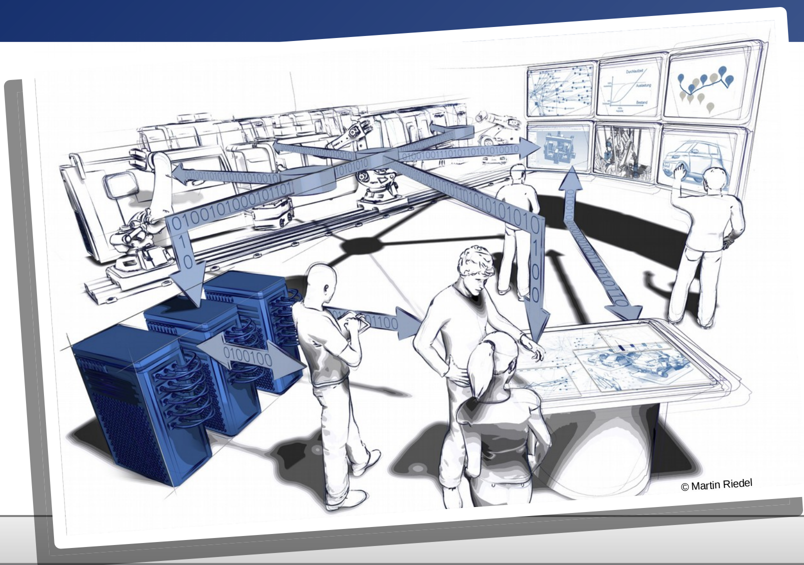


In-Network-Processing in Industrial Networks

Klaus Wehrle

Industrial Networks in Times of CPS and Industry 4.0

Networked control

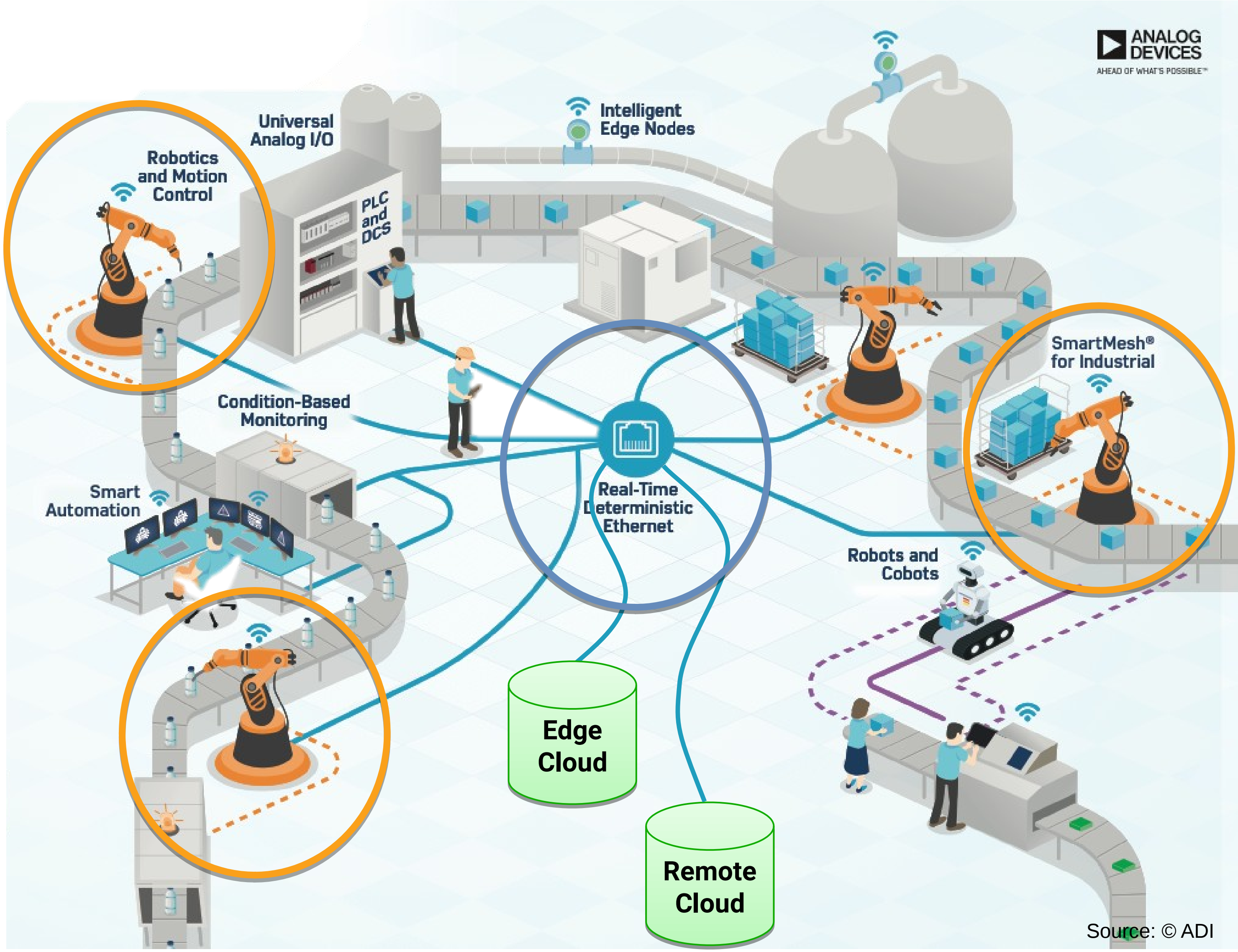
- Control loops at ultra low latency

Collect Process Data

- High data rate
- Data stream processing
- s/t immediate feedback

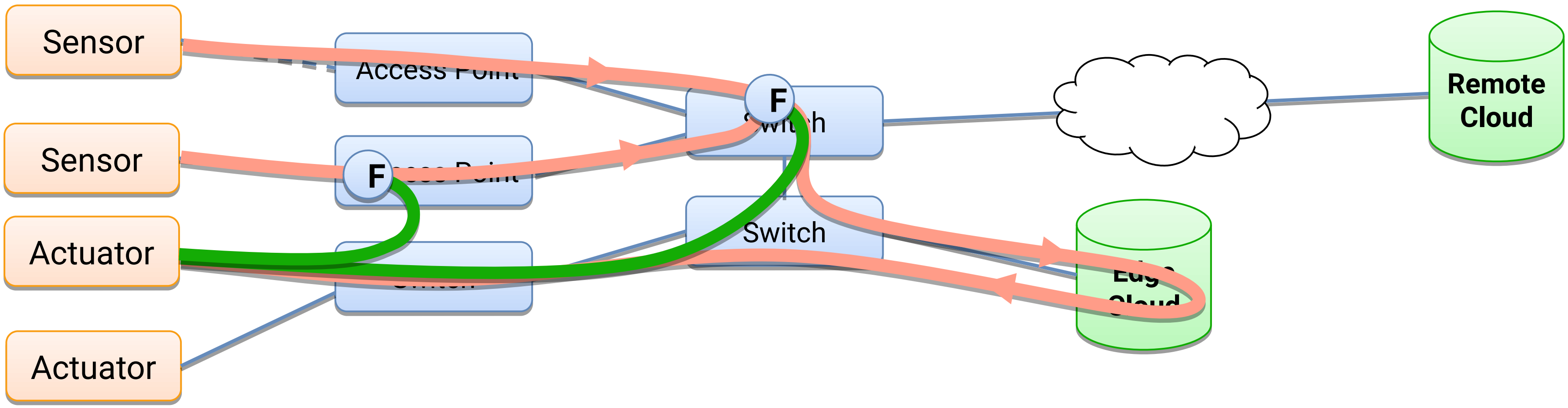
Offline Data Analysis

- Model extraction
- Data mining
- Machine learning
- Feedbacks new models for previous tasks



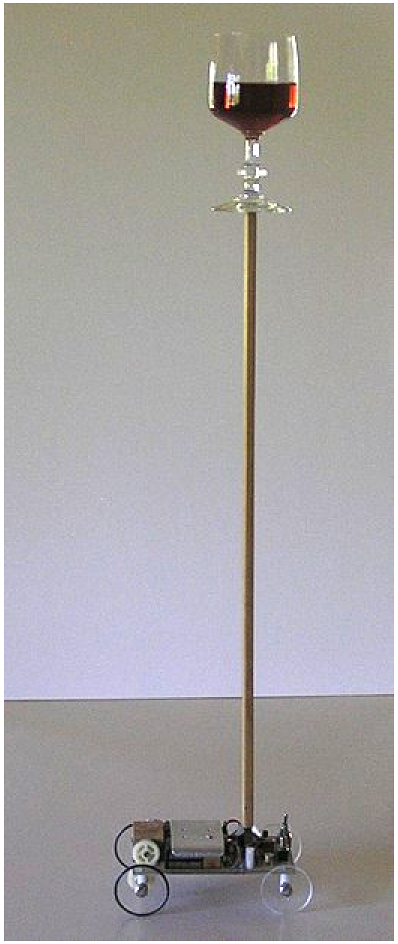
Source: © ADI

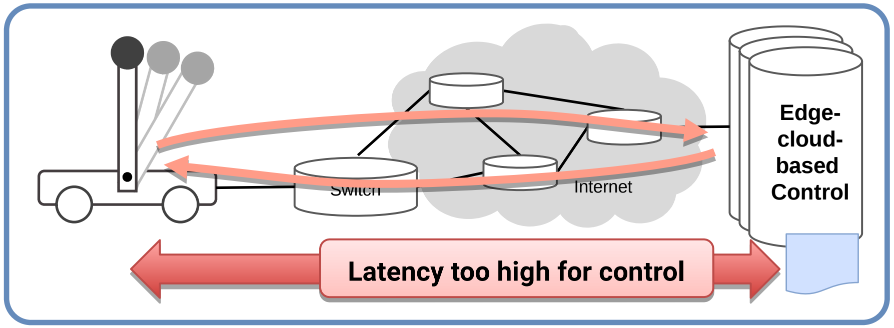
Low Latency Networked Control Loops



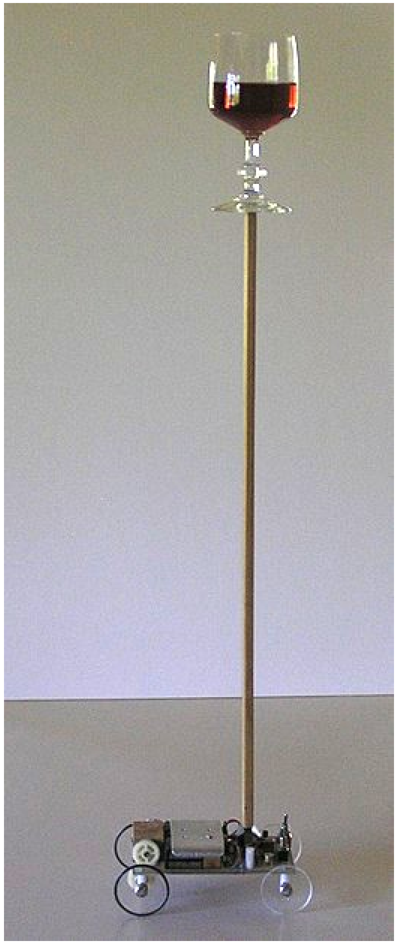
- **Networked control via edge cloud**
  - Remote cloud not feasible
  - Edge cloud still has higher latency – often missing real-time capability
- **Task separation: fast reaction by INP – slow processing in cloud**
  - Use computation in the network to execute simple tasks
    - Push simplified control algorithm (reflex) to the switch
    - Main control algorithm stays in edge cloud to do delay-insensitive adaptation
    - Cloud updates reflex if necessary, e.g. latency change, process is mobile, etc.

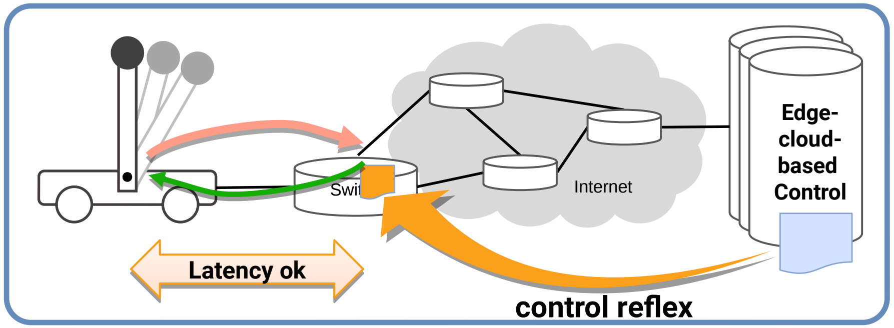
Academic Example: Balancing an Inverted Pendulum





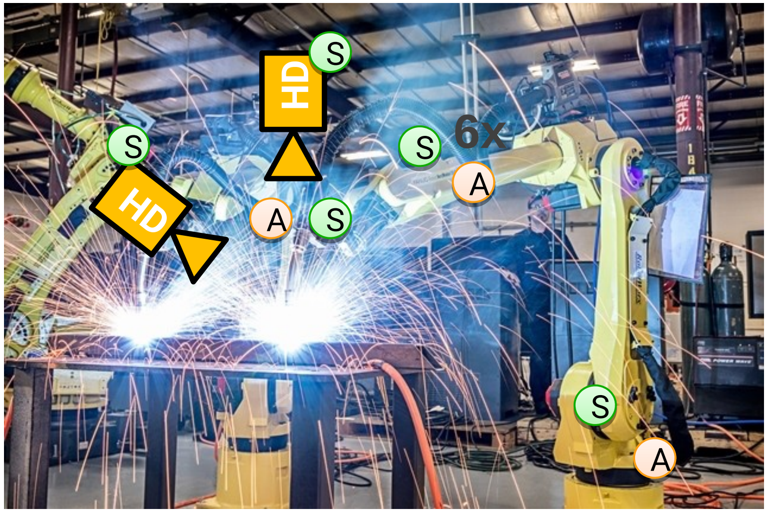
Academic Example: Balancing an Inverted Pendulum



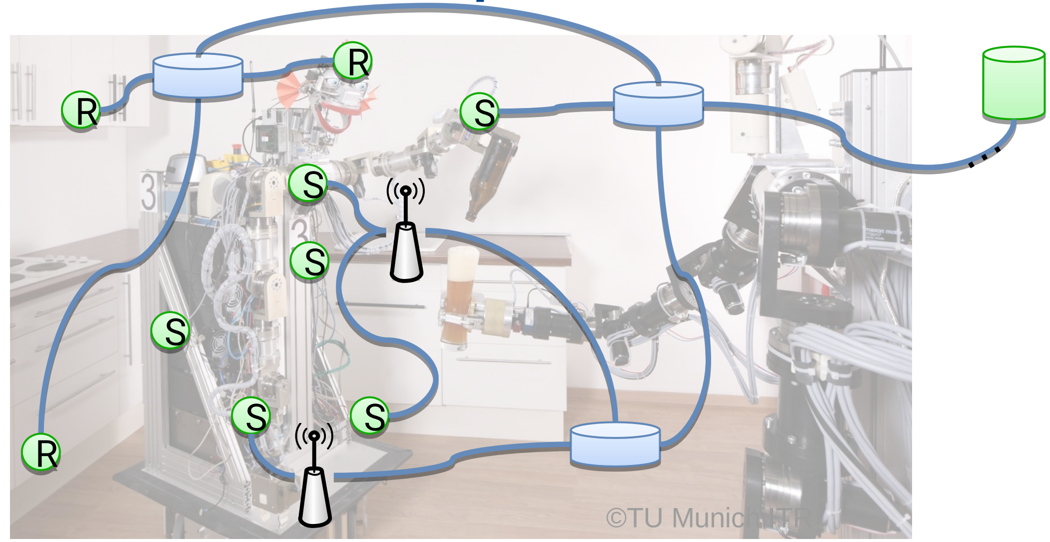


Two Real-world Examples

- **Arc welding robots**



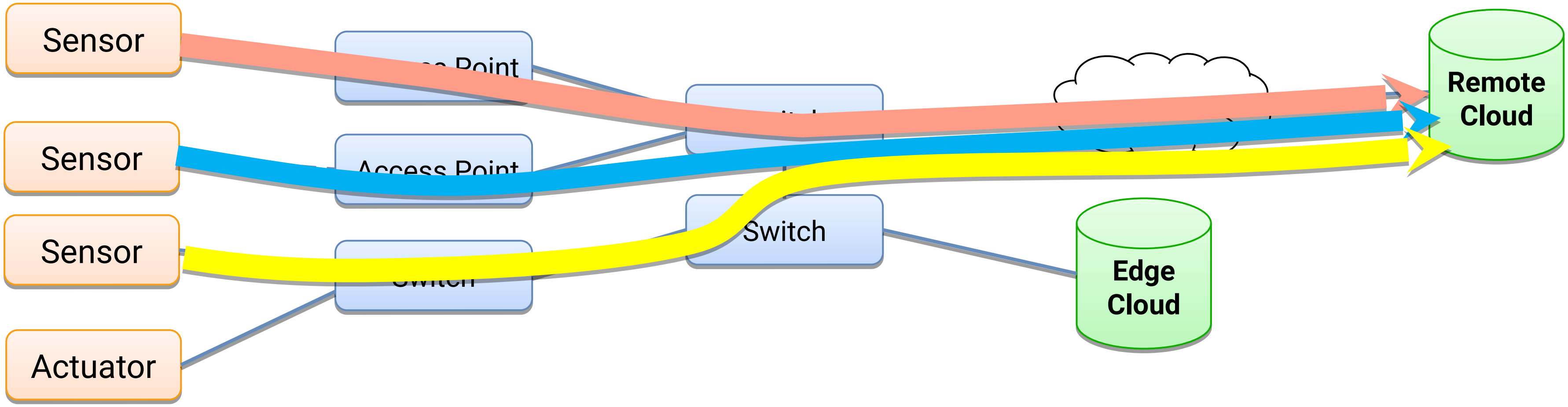
- **Mobile robot cooperation**



- **Control loops**
  - Single-digit millisecond latency
  - Multiple sensor sources
    - HD and infrared camera
    - Current draw of light arc
  - Actuators
    - Robot positioning
    - Light arc voltage

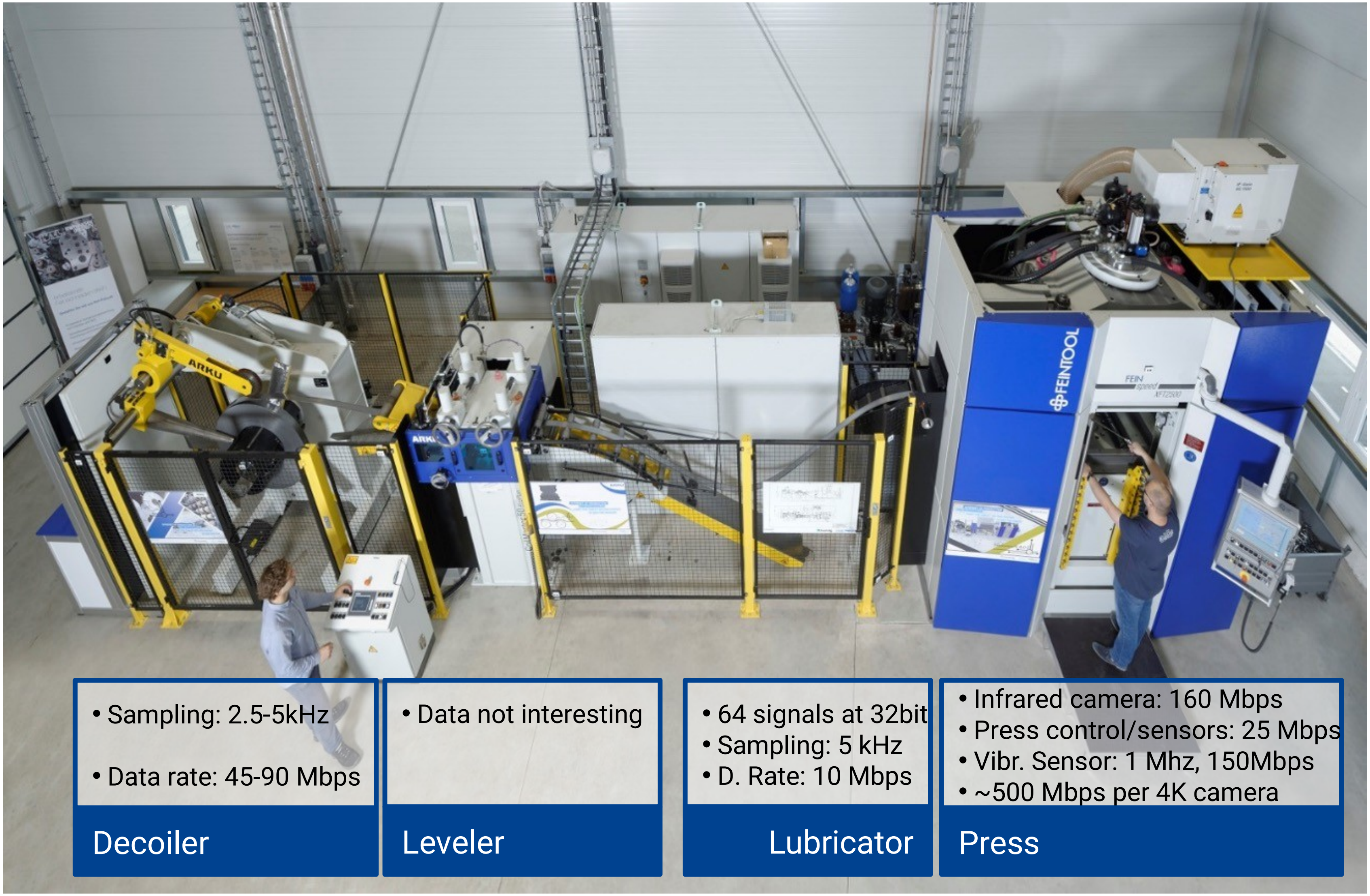
- **Control loops**
  - Positioning coordinated by many inputs
    - e.g. indoor coordinate system, camera, etc.
    - In-network coordinate transformation
  - Human in the loop detection (safety zone)
    - e.g. logical safety loop among cameras, lasers, Lidar
  - Robot interaction via multiple sensors
  - Augmented Reality …

Data Stream Processing



- **Collection and Analysis of Process Data**
  - Data-driven improvement of production and efficiency
    - Collect every data item the process and machines are emitting
    - Derive immediate feedback on process status and product quality
    - Realtime-feedback for production process
  - Problem: Data rate of produced process data

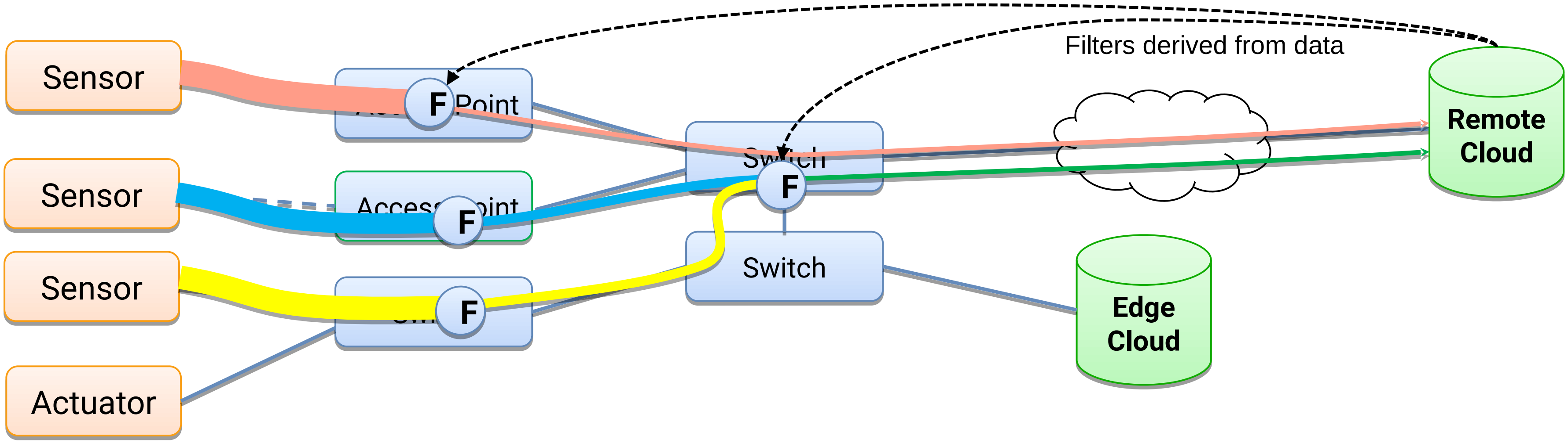
Real-world example: Fine blanking



Decoiler	Leveler	Lubricator	Press
• Sampling: 2.5-5kHz • Data rate: 45-90 Mbps	• Data not interesting	• 64 signals at 32bit • Sampling: 5 kHz • D. Rate: 10 Mbps	• Infrared camera: 160 Mbps • Press control/sensors: 25 Mbps • Vibr. Sensor: 1 Mhz, 150Mbps • ~500 Mbps per 4K camera

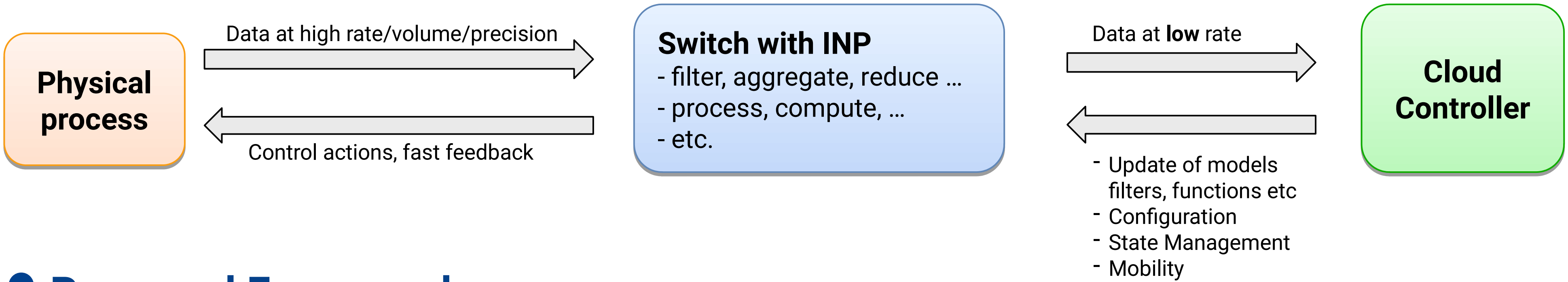
Glebke, Henze, Wehrle, Niemietz, Trauth, Mattfeld, Bergs: „A Case for Integrated Data Processing in Large-Scale Cyber-Physical Systems“, International Conference on System Sciences (HICSS), Wailea, HI, January 2019

Data Stream Processing at Line-Rate



- **Collection and Analysis of Process Data**
  - Data-driven improvement of production and efficiency
    - Collect every data item the process and machines are emitting
    - Derive immediate feedback on process status and product quality
    - Realtime-feedback for production process
  - Problem: Data rate of produced process data
- **Reduce/process the data as early as possible in the network**
  - Apply filtering, aggregation, compression, classification on the data path

Proposed Framework: INP/COIN for Industrial Networking



- **Proposed Framework**
  - Enable computation in the network elements (switches, access points, etc)
    - For simple control tasks
    - For filtering, aggregating, etc. data on the path to the cloud (at line rate)
    - For boosting data analysis in a data center (not discussed here)
  - Hierarchical placement of computational tasks
    - Simple and predictive computation in the **network**
      - Use to satisfy tight constraints (e.g. fast response)
    - Long-term computation, state management and coordination in the **cloud**
      - Use for complex tasks

ToDos for Academia and/or IETF for Enabling COIN for Industrial Networks

- **What do we need?**
  - More computational capabilities
    - some math operations would be nice and a bit of state
    - simple computations are ok, must not be Turing complete ☺
    - at line-rate or at least predictable execution times
  - Configuration, monitoring, and management
    - Interface: cloud ⟷ switch , northbound ⟷ southbound
    - “OpenFlow” for INP/COIN
    - Management and configuration of INP/COIN elements
    - State management
    - Mobility of processing elements
  - Transport protocol issues
    - Breaking of end-to-end principle
    - Encrypted data?

Implementing LQR control in a network element

- **You may wakeup now** ≡

- **First shot: Implement it in (e)BPF**
  - Can be deployed on Linux hardware (XDP)
  - Runs on Netronome SmartNICs
  - Is basically writing C code with some limitations
  - Pretty easily done

- **Second shot: Can we do it in $P4_{16}$?**

P4 implementation

- **P4 (also BPF) is not made for doing math**
  - Only integer support
    - Support for bit-depth, padding, and operands platform specific
  - Control problems typically specified over real numbers
    - We assume all numbers to be scaled by a fix-point → Integer
    - Computations need to account for this fix-point
      - Multiplications of two fix-points must be divided by the fix-points
      - Can easily overflow bit-depth
  - No divisions on signed integers

- **Control matrix stored as a table**
  - Lookup by flow 4-tuple

Controller computation in P4

- **Given a 1x4 matrix (K), and a 1x4 sensor reading (u)**
  - Compute $-K^T \cdot u$

- **In P4 this becomes**
- **In P4 this becomes**

```
myctrl = (((int<64>)-ctrl.k0 * (int<64>)hdr.sensor_data.data0) +
      ((int<64>)-ctrl.k1 * (int<64>)hdr.sensor_data.data1) +
      ((int<64>)-ctrl.k2 * (int<64>)hdr.sensor_data.data2) +
      ((int<64>)-ctrl.k3 * (int<64>)hdr.sensor_data.data3));

 if (myctrl < 0) {
    hdr.actuator_data.data = (int<32>)(
      ((int<64>) ((bit<64>)(myctrl * -1) / SCALINGFACTOR))
       * -1);
 }else {
    hdr.actuator_data.data = (int<32>)(
      (int<64>)((bit<64>)(myctrl) / SCALINGFACTOR)
      );
 }
```

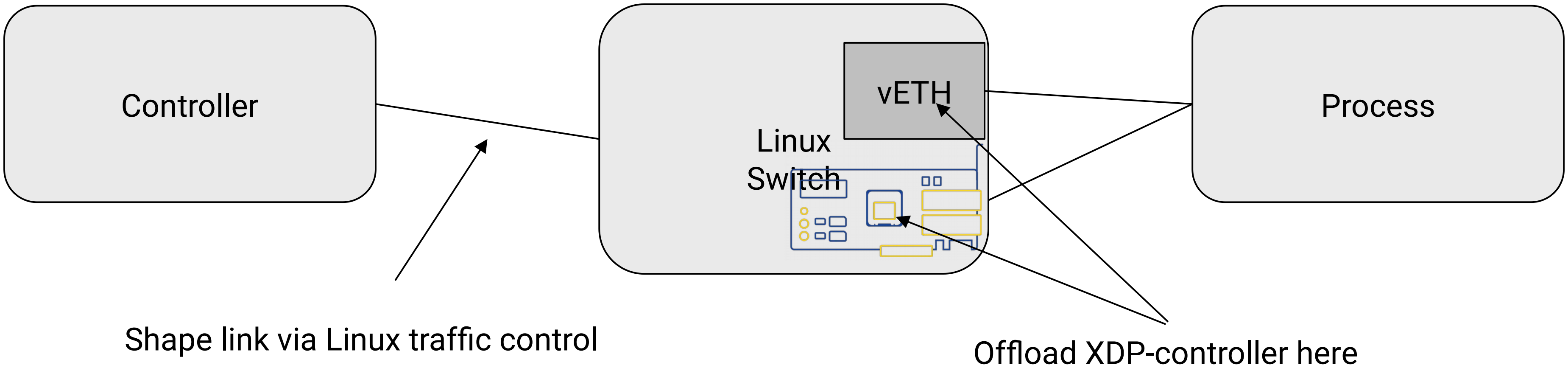
- **Ugly but does the job**
- **Ugly but does the job**

Jan Rüth, René Glebke, Klaus Wehrle, Vedad Causevic, Sandra Hirche

Evaluation

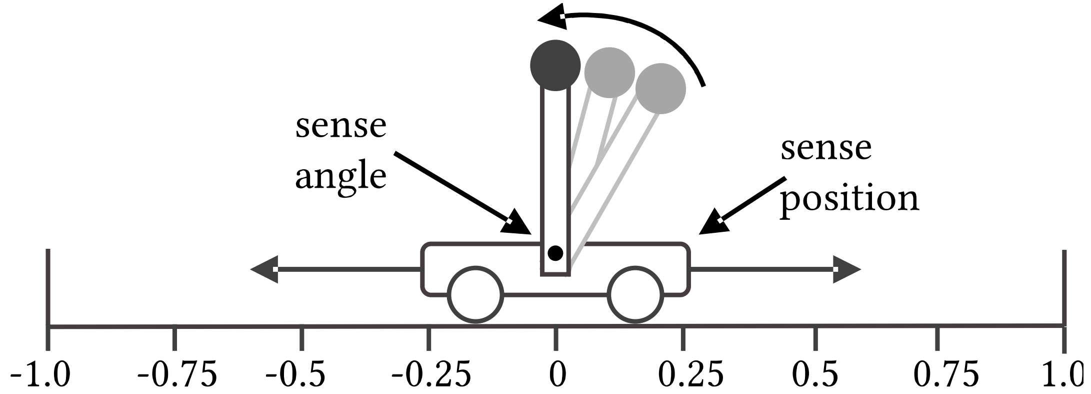
- **We compile the P4 switch description to BPF**
  - Using P4C-XDP
    - https://github.com/vmware/p4c-xdp

- **We evaluate in a testbed**



  - Can be emulated via mininet if desired

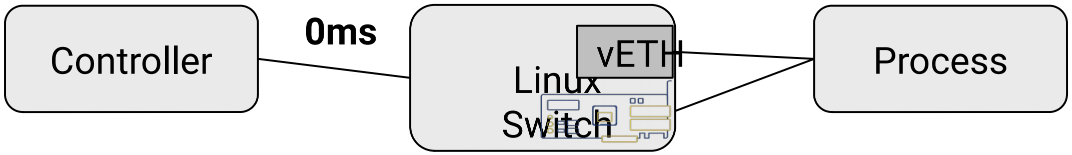
- **We use a real-time simulation of an inverted pendulum**
  - Other systems possible, controller is independent of the system

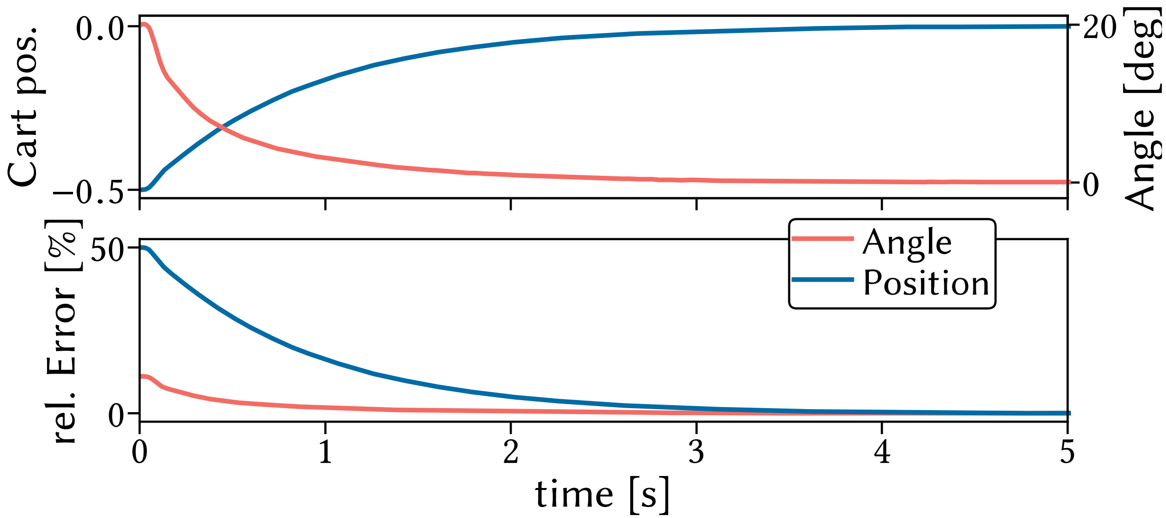


- **Keep the pendulum in an up right position in the center**
  - Like balancing a pen on your palm
- **Sensors acquire**
  - Position, change in position, angle, change in angle
- **Actuator**
  - Controller can move the cart

- **We measure the Quality-of-Control**
  - How fast can we move the cart to the center
  - How stable is the rod around the upright position

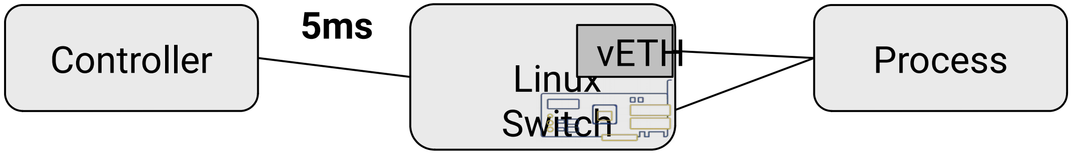
- **Without any delay**
  - Smooth transition
  - ~4 secs

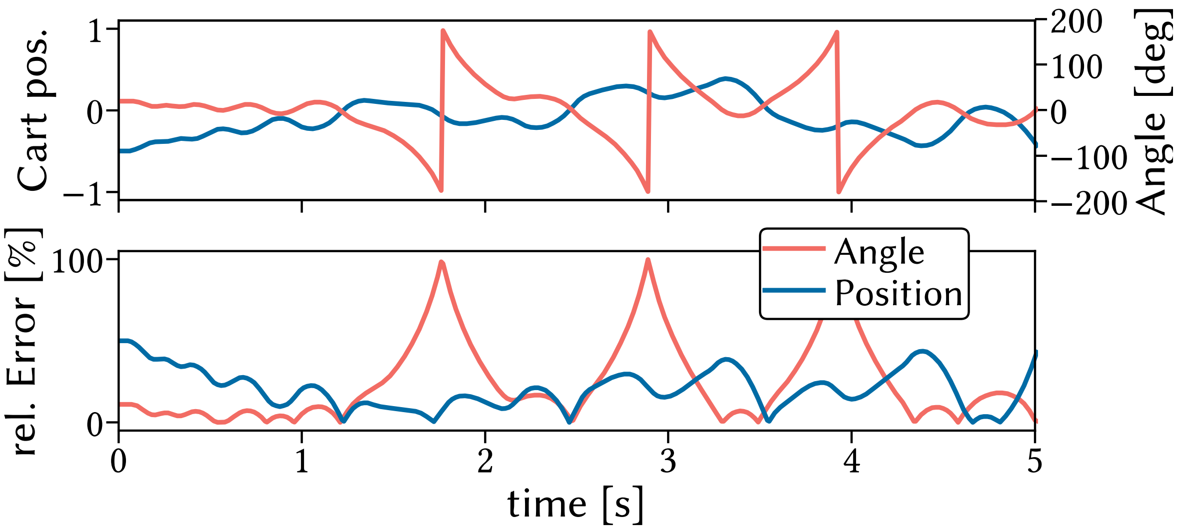


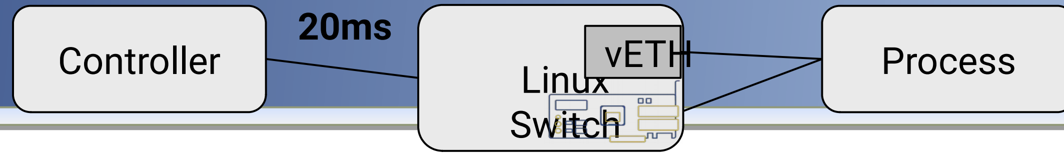


Jan Rüth, René Glebke, Klaus Wehrle, Vedad Causevic, Sandra Hirche

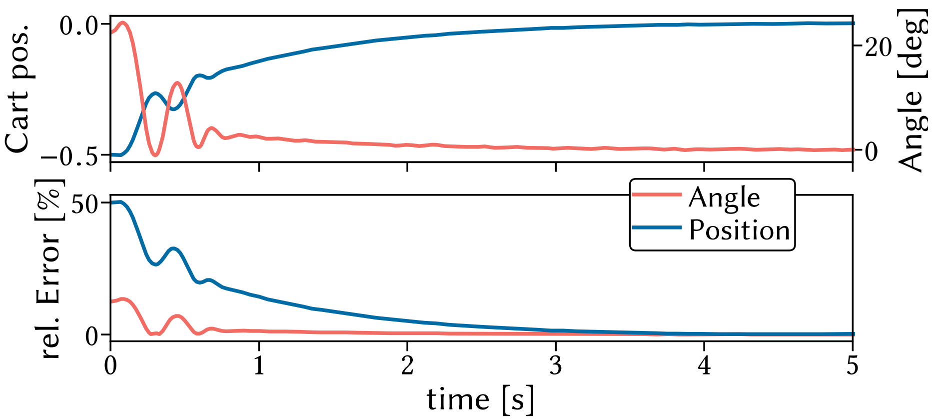
- **Add 5ms of delay**
  - Does not stabilize
  - Wobbles back and forth
  - Rotates 360°

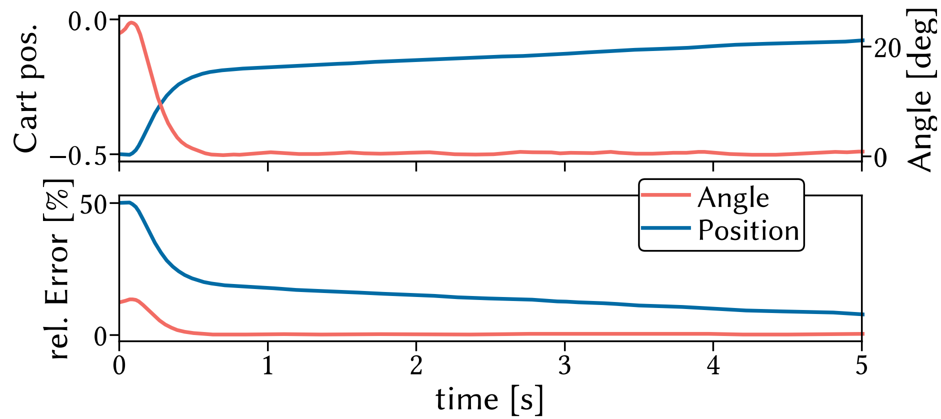






- **Modified controller that accounts for delay**
  - We add 20ms delay



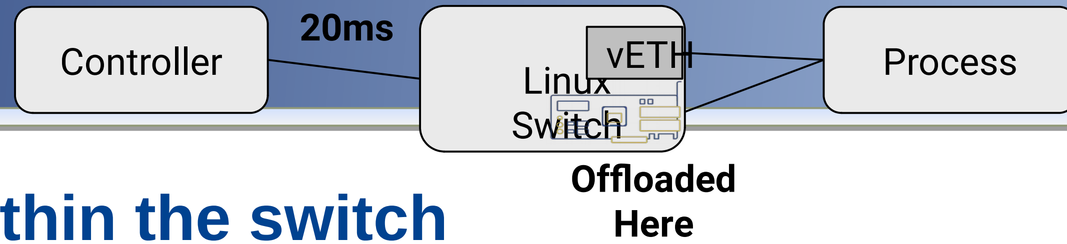


  - Heavy back end forth at the start
  - Stabilizes with slight wobbling
  - 4 sec

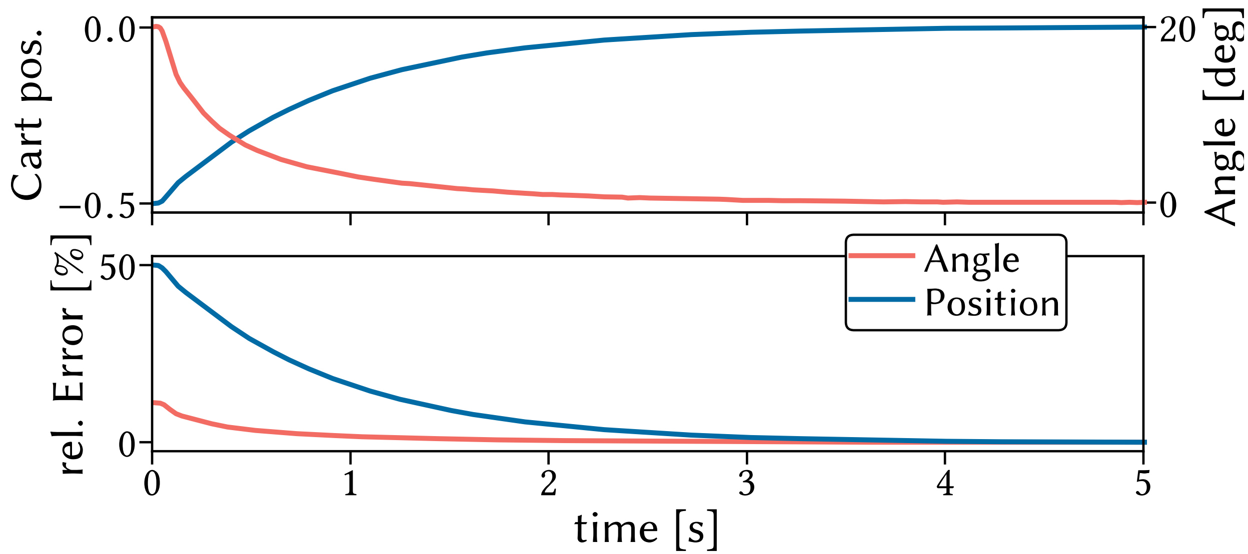
  - No more back end forth
  - Eventually stabilizes
  - >5 sec to stabilize

- **Both not optimal**

Evaluation



- **Activate P4-controller within the switch**
  - Intercepts packets on behalf of the controller



  - No wobbling
  - Stabilizes within 4 secs
  - As good as the real controller

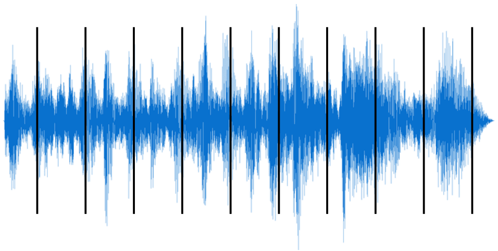
Future Challenges

- **Would be desirable to change tables from data path**
  - Accounting for delay bloats the matrices
  - Past computations need to be saved
  - Possible in BPF but (currently) not from the SmartNIC

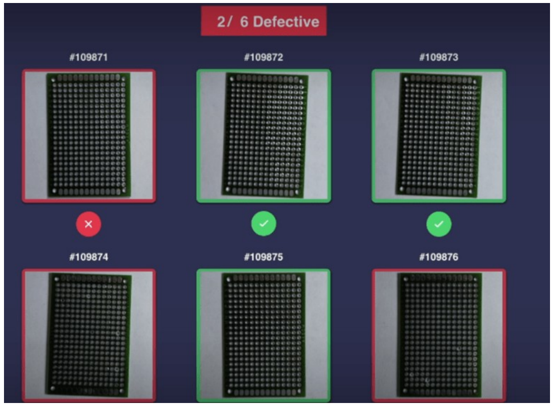
- **Networked control good for collaborative control**
  - Requires sharing recent computations with other controllers
  - Is it enough to do this from the control plane?
    - Better generate new packet from data path (PSA?)

- **What about other control problems?**

- **Audio processing heavily used**
  - Data spread over multiple packets
  - Detect vibrations
  - Must equipment be maintained?



- **Visual processing also heavily used**
  - Many packets
  - Computer vision can become heavy



Conclusion

Is it possible to implement a controller
in a network element?

With typical data plane languages such as P4?

- **Short answer: Yes, you can!**
  - Math is a hassle in P4
  - Advanced problems currently lack functionality

- **Which other tasks could be offloaded to the network?**