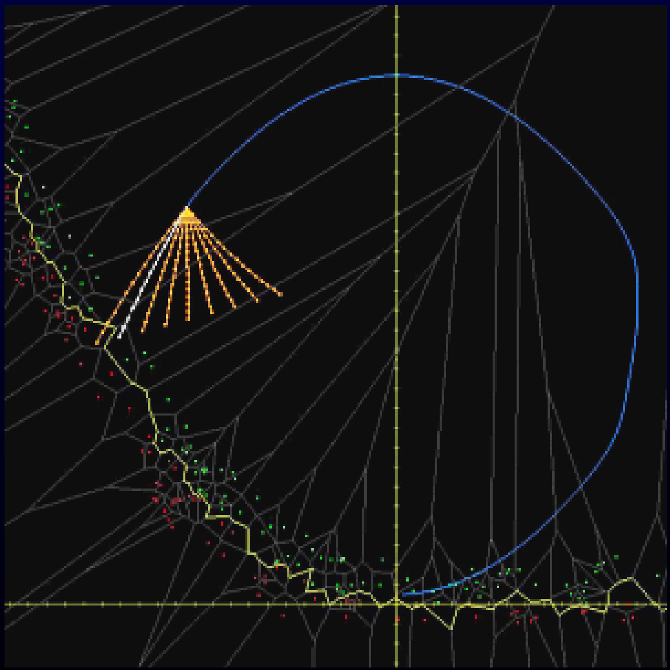
Approximate Safety Enforcement Using Computed Viability Envelopes

Maciej Kalisiak
<mac@dgp.toronto.edu>
University of Toronto

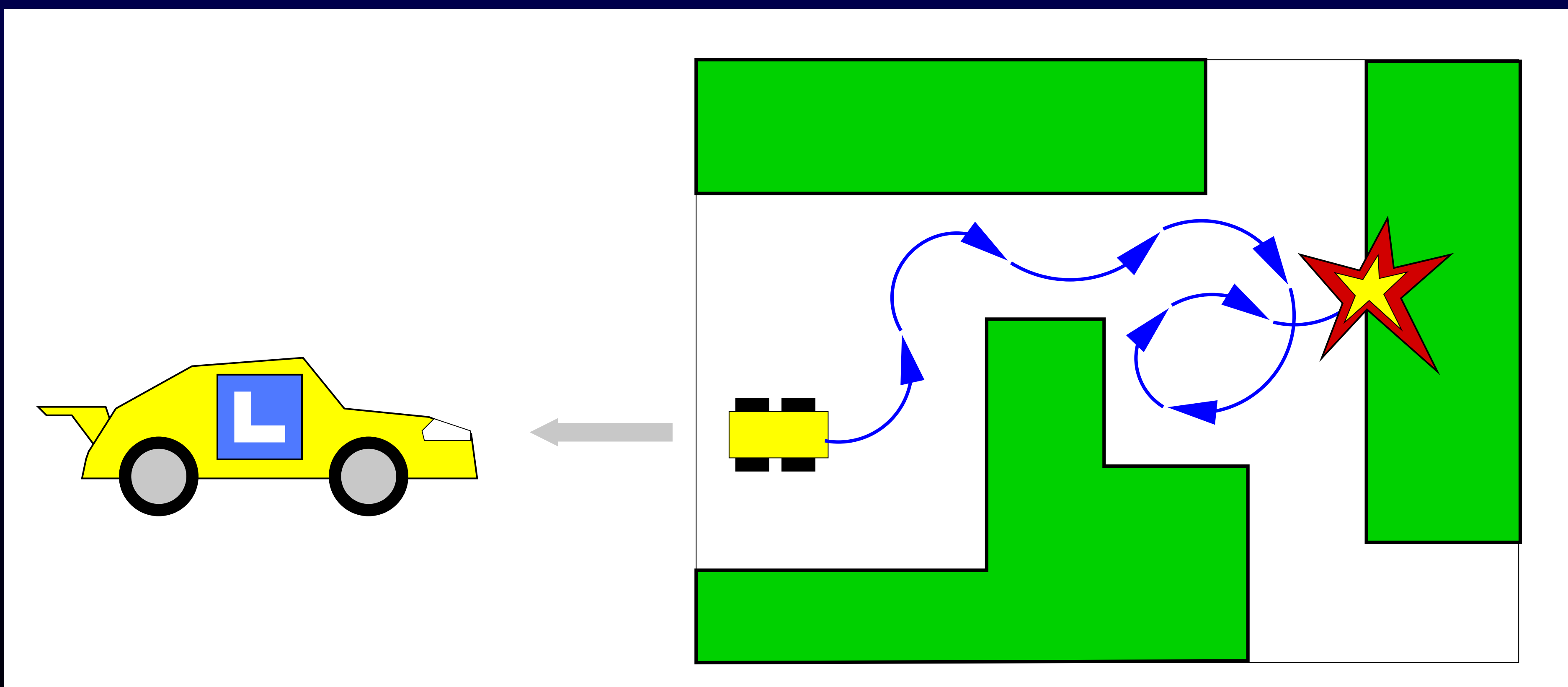
Michiel van de Panne
<van@cs.ubc.ca>
University of British Columbia



IEEE International Conference on Robotics and Automation 2004

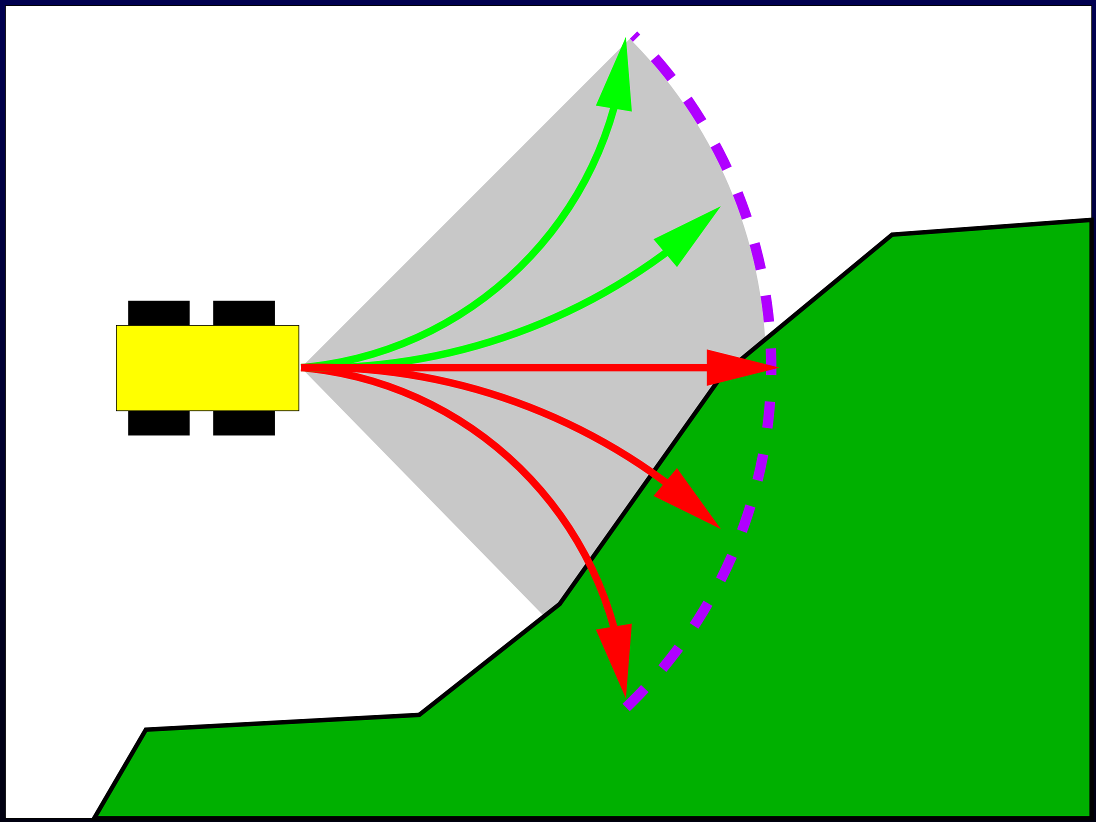
Problem & General Idea

- **problem**: user input can lead to failure
- **idea**: computer intervenes when necessary
- *[movie of desired result (4-obstacle example)]*



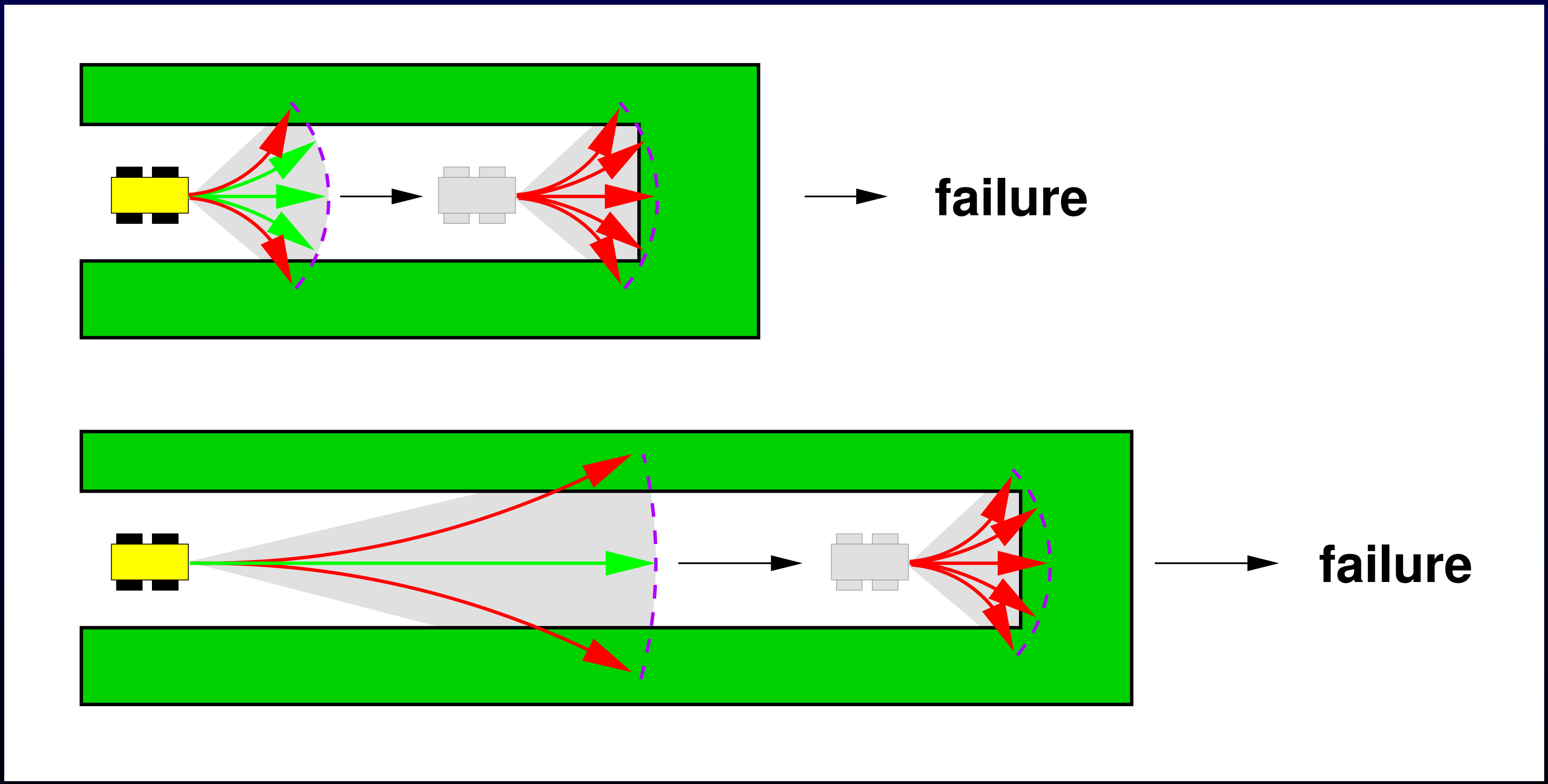
Naïve Implementation

- if user's input leads to failure within some given time horizon, override it with a failure-free input



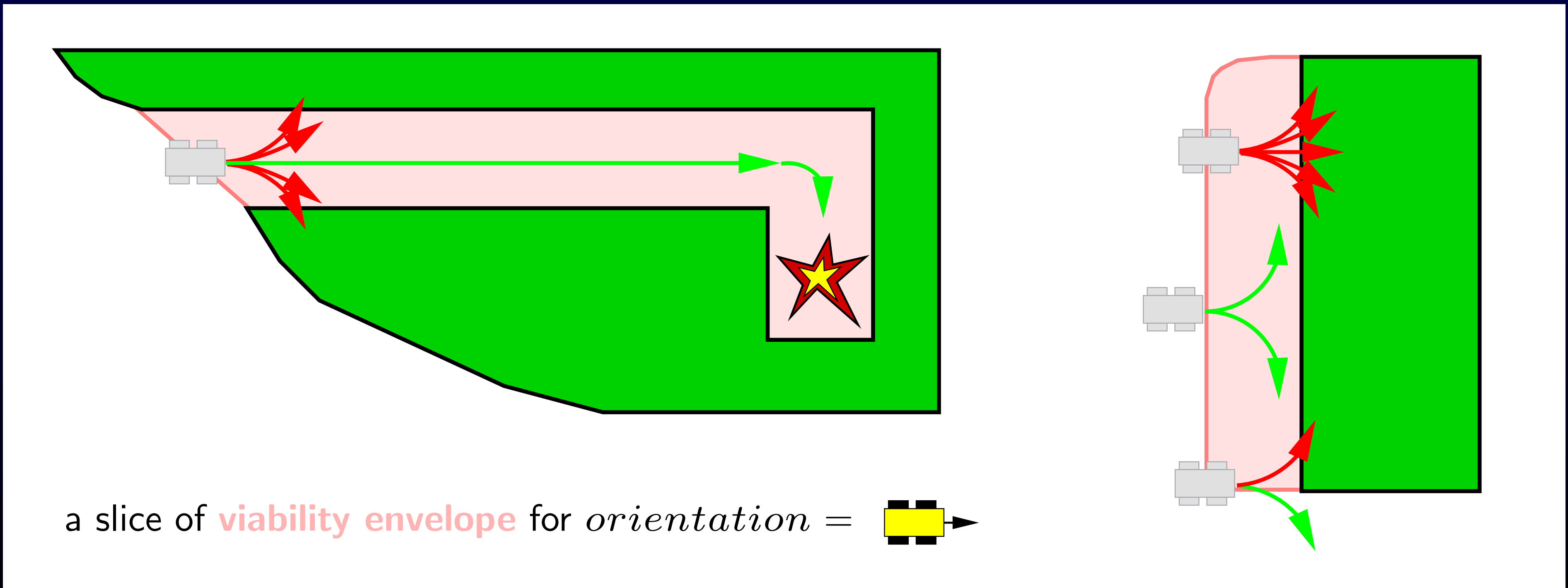
Naïve Implementation: Problem

- **problem**: one can get trapped in a "dead-end"
- *dead-end* > *time horizon* always possible



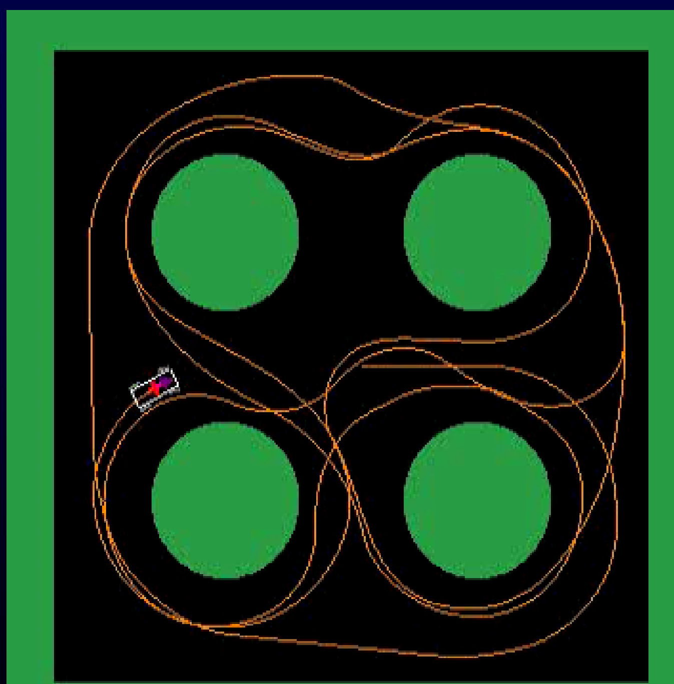
Viability Envelope

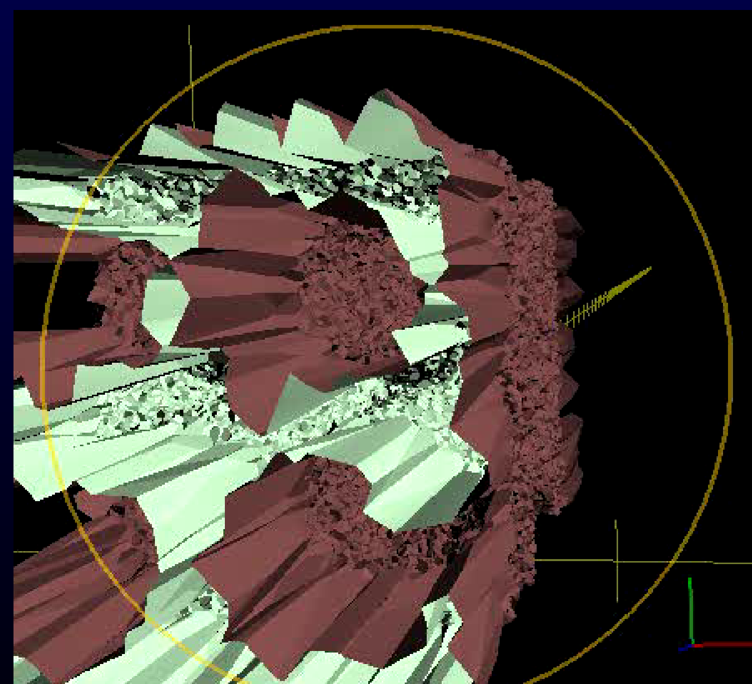
- **strategy**: mark all such "unavoidable failure" states as "out of bounds", then stay within bounds
- *viability envelope* = this bound
  = set of all "points of no return"



Viability Envelope (ctd.)

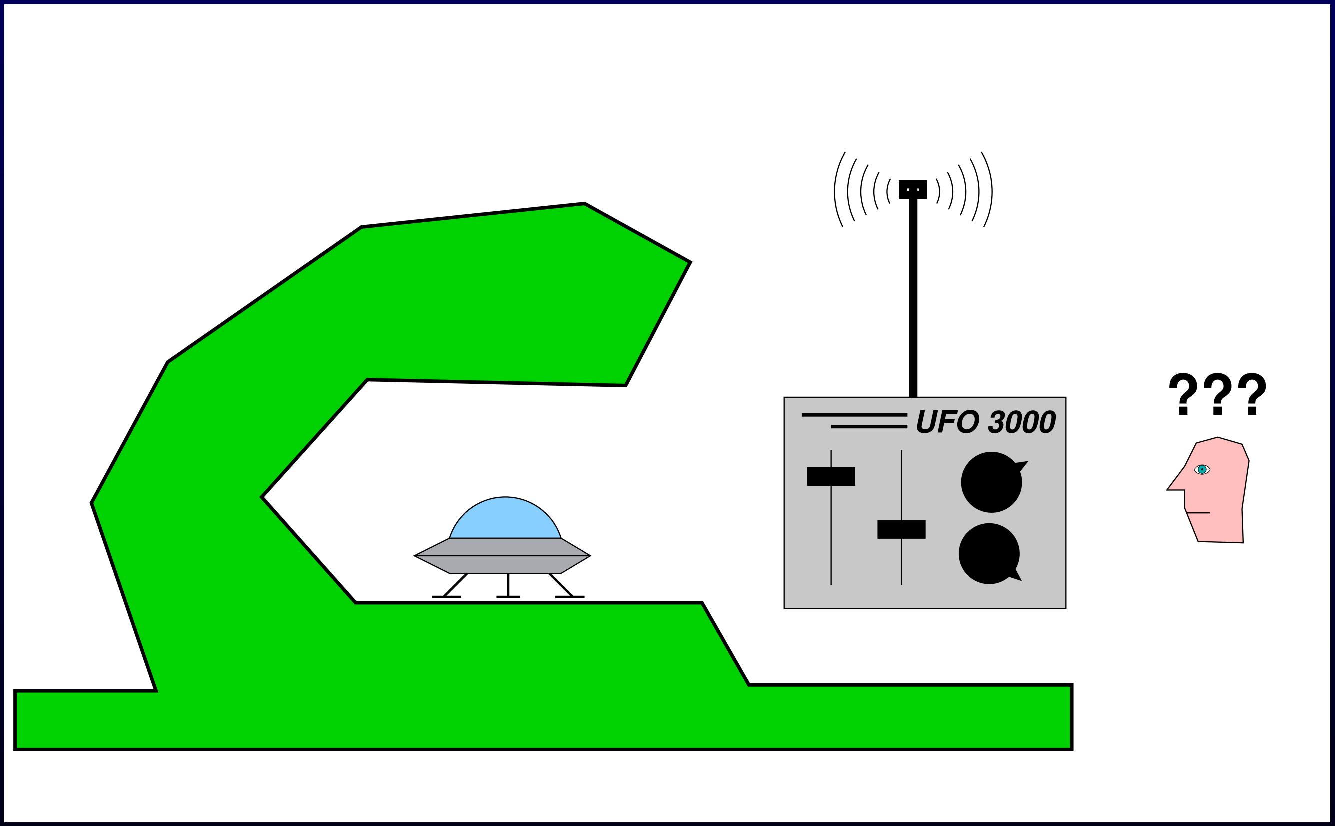
- the envelope is a manifold in the system's state-space
- for the simple car, state-space is 3D: $(x, y, orientation)$
- *[movie: 3D tumble of 4-obstacle envelope]*





Applicability

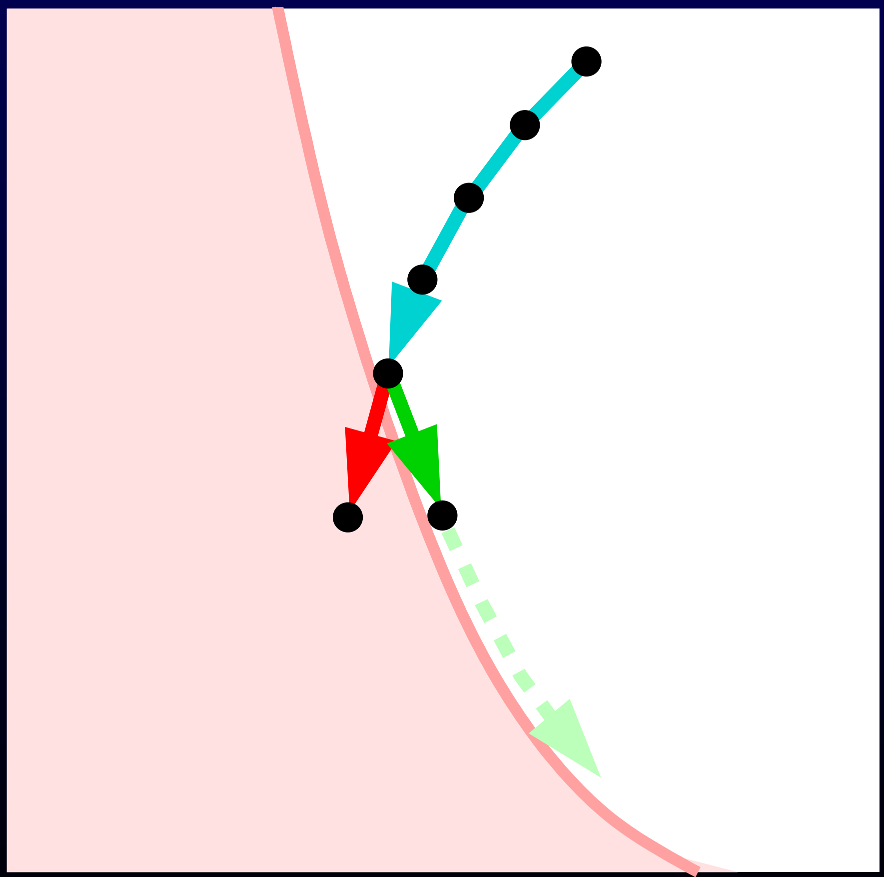
- applicable to any dynamical system with known dynamics



– Framework Details –

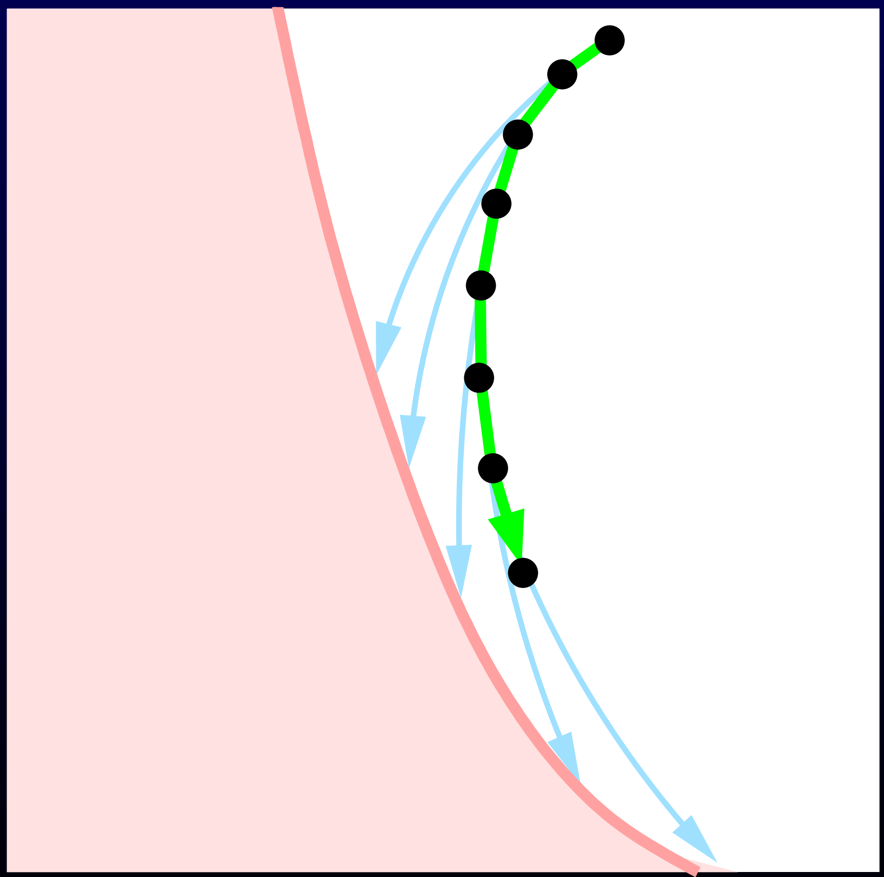
Single-step Containment

- correct the control input when about to cause a breach
- disadvantage: harsh and abrupt corrections



Multi-step Containment

- use predictive look-ahead, act on breaches earlier
- result: milder corrections



Time to Envelope Breach

- $T_{eb}(x, u)$: *"time to envelope breach"*
- how long until control input u causes breach from state x
- assumption: u is held constant

Time to Envelope Breach

- $T_{eb}(x, u)$: *"time to envelope breach"*
- how long until control input u causes breach from state x
- assumption: u is held constant
- very distant breaches irrelevant
- clamp T_{eb} at T_h, the *"time horizon"*
  (i.e., $T_{eb} \leq T_h$ or $T_{eb} = \infty$)

Time to Envelope Breach

- $T_{eb}(x, u)$: *"time to envelope breach"*
- how long until control input u causes breach from state x
- assumption: u is held constant
- very distant breaches irrelevant
- clamp T_{eb} at T_h, the *"time horizon"*
  (i.e., $T_{eb} \leq T_h$ or $T_{eb} = \infty$)
- "breach-free" implies "... within T_h"

System Meta-states and Control Policy

- four meta-states (think: "severity", "DEFCON"):
  - **L1**: user's control input is breach-free
  - **L2**: **L1** false, but a different input is breach-free
  - **L3**: **L2** false, but system still within envelope
  - **L4**: **L3** false (i.e., containment failed)

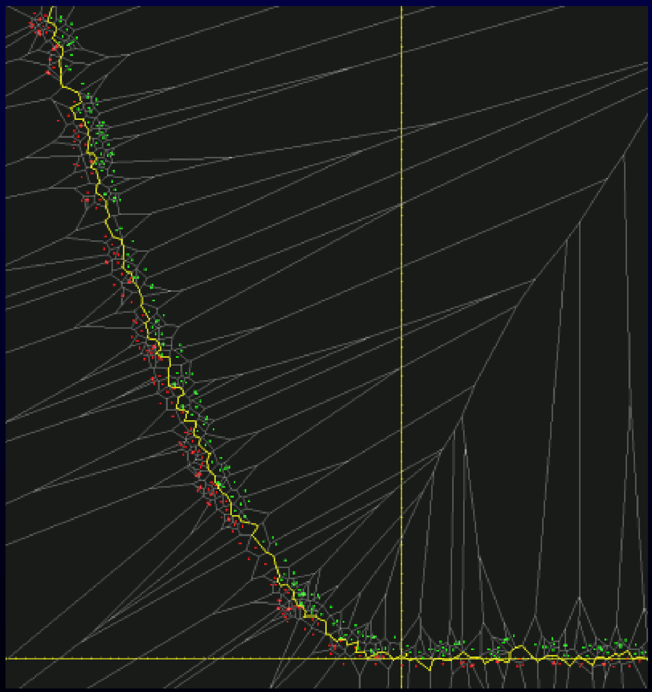
- control input actually applied:
  - **L1** $\longrightarrow$ user's control input
  - **L2** $\longrightarrow$ the breach-free control *"closest"* to user's
  - **L3** $\longrightarrow$ the control input with largest $T_{eb}{}^{\dagger}$
  - **L4** $\longrightarrow$ N/A†

(†: see "least detrimental" control)

– Practical Approximations –

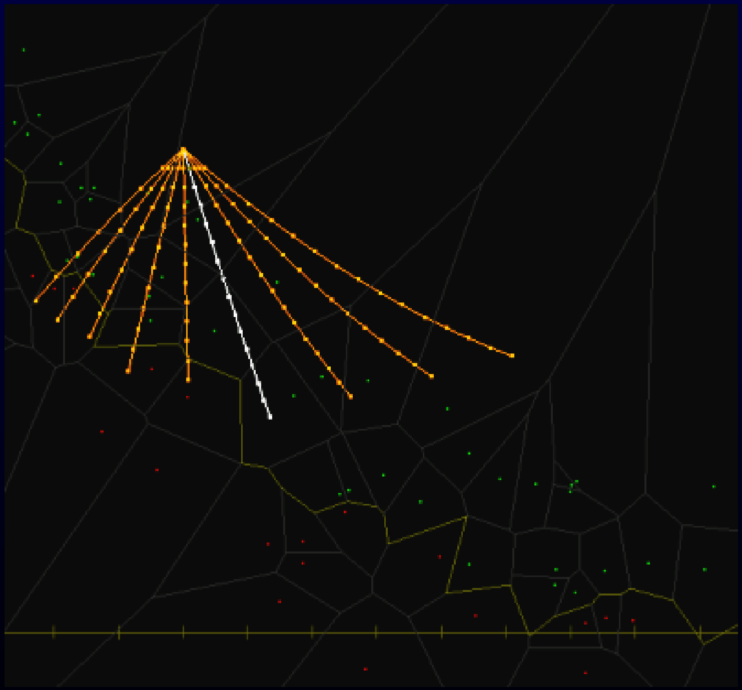
Envelope Approximation

- unlikely to have analytic representation
- must approximate (from samples, other data)
- used: *Nearest Neighbor* machine learning method



Discretization of Control Input

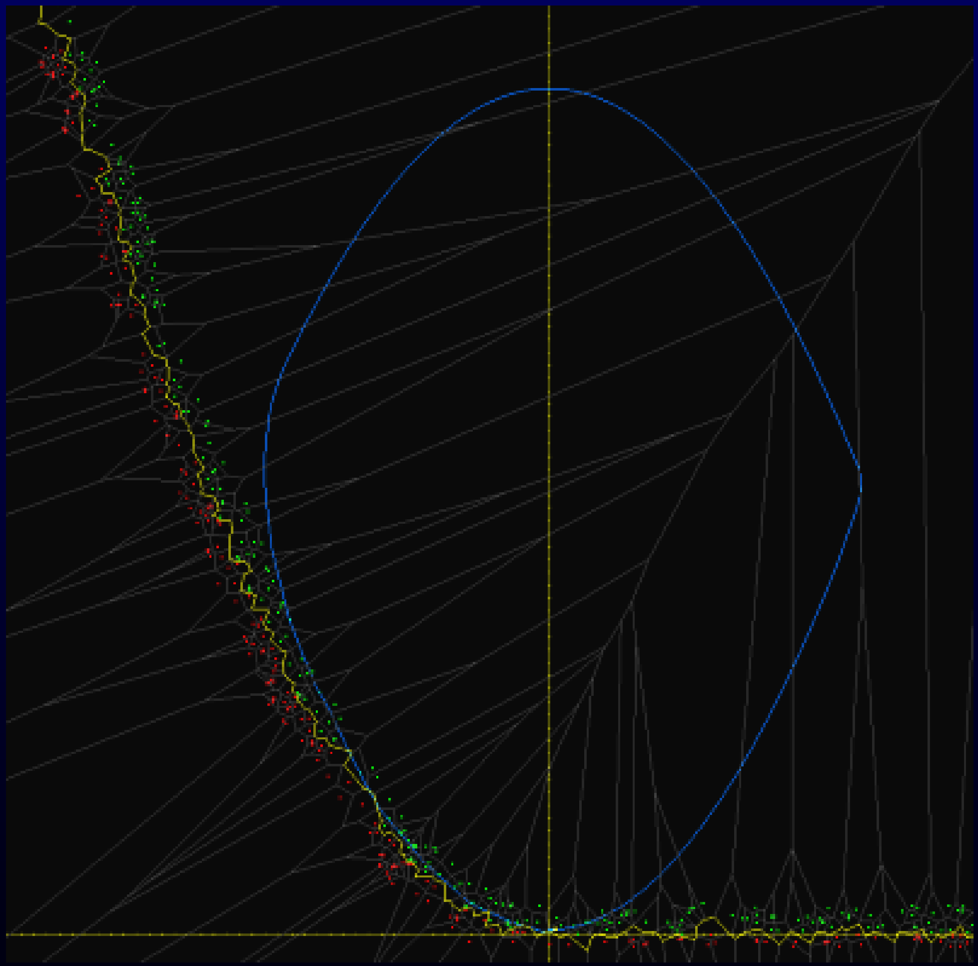
- often need to search or map over the input space, $\mathcal{U}$ (e.g., finding maximal $T_{eb}(x, u)$)
- intractable if $\mathcal{U}$ is large or continuous
- instead, work with a discretized subset, $\widehat{\mathcal{U}}$



– Some Results –

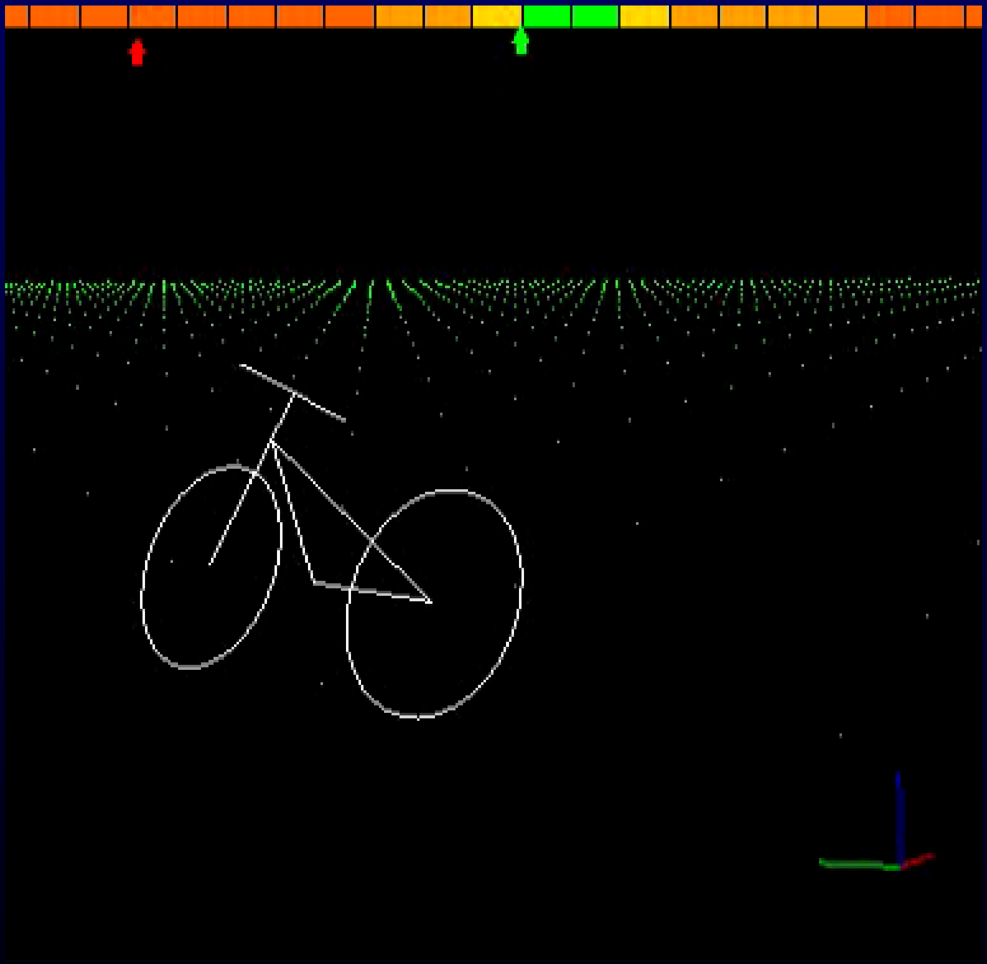
Rocket

- *[movies: world-space, state-space]*



Bike

- *[movie]*



Future Work

- evaluate with more complex systems (higher D)
- multi-dimensional inputs: how to spread corrections across the dimensions?
- incorporate haptics, literally do *“pushing the envelope”*
- what if only local environment known?

Summary & Take-away

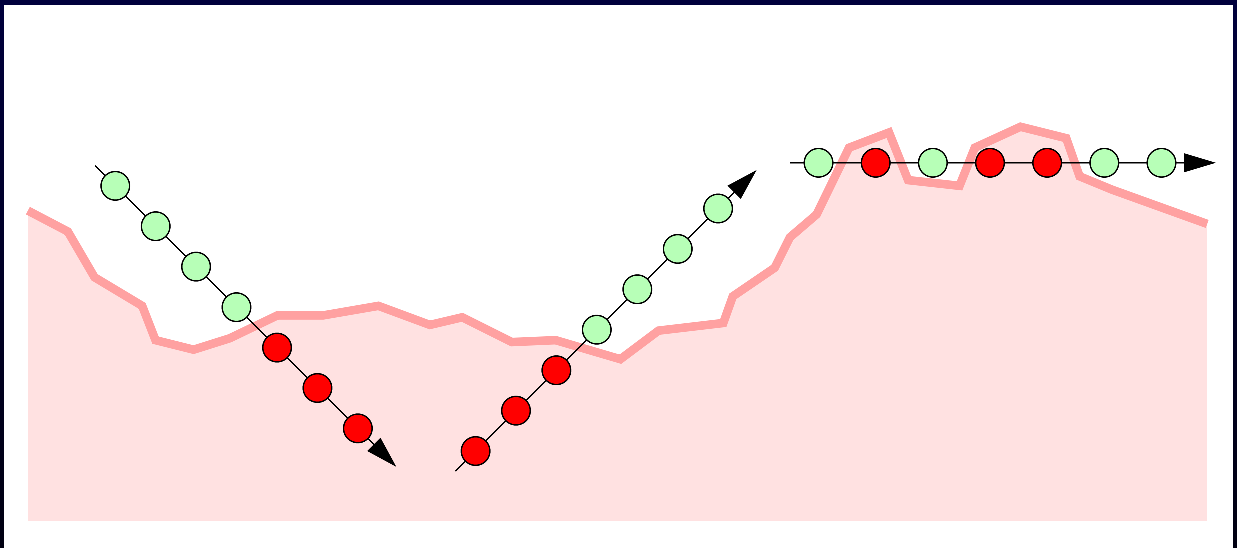
- real-time constraint of dynamical system to viable region
- predictive look-ahead using constant inputs
- T_{eb}, the *"time to envelope breach"* (clamped to T_h, the *"time horizon"*)
- used to choose among four control policies
- `http://www.dgp.toronto.edu/~mac/viab_env`

— ❦ *End* ❦ —

(supplementary material follows)

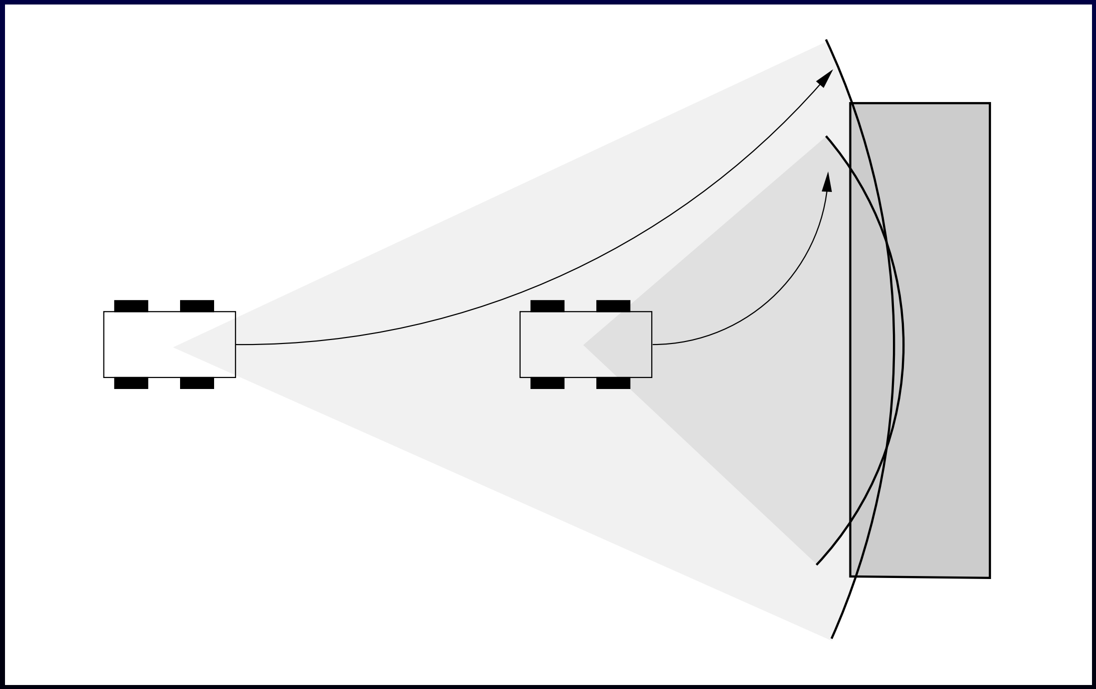
Grace Period

- a method to combat NN surface "noise"
- T_{gr}: max time system is allowed to cross NN envelope before being identified as a "true transition"



Why multi-step leads to milder corrections

- more time and space to maneuver
- can do no worse: at worst apply the same control signal as with a shorter time horizon



Why the “constant-input” assumption

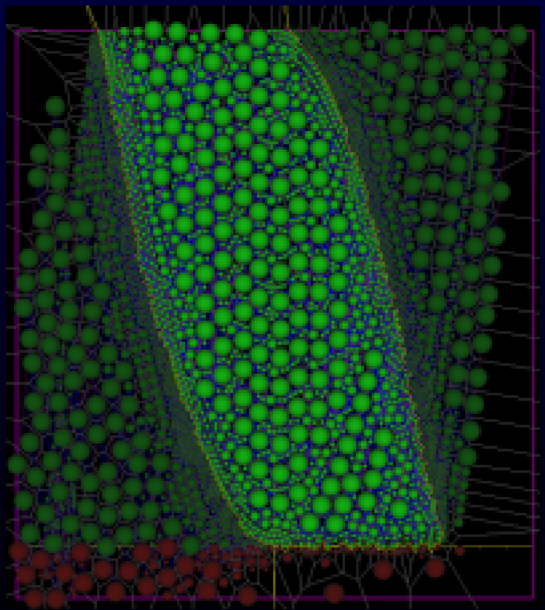
- in calculating $T_{eb}(x, u)$, need to make assumption about future values of u
- for non-constant input signals, no guiding principle to select the “optimal” one
- viability theory: *generalized inertia principle*
- also, user input tends to change slowly, relative to the time scale in question (T_h)
- hence assume constant-input

“Least detrimental” emergency control

- **problem**: meta-state **L4** can be reached
  - due to envelope approximation error
  - when all “recovery” trajectories out of an **L3** state require non-constant input
- **“solution”**: apply the control which spends least time outside envelope

Constructing Envelopes

- *Nearest Neighbor* used to approximate envelope
- possible NN sample sources: heuristic, empirical, analytic
- other forms can converted to NN samples through queries
- also can compute directly from dynamics (slow)

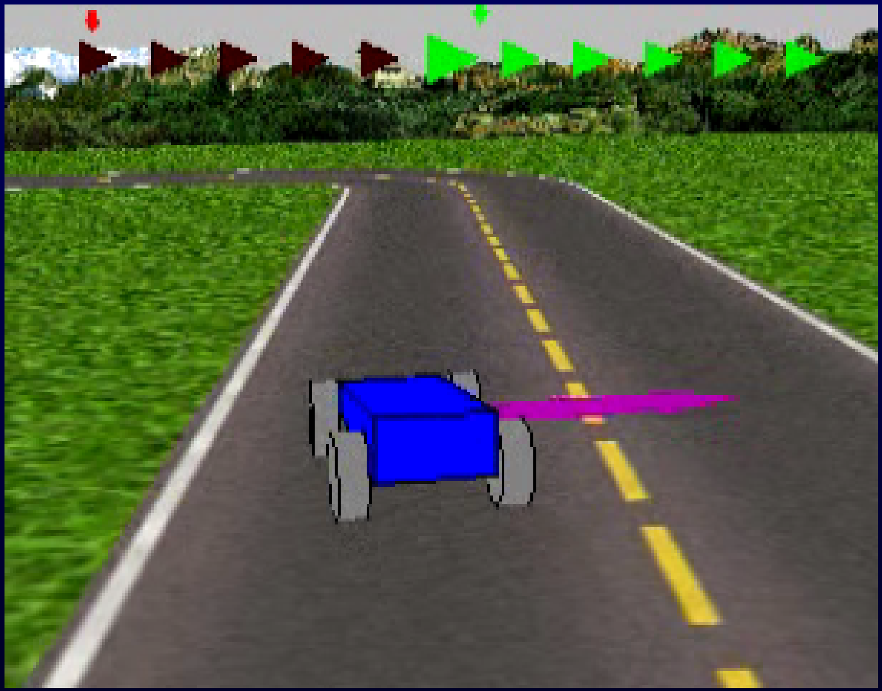


Scalability

- **online** algorithm: $O(|\widehat{\mathcal{U}}| \cdot T_h)$
- **offline** algorithm (envelope construction):
  - $\#$ of NN samples for equivalent-quality envelope tends to grow exponentially with state-space dimensionality
  - envelope geometry tends to be simple, relative to $\#$ of dimensions
  - perhaps other learning methods can give better scalability (SVM?)

Car – track

- *[movie]*



Leftovers

Motivation (short)

- **problem**: direct human control of dynamical systems is often difficult, prone to error and failure
  (e.g., control-by-wire of a bike)
- particularly difficult for users unfamiliar with system
- **idea**: computer aids the user by keeping system controllable
- **motivation**: *"pushing the envelope"* metaphor

Overview

- Framework
  - taxonomy of state-space
  - containment strategy
  - T_{eb}, system meta-states, and control policy

Overview

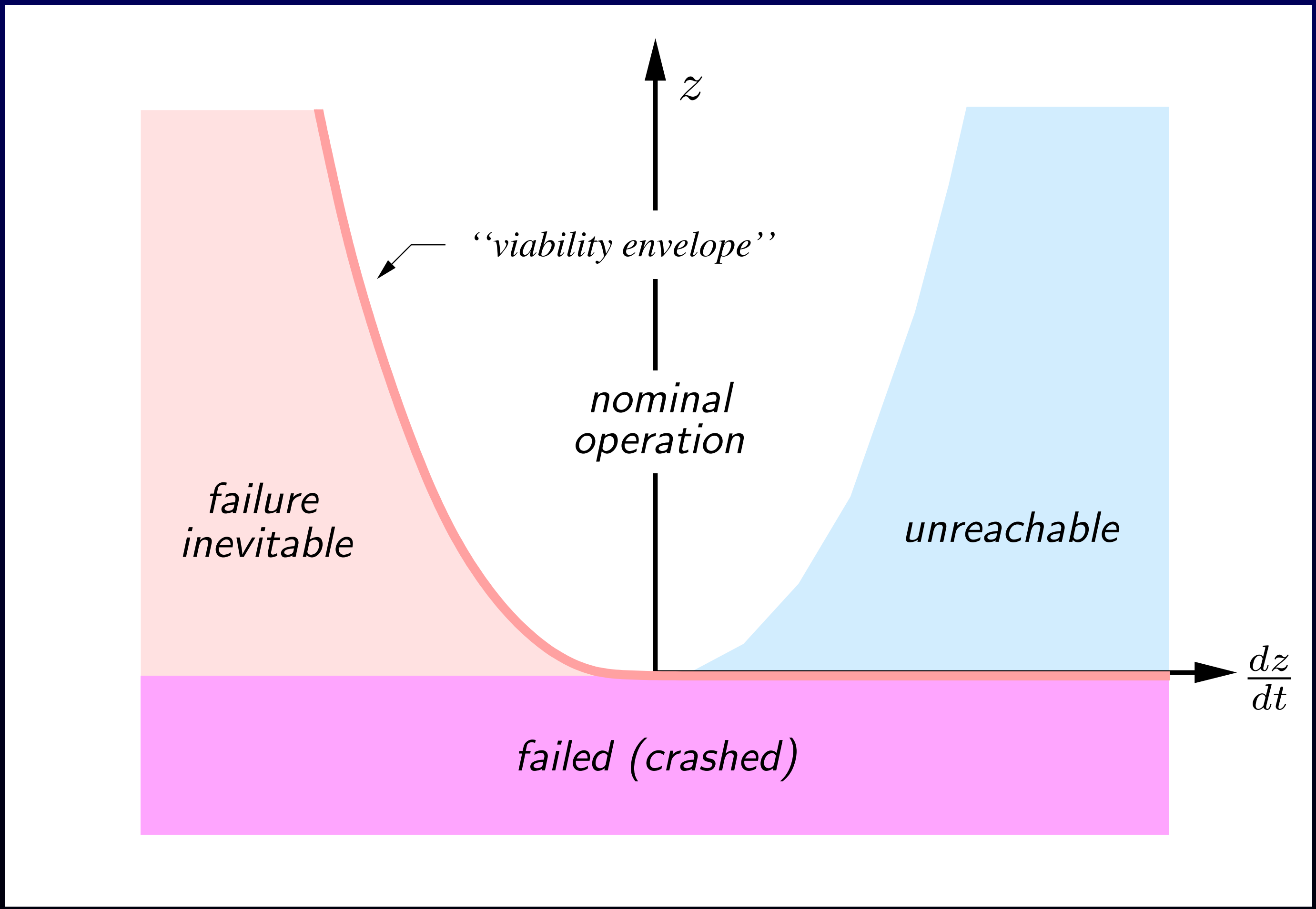
- Framework
  - taxonomy of state-space
  - containment strategy
  - T_{eb}, system meta-states, and control policy
- Practical approximations
  - approximating envelopes with *Nearest Neighbor*
  - discretization of control input

Overview

- Framework
  - taxonomy of state-space
  - containment strategy
  - T_{eb}, system meta-states, and control policy
- Practical approximations
  - approximating envelopes with *Nearest Neighbor*
  - discretization of control input
- Some results

Taxonomy of State-space

- a landing rocket with bounded thrust (z = altitude)



Car

- [movie]