# Lecture 4.
# Applications

# Summary

**Tools such as HyTech, CheckMate, Uppaal, Kronos have been used in many contexts**

typically to verify safety of a control design or to get tight bounds on parameters (e.g. steam boiler, audio control)

**This lecture shows where hybrid systems theory can fit in some application domains**

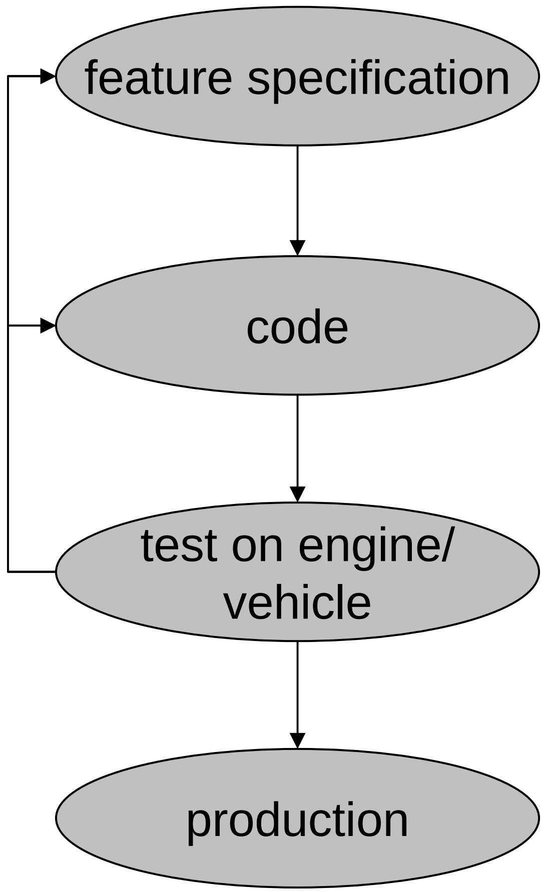
# Applications Outline

- ➲ Embedded Control Systems
- ❑ Autonomous Mobile Robots
- ❑ Biological Systems

# Embedded Controller Development Process For Automobile Transmissions

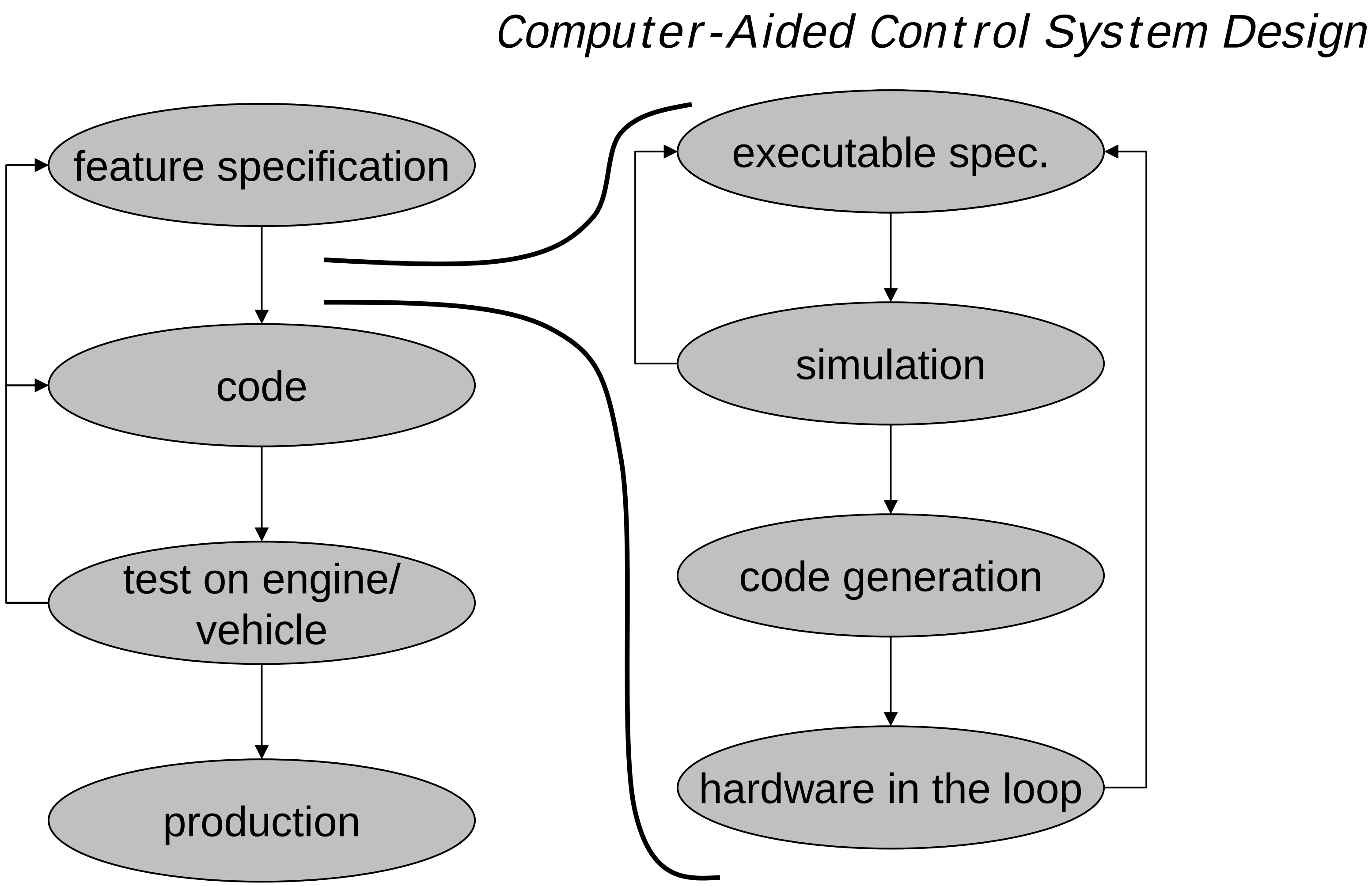
*Former Practice*

- redesign
- fix bugs

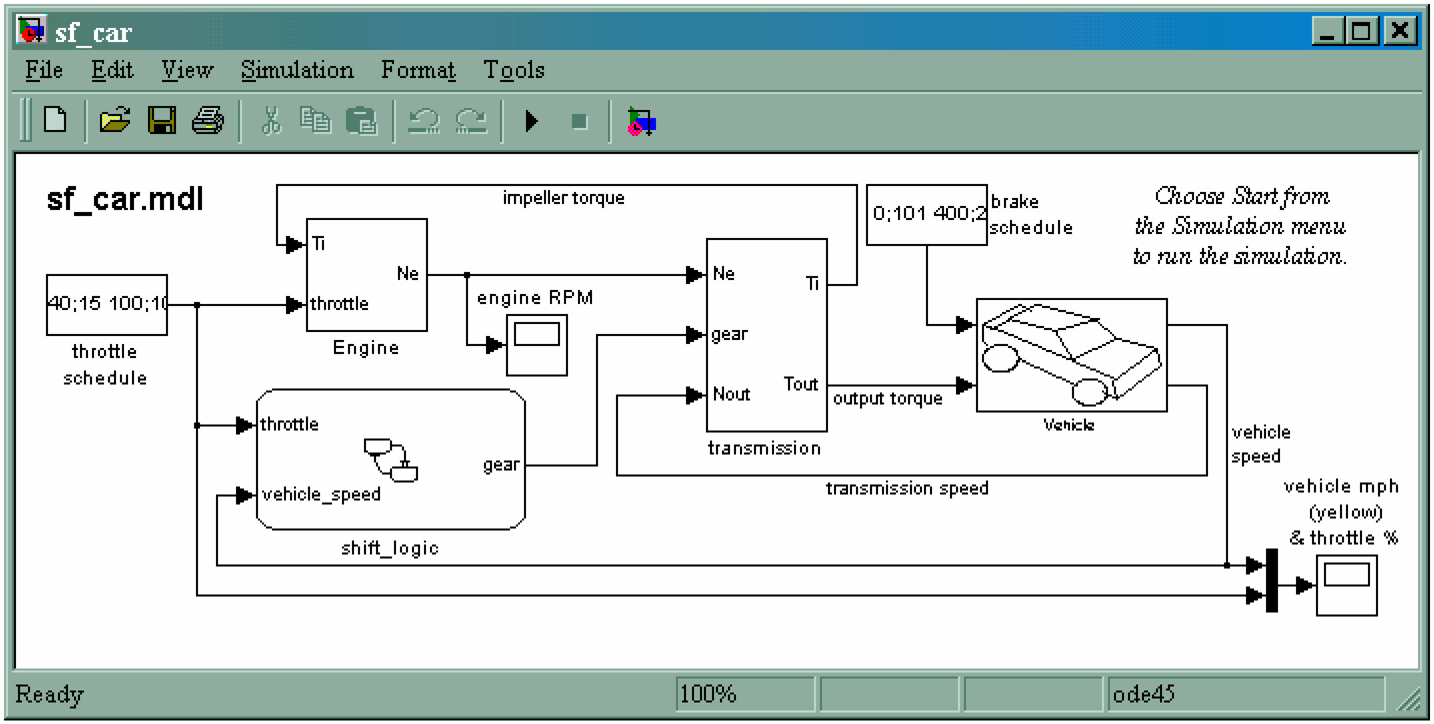



- describe control logic using relay-ladder diagrams
- write assembly code for the control microprocessor
- drive the car around the test track
- hope for the best

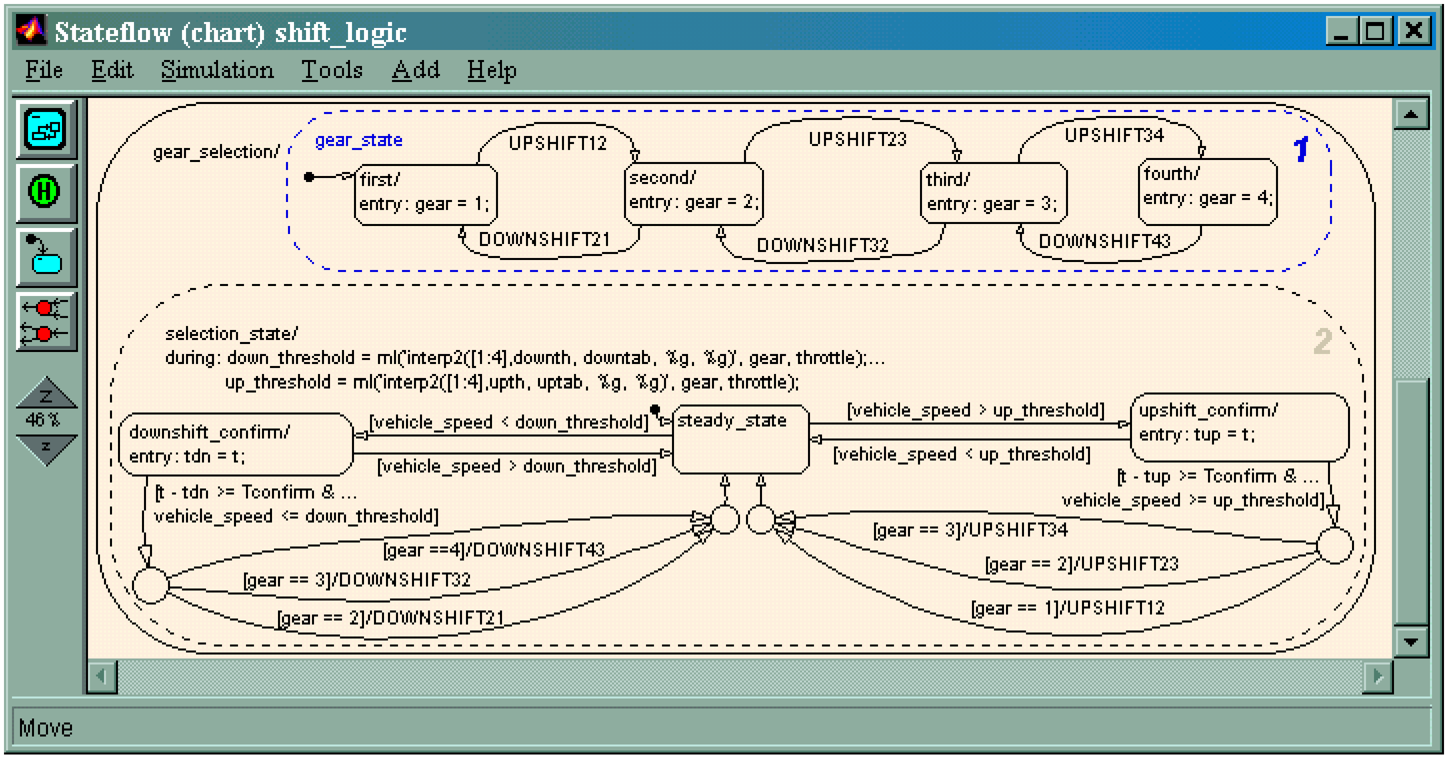
# Automotive Embedded Controller Design: State of the Practice



# Executable Specifications Using MATLAB/Simulink/Stateflow

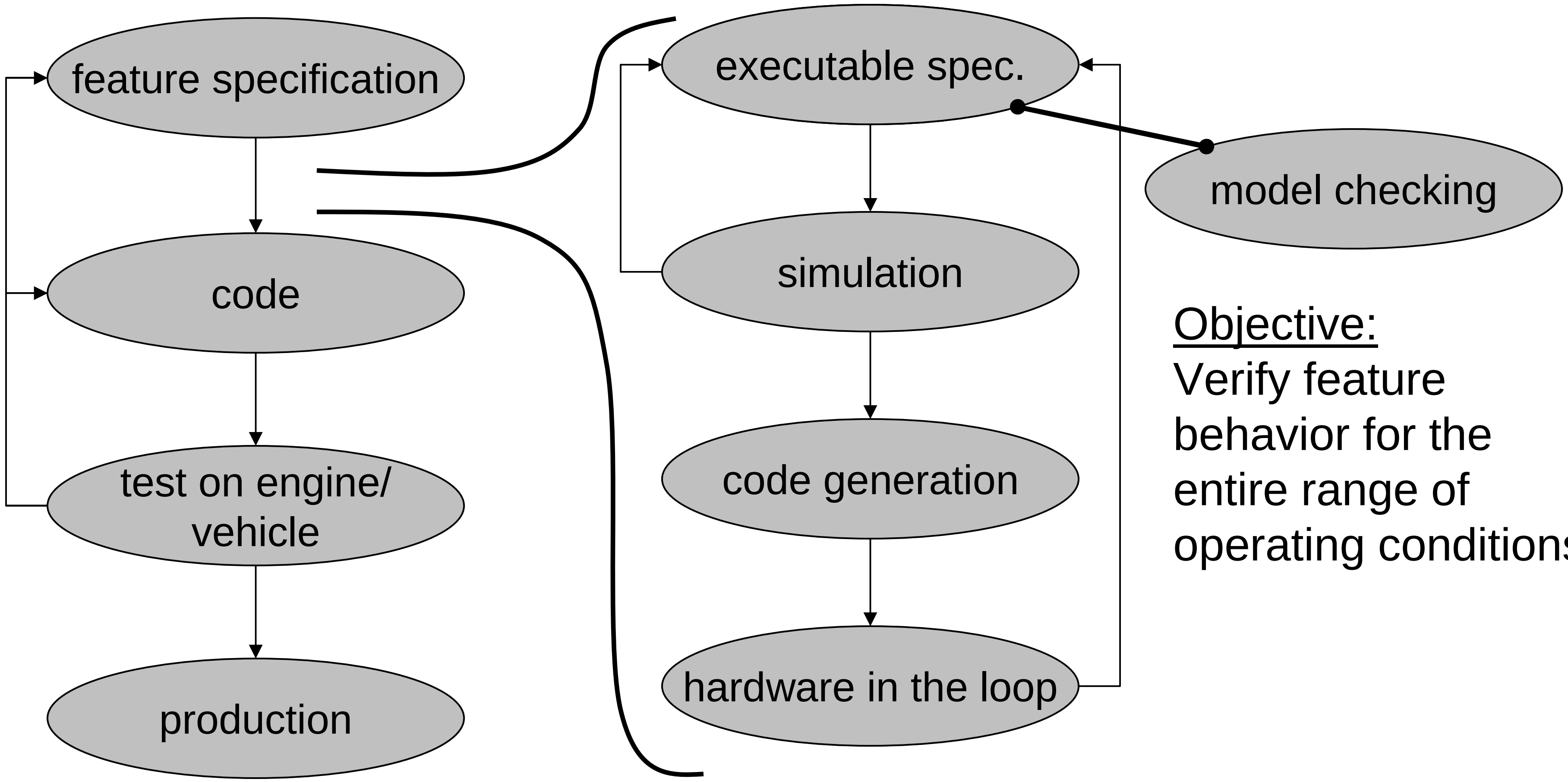


# Transmission Control Logic



# Opportunity to Apply Formal Verification Techniques

*Computer-Aided Control System Design*




Objective:
Verify feature behavior for the entire range of operating conditions.

# Automotive Engine Control in Cut-off Mode

**Control law:** Decide when to inject air/fuel for torque to minimize acceleration peaks during the cut-off operation.

**Problem:** Verify the event-driven implementation of a control law designed in continuous time.

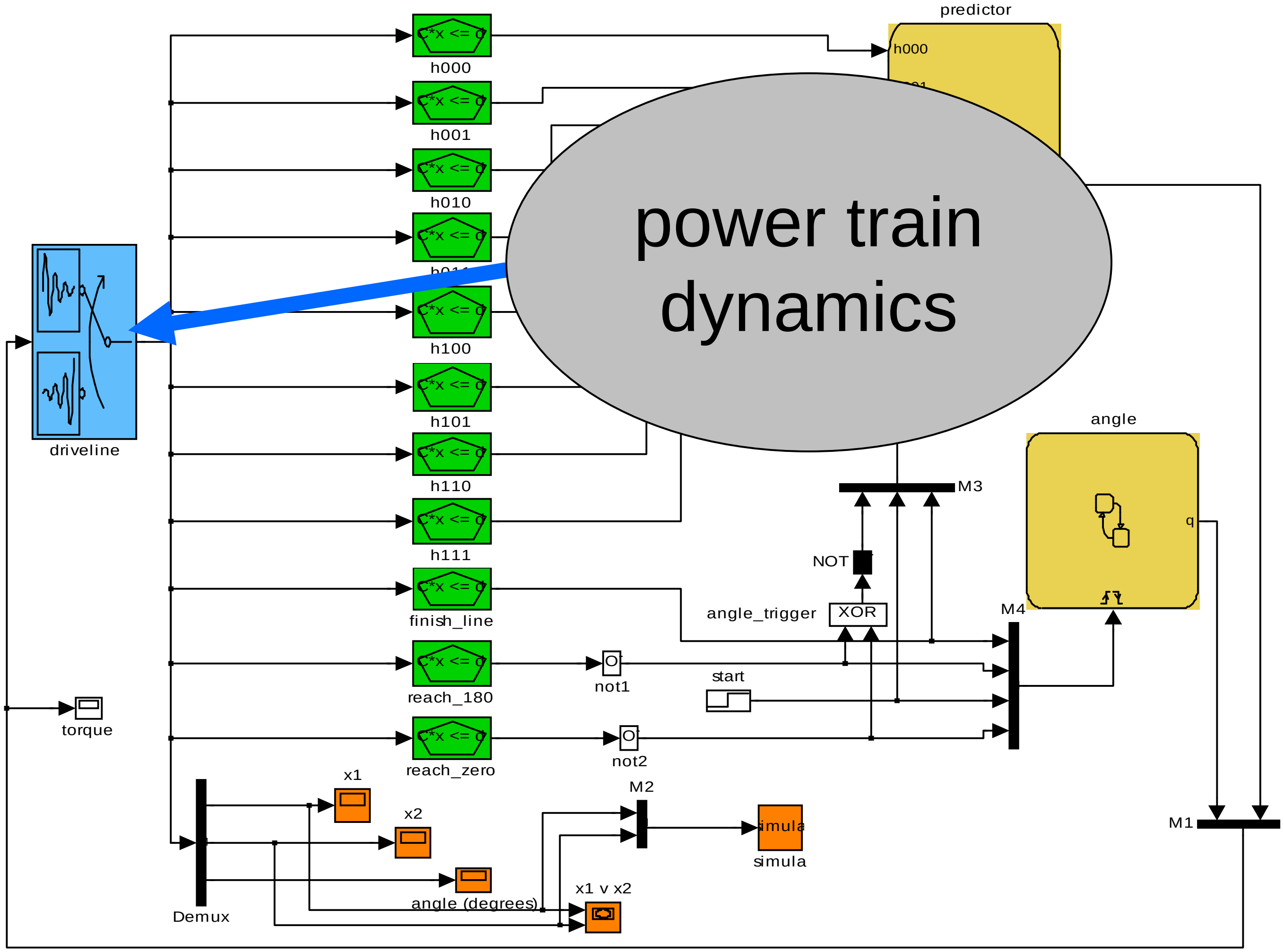
Application of CheckMate due to Krogh et al

# Automotive Powertrain Model

Model from Magneti Marelli Engine Control Division

- Four-stroke, four cylinder engine
- Continuous-time powertrain model
- Hybrid model for cylinder cycles

# CheckMate Model



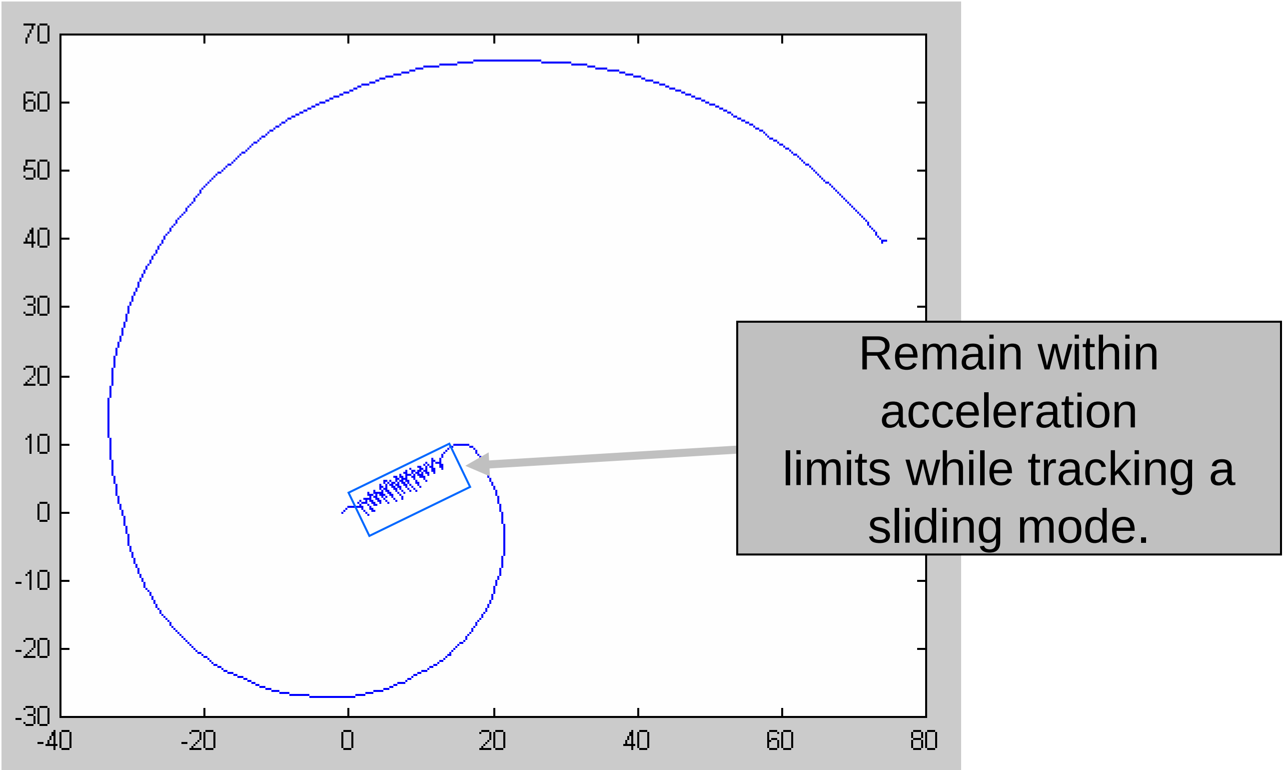
# Continuous Dynamics

$\dot{x} = Ax + Bu$ $\quad$ u = 0 (no air-fuel) or 10

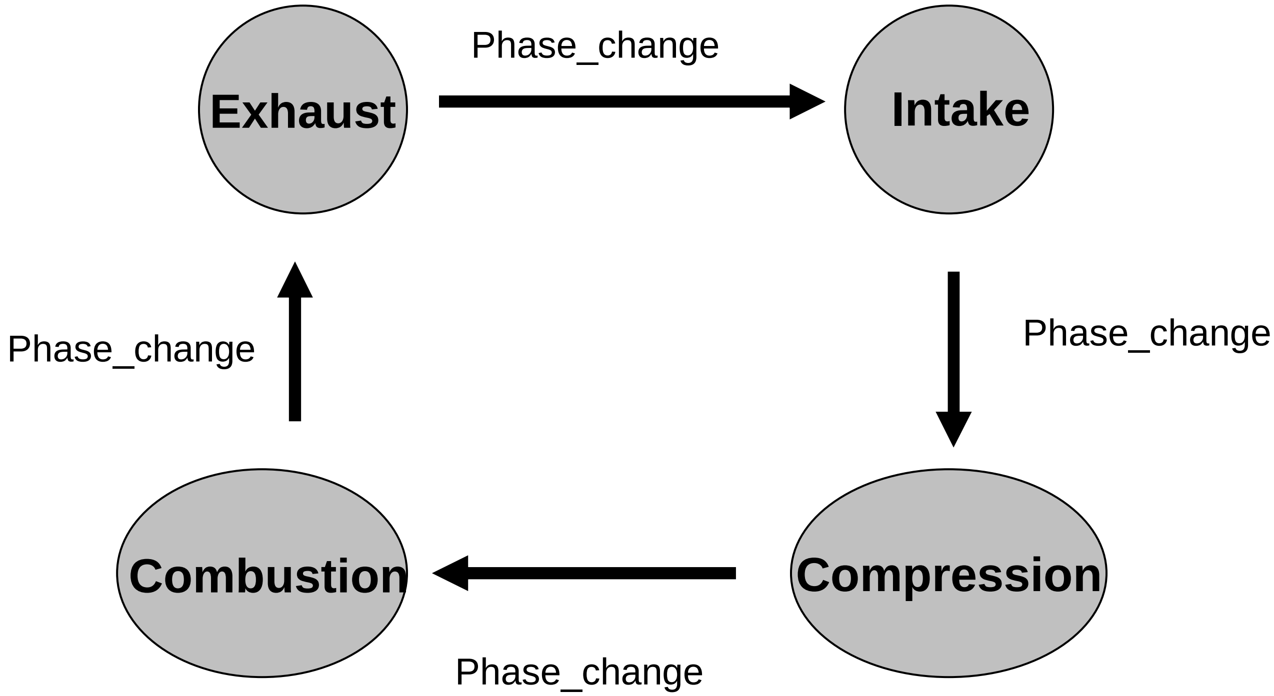
$x_1$ = engine block angle
$x_2$ = wheel revolution speed (radians)
$x_3$ = axle torsion angle (in radians)
$x_4$ = crankshaft revolution speed (rpm)
$x_5$ = crankshaft angle (degrees)

# Controller Specification

- Sliding mode control law derived in continuous time
- Hybrid implementation due to discrete torque decisions

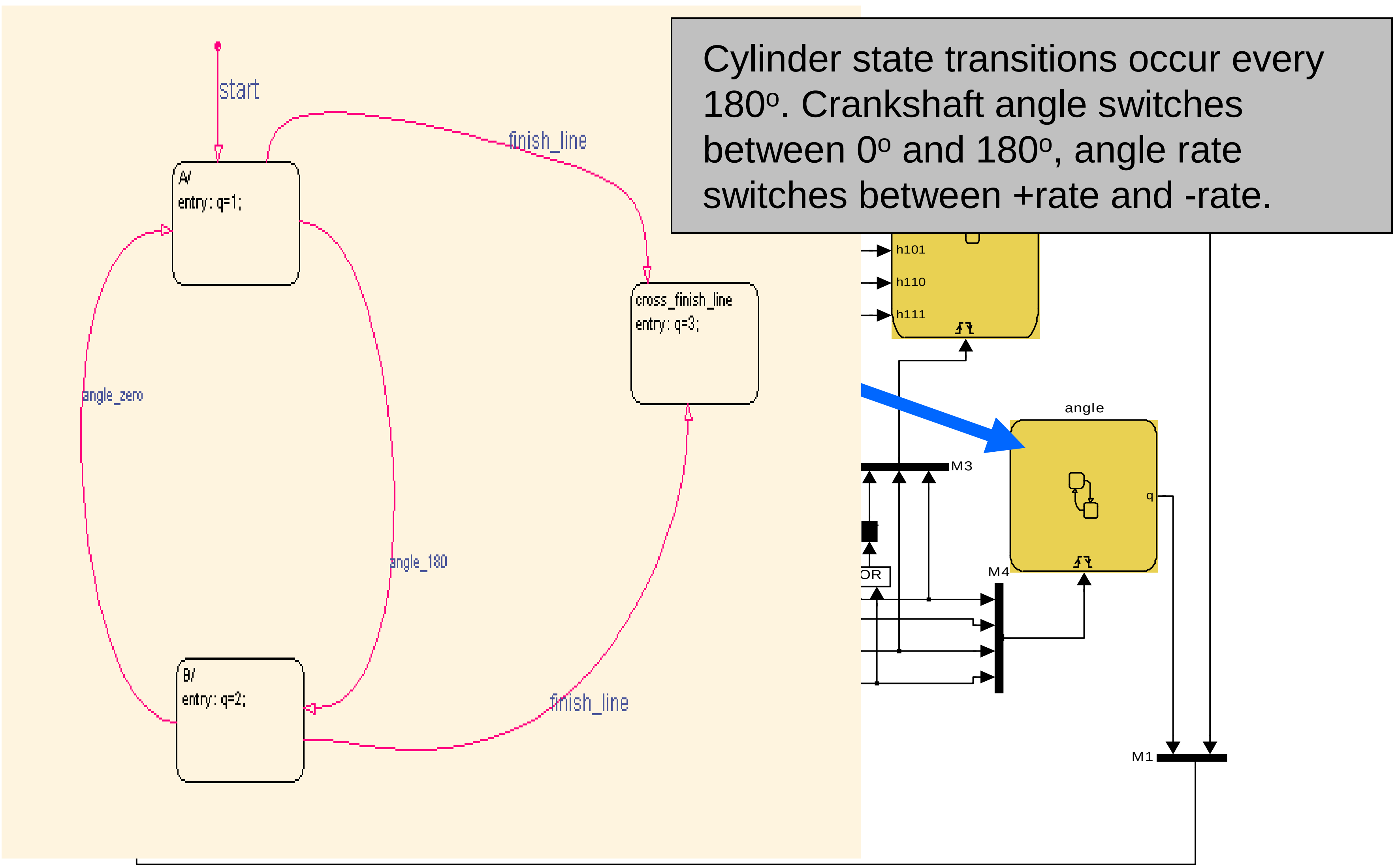


# Cylinder Cycle



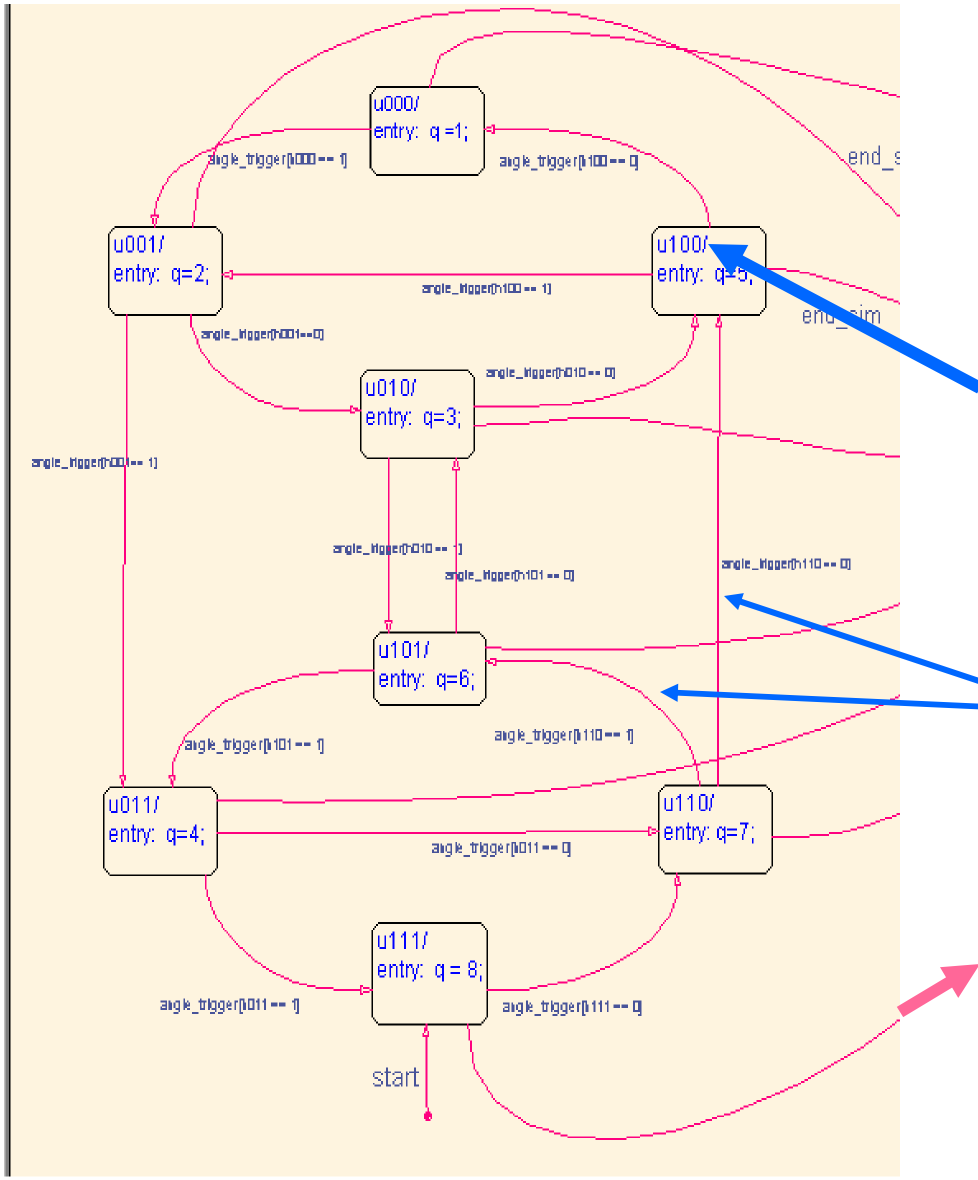

Control decision to apply torque on the power stroke must be made before the intake stroke $\Rightarrow$ three step lookahead.

# Crankshaft Angle Rate Logic




Cylinder state transitions occur every 180º. Crankshaft angle switches between 0º and 180º, angle rate switches between +rate and -rate.

# Predictive Control Logic

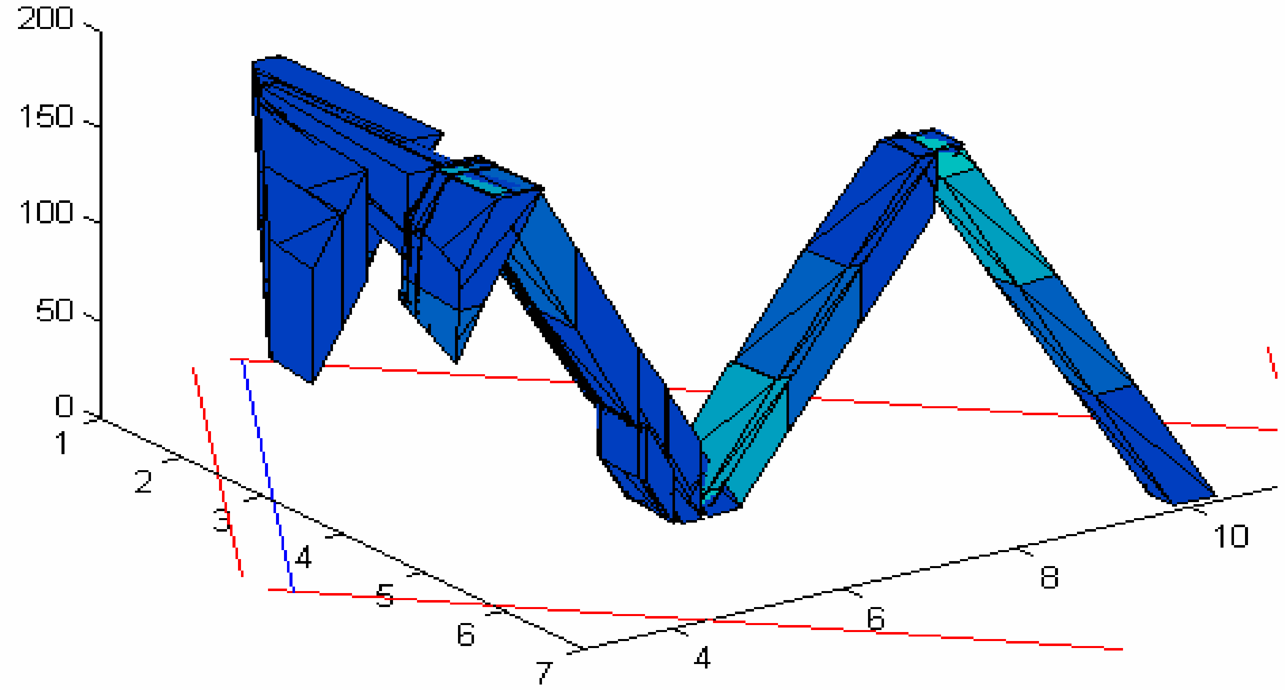



The discrete state indicates the torque decisions for the current and next two power strokes (i.e., for three of the four cylinders).

Transitions from each state depend on whether predicted state for the next power stroke is closer to the sliding mode with or without torque.

The 9th state (not shown) is the “end simulation” state--reachable from any of the other 8 states.

# Flowpipe for One Discrete Sequence



# Applications Outline

- ✓ Embedded Control Systems
- ➲ Autonomous Mobile Robots
- ❑ Biological Systems

# Programming Interacting Autonomous Robots

**Many modes**

Individual modes are well understood, but not their interaction.

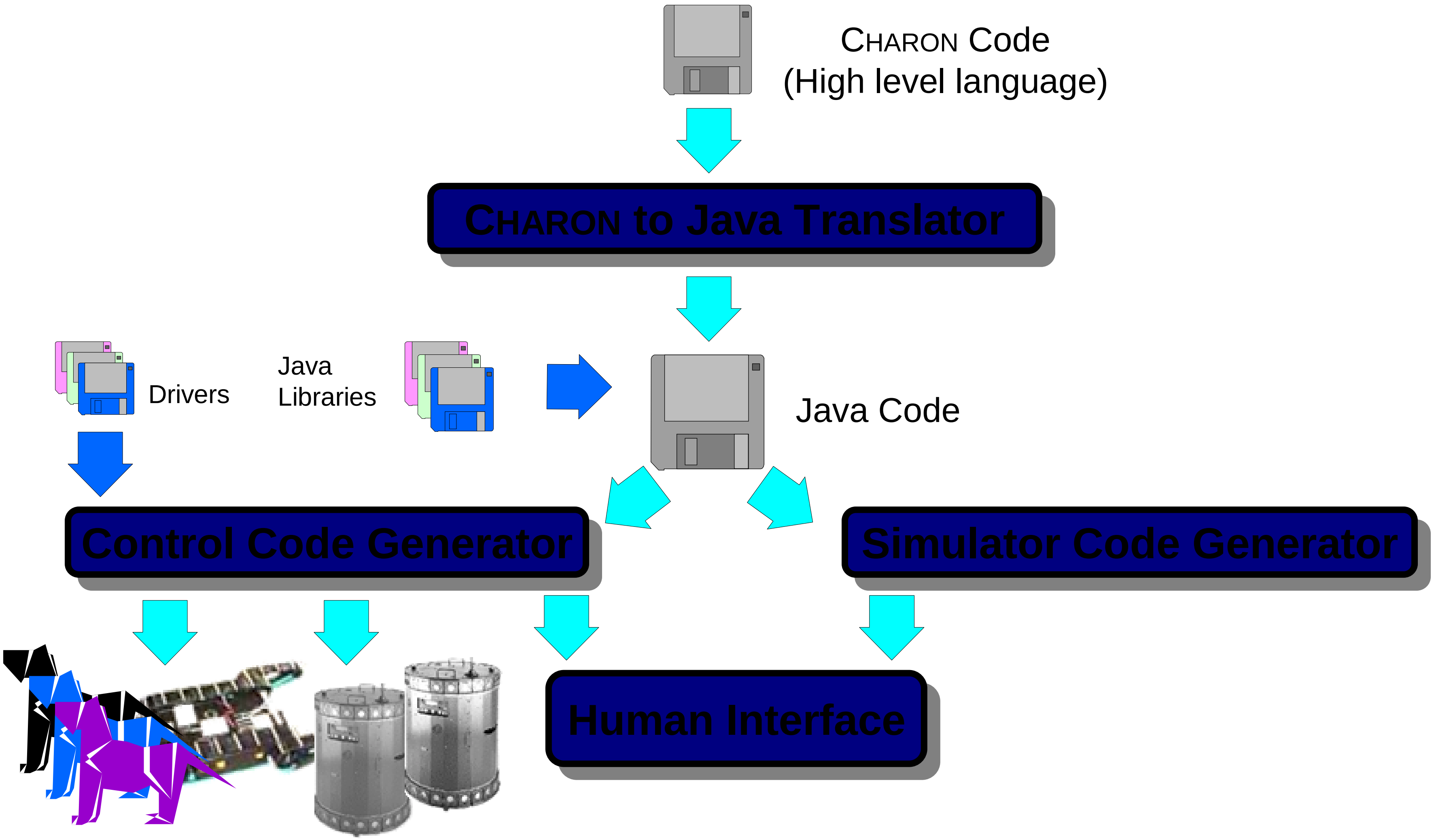
**Software design**

Modes designed bottom-up
Protocols top-down

**Modular design to ensure reusability**

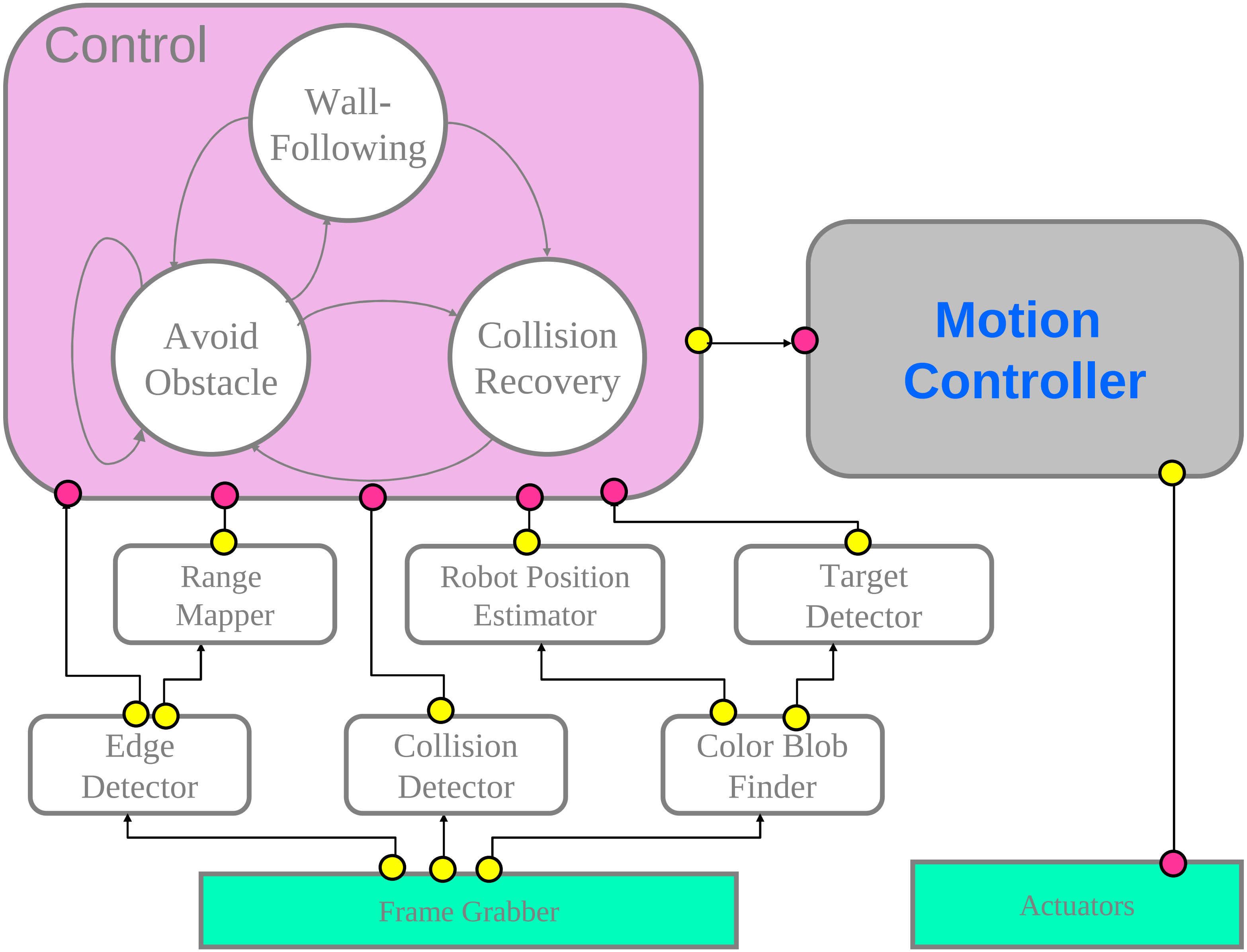
**Tasks: Formation control, cooperative control**

# Software Design Methodology

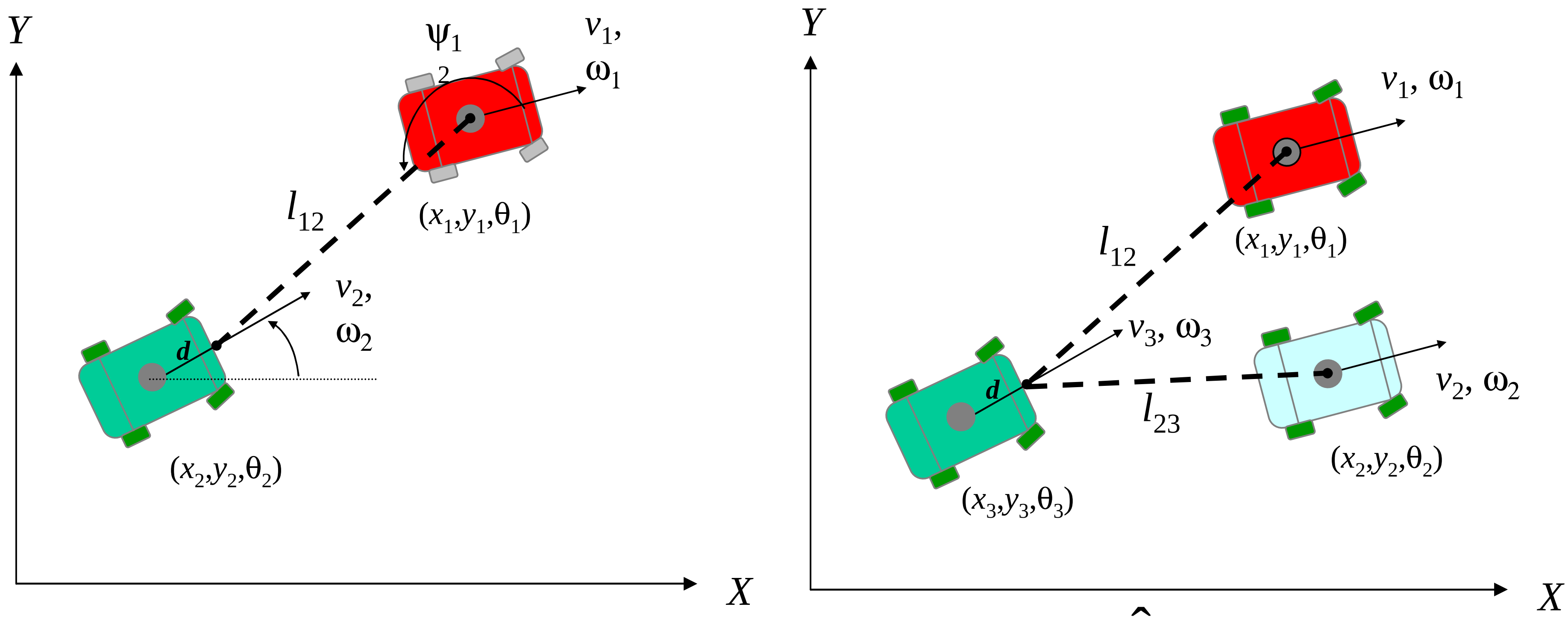


# Vision-Based Control: Mode Switching

# Reactive Vision Based Controllers



# Controllers for Maintaining Formation

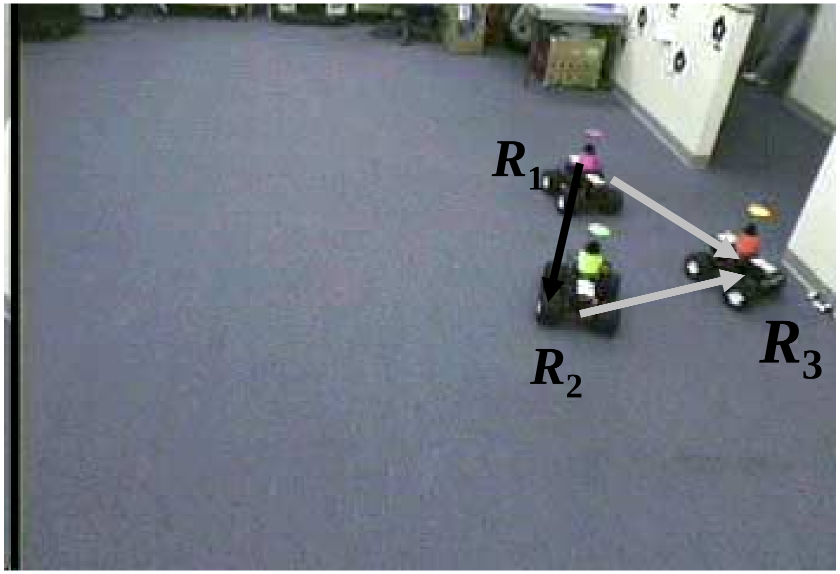



$$v_j = s_{ij}\cos\gamma_{ij} - l_{ij}\sin\gamma_{ij}(b_{ij} + \hat{\omega}_i) + \hat{v}_i\cos(\hat{\theta}_i - \theta_j)$$

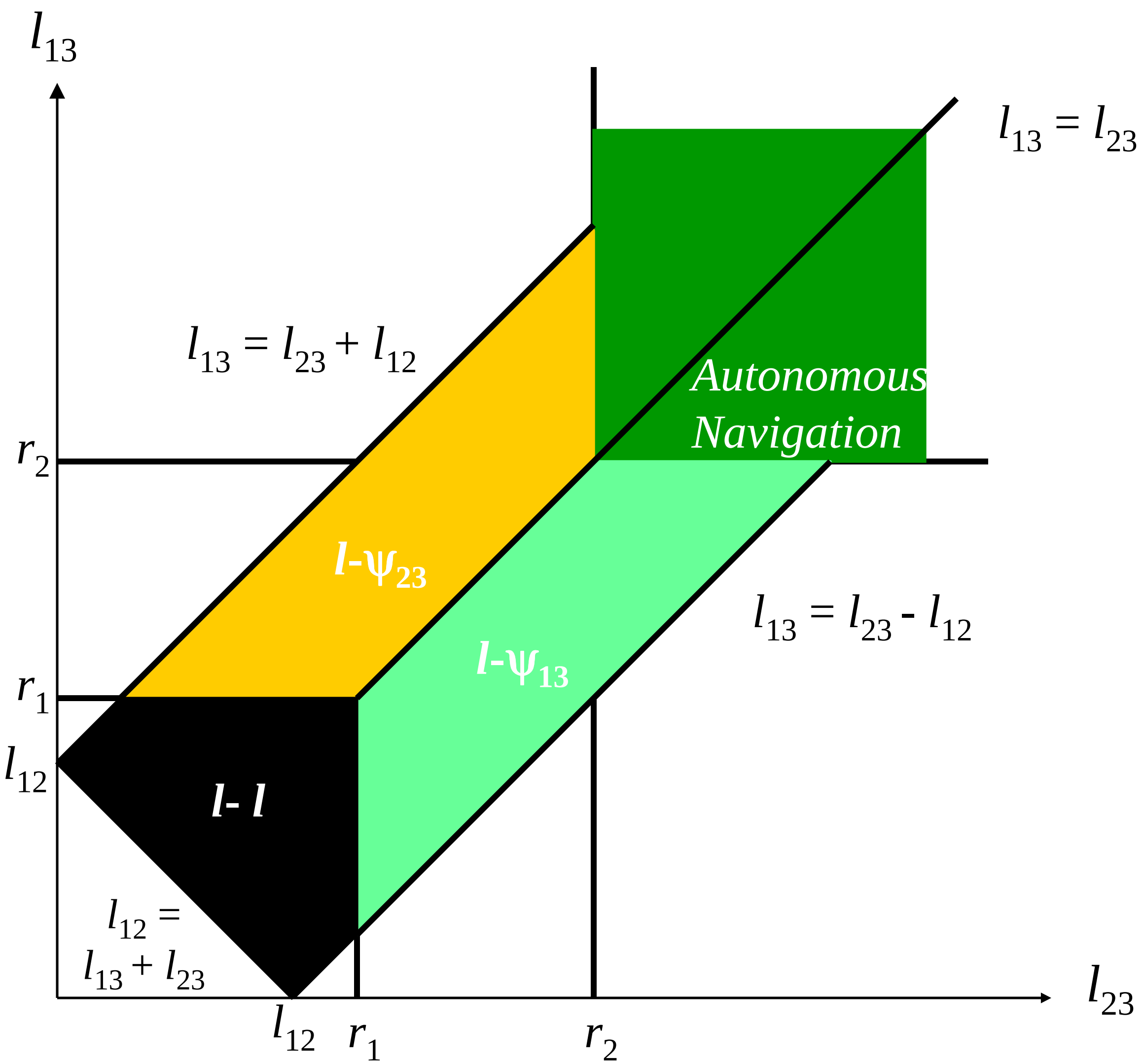
$$\omega_j = \frac{1}{d_j}[s_{ij}\sin\gamma_{ij} + l_{ij}\cos\gamma_{ij}(b_{ij} + \hat{\omega}_i) + \hat{v}_i\sin(\hat{\theta}_i - \theta_j)]$$

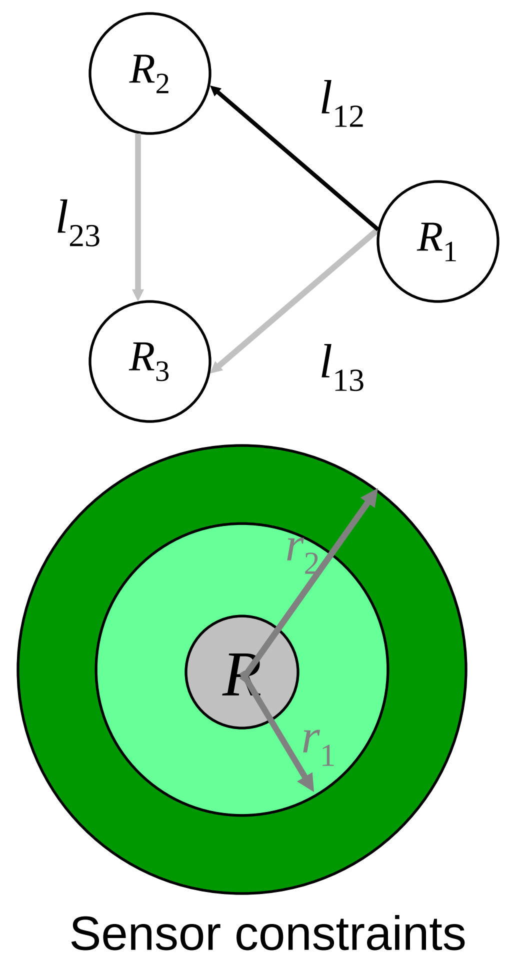
- Estimate of relative position, orientation, linear and angular velocities required.

# Multirobot Coordination




# Rules for Mode Switching

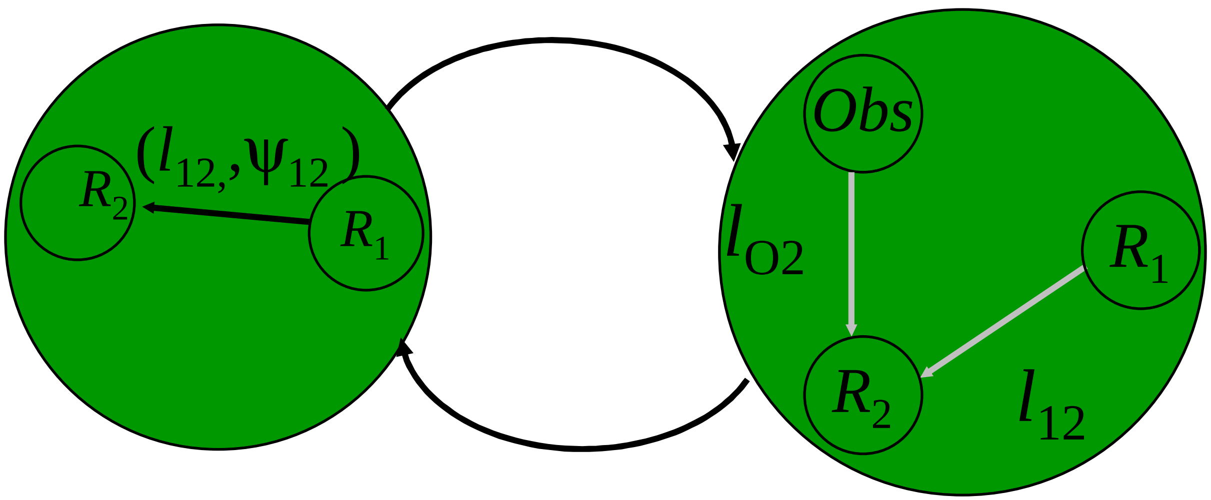


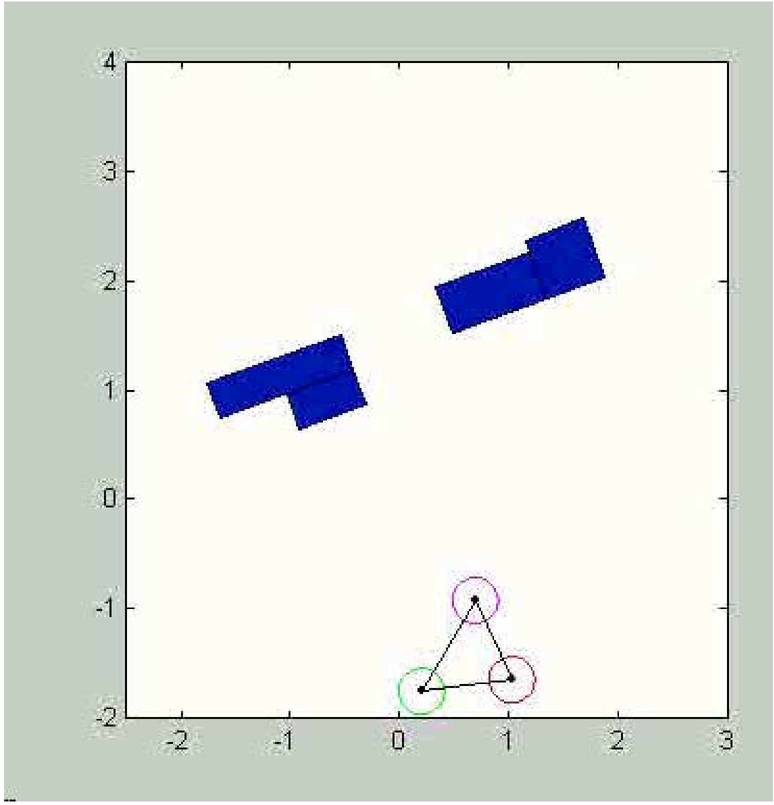
# Leader Follower and Obstacle Avoidance

Leader is teleoperated

Leader is autonomous



# Mode Switching and Maintain Formation



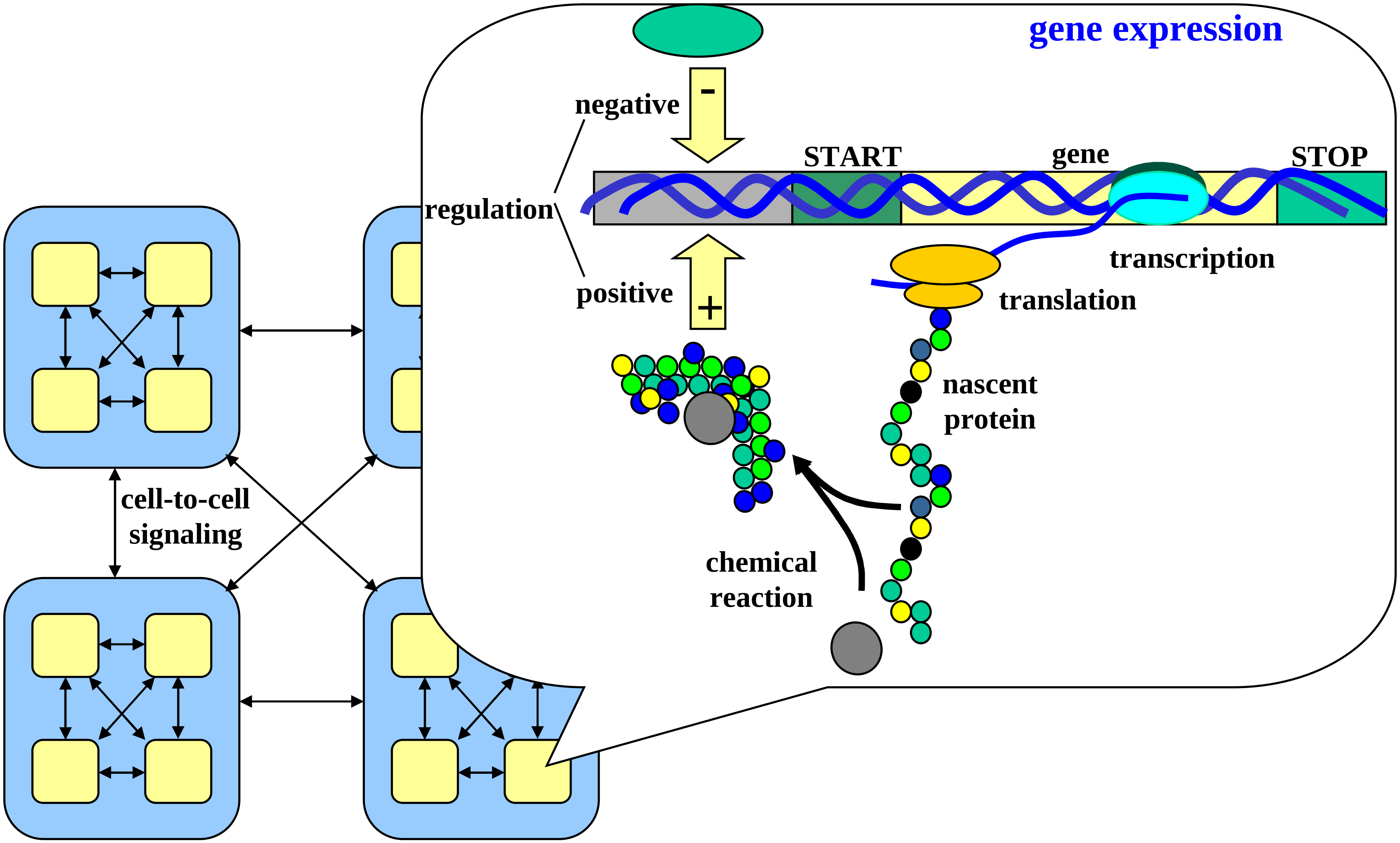
# Applications Outline

- ✓ Embedded Control Systems
- ✓ Autonomous Mobile Robots
- ➲ Biological Systems

# Cellular Networks

- Networks of interacting biomolecules carry out many essential functions in living cells (gene regulation, protein production)
- Both positive and negative feedback loops
- Design principles poorly understood
- Large amounts of data is becoming available
- Beyond Human Genome: Behavioral models of cellular networks
- Modeling becoming increasingly relevant as an aid to narrow the space of experiments

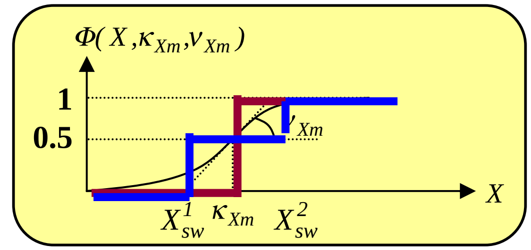
# Regulatory Networks

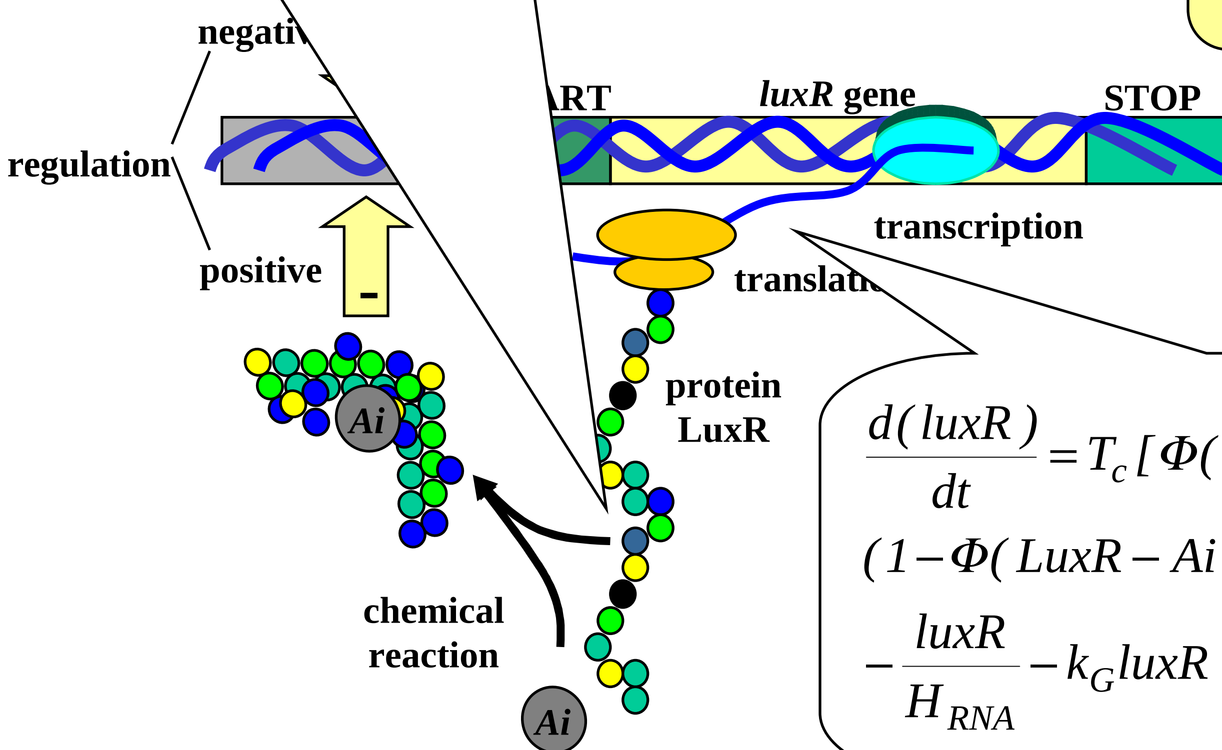


# Hybrid Modeling of Biological Systems

…ng smooth functions.

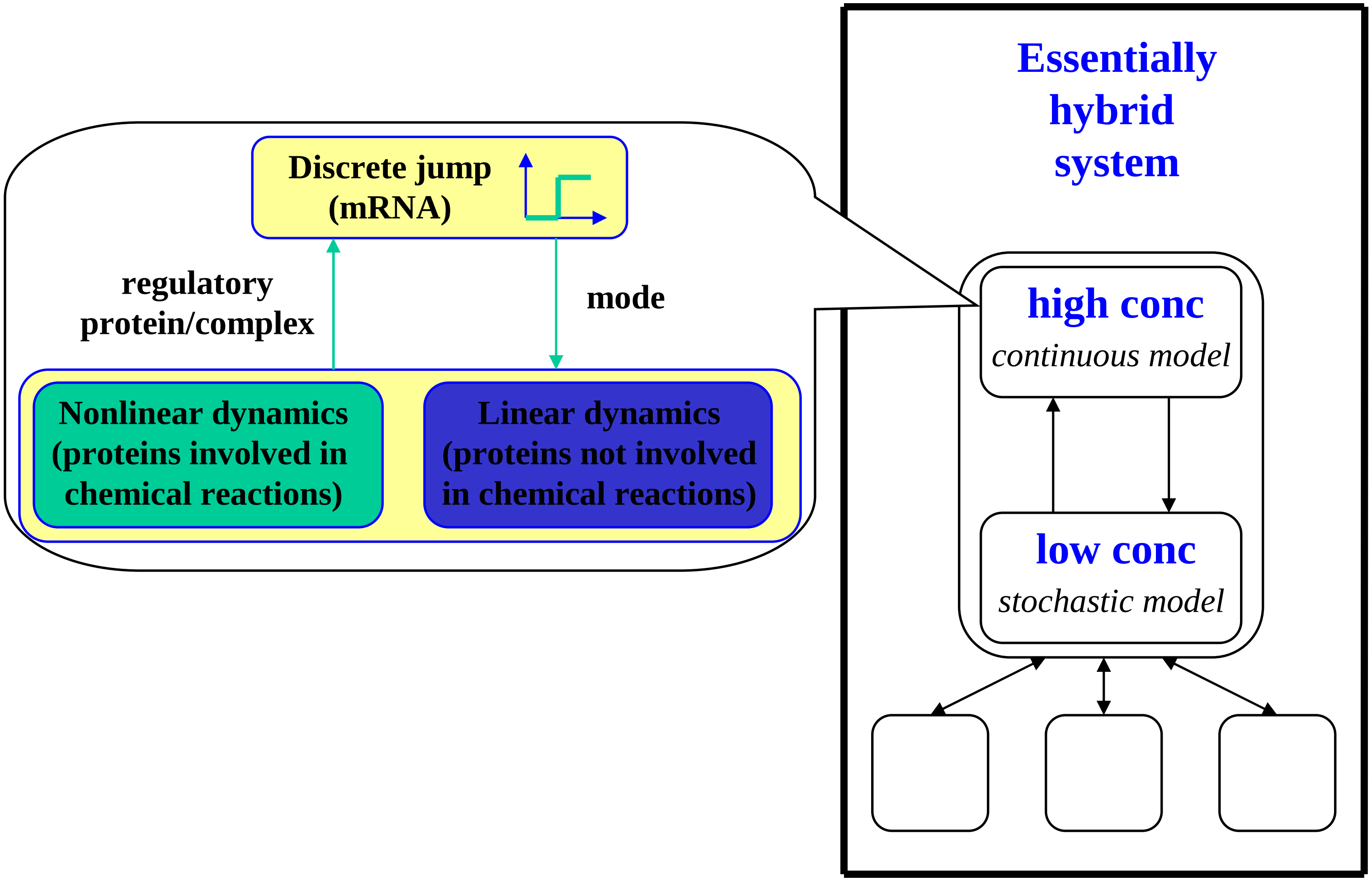
$$\frac{d(LuxR)}{dt} = T_l\ luxR - \frac{LuxR}{H_{sp}} - r^b_{LuxR/Ai}\ Ai\ LuxR + r^d_{LuxR/Ai}\ Co - k_G\ LuxR$$




$$\frac{d(luxR)}{dt} = T_c[\Phi(CRP, \kappa_{CRP}, \nu_{CRP})(1 - \Phi(LuxR-Ai, \kappa_{LuxR-Ai}, \nu_{LuxR-Ai})) + b) - \frac{luxR}{H_{RNA}} - k_G luxR$$

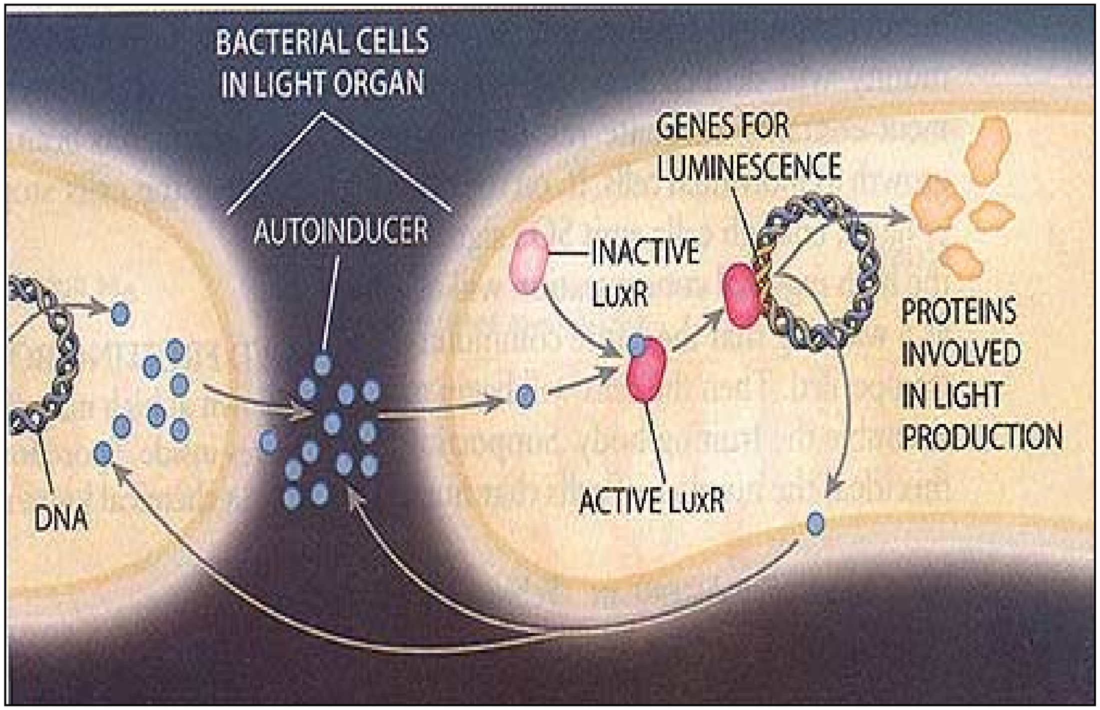
# Hybrid Modeling



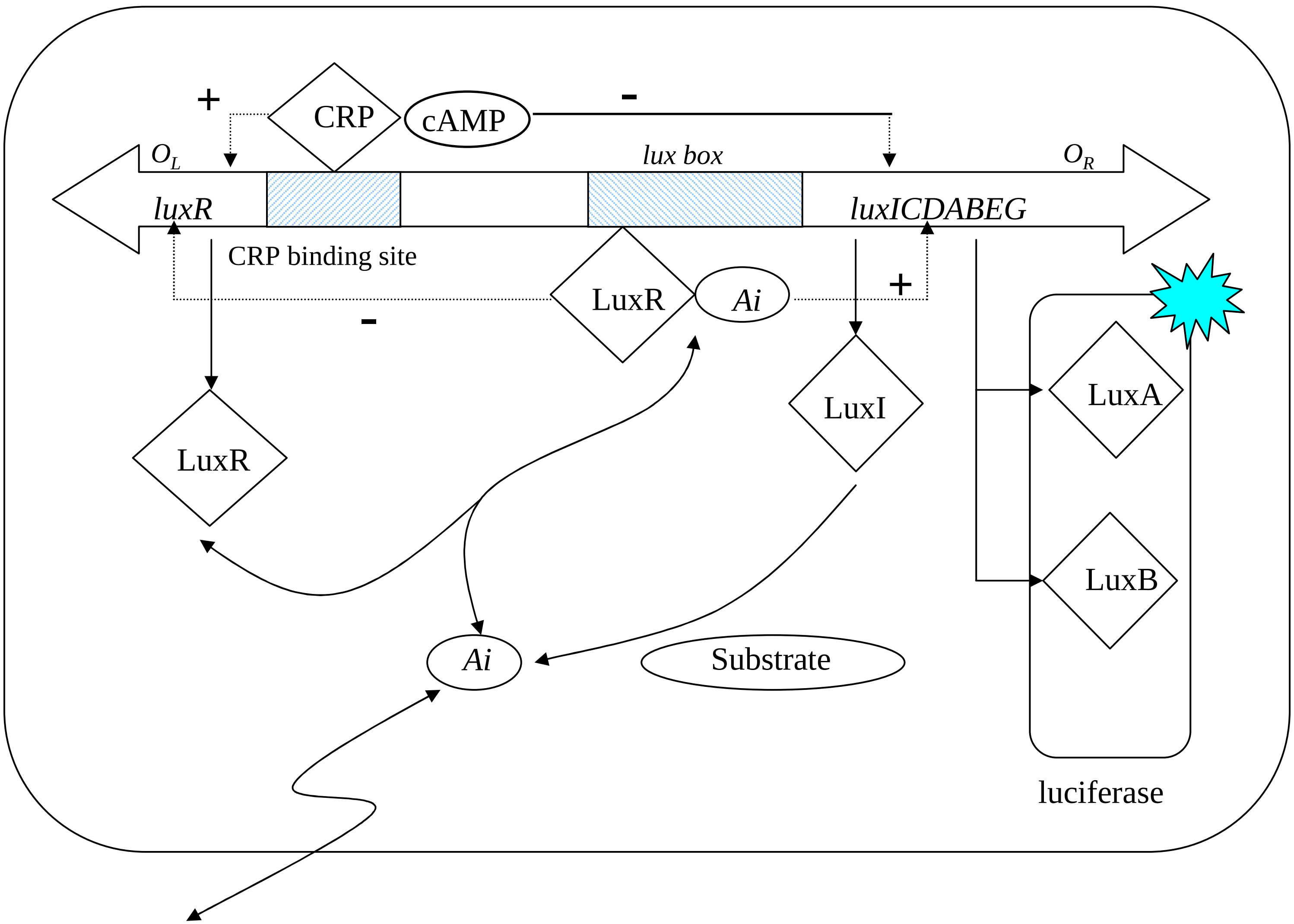

At low concentrations, a continuous approximation model might not be appropriate. Instead, a stochastic model should be used.

In some cases, the biological description of a system is itself hybrid.

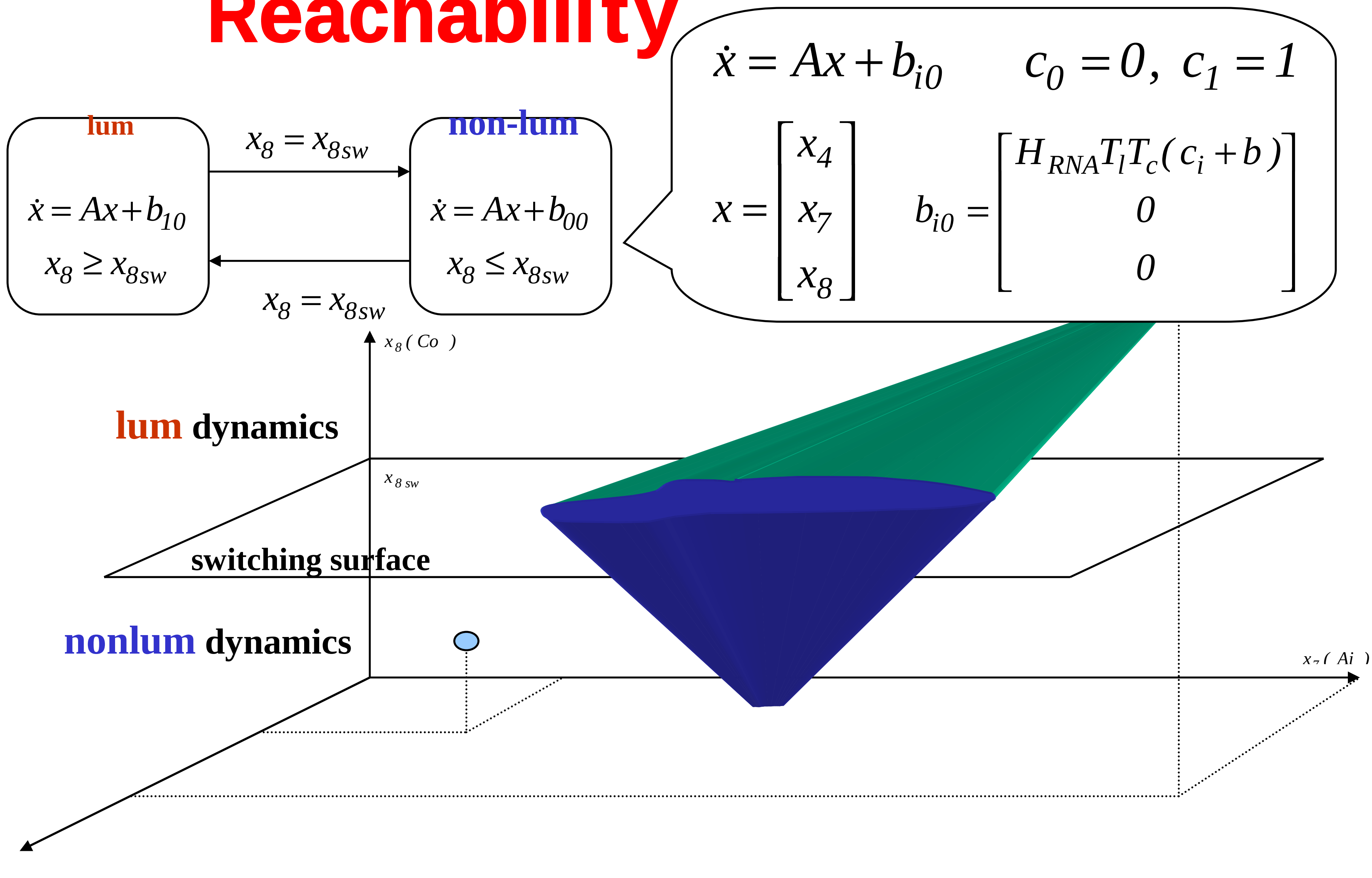
# Luminescence / Quorum Sensing in *Vibrio Fischeri*



# Luminescence Regulation



# Reachability



# Simulation Results

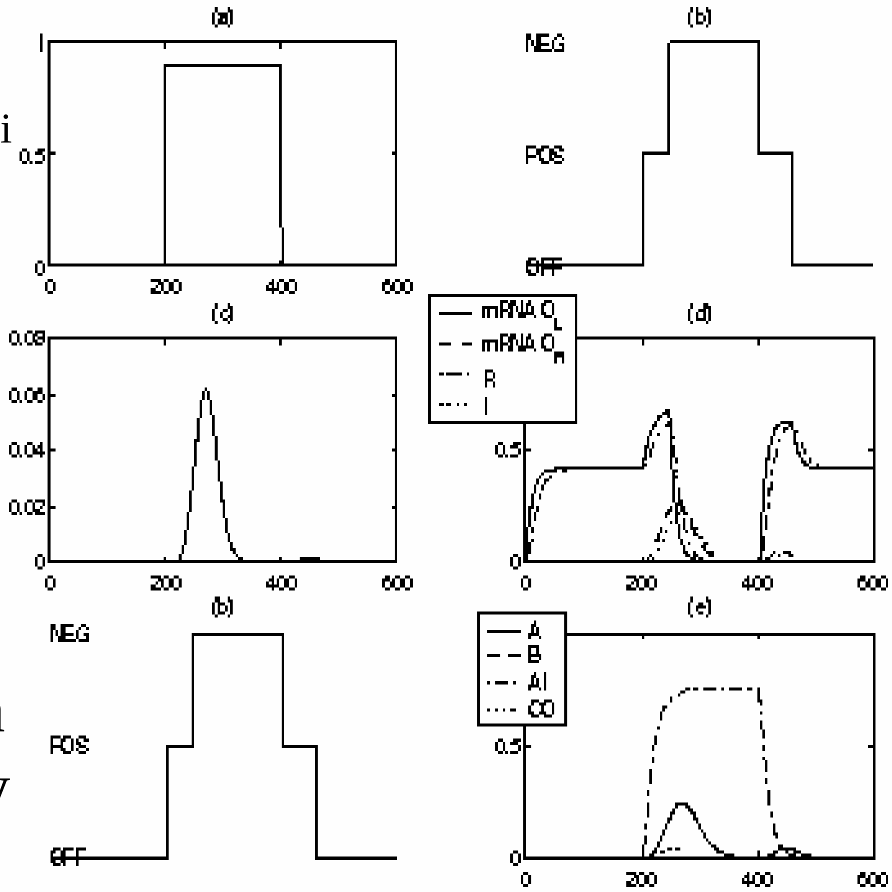
external $A_i$
(input)

switch
history

luminesence
(output)

concentrations
for various
entities

switch
history



# Summary

- Hybrid systems are necessary to model some biological regulatory networks.
- The simulation results of the luminescence control in Vibrio fischeri are in accordance with phenomena observed in experiments.
- Modeling concepts such as hierarchy, concurrency, reuse, are relevant for modular specifications
- Exploiting the structure of real biological systems will be essential to meet the challenge posed by the enormous complexity of biological regulatory networks.

# Conclusions

- A rich variety of domains match hybrid systems paradigm
- Traditional benefits: safety verification, design of hybrid controllers
- Formal models can be beneficial in more ways: modeling, understanding, programming, simulation
- Emerging potential for integration with software engineering design tools