Vizualizációs eljárások kapcsolatok síkbeli ábrázolásához és információk felületen való megjelenítéséhez

Egyetemi doktori (PhD) értekezés

A szerző neve: Papp György

A témavezető neve: Papp Ildikó

DEBRECENI EGYETEM
Természettudományi és Informatikai Doktori Tanács
Informatikai Tudományok Doktori Iskola
Debrecen, 2022.

Ezen értekezést a Debreceni Egyetem Természettudományi és Informatikai Doktori Tanács Informatikai Tudományok Doktori Iskola Diszkrét matematika, adatfeldolgozás és vizualizáció programja keretében készítettem a Debreceni Egyetem műszaki tudományokban doktori (PhD) fokozatának elnyerése céljából. Nyilatkozom arról, hogy a tézisekben leírt eredmények nem képezik más PhD disszertáció részét.

Debrecen, 2022. február 22.

.....
Papp György

Tanúsítom, hogy Papp György doktorjelölt 2016–2022 között a fent megnevezett Doktori Iskola Diszkrét matematika, adatfeldolgozás és vizualizáció programjának keretében irányításommal végezte munkáját. Az értekezésben foglalt eredményekhez a jelölt önálló alkotó tevékenységével meghatározóan hozzájárult. Nyilatkozom továbbá arról, hogy a tézisekben leírt eredmények nem képezik más PhD disszertáció részét.
Az értekezés elfogadását javasolom.

Debrecen, 2022. február 22.

.....
Papp Ildikó

Vizualizációs eljárások kapcsolatok síkbeli ábrázolásához és információk felületen való megjelenítéséhez

Értekezés a doktori (PhD) fokozat megszerzése érdekében
az informatika tudományágban

Írta: Papp György okleveles programtervező informatikus

Készült a Debreceni Egyetem
Informatikai Tudományok doktori iskolája
(*Diszkrét matematika, adatfeldolgozás és vizualizáció* programja)
keretében

Témavezető: Papp Ildikó

Az értekezés bírálói:

Dr. ....
Dr. ....
Dr. ....

A bírálóbizottság:

elnök: Dr. ....
tagok: Dr. ....
Dr. ....
Dr. ....
Dr. ....

Az értekezés védésének időpontja: 2022... ....

Tartalomjegyzék

1. Bevezetés **1**

2. Kapcsolatok körön alapuló vizualizációja **4**
2.1. Szakirodalmi áttekintés 4
2.1.1. Vizualizációs technikánk háttere 5
2.2. TabularVis 9
2.2.1. A diagram átláthatóságának növelése 16
2.2.2. Összehasonlítás 23
2.3. Bézier görbék lecserélésének vizsgálata kúpszeletekkel 25
2.3.1. Kúpszeletek szerkesztése 27
2.3.2. Szalagok kúpszeletekkel történő megadásának vizsgálata 38
2.4. Kapcsolatok vizualizációja körökkel 47

3. QR-kódok 3D nyomtatással alkotott felületekbe ágyazása **48**
3.1. Szakirodalmi áttekintés 48
3.2. QR-kód felületre történő vetítésének meghatározása 50
3.2.1. Paraméteres felület 51
3.2.2. Háromszögelt modell 56
3.3. QR-kód beágyazása 63
3.3.1. Beágyazás mélységének meghatározása GPU támogatással 66
3.3.2. Beágyazási mélység csökkentése minták használatával 70
3.4. Összehasonlítás 76
3.4.1. Paraméteres felület 77
3.4.2. Háromszögelt modell 82

4. Összefoglalás **95**

5. Summary **97**

Köszönetnyilvánítás **99**

Irodalomjegyzék **100**

A TabulrVis **110**
A.1. Keresőalgoritmusok paraméterei 110
A.2. Kapcsolatok vizualizációja merőlegesen metsző körökkel . 110

B Felületbe ágyazott QR-kód **111**
B.1. Paraméteres felület 111
B.2. Háromszögelt modellek 113

C Publikációs lista **115**

1. Bevezetés

Az adatvizualizáció területének egyik legkorábbi problémája a kapcsolatok megjelenítése, amely ma is fontos szerepet tölt be a legtöbb tudományterületen vagy iparágban. Az adatok közötti kapcsolatok vizualizációja egyre elengedhetetlenebb, ezért olyan hatékony megjelenítési megoldásokra van szükségünk, amely megfelel a felhasználók igényeinek és nem csak magát a kapcsolatot, hanem annak típusát vagy tulajdonságát is képes ábrázolni.

A táblázatos forma megfelelő reprezentációt biztosít a kapcsolatokat leíró információk tárolására. Azonban, hiába tudjuk a segítségével egyszerűen eltárolni az információkat, ha az adatok növekedésével nehézkessé válik a táblázat olvashatósága és kezelése. Emiatt nagy szükség van az olyan vizualizációs módszerekre, amelyekkel a felhasználók könnyedén azonosíthatják a táblázat által korábban rejtett információkat és kapcsolatokat, valamint mintázatokat ismerhetnek fel a létrehozott diagramban.

Ennek a megoldására alkottuk meg a körön alapuló vizualizációs technikával dolgozó TabularVis alkalmazásunkat, amelyben új eszközöket mutatunk be a kapcsolatok irányának, értékének valamint az általuk kirajzolt mintázat megjelenítésére. Mivel a diagramban található kapcsolatok közötti metszések zavarók, és csökkentik az ábra átláthatóságát, ezért egy olyan rendezési módszert javaslunk, amely a kapcsolatok legkevesebb metszést generáló elrendezését keresi. Több, lokális kereső és optimalizáló módszert implementálunk, amelyek eredményét összehasonlítva választottuk ki a legjobb teljesítményt elérő algoritmust. Továbbá egy hierarchikus élkötegelésen alapuló módszert is bemutatunk a diagram átláthatóságának növelésére. Ezt követően a kapcsolatok vizualizációjában méltán népszerű Bézier görbének, implicit egyenlettel rendelkező görbékkel való lecserélését tárgyaljuk. Ennek során a vizualizációban

használható másodrendű görbéket adtunk meg, valamint megvizsgáltuk az így kapott kúpszeletek viselkedését. A vizualizációs technikánkat, valamint a Bézier görbék helyettesítésére adott kúpszeleteket az értekezés első részében tárgyaljuk. A megoldásaink bemutatása előtt rövid szakirodalmi áttekintést követően ismertetjük a vizualizációs eljárásunk alapjául szolgáló Circos [26] táblázat vizualizációját, valamint annak problémáit, hogy az olvasó jobban megérthesse a megjelenítésünkhöz javasolt új eszközök mögötti motivációnkat.

Az értekezés második részében elrugaszkodunk a adatok síkbeli, diagramokkal történő prezentálásától, mivel sokszor mutatkozik igény az információk, valamilyen tárgy felületén történő megjelenítésre. A méretbeli korlátozások, valamint a könnyű és gyors olvashatóság érdekében, gyakran csak bizonyos eszközökkel dekódolható kódként kerülnek ezek az információk a felületre. Ez lehet egy címkére kinyomtatott vonalkód is, amit a kiválasztott felületre ragasztanak fel. Azonban, a differenciálgeometria alapvető elméleti korlátai miatt, az ilyen címkék elhelyezése a tárgyak kifejthető felületfoltjaira korlátozódik. Ezáltal elkerülhető a vignetta nagymértékű deformálódása, amely negatívan befolyásolná a kód olvasását. Ha egy tárgy nem rendelkezik ilyen felületfolttal, vagy az nem alkalmas (pl. túl kicsi) a címke elhelyezésére, akkor az információ felületbe történő beágyazása jelenthet megoldást.

Két különböző felülethez ismertetünk módszereket a QR-kód beágyazására. Az elsőben paraméteres felületekbe, a másodikban pedig háromszögelt modellek felületébe ágyazható be a QR-kód. Emellett, a javasolt eljárásaink használatával kiküszöbölhetők a [26, 58] módszerek hiányosságai és problémás esetei. A megoldásaink ismertetése során, először a QR-kód felületre vetítéséhez szükséges irányt, valamint a kód legnagyobb garantáltan beágyazható méretének meghatározását mutatjuk be. Ezt követően ismertetjük a QR-kód beágyazásának menetét, amelyhez a [26,58] munkákkal ellentétben nincs szükség előfeldolgozási lépés végrehajtására. Továbbá, bemutatjuk még a beágyazás mélységének csökkentésére kialakított mintáinkat, illetve a mélységkereső eljárásunkat, amely segítsé-

gével a korábbi munkákban látható futási idők töredéke alatt meghatározhatjuk a QR-kód mélységét. Végül pedig a már meglévő módszerekkel hasonlítjuk össze a megoldásainkat, amelyhez egy rövid szakirodalmi áttekintés keretében ismertetjük az összehasonlításhoz használt meglévő eljárásokat az értekezés második felének bevezetésként.

2. Kapcsolatok körön alapuló vizualizációja

Az értekezés első részében a kapcsolat megjelenítésére adott vizualizációs eljárásunkat, illetve az elemek közötti összeköttetések megalkotásához használt Bézier görbe lecserélése javasolt kúpszeleteket tárgyaljuk. Ehhez először szeretnénk egy rövid szakirodalmi áttekintést nyújtani, amelyen belül ismertetjük a vizualizációs technikánk alapjául szolgáló Circos alkalmazást is.

2.1. Szakirodalmi áttekintés

Sokféle technika létezik a táblázatos formában megadott adatok vizualizációjára, azonban ezek közül kiemelkedik a nemcsak számos vizualizációs alkalmazásban, hanem a gráfrajzolás területén is előszeretettel használt körön alapuló elrendezés. A használatával készített ábra értelmezéséhez szükséges szemmozgások számának csökkentése, a forgás-invariáns tulajdonsága, valamint az elemek olvasási sorrendjének a diagramra gyakorolt hatásának gyengítése csak három előny azokból, amelyeket Chuah és Eick [12] tanulmánya részletesen tárgyal. A különböző tudományágak szakirodalmában számos példát láthatunk a körön alapuló elrendezés használatára. Például van, amelyekben genomokkal [8, 25, 61, 63], hálózatokkal [13,23,24,39], vagy általános adatokkal [6,14,44,46] kapcsolatos vizualizáció elkészítése volt a cél.

Az egyik legnépszerűbb körön alapuló elrendezést használó alkalmazás a Circos [30], amelyet széles körben használnak a genomkutatás, és az ehhez kapcsolódó területeken. Emellett számos olyan alkalmazás létezik (OmicCircos [22], RCircos [70], vagy ggbio [68]), amelyek továbbfejlesztett és felhasználóbarátabb körön alapuló genom vizualizációt biztosítanak.

Két másik alkalmazást, a J-Circos-t [1] és Mizbee-t [41] megvizsgálva pedig azt láthatjuk, hogy a felhasználók gyorsabban és jobban megérthetik az adatokat, ha a körön alapuló elrendezést interaktív adatvizualizációs technikákkal kombináljuk. Azonban a genomi adatokkal dolgozó megjelenítések mellett igény van olyan megoldásokra is, amelyekkel a felhasználók általánosabb adatokhoz megadott kapcsolatokat képesek vizualizálni. Mint például a Circlize [19] (R programozási nyelven írt csomag), Circos Table Viewer [29], InfoVis könyvtár [37], illetve Jen Lowe's vizualizációs megoldása [4].

Bármelyik eddig felsorolt vizualizációs technikát is nézzük, gyorsan zsúfolttá válhat a diagram ha egyszerre sok kapcsolatot akarunk benne megjeleníteni. Ha pedig ez bekövetkezik, akkor nagyban megnehezítheti az elkészült vizualizáció értelmezését. Azonban a kapcsolatok közötti metszések számának redukálásával a diagram zsúfoltságát is csökkenthetjük. A gráfok éleit tekintve már egy jól ismert probléma a metszések számának csökkentése, amelyre Baur és mtsai. [2], valamint Gansner és mtsai. [16] is heurisztikus módszert javasolnak megoldásként. Emellett a Lhuillier és mtsai. [34] által összegyűjtött élcsomagolási technikák hatékony alternatívaként szolgálnak a diagram zsúfoltságának csökkentésére. Egy ezektől eltérő megoldással, külső útvonalak alkalmazásával [16] is elkerülhetjük kapcsolatok közötti metszéseket.

2.1.1. Vizualizációs technikánk háttere

A kapcsolatok megjelenítésére adott vizualizációs technikánkat a Circosban található problémás esetek kiküszöbölésére alkottuk meg. A megoldásunk, és a mögötte húzódó motiváció könnyebb megértése érdekében mutatjuk be röviden ebben a fejezetben, a Circos táblázat vizualizációját [29], valamint a problémás eseteit.

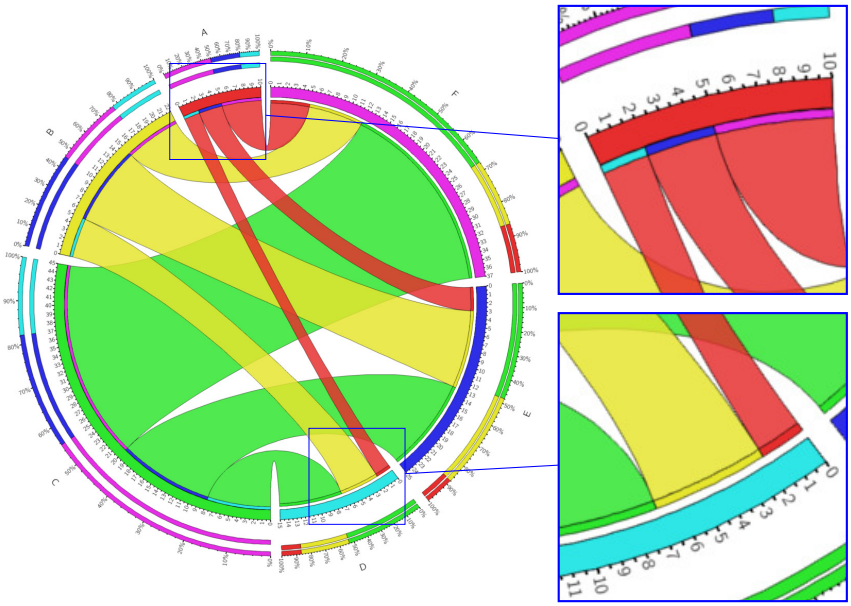
Circos táblázat vizualizációja

A Circos genomi információk megjelenítésére használt körön alapuló vizualizációs technikáját, univerzális, táblázatba rendezhető adatok ábrázolására is kiterjesztették. A cellákban megadott értékek a hozzájuk tartozó sor és oszlop közötti kapcsolatra, és annak valamilyen tulajdonságának a leírására szolgálnak. Az így definiált kapcsolatokat szalagok („ribbon”) reprezentálják a vizualizáció alapkörén belül, amelyeknek a hozzájuk tartozó cellában található pozitív szám határozza meg a szélességét (lásd 2.1. ábra). Ha egy cella értéke nulla, akkor a hozzá tartozó sor és oszlop között nincsen kapcsolat. A diagram alapkörének külső oldalán helyezkednek el a szegmensek, amelyek a táblázat sorait és oszlopait reprezentálják. A táblázatban megadott értékeket használja fel a szegmensek és kapcsolatok méretét meghatározó előfeldolgozási lépés (lásd 2.1. ábra). A kör mentén elhelyezkedő szegmensek körívét a hozzá tartozó oszlopban vagy sorban lévő értékek összege határozza meg.

2.1. ábra. A Circos táblázat vizualizációjának egy kapcsolata [28], ahol a jobb oldalon látható szalag vastagsága reprezentálja a bal oldali táblázat B sorában és E oszlopában lévő cella 10-es értékét.

A kapcsolatok végén olyan körgyűrűcikkek („ribbon cap”) találhatók, amelyek a kapcsolat másik végén elhelyezkedő szegmenstől kapják a színűket, a szélességük pedig megegyezik a kapcsolatéval (lásd 2.2. ábra). Az így kapott elemek célja a kapcsolatban résztvevő másik szegmensnek, a szalag végigkövetése nélküli meghatározása. A Circos úgy definiálja a kapcsolatok irányát, hogy az mindig abból a szegmensből indul ki, ame-

lyiket a hozzá tartozó szalag végpontja érint (lásd 2.2. ábra). A szegmenseken túl helyezkednek el a közreműködést jelző sávok („contribution track”). A szegmensbe érkező kapcsolatok értékének az egymáshoz viszonyított százalékos eloszlását mutatja be a kör centrumához közelebbi sáv, míg a távolabb elhelyezkedő sávban a szegmensből kiinduló kapcsolatokra hasonlóan kiszámolt százalékos értékeket láthatjuk. A diagram eddig felsorolt alkotóelemei a 2.2. ábrán tekinthetők meg.



2.2. ábra. A Circos táblázat vizualizációjával kapott diagramon kiemeltük, hogy miként kerülnek megjelenítésre a kapcsolatok irányai, valamint az úgynevezett „ribbon cap” körgyűrűcikkek.

A Circos táblázat vizualizációja a vastagabb kapcsolatok észlelésére, valamint az ezekhez tartozó nagyobb értékek gyors leolvasására a legalkalmasabb. Emellett jól használható a vastag szalagokkal összekapcsolt szegmensek megtalálására is. Azonban vannak olyan problémás esetek, amikor hiányossággal küzd a vizualizáció. A diagramban használt jelölések alapján a felhasználó nem tudja pontosan leolvasni a kapcsolatok értékét, csak megbecsülni képes azokat. Továbbá, a kapcsolatok értékeit csak nehézkesen lehet összehasonlítani a szalagok vastagsága alapján, ha

a megjelenítendő táblázatban található értékek közötti eltérés kicsi. Egy másik ilyen eset, amikor két kapcsolat vagy szegmens összehasonlítása azért ütközik problémákba, mivel azok távol helyezkednek el egymástól. Továbbá, a vastagabb szalagok akár teljes mértékben elfedhetik a vékonyabb társaikat, ha a szegmensek közötti különbség jelentős. Emiatt előfordulhat, hogy a felhasználó nem vesz észre lényeges részleteket a diagramon, mint például a kapcsolatok és az irányuk által alkotott mintázat.

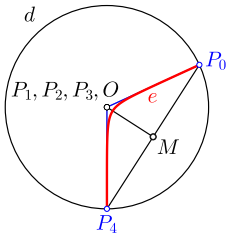
A szalagok végén két információ, a kapcsolatok iránya, valamint a kapcsolatban résztvevő másik szegmens is egy kis területen kerül megjelenítésre. Ennek következtében az egyes információkat reprezentáló vizualizációs elemek nem elég feltűnőek, és így sokszor alig ismerhetők fel a diagramban. Ez még inkább igaz akkor, ha valakinek nincs előismerete a vizualizációval kapcsolatban, és nem tudja, hogy hol kell keresni ezeket az információkat. Ezeken felül három korlátozása van a bemeneti fájlokra vonatkozóan a Circosnak. A fájlok csak ASCII karaktereket tartalmazhatnak, a szegmensek nevében nem szerepelhetnek szóköz karakterek, valamint a bemeneti fájlban nem engedélyezettek a duplikált sor- vagy oszlopnevek.

Circos által használt Bézier görbe

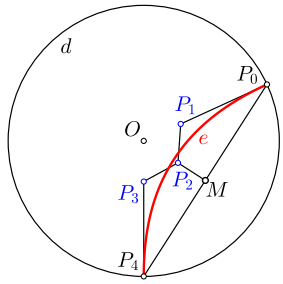
A Circos Bézier görbéjének ismertetésére, valamint az értekezés további részeiben hasonló jelöléseket használtunk, mint Glaeser és munkatársai [18][1]. A kapcsolatok kialakításához használt Bézier görbének öt kontrollpontja van, amelyek megadásához négy paramétert definiáltak. A *Diagram sugár*(r) paraméter határozza meg P_0 és P_4 pontok O ponttól vett távolságát ($\overline{OP_0} = \overline{OP_4} = r$), mivel a d kör sugarát adja meg (lásd 2.3a. ábra).

A következő, *Bézier sugár*(r_b) és a *Bézier sugár szabadság*(r_p) paraméterek együtt határozzák meg a P_2 pont helyzetét a P_0P_4 szakasz szakaszfe-

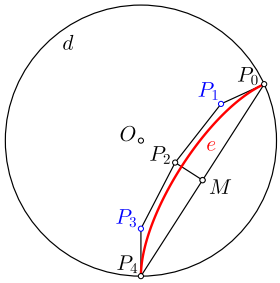
[1] A P_0 és P_2 pontok által definiált egyenest $[P_0, P_2]$ alakban jelöljük. Két pont által határolt szakasz jelölésére a P_0P_2 alakot használjuk, két pont távolságát pedig $\overline{P_0P_2}$ alakban jelöljük.



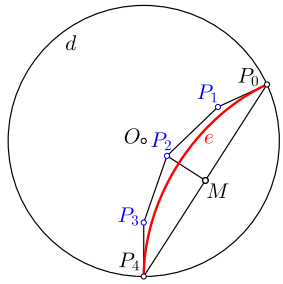
(a) Magasság: 1.0
Bézier sugár: 0.0
Diagram sugár: 0.75
Bézier sugár szabadság: 1.0



(b) Magasság: 1.0
Bézier sugár: 0.3
Diagram sugár: 1.0
Bézier sugár szabadság: 1.0



(c) **Magasság: 0.5**
Bézier sugár: 0.3
Diagram sugár: 1.0
Bézier sugár szabadság: 1.0



(d) Magasság: 0.5
Bézier sugár: 0.3
Diagram sugár: 1.0
Bézier sugár szabadság: 1.5

2.3. ábra. Példák, a Circos által használt Bézier görbék különböző paraméterbeállításokkal való megadására.

lező merőlegesén ($\overline{OP_2} = r_b \cdot r + \left(1 - r_p\right) \cdot \left| (r_b \cdot r) - \overline{OM} \right|$), amely a Bézier görbe szimmetriatengelye (lásd 2.3b. és 2.3d. ábrák). A *Magasság*(c) paraméterrel pedig a P_1 és P_3 pontok helyzete befolyásolható ($\overline{OP_1} = \overline{OP_3}, \overline{OP_3} = = r + \left(\overline{OP_2} - r\right) \cdot c$) a $[P_0O]$ és $[P_4O]$ egyeneseken (lásd 2.3c. ábra).

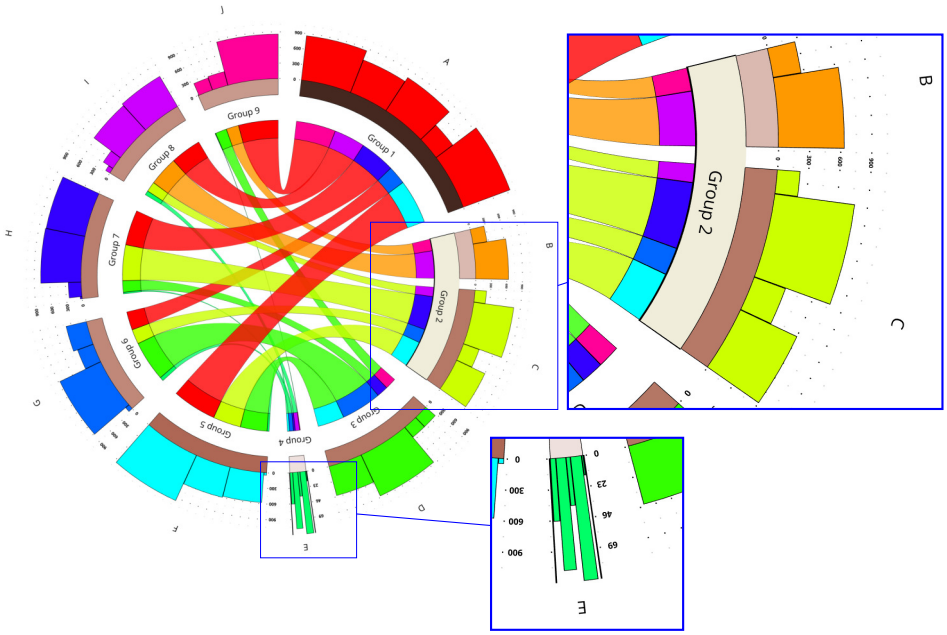
2.2. TabularVis

Az új vizualizációs technikánk megalkotásakor a diagram egyértelműségének növelését tűztük ki célul a Circos problémás eseteiben. Ennek érdekében néhány kritériumot fogalmaztunk meg, amelyek egyben iránymu-

tatásként is szolgálnak a vizualizációs technikánkra vonatkozóan. Az első, a Circos által használt körön alapuló vizualizációs technikának a saját megoldásunkban történő használata. Ezt követi a kapcsolatok és azok irányát reprezentáló elemek újradefiniálása, amellyel az elemekhez kapcsolt értékek meghatározását és összehasonlítását akarjuk segíteni. Az utolsó kritériumunk, a táblázat sorait és oszlopait reprezentáló elemek csoportokba rendezését, valamint a hozzájuk tartozó címkéik elhelyezéséhez nagyobb hely biztosítását tűzi ki elérendő célként. A megoldásunk elkészítése során a Circosban használt Bézier görbéket alkalmaztuk a kapcsolatok megadására. Azonban a fejezet második felében bemutatjuk, hogy a Bézier görbe miként helyettesíthető implicit egyenlettel rendelkező görbék segítségével. A kritériumok mentén megszületett funkciókat opcionális elemekként implementáltuk, hogy a felhasználó eldönthesse és interaktívan módosíthassa mely funkciót vagy azok kombinációját kívánja használni a legjobb eredmény elérésének érdekében.

Az első kritériumunk alapján lecseréltük a Circos, sorok és oszlop reprezentálására használt elemeit azért, hogy megkönnyítsük a kapcsolatok értékének az összehasonlítását és meghatározását. A 2.4. ábrán mutatjuk meg az újonnan definiált elemünket, amely a kapcsolat végén, annak kiterjesztéseként szolgál. Emiatt, a vastagsága és magassága is a kapcsolat értékétől függ. Az így kapott elemet a hozzá tartozó kapcsolattal a diagramunk legkisebb építő elemének tekintjük, és az értekezés további részében úgy fogunk hivatkozni erre az új elemre mint *sáv*. A vizualizációnk elkészítéshez használt adatokat tartalmazó táblázat sorait és oszlopait blokkok segítségével reprezentáljuk. A táblázatban, a sorok és oszlopok között definiált kapcsolatokat pedig a blokkok szalaggal történő összekötésével reprezentáljuk. Mivel egy sor vagy oszlop több kapcsolatot is leírhat a táblázatban, ezért minden olyan kapcsolat végén elhelyezkedő, újonnan definiált sáv elemünket tartalmaznia kell a blokknak, amelyet a blokkhoz tartozó sor vagy oszlop definiál. Így minden blokk egymás mellé helyezett sávokból épül fel, és az azonos blokkba tartozó sávok ugyanazzal a színnel jelennek meg az ábrán (lásd 2.4. ábra). Ezzel továbbra is

biztosítva marad a lehetőség, hogy a felhasználó minden sorhoz és oszlophoz különböző színt rendelhessen.

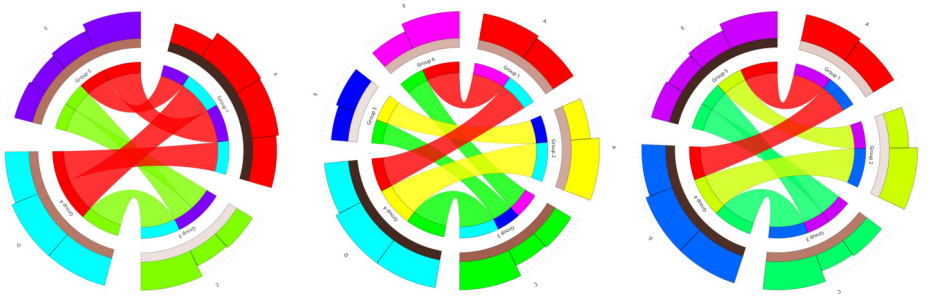


2.4. ábra. A kapcsolatok kiterjesztéseként definiált sávok közösen blokkokat alkotnak, amelyeket a képen látható (*B* és *C* blokk) csoportokba szervezhetünk. A vizualizáció alapkörével koncentrikusan elhelyezett segédkörök láthatók a sávok mögött. Ezeknek a skálázása akár blokkonként is megadható (*E* blokk). A képen látható kapcsolatok nem rendelkeznek iránnyal.

A diagramban csoportokat alakíthatunk ki, hogy azokban helyezzük el a blokkokat, és ezáltal egy összetettebb struktúrát hozzunk létre. Alapértelmezetten minden blokkot külön csoportba helyezünk el, ahogyan a 2.4. ábra legtöbb blokkjánál is látható. A felhasználó ezután kedve szerint módosíthatja a csoportokat és a bennük található blokkokat. Minden csoportot egy körgyűrűcikk jelöl a blokkok alatt, amely a csoport nevét tartalmazza (lásd 2.4. ábra).

A Circostól eltérően, az alkalmazásunk lehetővé teszi több, azonos nevű sor vagy oszlop megadását a táblázatban. Az összes megegyező címkével ellátott sort és oszlopot alapértelmezés szerint egy blokkba rendezzük,

ami miatt ezen sorok és oszlopok csoportja megegyezik (lásd 2.5a. ábra, *B* és *C* blokk). Azonban ezt az automatikus összevonást ki is kapcsolhatja a felhasználó, amely estén különálló blokkokba és csoportokba kerülnek az azonos címkéjű sorok és oszlopok (lásd 2.5b. ábra). Mindemellett kevert módon is történhet az összevonás. Ehhez a felhasználónak explicit meg kell adni a sorok és oszlopok csoportját, amely alapján csak az azonos csoportba tartozó, azonos címkéjű blokkok kerülnek összevonásra (lásd 2.5c. ábra). Az utolsó két lehetőség használatával az felhasználó összetettebb szerkezetű diagramokat hozhat létre.

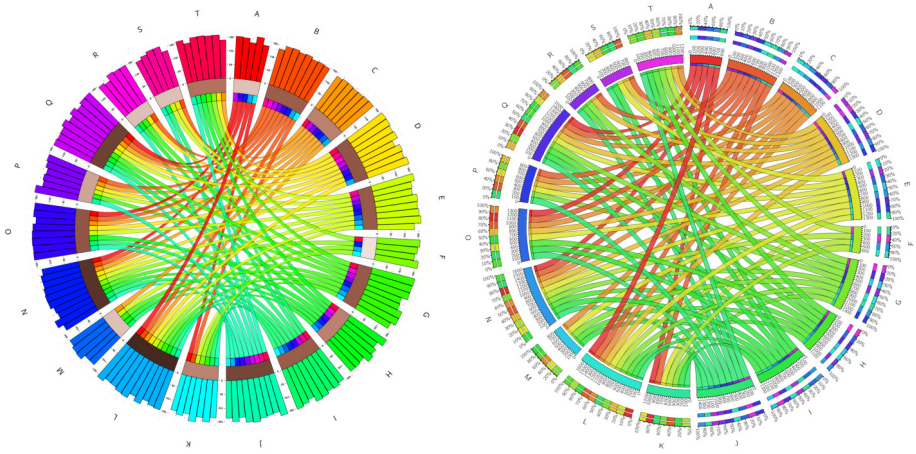


(a) összevont (b) különálló (c) kevert

2.5. ábra. A táblázatban azonos címkével ellátott sorok és oszlopok blokkokba rendezésének és csoportokba helyezésének lehetséges módjait mutatják be a képek.

A vizualizáció alapkörére koncentrikus segédköröket helyeztünk el a sávok mögött, amelyekből pontozott köríveket készítettünk. Ezeknek a két oldalán egy-egy számot helyeztünk el, amiket a legnagyobb sáv értéke alapján határoztunk meg. A körívek és címkék, együtt segítik a felhasználót, a sávok értékének meghatározásában és összehasonlításában (lásd 2.6. ábra). Ehhez a címkék betűméretét automatikusan határozzuk meg a csoportok és blokkok között megmaradt üres területek méretének a függvényében.

Továbbá, a felhasználó minden kapcsolat pontos értékét megjelenítheti a sávok tetején. Így lehetősége van úgy dönteni, hogy elrejti a címkéket és több helyet biztosít a kapcsolatok számára. Emellett az egyes blokkokat saját egyedi skálázással láthatja el, hogy azon blokkokban is jól láthatók legyenek a sávok, amelyek értékkészlete nagyban eltér a többitől



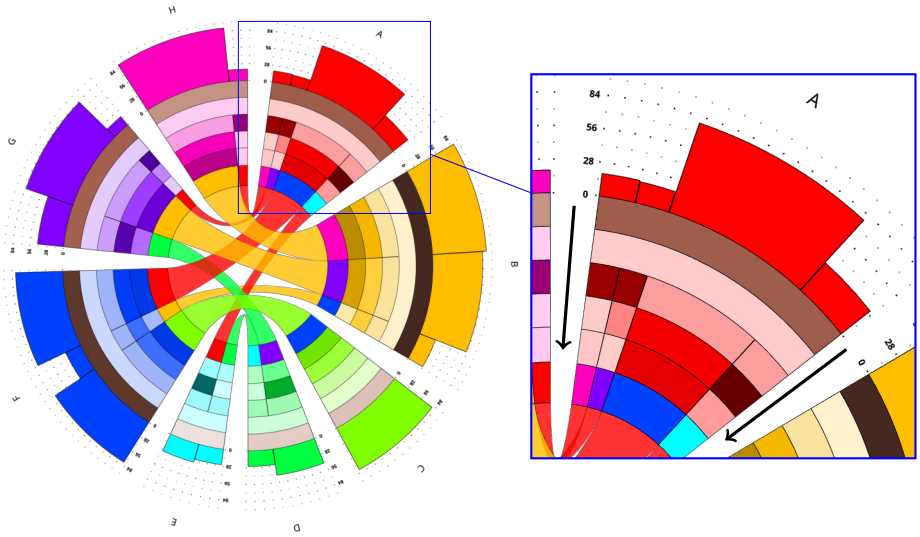
(a) Saját (b) Circos

2.6. ábra. A bemutatott sávok és koncentrikus körök használatával készült diagram értékeinek a leolvashatóságát hasonlítottuk össze a Circosal.

(lásd 2.4. ábra).

A blokkok nem kapcsolódnak közvetlenül a szalagokhoz, mert további, a blokkra vagy kapcsolatra vonatkozó, a felhasználó részére releváns információk kerültek elhelyezésre közöttük. Két részre bonthatjuk a megjeleníthető információkat annak függvényében, hogy blokkról vagy kapcsolatról szolgáltat új ismereteket. Az egyes információk mindig egy olyan újabb körgyűrűben kerülnek megjelenítésre, amelyben szegmenseket helyezünk el. Ezen szegmensek vastagsága megegyezik annak az elemnek a vastagságával, amelyikhez a megjelenített információ tartozik. Ez az elem lehet blokk, ha egy egész blokkra vonatkozik az információ (pl. a blokkban található kapcsolatok összértéke) vagy pedig egy kapcsolat (pl. a kapcsolatot reprezentáló szalag, mennyi szalagot keresztez a vizualizációban). A megjelenítendő értékek nagyságát pedig a szegmensek színének világossága jelzi. A 2.7. ábrán látható sorrendben jeleníthetők meg az információk a körgyűrűk által. Az első gyűrű a blokkok alatt, amely a blokkban található sávok összértékét jeleníti meg, alapértelmezetten mindig megjelenítésre kerül.

A sávok alkalmazásával lehetőségünk nyílt a kapcsolatok érték szerin-

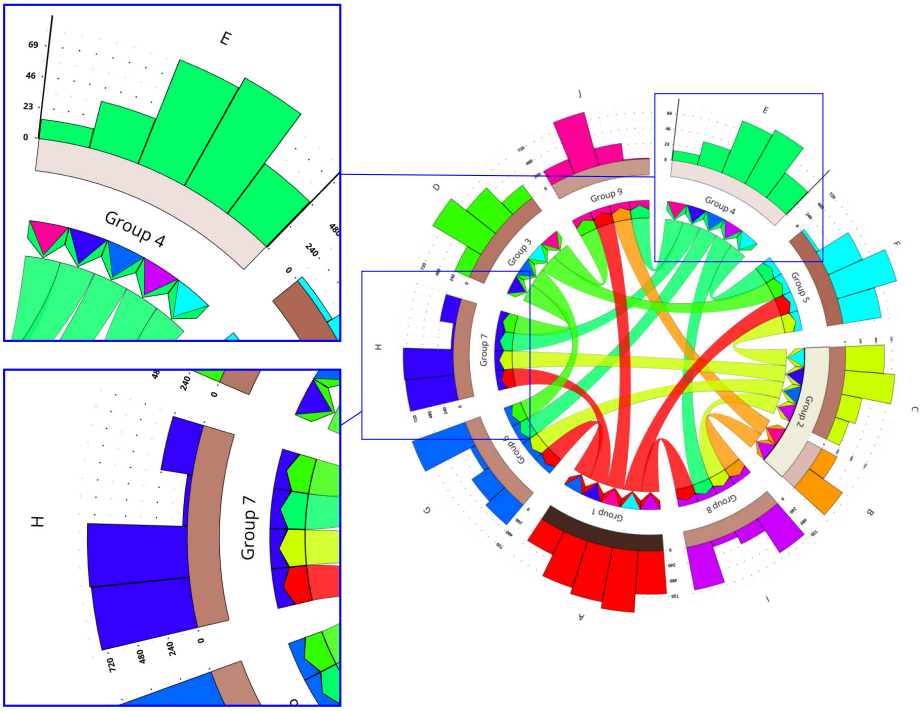


2.7. ábra. A blokkok és kapcsolatok között további információkat jeleníthet meg a felhasználó, amelyek a nyíllal jelölt irányban, a blokkoktól befelé a kapcsolatokig haladva a következők: blokkban található sávok összértéke, blokkban elrejtett sávok száma, legkisebb és legnagyobb értékkel rendelkező kapcsolat, kapcsolat által keresztezett kapcsolatok száma és a kapcsolat értéke.

ti szűrésére, amellyel a felhasználó egy tartomány megadásával szűkítheti le, hogy mely kapcsolatok és hozzájuk tartozó sávok jelenjenek meg a diagramban. Ennek köszönhetően csak a lényeges kapcsolatok kerülnek megjelenítésre. Fontos információ lehet a felhasználó számára a szűrés eredményeként kihagyott sávok száma, mivel ezáltal láthatja mennyi kapcsolat esett a kiválasztott tartományon kívülre minden blokkban. A blokkok és kapcsolatok között elhelyezett második körgyűrűben pontosan ezt az információt jeleníthetjük meg.

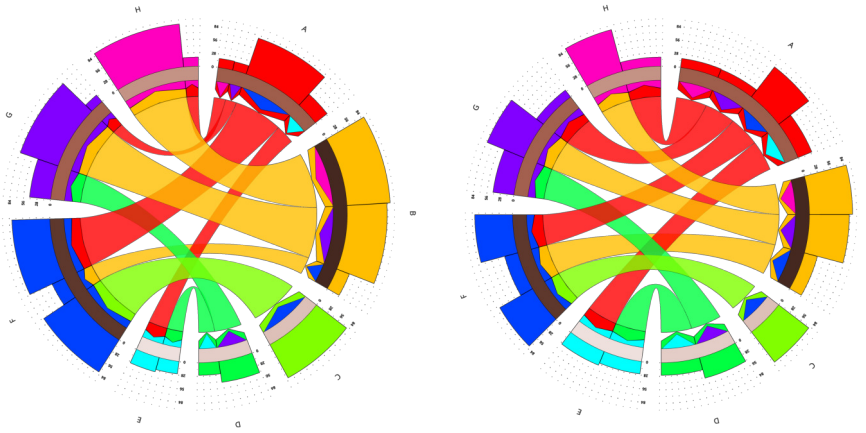
Egy másik informatívabb módot kívántunk nyújtani a kapcsolatok irányának megjelenítésére, amelyhez új alakzatokat vezettünk be. Ezeket a kapcsolatok végéhez fűztük hozzá, ahogy a 2.4. és 2.8. ábrákon is látható. Az új alakzatainknak három különböző formája lehet a kapcsolat iránya alapján. Ezek közül kettő, nyílszerű formával jelzi az irányt, míg az utolsó alakzat (egy szegmens) azt mutatja meg, hogy a kapcsolat nem rendelke-

zik iránnyal (lásd 2.4. ábra). Az elemek színe a kapcsolat másik végpontján található blokkot jelzi, ugyanúgy, mint a Circosnál.



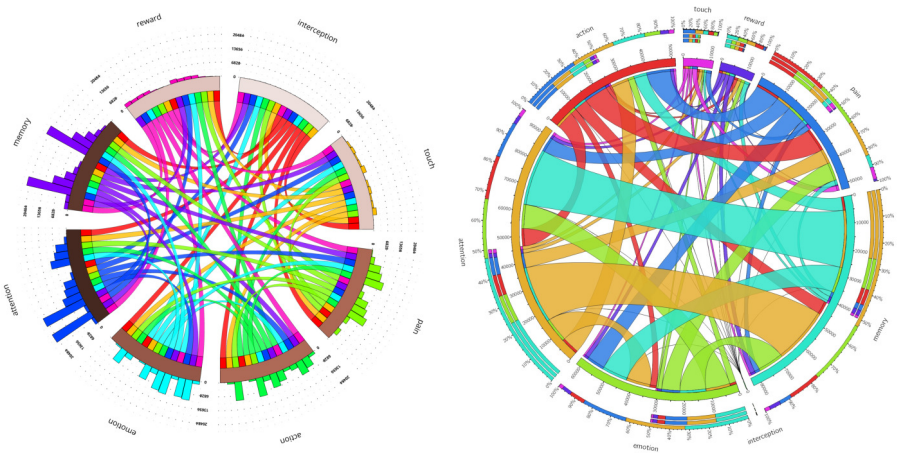
2.8. ábra. A kapcsolatok irányát jelző nyíl elemeinket azonos vastagságú szalagok végén helyeztük el. Az *E* blokk alatti nyilak a kapcsolat kiinduló blokkját, míg a *H* blokknál megfigyelhetők, a kapcsolat beérkező blokkját jelölik.

Minden kapcsolatot egyformán láthatóvá teszünk a vékonyabb kapcsolatok láthatóságának növelése érdekében azáltal, hogy az ábrán látható szalagok és sávok vastagságát egységesítjük (lásd 2.9. ábra). Ez azt eredményezi, hogy a szalagok már csak az oszlopok és sorok közötti kapcsolatokat reprezentálják, a táblázatban hozzájuk rendelt érték pedig már csak a sávok magasságából határozható meg. Az azonos vastagságú szalagok használta anélkül könnyíti meg a kapcsolatok által alkotott mintázatok felismerését, hogy az ábra egy adott részére irányítanák a figyelmet (lásd 2.10a. ábra). Emellett a funkció alkalmazása növeli a sávok láthatóságát és összehasonlíthatóságát, valamint elkerülhetjük vele a nyílszerű formáink vékony kapcsolatok okozta deformációs problémáját.



(a) eltérő vastagság (b) azonos vastagság

2.9. ábra. A kapcsolatok irányát meghatározó új elemeinket hasonlítjuk össze változó és megegyező vastagságú kapcsolatok esetén. Jól látszódik, hogy azonos vastagság használata mellett a kapcsolatok irányát ábrázoló elemeink nem deformálódnak.



(a) Saját (b) Circos

2.10. ábra. Az azonos vastagságú kapcsolatok által kirajzolt mintázat láthatóságát hasonlítjuk össze egy Circos-al készült vizualizációval. Az agy működésével kapcsolatos adatokat tartalmazó táblázat [9] felhasználásával készültek a diagramok.

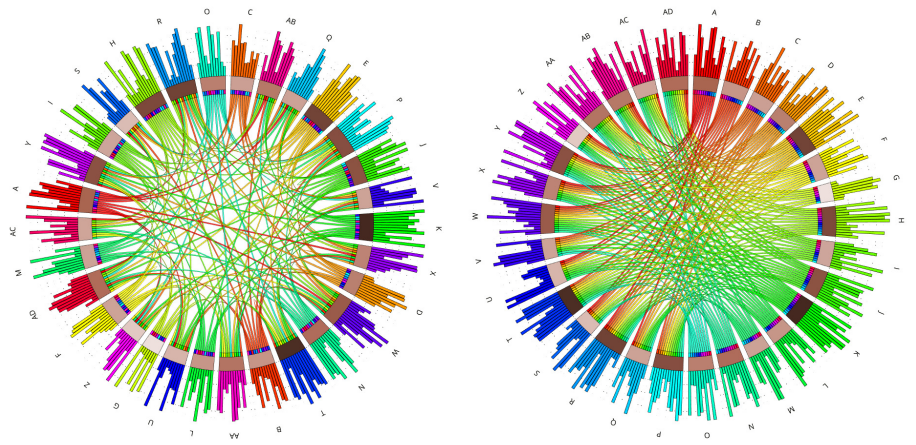
2.2.1. A diagram átláthatóságának növelése

Két megoldást is kínálunk a diagramban kialakulható káosz és zsúfoltság csökkentésére, amelyek megnehezíthetik a kapcsolatok és azok tulaj-

donságainak a leolvasását. Az első megoldásunkban egy olyan rendezési módszert javaslunk, amely a blokkok legkevesebb metszést eredményező elrendezését keresi. A másodikban pedig a hierarchikus élcsomagolási technika [21] segítségével kötegeljük az azonos csoportok között definiált kapcsolatokat a diagram átláthatóságának növelésére. A Circos szoftvercsomagban nincs ilyen vagy ehhez hasonló funkció, amely a kapcsolatok zsúfoltságát csökkentené és növelné a diagram átláthatóságát.

A kapcsolatok rendezése

A diagramban előforduló metszéspontok magas száma jelentősen csökkenti az ábra egyértelműségét, mivel minden egyes újabb metszésponttal még nehezebben követhetővé válnak a kapcsolatok. A csoportok és blokkok elhelyezkedésének jelentős hatása van a kapcsolatok között kialakult metszéspontok számára. Ezért a kör mentén pakolgatva a csoportokat és blokkokat próbálunk minél kevesebb metszést elérni. Az értekezés további részében, a kapcsolatok rendezéseként hivatkozunk erre a folyamatra. A 2.11. ábrán mutatjuk meg, hogy a javasolt rendezési módszerünk miként képes a diagramban megjelenő káosz és zsúfoltság csökkentésére.



(a) Rendezéssel

(b) Rendezés nélkül

2.11. ábra. A kapcsolatok rendezésére javasolt metódusunk összehasonlítása.

A gráfelmélet területén jól ismert problémára, az élek körön belüli metszéseinek minimalizálására, vezethető vissza a bemutatott rendezési feladatunk. Ez egy NP-nehéz feladat, amelyre lokális kereső és optimalizáló, valamint heurisztikus módszerek használatával kerestük a megoldást. Egy olyan teljesen automatikus funkciót kívánunk velük biztosítani a diagramban található metszéspontok számának csökkentésére, amely lehetővé teszi a csoportok és blokkok rendezés előtti tetszőleges elhelyezését. Emellett a megoldásunk lehetőséget biztosít a rendezés finomhangolására azáltal, hogy a csoportok és blokkok bármilyen kezdeti elrendezéssel megadhatók a rendezés számára.

A következő két lokális keresési algoritmus (hegymászó és minimális konfliktusok), valamint három optimalizálási módszer segítésével (keresztentrópia, szimulált hűtés, és méhek algoritmusa) rendeztük a kapcsolatokat [7,60]. Az algoritmusok tesztelése során a függelékben elhelyezett A.1. táblázatban ismertetett paramétereket használtuk. Azonban a módszerek nem garantálják, hogy egy olyan állapotot tudunk megtalálni a használatukkal, amelyben a legkevesebb metszés fordul elő. A legjobb elrendezés megtalálásához a csoportok és blokkok összes lehetséges elhelyezését tesztelni kellene, viszont nagyméretű táblázatok esetében ez túl sokáig tartana. A használt módszerekkel pedig nagy a kockázata annak, hogy egy olyan, csak lokálisan jó elrendezésnél állnak meg, amelynél létezik jobb elrendezés is. Emiatt az eredményünk csak a legjobb „elméleti" megoldás közelítése lesz. Azonban, ez a legtöbb esetben jól használható a vizualizáció átláthatóságának növelésére (lásd 2.11. ábra).

A sávok blokkon belüli elhelyezkedését a rendezés közvetlenül nem módosítja, mivel azok a blokkok és csoportok végső helyzete alapján kerülnek meghatározásra. Egy blokk sávokkal való feltöltéséhez, az adott blokktól indulva végighaladunk az összesen blokkon, és megvizsgáljuk, hogy van-e köztük kapcsolat. Ha van, akkor a kapcsolathoz tartozó sávot a blokkba helyezzük. Ezt addig folytatjuk tovább amíg körbe nem érünk. A sávok ilyen módon történő elhelyezése biztosítja, hogy a blokkon belüli kapcsolatok ne metsszék egymást.

A használt lokális kereső és optimalizáló módszerekhez olyan keresési teret adunk meg, amely segítségével a diagram minden lehetséges kimenetelét le tudjuk írni. Állapotnak nevezzük a csoportok és blokkok olyan módon megadott elrendezését, amelyen a rendezéshez használt algoritmusok operálni tudnak. Az állapotokban tároljuk el a csoportok és blokkok helyzetét, a közöttük lévő kapcsolatokat, valamint a kereső és optimalizáló algoritmusunk által elvégezhető műveleteket. Hierarchikus adatstruktúra használatával, az előbb definiált állapotunkat teljes egészében a kapcsolatok köré építettük, hogy minden rendezéshez szükséges információt egyszerűen és gyorsan hozzáférhető módon tudjunk tárolni. Az állapotunkban definiált műveletek a csoportok és blokkok pozícióját változtatják meg. Emiatt minden állapotváltás egy új elrendezését jelent a diagramban.

Egy tetszőleges A állapot szomszédjainak nevezzük az összes olyan egymástól különböző állapotot, amelyek az A állapotban definiált műveletek végeredményeként előállhatnak. Ezek két csoport vagy blokk pozíciójának a felcserélésével alkotnak egy új állapotot. Annyi ilyen új állapotba tudunk átlépni, ahányféleképpen képesek vagyunk kiválasztani két elemet a pozícióik felcserélésére.

$$g(i,n) = \left\lfloor \frac{i}{(n-1)} + 1 \right\rfloor \tag{2.1}$$

$$f(i,n) = g(i,n) + (i \bmod (n-1) + 1) \tag{2.2}$$

A 2.1 és 2.2 függvények segítségével egy adott állapot minden szomszédjához a $\left(g(i,n), f(i,n)\right)$ számpárt rendeljük, amely megadja, hogy az i indexel leírt szomszéd állapotba, mely két elem felcserélésével léphetünk át. A függvények i paraméterének az értékét a maximálisan kiválasztható szomszédok száma határozza meg: $n \cdot (n-1)$. Az n értéke attól függően lehet a diagrambeli csoportok száma, vagy pedig egy adott csoporton belüli blokkok száma, hogy csoportokat vagy blokkokat kellett felcserélni az új állapotba történő lépéshez. A rendezés során, először két csoport helyzetét cseréljük fel, majd pedig a két csoporthoz tartozó blokkokat rendez-

zük. A csoportok és blokkok pozíciójának a felcserélése, valamint rendezése ugyanúgy történik.

Egy olyan geometriai problémaként adtuk meg annak az eldöntését, hogy két kapcsolat metszi-e egymást, amelyben a kapcsolatok végpontjaival felírható két körív átfedését vizsgáljuk. Ehhez először meghatározzuk, mikor fedi két körív egymást. Legyen k és l két, ugyanahhoz a körhöz tartozó tetszőleges körív. Ezek akkor vannak átfedésben egymással, ha k és l körívnek legalább egy közös pontja van, és mindkét körívnek van legalább egy olyan pontja, amely különbözik a másiktól. A definíciónk alapján kizárjuk azt az esetet, amikor a két körív egybevágó, vagy az egyik körív tartalmazza a másikat.

A célfüggvényünkhöz geometriai információkká kell alakítanunk a blokkok és csoportok elrendezésbeli indexét, hogy az így kapott körívek alapján eldönthessük a kapcsolatok közötti metszéseket. A sávok egyéni elhelyezésének köszönhetően biztosított, hogy két kapcsolat nem fogja metszeni egymást, ha valamelyik végpontjuk ugyanahhoz a blokkhoz tartozik. Ennek köszönhetően a blokkokat elegendő pontokként kezelni körívek helyett, mivel csak azt kell ellenőrizni, hogy a két blokkhoz tartozó pont megegyezik-e.

A blokkok pozícióját polárkoordinátaként tároljuk, mivel a módszerünk nagymértékben támaszkodik az alapkör két tetszőleges pontja közötti szög meghatározására. Két blokk által felírható körívhez tartozó szöget a

$$d(\alpha,\beta) = \pi - \left|\pi - \left|\alpha - \beta\right|\right| \tag{2.3}$$

függvénnyel határozzuk meg, amelyre a továbbiakban távolságfüggvényként fogunk hivatkozni. A segítségével egyszerűen meghatározhatjuk a kapcsolatok hosszát azáltal, hogy kiszámítjuk azon két blokk közötti szöget, melyeket ez a kapcsolat összeköt. Emellett szükségünk van még a két blokk által meghatározott körív középpontjára,

$$m(\alpha,\beta) = \beta + \frac{1}{2}\left(-d(\alpha,\beta)\right) \cdot sgn(\alpha - \beta + \pi) \tag{2.4}$$

ahol $\beta > \alpha$.

A $d(\alpha,\beta)$ és $m(\alpha,\beta)$ függvények segítségével írhatjuk fel az alábbi egyenlőtlenséget, amely alapján eldönthetjük, hogy van-e metszés két kapcsolat között.

$$r\left(\alpha,\beta,\gamma,\delta\right) = \left|d(\alpha,\beta) - d(\gamma,\delta)\right| < 2d(m(\alpha,\beta), m(\gamma,\delta)) \\ < d(\alpha,\beta) + d(\gamma,\delta) \quad (2.5)$$

A 2.5 egyenlőtlenség segít eldönteni, hogy az α és β, valamint γ és δ szögek által meghatározott körívek fedik-e egymást. Két kapcsolatot akkor mondunk metszőnek ha a 2.5 egyenlőtlenség teljesül a kapcsolat végpontjainak szögeire. Ennek a meghatározásához az alábbi függvényt használjuk,

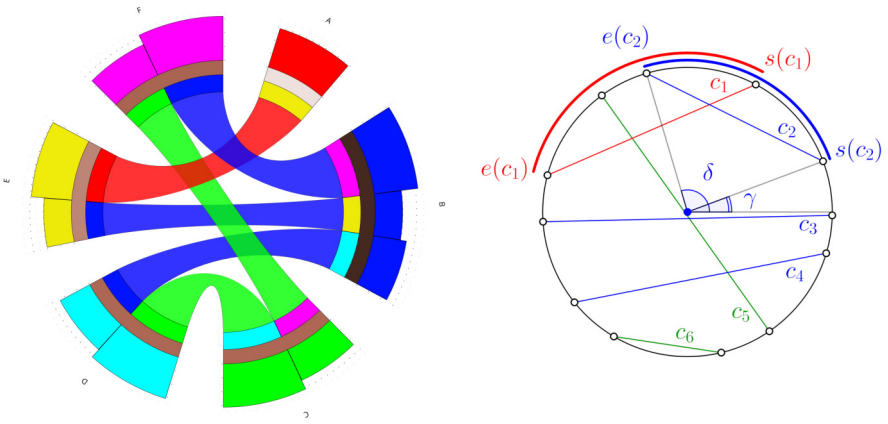
$$q(i,j) = \begin{cases} 1 & \text{ha } r\left(s(c_i), e(c_i), s\left(c_j\right), e\left(c_j\right)\right) \text{ teljesül} \\ 0 & \text{egyébként} \end{cases}, \quad (2.6)$$

ahol $c_1,\ldots,c_n$ jelöli az n darab kapcsolatot, $s(c_i)$ és $e(c_i)$ pedig a kapcsolatok blokkok szerinti kezdő és végpontjaihoz tartozó szögeket jelöli, $i \in [1,n]$. A 2.12. ábrán bemutatjuk az ismertetett függvények használatát két kapcsolat metszésének az eldöntésére. A diagramban előforduló metszések számát a $q\left(i,j\right)$ függvénynek a kapcsolatok minden kombinációján történő kiértékelésével kapjuk meg,

$$I = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} q(i,j). \quad (2.7)$$

A kapcsolatok csoportosítása

A vizualizáció átláthatóságának növelésére egy olyan élkötegelésen alapuló megoldást mutatunk be, amelynek megalkotásánál fontos volt, hogy a kötegelés elvégzése után is egyértelműen meghatározhatja a kapcsolatok végpontját, valamint azok irányát a felhasználó. Ezért a csoportba rendezésük alapján kötegeljük a kapcsolatokat, amelyhez hierarchikus élkötegelési technikát használunk. Ez azt jelenti, hogy két kapcsolat



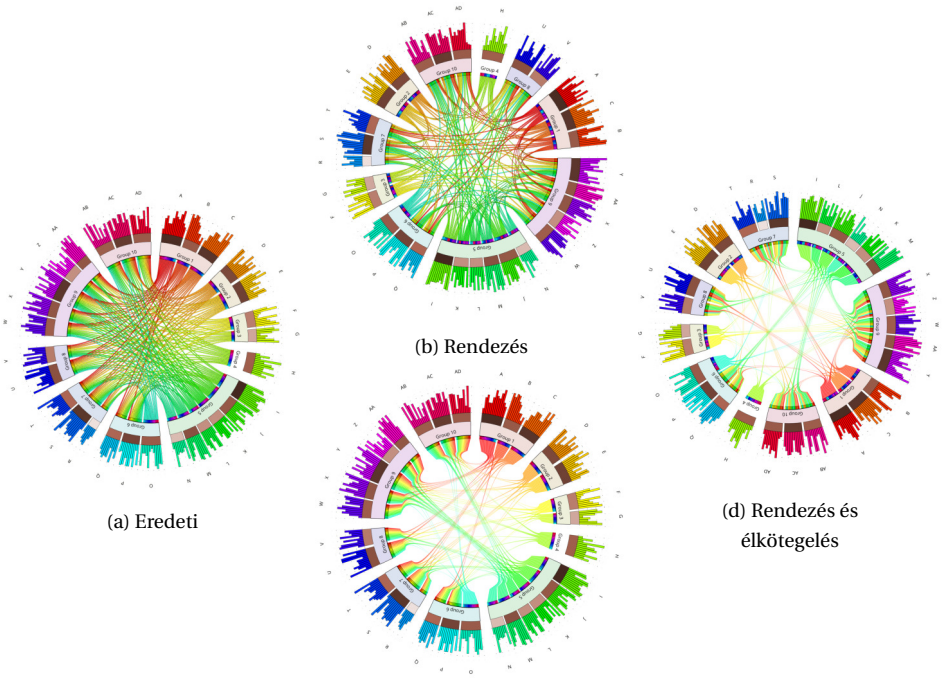
(a) kapcsolatok a diagramban

(b) kapcsolatok reprezentációja a rendezéshez

2.12. ábra. A képen látható, hogy miként vizsgálhatjuk a diagramban lévő kapcsolatok között megjelenő metszéseket azok végpontjai, és a hozzájuk tartozó szögek segítségével alkotott körívekkel. A c_1 és c_2 kapcsolatok metszésének vizsgálatához az $s(c_2)$ és $e(c_2)$ függvényekkel meghatározzuk a kapcsolat γ és δ szögeit. Az α és β szögeket hasonlóan határozhatjuk meg a 2.5. egyenlőtlenség számára, hogy a 2.6 függvénnyel eldöntsük a kapcsolatok metszését.

csakis akkor lehet kötegelve, ha ugyanazon csoportba tartoznak az általuk összekötött blokkok. A kötegelt kapcsolatok elkészítéséhez racionális Bézier görbéket alkalmazunk.

A 2.13. ábrán is láthatjuk, hogy az élkötegelés csak azokban az esetekben hasznos, amikor két csoport között egynél több kapcsolat van definiálva a diagramban. Továbbá megfigyelhető, hogy az élkötegelés használatával is kaphatunk túlzsúfolt ábrát. Azonban a rendezési metódusunk és az élkötegelés együttes alkalmazása egy átláthatóbb eredményt nyújt a felhasználó számára (lásd 2.13. ábra). Az élkötegelés nagy előnye a rendezéssel szemben, hogy a futási ideje nem függ a kapott táblázat méretétől, illetve a benne lévő kapcsolatok számától.



(a) Eredeti

(b) Rendezés

(c) Élkötegelés

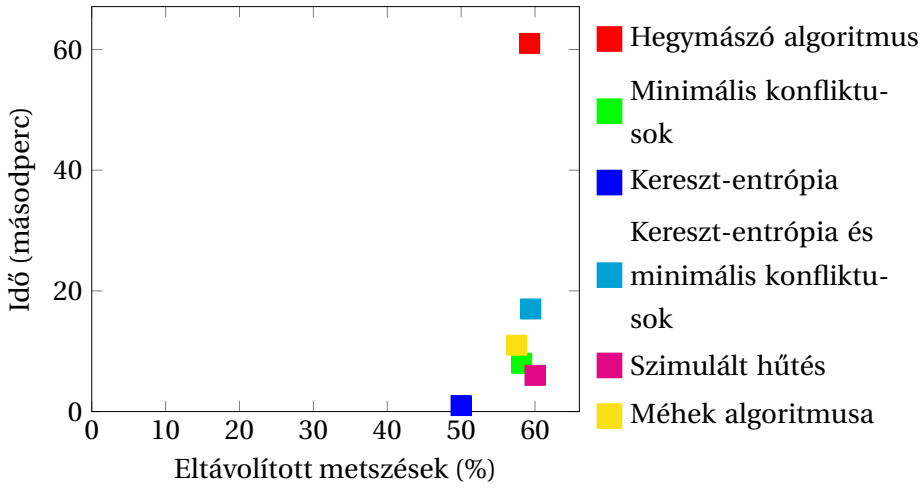
(d) Rendezés és élkötegelés

2.13. ábra. A rendezési és élkötegelési módszerünket, valamint ezek kombinációját hasonlítjuk össze a képeken.

2.2.2. Összehasonlítás

Négy különböző táblázatméretet (5 × 5, 10 × 10, 15 × 15, és 20 × 20) használtunk a diagram átláthatóságának növelésére bemutatott módszereink teljesítményének a mérése során. Minden méretből száz véletlenszerűen kitöltött táblázatot generálva teszteltük a rendezési metódusainkat. Az alkalmazásunkat egy Intel Core i7-8550U processzorral (8M Cache, akár 4,00 GHz) 16 GB memóriával rendelkező gépen vizsgáltuk. A 2.14 ábrán láthatjuk, hogy az egyes algoritmusok milyen mértékben csökkentették a kapcsolatok közötti metszések számát, valamint ehhez mennyi időre volt szükségük.

A 2.14. ábrán jól kirajzolódik, hogy a szimulált hűtés segítségével tudjuk a diagramban előforduló metszések legnagyobb részét eltávolítani, és mindehhez sok időre sincs szükség. Továbbá azt is láthatjuk, hogy a

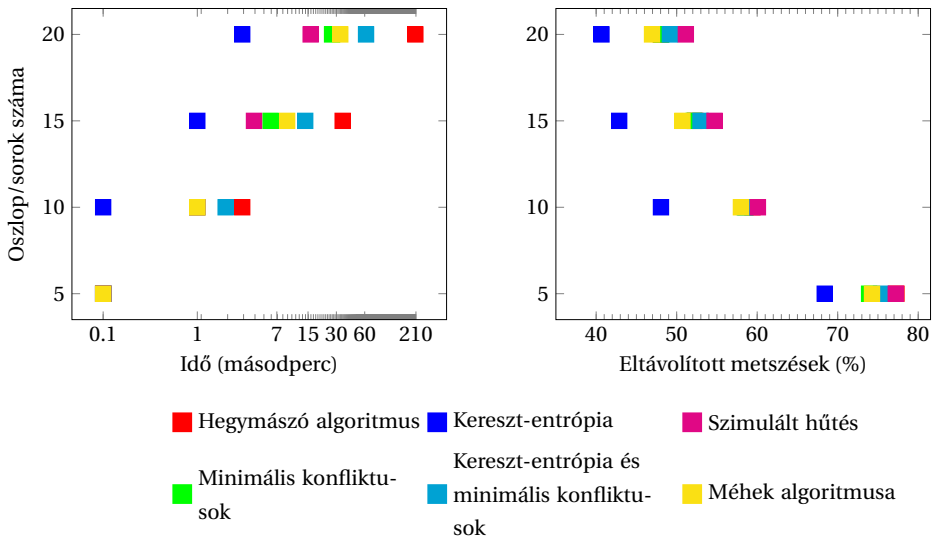


2.14. ábra. A használt keresési és optimalizálási algoritmusok teljesítménye és futási ideje. Annál jobb egy algoritmus teljesítménye, minél több az eltávolított metszések száma.

legrosszabbul teljesítő algoritmus, a kereszt-entrópia is képes átlagban a metszések felét eltávolítani, mialatt ez a leggyorsabb algoritmus a maga 1 másodperc körüli futási idejével. Számos kereső és optimalizációs metódus (méhek algoritmusa, minimális konfliktusok, kereszt-entrópia és minimális konfliktusok) csak kicsit van lemaradva a szimulált hűtés eredményeitől. Azonban, a hegymászó algoritmus hiába teljesít jól, ha a legnagyobb futási idővel rendelkezik. A különböző méretű táblázatokkal végzett tesztek összesített eredményét a 2.15. ábrán mutatjuk meg. Ezen jól látható, miként változik az algoritmusok viselkedése a táblázat méretének növelésével.

A 2.15. ábrán látható teszteredményeink jól szemléltetik, hogy azonos méretű táblázatok esetén nincsen jelentős különbség a módszerek teljesítménye között ($\approx 5\%$). Ezért a futási időt is mindenképpen figyelembe kell venni a megfelelő algoritmus rendezéshez történő kiválasztása során.

Két rendezési algoritmus (szimulált hűtés és minimális konfliktusok módszere) teljesítménye közötti különbséget mutatjuk meg a 2.16. ábrán. Ehhez egy 5, illetve 20 sorból és oszlopból álló táblázatot használtunk fel. A szimulált hűtéssel rendezett diagramok tartalmazzák a legkevesebb

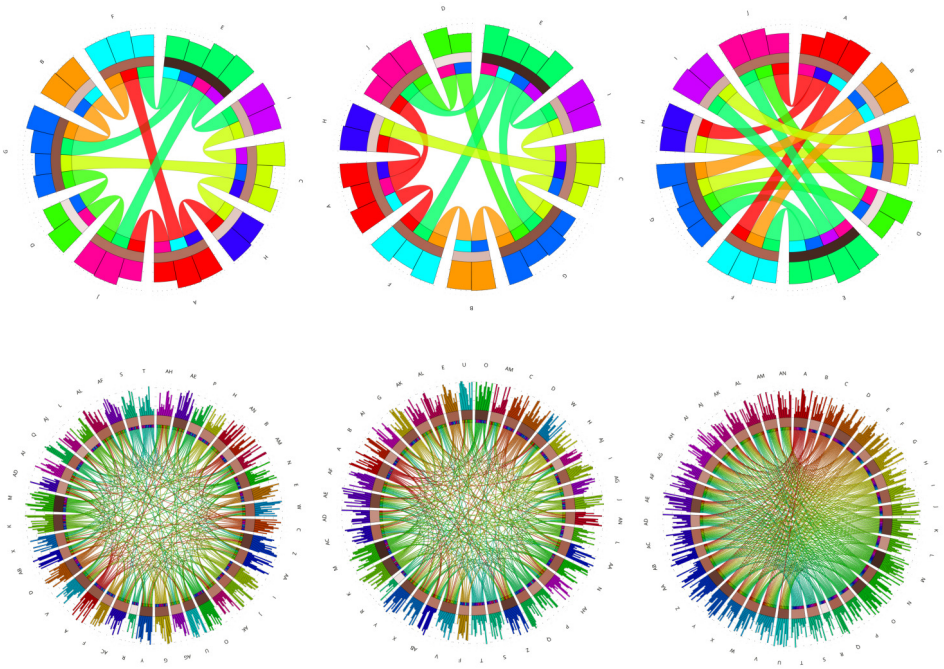


2.15. ábra. Az algoritmusok futási ideje és az eltávolított metszések száma különböző méretű táblázatok esetén.

metszést az ábrák közül, míg a minimális konfliktusok módszerével rendezettek már ≈ 5% több metszést tartalmaznak. A rendezés eredményei közötti különbségek a kis táblázatokkal készített diagramok esetében jóval egyértelműbbek, mint a nagyobb táblázatoknál. Az utóbbi esetben szinte teljesen elhanyagolható a rendezési algoritmusok között tapasztalt különbség. Mindezek mellett a 2.16. ábrán azt is láthatjuk, hogy egy kicsivel több metszést eredményező algoritmus kimenete is sokkal jobb, mint a rendezés nélküli diagram.

2.3. Bézier görbék lecserélésének vizsgálata kúpszeletekkel

A bemutatott alkalmazásunkban is látható, hogy a Bézier görbék jól teljesítenek a kapcsolatok vizualizációjában. Azonban felmerül a kérdés, hogy van-e másik olyan görbe, amely hasonlóan a Bézier görbéhez, jól használható a kapcsolatok vizualizációjához? A keresés során olyan görbékre



2.16. ábra. A rendezéshez használt szimulált hűtéssel (1. oszlop) és minimális konfliktusok módszerével (2. oszlop) elért eredményeinknek az összehasonlítása egymással és egy rendezés nélküli ábrával (3. oszlop). Az első sorban 5×5, míg a másodikban 20×20 táblázatmérettel készültek az ábrák.

koncentráltunk, amelyek rendelkeznek implicit egyenlettel, hogy a segítségükkel egyszerűen megvalósíthatóvá váljon a kapcsolatok vizualizációban történő interaktív kijelölése.

A másodrendű görbék rendelkeznek a kívánt implicit egyenlettel, valamint egyszerűen felírható a paraméteres egyenletük, amely megkönnyíti azok képernyőn történő megrajzolását. Továbbá egy-két adat megadásával bármelyik görbe egyértelműen megrajzolható, ellenben a bemutatott Bézier görbe különféle paramétereivel. Ezek miatt a továbbiakban azt vizsgáljuk, hogy miként használhatók a kúpszeletek, a Bézier görbék, kapcsolatok vizualizációjában történő helyettesítésre. Ehhez először megadjuk, hogy milyen kúpszeletekkel kívánjuk helyettesíteni a Bézier görbéket, majd pedig szerkesztési eljárásokat adunk meg a javasolt kúpszeletekhez.

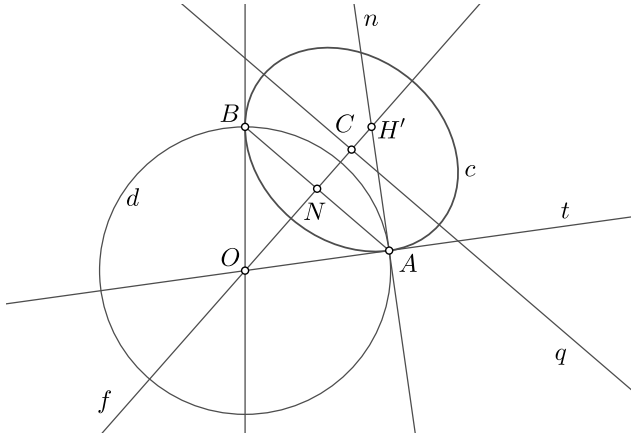
Továbbá kiemelünk néhány fontos tulajdonságot is, amelyek a szerkesztési módszereinkből adódnak. Ezt követően pedig megvizsgáljuk, hogy a javasolt kúpszeletek miként viselkednek, a kapcsolatok reprezentálására szalagokat alkalmazó vizualizációkban.

A Circos által használt Bézier görbének a d körrel vett metszéspontjaiban meghatározható kör- és görbeérintők egymásra merőlegesek. A Bézier görbék ezen tulajdonsága teszi lehetővé, hogy a belőlük alkotott linkek probléma nélkül egymás mellé helyezhetők. Ezért a javasolt kúpszeleteink is merőlegesen metszik a vizualizáció d alapkörét. Emellett meg akartuk őrizni a Circos kapcsolatainál használt szimmetriát, ezért úgy alkottuk meg a kúpszeleteket, hogy ugyanazt a szimmetriatengelyt használják, mint a Bézier görbék.

2.3.1. Kúpszeletek szerkesztése

A merőlegesen metsző kúpszeletek szerkesztésének ismertetése előtt bemutatunk néhány olyan jelölést, amelyekre az egész alfejezetben hivatkozni fogunk. A vizualizáció alapkörérre mint d, a középpontjára pedig O-ként fogunk hivatkozni. A c-vel jelölt kúpszelet egy kapcsolatot reprezentál, amelynek végpontjait két, a d körön definiált tetszőleges A és B ponttal jelölünk. Továbbá, ezt a két pontot felhasználva definiáljuk a kúpszeletek szimmetriatengelyét úgy, mint az AB szakasz szakaszfelező merőlegese. A c kúpszeleteink d körrel vett A és B metszéspontjaiban megkövetelt merőleges metszés miatt az $[AO]$ és $[BO]$ egyeneseket tekintjük a c kúpszelet A és B pontbeli érintőinek. Emellett jelöljük t és n egyenessel az c kúpszelet A pontbeli érintőjét és normálisát, a C pedig jelölje a középpontját. A 2.17. ábrán láthatók a szerkesztéshez megadott jelöléseink.

A továbbiakban ismertetjük a d kört merőlegesen metsző kúpszeleteink, valamint azok tengelypontjainak és kúpjának a szerkesztését. Ehhez két részre osztottuk a kúpszeleteink megadását bemutató részt aszerint, hogy miként választjuk meg a kúpszelet nagytengelyét. Az első részben azt a speciális esetet ismertetjük, amelyben a kúpszeletek nagytengelye



2.17. ábra. A merőlegesen metsző kúpszeletekhez használt jelöléseink.

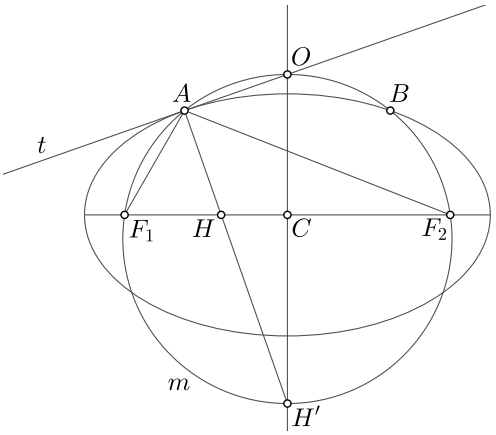
merőleges az f egyenesre, és az eredményül kapott kúpszelet minden esetben ellipszis lesz. Két, eltérő kiindulási információkat használó szerkesztést is bemutatunk az ellipszisekhez, így az éppen rendelkezésre álló adatok alapján választható ki a megfelelő módszer. Ezután az f egyenest nagytengelynek választva, két olyan általános szerkesztést ismertetünk a kúpszeletekhez, amelyben vagy a fókuszpontjuk, vagy pedig a középpontjuk szabadon megválasztható. Továbbá egy egységes módszert is javaslunk a fókuszpontok és középpontok kiválasztására. Ezek mellett bemutatunk még egy szerkesztést az általános eljárásunkkal megadott kúpszeletek tengelypontjainak a meghatározására.

Speciális ellipszisek

Elsőnek azt a speciális esetet ismertetjük, amikor a kúpszeletek nagytengelyének a d kör azon érintőjét választjuk, amely az f egyenes és d kör $AOB\sphericalangle$ által meghatározott körívének a metszéspontjában található. A 2.17. ábra alapján könnyen belátható, hogy a speciális ellipsziseink középpontjai, az N és H' pontok között helyezkedhetnek el. Azonban mivel az NH' szakasz elhelyezkedését az A és B pontok határozzák meg, ezért a speciális ellipszisek lehetséges középpontjainak helyzete a d kör, két összekötni kívánt pontjának helyzetétől függ. Emellett az is könnyen észrevehető a

2.17. ábra alapján, hogy a meghatározott nagytengely, valamint az A és B pont körön belüli összekötése miatt csak ellipsziseket szerkeszthetünk.

A 2.18. ábra az ellipszis egy olyan jól ismert tulajdonságát illusztrálja (lásd az [5] 60. oldalán), amelyen a következő szerkesztési módszerünk alapul.

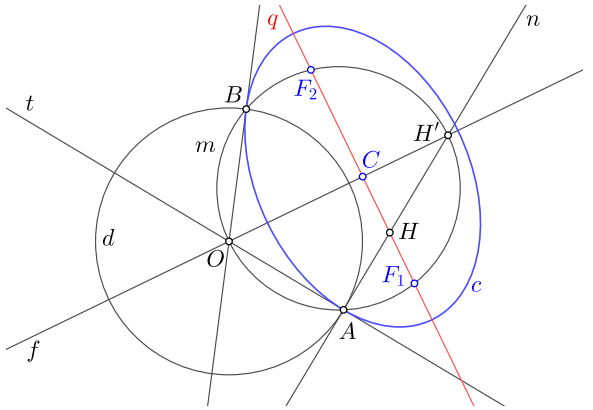


2.18. ábra. Ha az F_1 és F_2 pontok egy ellipszis fókuszpontjai, akkor annak egy tetszőleges A pontjában adott normális felezi az $F_1AF_2\sphericalangle$ szöget, valamint az A-beli érintő felezi az $F_1AF_2\triangle$ külső szögét. Emellett legyenek O és H' pontok az A pontbeli érintőnek és normálisnak az ellipszis melléktengelyével vett metszéspontjai. A segítségükkel látható, hogy az A, F_1, F_2, H', és O pontok konciklikusak.

2.1. Tétel. *Legyen c olyan f egyenesre merőleges q nagytengellyel adott ellipszis, amely a d kört két tetszőleges A és B pontban merőlegesen metszi. Az A, O és B pontok által definiált m körnek a q-val vett F_1 és F_2 metszéspontjai a c ellipszis fókuszpontjai.*

Bizonyítás. A 2.17. ábra már ismert jelöléseit használva, legyen az n és f egyenes metszéspontja H'. A 2.18. ábrán bemutatott tulajdonság alapján az $F_1AF_2\triangle$ köré írt m kör áthalad a H' és O pontokon. Az ellipszisek szimmetrikus tulajdonságait felhasználva pedig a B pont is az m körön helyezkedik el. Így egy m kört rajzolva az A, O és B pontokon keresztül, valamint az m körrel elmetszve a q egyenest kapjuk meg a szerkeszteni kívánt c ellipszis F_1 és F_2 fókuszpontjait. □

Jól látható a 2.19. ábrán, hogy több olyan ellipszis is megszerkeszthető a 2.1. tétel alapján, amely merőlegesen metszi d-t az A és B pontban. Azonban mi egy olyan általánosan alkalmazható megoldást akartunk adni, amely nem függ az NH' szakasz elhelyezkedésétől. Ezért úgy módosítjuk a javasolt megoldásunkat, hogy a d körön rögzítjük az ellipszis C középpontját. Ezzel garantáljuk, hogy bármilyen tetszőlegesen megválasztott az A és B pontpár esetén egyszerűen megszerkeszthetjük a merőlegesen metsző speciális ellipszist.

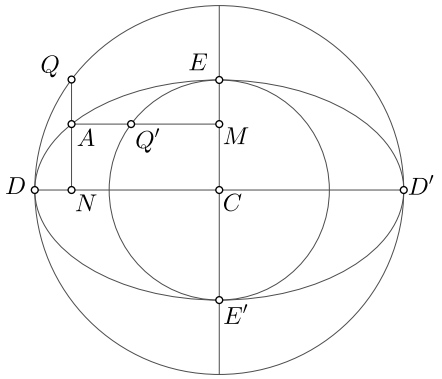


2.19. ábra. A 2.1. tétel felhasználásával megszerkeszthető ellipszisek közül mutatunk be egyet.

A második szerkesztési módszerünkkel ugyanazokat az ellipsziseket szerkeszthetjük meg mint az elsővel, azzal a kitétellel, hogy az ellipszisek C középpontjának a d körön kell elhelyezkednie. Ennél a szerkesztési eljárásnál is egy jól ismert tulajdonságot (lásd az [5] 59. oldalán) használunk fel a fókuszpontok megtalálásához, amelyet a 2.20. ábrán demonstrálunk.

2.2. Tétel. *Legyen c olyan k mellékkörrel, valamint d kört érintő és f-re merőleges q nagytengellyel adott ellipszis, amely a d-t két tetszőleges A és B pontban merőlegesen metszi. Továbbá jelölje F_1 és F_2 az O pontból k-hoz húzott érintők q-val vett metszéspontjait. Ekkor az F_1 és F_2 pontok a c ellipszis fókuszpontjai.*

Bizonyítás. Legyenek D, D' és E, E' a c ellipszis nagy- és kistengelyének a



2.20. ábra. Az ábra jelöléseit használva jól ismert, hogy: $\overline{AN} : \overline{QN} = \overline{EC} : \overline{DC}$ és $\overline{AM} : \overline{Q'M} = \overline{DC} : \overline{EC}$.

tengelypontjai. Emellett jelölje Q_1' és Q_2' az O pontból a c ellipszis k mellékköréhez húzott érintők érintési pontjait. Először c fél kistengelyének a hosszát fogjuk meghatározni a 2.20. ábrán bemutatott, jól ismert tulajdonságot felhasználva.

$$\frac{\overline{AM}}{\overline{Q_1'M}} = \frac{\overline{DC}}{\overline{EC}} \tag{2.8}$$

Jelölje r és b rendre a d kör sugarát és c fél kistengelyének a hosszát. A számítások egyszerűsítésének az érdekében a d kör sugarát 1-nek választjuk meg. A háromszög befogó és magasságtételeit felhasználva határozzuk meg a $\overline{Q_1'M}$ és $\overline{OM}$ hosszakat a $Q_1'CO\triangle$-ben. Az $\overline{AM}$ hosszt pedig az $AMO\triangle$-ben felírható Pitagorasz-tétel segítségével határozzuk meg.

$$\overline{Q_1'M} = b\sqrt{1-b^2} \qquad \overline{AM} = \sqrt{2b^2-b^4} \qquad \overline{OM} = 1-b^2 \tag{2.9}$$

A kapott eredményeket visszahelyettesítve a 2.8 egyenlőségbe kapjuk meg $\overline{DC}$-t.

$$\frac{\overline{AM}}{\overline{Q_1'M}} = \frac{\overline{DC}}{\overline{EC}} \Rightarrow \overline{DC} = \sqrt{\frac{2b^2-b^4}{1-b^2}} \tag{2.10}$$

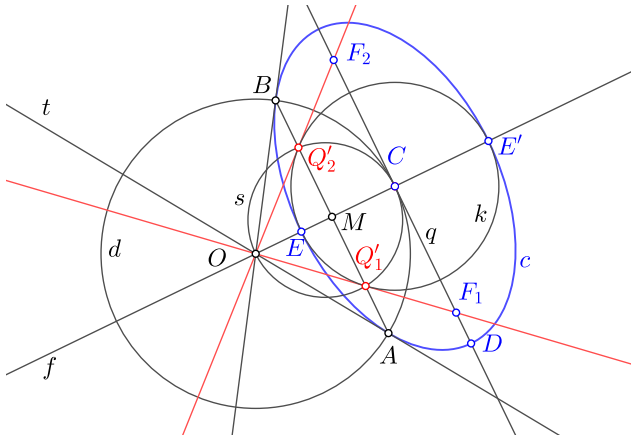
Ezt követően az $F_1CO\triangle$ és $Q_1'MO\triangle$ hasonló háromszöget felhasználva határozzuk meg $\overline{CF_1}$-et. Az $\overline{EF_1}$ hosszt a Pitagorasz-tétel segítségével szá-

moljuk ki a $F_1CE\triangle$-ből.

$$\overline{CF_1} = \frac{b}{\sqrt{1-b^2}} \qquad \overline{EF_1} = \sqrt{\frac{2b^2-b^4}{1-b^2}} \tag{2.11}$$

Láthatjuk hogy mivel $\overline{DC} = \overline{EF_1}$, ezért az F_1 pontnak a c egyik fókuszpontjának kell lennie. A bizonyítás hasonlóan vezethető le az F_2 pontra.

□



2.21. ábra. A 2.2. tétel alapján megszerkeszthető ellipszisek közül mutatunk be egyet.

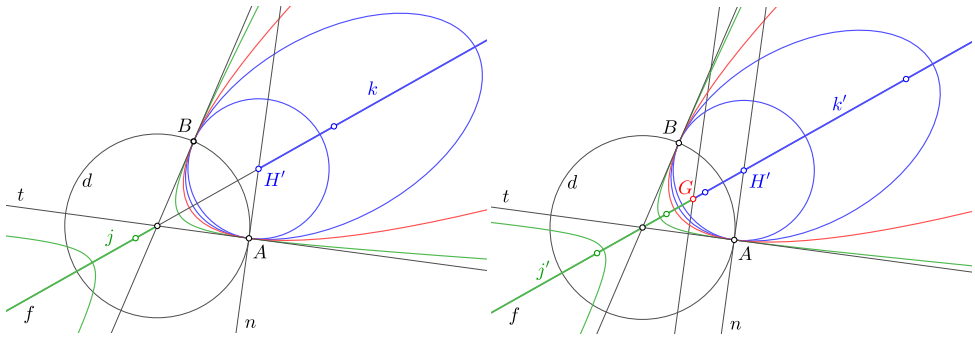
2.2.1. Következmény. *Legyen s egy $\overline{CO}$ átmérővel rendelkező kör, és Q_1' és Q_2' legyenek az s és AB szakasz metszéspontjai (lásd 2.21. ábra). A 2.2. tételben hivatkozott F_1 és F_2 fókuszpontokat a q egyenesnek az $[OQ_1']$ és $[OQ_2']$ egyenesekkel vett metszéspontjaiként szerkeszthetjük meg.*

2.2.2. Következmény. *A 2.21. ábrán látható, és a 2.2. tételen alapuló szerkesztési módszerünket arra is használhatjuk, hogy egy olyan d kört szerkesszünk, amely A és B pontban merőlegesen metszi a c ellipszist, valamint érinti c nagytengelyét. A c ellipszis egyik fókuszpontjából a k mellékköréhez húzott érintők f egyenessel vett metszéspontjai közül az A és B pontokhoz közelebb eső metszéspont lesz a d kör középpontja.*

Általános módszer

A következőkben két módszert fogunk ismertetni olyan kúpszeletek szerkesztésére, amelyek nagytengelye az f egyenes, valamint merőlegesen metszik a d kört két tetszőleges A és B pontban. Azonban mivel sok ilyen kúpszelet definiálható, ezért a módszereink első lépése, a kúpszelet C középpontjának vagy F_1 fókuszpontjának az f egyenesen történő megadása egy kúpszelet kiválasztásához. Ennek eredményeképp, a C vagy F_1 pontok elhelyezése a kiválasztott kúpszelet típusát is meghatározza.

Amint azt a 2.22. ábra is mutatja, az A és B pontok egyértelműen meghatározzák a kört és a parabolát. Emiatt ismerjük a középpontjuk és fókuszpontjuk helyzetét, amelyek segítségével könnyen megszerkeszthetjük őket (lásd 2.1. táblázat). Azonban, ha a C lehelyezésével egy ellipszis vagy hiperbola került kiválasztásra, akkor a szerkesztésükhöz meg kell határoznunk a fókuszpontjaikat. Az ellipszisek esetében a 2.18. ábrán látható tulajdonságot használva találhatjuk meg a fókuszpontokat, míg a hiperbolákra vonatkozó hasonló tulajdonság segítségével (lásd az [5] 98. oldalán) a hiperbolák fókuszpontját kaphatjuk meg. Abban az esetben, ha az F_1 fókuszpont lehelyezésével választottuk ki az ellipszist vagy a hiperbolát, akkor a szerkesztéshez szükséges második fókuszpontot a kúpszeletek optikai tulajdonságát felhasználva kapjuk meg (lásd a [18] 30. oldalán).



(a) Középpont elhelyezése f-en (b) Fókuszpont elhelyezése f-en

2.22. ábra. A középpont vagy fókuszpont f egyenesen történő lehelyezésével megszerkeszthető, különböző típusú kúpszeletekre mutatunk egy-egy példát.

	Középpont elhelyezés	Fókusz elhelyezés
Kör	H'	H'
Parabola	f végtelen távoli pontja	G
Ellipszis	k	k'
Hiperbola	j	j'

2.1. táblázat. A táblázat listázza hogy milyen típusú kúpszeletek készíthetők a szerkesztési eljárásunkkal, ha a C és F_1 pontokat az f egyenes valamelyik felsorolt részén helyezzük el.

A 2.22. ábrán látható, hogy az f különböző részein felvett C vagy F_1 pontokkal eltérő típusú kúpszeleteket kaphatunk. Az f egyenes különböző részeit, valamint az ezeken elhelyezett középponttal vagy fókuszponttal kapható kúpszeletek típusát a 2.1. táblázatban listázzuk.

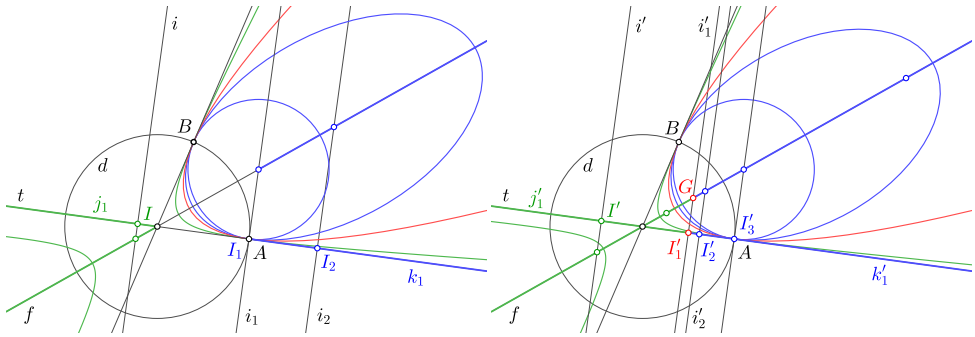
A javasolt szerkesztéseinkkel különböző típusú kúpszelettel is összeköthető két tetszőleges pontja a d körnek. Azonban ha egy jól meghatározott indok nélkül tartalmaz a vizualizáció különböző típusú kúpszeletekből alkotott kapcsolatokat, akkor az könnyen félrevezeti az olvasót, valamint csökkenti a vizualizáció átláthatóságát. Ennek az elkerülésére olyan könnyen használható módszert kell nyújtanunk a C és F_1 pontok elhelyezésére, amely biztosítja, hogy bármely a d kört merőlegesen metsző és f nagytengellyel rendelkező kúpszeletnek azonos a típusa, ha a C középpontjuk vagy F_1 fókuszpontjuk ugyanazon módszerrel lett megválasztva.

A C és F_1 pont, f egyenesen történő legkézenfekvőbb lehelyezése az lenne, ha minden kúpszelet esetében egy előre adott konstans távolságra helyeznénk el őket az O ponttól. Azonban így olyan kúpszeleteket kapnánk eredményül, amelyek nem tartanák meg a C és F_1 pont kiválasztásakor meghatározott típusukat a d körrel vett A és B metszéspontjaik módosításának a hatására. Emiatt ezen kúpszeleteknek a szalagokhoz történő felhasználása olyan különböző típusú kúpszeletekből felépülő kapcsolatokat eredményezne, amelyek félreértéseket okozva csökkentenék a vizualizáció átláthatóságát.

Azt kívánjuk elérni, hogy a C és F pont lehelyezése olyan típusú kúp-

szeleteteket eredményezzen a d kör bármelyik két tetszőlegesen kiválasztott pontjára, amelyeknek azonos a típusa. Ez viszont csak akkor lehetséges, ha a C vagy F_1 pont az f egyenes ugyan azon részében marad, amelyben először elhelyezésre került, függetlenül az A vagy B pont d körön történő mozgatásától.

Ehhez egy tetszőleges I pontot veszünk fel a t egyenesen, amelyben egy i merőlegest állítunk t-re. Az így kapott i egyenesnek az f-el vett metszéspontját használjuk a szerkesztésünk kiindulási C vagy F_1 pontjának (lásd 2.23. ábra).



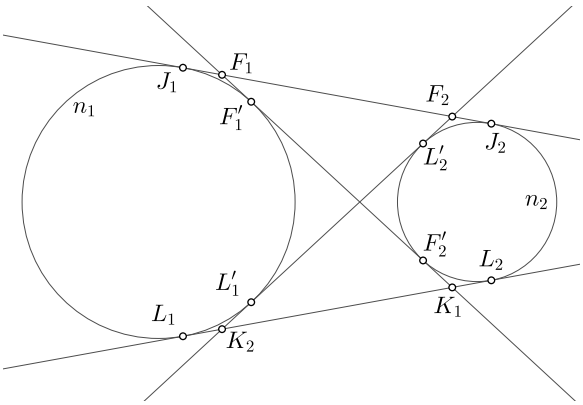
(a) Középpont megadása t-re merőleges egyenessel (b) Fókuszpont megadása t-re merőleges egyenessel

2.23. ábra. A kúpszeletek középpontjának és fókuszpontjának a t-re merőlegesként megadott egyenessel történő elhelyezésére mutatunk példákat, minden kúpszelet típus esetében.

Az f egyenest merőlegesen metsző i egyenesnek a C vagy F_1 pontok elhelyezéséhez történő használata biztosítja, hogy az A és B pont bármilyen tetszőleges módosításának hatására a C vagy F_1 pont mindig az f egyenes előre kiválasztott részén helyezkedjen el. Emiatt a megszerkeszteni kívánt kúpszelet típusát az határozza meg, hogy az I ponttal megadott és t egyenesre merőleges i egyenes hol metszi f-et (lásd 2.23. ábra). Az f egyenes t-re történő merőleges vetítésével áttranszformálhatjuk a kúpszelet típusának megválasztásához definiált részeit f-nek (lásd 2.23. ábra). Az így meghatározott részeit használjuk fel t-nek arra, hogy a kívánt típusú kúpszeletet kapjuk eredményül az I pont elhelyezése során.

A következőkben egy olyan szerkesztési metódust tárgyalunk, amely

segítségével az általános szerkesztési eljárásunkkal készíthető kúpszeletek tengelypontjait adhatjuk meg. Ehhez az F_1 fókuszpontot, A pontot, valamint f és t egyeneseket használjuk fel a 2.24. ábrán látható, két kör tangenseire vonatkozó tétellel (lásd a [20] 96. oldalán).



2.24. ábra. A következő jól ismert tulajdonságot illusztrálja az ábra. A külső érintők hossza, a külső érintők által a belső érintőkből kimetszett távolsággal egyenlő. $\overline{J_1J_2} = \overline{L_1L_2} = \overline{F_1K_1} = \overline{F_2K_2}$. A belső érintők hossza, a belső érintők által a külső érintőkből kimetszett távolsággal egyenlő. $\overline{F_1'F_2'} = \overline{L_1'L_2'} = \overline{F_1F_2} = \overline{K_1K_2}$.

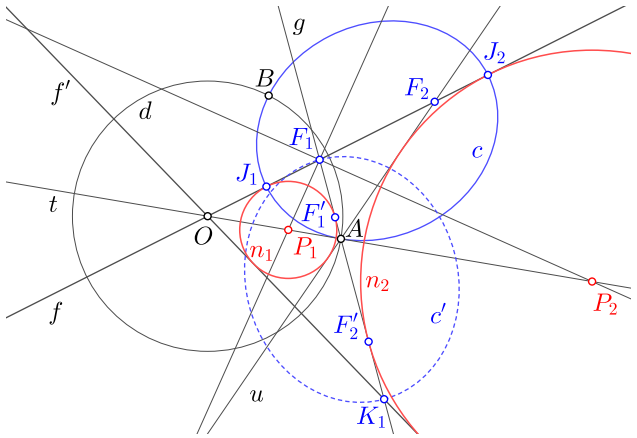
2.3. Tétel. *Jelölje c a d kört két tetszőleges A és B pontban merőlegesen metsző, f nagytengellyel adott kúpszeletet. Legyen g a c kúpszelet F_1 fókuszán és A pontján áthaladó egyenes, valamint legyenek P_1 és P_2 azon pontok, ahol a g és f egyenes szögfelezői metszik a t egyenest. Ha c ellipszis vagy hiperbola, akkor a P_1 és P_2 középponttal rajzolt, g-t érintő n_1 és n_2 körök f-et, a c kúpszelet J_1 és J_2 tengelypontjaiban érintik. Abban az esetben, amikor c parabola, csak a P_1 középpontú n_1 kör létezik, amely amellett, hogy érinti g-t, még az f-et is érinti a parabola J_1 tengelypontjában.*

Bizonyítás. Ha c ellipszis, akkor legyenek az n_1 és n_2 körök g-vel vett érintési pontjai F_1' és F_2' pontok. A 2.25. ábrán látható f' és u egyenesek szintén az n_1 és n_2 körök érintőegyenesei. Az érintők egymással vett metszéspontjai F_1, F_2 és J pontok szintén az 2.25. ábrán láthatók. A 2.24. ábrán

bemutatott, már ismert tulajdonságot felhasználva beláthatjuk az alábbi egyenlőségeket.

$$\overline{F_1'F_2'} = \overline{F_1F_2} \qquad\qquad \overline{F_1K_1} = \overline{J_1J_2} \tag{2.12}$$

Emiatt, az F_1' és F_2' fókuszpontokkal, valamint F_1 és K_1 tengelypontokkal definiált c' ellipszis egybevágó a J_1 és J_2 tengelypontokkal, illetve F_1 és F_2 fókuszpontokkal megadott c ellipszissel. A bemutatott technika minimális módosításokkal a hiperbolák esetének bizonyítására is alkalmazható. Azonban a kör és parabola esetében már más elgondolásra van szükség. Ezen esetek bizonyításához a 2.24. ábrán látható tulajdonság helyett, a külső pontból körhöz húzott érintő szakaszok egyenlőségét kell felhasználni, hogy belássuk a c és c' kúpszeletek egybevágóságát. □



2.25. ábra. Egy példát mutatunk a 2.23. ábrán bemutatott szerkesztésünkkel készített ellipszis tengelypontjainak, valamint kúpjának a megszerkesztésére.

2.3.1. Következmény. *Legyen c egy d kört A és B pontban merőlegesen metsző kúpszelet, amelynek nagytengelye az f egyenes. Ha c ellipszis vagy hiperbola, akkor n_1 és n_2 jelölje azokat a köröket, amelyek meghatározzák a c nagytengelyén elhelyezkedő tengelypontokat. Ha c parabola, akkor csak az n_1 kör határozza meg c tengelypontját az f egyenesen. Legyen t a tengelye és f egy alkotója annak a kúpnak, amelyből a rajzunk síkjára merőleges és g egyenest tartalmazó sík pontosan a c kúpszeletet metszi ki. A kapott kúp Dandelin-gömbjeinek merőleges vetülete az ellipszis és hiperbola esetében az n_1 és n_2 kör, míg a parabola esetében csak az n_1 kör.*

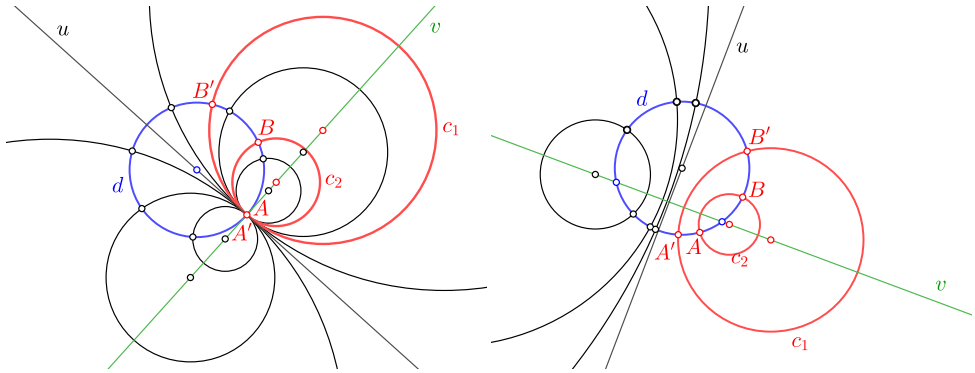
A 2.3.1. következményt felhasználva mutatjuk meg azon kúp merőleges vetületét a 2.25. ábrán, amely tartalmazza a javasolt kúpszeletünket. Ezen kúpnak a 2.25. ábrán látható merőleges vetületét az f és f' egyenesek határolják.

2.3.2. Szalagok kúpszeletekkel történő megadásának vizsgálata

Azt is meg kívántuk vizsgálni, hogy miként viselkednének a javasolt merőlegesen metsző kúpszeleteink ha a TabularVis [47] és Circos [30] alkalmazásokkal készült vizualizációkban látható szalagok készítéséhez kerülnek felhasználásra. A kapcsolatokat reprezentáló szalagokat azonos típusú kúpszeletekből alkottuk meg, hogy elkerüljük a különböző típusú kúpszeletek használatából adódó félreértéseket. Az ilyen módon megalkotott szalagok viselkedését az alapján vizsgáljuk, hogy miként teljesítenek a reprezentált kapcsolatok láthatóságának növelése kapcsán. Először a merőlegesen metsző körök, majd a 2.2. tétellel megadott speciális ellipszisek, és végül a javasolt általános szerkesztési módszerrel definiált kúpszeletekkel létrehozható szalagok viselkedését tárgyaljuk. Az alfejezetben tárgyalt c_1 és c_2 kúpszeletek mind merőlegesen metszik a d kört két-két tetszőleges A, B és A', B' pontban.

Kör

Először a c_1 és c_2 kúpszelet körként történő megadásával hozunk létre szalagokat, amelyek a d kör A, A' és B, B' pontjai által meghatározott köríveket $(\widehat{AA'})$ és $(\widehat{BB'})$ kötik össze (lásd 2.26. ábra). Az így kapott kapcsolatot alkotó c_1 és c_2 körök a 2.26. ábrán látható körsorokat definiálják. Parabolikus körsor eléréséhez úgy kell megválasztani a c_1 és c_2 kört, hogy a d körrel vett két-két metszéspontjuk közül, az egyik azonos legyen (lásd 2.26a. ábra). Ellenkező esetben, ahogyan a 2.26b. ábra is mutatja, elliptikus körsort alkot a c_1 és c_2 kör. A vizualizáció alapkörének számító d kör merőleges a szalag oldalait alkotó c_1 és c_2 körre, emiatt a d kör eleme a c_1 és c_2 kör által definiált körsor konjugált körsorának. Emellett a c_1 és c_2 kör d-n belül elhelyezkedő körívei a d kör által meghatározott Poincaré-féle körmodell egyenesei.



(a) Parabolikus körsor

(b) Elliptikus körsor

2.26. ábra. A c_1 és c_2 kör által alkotott körsorokat mutatjuk be. Az így kapott körsorok konjugált körsorának az egyik eleme a d kör. Az u egyenes ezen körsoroknak a hatványvonala, a v egyenes pedig a körsorok centrálisa.

2.4. Tétel. *Legyenek c_1 és c_2 kúpszeletek a d kört A, B és A', B' tetszőleges pontokban merőlegesen metsző körök. Ha a d kör $\widehat{AA'}$ és $\widehat{BB'}$ körívei közül az egyik tartalmazza a másikat, akkor a c_1 és c_2 kör nem metszi egymást. Továbbá, ha vagy $A = A'$ vagy $B = B'$ egyenlőség fennáll, akkor a c_1 és c_2 kör egyetlen közös pontja a d körön helyezkedik el.*

Bizonyítás. A körsorok ismert tételei (lásd a [20] 377. oldalán) alapján a sík egy pontja vagy a körsor egyetlen elemének, vagy pedig mindegyiknek a pontja. A körsorok definíciójából adódik, hogy a parabolikus körsorba tartozó köröknek egy közös pontja van, míg az elliptikus esetben nincsen közös pontjuk. A 2.26. ábrán is látható módon, a c_1 és c_2 kör parabolikus vagy elliptikus körsort definiál, ha a 2.4. tételben megadott módon szerkesztjük meg őket. A parabolikus vagy elliptikus körsor definíciója pedig biztosítja, hogy vagy csak a d kör egy pontjában, vagy pedig egyáltalán ne metssze egymást a c_1 és c_2 kör. □

A 2.4. tétel alapján elkerülhetjük a szalagok oldalainak metszését a d körön belül, ha körnek választjuk meg a c_1 és c_2 kúpszeletet. Az így eredményül kapott deformációmentes szalagokkal pedig növelhetjük a reprezentált kapcsolatok átláthatóságát a vizualizációban.

Ellipszis

Ebben a részben a 2.2. tétellel definiált speciális ellipsziseinkkel megalkotott szalagok viselkedését fogjuk tárgyalni. Elsőnek azon ellipszisekkel készített kapcsolatokat mutatjuk be, amelyeknek a nagytengelye merőleges az f egyenesre (lásd 2.21. ábra). Ezután az ismertetett általános szerkesztési eljárásunkkal megalkotható ellipszisekkel készített szalagokat fogjuk tárgyalni.

2.1. Lemma. *Legyenek c_1 és c_2 olyan ellipszisek, amelyek tengelyegyenesei azonosak, továbbá legyenek c_1 féltengelyei nagyobbak, mint c_2-é. Ekkor c_1 és c_2 nem metszi egymást.*

Bizonyítás. Legyen c_1 és c_2 olyan origó centrumú ellipszis, amelynek az x és y koordinátatengely rendre a nagy- és kistengelyegyenese. Jelölje a nagy és kis féltengelyek hosszát a c_1 és c_2 esetében rendre $a+c$, $b+d$ és a, b, ahol $c>0$ és $d>0$. Az ellipszisek implicit egyenletei a következők:

$$\frac{x^2}{(a+c)^2}+\frac{y^2}{(b+d)^2}=1, \qquad \frac{x^2}{a^2}+\frac{y^2}{b^2}=1, \tag{2.13}$$

melyekből kifejezve az y koordináták négyzetét:

$$(b+d)^2\left(1-\frac{x^2}{(a+c)^2}\right), \qquad b^2\left(1-\frac{x^2}{a^2}\right). \tag{2.14}$$

Rögzített $-a \leqslant x \leqslant a$ esetén az alábbi egyenlőtlenségek írhatók fel:

$$(b+d)^2 > b^2, \qquad 1-\frac{x^2}{(a+c)^2} > 1-\frac{x^2}{a^2}. \tag{2.15}$$

Így már jól látható, hogy az ellipszisek x szerint rögzített pontjához tartozó y koordináta négyzete c_2 esetén kisebb, mint a c_1 esetén. Ez azt jelenti, hogy c_2 a c_1 ellipszisen belül helyezkedik el. □

2.5. Tétel. *Legyenek c_1 és c_2 kúpszeletek a d kört A, B és A', B' tetszőleges pontokban merőlegesen metsző ellipszisek. Továbbá legyenek c_1 és c_2 tengelyegyenesi azonosak úgy, hogy a nagytengely merőleges az f egyenesre és érinti a d kört (a 2.2. tétel alapján szerkesztett ellipszisek). Ekkor c_1 és c_2 nem metszik egymást.*

Bizonyítás. Jelölje a d kör középpontját O, míg a c_1 és c_2 ellipszisét pedig C. A $COB\sphericalangle$ és $COB'\sphericalangle$ szögek azonosak az $AOC\sphericalangle$ és $A'OC\sphericalangle$ szögekkel, és a továbbiakban α-val és β-val jelöljük őket. A számolásaink egyszerűsítésének érdekében egységnyi sugarú d körrel dolgoztunk.

A c_1 ellipszis fél nagy- és kistengelyének hosszát az α szög segítségével határozzuk meg, majd pedig kiszámoljuk a határértéküket is, ha $\alpha \to 0$. Ehhez először az $\overline{AM}$ hosszt határozzuk meg az $OAM\triangle$-ben, az α és $\overline{OA}=$ $=1$ ismeretében. A $\overline{Q_1'M}$, és $\overline{EC}$ hosszakat pedig a háromszög befogó- és magasságtételét felhasználva számoljuk ki az $OQ_1'C\triangle$-ben.

$$\overline{Q_1M}=\sqrt{\cos\alpha-\cos^2\alpha} \qquad \overline{AM}=\sin\alpha \qquad \overline{EC}=\sqrt{1-\cos\alpha} \tag{2.16}$$

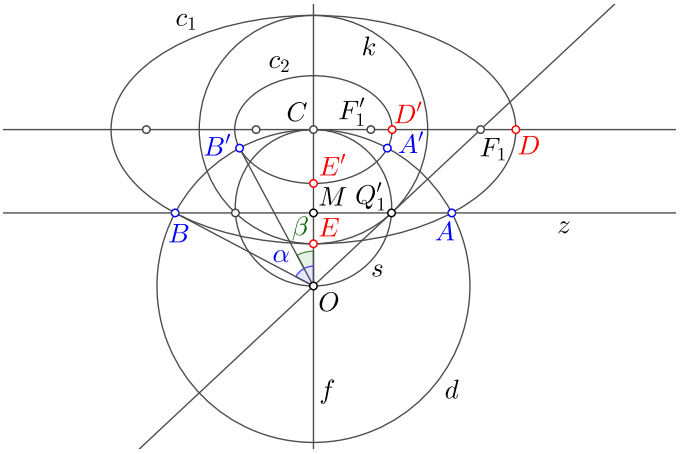
Ezt követően a $\overline{DC}$ hosszt a 2.20. ábrán illusztrált, jól ismert tulajdonság segítségével felírt 2.8. arányt használva határozzuk meg.

$$\overline{DC} = \frac{\sin\alpha}{\sqrt{\cos\alpha}} \tag{2.17}$$

Ha $\alpha \to 0$, akkor:

$$\lim_{\alpha\to 0} \overline{DC} = \lim_{\alpha\to 0} \frac{\sin\alpha}{\sqrt{\cos\alpha}} = 0 \qquad \lim_{\alpha\to 0} \overline{EC} = \lim_{\alpha\to 0} \sqrt{1-\cos\alpha} = 0 \tag{2.18}$$

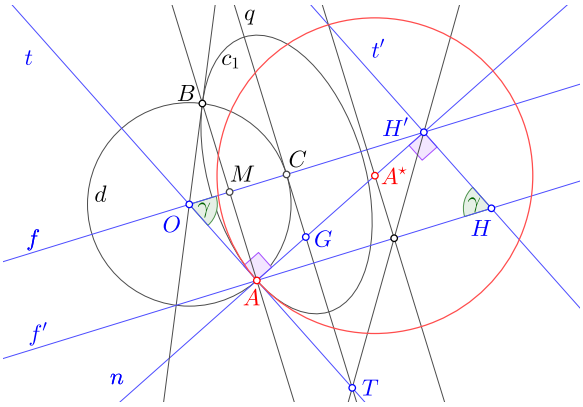
A létrehozni kívánt szalagok miatt B' mindig a c_1 ellipszisen belül helyezkedik el, amelyből pedig következik, hogy $\alpha > \beta$. A $\overline{DC}$-re és $\overline{EC}$-re felírt α szöggel megadott függvények monoton csökkenő tulajdonságát, valamint a belőlük képzett 2.18. határértékeket felhasználva beláthatjuk, hogy a nagyobb α szöggel rendelkező c_1 ellipszis mindig nagyobb fél nagy- és kistengelytávolsággal fog rendelkezni, mint a c_2 kisebb β szöggel rendelkező ellipszis. A 2.1. Lemma értelmében a c_2 ellipszis mindig a c_1 ellipszisen belül fog elhelyezkedni (lásd 2.27. ábra). □



2.27. ábra. A képen láthatjuk, hogy a c_1 ellipszis fél nagy- és kis-tengelyének a hossza nagyobb, mint a c_2 ellipszisé.

Ezt követően meg kívánjuk mutatni, hogy a γ-val jelölt $AOC\sphericalangle$ változtatásának hatására miként változik a görbület mértéke az ellipszis d körrel vett A és B metszéspontjaiban. Ehhez megszerkesztjük az ellipszis A

pontbeli görbületi középpontját ($A^\star$), amely segítségével meghatározhatjuk az ellipszis A pontbeli görbületét (lásd 2.28. ábra).



2.28. ábra. A c_1 ellipszis d körrel vett A metszéspontjához tartozó görbületi középpontra mutatunk egy példát.

2.1. Tulajdonság. Legyenek c_1 és c_2 olyan ellipszisek, amelyek a d kört A, B és A', B' tetszőleges pontokban merőlegesen metszik, továbbá c_1 és c_2 nagytengelye érintse a d kört (a 2.2. tétel alapján szerkesztett ellipszisek). Ha c_1 hosszabb féltengelyekkel rendelkezik, mint c_2, akkor c_2 görbülete az A' és B' pontban nagyobb, mint c_1 A és B pontbeli görbülete.

Bizonyítás. A c_1 ellipszis A pontbeli görbületi középpontja az ebben a pontban vett normálison helyezkedik el, amely a H' pontban metszi az ellipszis kistengelyét. A 2.28. ábrán látható, hogy az $OAH'\triangle$ és az $AHH'\triangle$ egybevágó. Ez alapján megszerkesztve az A pontbeli $A^\star$ görbületi középpontot adódik, hogy $\overline{A^\star H'} = \overline{AG}$. Mivel a CG és AG érintőszakaszok egyenlők, ezért:

$$\overline{AA^\star} = \overline{AH'} - \overline{CG}. \tag{2.19}$$

Hasonlóan a 2.5. tétel bizonyításához, válasszuk a d kör sugarát egységnyinek, így $\overline{AM} = \sin\gamma$ és $\overline{AH'} = \tan\gamma$. Az $MAH'\triangle$-ben alkalmazott Pitagorasz-tétel segítségével határozzuk meg $\overline{MH'}$-t, amelyből pedig kiszá-

molhatjuk a $\overline{CH'}$-t.

$$\overline{MH'} = \frac{\sin^2\gamma}{\cos\gamma} \qquad \overline{CH'} = \frac{1-\cos\gamma}{\cos\gamma} \tag{2.20}$$

Az $AH'M\triangle$ és $GH'C\triangle$ hasonló háromszögeket felhasználva számoljuk ki $\overline{CG}$-t.

$$\overline{CG} = \frac{\sin\gamma}{1+\cos y} \tag{2.21}$$

A $\overline{CG}$, $\overline{AH'}$, valamint a 2.19. egyenlet segítségével meghatározzuk a c_1 ellipszis simulókörének $\overline{AA^\star}$ sugarát.

$$\overline{AA^\star} = \tan\gamma - \frac{\sin\gamma}{1+\cos\gamma} \tag{2.22}$$

Ha $\gamma \to 0$, akkor:

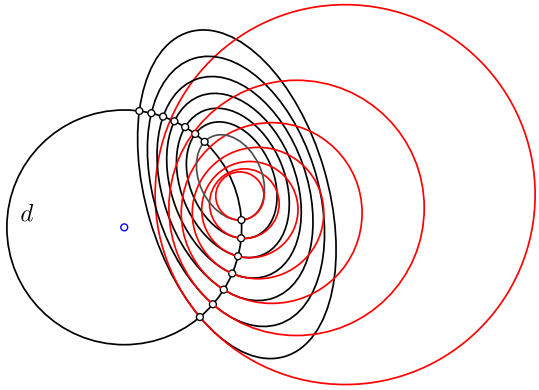
$$\lim_{\gamma\to 0} \overline{AA^\star} = \lim_{\gamma\to 0}\left(\tan\gamma - \frac{\sin\gamma}{1+\cos\gamma}\right) = 0 \tag{2.23}$$

Az $\overline{AA^\star}$-t leíró, γ szöggel megadott függvény monoton csökkenő tulajdonságát, valamint a belőle képzett 2.23. határértéket felhasználva beláthatjuk, hogy egy γ-nál kisebb γ' szöggel rendelkező c_2 ellipszis mindig nagyobb görbülettel rendelkezik a d körrel vett metszéspontjaiban, mint a nagyobb fél nagy- és kistengelytávolsággal rendelkező c_1 ellipszis a d körrel vett metszéspontjaiban (lásd 2.29. ábra).

□

Azt feltételezzük a 2.5. tétel és 2.1. tulajdonság alapján, hogy a 2.2. tétel segítségével megszerkeszthető ellipszisek nem metszik egymást, ha kapcsolatokat reprezentáló szalagok készítésére használjuk őket. Emellett feltételezzük még, hogy a körökhöz hasonlóan ezek az ellipszisek is alkalmasak a kapcsolatok láthatóságának a növelésére.

2.5.1. Sejtés. *Legyenek c_1 és c_2 olyan ellipszisek, amelyek a d kört A, B és A', B' tetszőleges pontokban merőlegesen metszik, továbbá c_1 és c_2 nagytengelye érintse a d kört (a 2.2. tétel alapján szerkesztett ellipszisek). Ha c_1 tartalmazza c_2-nek, vagy c_2 tartalmazza c_1-nek a d körrel vett metszéspontjait, akkor c_1 és c_2 nem metszi egymást.*



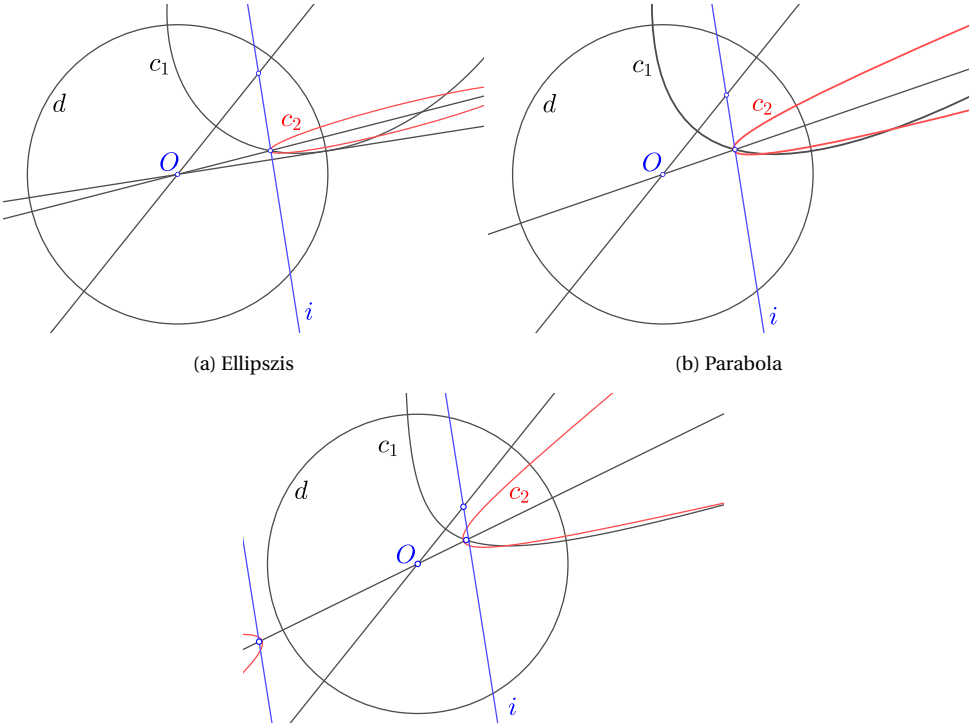
2.29. ábra. A 2.2. tétel alapján szerkesztett, különböző féltengelyekkel rendelkező ellipszisek d körrel vett metszéspontjaiban pirossal megrajzolt simulókörök jól demonstrálják 2.1. tulajdonságot.

A ellipszisek 2.23. ábrán látható módszerében a t érintőre állított i merőleges segítésével határozzuk meg az ellipszis f egyenesen elhelyezkedő középpontját vagy fókuszpontját. Legyen c_1 és c_2 azonos középpont vagy fókuszpont elhelyezéssel, a 2.23. ábrán látható módszerrel megszerkesztett ellipszis. Az így kapott c_1 és c_2 metszhetik egymást, mivel a c_1 fókuszpontját meghatározó i egyenes elmetszi c_2-t (lásd 2.30a. ábra). A c_1 ellipszis középpontját vagy fókuszpontjait definiáló i merőleges egyenes bármilyen megválasztása esetén belátható, hogy i metszeni fogja a c_2 ellipszist.

A bemutatott eredmények alapján a javasolt speciális ellipszisek feltehetően segítenek a kapcsolatok láthatóságának növelésében, míg az általános szerkesztési eljárással megkapható ellipszisek deformált szalagokat eredményezhetnek, ami csökkenti a kapcsolatok olvashatóságát.

Parabola és Hiperbola

A d körön vett két tetszőleges pont egyértelműen meghatározza a javasolt parabolánkat. Hasonlóan az ellipszishez, a parabolák esetében is fel tudjuk használni a fókuszpontot meghatározó i egyenest a parabolák közötti metszés előfordulásának megmutatására. A c_1 parabola fókuszpontját le-



(a) Ellipszis

(b) Parabola

(c) Hiperbola

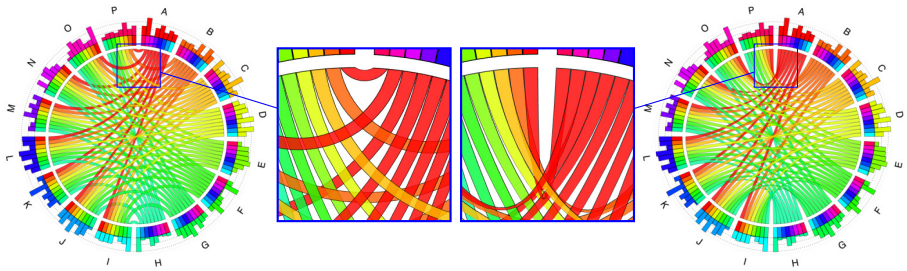
2.30. ábra. A fókuszpontok mozgását leíró egyenesek és bizonyos javasolt kúpszeleteink közötti metszést mutatjuk be. Emiatt ezen kúpszeletekkel készített szalagok használata során nem kerülhető el minden esetben az oldalak metszése.

helyező i mindig metszi a c_2 parabolát. Ez könnyen belátható, mivel a fókuszok mozgását az AO vagy BO szakasz szakaszfelező merőlegese írja le (lásd 2.30b. ábra).

A parabola vagy az ellipszis esetéhez hasonlóan, a hiperbolák metszését is a fókuszpontját meghatározó i egyenes segítségével mutatjuk meg. Az c_1 hiperbolához tartozó egyik fókuszpontot a c_2 hiperbola tartalmazhatja, mivel ezt a fókuszpontot lehelyező i egyenes metszi a c_2 hiperbolát (lásd 2.30c. ábra). Emiatt a parabola vagy hiperbola segítségével készített szalagok oldalai metszhetik egymást és olyan deformálódott kapcsolatokat eredményezhetnek, amelyek csökkenthetik a diagram átláthatóságát.

2.4. Kapcsolatok vizualizációja körökkel

A Bézier görbék lecserélésére javasolt kúpszeletekből, genomi és általános adatokat megjelenítő vizualizációban mutatjuk meg, a 2.4. tétel alapján felhasznált merőlegesen metsző körökkel készíthető szalagokat (lásd 2.31. ábra). A diagramban láthatjuk, hogy a körökkel alkotott kapcsolatok nem fedik el egymást úgy, mint amelyeket Bézier görbék felhasználásával készítettünk. Emellett a szalagok oldalait alkotó körök nem metszik egymást, aminek köszönhetően növelni tudtuk a kapcsolatok láthatóságát. Hasonló eredményeket láthatunk a függelékben elhelyezett, vonalak használatával készített vizualizációban (lásd A.1. ábra)



2.31. ábra. Kör (bal) és Bézier görbe (jobb) használatával alkotott kapcsolatok összehasonlítása általános adatok vizualizációjában.

3. QR-kódok 3D nyomtatással alkotott felületekbe ágyazása

Az információk felületeken történő megjelenítésére javasolt módszerünk QR-kódként ágyazza be azokat a felhasználó által kiválasztott felületrészbe. A megoldásunk elkészítése során fontos szempont volt, hogy megfizethető, hobbi 3D nyomtatók használatával is dekódolható eredményt tudjon nyújtani. Ennek megfelelően a módszereinket is ebbe a kategóriába sorolható nyomtatókon teszteltük (Prusa mini, Prusa i3 MK2.5).

Két metódust dolgoztunk ki, amelyek segítségével paraméteres felületekbe, valamint háromszögelt modellekbe ágyazható a QR-kód. Az így kapott megoldásaink több ponton is hasonlítanak egymásra, így azokat az eltérő részek kivételével közösen ismertetjük.

Hasonlóan az értekezés első részéhez, a QR-kódok beágyazására adott megoldásaink tárgyalása előtt is szeretnénk rövid áttekintést nyújtani a szakirodalomból, valamint a megoldásunk alapjául szolgáló módszerekből.

3.1. Szakirodalmi áttekintés

A vonalkódok egy elterjedt formája a QR-kód, amelynek a felhasználók számára történő vizuális feljavítása és vonzóbbá tétele, egy aktívan kutatott terület [11, 15, 17, 38, 40, 64, 66]. Azonban, önmagában nem elég egy szép és informatív QR-kód, ha egy ívelt felületen történő elhelyezés során megjelenő deformáció megnehezíti vagy lehetetlenné teszi annak a dekódolását. Szerencsére a leggyakoribb ívelt felület, a henger esetében több megoldás is elérhető a problémára [31–33, 35, 36].

A QR-kód nem csak címkék segítségével helyezhető el a tárgyakon, hanem be is ágyazható azok felületébe. Wei és mtsai. [65] az illegális ha-

misítványok elleni küzdelemhez ágyaz be QR-kódokat fém alkatrészekbe. Továbbá, az additív gyártási láncon keresztüli nyomon követés és a 3D nyomtatási támadások megelőzésére szintén használhatjuk a QR-kód komponensekbe történő ágyazását [10]. Ezeken felül pedig a 3D nyomtatott QR-kódok is lehetnek vonzók a felhasználók számára, ha azok Yang és mtsai. [67] javaslata alapján kerülnek elkészítésre.

A Kikuchi és mtsai. [26] által javasolt metódus csak a QR-kód fekete moduljait vési ki az adott B-spline felületből, hogy az így keletkező árnyékot felhasználva beolvasható QR-kódot nyújtson. A modulokat egy iteratív metódussal addig mélyíti, amíg el nem éri a QR-kód beolvasásához szükséges kontrasztot a fekete és fehér modulok között. Ehhez hasonló megoldást ismertetnek Peng és mtsai. [58], a beágyazáshoz szükséges mélység optimalizálásához. Első lépésben a QR-kód módosításával csökkentik az összefüggő fekete moduloknak a számát. Ezt követően pedig szimulációval találják meg a QR-kód nem egyenletes beágyazási mélységét, amiben a modulok csúcspontjainak külön-külön határozzák meg a beolvasáshoz szükséges mélységet. Azonban, a megoldásukhoz csak röviden vázolják fel a QR-kód felületre történő vetítését.

Egy olyan új QR-kód beágyazási módszert ismerhetünk meg az [57] munkában, amely normál megvilágítás mellett csak minimálisan észrevehető beágyazott QR-kódot generál. Az előző módszerekhez hasonlóan szintén a modell felületén hajtanak végre módosításokat, de az így létrehozott QR-kód csak egy adott irányból érkező megvilágítás esetén dekódolható. A QR-kód vetítési irányának meghatározásához a felületi normálvektorok átlagát használja a módszer, azonban a szerzők nem részletezik a mintavételezésükhöz használt eljárását. A módszerrel átlagosan $\approx 50\%$ kisebb mélységet lehet elérni, mint [58]. Azonban, a QR-kód erősen ívelt felületrészekbe ágyazása során nyilvánvaló korlátokba ütközhet a javasolt megoldás, a kiválasztott felületen megjelenő önárnyék, illetve a felület nem folytonos vetülete miatt.

A Screen Space Ambient Occlusion (SSAO) árnyékolási technika először Mittring [42] CryEngine2-vel kapcsolatos munkájában került bemu-

tatásra. Az évek során több ehhez hasonló technikát hoztak létre, a megvilágítás gyorsabb és valósághűbb meghatározásáért [3, 43, 59, 62, 69]. Az SSAO technikák segítségével gyorsan előállíthatjuk a valódi árnyék meggyőző közelítéstét egy adott nézőponthoz. Emiatt, a legtöbb számítógépes játék manapság már valamilyen SSAO technikát is használ, a minél természetesebb érzést nyújtó grafika biztosítására.

3.2. QR-kód felületre történő vetítésének meghatározása

A Peng és mtsai. [57, 58] által ismertetett módszerekben azon a feltételezésen alapul a QR-kódok felületre vetítéséhez szükséges irány meghatározása, hogy általában felülnézetből, merőleges irányból kerül dekódolásra a QR-kód. Ezekben a munkákban két vetítési irányt is definiálnak, azonban egyiket sem határozzák meg részletesen. Kikuchi és mtsai. [26] főkomponens-analízist (PCA) használnak a QR-kód B-spline felületekre történő vetítési irányának a megtalálására. Azonban ez nem minden esetben eredményez dekódolható QR-kódot. Ezért, mind a paraméteres felületek, mind a háromszögelt modellek esetében új vetítési eljárásokat dolgoztunk ki, amelyek megoldást nyújtanak a fenti problémákra.

A QR-kód felületbe ágyazásának első lépése, hogy a felhasználó által kiválasztott felületrészen meghatározzuk a QR-kód pontjait. Ezt egyszerűen meg tudjuk tenni a síkbeli kód felületre történő vetítésével, amelynek az elvégzéséhez ismernünk kell a felületen megadott középpontját és méretét az adott QR-kódnak. A módszerünk ezeket az információkat felhasználva egy, a kapott mérettel megegyező síkot helyez el a felületen meghatározott középpontba, amelynek a felületre vetítésével kapjuk meg a QR-kód beágyazásához szükséges pontokat. A megtalált sík normálvektorjának negáltját használjuk vetítési iránynak.

Az értekezésben két különböző módon leírt felület esetében vizsgáljuk a QR-kódok beágyazását. Az első, paraméteres felületekkel foglalkozó

részben egy módszert ismertetünk a QR-kód felületre vetítéséhez szükséges sík meghatározásához, míg a második, háromszögelt felületeket taglaló részben két módszert mutatunk be a sík megtalálásra.

A részletes ismertetésük során centrális vetítést használtunk, azonban lehetőség van a QR-kód felületen lévő pontjainak párhuzamos vetítéssel történő meghatározására is. Ehhez könnyen módosíthatók a javasolt eljárásaink, mivel csak a használt centrális vetítést kell mindenhol párhuzamosra cserélni. Először a paraméteres felülethez adott metódusunkat mutatjuk be, mivel ezt felhasználva alkottuk meg a háromszögelt felületekre adott megoldásainkat.

3.2.1. Paraméteres felület

Ebben a részben egy olyan metódust mutatunk be, amely a [26, 58] módszerek használata során fellépő problémás esetekre nyújt megoldást. Továbbá, a javasolt módszer elég robusztus ahhoz, hogy más felületeken alkalmazva is dekódolható QR-kódot állítson elő. Először a problémás esetek ismertetésével kezdjük a megoldásunk bemutatását.

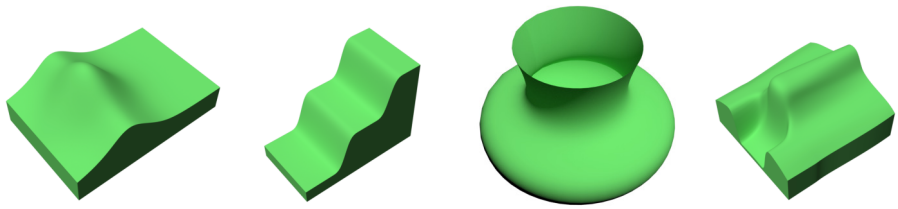
Problémás esetek

A legideálisabb felület QR-kódok elhelyezéséhez sima és egyenletes, mivel az esetükben nem fordul elő deformáció a beágyazott QR-kódban. Azonban, már kisebb torzulások figyelhetők meg a kódban, ha az elhelyezéshez kiválasztott felület enyhén ívelt, csavart, vagy hullámos. Ezeket a legtöbb QR-kódolvasó alkalmazás képes kezelni, és nehézségek nélkül be tudja olvasni a dekódolt információkat. Viszont előfordulhat, hogy a paraméteres felülettel leírt 3D modell nem rendelkezik lapos vagy csak kissé ívelt felületrésszel, amely lehetővé tenné a QR-kód zökkenőmentes beágyazását. Ha ez bekövetkezik, és a QR-kódot ívelt, nagy görbülettel rendelkező felületbe kell ágyazni, akkor olyan deformációs problémák jelenhetnek meg a QR-kódban, amelyek erősen korlátozzák a kód olvashatóságát. Ezt megoldhatjuk azzal, hogy a QR-kód méretét addig csökkent-

jük, amíg nem találunk egy olyan felületrészt, amelybe problémamentesen beágyazhatjuk. Azonban, ekkor fennáll a lehetősége, hogy az eredményül kapott QR-kód túl kicsi lesz a 3D nyomtatáshoz.

Jelölje $S(u,v),(u,v \in [0,1])$ azt a paraméteres formában megadott felületet, amelybe a QR-kódot akarjuk beágyazni. A deformációs probléma akkor fordulhat elő, ha valahol a QR-kód beágyazásához kiválasztott felületrészen belül nagy a felület átlaggörbülete. Azaz, ha két tetszőlegesen kiválasztott, egymáshoz közel elhelyezkedő felületi pontban ($S(u_i,v_i)$, és $S(u_j,v_j)$) vett egységnyi normálvektorok által bezárt szög jelentősen nagy. A felületen 10×10 pontot mintavételezve, az 5°-os eltérés két szomszédos pont egység normálvektorjai között már korlátozhatja a beágyazott QR-kód olvashatóságát.

Négy olyan példát mutatunk be erősen ívelt felületekre a 3.1. ábrán, amelyek esetében deformációk várhatók a QR-kód beágyazása során. A felületekbe ágyazott 3D nyomtatott QR-kódok a 3.20. ábrán tekinthetők meg.



3.1. ábra. Olyan nagy görbülettel rendelkező felületekre mutatunk példákat, amelyek esetében Kikuchi és mtsai. [26], valamint Peng és mtsai. [58] által javasolt QR-kód beágyazási módszerek kevésbé olvasható vagy deformált QR-kódot eredményezhetnek. A legutolsó felületet a [45] autómodell alapján hoztuk létre.

A QR-kód vetítési irányának meghatározása

A QR-kód középpontjának kiválasztása utáni első megválaszolandó kérdés, hogy milyen vetítési módszer segítségével határozzuk meg a QR-kód felületi pontjait. Ez azért fontos, mivel különböző eredményeket fogunk

kapni, ha ugyanazt a QR-kódot a felület ugyanazon pontjában, centrális és párhuzamos vetítéssel szeretnénk beágyazni (lásd 3.2. ábra). Párhuzamos vetítés választása esetén, a QR-kód középpontjától kiinduló, és a szélek irányába növekvő mértékű deformációt figyelhetünk meg. Ez nagyban megnehezíti, vagy akár lehetetlenné teszi az eredményül kapott QR-kód dekódolását. A beolvasáshoz használt kamerának a QR-kódhoz való közelítésével növekszik, távolításával pedig csökken a kialakuló deformáció mértéke. Emiatt előfordulhat, hogy olyan távolra kell mozgatni a kamerát a deformáció elkerülésének érdekében, ahonnan a QR-kód már túl kicsinek látszik a sikeres beolvasáshoz. Ellenben, a QR-kód centrális vetítéssel történő beágyazáshoz egyszerű olyan vetítési centrumot találni, amely a deformációs probléma megjelenése nélkül eredményez könnyen olvasható QR-kódot.

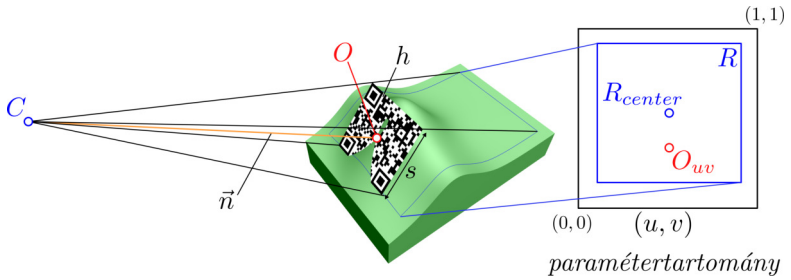
(a) Felület (b) QR-kód

3.2. ábra. A QR-kód beágyazásához használható centrális (bal), valamint párhuzamos (jobb) vetítéssel készült QR-kódok, és azok felületének az összehasonlítása.

A következő lépésben meghatározzuk a QR-kód kezdeti síkját, és a felületre vetjük. Ezután kerül sor a felületi normálvektorok mintavételezésére a vetítés eredményéül kapott felületen, hogy az összegyűjtött vektorok segítségével javítsuk a QR-kód síkjának jelenlegi állását. Ehhez egy olyan iteratív módszert alkottunk, amelyben a mintavételezés során kapott felületi normálvektorokkal minden iterációban folyamatosan módosítjuk a QR-kód síkjának az elhelyezkedését. A módszer által eredményül szolgáltatott sík normálvektorját, a QR-kód középpontját, valamint az ettől a ponttól mért centrum távolságot felhasználva határozzuk meg a centrális vetítést. Az iteráció akkor ér véget, amikor a sík két egymást követő pozíciójának normálvektorjai által bezárt szög kisebb, mint egy előre

meghatározott hibahatár (5°). A gyakorlatban azonban, az első mintavétel utána kapott sík már megfelelő megoldást nyújtott a QR-kód felületre történő vetítéséhez. A további iterációk csak kis mértékben vagy egyáltalán nem javítanának a QR-kód olvashatóságán, azonban magasabb számítási költségeket, és hosszabb futási időt eredményeznének.

A QR-kód felületen kiválasztott középpontjára az O, méretére pedig az s jelölést alkalmazzuk. Első lépésben a módszerünk meghatározza a kezdeti h síkot, amely ortogonális az O pontban vett felszíni normálvektorra. A kapott síkon elhelyezzük a QR-kódot, majd az 3.3. ábrán látható módon a felületre vetítjük.



3.3. ábra. A QR-kód felületre vetítéséhez szükséges sík megtalálásához elsőnek egy kezdeti O középpontú h síkot határozunk meg, amelyhez a képen látható módon helyezkedik el az (u, v) paramétertartományban.

Az O pont $\vec{n}$ egység normálvektorja által definiált egyenesen helyezkedik el a centrális vetítés centruma

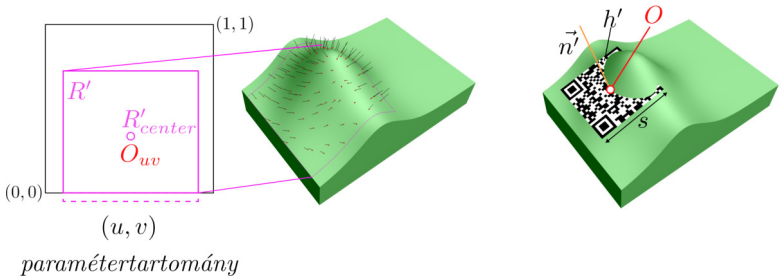
$$C = O + g \cdot \vec{n}, \tag{3.1}$$

ahol g paraméterrel az OC távolságot módosíthatja a felhasználó. Az így levetített sík egy régiót jelöl ki a felületen, amely a paramétertartományában pedig egy négyszögletes R területet eredményez (lásd 3.3. ábra). Az R paramétertartomány,

$$\vec{v} = R_{center} - O_{uv} \tag{3.2}$$

vektorral történő eltolásával egy új R' területet alkottunk, ahol R_{center} az R középpontját, O_{uv} pedig az O pont paramétertartománybeli képét jelöli (lásd 3.4a. ábra). Az R' területet, a 3.4a. ábrán látható módon korlátoztuk

azokban az esetekben, amikor az eltolás eredményül kapott R' túlér a felület paramétertartományának határain.



(a) (b)

3.4. ábra. A vetítési irány megtalálásának folytatásaként az első lépésben meghatározott h sík felületre történő vetítéseként kapott területen mintavételezzük a normálvektorokat (a), hogy meghatározzuk a QR-kód síkjának új helyzetét (b). Az (a) képen, szaggatott vonallal jelöltük az R' felületről lelógó részét.

Az így kapott R' kijelölésen belül végezzük el az $S(u,v)$ felület normálvektorjainak Halton szekvencia [27] használatával történő mintavételezését. A új h' sík

$$\vec{n}' = \frac{\sum_0^m \vec{n}_i}{||\sum_0^m \vec{n}_i||} \tag{3.3}$$

normálvektorját a mintavételezés eredményéül kapott $\vec{n}_i$ vektorokkal határozzuk meg.

A 3.4b. és 3.5. ábrákon mutatjuk be QR-kód moduljainak az új h' síkkal, valamint a

$$C' = O + g \cdot \vec{n}' \tag{3.4}$$

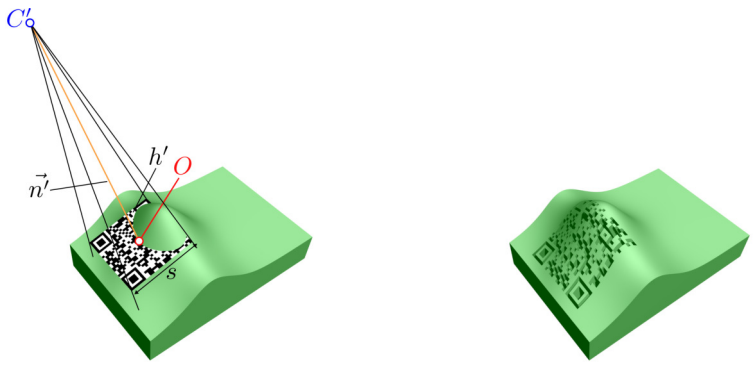
új vetítési centrummal történő felületre vetítését. A QR-kód minden egyes pontjához (Q_{kl}) külön irány tartozik a vetítés centrumából, amelyeket

$$d(k,l) = Q_{kl} - C' \tag{3.5}$$

alakban adtunk meg. A QR-kód felületre történő vetítésének eredményeként előálló

$$P_{kl} = C' + t \cdot d(k,l) \tag{3.6}$$

pontok t paraméterének az értékét a szokásosan alkalmazott Newton-módszer segítségével határoztuk meg.



3.5. ábra. A QR-kód h' síkjának centrális vetítéssel történő felületre vetítése a fekete modulok beágyazásához.

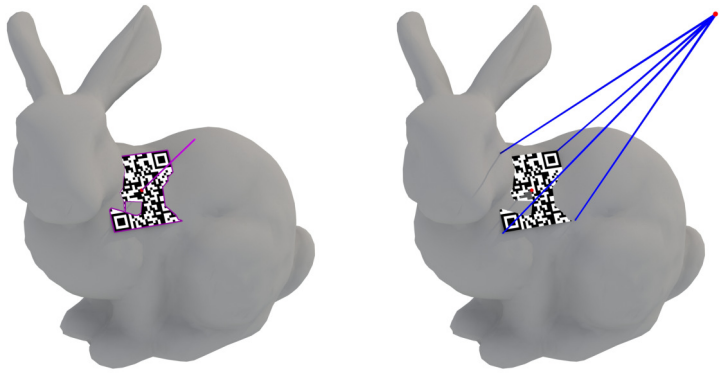
3.2.2. Háromszögelt modell

Két módszert javaslunk a QR-kód vetítési irányának háromszögelt modellekhez történő meghatározására. Mindkettőben a modell felületének normálvektorjait mintavételezzük a megfelelő irány megtalálásához, azonban a második metódusban a modell pontjainak a vetítés centrumától mért távolságát is felhasználjuk. A továbbiakban mélységértékként vagy mélységinformációként fogunk hivatkozni erre a távolságra.

A módszereink automatikusan meghatároznak egy síkot, amelyet a paraméteres felület esetében látott módon használhatunk fel a QR-kód moduljainak a háromszögelt modell felületére történő vetítéséhez. A sík meghatározásához elegendő csak a QR-kód középpontját kijelölni a felületen, mivel a megoldásunk képes a QR-kód méretét is automatikusan meghatározni.

A vetítési irány meghatározásának első lépése egy kezdeti sík definiálása, amely a háromszögelt modellek esetében is ortogonális a QR-kód középpontjában található normálvektorra. A 3.6. ábrán mutatjuk meg az így kapott síkot, valamint annak a modell felületére vetítését.

Az előző, paraméteres felületekhez adott megoldásunk, és a háromszögelt modellek felületéhez ismertetendő módszereink közötti fontos különbségek a mintavételezni kívánt terület meghatározásában, valamint az



(a) Kezdeti sík (b) Centrális vetítés

3.6. ábra. A kezdeti sík elhelyezését, illetve az általa meghatározott centrális vetítést mutatjuk be, amelyhez 25 cm-es centrum távolságot és 5 cm-es oldalhosszúságú QR-kódot használtunk.

ott található normálvektorok mintavételezésében találhatók.

Az új módszereinkben a vetítési irány által kijelölt felületrészt használjuk a mintavételezés során, ellentétben a paraméteres felületekhez ismertetett megoldásunkkal. Továbbá a háromszögelt modellek felületén kiválasztott terület normálvektorjait is különböző módon mintavételezzük az előzőleg ismertetett módszerhez képest. Mindezek mellett pedig azt is megváltoztattuk, hogy miként kezeljük azokat az állapotokat, amikor a mintavételezésre kiválasztott terület modellen kívüli részeket is tartalmaz.

Mindkét új megoldásunknak az első lépése egy kezdeti sík meghatározása, amelyet a modellre vetítve megkapjuk az első tartományt a felület normálvektorjainak mintavételezéséhez. Ehhez a felhasználó által megadott QR-kód középpontot, valamint az ebben a pontban található normálvektort használjuk fel, ahogyan a 3.6. ábrán is látható.

A megfelelő vetítési irány keresése során előfordulhat, hogy az újonnan megtalált sík nem fedi le teljesen a modellt az aktuális vetítési irányból nézve. Ilyenkor a síknak van olyan része, amely kívül esik a modell aktuális vetítési iránnyal meghatározott sziluettjén. Ha így határoznánk meg a modell normálvektorjait az új vetítési irány kiszámításához, ak-

kor az eredményül kapott textúra olyan értékeket is tartalmazna, amelyek nem a modell normálvektorjai. A vetítési irány kiszámításának helytelen normálvektorokat tartalmazó textúrával történő folytatása hibás végeredményt, deformált beágyazott QR-kódot eredményezhet. Ennek az elkerüléséhez lecseréljük a textúra helytelen értékeit, és az új vektorokat tartalmazó textúrával folytatjuk tovább a vetítési irány keresését. Első lépésben, a mélységi adatok segítségével meghatározzuk a textúra azon normálvektorjait, amelyeket le akarunk cserélni. Legyen $T_{n_{ij}}$, a sík modellen kívül eső $h_{i,j}$ pontjának a normálvektorokat tartalmazó T_n textúrabeli képe. Az $T_{n_{ij}}$ normálvektor értékét a 3.7. egyenletben megadott vektorra cseréljük ki, ahol O a QR-kód középpontja, az $\vec{n}$ vektor pedig a h sík normálvektorja.

$$T_{n_{ij}} = h_{ij} - O - \vec{n} \tag{3.7}$$

Az így kapott normálvektorok használatával segíthetünk a sík állását úgy módosítani, hogy annak egy nagyobb része vagy teljes egésze a modell sziluettjén belül helyezkedjen el. Azonban az ismertetett megoldásunk nem garantálja, hogy minden esetben olyan síkot találunk az iteratív megoldásunk végén, amely a modell sziluettjén belül helyezkedik el.

Az első módszerünkben a kezdeti sík felületre vetítésével meghatározott területen mintavételezzük a normálvektorokat, és kiszámoljuk az átlagukat. Azonban nem közvetlenül a felületen végzünk mintavételezést, hanem a vetítés által kiválasztott terület megjelenítése során előállított textúrán hajtjuk végre azt. Ehhez először interpolálni kell a modell normálvektorjait, amit a „smooth” interpolációs minősítő segítségével, perspektíva helyesen végeztünk el a kép raszterizálása során.

A modell megjelenítése közben, a csúcspontok normálvektorjait felhasználva, lineáris interpolációval határozzuk meg a kezdeti vetítési irány által kijelölt felület normálvektorjait, és mentjük az eredményt egy textúrába a későbbi mintavételezéshez. Ezután a Halton szekvencia segítségével m darab mintát veszünk a textúrából, hogy a minták átlagaként kiszámolt vektorra cseréljük ki a kezdeti sík normálvektorját. Az eredményül kapott sík szolgáltatja a QR-kódok beágyazásához szükséges vetítési

irányt.

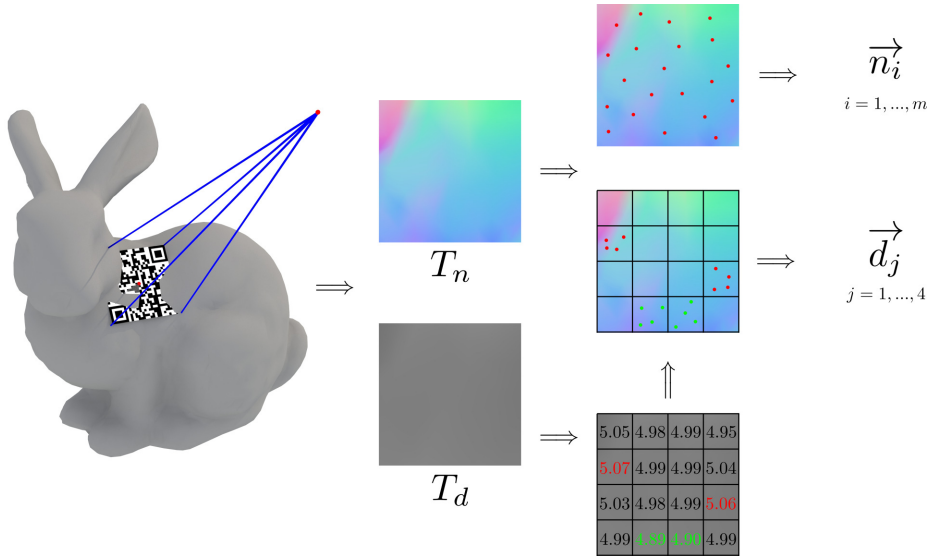
A második metódusunkban egy olyan iteratív módszert használtunk, amely a mintavételezési felület pontjai és a vetítés centruma közötti távolság használatával keresi a megfelelő vetítési irányt. A kezdeti síkot az előző módszerünkben ismertetett módon határozzuk meg, ezért a mintavételezési területük is megegyezik. Azonban, a renderelési folyamat kibővült a mélységértékek meghatározásával és textúrában történő tárolásával. Ennek köszönhetően a második metódusunk már két textúrával T_n és T_d dolgozik, ahol T_n a normálvektorokat, T_d pedig a mélységinformációkat tárolja. Ezekre egy $q \times q$ felosztású rácsot illeszt a módszerünk, amelyet felhasználva kiszámolja a T_d textúra rácsaiban elhelyezkedő mélységértékeknek az átlagát (lásd 3.7. ábra). A sík új normálvektorját ($\vec{n_f}$) több lépésben számoljuk ki. Először, a már bemutatott módszerünknél ismertetett módon mintavételezzük a T_n textúrát, majd ezt követően meghatározzuk az eredményül kapott vektorok átlagát.

$$n_{avg} = \frac{\sum_{i=1}^{m} \vec{n}_i}{\|\sum_{i=1}^{m} \vec{n}_i\|} \tag{3.8}$$

A második lépésben csak azon rácsain végzünk mintavételezést a T_n textúrának, amelyekhez a T_d textúra rácsaiban előzőleg kiszámolt átlag a két legnagyobb, illetve legkisebb volt. Az így kapott rácsokra megismételjük az első lépést, amely a $\vec{d}_j, 1 \leq j \leq 4$ vektorokat eredményezi. Ezeknek az átlagát a következőképpen kapjuk meg.

$$d_{avg} = \frac{\sum_{j=1}^{4} \vec{d}_j}{\|\sum_{j=1}^{4} \vec{d}_j\|} \tag{3.9}$$

Ezen felül szükség lehet egy harmadik lépésre, ha a mintavételezési tartomány meghatározásához használt h síknak van olyan része, amely kívül esik a modell aktuális vetítési irány segítségével meghatározott sziluettjén. Ebben először meghatározzuk azokat a területeket amelyek nem fedik a modellt, majd pedig az előzőekhez hasonlóan mintavételezzük a módosított (lásd 3.7. egyenlet) normálvektorokat. Jelölje l a h sík azon



3.7. ábra. A második módszerünk a normálvektorokat és mélységértékeket tartalmazó textúrákat használja a mintavételezéshez.

részeinek a számát, amelyek kívül esnek a modell sziluettjén. Ekkor az egyes területeken mintavételezett normálvektorok átlagát jelölje $\vec{e}_l$. Ez a lépés összességében hasonló az előzőekhez, csak a mintavételezett területekben különbözik. Ezért hasonlóan kapjuk meg a harmadik lépésben meghatározott vektorok átlagát is.

$$e_{avg} = \frac{\sum_{k=1}^{l} \vec{e}_l}{\|\sum_{k=1}^{l} \vec{e}_l\|} \tag{3.10}$$

A sík új normálvektorját az egyes lépésekben meghatározott normalizált vektorok összegének a normalizáltjaként fogjuk megkapni, és $\vec{n}_f$ vektorként jelöljük.

$$\vec{n}_f = \frac{n_{avg} + d_{avg} + e_{avg}}{\|n_{avg} + d_{avg} + sz_{avg}\|} \tag{3.11}$$

Értelemszerűen, ha nem volt szükség a harmadik lépésre, akkor a 3.11. egyenletből is kihagyjuk, és csak az első két lépésben meghatározott vektorok összegének az átlagával fogunk számolni. A későbbi iterációk mindig az előző iterációban megtalált síkot használják a mintavételezési te-

rület kiválasztásához. A 3.7. ábrán mutatjuk meg a textúrák mintavételezését, valamint az eredményül kapott mintákat. A iteráció akkor áll meg, ha a mélységértékek közötti variancia kisebb, mint egy küszöbérték (0.05) vagy pedig elérjük a maximális iterációk számát (5). Amikor az utóbbi miatt áll meg az iteráció, akkor a legkisebb varianciájú síkkal dolgozunk tovább. Mindkét módszerünkben a 3.1. táblázatban látható textúra paramétereket használtuk a normálvektorok mintavételezéséhez. A használt paramétereket egy 5 × 5 cm nagyságú QR-kódhoz határoztuk meg.

Paraméter	Érték
GL_TEXTURE_MIN_FILTER	GL_NEAREST
GL_TEXTURE_MAG_FILTER	
GL_TEXTURE_WRAP_S	GL_CLAMP_TO_EDGE
GL_TEXTURE_WRAP_T	

3.1. táblázat. A normálvektorok mintavételezéséhez használt textúrák paraméterei.

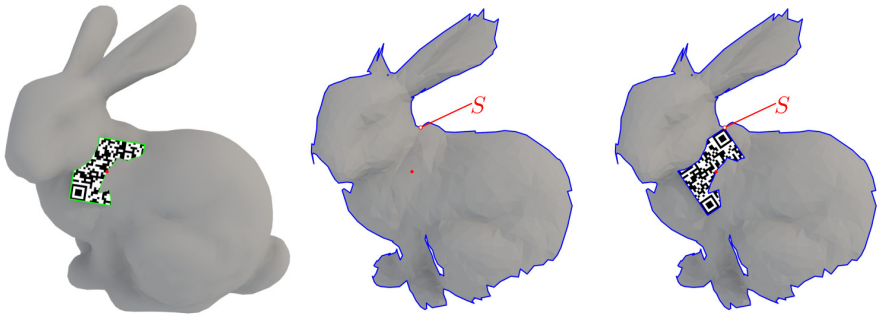
A normálvektorok textúrába mentéséhez lineáris interpolációt használtuk, amely a lapokhoz tartozó csúcspontban megadott normálvektorokból határozza meg a lapokon belüli normálvektorokat. Az általunk használt összes modellhez elérhetőek voltak a csúcspontbeli normálvektorok, így azok kiszámításának a problémáját nem érinti az értekezés, viszont jó irányt jelent a megkezdett kutatás folytatásának.

QR-kód méretének meghatározása

A modell felületén elhelyezni kívánt QR-kód mérete még csak nagyjából sejthető, amikor a felhasználó kijelöli a kód C középpontját. Ezért ebben a részben két módszert is ismertetünk egy olyan méret megtalálásához, amely a QR-kódot aszerint illeszti a modell sziluettjébe, hogy a felhasználó definiált-e preferált orientációt a QR-kód beágyazásához vagy sem. Amíg az első megoldásunkban a felhasználónak csak ki kell jelölnie a QR-kód középpontját a modell felületén, addig a másodiknál még ezen felül meg kell adnia a QR-kód orientációjának az irányát. Mindkét módszerünk

első lépése a QR-kód vetítési irányának megtalálása, amelynek az eredményét a modell sziluettjének a meghatározásához használjuk fel. A segítségével kiválasztjuk az összes olyan háromszöglapot, amely látható a centrális vetítésből és half-edge adatstruktúrát építünk belőlük.

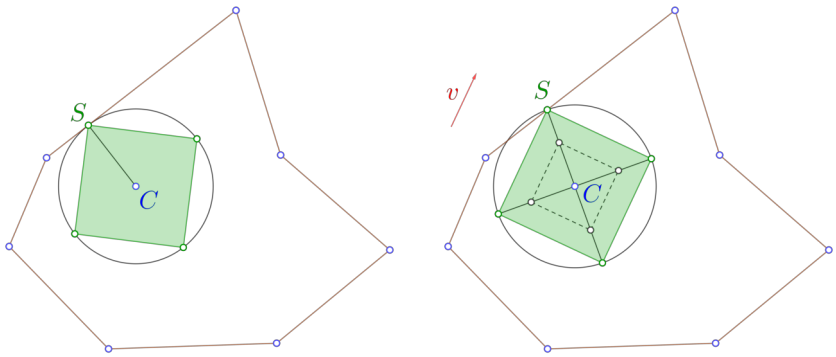
Az első módszerünkben nincs korlátozva a QR-kód orientációja a felhasználó által meghatározott középpont körül, ezért a vetítési irány meghatározása utána kapott sziluettnek a C középponthoz legközelebb eső S pontját keressük. A C pont köré írt, a sziluettet érintő kör segítségével könnyen megtalálhatjuk az S pontot (lásd 3.9a. ábra). A QR-kód méretét az $\overline{SC}$ távolságból kapjuk meg, amely a QR-kódhoz tartozó négyzet átlójának a fele. Minden ennél kisebb QR-kód biztosan a felületbe ágyazható, így ezt a beágyazás legnagyobb garantált méretének nevezzük (lásd 3.8. ábra). Azonban az így kapott garantált méret nem feltétlenül a QR-kód méretének az abszolút felső határa.



3.8. ábra. A modell egy adott QR-kód elhelyezéshez meghatározott sziluettjét, és a QR-kód középpontjához legközelebb eső S pontját mutatjuk meg. Az S ponttal kiszámolt méretnél kisebb QR-kódok biztosan a felületbe ágyazhatók.

A második módszerünkben a QR-kód középpontján felül egy $\vec{v}$ vektort is meg kell határozni a felhasználónak, amellyel megadja, hogy a QR-kód milyen irányba legyen elforgatva. Ehhez azt használjuk fel, hogy a QR-kódokat nagyon gyakran úgy tervezik meg, hogy egy adott $\vec{v}$ iránnyal legyen párhuzamos valamelyik szemközti oldalpárjuk. A sziluett megtalálása után, egy C középpontú, kis méretű, a $\vec{v}$ iránnyal párhuzamos oldalt tartalmazó négyzetet hozunk létre a QR-kód méretének a megtalálásához.

Ezt addig nagyítjuk középpontos hasonlóság segítésével, amíg az egyik csúcsa el nem éri a sziluettet (lásd 3.9b. ábra). A QR-kód mérete az előző esethez hasonlóan számítható ki.



(a) Irány nélkül (b) Irány megadásával

3.9. ábra. A QR-kód beágyazásának legnagyobb garantált méretét, a QR-kód C középpontjának és a sziluett ehhez legközelebbi S pontjának felhasználásával határozzuk meg (bal). Emellett megadhatjuk a keresésnek, hogy a QR-kód két oldala milyen $\vec{v}$ iránnyal legyen párhuzamos (jobb).

A 3.10. ábrán mutatjuk meg, hogy szabadon forgathatók a C pont körül azok a legnagyobb garantált mérettel megegyező vagy annál kisebb méretű QR-kódok, amelyek méretét az első módszerrel határoztuk meg. A 3.10a. ábrán azt az esetet szemléltetjük, amikor a QR-kód egyik csúcsa érinti a sziluettnek azon S pontját, amely a legközelebb helyezkedik el a C középponthoz. A 3.10b. ábrán úgy adtunk meg egy $\vec{v}$ vektort a QR-kód méretének meghatározásához, hogy a 3.10a. ábrán látható QR-kódhoz képest egy 45°-os szögben elforgatott kódot kapjunk eredményül.

3.3. QR-kód beágyazása

A QR-kód beágyazása során a fekete modulok felületbe vésésével [26, 58] alakítjuk ki a dekódolható QR-kódot. Mindkét felületre adott megoldásunkban ez alapján történik a QR-kód beágyazása.

Paraméteres felületek esetében a Kikuchi és mtsai. [26] által megalkotott módszerhez hasonló módon ágyazzuk be a QR-kódot. Azonban eltérő

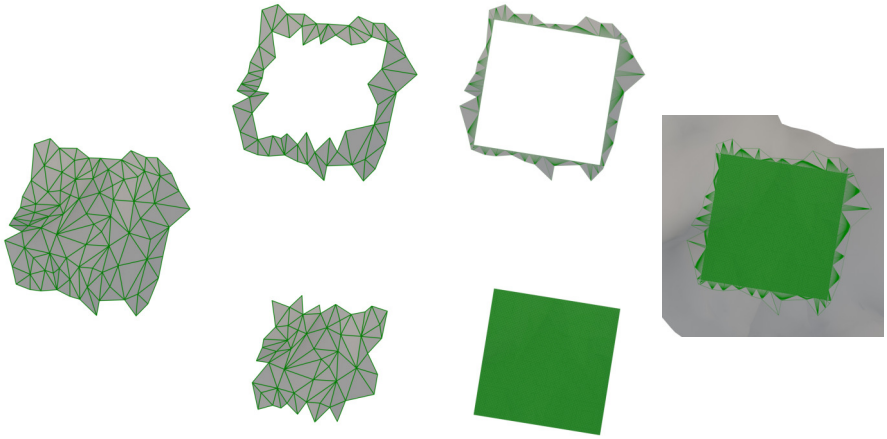
(a) (b)

3.10. ábra. A bal oldali képen, olyan QR-kód látható, amelynek egyik csúcspontja az alakzat sziluettjének a része (a vetítési centrumának nézőpontjából). A jobb oldali képen ennek a QR-kódnak, a C pont körüli elforgatásával kapott kód látható.

megoldást adtunk a QR-kód beágyazáshoz használt pontjainak, valamint a fekete modulok vésési mélységének a meghatározására. Kikuchi és mtsai. egy előfeldolgozási lépésben generálják majd illesztik be a szükséges csomópontokat a QR-kód moduljainak B-spline felületen való kialakításához. Mi ehelyett, a háromszögelt paraméteres felületre vetítve határozzuk meg a QR-kód moduljait. Emellett a beágyazáshoz szükséges mélység kiszámítását is a háromszögeléssel kapott felületen végezzük el a voxelizált felület helyett.

A háromszögelt modellek esetében már teljesen eltértünk a korábbi megoldásoktól, mivel azok nagy számítási kapacitást igényelnek. Peng és mtsai. [57,58] mindkét munkájában újraháromszögelésre kerül a QR-kód középpontja körüli terület, hogy egy sűrűn háromszögelt felületet kapjanak a beágyazáshoz. Ezzel szemben az általunk javasolt módszer előfeldolgozás nélkül ágyazza be a QR-kódot. Egyedül a folyamat utolsó lépéseként módosítjuk a QR-kód határvonalán elhelyezkedő háromszögeit a modellnek, hogy alacsonyan tartsuk a metódusunk számítási költségét.

A QR-kód bemélyesztését már a felületre vetített pontjainak a háromszögelésekor elkezdjük. Ehhez először a kiszámolt centrális vetítés segít-



3.11. ábra. A QR-kód háromszögelt modellek felületébe ágyazásának lépéseit mutatjuk be balról jobbra haladva.

ségével meghatározzuk a QR-kód által érintett háromszöglapokat, majd pedig két csoportra osztjuk a őket aszerint, hogy teljes egészében a centrális vetítés által leírt területen belül helyezkednek-e el vagy sem. A QR-kód moduljait egy 5 × 5-ös méretű rács használatával almodulokra osztottuk, hasonlóan az [58] munkához. Ezeknek a felületre vetített pontjait a belső lapokon végzett sugárkövetéssel határozzuk meg. Ezt követően eltávolítjuk a felhasznált háromszöglapokat, hogy a helyükön elkezdjük kialakítani a QR-kódot, valamint minimálisan a felületbe mélyesztjük a fekete modulok almoduljaihoz tartozó pontokat. Az eredményül kapott pontokból kialakított háromszögelt felület alkotja a mélységkereső metódusunk kezdeti QR-kódját. Utolsó lépésként pedig a külső lapokat háromszögeljük újra a QR-kód oldalainak felületre vetített pontjaival. A 3.11. ábrán szemléltetjük a beágyazás folyamatát.

3.3.1. Beágyazás mélységének meghatározása GPU támogatással

A [26, 57, 58] megoldások használatával időigényes a QR-kód beágyazása, mivel a módszereknek percekig tart megtalálni azt a mélységet, amellyel a QR-kód fekete moduljait a felületbe vésve egy olvasható QR-kódot kapunk. Ha összehasonlítjuk a beágyazási mélységet kereső algoritmus futási idejét a modell 3D nyomtatási idejével, akkor azt láthatjuk, hogy a QR kód beágyazásához szükséges idő csak egy töredéke annak az időnek, amelyre a 3D nyomtatónak van szüksége a modell elkészítéséhez. Emiatt a nagy különbség miatt fontos olyan metódusok létrehozása, amelyek segítségével a QR-kód gyorsan beágyazható, megtekinthető, és ha nem megfelelő akkor egyszerűen kijavítható. Ennek köszönhetően a modell iterációról iterációra javítható a tervezés során, és csak a végső modellt kell kinyomtatni. A javasolt módszerünkkel az a célunk, hogy a felhasználó számára meggyorsítsuk a tervezési folyamatot és könnyebbé tegyük egy olyan beágyazott QR-kód előállítását, amely jó jelölt lehet az időigényes 3D nyomtatáshoz.

Hasonlóan az említett megoldásokhoz [26, 57, 58], mi is egy iteratív módszert adtunk meg, amelyhez az alapján alkottuk meg a fekete modulokban megjelenő önárnyék kiszámítását, hogy a QR-kód felületre vetítéséhez és dekódolásához használt irány általában hasonló egymáshoz.

A „Screen Space Ambient Occlusion” (SSAO) technikák használatával egy adott nézőponthoz gyorsan meg tudjuk határozni az önárnyékot (pl. [59]). Ha a QR-kód beágyazásához használt vetítést adjuk meg az SSAO technika nézőpontjaként, akkor az előző feltevésünk alapján gyorsan kiszámolhatunk egy közelítő értékét a beolvasásnál jelentkező önárnyéknak. Emellett, az SSAO technika használatával a GPU számítási kapacitását is fel tudjuk használni az árnyék meghatározáshoz. Megoldásunk a „Horizon-Based Ambient Occlusion” (HBAO) technikát [3], valamint a beágyazáshoz megadott vetítést felhasználva jeleníti meg a QR-kódot. Az így kapott képet egy textúrában tároljuk el, amelyet később a QR-kód fe-

kete moduljaiban megjelenő önárnyék mértékének a kiértékelésére használunk fel.

A mélységkereső folyamatunk során addig vessük a QR-kód fekete moduljait, amíg azok, valamint a fehér modulok közötti kontraszt nem lesz elég nagy. A renderelés során kiszámolt megvilágítottságát a QR-kódnak egy textúrába mentjük el, hogy az egyes almodulokhoz tartozó képpontokból könnyen meghatározzuk azok átlagos megvilágítottságát. Ennek meghatározására a

$$V(p_x, p_y) = \frac{\sum_{i=p_x}^{p_x+s} \sum_{j=p_y}^{p_y+s} T_{i,j}}{s^2} \tag{3.12}$$

függvényt használjuk, ahol T a textúrát, p_x és p_y az almodul bal alsó sarkának a textúra koordinátáját, s pedig az almodul méretét jelöli. Úgy terveztük meg a módszerünket, hogy a beágyazási mélység alacsonyan tartása mellet bizonyos mértékig azokat az eseteket is kezelni tudja, amikor a QR-kód beágyazásához kiválasztott területen önárnyék jelenik meg. Ennek eléréséhez a fekete modulok megvilágítottságát, azok környezetének az átlagos megvilágítottságával hasonlítjuk össze a megfelelően nagy különbség elérésének vizsgálata során. A lokális módosításnak köszönhetően csak az önárnyékos részen kell a mélységet növelni, ami miatt pedig kisebb mértékben nő meg a QR-kód beágyazáshoz szükséges átlagos mélység. Azonban a beágyazási mélység ilyen módon történő alacsonyan tartása limitálja, hogy milyen sötét önárnyék mellett működőképes a megoldásunk. A QR-kód sikeres beolvasásához nem lehet nagy különbség a fekete modulok megvilágítottsági értékei között, így a megoldásunk csak azon esetekben működik jól, amikor a megjelenő önárnyék nem túl sötét a felület többi részéhez viszonyítva.

A 3.12 függvényt felhasználva határozzuk meg egy $AMV_{k,l,i,j}$ almodul megvilágítottságát, amelyben a k, l paraméterekkel a modult indexeljük, míg az i, j változókkal a modulon belül található almodulokra hivatkozunk. Az indexeket felhasználva meghatározhatók az almodulok p_x és p_y textúra koordinátái a megvilágítottság kiszámolásához.

Az $M_{k,l}$-el jelölt modul megvilágítottságát a benne található almodulokhoz kiszámolt $AMV_{k,l,i,j}$ érték felhasználásával határozzuk meg.

$$MV_{k,l} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} AMV_{k,l,i,j}}{n^2} \tag{3.13}$$

A 3.13 függvényben n-el jelöljük az almodulok meghatározásához használt négyzetes rács sorainak, oszlopainak a számát. Ezt követően egy modul környezetének az átlagos megvilágítottságát $KMV(k,l)$, a négy legsötétebb szomszédjához (8-szomszédsági viszony) kiszámolt megvilágítottság átlagaként kapjuk meg. A fenti számításokat az iteráció elején végezzük el, hogy felhasználhassuk őket az almodulok mélyítését irányító f függvényben.

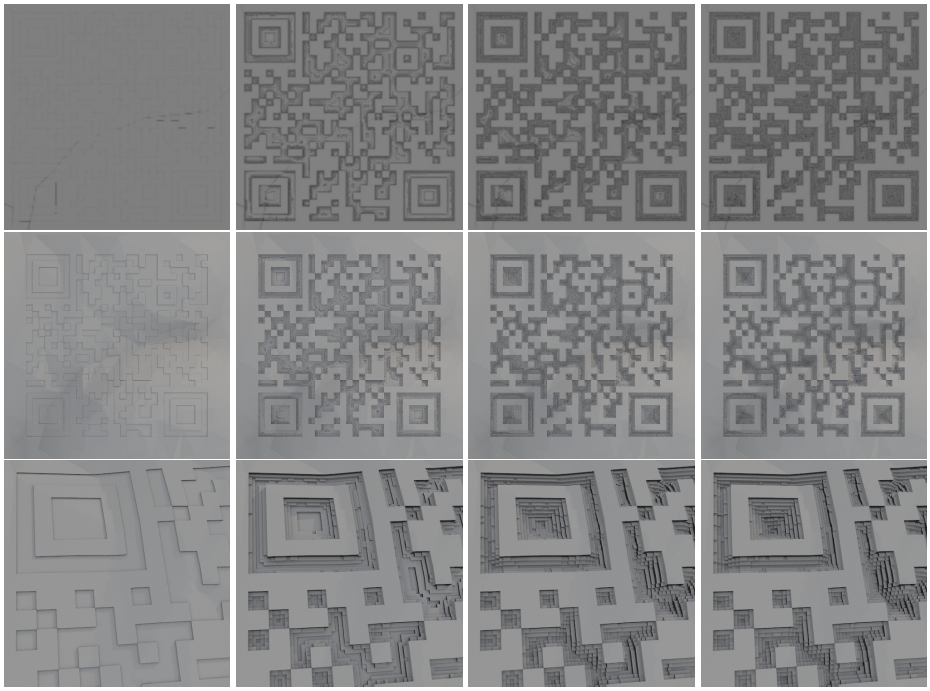
$$f(k,l,i,j) = KMV(k,l) \cdot (-1+c) + AMV_{k,l,i,j} \tag{3.14}$$

A 3.14 függvényben található c paraméterrel a fekete és fehér modulok kontrasztjához adhatjuk meg a kívánt küszöbértéket, amelyet mi hasonlóan 0,2-nek választottunk mint a [57, 58] munkákban. Az $M_{k,l}$ modul $AM_{i,j}$ almoduljának a mélységét akkor növeljük egy iteráció során, ha $f(k,l,i,j) > 0$. Az iteráció megállási feltételében a 3.14 függvény segítségével vizsgáljuk meg, hogy van-e még olyan almodul, amelyet mélyíteni kell, mert ha nincs, akkor megállunk. Ezt az alábbi 3.15 egyenlőtlenséggel ellenőrizzük.

$$\sum_{k=1}^{m}\sum_{l=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n} H\big(G(k,l,i,j)\big) \leq 0 \tag{3.15}$$

A 3.15 egyenlőtlenségben használt m paraméter a QR-kód oszlopaiban és soraiban lévő modulok számát jelenti, míg az n paraméter annak az $n \times n$ négyzetrácsnak a méretét jelenti, amelyet a modulok almodulokra osztására használunk. A 3.12. ábrán látható a módszerünkkel történő beágyazási folyamata egy QR-kódnak.

Ahelyett, hogy minden iterációban állandó értékkel növeljük meg az almodulok mélységét, egy olyan függvény használatát javasoljuk, amely dinamikusan változó értékkel tenné meg mindezt. Ezáltal az egyes iterációkban különböző értéket biztosíthatunk az alapján, hogy milyen régóta



3.12. ábra. Az első sor a mélység kereséshez használt renderelési folyamatunk kimenetét mutatja. Az ezt követő sorokban a módszerünk aktuális iterációjához tartozó beágyazott QR-kód látható.

nem változott meg jelentősen az almodul megvilágítottsága. A beágyazási mélység kontrollálására a tanh() függvényt alkalmazzuk a megoldásunkban.

$$O(x) = a - a \cdot b \left(\tanh\left(\frac{1}{d} - e\right) + \tanh\left(e - \frac{x}{d}\right)\right) \tag{3.16}$$

A 3.16 függvényben használt x változó adja meg egy almodul megvilágítottságában bekövetkező legutolsó szignifikáns változás óta eltelt iterációk számát. Akkor tekintünk egy almodul megvilágítottságban bekövetkezett változást szignifikánsnak, ha az újonnan meghatározott érték legalább 0.001-el kisebb. Tehát, csak abban az esetben növeltük x-et, és ezáltal az almodul beágyazásának növeléséhez használt értéket, ha az almodul megvilágítottsága lényegében nem változott. Azonban nem csak ez az egy érték határozza meg a mélyítés nagyságát. Emellett több paramétert is használunk annak a szabályozására, hogy milyen agresszíven történjen

a mélység keresésének folyamata. A beágyazás alapértelmezett értékét az a paraméter segítségével adjuk meg. Ennek a paraméternek 0,1 értéket választottunk, ami egyben a használt 3D nyomtató egyik rétegvastagsága. A további d, e, és f paramétereket a tanh() függvény transzformációjához használtuk. A segítségükkel úgy alakítottuk ki az $O(x)$ függvény alakját, hogy a jelentős változás óta eltelt iterációk számát jelölő x változóra adott legkisebb érték, az átlagos 0,4 mm átmerőjű fúvókával nyomtatható egyik legkisebb rétegvastagság ($0{,}1mm$), a legnagyobb pedig ~ 1,27 mm körüli érték lehet. A d, b, és e paraméterekhez választott értékeink: $d = 2$, $b = 3$ és $e = 2{,}5$.

A HBAO technika csak egy közelítő értéket tud adni a valódi árnyékhoz. Ennek a segítségével azonban megfelelően tudjuk becsülni a modulok és almodulok beágyazási mélységét, és percek helyett 5–10 másodperc alatt meghatározhatjuk azt. A HBAO technika paramétereit azonban a vésési folyamat előtt meg kell határozni, hogy a renderelés során minél jobb becslést tudjunk kapni.

3.3.2. Beágyazási mélység csökkentése minták használatával

A QR-kód beágyazásához megtalált mélység nagyban befolyásolja a kinyomtatott kód olvashatóságát, valamint a beágyazásra kiválasztható területet. Ezért két olyan megoldást ismertetünk, amelyek minták alkalmazásával érnek el több árnyékot a bemélyesztett QR-kódban, valamint ezáltal csökkentik a beágyazáshoz szükséges mélységet. Mindkét módszerünkben a QR-kód moduljait érintő módosításokat javaslunk. Az elsőben azok alján alakítunk ki mintázatokat, míg a másodikban teljesen megváltoztatjuk az alakjukat.

A javasolt mintáink használatával változatos irányú normálvektorokból álló felület alakítható ki a beágyazott QR-kódok alján, az eddigi sík felület helyett. Egy QR-kód bevésett fekete moduljában megadott P pont megvilágítottsága, az abból kibocsátott $\vec{r}_i$ sugarak hosszának a segítségé-

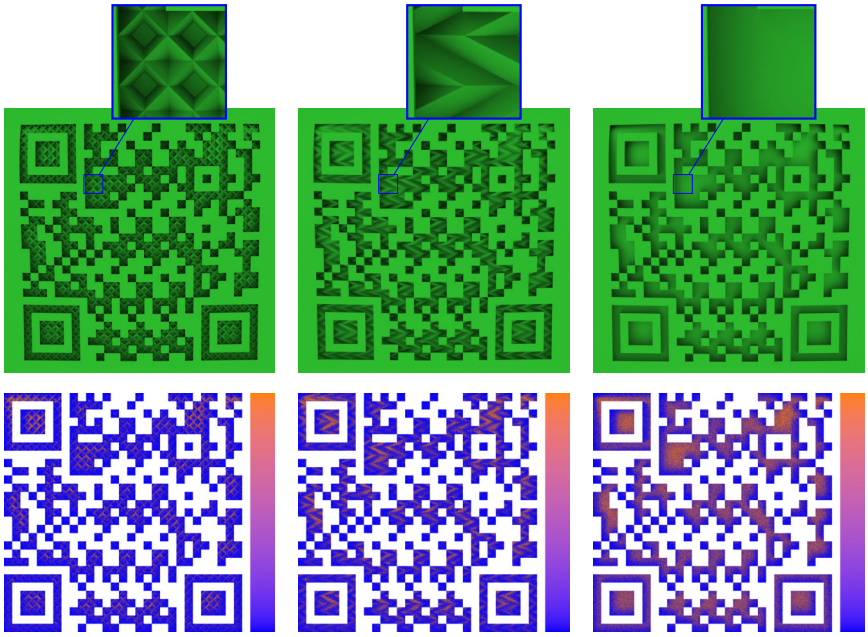
(a) Felülnézet (b) Oldalnézet

3.13. ábra. A P pont megvilágítottságának kiszámolására indított $\vec{r}_i$ sugarak, a P pont $\vec{n}$ felületi normálvektorja által definiált félgömbben.

vel becsülhető meg (lásd 3.13. ábra). Ezeknek a $\vec{r}_i$ sugaraknak az irányát nagyban befolyásolja a P pont felületi normálvektorja, mivel az definiálja a félgömböt, amiben a sugarak elhelyezkedhetnek. Emiatt, ha a P pontot tartalmazó felület úgy került megválasztásra, hogy az azt körülvevő beágyazott QR-kód blokkolja az $\vec{r}_i$ sugarakat, akkor a P pont megvilágítottsága csökkenni fog. Minél több sugár útját tudjuk blokkolni a mintáinkkal kialakított felülettel, annál sötétebbek lehetnek a QR-kód beágyazott részei. Emellett a QR-kód összefüggő fekete moduljai által közösen alkotott terület korlátozásával szintén csökkenteni tudjuk egy adott P pontból kiinduló $\vec{r}_i$ sugarak hosszát, hogy sötétebb felületet érjünk el.

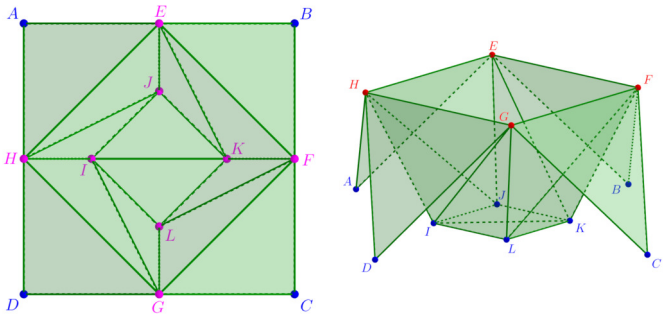
Az első megoldásunkban nem osztjuk fel a QR-kód moduljait kisebb almodulokra, valamint azok négyzetes formáját is érintetlenül hagyjuk. Azonban a fekete modulok bemélyesztett részét lecseréljük a minták alapján kialakított felületekre. Két különböző mintát javasolunk egyenetlen felületetek kialakítására a beágyazott modulok alján, amelyeket a 3.14. ábrán mutatunk meg és hasonlítunk össze.

A 3.15. ábrán látható mintát úgy alkottuk meg, hogy az általa leírt felület minden fekete modulban azonos legyen. Ehhez nyolc további pontot (E, F, G, H, I, J, K, L) definiáltunk a modulhoz. Az első négy pontot a modul oldalainak középpontjaként kapjuk meg. Az I pont, a modul aljának középpontját, a H ponttal összekötő szakasz felezőpontja. A I-hez hasonló módon határozhatjuk meg a K, L és J pontok helyzetét. Ezt követően az E, F, G, és H pontokat ortogonálisan kiemeljük a modul aktuális mély-



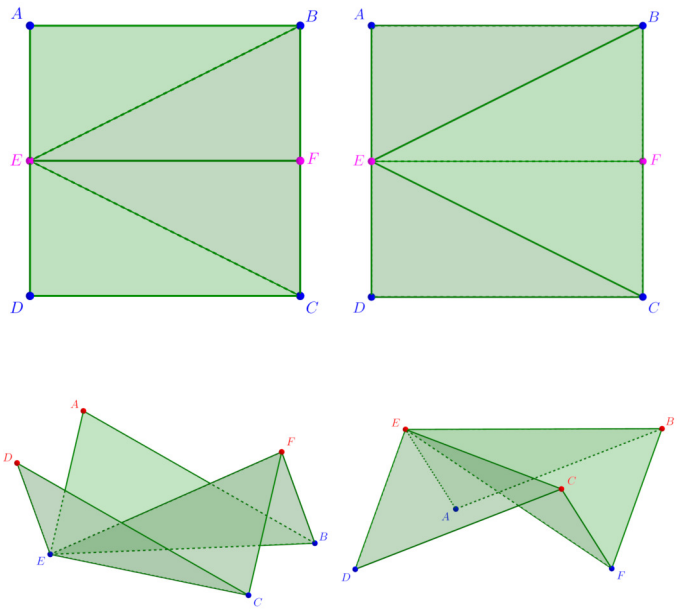
3.14. ábra. Az első két képen az árnyék mértékének növelésre javasolt mintáinkkal, míg a legutolsón egy minta nélkül beágyazott QR-kódot mutatunk meg. Alattuk, az összehasonlításukhoz kiszámolt [26] megvilágítottsága látható a QR-kódoknak. Kékkel a sötét, narancssárgával pedig a világos területeket jelöltük rajtuk.

ségének a feléig. Az így kapott csúcspontok helyzete és azok háromszögelése a 3.15. ábrán látható.



3.15. ábra. A modulok alján azonos felületet definiáló mintához szükséges új pontok (rózsaszín), illetve a pirossal jelölt pontok kiemelése után kapott felület látható.

A 3.16. ábrán két hasonló felületet mutatunk be, amelyeket felváltva használunk a QR-kód moduljainak a soraiban. A páros sorokban 180°-al elforgatva helyezzük el a minták által definiált felületet, hogy elérjük a 3.14. ábrán látható hullámos mintázatot. A felülethez csak két további pontot (E és F) szükséges beillesztünk (lásd 3.16. ábra), amelyek az AD és BC szakaszok felezőpontjaiként kerültek definiálásra. A minta első felületének megalkotásához, az A, D, és F pontokat emeljük fel a modul aktuális mélységének a feléig (lásd 3.16a. ábra), míg a második felület kialakításához a B, C, és E pontokat mozgatjuk hasonló módon (lásd 3.16b. ábra). Az új pontok és azok háromszögelésének az eredménye a 3.16. ábrán látható.



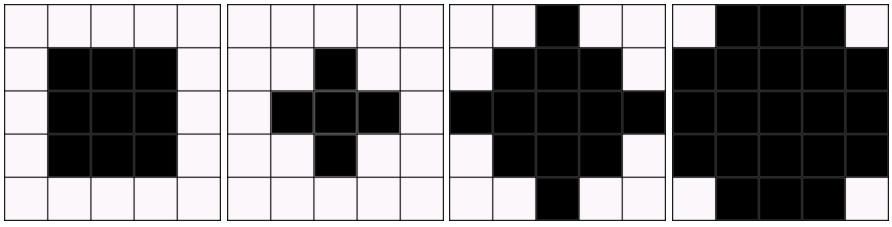
(a) (b)

3.16. ábra. A két fő részből álló, hullámos felületet eredményező minta új pontjai (rózsaszín), valamint a pirossal jelölt pontok kiemelése után kapott háromszögelt felületek láthatók.

A síkbeli QR-kódok esetében gyakran módosítják a fekete modulok alakját azért hogy a QR-kód művészibb, valamint tetszetősebb megjele-

néssel bírjon. Azonban, mi azért javasoljuk a fekete modulok négyzet alakjának lecserélését a második megoldásunkban, hogy csökkentsük a beágyazáshoz szükséges mélységét a QR-kódnak. Ehhez úgy alakítottuk ki a modulok új formáit, hogy könnyen szabályozhassuk velük az egymáshoz csatlakozó fekete modulok számát, valamint az együtt alkotott területük méretét.

Két különböző megközelítést alkalmaztunk, amelyek közül az elsőben minden fekete modult úgy alakítunk át, hogy az eredményül kapott formája kisebb legyen, mint a modul eredeti négyzet alakja. Így, az ezeket leíró minták használatával elvágjuk a modulok közötti összeköttetést. A második megközelítésünkben úgy terveztük meg a modulokat helyettesítő alakzatainkat, hogy megtartsák a szomszédos fekete modulok közötti csatlakozásokat, azonban csökkentsék azok méretét. A 3.17. ábrán mutatjuk be az összes javasolt mintánkat a modulok formájának módosítására. A 3.18. ábrán látható QR-kódok beágyazott moduljait úgy hoztuk létre, hogy a fekete moduloknak csak azon almoduljait mélyesztettük be, amelyeket a mintákban feketével jelöltünk.

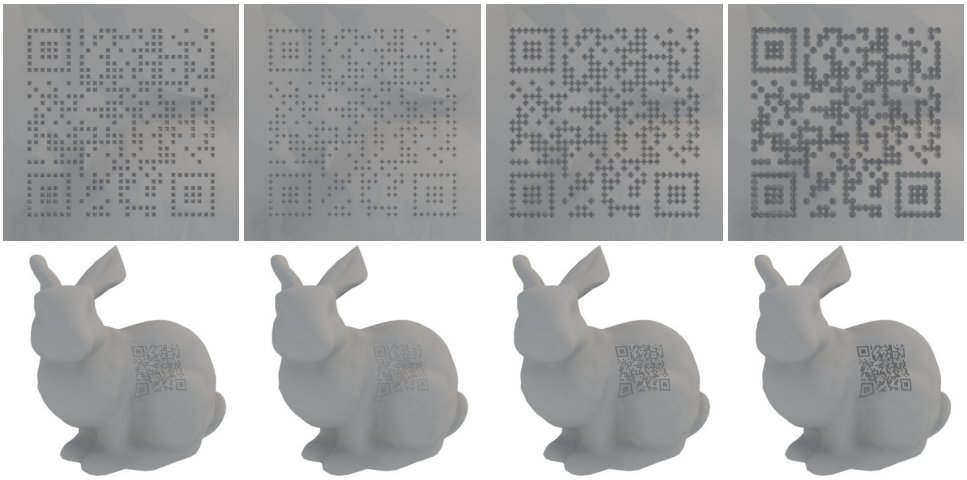


(a) Négyzet (b) Vékony plusz jel (c) Elforgatott négyzet (d) Vastag plusz jel

3.17. ábra. A modulok alakzatának módosítására javasolt mintáinkat mutatujuk be, amelyekben a fekete négyzetek jelölik a bemélyeszthető almodulokat.

Az első két mintázatunk elválasztja egymástól a modulokat, és megszünteti azokat a kapcsolódó modulok által alkotott nagy és egybefüggő területeket, amelyek növelik a QR-kód beágyazásának a mélységét (lásd 3.17a. és 3.17b. ábrát). Az első mintában csak a középső almodulokat engedélyeztük beágyazni. A másodikban pedig még tovább csökkentettük az engedélyezett modulok számát azáltal, hogy eltávolítottuk az előző min-

tának a sarkait, és csak egy vékony plusz jelet hagytunk meg a modul közepén.

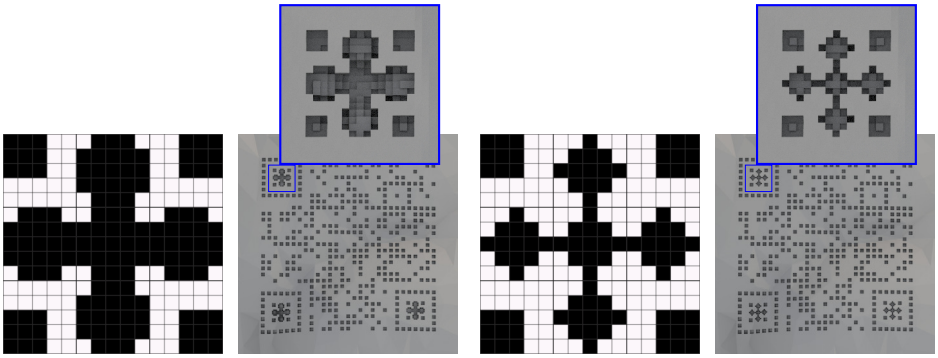


(a) Négyzet (b) Vékony plusz jel (c) Elforgatott négyzet (d) Vastag plusz jel

3.18. ábra. Beágyazott QR-kódok a javasolt mintáink használatával. Csak az alkalmazott minta által engedélyezett almodulok kerültek bemélyesztésre.

A következő két mintánk éppen az előző kettő ellenkezője, mivel megtartják a modulok közötti kapcsolatokat (lásd 3.17c. és 3.17d). A sarokban található almodulokon kívűl mindegyiket engedélyezzük a mintában, hogy a beágyazás során akadályokat képezzünk a nagy és egybefüggő területeken. A harmadik mintánk egy nagyobb, 45°-ban elforgatott négyzetet ábrázol. Az utolsó pedig hasonlóan lett kialakítva a második mintához, mivel a négyzetrács belső soraiban és oszlopaiban lévő almodulokat engedélyeztük benne.

A QR-kód lokalizációs mintái fontos szerepet játszanak a kód felismerésében. Ezért két különböző mintát javaslunk a bemélyesztésükre, amelyeket a fekete modulokhoz definiált minták kombinációjaként alkottunk meg. A 3.19. ábrán láthatók a QR-kód beágyazásához javasolt lokalizációs mintáink, amelyeknek a négy sarokmoduljában, egy kissé eltolt négyzetet helyeztünk el. Emellett egy példát is mutatunk a minták használatára.



(a) Vastag plusz jel (b) Elforgatott négyzet

3.19. ábra. A QR-kód beágyazásához javasolt lokalizációs mintáinak. Először a bemélyesztéshez engedélyezett modulokat mutatunk meg, majd pedig egy példát adunk a minták használatára.

3.4. Összehasonlítás

Mindkét megoldásunkhoz Dart programozási nyelven készítettünk egy webalkalmazást, amelyben WebGL alapú megjelenítést használtunk. Az alkalmazásainkat egy Intel Core i7-8550U processzorral (8M Cache, akár 4,00 GHz), 16 GB memóriával, valamint Nvidia Geforce MX150 grafikus kártyával rendelkező laptopon teszteltük. A modellek elkészítéséhez két 3D nyomtatót használtunk. A paraméteres felületeket a Prusa i3 MK2.5 segítségével, míg a háromszögelt modellek nyomtatásához a Prusa Mini 3D nyomtatót használtuk. A különböző gépek használatának köszönhetően két, eltérő árkategóriát képviselő 3D nyomtatóval tesztelhettük a beágyazott QR-kódok nyomtatásának a minőségét. De nem csak a nyomtatók különböznek egymástól, mivel az utóbbi esetében két különböző átmérőjű fúvókát használtunk, egy 0,4 és egy 0,25 mm átmerőjűt. A paraméteres felületek nyomtatásához csak a 0,4 mm átmérővel rendelkező fúvókát használtuk.

A beágyazott QR-kódok mérete mindkét megoldásunkban nagyban függ a bemutatott minták használatától. A modulok alján kialakítandó felületeket leíró mintáink esetében, a QR-kód méretét úgy kell megválasz-

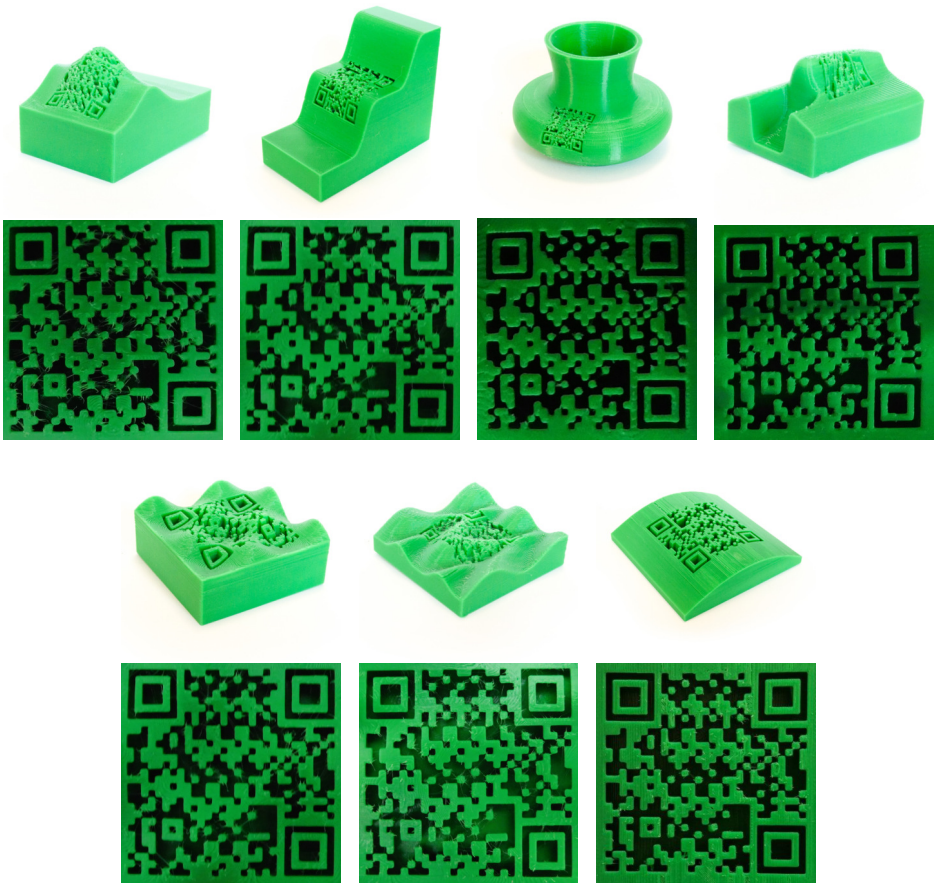
tani, hogy a beágyazott QR-kódban azután is jól kivehetőek legyenek a minták által leírt felületek, miután elvégeztük a modell 3D nyomtatásához szükséges szeletelését. A modellekbe ágyazott QR-kód esetében 1,2 mm-es modulméretet használtunk azok elkészítéséhez. Azonban, bizonyos esetekben ettől az értéktől eltértünk a módszerünk használhatóságának vizsgálata során. Ezekben külön feltüntettük az aktuálisan használt fúvóka átmérőjét, valamint a QR-kód moduljainak méretét. Az értekezésben látható összes modellt politejsav (PLA) műanyag szál használatával nyomtattuk ki, mivel ez az egyik legnépszerűbb és legköltséghatékonyabb alapanyag.

3.4.1. Paraméteres felület

Módszerünket először a problémás, majd pedig három általános felületen teszteltük, hogy bizonyítsuk annak robusztusságát. Amint azt a 3.20. ábra mutatja, módszerünk, nem csak erősen ívelt felületekkel használható jól, hanem kevésbé ívelt vagy hullámos felületek esetében is. A nagy átlaggörbülettel rendelkező felületrészeken a QR-kód középpontját az erősen ívelt területek közelében helyeztük el, a többi felületrészen pedig azok középpontjába pozicionáltuk a QR-kódot.

A beágyazott QR-kódok olvashatóságát Android és iOS okostelefonok segítségével hasonlítottuk össze. Tesztünk eredménye azt mutatta, hogy az iPhone saját kamera alkalmazása jobban teljesít a beágyazott QR-kódok olvasásakor, mint az Androidos alkalmazások (QRbot, QR & Barcode Scanner). Azonban, egy esetet leszámítva, az iOS és Androidos alkalmazások is kevesebb, mint 5 másodperc alatt beolvasták a módszerünk által készített összes QR-kódot (lásd 3.2. táblázat). Csak az iPhone tudta beolvasni azt a QR-kódot, amelyiknél a vetítés centrumát közel helyeztük el a felülethez. Viszont ehhez több percig kellett próbálkozni.

A problémás felületekhez (domb, lépcső, váza és autó) kapcsolódó olvashatósági teszteket a 3.2. táblázatban ismertetjük. Először a javasolt megoldásunk olvashatóságát hasonlítottuk össze a [26, 58] módszerek-



3.20. ábra. A paraméteres felületekre adott módszerünkkel beágyazott QR-kódok 3D nyomtatóval készített modelljei. A felső képeken mutatjuk a teljes felületet, alatta pedig a beágyazott QR-kódot. Az első négy képpáron azt mutatjuk meg, hogy a módszerünk problémamentesen képes QR-kódokat beágyazni erősen ívelt felületekbe. A fennmaradó három felületen pedig azt, hogy a módszerünk robusztus, és a QR-kódot hullámos vagy kevésbé ívelt felületekbe is olvashatón ágyazza be.

kel. Három kategóriát használtunk az elkészült QR-kód olvashatóságának mérésére: a „Könnyű” a „Nehéz” és a „Sikertelen” rövidítés a "Könnyen beolvasható", a "Nehezen beolvasható" (több percnyi próbálkozás) és a "Sikertelen beolvasás" kifejezésre hivatkozik. Ezután, a vetítési centrum különböző távolságainak a beágyazott QR-kódok olvashatóságára gyakorolt hatását vizsgáltuk.

Modell	Telefon op. rend.	Metódus		
		Saját	Kikuchi	Peng
Domb	iOS	Könnyű	Sikertelen	Nehéz
	Android	Könnyű	Sikertelen	Nehéz
Lépcső	iOS	Könnyű	Sikertelen	Sikertelen
	Android	Könnyű	Sikertelen	Sikertelen
Váza	iOS	Könnyű	Sikertelen	Könnyű
	Android	Könnyű	Sikertelen	Könnyű
Autó	iOS	Könnyű	Sikertelen	Könnyű
	Android	Könnyű	Sikertelen	Könnyű

3.2. táblázat. A táblázat Kikuchi és mtsai., Peng és mtsai., valamint a saját módszerünk által beágyazott QR-kódok olvashatóságát mutatja. Kikuchi és mtsai. módszeréhez párhuzamos vetítést, a saját és Peng és mtsai. modelljeinek az elkészítéséhez pedig centrális vetítést használtunk, 10 cm-es centrumtávolsággal.

A párhuzamos és centrális vetítés összehasonlításához megmértük azt a tartományt, amelyben a beágyazott QR-kódok problémamentesen és folyamatosan beolvashatók (lásd 3.3. táblázat). A tesztünk során öt különböző centrumtávolságot használtunk (5, 10, 12, 15 és 20 cm) a centrális vetítés meghatározásához. A 3.3. táblázatban láthatjuk a problémás felületek olvasási tartományai közötti különbségeket, amelyek a beágyazáshoz használt különböző QR-kód méretek miatt jelentek meg (Domb: 3 cm, Lépcső: 4.3 cm, Váza: 5 cm, Autó: 1 cm). A QR-kódok beágyazásához használt méreteket a kiválasztott elhelyezési pont, valamint a felület alakja befolyásolta. A méret és az olvasási tartományok közötti különbség mellett azt is láthatjuk, hogy a beágyazáshoz kiválasztott felület íveltségétől is függnek az olvasási tartományok.

A legkisebb centrumtávolságunkat (5 cm) használva nem tudtuk hatékonyan olvasni a nagy méretű QR-kódokat (domb és lépcső felület). A fennmaradó esetekben azonban sikerült gyorsan dekódolni azokat egy megnyugtatóan széles olvashatósági tartományon belül (lásd 3.3. ábra). Kis centrumtávolságú vetítéssel beágyazva egy nagyobb méretű QR-kódot,

Vetítés	Centrumtávolság	Domb		Lépcső		Váza		Autó	
		Min	Max	Min	Max	Min	Max	Min	Max
Centrális	5 cm	7	7	5	5	10	82	10	14
	10 cm	9	51	8	18	10	90	14	35
	12 cm	12	60	9	24	11	100	17	45
	15 cm	13	68	10	52	12	105	20	55
	20 cm	14	80	11	75	12	110	22	65
Párhuzamos	-	17	90	21	92	12	120	64	92

3.3. táblázat. Meghatároztuk azt a távolságtartományát a QR-kódoknak, ahonnan adott vetítés mellett még stabilan beolvashatók. Az ismertetett mérési eredményeket cm-ben kell érteni.

kisebb olvasási tartományt fogunk elérni. Azonban, az eltérő méretű QR-kódok olvasási tartományai közti különbségek, annál jelentéktelenebbek lesznek, minél nagyobb a használt centrumtávolság.

A hagyományos síkbeli QR-kódok olvasási tatormányait a megoldásunk vizsgálata során kapott eredményekkel hasonlítottuk össze. A centrális vetítéssel készített és a síkbeli QR-kód minimális olvasási távolságai közel azonosak, azonban maximális távolságnak már jóval kisebb értéket mértünk a bemélyesztett QR-kód esetében, mint a papírra nyomtatottnál.

A fenti vizsgálatainkból látható, hogy a centrális vetítéssel beágyazott QR-kód olvasási tartománya, a megválasztott centrumtávolság függvényében változik. Kellően nagynak választott centrumtávolság esetén olyan beolvasási tartományt kaphatunk, amely a párhuzamos vetítéssel elérhetőhöz hasonló. Így a centrális vetítés használatával, nemcsak elkerülhető a párhuzamos vetítéssel beágyazott QR-kódok esetében tapasztalható deformáció, hanem hasonlóan nagy beolvasási távolságok érhetők el vele, mint a párhuzamos vetítéssel.

A 3.14. ábrán bemutatott mintáink által definiált felületek anélkül növelik az árnyék mértékét a QR-kód bevésett fekete moduljainak az alján, hogy további számításokra lenne szükség (lásd 3.21. ábra). Továbbá, centrális és a párhuzamos vetítéssel egyaránt használhatók. A hullámos minta (lásd 3.21b. ábra) nagyobb fúvóka átmérővel (0,4 mm) és rétegvastag-

(a) Minta nélkül (b) Hullámos minta (c) Négyzetes minta

3.21. ábra. A 3.14. ábrán bemutatott mintáink használatával növelhetjük az árnyék mértékét, és a beágyazott QR-kód olvashatóságát. Azonban, ahogyan a (c) képen is látható, a QR-kód modulméretének kellően nagynak kell lennie a minta megfelelő nyomtatásához.

sággal (0,2 mm) történő nyomtatás után is látható marad, így jelentősen képes növelni a beágyazott QR-kód alján lévő árnyék mértékét. A négyzet alakú minta (lásd 3.21c. ábra) kevesebb árnyékot eredményez hasonló fúvóka átmérő és rétegvastagság mellett, mint a hullámos. Azonban, a QR-kód részletgazdagabb nyomtatásával (kisebb rétegvastagság és fúvóka átmérő) már fordított helyzet alakul ki, és a négyzetes minta teljesít jobban.

A hullámos mintával készített QR-kód nyomtatása ≈ 20% százalékkal több időt vesz igénybe, mint egy minta nélkül beágyazott QR-kód elkészítése. A négyzetes minta használata esetén a nyomtatási idő már ≈ 28% körüli értékkel ugrik meg, ugyanazon beállítások használata mellett. Ezt azonban tovább növelheti, ha kisebb rétegmagasság vagy fúvóka átmérő kerül kiválasztásra a részletesebb nyomtatás eléréséért. A minták használatának hatékonysága egyrészt attól függ, hogy milyen részletgazdag modellek készíthetők a kiválasztott 3D nyomtatóval, illetve, hogy mennyi időt szánhatunk a nyomtatására. Az ismertetett nyomtatási időket a Prusa i3 MK2.5 3D nyomató használatával kaptuk.

A függelékben található B.1. és B.2. ábrákon ismertetjük a saját megoldásunk összehasonlítását a [26,58] módszerekkel, amelyhez a 3.1. ábrán látható felületeket használtuk. Peng és mtsai. [58] módszerében a QR-kód beágyazáshoz szükséges felülnézeti irányt, a z tengely negatív irányaként

értelmeztük.

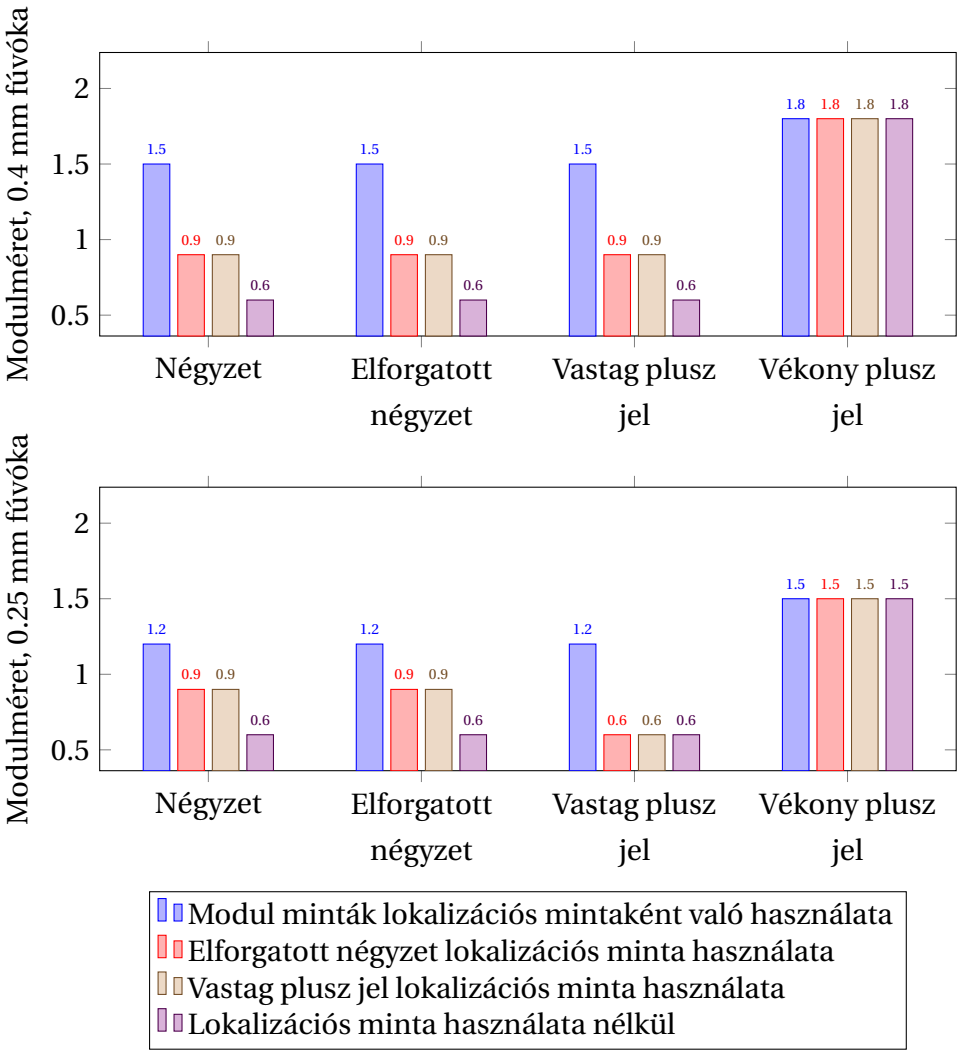
3.4.2. Háromszögelt modell

A QR-kód beágyazásával nyert 3D modellek interaktívan is megtekinthetők a weben[1], köszönhetően a böngészőben futtatható 3D modell nézegetőknek. Emellett egy videó is megnézhető az oldalon, amelyben a kinyomtatott nyúl és huszár modellekbe ágyazott QR-kódot olvassuk be egy Nokia 7.2 telefon segítségével.

Elsőnek azt vizsgáltuk meg, hogy a javasolt mintáink hogyan viselkednek a QR-kód modulméretének változtatására. A teszt során ugyanazokat a modulméreteket használtuk mint Peng és mtsai. [58]. A kapott eredményeinken jól látszik (lásd 3.22. ábra), hogy a QR-kódban található fekete modulok alakjának a beágyazási mélység csökkentése miatt történő lecserélése hatással volt a minimális modul méretre, amely mellett a beágyazott QR-kód még dekódolható.

A 3.22. ábrán látható hogy a mintáinkban használt alakzatok méretének csökkentése magával vonzza a beolvasáshoz szükséges QR-kód modulméret növekedését. Ez az effekt a lokalizációs mintázataink esetében is megfigyelhető, mivel csak ezen mintázatok használatának mellőzésével voltunk képesek kisebb modulméret mellett is dekódolható módon QR-kódokat beágyazni. A vastag plusz jel mintát alkalmazva kaptuk a legkisebb modulméretet (0.6 mm) a QR-kód beágyazásához, míg a vékony plusz jelet használva a legnagyobb méretet kaptuk a teszt során. A másik két modul mintázat (négyzet és elforgatott négyzet) és lokalizációs minták együttes használatával pedig egy szintén kicsinek mondható, 0.9 mm-es modulméretet értünk el, amely 37 mm oldalhosszúságú QR-kódot eredményezett. Azonban a modulméret csökkentésével egyre több műanyag szál jelent meg a nyomtatás során a mintázatokban, amelyek eltávolítása nélkül a QR-kód olvashatatlan volt. Ezen felül az is jól látszik a kapott eredményekből, hogy a modul mintázatainknak az összes fekete

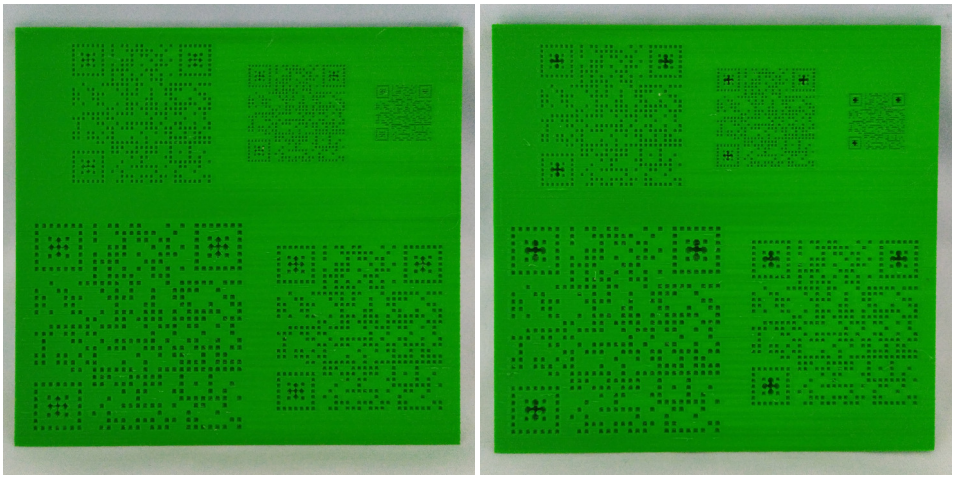
[1] `https://arato.inf.unideb.hu/papp.gyorgy/qr_in_mesh/`



3.22. ábra. Az egyes fúvókaátmérő használata mellett elérhető legkisebb modul méreteket mutatjuk be az ábrán minden javasolt mintázathoz. A [58] munkában található modulméreteket használtuk a teszteléshez.

modulon történő alkalmazásával minden esetben a legnagyobb modulméret mellett tudtunk csak olvasható kódot beágyazni. A 3.23. ábrán bemutatunk néhány, a teszthez használt különböző modulmérettel beágyazott QR-kódot.

A 0.4 mm átmerőjű fúvókát kisebb, 0.25 mm átmérőjűre cserélve rész-



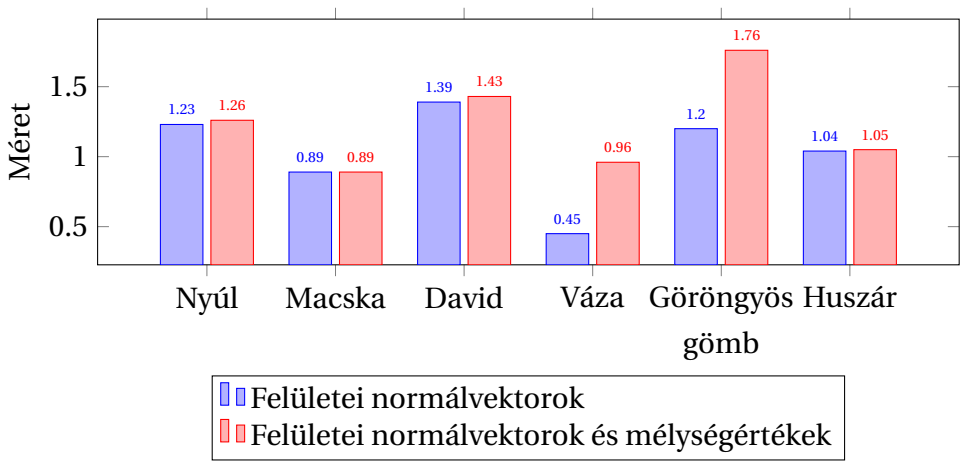
3.23. ábra. A képeken látható QR-kódokat különböző méretben (első sor: 1.2 mm, 0.9 mm, 0.6 mm, második sor: 1.8 mm, 1.5 mm), de ugyanazon modulmintázattal (négyzetes) ágyaztuk be. A jobb oldali QR-kódokhoz a vastag plusz jel lokalizációs mintát, míg a baloldalihoz az elforgatott négyzettel megalkotott lokalizációs mintázatot használtuk fel.

letgazdagabb módon nyomtathatjuk ki a javasolt mintáinkat, valamint egyes esetekben tovább csökkenthetjük a beolvasáshoz szükséges minimális modulméretet (lásd 3.22. ábra). A fenti eredmények alapján a QR-kód lehetséges legkisebb mérete nagyban függ a kiválasztott mintázattól. Emiatt pedig, ahogyan a későbbi összehasonlításban majd látható lesz, a kívánt beágyazási mélységnek is nagy szerepe lesz a QR-kód lehetséges méretének meghatározásában.

Két módszert mutattunk be a QR-kód beágyazásához szükséges vetítési irány meghatározásához, amelyekkel a háromszögelt modellek felületére vetíthetjük a QR-kódot. Emellett, szintén két módszert ismertettünk a QR-kód automatikus méretének megtalálásához, ha ismerjük a QR-kód felületen kiválasztott középpontját, valamint a hozzá tartozó vetítési irányt. Az első módszerünk egy olyan méretet ad eredményül, amellyel az adott vetítési irányból nézve a QR-kód nem lóg túl a modell körvonalán, legyen az bárhogyan forgatva a vetítési irány mentén. A második módszerünkben egy plusz $\vec{v}$ vektor paraméter segítségével előre definiáljuk a

QR-kód orientációját, és az így kapott QR-kód elhelyezéshez keressük azt a méretet, amelynél még deformációmentes marad a beágyazás.

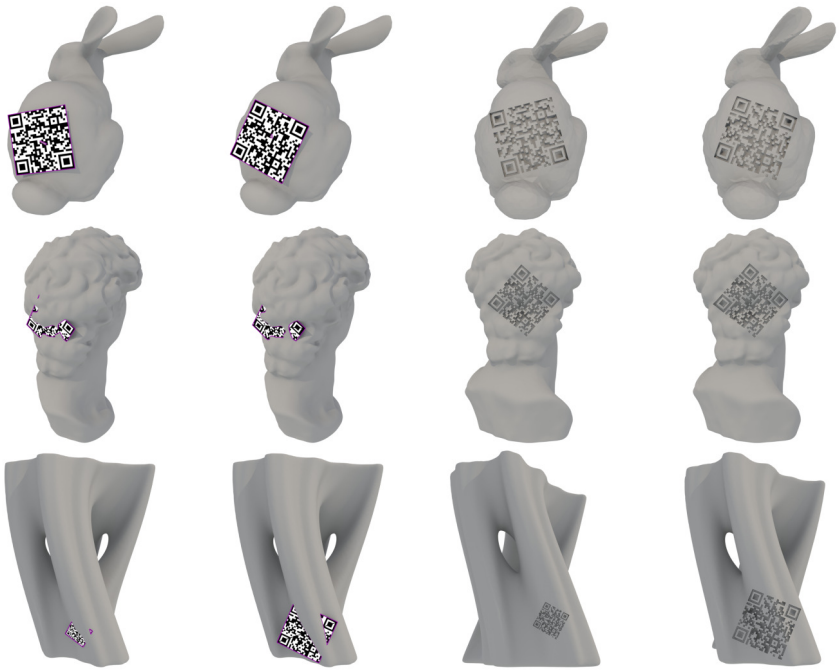
A következő tesztünkben a QR-kódok úgynevezett garantált méretét kerestük meg a két javasolt módszerünkkel. A módszerek bemeneti paramétereiként a vetítési irányt kereső algoritmusaink eredményét használtuk. Ezáltal a vetítési irányt meghatározó megoldásaink helyességét is ellenőrizni tudtuk. A 3.24. ábrán összehasonlítjuk, hogy melyik vetítési irányt kereső módszer eredményével találtunk nagyobb QR-kód méretet, majd a 3.25. ábrán pedig megmutatjuk a QR-kódok beágyazásához eredményül kapott, vetítési irányt jelző négyzetlapokat.



3.24. ábra. A bemutatott vetítési irányt kereső módszereink összehasonlítása, a QR-kód garantált beágyazási méretével.

A 3.25. ábrán látható a vetítési irányt kereső módszereink összehasonlítása. Mindkét megoldásunkkal meghatároztuk a vetítés irányát egy adott QR-kód középponthoz, majd az alapján pedig kiszámoltuk a QR-kód garantált beágyazási méretét. Az összehasonlításhoz további képeket helyeztünk el a függelék B.3. ábráján. Az itt bemutatott modellek és hozzájuk tartozó QR-kód elhelyezések közül többet is felhasználtunk a későbbi tesztjeink során.

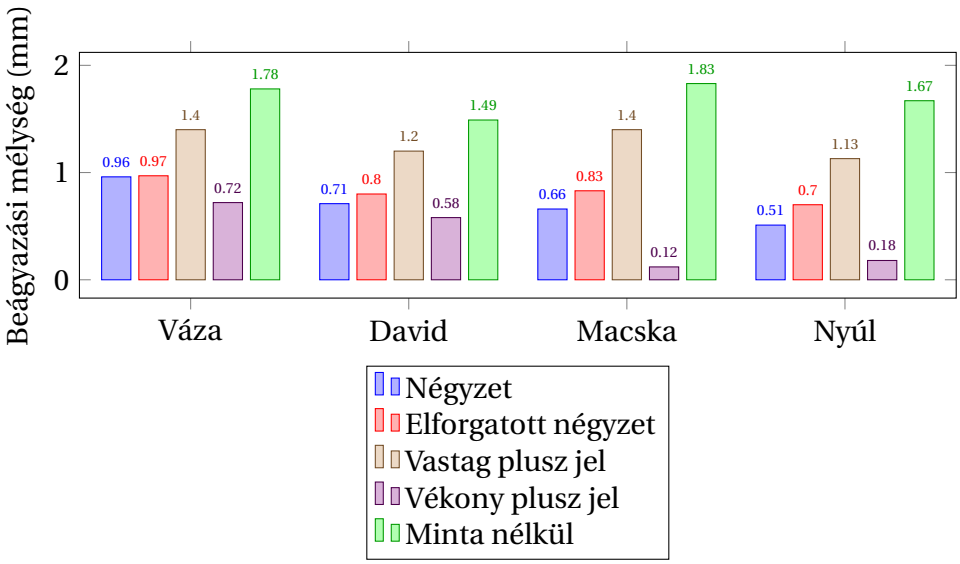
A kapott eredmények alapján, nagyobb méretű QR-kódot érhetünk el az erősen ívelt felületek esetén, ha a mélységértékeket is felhasználjuk a



3.25. ábra. A QR-kód vetítési irányának keresésre csak a felületi normálvektorokat használó módszerünk (1. és 3. oszlop) és a mélységértékeket is felhasználó megoldásunk (2. és 4. oszlop) által eredményezett síkok, valamint az azokból kiszámított legnagyobb garantált beágyazási mérettel készített QR-kódok láthatók az ábrán.

vetítési irány meghatározásához. Azonban a javasolt módszereink közötti különbség minimálisra csökken, ha a QR-kód számára kiválasztott felületrész hullámos, vagy nem erősen ívelt. Ezért a további tesztjeinkben, a második, mélységinformációkat is felhasználó módszerünket alkalmazzuk a vetítési irány megtalálására.

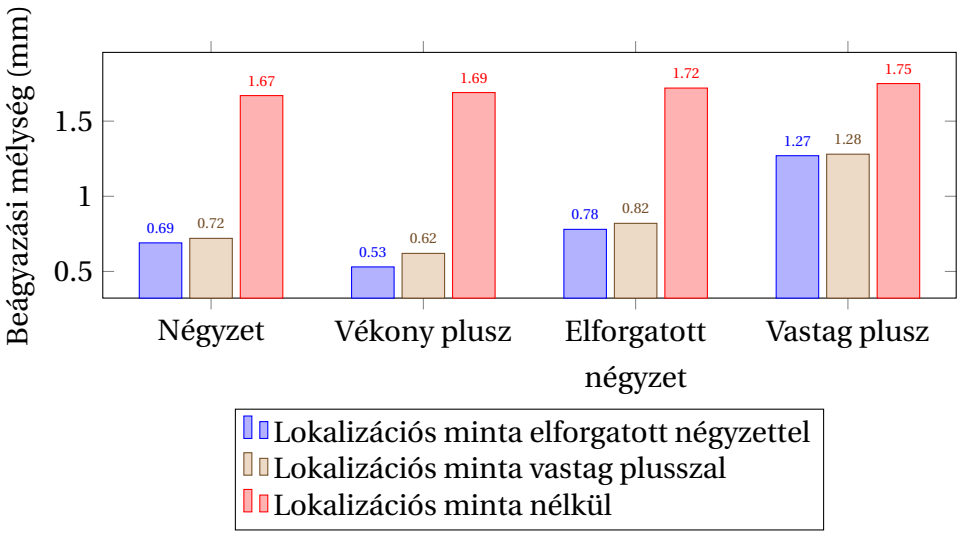
A QR-kód olvasható beágyazásához szükséges mélység csökkentését a fekete modulok formájának módosításával is elérhetjük. A következő tesztünkben ezen formákat leíró mintáink hatékonyságát vizsgáltuk. A különböző mintával, illetve anélkül elkészített QR-kódok beágyazásának a mélységeit hasonlítottuk össze a 3.26. ábra segítségével. Mindemellett azt is teszteltük, hogy milyen beágyazási mélység érhető el a modul és lokalizációs mintázataink kombinációjával. Az így kapott mérési eredmények a 3.27. ábrán láthatók.



3.26. ábra. Azon QR-kódok beágyazási mélységei, amelyek készítésekor a modulok formáját a javasolt mintáinkkal helyettesítettük.

A fekete modulok azon mélységét keressük a javasolt metódusunkkal, amellyel a QR-kód dekódolásához már megfelelő mértékű kontrasztot érhetünk el a fekete és fehér modulok között. A mérési eredményeink alapján csökkenteni tudjuk a fekete modulok mélységét azáltal, hogy lecseréljük az alakjukat a javasolt mintáinkban definiált formákra. A legkisebb beágyazási mélységet azon minták használatával érhetjük el, amelyek elválasztják egymástól a fekete modulokat (vékony plusz és négyzet). Méréseink során a vékony plusz minta teljesített legjobban, míg legrosszabbul a vastag plusz minta szerepelt. A négyzet és elforgatott négyzet mintáink beágyazási mélységei között csak kisebb különbségeket figyelhettünk meg, amelyek a QR-kód bemélyesztéséhez kiválasztott felületrész íveltségének növekedésével még csekélyebbek lettek

Emellett, kombináltuk egymással a QR-kód moduljainkhoz, valamint lokalizációs mintájához definiált formáinkat. Ezeknek a használatával elkészített QR-kódoknak szintén lemértük a beágyazási mélységét, hogy a kombinált formáink hatékonyságát is össze tudjuk hasonlítani (lásd 3.27. ábra). Láthatjuk, hogy sokkal mélyebb beágyazott QR-kódot eredményez,



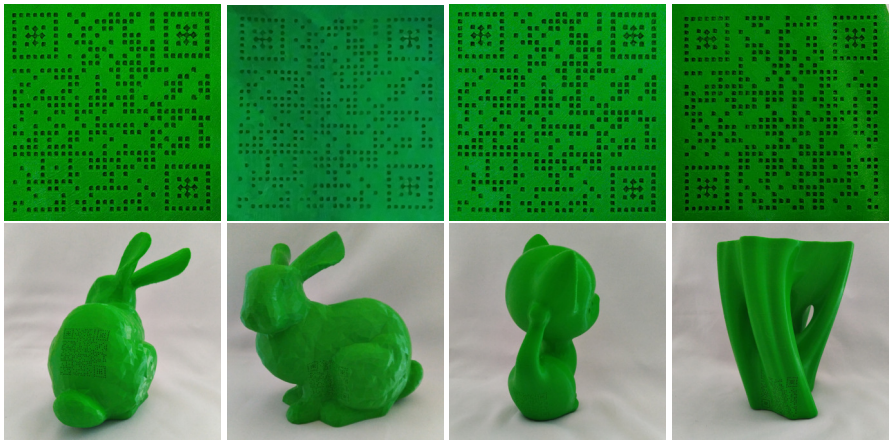
3.27. ábra. A modulokhoz és lokalizációs mintához adott formáink kombinációival készített QR-kódok beágyazási mélységeit hasonlítottuk össze.

ha a fekete modulok alakját lecseréljük, viszont a QR-kód lokalizációs mintáját nem változtatjuk meg. Emellett azt is megmutatja a 3.27. ábra, hogy mennyivel sekélyebben beágyazott QR-kódot lehet elérni a lokalizációs minta lecserélésére adott két formánkkal. Továbbá, jól szemlélteti az ábra, hogy milyen minimális eltérés van a lokalizációs mintákra adott formáinkkal elért mélységek között.

A bemutatott mintáinkkal készített QR-kódok olvashatóságát is megvizsgáltuk, amelyhez a 3D nyomtatott QR-kódokat az Android-os QRbot alkalmazás használatával próbáltuk beolvasni.

A 3.28. ábrán olyan 3D nyomtatott QR-kódok láthatók, amelyekben a négyzetes modul mintát az elforgatott négyzet lokalizációs mintánkkal használtuk együtt. Emellett további képeket helyeztünk el a függelékben (lásd B.4. ábra), amelyeken a Dávid szobor felületébe ágyazott QR-kódok láthatók a javasolt mintáink használatával. A képeken látható modelleket a [57,58] munkákból választottuk ki, hogy összehasonlíthassuk a javasolt vetítési irányt kereső algoritmusokat a korábbi munkákban bemutatott módszerekkel. Emellett azt is megmutatja a tesztünk, hogy a javasolt

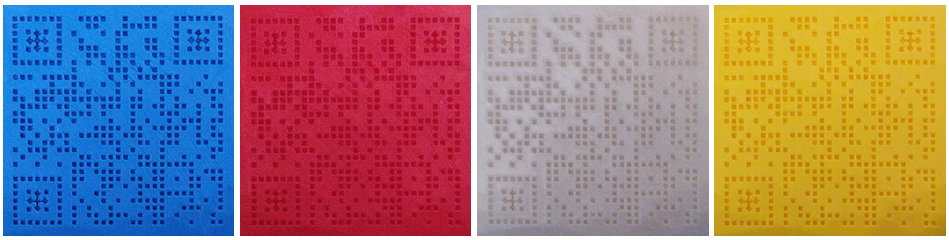
lokalizációs mintáink milyen jól teljesítenek a QR-kód beolvasása során szemben azzal, amikor nem használjuk őket, vagy a teljes QR-kódon a modulokra megadott mintákat használjuk.



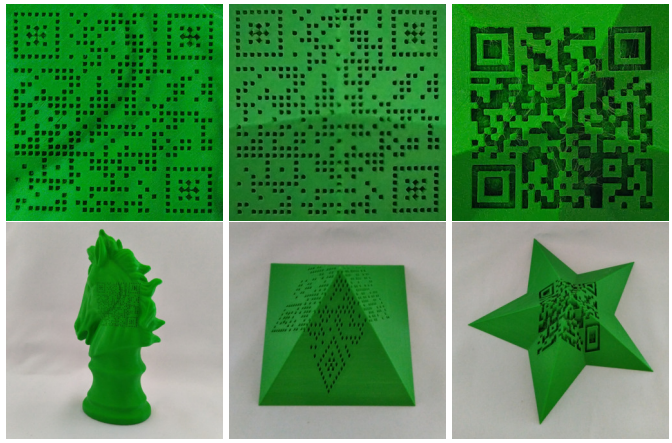
3.28. ábra. A 3D nyomtatóval előállított modellekbe ágyazott QR-kódok elkészítéséhez a négyzet (modul) és elforgatott négyzet (lokalizációs minta) mintáinkat használtuk fel.

Továbbá megvizsgáltuk, hogy a QR-kód beágyazásához javasolt vetítési irány kereső megoldásunk és mintázataink milyen jól teljesítenek ha a kódot különböző színű modelleken szeretnénk elhelyezni. A 3.29. ábrán látható kinyomtatott QR-kódok jól mutatják, hogy a modell színének megváltoztatása nem befolyásolja a beágyazott kód olvashatóságát.

A QR-kód beágyazásához javasolt módszereinket új modellek felhasználásával is teszteltük. Ehhez a huszár sakk figurát, a piramist és a csillag modelleket választottuk. A 3D nyomtatott modellek a 3.30. ábrán tekinthetők meg. Ahogyan az ábrán is látható, a csillag modellnél nem alkal-



3.29. ábra. QR-kódok különböző színű modellbe történő beágyazása.



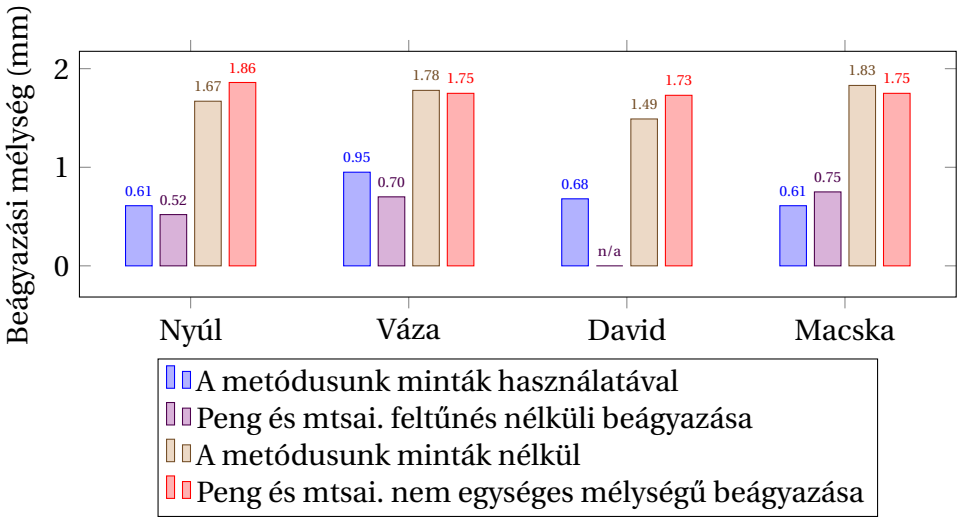
3.30. ábra. Különböző modellekbe ágyazott QR-kódok. Csak az első két modul esetében használtunk mintákat a beágyazáshoz, hogy a harmadik modell segítségével megmutassuk a módszereink robusztusságát.

maztunk mintát, hogy megmutassuk a javasolt módszereink robusztusságát.

A tesztünkben azt is megvizsgáltuk, hogy a lokalizációs minta alakjának a megváltoztatása miként befolyásolja az elkészült QR-kód olvashatóságát. Ha minden fekete modult a 3.17. ábrán látható minták valamelyikének a használatával módosítunk, akkor csak nehezen, vagy egyáltalán nem olvasható QR-kódokat kapunk eredményül. A modulok méretének növelésével könnyen kiküszöbölhető ez az olvasási probléma. Viszont ezt nem tudjuk megtenni, ha adott mérettel kell beágyazni a QR-kódot. Erre jelent megoldást, a lokalizációs minták lecseréléséhez javasolt formáink használata, amelyekkel a modulok méretének növelése nélkül kaphatunk dekódolható QR-kódokat. Azonban ezek használata megnöveli a QR-kód beágyazási mélységét.

A négyzet, valamint az elforgatott négyzet mintáink jól teljesítettek az olvashatósági teszt során. A vastag plusz mintánk használatával csak akkor kaptunk olvasható QR-kódot eredményül, ha a beágyazáshoz kijelölt felület nem volt erősen ívelt vagy hullámos. A negyedik, vékony plusz mintának az alakja túl kicsi lett az általunk használt 1,2 mm-es modulméretben, így olvashatatlan QR-kódokat kaptunk az alkalmazásukkal. A

teszt során azt is megfigyelhettük, hogy a mintával beágyazott QR-kód olvashatósága erősen függ 3D nyomtatás minőségtől és részletességétől. Emiatt, a bemutatott mintáink alkalmazásához nagyobb részletességet és jobb minőséget eredményező nyomtatási beállításra van szükség. A mintákkal készített QR-kódok beágyazási mélységei a 3.26. és 3.27. ábrákon láthatók. Ezek alapján a legjobb eredményt, a fekete modulokhoz használt négyzet mintának az elforgatott négyzeten alapuló lokalizációs mintával vett kombinációjával érhetjük el.



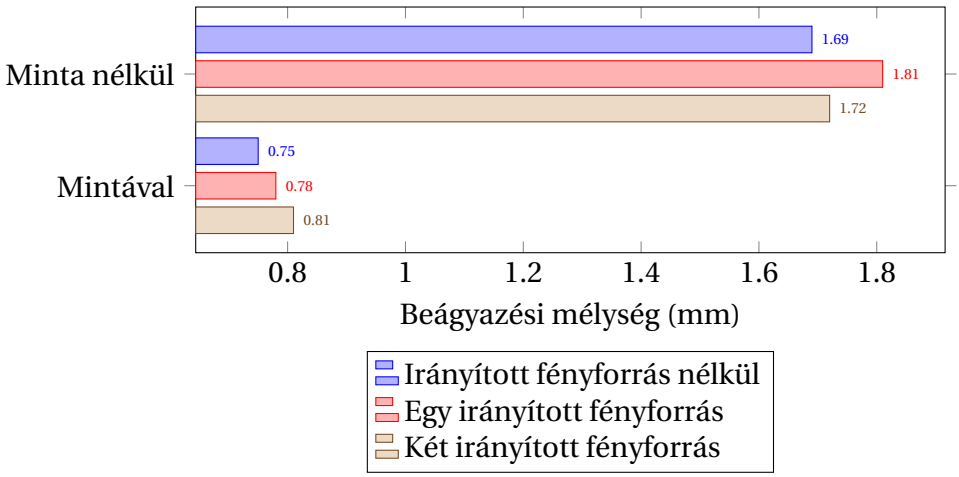
3.31. ábra. Saját módszerünk összehasonlítása az [57, 58] módszerekkel.

A 3.31. ábrán hasonlítjuk össze saját, valamint az [57, 58] módszerekkel kapott QR-kódok beágyazási mélységeit. A saját megoldásunkkal készített QR-kódokhoz a legjobban teljesítő mintánkat használtuk.

A módszerünkkel készített QR-kódok beágyazási mélysége kisebb, mint ami az [58] módszerrel elérhető. A legkisebb mélységet az [57] metódus használatával kaphatjuk a vizsgált modellek többségénél. Emellett az HBAO metóduson alapuló mélységoptimalizálási módszerünkkel képesek vagyunk hasonló beágyazási mélységeket megadni a minták nélkül elkészített QR-kódhoz mint [58]. Viszont a módszerünknek csak az [57, 58] megoldásokban ismertetett futási idő töredékére van szüksége a beágyazási mélység

meghatározásához.

Ezek mellett azt is teszteltük, hogy miként működik a mélységoptimalizálási módszerünk, ha irányított fényforrásokat is használunk benne. Ehhez a QR-kódot egy vagy két irányított fényforrás használatával ágyaztuk be, majd az így kapott eredményeket pedig összehasonlítottuk az eddigi eredményeinkkel (lásd 3.32. ábra).

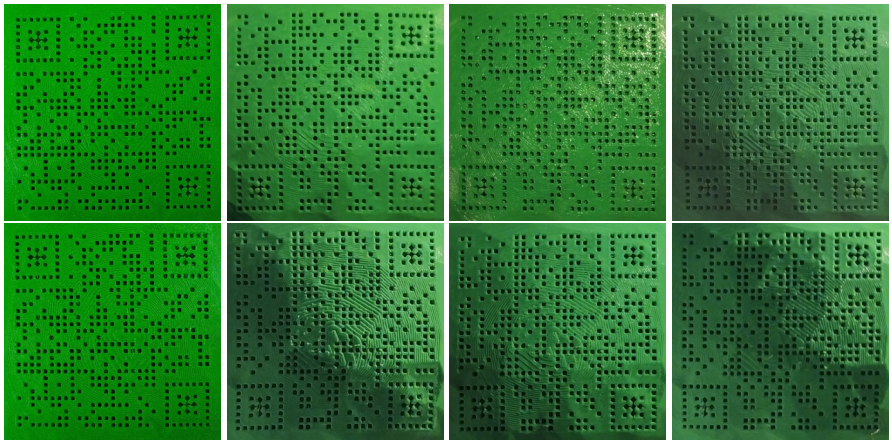


3.32. ábra. A QR-kód beágyazásának irányított fényforrások használatával történő tesztelése.

A 3.32. ábrán látható, hogy nagyobb mélységet eredményez, ha a QR-kódok irányított fényforrások hozzáadásával kerülnek beágyazásra. További fényforrások hozzáadásának a hatására, a QR-kód olvasható beágyazásához szükséges mélység attól függően nő vagy csökken, hogy használtunk-e mintánkat a QR-kód elkészítéséhez vagy sem. Emellett azt is teszteltük, hogy mennyire robusztus a mélység optimalizálási módszerünk a beágyaznál használt irányított fényforrások módosítására, vagy figyelmen kívül hagyására. A 3.33. ábrán látható példákon szemléltetjük a QR-kód elkészítéséhez alkalmazott fényirányoktól való eltérés hatását a kapott kód olvashatóságára.

Az irányított fények és minták együttes használata olvashatatlan QR-kódokat eredményezett, azonban a minták elhagyásával már dekódolható eredményt kaptunk. Emiatt, csak a minta nélkül beágyazott QR-kódok

(a) Minta nélkül



(b) Mintával

3.33. ábra. Példákat mutatunk a QR-kód irányított fényforrásokkal történő beágyazására és összehasonlítására. Az első oszlopban nem, míg a többiben egyetlen irányított fényforrást használtunk. A harmadik és negyedik oszlopban, a másodikban használt fényforrás iránytól felfelé és lefelé eltérő irányt alkalmaztunk, hogy egy jobban, és egy kevésbé jól megvilágított QR-kódot kapjunk. Az alsó sorokban nagynak választottuk meg a beágyazáshoz használt irányított fényforrás és a QR-kód síkjának normálvektorja által bezárt szöget, a felsőkben pedig kicsinek.

esetében ismertetjük a további vizsgálataink eredményét. A elkészült QR-kódunk annál robusztusabbnak bizonyult a beágyazásnál használt fényviszonyoktól való eltérésekre, minél kisebbnek választottunk meg az irányított fényforrás és a QR-kód síkjának normálvektorja által bezárt szö-

get. Azonban, ennek a csökkentése maga után vonja a QR-kód beágyazási mélységének a növekedését is. Ez azt eredményezi, hogy a beolvasás szempontjából robusztus QR-kódért cserébe fel kell áldoznunk a minél kisebb beágyazási mélységre való törekvésünket. A tesztünkben kisebb szöget bezáró (33°) irányított fény használatával készített QR-kód, maximum ≈ 30°-al nagyobb szögben beeső fény mellett volt még továbbra is olvasható. A másik esetben, amikor nagyobb szöget bezáró (72°) irányított fényt használatunk, a QR-kód csak ≈ 15°-al nagyobb eltérést tolerált a bejövő fény irányának növelésével szemben. A beeső fényirány csökkentése esetén, a QR-kód végig dekódolható marad a bezárt szögtől függetlenül.

4. Összefoglalás

Megadtunk egy körön alapuló elrendezést használó vizualizációs eljárást [47, 53], amely különböző eszközöket és interaktív funkciókat nyújt a Circos [30] szoftvercsomagban található hibák javítására. Az új eszközökkel a kapcsolatok értékének, irányának, és az általuk alkotott mintázatnak az olvashatóságát javítottuk. Továbbá növeltük a diagram átláthatóságát az abban található kapcsolatok közötti metszések számának csökkentésével. Ehhez egy lokális kereső és optimalizáló algoritmusokon alapuló rendezést adtunk meg. Emellett egy hierarchikus élkötegelést használó módszert ismertettünk a diagram zsúfoltságának csökkentésére.

Merőlegesen metsző kúpszeleteket [54–56] adtunk meg a Bézier görbék kapcsolatok vizualizációjában történő lecserélésre. A javasolt kúpszeletek nem csak paraméteres, hanem implicit egyenlettel is rendelkeznek, amely segítséget nyújt a kapcsolatok interaktív kijelölésében. Szerkesztési eljárásokat adtunk a javasolt kúpszeleteinkhez, valamint megvizsgáltuk, hogy miként viselkednek a kúpszeletekkel alkotott szalagok a kapcsolatok vizualizációjában.

Az információk tárgyak felületén való megjelenítésére adott módszereinkben [48–52], egy QR-kódot ágyazunk be a felhasználó által paraméteres felületen vagy háromszögelt modellen kiválasztott felületrészbe. A bemutatott módszereink automatikusan meghatároznak egy olyan vetítési irányt a QR-kód középpontjának és méretének az ismeretében, amely segítségével a nagy átlaggörbülettel rendelkező felületrészek kiválasztása esetén is dekódolható kódot ágyazunk be. A QR-kódot, háromszögelt modellek felületébe ágyazó módszerünk nem igényli a kiválasztott felületrész újraháromszögelését, valamint képes a QR-kódnak azt a legnagyobb méretét meghatározni, amellyel az garantáltan beágyazható. Továbbá olyan mintákat definiáltunk a QR-kód fekete moduljainak a módosítására, amelyek használatával sötétebb bemélyesztett modulok, va-

lamint kisebb beágyazási mélység érhető el. Mindemellett olyan iteratív módszert adtunk meg a QR-kód beágyazásához szükséges mélység megtalálására, amely a „Horizon-Based Ambient Occlusion” algoritmust használja, és az eddigi módszerek futási idejének töredéke alatt képes megfelelő eredményt szolgáltatni.

5 Summary

We proposed a circular layout based visualization technique [47, 53], that defines multiple tools and interactive functions to the problems of Circos [30]. These new tools are defined to increase the clarity of the connections' values, direction, and pattern in the diagram. Besides, we decreased the clutter in the diagram by reducing the number of the connections' intersections with a sorting method that uses local search and optimization methods. Also, we presented a hierarchical edge bundling technique for reducing the clutter and increasing the diagram clarity.

We proposed perpendicularly intersecting conic sections [54–56] to replace the Bézier curve in the relationship visualization. Using our proposed conics, one can take advantage of having both implicit and parametric equations. For example to interactively selecting the connections in the visualization. We presented construction methods for the conic sections and discussed how the conics behave when used for creating stripes in the visualization.

In our solutions for showing information on items' surface [48–52], we proposed to embed QR codes onto the user-selected surface patches of parametric surfaces and triangulated models

Our presented methods automatically determine the projection direction for the QR code using its user-selected center and size. We can embed a decodable QR code even on highly curved surfaces using this direction, which is found to project the QR code onto the surface. Our method for embedding a QR code onto triangulated meshes does not require a QR code size because it can find the largest one, which the QR code can be embedded onto the selected surface patch using any rotation. Also, this method does not need to perform any time-costly subdivision of the selected surface before the embedding. Besides, we proposed different patterns for modifying the black modules to increase the

shadow in the embedded QR code, and decrease its carving depth. Moreover, we presented an iterative method using the Horizon-Based Ambient Occlusion for finding the embedded QR codes' carving depth. This solution's running time is only a fraction of the previously presented methods' running time.

Köszönetnyilvánítás

Szeretném megköszönni témavezetőmnek, Papp Ildikónak az értekezés megírásához adott sok hasznos és elengedhetetlen tanácsot, ötletet és iránymutatást, valamint a doktori képzés elvégzéséhez nyújtott folyamatos támogatást.

Külön köszönöm családomnak azt a sok türelmet és támogatást, amit az egyetemi éveim alatt, valamint az értekezésem elkészítéséhez kaptam tőlük.

Köszönöm Hoffmann Miklósnak és Kunkli Rolandnak a közös munka során adott tanácsokat és iránymutatásokat.

Köszönöm még mindazoknak, akik segítséget nyújtottak a cikkeim elkészítéséhez, segítettek tesztelni az azokban ismertetett megoldásokat.

Mindezek mellett köszönöm mindazt a támogatást[1], amit a kutatásom során kaptam.

[1] A kutatást az „Integrált kutatói utánpótlás-képzési program az informatika és számítástudomány diszciplináris területein” (EFOP-3.6.3-VEKOP-16-2017-00002) című projekt támogatta. A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.

Irodalomjegyzék

[1] Jiyuan An – John Lai – Atul Sajjanhar – Jyotsna Batra – Chenwei Wang – Colleen C. Nelson: J-Circos: an interactive Circos plotter. *Bioinformatics*, 31. évf. (2015) 9. sz., 1463–1465. p.

[2] Michael Baur – Ulrik Brandes: *Crossing Reduction in Circular Layouts*. Berlin, Heidelberg, 2005, Springer Berlin Heidelberg, 332–343. p. ISBN 978-3-540-30559-0.

[3] Louis Bavoil – Miguel Sainz – Rouslan Dimitrov: Image-Space Horizon-Based Ambient Occlusion. In *ACM SIGGRAPH 2008 Talks*, SIGGRAPH '08 konferenciasorozat. New York, NY, USA, 2008, Association for Computing Machinery. ISBN 9781605583433. 1 p.

[4] Tilman Bayer: „A Thousand Fibers Connect Us" – WikiViz 2011 winner visualizes Wikipedia's global reach, 2011. October. `https://blog.wikimedia.org/2011/10/06/a-thousand-fibers-connect-us-wikiviz-winner-visualize-wikipedias-global-reach/`. Elérés időpontja: 2018. március 6.

[5] William Henry Besant: *Conic Sections Treated Geometrically*. 1895, G. Bell.

[6] Laia Blasco-Soplón – Josep Grau-Valldosera – Julià Minguillón: Visualization of Enrollment Data using Chord Diagrams. In *Proceedings of the 10th International Conference on Computer Graphics Theory and Applications - Volume 1: GRAPP, (VISIGRAPP 2015)* (konferenciaanyag). 2015, INSTICC, SciTePress, 511–516. p. ISBN 978-989-758-087-1.

[7] Jason Brownlee: *Clever Algorithms: Nature-Inspired Programming Recipes*. 1st. kiad. 2011, Lulu.com. ISBN 1446785068, 9781446785065.

[8] Diana Burkart-Waco – Caroline Josefsson – Brian Dilkes – Nora Kozloff – Otto Torjek – Rhonda Meyer – Thomas Altmann – Luca Comai:

Hybrid Incompatibility in Arabidopsis Is Determined by a Multiple-Locus Genetic Network. *Plant Physiology*, 158. évf. (2012) 2. sz., 801–812. p.

[9] Franco Cauda – Diana M-E. Torta – Katiuscia Sacco – Elisabetta Geda – Federico D'Agata – Tommaso Costa – Sergio Duca – Giuliano Geminiani – Martina Amanzio: Shared "Core" Areas between the Pain and Other Task-Related Networks. *PLOS ONE*, 7. évf. (2012. 08) 8. sz., 1–10. p.

[10] Fei Chen – Yuxi Luo – Nektarios Georgios Tsoutsos – Michail Maniatakos – Khaled Shahin – Nikhil Gupta: Embedding Tracking Codes in Additive Manufactured Parts for Product Authentication. *Advanced Engineering Materials*, 21. évf. (2019) 4. sz., 1800495. p.

[11] Hung-Kuo Chu – Chia-Sheng Chang – Ruen-Rone Lee – Niloy J. Mitra: Halftone QR Codes. *ACM Trans.Graph.*, 32. évf. (2013) 6. sz., 217:1–217:8. p. 09.

[12] Mei C. Chuah – Stephen G. Eick: Information rich glyphs for software management data. *IEEE Computer Graphics and Applications*, 18. évf. (1998) 4. sz., 24–29. p.

[13] Qian Dang – Xiaowen Lin – Megan Konar: Agricultural virtual water flows within the United States. *Water Resources Research*, 51. évf. (2015) 2. sz., 973–986. p.

[14] Yoshi Fujiwara: Visualizing Open Data of Input-output Tables in Kobe City. In *SIGGRAPH Asia 2015 Visualization in High Performance Computing*, SA '15 konferenciasorozat. New York, NY, USA, 2015, ACM, 18:1–18:1. p. ISBN 978-1-4503-3929-2. 1 p.

[15] Akshara M. Gaikwad – Kavita R. Singh: Embedding QR code in color images using halftoning technique. In *2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)* (konferenciaanyag). 2015, IEEE, 1–6. p.

[16] Emden R. Gansner – Yehuda Koren: Improved Circular Layouts. In Michael Kaufmann – Dorothea Wagner (szerk.): *Graph Drawing* (konferenciaanyag). Berlin, Heidelberg, 2007, Springer Berlin Heidelberg, 386–398. p. ISBN 978-3-540-70904-6.

[17] Gonzalo J. Garateguy – Gonzalo R. Arce – Daniel L. Lau – Ofelia P. Villarreal: QR Images: Optimized Image Embedding in QR Codes. *IEEE Transactions on Image Processing*, 23. évf. (2014) 7. sz., 2842–2853. p.

[18] Georg Glaeser – Hellmuth Stachel – Boris Odehnal: *The Universe of Conics: From the ancient Greeks to 21st century developments*. 2016, Springer.

[19] Zuguang Gu – Lei Gu – Roland Eils – Matthias Schlesner – Benedikt Brors: circlize implements and enhances circular visualization in R. *Bioinformatics*, 30. évf. (2014) 19. sz., 2811–2812. p.

[20] György Hajós: *Bevezetés a geometriába [Introduction to Geometry]*. 1984, Tankönyvkiadó.

[21] Danny Holten: Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *IEEE Transactions on Visualization and Computer Graphics*, 12. évf. (2006. Sept) 5. sz., 741–748. p.

[22] Ying Hu – Chunhua Yan – Chih-Hao Hsu – Qing-Rong Chen – Kelvin Niu – George A. Komatsoulis – Daoud Meerzaman: OmicCircos: A Simple-to-Use R Package for the Circular Visualization of Multidimensional Omics Data. *Cancer Informatics*, 13. évf. (2014), 13–20. p.

[23] Andrei Irimia – John D. Van Horn: The structural, connectomic and network covariance of the human brain. *NeuroImage*, 66. évf. (2013), 489–499. p.

[24] Andrei Irimia – John Van Horn: Systematic network lesioning reveals the core white matter scaffold of the human brain. *Frontiers in Human Neuroscience*, 8. évf. (2014), 51. p.

[25] Vijeth K. Katapadi – Mridula Nambiar – Sathees C. Raghavan: Potential G-quadruplex formation at breakpoint regions of chromosomal translocations in cancer may explain their fragility. *Genomics*, 100. évf. (2012) 2. sz., 72–80. p.

[26] Ryosuke Kikuchi – Sora Yoshikawa – Pradeep Kumar Jayaraman – Jianmin Zheng – Takashi Maekawa: Embedding QR codes onto B-spline surfaces for 3D printing. *Computer-Aided Design*, 102. évf. (2018), 215–223. p. ID: 271502.

[27] Ladislav Kocis – William J. Whiten: Computational Investigations of Low-Discrepancy Sequences. *ACM Trans. Math. Softw.*, 23. évf. (1997. június) 2. sz., 266–294. p. 29 p.

[28] Martin Krzywinski: Articles // CIRCOS Circular Genome Data Visualization. `http://circos.ca/presentations/articles/vis_tables1/#tables`. Elérés időpontja: 2018. március 6.

[29] Martin Krzywinski: Circos Table Viewer - Circular Visualization of Tabular Data // with Circos - Circular Genome Data Visualization. `http://mkweb.bcgsc.ca/tableviewer/`. Elérés időpontja: 2017. június 23., 2008.

[30] Martin Krzywinski – Jacqueline Schein – Ilatenanç Birol – Joseph Connors – Randy Gascoyne – Doug Horsman – Steven J. Jones – Marco A. Marra: Circos: An information aesthetic for comparative genomics. *Genome Research*, 19. évf. (2009) 9. sz., 1639–1645. p.

[31] Kuen-Tsair Lay – Lee-Jyi Wang – Pei-Lun Han: Decoding of QR codes printed on spheres. In *2016 5th International Symposium on Next-Generation Electronics (ISNE)* (konferenciaanyag). 2016, IEEE, 1–3. p.

[32] Kuen-Tsair Lay – Lee-Jyi Wang – Pei-Lun Han – Yi-Sheen Lin: Rectification of images of QR codes posted on cylinders by conic segmentation. In *2015 IEEE International Conference on Signal and Image*

Processing Applications (ICSIPA) (konferenciaanyag). 2015, IEEE, 389–393. p.

[33] Kuen-Tsair Lay – Ming-Hao Zhou: Perspective projection for decoding of QR codes posted on cylinders. In *2017 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)* (konferenciaanyag). 2017, IEEE, 39–42. p.

[34] A. Lhuillier – C. Hurter – A. Telea: State of the Art in Edge and Trail Bundling Techniques. *Computer Graphics Forum*, 36. évf. (2017) 3. sz., 619–645. p. 27 p.

[35] Kejing Li – Fanwu Meng – Zhipeng Huang – Qi Wang: A Correction Algorithm of QR Code on Cylindrical Surface. *Journal of Physics: Conference Series*, 1237. évf. (2019. jun), 022006. p.

[36] Xiaochao Li – Zhifeng Shi – Donghui Guo – Shan He: Reconstruct argorithm of 2D barcode for reading the QR code on cylindrical surface. In *2013 International Conference on Anti-Counterfeiting, Security and Identification (ASID)* (konferenciaanyag). 2013, IEEE, 1–5. p. ISBN 2163-5048.

[37] Zheng Li: InfoViz: information visualization library. `http://infoviz.org/`. Elérés időpontja: 2017. május 17., 2012.

[38] Lijian Lin – Song Wu – Sijiang Liu – Bo Jiang: Interactive QR code beautification with full background image embedding. In Xudong Jiang – Masayuki Arai – Guojian Chen (szerk.): *Second International Workshop on Pattern Recognition* (konferenciaanyag), 10443. köt. 2017, International Society for Optics and Photonics, SPIE, 211 – 215. p.

[39] Xiaowen Lin – Qian Dang – Megan Konar: A network analysis of food flows within the United States of America. *Environmental Science & Technology*, 48. évf. (2014) 10. sz., 5439–5447. p.

[40] Yi-Shan Lin – Sheng-Jie Luo – Bing-Yu Chen: Artistic QR Code Embellishment. *Computer Graphics Forum*, 32. évf. (2013) 7. sz., 137–146. p.

[41] M. Meyer – T. Munzner – H. Pfister: MizBee: A Multiscale Synteny Browser. *IEEE Transactions on Visualization and Computer Graphics*, 15. évf. (2009) 6. sz., 897–904. p.

[42] Martin Mittring: Finding next Gen: CryEngine 2. In *ACM SIGGRAPH 2007 Courses*, SIGGRAPH '07 konferenciasorozat. New York, NY, USA, 2007, Association for Computing Machinery, 97–121. p. ISBN 9781450318235. 25 p.

[43] O. Nalbach – E. Arabadzhiyska – D. Mehta – H. P Seidel – T. Ritschel: Deep Shading: Convolutional Neural Networks for Screen Space Shading. *Computer Graphics Forum*, 36. évf. (2017) 4. sz., 65–78. p.

[44] Michael Nicholas – Daniel Archambault – Robert S Laramee: Interactive Visualisation of Automotive Warranty Data Using Novel Extensions of Chord Diagrams. In N. Elmqvist – M. Hlawitschka – J. Kennedy (szerk.): *EuroVis - Short Papers* (konferenciaanyag). 2014, The Eurographics Association. ISBN 978-3-905674-69-9.

[45] Palmiga: Turbosquid - free car 3d model. `https://www.turbosquid.com/3d-models/free-car-3d-model/166348`. Elérés időpontja: 2020. június 25.

[46] Ke-Xi Pan – Han-Xiong Zhu – Zheng Chang – Kuan-Hong Wu – Yu-Li Shan – Zhi-Xing Liu: Estimation of Coal-Related CO_2 Emissions: The Case of China. *Energy & Environment*, 24. évf. (2013) 7-8. sz., 1309–1321. p.

[47] György Papp – Roland Kunkli: TabularVis: An Interactive Relationship Visualization Tool Supported by Optimization and Search Algorithms. In *International Joint Conference on Computer Vision,*

Imaging and Computer Graphics (konferenciaanyag). 2018, Springer, 167–192. p.

[48] György Papp – Miklós Hoffmann – Ildikó Papp: Improved projection for embedding QR code onto surfaces to be 3D printed. In *Graphics and Application The 12th Asian Forum on Graphic Science (AFGS 2019)* (konferenciaanyag). 2019, 116–117. p.

[49] György Papp – Miklós Hoffmann – Ildikó Papp: Embedding QR code onto triangulated meshes. In *The 12th Conference of PhD Students in Computer Science - Volume of short papers* (konferenciaanyag). 2020, 79–82. p.

[50] György Papp – Miklós Hoffmann – Ildikó Papp: Improved QR code embedding for meshes with 3D printing. In *ICAI 2020* (konferenciaanyag). 2020.

[51] György Papp – Miklós Hoffmann – Ildikó Papp: Embedding QR Code onto Triangulated Meshes using Horizon Based Ambient Occlusion. *Computer Graphics Forum*, 2021.

[52] György Papp – Miklós Hoffmann – Ildikó Papp: Improved Embedding of QR Codes onto Surfaces to be 3D Printed. *Computer-Aided Design*, 131. évf. (2021), 102961. p.

[53] György Papp – Roland Kunkli: TabularVis – a Circos-inspired interactive web client based tool for improving the clarity of tabular data visualization. In *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3: IVAPP,* (konferenciaanyag). 2018, INSTICC, SciTePress, 120–131. p. ISBN 978-989-758-289-9.

[54] György Papp – Ildikó Papp – Roland Kunkli: Construction and properties of conics in relationship visualization. 2021. Kézirat.

[55] György Papp – Ildikó Papp – Roland Imre Kunkli: Construction and Properties of Orthogonally Intersecting Conics in Relationship Visualization. In *Conference on Geometry: Theory and Applications; University of Innsbruck, June 3–7, 2019* (konferenciaanyag). 2019, 27–28. p.

[56] György Papp – Ildikó Papp – Roland Imre Kunkli: Possibilities of Using Orthogonally Intersecting Conics in Circular Layout Based Relationship Visualization. In *6th Winter School of PhD Students in Informatics and Mathematics* (konferenciaanyag). 2019, 28–29. p.

[57] Hao Peng – Peiqing Liu – Lin Lu – Andrei Sharf – Lin Liu – Dani Lischinski – Baoquan Chen: Fabricable Unobtrusive 3D-QR-Codes with Directional Light. *Computer Graphics Forum*, 39. évf. (2020) 5. sz., 15–27. p.

[58] Hao Peng – Lin Lu – Lin Liu – Andrei Sharf – Baoquan Chen: Fabricating QR codes on 3D objects using self-shadows. *Computer-Aided Design*, 114. évf. (2019), 91–100. p.

[59] Tobias Ritschel – Thorsten Grosch – Hans-Peter Seidel: Approximating Dynamic Global Illumination in Image Space. In *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games*, I3D '09 konferenciasorozat. New York, NY, USA, 2009, Association for Computing Machinery, 75–82. p. ISBN 9781605584294. 8 p.

[60] S.J. Russell – P. Norvig: *Artificial Intelligence: A Modern Approach*. Prentice Hall Series in Artifi sorozat. 2010, Prentice Hall. ISBN 9780136042594.

[61] Jeremy Schmutz – Phillip E McClean – Sujan Mamidi – G Albert Wu – Steven B Cannon – Jane Grimwood – Jerry Jenkins – Shengqiang Shu – Qijian Song – Carolina Chavarro és mások: A reference genome for common bean and genome-wide analysis of dual domestications. *Nature Genetics*, 46. évf. (2014) 7. sz., 707–713. p.

[62] Ville Timonen: Line-Sweep Ambient Obscurance. *Computer Graphics Forum*, 32. évf. (2013) 4. sz., 97–105. p.

[63] Mbaye Tine – Heiner Kuhl – Pierre-Alexandre Gagnaire – Bruno Louro – Erick Desmarais – Rute ST Martins – Jochen Hecht – Florian Knaust – Khalid Belkhir – Sven Klages és mások: European sea bass genome and its variation provide insights into adaptation to euryhalinity and speciation. *Nature Communications*, 5. évf. (2014), 5770. p.

[64] Sibing Wang – Tao Yang – Jing Li – Bowei Yao – Yanning Zhang: Does a QR code must be black and white? In *2015 International Conference on Orange Technologies (ICOT)* (konferenciaanyag). 2015, IEEE, 161–164. p.

[65] Chao Wei – Zhe Sun – Yihe Huang – Lin Li: Embedding anti-counterfeiting features in metallic components via multiple material additive manufacturing. *Additive Manufacturing*, 24. évf. (2018), 1–12. p. ID: 306190.

[66] Mingliang Xu – Hao Su – Yafei Li – Xi Li – Jing Liao – Jianwei Niu – Pei Lv – Bing Zhou: Stylized Aesthetic QR Code. *IEEE Transactions on Multimedia*, 21. évf. (2019) 8. sz., 1960–1970. p.

[67] Jingru Yang – Hao Peng – Lin Liu – Lin Lu: 3D printed perforated QR codes. *Computers & Graphics*, 81. évf. (2019), 117–124. p. ID: 271576.

[68] Tengfei Yin – Dianne Cook – Michael Lawrence: ggbio: an R package for extending the grammar of graphics for genomic data. *Genome Biology*, 13. évf. (2012) 8. sz., R77. p.

[69] Dongjiu Zhang – Chuhua Xian – Guoliang Luo – Yunhui Xiong – Chu Han: DeepAO: Efficient Screen Space Ambient Occlusion Generation via Deep Network. *IEEE Access*, 8. évf. (2020), 64434–64441. p.

[70] Zhang, Hongen and Meltzer, Paul and Davis, Sean: RCircos: an R package for Circos 2D track plots. *BMC Bioinformatics*, 14. évf. (2013) 1. sz., 244. p.

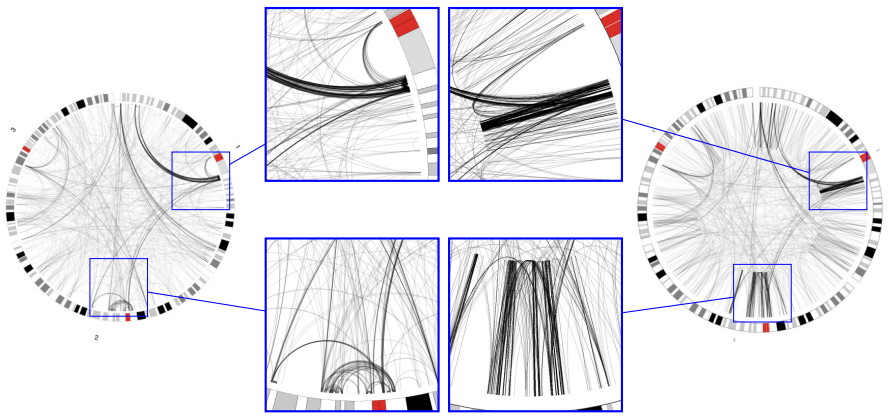
A. TabulrVis

A.1. Keresőalgoritmusok paraméterei

Algoritmusok	Paraméterek						
	Véletlenszerű állapotváltások száma	Iterációk száma	Generált állapotok száma	Elit állapotok száma	Elfogadási arány	Hőmérsékletenkénti iterációk száma	Kereső csoportok száma
Hegymászó algoritmus	10	10	-	-	-	-	-
Minimális konfliktusok	-	10	-	-	-	-	-
Kereszt-entrópia	-	20	20	5	-	-	-
Kereszt-entrópia és minimális konfliktusok	-	20	20	5	-	-	-
Szimulált hűtés	-	100	-	-	0.8	10	-
Méhek algoritmusa	-	10	10	2	-	-	2

A.1. táblázat. Az előre definiált paraméterei a rendezéshez használt algoritmusoknak.

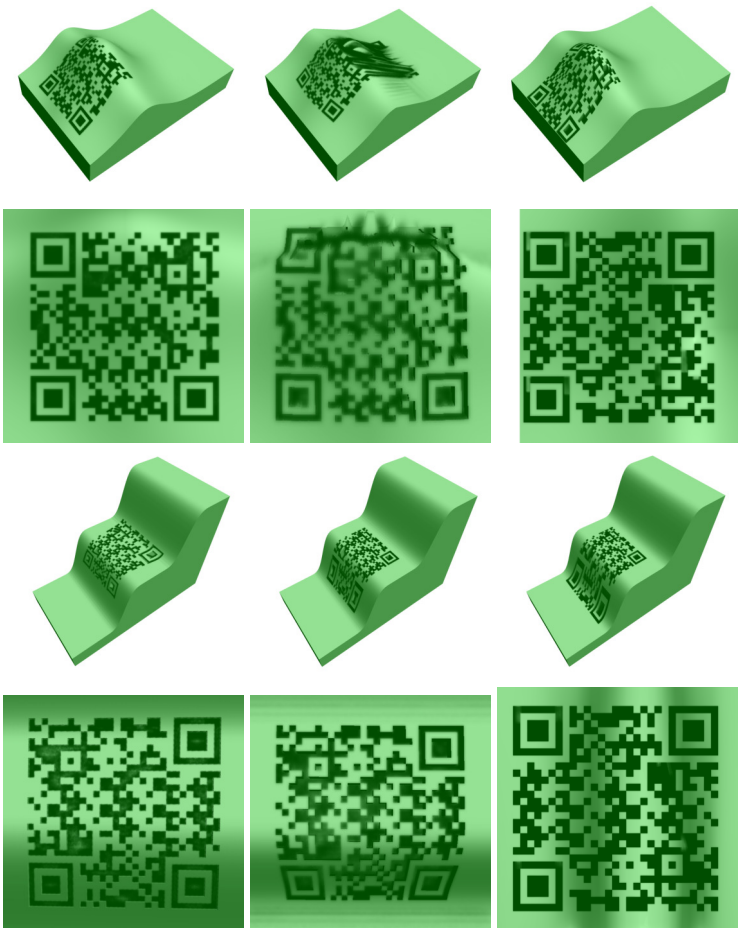
A.2. Kapcsolatok vizualizációja merőlegesen metsző körökkel



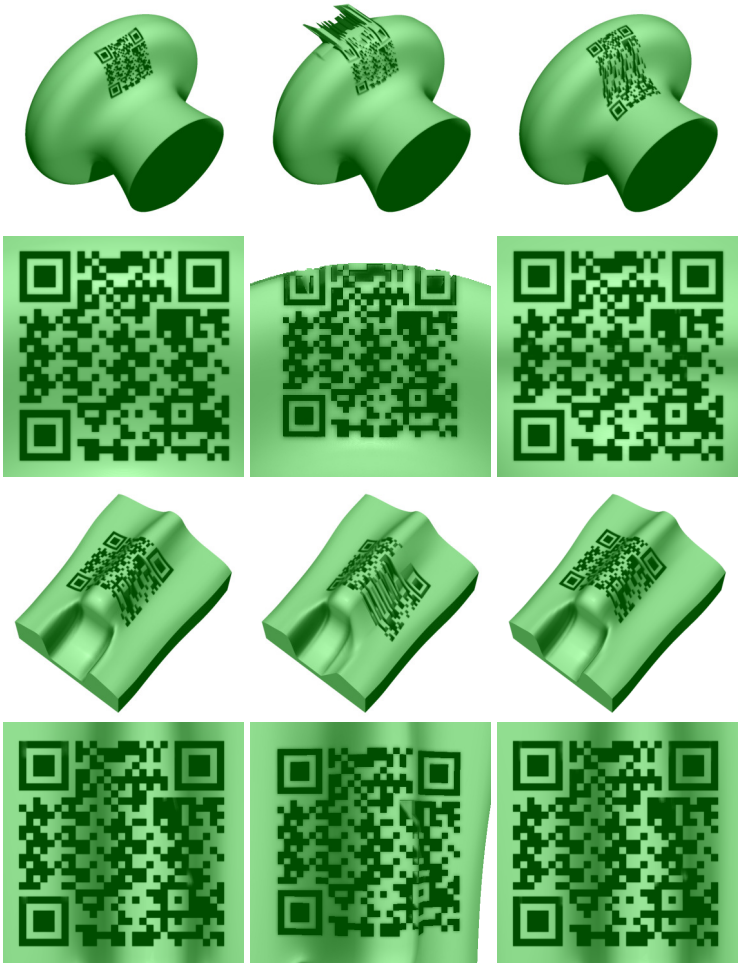
A.1. ábra. Kör (bal) és Bézier görbe (jobb) használatával alkotott kapcsolatok összehasonlítása genomi adatok vizualizációjában.

B. Felületbe ágyazott QR-kód

B.1. Paraméteres felület

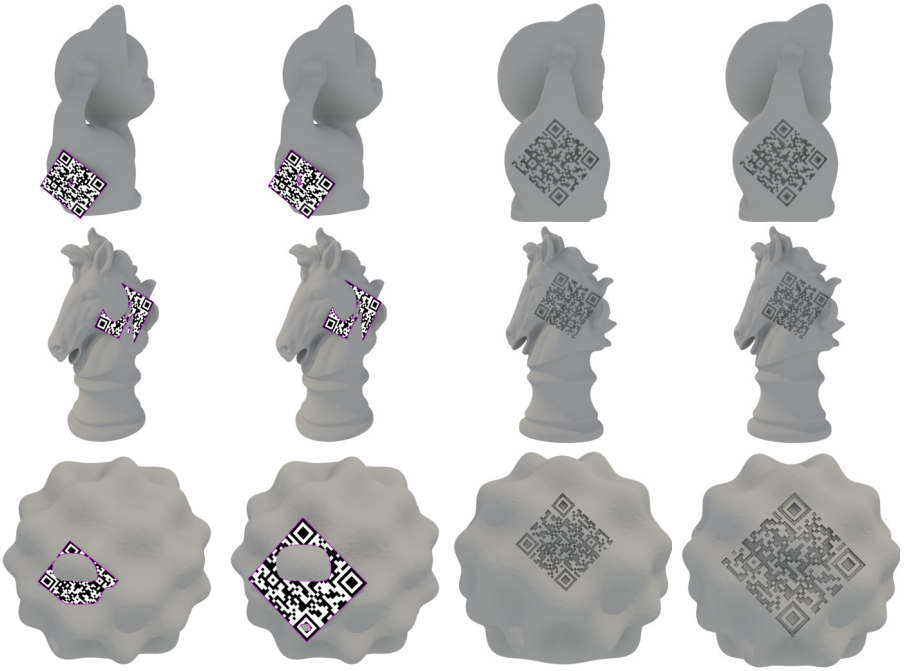


B.1. ábra. A módszerünk (1. oszlop) összehasonlítása a [26] (2. oszlop) és [58] (3. oszlop) módszerekkel. A 3.1. ábrán látható első két felületet használtuk a QR-kódok beágyazásához.

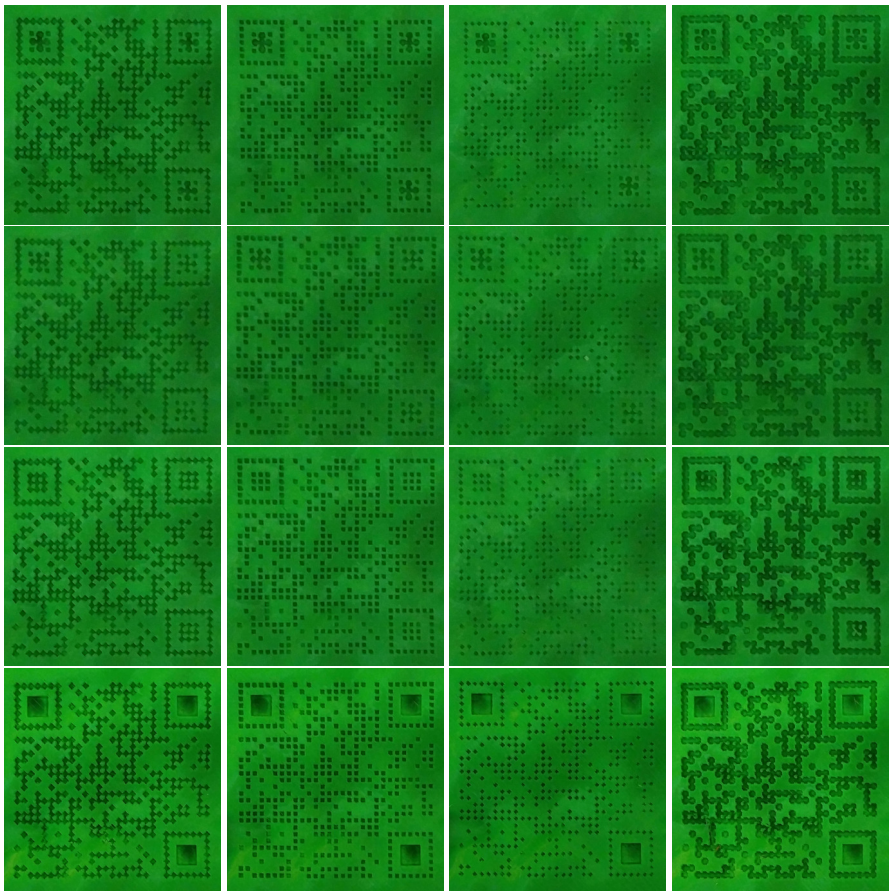


B.2. ábra. A módszerünk (1. oszlop) összehasonlítása a [26] (2. oszlop) és [58] (3. oszlop) módszerekkel. A 3.1. ábrán látható utolsó két problémás felületet használtuk a QR-kódok beágyazásához.

B.2. Háromszögelt modellek



B.3. ábra. A 3.25. ábrán bemutatott összehasonlítás elvégzése további modelleken: Macska, Huszár, és Göröngyös gömb.



B.4. ábra. A javasolt mintáink összehasonlítására, 3D nyomtatóval készített „David” fejmodellbe ágyazott QR-kódok. Minden sorban különböző modul mintát használtunk, hasonlóan ahhoz, ahogy minden oszlopban eltérő lokalizációs mintával ágyaztuk be a QR-kódot.

C. Publikációs lista

Referált folyóiratcikkek

[J1] György Papp – Miklós Hoffmann – Ildikó Papp: Embedding QR Code onto Triangulated Meshes using Horizon Based Ambient Occlusion. *Computer Graphics Forum*, 2021.

[J2] György Papp – Miklós Hoffmann – Ildikó Papp: Improved Embedding of QR Codes onto Surfaces to be 3D Printed. *Computer-Aided Design*, 131. évf. (2021), 102961. p.

Kéziratok

[M1] György Papp – Ildikó Papp – Roland Kunkli: Construction and properties of conics in relationship visualization. 2022. Kézirat.

Konferenicakötetben megjelent cikkek

[I1] György Papp – Roland Kunkli: TabularVis: An Interactive Relationship Visualization Tool Supported by Optimization and Search Algorithms. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics* (konferenciaanyag). 2018, Springer, 167–192. p.

[I2] György Papp – Miklós Hoffmann – Ildikó Papp: Embedding QR code onto triangulated meshes. In *The 12th Conference of PhD Students in Computer Science - Volume of short papers* (konferenciaanyag). 2020, 79–82. p.

[I3] György Papp – Roland Kunkli: TabularVis – a Circos-inspired interactive web client based tool for improving the clarity of tabular data

visualization. In *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3: IVAPP,* (konferenciaanyag). 2018, INSTICC, SciTePress, 120–131. p. ISBN 978-989-758-289-9.

További konferencia-előadások

[C1] György Papp – Miklós Hoffmann – Ildikó Papp: Embedding QR code onto triangulated meshes. In *The 12th Conference of PhD Students in Computer Science - Volume of short papers* (konferenciaanyag). 2020, 79–82. p.

[C2] György Papp – Miklós Hoffmann – Ildikó Papp: Improved QR code embedding for meshes with 3D printing. In *ICAI 2020* (konferenciaanyag). 2020.

[C3] György Papp – Roland Imre Kunkli: Table visualization with improved circular layout technique. In *ICAI 2017* (konferenciaanyag). 2017.

[C4] György Papp – Ildikó Papp – Roland Imre Kunkli: Construction and Properties of Orthogonally Intersecting Conics in Relationship Visualization. In *Conference on Geometry: Theory and Applications; University of Innsbruck, June 3–7, 2019* (konferenciaanyag). 2019, 27–28. p.

Demo absztrakt

[D1] György Papp – Ildikó Papp – Roland Imre Kunkli: Three-dimensional connection visualization based on tabular data. In *2017 8th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)* (konferenciaanyag). 2017, 289–290. p.

Poszterpretentációk

[P1] György Papp – Miklós Hoffmann – Ildikó Papp: Improved projection for embedding QR code onto surfaces to be 3D printed. Graphics and Application The 12th Asian Forum on Graphic Science (AFGS 2019), 2019.

[P2] György Papp – Roland Imre Kunkli: Connecting genomic regions with circular arcs. VIZBI 2018 - Visualizing Biological Data, Posters | VIZBI (https://vizbi.org/Posters/2018/), 2018.

[P3] György Papp – Roland Imre Kunkli: Relationship visualization of tabular data based on hyperbolic geometry. 5th Winter School of PhD Students in Informatics and Mathematics, 2018.

[P4] György Papp – Ildikó Papp – Roland Imre Kunkli: Possibilities of Using Orthogonally Intersecting Conics in Circular Layout Based Relationship Visualization. 6th Winter School of PhD Students in Informatics and Mathematics, 2019.