MCAI 2 expedition, Pittsburgh, 2009/10/31—11/01

Challenges in control/ command software analysis

Patrick Cousot

pcousot@cs.nyu.edu

http://cs.nyu.edu/~pcousot

November 1, 2009

Motivation

Computer controlled systems

Example of bug report

Australian Government
Australian Transport Safety Bureau

Department
Infrastructure Australia
Bureau of Infrastructure, Transport and Regional Economics

Minister | Help | Feedback | Contact Us | Search | Keyword | GO

14 October 2008 - Qantas Airbus A330 accident Media Conference

Transport

Transport Safety
About the ATSB
Newsroom
Aviation Safety
Marine Safety
Rail Safety
Road Safety
Publications and Investigation Reports
Accident and Incident Notification Forms

Aviation
Roads
Rail
Maritime
Regional
Local Government
The Department

Adjust font size:

MEDIA RELEASE

2008/43

Qantas Airbus A330 accident Media Conference

14 October 2008

The Australian Transport Safety Bureau's investigation into the accident involving an Airbus A330-300 aircraft operating as Qantas flight 72 on a flight from Singapore to Perth on 7 October 2008 is progressing well. The ATSB has scheduled the media conference this evening to coincide with the release of an Operators Information Telex/Flight Operations Telex, which is being sent by Airbus to operators of all Airbus aircraft. The aim of that telex is to:

- update operators on the factors identified to date that led to the accident involving QF72.
- provide operational recommendations to mitigate risk in the event of a reoccurrence of the situation which occurred on QF72.

To assist in understanding the following information, I would just like to refer you quickly to the diagrams projected on the screen specifically, the term angle of attack which refers to the difference in angle between the aircraft and its control surfaces, and the air stream as the aircraft moves through the air.

The next diagram is a simple representation of the aircraft and the components relevant to this explanation, which include the angle of attack sensors located on the outside of the aircraft; the Air Data Inertial Reference Units (ADIRUs), of which there are three, located in the avionics compartment inside the aircraft; the Flight Control Primary Computers of which there are also three located in the avionics compartment, and the elevators, located on the aircrafts horizontal stabiliser. In the context of this occurrence, the angle of attack sensors send raw data to the ADIRUs, which provide processed angle of attack information to the Flight Control Primary Computers, which in turn command the elevator position.

Returning to the circumstances of the 7 October flight, preliminary analysis of the Flight Data Recorder data, Post Flight Report data and Built-In Test Equipment data has enabled the investigation to establish a preliminary sequence of events this information is also contained in the Airbus telex.

The aircraft was flying at FL 370 or 37, 000 feet with Autopilot and Auto-thrust system engaged, when an Inertial Reference System fault occurred within the Number-1 Air Data Inertial Reference Unit (ADIRU 1), which resulted in the Autopilot automatically disconnecting. From this moment, the crew flew the aircraft manually to the end of the flight, except for a short duration of a few seconds, when the Autopilot was reengaged. However, it is important to note that in fly by wire aircraft such as the Airbus, even when being flown with the Autopilot off, in normal operation, the Flight Control computers will still command control surfaces to protect the aircraft from unsafe conditions such as a stall.

The faulty Air Data Inertial Reference Unit continued to feed erroneous and spike values for various aircraft parameters to the aircrafts Flight Control Primary Computers which led to several consequences including:

- false stall and overspeed warnings
- loss of attitude information on the Captain's Primary Flight Display
- several Electronic Centralised Aircraft Monitoring system warnings.

About 2 minutes after the initial fault, ADIRU 1 generated very high, random and incorrect values for the aircrafts angle of attack.

These very high, random and incorrect values of the angle attack led to:

- the flight control computers commanding a nose-down aircraft movement, which resulted in the aircraft pitching down to a maximum of about 8.5 degrees,
- the triggering of a Flight Control Primary Computer pitch fault.

The crew's timely response led to the recovery of the aircraft trajectory within seconds. During the recovery the maximum altitude loss was 650 ft.

The Digital Flight Data Recorder data show that ADIRU 1 continued to generate random spikes and a second nose-down aircraft movement was encountered later on, but with less significant values in terms of aircraft's trajectory.

At this stage of the investigation, the analysis of available data indicates that the ADIRU 1 abnormal behaviour is

likely as the origin of the event.

The aircraft contains very sophisticated and highly reliable systems. As far as we can understand, this appears to be the first event and Airbus has advised that it is not aware of any similar event over the many years of operation of the Airbus.

Airbus has this evening, Australian time, issued an Operators Information Telex reflecting the above information. The telex also foreshadows the issue of Operational Engineering Bulletins and provides information relating to operational recommendations to operators of A330 and A340 aircraft fitted with the type of ADIRU fitted to the accident aircraft. The Operational Engineering Bulletins will provide guidance to crews in the event of a reoccurrence. That includes guidance and checklists for crew response in the event of an Inertial Reference System failure.

Meanwhile, the ATSB's investigation is ongoing and will include:

- Download of data from the aircraft's three ADIRUs and detailed examination and analysis of that data. Arrangements are currently being made for the units to be sent to the component manufacturer's facilities in the US as soon as possible and for ATSB investigators to attend and help with that testing, along with representatives of the US National Transportation Safety Board, The French Bureau d'Enquêtes et d'Analyses (BEA) and Airbus.
- In addition, investigators have been conducting a detailed review of the aircraft's maintenance history, including its compliance with mandatory Airworthiness Directives. Airworthiness Directives, although initial indications are that the accident trail ISI relevant airworthiness requirements.
- Work is also ongoing to progress interviews, which will include with injured passengers to understand what they experienced during the flight, and with the flight crew and cabin crew.

There has been close and frequent communication between the ATSB, Qantas, Airbus, the BEA, and CASA. That close communication will continue as the investigation progresses to ensure that any additional safety action can be instigated as soon as possible should critical safety factors be identified. The ATSB expects to publish a Preliminary Report into the accident within 12 months of the occurrence.

Media Contact: David Hoyer 1800 020 616

Related Documents: | Audio file of media conference, 14 October 2008 (18 MB) |

Print

Last Updated: 14 October, 2008

Privacy | Copyright | Disclaimer | Linking to the ATSB website | Sitemap

"The Australian Transport Safety Bureau (ATSB) found that the main probable cause of this incident was a *latent software error* which allowed the ADIRU to use data from a failed accelerometer"

http://www.atsb.gov.au/newsroom/2008/release/2008_43.aspx,
http://en.wikipedia.org/wiki/Qantas_Flight_72
ADIRU = Air Data Inertial Reference Unit (provides air speed, altitude & position)

- The initial effects of the fault were:
  - false stall and overspeed warnings
  - loss of attitude information on the Captain's primary flight display
  - several Electronic Centralised Aircraft Monitoring (ECAM) system warnings
- About two minutes later, ADIRU #1, which was providing data to the captain's primary flight display, provided very high (and false) indications for the aircraft's angle of attack, leading to:
  - the flight control computers commanding a nose-down aircraft movement, which resulted in the aircraft pitching down to a maximum of about 8.5 degrees,
  - the triggering of a Flight Control Primary Computer pitch fault.
- On 15 January 2009 the EASA issued an Emergency Airworthiness Directive to address the above A330 and A340 Northrop-Grumman ADIRU problem of incorrectly responding to a defective inertial reference. [5]

© P. Cousot

- A memo leaked from Airbus on the Flight 447 (from Rio de Janeiro, Brazil, to Paris, that crashed into the Atlantic Ocean on 1 June 2009) suggests that there was no evidence that Honeywell manufactured ADIRU malfunction was similar to the failure of the Northrop Grumman manufactured ADIRU in Qantas flight incidents

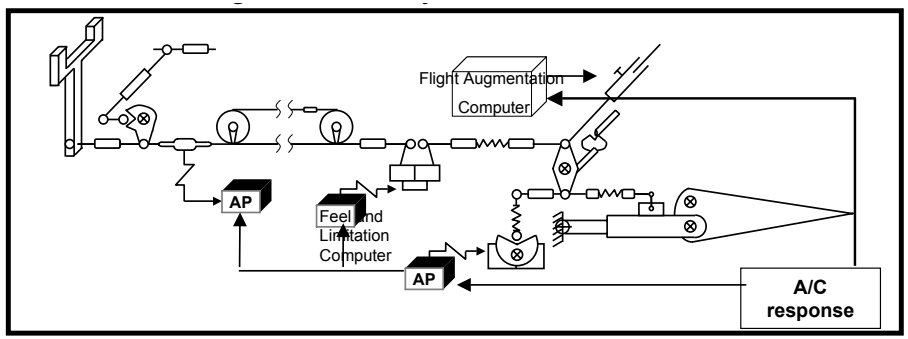

An Air Data Inertial Reference Unit (ADIRU) is a key component of the integrated Air Data Inertial Reference System (ADIRS), that supplies air data (airspeed, angle of attack and altitude) and inertial reference (position and attitude) information to the pilots' Electronic Flight Instrument System displays as well as other systems on the aircraft such as the engines, autopilot, flight control and landing gear systems

© P. Cousot

Brief overview of one approach to the design of an aircraft digital fly by wire control system

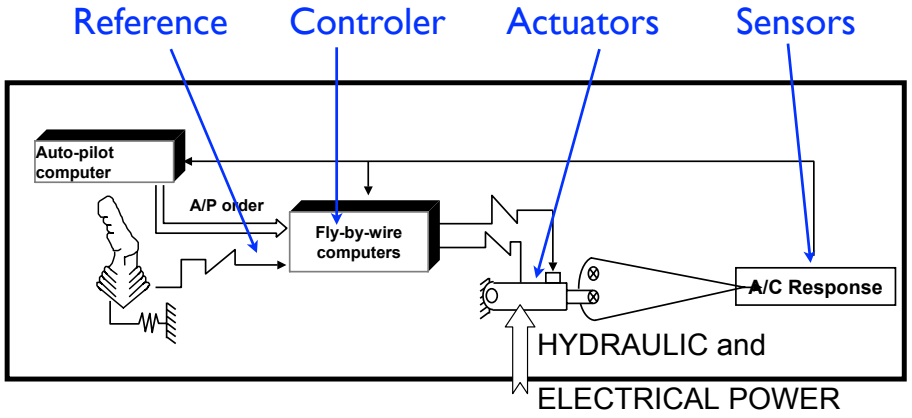
© P. Cousot

Mechanical flight-control systems



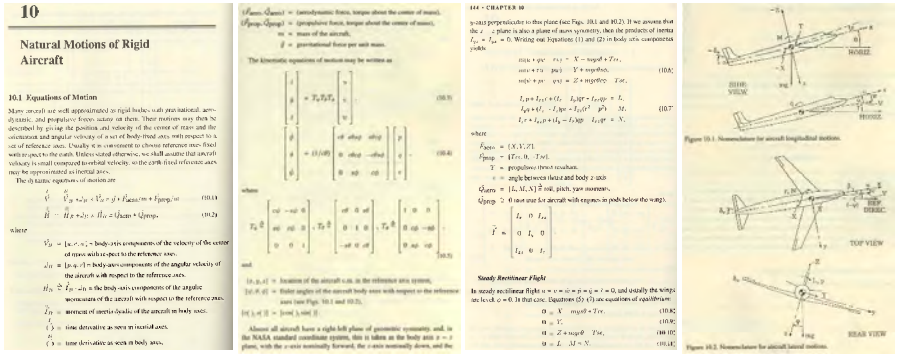
© P. Cousot

What is fly-by-wire?



© P. Cousot

Mathematical model of an airplane and its flight dynamics



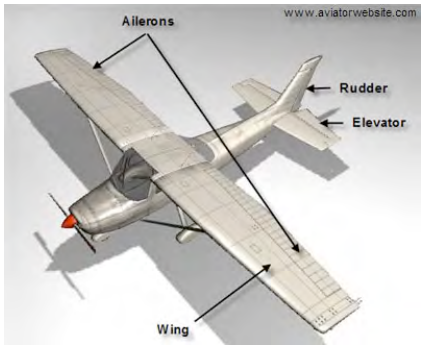
10

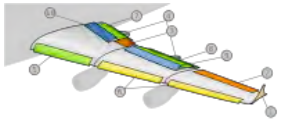
Natural Motions of Rigid Aircraft

10.1 Equations of Motion

© P. Cousot

Control surfaces





Control surfaces
1. Winglet
2. Low-Speed Aileron
3. High-Speed Aileron
4. Flap track fairing
5. Krüger flaps
6. Slats
7. Three slotted inner flaps
8. Three slotted outer flaps
9. Spoilers
10. Spoilers-Air brakes

© P. Cousot

Sensors and Actuators

- Position sensors:

- Force measurement sensors:

- Presure sensors:

- Speed meters (Pitot tubes):

- Pitch, roll a yaw meters/gyroscopes:

- etc

- Flap actuators:

- Trim actuators:

- Vibration control actuators

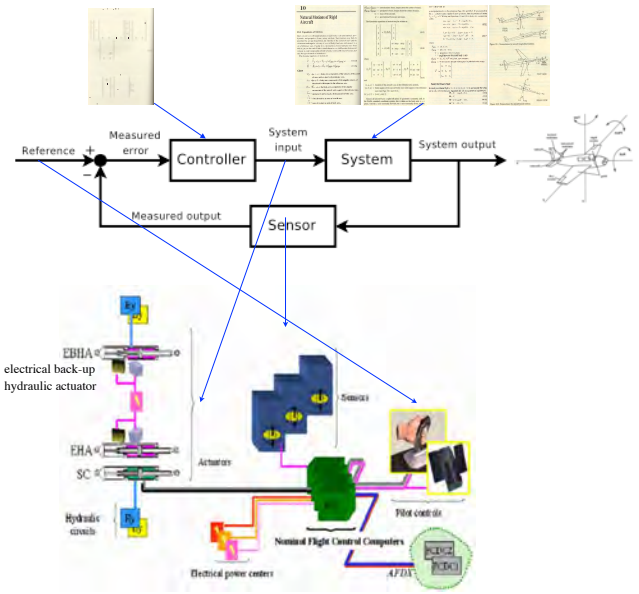
- Engine thurst vector control:

- Engine fuel metering:

- etc

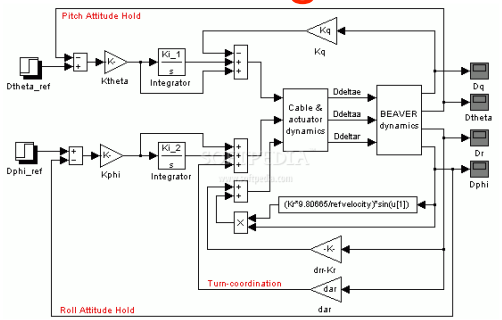
© P. Cousot

Design of the control model (flight control laws)



© P. Cousot

Simulation of the controlled system e.g. in Simulink™



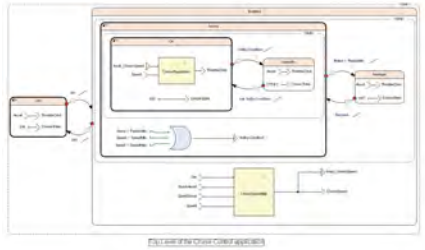
```
% Open a FlightGearAnimation object.
fg=fganimation;

% Convert Latitude, Longitude and Euler angles from
% degrees to radians using the "convang" function and
% set FlightGearAnimation object properties for timeseries.
fg.TimeseriesSource = [myflightdata(:,1) ...
convang(myflightdata(:,[3 2]),'deg','rad') ...
myflightdata(:,4) ...
convang(myflightdata(:,5:7),'deg','rad')];

% Play Back the Flight Trajectory
fg.play;
```

The MathWorks' Aerospace Toolbox 2.4

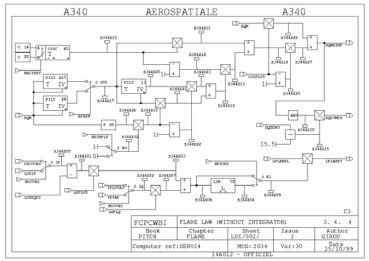
© P. Cousot

Formal specification of the control e.g. in Scade™ / SAO™

Scade™:



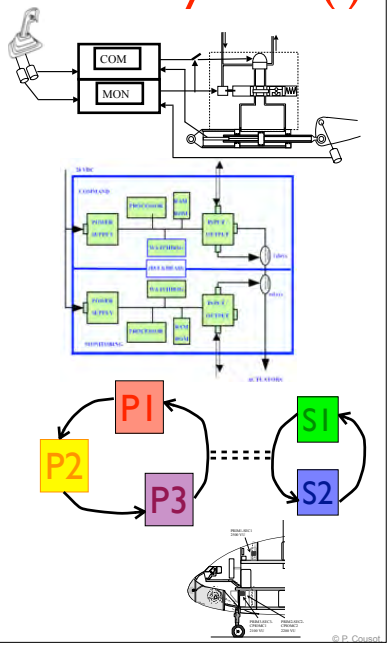
SAO™:



© P. Cousot

Design of a fault-tolerant control system (I)

- Separate flight computers with different software and inputs
- The results of these command and monitoring units are checked by a comparator
- In case of desagreement control switches to another pair of computers
- In total 10 primary and secondary computers



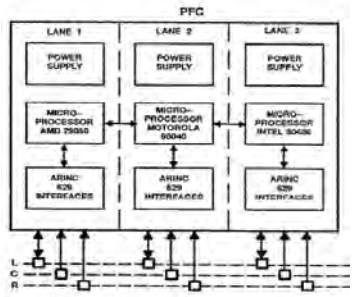
Traverse P., Lacaze I., Souyris J., « Airbus Fly-by-Wire: A Total Approach to Dependability », Proceedings 18th IFIP World Computer Congress, Building the Information Society, Toulouse, France, 22-27 août 2004, pp. 191-212.

(I) Airbus

© P. Cousot

Design of a fault-tolerant control system (II)

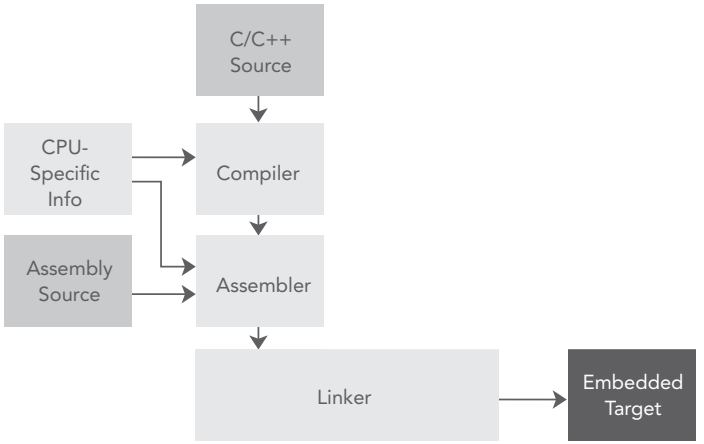
- Three different primary flight computers
- The computers have the same input signals and are all active
- The outputs are connected to a voter that compares signals
- The correct signal is chosen by a majority vote



Yeh Y.C., « Triple-Triple Redundant 777 Primary Flight Computers », Proceedings IEEE Aerospace Applications Conference, Aspen, CO, USA, 3-10 février 1996, pp. 293-307.

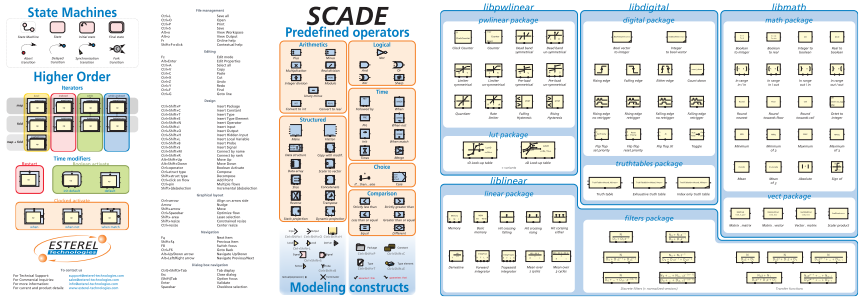
(II) Boeing

Automatic compilation and linkage in machine code



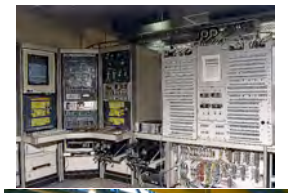
Generation of C code

- Automatic C code generation
- The primitives are programmed by hand



Simulation and testing

- From classical software testing and inspection techniques (e.g. Rational Suite DevelopmentStudio[1])
- To full-scale simulation benches for system integration:



- To the iron bird:

- To flight tests:

[1] www-01.ibm.com/software/awdtools/suite/dstudio/unix/

Successes for formal methods

Examples of static analyzers

- AiT WCET Analyzers (www.absint.de/ait/): compute tight bounds for the worst-case execution time (WCET) of tasks in synchronous real-time systems.
- Astrée (www.astree.ens.fr, www.absint.de/astree/): proof of absence of runtime error in the primary flight control software of the Airbus A340/600 & A380 fly-by-wire system.
- Fluctuat (www-list.cea.fr/labos/gb/LSL/fluctuat/flopsoft.html): contribution of individual program operations to the global error on float variables, used on DSPs (.
- StackAnalyzer (www.absint.de/stackanalyzer/): worst-case stack usage of the tasks in all applications.

Examples of model-checkers

- SCADE Design Verifier (www.esterel-technologies.com/products/scade-suite/design-verifier): check the logic of flight-control primitives

Examples of provers

- Caveat (http://www-list.cea.fr/labos/gb/LSL/caveat/index.html): Hoare logic prover with user-provided invariants, used to prove the Autotest primitive functions

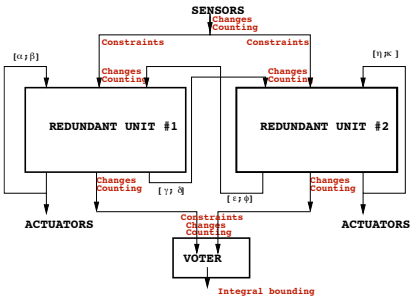
Short term challenges for formal methods

From sequential/synchronous to parallel/ distributed

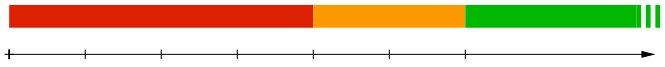
Flight warning system

Test is sometimes impossible...

Static analysis of communicating imperfectly clocked redundant units



Specification : no alarm raised with a normal input



input stability $< \Delta$:	Between $\frac{2}{3} \times \Delta$	input stability $> \Delta$: the analyzer
counter-example	and Δ : ?	proves the specification

Long-term challenges for formal methods

Future challenges

A software bug discovered during flight tests is extremely costly, but easy to correct during the design phase

- *Present day:* process validation + a posteriori software validation
- *Future:* early definition and validation of [software] systems + final product validation

Models and specifications

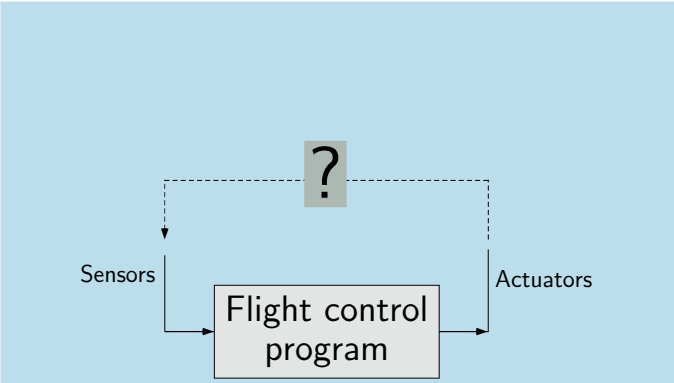
- Absence of formal specifications: model-based development in Simulink™, Scade™, etc. means that the model is the only available formal specification ⇒ no formal specification of the model itself

- Implicit specifications: can be embedded in verification tools (and documented e.g. in simple data files), examples:
  - Integrity: no computer can send wrong information to actuators
  - Availability: in a fault-tolerant system, no component failure can result in the unavailability of a service
  - Electromagnetic radiations: memory glitches cannot make programs to fail
  - etc

- Explicit formal specifications ultimately needed: in which form?

Environment specifications

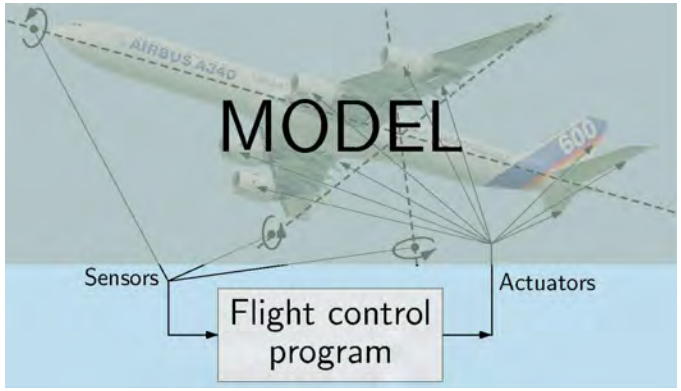
- Currently blind verification:



- Program abstraction : precise
- Environment abstraction : imprecise (bounds)

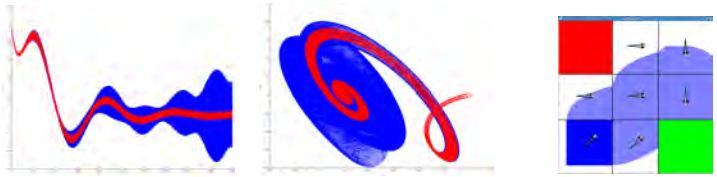
Environment specifications

- Precise verification:



- Program abstraction : precise
- Environment abstraction : precise

Hybrid models of discrete control programs and continuous physical systems



- Recent improvements on bounded time analysis of continuous and hybrid systems (1)
- Not so much in duration (``unbounded time")

⇒ should scale from seconds to minutes and more.

(1) Colas Le Guernic. Reachability Analysis of Hybrid Systems with Linear Continuous Dynamics. PhD thesis, University Joseph Fourier of Grenoble, October 29, 2009.

Conclusion

- Formal methods should be used:
  - All along the whole design chain
  - On functional properties (specification?)
  - On a long range of time (s $\rightarrow$ mn $\rightarrow$ h: scalability)
  - For parallel/distributed systems (semantics?)
  - For end-users (non-intrusion)

© P. Cousot

Thank you for your attention

© P. Cousot

Conclusion

- Formal methods should be used:
  - All along the whole design chain
  - On functional properties (specification?)
  - On a long range of time (s $\rightarrow$ mn $\rightarrow$ h: scalability)
  - For parallel/distributed systems (semantics?)
  - For end-users (non-intrusive methods)

    - Robust software has simple abstractions
    - Long proofs imply bugs

Paraphrasing John Doyle

© P. Cousot