Bezier Curves

Antoni A. Urban
Software Makers
Sandton, South Africa

NO REFERENCES

Abstract. Bezier curves were introduced not so many years ago, in the sixties. Immediately, they were used across the computer scene. Bezier curves of the 3rd degree were included into the Post-Script language (commands: *curveto*, *rcurveto*). They are now present in each printed document. We ask two questions: (1) why are Bezier curves so popular? (2) do we know everything about Bezier curves?
As well as the theory of Bezier curves, the article explores a new concept of *software proofs*. An author of a paper creates software which illustrates the theory considered in the paper. In certain cases the software can prove theorems from the paper. The software must be attached to the paper in open format, i.e. complete source code of the software. A reader opens the software utilising the Microsoft Visual Basic - probably the most popular compiler at the moment.

Let us consider the complex space **C**. An element of **C** is an ordered pair (x,y), where the both x and y are real numbers. Two elements (x_1,y_1) and (x_2,y_2) of **C** are equal iff $x_1=x_2$ and $y_1=y_2$.

DEFINITION 1. A finite sequence of complex numbers $u_0,\ldots,u_n$ is denoted by:

$$<u_0,\ldots,u_n> \;=\; <u_i>_{i=0}^{n}$$

and called *a frame of the n-th degree*. We denote by F_n the set of all frames of the n-th degree, and by F the set of all frames, i.e. $F = \bigcup_{n\in\mathbf{N}} F_n$

DEFINITION 2. *A curve in the complex space* **C** is a mapping γ from the interval $[0,1]\subset\mathbf{R}$ to the complex space **C**.

DEFINITION 3. Let us consider a curve γ in the complex space **C**. Let us assume that there exists a frame $<u_i>_{i=0}^{n}$ such that:

$$\forall\, t\in[0,1]:\ \gamma(t) = \sum_{i=0}^{n} t^i u_i$$

The curve γ is called *a Taylor curve of the n-th degree defined by the frame* $<u_i>_{i=0}^{n}$

DEFINITION 4. Let us consider a curve γ in the complex space **C**. Let us assume that there exists a frame $<p_i>_{i=0}^{n}$ such that:

$$\forall\, t\in[0,1]:\ \gamma(t) = \beta_n(p_0,\ldots,p_n,t)$$

where β_n is defined by the following induction:

$$(n=1)\ \ \beta_1(p_0,p_1,t) = (1-t)\,p_0 + t\,p_1$$

$$(n=k)\ \ \beta_{k+1}(p_0,\ldots,p_{k+1},t) = (1-t)\,\beta_k(p_0,\ldots,p_k,t) + t\,\beta_k(p_1,\ldots,p_{k+1},t)$$

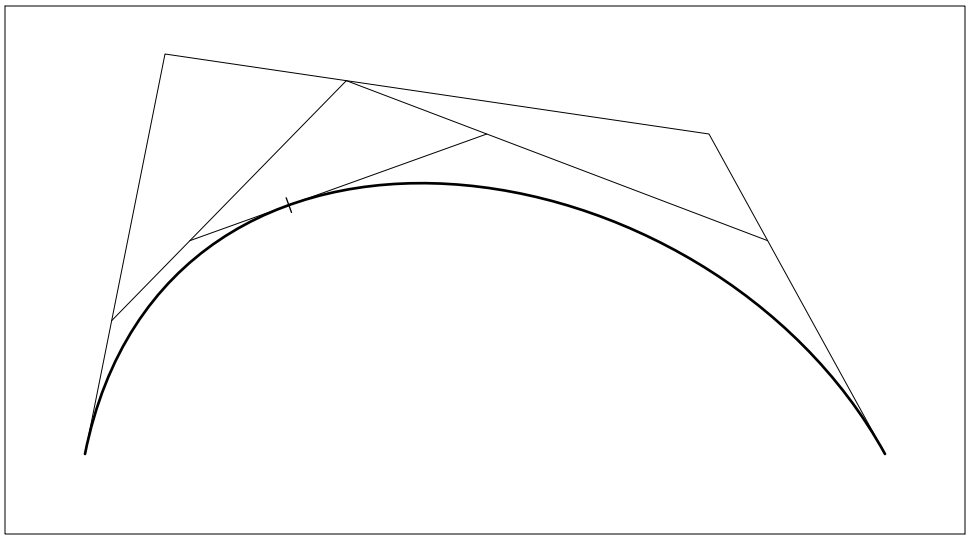
SAMPLE

$(n=1)\quad \beta_1(p_0,p_1,t) = p_0 + t\cdot(p_1-p_0)$

$(n=2)\quad \beta_2(p_0,p_1,p_2,t) = p_0 + t\cdot 2(p_1-p_0) + t^2\cdot(p_2-2p_1+p_0)$

$(n=3)\quad \beta_3(p_0,p_1,p_2,p_3,t) = p_0 + t\cdot 3(p_1-p_0) + t^2\cdot 3(p_2-2p_1+p_0) + t^3\cdot(p_3-3p_2+3p_1-p_0)$

PICTURE 1.



PROPOSITION 1. Let us consider two frames $<u_i>_{i=0}^{n}$ and $<p_i>_{i=0}^{n}$. The first frame $<u_i>_{i=0}^{n}$ defines the Taylor curve γ_1. The second frame $<p_i>_{i=0}^{n}$ defines the Bezier curve γ_2. The curves γ_1 and γ_2 are equal iff

$$\forall_{i=0}^{n}\ u_i = \binom{n}{i}\sum_{j=0}^{i}(-1)^{i-j}\binom{i}{j}p_j \qquad (1)$$

iff

$$\forall_{i=0}^{n}\ \binom{n}{i}p_i = \sum_{j=0}^{i}\binom{n-j}{n-i}u_j \qquad (2)$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$u_0 = p_0$

$u_1 = 3(p_1-p_0)$

$u_2 = 3(p_2-2p_1+p_0)$

$u_3 = p_3-3p_2+3p_1-p_0$

SAMPLE $(n=3)$

$p_0 = u_0$

$3p_1 = 3u_0+u_1$

$3p_2 = 3u_0+2u_1+u_2$

$p_3 = u_0+u_1+u_2+u_3$

The curve γ is a function from $[0,1]$ to **C**. In case of a Taylor curve it can be extended to the domain **R**:

$$\forall t \in \mathbf{R}:\ \gamma(t) = \sum_{i=0}^{n} t^i u_i$$

We now compose the extended curve γ with the function τ from $[0,1]$ to **R**, i.e. $\gamma \circ \tau : [0,1] \to \mathbf{C}$. It means that $\gamma \circ \tau$ is a curve. We are particularly interested in the following functions τ:

$$t = \tau_N(s) = s + 1$$

$$t = \tau_P(s) = s - 1$$

$$t = \tau_H(s) = s / 2$$

$$t = \tau_D(s) = 2s$$

$$t = \tau_R(s) = 1 - s$$

DEFINITION 5. We now define the unary operation $\mathbf{Next_T}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<u_i>_{i=0}^{n}$ and $<v_i>_{i=0}^{n}$. The frames define Taylor curves γ and δ in accordance with the Definition 3, i.e.

$$\forall t \in \mathbf{R}:\ \gamma(t) = \sum_{i=0}^{n} t^i u_i \text{ and } \delta(t) = \sum_{i=0}^{n} t^i v_i$$

We say that $\mathbf{Next_T}(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff $\delta = \gamma \circ \tau_N$.

PROPOSITION 2. $\mathbf{Next_T}(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} v_i = \sum_{j=i}^{n} \binom{j}{i} u_j \tag{3}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$$v_0 = u_0 + u_1 + u_2 + u_3$$
$$v_1 = u_1 + 2u_2 + 3u_3$$
$$v_2 = u_2 + 3u_3$$
$$v_3 = u_3$$

DEFINITION 6. We now define the unary operation $\mathbf{Prev_T}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<u_i>_{i=0}^{n}$ and $<v_i>_{i=0}^{n}$. The frames define Taylor curves γ and δ in accordance with the Definition 3, i.e.

$$\forall t \in \mathbf{R}:\ \gamma(t) = \sum_{i=0}^{n} t^i u_i \text{ and } \delta(t) = \sum_{i=0}^{n} t^i v_i$$

We say that $\mathbf{Prev_T}(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff $\delta = \gamma \circ \tau_P$.

PROPOSITION 3. $\mathbf{Prev_T}(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} v_i = \sum_{j=i}^{n} (-1)^{j-i} \binom{j}{i} u_j \tag{4}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$$v_0 = u_0 - u_1 + u_2 - u_3$$
$$v_1 = u_1 - 2u_2 + 3u_3$$
$$v_2 = u_2 - 3u_3$$
$$v_3 = u_3$$

DEFINITION 7. We now define the unary operation $\mathbf{Half_T}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<u_i>_{i=0}^{n}$ and $<v_i>_{i=0}^{n}$. The frames define Taylor curves γ and δ in accordance with the Definition 3, i.e.

$$\forall t \in \mathbf{R}: \ \gamma(t) = \sum_{i=0}^{n} t^i u_i \ \text{ and } \ \delta(t) = \sum_{i=0}^{n} t^i v_i$$

We say that $\mathbf{Half_T}(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff $\delta = \gamma \circ \tau_H$.

PROPOSITION 4. $\mathbf{Half_T}(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} v_i = 2^{-i} u_i \tag{5}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$$v_0 = 2^0 u_0$$
$$v_1 = 2^{-1} u_1$$
$$v_2 = 2^{-2} u_2$$
$$v_3 = 2^{-3} u_3$$

DEFINITION 8. We now define the unary operation $\mathbf{Double_T}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<u_i>_{i=0}^{n}$ and $<v_i>_{i=0}^{n}$. The frames define Taylor curves γ and δ in accordance with the Definition 3, i.e.

$$\forall t \in \mathbf{R}: \ \gamma(t) = \sum_{i=0}^{n} t^i u_i \ \text{ and } \ \delta(t) = \sum_{i=0}^{n} t^i v_i$$

We say that $\mathbf{Double_T}(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff $\delta = \gamma \circ \tau_D$.

PROPOSITION 5. $\mathbf{Double_T}(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} v_i = 2^{i} u_i \tag{6}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$$v_0 = 2^0 u_0$$
$$v_1 = 2^1 u_1$$
$$v_2 = 2^2 u_2$$
$$v_3 = 2^3 u_3$$

DEFINITION 9. We now define the unary operation **Reverse$_{\mathbf{T}}$** on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<u_i>_{i=0}^{n}$ and $<v_i>_{i=0}^{n}$. The frames define Taylor curves γ and δ in accordance with the Definition 3, i.e.

$$\forall t \in \mathbf{R}:\ \gamma(t) = \sum_{i=0}^{n} t^i u_i \ \text{ and } \ \delta(t) = \sum_{i=0}^{n} t^i v_i$$

We say that **Reverse$_{\mathbf{T}}$**$(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff $\delta = \gamma \circ \tau_{\mathbf{R}}$.

PROPOSITION 6. **Reverse$_{\mathbf{T}}$**$(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} v_i \ = \ \sum_{j=i}^{n} (-1)^i \binom{j}{i} u_j \tag{7}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$$v_0 = u_0 + u_1 + u_2 + u_3$$
$$v_1 = -u_1 - 2u_2 - 3u_3$$
$$v_2 = u_2 + 3u_3$$
$$v_3 = -u_3$$

DEFINITION 10. We now define the unary operation **Deriv$_{\mathbf{T}}$** on the set F as follows. Let $n \in \mathbf{N}$: $n > 1$ be given. Let us consider two frames $<u_i>_{i=0}^{n}$ and $<v_i>_{i=0}^{n-1}$. The frames define Taylor curves γ and δ in accordance with the Definition 3, i.e.

$$\forall t \in \mathbf{R}:\ \gamma(t) = \sum_{i=0}^{n} t^i u_i \ \text{ and } \ \delta(t) = \sum_{i=0}^{n-1} t^i v_i$$

We say that **Deriv$_{\mathbf{T}}$**$(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n-1}$ iff $\delta = \gamma'$.

PROPOSITION 7. **Deriv$_{\mathbf{T}}$**$(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n-1}$ iff

$$\bigvee_{i=0}^{n} v_i \ = \ (i+1)\, u_{i+1} \tag{8}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$$v_0 = u_1$$
$$v_1 = 2u_2$$
$$v_2 = 3u_3$$

DEFINITION 11. We now define the unary operation **Integr$_{\mathbf{T}}$** on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<u_i>_{i=0}^{n}$ and $<v_i>_{i=0}^{n+1}$. The frames define Taylor curves γ and δ in accordance with the Definition 3, i.e.

$$\forall t \in \mathbf{R}:\ \gamma(t) = \sum_{i=0}^{n} t^i u_i \ \text{ and } \ \delta(t) = \sum_{i=0}^{n+1} t^i v_i$$

We say that **Integr$_{\mathbf{T}}$**$(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n+1}$ iff $\delta' = \gamma$ and $v_0 = 0$

PROPOSITION 8. $\mathbf{Integr_T}(<u_i>_{i=0}^{n}) = <v_i>_{i=0}^{n+1}$ iff

$$v_0 = 0 \text{ and } \bigwedge_{i=1}^{n+1} v_i = i^{-1} u_{i-1} \tag{9}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$v_0 = 0$

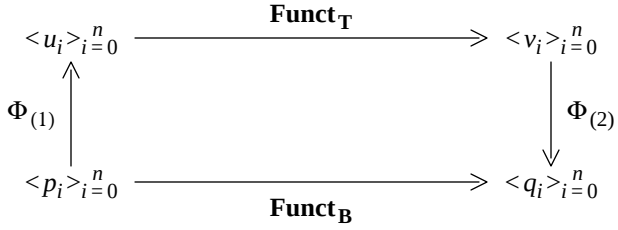
$v_1 = u_0$

$v_2 = 2^{-1} u_1$

$v_3 = 3^{-1} u_2$

$v_4 = 4^{-1} u_3$

PICTURE 2.



We have defined the following operators: $\mathbf{Next_T}, \mathbf{Prev_T}, \mathbf{Half_T}, \mathbf{Double_T}, \mathbf{Reverse_T}, \mathbf{Deriv_T}, \mathbf{Integr_T}$ on the set F, i.e. all frames in the complex space $\mathbf{C}$. Each frame defines a Taylor curve. Therefore, the above-mentioned operators are also operators on the set of all Taylor curves. We now define the corresponding operators: $\mathbf{Next_B}, \mathbf{Prev_B}, \mathbf{Half_B}, \mathbf{Double_B}, \mathbf{Reverse_B}, \mathbf{Deriv_B}, \mathbf{Integr_B}$ on the set of all Bezier curves. Picture 2 shows how the operator $\mathbf{Funct_B}$ is constructed:

$$\mathbf{Funct_B} = \Phi_{(2)} \circ \mathbf{Funct_T} \circ \Phi_{(1)}$$

where $\Phi_{(1)}$ is defined by formula (1) and $\Phi_{(2)}$ is defined by formula (2).

But firstly, we must define *Bezier's triangle*. Below are shown the first rows of the triangle.

1	row 0
1 2	row 1
1 4 4	row 2
1 6 12 8	row 3
1 8 24 32 16	row 4
1 10 40 80 80 32	row 5

In general, the n-th row consists of the coefficients $\{\binom{n}{m}\}$, with $m = 0, \ldots, n$ defined by the following induction:

(1) If $m < 0$ or $m > n$ then $\{\binom{n}{m}\} = 0$

(2) If $(n = 0)$ then $\{\binom{0}{0}\} = 1$

If $(n = k)$ then $\{\binom{k+1}{m}\} = 2 \cdot \{\binom{k}{m-1}\} + \{\binom{k}{m}\}$

It means we add twice the number situated above it to the left and the number situated above it to the right.

One can prove that:

$$\{ {n \atop m} \} = ({n \atop m}) \cdot 2^m \tag{10}$$

DEFINITION 12. We now define the unary operation $\mathbf{Next_B}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<p_i>_{i=0}^{n}$ and $<q_i>_{i=0}^{n}$. We say that:

$$\mathbf{Next_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n} \text{ iff } \mathbf{Next_T}(\Phi_{(1)}(<p_i>_{i=0}^{n})) = \Phi_{(1)}(<q_i>_{i=0}^{n})$$

PROPOSITION 9. $\mathbf{Next_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} q_i = \sum_{j=0}^{i} (-1)^j \{ {i \atop i-j} \} p_{n-j} \tag{11}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$q_0 = p_3$

$q_1 = 2p_3 - p_2$

$q_2 = 4p_3 - 4p_2 + p_1$

$q_3 = 8p_3 - 12p_2 + 6p_1 - p_0$

PROPOSITION 10. $\mathbf{Next_B} = \Phi_{(2)} \circ \mathbf{Next_T} \circ \Phi_{(1)}$

PROPOSITION 11. $\mathbf{Next_T} = \Phi_{(1)} \circ \mathbf{Next_B} \circ \Phi_{(2)}$

PROPOSITION 12. If $\mathbf{Next_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ then

$$q_0 = p_n \text{ and } q_1 - q_0 = p_n - p_{n-1}$$

DEFINITION 13. We now define the unary operation $\mathbf{Prev_B}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<p_i>_{i=0}^{n}$ and $<q_i>_{i=0}^{n}$. We say that:

$$\mathbf{Prev_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n} \text{ iff } \mathbf{Prev_T}(\Phi_{(1)}(<p_i>_{i=0}^{n})) = \Phi_{(1)}(<q_i>_{i=0}^{n})$$

PROPOSITION 13. $\mathbf{Prev_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} q_i = \sum_{j=i}^{n} (-1)^{j-i} \{ {n-i \atop n-j} \} p_{j-i} \tag{12}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$q_0 = 8p_0 - 12p_1 + 6p_2 - p_3$

$q_1 = 4p_0 - 4p_1 + p_2$

$q_2 = 2p_0 - p_1$

$q_3 = p_0$

PROPOSITION 14. $\mathbf{Prev_B} = \Phi_{(2)} \circ \mathbf{Prev_T} \circ \Phi_{(1)}$

PROPOSITION 15. $\mathbf{Prev_T} = \Phi_{(1)} \circ \mathbf{Prev_B} \circ \Phi_{(2)}$

PROPOSITION 16. If $\mathbf{Prev_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ then

$$q_n = p_0 \text{ and } q_n - q_{n-1} = p_1 - p_0$$

PROPOSITION 17. $\mathbf{Next_B} \circ \mathbf{Prev_B} = \mathbf{Id}_F$

PROPOSITION 18. $\mathbf{Prev_B} \circ \mathbf{Next_B} = \mathbf{Id}_F$

DEFINITION 14. We now define the unary operation $\mathbf{Half_B}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<p_i>_{i=0}^{n}$ and $<q_i>_{i=0}^{n}$. We say that:

$$\mathbf{Half_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n} \text{ iff } \mathbf{Half_T}(\Phi_{(1)}(<p_i>_{i=0}^{n})) = \Phi_{(1)}(<q_i>_{i=0}^{n})$$

PROPOSITION 19. $\mathbf{Half_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} q_i = 2^{-i} \sum_{j=0}^{i} \binom{i}{j} p_j \tag{13}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$$q_0 = p_0$$
$$q_1 = 2^{-1}(p_0+p_1)$$
$$q_2 = 2^{-2}(p_0+2p_1+p_2)$$
$$q_3 = 2^{-3}(p_0+3p_1+3p_2+p_3)$$

PROPOSITION 20. $\mathbf{Half_B} = \Phi_{(2)} \circ \mathbf{Half_T} \circ \Phi_{(1)}$

PROPOSITION 21. $\mathbf{Half_T} = \Phi_{(1)} \circ \mathbf{Half_B} \circ \Phi_{(2)}$

PROPOSITION 22. If $\mathbf{Half_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ then

$$q_0 = p_0 \text{ and } q_1 - q_0 = 2^{-1}(p_1 - p_0)$$

DEFINITION 15. We now define the unary operation $\mathbf{Double_B}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<p_i>_{i=0}^{n}$ and $<q_i>_{i=0}^{n}$. We say that:

$$\mathbf{Double_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n} \text{ iff } \mathbf{Double_T}(\Phi_{(1)}(<p_i>_{i=0}^{n})) = \Phi_{(1)}(<q_i>_{i=0}^{n})$$

PROPOSITION 23. $\mathbf{Double_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} q_i = \sum_{j=0}^{i} (-1)^{i-j} \left\{ {i \atop j} \right\} p_j \tag{14}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$$q_0 = p_0$$
$$q_1 = -p_0+2p_1$$
$$q_2 = p_0-2p_1+4p_2$$
$$q_3 = -p_0+6p_1-12p_2+8p_3$$

PROPOSITION 24. $\mathbf{Double_B} = \Phi_{(2)} \circ \mathbf{Double_T} \circ \Phi_{(1)}$

PROPOSITION 25. $\mathbf{Double_T} = \Phi_{(1)} \circ \mathbf{Double_B} \circ \Phi_{(2)}$

PROPOSITION 26. If $\mathbf{Double_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ then

$$q_0 = p_0 \text{ and } q_1 - q_0 = 2(p_1 - p_0)$$

PROPOSITION 27. $\mathbf{Half_B} \circ \mathbf{Double_B} = \mathbf{Id}_F$

PROPOSITION 28. $\mathbf{Double_B} \circ \mathbf{Half_B} = \mathbf{Id}_F$

PROPOSITION 29. $\mathbf{Next_B} = \mathbf{Double_B} \circ \mathbf{Next_B} \circ \mathbf{Next_B} \circ \mathbf{Half_B}$

PROPOSITION 30. $\mathbf{Prev_B} = \mathbf{Double_B} \circ \mathbf{Prev_B} \circ \mathbf{Prev_B} \circ \mathbf{Half_B}$

DEFINITION 16. We now define the unary operation $\mathbf{Reverse_B}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<p_i>_{i=0}^{n}$ and $<q_i>_{i=0}^{n}$. We say that:

$$\mathbf{Reverse_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n} \text{ iff } \mathbf{Reverse_T}(\Phi_{(1)}(<p_i>_{i=0}^{n})) = \Phi_{(1)}(<q_i>_{i=0}^{n})$$

PROPOSITION 31. $\mathbf{Reverse_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ iff

$$\bigvee_{i=0}^{n} q_i = p_{n-i} \tag{15}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$q_0 = p_3$

$q_1 = p_2$

$q_2 = p_1$

$q_3 = p_0$

PROPOSITION 32. $\mathbf{Reverse_B} = \Phi_{(2)} \circ \mathbf{Reverse_T} \circ \Phi_{(1)}$

PROPOSITION 33. $\mathbf{Reverse_T} = \Phi_{(1)} \circ \mathbf{Reverse_B} \circ \Phi_{(2)}$

PROPOSITION 34. If $\mathbf{Reverse_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ then

$$q_0 = p_n \text{ and } q_1 - q_0 = p_{n-1} - p_n$$

PROPOSITION 35. $\mathbf{Reverse_B} \circ \mathbf{Reverse_B} = \mathbf{Id}_F$

PROPOSITION 36. $\mathbf{Next_B} = \mathbf{Reverse_B} \circ \mathbf{Prev_B} \circ \mathbf{Reverse_B}$

PROPOSITION 37. $\mathbf{Prev_B} = \mathbf{Reverse_B} \circ \mathbf{Next_B} \circ \mathbf{Reverse_B}$

PROPOSITION 38. $\mathbf{Half_B} = \mathbf{Reverse_B} \circ \mathbf{Next_B} \circ \mathbf{Half_B} \circ \mathbf{Reverse_B}$

PROPOSITION 39. $\mathbf{Double_B} = \mathbf{Reverse_B} \circ \mathbf{Double_B} \circ \mathbf{Prev_B} \circ \mathbf{Reverse_B}$

DEFINITION 17. We now define the unary operation $\mathbf{Deriv_B}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<p_i>_{i=0}^{n}$ and $<q_i>_{i=0}^{n}$. We say that:

$$\mathbf{Deriv_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n} \text{ iff } \mathbf{Deriv_T}(\Phi_{(1)}(<p_i>_{i=0}^{n})) = \Phi_{(1)}(<q_i>_{i=0}^{n})$$

PROPOSITION 40. $\mathbf{Deriv_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n-1}$ iff

$$\bigwedge_{i=0}^{n-1} q_i = n(p_{i+1}-p_i) \tag{16}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$q_0 = 3(p_1-p_0)$
$q_1 = 3(p_2-p_1)$
$q_2 = 3(p_3-p_2)$

PROPOSITION 41. $\mathbf{Deriv_B} = \Phi_{(2)} \circ \mathbf{Deriv_T} \circ \Phi_{(1)}$

PROPOSITION 42. $\mathbf{Deriv_T} = \Phi_{(1)} \circ \mathbf{Deriv_B} \circ \Phi_{(2)}$

PROPOSITION 43. If $\mathbf{Deriv_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ then

$$q_0 = n(p_1 - p_0)$$

DEFINITION 18. We now define the unary operation $\mathbf{Integr_B}$ on the set F as follows. Let $n \in \mathbf{N}$ be given. Let us consider two frames $<p_i>_{i=0}^{n}$ and $<q_i>_{i=0}^{n}$. We say that:

$$\mathbf{Integr_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n} \text{ iff } \mathbf{Integr_T}(\Phi_{(1)}(<p_i>_{i=0}^{n})) = \Phi_{(1)}(<q_i>_{i=0}^{n})$$

PROPOSITION 44. $\mathbf{Integr_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n+1}$ iff

$$q_0 = 0 \text{ and } \bigwedge_{i=1}^{n+1} q_i = (n+1)^{-1} \sum_{j=0}^{i-1} p_j \tag{17}$$

SOFTWARE PROOF: `Bezier.vbp`

Download software from the same place as the main article. The software requires Microsoft Visual Basic ver. 6.0 (or higher) as a compiler. The software is easy to operate.

SAMPLE $(n=3)$

$q_0 = 0$
$q_1 = 1/4\, p_0$
$q_2 = 1/4\,(p_0+p_1)$
$q_3 = 1/4\,(p_0+p_1+p_2)$
$q_4 = 1/4\,(p_0+p_1+p_2+p_3)$

PROPOSITION 45. $\mathbf{Integr_B} = \Phi_{(2)} \circ \mathbf{Integr_T} \circ \Phi_{(1)}$

PROPOSITION 46. $\mathbf{Integr_T} = \Phi_{(1)} \circ \mathbf{Integr_B} \circ \Phi_{(2)}$

PROPOSITION 47. If $\mathbf{Integr_B}(<p_i>_{i=0}^{n}) = <q_i>_{i=0}^{n}$ then

$$(q_1 - q_0) = (n+1)^{-1} p_0$$

* * *