BABEŞ–BOLYAI UNIVERSITY OF CLUJ-NAPOCA
FACULTY OF MATHEMATICS AND INFORMATICS
SPECIALIZATION: COMPUTER SCIENCE

Diploma Thesis

Control point based curve and surface modeling in extended Chebyshev spaces

Abstract

By means of a basic software application, we would like to demonstrate the practical use of B-curves and B-surfaces defined over extended Chebyshev spaces (EC spaces) in control point based graphical modeling.

Using existing theoretical results we present numerical methods for computing the (zeroth and higher order) derivatives of the blending functions needed for generating EC B-curves and surfaces as well as algorithms for the general order elevation and subdivision of these. A basis transformation between the ordinary basis and the unique normalized B-basis (NB-basis) of an EC space is also presented, making control point based exact description possible for curves and surfaces given in a parametric form.

The application is written in C++ using the OpenGL and OpenMP libraries as well as the Qt Widgets framework. It provides the possibility of handling any EC space that can be identified by the solution space of a constant-coefficient homogeneous linear differential equation defined by the user. The multi-threaded implementation of the algorithms mentioned is numerically stable and efficient in practice for spaces with up to a sufficiently large dimension number.

I would like to thank my advisor, Ágoston Róth, who was of great help to me both in understanding and applying the theoretical results and in implementing the necessary numerical methods and OpenGL related functionalities as well.

This work is the result of my own activity. I have neither given nor received unauthorized assistance on this work.

2018 BEILAND ARNOLD

ADVISOR:
ASSOCIATE PROFESSOR
DR. ÁGOSTON ISTVÁN RÓTH

BABEŞ–BOLYAI UNIVERSITY OF CLUJ-NAPOCA
FACULTY OF MATHEMATICS AND INFORMATICS
SPECIALIZATION: COMPUTER SCIENCE

Diploma Thesis

Control point based curve and surface modeling in extended Chebyshev spaces

ADVISOR:

ASSOCIATE PROFESSOR
DR. ÁGOSTON ISTVÁN RÓTH

STUDENT:

BEILAND ARNOLD

2018

UNIVERSITATEA BABEŞ–BOLYAI, CLUJ-NAPOCA
FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ
SPECIALIZAREA INFORMATICĂ

Lucrare de licenţă

Modelarea curbelor și suprafețelor bazate pe puncte de control în spații Cebîșev-extinse

CONDUCĂTOR ŞTIINŢIFIC:

CONFERENȚIAR
DR. RÓTH ÁGOSTON ISTVÁN

ABSOLVENT:

BEILAND ARNOLD

2018

BABEŞ–BOLYAI TUDOMÁNYEGYETEM KOLOZSVÁR
MATEMATIKA ÉS INFORMATIKA KAR
INFORMATIKA SZAK

Szakdolgozat

Görbék és felületek kontrollpont-alapú modellezése kiterjesztett Csebisev-függvényterekben

TÉMAVEZETŐ:
DR. RÓTH ÁGOSTON ISTVÁN, DOCENS

SZERZŐ:
BEILAND ARNOLD

2018

Tartalomjegyzék

Bevezető

Jelen dolgozat célja egy példaalkalmazás keretein belül bemutatni a kiterjesztett Csebisev függvénytérek (KC-terek) felett értelmezett B-görbék és B-felületek felhasználási lehetőségeit a kontrollpont-alapú grafikus modellezésben.

Az 1. fejezet az elméleti háttér bemutatására irányul. A felhasznált eredmények és numerikus módszerek, illetve ezek helyességének és hatékonyságának vizsgálata [Róth, 2017]-ben megtalálható, jelen dolgozatban az innen átvett anyag összefoglalására törekszünk.

A tárgyalt numerikus módszerek egy része a kiterjesztett Csebisev-terek fölött értelmezett, úgynevezett B-görbék és B-felületek előállításához szükséges keverőfüggvények és ezek deriváltjainak kiszámítására irányul. Megvalósítható továbbá a görbék és felületek rendszámnövelése (mely a használt KC-tér dimenziószámának növekedését vonja maga után), ezek felosztása (az értelmezési tartomány tetszőleges belső pontja körül), illetve a paraméteres alakban megadott görbék és felületek kontrollpont-alapú egzakt leírása B-görbékkel és B-felületekkel (melyet egy, a KC-tér hagyományos és egyedi normalizált B-bázisa közötti bázistranszformáció tesz lehetővé).

A 2. fejezetben egy OpenGL-t felhasználó C++ alkalmazást mutatunk be, melynek célja az elméleti eredmények alkalmazási lehetőségeinek szemléltetése. A felhasználói felület elemeit a Qt Widgets keretrendszer biztosította, a saját grafikus elemeket viszont OpenGL segítségével rajzoltuk ki, így a görbék és felületek megjelenítésének paraméterei testreszabhatók (pl. hány görbepontnak megfelelő adat legyen tárolva a görbékhez társított vertex-pufferekben). Az ismertetett numerikus algoritmusok implementációjának párhuzamosítására az OpenMP könyvtárat használtuk.

Az alkalmazásban olyan KC-tereket tudunk definiálni és kezelni, amelyek egy konstansegyüttható s lineáris homogén differenciálegyenlet megoldásterének a megfelelő intervallumra való leszűkítéseként állnak elő. Egy ilyen differenciálegyenletet a felhasználó a gyökeinek (illetve ezek multiplicitásának) megadásával tud bevinni a rendszerbe.

A 3. fejezetben példákon keresztül mutatjuk be az alkalmazás használatát, illetve a hatékonyság szemléltetése céljából néhány futásidőre vonatkozó mérési eredményt is ismertetünk.

Túl nagy dimenziószámok, illetve túl kicsi értelmezési tartományok esetén sajnos nem elkerülhető a numerikus instabilitás, a gyakorlatban viszont megfelelően használható az implementáció, mivel a felületeket általában kisebb darabokból állítják elő, melyeket megfelelő folytonossági rend mellett összeillesztve kezelnek a CAD-rendszerek. A módszerek hatékonysága az általánosság ellenére sem jelent gondot, viszont kevésbé jó, mint azon eljárásoké, melyek egy bizonyos függvénytérre specializálódnak.

Köszönettel tartozom témavezetőmnek, Róth Ágostonnak, aki mind az elméleti eredmények megértésében, mind a felhasznált numerikus módszerek és a szükséges OpenGL eljárások implementálásában jelentős segítséget nyújtott.

1. fejezet

Elméleti háttér

Összefoglaló: *Bemutatjuk a felhasznált elméleti alapfogalmakat és eredményeket, minden esetben a megjelölt forrásműveket követve. Az értelmezések, tételek és bizonyítások leírása az adott művekben használt szerkezetet és jelölésrendszert követi.*

1.1. Kontrollpont-alapú modellezés

A grafikus modellezésben gyakran használt technika, hogy görbéket és felületdarabokat nem paraméteres alakban, hanem kontrollpontok és ezekhez társított keverőfüggvények lineáris kombinációjaként adunk meg. A kontrollpontok görbék esetében úgynevezett kontrollpoligont, felületeknél pedig kontrollhálót alkotnak. Ezáltal a felhasználónak lehetősége van módosítani a görbe vagy felület alakját (az egyes kontrollpontok helyének megváltoztatása által), ugyanakkor megmaradnak a keverőfüggvények jellegéből származó és a modellezésben hasznos tulajdonságok (pl. adott rendű folytonosság, a kontrollpontok lokális hatása, konvexitásmegőrzés, hullámzás- és hodográfcsökkentés, konvex burok tulajdonság stb.).

Egy görbét leggyakrabban a

$$\begin{cases} \mathbf{c} : [a,b] \to \mathbb{R}^\delta, \quad \delta \geq 2 \\ \mathbf{c}(u) = \displaystyle\sum_{i=0}^{n} \mathbf{p}_i f_i(u) \end{cases} \tag{1.1}$$

alakban adunk meg, ahol $n \geq 1$, a $\mathbf{p}_i \in \mathbb{R}^\delta$ vektorok a kontrollpontok, az f_i keverőfüggvények pedig az $[a,b]$ intervallumon értelmezett folytonos valós függvények, melyek normalizáltak, nemnegatívak, legalább n-edrendben folytonosan differenciálhatóak és lineárisan függetlenek.

A felületek alakja tenzorszorzat jellegű:

$$\begin{cases} \mathbf{s} : [a,b] \times [c,d] \to \mathbb{R}^\delta, \quad \delta \geq 3 \\ \mathbf{s}(u,v) = \displaystyle\sum_{i=0}^{n} \sum_{j=0}^{m} \mathbf{p}_{ij} f_i(u) g_j(v). \end{cases} \tag{1.2}$$

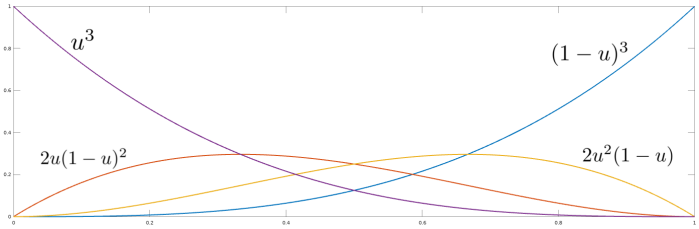
Itt a $\mathbf{p}_{ij} \in \mathbb{R}^\delta$ pontok egy kontrollhálót alkotnak. Az u- és v-irányú keverőfüggvény-rendszerek ez esetben lehetnek akár különböző dimenziószámúak, illetve különböző típusúak is:

$$\mathcal{F} = \left\{ f_i : [a,b] \to \mathbb{R} \right\}_{i=0}^{n}, \quad \mathcal{G} = \left\{ g_j : [c,d] \to \mathbb{R} \right\}_{j=0}^{m}.$$

A keverőfüggvények rendszere meghatározza a görbék és felületek tulajdonságait, emiatt érdemes a megfelelő függvények megválasztására külön figyelmet fordítani. A gyakorlatban a leggyakrabban használt görbék (felületek) az úgynevezett Bézier-görbék (felületek), melyek esetében a keverőfüggvények az n-edfokú Bernstein-polinomok:

$$\mathcal{B} = \left\{ B_{n,i} : [0,1] \to \mathbb{R}, \;\; B_{n,i}(u) = \binom{n}{i} u^i (1-u)^{n-i} \right\}_{i=0}^{n},$$

$$\mathbf{b}(u) = \sum_{i=0}^{n} \mathbf{p}_i B_{n,i}(u), \;\; u \in [0,1]\,.$$



1.1. ábra. Harmadfokú Bernstein-polinomok ($u \in [0,1]$)

A teljesség igénye nélkül íme néhány tulajdonság (részletesebb leírásért ld. [Róth, 2018], [Farin, 2002], illetve [Juhász, 1995]), melyeket általában elvárunk egy keverőfüggvény-rendszertől, illetve az általa előállított görbéktől – hasonló tulajdonságok fogalmazhatók meg a felületek esetében is:

1. Affin transzformációtól való függetlenség: a görbe legyen invariáns az affin transzformációkra nézve, tehát ha $T : \mathbb{R}^\delta \to \mathbb{R}^\sigma$ egy affin transzformáció, akkor

   $$T(\mathbf{c}(u)) = T\left(\sum_{i=0}^{n} \mathbf{p}_i F_i(u)\right) = \sum_{i=0}^{n} T(\mathbf{p}_i) F_i(u), \quad \forall u \in [a,b]$$

   teljesüljön. A gyakorlatban használt keverőfüggvények egységfelbontást alkotnak, azaz

   $$\sum_{i=0}^{n} F_i(u) = 1, \forall u \in [a,b].$$

   Direkt számítással belátható, hogy ekkor a generált görbék és felületek függetlenek az affin transzformációktól.

2. Konvex bennfoglalás: ha az egységfelbontás mellett a függvények pozitívak is, azaz

   $$F_i(u) \geq 0, \quad \forall u \in [a,b], \quad \forall i \in \{1,2,...,n\},$$

   akkor a generált görbe mindig a kontrollpoligonja konvex burkának belsejében marad, hiszen az egyes görbepontok a kontrollpontok egy-egy konvex kombinációjaként állnak elő.

3. Monotonitás megőrzése: azért fontos, hogy a kontrollpoligon és a görbe vonalainak bejárási iránya

mindig megegyezzen. Azt mondjuk, hogy egy

$$F = \{F_i : [a,b] \to \mathbb{R}\}_{i=0}^{n}$$

függvényrendszer megőrzi a monotonitást, ha $\forall \lambda_i \in \mathbb{R}$ esetén

$$\lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_n \Rightarrow \sum_{i=0}^{n} \lambda_i F_i(u)$$

növekvő függvény ($u \in [a,b]$).

4. Végpontbeli interpoláció: az intuitív kezelhetőség érdekében azt is elvárhatjuk, hogy a

$$\mathbf{c}(a) = \sum_{i=0}^{n} \mathbf{p}_i F_i(a) = \mathbf{p}_0, \text{ illetve } \mathbf{c}(b) = \sum_{i=0}^{n} \mathbf{p}_i F_i(b) = \mathbf{p}_n$$

egyenlőségek teljesüljenek, azaz a görbe az első kontrollpontból induljon és az utolsó kontrollpontban érjen véget.

5. Lineáris függetlenség: ahhoz, hogy tetszőleges meglévő adatpontok interpolációját is biztosítani tudjuk, a függvényeket úgy kell megválasztani, hogy lineárisan függetlenek legyenek.

A kutatásban ezért az ismert és gyakran előforduló függvényterek (függvényekből álló lineáris vektorterek) esetében szoktak olyan bázist keresni, mely teljesíti a keverőfüggvényektől elvárt tulajdonságokat is (tehát modellezésre alkalmas). Bár a Bézier-görbék a legtöbb célra megfelelnek, polinomiális jellegük miatt bizonyos esetben pontatlanabb közelítést engednek csak meg, mint más függvénytér (gondolhatunk például arra, hogy a szinuszfüggvény közelítésére a polinomok kevésbé alkalmasak).

1.2. Kiterjesztett Csebisev-terek (KC-terek)

A [Róth, 2017]-ben leírtakat követve elevenítsük fel a kiterjesztett Csebisev függvényterek fogalmát. Legyen $n \geq 1$ egy rögzített egész szám és tekintsük a $\varphi_{n,i}(u)$ bázisfüggvényekből álló

$$\mathcal{F}_n^{\alpha,\beta} = \left\{\varphi_{n,i}(u) : u \in [\alpha,\beta]\right\}_{i=0}^{n}, \quad \varphi_{n,0} \equiv 1 \tag{1.3}$$

rendszert, ahol $\varphi_{n,i} \in C^n([\alpha,\beta])$ és $-\infty < \alpha < \beta < \infty$. Értelmezés szerint (ld. [Karlin és Studden, 1966]) azt mondjuk, hogy az

$$\mathbb{S}_n^{\alpha,\beta} := \left\langle \mathcal{F}_n^{\alpha,\beta} \right\rangle := \operatorname{span} \mathcal{F}_n^{\alpha,\beta} \tag{1.4}$$

$(n+1)$-dimenziós függvénytér kiterjesztett Csebisev-típusú, ha

- bármely $0 \leq r \leq n$ egész számra,
- bármely szigorúan növekvő $\alpha \leq u_0 < u_1 < \ldots < u_r \leq \beta$ csomópontrendszerre,

– tetszőleges $\{m_k\}_{k=0}^{r}$ egész számokra (multiplicitásokra) melyekre teljesül, hogy $\sum_{k=0}^{r} m_k = n+1$, és

– bármely $\left\{\xi_{k,l}\right\}_{k=0,\ l=0}^{r,\ m_k-1}$ valós számokra

mindig létezik egy és csakis egy

$$f := \sum_{i=0}^{n} \lambda_{n,i}\varphi_{n,i} \in \mathbb{S}_n^{\alpha,\beta}, \ \ \lambda_{n,i} \in \mathbb{R}, \ \ i = 0, 1, ..., n \tag{1.5}$$

függvény, mely kielégíti a Hermite-féle

$$f^{(l)}(u_k) = \xi_{k,l}, \quad l = 0, 1, ..., m_k - 1, \quad k = 0, 1, ..., r \tag{1.6}$$

interpolációs feladat feltételeit és ezen egyenletrendszer együtthatómátrixának előjeltartó determinánsa szigorúan pozitív bármely, a fentiekben leírt megengedett paraméterbeállítás esetén. A KC-tér értelmezését tekintve az $1 \equiv \varphi_{n,0} \in \mathbb{S}_n^{\alpha,\beta}$ feltétel nem szükséges, jelen esetben viszont a konstansokat is az $\mathbb{S}_n^{\alpha,\beta}$ részévé tettük annak érdekében, hogy minden bázisa normalizálható legyen.

Zérushelyek szempontjából a definícióból következik, hogy bármely nem azonosan nulla eleme $\mathbb{S}_n^{\alpha,\beta}$-nak legfeljebb n-szer válik nullává az $[\alpha, \beta]$ intervallumon.

A továbbiakban $\mathcal{F}_n^{\alpha,\beta}$-ra az $\mathbb{S}_n^{\alpha,\beta}$ *hagyományos bázisa*ként hivatkozunk, és feltételezzük, hogy a deriváltak n-dimenziós $D\mathbb{S}_n^{\alpha,\beta} := \left\{f^{(1)} : f \in \mathbb{S}_n^{\alpha,\beta}\right\}$ tere szintén KC-típusú az $[\alpha, \beta]$ intervallumon. A [Carnicer és Peña, 1995]-nek az 5.1. tételét, illetve a [Carnicer et al., 2004]-nek a 4.1. tételét felhasználva kijelenthetjük, hogy az $\mathbb{S}_n^{\alpha,\beta}$ vektortérnek ezen feltételek mellett létezik szigorúan teljesen pozitív bázisa is (azaz olyan bázisa, melynek minden kollokációs mátrixának minden minorja szigorúan pozitív). Mivel a konstans $1 \equiv \varphi_{n,0}$ függvény eleme $\mathbb{S}_n^{\alpha,\beta}$-nak, az említett szigorúan pozitív bázis normalizálható, tehát a vektortérnek van egy egyedi normalizált B-bázisa (úgynevezett NB-bázisa):

$$\mathcal{B}_n^{\alpha,\beta} = \left\{b_{n,i}(u) : u \in [\alpha, \beta]\right\}_{i=0}^{n}. \tag{1.7}$$

A [Carnicer és Peña, 1995] 5.1. tételének és a [Mazure, 1999] (3.6)-os egyenletének megfelelően a $\mathcal{B}_n^{\alpha,\beta}$ bázis az egységfelbontás ($\sum_{i=0}^{n} b_{n,i}(u) \equiv 1, \ \ \forall u \in [\alpha, \beta]$) mellett teljesíti a következő tulajdonságokat is:

$$b_{n,0}(\alpha) = b_{n,n}(\beta) = 1, \tag{1.8}$$

$$b_{n,i}^{(j)}(\alpha) = 0, \ \ j = 0, 1, ..., i-1, \ \ b_{n,i}^{(i)}(\alpha) > 0, \tag{1.9}$$

$$b_{n,i}^{(j)}(\beta) = 0, \ \ j = 0, 1, ..., n-1-i, \ \ (-1)^{n-i} b_{n,i}^{(n-i)}(\beta) > 0. \tag{1.10}$$

A továbbiakban bemutatott algoritmusok érvényesek bármilyen $\mathbb{S}_n^{\alpha,\beta}$ KC-tér esetén amely teljesíti a

fentebbi feltételeket, az implementációjukban viszont feltételezzük, hogy $\mathbb{S}_n^{\alpha,\beta}$ az $(n+1)$-edrendű

$$\sum_{i=0}^{n+1} \gamma_i v^{(i)}(u) = 0, \;\; \gamma_i \in \mathbb{R}, \;\; u \in [\alpha, \beta] \tag{1.11}$$

konstansegyüttható homogén lineáris differenciálegyenlet megoldástere.

A megoldásteret a differenciálegyenlet karakterisztikus polinomjának (esetleg magasabb rendű) gyökei által meghatározott hagyományos bázisfüggvények feszítik ki. A karakterisztikus polinom

$$p_{n+1}(z) = \sum_{i=0}^{n+1} \gamma_i z^i, \;\; z \in \mathbb{C} \tag{1.12}$$

alakba írható és mivel biztosítani akarjuk, hogy $\varphi_{n,0} \equiv 1 \in \mathbb{S}_n^{\alpha,\beta}$, meg fogjuk követelni, hogy $z = 0$ a karakterisztikus polinomnak legalább egyszeres gyöke legyen.

A megoldástér eltolásfüggetlen, $C^\infty([\alpha, \beta])$ osztályú és a megfelelően kicsi $\beta - \alpha \in (0, l_n)$ hosszúságú intervallumra vett leszűkítése KC-típusú, ahol $l_n > 0$ az úgynevezett kritikus hossz, melyet a következőképpen tudunk meghatározni (ld. [Carnicer et al., 2004, 3.1. kijelentés]):

– jelölje

$$W_{[v_{n,0}, v_{n,1}, \ldots, v_{n,n}]}(u) := \left[v_{n,i}^{(j)}(u) \right]_{i=0,\, j=0}^{n,\, n}, \;\; u \in [\alpha, \beta] \tag{1.13}$$

az (1.11) azon sajátos

$$v_{n,i} := \sum_{k=0}^{n} \rho_{i,k} \varphi_{n,k} \in \mathbb{S}_n^{\alpha,\beta}, \;\; \left\{ \rho_{i,k} \right\}_{k=0}^{n} \subset \mathbb{R}, \;\; i = 0, 1, \ldots, n \tag{1.14}$$

integráljainak Wronski-féle mátrixát, melyek teljesítik a

$$\begin{cases} v_{n,i}^{(j)}(\alpha) = 0, & j = 0, 1, \ldots, i-1, \\ v_{n,i}^{(i)}(\alpha) = 1, & \\ v_{n,i}^{(j)}(\beta) = 0, & j = 0, 1, \ldots, n-1-i \end{cases} \tag{1.15}$$

kezdeti feltételeket, tehát a $\{v_{n,i}(u) : u \in [\alpha, \beta]\}_{i=0}^{n}$ egy bikanonikus bázis az $[\alpha, \beta]$ intervallum felett, úgy hogy az (1.13)-as Wronski-féle mátrix $u = \alpha$-ban egy alsó háromszögmátrix, az átlón pozitív (egység) elemekkel;

– tekintsük a következő függvényeket (Wronski-féle determinánsokat):

$$w_{n,i}(u) := \det W_{[v_{n,i}, v_{n,i+1}, \ldots, v_{n,n}]}(u), \;\; i = \left\lfloor \tfrac{n}{2} \right\rfloor + 1, \ldots, n, \tag{1.16}$$

$$\theta_{n,i}(u) := (-1)^{n(n+1-i)} \det W_{[v_{n,i}, v_{n,i+1}, \ldots, v_{n,n}]}(-u), \;\; i = \left\lfloor \tfrac{n}{2} \right\rfloor + 1, \ldots, n, \tag{1.17}$$

és értelmezzük az

$$l_n := \min_{i=\left\lfloor \frac{n}{2} \right\rfloor +1,\ldots,n} \min \left\{ |u-\alpha| : w_{n,i}(u) = 0 \text{ vagy } \theta_{n,i}(u) = 0,\ u \neq \alpha \right\} \tag{1.18}$$

kritikus hosszt (írhatjuk, hogy $l_n = +\infty$ abban az esetben, ha az (1.16)-os Wronski-féle determinánsoknak nincs α-tól különböző valós gyöke, mitöbb l_n végtelen, ha az (1.12)-es karakterisztikus polinomnak csak valós gyökei vannak, illetve véges, ha a karakterisztikus polinomnak tényleges - nem eltűnő képzetes résszel rendelkező - komplex gyökei is vannak).

A deriváltak $D\mathbb{S}_n^{\alpha,\beta}$ terének kritikus hosszát jelölje l'_n. Ahhoz, hogy biztosítsuk az NB-bázis létezését $\mathbb{S}_n^{\alpha,\beta}$-ban (ld. [Carnicer et al., 2004, 4.1. tétel]) a továbbiakban mindig feltételezzük, hogy $\beta \in (\alpha, \alpha + l'_n)$. A $\beta - \alpha \in (0, l'_n)$ intervallumhossz egy alakparaméternek is felfogható. Az egyértelműség érdekében l_n és l'_n mellett az $l\left(\mathbb{S}_n^{\alpha,\beta}\right)$ és $l'\left(\mathbb{S}_n^{\alpha,\beta}\right) := l\left(D\mathbb{S}_n^{\alpha,\beta}\right)$ jelöléseket is használni fogjuk.

A hagyományos bázis alakja megkapható a gyökökből a differenciálegyenletek megoldásánál megszokott módon:

- ha $z_0 \in \mathbb{R}$ egy m-szeres gyöke az (1.12)-es karakterisztikus polinomnak ($m \geq 1$), akkor a hozzá tartozó bázisfüggvények $u^k e^{z_0 u}$ alakúak ($k = 0, 1, ..., m-1$),
- ha $z_0 = a + b\mathbf{i} \in \mathbb{C} \setminus \mathbb{R}$ $(a, b \in \mathbb{R},\ b \neq 0,\ \mathbf{i}^2 = -1)$ egy m-szeres gyöke a polinomnak, akkor az $a - b\mathbf{i}$ szintén m-szeres gyöke és a $2m$ darab gyök által meghatározott bázisfüggvények:

$$\left\{ u^k e^{au} \cos(bu),\ u^k e^{au} \sin(bu) \right\}_{k=0}^{m-1},\ u \in [\alpha, \beta].$$

A bemutatott algoritmusok nem feltételezik, hogy (1.12) egy páros vagy páratlan függvény, de ha ezek egyike teljesül, akkor az általa meghatározott KC-tér tükrözésinvariáns is lesz. Tekintsük a [Róth, 2017]-ben bemutatott példákat. Ha $\{\omega_k\}_{k=1}^n$ páronként különböző valós számok és az $[\alpha, \beta]$ értelmezési tartomány hossza megfelelően rögzített, akkor a

$$\begin{aligned} p_{n+1}(z) &= z^{n+1}, \\ p_{(n+1)^2}(z) &= z^{n+1} \prod_{k=1}^{n} (z^2 + \omega_k^2)^{n+1-k}, \text{ illetve} \\ p_{(n+1)^2}(z) &= z^{n+1} \prod_{k=1}^{n} (z^2 - \omega_k^2)^{n+1-k} \end{aligned}$$

karakterisztikus polinomok által meghatározott tiszta polinomiális, kevert algebrai-trigonometrikus és

algebrai-hiperbolikus KC-terek:

$$\begin{aligned}
\mathbb{P}_n^{\alpha,\beta} &:= \left\langle \mathcal{P}_n^{\alpha,\beta} \right\rangle := \left\langle\, \{1, u, ..., u^n : u \in [\alpha, \beta]\} \,\right\rangle, \\
&\dim \mathbb{P}_n^{\alpha,\beta} = n+1, \\
\mathbb{AT}_{n(n+2)}^{\alpha,\beta} &:= \left\langle \mathcal{P}_n^{\alpha,\beta} \cup \left\{ u^l \cos(\omega_k u), u^l \sin(\omega_k u) \right\}_{k=1,\, l=0}^{n,\, n-k} \right\rangle, \\
&\dim \mathbb{AT}_{n(n+2)}^{\alpha,\beta} = (n+1)^2,
\end{aligned}$$

illetve

$$\begin{aligned}
\mathbb{AH}_{n(n+2)}^{\alpha,\beta} &:= \left\langle \mathcal{P}_n^{\alpha,\beta} \cup \left\{ u^l \cosh(\omega_k u), u^l \sinh(\omega_k u) \right\}_{k=1,\, l=0}^{n,\, n-k} \right\rangle, \\
&\dim \mathbb{AH}_{n(n+2)}^{\alpha,\beta} = (n+1)^2,
\end{aligned}$$

melyek nemcsak transzláció-, hanem tükrözésinvariánsok is és természetesen rendelkeznek egyedi normalizált B-bázissal.

Ha az NB-bázisfüggvényeket ismerjük, megfelelő számú kontrollpontot felvéve a szokványos módon előállíthatjuk az úgynevezett B-görbéket és B-felületeket.

1.1. Értelmezés (B-görbe)**.** A

$$\mathbf{c}_n(u) = \sum_{i=0}^{n} \mathbf{p}_i b_{n,i}(u), \quad u \in [\alpha, \beta], \quad \mathbf{p}_i = \left[p_i^l \right]_{l=0}^{\delta-1} \in \mathbb{R}^\delta \tag{1.19}$$

konvex kombinációt n-edrendű B-görbének nevezzük, ahol $[\mathbf{p}_i]_{i=0}^n$ a kontrollpoligont jelöli.

1.2. Értelmezés (B-felület)**.** Ha

$$\mathcal{B}_{n_r}^{\alpha_r,\beta_r} = \left\{ b_{n_r,i_r}(u_r) : u_r \in [\alpha_r, \beta_r] \right\}_{i_r=0}^{n_r}, \quad r = 0, 1$$

jelöli két KC-tér NB-bázisát, akkor az (1.19) típusú görbék tenzorszorzataként értelmezhetjük az

$$\begin{aligned}
&\mathbf{s}_{n_0,n_1}(u_0, u_1) = \sum_{i_0=0}^{n_0} \sum_{i_1=0}^{n_1} \mathbf{p}_{i_0,i_1} b_{n_0,i_0}(u_0)\, b_{n_1,i_1}(u_1), \\
&u_0 \in [\alpha_0, \beta_0]\,, \;\; u_1 \in [\alpha_1, \beta_1]\,, \;\; \mathbf{p}_{i_0,i_1} = \left[p_{i_0,i_1}^l \right]_{l=0}^{2} \in \mathbb{R}^3
\end{aligned} \tag{1.20}$$

B-felületet, ahol a $[\mathbf{p}_{i_0,i_1}]_{i_0=0,\, i_1=0}^{n_0,\, n_1}$ mátrix alkotja a kontrollhálót.

1.3. Az NB-bázisfüggvények (nullad- és magasabb rendű deriváltjainak) előállítása

Amint definiáltunk egy $\mathbb{S}_n^{\alpha,\beta}$ KC-teret az $\mathcal{F}_n^{\alpha,\beta}$ hagyományos bázisának megadása által (mely a karakterisztikus polinom gyökeinek rögzítésével történik), elő kell állítanunk a $\mathcal{B}_n^{\alpha,\beta}$ egyedi normalizált B-bázisát. Az NB-bázisfüggvények (nullad- és magasabb rendű) deriváltjai ugyanis nemcsak a görbe- és felületpontok meghatározásában, hanem a később bemutatott algoritmusok számításaiban is megjelennek. A továbbiakban az NB-bázis [Róth, 2017]-ben összefoglalt felépítési eljárását mutatjuk be.

Az $\mathbb{S}_n^{\alpha,\beta}$ KC-tér esetében a $\beta - \alpha \in (0, l'_n)$ feltétel teljesülése biztosítja a $\mathcal{B}_n^{\alpha,\beta}$ NB-bázis létezését, ennek felépítése ekkor a [Carnicer et al., 2004]-ben leírt eljárás segítségével valósítható meg.

Tekintsük a bikanonikus $\left\{v_{n,i}(u) \,:\, u \in [\alpha,\beta]\right\}_{i=0}^{n}$ bázist, melyet a az (1.15)-ös feltételek által meghatározott (1.14)-es vonalintegrálok alkotnak. Legyen $W_{[v_{n,n},v_{n,n-1},\ldots,v_{n,0}]}(\beta)$ a fordított sorrendbe rendezett $\left\{v_{n,n-i}(u) : u \in [\alpha,\beta]\right\}_{i=0}^{n}$ rendszer $u=\beta$-ban vett Wronski-féle mátrixa és határozzuk meg ennek az

$$L \cdot U = W_{[v_{n,n},v_{n,n-1},\ldots,v_{n,0}]}(\beta)$$

Doolittle-féle LU felbontását ahol L egy alsó háromszögmátrix egyesekkel az átlóján, U pedig egy nemszinguláris felső háromszögmátrix.

Számítsuk ki ezek inverzét:

$$U^{-1} := \begin{bmatrix} \mu_{0,0} & \mu_{0,1} & \cdots & \mu_{0,n} \\ 0 & \mu_{1,1} & \cdots & \mu_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_{n,n} \end{bmatrix}, \quad L^{-1} := \begin{bmatrix} \lambda_{0,0} & 0 & \cdots & 0 \\ \lambda_{1,0} & \lambda_{1,1} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ \lambda_{n,0} & \lambda_{n,1} & \cdots & \lambda_{n,n} \end{bmatrix},$$

melyek segítségével fel tudjuk építeni az NB-bázist:

$$\mathcal{B}_n^{\alpha,\beta} = \left\{ b_{n,i}(u) = \lambda_{n-i,0}\widetilde{b}_{n,i}(u) : u \in [\alpha,\beta] \right\}_{i=0}^{n}, \tag{1.21}$$

ahol

$$\begin{bmatrix} \widetilde{b}_{n,n}(u) & \widetilde{b}_{n,n-1}(u) & \cdots & \widetilde{b}_{n,0}(u) \end{bmatrix} = \begin{bmatrix} v_{n,n}(u) & v_{n,n-1}(u) & \cdots & v_{n,0}(u) \end{bmatrix} \cdot U^{-1},$$

illetve

$$\begin{bmatrix} \lambda_{0,0} & \lambda_{1,0} & \cdots & \lambda_{n,0} \end{bmatrix} = L^{-1} \cdot \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^T.$$

Ha az (1.12)-es karakterisztikus polinom páros vagy páratlan, akkor az általa meghatározott $\mathbb{S}_n^{\alpha,\beta}$ KC-tér invariáns a tükrözésekre, így ebben a sajátos esetben a

$$b_{n,i}(u) = b_{n,n-i}(\alpha+\beta-u), \;\; \forall u \in [\alpha,\beta], \;\; i = 0,1,\ldots,\left\lfloor \tfrac{n}{2} \right\rfloor$$

szimmetria is teljesül, ami azt jelenti, hogy elég csak a bázisfüggvények felét meghatározni (1.21)-ben.

1.3. Tétel (B-bázisfüggvények deriváltjai, [Róth, 2017, 2.3. tétel])**.** *Az (1.21)-ben megjelenő bázisfügg-*

vények (nullad- és magasabb rendű) deriváltjainak kiszámítása a következő számításra redukálódik:

$$\begin{aligned} b_{n,n-i}^{(j)}(u) &= \lambda_{i,0}\widetilde{b}_{n,n-i}^{(j)}(u) \\ &= \lambda_{i,0}\sum_{r=0}^{i}\mu_{r,i}v_{n,n-r}^{(j)}(u) \\ &= \lambda_{i,0}\sum_{r=0}^{i}\mu_{r,i}\sum_{k=0}^{n}\rho_{n-r,k}\varphi_{n,k}^{(j)}(u), \;\; \forall u\in[\alpha,\beta], \;\; i=0,1,...,n. \end{aligned} \tag{1.22}$$

Ha a KC-tér tükrözésinvariáns, akkor az (1.22)-es képletet csak az $i = 0, 1, ..., \left\lfloor \frac{n}{2} \right\rfloor$ indexekre kell alkalmazni, mivel ez esetben teljesül, hogy:

$$b_{n,i}^{(j)}(u) = (-1)^j b_{n,n-i}^{(j)}(\alpha+\beta-u), \;\; \forall u\in[\alpha,\beta], \;\; i=0,1,...,\left\lfloor \tfrac{n}{2} \right\rfloor. \tag{1.23}$$

1.4. Példa (Egy kilencdimenziós tükrözésinvariáns KC-tér)**.** Tekintsük a

$$v^{(9)}(u)+6v^{(7)}(u)+9v^{(5)}(u)+4v^{(3)}(u)=0, \;\; u\in\left[-\tfrac{\pi}{2},\tfrac{\pi}{2}\right]$$

differenciálegyenletet, majd bontsuk tényezőkre a karakterisztikus polinomját:

$$\begin{aligned} p_9(z) &= z^9+6z^7+9z^5+4z^3 \\ &= z^3\left(z^2+1\right)^2\left(z^4+2^2\right), \;\; z\in\mathbb{C}. \end{aligned}$$

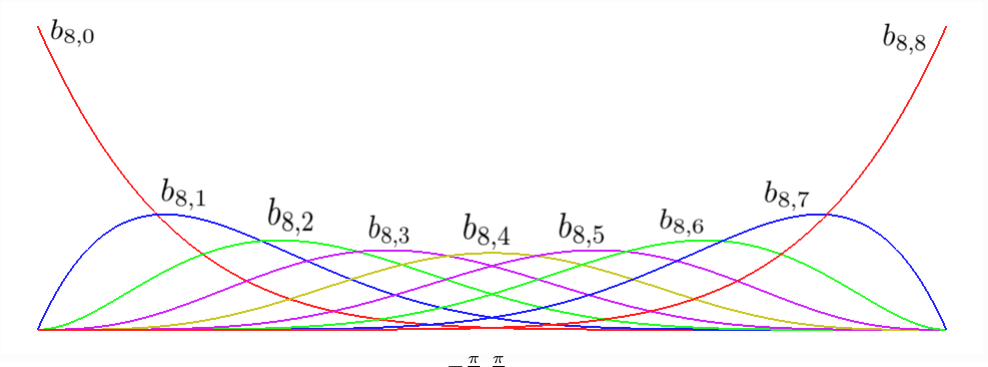
A gyökök és multiplicitásaik által meghatározott bázisfüggvények az

$$\begin{aligned} \mathbb{AT}_8^{-\frac{\pi}{2},\frac{\pi}{2}} &= \left\langle \mathcal{F}_8^{-\frac{\pi}{2},\frac{\pi}{2}} \right\rangle \\ &= \Big\langle \Big\{ \varphi_{8,0}(u)=1, \;\; \varphi_{8,1}(u)=u, \;\; \varphi_{8,2}(u)=u^2, \\ &\qquad \varphi_{8,3}(u)=\cos(u), \;\; \varphi_{8,4}(u)=\sin(u), \;\; \varphi_{8,5}(u)=u\cos(u), \\ &\qquad \varphi_{8,6}(u)=u\sin(u), \;\; \varphi_{8,7}(u)=\cos(2u), \;\; \varphi_{8,8}(u)=\sin(2u) \Big\} \Big\rangle \end{aligned} \tag{1.24}$$

9-dimenziós algebrai-trigonometrikus teret állítják elő (mely tükrözésinvariáns is, mivel a karakterisztikus polinom páratlan). Ha bemenetként megadjuk p_9 (esetleg magasabb rendű) gyökeit, az alkalmazás ki tudja számolni és meg tudja jeleníteni a KC-térhez tartozó NB-bázisfüggvényeket, melyek az 1.2. ábrán láthatók (a függvényeket az alkalmazás segítségével jelenítettük meg, majd utófeldolgozásként címkéztük).

1.5. Példa (Egy hatdimenziós KC-tér, mely nem tükrözésinvariáns)**.** Tekintsük a

$$v^{(6)}(u)-6v^{(5)}(u)+13v^{(4)}(u)-12v^{(3)}(u)+4v^{(2)}(u)=0, \;\; u\in[-1,1]$$



1.2. ábra. Az (1.24)-es $\mathbb{AT}_8^{-\frac{\pi}{2},\frac{\pi}{2}}$ KC-tér NB bázisfüggvényei.

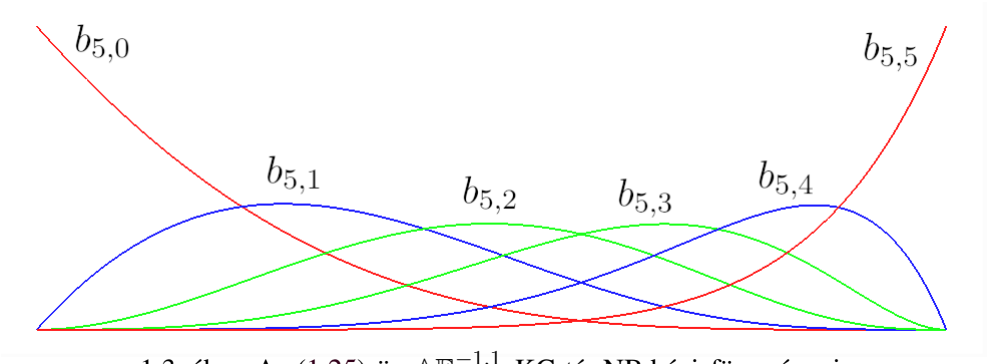
differenciálegyenletet, melynek karakterisztikus polinomja:

$$p_6(z) = z^2\,(z-1)^2\,(z-2)^2\,, \quad z \in \mathbb{C}.$$

A gyökök által meghatározott bázisfüggvények az

$$\begin{aligned}\mathbb{AE}_5^{-1,1} = \Big\langle \Big\{ &\varphi_{5,0}(u) = 1,\ \varphi_{5,1}(u) = u,\ \varphi_{5,2}(u) = e^u,\ \varphi_{5,3}(u) = ue^u,\\ &\varphi_{5,4}(u) = e^{2u},\ \varphi_{5,5}(u) = ue^{2u} \Big\} \Big\rangle\end{aligned} \tag{1.25}$$

6-dimenziós vegyes algebrai-exponenciális teret állítják elő, melynek NB-bázisfüggvényei az 1.3. ábrán láthatók. Mivel a karakterisztikus polinom nem is páros és nem is páratlan függvény, az előálló tér nem lesz tükrözésinvariáns, amint az a bázisfüggvények alakjából is látszik.



1.3. ábra. Az (1.25)-ös $\mathbb{AE}_5^{-1,1}$ KC-tér NB bázisfüggvényei.

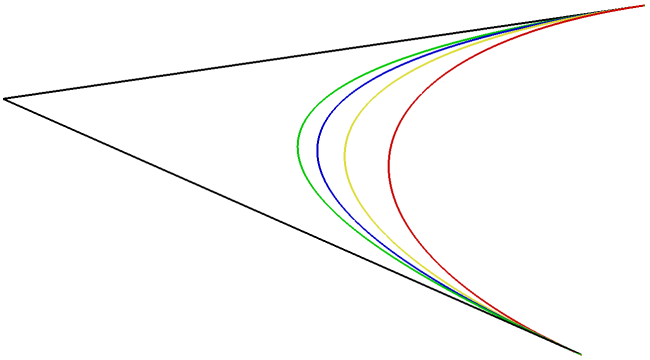
1.6. Példa (A $\beta - \alpha$ intervallumhossz alakváltoztatási hatása)**.** Tekintsük a

$$p(z) = z^3 + z^2 = z(z^2+1) \tag{1.26}$$

karakterisztikus polinom által generált

$$\mathbb{T}_1^{\alpha,\beta} = \langle \{1,\, \sin(u),\, \cos(u)\} \rangle \tag{1.27}$$

elsőrendű trigonometrikus teret, melynek kritikus hossza $l'_1 = \pi$. Legyen $\alpha = 0$, a β végpontnak pedig feleltessük meg rendre a $\{0.1,\, 1,\, 1.5,\, 2\}$ halmaz elemeit. Rögzített kontrollpontok esetén a különböző β értékeknek megfelelő terek NB-bázisfüggvényei által generált görbék alakja különböző (a $\beta - \alpha$ intervallumhossz tehát alakparaméternek fogható fel). Ezt a hatást szemlélteti az 1.4. ábra.



1.4. ábra. A β paraméter hatása az (1.27)-es KC-tér által generált görbe alakjára. A megjelenített (közös kontrollpontokkal rendelkező) görbék esetében felhasznált értékek: $\beta_1 = 0.1,\ \beta_2 = 1,\ \beta_3 = 1.5,\ \beta_4 = 2$.

1.4. B-görbék és B-felületek rendszámnövelése

A rendszámnövelés célja, hogy a kontrollpontok száma nőjön (azaz finomabb alakváltoztatást tudjunk megengedni a felhasználónak) úgy, hogy az új kontrollpontok által leírt görbe vagy felület az eredetivel megegyező maradjon. A következőkben a [Róth, 2017]-ben leírtak mentén haladva egy numerikus eljárást tárgyalunk, amellyel B-görbék és B-felületek esetén megvalósítható a rendszámnövelés.

Tekintsük az $\mathbb{S}_n^{\alpha,\beta}$ és $\mathbb{S}_{n+1}^{\alpha,\beta}$ KC-tereket úgy, hogy $1 \in \mathbb{S}_n^{\alpha,\beta} \subset \mathbb{S}_{n+1}^{\alpha,\beta}$ és tételezzük fel, hogy a deriváltak $D\mathbb{S}_n^{\alpha,\beta}$ és $D\mathbb{S}_{n+1}^{\alpha,\beta}$ terei is KC-típusúak az $[\alpha,\beta]$ intervallumon, azaz $0 < \beta - \alpha < \min\left\{ l'\left(\mathbb{S}_n^{\alpha,\beta}\right),\ l'\left(\mathbb{S}_{n+1}^{\alpha,\beta}\right) \right\}$. A terek NB-bázisait jelölje $\left\{b_{n,i}(u) : u \in [\alpha,\beta]\right\}_{i=0}^{n}$ és $\left\{b_{n+1,i}(u) : u \in [\alpha,\beta]\right\}_{i=0}^{n+1}$.

A következő tétel a [Mazure és Laurent, 1998] 3.1. tételének kissé átalakított változata (ld. [Róth, 2017, 2.7. lemma]), melyben az $[\alpha,\beta]$ értelmezési tartomány mindkét végpontját figyelembe vettük annak érdekében, hogy a megjelenő maximális deriválási rend a lehető legkisebb legyen (ez a hatékonyság és numerikus stabilitás miatt is fontos).

1.7. Tétel (Általános rendszámnövelés)**.** *A fentebbi jelöléseket felhasználva, a* $\mathbf{c}_n$ *n-edrendű B-görbére*

fennáll, hogy

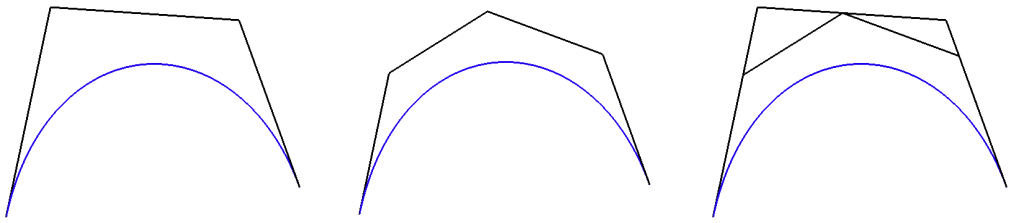
$$\mathbf{c}_n(u) = \sum_{i=0}^{n} \mathbf{p}_i b_{n,i}(u) \equiv \sum_{i=0}^{n+1} \mathbf{p}_{1,i} b_{n+1,i}(u) =: \mathbf{c}_{n+1}(u), \quad \forall u \in [\alpha, \beta],$$

ahol $\mathbf{p}_{1,0} = \mathbf{p}_0,\ \mathbf{p}_{1,n+1} = \mathbf{p}_n$ *és*

$$\mathbf{p}_{1,i} = \left(1 - \frac{b^{(i)}_{n,i}(\alpha)}{b^{(i)}_{n+1,i}(\alpha)}\right)\mathbf{p}_{i-1} + \frac{b^{(i)}_{n,i}(\alpha)}{b^{(i)}_{n+1,i}(\alpha)}\mathbf{p}_i, \quad i = 1, 2, ..., \left\lfloor \frac{n}{2} \right\rfloor, \tag{1.28}$$

$$\mathbf{p}_{1,n+1-i} = \frac{b^{(i)}_{n,n-i}(\beta)}{b^{(i)}_{n+1,n+1-i}(\beta)}\mathbf{p}_{n-i} + \left(1 - \frac{b^{(i)}_{n,n-i}(\beta)}{b^{(i)}_{n+1,n+1-i}(\beta)}\right)\mathbf{p}_{n+1-i}, \quad i = 1, 2, ..., \left\lfloor \frac{n+1}{2} \right\rfloor. \tag{1.29}$$

Bár az 1.7. tétel érvényes bármely olyan egymásba ágyazott $1 \in \mathbb{S}_n^{\alpha,\beta} \subset \mathbb{S}_{n+1}^{\alpha,\beta}$ terek esetében, melyekre a $D\mathbb{S}_n^{\alpha,\beta}$ és $D\mathbb{S}_{n+1}^{\alpha,\beta}$ terek is KC-típusúak, az implementációban feltételezni fogjuk, hogy a magasabb rendű $\mathbb{S}_{n+1}^{\alpha,\beta}$ tér is egy konstansegyüttható s lineáris homogén differenciálegyenlet megoldástere. Természetesen az 1.7. tétel kijelentése kiterjeszthető az (1.20) típusú felületek rendszámnövelésére is, amit az alkalmazással szintén elvégezhetünk.



(a) (b) (c)

1.5. ábra. Egy B-görbe és kontrollpoligonja rendszámnövelés előtt (a) és rendszámnövelés után (b). A harmadik ábrát (c) az előző kettő egymásra tételéből kaptuk. A görbe mögötti vegyes algebrai-exponenciális KC-teret meghatározó $p_4(z) = z^3(z-1)$ kiindulási karakterisztikus polinom esetében a 0 gyök multiplicitását eggyel növeltük. Látható, hogy míg a kontrollpontok száma eggyel nő, a görbe alakja változatlan marad.

1.5. B-görbék és B-felületek felosztási algoritmusa

A következőkben a B-görbék és B-felületek valamely, az értelmezési tartomány egyik belső pontjának megfelelő görbepont (illetve paramétergörbe) körüli felosztását szeretnénk megvalósítani. Ismertetjük a [Róth, 2017]-ben bemutatott módszert.

Minden normalizált B-bázisnak megfelel egy úgynevezett B-algoritmus az (1.19) típusú B-görbék felosztására, azaz egy tetszőlegesen lerögzített $\gamma \in (\alpha, \beta)$ paraméterérték esetén létezik egy, a klasszikus Bézier-görbék de Casteljau algoritmusához hasonló sorozatos saroklevágásokra épülő rekurzív felosztási

eljárás, mely a $\mathbf{p}_i^0(\gamma) = \mathbf{p}_i,\ i = 0, 1, ..., n$ kezdeti feltételekből indul és meghatározza a

$$\mathbf{p}_i^j(\gamma) = \Big[1 - \xi_i^j(\gamma)\Big] \cdot \mathbf{p}_i^{j-1}(\gamma) + \xi_i^j(\gamma) \cdot \mathbf{p}_{i+1}^{j-1}(\gamma),\ \ i = 0, 1, ..., n-j,\ \ j = 1, 2, ..., n \tag{1.30}$$

felosztási pontokat, ahol a $\left\{\xi_i^j : [\alpha, \beta] \to [0, 1]\right\}_{i=0,\ j=1}^{n-j,\ n}$ keverőfüggvények explicit zárt alakja általában nem ismert, vagy speciális esetek (például Bézier-görbék) kivételével bonyolult nemlineáris alakot öltenek, még kis dimenziószámú KC-terek esetén is.

Legyen $\mathbb{S}_n^{\alpha,\beta}$ egy KC-tér, úgy hogy $1 \in \mathbb{S}_n^{\alpha,\beta}$ és $\beta - \alpha \in (0, l'_n)$, továbbá legyenek $\mathcal{B}_n^{\alpha,\gamma} := \left\{b_{n,i}(u; \alpha, \gamma) : u \in [\alpha, \gamma]\right\}_{i=0}^n$ és $\mathcal{B}_n^{\gamma,\beta} := \left\{b_{n,i}(u; \gamma, \beta) : u \in [\gamma, \beta]\right\}_{i=0}^n$ a leszűkített $\mathbb{S}_n^{\alpha,\gamma} := \mathrm{span}\mathcal{F}_n^{\alpha,\gamma} := \mathrm{span}\mathcal{F}_n^{\alpha,\beta}\Big|_{[\alpha,\gamma]}$ és $\mathbb{S}_n^{\gamma,\beta} := \mathrm{span}\mathcal{F}_n^{\gamma,\beta} := \mathrm{span}\mathcal{F}_n^{\alpha,\beta}\Big|_{[\gamma,\beta]}$ KC-terek NB-bázisai. Tekintsük a következő, (1.30)-as eljárásnak megfelelő háromszögséma átlós $\left\{\boldsymbol{\lambda}_i(\gamma) := \mathbf{p}_0^i(\gamma)\right\}_{i=0}^n$ és $\left\{\boldsymbol{\varrho}_i(\gamma) := \mathbf{p}_i^{n-i}(\gamma)\right\}_{i=0}^n$ bejegyzéseit:

$$\begin{array}{llll}
\mathbf{p}_0 =: \boldsymbol{\lambda}_0(\gamma) & & & \\
\mathbf{p}_1 & \mathbf{p}_0^1(\gamma) =: \boldsymbol{\lambda}_1(\gamma) & & \\
\mathbf{p}_2 & \mathbf{p}_1^1(\gamma) & \mathbf{p}_0^2(\gamma) =: \boldsymbol{\lambda}_2(\gamma) & \\
\vdots & \vdots & \vdots & \cdots\ \ \mathbf{p}_0^n(\gamma) =: \boldsymbol{\lambda}_n(\gamma) =: \boldsymbol{\varrho}_0(\gamma) \\
\mathbf{p}_{n-2} & \mathbf{p}_{n-2}^1(\gamma) & \mathbf{p}_{n-2}^2(\gamma) =: \boldsymbol{\varrho}_{n-2}(\gamma) & \\
\mathbf{p}_{n-1} & \mathbf{p}_{n-1}^1(\gamma) =: \boldsymbol{\varrho}_{n-1}(\gamma) & & \\
\mathbf{p}_n =: \boldsymbol{\varrho}_n(\gamma) & & &
\end{array}$$

Ezeket a pontokat konvex módon kombinálva a $\mathcal{B}_n^{\alpha,\gamma}$ és $\mathcal{B}_n^{\gamma,\beta}$ bázisok függvényeivel, a B-görbe a bal és jobb oldali

$$\mathbf{l}_n(u) := \sum_{i=0}^n \boldsymbol{\lambda}_i(\gamma) \cdot b_{n,i}(u; \alpha, \gamma) \equiv \mathbf{c}_n(u),\ \ \forall u \in [\alpha, \gamma], \tag{1.31}$$

illetve

$$\mathbf{r}_n(u) := \sum_{i=0}^n \boldsymbol{\varrho}_i(\gamma) \cdot b_{n,i}(u; \gamma, \beta) \equiv \mathbf{c}_n(u),\ \ \forall u \in [\gamma, \beta] \tag{1.32}$$

ívre osztható, melyekre teljesül az is, hogy:

$$\mathbf{l}_n^{(j)}(u) = \mathbf{c}_n^{(j)}(u),\ \ \forall u \in [\alpha, \gamma], \tag{1.33}$$

$$\mathbf{r}_n^{(j)}(u) = \mathbf{c}_n^{(j)}(u),\ \ \forall u \in [\gamma, \beta], \tag{1.34}$$

minden $j \geq 0$ deriválási rendre.

A következőkben ismertetjük a [Róth, 2017]-ben bemutatott rekurzív eljárást, mellyel kiszámíthatók a $\{\boldsymbol{\lambda}_i(\gamma)\}_{i=0}^n$ és $\{\boldsymbol{\varrho}_i(\gamma)\}_{i=0}^n$ osztópontok a $\left\{\xi_i^j : [\alpha, \beta] \to [0, 1]\right\}_{i=0,\ j=1}^{n-j,\ n}$ keverőfüggvények ismerete nélkül is.

1.8. Tétel (Általános B-algoritmus)**.** *Tetszőlegesen rögzített $\gamma \in (\alpha, \beta)$ paraméterérték és a*

$$\boldsymbol{\lambda}_0(\gamma) = \mathbf{c}_n(\alpha) = \mathbf{p}_0, \tag{1.35}$$

$$\boldsymbol{\lambda}_n(\gamma) = \mathbf{c}_n(\gamma) = \boldsymbol{\varrho}_0(\gamma), \tag{1.36}$$

$$\boldsymbol{\varrho}_n(\gamma) = \mathbf{c}_n(\beta) = \mathbf{p}_n \tag{1.37}$$

kezdeti feltételek esetén az ismeretlen $\{\boldsymbol{\lambda}_i(\gamma)\}_{i=0}^n$ és $\{\boldsymbol{\varrho}_i(\gamma)\}_{i=0}^n$ osztópontok meghatározhatók a következő rekurzív képletek segítségével:

$$\boldsymbol{\lambda}_i(\gamma) = \frac{1}{b_{n,i}^{(i)}(\alpha;\alpha,\gamma)} \left(\mathbf{c}_n^{(i)}(\alpha) - \sum_{j=0}^{i-1} \boldsymbol{\lambda}_j(\gamma) \cdot b_{n,j}^{(i)}(\alpha;\alpha,\gamma) \right), \quad i = 1, ..., \left\lfloor \tfrac{n-1}{2} \right\rfloor, \tag{1.38}$$

$$\boldsymbol{\lambda}_{n-i}(\gamma) = \frac{1}{b_{n,n-i}^{(i)}(\gamma;\alpha,\gamma)} \left(\mathbf{c}_n^{(i)}(\gamma) - \sum_{j=0}^{i-1} \boldsymbol{\lambda}_{n-j}(\gamma) \cdot b_{n,n-j}^{(i)}(\gamma;\alpha,\gamma) \right), \quad i = 1, ..., \left\lfloor \tfrac{n}{2} \right\rfloor, \tag{1.39}$$

$$\boldsymbol{\varrho}_i(\gamma) = \frac{1}{b_{n,i}^{(i)}(\gamma;\gamma,\beta)} \left(\mathbf{c}_n^{(i)}(\gamma) - \sum_{j=0}^{i-1} \boldsymbol{\varrho}_j(\gamma) \cdot b_{n,j}^{(i)}(\gamma;\gamma,\beta) \right), \quad i = 1, ..., \left\lfloor \tfrac{n}{2} \right\rfloor, \tag{1.40}$$

$$\boldsymbol{\varrho}_{n-i}(\gamma) = \frac{1}{b_{n,n-i}^{(i)}(\beta;\gamma,\beta)} \left(\mathbf{c}_n^{(i)}(\beta) - \sum_{j=0}^{i-1} \boldsymbol{\varrho}_{n-j}(\gamma) \cdot b_{n,n-j}^{(i)}(\beta;\gamma,\beta) \right), \quad i = 1, ..., \left\lfloor \tfrac{n-1}{2} \right\rfloor. \tag{1.41}$$

Bizonyítás. Az ismeretlen $\{\boldsymbol{\lambda}_i(\gamma)\}_{i=0}^n$ és $\{\boldsymbol{\varrho}_i(\gamma)\}_{i=0}^n$ osztópontok kiszámítása visszavezethető az (1.33)-as és az (1.34)-es azonosságok, illetve az (1.8) – (1.10)-es Hermite-féle végpontbeli feltételek együttes alkalmazására, melyeket a $\mathcal{B}_n^{\alpha,\beta}$, $\mathcal{B}_n^{\alpha,\gamma}$, illetve $\mathcal{B}_n^{\gamma,\beta}$ bázisok egyaránt teljesítenek.

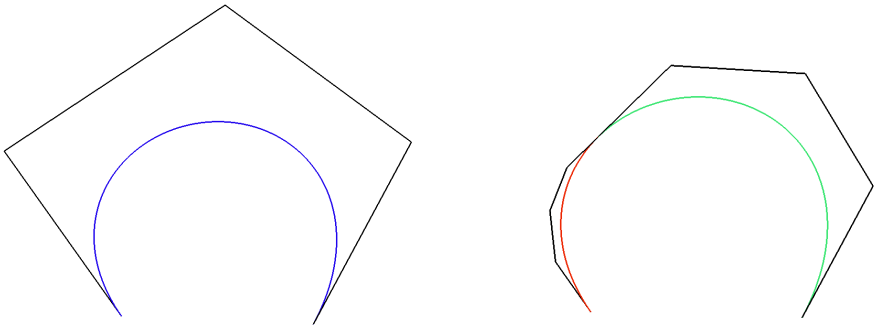
Például az (1.35)-ös kezdeti feltétel az (1.19)-es és az (1.31)-es B-görbék végpontbeli interpolációs tulajdonságából adódik, mivel

$$\mathbf{c}_n(\alpha) = \sum_{i=0}^{n} \mathbf{p}_i \cdot b_{n,i}(\alpha;\alpha,\beta) = \mathbf{p}_0 = \boldsymbol{\lambda}_0(\gamma) = \sum_{i=0}^{n} \boldsymbol{\lambda}_i(\gamma) \cdot b_{n,i}(\alpha;\alpha,\gamma) = \mathbf{l}_n(\alpha).$$

Ugyanakkor minden $i = 1, 2, ..., \left\lfloor \frac{n-1}{2} \right\rfloor$ deriválási rendre fennáll a $b_{n,i}^{(i)}(\alpha;\alpha,\gamma) > 0$ egyenlőtlenség, illetve hogy

$$\begin{aligned}
\mathbf{c}_n^{(i)}(\alpha) &= \mathbf{l}_n^{(i)}(\alpha) \\
&= \sum_{j=0}^{n} \boldsymbol{\lambda}_j(\gamma) \cdot b_{n,j}^{(i)}(\alpha;\alpha,\gamma) \\
&= \sum_{j=0}^{i} \boldsymbol{\lambda}_j(\gamma) \cdot b_{n,j}^{(i)}(\alpha;\alpha,\gamma) \\
&= \sum_{j=0}^{i-1} \boldsymbol{\lambda}_j(\gamma) \cdot b_{n,j}^{(i)}(\alpha;\alpha,\gamma) + \boldsymbol{\lambda}_i(\gamma) \cdot b_{n,i}^{(i)}(\alpha;\alpha,\gamma),
\end{aligned}$$

ahonnan következik az (1.38)-as képlet a $\boldsymbol{\lambda}_i(\gamma)$ osztópontra. A többi rekurzív összefüggés hasonlóan igazolható. Vegyük észre, hogy ezek mindegyike igaz lenne bármely $i \in 1, ..., n$ deriválási rendre. A tétel kijelentésében szereplő leszűkítés célja csak az, hogy a kiértékelendő legmagasabb deriválási rend a lehető legkisebb maradjon (a hatékonyság és stabilitás érdekében). ■



(a) (b)

1.6. ábra. Az $\mathbb{AT}_4^{0,\frac{\pi}{2}} = \langle \{1, u, u^2, \sin(u), \cos(u)\} \rangle$ algebrai-trigonometrikus KC-tér fölött értelmezett negyedrendű B-görbe felosztása a $\gamma = \frac{\pi}{6}$ paraméterértéknél. Az (a) ábrán az eredeti görbe és ennek kontrollpoligonja látható, a (b) ábrán a felosztás utáni bal és jobb oldali ív, különböző színekkel.

1.6. Bázistranszformáció KC-terekben

Szeretnénk felépíteni azt a bázistranszformációt, amely az $\mathbb{S}_n^{\alpha,\beta}$ KC-tér $\mathcal{B}_n^{\alpha,\beta}$ normalizált B-bázisát annak $\mathcal{F}_n^{\alpha,\beta}$ hagyományos bázisára képezi le (ahol $\beta - \alpha \in (0, l'_n)$):

$$\left[\varphi_{n,i}(u)\right]_{i=0}^{n} = \left[t_{i,j}^{n}\right]_{i=0,\ j=0}^{n,\ n} \cdot \left[b_{n,i}(u)\right]_{i=0}^{n}, \quad \forall u \in [\alpha, \beta]. \tag{1.42}$$

A következő tétel egy hatékony módszert ad az ismeretlen $[t_{i,j}^n]_{i=0,\ j=0}^{n,\ n}$ értékek kiszámítására.

1.9. Tétel (Hatékony bázistranszformáció, [Róth, 2017, 2.11. tétel])**.** *Az előbbi jelöléseket használva az (1.42) bázistranszformáció $[t_{i,j}^n]_{i=0,\ j=0}^{n,\ n}$ mátrixának elemei meghatározhatók az alábbi rekurzív összefüggésekkel:*

$$t_{i,j}^{n} = \frac{1}{b_{n,j}^{(j)}(\alpha)} \left[\varphi_{n,i}^{(j)}(\alpha) - \sum_{k=0}^{j-1} t_{i,k}^{n} b_{n,k}^{(j)}(\alpha)\right], \quad j = 1, 2, ..., \left\lfloor \frac{n}{2} \right\rfloor, \tag{1.43}$$

illetve

$$t_{i,n-j}^{n} = \frac{1}{b_{n,n-j}^{(j)}(\beta)} \left[\varphi_{n,i}^{(j)} - \sum_{k=0}^{j-1} t_{i,n-k}^{n} b_{n,n-k}^{(j)}(\beta)\right], \quad j = 1, 2, ..., \left\lfloor \frac{n-1}{2} \right\rfloor, \tag{1.44}$$

ahol a kezdőelemek $\left\{t_{0,j}^{n} = 1\right\}_{j=0}^{n}$, $\left\{t_{i,0}^{n} = \varphi_{n,i}(\alpha)\right\}_{i=1}^{n}$ *és* $\left\{t_{i,n}^{n} = \varphi_{n,i}(\beta)\right\}_{i=1}^{n}$.

Bizonyítás. Az (1.43)-as és az (1.44)-es összefüggések helyessége belátható, ha a

$$\varphi_{n,i}(u) = \sum_{k=0}^{n} t_{i,k}^{n} b_{n,k}(u), \;\; \forall u \in [\alpha, \beta], \;\; i = 1, ..., n$$

függvényegyenlőségeket rendre $j = 1, ..., \left\lfloor \frac{n}{2} \right\rfloor$-edrendben deriváljuk, majd az $u = \alpha$, illetve $u = \beta$ paraméterértékeknél alkalmazzuk az (1.8) – (1.10)-es végpontbeli feltételek valamelyikét. Például $u = \alpha$ esetében írhatjuk, hogy

$$\varphi_{n,i}^{(j)}(\alpha) = \sum_{k=0}^{n} t_{i,k}^{n} b_{n,k}^{(j)}(\alpha) \overset{(1.8)}{=\!=} \sum_{k=0}^{j} t_{i,k}^{n} b_{n,k}^{(j)}(\alpha) = \sum_{k=0}^{j-1} t_{i,k}^{n} b_{n,k}^{(j)}(\alpha) + t_{i,j}^{n} b_{n,j}^{(j)}(\alpha),$$

ahol $b_{n,j}^{(j)}(\alpha) > 0$. Tehát a $t_{i,j}^{n}$ elemek megkaphatók a megfelelő kivonás és osztás elvégzésével. ■

Ha a bázistranszformációs mátrix rendelkezésünkre áll, a hagyományos bázis függvényében paraméteres alakban megadott görbék és felületek egzakt leírására is lehetőség nyílik.

1.10. Tétel (Görbék egzakt leírása, [Róth, 2015])**.** *Az (1.19) típusú B-görbéket felhasználva, a*

$$\mathbf{c}(u) = \sum_{i=0}^{n} \boldsymbol{\lambda}_i \varphi_{n,i}(u), \;\; u \in [\alpha, \beta], \;\; \boldsymbol{\lambda}_i \in \mathbb{R}^{\delta}, \;\; \delta \geq 2 \tag{1.45}$$

hagyományos paraméteres alakban megadott görbére teljesül, hogy

$$\mathbf{c}(u) \equiv \mathbf{c}_n(u) = \sum_{j=0}^{n} \mathbf{p}_j b_{n,j}(u), \;\; \forall u \in [\alpha, \beta],$$

ahol

$$\begin{bmatrix} \mathbf{p}_0 & \mathbf{p}_1 & \cdots & \mathbf{p}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{\lambda}_0 & \boldsymbol{\lambda}_1 & \cdots & \boldsymbol{\lambda}_n \end{bmatrix} \cdot \left[t_{i,j}^{n} \right]_{i=0,\, j=0}^{n,\, n}. \tag{1.46}$$

1.11. Tétel (Felületek egzakt leírása – az előző tétel kiterjesztése [Róth, 2017])**.** *Legyen*

$$\mathcal{F}_{n_r}^{\alpha_r, \beta_r} = \left\{ \varphi_{n_r, i_r}(u_r) : u_r \in [\alpha_r, \beta_r] \right\}_{i_r=0}^{n_r}, \;\; \varphi_{n_r,0} \equiv 1$$

a hagyományos, illetve

$$\mathcal{B}_{n_r}^{\alpha_r, \beta_r} = \left\{ b_{n_r, j_r}(u_r) : u_r \in [\alpha_r, \beta_r] \right\}_{j_r=0}^{n_r}$$

az egyedi normalizált B-bázisa az $\mathbb{S}_{n_r}^{\alpha_r, \beta_r}$ KC-tereknek ($r = 0, 1$), valamint jelölje $\left[t_{i_r, j_r}^{n_r} \right]_{i_r=0,\, j_r=0}^{n_r,\, n_r}$ azt a reguláris négyzetes mátrixot, mely a $\mathcal{B}_{n_r}^{\alpha_r, \beta_r}$-t $\mathcal{F}_{n_r}^{\alpha_r, \beta_r}$-be alakítja. Tekintsük továbbá az

$$\mathbf{s}(u_0, u_1) = \begin{bmatrix} s^0(u_0, u_1) & s^1(u_0, u_1) & s^2(u_0, u_1) \end{bmatrix}^T \in \mathbb{R}^3, \;\; (u_0, u_1) \in [\alpha_0, \beta_0] \times [\alpha_1, \beta_1] \tag{1.47}$$

felületet, ahol

$$s^l(u_0,u_1)=\sum_{\zeta=0}^{\sigma_l-1}\prod_{r=0}^{1}\left(\sum_{i_r=0}^{n_r}\lambda_{n_r,i_r}^{l,\zeta}\varphi_{n_r,i_r}(u_r)\right),\ \ \sigma_l\geq 1,\ \ l=0,1,2.$$

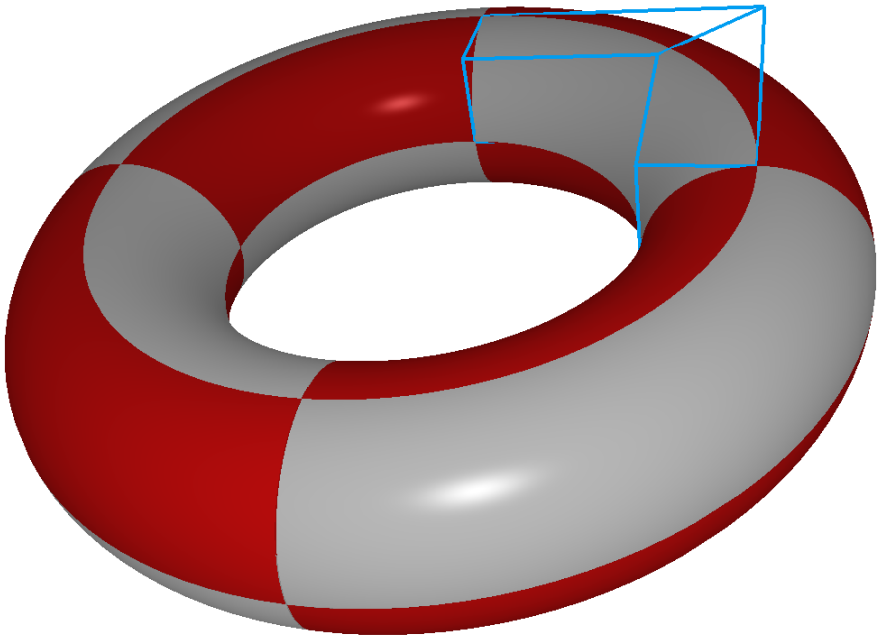
Ekkor (1.20) típusú B-felületeket felhasználva az (1.47)-es felületre teljesül, hogy

$$\mathbf{s}(u_0,u_1)\equiv\mathbf{s}_{n_0,n_1}(u_0,u_1)=\sum_{j_0=0}^{n_0}\sum_{j_1=0}^{n_1}\mathbf{p}_{j_0,j_1}b_{n_0,j_0}(u_0)\,b_{n_1,j_1}(u_1),\ \ \forall(u_0,u_1)\in[\alpha_0,\beta_0]\times[\alpha_1,\beta_1],$$

ahol a $\mathbf{p}_{j_0,j_1}=\left[p_{j_0,j_1}^l\right]_{l=0}^2\in\mathbb{R}^3$ *pontokat meghatározó koordináták*

$$p_{j_0,j_1}^l=\sum_{\zeta=0}^{\sigma_l-1}\prod_{r=0}^{1}\left(\sum_{i_r=0}^{n_r}\lambda_{n_r,i_r}^{l,\zeta}t_{i_r,j_r}^{n_r}\right),\ \ l=0,1,2. \tag{1.48}$$

Az (1.19) és (1.20) típusú B-görbéket és B-felületeket felhasználva az (1.46) – (1.48)-as képletek felhasználásával a bemutatott bázistranszformáció felhasználható olyan hagyományos paraméteres alakban megadott görbék és felületek egy nagy családjának egzakt leírására, melyek fontosak lehetnek az alkalmazott matematika számos területén. A bemutatott módszereken keresztül az alkalmazással kezelhető KC-terek magukba foglalnak olyan függvényeket, melyek gyakran használt geometriai objektumok leírását is lehetővé teszik, mint például ellipszisek, epi- és hipocikloidok, epi- és hipotrochoidok, Lissajous-görbék, tórusz csomók, rózsa-görbék, hiperbolák, ellipszoidok, archimédeszi és logaritmikus spirálok stb. Az 1.7. ábra egy tórusz B-felületi foltokból felépített egzakt előállítását tartalmazza.



1.7. ábra. Egy tórusz egzakt leírása és megjelenítése B-felületdarabokkal (az egyik felületdarab kontrollhálója kékkel kiemelve). A megjelenítés és a kontrollpontok helyének kiszámítása a bemutatott alkalmazás segítségével történt.

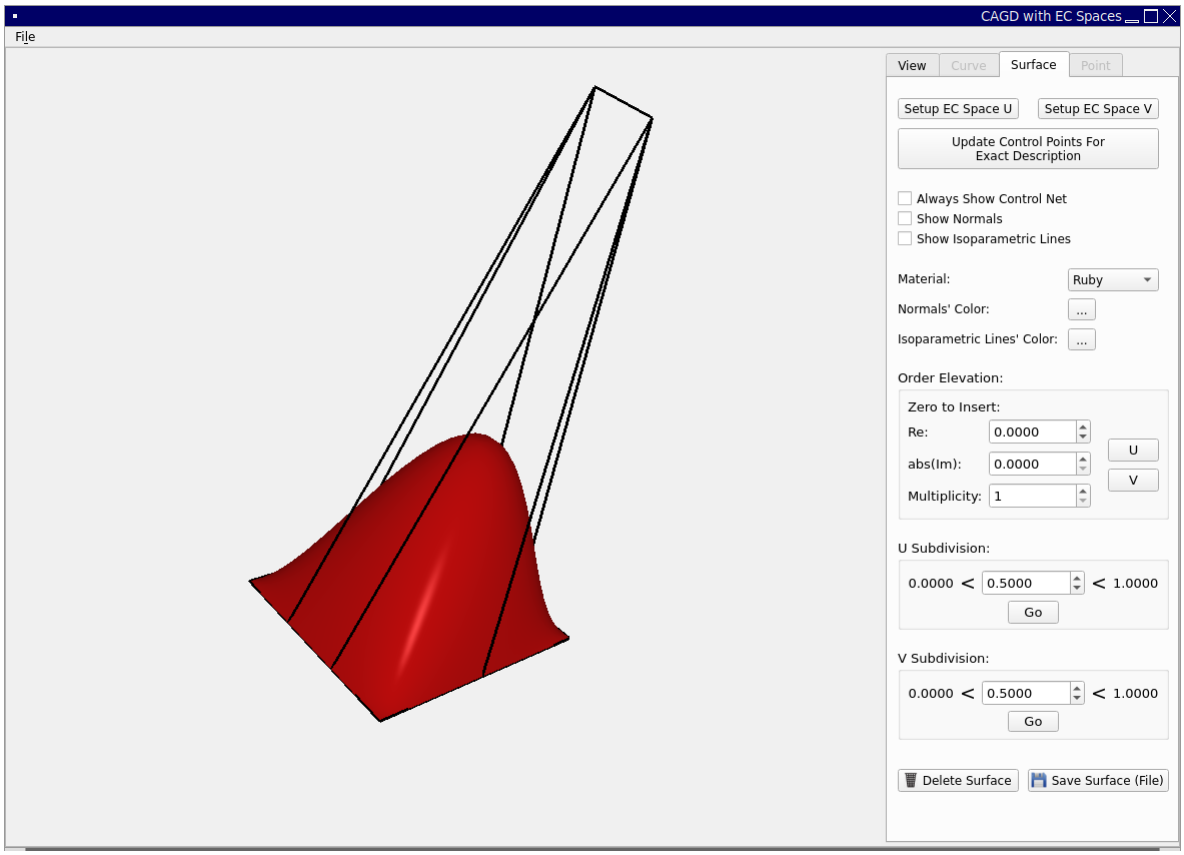
2. fejezet

Implementációs részletek

Összefoglaló: *Bemutatjuk az alkalmazás felhasználói felületét, ismertetjük a felhasznált keretrendszert és könyvtárakat, illetve bemutatjuk az 1. fejezetben tárgyalt módszerek implementációjának szerkezetét.*

2.1. Az felhasználói felület felépítése

Az alkalmazás indításakor megjelenő fő képernyőn egy üres színtér és egy jobboldalon elhelyezkedő eszköztár fogadja a felhasználót. Az eszköztáron a megjelenítési paraméterek beállítása mellett lehetőség nyílik új görbék vagy felületek hozzáadására, mentésére, törlésére, illetve már meglévők (vagy egy teljes gyűjtemény) betöltésére.



2.1. ábra. Fő képernyő, rajta egy felületi folt és a kontrollhálója

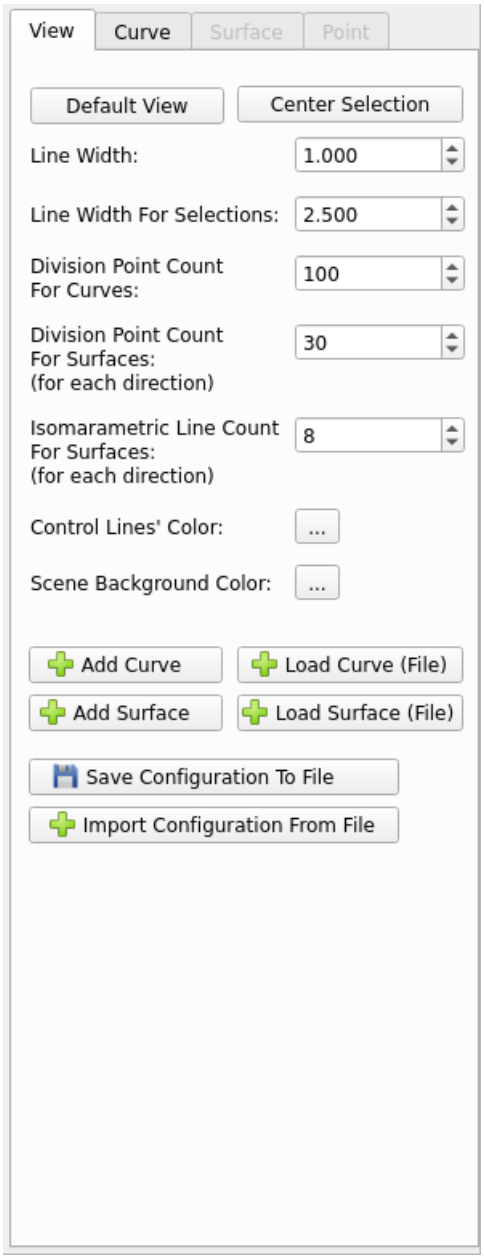
Az eszköztár négy füllel rendelkezik, ezek közül mindig azok aktívak, amelyeket a színtéren kijelölt elem meghatároz. Ha nincs semmi kijelölve, akkor csak a *Nézet (View)* fül aktív, ha egy felületdarab van kijelölve, akkor a *Felület (Surface)* is aktív lesz (ld. 2.1. ábra), ha egy görbét jelölünk ki, akkor a *Felület* helyett a *Görbe (Curve)* fül lesz aktív. Ha a görbének (felületnek) egy kontrollpontja a kijelölt elem, akkor pedig a *Pont (Point)* fül is aktív lesz az előbb említettek mellett. A következőkben bemutatjuk az eszköztár nyújtotta beállítási lehetőségeket.

2.1.1. A Nézet (View) fül

A 2.2. ábrán látható *Nézet* fül a globális beállításokat tartalmazza, ezért nem válik inaktívvá még akkor sem, ha semmi sincs kijelölve.

A tetején található két gomb elősegíti a kameranézet eredetire való visszaállítását, illetve olyan eltolását, melynek következtében az aktuálisan kijelölt elem középre kerül. Ezek alatt beállíthatjuk a vonalak általános vastagságát, illetve a kijelölt elemhez rendelt vonalvastagságot, melyet célszerű az előzőnél valamivel nagyobb beállításon hagyni, hogy a kijelöltség érzetét keltse. Beállíthatjuk még az osztópont-számokat egy-egy görbe és felületi folt megjelenítéséhez (ti. két osztópontot egyenes szakasz köt össze, a görbe vagy felület csak az egyes osztópontokban van kiértékelve). Beállíthatjuk a kontrollpoligonok és -hálók színét, illetve a teljes megjelenítési vászon háttérszínét is.

A fül alsó részén található gombok új elemek létrehozását, ezek fájlból való betöltését teszik lehetővé illetve lehetőségünk van a teljes konfiguráció lementésére vagy fájlból való betöltésére is. Az alkalmazás minden ki- és bemeneti állománya szöveges, az általa reprezentált grafikus vagy matematikai elem konfigurációjához szükséges együtthatókat és koordinátákat tartalmazza a megfelelő sorrendben, fehér karakterekkel elválasztva (a felhasználónak nem szükséges ismerni az egyes fájlok szerkezetét, viszont elnevezési konvencióval – például kiterjesztés megadásával – jelölheti, hogy melyik kimentett fájl mit tartalmaz).



2.2. ábra. Nézet (View) fül

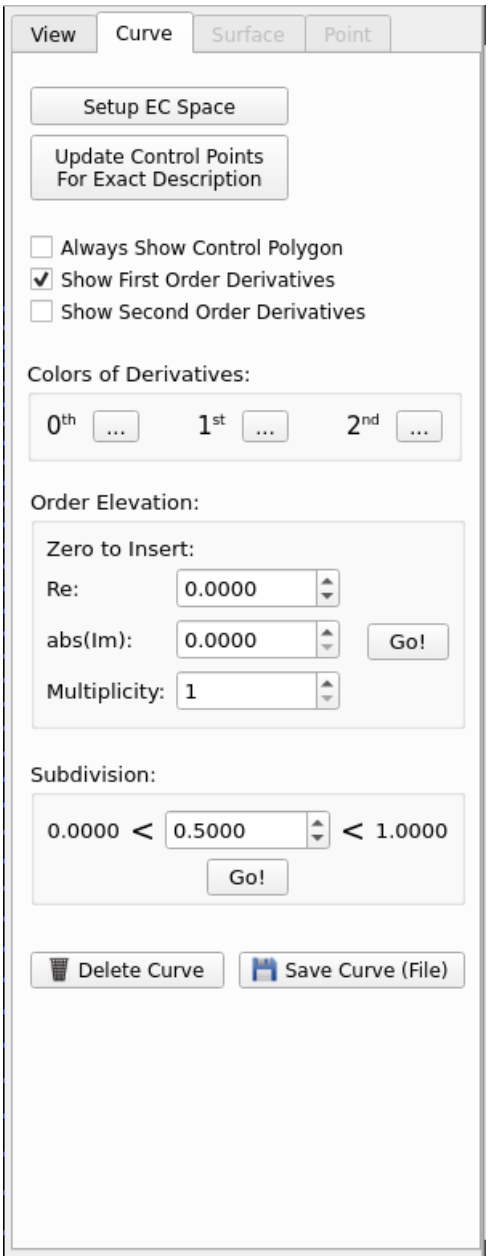
2.1.2. A Görbe (Curve) fül

A *Görbe* fül a 2.3. ábrán látható. Olyan beállításokat tartalmaz, amelyek az aktuálisan kijelölt görbére vonatkoznak (és a színtér többi elemét nem befolyásolják). Ha az aktuális elem nem egy görbe (illetve nem egy görbének kontrollpontja), akkor ez a fül inaktív, ami egyben az is jelenti, hogy ha előzőleg ki volt jelölve, akkor az eszköztár visszaugrik a *Nézet* fülre.

Az első gombra kattintva azon KC-teret tudjuk beállítani (ld. 2.1.5. pont), amely fölött a görbe értelmezett, a második gomb pedig az egzakt leíráshoz szükséges paraméteres alak beállítására szolgál (ld. 2.1.6. pont).

Jelölőnégyzetek segítségével beállíthatjuk, hogy mindig látható legyen-e a kontrollpoligon (nem csak amikor a görbe ki van jelölve), illetve hogy szeretnénk-e megjeleníteni a görbe egyes osztópontjaihoz tartozó első- és másodrendű deriváltakat. A görbének és deriváltjainak színét is külön-külön beállíthatjuk az ezek alatt elhelyezkedő három nyomógomb segítségével.

Ezeket követi egy elemcsoport, mely a rendszámnövelés megvalósítását teszi lehetővé. Ennek érdekében a KC-teret definiáló karakterisztikus polinomot úgy kell megváltoztatni, hogy a gyökeinek száma legalább egyel nőjön. Ezt teszi lehetővé a három bemeneti mező. Ha olyan gyököt állítunk be, amely már gyöke volt a karakterisztikus polinomnak, akkor annak multiplicitása a megfelelő értékkel nő. Ha a beállított gyök nem valós, akkor a konjugáltja is gyök lesz (ugyanolyan multiplicitással). A *Mehet! (Go!)* gomb megnyomásakor végrehajtódik a rendszámnövelés (a kontrollpontok száma a megadott multiplicitással – illetve nem valós gyök esetén annak kétszeresével – nő).



2.3. ábra. Görbe (Curve) fül

A következő elemcsoport a felosztást teszi lehetővé. Beállíthatjuk, hogy az aktuális értelmezési tartományt mely paraméterértéknél szeretnénk felosztani.

Végül egy-egy gomb segítségével ki lehet törölni a kijelölt görbét a színtérről, illetve le lehet azt menteni egy fájlba. A fájlból való betöltésre a *Nézet* fül ad csak lehetőséget, hiszen ez a művelet nem kapcsolódik az aktuálisan kijelölt elemhez.

2.1.3. A Felület (Surface) fül

Mint az a 2.4. ábrán látható, a *Felület* fül szintén a KC-terek illetve az egzakt leírás beállítására szolgáló gombokkal kezdődik. Itt akár különböző KC-tereket is beállíthatunk az egyes (u és v) paraméterirányoknak.

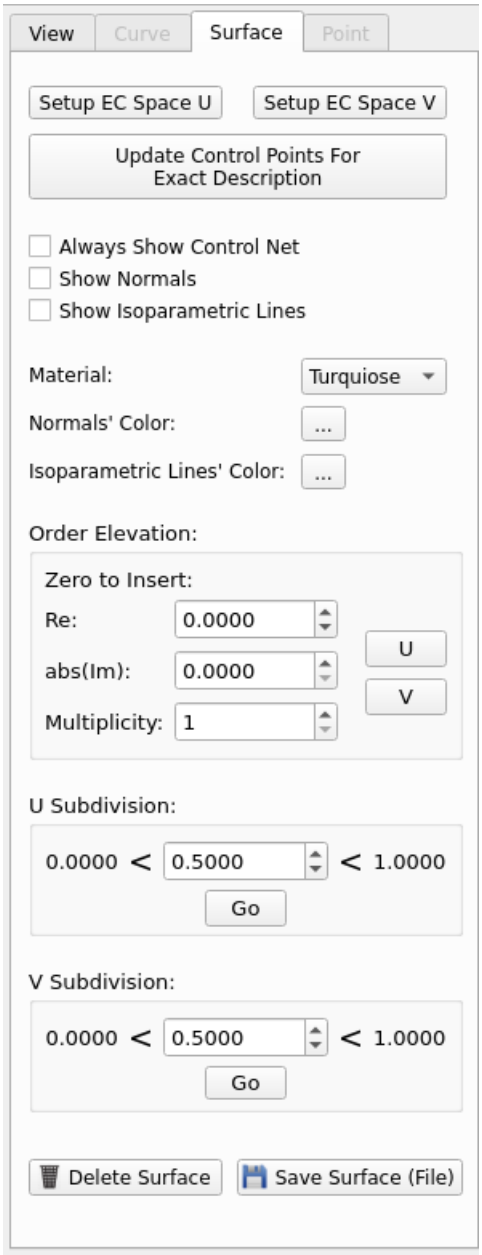
A gombokat követő három jelölőnégyzettel beállíthatjuk a kontrollháló állandó láthatóságát (hasonlóan az előbbiekben a kontrollpoligonhoz), azt, hogy szeretnénk-e látni a felületpontokhoz tartozó normálvektorokat, illetve, hogy szeretnénk-e a felületen izoparametrikus vonalakat megjeleníteni.

A következő legördülő listából néhány előre beállított anyagi jellemző közül választhatunk (melyek meghatározzák a felület színét és fényvisszaverő tulajdonságait), majd beállíthatjuk a normálvektorok és izoparametrikus vonalak színét.

A rendszámnövelésre szolgáló elemcsoport a görbékhez hasonló, viszont két gomb áll a rendelkezésünkre, mivel rendszámnövelést a felületek esetében külön az u vagy v paraméterirányban tudunk elvégezni. Mindkét esetben csak a paraméteriránynak megfelelő KC-tér kerül módosításra, valamint nő a kontrollháló sorainak vagy oszlopainak száma.

Következik két elemcsoport, az u- illetve v-irányú felosztás megvalósítására. Mindkét esetben a megfelelő KC-tér értelmezési tartományából választhatunk ki egy-egy belső pontot, melynek megfelelő (u- vagy v-irányú) paramétergörbe mentén történik majd a felület felosztása.

Végül egy-egy gomb ad lehetőséget a felületdarab törlésére illetve állományba mentésére.

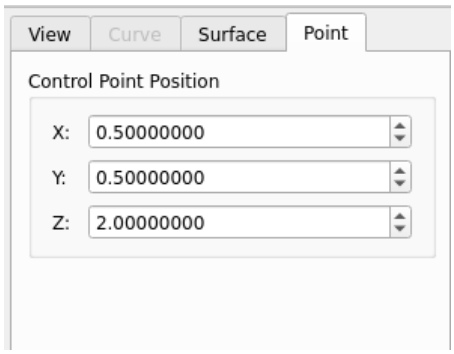


2.4. ábra. Felület (Surface) fül

2.1.4. A Pont (Point) fül

Bár a kontrollpontok egérrel mozgathatók, előfordulhat, hogy a felhasználó konkrét koordinátaértékre szeretné helyezni ezeket. Egy kijelölt kontrollpont koordinátáinak megtekintésére és megváltoztatására ad lehetőséget a 2.5. ábrán látható *Pont* fül.

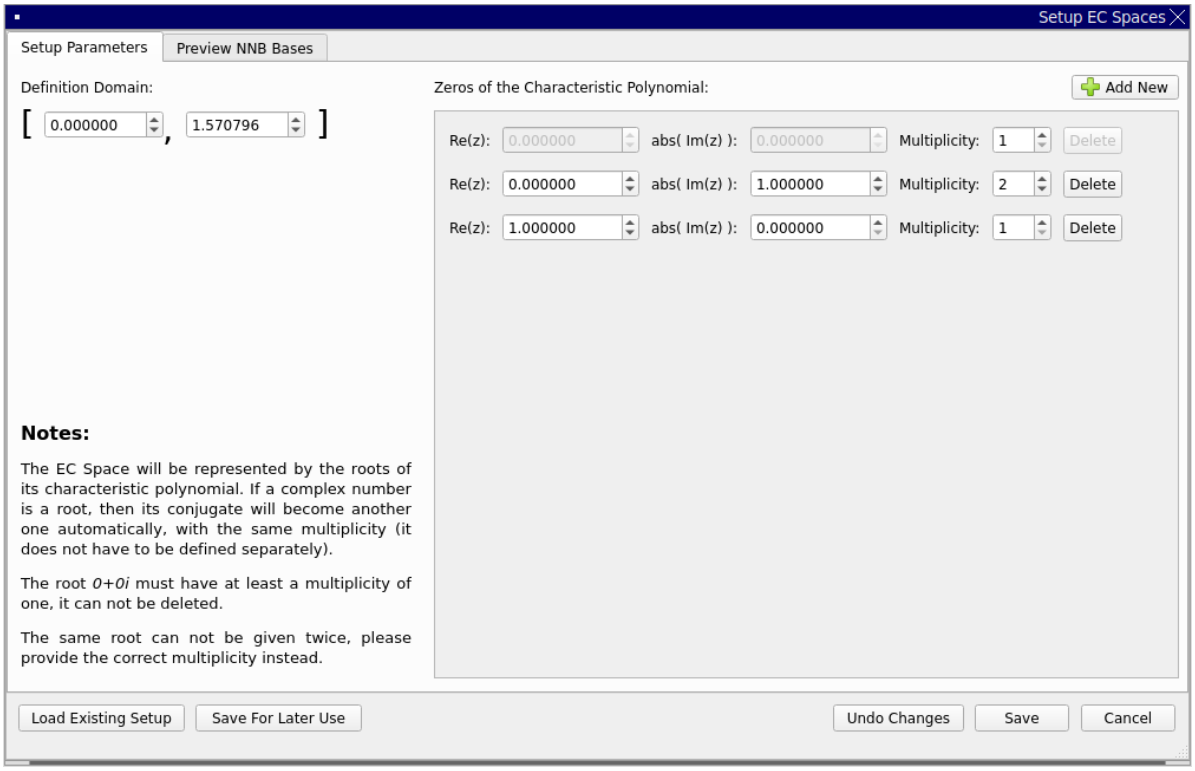
Mint minden eddigi beállítás esetében, a színtér élőben módosul jelen esetben is, ha az X, Y, Z koordinátákhoz társított értékeket megváltoztatjuk.



2.5. ábra. Pont (Point) fül

2.1.5. KC-terek konfigurálására szolgáló dialógusablak

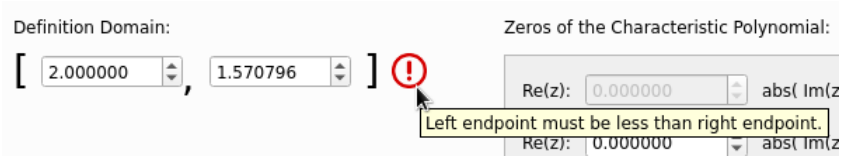
Amint azt az elméleti részben említettük, egy KC-teret az (1.12) alakú karakterisztikus polinom gyökeinek megadásával határozunk meg az alkalmazásban. Erre nyújt lehetőséget a 2.6. ábrán látható dialógusablak.



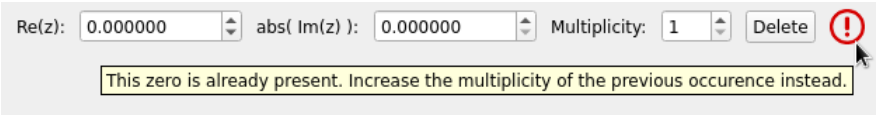
2.6. ábra. KC-tér definiálására szolgáló dialógusablak

Beállíthatjuk a tér függvényeinek közös értelmezési tartományát, majd hozzáadhatunk, kitörölhetünk és megváltoztathatunk gyököket, illetve ezek multiplicitásait. A komplex gyökpárok közül csak az egyiket kell megadni (vagyis az imaginárius résznek csak az abszolút értékét), hiszen a másik ugyanolyan multiplicitással feltétlenül gyök kell legyen, mert a karakterisztikus polinom valós együtthatós.

Ha deformált intervallumot adunk meg, hibaüzenetet kapunk (ld. 2.7. ábra). Hasonlóan abban az esetben is, ha egy gyököt kétszer adunk meg (ld. 2.8. ábra), ugyanis ilyenkor az előző gyök multiplicitását kellene növelni.



2.7. ábra. Hibás intervallum esetén mutatott hibaüzenet



2.8. ábra. Ismétlődő gyök esetén mutatott hibaüzenet

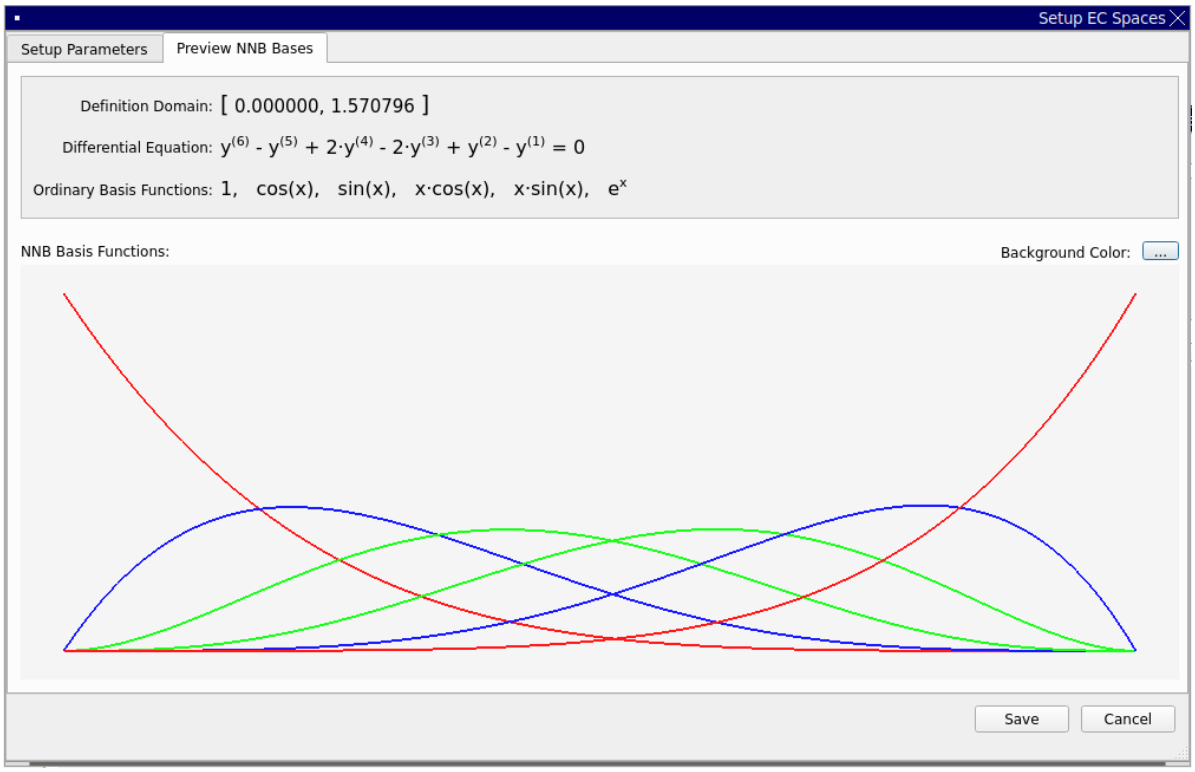
Az alul található gombok segítségével betölthetünk egy KC-tér konfigurációt egy memóriában található pufferből vagy fájlból (*Load Existing Setup*), illetve lementhetjük azt a memóriában tárolt pufferbe vagy fájlba (*Save For Later Use*). Lehetőség van az ablak megnyitásakor látott eredeti állapot visszaállítására (*Undo Changes*), a változtatások mentésére (*Save*) és az ablak bezárására mentés nélkül (*Cancel*).

A dialógusablak rendelkezik egy, a 2.9. ábrán látható másik füllel is, mely az előbbiekben definiált KC-tér keverőfüggvényeinek előnézetét mutatja. Ez azért is fontos, mert a keverőfüggvények alakját látva meggyőződhetünk arról, hogy nem ütköztünk stabilitási problémába, illetve hogy az általunk megadott intervallumon tényleg KC-teret generálnak-e a gyökök által meghatározott bázisfüggvények.

Ez utóbbi ugyanis nem akármilyen intervallumra igaz és az ennek előzetes meghatározására ismert módszer (ld. (1.18)-ban a kritikus hossz vizsgálatát) néha túl nagy elővigyázatosságot eredményez (az-az nagyobb intervallumot is gond nélkül használhatna a felhasználó). Ennek elkerülése érdekében az intervallum megfelelőségének (grafikus) ellenőrzését itt a felhasználóra bízzuk.

A bázisfüggvények alakján kívül látható még az aktuális értelmezési tartomány, a gyökök által meghatározott differenciálegyenlet, illetve a hagyományos bázist felépítő függvények zárt alakja. A megjelenítési vászon háttérszíne szintén állítható (alapértelmezetten fekete).

A felhasználónak javasolt munkamenet tehát az, hogy miután a tér paramétereinek beállítása megtörtént, mentés előtt térjen át erre a fülre az új keverőfüggvények ellenőrzése céljából. Így elkerülhető, hogy mentés után a görbe (vagy felület) alakja az instabilitás miatt (vagy a nem megfelelő értelmezési tartomány miatt) elromoljon.



2.9. ábra. KC-tér keverőfüggvényeinek nézete

2.1.6. Egzakt leírás beállítására szolgáló dialógusablak

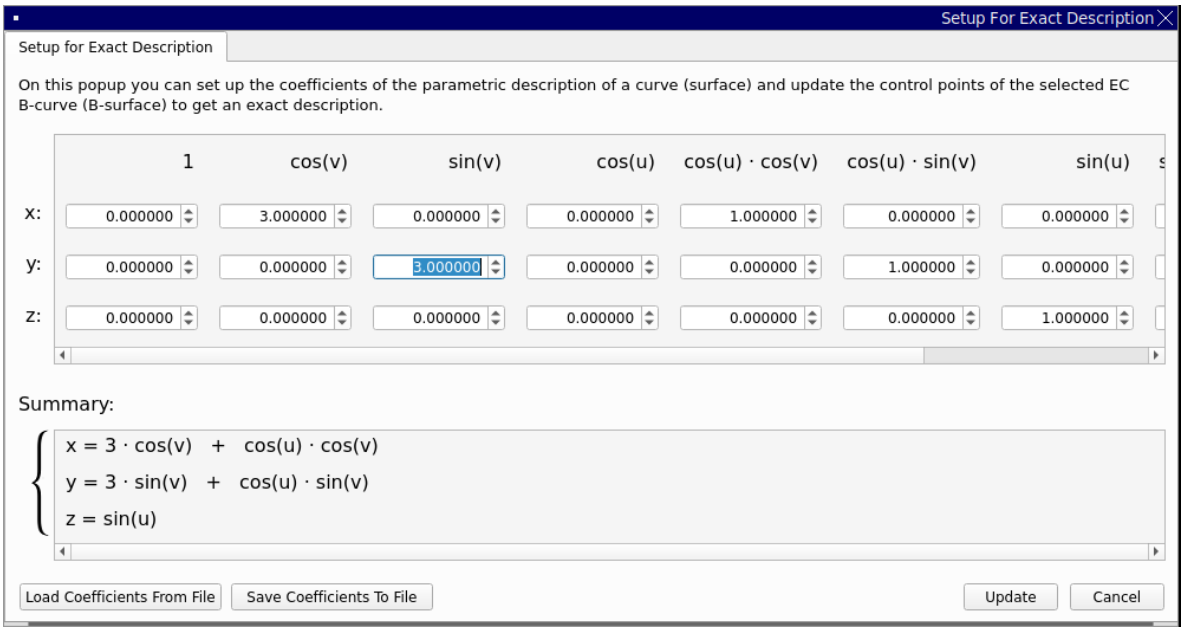
Az egzakt leírás megvalósításához lehetőséget kell adjunk a felhasználónak arra, hogy a hagyományos bázisoktól függő paraméteres alakot megadhassa. Erre szolgál a 2.10. ábrán látható dialógusablak.

Görbék esetén a hagyományos bázisfüggvények jelennek meg az oszlopokban, felületek esetén pedig az u- és v-irányú terek hagyományos bázisfüggvényeiből álló szorzatpárok. A felhasználó nem nulla együtthatók beállítása által tudja bevinni a rendszerbe a leírni kívánt görbe vagy felületdarab explicit paraméteres alakját.

A 2.10. ábrán például az 1.7. ábrán szereplő tórusz paraméteres alakjának megfelelően állítottuk be az együtthatókat. Mivel az oszlopok száma a terek dimenziószámának szorzata, ezeket nem mindig tudjuk egyszerre megjeleníteni, ezért egy görgetősávot használunk.

Az *Frissítés (Update)* gomb lenyomásakor a kijelölt görbe vagy felületi folt kontrollpontjainak helye az (1.42)-es bázistranszformáció, illetve az (1.46)-os és (1.48)-as képletek alapján megváltozik, így a paraméteres leírásnak megfelelő alakot kapjuk (az aktuális értelmezési tartományra leszűkítve, például az 1.7. ábrán látható tóruszt több folt együtteséből tudjuk csak előállítani).

Lehetőség van még az aktuális együttható-beállítás fájlba mentésére, illetve onnan történő betöltésére, valamint az ablak bezárására anélkül, hogy a kontrollpontok helyét módosítanánk.



2.10. ábra. Egzakt leírás beállítása

2.1.7. Egérműveletek a színtéren

Az implementált egérműveletek egyrészt a kameranézet állítását, másrészt pedig objektumok kijelölését (majd ezek elmozgatását) teszik lehetővé.

Az egérkurzor típusa annak függvényében változik, hogy az alatta lévő területen éppen milyen objektum található:

- az alapértelmezett kurzor (2.11a ábra) akkor látható, amikor alatta nem található objektum, ilyenkor kattintás hatására nem történik kijelölés, viszont tudjuk a kameranézetet mozgatni (bal egérgombot letartva), a nézet Ox és Oy tengelyei körül forgatni (jobb egérgombot letartva), illetve nagyítani vagy kicsinyíteni (görgetéssel);
- a megjelölő kurzor (2.11b ábra) akkor látható, ha az egérmutató alatt egy még nem kijelölt görbe vagy felületdarab található, bal kattintással ilyenkor kijelölhetjük az adott elemet;
- a pontmegjelölő kurzor (2.11c ábra) olyankor jelenik meg, ha a kijelölt görbe vagy felületdarab kontrollpoligonjának egyik csúcspontja van a közelben, ilyenkor bal egérgombbal kijelölhetjük az adott kontrollpontot;
- a mozgató kurzor (2.11d ábra) azt jelzi, hogy az egér alatt az aktuálisan kijelölt elem található, ilyenkor a bal egérgombot letartva tudjuk mozgatni a kijelölt elemet, a jobb egérgombot letartva tudjuk forgatni (a nézet Ox és Oy tengelyei körül), illetve a **[Ctrl]** billentyűt letartva a görgővel tudjuk nagyítani vagy kicsinyíteni.

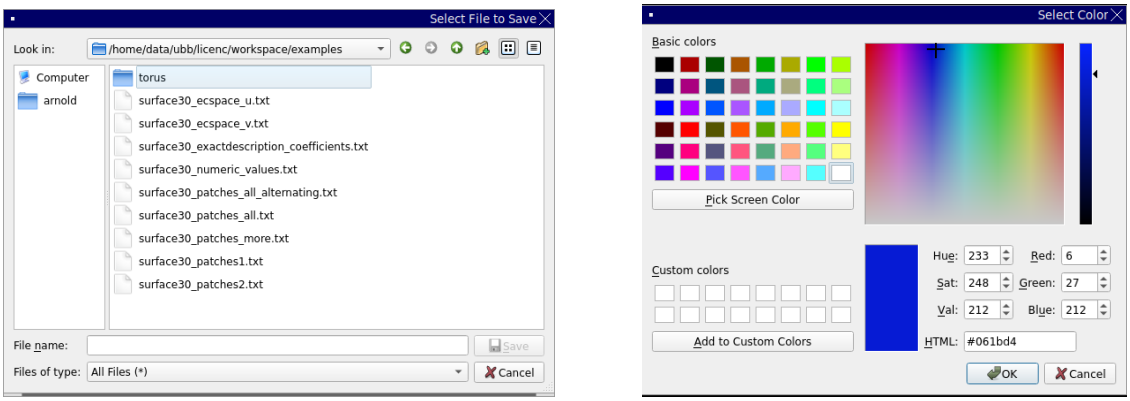
(a) Alapértelmezett kurzor (b) Megjelölő kurzor (c) Pontmegjelölő kurzor (d) Mozgató kurzor

2.11. ábra. Kurzortípusok

2.2. A forráskód szerkezete

A programkód *C++* nyelvben (ld. [Stroustrup, 2013]) íródott objektumorientált szemléletben, a felhasznált külső függőségek pedig:

– a *Qt 5* keretrendszer[1]: a grafikus felhasználói felület alapelemeit, illetve az ezekhez tartozó (signal – slot) típusú eseménykezelést biztosítja, továbbá rendelkezésünkre bocsát olyan dialógusablakokat, melyekkel a (ki- és bemeneti) fájlok, illetve a felhasználni kívánt színek kiválaszthatóak (ld. 2.12. ábra). Mindezen elemek megjelenítéséhez a keretrendszer *Widgets* alrészét használtuk. Az elkészített alkalmazást így bizonyos platformok között hordozhatóvá tehetjük (jelen esetben egy Linux disztribúción X11 alatt, illetve MS Windows 10 operációs rendszeren teszteltük az alkalmazást, csak a fordítás konfigurációja platformfüggő);



(a) Fájlválasztó ablak (*QFileDialog*) (b) Színválasztó ablak (*QColorDialog*)

2.12. ábra. Qt Widgets dialógusablakok

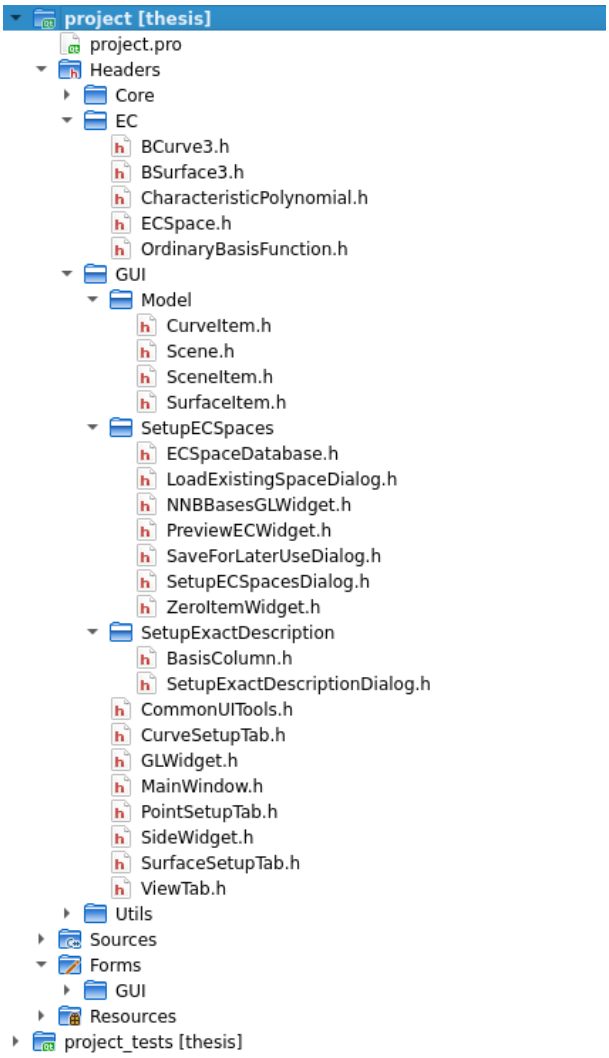
– az *OpenGL* könyvtár: az alacsonyszintű grafikus elemek (pont, szakasz, háromszög) megjelenítését, színek, anyagi jellemzők és árnyalók beállítását, illetve a színtér kialakítását (nézet beállítása,

1. https://doc.qt.io/qt-5/ (megtekintve 2018. június 20.)

megvilágítás) teszi lehetővé (részletes leírásért ld. [Kessenich et al., 2016]). Felhasználtuk továbbá a *GLU*[2] (OpenGL Utility Library) és *GLEW*[3] (OpenGL Extension Wrangler) segédkönyvtárakat.

– az *OpenMP*[4] (Open Multi-Processing) programozási interfész: az egyes számítások párhuzamosítására használtuk általában olyan esetekben, amikor ezek csak különböző memóriacímekre írnak. Ilyenkor a kölcsönös kizárás megvalósítása nem volt szükséges, és egyszerű `#pragma omp parallel for` típusú konstrukciókat tudtunk használni.

A forráskód megírásához, a felhasználói felület komponenseinek grafikus szerkesztéséhez, a fordításhoz, illetve a tesztek futtatásához a *Qt Creator* nevű integrált fejlesztői környezetet használtuk. A forráskód elrendezését a 2.13. ábra szemlélteti, mely a *Qt Creator* felhasználói felületén megjelenő hierarchia.



2.13. ábra. Forrásfájlok elrendezése

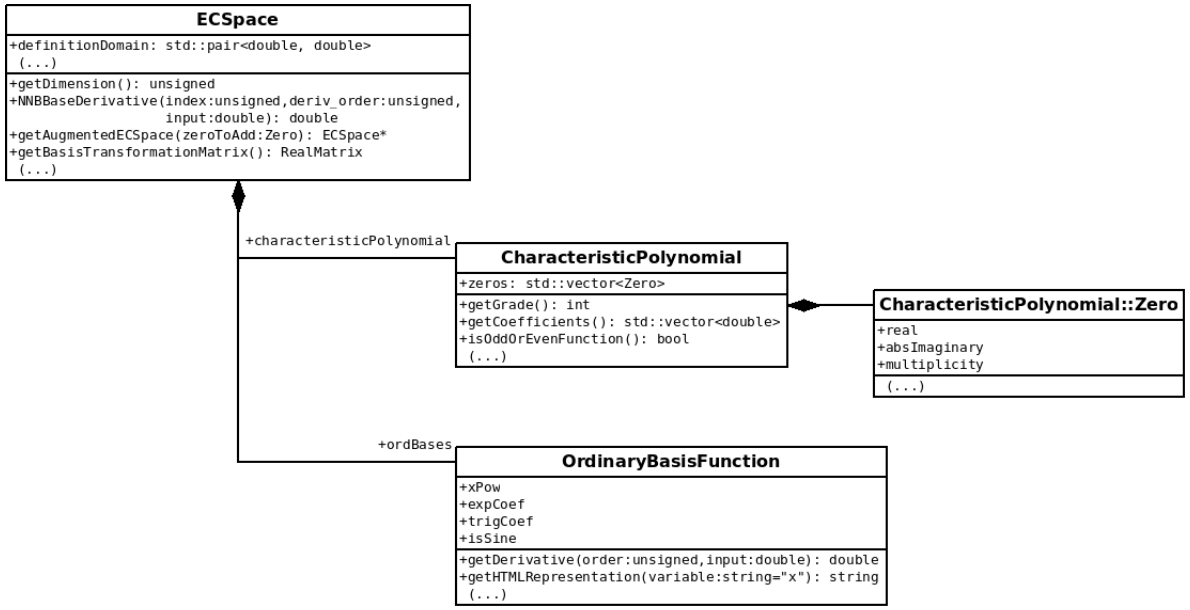
A *Core* mappa tartalmazza azon alapvető matematikai konstrukcióknak megfelelő osztályokat, melyeket az alkalmazás többi része ismételten felhasznál, ilyenek például a térbeli Descartes-féle és homogén koordináták (*DCoordinate3*, *HCoordinate3*), illetve mátrixok és mátrixfelbontások (*Matrix*, *RowMatrix*, *ColumnMatrix*, *TriangularMatrix*, *RealMatrix*, *PLUDecomposition*, *FactorizedUnpivotedLUDecomposition*). Szintén itt találhatóak meg az *OpenGL*-lel megjeleníthető görbét és felületet reprezentáló osztályok (*GenericCurve3*, *TriangulatedMesh3*), illetve azok is, amelyek ezeket kontrollpontok és keverőfüggvények (nullad- és magasabb rendű deriváltjai) alapján elő tudják állítani az (1.1)-es és (1.2)-es alaknak megfelelően (*LinearCombination3*, *TensorProductSurface3*). Végül néhány, a grafikus megjelenítést elősegítő osztály is megtalálható ebben a katalógusban: fények (*DirectionalLight*, *PointLight*, *SpotLight*), árnyaló (*ShaderProgram*), anyagi jellemzők (*Material*).

Az *EC* mappa tartalmazza az 1. fejezetben bemutatott elméleti eredmények implementációját. Az itt található osztályok kódja újrahasznosítható, nem

2. https://www.khronos.org/registry/OpenGL/specs/gl/glu1.3.pdf (megtekintve 2018. június 19.)
3. http://glew.sourceforge.net/ (megtekintve 2018. június 19.)
4. https://www.openmp.org/ (megtekintve 2018. június 20.)

függ sem a felhasználói felületünktől, sem a *Qt* keretrendszerben definiált más osztályoktól és függvényektől.

Az KC-terek kezeléséért és a bázisfüggvények kiértékeléséért felelős osztályok a 2.14. ábrán láthatók. A *CharacteristicPolynomial* osztály egy (1.11) alakú differenciálegyenlet karakterisztikus polinomját ábrázolja, az *OrdinaryBasisFunction* pedig az egyenlet megoldásterének egyik hagyományos bázisfüggvényét tudja tárolni, valamint kiértékelni ennek (nullad- és magasabb rendű) deriváltjait a Leibniz-szabály segítségével.



2.14. ábra. A KC-terek kezeléséért felelős osztályok

Az *ECSpace* osztály ábrázol egy kiterjesztett Csebisev-teret (neve az angol Extended Chebyshev megnevezésből adódik), fejállományának tartalmát láthatjuk a 2.1. kódrészletben.

2.1. Kódrészlet. Az ECSpace.h fejállomány

```
namespace cagd
{
    class ECSpace
    {
    public:
        std::pair<double, double> definitionDomain;
        CharacteristicPolynomial characteristicPolynomial;

        ECSpace();
        ECSpace(const ECSpace &ecSpace);
        const ECSpace &operator =(const ECSpace &rightSide);

        // Preprocessing is needed when the definitionDomain
        // or the characteristicPolynomial changes:
        void preprocessing();

        std::vector<OrdinaryBasisFunction> ordBases;
        unsigned getDimension() const;
```

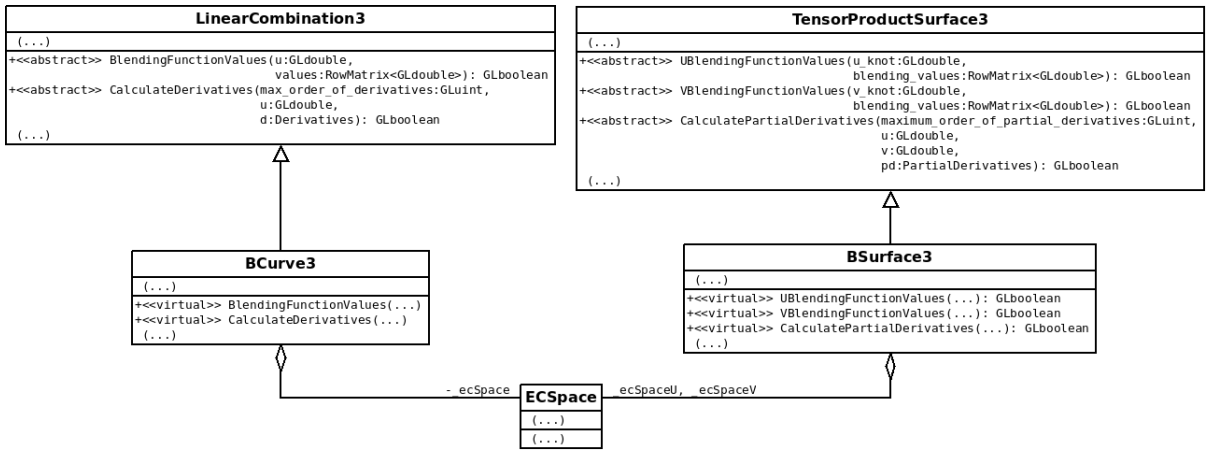
```
19
20          double NNBBaseDerivative(
21              unsigned index, unsigned deriv_order, double input) const;
22
23          ECSpace *getAugmentedECSpace(
24              CharacteristicPolynomial::Zero zeroToAdd) const;
25
26          RealMatrix getBasisTransformationMatrix() const;
27
28      private:
29          unsigned _dimension;
30          bool _isReflectionInvariant;
31          RealMatrix _rho;
32          RealMatrix _mu;
33          std::vector<double> _lambda;
34
35          void setupOrdBases();
36          void calculateRho();
37          void calculateMuAndLambda();
38      };
39
40      std::ostream &operator<<(std::ostream &stream, const ECSpace &space);
41      std::istream &operator>>(std::istream &stream, ECSpace &space);
42  }
```

Az értelmezési tartomány és a karakterisztikus polinom kívülről szabadon állíthatóak, az NB-bázisfüggvények kiértékelése előtt viszont meg kell hívni a 15. soron látható előfeldolgozó `preprocessing()` függvényt, mely előkészíti a kiértékeléshez szükséges ($\rho_{i,j}$, $\mu_{i,j}$, $\lambda_{i,j}$) együtthatókat az 1.3. szakasznak megfelelően. A 23. soron található `getAugmentedECSpace(...)` függvény a nagyobb dimenziószámú, adott új gyököt tartalmazó KC-teret téríti vissza. Bázistranszformációs mátrixot is generálhatunk az 1.6. szakaszban leírtaknak megfelelően a 26. soron megjelenő `getBasisTransformationMatrix()` függvény segítségével.

A *BCurve3* és *BSurface3* osztályok B-görbéket és B-felületeket tárolnak és a *Core* mappában található *LinearCombination3* illetve *TensorProductSurface3* osztályokat terjesztik ki (lásd 2.15. ábra), definiálva a keverőfüggvények kiértékelésére vonatkozó tiszta virtuális tagfüggvényeket. Mindkét osztály tartalmazza a felhasznált KC-teret (-tereket), illetve a rendszámnöveléshez, felosztáshoz és egzakt leírás konfigurálásához szükséges metódusokat, melyek implementációi az 1.4., 1.5. és 1.6. szakaszokban leírtakat követik. A 2.2. és 2.4. kódrészletek a *BCurve3* illetve *BSurface3* osztályokhoz tartozó fejállományok.

2.2. Kódrészlet. A BCurve3.h fejállomány

```
1  #pragma once
2
3  #include "../Core/LinearCombination3.h"
4  #include "ECSpace.h"
5
6  namespace cagd
7  {
8      class BCurve3: public LinearCombination3
9      {
10     private:
11         const ECSpace* _ecSpace;
12
13     public:
```



2.15. ábra. A *BCurve3* és *BSurface3* osztályok kapcsolatai

```
14          BCurve3(const ECSpace* ecSpace);
15
16          virtual GLboolean BlendingFunctionValues(
17              GLdouble u, RowMatrix<GLdouble>& values) const;
18          virtual GLboolean CalculateDerivatives(
19              GLuint max_order_of_derivatives,
20              GLdouble u,
21              Derivatives& d) const;
22
23          std::pair<ECSpace*, BCurve3*> performOrderElevation(
24              double reZero, double imZero, int multiplicity) const;
25          bool calculateDataForOrderElevatedCurve(
26              ECSpace *newECSpace, BCurve3 *newCurve) const;
27
28          struct SubdivisionResult {
29              ECSpace *ecSpaceLeft;
30              BCurve3 *bCurveLeft;
31              ECSpace *ecSpaceRight;
32              BCurve3 *bCurveRight;
33          };
34          SubdivisionResult performSubdivision(double gamma) const;
35          void fillSubdivisionValues(
36              SubdivisionResult &result,
37              double alpha,
38              double gamma,
39              double beta) const;
40
41          void updateControlPointsForExactDescription(
42              const std::vector<DCoordinate3> &coefficients);
43      };
44  }
```

2.3. Kódrészlet. A BSurface3.h fejállomány

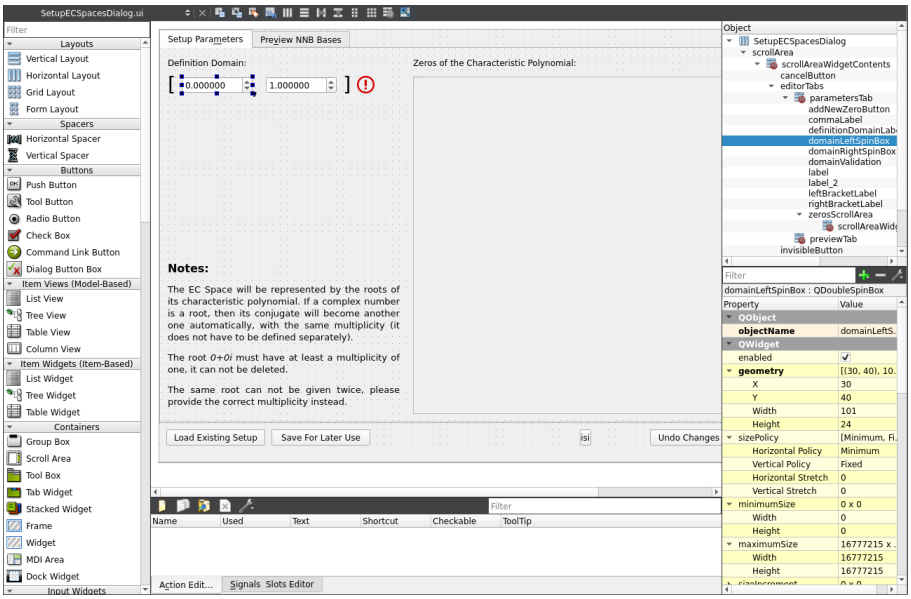
```
1  #pragma once
2  #include "ECSpace.h"
3  #include "../Core/TensorProductSurfaces3.h"
4  #include <unordered_map>
5
6  namespace cagd
7  {
```

```
8     class BSurface3: public TensorProductSurface3
9     {
10    private:
11        const ECSpace *_ecSpaceU;
12        const ECSpace *_ecSpaceV;
13
14        unsigned _cache_u_count;
15        unsigned _cache_v_count;
16        std::unordered_map<unsigned, double> _u_derivative_cache;
17        std::unordered_map<unsigned, double> _v_derivative_cache;
18
19        unsigned getKey(unsigned index, unsigned order, unsigned knot);
20        double getNNBDerivativeU(unsigned index, unsigned order, double u);
21        double getNNBDerivativeV(unsigned index, unsigned order, double v);
22
23    public:
24        BSurface3(const ECSpace *ecSpaceU, const ECSpace *ecSpaceV);
25
26        GLboolean UBlendingFunctionValues(
27            GLdouble u_knot,
28            RowMatrix<GLdouble> &blending_values);
29        GLboolean VBlendingFunctionValues(
30            GLdouble v_knot,
31            RowMatrix<GLdouble> &blending_values);
32
33        GLboolean CalculatePartialDerivatives(
34            GLuint maximum_order_of_partial_derivatives,
35            GLdouble u, GLdouble v,
36            PartialDerivatives &pd);
37
38        std::pair<ECSpace *, BSurface3 *> performUOrderElevation(
39            double reZero, double imZero, int multiplicity) const;
40        std::pair<ECSpace *, BSurface3 *> performVOrderElevation(
41            double reZero, double imZero, int multiplicity) const;
42
43        struct SubdivisionResult {
44            ECSpace *ecSpaceLeft;
45            BSurface3 *bSurfaceLeft;
46            ECSpace *ecSpaceRight;
47            BSurface3 *bSurfaceRight;
48        };
49        SubdivisionResult performUSubdivision(double gamma) const;
50        SubdivisionResult performVSubdivision(double gamma) const;
51
52        const ECSpace *getECSpaceU() const {return _ecSpaceU;}
53        const ECSpace *getECSpaceV() const {return _ecSpaceV;}
54
55        void updateControlPointsForExactDescription(
56            const std::vector<DCoordinate3> &coefficients);
57
58        virtual TriangulatedMesh3* GenerateImage(
59            GLuint u_div_point_count, GLuint v_div_point_count,
60            GLenum usage_flag = GL_STATIC_DRAW);
61    };
62 }
```

A *GUI* mappa tartalmaz minden, a felhasználói felületen megjelenő elemet, illetve a felhasználó által előidézett események kezelését. A *Model* katalógusban találhatók a színteret, illetve az ebben elhelyezhető elemeket tároló osztályok. A tárolás, megjelenítés és egérműveletek egységes kezelése érdekében a színtéren megjelenített görbéket és felületeket jelképező *CurveItem* és *SurfaceItem* osztályok egyaránt a *SceneItem* interfészt (csak tiszta virtuális tagfüggvényekkel rendelkező osztályt) implementálják. A *SetupECSpaces* katalógus tartalmazza a 2.1.5. pontban bemutatott KC-teret beállító dialógusablakot, a

SetupExactDescription pedig az egzakt leírás beállítását teszi lehetővé (ld. 2.1.6. pont). Végül az eszköztár egyes füleinek, illetve a színteret tartalmazó és OpenGL kontextussal rendelkező elemnek (*GLWidget*) is megfelel egy-egy osztály a *GUI* mappában.

A felhasználói felület egy-egy összetett eleméhez (melyek az ún. *Widget*-ek) az általunk bevezetett osztály mellett tartozik egy *Ui* névtér alatt található osztály is, melyet a grafikus szerkesztő által lementett XML fájlból a *Qt* keretrendszer fejlesztői csomagjának részét képező *qmake* program generál fordításkor. A grafikus felületszerkesztő jelentősen gyorsítja az egyes *Widget*-ek elkészítését.



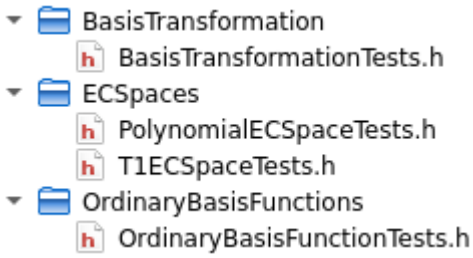
2.16. ábra. A *Qt Creator* grafikus szerkesztője

2.3. Néhány algoritmus implementációjának vázlatos tesztelése

Az ábrák alapján történő megítélés mellett szerettük volna, ha valamilyen automatikus módszerrel is meg tudunk győződni arról, hogy az ismertetett numerikus algoritmusok implementációja helyes, illetve az egyes módosítások és optimalizálási kísérletek során sem romlik el. Ezt a célt szolgálja a forráskód részét képező *project_tests* könyvtár tartalma, mely egy a *Qt Test* keretrendszerre épülő unit-teszt projekt, mely a 2.17. ábrán látható osztályokat foglalja magába (bár az általunk implementált tesztek inkább komponens-tesztek, melyek az *ECSpace*, *OrdinaryBasisFunction* és *CharacteristicPolynomial* osztályokat együttesen tesztelik, itt is a unit-teszteknél megszokott ellenőrző függvényeket használjuk, melyeket a *Qt* keretrendszer ezen alrésze biztosít, a futtatási környezet mellett).

Az *OrdinaryBasisFunctionTests* osztályban azt teszteljük, hogy a hagyományos bázisfüggvények (nullad- és magasabb rendű) deriváltjait helyesen értékeli-e ki az *OrdinaryBasisFunction* osztály. Az ott implementált Leibniz-szabály által adott eredményeket hasonlítjuk össze a kézzel kiszámított explicit alakok behelyettesítési értékével néhány rögzített esetben.

Az *ECSpaces* alkönyvtárban található tesztek azt a tényt használják ki, hogy bizonyos függvényte-



2.17. ábra. A tesztelést megvalósító osztályok

rek esetében az egyedi normalizált B-bázis explicit alakja is ismert. Ez lehetővé teszi, hogy az általános implementáció által kiszámított bázisfüggvényeket az explicit alakokkal hasonlítsuk össze (ezáltal észrevegyünk esetleges olyan eltéréseket, amik nem csak a számítási pontatlanságnak tudhatók be). A *PolynomialECSpaceTests* osztály a legfeljebb n-edfokú polinomok terében az ismert alakú Bernstein-polinomokból álló NB-bázist használja összehasonlítási alapként, míg a *T1ECSpaceTests* osztály az $\langle\{1, \sin(x), \cos(x) : x \in [0, \beta],\ \beta < \pi\}\rangle$ térben ismert NB-bázisfüggvényeket, melyek alakja:

$$
\begin{aligned}
b_{2,0}(u) &= \frac{1}{\sin^2\left(\frac{\beta}{2}\right)} \sin^2\left(\frac{\beta - u}{2}\right), \\
b_{2,1}(u) &= \frac{2\cos\left(\frac{\beta}{2}\right)}{\sin^2\left(\frac{\beta}{2}\right)} \sin\left(\frac{\beta - u}{2}\right) \sin\left(\frac{u}{2}\right), \\
b_{2,2}(u) &= \frac{1}{\sin^2\left(\frac{\beta}{2}\right)} \sin^2\left(\frac{u}{2}\right).
\end{aligned}
\tag{2.1}
$$

2.4. Kódrészlet. A *T1ECSpaceTests* osztály `valuesTest()` metódusa

```
1  void T1ECSpaceTests::valuesTest()
2  {
3      for (double input = 0; input <= 1; input += 0.001) {
4          QVERIFY2(
5              std::abs(_ecSpace->NNBBaseDerivative(0, 0, input) - explicit0(input)) < TOLERANCE,
6              "First basis function wrong");
7
8          QVERIFY2(
9              std::abs(_ecSpace->NNBBaseDerivative(1, 0, input) - explicit1(input)) < TOLERANCE,
10             "Second basis function wrong");
11
12         QVERIFY2(
13             std::abs(_ecSpace->NNBBaseDerivative(2, 0, input) - explicit2(input)) < TOLERANCE,
14             "Third basis function wrong");
15     }
16 }
```

Tekintsük például a *T1ECSpaceTests* osztály 2.4. kódrészletben látható `valuesTest()` metódusát. A $[0, 1]$ intervallumot ezred nagyságú lépésenként végigpásztázzuk és minden értékre teszteljük, hogy az `explicit0`, `explicit1` és `explicit2` függvényekben implementált (2.1)-beli $b_{2,0}(u)$, $b_{2,1}(u)$

illetve $b_{2,2}(u)$ függvényalakok behelyettesítési értékeivel közelítőleg megegyező értéket kapunk-e az *ECSpace* osztály `NNBBaseDerivative` metódusától. Amennyiben nem, a keretrendszer `QVERIFY2` makrója a megfelelő hibaüzenettel jelzi, hogy a teszteset sikertelen.

Végül a *BasisTransformationTests* osztály három különböző típusú KC-tér esetében ellenőrzi, hogy a generált bázistranszformációs mátrix helyesen alakít-e az át az NB-bázis és a hagyományos bázis között. Az alábbiakban a teszt-keretrendszer kimenete látható a *BasisTransformationTests* futtatása közben:

```
********* Start testing of BasisTransformationTests *********
Config: Using QtTest library 5.10.1, Qt 5.10.1
        (x86_64-little_endian-lp64 shared (dynamic) release build;
        by GCC 7.3.1 20180406)
PASS   : BasisTransformationTests::initTestCase()
PASS   : BasisTransformationTests::testForBernstein()
PASS   : BasisTransformationTests::testForTrigonometric()
PASS   : BasisTransformationTests::testForAlgebraicTrigonometric()
PASS   : BasisTransformationTests::cleanupTestCase()
Totals: 5 passed, 0 failed, 0 skipped, 0 blacklisted, 10ms
********* Finished testing of BasisTransformationTests *********
```

3. fejezet

Alkalmazási lehetőségek

3.1. Hatékonyság futásidő szempontjából

Az általánosság előnyei és az instabilitás megelőzésének problémája mellett a bemutatott algoritmusokban szereplő nem kevés számítás aggodalomra adhat okot a futásidő szempontjából. Az alábbi táblázatban összefoglalunk néhány átlagolt mérési eredményt.

KC-tér	Görbepontok kiértékelése (1000 darab)	Felületpontok kiértékelése (50 x 50 darab)
$\mathbb{P}_5$	4.164 ms	8.582 ms
$\mathbb{P}_{11}$	22.887 ms	47.753 ms
$\mathbb{T}_{11}$	24.204 ms	126.817 ms
$\mathbb{AT}_{13}$	66.364 ms	534.541 ms
$\mathbb{AE}_{10}$	52.468 ms	222.778 ms
$\mathbb{AET}_{13}$	195.819 ms	4044.96 ms

3.1. táblázat. Mért futásidők különböző KC-terek esetében

Az első oszlopban található jelöléseknek a következő KC-terek felelnek meg:

$$
\begin{aligned}
\mathbb{P}_5 &= \langle \{1,\, u,\, u^2,\, u^3,\, u^4,\, u^5 : u \in [0,1]\} \rangle, \\
\mathbb{P}_{11} &= \langle \{1,\, u,\, u^2,\, ...,\, u^{11} : u \in [0,1]\} \rangle, \\
\mathbb{T}_{11} &= \left\langle \left\{1,\, \cos(u),\, \sin(u),\, \cos(2u),\, \sin(2u),\, ...,\, \cos(5u),\, \sin(5u) : u \in [0, \frac{\pi}{2}]\right\} \right\rangle, \\
\mathbb{AT}_{13} &= \left\langle \left\{1,\ u^j \cos(ku),\ u^j \sin(ku) : u \in [0, \frac{\pi}{2}],\ \ j \in \{0,1,2\},\ \ k \in \{1,2\}\right\} \right\rangle, \\
\mathbb{AE}_{10} &= \left\langle \left\{1,\ u^j e^{ku} : u \in [0,1],\ \ j \in \{0,1,2\},\ \ k \in \{1,2,3\}\right\} \right\rangle, \\
\mathbb{AET}_{13} &= \Big\langle \Big\{1,\ u^j e^u \cos(u),\ u^j e^u \sin(u),\ u^j e^u \cos(2u),\ u^j e^u \sin(2u), \\
&\qquad u^j e^{2u} \cos(u),\ u^j e^{2u} \sin(u) : u \in [0,1],\ \ j \in \{0,1\}\Big\}\Big\rangle,
\end{aligned}
$$

A táblázat egyes celláiban görbék esetén 500, felületek esetén pedig 100 mérés átlaga látható. A méréseket egy *Intel(R) Core(TM) i7-3720QM* processzorral végeztük 2.6GHz-es órajel mellett. A programot a *GNU GCC* 8.1.0-s verziójával fordítottuk *-O2* optimalizációs szinten és 64 bites módban Linux kernel felett futtattuk.

A görbék esetében az értelmezési tartományon 1000 darab egyenlő közű osztópontot vettünk fel, s ezekben számoltuk ki a görbepont helyét. A felületek esetében mindkét paraméterirányban ugyanazzal a KC-térrel számoltunk és irányonként 50–50 osztópontot vettünk fel az értelmezési tartományon.

Látható, hogy magas dimenziószámú terek fölött értelmezett görbék és felületek esetében a megjelenítés már nem mondható valós idejűnek akkor, ha mozgás közben is szeretnénk látni az egyes elemeket. Ezen egyrészt javíthatunk úgy, hogy a minőségből feláldozva kevesebb osztópontot használunk. Ugyanakkor megfigyelhetjük, hogy a keverőfüggvények értékei nem változnak a kontrollpontok, illetve a teljes görbe vagy felület transzformációjakor, ezért ezeket a függvényértékeket (és deriváltakat) egy gyorsítótárban megőrizhetjük egészen addig, amíg a KC-tér, illetve az osztópontok száma változatlan marad. Az alkalmazásban a felületek megjelenítéséhez használtunk ilyen gyorsítótárat, melynek eredményeként megszűnt a látható késés az egérkurzor és egy kontrollpont mozgása között, amikor annak helyét megváltoztattuk.

3.2. Példák

3.2.1. Egy logaritmikus spirál előállítása

Az alkalmazás segítségével állítsuk elő az

$$\begin{cases} x = e^{\frac{1}{4}t}\cos(t) \\ y = e^{\frac{1}{4}t}\sin(t) \\ z = 0 \end{cases} \tag{3.1}$$

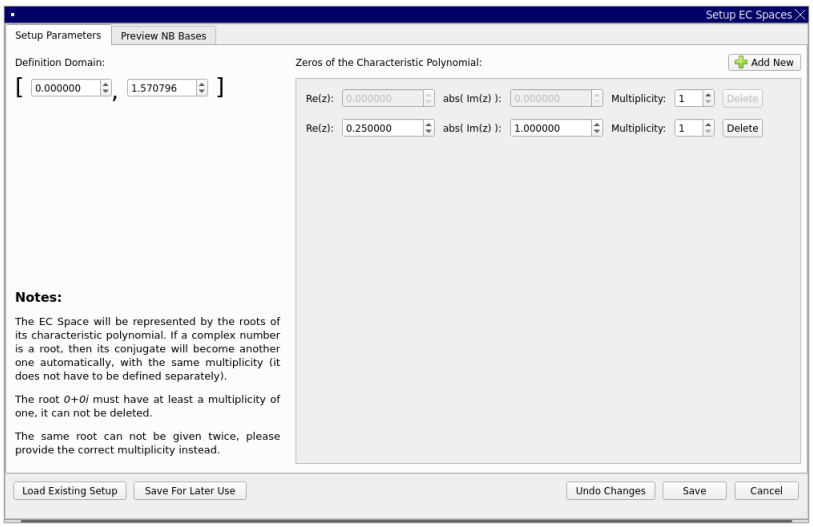
logaritmikus spirált, ahol $t \in [0, 3\pi]$.

Figyeljük meg, hogy a B-görbékkel történő egzakt leíráshoz szükséges lesz felvenni az

$$\mathbb{ET}_2^{\alpha,\beta} := \left\langle \left\{ 1,\ e^{\frac{1}{4}t}\cos(t),\ e^{\frac{1}{4}t}\sin(t) \right\} \right\rangle \tag{3.2}$$

KC-teret, melynek hagyományos bázisfüggvényeiből előállítható a (3.1)-es paraméteres alak, ahol az $[\alpha, \beta]$ intervallumot az $l'\left(\mathbb{ET}_2^{\alpha,\beta}\right)$ kritikus hossznál rövidebbre kell állítani (ezt a feltételt grafikusan fogjuk ellenőrizni). Az egzakt leírás beállításához tehát a 0 és $\frac{1}{4} \pm \mathrm{i}$ gyökök szükségesek, ugyanis a (3.2)-es tér az ezek által meghatározott $y^{(3)} - \frac{1}{2}y^{(2)} + \frac{17}{16}y^{(1)} = 0$ lineáris homogén differenciálegyenlet megoldástere.

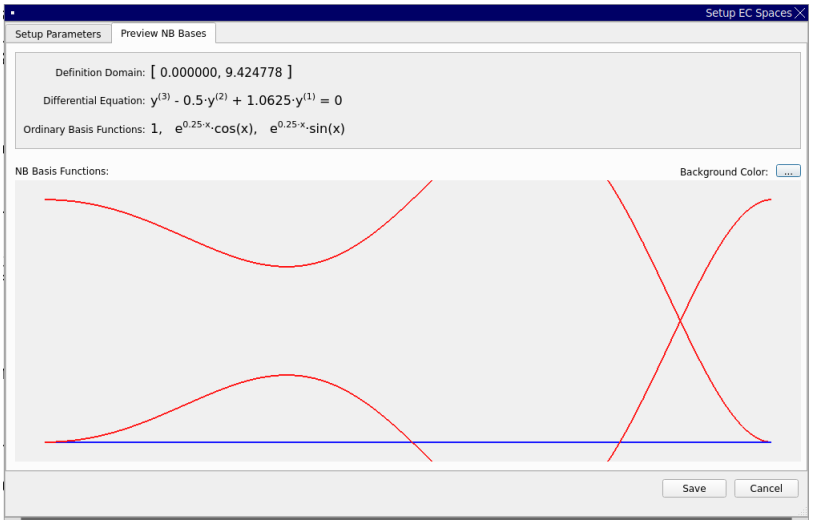
Indítsuk el az alkalmazást és az eszköztáron kattintsunk az *Add Curve* gombra. A megjelenő görbe lesz a spirál első íve. A *Curve* fülön kattintsunk a *Setup EC Space* gombra és állítsuk be a gyököket a 3.1. ábrának megfelelően. Ha a teljes $[0, 3\pi]$ intervallumot adtuk volna meg értelmezési tartománynak, a 3.2.



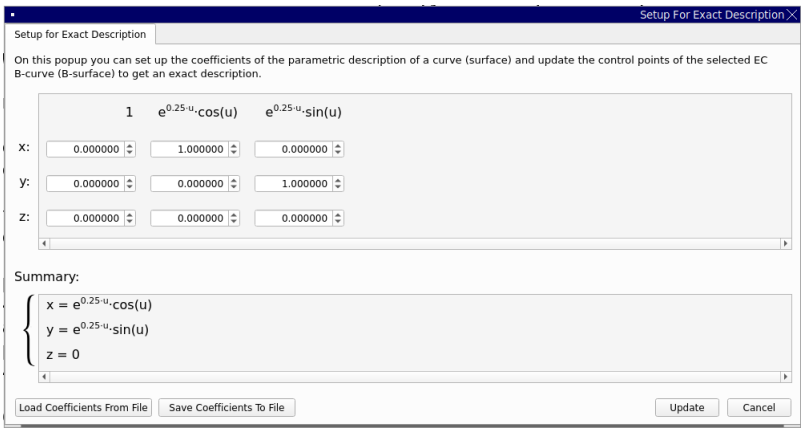
3.1. ábra. Az első ív KC-terének beállítása

ábrán látható eltorzult függvényeket láttuk volna az előnézeti fülön, a szükséges bázisfüggvényekből alkotott tér ugyanis nem KC típusú ezen az intervallumon, tehát nem létezik NB-bázis. Kísérletezéssel beláthatjuk, hogy az értelmezési tartomány $\frac{\pi}{2}$ hosszú részekre való feldarabolása már megfelel, ezért az első ív értelmezési tartományát a $\left[0, \frac{\pi}{2}\right]$ intervallumra állítottuk a 3.1. ábrán. Érdemes a beállított paramétereket elmenteni a *Save For Later Use* gombra kattintva, a további ívek esetében ugyanis csak az értelmezési tartományt kell változtatni.

Mentsük le a KC-tér beállítását, majd állítsuk be az együtthatókat az egzakt leíráshoz az *Update Control Points For Exact Description* gombra kattintva. A (3.1)-es paraméteres alaknak megfelelően a 3.3. ábrán látható együtthatókat kell megadnunk. Az *Update* gombra kattintva az első ív a végleges helyére kerül.

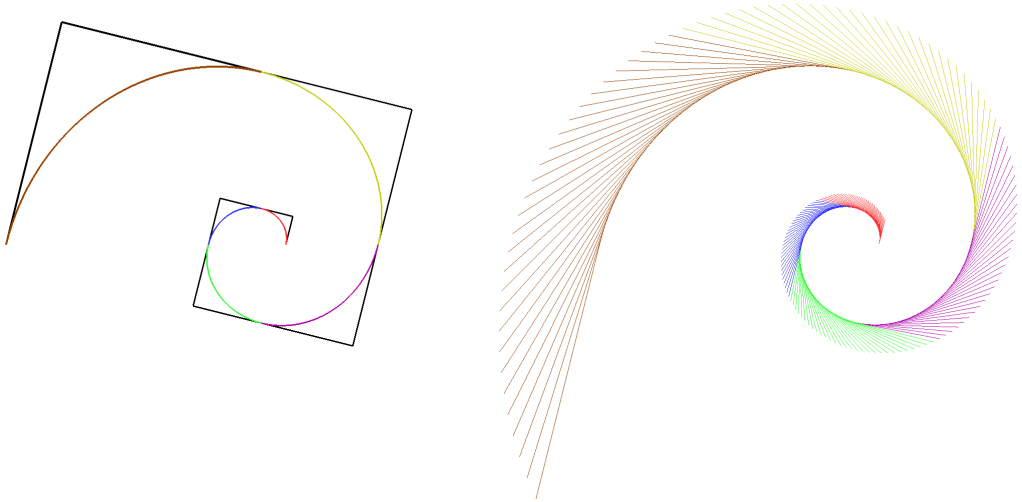


3.2. ábra. Túl nagy értelmezési tartomány, melyen a differenciálegyenlet megoldástere nem KC típusú



3.3. ábra. Az egzakt leíráshoz szükséges együtthatók

A fentebbi eljárást megismételve az értelmezési tartomány további öt darabjára az eredeti logaritmikus spiráldarab egy B-görbék segítségével történő egzakt leírását kapjuk meg, mely a 3.4. ábrán látható.



(a) az ívek és kontrollpoligonjaik (b) elsőrendű deriváltak

3.4. ábra. A (3.1)-es logaritmikus spirál

3.2.2. Egy vegyes exponenciális-trigonometrikus felület előállítása

Állítsuk elő egzakt módon a következő felületet ([Róth, 2017], (30)-as felület):

$$s(u,v) = \begin{bmatrix} s^0(u,v) \\ s^1(u,v) \\ s^2(u,v) \end{bmatrix} = \begin{bmatrix} (1-e^{\omega_0 u})\cos(u)\left(\frac{5}{4}+\cos(v)\right) \\ (e^{\omega_0 u}-1)\sin(u)\left(\frac{5}{4}+\cos(v)\right) \\ 7-e^{\omega_1 u}-\sin(v)+e^{\omega_0 u}\sin(v) \end{bmatrix}, \tag{3.3}$$

ahol $(u,v) \in \left[\frac{30\pi}{8}, \frac{50\pi}{8}\right] \times \left[\frac{-\pi}{3}, \frac{5\pi}{3}\right]$, $\omega_0 = \frac{1}{6\pi}$ és $\omega_1 = \frac{1}{3\pi}$. Ahhoz, hogy B-felülettel egzakt leírást végezzünk, u irányban az

$$\mathbb{ET}_6^{\alpha_0,\beta_0} := \left\langle \{1,\ \cos(u),\ \sin(u),\ e^{\omega_0 u},\ e^{\omega_1 u},\ e^{\omega_0 u}\cos(u),\ e^{\omega_0 u}\sin(u) : u \in [\alpha_0,\beta_0]\} \right\rangle,$$

v irányban pedig a

$$\mathbb{T}_2^{\alpha_1,\beta_1} := \left\langle \{1,\ \cos(v),\ \sin(v) : v \in [\alpha_1,\beta_1]\} \right\rangle$$

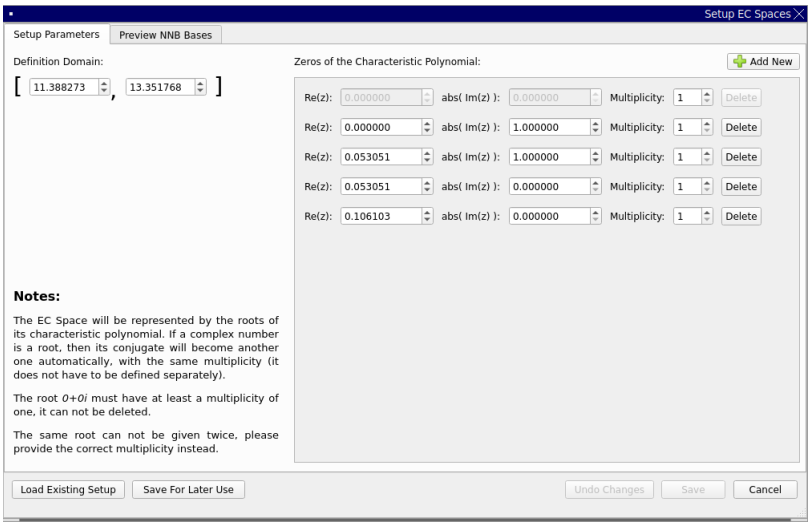
KC-tereket kell felvegyük, ahol $\beta_0 - \alpha_0 > 0$ és $\beta_1 - \alpha_1 > 0$ kisebbek kell legyenek, mint a megfelelő terek (és deriváltjaik) kritikus intervalluma (mely tulajdonságokat grafikusan fogunk ellenőrizni).

Indítsuk el az alkalmazást és a *View* fülön kattintsunk az *Add Surface* gombra. A megjelenő felület még nem a megfelelő típusú KC-terekre épül. Változtassuk meg az u paraméteriránynak megfelelő teret a *Surface* fül *Setup EC Space U* gombjára kattintva. Állítsuk be a gyököket az $\mathbb{ET}_6^{\alpha,\beta}$ tér hagyományos bázisfüggvényeinek megfelelően (ld. 3.5. ábra), majd vegyük észre, hogy ha a $\left[\frac{30\pi}{8}, \frac{50\pi}{8}\right]$ intervallumot választjuk értelmezési tartománynak, a függvények alakja nem megfelelő (vagyis kisebb intervallumon lesz csak KC típusú a tér, ez kísérletezéssel vagy az elméleti kritikus hossz kiszámításával is megkapható). Jelen esetben ilyen intervallumok lesznek a $\left[\frac{30\pi}{8}, \frac{35\pi}{8}\right]$, $\left[\frac{35\pi}{8}, \frac{40\pi}{8}\right]$, $\left[\frac{40\pi}{8}, \frac{45\pi}{8}\right]$ és $\left[\frac{45\pi}{8}, \frac{50\pi}{8}\right]$, ami azt jelenti, hogy u irányban négy sávra fogjuk osztani a felületet. A 3.5. ábrán látható esetben ezek közül az első intervallumot választottuk. Célszerű (fájlba vagy memóriába) menteni a beállítást a *Save For Later Use* (mentés későbbre) gombra kattintva, hiszen a soron következő foltok esetében már egyáltalán nem, vagy pedig csak az értelmezési tartományon kell majd változtatni.

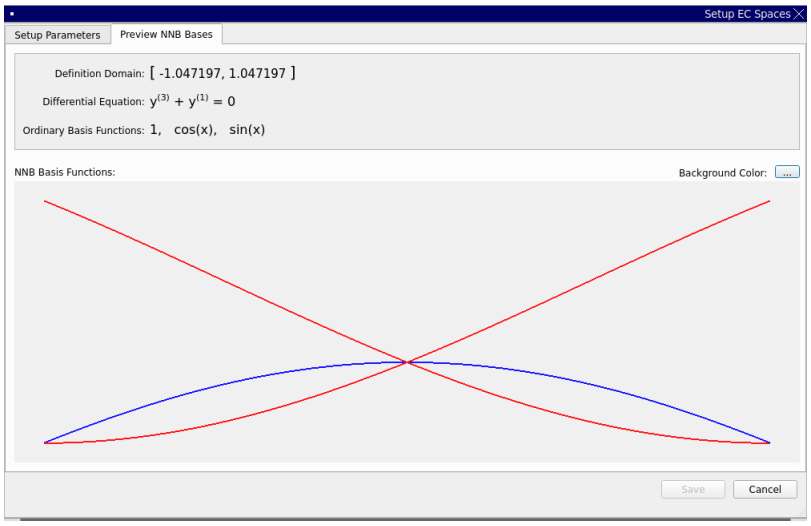
Állítsuk most be a v-irányú KC-teret. Itt csak egy valós gyököt (0) és egy komplex gyökpárt ($0 \pm 1\,\mathrm{i}$) kell megadnunk. A teljes $\left[\frac{-\pi}{3}, \frac{5\pi}{3}\right]$ intervallumon megint nem rendelkezik KC tulajdonsággal a konstrukció, viszont a $\left[\frac{-\pi}{3}, \frac{\pi}{3}\right]$, $\left[\frac{\pi}{3}, \frac{3\pi}{3}\right]$ és $\left[\frac{3\pi}{3}, \frac{5\pi}{3}\right]$ intervallumokon már igen, tehát v irányban három sávra bontjuk a felületet (így az összesen tizenkét foltból fog előállni). Használjuk most az első intervallumot ($\left[\frac{-\pi}{3}, \frac{\pi}{3}\right]$). Ekkor az NB-bázisfüggvények alakja a 3.6. ábrán látható.

Végül szükséges a kontrollpontokat beállítanunk úgy, hogy a felületi folt tényleg az eredeti felület egy (az értelmezési tartomány leszűkítései által meghatározott) darabját írja le. Ennek céljából a megfelelő együtthatókat be kell töltenünk az egzakt leírásért felelős dialógusablakba, melyet az *Update Control Points For Exact Description* gomb lenyomásával érhetünk el. A megfelelő paraméterek beállítása után (ld. 3.7. ábra) ismét érdemes az együtthatókat elmenteni, hiszen minden felületi folt esetén ugyancsak ezekre lesz szükség.

Ezek után a 3.8. ábrán látható felületi folt jelenik meg. A teljes felület leírásához az előbbi lépéseket el kell végeznünk mind a tizenkét lehetséges intervallumpárosítás esetén. Jelentősen megkönnyíthetik a



3.5. ábra. Az első felületi folt u irányú KC-terének beállítása

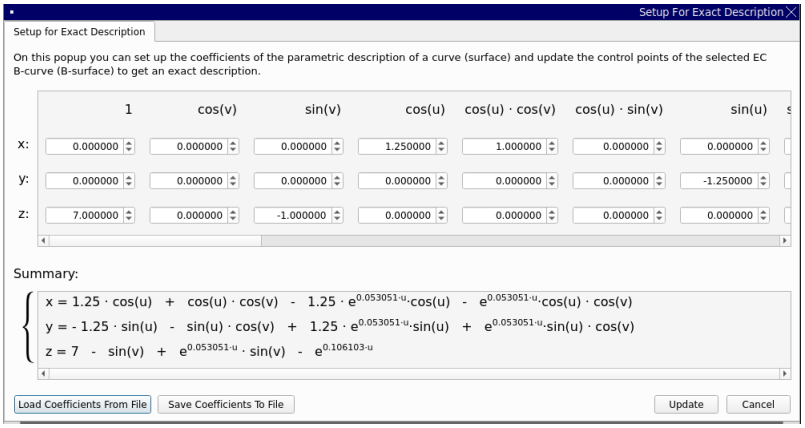


3.6. ábra. Az első felületi folt v-irányú KC-terének NB-bázisfüggvényei

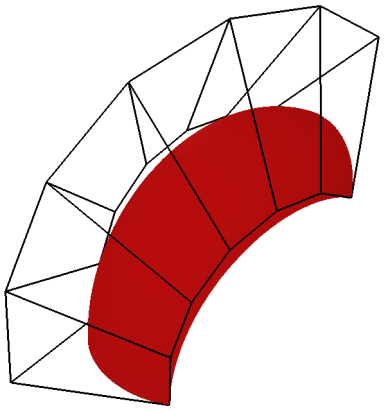
dolgunkat a menet közben készített mentések (általában elég lesz csak az intervallumokat átállítani és a kontrollpontokat befrissíteni minden folt esetén). A kész felület a 3.9. ábrán látható.

Az így előállított felület egyes darabjaira szükség szerint alkalmazhatjuk a rendszámnövelési és felosztási algoritmusokat. A kontrollpontok is elmozdíthatóak az egérrel, viszont a (nullad vagy magasabb rendben folytonos) illesztések megtartása az alkalmazásban jelenleg még nem implementált (de értékes továbbfejlesztési lehetőséget jelent).

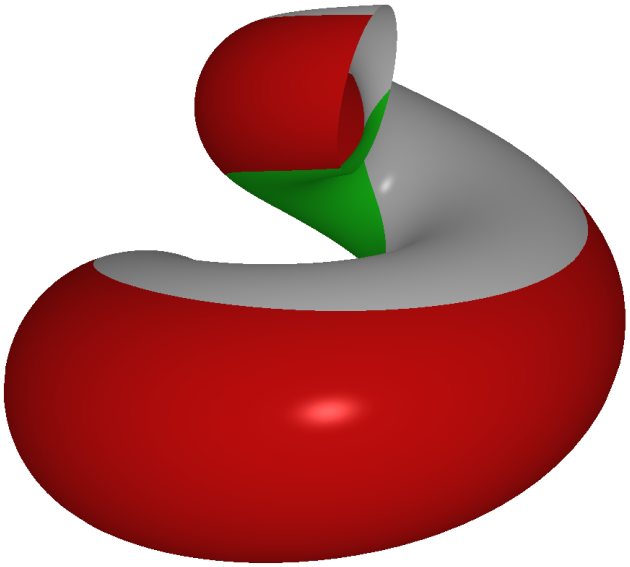
Ezek után elvégezhetjük például a 3.9. ábra egyik felületi foltjának u irányú felosztását az értelmezési tartomány harmadolópontjánál (lásd a 3.10. ábra), majd a finomabb kontroll érdekében rendszámnövelést is végrehajthatunk a kijelölt folton, például a v paraméterirányában eggyel megnövelve a 0 gyök multiplicitását (lásd a 3.11. ábra).



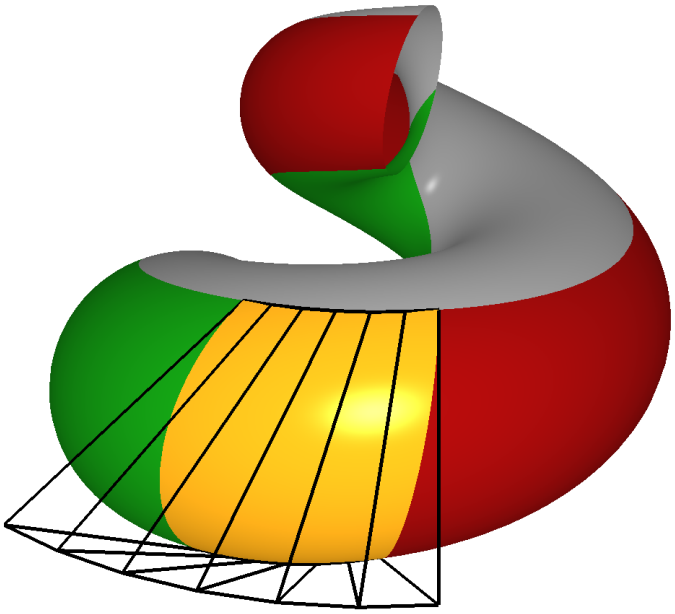
3.7. ábra. Az egzakt leírás beállítása



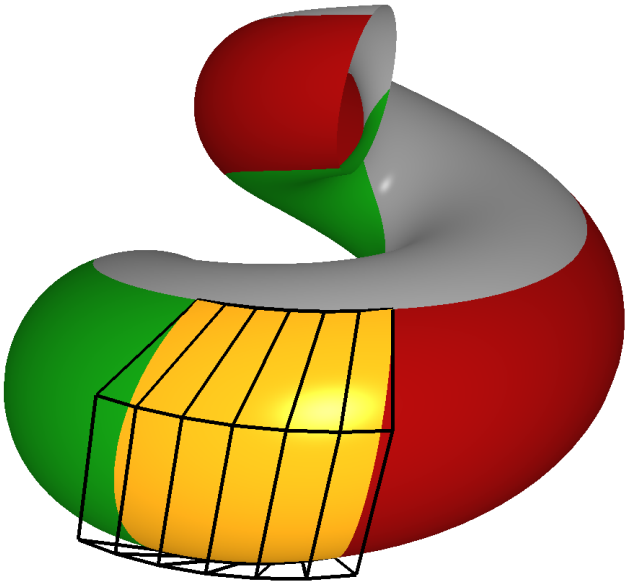
3.8. ábra. Az előállított első felületdarab (elfordítva)



3.9. ábra. A teljes előállított felület (elfordítva)



3.10. ábra. A felület egyik darabján felosztást végeztünk



3.11. ábra. Rendszámnövelés v paraméterirányban

Összefoglaló

A kiterjesztett Csebisev-terek felett értelmezett B-görbék és B-felületek az ismert és gyakorlatban használt görbe- és felülettípusok nagy részének egzakt leírását képesek általánosan megvalósítani. Hátrányuk a specializált keverőfüggvény-rendszerekkel szemben a nagyobb kiértékelési költség, illetve bizonyos esetekben az algoritmusok gyengébb numerikus stabilitása. Ugyanakkor számos előnnyel rendelkeznek:

- a függvénytér és az értelmezési tartomány módosítása által megengednek szabadon állítható alak- vagy feszültségi paramétereket;
- magasabb, vagy akár végtelen rendű (parciális) deriváltakra vonatkozó precizitást biztosítanak;
- olyan gyakorlati felhasználhatósággal és ipari jelentőséggel bíró, transzcendentális görbék / felületek egzakt leírását is lehetővé teszik, melyek a napjaink modellezőrendszereiben szabványként használt, nem feltétlenül egyenletes csomóvektorú, racionális B-spline-görbékkel / felületekkel legfeljebb csak közelíthetőek;
- a hagyományos paraméteres alakban adott, de nem törtfüggvényekkel leírt görbék / felületek esetén csupán kontrollpont-alapú, súlyvektor / súlymátrix nélküli egzakt előállítást biztosítanak;
- kondíciószám és numerikus kiértékelés szempontjából, valamely függvénytér nemnegatív normalizált B-bázisa az egyértelmű olyan teljesen pozitív, lineárisan független normalizált függvényrendszer, mely optimálisan stabil (ld. [Mainar és Pena, 1999, 3.4. következmény]) az adott függvénytér összes nemnegatív bázisa közül.

A releváns elméleti eredmények felidézése után bemutattuk a dolgozatot kísérő alkalmazást, mely ezen eredményeket próbálja meg gyakorlatba ültetve szemléltetni mindenek előtt a kísérleti tanulmányozhatóság céljából. Megvalósítottuk az NB-bázisfüggvények (nullad- és magasabb rendű deriváltjainak) kiértékelését, a görbék és felületek rendszámnövelését és felosztását, illetve a paraméteres alakban megadott görbék és felületek egzakt leírását.

Az implementáció során tanulságos volt az egyes egéresemények (kijelölések, mozgások) kezelésének megvalósítása is, illetve a *Qt* keretrendszer signal–slot mechanizmusának felhasználása a felhasználói felület bonyolultabb elemei közötti kommunikáció megvalósítására.

Az alkalmazásban nem szereplő, de hasonló modellező rendszertől elvárható tulajdonságok száma jelentősen nagy, továbbfejlesztési lehetőség például az elemek (adott folytonossági rendű) illesztése, egy a változtatások visszafordítására alkalmas kiegészítés (undo-redo mechanizmus), illetve foltokból álló sávok kigenerálásának lehetősége (például az adott értelmezési tartomány automatikus felosztása révén).

Irodalomjegyzék

Carnicer, J.-M. és Peña, J. M. On transforming a Tchebycheff system into a strictly totally positive system. *Journal of Approximation Theory*, 81, 2:274–295, 1995.
DOI: http://dx.doi.org/10.1006/jath.1995.1050.

Carnicer, J.-M., Mainar, E., és Peña, J. M. Critical length for design purposes and extended Chebyshev spaces. *Constructive Approximation*, 20, 1:55–71, 2004.
DOI: http://dx.doi.org/10.1007/s00365-002-0530-1.

Farin, G. *Curves and surfaces for CAGD: a practical guide (5th ed.)*. Morgan Kaufmann, San Francisco, 2002.

Juhász, I. *Számítógépi geometria és grafika*. Miskolci Egyetemi Kiadó, 1995.

Karlin, S. és Studden, W. *Tchebycheff systems: with applications in analysis and statistics*. Wiley, New York, NY, 1966.

Kessenich, J. M., Sellers, G., és Shreiner, D. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.5 (9th edition)*. Addison–Wesley Professional, 2016.

Mainar, E. és Pena, J. M. Corner cutting algorithms associated with optimal shape preserving representations. *Computer Aided Geometric Design*, 16, 9:883–906, 1999.
DOI: http://dx.doi.org/10.1016/S0167-8396(99)00035-7.

Mazure, M.-L. Chebyshev–Bernstein bases. *Computer Aided Geometric Design*, 16, 7:649–669, 1999.
DOI: http://dx.doi.org/10.1016/S0167-8396(99)00029-1.

Mazure, M.-L. és Laurent, P.-J. Nested sequences of Chebyshev spaces and shape parameters. *RAIRO–Modélisation mathématique et analyse numérique*, 32, 6:773–788, 1998. URL `http://www.numdam.org/article/M2AN_1998__32_6_773_0.pdf`.

Róth, Á. An OpenGL and C++ based function library for curve and surface modeling in a large class of extended Chebyshev spaces (*submitted to ACM Transactions on Mathematical Software*). 2017. URL `https://arxiv.org/abs/1708`.

Róth, Á. Control point based exact description of curves and surfaces in extended Chebyshev spaces. *Computer Aided Geometric Design*, 40:40–58, 2015.
DOI: http://dx.doi.org/10.1016/j.cagd.2015.09.005.

Róth, Á. Computer Aided Geometric Design, 2018. URL `https://sites.google.com/site/computeraidedgeometricdesign/`. Előadásjegyzet, megtekintve 2018. május 1.

Stroustrup, B. *The C++ Programming Language, 4th edition*. Addison–Wesley, 2013.