
Proceedings Paper:
Gonzalez Villarreal, O., Rossiter, J.A. orcid.org/0000-0002-1336-0633 and Shin, H. (2018) Laguerre-based adaptive MPC for attitude stabilization of quad-rotor. In: Proceedings of 2018 UKACC 12th International Conference on Control (CONTROL). Control 2018: The 12th International UKACC Conference on Control, 05-07 Sep 2018, Sheffield, UK. IEEE , pp. 360-365. ISBN 978-1-5386-2864-5

https://doi.org/10.1109/CONTROL.2018.8516876

Laguerre-based Adaptive MPC for Attitude Stabilization of Quad-rotor

1st O. J. Gonzalez Villarreal
Dept. of ACSE
The University of Sheffield
Sheffield, UK
ojgonzalezvillarreal1@sheffield.ac.uk

2nd J. A. Rossiter
Dept. of ACSE
The University of Sheffield
Sheffield, UK
j.a.rossiter@sheffield.ac.uk

3rd H. Shin
Center for A.C.P.S
Cranfield University
Cranfield, UK
h.shin@cranfield.ac.uk

***Abstract*—The application of predictive control methods in real-time to fast systems, such as quad-rotors, remains a challenge for its implementation in low-power embedded systems. This paper presents the application of an Adaptive Laguerre-based Model Predictive Controller (MPC) to the Attitude Stabilization of a Quadrotor. The formulation uses an Online System Identification algorithm based on Recursive Least Squares (RLS) with forgetting factor for parameter estimation, and a Laguerre-based Model Predictive Controller for achieving real-time calculation/update of the control law. The developed control system was experimentally tested in a real quad-rotor, and the results demonstrate its real-time applicability in a low-power embedded platform.**

***Index Terms*—MPC, Laguerre, Adaptive, UAV, Quadrotor**

I. INTRODUCTION

Over the last decade, quad-rotors have become a very popular research topic, given their relatively low-cost, complex nonlinear dynamics and high maneuverability. This has led to all kinds of different applications, such as surveillance, aerial photography, surface mapping, search and rescue, inspection, transport, military and the recently, more popular, FPV racing [1]. Apart from their own inherent complexity, these vehicles are constantly affected by external disturbances, such as wind, as well as changes in the systems' dynamics, which can affect their performance and may require re-tuning to achieve good stability characteristics. As an example, the payload in transportation quad-rotors affects the inertia and mass properties. Similarly, it is common to constantly attach/detach components such as cameras, batteries and external sensors to the vehicle, once again, changing the vehicle's dynamics. These challenges require the implementation of flexible control schemes that are capable of dealing with these uncertainties and sudden changes, and are therefore the motivation behind using the adaptive predictive control scheme presented in this paper.

Model Predictive Control is an advanced optimization-based control strategy that uses an inner model of the system to predict and optimize its future behavior [2], [3]. With the advances in computation platforms and optimization algorithms, the implementation of predictive control algorithms to fast system is now looking more feasible [1]. In the area of UAVs, several authors have implemented MPC, or even Nonlinear MPC, both in simulation and real experiments [1]. In [4], a Laguerre-based MPC was developed and experimentally validated for an hexacopter based on the methodology described in [5], but with no parameter estimation. Other authors, such as [6] and [7] looked at the combination of nonlinear MPC with parameter and state estimation techniques for improving the systems' performance, but only in simulation.

The main contribution of this paper is the formulation of a simple SISO model of a quad-rotor, which differs from models presented in [1] by including actuator (or possibly sensor) dynamics. This model is then used by a real-time feasible Laguerre-based MPC control law, able to match a PD control law with the main advantage of including the desired frequency content of the Laguerre Polynomials in the design. Furthermore, this is combined with a computationally inexpensive Online System Identification algorithm for estimation of 3 parameters that define the systems dynamics with the goal of achieving auto-tuning. Moreover, the entire formulation is experimentally tested in a quad-rotor UAV, and the tests demonstrate successful implementation in the relatively new Beaglebone Blue board, which is a low-power embedded platform. The entire formulation is available from https://github.com/OscarJGV26/LaguerreMPC using object oriented programming and Matlab codes. In summary, the paper presents the application of a novel Adaptive Laguerre-based Model Predictive Controller (MPC) for Attitude Stabilization, experimentally tested in a Quad-rotor using relatively new hardware.

Section II presents the full nonlinear attitude model of a quad-rotor, and derives a linear SISO model to be used by the formulation presented in this paper with its respective assumptions. Section III presents the formulated Online System Identification using Recursive Least Squares (RLS) with a forgetting factor and discusses important aspects to be considered for its implementation. Section IV outlines the Laguerre-based MPC formulation and discusses some important remarks. Section V presents the results obtained from real experiments where the system dynamics were automatically excited using a sinusoidal signal for auto-tuning, and the convergence and computational benefits of the overall control systems are discussed. Finally, Section VI gives conclusions and future work.

II. Quadrotor Modeling

It is well known that the attitude model of a quad-rotor has several sources of nonlinear dynamics such as quaternion/euler dynamics, thrust relations and coriolis-centripetal crossed-coupled angular velocity effects [6]. Nevertheless, many authors have simplified them into linear models [4]. This section presents the fully nonlinear attitude model of a quad-rotor, and derives the associated simple linear SISO model to be used for the control design presented in this paper.

Recalling the modeling done in [6], [4] and [2]; the full nonlinear attitude dynamics of a quad-rotor can be represented by:

$$\begin{bmatrix}\dot{q}_0\\ \dot{q}_1\\ \dot{q}_2\\ \dot{q}_3\end{bmatrix} = \frac{1}{2}\begin{bmatrix}0 & -p & -q & -r\\ p & 0 & r & -q\\ q & -r & 0 & p\\ r & q & -p & 0\end{bmatrix}\begin{bmatrix}q_0\\ q_1\\ q_2\\ q_3\end{bmatrix} \tag{1}$$

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}\frac{k_T l}{I_{xx}} & 0 & 0\\ 0 & \frac{k_T l}{I_{yy}} & 0\\ 0 & 0 & \frac{k_\tau}{I_{zz}}\end{bmatrix}\begin{bmatrix}1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\end{bmatrix}\begin{bmatrix}\omega_1^2\\ \omega_2^2\\ \omega_3^2\\ \omega_4^2\end{bmatrix} + \begin{bmatrix}\frac{I_{yy}-I_{zz}}{I_{xx}}qr\\ \frac{I_{zz}-I_{xx}}{I_{yy}}pr\\ \frac{I_{xx}-I_{yy}}{I_{zz}}pq\end{bmatrix} \tag{2}$$

where $[q_0, q_1, q_2, q_3]^T$ is the quaternion in the inertial frame, $[p, q, r]^T$ are the angular velocities in the body axes frame, $[\omega_1, \omega_2, \omega_3, \omega_4]^T$ are the propellers' angular velocities, I_{xx}, I_{yy}, I_{zz} are the vehicle's inertias, k_T, k_τ are constants relating the propellers' angular velocities and thrust/torque respectively, and l is the distance parallel to the respective axis from center of gravity (CG) to the propeller.

The dynamics of the quaternion (1) are affected by the data-fusion method used to correct drift, e.g. using accelerometer data [8]. Therefore, in order to avoid being affected by this in the Online System Identification phase, the formulation will focus on the rate dynamics (2) and use a cascade proportional control loop as common control loops. This can be improved further by using a quaternion based control such as in [9] or [10].

Now, assuming the cross-coupled angular velocity terms are negligible around the operating point $p \approx q \approx r \approx 0$, and assuming that at this operation point there exist a linear relationship between the input signal u_i (i.e. the signal that goes into the Electronic Speed Controllers (ESCs)) and the propellers' angular velocities ω_i, the rate dynamics can now be represented by:

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = K\begin{bmatrix}L\\ M\\ N\end{bmatrix} \tag{3}$$

where K is a diagonal matrix with the gains of each axis given by:

$$K = \begin{bmatrix}k_x & 0 & 0\\ 0 & k_y & 0\\ 0 & 0 & k_z\end{bmatrix} \tag{4}$$

L, M, N represent the allocated "virtual moments" as in [11] given by:

$$\begin{bmatrix}L\\ M\\ N\\ Z\end{bmatrix} = \frac{1}{4}\begin{bmatrix}1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\\ 1 & 1 & 1 & 1\end{bmatrix}\begin{bmatrix}u_1\\ u_2\\ u_3\\ u_4\end{bmatrix} \tag{5}$$

and the resulting control allocation is given by:

$$\begin{bmatrix}u_1\\ u_2\\ u_3\\ u_4\end{bmatrix} = \begin{bmatrix}1 & 1 & 1 & 1\\ -1 & 1 & -1 & 1\\ 1 & -1 & -1 & 1\\ -1 & -1 & 1 & 1\end{bmatrix}\begin{bmatrix}L\\ M\\ N\\ Z\end{bmatrix} \tag{6}$$

We introduce Z to represent a potential offset of "thrust" to be produced by each of the 4 motors. The derivation of the control allocation matrices is out of the scope of this paper.

These dynamics represent a simple integrator and implementing an MPC directly onto then result in a proportional controller which is unlikely to give the desired tracking performance in the rate dynamics whilst also being robust to external disturbances. A key assumption for this model is that the propeller angular velocities are "instantaneously" achieved, which is not true, and nevertheless, has been used by many authors (see [1], [4]) Based on this potential problem, the model was further augmented with unit-gain first-order dynamics representing the actuators (or possibly even the sensors) dynamics. Combining both models gives the following second order dynamics with an integrator and with $\tau > 0$ related to the time constant of the first order; these will be used for the control law formulation.

$$\begin{bmatrix}\ddot{p}\\ \ddot{q}\\ \ddot{r}\end{bmatrix} = -\frac{1}{\tau}\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} + K\begin{bmatrix}L\\ M\\ N\end{bmatrix} \tag{7}$$

III. Online System Identification

Although the parameters of the dynamic model of a vehicle can be calculated and pre-stored off-line using for example CAD or system identification methods, the application of online system identification allows quick recalculation of the system's true dynamic model and therefore, can enhance the system performance. Additionally, it can be used for supervisory control and fault-detection/isolation, as well as fault-tolerant control; however this is outside the scope of this paper.

A. Algorithm and Modeling

In this case, the Online System Identification was based on the Recursive Least Squares (RLS) with forgetting factor

formulation presented in [12]. The algorithm equations are given by:

$$\begin{aligned} e_k &= z_k - \Psi^T \Theta \\ K &= \tfrac{P\Psi}{1+\Psi^T P \Psi} \\ \Theta_k &= \Theta_{k-1} + K e_k \\ P &= \tfrac{I - K\Psi^T}{\lambda} P \end{aligned} \tag{8}$$

where e_k is the prediction error of the mode, Ψ is known as the regressors vector, Θ is the parameters vector, P is the covariance matrix, K is a Kalman-Filter-type gain and λ is the forgetting factor. For our system, a forgetting factor of $\lambda = 0.999$ was selected.

This formulation was combined with the assumed model (7) given in the previous section. Although a classic approach would formulate this model as a standard ARX model represented by a discrete difference equation of the form:

$$y_k = a_1 y_{k-1} + a_2 y_{k-2} + b_1 u_{k-1} + b_2 u_{k-2} \tag{9}$$

which has $n_a = 2$ recursive terms and $n_b = 2$ exogenous terms. In our case, we implemented the Δ modeling and identification approach given in [13]. This allows the embedding of the desired model structure into the system whilst also improving the precision of the coefficients by requiring 2 less coefficients to be estimated. Moreover, the learned system must consider possible disturbances or un-modeled biased errors. Therefore the parameters to be estimated were augmented with a constant disturbance. The algorithm entry-data (z_k, Ψ) are then given by:

$$\begin{aligned} z_k &= y_k - 2y_{k-1} + y_{k-2} \\ \Psi &= \begin{bmatrix} y_{k-1} - y_{k-2} \\ u_{k-1} \\ 1 \end{bmatrix} \quad \Theta = \begin{bmatrix} a \\ b \\ d \end{bmatrix} \end{aligned} \tag{10}$$

where y is a general output-variable, which in this case represents the angular velocity of the vehicle, and u_k is a general input-variable, which in this case represent the "virtual moments". Once the model is learned, by rearranging the equations, the state space model to be used later (see eqn. (13)) can be found.

B. Execution Rules and Parameter Constraints

The performance of the algorithm presented above is known to converge to the real parameters if and only if: (i) the assumed model (7), or in general, the entry data (10) can actually represent the system dynamics, and (ii) if the system is under persistent excitation periods [12]. Therefore, in order to prevent unwanted adaptation and learning actions in period of low excitation, several execution rules and parameter constraints were implemented and are listed and explained below based on the ideas discussed in [12], with the selected thresholds for our system.

1) Run the RLS algorithm only when the "angular acceleration" $y_{k-1} - y_{k-2} > 5$ is greater than a threshold to provide sufficient excitation and avoid running the algorithm when steady, e.g. when hovering. Additionally, the criteria found in [12] based on the normalized information was also used:
$$\frac{||P\Psi||_1}{||P||_1 ||\Psi||_1} < 0.1 \tag{11}$$
2) Limit the trace of the co-variance matrix (P) with
$$P = \frac{k_{lim}}{tr(P)} P \quad \textit{if} \quad tr(P) > k_{lim} \tag{12}$$
to prevent it from becoming ill-conditioned and have better control on the rate of convergence. A threshold of $k_{lim} = 10$ was selected for our system.
3) Limit the range of parameters a and b to an expected range to prevent incorrect modeling. For our system, the thresholds $-0.7 < a < 0$, $0.05 < b < 0.3$ for roll/pitch axis, and $0.005 < b < 0.05$ for yaw axis, were selected and can be obtained considering variation in the time constant and gain of the system.
4) Only update the control law (i.e. adapt) if the overall uncertainty of your first two parameters/coefficients a, b, which can be considered as the summation of $P_{1,1} + P_{2,2}$, is sufficiently small. This not only ensures that adaptations are made when you can actually trust your coefficients, but also when they are moving slowly. For this system, a maximum trace of $k_{max} = 0.0001$ was selected.

Remark 1 (Saturation): One thing to be careful with when using this formulation is the saturation of the actuators. Whenever this happens, the allocated control values do not represent the same values of the "virtual moments" in matrix (6). Therefore, in this case, the system must recalculate the actual allocated "virtual moments" values with matrix (5) after saturation.

IV. Laguerre-based Model Predictive Control

Laguerre-based MPC uses a set of discrete orthonormal basis functions embedded into the design. The main motivations behind using Laguerre are: (i) the possible recovery of the fully optimal solution; (ii) the acceleration of the computation times and (iii) a better frequency control on the system's response. Furthermore, by imposing a fast zero-decaying structure, it prevents the optimization from becoming ill-conditioned in case of plants with unstable/conditionally stable dynamics. The methodology can be found in [14] and [5].

A. Model

One important difference in this implementation is the model to be used. Most MPC implementations use the Δ modeling approach where the system is augmented with an integrator. However, based on the results from [15], it was chosen to use a disturbance estimation model which indeed gave better performance as well as improved control over disturbance rejection, thus motivating its use for unbiased-predictions. Furthermore, in order to consider a possible match to a proportional-derivative (PD) controller, the model was

transformed to an equivalent by using a simple backward-forward euler integration method. Finally, in order to be able to formulate the unbiased optimization-index (16), the model had to be represented with difference inputs (Δu_k), rather than the absolute values (u_k). Thus, the state space model used for this formulation is given by:

$$\begin{aligned} x_{k+1} &= Ax_k + B\Delta u_k \\ y_k &= Cx_k \end{aligned} \tag{13}$$

where:

$$A = \begin{bmatrix} 1 & T_s(1+a) & T_s & b \\ 0 & (1+a) & 1 & \frac{b}{T_s} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} b \\ \frac{b}{T_s} \\ 0 \\ 1 \end{bmatrix}$$
$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

T_s is the sampling time and the state defined as $x_k = \begin{bmatrix} y_k & \frac{y_k - y_{k-1}}{T_s} & d_k & u_{k-1} \end{bmatrix}^T$, with d_k being the disturbance to be estimated. From now on, we will refer to $\dot{y}_k = \frac{y_k - y_{k-1}}{T_s}$.

To estimate the disturbance a full Kalman Filter can be employed as in [15], or alternatively, the disturbance can be simply filtered with a unit-gain first-order discrete filter given by:

$$\begin{aligned} d_k = \alpha d_{k-1} + \\ (1-\alpha)(\dot{y}_k - (1+a)\dot{y}_{k-1} + \tfrac{b}{T_s}u_{k-1}) \end{aligned} \tag{14}$$

where $0 < \alpha < 1$ is the tuning constant. For our system, a value of $\alpha = 0.98$ was selected.

B. Control Law

The derivation of the control law is based on finite horizon optimal control using a relatively long horizon which is known to give good stability characteristics [3], [16]. Given that the purpose of the optimization is to calculate a control law in real-time whilst preserving computational simplicity, dual-mode approaches were avoided.

In this section, it will be shown that the formulation leads to a control law of the form:

$$u_k = u_{k-1} - K_x x_k + K_r r_k + K_{\dot{r}} \dot{r}_k \tag{15}$$

where K_x, K_r and $K_{\dot{r}}$ vary depending on the system dynamics.

To achieve this, a standard unbiased-optimization index of the form:

$$J = (\boldsymbol{r} - \hat{\boldsymbol{y}})^T (\boldsymbol{r} - \hat{\boldsymbol{y}}) + \boldsymbol{\Delta\hat{u}}^T R \boldsymbol{\Delta\hat{u}} \tag{16}$$

is defined, where $\hat{\boldsymbol{y}} = \begin{bmatrix} y_{k+1} & y_{k+2} & \cdots & y_{k+N_p} \end{bmatrix}^T$ and $\boldsymbol{\Delta\hat{u}} = \begin{bmatrix} \Delta u_k & \Delta u_{k+1} & \cdots & \Delta u_{k+N_p-1} \end{bmatrix}^T$ represent the vectors of predicted outputs and future input-increments trajectories respectively, N_p is the prediction horizon, and $\boldsymbol{r} = \begin{bmatrix} r_{k+1} & r_{k+2} & \cdots & r_{k+N_p} \end{bmatrix}^T$ represents a reference trajectory, typically a constant r_k.

The unbiased-predicted output $\hat{\boldsymbol{y}}$ is represented by:

$$\hat{\boldsymbol{y}} = Gx_k + H\boldsymbol{\Delta\hat{u}} \tag{17}$$

with

$$G = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N_p} \end{bmatrix} \quad H = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ CA^{N_p-1}B & \cdots & \cdots & CB \end{bmatrix} \tag{18}$$

Now, the dynamics of the Laguerre Polynomials can be found in [5] and are given by,

$$\begin{bmatrix} L(1) \\ L(2) \\ \vdots \\ L(N_L) \end{bmatrix}_{k+1} = \begin{bmatrix} a_L & 0 & \cdots & 0 \\ \beta & a_L & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \cdots & \cdots & \cdots & a_L \end{bmatrix} \begin{bmatrix} L(1) \\ L(2) \\ \vdots \\ L(N_L) \end{bmatrix}_k \tag{19}$$

where, a_L is the decay-rate, $\beta = (1 - a_L^2)$ and N_L is the number of Laguerre coefficients.

By taking $L_0 = \sqrt{\beta}\begin{bmatrix} 1 & -a_L & \cdots & (-1)^{N_L-1} a_L^{N_L-1} \end{bmatrix}^T$ as initial condition and iterating system (19) forward N_p times, the following input structure can be embedded into the optimization

$$\boldsymbol{\Delta\hat{u}} = L\eta \tag{20}$$

where $L = \begin{bmatrix} L_0^T & L_1^T & \cdots & L_{N_p-1}^T \end{bmatrix}^T$ and $\eta = \begin{bmatrix} c_1 & c_2 & \cdots & c_{N_L} \end{bmatrix}^T$ are the Laguerre coefficients [5]. For our system, the number of Laguerre coefficients was fixed with $N_L = 2$ and the decay-rate was fixed at $a_L = 0.5$, which contains a desirable frequency response of the input. Figure (1) shows the embedded input parameterization.

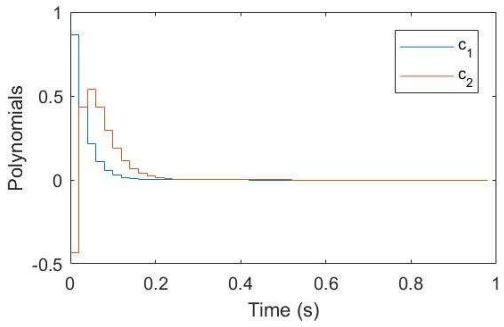

Fig. 1. First two Laguerre Polynomials.

By substituting equations (20) and (17) in (16), it can then be derived that the optimization is of the standard Quadratic Problem (QP) form:

$$J = \frac{1}{2}\eta^T E \eta + f^T \eta \tag{21}$$

where $E = L^T H^T H L + L^T R L$, $f^T = -L^T H^T (r - Gx_k)$. This optimization will give the optimal Laguerre coefficients η which can then be used to recover the solution in the original space using (20). Recalling the receding horizon strategy [5], only the first input is applied and the optimization is recalculated in the following next step. If the system dynamics

are linear time-invariant, then, the unconstrained solution of this QP is of the form of (15) with:

$$K_x = -\begin{bmatrix}1 & 0 & \cdots & 0\end{bmatrix} LE^{-1}L^T H^T G \tag{22}$$

and it can be shown that $K_r = K_x(1)$. To further match a PD controller and give better trajectory tracking, the control law included a desired angular acceleration, $\dot{r}_k = \frac{r_k - r_{k-1}}{T_s}$. Thus, the final control law is given by (15) with $K_{\dot{r}} = K_x(2)$.

Remark 2 (Coding): Significant computation savings can be achieved by proper coding and memory allocation of this optimization, e.g. by using recursive information [2]. Efficient Matlab code for this is available from https://github.com/oscarjgv26.

Remark 3 (Saturation): Because it is unconstrained, the saturation of the actuators will be done using anti-windup techniques as discussed in [4] as, in such cases, Δ based control laws are known to have good anti-windup properties.

V. Experimental Results

This formulation was tested in an F450 quad-rotor frame with EMAX MT2213 brushless motors, ESCs operating at 400 Hz and 1045 ABS propellers. The flight control system running this formulation was a Beaglebone Blue running @ 1000 MHz and the formulations were compiled using -O3 C flag. The filter of both on-board gyroscope and accelerometer was set at 10 Hz, and the resolutions were set at 2000 (deg/s) and $4G$ respectively. The accelerometer was fused using the double-stage Kalman Filter presented in [8] but only performing accelerometer correction with $Q = 10^{-6}I$ and $R_{acc} = 0.1$. The sampling time of the control system was $T_s = 20(ms)$ (50 Hz) and the prediction horizon was fixed at $N_p = 50$, i.e. $T_p = 1\ (s)$. The outer loop proportional gains for the Roll and Pitch axis were both set on $K_p = 4$ and left constant throughout the tests. The input range of each of the motors was $u_i = [0, 1000]$, and the allocated moments (v_i) were saturated at approximately 60% of the overall hovering throttle Z which depends on the mass, thus time-varying but for our system, approximately $v_i = [-300, 300]$.

One of the most important, and also difficult things to achieve was the stability of the combined online system identification algorithm and control law update. This is because if the standard RLS algorithm is applied, the system can diverge during periods of poor excitation which leads to the need for the rules presented in section III-B. Another important part was that, regardless of the identified model, or in general, the performance of the online system identification algorithm, there are still three constants that determine the performance of the system, namely the input-weight R, the disturbance estimation filter α and the Laguerre decay-rate a_L. For the purpose of this paper, these constants were fixed at the values $R = 0.1, \alpha = 0.98, a_L = 0.5$, where "stability" and "smoothness" were observed across the tests. However, in general, these parameters do need to be assigned carefully, in particular taking into account the possible embedded frequency response of both Laguerre decay-rate and disturbance estimation constant α which could have a substantial effect on the system.

A. Flight Tests

To test the formulation, the system was excited in the roll axis, starting from a poorly tuned control law, and moving iteratively towards the optimal value. Figure (2) shows approximately 7 seconds of the flight test data. At the beginning ($t < 1$), the system is showing the performance of the poorly tuned control law. At $t \approx 1$, an automatically generated sinusoidal signal of $\phi = 50\sin(4\pi)$ in the roll angle is implemented for approximately 3 seconds to provide initial excitation and the online system identification starts (see figure (3)). The control law starts updating at $t \approx 4$ (see figure (4)) and maintains the same values after the pilot terminates the sinusoidal excitation to test the performance. As it can be seen, the "chattering" of the inputs was reduced to ± 10, thus giving very precise corrections and reducing actuator wear. Once again, the flight data is available at https://github.com/OscarJGV26/LaguerreMPC.

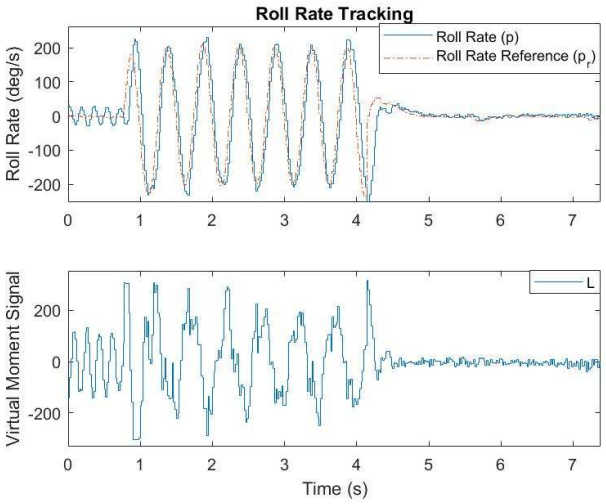

Fig. 2. Excitation Data - Roll Axis

Furthermore, to test the full learning capabilities, the RLS algorithm started from an empty parameter vector $\Theta = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^T$ during this test. The variation of the principal estimated coefficients (a, b) can be seen in figure (3). As it can be seen, they move near to the final value within less than 3 seconds whilst also staying within the constraints, and move smoothly afterwards as the algorithm iterates .

During the same test, figure (4) shows the movement of the control law parameters which can be seen to be moving equally smoothly and do not change during the first 3 seconds of the RLS algorithm where the uncertainty is still high. Similar results were obtained for the other 2 axis (pitch/yaw) and the formulation was able to tune all the axis simultaneously with the sinusoidal signal within 10 sec.

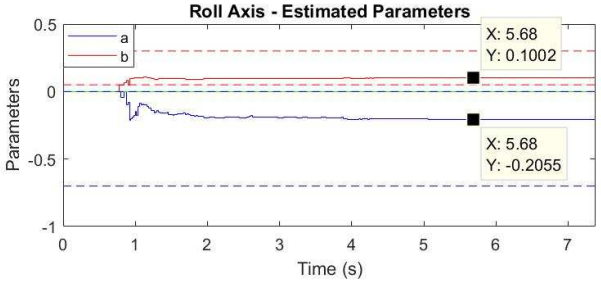

Fig. 3. Online System Identification - Estimated Coefficients

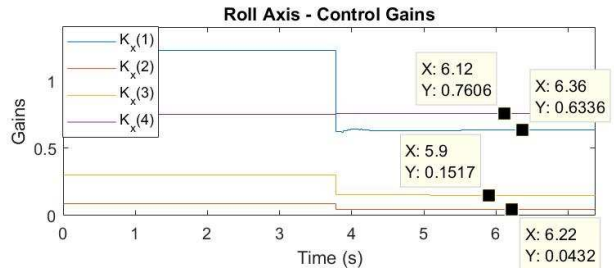

Fig. 4. Control Law Gains

B. Computation Times

One of the requirements of this formulation is that it is able to be run in real-time whilst performing other tasks such as data-fusion, telemetry, data-logging and so forth. Table I shows the computation times in miliseconds (t_c) of the formulation, where updates of the control law in 235 microseconds, and execution of the online system identification in 4.8 microseconds (per axis) can be seen, leaving more than enough time for other tasks. Furthermore, it includes the computation times of 30 and 50 iterations for solving the Discrete Algebraic Ricatti Equation (DARE) for comparison.

TABLE I
ONLINE COMPUTATION TIMES

Task	$t_c(ms)$
Control Law Execution on 3 axis	0.0033
Online System Identification on 3 axis (RLS)	0.0144
Laguerre MPC ($N_L = 2$)	0.235
LQR (30 iterations of DARE)	0.165
LQR (50 iterations of DARE)	0.273

Although 30 iterations of DARE for computing the LQR control law are faster, the tuning process of LQR didn't give good performance given the lack of frequency content embedded into the optimization. Thus, the motivation behind using Laguerre Polynomials was validated.

VI. Conclusion

This paper presented the implementation of a Laguerre-based Model Predictive Control formulation coupled with Recursive Least Squares with forgetting factor as an Online System Identification method for achieving adaptive self-tuning control of a quad-rotor UAV in real-time. This formulation was experimentally tested in a quad-rotor and could equally be applied to other UAVs with similar dynamics using the control allocation concept.

The developed control system combined the standard outer loop proportional cascade loop control with a control allocation strategy, an online system identification method and a Laguerre-based Model Predictive Control.

The performance of the developed formulation was tested using an automated sinusoidal signal for exciting the system's dynamics and the results demonstrate the capability of the formulation to achieve self-tuning of the system in real-time within less than 3 seconds per axis with overall smooth performance of the parameters. A demonstration of this implementation will be given at the conference.

Future work related to this formulation will be the extension to the multi-variable system identification case and consideration of other types of UAVs.

References

[1] A. Zannelli, G. Horn, G. Frison, and M. Diehl, "Nonlinear Model Predictive Control of a Human-sized Quadrotor," *European Control Conference*, vol. 16, no. 1, pp. 41–50, 2018.

[2] R. Quirynen, S. Gros, and M. Diehl, "Efficient NMPC for nonlinear models with linear subsystems," *Proceedings of the IEEE Conference on Decision and Control*, pp. 5101–5106, 2013.

[3] J. A. Rossiter, *A first course in predictive control: 2nd edition*. CRC Press, 2018.

[4] J. A. J. Ligthart, P. Poksawat, and L. Wang, "Experimentally Validated Model Predictive Controller for a Hexacopter," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4137–4142, 2017.

[5] L. Wang, *Model Predictive Control System Design and Implementation Using Matlab*. Springer, 2009.

[6] M. H. Tanveer, D. Hazry, S. F. Ahmed, M. K. Joyo, F. A. Warsi, H. Kamaruddin, M. Zuradzman, K. Wan, and A. B. Shahriman, "NMPC-PID Based Control Structure Design for Avoiding Uncertainties in Attitude and Altitude Tracking Control of Quadrotor," *IEEE 10th International Colloquium on Signal Processing & its Applications*, pp. 7–9, 2014.

[7] S. Iplikci, "RungeKutta model-based adaptive predictive control mechanism for non-linear processes," *Transactions of the Institute of Measurement and Control*, vol. 35, no. 2, pp. 166–180, 2013.

[8] S. Sabatelli, M. Galgani, L. Fanucci, and A. Rocchi, "A double-stage kalman filter for orientation tracking with an integrated processor in 9-D IMU," *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 3, pp. 590–598, 2013.

[9] S. Bouhired, M. Bouchoucha, and M. Tadjine, "Quaternion-based global attitude tracking controller for a quadrotor UAV," *2013 3rd International Conference on Systems and Control, ICSC 2013*, pp. 933–938, 2013.

[10] E. Fresk and G. Nikolakopoulos, "Full Quaternion Based Attitude Control for a Quadrotor," *European Control Conference*, pp. 3864–3869, 2013.

[11] T. A. Johansen and T. I. Fossen, "Automatica Control allocation A survey," *Automatica*, vol. 49, no. November, pp. 1087–1103, 2013.

[12] S. F. Campbell, N. T. Nguyen, J. Kaneshige, and K. Krishnakumar, "Parameter Estimation for a Hybrid Adaptive Flight Controller," *AIAA Infotech@ Aerospace Conference, Seattle, WA*, no. April, pp. 1–27, 2009.

[13] S. R. Anderson and V. Kadirkamanathan, "Modelling and identification of non-linear deterministic systems in the delta-domain," *Automatica*, vol. 43, no. 11, pp. 1859–1868, 2007.

[14] B. Khan and J. Rossiter, "Alternative Parameterisation within predictive Control: a systematic selection," *IJC*, vol. 86, no. 8, pp. 1397–1409, 2013.

[15] J. Huusom, N. Poulsen, S. Jorgensen, and J. Jorgensen, "Tuning of methods for offset free MPC based on ARX model representations," *American Control Conference (ACC), 2010*, pp. 2355–2360, 2010.

[16] E. F. Camacho, *Model predictive control*. Advanced textbooks in control and signal processing, London ; New York: Springer, 2nd ed. ed., 2003.