

Research and Implementation of the Fundamental Algorithms of Computer Graphics Based on VC

Honglin Li

School of Information Engineering, Qujing Normal University, Qujing Yunnan 655011, China

lihonglin28286@sina.com

Keywords: VC; Line; Circle; Bezier; Curve; Algorithm

Abstract. This paper analyzes the basic linear generation algorithm, circle generation and filling algorithm, Bezier curve generation algorithm in the graphics, and combined with the VC to achieve the algorithm. The theoretical and experimental comparison of 4 basic algorithms for linear generation. This paper discusses 5 methods of circular regions filling, the midpoint circle algorithm, Bresenham algorithm, circle equation, seed filling algorithm and symmetry points algorithm, and then the efficiency of these 5 kinds of Circle filling algorithm is compared by experiments. According to discuss and analyze the different implementation ways. It has a certain guiding significance. It is good for students to master relevant knowledge and improve their professional skills.

Introduction

Computer graphics is a more active computer science subjects, but also the computer related professional courses or elective courses, its application has been deep into all areas of society. The course principles is deep, and algorithms are abstract, class is limited, and with the mathematics, linear algebra, computer science and other related disciplines very closely, if the teaching process in some of the links are not handled properly, will make the students learning interest is not strong, weariness. Therefore, in the teaching, how to arrange the teaching content more reasonably, what method to take to make the students accept and understand the algorithm and can achieve on the computer is very important. Through my years of graphics teaching, in the algorithm to achieve the process of selecting a good programming software is very important. Because of Visual C++ (VC) in graphics programming has a larger advantage. Therefore, if we can combine the graphics in teaching of VC programming, can greatly improve the students' positive and initiative, so as to improve the students' learning efficiency.

Characteristics of the Course

This course involves many subjects, research contents, theory and practice. Because computer graphics is not the core course of computer undergraduate education, most colleges and universities have not given enough attention in the curriculum setting. Elective course is less, and most of the students who just hold elective credits with the idea, to study hard, but due to the limited time, teachers in the classroom can not be on all the algorithms in-depth explanation and explanation only ability of students to understand the principle of the algorithm is simple, only on the limited class time, they failed to thoroughly experience of rigorous, sophisticated algorithm and realization method; and the part of the teachers to teach the course itself did not accumulate enough graphics knowledge, also not engaged in graphics work, some content can only speak generally echo what the books say, explain unclear, some even skip, lead students do not understand, learning interest is not strong weariness. Based on this, in order to let the students learn the real knowledge, we must implement the related algorithms, deepen understanding in practice.

Research and Implementation the Based Algorithms of Computer Graphics

Linear Generation Algorithm. (1) The straight line generation algorithm (SimpleLine): the direct

use of the formula: $y=k*x+b$ generating a straight line. (2) Numerical differential line method (DDALine): an incremental algorithm. Its essence is to use numerical methods to solve differential equations, while the x and y each add a small increment to calculate the next step of the Y , value of $X[1]$. (3) The midpoint line drawing algorithm [1] (MidPointLine), as shown in Fig. 1. If P is the distance from the ideal line recently, the coordinates of P (x_p, y_p). Then, the next and linear nearest pixel only is the right $P_1(x_p+1, y_p)$ or $P_2(x_p+1, y_p+1)$. M is the midpoint of the P_1 and P_2 , then the coordinates of the point M for $(x_p+1, y_p+0.5)$. Suppose Q is the intersection of ideal straight line and vertical $x= x_p+1$. Midpoint line drawing algorithm is to determine the next pixel of P that is P_1 or P_2 , only to see the M is under the Q or over the Q , if M below the Q , P_2 from the line nearly which should take it as the next pixel; if M above the Q , P_1 from the line nearly should take it as the next pixel. If the coincidence between M and Q , the P_1 and P_2 are the distance of the line is equal, the next pixel desirable P_1 is also desirable P_2 .

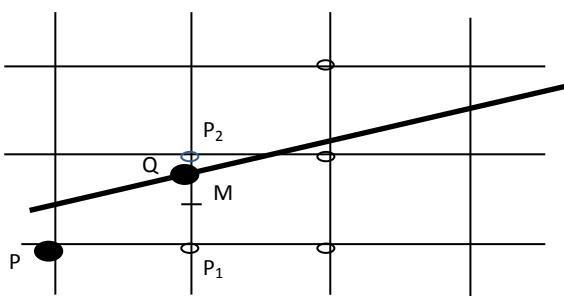


Figure 1. Schematic diagram of the principle of the midpoint line drawing algorithm

(4) (BresenhamLine): Bresenham line algorithm is an accurate and effective raster line generation algorithm which is proposed by Bresenham. In Fig. 1, to determine the next pixel of P is P_1 or P_2 , and only need to calculate the distance of P_1 to Q , it is t , distance of P_2 to Q 's distance is s , by comparing the s and t , if $t > s$. Is that Q is nearest of P_2 that should take the P_2 as the next pixel, otherwise if $t < s$, we can get P_1 as the next pixel; if $t = s$, the next pixel desirable P_1 is also desirable P_2 . The algorithm determines the nearest pixel of the ideal straight line in each column pixel to make a linear scan conversion. Through a point of intersection of rows, each row of pixel center to construct a set of virtual grid line, and then determine the column of pixels in the intersection of the nearest pixel. The clever of this algorithm is that it can be used in incremental calculation, so that for each column, as long as the symbol of the error term is checked, the object of the column can be determined.

Theoretical Contrast. Direct line generating algorithm involves floating point multiplication, addition and subtraction and rounding operation, efficiency is very low. Numerical differential method is faster than the using the formula $y=k*x+b$, but four shed five in operation and floating-point operations are still time-consuming. The midpoint line algorithm only integer arithmetic without multiplication and division, more suitable for hardware implementation. Bresenham line drawing algorithm does not need to calculate the slope of a straight line, so do not have to divide, do not float, only integer; only addition and multiplication 2 operations, the computer is used in the internal displacement operation, high efficiency. Therefore, the Bresenham algorithm is fast, and suitable for hardware implementation.

Experimental Result. All experiments in this paper are in the computer configuration is Intel (R) Core (TM) 2 CPU Duo, 2.0GB RAM, programming environment for VC++ 6. Linear $y=x$ as an example: To compare the efficiency of various algorithms, this paper generates 10000 identical straight lines. The starting point of the straight line is P_1 (60,60), the end point is P_2 (200,200), and the experimental results are shown in Table 1.

Table 1 4 algorithms to generate the same straight line 's results [Unit: Second]

	SimpleLine	DDALine	MidPointLine	BresenhamLine
Run time (3 times)	5.755732	5.728306	5.678694	5.687096
	5.935863	5.695791	5.691730	5.684538
	5.760866	5.713873	5.703679	5.683804
Average time	5.817487	5.712657	5.691368	5.685146

Circle Generation Algorithm. (1) Midpoint circle algorithm (MidPointCircle): assuming the center in coordinate dots, the radius r of round $1/8$ circumference. In Fig. 2, attempt to 1b region ($x = 0$ to $x = y$) as an example. Currently known from the ideal arc the nearest pixel $P(x_p, y_p)$, is a from the arc the nearest pixel only is right-hand $P_1(x_p+1, y_p)$ or right below the point $P_2(x_p+1, y_p-1)$. Order M for the P_1 and P_2 of the midpoint, the M point of the coordinates of $(x_p+1, y_p-0.5)$. If the M in the circle, then P_1 from the arc near, should be taken as the next pixel, or should take P_2 , as shown in Fig. 3. Can put $M(x_p+1, y_p-0.5)$ into the circular equation. we can obtain the discriminant d : $d=F(M)=F(x_p+1, y_p-0.5)=(x_p+1)^2+(y_p-0.5)^2-R^2$, we know that to determine the next pixel, the need to calculate the discriminant d , if $d>0$, then point M outside the circle, should take P_2 as the next pixel, if $d=0$, then point M is in the round, the next pixel desirable P_1 is also desirable P_2 . Of course, when the algorithm can be agreed to take P_2 as the next pixel point. when $d<0$, point M in the circle, should take P_1 as the next pixel.

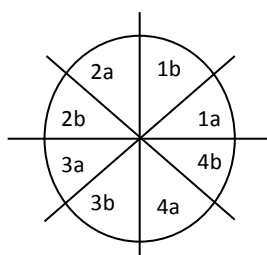


Figure 2. Eight Point Symmetry of Circle

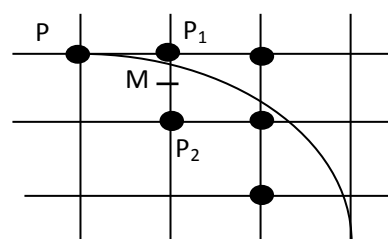


Figure 3. Midpoint Circle Algorithm

(2) Bresenham circle algorithm (BresenhamCircle): In Fig. 2, 1b area as an example. If the intersection of the ideal arc and $P_1 P_2$ is Q , the next pixel of $P(x_p, y_p)$, d_1 and d_2 are distance from P_1 to Q , P_2 to Q , if $d_1 < d_2$, then P_1 is near from the arc, take the positive right of the current pixel points as the next pixel; if $d_1 = d_2$, P_1 and P_2 distance to the arc is the same, arbitrarily choose any of these two points as the next pixel; if the $d_1 > d_2$ that P_2 from near arc, take the current pixel point right below as the next pixel.

(3) Experiment: Compare the efficiency of two algorithms in the generation of Circle, by generating a circle is not much, this paper by using these two algorithms to fill the same size of the inner circle. See Table 2.

Table 2 The 2 Algorithm Running Results of Radius of 900 Filled with its Internal [Unit: Seconds]

The name of the algorithm	MidPointCircle	BresenhamCircle
Run time (3 times)	6.997520	6.691706
	6.750443	6.741651
	6.928204	6.735134
Average time	6.892056	6.722830

Region Filling. Region filling is a fundamental problem in computer graphics and image processing. It has been widely used in industrial automatic detection, pattern recognition, robot vision and image analysis[2][3][4][5][6][7].

Region Filling Algorithm. (1) Bresenham algorithm filled (BreCirFill): using Bresenham circle algorithm for filling a radius r circle. The basic idea is: if the center coordinate is (x_c, y_c) , radius r from zero in order to increase a pixel, and finally, we get the multiple round effect that is the radius on the grounds of zero to r , so the radius r of the circular area has been filled.


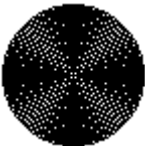


(2) Circle equation filled (CirFillPToC): by judging the distance of the point to the Center of a circle, if the distance is less than or equal to the radius, we can consider the point is inside the circle or in the circle, it is filled the given color, otherwise it is not filled.

(3) Seed filling algorithm (BouCirFill4): Seed filling algorithm divides interior point representation of the seed filling method (known as flood filling) and boundary filling method. Flood filling algorithm starting from a seed point, through the judgment whether the seed pixels and the surrounding pixels constitute a connected region, to decide whether to fill until you find all the pixels in the region or reach the contour boundary [6]. Boundary fill algorithm starts from any point inside of the polygon, according to judgment of adjacent pixels in a certain order, if not a boundary pixel and is not filled, the filling and repeat the process, until all pixel filling is completed. In this paper, 4 connected seed filling algorithm based on boundary representation is adopted.

(4) Based on circular symmetrical equal stipple line algorithm in the circular domain filling [7] (EqLCirFill): r is the radius of a circle, we can divide the circumference into n equal parts, and if n is equal to the perimeter of a circle, each decile point coordinates in order respectively (x_0, y_0) , (x_1, y_1) , (x_2, y_2) , ..., (x_n, y_n) . We can get (x_0, y_0) and (x_c, y_c) which is identified with the center line as the axis of symmetry, the left and right ends of the circular aliquots of symmetric point respectively as the starting point and end point of line, then draw a straight line. The figure that is obtained is the inner region of the circle.

Experimental Result. In this paper, a circle with a radius of 35 is used as an example to fill in the above four algorithms. The results of the operation are shown in Table 3.

Table 3 Operating results of the 4 algorithms when the radius is 35 [Unit: Second]

The name of the algorithm	BouCirFill4 EqLCirFill	BreCirFill	CirFillPToC	
Run time (3 times)	5.769727	0.036359	0.032798	
	0.001942			
	5.775719	0.036666	0.031999	
	0.001915			
Average time	6.074390	0.036678	0.032234	
	0.001921			
Average time	5.873279 0.001926	0.0365677	0.032344	
Operation result				
Remarks	Full fill	There are a lot of leak	Full fill	Full fill

Curve Generation. (1) Bezier Curve Definition. For a given position vector of space of $n+1$ points, P_i ($i=0,1,2,n$), then the Bezier parameter curve of each point on the coordinates of the interpolation formula for [8]:

$$P(t) = \sum_{i=0}^n P_i B_{i,n}(t) \quad (t \in [0,1])$$

In the formula, P_i constitute the characteristic polygon of the Bezier curve, $B_{i,n}(t)$ is the n curve basis function of Bernstein:

$$B_{i,n}(t) = C_n^i t^i (1-t)^{n-i} = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i} \quad (i = 0, 1, \dots, n)$$

In the formula, $0^0=1, 0!=1$ 。

(2) Bezier curve implementation: This paper takes the three Bezier curve generation as an example. When $n=3$, The vertices of the characteristic polygon are P_0, P_1, P_2, P_3 , we can define a three Bezier curve, the form of curve definition is as follows:

$$P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3$$

$$P(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

Matrix form:

Assuming that the coordinates of each vertex of the characteristic polygon are: $P_0(50,50), P_1(150,160), P_2(240,80), P_3(300,200)$, the generated three Bezier curves are shown in Fig. 4.

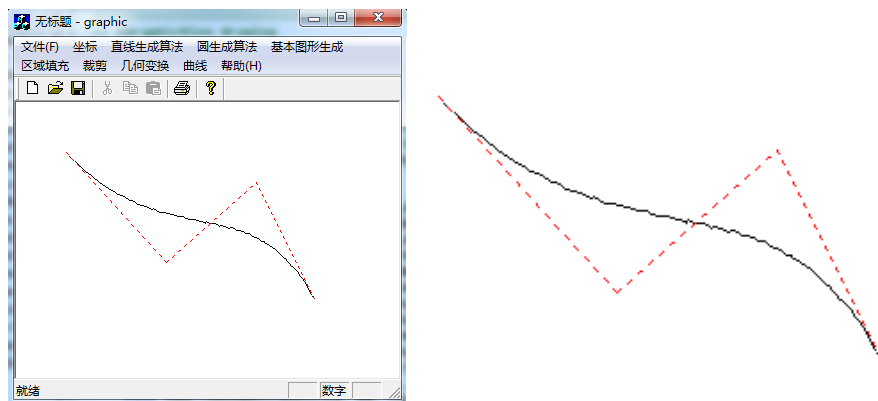


Figure 4. Three Bezier Curves

Conclusion

In this paper, we compare and analyze various graphics algorithms and programmed by VC, and we implemented the basic algorithm, completed the program design language to graphical images revealed that the transformation, we developed the graphics experiment platform, in certain procedures which can meet the demand of computer graphics, has provided the condition for the combination development of late algorithm, has a certain theoretical and practical significance.

References

- [1] Yin Hongxia, Du Sichun, Cai Lijun. Computer graphics [M]. Beijing: China Water Conservancy and Hydropower Press, 2005.
- [2] Rosenfeld A, Pfaltz J L. Sequential operations in digital picture processing [J]. Journal of the ACM, 1966, 13(4): 471-494..

- [3] Du Jianjun, Guo Xinyu, Sheng Lian Lu etc. Based on optimal neighborhood region filling algorithm [J]. China stereo and image analysis. 2013, 18 (2): 109-114.
- [4] Xu shengpan, Liu Zhengjun, ZuoZhiquan. Improved active-edge-table area filling algorithm [J]. Computer Engineering and Applications, 2014, 50(17):178-181.
- [5] Zhang Yikuan. Computer graphics [M]. Xi'an: Xi'an Electronic and Science University press, 2006:58-64.
- [6] Zhang Yi, Li Changhua. Filling algorithm for arbitrary shape based on boundary tracking [J]. Computer engineering and design, 2015, 36(3):725-728.
- [7] Du Jianjun, Guo Xinyu, Lu Shenglian. A region filling algorithm based on optimal neighborhood relativity [J]. Chinese journal of stereology and image analysis, 2013, 18(2):109-114.
- [8] Wang Jiexiong, Chen Guodong, Chen Yi. Liver CT image segmentation algorithm based on improved B-Snake model [J]. computer engineering and application,2015, 51 (9) :152-157.
- [9] Li Honglin, Liu Kun. Circle area filling based on the line drawing algorithm of circle symmetrical equidistant points [J]. Journal of Yunnan Nationalities University: Natural Science Edition, 2013, 22 (4) 292-294.
- [10] Fu Yuchen, Zhou Dong Ru. Computer graphics: principle, method and application. Wuhan: Huazhong University of Science and Technology press, 2003248-249.