oCPS

Platform-aware Model-driven Optimization of Cyber-Physical Systems

Designing image-based control systems considering workload variations*

Sajid Mohamed, Dip Goswami, Twan Basten

*In CDC 2019.

S. Mohamed, et al., “Optimising QoC of multiprocessor IBC systems considering workload variations,” In DSD 2018.

This project has received funding from the European Union’s Horizon 2020 Framework Programme for Research and Innovation under grant agreement no 674875.

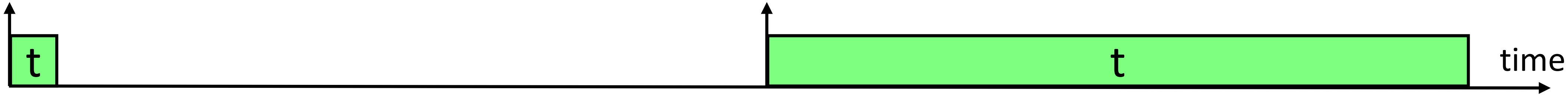
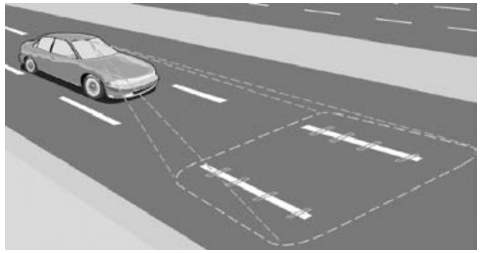
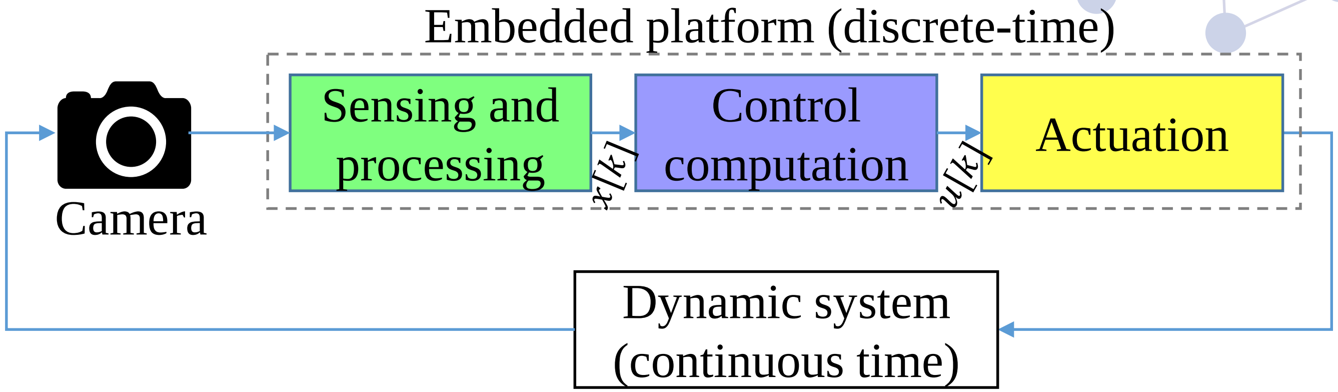
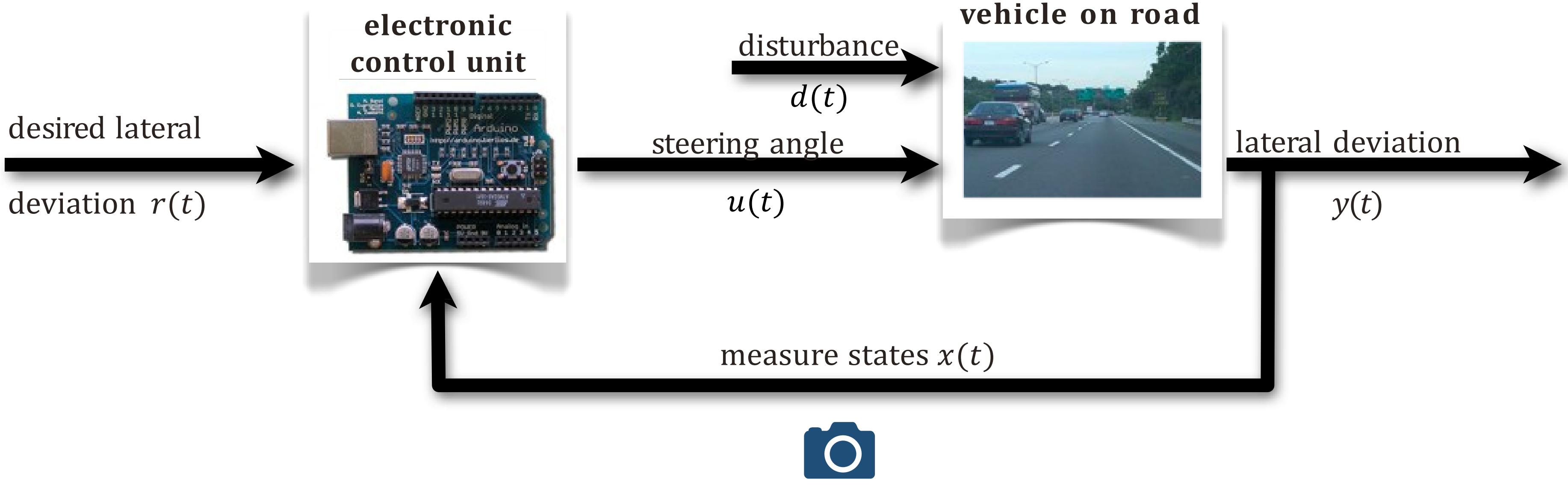

t
t
time

Image-based Control (IBC) System – an example



Vision-based lateral control of a vehicle





state-space model

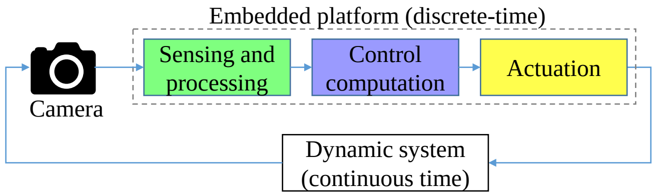
$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

control law

$$u(t) = Kx(t) + Fr(t)$$

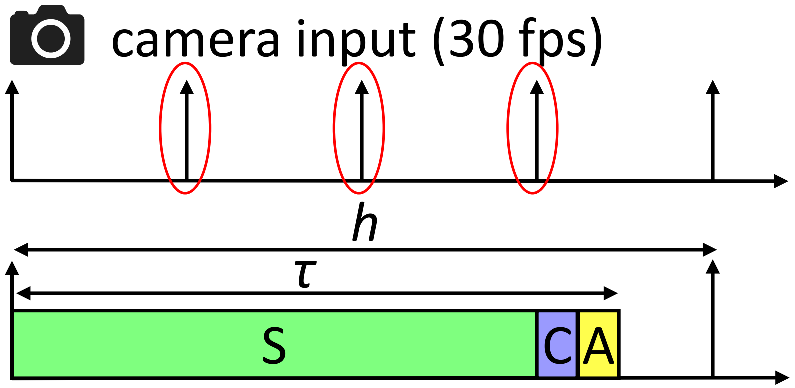
How to control the inputs $u(t)$ automatically to make the output $y(t)$ *track* the given reference $r(t)$?

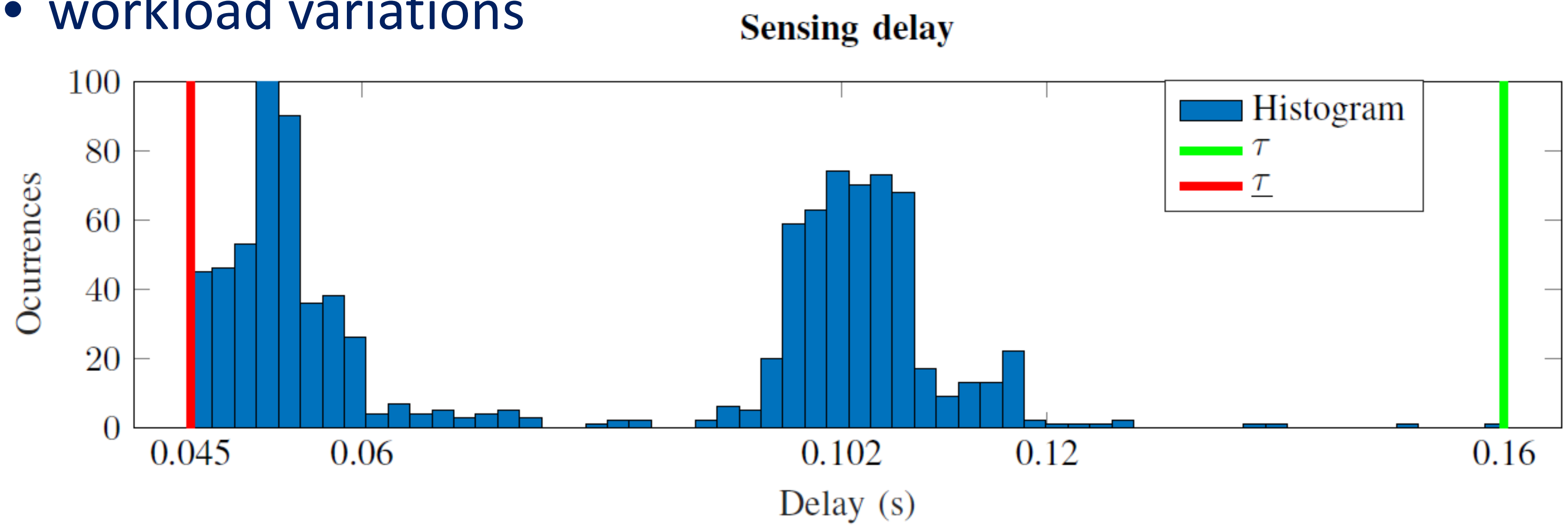
Control performance

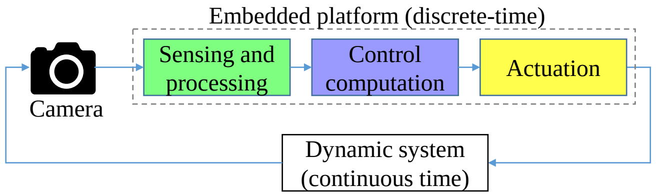


IBC system: characteristics

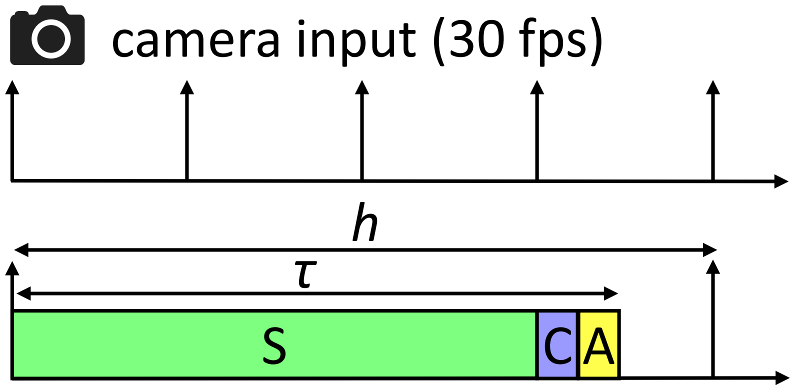
- Sensing delay is relatively long
  - applicable for 30 or 1000 frames per second (fps)
- Sensing delay more than frame rate
  - dropped camera frames
- Sensing delay is variable
  - workload variations



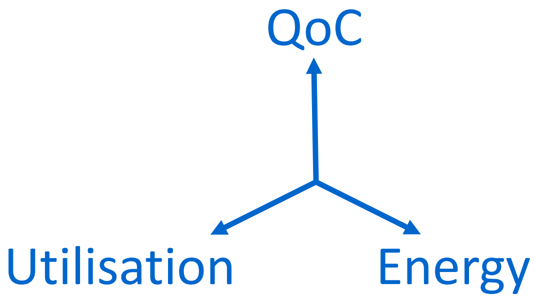




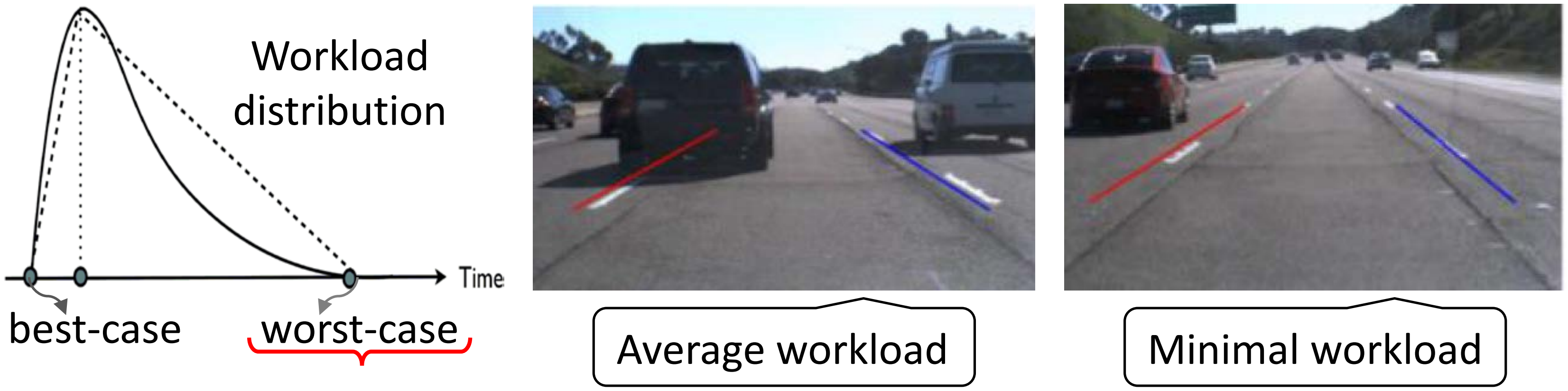
IBC system: characteristics

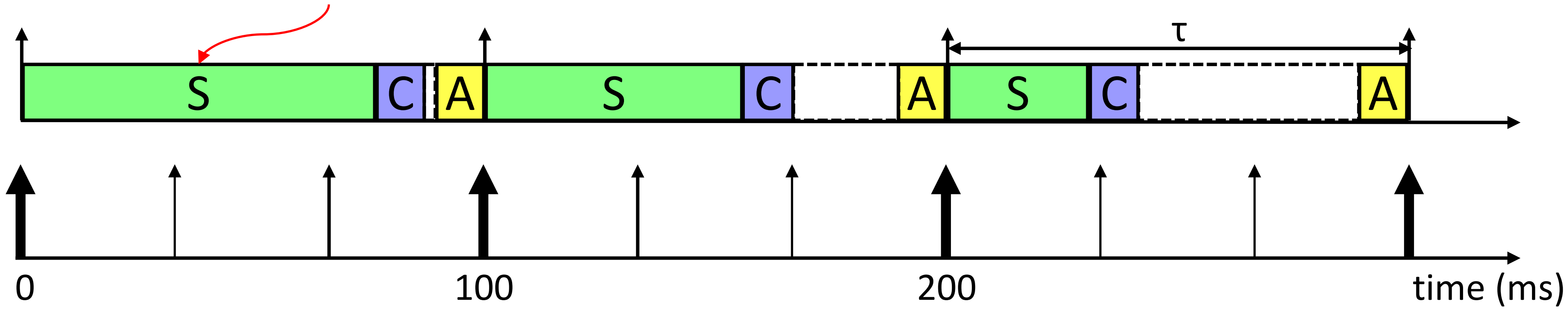


How to cope with this long, variable sensing delay
to improve system performance?



Workload Variations

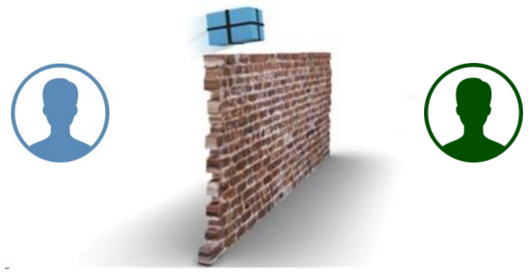




- Execution time of 'S' depends on workload variation
- A constant sensor-to-actuator delay τ and sampling period *h*
  → to guarantee system stability

Design vs Implementation

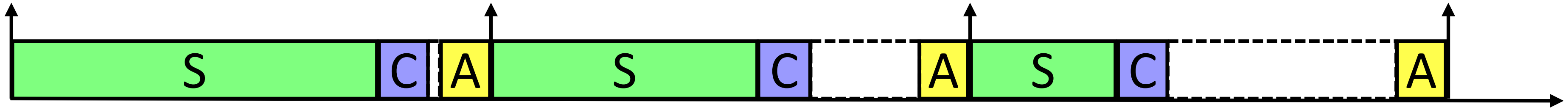
Control Design Engineer



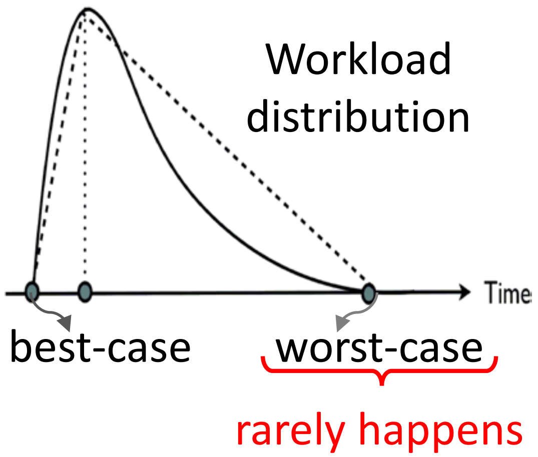
Embedded Systems Engineer

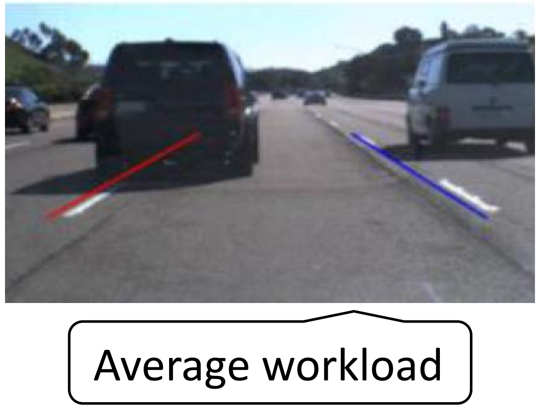
Worst-case workload is considered for design
Sensing task: black box

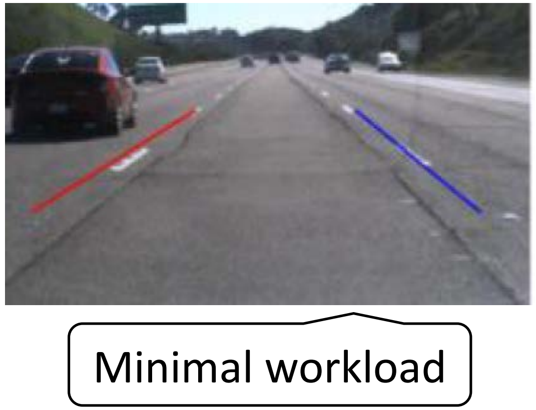
Allocates resources for the worst-case workload
Control design: black box

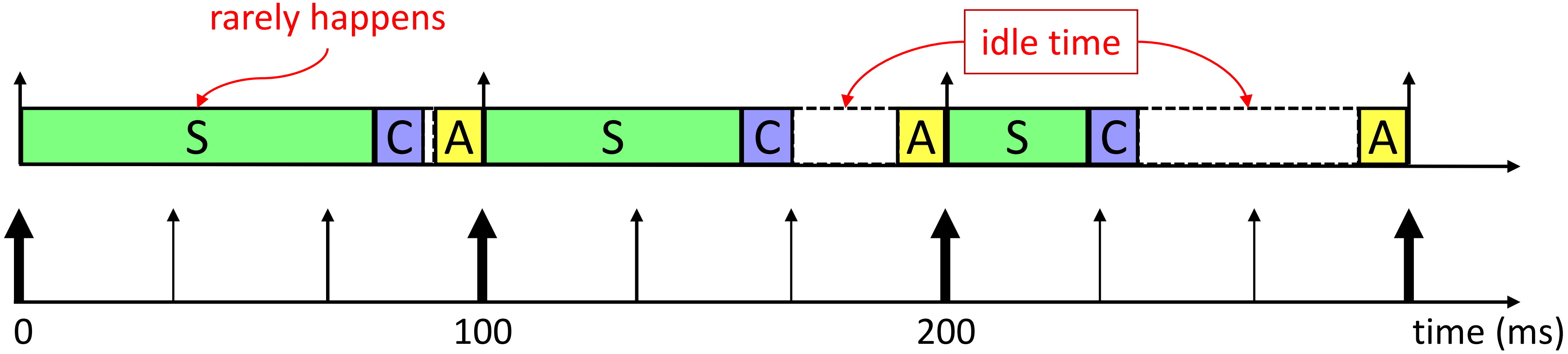


The Design-Implementation Gap



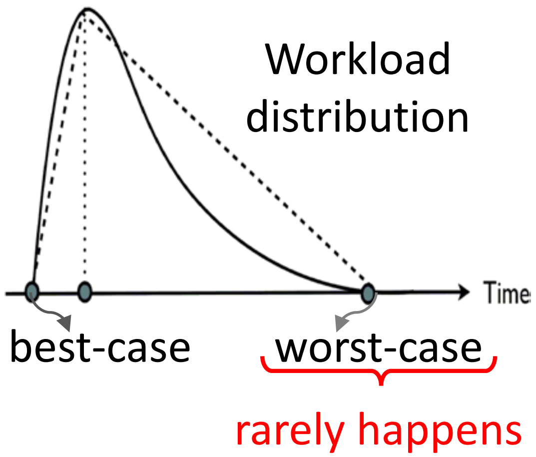


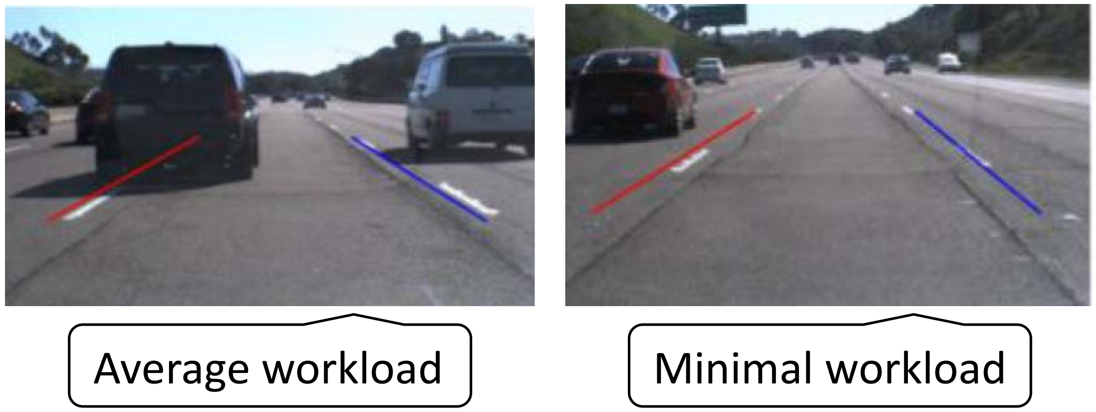


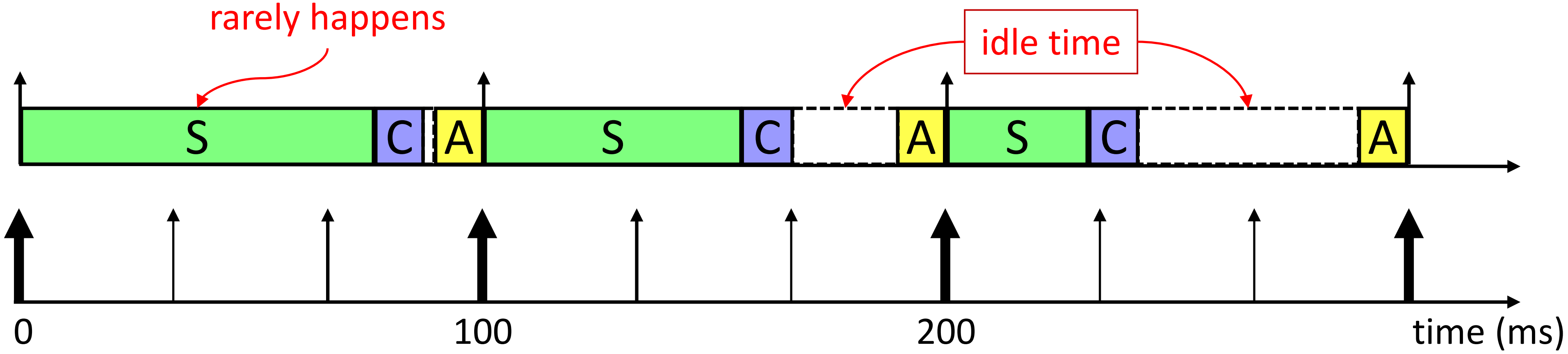


- Controllers designed for worst-case workload → rarely happens
- Idle resources for less workload → inefficient resource utilisation

Bridging the Gap

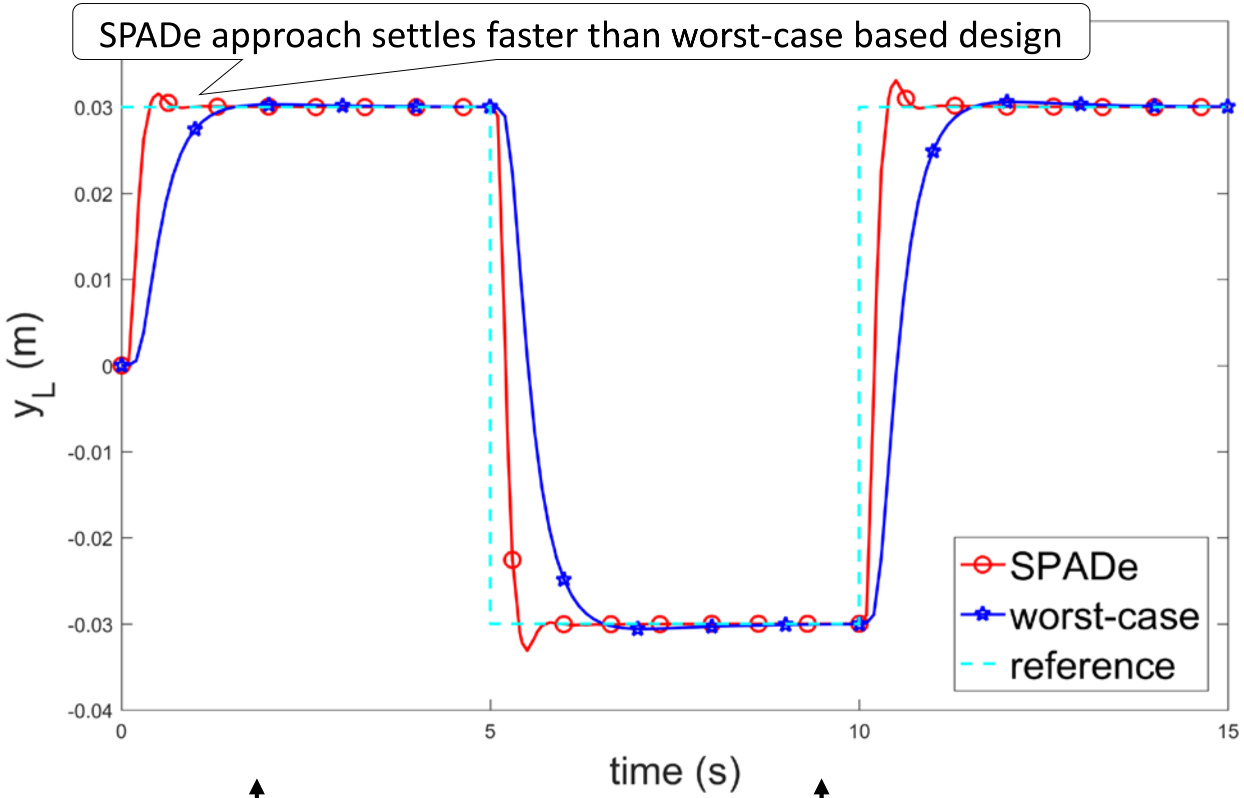


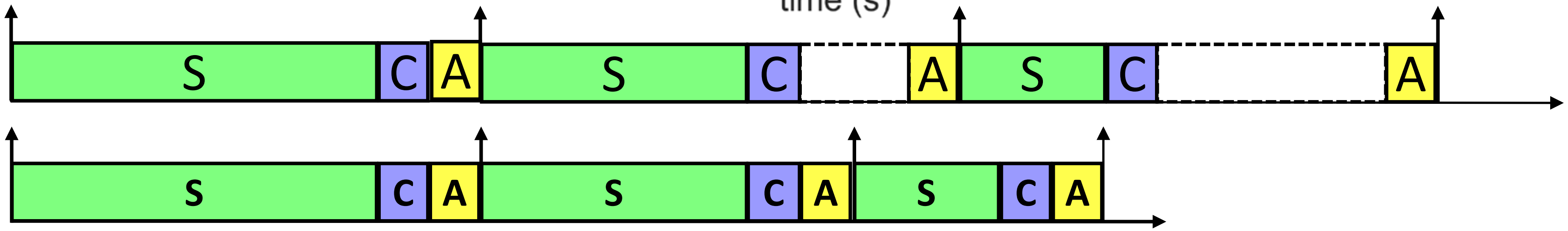




Can we optimise Quality-of-Control using multiprocessor technology?

Vision-based Lateral Control – Results



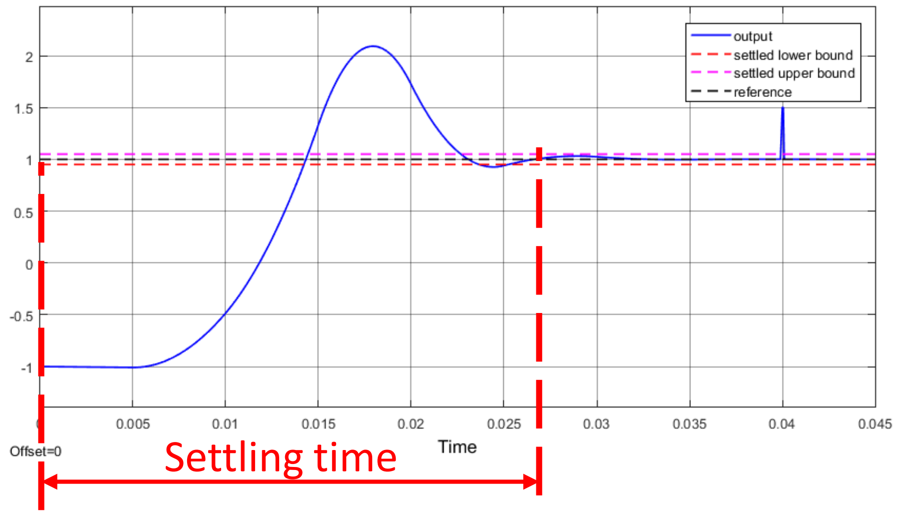


Can we optimise Quality-of-Control using multiprocessor technology?

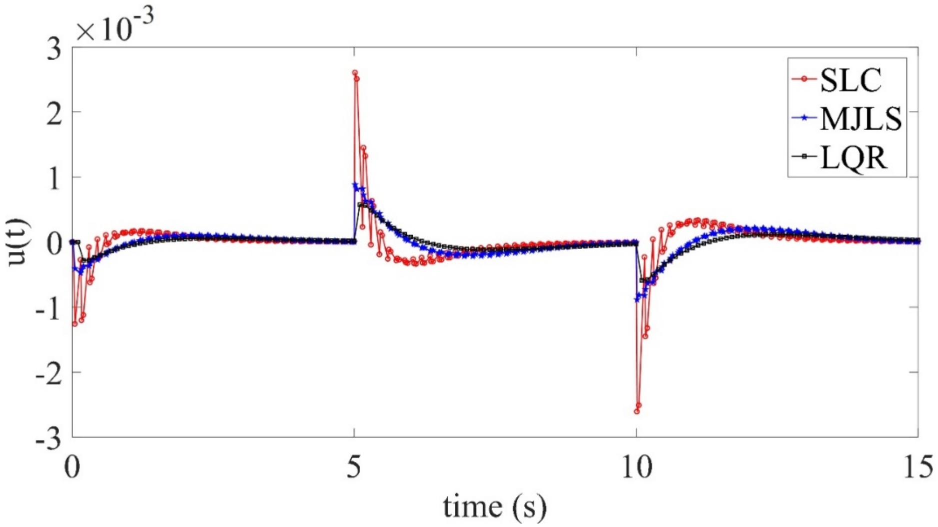
Yes, Scenario- and Platform-Aware Design (SPADe) approach.

Quality-of-Control (QoC) metrics

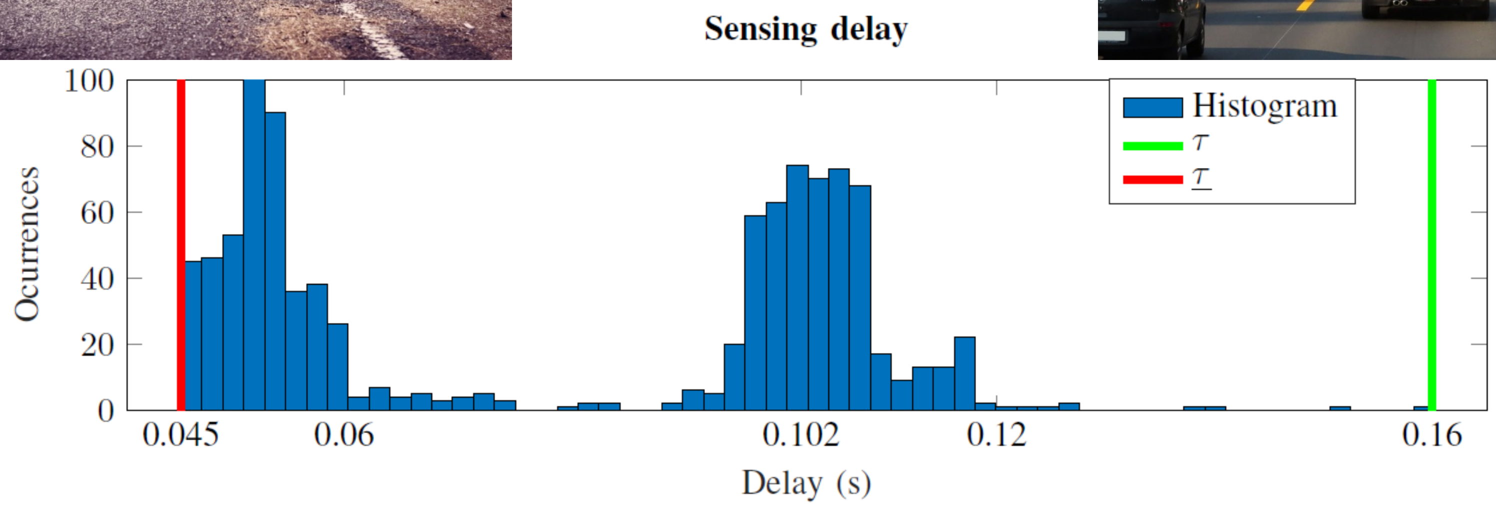
- Control performance
  - Settling time (ST)
    - vision-guided braking
  - Mean-square error (MSE)
    - $e = \frac{1}{n}\sum_{k=1}^{n}(x[k] - r)^2$
  - …



- Control energy/effort
  - Power spectral density (PSD)
  - Maximum control effort (MCE)
  - …



Scenarios based on workload



How to identify, model and characterise workload variations for IBC design?

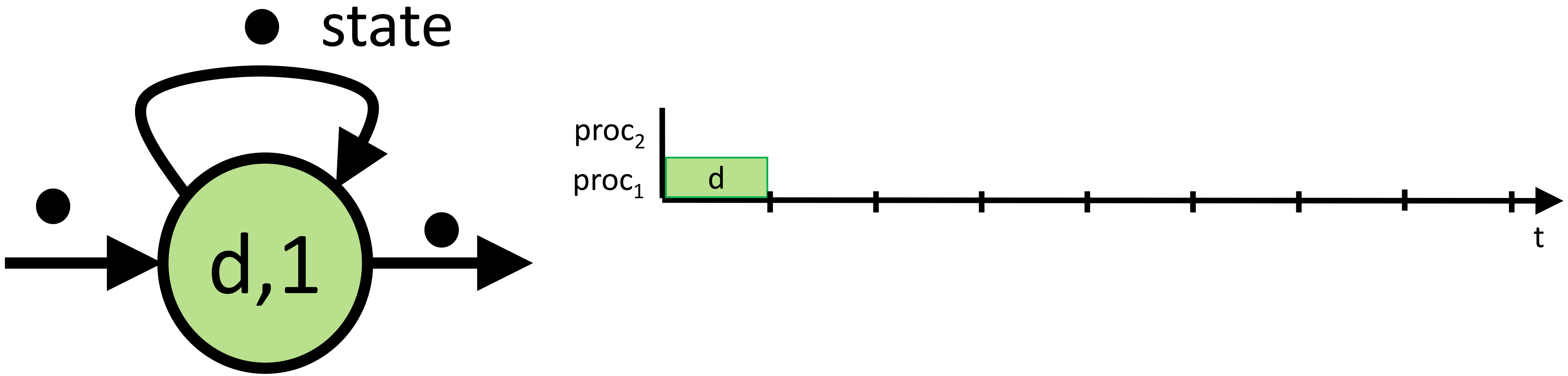
PERT distribution, Discrete-time Markov chain

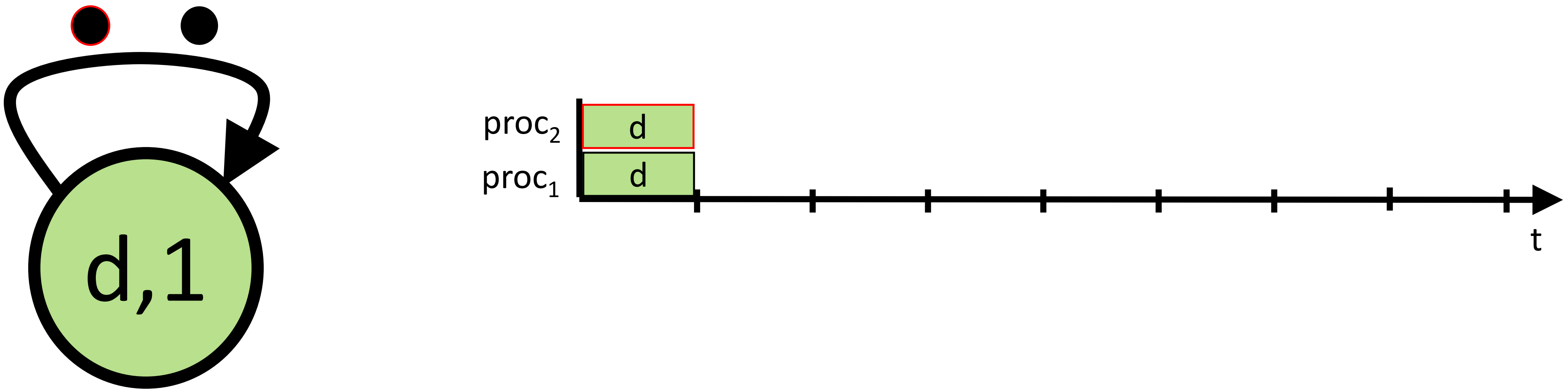
Why platform-aware?

- Model-of-Computation: synchronous dataflow (SDF)

Synchronous dataflow (SDF)

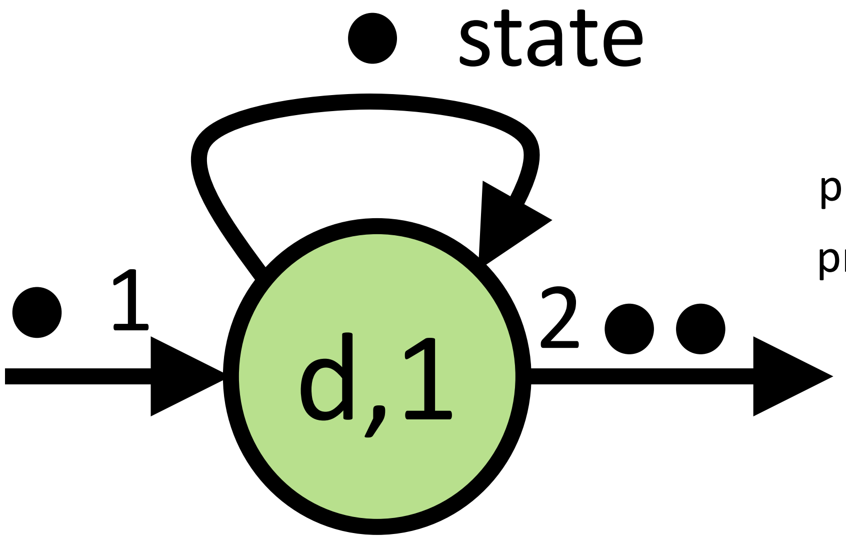
- Actor, channel, tokens, rates

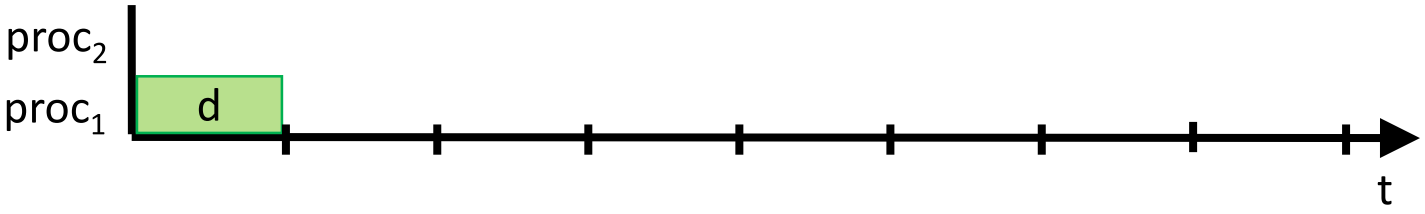


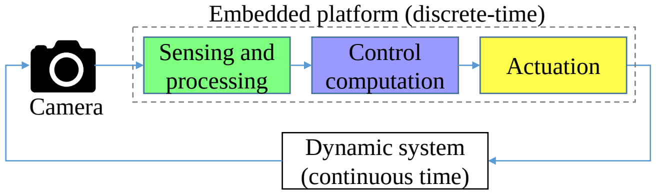


Synchronous dataflow (SDF)

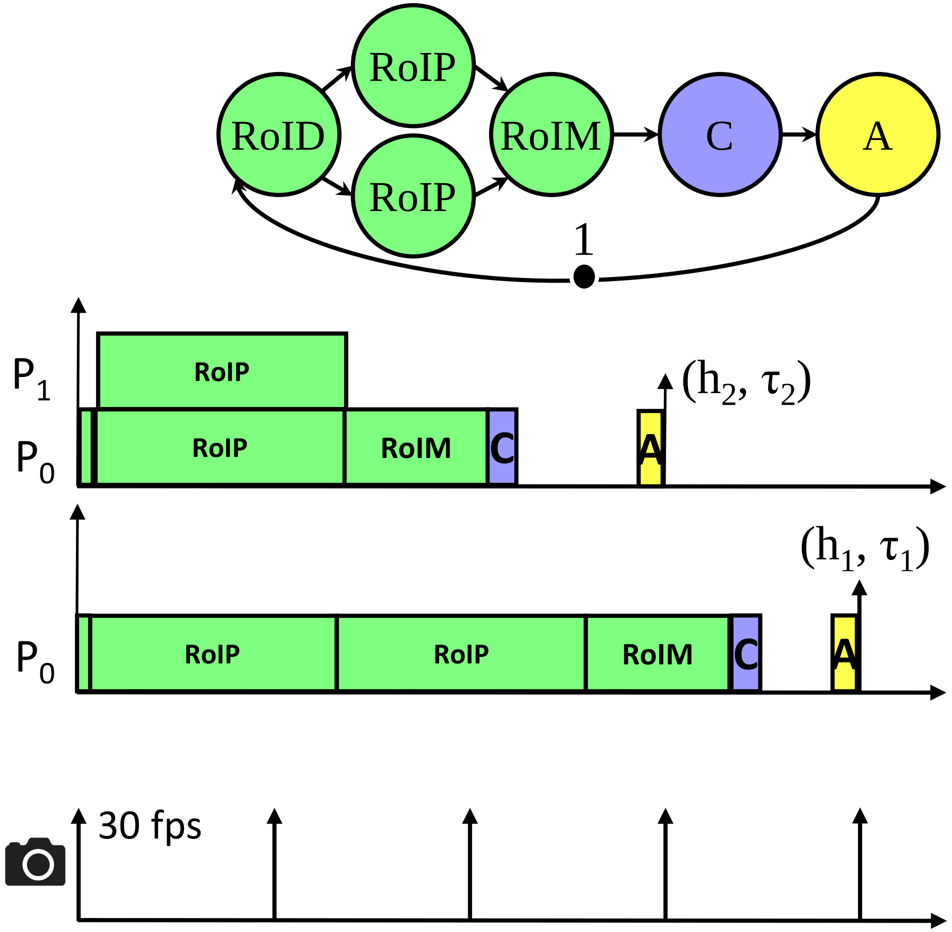
- Actor, channel, tokens, rates



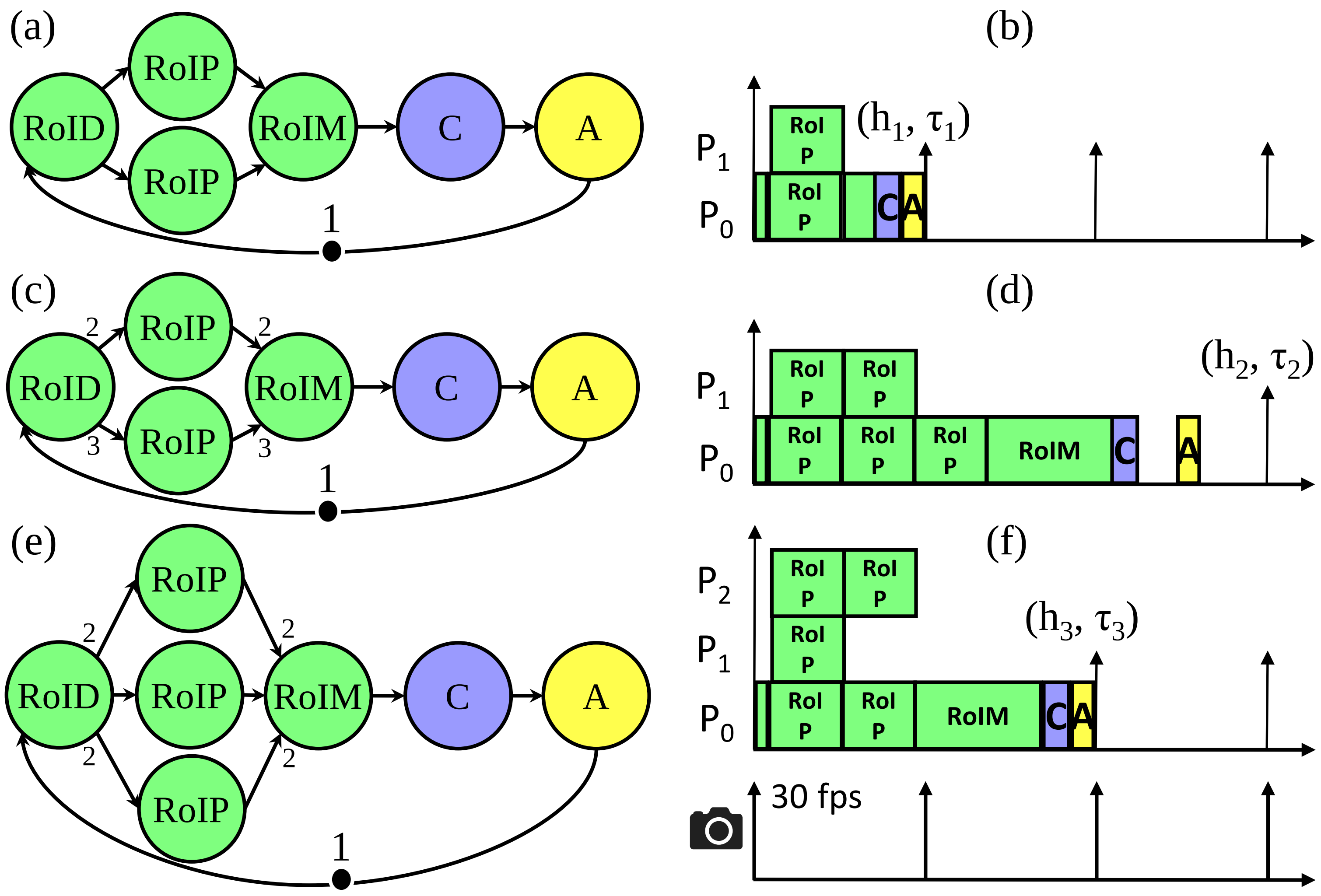




Why platform-aware?

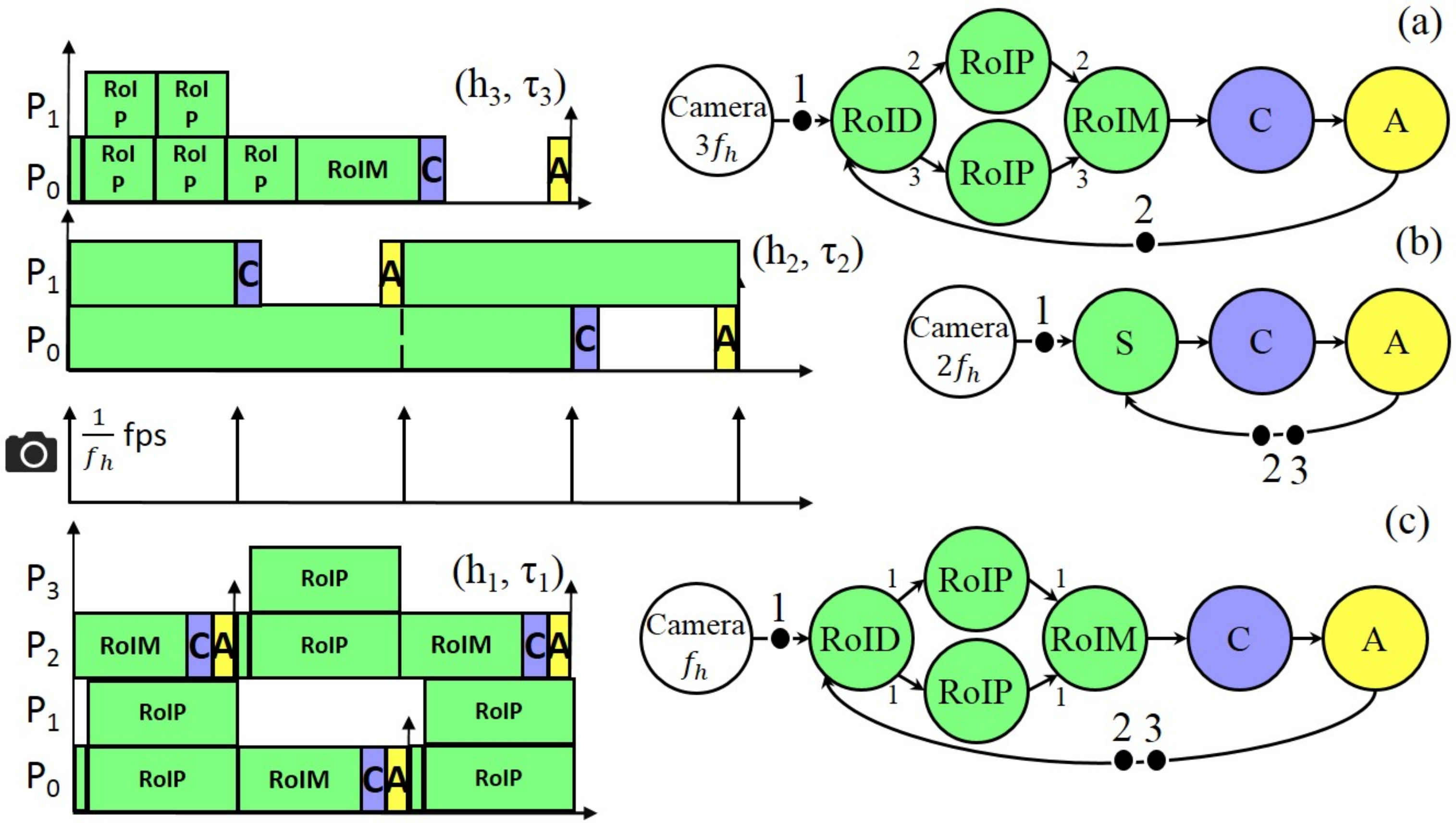


Workload scenarios

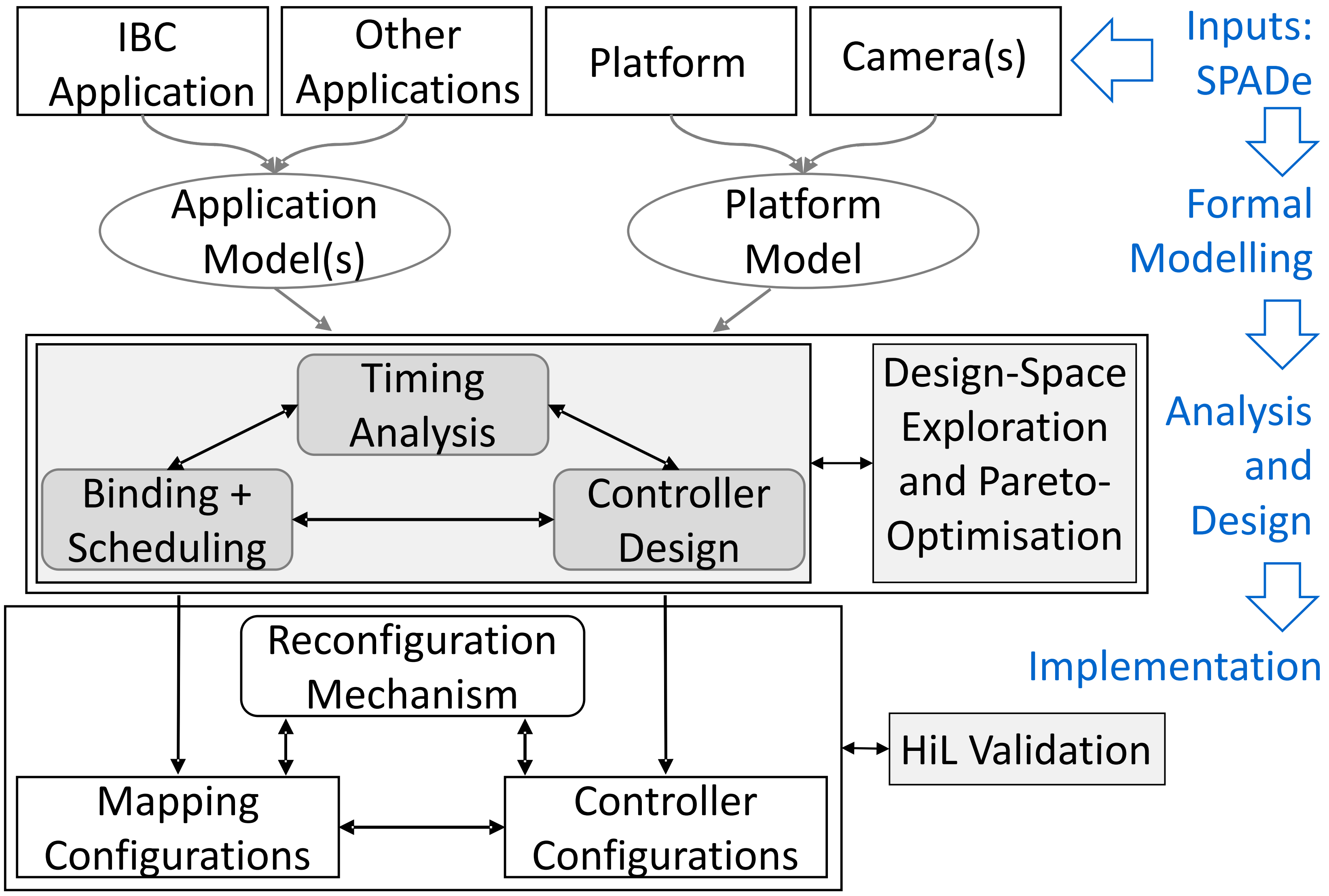


Pipelining and/or parallelization

- Throughput of the dataflow graph = sampling period
- Latency of the dataflow graph = sensor-to-actuator delay

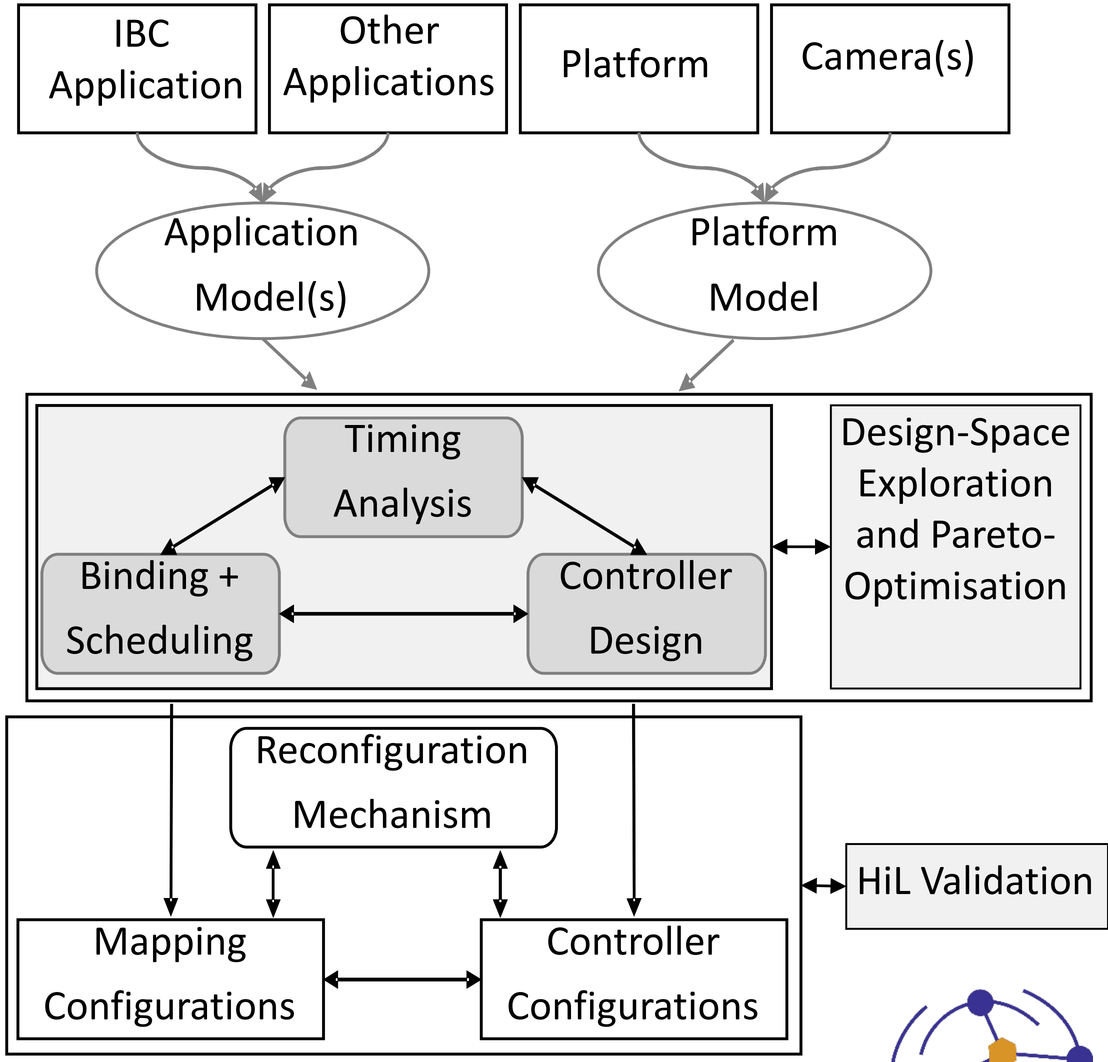


Scenario- and Platform-Aware Design flow



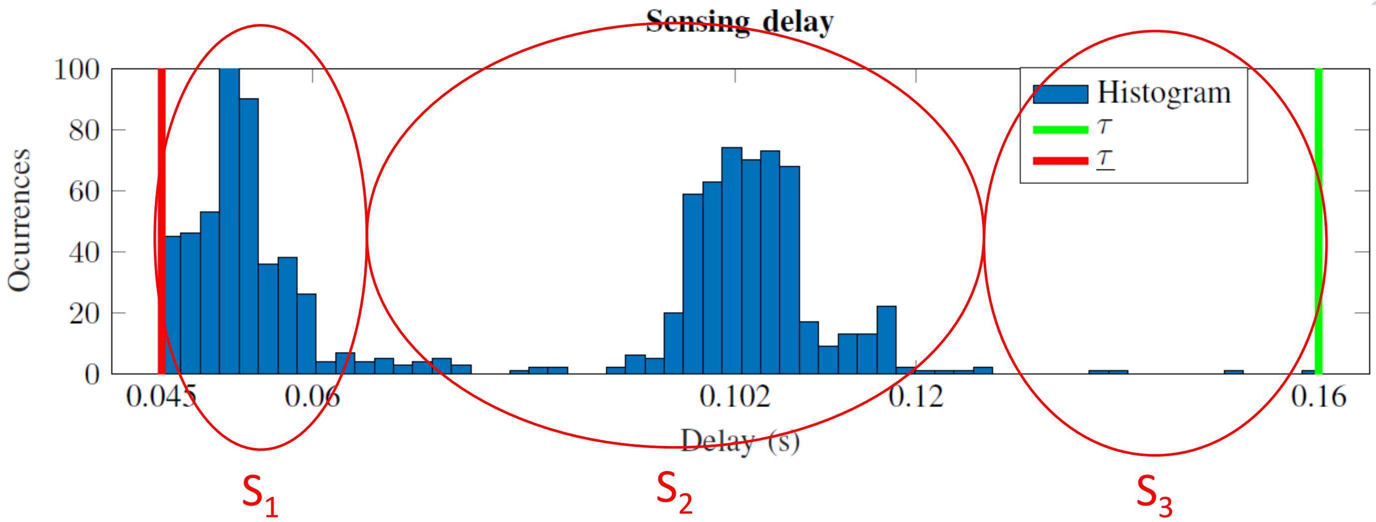
SPADe approach

1. Identify, model, and characterise workload (scenario) variations
   - PERT distribution
   - Discrete-time Markov chain
2. Find optimal mappings for a given platform allocation
   - SDF3 flow
3. Identify system scenarios
   - Implementation constraints
4. Design a controller
   - LQR with worst-case sampling period
   - Switched linear control (SLC) system
   - Markovian jump linear system (MJS)
   - Pipelined controller

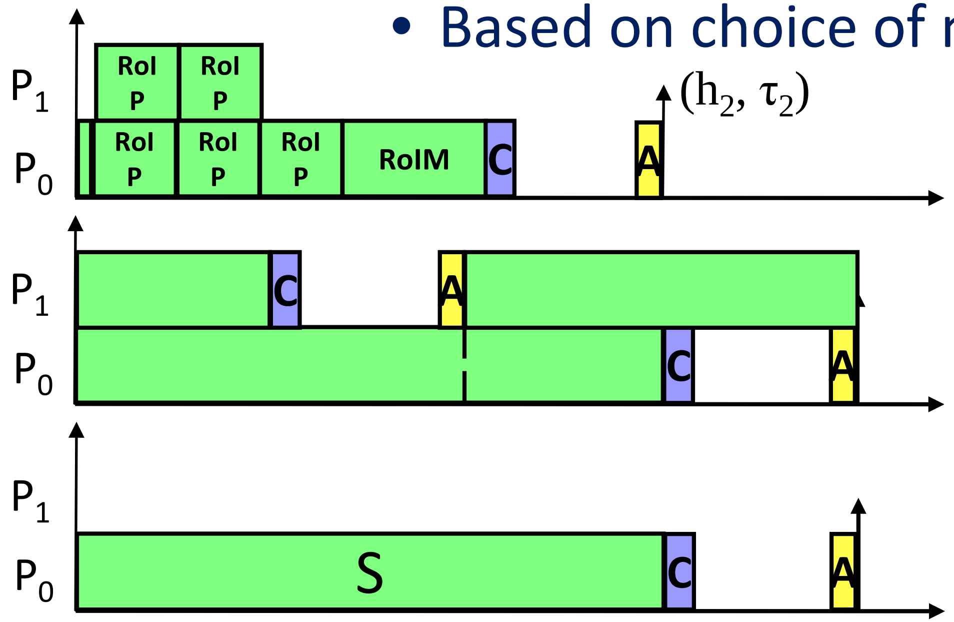


Scenario identification

- Based on workload



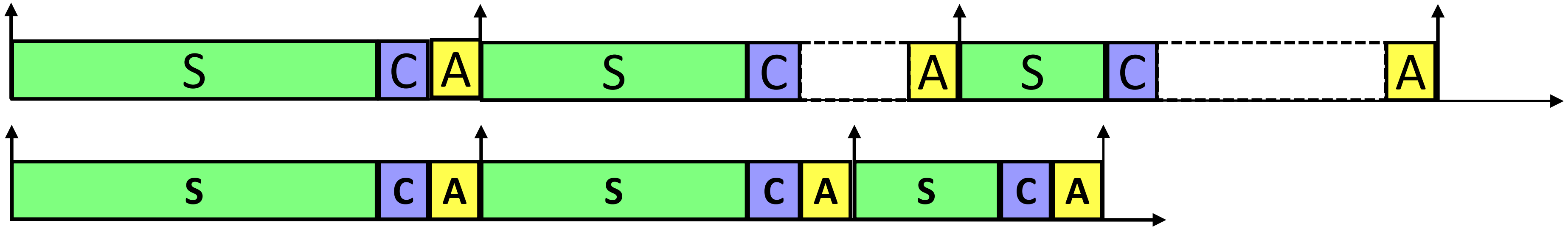
- Based on choice of mapping and choice of pipelining and/or parallelization



Control Design

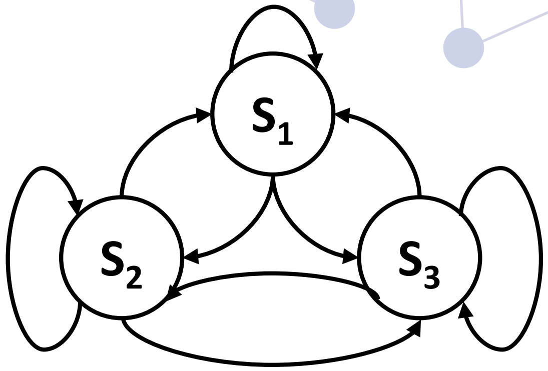
- LQR control
- Switched linear control (SLC)
  - Identify system scenarios (from e.g. PERT distribution)
  - Prove stability (common/switched quadratic Lyapunov function)

$$J(u) = \sum_{k=0}^{\infty} z[k]^T d_s C_{aug}^T C_{aug} z[k] + d_u^2 |u[k]|^2$$



Control Design

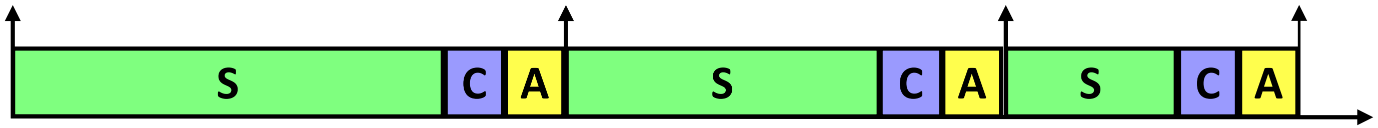
- Markovian jump linear system (MJS)
  - Model workload variations as discrete-time Markov chain



$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix}$$

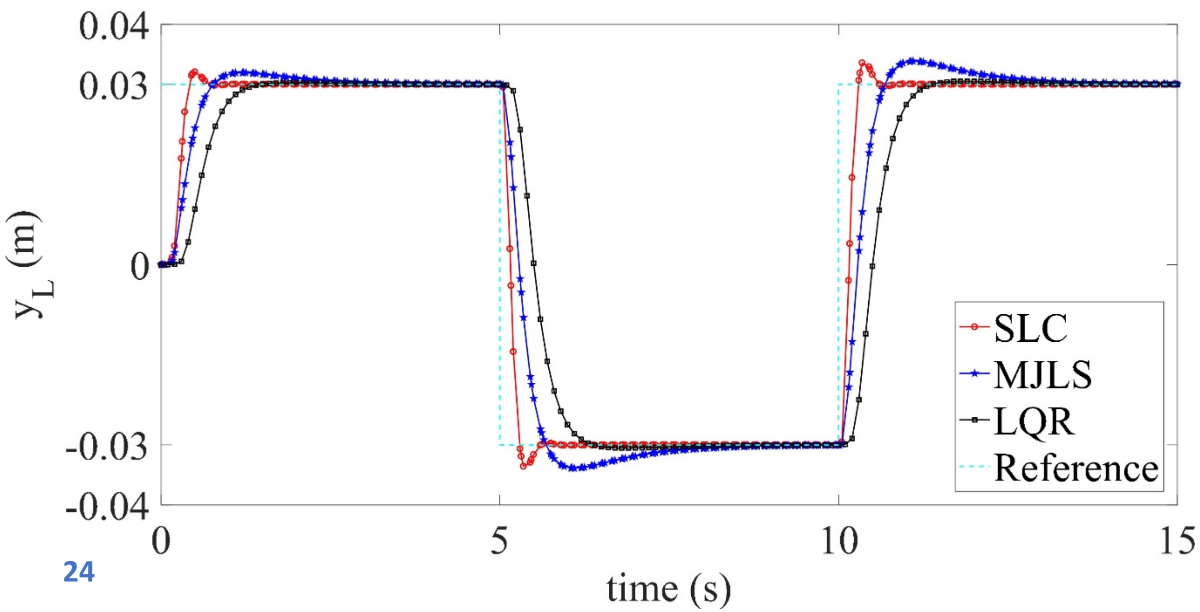
$$\begin{aligned} z[k+1] &= A_{aug,\theta[k]} z[k] + B_{aug,\theta[k]} u[k] \\ y_z[k] &= C_{aug} z[k] \end{aligned}$$

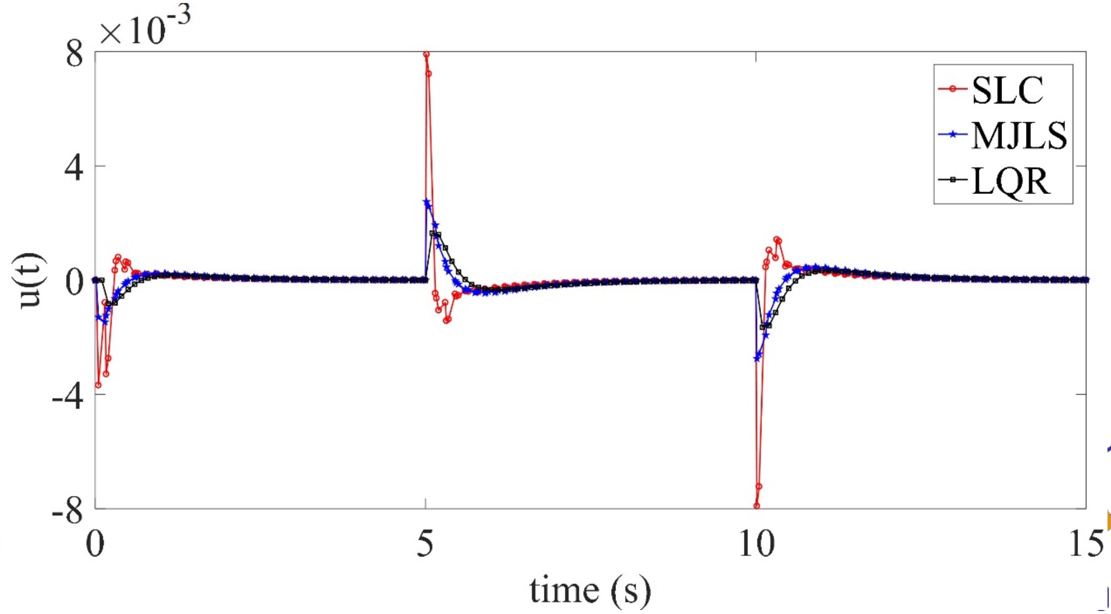
$$J(\theta[0], z[0], u[0]) = \sum_{k=0}^{\infty} \mathbb{E}[z[k]^T C_{aug}^T C_{aug} z[k] + d_u^2 |u[k]|^2]$$



Design guidelines

Available system knowledge	QoC metrics				
	Performance		Control energy		Performance and Energy
	MSE	ST	MCE	PSD	
Only worst-case workload information	LQR	LQR	LQR	LQR	LQR
Frequently occurring workloads as a PERT	SLC	SLC	LQR	LQR	SLC/ LQR
Frequently occurring workloads and their transition probabilities as a DTMC	SLC/ MJS	SLC/ MJS	MJS/ LQR	MJS/ LQR	MJS





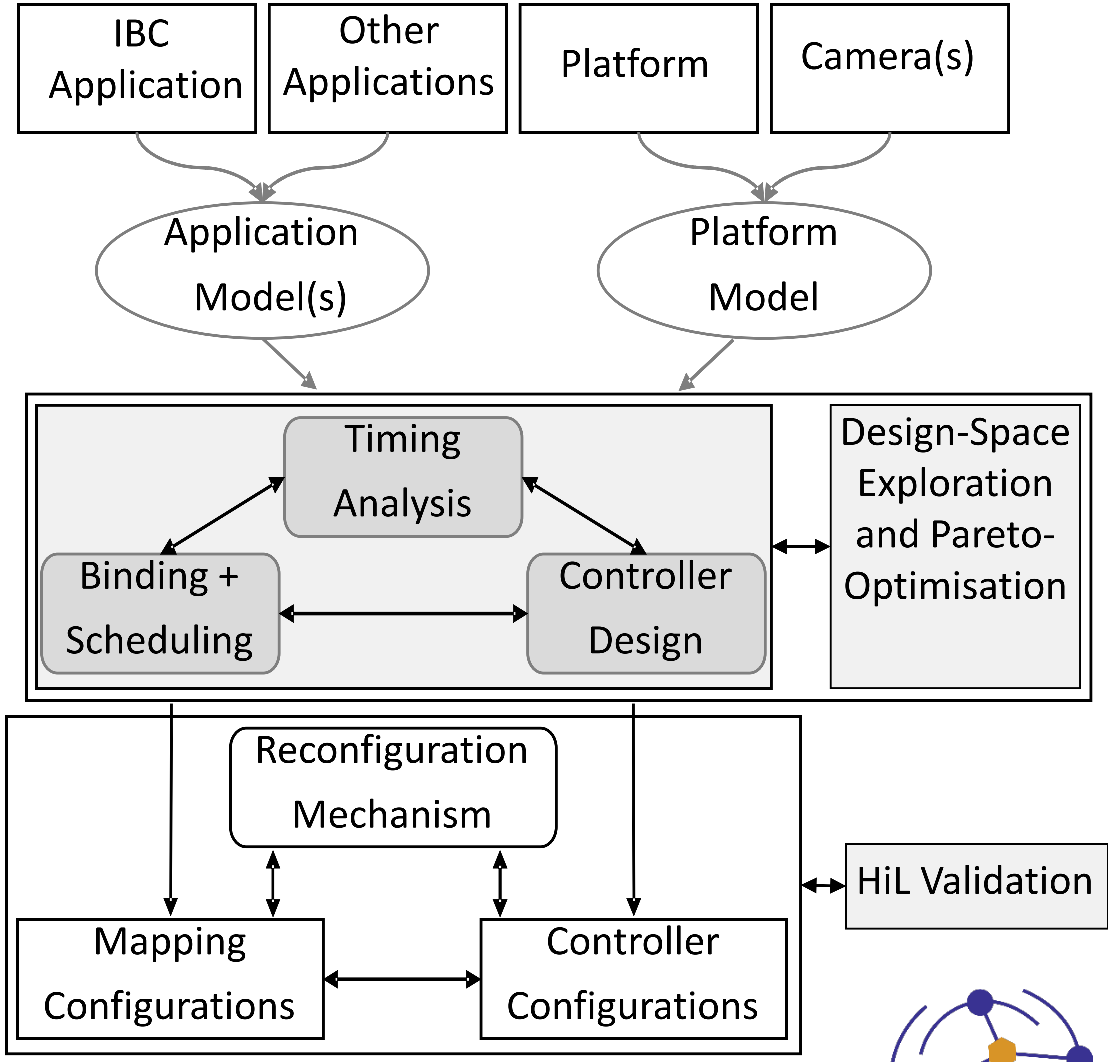
Conclusion

How to cope with long image sensing delay?

- SPADe approach
- Considering workload variations is beneficial
- Pipelining
- Parallelization

Future work:

- Develop I^{2}C tool based on SPADe approach
  - Inputs: application and platform model, requirements and constraints
  - Output: System configurations and controller code



ES ELECTRONIC SYSTEMS TU/e

S.Mohamed@tue.nl

linkedin.com/in/sajidmohamed

oCPS