Nonlinear $\mathcal{H}_\infty$ Control based on Gaussian Process for a quadrotor ⋆

Guilherme Rossi de Avelar Oliveira * **Roberto Santos Inoue** **

* *Department of Electrical Engineering, Federal University of São Carlos, São Carlos, SP (e-mails: guilhermeavelar@estudante.ufscar.br)*
** *Department of Computer Sciences, Federal University of São Carlos, São Carlos, SP (e-mail: rsinoue@ufscar.br)*

Abstract:
The use of Unmanned Aerial Vehicles (UAVs) can be subject to external disturbances, such as wind gusts, aerodynamic forces, and parametric uncertainties, depending on the flying environment. Therefore, the automatic control must be robust enough to execute a predefined task. In this work, we propose a robust control strategy based on the dynamic model to mitigate external disturbances and parametric uncertainties. The $\mathcal{H}_\infty$ criterion is employed to provide robustness and Gaussian processes are used to supplement the mathematical model of the quadrotor, estimating uncertainties not accounted by the nominal model. A comparative study among the proposed controller, the nonlinear $\mathcal{H}_\infty$ controller, and the neural network-based nonlinear $\mathcal{H}_\infty$ controller is presented, in which we analyze trajectory tracking under different levels of parametric uncertainty and external disturbances, and compare the performance and energy consumption of the controllers.

Resumo: A utilização de Veículos Aéreos Não Tripulados (VANTs), pode estar sujeita a perturbações externas, como rajadas de vento, forças aerodinâmicas e incertezas paramétricas, dependendo do ambiente em que voa. Portanto, o controle automático deve ser robusto o suficiente para executar uma tarefa pré-estabelecida. Neste trabalho, propomos uma estratégia de controle robusta baseada no modelo dinâmico para mitigar distúrbios externos e incertezas paramétricas. O critério $\mathcal{H}_\infty$ é utilizado para dar o caráter robusto e processos Gaussianos para complementar o modelo matemático do quadrotor, estimando as incertezas não modeladas do modelo nominal. Um estudo comparativo entre o controlador proposto, o controlador $\mathcal{H}_\infty$ não-linear, e o controlador $\mathcal{H}_\infty$ não-linear baseado em redes neurais é apresentado, no qual analisamos o rastreamento de trajetória em diferentes níveis de incerteza paramétrica e distúrbios externos, e comparamos o desempenho e consumo de energia dos controladores.

Keywords: Quadrotor; Gaussian Process; Adaptive control; H Infinity.
Palavras-chaves: Quadrotor; Processo Gaussiano; Controle Adaptativo; H Infinito.

1. INTRODUCTION

Aircraft face external disturbances and parameter uncertainties, making effective control challenging. The $\mathcal{H}_\infty$ technique attenuates disturbances, while adaptive methods handle uncertainties. For example, in Raffo et al. (2015), a robust strategy combines *backstepping* with $\mathcal{H}_\infty$ control for quadrotor stabilization amid disturbances and uncertainties. Ortiz et al. (2016) compares robust $\mathcal{H}_\infty$ with PID control for guidance. Similarly, Liu et al. (2018) integrates robust and adaptive *backstepping* to address disturbances and inertia uncertainty. Demircioglu and Basturk (2017) adapts height control for quadrotors, considering unknown parameters and wind disturbances.

In Pazelli et al. (2011) and Nogueira et al. (2013), adaptive $\mathcal{H}_\infty$ controls employ neural networks and fuzzy systems for manipulators, proving effective amid uncertainties. Researchers use Gaussian processes, like in Berkenkamp and Schoellig (2014), for learning nonlinear models and enhancing stability. Cao et al. (2016) applies Gaussian process-based hierarchical control, while Wang et al. (2017) captures quadrotor dynamics using recursive Gaussian processes.

Our approach introduces nonlinear $\mathcal{H}_\infty$ control based on Gaussian Processes. One approach adapts the process via control, the other uses pre-trained data, bolstering position control against disturbances and uncertainties. We implement adaptive nonlinear $\mathcal{H}_\infty$ control for a drone's trajectory tracking, guided by neural networks from Pazelli et al. (2011) and Nogueira et al. (2013). We compare outcomes via simulations, evaluating performance, energy use, and trajectory behavior in the presence of uncertainties.

⋆ The following Brazilian research agencies have supported this work: São Paulo Research Foundation (FAPESP) (#2014/50851-0), Brazilian National Council for Scientific and Technological Development (CNPq) (#465755/2014-3, #421131/2018-7), and by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Our paper is organized in the folowing way: Section 2 covers the problem formulation and quadrotor dynamics for nonlinear $\mathcal{H}_\infty$ control. Section 3 details Gaussian process-based control. Section 4 discusses simulation results, and Section 5 concludes.

2. PROBLEM FORMULATION

In this work, in Section 3, we propose a nonlinear robust control based on the $\mathcal{H}_\infty$ norm to attenuate external disturbances and solve trajectory tracking, with an adaptive Gaussian process algorithm to estimate parametric uncertainties. So, before we present the control strategy, in the following sections, we are going to present the quadrotor model and its state-space representation for the nonlinear $\mathcal{H}_\infty$ control problem.

2.1 Quadrotor

The Parrot Bebop 2.0 quadrotor, illustrated in Figure 1, is an autonomous aerial vehicle developed and marketed by Parrot. It was designed to be controlled by a joystick, smartphones, and tablets via specific protocols of WiFi communication. Bebop 2.0 is equipped with two embedded systems. One IMU contains a 3-axis magnetometer, a 3-axis gyroscope axes, a 3-axis accelerometer, an ultrasonic sensor for flight analysis above 8 meters of altitude, and a barometric sensor. It also has image processing that uses optical flow algorithms to detect movement patterns. This unit has two optical sensors, one in front and another located at the bottom of the aircraft, which are used to detect reference patterns in the soil to obtain stability of attitude in hovering. The main system of the quadrotor is capable of automatically performing take-off, and landing procedures, and stabilization of the aircraft's attitude, in addition to responding to movement commands.

Figure 1. Parrot Bebop 2 quadrotor.

2.2 Dynamic Modelling

To implement the tracking controllers, first, we need to define its dynamic model. The quadrotor has 6 degrees of freedom, 3 reference axis (x,y,z), and 3 Euler angles (θ, ϕ, ψ),

Rigid body mechanics for quadrotors are often formulated based on Euler-Lagrange equations, as seen in Johansson (1990) and Raffo et al. (2015).

The Parrot Bebop 2 quadrotor already has automatic control of the pitch and roll angles (ϕ, θ), considering that, we won't control these parameters. We can also eliminate the gravitational force term of our model, considering the built-in hover system present on the Bebop 2.

In that way, we can define the system vector transposed q^T below:

$$q^T = [x \ y \ z \ \psi]\,. \tag{1}$$

And following the steps of Johansson (1990), Raffo et al. (2015) and Castillo et al. (2005), the systems becomes:

$$\begin{bmatrix} mI & 0 \\ 0 & \mathcal{I}_{zz} \end{bmatrix} \begin{bmatrix} \ddot{\xi} \\ \ddot{\eta} \end{bmatrix} = \nu + \delta(\bar{q}), \tag{2}$$

where m is the quadrotor mass, $\xi = [x, y, z]^T$ is the translational vector, $\eta = [\phi, \theta, \psi]^T$ is the rotational vector, $\mathcal{I}_{zz}$ is the inertia around z-axis, ν the control input, and $\delta(q)$ the external forces.

Considering $\delta(q) = w$ and adding parametric uncertainties, the final system rearranged becomes:

$$\ddot{q} = B\nu + B_d\omega, \tag{3}$$

where B_d is the matrix that maps the external disturbances, ω represents the disturb vector and $B = M^{-1}$.

2.3 Parrot Bebop 2 input vector

Our control input vector is ν contains the commands to be sent to the quadrotor. The Parrot Bebop 2 specifically only receives values inside $[-1, 1]$ as commands. For example, we can send 1 to the $x+$ direction of the Bebop to accelerate the quadrotor to its maximum capacity in that direction. Having that in mind, we limit the control vector ν with the hyperbolic tangent function as below:

$$\nu_{sat} = tanh(\nu_{raw}), \tag{4}$$

where ν_{raw} is the raw control inputs and ν_{sat} is the vector adjusted between $-1, 1$.

Having adapted the input vector for the Bebop, we now need to translate these adjusted values to the dynamic model. In Pinto et al. (2020), and Benevides et al. (2019) calibration flights were made to determine the relationship between the Bebop inputs and the real quadrotor acceleration. In that regard, we can create the relation below

$$\nu = K_a\nu_{sat}, \tag{5}$$

where ν is the control input used on the control modelling and K_a is taken from Benevides et al. (2019) the Bebop-acceleration scaling matrix as follows

$$K_a = \begin{bmatrix} 4.33 & 0 & 0 & 0 \\ 0 & 4.12 & 0 & 0 \\ 0 & 0 & 4.42 & 0 \\ 0 & 0 & 0 & 5.92 \end{bmatrix}. \tag{6}$$

2.4 State-Space Representation for Nonlinear $\mathcal{H}_\infty$ Control

We adopt $\ddot{q}^d$ and $\dot{q}^d$, respectively, as the desired accelerations and velocities. We can add and subtract $\ddot{q}^d$ of one of the sides of the quadrotor model Equation 3, giving us:

$$\ddot{q} = \ddot{q}^d - \ddot{q}^d + B\nu + B_d\omega. \tag{7}$$

Defining $\tilde{q} = q - q^d$, $\dot{\tilde{q}} = \dot{q} - \dot{q}^d$ and $\ddot{\tilde{q}} = \ddot{q} - \ddot{q}^d$ as trajectory tracking error and its derivatives, the last equation becomes:

$$\ddot{\tilde{q}} = -\ddot{q}^d + B\nu + B_d\omega. \tag{8}$$

We set the vector $\tilde{x}$ to represent the space state based on the error:

$$\tilde{x} = \begin{bmatrix} \dot{\tilde{q}} \\ \tilde{q} \end{bmatrix} = \begin{bmatrix} \dot{q} - \dot{q}^d \\ q - q^d \end{bmatrix}. \quad (9)$$

The state-space representation, in space states, for the trajectory tracking control of the quadrotor is given by:

$$\dot{\tilde{x}} = \begin{bmatrix} 0 & 0 \\ I & 0 \end{bmatrix} \tilde{x} + \begin{bmatrix} I \\ 0 \end{bmatrix} u + \begin{bmatrix} B_d \\ 0 \end{bmatrix} \omega, \quad (10)$$

where

$$u = -\ddot{q}^d + B\nu, \quad (11)$$

or

$$\nu = B^{-1}(u + \ddot{q}^d). \quad (12)$$

Now we consider the following space state transformation:

$$\tilde{z} = \begin{bmatrix} \tilde{z_1} \\ \tilde{z_2} \end{bmatrix} = T_0 \tilde{x} = \begin{bmatrix} T_{11} & T_{12} \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{\tilde{q}} \\ \tilde{q} \end{bmatrix}, \quad (13)$$

where T_{11}, $T_{12} \in \Re^{2\times 2}$ constant matrixes to be determined. We assume T_{11} diagonal and to simplify, we choose $T_{11} = t_{11}I$. Applying this transformation in Equation (10) we obtain:

$$\dot{\tilde{x}} = A_T \tilde{x} + M_T(-F - \Delta F) + \nu) + B_T \omega, \quad (14)$$

where

$$A_T = T_0^{-1} \begin{bmatrix} 0 & 0 \\ T_{11}^{-1} & -T_{11}^{-1} T_{12} \end{bmatrix} T_0, \quad (15)$$

$$B_T = T_0^{-1} \begin{bmatrix} B_d \\ 0 \end{bmatrix} T_{11}, \; M_T = T_0 \begin{bmatrix} M \\ 0 \end{bmatrix}, \quad (16)$$

$$F = M(\ddot{q}^d - T_{11}^{-1} T_{12} \dot{\tilde{q}}), \quad (17)$$

$$\Delta F = \Delta M(\ddot{q}^d - T_{11}^{-1} T_{12} \dot{\tilde{q}}) + \delta, \quad (18)$$

ΔM is the uncertainty part of M, and δ are the unmodeled uncertainties.

3. NONLINEAR $\mathcal{H}_\infty$ CONTROL BASED ON GAUSSIAN PROCESS

This section proposes the nonlinear $\mathcal{H}_\infty$ control based on the Gaussian process. First, we show the adaptive Nonlinear $\mathcal{H}_\infty$ Control Section 3.1. and its formulation to get the three equations needed to solve the system parameters; after that, in Section 3.1.1 we describe the Gaussian Process as an online adaptive method; we then model the control law of the Nonlinear $\mathcal{H}_\infty$ Control using Gaussian Process Offline on Section 3.2 and explain the Gaussian Process modeling for the offline control and its calculations on Section 3.2.1.

3.1 Adaptive Nonlinear $\mathcal{H}_\infty$ Control

As seen in Pazelli et al. (2011) e Nogueira et al. (2013), consider the robotic system described in (14) with the disturbance $d \in \mathcal{L}_2[0, \infty)$. Given an attenuation factor $\gamma > 0$ and a weighting matrix $Q = Q^T > 0$, if there is a positive symmetric matrix $K = K^T > 0$ and a matrix T_0 satisfying the Riccatti algebraic function

$$K - T_0^T [I \;\; 0]^T \left(R^{-1} - \frac{1}{\gamma^2} T_{11}^2 \right) [I \;\; 0] T_0 + Q = 0 \quad (19)$$

with R the gain matrix in which $R < \gamma^2 I$, then the adaptative control law $\mathcal{H}_\infty$ based on intelligent structures

$$\nu = F + \Xi\Theta + T_{11}^{-1} u + u_s \quad (20)$$

with

$$\dot{\Theta} = Proj[-Z^{-1} \Xi^T T_{11} [I \;\; 0] T_0 \tilde{x}], \quad (21)$$

$$u = -R^{-1} [I \;\; 0] T_0 \tilde{x}, \quad (22)$$

$$u_s = -k(x_e) sgn(T_{11} [I \;\; 0] T_0 \tilde{x}), \quad (23)$$

where Z the adaptive gain

$$sgn(T_{11}[I \;\; 0]T_0\tilde{x}) \triangleq [sgn((T_{11}[I \;\; 0]T_0\tilde{x})_1) \dots \; sgn((T_{11}[I \;\; 0]T_0\tilde{x})_n)]^T \quad (24)$$

guarantees that all the variables in the closed-loop system (14, 20-23) are limited and the following inequality is satisfied

$$\int_0^T \left(\tilde{x}^T Q \tilde{x} + u^T R u \right) dt \leq \quad (25)$$

$$\tilde{x}^T(0) P_0 \tilde{x}(0) + \tilde{\Theta}^T(0) Z \tilde{\Theta}(0) + \gamma^2 \int_0^T (d^T d) dt.$$

Considering the matrix, Q factored in the following way

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^T & Q_{22} \end{bmatrix}. \quad (26)$$

we can break the Equation (19) in three other equations

$$q_{11} - t_{11}^2 r_u^{-1} + \gamma^{-2} t_{11}^4 = 0, \quad (27)$$

$$q_{22} - t_{12}^2 r_u^{-1} + \gamma^{-2} t_{11}^2 t_{12}^2 = 0, \quad (28)$$

$$k + q_{12} - t_{11} t_{12} r_u^{-1} + \gamma^{-2} t_{11}^3 t_{12} = 0. \quad (29)$$

In this way, t_{11}, t_{12} and $k > 0$ can be found for the smallest attenuation factor $\gamma > 0$.

Adaptive Gaussian Process Online Considering the $\mathcal{H}_\infty$ nonlinear control based on the model described in Section 3.1, we have that Equation (20) contemplates the term $\Xi\Theta$ that represents the uncertainties ΔF. In this section, we try to estimate this term.

Having established that, we can define the whole Gaussian process as follows:

$$\Delta \hat{F}_0(q_e) = \begin{bmatrix} \xi_1 & 0 & \dots & 0 \\ 0 & \xi_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \xi_n \end{bmatrix} \begin{bmatrix} \Theta_1 \\ \vdots \\ \Theta_n \end{bmatrix} = \Xi\Theta, \quad (30)$$

where ξ_k is the kernel given by

$$\xi_k = \sigma^2 exp\left(- \frac{||q_{e_i}^k - {q_{e_i}^d}^k||^2}{2l^2} \right) \quad (31)$$

σ^2 is the scaling factor that calculates the function variation based on its average, l is the length scale, $q_{e_i}^d$ is the sum of the desired positions, velocities, and accelerations, calculated at each iteration i and q_{e_i} is the sum of the actual positions, velocities, and accelerations at each iteration i. And the adaptive factor Θ is obtained by (21) during the trajectory tracking, so we consider the update Online.

3.2 Nonlinear $\mathcal{H}_\infty$ Control based on Gaussian Process Offline

We do not have the adaptive control law in the Nonlinear $\mathcal{H}_\infty$ Control based on Gaussian Process Offline. So the control law (20), becomes

$$\nu = F + \Delta\widehat{F} + T_{11}^{-1}u. \tag{32}$$

Gaussian Process Offline We call this process Offline because we estimate the uncertainties before the execution of the controller. In that regard, we use the variables q_e, q_e^d and F_0, previously saved from another implemented controller.

We define a Gaussian process given by

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')), \tag{33}$$

where $m(x)$ is the average value and the covariance is given by *kernel*

$$k(x, x') = \sigma^2 exp\left(- \frac{||x - x'||^2}{2l^2} \right), \tag{34}$$

where σ^2 is a scaling factor that determines the function variance based on its average, l is the length factor, x is the entry data, and x' is the test data.

We can then determine $\Delta\widehat{F}$ being

$$\Delta\widehat{F} = k(q_e, q_e^d)[k(q_e, q_e) + \sigma_n^2 I]^{-1} F, \tag{35}$$

where σ_n^2 is the parameter that estimates the expected disturbance on the data set, q_e^d is the sum of the desired positions, velocities, and accelerations calculated and saved beforehand, and q_e is the sum of the real positions, velocities, and accelerations calculated and saved beforehand.

We use the algorithm defined in Rasmussen and Williams (2006) to calculate the uncertainty of the dynamic model:

$$\begin{aligned} L &= cholesky(K + \sigma_n^2 I); \\ \alpha &= L^T(LF); \\ \Delta\widehat{F} &= k(q_e, q_e^d)^T \alpha, \end{aligned}$$

using this algorithm repeatedly for each entry data q_e.

4. RESULTS

The following results were achieved by simulating the dynamic model and controller equations in a Matlab environment, introducing the system's parametric uncertainties and wind disturbance as desired. The desired trajectory used in the experiments is an ellipse. The chosen speed is 0.3 m/s and 2000 iterations in each flight, which lasts 20 seconds.

The first experiment was the trajectory comparison among all controllers implemented. The conditions chosen were 50% parametric uncertainties on top of the quadrotors mass and inertia and wind disturbance of $5m/s^2$.

4.1 Controllers Parameters

In this work, we are going to compare the Nonlinear $\mathcal{H}_\infty$ Control based on Gaussian Process with the Feedback Linearization (Section 4.1.1), the Nonlinear $\mathcal{H}_\infty$ Control, and the adaptive nonlinear $\mathcal{H}_\infty$ control based on neural network, Pazelli et al. (2011) e Nogueira et al. (2013).

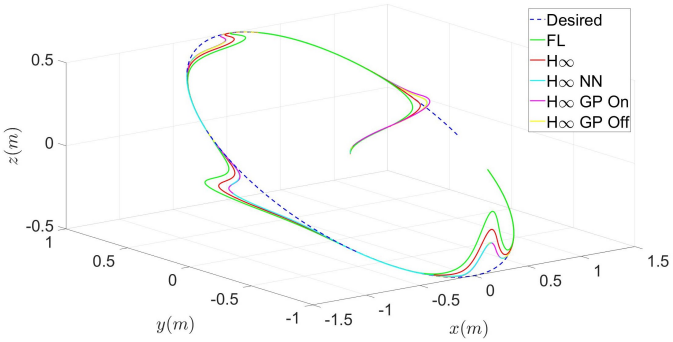

Figure 2. Trajectory comparison of all controllers in a flight with wind disturbances.

So in the section below, we show the tunning parameters used for each of the controls in the comparative study.

Feedback Linearization We use a Proportional-Derivative feedback linearization controller as the most basic controller in the experiments. Its control law can be defined as:

$$\nu = M(u + \ddot{q}^d), \tag{36}$$

where M is the system mass, $\ddot{q}^d$ is the desired trajectory acceleration, and with:

$$u = -K_p\tilde{q} + -K_d\dot{\tilde{q}}, \tag{37}$$

where K_p is the proportional gain and K_d is the derivative gain. The gains used in the experiments for this controller are

$$K_p \ = \ 50 \times I_{4\times4}, \ K_d \ = \ 15 \times I_{4\times4}$$

Adaptive Nonlinear $\mathcal{H}_\infty$ control based on Neural Network Consider n neural networks $\xi_k^T\Theta_k$, $k = 1, \ldots, n$, containing nonlinear neurons in all the hidden layers and linear neurons in the first and last layers, with adjustable parameters Θ_k on the final layer (Chang (2000), Chen et al. (1997)). The neural network equations are described as

$$\xi_k^T\Theta_k = \sum_{i=1}^{p_k} H\left(\sum_{j=1}^{5n} w_{ij}^k q_{ej} + m_i^k \right)\Theta_{ki} \tag{38}$$

and

$$\xi_k = \begin{bmatrix} H\left(\sum_{j=1}^{5n} w_{1j}^k q_{ej} + b_1^k\right) \\ \vdots \\ H\left(\sum_{j=1}^{5n} w_{p_kj}^k q_{ej} + b_{p_k}^k\right) \end{bmatrix}, \Theta_k = \begin{bmatrix} \Theta_{k1} \\ \vdots \\ \Theta_{kp_k} \end{bmatrix}, \tag{39}$$

where p_k is the number of neurons in the hidden layer. The weights w_{ij}^k and the bias b_i^k for $1 \leq i \leq p_k$, $1 \leq j \leq 5n$ e $1 \leq k \leq n$ are set constant and the activation function $H(\cdot)$ is chosen as the hyperbolic tangent

$$H(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}. \tag{40}$$

In that regard, the full neural network can be described by:

$$\Delta\hat{F}(q_e, \Theta) = \begin{bmatrix} \xi_1 & 0 & \dots & 0 \\ 0 & \xi_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \xi_n \end{bmatrix} \begin{bmatrix} \Theta_1 \\ \vdots \\ \Theta_n \end{bmatrix} = \Xi\Theta. \tag{41}$$

and Θ is calculated by Equation (21).

The nonlinear $\mathcal{H}_\infty$ control parameters used were obtained solving (27)-(29), and they are $\gamma = 0.028$, $q_{11} = 0.5$, $q_{22} = 23$, $q_{12} = 0$, and $r_u = 0.02$, giving us the parameters $t_{11} = 0.14$, $t_{12} = 0.94$ and $k = 3.39$. The adaptive law to adjust the neural network output is based on Equation (21) with $Z = 0.2 \times I_{28\times 28}$ and $k(x_e) = \frac{1}{0{,}2}\sqrt{\tilde{x}^T\tilde{x}}$. We define $\Delta\hat{F}_0(x_e, \Theta) = \left[\Delta\hat{F}_1(x_e, \Theta_1)\ \Delta\hat{F}_2(x_e, \Theta_2)\ \Delta\hat{F}_3(x_e, \Theta_3)\ \Delta\hat{F}_4(x_e, \Theta_4)\right]$ with $p_k = 7$ neurons on the hidden layer, the bias vector $b_k = [\ -3\ -2\ -1\ 0\ 1\ 2\ 3]$ and the weighting matrix for the hidden layer $\Omega_i^k = \left[\omega_{ij}^k\right] = [-1\ -1\ -1\ -1\ -1\ -1\ -1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1]$.

Initially, we determined the Θ values empirically. We started with random values and at the end of each experiment we updated the initialization with the last values found.

Nonlinear $\mathcal{H}_\infty$ Control based on Gaussian Process Offline

The parameters for the Gaussian process offline were defined empirically, obtaining the following values: $\sigma_n^2 = 1 \times 10^{-5}$, $\sigma^2 = 1 \times 10^{-5}$ and $l = 10$. The $\mathcal{H}_\infty$ parameters used were maintained from Sec. 4.1.2.

Nonlinear $\mathcal{H}_\infty$ Control based on Gaussian Process Online

The $\mathcal{H}_\infty$ parameters were the same as in section Sec. 4.1.2. The adaptive law to adjust the covariance, calculated by Equation (34), is implemented based on Equation (21) with $Z = 30 \times I_{4\times 4}$ and $k(q_e) = \frac{1}{0{,}2}\sqrt{\tilde{x}^T\tilde{x}}$. We set $\Delta\hat{F}_0(q_e, \Theta) = \Xi\Theta$ to be calculated and updated at every iteration i of the experiment, defined by the parameters $\sigma^2 = 1 \times 10^{-5}$ e $l = 10$. As in 4.1.2, we determine initially the values for Θ empirically, starting with random values and updating them at the end of each experiment with the last values found. The best values found for this parameter are

$$\Theta^T = [0.6128\ 0.1821\ 0.2602\ 0.4578]\,.$$

4.2 Performance analysis

To analyze the performance among all controllers, we look over to the Euclidean norm in the trajectory tracking simulations to measure the position error and control inputs at each iteration. We have the following equations:

$$E_{q_p}(i) = \frac{1}{N}\sum_{j=1}^{N} ||q_{p_j} - q_{p_j}^d||, \tag{42}$$

for position errors (x, y, z) and

$$V(i) = \frac{1}{N}\sum_{j=1}^{N} ||\nu_j||, \tag{43}$$

for control inputs, where q_p is the vector containing the x, y, and z axis positions; ν is the input control vector; j the current iteration and N the total number of iterations in each flight.

Table 1 shows the results obtained when comparing all the controllers to the Feedback Linearization control. For the performance comparison, we use the average tracking error obtained by Equation (42), and for energy consumption, we use Equation (43), using the Feedback Linearization energy consumption as 100% energy consumed.

Table 1. Comparative study.

-	$\mathcal{H}_\infty$	$\mathcal{H}_\infty$NN	$\mathcal{H}_\infty$off	$\mathcal{H}_\infty$on
Performance (%)	24.82	38.04	38.10	38.75
Energy consumption (%)	130.49	12.57	14.96	12.61

The Neural Network and both Gaussian processes methods give an equivalent performance improvement compared to the Feedback linearization control. The best performance achieved was by the online Gaussian process adaptive algorithm, giving also one of the best energy consumption improvements, right next to the Neural Network which was the lowest energy consumption controller.

4.3 Parametric uncertainties analysis

To grasp the influence of the parametric uncertainties on each controller, we made an experiment increasing the uncertainties on top of the mass and inertia of the system, matrix B_d.

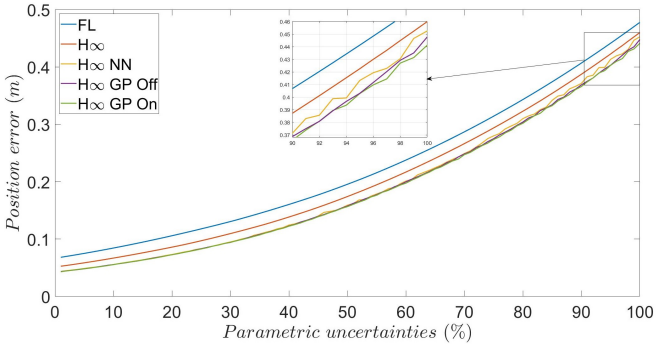

Figure 3. Average error per flight with increasing parametric uncertainty levels.

In Figure 3, we can see that both Gaussian Process controls deal better with uncertainties than all the other controls, achieving almost 10 centimeters of tracking precision against the Feedback Linearization control.

4.4 Wind disturbance analysis

The external disturbances experiment uses the same format as in Section 4.3. The disturbances simulate wind gusts during the quadrotor flight, at 4 seconds on the x axis, 9 seconds on the y axis, and 14 seconds on the z axis.

$$d_x = \nu_{pct}(sin(t-4)^2),$$
$$d_y = \nu_{pct}(sin(t-9)^2),$$
$$d_z = \nu_{pct}(sin(t-14)^2),$$

where ν_{pct} is the desired percentage of the maximum control input for the experiment.

We perform a sequence of increasing external disturbance simulation flights, based on the maximum control input taken from (6), from 0% to 120%.

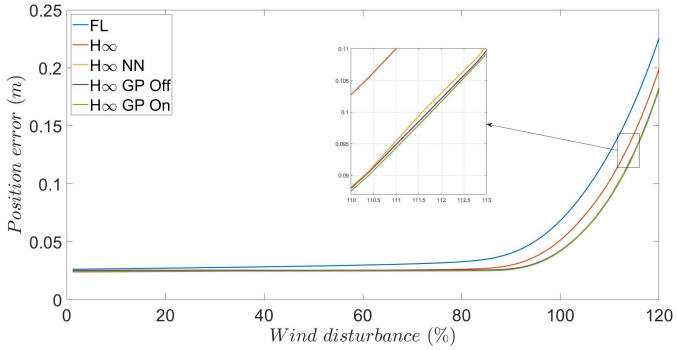

Figure 4. Average error per flight with increasing wind disturbance based on the maximum control input.

Regarding external disturbances, the Gaussian Processes control also performs better than the other controls. After 100% maximum input of disturbance, the error scales in a logarithmic manner since the control can't handle the disturbed acceleration. Until that point, the controls handle periodic wind disturbances with an average error of fewer than 10 centimeters.

5. CONCLUSION

We proposed a nonlinear $\mathcal{H}_\infty$ control based on Gaussian processes in this work. Two approaches were defined; in the first, the Gaussian process is tuned by an adaptive control law, and in the second, we use an offline-trained Gaussian process. We also wrote the dynamic model of a commercial quadrotor as a state-space representation for the nonlinear $\mathcal{H}_\infty$ control and implemented the nonlinear $\mathcal{H}_\infty$ control based on the Gaussian process for comparative analysis. Based on the Parrot Bebop model simulation results, we analyzed the performance and energy consumption, while also verifying the behavior of the controllers due to parametric uncertainties in mass and inertia and external disturbance as wind gusts. The intelligent approaches (Gaussian process and Neural Network) significantly enhanced performance while reducing the energy consumed under parametric variations and disturbances. The intelligent model based on the Gaussian process and Neural Network presented similar performance, with the Gaussian process online method achieving the best performance. In this way, the Gaussian process could be a feasible alternative to the Neural Network, as shown in the results.

REFERENCES

Benevides, J.R.S., Inoue, R.S., Paiva, M.A.D., and Terra, M.H. (2019). Parameter estimation based on linear regression for commercial quadrotors. In *Simpósio Brasileiro de Automação Inteligente (SBAI)*. Ouro Preto, MG.

Berkenkamp, F. and Schoellig, A.P. (2014). Learning-based robust control: Guaranteeing stability while improving performance. In *International Conference on Intelligent Robots and Systems (IROS)*.

Cao, G., Lai, E.M.K., and Alam, F. (2016). Gaussian process model predictive control of unmanned quadrotors. In *International Conference on Control, Automation and Robotics (ICCAR)*. HICO, Gyeongju, Korea.

Castillo, P., R.Lozano, , and A.Dzul. (2005). Stabilization of a mini rotrocraft with four rotors. In *IEEE Control Systems Magazine*.

Chang, Y.C. (2000). Neural network-based $\mathcal{H}_\infty$ tracking control for robotic systems. *IEE Proceedings of Control Theory Applications*, 147(3), 303–311.

Chen, B.S., Chang, Y.C., and Lee, T.C. (1997). Adaptive control in robotic systems with $\mathcal{H}_\infty$ tracking performance. 33(2), 227–234.

Demircioglu, H. and Basturk, H.I. (2017). Adaptive attitude and altitude control of a quadrotor despite unknown wind disturbances. In *IEEE Annual Conference on Decision and Control (CDC)*. Melbourne, Australia.

Johansson, R. (1990). Quadratic optimization of motion coordination and control. *IEEE Transactions on Automatic Control*, 35(11), 1197–1208.

Liu, Y., Ma, J., and Tu, H. (2018). Robust command filtered adaptive backstepping control for a quadrotor aircraft. *Journal of Control Science and Engineering.*, 2018, 1854648:1–1854648:9.

Nogueira, S.L., Pazelli, T.F.P.A.T., Siqueira, A.A.G., and Terra, M.H. (2013). Experimental investigation on adaptive robust controller designs applied to constrained manipulators. *Sensors*, 13, 5181–5204.

Ortiz, J.P., Minchala, L.I., and Reinoso, M.J. (2016). Nonlinear robust h-infinity pid controller for the multivariable system quadrotor. In *IEEE Latin America Transactions (LATAM T)*.

Pazelli, T.F., Terra, M.H., and Siqueira, A.A. (2011). Experimental investigation on adaptive robust controller designs applied to a free-floating space manipulator. *Control Engineering Practice*, 19, 395–408.

Pinto, A.O., Marciano, H.N., Bacheti, V.P., Moreira, M.S.M., Brandão, A.S., and Sarcinelli-Filho, M. (2020). High-level modeling and control of the bebop 2 micro aerial vehicle. In *International Conference on Unmanned Aircraft Systems (ICUAS)*. Athens, Greece.

Raffo, G.V., Ortega, M.G., and Rubio, F.R. (2015). Robust nonlinear control for path tracking of a quad-rotor helicopter. *Asian Journal of Control*, 17(1), 142–156.

Rasmussen, C.E. and Williams, C.K.I. (2006). *Gaussian Processes for Machine Learning*.

Wang, L., Theodorou, E.A., and Egerstedt, M. (2017). Safe learning of quadrotor dynamics using barrier certificates. In *International Conference on Robotics and Automation (ICRA)*. Marina Bay Sands, Singapore.