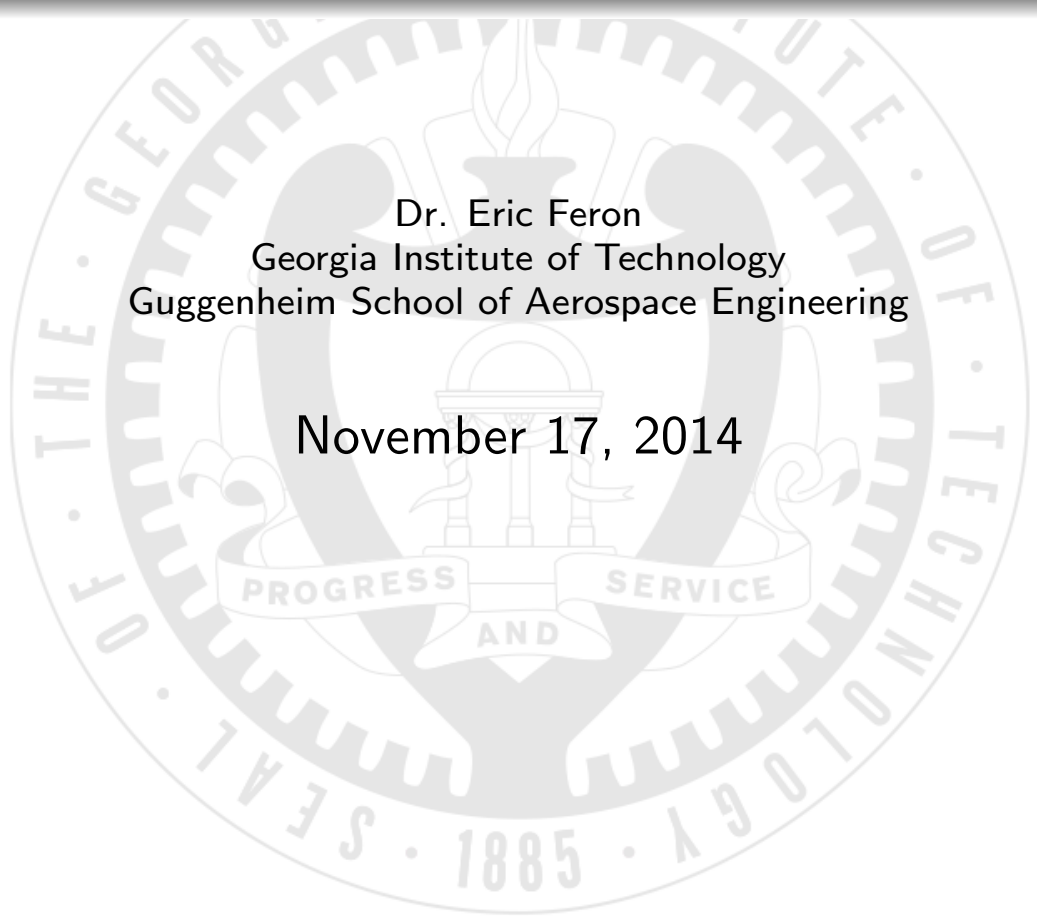
CPS Systems: *Dynamics and control*

Dr. Eric Feron
Georgia Institute of Technology
Guggenheim School of Aerospace Engineering

November 17, 2014



Georgia Institute of Technology

Outline

Georgia Institute of Technology

Dynamical Systems

Dynamical Systems: Intuition

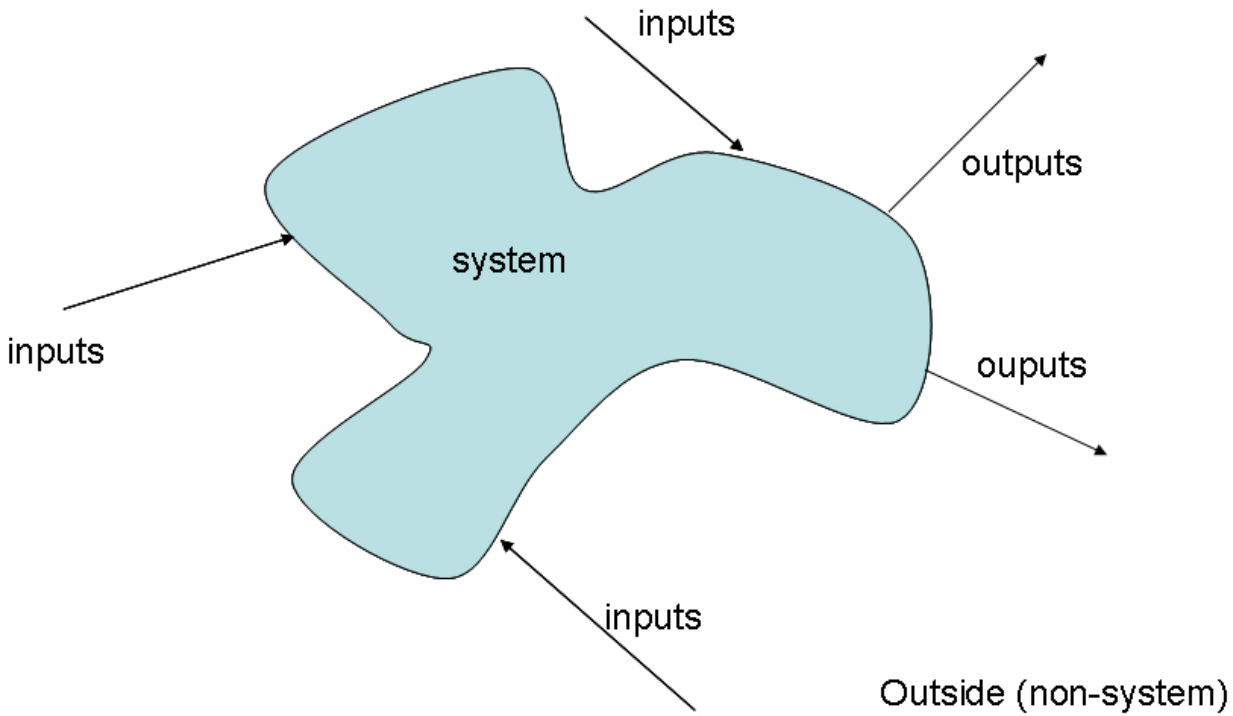


Figure: A generic system

Dynamical Systems: Experimental evidence

Experimental evidence: Proof that what we are talking about is worth it!

A human: Concept follows from observation (homo erectus, homo sapiens, homo sapiens sapiens).

A (cyber-physical) system: Concept *also* follows from observation.

Conceptual difficulty: How can concept follows from observation of *man-made object*? Can it be we build first, we name later? All the time. Meaning and function always precede name.

Georgia Institute of Technology

Dynamical Systems: Experimental evidence

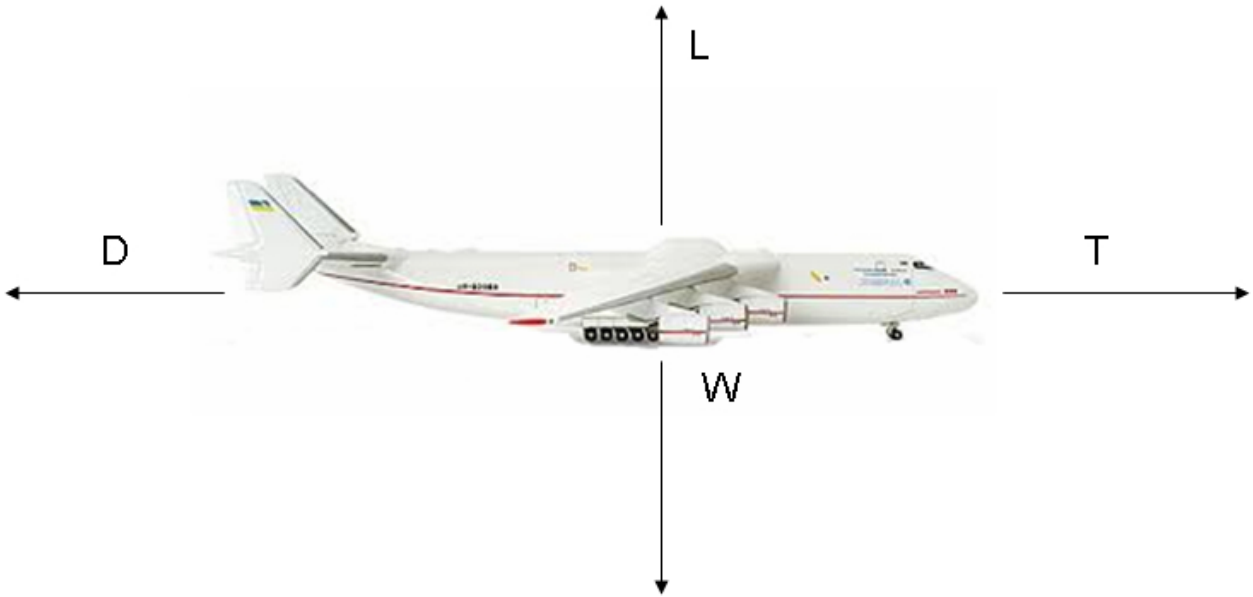


Figure: The Antonov 128

Dynamical Systems: Experimental evidence

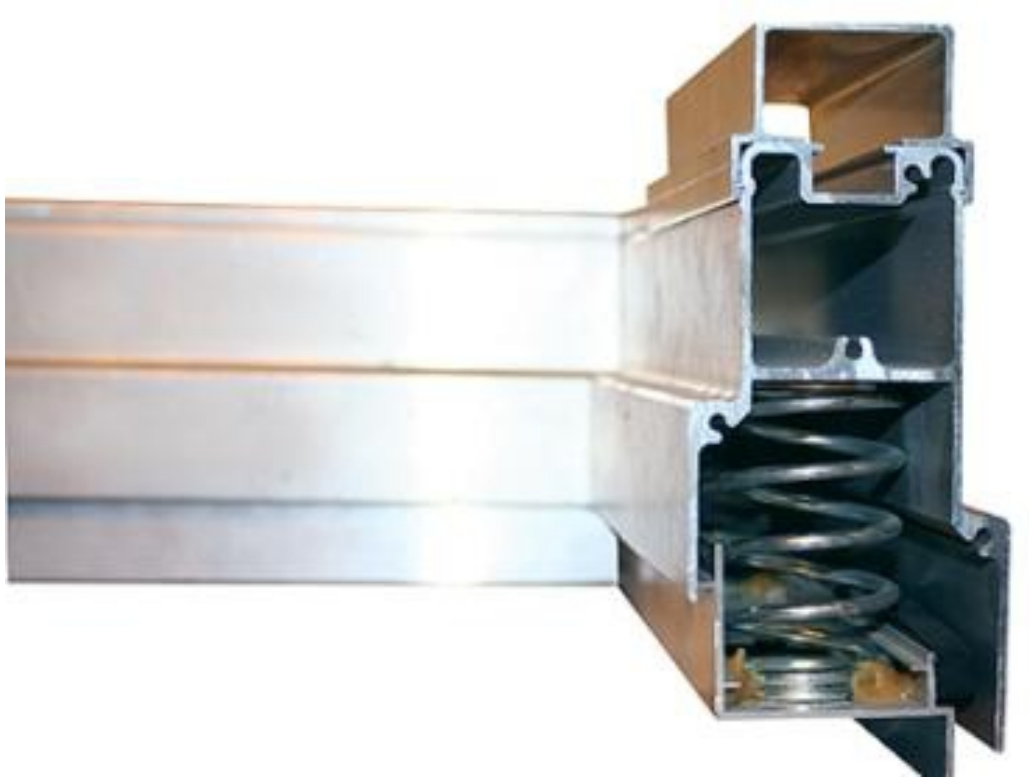


Figure: A Mass-Spring System

Dynamical Systems: Experimental evidence

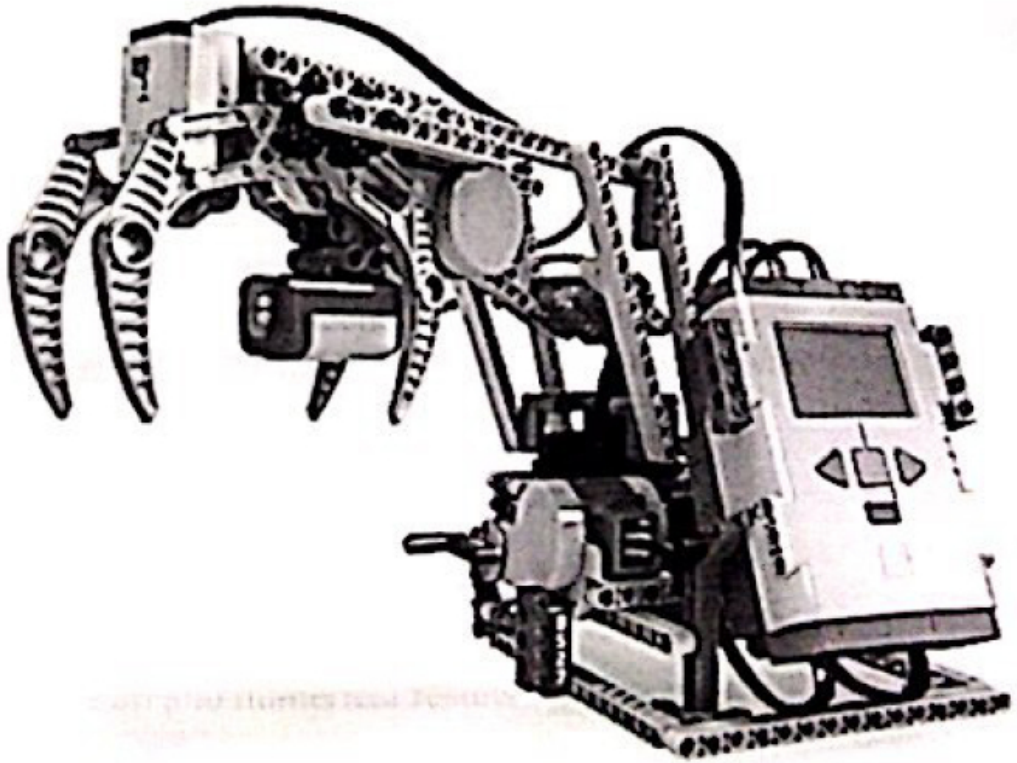


Figure: A cyber-lego

Dynamical Systems: Experimental evidence

Figure: A computer

Georgia Institute of Technology

Automaton

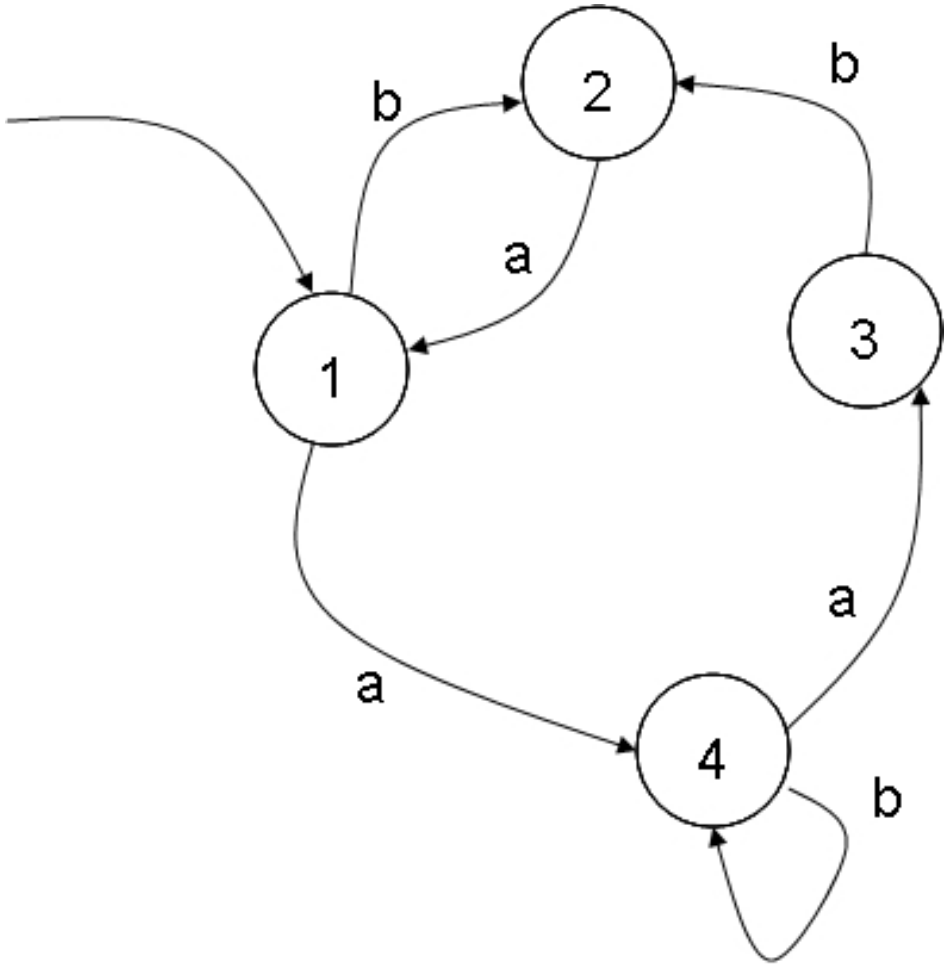


Figure: Automaton

Dynamical Systems: Supporting infrastructure

What math do we need to talk about systems?

- **Analysis:** Concepts of topologies, metrics, norms, functions....
- **Algebra:** Sets, groups, algebras, rings, fields, matrices, operators, etc...
- **Probabilities:** Measure theory, concept of probability, Gaussian, heavy-tailed, other distributions.
- **Logic:** Propositional logic, first-order logic, linear temporal logic.
- **Axioms:** '1' exists and follows '0', precedes '2'....

Role of math: To *clarify* and *disambiguate* what we are talking about, by using a *universal language*.
Is math obfuscation? If obfuscating, then probably lack of language knowledge from listener (less likely), speaker (more likely).
Why math in safety-critical CPS? Because safety suffers no approximate or illogical statements.

Georgia Institute of Technology

Dynamical Systems: Components

What makes an (interesting) system?

- **Time:** A very complicated construct, very useful. *eg* Tics of a clock, pulses of a thritium atom, an ordered set of events. A continuous variable $t \in \mathbf{R}$. A discrete variable $t \in \mathbf{Z}$. An ordered set of events $t \in \{e_1,\ e_2,\ e_3,\ e_4,\ e_5\}$.
- **State:** A collection (possibly infinite, possibly changing) of variables that capture what the system 'is': Position, speed, temperature, stored energy, open/closed, angle of attack, pitch angle. variables may be numbers, elements of a set, whole functions ('shapes').
- **Inputs:** A collection (possibly infinite, possibly changing) of 'exogenous variables' (independent from the system), that can act on the system and 'influence' its state: throttle/brakes of a car, instructor/movie/tablet on a human, force/moment on a rigid body.

Georgia Institute of Technology

Dynamical Systems: Components

What makes an (interesting) system?

- **Outputs:** A collection (possibly infinite, possibly changing) of 'endogenous variables' (dependent on the system), that the system can express to the world: position, speed, entropy, flight path angle.
- **Transition function:** A mathematical way to describe *How system changes with time*.

In computer science, a mathematical description of what a system does is called *operating semantics*
In physical sciences, the mathematical description is called *system dynamics*.

Georgia Institute of Technology

Dynamical Systems: Components

What makes an (interesting) system?
HUGE DIFFERENCE BETWEEN PHYSICAL AND CYBER SYSTEMS:

- The operating semantics of a program is exactly what it does
- the operating semantics of a system is never what it does. Only an approximation of it.

A whole panoply of physical/cyber science words with equivalent meaning! How do we marry the two sets of vocabulary, semantics? Dictatorship is not acceptable.

Georgia Institute of Technology

Mass-spring system

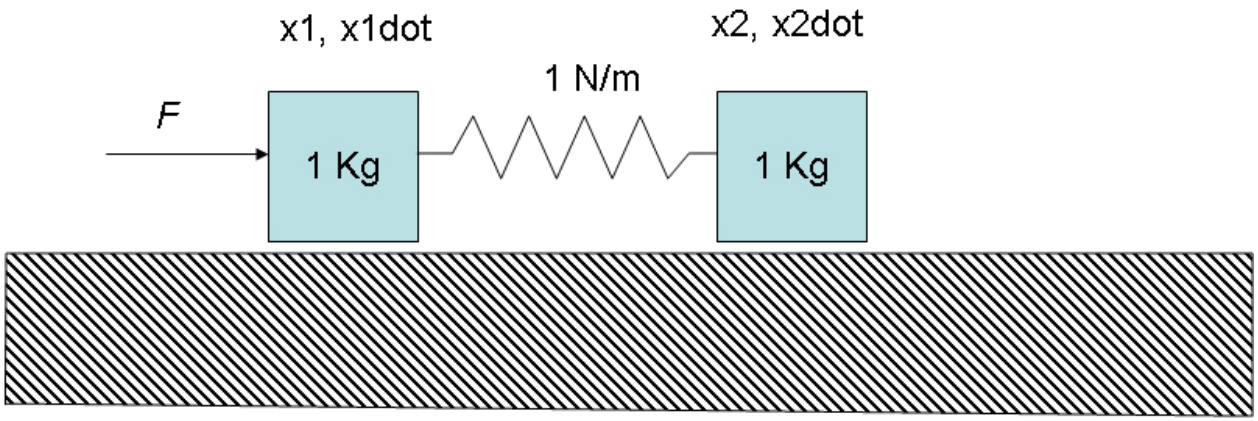


Figure: A Mass-Spring System

Mass-spring system

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} F,$$

$$x_1(0) = 1,\ x_2(0) = 2\ \dot{x}_1(0) = -2; \dot{x}_2(0) = 1,$$

$$y = \begin{bmatrix} 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix}$$

Georgia Institute of Technology

Navion

Figure 3-1. Photo of Ryan Navion (N66UT).

Figure: The Navion

Navion (ct'd)

$$\begin{bmatrix} \Delta\dot{u} \\ \Delta\dot{w} \\ \Delta\dot{q} \\ \Delta\dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.09148 & 0.04242 & 0 & -32.17 \\ 10.51 & -3.066 & 152 & 0 \\ 0.2054 & -0.05581 & -2.114 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta u \\ \Delta w \\ \Delta q \\ \Delta\theta \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -12.64 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta\delta_e \\ \Delta\delta_T \end{bmatrix}$$

Figure: Navion Longitudinal Dynamics

$$\begin{bmatrix} \Delta\dot{v} \\ \Delta\dot{p} \\ \Delta\dot{r} \\ \Delta\dot{\phi} \end{bmatrix} = \begin{bmatrix} -0.093 & 0 & -152.6 & 32.17 \\ -0.059 & -5.816 & 1.854 & 0 \\ 0.030 & -0.548 & -0.953 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta v \\ \Delta p \\ \Delta r \\ \Delta\phi \end{bmatrix} + \begin{bmatrix} 0 & 3.469 \\ 15.74 & 1.380 \\ 0.486 & -4.288 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta\delta_a \\ \Delta\delta_r \end{bmatrix}$$

Figure: Navion Lateral-Directional Dynamics

Georgia Institute of Technology

Douglas Corp. 3 (DC-3)

Figure: DC-3

Systems

Engine 1: good/failed

Engine 2: good/failed

(Imagine many more systems that can be good or failed: avionics, displays, Auxiliary Power Units)

Probability of individual engine failures: 0.01 over period of 1 hour

Georgia Institute of Technology

Douglas Corp. 3 (DC-3)

State: Good/Bad? Kind of coarse... use probabilities instead.
New state: P(good) and P(bad) for each engine.
Aircraft state: (P1g, P1b, P2g, P2b) = (P(engine1=good), P(engine2=good)) Aircraft output: (Pr, Pf, Pnf) = (P(aircraft running), P(aircraft flyable), P(aircraft non flyable))
Time: $t = 0,\ 1,\ 2,\ 3,\ \ldots$ hours.
Given $(P_{1g}, P_{1b}, P_{2g}, P_{2b})$ at time t, what are they at time $t+1$?

$$\begin{bmatrix} P_{1g}^{+} \\ P_{1b}^{+} \\ P_{2g}^{+} \\ P_{2b}^{+} \end{bmatrix} = \begin{bmatrix} 0.99 & 0 & 0 & 0 \\ 0.01 & 1 & 0 & 0 \\ 0 & 0 & 0.99 & 0 \\ 0 & 0 & 0.01 & 1 \end{bmatrix} \begin{bmatrix} P_{1g} \\ P_{1b} \\ P_{2g} \\ P_{2b} \end{bmatrix}$$

Outputs:

$$\begin{bmatrix} P_r \\ P_f \\ P_{nf} \end{bmatrix} = \begin{bmatrix} P_{1g}P_{2g} \\ P_{1g}P_{2b} + P_{1b}P_{2g} \\ P_{1b}P_{2b} \end{bmatrix}$$

Georgia Institute of Technology

A computer program

```
int8 a;
int8 i;
a:=5
i:=0
while (true) do
if (a-2*(a div 2) != 0)
a:=3*a+1;
i=i+a else
a := a div 2+i;
i = i+a od
```

A computer program operational semantics

```
int8 a
int8 a1
int8 i
int8 i1
a:=5
i:=0
while(true){ if (a-2 (a div 2) != 0) then
a1 := 3*a+1;
i1 := i+3*a+1;
else a1 := a div 2;
i1 := i+a div 2;

a:=a1;
i:=i1; }
```

So we make sure the program is written in 'first-order recurrence' form.

Dynamical System Languages

Mathematical representations of dynamical systems

- Differential Equations (ordinary, partial)
- Automata / regular expressions
- Hybrid systems
- Markov Chains / Continuous Markov Chains

Georgia Institute of Technology

The computer languages of dynamical systems

In short: Any Turing machine will do. But it's not totally easy to use. Thus *domain-specific languages*

- **Matlab** Free to you now, very costly later.
- **Xmath** Dead.
- **Scilab** Free to you now, free later.

Core characteristics: Custom-designed dynamical systems functions;

```
s= %s
num = s + 1;
den = s^2 + 3 * s + 3;
sys=num/den;
system=syslin('c',sys)
evans(system)
```

Georgia Institute of Technology

Graphical mathematical languages

Simulink:

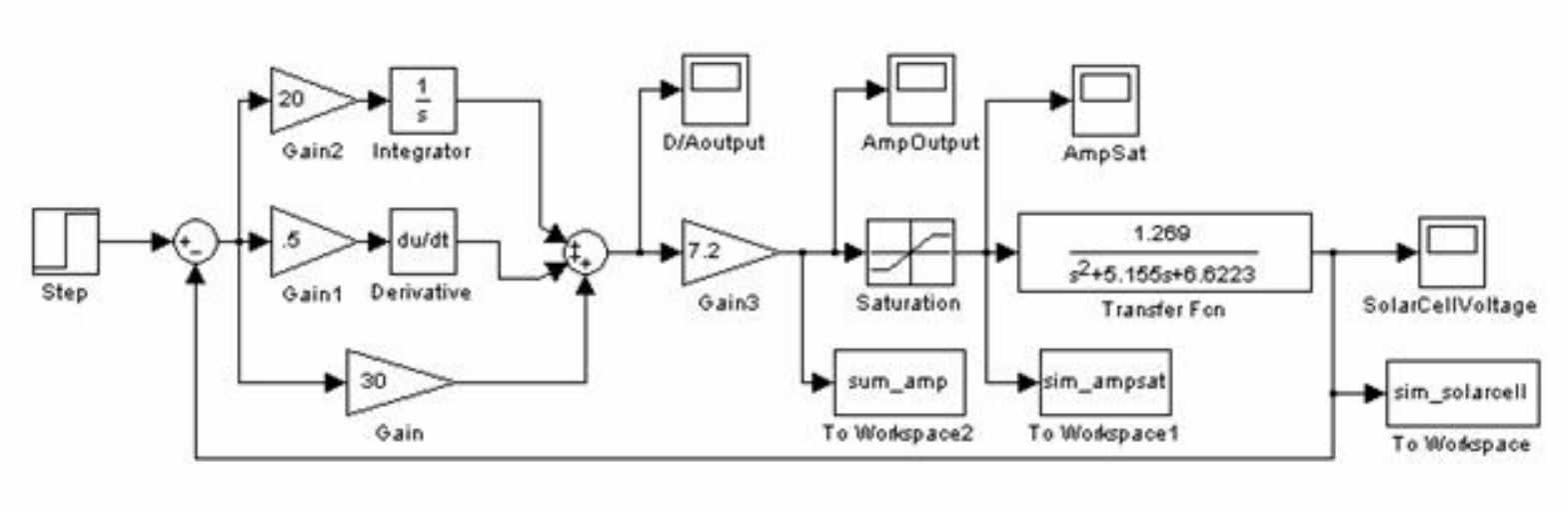


Figure: Simulink system example

Graphical mathematical languages

Stateflow:

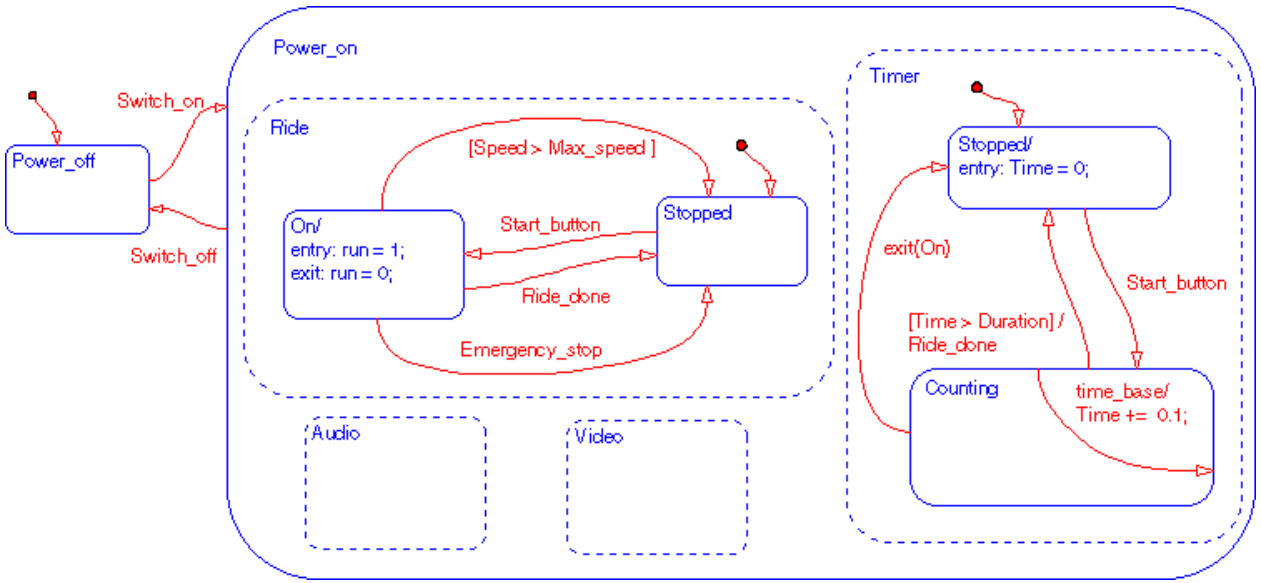


Figure: Stateflow system example

Graphical mathematical languages

SCADE:

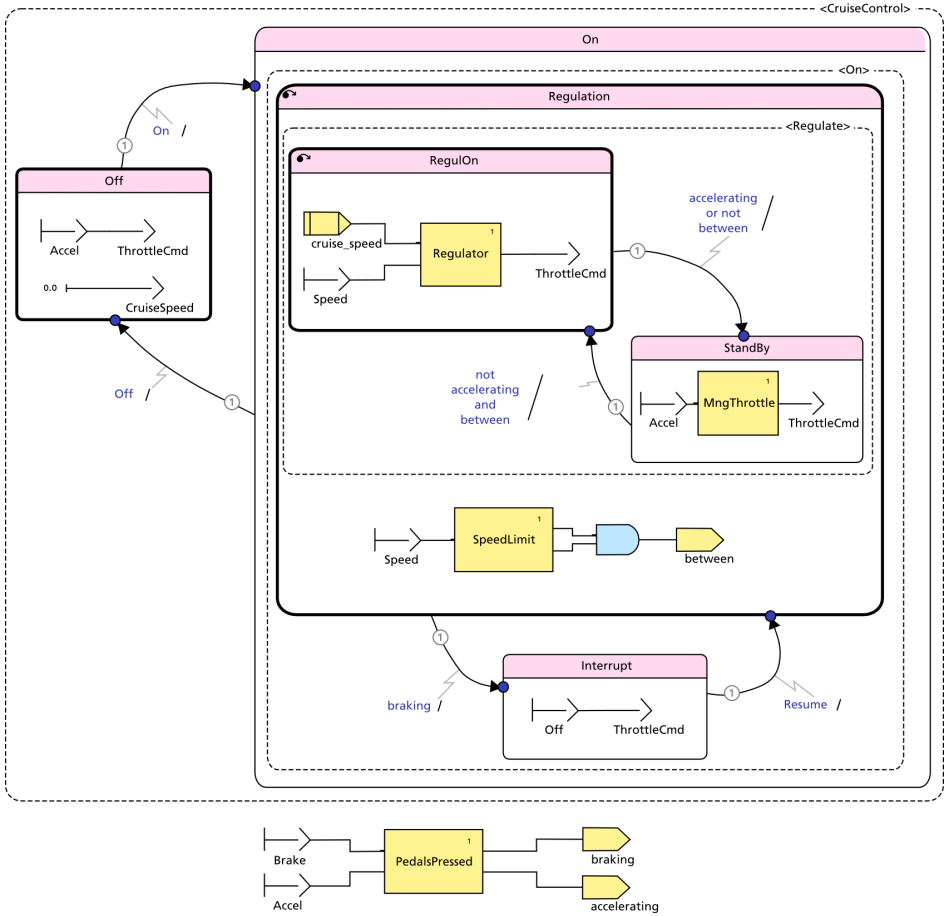


Figure: SCADE system example

Important system notions

Causality

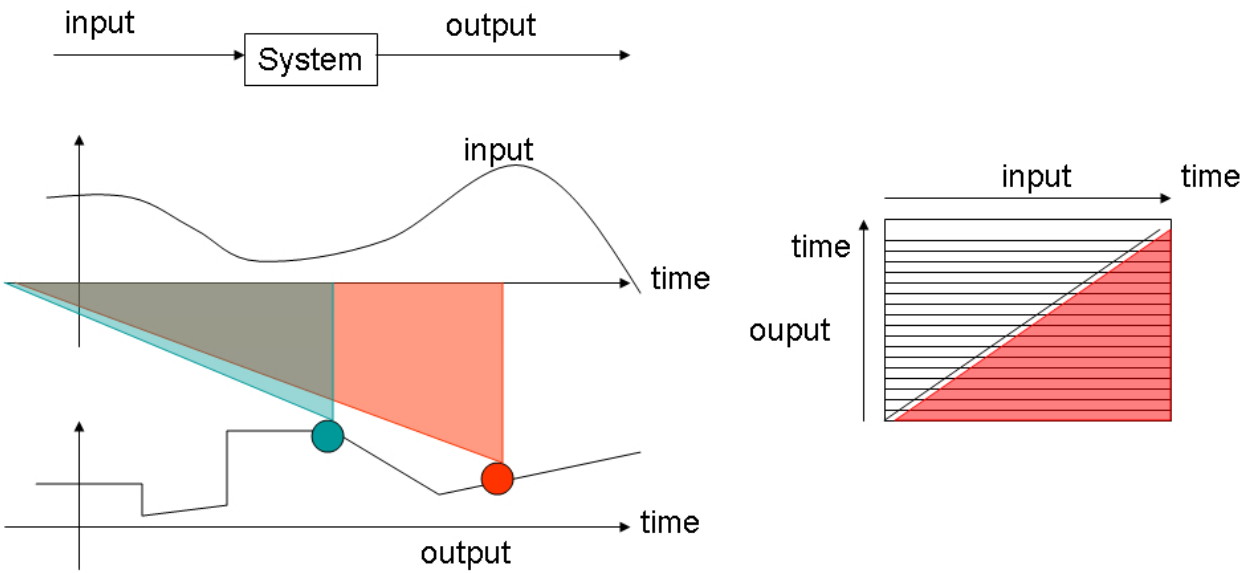


Figure: causality

'You can't act on what you do not know yet'

Causality

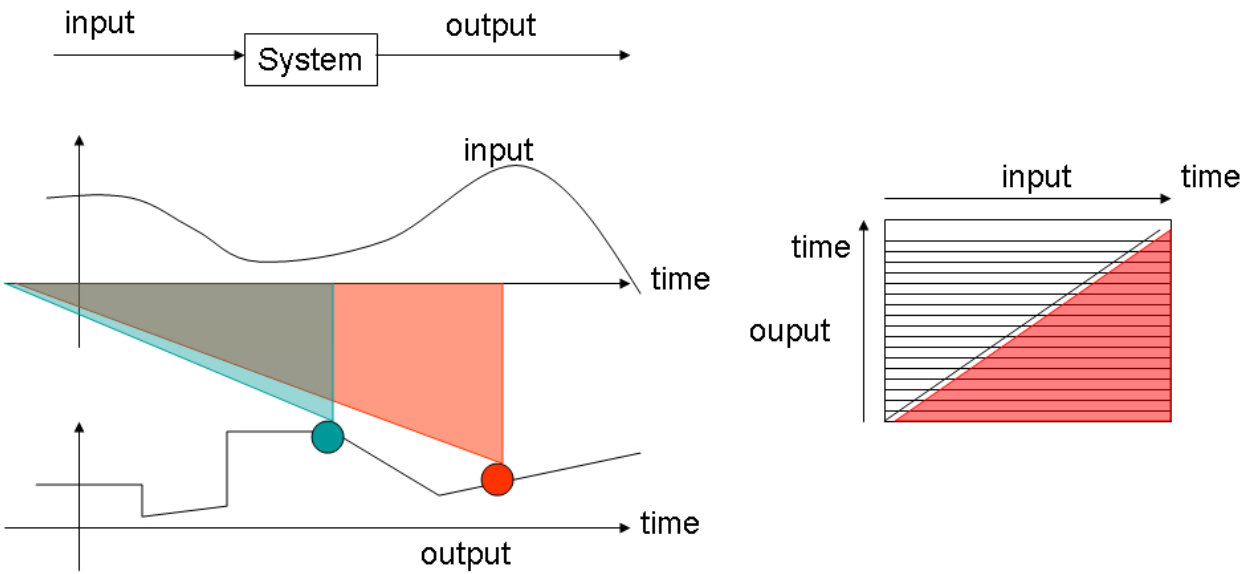


Figure: causality

'You can't act on what you do not know yet'
'Or really??'

Causality

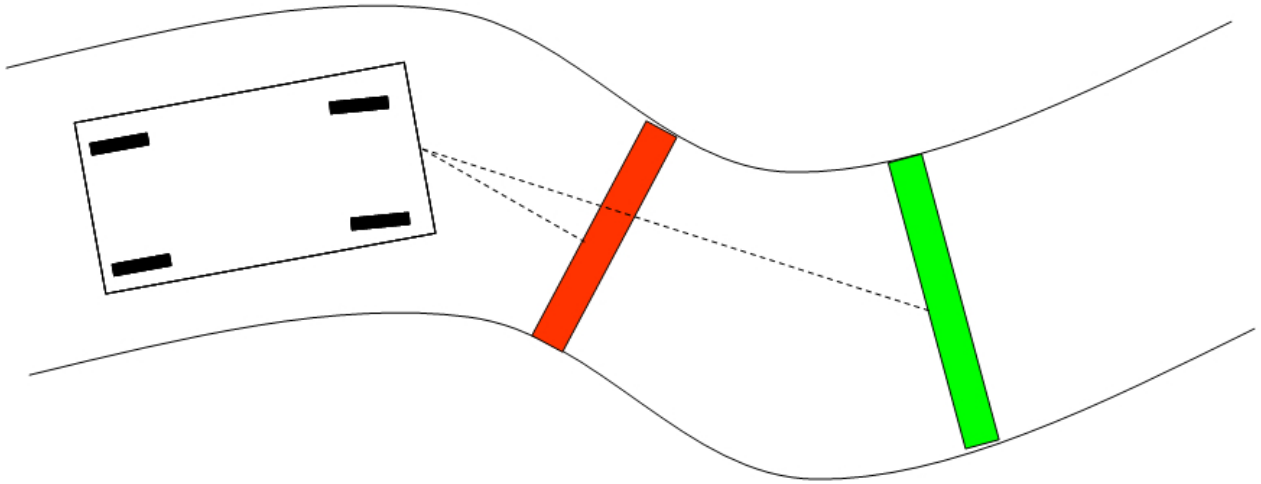


Figure: anti-causality in practice

We love anticausal.
No SEC or legal penalties for acting anti-causal in a car.

System Representations

Return to Mass-Spring system.

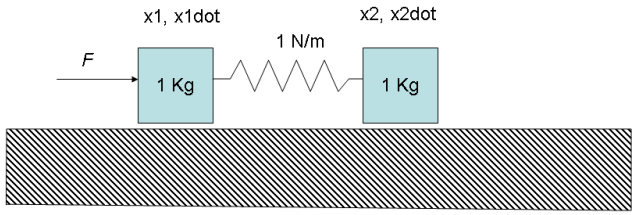


Figure: Controlled mass-spring system

Differential equation or difference equation?

$$\frac{d}{dt}x(t) = Ax(t) + BF(t),\ y(t) = Cx(t),\ \text{or}$$

$$x_{(k+1)\tau} = e^{A\tau}x(k\tau) + \left(\int_0^\tau e^{A\tau}d\tau\right)BF(k\tau),\ y(k\tau) = Cx(k\tau)?$$

We can capture the first via Newton's law, but we'll only see the second with LabView.

Uncertainty

Figure: DC-3

Is probability P of individual engine failures truly 0.01 over period of 1 hour? We introduce *Uncertainty Model*.

$$\begin{bmatrix} P_{1g}^{+} \\ P_{1b}^{+} \\ P_{2g}^{+} \\ P_{2b}^{+} \end{bmatrix} \in \left\{ \begin{bmatrix} 1-P & 0 & 0 & 0 \\ P & 1 & 0 & 0 \\ 0 & 0 & 1-P & 0 \\ 0 & 0 & P & 1 \end{bmatrix} \begin{bmatrix} P_{1g} \\ P_{1b} \\ P_{2g} \\ P_{2b} \end{bmatrix}, \; 0.008 \leq P \leq 0.012 \right\}$$

Abstractions

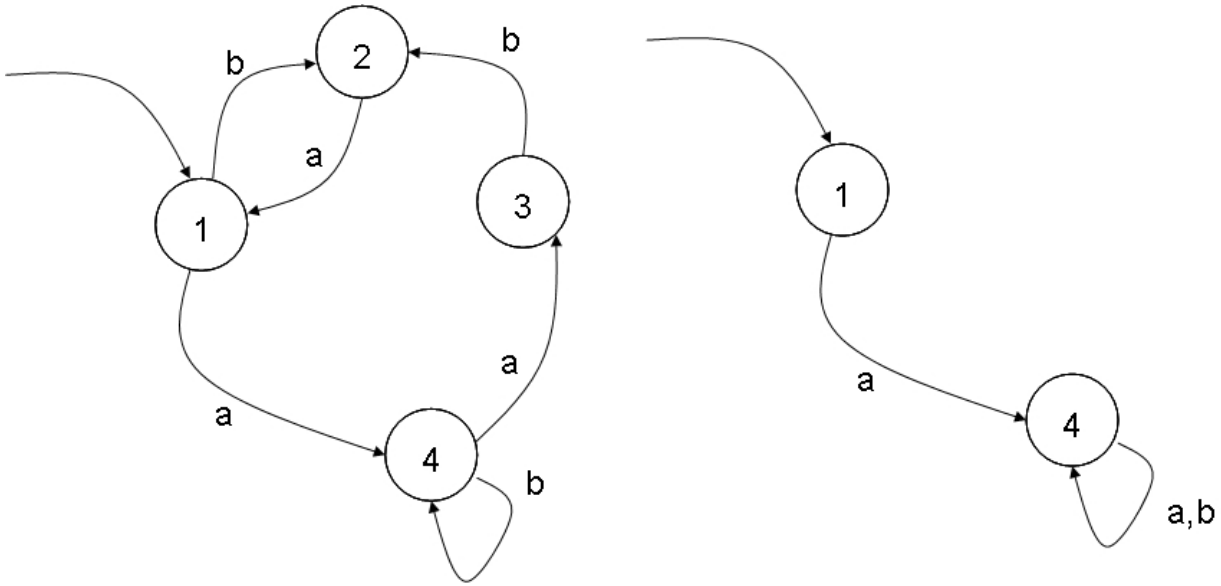


Figure: Automaton abstraction

Abstractions and uncertainty

Abstractions & Uncertainty models are same....
Abstractions sometimes because we do not have enough system knowledge, sometimes because we *want to forget* about system knowledge!
Why is that? See later....

Georgia Institute of Technology

Level of Detail and Accuracy

Level of detail: Number of states of a system.
Accuracy: Ability for a system to explain observed data and predict future behaviors.

Georgia Institute of Technology

Level of Detail and Accuracy: Boston Logan airport

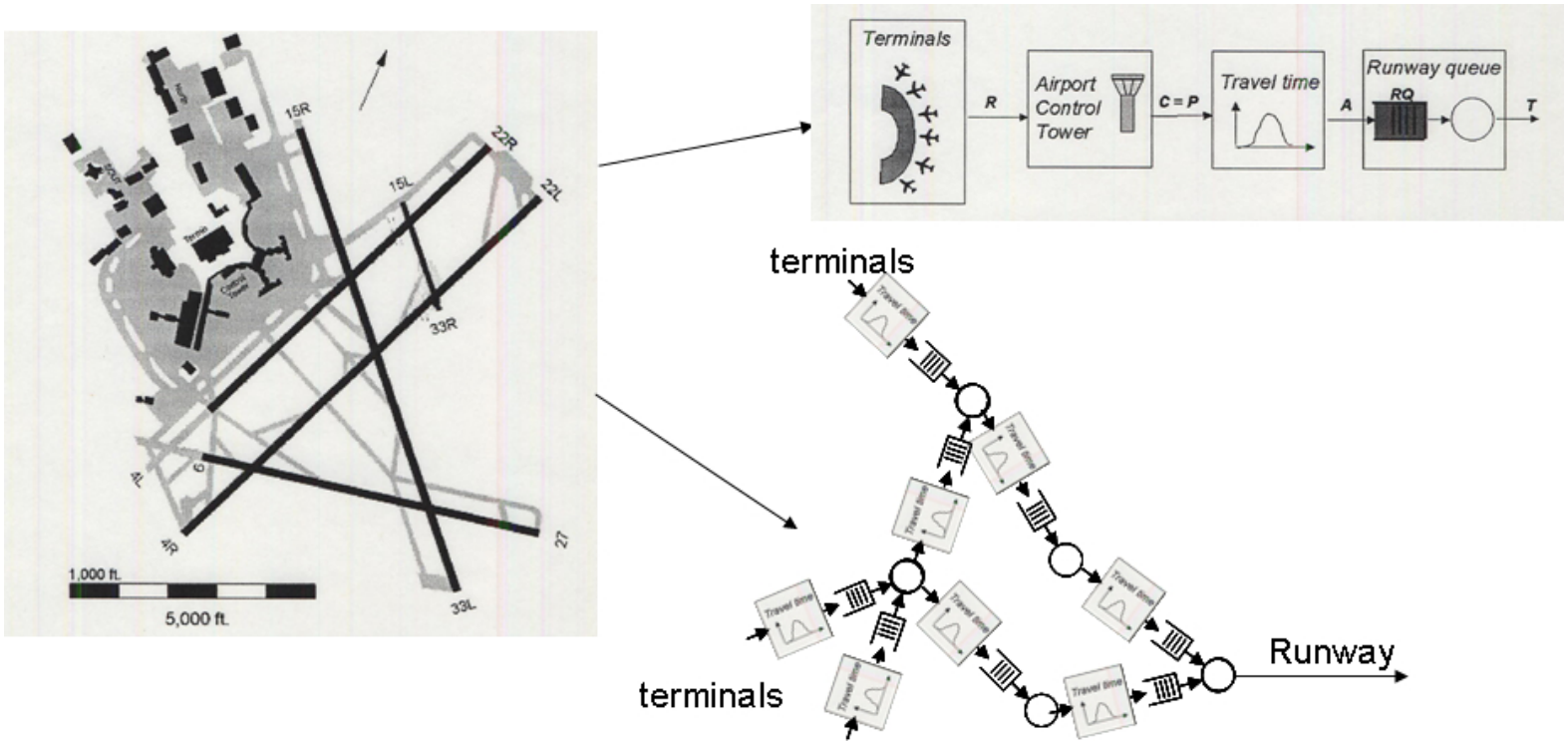


Figure: Boston Logan Airport and possible models

Level of Detail and Accuracy: Costs of calibration

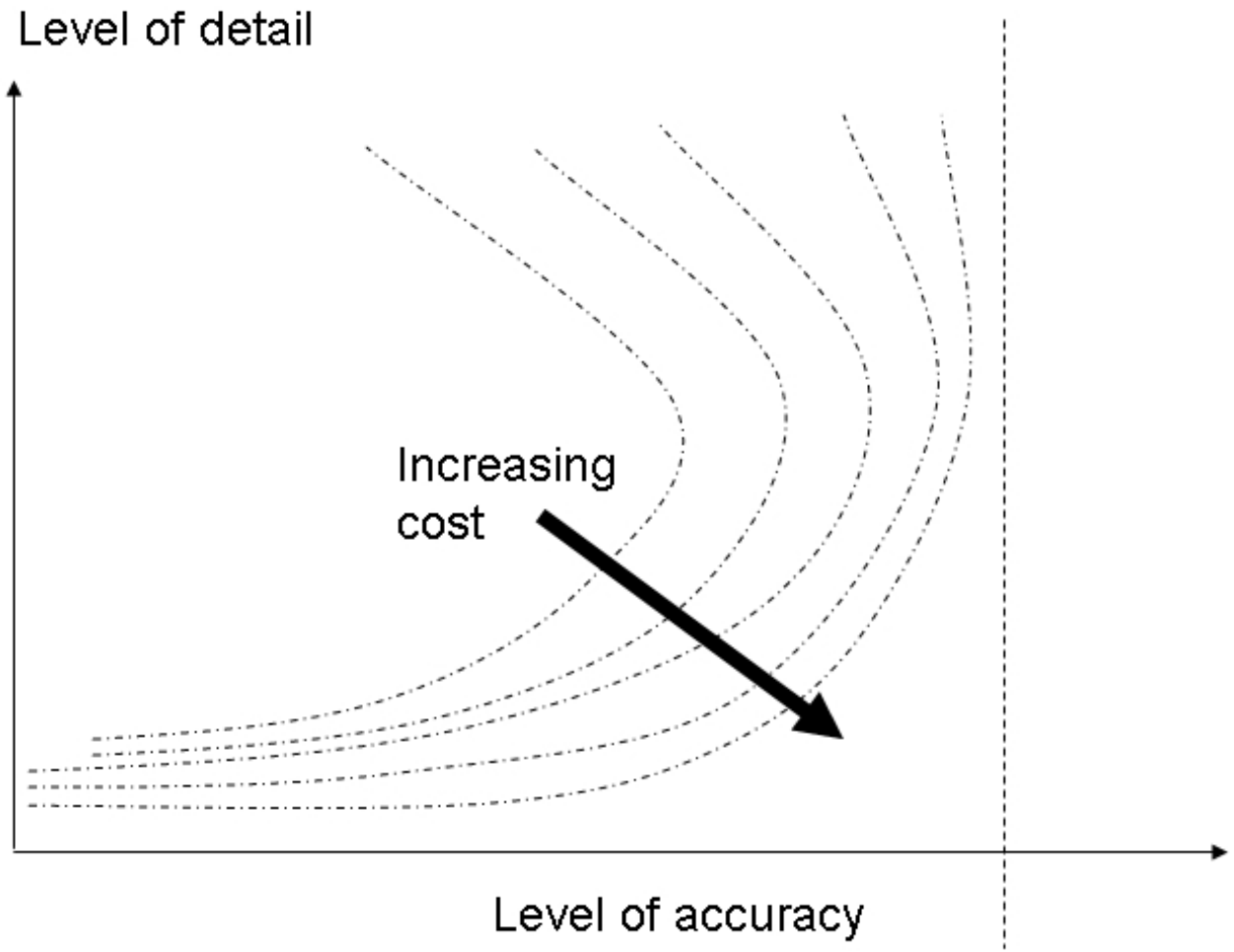


Figure: Notional Level of Detail vs. Accuracy contour plot

System axiomatic semantics

Axiomatic Semantics

What the *meaning* of a system is. What its properties are, and *why* these properties hold.
Core properties of safety-critical systems are *safety* and *liveness*

Safety: The system eventually always stays away from a (presumably dangerous) given set of states.
Liveness: The system always eventually returns to a (presumably desirable) given set of states.

Georgia Institute of Technology

nonfunctional safety: Absence of run-time errors

Run-time errors: Divide-by-zero, stack overflows, out-of-range number, out-of-bounds array index...
Dangerous sets: Divider variable is zero, stack is too tall, number is too large/small, array index is too large.

Safety: Dynamic system stability

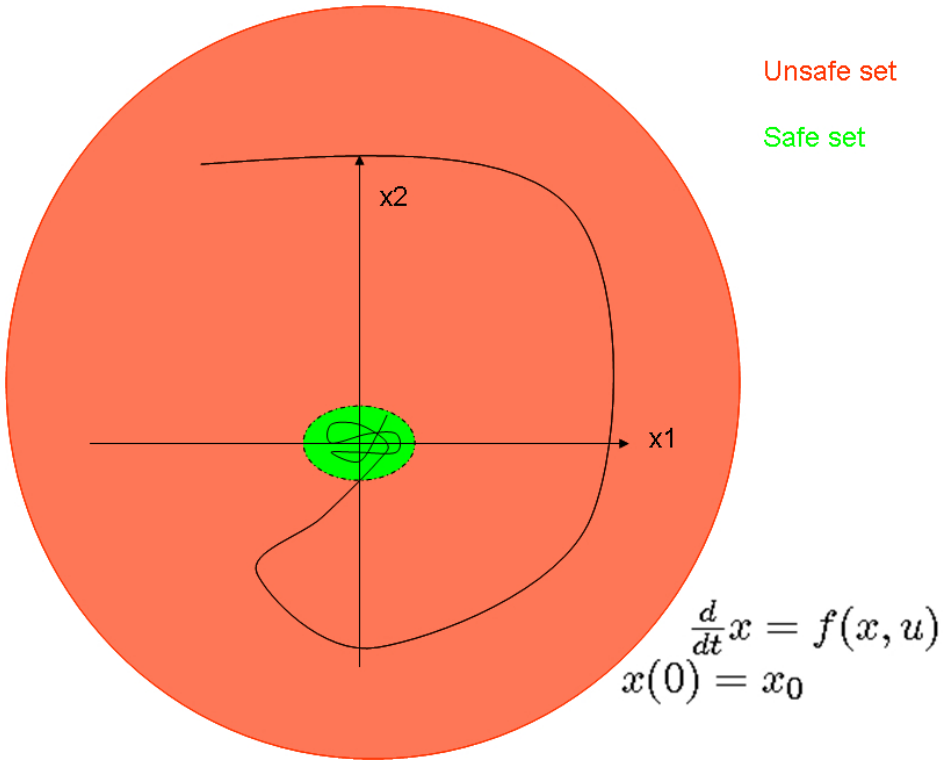


Figure: Dynamical system stability

Safety: Collision avoidance

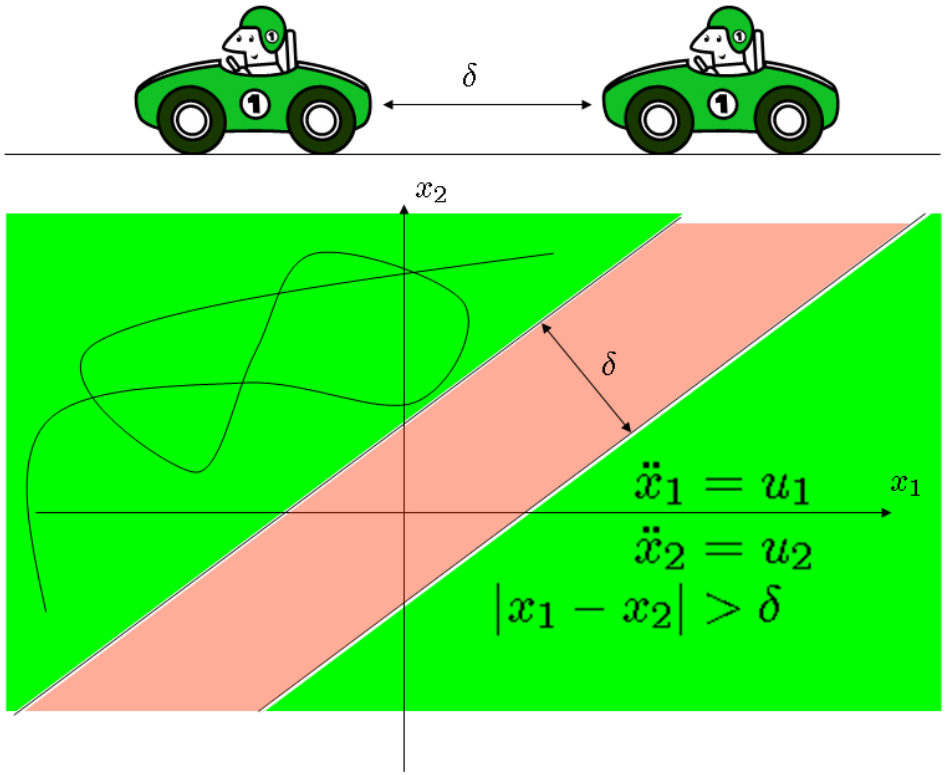
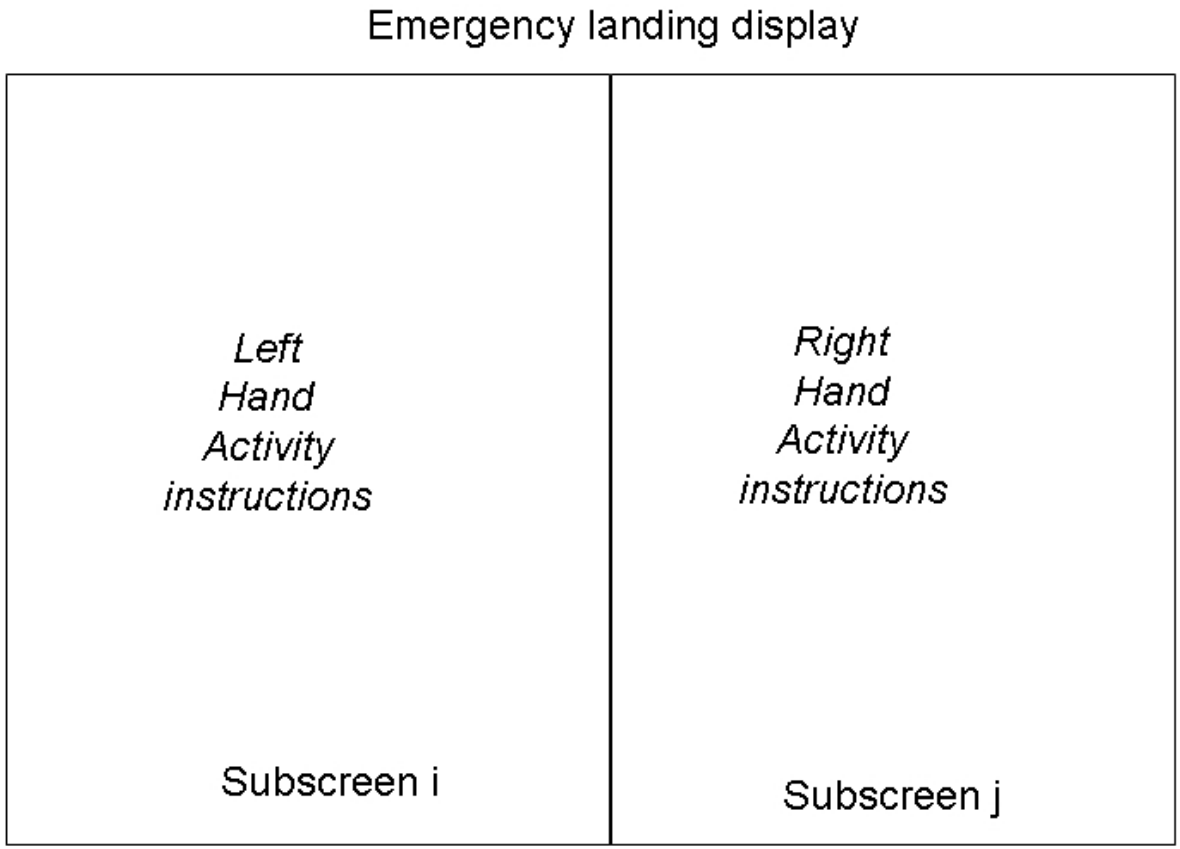


Figure: Collision avoidance

Safety: Collision avoidance



$$(i,j) \notin (right, left), (right, right), (unknown, *), (*, unknown)$$

Liveness: Traffic Light

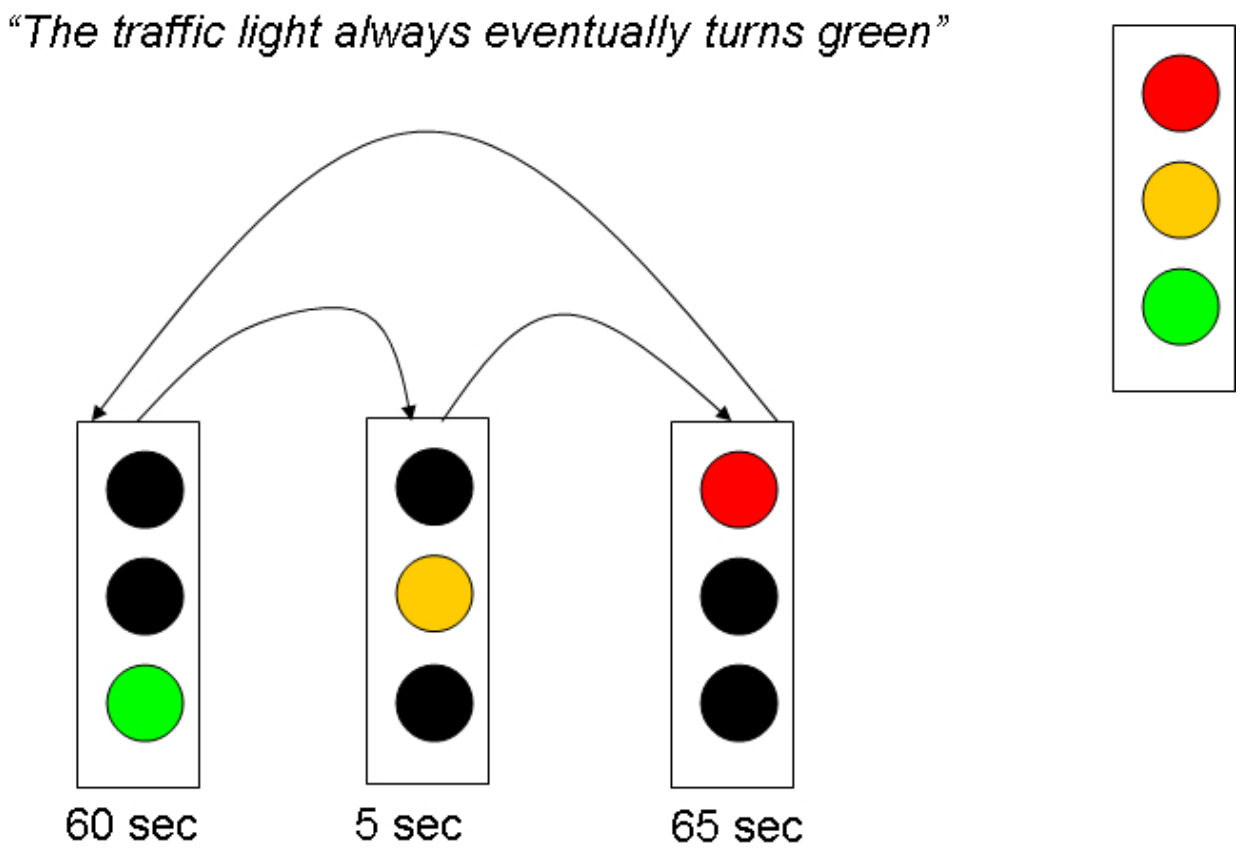


Figure: Traffic light liveness: Classical system

Liveness: Traffic Light

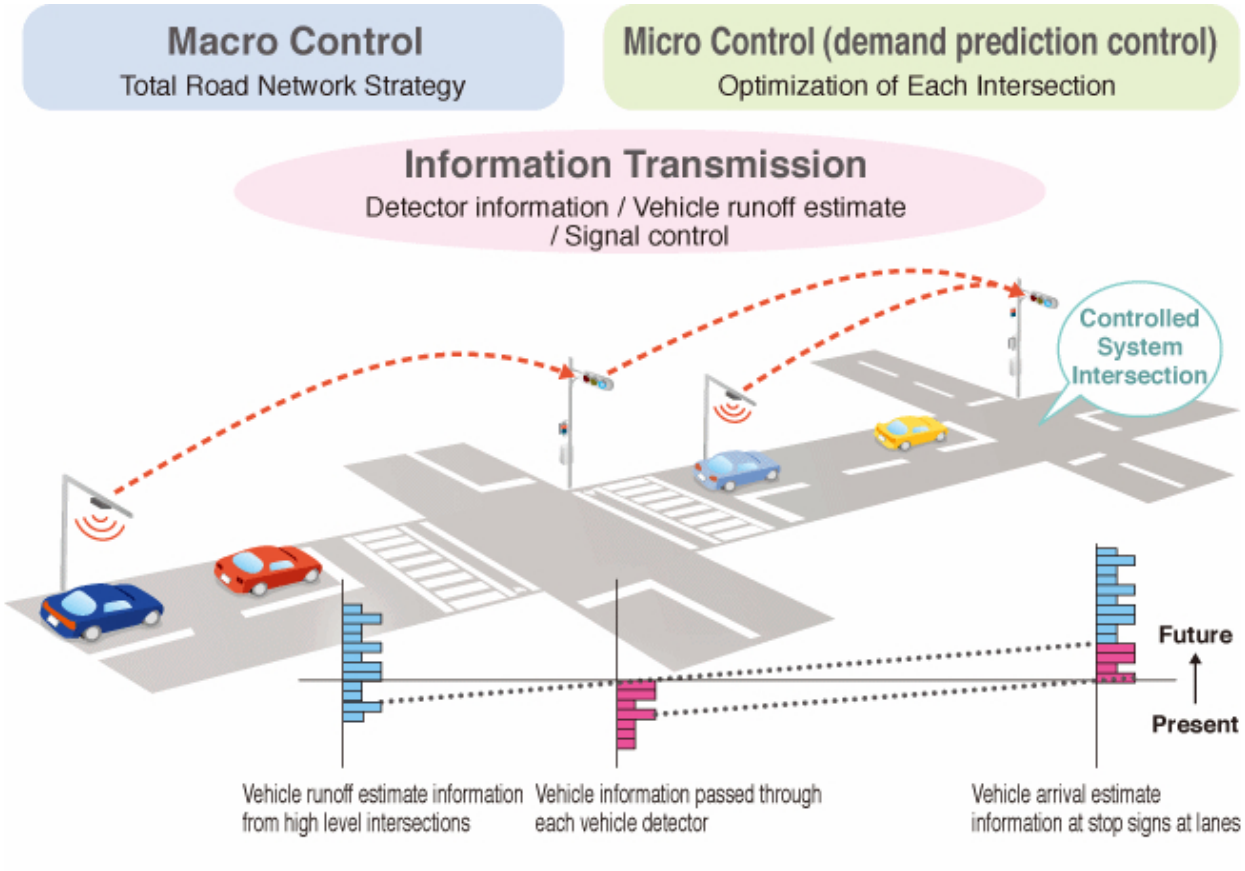


Figure: Traffic light liveness: Evolution

Liveness: Traffic Light

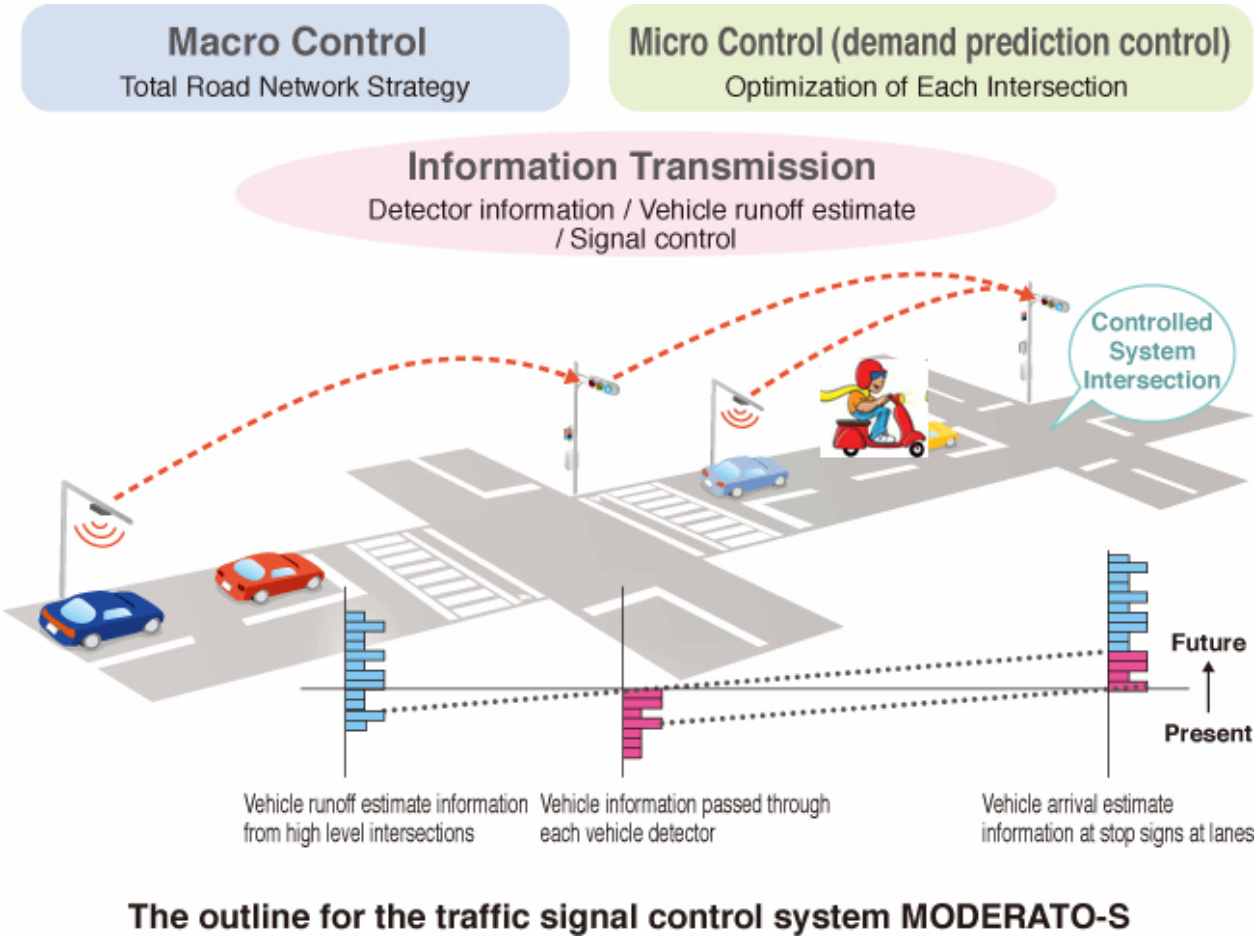


Figure: Traffic light liveness: How about my moped? Not enough magnetic footprint

Liveness: human expert

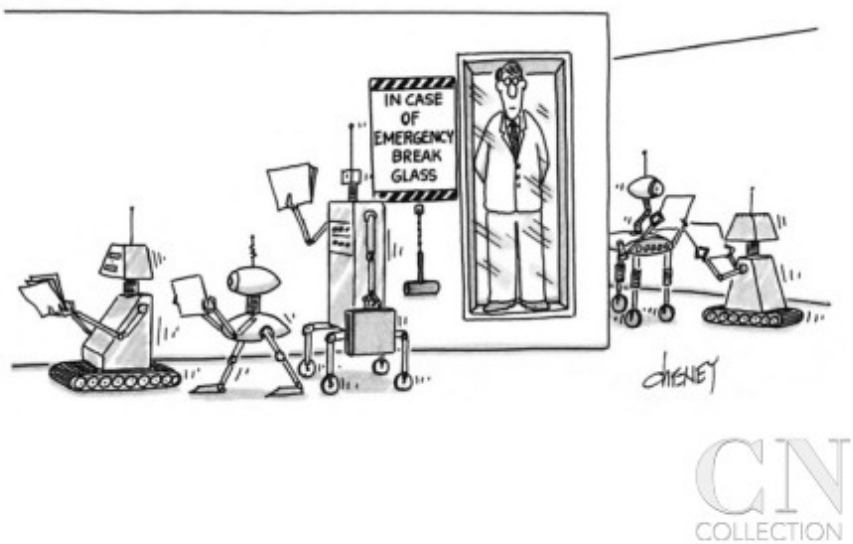


Figure: A human is always available to take care of emergencies = 'it is always true that a human can eventually fix technology's problems

Example: Hudson river landing
Counter-example: Air France 447.

The languages of semantics

Need *unambiguous languages* to express semantics of systems (what they do and why).
Logic together with field-specific knowledge.

1. **propositional logic:** "(I am clever) and (I play lego)"
2. **first-order logic:** "If (A is clever) then (A will pass)", "(I am clever)", "(I am clever) thus (I will pass)"
3. **Linear-Temporal logic:** "When (I am clever) then (I will pass)"

Hierarchy of increasingly sophisticated logical frameworks to address increasingly larger set of concerns. Ideally, need further logical environments to handle time explicitly. Handling of entire functions:
'if $t > 2$, then $(a(t) > 0.9)$ and $(a(t) < 1.1)$' (settling time)
'there exists (t=0.1) and (a(t)=1)' (rise time)

Georgia Institute of Technology

The computer languages of semantics

Needs: Languages to talk about properties of systems that can be written/read by computers and evidence supporting it.
Challenge: A lot more difficult than languages to describe systems. And much less common and commonly used.
Non-exhaustive list: ACSL (ANSI C Specification Language), PVS, Coq. Custom languages come with many static program analyzers (eg ASTREE). Certainly still room for improvements (see examples later).
Very few (if any) graphical languages to describe system semantics: Significant barrier that separate engineering from computer science.

Georgia Institute of Technology

Feedback

Feedback Systems: Basic Principles

What's needed: A system, some specifications, some computing ability, some power.

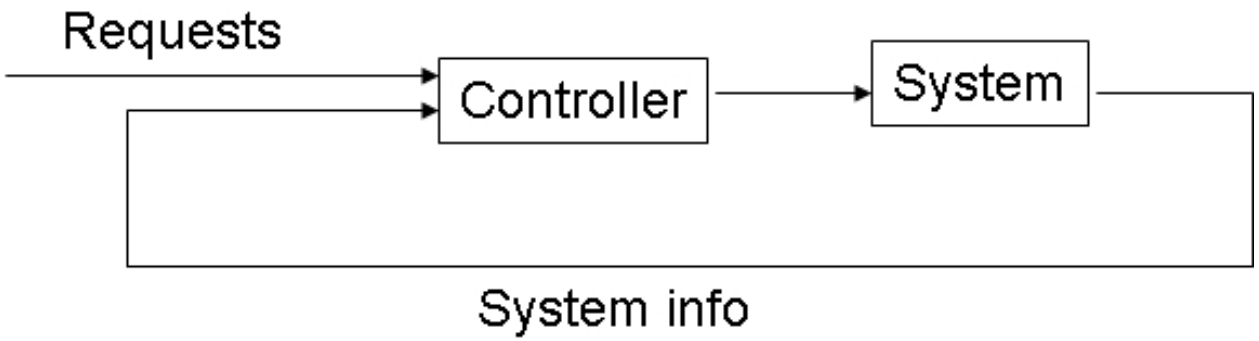


Figure: Basic feedback system

X-Cell MIT helicopter

Figure 2-1: Instrumented X-Cell helicopter in flight

Georgia Institute of Technology

Helicopter Control System

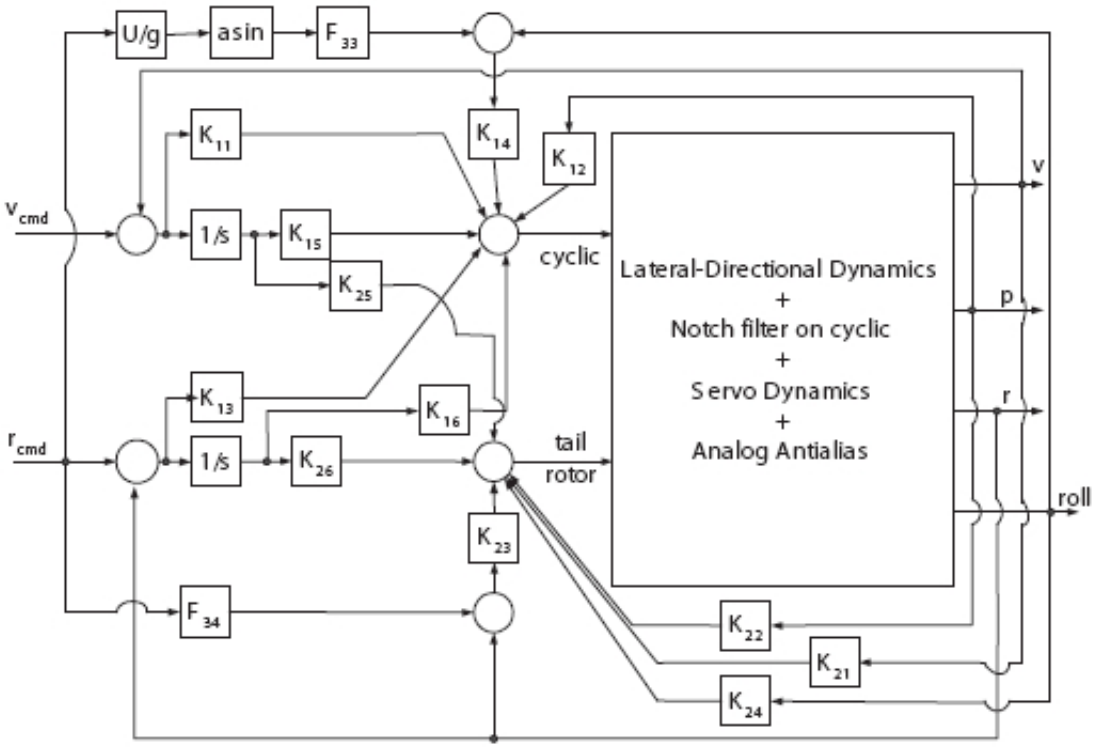


Figure: Lateral-directional controller

Helicopter Control System

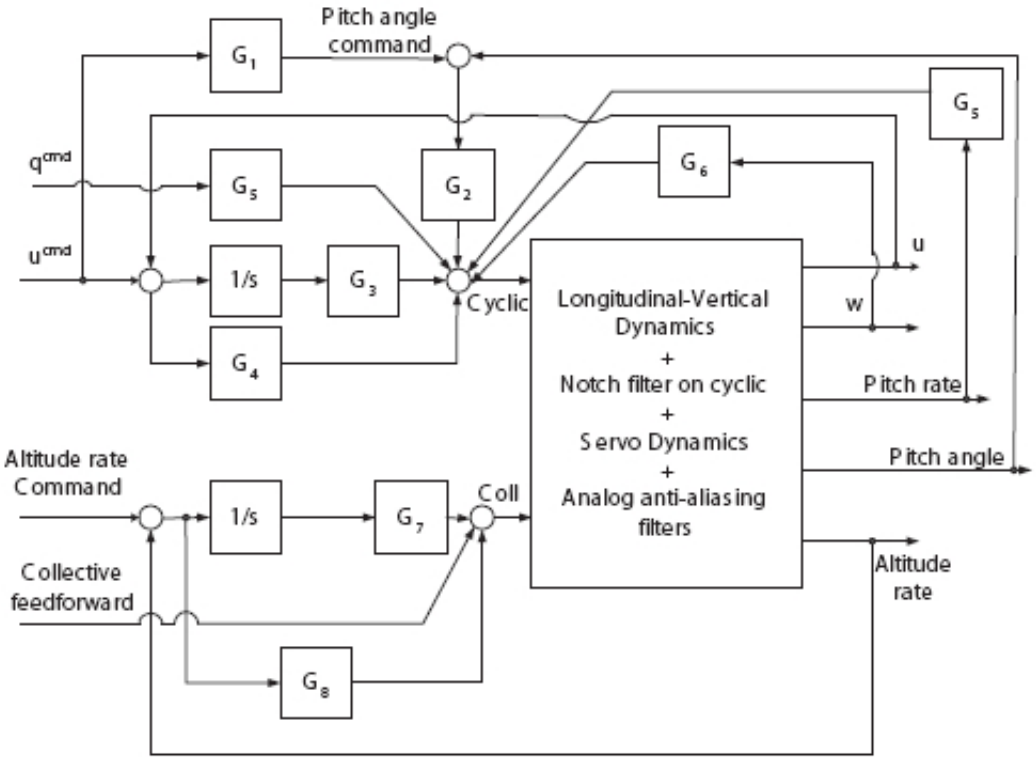


Figure: Longitudinal controller

Roll maneuver as automaton

Figure 4-11: Phases of an autonomous aerobatic maneuver

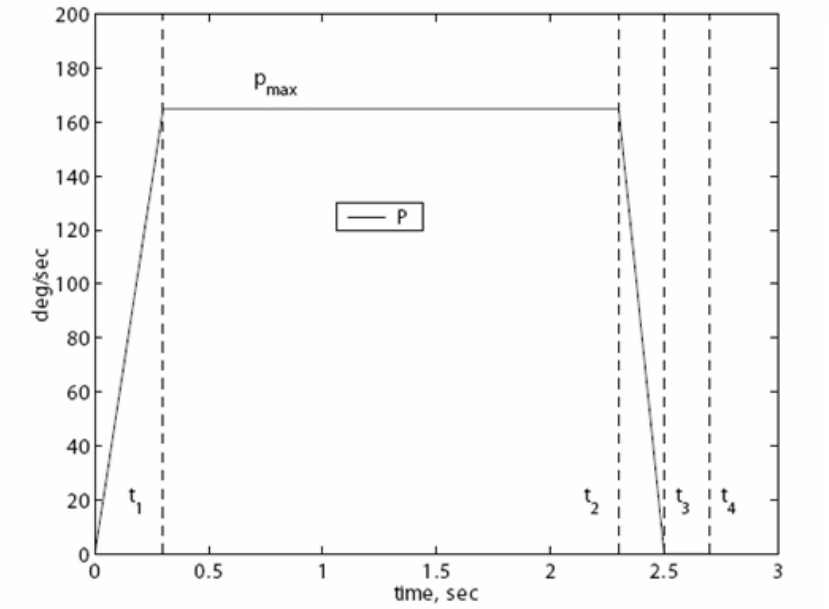
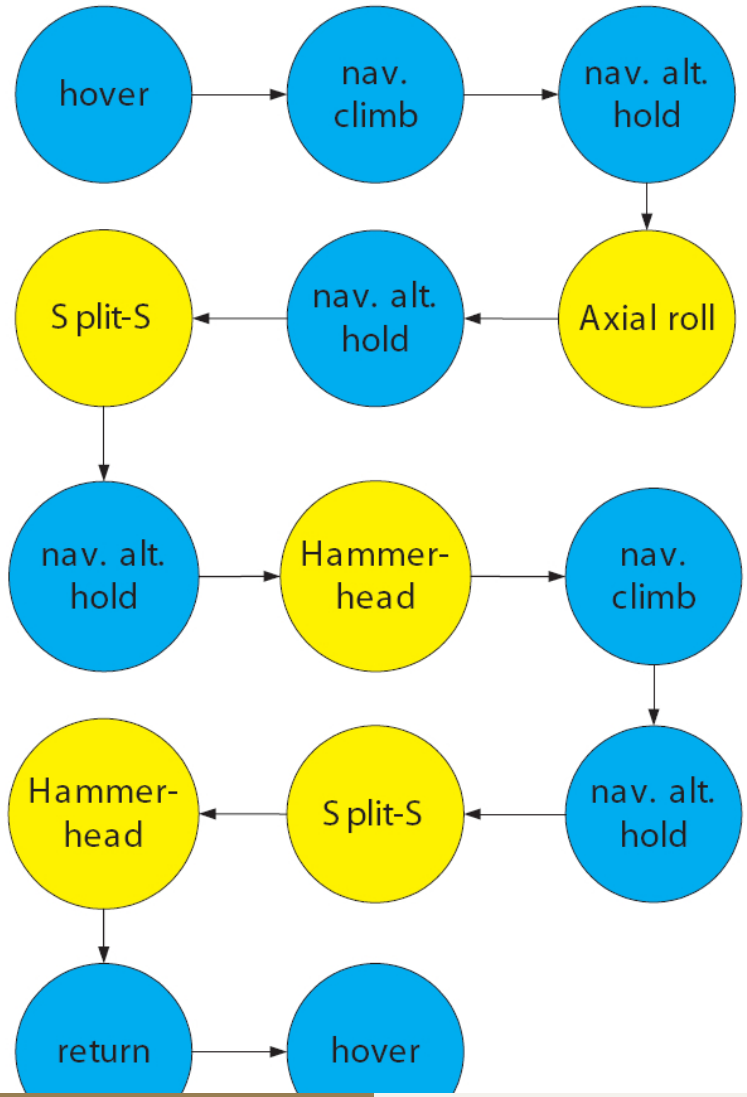
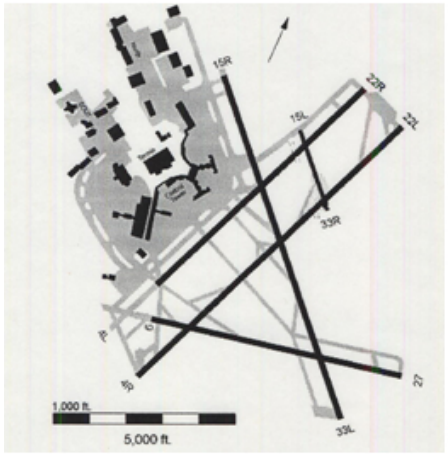


Figure: Roll Maneuver

Maneuver sequence as automaton



Closed-loop control: Boston Logan airport



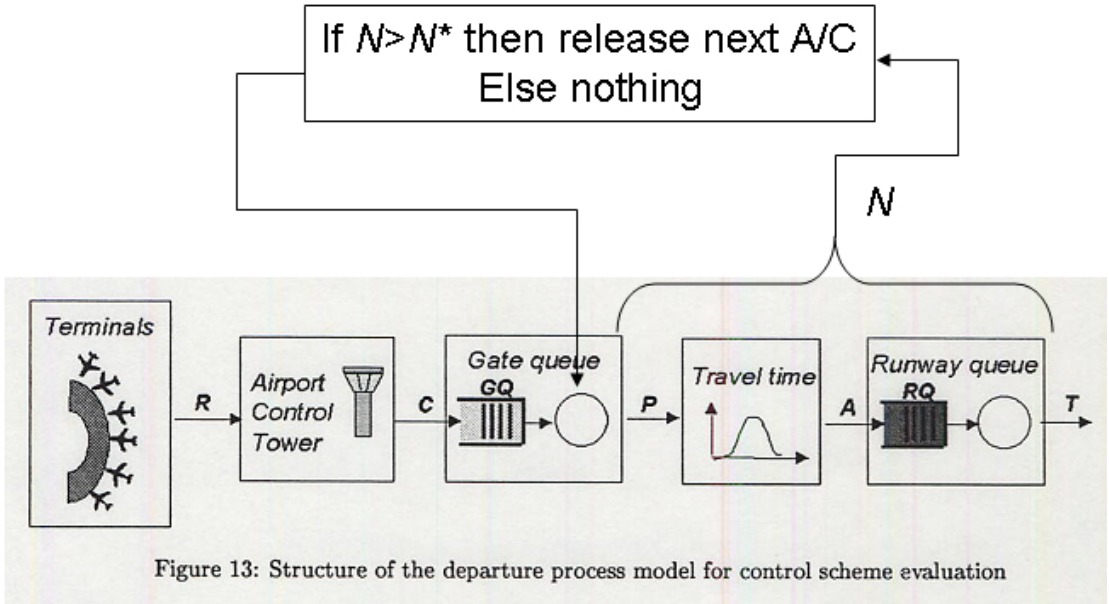


Figure 13: Structure of the departure process model for control scheme evaluation

Figure: Closed-loop Boston Logan congestion control

Implemented at BOS, JFK, SFO, MEM, CDG, MUN...

Georgia Institute of Technology