Verification and Synthesis Algorithms for Safe Autonomy

Chuchu Fan
Autonomy: moonshot goals and safety challenges

Tesla To Make Fully Autonomous Cars By 2016?

Tesla's effort to build an autonomous car could reach fruition by 2016 according to a new Reuters report today. The question isn't so much whether the technology is feasible, but whether the legal and economic market is ready.

Tesla's competition with Google on autonomous car technology has previously resulted in back-peddaling from CEO and iconoclast Elon Musk, however, dismissing his talk about self-driving cars as off-the-cuff remarks, rather than an organized plan.

Uber's Self-Driving Car Detected Pedestrian Before Fatal Crash, But "Decided" Not to Stop

When a self-driving car is involved in a fatal accident, who is at fault? Is it the "safety-driver," the company behind the car, or the lawmakers that allowed.

Today's SpaceX explosion is a major setback for Facebook's free internet ambitions

A $95 million deal is suddenly in danger

By Russell Brandon | Sep 1, 2016, 12:01pm EDT
Sensing and Perception
camera, LIDAR, GPS, computer vision, machine learning, neural networks, data

Decision and Control
navigation, path planning, physics, code

Act
computer, engine, steering, brake
Video from Prof. Shashua at World Knowledge Forum’17: Platform for Safe and Scalable AVs
How to assure an autonomous system works correctly
How to assure an autonomous system works correctly

With perfect information, a design engineer can simulate forward to check safety.

But there are sources of uncertainty:

- Estimation error in position/speed of other cars
- Models of other cars and drivers

In principle, no finite number of simulations or tests can cover all possible corner cases of the system and give absolute correctness assurances.
How to assure an autonomous system works correctly

In general, many other sources of uncertainty
- Road friction
- Failures, e.g., flat tire
- Distracted drivers

“In practice, 30 billion miles of test driving is needed to achieve acceptable levels of assurance!” [Koopman, CMU] [Shashua, CTO Mobileye]

We need methods to provide coverage guarantees across scenarios

We need principles to smartly direct testing and simulation resources to quickly find any incorrect behavior or “bugs”
Formal verification can provide coverage guarantees

System model $A$
Requirement (property) $R$

Formal verification

**yes**: all behaviors of $A$ satisfy $R$
Formal verification can provide coverage guarantees

System model $A$
Requirement (property) $R$

Formal verification

yes: all behaviors of $A$ satisfy $R$

no: find a behavior of $A$ (“bug”) that violates $R$
My research: design and verification for safe autonomous systems

Theory & algorithms

Algorithms with formal guarantees for verification and control synthesis for hybrid systems

[CAV14,17][ATVA15][EMSOFT16][NAHS17][FM18][TECS18][CAV18]

Tools

C2E2

[CAV16][TACAS15]

DryR

[HSCC18]

RealSyn

Z3 Yices CVC4

Applications

[IEEE DT18] [CAV15] [CDC17] [CAV18] [IEEE DT16]
Dubins’ car model

\[
\begin{align*}
\dot{v} &= a \\
\dot{s}_x &= v \cos(\psi) \\
\dot{s}_y &= v \sin(\psi) \\
\dot{\delta} &= v \delta \\
\dot{\psi} &= \frac{v}{l} \tan(\delta)
\end{align*}
\]

Speed
Horizontal position
Vertical position
Steering angle
Heading angle

Nonlinear dynamics

Generally, nonlinear ODEs do not have closed-form solutions!

Physical plant

\[
\begin{align*}
\dot{x} &= f(x, u) \\
x[t + 1] &= f(x[t], u[t]) \\
x &= [v, s_x, s_y, \delta, \psi] \\
u &= [a, v \delta]
\end{align*}
\]

System dynamics
State variables
Control inputs
Autonomous systems verification challenge

Nonlinear *hybrid* dynamics

Physical plant

\[ \dot{x} = f(x, u) \]
\[ x = [v, s_x, s_y, \delta, \psi] \]
\[ u = [a, v_\delta] \]

System dynamics

State variables

Control inputs

Decision and control software

- **speed up**
- **merge left**
  - close to front car
  - cruise
- **merge right**
  - speeding
  - slow down

Chuchu Fan · UIUC
Autonomous systems verification challenge

Nonlinear hybrid dynamics

Physical plant

\[ \dot{x} = f(x, u) \] System dynamics

\[ x = [v, s_x, s_y, \delta, \psi] \] State variables

\[ u = [a, v_\delta] \] Control inputs

Decision and control software

- Speed up
- Merge left
- Merge right
- Cruise
- Slow down
- Speeding
- Abort

Close to front car
Autonomous systems verification challenge

Nonlinear hybrid dynamics
Interaction between computation and physics can lead to unexpected behaviors

Physical plant
\[ \dot{x} = f_{\text{mode}}(x, u) \]

Software
- merge left
- merge right
- cruise
- speed up
- slow down

Hybrid system model
- speed up: \[ \dot{x} = f_{\text{up}}(x, u) \]
- merge left: \[ \dot{x} = f_{\text{left}}(x, u) \]
- merge right: \[ \dot{x} = f_{\text{right}}(x, u) \]
- cruise: \[ \dot{x} = f_{\text{crs}}(x, u) \]
- slow down: \[ \dot{x} = f_{\text{down}}(x, u) \]
Interaction between computation and physics can lead to unexpected behaviors

The engine of the car is stable in each of the modes.

Is it possible to switch between them to make the system unstable?
Interaction between computation and physics can lead to unexpected behaviors

Yes! By switching between them the system becomes unstable
We can model Autonomous systems as hybrid systems

But verifying hybrid systems is very hard!
<table>
<thead>
<tr>
<th>Outline</th>
<th>Verification</th>
<th>Synthesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory &amp; algorithms</td>
<td>Nonlinear hybrid systems [EMSOFT16][TECS18]</td>
<td>Scalable control synthesis for large dimensional linear systems [CAV18]</td>
</tr>
</tbody>
</table>

### Tools
- C2E2 [CAV16][TACAS15]
- DryR [HSCC18]
- RealSyn [CAV18]

### Applications
- [IEEE DT18]
- [CAV15]
- [IEEE E]
- [CAV18]
The bounded safety verification problem

Consider an autonomous system model $\dot{x} = f(x)$

Trajectory $x(t)$: state at $t$ from $x(0)$
The bounded safety verification problem

Consider an autonomous system model $\dot{x} = f(x)$

Trajectory $x(t)$: state at $t$ from $x(0)$

Given start set $S$, unsafe set $U$, time bound $T$, 
The bounded safety verification problem

Consider an autonomous system model $\dot{x} = f(x)$

Trajectory $x(t)$: state at $t$ from $x(0)$

Given start set $S$, unsafe set $U$, time bound $T$, let $Reach(S,T)$ be the set of reachable states from $S$ up to $T$

Bounded safety verification problem:

$Reach(S,T) \cap U = empty$ ?

Computing the reach set is hard!
Brief history of hybrid system verification

90’s: modeling frameworks and early theoretical results on computing unbounded time exact reach set
- Hybrid I/O Automata framework [Lynch 96]; A unified framework for hybrid control [Mitter 98]
- Decidable for timed automata [Alur and Dill 92]
- Undecidable even for constant dynamics [Henzinger 95]

Late 90’- Now: approximate bounded time reachability for linear hybrid systems
- Polytopes [Henzinger 97], ellipsoids [Kurzhanski 97], zonotopes [Girard 05], support functions [Frehse 08]
- Predicate abstraction [Alur 03], CEGAR [Clarke 03] [Prabhakar 13, 15], Generalized star [Bak 17]

This talk: Data-driven verification for nonlinear and black-box systems
- Use simulation data to algorithmically construct formal proofs
- Generalize single simulation using rigorous sensitivity analysis [Fan 15-18]
Basic idea: Generalize single simulation using rigorous sensitivity analysis

Let's say that the distance of the trajectories can be bounded in terms of the distance of the initial states.

This is called sensitivity of $x(t)$ to $x(0)$.

Discrepancy formalizes sensitivity.
Basic idea: Generalize single simulation using rigorous sensitivity analysis

Discrepancy formalizes sensitivity

**Definition:** Discrepancy is a continuous function $\beta$ that

1. Bounds the distance between neighboring trajectories

$$\|x_1(t) - x_2(t)\| \leq \beta(\|x_1(0) - x_2(0)\|, t)$$
Basic idea: Generalize single simulation using rigorous sensitivity analysis

Discrepancy formalizes sensitivity:

**Definition: Discrepancy** is a continuous function $\beta$ that

1. Bounds the distance between neighboring trajectories
   \[ \|x_1(t) - x_2(t)\| \leq \beta(\|x_1(0) - x_2(0)\|, t) \]

2. Error can be driven down:
   \[ \beta(\|x_1(0) - x_2(0)\|, t) \to 0 \text{ as } \|x_1(0) - x_2(0)\| \to 0 \]
A simple example of a discrepancy function

If $f(x)$ has a Lipschitz constant $L$:

$$\forall x, y \in \mathbb{R}^n, \|f(x) - f(y)\| \leq L\|x - y\|$$

then a discrepancy function is:

$$\|x_1(t) - x_2(t)\| \leq \|x_1(0) - x_2(0)\| e^{Lt} \beta(\|x_1(0) - x_2(0)\|, t)$$
A simple example of a bad discrepancy function

If $f(x)$ has a Lipschitz constant $L$:

$$\forall x, y \in \mathbb{R}^n, \|f(x) - f(y)\| \leq L\|x - y\|$$

then a **bad** discrepancy function is:

$$\|x_1(t) - x_2(t)\| \leq \|x_1(0) - x_2(0)\|e^{Lt}$$

$$\beta(\|x_1(0) - x_2(0)\|, t)$$

Related approaches: Lyapunov functions [Lyapunov 1892], Lyapunov exponents [Arnold 86], contraction metric [Slotine 96], etc.
What is a good discrepancy?

**General:** Applies to general nonlinear $f$

**Accurate:** Smaller error in $\beta$ => Fewer samples for verification

**Effective:** Computing $\beta$ is fast (in practice)
“Locally optimal reach set over-approximation for nonlinear hybrid systems.” *EMSOFT* 2016. [Best paper finalist]

“Simulation-Driven Reachability Using Matrix Measures.” *TECS* 2018. [Invited paper]
To find the sensitivity of $x(t)$ to $x(0)$

We have to see how $f(x(t))$ changes to $f(x(t) + \epsilon)$

This is $\frac{\partial f(x)}{\partial x}$, the **Jacobian matrix** $J(x)$ of $f$

We can prove $\frac{d}{dt}(x_2(t) - x_1(t)) =$

$$\left( \int_0^1 J((1-s)x_1 + sx_2) \, ds \right)(x_2(t) - x_1(t))$$

For $\dot{x} = Ax$, **matrix measure** $\mu(A) \equiv \lim_{t \to 0^+} \frac{||tA|| - 1}{t}$ is the growth rate of $\log ||x||$

So $\mu(J(\cdot))$ gives the growth rate of $\log ||x_1(t) - x_2(t)||$, which gives the theorem:
Matrix measure of Jacobian can be used to compute discrepancy

**Theorem [EMSOFT16]:** Over a compact set $\mathcal{D}$:

$$
\|x_1(t) - x_2(t)\| \leq \|x_1(0) - x_2(0)\|e^{ct},
$$

where $c = \max_{x \in \mathcal{D}} \mu(j(x))$

$L = \max_{x \in \mathcal{D}} \|J(x)\|$ is a Lipschitz constant

**Property:**

$$
\begin{aligned}
\mu(J(x)) &\leq \|J(x)\| \quad \text{(In practice } c \ll L!) \\

\end{aligned}
$$

But $\mu(J(x))$ has no closed-form solutions!

We bound $J(x)$ over $\mathcal{D}$ via interval matrices and estimate $\mu(J(x))$ **efficiently**, while **preserving the properties (1) and (2) of discrepancy**
Theorem [EMSOFT16]: Over a compact set \( \mathcal{D} \):

\[
\|x_1(t) - x_2(t)\| \leq \|x_1(0) - x_2(0)\|e^{ct},
\]

where \( c = \max_{x \in \mathcal{D}} \mu(j(x)) \)

How should we choose the optimal norm \( \| \cdot \| \) and why that matters?
How should we choose the optimal norm $\| \cdot \|$ and why that matters?

**2-norm** bounds $\text{Reach}(S, T)$ along all directions uniformly (*too conservative*)

Instead, **with coordinate transformation $P$** (ellipsoids) we can bound $\text{Reach}(S, T)$ more accurately

**Theorem [EMSOFT16]:** we can find the **best** ellipsoidal bound for $\text{Reach}(S, T)$ by solving a semi-definite program (SDP);

Solution gives the **smallest** $c$ for the discrepancy $\|P(x_1(0) - x_2(0))\|_2 e^{ct}$
How should we choose the optimal norm $\| \cdot \|$ and why that matters?

Overall locally optimal reachability algorithm
How should we choose the optimal norm $\| \cdot \|$ and why that matters?

First locally optimal reachability analysis for nonlinear hybrid system verification

This minimizes the number of simulations or samples needed for verification

Asymptotic convergence: Approximation error goes to 0 as time goes to infinity, for contractive systems.
Data-driven verification using discrepancy function

Bounded safety verification problem:

\[ \text{Reach}(S,T) \cap U = \text{empty} ? \]
Data-driven verification using discrepancy function

Bounded safety verification problem:

\[ \text{Reach}(S, T) \cap U = \text{empty} \ ? \]

1. Simulate from the center
Data-driven verification using discrepancy function

Bounded safety verification problem:

\[ \text{Reach}(S, T) \cap U = \text{empty} \, ? \]

1. Simulate from the center

2. Generalize simulation using discrepancy

3. Check safety
Data-driven verification using discrepancy function

Bounded safety verification problem:

\[ \text{Reach}(S,T) \cap U = \text{empty} \] ?

1. Simulate from the center
2. Generalize simulation using discrepancy
3. Check safety
4. Refine to make over-approximations more accurate

\( \beta \) computed using matrix measure, intervals matrices, and optimal ellipsoids, etc., preserve the 2 properties
Data-driven verification using discrepancy function

Bounded safety verification problem:

\[ \text{Reach}(S,T) \cap U = \text{empty} \]?

1. Simulate from the center
2. Generalize simulation using discrepancy
3. Check safety
4. Refine to make over-approximations more accurate
5. **Repeat** until find an counter-example or all covers are safe
Theoretical guarantees of data-driven verification

**Soundness:** If the data-driven verification algorithm returns \textit{safe}, then the system is \textit{indeed safe}. If the algorithm returns \textit{unsafe}, then the counter-example is \textit{indeed unsafe}.

**Relative completeness:** If the system is robustly safe or robustly unsafe, then the data-driven verification algorithm will \textit{terminate} with correct answer.
Theoretical guarantees of data-driven verification

Soundness: If the data-driven verification algorithm returns \textit{safe}, then the system is indeed safe. If the algorithm returns \textit{unsafe}, then the counter-example is indeed unsafe.

Relative completeness: If the system is robustly safe or robustly unsafe, then the data-driven verification algorithm will terminate with correct answer.

Simulation data + sensitivity (discrepancy) from models $\Rightarrow$ automatic safety proofs!

“Meeting a powertrain verification challenge.” CAV 2015.
Verify the overtake scenario in C2E2

Verify scenario with uncertainties like speeds in [90, 100] km/h and overtake distance in [20, 30] m

Safety requirement: separation of 1 m maintained always

Chuchu Fan · UIUC
C2E2 generated safety proof

CarSim simulator connected to C2E2
C2E2 finds bug: Blue accelerates as white returns to right lane

CarSim simulator connected to C2E2
C2E2 application: powertrain control

A suite of powertrain verification benchmarks presented in HSCC 2014 by Toyota [Jin14]

5 dimensional hybrid system with polynomial differential equations

\[ \dot{\theta} = 10(\theta_{in} - \theta) \]
\[ \dot{p} = c_1(2\theta(c_{20}p^2 + c_{21}p + c_{22}) - c_{12}(c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega p^2)) \]
\[ \dot{\lambda} = c_{26}(c_{15} + c_{16}c_{25}F_c + c_{17}c_{25}F_c^2 + c_{18}\dot{m}_c + c_{19}\dot{m}_c c_{25}F_c - \lambda) \]
\[ p_e = c_1(2c_{23}\theta(c_{20}p^2 + c_{21}p + c_{22}) - (c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega p^2)) \]
\[ i = c_{14}(c_{24}\lambda - c_{11}) \]
C2E2 application: powertrain control

A suite of powertrain verification benchmarks presented in HSCC 2014 by Toyota [Jin14]

5 dimensional hybrid system with polynomial differential equations

Verified the model with a set of driver behavior profiles (running time approx. 12 mins) [CAV15]

- First verification of Toyota powertrain model
- Best verification result award at CPSWeek’15
“DryVR: Data-driven verification and compositional reasoning for automotive systems.” CAV 2017.

Challenge: incomplete model

Nonlinear hybrid dynamics

Incomplete model

Hybrid model

- Speed up
- Merge left
  - Close to front car
  - Cruise
  - Abort
- Merge right
  - Speeding
  - Slow down

Mode cruise
“All models are wrong, some are useful”

Gain serenity, analyze real-world models

https://github.com/cfan10/DryVR
DryVR: A new model semantics for hybrid systems

Complete information of switching structure

Executable access to mode dynamics

Hybrid system

Control graph

Black-box simulator

DryVR’s Executable hybrid model
Global exponential discrepancy

\[ \|x_1(t) - x_2(t)\| \leq \|x_1(0) - x_2(0)\|e^{at+b} \]

Taking log:

\[ \forall t, \ln \frac{\|x_1(t) - x_2(t)\|}{\|x_1(0) - x_2(0)\|} \leq at + b \]

By setting the initial values \(x_1(0), x_2(0)\), the black-box simulator can return sampled simulation trajectories \(x_1(t), x_2(t)\)

Find \(a\) and \(b\) by learning a linear separator

Similar idea works for other formats of the discrepancy function

Also works for other machine learning algorithms
DryMesh: Learn discrepancy from data

Theorem [CAV17]: Given the training set, the global exponential discrepancy function that gives the tightest reach set over-approximation can be found by solving a Linear programming (LP) problem.

Proposition [CAV17]: \( \forall \epsilon, \delta > 0 \), if sampling number \( n \geq \frac{1}{\epsilon} \ln \frac{1}{\delta} \), then with probability \( 1 - \delta \), the algorithm finds \((a, b)\) such that \( err_D(\gamma, k) < \epsilon \). [PAC learning, Valiant 1984]

- For example: in theory, set \( \epsilon = 0.1\% \), \( \delta = 1\% \), \( n \approx 4600 \)

In practice, we show on >20 systems that it is enough to use \( n = 20 \) training trajectories.

Intuition: the algorithm learned the worst case global exponential sensitivity from the training trajectories.
DryR : Formal reasoning

Find a control graph for given requirements

Possible modes:
merge left, accelerate, merge right...

Simulation relation of control graph for behavior containment

\[
\text{Reach} \subseteq \text{Reach}
\]
DryVR can verify systems that combine white + black-box modules

by combining formal techniques with learned discrepancy
Verification of spacecraft rendezvous maneuver

Autonomous spacecraft rendezvous, proximity operations, and docking (ARPOD) problem developed by Air Force Research Lab as a challenge problem for verification and controller synthesis [Jewison & Erwin 2016]

Complex, hybrid dynamical system with uncertainties using third-party controllers [Chan 2017][Sanfelice 2017]

3 hour long missions verified in 10 minutes

Learning curve for an undergraduate student was about a month

C2E2 result with a LQR controller [Chan 2017]

DryVR result with a robust supervisor controller [Sanfelice 2017]
Verification of safe ADAS features

Risk analysis [IEEE DT17]
Case study: Automatic Emergency Braking
Determining Automotive Safety Integrity Levels (ASIL) mandated by the ISO26262 standard

ASIL risk = probability \times severity
Verification of safe ADAS features

Risk analysis [IEEE DT17]

Case study: Automatic Emergency Braking

Determining Automotive Safety Integrity Levels (ASIL) mandated by the ISO26262 standard

\[ \text{ASIL risk} = \text{probability} \times \text{severity} \]
Compositional verification of networked system

Input-to-state discrepancy function for $\dot{x} = f(x, u)$

$$\|x_1(t) - x_2(t)\| \leq \beta (\|x_1(0) - x_2(0)\|, t) + \int_0^t \gamma (\|u_1(s) - u_2(s)\|) ds$$

Theorem [CAV15]: $m(t)$ bounds the divergence of original system’s trajectories

Arbitrary topologies and over-approximation error goes to 0 with refinement

This enables us to verify $A$ (which can be a network with complex topologies) using simulations of $A$ and $B$ and only discrepancy computation of each component of $A$
Composition for networked autonomous system

IS-Discrepancy + our composition theorem => scalable automatic proofs for networked systems

What’s ahead?

\[ \frac{dy_2}{dt} = f_2(y_2, u_2) \]
\[ \text{dim}(y_2) = n_2 \]
\[ u_3 = y_1, y_2 \]

\[ \frac{dy_3}{dt} = f_3(y_3, u_3) \]
\[ \text{dim}(y_3) = n_3 \]
\[ m(t) \text{ goes to 0 as size of the start set goes to 0} \]

Arbitrary topologies
Verification for AS/CPS is an important but hard problem. Simulation + sensitivity (discrepancy) give proofs.

- First optimal reachability analysis for nonlinear hybrid dynamics.
- First hybrid model as white + black box with learned discrepancy.
- Compositional analysis with designed or learned components.

**Other works in verification**

- Theory & algorithms
- Tools
- Applications

**Compositional analysis**

- Pacemaker [IEEE DT15]

**Uncertain Inputs**

- Mixed-signal circuits [ADHS18]

**Applications**

- Parallel landing [Duggirala EMSOFT 17]
- Drones with RL [Sibai HSCC18]
Verification for AS/CPS is an important but hard problem. Simulation + sensitivity (discrepancy) give proofs. First optimal reachability analysis for nonlinear hybrid dynamics. First hybrid model as white + black box with learned discrepancy. Compositional analysis with designed or learned components.

First data-driven verification tool for nonlinear hybrid systems with guarantees. A significant step towards real-world verification.

Applications:
- Mode: Start up
- Distance between Car1 and Car2

Other works in verification:
- Pacemaker [IEEE DT15]
- Mixed-signal circuits [ADHS18]
- Parallel landing [Duggirala EMSOFT 17]
- Drones with RL [Sibai HSCC18]
Many thanks to my collaborators

Prof. Mitra’s group

Mahesh Viswanathan
UIUC

Marta Kwiatkowska
University of Oxford

Ulrich Schmid
TU Wien

Scott Smolka
Stony Brook University

Umang Mathur
UIUC

Xiaoying Jin
Apple

Jim Kapinski
Toyota