Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 4651084, 6 pages
https://doi.org/10.1155/2022/4651084

Research Article

Mathematical Modeling of Real-Time Systems Using Heun and Piecewise Methods

Urfa Malik Gul,[1] Anand Paul ,[1] and K.-W.-A. Chee [2,3]

[1]The School of Computer Science and Engineering, Kyungpook National University, Daegu, Republic of Korea
[2]School of Electronic and Electrical Engineering, Kyungpook National University, Daegu 41566, Republic of Korea
[3]School of Electronics Engineering, College of IT Engineering, Kyungpook National University, Daegu 41566, Republic of Korea

Correspondence should be addressed to K.-W.-A. Chee; kwac2@cantab.net

Received 28 January 2022; Revised 8 March 2022; Accepted 15 June 2022; Published 21 August 2022

Academic Editor: Ravi Samikannu

It is often said that mathematical modeling is an implementation of mathematics in real-world problems with the aim of better understanding them, so we can say that mathematical modeling is linked to the solution of problems. Some of the essential principles and procedures of mathematical modeling are discussed using formulas and equations. We investigate the stability and convergence characteristics and demonstrate the suitability of different mathematical methods in a set of numerical examples. The described methods in our paper are the best choices for the simulation of linear phenomena and are more efficient for use with high-order spatial discretization. We emphasized the importance of mathematical modeling technologies used in computational tools. Our study shows that these new methods are more stable with lower errors.

1. Introduction

Various numerical methods are used to obtain the initial value solution, and studies show great development on this topic in this century. Other factors also amount to the reason for the implementation of mathematical methods like electronic computers needing efficient numerical algorithms in mathematical modeling. Many complex methods that are driven by partial differential equations (PDE) evolve [1–3], which requires the search for the appropriate numerical solution to a system of ordinary differential equations (ODEs) in a wide range of situations [4–6]. The differential equations contain derivatives, either ordinary or partial derivatives, and along with an initial condition that determines the value of the unknown function at a certain point, make up the initial value problem (IVP). These equations are also used in semiconductor engineering, biology, chemistry, physics, and economics to solve various problems in research. There are several analytical approaches for determining the solution of differential equations [7, 8]. On the other hand, analytical methods cannot always solve complex differential equations. Therefore, it is important to make numerical approximations.

The advanced differential equation is solved by using numerical techniques for which computer programming is a very powerful tool. The well-known Runge–Kutta (RK) analysis is the most often used technique for integrating these technologies of ODEs in real time [7–10]. RK methods can be classified in a variety of ways, such as the structure of implicit or explicit, by convergence order, or the number of steps [1]. The complexity of the equation that must be solved determines the numerical stability of various numerical methods.

Figure 1 summarizes the flow of ordinary equations used to model real-time systems and the application of analytical and numerical methods to solve these equations. An example here is encapsulating the physical characteristics of a parachutist descending, where g is gravitational constant, m is mass, c is drag coefficient, and $\mathrm{d}v/\mathrm{d}t$ is an unknown function/differential equation (rate equation).

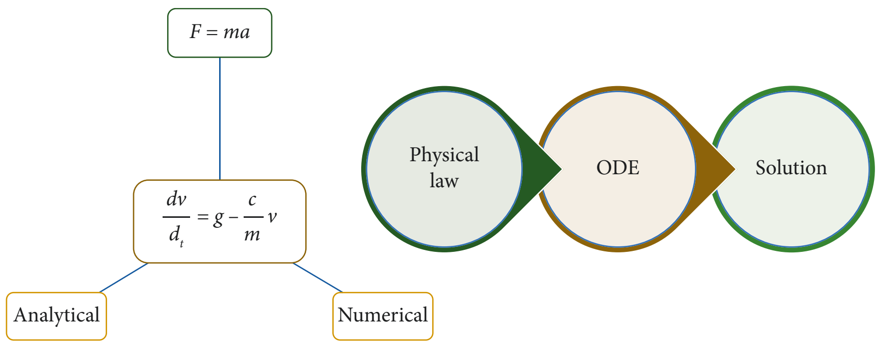

Figure 1: The summary of steps in the application of ODEs to engineering problem solving, such as a parachutist descending.

In the RK family, the most well-known member is the classic Runge–Kutta method (RK4), and (1) presents the initial problem as follows:

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y), y(t_0) = y_0. \tag{1}$$

In the above equation (1), y can be a scalar or vector function that is not known at time t that we want to estimate. $\mathrm{d}y/\mathrm{d}t$ is the time rate at which y changes and t_0 is the initial time; hence, the corresponding y value is y_0.

$$\begin{aligned} y_{n+1} &= y_n + \frac{1}{2}h(k_1 + 2k_2), \\ t_{n+1} &= t_n + h, \\ k_1 &= f(t_n, y_n), \\ k_2 &= f\left(t_n + \frac{h}{2}, y_n + h\frac{k_1}{2}\right), \\ k_2 &= f(t_n + h, y_n + hk_1). \end{aligned} \tag{2}$$

The k_1 and k_2 are known as stages of the RK method which are shown in (2) and (3). They relate to various estimations of solutions for the slope. We note here that $y_n + hk_1$ is the Euler step from (t_n, y_n) with stepsize h. Now, the following equation shows an explicit second-order RK method by supposing

$$\begin{aligned} y(t+h) &= y(t) + h\left[b_1\tilde{k} + b_2\tilde{k}_2\right] + O\left(h^3\right), \\ \tilde{k}_1 &= f(t, y), \\ \tilde{k}_2 &= f\left(t + c_2h, y + ha_{21}\tilde{k}_1\right). \end{aligned} \tag{3}$$

Therefore, the second-order RK supposition becomes

$$\begin{aligned} f\left(t + c_2h, y + ha_{21}\tilde{k}_1\right) &= f(t, y)c_2hf_t(t, y) + ha_{21}f_y(t, y)\tilde{k}_1 + O\left(h^2\right), \\ &\cdot f(t, y)c_2hf_t(t, y) + ha_{21}f_y(t, y)\tilde{k}_1 f(t, y) + O\left(h^2\right), \\ y(t+h) &= y(t) + h\left[b_1 f(t, y) + b_2\left\{f_t\left(t_, y\right) + c_2hf_t(t, y)ha_{21}f_y\left(t_, y\right)f(t, y)\right\}\right] + O\left(h^3\right), \\ &\cdot y(t) + (b_1 + b_2)hf\left(t_, y\right) + b_2h^2\left[c_2hf_t(t, y) + a_{21}f_y\left(t_, y\right)f(t, y)\right] + O\left(h^3\right). \end{aligned} \tag{4}$$

Therefore, for four unknowns, we have three nonlinear equations which lead to Euler's method (first-order method). Euler's method is commonly used to build more complicated approaches like the predictor/corrector method and is sometimes also called the midpoint rule as follows:

$$\begin{aligned} y_{n+1} &= y_n + hk_2, \\ k_1 &= f(t_n, y_n), \\ k_2 &= f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right). \end{aligned} \tag{5}$$

As we know, the RK methods are a set of implicit and explicit optimization methods in numerical analysis, including the well-known Euler's method, which is usually used in temporal discretization for approximate solutions of ODEs [1, 11]. Here, the differential equation is considered as a formula that can be used to compute the slope of the tangent line to the curve at any point once we determined the location of that point. The solution to the ODE at a time $t_n : y_n \approx y(t_n)$ is approximated by the value of y_n. The explicit Euler's method shows that the solution y_{n+1} is an explicit function of y_i for $i \leq n$. We must mention here that there is no assurance about any curves' concavity that will remain consistent. It may shift from concave-up to concave-down. Because the real solution curve goes up and down, there is no approximate way to estimate whether our next point will be an over- or underestimate. To overcome this deficiency, we have a clever Heun's method to use.

2. Heun's Method

Heun's method is the improved (or modified) Euler method and the second order of the RK method. The modified Euler method allows us to find a clear expression for y given a finite number of elementary functions, x. It is possible to solve the ODE for y and x given their initial values.

The Euler's method algorithm is as follows:

```
(1) Start
(2) Define function
(3) Determine $x0$, $y0$, $h$, and $xn$ (here $x0$ and $y0$ are initial conditions while $h$ is interval and $xn$ is required value)
(4) $n = (xn - x0/h + 1)$
(5) From $i = 1$ ton start looping
(6) $y = y0 + h \times f(x0, y0)$, $x = x + h$
(7) Print the $y0$ and $x0$ values
(8) If $x < xn$, assign $x0 = x$ and $y0 = y$
    If no, go to 9
(9) Loop $i$ end
(10) Stop
```

We can use the algorithm and flowchart in a high-level programming language to construct a program for Euler's method though this approach is not one of the best techniques to solve ODEs.

Heun's formula is used to solve the ODE which is given in the initial state [10, 12]. Euler's method is the foundation of Heun's method. As we discussed earlier, Euler's method fails to converge in the case of a concave-up curve because it overestimates the next point. Heun's method solves this solution problem by taking two tangents on each side of the curve. One tangent underestimates and the other overestimates, and then, Heun's method estimates the next point from both tangents using Euler's method. Using (6), we can determine the numerical solution to the problem of initial values using the following method:

$$\begin{aligned} y_{(i+1)} &= y_i + \frac{(k_1 + k_2)(h)}{2}, \\ k_1 &= f(x_i, y_i), \\ k_2 &= f(x_i + h, y_i + k_1 h). \end{aligned} \tag{6}$$

where k_1 and k_2 are under- and overestimates. The step size is denoted by h, and initial conditions are x_i and y_i. The accuracy of Euler's method improves linearly as step size decreases but the accuracy of Heun's method improves quadratically [10, 12–15].

3. Derivations

While using the elementary idea that the line's slope equals its rise over run, the parameters at the endpoint can be calculated using the following symbolic development:

$$\begin{aligned} \text{slope}_{\text{left}} &= f(x_i, y_i), \\ \text{slope}_{\text{right}} &= f(x_i + h, y_i + hf(x_i, y_i)), \\ \text{slope}_{\text{ideal}} &= \frac{\Delta y}{h}, \\ \text{slope}_{\text{ideal}} &= \frac{1}{2}\left(\text{slope}_{\text{left}} + \text{slope}_{\text{right}}\right), \\ \Delta y &= h\left(\text{slope}_{\text{ideal}}\right), \\ x_{i+1} &= x_i + h, y_{i+1} = y_i + \Delta y, \\ y_{i+1} &= y_i + h\text{slope}_{\text{ideal}}, \\ y_{i+1} &= y_i + \frac{1}{2h}\left(\text{slope}_{\text{left}} + \text{slope}_{\text{right}}\right), \\ y_{i+1} &= y_i + \frac{h}{2}\left(f(x_i, y_i) + f(x_i + h, y_i + hf(x_i, y_i))\right). \end{aligned} \tag{7}$$

Using Euler's method, the next point in the numerical solution is estimated roughly, and the initial estimation could be predicted or corrected using this information. Therefore, we can calculate the slope of the prediction line by taking the average of the slopes of the left and right tangent lines, at either end of the interval. The values of the $f(x, y)$ on the right-hand side of Equation (7) can be determined [16]. Following a rough estimate of the location of the next solution point from Euler's method, the coordinates may be used to determine the slope of the tangent line at the right end of the interval.

3.1. Modified Algorithm.

3.1. Modified Algorithm. The initial or boundary conditions must be given for both the algorithm and the flowchart, as follows:

```
(1) Start
(2) Define function for slope calculation $f(x, y)$
(3) State variables
(4) Input value
(5) Find slope while using initial value
(6) Find new $y$, $y = y0 + m0 \times h$
(7) Increase value of $x$, which is $x1 = x0 + h$
(8) Find new slope using $x1$ and $y$
(9) Calculate the mean of $m0$ and $m$
10) (Find $y$ new and assign $y$ new with $y$ again
(11) From step 6, repeat until two consecutive $y$ are equal
(12) Repeat step 5 until $x = xn$
(13) Print $x$ and $y$ corresponding
(14) Stop
```

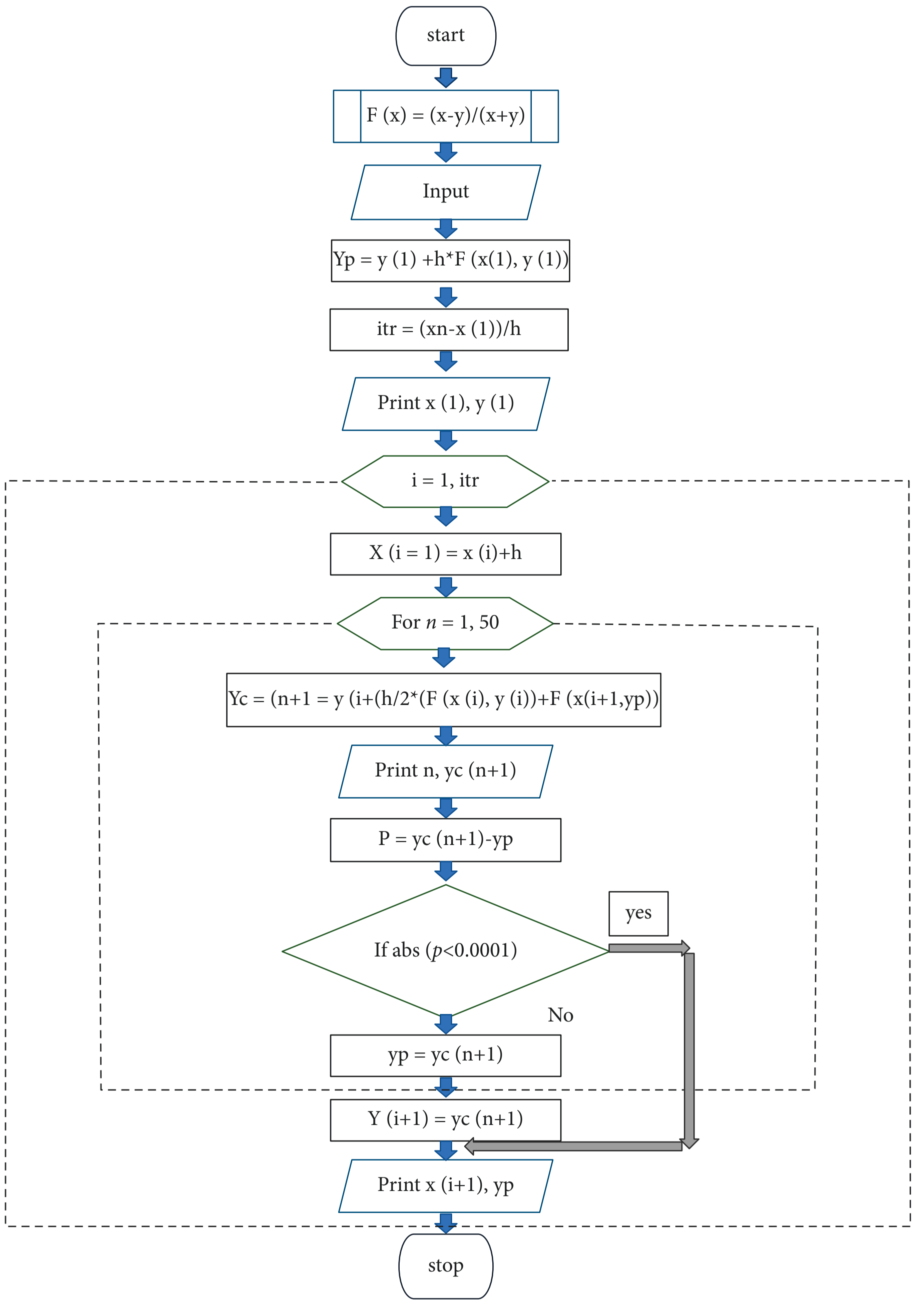

Figure 2: Modified Euler's method flowchart.

3.2. Flowchart. The numerical solution of ODEs with Heun's method is simple from both the numerical and programming points of view as compared to analytical techniques. The updated Euler's method algorithm and flowchart are described in Figure 2. They may be used to produce source code in any high-level programming language.

In the case of complex problems, analytical methods frequently fail; however, Heun's method does not fail but provides better solution accuracy and high speed. It also improves Euler's technique with a lengthy code where we can pick a lower number of h to get high accuracy.

4. Piecewise Method

Nonlinear programming is used to solve a variety of optimization issues in science and technology. Piecewise linearization methods are used in recent years to turn nonlinear programming into linear programming or a mixed convex programming problem to get the desired solution [16–19]. The piecewise function is a hybrid function that is defined by numerous subfunctions and each corresponding to a different interval of the main function domain or a subdomain. Piecewise can describe the nature of the function rather than just be a way of expressing it. In addition, the term piecewise can also refer to any piecewise-defined function that applies to each piece but may not be applicable for the whole function. The piecewise or continuous differentiable behavior of a function is that each piece is differentiable within its domain, regardless of whether the whole function is differentiable at the intersection. When functions are analyzed convexly, the notation of a derivative can be replaced by that of the subderivative [17]. A function can indeed be described as "piecewise linear" or "piecewise continuous" or "piecewise differentiable" even if the pieces are not intervals. Equation (8) shows a piecewise notation that expresses the absolute value function as follows:

$$|x| = \begin{cases} -x, & if\, x < 0, \\ +x, & if\, x \geq 0. \end{cases} \tag{8}$$

To turn negative values into positive, the first function -x is used for all x values, and this invalidates the sign of the input value. The second function x is applied for all x values which could be larger than or equal to zero.

4.1. Linear Piecewise. The piecewise linear function is a collection of quadratic functions, each defined on a set of intervals of real numbers. A finite collection of such intervals must exist if the domain of the function will either need to be finite or be locally finite [20–23].

There are several contexts in which piecewise linear functions are relevant. In general, the piecewise linear function can be defined on any vector space or affine space and simplicial complexes [18, 19, 24]. A linear function is not simply a linear transformation, but in these contexts, it is an affine linear function. There are two major subclasses of a piecewise linear function, called continuous and convex. As we can see from (9), every n-dimensional continuous function, in general, will have the following formulas:

$$\begin{aligned} & \%\Pi \int P\left(P\left(\mathbb{R}^{n+1}\right)\right), \\ fl_x = & \underset{\sum \int \%\Pi\ (a,b)}{\min} \quad \underset{\int \%\Sigma}{\max} \quad \vec{a} \cdot \vec{x} + b, \\ & \Sigma \int P\left(\mathbb{R}^{n+1}\right), \\ & f(\vec{x}) = \max \vec{a} \cdot \vec{x} + b \\ & (\vec{a}, b) \int \%\Sigma \qquad . \end{aligned} \tag{9}$$

5. Conclusion

We provided a class of numerical approaches for approximating the solutions of differential equations in this study which are based on a modified version. When applied to ODEs, these approaches yield a continuous approximate solution that is accurate to order $2n$ at the nodes and to order $n + 1$ evenly over the interval. To extend the approaches to delay differential equations, some a posteriori tweaks boost the uniform accuracy to order $2n$. Real-world issues are subjected to numerical testing and compared with other methodologies. Four methods have been presented for linear system integration of ODEs, and those methods require less memory and less computational effort than the low storage methods. The notations and formulations have been studied in such a way that subsequent development of the theory of differential equations will follow the structure of ODEs. We also presented that when a nonstandard piecewise function is used correctly, it produces fractional-order differential equations that converge to the original equation as the parameter leads to zero. Even though it was not developed from quadrature, the method has been demonstrated to be equivalent to quadrated-based methods. Nonuniform time steps can be used to accomplish the approach. The difference equation may appropriately describe the dynamics of the underlying fractional differential equation, as demonstrated in the formulas.

Data Availability

Data will be shared on request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was carried out with the support of the Dongil Cultural Scholarship Foundation for academic research in 2022, the Kyungpook National University Research Fund, 2021, the Kyungpook National University Excellent New Research Fund, and the Brain Korea (BK) 21 four project funded by the Ministry of Korea (4199990113966) and the National Research Foundation of Korea.

References

[1] J. A. D. Sandretto and A. Chapoutot, "Validated explicit and implicit Runge-Kutta methods. Reliable computing electronic edition," 2016, https://hal.archives-ouvertes.fr/hal-01243053/.

[2] N. C. Nguyen, ""A multiscale reduced-basis method for parametrized elliptic partial differential equations with multiple scales"," *Journal of Computational Physics*, vol. 227, no. 23, pp. 9807–9822, 2008.

[3] P. Colella, J. Bell, N. Keen, T. Ligocki, M. Lijewski, and B. V. Straalen, "Performance and scaling of locally-structured grid methods for partial differential equations," *J. Phys. Conf. Ser.*,vol. 78, no. 1, Article ID 012013, 2007.

[4] A. Slavík, "Dynamic equations on time scales and generalized ordinary differential equations," *Journal of Mathematical Analysis and Applications*, vol. 385, no. 1, pp. 534–550, 2012.

[5] O. Guner and A. Bekir, "A novel method for nonlinear fractional differential equations using symbolic computation," *Waves in Random and Complex Media*, vol. 27, no. 1, pp. 1–8, 2016.

[6] M. Rodríguez, F. Blesa, and R. Barrio, "OpenCL parallel integration of ordinary differential equations: applications in computational dynamics," *Computer Physics Communications*, vol. 192,pp. 228–236, Jul, 2015.

[7] H. Cao, L. Kang, Y. Chen, and J. Yu, "Evolutionary modeling of systems of ordinary differential equations with genetic programming," *Genetic Programming and Evolvable Machines*, vol. 1, no. 4, pp. 309–337, 2000.

[8] P. E. Kloeden and E. Platen, "Time discrete approximation of deterministic differential equations," *Numerical Solution of Stochastic Differential Equations*, pp. 277–303, 1992.

[9] H. Zhang, J. Yan, X. Qian, and S. Song, "Numerical analysis and applications of explicit high order maximum principle preserving integrating factor Runge-Kutta schemes for Allen-Cahn equation," *Applied Numerical Mathematics*, vol. 161, pp. 372–390, Mar. 2021.

[10] M. Molavi-Arabshahi and O. Nikan, "Solving some ordinary differential equations in mechanical engineering using Runge Kutta and Heun's methods," *Proc. 2nd Int. Conf. Comb. Cryptogr. Comput.*,vol. 7, no. 1, 2006.

[11] N. Ahmady, T. Allahviranloo, and E. Ahmady, "A modified Euler method for solving fuzzy differential equations under generalized differentiability," *Computational and Applied Mathematics*, vol. 39, no. 2, pp. 104–121, May 2020.

[12] W. Lay, S. Y. Slavyanov, T. Wiseman, W. Lay, and S. Y. Slavyanov, "The central two-point connection problem for the Heun class of ODEs," *Journal of Physics A: Mathematical and General*, vol. 31, no. 18, pp. 4249–4261, 1998.

[13] M. Hortaçsu, "Heun functions and some of their applications in physics," *Advances in High Energy Physics*, vol. 2018, Article ID 8621573, 14 pages, 2018.

[14] P. P. Fiziev, "The Heun functions as a modern powerful tool for research in different scientific domains," 2015, https://arxiv.org/abs/1512.04025v1.

[15] O. V. Motygin, "On numerical evaluation of the Heun functions," *2015 Days on Diffraction (DD)*, vol. 2015, Article ID 7354864, 227 pages, 2015.

[16] P. P. Fiziev, H. Zhong, M. T. Batchelor, and P. P. Fiziev, "Novel relations and new properties of confluent Heun's functions and their derivatives of arbitrary order," *Journal of Physics A: Mathematical and Theoretical*, vol. 43, no. 3, Article ID 035203, 2009.

[17] A. Magnani and S. P. Boyd, "Convex piecewise-linear fitting," *Optimization and Engineering*, vol. 10, no. 1, pp. 1–17, 2008.

[18] J. Pittman and C. A. Murthy, "Fitting optimal piecewise linear functions using genetic algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 7, pp. 701–718, 2000.

[19] M. H. Lin, J. G. Carlsson, D. Ge, J. Shi, and J. F. Tsai, "A review of piecewise linearization methods," *Mathematical Problems in Engineering*, vol. 2013, Article ID 101376, 8 pages, 2013.

[20] A. Atangana and S. İğret Araz, "New concept in calculus: piecewise differential and integral operators," *Chaos, Solitons & Fractals*, vol. 145, Article ID 110638, 2021.

[21] A. Rantzer and M. Johansson, "Piecewise linear quadratic optimal control," *IEEE Transactions on Automatic Control*, vol. 45, no. 4, pp. 629–637, 2000.

[22] R. Ambrosino, E. Garone, M. Ariola, and F. Amato, "Piecewise quadratic functions for finite-time stability analysis," in *Proceedings of the 2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pp. 6535–6540, Maui, HI, USA, December 2012.

[23] J. Puerto, A. M. Rodríguez-Chía, and A. Tamir, "On the planar piecewise quadratic 1-center problem," *Algorithmica*, vol. 57, no. 2, pp. 252–283, 2008.

[24] L. Rodrigues, J. P. How, and L. Rodrigues, "Observer-based control of piecewise-affine systems," *International Journal of Control*, vol. 76, no. 5, pp. 459–477, 2003.