Resilience via Blackbox Self-Piloting Plants

Michel Barbeau[1], Joaquin Garcia-Alfaro[2,5,1], Christian Lübben[3,6], Marc-Oliver Pahl[4,5,6,1] and Lars Wüstrich[3,6]

[1]*School of Computer Science, Carleton University, 1125 Colonel By Drive, Ottawa, Ontario,Canada K1S 5B6*

[2]*Télécom SudParis, SAMOVAR, Institut Polytechnique de Paris, 91120, Palaiseau, France*

[3]*Institut für Informatik der Technischen Universität München Lehrstuhl I8 Boltzmannstr. 3 85748 Garching bei München - Germany*

[4]*IMT Atlantique, Rennes, France IRISA, UMR IRISA CNRS 6074*

[5]*IMT Industrial chair Cybersecurity for Critical Networked Infrastructures (CyberCNI.fr)*

[6]*TUM Smart Space Orchestration team / Chair for Network Architectures and Services (s2o.net.in.tum.de)*

AIRBUS

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

TELECOM Paris

BNP PARIBAS
La banque d'un monde qui change

Région Bretagne

EDF

SNCF

PÔLE D'EXCELLENCE CYBER

TELECOM SudParis

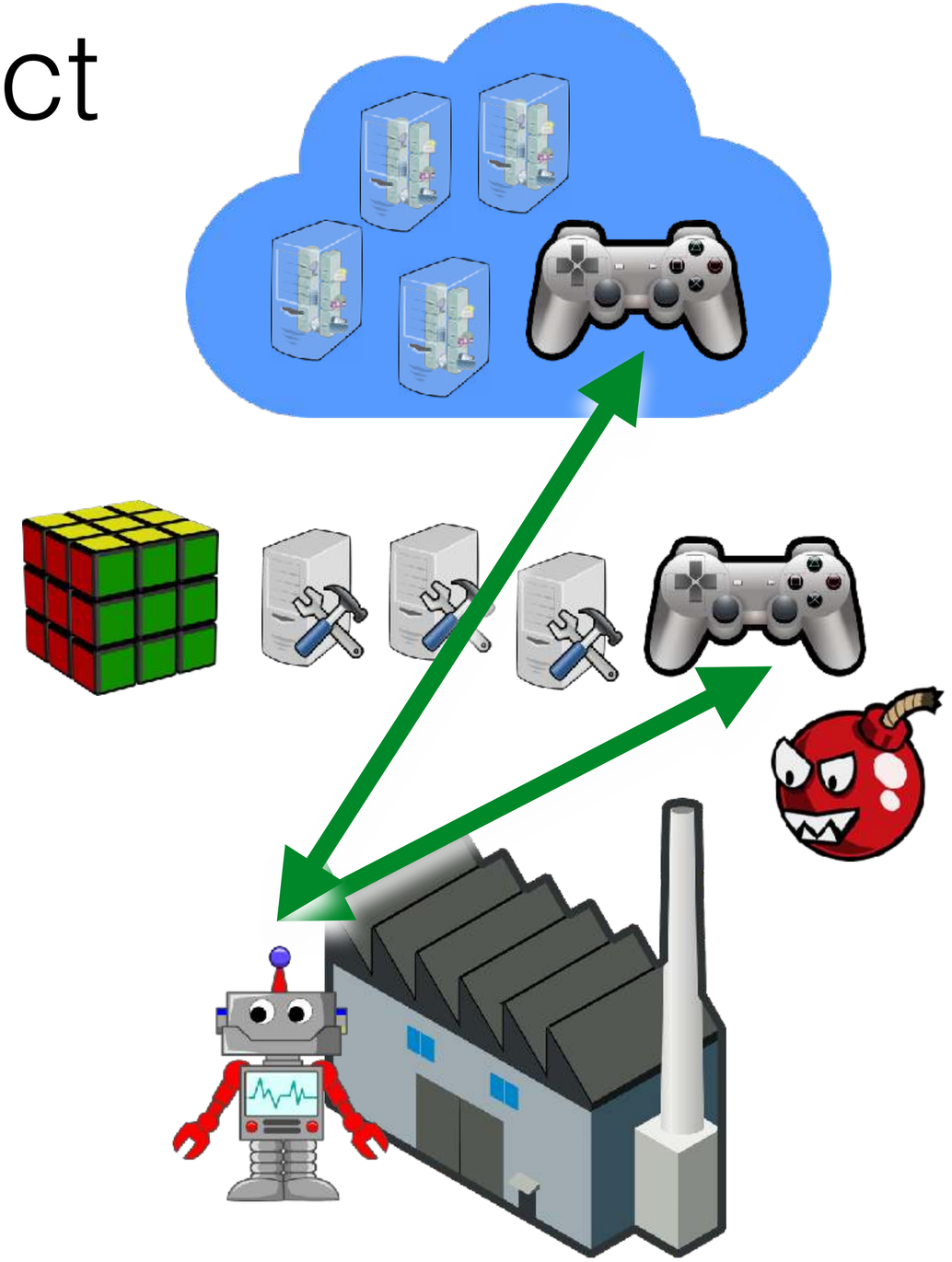
CHAIRE CYBER CNI
Sécurité des infrastructures critiques

cyberCNI.fr/

Abstract

Distributed control is a reality of today's **industrial automation** and systems. Parts of a system are **on-site**, and other elements are on the **edge** of the **cloud**. The overall system-functioning **relies on the reliable operation of local and remote components**.

However, all system parts can be attacked. Typically, local entities of a cyber-physical system, such as robot arms or conveyor belts, get affected by cyber attacks. However, **attacking the control and monitoring channels between a plant and its remote controller is attractive**, too. There is a diversity of attacks, such as **manipulating a plant's input signals**, **controller logic**, and **output signals**.

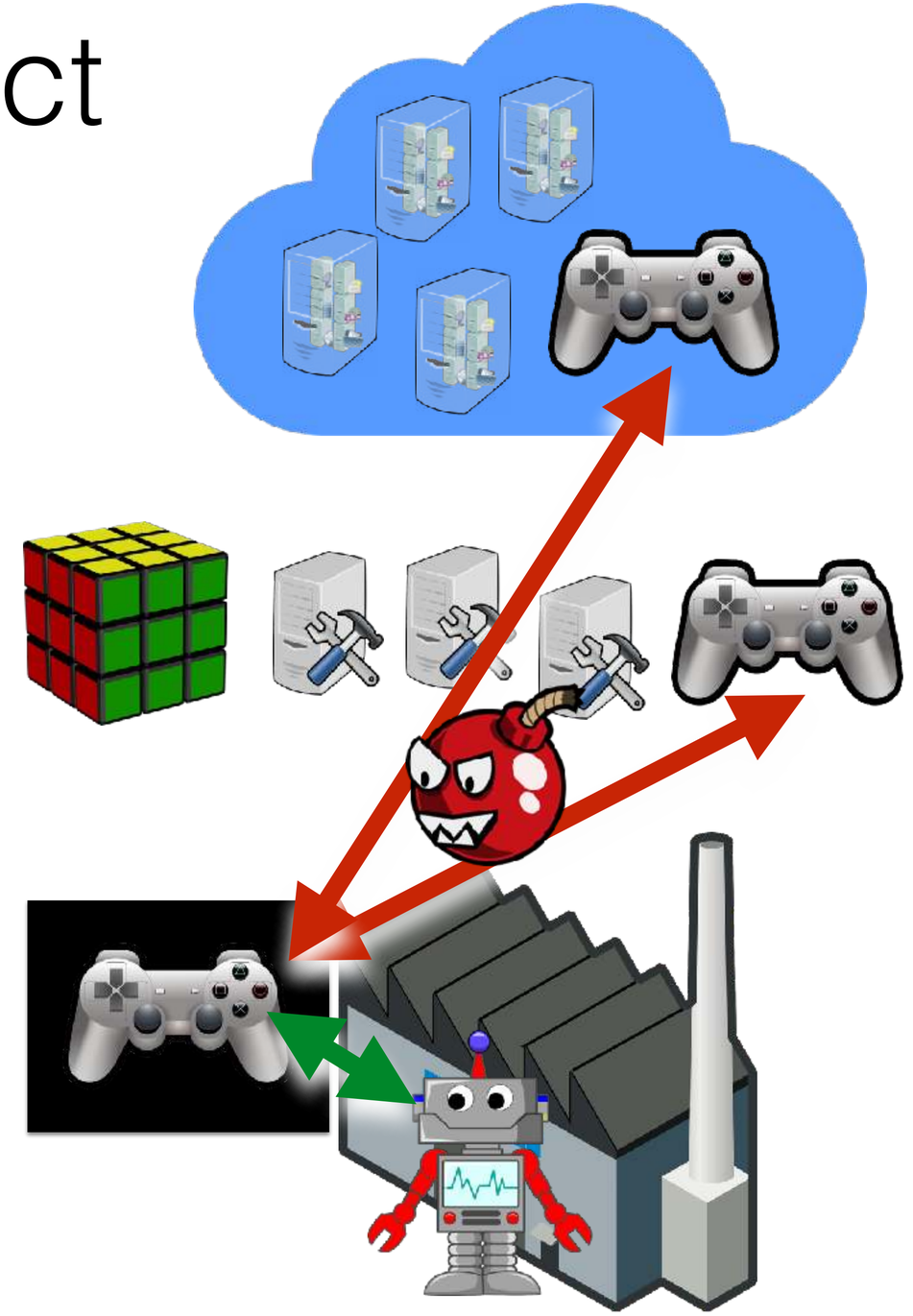


Abstract

To detect and mitigate the impact of such various attacks and to make a plant more resilient, **we introduce a self-learning controller proxy in the plant's communication channel to the controller**.

It acts as a **local trust anchor to the commands received from a remote controller**. It does **black box self-learning** of the controller algorithms and **audits** its operations. Once an attack is detected, the plant pivots into **self-piloting mode**.

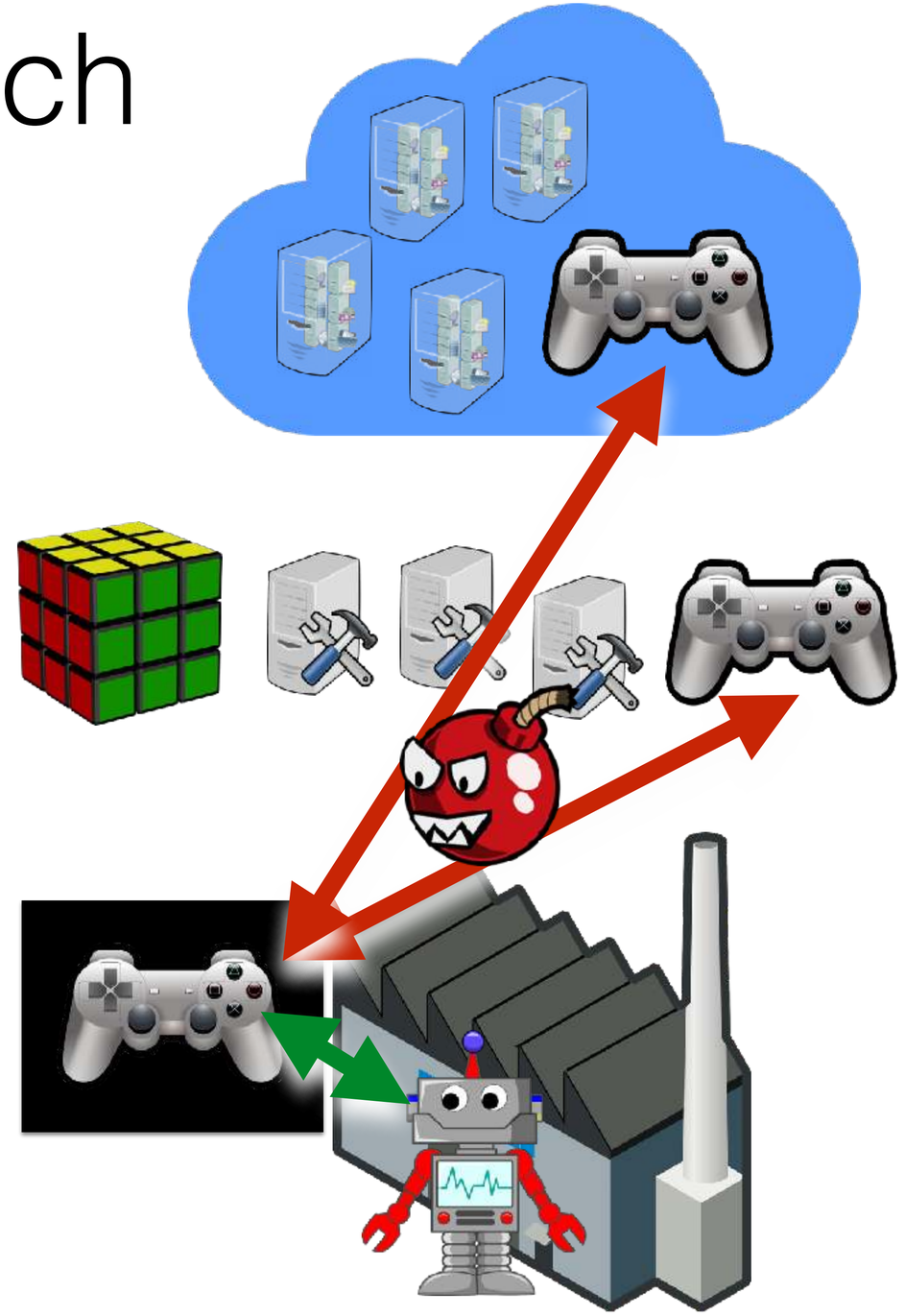
We investigate **design alternatives for the controller proxy**. We **evaluate how complex the control algorithms can be** to enable self-piloting resilience.



Approach

Learning the parameters of the local model by monitoring the pass-by traffic.

$controllerInput_t = \langle i_1, \ldots, i_n \rangle$



Related work

- [11] D. E. Quevedo, E. I. Silva, G. C. Goodwin, **Packetized predictive control over erasure channels,** in: 2007 American Control Conference, 2007, pp. 1003–1008.

- [12] G. Franzè, F. Tedesco, D. Famularo, **Model predictive control for constrained networked systems subject to data losses**, Automatica 54 (2015) 272 – 278. URL: http://www.sciencedirect.com/science/article/pii/S0005109815000710. doi:https://doi.org/10.1016/j.automatica.2015.02.018.

- [13] G. Franzè, F. Tedesco, W. Lucia, **Resilient control for cyber-physical systems subject to replay attacks**, IEEE Control Systems Letters 3 (2019) 984–989.

- Similar approaches (predicting input/output) but novel application area (industrial controller)

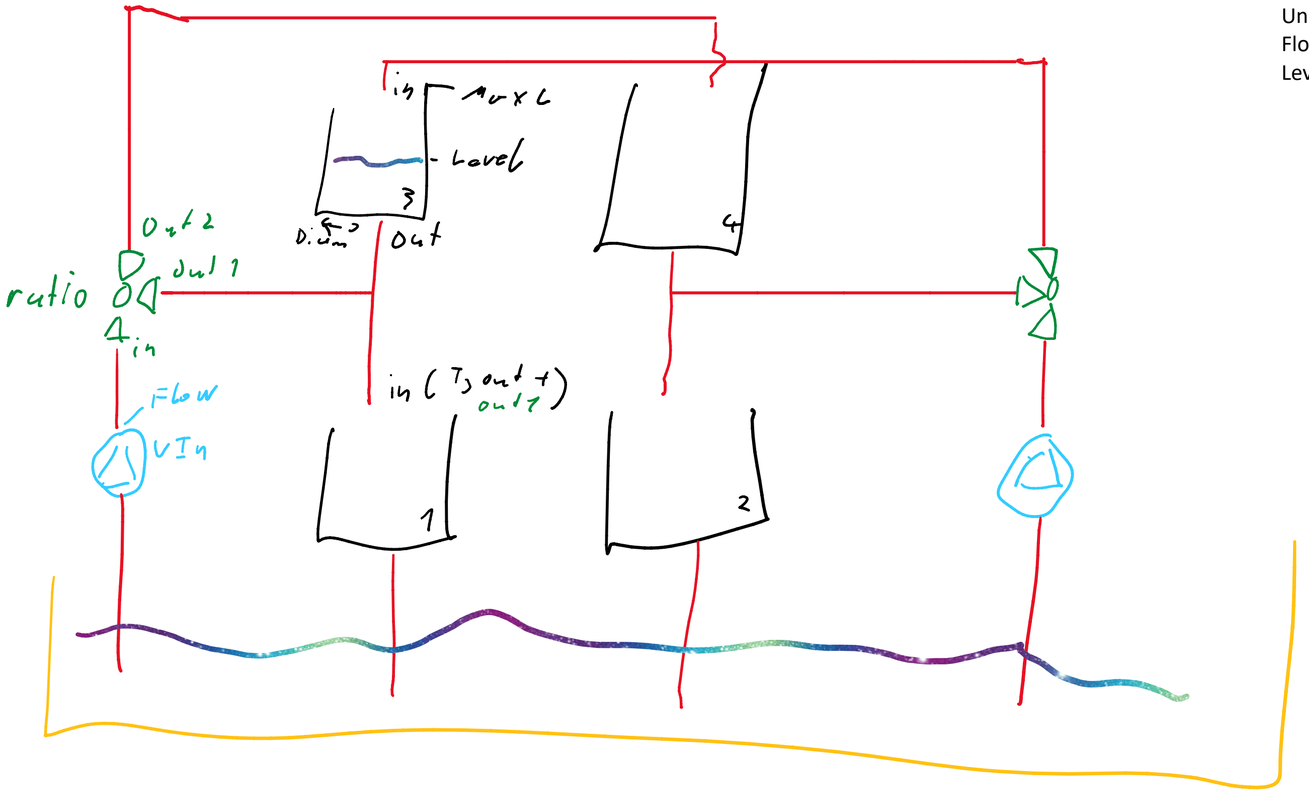
System Model: Water Tanks

Classes

Tank

Pump

Valve



Units: cm, sec, percent
Flow: cm^3/sec
Levels: cm

State
Actuator Input

$$x_{i+1} = f(x_i, u_i) \ (1)$$

$$y_i = g(x_i) \ (2)$$

Sensor Output

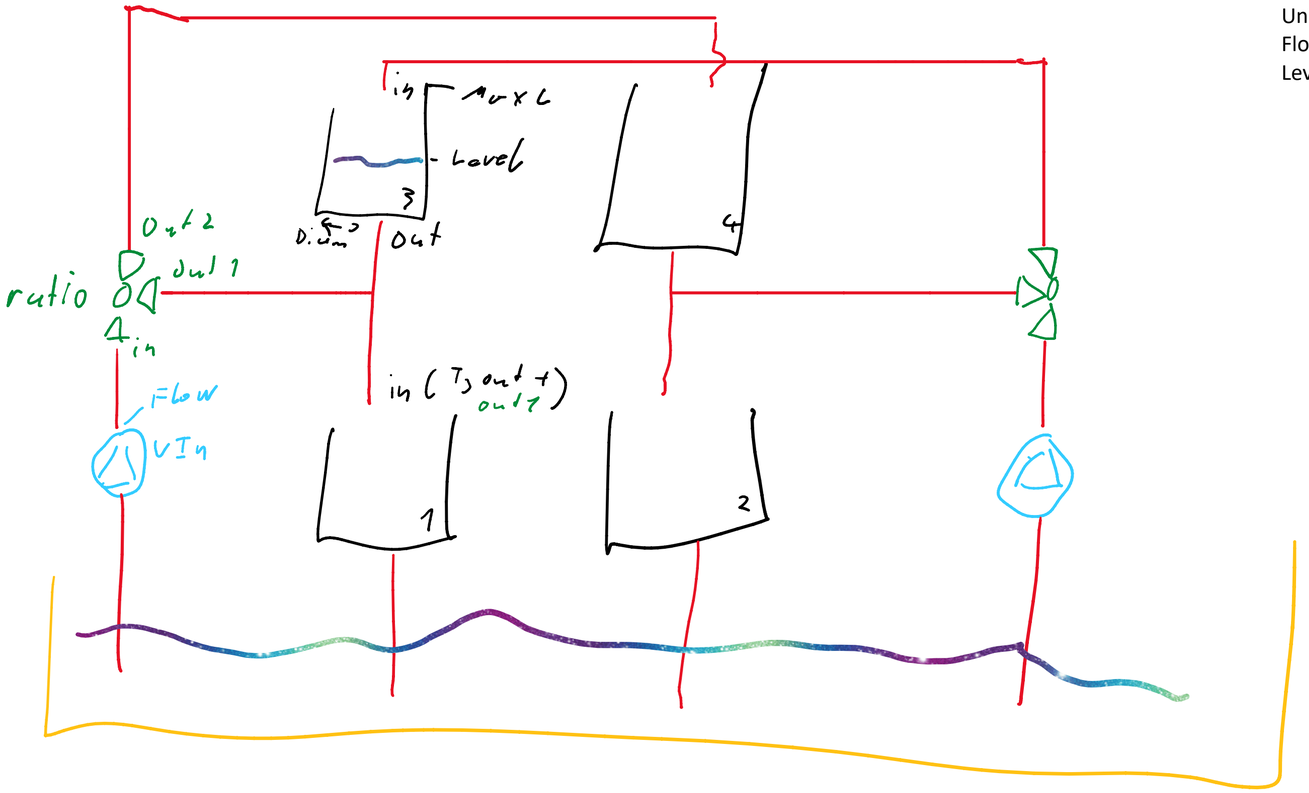
System Model: Water Tanks

Classes

Tank

Pump

Valve



Units: cm, sec, percent
Flow: cm^3/sec
Levels: cm

$$\frac{dh(t)}{dt} = \frac{F(t) - a\sqrt{h(t)}}{\alpha}$$

$$\begin{aligned} x_{i+1} &= f(x_i, u_i) = x_i + \frac{u_i - a\sqrt{x_i}}{\alpha} \\ y_i &= g(x_i) = x_i \end{aligned}$$

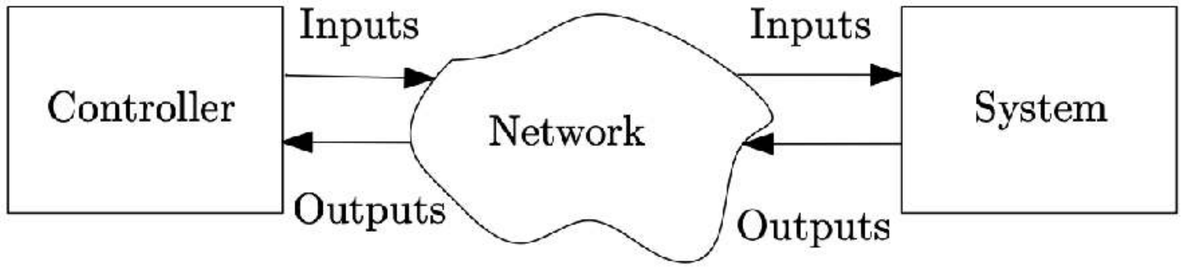
State
Actuator Input

$$x_{i+1} = f(x_i, u_i) \ (1)$$

Sensor Output

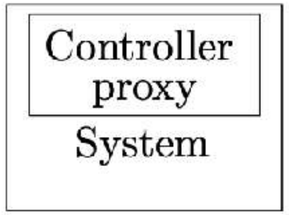
$$y_i = g(x_i) \ (2)$$

Attack Model

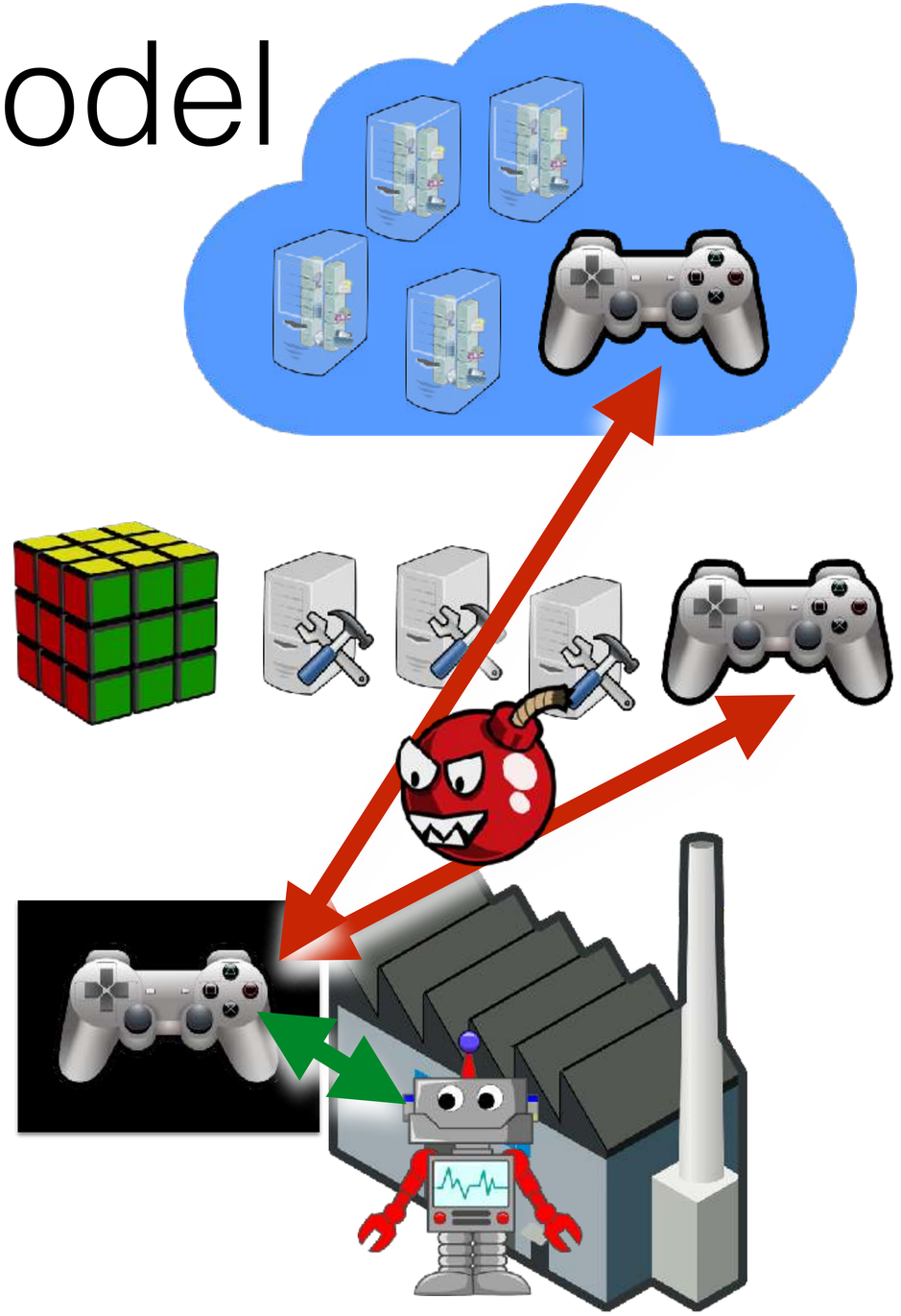


(a) Architecture of a Networked-Control System.

(b) Attack model.



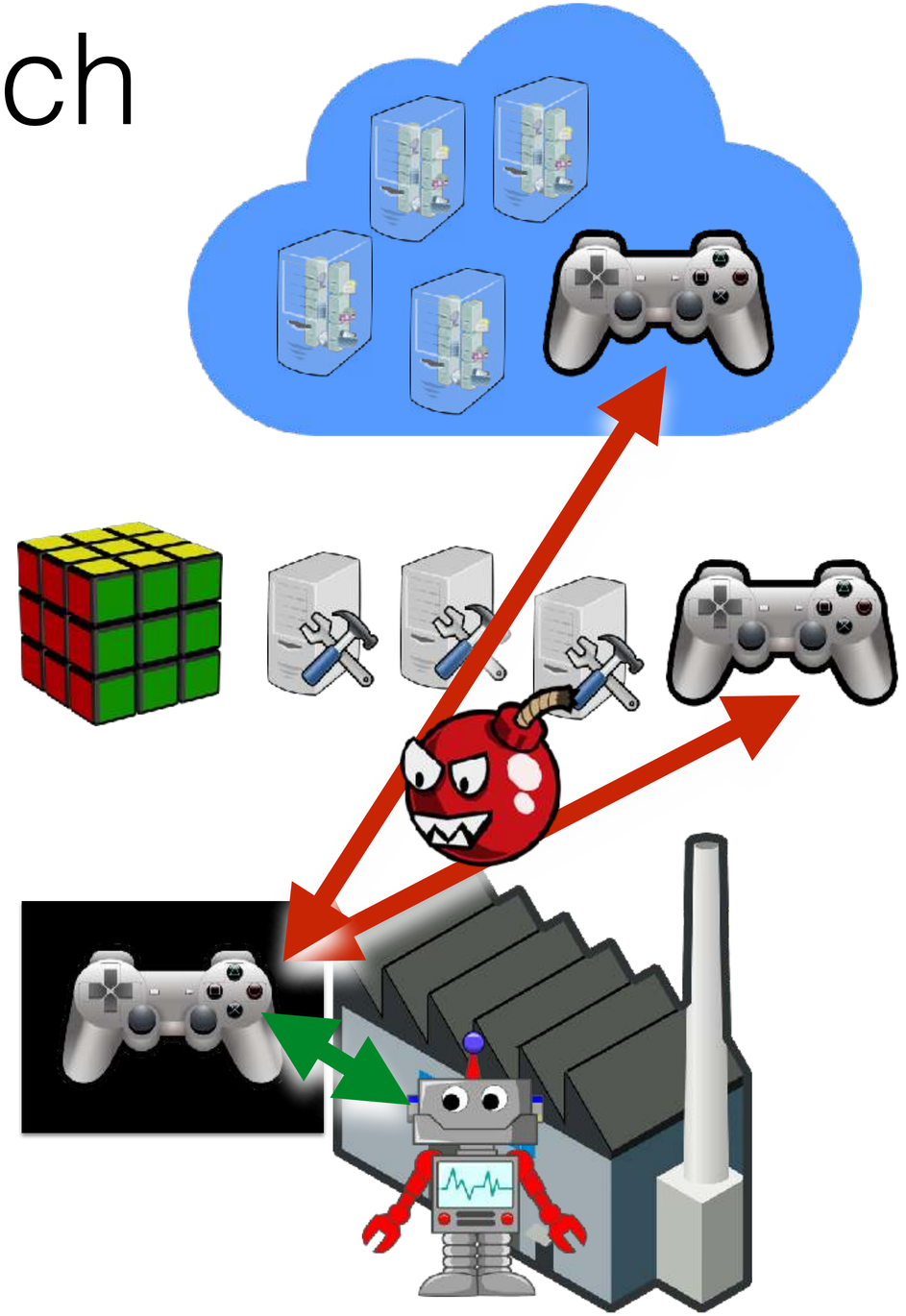
(c) Resilience model.



Approach

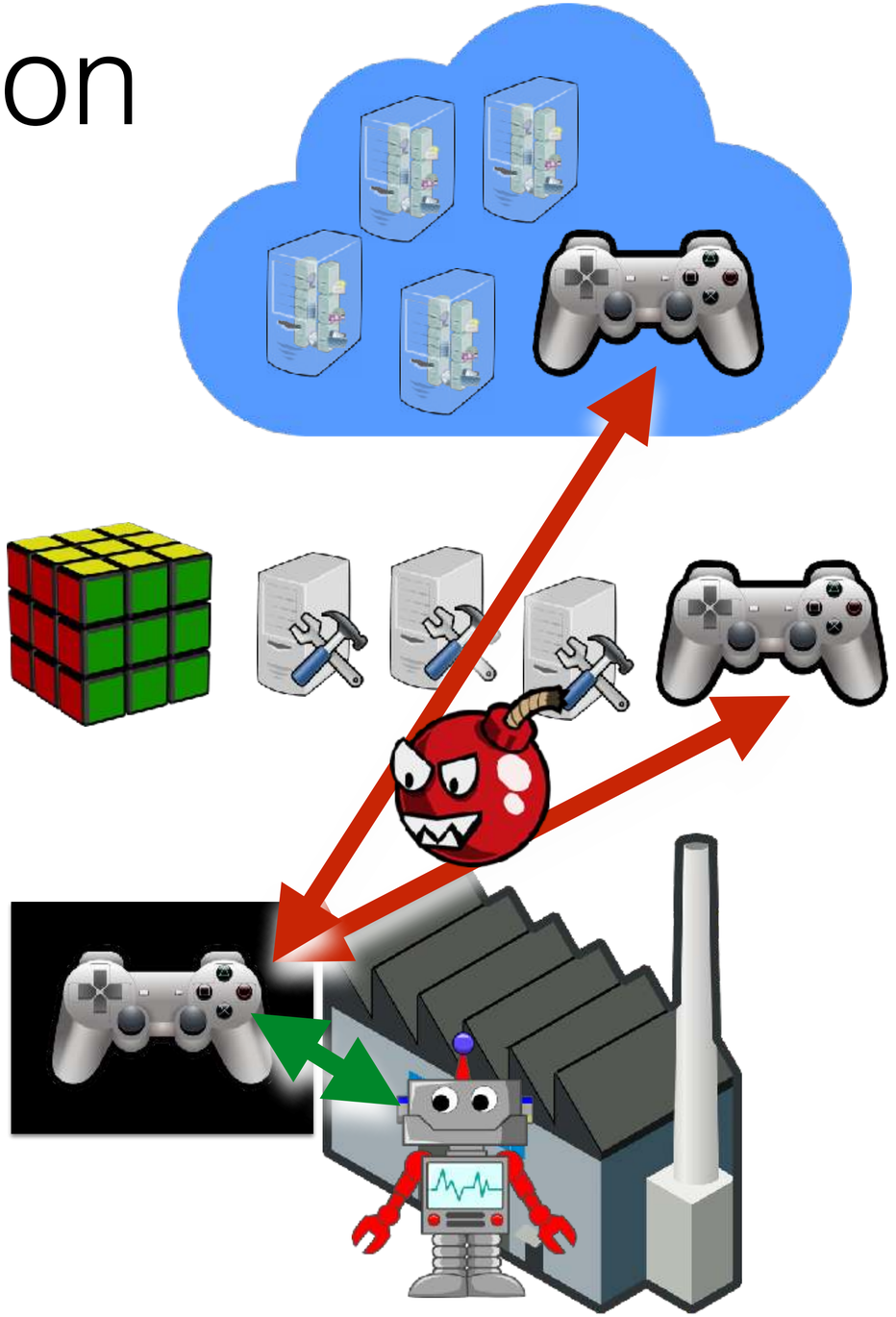
Imitation Learning

Behavioural Cloning + DAGGER



Evaluation

Implementation as emulation with the Virtual State Layer (VSL) rapid prototyping framework.



Evaluation

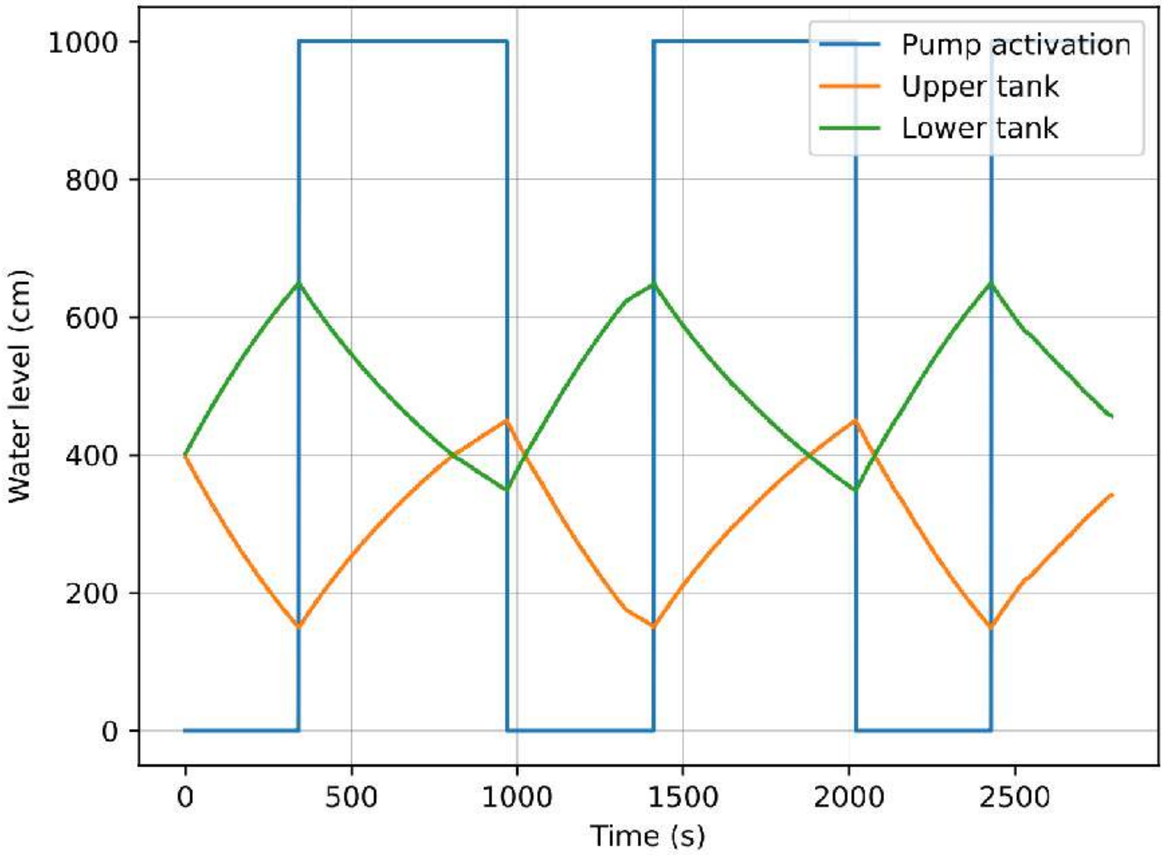
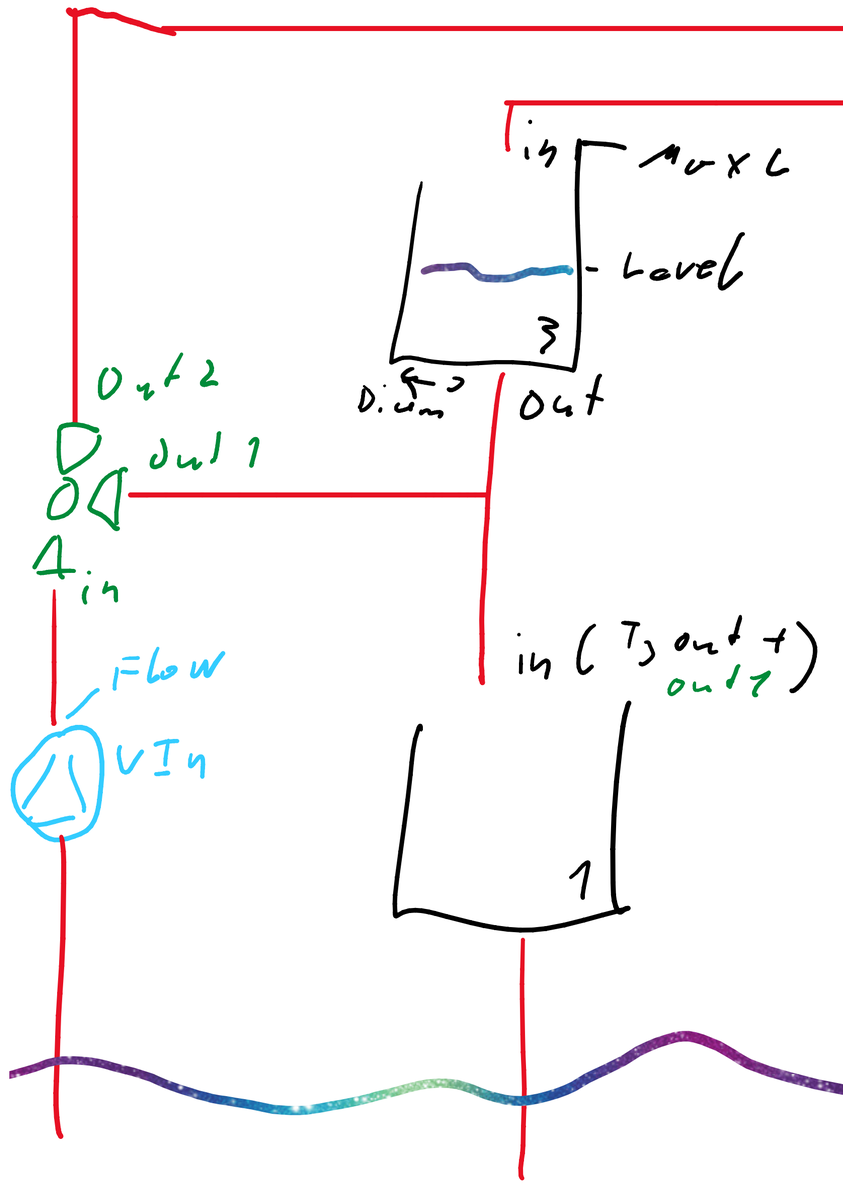


Figure 3: Cyclic behavior of the plant over time. The horizontal axis represents time (seconds). The vertical axis corresponds to the pump state (on or off) or tank water level (cm). The blue line tracks the pump's on/off (low/high) state. The green (orange) line plots the water level in the lower (upper) tank.



Evaluation

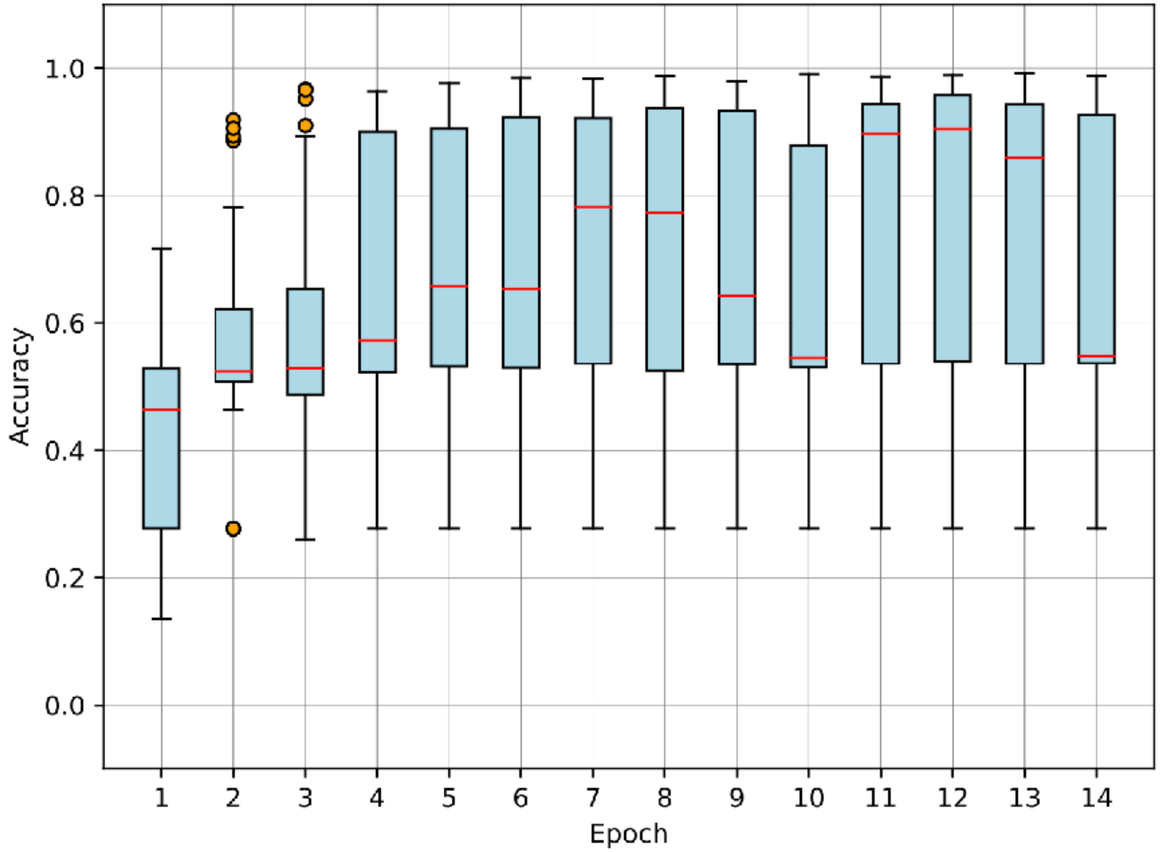
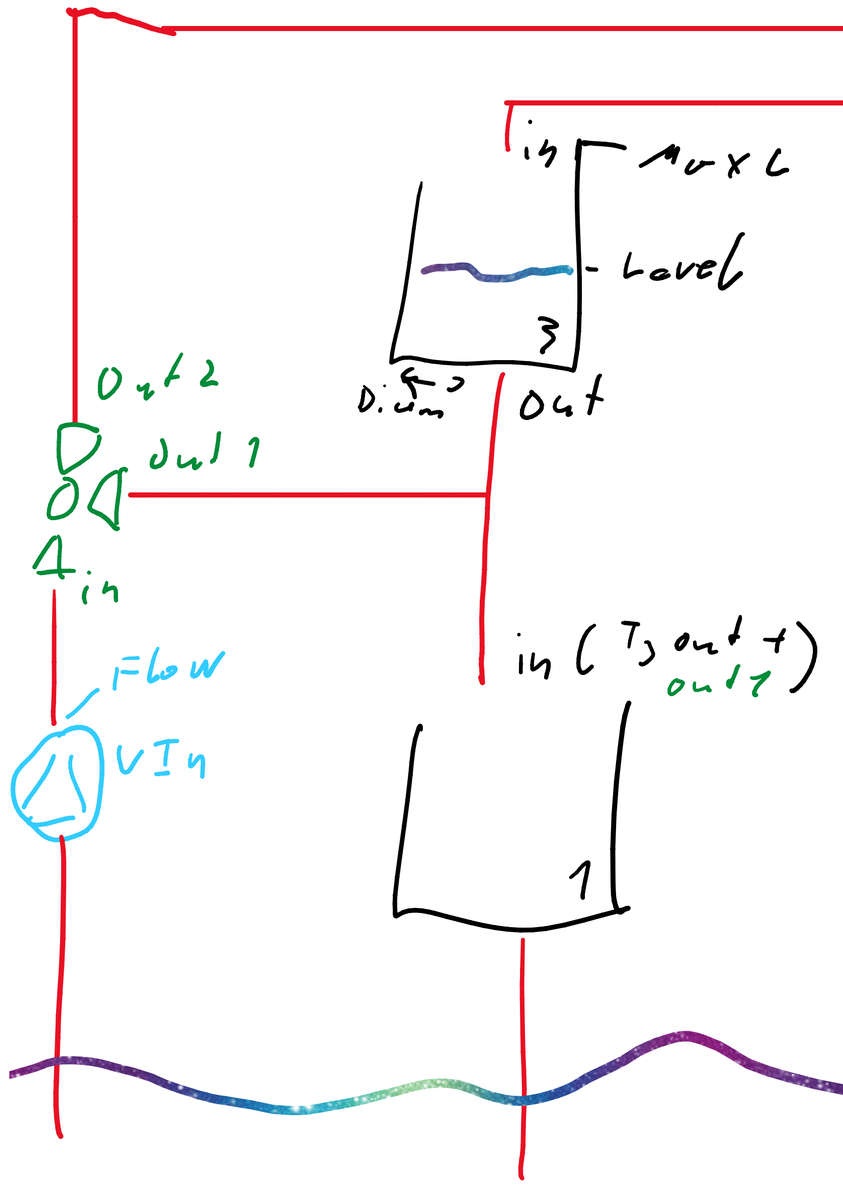


Figure 4: Accuracy vs. number of epochs. The horizontal axis corresponds to the number of epochs used for training, one to 14. The vertical axis represents the accuracy (on a scale of zero to one). We use statistical boxplots. For every box, the central (red) mark indicates the median. The bottom and top edges of the box indicate the 25th and 75th percentiles. The whiskers extend to the most extreme data points not considering outliers. Outliers are represented by the orange circles.



Evaluation

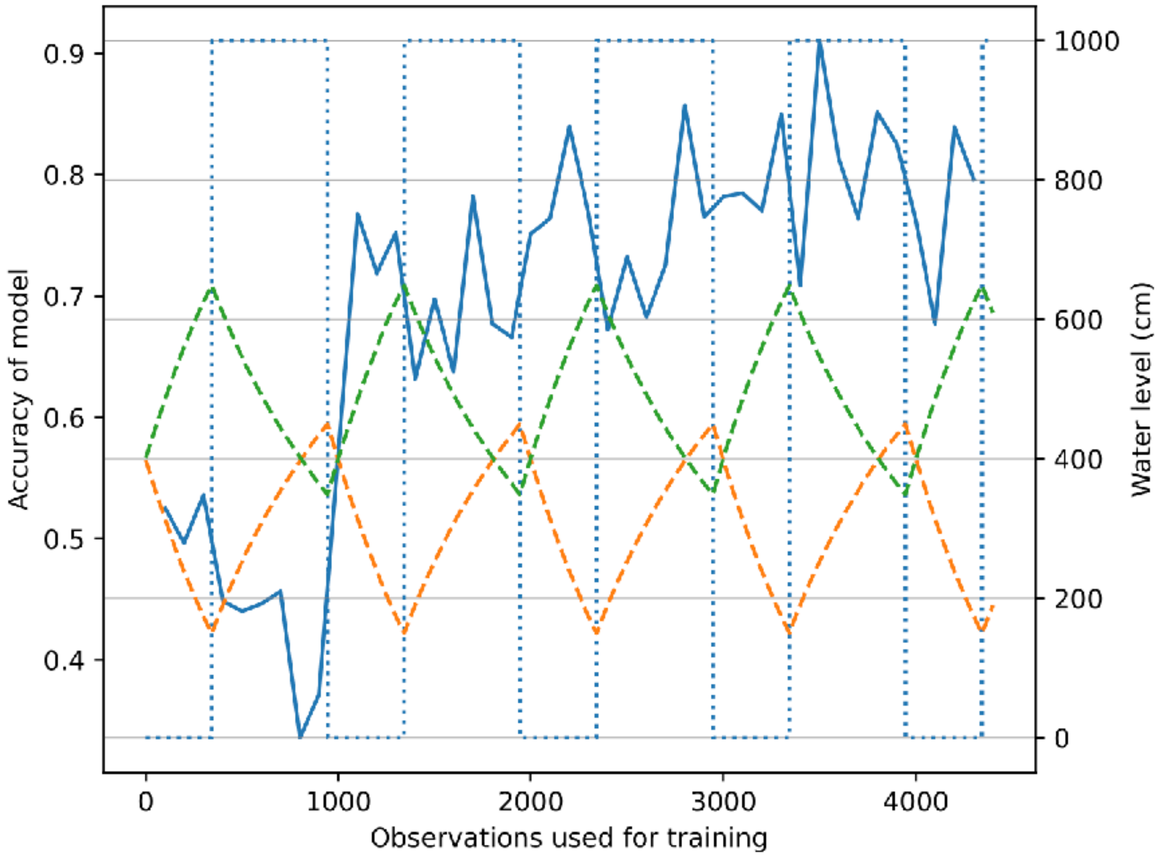
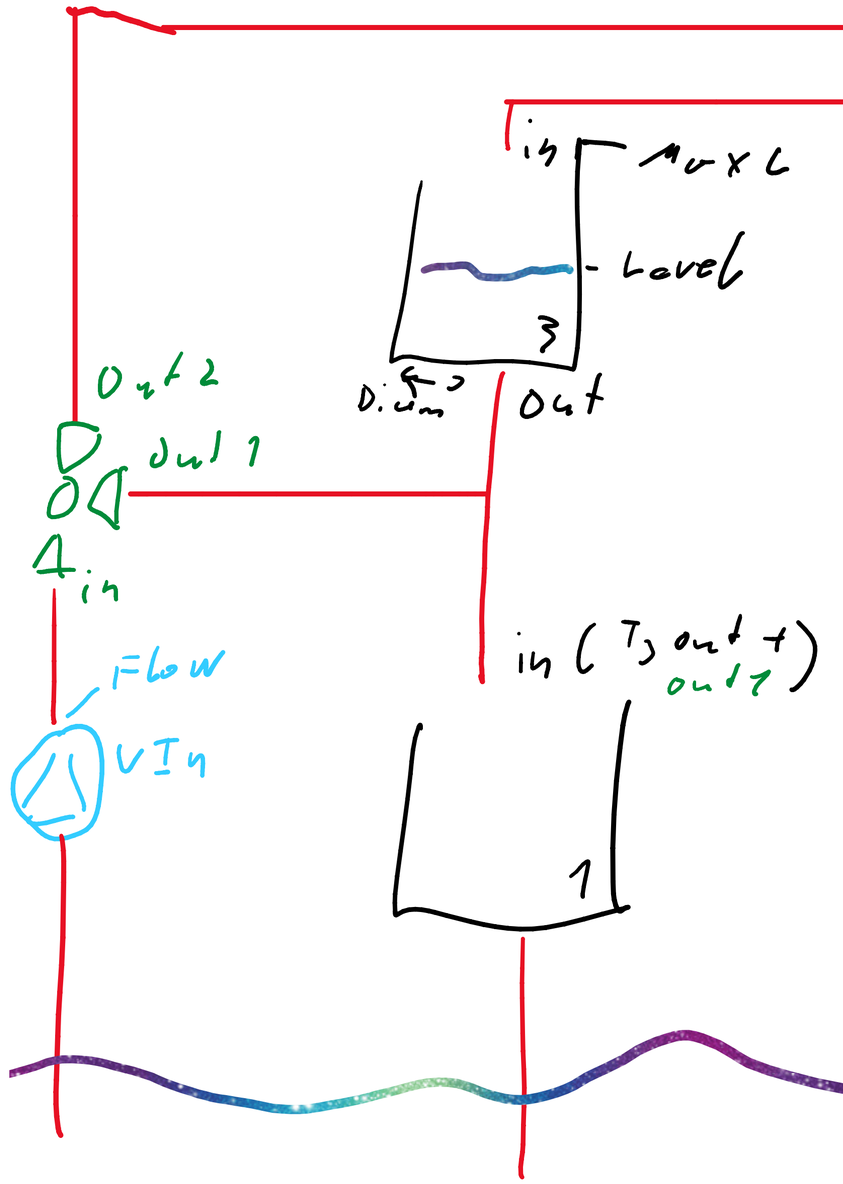
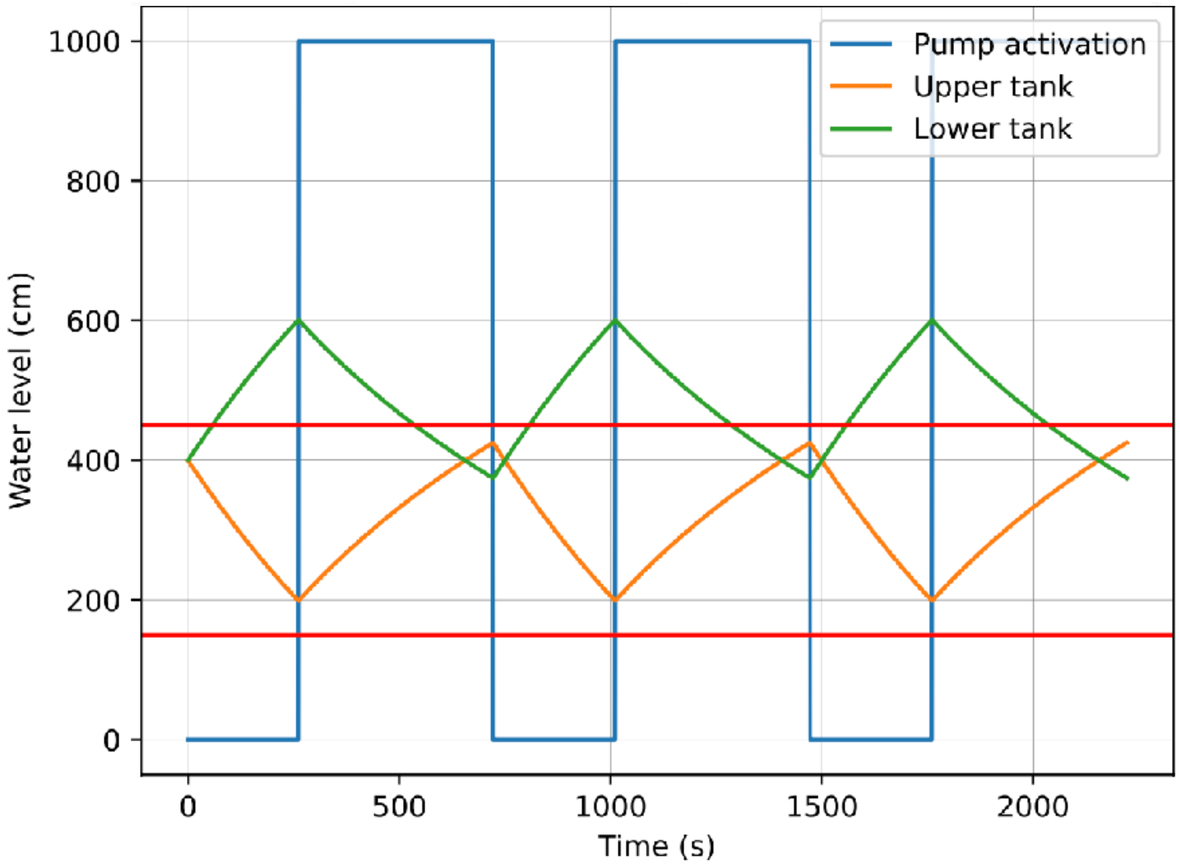


Figure 5: Performance vs. training instances. The horizontal axis corresponds to the number of observations used for training. It ranges from 100 to 4300 observations. The left vertical axis indicates the accuracy of the model together with the blue solid line. The blue dotted line tracks the pump's on/off (low/high) state. The right vertical axis corresponds to the tank water level (cm). The green (orange) line plots the water level in the lower (upper) tank.



Evaluation



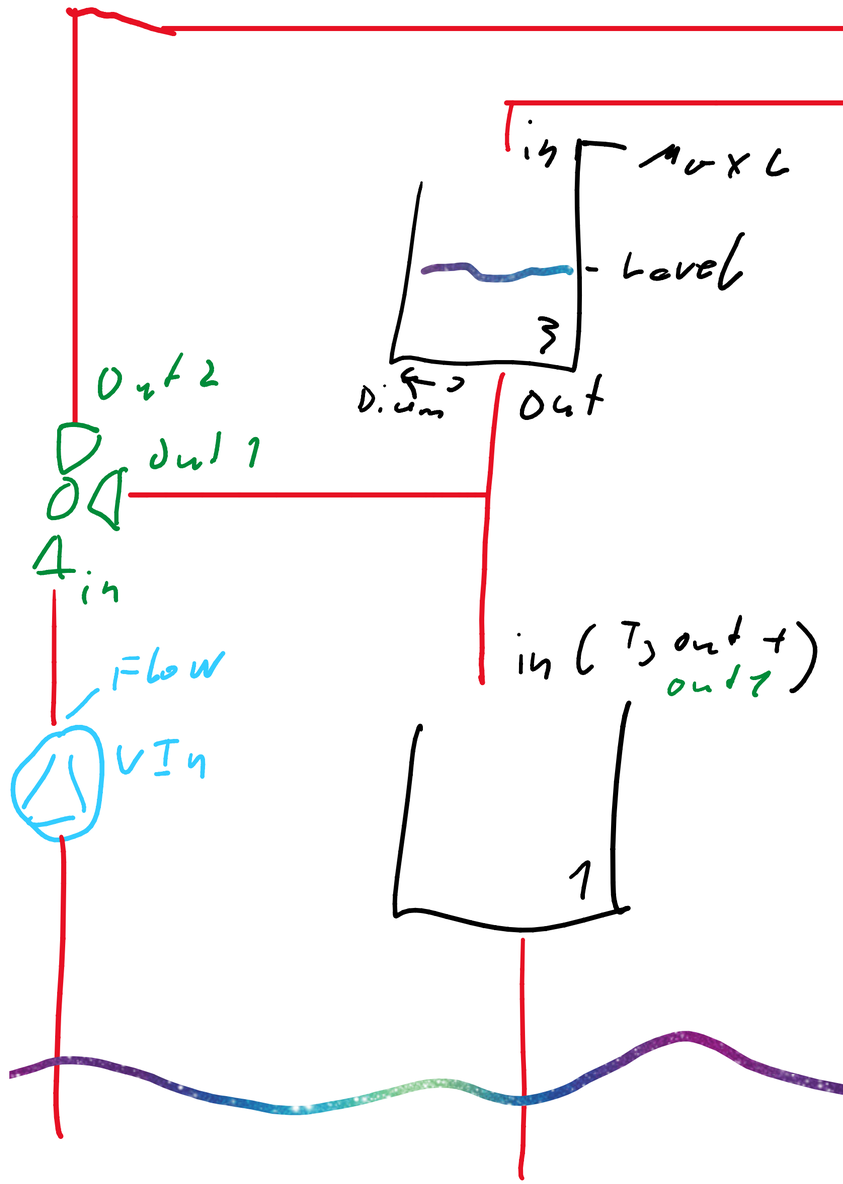


Figure 6: Proxy controller applied to live data. The red lines represent the limits of the upper tank. pump's on/off (low/high) state. The green (orange) line plots the water level in the lower (upper) tank.

Evaluation

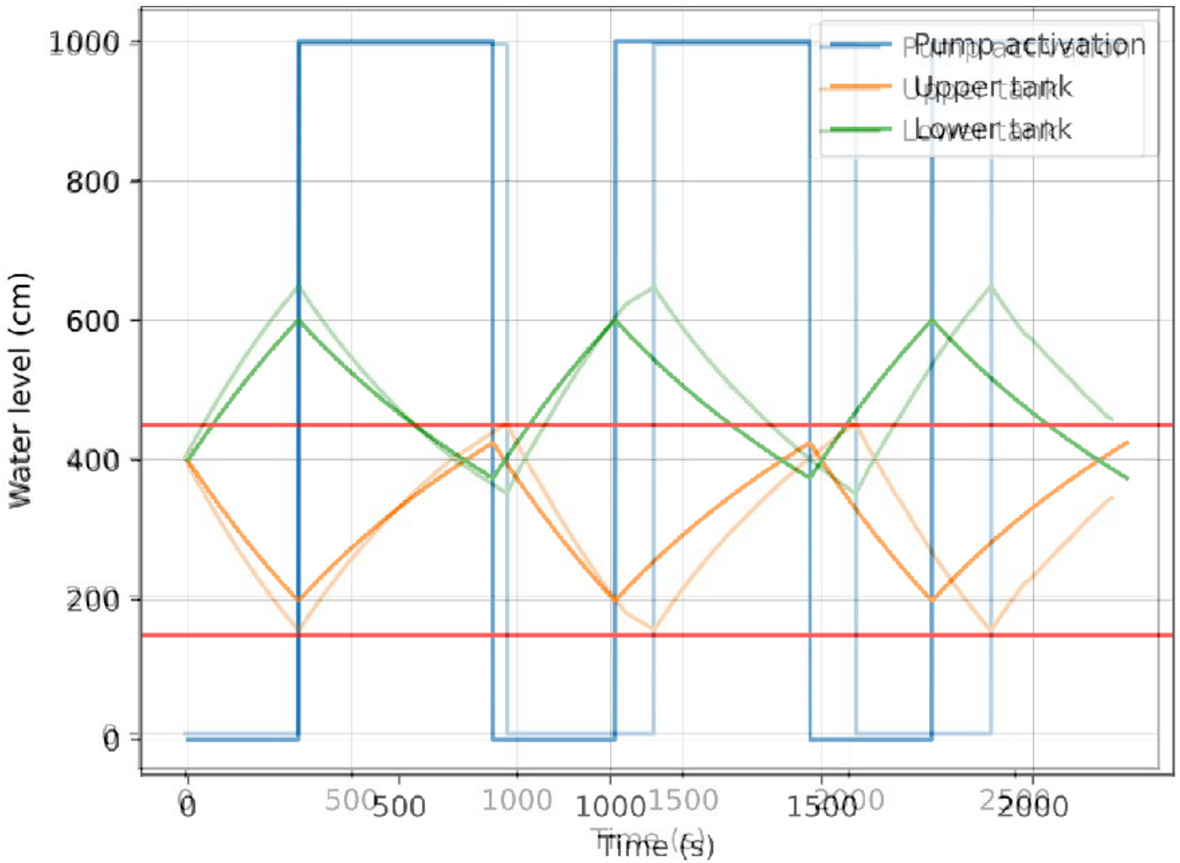


Figure 3: Cyclic behavior of the plant over time. The horizontal axis represents time (seconds). The vertical axis corresponds to the pump state (on or off) or tank water level (cm). The blue line tracks the pump's on/off (low/high) state. The green (orange) line plots the water level in the lower (upper) tank.

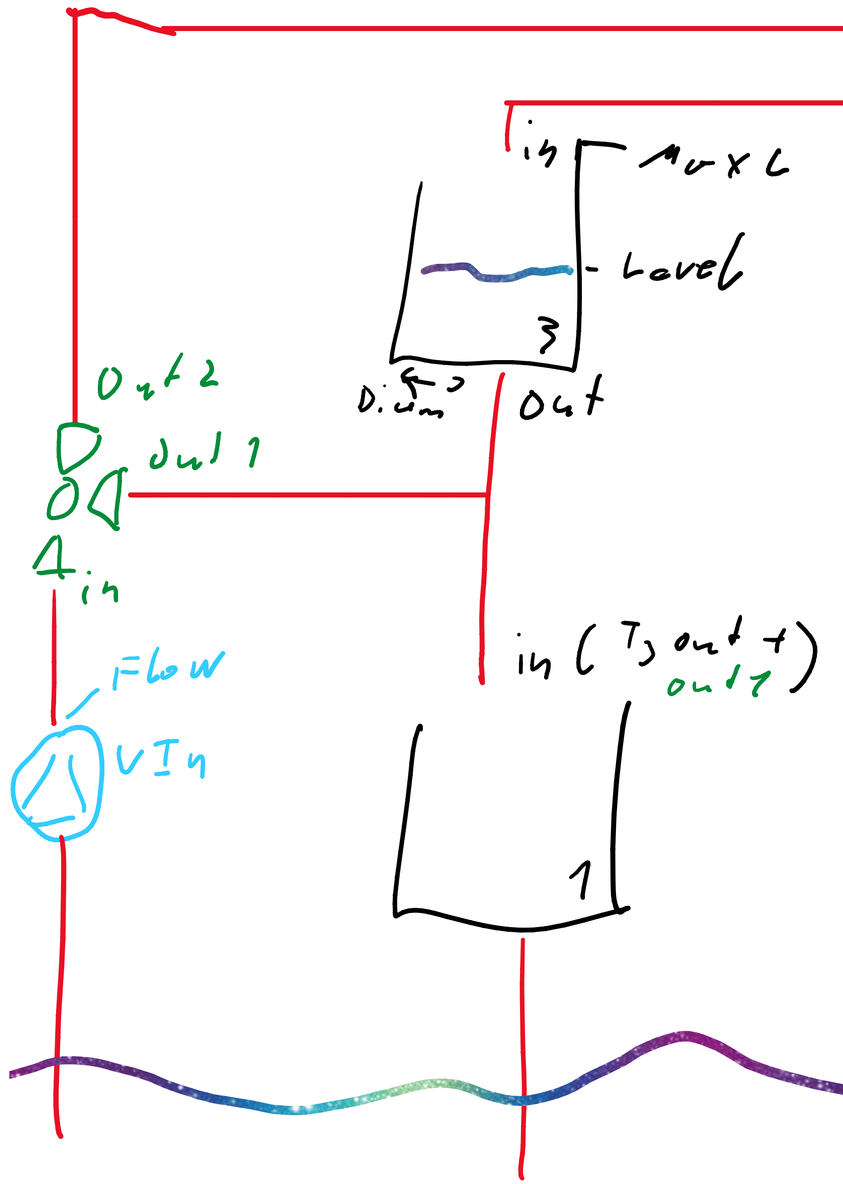


Figure 6: Proxy controller applied to live data. The red lines represent the limits of the upper tank.

Conclusion

- It works for non-linear systems such as water tanks.
- We will follow-up on the approach.