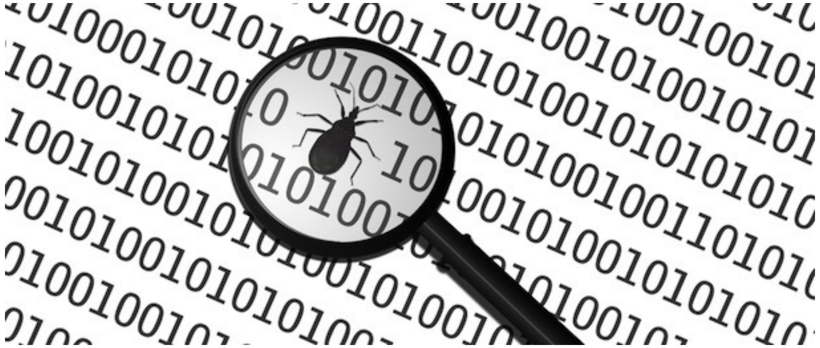
In Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS 2018)

Guaranteed Physical Security with Restart-Based Design for Cyber-Physical Systems

Fardin Abdi*, Chien-Ying Chen*, Monowar Hasan*, Songran Liu†,
Sibin Mohan*, and Marco Caccamo*

* University of Illinois at Urbana-Champaign, USA
† Northeastern University, China

Motivation

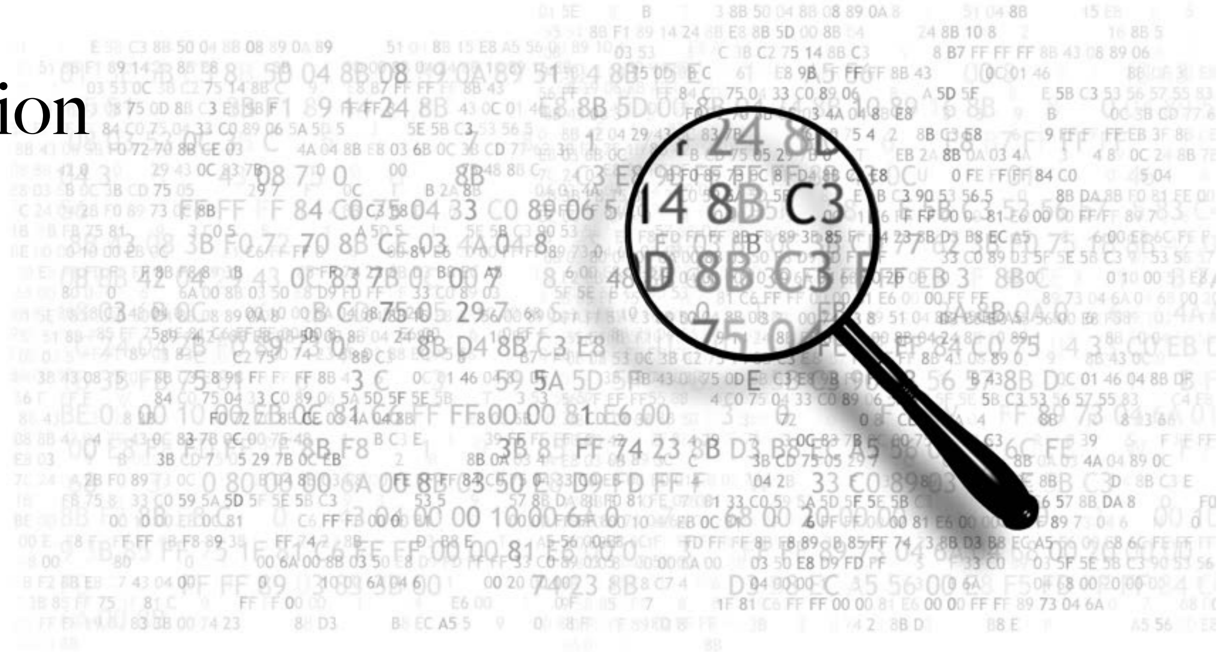


- Faults are a norm
- Fault-tolerant designs for safety-critical systems

- Attacks have the same effect
- What about safety in presence of security attacks?

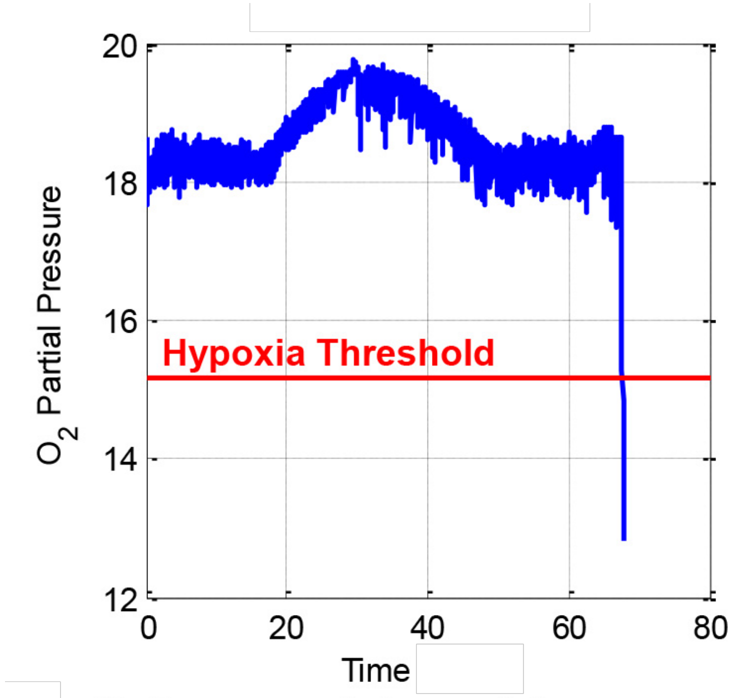
Focus of Existing Security Research

- Building attack proof sub-components e.g., TrustZone
- Identifying vulnerabilities
- Runtime monitoring and detection
- Integrity preservation
- …

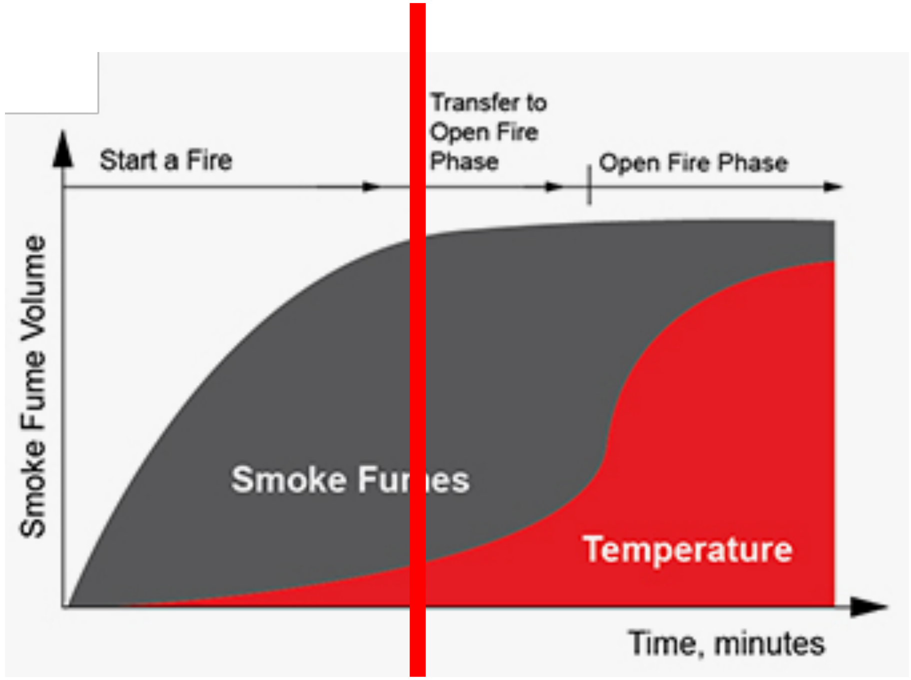


This Work

- Offering a methodology for guaranteeing critical safety invariants of a physical system, in presence of an attacker.



Life Support Unit O2 Partial Pressure



Fire Threshold

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Secure Execution Interval (SEI)

- During SEI we can trust that the system is going to execute uncompromised software and adversary cannot interfere with the system in any way.

Restart-Based Recovery

In this work, restarting the system and reloading uncompromised software triggers the SEI.

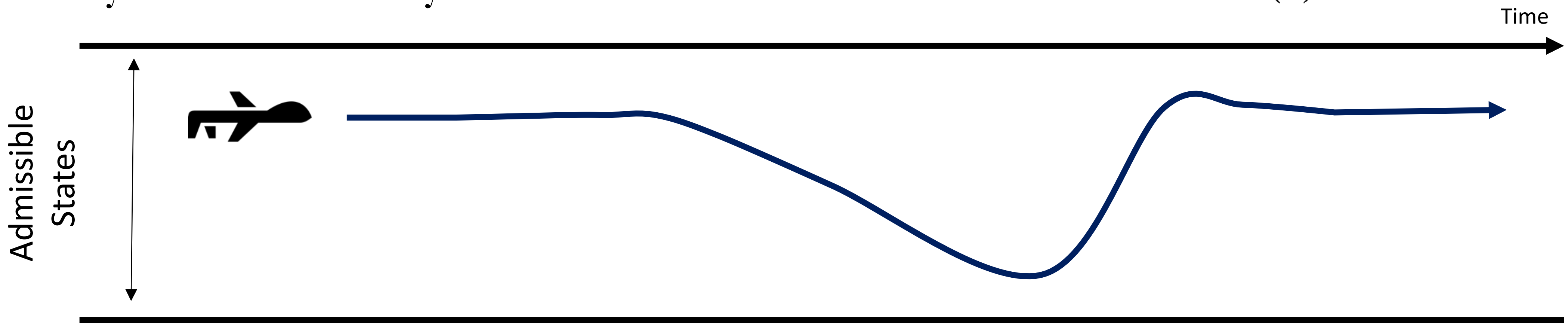
- Deterministic in terms of time and state.
- Reclaims stale resources e.g., memory, file pointers.

- External timer issues the restart command

Admissible States

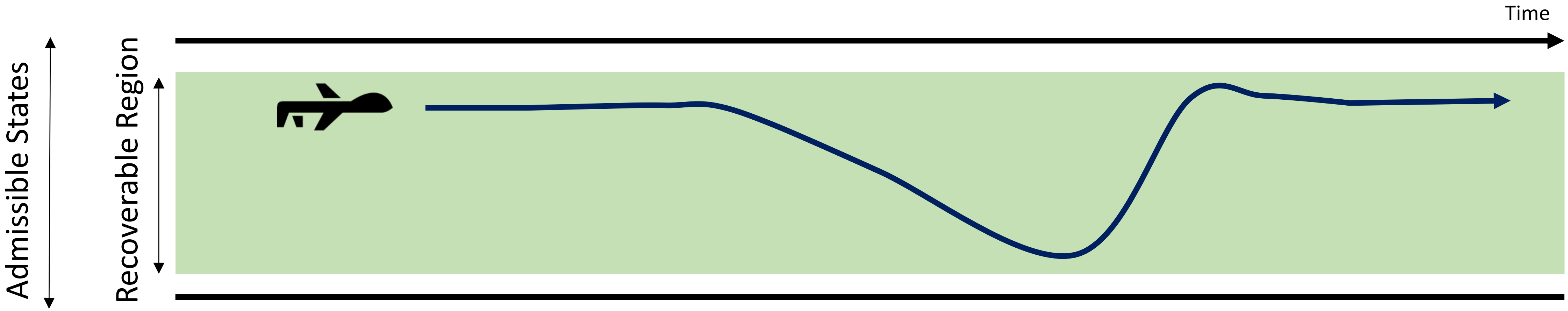
- States that do not violate any of the operational constraints of the physical plant are referred to as admissible states and denoted by S.

- Safety Invariant:
  - System must always remain inside **Admissible States**: $\forall t : x(t) \in \mathcal{S}$



Recoverable States and Safety Controller

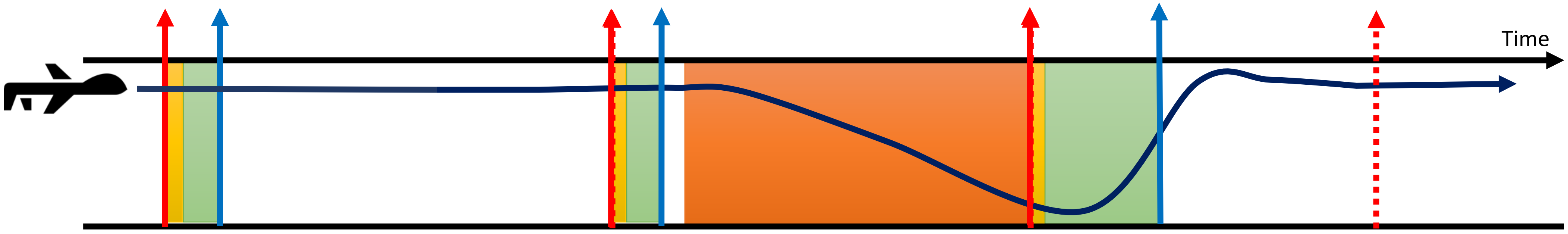
- Defined with regards to a given Safety Controller (SC) and are denoted by R. R is a subset of S such that if the given SC starts controlling system from x ∈ R, all future states will remain admissible.



Intuition Behind Guaranteed Safety

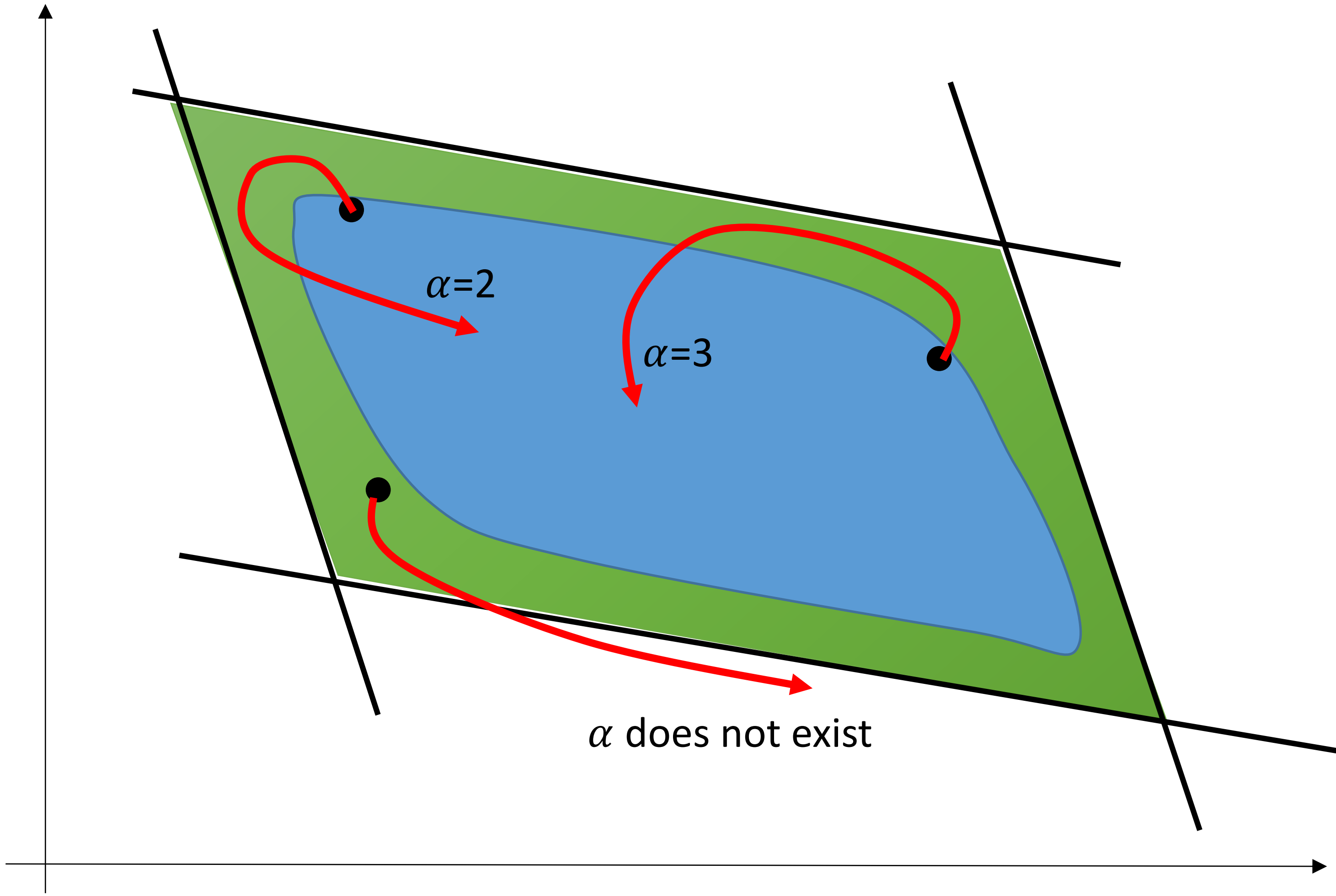
I. Due to physical inertia, the plant cannot immediately move to an inadmissible state.

II. SEI activations are separated such that the SC can stabilize the plant at the beginning of the following SEI.



True Recoverable States:

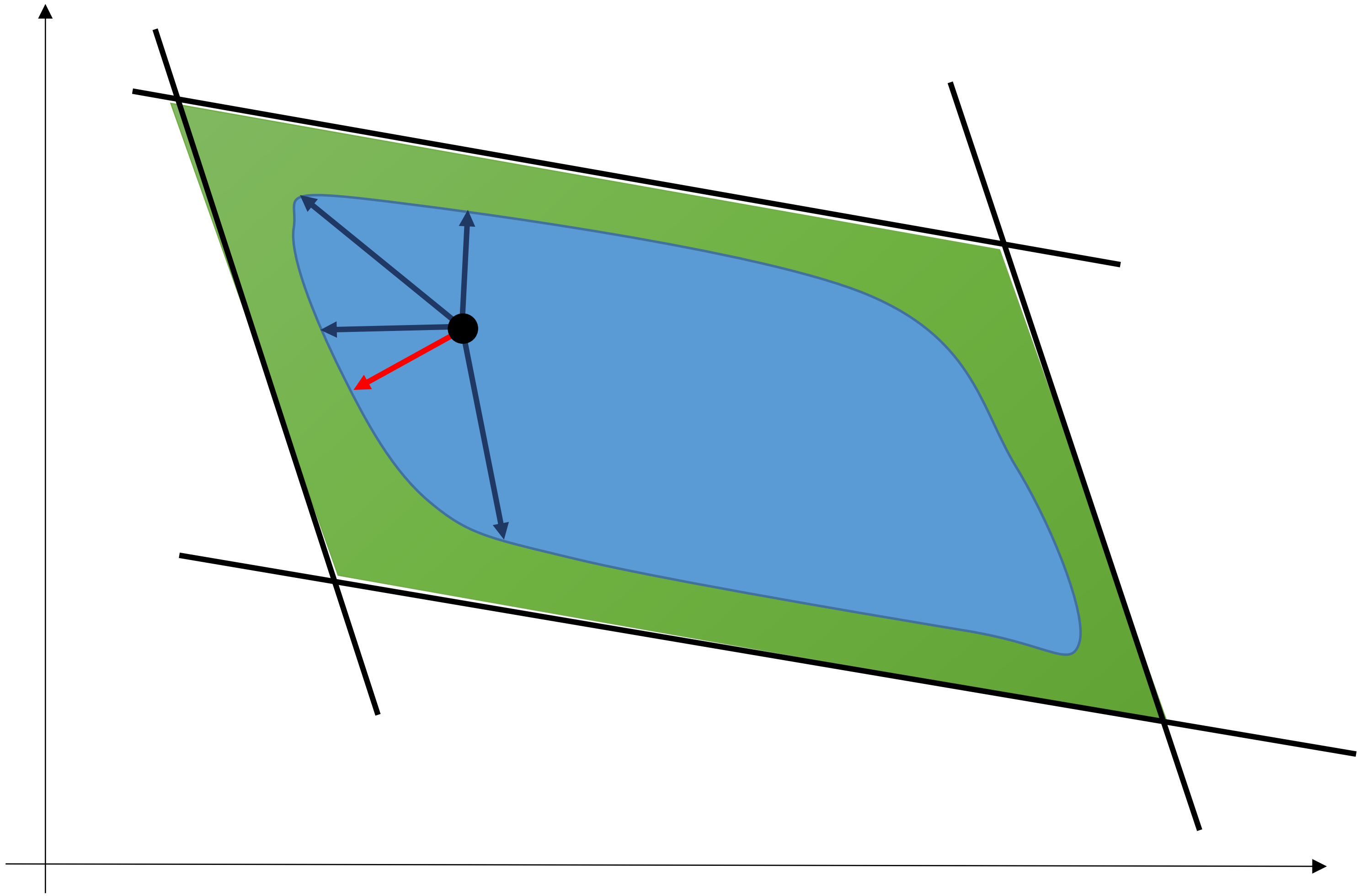
$$\mathcal{T} = \{x \mid \exists \alpha > 0 : \mathsf{Reach}_{\leq \alpha}(x, SC) \subseteq \mathcal{S} \;\&\; \mathsf{Reach}_{= \alpha}(x, SC) \subseteq \mathcal{R}\}$$



ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

If from a given state, we could calculate the shortest time to unsafety

$$\gamma(x) = \min\{\Delta(x, x') \mid \text{ for all } x' \notin \mathcal{T}\}$$



If from a given state, we could calculate the shortest time to unsafety

$$\gamma(x) = \min \{\Delta(x, x') \mid \text{ for all } x' \notin \mathcal{T}\}$$

Then we could trigger SEI before $\gamma(x)$ and system would remain safe.

$$\mathsf{Reach}_{\leq \gamma(x(t)) - \epsilon}(x(t), UC) \subseteq \mathcal{S}$$

$$\mathsf{Reach}_{= \gamma(x(t)) - \epsilon}(x(t), UC) \subseteq \mathcal{T}$$

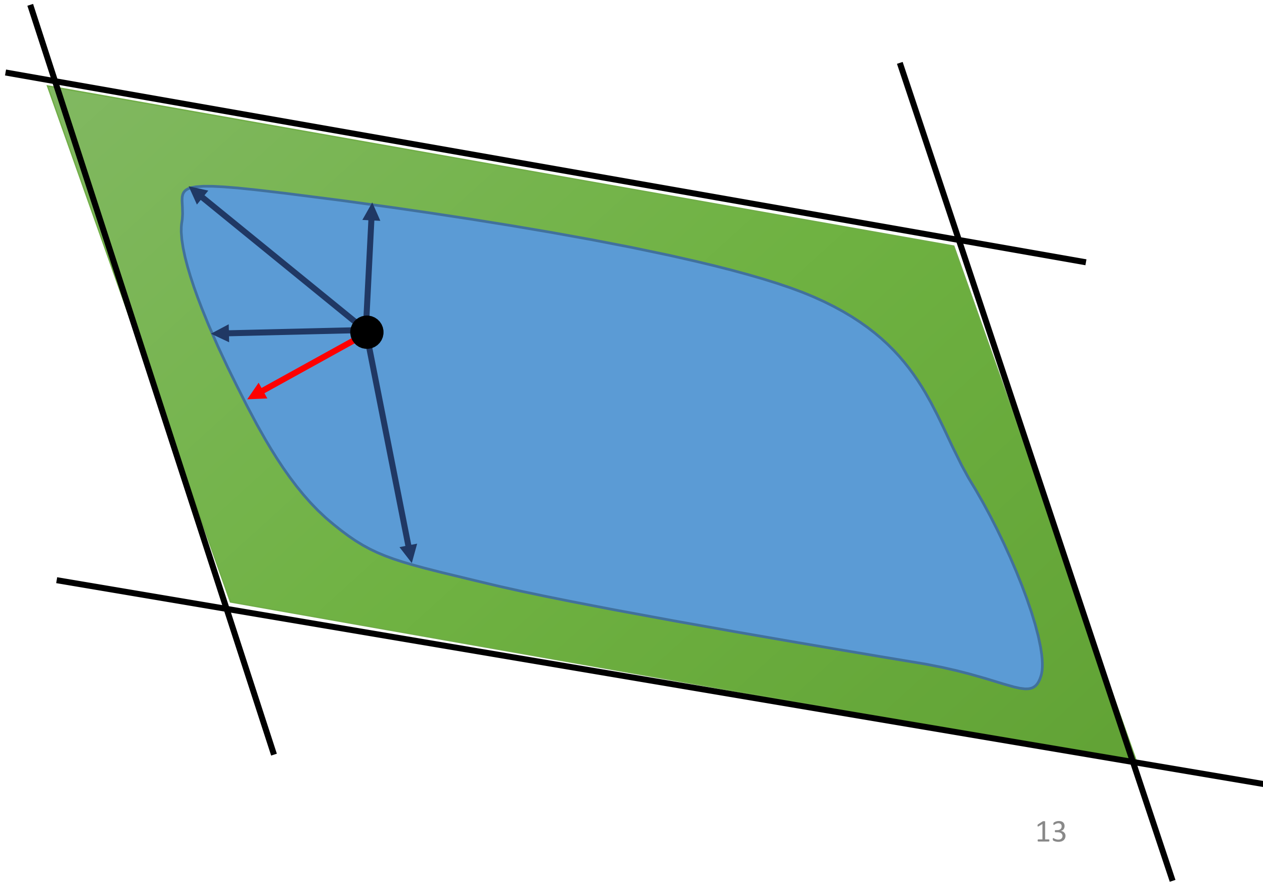
where $\epsilon \to 0$

$$\gamma(x) = \min\{\Delta(x, x') \mid \text{ for all } x' \notin \mathcal{T}\}$$

$$\text{Reach}_{\leq \gamma(x(t))-\epsilon}(x(t), UC) \subseteq \mathcal{S}$$

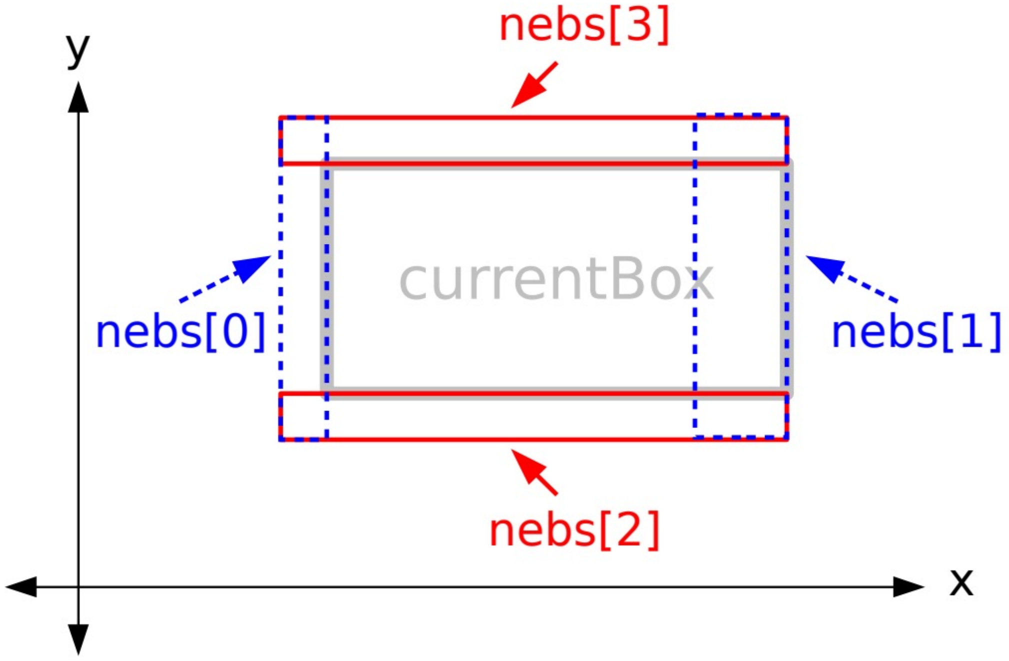
$$\text{Reach}_{= \gamma(x(t))-\epsilon}(x(t), UC) \subseteq \mathcal{T}$$

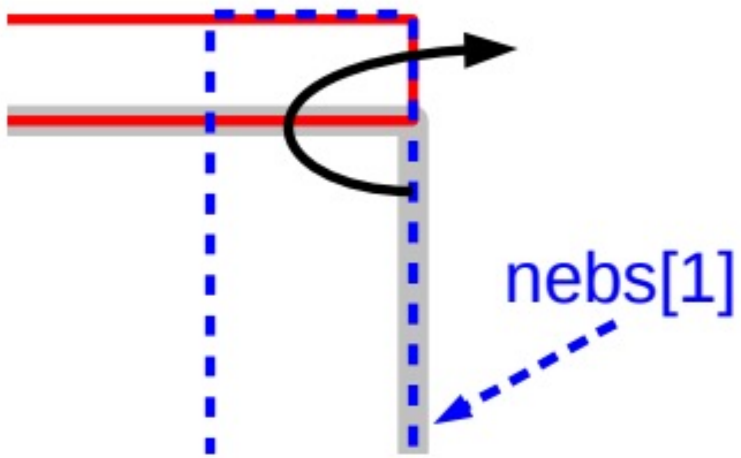
where $\epsilon \to 0$



Real-Time Reachability

- *"Real-Time Reachability for Verified Simplex Design"*, Stanley Bak, Taylor Johnson, Marco Caccamo, Lui Sha, 35th IEEE Real-Time Systems Symposium (RTSS 2014)





Using Real-Time Reachability

Evaluate specific λ as restart time with fixed α

$$\begin{cases} \text{Reach}_{\leq\lambda}(x(t), UC) \subseteq \mathcal{S} \\ \text{Reach}_{=\lambda}(x(t), UC) \subseteq \mathcal{T}_\alpha \end{cases}$$

$$\begin{cases} \text{Reach}_{\leq\alpha}(\text{Reach}_{=\lambda}(x(t), UC), SC) \subseteq \mathcal{S} \\ \text{Reach}_{=\alpha}(\text{Reach}_{=\lambda}(x(t), UC), SC) \subseteq \mathcal{R} \end{cases}$$

Finding a Safe Restart Time During SEI

Algorithm 1: Find Restart Time

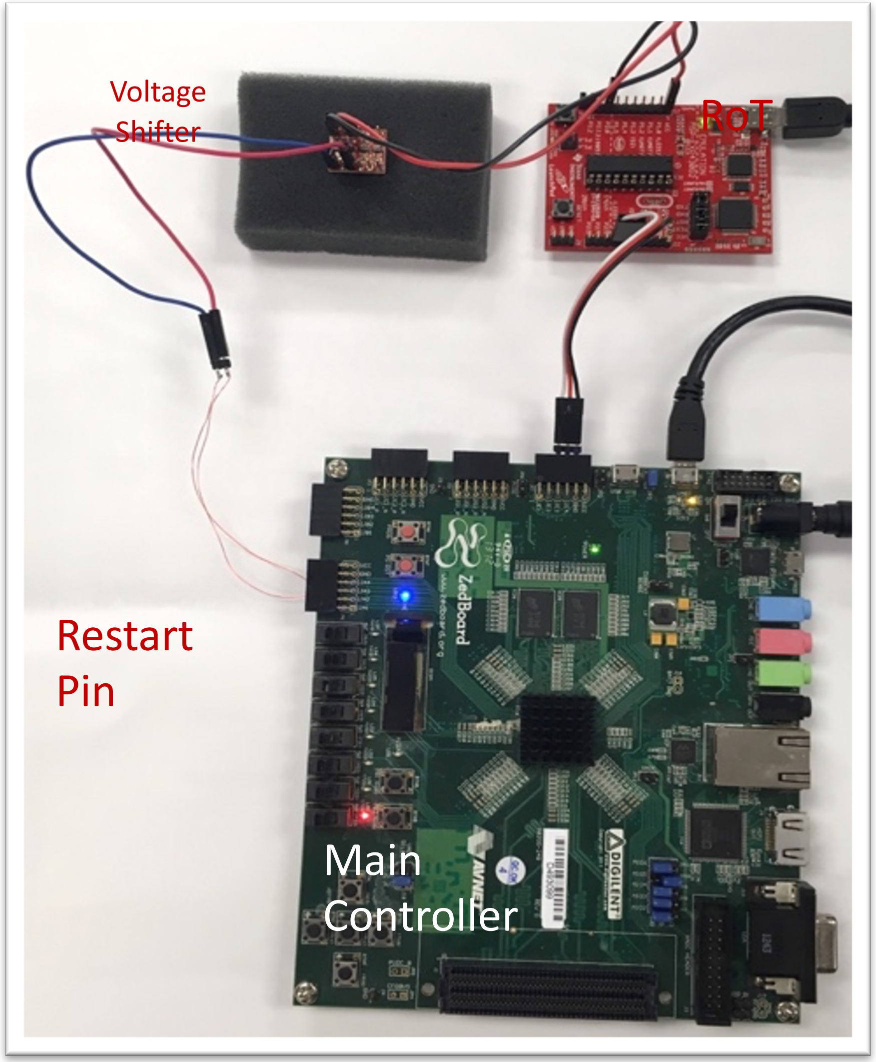
FindRestartTime$(x, \lambda_{\text{candidate}})$

startTime = currentTime()
RangeStart = 0;
RangeEnd = $\lambda_{\text{candidate}}$ /*Initialize range of binary search for λ_r*/
while currentTime() - startTime $< T_s$ **do**
  if conditions of Equation (4) are true for $\lambda_{\text{candidate}}$ **then**
    $\lambda_{\text{safe}} := \lambda_{\text{candidate}}$
    RangeStart = λ_{safe} ; RangeEnd $= 2 * \lambda_{\text{safe}}$ /* *increase the* $\lambda_{candidate}$ */
  else
    RangeEnd $= \lambda_{\text{candidate}}$ /* *decrease the* $\lambda_{candidate}$ */
  end if
  $\lambda_{\text{candidate}} := (\text{RangeStart} + \text{RangeEnd})/2$
end while
if $\lambda_{\text{safe}} >$ currentTime() - startTime **then**
  $\lambda_{\text{safe}} = \lambda_{\text{safe}} - $(startTime - currentTime())
  return True , λ_{safe}
end if
return False , –

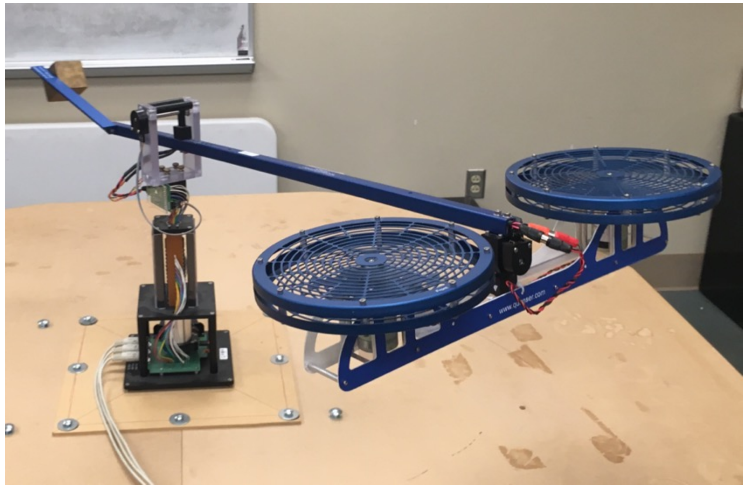
Controller

- Main controller:
  - ARM Cortex-A9 core of Xilinx's Zynq-7000
  - *FreeRTOS*
  - *Control Period: 20 ms* (50Hz)
  - Full Reboot Time: 390ms

- Root Of Trust:
  - MSP430G2452 micro-controller
  - 16 bit internal timer



Physical Plants



3 Degree of Freedom Helicopter

Warehouse Temperature Management
(Hardware in the loop Simulation)

Not to crash to the surface of table

keep the temperature within the range of 20 and 30 degrees Celsius

Attacks I: Killing the Controller

- Attacker killed the main controller task.
- The attack was activated at a random time after the end of SEI.
- Under this attack, the 3DOF helicopter always remained within the set of admissible states.

Attack II: Malicious Inputs

- Replaced sensor readings of the system with corrupted values with the aim of destabilizing the plant
- The attack was activated at a random time after the end of SEI.
- Under this attack, the 3DOF helicopter always remained within the set of admissible states.

Attack III:

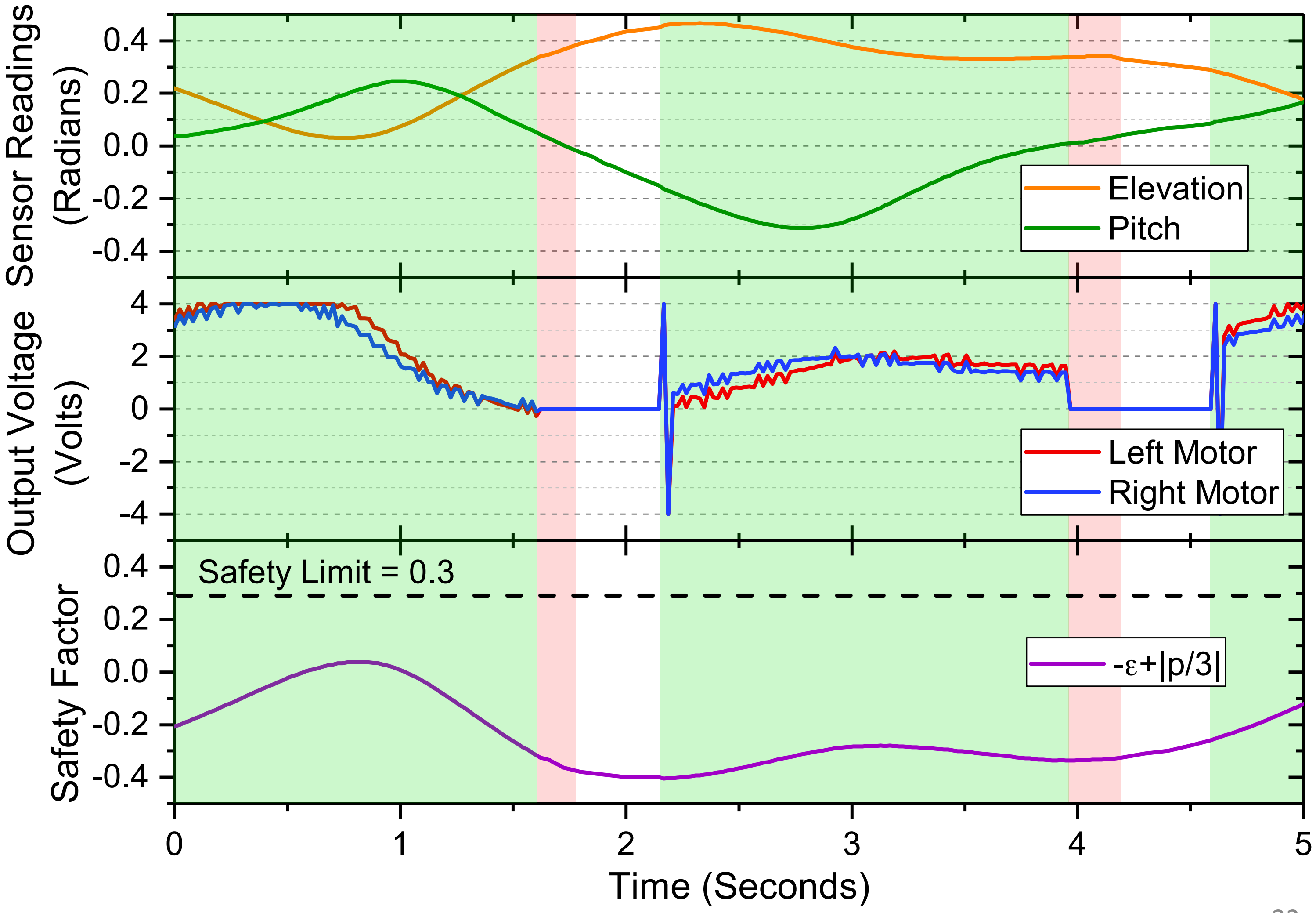
- Active immediately after the SEI
- Replaces the original controller with a malicious process that turns off the fans of the helicopter forcing it to hit the surface.
- Under this attack, the 3DOF helicopter always remained within the set of admissible states.

Attack III:

Green: SEI

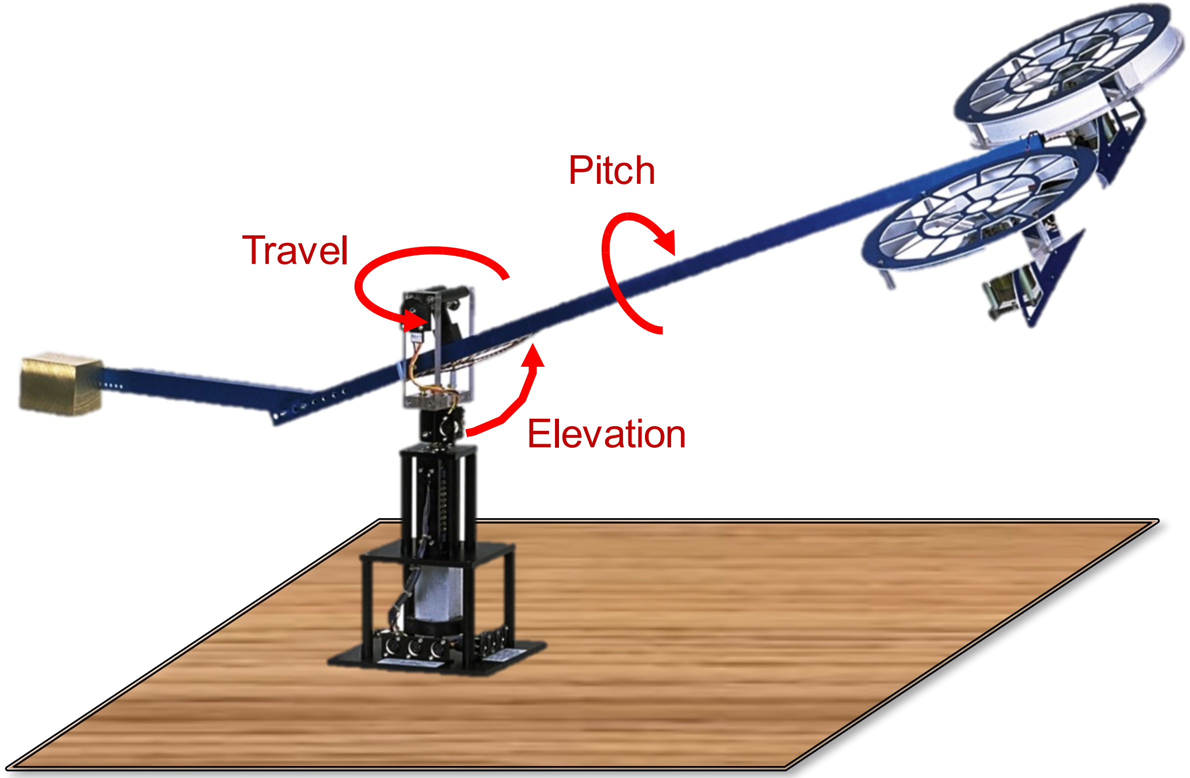
Red:
Normal operation
(In this case attacker)

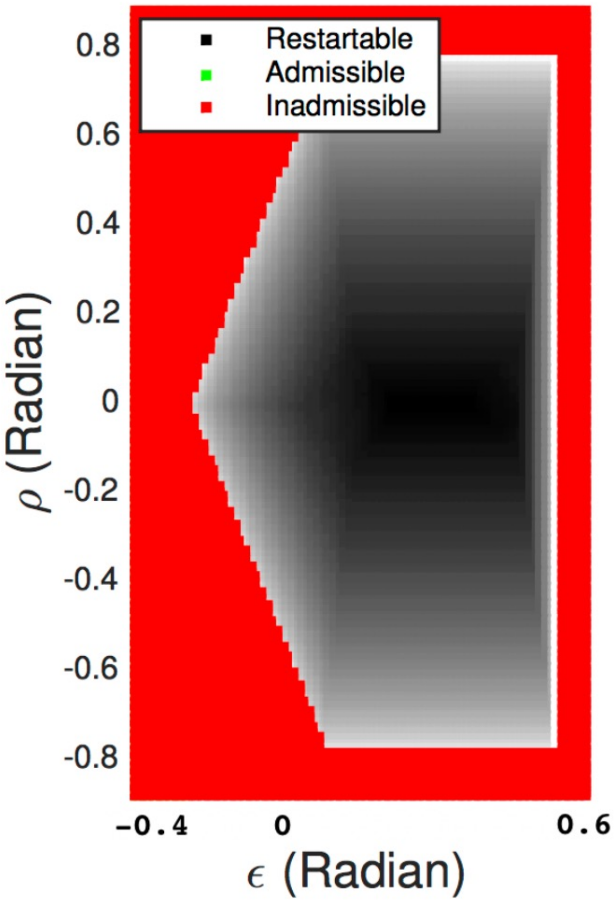
White: System Reboot



ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

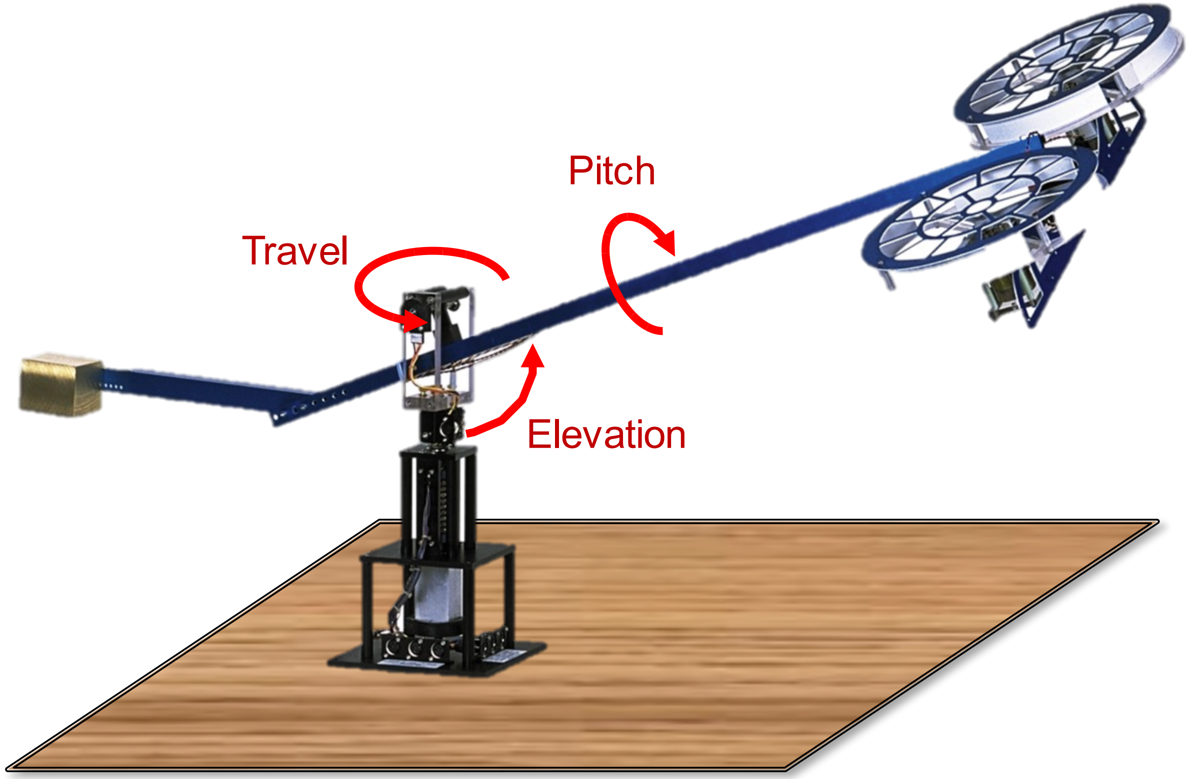
Safe Restart Time I

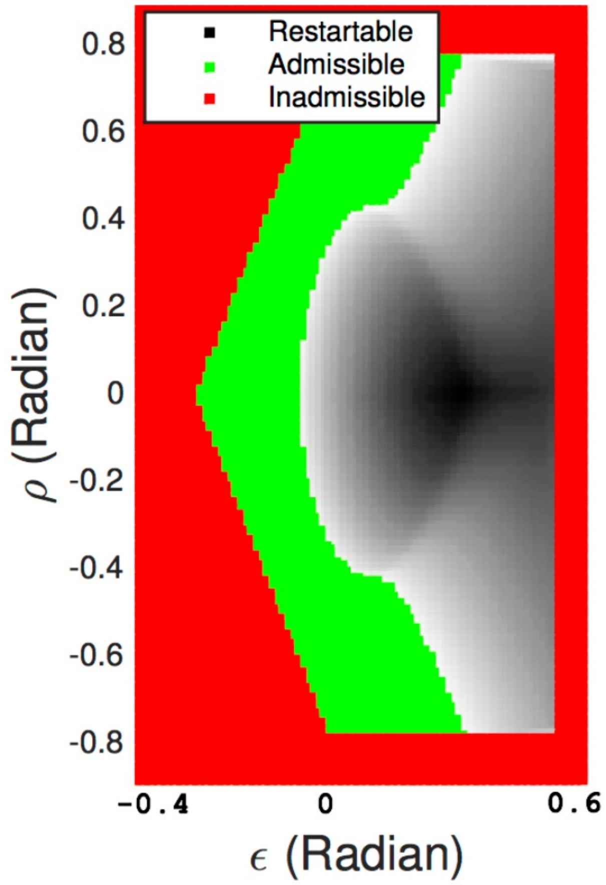




(a) Projection of the state space into the plane $\dot{\epsilon} = 0$, $\dot{\rho} = 0$, $\lambda = 0$, and $\dot{\lambda} = 0.3 Radian/s$

Safe Restart Time II

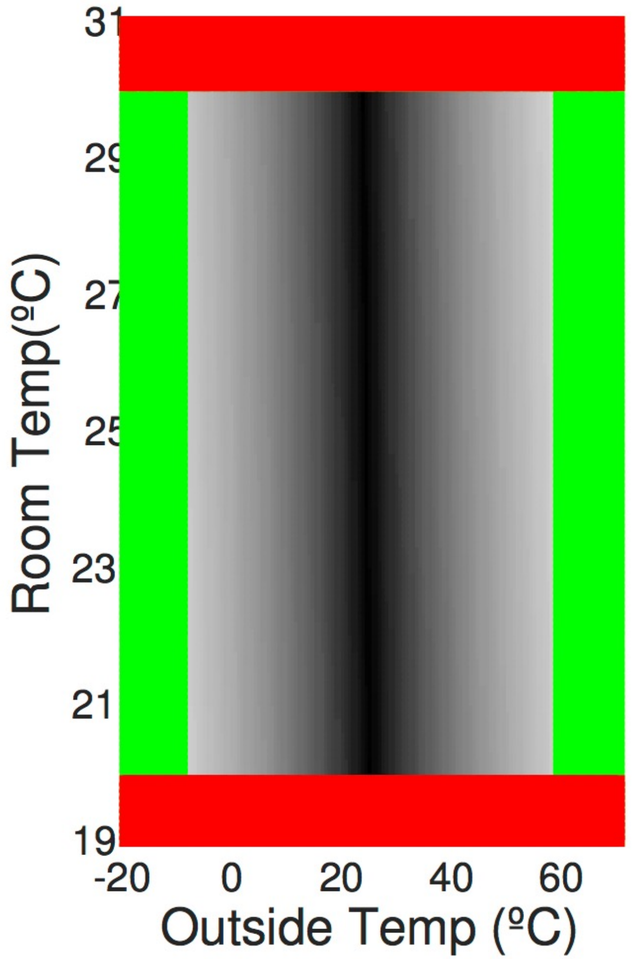




(b) Projection of the state space into the plane $\dot{\epsilon} = -0.3\text{Radian}/s$, $\dot{\rho} = 0$, $\lambda = 0$, and $\dot{\lambda} = 0.3\text{Radian}/s$

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Safe Restart Time III



(a) Projection of states to $T_F = 25^oC$

Impact on Availability of Controller

	Regions (From most common to least Common)	Avail.
Temperature Control System	$15 < T_O < 40$	%99.9
	$0 < T_O < 15$ or $40 < T_O < 60$	
	$T_O < 0$ or $60 < T_O$	
3DOF Helicopter	$-\epsilon + \|\rho\| < 0.1$ & $\epsilon < 0.2$ & $\|\rho\| < \pi/8$	%64.3
	$0.2 < -\epsilon + \|\rho\| < 0.1$ & $0.2 < \epsilon < 0.3$ & $\pi/8 < \|\rho\| < \pi/6$	
	$-\epsilon + \|\rho\| < 0.2$ & $0.3 < \epsilon$ & $\pi/6 < \|\rho\|$	

Thank you!

abditag2@illinois.edu

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

1: Start Safety Controller. */* SEI begins */*

2: `timeFound` := `False`

3: $\lambda_{\text{safe}} = \lambda_{\text{init}}$ /*Initializing the restart time*/

4: **while** `timeFound` == `False` **do**

5:     x := obtain the most recent state of the system from Sensors

6:     (`timeFound`, λ_{safe}) :=`FindRestartTime`(x, λ_{safe})

7: **end while**

8: Send λ_{safe} to RoT. */* Set the next restart time. */*

9: Activate external interfaces. */* SEI ends. */*

10: Terminate SC and start the main controller.

11: When RoT sends the restart signal to hardware restart pin:

12:     Restart the system

13:     Repeats the procedure from beginning (from Line 1)