+

Non-Linear Model Predictive Control for Virtual Motorcycle Riding

E. Picotti, M. Bruschetta
Università di Padova

2024 VI-Grade ZPS

Introduction
Virtual Rider

- Virtual prototyping tools are widely used in automotive industry
  - Reduction of time-to-market
  - Reduction of costs for real prototypes
  - Increased safety
- Human-like virtual riders are needed for simulations, and several solutions have been studied in the late years
  - in [1], MPC has been applied to the steering control of a bike
  - in [2], a family of linear models is generated to be used in MPC at the different working conditions
  - in [3], an MPC is combined with a feedforward action based on quasi-steady state analysis of the bike during the reference maneuver

[1] J. S. Rowell, A. A. Popov, and J. P. Meijaard, "Predictive control to modelling motorcycle rider steering," International journal of vehicle systems modelling and testing , vol. 5, no. 2-3, pp. 124–160, 2010
[2] M. Massaro, "A nonlinear virtual rider for motorcycles," Vehicle system dynamics, vol. 49, no. 9, pp. 1477–1496, 2011
[3] D. M. Giner, Symbolic-numeric tools for the analysis of motorcycle dynamics: development of a virtual rider for motorcycles based on model predictive control. PhD thesis, Universidad Miguel Hernández de Elche, 2016.

+

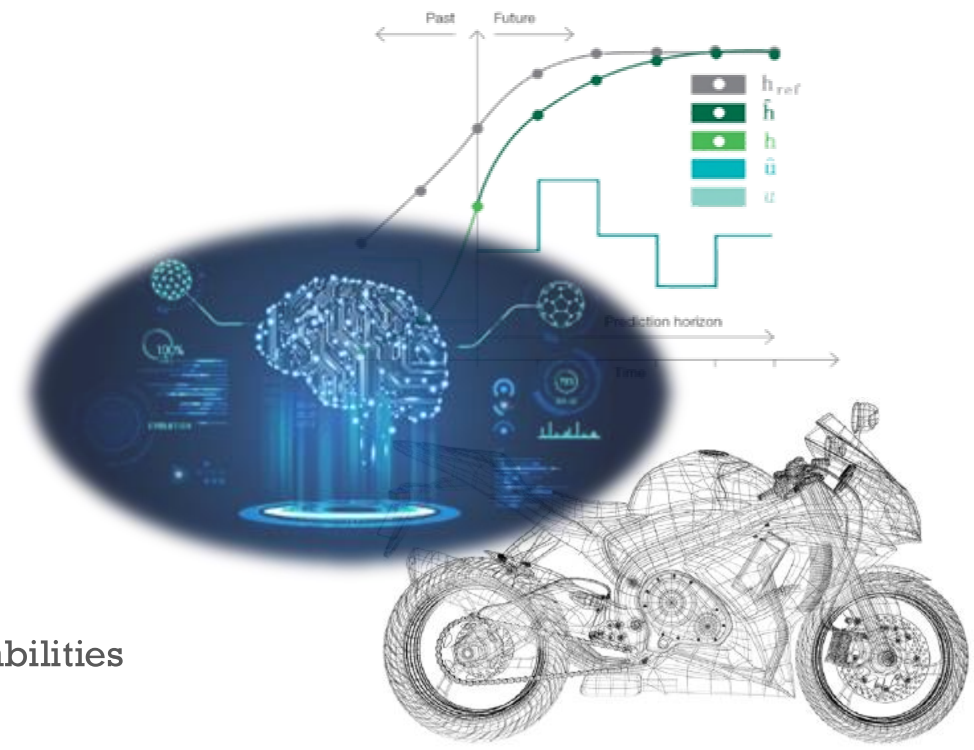
Introduction
Motorcycle Rider

- Rider is necessary
  - Intrinsic instability of the vehicle
  - Impossibility to perform reliable open-loop tests
- Rider presence influences motorcycle dynamics
  - Rider (and passenger/load) mass is comparable with the motorcycle mass
  - Rider steering stiffness perturbs the motorcycle oscillation modes
- Riding style is highly significant
  - Rider leaning behaviour affects the rolling and cornering dynamics
  - Rider oversteering behaviour at cornering exit is substantial

Introduction
Our work

- Virtual Rider based on Non-linear Model Predictive Control
  - Analytical model based on Lagrangian formulation
  - Rider inputs are steering, throttle, brake, rider lateral position
    - Rider represented as point mass with lateral DoF
    - Possibility to enforce rider movement behaviour

- Virtual Rider based on Learning-based NMPC schemes
  - Data-driven Tuning Strategy through Genetic Algorithms
    - NMPC tuning obtained by an algorithmic procedure that minimizes lap-time
  - Learning Dynamics Strategy through Gaussian Process
    - NMPC model enhanced by data-driven techniques to improve tracking capabilities



Design and implementation of a high-performance Nonlinear MPC-based virtual motorcycle rider

E. Picotti[1], M. Bruschetta[1], A. De Simoi [4], Y. Chen[2], A. Beghi[1], M. Nishimura[3], Y. Tezuka[3], F. Ambrogi[4]

[1]Department of Information Engineering, Università di Padova, email: **enrico.picotti@dei.unipd.it**

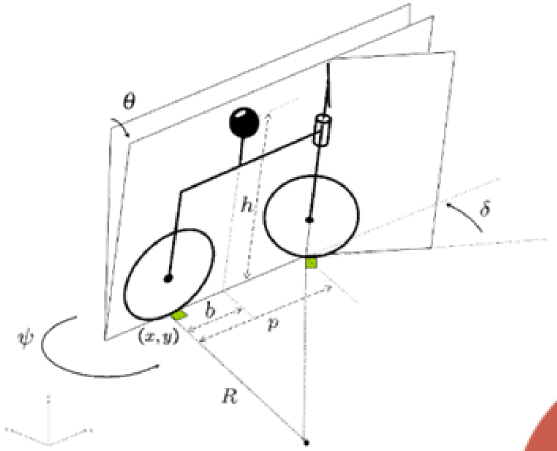
[2]Department of Electrical Engineering, Eindhoven University of Technology

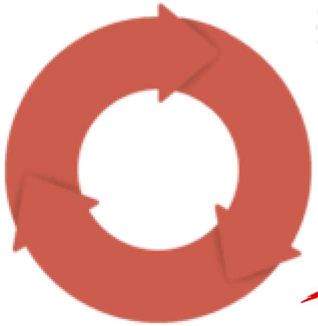
[3]Honda Racing Corporation, Senzui, Japan

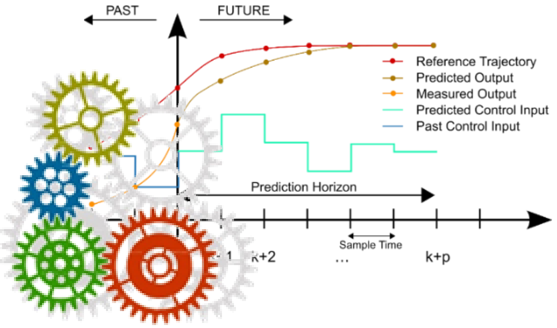
[4]VI-Grade, Via Galileo Galilei, Tavagnacco, Udine, Italy

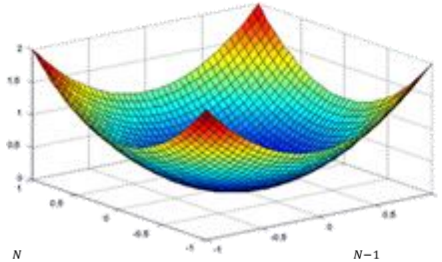
IEEE Transactions on Control Systems Technology, September 2021

NMPC Virtual Rider
NMPC structure

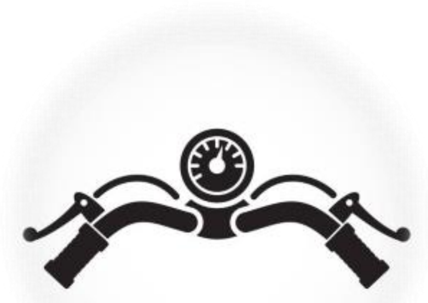


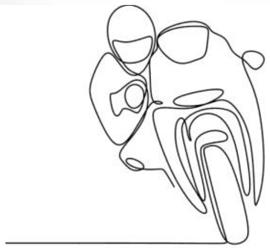






$$\sum_{k=0}^{N}\left(l_k(x_k,u_k)-l_{ref}\right)^T \boldsymbol{Q}\left(l_k(x_k,u_k)-l_{ref}\right)+\sum_{k=0}^{N-1} u_k^T \boldsymbol{R} u_k$$





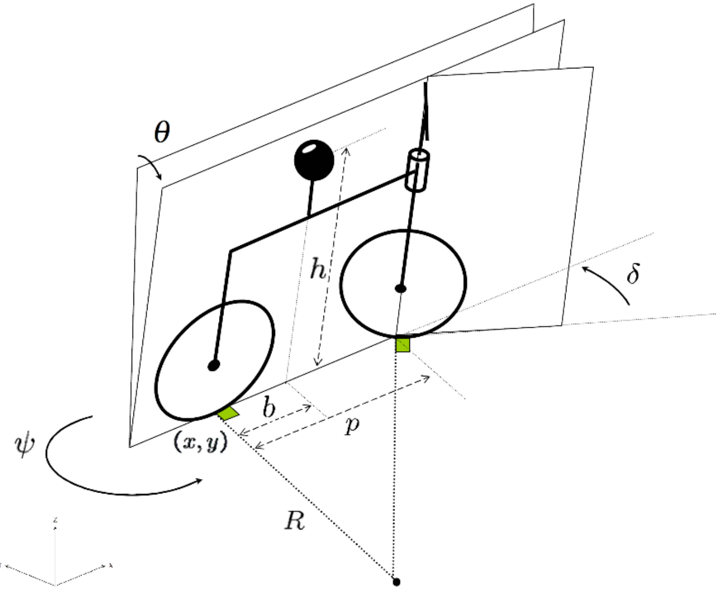
NMPC Virtual Rider
Motorcycle-rider model

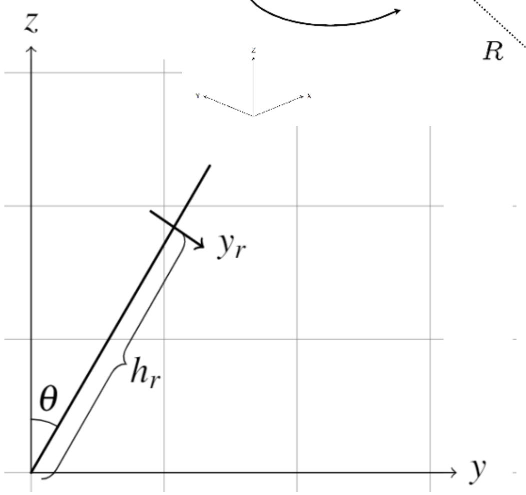
- Model characteristics
  - Sliding plane model with moving mass representing the rider
  - Linear longitudinal forces and Pacejka lateral forces
  - Dynamic LT model with longitudinal drag force
  - Gyroscopic effects

- Cost function terms

$$h = [e_\psi,\ e_y,\ \theta,\ v_x,\ v_y,\ \dot{\psi},\ \dot{\theta},\ \delta,\ \gamma_t,\ \gamma_b,\ y_r]$$

  - e_ψ , e_y are the heading and lateral error
  - $v_x,\ v_y,\ \dot{\psi},\ \dot{\theta}$ are the longitudinal, lateral, yaw and roll velocities
  - $\delta,\ \gamma_t,\ \gamma_b,\ y_r$ are the actual control inputs

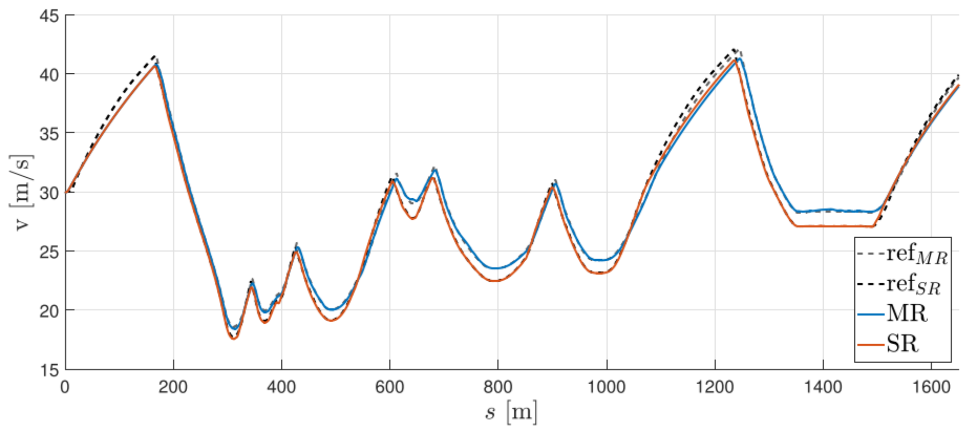




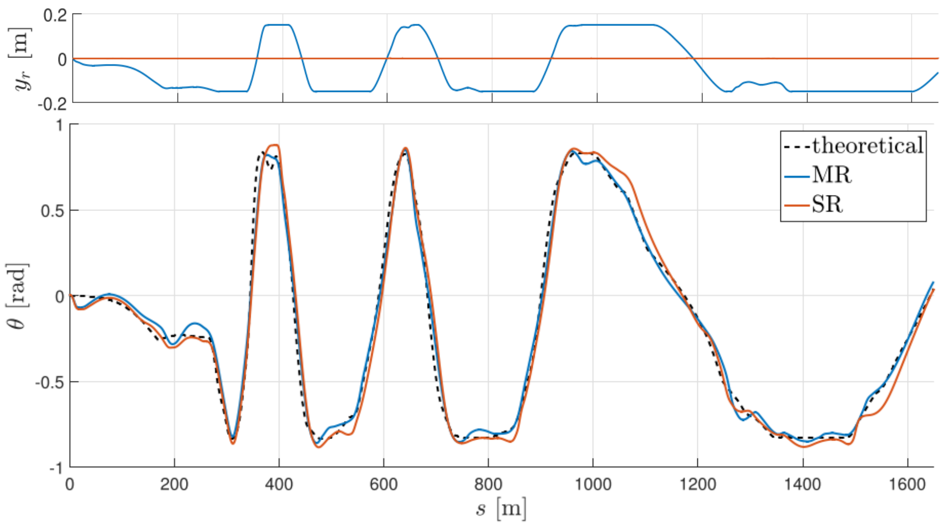
NMPC Virtual Rider
Results

- Comparison between
  - Previously achieved maximum performance with static rider
  - Obtained maximum performance with moving rider

Velocity profiles for moving rider (MR) and static rider (SR).



Roll and rider movement profiles for moving rider (MR) and static rider (SR).



- With moving rider, a higher velocity has been maintained during cornering with similar or lower roll angles

Data-driven Tuning of a NMPC Controller for a Virtual Motorcycle through Genetic Algorithm

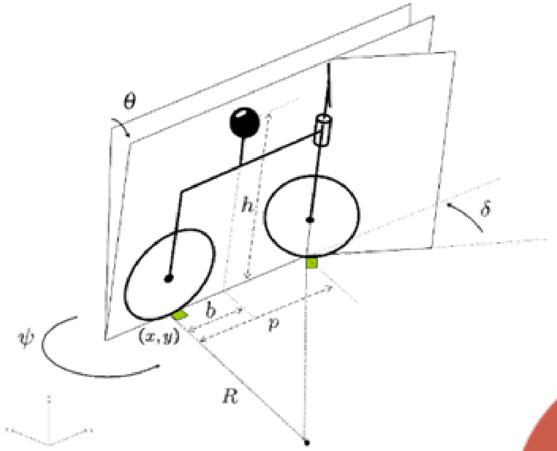
E. Picotti[1], L. Facin[1], A. Beghi[1], M. Nishimura[2], Y. Tezuka[2], F. Ambrogi[3], M. Bruschetta[1]

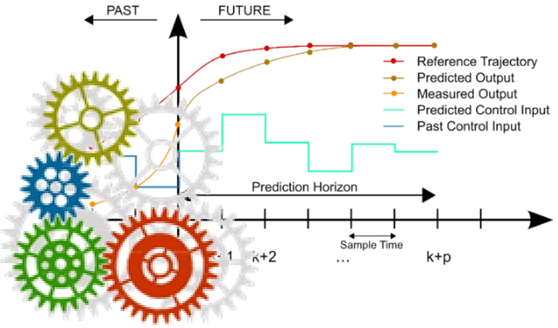
[1]Università di Padova
[2]Honda Racing Corporation
[3]VI-Grade GmbH

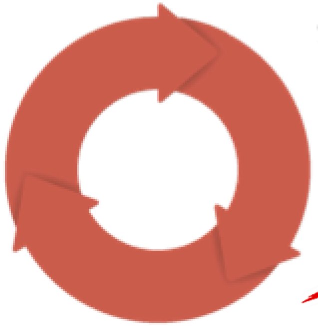
IEEE Conference on Control Technology and Applications, August 2022

Data-driven Tuning Strategy

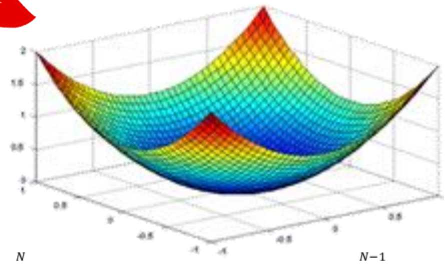
LbNMPC structure







data-driven
cost function tuning

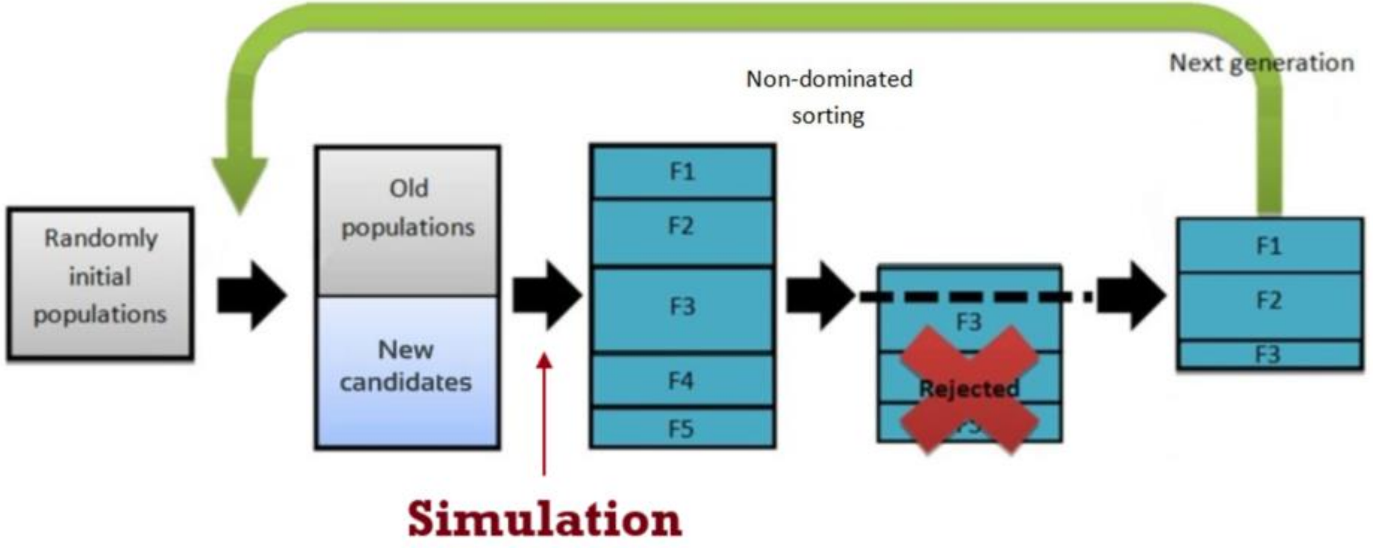


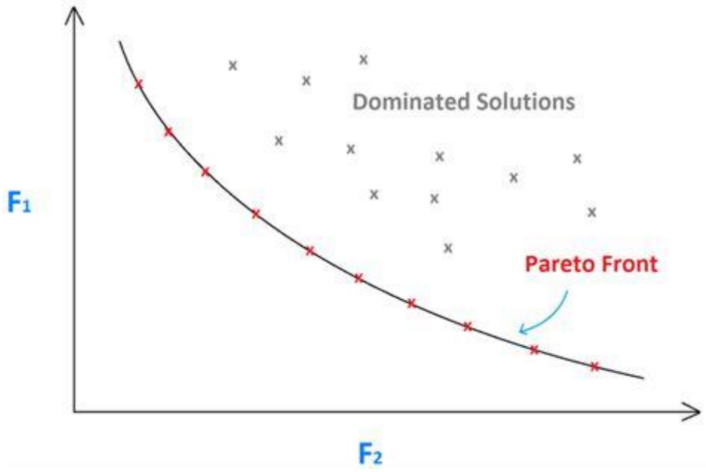
$$\sum_{k=0}^{N}\left(l_k(x_k,u_k)-l_{ref}\right)^T \boldsymbol{Q}\left(l_k(x_k,u_k)-l_{ref}\right)+\sum_{k=0}^{N-1} u_k^T \boldsymbol{R} u_k$$

Data-driven Tuning Strategy

Genetic algorithm

- Non-dominated Sorting Genetic Algorithm (NSGA-II)
  - Constrained multi-objective optimization: minimization of cost functions





Data-driven Tuning Strategy

Optimization functions

1. Lap-time performance

$$\mathbf{J}_1 = \begin{cases} t_{end} & \text{if simulation does conclude the lap} \\ t_{max} - t_{end} & \text{if simulation does } \textit{not} \text{ conclude the lap} \end{cases}$$

- t_{end} is the simulation time
- t_{max} is user-defined value higher than any possible concluding lap-time

- It measures the lap-time performance and, if no simulations are concluding, the ones that run closer to the end are preferred

2. Sensitivity w.r.t. tuning parameters variation

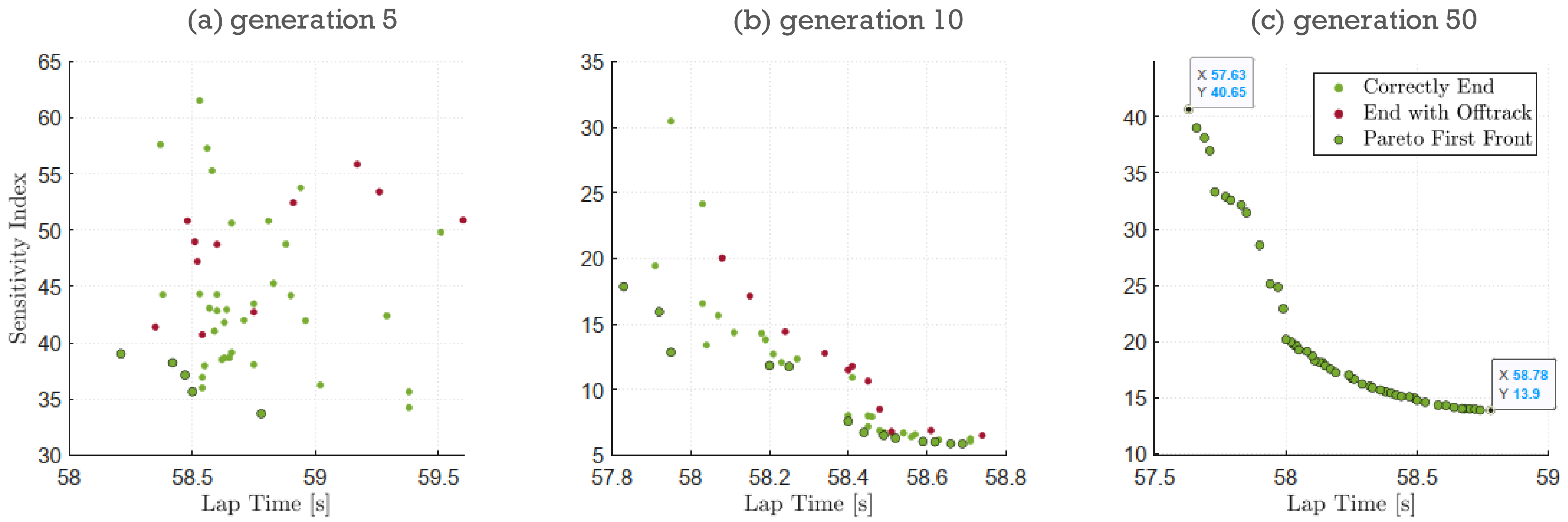
$$S_j = \frac{\sum_i q_i \cdot C_{j,i}}{\sum_i q_i}$$

- q_i weights the distance between *i*-th and *j*-th simulations
- $C_{j,i}$ compares their performance

- It measures the variations of the performance in the neighborhood of the current tuning parameters

Data-driven Tuning Strategy
Results (1)

Data-driven tuning results after 5, 10, and 50 generations.

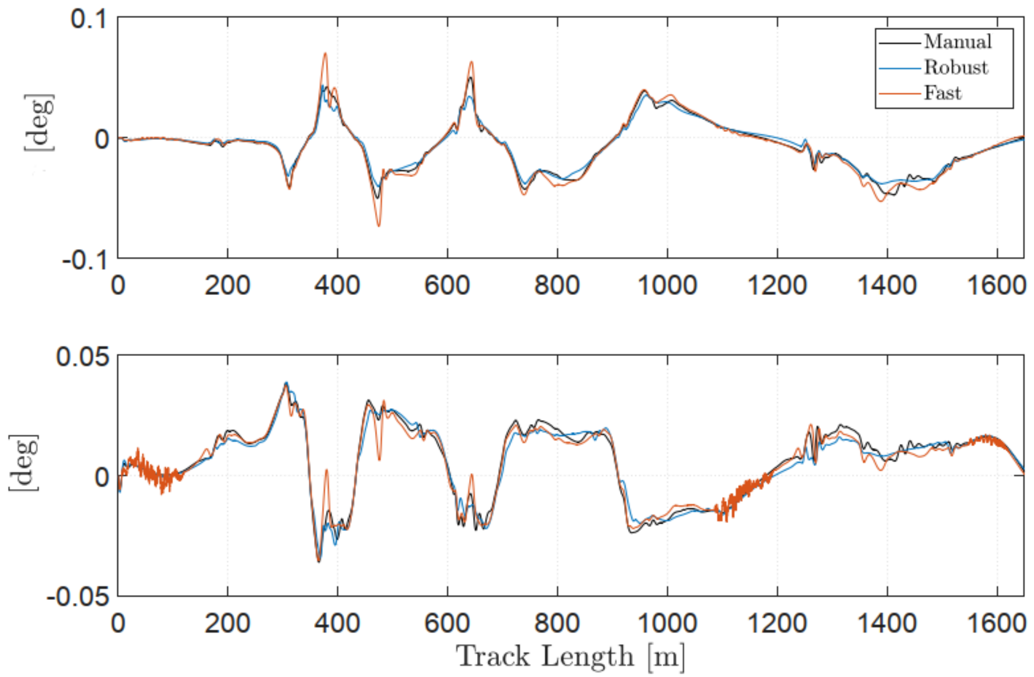


- At first generations, low number of configurations completing the track and high number of off-track simulations
- After 50 generations, only correctly ending simulations are present and Pareto is reached

Data-driven Tuning Strategy
Results (2)

- Comparison between
  - Previously published manually tuned configuration
  - Most robust configuration at 50^{th} generation
  - Fastest configuration at 50^{th} generation
- Manually tuned and robust configurations have very similar behavior
- Fast configuration has very high frequency *unstable* behavior and concludes the lap in 0.91s less

Sideslip (top) and steering (bottom) behavior for manual tuning, robust and fast configurations.



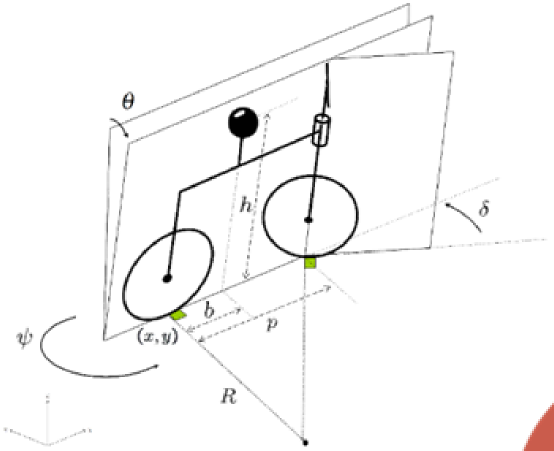
+

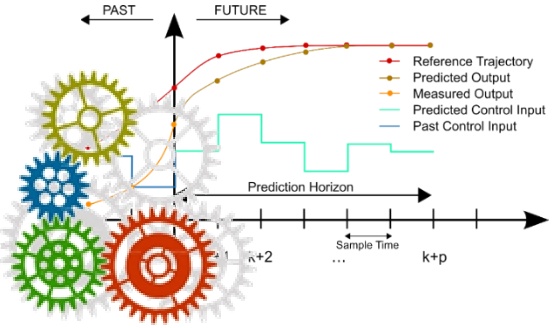
Learning-based Nonlinear Model Predictive Control of a Virtual Motorcycle in Real-Time

E. Picotti[1], F. Bianchin[2], M. Bruschetta[1]
[1]Università di Padova
[2]Technical University of Munich

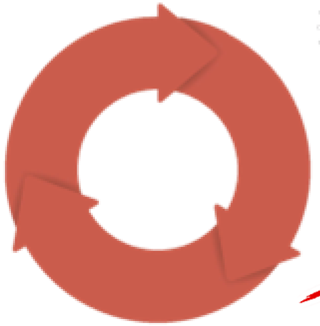
Elsevier Control Engineering Practice, January 2024

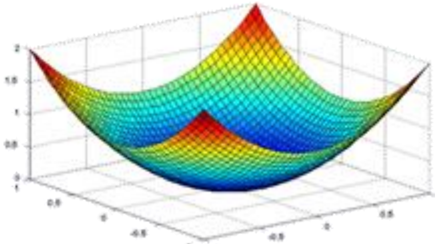
Learning Dynamics NMPC
LbNMPC structure



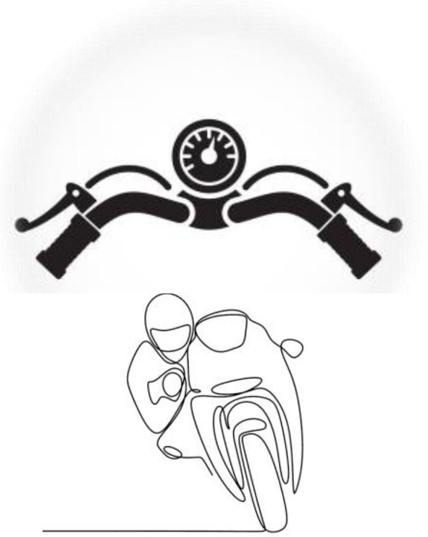


data-driven
prediction model





$$\sum_{k=0}^{N}\left(l_k(x_k,u_k)-l_{ref}\right)^T \boldsymbol{Q}\left(l_k(x_k,u_k)-l_{ref}\right)+\sum_{k=0}^{N-1} u_k^T \boldsymbol{R} u_k$$



Learning Dynamics NMPC
Model dynamics

- General ODE model
$$\dot{x}(x,u) = \underbrace{\begin{bmatrix} \dot{q}(x) \\ \ddot{q}(x,u) \end{bmatrix}}_{\text{nominal dynamics}} + \underbrace{\begin{bmatrix} 0 \\ \psi_{\ddot{q}}(x,u) \end{bmatrix}}_{\text{data-driven augmentation}}$$

→ Grey-box dynamics

- System input u
- System state $x = [q, \dot{q}]^T$

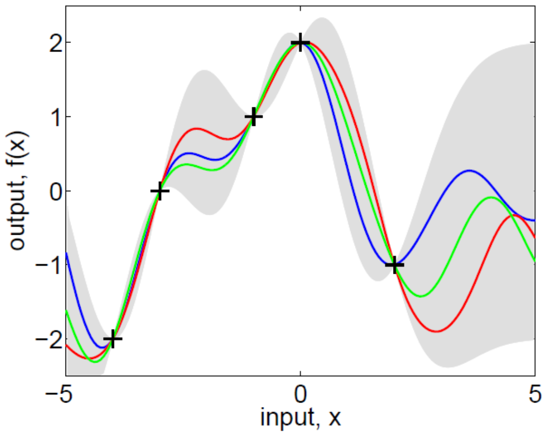
positions

velocities

- Gaussian Process as augmented model

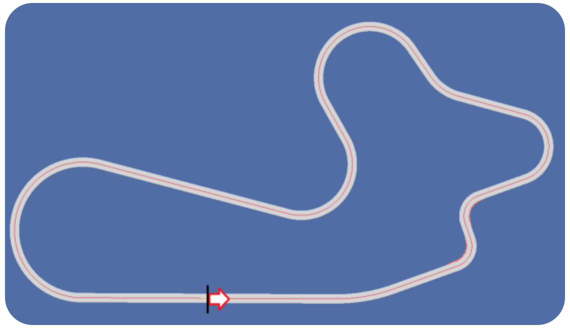
$$f(\mathbf{x}) \sim \mathcal{GP}\big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\big)$$

  - function approximator
  - estimates the function values given the past observations



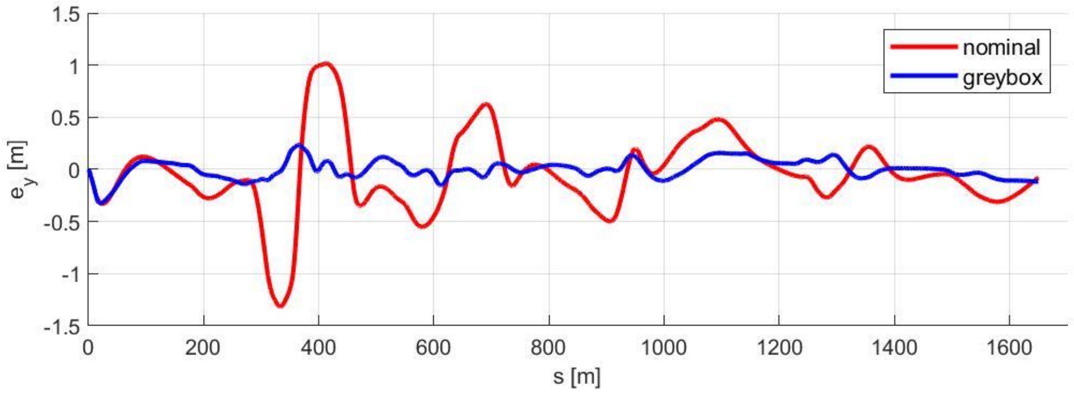
Learning Dynamics NMPC
Results (1)

- Performance comparison
  - Nominal physics-based
  - Grey-box model

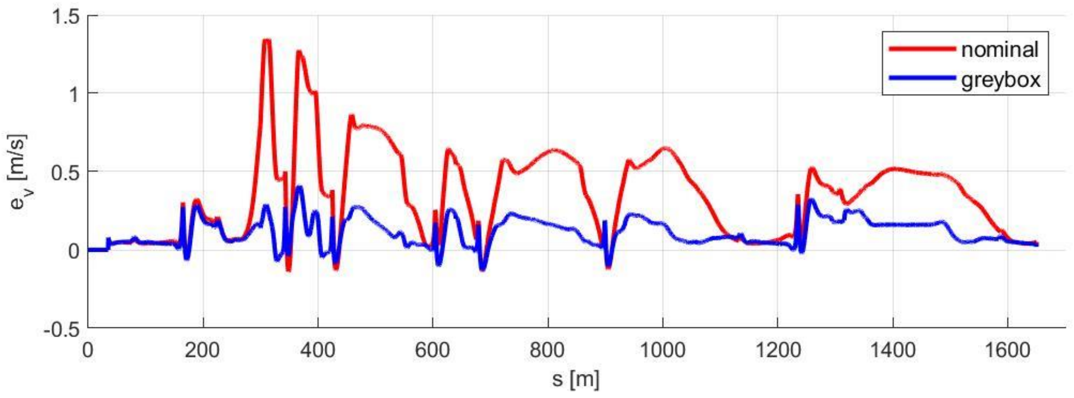


Learning Dynamics NMPC
Results (2)

- High reduction in all tracking errors, i.e.,
  - Lateral more then 75%
  - Angular more then 60%
  - Velocity more then 60%
- Computational time increase around 75%, still real-time at 100Hz



	Nominal	Grey-box
e_y $[m]$	0.40	0.009 (-77.7%)
e_φ $[deg]$	1.45	0.53 (-63.0%)
e_v $[m/s]$	0.41	0.16 (-60.5%)
T_{solver} $[ms]$	3.74	6.63 (+77.2%)

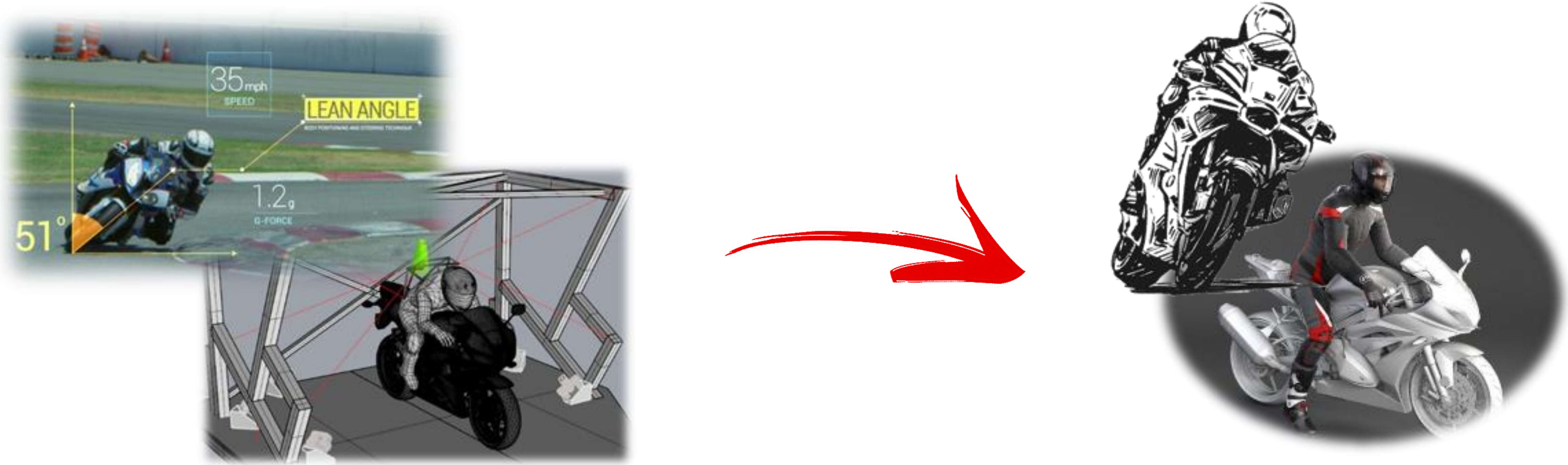


Conclusion
Discussion

- A human-like motorcycle rider is fundamental for achieving reliable simulative results
- Learning-based strategies can improve the reliability of the virtual rider implementation
- Realistic virtual rider movement still hard to obtain by algorithmic strategies

Conclusion
Future Possibilities

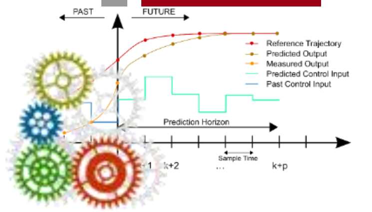
- Acquired data – possibly on a simulator - could be used for virtual rider definition and enhancement
- The virtual rider could be adapted on specific riding style



Q&A

THANK YOU FOR
YOUR ATTENTION

Theoretical Framework
NMPC procedure



Optimal Control Problem

$$\begin{aligned} \min_{u(\cdot)} \mathcal{F} &= \int \mathcal{L}\big(x(t),u(t)\big)dt \\ s.t.\ \ x(t_0) &= \hat{x}_0 \\ \boxed{\dot{x}(t) = \mathcal{G}\big(x(t),u(t)\big)} \\ \mathcal{H}\big(x(t),u(t)\big) &\leq 0 \end{aligned}$$

prediction model can be data-driven

Non-Linear Programming Problem

$$\begin{aligned} &\min_{w} f(w) \\ s.t.\ \ &x(0) = \hat{x}_0 \\ &g(w) = 0 \\ &h(w) \leq 0 \end{aligned}$$

Quadratic Programming Problem

$$\begin{aligned} &\min_{\Delta w} \frac{1}{2} \Delta w^T B \Delta w + F^T \Delta w \\ s.t.\ \ &G^T \Delta w + g(\overline{w}) = 0 \\ &H^T \Delta w + h(\overline{w}) \leq 0 \end{aligned}$$

(sub-)optimal solution

Discretization

Direct Multiple Shooting

$$w = [w_1, \dots, w_N] = [x_1, u_1, \dots, x_N, u_N]$$

$$\begin{aligned} \int \mathcal{L}\big(x(t),u(t)\big)dt \ &\Rightarrow f(w) = \sum_{k=1}^{N} l(w_k) \\ \dot{x}(t) = \mathcal{G}\big(x(t),u(t)\big) &\Rightarrow g(w) \triangleq x_{k+1} - \phi(w_k) = 0 \\ \mathcal{H}\big(x(t),u(t)\big) \ &\Rightarrow h(w) \end{aligned}$$

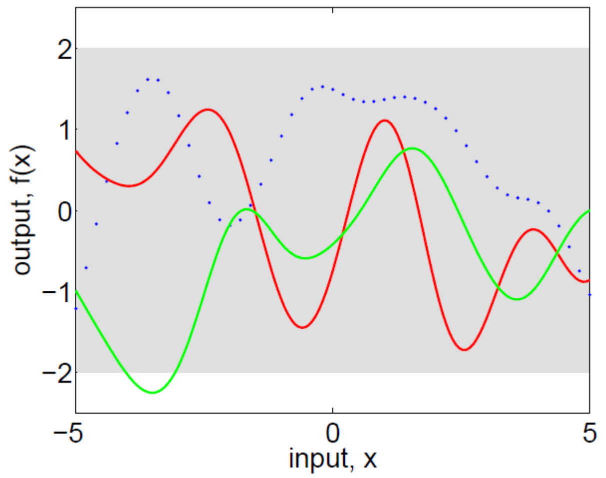
Linearization

Sensitivities Computation

$$B \cong \nabla^2 \mathcal{L},\ F = \nabla f,$$
$$G = \nabla g,\ H = \nabla h,$$
$$\mathcal{L} = f(w) + \lambda^T g(w) + \mu^T h(w)$$

Theoretical Framework
Gaussian Processes

- A Gaussian Process is defined as *a collection of random variables, any finite number of which have a joint Gaussian distribution* [16]
- It can be interpreted as a **distribution over functions** with a continuous domain, e.g. time or space
  - For real functions $f \in F, f\colon \mathcal{X} \to \mathbb{R}$
    - $f(x), \boldsymbol{x} \in \mathcal{X}$ is a random variable with gaussian distribution
    - evaluating a sample $f(x)$, for n different points it is possible to generate a random vector $\mathbf{f} = [\mathbf{f}(\boldsymbol{x}_1) \ldots \mathbf{f}(\boldsymbol{x}_n)]$
  - All the sample functions have similar characteristics, such as expected rate of change and magnitude



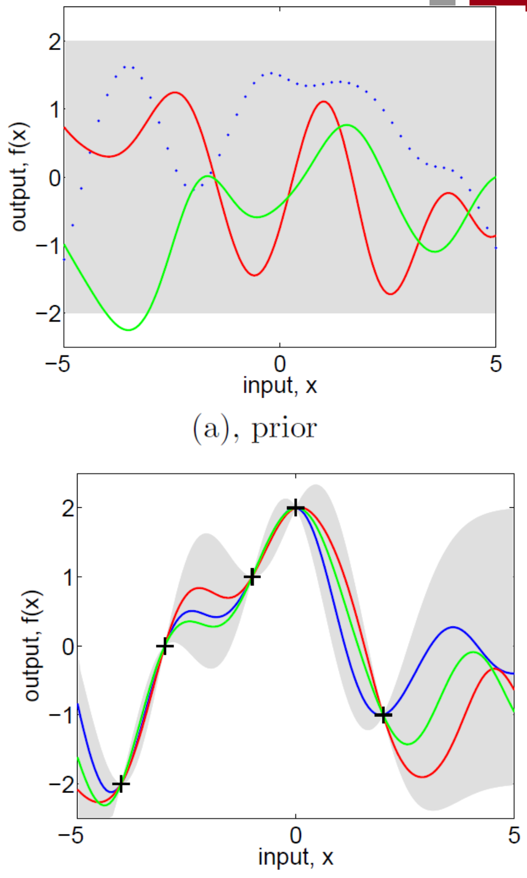
- A Gaussian Process is completely defined by its mean $m(\boldsymbol{x})$ and covariance $k(\boldsymbol{x}, \boldsymbol{x}')$ functions

$$f(\mathbf{x}) \sim \mathcal{GP}\big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\big)$$

[16] C. E. Rasmussen, "Gaussian processes in machine learning," in Summer school on machine learning. Springer, 2003, pp. 63–71.

Theoretical Framework
Why Gaussian Process Regression?

- Relies on Bayesian Regression to find the posterior distribution over the parameters, given the **observations of the real system**
- It is computationally efficient in generating the function prediction at a sample point $\boldsymbol{x}^*$, given the observations $\mathbf{y}$ for a dictionary X, i.e.,
$$f_{\boldsymbol{x}^*} = k(\boldsymbol{x}^*, \mathrm{X})^T [k(\mathrm{X}, \mathrm{X}) + \sigma^2 I]^{-1} \mathbf{y}$$
- Possibility to **reduce the dimension** of the dictionary both **offline**, e.g., by optimization techniques, and **online**, e.g., by choosing the nearest samples in the dictionary
- Possibility to **embed specific behaviours** by the covariance function definition, e.g., by choosing the Squared Exponential kernel, it is guaranteed the model continuity, and allow controlling the smoothness



(a), prior

(b), posterior

Learning Dynamics NMPC
Results (2)

- High reduction in all tracking errors, i.e.,
  - Lateral more then 75%
  - Angular more then 60%
  - Velocity more then 60%
- Computational time increase around 75%, still real-time at 100Hz