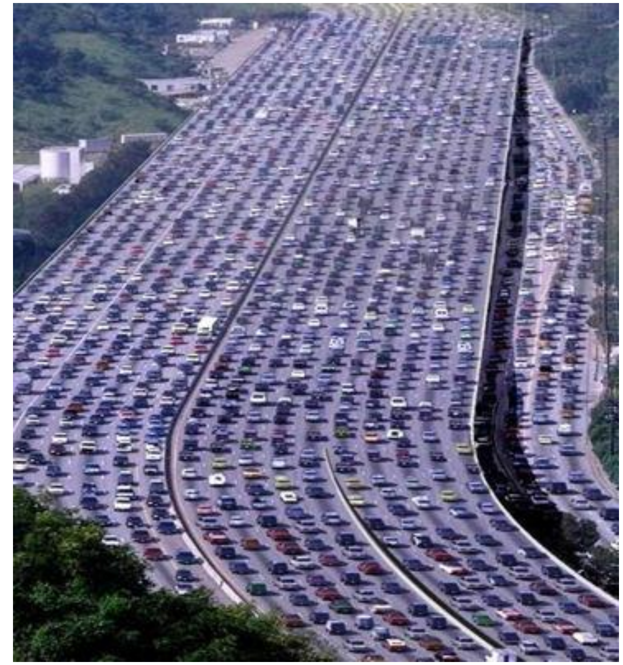
# Verifying Distributed Car and Aircraft Systems with Logic and Refinement

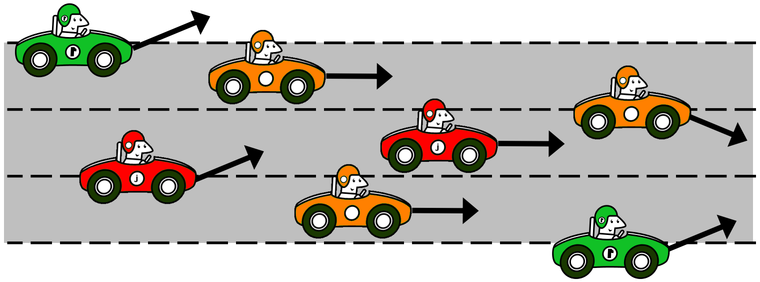
Sarah M. Loos
Computer Science Department
Carnegie Mellon University

Joint work with: Ligia Nistor, David Renshaw, Stefan Mitsch, Khalil Ghorbal, André Platzer

# Challenge: Cyber-Physical Systems

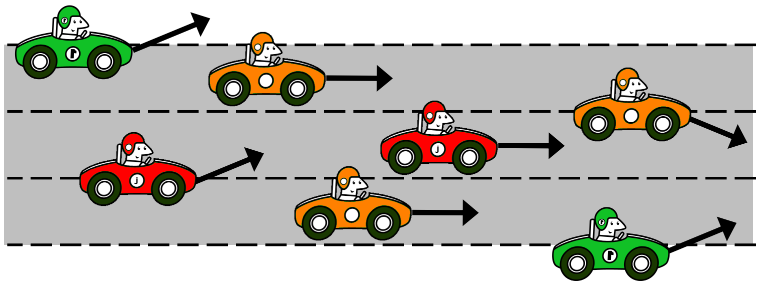
# Verified Cyber-Physical Systems

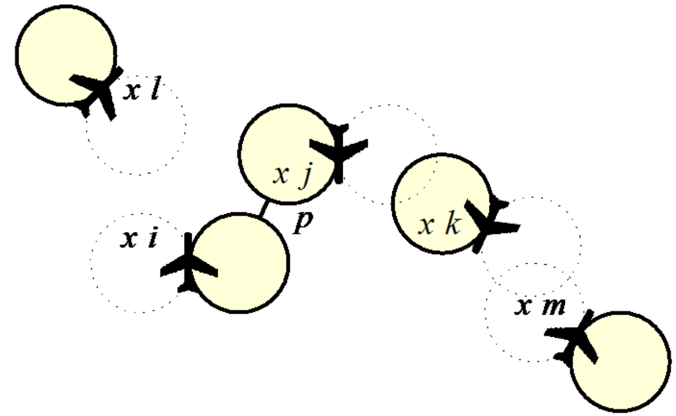




[FM11]

# Verified Cyber-Physical Systems

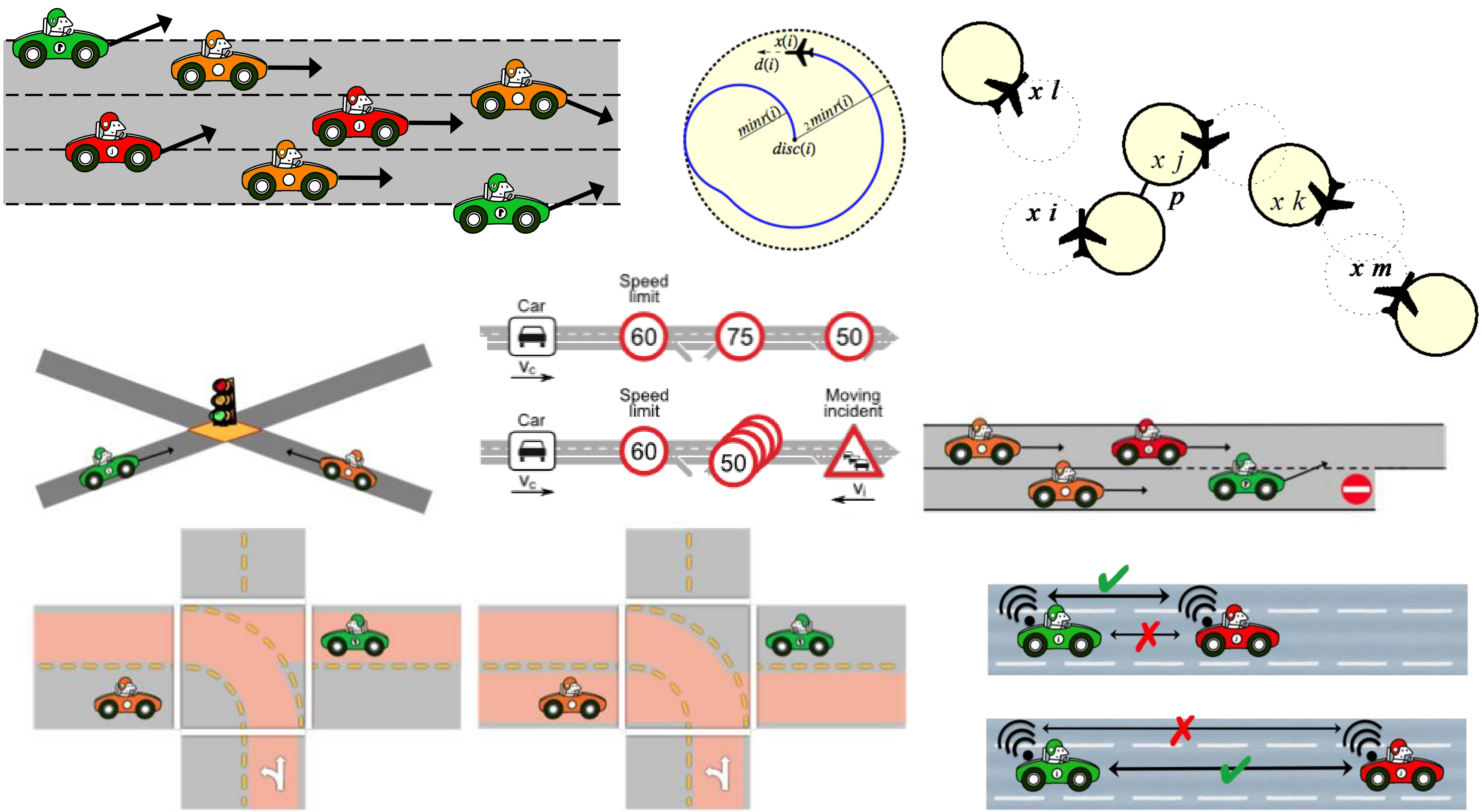







[FM11, HSCC13]

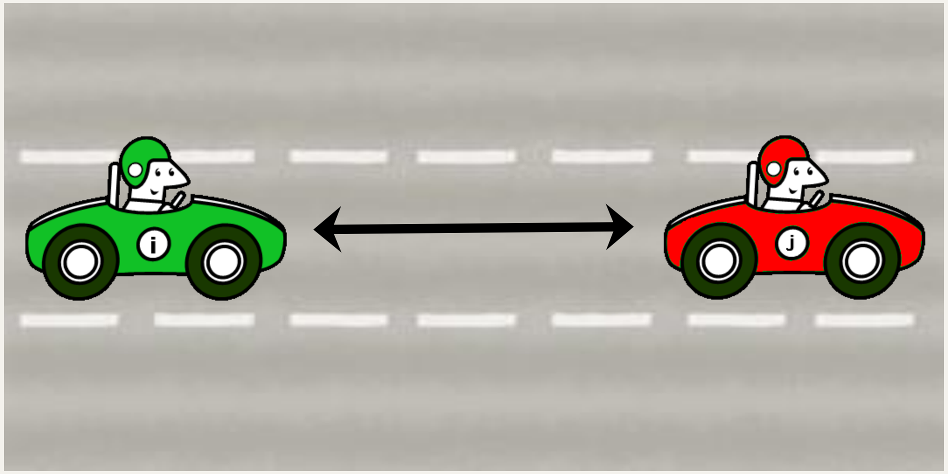
# Verified Cyber-Physical Systems



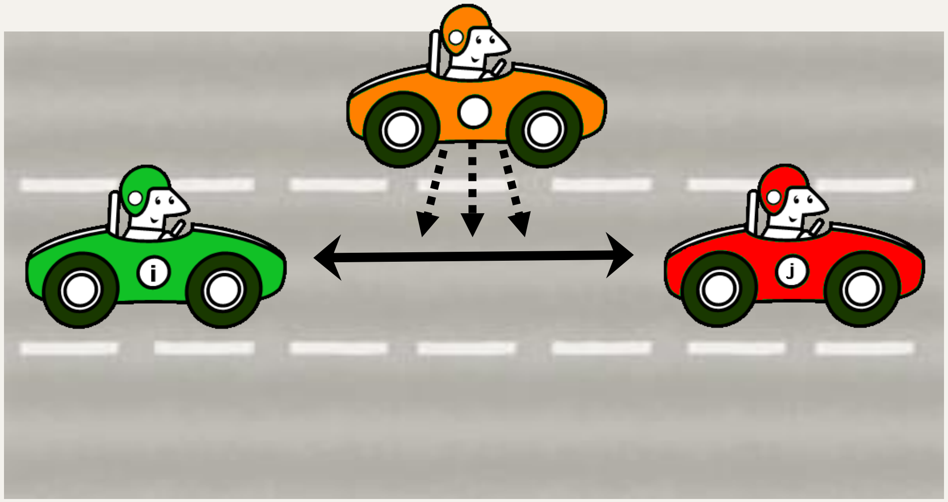



[FM11, ITSC11, ICCPS12, HSCC13, ITSC13]

# Distributed Car Control

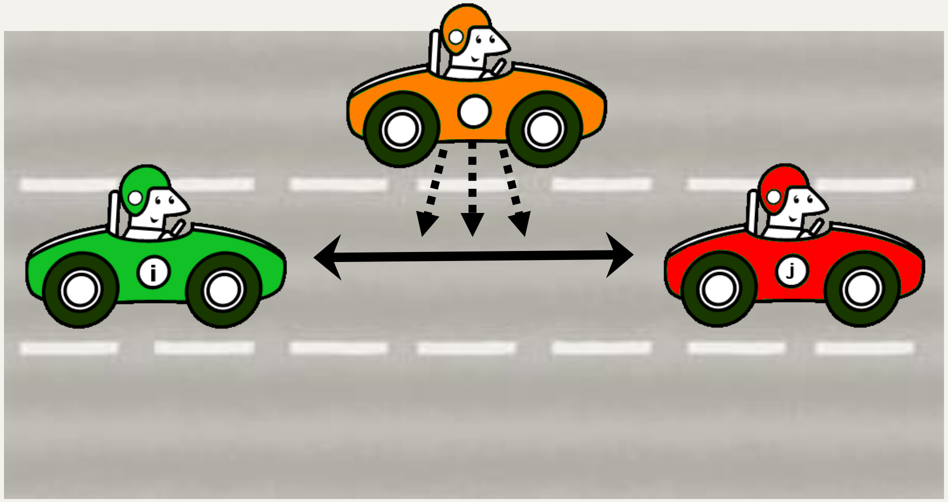


# Distributed Car Control




Sensor limits on actual cars are always local.

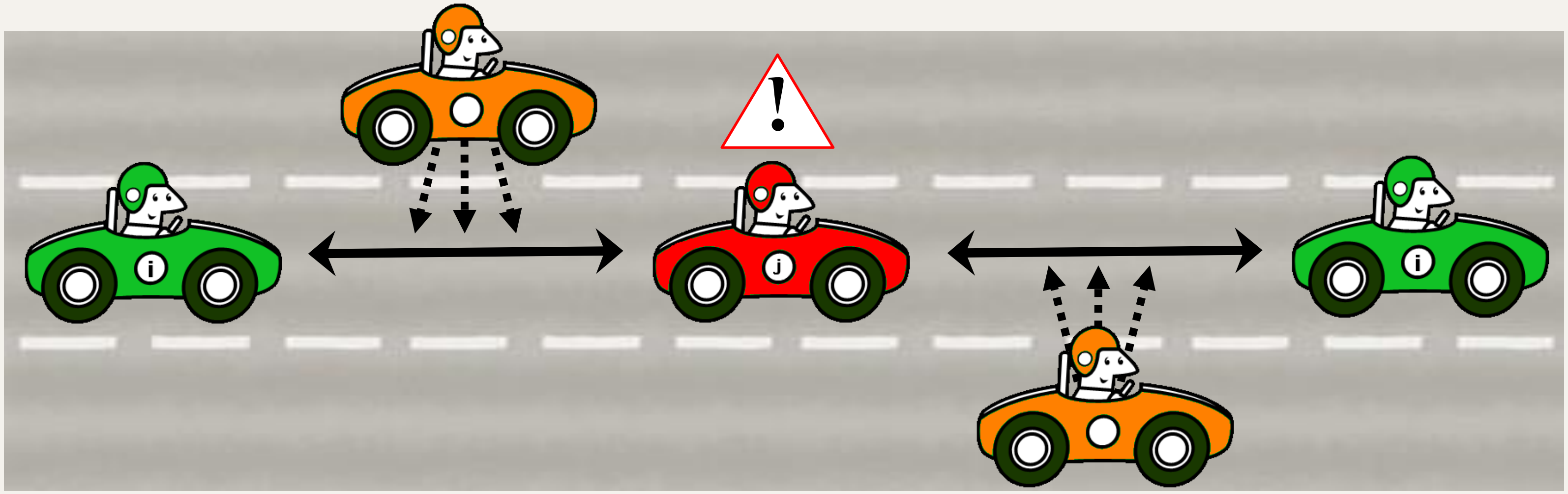
# Distributed Car Control




Sensor limits on actual cars are always local.

Sometimes a maneuver may look safe locally…

# Distributed Car Control



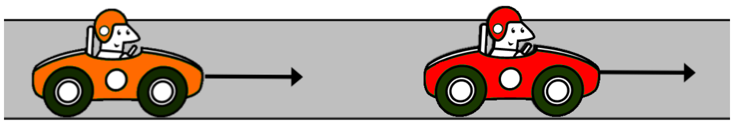

Sensor limits on actual cars are always local.

Sometimes a maneuver may look safe locally…

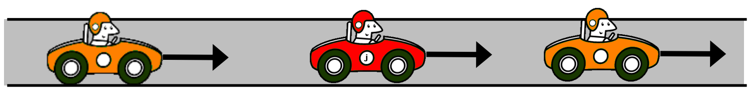
But is a terrible idea when implemented globally.

# Car Control: Proof Sketch

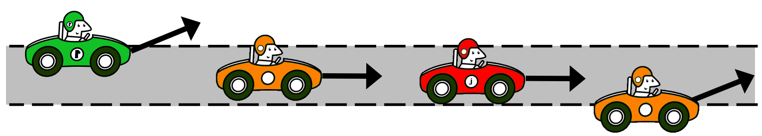
## Local Lane Control



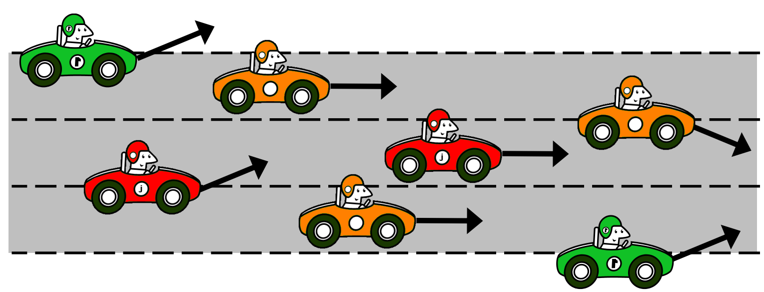
## Global Lane Control



## Local Highway Control



## Global Highway Control

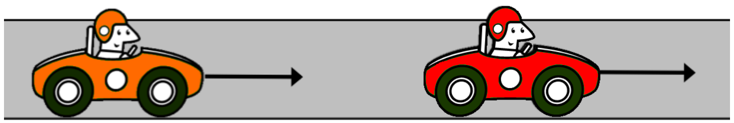




[FM11]

# Car Control: Proof Sketch

## Local Lane Control



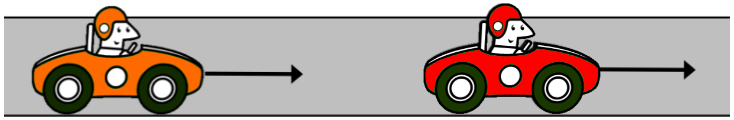
- 2 vehicles
- 1 lane
- no lane change



[FM11]

# Car Control: Proof Sketch

## Local Lane Control



$$(a := \theta; x'' = a)^*$$

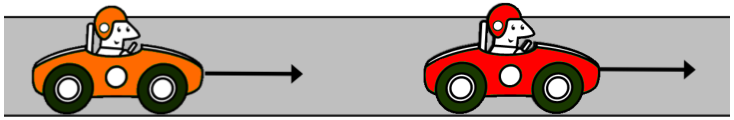
- 2 vehicles
- 1 lane
- no lane change



[FM11]

# Car Control: Proof Sketch

## Local Lane Control



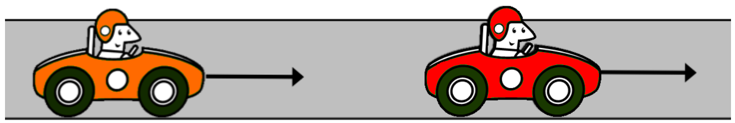
- 2 vehicles
- 1 lane
- no lane change



[FM11]

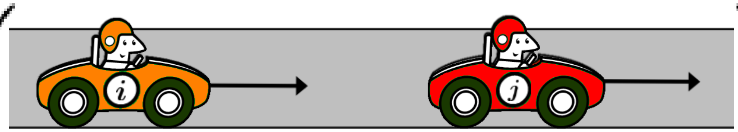
# Car Control: Proof Sketch

## Local Lane Control



- 2 vehicles
- 1 lane
- no lane change

## Global Lane Control

$\forall i, j : C\Big($  $\Big)$



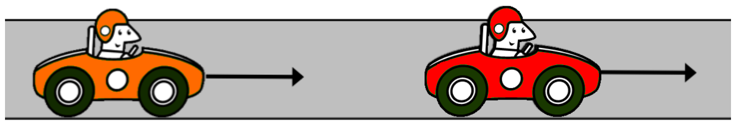
- **n vehicles**
- 1 lane
- no lane change



[FM11]

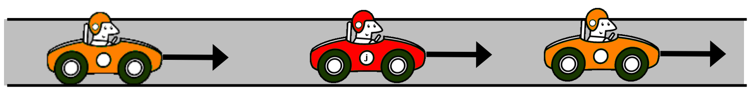
# Car Control: Proof Sketch

## Local Lane Control



- 2 vehicles
- 1 lane
- no lane change

## Global Lane Control



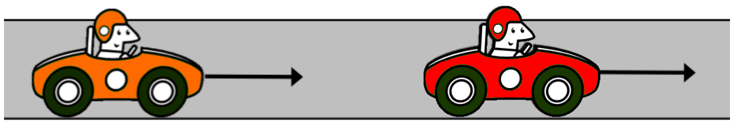
- **n vehicles**
- 1 lane
- no lane change



[FM11]

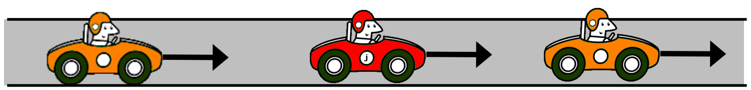
# Car Control: Proof Sketch

## Local Lane Control



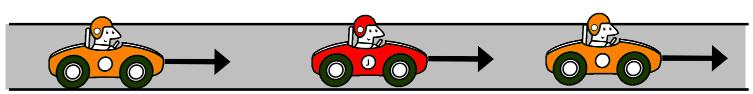
- 2 vehicles
- 1 lane
- no lane change

## Global Lane Control



- **n vehicles**
- 1 lane
- no lane change

## Local Highway Control

$\Big( delete^*; create^*;$  $\Big)^*$

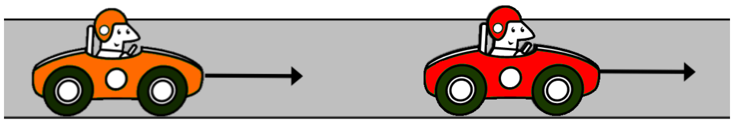
- n vehicles
- 1 lane
- **lane changes**



[FM11]

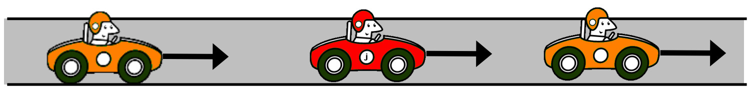
# Car Control: Proof Sketch

## Local Lane Control



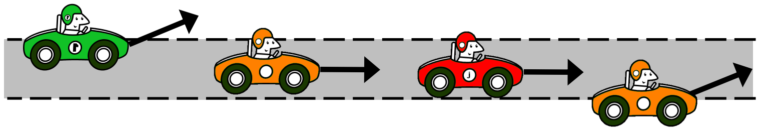
- 2 vehicles
- 1 lane
- no lane change

## Global Lane Control



- **n vehicles**
- 1 lane
- no lane change

## Local Highway Control



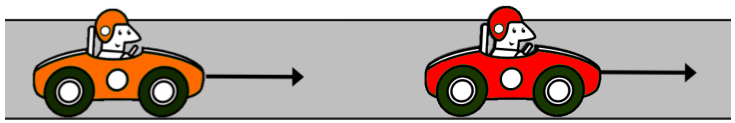
- n vehicles
- 1 lane
- **lane changes**



[FM11]

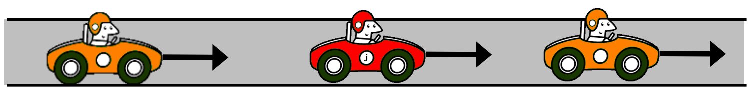
# Car Control: Proof Sketch

## Local Lane Control



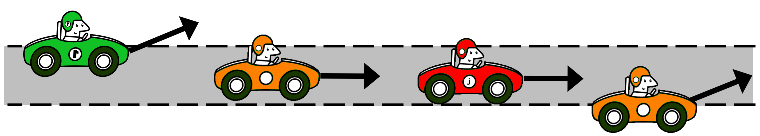
- 2 vehicles
- 1 lane
- no lane change

## Global Lane Control



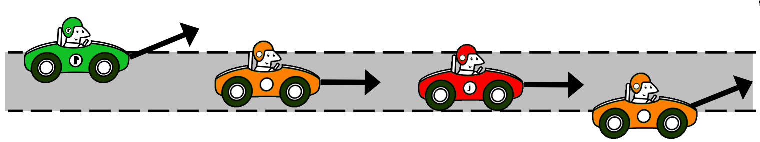
- **n vehicles**
- 1 lane
- no lane change

## Local Highway Control



- n vehicles
- 1 lane
- **lane changes**

## Global Highway Control

$\forall l : L\Bigg($  $\Bigg)$

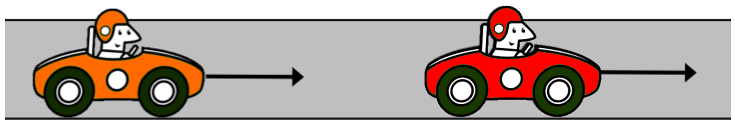
- n vehicles
- **m lanes**
- lane changes



[FM11]

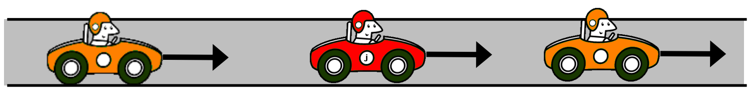
# Car Control: Proof Sketch

## Local Lane Control



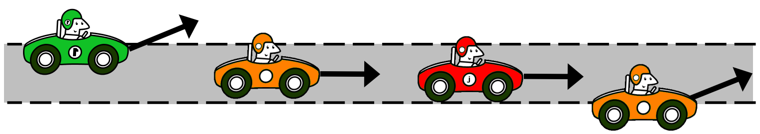
- 2 vehicles
- 1 lane
- no lane change

## Global Lane Control



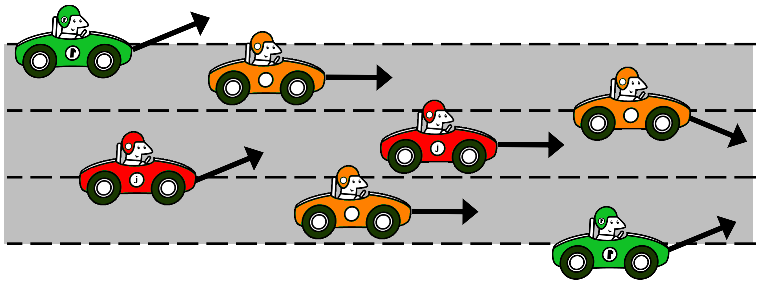
- **n vehicles**
- 1 lane
- no lane change

## Local Highway Control



- n vehicles
- 1 lane
- **lane changes**

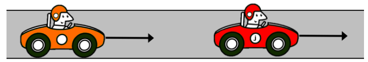
## Global Highway Control



- n vehicles
- **m lanes**
- lane changes

[FM11]

# Car Control: Proof



$$\forall i\; x(i) \ll x\big(L(i)\big) \to [glc]\; \forall i\; x(i) \ll x(L(i))$$

Transitivity

$$\forall i\; x(i) \ll x\big(L(i)\big) \to \;\forall i\; x(i) \ll x(L^*(i))$$

$$[glc]\; \forall i\; x(i) \ll x\big(L(i)\big) \to \;[glc]\; \forall i\; x(i) \ll x(L^*(i)) \quad \textbf{([] gen)}$$

Transitivity

$$\forall i\; x(i) \ll L\big(x(i)\big) \to [create^*]\forall i\; x(i) \ll L^*(x(i))$$

**(cut)**

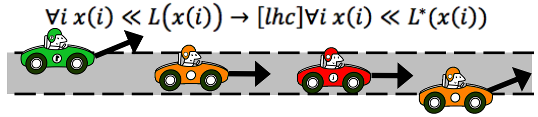
$$\forall i\; x(i) \ll x\big(L(i)\big) \to \;[glc]\; \forall i\; x(i) \ll x(L^*(i))$$

Transitivity

$$\forall i\; x(i) \ll L\big(x(i)\big) \to [delete^*]\forall i\; x(i) \ll L^*(x(i))$$

$$\forall i\; x(i) \ll L\big(x(i)\big) \to [delete^*][create^*][glc]\forall i\; x(i) \ll L^*(x(i)) \quad \textbf{([] split)}$$

$$\forall i\; x(i) \ll L\big(x(i)\big) \to [lhc]\forall i\; x(i) \ll L^*(x(i)) \quad \textbf{([;])}$$



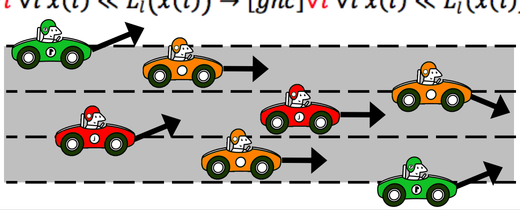
$$\forall i\; x(i) \ll L_l\big(x(i)\big) \to [lhc]\; \forall i\; x(i) \ll L_l^*\big(x(i)\big)$$

$$\forall l\; \Big(\forall i\; x(i) \ll L_l\big(x(i)\big) \to [lhc]\forall i\; x(i) \ll L_l^*\big(x(i)\big)\Big)$$

$$\forall l\; \forall i\; x(i) \ll L_l\big(x(i)\big) \to \forall l\; [lhc]\forall i\; x(i) \ll L_l^*\big(x(i)\big)$$

$$\forall l\; \forall i\; x(i) \ll L_l\big(x(i)\big) \to [\forall l\; (lhc)\;\;]\forall l\; \forall i\; x(i) \ll L_l^*\big(x(i)\big)$$

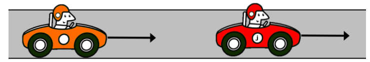
$$\forall l\; \forall i\; x(i) \ll L_l\big(x(i)\big) \to [ghc]\forall l\; \forall i\; x(i) \ll L_l^*\big(x(i)\big)$$





[FM11]

# Car Control: Proof



$$\forall i\, x(i) \ll x(L(i)) \rightarrow [glc]\, \forall i\, x(i) \ll x(L(i))$$

Transitivity

$$\forall i\, x(i) \ll x(L(i)) \rightarrow \forall i\, x(i) \ll x(L^*(i))$$

([] gen)

$$[glc]\, \forall i\, x(i) \ll x(L(i)) \rightarrow [glc]\, \forall i\, x(i) \ll x(L^*(i))$$

Transitivity

$$\forall i\, x(i) \ll L(x(i)) \rightarrow [create^*]\forall i\, x(i) \ll L^*(x(i))$$

(cut)

$$\forall i\, x(i) \ll x(L(i)) \rightarrow [glc]\, \forall i\, x(i) \ll x(L^*(i))$$

Transitivity

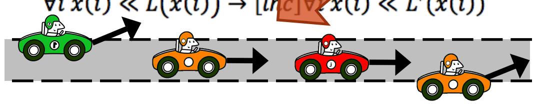
$$\forall i\, x(i) \ll L(x(i)) \rightarrow [delete^*]\forall i\, x(i) \ll L^*(x(i))$$

([] split)

$$\forall i\, x(i) \ll L(x(i)) \rightarrow [delete^*][create^*][glc]\forall i\, x(i) \ll L^*(x(i))$$

([;])

$$\forall i\, x(i) \ll L(x(i)) \rightarrow [lhc]\forall i\, x(i) \ll L^*(x(i))$$



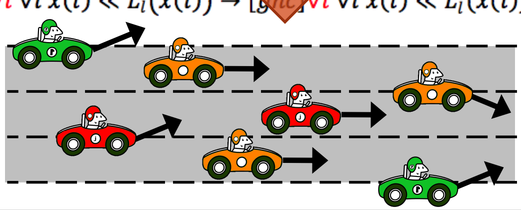
$$\forall i\, x(i) \ll L_l(x(i)) \rightarrow [lhc]\, \forall i\, x(i) \ll L_l^*(x(i))$$

$$\forall l\, \left(\forall i\, x(i) \ll L_l(x(i)) \rightarrow [lhc]\forall i\, x(i) \ll L_l^*(x(i))\right)$$

$$\forall l\, \forall i\, x(i) \ll L_l(x(i)) \rightarrow \forall l\, [lhc]\forall i\, x(i) \ll L_l^*(x(i))$$

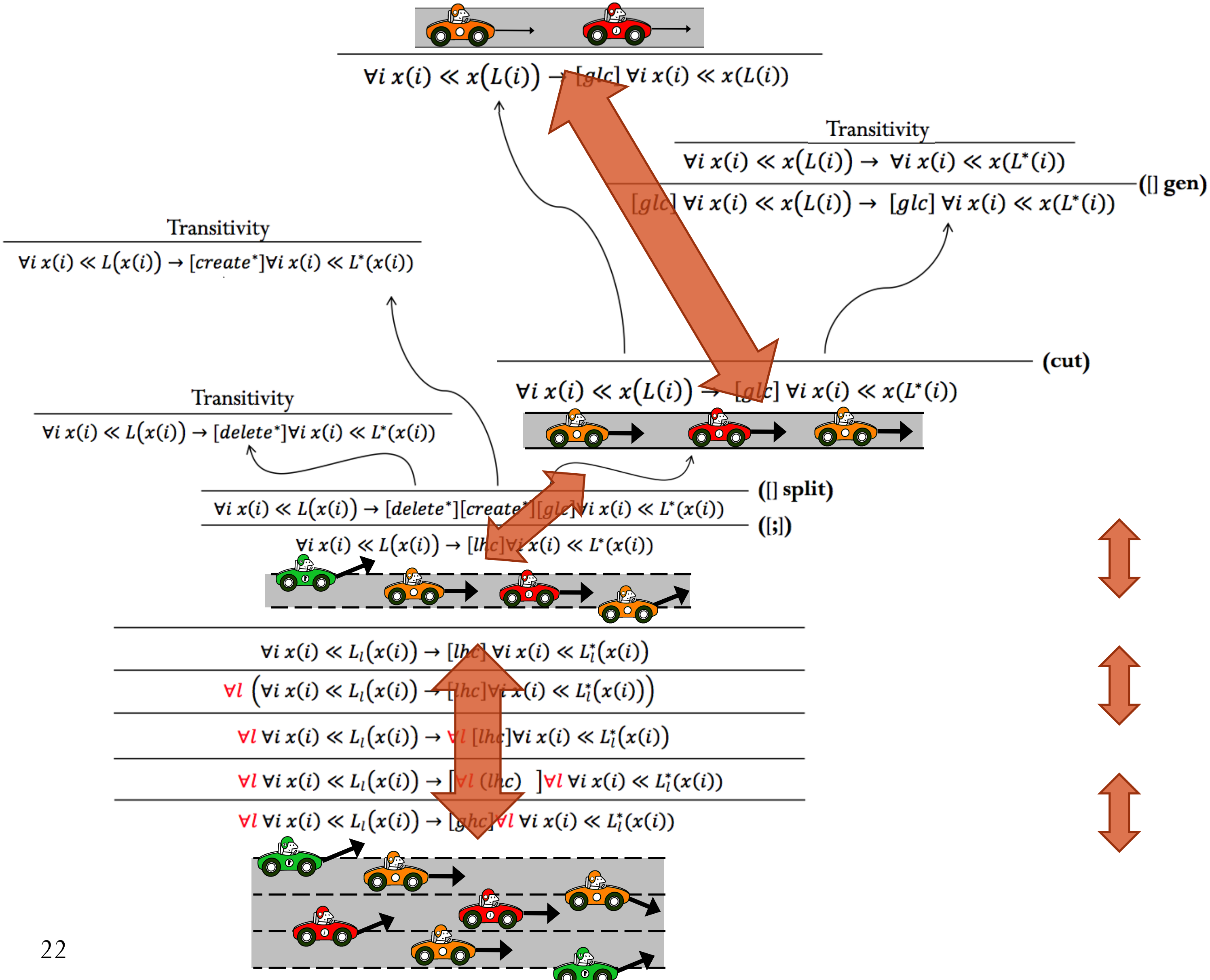
$$\forall l\, \forall i\, x(i) \ll L_l(x(i)) \rightarrow [\forall l\, (lhc)\,]\forall l\, \forall i\, x(i) \ll L_l^*(x(i))$$

$$\forall l\, \forall i\, x(i) \ll L_l(x(i)) \rightarrow [ghc]\forall l\, \forall i\, x(i) \ll L_l^*(x(i))$$

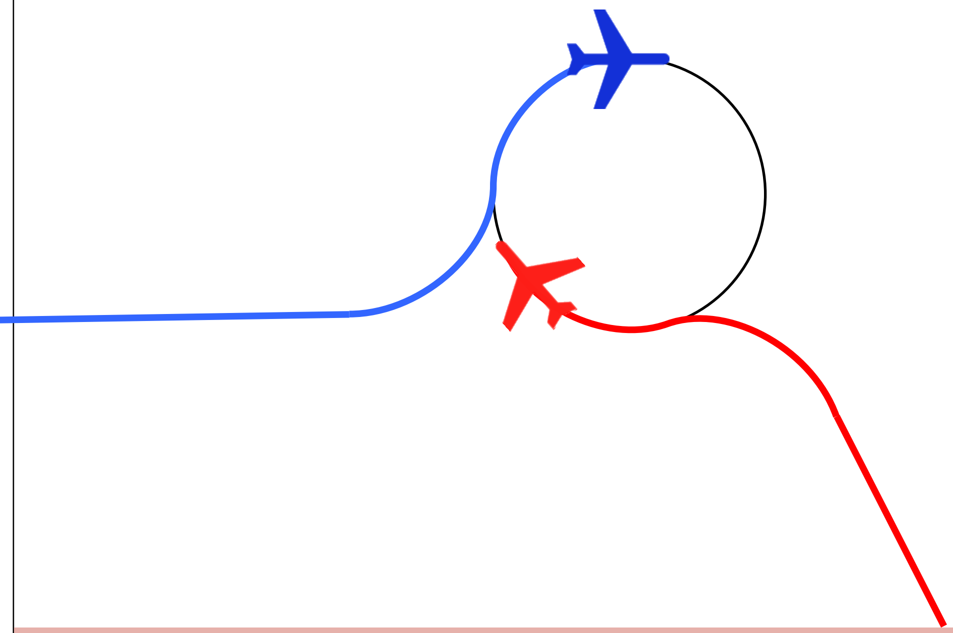


[FM11]

# Car Control: Proof

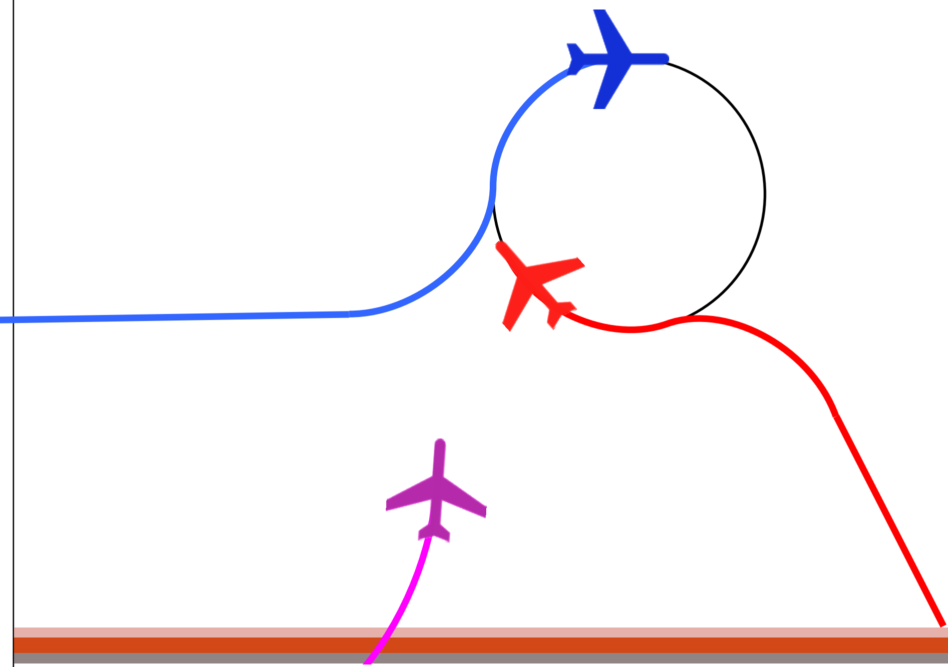


# How Can We Prove Distributed Airspace?



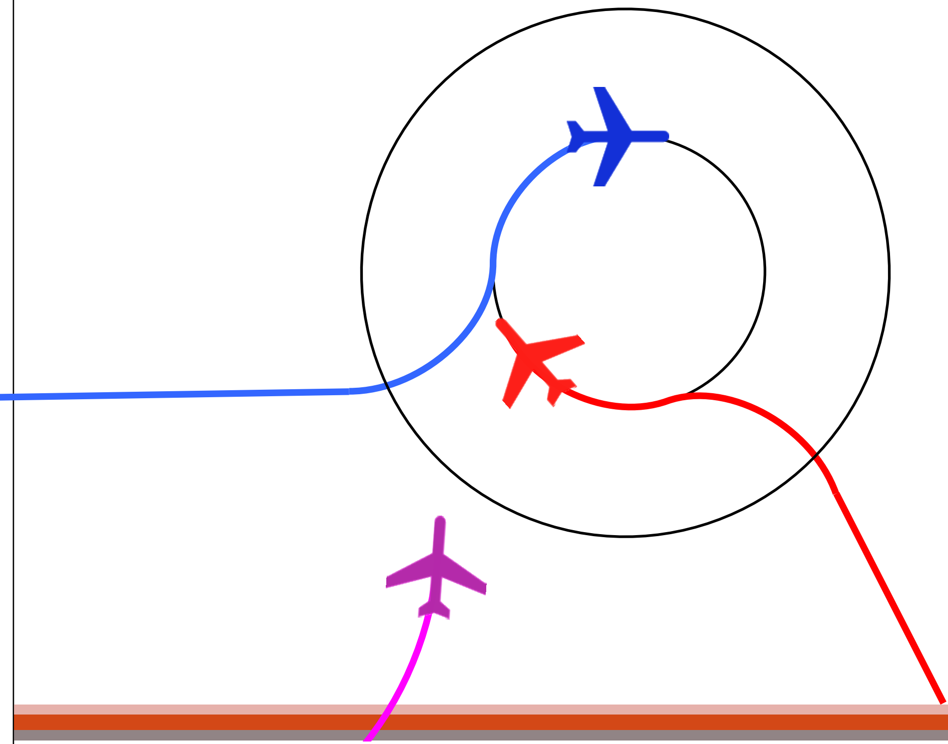
Sensor limits on aircraft are local.

# How Can We Prove Distributed Airspace?



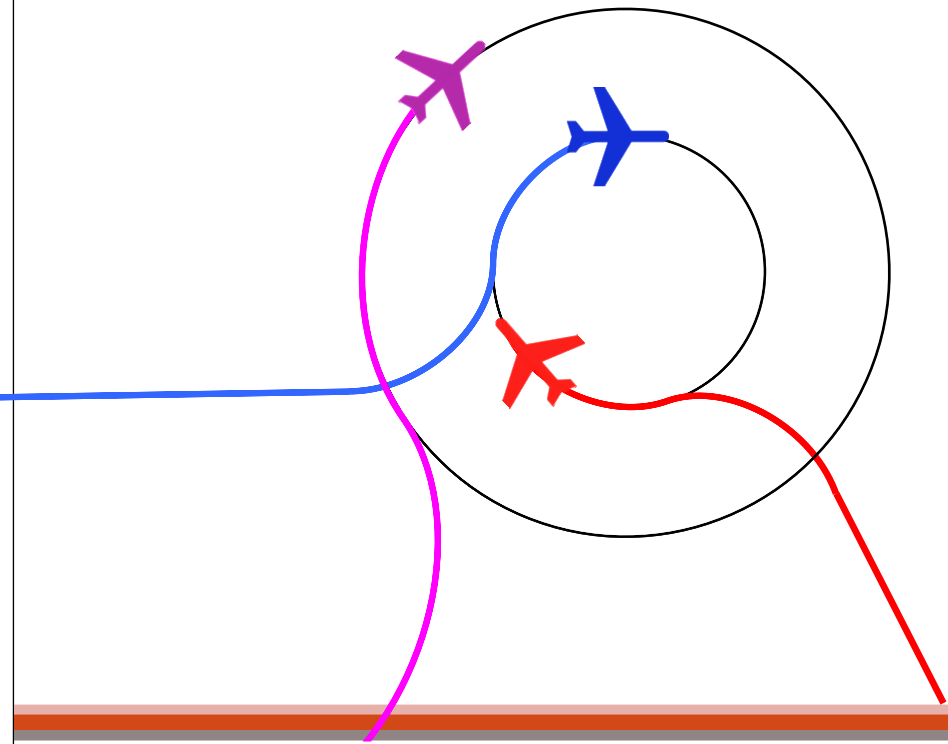
Sensor limits on aircraft are local.

# How Can We Prove Distributed Airspace?



Sensor limits on aircraft are local.

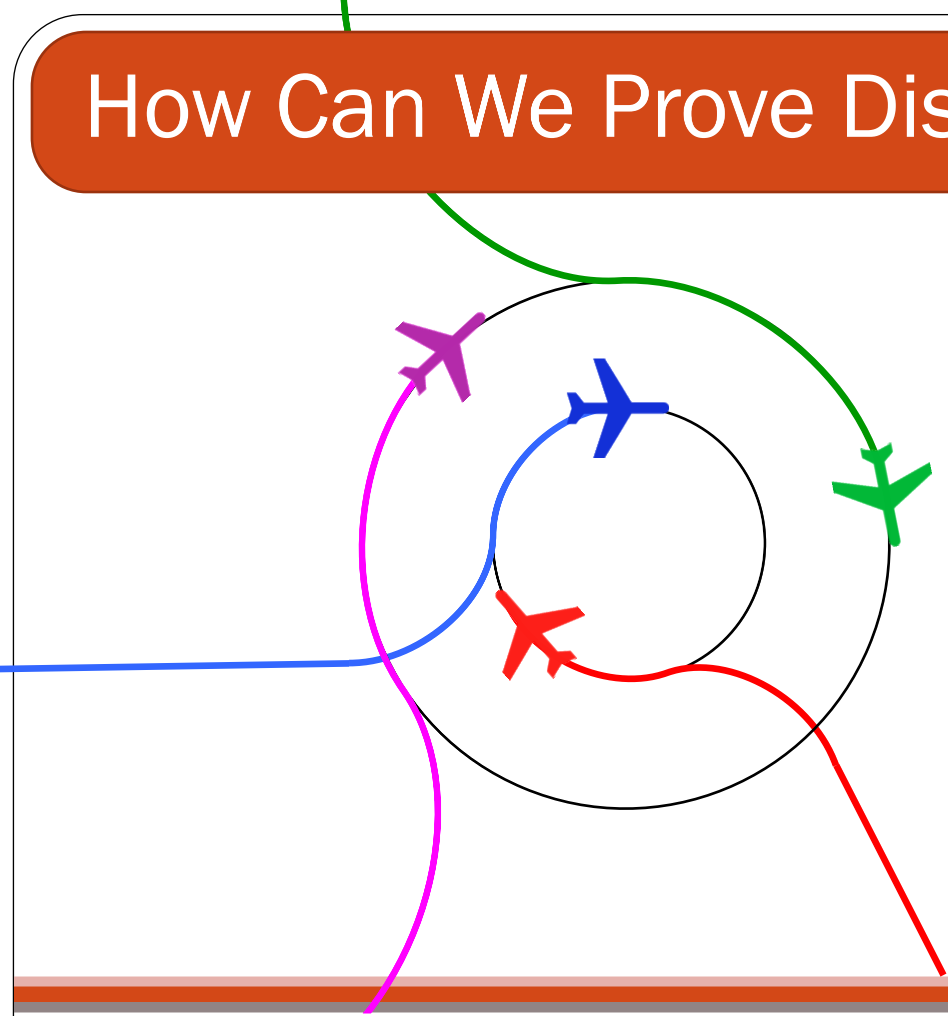
# How Can We Prove Distributed Airspace?



Sensor limits on aircraft are local.

Sometimes a maneuver may look safe locally…

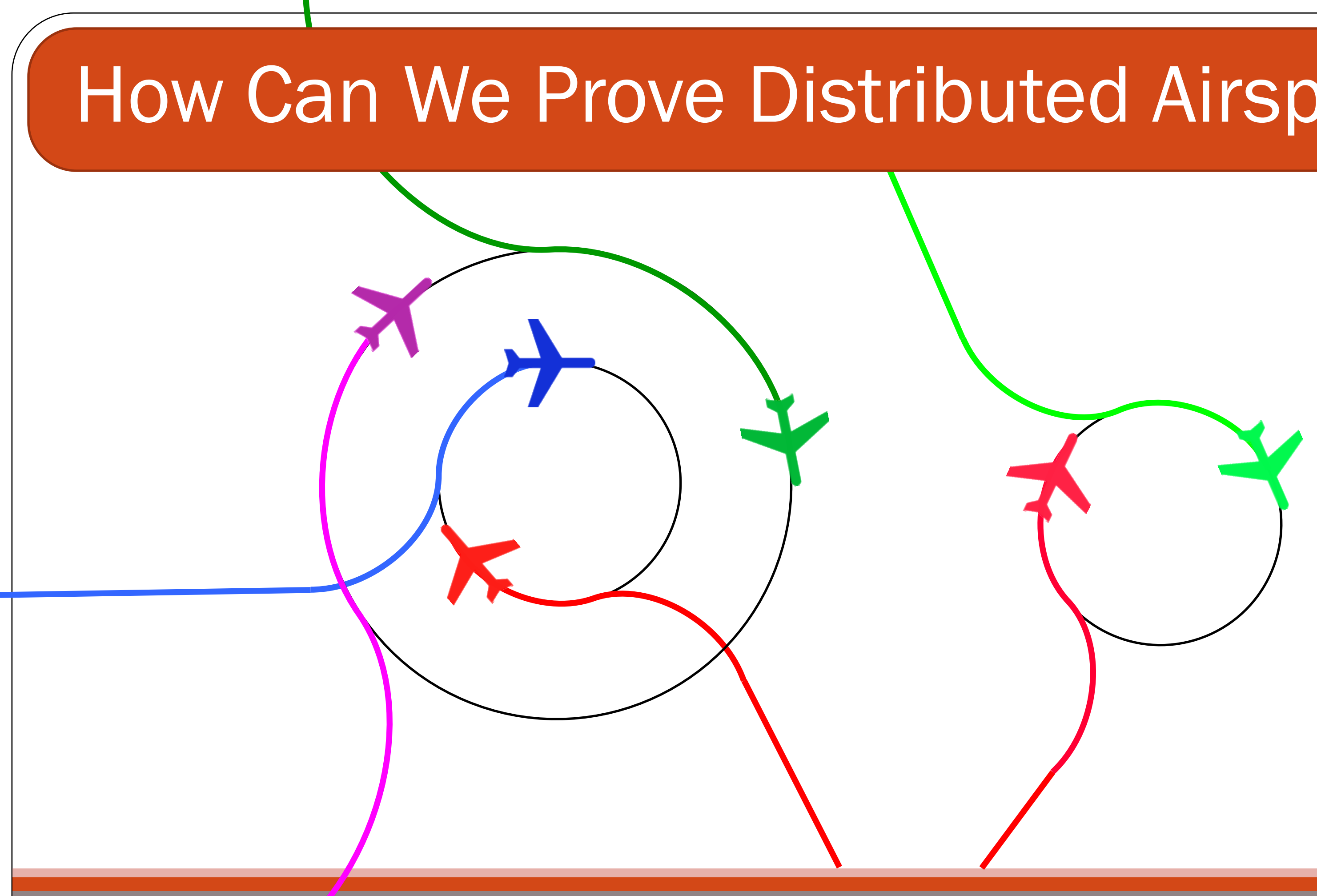
# How Can We Prove Distributed Airspace?



Sensor limits on aircraft are local.

Sometimes a maneuver may look safe locally...

# How Can We Prove Distributed Airspace?

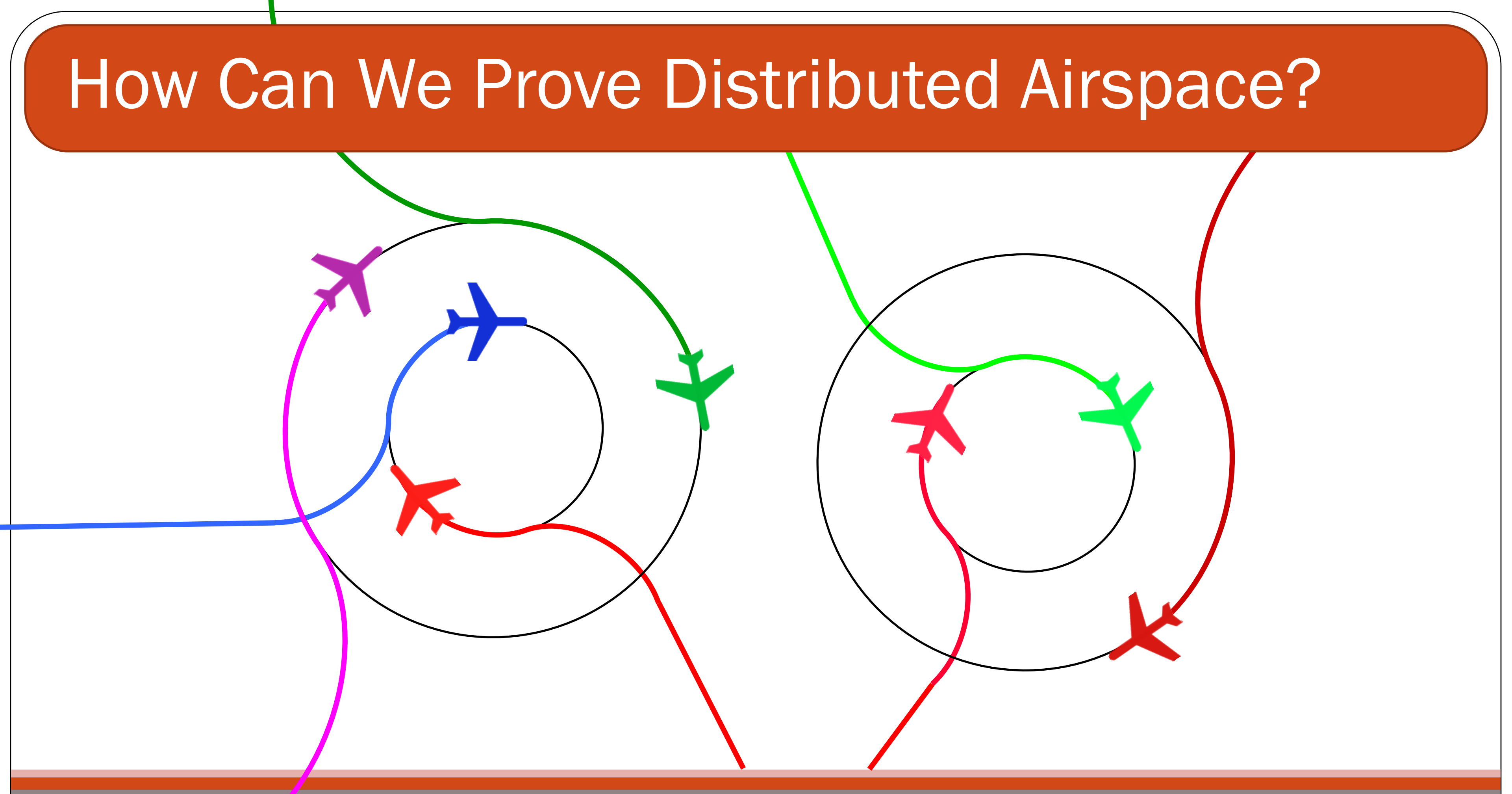


Sensor limits on aircraft are local.

Sometimes a maneuver may look safe locally…

But is a terrible idea when implemented globally.

# How Can We Prove Distributed Airspace?

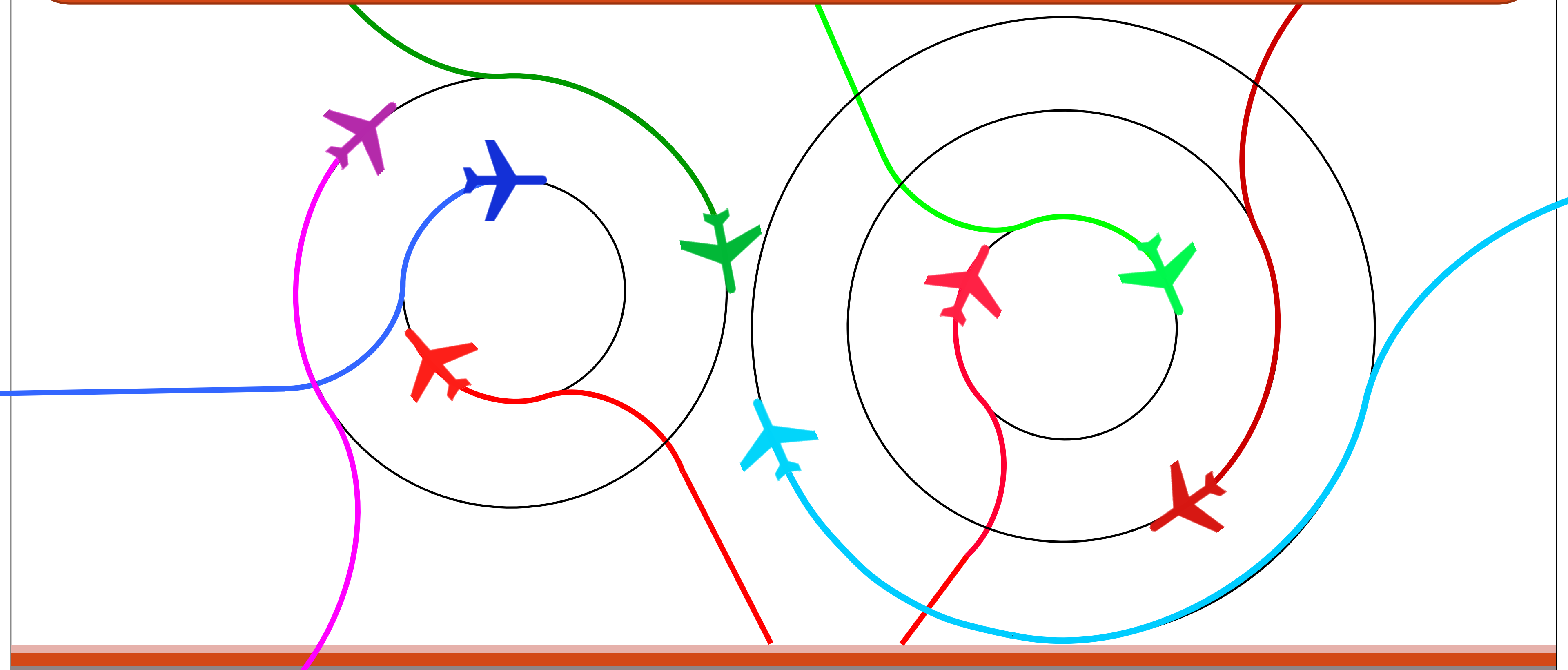


Sensor limits on aircraft are local.

Sometimes a maneuver may look safe locally…

But is a terrible idea when implemented globally.

# How Can We Prove Distributed Airspace?

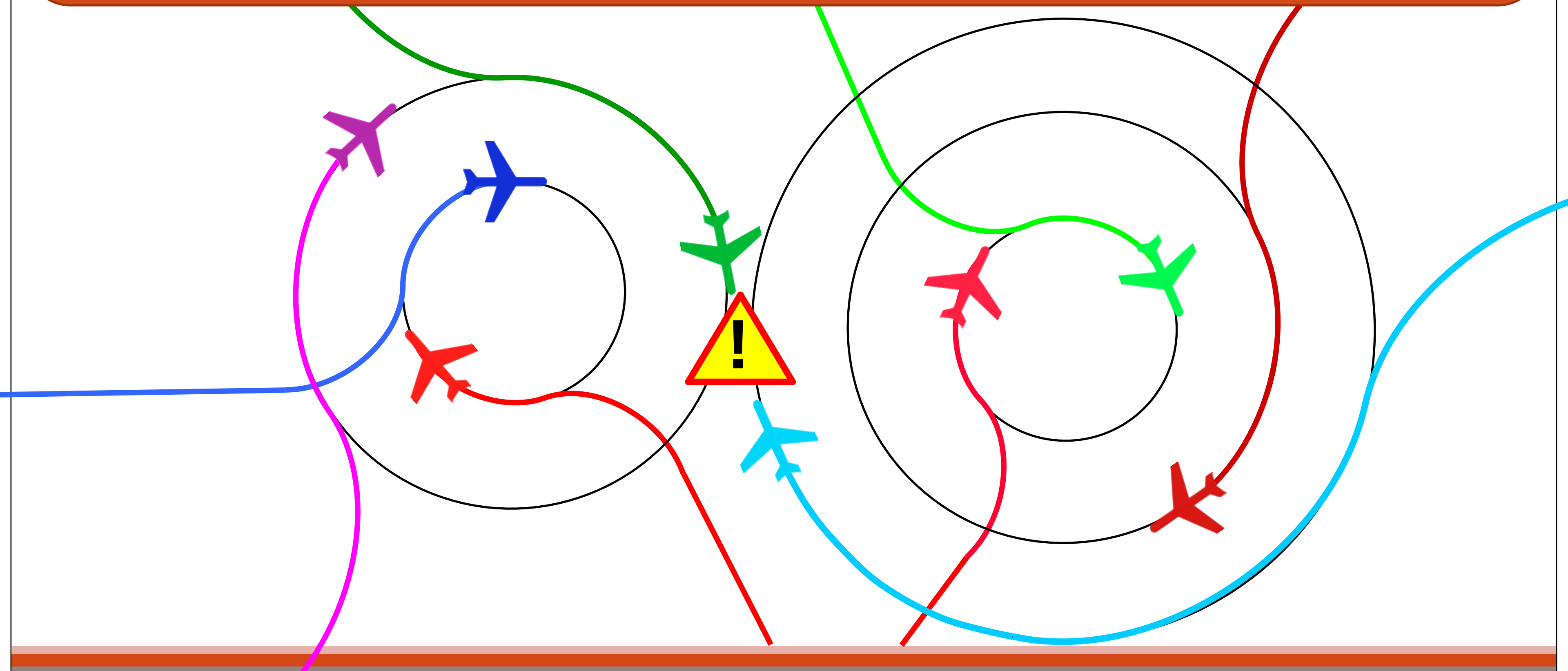


Sensor limits on aircraft are local.

Sometimes a maneuver may look safe locally…

But is a terrible idea when implemented globally.

# How Can We Prove Distributed Airspace?



Sensor limits on aircraft are local.

Sometimes a maneuver may look safe locally…

But is a terrible idea when implemented globally.

# Assumptions and Requirements

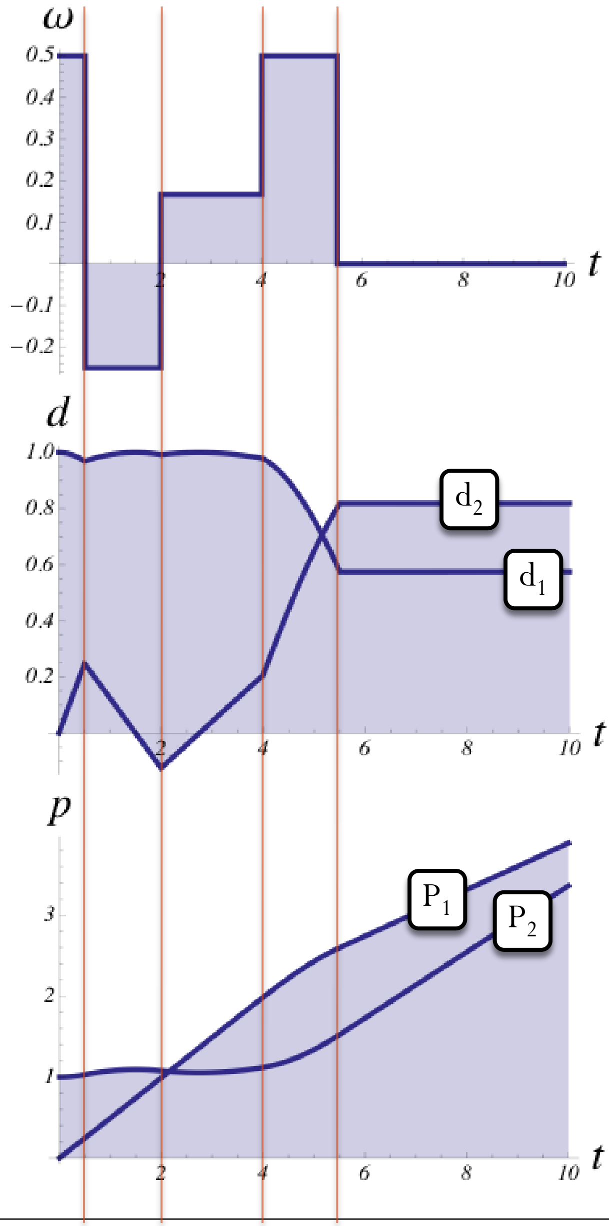
## Requirements

- **Safety**: At all times, the aircraft must be separated by distance greater than $p$.
- Aircraft trajectories must always be **flyable**.
- An **arbitrary number** of aircraft may enter the maneuver at any time.

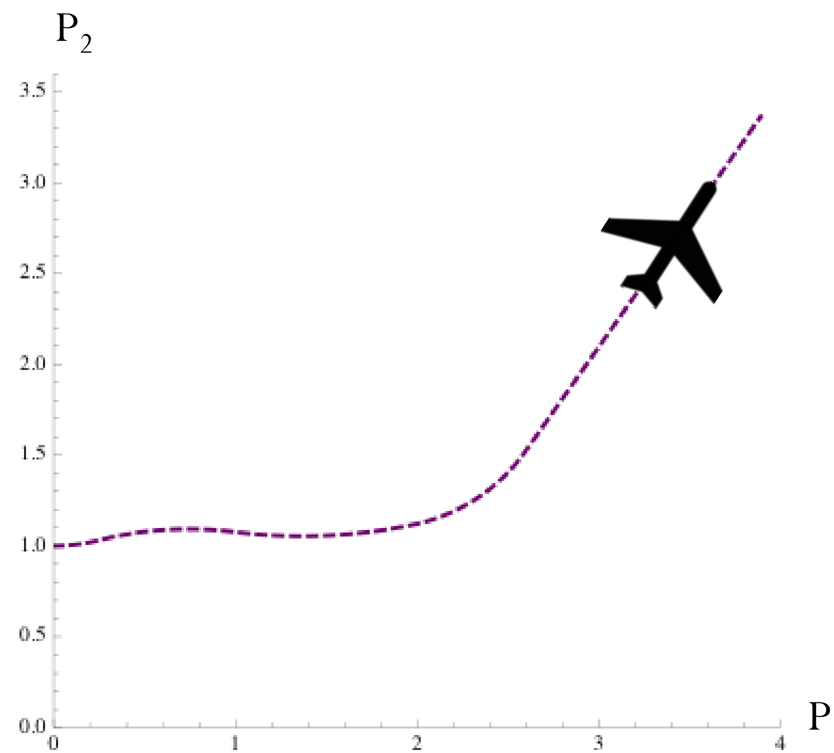
## Assumptions

- Aircraft maintain constant velocity.
- Sensors are accurate and have no delay.
- Collision avoidance maneuvers are executed on the 2D plane.

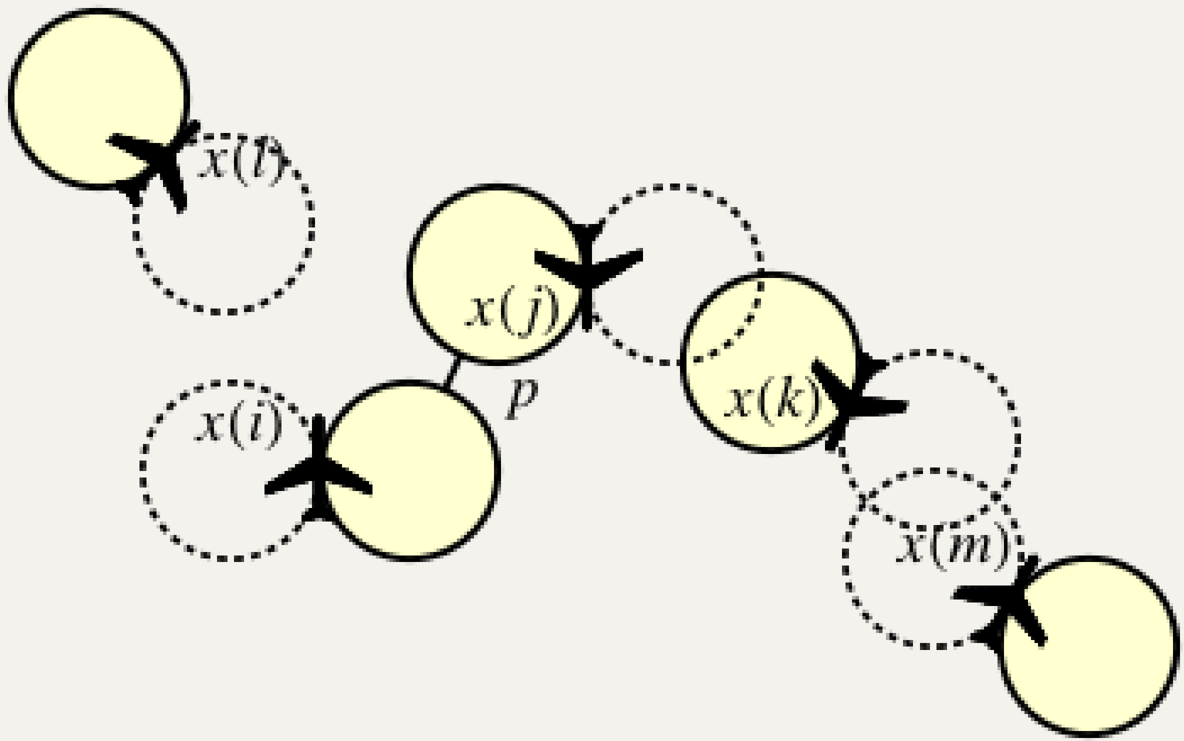
# Hybrid Dynamics




Aircraft are controlled by steering, through discrete changes in angular velocity $\omega$.



# Distributed Aircraft Control




- Each aircraft is associated with a buffer disc.
- The discs should never come within p of each other.
- Discs follow aircraft when *not in* collision avoidance.
- Each aircraft circles its stationary disc when *in* collision avoidance.

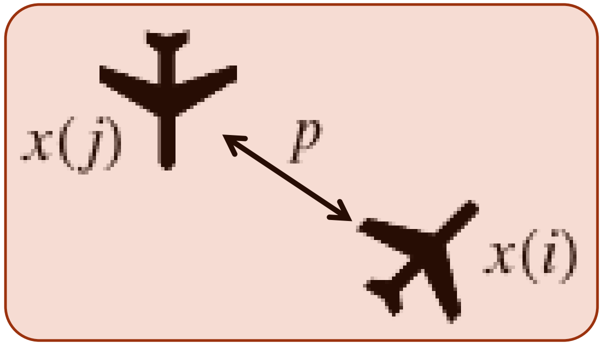


[PallottinoSBF07, LoosRP13]

# Modular Proof for Distributed Aircraft

## To Prove:

Safe separation of aircraft.



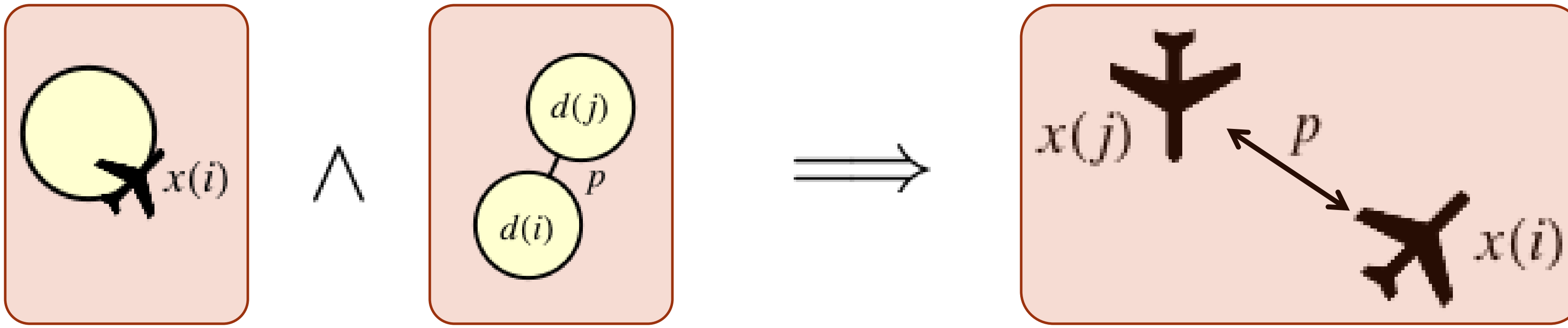

$$\forall i \neq j : A$$
$$\|x(i) - x(j)\| \geq p$$

[LoosRP13]

# Modular Proof for Distributed Aircraft

**To Prove:**

Safe separation of aircraft.



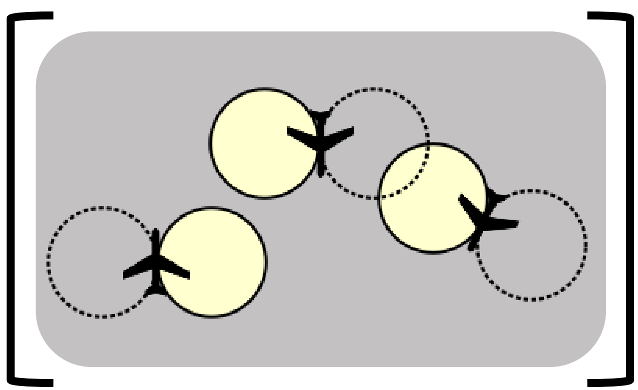

$$\forall i : A \quad \|x(i) - d(i)\| = r \;\wedge\; \forall i \neq j : A \quad \|d(i) - d(j)\| \geq 2r + p \implies \forall i \neq j : A \quad \|x(i) - x(j)\| \geq p$$

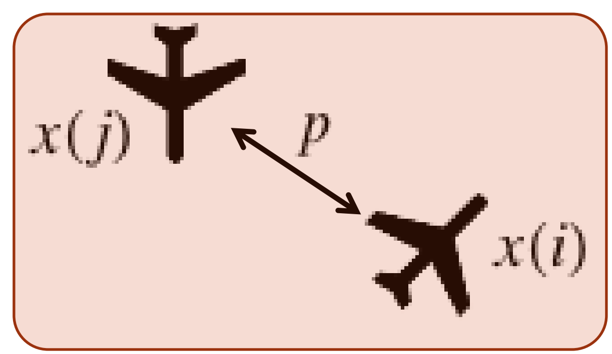
[LoosRP13]

# Modular Proof for Distributed Aircraft

Model

Safety Property



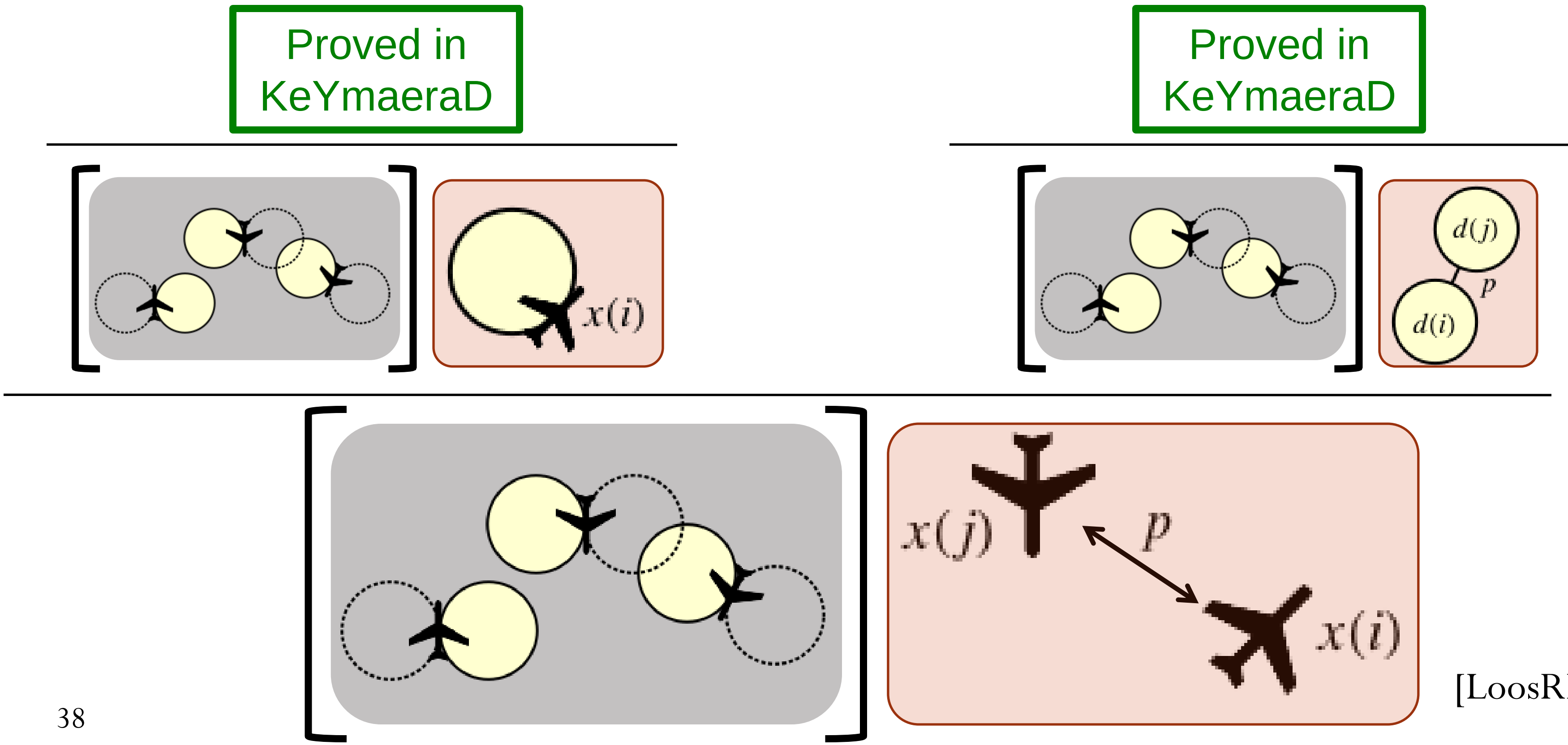



[LoosRP13]

# Modular Proof for Distributed Aircraft

Model

Safety Property



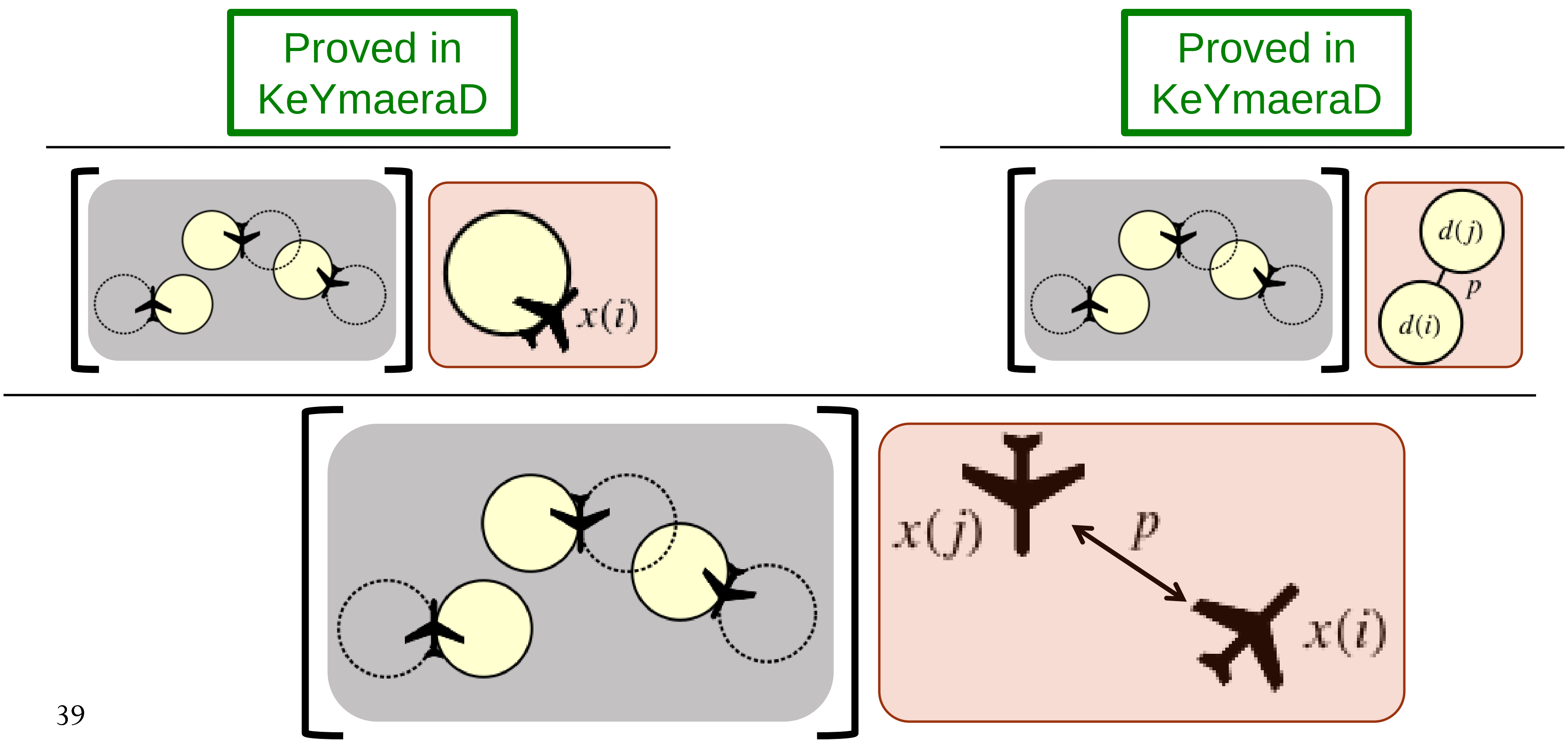

[LoosRP13]

# Modular Proof for Distributed Aircraft

Model

Safety Property

... but that isn't the end of the story.

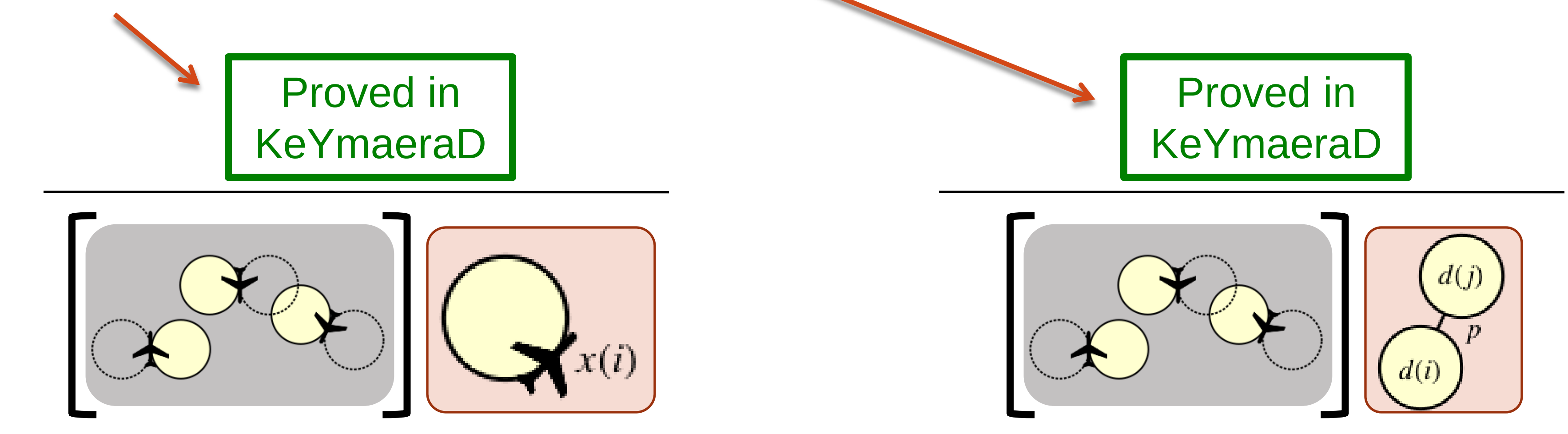


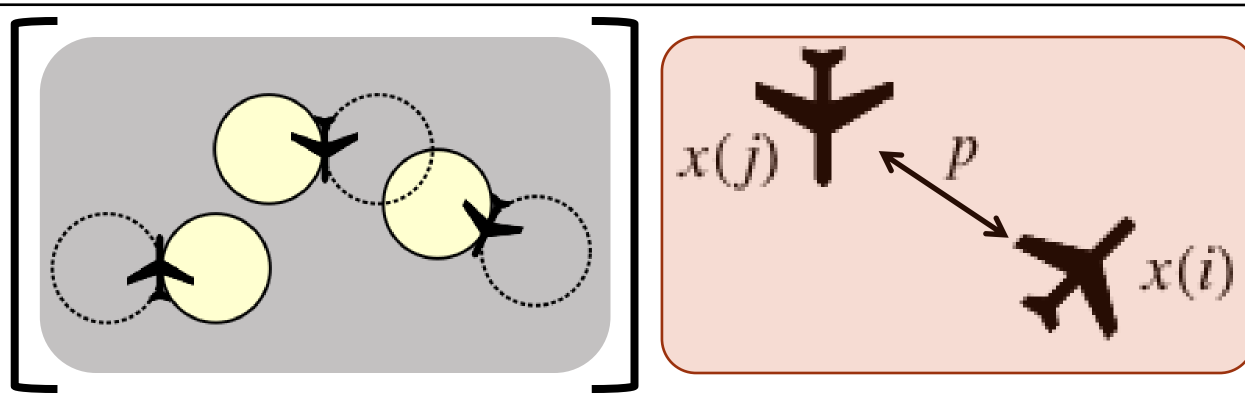
# Modular Proof for Distributed Aircraft

These proofs are hard.
Could we simplify them by changing the model in a sound way?

Model

Safety Property

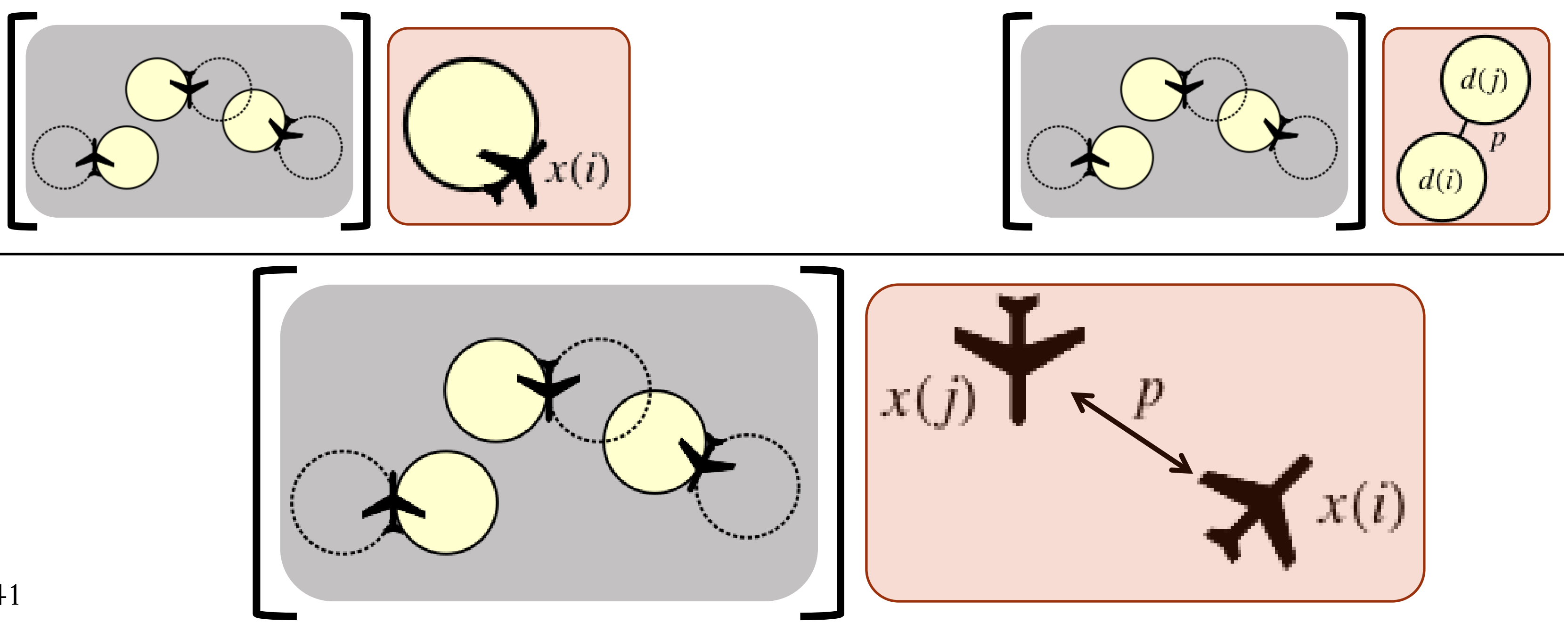


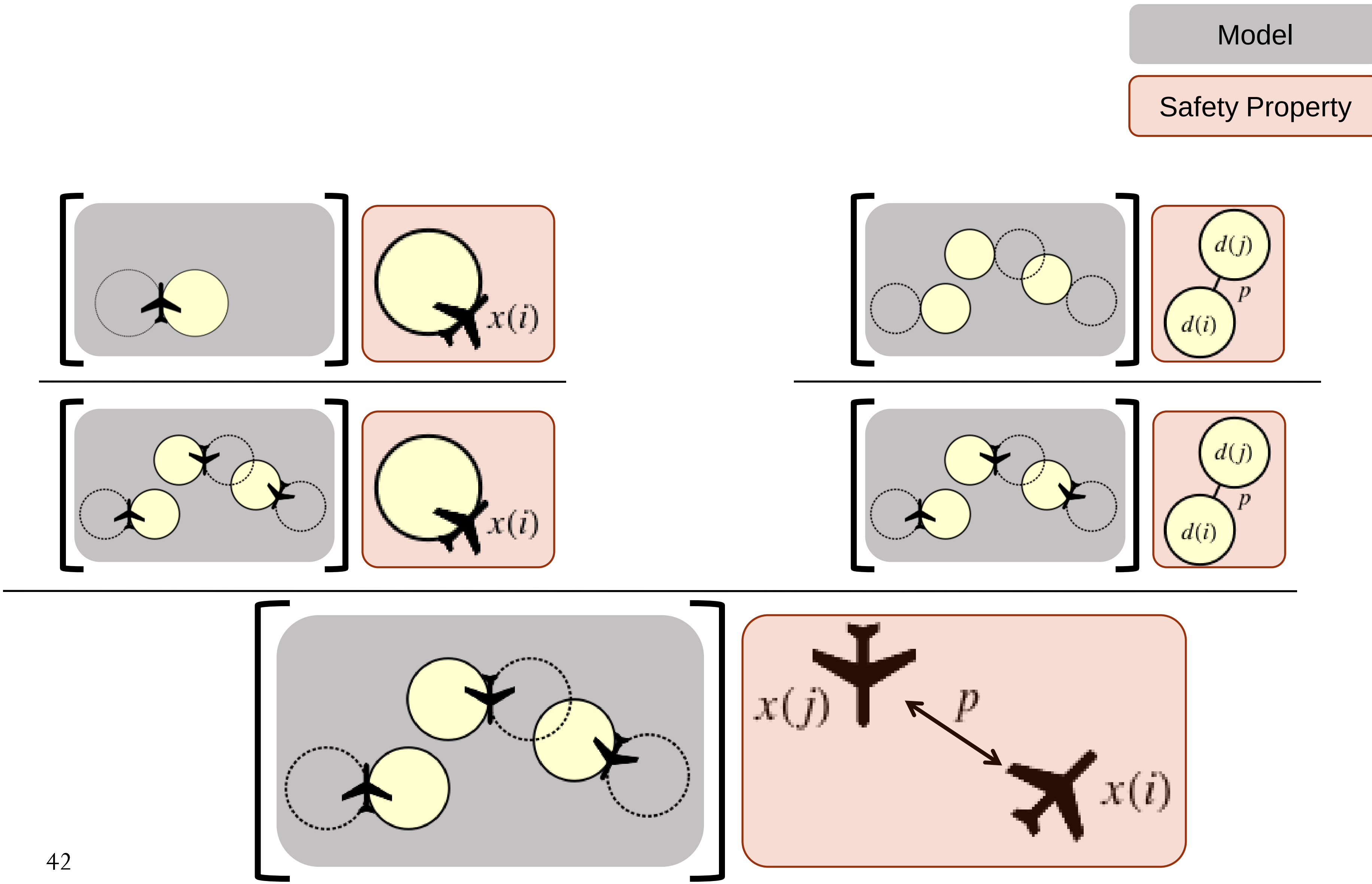
# Future Work for Distributed Aircraft

Model

Safety Property

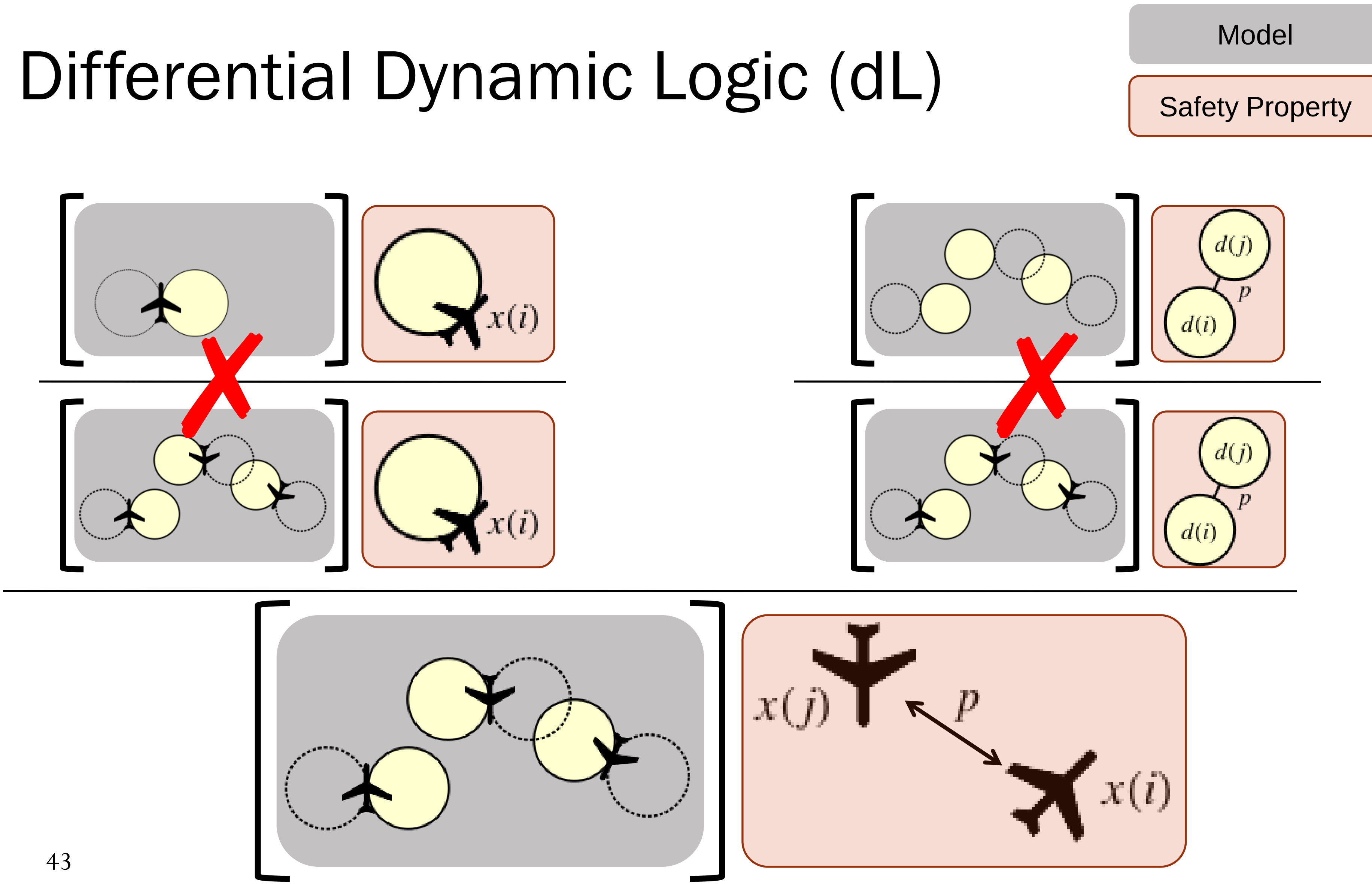


# Future Work for Distributed Aircraft



# Future Work for Distributed Aircraft

## Differential Dynamic Logic (dL)

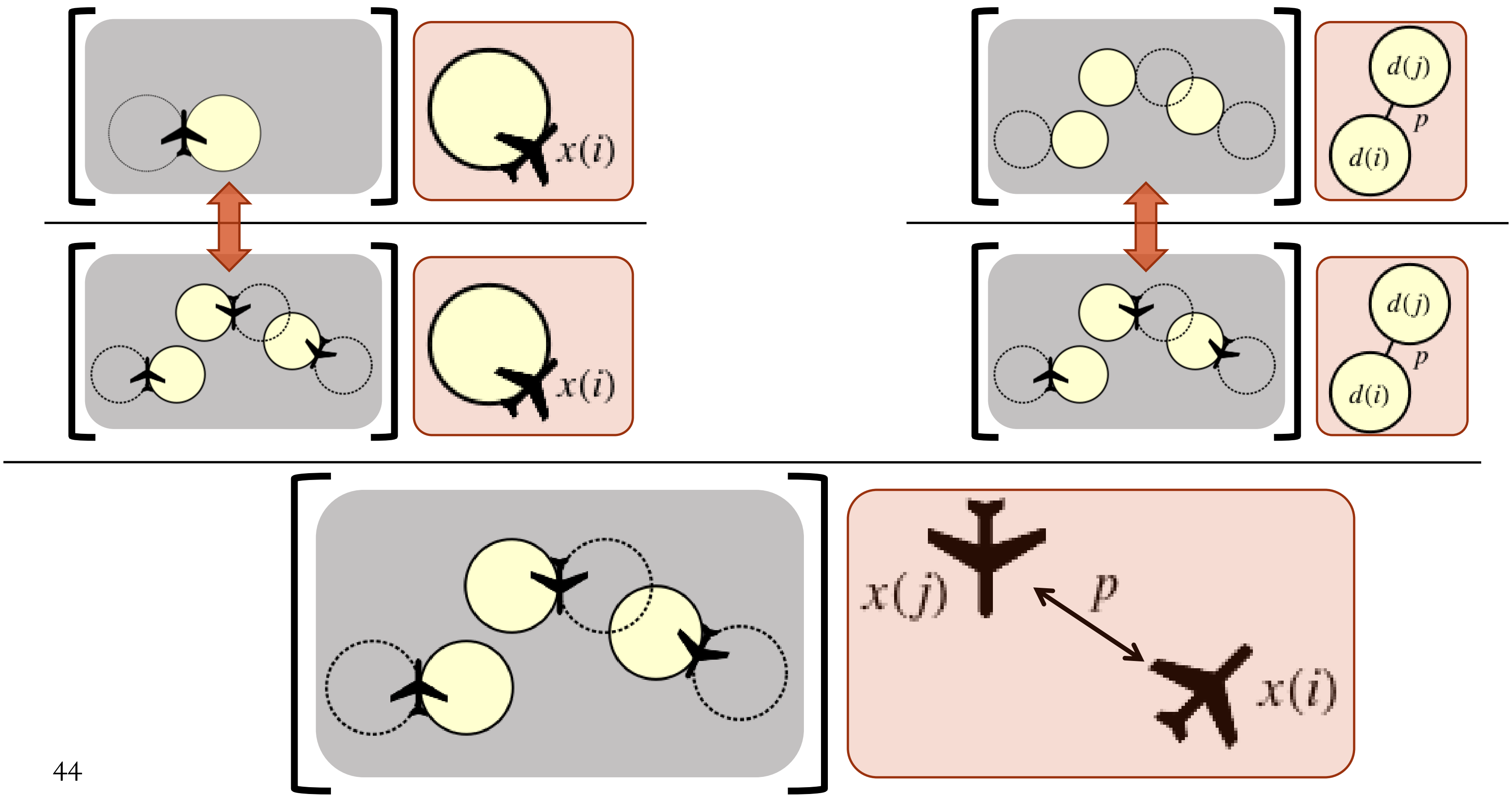


# Future Work for Distributed Aircraft

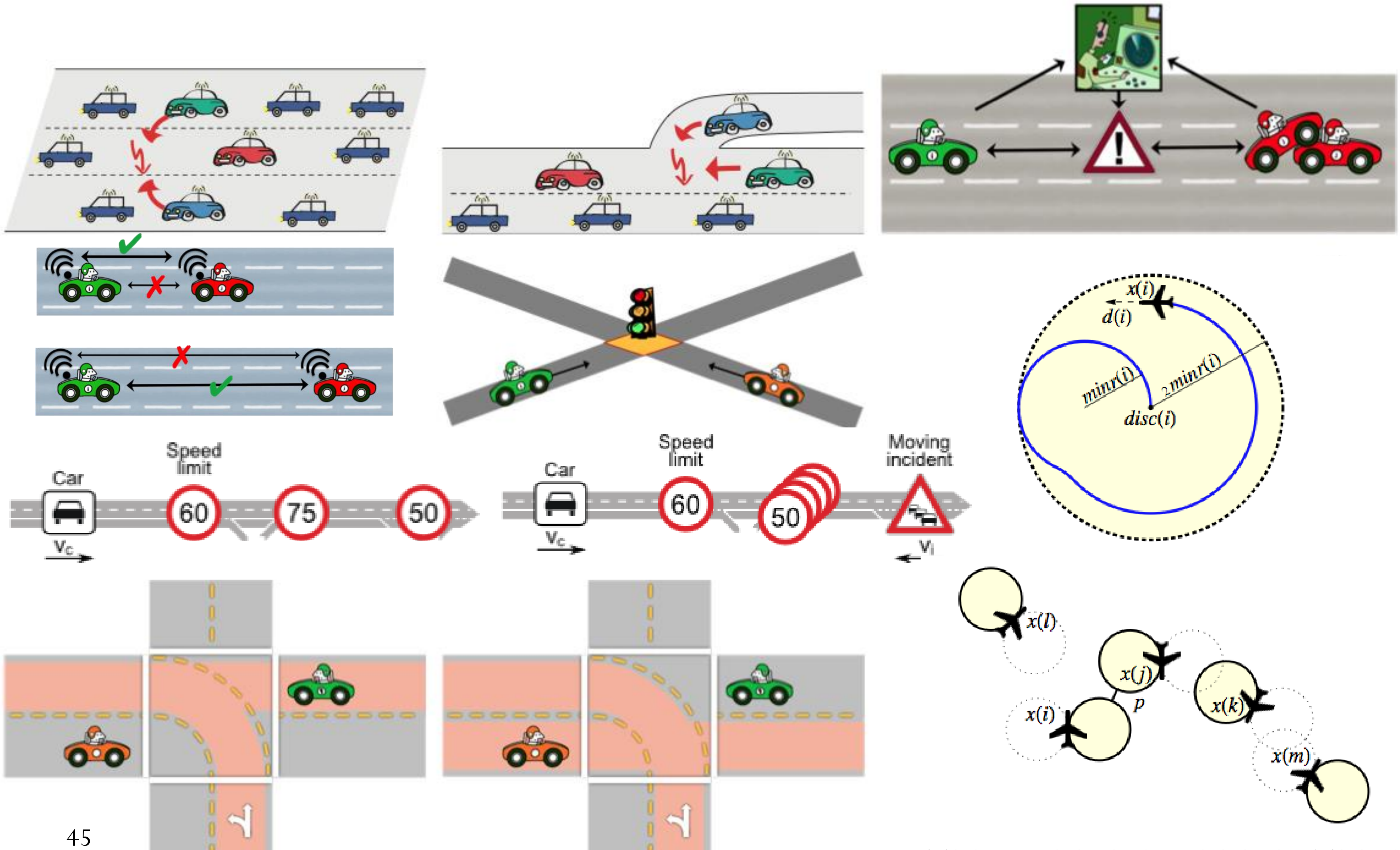
Model

Safety Property

## Differential *Refinement* Logic (dRL)



# Case studies now in scope for theorem proving

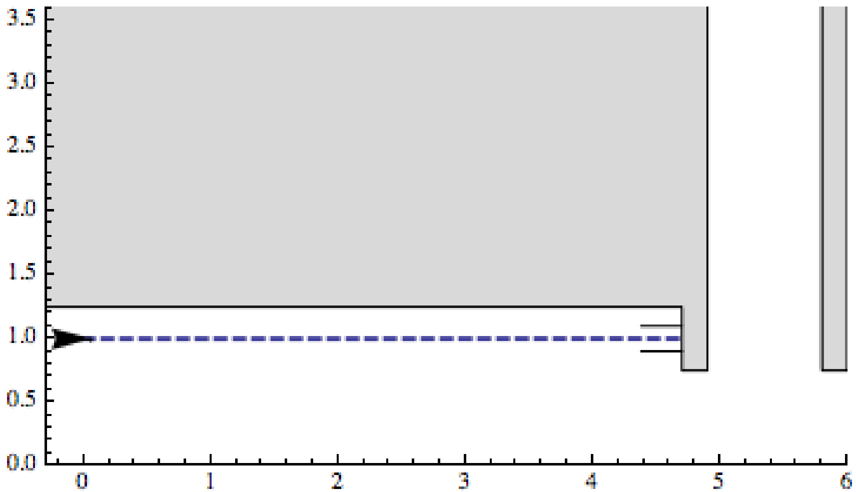




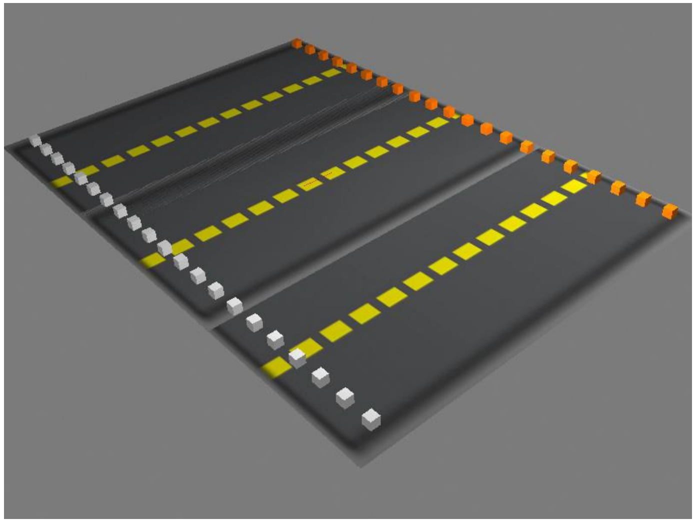

[FM11, ITSC11, ICCPS12, HSCC13, ITSC13]

# A Note on Pedagogy

## Charging Station Lab:




## Individual Simulations:

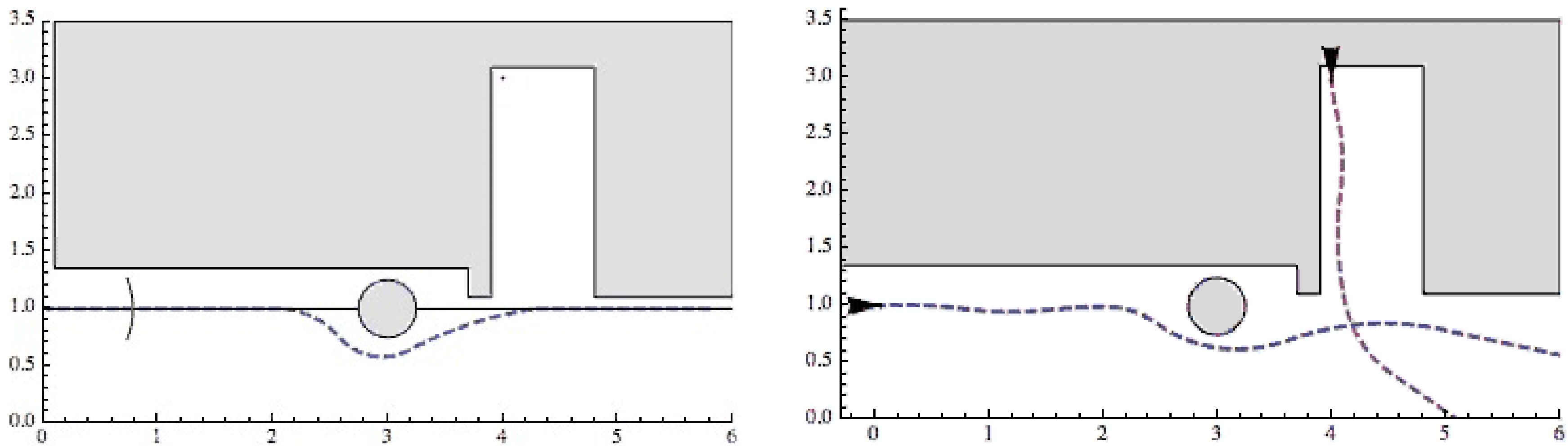


## Foundations of Cyber-Physical Systems:

- Offered Fall 2013 and Fall 2014 to ~20 undergraduate students.
- Covered background materials in both logic and differential equations.
- Students submitted practical labs using the KeYmaera theorem prover.
- Takeaway: theorem proving for CPS is in scope for undergrads!

# 2D Motion

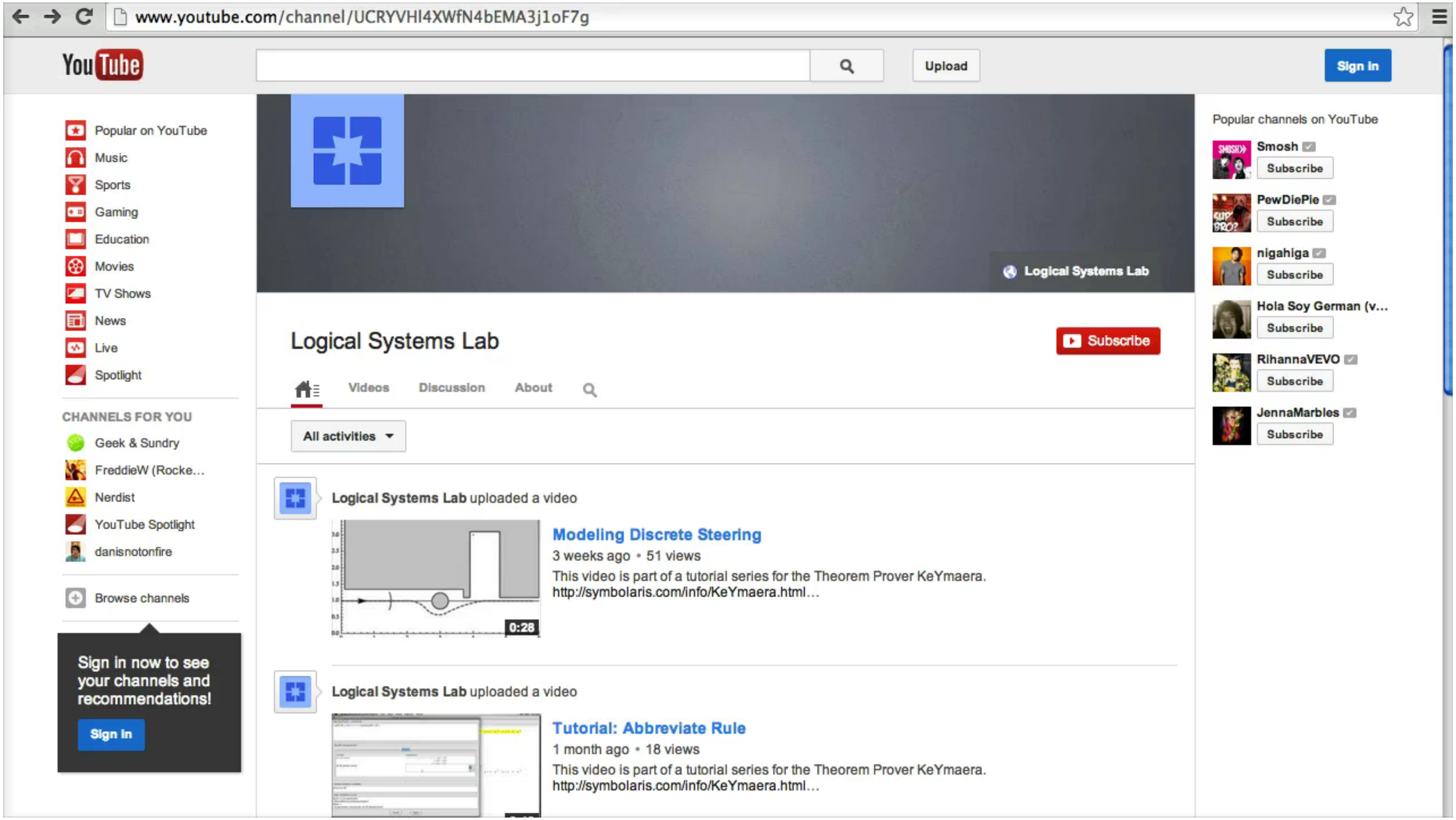
## with static and dynamic obstacles




**Challenges:**

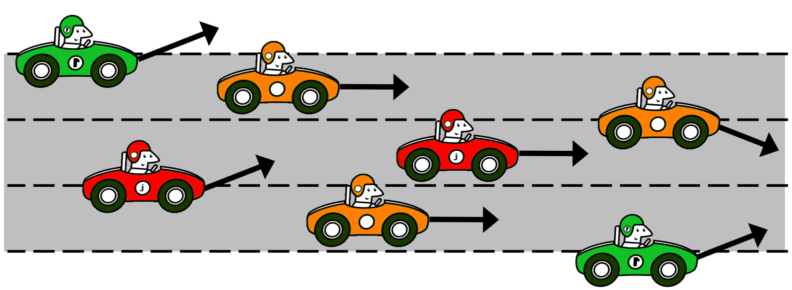
- System Loops
- 2D Motion (Dubins Model)
- Nondeterministic Controller
- Differential Equations
- Nonlinear Controller
- Complex Differential Invariants
- Proof Interactions and Branching
- Passive vs. Active Safety

# YouTube Video Tutorials



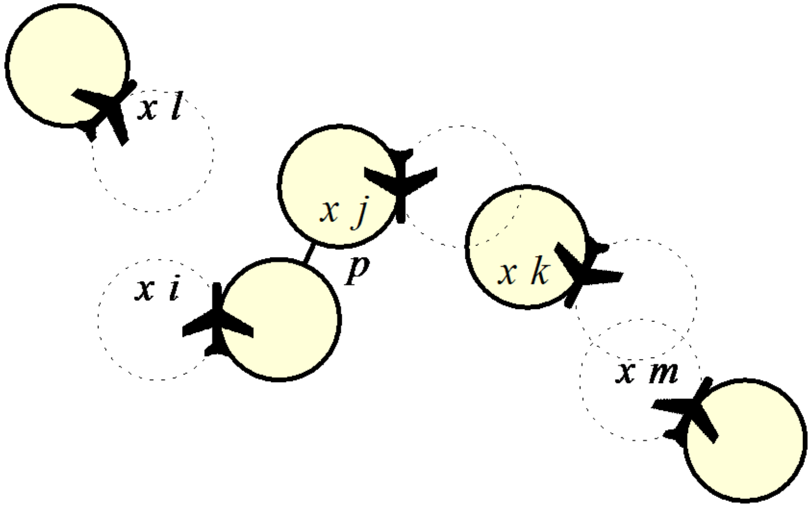
## Challenges

- Infinite, continuous, and evolving state space, $\mathbb{R}^\infty$
- Continuous dynamics
- Discrete control decisions
- Distributed dynamics
- Arbitrary number of aircraft
- Emergent behaviors



## Solutions

- Refinement gives hierarchical and modular proofs
- Quantifiers for distributed dynamics
- Non-linear flight paths allow flyable maneuvers
- Unbounded time horizon



# Thank You!


DOE CSGF

# References (page 1)

**Sarah M. Loos**, David Renshaw, and André Platzer. Formal Verification of Distributed Aircraft Controllers. In Calin Belta and Franjo Ivancic, editors, Hybrid Systems: Computation and Control (HSCC), 2013.

André Platzer and Jan-David Quesel. KeYmaera: A hybrid theorem prover for hybrid systems. In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *IJCAR*, volume 5195 of *LNCS*, pages 171-178. Springer, 2008

Platzer, André. "Differential dynamic logic for hybrid systems." Journal of Automated Reasoning 41.2 (2008): 143-189.

Nikos Aréchiga, **Sarah M. Loos**, André Platzer, and Bruce H. Krogh. Using theorem provers to guarantee closed-loop system properties. In the American Control Conference, ACC, Montréal, Canada, 2012.

Stefan Mitsch, **Sarah M. Loos**, and André Platzer. Towards Formal Verification of Freeway Traffic Control. In the International Conference on Cyber-Physical Systems, ICCPS, Beijing, China, 2012.

Lucia Pallottino, Vincenzo Giovanni Scordio, Antonio Bicchi, and Emilio Frazzoli. "Decentralized cooperative policy for conflict resolution in multivehicle systems." *Robotics, IEEE Transactions on* 23, no. 6, pages 1170-1183, 2007.

Kozen, Dexter. "Kleene algebra with tests." *ACM Transactions on Programming Languages and Systems (TOPLAS)* 19.3 (1997): 427-443.

# References (page 2)

Akshay Rajhans, Ajinkya Bhave, **Sarah M. Loos**, Bruce H. Krogh, André Platzer, and David Garlan. Using parameters in architectural views to support heterogeneous design and verification. In the IEEE Conference on Decision and Control and European Control Conference. 2011.

**Sarah M. Loos** and André Platzer. Safe Intersections: At the Crossing of Hybrid Systems and Verification. In the International IEEE Conference on Intelligent Transportation Systems, ITSC 2011, Washington, D.C., USA, Proceedings, 2011.

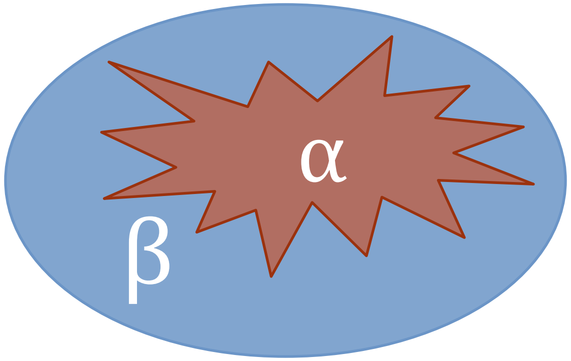
David Renshaw, **Sarah M. Loos**, and André Platzer. Distributed theorem proving for distributed hybrid systems. In the International Conference on Formal Engineering Methods, ICFEM'11, Durham, United Kingdom, Proceedings, LNCS. Springer, 2011.

**Sarah M. Loos**, André Platzer, and Ligia Nistor. Adaptive cruise control: Hybrid, distributed, and now formally verified. In the 17th International Symposium on Formal Methods, FM, Limerick, Ireland, Proceedings, LNCS. Springer, 2011.

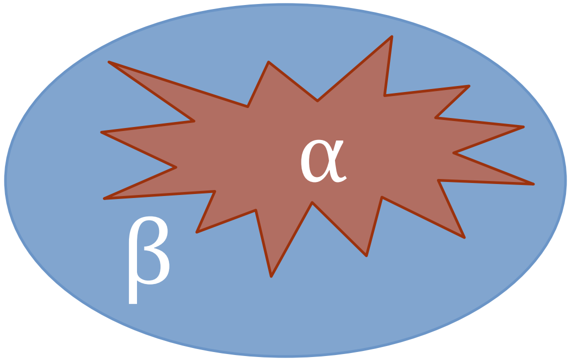
André Platzer. Quantified differential dynamic logic for distributed hybrid systems. In Computer Science Logic. Volume 6247 of LNCS. Springer, 2010.

Dubins, L.E. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. Am J Math 79(3), pages 497–516, 1957.

# Refinement Relation

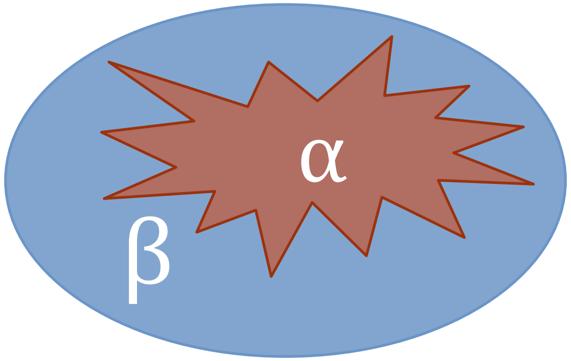


# Refinement Relation




$$\alpha \leq \beta$$

# Refinement Relation

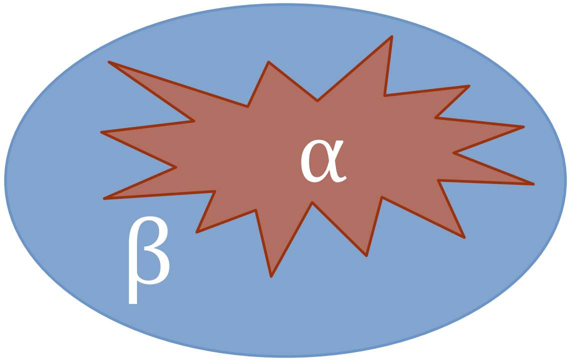



$$\alpha \leq \beta$$

$$\left((?\phi; a := \theta \cup a := -B); x'' = a \,\&\, \psi\right)^*$$

$$\left((?\phi; a := * \cup a := -B); x'' = a\right)^*$$

# Refinement Relation

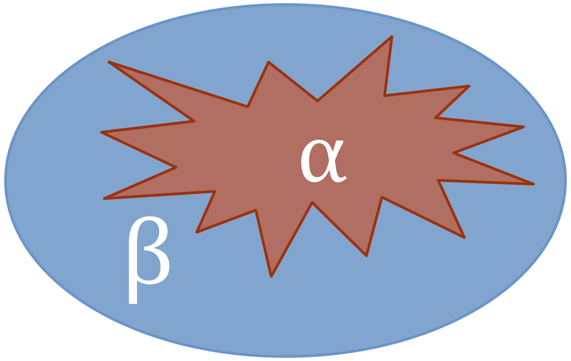



$$\boldsymbol{\alpha} \leq \boldsymbol{\beta}$$

$$\left((?\phi; a := \theta \cup a := -B); x'' = a \ \&\ \psi\right)^*$$

$$\left((?\phi; a := * \cup a := -B); x'' = a\right)^*$$

# Refinement Relation

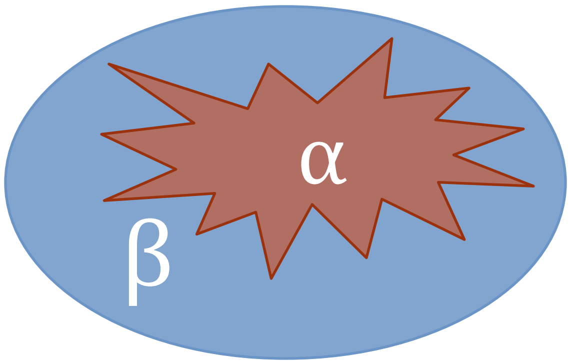



$$\alpha \leq \beta$$

$$((?\phi; a := \theta \cup a := -B); x'' = a \,\&\, \psi)^*$$

$$((?\phi; a := * \cup a := -B); x'' = a)^*$$

# Refinement Relation



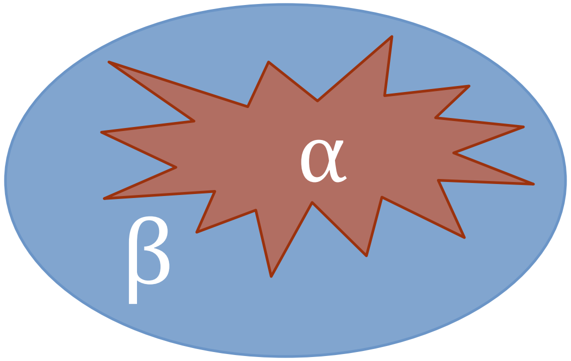

$$\boldsymbol{\alpha} \leq \boldsymbol{\beta}$$

$$((?\phi; a := \theta \cup a := -B); x'' = a \;\&\; \psi)^*$$

$$\leq$$

$$((?\phi; a := * \cup a := -B); x'' = a)^*$$

# Refinement Relation



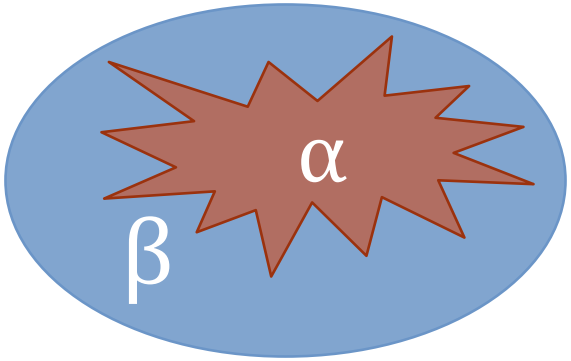

$$\alpha \leq \beta$$

$$\left((?\phi; a := \theta \cup a := -B); x'' = a \ \& \ \psi\right)^*$$

$$\leq$$

$$\left((?\phi; a := * \cup a := -B); x'' = a\right)^*$$

# Refinement Relation



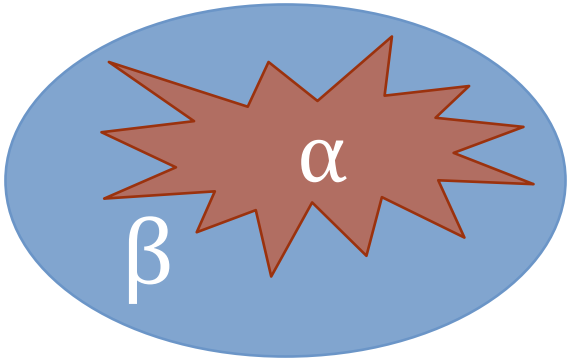

$$\alpha \leq \beta$$

$$\big((?\phi;\, a := \theta \cup a := -B);\, x'' = a \;\&\; \psi\big)^*$$

$$\leq \qquad\qquad\qquad\qquad \leq$$

$$\big((?\phi;\, a := * \cup a := -B);\, x'' = a\big)^*$$

# Refinement Relation




$$\alpha \leq \beta$$

$$\left((?\phi; a := \theta \cup a := -B); x'' = a \,\&\, \psi\right)^*$$

$$\leq$$

$$\left((?\phi; a := * \cup a := -B); x'' = a\right)^*$$

# So, what does dRL look like exactly?

Syntax of a dRL formula:

$$\phi, \psi ::= \theta_1 \leq \theta_2 \mid \neg\phi \mid \phi \wedge \psi \mid \forall x \phi$$
$$\mid [\alpha]\phi \mid \langle\alpha\rangle\phi$$

Syntax of a hybrid program:

$$\alpha, \beta ::= x := \theta \mid x' = \theta \,\&\, \psi \mid ?\psi$$
$$\mid \alpha \cup \beta \mid \alpha ; \beta \mid \alpha^*$$

d$\mathcal{L}$

[Platzer08]

# So, what does dRL look like exactly?

Syntax of a dRL formula:

$$\phi, \psi ::= \theta_1 \leq \theta_2 \mid \neg\phi \mid \phi \wedge \psi \mid \forall x \phi$$
$$\mid [\alpha]\phi \mid \langle\alpha\rangle\phi$$
$$\mid \boxed{\alpha \leq \beta}$$

dRL extends dL by adding refinement directly into the grammar of formulas

Syntax of a hybrid program:

$$\alpha, \beta ::= x := \theta \mid x' = \theta \;\&\; \psi \mid ?\psi$$
$$\mid \alpha \cup \beta \mid \alpha;\beta \mid \alpha^*$$