Distributed Synthesis of Distributed Control Protocols

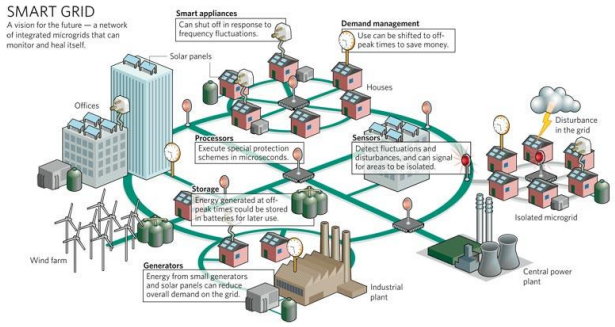
Necmiye Ozay
Caltech, CDS

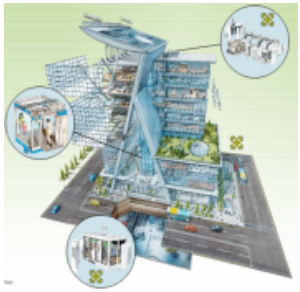
Keck Institute of Space Studies
Engineering Resilient Space Systems Workshop
1 August, 2012

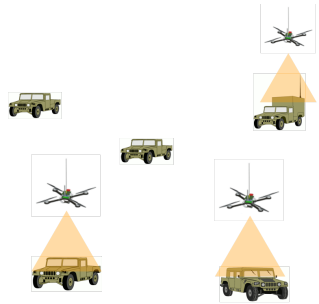
Distributed Control Protocols

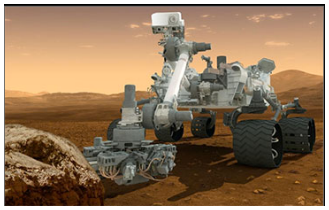
Motivation and Applications

- Large-scale, complex, distributed sensing, actuation and control systems:
  - **Smart grid, Smart buildings, Aircraft/ Spacecraft systems, Automotive, Robotics, Automation, Security**
- Centralized control protocols:
  - **Infeasible, unreliable (not robust to failure), lacking modularity**
- Scalable design and verification tools (theory and software) are lagging
- **Approach:** model-based, formal methods for specification, modular design, correct-by-construction distributed embedded controllers









Synthesis of Control Protocols

Given

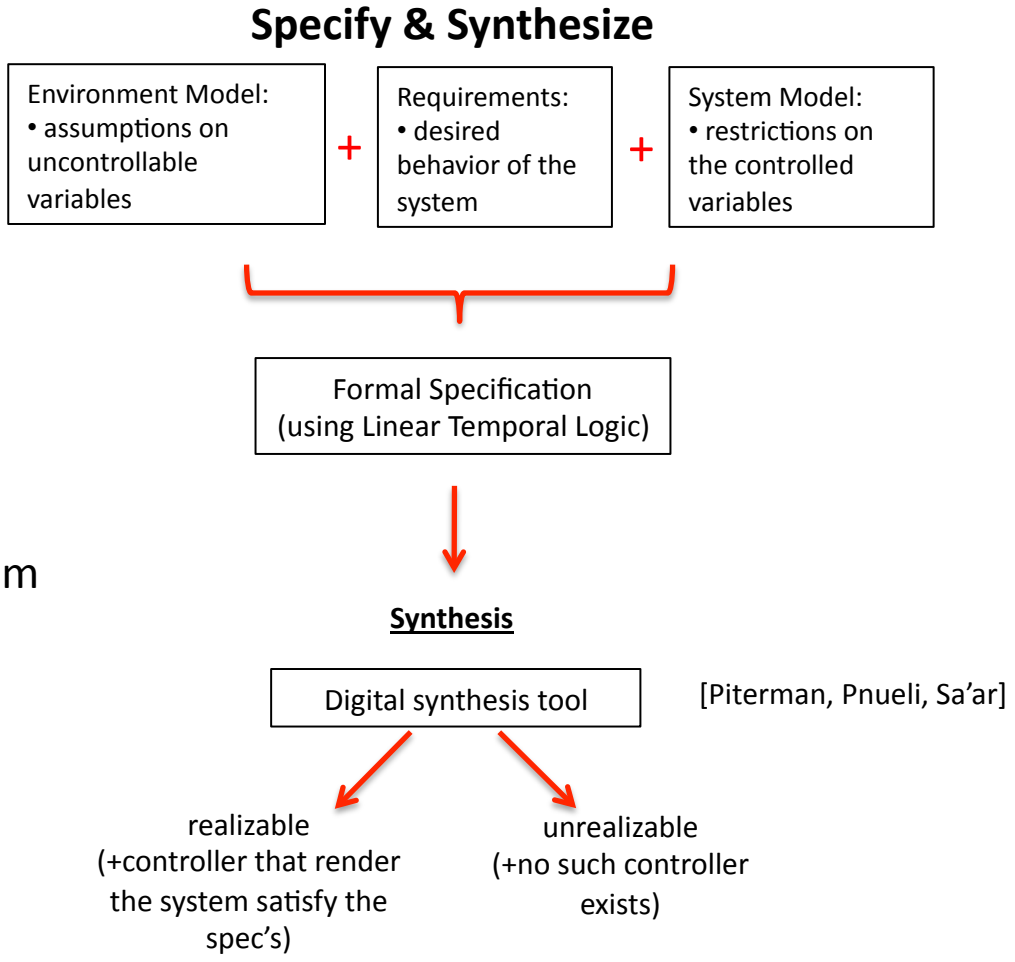
- models for the system and its environment
- specifications for the desired behavior

how to automatically design control protocols that

- manage the behavior of the system
- respond to changes in
  - internal system state
  - external environment

with

- ``correctness" guarantees?



Synthesis of Control Protocols

Given

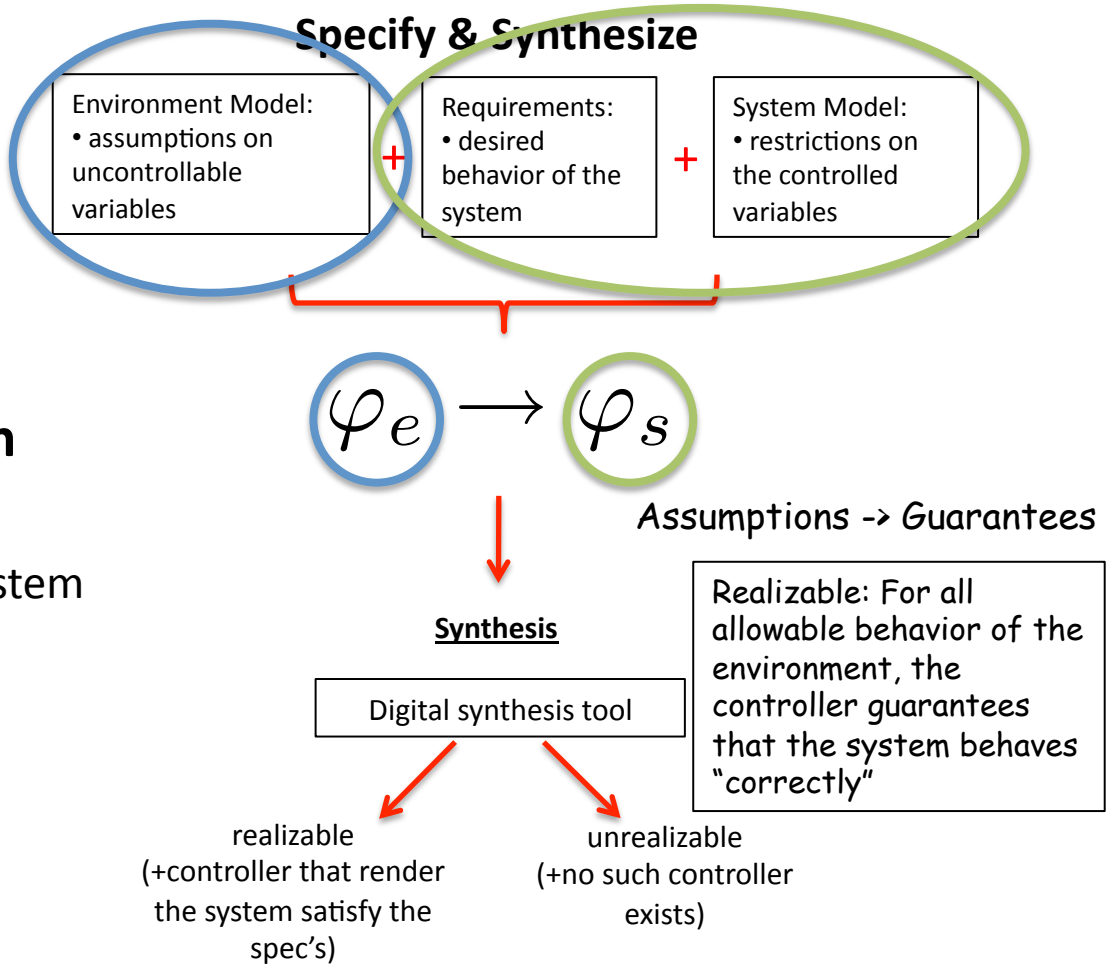
- models for the system and its environment
- specifications for the desired behavior

how to automatically design control protocols that

- manage the behavior of the system
- respond to changes in
  - internal system state
  - external environment

with

- ``correctness" guarantees?



Specifying Behavior Using Linear Temporal Logic (LTL)

Extends propositional logic with temporal operators

$\wedge$ (and), $\vee$ (or), $\rightarrow$ (implies), $\neg$ (not),

$\diamond$ (eventually), $\square$ (always), $\mathcal{U}$ (until), $\bigcirc$ (next),

- Allows to reason about infinite sequences of states
- Specifications (formulas) describe sets of allowable and desired behavior
  - safety specs: what actions are "not bad" or allowed
  - fairness: when an action can be/should be taken (e.g., infinitely often)

Specifying Behavior Using Linear Temporal Logic (LTL)

Extends propositional logic with temporal operators

$$\wedge \text{ (and)}, \ \vee \ \text{(or)}, \\ \rightarrow \ \text{(implies)}, \ \neg \text{ (not)},$$

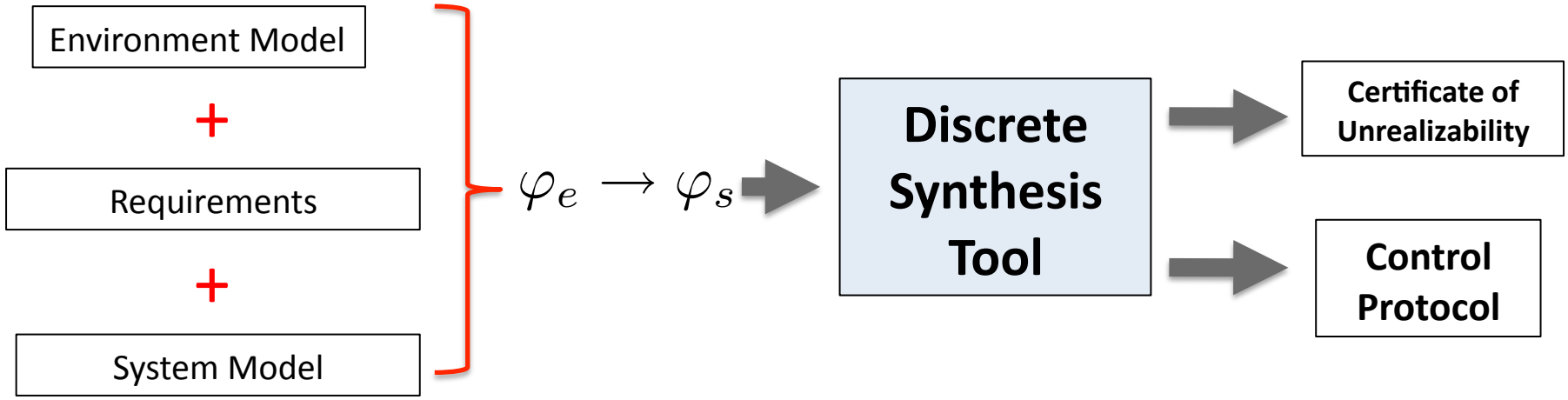
$$\diamond \text{ (eventually)}, \ \square \text{ (always)}, \\ \mathcal{U} \text{ (until)}, \ \bigcirc \ \text{(next)},$$

- Allows to reason about infinite sequences of states
- Specifications (formulas) describe sets of allowable and desired behavior
  - safety specs: what actions are "not bad" or allowed
  - fairness: when an action can be/should be taken (e.g., infinitely often)
- LTL operators can be combined to specify interesting behavior:

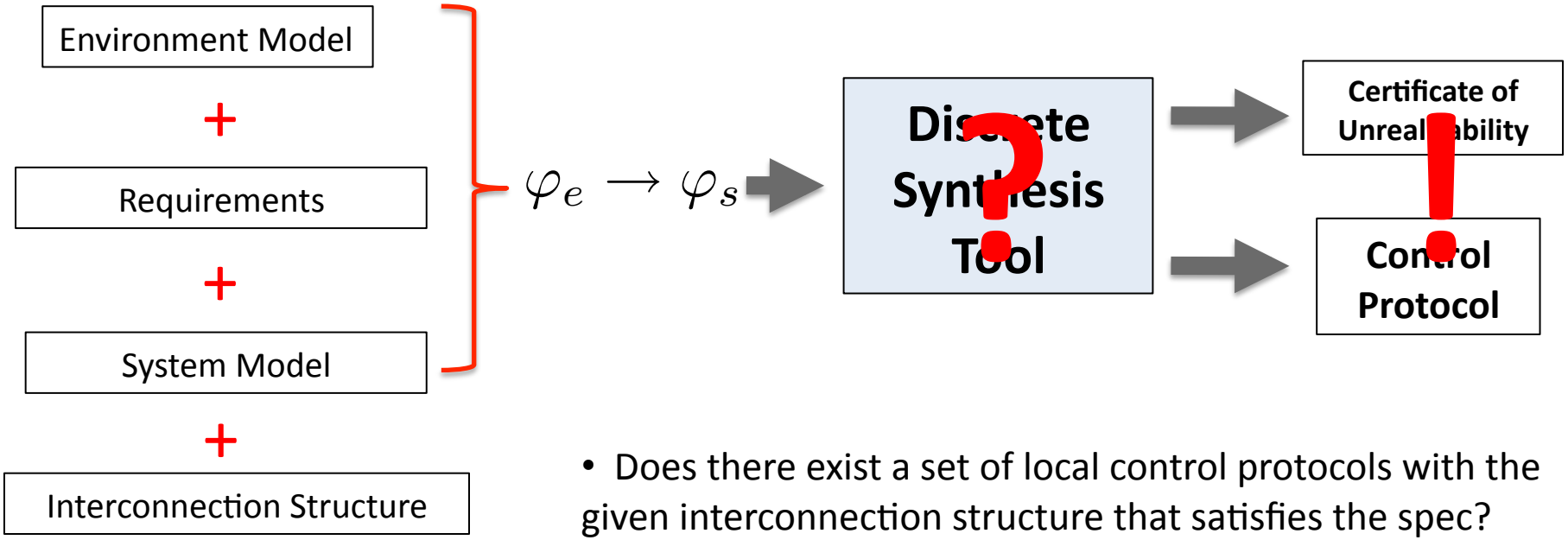
$$\square((\text{detect suspicious}) \rightarrow (\text{issue warning}))$$

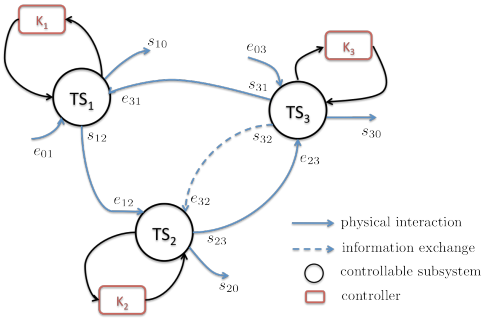
$$\text{takeoff} \rightarrow (\text{climb} \, \mathcal{U} \, (\text{cruise} \, \mathcal{U} \, (\text{descent} \, \mathcal{U} \, \text{land})))$$

Synthesis of Control Protocols



Synthesis of Distributed Control Protocols



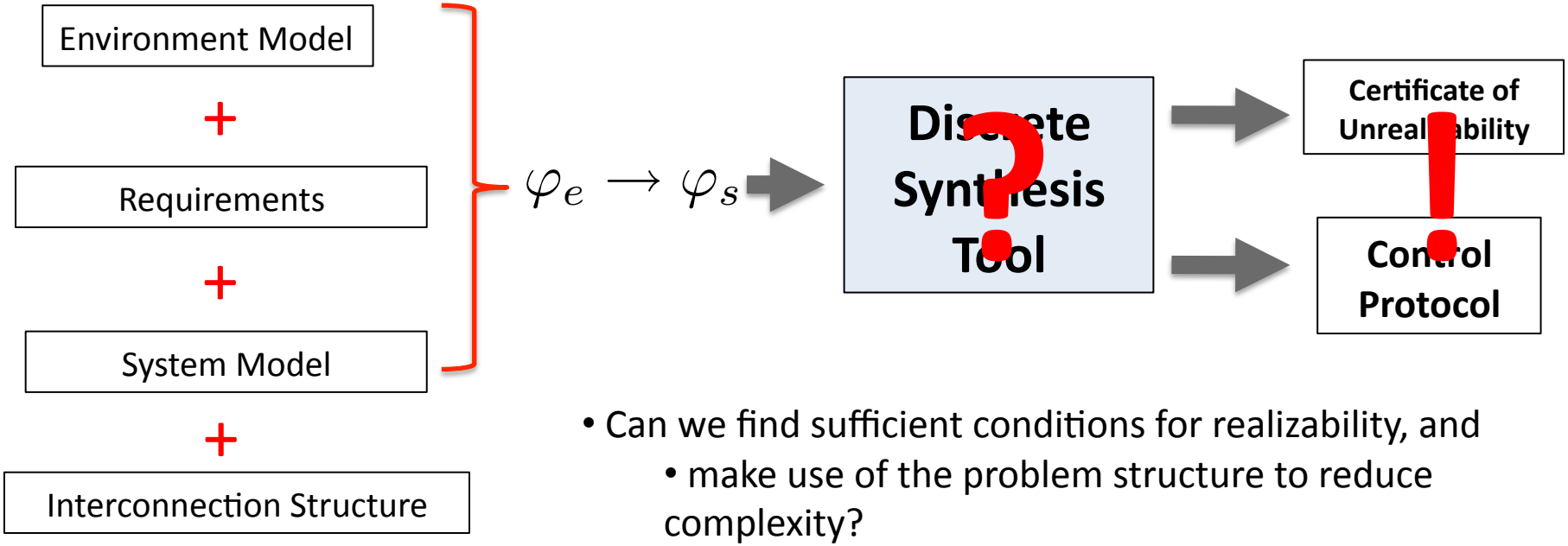


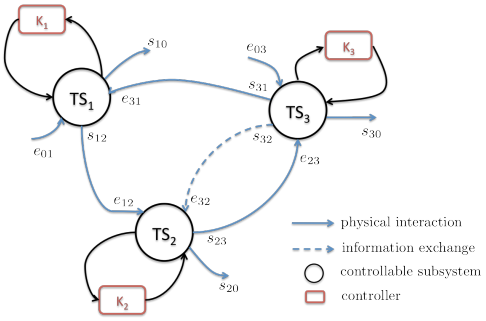
- Does there exist a set of local control protocols with the given interconnection structure that satisfies the spec?

UNDECIDABLE!

Pnueli, Rosner 90

Synthesis of Distributed Control Protocols

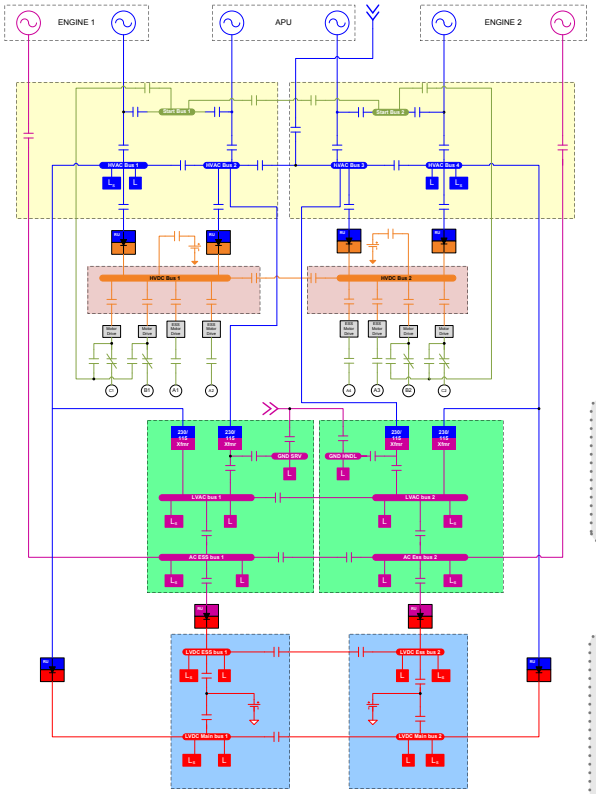


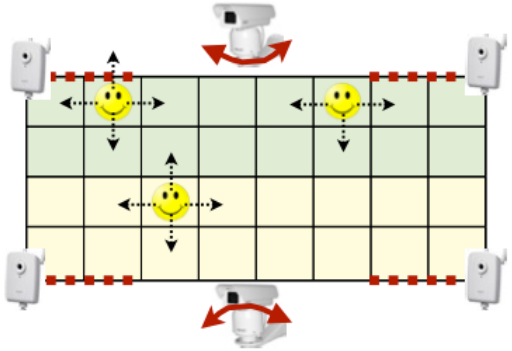


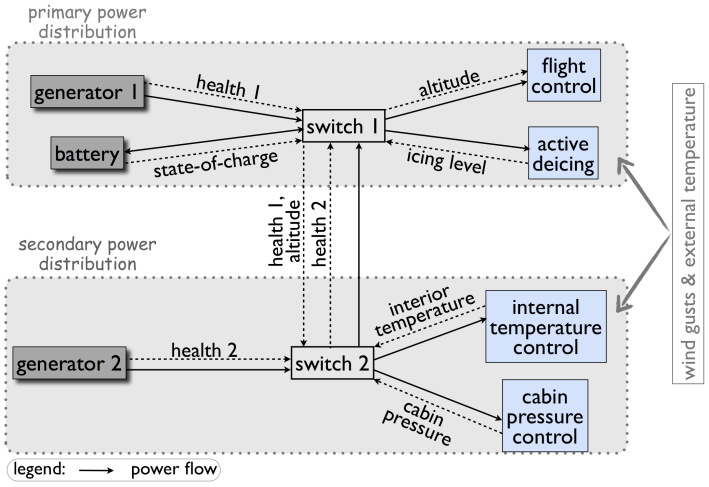
- Can we find sufficient conditions for realizability, and
  - make use of the problem structure to reduce complexity?
  - design control protocols that can be
    - synthesized
    - implemented

  in a decentralized way?
- What information exchange and interface models are needed?

Synthesis of Distributed Control Protocols





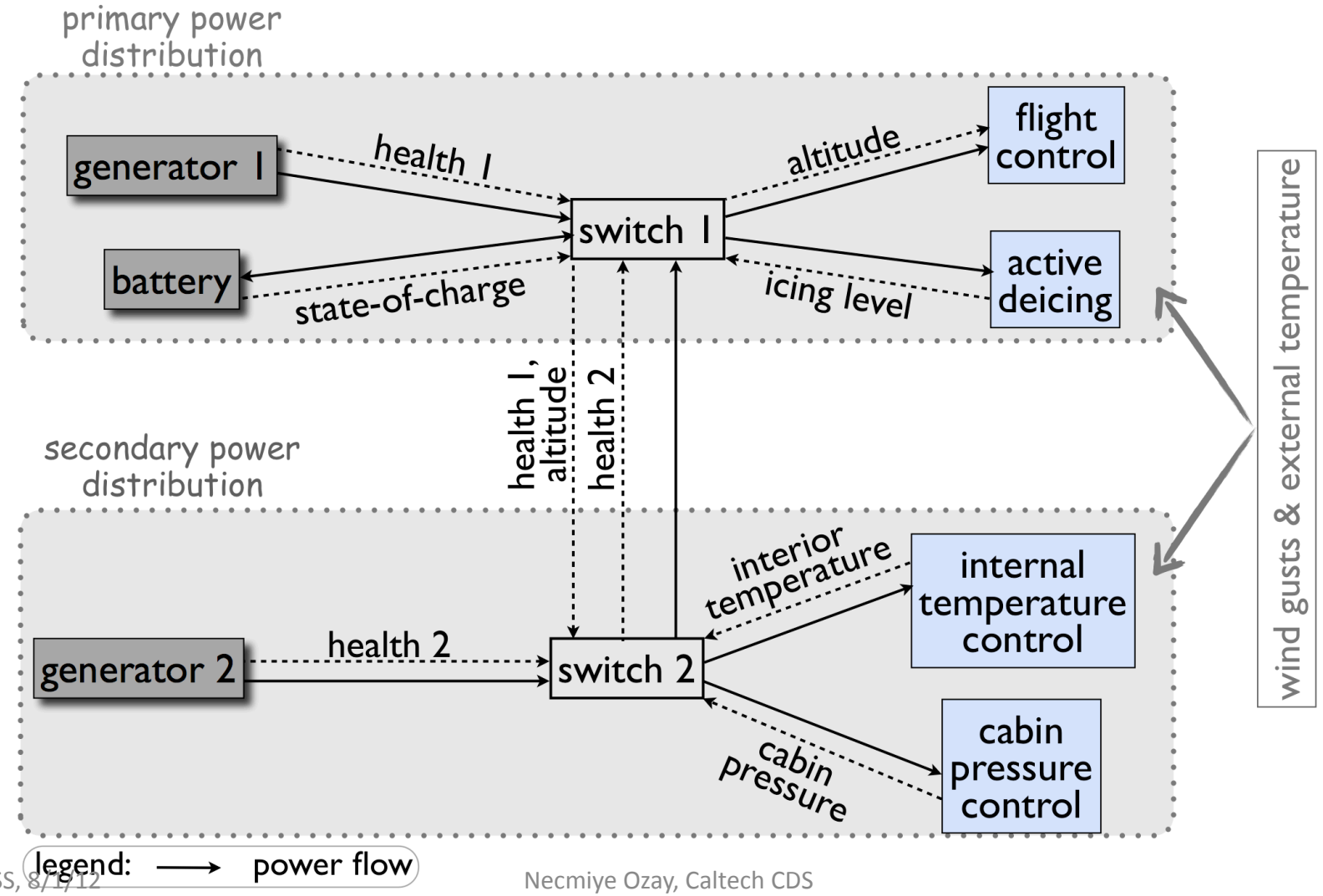


Collaborative decision making for groups of systems trying to achieve a common task:

- Power distribution for more electric aircraft
- Combined flight control and power allocation
  - Distributed surveillance (camera networks)

- Reliability
- Modularity
- Computational efficiency

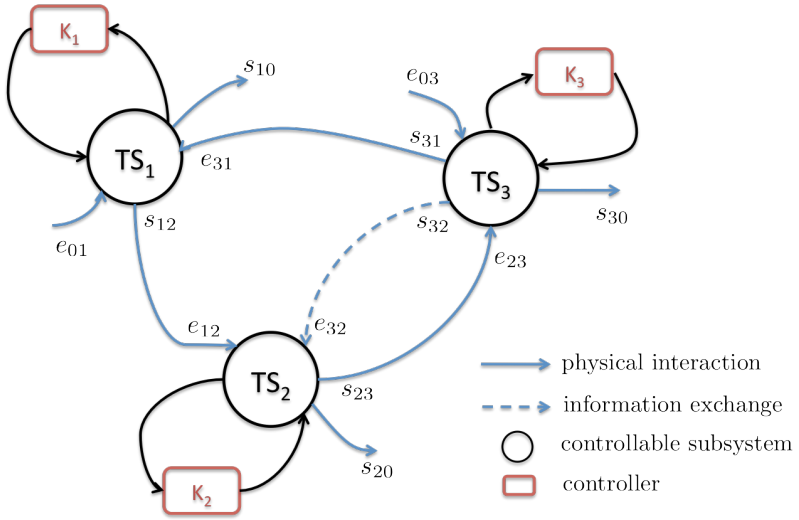
Synthesis of Distributed Control Protocols



Synthesis of Distributed Control Protocols

MAIN IDEA: Decompose the global specification $\varphi_e \longrightarrow \varphi_s$ into local ones, $\varphi_{e_i} \longrightarrow \varphi_{s_i}$.

- Decomposition induced by underlying network structure
- Physical constraints to avoid deadlocks and over-writing decisions



Theorem:

- If there exists local specifications $\varphi_{e_i} \longrightarrow \varphi_{s_i}$ s.t.

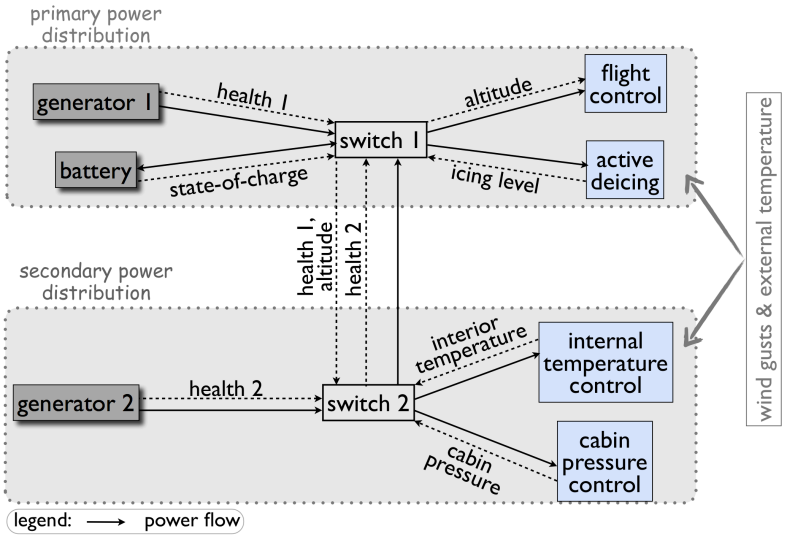
$$\wedge_i \varphi_{e_i} \longrightarrow \varphi_e \longrightarrow \varphi_s \longrightarrow \wedge_i \varphi_{s_i}$$

and each local specification is realizable by some controller K_i, then implementing K_i simultaneously satisfies the global spec.

Synthesis of Distributed Control Protocols

MAIN IDEA: Decompose the global specification $\varphi_e \longrightarrow \varphi_s$ into local ones, $\varphi_{e_i} \longrightarrow \varphi_{s_i}$.

- Decomposition induced by underlying network structure
- Physical constraints to avoid deadlocks and over-writing decisions



Theorem:

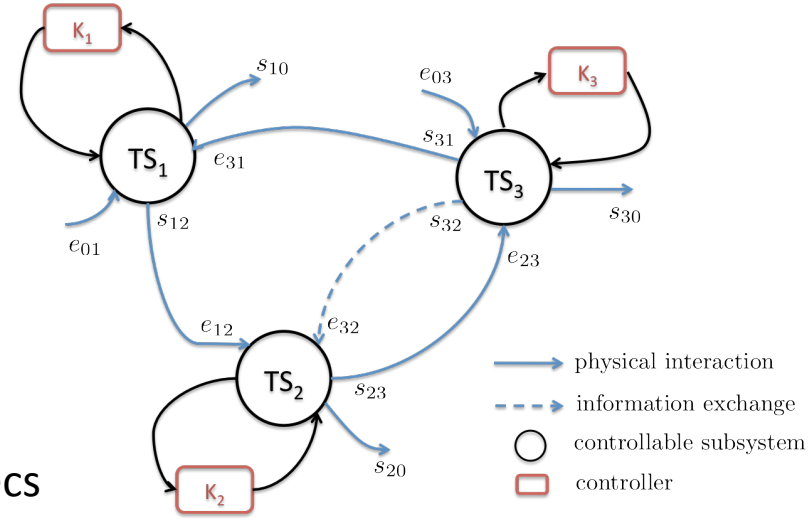
- If there exists local specifications $\varphi_{e_i} \longrightarrow \varphi_{s_i}$ s.t.

$$\wedge_i \varphi_{e_i} \longrightarrow \varphi_e \longrightarrow \varphi_s \longrightarrow \wedge_i \varphi_{s_i}$$

and each local specification is realizable by some controller K_i, then implementing K_i simultaneously satisfies the global spec.

Synthesis of Distributed Control Protocols

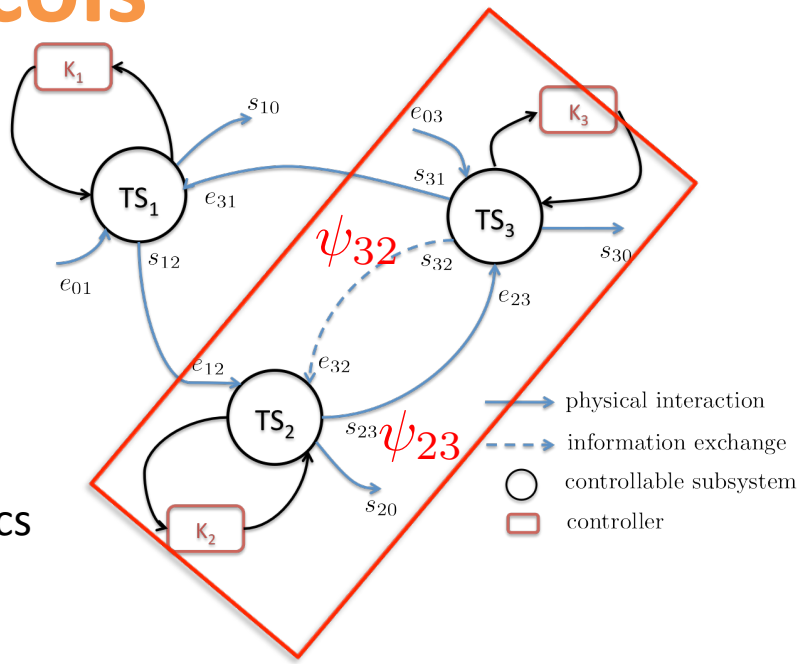
MAIN IDEA: Decompose the global specification $\varphi_e \longrightarrow \varphi_s$ into local ones $\varphi_{e_i} \longrightarrow \varphi_{s_i}$.



If the decompositions satisfy the logical and physical conditions, but some local specs are unrealizable; then one can refine the local specs by defining explicit interface rules.

Synthesis of Distributed Control Protocols

MAIN IDEA: Decompose the global specification $\varphi_e \longrightarrow \varphi_s$ into local ones $\varphi_{e_i} \longrightarrow \varphi_{s_i}$.



If the decompositions satisfy the logical and physical conditions, but some local specs are unrealizable; then one can refine the local specs by defining explicit interface rules.

Feedback interconnection refinement: Assume both $\varphi_{e_2} \longrightarrow \varphi_{s_2}$ and $\varphi_{e_3} \longrightarrow \varphi_{s_3}$ are unrealizable. If there exist ψ_{32} and ψ_{23} such that

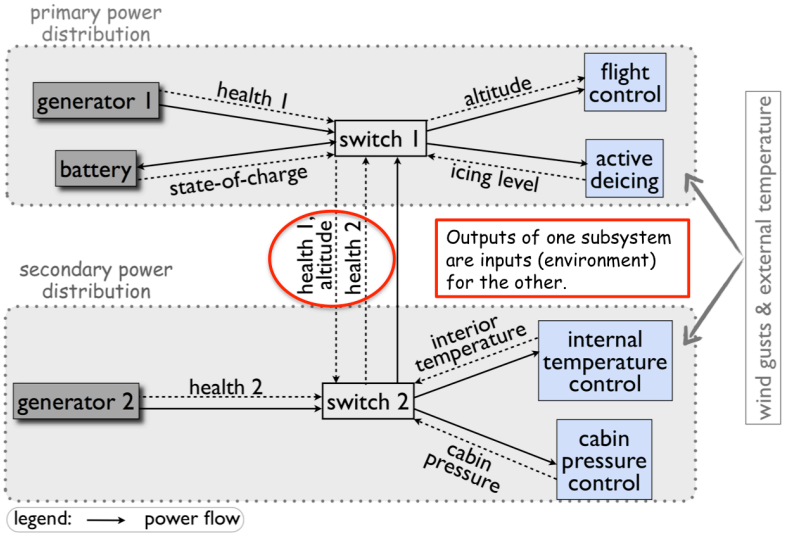
$$\psi_{32} \wedge \varphi_{e_2} \longrightarrow \varphi_{s_2} \wedge \psi_{23} \quad \text{and} \quad \psi_{23} \wedge \varphi_{e_3} \longrightarrow \varphi_{s_3} \wedge \psi_{32}$$

Is realizable, then the local control protocols for the refined spec's, guarantee that the global spec is satisfied.

need to be careful about circular reasoning!!!

Synthesis of Distributed Control Protocols

MAIN IDEA: Decompose the global specification $\varphi_e \longrightarrow \varphi_s$ into local ones $\varphi_{e_i} \longrightarrow \varphi_{s_i}$.



If the decompositions satisfy the logical and physical conditions, but some local specs are unrealizable; then one can refine the local specs by defining explicit interface rules.

Feedback interconnection refinement: Assume both $\varphi_{e_2} \longrightarrow \varphi_{s_2}$ and $\varphi_{e_3} \longrightarrow \varphi_{s_3}$ are unrealizable. If there exist ψ_{32} and ψ_{23} such that

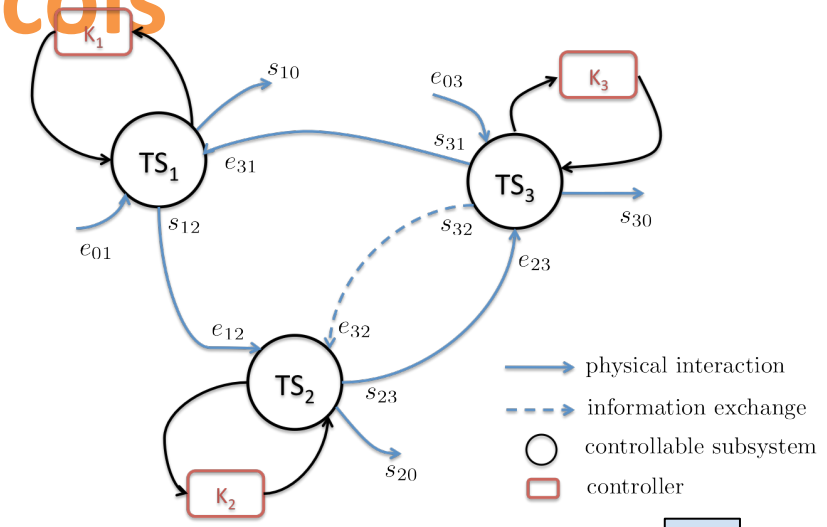
$$\psi_{32} \wedge \varphi_{e_2} \longrightarrow \varphi_{s_2} \wedge \psi_{23} \quad \text{and} \quad \psi_{23} \wedge \varphi_{e_3} \longrightarrow \varphi_{s_3} \wedge \psi_{32}$$

Is realizable, then the local control protocols for the refined spec's, guarantee that the global spec is satisfied.

need to be careful about circular reasoning!!!

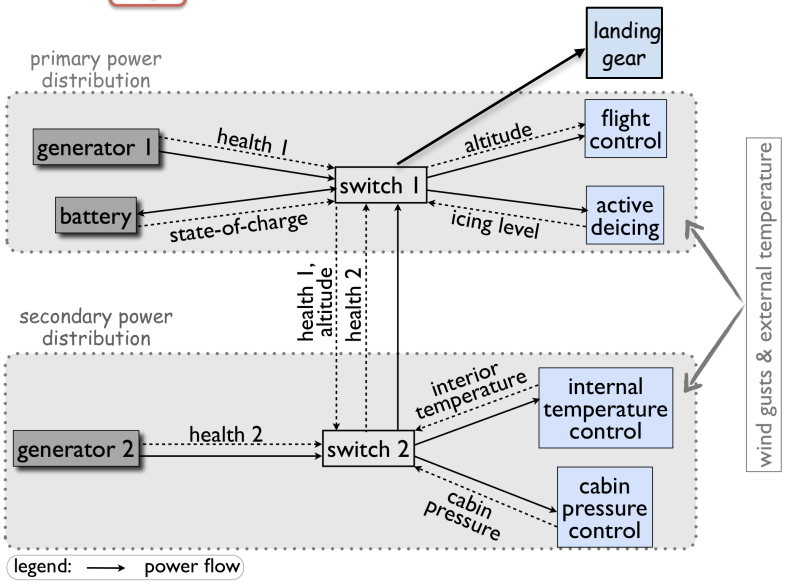
Synthesis of Distributed Control Protocols

MAIN IDEA: Decompose the global specification $\varphi_e \longrightarrow \varphi_s$ into local ones $\varphi_{e_i} \longrightarrow \varphi_{s_i}$.



These decompositions:

- allow local controllers to be
  - separately synthesized (substantial reductions in computational complexity),
  - locally implemented (increases reliability)
- provide assume/guarantee ``contracts'' for each subsystem (increases design modularity)



Distributed Control Protocols for Camera Networks

Tracking Software | Communication Protocol | **Control Protocol** | Communication Protocol | Continuous Dynamics

Environment Assumptions:

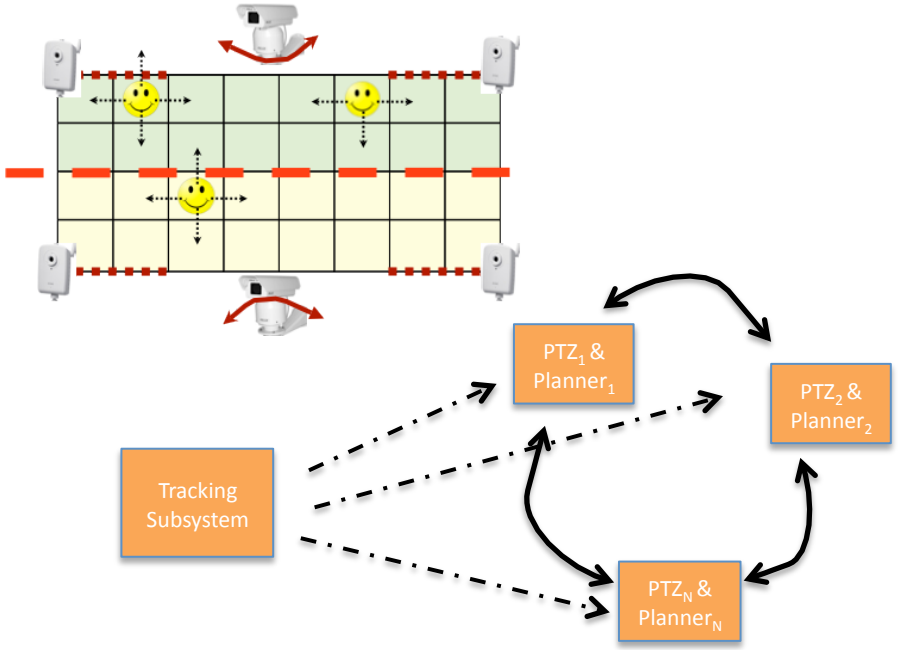
- At most N targets at a time
- Every target remains at least T time steps and eventually leaves
- Can only enter/exit through doors
- Can at most move a certain distance at each time step

System Model:

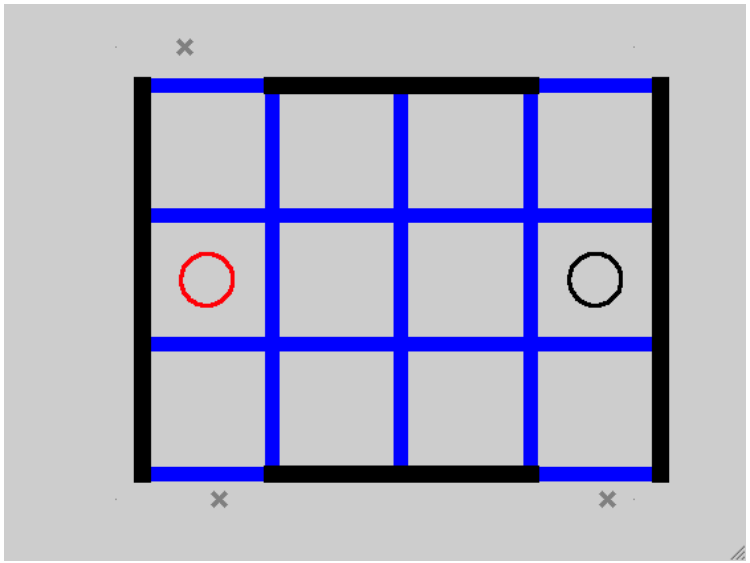
- Area of coverage of each PTZ
- Finite transition system representing PTZ motion

Sample Requirements:

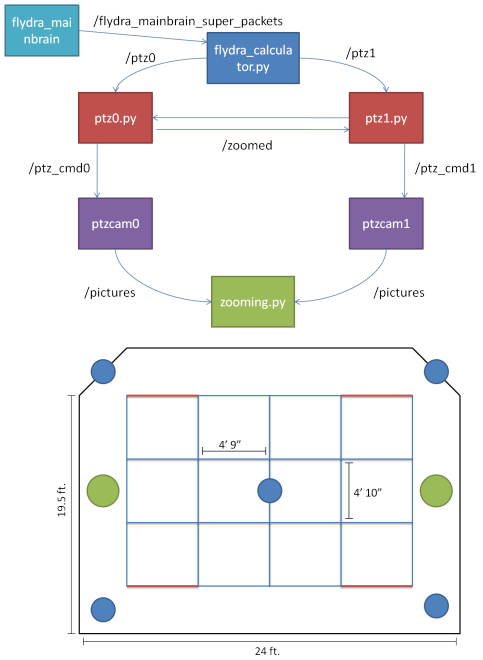
- Take a high resolution picture of each target before they leave the area
- Zoom into certain regions infinitely often



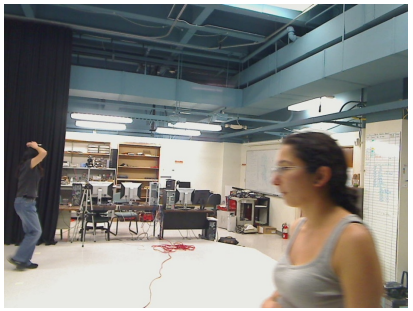
Simple Example

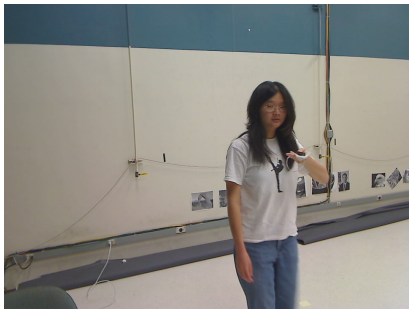


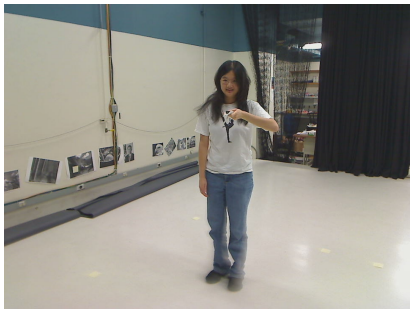
Lab camera network setup:



Results from our lab camera network setup:





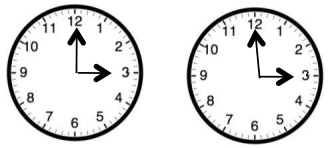


Ties to Resilient Systems?

- Temporal Logic Planning:
  - Convert specification into a design criteria: “specify and synthesize” instead of “design and verify”
  - Formal framework for specifying goals (science objectives) and requirements (fault management, hazard avoidance)
- Distributed synthesis: reduces complexity, enables local implementations
- Interface rules → modularity, contract based design
- Automatic synthesis of control protocol as an enabler for flexible autonomy (re-synthesize after learning more about operating conditions?)

Current Directions

- Current Directions:
  - More real-time aspects; models for communication delays
  - What does the controller need to know about implementation?



  - We assume synchronous execution in synthesis, can we allow/tolerate asynchrony to some extend?
  - Automatic exploration of distributed control architectures (information graph, logical decompositions)
  - Need to develop tools for automating
    - the initial decomposition of spec's in distributed synthesis
    - the refinement step (interface rules)