Memorias del Congreso Nacional de Control Automático
ISSN: 2594-2492

Quadrotor Control Simulation with Multirate Integration Methods ⋆

Antonio Matus-Vargas * **Gustavo Rodriguez-Gomez** *
Jose Martinez-Carranza *,**

* *Instituto Nacional de Astrofísica, Óptica y Electrónica, Puebla, PUE, 72840 Mexico (e-mail: grrodrig,matusv@inaoep.mx).*
** *University of Bristol, Bristol, BS8 1UB UK (e-mail: carranza@inaoep.mx)*

Abstract: In this paper, we apply multirate integration methods for quadrotor simulation. Models described by ordinary differential equations can be separated by slow and fast dynamics that can be solved by numerical strategies tailored to such dynamics. Multirate methods take advantage of this fact to integrate the slow subsystem with a larger step than the fast one, which saves computational effort. We compare execution times and errors of the multirate strategies with the traditional Runge-Kutta method. Moreover, we deploy a multirate scheme in a control application and present simulation results.

Keywords: Quadrotor, Multirate Integration, Control Application

1. INTRODUCTION

Currently, quadcopters are used in a wide range of applications. Some of these applications require to continuously compute calculations involving the quadcopter's model to adjust some parameters, for instance, those parameters involved in the attitude control. In this case, the demand for computational resources increases significantly.

In general, the dynamic behavior of the Ordinary Differential Equations (ODEs) of the quadrotor can be decomposed in a small set of fast components and a set of slow components. Traditional numerical techniques fail to efficiently approximate the solution since the fast dynamics restricts the integration step for the slow components. Using special integration strategies, such as multirate integration methods, allows to associate a numeric method and a different integration step to each dynamics of the quadrotor; the fast components are associated with the smallest integration step. Execution time saving is achieved because the greater number of equations contained in the slow dynamics are integrated fewer times than the fast system.

Multirate integration methods are widely used in real-time simulation (Gear, 1981), modular dynamic system simulation (Rükgauer and Schiehlen, 1998), and electrical network simulation (Günther and Rentrop, 1993). To the authors' knowledge, this type of scheme has not been applied in quadcopter simulation. In this work, we focus on applying this special numerical technique to the model of the quadrotor to attain highly efficient numerical simulations.

We explore the dynamic behavior of the quadrotor to propose a partition of the fast and slow variables. The partition considers four fast state variables and eight slow state variables. In this research, a multirate strategy is applied to the nonlinear model of the quadrotor, which is described by twelve ODEs.

The applied multirate strategy consists of integrating the fast and slow dynamics with the explicit Euler method. We employ three techniques for implementing the multirate strategy: the first interpolates the state variables from the slow to the fast system, the second and the third use forward and backward information, respectively.

Through a set of simulations, we compare the efficiency of the multirate technique with a traditional implementation of the fourth-order Runge-Kutta. The saving in execution time is up to 82% with respect to the traditional strategy. Furthermore, we deploy the multirate model in a control application and evaluate the system in a realistic simulation.

The structure of the article is the following: Section 2 presents the literature considering the application of multirate integration methods. The quadrotor model is described in Section 3. In Section 4, we explore the dynamics of the quadrotor and propose a partitioning scheme. We evaluate the multirate methods in Section 5. Section 6 describes the application and presents the results. Lastly, conclusions are given in Section 7.

⋆ This work was supported by CONACyT.

2. RELATED WORK

Application areas of multirate integration methods include electrical network simulation (Bartel and Günther, 2002), robotic systems (Esposito and Kumar, 2001), and multibody systems (Arnold, 2007). Besides, these integration strategies are widely used in the real-time simulation of full scope power plants and flight simulators.

Mauritis et al. (1998) adopted multirate integration methods to approximate the solution of the convection-dominated flow problem. In that work, some theoretical results about the stability and accuracy of these type of methods are proved.

Multirate integration has been used in circuit simulations (van Eijndhoven et al., 1990). They partitioned the system equations in one large sparse set of network equations and many small sets of component equations. The integration time step of the fast dynamic is assigned independently and individually for each component. Also, they use piece-wise linear relation for all nonlinear equations.

Thiele et al. (2014) implemented in the environment Modelica a strategy that allows the modeler to use multirate integration methods. They used interpolation polynomial extrapolation to couple the slow dynamics with the fast dynamics.

3. QUADROTOR MODEL

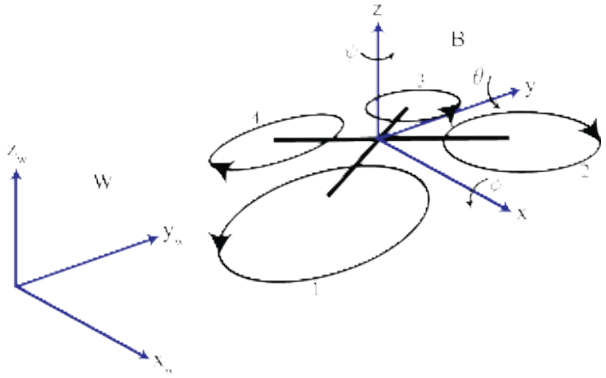

Fig. 1. Coordinate frames definition.

The mathematical model of the quadrotor can be deduced by introducing a world-fixed coordinate system $\{W\}$ and a body-fixed coordinate system $\{B\}$, see Fig. 1. The nonlinear model of the quadrotor can be obtained using the Newton-Euler equations or the Euler-Lagrange formalism (Bouabdallah and Siegwart, 2007). We have chosen a model that describes the position and orientation of the body frame with respect to the world frame. This model consists of the following 12 ODEs:

$$
\begin{aligned}
\dot{x} &= v_x \\
\dot{y} &= v_y \\
\dot{z} &= v_z \\
\dot{v}_x &= u_1(\sin\psi\sin\phi + \cos\psi\sin\theta\cos\phi)/m - c_d v_x \\
\dot{v}_y &= u_1(\sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi)/m - c_d v_y \\
\dot{v}_z &= u_1\cos\theta\cos\phi/m - g - c_d v_z \\
\dot{\phi} &= p + r\cos\phi\tan\theta + q\sin\phi\tan\theta \\
\dot{\theta} &= q\cos\phi - r\sin\phi \\
\dot{\psi} &= (r\cos\phi + q\sin\phi)/\cos\theta \\
\dot{p} &= [u_2 + qr(I_{yy} - I_{zz})]/I_{xx} \\
\dot{q} &= [u_3 + pr(I_{zz} - I_{xx})]/I_{yy} \\
\dot{r} &= [u_4 + pq(I_{xx} - I_{yy})]/I_{zz}
\end{aligned}
\quad , \qquad (1)
$$

where $s = [x, y, z]^T$ is the position vector from the origin of the body frame to the origin of the world frame, $[v_x, v_y, v_z]^T$ is the velocity of the vehicle in the world frame, $\eta = [\phi, \theta, \psi]^T$ is the vector of Euler angles (roll, pitch, and yaw), $\omega = [p, q, r]^T$ is the angular velocity in the body frame, m is the mass of the vehicle, $I = \text{diag}(I_{xx}, I_{yy}, I_{zz})$ is the inertia matrix, c_d is the normalized drag coefficient, g is the gravity acceleration, and $[u_1, u_2, u_3, u_4]^T$ is the control input vector. The inclusion of the drag coefficient is justified for high speed and even moderate flight regimes (Svacha et al., 2017).

The values of the model parameters used in this work are presented in Table 1. These values approximately correspond to the parameters of a Parrot Bebop 2 quadcopter. Throughout Sections 4 and 5, it is assumed that control inputs come from a stable cascade control scheme of the form presented in Matus-Vargas et al. (2019b), for which parameters are not shown for the sake of brevity.

Table 1. Model parameter values

Parameter	Value
m	0.4875 kg
I_{xx}	0.00223 kg m^2
I_{yy}	0.00299 kg m^2
I_{zz}	0.00480 kg m^2
c_d	0.1 s^{-1}
g	9.81 m s^{-2}

4. QUADROTOR MODEL PARTITIONING

In this section, we shall provide a discussion of the quadrotor model that will aid in the design of the partitioning scheme for the multirate integration strategy.

4.1 Heuristic Analysis

In the design of control systems for MAVs, a common assumption is the time-scale separation between the position and attitude dynamics, which correspond to the slow and fast time-scale of the vehicle model (Bertrand et al., 2008). If we further separate the attitude dynamics, we can naturally claim that the angular velocity vector has

the fastest time-scale of the model. Rating the fastness of the components of this vector can be done by looking at the values of the moment of inertia about each axis. The bigger the moment of inertia, the slower the dynamics. Consequently, sorting the components from the fastest to the slowest, we get q, p, and r.

The partition could be formed with only p and q. Nevertheless, research has shown that there exists a connection between the coupling of the slow and fast subsystems and the stability of multirate methods. In particular, the multirate stability region collapses when the coupling increases (Rodriguez-Gomez et al., 2004). Taking this information into account and noting that v_z remains close to zero in normal operation, we also consider as fast the variables v_x and v_y.

Variables v_x and v_y are expressed in the world coordinate system, whereas p and q in the body-fixed coordinate system. Intuitively, this means that r couples the aforementioned variables. We could include r as in the fast subsystem, but we would reduce the saving in execution time. Instead, we opt for taking v_x, v_y, q, and r as the fast subsystem. This arrangement provides a good trade-off between the execution time saving and coupling reduction.

To try to confirm that such a partition might be possible, we can check the eigenvalues of the original system across the solution trajectory. In Fig. 2, we have plotted the eigenvalues of the system for six different times across the solution. There is a gap of approximately five units along the real axis between the most negative eigenvalues and the ones that concentrate near the imaginary axis. This provides evidence that the original system possesses a fast time-scale and a few slow time-scales.

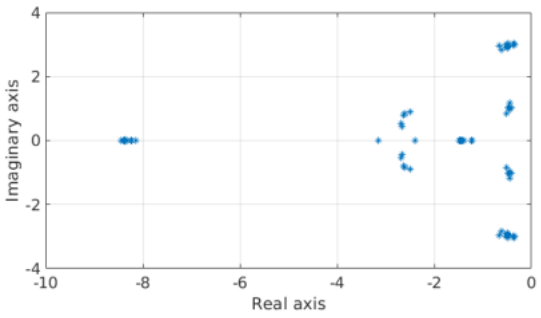

Fig. 2. Eigenvalues of the original system for different times across the solution.

5. MULTIRATE EVALUATION

We are interested in evaluating the performance of the multirate method applied to the quadrotor model. For that, we implemented the multirate integration method in C++. Also, we utilized a C++ version of the fixed-step fourth-order Runge-Kutta (RK4) as the baseline. The integration routines were executed on a laptop that is equipped with an Intel Core i7-4710HQ processor and a DDR3-1600 memory module.

We chose to evaluate two main indicators, the error and the execution time. The latter was estimated with the chrono library. The former was computed using the relative error formula taking as the "true" values the results of the RK4 method. The outcomes for the time and the error were obtained from separate tests since the output stream was disabled for the time tests.

The baseline (RK4) trajectories were obtained using a time step of 0.001 s. For the multirate variant, we set the multirate factor to the lowest possible value, which is two, and search for the largest time step in the set $\{0.10, 0.05, 0.04, 0.02, 0.01\}$ such that the maximum relative error of the position trajectories is below 10%, which is a reasonable selection for most indoor flights. We use this setup as a starting point to collect a total of four sets of results which are obtained by doubling the multirate factor and halving the time step. The same initial and final times are used for all integration procedures, $t_i = 0$ and $t_f = 5$ s. For the execution times, we perform 10 runs keeping the same setup and then obtain the average.

In summary, we are going to evaluate the multirate partition of the model with three numerical strategies: interpolation, forward information, and backward information. For each numerical strategy, four tests are going to be reported, totaling 12 cases.

Numeric results for our partition appear in Table 2. The partition was created by identifying the fast variables and their coupling with other variables. Here, the fast subsystem is formed with four variables: v_x, v_y, q, and r; the remaining variables are assigned to the slow subsystem.

Table 2. Results for the multirate partition.

#	Type	Step [s]	MR factor	Max. pos. rel. error	Avg. exec. time [μs]
1	Interp.	0.04	2	9.47%	316.9
2		0.04	4	19.92%	528.5
3		0.02	2	8.33%	600.5
4		0.02	4	7.65%	1016.5
5	Forw.	0.10	2	15.97%	147.1
6		0.10	4	8.73%	215.5
7		0.05	2	6.89%	251.9
8		0.05	4	4.07%	376.5
9	Backw.	0.04	2	16.71%	290.3
10		0.04	4	13.23%	460.6
11		0.02	2	6.64%	561.2
12		0.02	4	5.39%	930.7

From the table, it can be seen that, excepting the first interpolation pair, doubling the multirate factor decreases the error while halving the time step reduces the error for all strategies. In the table, the best results are achieved with the forward information strategy since errors under 10% are achieved with smaller average execution times compared to the rest. The best configuration for this partition is deemed to be the seventh test. This configu-

ration offers the best trade-off between error magnitude and average execution time.

To provide a qualitative comparison between the baseline and the multirate solutions, we will plot the variable with the greatest error resulting from the multirate solution and the same variable from the baseline solution. To find such variable, we first obtain the dimensionless with the min-max normalization method for both integration methods. Then, we compute the Mean Absolute Error (MAE) of each state variable. Finally, the variable to be plotted is identified as the one with maximum MAE.

We found that the variable with maximum MAE is the third component of the angular velocity, r. For reference, the value of the MAE was 0.0108 rad/s. The trajectories of this variable resulting from the baseline and the best multirate configuration are shown in Fig. 3. Good qualitative behavior is observed.

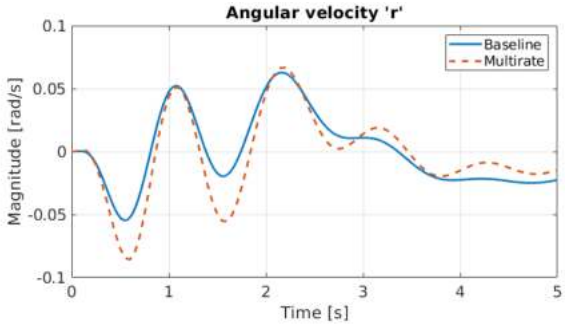

Fig. 3. Solution trajectories of the angular velocity r; fourth order Runge-Kutta method (solid), and multirate method (dashed).

For comparison purposes, we have collected execution times of the baseline and the multirate approach under similar conditions. Recalling, the time step of the baseline is 0.001 s, so we set the fast time step of the multirate method to that same duration. Similar to the stages of the RK4 method, we selected the multirate factor to be equals to four. The initial and final times were the same as in previous tests. A total of twenty tests were performed for each integration method. The results are summarized in Fig. 4. The multirate method generates an improvement of about 82% in execution time while maintaining the relative error of the position under 5%.

6. APPLICATION

Given that we have found that the multirate method with forwards information is the most suitable for our quadrotor mathematical model, we deployed this numerical integration scheme in a control application.

6.1 Implementation Details

We are particularly interested in developing a position controller for a quadcopter. This controller runs as the

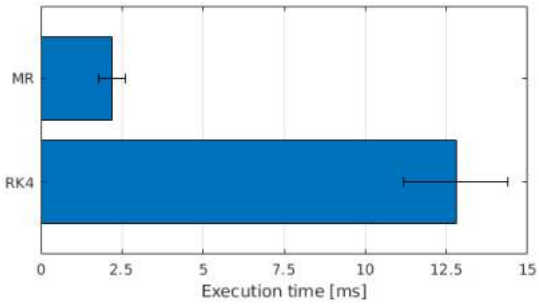

Fig. 4. Arithmetic mean with error bars of the execution times for the fourth order Runge-Kutta method (RK4), the multirate method (MR).

outer loop, while a low-level attitude controller runs as the inner loop. We consider that the vehicle is equipped with an onboard monocular camera. In addition to the RGB image, we can construct a synthetic depth image by assuming that the ground is flat and knowing the height and angle of the camera (Rojas-Perez and Martinez-Carranza, 2015). With this information, we are in a position to use the RGB-D version of the ORB-SLAM2 algorithm to estimate the pose of the camera, which will be the feedback of the position controller.

The SLAM system produces an estimate of the pose, in essence, position and orientation. However, the state space of the model contains also linear and angular velocities. We estimate these velocities by applying the first-order divided difference formula. Specifically, linear velocities are computed from measurements of the position, and angular velocities are obtained from measurements of the Euler angles assuming that $\omega \approx \dot{\eta}$, which is valid for small-angle incursions.

The process of constructing the synthetic depth image after the reception of the RGB image causes a drop in the output frequency of the SLAM system. A drop in measurement frequency would limit the effectiveness of the controller. To overcome this issue, we will make use of the multirate model to make predictions of the vehicle's state and use them as feedback. We will also take advantage of the model to predict the current state with the model given a delayed measurement. The particular objective is to run the outer loop at a higher frequency than the SLAM system.

To use the multirate model, we need to determine the model integration time for state prediction when we get a measurement, Δt_m. After we predict the state from the measurement, the model will be integrated afterward at fixed steps with the same duration of the control loop period to fill the gaps until we get a new measurement, we will refer to it as Δt_c. The other two parameters are the integration time step to reach Δt_m and Δt_c, which will be referred to as δt_m and δt_c, respectively. The parameter Δt_c is easily determined since it must be equal to the control loop period.

For the control of position, let us define u_x, u_y, u_z, and u_ϕ as the global control signals for the x, y, z, and ϕ state variables, respectively. Each control signal is a function of the error defined generically as $e_* = \rho - \gamma$, where ρ is the reference signal and γ is the measurement. In order to preserve the direction in global coordinates of the x-y projection of the vector generated by the position control signals, the transformation in (2) must be applied.

$$\begin{aligned} F &= \sqrt{u_x^2 + u_y^2} \\ u_{x,b} &= F\cos\left[\arctan\left(\frac{u_y}{u_x}\right) - \psi\right] \\ u_{y,b} &= F\sin\left[\arctan\left(\frac{u_y}{u_x}\right) - \psi\right] \end{aligned} \tag{2}$$

In addition, we consider that the previous outputs are related to the references of the inner loop as follows:

$$\begin{aligned} \phi_r &= u_{y,b}/\alpha_{max} \quad \theta_r = u_{y,b}/\alpha_{max} \\ v_{z,r} &= u_z/v_{z,max} \quad \dot{\psi}_r = u_\psi/\omega_{max} \end{aligned}, \tag{3}$$

where α_{max} is the maximum tilt angle, $v_{z,max}$ is the maximum vertical speed, and ω_{max} is the maximum rotation speed. In all tests, these parameters are set to 0.349 rad, 1 m/s, and 1.745 rad/s, respectively. The left hand side of (3) represent the final commands that are sent to the vehicle. In practice, the combination of (2) and (3) has been proven effective for waypoint tracking (Matus-Vargas et al., 2019a).

Finally, we estimate the input of the simulation model in the following form:

$$\begin{aligned} u_1 &= m(g + k_{pred}\dot{v}_{z,r}) \quad u_2 = I_{xx}k_{pred}\ddot{\theta}_r \\ u_3 &= I_{yy}k_{pred}\ddot{\phi}_r \quad\quad\quad u_4 = I_{zz}k_{pred}\ddot{\psi}_r \end{aligned}, \tag{4}$$

in which k_{pred} is the feedback prediction factor, which indicates to what extent the desired state will be achieved during Δt_c. We fixed the value of k_{pred} to 0.2 for all tests; we found this value empirically to deal with overshoot. Derivatives in (4) are estimated by applying the first-order divided difference formula along with the signals in (3).

6.2 Results

We evaluate the performance of our application in a realistic simulation. For this purpose, we employed the Gazebo robot simulator. Additionally, we used an implementation of the AR.Drone 2.0 [1] . This implementation allows us to access to the video stream of the down-looking camera and the ultrasonic sensor measurements. In the 3D environment, we load the drone and add texture to the ground for the ORB-SLAM2 algorithm. In the simulation, the pose measurement produced by the modified ORB-SLAM2 is reduced to ∼15 Hz. Our objective is to run the position control loop at 40 Hz, which means that $\Delta t_c = 1/40 = 0.025$ s. For simplicity, we consider that $\delta t_m = \delta t_c = 0.005$ s. Since we are trying to evaluate our application as close as possible to reality, we do not use the pose measurements provided by Gazebo.

In order to test for any performance enhancement, we shall compare the control performance without and with the multirate model. To minimize the control influence, we assign proportional controllers to u_x, u_y, u_z, and u_ϕ, so that $u_* = k_* e_*$, and tune their gains to achieve good end-point tracking. Then, we further increase k_x until we noticed oscillations. For clearness, we focused on finding differences in performance without and with the model only on this axis. The final gains were assigned in the following way: $k_x = 0.8$, and $k_y = k_z = k_\phi = 0.1$.

The first set of tests was performed to find a suitable value for Δt_m. In Fig. 5, we show the error response of five simulations, where the "w/o model" legend refers to the response without using the multirate model. In this figure, it can be seen that the value of $\Delta t_m = 0.15$ s yields the best result, so we keep this value for additional tests.

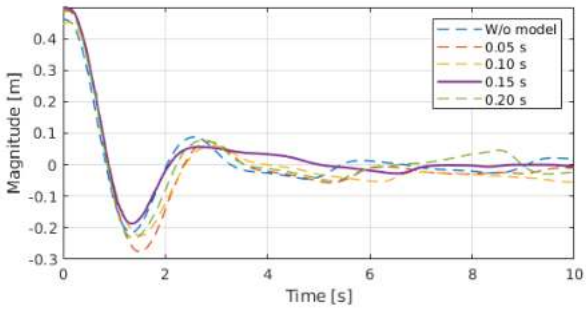

Fig. 5. Error responses of the x-axis for various values of Δt_m in simulation.

We proceed to evaluate formally any improvement on performance with the use of the multirate model. For that, we collect results from five simulations without and with the model keeping $\Delta t_m = 0.15$ s. Using the trajectory of the error, we compute both the Root Mean Square Error (RMSE) and the Integral Absolute Error (IAE). Results for these measures are summarized in Fig. 6. The plot shows that there is a significant RMSE reduction while using the model. Meanwhile, the reduction in the IAE is not clear. To ratify these results, we perform a t-test assuming unequal variances for each error measure. The alternative hypothesis is that the error measure using the model is significantly lower than the error measure without using the model, and the null hypothesis is that the error measure for both cases is equal. As expected, we found that with 99.99% of confidence we reject the null hypothesis and claim that the RMSE with the model is significantly lower than the RMSE without the model. Alternatively, we found that with 98.6% of confidence we

[1] `http://wiki.ros.org/tum_simulator`

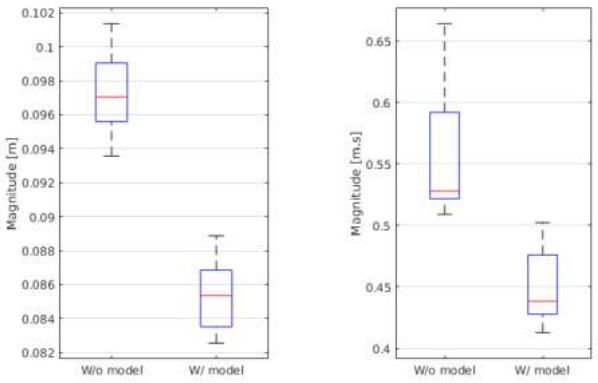

Fig. 6. Box plots of the error measures obtained from simulations.

reject the null hypothesis and claim that the IAE with the model is significantly lower than the IAE without the model. The use of the model improved the mean of the RMSE in 12.27% and the mean of the IAE in 19.35%.

7. CONCLUSIONS

In this paper, we presented an effective multirate integration method for quadrotor simulation. To find it, we proposed a partitioning scheme for the original model. Furthermore, we explored three strategies to approximate the values of the slow subsystem that are not in the mesh: interpolation, backward information, and forward information. In this exploration, we found evidence that the forward information strategy had larger stability region than the other two strategies. In comparison to the fourth-order Runge-Kutta method, the multirate method was able to execute 82% faster with relative errors of the position vector below 5%.

We deployed the multirate integration method with forwards information in a control application for quadrotors. We tested the application in a simulated environment and evaluated its performance in terms of the RMSE and the IAE. Our system was able to reduce the first error measure in 12.27% and the second in 19.35%.

Further research is needed. We plan to evaluate our system in more complicated missions, for example, tracking large references, waypoint following, and flying figures. Besides, the interaction with other than proportional action needs to be examined. Alternatively, the estimation of derivatives can be enhanced by applying proper filtering. Finally, our system needs to be evaluated in experiments with a real quadcopter platform.

REFERENCES

Arnold, M. (2007). Multi-rate time integration for large scale multibody system models. In *Proc. of the IUTAM Symposium*.

Bartel, A.. and Günther, M. (2002). A multirate W-method for electrical networks in statespace formulation. *J. of Computational and Applied Mathematics*, 147, 411–425.

Bertrand, S., Hamel, T., and Piet-Lahanier, H. (2008). Stability analysis of an uav controller using singular perturbation theory. In *Proc. of the 17th IFAC World Congress*.

Bouabdallah, S. and Siegwart, R. (2007). Full control of a quadrotor. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*.

Esposito, J.M. and Kumar, V. (2001). Efficient dynamic simulation of robotic systems with hierarchy. In *Proc. of the 2001 IEEE Int. Conf. on Robotics & Automation*.

Gear, G.W. (1981). Numerical solution of ordinary differential equations: is there anything left to do? *SIAM Review*, 23, 10–24.

Günther, M. and Rentrop, P. (1993). Multirate row methods and latency of electric circuits. *Applied Numerical Mathematics*, 13, 83–102.

Matus-Vargas, A., Rodriguez-Gomez, G., and Martinez-Carranza, J. (2019a). Aerodynamic disturbance rejection acting on a quadcopter near ground. In *2019 6th Int. Conf. on Control, Decision and Information Technologies*. IEEE.

Matus-Vargas, A., Rodriguez-Gomez, G., and Martinez-Carranza, J. (2019b). Parametric optimization for nonlinear quadcopter control using stochastic test signals. In *Studies in Computational Intelligence*, 55–79. Springer International Publishing.

Mauritis, N.M., van der Ven, H., and Veldman, A.E.P. (1998). Explicit multi-time stepping methods for convection-dominated flow problems. *Comput. Methods in Appl. Mech. and Engrg.*, 157, 133–150.

Rodriguez-Gomez, G., Gonzalez-Casanova, P., and Martinez-Carballido, J. (2004). Computing general companion matrices and stability regions of multirate methods. *Int. J. for Numerical Methods in Engineering*, 61, 255–273.

Rojas-Perez, L.O. and Martinez-Carranza, J. (2015). Metric SLAM and colour segmentation for multiple obstacle avoidance in autonomous flight. In *Workshop on Research, Education and Development of UAS*.

Rükgauer, A. and Schiehlen, W. (1998). Simulation of modular dynamic systems. *Mathematics and Computer in Simulation*, 46, 535–542.

Svacha, J., Mohta, K., and Kumar, V. (2017). Improving quadrotor trajectory tracking by compensating for aerodynamic effects. In *2017 Int. Conf. on Unmanned Aircraft Systems*. IEEE.

Thiele, B., Otter, M., and Mattson, S.E. (2014). Modular multi-rate and multi-method real-time simulation. In *Proc. of the 10th Int. Modelica Conf.*

van Eijndhoven, J.T.J., van Stiphout, M.T., and Buurman, H. (1990). Multirate integration in a direct simulation method. In *Proc. of the European Design Automation Conf.*

