A simulation-based study to calculate all the possible trajectories of differential drive mobile robot

Cite as: AIP Conference Proceedings **2333**, 070008 (2021); https://doi.org/10.1063/5.0041751
Published Online: 08 March 2021

Ahmed Ismaiel, and M. Yu. Filimonov

A Simulation-Based Study to Calculate All the Possible Trajectories of Differential Drive Mobile Robot

Ahmed Ismaiel[1, a)] and M. Yu. Filimonov[1, 2, b)]

[1)]*Ural Federal University, Yekaterinburg, Russia*

[2)]*Krasovskii Institute of Mathematics and Mechanics, Yekaterinburg, Russia*

[a)]Corresponding author: en.ismaiel@gmail.com

[b)]fmy@imm.uran.ru

Abstract. Deferential Drive Mobile Robot (DDMR) is being used in many applications as it is easy to be modeled and controlled. This research presents the idea of using DDMR turning motion behavior to develop an algorithm that calculate all the circular trajectories. This can be used to navigate DDMR in a curvature paths instead of linear ones. In this research we design and simulate Differential Drive Mobile Robot (DDMR) model. Then we use the simulated model to calculate all the possible trajectories that DDMR can follow with each left and right wheel velocity configurations. Results are saved in a navigation look-up table that can be implemented in DDMR navigation's approach.

INTRODUCTION

Differential Drive Mobile Robot (DDMR) is a ground wheel-based mobile robot with three degree of freedom and two actuators, that is why DDMR is described as a non-holonomic robot. DDMR consists of two independently driven wheels on the left and right of robot's chassis and one caster wheel in the front to balance the whole robot structure. Its motion behavior depends on the velocity configuration of each wheel. DDMR is commonly used in robotic research applications to test different navigation and obstacle avoidance algorithms due to its mobility and simple mechanical implementation [1]. DDMR proved its efficiency as a reliable mobile robot in many applications, such as floor cleaning mobile robots [2], inspection activities [3], transportation [4] and in many other applications. DDMR Navigation and control algorithm topics attract many researchers as these are a cornerstone in any autonomous robot application. Navigating DDMR through an environment with obstacle is demanding that DDMR follows specific trajectories that are collision-free and lead to the desired goal position and orientation. Most of navigation techniques to move DDMR from start point to goal point are using paths consist of straight lines and joints, where DDMR follows the straight line and turning only in the joint point like in [5] and [6]. There are other researches include DDMR curvature trajectory in DDMR control model as in [7]. While in this article two different algorithms are proposed to provide DDMR wheels' velocities configurations for given curvature trajectories. This research is a part of a PhD project to develop and control DDMR for smart city applications and to implement collision avoidance algorithm explained in [8].

RESEARCH OVERVIEW

This research aims to find DDMR possible circular trajectories instead of linear trajectories. As per DDMR kinematics, it follows circular trajectory if its driving wheels have different angular velocity values, and the radius of this trajectory can be calculated. In this article, we present the DDMR configuration and derive its kinematic model equations. We use this kinematic model to calculate all the possible circular trajectories that DDMR can follow with different wheels' angular velocities configurations. Trajectories' data is saved in a look up table. DDMR motion behavior and depicted trajectories are presented in Results section. Two different algorithms (A and B) are described to calculate right and left wheel angular velocity as a function of trajectory radius and linear velocity. We evaluate the performance of each algorithm and explain the Integration of both algorithms to navigate the DDMR. At the end of this paper, we conclude the research result.

Applications of Mathematics in Engineering and Economics (AMEE'20)
AIP Conf. Proc. 2333, 070008-1–070008-7; https://doi.org/10.1063/5.0041751
Published by AIP Publishing. 978-0-7354-4077-7/$30.00

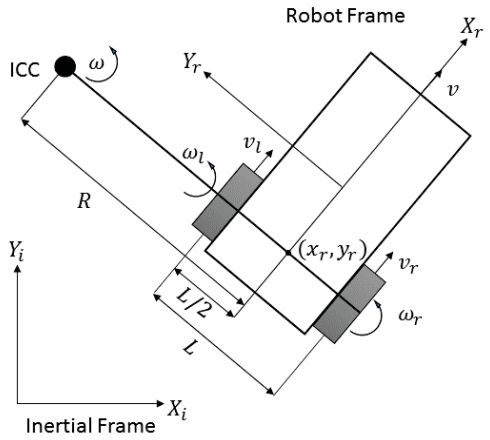


FIGURE 1. DDMR position and orientation.

DDMR KINEMATIC MODEL

Kinematic problem is the study of mobile robot motion and its mechanical system behavior [9]. It is important in order to configure robot's design and controller parameters. In this section, we derive the mathematics of DDMR motion without considering any forces that affect the motion. DDMR main controlling parameters are the left and right wheels' angular velocities and its configuration space which is located in two dimensional surface [10]. DDMR position and orientation are described by two different frames as in figure 1: Inertial frame X_i, Y_i and Robot frame X_r, Y_r. The first frame in the base fixed coordinates where DDMR position coordinates are (x_i, y_i) and DDMR orientation angle is θ_i. The second frame is a moving frame attached to the DDMR where DDMR position coordinates are (x_r, y_r) and DDMR orientation is θ_r. DDMR wheelbase is an axis connects center of rotation of both DDMR left and right wheel and has the length L. DDMR heading direction is the direction of X_r. Transformation matrix is used to convert between Inertial frame and Robot frame as following:

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\theta}_r \end{bmatrix}. \tag{1}$$

The following parameters are used to derive the kinematic model equations: ω_r and ω_l are the right and left wheels' angular velocities respectively. v_r and v_l are the right and left wheels' linear velocities respectively. v and ω are the DDMR linear and angular velocities respectively. Assume that DDMR wheels are rolling with different angular velocities $\omega_r \neq \omega_l$, each wheel contacts the ground at a single point, and each wheel rolling without any slipping on the ground. In this case, both wheels follow a circular trajectory with common center point called ICC (Instantaneous Center of Curvature) and DDMR moves with linear velocity v and angular velocity ω along the circular trajectory with radius R around ICC. R is the length of line connecting ICC point and the center point of wheelbase axis. We can get DDMR linear and angular velocities when it completes one full turn around ICC during time T:

$$v = 2\pi R/T, \tag{2}$$

$$\omega = 2\pi/T. \tag{3}$$

From equations (2) and (3), we can get DDMR linear velocity as a function of angular velocity

$$v = \omega R. \tag{4}$$

The linear velocities of each wheel can be calculated by associating the corresponding trajectory circular radius for each wheel in equation (4)

$$v_r = \omega(R + l/2), \tag{5}$$

$$v_l = \omega(R - l/2). \tag{6}$$

The linear velocity of each wheel can be represented by its angular velocity and wheel radius as following $v_r = \omega_r r$ and $v_l = \omega_l r$. From equation (5) and (6) we calculate the radius of the circular trajectory that DDMR will follow R as in equation (7). DDMR linear and angular velocities are represented by both wheels' angular velocities as in equations (9), while equation (8) is derived from equations (4), (5) and (6)

$$R = \frac{rL}{2}\frac{(\omega_r + \omega_l)}{(\omega_r - \omega_l)}, \tag{7}$$

$$v = \frac{r}{2}(\omega_r + \omega_l), \tag{8}$$

$$\omega = \frac{r}{L}(\omega_r - \omega_l). \tag{9}$$

Then we describe DDMR linear velocity v in robot frame $\dot{x_r}$ and $\dot{y_r}$ directions. As per our assumptions, wheels rolling with no lateral slip, linear velocity component in $\dot{y_r}$ equal to zero and angular velocity $\omega = \dot{\theta}_r$.

$$v = \dot{x_r} = \frac{r}{2}(\omega_r + \omega_l), \qquad \dot{y_r} = 0. \tag{10}$$

DDMR velocity components in robot frame can be shown in equation (11). substituting in equation (1) we get DDMR kinematic model equation that describes the velocity of DDMR as a function of it's right and left wheel angular velocities.

$$\begin{bmatrix} \dot{x_r} \\ \dot{y_r} \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ 0 & 0 \\ \frac{r}{2} & \frac{-r}{2} \end{bmatrix} \quad \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}, \tag{11}$$

$$\begin{bmatrix} \dot{x_i} \\ \dot{y_i} \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} \frac{r}{2}\cos\theta & \frac{r}{2}\cos\theta \\ \frac{r}{2}\sin\theta & \frac{-r}{2}\sin\theta \\ \frac{r}{L} & \frac{-r}{L} \end{bmatrix} \quad \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}. \tag{12}$$

we use equation (12) to programmatically simulate DDMR model and to get the position and trajectory that the model follow by calculating its new position $x_i(t+\delta_t)$, $y_i(t+\delta_t)$ and orientation $\theta_i(t+\delta_t)$ for each simulation iteration with time increment δ_t:

$$x_i(t+\delta_t) = x_i(t) + \dot{x_i},$$

$$y_i(t+\delta_t) = y_i(t) + \dot{y_i},$$

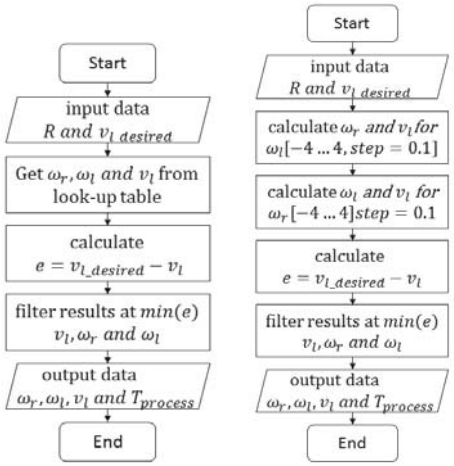
$$\theta_i(t+\delta_t) = \theta_i(t) + \dot{\theta}_i.$$

SIMULATION TOOLS

DDMR modeling and kinematic equations are simulated by using Processing programming based on Java. Processing provides researchers with both programming language and integrated development environment (IDE) built for the media arts communities [11]. DDMR all trajectories images are prepared and exported by using Processing. Data analysis, charts and algorithms are programmed by Python. Python is one of the simplest, high-level, general-purpose programming language with efficient libraries in the scope of data analysis and machine learning [12]. Algorithm are programmed with the use of "numpy" library [13]. However, explanation charts are depicted with the use of "matplotlib" library [14].

SIMULATION AND ALGORITHM

In the previous section, we derived DDMR kinematic equations. Now, we implement DDMR model with the kinematic equation. DDMR simulated model has the following design parameters: maximum wheel angular velocity $v_{max} = 4\ \ deg/sec$, minimum angular velocity step $\omega_{step} = 0.1$, wheelbase $L = 20\ \ cm$ and wheel radius $r = 2\ \ cm$. Each wheel has 81 angular velocity values in the range of $[-4...4, step = 0.1]$. By using equation (7), we get 6561



(a) Algorithm A (b) Algorithm B

FIGURE 2. Algorithm (A) and Algorithm (B) flowchart.

circular trajectories' radii for each left and right wheel angular velocity. The resulting 6561 radii values are stored in look up navigation table. DDMR navigation problem is to find the optimum left and right wheel angular velocities for the desired circular trajectory, here is two different algorithm A and B to find DDMR angular velocities configurations as a function of trajectory radius and linear velocity. Algorithm A depends on the navigation look up table where Algorithm B depends on mathematical calculations to get DDMR angular velocities configurations. Figure 2 shows the flowchart of both algorithms. the first step in both algorithms is setting the input variables. Input data are the desired trajectory radius R and desired DDMR linear velocity $v_{l_{desired}}$. Algorithm A gets the desired R and searching in the navigation look-up table to get the corresponding ω_r, ω_l and v_l. Algorithm B calculates the possible wheel angular velocities configuration using equation (13) derived from equation (7).

$$\omega_r = \omega_l \frac{R+k}{R-k}, \qquad k = \frac{rL}{2}. \tag{13}$$

All the values of ω_r and v_l are calculated for the desired radius and for each value of ω_l in the range of $[-4...4, step = 0.1]$. Then algorithm B repeats this calculation for ω_l with each value of ω_r in the range of $[-4...4, step = 0.1]$. The last three steps are the same for both algorithm A and B. In these steps, error values e between the desired linear velocity $v_{l_{desired}}$ and resulting linear velocity v_l for each wheels' angular velocities configurations are calculated. Then, the angular velocities configuration with the minimum error values are sent to the algorithms data as an output. The performance of each algorithm is evaluated by measuring the processing time T_p "The time taken by the computer to process the algorithm calculation and give the resulting angular velocities" and the error value of the resulting linear velocity and the current DDMR linear velocity. Algorithms programming codes are executed by Google Colaboratory that provides typical processing performance for both algorithms [15].

RESULTS

From equation (7) DDMR motion behaviors can be categorized to separate groups, for example when both wheels rotates around their axes in the same angular velocity, that results in positive or negative infinity trajectory radius,

TABLE 1. DDMR motion behaviour groups.

No	Motion type	wheels' velocity configuration conditions	
1	Stand Still	$v_l = v_r = 0$	
2	Straight Forward	$\|v_l\| = \|v_r\|$	$v_l, v_r > 0$
3	Straight Backward	$\|v_l\| = \|v_r\|$	$v_l, v_r < 0$
4	Turning Forward Right	$\|v_l\| > \|v_r\|$	$v_l > 0$
5	Turning Forward Left	$\|v_l\| < \|v_r\|$	$v_r > 0$
6	Turning Backward Right	$\|v_l\| > \|v_r\|$	$v_l < 0$
7	Turning Backward Left	$\|v_l\| < \|v_r\|$	$v_r < 0$
8	Turning In-place Right	$\|v_l\| = \|v_r\|$	$v_l > 0$, $v_r < 0$
9	Turning In-place Left	$\|v_l\| = \|v_r\|$	$v_l < 0$, $v_r > 0$

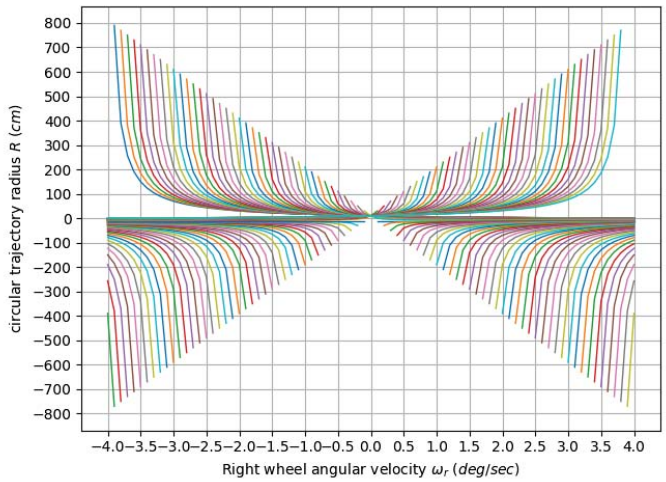


FIGURE 3. Trajectories radii for ω_r, $\omega_l = [-4...4]$ and $\omega_{step} = 0.1$.

in this case DDMR motion behavior is straight forward or backward. DDMR has nine motion behaviors rely on each wheel linear velocity configurations. Table I shows DDMR motion behavior groups. The result of all possible trajectories radii is shown in figure 3. This chart include 81 curves, each curve represents trajectories radii at one value of ω_l in the range of $[-4...4, step = 0.1]$. x-axis represents the values of ω_r, y-axis represents the values of resulting radius. The depicted radii are graphically shown in figure 6. One radius curve $\omega_l = 2$ and $\omega_r = [-4...4, step = 0.1]$ is shown in figure 4. This shows that the maximum value of trajectory radius can be 390.0 *cm* for turning left movement and 410.0 *cm* for turning right movement. From the chart in figure 3, for a single trajectory radius there can be different ω_r and ω_l configuration. These configurations are being used for DDMR navigation purposes in order to steer DDMR in a curvature trajectory with suitable wheels' angular velocities as we explained before. In our DDMR model radius values up to 790.0 *cm* can be achieved.

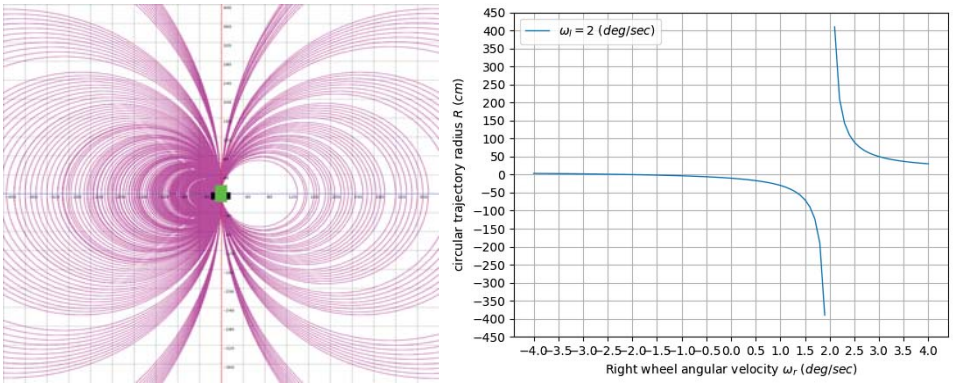
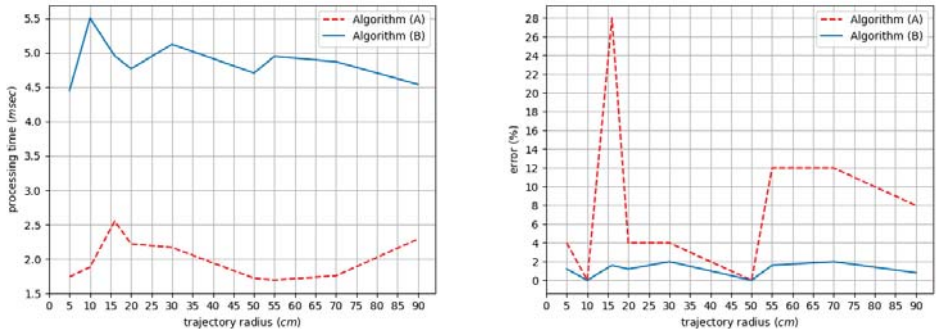


FIGURE 4. Trajectories' radii for $\omega_l = [-4...4]$, $\omega_r = 2$ and $\omega_{step} = 0.1$.

All the possible wheel's angular velocities configurations stored in navigation look-up table and implemented in algorithm A to be used for DDMR navigation applications. However, algorithm B make all the calculations during DDMR navigation to find the suitable wheel's angular velocities. Performance evaluation results of both algorithm A and B are shown in figure . In this evaluation, trajectory radius and linear velocity values are given as input to each algorithm, then figure (a) shows response time T for algorithm A and B. Figure (b) the percentage error between the desired DDMR linear velocity and the calculated one by each algorithm. It's obvious that algorithm A has better response time than algorithm B. However the later one has better accuracy in terms of the DDMR linear velocity v_l



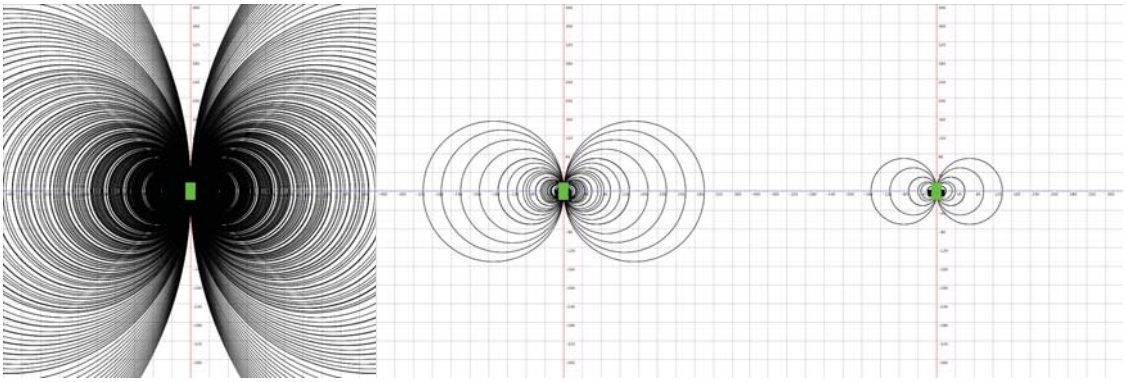
(a) Algorithms processing time

(b) Algorithms results accuracy

FIGURE 5. Algorithms (A) and (B) processing time and accuracy.

CONCLUSION

Using circular trajectories to navigate DDMR can be effective method for mobile robots application. as Shown in figure 6, DDMR can reach all points in its configuration space by configure DDMR wheels' angular velocities. Wheels' angular velocity resolution affects the quantity of available circular trajectories that can be followed by DDMR. Figure 6 shows the available trajectories for each wheel angular velocity resolution. More quantity and wider radius trajectories can be achieved with lower wheels' angular velocity resolution. Total quantity of trajectories N can be calculated as following $N = 2\ \frac{v_{max}}{\omega_{step}}$, there is an inverse proportion between trajectories quantity and angular velocity resolution.



(a) $\omega_{step} = 0.1$ (b) $\omega_{step} = 0.5$ (c) $\omega_{step} = 1.0$

FIGURE 6. Trajectories' radii at ω_r,$\omega_l = [-4...4]$ and $\omega_{step} = \{1.0, 0.5, 0.1\}$.

Regarding the searching algorithms A and B, both of them are sufficient to be used in DDMR navigation applications to find the optimum ω_r and ω_l. However, it is relatively depends on the application priorities. For applications that give more attention to trajectory accuracy over the time of processing, algorithm B is more suitable. But for other application that focus on the rapid response, its more suitable to go with algorithm A. Hybrid approach of using both algorithm A and B can also be implemented in the same DDMR to switch between both algorithms according to specific conditions.

REFERENCES

1. R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*. MIT press, 2011.
2. E. Prassler, A. Ritter, C. Schaeffer, and P. Fiorini, "A short history of cleaning robots," *Autonomous Robots*, vol. 9, no. 3, pp. 211–226, 2000.
3. S.-g. Roh and H. R. Choi, "Differential-drive in-pipe robot for moving inside urban gas pipelines," *IEEE transactions on robotics*, vol. 21, no. 1, pp. 1–17, 2005.
4. B. Stouten and A.-J. de Graaf, "Cooperative transportation of a large object-development of an industrial application," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 3, pp. 2450–2455, IEEE, 2004.
5. N. S. Utami, A. Jazidie, and R. E. A. Kadier, "Path planning for differential drive mobile robot to avoid static obstacles collision using modified crossover genetic algorithm," in *2019 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, pp. 282–287, IEEE, 2019.
6. J.-H. Li, Y.-S. Ho, and J.-J. Huang, "Line tracking with pixy cameras on a wheeled robot prototype," in *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, pp. 1–2, IEEE, 2018.
7. J. Singh and P. S. Chouhan, "A new approach for line following robot using radius of path curvature and differential drive kinematics," in *2017 6th International Conference on Computer Applications In Electrical Engineering-Recent Advances (CERA)*, pp. 497–502, IEEE, 2017.
8. I. Ahmed and M. Y. Filimonov, "Collision avoidance algorithm with performance optimization and speed control for multi-robot autonomous system," in *48th International Youth School Conference "Modern Problems in Mathematics and its Applications", SoProMat 2017*, vol. 1894 of *CEUR-WS Proceedings*, pp. 115–122, 2017.
9. S. K. Malu and J. Majumdar, "Kinematics, localization and control of differential drive mobile robot," *Global Journal of Research In Engineering*, 2014.
10. T. Hellström, *Kinematics equations for differential drive and articulated steering*. Department of Computing Science, Umeå University, 2011.
11. C. Reas and B. Fry, *Processing: a programming handbook for visual designers and artists*. Mit Press, 2007.
12. M. F. Sanner *et al.*, "Python: a programming language for software integration and development," *J Mol Graph Model*, vol. 17, no. 1, pp. 57–61, 1999.
13. T. E. Oliphant, *A guide to NumPy*, vol. 1. Trelgol Publishing USA, 2006.
14. J. Hunter and D. Dale, "The matplotlib user's guide," *Matplotlib 0.90. 0 user's guide*, 2007.
15. T. Carneiro, R. V. M. Da Nóbrega, T. Nepomuceno, G.-B. Bian, V. H. C. De Albuquerque, and P. P. Reboucas Filho, "Performance analysis of google colaboratory as a tool for accelerating deep learning applications," *IEEE Access*, vol. 6, pp. 61677–61685, 2018.