Machine Learning and Modeling Seminar

Symbolic Regression Methods in Reinforcement Learning

Erik Derner

erik.derner@cvut.cz

Czech Institute of Informatics, Robotics, and Cybernetics

& Faculty of Electrical Engineering

Czech Technical University in Prague, Czech Republic

5 December 2019

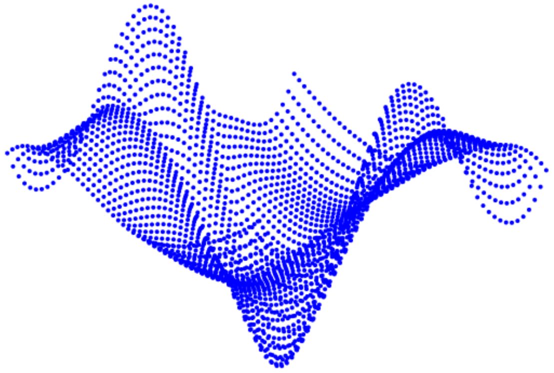
Robotics for Industry 4.0

http://r4i.ciirc.cvut.cz/

- Fields of interest
  - Industrial robotic manipulators
  - Mobile robotics
  - Machine perception and learning
  - Networked control systems
- Based at CIIRC CTU in Prague
- Czech partner universities
  - Brno University of Technology
  - University of West Bohemia in Pilsen
- International partner universities
  - Delft University of Technology

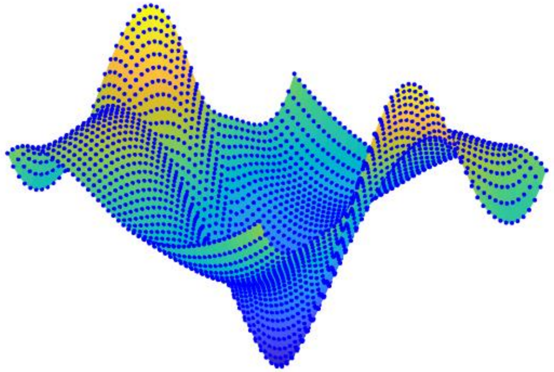
Motivation

- Data modeling approaches
    - Time-varying linear models
    - Gaussian processes
    - Deep neural networks
    - Local linear regression
- Drawbacks
    - Large number of parameters
    - Local nature of the approximator
    - Data-hungry
    - Black box
- Symbolic regression
    - Low number of parameters
    - Small data sets
    - Analytic expressions



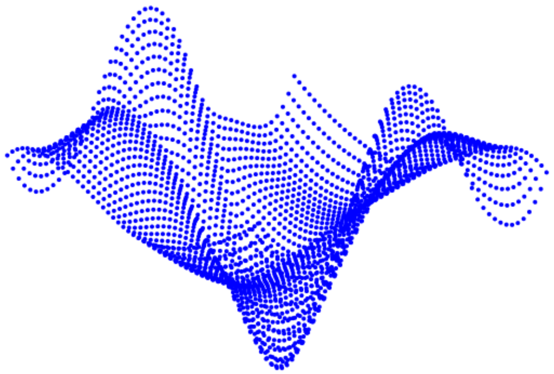
-3.141592654	-30	-23.34719731
-2.932153143	-29	-22.67195916
-2.722713633	-28	-22.07798667
-2.513274123	-27	-21.63117778
-2.303834613	-26	-21.2992009
...	...	...

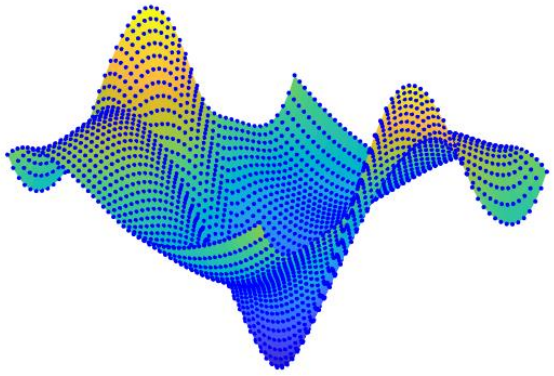
INPUTS OUTPUT



Symbolic Regression (SR)

- Fitting models in the form of mathematical expressions to a set of discrete data points
- Model found by SR will be called analytic model in this talk



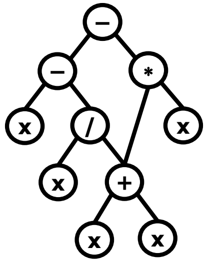
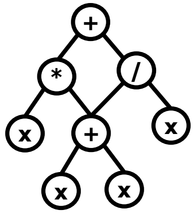


-3.141592654	-30	-23.34719731
-2.932153143	-29	-22.67195916
-2.722713633	-28	-22.07798667
-2.513274123	-27	-21.63117778
-2.303834613	-26	-21.2992009
…	…	…

f = -15.42978401 + 2.42980826 * ((x1 – (x1 * -1.49416733 + x2 * 0.51196778 + 0.00000756)) + (sqrt(power((x1 – (x1 * -1.49416733 + x2 * 0.51196778 + 0.00000756)), 2) + 1) – 1) / 2) ...

Symbolic Regression Algorithms

$$M = \sum_{j=0}^{n_f} \alpha_j F_j(x_1, \dots, x_n)$$

- Finding models composed of several features („trees“)
    - Multiple Regression Genetic Programming [1]
    - Evolutionary Feature Synthesis [2]
    - Multi-Gene Genetic Programming [3]
    - Single Node Genetic Programming [4, 5]

[1] I. Arnaldo et al.: Multiple regression genetic programming (2014)

[2] I. Arnaldo et al.: Building predictive models via feature synthesis (2015)

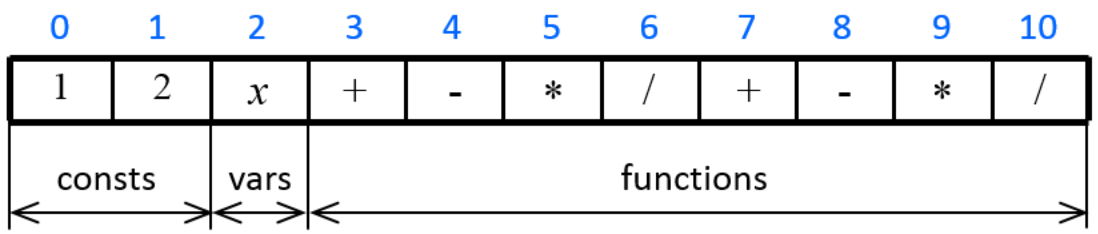
[3] M. Hinchliffe et al.: Modelling chemical process systems using a multi-gene genetic programming algorithm (1996)

[4] D. Jackson: Single node genetic programming on problems with side effects (2012)

[5] J. Kubalík et al.: An improved Single Node Genetic Programming for symbolic regression (2015)

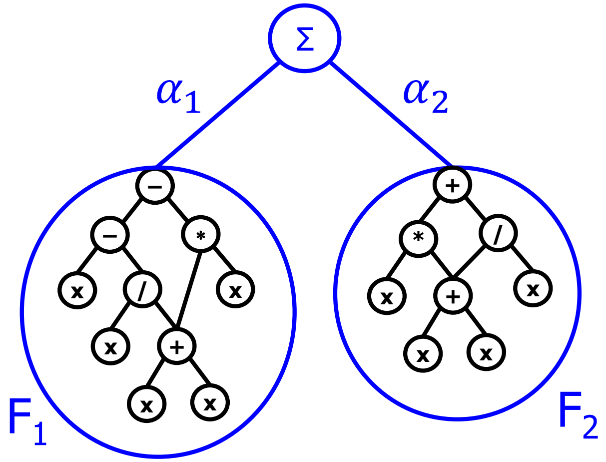
Single Node Genetic Programming (SNGP)

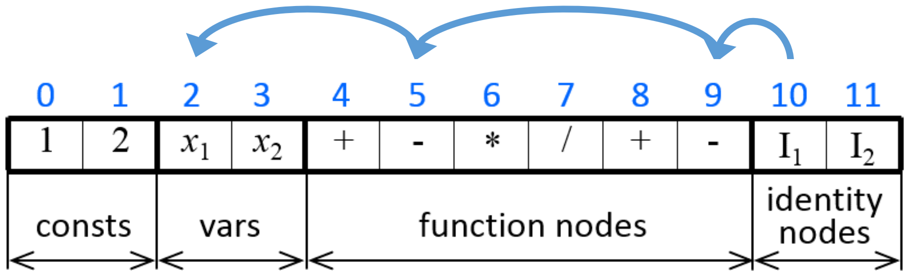
- Graph-based GP technique
- Evolves a population organized as an ordered linear array of individuals, each representing a single program node
- Program node types
  - Terminals – variables, constants
  - Functions
- Evolutionary process
  - SMUT – successor mutation
  - Acceptance rule – best fitness in the population has improved



Analytic Model Structure

- $M = \sum_{j=0}^{n_f} \alpha_j F_j(x_1, \dots, x_n)$
- $F_0 = 1$
- Linear combination of features
- Coefficients α_j can be calculated e.g. by least squares

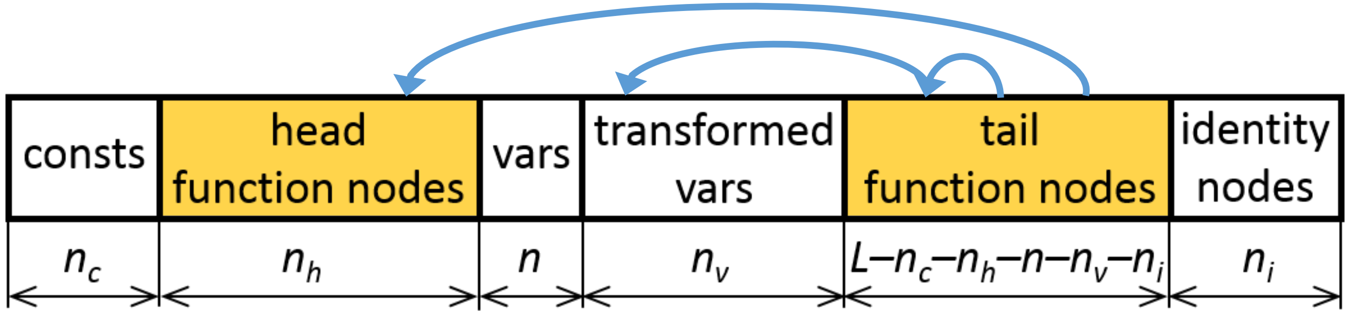




[6] J. Kubalík et al.: Hybrid single node genetic programming for symbolic regression (2016)

Partitioned Population

- Division of population to head and tail partition
- Head partition
  - Root nodes of expressions producing only constant output
- Tail partition
  - Root nodes of variable-output or constant expressions



SNGP Algorithm Overview

Epochs

Threads

Generations

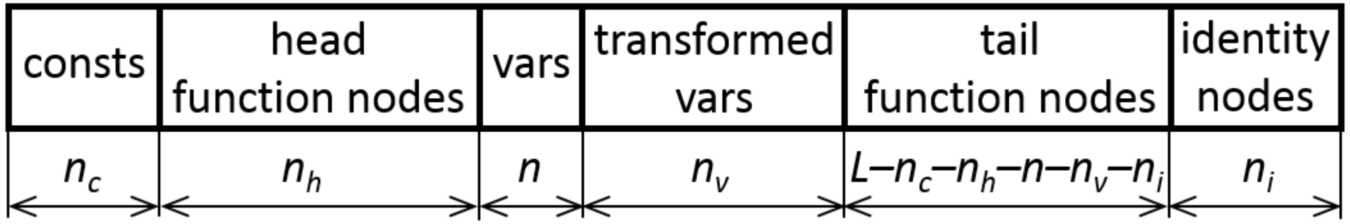
Local tuning

Save best model

initialize population P
build regression model M using P
evaluate M
$e \leftarrow 0$
do
$t \leftarrow 0$
do
$P_t \leftarrow P$
$M_t \leftarrow M$
$generation \leftarrow 0$
do
$s \leftarrow selectNode(P_t)$
$P'_t \leftarrow mutate(P_t, s)$
build model M'_t using P'_t
evaluate M'_t
if (M'_t is not worse than M_t)
$P_t \leftarrow P'_t$
$M_t \leftarrow M'_t$
$generation \leftarrow generation + 1$
while ($generation < l_e$)
$[P_t, M_t] \leftarrow optIdentityNodes(P_t, l_i)$
$[P_t, M_t] \leftarrow optTransfVars(P_t, l_v)$
$t \leftarrow t + 1$
while ($t < n_t$)
$b \leftarrow \underset{t=1,\dots,n_t}{argbest}(M_t)$
$P \leftarrow P_b$
$M \leftarrow M_b$
$e \leftarrow e + 1$
while ($e < n_e$)
return M

SNGP Parameters

- Population size (e.g. 500 individuals)
- Number of epochs (e.g. 30 epochs)
- Epoch length (e.g. 1000 generations)
- Tail function set (e.g. Plus, Minus, Multiply, Sine, Cosine)
- Maximum number of features (e.g. 10 features)
- Maximum depth of tree-like expressions (e.g. 7 levels)



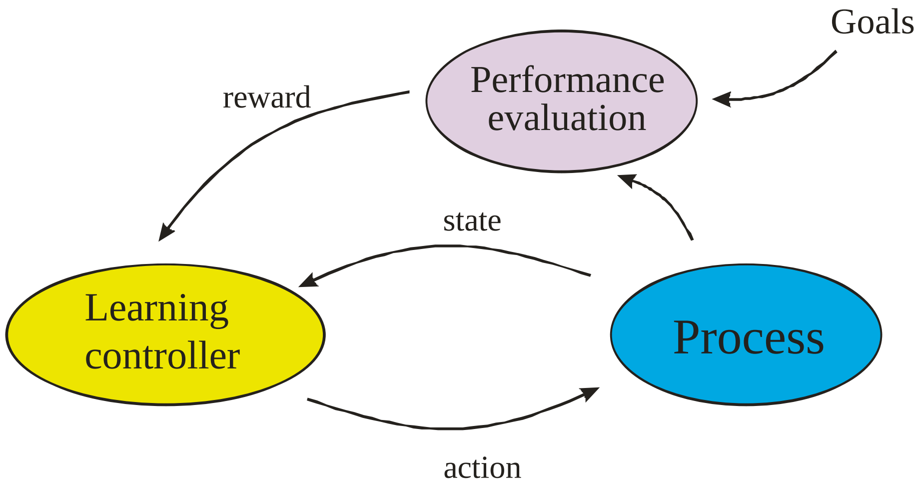
Model Identification – Outline

- Symbolic regression (SR)
  - Single Node Genetic Programming (SNGP)
  - Multi-Gene Genetic Programming (MGGP)
- Constructing models of the system using SR
  - State-space models
  - Input–output models (NARX, nonlinear autoregressive with exogenous input)
- Control using SR models
  - Reinforcement learning (RL) framework
- Data selection
  - Identification of informative samples from a large set collected in a long-term scenario

E. Derner, J. Kubalík, and R. Babuška. **Data-driven Construction of Symbolic Process Models for Reinforcement Learning.** In 2018 IEEE International Conference on Robotics and Automation (ICRA), 5105–5112, Brisbane, Australia.

E. Derner, J. Kubalík, and R. Babuška. **Reinforcement Learning with Symbolic Input–Output Models.** In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 3004–3009, Madrid, Spain.

Reinforcement Learning (RL)



Goal:

Learn a control strategy (policy) so that the sum of rewards over time is maximal.

Reinforcement Learning (RL) – Theoretical Background

- Nonlinear model

$$x_{k+1} = \underline{f}(x_k, u_k)$$

- x_k ... current state
- u_k ... current input
- x_{k+1} ... next state

- Reward function

$$r_{k+1} = \rho(x_k, u_k, x_{k+1})$$

- Bellman equation (value function, V-function)

$$\hat{V}^*(x) = \max_{u \in \mathcal{U}} \left[\rho\left(x, \pi(x), \underline{f}(x,u)\right) + \gamma \hat{V}^*\left(\underline{f}(x,u)\right) \right]$$

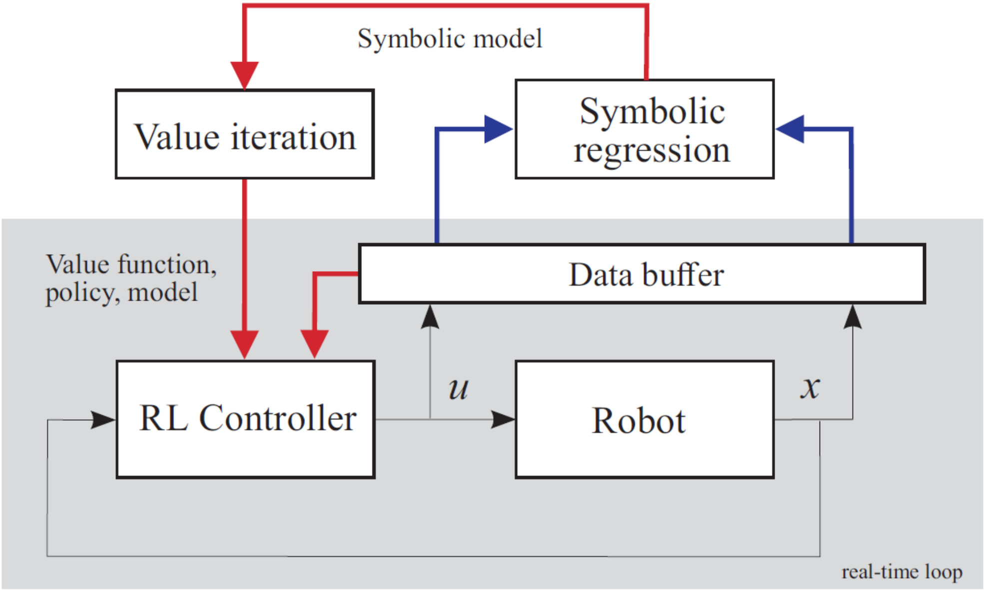
- Optimal action

$$u = \underset{u' \in U}{\operatorname{argmax}} \left[\rho(x, u', \underline{f}(x,u')) + \gamma V(\underline{f}(x,u')) \right]$$

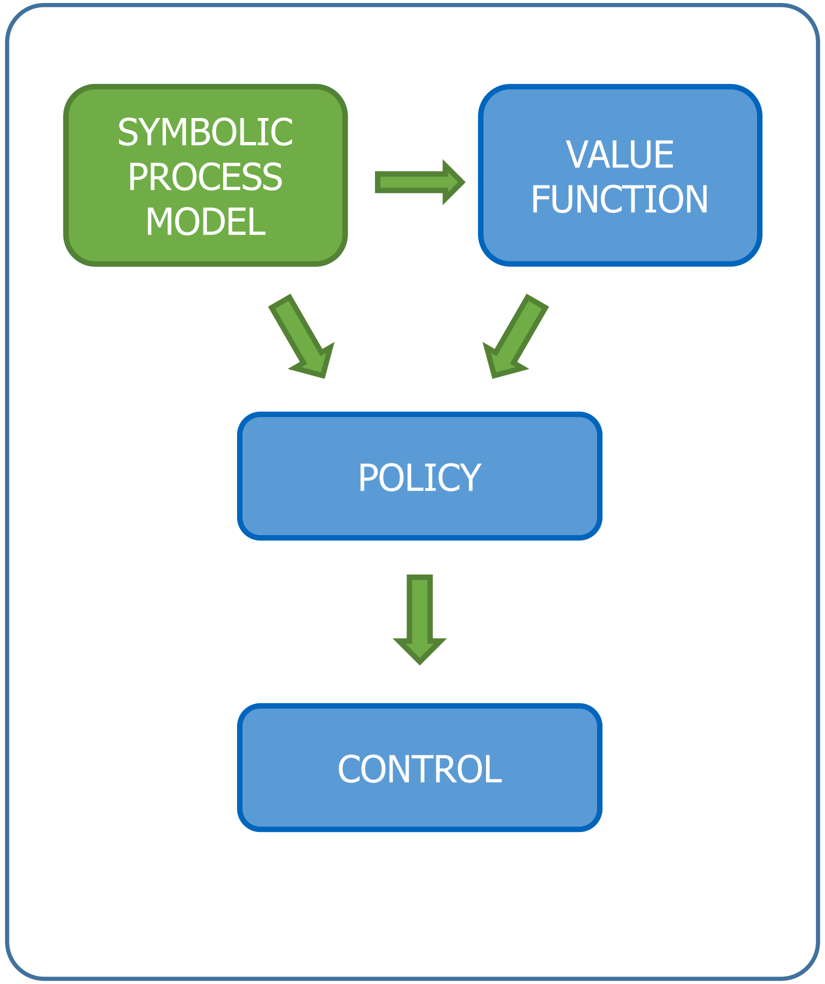
- γ ... discount factor

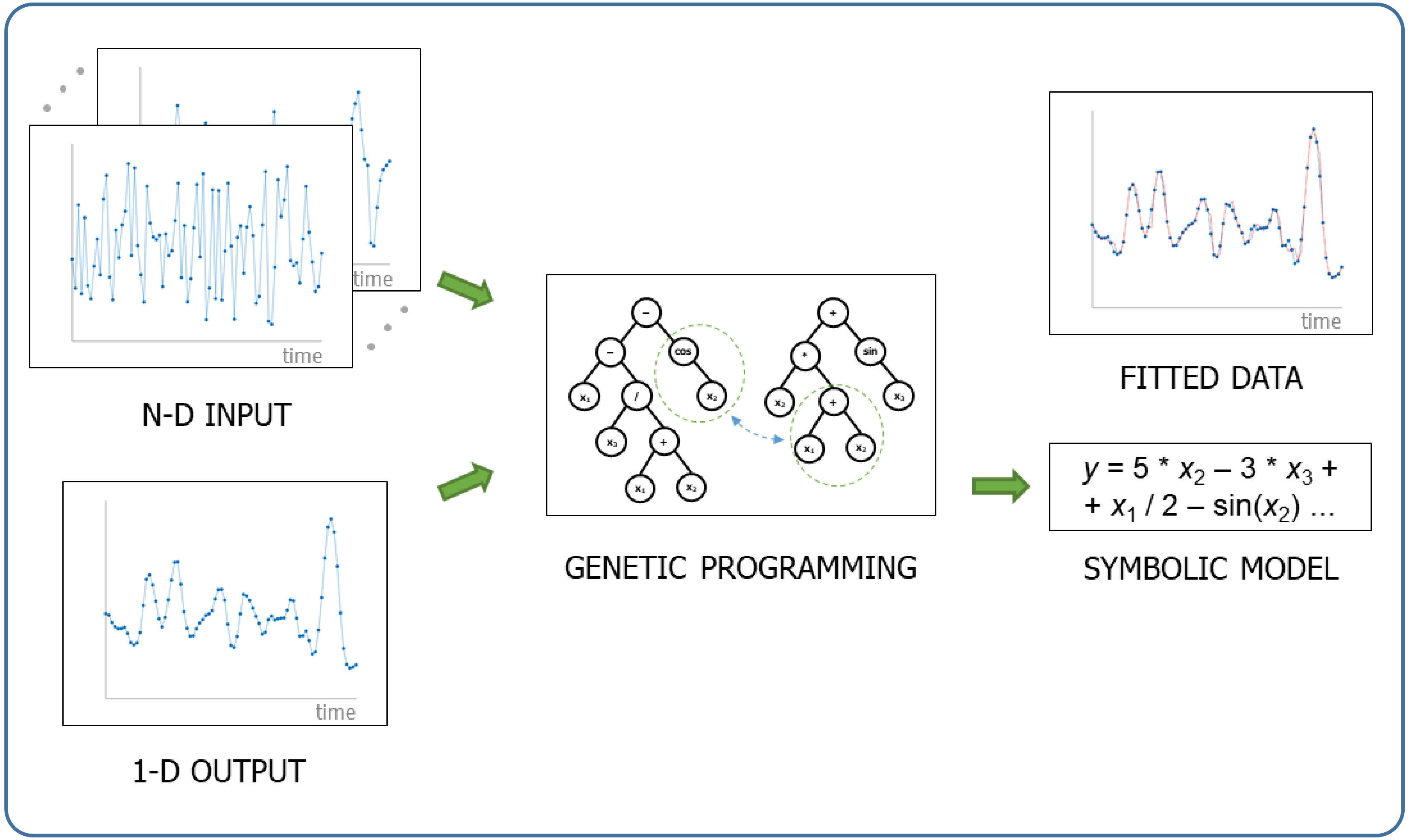
Model-Based RL Scheme

- Control loop and data logging in the buffer run in real time
- Symbolic regression and value iteration are computed offline in a parallel process
- Sample-efficient methods to construct interpretable analytic model from data
- Application in self-learning control
- Limited amount of data available
- Exploration is costly (safety, wear)
- Inclusion of prior knowledge



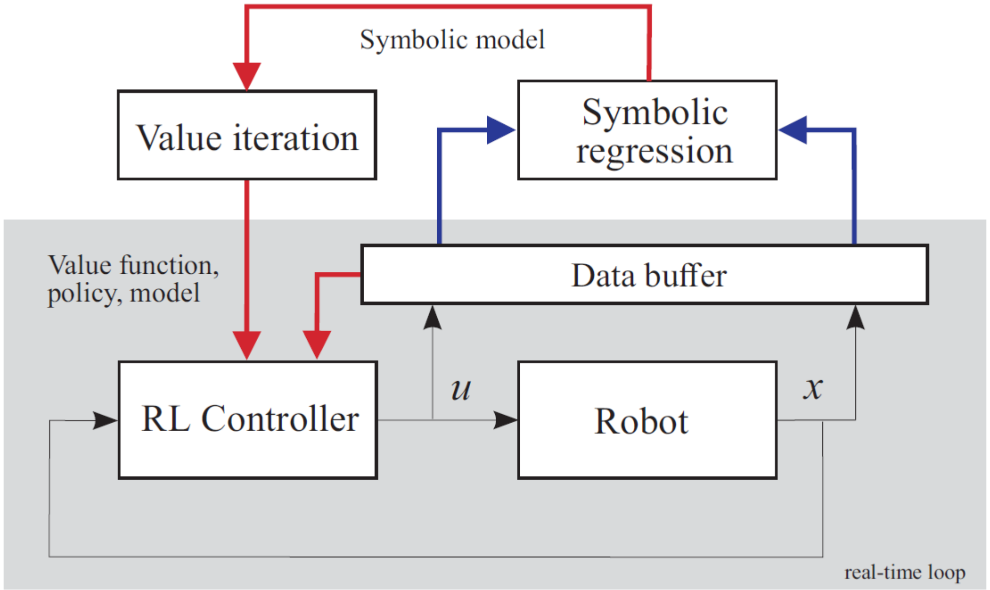
Symbolic Regression for RL – State-Space Models





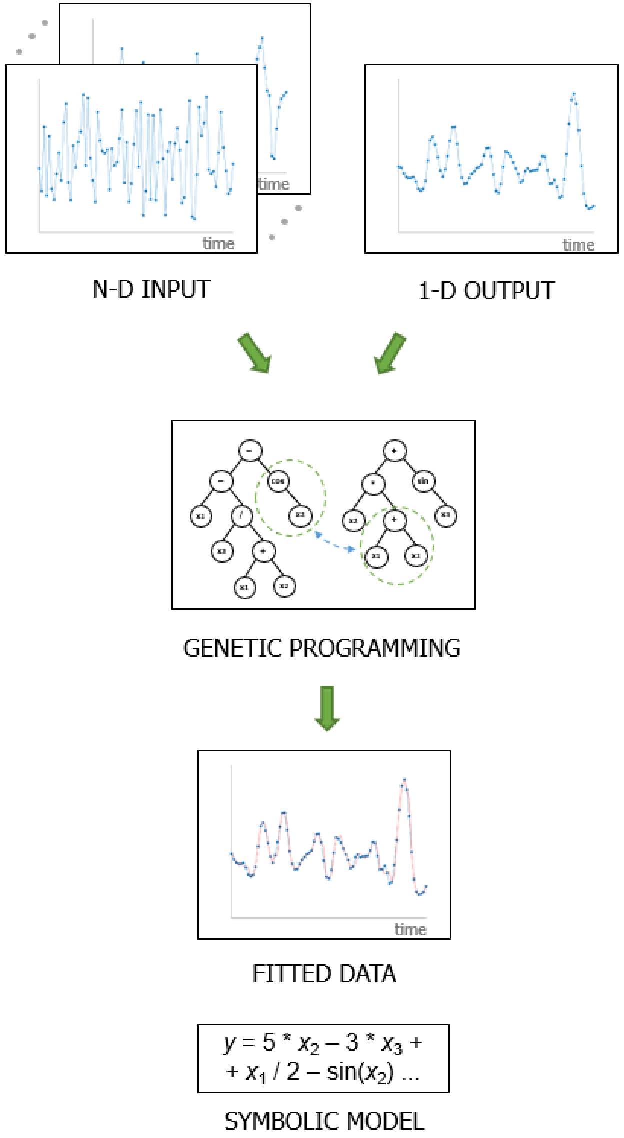
Model-Based RL with Symbolic Regression – Motivation

- RL agent optimizes its behavior by interacting with the environment
- The goal is to find an optimal policy maximizing the long-term cumulative reward
- RL can work in a completely model-free fashion
- The absence of a model requires a lot of interaction with the system, which is costly and many real systems cannot withstand it
- To speed up learning, we propose to use symbolic regression to find process models of unknown systems



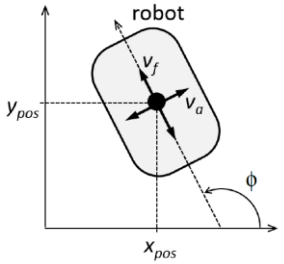
Problem Statement

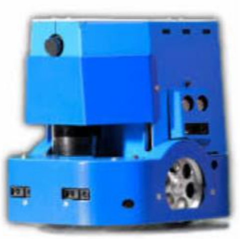
- SR is used to estimate the state-transition function of the system
- Given a set of training samples:
  - Multidimensional inputs
  - Known outputs
- Genetic programming is used to form a model composed of features represented as trees
- User-defined parameters of SR
  - Functions used in the inner nodes of the trees
  - Depth of the trees
  - Number of features

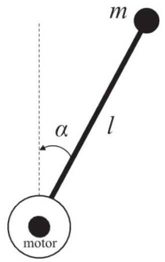


Experiments

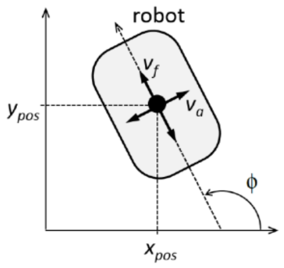
- Simulated experiments to evaluate the method for different number of features and various sizes of training sets
  - Mobile robot
  - Inverted pendulum
- Accurate analytic models can be found even for small training sets
  - Only tens of samples
  - Generated using the Euler approximation of the physical process model
- Real-world experiments
  - Inverted pendulum lab setup
  - Analytic process models used within a RL controller to perform the swing-up task

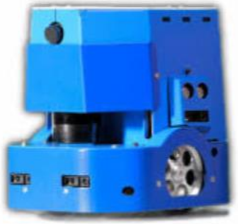






Mobile Robot – Illustrative Example





Continuous-time dynamics

$$\dot{x}_{pos} = v_f \cos(\phi),$$
$$\dot{y}_{pos} = v_f \sin(\phi),$$
$$\dot{\phi} = v_a\,.$$

x_{pos} ... pose x-coordinate

y_{pos} ... pose y-coordinate

ϕ ... pose angle

v_f ... linear („forward“) velocity

v_a ... angular velocity

Discrete-time dynamics

$$x_{pos,k+1} = x_{pos,k} + 0.05\, v_{f,k} \cos(\phi),$$
$$y_{pos,k+1} = y_{pos,k} + 0.05\, v_{f,k} \sin(\phi),$$
$$\phi_{k+1} = \phi_k + 0.05\, v_{a,k}\,.$$

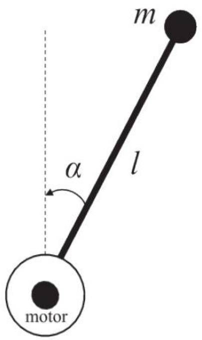
Euler approximation

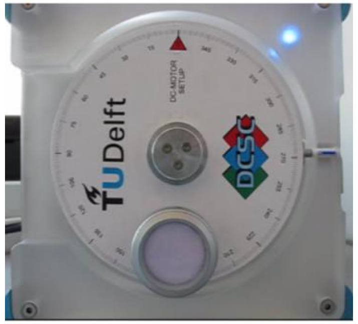
Example of an analytic model found by SR

$$\hat{x}_{pos,k+1} = 1.0 x_{pos,k} + 0.0499998879\, v_{f,k} \cos(\phi_k)$$
$$\hat{y}_{pos,k+1} = 1.000000023\, y_{pos,k} + 0.0500000056\, v_{f,k} \sin(\phi_k) + \\ + 0.0000000191$$
$$\hat{\phi}_{k+1} = 0.9999982931\, \phi_k + 0.0500000536\, v_{a,k} - \\ - 0.0000059844$$

Real Inverted Pendulum System

$$\ddot{\alpha} = \frac{1}{J} \cdot \left(\frac{K}{R} u - m g l \sin(\alpha) - b \dot{\alpha} - \frac{K^2}{R} \dot{\alpha} - c \, \mathrm{sign}(\dot{\alpha}) \right)$$





$J = 1.7937 \times 10^{-4}\ \mathrm{kg\ m^2}$

$K = 0.0536\ \mathrm{N\ m\ A^{-1}}$

$R = 9.5\ \Omega$

$m = 0.055\ \mathrm{kg}$

$g = 9.81\ \mathrm{m\ s^{-2}}$

$l = 0.042\ \mathrm{m}$

$b = 1.94 \times 10^{-5}\ \mathrm{N\ m\ s\ rad^{-1}}$

$c = 8.5 \times 10^{-4}\ \mathrm{kg\ m^2\ s^{-2}}$

α ... angle $[\mathrm{rad}]$

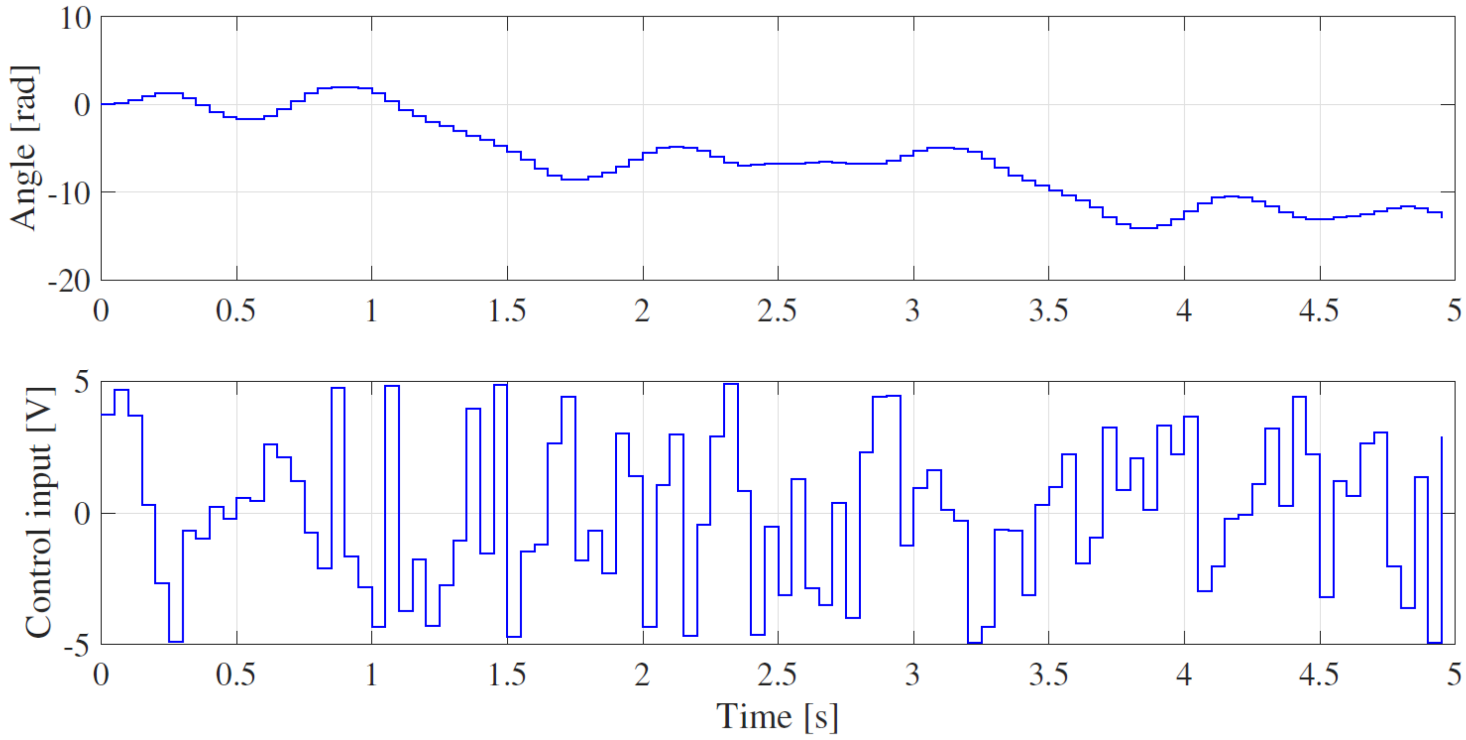
$\dot{\alpha}$ ... angular velocity $[\mathrm{rad\ s^{-1}}]$

$\ddot{\alpha}$ ... angular acceleration $[\mathrm{rad\ s^{-2}}]$

u ... voltage $[\mathrm{V}]$ – control input

Real Inverted Pendulum Swing-Up

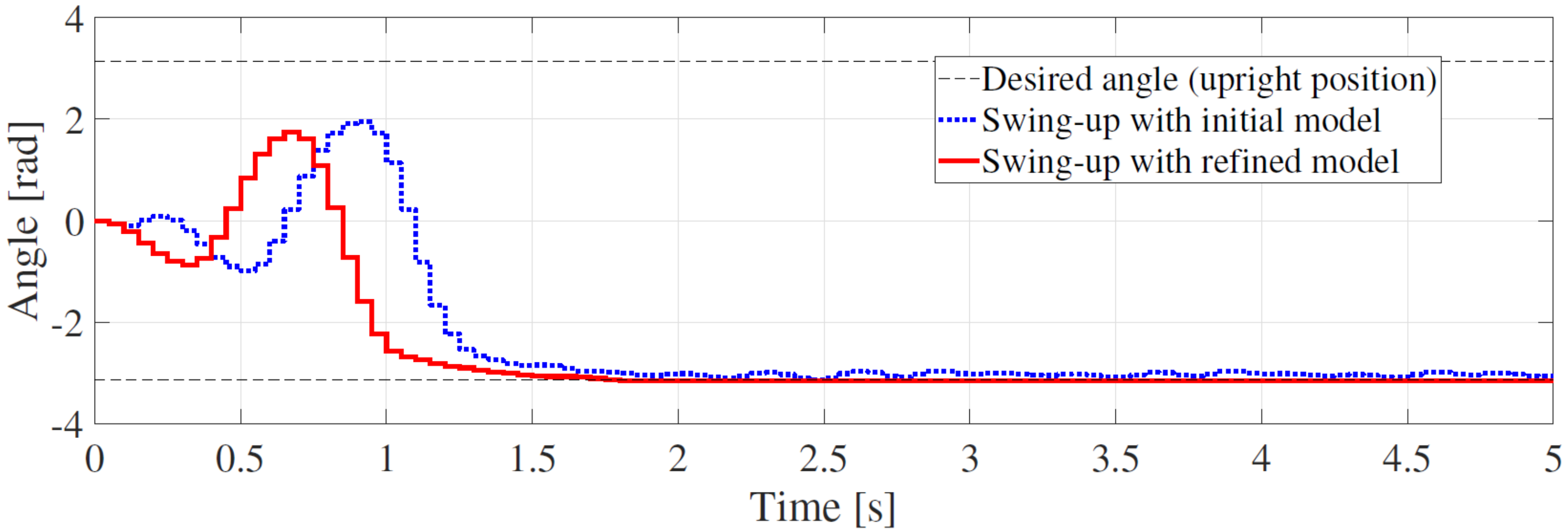
- Under-actuated swing-up task (limited voltage, cannot swing up at once)
- Training data were collected while applying random input to the system



Real Inverted Pendulum Swing-Up

- Only 5 seconds of the random interaction with a sampling period $T_s = 0.05$ s is sufficient to find a symbolic process model that can be used to perform the swing-up task successfully

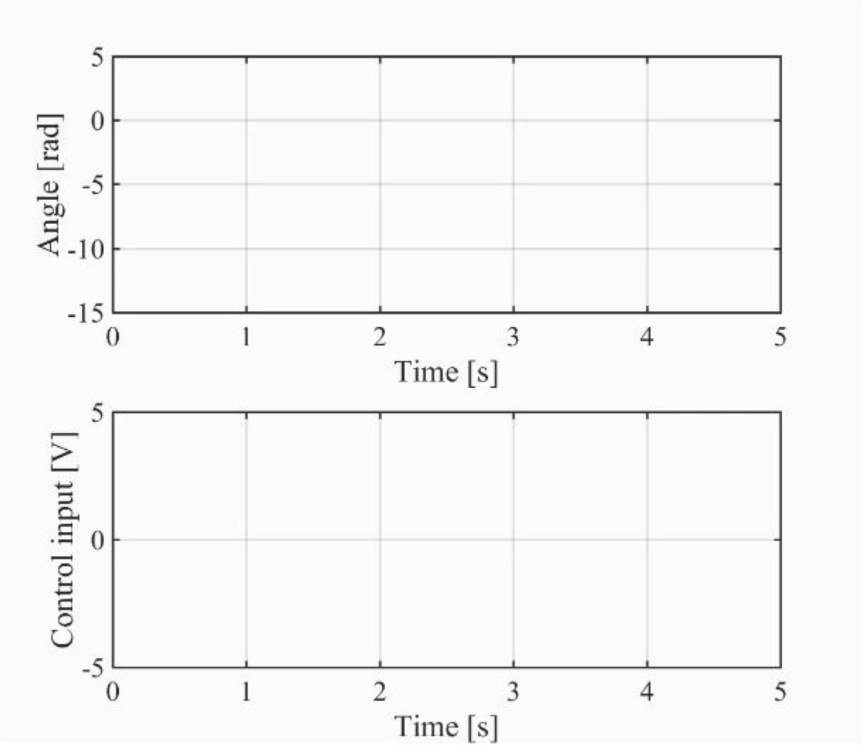
- Data from several executions of the swing-up task were collected and used together with the initial data set to train the refined model, which shows even better performance



Experiment – Pendulum Swing-Up

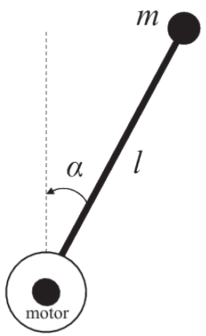
Control task: Make the underactuated inverted pendulum point up.

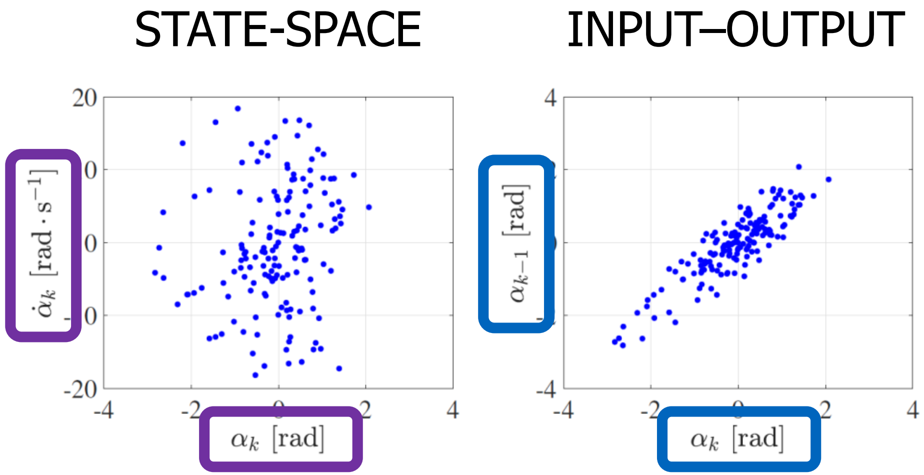
Collection of training data: random input



Input–Output (NARX) Models

- Motivation: the whole state is often not measurable, needs to be approximated
- $\underbrace{\hat{y}_{k+1}}_{\text{Predicted output}} = f\big(\underbrace{y_k, y_{k-1}, \ldots, y_{k-n_y+1}}_{\text{Past outputs}}, \underbrace{u_k, u_{k-1}, \ldots, u_{k-n_u+1}}_{\text{Past inputs}}\big)$





Experiment – Hopping Robot

Body:

$$\ddot{x}_1 = \frac{\kappa \Delta x}{m_1 l}(L_0 - l)$$

$$\ddot{y}_1 = -g + \frac{\kappa \Delta y}{m_1 l}(L_0 - l)$$

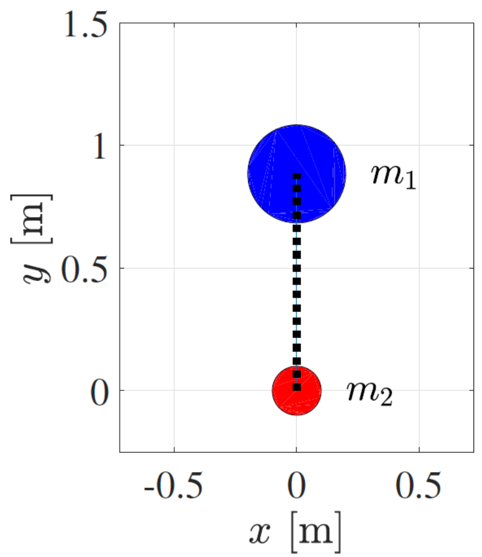
Spring length:

$$l = \sqrt{\Delta x^2 + \Delta y^2}$$

Foot:

$$\ddot{x}_2 = \frac{\kappa \Delta x}{m_2 l}(L_0 - l) - \frac{1}{m_2} b \dot{x}_2$$

$$\ddot{y}_2 = -g + \frac{\kappa \Delta y}{m_2 l}(L_0 - l) - \frac{1}{m_2} b \dot{y}_2$$



m_1, m_2 ... body and foot mass, connected by a spring

κ ... variable spring constant

g ... gravitational acceleration

L_0 ... equilibrium spring length

l ... actual spring length

b ... damping coefficient

Simplification of the problem statement:

$x_1, x_2 = 0$ ... x-coordinate is fixed

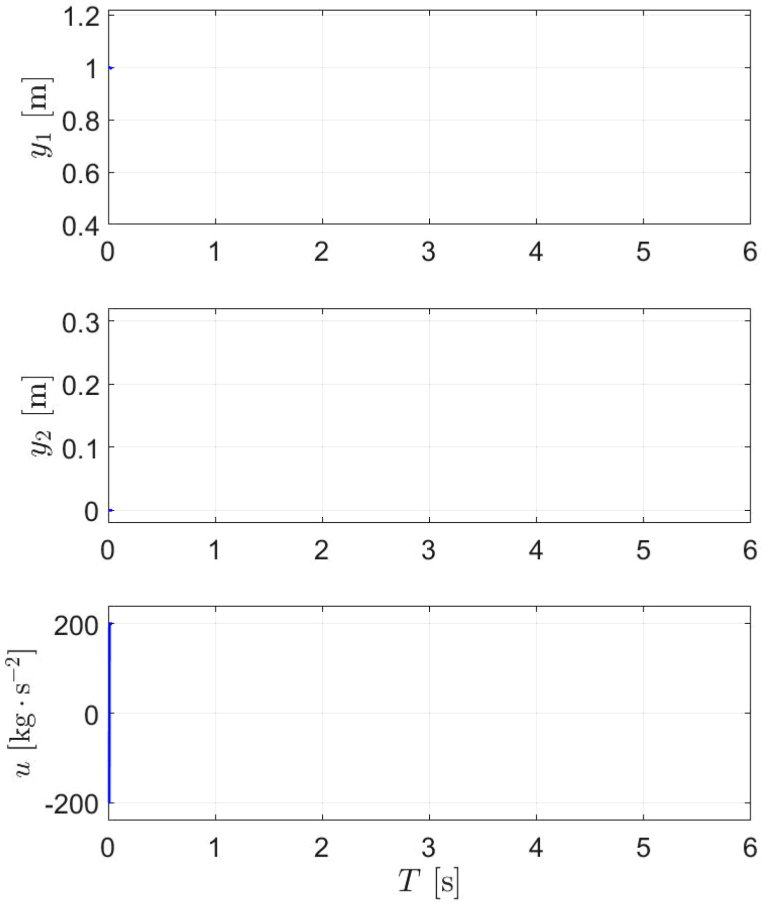
Control input u:

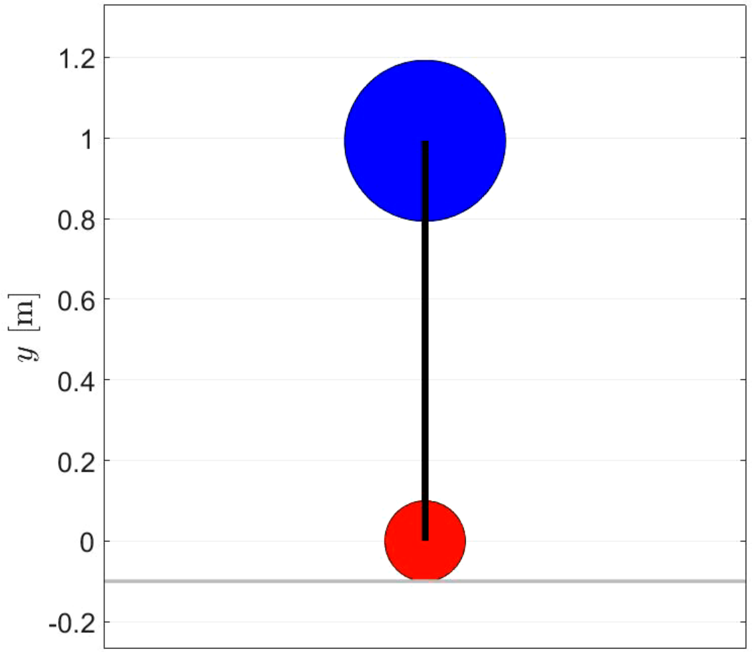
$\kappa = \kappa' + u$

κ' ... nominal spring constant

Experiment – Hopping Robot

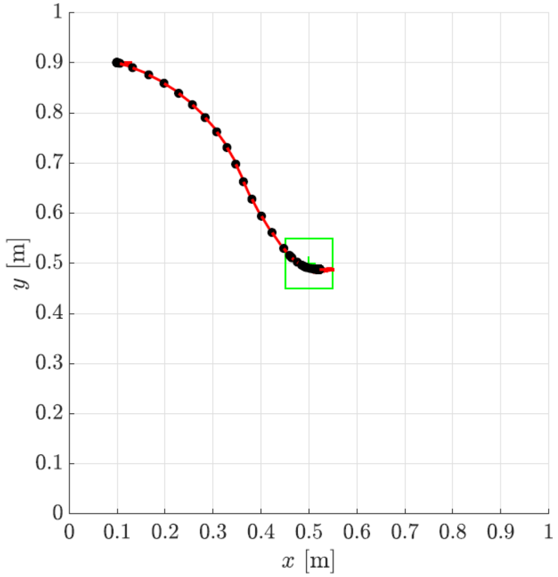
Control task: Keep the robot hopping.



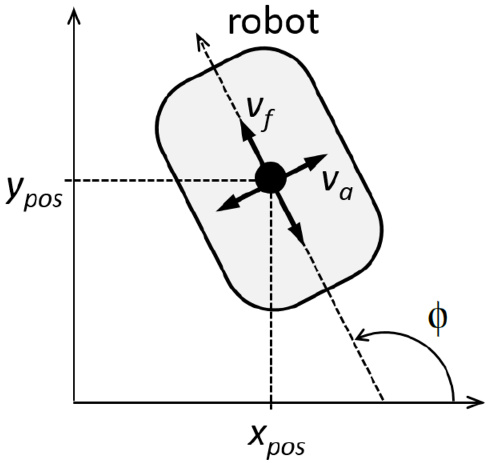


Data Selection

- A robot collects a large amount of data during its long-term operation
- Only some data samples are informative
- The method iteratively adds samples, starting with a very small data set
- In every iteration, a set of models of the robot's dynamics is constructed
- The proposed sample selection method is based on the prediction error of the models from the previous iteration



Experiment – Mobile Robot



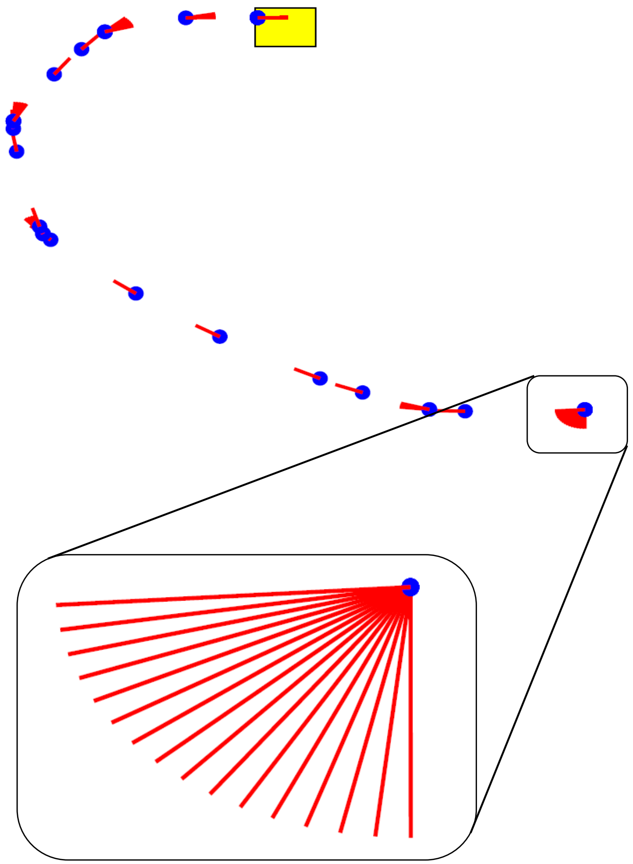
Mechanistic model:

$$\dot{x}_{pos} = v_f \cos(\phi)$$
$$\dot{y}_{pos} = v_f \sin(\phi)$$
$$\dot{\phi} = v_a$$

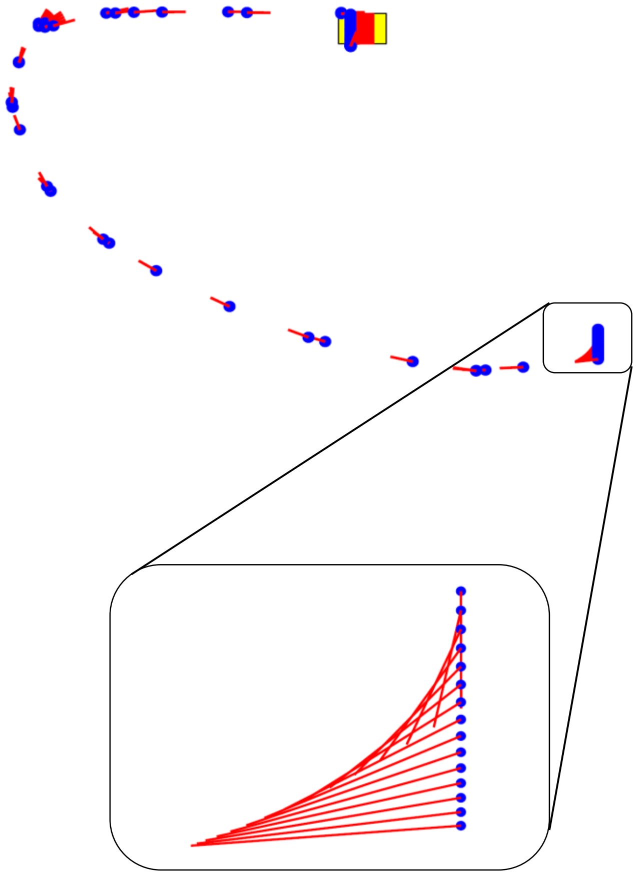
- Mechanistic model correctly represents the physics, but is inaccurate as a prediction model (actuator nonlinearities)
- Data-driven model constructed via symbolic regression is accurate, but does not necessarily respect the physical constraints

Experiment – Mobile Robot

Motion planning with mechanistic model

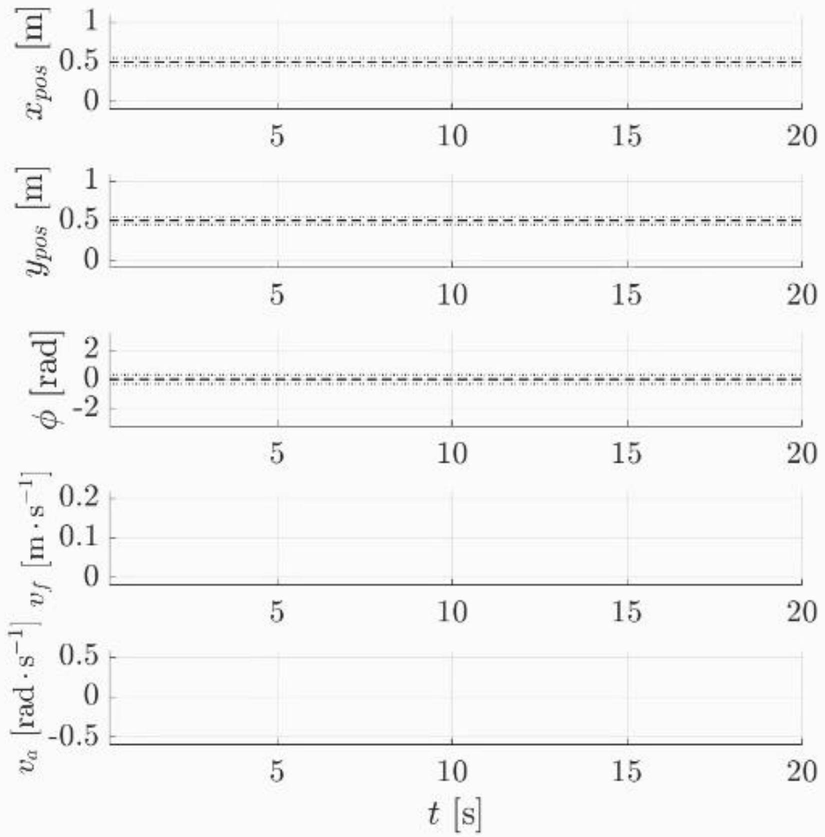


Motion planning with data-driven model



Experiment – Mobile Robot

17 training samples



Conclusions

- Genetic programming methods allow to automatically construct analytic models
- Such models can be easily plugged into other algorithms and facilitate further analysis
- In a long-term scenario, a robot collects a large amount of data
- If the data are selected in an informed way, only a few samples are necessary to train a precise model of the robot's dynamics
- Experimental evaluation has shown that a model trained on only 24 samples can be used in a RL framework to perform the control task successfully

Future Work

- Reinforcement learning powered by symbolic regression
    - Analytic model of the system dynamics
    - Symbolic V-function
    - Completely automated construction of an RL controller
    - Evaluate on the inverted pendulum and on the mobile robot, then on high-dimensional systems
- Data selection in long-term scenarios
    - Novel algorithm for sample selection with outlier detection (data loss, sensor faults)
    - Automated data set maintenance (removal of wrong data)
    - Real-world long-term autonomy experiment

Thank you for your attention!

http://people.ciirc.cvut.cz/derneeri