



Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Irányítástechnika és Informatika Tanszék

Poliéderlekerekítő algoritmusok

SZAKDOLGOZAT

Készítette
Bukits Tamás

Konzulens
Dr. Várady Tamás László

2021. május 23.

Tartalomjegyzék

Kivonat	i
Abstract	ii
1. Bevezetés	1
1.1. Előzmények, motiváció	1
1.2. 3D számítógépes geometria	1
1.3. Szabadformájú testek	1
1.4. A diplomaterv felépítése	2
2. Kontrollpoliédereken alapuló technikák	3
3. Felületgeneráló algoritmus	4
3.1. Az algoritmus áttekintése	4
3.2. Input: kontrollpoliéder	5
3.3. Konkáv lapok konvexekre osztása	5
3.4. Az X-konstrukció	5
3.4.1. Az X-konstrukció lépései	6
3.4.2. Az X-csúcs tartópontjainak meghatározása	7
3.5. Határológörbék illesztése	13
3.5.1. Hármas segédstruktúra	13
3.5.2. Bézier-görbék illesztése	14
3.5.2.1. Kontrollpontok meghatározása	14
3.5.2.2. Bézier-görbék egyenlete	15
3.5.2.3. Bernstein-függvények	15
3.6. Felületelemek (patch-ek) illesztése	16
3.6.1. Hármas segédstruktúra illesztése 4-oldalú patch-ekre	16
3.6.2. Bézier-felületek illesztése	16
3.6.2.1. Kontrollpontok meghatározása	16
3.6.2.2. Bézier-felületek egyenlete	18
3.6.3. Hármas segédstruktúra illesztése N-oldalú patch-ekre	18
3.6.3.1. Az n oldalú patch-ek határológörbéinek felezése	19
3.6.3.2. Belső pontok meghatározása	21
3.6.3.3. Belső twist kontrollpontok meghatározása	26
4. Szomszédos felületek sima összekapcsolódása	27
4.1. Parametrikus és geometriai folytonosság	27
4.2. Közös irányblend technika	28
5. Tesztrendszer	30
5.1. A tesztprogram szervezése	30

5.2. A tesztprogram felhasználói felülete	31
5.3. Poliéderek beolvasása	32
5.4. Az X-struktúra létrehozása	32
5.5. Az X-csúcs tartópontjainak kiszámítása	33
5.6. Görbék illesztése	33
5.7. Felületelemek illesztése	33
5.7.1. STL fájlok használata	33
5.7.2. Egységes orientáció beállítása	34
6. Teszteredmények	35
6.1. Az algoritmus végrehajtása egy egyszerű modellen	35
6.2. Harmadfokú lekerekítés különböző értékekkel	39
6.3. Az algoritmus végrehajtása komplex modellekben	40
7. Összefoglalás és további tervezetek	42
Köszönetnyilvánítás	43
Irodalomjegyzék	44

HALLGATÓI NYILATKOZAT

Alulírott *Bukits Tamás*, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2021. május 23.

Bukits Tamás
hallgató

Kivonat

A 3D-s geometriai modellezés egy jól ismert módszere a kontrollpoliédereken alapú tervezés, mely segítségével indirekt módon lehet létrehozni komplex szabadformájú objektumokat. A legelterjedtebb technika a rekurzív felosztás, ahol egyszerű szabályok alapján a poliéder addig finomítjuk, míg az vizuálisan folytonos nem lesz. Jelen szakdolgozatban a határolófelületeket közvetlenül, egy lépésben állítjuk elő. A reprezentáció az ún. X-struktúrán alapul, amelyből különböző módokon lehet előállítani sima kompozit felületeket. Mindegyik módszer szabványos, simán összekapcsolt Bézier-felületeket állítanak elő; különböző görbék és geometriai folytonosságok biztosítása mellett.

Dolgozatomban bemutatom az ún. hármas segédstruktúrát, amely egyszerre tudja kezelní az eltérő fokszámú görbékét. Ismertetem a kontrollpontok meghatározását, valamint a szomszédos patch-ek sima összekapcsolódásához szükséges algebrai feltételeket (G^1 geometriai folytonosság, közös irányblend).

A fenti technika szemléltetésére egy 3D-s tesztprogramot fejlesztettem, amely egy beolvasott kontrollpoliéder alapján simán kapcsolódó felületeket állít elő. A felhasználónak lehetősége van három módszer alkalmazására, és az ezek alapját képező geometriai jellemzők megjelenítésére is. A dolgozat végén néhány tesztpéldával illusztráltam az egyes algoritmusok eredményét.

Abstract

A well-known method in 3D geometric modelling is control polyhedron based design, which allows to create complex free-form objects in an indirect way. The most widespread technique is recursive subdivision where we smooth the polyhedron until it becomes visually continuous. In the present thesis, the boundary surfaces are produced directly in one step. The representation is based on the so-called X-construction, by means of which smooth, composite surfaces can be produced in different ways. Each method produces standard, smoothly connected Bézier-surfaces; while providing different types of boundary curves and geometric continuities.

In my thesis I present the so called three-chord segment structure which can handle curves with different degrees. I describe the definition of its control points and the necessary algebraic conditions for the smooth connection of adjacent patches (G^1 geometric continuity, common direction blending).

To illustrate the above technique, I have developed a 3D test program that produces a smoothly connected surface using a given control polyhedron. The user also has the option of applying three methods and displaying the geometric features of the constructions. In the last section of the thesis, I also illustrate the results of each algorithm by some test examples.

1. fejezet

Bevezetés

Ebben a fejezetben bemutatom a dolgozat előzményeit és motivációját, értelmezem a kapott feladatot, példákat mutatók a felhasználására a gyakorlatban, majd röviden ismertetem a szakdolgozatom felépítését.

1.1. Előzmények, motiváció

Érdeklődési körömbe a számítógépes grafika, képi megjelenítés, geometriai tervezés és az ezekhez alapul szolgáló matematika tartozik, ezért a választásom a 3D-s számítógépes geometria és alakzatrekonstrukció tématerületre esett a témalaboratórium tantárgy félévében. Első feladatként két tetszőleges fokszámú Bézier-felületet kellett összeillesztenem úgy, hogy a kapcsolódást az összeillesztési határ mentén ne lehessen szabad szemmel észrevenni. Az elkövetkező két félévben önálló laboratórium és diplomatervezés tantárgy keretében az előző félévben megszerzett ismereteket felhasználva, egy olyan algoritmus fejlesztése volt a feladatom, amely kontrollpoliéderkből simán lekerekített alakzatokat tud létrehozni.

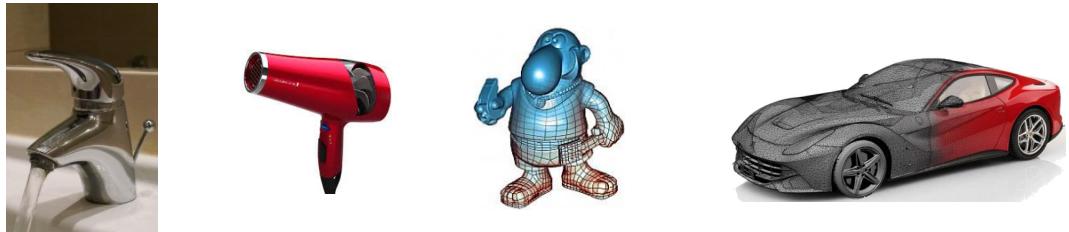
1.2. 3D számítógépes geometria

A 3D-s számítógépes geometria három dimenziós pontfelhők, poligonhálók, görbék és felületek, valamint szilárd testek számítógépes reprezentációjával (adatstruktúrák, matematikai egyenletek), legfontosabb algoritmusaival és ezek alkalmazásával foglalkozik. Segítségével digitálisan reprezentálhatjuk akár a valós, akár egy virtuális világ objektumait. Előbbiit számítógéppel segített tervezésnek (CAD), utóbbiit digitális alakzat rekonstrukciónak (DSR) nevezzük. Az így előállított digitális objektumok felhasználása számos területen elterjedt: mérnöki alkalmazásokban, orvostudományban, szórakoztatáiparban (játékok, filmek).

1.3. Szabadformájú testek

Sok olyan objektum vesz körül bennünket, ahol a határoló felületeket nem lehet egyszerű szabályos felületekkel (sík, henger) reprezentálni. Ilyen objektumok lehetnek például: autókarosszéria, használati tárgyak (hajszáritó, kanna, bukósisak, stb.), dísztárgyak (lámpa, bútor, stb.), animációkban felhasznált karakterek (1.1. ábra).

Különböző módon lehet ilyen felületeket létrehozni. Egy jól ismert módszer a kontrollpoliédereken alapuló tervezési eljárás, amely indirekt módon definiálja a kívánt szabadformájú testet. A bemenetként megadott kontrollpoliéderlek lekerekítésével lehet eljutni különböző sima szabadformájú objektumokig.



1.1. ábra. Szabadformájú objektumok [1]

1.4. A diplomaterv felépítése

Szakdolgozatom második fejezetében ismertetem, hogy milyen kontrollpoliéder algoritmusok léteznek. Szakdolgozatom a [16] cikkben bemutatott, úgynevezett X-konstrukció módszer alkalmazására épül, amelyet a harmadik fejezetben mutatok be lépésekre lebontva, matematikai vezetésekkel és indoklással kísérve. A negyedik fejezetben egy, az algoritmus futásakor felmerülő folytonossági problémát ismertetek [14]. A következő fejezetben a félévek alatt fejlesztett tesztprogramot mutatom be, és ismertetem a legfontosabb felhasználói parancsokat. Kitérek a tesztprogram fejlesztése során felmerült problémákra. Az alkalmazás eredményeit a hatodik fejezetben mutatom be tesztpéldákon keresztül. Először egyszerű modelleken, később bonyolult, látványos modelleken. Dolgozatomat összegzéssel és a további tervek felvázolásával zárom. A vonalas ábrák a Dia nevű [4] rajzolóprogrammal, az összes többi ábra a tesztprogram segítségével készültek.

2. fejezet

Kontrollpoliédereken alapuló technikák

Széles körben alkalmazott, klasszikus technika komplex 3D-s szabadformájú objektumok létrehozására a kontrollpoliéder alapú tervezés.

Számos vonatkozó algoritmust publikáltak, melyek két nagy csoportba sorolhatók, lásd [6]. Az első csoportba az ún. *rekurzív felosztáson* alapuló eljárások tartoznak, itt az input poliéderből, a tartópontok lineáris kombinációja alapján egy új finomított poliéderet hoznak létre, majd az eljárást rekurzíven alkalmazzák, mindaddig míg a poliédersorozat egy kellően sima felületreprezentációhoz vezet. A legismertebb ilyen eljárások a Doo-Sabin [5] és a Catmull-Clark felosztás [3].

A második csoportba tartozó módszerek nem rekurzív módon definiálják az objektumot, hanem *közvetlenül*, különböző szabályrendszerek alapján hoznak létre egy egymáshoz simán kapcsolódó felületegyüttest, tehát ebben az esetben nincs szükség iterációra. A közvetlen módszereket sok szempontból lehet tovább osztályozni – (i) hogyan keletkezik a patch struktúra a poliéderből, (ii) milyen patch reprezentációt alkalmazunk, azaz szabványos tenzorszorzat patch-eket (pl. Bézier vagy B-spline) rakunk össze vagy n-oldalú patch-ekkel dolgozunk. Az előző módszer előnye, hogy szabványos reprezentáció jön létre, ellenben ezek a módszerek bonyolultabbak, mint az n-oldalú felületreprezentációk, és ügyelni kell ezek megfelelő összeillesztésére. Egy további fontos szempont, hogy (iii) az egymás mellett lévő patch-ek minden folytonossággal kapcsolódnak össze. Például tenzorszorzat felületeket alkalmaznak a [11] vagy [8] cikkekben; n-patch-eket használnak a [7] vagy [15] cikkekben.

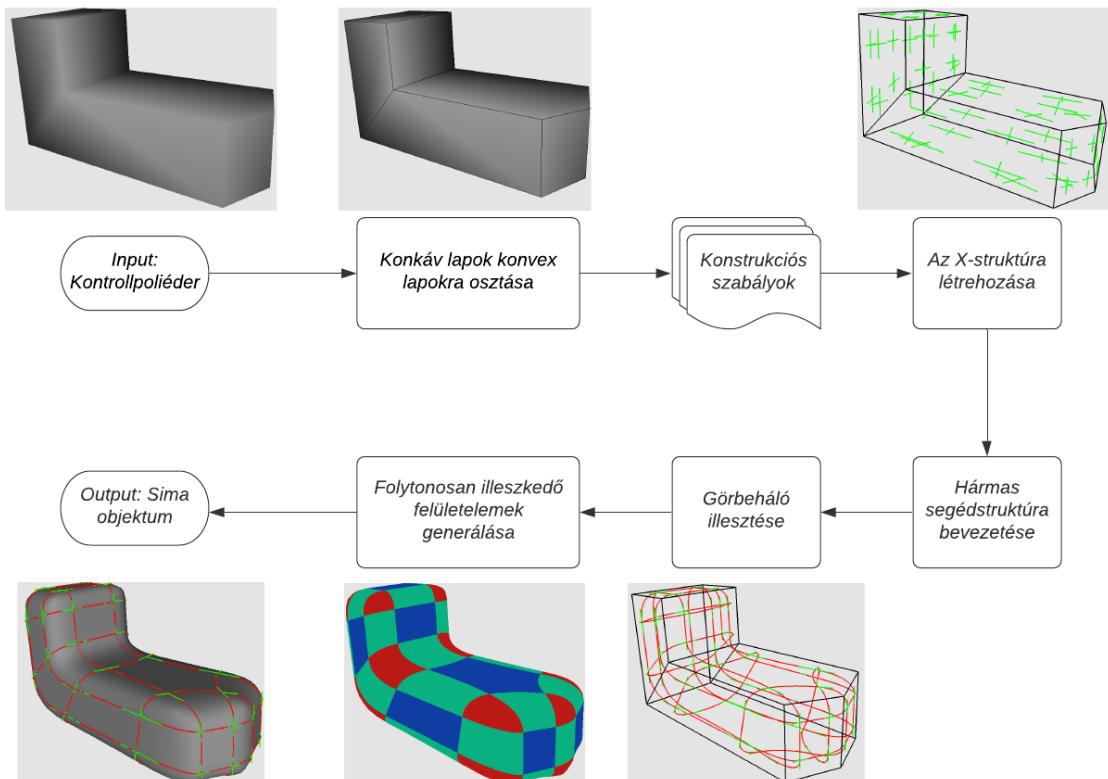
Jelen szakdolgozat a tenzorszorzat megközelítést vizsgálja egy régi publikáció [16] alapján. Az ún. X-konstrukció mindig egy Doo-Sabin-szerű topológiai struktúrán alapul, és alacsony fokszámú (másod-, harmad- vagy negyedfokú) Bézier felületeket fűz össze. Amint ezt a későbbiekben kifejtem, különböző felületreprezentációkat lehet létrehozni, különböző szabályrendszerek alkalmazásával. Ily módon minden kontrollpoliéderhez többféle, különböző formájú és matematikai tulajdonságokkal rendelkező felületegyüttes is rendelhető. A felhasználónak kell kiválasztania, hogy ezen reprezentációk közül melyik illeszkedik legjobban az adott alkalmazáshoz.

3. fejezet

Felületgeneráló algoritmus

Ebben a fejezetben bemutatásra kerül a kiinduló kontrollpoliéderből sima felületeket generáló algoritmus. A fejezet bevezető pontjában egy áttekintő diagrammot (3.1. ábra) ismertetek az algoritmus lényegi lépéseiiről, felhasználva a programom egy tesztobjektumát. A további alpontokban az algoritmus részleteit fogom kifejteni matematikai bizonyításokkal és képletekkel.

3.1. Az algoritmus áttekintése



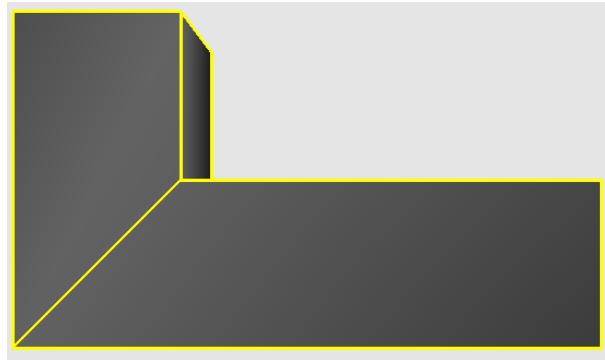
3.1. ábra. A felületgeneráló algoritmus blokkdiagramja

3.2. Input: kontrollpoliéder

A program bemenete egy általános topológiájú kontrollpoliéder, amelynek n oldalú lapjai és n vegyértékű csúcsai vannak. Ennek segítségével szeretnénk meghatározni a kívánt szabadformájú objektumot.

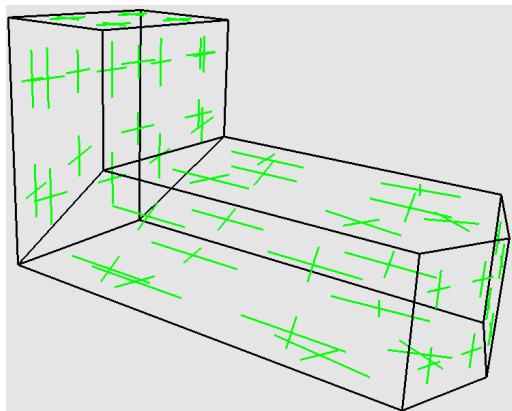
3.3. Konkáv lapok konvexekre osztása

Az algoritmus előfeldolgozásaként a bemeneti kontrollpoliéder konkáv lapjait fel kell osztani konvex lapokra; másszóval „virtuális” éleket kell hozzáadnunk a struktúrához (3.2. ábra). Ez a felosztás gyakran nem egyértelmű, általában a legrövidebb összekötő éleket célszerű beiktatni.



3.2. ábra. L lapka konvex felosztása

3.4. Az X-konstrukció



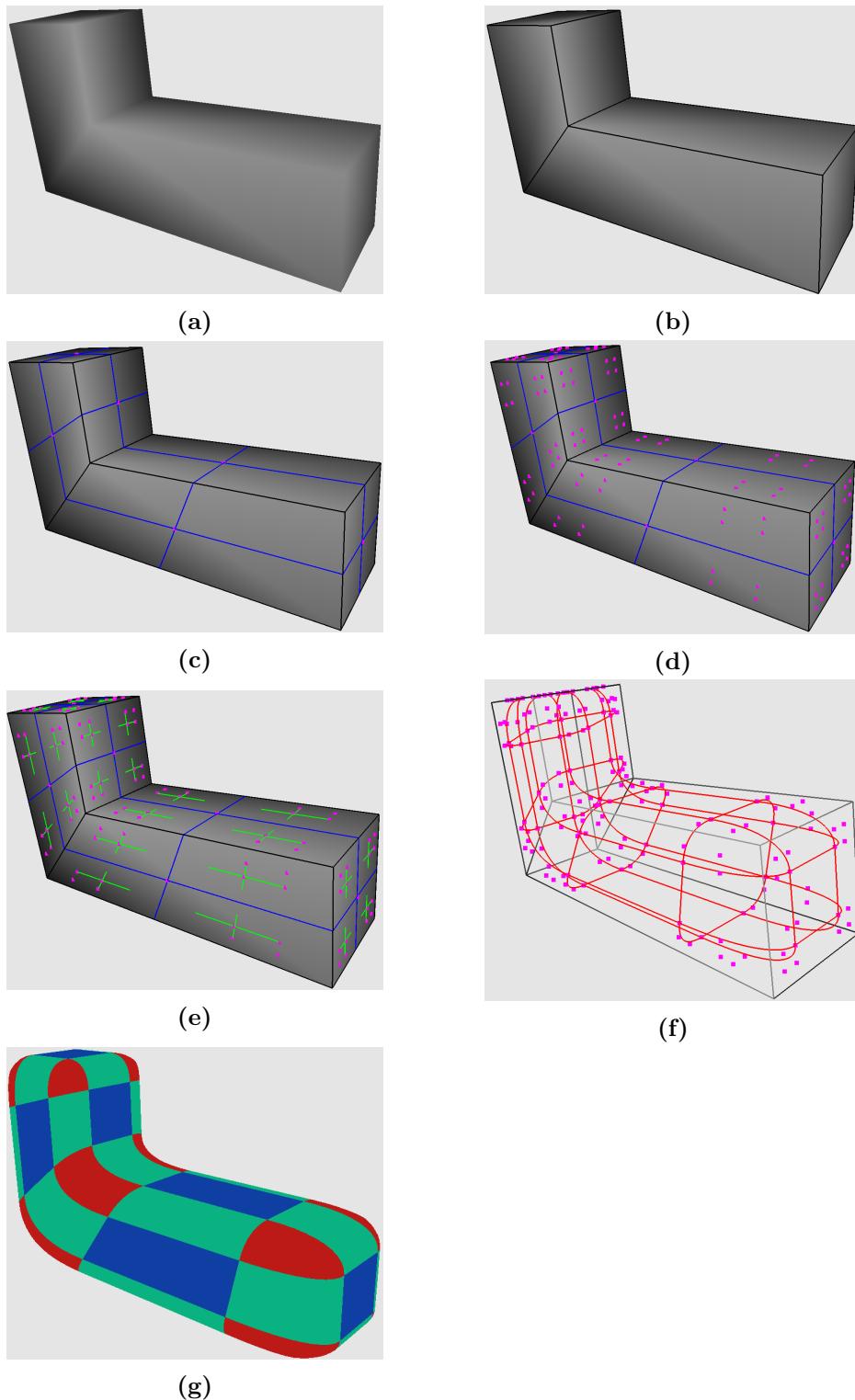
3.3. ábra. A konstrukció alapját képező X-csúcsok

kezelni tudjuk. Ezt a struktúrát *hármas segédstruktúrának* nevezik. Ezt követően a Bézier felületek létrehozásának az algoritmusát tekintem át.

Az algoritmus az ún. *X-konstrukció* létrehozásán alapul, amellyel egy kontrollpoliéderen simítás és lekerkítés operációkat hajthatunk végre. Az X-konstrukció egy speciális görbehálózatot hoz létre másodfokú vagy harmadfokú szegmensekből, amelyek az eredeti kontrollpoliéderen végrehajtott különböző konstrukciós szabályok alapján jönnek létre. Közvetve keresztirányú derivált függvényeket is definiálunk, amelyeket másod- és harmadfokú Bézier-felületekkel fogunk interpolálni, ezzel G^1 folytonosságot biztosítva az alkotó elemek között. Az első alfejezetben egy áttekintést adok, az X-konstrukció létrehozásának lépéseiiről. A folytatásban részletesen kifejtem a konstrukciós szabályokat. Mivel különböző fokszámú szegmenseink jönnek létre, ezért egy közös struktúra létrehozására van szükség, amellyel minden esetet

3.4.1. Az X-konstrukció lépései

Ebben a fejezetben az X-konstrukció lépéseit fogom ismertetni. A 3.4/b6 ábra lépése mutatja a konkáv lapok konvexekre osztását. A konstrukció lényegi lépései innen indulnak.



3.4. ábra. Az X-konstrukció lépései

A kontrollpoliéder összes lapján egy felosztást alkalmazunk, amely ezekből a lapokból annyi belső négyszöget készít, amennyi az adott lapon lévő csúcsoknak a száma. Ez a

"klasszikus" felosztásos lépés olyan élek behúzásával fog történni, amelyek a kontrollpoliédernek éleinek felezőpontjait kötik össze a megfelelő lap középpontjával (3.4/c ábra). Így az algoritmus előállítja az úgynevezett QC objektumokat, amelyek csak konvex négyzetekből tevődnek össze.

A továbbiakban minden felosztott konvex négyzetre létre fogok hozni egy-egy X -csúcset (3.4/e ábra), amelyek kontrollponjainak (3.4/d ábra) kiszámítása különböző szabályok szerint történhet. Ezeket a számítási eljárásokat a 3.4.2. fejezetben fogom részletesen kifejteni. Ezek az X -csúcsok lekerekítő funkció alkalmazásakor pontosan a négyzöglapon fognak elhelyezkedni, más esetben (pl. simítás) pedig a négyzög közelében. Az X -csúcsokhoz hozzárendelünk keresztirányú érintőket is, innen kapta a struktúra az X elnevezést. Ezek négy irányba egy-egy görbét fognak definiálni, amelyek a definícióból adódóan érintőfolytonosak lesznek. Az előbb említett tangens pontok fogják meghatározni ezeknek a görbéknek az első deriváltjait.

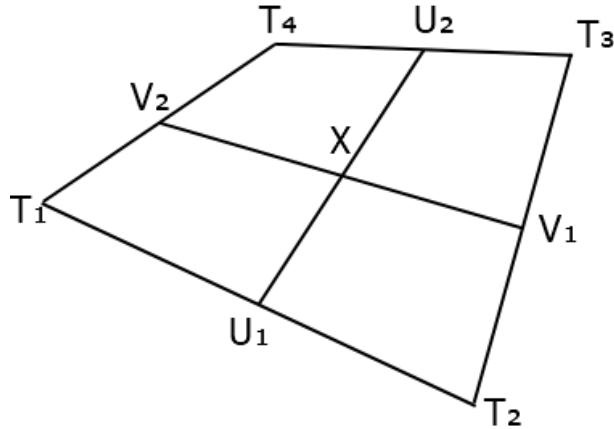
Két szomszédos X -csúcs megfelelő tangens pontjainak összekötésével előáll a kontrollpoliéderre illeszkedő görbeháló (3.4/f ábra). Attól függően, hogy az algoritmus melyik konstrukciós szabályt alkalmazta, a görbehálót másodfokú vagy harmadfokú görbeszegmensek alkotják. Ezért lesz fontos egy közös struktúra bevezetése, amely minden esetet tudja kezelnı.

Ahhoz, hogy egy lekerekített objektumot kapjunk az eredeti kontrollpoliéderből, az előző lépésben előállított Bézier-görbevázra felületelemeket, úgynevezett „patch”-eket (innenől így fogok a felületekre hivatkozni) kell illesztenünk (3.4/g ábra). A patch-eket körülvevő X -csúcsok darabszámától és elhelyezkedésétől függően különböző típusú patch-ek jöhettek létre a lekerekített objektumon. Azok az X -csúcsok, amelyek az eredeti alakzat egy csúcsa körül helyezkednek V (vertex) típusú patch-eket fognak alkotni (3.4/g ábrán piros színnel jelölt felületek). A V-patch oldalainak száma attól függ, hogy az adott csúcsban hány él fut össze. E (edge) típusú patch-eket alkotnak azok az X -csúcsok, amelyek az eredeti élen egy élfelező pont körül helyezkednek el (3.4/g ábrán zöld színnel jelölt felületek). Az E típusú éleket mindig négy X -csúcs veszi körbe. F (face) típusú lapok jönnek létre azon X -csúcsok között, amelyek az eredeti kontrollpoliéder lapjaihoz tartoznak (3.4/g ábrán kék színnel jelölt felületek). Az F-patch oldalainak számát az határozza meg, hogy az eredeti lapnak hány élé van.

3.4.2. Az X -csúcs tartópontjainak meghatározása

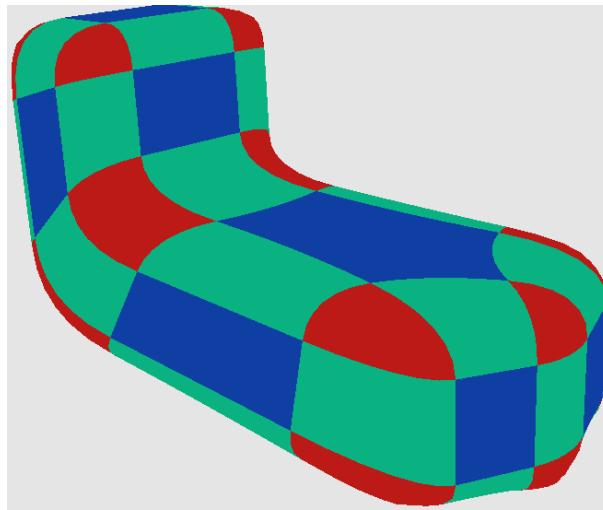
Az előbbi fejezetben létrejött X -csúcsok különböző konstrukciós szabályoknak megfelelően jönnek létre. Ezek a kényszerek különféle módon határozzák meg az X -csúcs X középpontját, amelynek következtében a tangens kontrollpontok is eltérőek lesznek. Ez a négy pont határozza meg az eredeti kontrollpoliéder új „éleinek” meredekségét, amelyre majd a felületeket fogja illeszteni az algoritmus egy későbbi feldolgozása során. minden X -csúcshoz a konstrukció definiál egy twist-vektort, amely meghatározza a felület befelé irányuló görbületét. Ezek a twist-vektorok parciális deriváltak kombinációjából tevődnek össze, amelyek indirekt meghatározzák a másodfokú vagy harmadfokú keresztdériváltakat az X élei mentén. A twist mennyisége négy kezdő twist kontrollpontból áll, amelyek közül az egyik szabadon választható, a többi pedig egyedileg meghatározható. Az 3.5. ábrán $U_i, V_i (i = 1, 2)$ jelöli a tangens, $T_i (i = 1, \dots, 4)$ pedig a twist kontrollpontokat.

Ezek a twist mennyiségek a dolgozat későbbi fejezetében fontos szerepet kapnak. Segítséggel az X -csúcsból induló görbék „teltség”(fullness) paraméterét és a keresztirányú deriváltak „sebességét” lehet befolyásolni, ezáltal biztosítva a G^1 folytonosságot az illeszkedő felületelemek között.



3.5. ábra. Az X kontroll pontjai

1. szabályrendszer (másodfokú lekerekítés)



3.6. ábra. Másodfokú lekerekítés

A másodfokú lekerekítés az egyik legegyszerűbb technika szabad formájú objektumok generálására, ahol az algoritmus távolságokat oszt fel arányosan (3.7. ábra). Először meghatározzuk a konvex lap középpontjait, majd további felezésekkel előáll a négyzet középpontja is X , amely az X-csúcs középpontjai lesznek. Az algoritmus következő lépései kiszámítja a C, E_1, P, E_2 pontok által meghatározott négyzet tangens és twist kontrollpontjait, a pontok lineáris kombinációjaként. Ahhoz, hogy előálljanak az X-csúcs U, V, T pontjai, be kell vezetni H_i pontokat amelyek felezik a megfelelő E_i és az aktuális P csúcs közötti távolságot. Az algoritmus későbbi lépéseiben a tangens kontrollpontokra görbüket fogunk illeszteni. Az X-csúcsból induló négy ilyen görbét alkotó tangens kontrollpontból már kettő előállt U, V , azonban kettő még hiányzik U^*, V^* . Ahhoz, hogy ezeket az ellentétes irányokban levő tangens kontrollpontokat és a további twist T_1, T_2, T_{12} kontrollpontokat is meghatározzuk, szükségünk lesz még két segédpont bevezetésére, Q_1, Q_2 . Ezekkel a segédpontokkal az eddigi T, U, V kontrollpontok meghatározásához hasonlóan a befoglaló négyzetek átlagából számíthatóak az új pontok. Az eddigi elmondottak alapján az alábbi kényszeregyenleteket lehet felírni az X-csúcs tartópontjainak kiszámításához:

Segédpontok:

$$\begin{aligned} H_1 &= 0.5(E_1 + P) & H_2 &= 0.5(E_2 + P) \\ Q_1 &= 0.5(C + E_1) & Q_2 &= 0.5(C + E_2) \end{aligned} \quad (3.1)$$

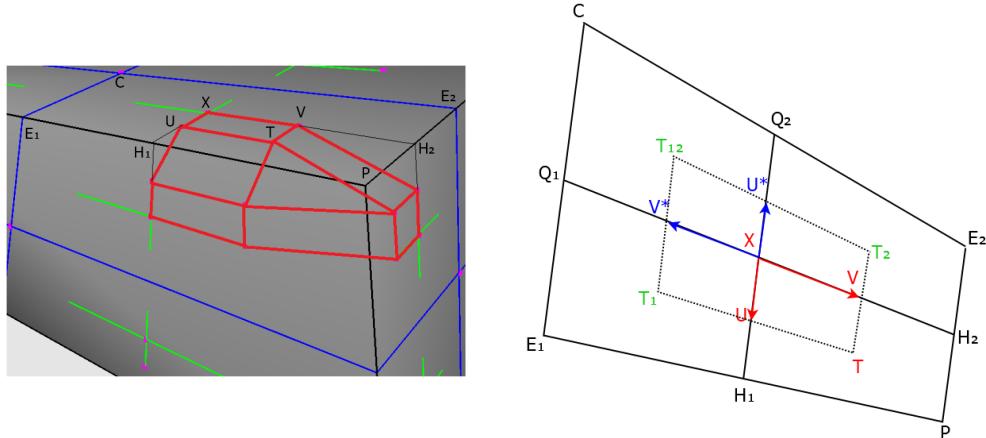
Az X-csúcs tangens pontjai:

$$\begin{aligned} X &= 0.25(C + E_1 + E_2 + P) \\ U &= 0.5(H_1 + X) & V &= 0.5(H_2 + X) \\ U^* &= 0.5(Q_1 + X) & V^* &= 0.5(Q_2 + X) \end{aligned} \quad (3.2)$$

Az X-csúcs twist pontjai:

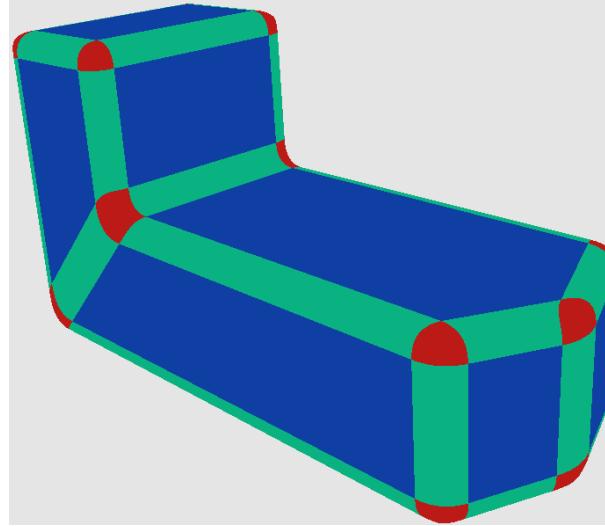
$$\begin{aligned} T &= 0.25(X + H_1 + P + H_2) \\ T_{12} &= 0.25(C + Q_1 + Q_2 + X) \\ T_1 &= 0.25(Q_1 + E_1 + H_1 + X) \\ T_2 &= 0.25(Q_2 + E_2 + H_2 + X) \end{aligned} \quad (3.3)$$

Könnyen belátható, hogy a fent kiszámított tartópontok másodfokú görbéket és másodfokú keresztrányú deriváltakat fognak előállítani. Ennek a magyarázatára egy későbbi fejezetben fogok kiteríni. Az algoritmus előnye, hogy egyszerű, a generált objektum alacsony fokszámú felületekből fog összeállni. A 3.6. ábrán generált objektum 5-oldalú lapjánál az új éleket alkotó görbék (él blend-ek) nem párhuzamosak az eredeti kontrollepoliéder megfelelő éleivel. Mivel az X-csúcs pontjai a négyzetek pontjainak átlagolásából, nem pedig a kapcsolódó élek geometriája alapján számítódnak.



3.7. ábra. Másodfokú lekerekítés(tesztprogram)(a), X-csúcs kontrollpontjai(b)

2. szabályrendszer (harmadfokú lekerekítés)

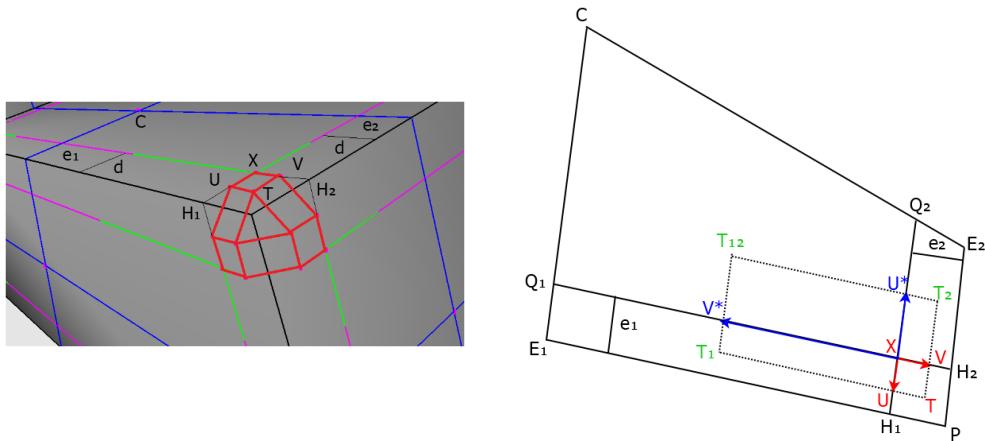


3.8. ábra. Harmadfokú lekerekítés

A harmadfokú lekerekítésnél a lekerekített élek párhuzamosak az eredeti kontrollpoliéder megfelelő éleivel (3.8. ábra). Ezt úgy tudjuk megtenni, ha az algoritmus explicit módon definiálja ezeket a lekerekített éleket e_1, e_2 . Ezek az élek az úgynévezett „offset” egyenesek mentén fognak futni (3.9. ábra lila színnel jelölt egyenesek), amelyeket a megfelelő élekre vetített merőleges irányban d nagysággal való eltolással lehet megkapni. Ezt a d értéket az igényeknek megfelelően tetszőlegesen meg lehet választani a kontrollpoliéder minden éléhe. (A tesztprogramban ez a d érték az „alpha” nevet kapta, amelynek értéke az egyszerűség kedvéért minden érre állandó (lásd 6.2. fejezet)). Az offset élek e_1, e_2 metszeteként előáll az X-csúcs X pontja. Az U, V tangenspontokat megkaphatjuk a megfelelő H_1, H_2 pontok és az X pontot összekötő egyenes felezéséből, pontosan úgy, ahogyan a másodfokú illesztésnél. A twist T pont számítása különbözik az előző megközelítésnél alkalmazotttól, hiszen itt nem a négyszög pontjainak átlagolásával számoljuk a twist pontokat T, T_1, T_2, T_{12} , hanem az X pont és a két megfelelő érintő irányába vett vektorok eltolásával. Az egyenesek egyenletét az egyenesek irányából (érintő irány) és az egyenes egy adott pontjából számítottam. A metszéspont kiszámítható a két egyenes egyenletéből alkotott lineáris egyenletrendszer megoldásával. U^*, V^* és U, V pontok meghatározásához szükség van még a Q_1, Q_2 és H_1, H_2 segédpontok előállítására, amely szintén különbözik az eddig látott másodfokú lekerekítéstől. A $Q_i(i = 1, 2)$ pontokat két-két egyenes határozzák meg: $e_i(i = 1, 2)$ offset egyenes és a C és a megfelelő $E_i(i = 1, 2)$ pontot összekötő egyenes. A $H_i(i = 1, 2)$ pontok előállítása is hasonló: $e_i(i = 1, 2)$ offset egyenest kell metszeni az $E_j(j = 1, 2)$ P pontot összekötő egyenessel ahol $j \neq i$. A kényszeregyenletek közül azokat emeltem ki, amelyek különböznek a másodfokú lekerekítéstől.

Az X-csúcs twist pontjai:

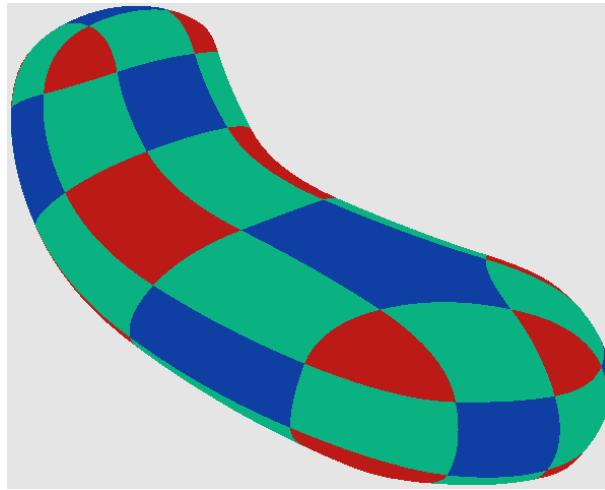
$$\begin{aligned}
 T &= X + \overrightarrow{XU} + \overrightarrow{XV} \\
 T_{12} &= X + \overrightarrow{XU^*} + \overrightarrow{XV^*} \\
 T_1 &= X + \overrightarrow{XU} + \overrightarrow{XV^*} \\
 T_2 &= X + \overrightarrow{XU^*} + \overrightarrow{XV}
 \end{aligned} \tag{3.4}$$



3.9. ábra. Harmadfokú lekerekítés(tesztprogram)(a), az X-csúcs kontrollpontjai(b)

A szabályrendszer harmadfokú görbéket és keresztdériváltakat fog generálni, a görbekre illeszkedő felületek fokszáma három vagy négy lesz. A másodfokú lekerekítéshez hasonlóan a felületek fokszáma alacsony marad, de a poliéder lekerekített élei párhuzamosan futnak az eredeti kontrollpoliéder éleivel.

3. szabályrendszer (simítás)



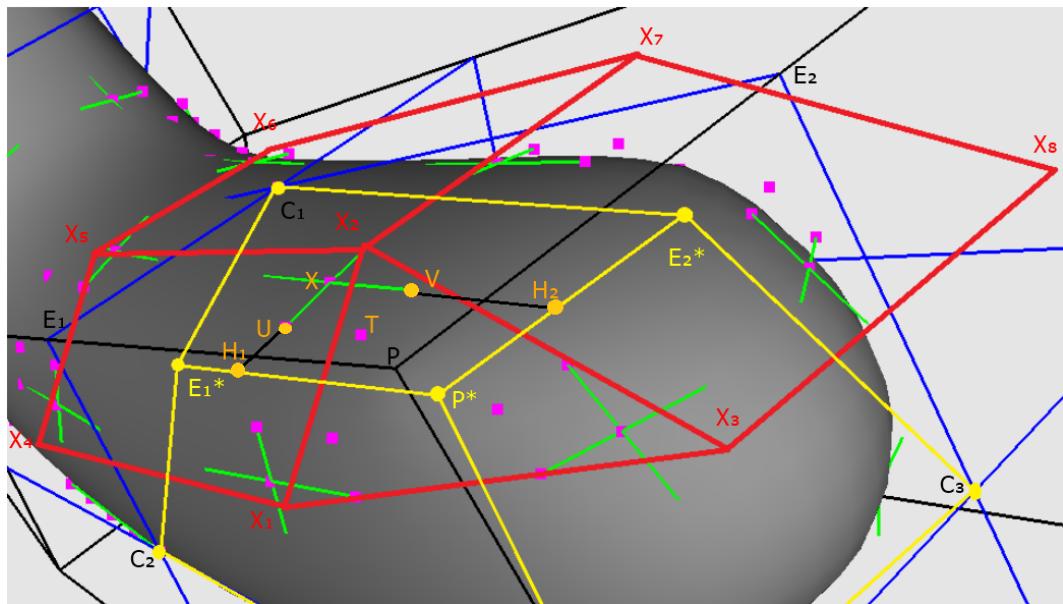
3.10. ábra. Simítás

A simítási szabálytól nem várjuk el, hogy megtartsa az eredeti kontrollpoliéder bármely geometriai elemét. Az előző szabályrendszer megőrizik az eredeti kontrollpoliéder lapjait, és minden esetben azokat zsugorított formában a globális struktúrában felhasználták. A simításnál ezt a megkötést feloldjuk, és lehetővé tesszük, hogy ezek a belső lapok is görbülték legyenek az éleknek, csúcsoknak valamelyen „letörésével”. Az előző konstrukciók olyan négyszögek előállításán alapultak, amelyek részei voltak az úgynevezett QC objektumnak. Simításnál az X-csúcsot egy új, alternatív négyszögöből fogjuk számítani (3.11. ábra sárga színnel jelölt egyenesekkel határolt négyszögek). A pontok E_1^* , E_2^* , P^* jelölésének szisztemája az eredeti kontrollpoliéder pontjaiéhoz E_1 , E_2 , P hasonló, elhelyezkedésük azonban meg fog változni. Ezeket a pontokat a piros színnel jelölt kontrollhálón alkalmazott másodfokú lekerekítés által előállt X-csúcsok $X_i(i = 1, \dots, 8)$ X pontjainak

átlagolásából kaphatjuk meg. Az új kontrollháló pontjai (3.11. ábrán sárga színnel jelölt pontok) előállíthatóak az alábbi képletek szerint:

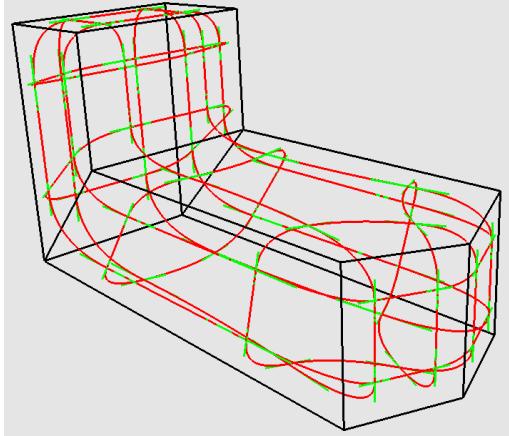
$$\begin{aligned} P^* &= \frac{X_1 + X_2 + X_3}{3} \\ E_1^* &= \frac{X_1 + X_4 + X_5 + X_2}{4} \\ E_2^* &= \frac{X_3 + X_2 + X_7 + X_8}{4} \end{aligned} \quad (3.5)$$

Az új, felosztott négyszögek meghatározzák a tényleges X-csúcsokat (3.11. ábrán zöld színnel jelölt objektum) és azok kontrollpontjait (3.11. narancssárga színnel jelölt pontok) az eddig konstrukciók bármelyikével (másodfokú-, harmadfokú lekerekítés). A tesztprogramban másodfokú lekerekítéssel számoltam az X-csúcsokat.



3.11. ábra. Simítás (tesztprogram)

3.5. Határolgóörbék illesztése

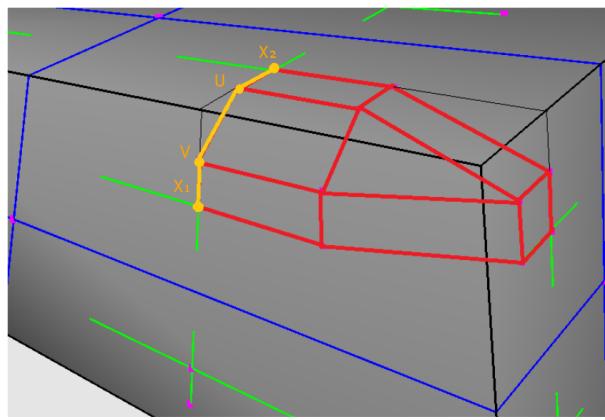


3.12. ábra. Illeszkedő görbehálózat

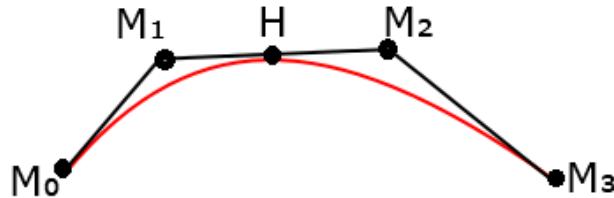
Az előző fejezetben előállított X-csúcs tartópontjai alapján határoló görbéket tudunk illeszteni ezen tartópontokat összekötve két szomszédos X-csúccsal. Ezek a görbek a kiválasztott konstrukciós szabálytól függően lehetnek másodfokúak és harmadfokúak is. Ahhoz, hogy minden esetet is kezelni tudjuk egy egységes adatstruktúrát alkalmazunk, az ún. *hármas segédstruktúrát*, amit az első alfejezetben fogunk bemutatni. A későbbi fejezetben ismertetem, hogy ezekre a kontrollpontokra hogyan lehet a Bézier-görbe kontrollpontjait meghatározni, amiből majd a tényleges határoló görbe fog kialakulni. Az 3.13. ábrán látható, hogy két szomszédos X-csúcs tartópontjai X_1, V, U, X_2 alkotják a hármas segédstruktúra kontrollpontjait.

3.5.1. Hármas segédstruktúra

A hármas segédstruktúra előnye, hogy egységesen lehet használni minden másodfokú, minden harmadfokú esetben. Legyen $M_i (i = 0, \dots, 3)$ a hármas segédstruktúra négy kontrollpontja (3.14. ábra) és három húrja (pl.: $M_1 - M_2$). Egyik fontos tulajdonsága, hogy az érintőgörbék a végpontokban a szélső húrokhoz simulnak (3.14. ábra piros színnel jelölve). Érdekes tulajdonsága még, hogy a középső húr felezőpontja H azonos a görbe parametrikus felezőpontjával.



3.13. ábra. Szomszédos X-csúcsok által meghatározott hármas segédstruktúra (narancssárga színnel)



3.14. ábra. Hármas segédstruktúra

3.5.2. Bézier-görbék illesztése

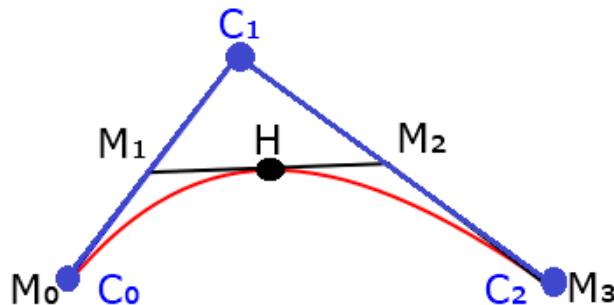
A szomszédos X-csúcsok összekötését Bézier-görbékkel végeztem el, ehhez szükséges előállítanunk a görbét meghatározó kontrollpontokat, majd meg kell adni az egyenletet, ami meghatározza a görbe pontjait a kontrollpontok alapján.

3.5.2.1. Kontrollpontok meghatározása

A Bézier-görbék kontrollpontjait a hármas segédstruktúrából a megválasztott konstrukciós szabálytól függően kell előállítani. A továbbiakban jelöljük az n -ed fokú, $n + 1$ kontrollpontból álló Bézier-görbe kontrollpontjait $C_i(i = 0, \dots, n)$ -vel. A hármas segédstruktúra kontrollpontjai továbbra is: $M_j(j = 0, \dots, 3)$.

Másodfokú Bézier-görbe meghatározásához három kontrollpontra lesz szükség C_0, C_1, C_2 . A C_0 pont megegyezik, M_0 -val, ugyanígy C_2 M_3 -val. A C_1 kontrollpontot a 3.6 kényszeregyenletek valamelyikével lehet meghatározni: a szélső húrok kétszeresével való eltolással.

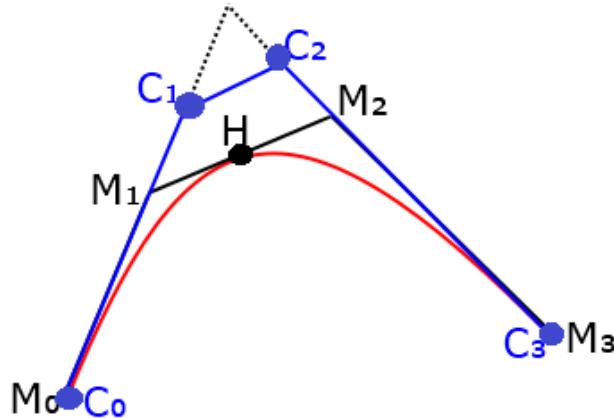
$$\begin{aligned} |C_1 - C_0| &= 2|M_1 - M_0| \\ |C_1 - C_2| &= 2|M_2 - M_3| \end{aligned} \tag{3.6}$$



3.15. ábra. Másodfokú Bézier-görbe kontrollpontjai

Harmadfokú Bézier-görbe meghatározásához négy kontrollpontra lesz szükség C_0, C_1, C_2, C_3 . A C_0 pont megegyezik, M_0 -val, ugyanígy C_3 M_3 -val. Teljesülne kell a harmadfokú lekerekítéshez a 3.7 kényszeregyenleteknek, amelyekből ki lehet számolni a C_1 illetve a C_2 kontrollpontokat

$$\begin{aligned}|C_1 - C_0| &= \frac{4|M_1 - M_0|}{3} \\ |C_2 - C_3| &= \frac{4|M_2 - M_3|}{3}\end{aligned}\tag{3.7}$$



3.16. ábra. Harmadfokú Bézier-görbe kontrollpontjai

3.5.2.2. Bézier-görbék egyenlete

A $C_i (i = 0, \dots, n)$ kontrollpontok ismeretében és egy $t \in [0; 1]$ paraméter segítségével kiszámíthatjuk a görbe pontjait:

$$r(t) = \sum_{i=0}^n C_i B_i^n(t) \quad 0 \leq t \leq 1,\tag{3.8}$$

ahol $B_i^n(t)$ a C_i -hez tartozó súly, azaz az i -edik n -ed fokú, t paraméter értéknél felvett Bernstein-függvény.

3.5.2.3. Bernstein-függvények

A k -adik n -ed fokú Bernstein-függvényt definíció szerint a következő egyenlettel lehet leírni:

$$B_k^n(t) = \binom{n}{k} (1-t)^{n-k} t^k \quad k = 0, 1, \dots, n, \text{ ahol}\tag{3.9}$$

ahol $\binom{n}{k}$ a binomiális együttható, amely a következőképp számolható:

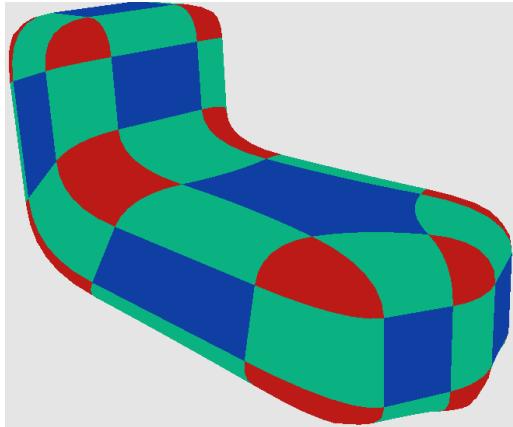
$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \quad k = 0, 1, \dots, n\tag{3.10}$$

A Bézier-görbe interpolálja az első és az utolsó kontrollpontot, a belső kontrollpontokat pedig approximálja. Ismert tulajdonság, hogy a görbe minden a kontrollpontok által kifeszített konvex burok belséjében halad. Ennek feltétele, hogy a Bernstein-függvények

t) pozitívak legyenek és az összegük egy legyen (binomiális tétel alapján).

$$\sum_{i=0}^n B_i^n(t) = \sum_{i=0}^n \binom{n}{k} (1-t)^{n-k} t^k = ((1-t) + t)^n = 1 \quad (3.11)$$

3.6. Felületelemek (patch-ek) illesztése



3.17. ábra. Különböző típusú patch-ek

A határoló görbürekre illesztett patch-eket Bézier-felületek fogják alkotni. Itt is elő fog állni az a probléma, mint a görbek létérehozásakor, hogy a konstrukciós szabálytól létrejöhetnek másodfokú illetve harmadfokú Bézier-felületek is. Ezek egységes kezelésére fogom használni a hármas segédstruktúrát, amit rögtön a fejezet elején ismertetek. A fejezet hátralévő részében különbséget fogok tenni 4-oldalú illetve n -oldalú patch-ek létrehozása között. Attól függően, hogy hányszöd van a patch-nek, különböző algoritmusok fogják előállítani az illesztést a görbehálózatra.

3.6.1. Hármas segédstruktúra illesztése 4-oldalú patch-ekre

Hasonlóan, mint amikor a görbehálózatot hoztam létre, itt is a szomszédos X-csúcsok tartópontjainak fogjuk megfeleltetni a hármas segédstruktúra kontrollpontjait M_{ij} ($i, j = 0, \dots, 3$) annyi különbséggel, hogy itt már szerepet fognak kapni az X-csúcsok twist kontrollpontjai is, ezáltal egy tényleges felületet kapva. A patch típusától függően a hármas segédstruktúra belső pontjait az X-csúcs twist kontrollpontjainak megfelelő kiválasztásával lehet megadni (3.5. ábra). A 3.18. ábrán egy V-patch típusú illeszkedést látunk, a megfelelő twist kontrollpontokkal.

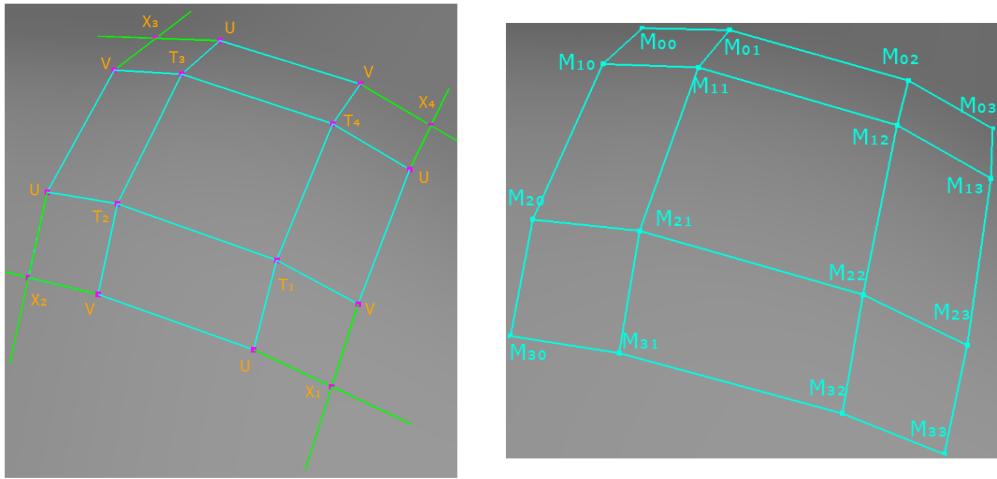
A keresztrányú deriváltak a következőképpen állnak elő a hármas segédstruktúra kontrollpontjaiból: $D_i = M_{i1} - M_{i0}$.

3.6.2. Bézier-felületek illesztése

A szomszédos X-csúcsok által meghatározott hármas segédstruktúra pontjaira Bézier-felületet illesztettem, amelyhez szükséges előállítanunk a felületet meghatározó kontrollpontokat, majd meg kell adni az egyenletet, ami meghatározza a felület pontjait a kontrollpontok alapján.

3.6.2.1. Kontrollpontok meghatározása

A Bézier-felületek kontrollpontjait a hármas segédstruktúrából, a megválasztott konstrukciós szabálytól függően kell előállítani. Legyen u irányban n -ed fokú, $n+1$ kontrollpontból álló, v irányban m -ed fokszámú, $m+1$ kontrollpontból álló Bézier-felület. A dolgozatomban olyan Bézier-felületekkel dolgoztam, amelyeknek a fokszáma megegyezik u, v irányban. A

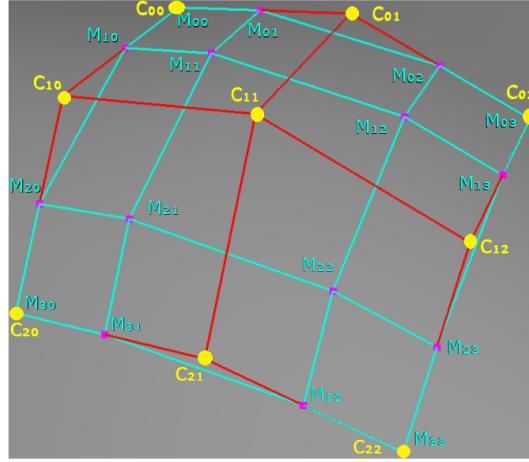


3.18. ábra. Hármas segédstruktúra illeszkedése az X-csúcsokra (a)
és kontrollpontjai (b)

Bézier-felületek kontrollpontjai tehát: $C_{ij}(i, j = 0, \dots, n)$. A hármas segédstruktúra kontrollpontjai továbbra is: $M_{ij}(i, j = 0, \dots, 3)$.

A másodfokú Bézier-felület kilenc kontrollpontból áll, amelyeket a következő képletek segítségével lehet meghatározni:

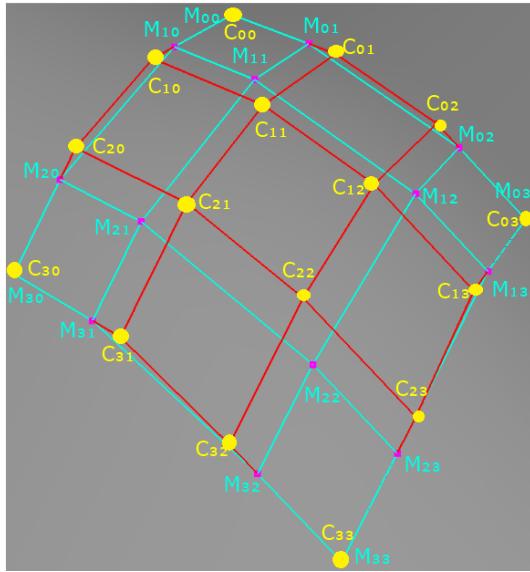
$$\begin{aligned} C_{00} &= M_{00} \\ |C_{01} - C_{00}| &= 2|M_{01} - M_{00}| \\ |C_{10} - C_{00}| &= 2|M_{10} - M_{00}| \\ |C_{11} - C_{00}| &= 2|M_{11} - M_{00}| \end{aligned} \quad (3.12)$$



3.19. ábra. Másodfokú Bézier-felület kontrollpontjai

A harmadfokú Bézier-felület tizenhat kontrollpontból áll, amiket a következő képletek segítségével lehet meghatározni:

$$\begin{aligned}
 C_{00} &= M_{00} \\
 |C_{01} - C_{00}| &= 4 \frac{|M_{01} - M_{00}|}{3} \\
 |C_{10} - C_{00}| &= 4 \frac{|M_{10} - M_{00}|}{3} \\
 |C_{11} - C_{00}| &= 4 \frac{|M_{11} - M_{00}|}{3}
 \end{aligned} \tag{3.13}$$



3.20. ábra. Harmadfokú Bézier-felület kontrollpontjai

3.6.2.2. Bézier-felületek egyenlete

A Bézier-felületek egyértelmű meghatározásához (3.14) ismernünk kell a felület kontrollpontjait C_{ij} és a felület fokszámát (n, m) u és v irányban egyaránt. Ha az egyik irányban a fokszám n , akkor ebben az irányban a felületet $n + 1$ kontrollpont határozza meg.

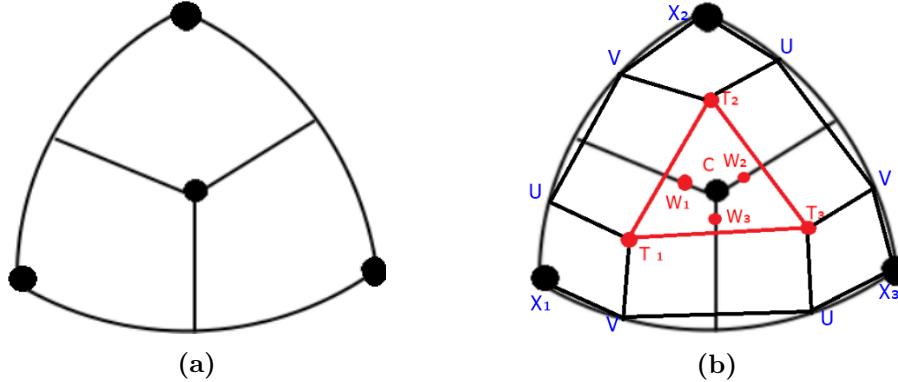
$$s(u, v) = \sum_{i=0}^n \sum_{j=0}^m C_{ij} B_i^n(u) B_j^m(v), \tag{3.14}$$

ahol $B_i^n(u)$ u iránynak megfelelő i -edik n -ed fokú Bernstein-függvény és $B_j^m(v)$ v iránynak megfelelő j -edik m -ed fokú Bernstein-függvény, a fokszámok megegyezése mellett $n = m$.

3.6.3. Hármas segédstruktúra illesztése N-oldalú patch-ekre

Ebben a fejezetben a hármas segédstruktúra n -oldalú lapokra való illesztése kerül bemutatásra. minden E-lap 4-oldalú egyszeres felülettel lesz megvalósítva, viszont az n-oldalú F-, V lapokra n darab 4-oldalú felületeket kell definiálnunk. Ezeket a 4-oldalú felületeket úgynévezett középponti felosztás módszerrel hozzuk létre. Ezt a felosztást olyan görbékkel végezzük, amelyek az adott határoló görbe paraméteres felezési pontjából indulnak és patch középpontjában C -ben végződnek (3.21/a ábra). A C pont kiszámítása majd a belső kontrollpontok meghatározásánál fog szerepelni. Az eddigiekhez hasonlóan itt is hármas segédstruktúrát fogunk illeszteni a megfelelő X-csúcsokra, annyi különbséggel, hogy a

darabszámuk nem egy lesz, hanem annyi ahány oldala van az adott patch-nek. Első lépésként a patch határai mentén szeretnénk beállítani a folytonosságot a szomszédos egymást határoló felületek között (3.6.3.1. fejezet). Utána következik a belső pontok meghatározása (3.21/b ábra), amihez a egy felosztott hármas segédstruktúrát fogunk a továbbiakban használni (3.6.3.2. fejezet). A segédstruktúra létrehozása mellett definiálunk kell a felület érintősíkját és normálvektorát is, ahoz, hogy a belső kontrollpontokat meg lehessen határozni. Legvégül az n-oldalú patch-en belül a Bézier-felületek G^1 kapcsolódását is biztosítani kell (4. fejezet).



3.21. ábra. N-oldalú patch (a) és a kapcsolódó X-csúcsok kontrollpontjai (b)

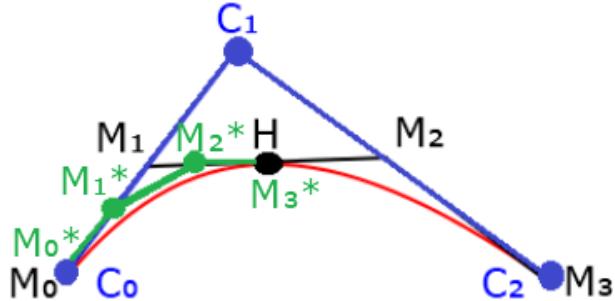
3.6.3.1. Az n oldalú patch-ek határológörbéinek felezése

Ahhoz, hogy simán illeszkedő felületeket kapjunk az algoritmus kimeneteként, biztosítanunk kell, hogy az adott patch-et határoló élek is folytonosan kapcsolódnak az öt határoló patch-ekhez. Ezt egy olyan módszerrel érhetjük el, amely az eredeti hármas segédstruktúra bal, illetve jobb szegmensének pontjait, kiegészítve a paraméteres felezési ponttal lefelezünk. Ez a felezés különböző másodfokú, illetve harmadfokú esetekben, amelyek lépésit ebben a fejezetben ismertetem.

Másodfokú lekerekítés

Másodfokú lekerekítésnél az M_0, M_1, H pontokat összekötő szakaszokat lefelezzük. Az így számított új hármas struktúra pontjai a következők:

$$\begin{aligned} M_0^* &= M_0 \\ M_1^* &= \frac{M_0 + M_1}{2} \\ M_2^* &= \frac{M_1 + H}{2} \\ M_0^* &= H \end{aligned} \tag{3.15}$$

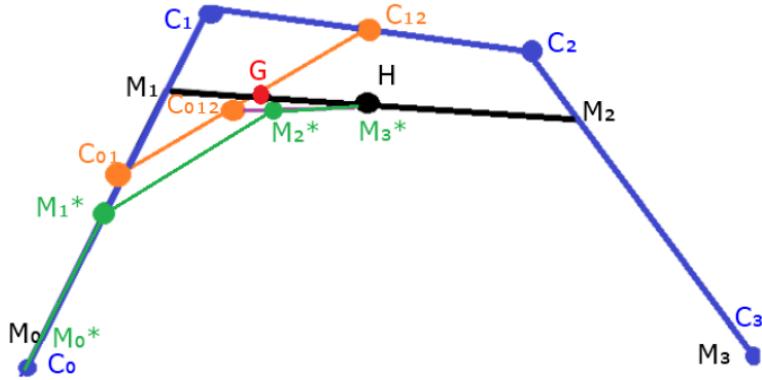


3.22. ábra. Hármas segédstruktúra bal szegmensének felezése másodfokú lekerekítésnél

Harmadfokú lekerekítés

Harmadfokú esetben a generált Bézier-görbühez húzott érintő nem biztos, hogy párhuzamos lesz a hármas segédstruktúra középső húrjával. Ezért a harmadfokú lekerekítésnél a felezés két lépésből fog állni: (1) elvégezzük a másodfokú esethez hasonlóan az M_0, M_1, H pontokat összekötő szakaszok felezését, (2) az így lefelezett pontok közül az utolsó szakaszt felező pont helyét módosítjuk. A 3.23. ábrán ez a módosítandó pont a pirossal jelölt G pont lesz. Meghatározásához (1) egy pontot kell kiszámítani a Bézier struktúrában (narancssárga színnel jelölt rész), (2) majd ahhoz hogy hármas segédstruktúrájú pontot kapjak vissza ki kell számítani a a szükséges pontot egy újabb egyenlet segítségével. A Bézier struktúrában C_{012} pont az a C_{01} és C_{12} pontokat összekötő szakasz felezési pontja. C_{12}, C_1 és C_2 Bézier pontok, C_{01}, C_0 és C_1 Bézier pontokat összekötő szakasz felezési pontjaként áll elő. A keresendő pontunkat M_2^* az előzőleg megkapott C_{012} és a M_3^* pontot összekötő szakasz háromnegyedelésével lehet megkapni. Az új hármas segédstruktúra pontjait létrehozó egyenletek a következők:

$$\begin{aligned}
 M_0^* &= M_0 \\
 M_1^* &= \frac{M_0 + M_1}{2} \\
 C_{012}^* &= \frac{C_{01} + C_{12}}{2} \\
 |M_2^* - M_3^*| &= \frac{3|C_{012} - M_3^*|}{4} \\
 M_0^* &= H
 \end{aligned} \tag{3.16}$$



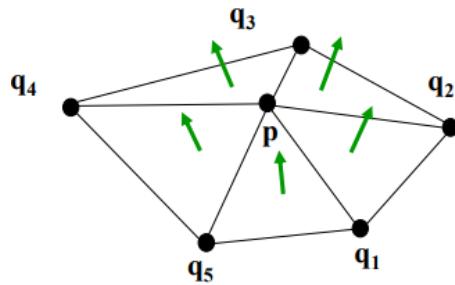
3.23. ábra. Hármas segédstruktúra bal szegmensének felezése harmadfokú lekerekítésnél

3.6.3.2. Belső pontok meghatározása

A határmenti folytonosság beállítása után meghatározzuk a hármas belső struktúra belső pontjait, amelyekre majd Bézier-felületet fogunk illeszteni, mint ahogy tettük 4-oldalú patch-ek esetében. Először a 3.27. ábrának megfelelő hármas segédstruktúra sárga színnel jelölt pontjait (1), majd a zöld színnel jelölt pontokat határozzuk meg (2). Előbbi esetben szükséges lesz meghatározni a patch érintő síkját és normálvektorát is.

Sárga színnel jelölt pontok meghatározása (1) - legjobb lapsík (érintősík)

A C lapközéppontot a twist-vektorok T_i átlagolásából lehet kiszámítani: $C = \frac{\sum_{i=1}^n T_i}{n}$. A középpontnak a meghatározása egyszerű $n = 3$ oldalú patchek-nél, hiszen garantált, hogy a twist-vektorok egy síkban helyezkednek el. Azonban $n = 5$ esetén ez a feltétel már nem minden esetben áll fenn. Ekkor előfordulhat, hogy a középpontból az élfelezőkbe húzott egyenesek sem esnek egy síkba, így az induló tangensek sem. Tehát a csúcsban találkozó görbék folytonosan kapcsolódása sérül. Négynél több oldalú lapkák esetén meg kell határozni egy „legjobb síket”, amit a lapka középpontja és egy becsült normálvektor fog megadni. A lapkát felosztjuk háromszögekre, meghatározzuk ezek normálvektorjait az élek keresztszorzatával:



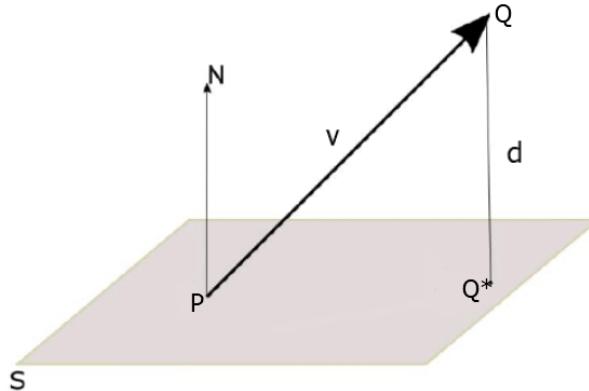
3.24. ábra. Lapka normálvektor becslése

$$n_i = (q_i - p) \times (q_{i+1} - p) \quad (3.17)$$

A legjobb lapka normálvektorát N_C ezen normálvektorok átlaga adja:

$$N_C = \frac{\sum_{i=1}^n n_i}{n} \quad (3.18)$$

Az így kapott normálvektor által meghatározott síkra levetítjük a patch twist-vektorjait $T_i (i = 1, \dots, n)$, ezzel a vetítéssel lesz minimális az eredeti és vetített pontok közti eltérés. A vetítés során vesszük a vetítendő pont előjeles távolságát a síktól, majd ezzel a távolsággal eltoljuk a legjobb normálvektor mentén. Először a normálvektort normalizálni kell, hogy egységnyi hosszú vektort kapunk: $|n| = 1$



3.25. ábra. Pont vetítése

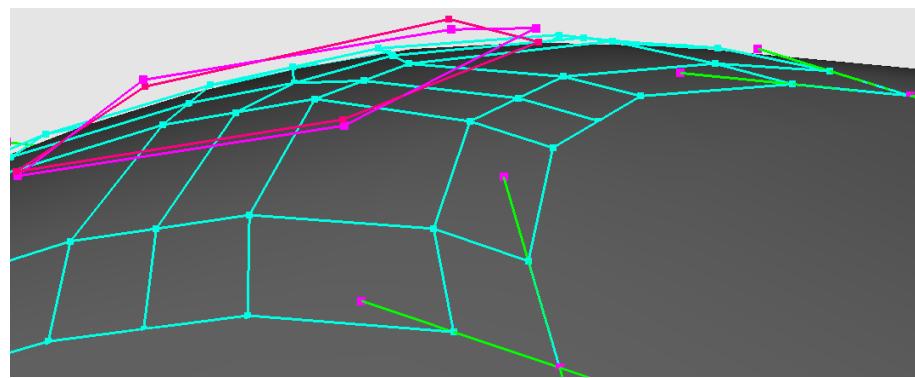
A távolság a normálvektor és a sík pontjából a vetítendő pontba mutató vektor skaláris szorzata:

$$\begin{aligned} v &= Q - P \\ d &= n \cdot v \end{aligned} \quad (3.19)$$

Ezzel az előjeles távolsággal eltolva a Q pontot n irányban megkaphatjuk az új, síkra vetített Q^* pontot:

$$Q^* = Q - dn \quad (3.20)$$

Az 3.26. ábrán egy 5-oldalú patch érintősíkját láthatjuk a twist-vektorok vetítés előtt (halványlila színnel jelölve) és után (rózsaszínnel jelölve):



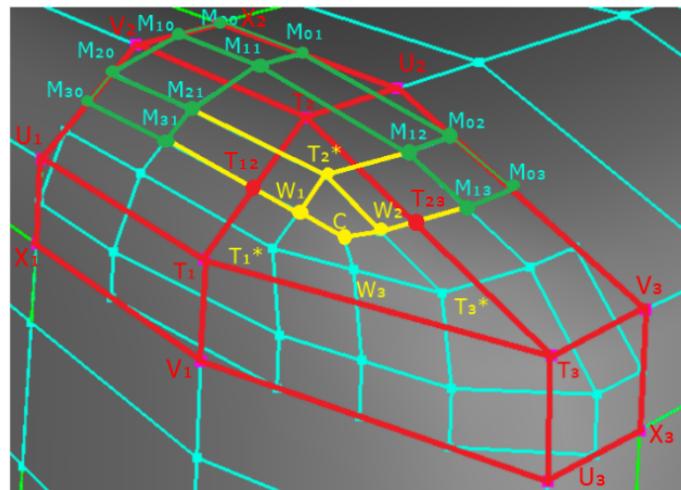
3.26. ábra. 5-oldalú lap legjobb síkja (tesztprogram)

A twist pontok és a középpont meghatározása után kiszámíthatjuk a többi sárga színnel jelölt pontokat is. $T_i^*(i = 1, \dots, n)$ pontokat a megfelelő T_i twist-vektor és a C középpontot összekötő egyenes felezési pontjaként kapjuk meg: $T_i^* = 0.5(T_i + C)$. A $W_i(i = 1, \dots, n)$ kontrollpontok lesznek a felező görbék tangens kontrollpontjai, amelyet az aktuális T_i^* és a megfelelő irányba vett szomszédos T_{i+1}^* pontot összekötő egyenes felezési pontjaként lehet megkapni: $W_i = 0.5(T_i^* + T_{i+1}^*)$. Ezek a kontrollpontok a patch érintősíkjában lesznek, hiszen a C és a T_i pontok is az érintősíkban vannak.

Zöld színnel jelölt pontok meghatározása (2)

A határoló görbéken levő pontokat $M_{i0}(i = 0, \dots, 3)$ és $M_{0j}(j = 0, \dots, 3)$ már az előző fejezetben kiszámítottuk, tehát az $M_{i1}(i = 1, \dots, 3)$ és $M_{1j}(i = 2, \dots, 3)$ pontok meghatározására fogunk koncentrálni a jelen fejezetben.

Másodfokú lekerekítés

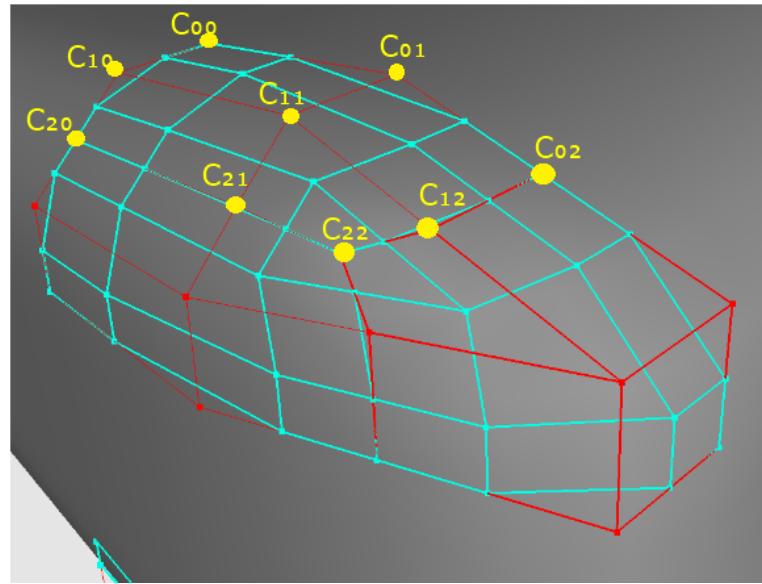


3.27. ábra. Felosztott struktúra létrehozása másodfokú lekerekítésnél

A kérdéses pontokat a pontot körülvevő pontok negyedelési középpontjaként számítottam ki. M_{21} és M_{12} pont meghatározásához szükséges bevezetni a twist kontrollpontokat összekötő egyenesek felezési pontját (T_{12}, T_{23}).

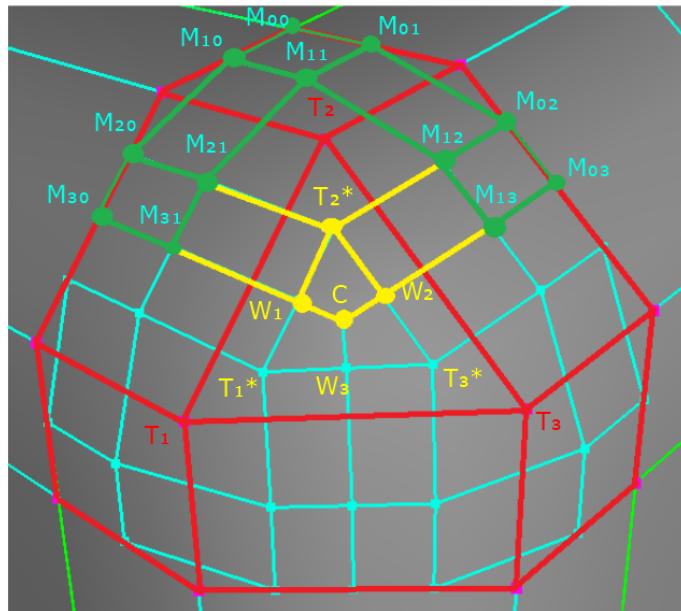
$$\begin{aligned} M_{11} &= \frac{X_2 + U_2 + V_2 + T_2}{4} \\ M_{21} &= \frac{M_{30} + T_{12} + V_2 + T_2}{4} \\ M_{31} &= \frac{T_1 + U_1 + V_2 + T_2}{4} \\ M_{12} &= \frac{M_{03} + U_2 + T_2 + T_{23}}{4} \\ M_{13} &= \frac{T_3 + U_2 + V_3 + T_2}{4} \end{aligned} \quad (3.21)$$

Az így előállított felosztott hármas segédstruktúra pontjainak Bézier-felület kontrollpontokat feleltetünk meg (3.28. ábra), melyek elhelyezkedését a 3.12 egyenlet alapján számítjuk ki.



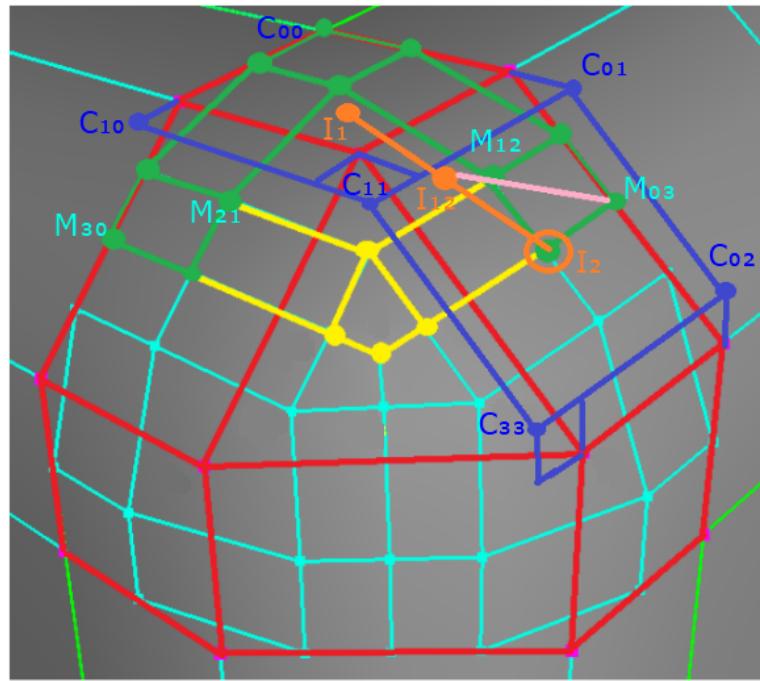
3.28. ábra. Másodfokú Bézier-felület kontrollpontjainak illesztése a hármas segédstruktúra pontjaira

Harmadfokú lekerekítés



3.29. ábra. Felosztott struktúra létrehozása harmadfokú lekerekítésnél

A harmadfokú esetben a felosztott hármas segédstruktúra pontjait hasonló elven fogom meghatározni, mint ahogyan a határoló görbék esetében tettek. Első lépésként alkalmazom a másodfokú felezést, így megkapom az M_{11}, M_{31}, M_{13} kontrollpontokat. Második lépésként az M_{21}, M_{12} pontokat fogom meghatározni az eredeti hármas segédstruktúra kontrollpontjainak megfeleltetett Bézier pontokból. Ezt a felosztást a 3.30. ábrán fogom ismertetni, az M_{12} pont meghatározásával:



3.30. ábra. M_{12} pont meghatározása harmadfokú felezéssel

Első lépésként az eredeti hármas segédstruktúra (piros) kontrollpontjaira meg kell határozni a Bézier pontok helyét (kék) az alábbi képletek felhasználásával: 3.13. Az I_i illetve I_2 pontokat a Bézier pontok negyedelésével lehet előállítani:

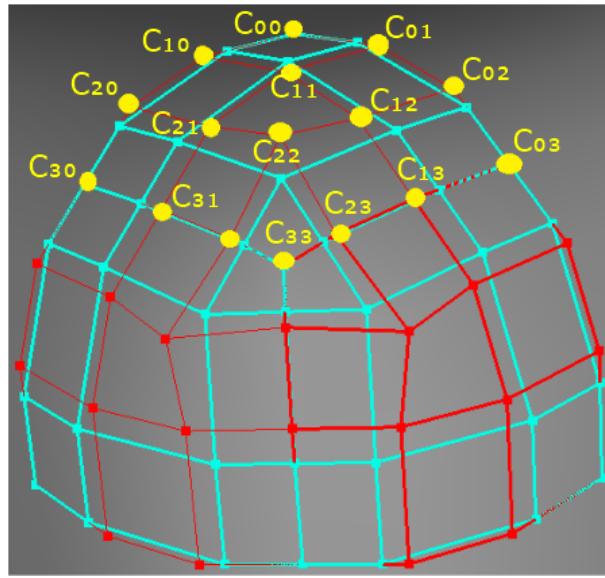
$$\begin{aligned} I_1 &= \frac{C_{10} + C_{00} + C_{11} + C_{01}}{4} \\ I_2 &= \frac{C_{33} + C_{02} + C_{11} + C_{01}}{4} \end{aligned} \quad (3.22)$$

I_{12} pont az előbb kiszámított pontok által alkotott egyenes felezőpontja. Ahhoz, hogy Bézier struktúrából visszatérjünk a hármas segédstruktúrába, ezért az M_{12} pontot az alábbi képlettel tudjuk meghatározni:

$$|M_{12} - M_{03}| = \frac{3|I_{12} - M_{03}|}{4} \quad (3.23)$$

Hasonlóan az M_{21} pont helyzetét is ugyanezzel a módszerrel lehet meghatározni, értelemszerűen a megfelelő segédpontok felhasználása mellett.

Az így előállított felosztott hármas segédstruktúra pontjainak Bézier-felület kontrollpontokat feleltetünk meg (3.31. ábra), melyek elhelyezkedését az 3.13. egyenlet alapján számítjuk ki.

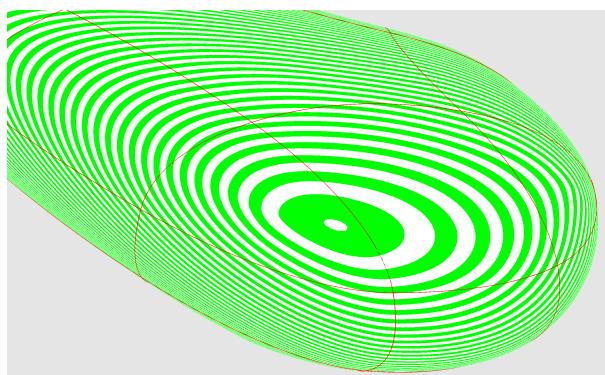


3.31. ábra. Harmadfokú Bézier-felület kontrollpontjainak illesztése a hármas segédstruktúra pontjaira

3.6.3.3. Belső twist kontrollpontok meghatározása

Az n -oldalú felületelemek felosztásakor, a részpatch-ek twist kontrollpontjai a középpont körül nem függetlenek egymástól. Az ezek között fennálló geometriai kényszereket egy összetett vektoregyenletrendszer írja le.

Jelen projektben 3-, 4- és 5-oldalú felületelemekkel foglalkozunk; $n = 4$ -nél nincs szükség belső felosztásra, $n = 3, 5$ -nél viszont elvben meg kellene határozni a fenti, kompatibilis twist-vektorokat. Ehelyett az előzőekben leírt beállítást alkalmazzuk, amely az n -oldalú patch globális hármas segédstruktúrája, és a középpont alapján felezéssel állítja be a twist kontrollpontokat. Ez a megoldás algebrailag nem tökéletes, ellenben gyakorlati szempontból igen jó minőségű G^1 folytonosság érhető el. A szomszédos részpatch-ek normálvektorai között az eltérés általában 1 fok alatt van, ami viszualisan tökéletes kapcsolódást biztosít. A 3.32. ábra egy 3-oldalú V-patch szeletelő vonalait mutatja.



3.32. ábra. V-patch szeletelő vonalai (tesztprogram)

4. fejezet

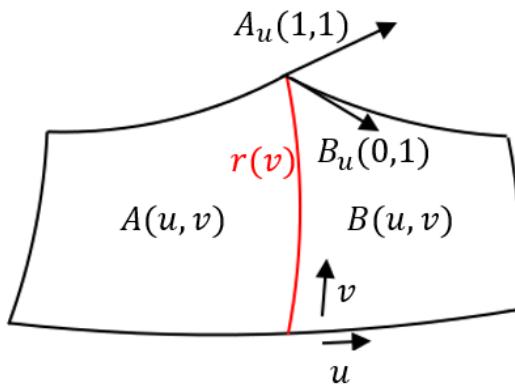
Szomszédos felületek sima összekapcsolódása

Ebben a fejezetben két szomszédos parametrikus felületelem folytonos összekapcsolásának kérdéseit vizsgáljuk. Adott két felület $A(u, v)$ és $B(u, v)$, paraméterezeit a 4.1 ábra mutatja. Feltételezzük, hogy $A(1, v) = B(0, v) = r(v)$, ahol $r(v)$ a két felület közös határgörbéje v szerint paraméterezve.

4.1. Parametrikus és geometriai folytonosság

Parametrikus folytonosságról beszélünk, ha a határgörbe pontjaiban a két felület u szerinti parciális deriváltjai minden v értékre megegyeznek.

C^1 esetben: $A_u(1, v) = B_u(0, v)$, C^2 esetben: $A_{uu}(1, v) = B_{uu}(0, v)$.



4.1. ábra. Bézier-patchek kapcsolódása közös határgörbe mentén

Parametrikus folytonosságot nem lehet értelmezni abban az esetben, ha a felületek kapcsolódó határgörbék törnek, azaz az u -szerinti deriváltak különböznek; például a 4.1. ábrán $A_u(1, 1)$ nem egyenlő $B_u(0, 1)$ -gyel. Ebben az esetben az ún. *geometriai folytonosságot* szokták biztosítani. A G^1 folytonosság azt jelenti, hogy a felületek a határgörbe pontjaiban közös érintőíkkal rendelkeznek, másszóval a bal- illetve jobboldali egység normálvektorok megegyeznek.

$$N_{A0}(1, v) = N_{B0}(0, v) \quad (4.1)$$

A G^2 folytonosság azt jelenti, hogy a felületi görbületek a két oldalt megegyeznek, azaz a bal- illetve jobboldali főgörbületi irányvektorok azonosak:

$$\begin{aligned}\kappa_{1A}(1, v) &= \kappa_{1B}(0, v) \\ \kappa_{2A}(1, v) &= \kappa_{2B}(0, v)\end{aligned}\quad (4.2)$$

4.2. Közös irányblend technika

Jelen poliéderes lekerekítés projektben sima G^1 átmenetet kívánunk biztosítani az egyes felületelemek között. Az előző fejezetben ismertetett mindhárom X-konstrukciós szabályrendszer biztosítja, hogy a szomszédos E-patch-ek és V-patch-ek, valamint a szomszédos E-patch-ek és F-patch-ek simán kapcsolódjanak, hiszen a hármas belső struktúra kontrollvektorjai egymással szemben állnak, így a C^1 parametrikus folytonosság azonnal teljesül (4.2/a. ábra). Amikor felosztunk egy n -oldalú V- vagy F-patch-et a határgörbék felezésével, a 3. fejezetben leírt eljárás biztosítja, hogy a határgörbék mentén az eredeti keresztderivált függvény megmarad, tehát itt nincs folytonossági probléma (4.2/b. ábra). Ellenben a szomszédos belső patch-ek közötti az osztógorbék mentén biztosítanunk kell a sima kapcsolódást (4.2/c. ábra). Jelen szakdolgozat terjedelmi korlátjai nem teszik lehetővé a pontos matematikai konstrukció ismertetését, a részleteket a [16] publikáció tartalmazza.

Ebben a fejezetben csak a konstrukció alapját képező közös irányblend módszer ismertetésére vállalkozunk. Térjünk vissza a 4.1. ábrához. Tételezzük fel, hogy létezik egy közös vektorfüggvény $D(v)$, amely segítségével az A felület keresztderiváltját fel tudjuk írni a következő alakban:

$$A_u(1, v) = D(v) + \alpha(v)r_v(v) \quad (4.3)$$

Hasonlóképpen a másik felületre:

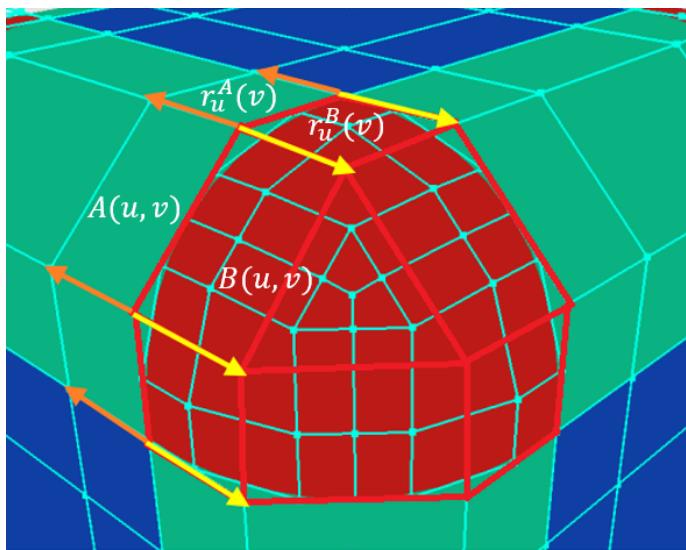
$$B_u(0, v) = D(v) + \beta(v)r_v(v) \quad (4.4)$$

Ebben az esetben

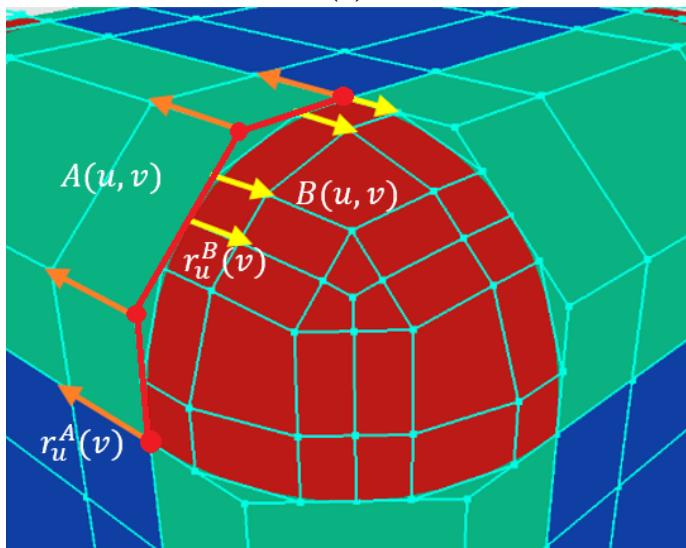
$$\begin{aligned}N_{A0}(1, v) &= A_u(1, v) \times r_v(v) \\ N_{B0}(0, v) &= B_u(0, v) \times r_v(v)\end{aligned}\quad (4.5)$$

így a két normálvektor értéke - $D(v) \times r_v(v)$ megegyezik, és a G^1 érintő sík folytonosság teljesül.

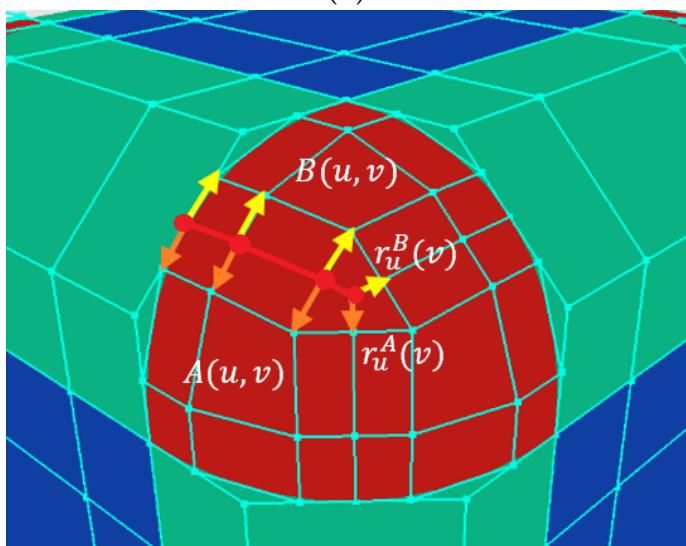
A különböző X-konstrukcióknál a $D(v)$ függvényt és az α és β súlyozó függvényeket másképpen kell beállítani, de végeredményben mindenkor minden hármas segédstruktúrát kapunk a részpatch-ekre ahol a G^1 folytonosság biztosítható. Ezeket a részpatch-eket a másodfokú lekerekítés és simítás konstrukcióknál harmadfokú Bézier patch-ekkel lehet reprezentálni; a harmadfokú lekerekítés konstrukciójánál viszont negyedfokú Bézier patch-ek használatára van szükség.



(a)



(b)



(c)

4.2. ábra. Bézier-felületek kapcsolódása (tesztprogram)

5. fejezet

Tesztrendszer

Az elméletben megalkotott algoritmusokat gyakorlatban is kipróbáltam. Tesztprogramomat egy 3D testek megjelenítésére alkalmas keretrendszerbe [13] integráltam, amelyet C++ programozási nyelven fejlesztettem. A program gondoskodott poliéderek és Bézier-felületek beolvasásáról és megjelenítéséről, így nekem csak az algoritmus megalkotásával kellett foglalkoznom. Ebben a fejezetben bemutatásra kerülnek azok a kritikus problémák is, amelyek felmerültek a fejlesztés során.

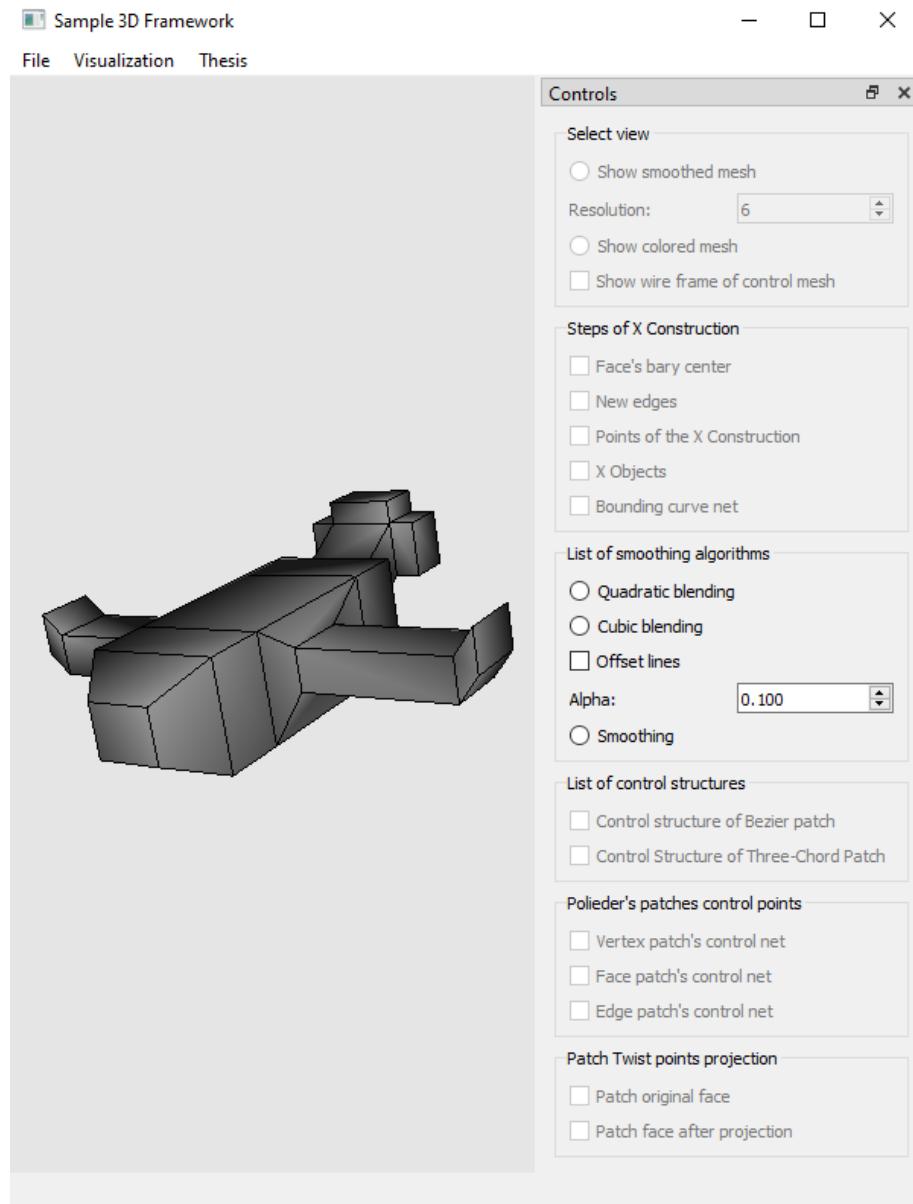
A tesztprogram felhasználó felülete a Qt keresztpalatformos keretrendszerre [12] épül. A felhasználói eseményeket, interakciókat is a Qt könyvtár segédfüggvényeivel valósítottam meg. A felhasználói felület elemeinek (gombokat, checkboxokat, stb.) kialakítása is mind Qt widgetek felhasználásával történt. A különböző grafikai elemeket (kontrollpontok, görbék, stb.) és a geometriai segédváltozókat OpenGL [9] metódusok segítségével rajzoltam ki. A programban az OpenMesh könyvtár [10] nagy segítséget nyújtott a meshstruktúrák feldolgozásához, amely információkból már fel tudtam építeni az algoritmushoz szükséges struktúrákat (X-struktúra). Ezeket az információkat szintén az OpenMesh segítségével beolvasott kontrollpoliéder csúcsain, élein való navigálással tudtam elérni iterátorok segítésével.

A program kimenete egy szabványos Bézier-felületekből álló összetett felület. A patcheket külön-külön háromszögeltem, majd ügyelve ezek helyes orientációjára összefűztem egy egységes adatstruktúrába. Ily módon lehetővé vált, hogy a különböző minőségjellemző mennyiségeket korrekt módon jelenítsem meg.

5.1. A tesztprogram szervezése

A tesztprogramfejlesztés során igyekeztem betartani az objektum-orientált programozás legfőbb elveit, ezáltal a programom átlátható, az osztályok közötti kevés függőség miatt könnyen bővíthető és kicserélhető is egyben. A kiinduló programban az ún. MyViewer osztály volt felelős a grafikai elemek megjelenítéséről és az esetleges számításokért. A fejlesztés kiinduló pontjaként ebből az osztályból származtattam az általam fejlesztett, egyedi funkcionálitással rendelkező a program megjelenítéséről felelős osztályt, a SmoothingViewer-t. A GUI kezeléséért felelős funkciókat elkülönítettem a MyWindow nevű osztályban, ahol minden felhasználói elem létre jön. A különböző felhasználói interakciók feldolgozása is itt található, amelyet a Qt-ben a (connection-slot) elvvel oldottam meg. minden lényegesebb funkciókkal rendelkező és az algoritmusban fontos szerepet kapó objektumnak (X-csúcs, Bézier-felület, V-patch, stb.) létrehoztam egy-egy osztályt. Az OOP-nek megfelelően az osztályokban biztosítottam az egységbázárás elvét, melyet a megfelelő osztályok publikus láthatóságú metódusainak hívásaival oldottam meg. A különböző típusú patch-ek eltárolásához (V-patch, E-patch, F-patch) polimorfizmust alkalmaztam.

5.2. A tesztprogram felhasználói felülete



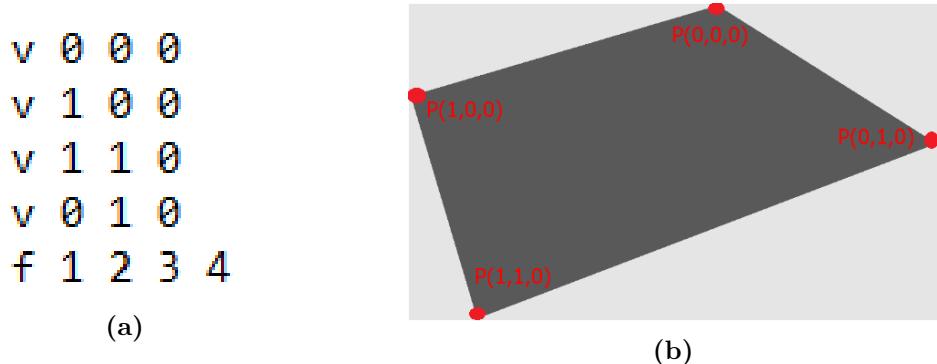
5.1. ábra. A tesztprogram felhasználói felülete

A sima felületet előállító algoritmushoz felhasznált kontrollpoliéder leíró .obj fájlt a felhasználó kiválasztja a fájlrendszerből a tallózó segítségével. A keretrendszer az egyszerű mesh megjelenítése mellett billentyűparancsok segítségével lehetőséget ad a felület drótvázának (wireframe), átlaggörbületének (mean curvature), isophote vonalainak és szeletelésének (slicing) megjelenítésére. Az alapfunkciókat kiegészítettem egy általam létrehozott osztállyal, amelyben implementáltam a szükséges algoritmusokat. A felhasználó ezeket a funkciókat egy „Thesis” nevű Dock Window-ból tudja elérni. A fejlesztés során figyeltem arra, hogy az algoritmus lépéseinek megtékintésekor a felhasználónak van-e lehetősége az adott parancsot végrehajtani az algoritmus adott állapotában. Ha a felhasználó nem rendelkezik azzal a lehetőséggel, hogy végre hajthasson egy lépését az algoritmusnak, akkor az adott elemen látszik, hogy nem reagál a felhasználói interakciókra. Ez látszik a 5.1. ábrán is: hogyha nincsen kiválasztva, hogy milyen struktúrát akarunk megjeleníteni, ak-

kor értelmetlen az algoritmus szempontjából, hogy milyen típusú patch kontrollhálóját szeretnénk látni. Az algoritmus lényegi lépését, az X-konstrukció létrehozását a megfelelő konstrukciós szabályok kiválasztásával lehet elvégezni. Harmadfokú lekerekítés esetében meg lehet jeleníteni az eltolt egyeneseket (offset), és dinamikusan meg lehet választani ezen egyenesek távolságát a kontrollpoliéder adott élétől (alpha). Az X-konstrukció létrehozásért felelős lépéseket check-box-ok ki/be kapcsolatával bármelyik konstrukciós szabály végrehajtása után el lehet végezni. Az előállított simán illeszkedő objektumot a felhasználó kétféleképpen is megjelenítheti: (1) felületek, amelyek a patch típusának megfelelően vannak színezve, (2) szürke felület, amelyen el lehet végezni a felület folytonosságát ellenőrző funkciókat (isophote, slicing, átlaggörbület térkép). A Bézier-felületek felbontásának változtatásával igénynek megfelelően lehet részletes, művész szempontból érdekes objektumot létrehozni. A felhasználónak lehetősége van még a patch-eket alkotó hármas belső struktúrák kontrollhalójának, illetve a Bézie-pontok által alkotott háló megjelenítésére különböző színekkel. 5-oldalú lapok esetében érdekes funkció lehet a patch-et alkotó-twist vektorok vetítés előtti és utáni állapotának megjelenítése is.

5.3. Poliéder beolvasása

A program különböző típusú fájlok beolvasását támogatja, én csak úgynevezett .obj kiterjesztésű fájlokkal dolgoztam a tesztelés során. Egy ilyen fájl tartalmazza a poliéder kontrollpontjainak koordinátahármasait, illetve annak lapjait. Egy lap megadásához az általa tartalmazott csúcsok sorban elfoglalt indexük alapján kell felsorolni. Az .obj fájlok előállítására a Blender nevű programot [2] használtam fel.



5.2. ábra. .obj kiterjesztésű fájl példa (a) és a fájl alapján kapott négyzet (b)

5.4. Az X-struktúra létrehozása

Az X-struktúra létrehozásához az eredeti kontrollpoliéder kontrollpontjai alapján létrehoztam az X-csúcsokat. Létezik egy kontrollpoliéder beolvasó metódusom, amelyben három iterációban dolgozom fel a szükséges információkat. Létrehozok a poliéder minden egyes lapjának sarokpontjaira egy-egy X-csúcsot. Első iterációban poliéder lapjain, második iterációban az aktuális lap csúcsain iterálok végig. Van még egy belső ciklus is ami az aktuális lap aktuális csúcsának szomszédos csúcsairól gyűjt információt. Ezt az OpenMesh navigáló funkcióival az ún. cirkulátorokkal értem el. Az OpenMesh beépített metódusaival számoltam ki a lapközép, élfelező pontokat, amelyekre később a közös él mentén tudok hivatkozni.

5.5. Az X-csúcs tartópontjainak kiszámítása

Az X-csúcs tartópontjait a megválasztott konstrukciós szabálytól függően lehet meghatározni. minden egyes konstrukciós szabályt egy metódusban valósítottam meg. Simító konstrukciós szabály esetében két ilyen metódust hívok meg: másodfokú lekerekítés, majd magát a simításért felelős metódust. Az X-csúcs U, V pontok megfeleltetésének problémáját az okozta, hogy hogyan lehet definiálni, hogy a kiszámolt pontok megfelelnek-e U, V iránynak. Ezért a kiszámolt pontokhoz minden a hozzáillő kontrollpoliéder éleit tárolom el, amelyet majd később amikor szükség lesz a megfelelő tartópontra, akkor a közös él mentén tudok rá hivatkozni.

5.6. Görbék illesztése

Ahhoz, hogy görbéket tudunk illeszteni két szomszédos X-csúcsra, regisztrálni kell az X-csúcsok szomszédosságát. Ehhez létrehoztam a különböző típusú patch objektumokat (E-face, V-face, F-face), amelyek minden egy adott irány mentén (óramutató járásnak megfelelően) tárolják azokat az X-csúcsokat, amelyek az adott felület elemhez tartoznak. A megfelelő U, V tartópontok kiválasztását a hozzájuk eltárolt élreferenciák megkeresésével végeztem. A szomszédos X-csúcsokon való iterálással innentől a tartópontokra már tudtam görbéket illeszteni. A görbe megalkotásánál nem használtam ki, hogy jelenleg csak másodfokú vagy harmadfokú Bézier-görbékkel számolok, a pontokat számító függvény tetszőleges fokszámmal (n) hívható, de ehhez $n + 1$ darab kontrollpontot is meg kell adni. A felbontás finomságának beállítása után kapott görbeponkokra GL_LINES primitíveket illesztettem, amelyek megfelelő felbontás mellett már simán közelítik a görbét.

5.7. Felületelemek illesztése

A létrejött felületmodellt alkotó Bézier-patch-eknek egy egységesen orientált, összefűzött felületet kell alkotniuk ahhoz, hogy a modell minőségét görbület térképek és isophote csíkok segítségével vizsgálni lehessen. Ezek száma értelemszerűen az adott felület oldalainak számától függ. A Bézier-felületek felbontása a programból tetszőlegesen változtatható, ezáltal az igénynek megfelelő finomítású felületet létrehozva. A fejlesztés során fellépett néhány olyan probléma, amelyek megoldását a következő alfejezetekben ismertetek.

5.7.1. STL fájlok használata

A patch-eket alkotó Bézier-felületeknek egy felületet kell alkotniuk ahhoz, hogy a végső simított felületen a görbületet vizsgálni tudjam isophot-tal. Ezt a Bézier-felületek pontjainak való kiírásával majd beolvasásával oldottam meg egy ún. STL-fájlba. AZ STL fájlformátum a felület geometriáját írja le, így színek, textúra koordináták nem szerepelnek benne. A fájl a felület csúcsait (3D-s Descartes-féle koordináták) és a csúcsok egység hosszúságú normál vektorait tárolja el bináris vagy ASCII reprezentációban. Feladatom megoldásakor a bináris kiírást választottam, amelynek formátuma a 5.3. ábrán látható. Bináris kiiírás esetén a létrehozott .stl fájlmérete sokkal kisebb lett. A fájl beolvasását az OpenMesh könyvtár readMesh metódusával végeztem, amely képes STL fájlok feldolgozására. Az STL fájlformátum biztosította azt is, hogy minden pont pontosan egyszer szerepel az elkészült mesh-ben.

```

UINT8[80]      - Header           -     80 bytes
UINT32         - Number of triangles   -      4 bytes

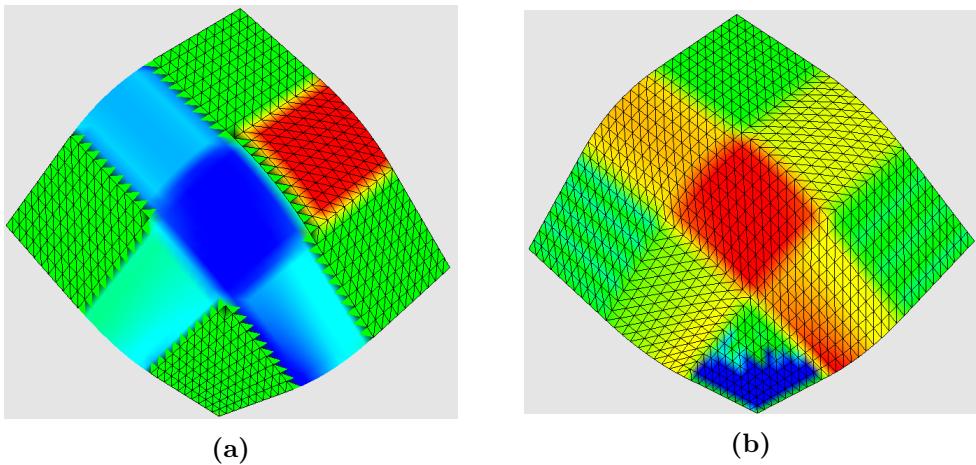
foreach triangle          - 50 bytes:
    REAL32[3] - Normal vector      - 12 bytes
    REAL32[3] - Vertex 1           - 12 bytes
    REAL32[3] - Vertex 2           - 12 bytes
    REAL32[3] - Vertex 3           - 12 bytes
    UINT16    - Attribute byte count - 2 bytes
end

```

5.3. ábra. Bináris STL fájl formátuma

5.7.2. Egységes orientáció beállítása

Fejlesztés során észrevettem, hogy az egyes patch-ek orientációja különböző. A görbületi térkép nem adott helyes eredményt a végső felület görbületi információjáról (5.4. ábra). Ennek oka az volt, hogy a patchek szomszédos X-csúcsait, amelyek egy listában voltak másképp dolgozta fel. Így a Bézier-felületek megalkotásánál is más sorrendben, azaz más orientációs iránytalppal lettek feldolgozva a kontrollpontok. Ezt egy utófeldolgozó algoritmusban rendeztem, ahol az összes patch-en való iterálással megvizsgáltam, hogy egy patch szomszédjainak sorrendje a listában megegyezik-e. Ha nem egyezett meg akkor az érintett patch-nek a listájában az X-csúcsokat felcseréltem, így egységes orientációt kaptam az egész felületre.



5.4. ábra. Görbületi térkép inkonzisztens (a) illetve egységes orientációval (b)

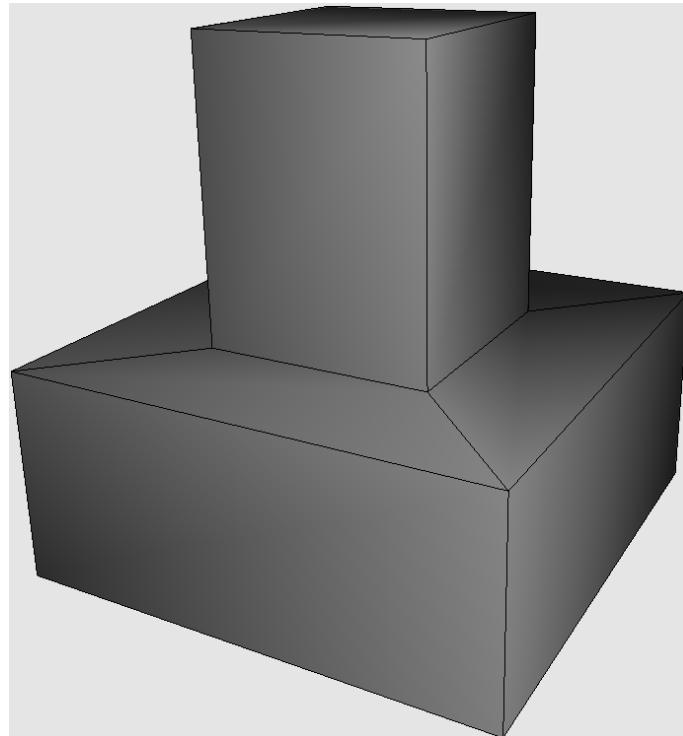
6. fejezet

Teszteredmények

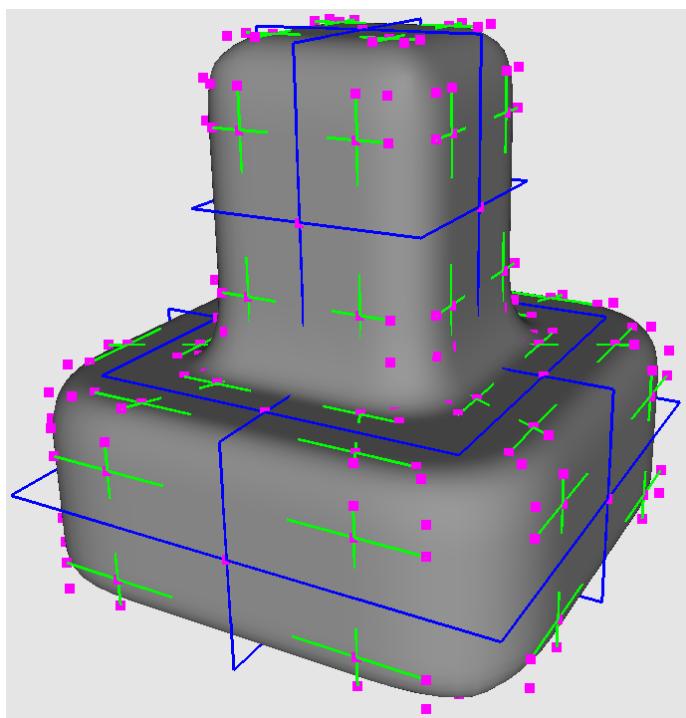
A fejezet első részében egy egyszerű modellen fogom bemutatni a keretrendszer különböző operációit. Ezután pedig bemutatok néhány komplex kontrollpoliéder, amelyek különböző konstrukciós szabályok szerint különböző szabadformájú objektumokat definiálnak.

6.1. Az algoritmus végrehajtása egy egyszerű modellen

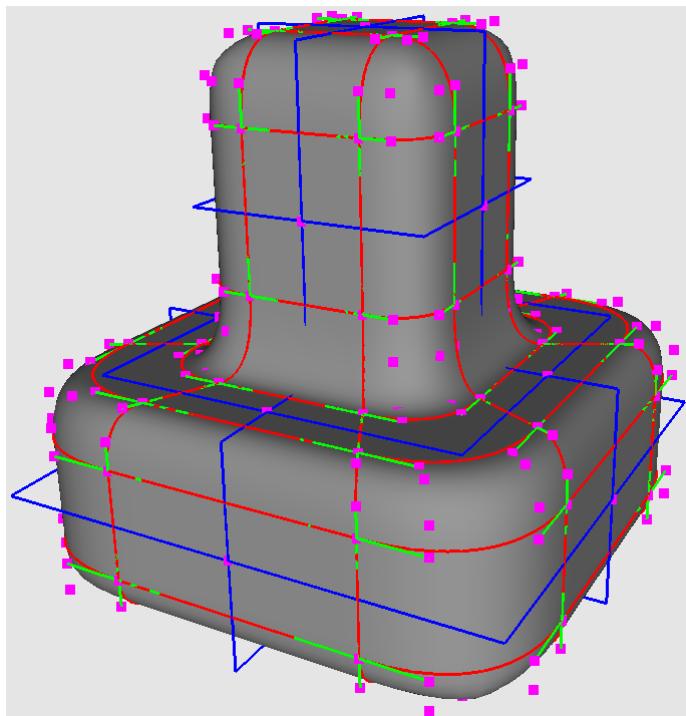
Az X-konstrukciót másodfokú lekerekítéssel végeztem el ezen a tesztpéldán.



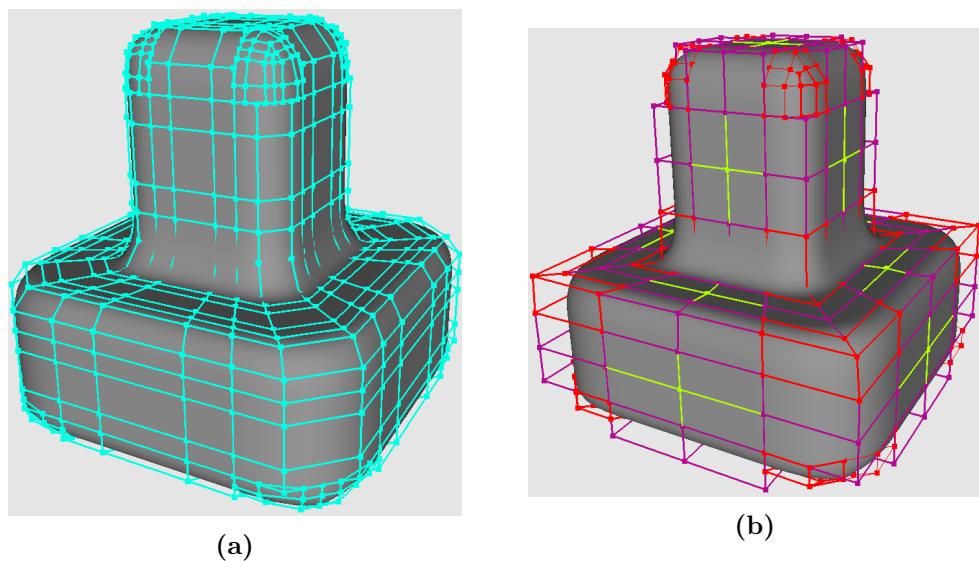
6.1. ábra. Input - konvex lapok



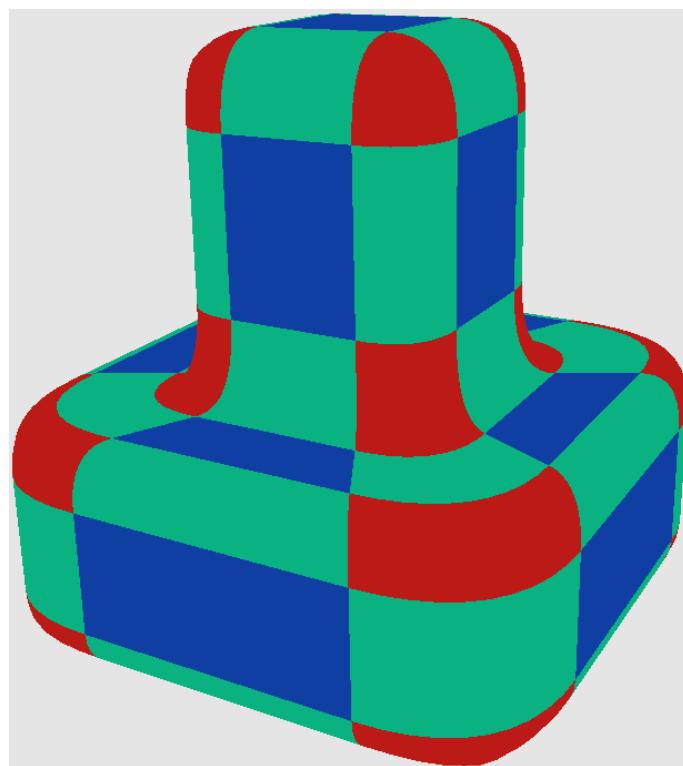
6.2. ábra. Az X-konstrukció lépései



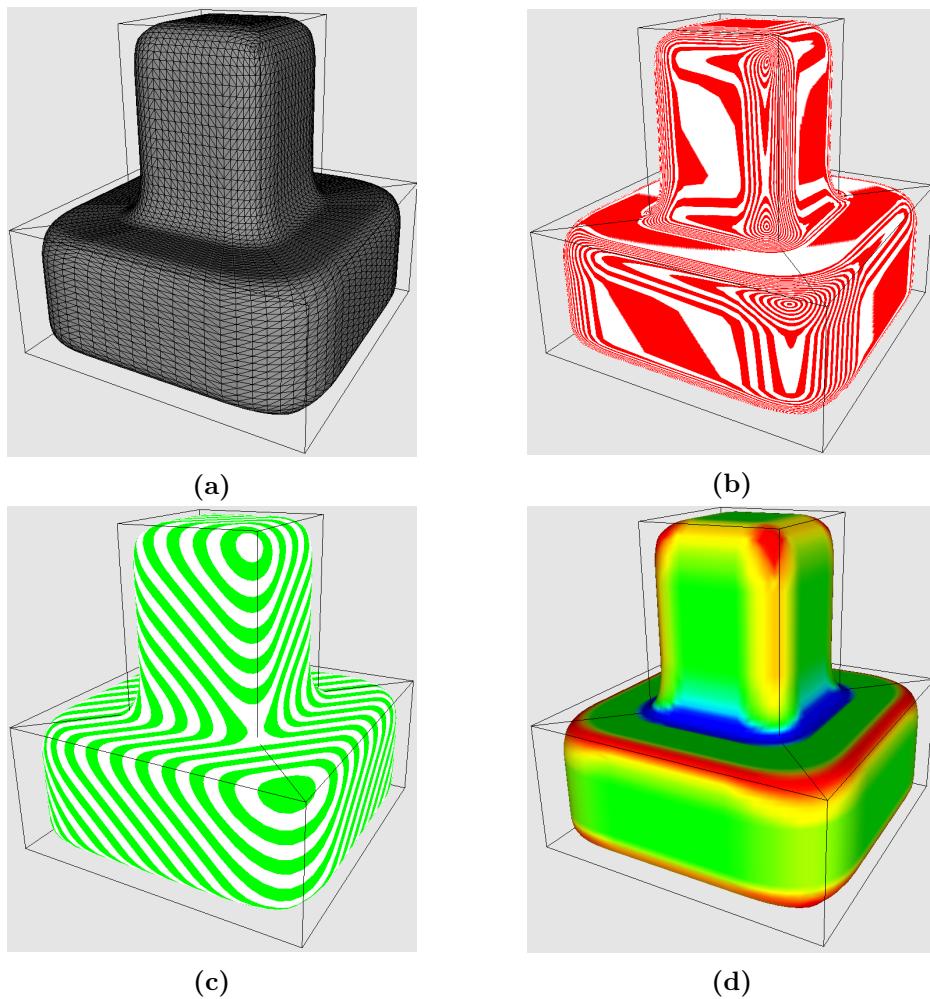
6.3. ábra. Görbehálózat illesztése



6.4. ábra. Hármas segédstruktúra(a) Bézier-felületek illesztése(b)



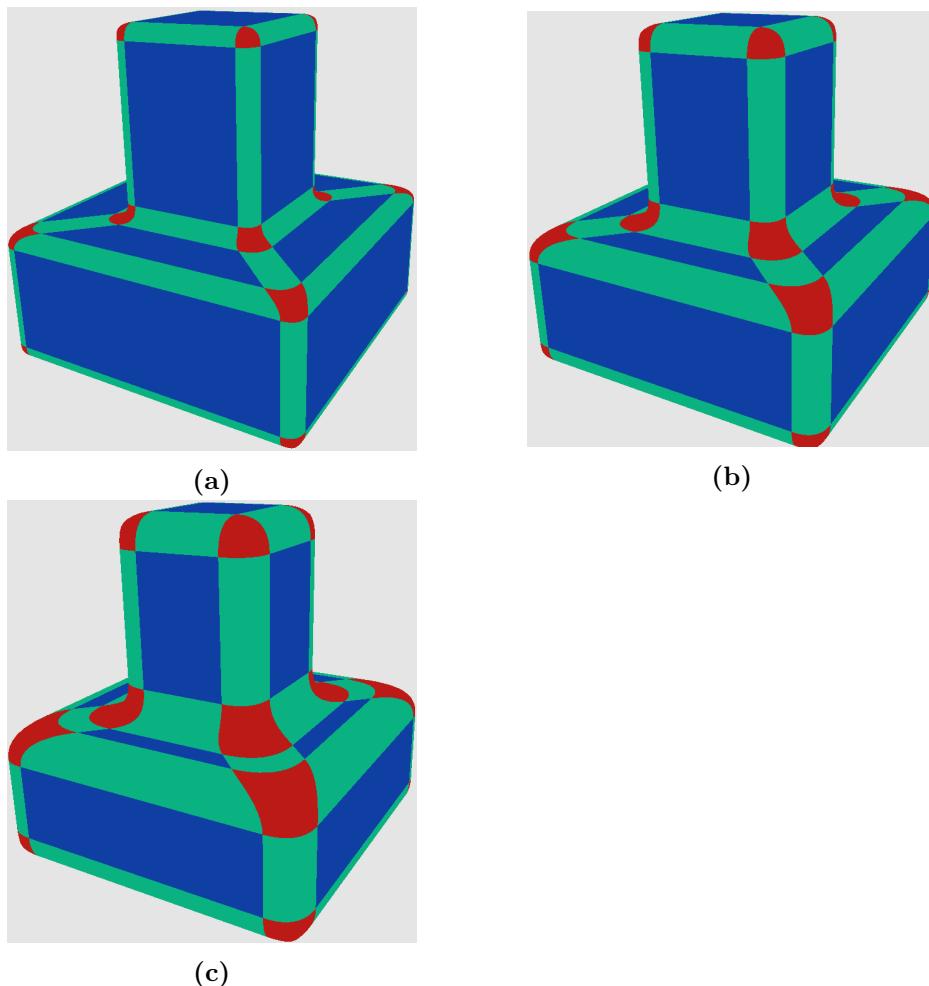
6.5. ábra. Felületek illesztése



6.6. ábra. Bézier-felületek háromszögelve(a) Isophote-ok(b), Szelőtelés(c), Görbülettérkép(d)

6.2. Harmadfokú lekerekítés különböző értékekkel

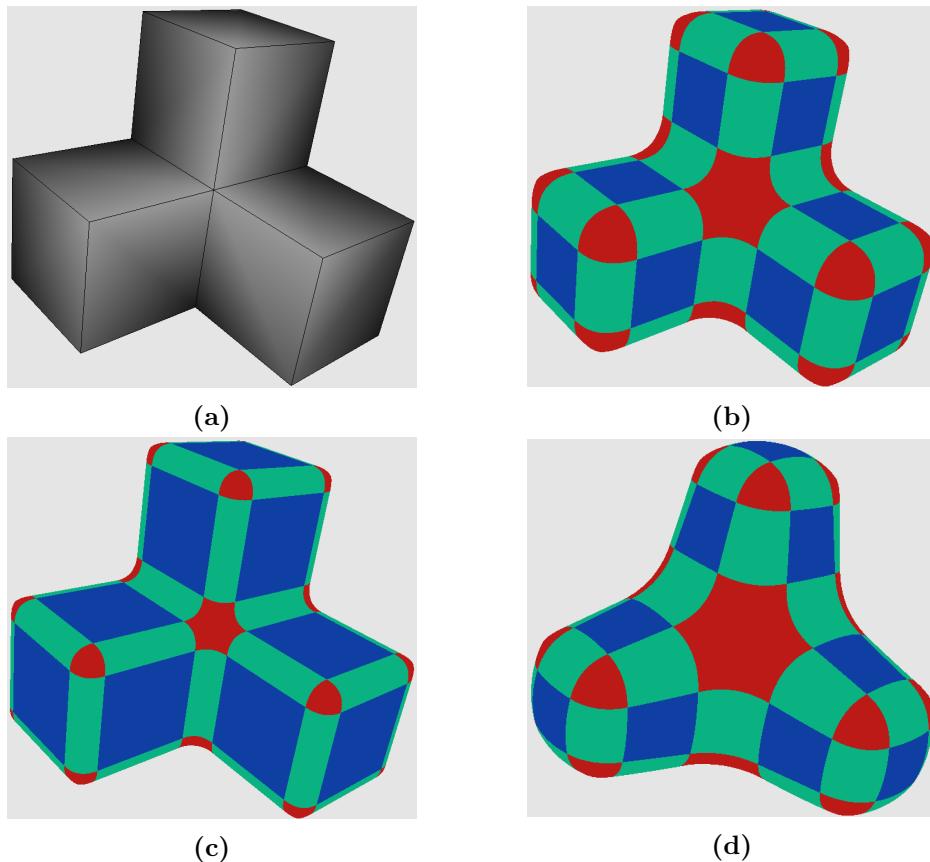
Az alábbi sorozat bemutatja, hogy az offset paraméter különböző értékei, hogyan módosítják a lekerekített poliéder alakját.



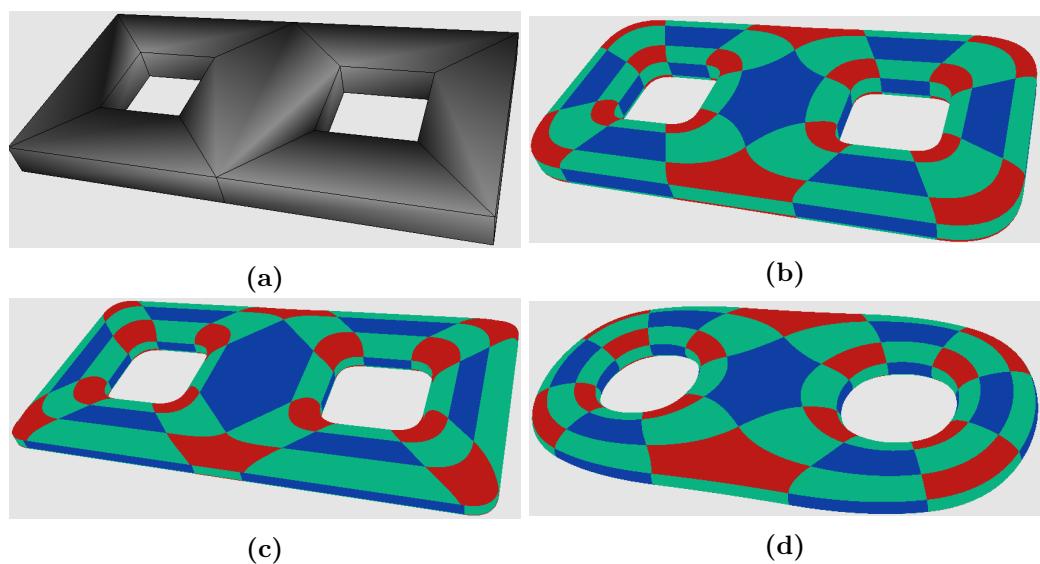
6.7. ábra. Harmadfokú lekerekítés különböző α értékekkel $\alpha = 0.1$ (a), $\alpha = 0.15$ (b), $\alpha = 0.2$ (c)

6.3. Az algoritmus végrehajtása komplex modelleken

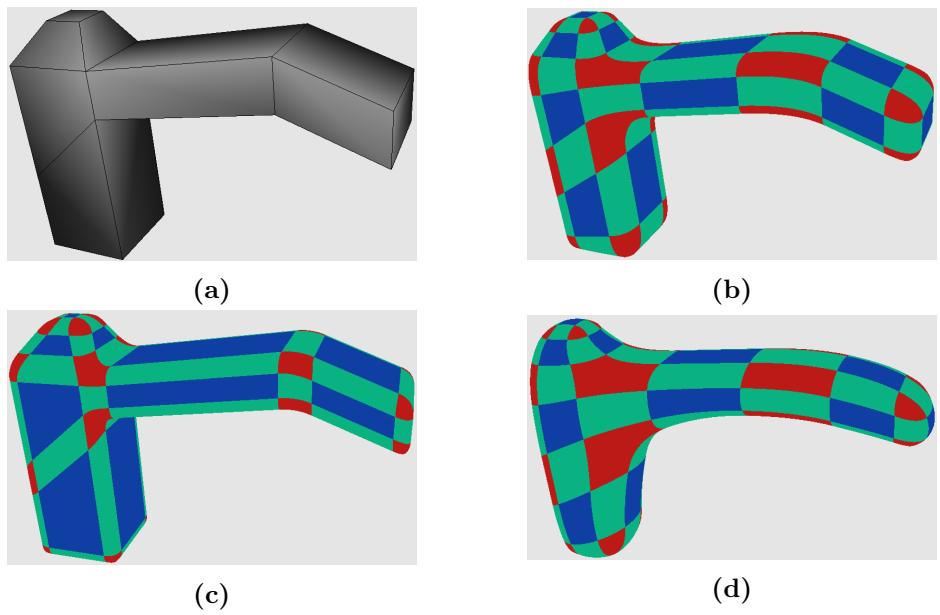
Ebben a fejezetben különböző bonyolultabb és érdekesebb kontrollpoliédereken alkalmaztam a konstrukciós szabályokat. A mintábrák címkézése a következők: kiinduló kontrollpoliéder(a), másodfokú lekerekítés(b), harmadfokú lekerekítés(c), simítás(d).



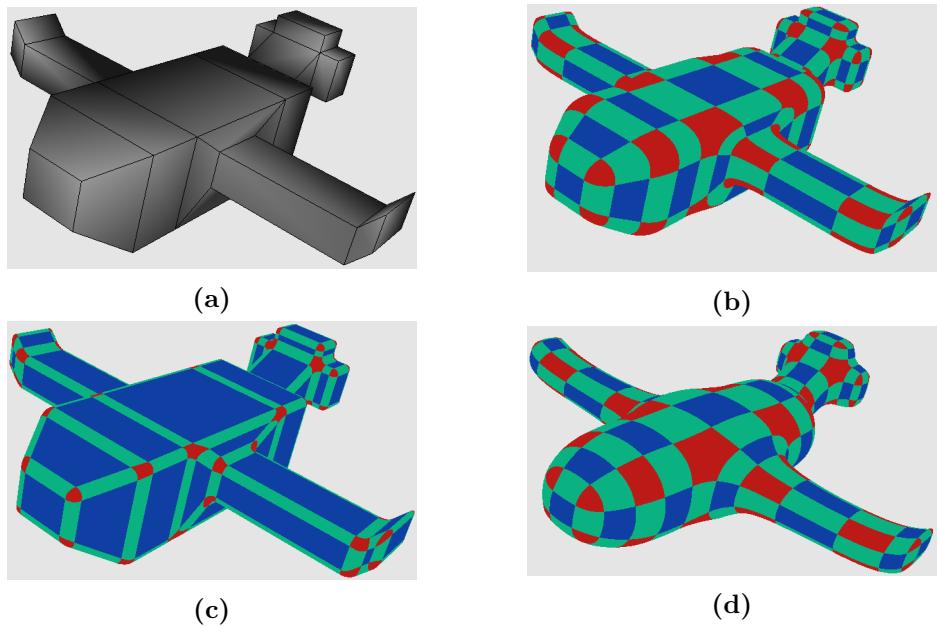
6.8. ábra. Trebol objektum



6.9. ábra. Dupla-tórusz



6.10. ábra. Csap modell



6.11. ábra. Repülő modell

7. fejezet

Összefoglalás és további tervezek

Szakdolgozatomban komplex szabadformájú testek kontrollpoliéderrel történő meghatározásával foglalkoztam. Megismerkedtem az X-konstrukció fogalmával és lépéseiivel, amellyel egy olyan struktúra tartópontjait lehet meghatározni különböző módszerek segítségével, amelyekre egy görbehálót majd Bézier-felületeket lehet illeszteni. Megalkottam a hármas segédstruktúrát, amely egységesen tudja kezelnı a különböző módszerekkel létrejött eltérő fokszámú görbüket és patch-eket.

A fenti tervezést egy 3D-s tesztprogram segítségével valósítottam meg. Az elkészült alkalmazás a kontrollpoliéder beolvasása után a kiválasztott szabályrendszertől függően simán illeszkedő Bézier-felületekből egy új komplex felületet hoz létre. Az eredményként előállt objektumon lehetőség van megjeleníteni az elvégzett lépéseket és a kialakult struktúrák geometriai jellemzőit. A felület minőségéről a görbület térkép, slicing, isophote vonalak nyújtanak támogatást.

Jelen projektben az algoritmus $n = 3, 4, 5$ oldallal rendelkező lapokat tud kezelni. A tesztprogram a középpont körüli twisteket közelítő módon határozza meg; ezt hosszútávon érdemes lecserélni pontos, kompatibilis twist-vektorokkal. Távlati tervezem közé tartozik egyéb konstrukciós szabályok vizsgálata és implementálása.

Köszönetnyilvánítás

Szeretnék köszönetet mondani Dr. Várady Tamásnak, hogy a saját idejét nem sajnálva, minden rendelkezésemre állt, és segítséget nyújtott a szakdolgozat mind formai, mind tartalmi elkészítése terén. Külön köszönettel tartozom Dr. Salvi Péternek, aki rendelkezésemre bocsájtotta a 3D-s tesztprogramot, ami remek kiindulási alapja volt a projektemnek, ezen felül pedig aktívan segített a fejlesztési és matematikai problémák megoldásában.

Irodalomjegyzék

- [1] 3D - számítógépes geometriai tervezés.
URL https://www.iit.bme.hu/system/files/uploads/module_files/3D_SzGA_ShowAll_V4.pdf.
- [2] Blender. URL <https://www.blender.org>.
- [3] E. Catmull – J. Clark: Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-aided design*, 10. évf. (1978) 6. sz., 350–355. p.
- [4] Dia. URL <http://dia-installer.de>.
- [5] D. Doo: A subdivision algorithm for smoothing down irregularly shaped polyhedrons. *Computer Aided Design*, 1978., 157–165. p.
- [6] G. Farin – J. Hoschek – M-S. Kim: *Handbook of Computer Aided Geometric Design*. 2002, Elsevier.
- [7] C. Loop – T. DeRose: Generalized B-spline surfaces of arbitrary topology. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques* (konferenciaanyag). 1990, 347–356. p.
- [8] C. Loop – S. Schaefer: Approximating Catmull-Clark subdivision surfaces with bicubic patches. *ACM Transactions on Graphics (TOG)*, 27. évf. (2008) 1. sz., 1–11. p.
- [9] OpenGL. URL <https://www.opengl.org>.
- [10] OpenMesh. URL https://www.graphics.rwth-aachen.de/media/openmesh_static/Documentations/OpenMesh-Doc-Latest/index.html.
- [11] J. Peters: Smooth free-form surfaces over irregular meshes generalizing quadratic splines. *Computer Aided Geometric Design*, 10. évf. (1993) 3-4. sz., 347–361. p.
- [12] Qt keretrendszer. URL <https://www.qt.io>.
- [13] P. Salvi: Használt keretrendszer.
URL <https://github.com/salvipeter/sample-framework>.
- [14] T. Várady: Négyoldalú felületekből összetett 3D modellek.
URL http://cg.iit.bme.hu/portal/sites/default/files/oktatott-targyak/3d-geometria-2/Quadrilaterals_RMFI_V15.pdf.
- [15] T. Várady – P. Salvi – I. Kovács: Enhancement of a multi-sided Bézier surface representation. *Computer Aided Geometric Design*, 55. évf. (2017), 69–83. p.
- [16] T. Várady – I. A. Stroud: The X construction: bipolynomial patches for smoothing and blending polyhedral objects. *Geometric Modelling Studies, MTA Sztaki*, 1999.