Hybrid Systems: The Early Years

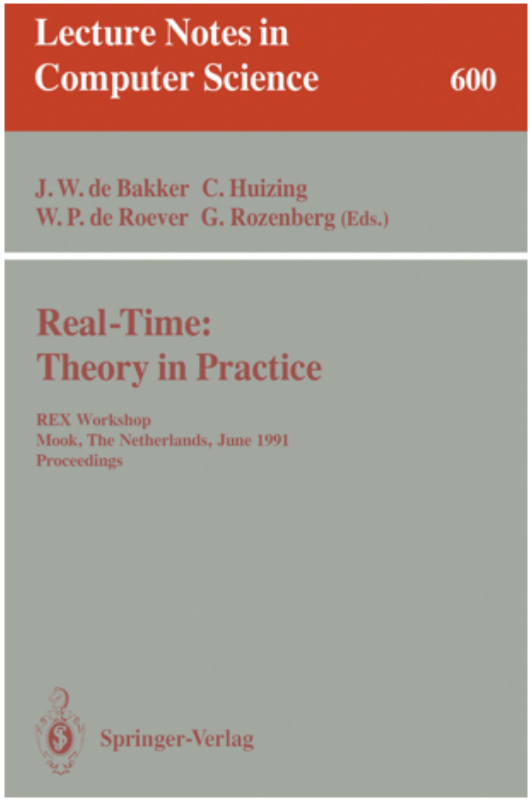
A Tribute to Oded Maler

Rajeev Alur
University of Pennsylvania

Real-Time: Theory in Practice, REX Workshop Mook, The Netherlands

Meeting: June 1991; Proceedings: Summer 1992

J. W. de Bakker, W.P. de Roever, G. Rozenberg



Lecture Notes in Computer Science 600

J. W. de Bakker C. Huizing
W. P. de Roever G. Rozenberg (Eds.)

Real-Time: Theory in Practice

REX Workshop
Mook, The Netherlands, June 1991
Proceedings

Springer-Verlag

A Melody of Formalisms for Timed Systems

- Timed automata (Alur, Dill)
- Timed temporal logics (Alur, Henzinger)
- Real-time logic RTL (Jahanian, Mok)
- Timed CSP (Reed, Roscoe)
- Timed transition systems (Henzinger, Manna, Pnueli)
- Timed I/O automata (Lynch, Vaandrager)
- Communicating shared resources (Gerber, Lee)
- TLA (Abadi, Lamport)

and many more

Lively discussion:

How to add continuous dynamics to formal models ?

The Grand Challenge for Hybrid Systems !!

Initial position y
Speed u
Start time Δ

Initial position x
Speed v
Start time 0

Does the cat catch the mouse ?

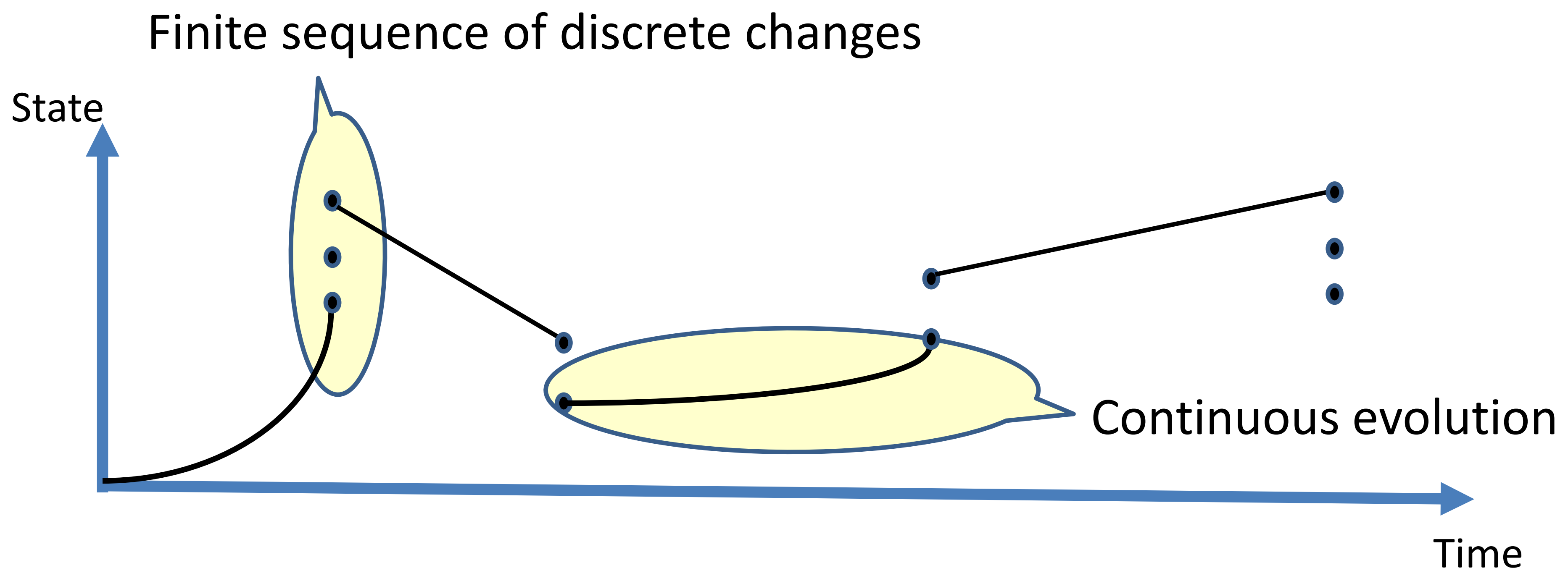
From Timed to Hybrid Systems
Maler, Manna, and Pnueli

Real-Time: Theory in Practice
LNCS 600, pp. 447—484, 1992

[illegible]

We propose a framework for the formal specification and verification of *timed* and *hybrid* systems. For timed systems we propose a specification language that refers to time only through *age* functions which measure the length of the most recent time interval in which a given formula has been continuously true. We then consider hybrid systems, which are systems consisting of a non-trivial mixture of discrete and continuous components, such as a digital controller that controls a continuous environment. The proposed framework extends the temporal logic approach which has proven useful for the formal analysis of discrete systems such as reactive programs. The new framework consists of a semantic model for hybrid time, the notion of *phase transition systems*, which extends the formalism of discrete transition systems, an extended version of Statecharts for the specification of hybrid behaviors, and an extended version of temporal logic that enables reasoning about continuous change.

Hybrid Traces: Modeling Executions of Hybrid Systems



Time moment = (real time t, discrete time n)

Lexicographic order over time moments

Phase Transition Systems

- State variables V partitioned into V_c and V_d
- Assertion I specifying initial states
- Set T of (discrete) transitions, where each transition τ is a function from states to sets of states (transition τ is enabled in state s if $\tau(s)$ is non-empty)
- Lower bound l_τ for each transition τ (specifies how long a transition must be enabled before it is taken)
- Upper bound u_τ for each transition τ (specifies a bound on how long a transition can be enabled without being taken)
- Set A of (continuous) activities, where each activity is a conditional differential equation: guard(V_d) $\rightarrow$ { dy/dt = exp | y in V_d }

Formal semantics of a phase transition system = Set of hybrid traces

Modeling the Grand Challenge

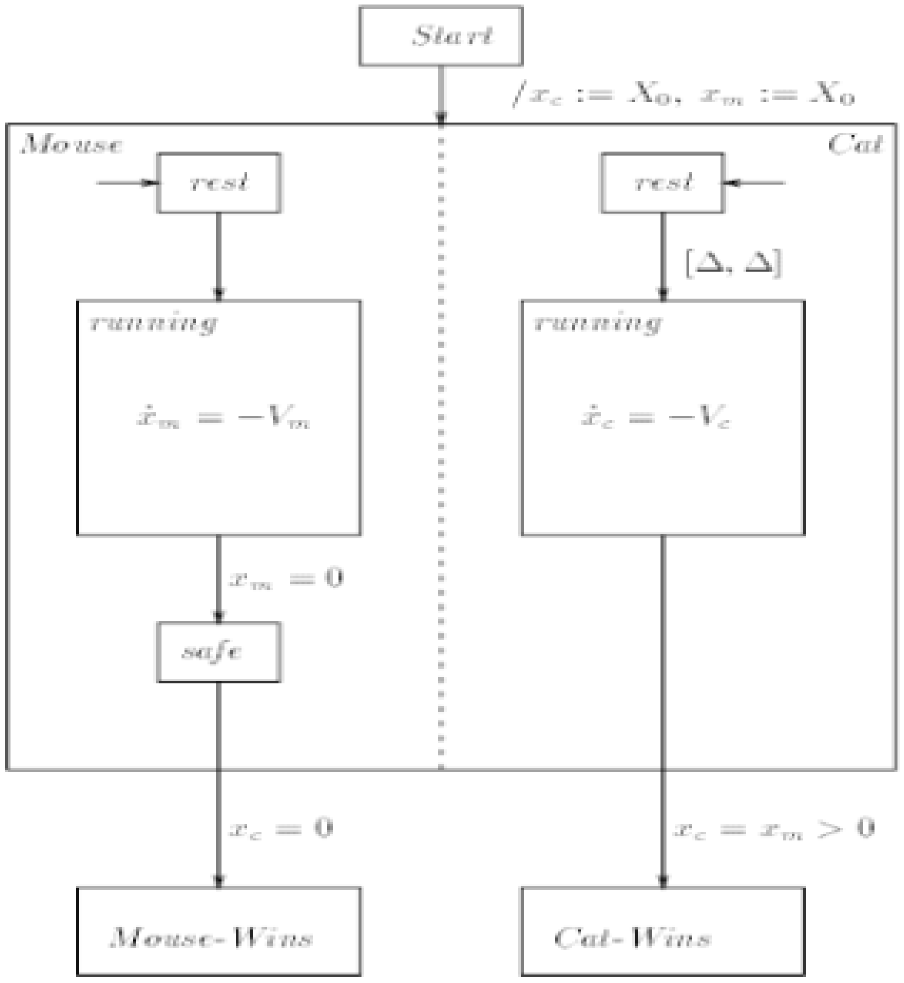
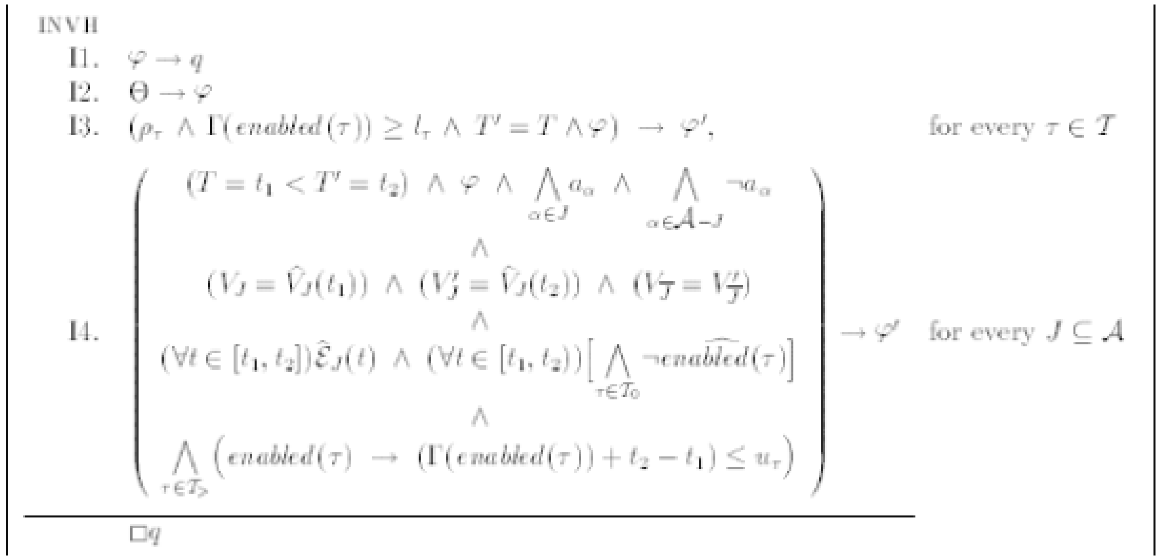


Figure 3: Specification of Cat and Mouse.

Proof Rules for Verification



INVH

I1. $\varphi \rightarrow q$

I2. $\Theta \rightarrow \varphi$

I3. $(\rho_\tau \wedge \Gamma(enabled(\tau)) \geq l_\tau \wedge T' = T \wedge \varphi) \rightarrow \varphi'$, for every $\tau \in \mathcal{T}$

I4. $$\left(\begin{array}{c} (T = t_1 < T' = t_2) \wedge \varphi \wedge \bigwedge_{\alpha \in J} a_\alpha \wedge \bigwedge_{\alpha \in \mathcal{A} - J} \neg a_\alpha \\ \wedge \\ (V_J = \widehat{V}_J(t_1)) \wedge (V'_J = \widehat{V}_J(t_2)) \wedge (V_{\overline{J}} = V'_{\overline{J}}) \\ \wedge \\ (\forall t \in [t_1, t_2]) \widehat{\mathcal{E}}_J(t) \wedge (\forall t \in [t_1, t_2)) \Big[\bigwedge_{\tau \in \mathcal{T}_0} \neg \widehat{enabled}(\tau) \Big] \\ \wedge \\ \bigwedge_{\tau \in \mathcal{T}_>} \Big(enabled(\tau) \rightarrow (\Gamma(enabled(\tau)) + t_2 - t_1) \leq u_\tau \Big) \end{array} \right) \rightarrow \varphi'$$ for every $J \subseteq \mathcal{A}$

$\Box q$

Generalization of inductive invariants for fair transition systems

Relies on explicit solution of differential equations

An Invariant for the Cat and Mouse Specification

A more interesting invariant concerns the Cat and Mouse example. Here we may want to determine conditions under which the cat will never catch the mouse. A simple calculation leads to the requirement

$$\frac{X_0}{V_m} < \Delta + \frac{X_0}{V_c} \tag{1}$$

The condition states that the time it takes the mouse to reach the wall is smaller than the time it takes the cat to reach the wall. Since both run at constant speed, this guarantees that they will not meet, except at the wall, the mouse arriving there first.

Assume that condition (1) is given and that $\Delta > 0$). We then would like to establish the invariant

$$\mathit{Cat.running} \wedge (x_c = x_m) \Rightarrow x_m = 0. \tag{2}$$

To prove this invariant we use an auxiliary assertion φ given by a conjunction of the following implications

$$\begin{array}{rcll}
\mathit{Mouse.rest} & \rightarrow & x_m = X_0 & \\
\mathit{Mouse.running} & \rightarrow & x_m = X_0 - V_m \cdot \Gamma(\mathit{Mouse.running}) \geq 0 & (3)\\
\mathit{Mouse.safe} & \rightarrow & x_m = 0 & (4)\\
\mathit{Cat.rest} & \rightarrow & x_c = X_0 & \\
\mathit{Cat.running} & \rightarrow & x_c = X_0 - V_c \cdot \Gamma(\mathit{Cat.running}) \geq 0 & (5)\\
\mathit{Cat.rest} & \rightarrow & \mathit{Mouse.rest} \vee \mathit{Mouse.running} \vee \mathit{Mouse.safe} & \\
\mathit{Cat.running} & \rightarrow & \mathit{Mouse.running} \vee \mathit{Mouse.safe} & (6)\\
\mathit{Mouse.rest} \wedge \mathit{Cat.rest} & \rightarrow & \Gamma(\mathit{Mouse.rest}) = \Gamma(\mathit{Cat.rest}) = 0 & \\
\mathit{Mouse.running} \wedge \mathit{Cat.rest} & \rightarrow & \Gamma(\mathit{Mouse.running}) = \Gamma(\mathit{Cat.rest}) \leq \Delta & \\
\mathit{Mouse.running} \wedge \mathit{Cat.running} & \rightarrow & \Gamma(\mathit{Mouse.running}) = \Gamma(\mathit{Cat.running}) + \Delta & (7)
\end{array}$$

Assuming that φ has been shown to satisfy premises I2–I4, we will show that it implies $q : (\mathit{Cat.running} \wedge (x_c = x_m)) \rightarrow x_m = 0$. By implication (6), when the cat is running the mouse is either in state *running* or in state *safe*. If it is in state *safe* then, by implication (4), $x_m = 0$. If it is in state *running*, then the assumption $x_c = x_m$ with implications (3), (5), and (7), leads to $V_c \cdot \Gamma(\mathit{Cat.running}) = V_m \cdot (\Gamma(\mathit{Cat.running}) + \Delta)$. From this, we can conclude $V_c > V_m$ and

$$\Gamma(\mathit{Cat.running}) = \frac{V_m \cdot \Delta}{V_c - V_m}.$$

On the other hand, from implication (5) we obtain $X_0 - V_c \cdot \Gamma(\mathit{Cat.running}) \geq 0$ which can be written as

$$\Gamma(\mathit{Cat.running}) \leq \frac{X_0}{V_c}.$$

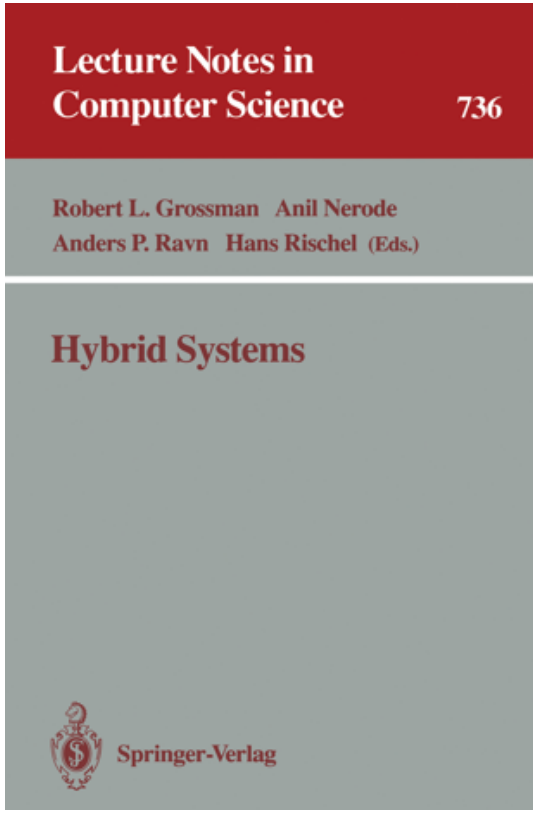
Comparing the equality and inequality involving $\Gamma(\mathit{Cat.running})$, we obtain

$$\frac{V_m \cdot \Delta}{V_c - V_m} \leq \frac{X_0}{V_c},$$

Hybrid Systems Workshop
Technical Univ of Denmark, Lyngby

Meeting: October 1992; Proceedings: Summer 1993

R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel



Lecture Notes in Computer Science 736

Robert L. Grossman Anil Nerode Anders P. Ravn Hans Rischel (Eds.)

Hybrid Systems

Springer-Verlag

Hybrid Automata: An algorithmic approach to the specification and verification of hybrid systems

Alur, Courcoubetis, Henzinger and Ho

Hybrid Systems
LNCS 736, pp. 209--229, 1992

We introduce the framework of *hybrid automata* as a model and specification language for hybrid systems. Hybrid automata can be viewed as a generalization of timed automata, in which the behavior of variables is governed in each state by a set of differential equations. We show that many of the examples considered in the workshop can be defined by hybrid automata. While the reachability problem is undecidable even for very restricted classes of hybrid automata, we present two semidecision procedures for verifying safety properties of *piecewise-linear* hybrid automata, in which all variables change at constant rates. The two procedures are based, respectively, on minimizing and computing fix-points on generally infinite state spaces. We show that if the procedures terminate, then they give correct answers. We then demonstrate that for many of the typical workshop examples, the procedures do terminate and thus provide an automatic way for verifying their properties.

Hybrid Systems: Towards a sustained discipline

- Hybrid Systems II, Ithaca, October 1994 (LNCS 999, 1995)
  P. A. Antsaklis, W. Kohn, A. Nerode, and S. Sastry
- Hybrid Systems III: Verification and Control
  DIMACS, New Jersey, Oct 1995 (LNCS 1066, 1996)
  R. Alur, T. A. Henzinger, and E. Sontag
- Hybrid Systems IV, Ithaca, October 1996 (LNCS 1723, 1997)
  P. A. Antsaklis, W. Kohn, A. Nerode, and S. Sastry
- Hybrid and Real-Time Systems, Grenoble, March 1997 (LNCS 1201)
  O. Maler
- Hybrid Systems: Computation and Control (HSCC)
  First International Workshop, Berkeley, April 13-15, 1998
  T. A. Henzinger and S. Sastry (LNCS 1386)

Reachability Analysis via Face-Lifting

Dang and Maler

First HSCC, LNCS 1386, pp. 96--109, 1998

In this paper we discuss the problem of calculating the reachable states of a dynamical system defined by ordinary differential equations or inclusions. We present a prototype system for approximating this set and demonstrate some experimental results.

Test-of-Time Award, HSCC 2019

Example: Airline safety

$$\dot{x}_1 = -\frac{a_D x_1^2}{m} - g \sin x_2 + \frac{u_1}{m}$$

$$\dot{x}_2 = \frac{a_L x_1 (1 - c x_2)}{m} - \frac{g \cos x_2}{x_1} + \frac{A_L c x_1}{m} u_2$$

Evolution of velocity and flight path angle

Model based on:

Multiobjective hybrid controller synthesis
J. Lygeros, C. Tomlin, and S. Sastry,
Proc. HART, 1997

Example: Airline safety

$$\dot{x}_1 = -\frac{a_D x_1^2}{m} - g \sin x_2 + \frac{u_1}{m}$$

$$\dot{x}_2 = \frac{a_L x_1 (1 - c x_2)}{m} - \frac{g \cos x_2}{x_1} + \frac{A_L c x_1}{m} u_2$$

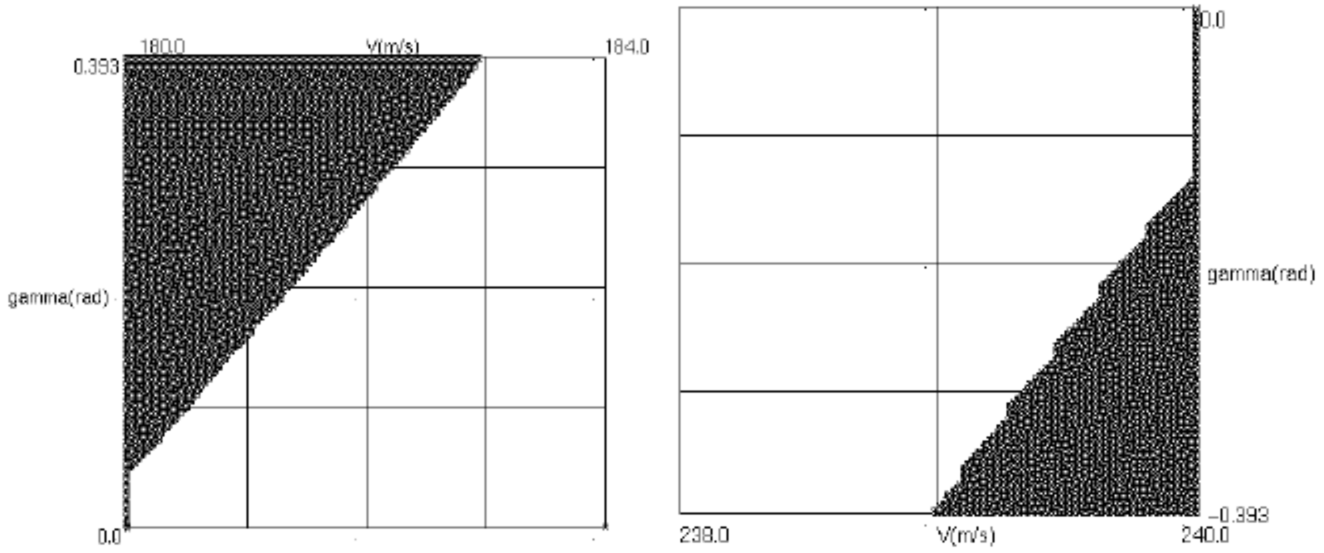


Fig. 7. Airplane Safety

Recap: Hybrid Systems, 1990s

Real-Time: Theory in Practice, REX Workshop
Mook, The Netherlands, 1991

Modeling and semantics

Hybrid Systems: Computation and Control
First International Workshop, Berkeley, 1998

Computational tools and case studies