International Journal of Networked and Distributed Computing
https://doi.org/10.1007/s44227-023-00016-7

REVIEW ARTICLE

Enhancements in Circle Rendering: An Improved Approach to the Midpoint Circle Drawing Algorithm

N. S. Nithya[1] · M. Javed Idrisi[2]

Received: 23 October 2023 / Accepted: 14 November 2023

Abstract

Rendering circles on digital platforms is essential in computer graphics. The widely adopted midpoint circle algorithm is a premier tool for this task. This article dives deep into the foundational aspects of this algorithm, highlighting its prowess in enabling accurate and effortless circle depiction on digital displays. It further contrasts its merits with alternative circle drawing techniques. Importantly, our research presents a refined technique aimed at minimizing inaccuracies inherent in the standard cartesian-based Midpoint Circle Drawing Algorithm (MPCDA). By incorporating an error-reduction strategy, we ensure greater accuracy in depicting circles. Comparative tests on diverse circle sizes and placements confirmed the efficacy of our enhanced method. When set against results from the traditional MPCDA, our approach showcases significant improvements.

Keywords Computer graphics · Circle drawing · MPCDA · Error-reduction algorithm

1 Introduction

In the realm of computer graphics, representing perfect circles on digital canvases is a challenge that's long been faced by developers and designers. The Midpoint Circle Drawing Algorithm elegantly addresses this challenge, acting as a cornerstone technique for circle generation on raster displays. The algorithm harnesses the inherent symmetry of circles, allowing for efficient and accurate rendering. Instead of calculating every individual pixel value for a circle, it computes for just a segment and then extrapolates for the entire shape, thereby optimizing computational resources. This not only enhances the speed of rendering but also ensures precision, a critical aspect especially when graphics require exactness in representation. Whether it's in sophisticated graphic design software, video game renderings, or basic GUI elements in software applications, the ubiquity of this algorithm underscores its importance in the digital world. Its blend of mathematical simplicity with practical efficiency makes it a fundamental pillar in computer graphics.

In computer graphics, circle drawing stands as a foundational technique. Most graphics software, when generating graphs and visuals, relies heavily on this geometric shape. Over time, various methodologies have emerged for rasterizing circles on screens. Bresenham's algorithm, introduced by Bresenham [1], remains the most notable circle creation technique, with its variations detailed in works by McIlroy [2] and Liu [3]. An advanced method, introduced by Wu and Rokne [4], offers scan conversion of lines and circles at twice the pixels per iteration, essentially quickening curve generation by increasing increment size. Their findings indicate that their double-step algorithms work as efficiently as single-step ones but with half the iterations. Kuzmin [5] proposed an incremental algorithm that, while producing results identical to Bresenham's, uses fewer arithmetic operations. Liu and Li [6] advocated for a double step circle drawing approach, relying solely on integer arithmetic, and demonstrated its superior speed compared to existing methods. Aken [7] showcased a highly precise midpoint algorithm for drawing ellipses on raster displays, emphasizing its accuracy even when Bresenham's extended algorithm falls short. In

✉ M. Javed Idrisi
javed@mtu.edu.et

N. S. Nithya
nithya.ns@vit.ac.in

[1] Department of Software Systems, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamilnadu, India

[2] Department of Mathematics, College of Natural and Computational Science, Mizan-Tepi University, Tepi, Ethiopia

Published online: 06 December 2023

a different vein, Kappel [8] combined previous techniques to formulate an optimized approach for drawing discrete ellipses. Similarly, Fellner et al. [9] have explored efficient algorithms for perfect ellipse approximations. Agathos et al. [10], inspired by Bresenham's methodology, introduced a fast method for rendering conic sections with alignment to coordinate axes. Their method was efficient, used a smaller integer range, and maintained accurate area transitions, even allowing for easy integration of antialiasing. Haiwen et al. [11], using a combined strategy, suggested swift techniques for ellipse drawing. Finally, according to Dimri et al. [12], the midpoint ellipse algorithm remains a preferred method for drawing ellipses, especially with techniques leveraging the reflection around the line $x = y$ using the ellipse's parametric equation. In [13] proposed a method which efficiently detects concentric circles in grayscale images by combining the minimum envelopment circle and the least square ellipse fitting approach. Experimental results indicate superior detection performance compared to classical algorithms. In [14] proposed a circle detection algorithm that corrects elliptical distortions in optical CCD detection by employing ellipse de-falsification and various preprocessing techniques, including the use of RANSAC for outlier removal. The efficacy of this approach is evident in the experimental results, which highlight a minimal detection error of about 0.3%. In [15] presented two optimizations for the Dijkstra algorithm, focusing on urban express traffic: one enhances directional calculation to reduce computational load, and the other optimizes the graphical model by reducing node count. Comparative analysis validates the efficiency of their improved method, streamlining complexity and boosting practical applicability. The study of [16] introduces an approach to detecting multiple ellipses in data, treating the problem through the lens of cluster analysis and envisioning each ellipse as a Mahalanobis circle. By utilizing modified k-means and DIRECT algorithms, the method efficiently identifies ellipses even when their quantity is undetermined, with a novel Geometrical Objects-index (GO-index) assisting in selecting the optimal partition. Notably, while many existing algorithms falter with unclear or noisy ellipse edges, this method excels in such scenarios, as evidenced by numerous test examples.

In this study, we introduce a novel method to minimize errors in the prevailing MPCDA by employing diverse 'h' values to achieve optimal accuracy. The research article is structured into five distinct sections. Section 1 serves as the introduction and sheds light on the latest advancements related to MPCDA. Section 2 delves deeper into the intricacies of the midpoint circle drawing algorithm. In Sect. 3, we illustrate the method by plotting a circle with an 8-unit radius using various 'h' values. The findings are thoroughly discussed in Sect. 4, while Sect. 5 offers a comprehensive conclusion. The final segment of the paper lists the references.

2 Enhancements in Circle Rendering: A Dive into the Improved Algorithm

In our refined algorithm, the x and y coordinates' progression is contingent upon increments or decrements of 'h' units. Our initial focus is on plotting points within the first octant. Leveraging the inherent symmetry of circles, these points then guide the plotting in the subsequent octants. Specifically, for the first octant, plotting begins at the coordinate $(0, r)$, where 'r' denotes the circle's radius.

Taking a circle centered at the origin with a radius of 'r' units, its equation can be defined as $x^2 + y^2 = r^2$. From this, we derive a function $f(x, y)$ represented as $f(x, y) = x^2 + y^2 - r^2$. The circle encompasses all points (x, y) that satisfy $f(x, y) = 0$, assuming x and y as real values. Points satisfying $f(x, y) < 0$ are found within the circle, whereas those fulfilling $f(x, y) > 0$ lie outside the circle's boundary. This delineation is visually presented in Fig. 1.

Any point in the first octant may be chosen as (x_k, y_k), and it is assumed that this point is moving clockwise in the xy-plane. In the first octant, the next point to (x_k, y_k) is therefore provided by $(x_k + h, y_k)$ or $(x_k + h, y_k - h)$, where h is the grid's width. So long as (x_m, y_m) represents the midpoint between $(x_k + h, y_k)$ and $(x_k + h, y_k - h)$, we have

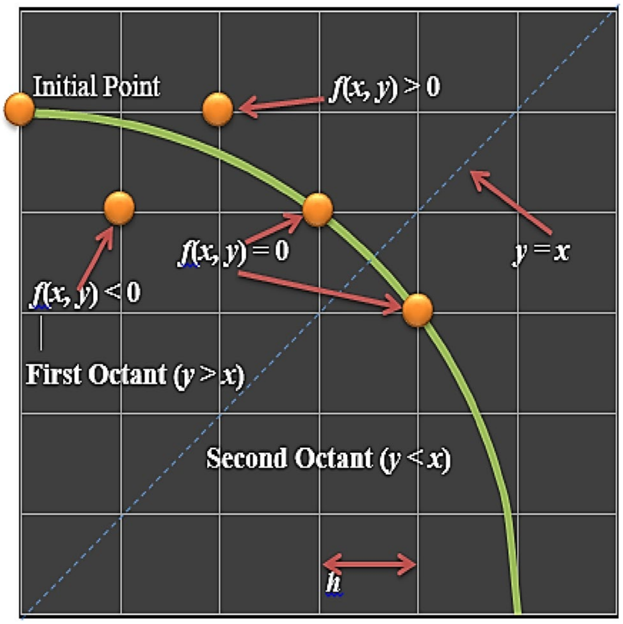

Fig. 1 The first and second octant of a circle $x^2 + y^2 = r^2$

$(x_m, y_m) = (x_k + h, y_k - h/2)$.

If P_k represents the value of $f(x, y)$ at the mid-point (x_m, y_m), therefore,

$$P_k = f(x_m, y_m) = (x_k + h)^2 + (y_k - h/2)^2 - r^2. \tag{1}$$

Thus, P_{k+1} will be given by,

$$P_{k+1} = (x_{k+1} + h)^2 + (y_{k+1} - h/2)^2 - r^2 \tag{2}$$

where, $x_{k+1} = x_k + h$ and $y_{k+1} = y_k$ or $y_k - h$. Thus, form Eq. (2), we have

$$P_{k+1} = (x_k + 2h)^2 + (y_{k+1} - h/2)^2 - r^2. \tag{3}$$

On subtracting Eq. (1) from Eq. (3), and then simplifying, we get

$$P_{k+1} = P_k + 2h(x_k + h) + (y_{k+1}^2 - y_k^2) - h(y_{k+1} - y_k) + h^2. \tag{4}$$

2.1 Initial Decision Parameter

Let the initial point be $(0, r)$, i.e., $x_k = 0$ and $y_k = r$, therefore from Eq. (1), we have

$$P_k = h^2 + (r - h/2)^2 - r^2 = \frac{5}{4}h^2 - hr \approx h^2 - hr.$$

Thus,

$$P_k = h(h - r). \tag{5}$$

If $P_k \geq 0$, then $y_{k+1} = y_k - h$ and the next point will be $(x_k + h, y_k - h)$, which gives

$$P_{k+1} = P_k + 2h(x_k - y_k + h) + h^2. \tag{6}$$

If $P_k < 0$, then $y_{k+1} = y_k$ and the next point will be $(x_k + h, y_k)$, which gives

$$P_{k+1} = P_k + 2h(x_k + h) + h^2. \tag{7}$$

If we take $h = 1$, then the initial decision parameter from Eq. (5) is $P_k = 1 - r$ and

$$P_{k+1} = \begin{cases} P_k + 2(x_k - y_k) + 3, & P_k \geq 0 \\ P_k + 2x_k + 3 &, P_k < 0 \end{cases}$$

gives the classical case of MPCDA.

Table 1 Points in the first octant for $h = 1$

x_k	y_k	P_k	x_{k+1}	y_{k+1}
0	8	− 7	1	8
1	8	− 2	2	8
2	8	5	3	7
3	7	0	4	6
4	6	− 1	5	6

Table 2 Points in the first octant for $h = 0.5$

x_k	y_k	P_k	x_{k+1}	y_{k+1}
0	8	− 3.75	0.5	8
0.5	8	− 2.5	1	8
1.0	8	− 0.75	1.5	8
1.5	8	1.5	2	7.5
2.0	7.5	− 3.25	2.5	7.5
2.5	7.5	0	3	7
3.0	7	− 3.25	3.5	7
3.5	7	1	4	6.5
4.0	6.5	− 0.75	4.5	6.5
4.5	6.5	4.5	5	6
5.0	6	4.25	5.5	5.5

3 Results

In this particular analysis, we are implementing the adjusted version of the midpoint circle drawing algorithm to graphically represent a circle, which is defined by a radius spanning 8 units. To simplify this task, we are focusing exclusively on pinpointing and plotting the points located within the first octant, or one-eighth segment of the circle. For this purpose, our starting point is designated as (0, 8).

A noteworthy aspect of this endeavor is the variable 'h'. We have chosen three distinct values for 'h' to examine how they influence the circle's representation. The chosen values are $h = 1$, $h = 0.5$, and $h = 0.1$. Detailed numerical breakdowns for each of these 'h' values are meticulously presented in three separate tables. Specifically, Table 1 is dedicated to the results for $h = 1$, Table 2 focuses on $h = 0.5$, and Table 3 encapsulates the data for $h = 0.1$. To visually illustrate how the circle appears based on these varying 'h' values within the first octant, we have prepared diagrams. Figures 2, 3, and 4 provide graphical representations of the circle corresponding to the 'h' values of 1, 0.5, and 0.1, respectively.

Moreover, we have taken it a step further. Beyond just the first octant, we also visualize how the circle is depicted within the entire first quadrant (or one-fourth of the circle) for the aforementioned 'h' values. This broader view can be observed in Figs. 5, 6, and 7, which correspond to the 'h' values of 1, 0.5, and 0.1, respectively.

Table 3 Points in the first octant for $h=0.1$

x_k	y_k	P_k	x_{k+1}	y_{k+1}	x_k	y_k	P_k	x_{k+1}	y_{k+1}
0	8	− 0.79	0.1	8	2.8	7.5	0.47	2.9	7.4
0.1	8	− 0.74	0.2	8	2.9	7.4	− 0.4	3.0	7.4
0.2	8	− 0.67	0.3	8	3.0	7.4	0.23	3.1	7.3
0.3	8	− 0.58	0.4	8	3.1	7.3	− 0.58	3.2	7.3
0.4	8	− 0.47	0.5	8	3.2	7.3	0.09	3.3	7.2
0.5	8	− 0.34	0.6	8	3.3	7.2	− 0.66	3.4	7.2
0.6	8	− 0.19	0.7	8	3.4	7.2	0.05	3.5	7.1
0.7	8	− 0.02	0.8	8	3.5	7.1	− 0.64	3.6	7.1
0.8	8	0.17	0.9	7.9	3.6	7.1	0.11	3.7	7
0.9	7.9	− 1.2	1.0	7.9	3.7	7	− 0.52	3.8	7
1.0	7.9	− 0.97	1.1	7.9	3.8	7	0.27	3.9	6.9
1.1	7.9	− 0.72	1.2	7.9	3.9	6.9	− 0.3	4.0	6.9
1.2	7.9	− 0.45	1.3	7.9	4.0	6.9	0.53	4.1	6.8
1.3	7.9	− 0.16	1.4	7.9	4.1	6.8	0.02	4.2	6.7
1.4	7.9	0.15	1.5	7.8	4.2	6.7	− 0.45	4.3	6.7
1.5	7.8	− 1.08	1.6	7.8	4.3	6.7	0.44	4.4	6.6
1.6	7.8	− 0.73	1.7	7.8	4.4	6.6	0.03	4.5	6.5
1.7	7.8	− 0.36	1.8	7.8	4.5	6.5	− 0.34	4.6	6.5
1.8	7.8	0.03	1.9	7.7	4.6	6.5	0.61	4.7	6.4
1.9	7.7	− 1.1	2.0	7.7	4.7	6.4	0.3	4.8	6.3
2.0	7.7	− 0.67	2.1	7.7	4.8	6.3	0.03	4.9	6.2
2.1	7.7	− 0.22	2.2	7.7	4.9	6.2	− 0.2	5.0	6.2
2.2	7.7	0.25	2.3	7.6	5.0	6.2	0.83	5.1	6.1
2.3	7.6	− 0.78	2.4	7.6	5.1	6.1	0.66	5.2	6
2.4	7.6	− 0.27	2.5	7.6	5.2	6	0.53	5.3	5.9
2.5	7.6	0.26	2.6	7.5	5.3	5.9	0.44	5.4	5.8
2.6	7.5	− 0.69	2.7	7.5	5.4	5.8	0.39	5.5	5.7
2.7	7.5	− 0.12	2.8	7.5	5.5	5.7	0.38	5.6	5.6

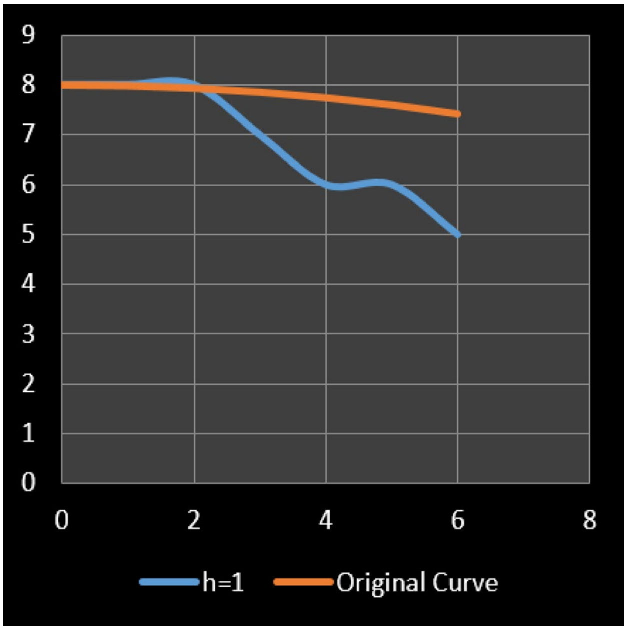

Fig. 2 Original curve versus curve for $h=1$ in the first octant

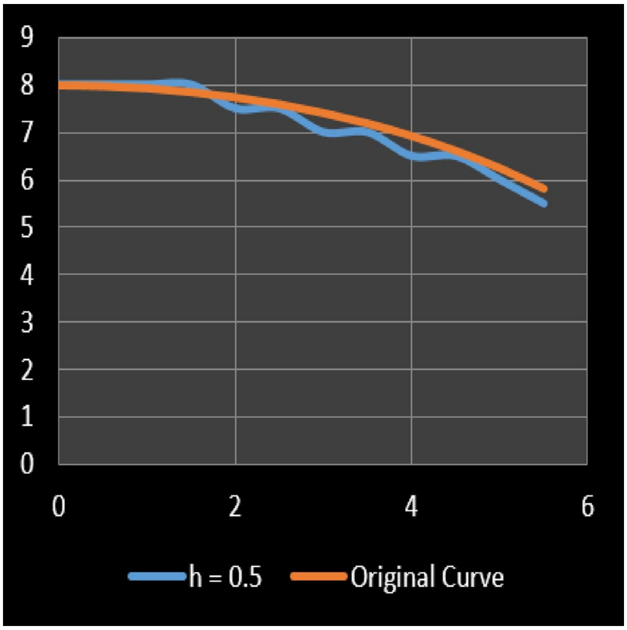

Fig. 3 Original curve versus curve for $h=0.5$ in the first octant

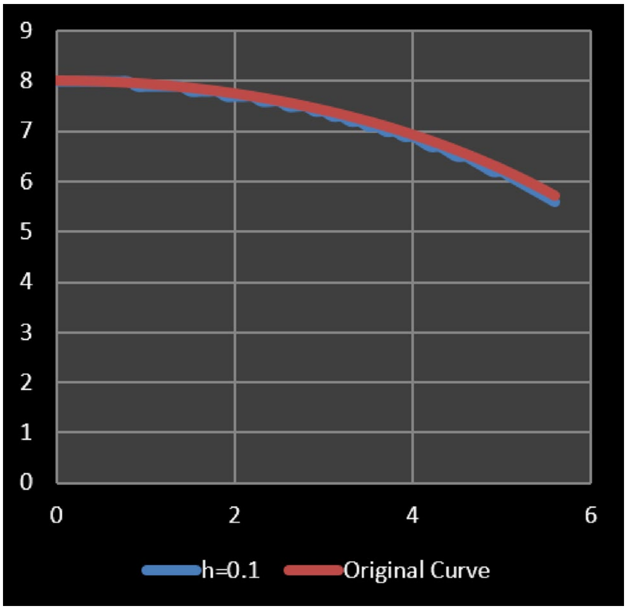

Fig. 4 Original curve versus curve for $h = 0.1$ in the first octant

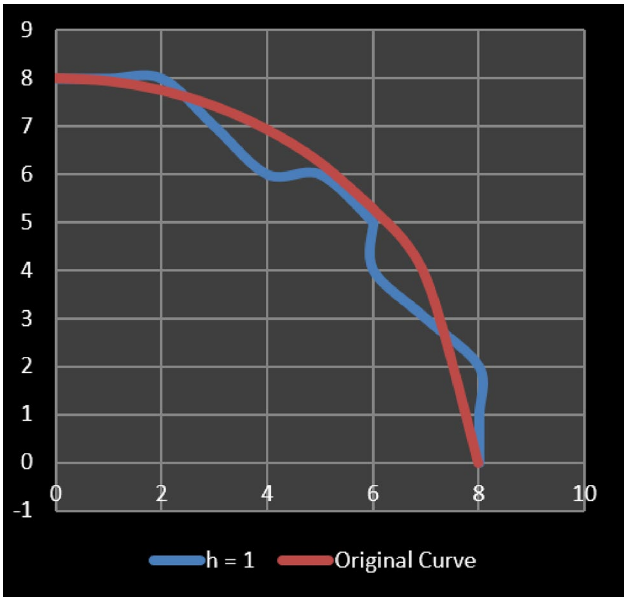

Fig. 5 Original curve versus curve for $h = 1$ in the first quadrant

In essence, this segment of the study provides both numerical and visual insights into how the modified midpoint circle drawing algorithm functions under different '*h*' values, and the results they yield.

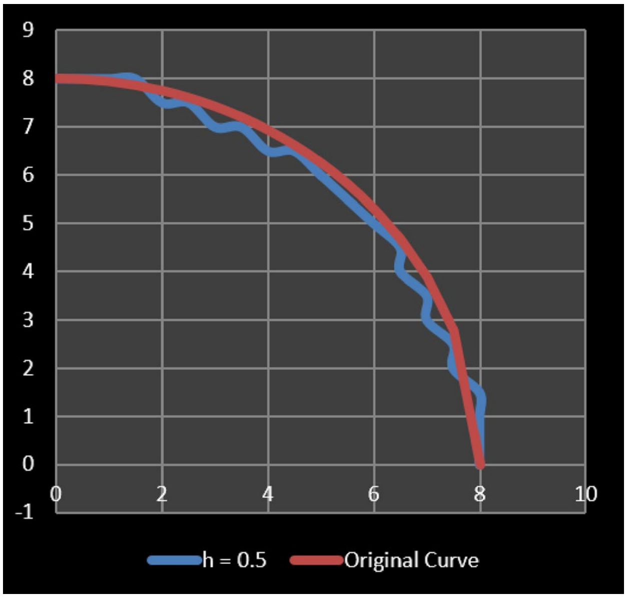

Fig. 6 Original curve versus curve for $h = 0.5$ in the first quadrant

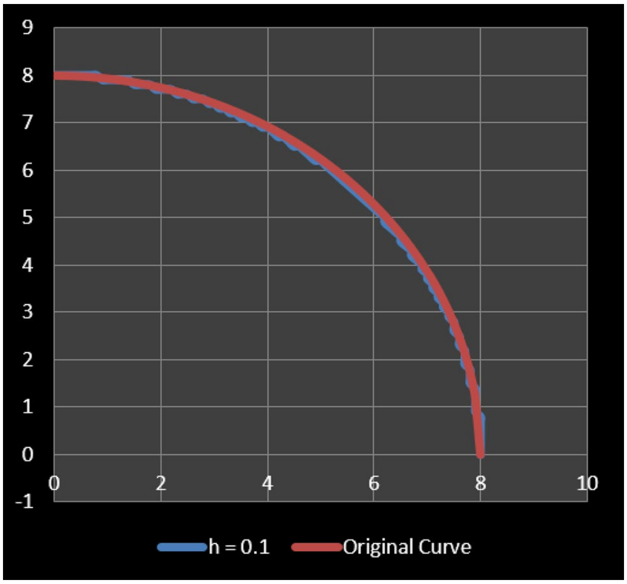

Fig. 7 Original curve versus curve for $h = 0.1$ in the first quadrant

4 Discussion

In the research paper at hand, a novel methodology aimed at error minimization for the prevailing Midpoint Circle Drawing Algorithm (MPCDA) is introduced. One of the distinctive features of this approach is the incorporation of variable '*h*' values. By fine-tuning these values, it becomes feasible to considerably lower the associated errors, pushing the limits of accuracy in circle rendering.

Proceeding to Sect. 2, a deep dive into the intricacies of the midpoint circle drawing algorithm is undertaken. Here, readers are offered a detailed understanding of the algorithm's core mechanics, functioning, and its foundational principles.

Section 3 transitions into a practical application of the discussed algorithm. In this segment, a circle, characterized by a radius measuring 8 units, is plotted. The unique aspect here is the deployment of diverse '*h*' values during the plotting process, showcasing the algorithm's adaptability and the influence of varying '*h*' values on the outcome.

Subsequently, Sect. 4 delves into a thorough discussion of the results obtained from the previously demonstrated example. This section aims to shed light on the implications, significance, and potential applications of the findings, making it crucial for those seeking actionable insights or future research directions.

5 Conclusion

In conclusion, the rendering of circles on digital platforms holds undeniable importance in the realm of computer graphics, and the widely embraced midpoint circle algorithm stands out as a premier tool for this fundamental task. Throughout this article, we have delved into the foundational aspects of the algorithm, shedding light on its remarkable capabilities in facilitating accurate and seamless circle depiction on digital displays. The significance of the midpoint circle algorithm becomes even more apparent as we contrast its merits with alternative circle drawing techniques. Its efficiency and effectiveness in achieving precision set it apart as a reliable choice for graphic rendering. However, acknowledging the inherent inaccuracies in the standard cartesian-based Midpoint Circle Drawing Algorithm (MPCDA), our research has introduced a refined technique. The core focus of our enhancement lies in the integration of an error-reduction strategy, aimed at minimizing inaccuracies that may arise during the rendering process. This nuanced approach ensures greater accuracy in depicting circles, addressing a key concern associated with the traditional MPCDA. The comparative tests conducted on diverse circle sizes and placements substantiate the efficacy of our enhanced method. When pitted against results obtained using the traditional MPCDA, our approach not only demonstrates its prowess but also showcases significant improvements. The refined technique not only preserves the strengths of the original algorithm but elevates its performance, making it a compelling choice for digital circle rendering in computer graphics.

In essence, the research presented here contributes to the ongoing evolution of circle rendering techniques, providing a valuable refinement to a widely adopted algorithm. As digital displays continue to play an increasingly integral role in various fields, the pursuit of accuracy and efficiency in graphical representation becomes paramount, and our enhanced midpoint circle algorithm contributes significantly to this endeavor.

Data Availability The data used to support the findings of this study are included within the article. For simulation, we have used data from other research papers which are properly cited.

Declarations

Conflict of Interest The authors declare that there are no conflicts of interest regarding publication of this manuscript.

References

1. Bresenham J (1977) A linear algorithm for incremental digital display of circular arcs. Commun ACM 20:100–106
2. McIlroy MD (1983) Best approximate circles on integer grids. ACM Trans Graph 2:237–263
3. Liu YK (1993) Algorithm for circle approximation and generation. Comput Aided Des 25:169–171
4. Wu X, Rokne JG (1987) Double-step incremental generation of lines and circles. CVGIP 37:331–344
5. Kuzmin YP (1990) An efficient circle drawing algorithm. Comput Graph Forum 9:337–348
6. Liu YK, Li XN (2007) Double step circle drawing algorithm with and without grey scale. In: 4th International Journal of Image and Graphics, pp 886–891
7. Aken JV (1984) An efficient ellipse-drawing algorithm. IEEE Comput Graph Appl 4:24–35
8. Kappel MR (1985) An ellipse-drawing algorithm for raster displays. Fundam Algorithms Comput Graph 17:257–280
9. Fellner DW, Helmberg C (1994) Best approximate general ellipses on integer grids. Comput Graph 18:143–151
10. Agathos A, Theoharis T, Boehm A (1998) Efficient integer algorithms for the generation of conic sections. Comput Graph 22(5):621–628
11. Haiwen F, Lianqiang N (2012) A hybrid generating algorithm for fast ellipses drawing. In: International Conference on Computer Science and Information Processing (CSIP 2012), Xian, Shaanxi, pp 1022–1025
12. Dimri SC, Ram M (2017) An efficient algorithm to scan conversion of ellipse under auxiliary circle. Nonlinear Stud 24:181–191
13. Gu Y, Yang H, Huo J (2019) Concentric circle detection method based on minimum enveloping circle and ellipse fitting. In: 2019

IEEE 10th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, pp 523–527

14. Guo S, Yang S, Zhang P (2021) A circle detection algorithm based on ellipse removal. J Image Process Theory Appl 4(1):42–50
15. Bing H, Lai L (2022) Improvement and application of Dijkstra algorithms. Acad J Comput Inf Sci 5:97–102
16. Scitovski R, Sabo K, Nikic P, Majstorovic S (2023) A new efficient method for solving the multiple ellipse detection problem. Expert Syst Appl 222:119853

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.