Simplex Architecture for Run Time Assurance of Hybrid Systems

Abhishek Murthy
Computer Science, Stony Brook University

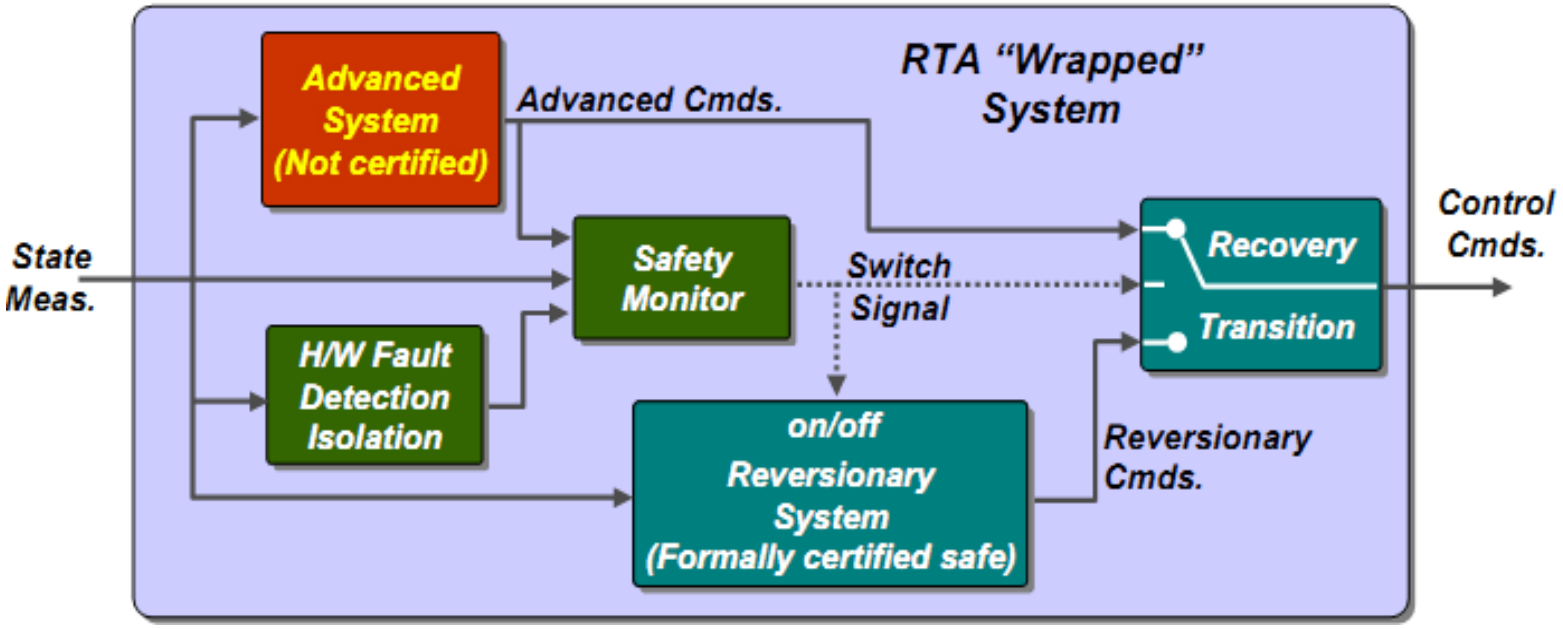
Joint work with: Tushar Deshpande, Ariful Islam, Justin Seyster,
Ezio Bartocci, Erez Zadok, Scott Stoller, Scott Smolka and Radu Grosu

Safe and Secure Systems and Software Symposium (S5) – Dayton , OH
June 12-14 2012

Outline

1. Run Time Assurance of Complex Systems
2. Simplex Architecture
3. Hybrid Systems Modeling Framework
4. Barrier Certificates
5. Model Predictive Controllers for Neurons
6. Summary

Run-Time Assurance (RTA) - Wrapper Architecture



The RTA wrapper architecture - Anthony M. Aiello et. al. "Run-Time Assurance for Advanced Flight-Critical Control Systems" 2010

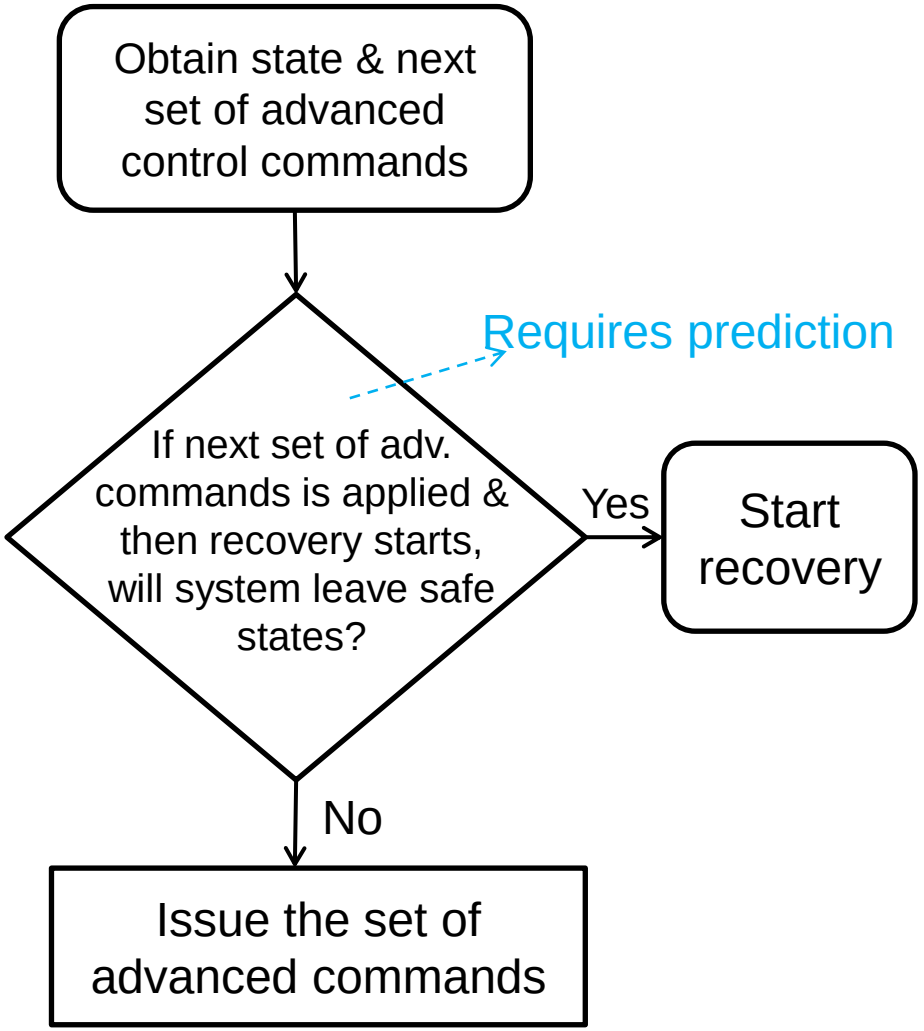
- h/w fault detection
- safety monitor
- reversionary system
- transition logic

Certified ✓

⇒ Run-time safety of system

RTA Safety Argument – Soundness & Completeness

- **Reversionary Safety Envelope (RSE)** = subset of "admissible states" where reversionary system is certified to control plant
- **Recovery** = transfer of control from advanced to reversionary system
- **Obtainable recovery** = system stays within RSE throughout recovery
- **RTA Safety Argument =** "set of states computed at run-time or offline such that recovery is obtainable if initiated from some state within it"
- Soundness = recovery should be guaranteed from states identified in safety argument
- Completeness = Safety argument covers all states where recovery must be initiated
- Reactive approach (revert when unsafe) is neither sound nor complete

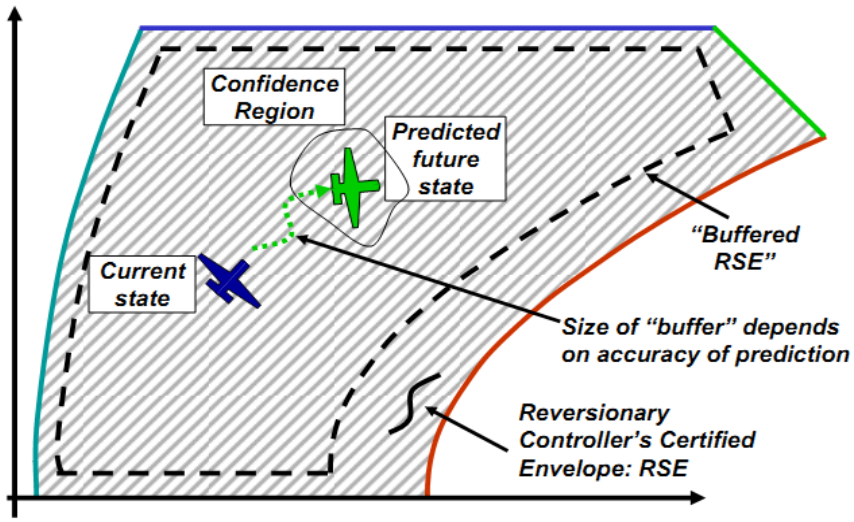


RTA – Offline vs. Online Approaches

1. Complete run-time prediction

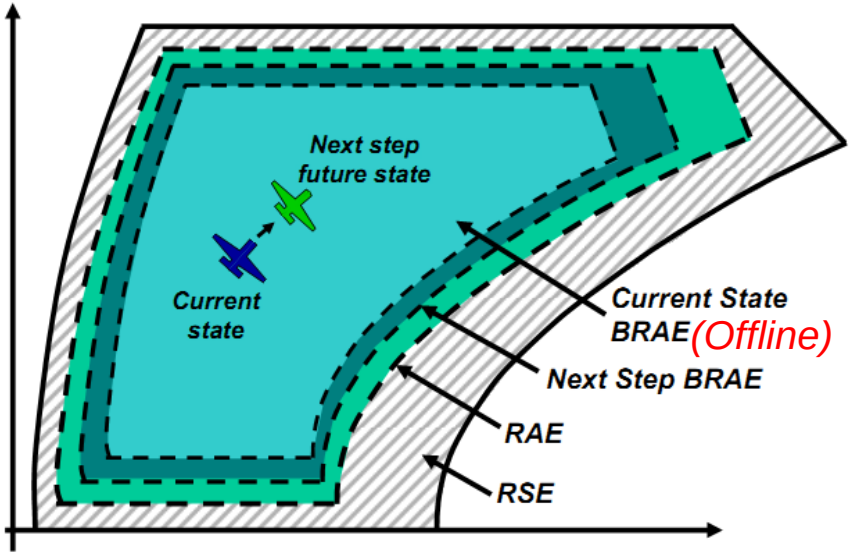
- Inspect trajectory of detailed simulation: left RSE? initiate recovery, else adv. commands

2. Complete offline prediction



Complete offline prediction approach to RTA – theoretical (source: A.M Aiello et. al.)

3. Combined approach



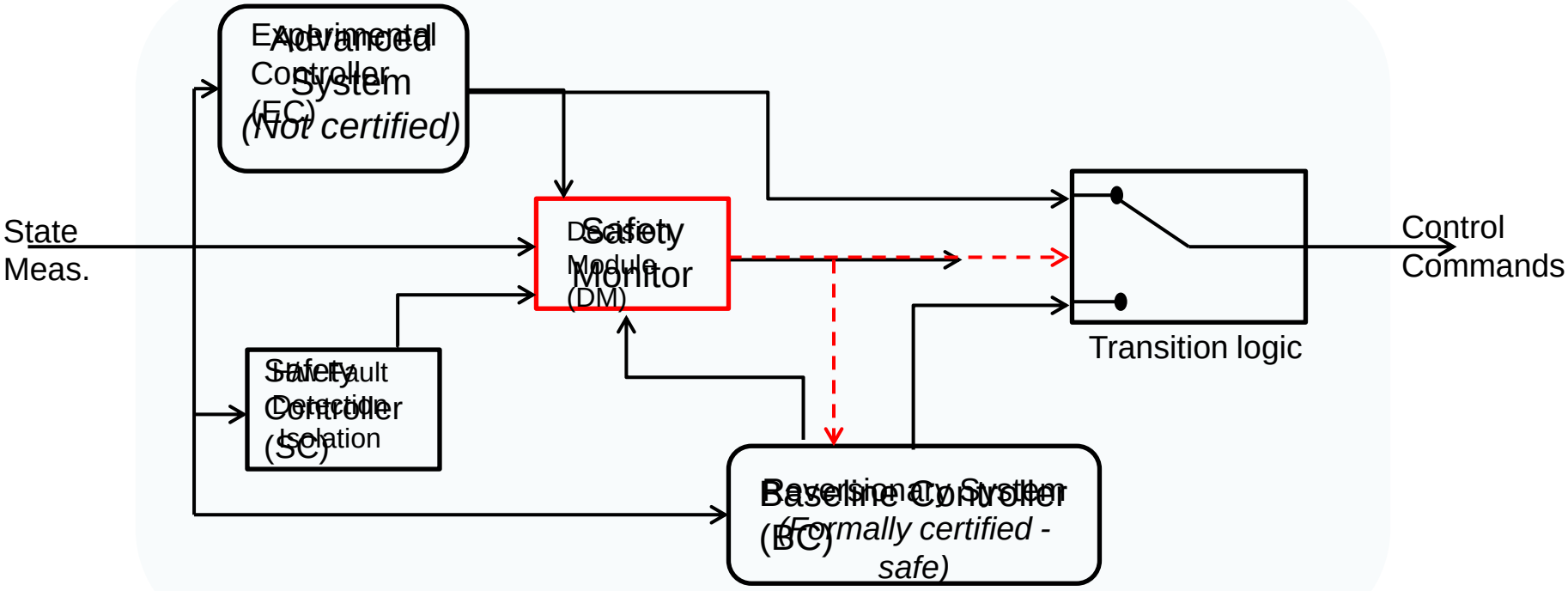
Combined & current-state (offline) RTA approaches (source: A.M Aiello et. al.)

Recovery Achievability Envelope (RAE) = set of states where full recovery to reversionary system achievable

$\{x: \phi_i(x) \in RSE, \forall i \in \{1, \ldots, m\}\}$, where, ϕ - transition function representative of m-step recovery (ideal case, usually not in closed-form)

Simplex Architecture v2.0
- D. Seto, B. Krogh, L. Sha & A. Chutinan

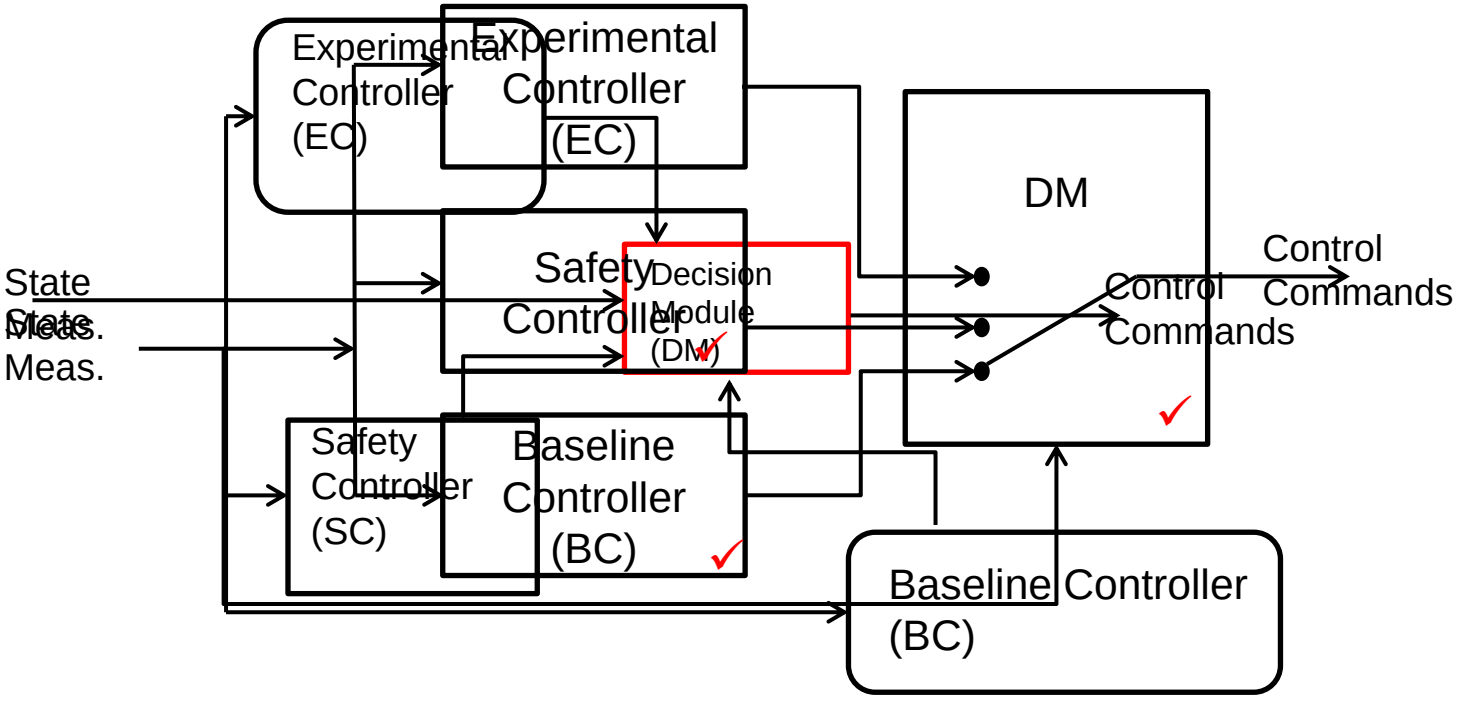
- Assures safety by exploiting analytical redundancy between advanced/upgraded unverified controller (EC) & a certified controller (BC)
- No run-time prediction, pre-computed regions of ensured stability



Schematic of the Simplex Architecture, source: Seto et. al. "The Simplex Architecture for On-Line Control System Upgrades"

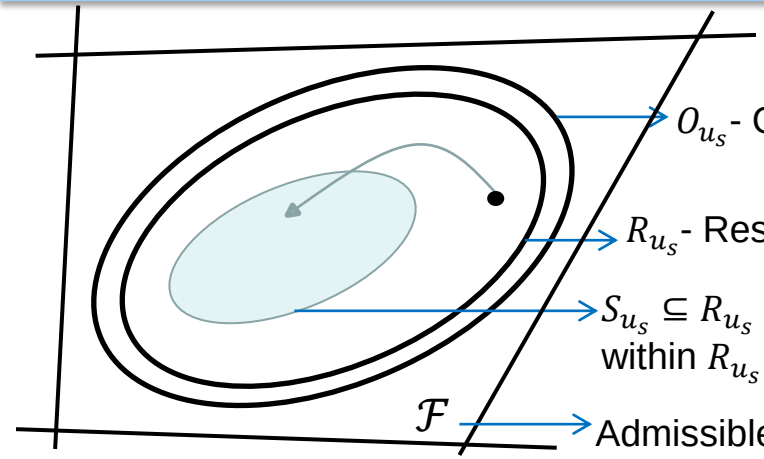
Simplex Architecture v2.0
- D. Seto, B. Krogh, L. Sha & A. Chutinan

- Assures safety by exploiting analytical redundancy between advanced/upgraded unverified controller (EC) & a certified controller (BC)
- No run-time prediction, pre-computed regions of ensured stability



Schematic of the Simplex Architecture, source: Seto et. al. "The Simplex Architecture for On-Line Control System Upgrades"

Simplex Architecture – Switching Logic



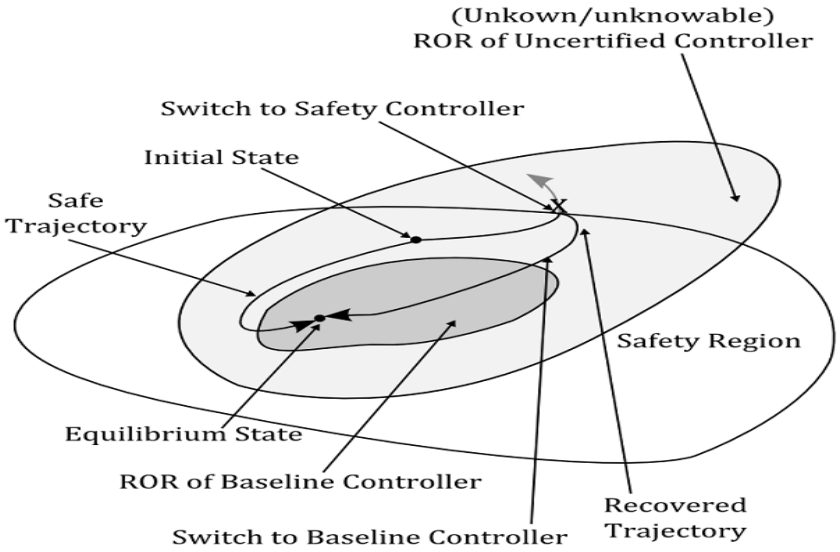
O_{u_s} - Operational Region (OR) – invariant for SC

R_{u_s} - Restricted OR (ROR) for SC – safety region

$S_{u_s} \subseteq R_{u_s}$ to which SC can invariably recover from within R_{u_s}, $S_{u_s} \subseteq R_{u_B}$, by design

$\mathcal{F}$ Admissible states (polytope)

Recovery
SC steers the plant to a state inside R_{u_b}, where BC can take over

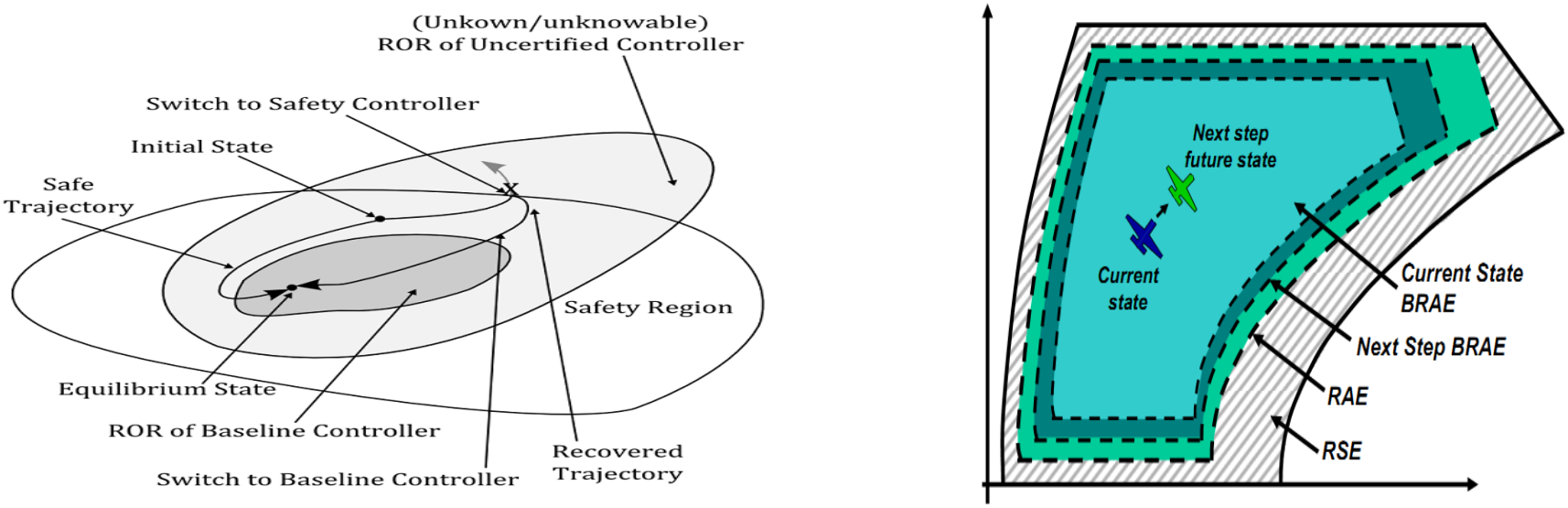


Switching logic

1. Monitor plant when EC is active
2. If state reaches boundary of R_{u_s}, switch to SC
3. When SC has steered plant into S_{u_s}, switch to BC

source: Seto et. al. "The Simplex Architecture for On-Line Control System Upgrades"

Simplex Architecture versus RTA



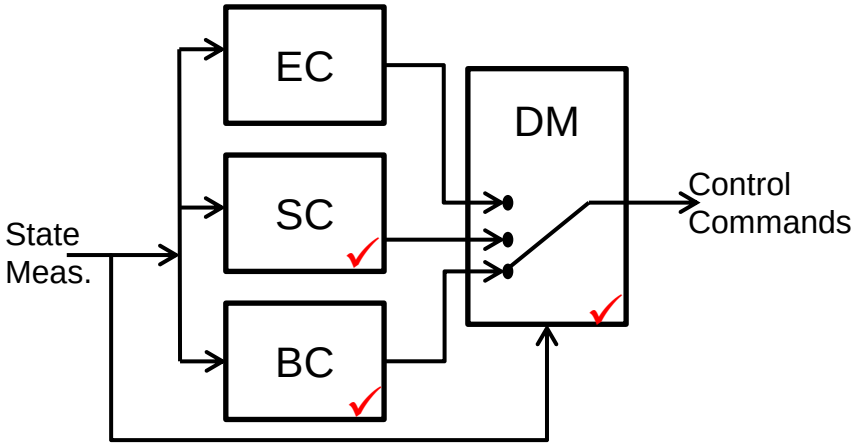
Comparison of switching logics of the Simplex & the RTA-wrapper architecture

1. ROR of Baseline controller ≈ RSE
2. Safety controller's steering ≈ transitions during recovery performed by transition logic

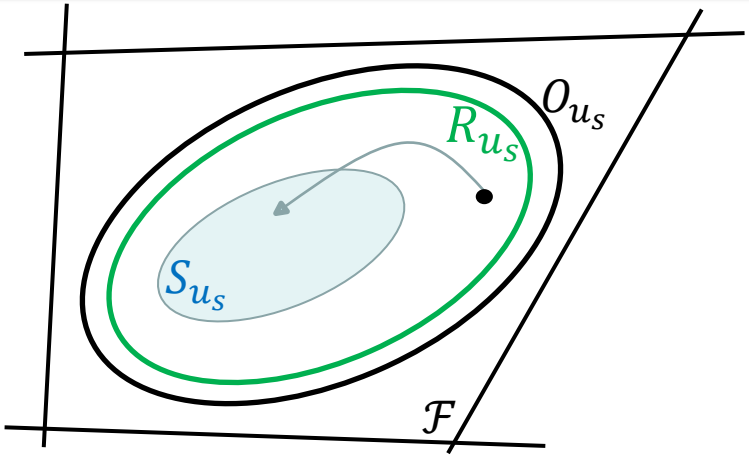
Simplex does not perform any run-time prediction.

Simplex allows experimental controller to operate over a region larger than Baseline Controller's OR.

Simplex Architecture - Design Issues



The Simplex architecture



Invariants & design requirements

$S_{u_s} \subseteq R_{u_B}$

1. Verification (safety) of SC and BC
2. Designing and verifying DM's switching logic, which performs recovery

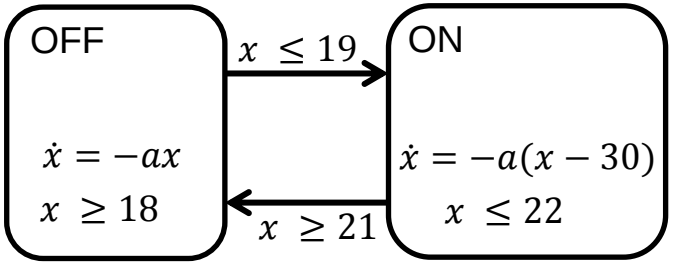
Nonlinear systems: software, embedded systems, state-space models of complex systems

- Difficult to design controllers & verify

Hybrid Systems

Hybrid Systems – Computational modeling framework for systems with both discrete-time & continuous-time components

Enables Simplex architecture design for nonlinear systems



Hybrid Automata (HA), S

Verification

$S \cong_{B/S} S_{spec.}$?

$S \preccurlyeq_{B/S} S_{spec.}$?

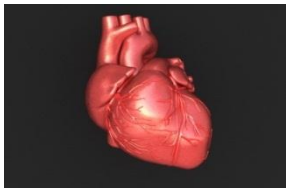
Control

Design C such that

$S \times C \cong_{B/S} S_{spec.}$

$S \times C \preccurlyeq_{B/S} S_{spec.}$

Linearization, Hybridization, Preprocessing



Modeling

$$\frac{dx_i}{dt} = f_i(t, x_1(t), x_2(t) \,..\, x_i(t) \,....\, x_n(t), \mathbf{u})$$

Nonlinear State-Space Models

Complex phenomena - biology, avionics

Hybrid Systems

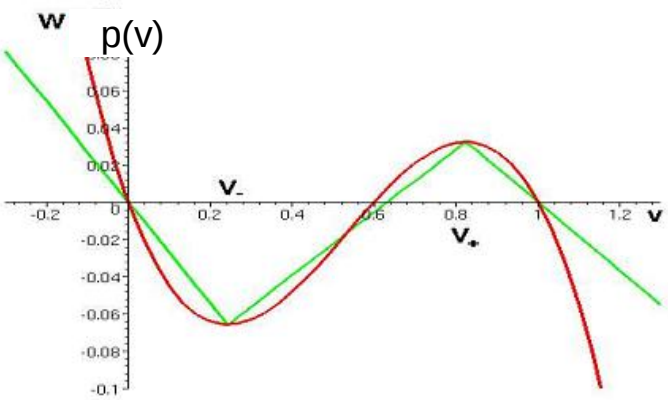
Linear Hybrid Automata (LHA): Mode-specific vector field is linear

Continuous-time state-space models to LHA

1. Jacobian
2. Dynamic programming-based optimal linearization of time series
3. Linearizing level curves of derivative terms
4. State-space partitioning-based: conservative approximation of nonlinear model

Consider the FitzHugh-Nagumo (FHN) model of neuron excitation

$$\dot{v} = v(1-v)(v-a) - w + I$$
$$\dot{w} = bv - cw$$



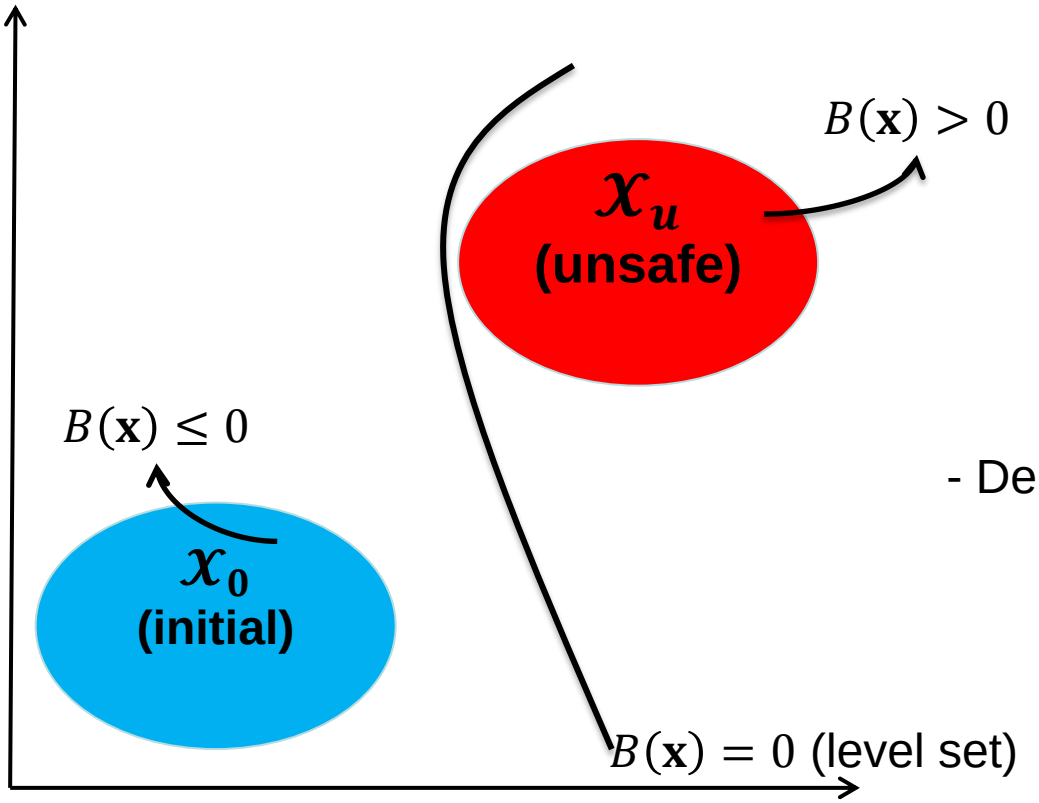
Dumas's hybrid model

$$\dot{v} = \tilde{p}(v) - w + I \;,\; \dot{w} = bv - cw$$

$$\tilde{p}(v) = \begin{cases} \dfrac{p(v_-)}{v_-} v & v < v_- \\ \dfrac{p(v_+) - p(v_-)}{v_+ - v_-} v & v_- \le v \le v_+ \\ \dfrac{p(v_+)}{1 - v_+} (1 - v) & v > v_+ \end{cases}$$

Barrier Certificates (BaCs)

- Method for safety verification of hybrid systems, without explicit construction of reachable states
- BaC $B(\mathbf{x})$ - convex function of state-space determined using SemiDefinite Programming (SDP)



$$\frac{\partial B}{\partial \mathbf{x}}(\mathbf{x}).f(\mathbf{x}) \leq 0 \qquad \forall \mathbf{x} \in \mathbb{R}^n$$

- Derivative w.r.t time along trajectory ≤ 0

Barrier Certificates (BaCs) – Hybrid systems

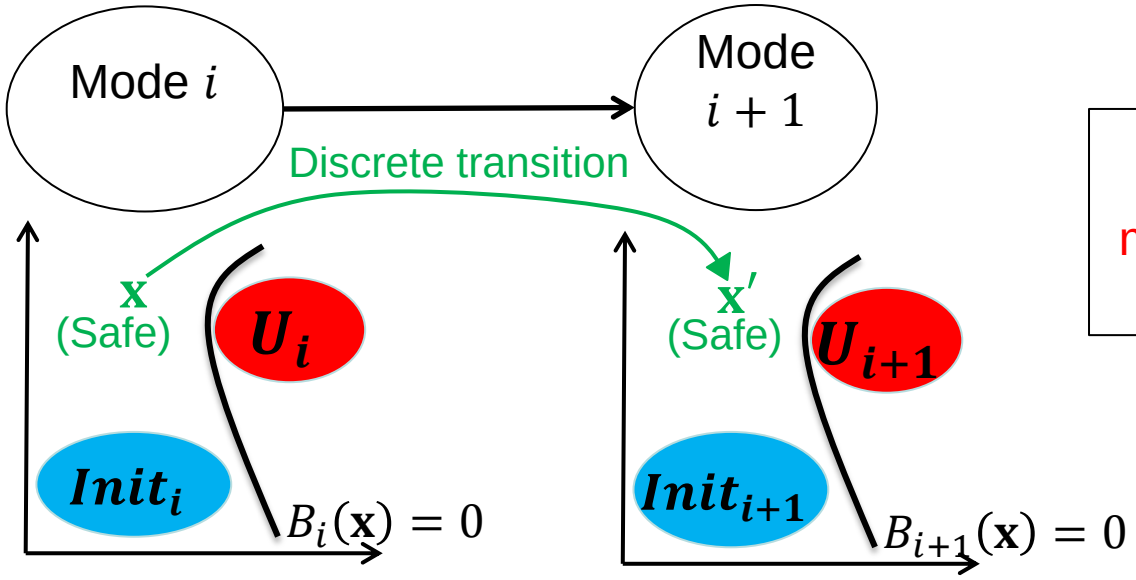
- Hybrid systems – Existence of BaC for each mode and safe transitions ⇒ safety

$$B_i(\mathbf{x}) > 0 \quad \forall \mathbf{x} \in U_i \ \text{(unsafe states within invariant of Mode } i\text{)}$$

$$B_i(\mathbf{x}) \leq 0 \quad \forall \mathbf{x} \in Init_i$$

$$\frac{\partial B_i}{\partial \mathbf{x}}(\mathbf{x}).f_i(\mathbf{x}) \leq 0 \quad \forall x \in Inv_i$$

$$B_i(\mathbf{x}) \leq 0, \forall \mathbf{x} \in Reset(i', i)(\mathbf{x}') \ for\ some\ i' \in L\ and\ \mathbf{x}' \in Guard(i', i)\ \ with\ B_{i'}(\mathbf{x}') \leq 0$$

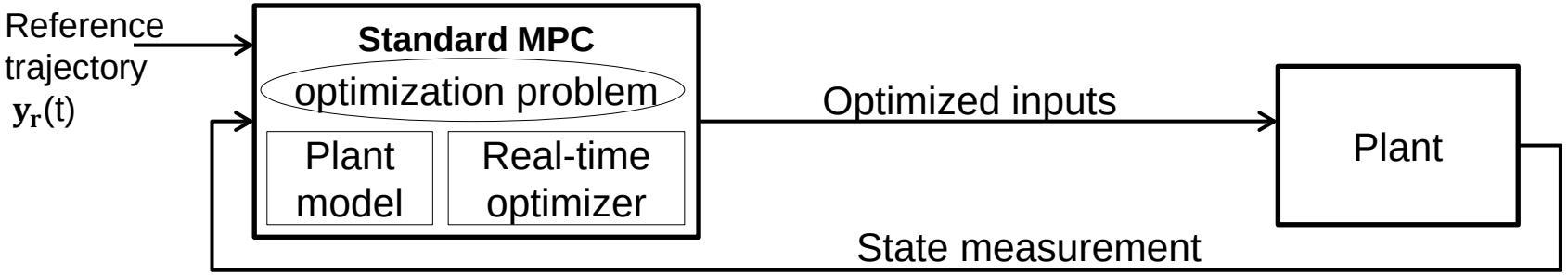


BaC in each mode =

mode-specific OR for Simplex architecture

Model Predictive Controllers (MPC)

Proposed EC in Simplex architecture



$$f(\mathbf{u}) = \sum_{i=1}^{W} \alpha^{i-1} \left[\mathbf{y_r}(t_0 + iT) - \mathbf{y}(t_0 + iT) \right]^2 + \beta_i \mathbf{u}(t_0 + iT)$$

horizon

sampling time

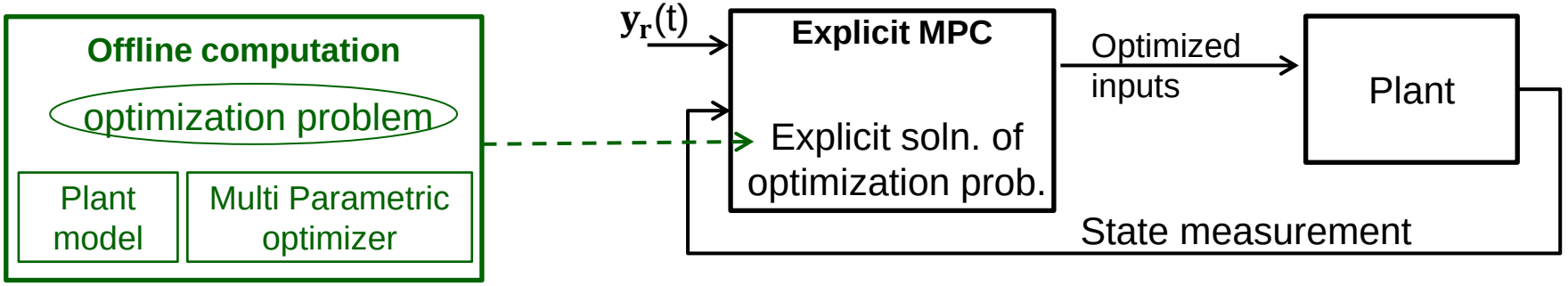
Obj. func. minimized by optimizer

receding horizon

outputs from reference trajectory

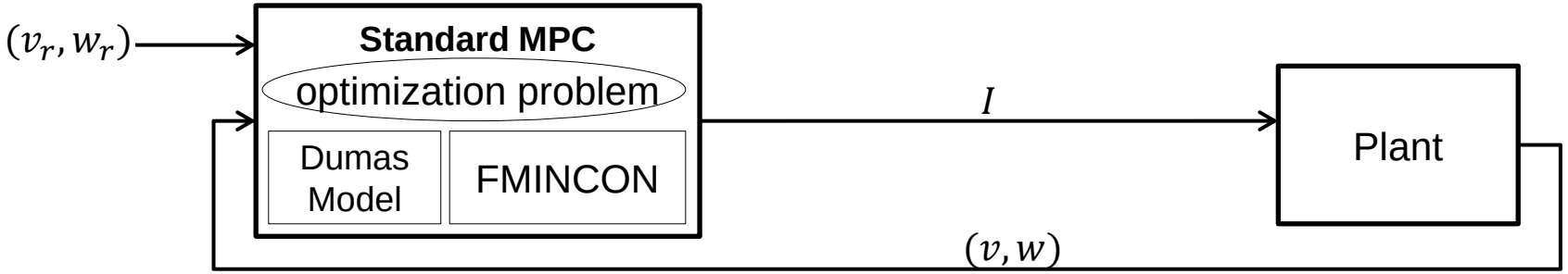
outputs predicted using plant model

weighted input

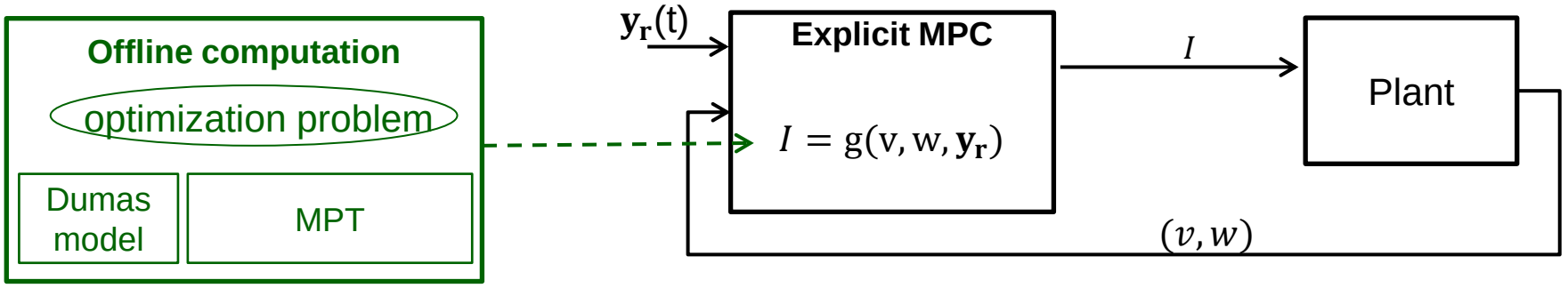


Model Predictive Controllers (MPC)

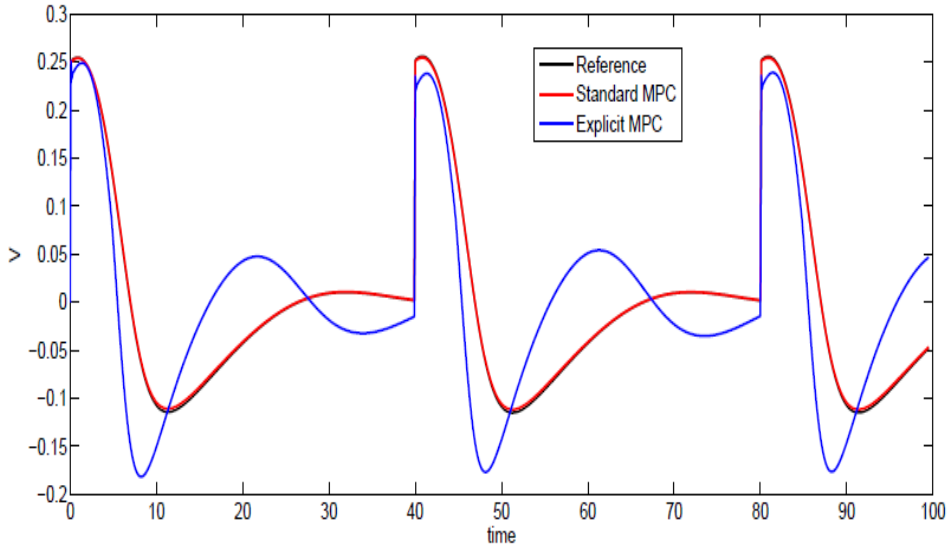
Proposed EC in Simplex architecture

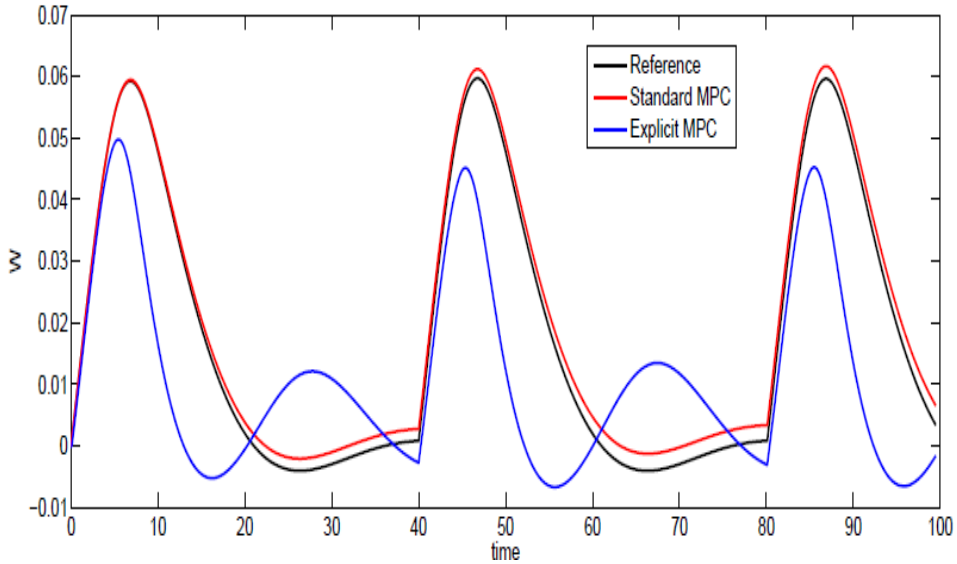


$$f(I) = \sum_{i=1}^{3} (0.8)^{i-1} \left(\begin{bmatrix} v_r \\ w_r \end{bmatrix} (t + iT) - \begin{bmatrix} v \\ w \end{bmatrix} (t + iT) \right)^2 + 0.1I(t_0 + iT)$$



MPC Implementation Results





- ≈2X speedup for explicit MPC
- Explicit MPC commits larger error

Currently investigating speed versus accuracy tradeoffs

Summary

- ✓ RTA-wrapper architecture (Barron's)
  - Safety argument, soundness & completeness
  - Complete run-time prediction, complete offline & combined approaches
- ✓ CMU's Simplex architecture
  - Safety Controller (SC) - transition from the Experimental Controller (EC) to the certified Baseline Controller (BC)
  - Establishing Operational Regions (OR) and DM's switching logic
  - **Allows EC to control plant over a region larger than the certified region for SC**
- ✓ Hybrid systems
  - Modeling framework for systems with both discrete-time & continuous-time components
  - Linear Hybrid Automata (LHA) – amenable to analysis & control
  - Linearizing nonlinear systems using Jacobian, Dynamic Programming on time series, derivative level curves or state-space partitioning
- ✓ Model Predictive Controller (MPC) design at Stony Brook
  - Fast & accurate MPC for highly nonlinear systems, using hybrid automata