Algorithm for Probability Calculation of B-Spline Curve

Jiaqi Qiu¹, Yi Xie²*

¹School of International, GuangXi University, GuangXi, China
²School of Computer & Information Science, Southwest University, Chongqing, China

*Corresponding author e-mail: 641644938@qq.com

Abstract. This paper presents a probability calculation method for B-spline curve, which is made up of coordinate transformation unit, normalization unit, binarization unit, probability calculation unit, data decoding unit. This thesis offers the general structure and algorithm principle of generating B-spline curve, and draws the B-spline curve on the simulation hardware of VC++ 6.0 software. Experimental results show that compared with the traditional method of directly calculating B-spline, Paper's method has lower algorithm complexity and faster curve generation, which is more beneficial to the realization of high-order B-spline curve.

Keywords: Pseudo Random Code, Probability Calculation, B-Spline Curve

1. Introduction
Probability calculation [1] is a numerical representation and calculation method. In probability calculation, the ratio of the number of 1 in the sequence to the length of the whole sequence is used to represent the value, and the multiplication and addition operations are completed by constructing a simple gate circuit. In recent years, the method of probability calculation has been applied to digital filter, FFT, Turbo decoder, multi-bit rate LDPC decoder and other research fields [2]. Using probability calculation can reduce computation, power consumption and cost.

B-spline curve is very widely used, it is useful in many ways [3-5]. When the traditional direct calculation method is used to calculate and generate the higher-order B-spline curve, the complexity is too high and the speed is slow. The method of probability calculation is used to directly generate the B-spline curve of higher order, which can improve the speed of curve generation in various applications.

2. Algorithm analysis
Figure 1 shows the work-flow of the entire system. Including coordinate transformation unit, normalization unit, binarization unit, probability calculation unit, data decoding unit. Data processing process: the data input includes the data point coordinates of B-spline curve and the value of ρ. The value of ρ is between 0 and 1, and Direct input to the binary unit. The data point coordinates are first transformed by the coordinate change unit, then converted into numbers between 0 and 1 by the normalization unit, and then converted into binary string by the binarization unit. The probability
calculation unit is input for calculation, and then decoded by the data decoding unit, and then the B-spline curve is drawn.

2.1. calculation of B-spline curve

\[ \Gamma(\tau) = \sum_{\theta=0}^{\theta=n} \left( \frac{1}{n!} \sum_{j=0}^{n-j} \left( -1 \right)^{j} C_{n-j}^{j} \left( \tau + n - \theta - j \right)^{j} \right) \quad 0 \leq \tau \leq 1 \]  

(1)

Formula (1) is the formula for calculating B-spline curve for n times, where \( \Gamma_{0}, \Gamma_{1}, \ldots \Gamma_{n} \) are the data points of the curve. n is the order of B-spline curve; the n of Third order B-spline is 3; \( \tau \) is a decimals value between 0 and 1; \( \theta \) is the \( \theta^{th} \) point, and the value range of \( \theta \) is 0, 1, 2,...n.

2.2. Coordinate Transformation unit

Since the data points in formula (1) are not convenient for direct probability calculation, coordinate transformation is needed. Take Third order B-spline curve as an example to carry out formula transformation. The formula for calculating Third order B-spline curve is expanded as follows:

\[ \Gamma(\tau) = \frac{1}{6}(\Gamma_{0}(-\tau^{3} + 3\tau^{2} - 3\tau + 1) + \Gamma_{1}(3\tau^{3} - 6\tau^{2} + 4) + \Gamma_{2}(-3\tau^{3} + 3\tau^{2} + 3\tau + 1) + \Gamma_{3}\tau^{3}) \]  

(2)

Formula (3) can be obtained from formula 2:

\[ \Gamma(\tau) = \frac{1}{6}((\Gamma_{0} + 4\Gamma_{1} + \Gamma_{2}) + (-3\Gamma_{0} + 3\Gamma_{2})\tau + (3\Gamma_{0} - 6\Gamma_{1} + 3\Gamma_{2})\tau^{2} + (-\Gamma_{0} + 3\Gamma_{1} - 3\Gamma_{2} + \Gamma_{3})\tau^{3}) \]  

(3)

Convert formula (3) to formula (4). Formula (5) and (6) can be obtained by calculating the coordinates of corresponding data points.

\[ \Gamma(\tau) = H_{0}C_{3}^{0}\tau^{0}(1 - \tau)^{3} + H_{1}C_{3}^{1}\tau^{1}(1 - \tau) + H_{2}C_{3}^{2}\tau^{2}(1 - \tau)^{1} + H_{3}C_{3}^{3}\tau^{3}(1 - \tau)^{0} \]  

(4)

\[ H_{0} = \frac{1}{6}(\Gamma_{0} + 4\Gamma_{1} + \Gamma_{2}) \quad H_{1} = \frac{1}{6}(4\Gamma_{1} + 2\Gamma_{2}) \]  

(5)

\[ H_{2} = \frac{1}{6}(2\Gamma_{1} + 4\Gamma_{2}) \quad H_{3} = \frac{1}{6}(\Gamma_{1} + 4\Gamma_{2} + \Gamma_{3}) \]  

(6)

Input data point coordinates: \( \Gamma_{0}, \Gamma_{1}, \Gamma_{2}, \Gamma_{3} \). After formula (5) and (6) are calculated, obtained the data point coordinates \( H_{1}, H_{2}, H_{3}, H_{4} \). Then do the following probability calculations.

2.3. Normalization Unit

The normalization unit converts the value mapping of data point \( H_{0}, H_{1}, H_{2}, \ldots, H_{n} \) to the range of [0,1]. The normalization mapping conversion formula is as follows:

\[ GH_{i} = (H_{i} - \zeta(H_{0}, H_{1}, H_{2}, \ldots, H_{n}))/((\psi(H_{0}, H_{1}, H_{2}, \ldots, H_{n}) - \zeta(H_{0}, H_{1}, H_{2}, \ldots, H_{n})),(i = 0, 1, 2, \ldots n) \]  

(7)
\( \zeta(.) \) is the minimum function, and \( \psi(.) \) is the maximum function. After normalization, the value of \( GH_i \) is between [0,1].

2.4. Binarization Unit

Binarization unit converts \( \tau_\theta \) and \( GH_i \) into binary series \( L_i \) and \( E_i \) respectively, which is composed of pseudo-random code generation module, comparator, input and output. When the input is \( \tau_\theta \), the output is \( L_i \); When input \( GH_i \), output \( E_i \).

The pseudo-random code generation module is generated by the shift of the original polynomial[6]. Pseudo random code is a random character in which the number of zeros and the number of ones are almost equal. Pseudo-random codes are generally generated by primitive polynomials, Formula (8) represents the original polynomial:

\[
d(L) = C_0 + C_1L + C_2L^2 + C_3L^3 + \ldots + C_nL^n
\]  

(8)

In this paper, \( n \) is set as 10, and feedback shift of 110000101 is used to generate \( n \)-bit random code \( R_i \). In the experiment, \( n \) is set as 256 bits, 512 bits, 1024 bits and 2048 bits respectively.

![Figure 2](image)

**Figure 2.** Binarization unit generation diagram

In figure 2, the shift comparison between \( GH_i \) and \( R_i \) (\( i=0,1,2,\ldots,n \)) is carried out. If \( GH_i \) is greater than or equal to \( R_i \), the output is one else the output is zero. The value of \( GH_i \) is equal to the ratio of the number of ones in the binary string of \( E_i \). If the value of \( GH_i \) is 0.6 and \( N \) is 256 bits, then the number of ones in \( E_i \) is 256*0.6. Similarly, after the shift comparison between \( \tau_\theta \) and \( R_i \) can obtain a random binary string \( L_i \).

![Figure 3](image)

**Figure 3.** Probability Calculation unit

\begin{align*}
\text{Pseudo random code generation module} & \rightarrow \text{Pseudo random code} \rightarrow \text{The comparator} \\
\text{Primitive polynomial} & \rightarrow \text{Pseudo random code generation module} \rightarrow \text{input} \rightarrow \text{output}
\end{align*}
2.5. Probability Calculation Unit

The probability calculation unit is composed of adder and multiplexer.

If it is n times B-spline, then the shift comparison between $r_i$ and $R_i$ is carried out, and n binary series $L_1, L_2, ... L_n$ are generated. $L_1, L_2, ... L_n$ are added to each other to get $S(1)S(2) ... S(N)$, where $S(i) = L_i(i) + L_2(i) + ... + L_n(i)$ $(i=0,1,2,...,n)$. $S(1)S(2) ... S(N)$ as the control input of the multiplex gate, and $E_0, E_1, ... E_n$ as the coefficient input, output $P(i) = E_{S(i)}(i = 0,1,2,...N)$. And we end up with the output value $(P(1)P(2) ... P(N))$. The truth table calculated by $E_{S(i)}$ is shown in table 1:

| The input signal E | Gating signal S | The output signal P |
|--------------------|----------------|---------------------|
| $E_0(1)E_0(2)E_0(3)...E_0(N)$ | 0               | $E_0(t)$            |
| $E_1(1)E_1(2)E_1(3)...E_1(N)$ | 1               | $E_1(t)$            |
| ...                | ...             | ...                |
| $E_n(1)E_n(2)E_n(3)...E_n(N)$ | n               | $E_n(t)$            |

2.6. Data Decoding Unit

The data decoding unit converts the output binary string into the final point position coordinate. The conversion is calculated by formula 9 and formula 10.

$$ P = \left(\frac{\sum_{i=1}^{N} P(i)}{N}\right) $$  (9)

Formula (9) is the calculation formula of random decoding, which calculate the ratio of the number of ones in $(P(1)P(2) ... P(N))$ in binary string. Then through the inverse process of normalization of formula (10), the position coordinate of the point is finally obtained.

$$ DP = P(\psi(H_0,H_1,H_2,...H_n) - \zeta(H_0,H_1,H_2,...H_n)) + \zeta(H_0,H_1,H_2,...H_n) $$  (10)

$DP$ is the output coordinate value associated with the input value $r_i$ and the data point $H_1,H_2,H_3,H_4$ obtained by probability calculation. $\zeta(.)$ is the minimum function, and $\psi(.)$ is the maximum function. The value of $DP$ can be used to plot the B-spline curve directly.

3. Experimental results and analysis

3.1. Experimental design

The experimental environment is VC++6.0. This paper use formula (1) to directly calculate and draw the curve of the third order B-spline curve, and compare with the method in the paper. The result is shown in the Figure 4. The two curves basically coincide. The code number used in the paper is 512 bits.(solid lines are drawn by formula (1), and dashed lines are drawn by probability calculation)

3.2. Experimental results and error analysis

Table 2 is a comparison of specific values of each point. The data in the table is the calculated value of partial $r_i$ of Third order B-spline curve.
As can be seen from Table 2, there are some errors between the method of paper and the result of direct formula (1), but the errors are less. We continue to calculate the probability of 256 bits, 512 bits, 1024 bits and 2048 bits pseudo-random codes respectively, and the calculated results are shown in Table 3.

![Figure 4. Screenshot of experimental results](image)

**Table 2.** Result of probability calculation method for 512 bit pseudo-random code.

| \( \tau_\theta \) | The probability calculates for value of X | The direct calculation for value of X | The difference of X | The probability calculates for value of Y | The direct calculation for value of Y | The difference of Y |
|---|---|---|---|---|---|---|
| \( \theta_1 = 0.1 \) | 0.6640 | 0.6547 | 0.0093 | 0.0937 | 0.0941 | 0.0004 |
| \( \theta_1 = 0.3 \) | 0.7285 | 0.7420 | 0.0135 | 0.1347 | 0.1252 | 0.0095 |
| \( \theta_1 = 0.5 \) | 0.8144 | 0.8160 | 0.0016 | 0.1679 | 0.1753 | 0.0074 |
| \( \theta_1 = 0.7 \) | 0.8964 | 0.8767 | 0.0197 | 0.2324 | 0.2430 | 0.0106 |
| \( \theta_1 = 0.9 \) | 0.9218 | 0.9240 | 0.0022 | 0.3125 | 0.3272 | 0.0147 |

**Table 3.** Probability calculation results of B spline curve with different length pseudo-random code

| Pseudo-random code number | Average difference of X | Average difference of Y | Composite average difference |
|---|---|---|---|
| 256 bits | 0.1895 | 0.1888 | 0.1891 |
| 512 bits | 0.0715 | 0.0749 | 0.0732 |
| 1024 bits | 0.0578 | 0.0762 | 0.0670 |
| 2048 bits | 0.0451 | 0.0479 | 0.0465 |

In Table 3, it can be seen that since it is a probability calculation, the more digits, the more accurate. As the number increases, the error gets smaller and smaller. In practice, if you want to improve the accuracy, you can choose more bits. The drawing results of different digits are shown in figure 5 (solid lines are drawn by direct calculation, and dashed lines are drawn by probability calculation)
3.3. Performance analysis

In formula (1), multiplication and addition operations are needed. The higher the order of B-spline, the more complex the formula is and the longer the calculation time is. The time complexity of n order B-spline is $O(n^t)$.

The calculation of the method in the paper consists of adder, comparator, multiplex gate selector and so on. As can be seen from Figure 3, 6 comparators, 1 adder and 1 4-choice multiplexer are needed for Third order B-spline curves. Four B-spline require eight comparators, 1 adder, and 1 five-choice multiplexer. The n times B-spline curve requires 2n comparators, 1 addition and n+1 multiplexer. Therefore, with the increasing number of B-spline curves, the hardware required for probability calculation increases linearly and the computational complexity is low.

4. Conclusion

This paper proposes a method to realize B-spline curve, which has the advantages of low computational complexity, small computational amount and fast computational speed, and is easy to be implemented on FPGA and other hardware.

References

[1] Jianhao Hu, Jienan Chen, Application of Probability Calculation in the Realization of Communication Signal Processing System [J]. Radio Communications Technology, 41 (2) : 01-06, 2015.

[2] Hanshan LI, Xiaqian Zhang, Laser echo characteristics and detection probability calculation on the space projectile proximity fuze [J]. Optik, v 183, p 713-722, April 2019.

[3] Mirzaee, Farshid, Etc., Cubic B-spline approximation for linear stochastic integro-differential equation of fractional order [J]. Journal of Computational and Applied Mathematics, v 366, 1 March 2020.

[4] Mirzaee, Farshid, An efficient cubic B-spline and bicubic B-spline collocation method for numerical solutions of multidimensional nonlinear stochastic quadratic integral
equations[J]. Mathematical Methods in the Applied Sciences, v 43, n 1, p 384-397, January 15, 2020.

[5] Jie Liu, Jingsong Xie, Bing Li, Bingbing Hu, Regularized cubic b-spline collocation method with modified L-curve criterion for impact force identification[J]. IEEE Access, v 8, p 36337-36349, 2020.

[6] ShaoKun Liu, Ping Li, HongHai Yu, Design of Jamming Signal on Pseudo-random Code Phase-modulation and Pulse Doppler Combined Fuze Based on Code Reconstruction[J]. Binggong Xuebao/Acta Armamentarii in Chinese, v 39, n 6, p 1088-1094, June 1, 2018.