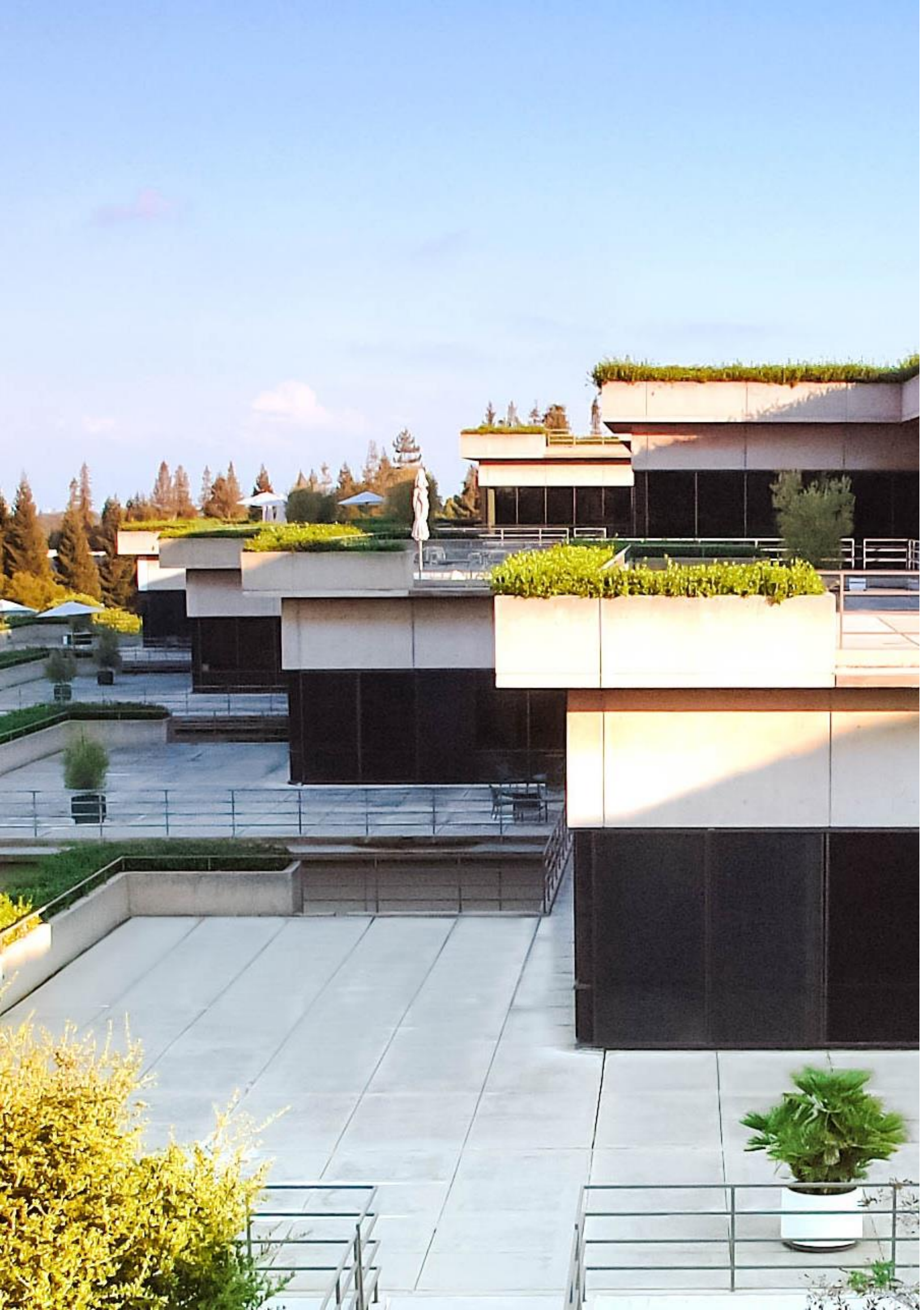


parc
A Xerox Company

Learning algorithms for physical systems: challenges and solutions

Ion Matei

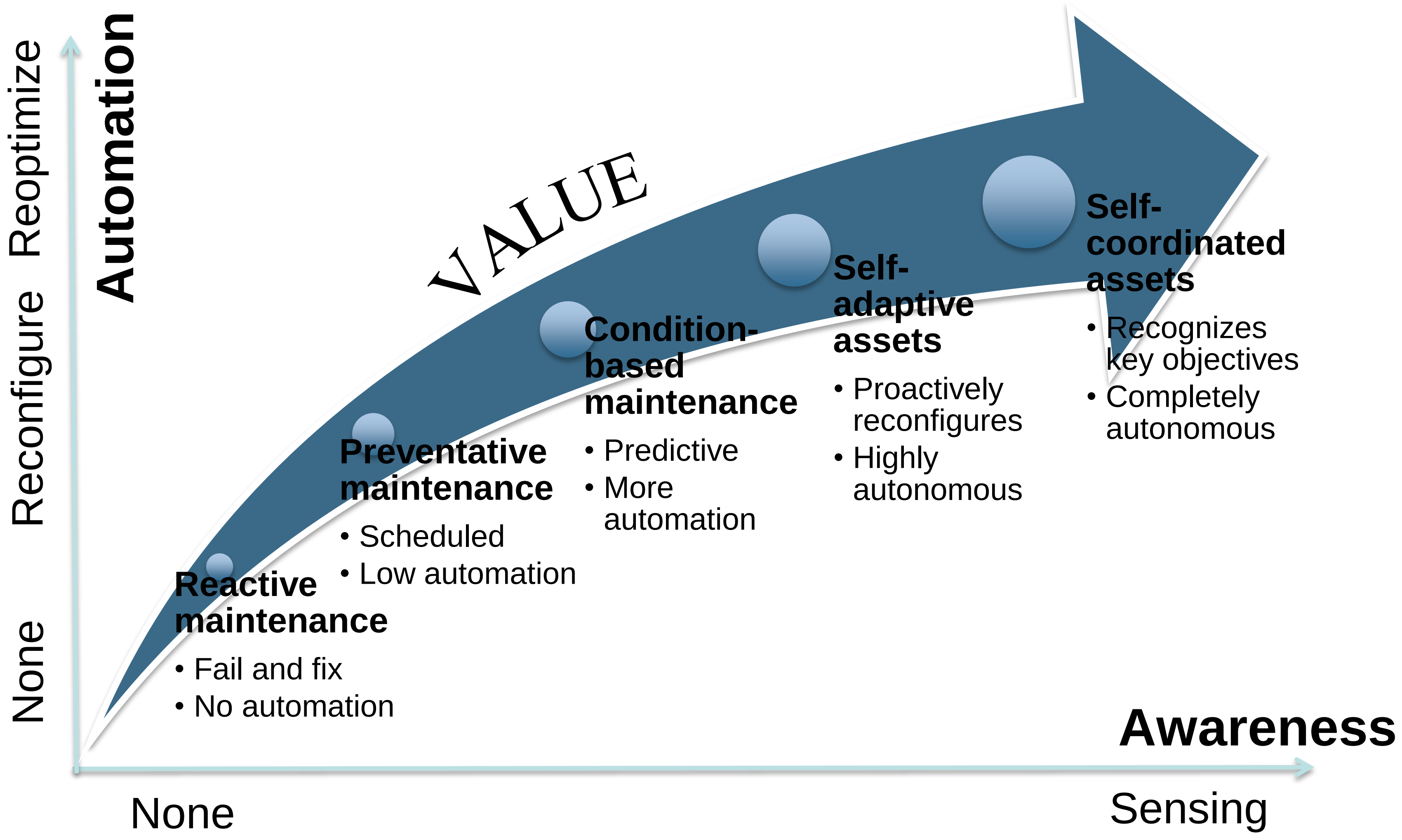
Palo Alto Research Center

- System analytics: how things are done
- Use of models (physics) to inform AI algorithms
- Use of AI algorithms to inform about the physics

What are we trying to avoid

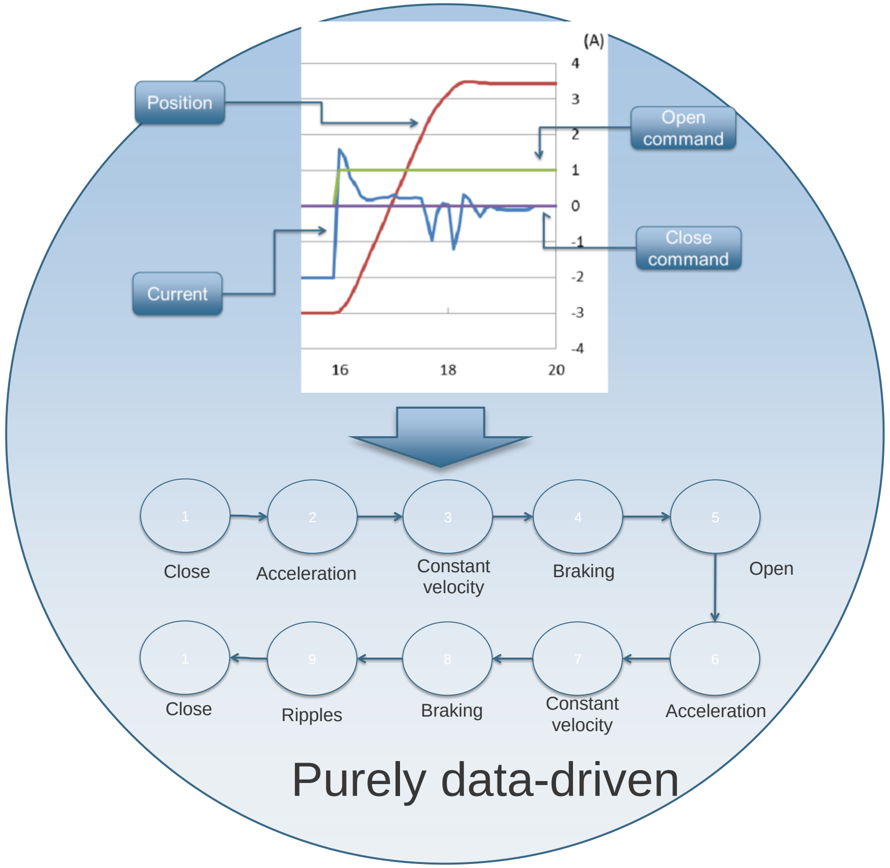
- **Solution:**
  - **ADD MORE INTELIGENCE**

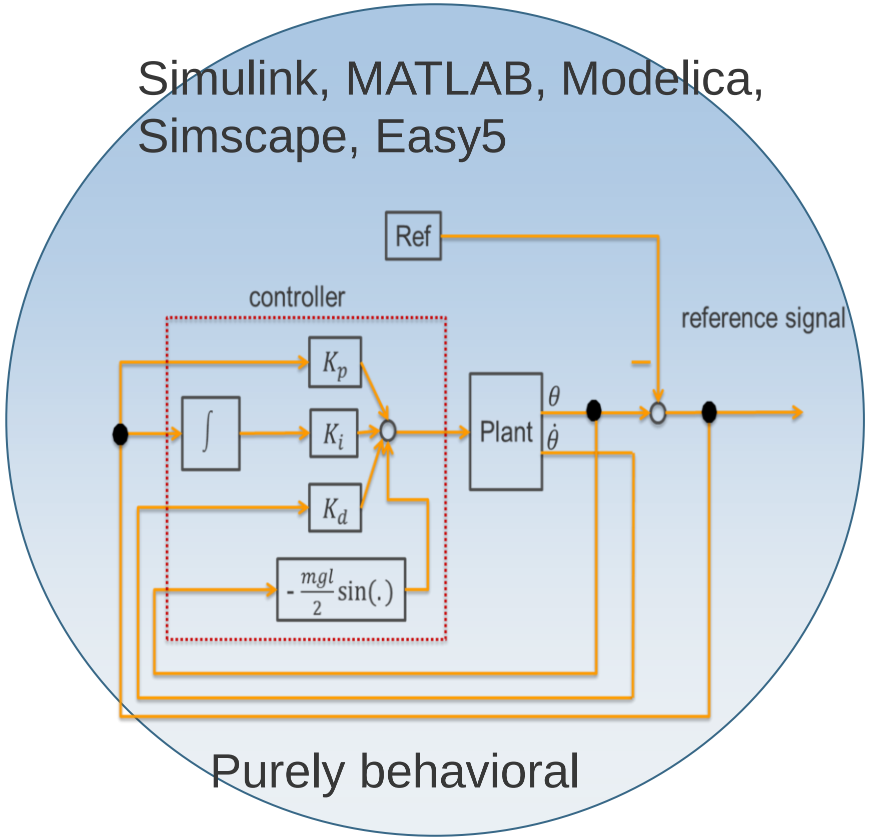
Path forward



parc
A Xerox Company

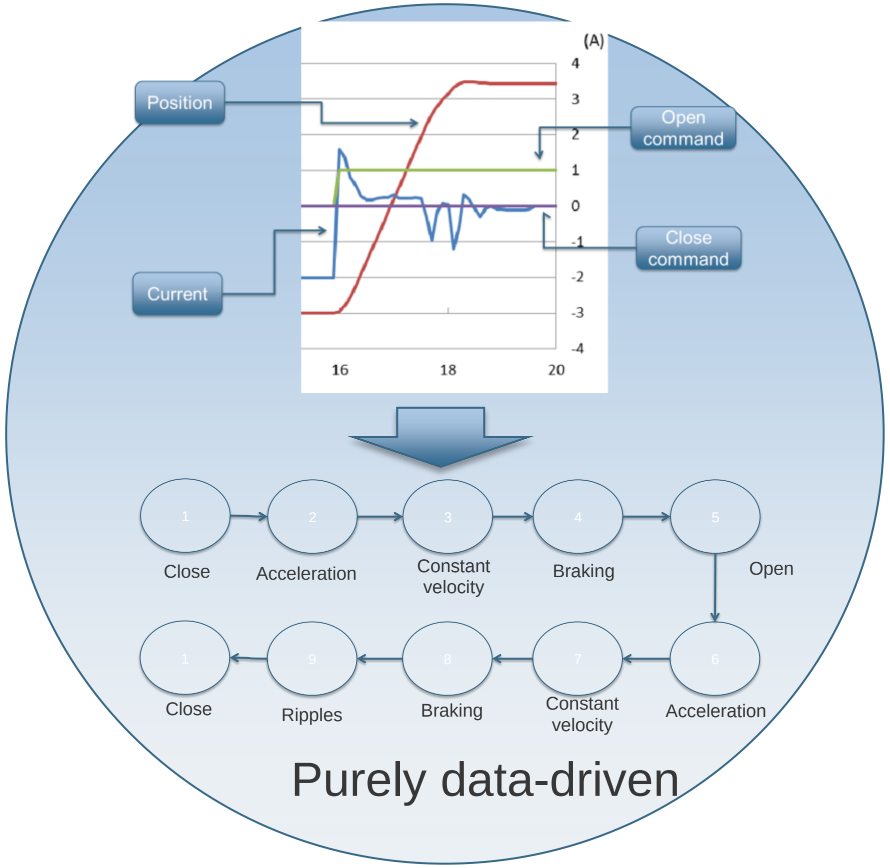
Two Systematic Approaches to Increased Automation





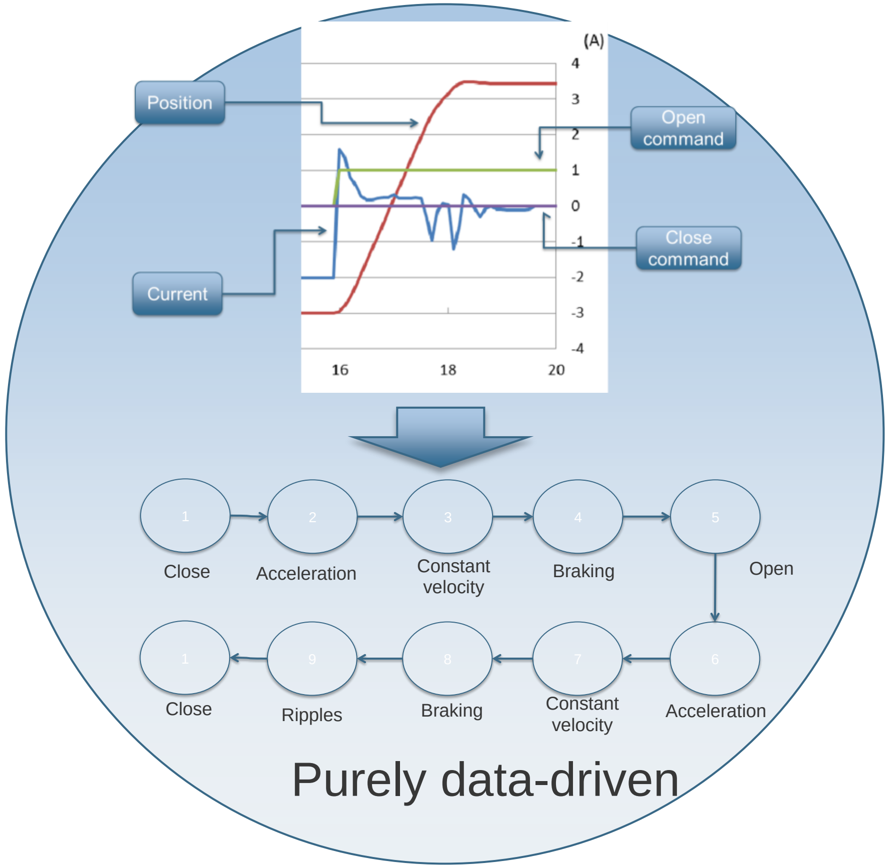
Model-based

Data driven methods



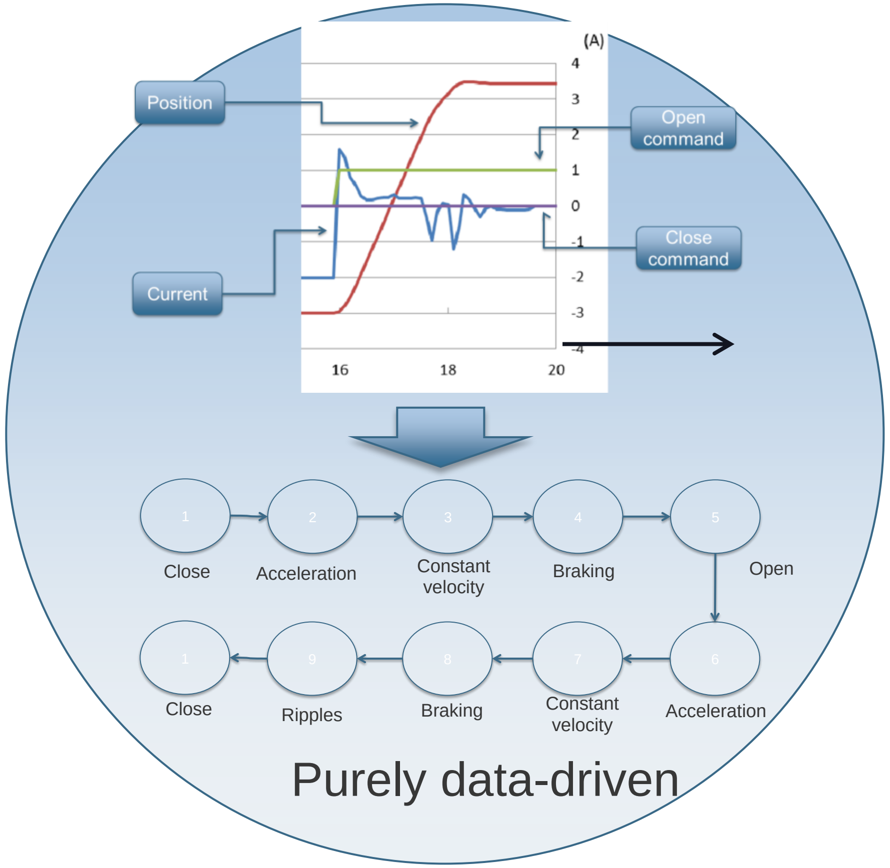
- **Pros:**
  - A plethora of statistical models (regression models, decision trees, neural networks)
  - More and more efficient algorithms for training
  - Easy access to documentation and training platforms
  - Do not require “very” specialized training

Data driven methods

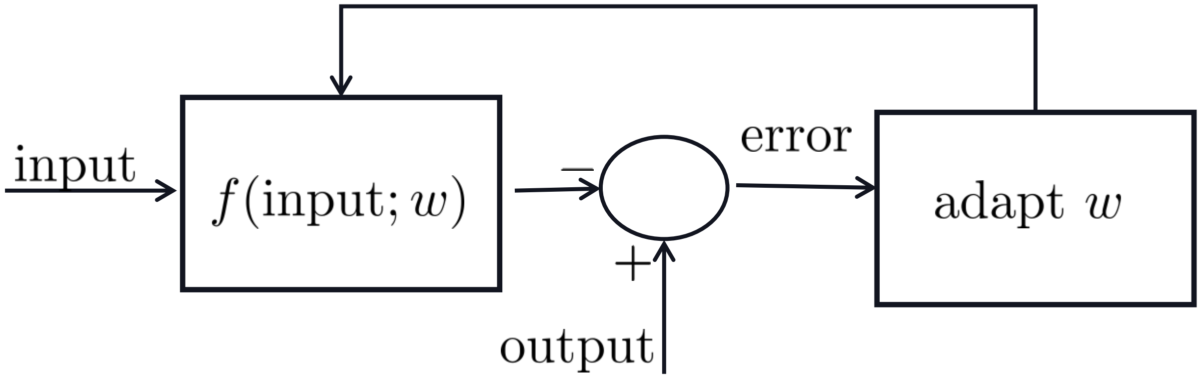


- **Cons:**
  - **May require a lot of data which is not always easy to obtain (assets do NOT fail very often)**
  - **May take a long time to train**
  - **Loss of explainability (systems are seen as black boxes – nothing is known about the internal structure and behavior)**

Data driven methods



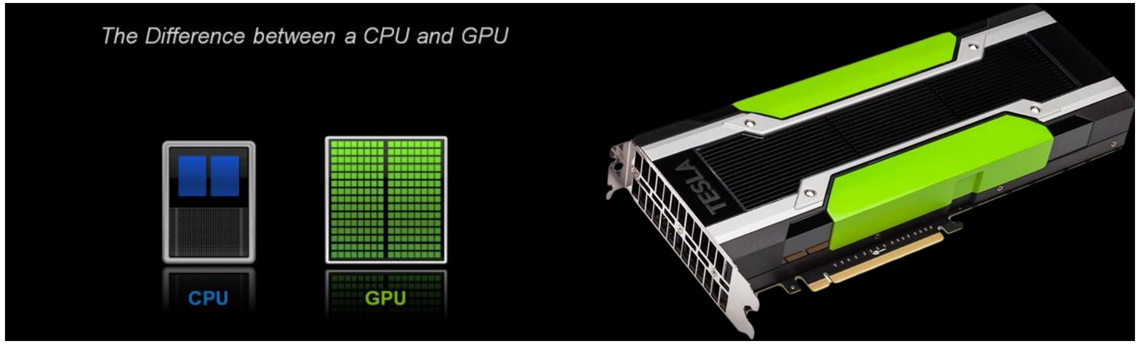
- **Training:**



parc
A Xerox Company

Data driven methods enablers

- **Hardware Technology**
  - **GPU**
  - **Cheap storage**



NVIDIA

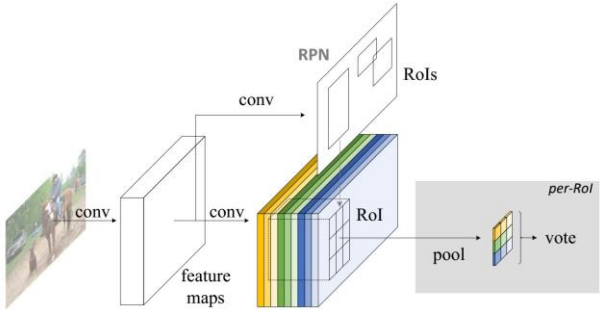
- **“Big data”**

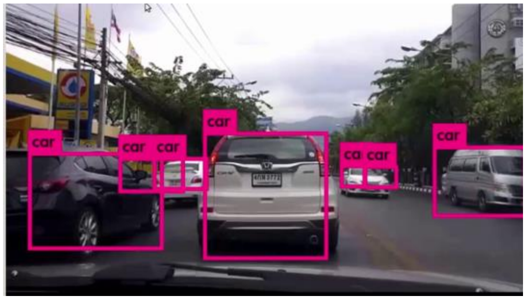
www.blogs.gartner.com

parc
A Xerox Company

Data-driven methods success stories

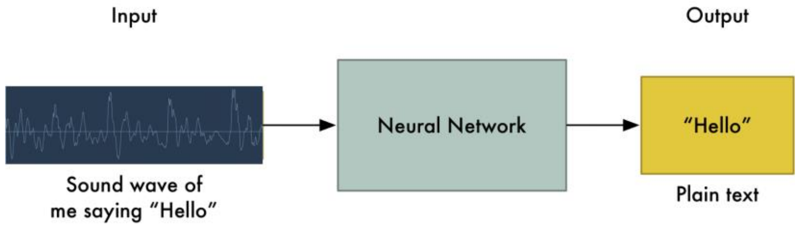
- Object detection and tracking





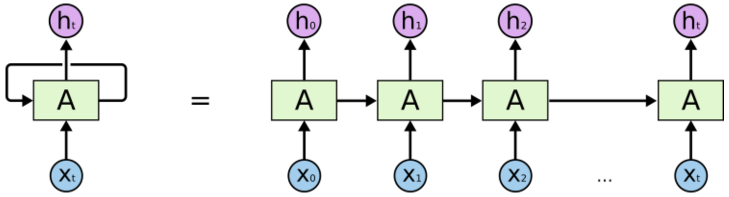
www.towardsdatascience.com

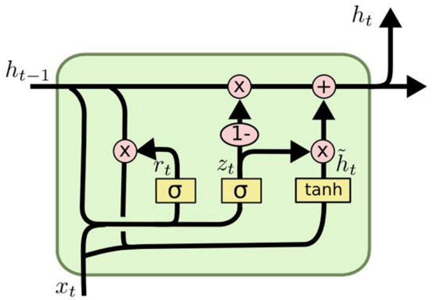
- Speech recognition and translations



www.medium.com

- Text generation





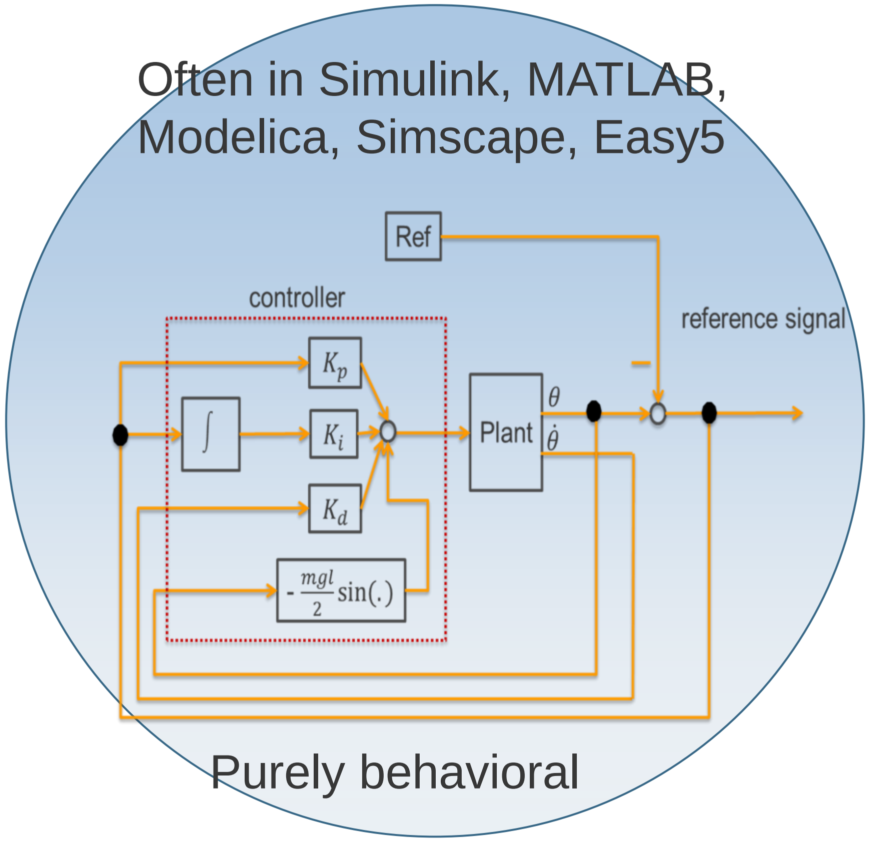
http://colah.github.io

Data-driven methods success stories, but…

Judea Pearl, 2011 ACM Turing award

"All the impressive achievements of deep learning amount to just curve fitting."

Model-based methods

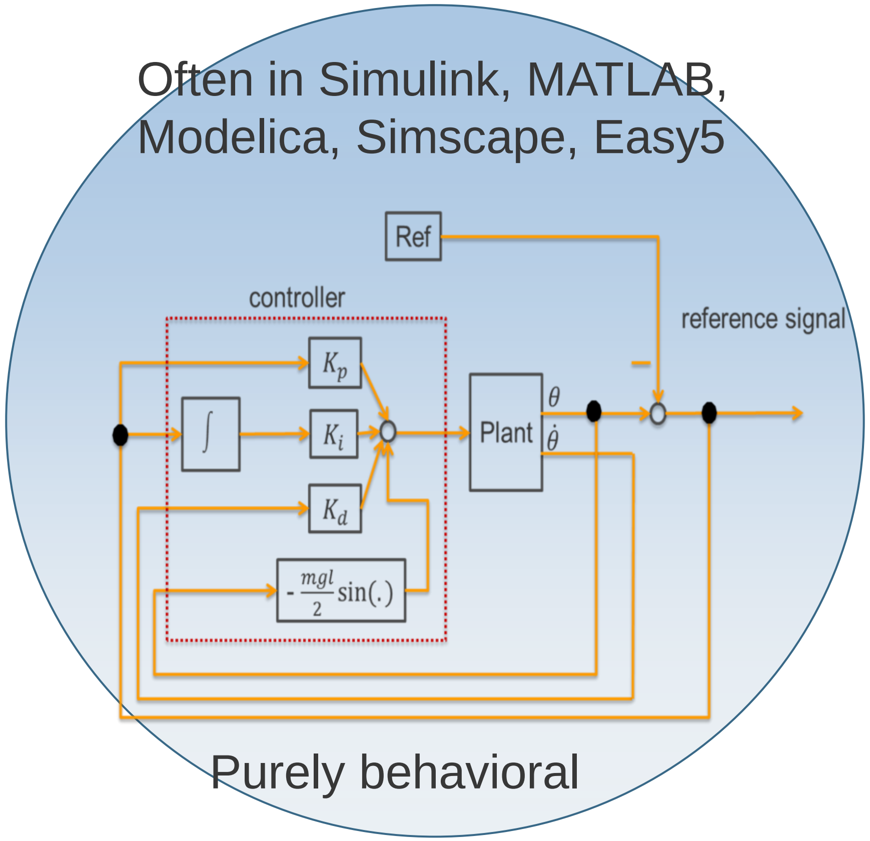


Model-based

- **Pros:**
  - **Vast history of model-based results (reasoning, control, diagnosis, prognosis)**
  - **Well established modeling languages and tools (Matlab, Modelica, Simulink, OpenModelica)**
  - **Support for both causal (input-output maps) and acausal (physics-based) models**
  - **Explainability (can connect to particular components and physical processes)**
  - **Require less data**

parc
A Xerox Company

Two Systematic Approaches to System Analytics

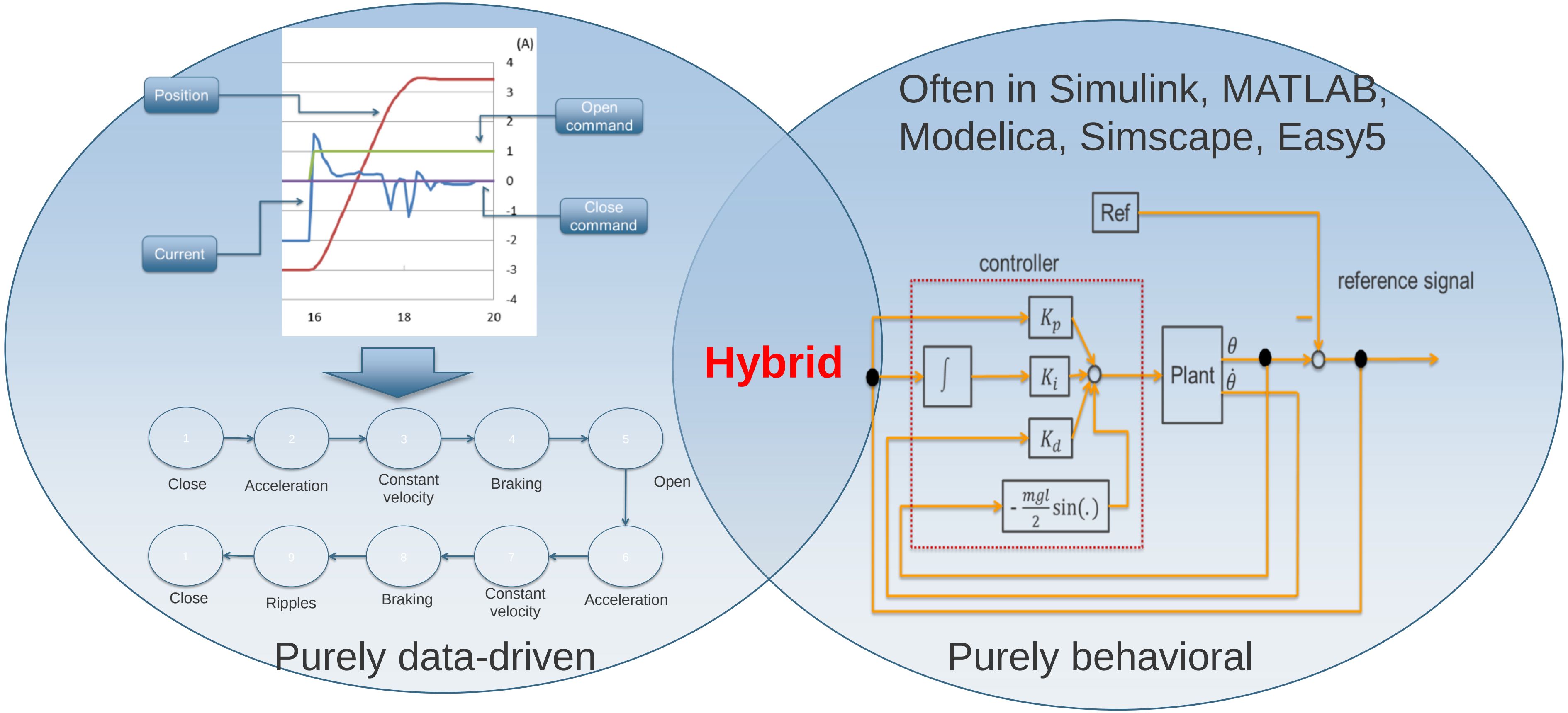


Model-based

- **Cons:**
  - Work well for specific classes of problems (e.g. linear systems)
  - Does not always scale to complex systems
  - Modeling complex systems can be expensive
  - Requires deep expertise
  - Models are not very accurate due to simplifying assumptions

parc A Xerox Company

Model-based methods



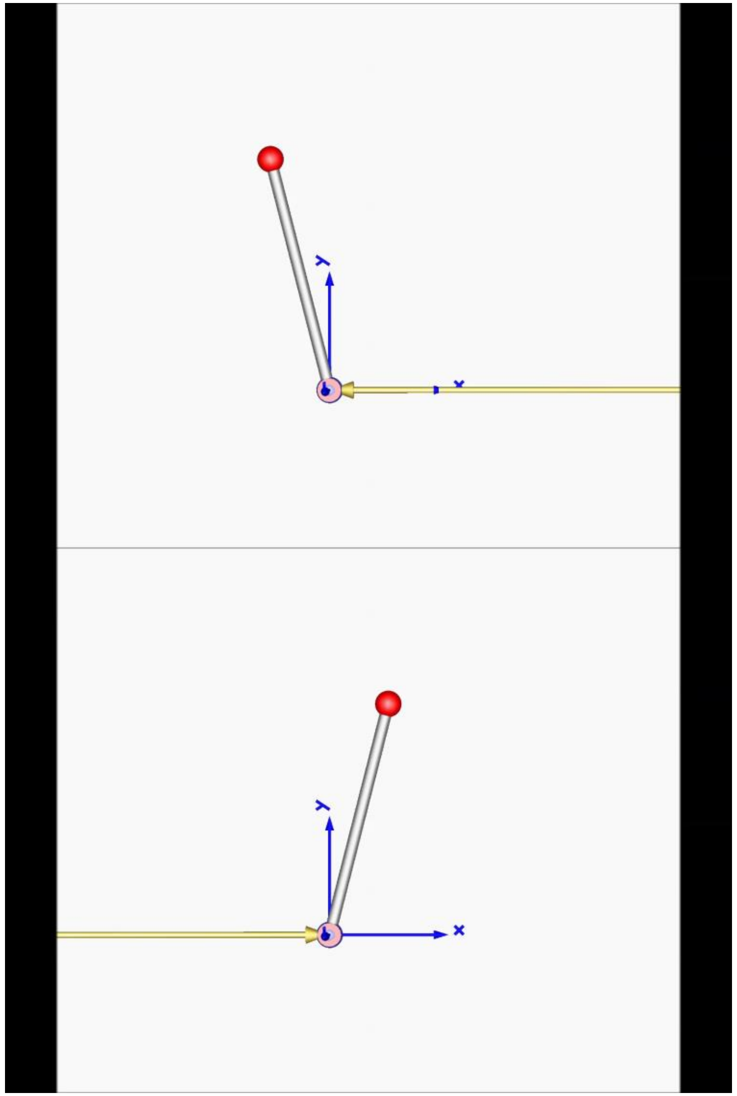
How to benefit the from the combining the two approaches?

parc
A Xerox Company

MODELS → MACHINE LEARNING

MAKE USE OF SYSTEM PROPERTIES (REGULARITIES) THAT ARE INFORMED BY (PARTIAL) MODELS

System regularities

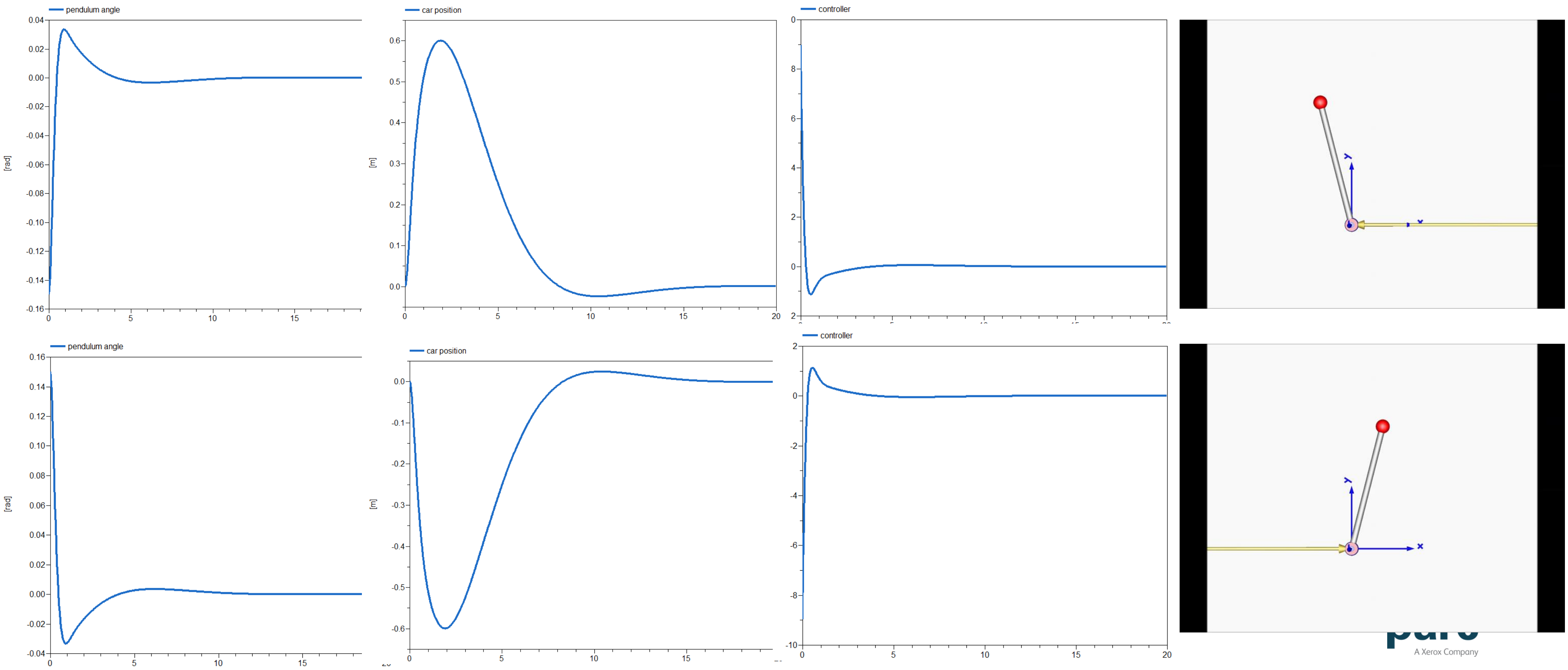


- **Inverted pendulum control**
  - Pendulum stabilization when starting from two opposing angles
  - The force needed to stabilize the pendulum when starting from a left angle can be used when starting from an opposing right angle by changing its sign

Animation produced using a Modelica model

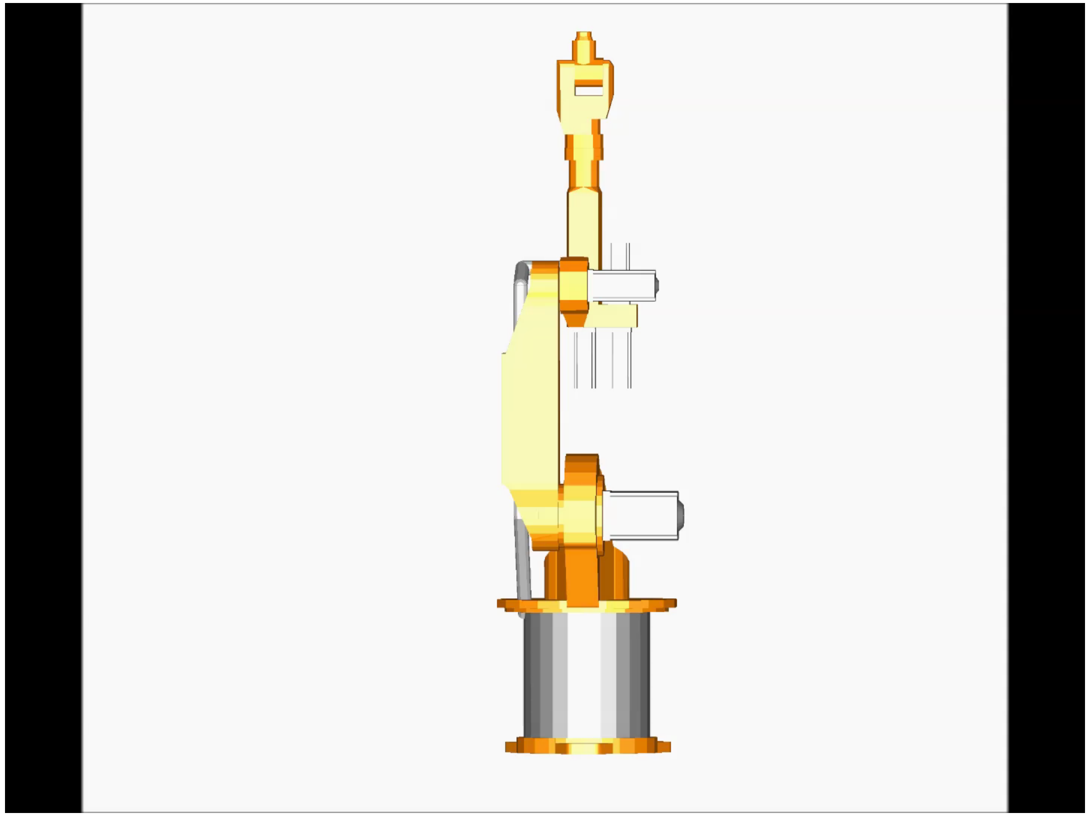
System regularities

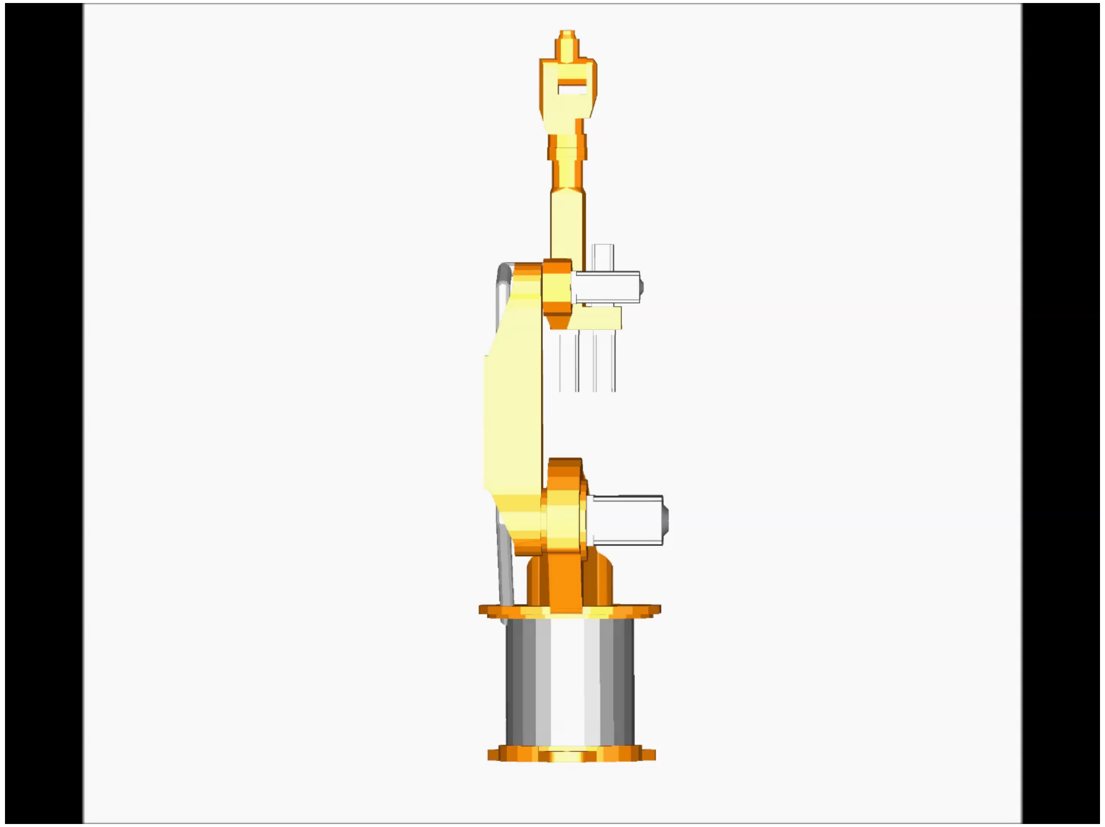
- **Inverted pendulum control:** symmetric motion and control



System regularities

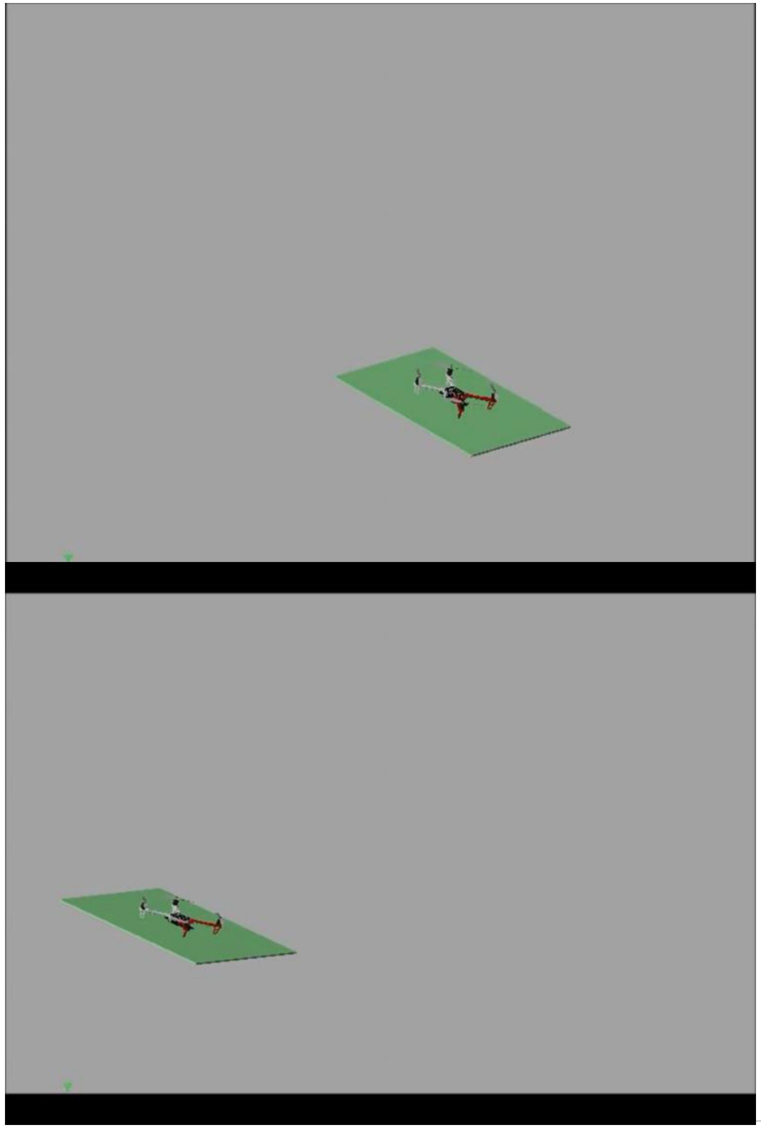
- **Robotics:** symmetries in complex robot motion

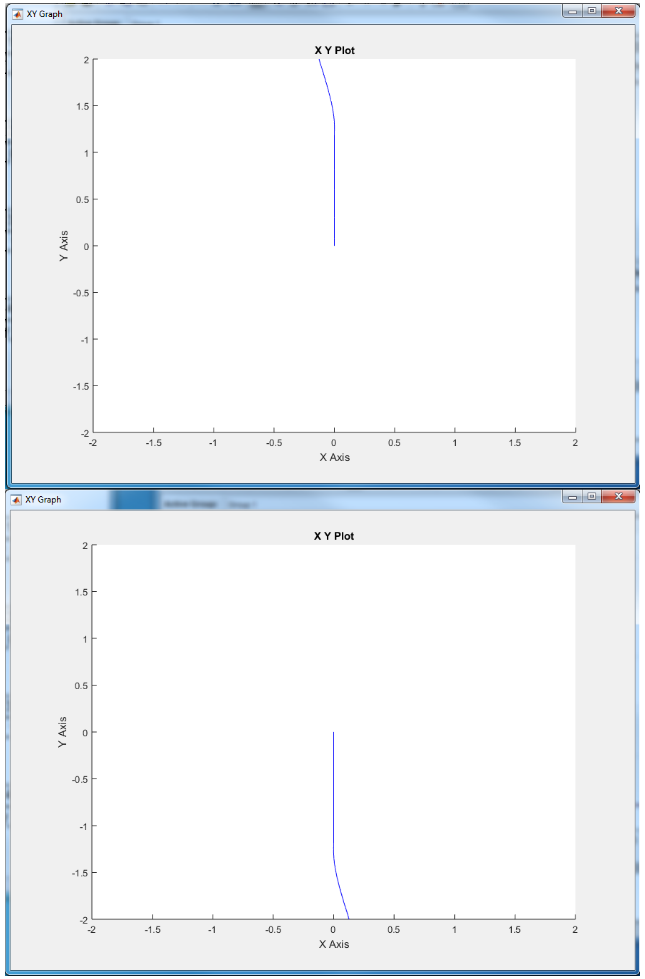




Animation produced using a Modelica model

System regularities





- **Quadrotor motion planning**
  - Using rotational symmetries we can generate a multitude of additional feasible trajectories from one initial trajectory

parc
A Xerox Company

System regularities

Geometric symmetries

- Transformations that take a system trajectory and produce another system trajectory
- Can have discrete symmetries:
  - Rotations at a fixed angle

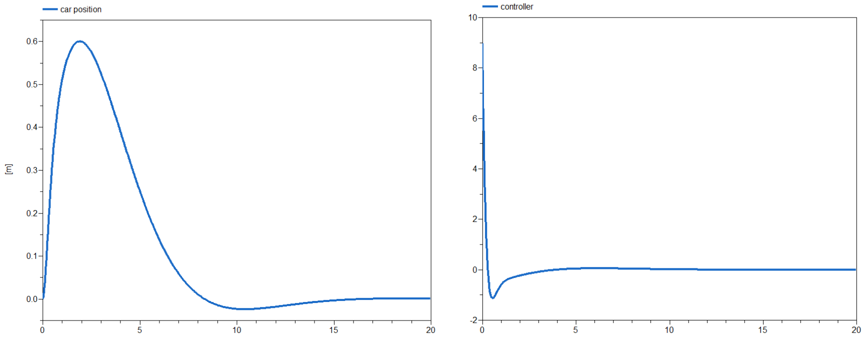
$$\Gamma(x,u) = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix}$$

- Parametrized symmetries
  - Lie symmetries

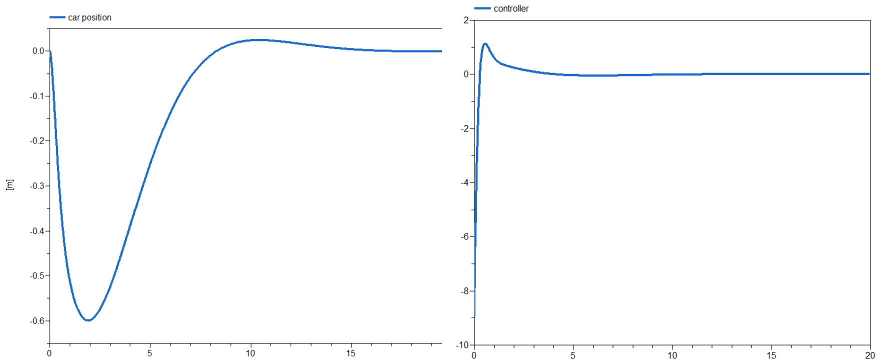
$$\Gamma_\varepsilon(x,u) = (x + \varepsilon\xi + O(\varepsilon^2), u + \varepsilon\eta + O(\varepsilon^2))$$

- Can check for typical symmetries (scaling, translations, rotations)
- Can be learned from data

$$\frac{dx}{dt} = f(x,u)$$



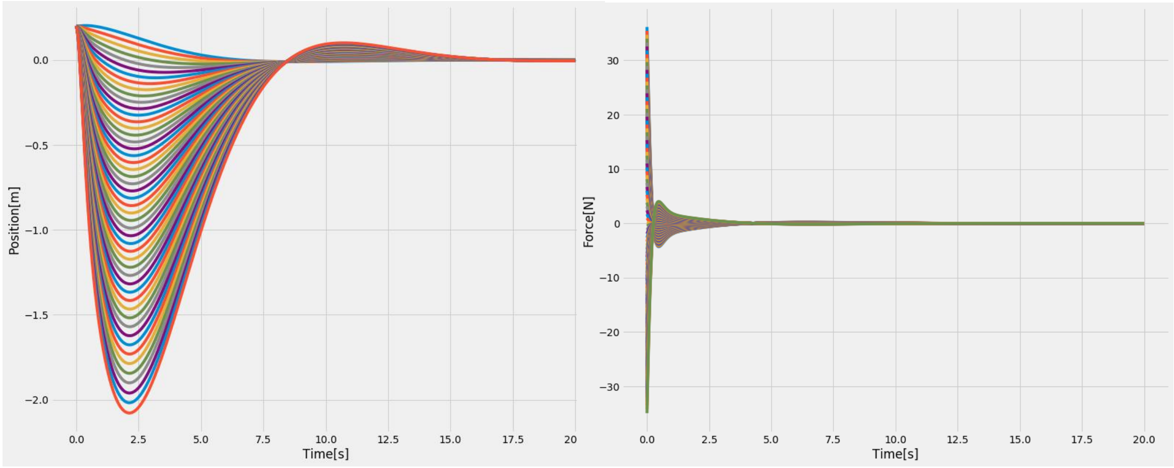
$$\Gamma(x,u) = -(x,u) = (\hat{x},\hat{u})$$



$$\frac{d\hat{x}}{dt} = f(\hat{x},\hat{u})$$

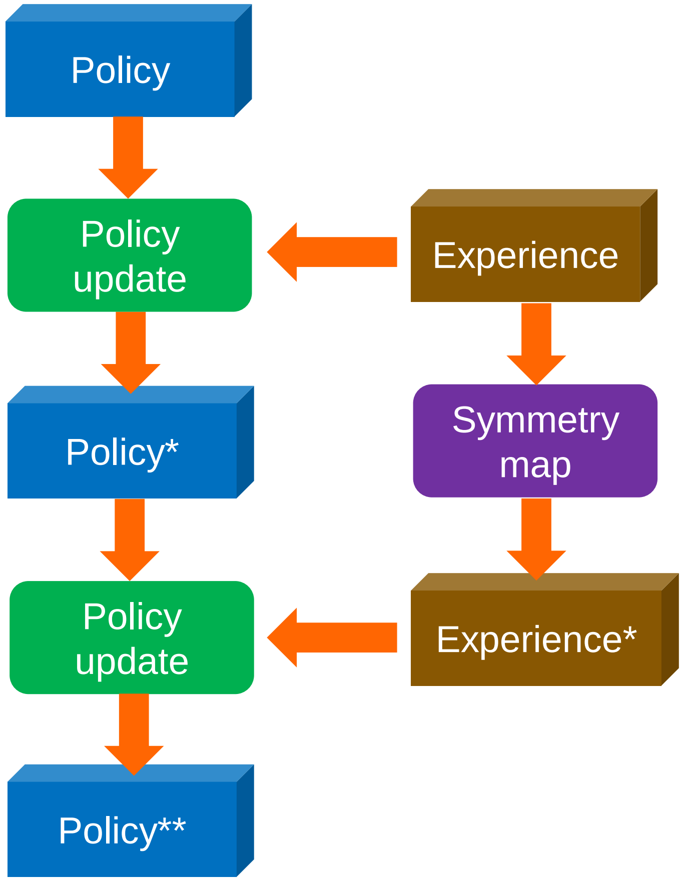
Leveraging system regularities for policy learning

- **Reinforcement learning**:
  - **Area of machine learning and control that tells us what actions (controls) should we take when interacting with an environment to maximize some reward.**
  - **Suitable for incomplete, or model free case (or when other methods fail)**
  - **Requires large amount of data (experience) to learn the best policy**



Leveraging system regularities for policy learning

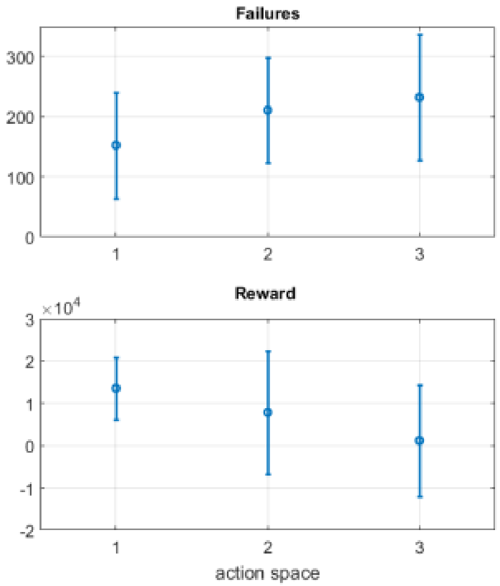
- **Symmetry enhanced reinforcement learning:**
  - **Use geometric symmetries to augment the training data set used for policy learning**
  - **Using a discrete symmetry can double the size of the training data set**
  - **More symmetries … more data**



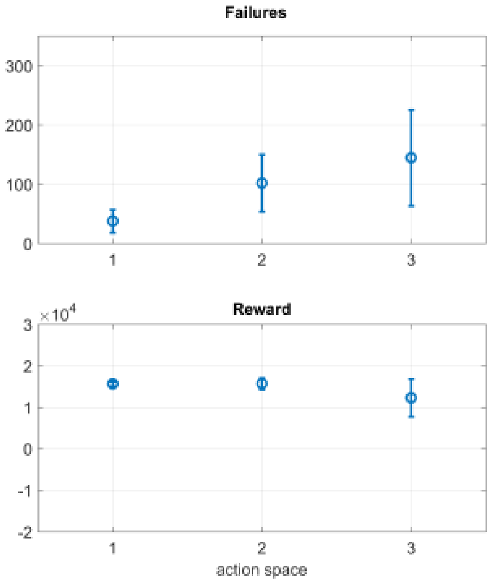
Leveraging system regularities for policy learning

- **Inverted pendulum example**:
  - **State-space discretization: 7 points for the positions, 2 points for the velocity, 8 points for the angle, 2 points for the angular velocity**
  - **Action space:**

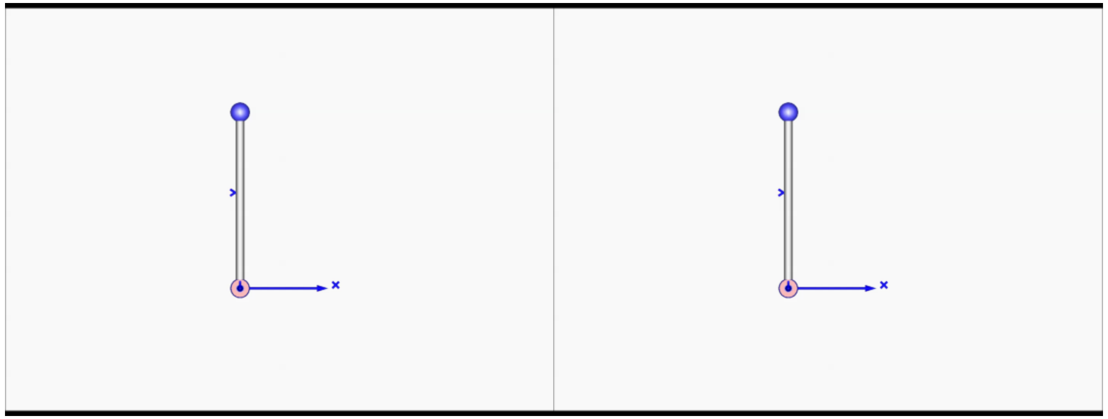
action space 1	action space 2	action space 3
$\{-1, 1\}$	$\{-1, 0, 1\}$	$\{-1, -0.5, 0.5, 1\}$



Standard reinforcement algorithm



Symmetry enhanced reinforcement algorithm



Training with symmetry use

Training without symmetry use

- **Smaller number of failures (faster convergence)**
- **Higher rewards**
- **Less uncertainty**

MACHINE LEARNING → MODELS

MAKE USE OF MACHINE LEARNING TOOLS (ALGORITHMS) TO LEARN (PHYSICAL) MODELS

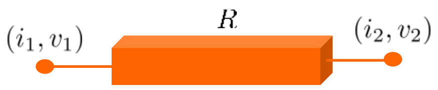
CAUSAL VS. ACAUSAL MODELS

- **CAUSAL MODELS**
  - Output is completely determined by the current and past inputs, states and outputs
  - Typical in machine learning, control and signal processing

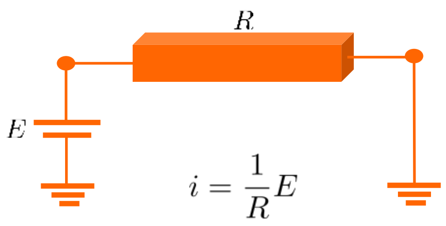
$$y = f(u)$$

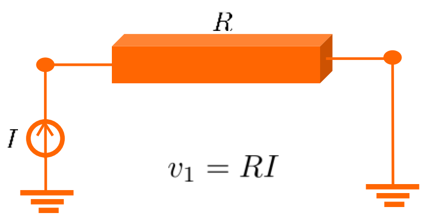
$$p(y|u)$$

- **ACAUSAL MODELS**
  - There is no clear notion of input and outputs
  - Behavior described by constraints between variables



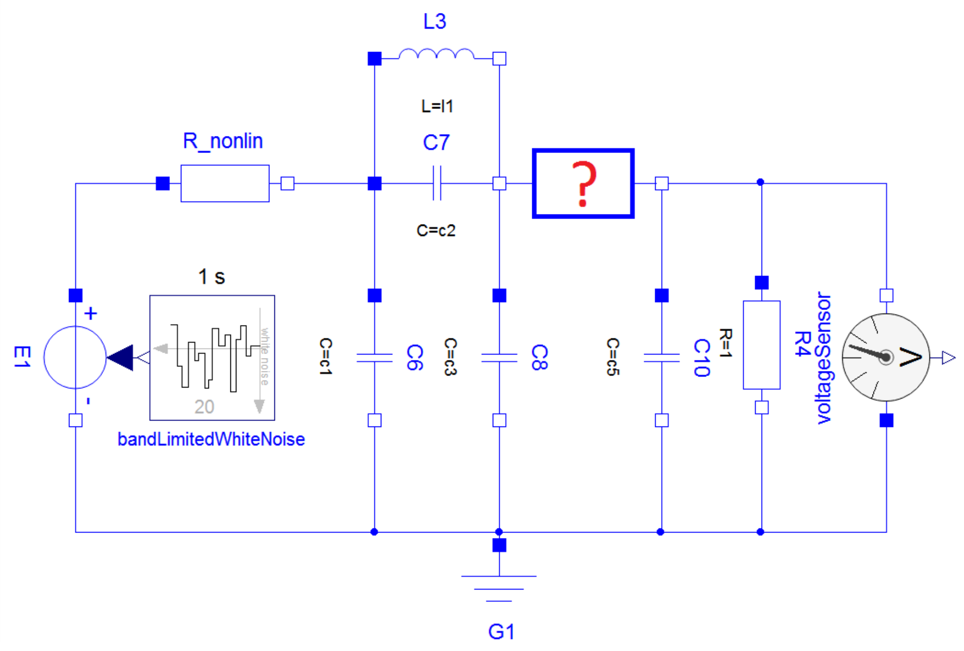
$$i_1 + i_2 = 0$$
$$Ri_1 = v_1 - v_2$$





parc A Xerox Company

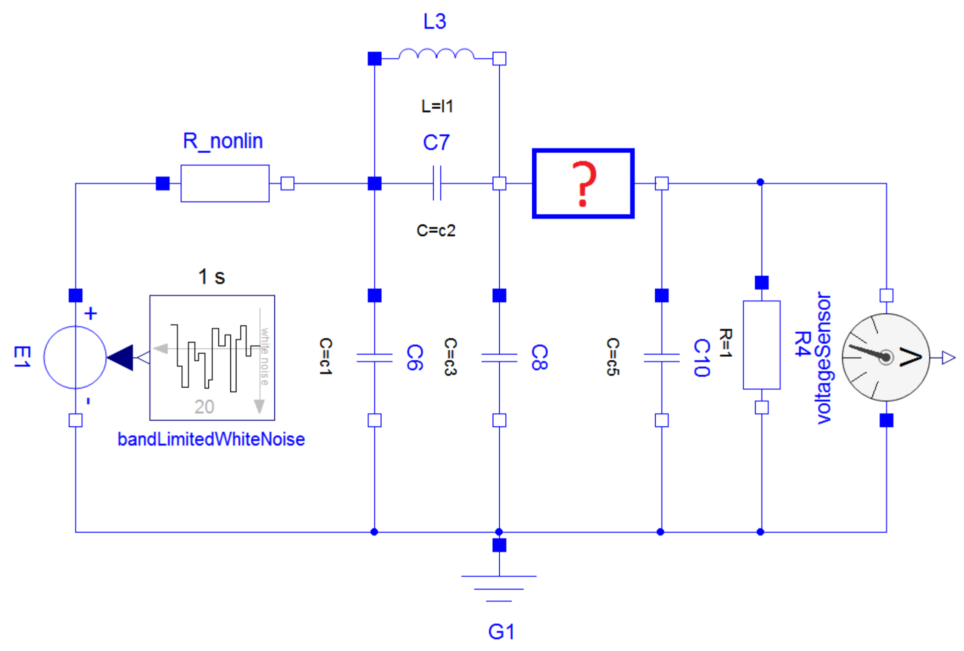
LEARNING PHYSICAL MODELS IN PARTIALLY KNOWN SYSTEMS



- **Learn a model that fits the data**
- **What representation?**
- **How can I make sure it makes sense?**

parc
A Xerox Company

LEARNING PHYSICAL MODELS IN PARTIALLY KNOWN SYSTEMS



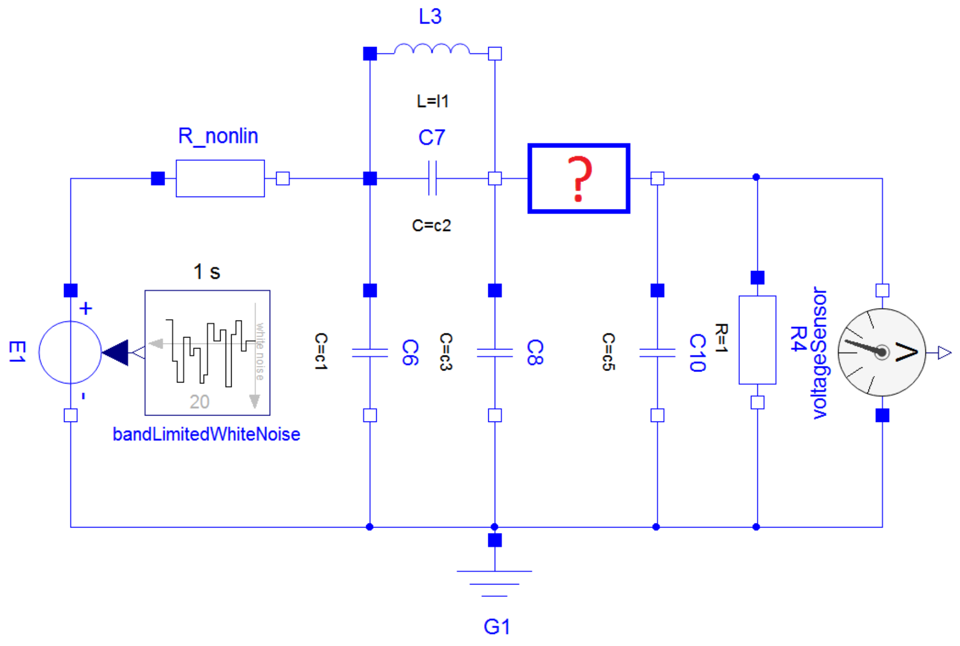
- **What representation?**
  - **Parametrized constraint equation:**

$$f(x; w) = 0$$

- **How can I make sure it makes sense?**
  - **Impose a priori feasibility constraints on parameters (component does not generate energy, e.g., passive)**

parc
A Xerox Company

LEARNING PHYSICAL MODELS IN PARTIALLY KNOWN SYSTEMS



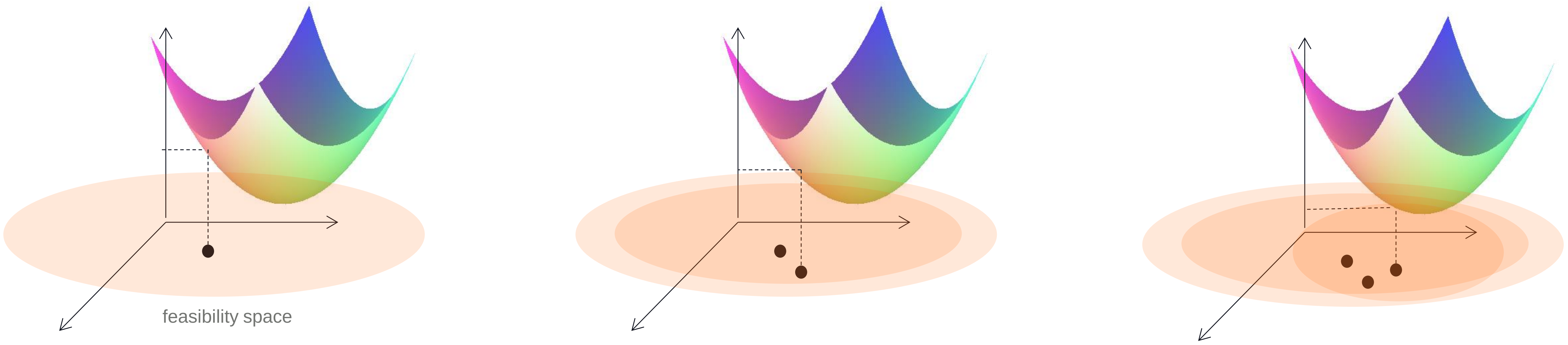
- **What representation?**
  - **Parametrized constraint equation:**

$$f(x;w) = 0$$

- **How can I make sure it makes sense?**
  - **Joint learning of best parameter and their feasibility space (optimization problem with unknown constraints)**

LEARNING PHYSICAL MODELS IN PARTIALLY KNOWN SYSTEMS

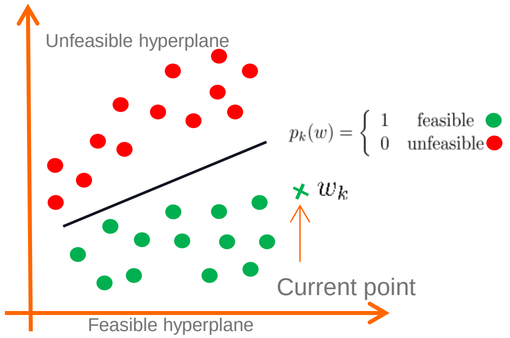
- **Strategy for joint learning of parameters and their constraints:**
  - **explore-exploit**

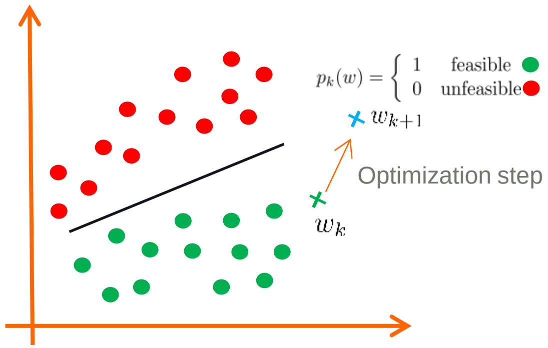


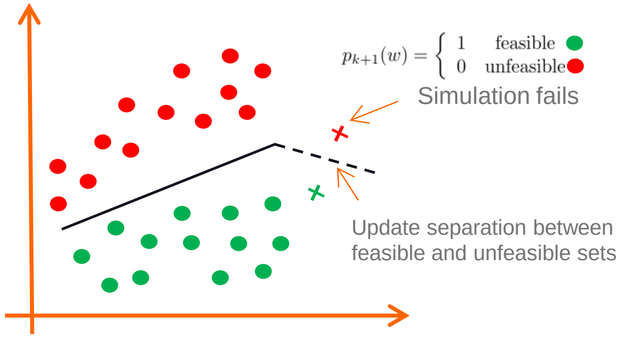
parc
A Xerox Company

LEARNING PHYSICAL MODELS IN PARTIALLY KNOWN SYSTEMS

- **How do I learn the feasibility space?**
  - Train a classifier
  - Use the model simulator to label points

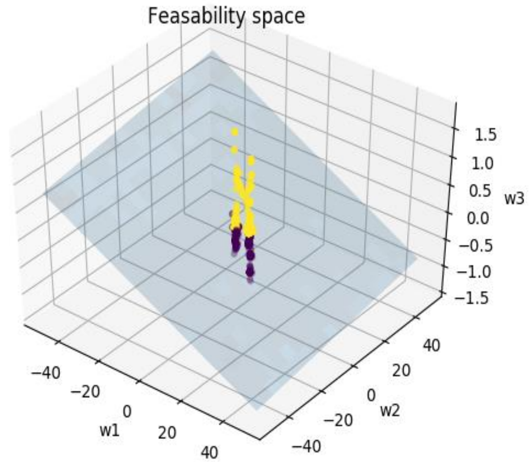




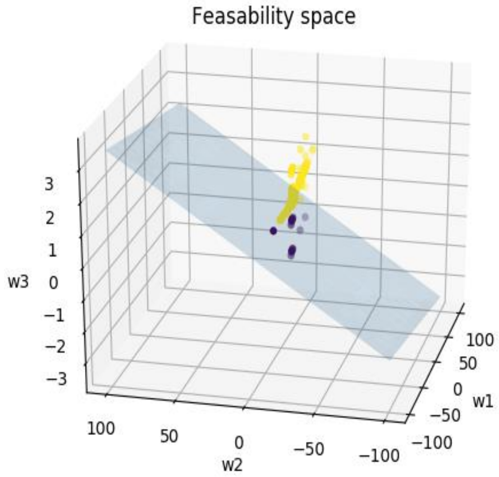


LEARNING PHYSICAL MODELS IN PARTIALLY KNOWN SYSTEMS

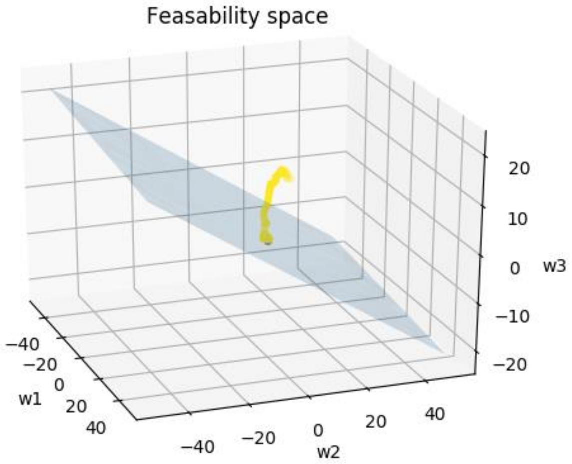
- **What you would expect to learn?**
  - At least a local separating hyperplane



Powell



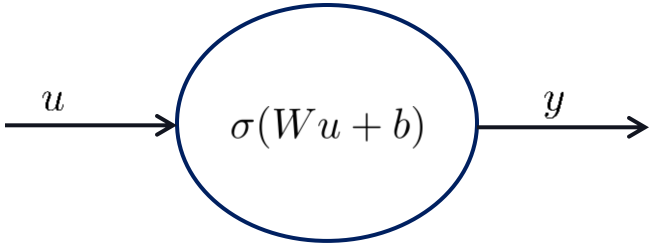
Simplex



Quadratic programming

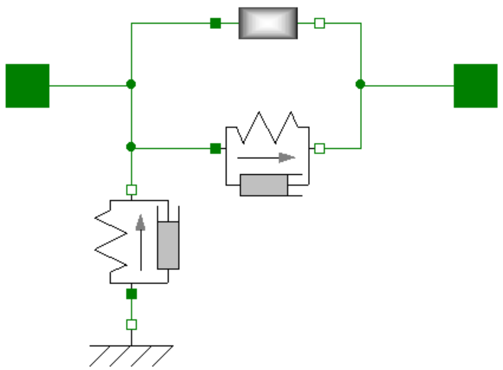
LEARNING PHYSICAL MODELS: OTHERS IDEAS

- **Build “neural network” like representations**



Neural network cell:

Linear part + activation function (nonlinear)

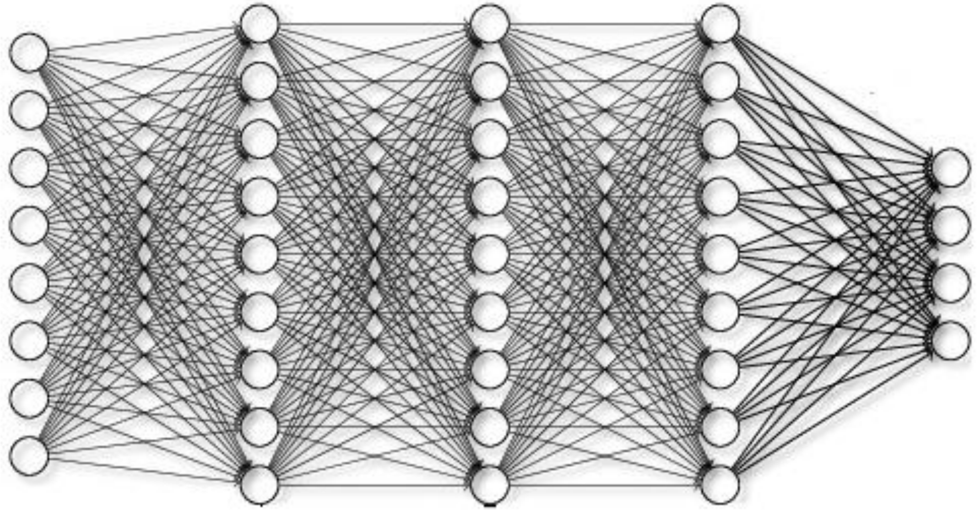


Acausal neural network cell

Nonlinearities in the damper and springs

LEARNING PHYSICAL MODELS: OTHERS IDEAS

- **Build "neural network" like representations**

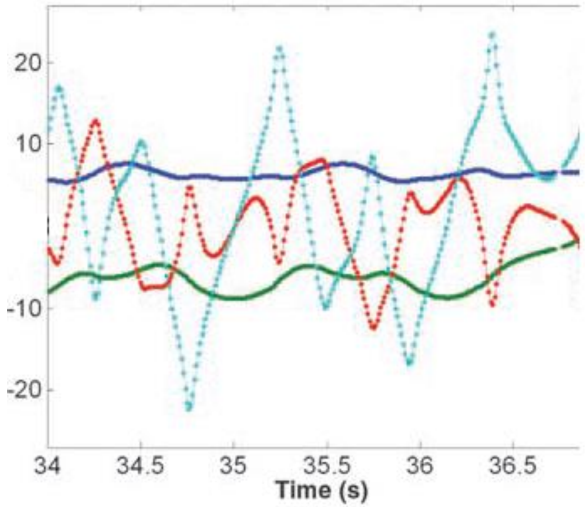


  - **Need to add boundary conditions**
  - **Much more difficult to train**
    - **forward propagation needs to simulate a dynamical system**
    - **backward propagation require computation of gradients**

parc
A Xerox Company

LEARNING PHYSICAL MODELS: OTHERS IDEAS

- **Discovering Hamiltonians, Lagrangians and other laws of geometric and momentum conservations:**
  - **Symmetries and invariants underlie almost all physical law**
  - **Use of genetic algorithms (the main challenge is avoiding trivialities)**

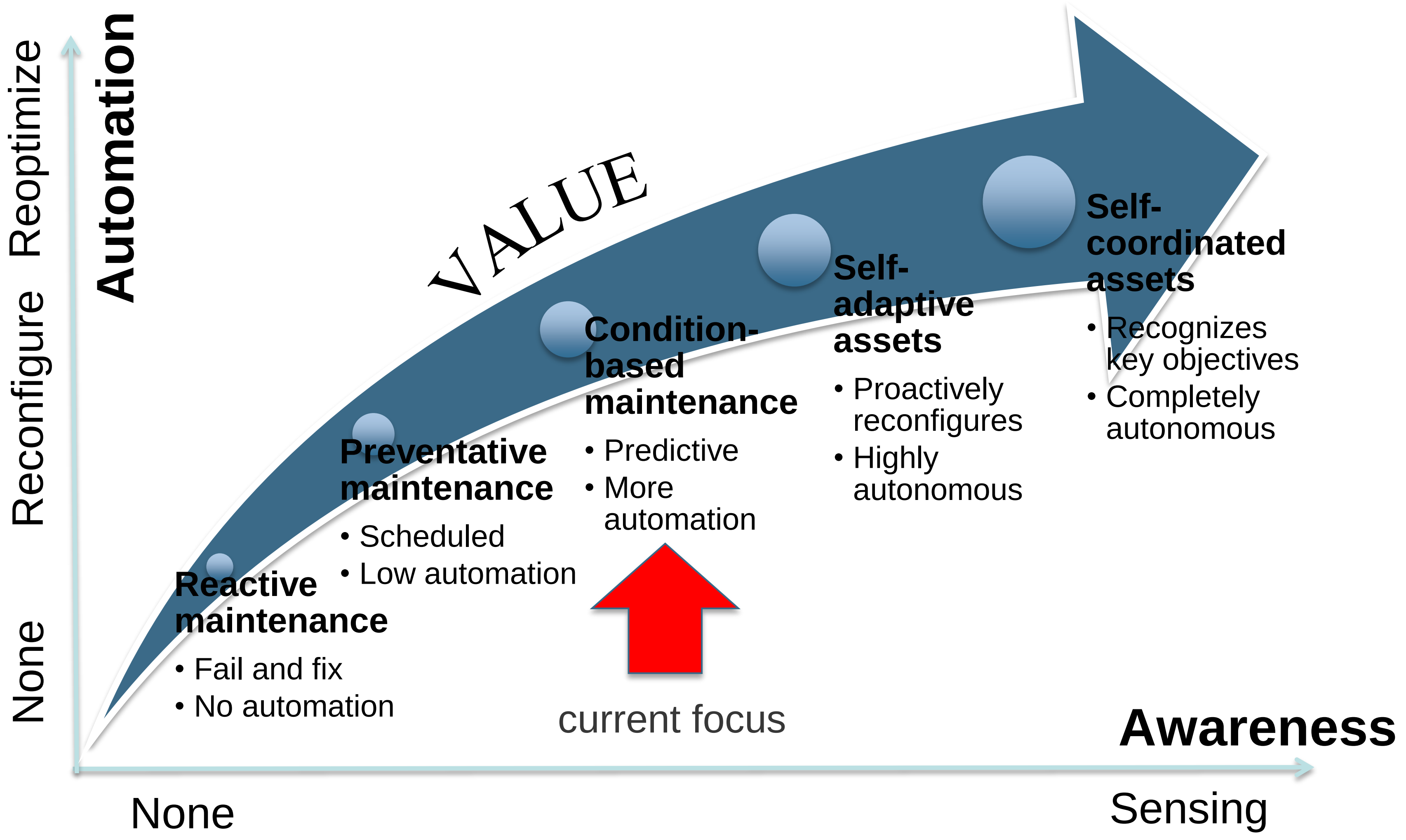


$$L_1^2(m_1+m_2)\omega_2^2 + m_2 L_2^2 \omega_2^2 + m_2 L_1 L_2 \omega_1 \omega_2 cos(\theta_1-\theta_2) - 19.6 L_1(m_1+m_2)\cos\theta_1 - 19.6 m_2 L_2 \cos\theta_2$$

Hamiltonian

M. Schmidt, Hod Lipson, “Distilling Free-Form Natural Laws from Experimental Data,” Science Magazine

Where we are today



parc A Xerox Company

What's next…

- Seamless integration between model-based and data-driven methods
- Explainable AI (what are we learning?)
- Assured AI (what can we guarantee?)
- Design use cases