applicable), definition, and some other attributes:

$$R[v^j] = \{ i,\ v_{min}, v_{max}, Nm, Un, Sys, Def,\ \dots \} \quad (1)$$

For example, $R[v^{22}]$ = { 22, -25.0, 25.0, roll_rate, degr/s, body, "rate of change of bank angle" }. This frame describes a flight variable v^{22}, $v^{22} \equiv p_b$, which is the roll rate measured in body reference axes in degrees per second. This variable has the code 22 in the vocabulary V. It can be depicted as a graphic time-history using the scale from -25.0 to 25.0 deg/s.

When planning a flight variable, it is important to remember about its physical unit and reference frames if applicable.

FLIGHT EVENT – The *flight event*, **E**, is a special state of the system, which indicates a noticeable change in the current flight situation. Events are important to the pilot or an automatic control system as they are used to plan and modify flight scenarios and control tactics. Events are essentially discrete components of the flight situation model; an event lasts from a fraction of second to one-two seconds. A list of flight event examples follows (subscripts stand for event codes):

{ $\mathbf{E}_1$: "*situation start"*, $\mathbf{E}_3$: *"speed VR achieved"*, $\mathbf{E}_4$: *"pitch 10°"*, $\mathbf{E}_6$: *"altitude 1,200 ft"*, $\mathbf{E}_{13}$: *"altitude 30 ft"*, $\mathbf{E}_{11}$: *"touchdown"*, $\mathbf{E}_{15}$: *"left wheel off the runway"*, $\mathbf{E}_{17}$: "*left engine out"*, $\mathbf{E}_{19}$: *"go-around decision"*, $\mathbf{E}_{23}$: *"high AoA"*, $\mathbf{E}_{90}$: *"situation end"* }. (2)

A flight event is graphically depicted as a circle or ellipse with the event name and code.

There are several types of flight events, including:

- *independent* and *dependent* (in the latter case a precondition, or "if-event", should be checked first)
- *simple* and *compound* (determined by the number of elementary criteria in the event recognition criterion - see below)
- *"precise"* and *fuzzy* (determined by the type of model variable in the event recognition criterion)
- *momentarily recognizable* and *recognizable with a delay*
- *unique* and *periodical* (*repetitive*), and
- *single* and *serial*.

Note that these class pairs may have non-empty intersections.

The main attribute of a flight event is the *recognition criterion*, which has the following generic format:

$$\mathbf{Cr} = ((v \square R)_1\ l_{12}\ (v \square R)\ l_{23}\ (v \square R)_{34} \dots) \Rightarrow (\mathbf{E} \in \Omega^A(\mathbf{E})). \quad (3)$$

The relationship (3) means that the event **E** becomes recognized, or "active", in the model, i.e. $\mathbf{E} \in \Omega^A(\mathbf{E})$, if the compound logical condition $((v \square R)_1\ l_{12}\ (v \square R)\ l_{23}\ (v \square R)_{34} \dots)$ is true. An example of a compound recognition criterion for the flight event $\mathbf{E}_{11}$: *"runway touchdown"* is as follows: **Cr** = (H LE 0.0 ft) AND (NZ_MAIN GT 0.0 kN) AND (CR_DURATION GE 1.5 s]). It means that this event is recognized in the model, if:

- the altitude (H) is less or equal zero, *and*
- the vertical load on main wheels (NZ_MAIN) is positive, *and*
- the duration of the true condition for the criterion **Cr**, CR_DURATION, is not less than 1.5 s.

Another example of a recognition criterion for a compound event $\mathbf{E}_4$: *"at circuit altitude"* is: **Cr** = (H AE 1200 ft) AND (V_z BEL [-1.0; 1.0] ft/s), or, in mathematical notations, **Cr** = ($H \approx 1200$ ft) & ($V_z \in$ [-1.0; +1.0] ft/s). This criterion defines $\mathbf{E}_4$, when the altitude is approximately equal to 1,200 ft and does not change significantly, i.e. vertical speed remains within 1 ft/s up or down.

In the relationship (3), $(v \square R)_i$ is an "elementary recognition criterion", which is a condition, which specifies a certain (i-th) important aspect or component of the event, i = 1, 2, …. These elementary criteria $(v \square R)_i$ are connected by logical links, thus representing the compound event as a logical composition of its important aspects. In the elementary criterion $(v \square R)_i$, its right part R can be defined by one of the following two methods: $R \equiv$ **a** *Un* or $R \equiv$ [**a; b**] *Un*, where **b** > **a**. For example, $R \equiv$ 300 ft, or $R \equiv$ [2.0; 10.0] degr.

A list of events, which may occur in some flight situation or a group of situations, is called the *flight events calendar*, $\Omega(\mathbf{E})$. The flight event calendar is a discrete framework of a flight situation. During simulation, each event from $\Omega(\mathbf{E})$ can be in one of the following states: "not recognized" (**NR**), "just (or newly) recognized" (**JR**), "frozen" (**FR**), or "recognized (past)" (**R**). Therefore,

$$\Omega(\mathbf{E}) = \Omega^{NR}(\mathbf{E}) \cup \Omega^{JR}(\mathbf{E}) \cup \Omega^{FR}(\mathbf{E}) \cup \Omega^{R}(\mathbf{E}). \quad (4)$$

Note that a subset $\Omega^A(\mathbf{E})$, $\Omega^A(\mathbf{E}) = \Omega^{JR}(\mathbf{E}) \cup \Omega^{FR}(\mathbf{E})$, contains events in a currently "active" state.

All events from $\Omega^A(\mathbf{E})$ are to be specified for modeling. A flight event can be defined by a subset of key attributes. These key attributes are as follows: code, event-precondition ("if-event"), name, list of variables to be memorized when the event occurs, recognition criterion **Cr**, delay (minimum duration of the "true" condition for the criterion **Cr** required before the event is considered as "just recognized"), life cycle (for periodic events only), and some

other. Thus, a generic frame-specification of a flight event can be defined as follows:

$$R[\mathbf{E}] = \{ i, j^{IF}, Nm, (v_1, \ldots, v_n), \mathbf{Cr}, \tau, \Delta, \ldots \}. \quad (5)$$

If an elementary criterion can be used instead of a compound criterion for reliable event recognition, then frame (5) gets a simpler format:

$$R[\mathbf{E}] = \{ i, j^{IF}, Nm, (v_1, \ldots, v_n), (v \square R), \tau, \Delta t, \ldots \}. \quad (6)$$

For example, specifications of flight events $\mathbf{E}_1$, $\mathbf{E}_3$, $\mathbf{E}_4$, and $\mathbf{E}_{15}$, which correspond to (6), follow[8]. $R[\mathbf{E}_1]$ = { 1 0 "situation start" (20 77 32 76) (41 GT 0.0 s) 0.0 0.0 }, $R[E_3]$ = { 3 1 "speed VR achieved" (3 19 14 1) (77 AE 290.0 km/h) 0.0 0.5 }, $R[\mathbf{E}_4]$ = { 4 3 "pitch 10 degr." (77 1 20 3 (14 GE 10.0 degr) 0.0 0.3 }, $R[\mathbf{E}_{15}]$ = { 15 3 "left wheel off r/w" (84 77 14 12) (84 GE 0.0 kN) 0.0 0.5 }.

For example, the last frame $R[\mathbf{E}_{15}]$ defines the flight event $\mathbf{E}_{15}$: *"left wheel off the runway"*, which is a conditional event depending on the occurrence of the "if-event" $\mathbf{E}_3$: *"speed VR achieved"*. The event E_{15} will not be taken for processing in the model until E_3 has occurred. After that, the event $\mathbf{E}_{15}$ is included into the subset of events $\Omega^{NR}(\mathbf{E})$ for recognition. The recognition criterion is simple: $N_{z\ LEFT} \geq 0$, where $N_{z\ LEFT} \equiv v^{84}$. Note that this criterion must remain true during 0.5 second (τ = 0.5 s) before the event is recognized. When $\mathbf{E}_{15}$ is "just recognized", i.e. $\mathbf{E}_{15} \in \Omega^{JR}(\mathbf{E})$, the following variables will be memorized for further analysis: { v^{84}, v^{77}, v^{14}, v^{8} }, or { $N_{z\ LEFT}$, V_{IAS}, ϑ, M }.

Below there are some recommendations for flight event planning in the model:

- events should capture the physics and logic of sudden changes in a flight situation under study
- physical units of variables used in recognition criteria should be carefully checked
- it is important to remember about logical and physical dependencies between flight events
- the "if-event" capability should be utilized in order to make flight scenarios even more robust and generic.

FLIGHT PROCESS – Flight processes are the second major component of the model after flight events. The *flight process*, Π, can be defined as a time-history of one or several flight variables, which characterize a certain continuous aspect of the behavior of the "pilot (automaton) - vehicle - operational environment" system in a given situation. Depending on physical background, flight processes may be divided into four groups:

- aircraft flight dynamics (longitudinal and lateral motion[9])
- pilot's tactical decision making and pilot errors - "piloting task" (**T**), system "state observer" (**O**), "control procedure" (**P**), "pre-defined time-history" (**H**), and some other
- external operational conditions - "wind" (**W**), "rain" (**R**), "runway surface condition" (**Y**), etc.
- onboard system functioning and system failures - "function" (**B**) and "failure" (**F**).

Examples of flight processes are as follows: { $\mathbf{T}_2$: *"keep pitch at about 10°"*, $\mathbf{T}_8$: *"perform right turn at a 25° bank angle and zero sideslip"*, $\mathbf{O}_6$: *"observe bank angle and roll rate"*, $\mathbf{P}_5$: *"flaps - down 0°→30°"*, $\mathbf{P}_2$: *"wheels - up"*, $\mathbf{W}_1$: *"strong wind shear, accident of 03/06/85"*, $\mathbf{R}_2$: *"tropical shower of a trapezoid profile with the maximum intensity of 400 mm/hr"*, $\mathbf{Y}_3$: *"wet runway"*, $\mathbf{B}_1$: *"yaw SCAS operative"*, $\mathbf{F}_1$: *"engine #1 failure"*, $\mathbf{F}_{19}$: *"rudder hardover to +25°"* }.

Flight processes are used for modeling control tactics and various operational factors: manual piloting, functions and malfunctions of onboard systems, and weather conditions. Each process has a specific purpose in the logical structure of a flight situation. Unlike flight events, flight processes are continuous (3-60 seconds long) components of flight logic. A flight process is depicted as an arrow with its tail emerging from one, "source", event and the head pointing to another, "target", event - see the notion of elementary situation below for more detail.

Note. The level of flight situation formalization by means of events and processes may vary depending on the problem. A criterion for adequate mapping of a flight situation into a scenario is how realistically and reliably this scenario reconstructs a particular situation and how sensitive the model is to the input parameters of its events and processes.

A united list of flight processes can be represented as follows:

$$\begin{aligned}\Omega(\Pi) = \; & \Omega(\mathbf{T}) \cup \Omega(\mathbf{O}) \cup \Omega(\mathbf{P}) \cup \Omega(\mathbf{H}) \ldots \\ & \Omega(\mathbf{B}) \cup \Omega(\mathbf{F}) \cup \ldots \\ & \Omega(\mathbf{R}) \cup \Omega(\mathbf{W}) \cup \Omega(\mathbf{Y}) \ldots \end{aligned} \quad (7)$$

Flight processes are modeled according to their state transition automaton, which is similar to the flight event's state transition logic. There are four possible states of a flight process: "not open" (**NO**), "active" (**A**), "frozen" (**F**), and "closed" (**CL**), i.e. $\Omega(\Pi) = \Omega^{NO}(\Pi) \cup \Omega^{A}(\Pi) \cup \Omega^{F}(\Pi) \cup \Omega^{CL}(\Pi)$. "Active" and "frozen" processes constitute a subset of "open" processes, i.e. $\Omega^{O}(\Pi) = \Omega^{A}(\Pi) \cup \Omega^{F}(\Pi)$.

ELEMENTARY SITUATION – In the flight situation model, each process Π normally runs between two events, the "source" event and the "target" event. The *source event,* $\mathbf{E}_*$, opens Π whilst the *target event*, $\mathbf{E}^*$, closes the process during flight. Sometimes, a flight process may not have a target event associated with it. In this case, it means that the process is closed automatically, i.e. when it reaches its objective. There may also be several processes starting or/and finishing at the same event. An

8. in frame-specification examples separating commas are omitted
9. in the current version of the situational model, flight dynamics type processes are embedded into a flight dynamics code of the vehicle

interrelated triplet **s**, **s** = ($\mathbf{E}_*$, Π, $\mathbf{E}^*$), is called the *elementary situation*, like, for example, the triplet ($\mathbf{E}_3$, $\mathbf{T}_2$, $\mathbf{E}_4$) in the examples above.

Due to heterogeneous physical nature of flight processes, input frame-specifications of a flight process depends on its type. Main types of flight processes, which may be useful for accident reconstruction and "neighborhood" analysis, include: "piloting task", "control procedure", "failure", "time-history", "wind", "rain", and some other.

PILOTING TASK – The *piloting task*, **T**, is a flight process used to formalize manual flight control in the model. A piloting task represents some characteristic segment of pilot's goal-oriented control with feedback. Piloting tasks are normally carried out by means of vehicle's primary controls (elevator, ailerons, rudder, power levers, or equivalent devices). It is convenient to formalize a piloting task based on the type of vehicle motion it controls, i.e. longitudinal or lateral. Within each of these two groups, piloting tasks can be further divided according to the type of state variable controlled (linear, angular, etc.). Each piloting task requires measurement (observation) of current system states (a "state observer" type process - ref. below) and tactical flight objectives. The latter represent desired (goal) states of the system. Examples of piloting tasks are as follows: $\mathbf{T}_2$: *"keep the runway's centerline"*, $\mathbf{T}_4$: *"maintain pitch angle at about 10°"*, $\mathbf{T}_{12}$: *"make a coordinated turn at a 20° bank"*, $\mathbf{T}_{19}$: *"follow a bank angle time-history recorded in Flight No. 760 of 07/11/85".*

A list of piloting tasks, $\Omega(\mathbf{T})$, which may be used for modeling manual control tactics in the model, should be specified. The following attributes define a piloting task: code, name, source and target events (i.e. the events, which will initiate and stop the process, respectively), a vector of control variables, which implement the task, a priority level with respect to other processes, time increments for control input application, and some other. Each piloting task can be uniformly represented by the following frame regardless of the type of vehicle motion, tactical objective and feedback type:

$$R[\mathbf{T}_i] = \{ i, j(\mathbf{E}_*), j(\mathbf{E}^*), \xi, Nm, (j(u_1), \ldots, j(u_n)), (\Delta_1, \ldots, \Delta_n), \ldots \} \quad (8)$$

For example, a piloting task $\mathbf{T}_{12}$: *"make a coordinated turn at 20° bank"* is defined by the following frame: R[$\mathbf{T}_{12}$] = { 12 9 28 0 "coordinated turn at 20 degr." (4 10 0 0) (.01 .01 .0 .0) }. According to (8), this task starts at the event $\mathbf{E}_9$ and ends at $\mathbf{E}_{28}$. It is implemented by means of two control variables, aileron and rudder, i.e. (v^4, v^{10}) ≡ (χ, ζ). The frequency[10] for updating the control vector (v^4, v^{10}) in $\mathbf{T}_{12}$ is 100 Hz, i.e. (Δ_1, Δ_2) = (0.01, 0.01).

The "event-process" flight formalization language provides a powerful means for modeling flight control tactics on the cause and effect level. This allows automated and flexible planning and execution of flight simulation experiments by a non-pilot user.

The following hints may be useful in piloting task modeling:

- potential conflicts between tasks and other processes should be avoided though a conflict is detected automatically
- the flight situation model provides an opportunity for flexible virtual testing of various piloting methods, which cannot be checked in actual flight (ref. the example $\mathbf{T}_{19}$ in the task list above).

STATE OBSERVER – The *system state observer* (**O**) is a process of evaluation of the current state of the system and comparing these states with a desired state (tactical objective). Each state observer consists of several elementary observers, i.e.:

$$\mathbf{O}_k = (\mathbf{O}^1{}_k, \ldots, \mathbf{O}^i{}_k, \ldots, \mathbf{O}^{n(O)}{}_k). \quad (9)$$

The goal of system state observation is to detect an error between these two states sufficient to change the piloting task associated with the observer. For example, a piloting task $\mathbf{T}_1$: "*hold roll and sideslip angles at zero*", which is performed by means of ailerons and rudder, requires two state "observers" to monitor the vehicle's bank and sideslip motion, $\mathbf{O}_1$ and $\mathbf{O}_2$. The first observer, $\mathbf{O}_1$ = ($\mathbf{O}^1{}_1$, $\mathbf{O}^2{}_1$, $\mathbf{O}^3{}_1$), or $\mathbf{O}_1$ = ($\mathbf{O}^1{}_1$: *"Roll observation"*, $\mathbf{O}^2{}_1$: *"Roll rate observation"*, $\mathbf{O}^3{}_1$: *"Roll acceleration observation"*), is designed for aileron control. The second observer, $\mathbf{O}_2$ = ($\mathbf{O}^1{}_2$, $\mathbf{O}^2{}_2$, $\mathbf{O}^3{}_2$), or $\mathbf{O}_2$ = ($\mathbf{O}^1{}_2$: *"Sideslip observation"*, $\mathbf{O}^2{}_2$: *"Sideslip rate observation"*, $\mathbf{O}^3{}_2$: *"Sideslip acceleration observation"*), is used in rudder control.

ELEMENTARY OBSERVER – The *elementary observer*, $\mathbf{O}^i{}_k$, is the *i*-th component of the vector $\mathbf{O}_k$ in (9); it specifies the observation process along one state variable. $\mathbf{O}^i{}_k$ is used to measure errors between the current and desired (goal) values of that variable. In the examples above, each of the two observers ($\mathbf{O}_1$ and $\mathbf{O}_2$) consists of three elementary observers, which respectively measure angular attitudes, angular rates and angular accelerations.

The input specification of an elementary observer has the following format[11]:

$$R[\mathbf{O}^i{}_k] = \{ j(\mathbf{T}), j(u), j(x), Nm(x), [\mathbf{G}], \mathbf{k}, x_1, x_2, \Delta_o, \ldots \}, \quad (10)$$

where

- $j(\mathbf{T})$ is the code of a piloting task **T**, which employs the elementary observer $\mathbf{O}^i{}_k$, $\mathbf{T} \in \Omega(\mathbf{T})$; $j(\mathbf{T}) \in \{1, 2, \ldots\} \cup \{0\}$.

10. for a fuzzy or neural network control model a set of task's attributes will be slightly different

11. this specification is applicable to the proportional-integral type of feedback; for neural network or fuzzy control models the specifications will differ

- $j(u)$ is the code of the control variable u, for which $\mathbf{O}^i_k$ supplies state error information, $u \in (u_1, \ldots, u_n) \cup \{0\}$ and $j(u) \in (j(u_1), \ldots, j(u_n)), (j(u_1), \ldots, j(u_n)) \in R[\mathbf{T}]$
- $j(x)$ and $Nm(x)$ are, respectively, the code of the observed state variable x, $x \in V$
- [**G**] is the description of a goal state for the variable x:, which may be specified in two ways: (1) $\mathbf{G} \in \Re$ or (2) $\mathbf{G} \equiv j(\mathbf{H})$, where $j(\mathbf{H})$ is a pointer to a time-history type process **H** (see the definition of the **H** type process below)
- **k** is the gain coefficient used to account for observation errors $\varepsilon(t)$, $\varepsilon(t) = x(t) - \mathbf{G}$, in the feedback model[12], which represents the pilot's perceptual-motor function, $u(t) = f(\mathbf{k}; \varepsilon(t); \ldots)$
- x_1 and x_2 are thresholds of the observation insensitivity zone, $x_1 < x_2$; $x_1, x_2 \in \Re$
- Δ_o is the observation step (seconds), $\Delta_o > 0$.

In the frame (10), if $j(\mathbf{T}) = 0$, then the "elementary observer" $\mathbf{O}^i_k$ will be used by any piloting task **T**, $\mathbf{T} \in \Omega(\mathbf{T})$, which has the control variable u, $j(u) \in R[\mathbf{O}^i_k]$. Also, if $j(u) = 0$ in (10), then $\mathbf{O}^i_k$ will be automatically applied to any control variable u, $u \in (u_1, \ldots, u_n)$, within the task **T**.

Examples of elementary observers constituting $\mathbf{O}_1$ and $\mathbf{O}_2$ are as follows: $R[\mathbf{O}^1_1]$ = { 1 4 12 RollObs [-2.0] 0.5 0.1 0.3 0.005 }, $R[\mathbf{O}^2_1]$ = { 0 4 28 RollRateObs [0.0] 0.5 0.0 0.0 0.005 }, $R[\mathbf{O}^1_2]$ = { 1 10 11 SideslipObs [9.0] -0.5 0.2 0.5 0.05 }, $R[\mathbf{O}^3_2]$ = { 0 10 150 YawAccelObs [0.0] 0.5 0.0 0.0 0.005 }.

For example, the frame $R[\mathbf{O}^1_2]$ describes the elementary observer $\mathbf{O}^1_2$: *"Sideslip observation"*, which is used to monitor the sideslip variable, $x \equiv v^{11} \equiv \beta$, for the piloting task $\mathbf{T}_1$. The goal sideslip angle of 9.0° is pursued by means of rudder, $u \equiv v^{10} \equiv \zeta$. Observation errors $\varepsilon_\beta(t)$, $\varepsilon_\beta(t) = \beta(t) - 9.0°$, are accounted for in the feedback model $\zeta = f(\mathbf{k}; \varepsilon_\beta(t); \ldots)$ with a 20 Hz frequency ($\Delta_o = 0.05$ s) and gain **k** of -0.5 according to the following rule.

The elementary observer $\mathbf{O}^1_2$ is switched "on", i.e. $\varepsilon_\beta(t) = \beta(t)$-**G**, beginning from a time instant t_i, when $|\varepsilon_\beta(t_i)| > x_2$ and $|\varepsilon_\beta(t_i - \Delta_o)| \le x_2$, and until $|\varepsilon_\beta(t_j)| > x_1$ and $|\varepsilon_\beta(t_j - \Delta_o)| \le x_1$, where $x_1 = 0.2°$ and $x_2 = 0.5°$. Otherwise ($\mathbf{O}^1_2$ is switched "off"), $\varepsilon_\beta(t) = 0$.

This condition means that sideslip errors are taken into account in rudder control, if the accuracy of maintaining the required sideslip angle (9°) exceeds the threshold of the first insensitivity zone (±0.5°). If, after this, the error $\varepsilon_\beta(t)$ is reduced to the threshold of the second zone (±0.2°), $\varepsilon_\beta(t)$ is set to zero until the first threshold (±0.5°) has been violated again, etc.

When selecting state observers for piloting tasks, the following recommendations are useful:

- the choice of proper coordinate systems for observed state variables is essential (e.g.: body vs. earth frames)
- the number of elementary observers in an observer is within the range of 2 to 4
- $|\mathbf{k}| \in [0.1; 1.0]$ for majority of conventional aircraft types and flight missions[13].

CONTROL PROCEDURE – The use of secondary controls (flaps, spoilers, etc.), as well as single inputs by primary controls and other discrete type control actions, are described in the model by the process type called the *control procedure*, **P**. For example, $\mathbf{P}_1$: *"wheels - up"*, $\mathbf{P}_2$: *"unstick"*, $\mathbf{P}_3$: *"retract flaps up from 30° to 15°"*, $\mathbf{P}_4$: *"move engine throttles to MCPR"*, etc. Each control procedure can be uniformly described by the following attributes: code, name, vector of control variables, source and target events, objective, and some other parameters. A unified frame-specification of a control procedure has the following format:

$$R[\mathbf{P}_i] = \{ i, j(\mathbf{E}_*), j(\mathbf{E}^*), \xi, Nm, m, (j(u_1), \ldots, j(u_4)), [\mathbf{G}], \tau, \mathbf{k}_{du/dt}, \ldots \}. \quad (11)$$

In the frame (11), the attribute ξ is the priority level of the control procedure $\mathbf{P}_i$ among other control processes. For example, this attribute can be used to resolve conflicts between two open processes (e.g.: $\mathbf{P}_i$ and $\mathbf{P}_j$ from $\Omega^O(\mathbf{P})$), which share the same control variables, i.e.: $(u_1, \ldots, u_4)|\mathbf{P}_i \cap (u_1, \ldots, u_4)|\mathbf{P}_j \neq \varnothing$). Then a process (e.g.: $\mathbf{P}_i$), which has a higher priority level, will be taken for execution in the model, i.e. $(\xi|\mathbf{P}_i > \xi|\mathbf{P}_j) \Rightarrow (\mathbf{P}_i \in \Omega^A(\mathbf{P}) \;\&\; \mathbf{P}_j \in \Omega^F(\mathbf{P}))$.

Another important attribute in (11) is the method of defining the control objective in $\mathbf{P}_i$, m. If m = REL, then the objective value of $(u_1, \ldots, u_4)$ in $\mathbf{P}_i$ is defined as **G** plus the current value of $(u_1, \ldots, u_4)$ when $\mathbf{P}_i$ is just started (relative specification of **G**). If m = ABS, then the objective for $(u_1, \ldots, u_4)$ is set to **G** (absolute specification).

There is also a special type of control procedures for maintaining a required airspeed profile by throttles (m = THR). The coefficient $\mathbf{k}_{du/dt}$ is used to modify the standard rate of change of the variables $(u_1, \ldots, u_4)$: $du/dt = \mathbf{k}_{du/dt} (du/dt)_S$.

Examples of control procedure specifications are: $R[\mathbf{P}_1]$ = { 1 3 0 1 "elevator up by –8 deg." REL (3 0 0 0) -8.0 0.0 0.5 }, $R[\mathbf{P}_2]$ = { 2 13 0 0 "wheels up" ABS (89 0 0 0) 0.0 0.5 1.0 }, $R[\mathbf{P}_3]$ = { 3 18 0 0 "flap 8-->4 deg." ABS (25 0 0 0) 4.0 0.0 1.0 }. For example, $R[\mathbf{P}_2]$ describes the control procedure $\mathbf{P}_2$: *"wheels – up"* implemented by means of the variable v^{89}, $v^{89} \equiv k_{LG}$ (undercarriage control), beginning from the event $\mathbf{E}_{13}$: *"altitude 30 ft"*. The goal is to retract the undercarriage ($k_{LG} = 0$). This procedure starts 0.5 s after the event $\mathbf{E}_{13}$ has been recognized. The rate of undercarriage retraction is standard ($\mathbf{k}_{du/dt} = 1.0$). Note that the target event for $\mathbf{P}_2$ is not specified: the procedure terminates automatically when $k_{LG} = 0$.

ONBOARD SYSTEM FAILURE – The *onboard system failure* (**F**) is a process, which imitates abnormal function-

12. in the proportional-integral or similar feedback type

13. proportional-integral type of feedback

ing of an onboard system. Examples are $\mathbf{F}_2$: *"left engine failure"*, $\mathbf{F}_8$: *"uncommanded deployment of thrust reverser"*, $\mathbf{F}_{27}$: *"rudder hardover to +25°"*. In the model, system failures are specified as artificial control procedures (see the definition above). Thus, the frame-specification of an onboard system failure is similar to R[$\mathbf{P}_i$]:

$$\begin{aligned}\mathrm{R}[\mathbf{F}_i] = \; & \{\, i,\, j(\mathbf{E}_*),\, j(\mathbf{E}^*),\, \xi,\, Nm,\, m,\, (j(u_1),\, \ldots,\, j(u_4)),\\ & [\mathbf{G}],\, \tau,\, \mathbf{k}_{du/dt},\, \ldots\,\}. \end{aligned} \quad (12)$$

The differences between attributes in (11) and (12) are as follows:

- the "control" vector (u_1, …, u_4) represents a variable, which models an onboard subsystem's failure
- the "goal" **G** is the value of (u_1, …, u_4), which corresponds to a degraded performance of the subsystem
- the rate adjustment factor $\mathbf{k}_{du/dt}$ is used to model abrupt ($\mathbf{k}_{du/dt} > 1 \ldots 5$) or slowed down ($0 \geq \mathbf{k}_{du/dt} > 1$) changes of the "control" variable.

Two examples of failures are presented below: R[$\mathbf{F}_5$] = { 5 4 0 0 "engine#4 out" ABS (66 0 0 0) 5.0 0.0 1.9 }, R[$\mathbf{F}_6$] = { 6 27 0 0 "flap jam" ABS (25 0 0 0) 0.0 0.0 7.0 }. For example, R[$\mathbf{F}_5$] describes the failure $\mathbf{F}_5$: *"engine#4 out"*, which occurs at the event $\mathbf{E}_4$: *"pitch 10°"*. This engine failure is modeled as an artificial control procedure performed by means of throttle #4 ($v^{66} \equiv \delta_{THR.4}$). The goal value of v^{66} is 5% (the autorotation level of r.p.m. Note that during this engine failure, the rate of r.p.m. decline is 1.9 times faster than a normal r.p.m. deceleration rate after fuel cutoff.

RAIN-TYPE PROCESS – The *rain-type process* (**R**) is used to model effects of rain on the vehicle aerodynamics and flight dynamics. The following effects can be modeled: increased roughness and waviness of the wing and fuselage surfaces due to rain, as well as dynamic effects of rain drops on the vehicle. Corresponding increments in the lift and drag force and pitching moment coefficients (ΔC_{Lrain}, ΔC_{Drain}, and ΔC_{Mrain}) can be calculated as a function of the rain intensity J [mm/hr]. The underlying mathematical model of rain describes various rain conditions, ranging from "light rain" (with a minimum intensity J = 50 mm/h) up to "heavy tropical shower" ($J \in$ [200; 500] mm/h). An example of the rain type process is $\mathbf{R}_1$: *"heavy rain of a 250 mm/h intensity; visibility 1,500 ft"*.

A list of attributes of a rain type process (rain profile) includes: code, name, codes of source and target events, name of the lookup table, which contains rain data, code of the variable-argument in the lookup table, first and last rows from the table, which contain rain intensity data for modeling, and some other parameters. A generic frame-specification of a rain-type process is as follows:

$$\begin{aligned}\mathrm{R}[\mathbf{R}_i] = \; & \{\, i,\, Nm,\, j(\mathrm{E}_*),\, j(\mathrm{E}*),\, (RT,\, x_{arg},\, r_*,\, r^*,\, n),\, \Delta J,\\ & \ldots\,\}, \end{aligned} \quad (13)$$

where $J(t) = J_T(x_{arg}) + \Delta J$ is the current rain intensity and RT is the code xxx of a lookup table-file gxxx, which contains the rain profile data in the form { $(J_T,\, x_{arg})_1$, …, $(J_T,\, x_{arg})_n$ }.

For example, the rain-type process $\mathbf{R}_1$: *"trapezoid rain profile of 200 mm/h"* can be described by the following frame: R[$\mathbf{R}_1$] = { 1 "trapezoid rain profile 200 mm/h" 1 90 (961 41 36 42 7) –50.0 }. This specification defines a heavy rain process $J = f(t)$ of a trapezoid profile running between events $\mathbf{E}_1$: *"situation start"* and $\mathbf{E}_{90}$: *"situation stop"* with the maximum intensity of 200 mm/h, where $t \equiv v^{41}$. This profile is loaded from the lookup table-file g961, rows 36-42, i.e. { $(J_T,\, t)_1$, …, $(J_T,\, t)_7$ }. The rain intensity profile is modeled as $J(t) = J_T(t) - 50$.

The structure of the lookup table containing rain intensity data is as follows:

$$\begin{aligned}\mathrm{R}[RT] = \; & \{\, \ldots,\\ & (x_{arg},\, I_{arg})_1,\\ & (x_{arg},\, I_{arg})_2,\\ & \ldots,\\ & (x_{arg},\, I_{arg})_n,\\ & \ldots\,\}, \end{aligned} \quad (14)$$

An example of Frame (14) corresponding to $\mathbf{R}_1$ is as follows:

r_*	0.0	0.0
r_*+1	5.0	50.0
r_*+2	10.0	200.0
…	…	…
r_*+n-2	30.0	200.0
r_*+n-1	150.0	0.0

Some useful hints for planning rain type processes are:

- the argument of a rain profile must be a monotonous variable, such as time, distance, altitude (at climb or descent) or other
- the effect of heavy rain on aircraft aerodynamics can be essential (a 3-12% negative effect on the aerodynamic lift and drag coefficients).

WIND-TYPE PROCESS – The *wind-type process* type (**W**) is used to model various non-stochastic wind effects on the vehicle dynamics and flight control. This process type belongs to the group of external operational conditions. The following wind effects can be modeled: gusts, wind shear, "microburst", cross wind, head wind, tail wind, and any other 1-, 2-, and 3-dimensional symmetric and asymmetric (with respect to body frames) wind profiles. Examples of wind processes are $\mathbf{W}_1$: *"strong wind shear of 14 ft/s per 90ft of altitude"* and $\mathbf{W}_5$: *"3D microburst field"*.

Each wind process is described by means of a generic frame-specification similar to (13):

$$\begin{aligned}\mathrm{R}[\mathbf{W}_i] = \; & \{\, i,\, Nm,\, j(\mathrm{E}_*),\, j(\mathrm{E}*),\, (WT,\, x_{arg},\, r_*,\, r^*,\, n),\\ & \ldots.\,\}. \end{aligned} \quad (15)$$

For example, the frame R[$\mathbf{W}_1$], R[$\mathbf{W}_1$] = { 1 "windshear 14ft/s per 90ft H" 1 90 (950 41 1 10 10) }, specifies the wind-type process $\mathbf{W}_1$, which runs in the model between the events $\mathbf{E}_1$ and $\mathbf{E}_{90}$. The wind speed components,

W_{xg}, W_{yg}, W_{zg} = $f(t)$ are read from a lookup table *WT*, *WT* ≡ g950, rows 1-10, where $t \equiv x_{arg} \equiv v^{41}$. The structure of a lookup table *WT* containing wind data is similar to (14):

$$R[RT] = \{ \ldots, (x_{arg}, W_{xg}, W_{zg}, W_{yg})_1, (x_{arg}, W_{xg}, W_{zg}, W_{yg})_2, \ldots, (x_{arg}, W_{xg}, W_{zg}, W_{yg})_n, \ldots \}, \quad (16)$$

Example:

r_*	0.0	0.0	0.0	0.0
r_*+1	10.0	-10.0	-0.5	0.0
r_*+2	15.0	-12.0	-2.0	0.0
…	…	…	…	…
r_*+n-2	39.5	3.0	-1.0	0.0
r_*+n-1	80.0	2.5	0.0	0.0

TIME-HISTORY – The *time history* type process (**H**) is a specially arranged lookup table or a graph of numeric values of a flight variable as a function of time, **H** = $f(t)$. Time-histories are used as a generic means to represent and simulate real, hypothetical or mixed flight data in a flight scenario. Examples are as follows. $\mathbf{H}_1$: *"pitch angle time-history, accident, dd/mm/yy"*, $\mathbf{H}_5$: *"throttles ##1-4 control record from landing of dd/mm/yy"*, $\mathbf{H}_{14}$: *"aileron pulses, en-route mode, 50 s"*, etc. Experience of previous applications demonstrates that practically any flight accident, flight test or certification program can be reconstructed in the model using this type of flight processes. The frame-specification of a time-history has the following format:

$$R[\mathbf{H}_i] = \{ i, (j(x_1), ..., j(x_n)), Nm, \xi, j(\mathbf{E}_*), j(\mathbf{E}^*), (HT, r_*, r^*, n), \Delta t, \Delta x, \ldots \}, \quad (17)$$

where $(j(x_1), ..., j(x_n))$ is the vector of codes of flight variables, which implement the time history, $x = (x_1, ..., x_n)$, and Δt is the time step in the lookup table *HT*, $\Delta t > 0$; $x(t) = x_T(t) + \Delta x$. The lookup table has the following simple format:

$$R[HT] = \{ x_T[1], \ldots, x_T[10], x_T[11], \ldots, x_T[20], \ldots, x_T[n] \}, \quad (18)$$

where $x_T[k]$ is the k-th value from the lookup table *HT*, $k = 1, \ldots, n$.

Two examples of time-history specification frames $R[\mathbf{H}_i]$ are shown below. $R[\mathbf{H}_3]$ = { 3 (235 0 0 0) NX_exp. 0 1 20 (843 143 213 631) 0.5 0.0 }, $R[\mathbf{H}_4]$ = { 4 (214 0 0 0) pitch_exp. 0 1 20 (843 214 284 631) 0.5 0.0 }. The frame $R[\mathbf{H}_3]$ specifies a time-history $\mathbf{H}_3$, $n_x = f(t)$, $n_x \equiv v^{235}$, which is modeled between the events $\mathbf{E}_1$ and $\mathbf{E}_{20}$. Recorded ("experimental") values of the longitudinal acceleration variable (n_x) are stored in the table *HT* ≡ g843, rows 143 - 213, with the time increment Δt = 0.5s (the total number of nodes is 631).

FLIGHT SITUATION SCENARIO – The *flight situation scenario (flight scenario)*, **S**, is a plan for implementing the content of a flight situation and associated piloting tactics in simulation or operation. Flight scenarios are formed of two types of objects - "flight events" and "flight processes", which represent, respectively, discrete and continuous components of flight. A flight scenario may be depicted as a directed graph: **S** = Ω(**E**) ∪ Ω(Π). Its vertices (flight events), Ω(**E**), and directed arcs (processes), Ω(Π), are linked together forming a logical cause-and-effect model of the situation under study. Note that a flight scenario may be viewed as a union of its elementary situations. Flight scenarios capture cause-and-effect and other key relationships between discrete and continuous elements of flight, thus mapping its invariant structure.

Fig. 1 depicts a realistic flight scenario of an accident with a transport airplane, titled $\mathbf{S}_0$: *"Takeoff under microburst conditions"* [4].

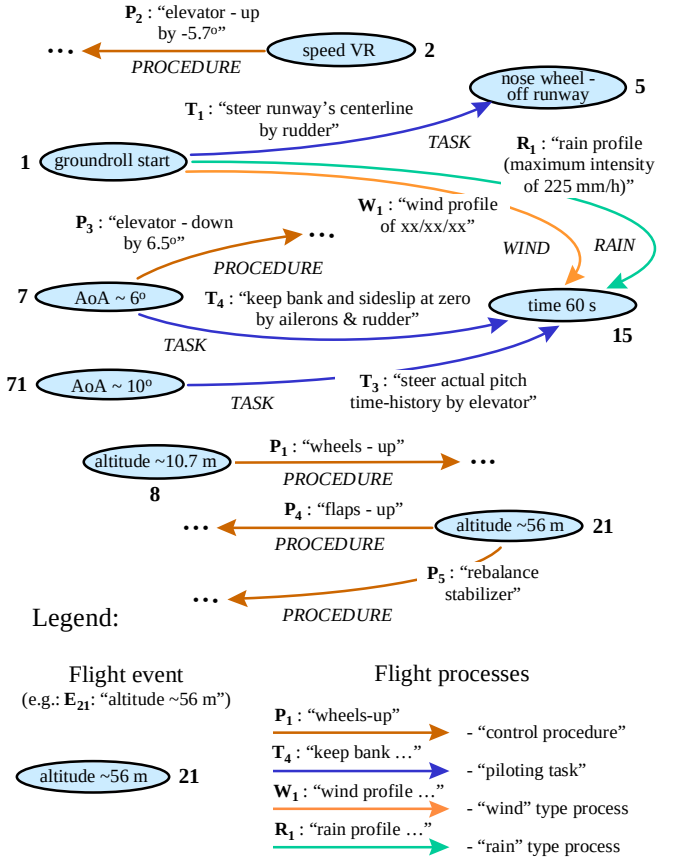

Figure 1: Flight accident scenario $\mathbf{S}_0$: "Takeoff under microburst conditions" [4]

Other examples are as follows: $\mathbf{S}_2$: *"normal takeoff"*, $\mathbf{S}_4$: *"aborted landing"*, $\mathbf{S}_6$: *"level flight at 450 knots and 10 km of altitude"*, $\mathbf{S}_7$: *"coordinated turn at 15° bank"*, $\mathbf{S}_{10}$: *"stall in takeoff configuration"*.

Following are a few recommendations on flight scenario planning:

- the reliability of flight simulation results largely depends on the fidelity of mapping of the actual events and processes into sets $\Omega(\mathbf{E})$ and $\Omega(\Pi)$, i.e. the user's piloting tactics is implemented
- the flight dynamics model (aerodynamic and other input characteristics) must cover the sub-domain of flight modes under study
- it is important to remember about the underlying assumptions and limitations of the physical model of flight.

The frame-specification of a flight scenario is as follows:

$$R[\mathbf{S}_i] = \{ \Omega(v), \Omega(\mathbf{E}), \Omega(\Pi), \dots \}. \quad (19)$$

In programming terms, the set (19) is a table with references (names gxxx) to the data files, which contain frame-specifications of events and processes of the flight situation under study.

AUTONOMOUS FLIGHT SITUATION MODEL – The *autonomous flight situation model* (***M***) is a system of algorithms and data structures, which emulates the behavior of the 'pilot – vehicle – operational environment' system in a complex flight situation. A formal relationship for executing a flight scenario in autonomous simulation can be defined as follows:

$$(\forall \mathbf{S})\ (\mathbf{S} = \Omega(\mathbf{E}) \cup \Omega(\Pi))\ (\exists \mathbf{s})\ (\mathbf{s}` = (\mathbf{E}_i, \mathbf{E}_k, \Pi_j)$$
$$(((\mathbf{E}_i \in \Omega^P(\mathbf{E}) \wedge \mathbf{E}_k \notin \Omega^P(\mathbf{E}) \wedge \Pi_j \notin \Omega^{CL}(\Pi)) \wedge (t \geq$$
$$t[\mathbf{E}_i \in \Omega^P(\mathbf{E})] + \tau)) \Rightarrow \Pi_j \in \Omega^A(\Pi)) \vee ((\mathbf{E}_k \in \Omega^P(\mathbf{E})$$
$$\Rightarrow \Pi_j \in \Omega^{CL}(\Pi)), \quad (20)$$

where τ is a delay in event recognition. The relationship (20), together with the flight event processor and models of flight processes, constitute a generic computational algorithm of ***M.***

MODEL BASED FLIGHT ACCIDENT ANALYSIS PROCESS

In this section, main steps of the model based flight accident analysis process are briefly described. Each step is named after the main objects under processing during this step (ref. object descriptions in the previous section and [1, 4-6]). An example of complex flight accident analysis is described in [4].

1. SOURCE DATA - Collect and analyze source data of the accident from aircraft's flight recorder, radar tapes, cockpit voice recorder, and other accident materials, etc. Select a segment of the flight path for modeling.
2. WORK HYPOTHESES - Define work hypotheses concerning probable causes and key operational factors of the flight accident. Give a concise formulation of the accident analysis problem.
3. INPUT DATA FILES - Process source data on the flight accident. Prepare a file containing flight recorder data in the "time-history" format.
4. FLIGHT VARIABLES - Select a vector of key model variables for quantitative analysis. Define graphic and tabular output forms for these variables.
5. EXPERIMENTATION PLAN - Plan flight simulation experiments with the model. To do this, compile a list of main flight scenarios and their modifications (derivative scenarios) for checking the work hypotheses.
6. MAIN FLIGHT SCENARIO - Develop the main flight scenario(s). Create the derivative scenarios by modifying the following components of the main scenario: "flight events", "piloting tasks", "state observers", "control procedures", "system failures", and "time-histories". Tune the model to the flight mode under examination.
7. OPERATIONAL CONDITIONS (FACTORS) - If required, identify operational conditions (weather, onboard system malfunctions and errors, etc.) of the accident by applying the direct flight simulation and inverse flight dynamics techniques.
8. ACCIDENT RECONSTRUCTION - Reconstruct the flight accident situation using the autonomous flight model. Graphically compare simulated and real data of the flight accident. Evaluate the model's accuracy under given operational conditions [4].
9. "CHAIN REACTION" MECHANISM - Identify main components (events and processes) of a "chain reaction" (cause-and-effect) mechanism of the flight accident. Depict this mechanism in a graphic format [4].
10. SITUATIONAL TREE - Provided that the actual and modeled flight paths are close enough, simulate "neighboring" situations to check the work hypotheses using the derivative scenarios. Construct (draw) a situational tree of the accident's "neighborhood" (accident sub-domain tree). Analyze the vehicle's energy status, transformation and management in all situations from the accident domain. An example of a situational tree, which is constructed based on the flight accident scenario $\mathbf{S}_0$ and derivative scenarios (**Fig. 1**), is shown in **Fig. 2**. (ref. [4] for more detail).
11. FUZZY FLIGHT CONSTRAINTS - Specify all fuzzy flight constraints applicable to this accident case. Check the compliance of the system states with the operational constraints for all the situations from the accident tree. Detect variables violating these constraints. Identify and depict a mechanism of flight constraint violation dynamics. Calculate the complexity and safety metrics of the flight accident sub-domain under examination.
12. FLIGHT SAFETY SPECTRA - Specify flight safety colors, e.g.: "green", "amber", "red", and "black" [4]. Calculate and draw flight safety spectra for each of the flight path-branches from the situational tree.

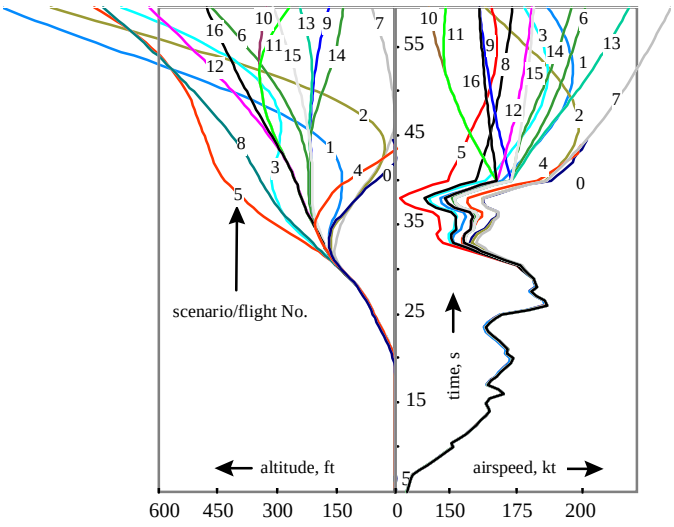

Figure 2: A situational tree of the flight accident and its "neighborhood" [4]

13. SAFE AND UNSAFE FLIGHT DOMAINS - Group, depict and compare simulated flight path-branches from the situational tree along 2 to 5 key flight variables. Conduct qualitative and quantitative analysis of the safety status of the accident domain using results from Steps 7-12: identify subsets of unsafe (catastrophic) flight paths and, if possible, safe (recovery) flight paths. Determine reliable criteria for early recognition (in flight) of "chain reaction" conditions of the given and derivative (neighboring) accident patterns. Identify recognition criteria for safe recovery situations if any.
14. CONCLUSIONS AND RECOMMENDATIONS - Derive conclusions from the conducted simulation experiments. State main assumptions and limitations of the analysis. List key contributing operational conditions (factors) and their interrelationships. Formulate the likely causes of the accident. Summarize findings from Steps 7-12, including energy management, constraints violation dynamics, flight safety spectra, etc. Recommend possible recovery tactics, etc.
15. FINAL REPORT – Prepare a final report with findings and conclusions of the analysis. Propose remedial measures, which would be required to reduce the chances of future occurrences of similar accident patterns. These measures may relate to aircraft piloting tactics, pilot training, situational awareness, vehicle and system design, and other issues. Also propose how to improve the vehicle flight dynamics model (aerodynamics, engine characteristics, etc.) and an onboard flight data recording system.

CONCLUSIONS

Complex flight situations, including accidents and incidents, can be formalized by compact data structures in the form of scenarios. The complexity of the flight accident scenario planning and simulation task does not grow with the complexity of the accident. The autonomous flight situation model can be used as a virtual test article to study complex dynamics of the "pilot (automaton) – vehicle – operational environment" system in a flight accident. The concept of flight scenarios helps to better understand and manage, during simulation experiments with the model, physical and logical interrelationships, which determine the system dynamics in an emergency.

Using this method, key operational factors of flight, as well as extreme or unusual combinations of these factors, which may lead to a "chain reaction" flight accident, can be formalized, quantified and evaluated in detail. A much higher speed and quality of flight accident investigation can be achieved with a simultaneous increase in the amount and quality of knowledge of a complex flight accident domain. Piloting and programming skills are not mandatory for the user.

REFERENCES

1. Burdun, I.Y. and Mavris, D.N, "A Technique for Testing and Evaluation of Aircraft Flight Performance In Early Design Phases" (Paper No. 975541), *Proceedings of the World Aviation Congress (WAC'97), Anaheim, October 1997,* AIAA, 13 pages
2. Parks, E.K., Bach Jr, R.E., and Shin, J.H. "Reconstruction of the 1994 Pittsburgh Airplane Accident Using a Computer Simulation", *Journal of Aircraft*, Vol. 35, No. 5, September-October 1998, AIAA, pp. 665-670
3. McKenna, J. T. "NTSB Weighs Call For Recorder Backups", *Aviation Week & Space Technology*, February 1, 1999, p. 75
4. Burdun, I.Y., "The Intelligent Situational Awareness And Forecasting Environment (The S.A.F.E. Concept): A Case Study" (Paper No. 981223), *Proceedings of the SAE Advances in Flight Safety Conference and Exhibition, April 6-8, 1998, Daytona Beach, FL, USA" (P-321)*, AIAA - SAE Aerospace, 1998, pp. 131-144
5. Burdun, I.Y., DeLaurentis, D.A., Mavris, D.N., "Modeling and Simulation of Airworthiness Requirements for an HSCT Prototype in Early Design" (Paper No. AIAA-98-4936), *Seventh AIAA/USAF/ NASA/ ISSMO Symposium on Multidisciplinary Analysis and Optimization, September 2-4, 1998*, St. Louis, 10 pages
6. Burdun, I.Y. and Parfentyev, O.M., "Fuzzy Situational Tree-Networks for Intelligent Flight Support", *International Journal of Engineering Applications of Artificial Intelligence* (will be published in 1999).

NOMENCLATURE

$\Re$	Set of real numbers
ϑ	Pitch angle
χ	Aileron
ζ	Rudder
η	Elevator
$\varepsilon_\beta(t)$	Sideslip observation error at *t*
$\varepsilon(t)$	Observation error at time instant *t*
$\beta(t)$	Sideslip angle at time instant *t*
ΔC_{Drain}	Drag coefficient increment due to rain
ΔC_{Lrain}	Lift coefficient increment due to rain
ΔC_{Mrain}	Pitching moment coefficient increment due to rain
$(du/dt)_S$	Standard rate of change of control variable *u*
$\delta_{THR.4}$	Engine #4 control lever
[G]	Description of tactical objective
a	Real number, $\mathbf{a} \in \Re$
A	“Active” state of flight event
AE	"Approximately equal" relation type ($\approx$ or $\cong$)
AND, $\wedge$, &	Logical link "and"
AoA	Angle of attack
b	Real number, $\mathbf{b} \in \Re$
B	“Onboard system function" type process
BEL	"Belongs" relation type ($\in$)
Cr	Flight event recognition criterion
Def	Definition
E	Flight event
$\mathbf{E}_*$	Source flight event
$\mathbf{E}^*$	Target flight event
E_0	Euler parameter, $e_0 \in \{ e_0, e_1, e_2, e_3 \}$
EQ	"Equal" relation type (=)
F	“Onboard system failure" type process
f(…)	Function of …
FR	“Frozen” state of flight event
GE	"Greater or equal than" relation type ($\geq$)
GT	"Greater than" relation type (>)
H	“Time-history” type process
i, *j*	Numeric code of a model's object, $i, j \in \{1, 2, \dots\}$
IAS	Indicated airspeed
ij	Pair code, ij $\in$ {12, 23, 34, ...}
J	Rain intensity, mm/h
j(**E**$_*$)	Source event code, $\mathbf{E}_* \in \Omega(\mathbf{E})$
j(**E***)	Target event code, $\mathbf{E}^* \in \Omega(\mathbf{E})$
j(**H**)	Pointer to a “time-history” type process
j(**T**)	Piloting task code in "elementary observer"
J(*t*)	Current rain intensity, mm/h
j(*u*)	Control variable code
$j(u_1), \dots, j(u_n)$	Codes of variables used in a control process, $u_k \in V$ and $k \in \{ 1, ..., 4 \}$
j(*x*)	Code of state variable in elementary observer, $x \in V$
j^{IF}	"If-event" (event precondition)
JR	“Just recognized” state of flight event
J_T	Rain intensity profile data (from *RT*), mm/h
$J_T(x_{arg})$	Rain intensity from the table *RT* at *t*, mm/h
k	Gain
$\mathbf{k}_{du/dt}$	Adjustment factor for the rate of change of control variable *u*, $u` = u`_{nom}\, \mathbf{k}_{du/dt}$
$\mathbf{k}_{LG}$	Undercarriage on/off control variable, $k_{LG} \in [0; 1]$
LE	"Less or equal" relation type ($\leq$)
l_{ij}	Logical link between elementary recognition criteria, $l_{ij} \in \{ OR; AND \}$
LT	"Less than" relation type (<)
M	Mach number
m	Mode (REL or ABS)
MCPR	Maximum continuous power rating
n	Total number of nodes in a rain or wind profile
N(…)	Number of elements in set …
NA	"Approximately not equal" relation type ($\neg\cong$)
NE	"Not equal" relation type ($\neq$)
Nm	Object name or identifier
Nm(*x*)	Name of flight state variable x in "elementary observer"
NR	“Not recognized” state of flight event
$N_{z\ LEFT}$	Vertical reaction on the left undercarriage strut
O, $\mathbf{O}_k$	System state "observer", $\mathbf{O}_k = (\mathbf{O}^1_k, \dots, \mathbf{O}^i_k, \dots, \mathbf{O}^n_k)$
$\mathbf{O}^i_k$	Elementary "observer", $\mathbf{O}^i_k \in (\mathbf{O}^1_k, \dots, \mathbf{O}^n_k)$
OR, $\vee$	Logical link "or"
P	“Control procedure" type process
p_b	Roll rate (body frames)
R	“Recognized” (past) state of flight event
R	Right part of the recognition criterion, $R \equiv \mathbf{a}\ Un$, or $R \equiv [\ \mathbf{a}; \mathbf{b}\ ]\ Un$; $\mathbf{b} > \mathbf{a}$
R	"Rain" type process
r_*	Number of the first row in a lookup table-file
r^*	Number of the last data row in a lookup table-file
R[…]	Input frame-specification of object …
RT	Name of a lookup table-file with rain data, $RT \equiv$ gxxx
S	Elementary flight situation
Sys	Coordinate system (frames of reference)
T	"Piloting task" type process
t, t_j	Time
TAS	True airspeed
u	Flight control vector
u`	Rate of change of variable *u*
$u`_{nom}$	Nominal rate of change for control variable *u*
u^i	Flight control variable, $u^i \in \boldsymbol{u}$
Un	Physical measurement unit
V	Vocabulary of flight variables
$v \square R$	Elementary recognition criterion for a flight event
$v_1, \dots, v_n$	List of variables for memorization when flight event occurs, $v_i \in V$
v^j, *v*	Generic flight variable, *v*, $v^j \in V$
V_{IAS}	Indicated airspeed
v_{max}	Upper graphic limit of variable *v*
v_{min}	Lower graphic limit of variable *v*
V_z	Vertical speed
w	Vector of external flight conditions
W	"Wind" type process
w^j	External condition variable, $w^j \in \boldsymbol{w}$
WT	Name of a lookup table-file containing wind profile's data, $WT \equiv$ gxxx
x	Flight dynamics vector
x_1	Threshold of observation insensitivity zone (observer “off”), $x_1 \in \Re$
x_2	Threshold of observation insensitivity zone (observer “on”), $x_2 \in \Re$
x_{arg}	Code of lookup table argument
x^j	Flight dynamics variable, $x^j \in \boldsymbol{x}$
Y	"Runway surface condition" type process
$\Omega^{NR}(\mathbf{E})$	Subset of "not recognized" events
$\Delta_1, \dots, \Delta_n$	Time steps for control inputs application (in a piloting task)
Δ_i	Time step for control input increments, s
ΔJ	Rain profile shift, mm/h
Δ_o	Time step for system state observation, s
Δt	Event's life cycle (for periodic events $\Delta t > 0$)
$\Omega(\mathbf{B})$	List of onboard system "functions"
$\Omega(\mathbf{E})$	Flight events calendar
$\Omega(\mathbf{F})$	List of onboard system "failures"
$\Omega(\mathbf{H})$	List of “time-histories”
$\Omega(\mathbf{O})$	List of elementary "state observers"
$\Omega(\mathbf{P})$	List of "control procedures"
$\Omega(\mathbf{R})$	List of "rain" type processes

$\Omega(\mathbf{T})$ List of "piloting tasks"

$\Omega(\mathbf{W})$ List of "wind" type processes

$\Omega(\mathbf{Y})$ List of "runway condition" type processes

$\Omega(\Pi)$ United list of flight processes

$\Omega^{A}(\mathbf{E})$ Subset of currently "active" events

$\Omega^{A}(\Pi)$ Subset of "open" processes

$\Omega^{CL}(\Pi)$ Subset of "closed" processes

$\Omega^{F}(\Pi)$ Subset of "frozen" (temporarily inactive) processes

$\Omega^{FR}(\mathbf{E})$ Subset of "frozen" events

$\Omega^{JR}(\mathbf{E})$ Subset of "just (or newly) recognized" events

$\Omega^{NO}(\Pi)$ Subset of "not open" processes

$\Omega^{O}(\Pi)$ Subset of "open" processes

$\Omega^{R}(\mathbf{E})$ Subset of "recognized" (past) events

$\square$ Relation in the criterion for flight event recognition, $\square \in$ { GT, LT, EQ, BEL, GE, LE, NE, AE, NA, … }

τ Delay (in flight event recognition, or control process initialization), $\tau \geq 0$

ξ Process priority level used when $\Pi \in \Omega^{A}(\Pi)$; $\xi \in$ { 0, ... , 99 }

$\Rightarrow$ Implication