Data-Efficient Methods for Model Learning and Control in Robotics

Erik Derner

erik.derner@cvut.cz

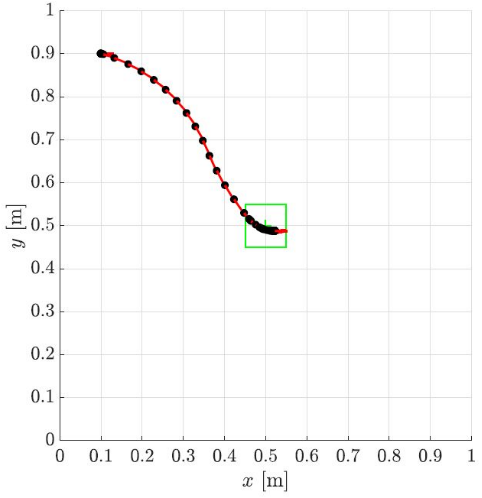
Czech Institute of Informatics, Robotics, and Cybernetics

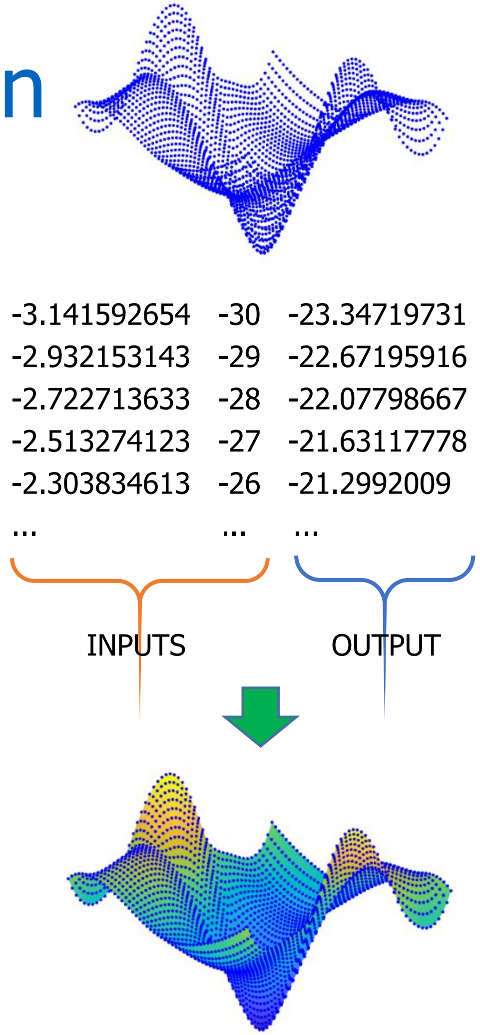
Czech Technical University in Prague, Czech Republic

28 March 2022

Model Learning & Control Using Symbolic Regression

- Symbolic regression (SR) automatically finds *accurate analytic models* fitting measured data
- Genetic operators (mutation) are applied to tree-like structures representing the models to *evolve them*, gradually improving their accuracy
- *Informed sample selection* improves efficiency: only 24 data samples needed to learn a model allowing for accurate control of a mobile robot
- *Formal constraints* allow for incorporating prior knowledge about the physical system and evolving accurate & physically meaningful models through bi-objective optimization



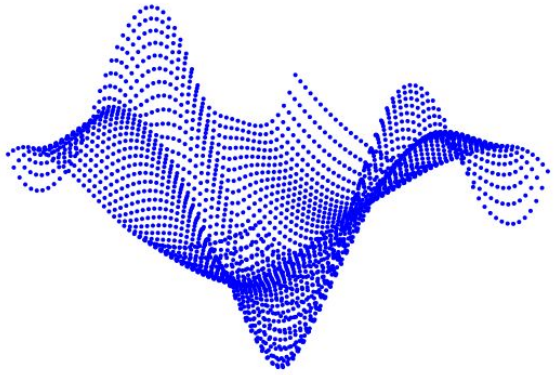


Derner, E., Kubalík, J., Ancona, N., & Babuška, R. (2020). Constructing Parsimonious Analytic Models for Dynamic Systems via Symbolic Regression. *Applied Soft Computing*, 94, 106432. [Q1 journal]

Derner, E., Kubalík, J., & Babuška, R. (2021). Selecting Informative Data Samples for Model Learning Through Symbolic Regression. *IEEE Access*, 9, 14148-14158. [Q1 journal]

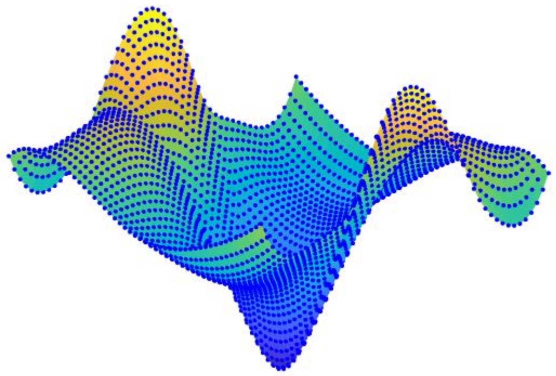
Symbolic Regression – Motivation

- 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



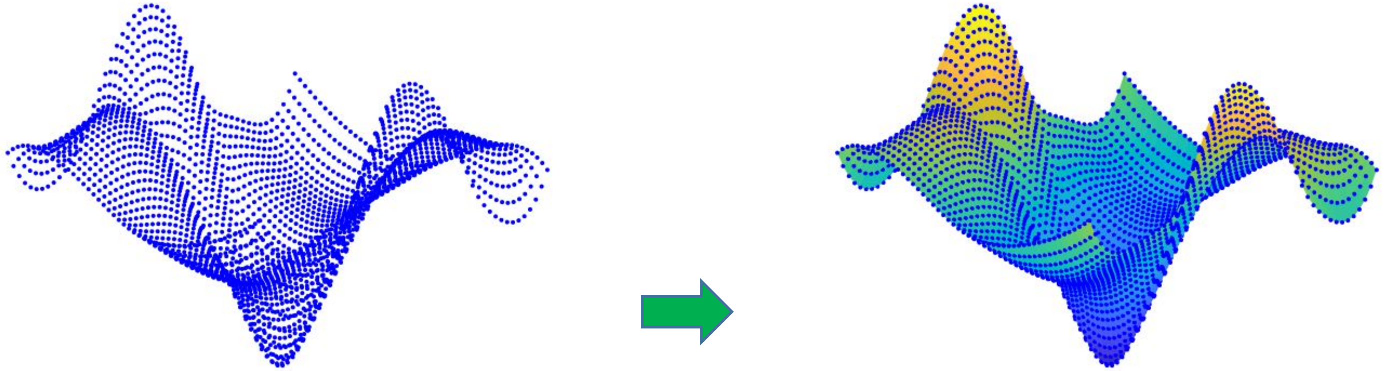
-3.141592654	-30	-23.34719731
-2.932153143	-29	-22.67195916
-2.722713633	-28	-22.07798667
-2.513274123	-27	-21.63117778
-2.303834613	-26	-21.2992009
...	...	...

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

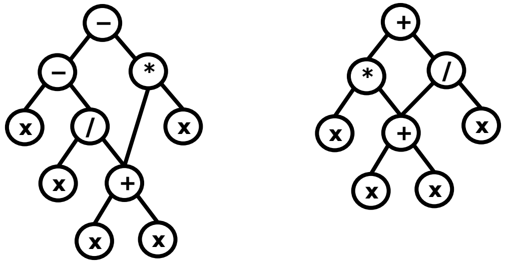


-3.141592654	-30	-23.34719731
-2.932153143	-29	-22.67195916
-2.722713633	-28	-22.07798667
-2.513274123	-27	-21.63117778
-2.303834613	-26	-21.2992009
…	…	…

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)

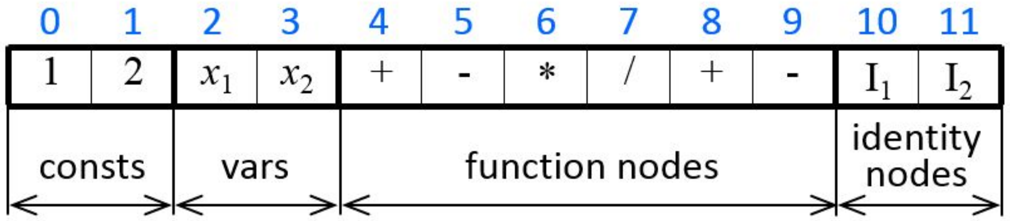
[3] M. Hinchliffe et al.: Modelling chemical process systems using a multi-gene genetic programming algorithm (1996)

[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)

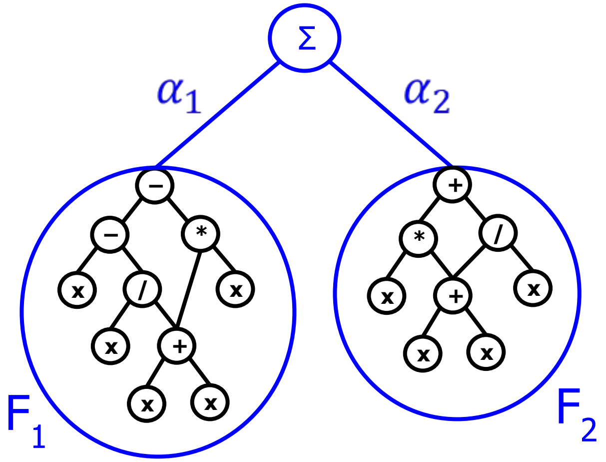
Single Node Genetic Programming (SNGP)

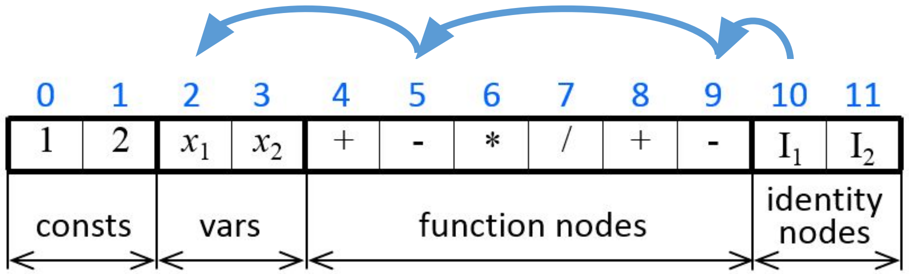
- Graph-based GP technique
- Evolves a population organized as an ordered linear array of individuals, each representing a single program node
- Program node types
  - Terminals – variables, constants
  - Functions
- Evolutionary process
  - SMUT – successor mutation
  - Acceptance rule – best fitness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



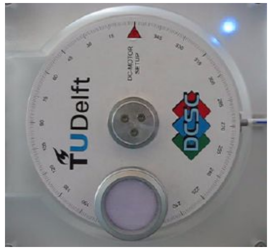


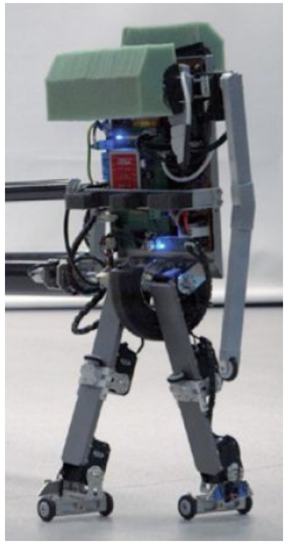
[6] J. Kubalík et al.: Hybrid single node genetic programming for symbolic regression (2016)

Main 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



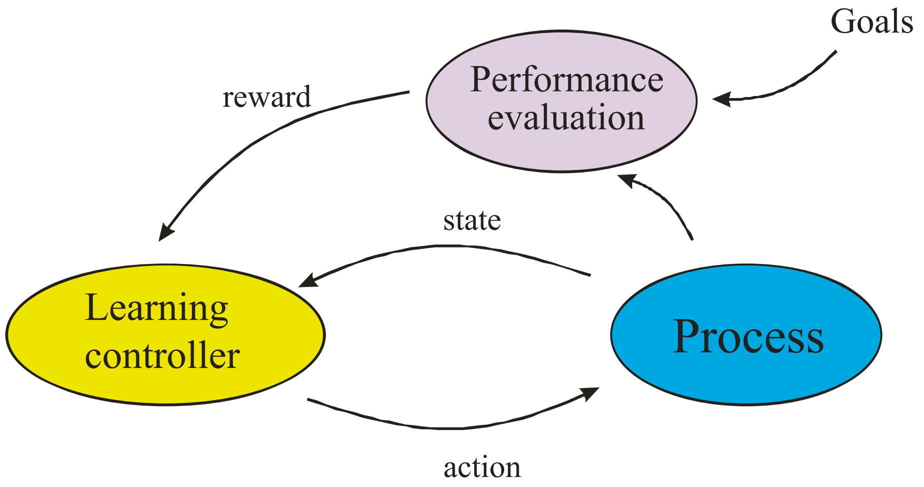


- Symbolic regression (SR)
    - Single Node Genetic Programming (SNGP)
    - Multi-Gene Genetic Programming (MGGP)
- Constructing models of the system using SR
    - State-space models
    - Input–output models (NARX, nonlinear autoregressive with exogenous input)
- Control using SR models
    - Reinforcement 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)

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

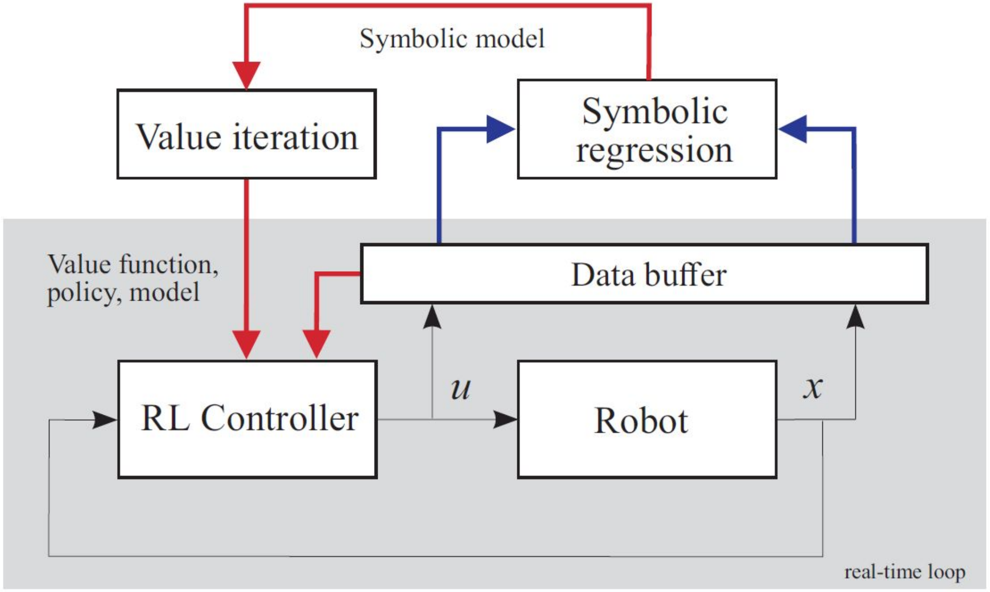
- Optimal action

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

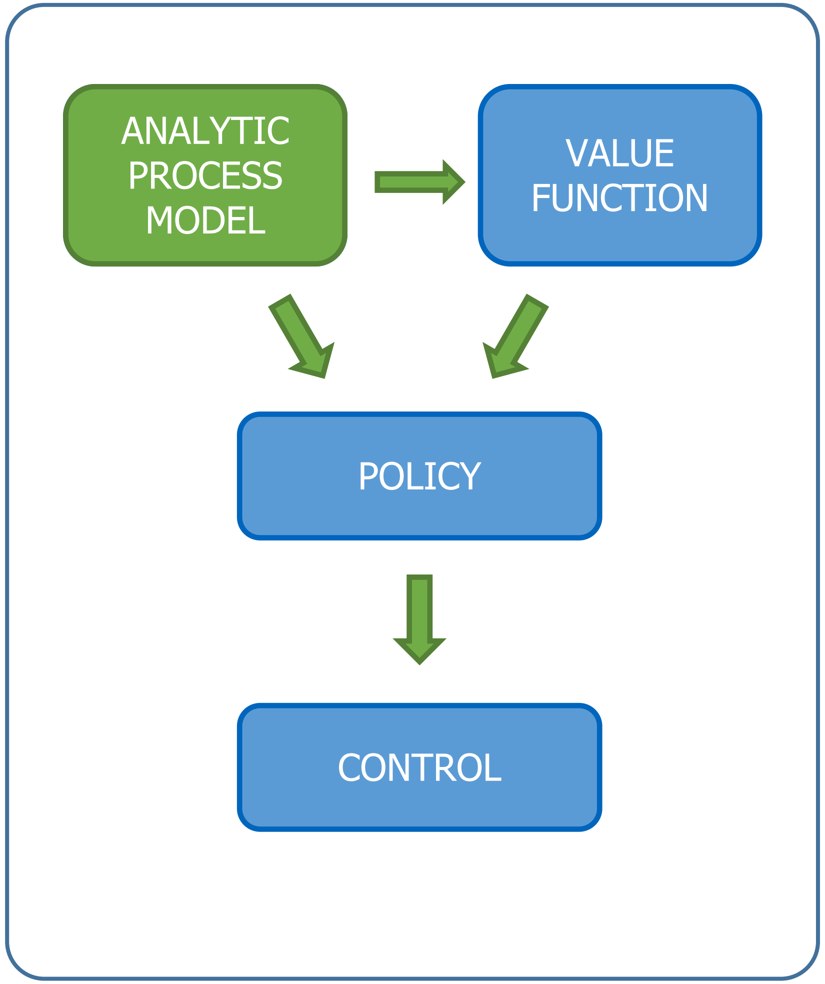
- γ ... discount factor

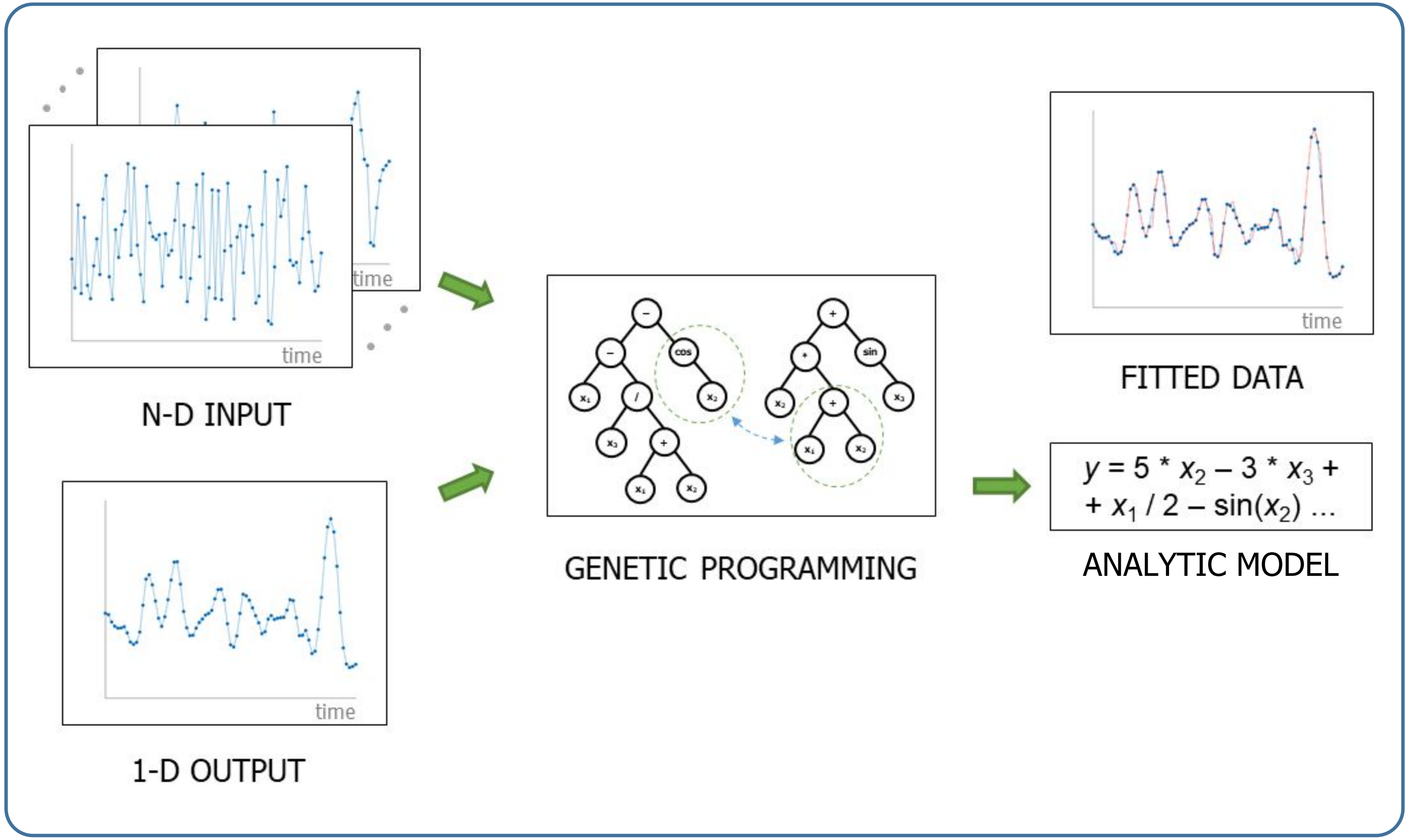
Model-Based RL Scheme

- 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



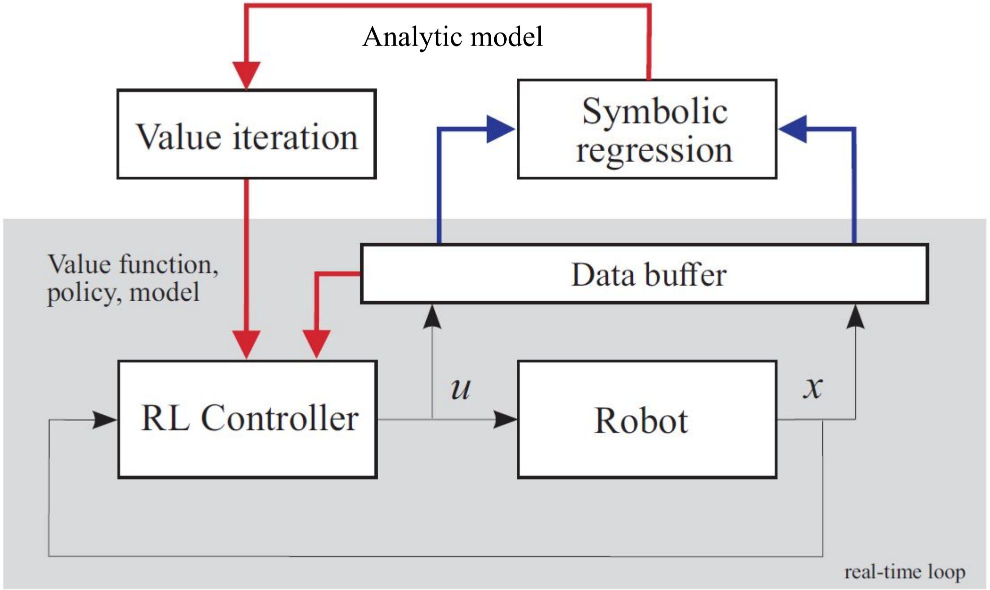
Symbolic Regression for RL – State-Space Models





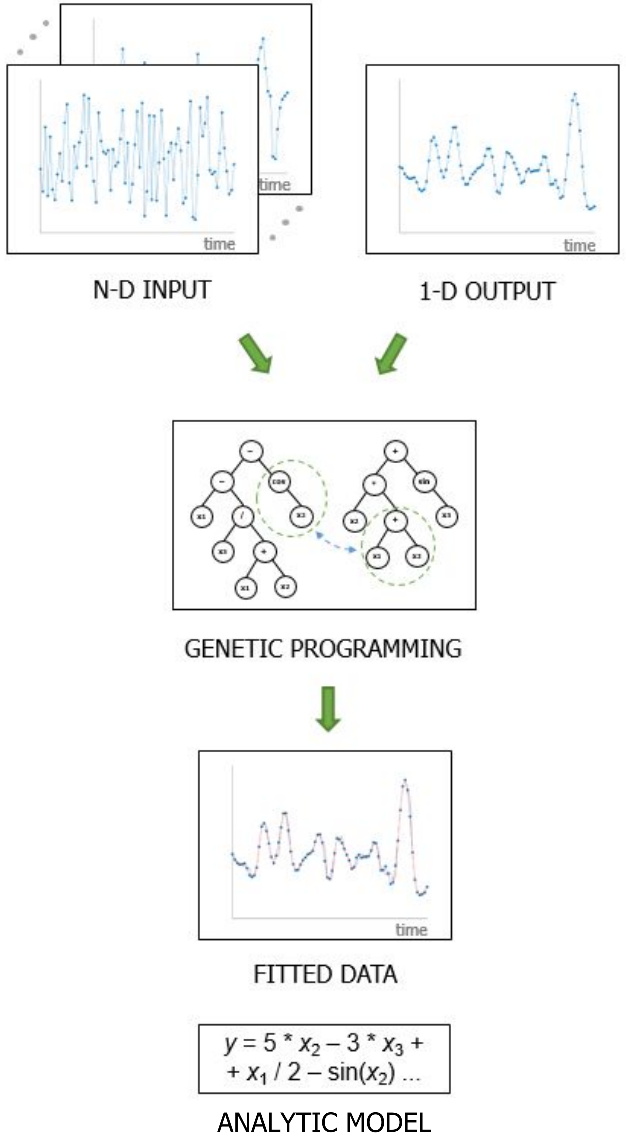
Model-Based RL with Symbolic Regression – Motivation

- 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



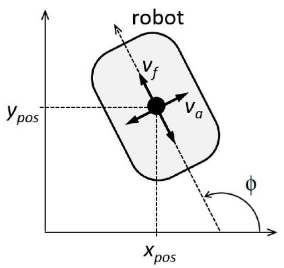
Problem Statement

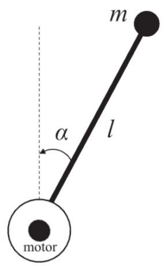
- SR is used to estimate the state-transition function of the system
- Given a set of training samples:
  - Multidimensional inputs
  - Known outputs
- Genetic programming is used to form a model composed of features represented as trees
- User-defined parameters of SR
  - Functions used in the inner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



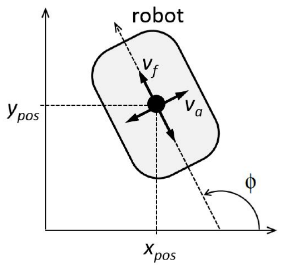
Experiments

- 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

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

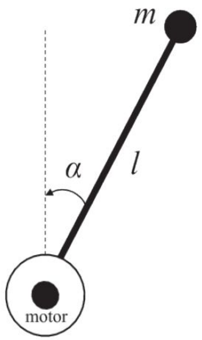
Euler approximation

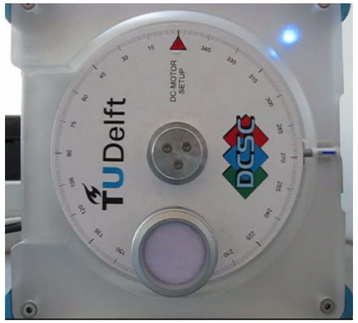
Example of an analytic model found by SR

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

Real Inverted Pendulum System

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





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

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

$R = 9.5\ \Omega$

$m = 0.055\ \mathrm{kg}$

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

$l = 0.042\ \mathrm{m}$

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

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

α ... angle $[\mathrm{rad}]$

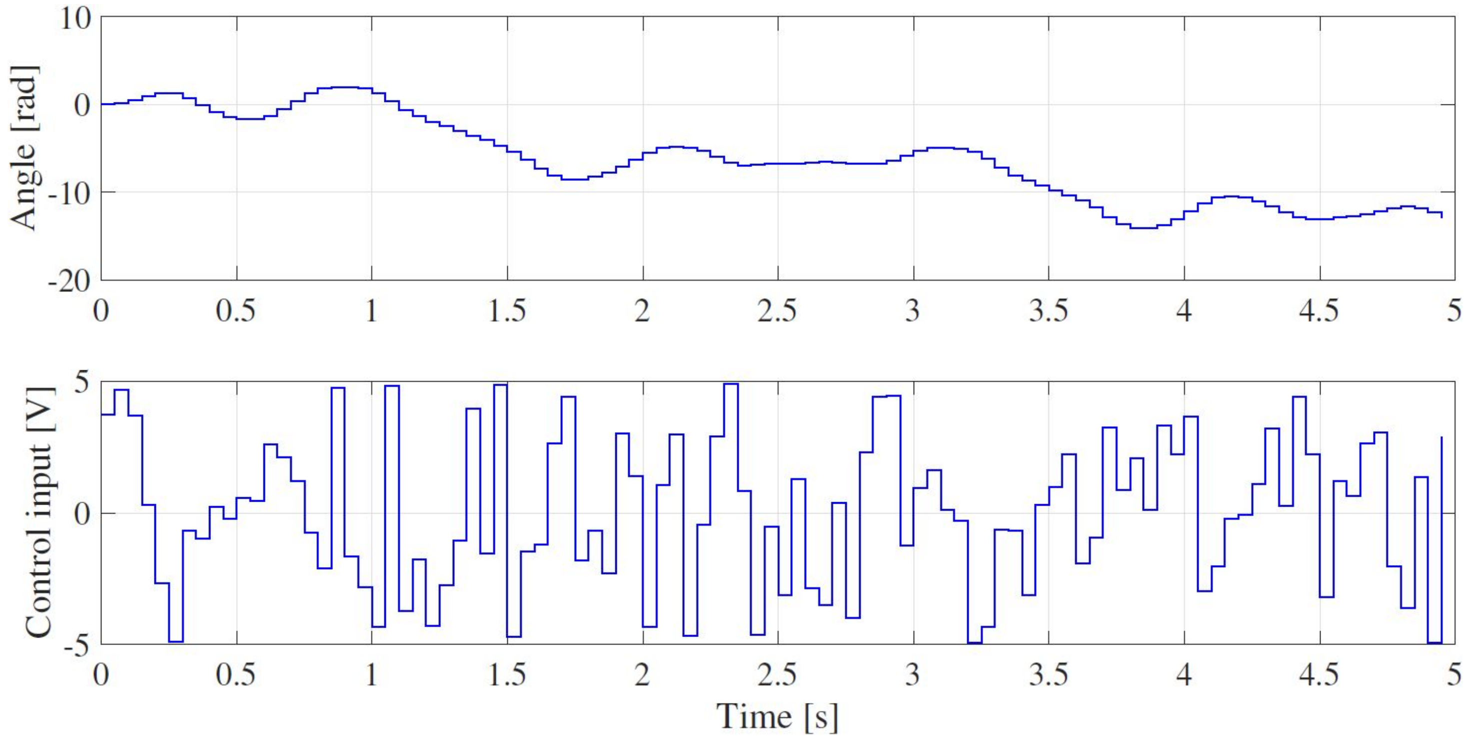
$\dot{\alpha}$ ... angular velocity $[\mathrm{rad\,s^{-1}}]$

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

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

Real Inverted Pendulum Swing-Up

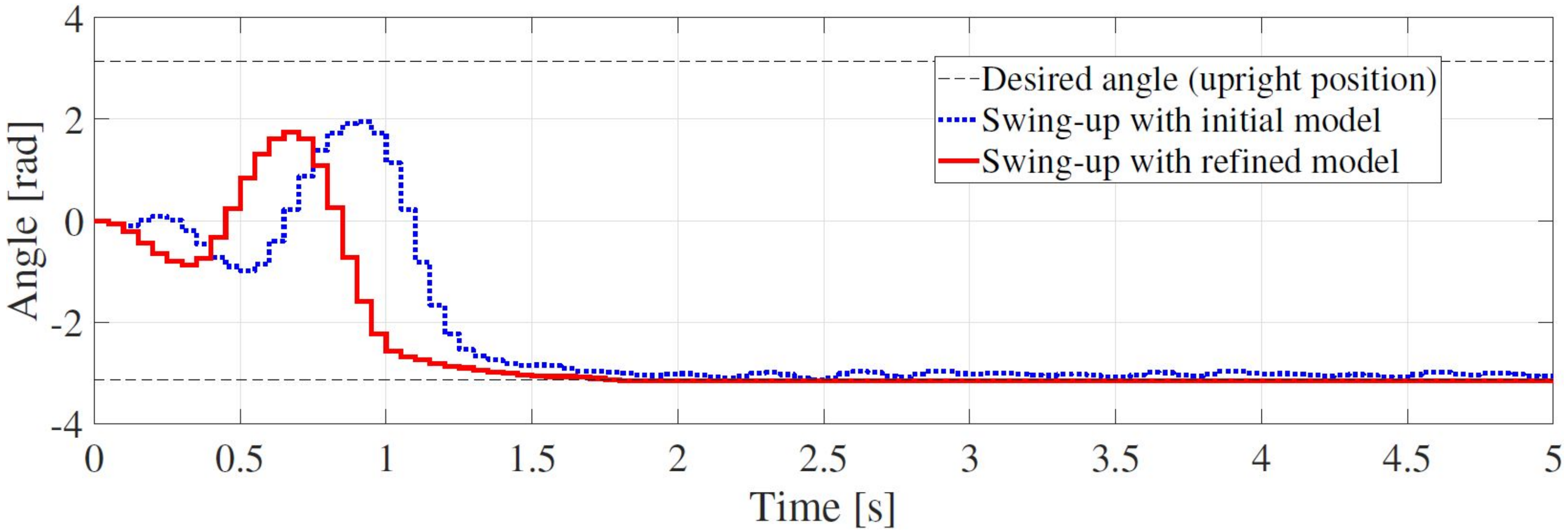
- Under-actuated swing-up task (limited voltage, cannot swing up at once)
- Training data were collected while applying random input to the system



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 an analytic process model that can be used to perform the swing-up task successfully

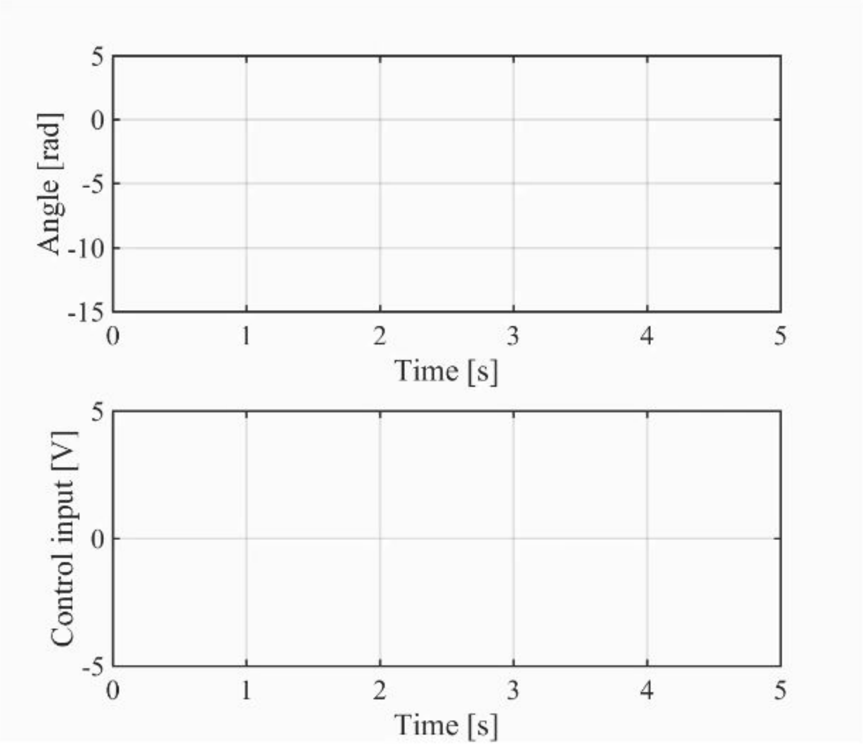
- Data from several executions of the swing-up task were collected and used together with the initial data set to train the refined model, which shows even better performance



Experiment – Pendulum Swing-Up

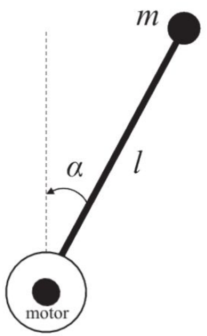
Control task: Make the underactuated inverted pendulum point up.

Collection of training data: random input

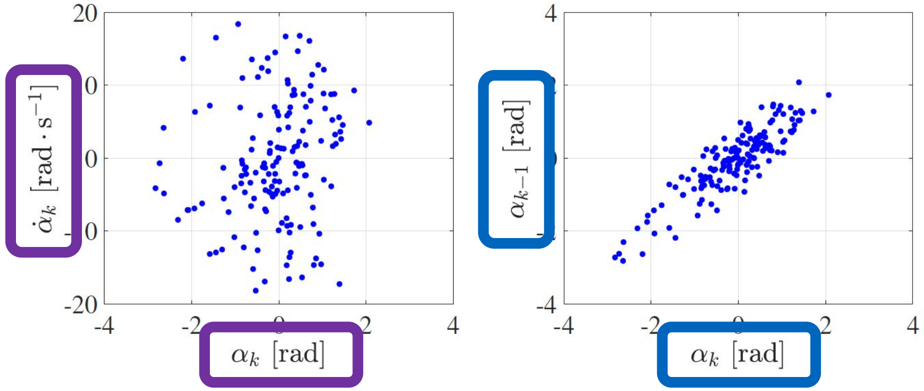


Input–Output (NARX) Models

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



STATE-SPACE INPUT–OUTPUT



Experiment – Hopping Robot

Body:

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

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

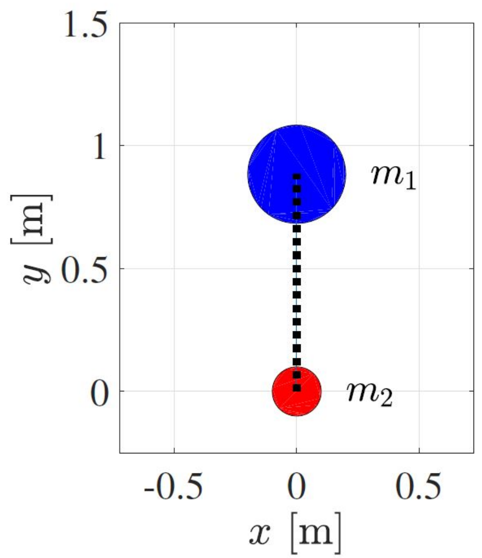
Spring length:

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

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

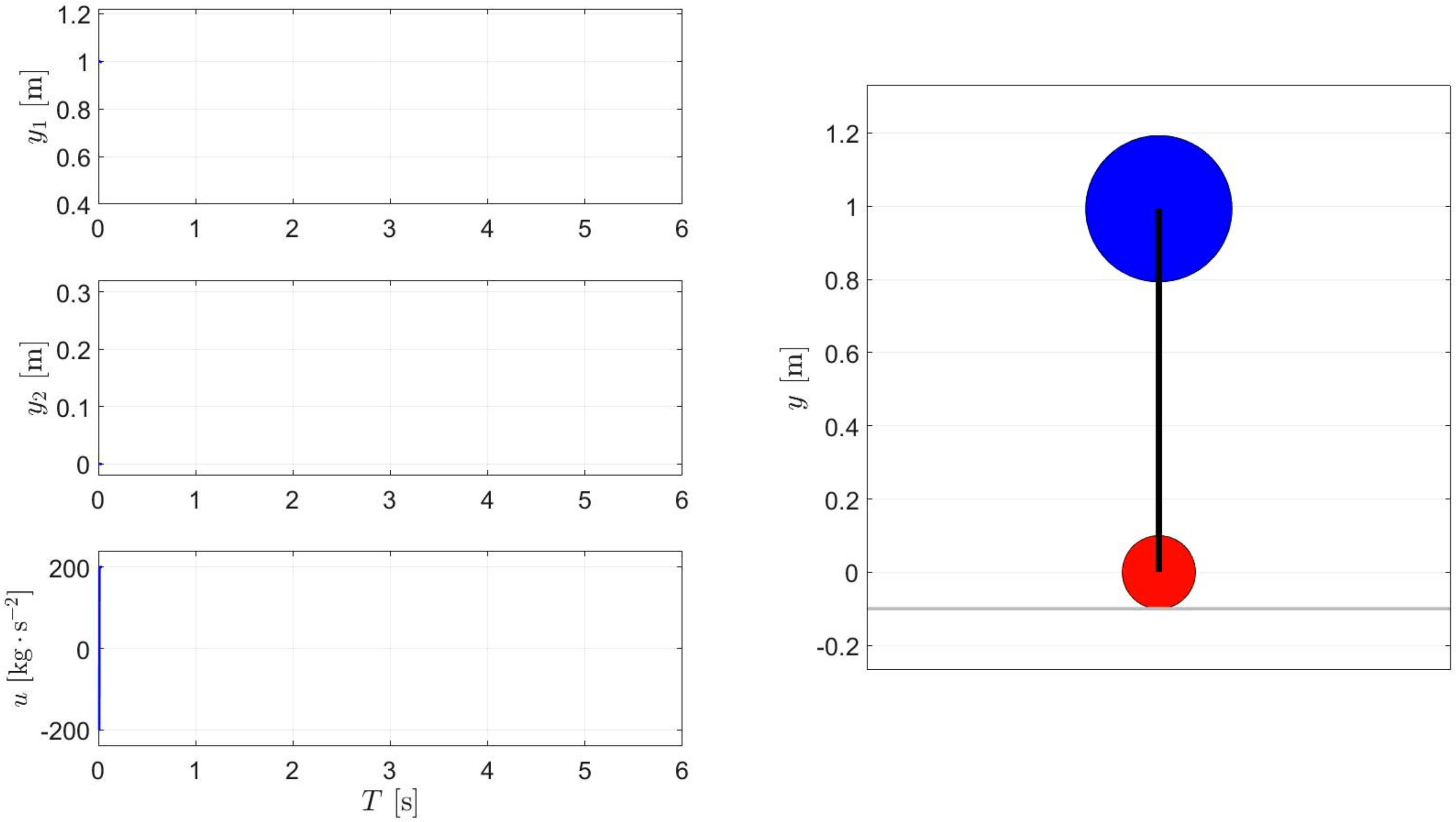
Control input u:

$\kappa = \kappa' + u$

κ' ... nominal spring constant

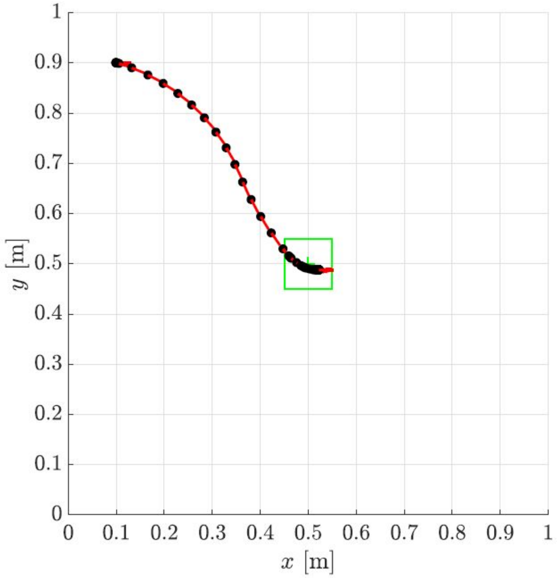
Experiment – Hopping Robot

Control task: Keep the robot hopping.



Model Learning with Sample Selection – Motivation

- 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



Model Learning with Sample Selection – Algorithm

- **Input:** sample-selection method, *Buffer*, *TestSet*, n_0, n_s, n_i, n_r
- $i \leftarrow 0$
- $\mathit{TrainingSet} \leftarrow S_{n_0}$ (first n_0 samples in *Buffer*)
- $\mathit{Buffer} \leftarrow \mathit{Buffer} \setminus S_{n_0}$

repeat

  $i \leftarrow i + 1$

  for each state variable **do**

- run n_r instances of SR to construct models f_r
- $f^* \leftarrow f_r$ with the lowest RMSE on *TestSet*
- $S \leftarrow n_s$ samples from *Buffer*, chosen by the sample-selection method
- $\mathit{TrainingSet} \leftarrow \mathit{TrainingSet} \cup S$
- $\mathit{Buffer} \leftarrow \mathit{Buffer} \setminus S$

  end for

- **until** $i = n_i$ or termination condition on model quality is met

Sample-Selection Methods

- Uninformed methods
  - Sequential – new samples are added in the order in which they have been stored to the buffer
  - Random – new samples are selected from the buffer randomly
- Informed methods
  - Maximum variance – a set of models is generated and the outputs of these models are calculated for all buffer samples; the sample with the highest variance in model output is added to the training set
  - Maximum output domain coverage – new samples are added from the buffer to cover the output domain as well as possible
  - Maximum prediction error (PERMIT) – a set of models is generated and the outputs of these models are calculated for all buffer samples; the sample with the highest average error is added to the training set

Experiments – Mobile Robot

- Continuous-time dynamics:

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

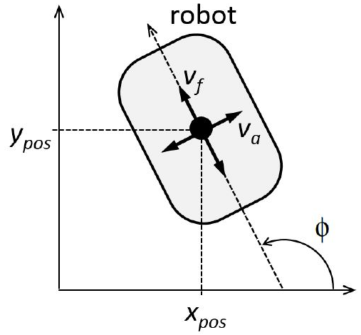
x_{pos} ... pose x-coordinate

y_{pos} ... pose y-coordinate

ϕ ... pose angle

v_f ... linear („forward") velocity

v_a ... angular velocity

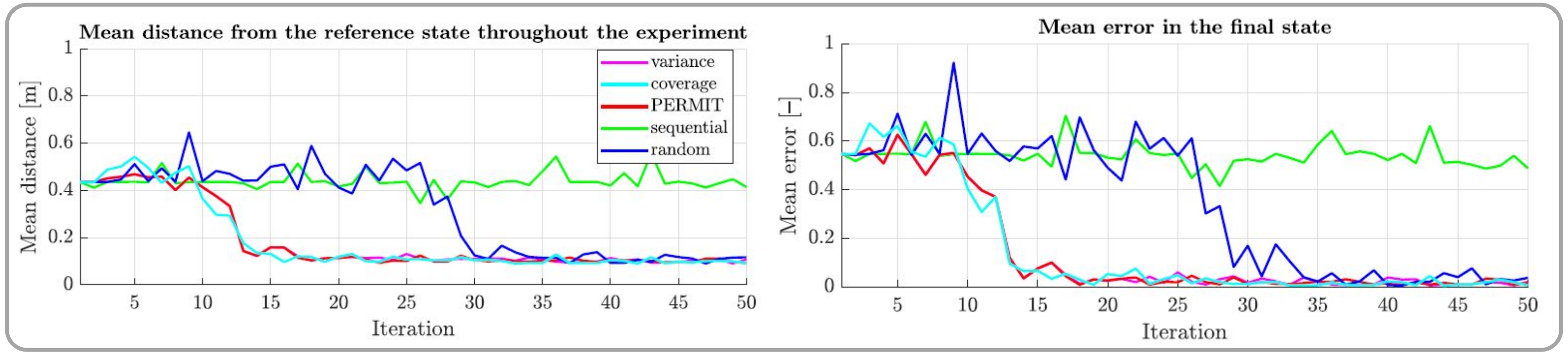


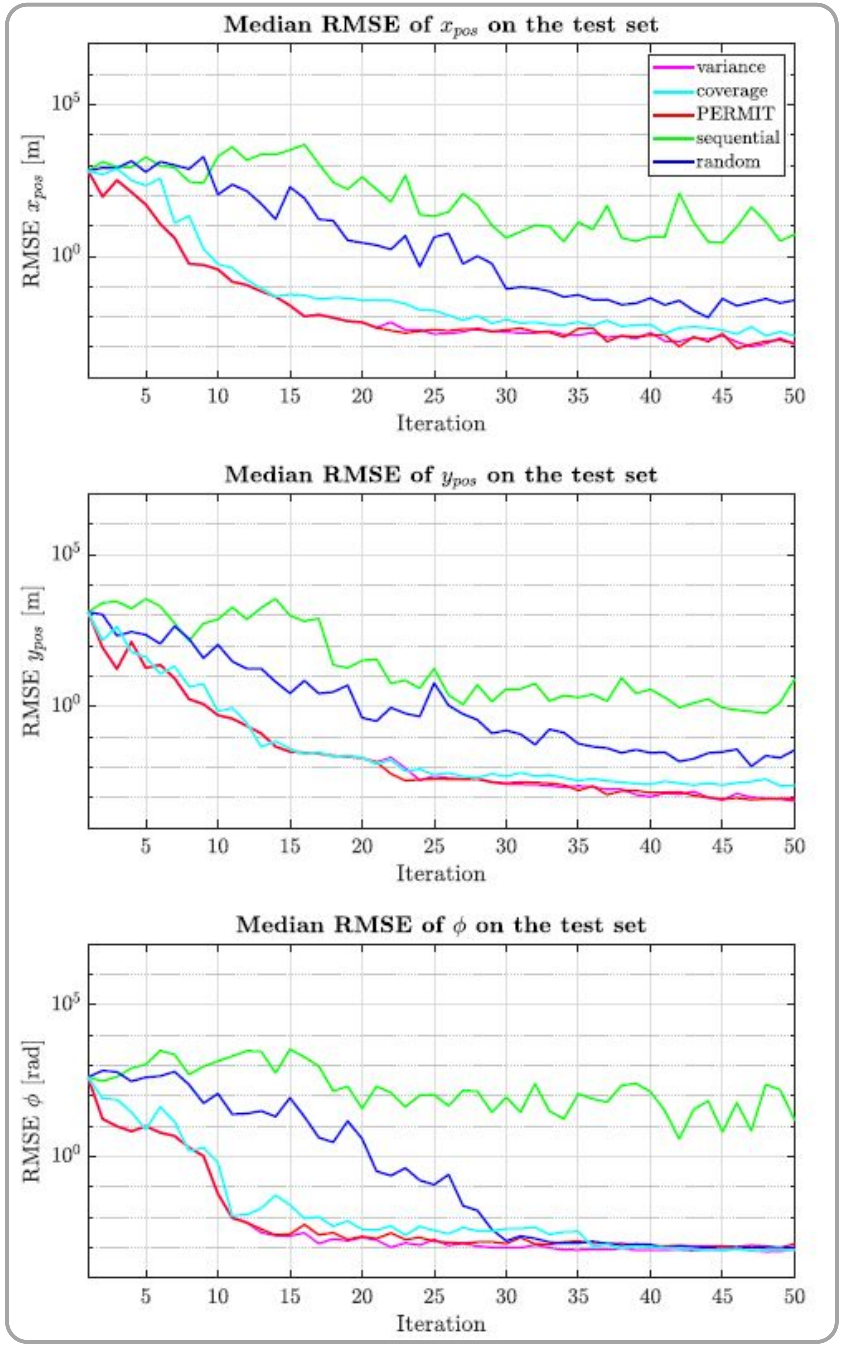
- Samples are collected in the following form:

$$\mathbf{s}_k = (x_{pos,k}, y_{pos,k}, \phi_k, v_{f,k}, v_{a,k}, x_{pos,k+1}, y_{pos,k+1}, \phi_{k+1})$$

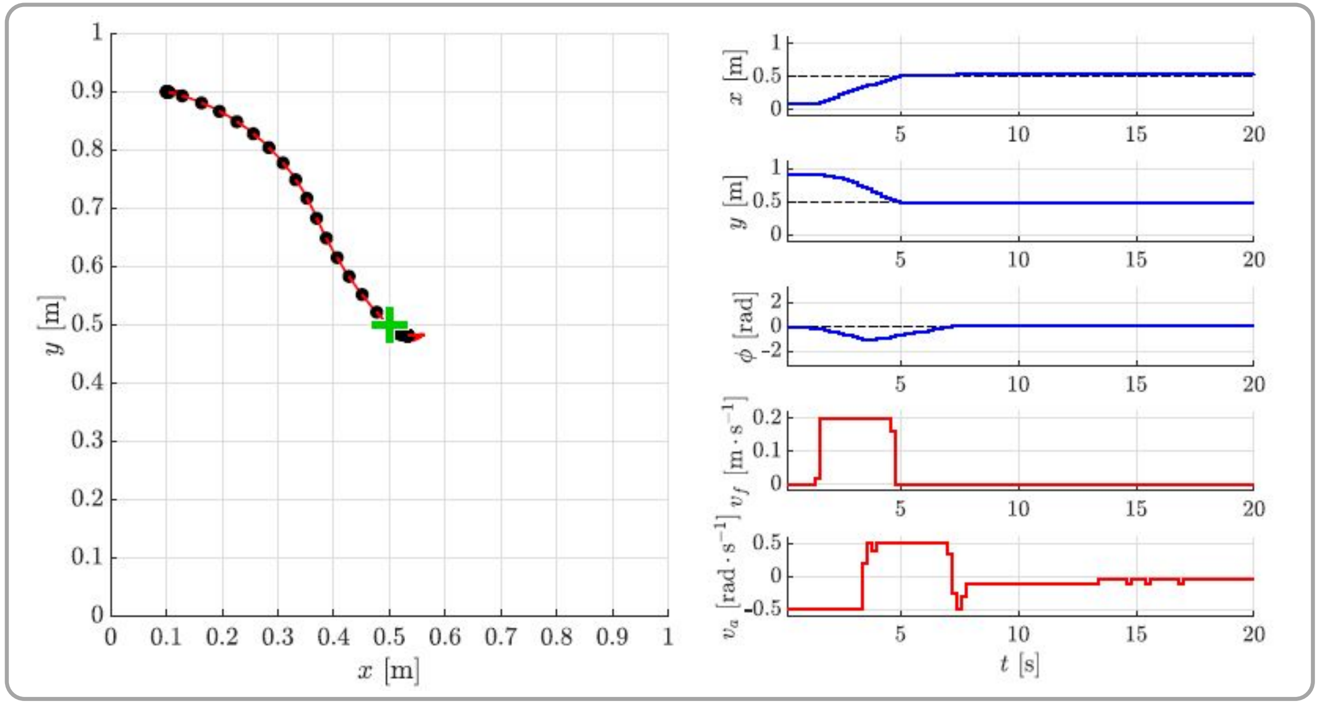
Results – Simulated Mobile Robot

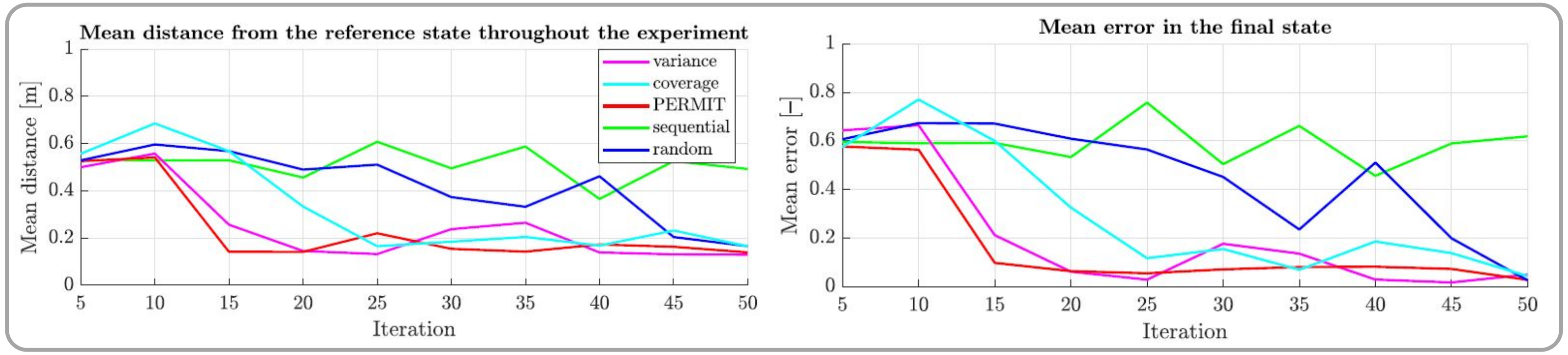
- 500 samples collected on a trajectory from a repetitive task with 20 % of random input
- Starting with only 5 training samples, adding 1 sample in each iteration
- 50 models generated in each iteration
- Comparison of all sample-selection methods
- Control task using reinforcement learning

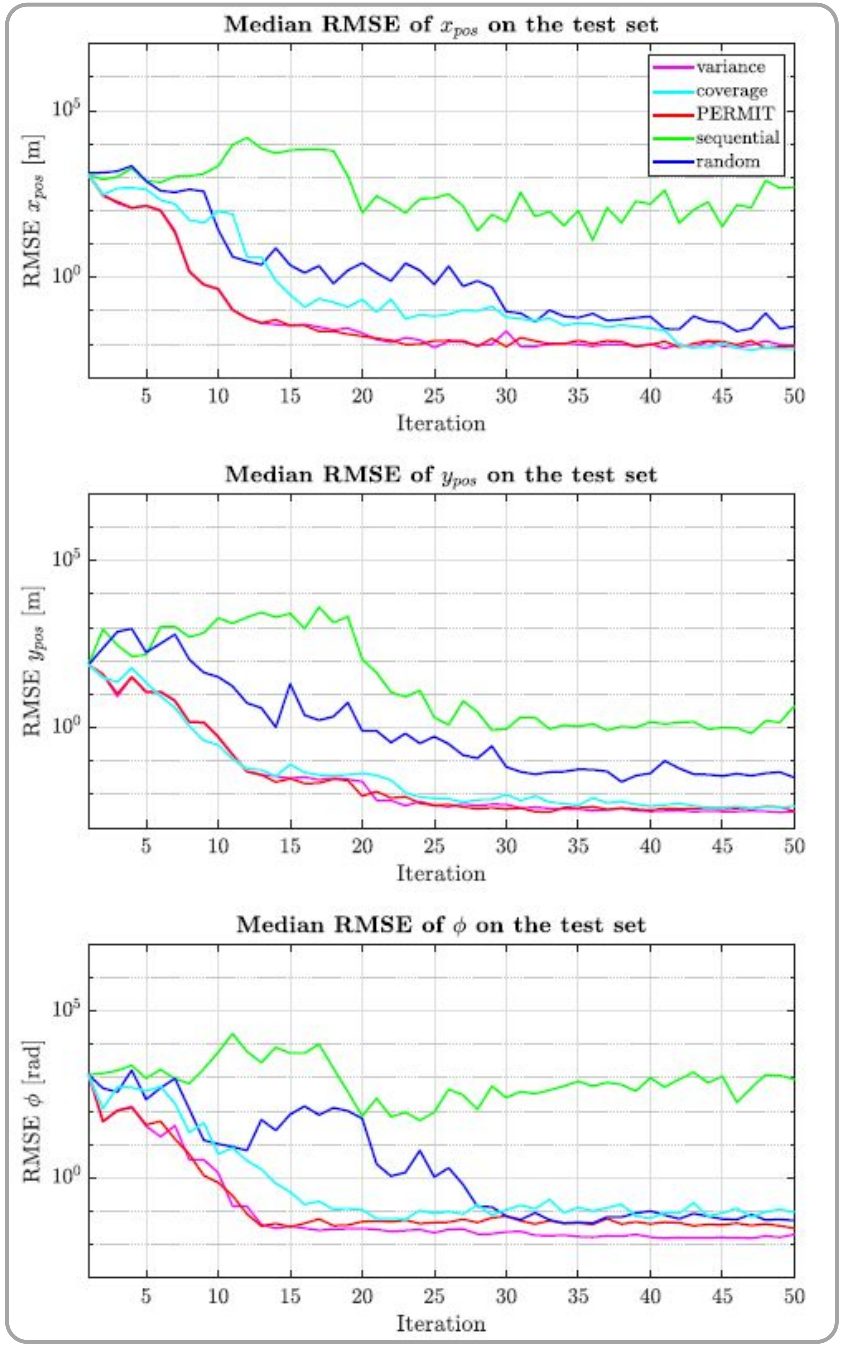




Results – Real Mobile Robot

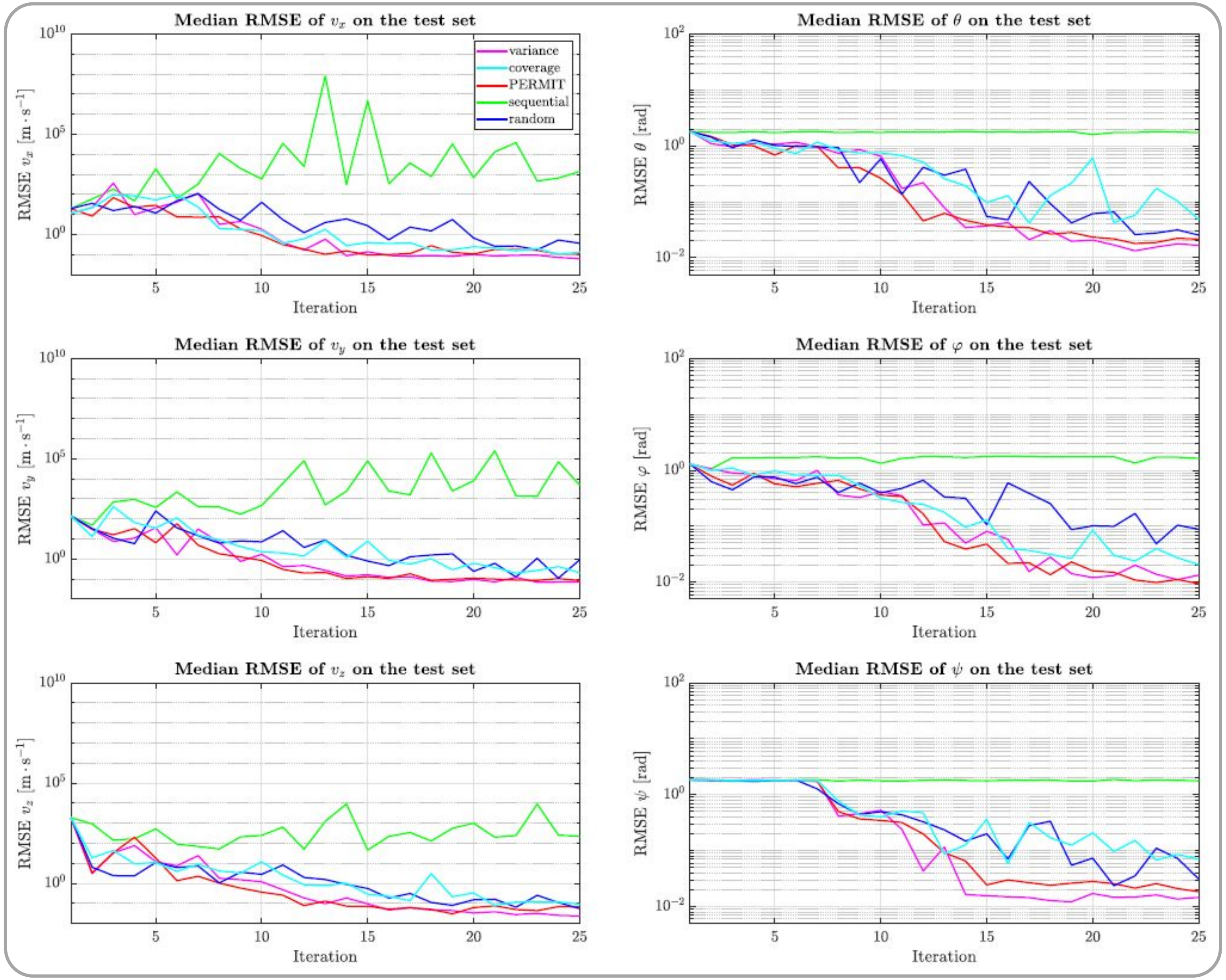






Results – Drone

- Modeling six state variables of the drone
- Buffer of ~1000 samples collected by teleoperating the real drone
- Starting with only 5 training samples, adding 1 sample in each iteration
- 10 models generated in each iteration



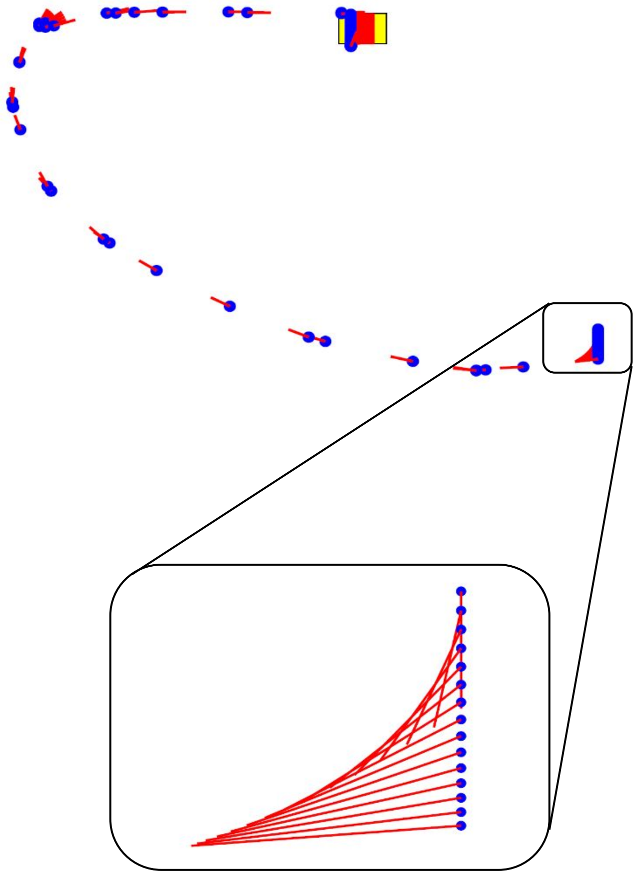
Model Learning with Sample Selection – Summary

- Selection of training samples is essential to efficiently construct accurate models from a large amount of data
- Informed methods clearly outperform the uninformed methods
- PERMIT and the variance method achieve the best performance
- Using the PERMIT method, an analytic model constructed by symbolic regression using 24 samples can be used to design a near-optimal RL controller for the real robot
- Future work
  - Training set maintenance, such as outlier detection and removal
  - Real-world long-term autonomy experiment to evaluate how the sample-selection method deals with unforeseen situations

Combining Data and Prior Knowledge – Motivation

- Models found using symbolic regression accurately fit the training data.

- Models may not comply with the physics of the robot (non-holonomic constraints, in this case).



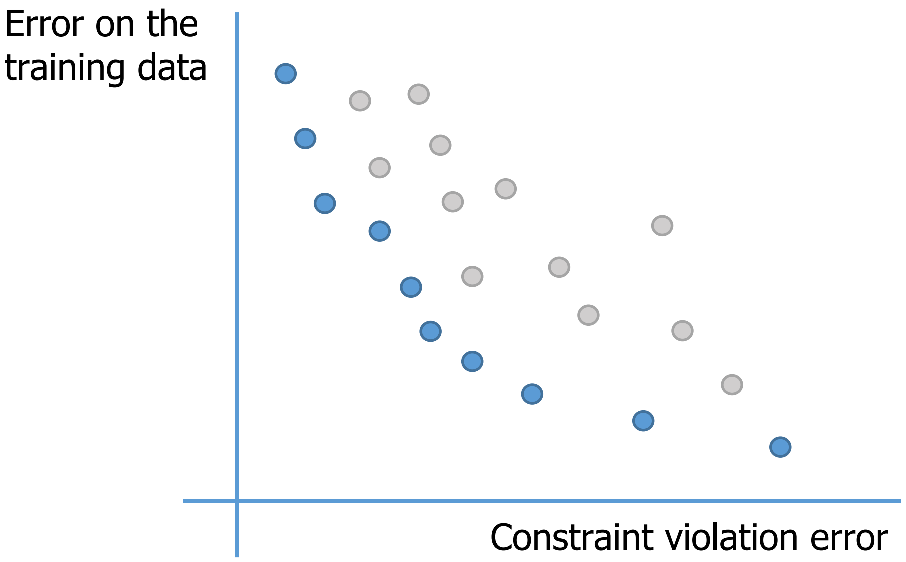
Robot model found using baseline SR

Prior Knowledge

- Prior knowledge captures important high-level characteristics of the system's physical laws without requiring in-depth knowledge of the physical model.
- It can be expressed as constraints on the model parameters or function values, data representing steady-state behavior of the system, velocity and acceleration trends under specific input, etc.
- Prior knowledge of the model's properties can be included in the model construction process as formal constraints or as a partial model.

Formal Constraints

- Desired model properties, such as monotonicity or symmetry, can be written as equality and inequality constraints.
- Extent to which the candidate models violate the constraints is calculated on synthetic, randomly sampled data.
- Multi-criteria optimization:
    - Error on the training data set
    - Constraint violation error



Prior Model

- Approximate or partial theoretical or empirical model of the robot is often known.
- This information can be included in the model structure as one or more prior features.
- Decomposing the prior model into several features allows to tune some of its inner parameters.
- Features evolved by genetic programming compensate for the prior features' deficiency.

$$f(\xi) =$$

Experiments

- Two robotic benchmarks
  - Mobile robot TurtleBot 2
  - Drone Parrot Bebop 2
- Two scenarios
  - Baseline SNGP
  - SNGP with formal constraints
- SR parameters
  - Elementary function set = $\{+,\ -,\ \times,\ \cdot^2,\ \cdot^3,\ \sin(\cdot),\ \cos(\cdot)\}$
  - Maximum number of features = 10
  - Maximum tree depth = 7

Mobile Robot

- Continuous-time dynamics:

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

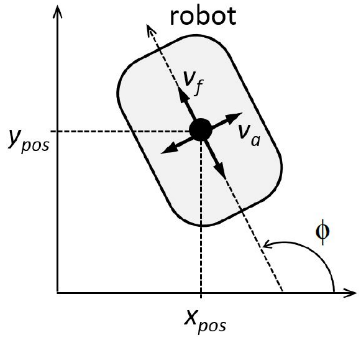
x_{pos} ... pose x-coordinate

y_{pos} ... pose y-coordinate

ϕ ... pose angle

v_f ... linear velocity

v_a ... angular velocity



- The goal is to find continuous-time model fitting the measured data:

$$\hat{\dot{x}}_{pos} = f_{\dot{x}_{pos}}(x_{pos}, y_{pos}, \phi, v_f, v_a)$$
$$\hat{\dot{y}}_{pos} = f_{\dot{y}_{pos}}(x_{pos}, y_{pos}, \phi, v_f, v_a)$$

- 87 discrete-time training samples (T_s = 0.2 s) of the form $[\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{x}_{k+1}]$
- State derivatives approximated using forward difference

Mobile Robot – Prior Knowledge

- Formal constraints for $\dot{x}_{pos}$:
  - Velocity along the x-axis is zero if the linear velocity v_f is zero.
  - Velocity along the x-axis is zero if the robot is moving in the positive or negative direction of the y-axis and it is not rotating at the same time.
- Similarly, we define constraints for $\dot{y}_{pos}$.
- Theoretical models are used as prior features:

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

- Bi-objective SNGP: Final model – lowest training RMSE among all models with the training constraint error less than 0.05

Mobile Robot – Example of Learned Model

Prior feature

$$\boxed{\dot{x}_{pos} = v_f \cos(\phi)}$$

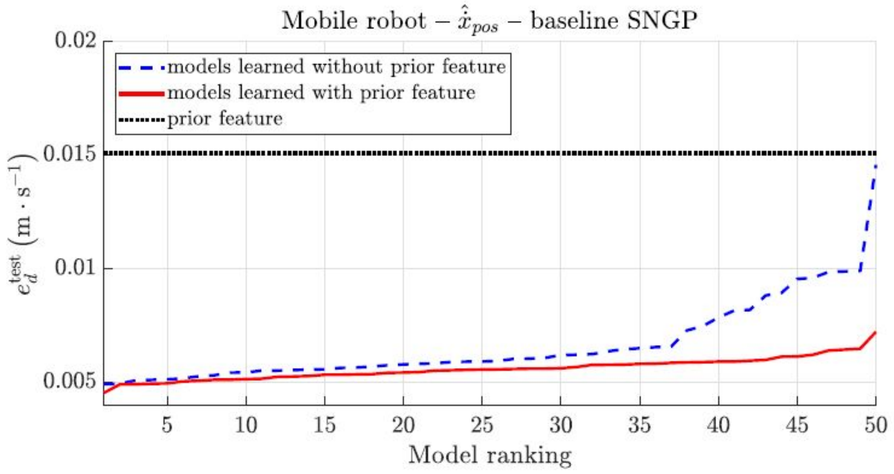
$$\boxed{\dot{y}_{pos} = v_f \sin(\phi)}$$

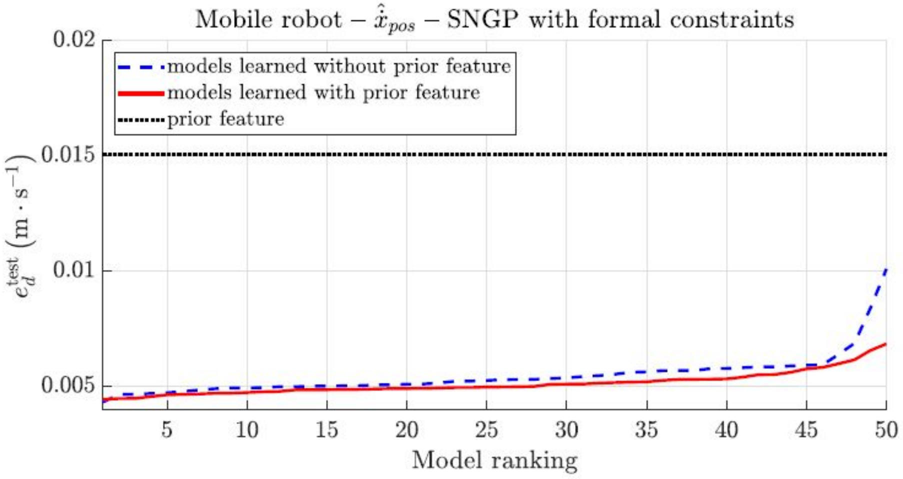
Final model

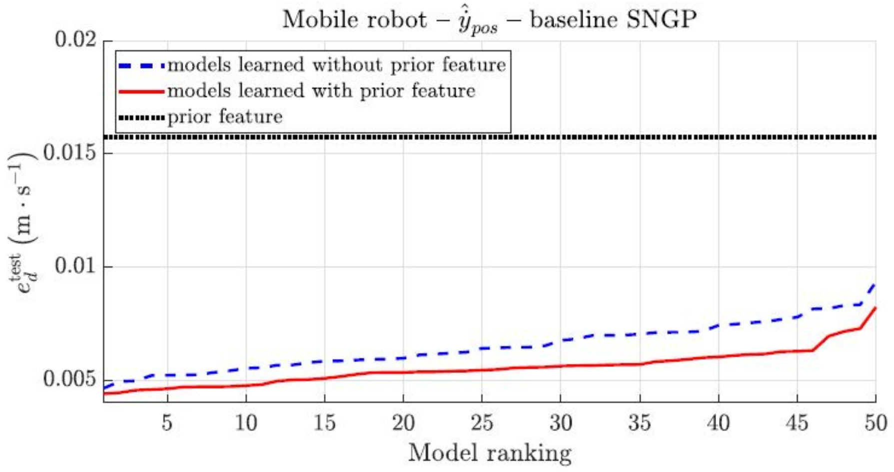
$$\begin{aligned}\hat{x}_{pos} = {} & \boxed{8.5 \times 10^{-1} v_f \cos(\phi)} - 1.2 \times 10^{-2} \sin(\sin(x_{pos})) \\ & + 1.3 \times 10^{-2} \sin(x_{pos})^2 - 7.7 \times 10^{-3} \cos(\phi)^3 \\ & + 3.7 \times 10^{-3} (\phi + v_a) + 2.8 \times 10^{-3} y_{pos} \cos(\phi + v_a) \\ & - 3.2 \times 10^{-3} (v_f + 2)(\sin(\sin(\phi)) - \cos(\phi)^3 \cos(x_{pos})) \\ & - 1.9 \times 10^{-3} (\phi + v_a)^2 + 1.8 \times 10^{-3} \cos(x_{pos})(\phi - 3.1) \\ & + 5.5 \times 10^{-4} y_{pos} - 4.8 \times 10^{-4} v_f - 3.1 \times 10^{-4}\end{aligned}$$

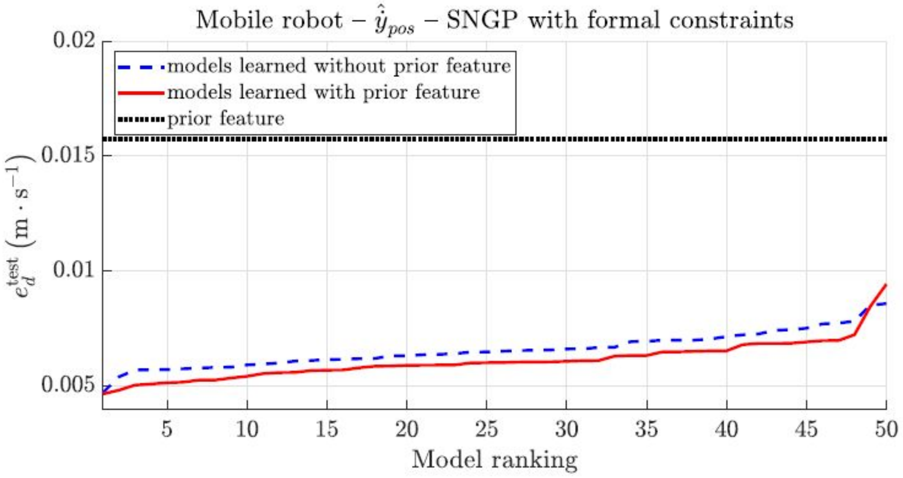
$$\begin{aligned}\hat{y}_{pos} = {} & \boxed{8.6 \times 10^{-1} v_f \sin(\phi)} - 2.3 \times 10^{-1} \cos(1.4 v_f) \\ & - 9.0 \times 10^{-2} v_f \sin(\cos(v_f^2)) - 8.3 \times 10^{-3} \cos(\phi - 2.8 v_f)^4 \\ & + 5.5 \times 10^{-3} (\phi + v_a) + 7.6 \times 10^{-4} x_{pos} - 1.6 \times 10^{-16} y_{pos}^{18} \\ & - 3.0 \times 10^{-3} \sin(x_{pos})^2 - 5.2 \times 10^{-3} \cos(v_a)^9 \\ & - 6.4 \times 10^{-4} (\phi - 2.8 v_f)^2 (y_{pos} - \sin(x_{pos})) \\ & + 6.1 \times 10^{-5} (y_{pos} - \sin(x_{pos}))^3 + 2.4 \times 10^{-1}\end{aligned}$$

Mobile Robot – Results

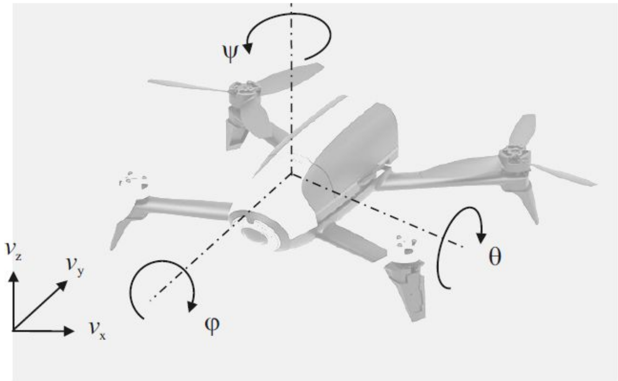








Drone



- Continuous-time dynamics:

$$\dot{v}_x = g\cos\psi\frac{\tan\theta}{\cos\varphi} + g\sin\psi\tan\varphi - k_D v_x$$

$$\dot{v}_y = g\sin\psi\frac{\tan\theta}{\cos\varphi} - g\cos\psi\tan\varphi - k_D v_y$$

$g = 9.81\ \mathrm{m} \cdot \mathrm{s}^{-1}$
$k_d = 0.28\ \mathrm{s}$

v_x, v_y, v_z ... translational velocities
θ, φ, ψ ... body angles (pitch, roll, yaw)

- The goal is to find continuous-time model fitting the measured data:

$$\hat{\dot{v}}_x = f_{\dot{v}_x}(v_x, \theta, \varphi, \psi)$$
$$\hat{\dot{v}}_y = f_{\dot{v}_y}(v_y, \theta, \varphi, \psi)$$

- 160 discrete-time training samples (T_s = 0.05 s) of the form $[\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{x}_{k+1}]$
- State derivatives approximated using forward difference

Drone – Prior Knowledge

- Formal constraints for $\dot{v}_x$ (similarly defined also for $\dot{v}_y$):
  - Given a zero velocity along the x-axis, zero pitch, yaw orienting the drone in the positive or negative direction of the x-axis, and a non-zero roll, the acceleration in the direction of the x-axis has to be zero.
  - Analogously for zero roll and yaw orienting the drone along the y-axis
- Empirical models are used as prior features in two variants:

1 feature

$$\dot{v}_x = g\cos\psi\frac{\tan\theta}{\cos\varphi} + g\sin\psi\tan\varphi - k_D v_x \qquad \dot{v}_y = g\sin\psi\frac{\tan\theta}{\cos\varphi} - g\cos\psi\tan\varphi - k_D v_y$$

3 features

$$\begin{aligned} \bar{f}_1 &= g\cos\psi\tan\theta/\cos\varphi & \qquad \bar{g}_1 &= g\sin\psi\tan\theta/\cos\varphi \\ \bar{f}_2 &= g\sin\psi\tan\varphi & \bar{g}_2 &= -g\cos\psi\tan\varphi \\ \bar{f}_3 &= -k_D v_x & \bar{g}_3 &= -k_D v_y \end{aligned}$$

Results Summary

Mobile robot

	Scenario	Prior feature	Median e_d^{test} $(\mathrm{m} \cdot \mathrm{s}^{-1})$
$\hat{x}_{pos}$	Baseline	Not included	5.920×10^{-3}
		Included	5.562×10^{-3}
	Constrained	Not included	5.273×10^{-3}
		Included	4.973×10^{-3}
$\hat{y}_{pos}$	Baseline	Not included	6.414×10^{-3}
		Included	5.455×10^{-3}
	Constrained	Not included	6.492×10^{-3}
		Included	6.010×10^{-3}

- In 88 % prior model improves accuracy
- Statistically significant improvement ($p \ll 0.01$)
- In 3/4 cases 3 prior features are better than 1

Drone

	Scenario	Empirical model	Median e_d^{test} $(\mathrm{m} \cdot \mathrm{s}^{-2})$
$\hat{v}_x$	Baseline	Not included	7.508×10^{-1}
		1 prior feature	6.877×10^{-1}
		3 prior features	7.237×10^{-1}
	Constrained	Not included	1.153×10^{0}
		1 prior feature	2.245×10^{-1}
		3 prior features	1.980×10^{-1}
$\hat{v}_y$	Baseline	Not included	6.803×10^{-1}
		1 prior feature	8.536×10^{-1}
		3 prior features	8.260×10^{-1}
	Constrained	Not included	1.987×10^{-1}
		1 prior feature	1.727×10^{-1}
		3 prior features	1.639×10^{-1}

Symbolic Regression for Model Learning – Conclusions

- Symbolic regression allows to automatically construct analytic models of dynamic systems
- Such models can be easily plugged into other algorithms and facilitate further analysis
- If the data from a long-term continuous data stream are selected in an informed way, only a few samples are necessary to train a precise model of the robot's dynamics
- Model learning through symbolic regression is extended by including a prior (theoretical, empirical) model
- Including prior information to the model construction process yields accurate and physically plausible models that compensate for data deficiencies
- 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

- Symbolic regression methods in robotics
  - Direct tuning of the model accuracy-complexity tradeoff (progressive model construction and reduction)
  - Modeling value functions (V-functions) in RL using the proposed SR extensions (sample selection, prior knowledge)
- 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