Lecture 9
Extensions and Open Problems

Richard M. Murray
Nok Wongpiromsarn   Ufuk Topcu
California Institute of Technology

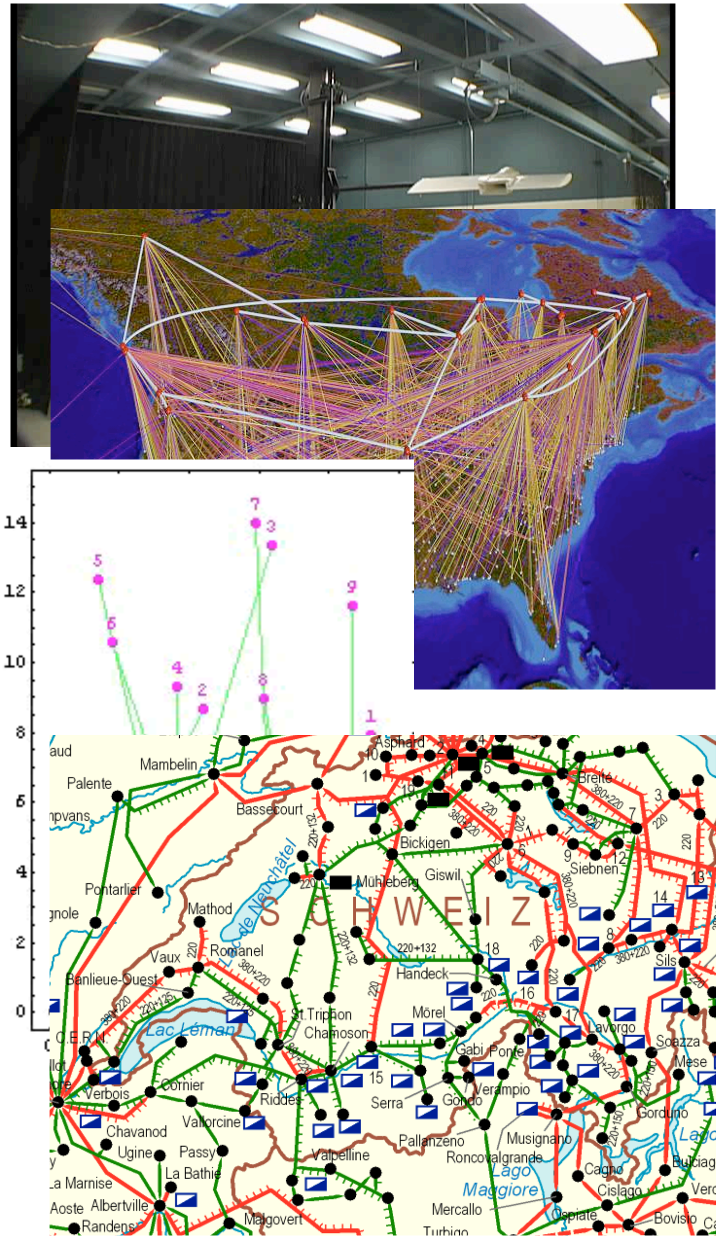
EECI, 18 May 2012

Outline:

- Review key concepts from the course
- Discussion open issues, technical challenges and risks
- Hand out certificates!

“$TS \models \Box(\neg b \rightarrow \Box(a \wedge \neg b)$”

Some Important Trends in Control in the Last Decade



(Online) Optimization-based control

- Increased use of online optimization (MPC/RHC)
- Use knowledge of (current) constraints & environment to allow performance and adaptability

Layering and architectures

- Command & control at multiple levels of abstraction
- Modularity in product families via layers

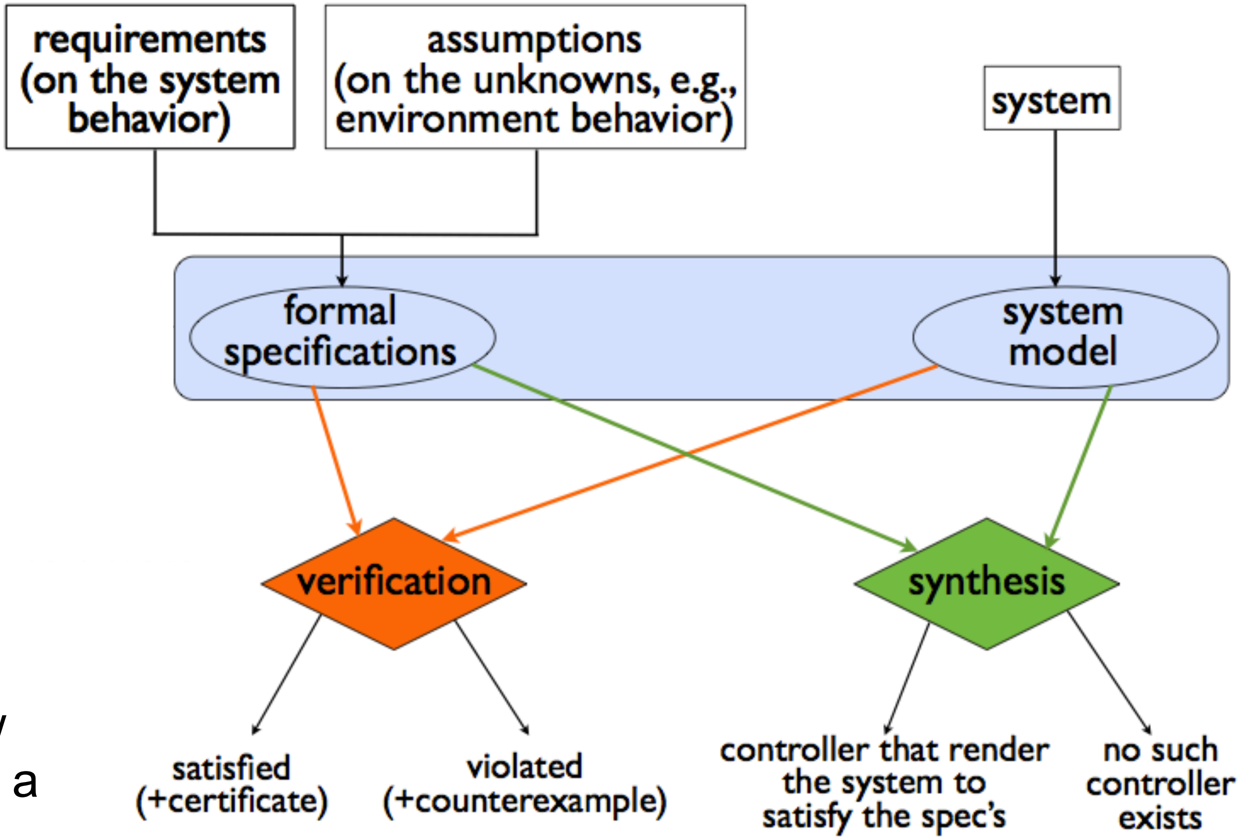
Formal methods for analysis, design and synthesis

- Combinations of continuous and discrete systems
- Formal methods from computer science, adapted for hybrid systems (mixed continuous & discrete states)

Components → Systems → Enterprise

- Movement of control techniques from “inner loop” to “outer loop” to entire enterprise (eg, supply chains)
- Use of *systematic* modeling, analysis and synthesis techniques at all levels
- Integration of “software” with “controls” (Internet of things, cyber-physical systems, etc)

Formal Methods for System Verification & Synthesis


requirements (on the system behavior)
assumptions (on the unknowns, e.g., environment behavior)
system
formal specifications
system model
verification
synthesis
satisfied (+certificate)
violated (+counterexample)
controller that render the system to satisfy the spec's
no such controller exists

Specification using LTL

- Linear temporal logic (LTL) is a math'l language for describing linear-time prop's
- Provides a particularly useful set of operators for constructing LT properties without specifying sets

Methods for verifying an LTL specification

- *Theorem proving*: use formal logical manipulations to show that a property is satisfied for a given system model
- *Model checking*: explicitly check all possible executions of a system model and verify that each of them satisfies the formal specification

Methods for *synthesis* of correct-by-construction control protocols

- Build on results in logic synthesis and (recent) results in GR(1) synthesis
- Key challenges: dynamics, uncertainty, complexity

Hybrid, Multi-Agent System Description

Subsystem/agent dynamics - continuous

$$\begin{aligned} \dot{x}^i &= f^i(x^i, \alpha^i, y^{\sim i}, u^i) \qquad & x^i \in \mathbb{R}^n, u^i \in \mathbb{R}^m \\ y^i &= h^i(x^i, \alpha^i) & y^i \in \mathbb{R}^q \end{aligned}$$

Agent mode (or "role") - discrete

- $\alpha \in \mathcal{A}$ encodes internal state + relationship to current task
- Transition $\alpha' = r(x, \alpha)$

Communications graph $\mathcal{G}$

- Encodes the system information flow
- Neighbor set $\mathcal{N}^i(x, \alpha)$

Communications channel

- Communicated information can be lost, delayed, reordered; rate constraints

$$y^i_j[k] = \gamma y^i(t_k - \tau_j) \qquad t_{k+1} - t_k > T_r$$

- γ = binary random process (packet loss)

Task

- Encode task as finite horizon optimal control + temporal logic (assume coupled)

$$J = \int_0^T L(x, \alpha, u)\, dt + V(x(T), \alpha(T)),$$

$$(\varphi_{init} \ \wedge \ \Box\varphi_e) \implies (\Box\varphi_s \ \wedge \ \Diamond\varphi_g)$$

Strategy

- Control action for individual agents

$$u^i = \gamma(x, \alpha) \qquad \{g^i_j(x, \alpha) : r^i_j(x, \alpha)\}$$

$$\alpha^{i\,\prime} = \begin{cases} r^i_j(x, \alpha) & g(x, \alpha) = \text{true} \\ \text{unchanged} & \text{otherwise.} \end{cases}$$

***Decentralized* strategy**

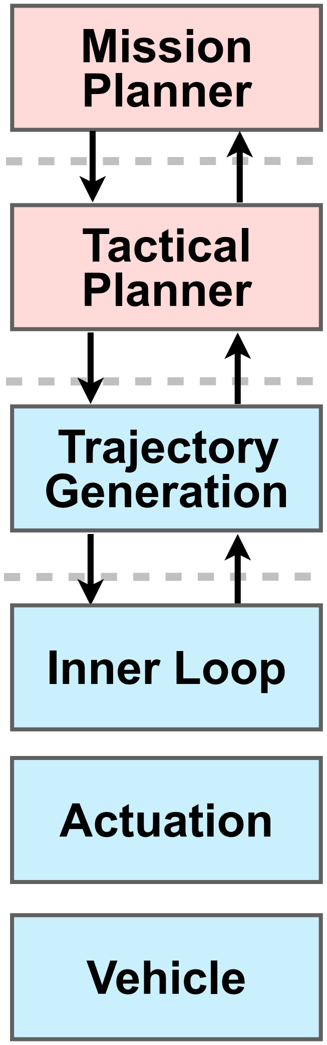
$$u^i(x, \alpha) = u^i(x^i, \alpha^i, y^{-i}, \alpha^{-i})$$

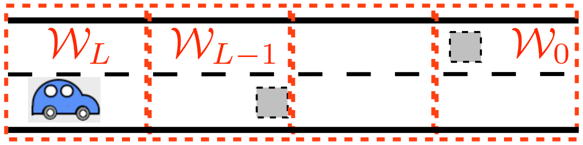
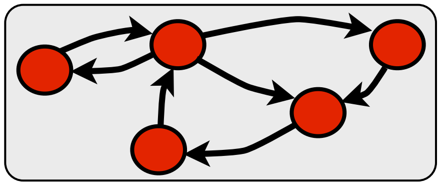
$$y^{-i} = \{y^{j_1}, \dots, y^{j_{m_i}}\}$$

$$j_k \in \mathcal{N}^i \qquad m_i = |\mathcal{N}^i|$$

- Similar structure for role update

Hierarchical, Networked Control Systems



Different views	Multi-scale models	Implementation
"long-horizon specification"	$\mathcal{W}_0 \prec \ldots \prec \mathcal{W}_{L-1} \prec \mathcal{W}_L$ 	• High level "mission" abstraction (iterative graph search)
"short-horizon specification"		• Reactive planner: decision-making based on events in environment
continuous dynamics& constraints	$\min \int_{t_0}^{T} L(x,u)dt$ $\text{s.t.} \quad \dot{x} = f(x,u)$ $g(x,u) \leq 0$	• Continuous control action based on "mode" and cost

Multi-layer approach

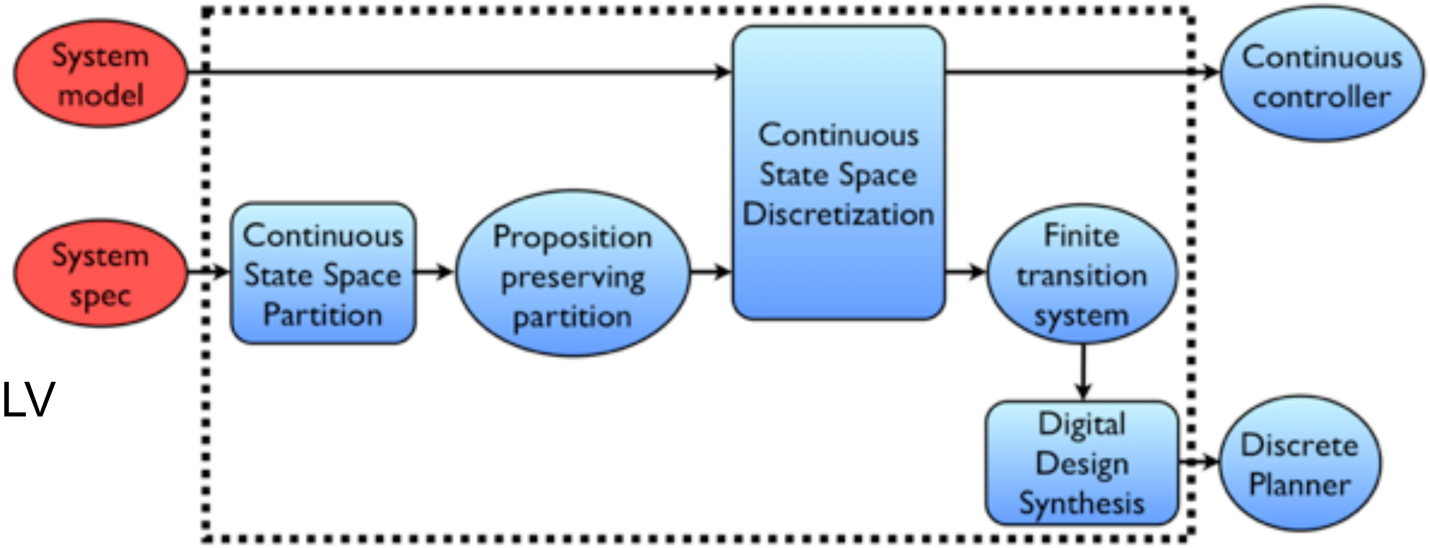
- Use optimal trajectory generation to create a discrete abstraction that captures the dynamics at a simplified level
- Reactive planner based on GR(1) synthesis (possibly RHC)
- High level planner sends specifications to reactive planer
- Online versus offline decisions at each level

Temporal Logic Planning (TuLiP) toolbox

http://tulip-control.sourceforge.net

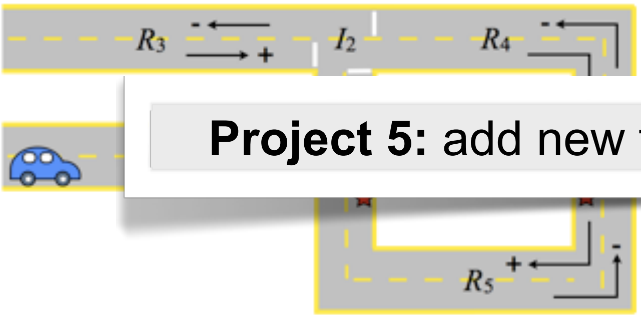
Python Toolbox

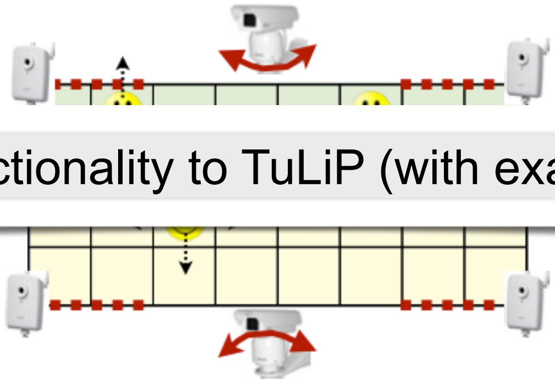
- GR(1), LTL specs
- Nonlin dynamics
- Supports discretization via MPT
- Control protocol designed using JTLV
- Receding horizon compatible

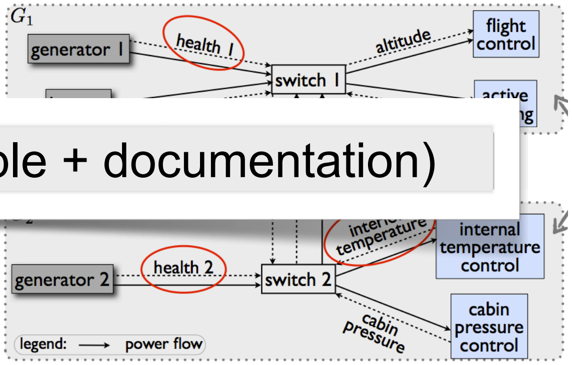


Applications of TuLiP in the last few years

- Autonomous vehicles - traffic planner (intersections and roads, with other vehicles)
- Distributed camera networks - cooperating cameras to track people in region
- Electric power transfer - fault-tolerant control of generator + switches + loads

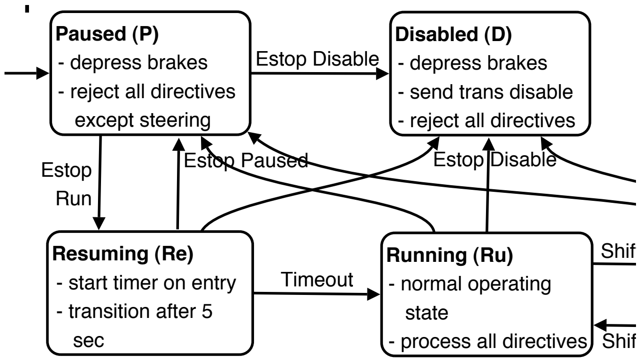






Project 5: add new functionality to TuLiP (with example + documentation)

Project Possibilities



Paused (P)
- depress brakes
- reject all directives except steering

Disabled (D)
- depress brakes
- send trans disable
- reject all directives

Resuming (Re)
- start timer on entry
- transition after 5 sec

Running (Ru)
- normal operating state
- process all directives

Estop Disable

Estop Run

Estop Paused

Estop Disable

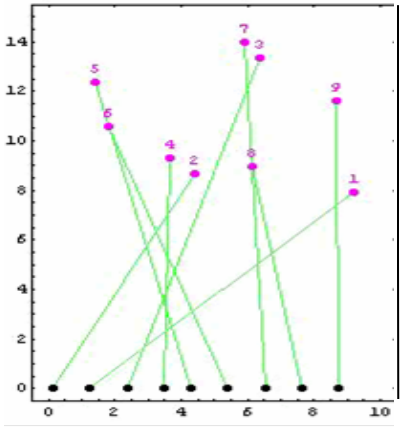
Timeout

Shif

Shif

Possibilities mentioned during the course

1. Verify correctness of actuation mode logic for Alice using SPIN model checker and message channels
2. Create a model of the RoboFlag drill (simplified) in Promela and verify correctness using SPIN
3. Create a specification for the RoboFlag drill and synthesize a (decentralized) protocol to solve it
4. Synthesize intersection logic for a car at an intersection
5. Add new functionality to TuLiP (with document'n + examples)

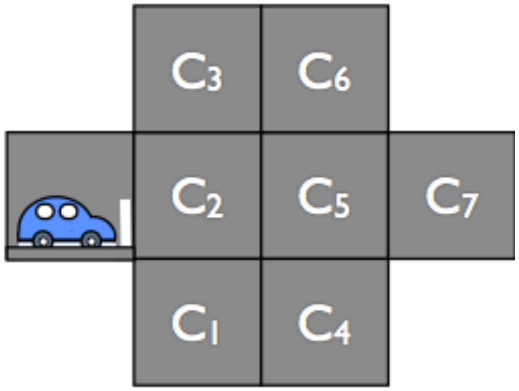


Project from your own research area

- Pick something where you can do verification and/or synthesis of a control protocol
- Could be theoretical or computational

Process

- Send e-mail to Richard & Ufuk in the coming week with a specific proposal
- Work through the project and write up a 4-8 page report
- No specific due date, but June or July would be good



Open (Research) Issues

Optimality: "language-constrained, optimal trajectory generation"

$$(\varphi_{init} \ \wedge \ \Box\varphi_e) \implies (\Box\varphi_s \ \wedge \ \Diamond\varphi_g) \qquad J = \int_0^T L(x, \alpha, u)\, dt + V(x(T), \alpha(T)),$$

Partial order computation and hierarchical structure

- How do we determine the partial order for RHTLP and link to "supervisory" levels?

Verification and synthesis with (hard) real-time constraints

- How do we incorporate time in our specifications, verification and synthesis tools?
- Note: time automata and timed temporal logic formulas available...

Contract-based design: automate search interfaces for distributed synthesis

- How do we decompose a larger problem into smaller pieces?
- Especially important for large scale projects with multiple teams/companies

Uncertainty and robustness

- How to specify uncertainty for transition systems, robustness for controllers, specs
- New methods for describing robustness by Tabuada et al: look at how much the specifications must be enlarged to capture new behaviors based on uncertainty

Many other directions: incremental, probabilistic, performance metrics, ...

- Identify problems where knowledge of dynamics, uncertainty and feedback matter

Wolff, Topcu, Murray
RSS 2012 (s)

Optimal Synthesis with Weighted Average Costs

Problem Setting

- Deterministic weighted transition system TS
- LTL specification ϕ
- $J(\sigma) := \limsup_{n\to\infty} \sum_{i=0}^{n} c(\sigma_i, \sigma_{i+1})$
- Problem: Compute run σ that minimizes J over all σ and satisfies ϕ.

Main Results

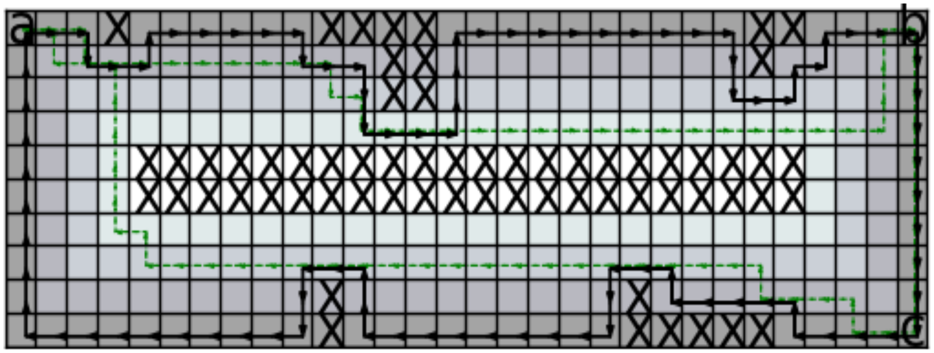
- Reduce problem to finding optimal cycle in product automaton P.
- Dynamic programming recurrence computes optimal cost cycle on P = (V,E). $F_k(v)$ is minimum cost walk of length k between vertices s, v in V.

$$F_k(v) = \min_{(u,v)\in E} [F_{k-1}(u) + c(u,v)]$$

- Complexity: $O(n_a(mn + n^2 \log(n))$ for 0-1 weights, where n_a is the number of accepting states.

Example

- Costs lower near boundary
- $\phi = \Box\Diamond a \wedge \Box\Diamond b \wedge \Box\neg x$
- Optimal (black) and feasible using DFS (green)



(shading represents cost)

Questions

- Nondeterministic transition system?
- Reactive environments?
- Multi-objective?
- Discounted cost function?

Wolff, Topcu, Murray
CDC 2012 (s)

Markov Decision Processes with LTL Specifications

Problem Setting

- Markov decision process (MDP) system model, with uncertainty in transitions (disturbances, failures)
- LTL specification ϕ (probably GR(1))
- Problem: Maximize probability of MDP satisfying ϕ over uncertainty set:

$$\max_{\pi\in\Pi}\ \min_{\tau\in\mathcal{T}} \mathbb{P}^{\pi,\tau}(s_0 \models \varphi)$$

Main Results

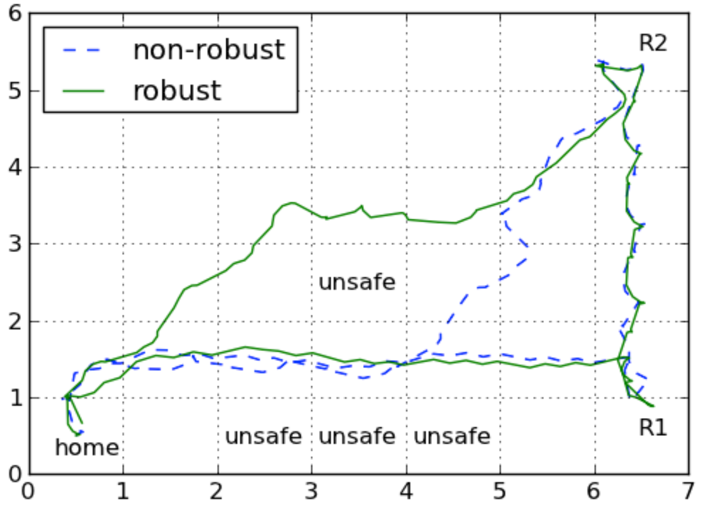
- Transform P = MDP x LTL to stochastic shortest path (SSP) form
- Compute satisfaction probabilities on SSP with robust dynamic program’g

$$(TJ)(s) \ := \ \min_{a\in A(s)} \left[c(s,a) + \max_{p\in\mathcal{P}_s^a} p^T J\right]$$

- Project policy π back to MDP
- Complexity: O(n2m log(1/ε)2) for ε-suboptimal policy

Example

- Differential drive robot (x,y,theta)
- Transition probabilities estimated (MC)
- ϕ = ◊(R1 ∧ ◊R2) ∧ □¬unsafe ∧
  ◊□ home



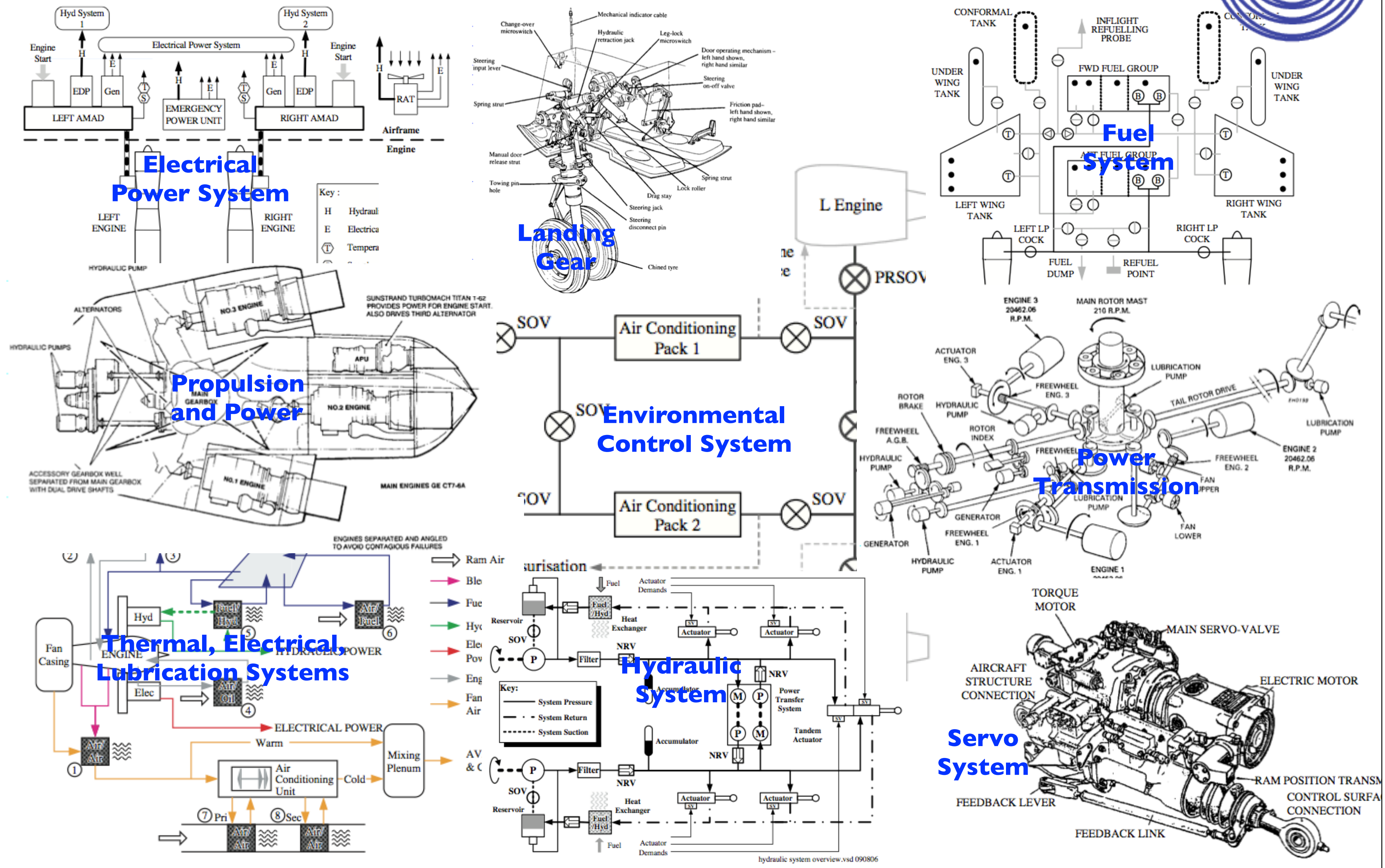
Questions

- Simpler fragments of temporal logic?
- Tradeoffs between costs and probability of success?
- Principled abstraction of MDPs from continuous systems?

Technical Challenges and Risks

1. **Writing LTL (or other temporal logic) specifications is not a job for mortals**
    - Easy to make mistakes when writing LTL and hard to interpret complex formulas
    - Possible approach: domain specific tools that provide engineer-friendly interface
2. **Model checking and logic synthesis tools won't work on large problems**
    - Combinatorial explosion in discrete states for modest engineering systems will make it impossible to apply model checking/synthesis to "raw" problem
    - Approaches: abstraction layers and modularity via interfaces
        - Vertical layering: apply tools to different layers and enforce bisimulation
        - Horizontal contracts: define formal subsystem interfaces & reason about them
3. **Expertise in modeling and specification not yet developed**
    - Engineers in domains in which these tools are needed don't yet have experience developing models that ignore the right sets of things
        - Compare to reduced order models for aircraft (aerodynamic, aeroelastic) and agreed on specifications (bandwidth, response time, stability margins, etc)
        - Particularly worried about dynamics, uncertainty, interconnection
        - How do we convince FAA to allow use of these tools?
    - Approach (?): explore application domains, moving from modest to complex problems, and develop expertise, tools, tool chains, processes, ...

Aircraft Vehicle Management Systems



Electrical Power System

Landing Gear

Fuel System

Propulsion and Power

Environmental Control System

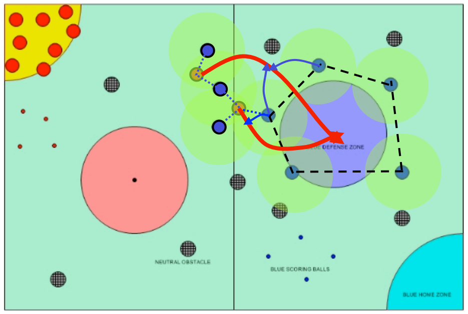
Power Transmission

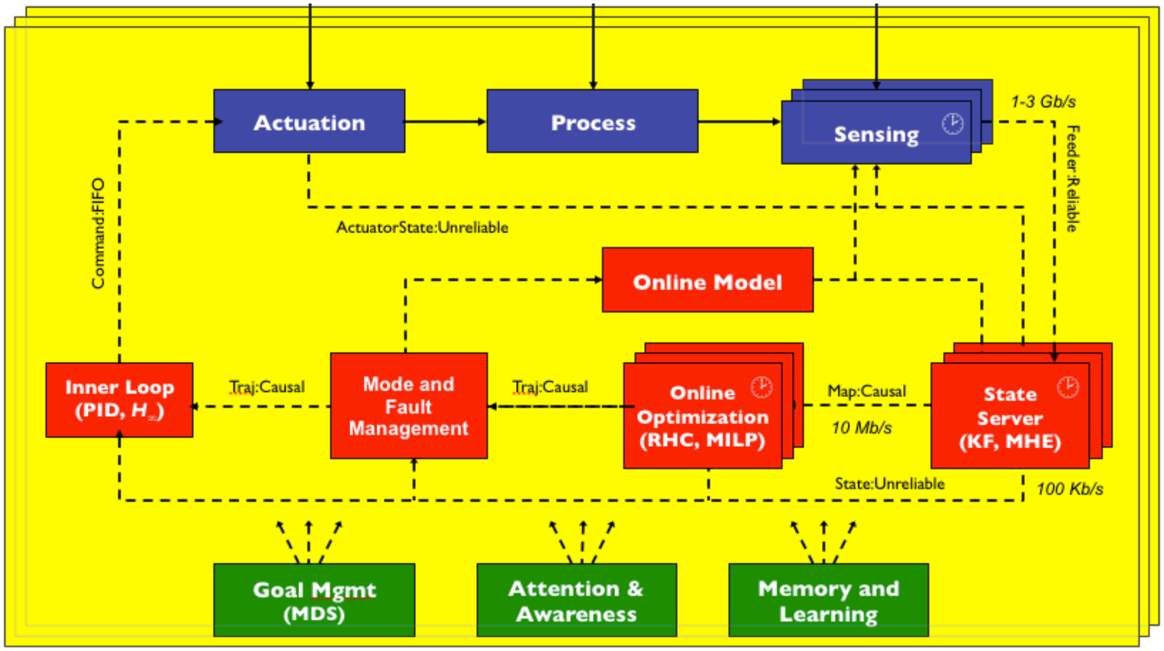
Thermal, Electrical, Lubrication Systems

Hydraulic System

Servo System

Specification, Design and Verification of NCS





Specification

- How do we describe correct behavior?

Design

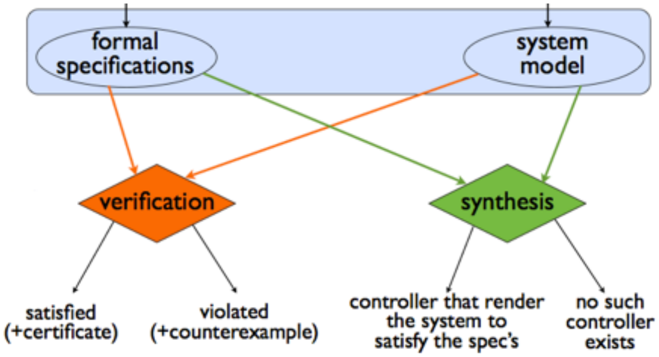
- What tools can we use to design protocols to implement that behavior?

Verification

- How do we know if it is actually correct?

$$J = \int_0^T L(x, \alpha, u)\, dt + V(x(T), \alpha(T)),$$

$$(\varphi_{init} \ \wedge \ \Box\varphi_e) \implies (\Box\varphi_s \ \wedge \ \Diamond\varphi_g)$$


formal specifications
system model
verification
synthesis
satisfied (+certificate)
violated (+counterexample)
controller that render the system to satisfy the spec's
no such controller exists