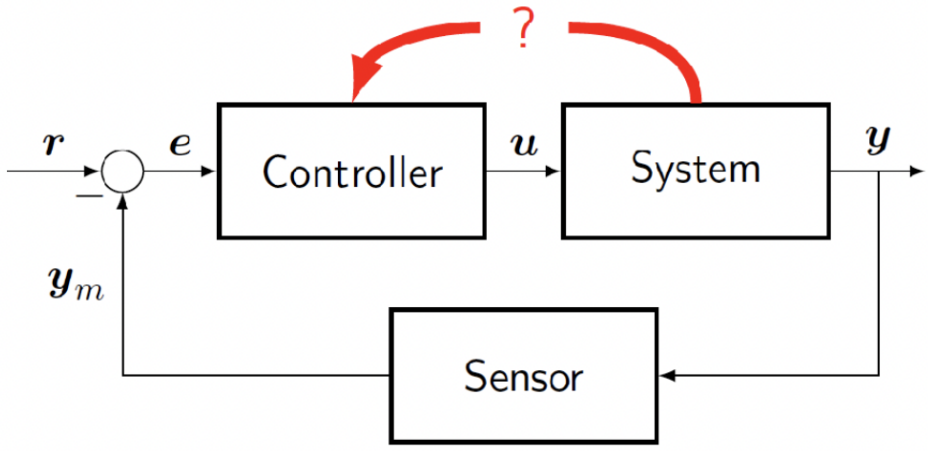
An Introduction to data-driven control with Gaussian process regression

Leonardo J. Colombo

Centre for Automation and Robotics
Spanish National Research Council

leonardo.colombo@csic.es

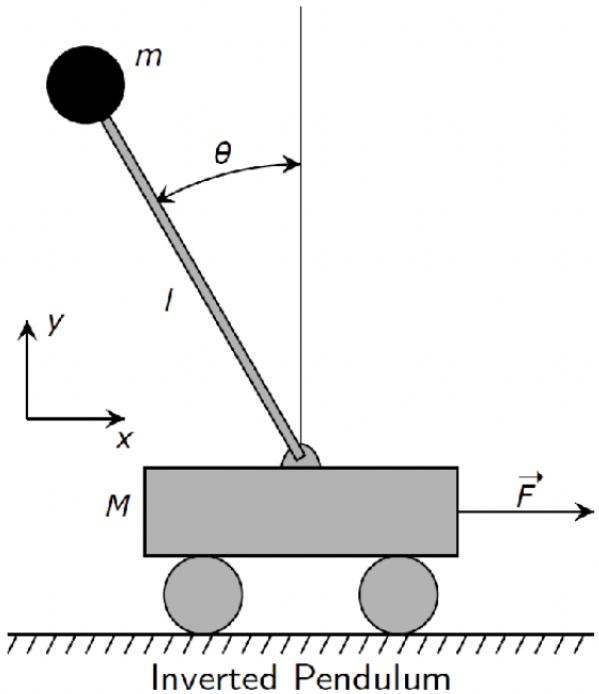
Motivation



Design of controller typically requires a precise model of the system

Motivation

How to model a system?



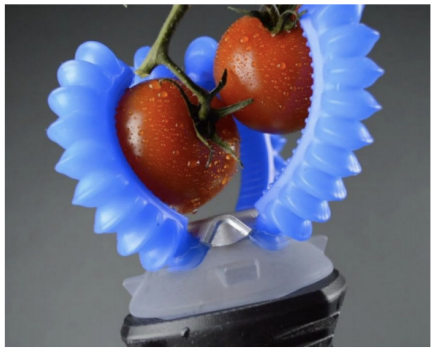
e.g. using Lagrange-equation:

$$(M+m)\ddot{x} - ml\ddot{\theta}\cos\theta + ml\dot{\theta}\sin\theta = F$$

$$l\ddot{\theta} - g\sin\theta = \ddot{x}\cos\theta$$

Motivation

Modeling control systems based on data



[Soft robotics]

[Kuka]

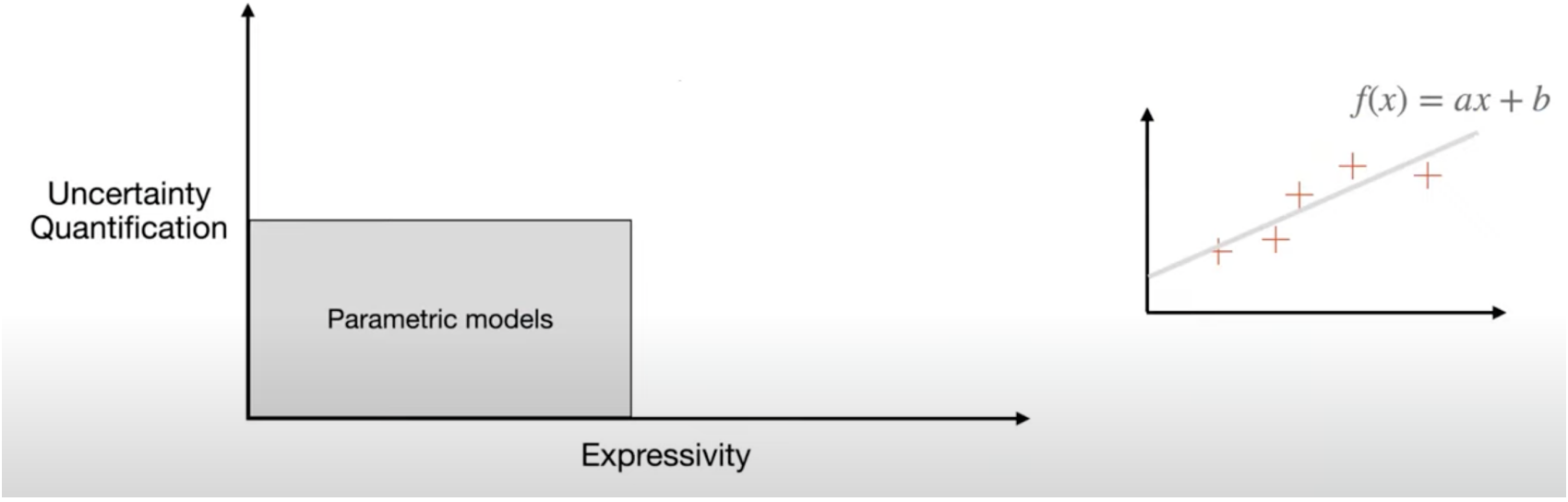
[Bitcraze]

- The parametric modeling is very time-consuming or even unfeasible.
- Limited to the number of parameters to estimate.
- The modern control methods based on data-driven approaches as neural networks, in general, does not present safety guarantees.

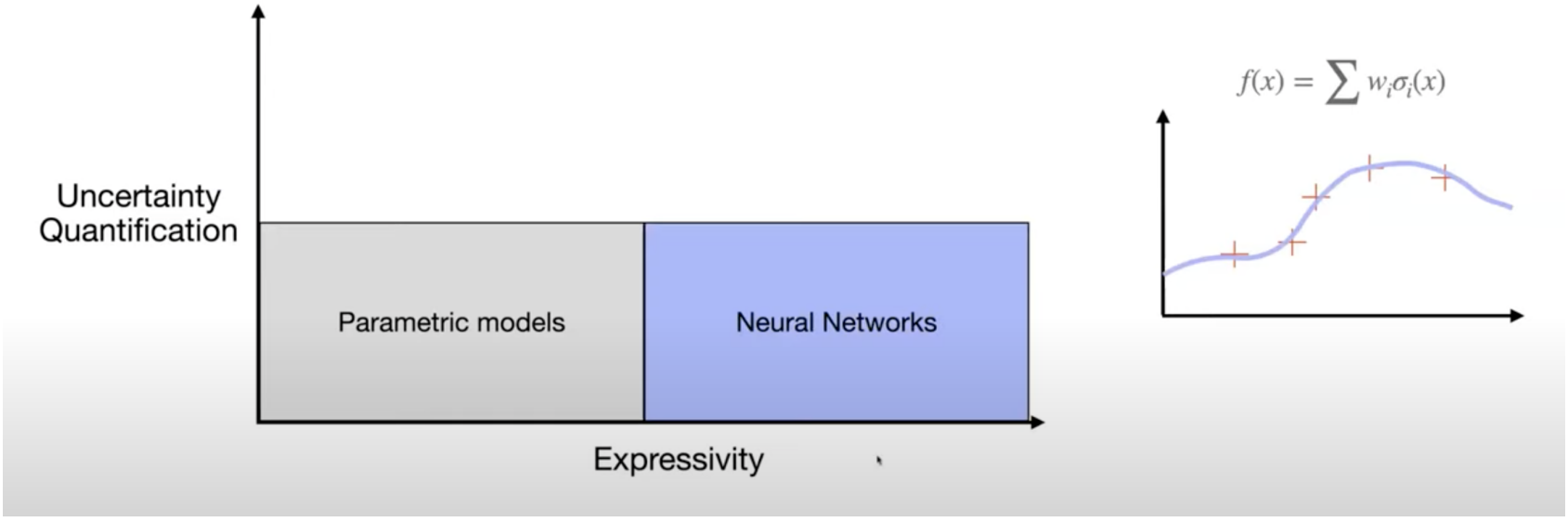
Lack of safety guarantees

- There is a need to construct algorithms based on data guaranteeing the security of the system.

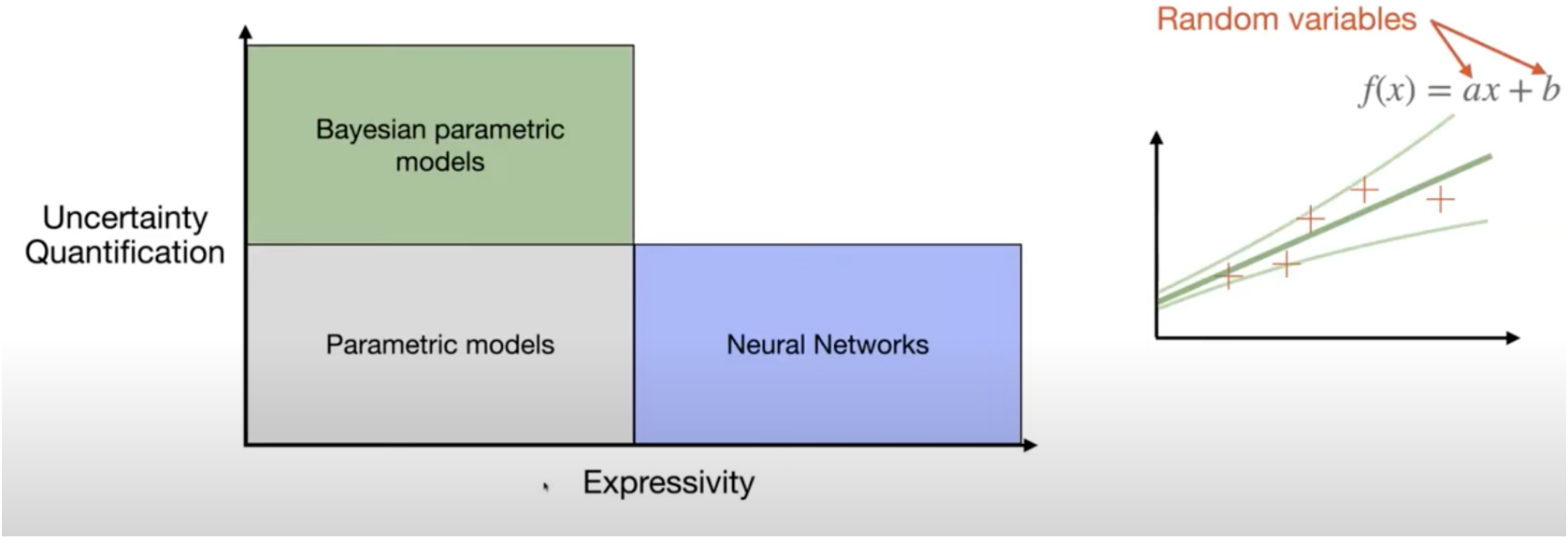
Learning methods



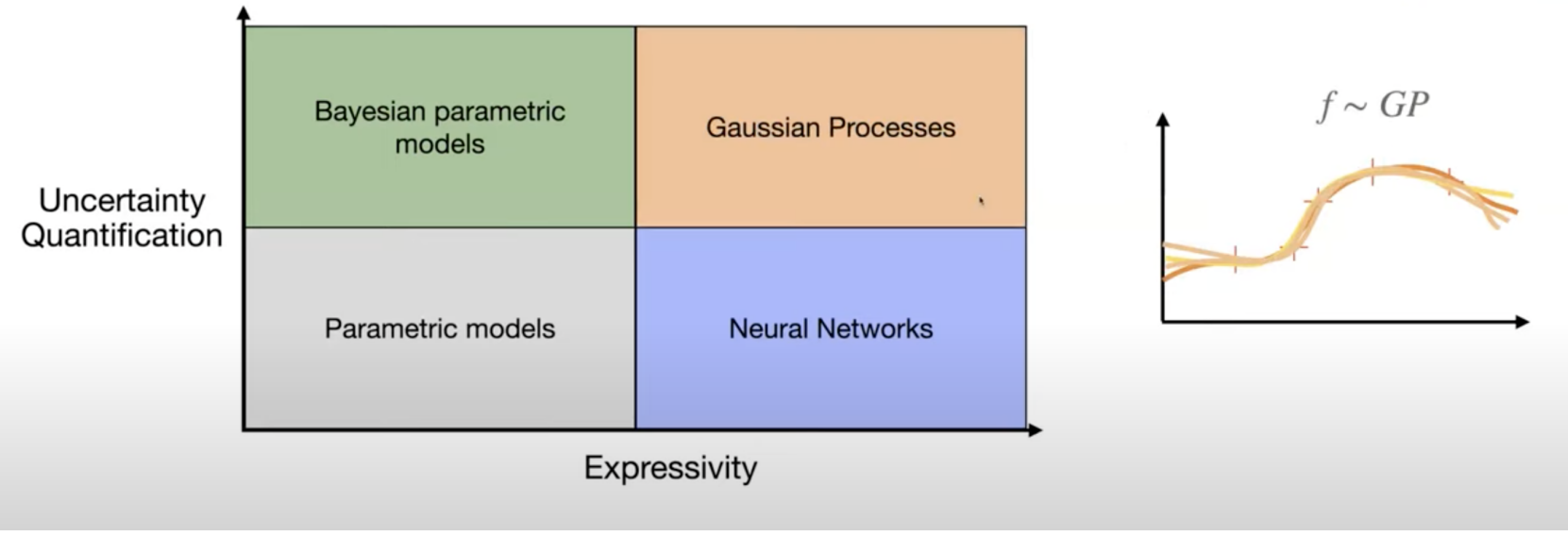
Learning methods



Learning methods

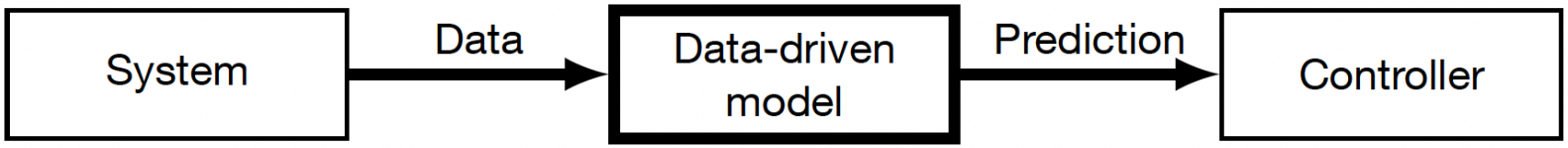


Learning methods



Gaussian Processes: Prior knowledge + nonparametric + uncertainty quantification

Data-driven control design

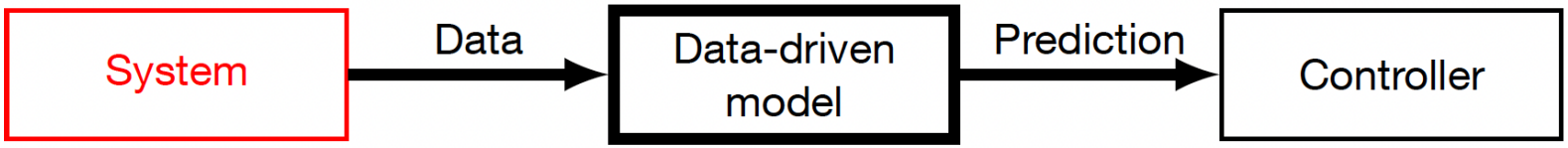


Data-driven control design

Outlook:

- Benefits of data driven models

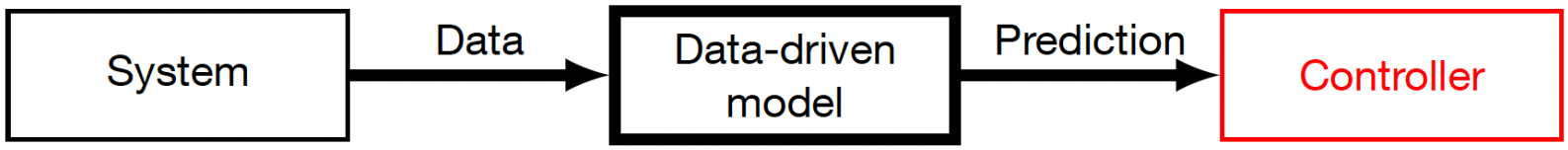
Data-driven control design



Outlook:

- Benefits of data driven models
- Training and prior knowledge

Data-driven control design

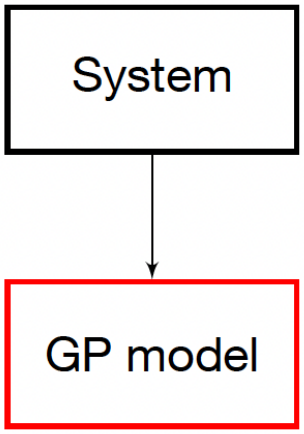


Outlook:

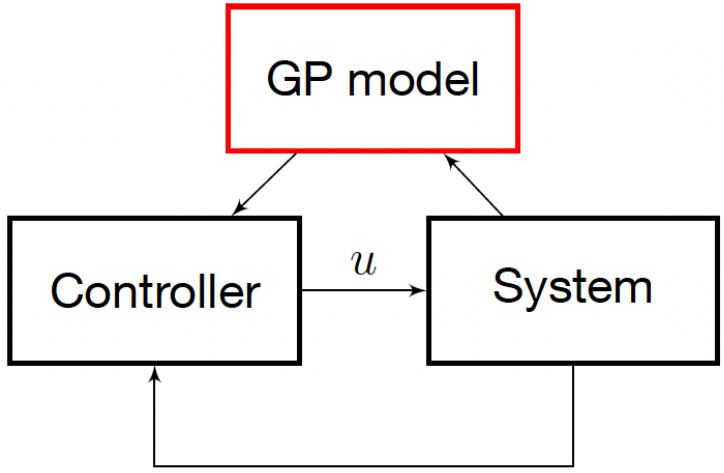
- Benefits of data driven models
- Training and prior knowledge
- Data driven based control

Data-driven control design

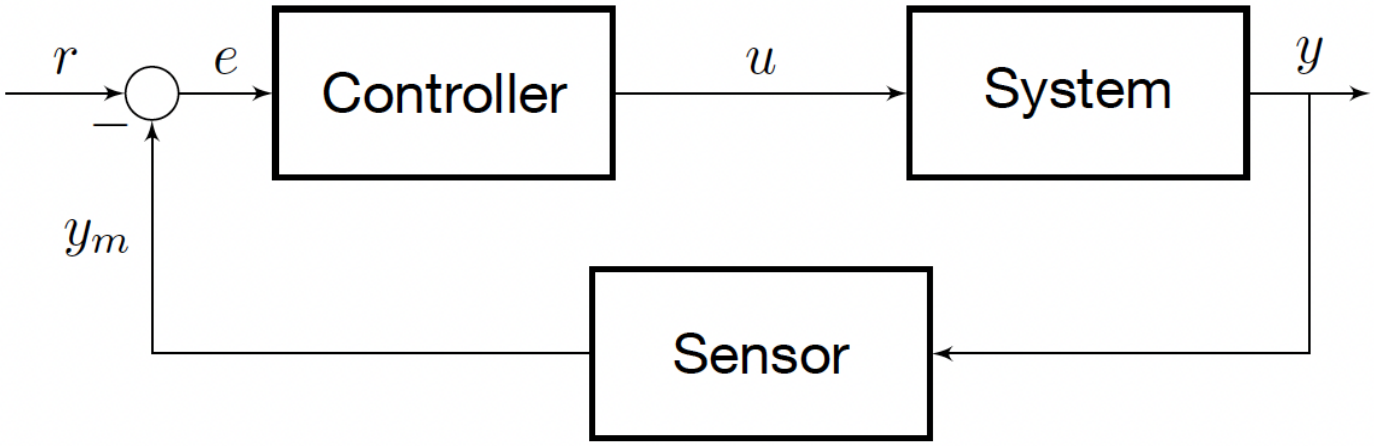
1. Step:
Learning the system dynamics



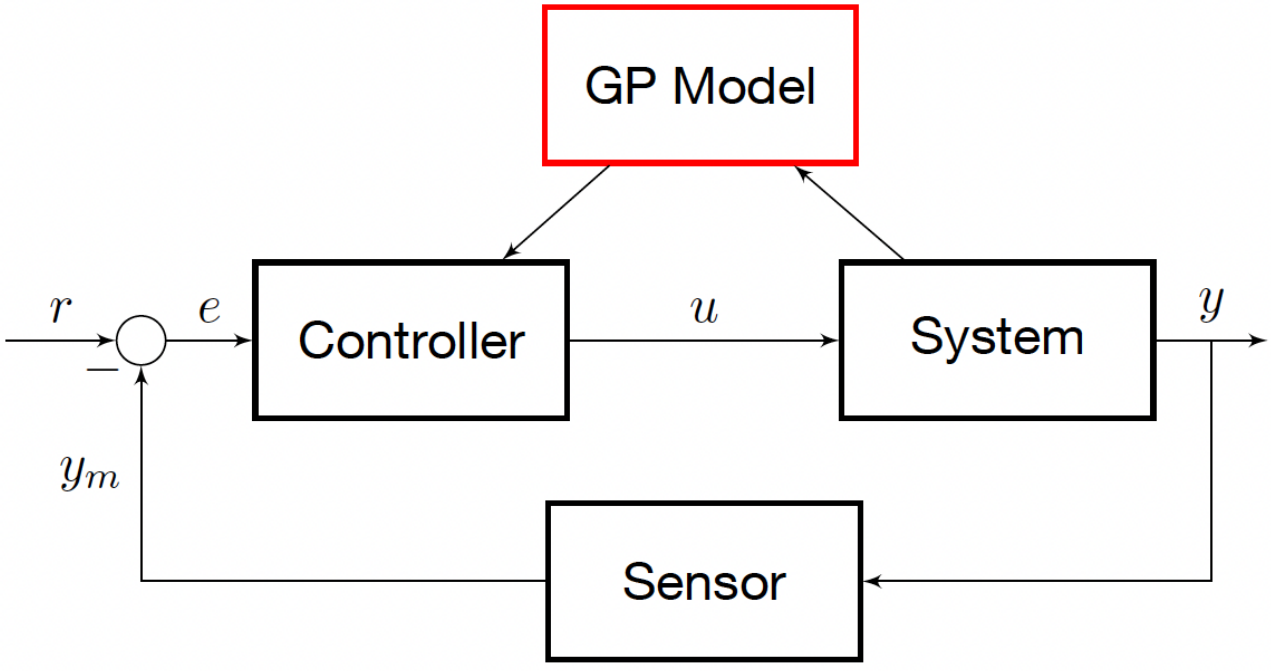
2. Step:
Using the GP model in the controller



Data-driven control design



Data-driven control design



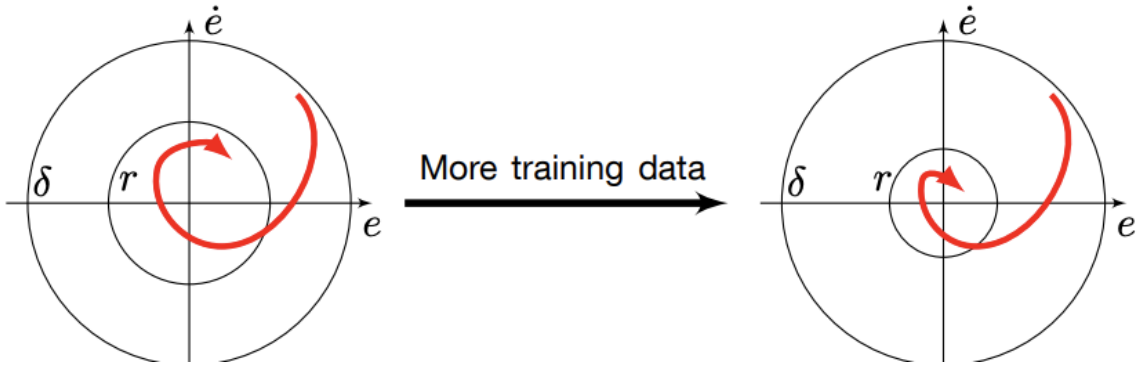
Goal: Improved performance with stability guarantees

Safety guarantees

Boundedness of the tracking error: The data-driven controller guarantees that, for any desired maximum tracking error $\varepsilon > 0$, the control law guarantees that the tracking error is bounded in probability and exponentially converges such that

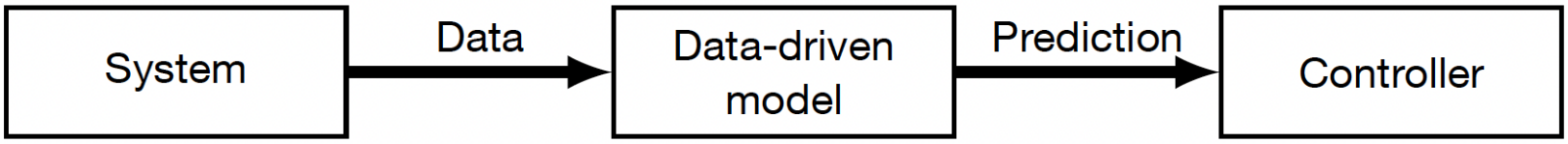
$$\mathbb{P}\{\|\boldsymbol{e}(t)\| \leq \underbrace{\gamma_1 \exp^{-\gamma_2 (t-t_0)} \|\boldsymbol{e}(0)\| + \varepsilon}_{:=r}\} \geq \delta$$

with $r = r(||u||_{\mathcal{H}}, \mathcal{D})$.

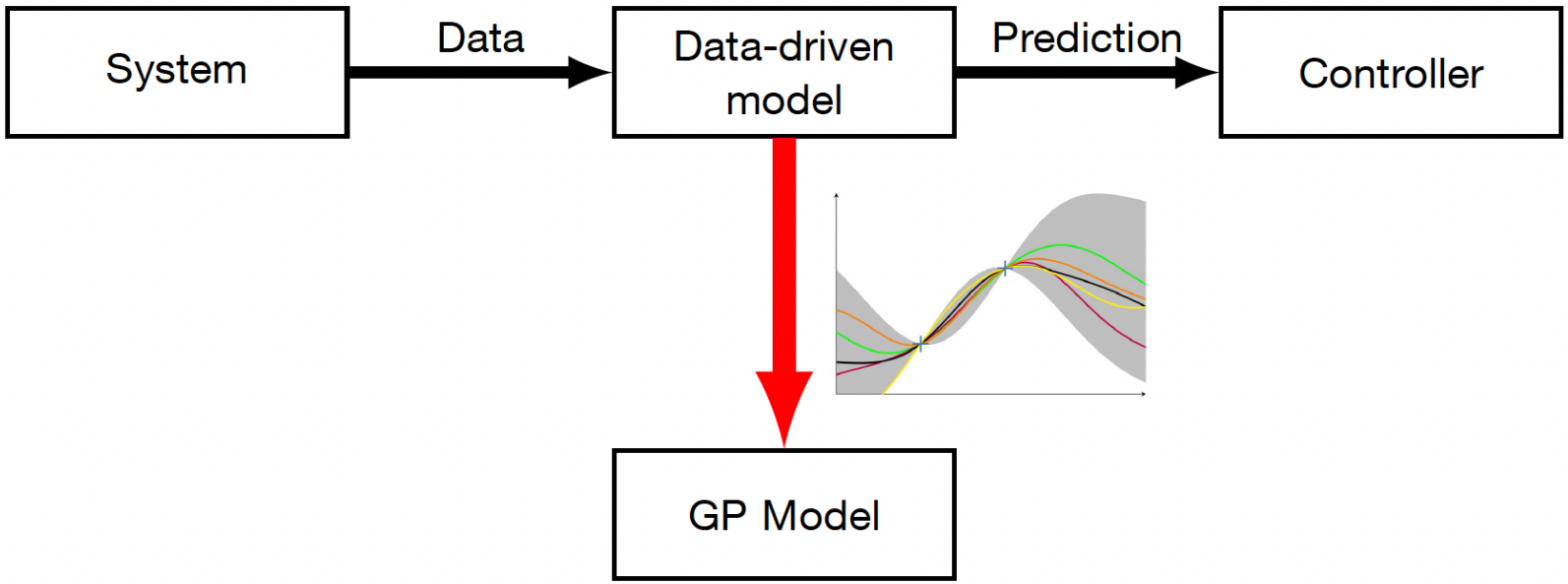


T. Beckers, L. Colombo, M. Morari, G. Pappas. *Learning-based Balancing of Model-based and Feedback Control for Second-order Mechanical Systems. 2022 IEEE 61th Annual Conference on Decision and Control (CDC).*

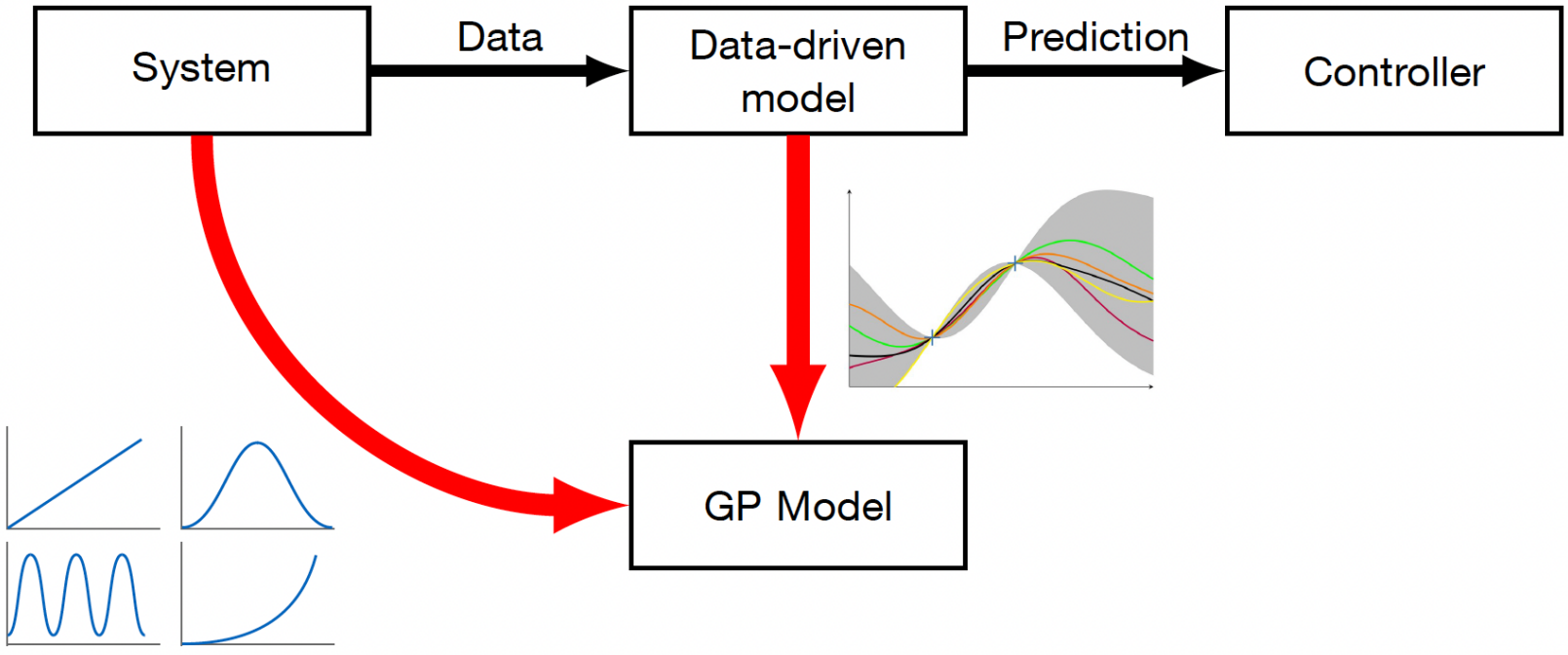
Data-driven tracking control: Summary



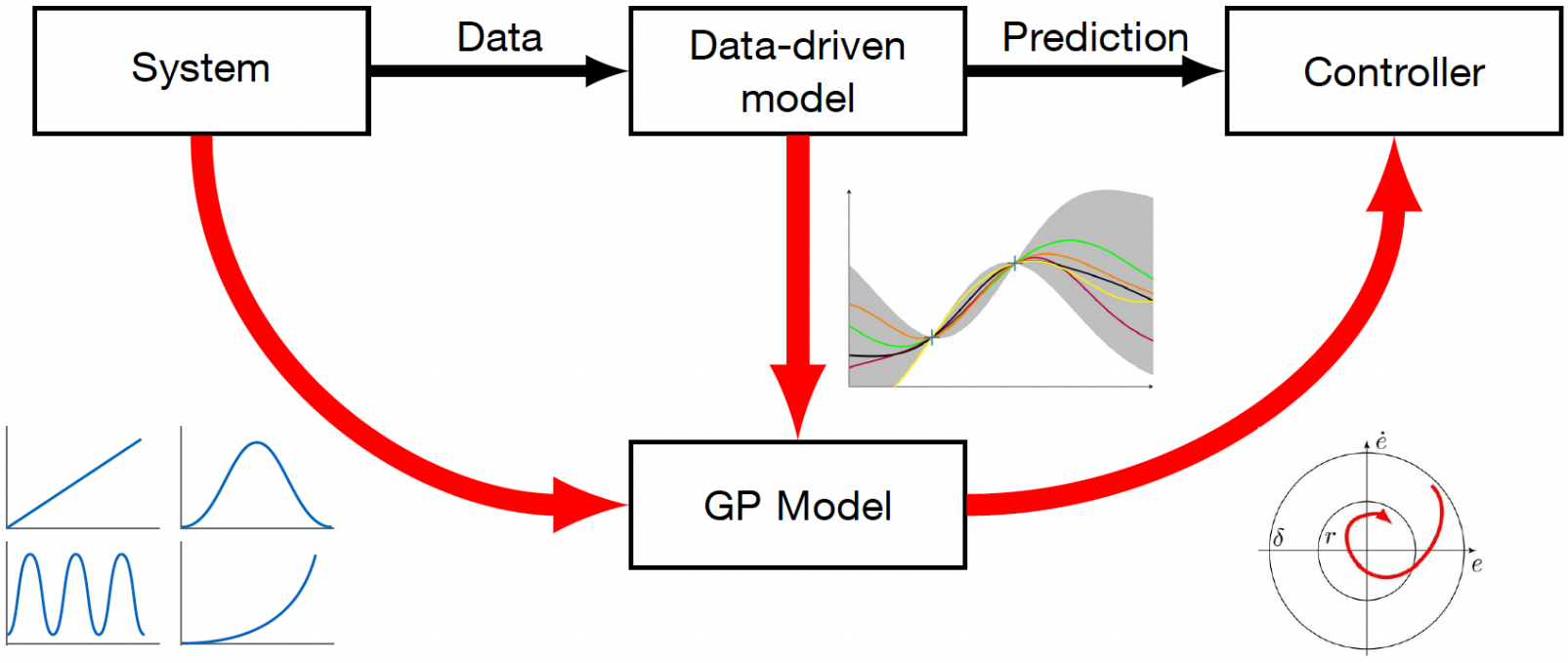
Data-driven tracking control: Summary



Data-driven tracking control: Summary



Data-driven tracking control: Summary

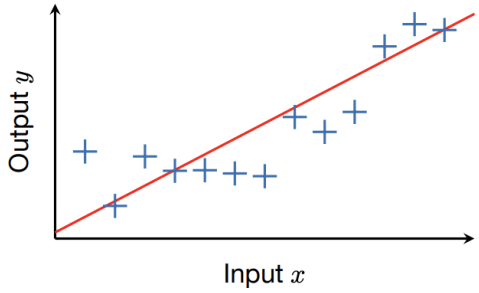


Data-driven control design

Outlook:

- Benefits of data driven models: Gaussian Process Regression

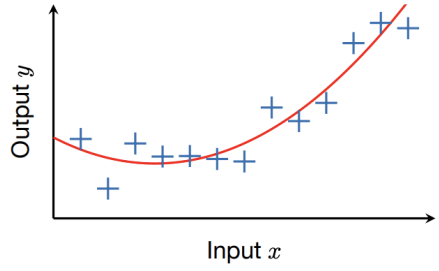
Parametric modeling



$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta})$

Input x and Output y
Model $\boldsymbol{f}$
Parameter $\boldsymbol{\theta}$

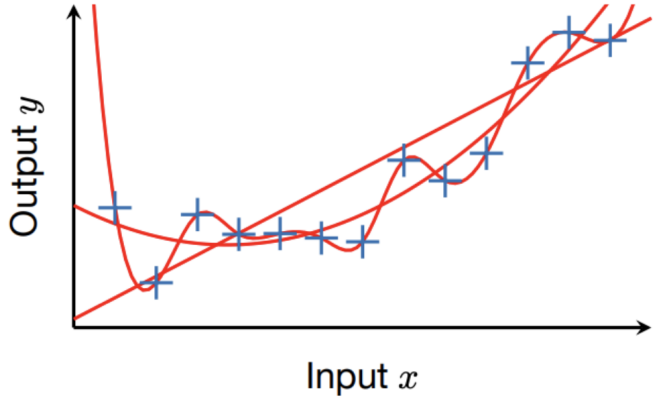
Linear: $y = ax + b$



$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta})$

Input x and Output y
Model $\boldsymbol{f}$
Parameter $\boldsymbol{\theta}$

Quadratic: $y = ax^2 + bx + c$

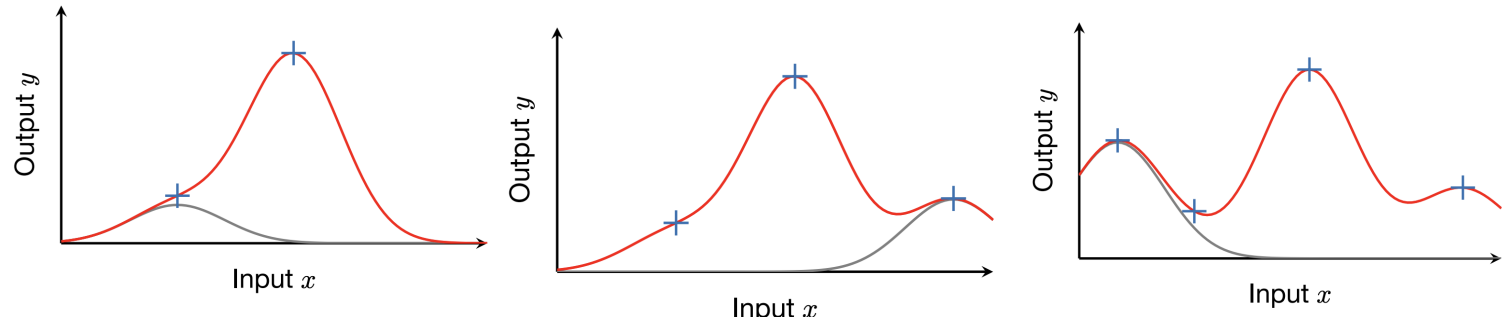


¿Which model is correct?

Number of fixed parameters to estimate $\boldsymbol{\theta}$, with prediction based on the parametric model estimated.

The complexity of the model is limited due to fixed number of parameters

Non-parametric modeling - Data driven modeling



Properties:

- The model scales/adapts with the number of data N.
- Depends on the set of data $\mathcal{D} = \{X, Y\}$,
- The complexity of the model is not limited, the model learns from the data
- Very flexible but often black-box behavior (lost of interpretability).
- Probabilistic interpretation with Bayesian statistics. In particular, Gaussian regression models

Data-driven models

Outlook:

Gaussian processes extends the concept of Gaussian distribution to an infinity collection of variables.

- This extension permits to think a Gaussian process as a distribution over functions and not only over vectors of random variables.
- In general, Gaussian processes are defined as a distribution over probability functions.

Gaussian processes

What is a Gaussian Process? It is a Gaussian distribution over function space.

Gaussian Processes are defined by the mean and covariance functions:

$$\boxed{\varphi_{GP}(\xi) \sim \mathcal{GP}\left(m(\xi), k\left(\xi, \xi'\right)\right)}$$

- ξ values in the domain
- (ξ, ξ'),all possible pairs in the domain,
- $m(\xi)$ is the mean function
- $k\left(\xi, \xi'\right)$ is the covariance function.

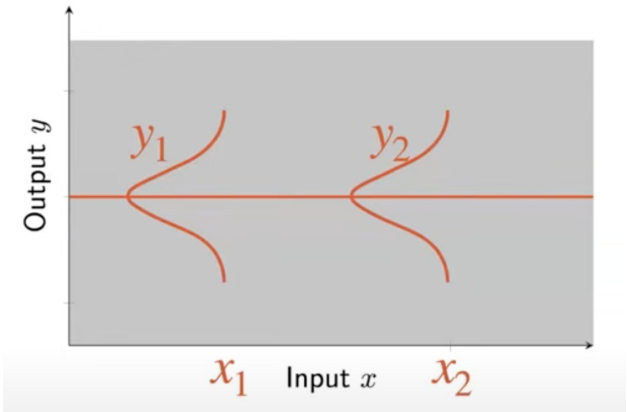
C.E. Rasmussen. *Gaussian processes for machine learning. MIT Press.* 2006.

C. Bishop. Pattern Recognition and Machine Learning. *Springer*, 2006.

Gaussian processes

In other words, a Gaussian process is a collection of random variables, of which any finite number has a joint Gaussian distribution.

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(x_1) \\ m(x_2) \end{bmatrix} \begin{bmatrix} k(x_1, x_1) & k(x_1, x_2) \\ k(x_2, x_1) & k(x_2, x_2) \end{bmatrix} \right)$$



Gaussian processes

• To understand the basic functioning of the GP, we have to move our point of view from function to feature space.

• The so called kernel trick allows for computationally efficient implicit calculations in the feature space.

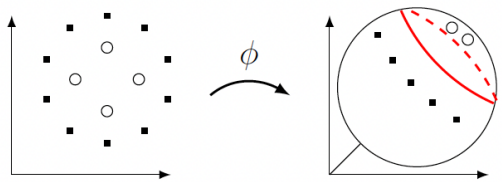


Figure: The mapping ϕ transforms the data points into a feature space where linear regressors can be applied to predict the output.

• The high-dimensional space is found by a feature map ϕ from the low dimensional space to the high-dimensional space. Instead of using the input x we use the feature map for that input $\phi(x)$.

The nice thing is that the feature map ϕ is not needed as it appears as $\langle \phi(x), \phi(x') \rangle =: k(x, x')$

• Therefore, the essential part in GP modeling concerns the kernel $k(\cdot, \cdot)$.

Different kernels for the covariance function

Let the covariance matrix be denoted by $K_{i,j} = k(x_i, x_j; \theta_K)$, where k refers to the function kernel or covariance function, which establish the correlations among different points of the process.

One of the most commonly used kernels in control theory is the square exponential (SE) kernel with the form

$$k_{\mathrm{SE}}\left(x_i, x_j; A, L\right) := A^2 \exp\left\{-\frac{\left(x_i - x_j\right)^2}{2L^2}\right\} + \sigma_n^2 \delta_{ij},$$

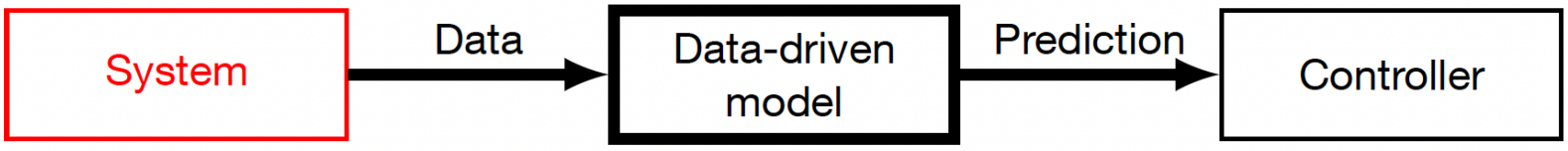
where $\theta_K = A, L, \sigma_n \in \mathbb{R}$ to be determined are called hyperparameters.

The hyperparameter A describes the signal variance which determines the average distance of the data-generating function from its mean.

The length-scale L defines how far it is needed to move along a particular axis in input space for the function values to become uncorrelated.

σ_n is a signal noise. The squared exponential kernel is infinitely differentiable, which means that the GPR exhibits a smooth behavior.

Data-driven control design



Outlook:

- Benefits of data driven models: Gaussian Process Regression
- Training and prior knowledge

Bayesian approach for model training

The Bayesian approach to learning models differs from other methods in two fundamental ways:

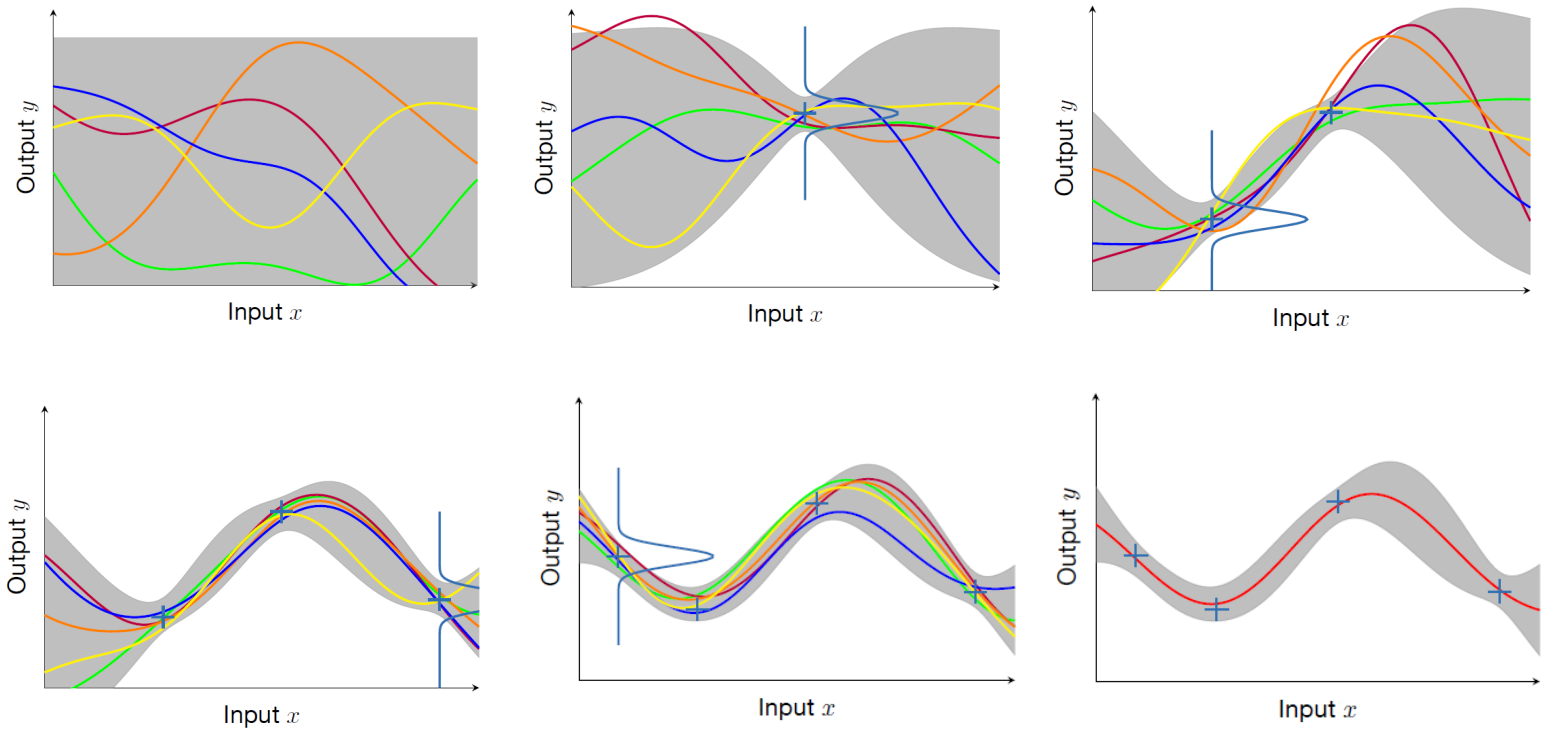
- an *a prior* distribution is selected based on prior knowledge,
- predictions about future observations are made by integrating the model's predictions regarding the *a posterior* distribution of the parameters, which is obtained by actualizing the *a prior* distribution with data.
- Combines previous knowledge with data to obtain an improved model through Bayes theorem
- Model $\mathcal{M}$, data $\mathcal{D}$, previous information of the model $\boldsymbol{\theta}$.

$$\underbrace{\Pr(\mathcal{M}|\mathcal{D},\boldsymbol{\theta})}_{\text{Posterior}} \propto \underbrace{\Pr(\mathcal{D}|\mathcal{M},\boldsymbol{\theta})}_{\text{Likelihood}} \underbrace{\Pr(\mathcal{M}|\boldsymbol{\theta})}_{\text{Prior}}$$

- Prior: Previous knowledge without data.
- Posterior: New knowledge with data.
- Likelihood: Likelihood of the data $\mathcal{D}$ with model $\mathcal{M}$ and parameters $\boldsymbol{\theta}$.

Training a model by using GPR

Bayesian approximation: Update of the prior distribution with new data.



(a) Space of possible (parametric) a priori models. (b) Given a piece of data, the process learns from the data. Note that we not only have a point estimate but also a probabilistic distribution on the estimated data. (f) Posteriori distribution.

The posteriori distribution is again a Gaussian process

Specific kernels generates bounded trajectories (confidence of the model)

Training a model by using GPR

Bayesian approach: Update of the prior distribution with new data

- $\mathcal{D} = \{X, Y\}$, $X = \left[\boldsymbol{x}^{\{1\}}, \boldsymbol{x}^{\{2\}}, \ldots, \boldsymbol{x}^{\{m\}}\right] \in \mathbb{R}^{p \times m}$ e $Y = \left[\boldsymbol{y}^{\{1\}}, \boldsymbol{y}^{\{2\}}, \ldots, \boldsymbol{y}^{\{m\}}\right] \in \mathbb{R}^{p \times m}$ are the training data.

- For the test input $\boldsymbol{x}^* \in \mathbb{R}^p$ the prediction for $\boldsymbol{f}(\boldsymbol{x}^*)$ are obtaining by conditioning over the data which gives rise to the posteriori distribution.

$$\mu\left(f_i \,|\, \boldsymbol{x}^*, \mathcal{D}\right) = m(\boldsymbol{x}^*) + \boldsymbol{k}\left(\boldsymbol{x}^*, X\right)^\top \left(K + I\sigma^2\right)^{-1} Y_{:,i},$$

$$\text{var}\left(f_i \mid \boldsymbol{x}^*, \mathcal{D}\right) = k\left(\boldsymbol{x}^*, \boldsymbol{x}^*\right) - \boldsymbol{k}\left(\boldsymbol{x}^*, X\right)^\top \quad \left(K + I\sigma^2\right)^{-1} \boldsymbol{k}\left(\boldsymbol{x}^*, X\right)$$

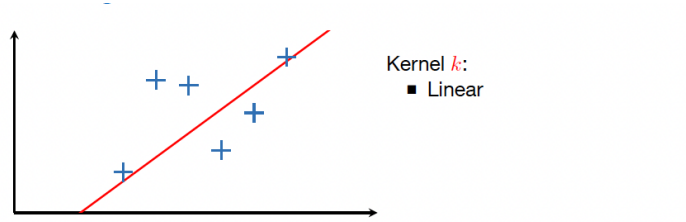
for all $i \in \{1, \ldots, p\}$, where $Y_{:,i}$ denotes the i-th column of Y.

- The kernel k measures the correlation between two entries $(\boldsymbol{x}, \boldsymbol{x}')$. The function $K \colon \mathbb{R}^{p \times m} \times \mathbb{R}^{p \times m} \to \mathbb{R}^{m \times m}$ is the Gram matrix with elements given by $K_{j',j} = k(X_{:,j'}, X_{:,j}) + \delta(j, j')\sigma^2$ for all $j', j \in \{1, \ldots, m\}$ with the delta function $\delta(j, j') = 1$ for all $j = j'$ and zero otherwise.

- The vector valued function $\boldsymbol{k} \colon \mathbb{R}^p \times \mathbb{R}^{p \times m} \to \mathbb{R}^m$, with elements $k_j = \boldsymbol{k}(\boldsymbol{x}^*, X_{:,j})$ for all $j \in \{1, \ldots, m\}$, express the covariance between $\boldsymbol{x}^*$ the training data with entry X.

Training a model by using GPRs: An overview

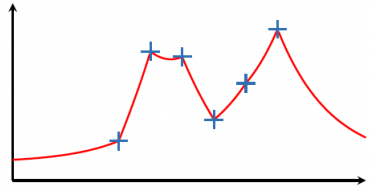
- The choice of the kernel and the determination of the corresponding hyperparameters can be seen as degrees of freedom of the regression procedure.



Kernel k:
- Linear

$$k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^\top \boldsymbol{x}'$$

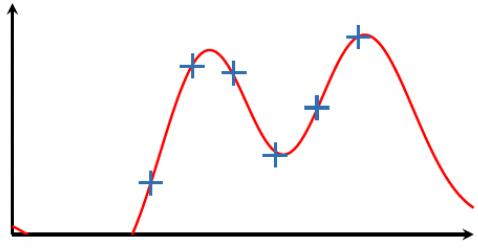
$$\mu(\boldsymbol{x}|\mathcal{D}) = \sum_{j=0}^{N} w_j k(\boldsymbol{x}, X_j)$$



Kernel k:
- Linear
- Matern, continuous

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|}{2l}\right)$$

$$\mu(\boldsymbol{x}|\mathcal{D}) = \sum_{j=0}^{N} w_j k(\boldsymbol{x}, X_j)$$



Kernel k:
- Linear
- Matern, continuous
- Squared Exponential

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2l^2}\right)$$

$$\mu(\boldsymbol{x}|\mathcal{D}) = \sum_{j=0}^{N} w_j k(\boldsymbol{x}, X_j)$$

Kernel defines the properties of the model

Model training: optimization of hyperparameters

• Model training refers to optimizing the parameters of the mean and the kernel function. We consider a smooth but unknown mapping f with $f(x) = y$.

• Let $\mathcal{D}_\nu = \{x_i, y_i\}_{i=1}^{\nu}$ be the training data set consisting of n-dimensional input-output pairs, where the output measurements posible are noisy.

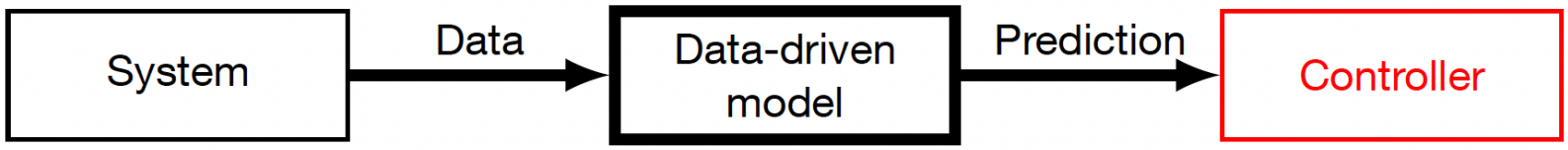
• Usually, the hyperparameter optimization is performed using standars gradiendt decent methods to minimize the negative marginal log likelihood, which can be calculated analytically,

$$h_l^* = \text{arg min}_{h_l} - \log p(\{(y_i)_{i=1}^{\nu}\}|\{x_i\}_{i=1}^{\nu}, h_l).$$

• Since an analytic solution of the derivation of $\log p(\{(y_i)_{i=1}^{\nu}\}|\{x_i\}_{i=1}^{\nu}, h_l)$ is impossible, a gradient based optimization algorithm is typically used to minimize the function.

• However, the negative log likelihood is non-convex in general such that there is no guarantee to find the optimum h_l^*. In fact, every local minimum corresponds to a particular interpretation of the data.

Data-driven control design



Outlook:

- Benefits of data driven models
- Training and prior knowledge
- Data driven based control

Multirotor UAVs

Multirotors UAVs have become very popular in recent years, thanks to the number of applications in which they have proven to be useful. From the transportation of medical supplies to remote places, the inspection of civil works or power lines, assistance to firefighters, lifeguards, agricultural applications, among others.

Underactuated vehicles

We assume a single underactuated rigid body with position $\boldsymbol{p} \in \mathbb{R}^3$ and orientation matrix $R \in SO(3)$. The body-fixed angular velocity is denoted by $\boldsymbol{\omega} \in \mathbb{R}^3$. The vehicle has mass $m \in \mathbb{R}_{>0}$ and rotational inertia tensor $J \in \mathbb{R}^{3\times3}$. The state space of the vehicle is $S = SE(3) \times \mathbb{R}^6$ with $\boldsymbol{s} = ((R, \boldsymbol{p}), (\boldsymbol{\omega}, \dot{\boldsymbol{p}})) \in S$ denoting the state of the system.

Underactuated vehicles

The vehicle is actuated with torques $\boldsymbol{\tau} \in \mathbb{R}^3$ and a force $u \in \mathbb{R}$, which is applied in a body-fixed direction defined by a unit vector $\boldsymbol{e} \in \mathbb{R}^3$. We can model the system as

$$\begin{aligned} m\ddot{\boldsymbol{p}} &= R\boldsymbol{e}u + \boldsymbol{f}(\boldsymbol{p}, \dot{\boldsymbol{p}}), \quad \dot{R} = R\check{\boldsymbol{\omega}} \\ \dot{\boldsymbol{\omega}} &= J^{-1}\big(J\boldsymbol{\omega} \times \boldsymbol{\omega} + \boldsymbol{\tau} + \boldsymbol{f}_\omega(\boldsymbol{s})\big), \end{aligned} \tag{1}$$

where the map $\check{(\cdot)} \colon \mathbb{R}^3 \to \mathfrak{so}(3)$ is given by

$$\check{\boldsymbol{\omega}} = \begin{bmatrix} 0 & -\omega_3 & -\omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}, \tag{2}$$

with the components of the angular velocity $\boldsymbol{\omega} = [\omega_1, \omega_2, \omega_3]^\top$. The functions $\boldsymbol{f} \colon \mathbb{R}^6 \to \mathbb{R}^3$ and $\boldsymbol{f}_\omega \colon S \to \mathbb{R}^3$ are state-depended disturbances and/or unmodeled dynamics. It is assumed that the full state $\boldsymbol{s}$ can be measured.

Underactuated vehicles

The general objective is to track a trajectory specified by the functions $(R_d, \boldsymbol{p}_d) \colon [0, T] \to SE(3) \simeq SO(3) \times \mathbb{R}^3$.

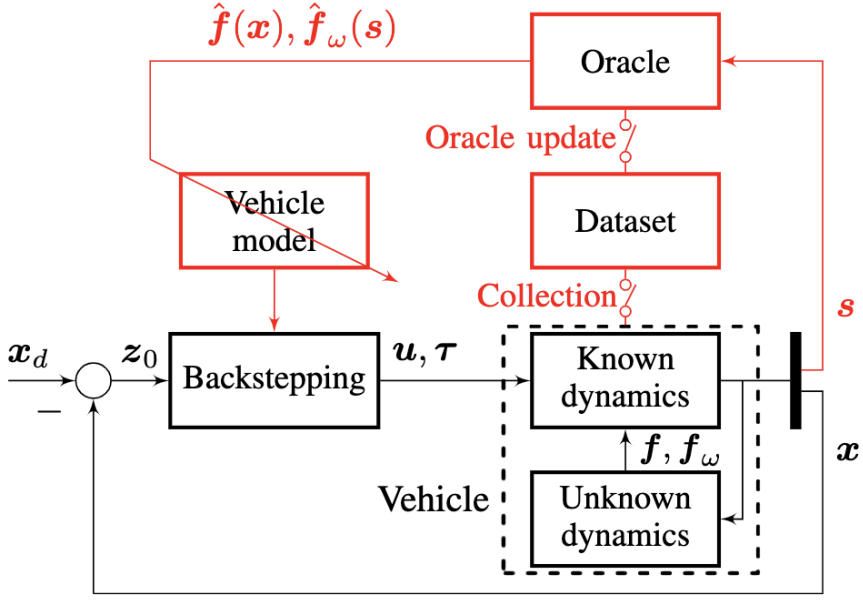


Figure: Control archichecture.

T. Beckers, L. Colombo, G. Pappas, S. Hirche. Online learning-based trajectory tracking for underactuated vehicles with uncertain dynamics. IEEE Control System Letters, 2021.

Learning

For the learning of the unknown dynamics of (7), we consider an oracle which predicts the values of $\boldsymbol{f}, \boldsymbol{f}_\omega$ for a given state $\boldsymbol{s}$. For this purpose, the oracle collects $N(n)\colon \mathbb{N} \to \mathbb{N}$ training points of the system (7) to create a data set

$$\mathcal{D}_{n(t)} = \{\boldsymbol{s}^{\{i\}}, \boldsymbol{y}^{\{i\}}\}_{i=1}^{N(n)}. \tag{3}$$

The output data $\boldsymbol{y} \in \mathbb{R}^6$ are given by $\boldsymbol{y} = [(m\ddot{\boldsymbol{p}} - R\boldsymbol{e}u)^\top, (J(\dot{\boldsymbol{\omega}} - \boldsymbol{\omega} \times \boldsymbol{\omega}) - \boldsymbol{\tau})^\top]^\top$ such that the first three components of $\boldsymbol{y}$ correspond to $\boldsymbol{f}$ and the remaining to $\boldsymbol{f}_\omega$. The data set $\mathcal{D}_{n(t)}$ with $n\colon \mathbb{R}_{\geq 0} \to \mathbb{N}$ can change over time t, such that at time $t_1 \in \mathbb{R}_{\geq 0}$ the data set $\mathcal{D}_{n(t_1)}$ with $N(n(t_1))$ training points exists.

Learning

Assumption 1: Consider an oracle with the predictions $\hat{\boldsymbol{f}}_n \in \mathcal{C}^2$ and $\hat{\boldsymbol{f}}_{\omega,n} \in \mathcal{C}^0$ based on the data set $\mathcal{D}_n$ (8). Let $S_{\mathcal{X}} \subset (SE(3) \times (\mathcal{X} \subset \mathbb{R}^6))$ be a compact set where the derivatives of $\hat{\boldsymbol{f}}_n$ are bounded on $\mathcal{X}$. There exists a bounded function $\bar{\rho}_n : S_{\mathcal{X}} \to \mathbb{R}_{\geq 0}$ such that, if $||\cdot||$ denotes the Euclidean norm, the prediction error is bounded by

$$P\left\{ \left\| \begin{bmatrix} \boldsymbol{f}(\boldsymbol{x}) - \hat{\boldsymbol{f}}_n(\boldsymbol{x}) \\ \boldsymbol{f}_\omega(\boldsymbol{s}) - \hat{\boldsymbol{f}}_{\omega,n}(\boldsymbol{s}) \end{bmatrix} \right\| \leq \bar{\rho}_n(\boldsymbol{s}) \right\} \geq \delta \tag{4}$$

with probability $\delta \in (0, 1]$ for all $\boldsymbol{x} \in \mathcal{X}, \boldsymbol{s} \in S_{\mathcal{X}}$.

Assumption 2: The number of data sets $\mathcal{D}_n$ is finite and there are only finitely many switches of $n(t)$ over time, such that there exists a time $T \in \mathbb{R}_{\geq 0}$ where $n(t) = n_{\text{end}}, \forall t \geq T$

Assumption 3: The kernel k is selected such that $\boldsymbol{f}, \boldsymbol{f}_\omega$ have a bounded reproducing kernel Hilbert space (RKHS) norm on $\mathcal{X}$ and $S_{\mathcal{X}}$, respectively, i.e. $\|f_i\|_k < \infty, \|f_{\omega,i}\|_k < \infty$ for all $i = 1, 2, 3$.

Model error Lemma

Consider the unknown functions $\boldsymbol{f}, \boldsymbol{f}_\omega$ and a GP model satisfying Assumption 3. The model error is bounded by

$$P\left\{\left\|\boldsymbol{\mu}\left(\left[\begin{matrix}\hat{\boldsymbol{f}}_n(\boldsymbol{x})\\ \hat{\boldsymbol{f}}_{\omega,n}(\boldsymbol{s})\end{matrix}\right]\middle|\boldsymbol{s},\mathcal{D}_n\right)-\left[\begin{matrix}\boldsymbol{f}(\boldsymbol{x})\\ \boldsymbol{f}_\omega(\boldsymbol{s})\end{matrix}\right]\right\| \leq \left\|{\boldsymbol{\beta}_n}^\top \Sigma^{\frac{1}{2}}\left(\left[\begin{matrix}\hat{\boldsymbol{f}}_n(\boldsymbol{x})\\ \hat{\boldsymbol{f}}_{\omega,n}(\boldsymbol{s})\end{matrix}\right]\middle|\boldsymbol{s},\mathcal{D}_n\right)\right\|\right\} \geq \delta$$

for $\boldsymbol{x} \in \mathcal{X}, \boldsymbol{s} \in S_\mathcal{X}, \delta \in (0,1)$ with $\boldsymbol{\beta}_n \in \mathbb{R}^6$

With Assumption 3 and the fact, that universals kernels exist which generate bounded predictions with bounded derivatives, GP models can be used as oracle to fulfill Assumption 1. In this case, the prediction error bound is given by $\bar{\rho}_n(\boldsymbol{s}) := \|{\boldsymbol{\beta}_n}^\top \Sigma^{\frac{1}{2}}([\hat{\boldsymbol{f}}_n(\boldsymbol{x})^\top, \hat{\boldsymbol{f}}_{\omega,n}(\boldsymbol{s})^\top]^\top|\boldsymbol{s},\mathcal{D}_n)\|$.

Under these assumptions and the model error Lemma, we can design a feedback controller based on backstepping method to exponentially track a desired trajectory.

Numerical Example 1

To demonstrate the application relevance of our proposed approach, we consider the task of an quadcopter to explore a terrain with unknown thermals.

The dynamics of the vehicle is described with mass $m = 1$ kilogram, inertia $J = I_3 kg/m^2$ and the direction $\boldsymbol{e} = [0, 0, 1]^\top$ of the force input u.

The data of the thermals is taken from publicly available paragliding data.

The thermals are assumed to act on the quadcopter as a disturbance in the direction of x_3, i.e., the altitude, as well as an angular momentum in the direction of ω_1. A GP model is then used as oracle to predict $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{f}_\omega(\boldsymbol{s})$ based on the collected data set with the squared exponential kernel.

Numerical Example 1

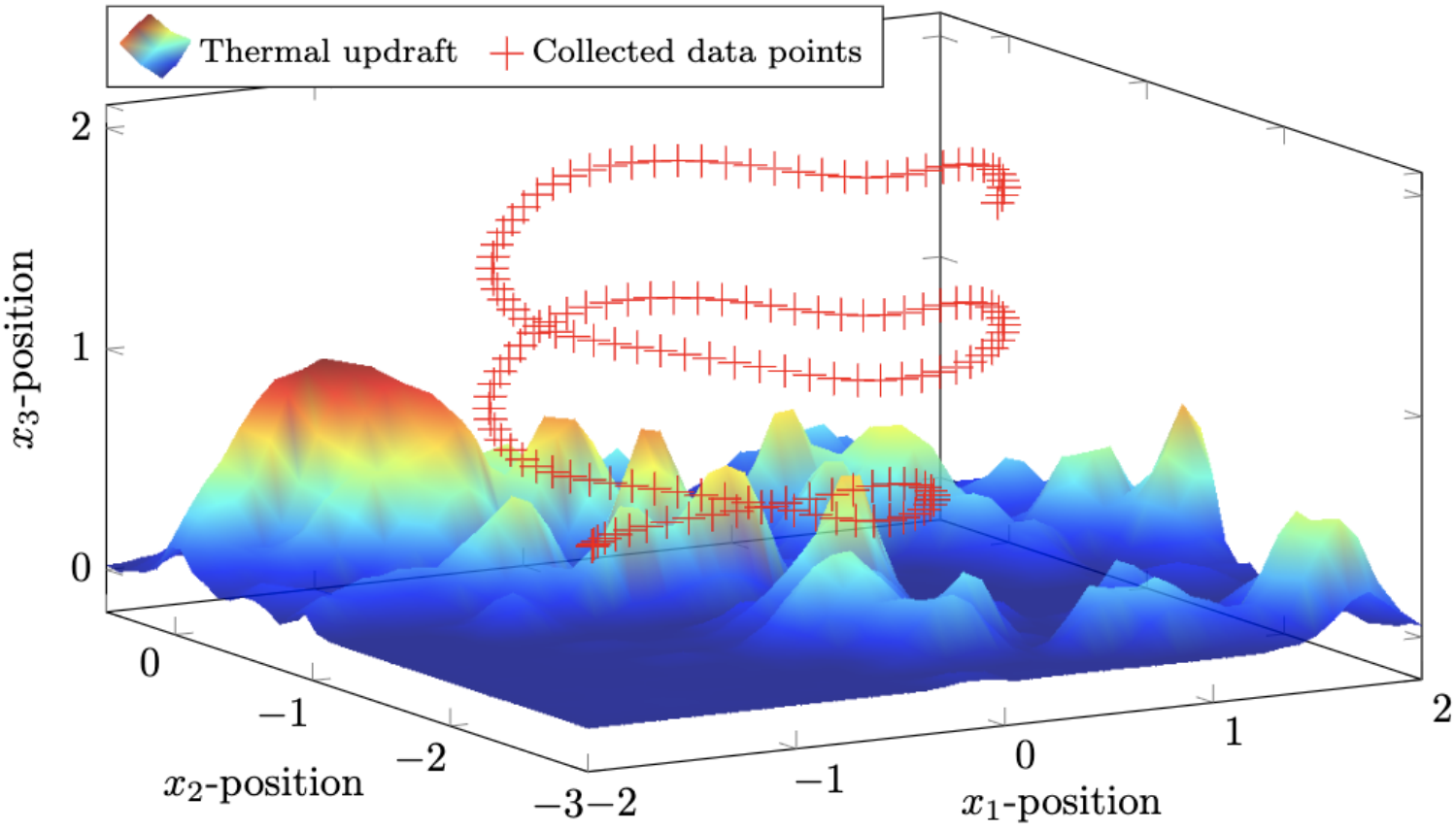


Figure: Visualization of the normalized magnitude of the thermal updraft acting on the quadcopter and the recorded training data points (red crosses).

Numerical Example 1

First, we start with the collection of training data for the GP model. For this purpose, the control inputs for the aerial vehicle are generated by a controller but without an oracle, i.e. $\hat{\boldsymbol{f}}(\boldsymbol{x}) = \hat{\boldsymbol{f}}_\omega(\boldsymbol{s}) = \boldsymbol{0}, \forall \boldsymbol{x} \in \mathbb{R}^6, \boldsymbol{s} \in \mathcal{S}$.

The desired trajectory is given by $x_{d,1}(t) = \sin(t), x_{d,2}(t) = \cos(t) - 1$, $x_{d,3}(t) = t/10$.

Every 0.1 second a training point has been recorded. Each training point consists of the actual state $\boldsymbol{s}$ and $\boldsymbol{y}$.

Since the training points depend on the typically noisy measurement of the accelerations $\ddot{\boldsymbol{p}}$ and $\dot{\boldsymbol{\omega}}$, a Gaussian distributed noise $\mathcal{N}(0, 0.08^2 I_3)$ is added to the measurement. After the simulation time of 15 seconds, the data set $\mathcal{D}$ consists of 150 training points. Based on this data set, a GP model is trained and the hyperparameters are optimized by means of the likelihood function.

Numerical Example 1

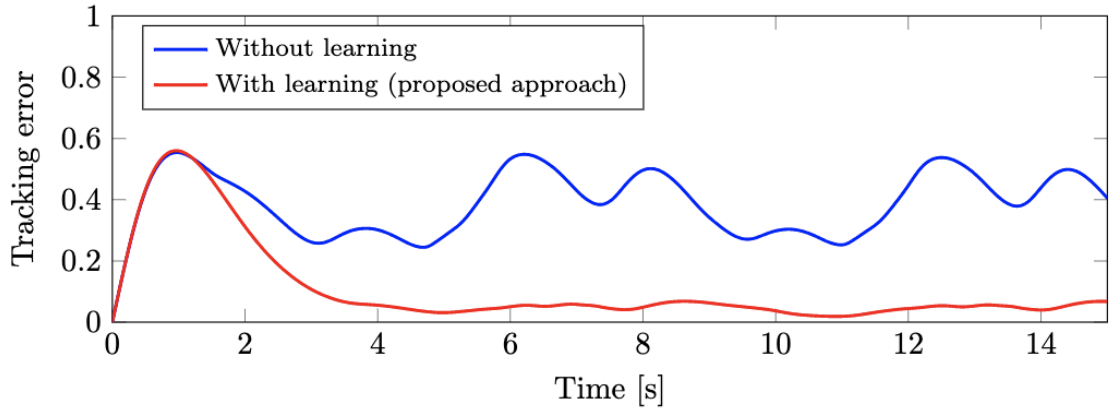


Figure: Tracking error of the quadcopter with control law $\boldsymbol{u}$ without learning (blue) and with our proposed learning-based approach (red).

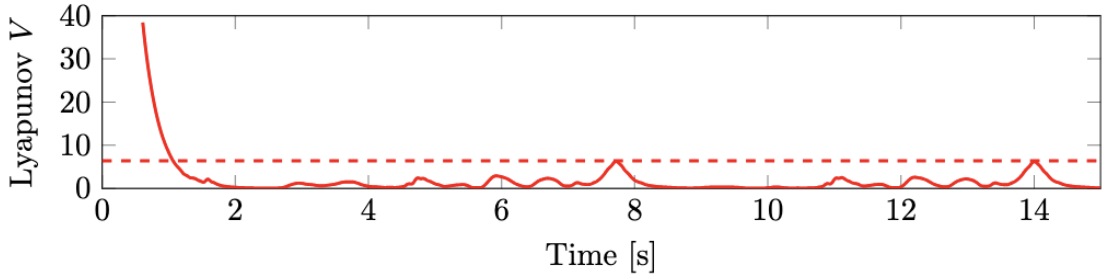


Figure: Lyapunov function (solid) converges to a tight set around zero (dashed line) and stays inside this set with high probability.

Numerical Example 1

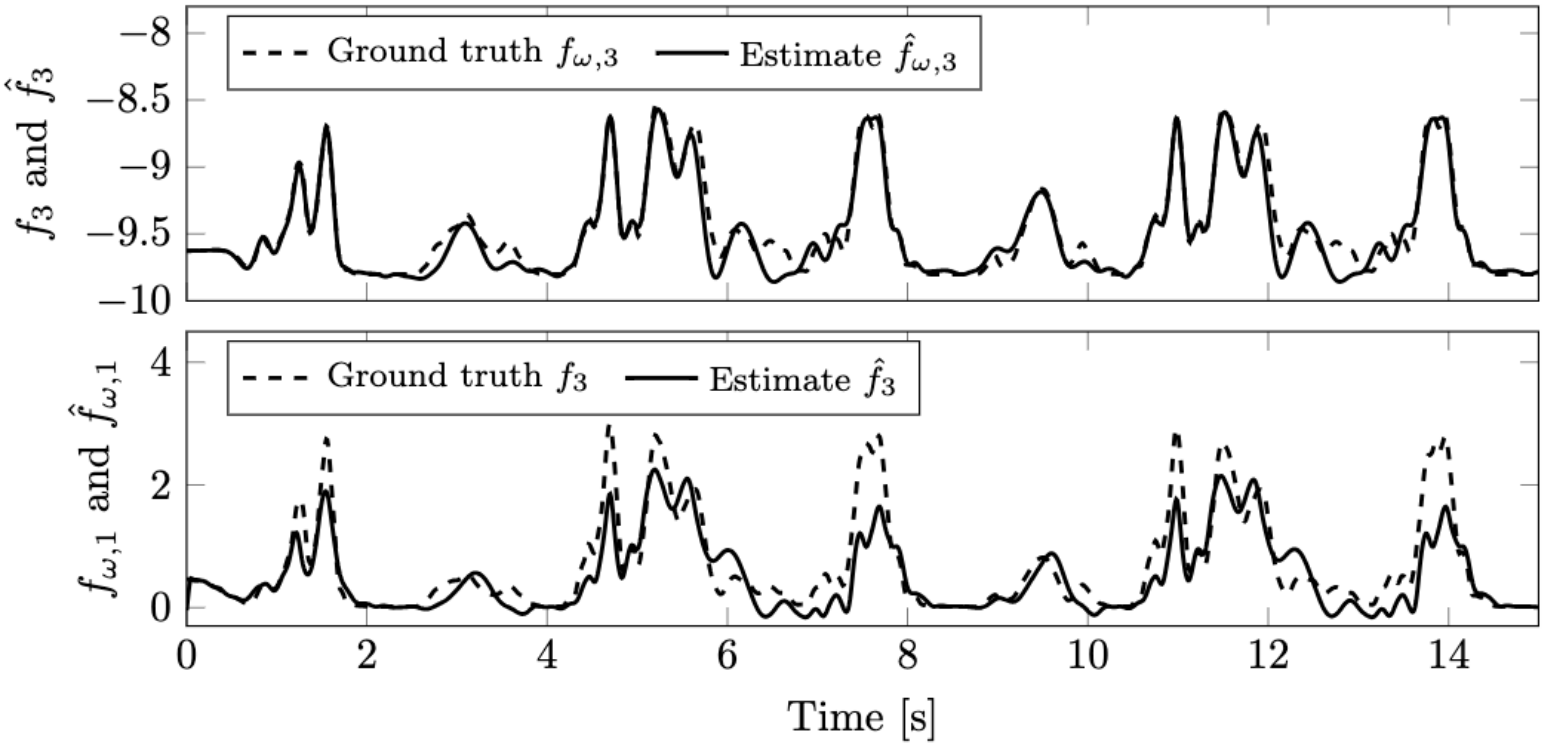


Figure: Ground truth (dashed) of the unknown dynamics and estimates of the GP (solid). Top: Unknown dynamics acting on the x_3-position of the quadcopter. Bottom: Unknown dynamics acting on the first component of the angular acceleration $\dot{\omega}$ of the quadcopter.

Numerical Example 2

The dynamics of the quadrocopter are described by (1) with mass $m = 1$ kilogram, inertia $J = \mathrm{diag}(2, 2, 1)kg/m^2$ and the direction $\boldsymbol{e} = [0, 0, 1]^\top$ of the force input u. As unknown dynamics $\boldsymbol{f}$ and $\boldsymbol{f}_\omega$, we consider an arbitrarily chosen wind field and the gravity force given by

$$\boldsymbol{f}(\boldsymbol{x}) = [0, 0, 2\sin(x_1) + \exp(-5x_2^2) - 9.81]^\top \tag{5}$$

$$\boldsymbol{f}_\omega(\boldsymbol{s}) = [2\exp(-x_1^2 - x_2^2) + \omega_1 \cos(x_2)^2, 0, 0]^\top. \tag{6}$$

At starting time $t = 0$, the data set $\mathcal{D}_n$ is empty such that the prediction is solely based on the mean function.

The initial position of the quadrocopter is $\boldsymbol{p}(0) = [0.1, -0.1, 0]^\top$ whereas the desired trajectory starts at $\boldsymbol{p}_d(0) = [0, 0, 0]^\top$ due to an assumed position measurement error. We employ an online learning approach which collects a new data point every 0.1 seconds such that the total number of data points is $N = 5n$.

The GP model is updated every second until $t = 12$ seconds, where the last 10 collected training points are appended to the set $\mathcal{D}_n$ and the hyperparameters are optimized by means of the likelihood function.

Numerical Example 2

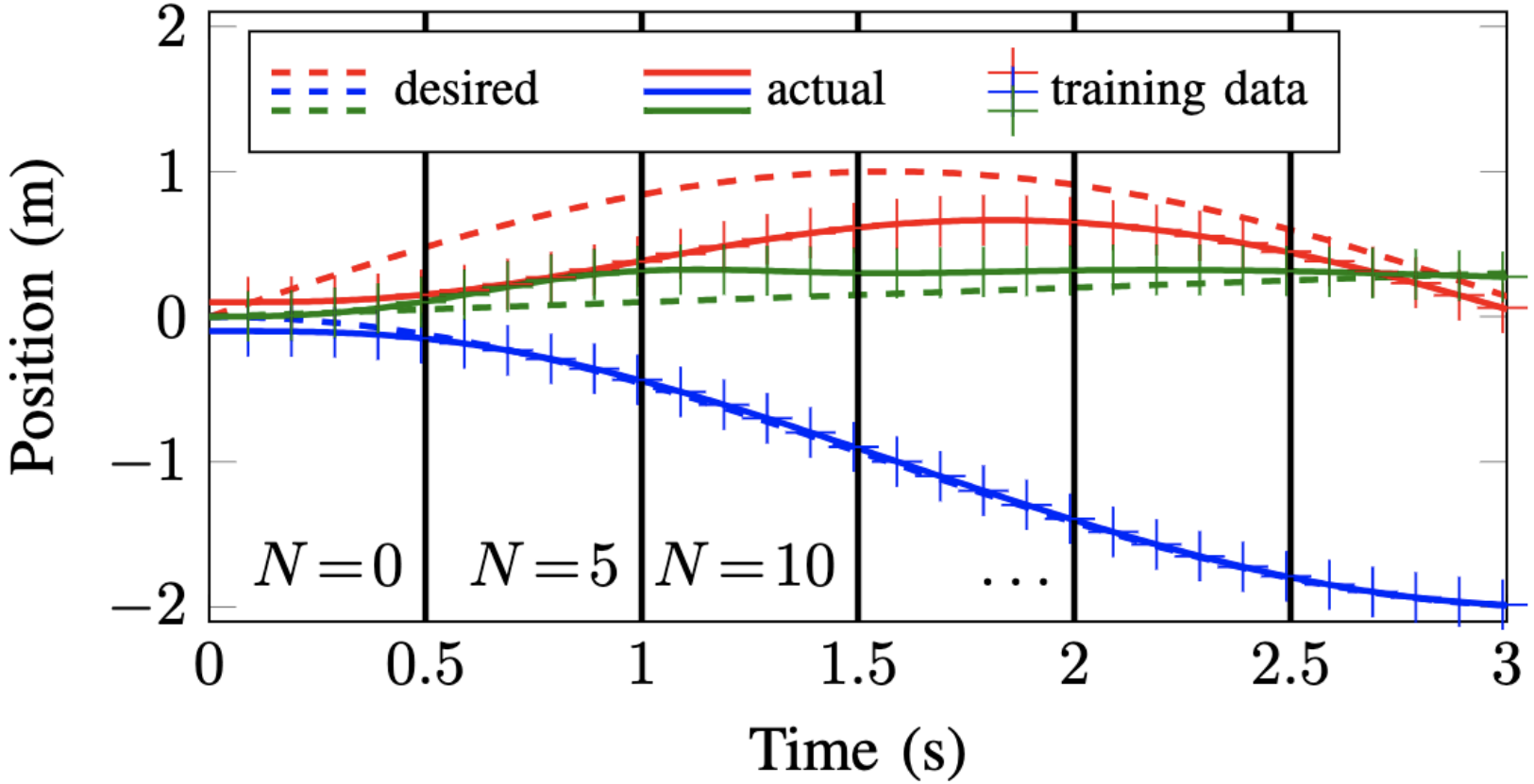


Figure: A segment of the desired (dashed) and actual trajectory (solid). Every 0.1 seconds a training point (cross) is recorded. Every 0.5 seconds the oracle is updated based on all collected training points N. The additional training data allows to refine the model such that the tracking error is decreasing.

Numerical Example 2

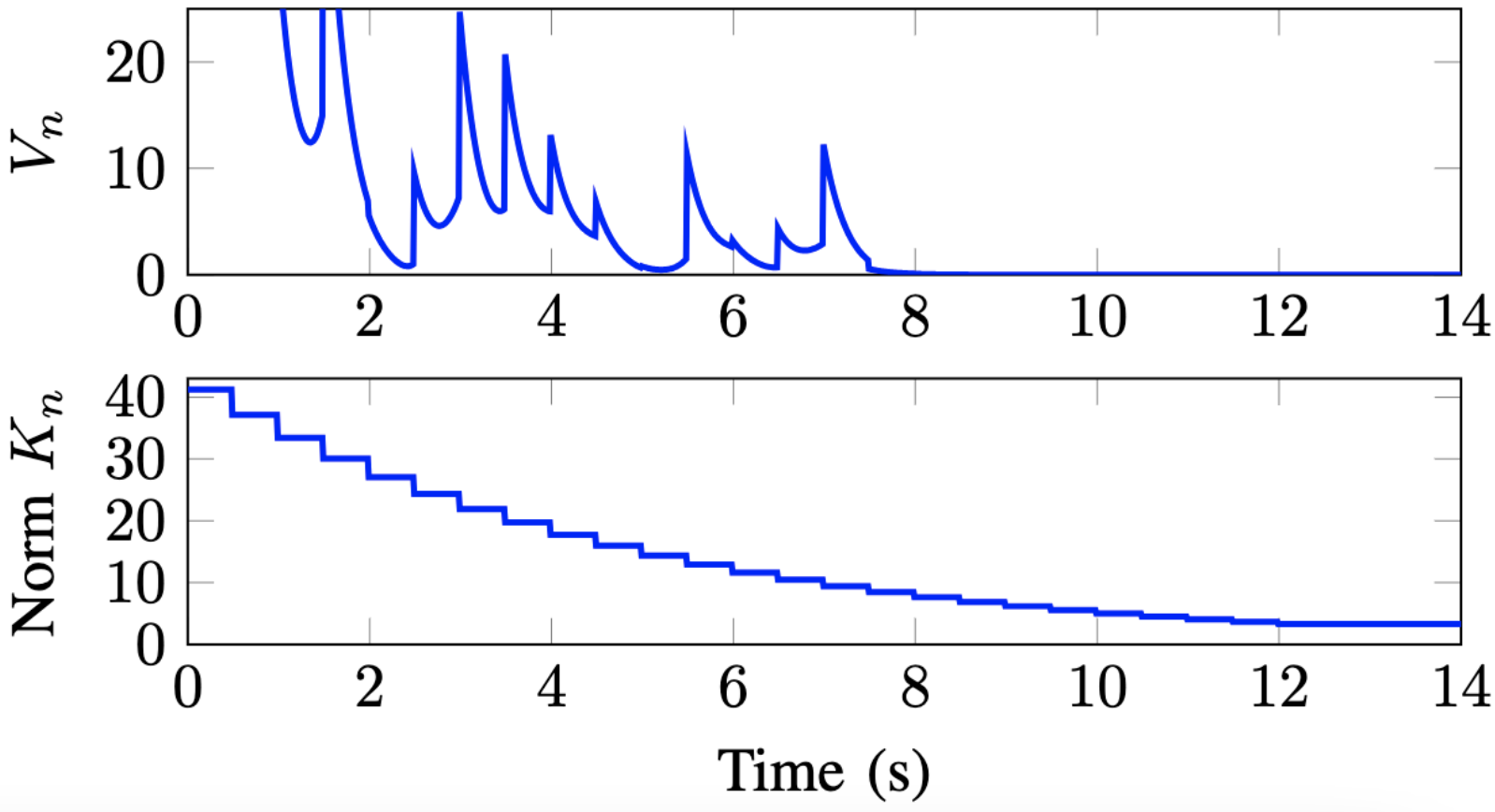


Figure: Top: Lyapunov function converges to a tight set around zero. The jumps occur when the oracle is updated. Bottom: Norm of the feedback gain matrix is decreasing due to improved accuracy of the oracle.

Numerical Example 2

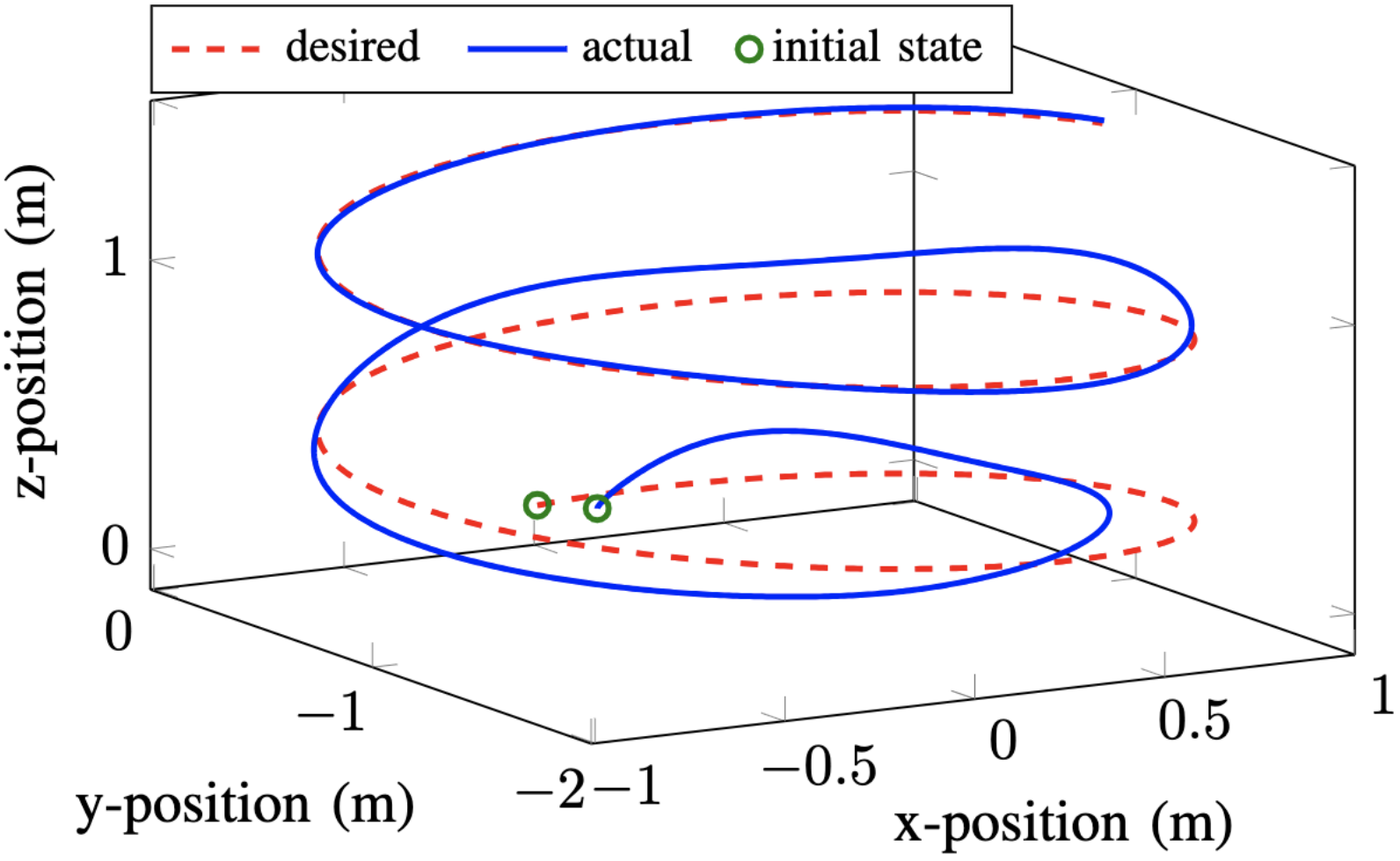


Figure: Actual trajectory converges to desired trajectory

Failures in the vehicles

Failures in the vehicles

Figure: Not only collisions with aircraft or obstacles are risks. What happens if a vehicle failure occurs? Is it possible to avoid an accident? What are the technologies to avoid accidents? The DJI company has tried to offer partial solutions to this problem in its products for professional use.

How we improve vehicle safety?


Human faults (users).


Failures in the software


Failures in sensors


Failures in batteries


Engine and/or propeller failures

Multirotors UAVs

Problem Setting

Assume a single rigid body on $\mathrm{SE}(3)$ with position $\boldsymbol{p} \in \mathbb{R}^3$ and orientation matrix $R \in \mathrm{SO}(3)$. The body-fixed angular velocity is denoted by $\boldsymbol{\omega} \in \mathbb{R}^3$ and the linear velocity by $\boldsymbol{v} \in \mathbb{R}^3$. The vehicle has mass matrix $m > 0$ and rotational inertia tensor $J \in \mathbb{R}^{3\times 3}$, symmetric and positive definite.

The state space of the vehicle is $Q = \mathrm{SE}(3) \times \mathbb{R}^6$ with $\boldsymbol{q} = ((R, \boldsymbol{p}), (\boldsymbol{\omega}, \boldsymbol{v})) \in Q$ denoting the entire state of the system. The vehicle is actuated with control input vectors $\boldsymbol{u}_1 \in \mathbb{R}^3$ and $\boldsymbol{u}_2 \in \mathbb{R}^3$, representing the 6D generalized actuation force acting on the system.

We can model the system with the following set of differential equations representing the kinematics of the rigid body and its uncertain dynamics

$$\begin{aligned} \dot{R} &= RS(\boldsymbol{\omega}),\ \dot{p} = R\boldsymbol{v}, \\ J\dot{\boldsymbol{\omega}} &= -\boldsymbol{\omega} \times J\boldsymbol{\omega} + \boldsymbol{u}_1 + \boldsymbol{f}_\omega(\boldsymbol{q}), \\ m\dot{\boldsymbol{v}} &= -\boldsymbol{\omega} \times m\boldsymbol{v} + mgRe_3 + \boldsymbol{u}_2 + \boldsymbol{f}_v(\boldsymbol{q}), \end{aligned} \tag{7}$$

where the operator $S\colon \mathbb{R}^3 \to \mathfrak{so}(3)$ is given by $S(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}$.

Problem Setting

The functions $\boldsymbol{f}_v : Q \to \mathbb{R}^3$ and $\boldsymbol{f}_\omega : Q \to \mathbb{R}^3$ are state-depended unknown dynamics. It is assumed that the full state $\boldsymbol{q}$ can be measured.

The general objective is to track a desired trajectory described by the functions $(R_d, \boldsymbol{p}_d) : [0, T] \to \mathrm{SE}(3)$.

Even though in this formulation we consider a fully actuated system, for some under-actuated systems a virtual control input can be defined to transform the system in a suitable form.

Suppose that $\hat{\boldsymbol{f}}$ is known model of the disturbances, then we can rewrite the equations and estimate $\hat{\boldsymbol{f}} - \boldsymbol{f}$ instead of $\boldsymbol{f}$.

For instance, if we want to include the aerodynamic model of the disturbances after tilting a rotor, and estimate the difference with respect to this aerodynamic model, we can do it in this way. Here, we assume that we do not have such a model and the problem is to estimate $\boldsymbol{f}$.

Fault-tolerant control systems

As it is shown in equation (7), the vehicle is controlled by a proper choice of control input $\boldsymbol{u}$, which is generated controlling the velocity of each six rotors.

Each rotor $i = 1, ..., 6$ is commanded with a Pulse-width modulation (PWM) signal $0 \leq \eta_i \leq 1$, with 0 corresponding to the rotor completed stop and 1 to its maximum speed.

There is a linear map A between the PWM signals $\boldsymbol{\eta} \in \mathbb{R}^6$ and the control signal $\boldsymbol{u}$. The matrix A depends on where each rotor is located, its orientation, the center of mass of the vehicle and the mechanical constants of the rotors and propellers.

Given a desired torque $\boldsymbol{u_1} = \boldsymbol{\tau}_{cmd} \in \mathbb{R}^3$ and thrust $\boldsymbol{u_2} = f_z$, the so-called control allocation problem is to find PWM signals $\boldsymbol{\eta} \in \mathbb{R}^6$ ($0 \leq \eta_i \leq 1$, with $i = 1, ..., 6$) such that $\boldsymbol{u} = A\boldsymbol{\eta}$.

Notice that each column i of the matrix A is the contribution of the rotor i to the control signal $\boldsymbol{u}$.

Fault-tolerant control systems

In several works, the fault-tolerant design for UAV is studied as a control allocation problem. More precisely, when a rotor fails, the corresponding column of the matrix A is replaced by zeros, since the contribution of this rotor to the control signal is null.

J. I. Giribet, R. S. Sanchez-Pena, and A. S. Ghersin. "Analysis and design of a tilted rotor hexacopter for fault tolerance." IEEE Trans. on aerospace and electronic systems 52.4 (2016): 1555-1567.

shows that given a control signal $\boldsymbol{u}$, for a hexacopter, it is possible to find a PWM signal $\boldsymbol{\eta}$ such that $\boldsymbol{u} = A\boldsymbol{\eta}$, even when one column of matrix A is replaced by zeros for any column.

When a failure occurs, the speed of each rotor is modified to achieve the same control signal $\boldsymbol{u}$ as when the vehicle was flying in nominal conditions.

The fault detection system must be fast enough in order to detect the failure and relocate the PWM signals to achieve the desired control signal $\boldsymbol{u}$.

Fault-tolerant control systems

Although it is possible to achieve torque in any direction under failure conditions, the magnitude is limited making the vehicle not suitable for real applications.

However, by adding a mechanism to tilt one rotor sideway in case of failure, a standard hexarotor vehicle can be converted into a robust fault-tolerant one.

While it is possible to maintain the ability to exert torque in any direction even with the failure of one rotor, in practice several factors affect the control performance after a failure, mainly because of non-linearities.

Abbaraju, P., Ma, X., Jiang, G., Rastgaar, M., & Voyles, R. M. Aerodynamic Modeling of Fully-Actuated Multirotor UAVs with Nonparallel Actuators. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

shows that the aerodynamic effects caused due to tilt angled propeller configurations has impact on the vehicle performance, and in particular characterize experimentally some effects that were not particularly considered before as the blade flapping effect for cant angled propellers.

Fault-tolerant control systems: distubrances

The authors proposed an aerodynamic model for these effects and could be compensated for, but wind tunnel experiments must be carried out, which is not an easy task.

Furthermore, it is not easy to isolate these effects and some others which we are not aware and are responsible for the unknown dynamics $\boldsymbol{f}$ in equation (7).

Although the control algorithm is robust to failures, it could be adapted to improve its performance to compensate the unknown disturbances $\boldsymbol{f}$, and for this a learning-based control technique is proposed here.

Learning with Gaussian Processes

For the compensation of the unknown dynamics of (7), we use GPs to estimate the values of $\boldsymbol{f}_v, \boldsymbol{f}_\omega$ for a given state $\boldsymbol{q}$. For this purpose, $N(n) : \mathbb{N} \to \mathbb{N}$ training points of the system (7) are collected to create a dataset

$$\mathcal{D}_{n(t)} = \{\boldsymbol{q}^{\{i\}}, \boldsymbol{y}^{\{i\}}\}_{i=1}^{N(n)}. \tag{8}$$

The output data $\boldsymbol{y} \in \mathbb{R}^6$ is given by $\boldsymbol{y} = [(m\dot{\boldsymbol{v}} + \boldsymbol{\omega} \times m\boldsymbol{v} - mgRe_3 - \boldsymbol{u}_2)^\top, (J\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times J\boldsymbol{\omega} - \boldsymbol{u}_1)^\top]^\top$ such that the first three components of $\boldsymbol{y}$ correspond to $\boldsymbol{f}_v$ and the remaining to $\boldsymbol{f}_\omega$.

The dataset $\mathcal{D}_{n(t)}$ with $n\colon \mathbb{R}_{\geq 0} \to \mathbb{N}$ can change over time t, such that at time $t_1 \in \mathbb{R}_{\geq 0}$ the dataset $\mathcal{D}_{n(t_1)}$ with $N(n(t_1))$ training points exists. This allows to accumulate training data over time.

The time-dependent estimates of the GP are denoted by $\hat{\boldsymbol{f}}_{v,n}(\boldsymbol{q})$ and $\hat{\boldsymbol{f}}_{\omega,n}(\boldsymbol{q})$ to highlight the dependence on the corresponding dataset $\mathcal{D}_n$.

Learning with Gaussian Processes

Assumption 1: The number of datasets $\mathcal{D}_n$ is finite and there are only finitely many switches of $n(t)$ over time, such that there exists a time $T \in \mathbb{R}_{\geq 0}$ where $n(t) = n_{\text{end}}, \forall t \geq T$.

Note that Assumption 1 is little restrictive since the number of sets is often naturally bounded due to finite computational power or memory limitations and since the unknown functions $\boldsymbol{f}_v, \boldsymbol{f}_\omega$ in (7) are not explicitly time-dependent, long-life learning is typically not required.

Therefore, there exists a constant dataset $\mathcal{D}_{n_{end}}$ for all $t > T_{end}$. Furthermore, Assumption 1 ensures that the switching between the datasets is not infinitely fast which is natural in real world applications.

Learning with Gaussian Processes

Assumption 2: Consider a Gaussian process with the predictions $\hat{\boldsymbol{f}}_{v,n}$ and $\hat{\boldsymbol{f}}_{\omega,n} \in \mathcal{C}^0$ based on the dataset $\mathcal{D}_n$. Let $Q_{\mathcal{X}} \subset (\mathrm{SE}(3) \times (\mathcal{X} \subset \mathbb{R}^6))$ be a compact set where $\hat{\boldsymbol{f}}_{v,n}, \hat{\boldsymbol{f}}_{\omega,n}$ are bounded on $Q_{\mathcal{X}}$. There exists a bounded function $\bar{\rho}_n : Q_{\mathcal{X}} \to \mathbb{R}_{\geq 0}$ such that, the prediction error is bounded by

$$P\left\{ \left\| \begin{bmatrix} \boldsymbol{f}_v(\boldsymbol{q}) - \hat{\boldsymbol{f}}_{v,n}(\boldsymbol{q}) \\ \boldsymbol{f}_\omega(\boldsymbol{q}) - \hat{\boldsymbol{f}}_{\omega,n}(\boldsymbol{q}) \end{bmatrix} \right\| \leq \bar{\rho}_n(\boldsymbol{q}) \right\} \geq \delta \tag{9}$$

with probability $\delta \in (0, 1]$, $\boldsymbol{q} \in Q_{\mathcal{X}}$.

Assumption 2 ensures that on each dataset $\mathcal{D}_n$, there exists a probabilistic upper bound for the error between the prediction $\hat{\boldsymbol{f}}_{v,n}(\boldsymbol{q}), \hat{\boldsymbol{f}}_{\omega,n}(\boldsymbol{q})$ and the actual $\boldsymbol{f}_v(\boldsymbol{q}), \boldsymbol{f}_\omega(\boldsymbol{q})$ on a compact set.

Learning with Gaussian Processes

Assumption 3: The kernel k is selected such that $\boldsymbol{f}_v, \boldsymbol{f}_\omega$ have a bounded reproducing kernel Hilbert space (RKHS) norm on $Q_{\mathcal{X}}$,
i.e., $\|\boldsymbol{f}_{v,i}\|_k < \infty$ and $\|\boldsymbol{f}_{\omega,i}\|_k < \infty$ for all $i = 1, 2, 3$.

Assumption 3 requires that the kernel must be selected in such a way that the functions $\boldsymbol{f}_v, \boldsymbol{f}_\omega$ are elements of the associated RKHS. This sounds paradoxical since this function is unknown. However, there exist some kernels, namely universal kernels, which can approximate any continuous function arbitrarily precisely on a compact set such that the bounded RKHS norm is a mild assumption.

For the later stability analysis of the closed-loop system, we introduce the following assumptions. In addition, we implicitly assume i.i.d data.

Learning with Gaussian Processes: Model error

Lemma

Consider the unknown functions $\boldsymbol{f}_v, \boldsymbol{f}_\omega$ and a GP model satisfying Assumption 3. The model error is bounded by

$$P\left\{ \left\| \boldsymbol{\mu}\left(\begin{bmatrix} \hat{\boldsymbol{f}}_{v,n}(\boldsymbol{q}) \\ \hat{\boldsymbol{f}}_{\omega,n}(\boldsymbol{q}) \end{bmatrix} \middle| \boldsymbol{q}, \mathcal{D}_n \right) - \begin{bmatrix} \boldsymbol{f}_v(\boldsymbol{q}) \\ \boldsymbol{f}_\omega(\boldsymbol{q}) \end{bmatrix} \right\| \right.$$
$$\left. \leq \left\| \boldsymbol{\beta}_n^\top \, \Sigma^{\frac{1}{2}} \left(\begin{bmatrix} \hat{\boldsymbol{f}}_{v,n}(\boldsymbol{q}) \\ \hat{\boldsymbol{f}}_{\omega,n}(\boldsymbol{q}) \end{bmatrix} \middle| \boldsymbol{q}, \mathcal{D}_n \right) \right\| \right\} \geq \delta$$

for $\boldsymbol{q} \in Q_{\mathcal{X}}, \delta \in (0,1)$ with $\boldsymbol{\beta}_n \in \mathbb{R}^6$,

$$(\boldsymbol{\beta}_n)_j = \sqrt{2 \left\| \rho_j \right\|_k^2 + 300 \gamma_j \ln^3 \left(\frac{N(n)+1}{1-\delta^{1/6}} \right)}. \tag{10}$$

The variable $\gamma_j \in \mathbb{R}$ is the maximum information gain

$$\gamma_j = \max_{\boldsymbol{q}^{\{1\}}, \ldots, \boldsymbol{q}^{\{N(n)+1\}} \in Q_{\mathcal{X}}} \frac{1}{2} \log \left| I + \sigma_j^{-2} K\left(\boldsymbol{x}, \boldsymbol{x}'\right) \right| \tag{11}$$

$$\boldsymbol{x}, \boldsymbol{x}' \in \left\{ \boldsymbol{q}^{\{1\}}, \ldots, \boldsymbol{q}^{\{N(n)+1\}} \right\}. \tag{12}$$

Control design

We prove the stability of the closed-loop with a proposed control law with multiple Lyapunov functions, where the n-th function is active when the GP predicts based on the corresponding training set $\mathcal{D}_n$.

Position controller: Let $\boldsymbol{p}_d \in \mathbb{R}^3$ be the desired position. Define the position error by $\boldsymbol{e} = R^T(\boldsymbol{p} - \boldsymbol{p}_d) \in \mathbb{R}^3$. By differentiation the latter with respect to time, the error dynamics can be written as

$$\dot{\boldsymbol{e}} = -S(\boldsymbol{\omega})\boldsymbol{e} + \boldsymbol{v}.$$

Let $\boldsymbol{z} \in \mathbb{R}^3$ be an error signal representing the difference between the desired and actual linear velocities, $\boldsymbol{z} = \boldsymbol{v} - \boldsymbol{v}_d \in \mathbb{R}^3$. Consider the Lyapunov function $V_{1,n} : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}_{\geq 0}$,

$$V_{1,n}(\boldsymbol{e}, \boldsymbol{z}) = \frac{1}{2}||\boldsymbol{e}||^2 + \frac{1}{2}m\boldsymbol{z}^T\boldsymbol{z} \geq 0$$

and note that $\Lambda_1||\boldsymbol{\zeta}||^2 \leq V_{1,n}(\boldsymbol{e}, \boldsymbol{z}) \leq \Lambda_2||\boldsymbol{\zeta}||^2$, where $\boldsymbol{\zeta} = [\boldsymbol{e}^T, \boldsymbol{z}^T]^T$, $\Lambda_1 = \frac{1}{2}\mathsf{min}\{1, m\}$, $\Lambda_2 = \frac{1}{2}\mathsf{max}\{1, m\}$.

Control design

By differentiating $V_{1,n}$ with respect to the time along the trajectories of the system, using the expresions for $\boldsymbol{e}, \dot{\boldsymbol{e}}$, $\boldsymbol{y}, \dot{\boldsymbol{y}}$, the fact that $S(\boldsymbol{\omega})$ is skew-symmetric, and (7), we obtain that

$$\begin{aligned}\dot{V}_{1,n}(\boldsymbol{e}, \boldsymbol{z}) =& \boldsymbol{e}^\top \boldsymbol{v}_d \\ &+ \boldsymbol{z}^\top \{\boldsymbol{e} - S(\boldsymbol{\omega}) m \boldsymbol{v} + \boldsymbol{f}_v + \boldsymbol{u}_2 - m \dot{\boldsymbol{v}}_d\}.\end{aligned} \tag{13}$$

We design the desired velocity as $\boldsymbol{v}_d = -k_1 \boldsymbol{e}$, and the position controller as

$$\begin{aligned}\boldsymbol{u}_2 =& -k_2 \boldsymbol{z} - \boldsymbol{e} + S(\boldsymbol{\omega}) m \boldsymbol{v} \\ &- k_1 m (S(\boldsymbol{\omega}) \boldsymbol{e} + \boldsymbol{v}) - \mu(\boldsymbol{f}_{v,n} \mid \boldsymbol{q}, \mathcal{D}_n),\end{aligned} \tag{14}$$

where $k_1, k_2 \in \mathbb{R}_{>0}$ are controller gains to be tuned.

Control design

Consider the system (7) and a GP model trained with (8) satisfying assumptions $1, 2$ and 3. The position control law (14) guarantees that the tracking error $\boldsymbol{\zeta}$ is uniformly ultimately bounded in probability by

$$P\left\{\|\boldsymbol{\zeta}(t)\| \leq \sqrt{\frac{\Lambda_2}{\Lambda_1}}\max_{\boldsymbol{q}\in Q_{\mathcal{X}}}\ \bar{\rho}_{n_{\text{end}}}(\boldsymbol{q}), \forall t \geq T\right\} \geq \delta \tag{15}$$

for all $\boldsymbol{q}$ on $Q_{\mathcal{X}}$, $\delta \in (0, 1)$ with $T \in \mathbb{R}_{\geq 0}$, and exponentially converges to zero.

L. Colombo, J. Giribet.Learning-Based Fault-Tolerant Control for an Hexarotor with Model Uncertainty. In IEEE Transactions on Control Systems Technology, 2024.

Exprerimental results

Next, the control algorithm proposed above is applied to a fault tolerant hexarotor vehicle. The main objective is to show that the GP estimates allow to improve the performance of the control after a failure

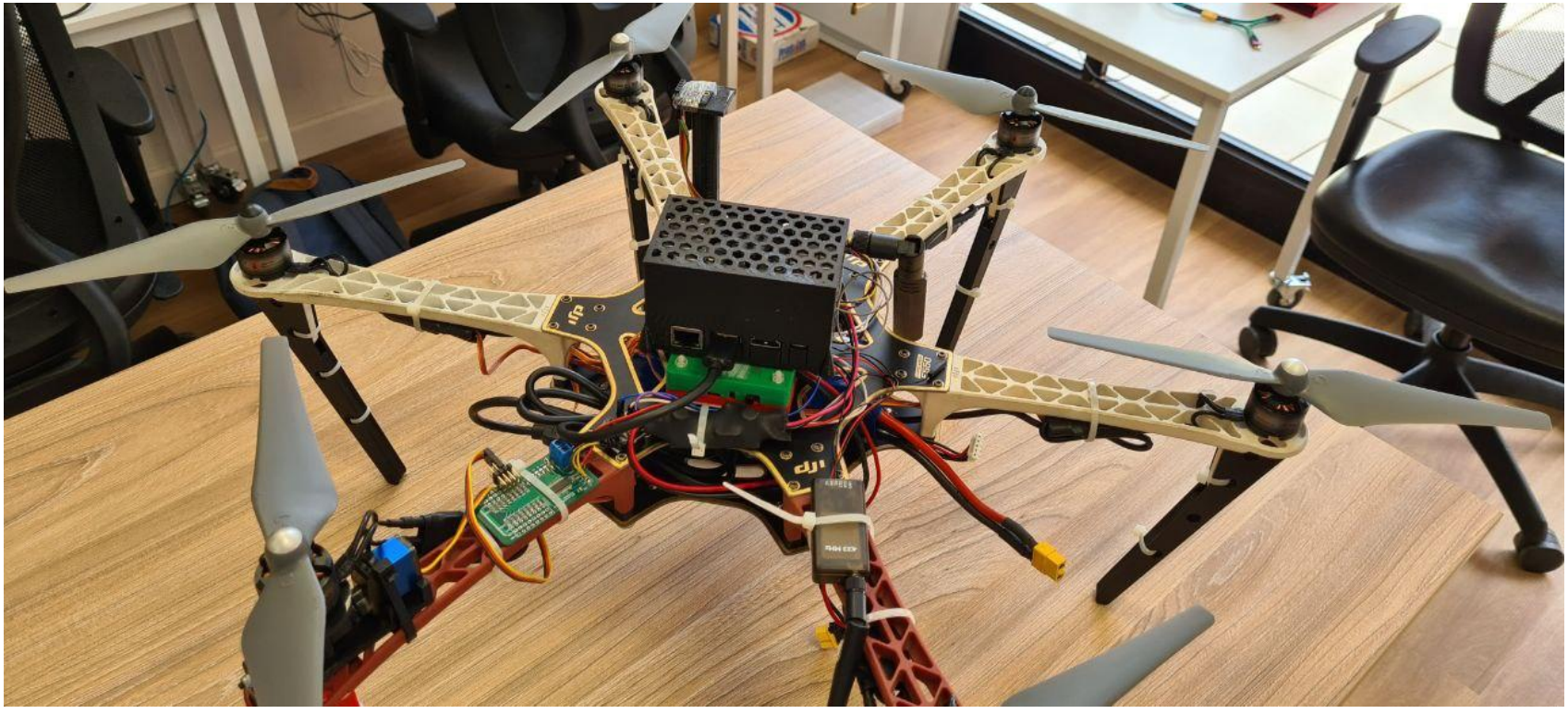


Figure: Hexarotor fault-tolerant vehicle. On the bottom-left arm, a servo allows to tilt the re-configurable motor. Two computers can be seen: inside a black case, the NVIDIA Jetson TX2, and inside a green case, the flight controller.

Experimental results

Experimental results

The low-level computer is responsible for executing critical algorithms such as attitude control, fault detection and identification (FDI), as well as applying low-pass and signal conditioning filters.

The high-level computer is tasked with running the position control algorithm and the GP-based estimation algorithm.

It receives sensor data from the low-level computer at a rate of 100Hz, including navigation data (attitude, position, and velocity) and angular velocity information filtered through a low-pass filter to reduce measurement noise.

While this filtering may theoretically affect the assumption of independent and identically distributed (iid) Gaussian noise, the algorithm exhibits satisfactory performance in practical applications.

It is worth mentioning that, failure detection scheme, as well as the control relocation scheme, is independent of the GP.

Experimental results

In the first experiment, the GP estimates are not used in the control loop. The control algorithm is the same in nominal conditions and after the failure occurs, the control allocation matrix is capable of adapting the 6 PWM signals $\boldsymbol{\eta} \in \mathbb{R}^{\mathbf{6}}$, to achieve the desired control signal.

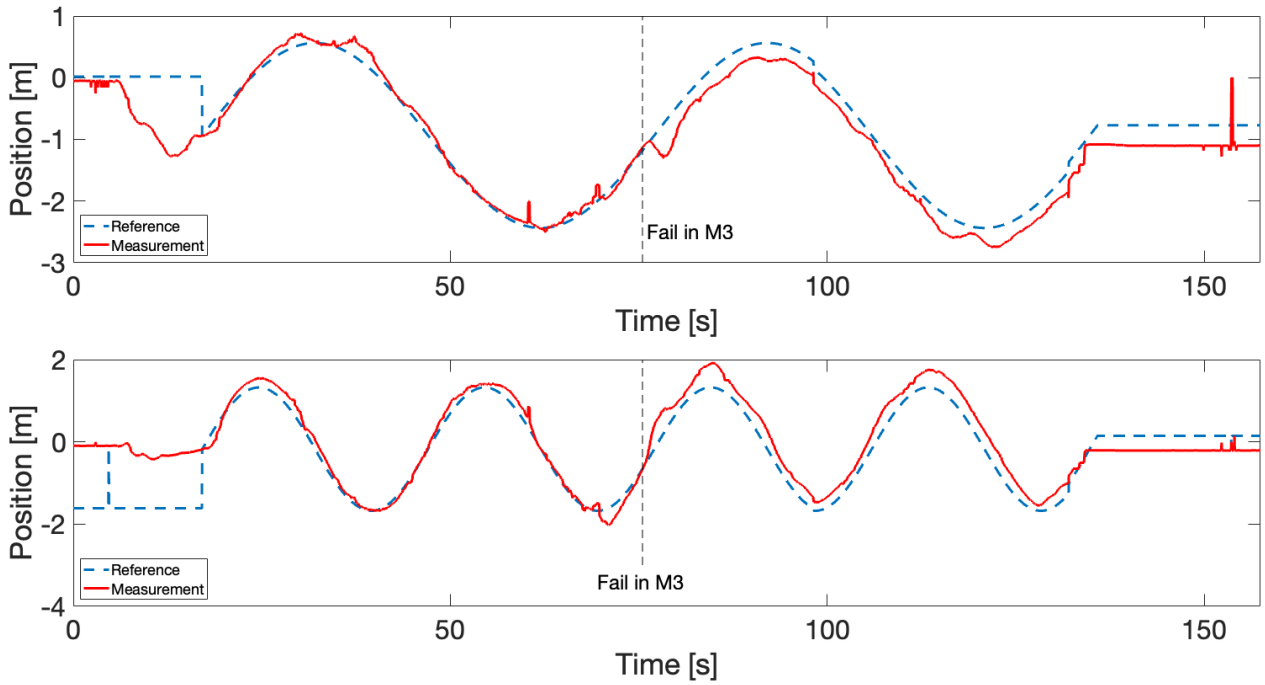


Figure: Trajectory tracking with a fault-tolerant hexarotor. The blue dot line is the reference trajectory. In red (solid) line is the UAV meassured position. A fail is activated in Rotor 3 (M3) at 75sec. It can be noted a fail in motor 3 is activated, and then the tracking performance is degraded.

Experimental results

To test how the control algorithm performs when the GP estimates are used to compensate the disturbances a second experiment was carried out in outdoor conditions

In these experiments, the GP receives data from the low-level computer and creates a dataset of 500 samples. The GP corrections is activated after the failure is detected and compensated. A squared exponential (SE) kernel was used.

In this study, we initially tuned the hyperparameters of the GP using a numerical simulator that we developed.

This allowed us to optimize the model's performance and capture the underlying dynamics of the system. However, when transitioning to experimental tests, we found it necessary to make adjustments to these hyperparameters.

This adaptation was crucial to ensure that the GP model effectively mitigated the effects of model disturbances in the real-world scenarios.

Experimental results

After the failure, the value $\hat{\boldsymbol{f}}_{v,n}(\boldsymbol{q})$ detects a change in the disturbances, which is consistent with a degradation in the control system performance.

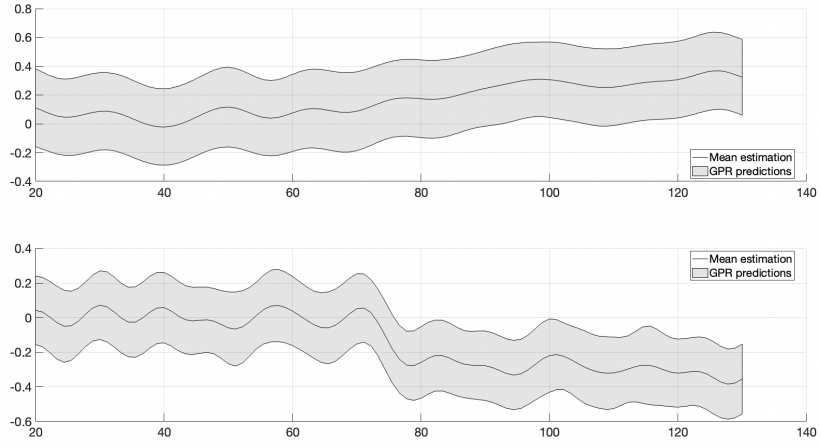


Figure: GP estimation of the horizontal components of the perturbations $\hat{\boldsymbol{f}}_{v,n}$. The solid line is the mean and the shadow represents the 95% prediction interval. The GP perturbation estimates are not used as a feedback in the control loop, just for estimating the disturbances.

The source of this perturbation is not clear, for instance an error in the tilting angle, aerodynamics perturbations, among others could be affecting the vehicle. But, it is not relevant here to determine the source or sources, but estimate the resultant effect.

Experimental results

By iteratively refining the hyperparameters based on empirical observations, we achieved a better alignment between the GP model and the experimental data, further enhancing the resilience of our approach.

In order to validate the performance of the control algorithm a flight was performed with a trajectory as in the outdoor case, and a rotor failure injected during the flight.

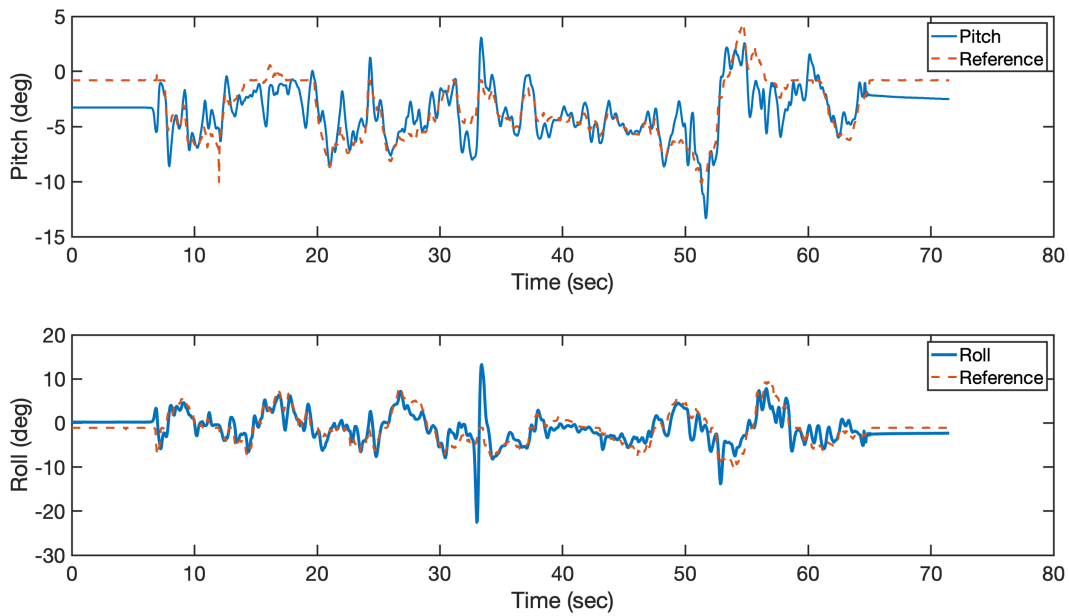


Figure: Orientation of the hexarotor, for a failure occurring during the flight. Vehicle with GP estimates compensation.

Experimental results

It can be noted that, at approximately $35sec$ a failure is introduced and the vehicle recovers stability after a rotor reconfiguration.

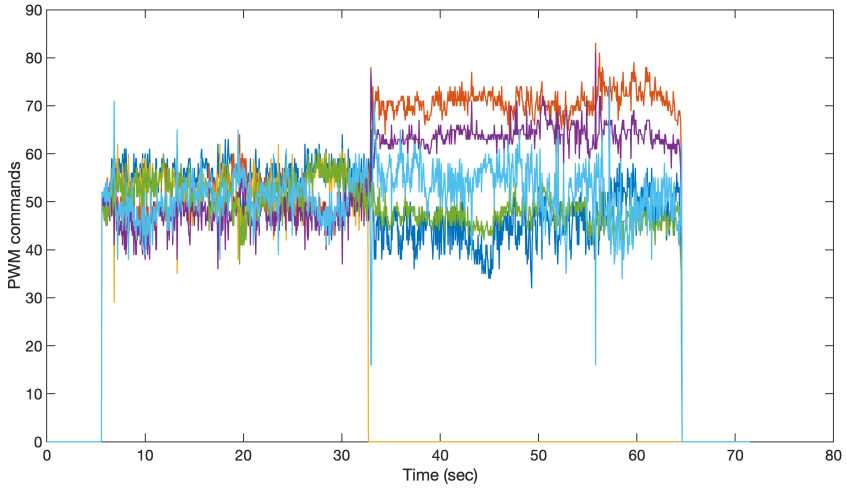
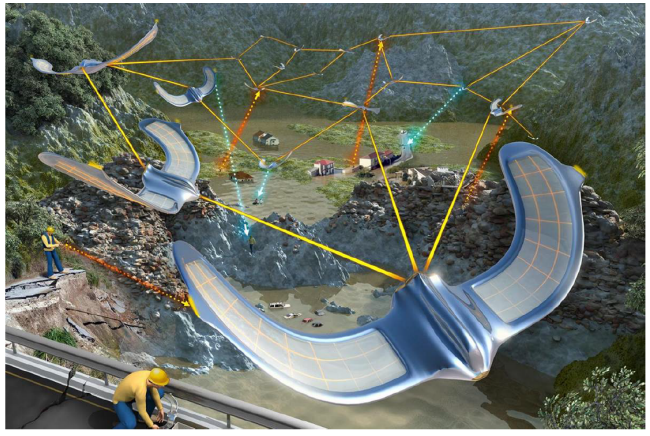


Figure: PWM signals of the vehicle, for a failure occurring during the flight. Vehicle with GP estimates compensation.

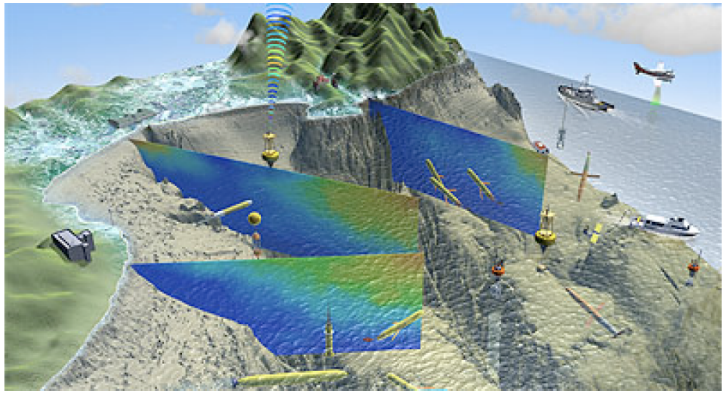
In this figure, it can be noticed that, when rotor 3 stops (yellow line) the remaining rotors increase the velocities, in order to compensate for the failure.

Robotic Swarms are coming!



Laboratory of Intelligent Systems, EPFL

Image: NBC News



ASAP, Leonard et al

Persistent Surviellance

over geographically extended areas, situational awareness, suggested route for emergency services.

Environmental Monitoring

ocean dynamics sampling, evaluating the impact of pollutants on biological populations, validation of climate models, wildlife monitoring, storm modeling and prediction.

How do we control swarms?

Challenges to go beyond *the centralized paradigm of the coordination*.

- Individual capacities are limited.
- The information is local, partial and may be erroneous.
- Distributed interactions (communications) prone to failure.
- Heterogeneous dynamics, multiple scales.

Autonomy engineering

How to coordinate individual elements into an overall coherent?

- From top to bottom: design coordination to design the desired behavior.
- from bottom to top: global behavior of local interaction rules

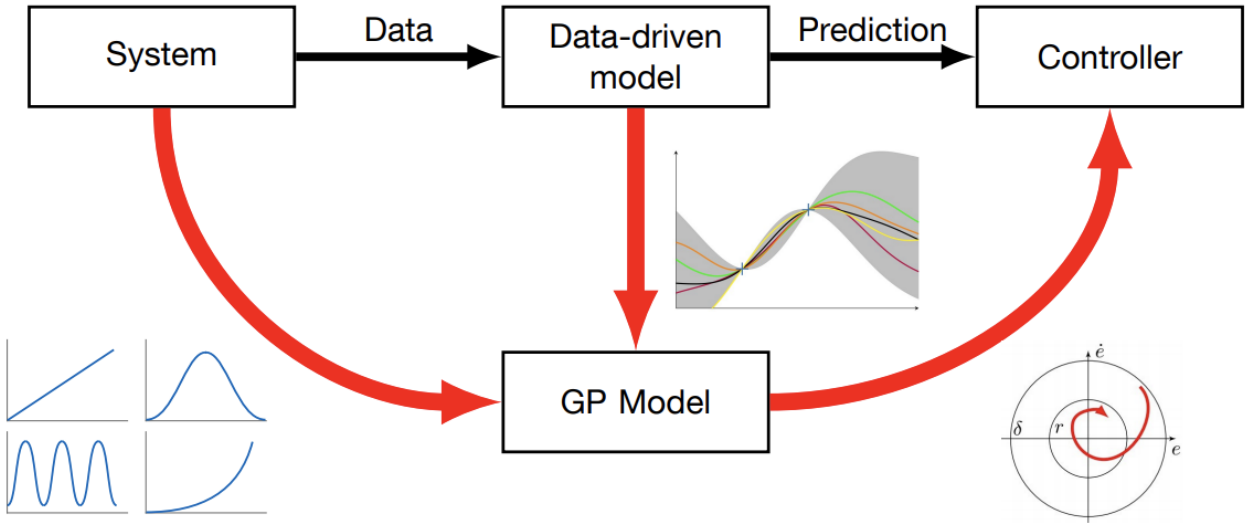
Limitations for the coordination of robotic swarms

Robotic networks

Limitations

- Computational / memory: precision with respect to the physical system. Global tasks hard to compute/store.
- Physical modeling: Heterogeneous robots, multiple scales, unknown dynamics.
- Comunication: decentralized, asynchronous.
- Sensorial: Noise, partial (e.g., lack of information on the position)

Data-driven control with GPR



There are no data-driven control laws for complex, multi-agent cooperative systems with safety guarantees.

Research line founded by *Leonardo Fellowship for researhers and cultural creators of the BBA Foundation*. Project: "Safety guarantees with data-driven control for cooperative systems."

Fundación BBVA

Decentralized and almost-decentralized coordination strategies

- Decentralized formation control strategies

T. Beckers, S. Hirche, L. Colombo. Safe Online Learning-based Formation Control of Multi-Agent Systems with Gaussian Processes. Proceedings of the 60th. Conference on Decision and Control, 2021.

- Flocking control (almost-decentralized)

T. Beckers, G. Pappas, L. Colombo. Learning rigidity-based flocking control using Gaussian Processes with probabilistic stability guarantees. Proceedings of the 61st Conference on Decision and Control, 7254-7259, 2022.