Applied Mathematics and Computational Intelligence
Volume 15, No. 2, 2026 [1-12]

Three-Term Conjugate Gradient Method Using Strong Wolfe Line Search for Robotic Motion

Dzul Dzaihan Dzul Dzailani[1], Norhaslinda Zullpakkal[2*], Hannah Telen Luyoh[3], Nurfatihah Anizan[4], Michelle Ubong Sari[5], Sulaiman Mohamed Ibrahim[6]

[1]Wisma Mercedes-Benz, 16A Jalan BK 1/13, Taman Perindustrian Bandar Kinrara, Puchong, Malaysia
[2,3,4,5]Universiti Teknologi MARA Cawangan Terengganu Kampus Kuala Terengganu, Terengganu, Malaysia
[6]School of Quantitative Sciences, Universiti Utara Malaysia, UUM Sintok, Kedah, Malaysia

[*] Corresponding author: lindazullpakkal@uitm.edu.my

Received: 6 May 2025
Revised: 19 August 2025
Accepted: 26 August 2025

ABSTRACT

Optimization involves finding the optimal solution of the objective function. One of the best optimization methods for solving unconstrained optimization problems is Conjugate gradient (CG) method. CG methods are classified into several categories, including scaled, three-term and hybrid CG methods. CG method is widely implemented in various applications such as robotic motion, image restoration and regression analysis. However, some of the CG methods are not applicable for solving real life problems. This research focuses on the performance of three term CG method (TTCG) under strong Wolfe line search and its applicability in robotic motion control to improve robot motion paths. The numerical performance of four TTCG methods, TTLAMR, TTRMIL, TTSMAR, and TTKMAR coefficients are tested using 15 standard test functions with different initial points and variables ranging from 2 to 10,000. The numerical results are evaluated based on the number of iterations (NOI) and CPU time. These results are plotted using a performance profile to evaluate efficiency and robustness. Numerically, TTLAMR outperforms other methods by solving all test functions and it is followed by TTSMAR, TTRMIL, and TTKMAR. Lastly, TTLAMR is implemented in robotic motion control. It shows that TTLAMR is able to be applied to the motion control of two joint planar robotic manipulators.

Keywords: Conjugate Gradient, Optimization, Robotic Motion, Strong Wolfe Line Search

1 INTRODUCTION

Optimization focuses on maximizing or minimizing an objective function while staying within constraints, defining the search area for possible solutions [1]. Optimization problems are commonly applied in fields like computer science, engineering, and economics [2]. These problems are represented by the following mathematical expression:

$$\min_{x \in R^n} f(x) \tag{1}$$

Optimization is solved using an iterative method,

$$x_{k+1} = x_k + a_k d_k, k = 0, 1, 2 \ldots \quad (2)$$

where x_k is the current iteration point, d_k refers to the search direction, and a_k refers to step size. The search direction must ensure that the objective function value decreases, as given by:

$$f(x_{k+1}) < f(x_k) \quad (3)$$

The step size, a_k can be determined using inexact or exact line searches. Both line searches have their advantages. Exact line search produces the exact value of step size while inexact line search produces approximate value of step size. These line searches determine the step size by ensuring a sufficient decrease in the function value. The strong Wolfe line search is quite famous among researchers since it is faster compared to others. The conditions of strong Wolfe line search are stated as follows:

$$f(x_k + a_k d_k) \leq f(x_k) + \delta a_k g_k^T d_k \quad (4)$$

$$|g(x_k + a_k d_k)^T d_k| \leq \sigma \left| g_k^T d_k \right| \quad (5)$$

where $0 < \delta < \sigma < 1$.

These optimization problems may be solved using any four optimization methods that satisfy sufficient descent and global convergence properties. Newton's and Quasi-Newton methods are inefficient for large scale problems due to the need for Hessian matrix calculations. However, Hestenes and Stiefel in 1952 proposed the Conjugate Gradient (CG) method for solving linear systems and later extended for nonlinear optimization problems [3]. The CG method is efficient, requiring minimal storage with no matrix calculations and offering good convergence making it ideal for large problems [4]. The general search direction of CG method is stated as

$$d_k = \begin{cases} -g_k \; if \; k = 0, \\ -g_k + \beta_k d_{k-1} \; if \; k > 0. \end{cases} \quad (6)$$

where g_k is the gradient, and β_k is a CG coefficient.

CG method is classified into scaled, hybrid, three-term, parametric and classical. Over the years, classical CG coefficients have been improved by recent researchers such as Linda-Aini-Mustafa-Rivaie (LAMR) by [5], Sulaiman-Mustafa-Abdelrahman-Rivaie (SMAR) by [6], Kamilu-Mustafa-Abdelrahman-Rivaie (KMAR) by [7], and Rivaie-Mustafa-Ismail-Leong (RMIL) by [8]. The CG coefficients are listed respectively as below:

$$\beta_k^{LAMR} = \frac{g_k^T \left(\frac{\|d_{k-1}\|}{\|d_{k-1} - g_k\|} g_k - g_{k-1} \right)}{\frac{\|d_{k-1}\|}{\|d_{k-1} - g_k\|} \|d_{k-1}\|^2} \quad (7)$$

$$\beta_k^{SMAR} = \frac{g_k^T \left(g_k - \frac{\|g_k\|}{\|g_{k-1}\|} d_{k-1} \right)}{d_{k-1}^T d_{k-1}} \quad (8)$$

$$\beta_k^{KMAR} = \frac{g_k^T(g_k - g_{k-1})}{g_{k-1}^T(g_k + g_{k-1})} \quad (9)$$

$$\beta_k^{RMIL} = \frac{g_k^T(g_k - g_{k-1})}{\|d_{k-1}\|^2} \quad (10)$$

The first three-term CG (TTCG) method is proposed by Beale in 1972 [9]. This method is known for its stability and efficiency. It improves the convergence rate by adding a restart term to the standard search direction. At present, more researchers have introduced new TTCG coefficients such as TTBZAU [10] and TTHS [11]. Basically, TTCG have been applied in various applications such as TTCG with Polak–Ribière–Polyak coefficient for bicriteria optimization [12] and Dai–Yuan coefficient for nonlinear monotone equations with signal recovery problems [13]. Besides, Ibrahim et al. [14] proposed spectral TTCG to recover a sparse signal from incomplete and contaminated sampling measurement.

The CG methods proposed by Ibrahim et al. [15]. Diphofu et al [16] and Sabi'u et al. [17] has been used to solve problems in robotic manipulators by improving motion control accuracy with minimal errors. Besides, Halilu et al. [18] successfully applied CG method in two-joint planar robotic systems with a convergence error as small as 10^{-5} . As stated by Lv et al. [19], effective motion tracking and control help in robotic precision and performance. Thus, it is a good idea to implement TTCG in robotic motion.

2 METHODOLOGY

The important part of the methodology is numerical testing and implementation in robotic motion control. The most crucial aspect of robotics is figuring out the optimal motions for objects. These optimized paths have a direct effect on how quickly, accurately and safely a robot can move. As the compass for the robots, these goals show them how to move, what they need to do, and what their roles are. The main goals in the fascinating field of robotic motion are to make sure that robotic parts work together perfectly, to make route planning algorithms better and to make robots more maneuverable. Thus, it is a good idea to implement TTCG in robotic motion control to evaluate its applicability. The numerical test on LAMR, RMIL, SMAR and KMAR coefficients with three-term search direction is conducted using formula stated below,

$$d_k = -g_k + \beta_k d_{k-1} + \theta_k d_{k-1} \quad (11)$$

where $\theta_k = \frac{g_k^T y_{k-1}}{d_t^T y_{k-1}}$, $y_{k-1} = g_k - g_{k-1}$

These four coefficients are chosen since they are proven theoretically by previous researchers. These coefficients are tested using Matlab software. The algorithm of the TTCG method is given as follows:

Step 1: Initialization. Given x_0 , set $k = 0$.

Step 2: Compute the search direction, d_k as in (11).

Step 3: Compute all coefficients as in (7), (8), (9) and (10).

Step 4: Compute all step size, by using Wolfe line search as in (4).

Step 5: Update a new point based on an iterative formula in (2).

Step 6: Convergence test and stopping criterion. If $f(x_{k+1}) < f(x_k)$ and $\|g_k\| \leq 10^{-6}$, then stop. Otherwise go to Step 1 with $k = k + 1$.

3 RESULTS AND DISCUSSION

All the TTCG coefficients are tested numerically and applied in robotic motion control.

3.1 Numerical Results

Each coefficient is tested on 15 standard test functions. The initial points have been selected randomly. Each method is evaluated over a few dimensions which are 2, 4, 10, 100, 500, 1000, and 10000.

Table 1 : List of Test Functions

No	Test Functions	Initial Points	Dimensions
1	Power	(5,5), (10,10), (15,15), (20,20)	2, 4
2	FLETCHCR	(5,5), (8,8), (15,15), (23,23)	2, 4, 10
3	Hager	(1,1), (2,2), (3,3), (4,4)	2, 4, 10, 100
4	Rayden	(1,1), (2,2), (3,3), (4,4)	2, 4, 10, 100
5	Extended Penalty	(3,3), (10,10), (23,23), (30,30)	2, 4, 10, 100
6	Extended Maratos	(3,3), (13,13), (25,25), (32,32)	2, 4, 10, 100
7	Extended Freudenstein and Roth	(4,4), (8,8), (18,18), (39,39)	2, 4, 10, 100
8	Extended Himmerlblau	(2,2), (6,6), (14,14), (25,25)	2, 4, 10, 100, 500, 1000, 10000
9	Extended White and Holst	(2,2), (9,9), (11,11), (31,31)	2, 4, 10, 100, 500, 1000, 10000
10	Shalow	(2,2), (13,13), (28,28), (39,39)	2, 4, 10, 100, 500, 1000, 10000

11	Extended DENSCHNB	(3,3), (8,8), (15,15), (19,19)	2, 4, 10, 100, 500, 1000, 10000
12	Extended Rosenbrock	(4,4), (8,8), (14,14), (22,22)	2, 4, 10, 100, 500, 1000, 10000
13	Sphere	(8,8), (16,16), (24,24), (32,32)	2, 4, 10, 100, 500, 1000, 10000
14	Diagonal 4	(10,10), (20,20), (30,30), (40,40)	2, 4, 10, 100, 500, 1000, 10000
15	Extended Tridiagonal	(25,25), (35,35), (45,45), (55,55)	2, 4, 10, 100, 500, 1000, 10000

Based on total functions, variables and dimensions in Table 1, there are 292 tests that have been solved by each coefficient. These coefficients are evaluated based on the number of iterations (NOI) and CPU time. The numerical result of tested methods is illustrated using performance profile by Dolan and More [20]. Basically, the efficiency and robustness of the tested methods are referred to the left and the right side of the graphs. The performance profiles of the number of iterations (NOI) and CPU time are shown in Figure 1 and Figure 2 respectively.

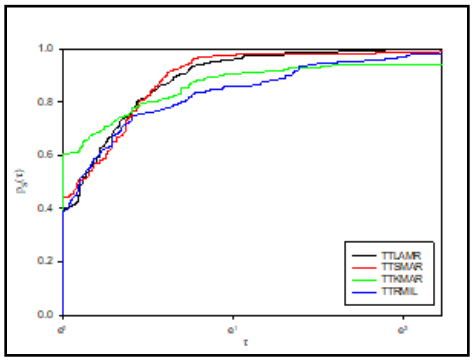

Figure 1 : The Performance Profile for NOI.

Figure 1 illustrates the performance profile for the NOI. Among the tested methods, TTKMAR exhibits the highest curve compared to others and it is followed by TTSMAR, TTRMIL and TTLAMR. However, TTLAMR outperforms others in certain test problem.

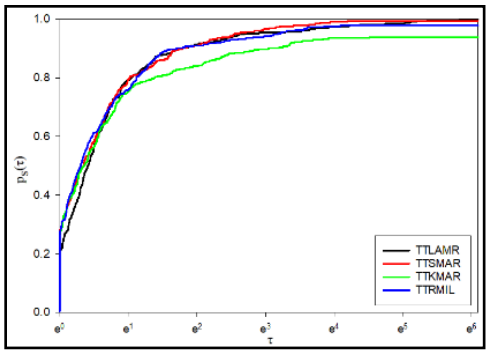

Figure 2: The Performance Profile for CPU Time.

Figure 2 shows that TTSMAR and TTRMIL yields better results in terms of CPU time. However, TTLAMR surpassed them at certain points. As seen in Figure 1 and Figure 2, both graphs yield the same curve. Based on both figures, the results of the right side are tabulated as in Table 2.

Table 2: The Success Rate of TTCG Method.

No	(A) TTCG Coefficients	Success Rate
1	TTLAMR	100%
2	TTSMAR	99.38%
3	TTKMAR	93.83%
4	TTRMIL	97.84%

TTLAMR has the greatest success rate of 100% when solving test functions, while TTSMAR, TTRMIL and TTKMAR solve 99.38%, 97.84% and 93.83%, respectively. Thus, TTLAMR is known as the most robust method. The example of performance ratio and success rate can be computed as follows.

1. Find minimum value of $t_{p,s}$:

$$\min\{0.0086, 0.0001197, 0.0002071, 0.0000995\} = 0.0000995$$

where p is a collection of test functions, s is a set of algorithms, $t_{p,s}$ is the cost of solving problem $p \in P$ by algorithm $s \in S$.

2. Next, calculate the performance ratio:

$$r_{p,s} = \frac{t_{p,s}}{min\{t_{p,s} : s \in S\}}$$

$$r_{p,s} = \frac{0.0086}{0.0000995} = 86.4321608$$

3. Calculate the percentage:

The calculation for $\rho_s(\tau)$ assumes a total of 324 cases.

$$\rho_s(\tau) = \frac{324}{324} \times 100 = 100\%$$

3.2 The Application of Robotic Motion Control

TTLAMR has been chosen to be applied in robotic motion control since it produces the best numerical result. TTLAMR method is designed to enhance the trajectory tracking and efficiency of robotic manipulators. TTLAMR is implemented into robotic motion algorithm under strong Wolfe line search.

TTLAMR coefficient begins with initializing the current state x_k and sets up parameters such as γ, ρ, and Q_0. The objective is to minimize the error between the desired position rd and the actual position achieved by the robotic manipulator, which is represented by r_k. This error is calculated using the norm of the difference between the cosine and sine components of the joint angles x_k.

During each iteration, the algorithm calculates the gradient of the error with respect to the joint angles. The TTCG method is employed to refine the joint angles, ensuring smoother and more accurate movements towards the desired trajectory. This iterative process continues until a predefined convergence criterion is met. It is to ensure the norm of the gradient falls below a specified threshold.

In the implementation for a 3-degree-of-freedom (3DOF) robotic manipulator, the algorithm iterates over a set period, adjusting the joint angles based on the TTLAMR updates. This iterative adjustment aims to minimize the discrepancy between the actual and desired trajectories, thereby improving the overall tracking performance of the robotic system.

Visualizations of the trajectory, including the desired path and the actual trajectory achieved by the robotic manipulator, are plotted to evaluate the algorithm's effectiveness. These plots provide insights into the convergence behaviour and accuracy of the TTLAMR method in robotic motion.

A discrete-time kinematics equation of a three-joint planar robot manipulator is described by the following model at the position level.

$$\Gamma(\mu_k) = \eta_k, \tag{13}$$

where the joint angle vector $\mu_k \in \mathbb{R}^2$ and the end effector vector location $\eta_k \in \mathbb{R}^2$ are shown, respectively. The kinematic function with the following structure is represented by the vector valued function τ.

$$\Gamma(\mu_k) = \begin{bmatrix} \tau_1 \cos(\mu_1), \tau_2 \sin(\mu_1) \\ \tau_1 (\cos(\mu_1) + \cos(\mu_1 + \mu_2)), \tau_2 (\sin(\mu_1) + \sin(\mu_1 + \mu_2)) \\ \tau_1 (\cos(\mu_1) + \cos(\mu_1 + \mu_2) + \tau_2 \cos(\mu_1 + \mu_2 + \mu_3)), \\ \tau_1 (\sin(\mu_1) + \sin(\mu_1 + \mu_2) + \tau_2 \sin(\mu_1 + \mu_2 + \mu_3)) \end{bmatrix} \quad (14)$$

with τ_1 , τ_2 and τ_3 standing for the lengths of the first, second and third rods, respectively. The following nonlinear least squares model is to be minimised in the context of motion control at each instantaneous computational time interval $[t_k, t_k + 1] \subseteq [0, t_f]$, where t_f is the end of task duration. The following nonlinear least squares model is to be minimized.

$$\min_{\Gamma_k \in \mathbb{R}^2} \frac{1}{2} \left\| \Gamma_k - \hat{\Gamma}_k \right\|^2 \quad (15)$$

where the end effector-controlled track is indicated by $\hat{\Gamma}_k$. As stated by Sulaiman et al. [21], the controlled end effector tracking of a Lissajous curve provided as,

$$\hat{\Gamma}_k = \begin{bmatrix} \frac{3}{2} + \frac{1}{5} \sin\left(\frac{\pi t_k}{5}\right) \\ \frac{\sqrt{3}}{2} + \frac{1}{5} \sin\left(\frac{2\pi t_k}{5} + \frac{\pi}{3}\right) \end{bmatrix} \quad (16)$$

The Lissajous curve describes the graph of parametric equations where two perpendicular oscillations in x and y directions. TTLAMR is implemented using the parameters $\tau_1 = 1, \tau_2 = 1, \tau_3 = 3$ and $t_f = 20$ seconds. Beginning with $\mu_0 = [\mu_1, \mu_2, \mu_3] = \left[0, \frac{\pi}{3}, \frac{\pi}{2}\right]^T$ where 200 identical segments make up the task duration [0, 20].

Based on the implementation of TTCG in robotic motion, the robot arm motion trajectory, desired vs actual trajectory, error in x and y coordinates are shown in Figures 3 – 6 respectively.

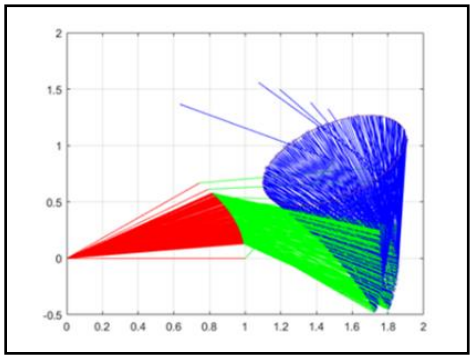

Figure 3: Robot Arm Motion

Figure 3 shows how the 3-joint robot arm moves. The red, green, and blue lines represent different parts of the arm. As time passes, you can see the arm moving from its starting position to its final position. The lines help us understand how each part of the arm works together to follow the planned path.

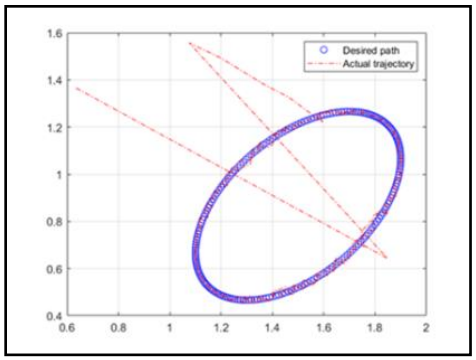

Figure 4: Desired Path vs Actual Trajectory

Figure 4 compares the planned path (blue spheres) with the actual path (red line) that the robot's end-effector (the tool at the end of the arm) followed. It helps us see how well the robot was able to follow the desired path, and any differences or errors between the two paths.

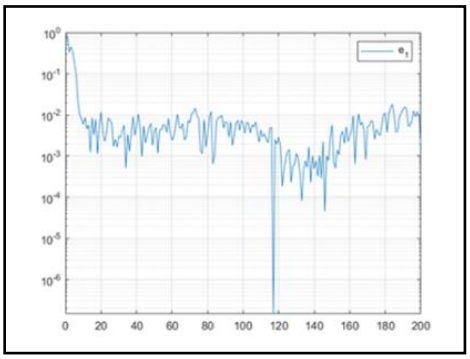

Figure 5: Error in x-Coordinate (e1)

Figure 5 shows the error in the robot's horizontal (x) movement over time. Again, the error starts off large but decreases as the robot makes corrections. The graph shows that the robot's control system is effectively reducing the error and improving the accuracy of the movement.

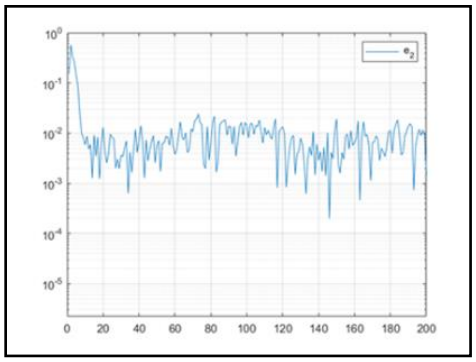

Figure 6: Error in y-Coordinate (e2)

Similar to Figure 5, Figure 6 shows how the error in the robot's vertical (y) movement changes over time. The graph uses a logarithmic scale, which makes it easier to see how the error decreases. At first, the error is big, but as the robot adjusts its movement, the error gets smaller, meaning the robot is improving its accuracy.

Table 3: Mean Absolute Error for X and Y Desired vs Actual Coordinate.

No	Coordinate	Mean Absolute Error
1	X	0.010428678
2	Y	0.011683541

Table 3 shows the average difference between the robot's planned position and the actual position it reached for both the X and Y coordinates. The average error for the X coordinate is 0.0104 units, meaning the robot's position was off by this amount on average. The average error for Y coordinate is a bit larger compared to X coordinate which is 0.0117 units. These small error values indicate that the robot is able to follow the desired path quite accurately, with only slight differences between the planned and actual movements.

4 CONCLUSION

As a conclusion, TTLAMR method under strong Wolfe line search is the most robust method since it is able to solve all the test functions. The numerical result of TTLAMR is promising in terms of NOI and CPU time. Besides, the implementation of TTLAMR in robotic motion control is a success, particularly in improving motion control and energy efficiency. Thus, this method can be further studied by using other line searches and applications such as Genetic Algorithm (GA), and Particle Swarm Optimization (PSO).

REFERENCES

[1] M. Nazari-Heris and B. Mohammadi-Ivatloo, "Application of robust optimization method to power system problems," in *Classical and Recent Aspects of Power System Optimization*, pp. 19–32, 2018.

[2] D. Mehta and C. Grosan, "A collection of challenging optimization problems in science, engineering and economics," in *Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2697–2704, 2015.

[3] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *Journal of Research of the National Bureau of Standards*, vol. 49, no. 6, pp. 409–436, 1952.

[4] J. S. Arora, *Introduction to Optimum Design*. Elsevier, 2004.

[5] N. Zullpakkal, N. Shapiee, S. M. Zokri, and M. Rivaie, "The numerical calculation of hybrid conjugate gradient method under Armijo line search and its application," *Matematika*, pp. 147–155, 2021.

[6] I. Sulaiman, S. Supian, and M. Mamat, "New class of hybrid conjugate gradient coefficients with guaranteed descent and efficient line search," in *IOP Conference Series: Materials Science and Engineering*, p. 012021, 2019.

[7] K. U. Kamfa, M. Mamat, A. Abashar, M. Rivaie, P. L. B. Ghazali, and Z. Salleh, "Another modified conjugate gradient coefficient with global convergence properties," *Applied Mathematical Sciences*, vol. 9, no. 37, pp. 1833–1844, 2015.

[8] M. B. Yousef, M. Mamat, and M. Rivaie, "A new modified RMIL CG method with global convergence properties," in *AIP Conference Proceedings*, p. 060050, 2019.

[9] E. Beale, "A derivation of conjugate-gradients," in *Numerical Methods for Non-Linear Optimization*, 1972.

[10] B. Baluch, Z. Salleh, A. Alhawarat, and U. Roslan, "A new modified three-term conjugate gradient method with sufficient descent property and its global convergence," *Journal of Mathematics*, vol. 2017, no. 1, p. 2715854, 2017.

[11] A. Alhawarat, Z. Salleh, H. Alolaiyan, H. El Hor, and S. Ismail, "A three-term conjugate gradient descent method with some applications," *Journal of Inequalities and Applications*, vol. 2024, no. 1, p. 73, 2024.

[12] Y. Elboulqe and M. El Maghri, "An explicit three-term Polak–Ribière–Polyak conjugate gradient method for bicriteria optimization," *Operations Research Letters*, vol. 57, p. 107195, 2024.

[13] A. Yusuf, N. H. Manjak, A. M. Kwami, and M. Abdulhameed, "A modified three-term of Dai–Yuan type derivative-free algorithm for nonlinear monotone equations with signal recovery problems," *Franklin Open*, vol. 11, p. 100255, 2025.

[14] A. H. Ibrahim, M. Alshahrani, and S. Al-Homidan, "Two classes of spectral three-term

derivative-free method for solving nonlinear equations with application," *Numerical Algorithms*, vol. 96, no. 4, pp. 1625–1645, 2024.

[15] A. H. Ibrahim and S. Al-Homidan, "A spectral conjugate gradient method for motion control of robotic manipulators," *Engineering Computations*, vol. 42, no. 5, pp. 1717–1727, 2025.

[16] T. Diphofu and P. Kaelo, "An efficient conjugate gradient method based on the conjugacy condition with an application in motion control," *Iranian Journal of Science*, pp. 1–8, 2025.

[17] J. Sabi'u and R. Z. Al-Kawaz, "An efficient modified conjugate gradient parameter for solving the system of symmetric nonlinear equations with application in motion control of coplanar robot," *Computational and Applied Mathematics*, vol. 44, no. 1, p. 49, 2025.

[18] A. S. Halilu, A. Majumder, M. Y. Waziri, K. Ahmed, and A. M. Awwal, "Motion control of the two joint planar robotic manipulators through accelerated Dai–Liao method for solving system of nonlinear equations," *Engineering Computations*, vol. 39, no. 5, pp. 1802–1840, 2021.

[19] X. Lv, X. Huang, and M. Wang, "Trajectory snakes for robotic motion tracking," in *Proceedings of the 2006 International Conference on Mechatronics and Automation*, pp. 1152–1157, 2006.

[20] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical Programming*, vol. 91, no. 2, pp. 201–213, 2002.

[21] I. M. Sulaiman, M. Malik, A. M. Awwal, P. Kumam, M. Mamat, and S. Al-Ahmad, "On three-term conjugate gradient method for optimization problems with applications on COVID-19 model and robotic motion control," *Advances in Continuous and Discrete Models*, vol. 2022, no. 1, 2022.