Barbaric Science from a Captive Poet: Oded Maler, *in memoriam*

Simulation-Based and Data-Driven Reasoning for CPS

April 18, 2019

Hybrid Systems: Computation and Control 2019

Jyo Deshmukh

USC Viterbi
School of Engineering
Department of Computer Science

Avoid Crashing, Falling, Burning, …

- We want to design systems that do all the things we envisioned them to do, while not doing stupid things that we did not envision.
- We want to develop methods to check whether bad things can happen:
  1) Formal techniques: where we build models, ***prove*** properties of the models
  2) Virtual testing techniques: where we build models, and ***test*** them extensively
  3) Real-world testing techniques: where we take the implementations in the ***real-world*** and test (as much as possible)
- From the purist/mathematical elitist view point, we mostly want to do (1)
- From the computer science/model-based perspective, we never want to do (3)
- (2) is the "***barbaric***" tradeoff that this talk is going to delve into …

On models and justification for barbarism

- Software: chain of semantics-preserving models (high-level PL to transistors)
- Not true in the physical world, where models are approximations

> *"This fact renders our early heroic CS efforts to prove decidability results on hybrid systems somewhat misguided, at least from an applicative point of view. In one of the early hybrid systems meetings I organized in Grenoble in the 90s, Paul Caspi presented a cartoon of a dialog between a control engineer, saying: it is trivial and a theoretical computer scientist responding: it is undecidable! But the noble activity of doing math for its own sake is common in all academic engineering domains, control included."*

[1] Oded Maler, Some Thoughts on Runtime Verification, (2016)

Real models are ugly!

$$
\begin{aligned}
\dot{p} &= c_1 \left(2\hat{u}_1 \sqrt{\frac{p}{c_{11}} - \left(\frac{p}{c_{11}}\right)^2} - (c_3 + c_4 c_2 p + c_5 c_2 p^2 + c_6 c_2^2 p) \right) \\
\dot{r} &= 4 \left(\frac{c_3 + c_4 c_2 p + c_5 c_2 p^2 + c_6 c_2^2 p}{c_{13}(c_3 + c_4 c_2 p_{est} + c_5 c_2 p_{est}^2 + c_6 c_2^2 p_{est})(1 + i + c_{14}(r - c_{16}))} - r \right) \\
\dot{p}_{est} &= c_1 \left(2\hat{u}_1 \sqrt{\frac{p}{c_{11}} - \left(\frac{p}{c_{11}}\right)^2} - c_{13} \left(c_3 + c_4 c_2 p_{est} + c_5 c_2 p_{est}^2 + c_6 c_2^2 p_{est} \right) \right) \\
\dot{i} &= c_{15}(r - c_{16})
\end{aligned}
$$

Very nonlinear

Formal verification out of reach

My (narrow) view into Oded's broad impacts

- Barbaric Reachability Analysis
  - A quest to identify techniques that work for general, real-world models
- Requirement-guided Testing/Falsification
  - A quest to impress the engineers with logics and magics
- Learning temporal abstractions from data
  - A quest to "civilize" machine learning

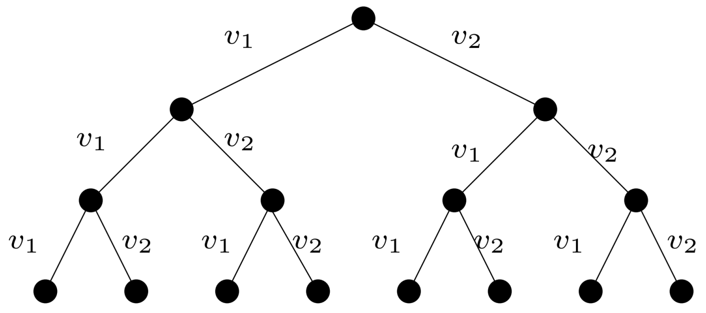
Barbaric Reachability

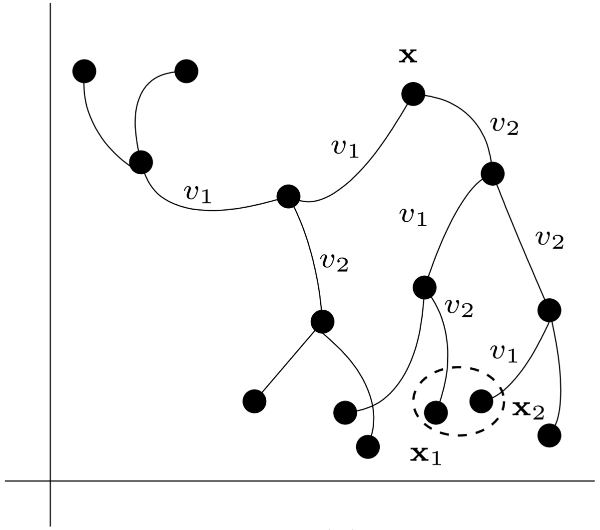
Simulation-guided Reachability Analysis

Postulate 1 (Simulation is Fine) *An intelligent mortal can solve the reachability problem for a well-behaved closed continuous system using a finite amount of numerical simulation.*

Main ideas:

1. Discretize input signal space
2. Express exploration of inputs as a tree
3. Merge nearby already explored regions

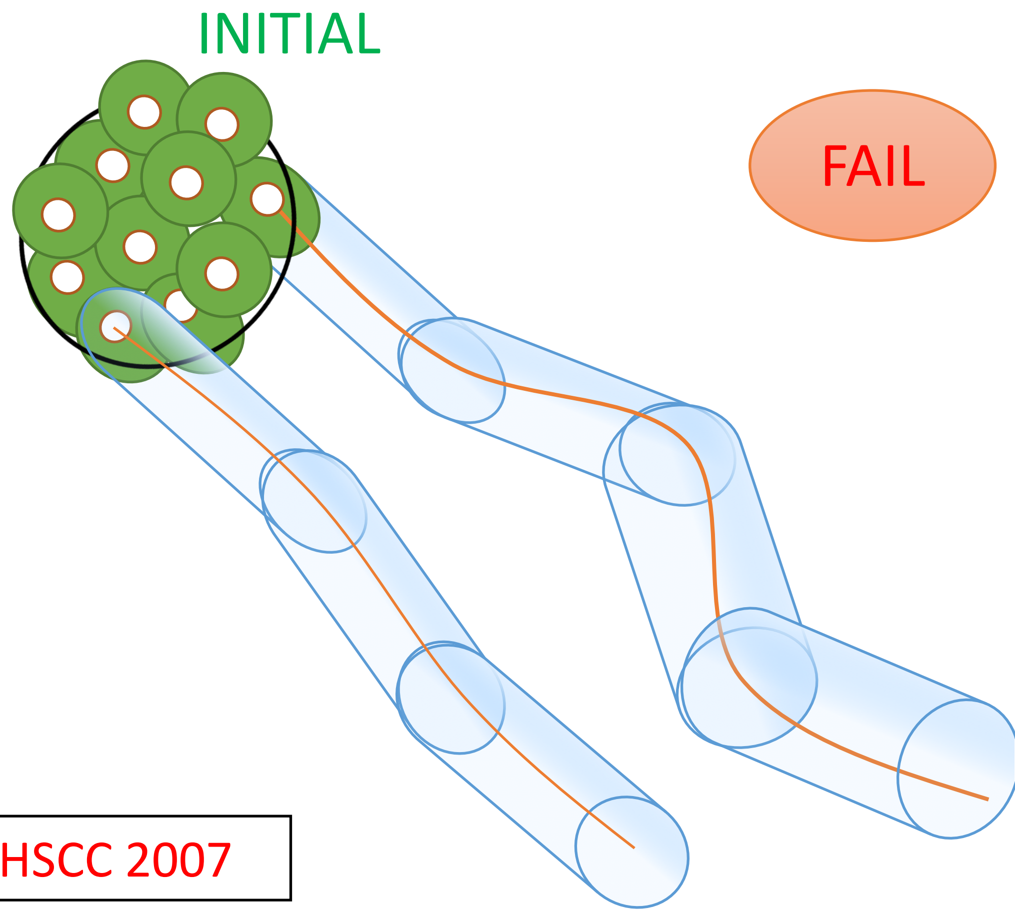




On Systematic Simulation of Open Continuous Systems, Jim Kapinski, Bruce H. Krogh, Oded Maler, and Olaf Stursberg, *HSCC, 2003*

Systematic Simulation with Sensitivity Analysis

- Birth of the Breach tool (Barbaric Reachability)
- Sample initial states in a way that *covers* the initial states set
- Simulate from each sampled initial state
- *Expand* simulation trajectories into tubes using the (numerically approximated) sensitivity of the system
- Gives one of 3 verdicts:
  - Safe: If union of tubes does not intersect fail set
  - Unsafe: If there is a concrete trajectory that lands in fail
  - Unknown: Otherwise (leads to refinement iterations)



A. Donzé, & O. Maler, Systematic simulation using sensitivity analysis. HSCC 2007

Merits of barbarism

- *"resolves the eternal tension between finite algorithmic termination and potential infinite precision of real numbers"*

- Practitioners already use simulation tools extensively (Simulink, LabView)
- Very scalable!
- Quantities like sensitivity can be readily obtained from numeric integrators used in simulation tools

Impacts of barbarism

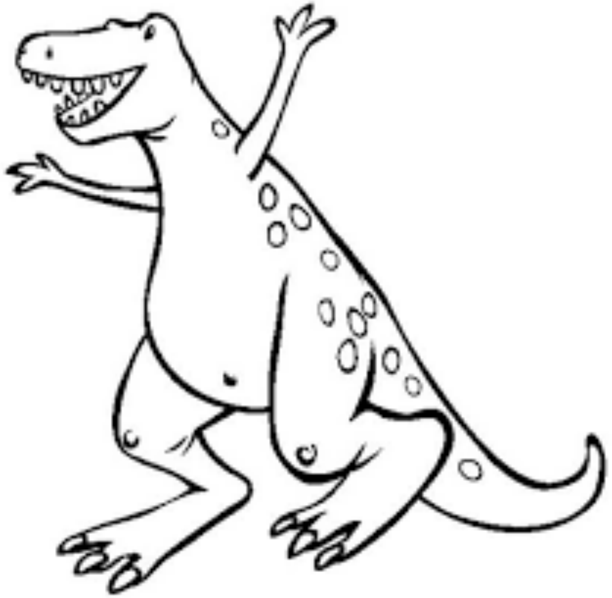
- Important contribution in the "verification by simulation" literature[1,2,3]
- Recent work on C2E2[4], DryVr[5], take this idea further with the notion of discrepancy functions
  - Led to exciting results: safety of industrial closed-loop control models
- Inspired: simulation-guided Lyapunov analysis[6], contraction analysis[7], …

1. A. Bhatia, E. Frazzoli. *Incremental search methods for reachability analysis of continuous and hybrid systems.* HSCC 2004
2. A. Girard, and G. J. Pappas. *Verification using simulation*. HSCC 2006
3. M. Branicky, et al. *Sampling-based planning, control and verification of hybrid systems. IEE Proceedings*
4. P. S. Duggirala, S. Mitra, M. Viswanathan, & M. Potok, M. *C2E2: A verification tool for stateflow models*. TACAS 2015
5. B. Qi, C. Fan, M. Jiang, S. Mitra, *DryVR 2.0: A tool for verification and controller synthesis of black-box CPS.* HSCC 2018
6. J. Kapinski, J. V. Deshmukh, S. Sankaranarayanan, N. Arechiga, *Simulation-guided lyapunov analysis for hybrid dynamical systems*. HSCC 2014
7. A. Balkan, J. V. Deshmukh, J. Kapinski, P. Tabuada, *Simulation-guided Contraction Analysis. ICC 2015.*

Requirement-Guided Testing/Falsification

Not so long time ago in the automotive world …

- Wild world where no one wrote safety requirements!
- Software design decisions were taken based on engineering experience
- Word documents in English, German, Japanese, Korean were used to define safety
- Test cases were written by hand, and relied on engineer insight
- Then one day, rode in STL with shining armor and the sword of formal methods!



Ad hoc safety practices

[1] G. E. Fainekos, S. Sankaranarayanan, K. Ueda, H. Yazarel, Verification of automotive control applications using S-Taliro, ACC '12
[2] J. Kapinski *et al*, *ST-Lib: a library for specifying and classifying model behaviors,* SAE Technical Paper, 2016
[3] H. Roehm, R. Gmehlich, T. Heinz, J. Oehlerking, M. Woehrle, Industrial Examples of Formal Specifications for Test Case Generation. In *ARCH@ CPSWeek*

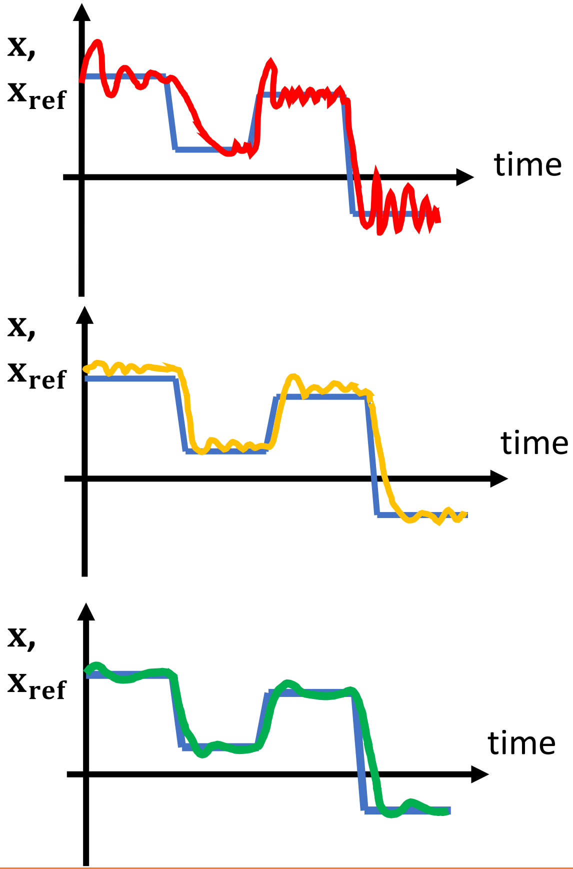
How it all happened: in the days before STL

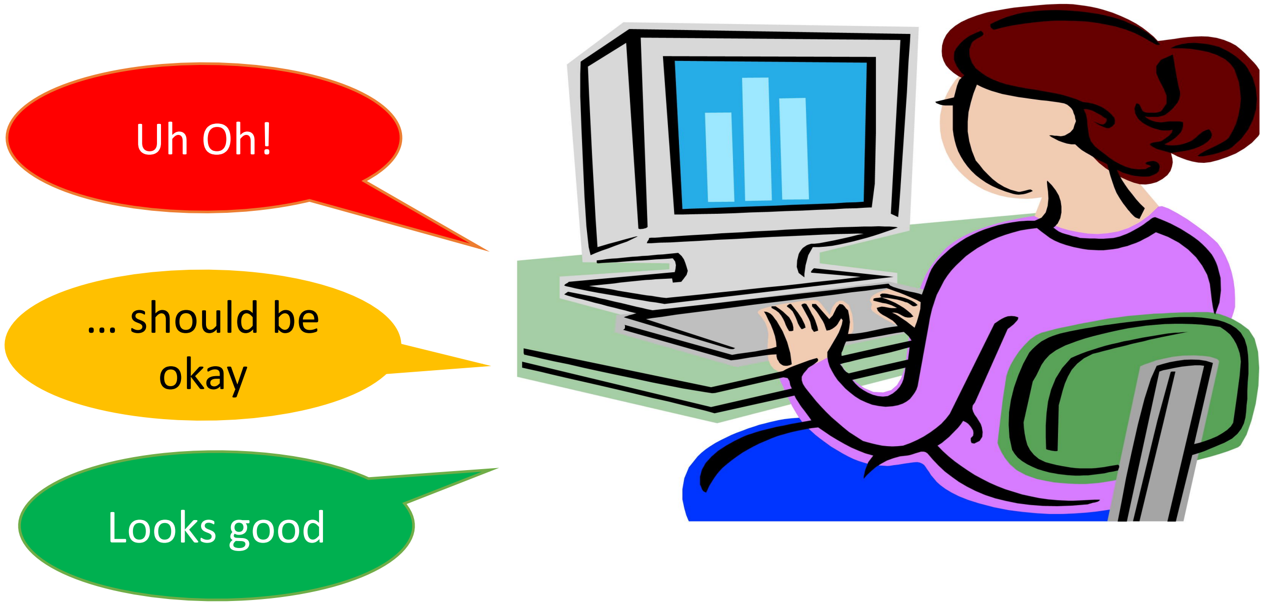
Check transient response of x
when driving with highway 73
pattern with temperature
below 15°C

Chief Engineer

Control Designer

Correctness was uh-oh, should be okay, looks good!





Can we formalize "*Uh-oh, should be okay, looks good, weird, clearly wrong, fuzzy?*"

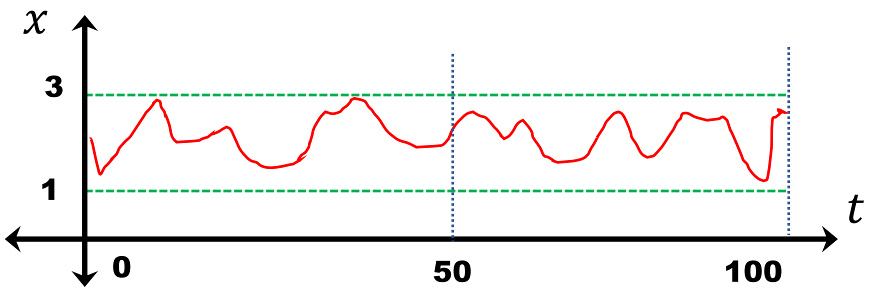
[1] O. Maler, and Dejan Nickovic. *Monitoring temporal properties of continuous signals*. *FORMATS*, 2004.
[2] A. Donzé, and O. Maler. *Robust satisfaction of temporal logic over real-valued signals*. FORMATS 2010.

Our observation: Yes, using Signal Temporal Logic[1,2]

What do STL specifications look like?

$\mathbf{Always}_{[0,100]}\ (1 \leq x(t) \leq 3)$

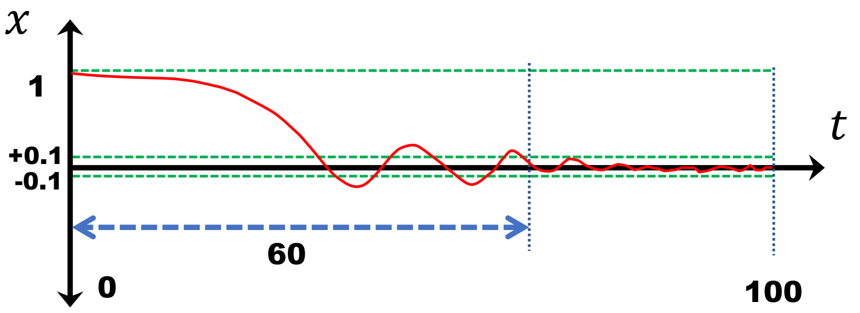
Always between time 0 and 100



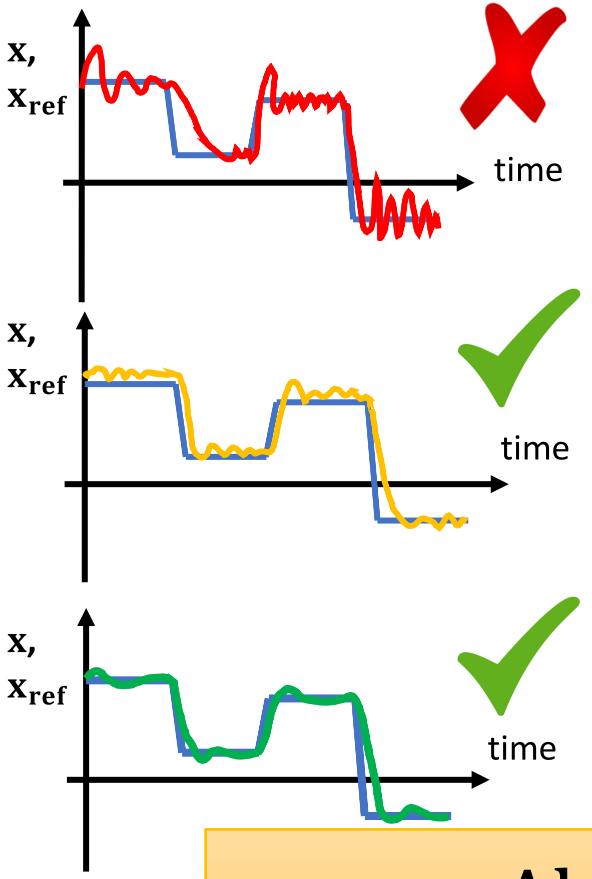
$\mathbf{Eventually}_{[20,60]}(\mathbf{Always}\ (|x(t)| < 0.1)\ )$

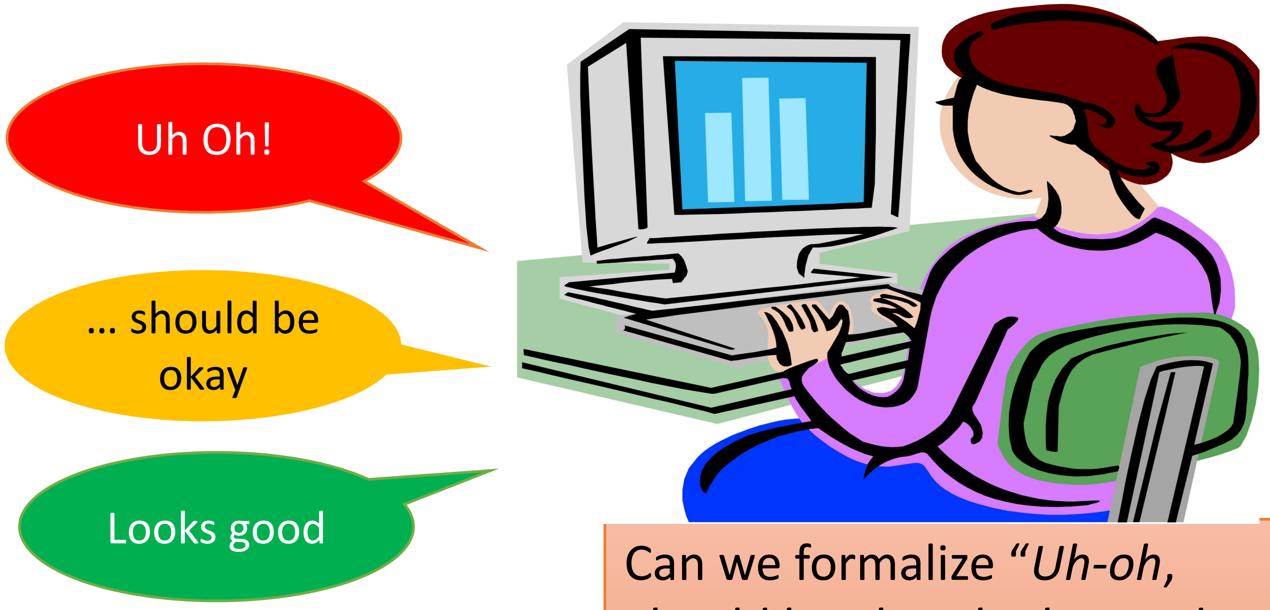
Eventually at **some time** ***t*** between time 20 and 60

From that time ***t*****,** always till the end of the signal trace



Correctness was now an STL formula



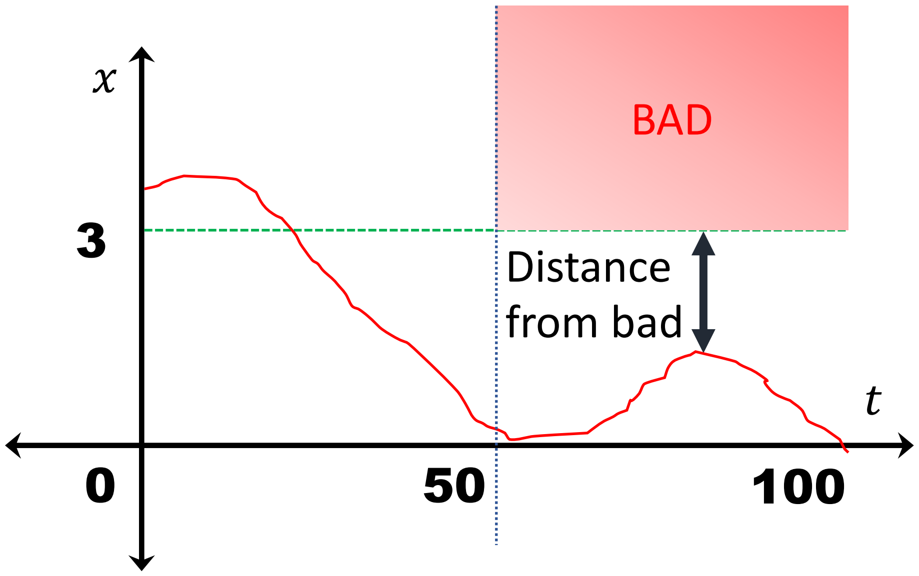


Can we formalize "*Uh-oh, should be okay, looks good, weird, clearly wrong, fuzzy?*"

$$\varphi \equiv \mathrm{Alw}_{[0,10]}\big(\mathrm{step} \Rightarrow \mathrm{Alw}_{[0,2]}\big(|x - x_{ref}| < 0.05x_{ref}\big)\big)$$

Beyond Boolean satisfaction: STL speaks numbers

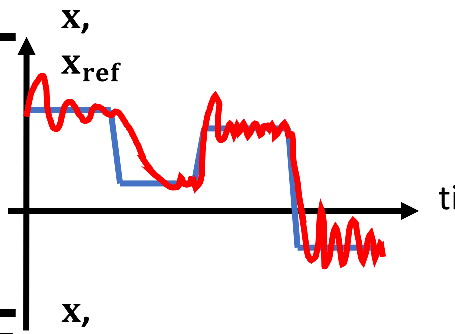
- Aka *Robust Satisfaction Value*, or *Robustness*
- Robustness[1,2]: function that
  - for a given trace $x(t)$,
  - and formula φ,
  - maps φ, $x(t)$ to some real value for each time t



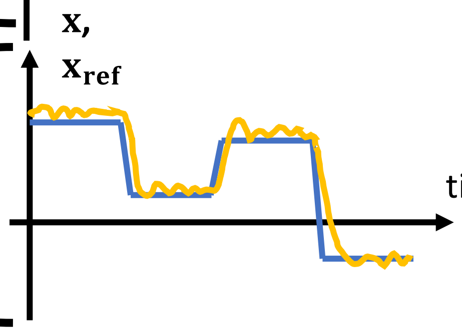
- Intuition:
  - Compute "signed distance" of the given trace x to the set of all traces satisfying φ
  - Distance ≥ 0 : $x \in$ set of traces satisfying φ
  - Distance < 0 : $x \notin$ set of traces satisfying φ
  - Going from positive to negative = going towards violation of φ

[1] G. Fainekos, and G. J. Pappas. *Robustness of temporal logic specifications for continuous-time signals*. TCS 2009.
[2] A. Donzé, and O. Maler. *Robust satisfaction of temporal logic over real-valued signals*. FORMATS 2010

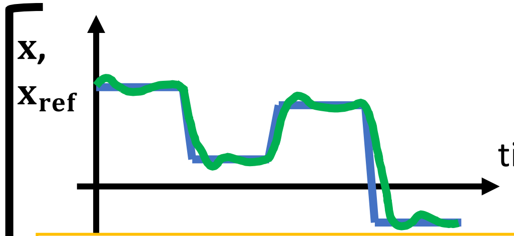
Robustness quantifies degree of satisfaction

f_φ 

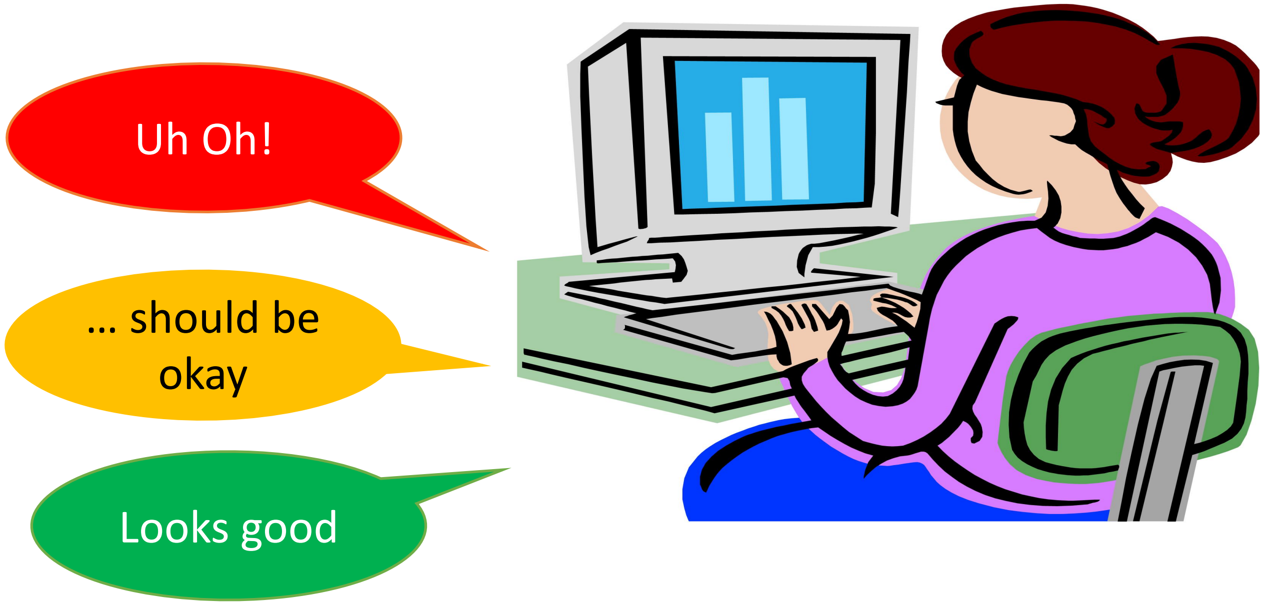
$= -0.2$

f_φ 

$= 0.01$

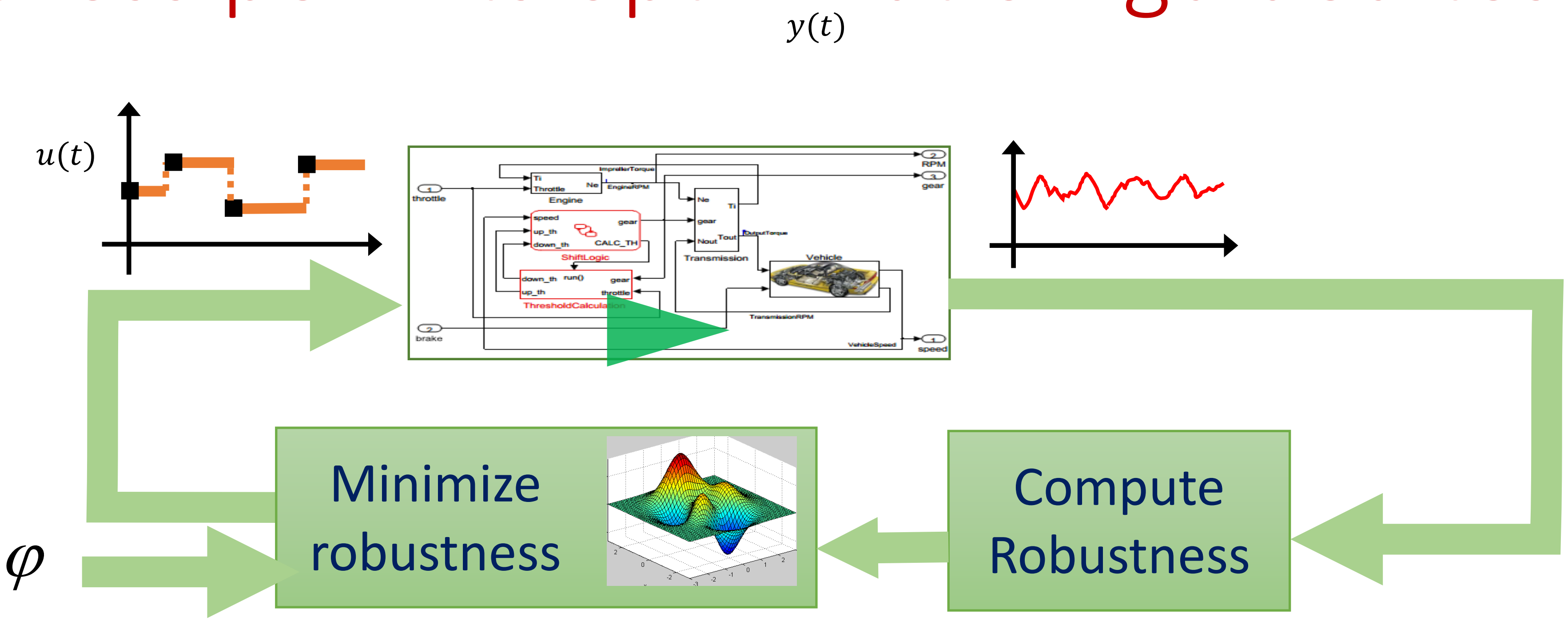
f_φ 

$= 0.2$



$$\varphi \equiv \mathrm{Alw}_{[0,10]}\left(\mathrm{step} \Rightarrow \mathrm{Alw}_{[0,2]}\left(\left|x - x_{ref}\right| < 0.05 x_{ref}\right)\right)$$

Robustness permits optimization-guided testing



1. S-TaLiRo [Fainekos, Sankaranarayanan, et al., TACAS '11, HSCC '10, ACC '12]: Cross Entropy, Simulated Annealing, Genetic Algorithms, Ant Colony
2. Breach [Donzé et al., CAV '10, NSV '13]: Derivative-free Nelder-Mead, Evolutionary algorithms

Collaborating with Oded was a unique experience

Oded: Jyo, why don't you do local search?

Jyo, Jim, Xiaoqing: Oded, this has already been done.

Oded: You should try even simpler (i.e. barbaric) local search

Jyo: But it is simpler and more trivial than what people have tried

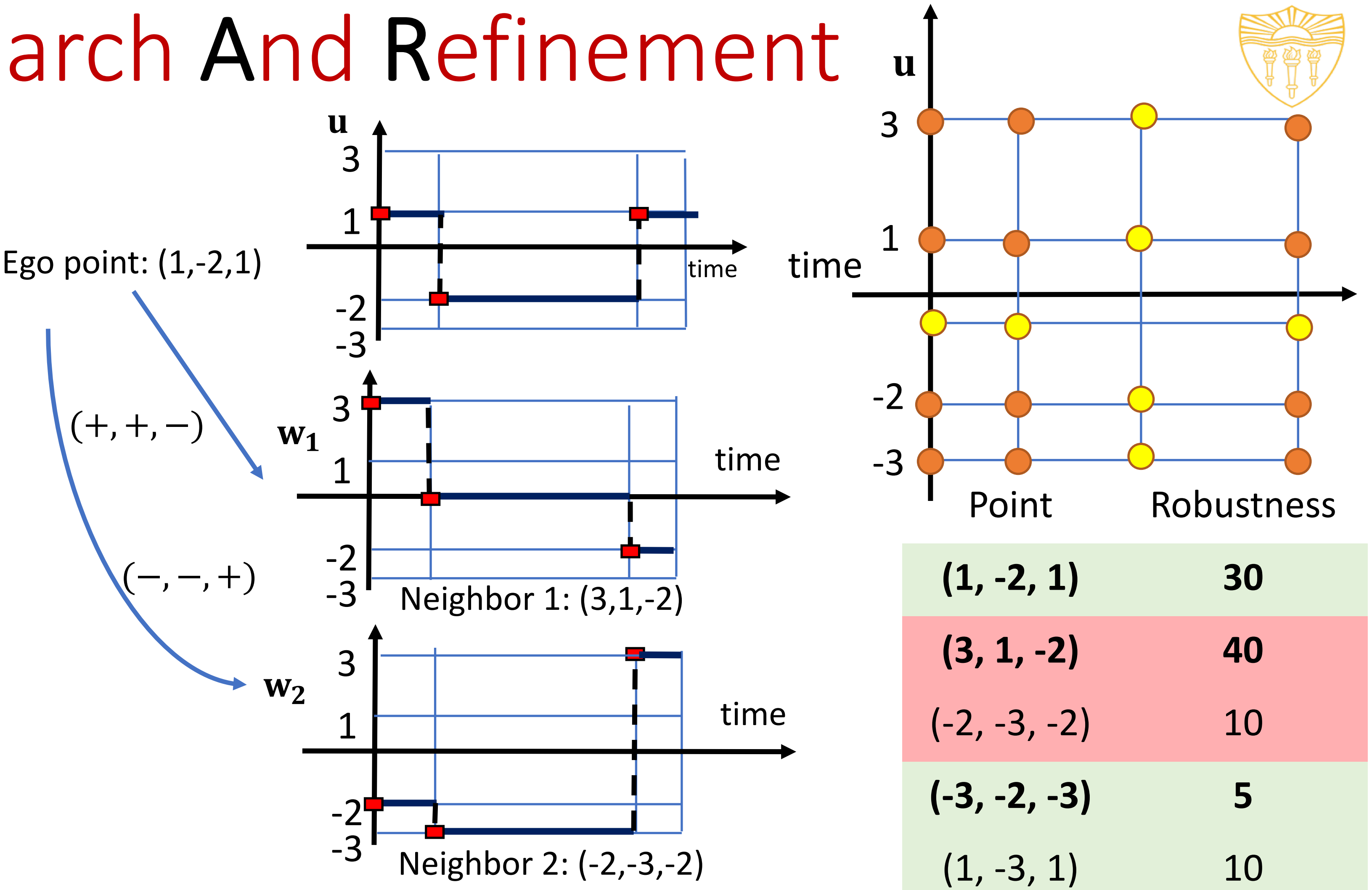
Oded: Maybe, it will work better!

Let us also remark that giving preference to results proved with respect to the most general existing definitions is a mathematicians attitude that should not be adopted without a critical examination. Such an attitude can be counter-productive in young domains where "classical" results and definitions are only decade old, and the most appropriate formalization has not yet stabilized.

From: O. Maler, D. Nickovic, and A. Pnueli. *From MITL to timed automata,* FORMATS 2006.

StochastIc Tabu search And Refinement

- Make search space finite
- *Stochastically* estimate least cost neighbor and descend
- Tabu-list to avoid revisiting
- Randomness to escape local optima
- Refine search space in promising regions



Point	Robustness
(1, -2, 1)	**30**
(3, 1, -2)	**40**
(-2, -3, -2)	10
(-3, -2, -3)	**5**
(1, -3, 1)	10

Tabu List

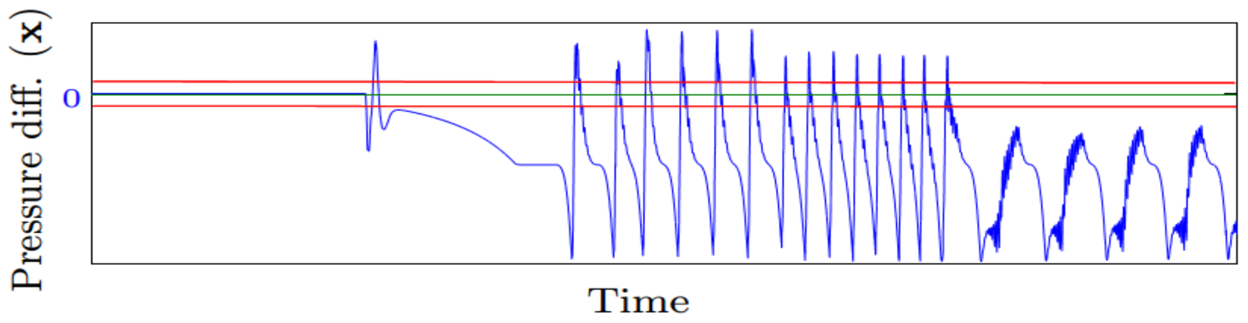
Falsification helps Toyota control designers

SITAR helps MIRAI control designer

- Model of Controller regulating air-flow through fuel-cell stack
- More than 7,000 Simulink blocks
- 5x slower than real-time to simulate
- Found violations of Overshoot on air-flow rate

Helps find rare bug in prototype Diesel Engine controller

- About 4000 Simulink blocks
- Successfully mined worst overshoot in 7 hours
- Found "worst-case" behavior using a combination of **Breach and S-TaLiRo**



Civilizing Machine Learning

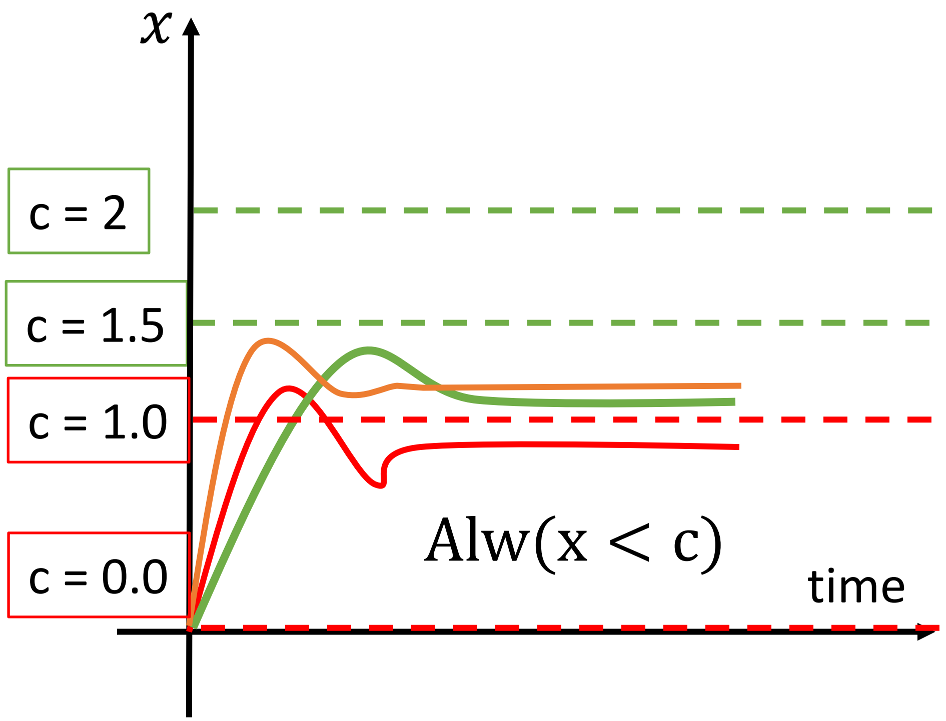
A brave new data-driven world

- Oded-trivia: His Ph.D. thesis was about learning!
  - L*-like active learning algorithm for ω-regular languages
- Machine learning is pretty barbaric. Maybe too barbaric!
- Can we learn interpretable/explainable/understandable artifacts from data?
- Can we civilize it by using formal interpretable artifacts like STL?[1,2,3,4,5,6]

[1] G. Bombara,C. I. Vasile, F. Penedo, H. Yasuoka, C. Belta. A decision tree approach to data classification using STL, HSCC 2016.
[2] Z. Kong, A. Jones, A. M. Ayala, E. A. Gol, and C. Belta. *TL inference for classification and prediction from data.* HSCC 2014
[3] E. Bartocci, L. Bortolussi, and G. Sanguinetti. *Data-driven statistical learning of temporal logic properties* FORMATS 2014.
[4] L. Nenzi, S. Silvetti, E. Bartocci, L. Bortolussi, *A robust genetic algorithm for learning temporal specs from data,* QEST 2018
[5] S. Jha, A. Tiwari, S. A. Seshia, T. Sahai, N. Shankar, Telex: Passive STL learning using only positive examples. RV 2017
[6] X. Jin, A. Donzé, J.V. Deshmukh, S. A. Seshia, *Mining requirements from closed-loop control models.* HSCC 2013, TCAD 2015
[7] B. Hoxha, A. Dokhanchi, and G. Fainekos.. *Mining parametric TL properties in MBD for CPS,* STTT 2018

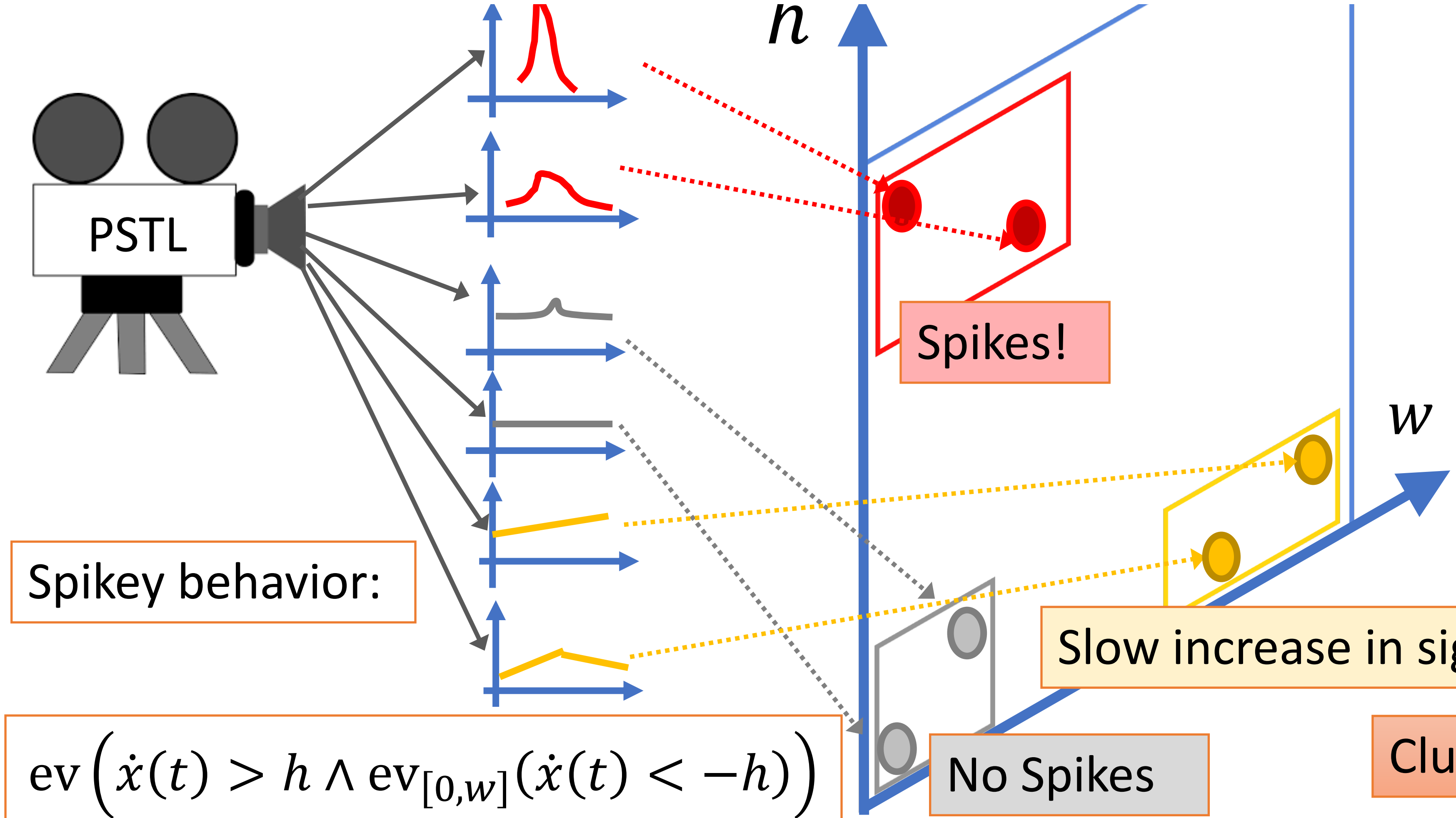
Inferring Parameter Values for PSTL from data

- Given:
  - PSTL formula $\varphi(\mathbf{p})$, $[\mathbf{p} = (p_1, p_2, \dots, p_m)]$
  - Traces $x_1, \dots, x_n$
- Find:
  - Valuation $\nu(\mathbf{p})$ such that: $\forall i : x_i \models \varphi(\nu(\mathbf{p}))$
  - And $\exists i{:}x_i \not\models \varphi(\nu(\mathbf{p}) \pm \delta)$: (small perturbation in $\nu(\mathbf{p})$ makes some trace not satisfy φ)
- Polarity fragment of PSTL (monotonicity in parameters) allows using binary search



E. Asarin, A. Donzé, O. Maler, D. Nickovic, *Parametric identification of temporal properties*. RV 2011

Logical clustering of time-series data



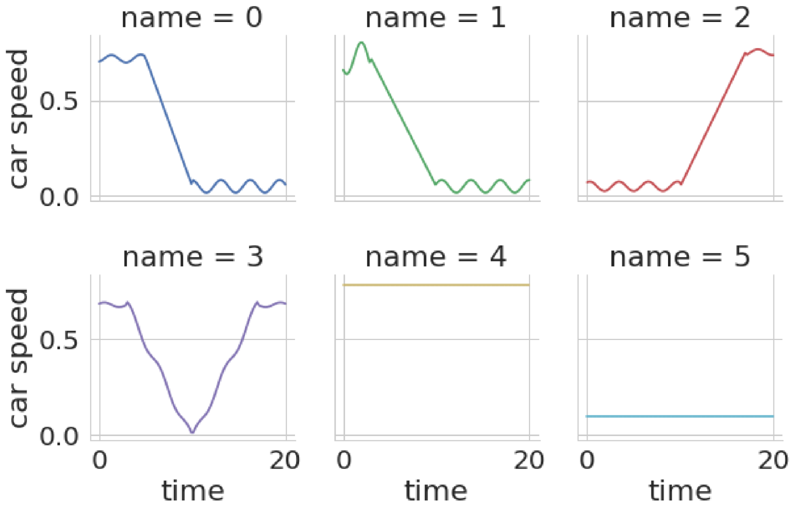
$$\text{ev}\left(\dot{x}(t) > h \wedge \text{ev}_{[0,w]}(\dot{x}(t) < -h)\right)$$

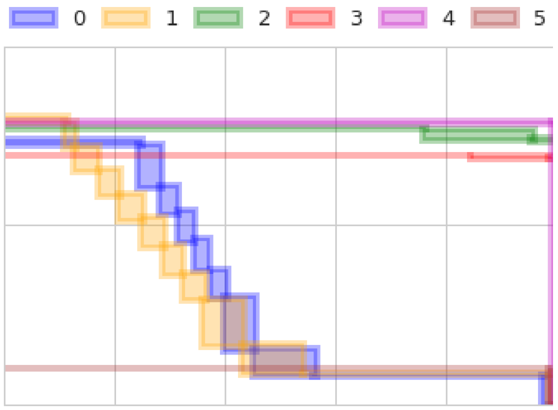
- Use PSTL for feature extraction
- Project each trace to δ-tight valuation for trace of given formula φ
- Cluster δ-tight valuations
- Each hyper-box cluster is an STL formula

Clusters have logical meaning!

Projecting to validity boundaries

- Multi-parameter PSTL formulas have infinitely many δ-tight satisfying valuations; picking one requires *ad hoc* choice
- Alternative approach: consider the entire validity domain boundary, and use that to cluster[1]; but how to compute efficiently?
- Oded's recent work: multi-criteria optimization for monotone functions
- Focus for his unfinished Latexotherapy exercise: how do you do supervised, unsupervised, semi-supervised learning of STL formulas from data?





[1] M. Vazquez-Chanlatte, S. Ghosh, J. V. Deshmukh, A. Sangiovanni-Vincentelli , S. A. Seshia, *Time-Series Learning Using Monotonic Logical Properties*, RV 2018

Summary and personal reflections

- Do not be afraid to reinvent the wheel
- Beauty and Elegance in everything are worthy pursuits, and they often come from Simplicity
- There does not need to be a compromise between “very interesting math with little practical application” and “very barbaric methods with many practical applications”:
  - Oded was a person who did both, and was unapologetic about either
  - His self-awareness about *why* he worked on a particular problem was rare, and refreshing

Thanks to collaborators

- Jim Kapinski
- Xiaoqing Jin
- Tommaso Dreossi
- Thao Dang
- Alexandre Donzé
- Marcell Vazquez-Chanlatte
- Isaac Ito
- Sanjit Seshia