Thesis Seminar

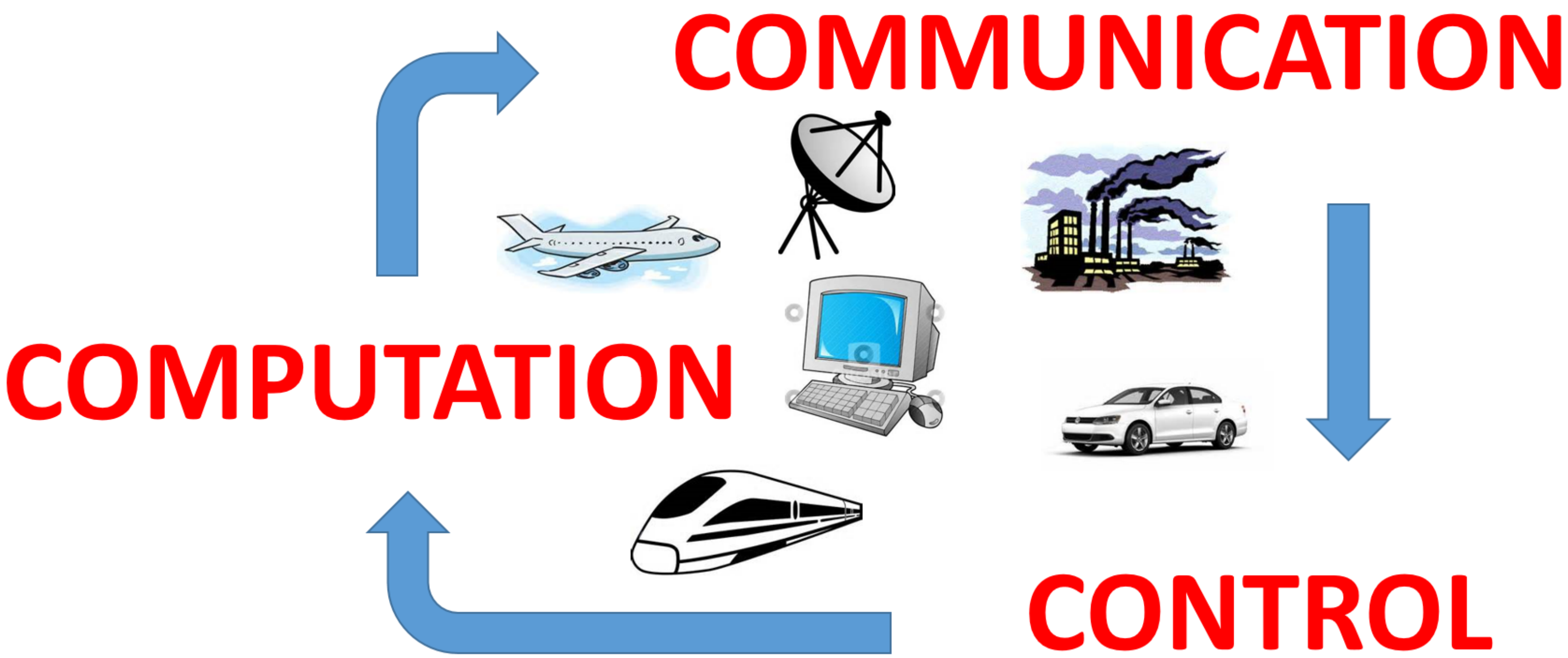
Adaptive Task Scheduling for Cyber-Physical Control Systems

Xiaotian Dai
Supervised by Alan Burns

Real-Time Systems Group, University of York, UK

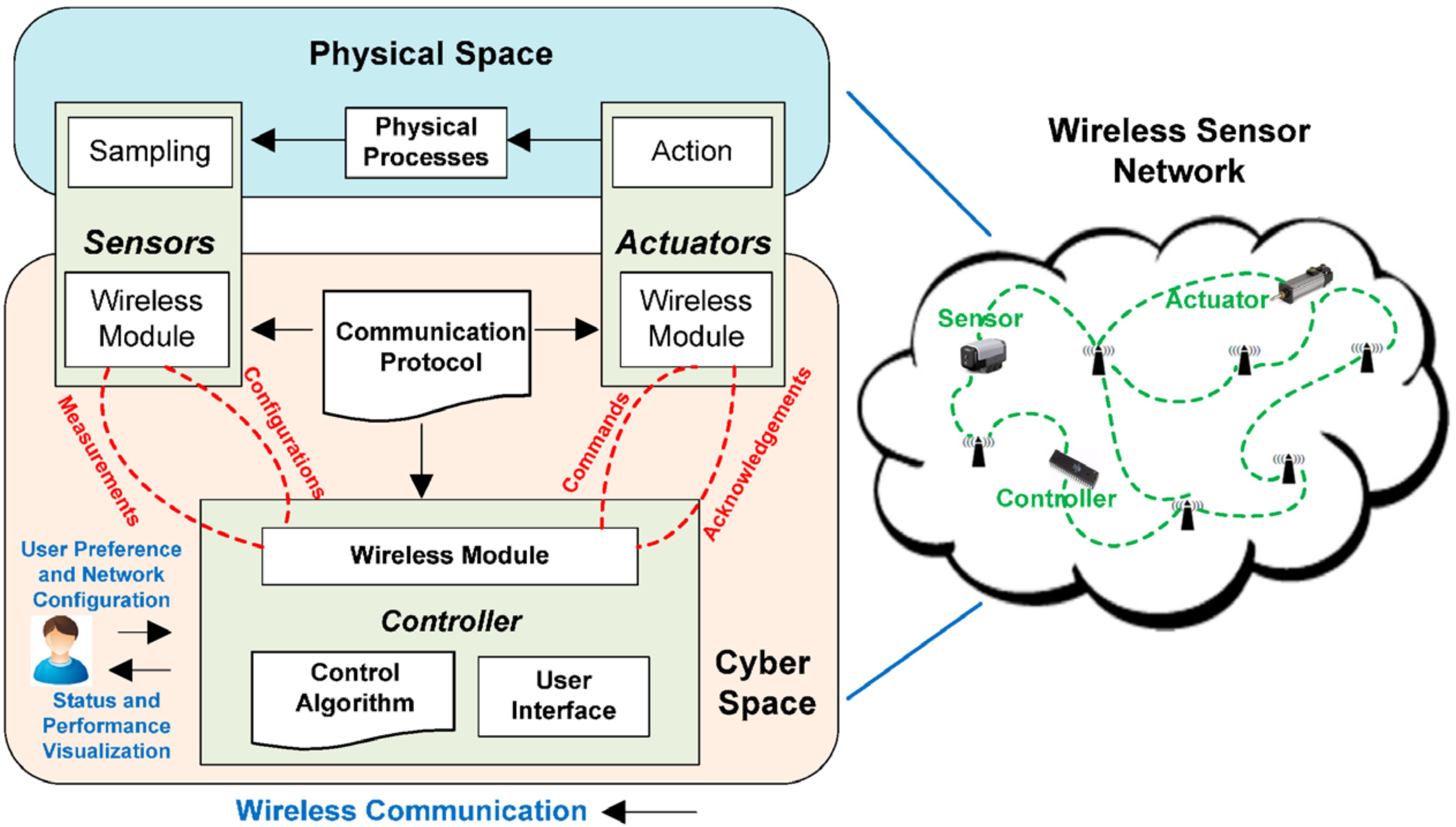
Background

- Cyber-Physical Control Systems (CPCS)
- Basic elements: Computation, Control and Communication
- i.e. large scale embedded systems with network connectivity



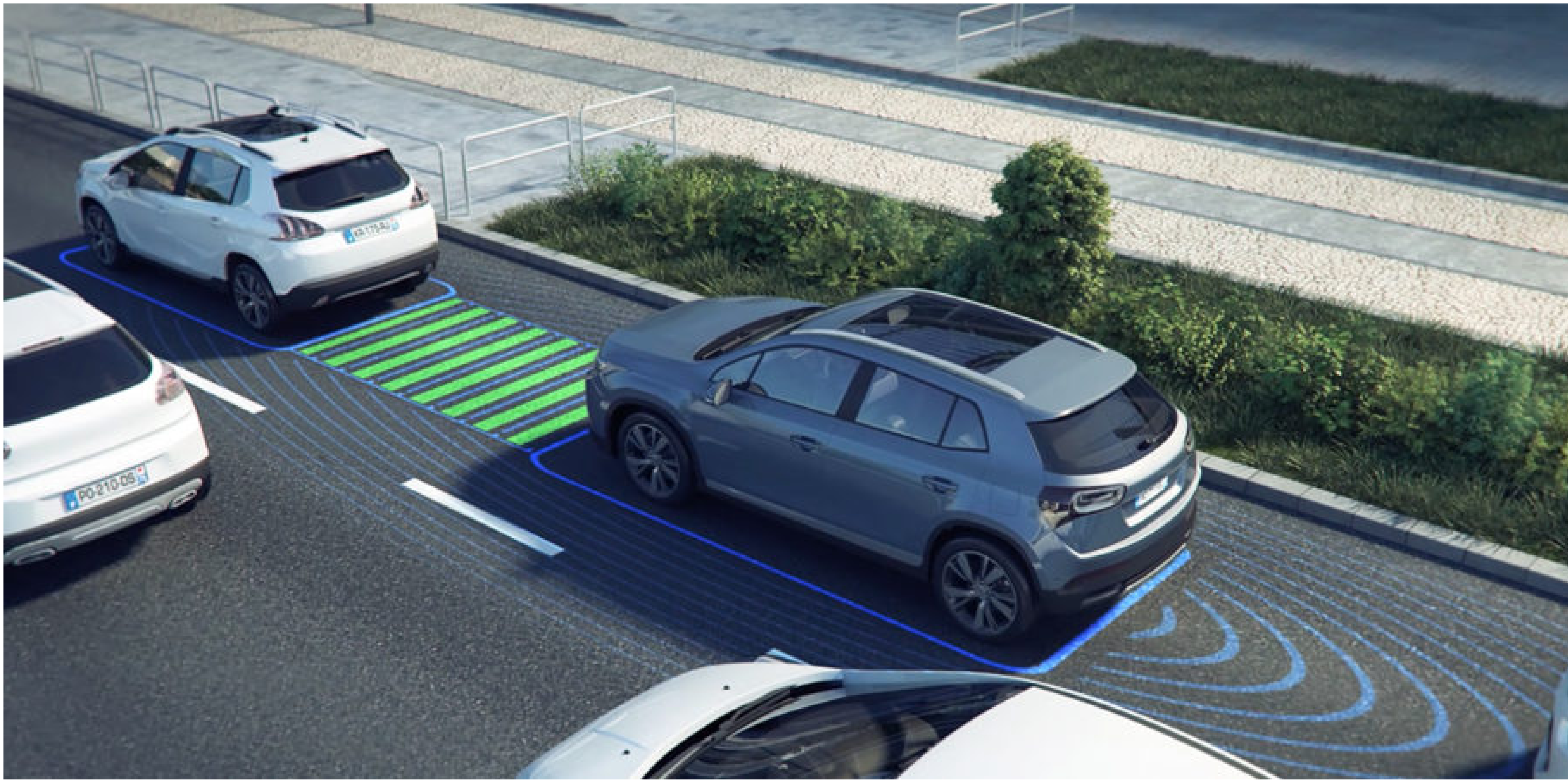
Background

- Example 1: Wireless Sensor Network (WSN)



Background

- Example 2: Autonomous driving



Background

Requirements of Cyber-Physical Control Systems (CPCS):

- System should be flexible
- High robustness and resilience
- Long-term deployment subject to ageing and degradation

Traditional scheduling systems cannot fully satisfied!

Thesis Contributions

Research questions:

1. How could a cyber-physical control system (CPCS) **be more robust to timing uncertainties,** especially **in a long term**?
2. How could **online information** and **cloud computing** be utilized to improve the **resilience** of a CPCS?

Thesis Contributions

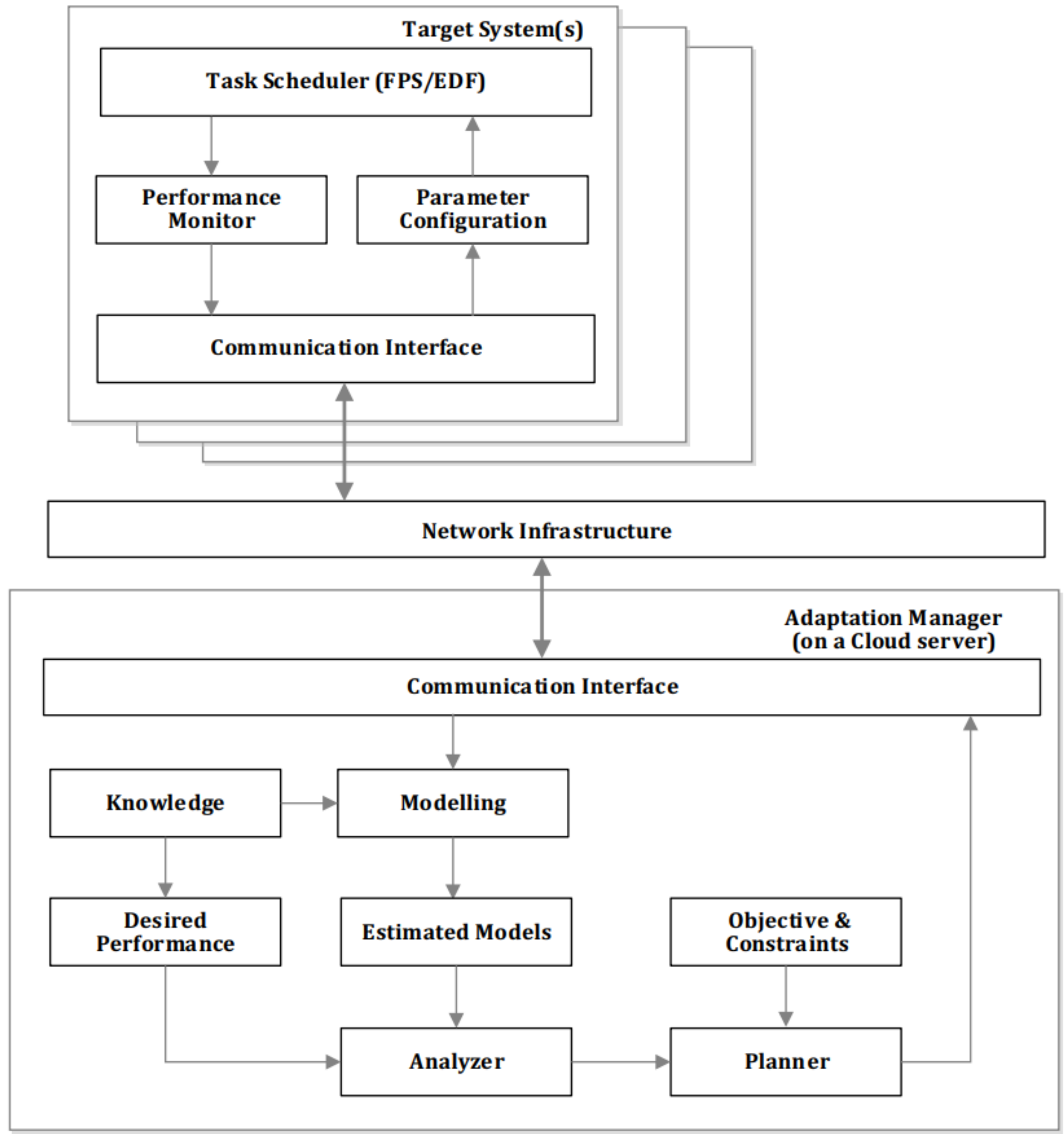
Adaptive

TAsk

Scheduling

(ATAS)

Framework



Thesis Contributions

In this thesis, we have made the following contributions:

1. **Trend Analysis of Worst-Case Execution Times**
2. **Period Adaptation**
3. **A Dual-Period Task Model**

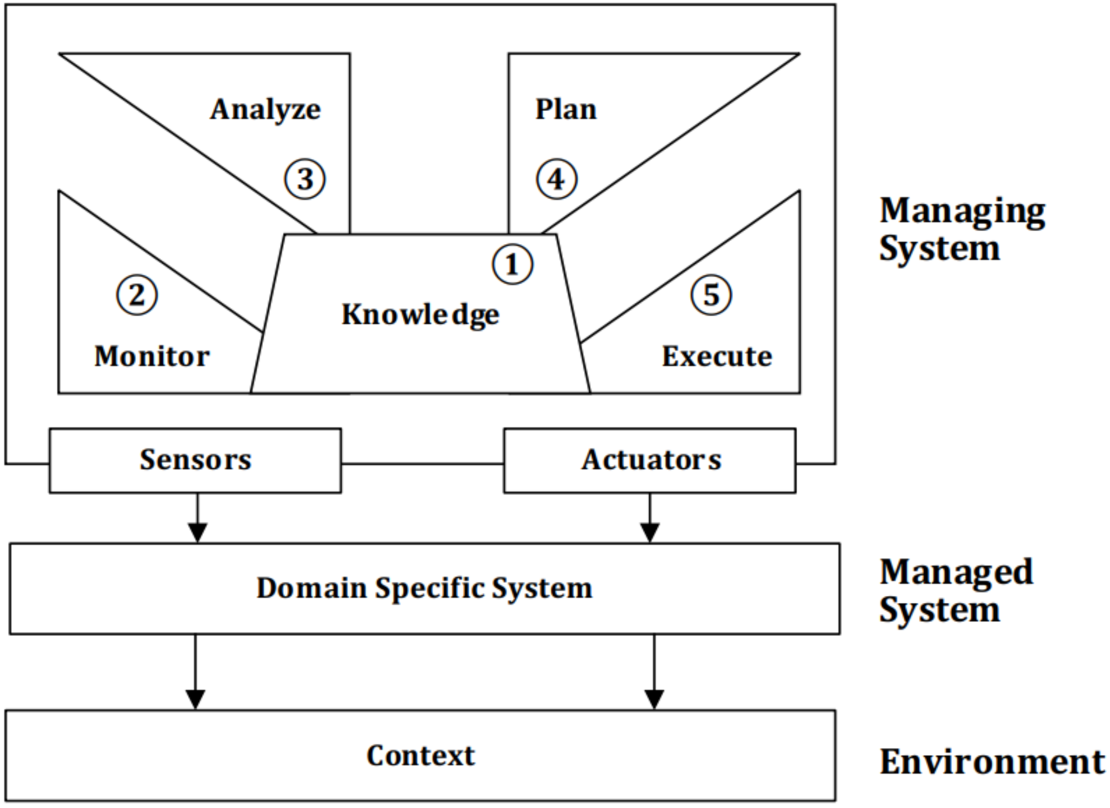
For this seminar, I will focus on the period adaptation.

Seminar Outline

1. Background
2. Control Task Timing
3. Period Adaptation
4. Evaluation
5. Conclusion

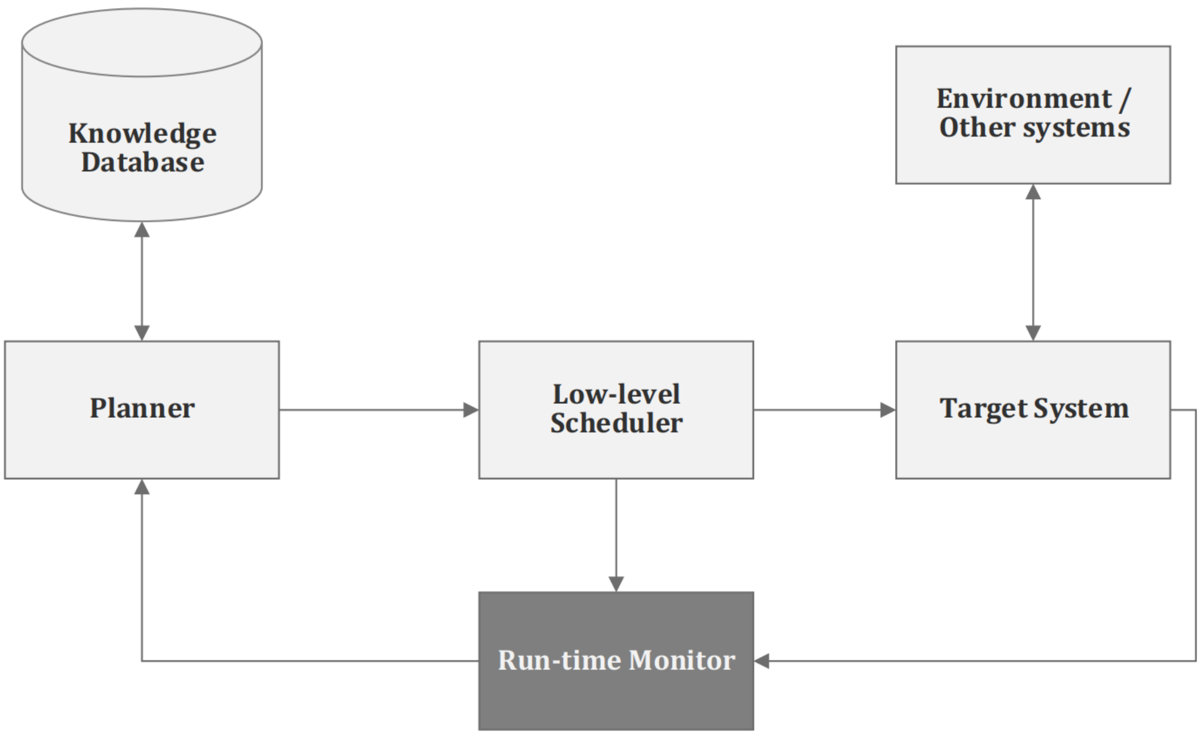
MAPE-K Framework

- An architecture introduced by IBM recently
- Support adaptiveness for computer systems
- Standard framework for self-adaptive systems



Period Adaptation: Structure

- MAPE-K in scheduling
- Focus on control applications



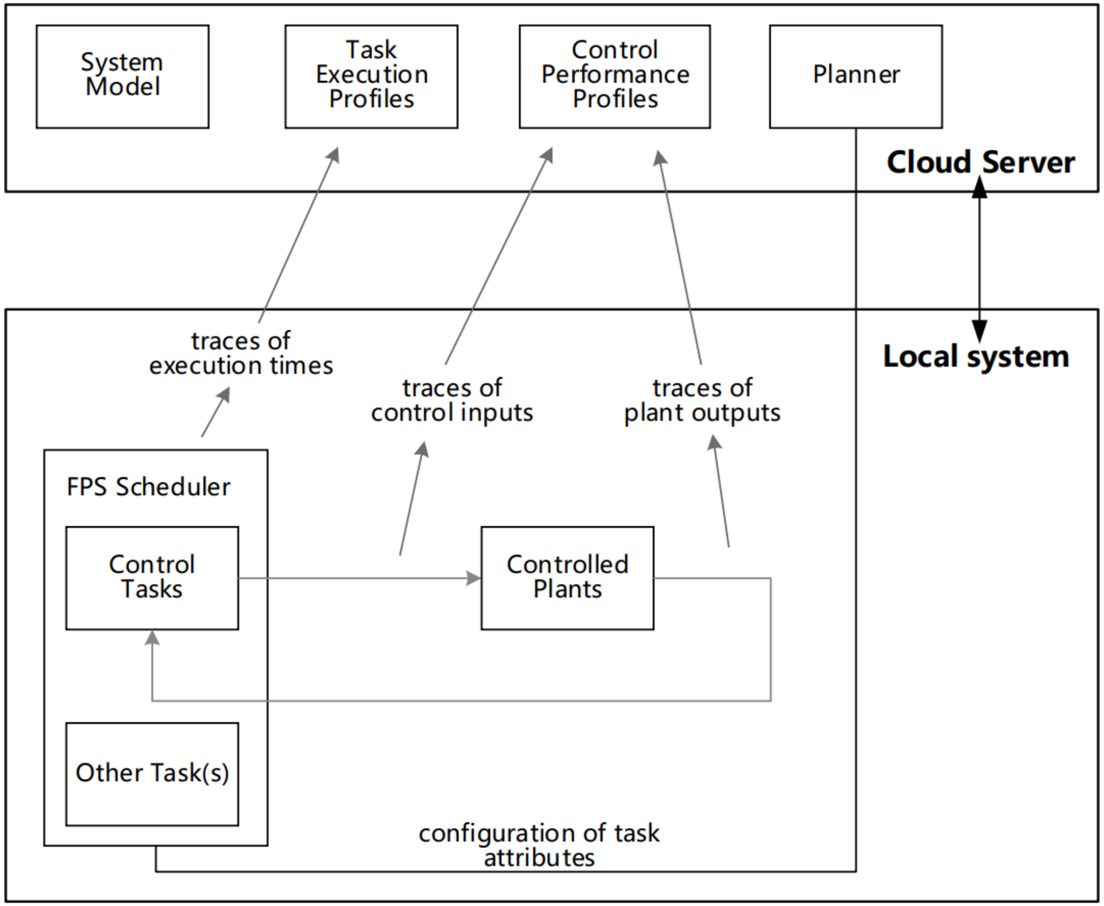
Period Adaptation: Structure

Two sub-systems:

- Local system
- Cloud Server

M, E on the local machine

A, P and K on the cloud



Assumptions

1. The embedded computer (local system) consists a uniprocessor and a preemptive scheduler using fixed-priority scheduling (FPS).
2. Control tasks are released periodically, i.e., the system applies time-triggered rather than event-triggered controllers.
3. All tasks in the complete system are initially schedulable, given the control tasks are using periods that can satisfy control specifications.
4. The system has the ability to monitor task execution times (supported by POSIX-RT). The kernel has the ability to change task periods at run-time.
5. The system itself has limited resources but has connectivity to a more powerful cloud computer.

Period Adaptation

Observation:

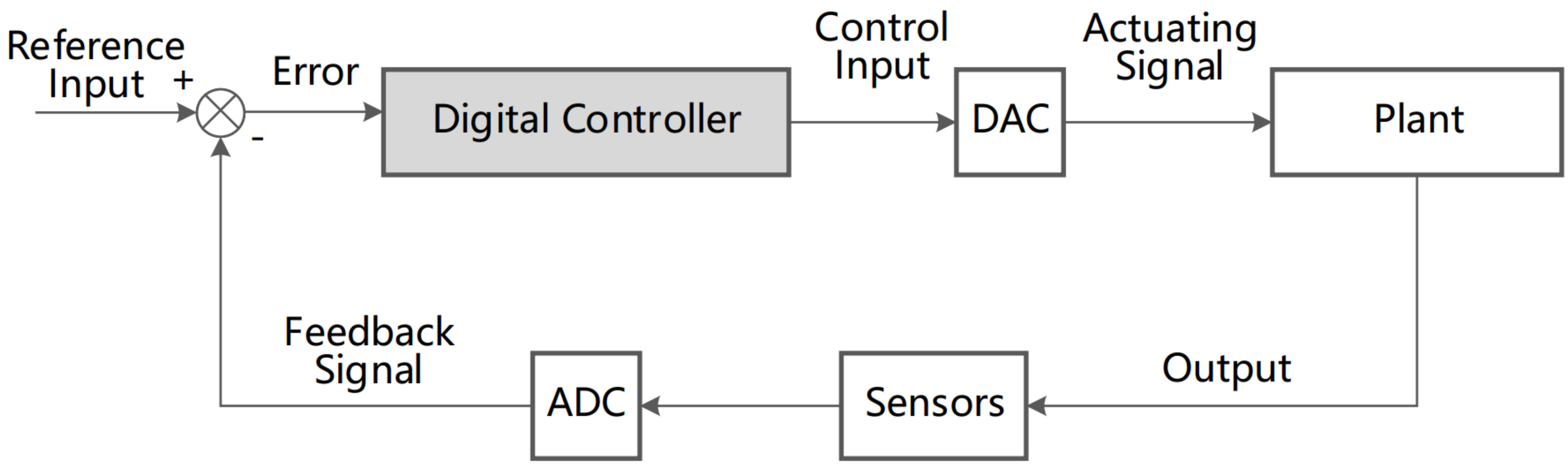
- Task period of a digital controller often have to be conservatively selected.

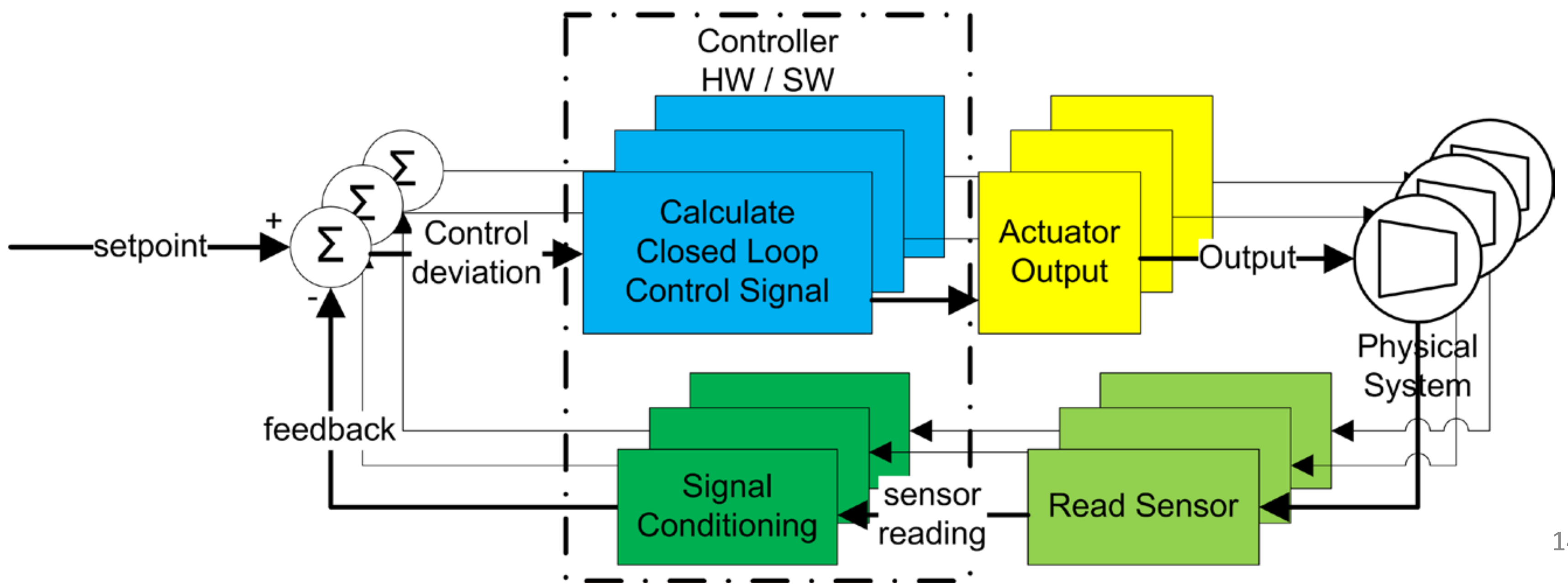
Key Ideas:

- Partition the taskset into control tasks Γc and non-control tasks Γnc
- Evaluate the new period based on observations and predictions
- Reduce the period of control tasks to save CPU time
- The period is gradually changed so minimal disruption will be created

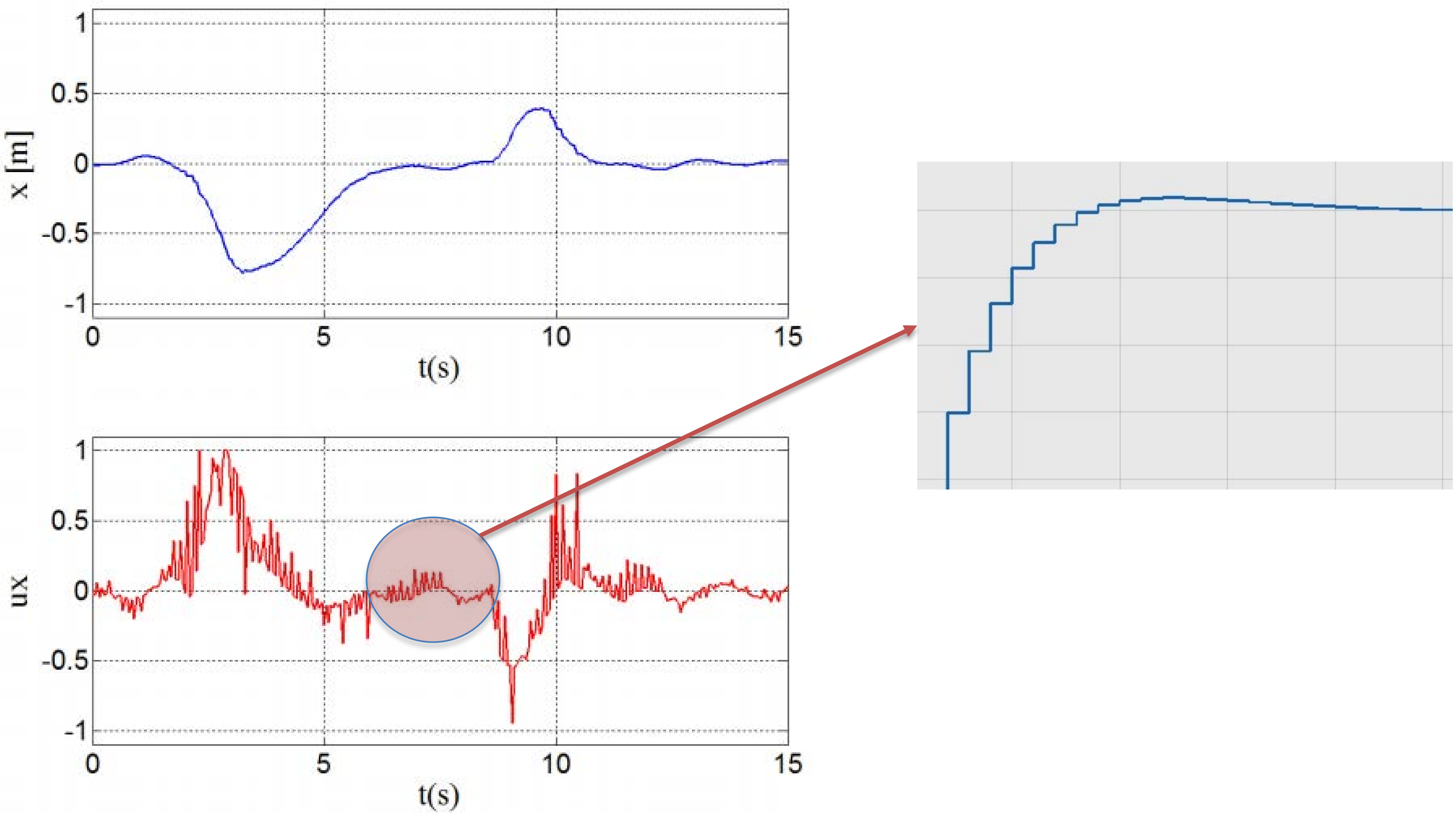
However, change control task period will affect control performance.

Digital Control Systems



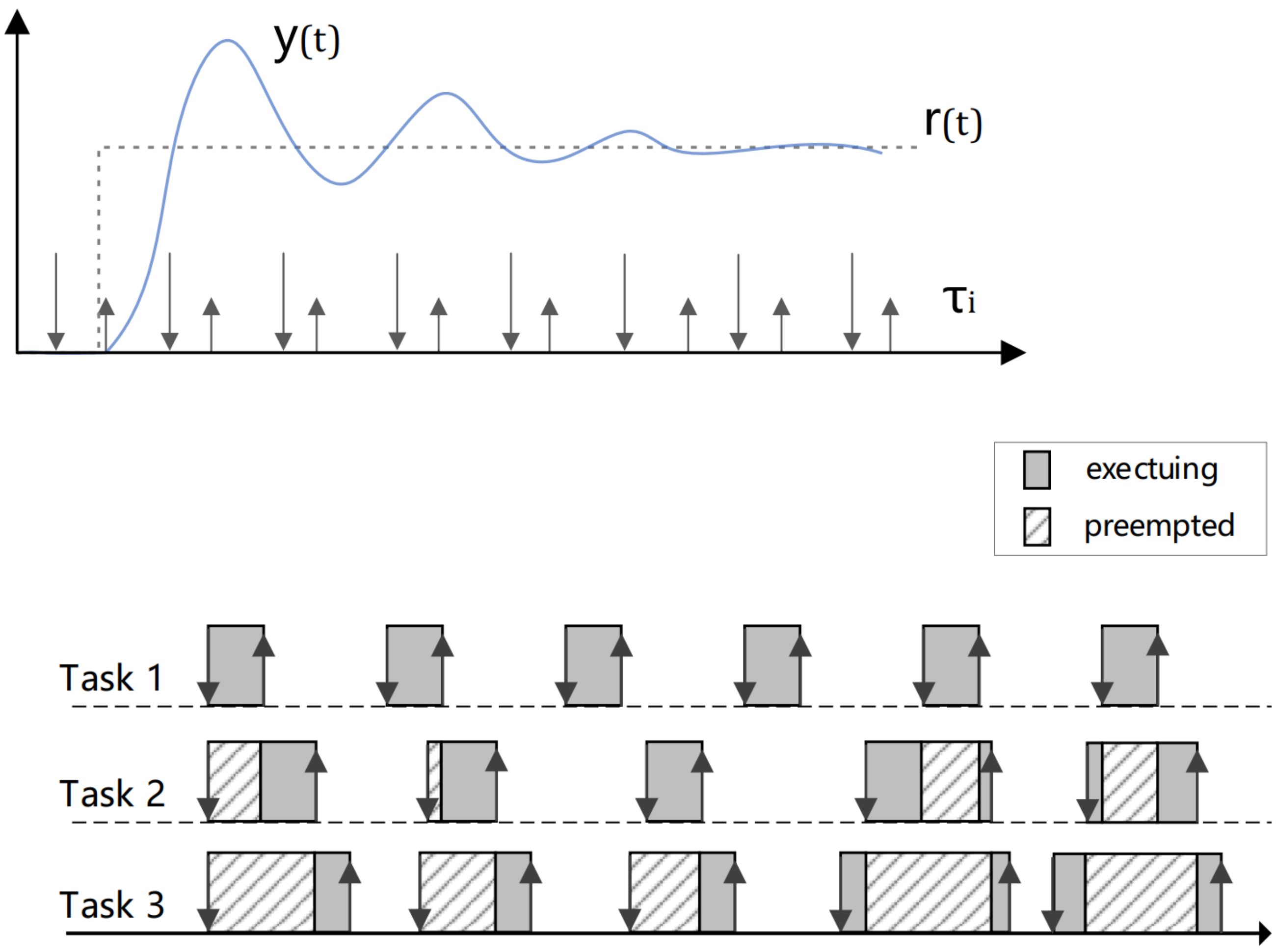


Digital Control Systems

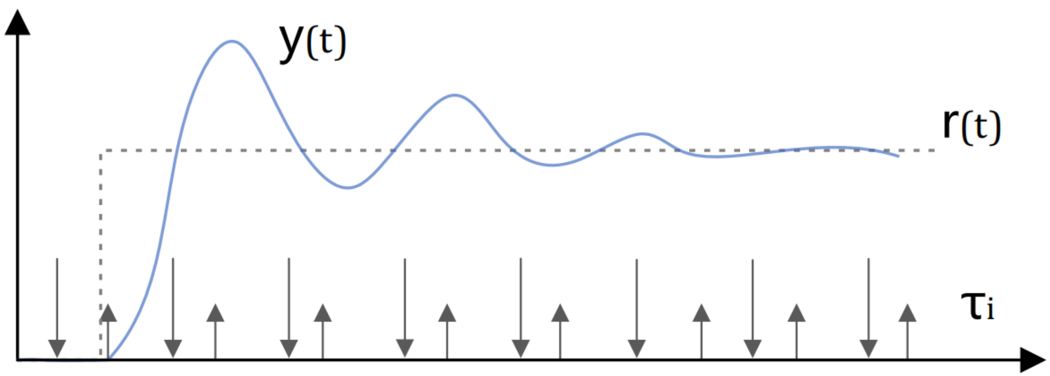


Each change is the result of the completion of a computation task

Control Performance and Timing



Control Performance and Timing

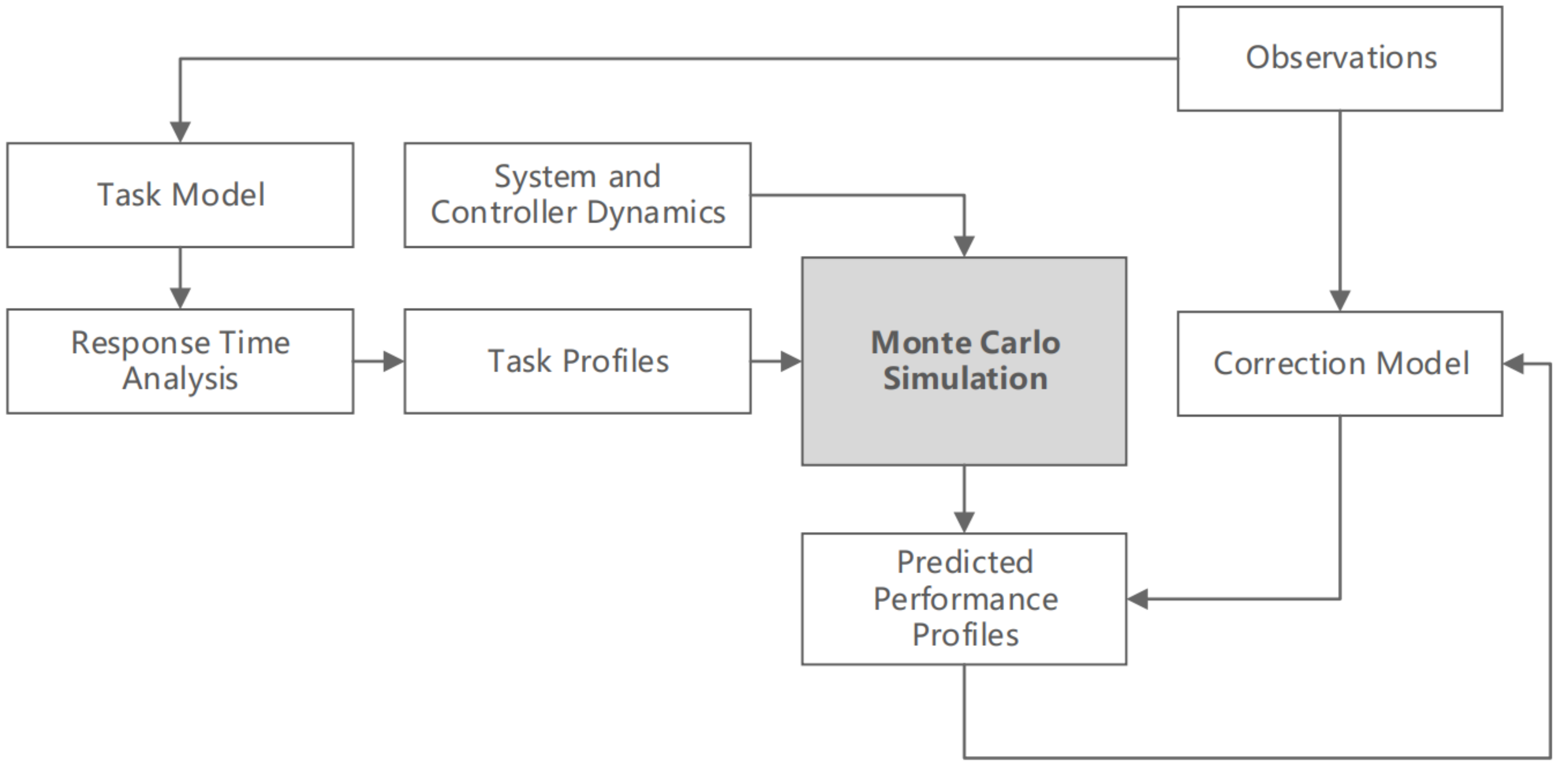


- Generally, larger period produce worse control
- Evaluating control performance: Integral of Absolute Error (IAE)

$$J = \int_0^\infty |r(t) - y(t)| \cdot dt$$
$$= \int_0^\infty |e(t)| \cdot dt$$

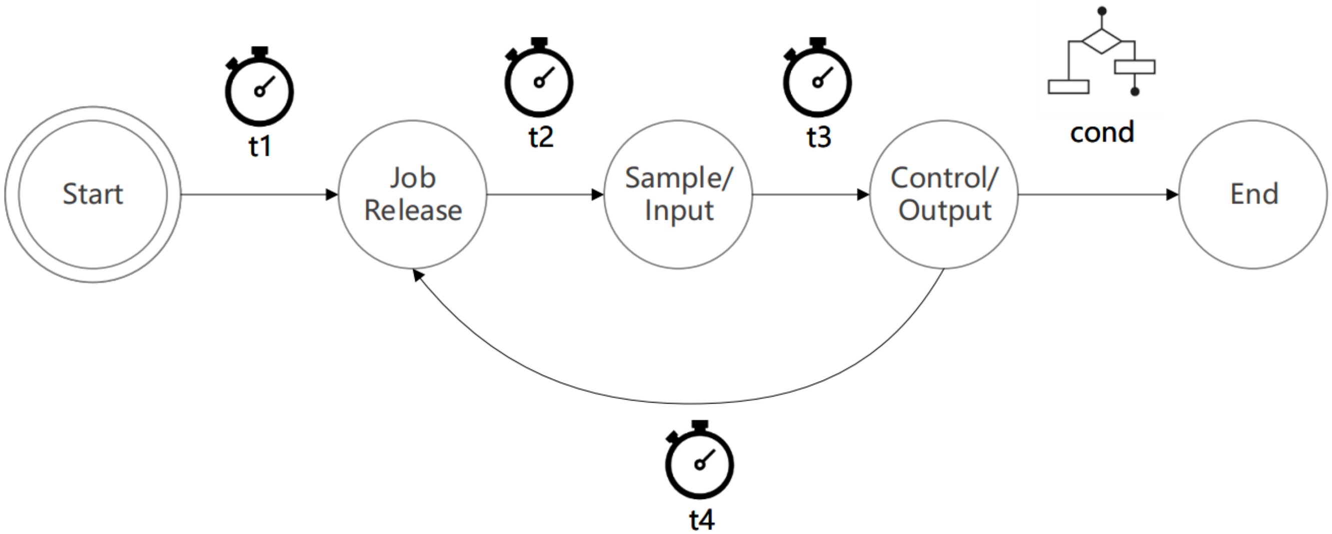
- Control performance will be hard to predict due to timing
- There will also be variations in IAE measurements!

Period Adaptation



- Monte Carlo: prediction based on a large number of experiments.
- a hybrid system = continuous + discrete

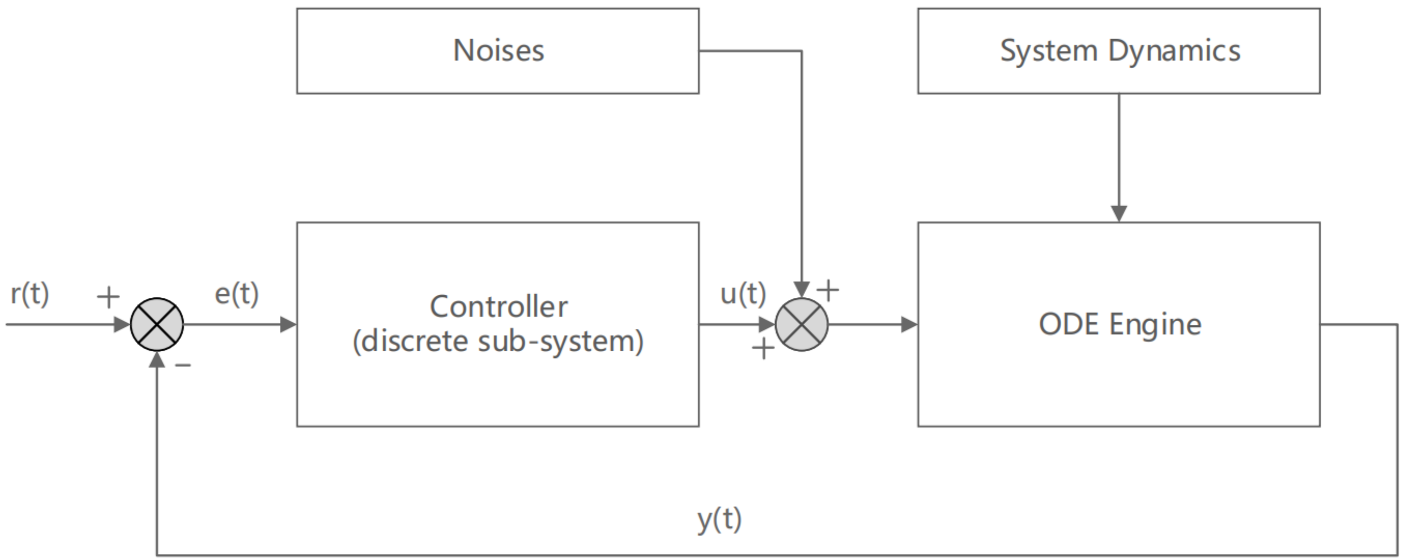
Monte Carlo Simulation – Discrete Model



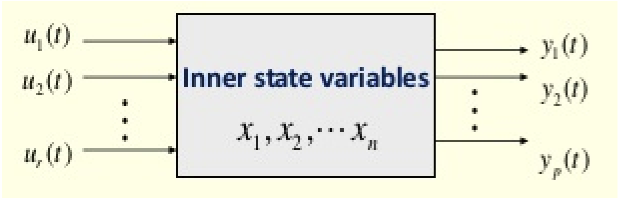
Execution timing model:

- t_1: initial task phasing
- t_2: sampling delay
- t_3: input-output latency
- t_4: task reset time

Monte Carlo Simulation – Continuous Model



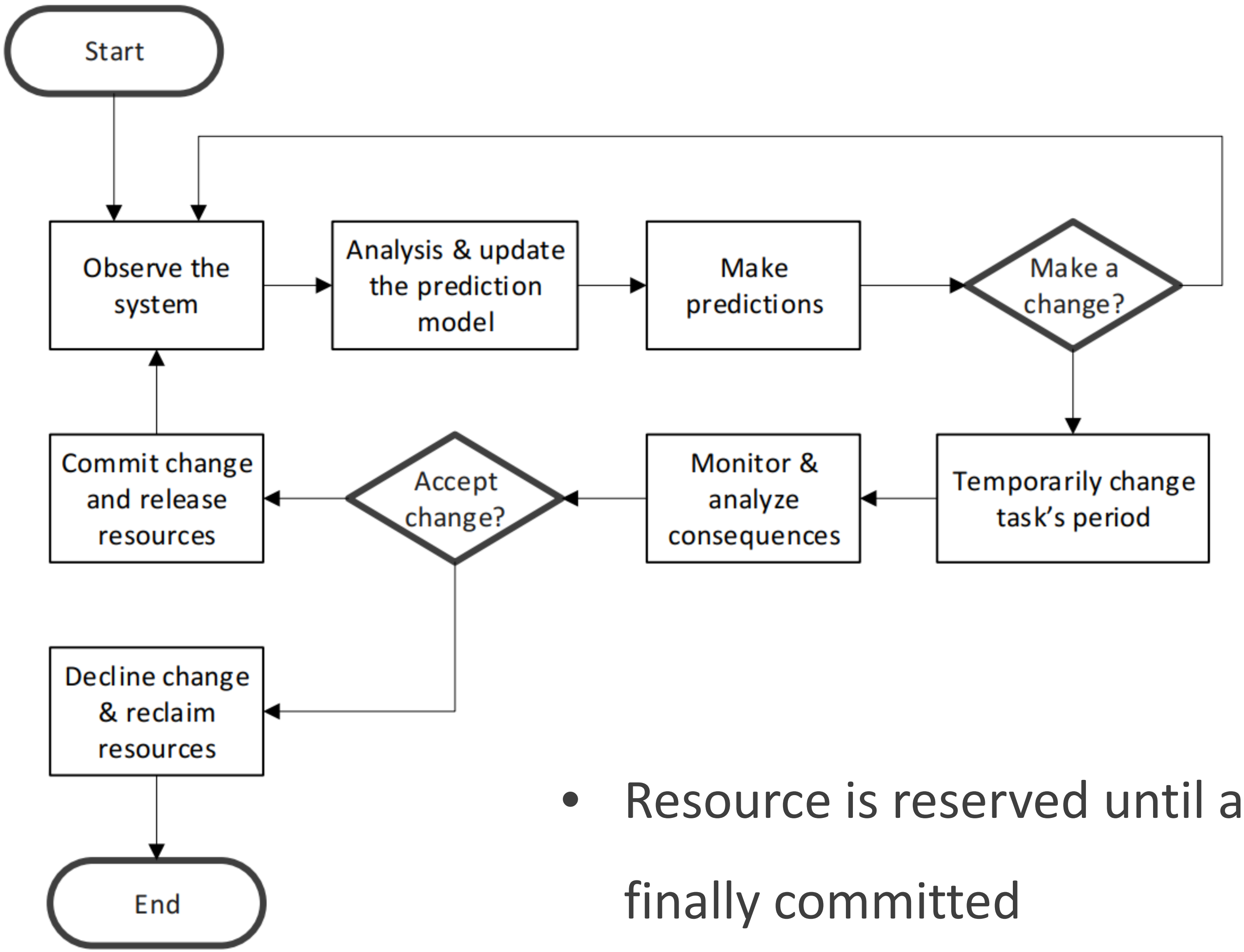
- Controllers: PID, MPC, LQR
- System dynamics equation: state-space



$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -5 & -26 & -5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Period Adaptation - Flowchart



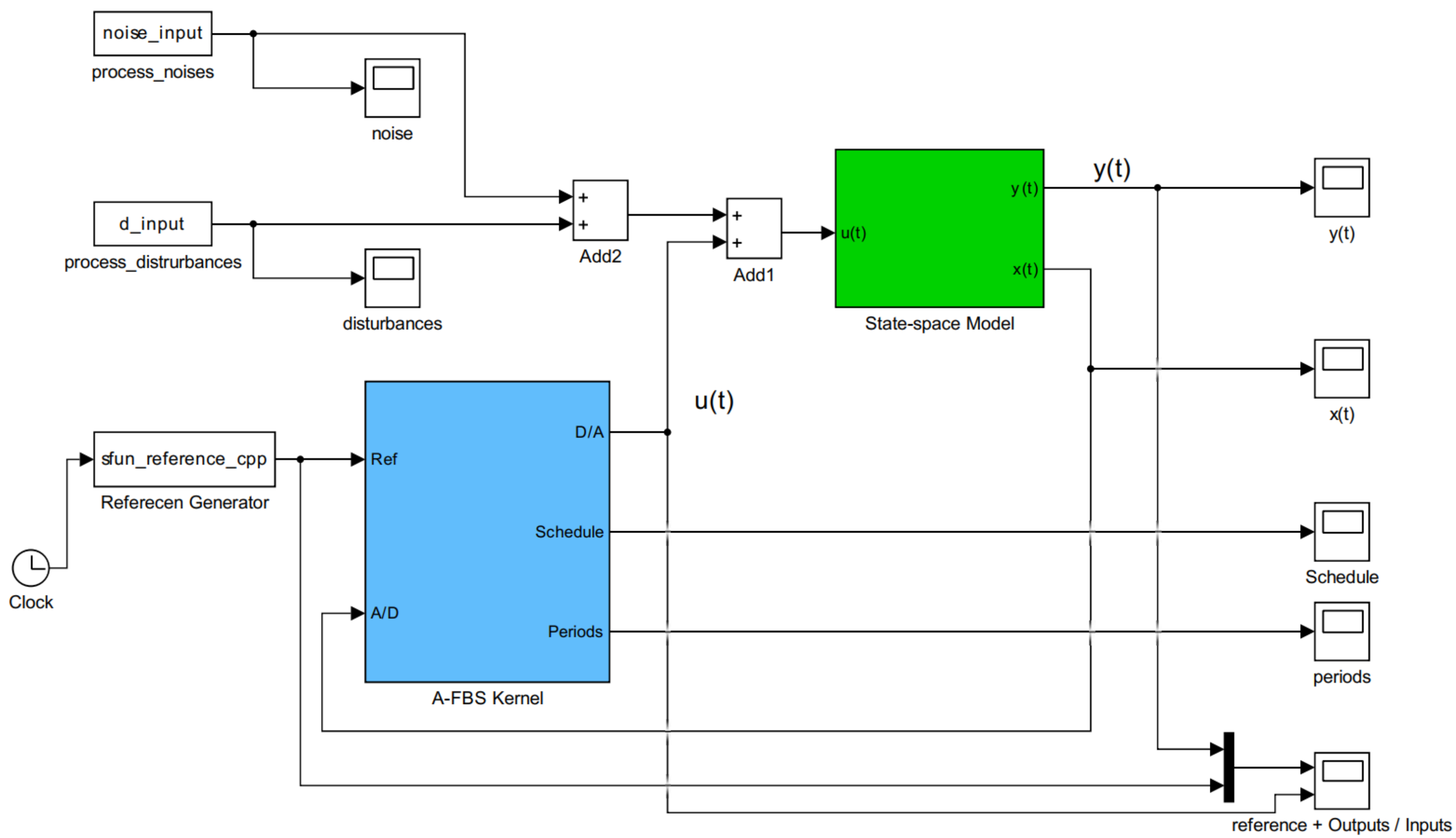
- Resource is reserved until a change is finally committed

Evaluation

1. Experiment Setup
2. Demonstration
3. Utilization v.s. Control Quality
4. Robustness
5. Multiple tasks
6. Overhead analysis

Evaluation – Environment

- MATLAB / Simulink: dynamic systems simulation
- Task scheduler: s-function



Evaluation – Environment

- System dynamics:

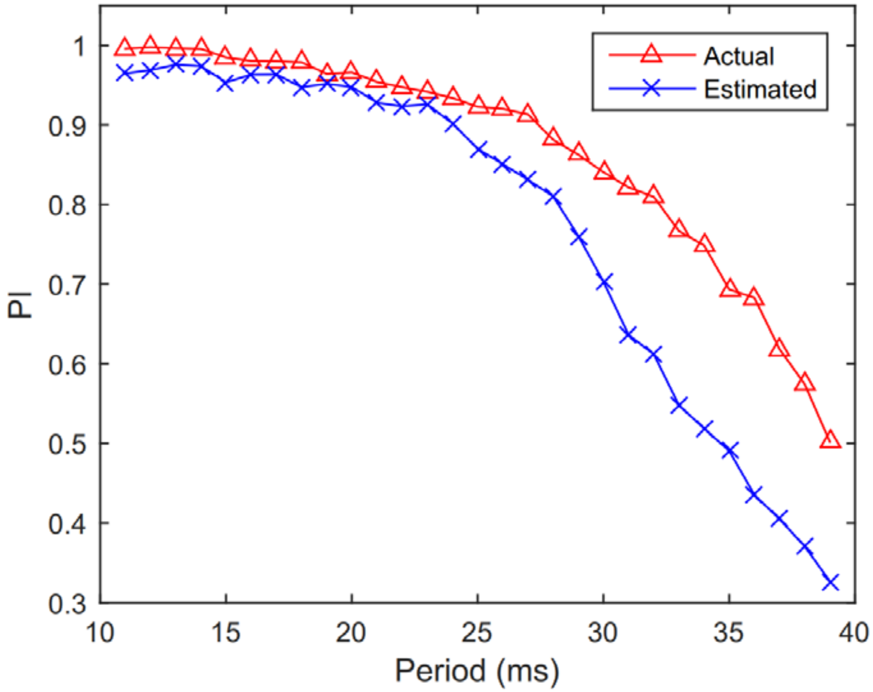
$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 10 & 25 \\ -25 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1.6 \end{bmatrix} u$$

$$y = \begin{bmatrix} 2.5 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- Taskset:

Task	C_i (ms)	T_i (ms)	Control Task
τ_0	0.42	1.57	
τ_1	0.10	2.15	
τ_2	0.53	4.99	
τ_3	0.87	7.77	
τ_4	0.48	8.01	
τ_5	1.00	10.00	(x)

Evaluation



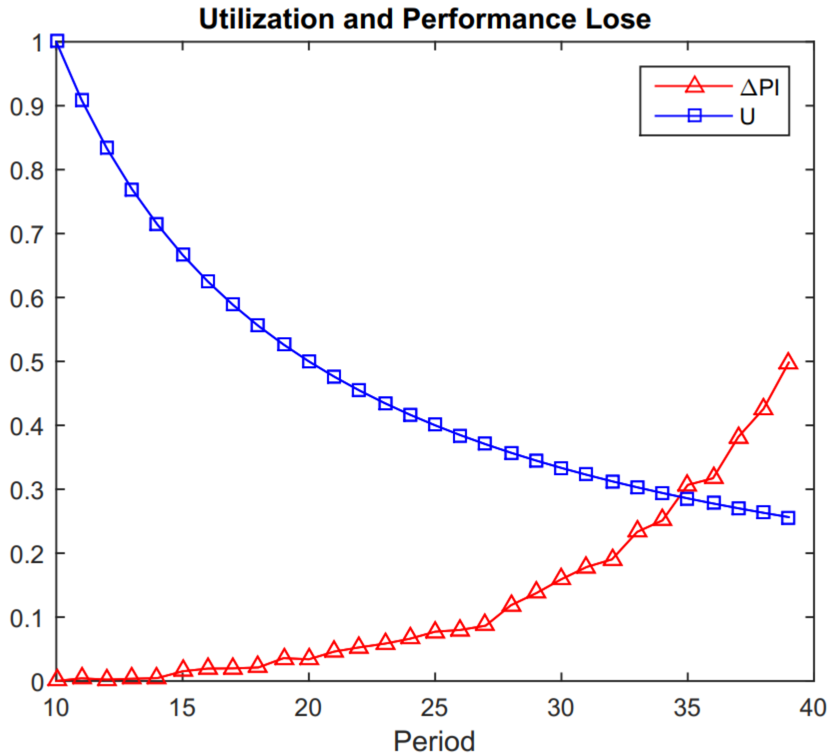
1. Minimal PI = 0.3, step size = 1 ms
2. In each step:

   5,000 samples + 3,000 predictions
3. Prediction is one step ahead
4. Terminated at T_i = 39 ms
5. Prediction error increases but is

   conservative

Evaluation

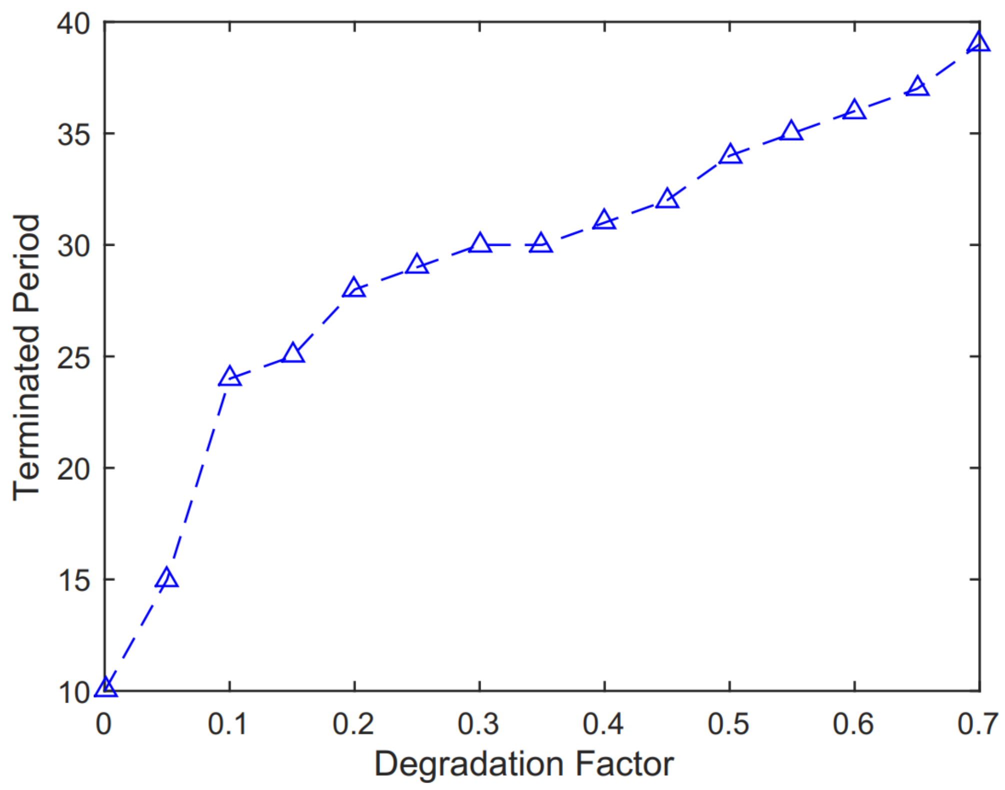
- ΔPI and Utilization



- Trade-off between control performance and CPU usage

Evaluation

- If varying the degradation factor (ΔPI):



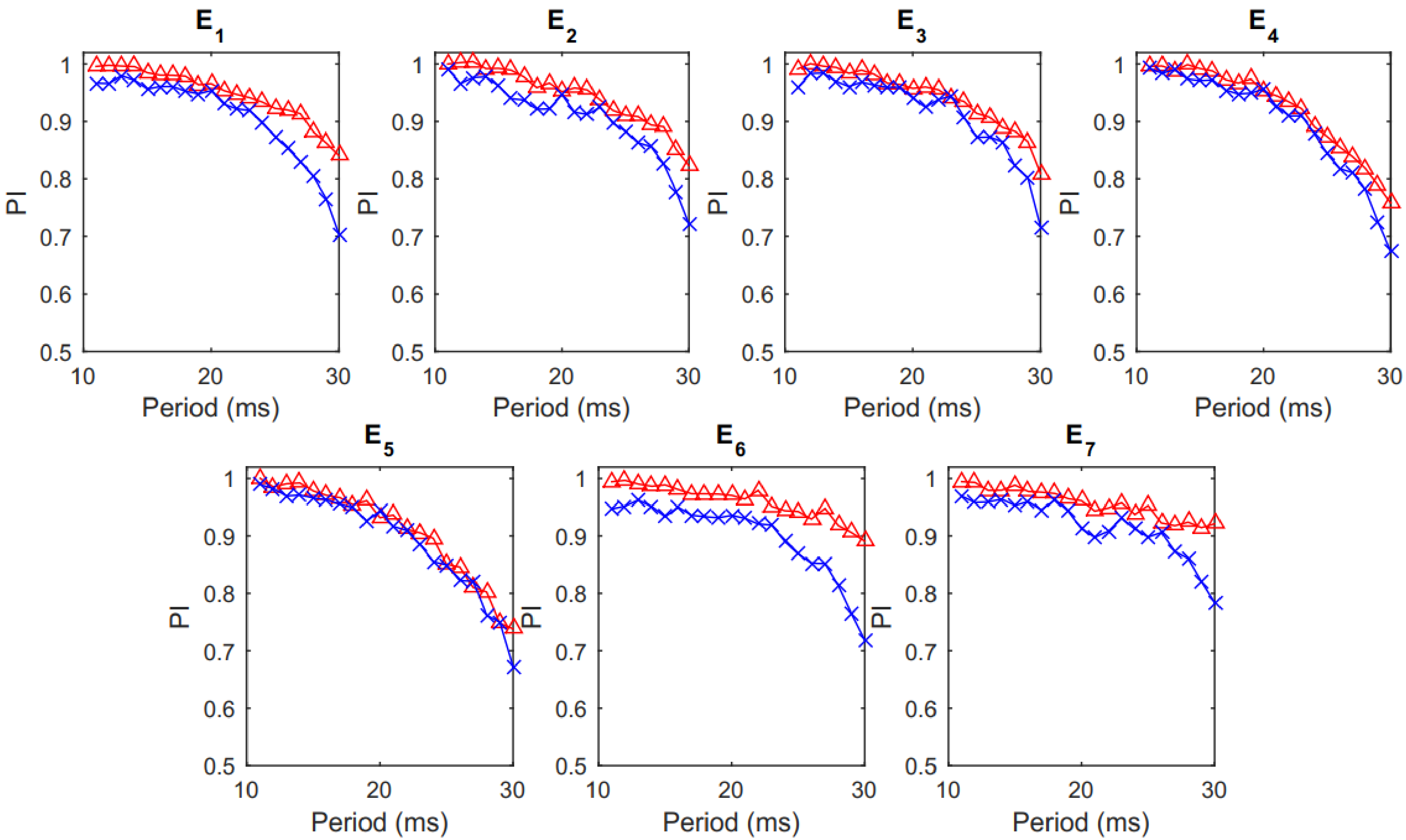
- In this example, 0.2 is a good trade-off point.

Evaluation - Robustness

- Experiment sets:

Experiment	System Dynamics	Task Model
E_1	Ideal	Normal
E_2	-5%	
E_3	+5%	
E_4	Ideal	WCET
E_5	+5%	
E_6	Ideal	BCET
E_7	-5%	

Evaluation - Robustness



Multiple Control Tasks

- **Highest Priority First (HPF)** The highest priority task has the largest margin and the highest interferes with non-control tasks, and also changing higher priority task first can avoid the need for recalculating periods of lower priority task.

- **Least Sensitivity First (LSF)** The task is selected according to its sensitivity of performance index by changing its period. This is evaluated by a sensitivity function:

$$\Delta J = \frac{\partial J}{\partial h} \Delta h \Big|_{h=T_i(k-1)}$$

- **Least Uncertainty First (LUF)** Select the controller that behaves closest to its prediction model, i.e., minimal prediction bias and error.

Discussion on Overheads

- **Computation**

- calculating performance statistics and run-time monitoring: only addition operations
- run as a low priority background service to create minimum interferences to other running tasks in the system.

- **Communication**

- only packets contain statistical data are transferred to the cloud server
- the communication bandwidth and reliable requirement is low.

- **Memory**

- buffer traces and statistical data
- a hundred bytes to a few kilobytes depending on the sampling rate

Conclusion

In this work, we have made the following contributions:

1. Adaptive task scheduling for CPCS
2. Monte Carlo to solve control scheduling co-design
3. Utilizing statistical information for performance prediction and error estimation
4. Proposed an adaptation method and evaluated its performance.

Future Work

1. Fixed adaptation step -> variable step
2. Uniprocessor -> multicores
3. Task dependencies: control <-> control, control <-> non-control

Thank You!