Predictive Sensing and Adaptive Management For Real-Time Applications

Koji Inoue

Kyushu University

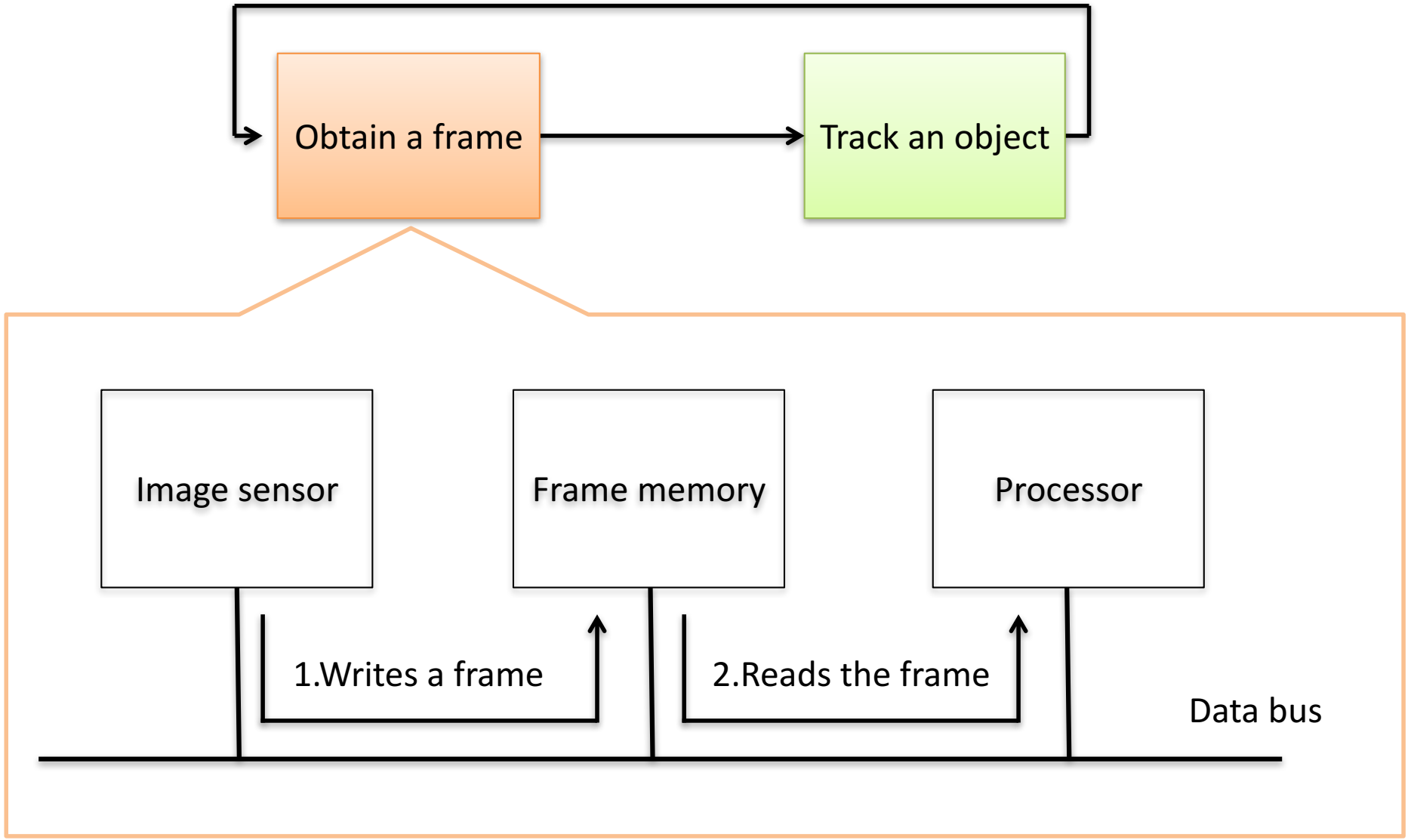
Outline of this talk

Let's predict future and
improve system management efficiency!

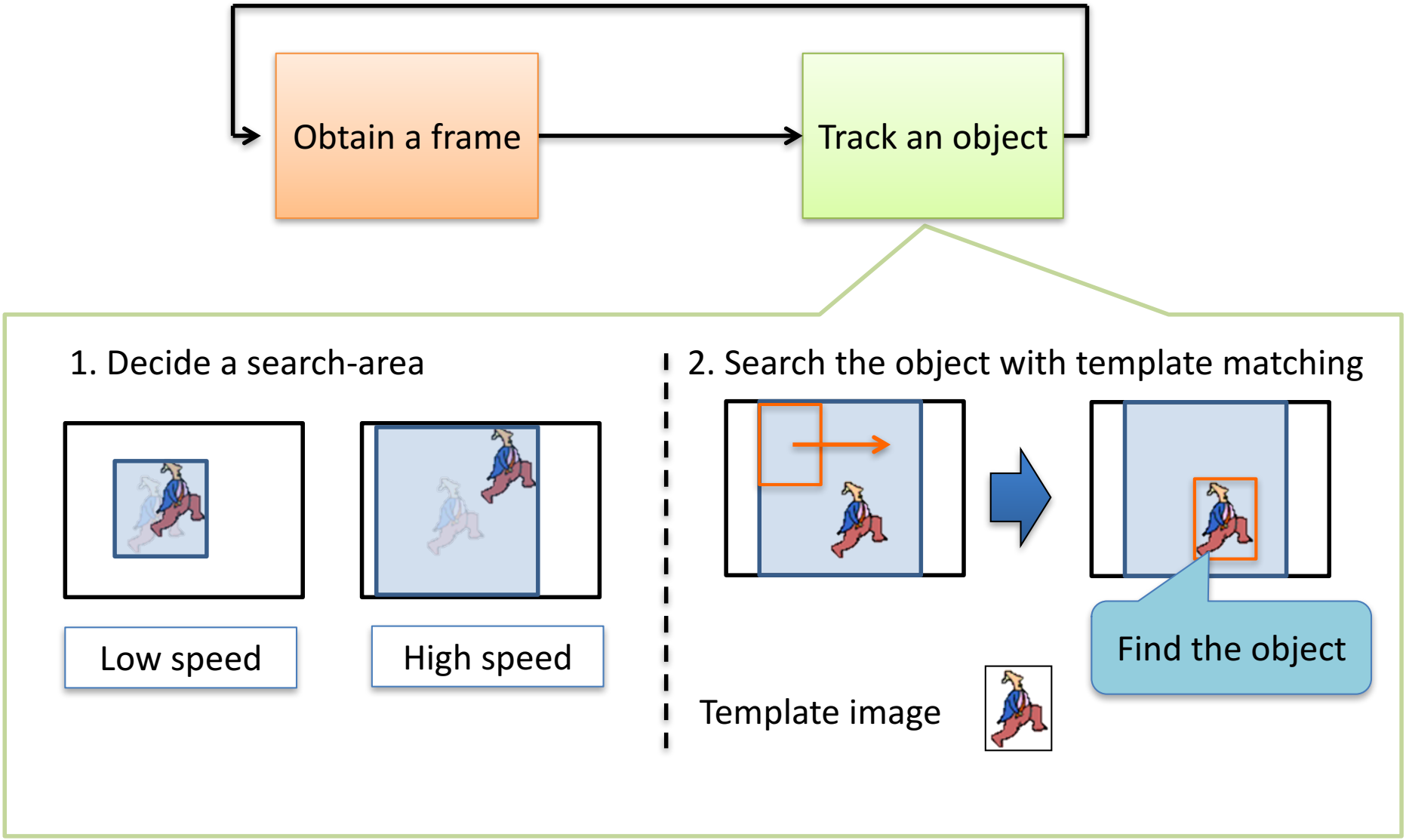
1: Improve energy efficiency of on-line object tracking

2: Accelerate performance of control systems

Procedure of on-line object tracking

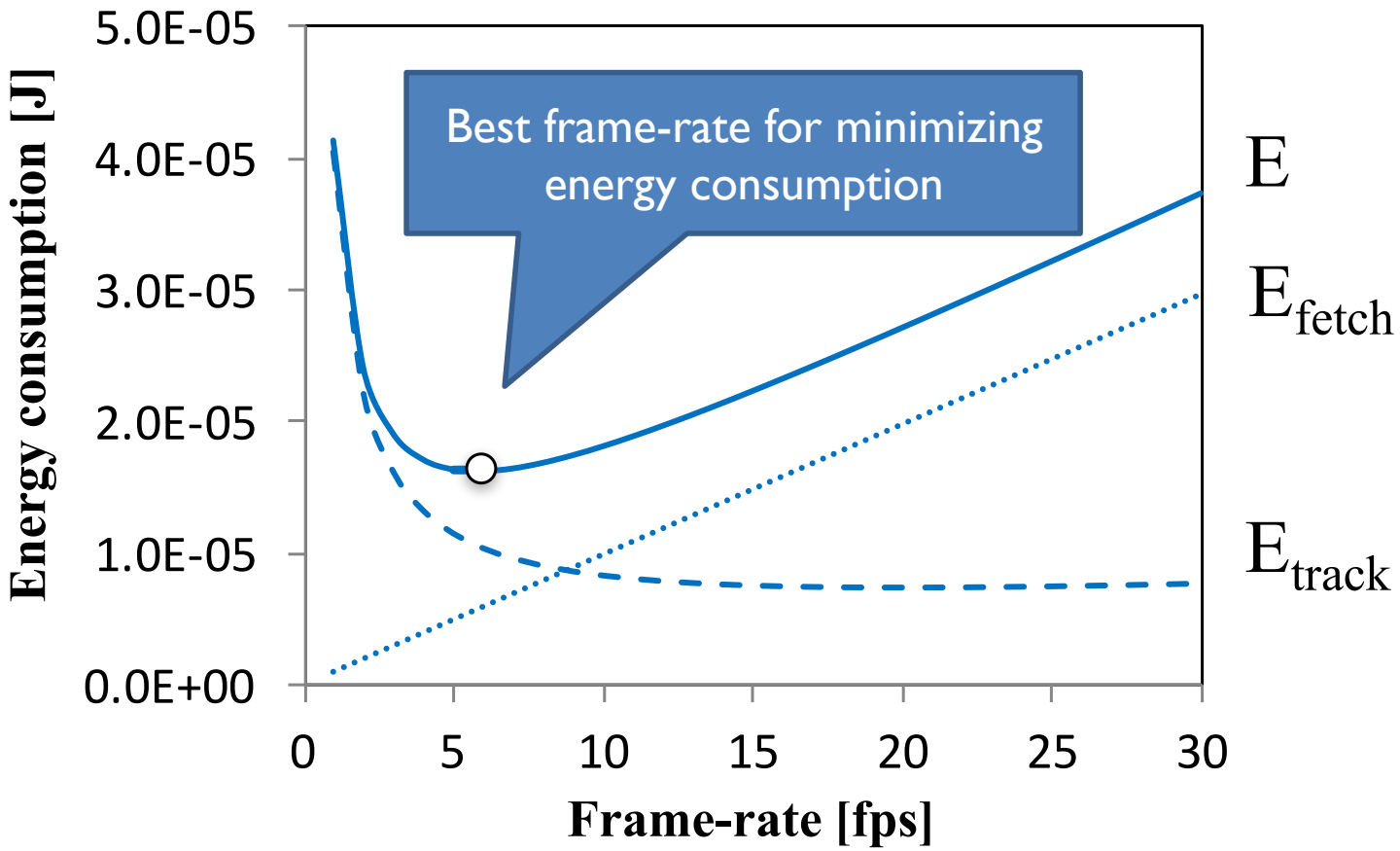


Procedure of on-line object tracking

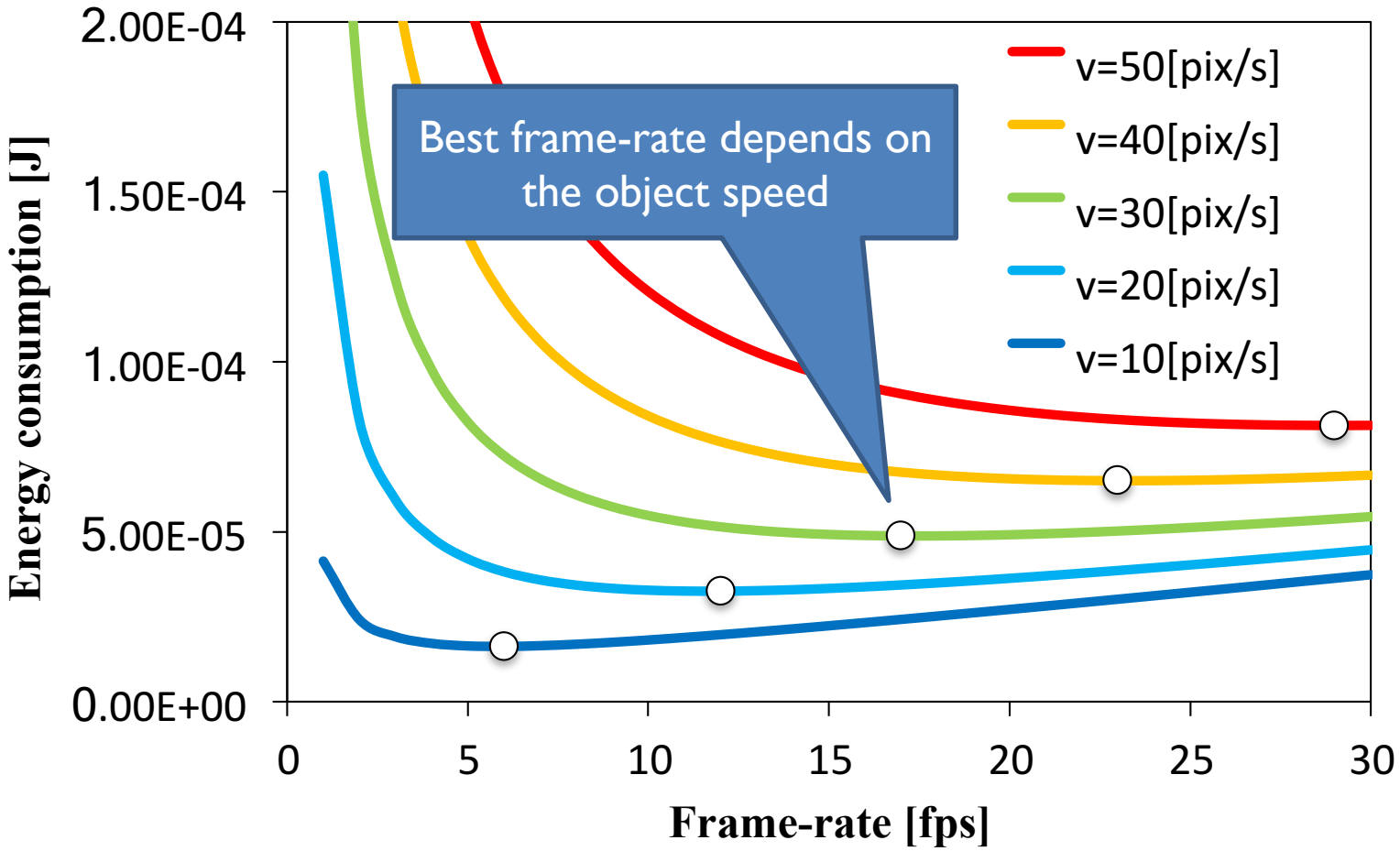


Motivation (1/2)

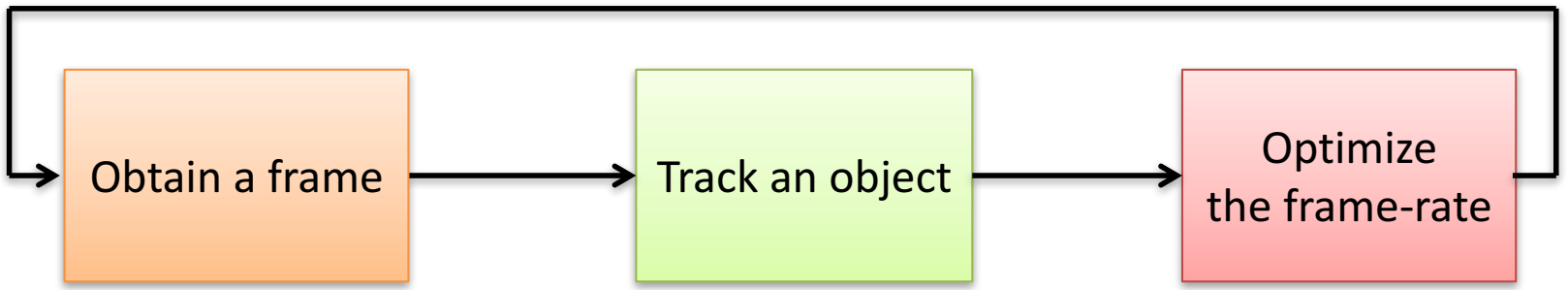
E_{fetch} : Energy required for obtaining frames
E_{track} : Energy required for object tracking
E : Total Energy (E_{fetch} + E_{track})



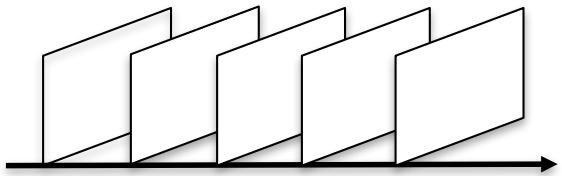
Motivation (2/2)



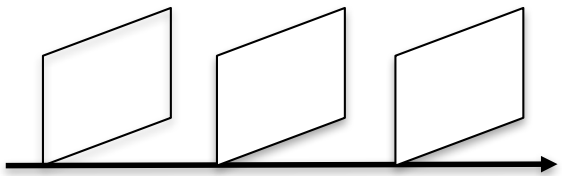
Adaptive frame-rate optimization



Decides frame-rate based on the object speed



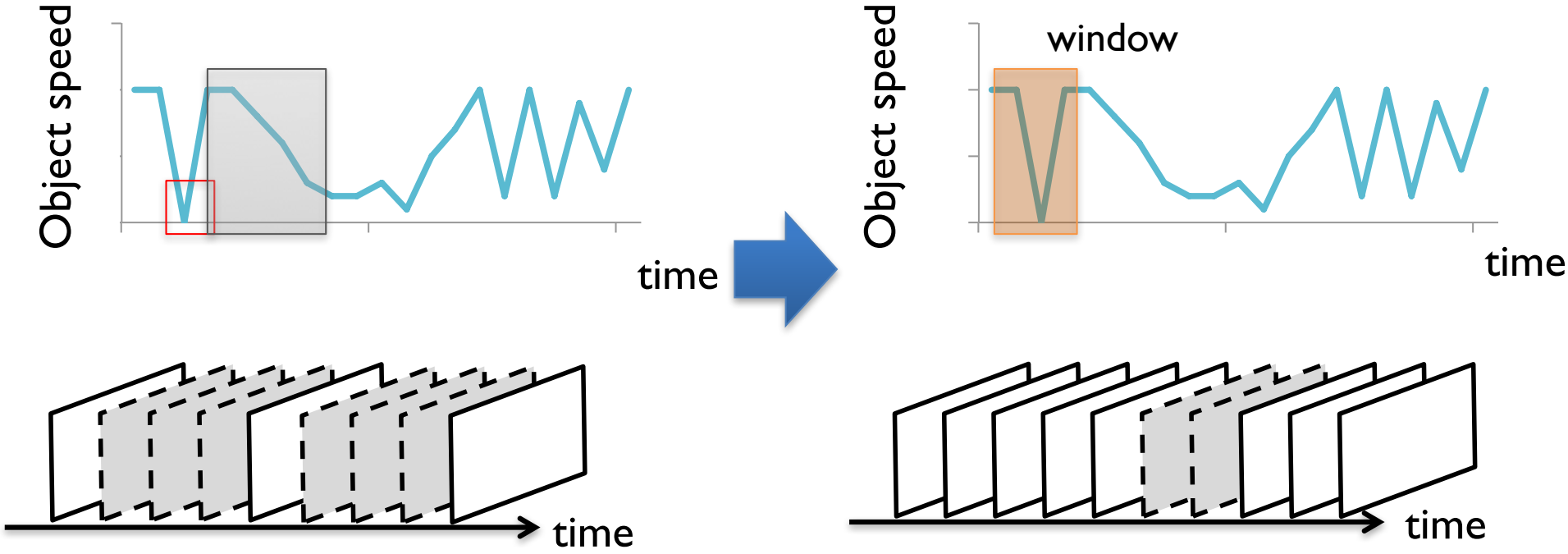
Object speed is high time



Object speed is low time

Accuracy oriented optimization

Proposed method with optimized frame-rate



Immediate frame-rate reduction worsens tracking accuracy.

Maintain the frame-rate IF the object speed variation in the window exceeds a threshold.

Experiments (1/2)

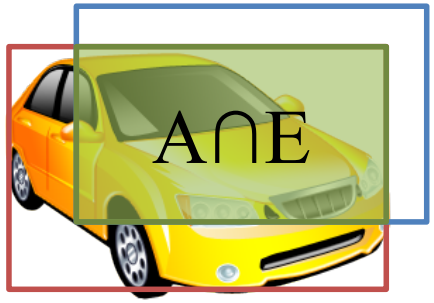
- **Methodology**
  - Simulator implemented by using the OpenCV library
- **Benchmark**
  - Videos : Seven input videos from Tracker Benchmark[1]
  - Features : Illumination variation, scale variation, etc.
  - Target frames : ~ 600 frames
- **Evaluation models**
  - FIX : Tracking with fixed frame-rate (30 fps)
  - ADAPT : Adaptive frame-rate optimization

[1] Yi Wu, Jongwoo Lim, Ming-Hsuan Yang, "Online Object Tracking: A Benchmark," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013

Experiments (2/2)

- **Metrics**
  - Tracking accuracy

    Overlap ratio between two rectangles



Estimated rectangle

Answer rectangle

$$\text{Accuracy} = \frac{area(A) \cap area(E)}{area(A) \cup area(E)} \times 100\ [\%]$$

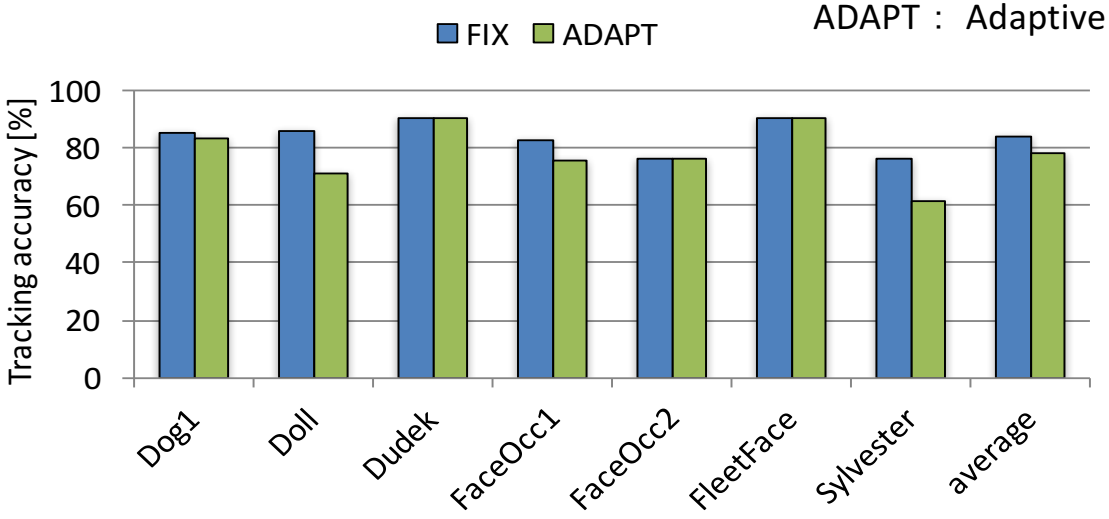
  - Energy consumption

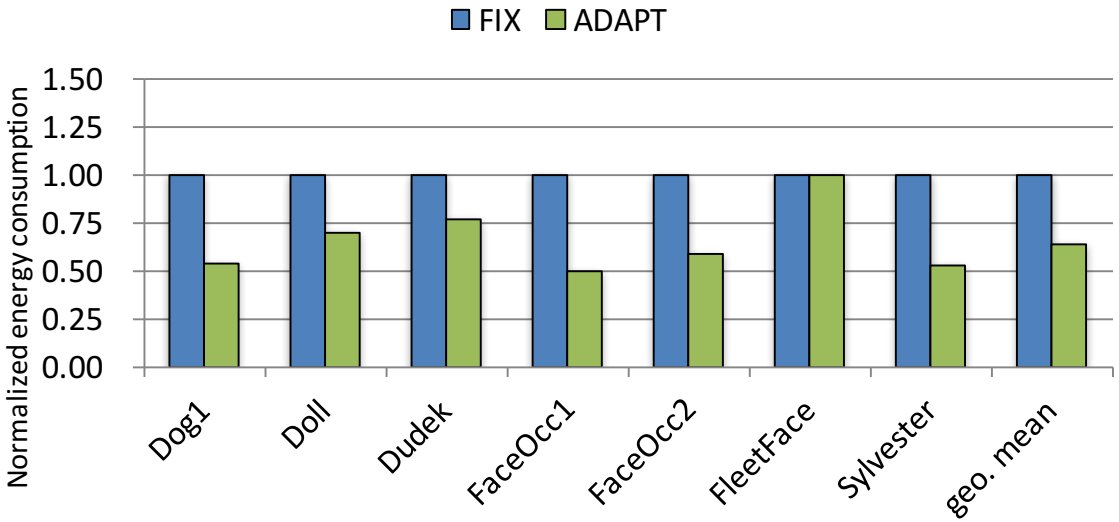
    Energy model for object tracking system

    $E = E_{fetch} + E_{track}$

Tracking accuracy

FIX : Tracking with fixed frame-rate (30 fps)
ADAPT : Adaptive frame-rate optimization





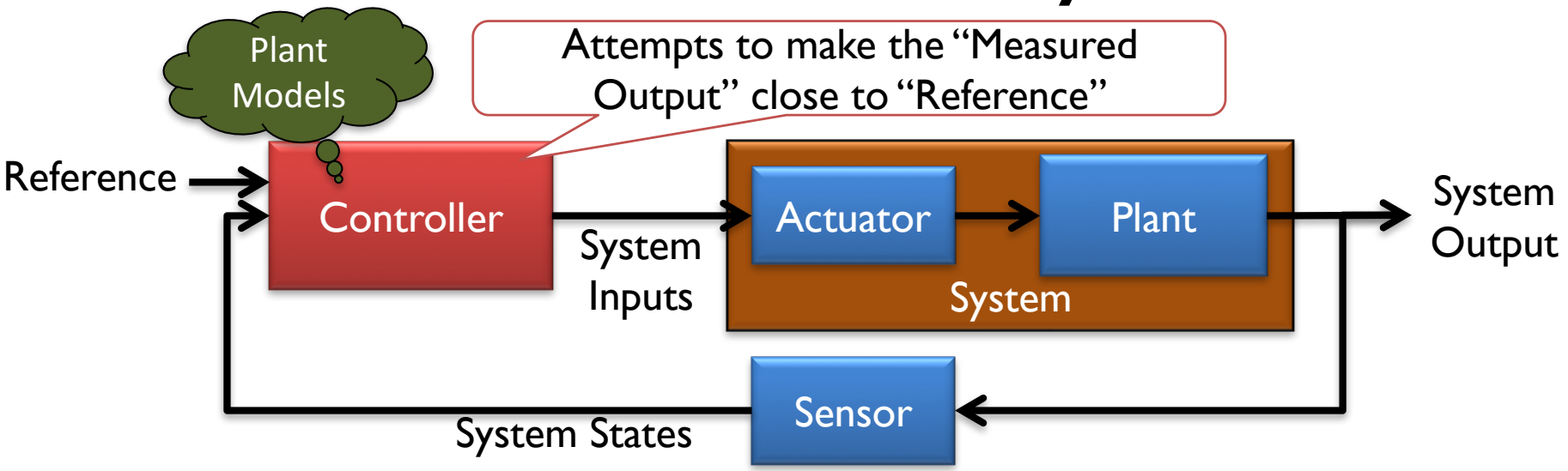
Outline of this talk

Let's predict future and
improve system management efficiency!

1: Improve energy efficiency of on-line object tracking

2: Accelerate performance of control systems

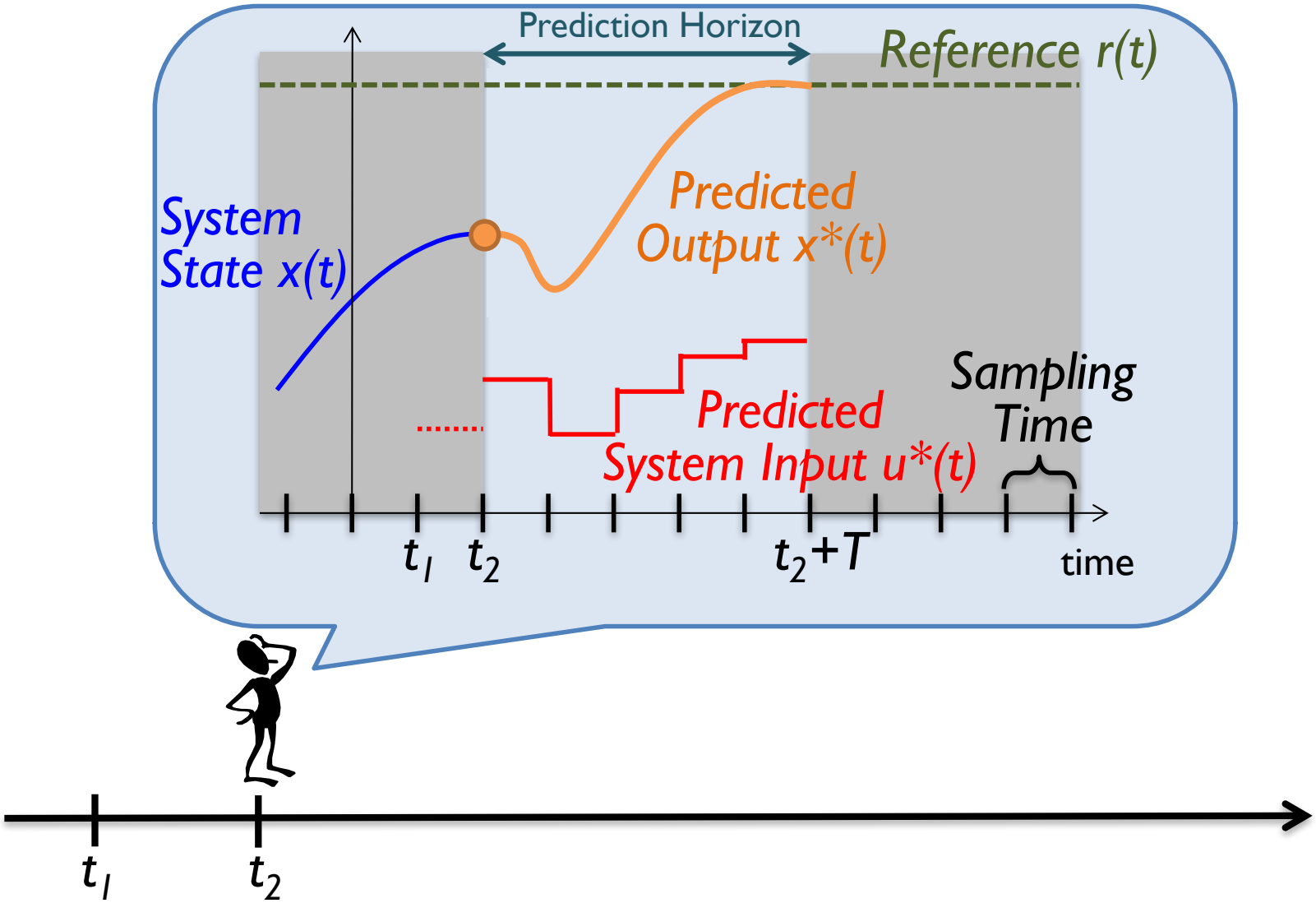
Feedback Control Systems



Real-Time Model Predictive Control (MPC)

- Features
  - Involves state equations (or plant models) in the controller
  - Decide how to manipulate the actuator based on predicted future plant behavior by solving optimal control problems
  - Needs to satisfy real-time periodical operations
- Problem
  - $O(N^3)$ computational complexity

Overview of Model Predictive Control

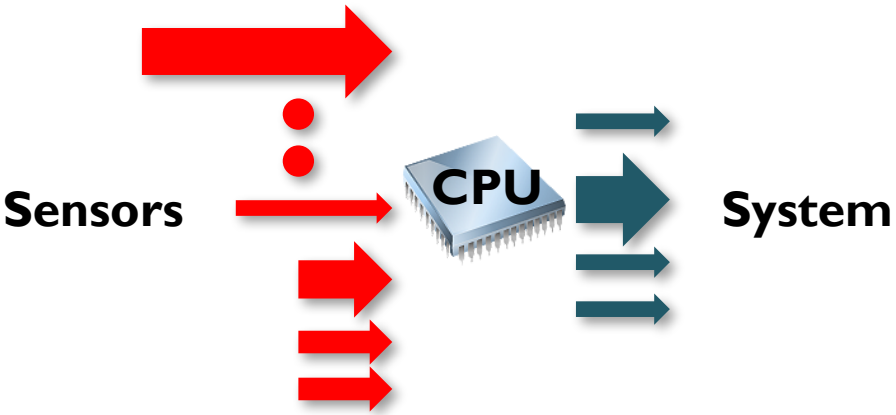


Can Manycore be a Solution?

Data-Level Parallelism

➔NO!

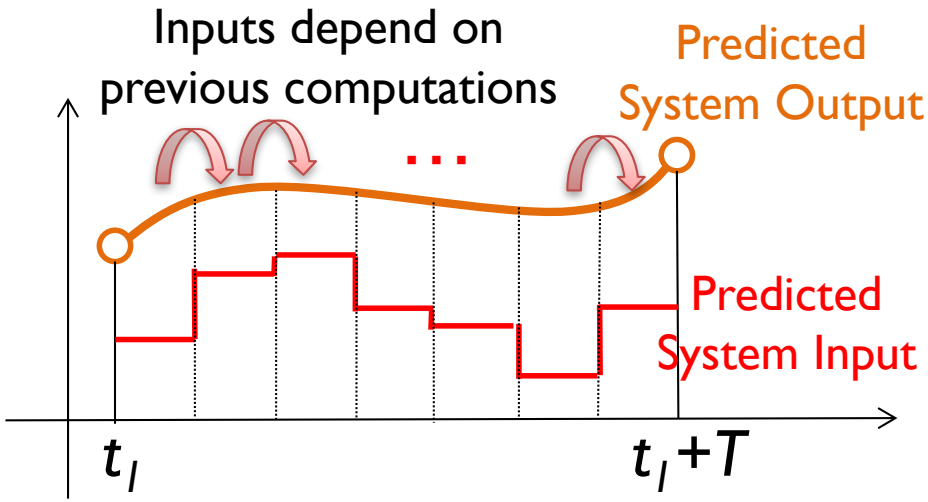
Small size data are fed sequentially!



Thread-Level Parallelism

➔NO!

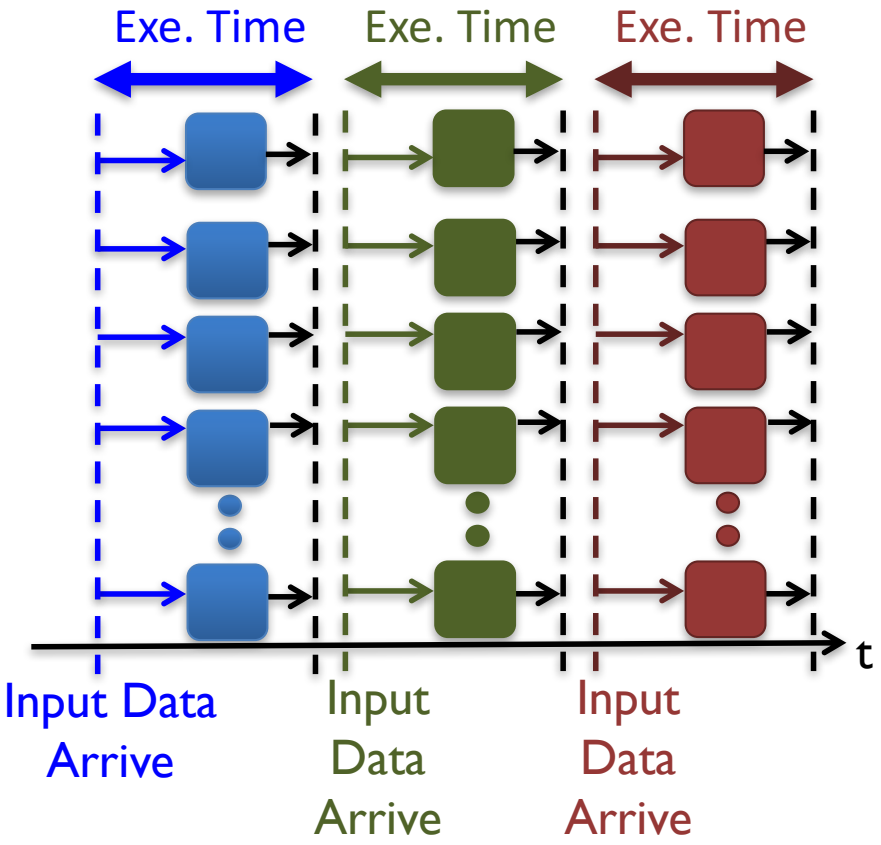
Each step is executed sequentially!



Our Approach
~ Speculative Execution on Manycore ~

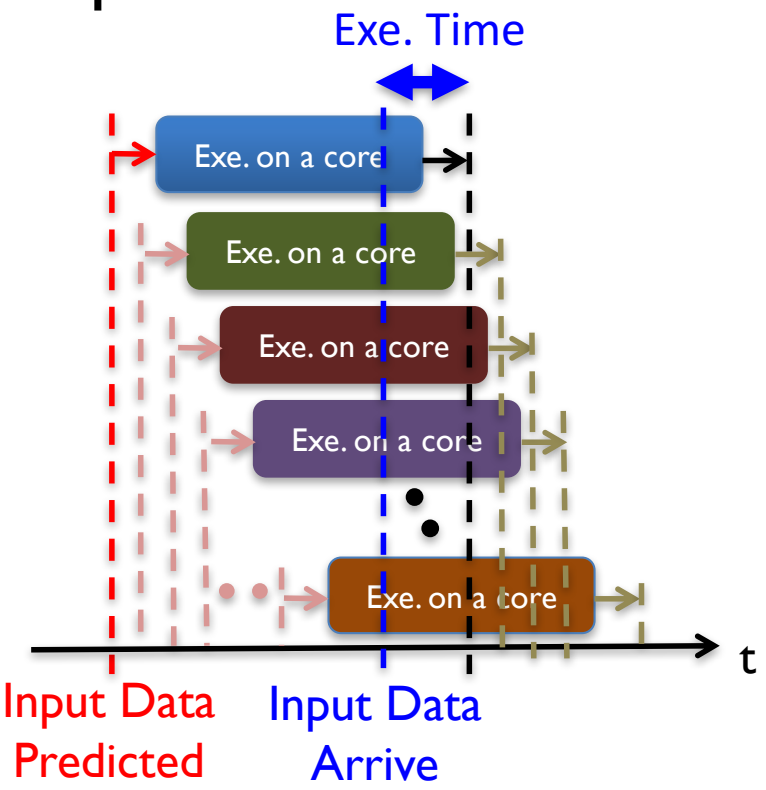
[Conventional]

Spatial Parallel Exe.

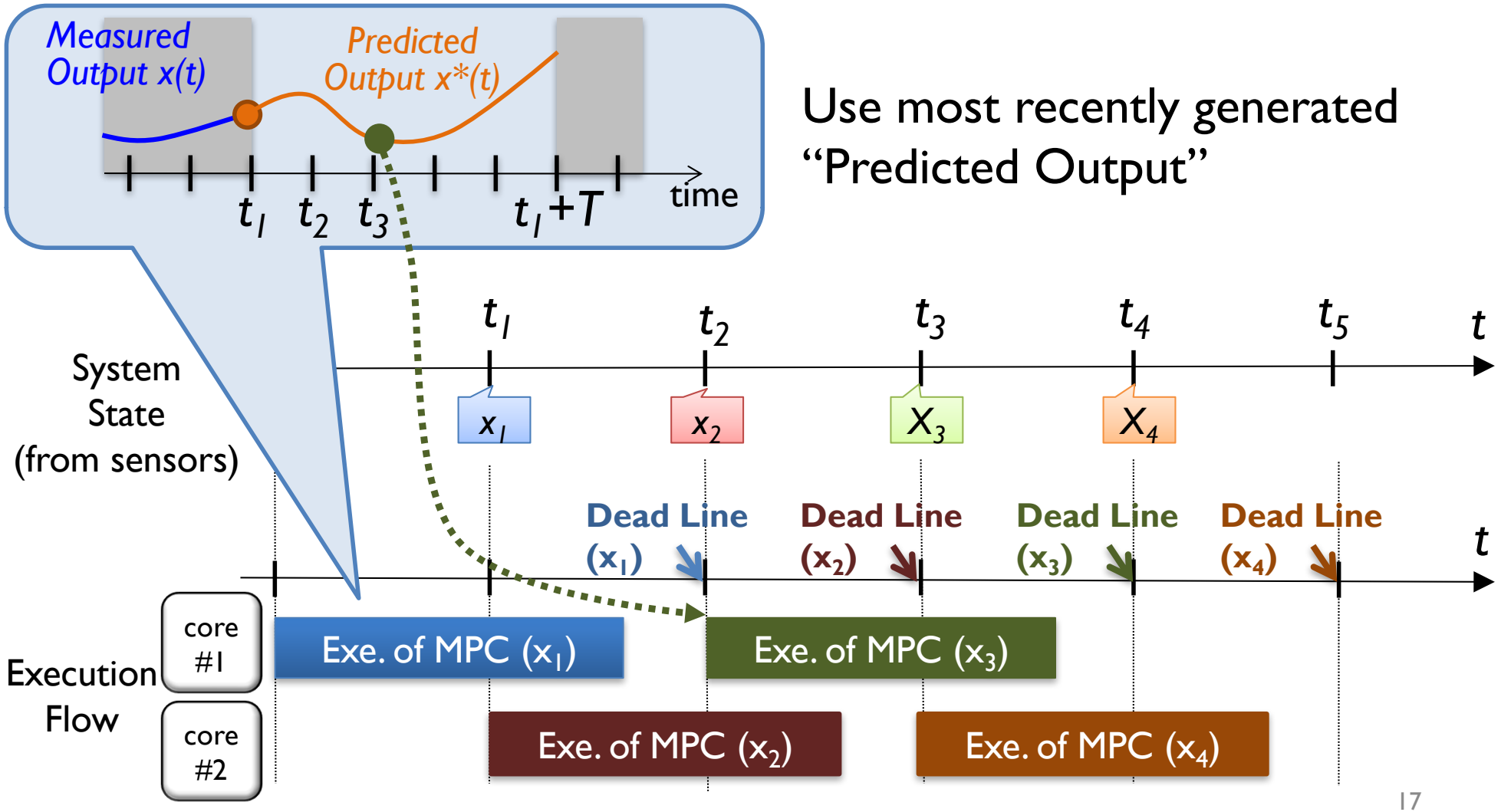


[Proposed]

Temporal Parallel Exe.
w/ Input Value Prediction

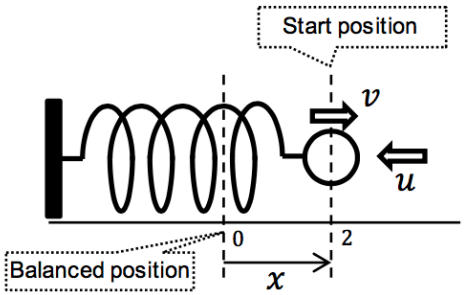


How Can We Predict Input Data for Speculative Executions?

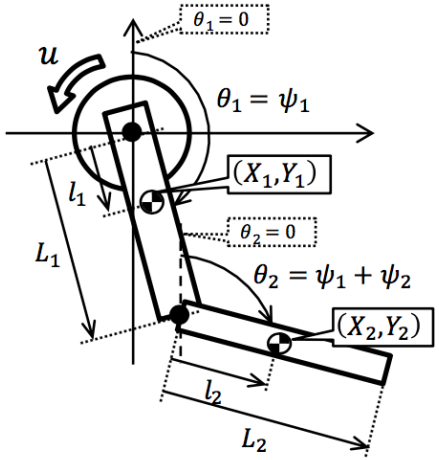


Experimental Setup

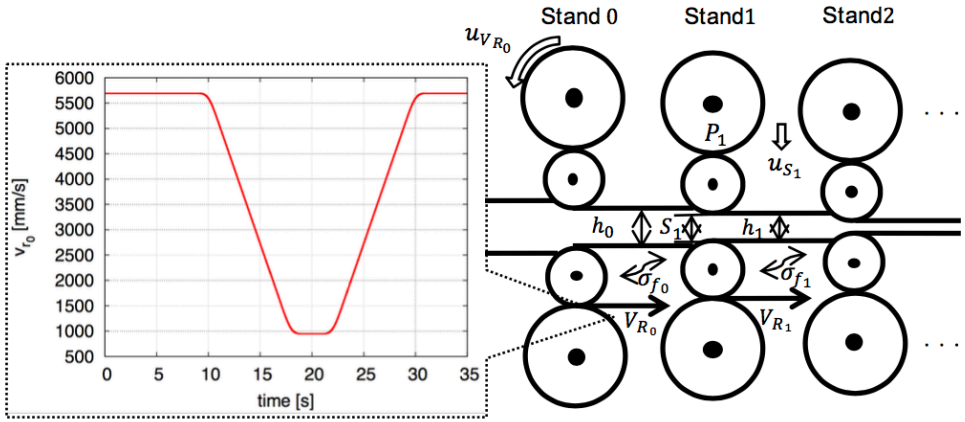
- Target 3 control systems
- Implement on a Xeon-Phi manycore platform



(a) Nonlinear spring



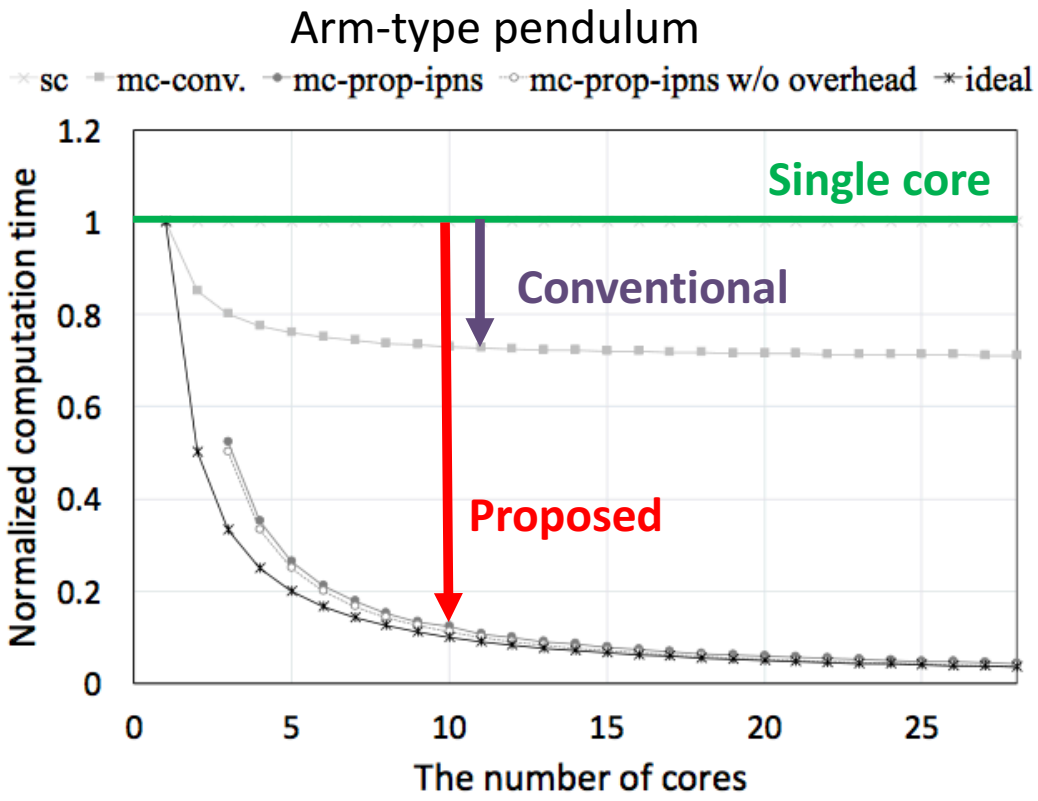
(b) Arm-type pendulum



(c) Tandem cold mill

Implementation Results

- Does not have any redundant cores.
- Implement a dedicated thread for prediction purpose.



Conclusions

- You can predict physical world!
- Predict sensor inputs, and then:
  - Improve energy efficiency,
  - Boost system performance,
  - and so on.
- X-layer optimization (sensing and controlling) is a key challenge!