Predictive Minimum Variance Control: a Case Study on Wireless Altitude Hold Autopilot

Antonio Silveira, Carlos Nogueira, Lucas Sodré, Daniel Silva

Laboratory of Control and Systems, Federal University of Pará, PA,
(e-mails: asilveira@ufpa.br, carlos.nogueira@itec.ufpa.br,
lucas.sodre@itec.ufpa.br, daniel.abreu@itec.ufpa.br).

Abstract: Unmanned aerial vehicles, or UAVs, used in collaborative missions, are commonly supervised in flight by a master guidance, navigation and control system that must verify that every flight vehicle is performing the mission as prescribed. Within this scenario, the control system may be separated from the controlled UAVs by great distances, leading to data loss, increased transport time delays and reduced control stability margins. In this work this scenario is reproduced experimentally using a single UAV quadcopter and a remote master control unit connected by a wireless network. By indoor laboratory flight measurements a considerable time delay was verified and embedded in the UAV thrust to altitude estimated model. Then, a long-range prediction horizon control solution, based on the Generalized Minimum Variance Control in the State-Space, was investigated and its results compared to a benchmark controller. The predictive controller outperformed the benchmark one with respect to reference tracking dispersion minimization and control signal chattering reduction in the task of controlling the altitude of a quadcopter over a wireless network in real flight experiments. This work also discusses further insights on predictive minimum variance control from the perspective of the so-called general Bolza optimal control problem in order to remark the differences from most common model predictive control techniques.

Keywords: Minimum variance control, Kalman filter, Model predictive control, Unmanned aerial vehicle, Altitude hold autopilot.

1. INTRODUCTION

In this work, it is investigated the experimental application of the Generalized Minimum Variance in State-Space method (GMVSS) to design an altitude hold autopilot for a quadcopter and the theoretical background of Predictive Minimum Variance Control is revisited in order to analyze its optimization aspects from the perspective of the so-called Bolza optimal control problem (Sagliano, 2018; Sagliano et al., 2022), which assumes the optimization problem comprised of: the end of mission cost and the running or transitory cost.

Within the experimental part of this work, the main problem is the design of a networked autopilot for a quadcopter which is connected over a wireless network in an indoor laboratory environment. This configuration can be used in collaborative missions with unmanned aerial vehicles (UAV) when a master guidance, navigation and control system (GNC) is required to supervise and command every flight vehicle involved.

This distributed control system counts on embedded stability and control augmentation systems aboard the UAVs to allow them to be stable and maneuverable, but the required maneuvers come from a remote guidance and navigation system, such as the collaborative missions done with quadcopters at ETH Zurich Flying Machine Arena (FMA): inverted pendulum pole throwing and catching between two quadcopters (Brescianini et al., 2013); cooperative construction with flying machines (Augugliaro et al., 2014); and rhythmic flight performances with quadcopters merging controls and arts (Schoellig et al., 2014).

The Flying Machine Arena used a very complex set of equipment and software that was being architectured and developed since 2007. Lupashin et al. (2014) presented some of FMA's major features, such as how the control and feedback data flows inside the FMA. They used multicast User Datagram Protocol (UDP) stream, which means, "send once to anyone that listens, unreliably but often" (Lupashin et al., 2014).

User Datagram Protocol is the same communication protocol we have been using to control drones at the Federal University of Pará, in the Laboratory of Controls and Systems (LACOS). It is being used over a TCP/IP wireless network that connects the quadcopters to middleware running inside a PC. And in spite of its unreliability due to the lack of sync, UDP gives better sampling speeds to work with the distributed control system. This setup also allows us to test complex estimation and control algorithms, since this systems can be embedded in a PC instead of a more restricted hardware aboard the UAV.

A drawback of this distributed control topology is that it may experience loss of network packets, which means loss of sensor and command data. Also, the control transmission latency, which is the transport time delay of commands over the wireless network plus the time delay of

sensor data to come back to the ground station, is another disadvantage. At the Flying Machine Arena, for example, Lupashin et al. (2014) presented experimental results of this latency compensation by state prediction using a scheme that assumes there is no noise or uncertainties in the plant model. The predicted state was considered to be the current state when the control command arrives at the quadcopter. In Hofer et al. (2016), a model-based predictive control scheme was implemented and tested at FMA, but embedded aboard the quadcopter.

Model Predictive Control (MPC) is one good option to workaround networked UAV control problems. Thus, we evaluate such an option by testing a MPC technique that can optimally deal with the problem of model uncertainties and noise, along with network latency compensation.

This work intends to assess in flight UAV experiments with a distributed GNC system using long-range predictive minimum variance control (PMVC). The vertical thrust to altitude control-loop is analyzed in order to build an altitude hold autopilot which minimizes the control signal chattering due to the presence of noisy measurements. Beyond this introductory part, this work presents the following: the GMVSS method is presented; the experimental setup is discussed and the networked quadcopter is modeled by means of least-squares system identification; simulations and experimental flight results are presented, followed by the conclusions.

2. THE GMVSS METHOD

The GMVSS is a Predictive Minimum Variance Control method based on the following model:

$$x(k) = Ax(k-1) + Bu(k-d) + \Gamma w(k-1), \quad (1)$$

$$y(k) = Cx(k) + v(k), \quad (2)$$

where $x(k)$ is the state vector, $u(k)$ is the input vector, $y(k)$ is the output vector, $w(k)$ and $v(k)$ are Gaussian disturbances with respectively σ_w^2 and σ_v^2 variances. The A, B, C, Γ are the system's matrices of appropriate dimensions, $d \geq 1$ is the discrete time delay.

When the stochastic part of the state model shown in (1) and (2) is known, i.e. $\Gamma w(k-1)$ and $v(k)$, the GMVSS method can exploit this to design an optimal stochastic state predictor to cope with minimum variance control objectives, such as: increased performance with reduced sensitivity to noise, uncertainties and delay.

The GMVSS method was first published in 2011 while aiming to avoid the solution of the Diophantine equations in the design of long-range minimum variance predictors (MVP) and it has consequently allowed its use as a MPC controller (Silveira et al., 2016, 2020). The major differences from other well-known MPC methods is that it can use a *tricked* predictor, by a virtual time delay, and only the final or steady-state performance is considered in the optimization.

For a better understanding of such differences, let us consider the continuous-time optimal control problem, posed in the so-called generalized Bolza form as follows (Sagliano, 2018):

$$\min \; J = \mathbf{\Phi}\left[t_f, \mathbf{x}(t_f)\right] + \int_{t_0}^{t_f} \mathbf{\Psi}\left[t, \mathbf{x}(t), \mathbf{u}(t)\right] dt, \quad (3)$$

subject to the controlled system in which its state, $\mathbf{x}(t)$, is governed by

$$\dot{\mathbf{x}}(t) = \mathbf{f}\left(t, \mathbf{x}, \mathbf{u}\right), \quad (4)$$

and to the constraints

$$\mathbf{g}_L \leq \mathbf{g}\left(t, \mathbf{x}, \mathbf{u}\right) \leq \mathbf{g}_U. \quad (5)$$

For every t there exists a control solution $\mathbf{u}(t)$ that complies with the minimization of the final (e.g., steady-state, end of mission) performance, $\mathbf{\Phi}[.]$, and the running cost, $\mathbf{\Psi}[.]$, from the initial time t_0 (supposed to be known) to the final time t_f, also guaranteeing that the path constraints $\mathbf{g}(.)$ are kept within the specified lower and upper bounds, $\mathbf{g}_L$ and $\mathbf{g}_U$, respectively, and similar constraints also hold for lower and upper bounds for $\mathbf{x}(t)$ and $\mathbf{u}(t)$.

The general Bolza optimal control problem in (3) is only covered partially in this work in predictive minimum variance control, where in the discrete-time form the final mission time t_f is substituted by a virtual discrete time delay N_y. Then, this optimal control problem is described as:

$$\min \; J = \mathbf{E}\left[\phi(k+N_y)\right]^2 \approx \left[\hat{\phi}(k+N_y|k)\right]^2 \quad (6)$$

subject to (1) and (2), and the estimated generalized output, predicted $k+N_y$ steps in the future, but based on information known from the past and up to time k, given by:

$$\hat{\phi}(k+N_y|k) = \hat{y}(k+N_y|k) - y_r(k) + \lambda u(k), \quad (7)$$

which is consisted of the predicted output $\hat{y}(k+N_y|k)$, based on information known up to instant k; the reference sequence $y_r(k)$; and the λ weighted control effort. N_y is the final time which tricks the predictor as a virtual delay, thus can also be considered as the prediction horizon in MPC and can be used along with λ to tune the control-loop, where long-range prediction minimize the variance of $\hat{\phi}(k+N_y|k)$, and higher λ reduces the control effort.

Observe that (6) is an expected cost, as denoted by the expectation operator $\mathbf{E}[.]$, since the generalized output $\phi(k+N_y)$ is stochastic. Also, the predictive minimum variance control differs from commonly known MPC, for example the Generalized Predictive Control or Dynamic Matrix Control, because it lacks the running cost part, $\mathbf{\Psi}[.]$. Thus, all efforts from PMVC are directed to the final time or the end of mission.

The MVP in the state-space was shown by Silveira et al. (2016) to be

$$\hat{x}(k+N_y|k) = (A^{N_y} - FC)\bar{x}(k) + \sum_{i=1}^{Ny} A^{(N_y-i)}Bu(k-N_y+i) + Fy(k), \quad (8)$$

where the state-space MVP gain is

$$F = A^{(N_y-1)}L, \quad (9)$$

in which L is the steady-state optimal gain of the Kalman filter state estimator for the system (1), given by:

$$\bar{x}(k+1) = (A - LC)\bar{x}(k) + Bu(k-d+1) + Ly(k). \quad (10)$$

L can be solved by iterating the estimator algebraic Riccati difference equation (Lewis et al., 2008):

$$S(k+1) = AS(k)A^T \tag{11}$$
$$- AS(k)C^T\left(CS(k)C^T + R_{KF}\right)^{-1}CS(k)A^T + Q_{KF}.$$

When $k \to \infty$ the steady-state error covariance matrix S_∞ is used to solve L, such that

$$L = AS_\infty C^T\left(CS_\infty C^T + R_{KF}\right)^{-1} \tag{12}$$

For the minimum variance case, the Kalman filter weighting matrices Q_{KF} and R_{KF} must cope with the stochastic case which is based on the covariance matrices of $w(k)$ and $v(k)$, respectively Q and R. The required weighting matrices are then defined as (Lewis et al., 2008):

$$\begin{aligned} Q_{KF} &= \Gamma Q \Gamma^T, \\ R_{KF} &= R. \end{aligned} \tag{13}$$

Remark: the generalized minimum variance control formulation is based on ARMAX polynomial models and the GMVSS algorithm was first developed to cope with an ARMAX canonical state-space realization. In the ARMAX approach, the Kalman gain $L = \Gamma$ (Silveira et al., 2016). The case shown in this present paper, based on a more generalized state-space realization, such as (1) and the Kalman filter weighting matrices shown in (13), the PMVC approach is adopted as described in Silveira et al. (2020).

The PMVC control law that minimizes (6) is given by

$$u(k) = \frac{y_r(k) - \left(CA^{N_y} - CFC\right)\bar{x}(k) - CFy(k)}{\sum_{i=1}^{N_y} CA^{(N_y-i)}Bq^{-(N_y-i)} + \lambda} \tag{14}$$

where q^{-1} is the backward shift operator.

2.1 System augmentation by integrator addition

For type-0 systems, such as the control problem to be covered in this paper, in order to achieve guaranteed asymptotic reference tracking the system in (1) must be augmented in the following manner:

$$\begin{bmatrix} y(k+1) \\ \Delta x(k+1) \end{bmatrix} = \begin{bmatrix} I & CA \\ 0 & A \end{bmatrix}\begin{bmatrix} y(k) \\ \Delta x(k) \end{bmatrix} + \begin{bmatrix} CB \\ B \end{bmatrix}\Delta u(k-d+1) + \begin{bmatrix} I & C\Gamma \\ 0 & \Gamma \end{bmatrix}\begin{bmatrix} z(k) \\ w(k) \end{bmatrix} \tag{15}$$

$$y_a(k) = y(k) = [I\ 0]\begin{bmatrix} y(k) \\ \Delta x(k) \end{bmatrix} \tag{16}$$

where $z(k) = v(k+1)$, $x_a(k) = [y(k)\ \Delta x(k)]^T$ is the augmented state vector, $y_a(k)$ is the output of the augmented system and $\Delta u(k)$ is the control increment, such that $\Delta = 1 - q^{-1}$. Then, the incremental control action is

$$u(k) = u(k-1) + \Delta u(k). \tag{17}$$

3. DRONE'S THRUST TO ALTITUDE SUBSYSTEM

A picture of the LACOS flying room is shown in Fig. 1, where small quadcopter drones can be safely tested solely for position hold experiments and where a Microsoft Kinect depth sensor, fixed to the ceiling, can be used as a 3-dimensional local positioning system. The UAV in use in this research is the Parrot's AR.Drone 2.0 quadcopter.

A detailed description of AR.Drone's functions can be found in Mac et al. (2018). For this present work and for the sake of compactness it is simply stated that the robust

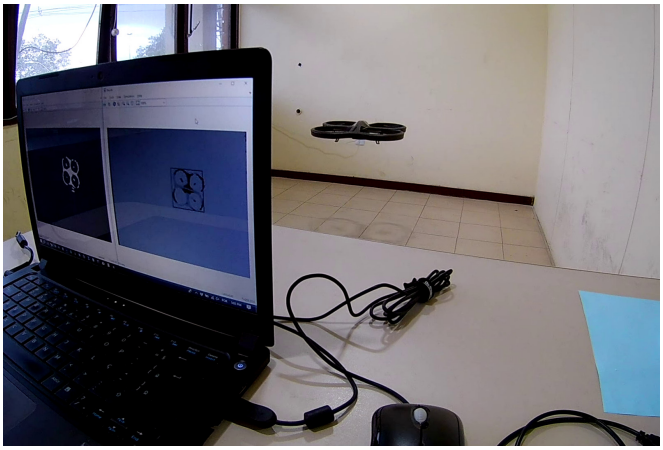

Figure 1. AR.Drone 2.0 flying at LACOS' LPS.

centralized multivariable linear quadratic Gaussian (LQG) speed hold autopilot published in Silveira et al. (2018) is used to stabilize longitudinal and lateral dynamics and then the vertical thrust to altitude subsystem is assumed to be decoupled from the horizontal velocities subsystems.

The input $u(k)$ is a vertical thrust command in the range of $[-1, 1]$ and the output $y(k)$ is the quadcopter's altitude measured in meters. Also, the middleware in use to link the UAV and a desktop PC is the free add-on toolbox *AR Drone Simulink Development-Kit V1.13* made by Sanabria and Mosterman (2014).

3.1 Altitude subsystem modeling in state-space

Adapting the transfer function based least-squares estimator algorithm to the state-space case, a non-recursive least squares estimator was used to identify the following second-order system parameters:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} \hat{a}_{11} & \hat{a}_{12} \\ \hat{a}_{21} & \hat{a}_{22} \end{bmatrix}\begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} + \begin{bmatrix} \hat{b}_{11} \\ \hat{b}_{21} \end{bmatrix}u(k-d) + \begin{bmatrix} \hat{\gamma}_{11} & \hat{\gamma}_{12} \\ \hat{\gamma}_{21} & \hat{\gamma}_{22} \end{bmatrix}\begin{bmatrix} w_1(k-1) \\ w_2(k-1) \end{bmatrix} \tag{18}$$

$$y(k) = [1\ 0]\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + v(k) \tag{19}$$

where $x_1(k)$ and $x_2(k)$ are respectively the altitude and vertical velocity.

The parameters vectors can be defined as

$$\begin{aligned} \hat{\theta}_1^T &= [\,\hat{a}_{11}\ \hat{a}_{12}\ \hat{b}_{11}\ \hat{\gamma}_{11}\ \hat{\gamma}_{12}\,] \\ \hat{\theta}_2^T &= [\,\hat{a}_{21}\ \hat{a}_{22}\ \hat{b}_{21}\ \hat{\gamma}_{21}\ \hat{\gamma}_{22}\,] \end{aligned} \tag{20}$$

and future observations are based on the following vector of regressors:

$$\phi_{lq}^T(k) \quad = [\,[x(k-1)]^T\ u(k-d)\ [w(k-1)]^T\,]. \tag{21}$$

The vector of regressors are organized to form the matrix of regressors,

$$\mathbf{\Phi}_{lq} = \begin{bmatrix} \phi_{lq}^T(k) \\ \vdots \\ \phi_{lq}^T(N) \end{bmatrix}_{N\times 5}, \tag{22}$$

leading to the estimation error covariance matrix,

$$\mathbf{P} = \left[\left(\mathbf{\Phi}_{lq}^{\mathbf{T}} \mathbf{\Phi}_{lq} \right)^{-1} \right]_{5\times 5}. \tag{23}$$

Considering X_1 and X_2 the measurements vectors of altitude and vertical velocity, respectively, the vectors of parameters can be computed in the following form:

$$\left[\hat{\theta}_1 \;\; \hat{\theta}_2 \right]_{5\times 2} = \mathbf{P} \mathbf{\Phi}_{lq}{}^{\mathbf{T}} \left[X_1 \;\; X_2 \right]_{N\times 2} \tag{24}$$

It must be remarked in (18) that in order to identify the state Gaussian process noise $\Gamma w(k)$, as well as the output process noise $v(k)$, the offline identification procedure must be made at least twice, in order to compute $\Gamma w(k)$ and $v(k)$ as the process identification error and estimated output error, respectively. In the first run of the algorithm these errors are computed and the model can be refined afterwards.

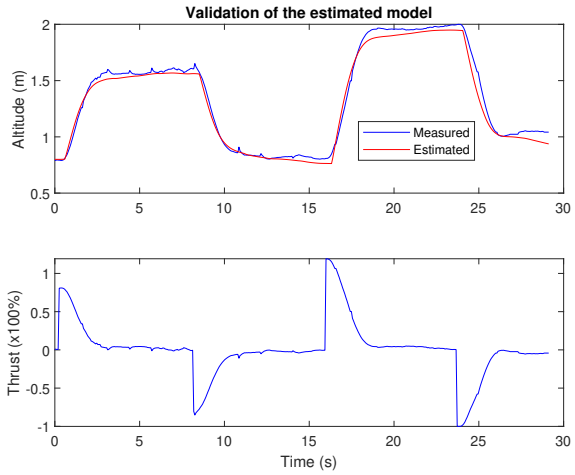

Figure 2. Vertical thrust to altitude model validation.

The thrust to altitude identified system model validation is shown in Fig. 2. The sampling time of 65 ms was used and the drone model normalized root mean squared error, NRMSE, was 87.4%. The experimentally observed time delay was approximately of 0.4 second. Thus, an approximated discrete time delay of $d = 6$ was included in the model. The estimated vectors of parameters, as in (20), are as follows:

$$\hat{\theta}_1^T = [\, 0.9995 \;\; 0.0119 \;\; 0.0370 \;\; 0.0014 \;\; 0.0012 \,]$$
$$\hat{\theta}_2^T = [\, -0.0080 \;\; 0.1823 \;\; 0.5699 \;\; 0.0214 \;\; 0.0182 \,]$$

The estimated variances for $w_1(k)$ and $w_2(k)$ were $\sigma_{w_1}^2 = 0.0016$ and $\sigma_{w_2}^2 = 0.0210$, respectively. And, for $v(k)$, $\sigma_v^2 = 0.0016$.

4. SIMULATION AND EXPERIMENTAL RESULTS

4.1 Defined UAV altitude hold requirements

According to Stevens et al. (2016), an altitude hold autopilot of a modern passenger aircraft typically hold the aircraft well within ±60.96 m (200 ft) and provide a warning signal if the deviation exceeds ±30.48 m (100 ft). In this sense, in order to have UAVs sharing the airspace in a near future, they must meet these standards. However, the vertical separation of passenger aircrafts are established in thousands of feet. For the laboratory scale, a 1 m separation is being considered. Then, the deviation requirement was scaled down and the autopilot must hold the UAV within ±0.2 m.

4.2 Designed control system and simulations

The AR Drone Simulink Development-Kit comes with a *baseline* altitude controller, ready to be used, thus assumed as a benchmark controller since it has established a common ground for researchers using the same UAV. This baseline controller is described by Sanabria and Mosterman (2014) as a Proportional controller, thus henceforth it is referred as P, while our designed predictive control system is referred as GMVSS.

Both in simulations and experimental applications with the real UAV, the sampling time $T_s = 65$ ms was used, since it is the default setting defined by the manufacturer.

The prediction horizon (or virtual time delay), $N_y = 25$, was selected. Thus, as of a 25 samples time spam using a $T_s = 65$ ms, the aimed final time optimization criteria was 1.625 s. However such steady-state optimization requirement was detuned by the selection of an increment control weighting factor $\lambda = 5$, which was fine tuned by trial and error while trying to avoid control signal saturation. These N_y and λ values were selected by following the guidelines shown in Silveira et al. (2020). Such trial and error tuning procedures are quite common among MPC practitioners, since the increase of N_y and λ generally gives a smooth closed-loop response.

The implemented control design was based on the augmented model. The steady-state optimal gain of the Kalman filter was obtained for the following estimated weighting matrices:

$$Q_{KF} = \Gamma_a \begin{bmatrix} \sigma_v^2 & 0 & 0 \\ 0 & \sigma_{w_1}^2 & 0 \\ 0 & 0 & \sigma_{w_2}^2 \end{bmatrix} \Gamma_a^T, \tag{25}$$
$$R_{KF} = \sigma_v^2.$$

where the a index in Γ_a denotes the augmented matrix. Then, the computed Kalman gain and MVP gain were as follows:

$$L = \begin{bmatrix} 0.6229 \\ 0.0030 \\ 0 \end{bmatrix}, \; F = \begin{bmatrix} 0.6946 \\ 0.0030 \\ 0 \end{bmatrix}. \tag{26}$$

For $N_y = 25$ a polynomial of degree 24 was computed for the denominator of the control law in (14), i.e. $R(q^{-1})$, leading to

$$R(q^{-1}) = r_0 q^0 + r_1 q^{-1} + \cdots + r_{24} q^{-24}. \tag{27}$$

Because of the great dimension of this polynomial, its values were omitted.

The main simulation result from the application of the designed GMVSS is shown in Fig. 3. It is possible to verify that, from the controlled altitude signals, the GMVSS and P controllers seem to cope with the requirements, since they respect the altitude hold limit of ±0.2 m in the presence of process and measurement noises. It must be remarked, however, that when it comes to the vertical thrust control signals, GMVSS is the one exhibiting a more conservative signal with less chattering.

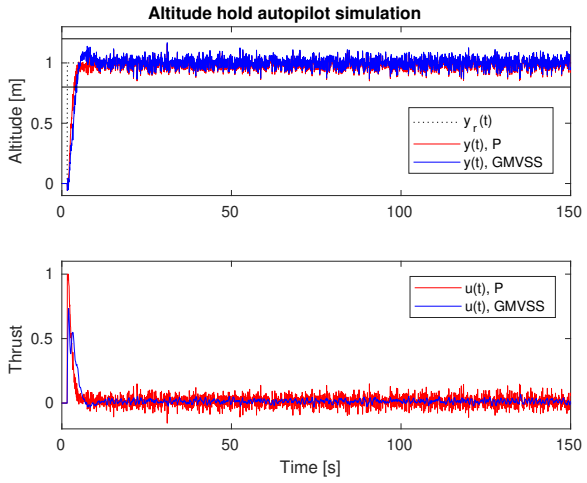

Figure 3. Altitude hold autopilot simulation.

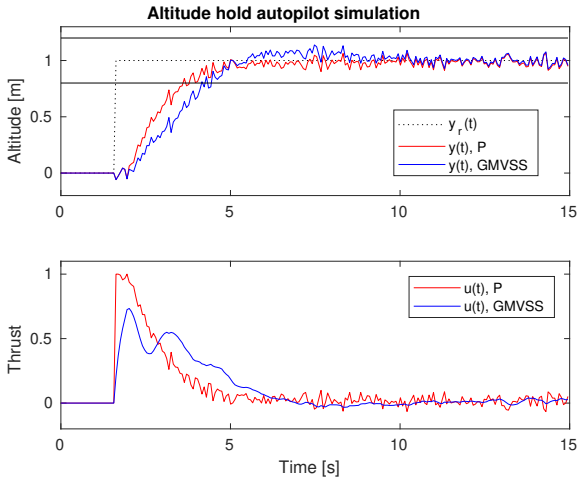

Figure 4. Altitude hold transitory simulation.

Benefits of long-range prediction in predictive minimum variance control have been recently addressed in Silveira et al. (2020). The linear power of the signals tend to decay with the long-range prediction and as a consequence a more efficient and economic control-loop can be achieved. This is numerically confirmed by the variances shown in Table 1.

Table 1. Simulations' performance indexes.

Ctrl. Type	P	GMVSS
$\sigma^2_{\Delta u(k)}$	0.00373	0.00009
$\sigma^2_{[y(k+N_y)-y_r(k)]}$	0.00364	0.00356
$\sigma^2_{\phi(k+N_y)}$	0.09449	0.00454

Between the performance indexes presented in Table 1, one that is remarkably better is the variance of $\Delta u(k)$ that can be associated to the power transitions required to drive and hold the system to the desired altitude. At the same time, the power of the error, associated to $\sigma^2_{[y(k+N_y)-y_r(k)]}$, is similar for both P and GMVSS, confirming the increased efficiency of predictive minimum variance control, despite its design complexity. The $\sigma^2_{\phi(k+N_y)}$ index is in fact an efficiency index, since it associates tracking error and control effort simultaneously and is directly related to the performance index that the GMVSS algorithm minimizes.

In Fig. 4 it is shown a zoomed view of the transitory behavior in the simulations with P and GMVSS controllers. It is apparent a small altitude overshoot with GMVSS but that do not exceed the upper limit and meets the autopilot's requirements. And the P controller, despite not very clear in this figure, can not guarantee asymptotic reference tracking as expected. However, this becomes clear in the experimental results shown next, since the plant-model-mismatch is of course true in real flights with the drone.

4.3 Experimental flight results

Real flight results for both P and GMVSS are shown in Fig. 5. Different altitude reference values were tested in order to experimentally evaluate the robustness and asymptotic tracking properties. GMVSS is constantly re-optimizing for 1.625 s in the future, accounting for delay and disturbances, whereas P is always 0.39 s late and every disturbance causes great altitude dispersion.

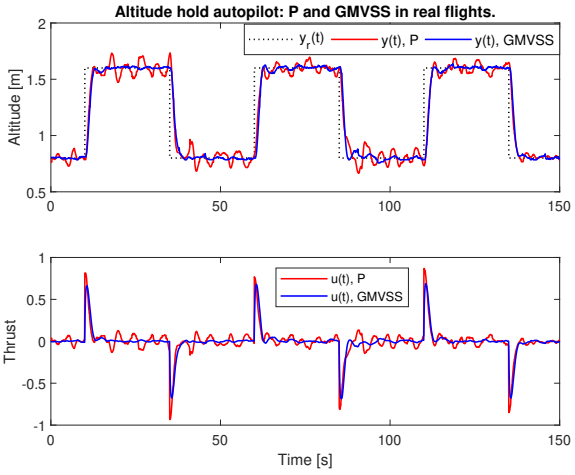

Figure 5. Altitude hold autopilot in real flights.

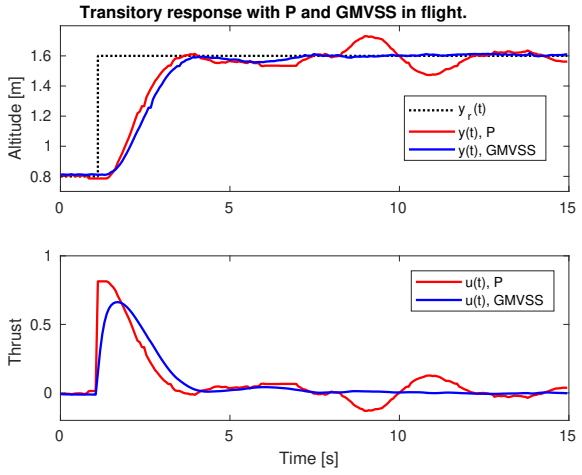

Figure 6. Altitude hold transitory in real flight.

Fig. 6 shows a zoomed view of the transitory behavior. The conducted flight tests confirmed what was being expected from theoretical results, where the GMVSS control-loop is

less sensible to disturbances and consequently holds the altitude more efficiently. This efficiency is also numerically confirmed by performance indexes shown in Table 2, computed based on the registered flight data.

Table 2. Performance indexes of real flights.

Ctrl. Type	P	GMVSS
$\sigma^2_{\Delta u(k)}$	0.00178	0.00024
$\sigma^2_{[y(k+N_y)-y_r(k)]}$	0.00876	0.00470
$\sigma^2_{\phi(k+N_y)}$	0.04945	0.00708

5. CONCLUSIONS

The designed GMVSS as a MPC presented efficient results in the task of controlling the altitude of the drone. Despite the controlled test environment, in terms of lack of atmospheric disturbances since flying indoors, the drone produced sufficient turbulent air flow around itself because it flies close to the floor and walls, so to generate a complex experimental system. Within this scenario, the controller under investigation could avoid altitude dispersion with significant control signal chattering reduction (cf. Fig. 5).

The complexity of the GMVSS controller in a non-adaptive case as the one investigated in this paper is similar to any other industrial controller, despite the increased order of its filters and systems that follow the adopted virtual delay N_y. However, when compared to more conventional MPC, such as Generalized Predictive Control or Dynamic Matrix Control, GMVSS solves the optimal control problem without the running cost of the so-called general Bolza problem, leading to a different form of tricked predictor or virtual MPC, while assuming that the controlled system has a delay bigger than the real one. In this sense, much effort is given to the steady-state or the end of mission optimization. The running cost introduction is still open for investigation.

The end of mission constraints are generally mandatory in guidance and control of UAVs, such as terminal flight conditions as in the landing procedure in which the final touchdown must be soft and with no further positional dispersion or the vehicle may crash into the floor. In this sense, the running cost is eventually considered due to the receding horizon control embedded in the GMVSS procedure, so that as time goes by the system is re-optimized to minimize the end of mission or $k + N_y$ problem ahead, thus conically converging to the end of mission requirement, despite, up to now, not assuming any constraints, which is another open issue.

For a future work, the idea is to extend this results with focus on tests with impulse noise, tone and complex interference, such as to simulate parasitic dynamics. Another open issue is including constraints. However, since GMVSS and predictive minimum variance control do not consider the running cost, different from most MPCs, the way to introduce constraints may not be the same, but may be feasible to achieve with dynamic programming techniques and the transcription procedure addressed in Sagliano (2018) and Sagliano et al. (2022), thus mixing within the GMVSS method both the end of mission and running cost requirements in a future work.

REFERENCES

Augugliaro, F., Lupashin, S., Hamer, M., Male, C., Hehn, M., Mueller, M.W., Willmann, J.S., Gramazio, F., Kohler, M., and D'Andrea, R. (2014). The flight assembled architecture installation: Cooperative construction with flying machines. *IEEE Control Systems Magazine*, 34(4), 46–64. doi:10.1109/MCS.2014.2320359.

Brescianini, D., Hehn, M., and D'Andrea, R. (2013). Quadrocopter pole acrobatics. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3472–3479. doi:10.1109/IROS.2013.6696851.

Hofer, M., Muehlebach, M., and D'Andrea, R. (2016). Application of an approximate model predictive control scheme on an unmanned aerial vehicle. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2952–2957. doi:10.1109/ICRA.2016.7487459.

Lewis, F.L., Xie, L., and Popa, D. (2008). *Optimal and Robust Estimation*. CRC Press. Taylor & Francis Group, 2nd edition.

Lupashin, S., Hehn, M., Mueller, M.W., Schoellig, A.P., Sherback, M., and D'Andrea, R. (2014). A platform for aerial robotics research and demonstration: The flying machine arena. *Mechatronics*, 24(1), 41 – 54.

Mac, T.T., Copot, C., Keyser, R.D., and Ionescu, C.M. (2018). The development of an autonomous navigation system with optimal control of an UAV in partly unknown indoor environment. *Mechatronics*, 49, 187 – 196.

Sagliano, M. (2018). Pseudospectral convex optimization for powered descent and landing. *Journal of Guidance, Control, and Dynamics*, 41(2), 320–334.

Sagliano, M., Seelbinder, D., Theil, S., Im, S., Lee, J., and Lee, K. (2022). Booster dispersion area management through aerodynamic guidance and control. In *AIAA SCITECH 2022 Forum*.

Sanabria, D.E. and Mosterman, P.J. (2014). ARDrone Simulink Development Kit v1.13. mathworks.com/matlabcentral/fileexchange. Accessed: February 22, 2020.

Schoellig, A.P., Siegel, H., Augugliaro, F., and D'Andrea, R. (2014). So you think you can dance? rhythmic flight performances with quadrocopters, in controls and art. In *Controls and Art*, 73–105. Springer International Publishing.

Silveira, A., Silva, A., Coelho, A., Real, J., and Silva, O. (2020). Design and real-time implementation of a wireless autopilot using multivariable predictive generalized minimum variance control in the state-space. *Aerosp Sci Technol*, 105.

Silveira, A., Trentini, R., Coelho, A., Kutzner, R., and Hofmann, L. (2016). Generalized minimum variance control under long-range prediction horizon setups. *ISA Transactions*, 62(1), 325 – 332.

Silveira, A.S., Silva, A.F., Real, J.A.F., and Silva, O.F. (2018). Centralized multivariable lqg control system for longitudinal and lateral speed hold autopilot for the AR.Drone 2.0 quadcopter. In *Proceedings of the XXII Brazilian Conference on Automatica*.

Stevens, B.L., Lewis, F.L., and Johnson, E.N. (2016). *Aircraft Control and Simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, Inc., 3rd edition.