SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES
VOL. 5, NO. 3, DECEMBER 2022
DOI: 10.35377/saucis.05.03.1122506

Research Article

Software Development for the Use of Generalized Parabolic Blending in Data Prediction Processes

Hakan Üstünel[1]

[1] Corresponding Author; Department of Software Engineering, Faculty of Engineering, Kirklareli University, Kirklareli, Türkiye; hakanustunel@hotmail.com; hakanustunel@klu.edu.tr

Received 27 May 2022; Revised 26 September 2022; Accepted 18 November 2022; Published online 31 December 2022

Abstract

Parabolic blending (PB) is one of the important topics in applied mathematics and computer graphics. The use of generalized parabolic blending (GPB) for different scenarios adds flexibility to the polynomial. Overhauser (OVR) elements is a special case in GPB (r=0.5, s=0.5). GPB can also be used in estimation. In this study, data obtained from thickness distribution of a 3mm thick high impact polystyrene product after thermoforming using a mold was used for data estimation. For this purpose, software has been developed. The software development steps and formula usages are explained. Using the developed software, polynomials for GPB and default PB (OVR) were created. The data set was compared with the y values produced by the polynomials for certain x values. At the end of the research, it was determined that the results obtained from the GPB were 0.1728 percent more accurate than the data obtained from the PB for the default values.

Keywords: Generalized parabolic blending, computer graphics, visualization, software development, thickness distribution

1. Introduction

Curves have been an important subject of mathematics and applied sciences for hundreds of years. It has been possible for curves to be applied in many different fields and to be defined in different ways in this old subject. In its general definition, it is possible to define a curve with at least a quadratic function.

A curve can also be created by defining a function for the curve using existing vertices (control points, vectors). The basic approach in this matter can be shown by defining the smallest degree polynomial passing through all control points, as in the Lagrangian interpolation polynomial or the Newton divided difference polynomial. Parabolic curves also find a place in the areas of practical solutions of theoretical calculations such as multi-segment trajectory tracking of the redundant space robot for smooth motion [1], free vibration analysis of a variable stiffness composite laminate square plate with circular cutout [2] and describing the viscoelastic behavior of a shape memory polymer blend [3].

Curves can also be defined using the parabolic blending (PB) approach by following a different path than curve interpolation. PB, which is preferred especially in places where smooth transition should occur, is an important subject of computer graphics. Overhauser [4] created the primary definitions of PB. In later studies, PB was also referred to as Overhauser elements, Overhauser's parabolic blending interpolation, or Overhauser's curve [5-7]. Brewer & Anderson [8] presented the applications of PB in computer graphics. Schneider [9] has worked on adding tension to the formula. PB can be applied in boundary conditions [10-14] as Boundary Element Method (BEM). PB has also been an important field of study in the estimation process [18].

Having four control vertices/points/vectors in PB is sufficient to define the curve. The main difference between PB and the interpolation polynomial generation process is that the weights of the control vertices on the curve are calculated in different ways. When the interpolation polynomial is constructed for the curve passing through a multi-element series of vertices, the polynomial will have a high degree. This will cause cumbersome operations, especially for programs with multiple repetitions.

Hadavinia et al. [17] named their work as general parabolic blending (GPB) elements by formulating the variation of the parameters used to give the general form of the PB and corresponding to the intermediate vertices. Thus, Overhauser (OVR) elements are shown as a special case of GPB.They claimed that the results obtained with the GPB elements would be more accurate than the OVR, and they formulated the parameter variation for the intermediate vertices in its general form. They supported their claims with the results they obtained from the sample functions they had used in their studies.

In this study, the notation that Hadavinia et al. [17] call GPB is explained with examples for different parameters. Afterwards, multi-repetitive software was developed to be used in the estimation process to be run on a real dataset. The results obtained are visualized to facilitate comparison.

2. Parabolic Blending for Changing Parameters

The process of creating a cubic between two parabolas is called PB [17,18] (also called OVR elements). Three vertices are selected from four control vertices (P_1-P_4) to form two parabolas (Figure 1). P_{1-3} defines the P(r) parabola, while P_{2-4} defines the Q(s) parabola [4,18].

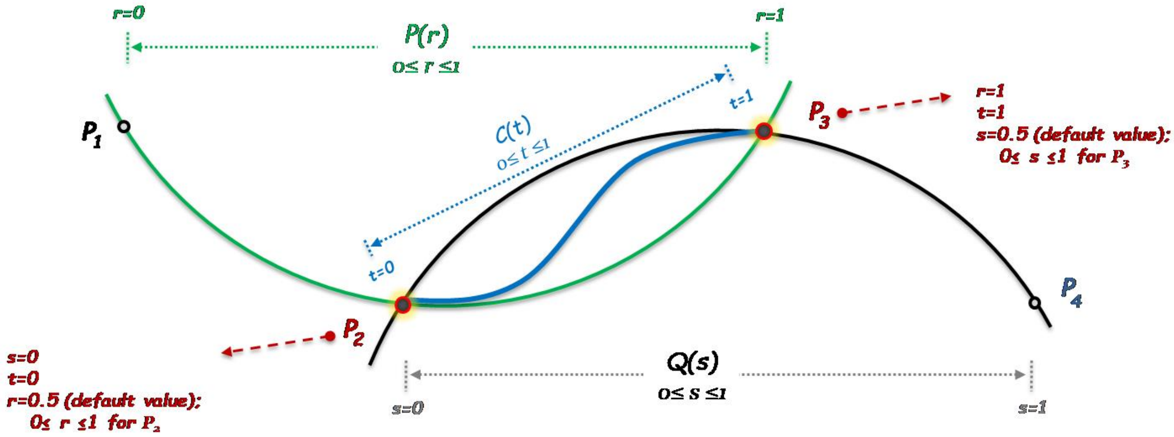

Figure 1 Parabolas, parameters and control vertices that define parabolic blending

With the effect of the weights of these two parabola, C(t) is given its general form. The effect of P(r) and Q(s) on C(t) is linear (line equation) (Formula 1).

$$\boldsymbol{C(t) = (1-t)P(r) + t * Q(s)} \tag{1}$$

In Formula 1, the weight of the parabolas on C(t) changes depending on the value of t (0≤ t ≤1). For the value of t 0.5, the weights of the parabolas on C(t) are equal. If the t value is less than 0.5, the effect of P(r) on the curve is greater than that of Q(s). If the t value is greater than 0.5, the effect of P(r) on the curve is less than that of Q(s). Depending on the t value, any vertices on C(t) (x,y) can also be reached depending on the control vertices. Equations of parabolas P(r) and Q(s) must be formed.

If the general P(r) equation is created depending on the r values, Formula 2 is obtained (0≤ r ≤1). [B] is the coefficient matrix to be calculated based on the control vertices where the r parameter is equalized.

$$P(\boldsymbol{r}) = [\boldsymbol{r^2} \quad \boldsymbol{r} \quad \boldsymbol{1}][\boldsymbol{B}] \tag{2}$$

At this stage, the r value is used to give P(r) its general form. If the r value is 0, the P_1 is obtained, and if the r value is 1, the P_3 is obtained. Any r value between these two vertices is equated with the P_2 to form the parabola equation. The value of r for P_2 will give P(r) its general form. The default r value for P_2 is 0.5 in parabolic blending.

$$\boldsymbol{r = 0 \quad => \quad P(0) = P1;} \qquad \boldsymbol{P1 = [0^2 \quad 0 \quad 1][B]}$$

$$\boldsymbol{r = 0.5 \quad => \quad P(0.5) = P2;} \qquad \boldsymbol{P2 = [0.5^2 \quad 0.5 \quad 1][B]}$$

$$\boldsymbol{r = 1 \quad => \quad P(1) = P3;} \qquad \boldsymbol{P3 = [1^2 \quad 1 \quad 1][B]}$$

Formula 3 occurs when the accepted r values for control vertices are combined to the side. If the r value was 0.3 for P_2, the row with index 1 of the matrix would be [0.32 0.3 1].

$$\begin{bmatrix} \boldsymbol{P1} \\ \boldsymbol{P2} \\ \boldsymbol{P3} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{0.25} & \mathbf{0.5} & \mathbf{1} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} \end{bmatrix} [\boldsymbol{B}] \tag{3}$$

If the matrix that creates the r values for the selected control vertices is called [Mp] and then [B] is left alone, Formula 4 is obtained.

$$\begin{bmatrix} \boldsymbol{P1} \\ \boldsymbol{P2} \\ \boldsymbol{P3} \end{bmatrix} = [\boldsymbol{Mp}][\boldsymbol{B}]; \qquad [\boldsymbol{B}] = [\boldsymbol{Mp}]^{-1} \begin{bmatrix} \boldsymbol{P1} \\ \boldsymbol{P2} \\ \boldsymbol{P3} \end{bmatrix} \tag{4}$$

If Formula 4 is substituted in Formula 2, Formula 5 is obtained.

$$\boldsymbol{P(r)} = [\boldsymbol{r^2} \quad \boldsymbol{r} \quad \mathbf{1}][\boldsymbol{Mp}]^{-1} \begin{bmatrix} \boldsymbol{P1} \\ \boldsymbol{P2} \\ \boldsymbol{P3} \end{bmatrix} \tag{5}$$

As can be seen, depending on the r value at the P_2, [Mp] and therefore $[Mp]^{-1}$ will change, so the P(r) equation will also change (Formula 5). This change will also affect the general form of C(t) (Formula 1).

While creating the P(r) equation, it has been accepted that the P_2 will be reached if the r value is 0.5 (default state for PB). Until this stage, the equation will change, since [Mp] values will change for different values of the r value. For example, if P_2 corresponds to r=0.2 or r=0.7 value instead of 0.5 value of r;

$$[\boldsymbol{Mp}] = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{0.04} & \mathbf{0.2} & \mathbf{1} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} \end{bmatrix} \qquad [\boldsymbol{Mp}] = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{0.49} & \mathbf{0.7} & \mathbf{1} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} \end{bmatrix}$$

Formula 6 is obtained when the operations performed in P(r) are repeated for the vertices $P_{2\text{-}4}$ to calculate the Q(s) equation. While creating the Q(s) equation, it has been accepted that P_3 will be reached if the s value is 0.5 (default state for PB). Similar to P(r), it matters for the general form of Q(s) at which P_3 is reached, versus which value of s (0≤ s ≤1).

$$\boldsymbol{Q(s)} = [\boldsymbol{s^2} \quad \boldsymbol{s} \quad \mathbf{1}][\boldsymbol{Mq}]^{-1} \begin{bmatrix} \boldsymbol{P2} \\ \boldsymbol{P3} \\ \boldsymbol{P4} \end{bmatrix} \tag{6}$$

If Formula 5 and Formula 6 are substituted in Formula 1, Formula 7 is obtained.

$$\boldsymbol{C(t)} = (\mathbf{1} - \boldsymbol{t}) * [\boldsymbol{r^2} \quad \boldsymbol{r} \quad \mathbf{1}][\boldsymbol{Mp}]^{-1} \begin{bmatrix} \boldsymbol{P1} \\ \boldsymbol{P2} \\ \boldsymbol{P3} \end{bmatrix} + \mathbf{t}^*[\boldsymbol{s^2} \quad \boldsymbol{s} \quad \mathbf{1}][\boldsymbol{Mq}]^{-1} \begin{bmatrix} \boldsymbol{P2} \\ \boldsymbol{P3} \\ \boldsymbol{P4} \end{bmatrix} \tag{7}$$

After Formula 7 is created, the value of r and s must be calculated for any value of t (for vertices where P(r) and Q(s) intersect with C(t); P_2, P_3). The variation of r with respect to t is their line equation (Formula 8).

$$\boldsymbol{r} = \boldsymbol{k}\mathbf{1} * \boldsymbol{t} + \boldsymbol{k}\mathbf{2} \tag{8}$$

In Formula 5, the k_1 and k_2 values must be calculated. If P_2 is reachable for the default value (0.5) of r, Formula 9 is obtained.

$$\boldsymbol{P2}\text{:}\ \boldsymbol{r=0.5};\qquad \boldsymbol{t=0 =>}\quad \boldsymbol{r=k1*t+k2}\qquad \boldsymbol{0.5=k1*0+k2}\quad \boldsymbol{=> k2=0.5}$$

$$\boldsymbol{P3}\text{:}\ \boldsymbol{r=1};\qquad \boldsymbol{t=1 =>}\quad \boldsymbol{r=k1*t+k2}\qquad \boldsymbol{1=k1*1+0.5}\quad \boldsymbol{=> k1=0.5}$$

$$\boldsymbol{r(t)=0.5*t+0.5} \tag{9}$$

If P_2 is reachable for the 0.3 of r, Formula 10 is obtained.

$$\boldsymbol{P2}\text{:}\ \boldsymbol{r=0.3};\qquad \boldsymbol{t=0 =>}\quad \boldsymbol{r=k1*t+k2}\qquad \boldsymbol{0.3=k1*0+k2}\quad \boldsymbol{=> k2=0.3}$$

$$\boldsymbol{P3}\text{:}\ \boldsymbol{r=1};\qquad \boldsymbol{t=1 =>}\quad \boldsymbol{r=k1*t+k2}\qquad \boldsymbol{1=k1*1+0.3}\quad \boldsymbol{=> k1=0.7}$$

$$\boldsymbol{r(t)=0.7*t+0.3} \tag{10}$$

For different values of r, Formula 8 will change. If a pattern is extracted using Formula 8, Formula 9 and Formula 10, the equation k_1=1-r and k_2=r can be formed. In this way, Formula 11 is obtained when Formula 5 is generalized.

$$\boldsymbol{r(t)=(1-r)*t+r} \tag{11}$$

With r(t) in Formula 11, an instant r value depending on the t value is obtained. It is the position of P_2 that gives its general form to P(r), with the r parameter on the right side of the equation. In order not to confuse the terms in the next stages, $r_{\text{instant(t)}}$ will be used instead of r(t) and after the equality is solved, the value obtained is called rinstant and the formula is rearranged (Formula 12).

$$\boldsymbol{rInstant(t)=(1-r)*t+r} \tag{12}$$

As with the t-dependent change in the r value, the t-dependent change in the s value is also a line equation (Formula 13).

$$\boldsymbol{s=k3*t+k4} \tag{13}$$

In Formula 13, the k_3 and k_4 values must be calculated. If P_2 is reachable for the default value (0.5) of s, Formula 14 is obtained.

$$\boldsymbol{P2}\text{:}\ \boldsymbol{s=0};\qquad \boldsymbol{t=0 =>}\quad \boldsymbol{s=k3*t+k4};\qquad \boldsymbol{0=k3*0+k4}\quad \boldsymbol{=> k4=0}$$

$$\boldsymbol{P3}\text{:}\ \boldsymbol{s=0.5};\qquad \boldsymbol{t=1 =>}\quad \boldsymbol{s=k3*t+k4}\qquad \boldsymbol{0.5=k3*1+0}\quad \boldsymbol{=> k3=0.5}$$

$$\boldsymbol{s(t)=0.5*t} \tag{14}$$

If P_2 is reachable for the 0.1 of s, Formula 15 is obtained.

$$\boldsymbol{P2}\text{:}\ \boldsymbol{s=0};\qquad \boldsymbol{t=0 =>}\quad \boldsymbol{s=k3*t+k4};\qquad \boldsymbol{0=k3*0+k4}\quad \boldsymbol{=> k4=0}$$

$$\boldsymbol{P3}\text{:}\ \boldsymbol{s=0.1};\qquad \boldsymbol{t=1 =>}\quad \boldsymbol{s=k3*t+k4}\qquad \boldsymbol{0.1=k3*1+0}\quad \boldsymbol{=> k3=0.1}$$

$$\boldsymbol{s(t)=0.1*t} \tag{15}$$

For different values of s the Formula 13 will change. If a pattern is extracted using Formula 13, Formula 14 and Formula 15, the equation k_3=r and k_4=0 can be formed. In this way, Formula 16 is obtained when Formula 13 is generalized.

$$\boldsymbol{s(t)=s*t} \tag{16}$$

With s(t) in Formula 16, an instant s value depending on the t value is obtained. It is the position of P_2 that gives its general form to Q(s) with the s parameter on the right side of the equation. In order not to confuse the terms in the next stages, $s_{\text{instant(t)}}$ will be used instead of s(t) and after the equality is solved, the value obtained is called sinstant and the formula is rearranged (Formula 17).

$$\boldsymbol{sInstant(t)=s*t} \tag{17}$$

As a result, it is first necessary to determine which r and s values will be used to reach the vertices P_2 and P_3, respectively. Since these values will change the values of the matrices in the formula for P(r) and Q(s) (Formula 5, Formula 6), the weights of the parabolas on C(t) will also change. In the next step, the rinstant and sinstant values for a given t value will be calculated using generalized formulas (Formula 12, Formula 17), and should be written into the general formula (Formula 7).

3. Methodology

In this section, the dataset and the pattern used in the research and the purpose of the research are mentioned. The dataset of this study was taken from a previous study, in which the author of this study was one of the researchers [18]. This dataset was created as a result of measuring the thickness distribution of a 3mm thick high impact polystyrene product after thermoforming using a mold (Column 2 in Table 2). In a study by Ekşi and Üstünel [16], PB was used to estimate the thickness distribution of the thermoformed plate. With the coded program, correct estimation was performed with a 4.3866 percent error value. In the same study, while determining the PB steps, P_2 and P_3vertices were accessed for the 0.5 value of the r and s parameters; that is, the default values were used. In this research, while determining the PB steps, instead of a fixed value such as 0.5 for the r and s parameters, access to the P_2 and P_3 vertices was provided for the values in a varying range depending on a certain precision value. This process can be expressed as optimizing the PB process depending on the r and s values that change independently of the t value. In order to compare the results obtained in the optimization of the PB with the results of the previous research [16], the same dataset and pattern were used in this study.

The aim of this research is to ensure that more accurate results can be produced in the estimation processes by optimizing PB. For this purpose, software has been developed, and its steps are explained. Afterwards, the results of the error calculations performed using the software were examined. In C(t), the value of y is calculated based on the calculated x value for any value of t. Then, the error value is calculated by comparing this data with the dataset([16], Table 3 Column 2-4) regarding the thickness distribution of the thermoformed material. In this study, after this step, the case where the r and s parameters are 0.5 will be called the default PB (instead of OVR elements). In addition, the situation in which C(t) also changes with each change of the r and s parameters will be called GPB, depending on the nomenclature of Hadavinia et al. [17].

The flow chart of the developed program is shown in Figure 2. Under this heading, flowchart sections are explained. In Section 1, variable definitions and value assignments, especially precision, are carried out. The program has been implemented as a Visual Studio 2019 C# form application. The results obtained from the calculations were transferred to an spreadsheetfile by the ptogram. The output file (spreadsheet file) is processed in MatLAB® in the next step, and the results are finally visualized.

The developed software mainly consists of the following parts. Assigning the initial values (section 1), selecting the precision values to be used in the calculations (section 2), performing the assignments for the P(r) values (Section 3), performing the assignments for the Q(s) values (Section 4), for the C(t) values performing assignments and generating error values and saving them to the spreadsheet file (Section 5).

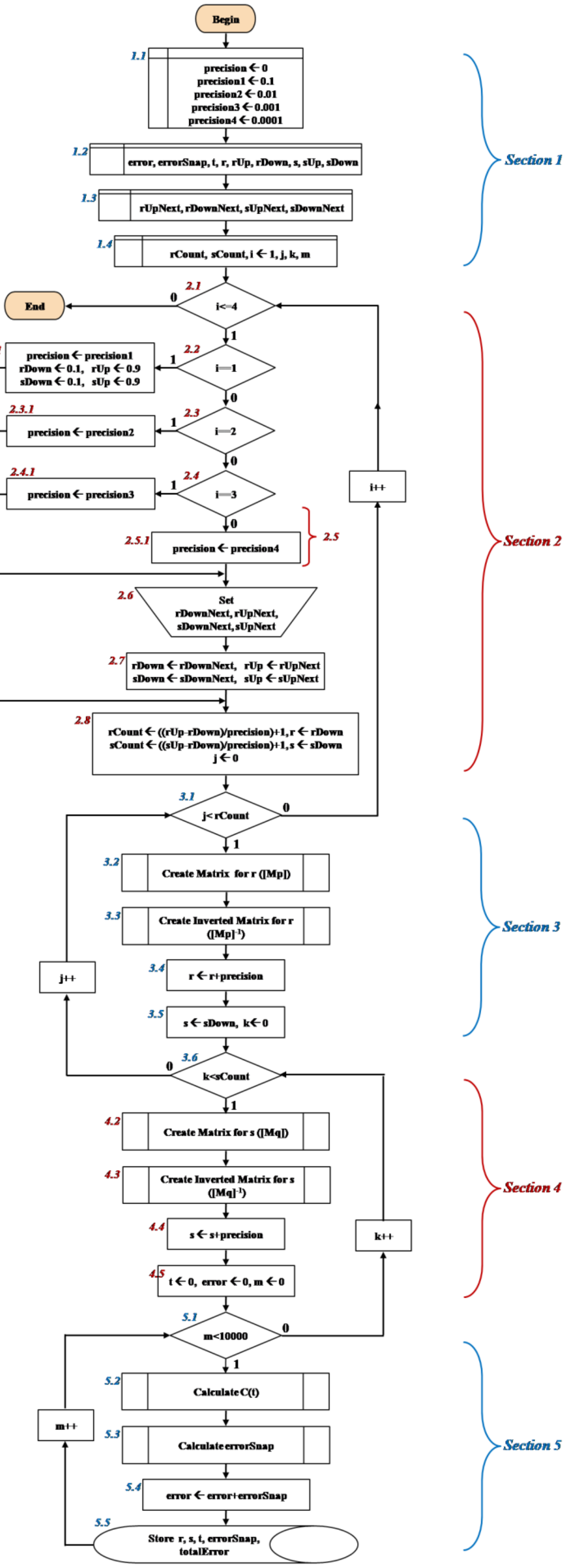

Figure 2 General flow diagram of the software

Assignment operations required for four different precision (i) values are performed in Section 2. For example, if the value of i is equal to 1 (Section 2.2.1), the precision value will be 0.1. A range for r and s will be defined, and C(t) will be calculated on this range. The closed interval [0.2, 0.9] is defined for r and s. A value of 0.1 (precision1) is determined as r/sDown and a value of 0.9 as r/sUp. For r and s values, the number of iterations to be made in the outer two-dimensional loop must be specified (Section 2.8).

After the first cycle of the loop, the results were filtered and examined. In case the i value is 2, 3 and 4 (Section 2.2 – Section2.4), the necessary value assignment (r/sDown,r/sUp) operations were performed manually (Section 2.6 and Section 2.7). If i is equal to 3 (Section 2.4), precision value 0.001 (Section 2.4.1) will be selected.

For example, if the interval values for r and s were chosen as [0.45, 0.55], [0.32, 0.51], respectively, the calculated values for precision=0.01 (i=2) would be as shown below.

$$rCount \leftarrow \left(\frac{rUp - rDown}{precision}\right) + 1$$

$$rCount \leftarrow \left(\frac{0.55 - 0.45}{0.01}\right) + 1$$

$$rCount \leftarrow 11$$

$$sCount \leftarrow \left(\frac{sUp - sDown}{precision}\right) + 1$$

$$sCount \leftarrow \left(\frac{0.51 - 0.32}{0.01}\right) + 1$$

$$sCount \leftarrow 20$$

Thus, calculations will be made for rCount*sCount (11*20)with 220 different C(t) (Formula 7).P(r) will be in the row of the two-dimensional loop, where C(t) will be calculated for different r and s values (Section 3). P(r) will be in the row of the two-dimensional loop where C(t) will be calculated for different r and s values. For a certain value of r, the matrix [Mp] (Section 2.2) and the inverse matrix [Mp]-1(Section 2.3) must be calculated. Then, the precision value, which has been assigned a value based on the value of i, is assigned to r. At each step of the loop, the s value is determined as the lower limit (sDown) of the range.

The operations for r in Section 3 are repeated for s in Section 4. At the end of Section 4, the error and t values are also reset.

After generating C(t) for a given r and s values, rinstant and sinstant values are calculated based on the t value. These three parameters are written into their places in Formula 7. This process is repeated twice for the x and y values of C(t). The x values of the control vertices are used in the calculation for x, and the y values of the control vertices are used for y. Thus, x and y values are produced for any value of t.

For a given r-s combination, a C(t) is generated. In this study, C(t) is calculated by choosing the sensitivity value 10-5 for the sequential increase of the t value. For any value of t, when x parameters of control vertices are used in Formula 7, x value for C(t) is produced. For the same t value, when the y values of the control vertices are used in Formula 7, the y value for c(t) is produced. Thus, (x,y) values are produced for a certain t value. When this x value produced by the program matches the x value in the dataset, the relative difference between the y value in the dataset and the y value produced by the program creates the error value. The resulting error value is calculated by averaging these error values (16 vertices in the dataset).

In the reference study, this error value was calculated as 4.386 for the default r-s values used for PB [16]. In this study, the aim is to obtain a lower error rate by changing C(t) and by changing the r-s values of PB.

4. Experimental Study and Results

A program was written using C# programming language to generate different C(t) equations using different r*s combinations. The results produced by the program were compared with the reference dataset (Column 2 in Table 2). The program exports these comparison results to the spreadsheet files. As a result of filtering the data in the exported file separately for each different precision value, the interval value for r and s in the next step (precision) is determined. An example of the results produced by the program for different r and s parameters is shown in Figure 3. The blue colored curve ("Real") is the reference dataset used in the study.

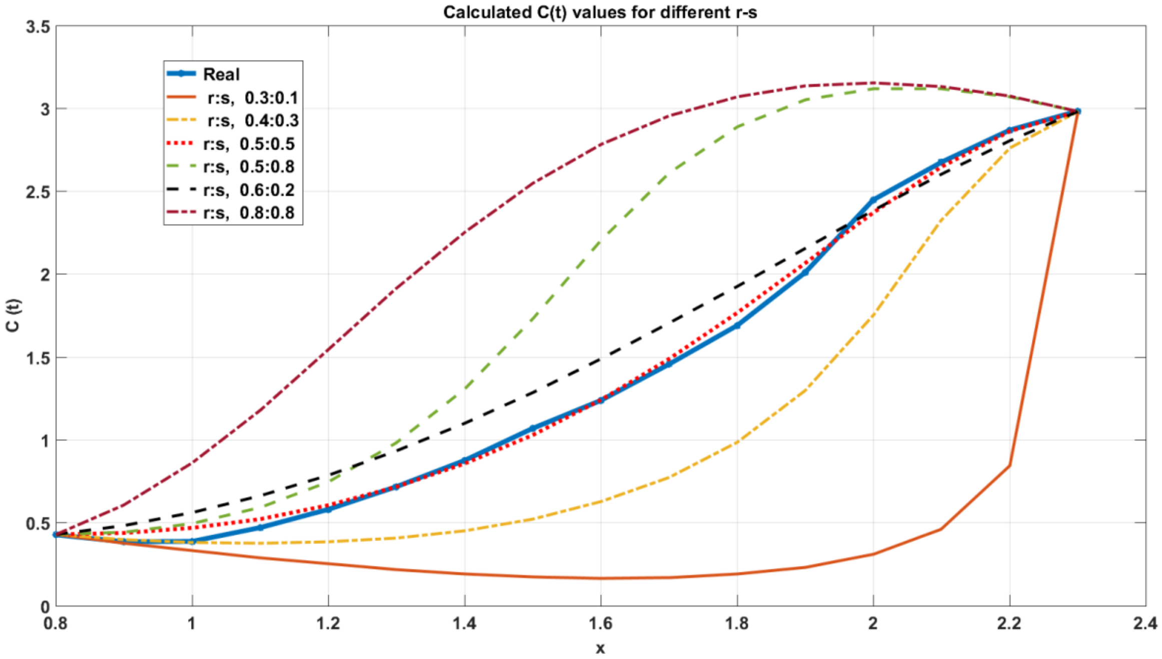

Figure 3 Calculated c(t) values for different r-s➔P_2-P_3; precision🡐0.1

The r and s interval values for precision=0.1 were determined as [0.2, 0.9] and [0.1, 0.8], respectively. Therefore, the r value is calculated as 8 and the s value as 8 (Figure 2, Section 2.2.1).

$$rCount \leftarrow \left(\frac{rUp - rDown}{precision}\right) + 1$$

$$rCount \leftarrow \left(\frac{0.9 - 0.2}{0.1}\right) + 1$$

$$rCount \leftarrow 8$$

$$sCount \leftarrow \left(\frac{sUp - sDown}{precision}\right) + 1$$

$$sCount \leftarrow \left(\frac{0.8 - 0.1}{0.1}\right) + 1$$

$$sCount \leftarrow 8$$

In this way, 8*8 (r*s) C(t) formulas are created (Formula 7). 64 error values are calculated by comparing the values calculated using these formulas with the measured thickness distribution of a 3mm thick high impact polystyrene product after thermoforming using a mold values ("Real" in Figure 3). The error values obtained for some r and s values from these 64 different error values are shown in Table 1 (Branch A).

Table 1 Examples of error values calculated based on r and s values for different precision

Branch	R	S	Average Error
A *precision 0.1* *i=1*	0.3	0.8	44.5248
	0.4	0.7	16.3331
	0.5	*0.5*	*4.3861*
	0.5	0.8	36.0096
	0.7	0.5	35.2096
	0.9	0.5	46.6825
B *precision 0.01* *i=2*	*0.49*	*0.53*	*4.2374*
	0.49	0.51	4.3546
	0.49	0.54	4.239
	0.49	0.56	4.3507
	0.48	0.59	4.6175
	0.41	0.61	17.7456
C *precision 0.001* *i=3*	0.487	0.539	4.2201
	0.487	*0.541*	*4.2148*
	0.487	0.543	4.22
	0.491	0.531	4.2416
	0.502	0.501	4.557
	0.502	0.506	4.5979
D *precision 0.0001* *i=4*	0.487	0.5405	4.2139
	0.487	*0.5407*	*4.2133*
	0.4876	0.5387	4.2168
	0.4881	0.546	4.248
	0.4937	0.5403	4.3049
	0.4862	0.5259	4.4408

The data related to these error values obtained were transferred to MatLAB® and graphed (Figure 4). In this graph, the vertices where the error value is small (r,s) are seen as darker blue.

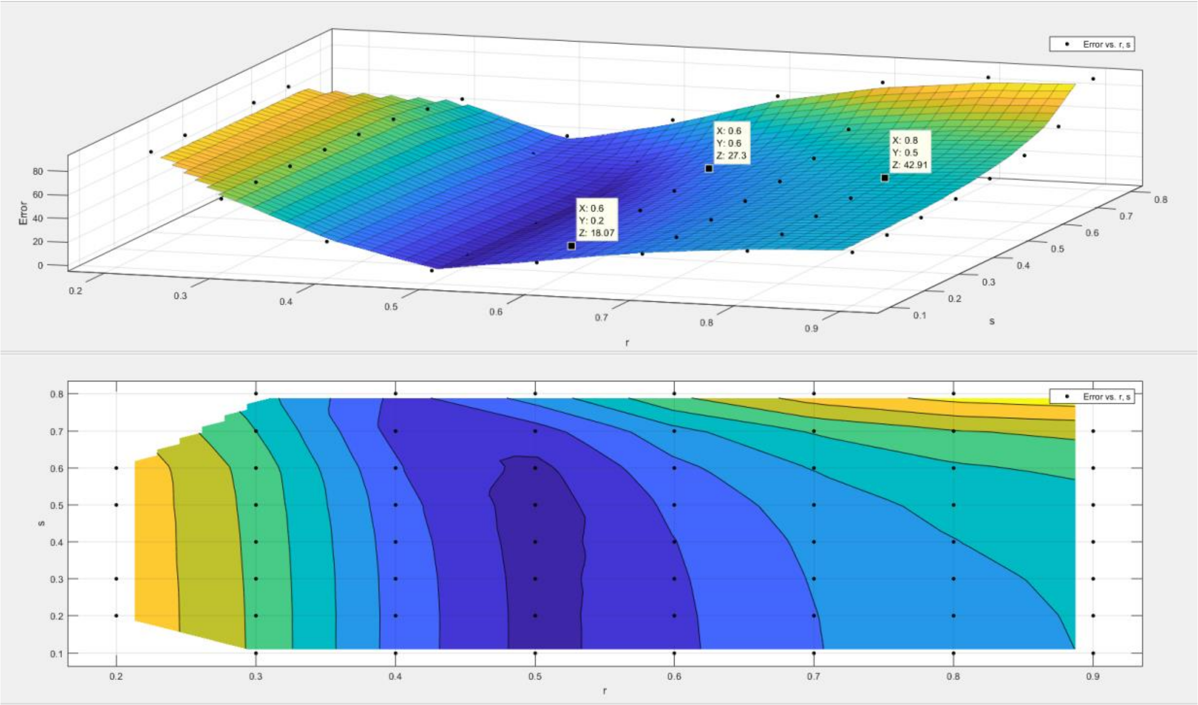

Figure 4 Visualization of the results achieved, precision←0.1

The r and s interval values for precision=0.01 were determined as [0.40, 0.60] and [0.10, 0.70], respectively. Therefore, the r value is calculated as 21 and the s value as 61 (Figure 2, Section 2.3.1 - 2.8).

$$rCount \leftarrow \left(\frac{rUp - rDown}{precision}\right) + 1$$

$$rCount \leftarrow \left(\frac{0.60 - 0.40}{0.01}\right) + 1$$

$$rCount \leftarrow 21$$

$$sCount \leftarrow \left(\frac{sUp - sDown}{precision}\right) + 1$$

$$sCount \leftarrow \left(\frac{0.70 - 0.10}{0.01}\right) + 1$$

$$sCount \leftarrow 61$$

In this way, 21*61 (r*s) C(t) formulas are created, and1,281 error values are calculated by comparing the values calculated using these formulas with the dataset. Some error values for r and s from these 1,281 different error values are shown in Table 1 (Branch B). In addition, the results obtained are shown in Figure 5 as a graph.

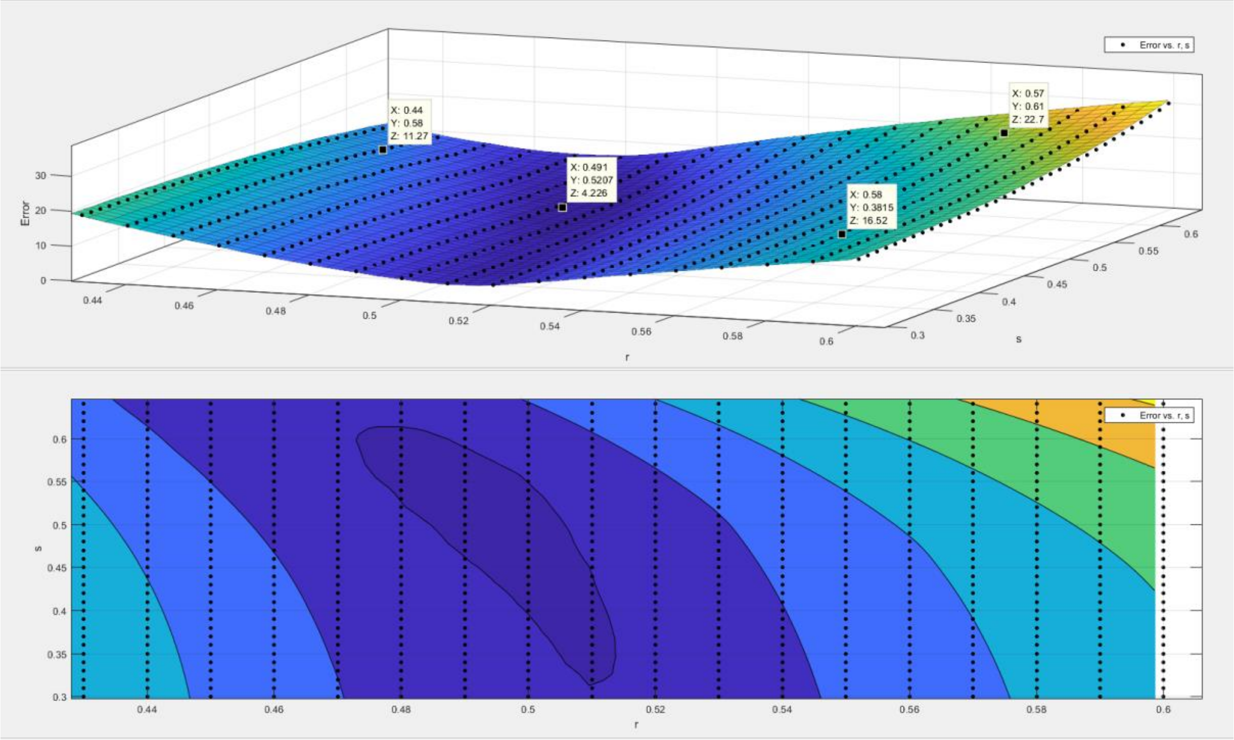

Figure 5 Visualization of the results achieved, precision←0.01

The r and s interval values for precision=0.001 were determined as [0.480, 0.510] and [0.480, 0.570], respectively. Therefore, the r value is calculated as 31 and the s value as 91 (Figure 2, Section 2.4.1 – 2.8).

$$rCount \leftarrow \left(\frac{rUp - rDown}{precision}\right) + 1$$

$$rCount \leftarrow \left(\frac{0.510 - 0.480}{0.001}\right) + 1$$

$$rCount \leftarrow 31$$

$$sCount \leftarrow \left(\frac{sUp - sDown}{precision}\right) + 1$$

$$sCount \leftarrow \left(\frac{0.570 - 0.480}{0.001}\right) + 1$$

$$sCount \leftarrow 91$$

In this way, 31*91 (r*s) C(t) formulas are created, and 2,821 error values are calculated by comparing the values calculated using these formulas with the dataset. Some error values for r and s from these 2,821 different error values are shown in Table 1 (Branch C). In addition, the results obtained are shown in Figure 6 as a graph.

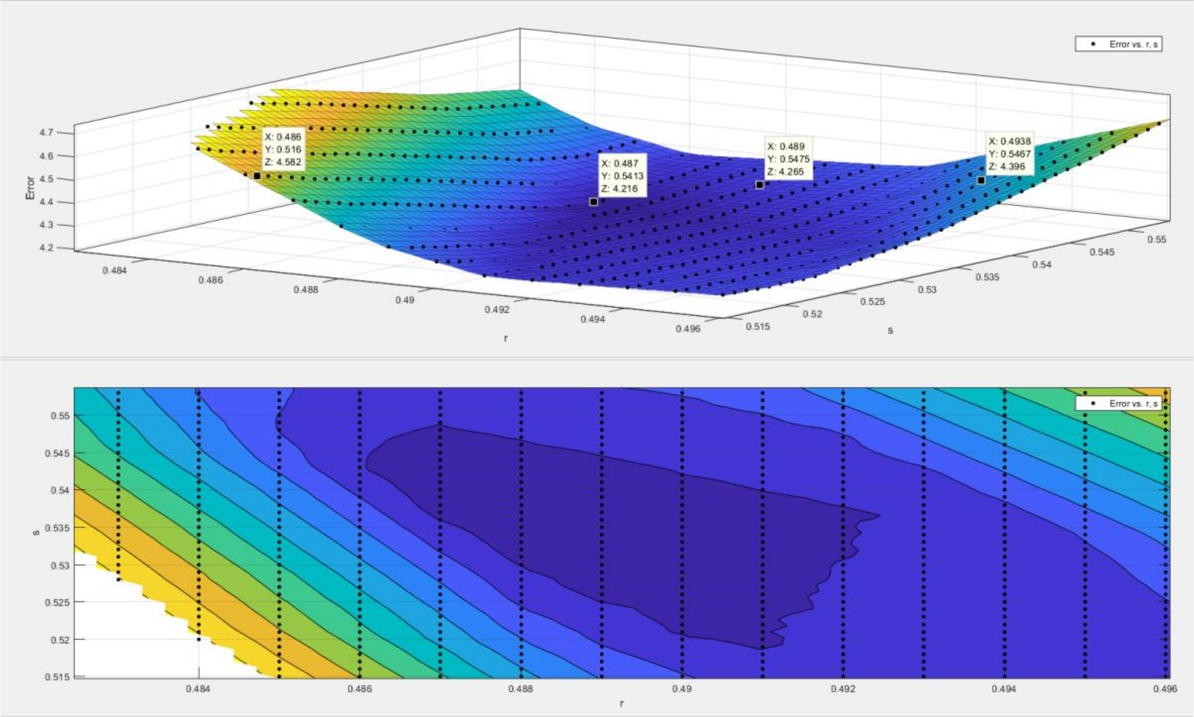

Figure 6 Visualization of the results achieved, precision🡐0.001

The r and s interval values for precision=0.0001 were determined as [0.4860, 0.4940] and [0.5175, 0.5475], respectively. Therefore, the r value is calculated as 81 and the s value as 301 (Figure 2, section 2.5.1 – 2.8).

$$rCount \leftarrow \left(\frac{rUp - rDown}{precision}\right) + 1$$

$$rCount \leftarrow \left(\frac{0.4940 - 0.4860}{0.0001}\right) + 1$$

$$rCount \leftarrow 81$$

$$sCount \leftarrow \left(\frac{sUp - sDown}{precision}\right) + 1$$

$$sCount \leftarrow \left(\frac{0.5475 - 0.5175}{0.0001}\right) + 1$$

$$sCount \leftarrow 301$$

In this way, 81*301 (r*s) C(t) formulas are created, and 24,381 error values are calculated by comparing the values calculated using these formulas with the dataset. Some error values for r and s from these 24,381 different error values are shown in Table 1 (Branch D). In addition, the results obtained are shown in Figure 7 as a graph.

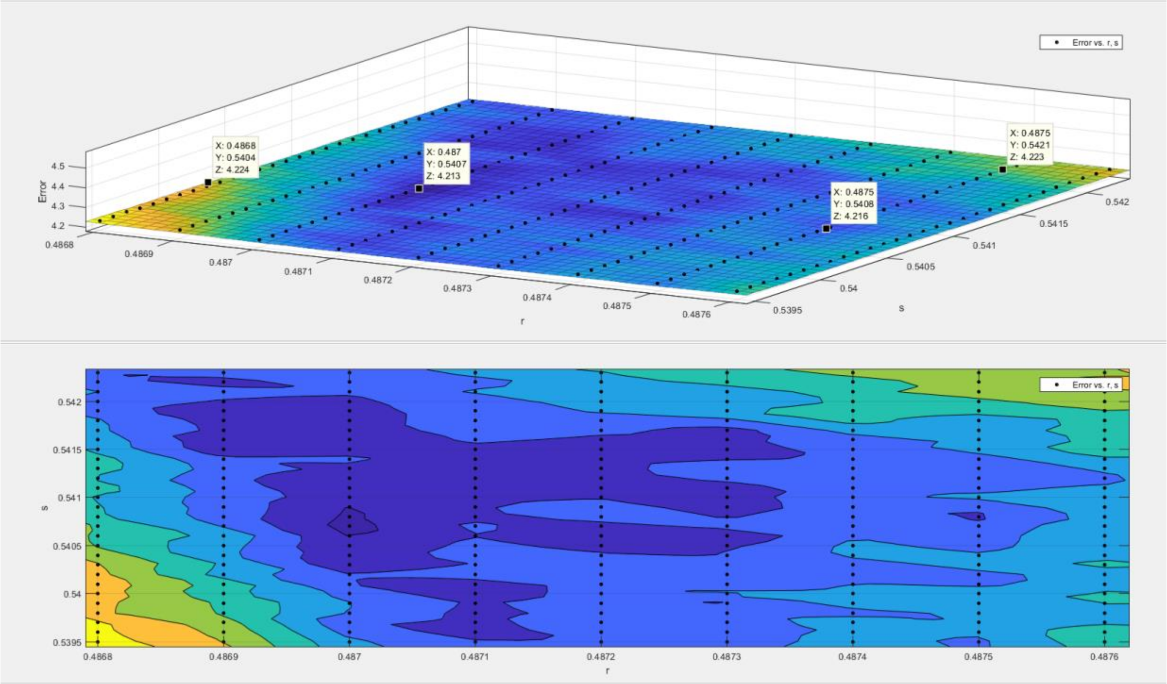

Figure 7 Visualization of the results achieved, precision←0.0001

The error value decreased as the interval value for r and s values was reduced in a controlled manner, taking into account the precision value. The error value for r=0.487, s=0.5407 was calculated as 4.2133 percent (Table 1, Branch D). This value was calculated as 4.3861 percent for r=0.5, s=0.5 (Table 1, Branch A) for PB's default state. This value also confirms the value obtained in the reference study. By optimizing the PB, the error value was reduced by 0.1728 percent.

The total absolute percentage relative error value for generalized C(t) was calculated as 67.4132 ($\sum_{n=8}^{23}|relativepercentageerror|$), and the average absolute percentage relative error value was calculated as 4.2133 (($\sum_{n=8}^{23}|relativepercentageerror|$)/16) (Table 2). This value means that the program produced a more accurate result in this study than in the previous study [16].

Table 2 PB curve values (default PB and GPB) for dataset

		Default PB(r=0.5; s=0.5)		***GPB (r=0.4870; s=0.5470)***	
X axis values	*Measured thick ness value (mm)*	*Y axis value produced by the program, C(t)*	*Relative error (%)*	*Y axis value produced by the program, C(t)*	*Relative error (%)*
0.8	0.4283	0.4283	0	0.4283	0
0.9	0.3866	0.4381	13.3213	0.4328	11.9503
1	0.3883	0.4694	20.8859	0.4582	18.0015
1.1	0.4716	0.5244	11.1959	0.5069	7.4852
1.2	0.5816	0.6055	4.1094	0.5817	0.0172
1.3	0.7183	0.7153	0.4177	0.6856	4.5524
1.4	0.8766	0.8563	2.3158	0.8221	6.2172
1.5	1.07	1.0312	3.6262	0.9948	7.028
1.6	1.2383	1.2417	0.2746	1.2068	2.5438
1.7	1.4566	1.4881	2.1626	1.4597	0.2128

Table 2 PB curve values (default PB and GPB) for dataset (cont.)

		Default PB(r=0.5; s=0.5)		*GPB (r=0.4870; s=0.5470)*	
X axis values	*Measured thick ness value (mm)*	*Y axis value produced by the program, C(t)*	*Relative error (%)*	*Y axis value produced by the program, C(t)*	*Relative error (%)*
1.8	1.69	1.7667	4.5385	1.7509	3.6036
1.9	2.01	2.0676	2.8657	2.0691	2.9403
2	2.4483	2.3711	3.1532	2.3896	2.3976
2.1	2.675	2.6471	1.043	2.6738	0.0449
2.2	2.8683	2.8606	0.2685	2.8803	0.4184
2.3	2.9833	2.9833	0	2.9833	0

The mean percent error with the GPB was calculated as 4.2133. This value was 4.3861 for the default PB (Table 1 Branch A and [16]). By optimizing the PB, an improvement of 0.1728 percent was achieved in the error value.

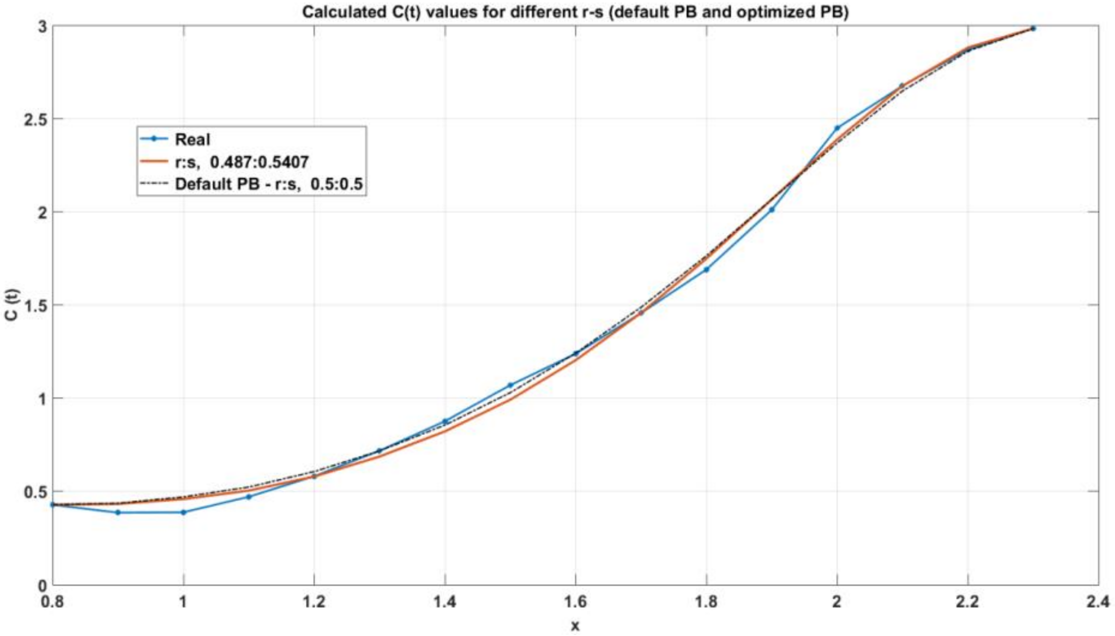

Figure 8 Default PB and GPB versus dataset (real measurement results)

5. Conclusion

The comparison of the calculated C(t) for the default r:s values and the calculated C(t) for the generalized r:s values with the dataset is shown in Figure 8. When the figure is examined, it is seen that C(t) calculated with the GPB outside the range of [1.2, 1.6] approaches the dataset with less error than the GPB.

The dataset used in this research was created as a result of measuring the thickness distribution of a 3 mm thick high-impact polystyrene product after thermoforming using a mold. For this dataset, the most suitable r and s values that can be used with a precision of 10^{-4} were calculated as r=0.487 and s=0.5407. These r and s values may vary for another dataset. With the algorithm and software developed in this study, the most appropriate r and s values can be determined for a new dataset. In every study where PB can be applied, the optimization steps revealed in this study can be applied.

OVR is a special case of GPB for the values for which it will take the r and s parameters. For this reason, it is not possible to say unconditionally that GBP produces more accurate results in all cases or for all types of datasets. Hadavinia et al. [17], working on sample functions, stated that GPB can produce more accurate results than OVR. In this study, the results obtained by working on a real dataset were evaluated. The results obtained confirm the results and claims of Hadavinia et al. In addition, the results obtained through this study have been visualized in an easy-to-follow manner, and a software development process algorithm has been presented for use in further studies. In addition, it will be less costly for the software development process to fit curves in a flexible structure using four vertices instead of creating a high-order interpolation polynomial using all vertices.

References

[1] Shrivastava, A., & Dalla, V. K. (2022). Multi-segment trajectory tracking of the redundant space robot for smooth motion planning based on interpolation of linear polynomials with parabolic blend. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 09544062221088723, doi.org/10.1177/09544062221088723

[2] Hachemi, M., Hamza-Cherif, S. M., & Houmat, A. (2017). Free vibration analysis of variable stiffness composite laminate plate with circular cutout. Australian Journal of Mechanical Engineering, doi.org/10.1080/14484846.2017.1385694

[3] Ben Abdallah, A., Kallel, A., Hassine, T., Gamaoun, F., & Tcharkhtchi, A. (2022). Modeling of viscoelastic behavior of a shape memory polymer blend. Journal of Applied Polymer Science, 139(13), 51859, doi.org/10.1177/09544062221088723

[4] Overhauser, A. W. (2005). Analytic definition of curves and surfaces by parabolic blending. arXiv preprint cs/0503054.

[5] Liu, Y., & Rizzo, F. J. (1991). Application of Overhauser C 1 Continuous Boundary Elements to “Hypersingular” BIE for 3-D Acoustic Wave Problems. In Boundary elements XIII (pp. 957-966). Springer, Dordrecht, doi.org/10.1007/978-94-011-3696-9_75

[6] Walters, H. G., & Gipson, G. S. (1994). Evaluation of overhauser splines as boundary elements in linear elastostatics. Engineering analysis with boundary elements, 14(2), 171-177, doi.org/10.1016/0955-7997(94)90093-0

[7] Durodola, J. F., & Fenner, R. T. (1996). Overhauser triangular elements for three-dimensional potential problems using boundary element methods. International journal for numerical methods in engineering, 39(24), 4183-4198, doi.org/10.1002/(SICI)1097-0207(19961230)39:24%3C4183::AID-NME38%3E3.0.CO;2-9

[8] Brewer, J. A., & Anderson, D. C. (1977). Visual interaction with overhauser curves and surfaces. ACM SIGGRAPH Computer Graphics, 11(2), 132-137, doi.org/10.1145/965141.563883

[9] Schneider, W. (1986). A simple technique for adding tension to parabolic blending interpolation. Computers & Mathematics with Applications, 12(11), 1155-1160, doi.org/10.1016/0898-1221(86)90019-2

[10] Qian, X., Yuan, H., Zhou, M., & Zhang, B. (2014). A general 3D contact smoothing method based on radial point interpolation. Journal of Computational and Applied Mathematics, 257, 1-13.

[11] El-Abbasi, N., Meguid, S. A., & Czekanski, A. (2001). On the modelling of smooth contact surfaces using cubic splines. International Journal for Numerical Methods in Engineering, 50(4), 953-967, doi.org/10.1002/1097-0207(20010210)50:4%3C953::AID-NME64%3E3.0.CO;2-P

[12] Chung, K. H., Kim, J. W., Ryu, K. W., Lee, K. T., & Lee, D. J. (2006). Sound generation and radiation from rotor tip-vortex pairing phenomenon. AIAA journal, 44(6), 1181-1187, doi.org/10.2514/1.22548

[13] De Almeida Barros, P. L., & de Mesquita Neto, E. (2000). Singular-ended spline interpolation for two-dimensional boundary element analysis. International Journal for Numerical Methods in Engineering, 47(5), 951-967.

[14] Kunz, T., & Stilman, M. (2012). Time-optimal trajectory generation for path following with bounded acceleration and velocity. Robotics: Science and Systems VIII, 1-8.

[15] Burgoyne, C. J., & Crisfield, M. A. (1990). Numerical integration strategy for plates and shells. International journal for numerical methods in engineering, 29(1), 105-121, doi.org/10.1002/nme.1620290108

[16] Ekşi, O., & Üstünel, H. (2020). Application of parabolic blending for the estimation of thickness distribution in thermoformed products. Journal of Elastomers & Plastics, 0095244320959801, doi.org/10.1177%2F0095244320959801

[17] Hadavinia, H., Travis, R. P., & Fenner, R. T. (2000). C1-continuous generalised parabolic blending elements in the Boundary Element Method. Mathematical and Computer Modelling, 31(8-9), 17-34, doi.org/10.1016/S0895-7177(00)00057-1

[18] Rogers DF and Adams JA. Mathematical elements for computer graphics. 2nd ed. New York: McGraw-Hill, 1989.

Notation List

P, Q	: Parabolas
C	: Cubic function (curve)
P_n	: n^{th} point (It can take integer values between 1 and 4)
r,s,t	: Parameters of P, Q and C respectively *(it can take float values between 0 and 1, including limit values)*
[B]	: Coefficient matrix for P
k_n	: Coefficients of the line equation used in the calculation of t-dependent change in the r and s values *(1 and 2 for r, 3 and 4 for s)*
$r_{instant}$ (t)	: The calculated r value for any value of the t parameter
$s_{instant}$ (t)	: The calculated s value for any value of the t parameter
rCount	: The number of steps in the loop for the precision value set for r *(in the software)*
sCount	: The number of steps in the loop for the precision value set for s *(in the software)*
rUp, rDown)	: Upper and lower limit value for r value used when calculating rCount *(in the software)*
sUp, sDown)	: Upper and lower limit value for s value used when calculating sCount *(in the software)*